Investigation of Al_2O_3 Nanoparticle Laden Lubricants and Refrigerant Mixtures During Two-Phase Flow Boiling

by

Andrea A. M. Bigi

A dissertation submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Doctor of Philosophy

Auburn, Alabama
May 5, 2018

Keywords: nanofluid, nanolubricant, two-phase flow

Approved by

Lorenzo Cremaschi, Chair, Associate Professor of Mechanical Engineering
Sushil Bhavnani, Associate Department Chair of Mechanical Engineering
Robert Jackson, Professor of Mechanical Engineering
German Mills, Professor of Chemistry and Biochemistry

Abstract

In space conditioning and in cooling systems for high power density electronics, vapor compression cycles provide cooling. The working fluid is a refrigerant and oil mixture. A small amount of oil is needed to lubricate and to seal the sliding parts inside the compressors but, when mixed with refrigerant and carried through the system and in heat exchangers, the lubricant in excess penalizes the heat transfer coefficient and increases the flow losses: both effects are highly undesired yet unavoidable.

Nanolubricants - a lubricant with dispersed nano-size particles - can be a cost-effective technology to address this problem and for improving the efficiency and performances of vapor compression cycles. Several researchers postulated that the magnitude of the heat transfer enhancement due to the presence of nanoparticles is much higher than the gain in the liquid thermal conductivity and that the nano-scale interactions between the nanoparticles and the refrigerant/oil liquid layers are responsible for the heat transfer intensification.

This research aims at understanding the mechanisms responsible for heat transfer intensification during two-phase flow processes when nano-thermal vectors are used. This was achieved by providing a consistent set of experimental data that document the effects of the nanoparticles on the two-phase flow heat transfer coefficient and pressure drop, and by developing a theoretical model that captures the effects of the nanoparticles.

The effects of Al_2O_3 nanoparticles on the thermophysical properties of a polyolester lubricant (POE) were measured at different nanoparticles mass concentrations (0%, 10%, and 20%) and with different surfactants, used as dispersants.

Tests on evaporative two-phase flow of mixtures of refrigerant R410A and Al_2O_3 nanolubricant were reported for different nanoparticles mass concentrations (0%, 0.05%, 1%, and 3%), and it was observed that the two-phase flow heat transfer coefficient was either enhanced or degraded depending on oil and nanoparticle concentrations, and on the mass flux. Interestingly, pressure drop did not seem to be affected by the presence of nanoparticles.

A simulation tool based on correlations and theoretical models was developed to describe the nanoparticle distribution and behavior within the liquid film. The model provided a platform for future investigations into the behavior of high viscosity nanoparticle suspensions.

To my father, who taught me to always aim for the top with passion.

To my mother, who taught me to be prudent at every step.

Acknowledgments

At the end of this adventure, I want to express my gratitude first of all to my adviser Dr. Lorenzo Cremaschi, who has been an example of dedication and passion both in his work and in his life. I am grateful for the opportunity to work with him, walking along roads I could not imagine...even the ones that take you from Oklahoma to Alabama! Many are the "technical things" that I learnt under his guidance, and many more are the "personal development" lessons that taught me to push myself beyond what I thought I was capable of.

I would also like to thank all the Professors I met and worked with during these years; especially those on my committee at Auburn University, and those I met at Oklahoma State University. I am thankful for the time they dedicated to my research, and much more for creating an encouraging and collaborative environment that I always felt I could rely on. I want to express my gratitude also to many other university personnel and technicians, and in particular to Gary Don Thacker for his generosity and the great help he gave in the lab.

I could not forget to mention my colleagues with whom I shared joys and tears: Pratik Deokar whose presence and friendship were of incredible value, Thiam Wong, Pedro Perez, Stefano DellOrto, Carlo Andres. I also want to thank other lab members: Ellyn Jesperson, Jeremy Smith, Sarath Mulugurthi, Shanshan Cai, Weiwei Zu, Xiaoxiao Wu.

Many more are the people and friends I met both in Oklahoma and in Alabama, on-campus and off-campus, and who have been of great support. I could not think of my Ph.D. without remembering them.

Finally, I want to express all my gratitude to my parents and my siblings, and to all my Italian family and friends. They always motivated me to be the better version of myself, and their example, love and support is at the very core of this work.

Table of Contents

Abstract ii

Acknowledgments v

List of Abbreviations xvii

1 Introduction 1

1.1 Problem Statement 2

2 Motivation and Objectives 4

3 Literature Review 6

3.1 Use of Nanofluids 6

3.2 Use of Additives or Surfactants 8

3.2.1 Nanoparticle Stability 9

3.3 Nanolubricant-Refrigerant Mixture Thermophysical Properties 10

3.3.1 Nanoparticle Sedimentation and Agglomeration in Clusters 10

3.3.2 Thermophysical Properties 10

3.4 Nanolubricant-Refrigerant Pool Boiling 13

3.4.1 Nanorefrigerant Pool Boiling 13

3.4.2 Lubricant-Refrigerant Pool Boiling 14

3.4.3 Nanolubricant-Refrigerant Pool Boiling on a Smooth Surface 15

3.4.4 Nanolubricant-Refrigerant Pool Boiling on a Finned Surface 17

3.5 Nanolubricant-Refrigerant Flow Boiling 19

3.5.1 Nanofluid Convective Vaporization 20

3.5.2 Lubricant-Refrigerant Convective Vaporization 25

4 Experimental Work 28

4.1 Measurement of Thermophysical Properties and Two-Phase Flow Performances 28

4.1.1 Equipment and Instrumentation 28

4.1.2 Equipment for Mixing the Nanoparticles in the POE Lubricant 28

4.1.3 Equipment for Measuring the Nanoparticle Sizes in Dispersion in POE Lubricant 29

4.1.4 Equipment for Measuring the Specific Heat of Nanolubricants 29

4.1.5 Equipment for Measuring the Solubility of Refrigerant R410A in Nanolubricants 30

4.1.6 Equipment for Measuring the Thermal Conductivity of Nanolubricants . 31

4.1.7 Equipment for Measuring Two-Phase Flow Performances 31

4.2 Experimental Methodology 33

4.2.1 Procedure to Measure the Potential of Nanoparticle Sedimentation and Agglomeration 33

4.2.2 Procedure to Measure the Specific Heat of the Nanolubricants 34

4.2.3 Procedure to Measure the Solubility of Refrigerant R410A in the Nanolubricants 35

4.2.4 Procedure to Measure the Thermal Conductivity of the Nanolubricants . 36

4.2.5 Procedure to Measure Two-Phase Flow Performances 36

4.3 Calibration, Validation, and Uncertainty Analysis 38

4.4 Results and Discussion 40

4.4.1 Sedimentation Test Results 41

4.4.2 Specific Heat Test Results 43

4.4.3 Solubility Test Results 44

4.4.4 Thermal Conductivity Test Results 46

4.4.5 Viscosity Test Results . . . 47
4.4.6 Miscibility Test Results . . . 47
4.4.7 Two-Phase Flow Performances . . . 48

5 Simulation Work . . . 52
5.1 Segment-by-Segment Model . . . 52
5.1.1 Lubricant Properties . . . 53
5.1.2 Refrigerant and Lubricant Mixture Properties . . . 55
5.1.3 Nanoparticle Properties . . . 59
5.1.4 Nanolubricant Properties . . . 59
5.1.5 Refrigerant and Nanolubricant Mixture Properties . . . 62
5.1.6 Segment Inventory . . . 64
5.2 Segment-by-Segment Model Validation . . . 65
5.2.1 Experimental Validation of the Pressure Drop Models . . . 65
5.2.2 Experimental Validation of the Heat Transfer Coefficient Models . . . 67
5.2.3 Simulation Results and Discussion . . . 69
5.2.4 Discussion of the Simulation Results for Two-Phase Flow Pressure Drop of Nanolubricants . . . 71
5.2.5 Discussion of the Simulation Results for Two-Phase Flow Heat Transfer Coefficients of Nanolubricants . . . 74
5.3 Fundamental Approach . . . 75
5.3.1 Analysis of Slip Mechanisms . . . 76
5.3.2 Continuum Assumption . . . 77
5.3.3 Particle Rotation and Translation . . . 79
5.3.4 Inertia . . . 82
5.3.5 Brownian Diffusion . . . 86
5.3.6 Thermophoresis . . . 87

5.3.7 Diffusiophoresis . . . 89
5.3.8 Magnus Effect . . . 89
5.3.9 Gravity . . . 92
5.3.10 Drainage . . . 92
5.3.11 Wall Lubrication . . . 93
5.3.12 Conclusions . . . 93
5.4 Correlation Development . . . 94
5.4.1 Single Phase Radial Analysis and Buongiorno Model . . . 95
5.4.2 Two-Phase Radial Analysis and Radermacher-Cremaschi Model . . . 98
5.4.3 Friction Factor of Microfins . . . 101
5.4.4 Superposition Model . . . 102

6 Results and Discussion . . . 106
6.1 Buongiorno Validation . . . 106
6.2 Comparison with Literature Correlations . . . 108
6.3 N_{BT} Sensitivity Analysis . . . 110
6.3.1 Single Phase Radial Analysis . . . 112
6.3.2 Two-Phase Radial Analysis . . . 121
6.3.3 Convective Heat Transfer Analysis . . . 134
6.4 Effect of Microfins . . . 139
6.5 Correlation for Two-Phase Heat Transfer . . . 140
6.5.1 Approach for Correlation Development . . . 141

7 Conclusions and Recommendations . . . 146
7.1 Conclusions of the Experimental Work . . . 146
7.2 Conclusions of the Simulation Work . . . 148
7.3 Recommendations for Future Work . . . 150

References 152

Appendices 166

A Brief Review on Correlations for Convective Vaporization 167

A.1 Procedure to Estimate the Coefficients S and F 169

A.1.1 Chen Correlation 169

A.1.2 Sawant Correlation 170

B Two-Phase Flow Experimental Data Set 172

C Code Script 183

List of Figures

1.1 Assumed behavior of nanoparticles creating nanoscale convection. 3

3.1 Schematic of the lubricant excess layer, l_e, and bubble departure at different lubricant mass fraction, x_b (figure from Kedzierski (2003)). 15

3.2 Wall region representation (figure adapted from Buongiorno (2006)). 23

3.3 Force balance on the control volume. 26

4.1 Experimental setups for measuring specific heat of nanolubricants. 30

4.2 Experimental setups for measuring solubility of nanolubricants. 31

4.3 (a) Schematic of the test apparatus, (b) detailed schematic of the test section, and (c) details of the location of the surface thermocouples (Figure from Deokar et al. (2016)). 32

4.4 Sedimentation test results for three types of nanolubricants. 41

4.5 Visual observation of sedimentation for type 3 nanolubricant. 42

4.6 Number-weighted average distribution of Al_2O_3 nanoparticle sizes measured with DLS. 43

4.7 Specific heat vs. Temperature of POE. 43

4.8 Specific heat ratio vs. Temperature of nanolubricants. 44

4.9 Pressure vs. wt.% R410A+T1S20. 45

4.10 Pressure vs. wt.% R410A+T2S20. 45

4.11 Thermal conductivity vs. Temperature of POE. 46

4.12 Nanolubricant-POE thermal conductivity ratio. 46

4.13 (a)HTF and (b)PDF of various concentration of Al_2O_3 based nanolubricant in R410A at $\dot{m}$ = 350 kg/m^2s and q" = 12 kW/m^2, at 1 wt.% and 3 wt.% OMF. . 48

4.14 (a)HTF and (b)PDF of 20 wt.% Al_2O_3 based nanolubricant in R410A at $\dot{m}$ of 183, 255, 350, 425 kg/m^2s and q" of 12 kW/m^2 and 15 kW/m^2, at 3 wt.% OMF. 50

5.1 Comparison of predicted pressure drops (ΔP) vs. experimental data. 67

5.2 Comparison of predicted heat transfer coefficients (HTC) vs. experimental data. 68

5.3 Experimental and simulation trends of different refrigerant/lubricant mixtures for pressure drop. 69

5.4 Experimental and simulation trends of different refrigerant/lubricant mixtures for heat transfer coefficient. 70

5.5 Pressure drop for (a) low quality, (b) medium quality and (c) high quality of different refrigerant-lubricant mixtures at test conditions of 250 kg/m^2s and 12 W/m^2 (the simulation data in this figure were obtained from application of Choi et al. (1999) correlation) 73

5.6 Heat transfer coefficient for (a) low quality, (b) medium quality and (c) high quality of different refrigerant-lubricant mixtures at test conditions of 250 $\frac{kg}{m2-s}$ and 12 $\frac{W}{m2}$ (the simulation data in this figure were obtained from application of Hamilton et al. (2008) correlation) 75

5.7 Schematic representation of the segmentation of the laminar sublayer in a single phase flow ($n = 3$). 97

5.8 Schematic representation of the segmentation of the liquid film in a two-phase flow (in this schematic, the sublayer is assumed to be all laminar and $n = 3$). . . 100

6.1 Validation of the correct implementation of Buongiorno's model at (a) ϕ = 0, (b) ϕ = 0.01, and (c) ϕ = 0.03. 107

6.2 Verification of Buongiorno's model against extrapolated single phase data. . . . 108

6.3 Comparison of the experimental data with the correlation by Sawant (2012). . . 109

6.4 Comparison of the experimental data with the correlation by Chen (1966). . . . 109

6.5 Comparison of the experimental data with the correlation by Gungor and Winterton (1986). 110

6.6 Distribution of nanoparticle concentration within the laminar sublayer as a function of N_{BT} (adapted from Buongiorno (2006)). 111

6.7 Thermophysical properties ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) within the laminar sublayer as a function of N_{BT}, for the case of a R410A based nanofluid (ϕ_b = 0.01). 113

6.8 Thermophysical properties ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) within the laminar sublayer as a function of N_{BT}, for the case of a R410A-lubricant based nanofluid (OMF = 1%, NMF = 20%) . . . 114

6.9 Thermophysical properties ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) within the laminar sublayer as a function of N_{BT}, for the case of a R410A-lubricant based nanofluid (OMF = 3%, NMF = 20%) . . . 114

6.10 Conventional velocity profile of a single phase fluid inside a tube. 116

6.11 (a) Velocity profile obtained from radial analysis in case of presence of nanoparticles and (b) zoom into the laminar region. 117

6.12 R410A nanofluid velocity profile variation at different concentrations ((a) $\phi = 0.01$, (b) $\phi = 0.03$) and at different values of N_{BT} ((c) $N_{BT} = 175$, (d) $N_{BT} = 0.0005381$). 118

6.13 1% OMF nanolubricant-refrigerant velocity profile variation at different concentrations ((a) $\phi = 0.00065$, (b) $\phi = 0.00128$) and at different values of N_{BT} ((c) $N_{BT} = 436.8$, (d) $N_{BT} = 0.000437$). 119

6.14 3% OMF nanolubricant-refrigerant velocity profile variation at different concentrations ((a) $\phi = 0.0019$, (b) $\phi = 0.0038$) and at different values of N_{BT} ((c) $N_{BT} = 296.3$, (d) $N_{BT} = 0.000297$). 120

6.15 10% OMF nanolubricant-refrigerant velocity profile variation at $\phi = 0.016$ and $N_{BT} = 0.075$. 121

6.16 Thermophysical properties of a R410A based nanofluid ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) calculated at different volume concentrations (ϕ = 0, 0.01, and 0.03), at x = 0.2, and 250 kg/m^2s mass flux. 122

6.17 Thermophysical properties of a R410A based nanofluid ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) calculated at different volume concentrations (ϕ = 0, 0.01, and 0.03), at x = 0.8, and 250 kg/m^2s mass flux. 123

6.18 Thermophysical properties of a R410A based nanofluid ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) calculated at different volume concentrations (ϕ = 0, 0.01, and 0.03), at x = 0.2, and 373 kg/m^2s mass flux. 125

6.19 Thermophysical properties of a R410A based nanofluid ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) calculated at different volume concentrations (ϕ = 0, 0.01, and 0.03), at x = 0.8, and 373 kg/m^2s mass flux. 126

6.20 Thermophysical properties of a R410A based nanofluid ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) calculated for OMF = 1%, at different mass concentrations (NMF = 0, 0.2, and 0.7), at x = 0.2, and 250 kg/m^2s mass flux. 127

6.21 Thermophysical properties of a R410A based nanofluid ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) calculated for OMF = 1%, at different mass concentrations (NMF = 0, 0.2, and 0.7), at x = 0.8, and 250 kg/m^2s mass flux. 128

6.22 Thermophysical properties of a R410A based nanofluid ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) calculated for OMF = 3%, at different mass concentrations (NMF = 0, 0.2, and 0.7), at x = 0.2, and 250 kg/m^2s mass flux. 129

6.23 Thermophysical properties of a R410A based nanofluid ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) calculated for OMF = 3%, at different mass concentrations (NMF = 0, 0.2, and 0.7), at x = 0.8, and 250 kg/m^2s mass flux. 130

6.24 Velocity profiles of a R410A based nanofluid for different N_{BT} ranges ((a) and (b): $1.35 < N_{BT} < 35$; (c) and (d): $0.13 < N_{BT} < 3.5$, (e) and (f):$0.0013 < N_{BT} < 0.035$), calculated at different volume concentrations (ϕ = 0, 0.01, and 0.03), at x = 0,2 and 0.8, and 250 kg/m^2s mass flux. 131

6.25 Velocity profiles of a R410A based nanofluid for different N_{BT} ranges ((a) and (b): $1.35 < N_{BT} < 35$; (c) and (d): $0.13 < N_{BT} < 3.5$, (e) and (f):$0.0013 < N_{BT} < 0.035$), calculated at different volume concentrations (ϕ = 0, 0.01, and 0.03), at x = 0.2 and 0.8, and 373 kg/m^2s mass flux. 133

6.26 Velocity profiles of a 250 kg/m^2s mass flux two-phase flow of R410A-oil mixture at different oil concentrations ((a) and (b): 1% OMF, $0.1 < N_{BT} < 0.25$; (c) and (d): 3% OMF, $0.03 < N_{BT} < 0.13$), calculated at different nanoparticle mass concentrations (NMF = 0, 0.2, and 0.7), at x = 0.2 and x = 0.8 134

6.27 Single phase R410A based nanofluid: Nu vs. Re ((a), (c) and (e)), and Pr_b/Pr_v vs. Re ((b), (d) and (f)) for different nanoparticles concentrations and for increasing values of N_{BT}. 136

6.28 Single phase R410A and 3% oil: Nu vs. Re for a refrigerant-nanolubricant mixture at different nanoparticles concentrations. 138

6.29 Effect of the use of a friction factor correlation for finned tubes on the prediction of the heat transfer coefficient. 139

6.30 Trend of liquid film thickness for simulation tests at 425 kg/m^2s mass flux, at different values of OMF. 140

6.31 Correlation predictions for (a) OMF = 0, (b) OMF = 1%, and (3) OMF = 3 %, using Chen method. 143

6.32 Correlation predictions for (a) OMF = 0, (b) OMF = 1%, and (3) OMF = 3 %, using Sawant method. 144

6.33 Correlation predictions for (a) OMF = 0, (b) OMF = 1%, and (3) OMF = 3 %, using a modified Sawant method. 145

A.1 Superposition model (figure adapted from Webb and Gupte (1992)). 168

List of Tables

4.1 Geometry details of the internally enhanced tube. 33

4.2 Test conditions of the two-phase flow measurements. 38

4.3 Maximum uncertainty from experiments on thermophysical properties. 39

4.4 Experimental uncertainties and repeatability of two-phase flow measurements. . 40

5.1 Sensitivity analysis for kinematic viscosity and thermal conductivity. 71

5.2 Fluid thermophysical properties for different sets of operational conditions (at NMF = 20%, $T_{sat} = 4^{\circ}C$, q" = 12 kW/m^2). 77

5.3 Fluids chemical characterization. 78

5.4 Carbon-to-element bond length (Weast, 1984). 79

5.5 Estimation of the Knudsen number. 79

5.6 Peclet number estimates. 81

5.7 Estimation of the translational velocity and corresponding diffusion time. . . . 82

5.8 Liquid film Reynolds number, superficial velocity, and thickness. 84

5.9 Estimation of relaxation time, flow velocity, and stopping distance. 84

5.10 Length and time scales of small turbulent eddies. 85

5.11 Brownian diffusion coefficient and corresponding diffusion time. 87

5.12 Thermophoretic velocity and corresponding diffusion time. 89

5.13 Gravity velocity and corresponding diffusion time. 92

List of Abbreviations

English Symbols

$\dot{m}$	Mass Flow Rate [$\frac{kg}{s}$]
A_S	Surface Area [m^2]
Al_2O_3	Aluminium Oxide
c_p	Specific Heat [$\frac{kJ}{kg-C}$]
D_B	Brownian Diffusion Coefficient [$\frac{m^2}{s}$]
D_T	Thermophoresis Diffusion Coefficient [$\frac{m^2}{s}$]
G_{flux}	Mass Flux [$\frac{kg}{m^2 s}$]
h_{LV}	Enthalpy of Vaporization [$\frac{J}{kg}$]
M_W	Molecular Weight [$\frac{g}{mol}$]
N_A	Avogadro Number
N_{BT}	Ratio Between Brownian and Thermophoretic Diffusivities
S_p	Particle Stopping Distance [m]
$wt.\%$	Weight Percentage [$\%$]
X_{tt}	Martinelli Parameter, turbulent-turbulent
D	Diameter [m]

F	Two-Phase Convection Multiplier [-], or Force [N]
f	Friction Factor
g	Gravitational Acceleration [$\frac{m}{s^2}$]
h	Enthalpy [$\frac{kJ}{kg}$]
HTF	Heat Transfer Factor
j	Mass Flux [$\frac{kg}{m^2 s}$]
k	Conductivity [$\frac{W}{m-C}$]
Kn	Knudsen Number
L	Length [m]
M, m	Mass [kg]
MM	Molecular Mass [$\frac{g}{mol}$]
N	Quantity Number
NMF	Nanoparticle Mass Fraction in Oil
Nu	Nusselt Number
OMF	Oil Mass Fraction
P	Pressure [Pa]
PDF	Pressure Drop Factor
POE	Polyolester Oil
Pr	Prandtl Number
R, r	Radius [m]
Re	Reynolds Number

S Suppression Factor

T Temperature [°C]

t Time [s]

T1S10 Type 1 nanolubricant. 10% by weight Al2O3 in POE oil

T1S20 Type 1 nanolubricant. 20% by weight Al2O3 in POE oil

T2S10 Type 2 nanolubricant. 10% by weight Al2O3 in POE oil

T2S20 Type 2 nanolubricant. 20% by weight Al2O3 in POE oil

v, u Velocity [$\frac{m}{s}$]

Vol Volume [m^3]

x Thermodynamic Quality

Greek Symbols

α Void Fraction [-], or Heat Transfer Coefficient [kW/m^2C]

β Thermophoretic Coefficient

δ Thickness [m]

δ_v^+ Dimensionless Thickness

ϵ_H Eddy Diffusivity for Heat [$\frac{m^2}{s}$]

ϵ_M Eddy Diffusivity for Momentum [$\frac{m^2}{s}$]

ϵ_p Eddy Diffusivity for Particles [$\frac{m^2}{s}$]

γ Fin Helix Angle [°]

λ Molecule Mean Free Path [m]

μ Dynamic Viscosity [$Pa \cdot s$]

ν	Kinematic Viscosity [$\frac{mm^2}{s}$]
ω	Mass Fraction
ϕ	Nanoparticle Volume Fraction
ψ	Modified Mole Fraction
ρ	Density [$\frac{kg}{m^3}$]
σ	Surface Tension [$\frac{N}{m}$]
τ	Shear Stress [Pa]
τ_p	Particle Relaxation Time [s]
ξ	Mole Fraction

Subscripts

H_2O	water
b	bulk
bub	bubble
e	equivalent
exp	experimental
f	fluid
g	gas, vapor
h	hydraulic
i	interface
L	liquid phase
LV	latent heat of vaporization

mix mixture

nanomix mixture of nanolubricant and refrigerant

nb nucleate boiling

nl nanofluid or nanolubricant

np nanoparticle

o oil

p particle

ref refrigerant

root fin root

s surfactant

sat saturation

seg segment

tp two-phase

V vapor phase

v laminar sublayer

vt laminar-turbulent interface

w wall

Chapter 1

Introduction

The International Energy Outlook (Briefing, 2013) released in 2013 by the United States Energy Information Administration (EIA) projected an increase in the world energy consumption by 56 percent between 2010 and 2040. As reported by the Department Of Energy (DOE) (Conti et al., 2016), to date, in the United States, commercial space conditioning (heating, cooling and ventilation) and refrigeration use respectively roughly 7.0 and 1.3 quads per year of primary energy that correspond to 40 and 7 percent of the total commercial use. Furthermore, the residential sector uses 56 percent of the total energy demand in heating and cooling. In this context, the need for improvements in system efficiency and energy saving pushed the research towards new non-traditional approaches.

Thanks to the development of new technologies, the manufacturing and production of nanometer powders of metal was made easier and more affordable, opening scientists to the possibility of the use of nanofluids. Nanofluids are commonly defined as fluids containing a colloidal solution of nanometer-sized particles and the study of their application is wide today and not limited to energy-related fields.

Since the late 1990s, researchers dedicated their work to the observation of the behavior of nanofluids for applications on thermal systems. Commonly used fluids with well-known properties (e.g. water, ethylene glycol) were first adopted to be the base fluid for the nanoparticle dispersion. Particular attention was then given to the variation in thermophysical properties and heat transfer capabilities with the change in parameters or variables such as particle material, size, shape, concentration and dispersion. General mechanisms describing the particles behavior were also observed as they could locally or systematically affect the fluid properties.

Several studies reported experimental results on both nanofluids' thermophysical properties (mainly thermal conductivity, viscosity, specific heat and density) and on pool and flow boiling tests and most researchers agreed on general observations regarding the overall increase of nanofluids' thermal conductivity, viscosity and heat transfer. However there is still a lack of agreement on some of the results and the attempts to model the experimental observations were not always successful or generally applicable; for these reasons, different authors provided frequent overviews in order to keep track of the research state-of-the-art.

1.1 Problem Statement

In air conditioning and refrigeration, the vapor compression cycle generally consists of two heat exchangers (e.g. plate, fin-and-tube, microchannel, etc.), one expansion valve and a compressor. Oil is necessary inside the compressor case for the lubrication and sealing of mechanical parts and for this reason, it comes in contact with the refrigerant. However, while most of the oil is separated and stays in the compressor, a small portion of the oil circulates with the refrigerant flow through the cycle components. The circulating oil can form a fairly homogeneous mixture with the liquid refrigerant but, depending on the oil concentration, the heat exchanger geometry and the operational conditions, it penalizes the heat transfer coefficient and increases the flow losses: both effects are highly undesired yet unavoidable (Cremaschi et al., 2005a). The excess lubricant-rich film resides in a layer on the surface and it affects the heat transfer performance, giving either an enhancement or degradation (Kedzierski, 2003).

Kedzierski (2002) showed that the lubricant excess layer causes an average enhancement of the heat flux of approximately 24% for a 0.5% lubricant mass fraction mixture relative to pure R134a during pool boiling. However, at 1% and 2% lubricant mass fraction of the mixtures, an average degradation of approximately 60% in the heat flux relative to pure R134a was observed. Negative effects of lubricants on the flow boiling heat transfer in heat exchangers were also experimentally observed by Zhao et al. (2002). Therefore, the overall performances of the system are negatively affected. As a new research frontier, nano-thermal vectors are suitable to address this problem and bring major improvements to the heat transfer processes in space-conditioning, refrigeration, the transportation sector and electronic cooling Cheng et al. (2008);

Choi (2009). Nanolubricants - nano-size particle laden lubricants - have the potential to be a cost-effective technology for reducing the energy consumption because of their unprecedented thermal transport phenomena (Choi, 2009). It is assumed that nanoparticles relocate close to the gas-liquid interface driven by the refrigerant that boils off from the liquid mixture. At the interface they are expected to tumble on each other, as shown in Figure 1.1

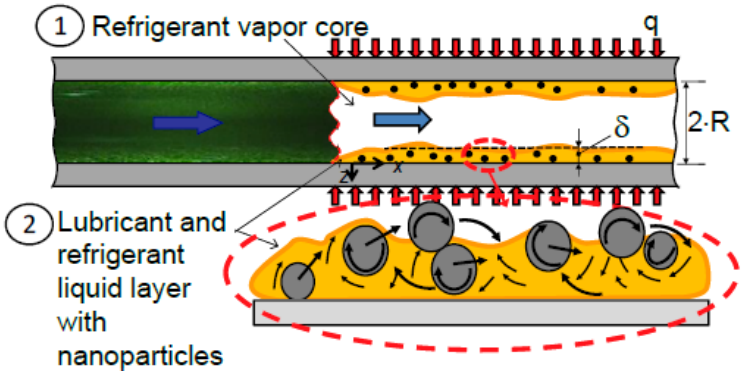


Figure 1.1: Assumed behavior of nanoparticles creating nanoscale convection.

The non-uniform viscosity gradients and shear rate in the lubricant tend to re-direct the nanoparticles toward the wall. Thus, a continuous micro-convective flow is established in the radial direction within the liquid mixture.

Chapter 2

Motivation and Objectives

Several researchers postulated that the magnitude of the heat transfer enhancement is much higher than the gain in the liquid thermal conductivity and that the nano-scale interactions between the nanoparticles and the refrigerant/oil liquid layers are responsible for the heat transfer intensification. Enhancements were observed in pool boiling (Wen and Ding, 2004; Peng et al., 2010; Kedzierski, 2009, 2011) and in experimental work for flow boiling in a horizontal tube (Bartelt et al., 2008). However, the study of nanolubricant two-phase flow boiling heat transfer is still in its infancy and there is controversy in the literature on several relevant matters: how can nano-thermal vector fluids be stabilized so that nanoparticles will not foul the heat transfer surfaces? In two-phase flow processes, do nanoparticles produce micro-convection within the liquid phase that effectively increases the heat transfer coefficient beyond their augmented thermal conductivity? How do nanoparticles distribute within the liquid layer? What is the enhancement level on the heat transfer coefficient and the effect on the corresponding pressure drop? How do they correlate with nanoparticle type and concentration? What dependency can be found from operational conditions such as mass flux, heat flux, saturation temperature, surface geometry? To date, these are open questions that this work aims at answering with a systematic experimental investigation and model description. This study focuses mainly on the use of nanoparticles of Alumina (Aluminium oxide, Al_2O_3) with a $40 \div 60$ nm nominal diameter and spherical shape. The nanoparticles were dispersed at different mass concentrations (0%, 10%, and 20%) in a common ester oil with density of 0.981 g/ml at 20°C and kinematic viscosity of 31.2 cSt and 5.6 cSt, respectively at 40°C and 100°C. The base fluid was refrigerant R410A and oil concentration ranged between 0% and 3%. The experimental operational

conditions chosen for mass flux (180 kg/m^2s to 425 kg/m^2s), heat flux (12 kW/m^2), and saturation temperature (4°C) were similar to those of a real case scenario.

In particular, these are the main objectives of this research:

- Characterize experimentally the thermophysical properties of Al_2O_3 nanolubricants.
- Observe and measure the behavior of mixtures of nanolubricant and refrigerant R410A during two-phase flow, while changing major variables such as:
  - nanoparticle concentration
  - lubricant concentration
  - saturation temperature
  - mass flow rate
- Develop a simulation model that:
  - verifies the validity of existing correlations to describe thermophysical properties of nanolubricants and nanolubricant-refrigerant mixtures
  - describes the behavior of nanolubricant-refrigerant mixtures during evaporative two-phase flow

Chapter 3

Literature Review

The worldwide growing energy demand and the increasing control on carbon dioxide emissions in the atmosphere have, in the last decades, strongly affected the ways energy is managed, from production to consumption. In this context, the need for improvements in system efficiency and energy saving pushed the research into new non-traditional areas of applications. The development of new technologies facilitated the manufacturing and production of nanometer powders of metal, making the use of nanofluids easier and more affordable. Nanofluids are commonly defined as fluids containing a colloidal solution of nanometer-sized particles and the study of their use and application is wide and not limited to energy-related fields (Taylor et al., 2013).

3.1 Use of Nanofluids

For many practical applications, it is fundamental to understand the behavior of fluids in forced convection. Nanofluids are multi-component fluids and they always present more than one phase coexisting at the same time; for these reasons they are generally classified as multi-phase fluids. Depending on the assumptions, multiphase flows are described in literature as either homogeneous fluids or heterogeneous mixtures. Under the assumption that nanoparticles are ”small enough” to be in thermodynamic equilibrium with the base fluid, nanofluids are sometimes treated as single-phase fluids and the slip and shear stress between particles is often neglected. However it was observed that phenomena that are peculiar of solid-fluid mixtures (such as particle Brownian motion, friction and sedimentation) might still be necessary to describe the behavior of nanofluids.

The increase in heat transfer could be the result of at least two predominant occurrences: not only the increase in thermal conductivity but also the fact that particles can act as thermal vectors moving randomly, causing an increase in the heat exchange by increasing the temperature difference between the tube wall and the bulk flow. Many review papers collected and summarized the experimental findings on both the pool boiling and the convective heat transfer of nanofluids (Kakac and Pramuanjaroenkij, 2009; Lotfi et al., 2010; Godson et al., 2010; Murshed et al., 2011; Sarkar, 2011). As oftentimes reported, the effects of nanofluids in heat transfer applications are sometimes conflicting and, for this reason, it was difficult to formulate a consistent model that was able to describe the behavior and predict the performances of the nanofluid.

With respect to pool boiling, the majority of researchers used water-based fluids with Alumina nanoparticles and the observed contradictory results were sometimes attributed to the different particles sizes and concentrations. After correction of the base fluid thermophysical properties in order to account for nanoparticle dispersion, the use of classical correlations such as those by Rohsenow and Zuber, was not enough to describe the experimental results. In convective heat transfer, the database of experimental results is not as large as for pool boiling. Nevertheless, interesting observations were made as it was shown that the heat transfer coefficient increased with particle concentration and with higher Peclet numbers.

Other research groups observed instead a deterioration of the heat transfer coefficient with an increase in particle concentration that could have caused an increase in viscosity or favored sedimentation and agglomeration into clusters. From a theoretical point of view, the measured enhancement of thermal conductivity was not enough to give reasons for the enhancement in heat transfer. Even accounting for Brownian motion and thermo- or diffusiophoresis, the available literature correlations were not able to predict the experimental results consistently. For these reasons, traditional expressions of the Nusselt number (Nu) such as the Dittus-Boelter equation, were modified to account for more variables (Sarkar, 2011) as described in Equation 3.1:

$$Nu = f\left(Re, Pr, \frac{k_p}{k_{bf}}, \frac{(\rho c_p)_p}{(\rho c_p)_{bf}}, \phi, D_p, particle\ shape, flow\ regime\right) \tag{3.1}$$

where the subscripts 'p' and 'bf' stand for particle and base fluid. Many correlations in the literature are based on Equation 3.1 but oftentimes they underpredicted the heat transfer experimental results. Further investigation led to the observation that Brownian diffusion and thermophoresis were predominant mechanisms in convective heat transfer. In fact, the thermophoresis pushes the nanoparticles far from the wall and flattens the profile of temperature between the wall and the fluid bulk. The Peclet number (Pe) gives a measure of the amount of convective heat transmitted to the fluid with respect to the conductive heat transported through the fluid. Some researchers observed that both an increase in particle size and in Reynolds number, caused an increase in the Peclet number. At smaller particle sizes, the axial distribution became more uniform across the pipe. Finally, lower viscosity permitted an increase in wall heat transfer by reducing the shear stresses.

When the nanofluid was treated as a heterogeneous mixture, researchers implemented models to solve the equations of conservation of mass, momentum and energy, but taking into consideration also the particles' diffusion and convection. The solution of these equations could be more detailed, depending on the specific boundary conditions, but generally it is computationally more costly and does not guarantee consistency. For these reasons, not very many models are available in literature.

3.2 Use of Additives or Surfactants

In the study of liquids' heat transfer, the use of surfactant additives (solutes) mixed with a base working fluid (solvent) showed promising results as it was observed that a small quantity of surfactant can greatly affect the boiling heat transfer (Hetsroni et al., 2001). Surfactants are commonly defined as chemical amphiphilic compounds, presenting both a water-soluble (hydrophilic) group and a water-insoluble (hydrophobic) group and, based on the kind of hydrophilic group, they are generally classified as anionic, cationic and nonionic.

The double nature of these compounds enables them to act at the interface between water and air, inducing a depression in the liquid surface tension. Depending on the surfactant type and concentration, the surface tension can be lowered asymptotically down to a minimum

critical value. Because of this critical value, the surfactant mass percentage in solution is generally small and the other thermophysical properties are generally not affected, except for the viscosity. An increase in the surfactant mass percentage would lead to an increase in viscosity, with effects on the behavior of the heat transfer. Several studies were conducted to observe the effects of surfactants (Peng et al., 2011) and the general outcomes showed an enhancement of the boiling CHF (Critical Heat Flux): because of the reduced surface tension, according to the Hsu model, the energy required to form bubbles at the heated surface was smaller, allowing the formation of more bubbles with smaller diameter.

3.2.1 Nanoparticle Stability

As observed by Wen and Ding (2004); Lin et al. (2016), one of the major concerns regarding the use of nanofluids is that nanoparticles need to be stabilized to avoid agglomeration and sedimentation. Different methods were investigated to stabilize the colloidal suspension (Ghadimi et al., 2011). Electrostatic repulsion relies on the use of Coulombic forces to keep nanoparticles from colliding, by electrically charging their surface. Chemical functionalization is a process that adds new properties to nanoparticles by changing their surface chemistry; a frequent example of this technique is the use of amphiphilic organic compounds (surfactants or stabilizers) to provide hindrance between nanoparticles by steric repulsion.

The main scope served by surfactants is therefore the prevention of nanoparticles agglomeration. However, according to what was mentioned with regards to additives, it is possible to speculate over the impact that the surfactants alone might have on the thermophysical properties of the base fluid. The use of surfactants was investigated for their specific utilization in refrigerant-based nanofluids, where thermophysical properties (e.g. viscosity, density) and chemical properties (e.g. dielectric constant, polarity) are different from other more widely studied fluids, such as water or ethylene glycol. A few studies are available where a screening of surfactants of different ionic nature (e.g. SDBS (anionic), CTAB (cationic), NP-10 (non-ionic)) was carried out to investigate the stability of nanoparticles of different diameters and concentrations in liquid refrigerant (Peng et al., 2011; Lin et al., 2015).

3.3 Nanolubricant-Refrigerant Mixture Thermophysical Properties

The literature offers many reviews on the status of nanofluids research (Angayarkanni and Philip, 2015). In the next section, some fundamental aspects regarding thermophysical properties of nanofluids are presented.

3.3.1 Nanoparticle Sedimentation and Agglomeration in Clusters

Two critical factors that must be characterized when developing nanolubricants for heat transfer enhancement are the potential for agglomeration of the nanoparticles into large clusters and for sedimentation of the nanoparticles on the heat transfer surfaces. The sedimentation due to clustering and agglomeration of nanoparticles was observed in nanofluids (Wen and Ding, 2004). Agglomeration and sedimentation of nanoparticles in the lubricant might interfere with the heat transfer process (Das et al., 2003). Most heat transfer surfaces have nucleation sites that enhance heat transfer due to eddies created by the nucleation sites (Cieliski and Targaski, 2007).

Sedimentation of nanoparticles that are immersed in the heat transfer fluid might deposit into the nucleation sites creating a smoother surface (Bang and Heung Chang, 2004). According to Das et al. (2003) the resulting smoother surfaces can cause a considerable deterioration of the heat transfer coefficient. From previous studies, it was observed that stable suspensions of nanoparticles had minimum sedimentation. To develop such stable suspensions, the base fluid had high viscosity such as the case with polyolester oils. The addition of dispersants and surfactants could prevent clustering and findings the correct combination required often a trial and error approach. In this approach the size of nanoparticles in suspensions is often measured by using dynamic light scattering (DLS), also referred to as quasi-elastic light scattering technique.

3.3.2 Thermophysical Properties

Abundant literature exists on refrigerant and lubricant mixture properties and on water based nanofluids and a review of these areas is beyond the scope of this work. Instead the emphasis

is on studies in the literature that focused on nanoparticles dispersed in high viscosity suspensions. At present, there is very limited information on the thermodynamic, thermal, and transport properties of nanoparticles in POE lubricants and studies on solubility and miscibility are missing in the open domain literature. The main properties investigated in this work and a summary of the associated studies in the literature are discussed next.

Specific Heat of Nanolubricants

Model for water based nanofluids are often used to predict the specific heat of nanolubricants but their accuracy was seldom verified. Nanofluids have lower specific heats than their base fluids, according to Equation 3.2 valid for an ideal liquid-particle mixture:

$$c_{p,nl} = \phi c_{p,p} + (1 - \phi) c_{p,fluid} \tag{3.2}$$

In several experiments, it was observed that the specific heat decreased if the volume concentration of nanoparticles, ϕ, increased. Specific heat also increased with increase in temperatures (Vajjha and Das, 2009). Experiments conducted by Murshed et al. (2008) used a double hot-wire technique to measure the effective specific heat of different types of nanofluids. Their study concluded that fluids with nanoparticles had lower specific heat than their base fluids, and that the values for specific heat decreased with increasing volume fraction of the nanoparticles. A thorough study was conducted by Clary and Mills (2011) in which CuO nanoparticles of about 9 nm in diameter were dispersed in hydrocarbons. The thermodynamic measurements showed a decrease in the solution specific heat, larger at higher concentrations and lower temperatures. Moreover, the observed increase in the solvent viscosity was considered to be casue of the decrease in convection within the liquid phase.

Puliti et al. (2011) presented a comprehensive review of available literature on nanofluids. For specific heat, most studies have reported that nanofluids have lower specific heats than their base fluids. However conflicting studies were also presented where the specific heat was higher than the base fluids. It was recommended to conduct more experiments for measuring the specific heat of nanofluids and for verifying the correlations.

Solubility and Miscibility of Refrigerant R410A with Nanolubricants

Solubility and miscibility of oil-refrigerant mixtures affects the density, viscosity, specific heat, and conductivity of the liquid phase of the mixture in the two phase region. Nanoparticles dispersed in POE oil with surfactants might alter the degree of solubility of the refrigerant. In addition, quote, "taking into account the presence of oil in the enthalpy calculation, which often is neglected, can have drastic consequences on the enthalpy change through the evaporator under particular conditions" (Youbi-Idrissi et al., 2003).

Studies conducted by Cremaschi et al. (2005b) suggested that poor solubility and miscibility between oil and refrigerant, can cause a higher amount of oil retention in evaporators and condensers and it was observed that the COP of the system might be penalized by as much as 9% due to a drop in cooling capacity. Solubility of refrigerant in oil depends on the temperature and pressure of the mixture. In previous experiments, solubility of refrigerant in oil was determined by analyzing the weight fraction of refrigerant present in oil equilibrated at particular temperature and pressure conditions (Bobbo et al., 2010). For oil-refrigerant mixtures, solubility and miscibility are well known for various oil and refrigerant mixtures. In particular, data for R410A and ISO VG 32 POE mixed acid POE oil can be found in the ASHRAE Refrigeration handbook (ASHRAE, 2010). However, there is lack of information about the changes in miscibility and solubility as a result of addition of nanoparticles (Bobbo et al., 2010) or of surfactants.

Thermal Conductivity and Viscosity of Nanolubricants

The increase in thermal conductivity of nanofluids due to the addition of nanoparticles was investigated by numerous researchers and a comprehensive review can be found in a paper by Buongiorno et al. (2009) and in a paper by Oezerinc et al. (2010). Nanofluids have often higher thermal conductivity than that predicted by the macroscopic theory. Venerus and Jiang (2011) pointed out that for systems composed of larger diameter nanoparticles ($\sim 30nm$), there was a good agreement between the measured thermal conductivity enhancement and the one predicted by the classical Maxwell-Garnett model. The thermal conductivity of nanolubricants

was estimated by using Equation 3.3 (Cremaschi, 2012), which was previously proposed by Wen and Ding (2005).

Several existing models can be used to predict the thermal conductivity of the nanolubricant (Buongiorno et al., 2009; Jain et al., 2009; Phillips et al., 1992), and their viscosity and specific heat (Venerus et al., 2010). An example for the viscosity of the lubricant and liquid refrigerant mixture is given in Equation 3.4 (Batchelor, 1977) where k_1 was 2.5 and k_2 was 6.2 and they were modified by Wen and Ding to account for the addition of nanoparticles in the base fluid (Wen and Ding, 2005). Equation 3.4 applies to suspensions of non-interacting particles with a concentration smaller than about 5% by volume. $\mu_{mix,fluid}$ is the dynamic viscosity of the lubricant and liquid refrigerant mixture and it accounted for the lubricant solubility of the refrigerant at given saturation temperatures. Effects of metal oxide nanoparticles dispersed in oil suggest that both thermal conductivity and viscosity increase with the presence of nanoparticles but with different trends depending on temperature range, volume fraction and particle type (Cremaschi, 2012).

$$\frac{k_{nl}}{k_{POE}} = \frac{(1-\phi)(k_p + 2f_{fluid}) + 3\phi k_p}{(1-\phi)(k_p + 2f_{fluid}) + 3\phi k_{fluid}} \tag{3.3}$$

$$\frac{\mu}{\mu_{mix,fluid}} = 1 + k_1\phi + k_2\phi^2 \tag{3.4}$$

3.4 Nanolubricant-Refrigerant Pool Boiling

3.4.1 Nanorefrigerant Pool Boiling

More recent studies investigated the nucleate pool boiling of refrigerant-based nanofluids with surfactants. It was observed that surfactants affect differently the heat transfer performances, and in particular, they enhance heat transfer coefficients when they are used at an optimal concentration (Peng et al., 2011). At too high concentrations, surfactants tended to affect negatively the heat transfer and the reason of this effect was found in the increase of viscosity of the base liquid.

3.4.2 Lubricant-Refrigerant Pool Boiling

The study of pool boiling for lubricant and refrigerants mixtures generally showed a degradation of the boiling heat transfer coefficients with increasing oil concentration because of the lubricant lower vapor pressure, and because of the deposition of an excess lubricant layer on the heated surface, affecting the number of active nucleation sites and the growth of bubbles (Stephan, 1964). The modeling of the pool boiling for lubricant-refrigerant mixture started from the use of traditional single component correlations, where pure component thermophysical properties were initially simply replaced by bulk mixture properties (Chongrungreong and Sauer, 1980). Jensen and Jackman (1984) followed a similar approach including the effect of oil concentration and mass diffusion, however a more detailed model including the effects of lubricant viscosity, miscibility, and concentration was only more recently developed by Kedzierski (2003) for the case of a roughened, horizontal flat surface.

Assuming that during the bubble growth (1) the excess layer is purely composed of lubricant, (2) the lubricant in the excess layer is lifted as a cap on top of each refrigerant bubble, (3) the temperature profile is differently approximated as a linear function within the lubricant excess layer, and as an exponential function beyond the lubricant excess layer (see Figure 3.1), Kedzierski (2003) developed a semi-theoretical model for a generalized range of excess surface densities, and accounting for the influence of specific lubricant properties such as viscosity and miscibility at different saturation temperatures (i.e. for different lubricant-refrigerant mixtures developing different temperature profiles within the liquid layer).

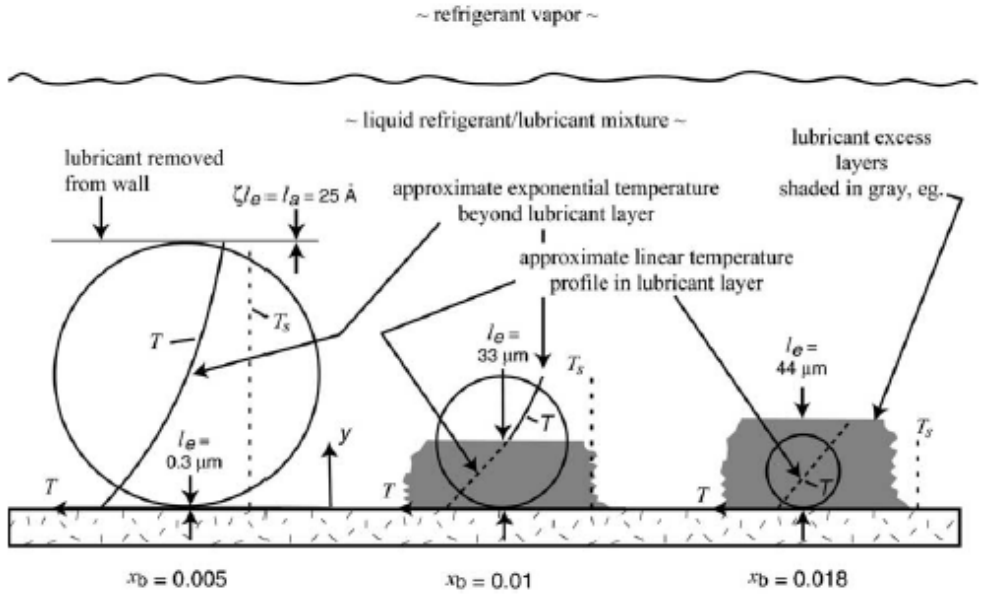


Figure 3.1: Schematic of the lubricant excess layer, l_e, and bubble departure at different lubricant mass fraction, x_b (figure from Kedzierski (2003)).

Kedzierski (2003) observed that a thin lubricant excess layer acts as a surfactant helping to enhance the heat transfer coefficient by reducing the liquid-solid surface tension energy. However for lubricant mass fractions higher that 0.001 the heat transfer coefficient was penalized with respect to the pure refrigerant case. This observation lead to the conclusion that higher lubricant mass fractions are generally responsible for a reduction in bubble size causing lower vapor generation.

3.4.3 Nanolubricant-Refrigerant Pool Boiling on a Smooth Surface

Later works by Kedzierski (Kedzierski and Gong, 2009; Kedzierski, 2011) investigated pool boiling of mixtures of refrigerant R134a and nanolubricants on a roughened flat surface. Nanoparticles of CuO and Al_2O_3 were dispersed in an ISO VG 68 POE oil at mass fractions of about 3.6% and 5.6%, respectively. The nanolubricants were then mixed at different mass concentrations (0.5%, 1%, 2%) with refrigerant R134a. Pool boiling tests were conducted at saturation temperature of 277.6 K for a range of heat fluxes from 7 to 130 $\frac{kW}{m^2}$. The nanoparticles were stabilized by use of a proprietary surfactant at a mass between 5% and 20% of the nanoparticles mass.

The experimental results showed that the use of nanoparticles induced a significant enhancement of boiling heat transfer compared to the case of pure oil. Enhancements were higher (up to 275% for CuO and up to 400% for Al_2O_3) at lower nanoparticle mass concentration (0.5%). For the case of CuO nanoparticles, enhancements were measured at all ranges of heat fluxes, except for the 2% mass concentration that measured a degradation with respect to the pure oil case for heat fluxes higher than 55 $\frac{kW}{m^2}$. For the case of Al_2O_3 nanoparticles, enhancements were measured for all mass concentrations at heat fluxes lower than 40 $\frac{kW}{m^2}$. Degradation were measured for the 1% mass concentration at heat fluxes greater than 40 $\frac{kW}{m^2}$, and for the 0.5% mass concentration at heat fluxes greater than 70 $\frac{kW}{m^2}$.

Kedzierski and Gong (2009) proceeded to investigate the relative effect of the nanofluid thermal conductivity enhancement due to the presence of nanoparticles and concluded that the increase in the measured pool boiling heat transfer could not be justified solely by an increase of thermal conductivity. More mechanisms involving the nanoparticles were hypothesized to be taking place at the wall surface. In particular, it was suggested that nanoparticles may induce "secondary nucleation" on the wall surface; nanoparticles agglomeration could generate or increase the porosity of the wall surface; nanoparticles movements (Brownian motion) may favor fluid mixing. However, nanoparticles could also deposit and clog the surface cavities, therefore causing a degradation of performances. In conclusion, nanoparticles material, shape, size, distribution, and concentration are of fundamental interest to better understand the impact and convenience of nanoparticles on the heat transfer performances.

Kedzierski (2011) proposed a model based on the assumption that the nanoparticles are well dispersed in the lubricant excess layer, and that nanoparticles do not affect the nucleation site density nor the bubble formation frequency. The heat transfer enhancement is described as the consequence of momentum transfer upon impact from the nanoparticles moving at higher velocity, to the bubble mass. The model could estimate the heat flux of a nanolubricant-refrigerant mixture (q''_{np}) by applying a correction factor to the heat flux of a pure oil-refrigerant mixture (q''_{pL}). The correlation is reported in Equation 3.5.

$$\frac{q''_{np}}{q''_{pL}} = 1 + \frac{3.45 \cdot 10^{-9}[s]\phi\sigma\nu_L\rho_v x_b^2}{D_{np}^4(q''_n)^{3/2}\rho_L(\rho_{np}-\rho_L)g(1-x_b)^2} \tag{3.5}$$

where: ϕ is the nanoparticle volume fraction, x_b is the bulk lubricant mass fraction, σ is the refrigerant surface tension ($\frac{N}{m}$), ν_L is the pure lubricant kinematic viscosity ($\frac{m^2}{s}$), ρ_L is the pure lubricant density ($\frac{kg}{m^3}$), ρ_v is the vapor refrigerant density ($\frac{kg}{m^3}$), ρ_{np} is the nanoparticle density ($\frac{kg}{m^3}$), D_{np} is the nanoparticle diameter (m), q''_n is equal to q''_{pL} normalized by 1 $\frac{W}{m^{-2}}$. The model predicts that smaller particles and larger volume fractions induce higher heat transfer enhancements. Finally, the model underpredicted the experimental data for heat fluxes greater than 20 $\frac{kW}{m^2}$ with a deviation up to 25%.

3.4.4 Nanolubricant-Refrigerant Pool Boiling on a Finned Surface

The same experimental setup was again used by Kedzierski (2012) for a study on pool boiling of a mixture of refrigerant R134a and Al_2O_3 nanolubricant on a finned surface. The lubricant used was an ISO VG 68 POE. Nanoparticles were dispersed in oil at higher concentrations because the finned surface presented a larger heat transfer active area compared to the case of a smooth surface. Three particles mass fractions were tested: 3.6%, 8.2%, 12.2% (corresponding to volume fractions ϕ of 1.0%, 2.3%, 3.6% respectively). Pool boiling tests were conducted at saturation temperature of 277.6 K for a range of heat fluxes from 10 to 140 $\frac{kW}{m^2}$. The nanoparticles were stabilized by use of a proprietary surfactant at a mass of 6.2% of the nanoparticles mass.

The purpose of this study was to observe the effect of surface characterization on heat transfer, and whether is actually the nanoparticle concentration in oil, or rather the nanoparticle surface density in the excess layer (defined as $\frac{N_{np}}{A_s}$: number of nanoparticles, N_{np} per unit of surface area, A_s) to determine a change in performances. A baseline for experimental comparison was first obtained from testing both pure refrigerant R134a, and mixtures of pure oil and refrigerant in mass ratios of 0.5/99.5 and 1/99.

For a finned surface, the pure refrigerant heat transfer coefficients were higher than the case of a smooth surface. When oil was present, the results reported a heat transfer degradation

for heat fluxes higher than 9 $\frac{kW}{m^2}$. The degradation for finned surface was larger than the one measured for smooth surface at the same oil fractions and superheat values, and it was even higher for higher oil mass fractions. Tests with nanolubricants were then conducted both with samples of different nanoparticles concentrations (ϕ) but same nanoparticle surface density ($\frac{N_{np}}{A_s}$), and with nanolubricant samples of same nanoparticles concentrations but mixed with refrigerant at different mass fractions (0.5%, 1%).

The experimental results showed that for the case with low charge of nanoparticles (1.0% in volume) in oil and 0.5% mass concentration in refrigerant, the heat transfer did not increase with respect to the pure lubricant case at same concentration in refrigerant, and for the same superheat. However, increasing nanolubricant mass fraction from 0.5% to 1% (corresponding to an increase in surface density from $6.4 \cdot 10^{19} m^{-2}$ to $1.3 \cdot 10^{20} m^{-2}$) increased the boiling heat transfer up to about 25% for heat fluxes between 10 $\frac{kW}{m^2}$ to 108 $\frac{kW}{m^2}$. Interestingly, tests with nanolubricants with different nanoparticles concentrations (2.3%, 3.6% in volume) but same nanolubricant-refrigerant mixture mass ratio (1/99) and same nanoparticle surface density ($7.3 \cdot 10^{20} m^{-2}$) showed similar enhancements in heat transfer, up to 155% for heat fluxes between 10 $\frac{kW}{m^2}$ to 120 $\frac{kW}{m^2}$. Based on this last observation, Kedzierski (2012) concluded that the heat transfer enhancements correlated better with the values of nanoparticle surface density ($\frac{N_{np}}{A_s}$), rather than with the values of nanoparticle volume fraction (ϕ). For this reason, Equation 3.5 was rewritten by the same author in terms of surface density, as shown in Equation 3.6

$$\frac{q''_{np}}{q''_{pL}} = 1 + \frac{1.45 \cdot 10^{-9} [\frac{s}{m}] \left.\frac{N_{np}}{A_s}\right|_G \sigma \nu_L \rho_v x_b^2}{D_{np}^4 (q''_n)^{3/2} \rho_L (\rho_{np} - \rho_L) g (1 - x_b)^2} \tag{3.6}$$

where $\left.\frac{N_{np}}{A_s}\right|_G$ is for the smooth surface equal to the actual $\frac{N_{np}}{A_s}$. It was speculated that at low heat fluxes, nanoparticles tend to reside on top of the fins, not interacting as much with the bubbles forming closer to the bottom of the fins. As the heat flux increases, more surface is activated and the interaction between bubbles and nanoparticles increases. Therefore, an expression of $\left.\frac{N_{np}}{A_s}\right|_G$ for finned surfaces can be developed but it should be made function of both the charged nanoparticle surface density ($\frac{N_{np}}{A_s}$), and the heat flux (q''_n). Kedzierski proposed the following expression, valid for the specific fin geometry:

$$\left.\frac{N_{np}}{A_s}\right|_G = 4.15 \cdot 10^8 (q_n^{''})^{2.53} \left(\frac{N_{np}}{A_s} \cdot 10^{-20}\right)^{1.47} \tag{3.7}$$

The model predicted the experimental data with heat fluxes lower than 100 $\frac{kW}{m^2}$ with a deviation within 10%.

3.5 Nanolubricant-Refrigerant Flow Boiling

Very few works can be found in the literature regarding forced convective boiling of a mixture of refrigerant and nanolubricants, as most of the work so far was focused on the study of thermophysical properties and pool boiling. One experimental work was conducted by Baqeri et al. (2014) who investigated the convective boiling heat transfer of a mixture of refrigerant R600 and CuO laden lubricant, in a smooth horizontal tube of 103 mm in diameter. The oil used was a RL68H POE and its mixture concentration was about 1 wt.%. CuO nanoparticles had a nominal diameter of 40nm and their concentration in oil varied between 0 wt.% and 5 wt.%. No surfactant was used to stabilize nanoparticles, although data was collected in a time frame of about 8 hours from nanoparticles injection, supposed to guarantee no agglomeration or sedimentation. Tests were performed at low vapor qualities (less than 0.25), at saturation pressure of 2.9-4.3 bar, and for a range of mass fluxes between 50 kg/m^2s and 700 kg/m^2s and heat fluxes between 3 kW/m^2 and 6 kW/m^2. Baqeri et al. (2014) observed that the heat transfer coefficient was enhanced up to 32.6% for increasing nanoparticle concentration up to 2 wt.%. However at 5 wt.% nanoparticle concentration, the heat transfer coefficient decreased 7.94% with respect to the refrigerant-pure oil baseline. No data was provided on pressure drop, and it was speculated that the results observed were justified by an increase in the fluid thermal conductivity, an augmentation of the convective heat transfer due to Brownian motion, a thinning of the boundary layer, an augmentation of surface tension (increasing wettability), and the formation of molecular layer adsorption on the surface of nanoparticles. However, no further investigation was conducted to investigate those phenomena.

Although no other major research study was found on forced convective boiling of a mixture of refrigerant and nanolubricants, in this section, a few works are presented in detail because of their relevance to this dissertation. In particular, the first work describes a recent model for water-based nanofluids in single phase convective heat transfer, and it is presented for its fundamental approach to the modeling of nanoparticles behavior inside a non-static liquid; the second work describes an approach to modeling adiabatic two-phase annular flow for mixtures of refrigerant R410A and POE oil.

3.5.1 Nanofluid Convective Vaporization

Remarkable work on single phase convective heat transfer was done recently by Buongiorno (2006). Buongiorno developed a model based on conservation equations to describe the single phase convective flow of nanofluids by studying the behavior of nanoparticles and the change in thermophysical properties inside the wall boundary layer. The study was made for a water-based nanofluid loaded with Al_2O_3 or TiO_2 nanoparticles with diameters smaller than 100 nm, assuming spherical shapes, at volumetric concentrations of 0.01 and 0.03. The nanofluid flows in a smooth tube with an hydraulic diameter of 1 cm. The model was based on the following assumptions:

1. Incompressible flow
2. No chemical reactions
3. Negligible external forces
4. Negligible viscous dissipation
5. Negligible radiative heat transfer
6. Dilute mixture
7. Base fluid and nanoparticles locally in thermal equilibrium

The conservation equations used to describe a multi-component mixture in two-phase flow were reported according to the formalism used by Bird et al. (2002). Overall four equations

needed to be used: two mass balance equations, respectively for the fluid and for the nanoparticles, one momentum balance equation, and one energy balance equation. The fluid and the nanoparticles continuity equations are reported respectively in Equations 3.8 and 3.9:

$$\nabla \cdot v = 0 \tag{3.8}$$

$$\frac{\partial \phi}{\partial t} + v \cdot \nabla \phi = -\frac{1}{\rho_p} \cdot j_p \tag{3.9}$$

where v is the velocity of the nanofluid, ϕ is the nanoparticle volume fraction, ρ_p is the nanoparticle density, and j_p represents the nanoparticle mass flux with respect to the fluid velocity. Buongiorno (2006) analyzed different mechanisms that could affect the relative velocity of nanoparticles with respect to the base fluid, and therefore affect the nanofluid performances. The mechanisms listed by Buongiorno are: inertia, Brownian diffusion, thermophoresis, diffusiophoresis, Magnus effect, fluid drainage, and gravity settling. Of all these mechanisms, two were found to have a significant impact on the nanoparticle distribution within the boundary layer: the Brownian diffusion and the thermophoresis. The nanoparticle diffusion mass flux j_p can therefore be calculated as the summation of these two slip mechanisms or diffusion contribution, as in Equation 3.10:

$$j_p = j_{p,B} + j_{p,T} = -\rho_p D_B \nabla \phi - \rho_p D_T \frac{\nabla T}{T} \tag{3.10}$$

where D_B and D_T are diffusion coefficients. More details regarding these coefficients can be found in Section 5.3.1 of this dissertation.

Substituting Equation 3.10 into Equation 3.9, leads to:

$$\frac{\partial \phi}{\partial t} + v \cdot \nabla \phi = \nabla \cdot \left[D_B \nabla \phi + D_T \frac{\nabla T}{T} \right] \tag{3.11}$$

where the right hand side represents the nanoparticle slip velocity relative to the base fluid, due to Brownian diffusion and thermophoresis. The fluid momentum balance equation and energy balance equation are reported respectively in 3.12 and 3.13:

$$\rho\left[\frac{\partial v}{\partial t}+v\cdot\nabla v\right]=-\nabla P-\nabla\cdot\tau \tag{3.12}$$

$$\rho c_{p,f}\left[\frac{\partial T}{\partial t}+v\cdot\nabla T\right]=-\nabla\cdot q+h_p\nabla\cdot j_p \tag{3.13}$$

where q is the summation of heat flux due to conduction and to nanoparticles diffusion, represented by Equation 3.14:

$$q=-k\nabla T+h_p j_p \tag{3.14}$$

Equations 3.10, 3.13 and 3.14 lead to a final form of the energy balance equation:

$$\rho c_{p,f}\left[\frac{\partial T}{\partial t}+v\cdot\nabla T\right]=\nabla\cdot k\nabla T+\rho_p c_{p,n}\left[D_B\nabla\phi\cdot\nabla T+D_T\frac{\nabla T\cdot\nabla T}{T}\right] \tag{3.15}$$

where the second term on the left hand side represents convection, the first term on the right hand side represents conduction, and the second and third terms on the right hand side represent contributions due to nanoparticle diffusion. In the case of fully developed turbulent flow, the conservation equations need to be modified to account for the presence of eddy diffusivities. Therefore, in the case of a steady-state flow, near the wall of a round tube, Equations 3.11, 3.12 and 3.15 can be modified to include the effect of momentum, energy, and particle eddy diffusivities (respectively, ε_M, ε_H, ε_p):

$$(D_B+\varepsilon_p)\frac{d\phi}{dy}+\frac{D_T}{T}\frac{dT}{dy}=0 \tag{3.16}$$

$$(\mu+\rho\varepsilon_M)\frac{dv}{dy}=\tau_w \tag{3.17}$$

$$(k+c_{p,n}\rho\varepsilon_H)\frac{dT}{dy}=-q_w \tag{3.18}$$

Assuming that the wall region can be divided as in Figure 3.2, boundary conditions can be adopted to solve the continuity, momentum and energy Equations 3.16, 3.17 and 3.18.

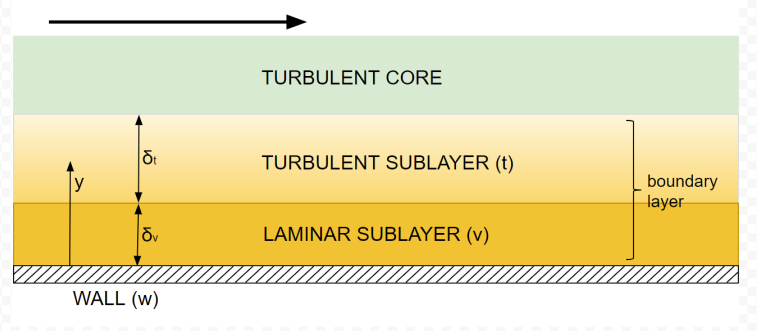


Figure 3.2: Wall region representation (figure adapted from Buongiorno (2006)).

Turbulent Sublayer

Within the turbulent sublayer it can be assumed that $D_B \ll \varepsilon_p$, $\mu \ll \varepsilon_M$, $k \ll \varepsilon_H$. According to Buongiorno's analysis, it is reasonable to assume that $\varepsilon_p \sim \varepsilon_M$ and, by the Reynolds analogy ($Pr = \nu/\alpha = 1$), also $\varepsilon_H \sim \varepsilon_M$. Equations 3.17 and 3.18 can then be integrated over the thickness of the turbulent sublayer and, taking the ratio q_w/τ_w, results in:

$$\frac{q_w}{\tau_w} = c_{p,n}\frac{T_{vt} - T_i}{u_i - u_{vt}} \tag{3.19}$$

Laminar Sublayer

Within the turbulent sublayer it can be assumed that $D_B \gg \varepsilon_p$, $\mu \gg \varepsilon_M$, $k \gg \varepsilon_H$. Equation 3.18 was substituted in Equation 3.16, leading to:

$$D_B\frac{d\phi}{dy} - \frac{D_T}{T}\frac{q_w}{k} = 0 \tag{3.20}$$

Equation 3.20 can be integrated to obtain the distribution of nanoparticle concentration:

$$\phi = \phi_b e^{-(\frac{1}{N_{BT}})(1-\frac{y}{\delta_v})} \tag{3.21}$$

where

$$N_{BT} = \frac{D_B T_b \rho}{\beta\mu\left[\frac{q_w\delta_v}{k}\right]} \tag{3.22}$$

and it represents the ratio between the Brownian and the thermophoretic diffusion coefficients. The integration of Equation 3.21 gives the average nanoparticle volume fraction across the liquid laminar sublayer of thickness δ_v:

$$\phi_v = \frac{1}{\delta_v} \int_0^{\delta_v} \phi dy \tag{3.23}$$

$$\phi_v = \phi_b N_{BT} \left(1 - e^{\frac{-1}{N_{BT}}}\right) \tag{3.24}$$

The determination of ϕ_v is important because it is a function of N_{BT} (representing the effect of nanoparticles' diffusion mechanisms), and it represents the variation of nanoparticles concentration in the laminar sublayer with respect to the bulk concentration. The liquid thermophysical properties of the bulk and of the laminar sublayer are then calculated with a set of correlations that are function of the specific nanoparticles concentration. Equations 3.17 and 3.18 can then be integrated over the thickness of the laminar sublayer, leading to:

$$\frac{q_w}{\tau_w} = \frac{k_v}{\mu_v} \frac{T_w - T_{vt}}{u_{vt}} \tag{3.25}$$

Heat Transfer Coefficient Calculation

Buongiorno developed a heat transfer correlation by introducing the definition of heat transfer coefficient ($h = q_w/(T_w - T_{core})$) in the summation of Equations 3.19 and 3.25, leading to the following expression:

$$h = \frac{\tau_w}{\frac{\mu_v u_{vt}}{k_v} + \frac{u_i - u_{vt}}{c_{p,n}}} \tag{3.26}$$

After more substitutions, the final form of the heat transfer correlation proposed by Buongiorno (2006) for nanofluids was:

$$Nu = \frac{\frac{f}{8} \left(Re_b - 1000\right) Pr_b}{1 + \delta_v^+ \sqrt{\frac{f}{8}} \left(Pr_v^{2/3} - 1\right)} \tag{3.27}$$

where the subscripts b and v indicate quantities calculated using thermophysical properties respectively of the "bulk" and of the "laminar sublayer", f is the friction factor, and δ_v^+ is a value of dimensionless thickness of the laminar sublayer that should be determined experimentally. This correlation proved successful in predicting experimental data reported by other sources for single phase turbulent heat transfer in nanofluids.

It was speculated that the enhancement in heat transfer observed experimentally was the consequence of the redistribution of nanoparticles in the laminar sublayer close to the wall of the tube: due to the effect of thermophoresis and higher temperatures, the particles tend to move away from the wall, locally reducing the viscosity of the fluid, and leading to an increase in heat transfer. It is therefore not sufficient to just correct the thermophysical properties of the base fluid (such as the thermal conductivity and viscosity), to justify the enhancement in heat transfer performances. The relevant characteristics of this approach are also that the model is physically based (an exception is made for the value of δ_v^+), and that it collapses to a pure fluid correlation in case the volumetric concentration of nanoparticles is equal to zero.

3.5.2 Lubricant-Refrigerant Convective Vaporization

The two-phase evaporative flow was extensively described in the literature and an exhaustive description of the theories for annular two-phase flow can be found in (Hewitt and Hall-Taylor, 1970). In this work it is relevant to focus on the study of annular flow for mixtures of refrigerant and oil, and the model developed by Radermacher et al. (2006) to study the oil retention inside evaporators and suction lines was used as a starting point. The model described the annular two-phase flow of oil-refrigerant mixtures using a Navier-Stokes approach applied to both fluid phases. The following assumptions were made:

1. Axial symmetric flow
2. Steady-state and fully developed flow
3. Negligible liquid droplet entrainment
4. Uniform properties of oil and liquid refrigerant mixtures inside the liquid film

5. Flat plate approximation ($\delta_f << R$)

The shear stress distribution in the liquid layer was calculated from the force balance on the control volume represented in Figure 3.3, according to the analysis conducted by Hewitt and Hall-Taylor (1970).

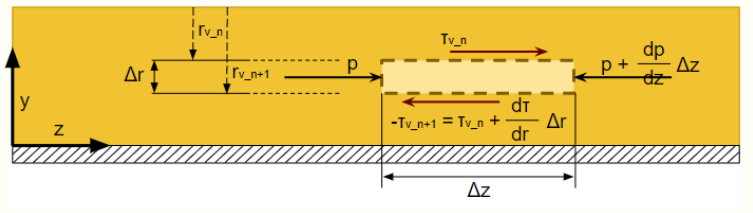


Figure 3.3: Force balance on the control volume.

The shear stress formulation is reported in Equation 3.28:

$$\tau = \tau_i \left(\frac{r_i}{r}\right) + \frac{1}{2}\left(\rho_L g + \frac{dp}{dz}\right)\left(\frac{r_i^2 - r^2}{r}\right) \tag{3.28}$$

where τ_i is the shear stress at the interface between liquid and vapor phase, and it was calculated as in Equation 3.29:

$$\tau_i = \frac{1}{2} f_i \rho_V u_{core}^2 \tag{3.29}$$

proposed in the literature as a general definition for interfacial shear stress. In the equation, f_i represents the interfacial friction factor, generally estimated from empirical correlations. The vapor core velocity was calculated according to Equation 3.30:

$$u_{core} = \frac{G_{flux} x}{\rho_V \alpha} \tag{3.30}$$

where $G_{flux}x$ represent the vapor mass flux, and α is the void fraction calculated according to the theoretical definition ($(D_h - \delta_f)/D_h$). The pressure gradient was found from a balance of forces on the vapor core, and assuming that the pressure drop in the vapor core is the same as the one in the liquid film. The expression for the pressure drop is:

$$\frac{dp}{dz} = \frac{-4\tau_i}{D_h} \tag{3.31}$$

The velocity profile within liquid film was calculated by integrating the shear stress across the liquid film, and is represented by Equation 3.32:

$$u = \frac{1}{\mu_L}\left\{\left[\tau_i r_i + \frac{1}{2}\left(\rho_L g + \frac{dp}{dz}\right) r_i^2\right] \ln\frac{r_0}{r} - \frac{1}{4}\left(\rho_L g + \frac{dp}{dz}\right)\left(r_0^2 - r^2\right)\right\} \tag{3.32}$$

Finally, the liquid film flow rate was calculated by integrating the velocity profile across the liquid film, as in Equation 3.33:

$$\dot{m}_L = \frac{2\pi\rho_L}{\mu_L}\left\{\left[\tau_i r_i + \frac{1}{2}\left(\rho_L g + \frac{dp}{dz}\right) r_i^2\right]\left[\frac{1}{4}\left(R^2 - r_i^2\right) - \frac{1}{2} r_i^2 \ln\frac{R}{r_i}\right] - \frac{1}{16}\left(\rho_L g + \frac{dp}{dz}\right)\left(R^2 - r_i^2\right)^2\right\} \tag{3.33}$$

The system of equations had the following boundary conditions:

1. Symmetry boundary condition: $\frac{du_{core}}{dr}|_{r=0} = 0$
2. Liquid-vapor interfacial shear stress condition: $\tau_{core}|_{r=r_i} = \tau_L|_{r=r_i}$
3. No slip condition at the wall: $u_L|_{r=R} = 0$
4. No slip condition at the liquid-vapor interface: $u_{core}|_{r=r_i} = u_L|_{r=r_i}$

Radermacher et al. (2006) implemented this set of correlations to determine semi-empirical expressions of the friction factor at different operating conditions, by comparison with experimental results for oil retention, and direct measurements of the oil film thickness. The same model was then further expanded by Cremaschi (2012) to describe the heat transfer characteristics of nanorefrigerants, although in that case the friction factor f_i was estimated according to:

$$f_i = 0.005\left(1 + 300\frac{\delta_f}{D_h}\right) \tag{3.34}$$

Chapter 4

Experimental Work

4.1 Measurement of Thermophysical Properties and Two-Phase Flow Performances

4.1.1 Equipment and Instrumentation

The nanolubricant samples were prepared in-house with the equipment described in this section and the thermal and transport properties were measured with the instrumentation described below.

4.1.2 Equipment for Mixing the Nanoparticles in the POE Lubricant

An ultrasonic mixer was used for the developing uniform dispersions of the Al2O3 nanoparticles in the POE oil. The net power output of the sonicator was 750 Watts, at a frequency of 20 kHz. Different probes were used with this device based on the amount of nanolubricant that had to be prepared. For the processing of smaller samples, a $1/2$" (13 mm) diameter probe was used with a griffin beaker while for the processing of larger volumes a graduated cylinder was used with the 1" (25 mm) diameter probe. The time of sonication varied from 8 hours to 24 hours, depending on the volume of the nanolubricant sample that was processed. The sonication was pulsed in cycle of 30 seconds on/off. The concentration of Al_2O_3 nanoparticles in the POE oil, $w_{\%NL}$, was defined as weight percentage of the nanoparticles in the total solid-liquid mixture, as shown in Equation 4.1.

$$w_{\%NL} = \frac{w_{Al_2O_3}}{w_{Al_2O_3} + w_{POE}} \tag{4.1}$$

4.1.3 Equipment for Measuring the Nanoparticle Sizes in Dispersion in POE Lubricant

A DLS instrument was used for measuring the size of the nanoparticles. The device is capable of measuring particle sizes ranging from 4 nm to 10 m diameter. Temperature of the samples was close to room temperature for all the particle measurements reported in this work. The DLS instrument implemented an electrophoretic light scattering technique with a He-Ne laser of 633 nm wavelength. An interface software of the instrument was used to analyze the measurements on-line and correlate the back scattering reflection intensity of the laser to the mean particle sizes of the sample. It should be noted that the nanolubricant was sampled and diluted with POE oil to concentration of less than 1 weight percent before measuring the particle size in order to improve the reliability and accuracy of the particle size measurements.

4.1.4 Equipment for Measuring the Specific Heat of Nanolubricants

The instrument for measuring the specific heat of the nanolubricant was custom built for this work (see Figure 4.1). It consisted of three main components: a temperature bath, a small steel reservoir for the nanolubricant, and an electric heater. A precision temperature sensors and a volt meter were used to read temperature and power. The high precision temperature bath was used to maintain constant boundary temperature conditions around the insulated reservoir. A wire heater rated at 60 W at 120V AC provided heat to the small steel container with the nanolubricant inside it. A variable voltage transformer was used to regulate the power to the electric heater, which was firmly wrapped around the walls of the steel container. A custom made cylindrical stainless steel container with an internal volume of 150 mL was used to store the nanolubricant during the experiments.

Temperature measurements were made by using a precision thermometer with a resolution of 0.01°C and an accuracy of $\pm$0.06 °C. The probe was immersed in the center of the nanolubricant reservoir. An adiabatic condition around the small steel container was obtained by insulating the container with about 2 cm thick layer of rubber flexible foam insulation and by immersing the container in the water inside the temperature bath. A plastic water jacket was installed around the insulation to avoid water ingress into the insulation. The temperature of

the bath was controlled to limit the temperature gradient between the nanolubricant inside the container and the environment surrounding the container.

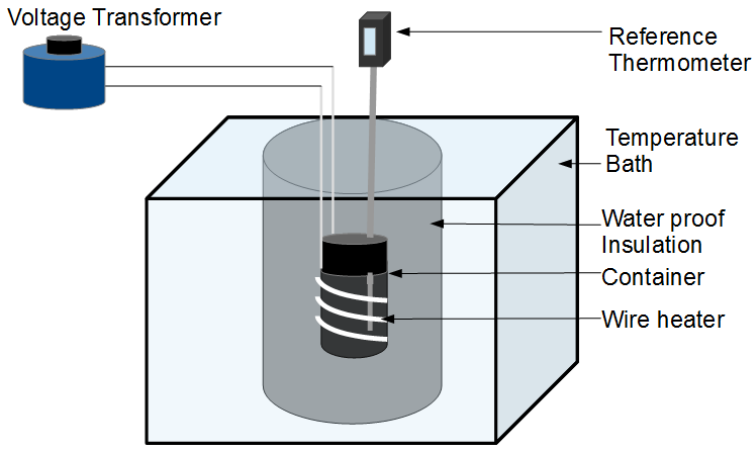


Figure 4.1: Experimental setups for measuring specific heat of nanolubricants.

4.1.5 Equipment for Measuring the Solubility of Refrigerant R410A in Nanolubricants

The instrument for measuring the solubility of refrigerant in nanolubricant was custom build for this work (see Figure 4.2) and it consisted of mainly four components: a temperature bath, a large reservoir, a smaller sample bottle, and a pressure transducer. A vacuum pump was used for depressurization of the large reservoir. For weight measurements, a precision scale with an accuracy of $\pm$0.2g was used. The large reservoir was a stainless steel tank with a working pressure of 1800 psig (12410 kPa) and with a 1 gallon (0.0037 m^3) volumetric capacity. The smaller sample bottle was a custom made 500mL leak proof tank made out of copper. Figure 4.2 shows the schematic of the instrumentation used for measuring the specific heat and the solubility of the nanolubricant samples.

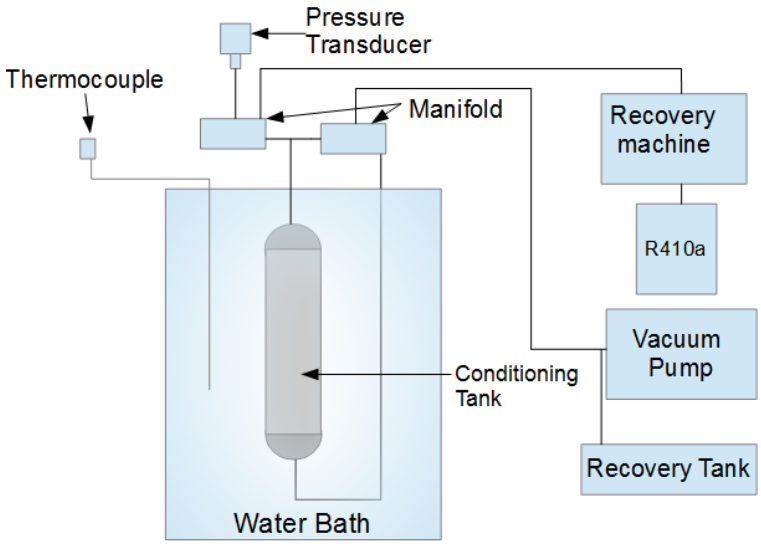


Figure 4.2: Experimental setups for measuring solubility of nanolubricants.

4.1.6 Equipment for Measuring the Thermal Conductivity of Nanolubricants

The instrumentation for measuring the thermal conductivity of the nanolubricant included a thermal conductivity probe and a temperature bath. The thermal conductivity probe (KD2 Pro from Decagon Devices) had a built-in controller and it measured the thermal conductivity of the nanolubricant directly based on a double hot-wire technique. The accuracy of the probe was $\pm 0.01 \frac{W}{m-K}$ for the range from 0.02 to 0.2 $\frac{W}{m-K}$.

4.1.7 Equipment for Measuring Two-Phase Flow Performances

Flow measurements were performed to observe the heat transfer and pressure drop performances of different nanolubricant-refrigerant mixtures in a two-phase flow. Figure 4.3(a) shows a schematic of the test setup built for this purpose. Subcooled refrigerant was circulated to a Coriolis-type flow meter and temperature and pressure were measured to calculate the enthalpy of the refrigerant at the inlet of the preheater. Then the refrigerant entered the preheater, which consisted of a counter flow tube-in-tube heat exchanger. Hot water was circulated in the outer tube of this heat exchanger. The total heat transfer rate from the water side to the refrigerant side was measured and thus, the enthalpy and quality of the refrigerant exiting the preheater, which were identical to the enthalpy and quality at the inlet of the test section, were controlled.

The schematic of the test section is represented in Figure 4.3(b). A thermal amplification technique was used in this work to control the heat flux in the test section. Pressure taps were installed at the inlet and at the outlet of the test section. The absolute pressure was measured at the outlet of the test section and three differential pressure transducers, which had different ranges of application, were used to measure the pressure drop across the test section. The mass flow of the refrigerant-nanolubricant mixture entering the test section was regulated using a variable speed gear pump.

From the test section the refrigerant was circulated to a post-cooler where it was brought to subcooled liquid before entering the gear pump and starting the cycle again. Nanolubricants were metered in the refrigerant two-phase flow with very slow rates and in an incremental fashion to guarantee that the nanolubricants were well mixed with the refrigerant flow and that the oil concentration was as uniform as possible. More details of this procedure and on this experimental facility are reported in Deokar et al. (2016).

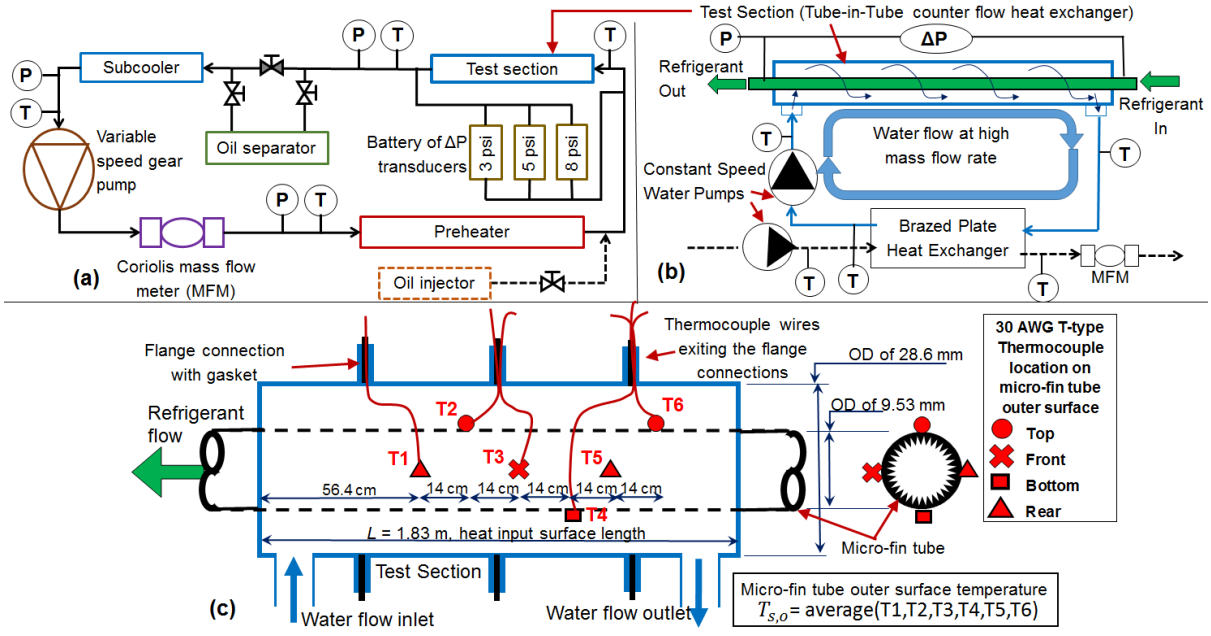


Figure 4.3: (a) Schematic of the test apparatus, (b) detailed schematic of the test section, and (c) details of the location of the surface thermocouples (Figure from Deokar et al. (2016)).

The test section was a counter flow tube-in-tube heat exchanger, which was custom made with embedded thermocouples that measured the wall temperature of the outer surface of the refrigerant tube (see Figure 4.3(c) for details). The heat flux and wall temperature in the test

section were controlled based on a thermal amplification technique described in Smith and Cremaschi (2014). Table 4.1 reports the geometry of the internally enhanced heat transfer surface tube.

Table 4.1: Geometry details of the internally enhanced tube.

PARAMETER	DIMENSION
Length (L)	1.83 m
Outer diameter (D_o)	9.53 mm
Equivalent diameter (D_e)	8.8 mm
Hydraulic diameter (D_h)	5.45 mm
Number of internal fins (N_{fins})	60
Fin length/height (h_{fin})	0.203 mm
Apex angle (α)	30°
Helical angle (γ)	18°
Wall thickness	0.3 mm
Cross sectional area (A_c)	60.8 mm^2
Wetted perimeter	46.7 mm
Inner heat transfer surface	107,040 mm^2

4.2 Experimental Methodology

The following paragraphs describe the procedures followed to collect the experimental data.

4.2.1 Procedure to Measure the Potential of Nanoparticle Sedimentation and Agglomeration

The procedure for conducting the sedimentation tests included the preparation of the nanolubricant samples, the storage of the samples, and the measurements of samples with few droplets of the nanolubricant from the bottom of the container and from the top of the containers used to store the nanolubricant test specimens. The measurement of particle size was performed using the nanosizer. About 80 mL of each type of nanolubricant were created with a nanoparticle

concentration of 0.5 wt % and of 1 wt % (i.e. two concentration for each type of nanolubricant). A 100 mL beaker was used, and the dry weight of the beaker was measured. The mass of oil required was added into the beaker using a 10 mL syringe. The concentrated solution of nanoparticle and POE oil was then added to the POE oil to achieve the required concentration according to Equation 4.1.

The nanoparticle and oil mixture was sonicated for 24 hours with pulse on/off cycle of 30 seconds each. After the nanolubricant samples were prepared the particle size was immediately measured. Small droplets of nanolubricant were taken from the top and from the bottom of the 100 mL container that stored the nanolubricant samples. This first measurements were used as initial size of the nanoparticles and then the particle size was measured regularly every 2 weeks for a period of five months in order to check for potential agglomeration and sedimentation effects. If there was agglomeration of particles, the size measured at both the top and bottom of the 80 mL sample would increase, if there was sedimentation as a result, the measured size of the bottom samples from the 80 mL beaker would increase. Since only the surfactant coated nanoparticles (dispersant) and the POE (medium) are present in the samples, the size of the nanoparticles are measured by the DLS instrument.

4.2.2 Procedure to Measure the Specific Heat of the Nanolubricants

Maintaining heat loss at a minimum was crucial to acquire good results for the specific heat tests. A 150 mL container was used to hold the sample. The container was sealed and a thermometer was fixed onto the container using an air tight sealing putty. This setup was placed into the insulation and inside the thermal bath. The heater was then switched on and timed. The voltage transducer was dialed up to 60 V. The fluid temperature increased and was continuously measured by the reference thermometer inserted inside the container. The bath temperature was raised to match the inside temperature as the nanolubricant was heated.

Four different temperature ranges were taken, from 2°C to 12°C, from 12°C to 22°C, from 22°C to 32°C and from 32°C to 42°C. After each temperature was reached, the heater was turned off, the time was stopped and the whole system was allowed to come to thermal equilibrium. The water was stirred slightly in order to promote even temperature on the entire

nanolubricant sample. The final temperature was read and was used to calculate the specific heat of the nanolubricant. The resistance of the heater was also measured. For each heating phase of the nanolubricant, the temperature of the bath was at the initial temperature of the nanolubricant. This ensured repeatability of the experiments and limited the heat losses.

4.2.3 Procedure to Measure the Solubility of Refrigerant R410A in the Nanolubricants

The solubility tests were conducted by measuring the weight of refrigerant that was solubilized in the nanolubricant. This was done by submerging the conditioning tank into the water bath which was maintained at constant temperature. The conditioning tank was then depressurized to approximately 1.5 psia (10.34 kPa) using the vacuum pump. 200 mL of nanolubricant was introduced into the conditioning tank through the Schrader valve at the bottom of the conditioning tank, taking advantage of the pressure difference between the atmosphere and the conditioning tank. Refrigerant R410A was introduced into the tank through the bottom of the tank until the required pressure was achieved. The mixture was allowed to reach thermal equilibrium under the specific temperature and pressure. The temperature of the bath was monitored using a thermocouple, and the pressure was monitored using an absolute pressure transducer.

To make sure that equilibrium was achieved, the vapor pressure of the conditioning tank was monitored until the pressure stabilized at the desired pressure of measurement. The recovery tank was then depressurized to a pressure of about 1 psia (6.8 kPa), using the vacuum pump. The tare weight of the recovery tank was measured and recorded as ω_0. The refrigerant-nanolubricant mixture was then extracted into the recovery tank and the weight of the mixture and the recovery tank was recorded as w_{NL+Ref}. The recovery tank was then placed into a hot water bath at a temperature of about 600 °C, the vacuum pump was used to depressurize the tank to approximately 1 psia (6.8 kPa). The remaining oil in the recovery tank was then measured and recorded as w_{NL}. The weight of the refrigerant vacuumed out of the recovery tank divided by the weight of the refrigerant and oil after extraction was the weight percentage of refrigerant in the nanolubricant shown in Equation 4.2

$$w_{\%ref} = \frac{w_{ref}}{w_{ref} + w_{NL}} \tag{4.2}$$

4.2.4 Procedure to Measure the Thermal Conductivity of the Nanolubricants

The nanolubricant sample was kept at rest in a sample container provided by the manufacturer of the thermal conductivity probe. The sample container was filled with the nanolubricant and it was immersed in a thermal bath to ensure that the sample temperature was controlled. For each measurement the temperature bath was switched off before immersing the probe in order to limit forced convection effect due to the vibrations coming from the thermal bath pump. Thermal conductivity was a direct output of the probe immersed in the nanolubricant. Each measurement took few minutes for achieving thermal equilibrium and each measurement was repeated 3 times.

4.2.5 Procedure to Measure Two-Phase Flow Performances

The experimental methodology and uncertainty analysis is described in detail in Smith and Cremaschi (2014) and it is summarized next for completeness. The heat flux at the tube wall, q", was directly measured on the water side of the test section. The heat transfer coefficient, α, the heat transfer factor, HTF, and the pressure drop factor, PDF, were calculated according to Equations 4.3, 4.4, and 4.5. The refrigerant reference temperature, $T_{r,ref}$, in Equation 4.3 was calculated according to Equation 4.6. The use of a reference temperature (instead of the saturation temperature at the measured absolute pressure) was proposed by Sawant et al. (2007) to account for the deviation of the refrigerant composition from the standard blend properties due to (i) the presence of stray impurities as result manufacturing processes, and (ii) the presence of oil in the liquid refrigerant. The constants A_1 and A_2 and the polynomials a and b, accounted for deviations of the refrigerant saturation temperature. α_0 and ΔP_0 are the representative baseline heat transfer coefficient and pressure drop for the oil free case.

For each test with refrigerant R410A and POE oil (or with R410A and nanolubricant), the measured α and ΔP were compared with the corresponding α_0 and ΔP_0 at the same mass flux, heat flux, and quality. The HTF and PDF were then calculated and they represented figure of merits that isolated and quantified the effects on the heat transfer coefficient and on the pressure drop due to the presence of POE oil or of nanolubricant in the refrigerant flow.

$$\alpha = \frac{q"}{T_{r,ref} - T_{wall,in}} \tag{4.3}$$

$$HTF = \frac{\alpha - \alpha_0}{\alpha_0} * 100 \tag{4.4}$$

$$PDF = \frac{\Delta P - \Delta P_0}{\Delta P_0} * 100 \tag{4.5}$$

$$\begin{aligned} T_{r,ref} &= \frac{a}{ln(P_{ref}) - b + \frac{A_2}{A_1}x} \\ a &= a_0 + 182.5\omega - 724.2\omega^3 + 3868\omega^5 - 5268.9\omega^7 \\ b &= b_0 - 0.722\omega + 2.391\omega^3 - 13.779\omega^5 + 17.066\omega^7 \end{aligned} \tag{4.6}$$

Pressure drop in the test section was measured using one of three differential pressure transducers installed across the test section. The differential pressure transducers covered a range of 0-55 kPa but one transducer was used for the small range from 0 to 20 kPa, one for the medium range from 0 to 34 kPa, and one for the high range from 0 to 55 kPa. The accuracy of each pressure transducer was $\pm 0.1\%$ of the full scale. The measured pressure drop was divided by the length between the two pressure taps in the test section in order to obtain an average pressure gradient along the test section for each saturation temperature and quality.

Two phase flow boiling heat transfer coefficient and pressure drop were measured for refrigerant R410A and nanolubricant mixtures at saturation temperature of $3.5 \pm 0.9°C$, OMF ranging from 0 to 3%, mass flux ranging from 180 kg/m^2s to 425 kg/m^2s, and heat flux of 12 kW/m^2 and 15 kW/m^2. Two nanolubricants were tested at nanoparticles concentration that varied from 2 to 20 wt.%. Table 4.2 reports the test conditions of the two-phase flow measurements.

Table 4.2: Test conditions of the two-phase flow measurements.

LUBRICANT* $[NP_{conc}(wt.\%)]$	HEAT FLUX† $[kW/m^2]$	OMF• $[\%]$	MASS FLUX⋆ $[kg/m^2s]$
-	12, 15	0.5, 1, 3	183, 255, 350, 425
POE	12	3	255, 350
	15	3	255
		0.5, 1, 3	350
γ-Al_2O_3 $[2wt.\%, 10wt.\%]$ in POE	12	1, 3	350
γ-Al_2O_3 $[20wt.\%]$ in POE	12	1, 3	183, 255, 425
		3	350
	15	1, 3	350

*NP_{conc} defined as: $m_{np}/(m_{np} + m_{oil})$; †Heat flux: $12 \pm 0.2kW/m^2$, $15 \pm 0.3kW/m^2$;

•OMF defined as: $m_{oil}/(m_{oil} + m_{ref})$; ⋆Mass flux: $183 \pm 4kg/m^2s$, $255 \pm 8kg/m^2s$, $350 \pm 8kg/m^2s$, $425 \pm 9kg/m^2s$.

4.3 Calibration, Validation, and Uncertainty Analysis

For specific heat and solubility tests, in which newly developed test apparatuses were used, preliminary experiments were conducted by using water and by using POE oil only in order to calibrate the instrumentation and refine the test procedures. A completely adiabatic system could never be achieved during the specific heat tests of this work and three sets of calibration tests were performed to estimate the heat losses in POE oil and to confirm the repeatability of the tests. The experiments showed that the heat losses were small (but not negligible!) and repeatable. Using a heat loss correction factor, the specific heat of POE lubricant was measured and the data in this work were within 3 $\%$ error when compared to values in the literature (Thome, 1995).

Similarly, preliminary tests were made to estimate the solubility of refrigerant R410A in POE lubricant by using the developed test apparatus. This calibration provided refinements in

the sampling procedure and in the mixing of the refrigerant and lubricant inside the reservoir before making the solubility measurements. The data of solubility with refrigerant R410A and POE lubricant were within 5 $\%$ agreement with respect to the Cavestiris relations provided in the ASHRAE Refrigeration Handbook (ASHRAE, 2010).

The uncertainty in the measurements was calculated by using a Taylor series expansion method (the calculations were carried out using EES software (Engineering Equation Solver)). The maximum uncertainties calculated for each experiment are given in Table 4.3.

Table 4.3: Maximum uncertainty from experiments on thermophysical properties.

TEST	MEASUREMENT OBJECTIVE	MAX UNCERTAINTY
Sedimentation	Nanoparticle size	$\pm 2\%$
Solubility	Weight percent of refrigerant in the nanolubricant	$\pm 1.2\%$
Specific Heat	Specific heat of the nanolubricant	$\pm 2.3\%$
Thermal Conductivity	Thermal conductivity of the nanolubricant	7.2%

For two-phase flow measurements, the error analysis and uncertainty propagation are summarized in Table 4.4. Several tests were repeated multiple times during the experimental campaign to assess the variability and repeatability of the experimental data due to human error, charging procedures, and cleaning and purging of residual lubricant and nanolubricants inside the test apparatus at the end of a series of tests.

Table 4.4: Experimental uncertainties and repeatability of two-phase flow measurements.

VARIABLE	UNITS	ACCURACY / UNCERTAINTY	DEVIATION DURING REPEATED TESTS
Measured Variables			
T_{sat}	°C	± 0.1°C	(within accuracy)
P_{ref}	kPa	± 1.0 kPa	(within accuracy)
m	kg/m^2s	± 0.1%	(within accuracy)
q”	kW/m^2	± 0.53%	(within accuracy)
x	-	± 1.8%	(within accuracy)
ΔP	kPa	± 0.07 kPa	± 1 kPa (x=0.3), ± 2 kPa (x=0.8)
Derived Variables			
T_{ref}	°C	± 0.2°C	
α	kW/m^2C	± 10.7%	± 0.13 kW/m^2C (0.3<x<0.6), ± 0.5 kW/m^2C (x=0.8)
HTF	%	± 14.5%	± 5% (0.3<x<0.6), ± 18% (x=0.8)
PDF	%	± 1.0%	± 9.5% (x=0.3), ± 16% (x=0.8)

4.4 Results and Discussion

In the following section the experimental results are presented and discussed. When found, the acronyms T1S10, T1S20, T2S10, and T2S20 stand for nanolubricant using either surfactant type 1 or type 2 (T1 or T2), with a nanoparticle mass fraction in POE oil or 10% or 20% (S10 or S20).

4.4.1 Sedimentation Test Results

The nanoparticle sedimentation tests results are plotted in Figure 4.4. The x-axis represents the time in weeks and the y-axis shows the nanoparticle normalized diameter which is the ratio of the measured diameter and the smallest measured diameter for the entire experiment. Tests were conducted for three types of nanolubricants (type 1, type 2, and type 3).

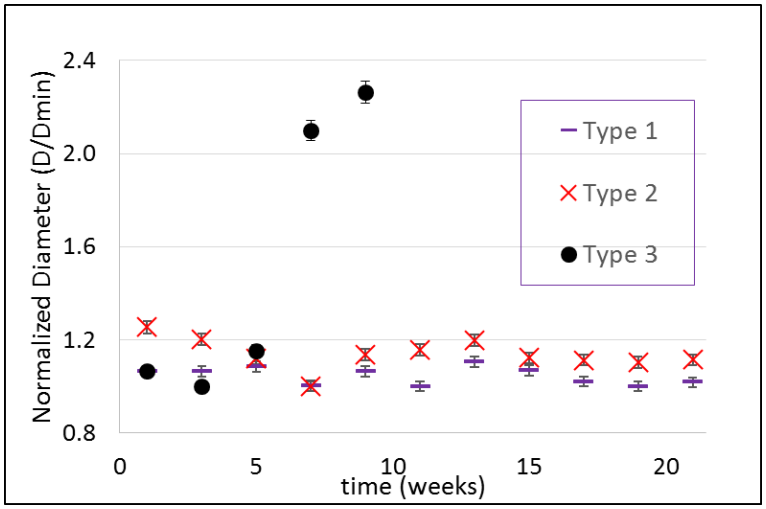


Figure 4.4: Sedimentation test results for three types of nanolubricants.

The concentration of the nanolubricant samples for the sedimentation tests was 1 weight percent. This was the minimum nanoparticle concentration and thus the least viscous solution possible. Type 1 and type 2 samples, which have the same Al_2O_3 nanoparticle type but different surfactants, showed that the nanoparticle size did not increase over a 14 week period. These results indicated that there was no agglomeration and no signs of clusters of the nanoparticles in these nanolubricant types, as there was no increase in particle size over time for type 1 and type 2 samples. The data also shows that both top and bottom layers of the containers had same nanoparticle size. These results indicated that there were not any signs of sedimentation and the nanoparticle suspensions type 1 and type 2 were stable. The ratio of the measured particle size over the minimum particle size was within the range of 1 to 1.5. Visual confirmation of these results are illustrated in Figure 4.5.

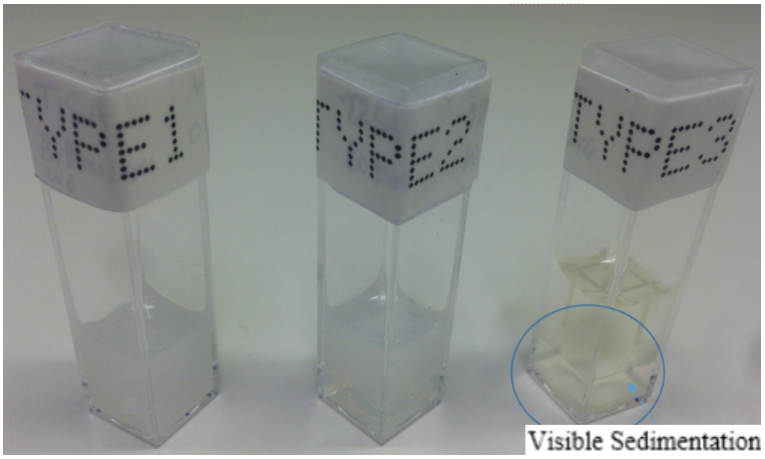


Figure 4.5: Visual observation of sedimentation for type 3 nanolubricant.

Type 3 nanolubricant, which had same nanoparticles but used a different surfactant than either type 1 and type 2, showed agglomeration in Figure 4.4 (solid round data points) and sedimentation (see Figure 4.5 type 3 within the blue circle). The particle sizes for type 3 increased with time, starting from a size ratio of about 1 and increasing over time to about 2.5 indicating that the particles were agglomerating and sedimentation was taking place. The samples were taken from the bottom of the sample where the largest concentration was present due to sedimentation of the type 3 nanolubricant.

It is to be noted that the given Al_2O_3 particle size in the dry state is about 40nm according to the manufacturer. However, DLS measurements recorded particle sizes of about 120-200nm for the stable type 1 and type 2 samples. The average measurement results were 126.7 nm for Al_2O_3 as shown in Figure 4.6. It was confirmed with the manufacturer of the nanoparticles who used the same DLS measurement technique that a particle size within the range recorded with the DLS measurements corresponded to a particle size of about 40nm using other measurement techniques. Kedzierski et al. (2017) used similar nanoparticles and they suggested that the manufacturer reported the surface-area weighted diameter while the DLS measured the number-weighted diameter.

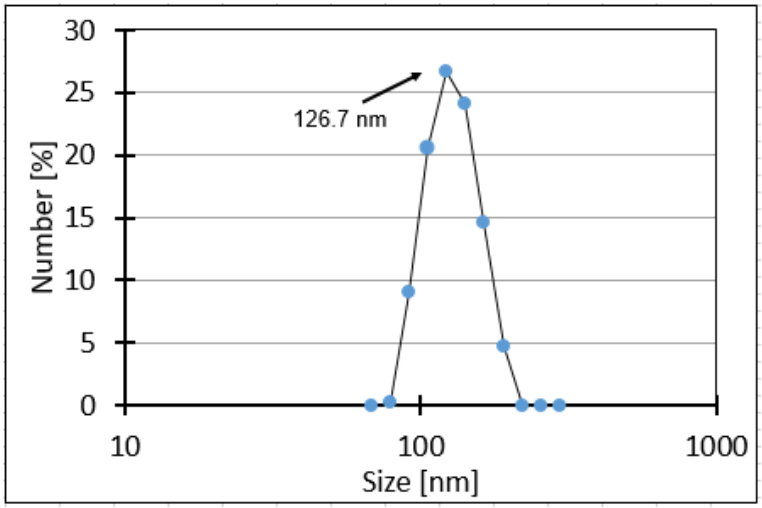


Figure 4.6: Number-weighted average distribution of Al_2O_3 nanoparticle sizes measured with DLS.

4.4.2 Specific Heat Test Results

The specific heat of POE oil is showed in Figure 4.7 and the measured data (represented by solid symbols) and the literature values (Thome, 1995) (dashed line) are plotted for a temperature range from 10 to 40°C. The ratio of the specific heat of the nanolubricants type 1 and type 2 at concentration of 10 and 20 wt.% over the specific heat of POE oil at the same temperature are given in Figure 4.8.

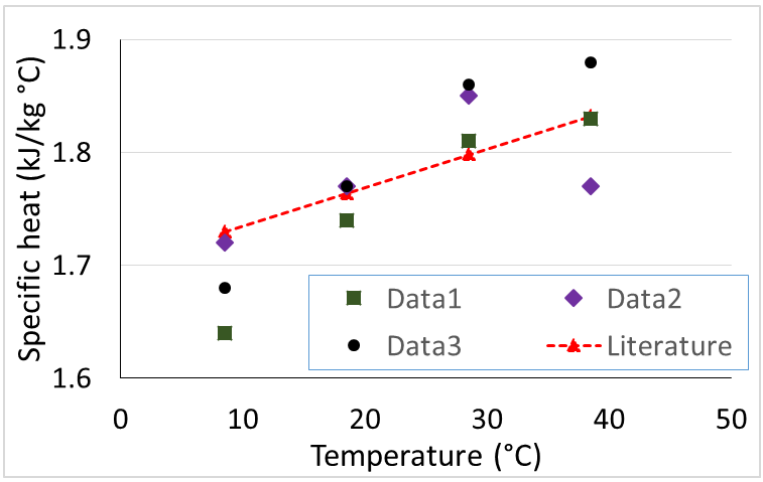


Figure 4.7: Specific heat vs. Temperature of POE.

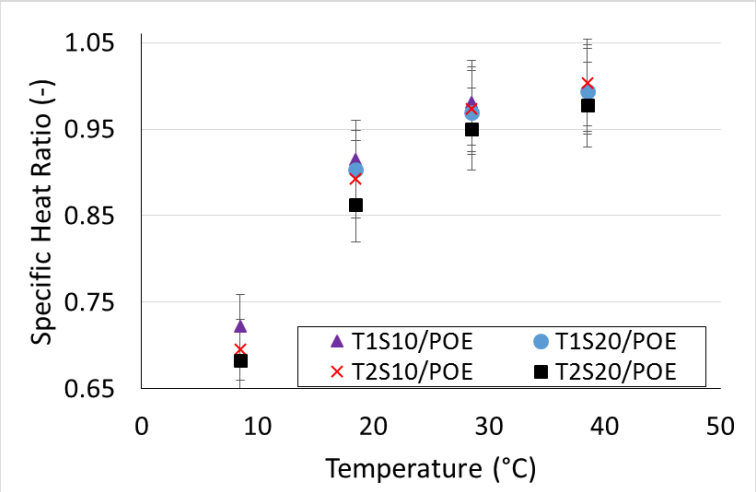


Figure 4.8: Specific heat ratio vs. Temperature of nanolubricants.

The specific heat of the nanolubricants was lower than that of POE oil and the difference was greater at temperatures of about 10°C. When the temperatures were closer to 40°C the nanolubricants had similar specific heat as the POE lubricant. Moreover, the measurements showed a slightly lower specific heat with increasing concentration. This result is in general agreement with other results reported in the literature, although in this case the measurement difference between 10 wt.% and 20 wt.% concentrations is very small. It was speculated that this was caused by high uncertainties of the experimental methodology. The recent investigation by Kedzierski (2018) on the specific heat of Al_2O_3 nanolubricants also considered the effect of surfactant. In their work, they tested a higher viscosity POE oil and, even if they confirmed the trends measured in this work, they also observed larger degradation of the specific heat with increasing particle concentration.

4.4.3 Solubility Test Results

Figures 4.9 and 4.10 show the solubility test results obtained for the nanolubricants with refrigerant R410A. The weight percentage of R410A in nanolubricant is plotted on the x-axis and pressure on the y-axis. Each line represents a specific temperature and the symbols shows the actual data points taken for this work. The dashed lines represent the literature correlations from the ASHRAE handbook (ASHRAE, 2010) and the triangular symbols represent the baseline series of experiments conducted in this work to verify the solubility of R410A in POE.

This baseline series is used to compare the behavior of the nanolubricants at same temperature and pressure conditions.

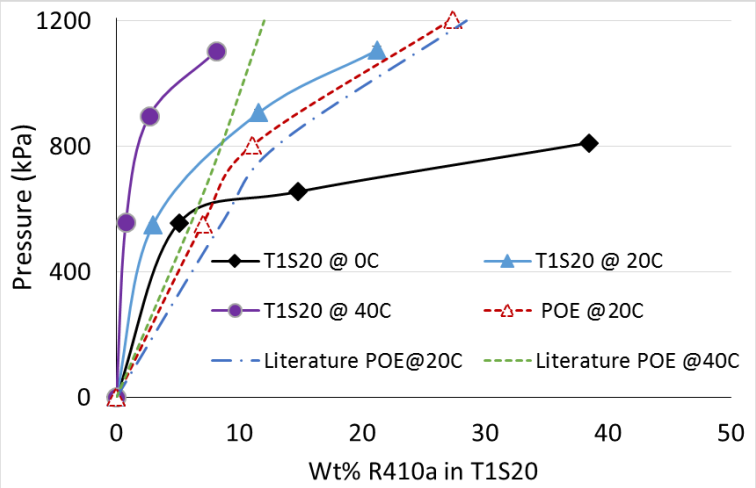


Figure 4.9: Pressure vs. wt.% R410A+T1S20.

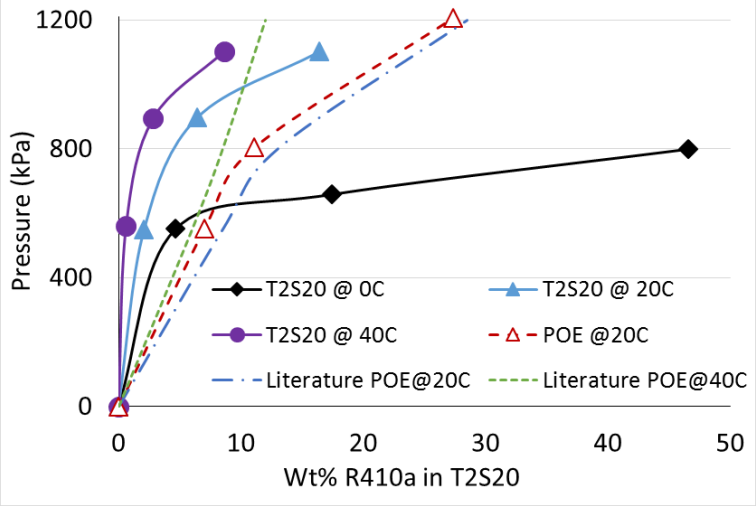


Figure 4.10: Pressure vs. wt.% R410A+T2S20.

Both type 1 and type 2 nanolubricants had lower solubility than that of POE oil with no nanoparticles (and with no surfactants). For example, at 400 kPa and 20°C the solubility of R410A in nanolubricant type 1 was less than 2 % while the solubility of R410A in POE oil was close to 5 %. T2S20 showed the maximum solubility at 46 weight percent refrigerant in lubricant and T1S20 was soluble up to 38.5 weight percent at approximately 0°C and 800 kPa. However, T2S20 showed lower solubility at 20°C relative to T1S20. Both nanolubricants showed about the same solubility characteristics at 40°C.

4.4.4 Thermal Conductivity Test Results

The thermal conductivity of POE lubricant is shown in Figure 4.11 and the ratio of thermal conductivity of each nanolubricant over that of POE oil at the same temperature is shown in Figure 4.12.

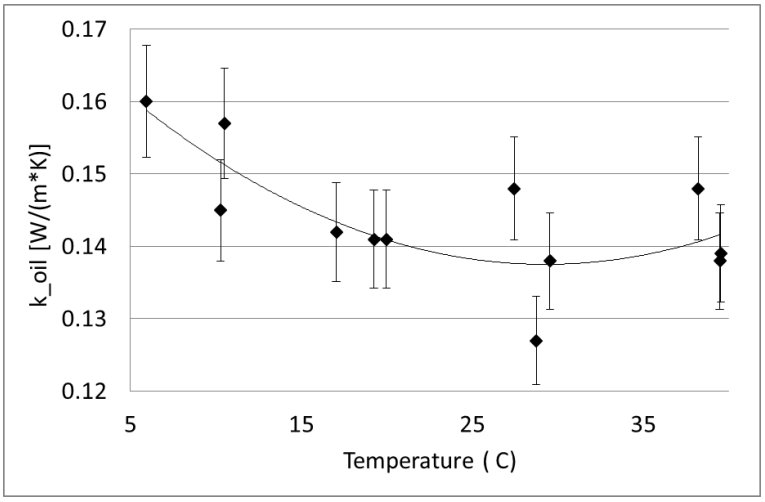


Figure 4.11: Thermal conductivity vs. Temperature of POE.

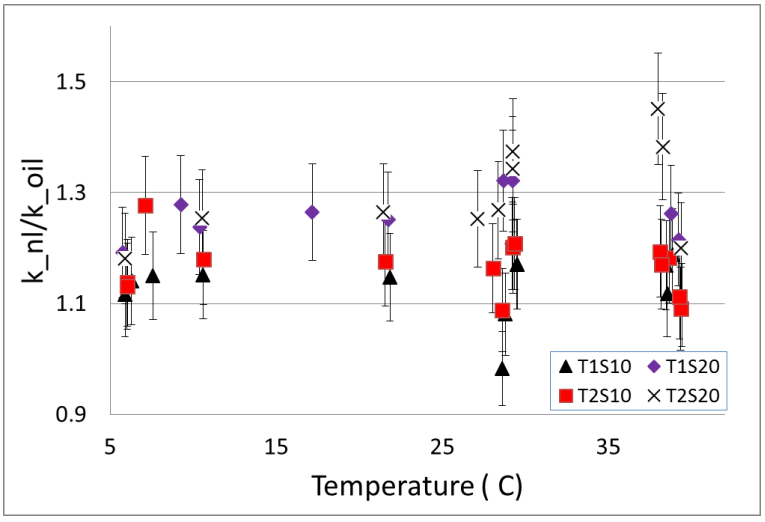


Figure 4.12: Nanolubricant-POE thermal conductivity ratio.

Although there are scattered data in Figure 4.11, it appears that the POE oil thermal conductivity decreased slightly if the temperature increased from 5 to 30°C. Figure 4.12 shows that the highest thermal conductivity was measured for the T2S20 nanolubricant sample followed by T1S20 sample. These samples had the highest concentration of Al_2O_3 nanoparticles of 20

weight percent and their thermal conductivity ranged from 1.5 times higher at 5°C to 2 times higher at 40°C than the thermal conductivity of POE oil at similar temperature. The sample T2S10 showed a higher thermal conductivity relative to T1S10 and both had the 10 weight percent Al_2O_3 nanoparticles concentration. It appears that the surfactant that was used to stabilize the nanoparticles had an effect on the thermal conductivity of the anolubricant. This is evident from the T2S20 and T2S10 data, which had higher thermal conductivity in both 10 and 20 wt.% concentrations when compared their Type 1 sample counterparts.

4.4.5 Viscosity Test Results

The viscosity of the nanolubricants was measured by using a Cannon-Fenske type viscometer. The viscosity of the nanolubricants at 45°C was the same as that of POE lubricant and the type of surfactant did not affect the viscosity at this temperature. The viscosity ratio, defined as the viscosity of a nanolubricant over the viscosity of POE lubricant at similar temperature, at 10 weight percent of nanoparticle concentration ranged from 1.8 if the temperature was 20°C to 2.9 when the temperature was 0°C. The viscosity ratio of nanolubricant with 20 weight percent nanoparticle concentration ranged from 1.9 if the temperature was 20°C to 3.3-3.8 when the temperature was 0°C.

4.4.6 Miscibility Test Results

The refrigerant/nanolubricant concentration were tested for 95/5 % to 30/70 % at a temperature range of -30°C to 60°C. Miscibility results were identical for T1S5 and T1S10 samples. The refrigerant-nanolubricant samples were miscible for all concentrations tested except for 80/20 % at 55-60°C, and 70/30 % at 50-60°C. T1S20 was miscible at a concentration of 60/40 % from -30°C to 55°C. At a concentration of 30/70 % the samples were miscible for the entire range of temperatures. The results indicate that miscibility is dependent on the ratio of refrigerant and nanolubricant as seen in T1S10 which was not miscible for a concentration range of 95/5 % to 70/30 % but was miscible for 60/40 % to 30/70 % concentrations. Temperature also determines the miscibility of the samples as seen in T1S5 and T1S10 which were miscible at 80/20 % for a temperature range of -30°C to 50°C but immiscible for 55-60°C.

4.4.7 Two-Phase Flow Performances

The experimental results of two-phase flow tests are reported here for the heat transfer factor (HTF) and the pressure drop factor (PDF). Figure 4.13 shows the HTFs and PDFs of the refrigerant and POE mixture and of the refrigerant and nanolubricants mixtures at refrigerant saturation temperature of 3.5 $\pm$ 0.9°C. An enhancement of the two phase flow heat transfer coefficient is represented by a positive sign of the HTF, in percentage, on the y-axis while negative percentages of the HTFs in Figure 4.13 mean degradation of the two phase flow boiling heat transfer coefficient of the mixtures with respect to that of refrigerant R410A (i.e. R410A with no POE oil and no nanolubricants resulted in the zero horizontal solid lines at HTF = 0% and PDF = 0% in Figure 4.13).

The experimental uncertainty of the data is reported for few but representative data points shown in Figure 4.13 and it should be noted that all data points in Figure 4.13 have similar error bars due to experimental uncertainty and statistic repeatability confidence of the measurements.

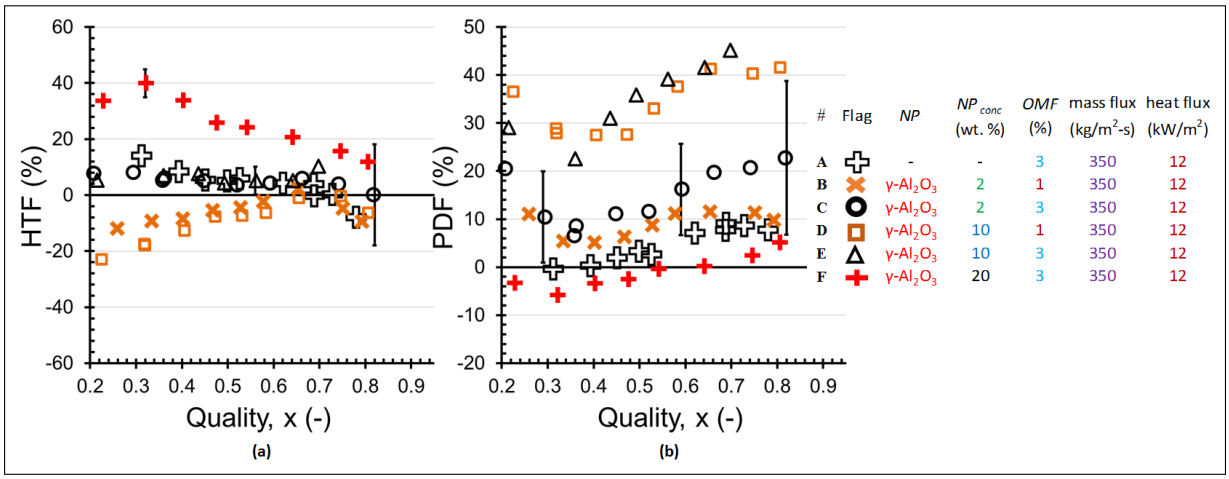


Figure 4.13: (a)HTF and (b)PDF of various concentration of Al_2O_3 based nanolubricant in R410A at $\dot{m}$ = 350 kg/m^2s and q" = 12 kW/m^2, at 1 wt.% and 3 wt.% OMF.

The flow boiling in presence of 3% OMF of POE oil, see series A in Figure 4.13(a), showed up to +10% higher HTF compared to the tests with R410A flow boiling and no oil (i.e., 0% OMF). This intriguing finding was consistent with the observations by Bandarra Filho et al. (2009), in which oil promoted wetting and foaming and slightly increased the heat transfer coefficient. Adding Al_2O_3 nanoparticles at only 2 wt. % concentration in 1% OMF and 10%

OMF of POE oil and refrigerant mixture, see series B and D, caused a sudden decrease in HTF at lower refrigerant qualities.

It is useful to point out that pure POE and 2 wt. % Al_2O_3 nanolubricant had very similar thermal conductivity, and thus the degradation in HTF observed in Figure 4.13 when small percentage of Al_2O_3 nanoparticles were added to the POE oil must have been caused by some phenomena other than the variation of the liquid phase thermal conductivity. Series C and E, in Figure 4.13(a), are representative of 3% OMF and with Al_2O_3 nanoparticles concentration of 2 wt. % and 10 wt. %. The data indicated small variations of the heat transfer coefficient and within the uncertainty of the test apparatus when compared to the data of heat transfer coefficient measured for POE of series A. Only when charging nanoparticles with 20 wt. % concentration and with 3% OMF, a measurably increased heat transfer coefficient was observed. This scenario is indicated by series F in Figure 4.13(a), and the associated HTF was about +35% at low refrigerant thermodynamic qualities and +15% at high qualities.

The deviations of the measured heat transfer coefficients and pressure drops during the R410A repeated tests contributed to the uncertainty error bars indicated in Figure 4.13 (Deokar et al., 2016). While the perturbations did not affect the actual measured pressure drop, they yielded to up to 16% uncertainty when calculating the relative variation of pressure drop with oil versus the pressure drop without oil for similar quality. Only series D and E seems to be higher than all the other series but it was speculated that those series were outliers and had high uncertainty associated on their PDFs. This aspect needs to be further clarified in future follow up research.

On the other hand, even when accounting for the deviation of the repeated tests, the error bars on the HTFs were small and effects of the POE oil and of the nanolubricants on the HTFs were measurable in the experimental data of this work. The representative error bar in the data points in Figure 4.13(a) were smaller than the error bars for the PDFs shown in Figure 4.13(b). In simpler terms, by adding oil and nanolubricants to the refrigerant R410A during two phase flow boiling at saturation temperature of 3.5 $\pm$ 0.9°C, the effect on the heat transfer coefficient was more marked, (and more important measurable!) than the effect on the two phase flow pressure drop, which was very small and not measurable with the present test set up. This

finding was in agreement with the observation reported in the literature that nanolubricants enhanced heat transfer rate without compromising the two phase flow pressure drops (Bartelt et al., 2008).

To further verify the hypothesis that the magnitude of the enhancements of the two phase flow boiling heat transfer coefficient and of the penalization of the pressure drop due to the nanoparticles were dependent on the mass flux and shear rates within the liquid phase of the mixture, additional tests were conducted at very low and very high mass fluxes with respect to design operating mass fluxes for the evaporator tube used in this work. Figure 4.14 provides the experimental results of HTF and PDF (on the y-axis) versus mass flux on the x-axis and for low to high refrigerant thermodynamic qualities. The data in these figures were taken from series of flow boiling tests in micro-fin tube with quality ranging from 0.3 to 0.8 and only 5 representative qualities are shown in Figure 4.14.

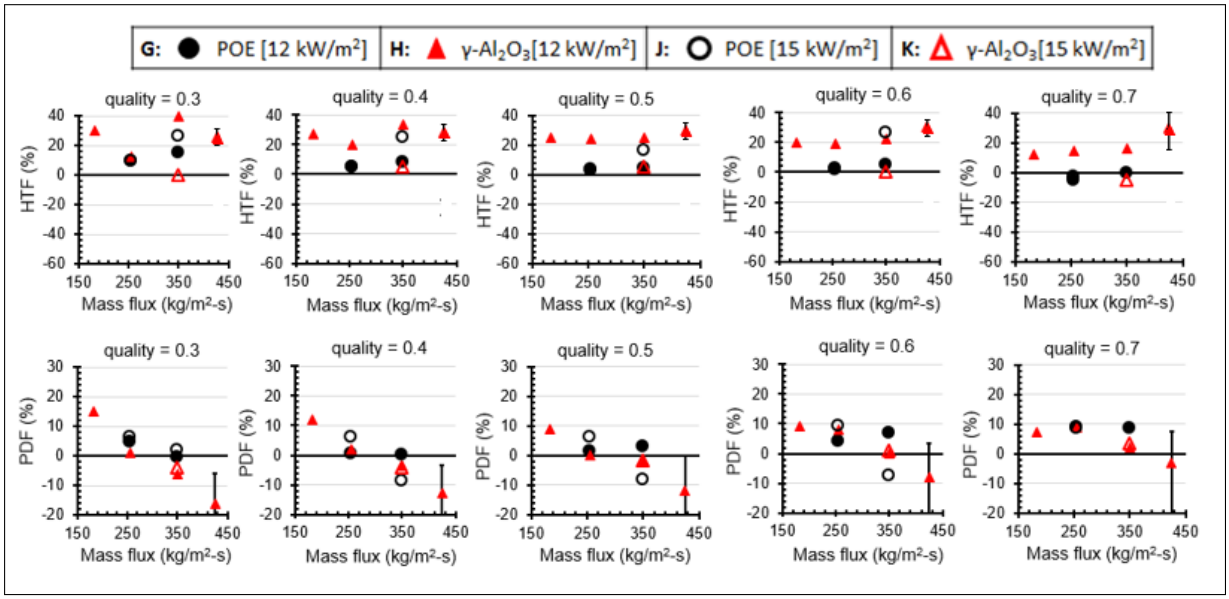


Figure 4.14: (a)HTF and (b)PDF of 20 wt.% Al_2O_3 based nanolubricant in R410A at $\dot{m}$ of 183, 255, 350, 425 kg/m^2s and q" of 12 kW/m^2 and 15 kW/m^2, at 3 wt.% OMF.

The Al_2O_3 nanolubricant at 3% OMF and 20% particle concentration, which is indicated by the solid red triangle data points and legend H in Figure 4.14(a), had a gradual increase from 20% HTF at 183 kg/m^2s to 30% HTF at 425 kg/m^2s, at medium quality of x=0.5. At higher quality of x=0.7, the HTF had a gradual increase from 10% HTF at 183 kg/m^2s to 30% HTF at 425 kg/m^2s. This increase in trend was fairly consistent from medium to high quality. At low

to medium qualities, that is at x=0.3 and x=0.4, the HTFs were always positive and scattering of the data was observed, which suggested that the HTF was fairly independent of the mass flux below quality of x=0.5. Flow boiling with POE at 3% OMF, solid black circle data points and Legend G in Figure 4.14(a), had measurable 10% lower performance than Al_2O_3 nanolubricant in tube with internal micro-grooves. At 3% OMF the Al_2O_3 based nanolubricant had similar PDFs as those of POE for all mass fluxes, as shown in series G, J and H, K in Figure 4.14(b). If the spherical shaped Al_2O_3 nanoparticles generated any additional shear stresses within the liquid phase of the refrigerant and lubricant mixture, the macroscopic scale effect was still not detectable by the test apparatus of this work.

Chapter 5

Simulation Work

5.1 Segment-by-Segment Model

A new simulation model was developed to describe and investigate the behavior of refrigerants and nanolubricants mixtures during two-phase flow boiling. The simulation code was written in the C++ programming language and thermophysical properties of refrigerants were calculated using the CoolProp 5.1.2 open-source library (Bell et al., 2014). An input file was provided as a user interface to define both the geometry of the evaporator tube and the fluid inlet conditions, that is, the type of refrigerant, lubricant, nanoparticle material, shape and mass fraction, and mass flow rates.

Additional inputs to the present model were the evaporator tube inlet pressure and inlet enthalpy of the refrigerant and nanolubricant mixtures. The heat capacity of the evaporator tube was used for setting the heat flux boundary conditions. The simulation solved the mass and energy balances in the evaporator tube. Using existing two-phase flow heat transfer, pressure drop, and void fraction correlations from the open domain literature, the tube heat transfer coefficient and pressure drop were calculated. For the calculation of the pressure drop, an estimate of the outlet conditions was first made based on the inlet conditions and then an iterative loop was implemented to calculate the actual outlet pressure until convergence was achieved. During the convergence process, the local thermophysical properties of the refrigerant and nanolubricant mixtures were updated at each step in order to account for the local concentration of nanolubricant.

To calculate the thermophysical properties of the refrigerant and lubricant and refrigerant and nanolubricant mixtures, five sets of correlations were implemented in the present model. These sets are the lubricant properties, refrigerant and lubricant mixture properties, nanoparticle properties, nanolubricants properties, and refrigerant and nanolubricants properties and they are summarized in the following paragraphs. The thermophysical properties were calculated at the beginning of the analysis of the evaporator tube and then used in the correlations for the local two-phase flow heat transfer coefficient and pressure drop as function of the local quality. The script of the simulation code can be found in Appendix C.

5.1.1 Lubricant Properties

The thermophysical properties of the pure POE were calculated with the following set of correlations.

Lubricant Conductivity

Lottin et al. (2003) suggested the use of the correlation reported in Equation 5.1 to estimate the oil thermal conductivity:

$$k_o = 0.1172 \cdot \frac{(1 - 0.0054 \cdot T)}{\rho_o / \rho_{H_2O}} \tag{5.1}$$

However, the correlation used in this work to estimate the conductivity of oil was developed from experimental tests conducted with the hot wire technique described previously. The correlation is applicable for a range of temperatures between 5°C and 40°C and its uncertainty is $\pm 6\%$. The correlation is reported in Equation 5.2:

$$k_o = 6 \cdot 10^{-6} \cdot T^2 - 0.0006 \cdot T + 0.1513 \tag{5.2}$$

Lubricant Density

Correlations to estimate the density of ISO VG 68 POE oil were suggested by various authors. Reported here are Equations 5.3 and 5.4, respectively used by Hu et al. (2008a) and Kedzierski (2013):

$$\rho_o = (0.97386 - 6.91473 \cdot 10^{-4} \cdot T) \cdot 1000 \tag{5.3}$$

$$\rho_o = \frac{1}{0.7979 \cdot 10^{-3} + 0.7647 \cdot 10^{-6} \cdot T[K]} \tag{5.4}$$

The correlation used to estimate the density of the oil used in this work (Emkarate RL 32-3MAF) was provided directly by the POE oil manufacturer and cannot be disclosed. However, Kedzierski et al. (2017) experimentally developed a correlation for the same kind of oil (see Equation 5.5):

$$\rho_o = \frac{1}{0.7972 \cdot 10^{-3} + 0.2003 \cdot 10^{-3} \cdot T[K]} \tag{5.5}$$

Lubricant Kinematic Viscosity

Correlations to estimate the kinematic viscosity of ISO VG 68 POE oil were suggested by various authors. Equations 5.6 and 5.7 reported here were respectively used by Hu et al. (2008a) and Kedzierski (2013).

$$\nu_o = 1062.075 \cdot e^{-T/32.39} + 4.90664 \tag{5.6}$$

$$\nu_o = exp(-52.1912 + \frac{58.8418}{T[K]/273.15} + 36.8165 \cdot \ln\left(\frac{T[K]}{273.15}\right) \tag{5.7}$$

The correlation used to estimate the kinematic viscosity of the oil used in this work (Emkarate RL 32-3MAF) was provided directly by the POE oil manufacturer and cannot be

disclosed. However, Kedzierski et al. (2017) experimentally developed a correlation for the same kind of oil (see Equation 5.8):

$$\nu_o = exp(-45.0487 + \frac{50.5360}{T[K]/273.15} + 31.9522 \cdot \ln\left(\frac{T[K]}{273.15}\right) \tag{5.8}$$

Lubricant Surface Tension

Oil surface tension was estimated with a correlation used by Hu et al. (2008b) for a ISO VG 68 POE oil. The correlation is reported here in Equation 5.9:

$$\sigma_o = 29 - 0.4\frac{T}{1000} \tag{5.9}$$

Lubricant Specific Heat

The correlation used to estimate specific heat of oil was recommended by Thome (1995) as a function of the oil specific gravity (s_g = ρ_o/ρ_{H_2O}) at 15.6°C. The Equation is applicable for a range of temperatures between -18°C and 204°C, and it reported in Equation 5.10:

$$c_{p,o} = 4.186\frac{0.388 + 0.00045 \cdot (1.8 \cdot T + 32)}{\sqrt{s_g}} \tag{5.10}$$

Lubricant Specific Enthalpy

From the integration of Equation 5.10, Lottin et al. (2003) proposed a correlation to estimate the specific enthalpy of oil. The correlation was used in this work and it is reported in Equation 5.11:

$$h_o = 4.186\frac{0.4024 \cdot T + 0.000405 \cdot T^2}{\sqrt{s_g}} \tag{5.11}$$

5.1.2 Refrigerant and Lubricant Mixture Properties

Liquid mixture properties were estimated as a function of the local oil mass fraction, calculated as in Equation 5.12:

$$\omega_{local} = \frac{m_{oil}}{m_{oil} + m_{ref,L}} = \frac{OMF}{1 - x_{mix}} \tag{5.12}$$

where $x_{mix} = x(1 - OMF)$.

OMF represents the oil mass fraction mixed with the refrigerant, and it is equivalent to:

$$OMF = \omega = \frac{m_{oil}}{m_{oil} + m_{ref}} \tag{5.13}$$

Refrigerant and Lubricant Conductivity

The refrigerant-lubricant mixture thermal conductivity was calculated with the expression proposed by Filippov and Novoselova (1955) and represented in Equation 5.14:

$$k_{mix} = k_{ref,L}(1 - \omega_{local}) + k_o\omega_{local} - 0.72(k_o - k_{ref,L})(1 - \omega_{local})\omega_{local} \tag{5.14}$$

Refrigerant and Lubricant Density

The correlation used to estimate the density of a refrigerant-lubricant mixture was taken from Jensen and Jackman (1984) and reported in Equation 5.15:

$$\rho_{mix} = \frac{1}{\frac{\omega_{local}}{\rho_o} + \frac{1 - \omega_{local}}{\rho_{ref,L}}} \tag{5.15}$$

Refrigerant and Lubricant Dynamic Viscosity

The correlation proposed by Yokozeki (1994) was used to calculate the dynamic viscosity of a mixture of oil and refrigerant. The correlation is reported in Equation 5.16, where ψ represents the mole fraction of each component, M represents the molecular mass, and ξ represents the modified component mole fraction according to Yokozeki. The value of the constant k was suggested to be equal to 0.58.

$$
\begin{aligned}
\psi_o &= \frac{\omega_{local}\frac{MM_{ref}}{MM_o}}{1-\omega_{local}+\omega_{local}\frac{MM_{ref}}{MM_o}},\\
\psi_{ref} &= 1-\psi_o,\\
\xi_o &= \frac{MM_o^k\psi_o}{MM_{ref}^k\psi_{ref}+MM_o^k\psi_o},\\
\xi_{ref} &= \frac{MM_{ref}^k\psi_{ref}}{MM_{ref}^k\psi_{ref}+MM_o^k\psi_o},\\
\mu_{mix} &= exp(\xi_{ref}\ln\mu_{ref,L}+\xi_o\ln\mu_o)
\end{aligned}
\tag{5.16}
$$

Refrigerant and Lubricant Surface Tension

The refrigerant-lubricant mixture correlation for surface tension is given by Jensen and Jackman (1984) and it shown in Equation 5.17:

$$\sigma_{mix} = \sigma_{ref,L} + (\sigma_o - \sigma_{ref,L})\sqrt{\omega_{local}} \tag{5.17}$$

Refrigerant and Lubricant Specific Heat

Equation 5.18 reports the correlation proposed by Jensen and Jackman (1984) for the specific heat of a refrigerant-lubricant mixture:

$$c_{p,mix} = \omega_{local}c_{p,o} + (1-\omega_{local})c_{p,ref,L} \tag{5.18}$$

Refrigerant and Lubricant Bubble Temperature

Given a saturation pressure and an oil mass fraction, the bubble temperature of a refrigerant-lubricant mixture was estimated with the empirical vapor pressure correlation verified by Thome (1995) and reported here in Equation 5.19:

$$
\begin{aligned}
T_{bub}[K] &= \frac{AA}{\ln \frac{P_{sat}}{1000000} - BB} \\
AA &= a_0 + a_1\omega_{local} + a_2\omega_{local}^3 + a_3\omega_{local}^5 + a_4\omega_{local}^7 \\
BB &= b_0 + b_1\omega_{local} + b_2\omega_{local}^3 + b_3\omega_{local}^5 + b_4\omega_{local}^7
\end{aligned}
\tag{5.19}
$$

where the constants had the following values: a_1 = 182.52; a_2 = -724.21; a_3 = 3868.0; a_4 = -5268.9; b_1 = -0.72212; b_2 = 2.3914; b_3 = -13.779; b_4 = 17.066. The values of a_0 and b_0 are specific to each refrigerant and they are computed following the procedure described in chapter 15 of Thome (2004).

Refrigerant and Lubricant Bubble Temperature According to Sawant

Sawant et al. (2007) proposed a correction of the saturation temperature of refrigerant R410A for a measured absolute pressure. The formula is presented in Equation 5.20. The constants A_1 and A_2 and the polynomials a and b, accounted for deviations of the refrigerant R410A composition used in the present work from the standard blend properties due to (i) the presence of stray impurities as result manufacturing processes, and (ii) the presence of oil in the liquid refrigerant.

$$
T_{bub}[K] = \frac{AA}{\ln \frac{P_{sat}}{1000} - BB + \frac{A_2}{A_1}x} \tag{5.20}
$$

In this work, the expressions for AA and BB are the same as reported in Equation 5.19, except the constants changed for the data set reported in Appendix B. For test series from 1 to 28 the coefficients calculated are: A_1 = -0.000605742908; A_2 = -4.20111057 $\cdot 10^{-6}$; a_0 = -1650.949; b_0 = 12.79103. For test series from 29 to 63 the coefficients calculated are: A_1 = -0.000128240216; A_2 = -4.20111057 $\cdot 10^{-6}$; a_0 = -5719.029; b_0 = 27.45913.

Refrigerant and Lubricant Enthalpy

The refrigerant-lubricant mixture change in enthalpy during evaporation or condensation is represented by the heat release enthalpy curve, also discussed by Thome (1995). The formula

is shown in Equation 5.21 and in case of a pure refrigerant, it reduces to the latent heat of vaporization.

$$dh = h_{LV}(x_{mix,out} - x_{mix,in}) + (1 - x_{mix})c_{p,mix}(T_{bub,out} - T_{bub,in}) + x_{mix}c_{p,g}(T_{bub,out} - T_{bub,in}) \tag{5.21}$$

5.1.3 Nanoparticle Properties

Nanoparticle Conductivity

The thermal conductivity of Al_2O_3 nanoparticle was estimated as presented by Morrell (1987) and reported in Equation 5.22:

$$k_{nano} = 5.5 + 34.5e^{-0.0033 \cdot T} \tag{5.22}$$

Nanoparticle Density

The value of density of the Al_2O_3 nanoparticles used in this work was provided directly by the nanolubricant manufacturer (Sarkas, 2014):

$$\rho_{nano} = 3600 \ \ [kg/m^3] \tag{5.23}$$

Nanoparticle Specific Heat

The specific heat of Al_2O_3 nanoparticle was obtained from the work by Touloukian (1970) and reported in Equation 5.24:

$$c_{p,nano} = 1.0446 + 0.0001742 \cdot T - 27960 \cdot T^{-2} \tag{5.24}$$

5.1.4 Nanolubricant Properties

The thermodynamic properties of the nanolubricant were calculated as a function of the nanoparticles volume concentration ϕ, defined as:

$$\phi = \frac{NMF}{NMF + (1 - NMF)\,\frac{\rho_{np}}{\rho_{oil}}} \tag{5.25}$$

where NMF is the nanoparticle mass fraction in oil , and it is equal to:

$$NMF = \frac{M_{np}}{M_{np} + M_{oil}} \tag{5.26}$$

Recent works by Kedzierski (2013) and Kedzierski et al. (2017) investigated viscosity, density and thermal conductivity of polyolester-based Al_2O_3 nanoparticle dispersions, using POE oils of different viscosities (ISO VG 32 and 68). The presence of a surfactant used as nanoparticle dispersant was also studied and its impact on thermophysical properties was modeled in the set of correlations provided. Kedzierski (2013) observed that for quasi-spherical nanoparticles of 60 nm, the surfactant had different effects at different temperature ranges: between 300 K and 318 K the nanolubricant viscosity increased with an increase in surfactant mass fraction; between 288 K and 300 K the opposite behavior was observed. In the later work (Kedzierski et al., 2017), the correlations developed were not function of the nominal surface-area based nanoparticle diameter (about 60 nm), but of the number-based equivalent diameter measured with DLS. In this work, the equivalent diameter of Al_2O_3 nanoparticles measured with DLS was 126.7 nm (see Section 4.4).

Nanolubricant Conductivity

The thermal conductivity of a Al_2O_3 nanolubricant was estimated with the correlation suggested by Kedzierski et al. (2017) and reported in Equation 5.27. Equation 5.27 is very similar to the one formulated by Maxwell (1881), but modified by the use of a "sphericity" parameter (φ) that makes the correlation applicable also to non-spherical particles.

$$k_{nl} = k_o \frac{k_{nano} + (\frac{3}{\varphi} - 1)k_o - (\frac{3}{\varphi} - 1)\phi(k_o - k_{nano})}{k_{nano} + (\frac{3}{\varphi} - 1)k_o + \phi(k_o - k_{nano})} \tag{5.27}$$

Nanolubricant Density

Kedzierski et al. (2017) developed a model to describe Al_2O_3 nanolubricant and the correlation is reported in Equation 5.28:

$$\rho_{nl} = \frac{1}{\frac{x_s}{\rho_s} + \frac{x_{nano}}{\rho_{nano}} + \frac{x_o}{\rho_o}} \tag{5.28}$$

where the subscript "s" stands for "surfactant". However it is not clear why, but the implementation of this model in the simulation developed for this work, was not successful. Instead, the model by Pak and Cho (1998) (shown in Equation 5.29) was able to replicate the experimental data reported by Kedzierski et al. (2017).

$$\rho_{nl} = (1 - \phi)\rho_o + \phi\rho_{nano} \tag{5.29}$$

Nanolubricant Kinematic Viscosity

Al_2O_3 nanolubricant kinematic viscosity was also described by Kedzierski et al. (2017) and reported here in Equation 5.30:

$$\nu_{nl} = exp(x_o^{1.25} \ln \nu_o + x_{nano}^{1.25} \ln \nu_{nano} + x_s^{1.25} \ln \nu_s) \tag{5.30}$$

where the subscript "s" stands for "surfactant". The correlation by Batchelor (1977) for dynamic viscosity was also investigated in this work.

Nanolubricant Surface Tension

In this work, the surface tension of a Al_2O_3 nanolubricant was assumed to be equivalent to the surface tension of pure POE oil (see Equation 5.9).

$$\sigma_{nl} = \sigma_o \tag{5.31}$$

Nanolubricant Specific Heat

The correlation proposed by Murshed (2011) was utilized to estimate the specific heat of a Al_2O_3 nanolubricant. The correlation is reported in Equation 5.32:

$$c_{p,nl} = \frac{\phi\rho_{nano}c_{p,nano} + (1-\phi)\rho_o c_{p,o}}{\phi\rho_{nano} + (1-\phi)\rho_o} \tag{5.32}$$

5.1.5 Refrigerant and Nanolubricant Mixture Properties

The correlations used in this work to describe the thermophysical properties of a mixture of refrigerant and Al_2O_3 nanolubricant are similar to the correlation presented earlier for mixtures of refrigerant and pure POE oil. Similarly, the liquid mixture properties were estimated as a function of the local nanolubricant mass fraction, calculated as in Equation 5.33:

$$\omega_{nano,local} = \frac{m_{nanooil}}{m_{nanooil} + m_{ref,L}} = \frac{OMF_{nano}}{1 - x_{nanomix}} \tag{5.33}$$

where $x_{nanomix} = x(1 - OMF_{nano})$.

OMF_{nano} represents the nanolubricant mass fraction mixed with the refrigerant, and it is equivalent to:

$$OMF_{nano} = \omega_{nano} = \frac{m_{nanooil}}{m_{nanooil} + m_{ref}} \tag{5.34}$$

Refrigerant and Nanolubricant Conductivity

The refrigerant and nanolubricant mixture thermal conductivity was estimated with the correlation presented by Filippov and Novoselova (1955) and reported in Equation 5.35:

$$k_{nanomix} = k_{ref,L}(1-\omega_{nano,local}) + k_{nl}\omega_{nano,local} - 0.72(k_{nl} - k_{ref,L})(1-\omega_{nano,local})\omega_{nano,local} \tag{5.35}$$

Refrigerant and Nanolubricant Density

The density of a refrigerant and nanolubricant mixture was calculated with the correlation from Jensen and Jackman (1984), represented in Equation 5.36:

$$\rho_{nanomix} = \frac{1}{\frac{\omega_{nano,local}}{\rho_{nl}} + \frac{1 - \omega_{nano,local}}{\rho_{ref,L}}} \tag{5.36}$$

Refrigerant and Nanolubricant Dynamic Viscosity

The correlation used to estimate the dynamic viscosity of a mixture of refrigerant and nanolubricant was taken from Kedzierski and Kaul (1998), presented here in Equation 5.37:

$$\mu_{nanomix} = exp(\omega_{local,nano} \ln \mu_{nl} + (1 - \omega_{local,nano}) \ln \mu_{ref,L}) \tag{5.37}$$

Refrigerant and Nanolubricant Surface Tension

The refrigerant-nanolubricant mixture correlation for surface tension is the same found in the work by Jensen and Jackman (1984) for refrigerant-oil mixtures. The correlation is shown in Equation 5.38:

$$\sigma_{nanomix} = \sigma_{ref,L} + (\sigma_{nl} - \sigma_{ref,L})\sqrt{\omega_{local,nano}} \tag{5.38}$$

Refrigerant and Nanolubricant Specific Heat

Equation 5.39 reports the correlation used in this work for the calculation of the specific heat of a refrigerant-nanolubricant mixture (Jensen and Jackman, 1984).

$$c_{p,nanomix} = \omega_{local,nano} c_{p,nl} + (1 - \omega_{nano,local}) c_{p,ref,L} \tag{5.39}$$

Refrigerant and Nanolubricant Bubble Temperature

In the case of a refrigerant-nanolubricant mixture, the bubble temperature and the enthalpy correction were calculated with the same correlations presented for refrigerant-oil mixtures (see Equations 5.19, 5.20, and 5.21).

5.1.6 Segment Inventory

The mass of each component (vapor and liquid refrigerant, oil, and nanoparticles) was calculated in each segment according to the following set of equations:

$$Vol_{seg} = \frac{\pi D_h^2}{4} L_{seg} \tag{5.40}$$

$$M_{ref,V,seg} = Vol_{seg} \alpha \rho_V \tag{5.41}$$

$$M_{ref,L,seg} = Vol_{seg} \left(1 - \alpha\right) \left(1 - \omega_{local}\right) \rho_L \tag{5.42}$$

$$M_{nl,seg} = Vol_{seg} \left(1 - \alpha\right) \omega_{local} \rho_L \tag{5.43}$$

$$M_{oil,seg} = M_{nl,seg}(1 - NMF_{nl}) \tag{5.44}$$

$$M_{np,seg} = M_{nl,seg} NMF_{nl} \tag{5.45}$$

$$V_{np} = \frac{\pi}{6} D_{np}^3 \tag{5.46}$$

$$N_{np,seg} = \frac{M_{np,seg}}{\rho_{np} V_{np}} \tag{5.47}$$

where α is the void fraction and NMF_{nl} is the nanoparticle mass fraction in oil only.

5.2 Segment-by-Segment Model Validation

The model developed in this work was validated against the experimental data presented by Deokar et al. (2016) for two-phase flow boiling (i) of refrigerant R410A, (ii) of refrigerant and POE oil mixtures at 3% oil mass fraction, and (iii) of refrigerant and Al_2O_3 nanolubricant mixtures with oil mass fraction of 3% and nanoparticle mass concentration in the lubricant of 10 and 20% (that corresponds to a nanoparticle volume concentration in oil of about 2.6 and 5.8%).

Data were for a horizontal 9.5 mm micro-fin tube evaporator with hydraulic diameter of 5.45mm. The refrigerant saturation temperature varied from 3.1°C to 4.0°C, the mixture mass flux varied between 180 kg/m^2s and 425 kg/m^2s and tube heat flux ranged from 12 kW/m^2 to 15 kW/m^2. It is important to point out that the experimental uncertainty on the data of heat transfer coefficient ranged from 4 to 11% and the uncertainty on the pressure drop data ranged from 9 to 16%. This uncertainty should be considered when comparing the predicted pressure drops and heat transfer coefficients against the experimental data. The experimental data set collected during the experimental campaign and used here for validation is reported in Appendix B.

5.2.1 Experimental Validation of the Pressure Drop Models

The simulation predictions for two-phase flow pressure drop in a microfin tube are reported in Figure 5.1. For refrigerant R410A and for refrigerant and lubricant mixture, Figure 5.1 reports the simulation results obtained with the correlation by Choi et al. (1999) (blue solid circles) and with the correlation by Hu et al. (2008b) (orange solid triangles). Figure 5.1 also reports the results of the application of the two correlations to the cases with refrigerant and nanolubricant mixture (red solid squares and green solid diamonds). The comparison with the experimental data showed that both the correlations from the literature underpredicted the experimental data. The refrigerant R410A was underpredicted by up to -40%.

For the refrigerant and POE oil, the correlation by Choi et al. (1999) calculated the total pressure drop and was designed for blends of refrigerants and refrigerant and lubricant mixtures

flowing through a microfin tube with outside diameter of 9.52 mm. Their tube geometry was similar to the one used by Deokar et al. (2016) and in this work. However, there was a lack of specific information about the specific properties of the particular POE lubricant used in the work of Choi et al. and these properties were estimated in this work considering a general ISO VG 32 POE lubricant.

Additives and surfactants used in the specific POE lubricant might change some of its properties and could lead to significant variations of the predicted pressure drops from the present model. For this reason, a sensitivity study was performed and will be presented later in this work. In the sensitivity study, the viscosity of the base lubricant was purposely varied to up to 25% higher than what is generally estimated for ISO VG 32 POE lubricant in order to investigate the impact of lubricant viscosity on the predicted pressure drop and heat transfer coefficient of the refrigerant and oil mixture during flow boiling. As shown in Figure 5.1, and as also pointed out by the original authors of the correlation of Choi et al. (1999), the Choi et al. correlation seemed to underpredict the two-phase flow boiling pressure drop of refrigerant and lubricant mixtures, and the error was up to -50%. Similar findings were observed in the work by Hu et al. (2008b) who proposed a new vapor-phase multiplier correlation of frictional pressure drop for boiling mixture of R410A/lubricant flowing inside a microfin tube with a 7 mm outside diameter. They observed higher pressure drops with increasing oil mass fractions and mass fluxes. The oil used was slightly more viscous than the one used in this work. In their work, Hu et al. reported a maximum deviation of their correlation of 15% and their correlation provided better predictions, that is, within -20%, of the experimental data for refrigerant and refrigerant and oil mixture reported in Figure 5.1.

It should be noted that in the present model, the momentum pressure drop was calculated by using the void fraction correlation by Rouhani and Axelsson (1970) and the pressure drop correlations were implemented using the thermodynamic properties of refrigerant and lubricant and refrigerant and nanolubricant mixtures described in Section 5.1.

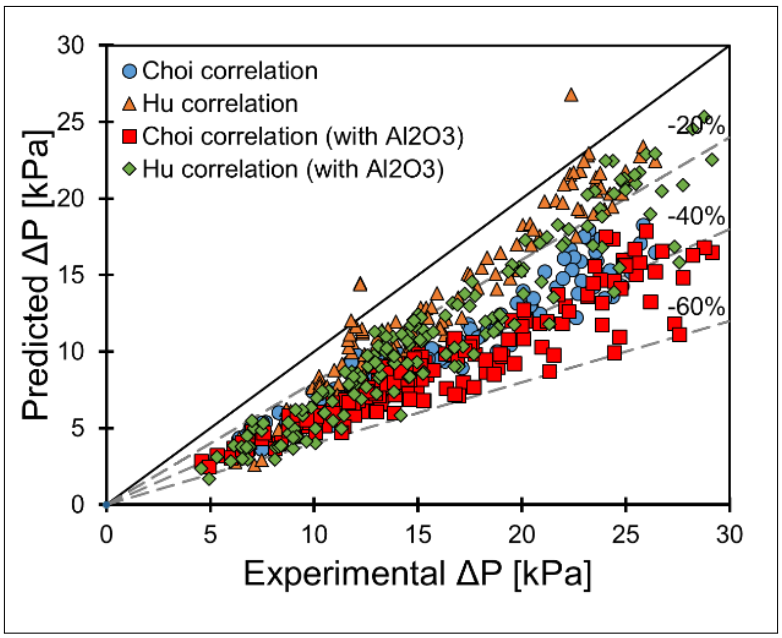


Figure 5.1: Comparison of predicted pressure drops (ΔP) vs. experimental data.

5.2.2 Experimental Validation of the Heat Transfer Coefficient Models

The comparison between the experimental data of heat transfer coefficients taken from Deokar et al. (2016) paper and the predicted two-phase flow heat transfer coefficients from the model developed in this work are summarized in Figure 5.2. For refrigerant R410A and refrigerant and lubricant mixture, Figure 5.2 reports the simulation results obtained with the correlation by Hamilton et al. (2008) (blue solid circles) and with the correlation by Hu et al. (2008a) Hu et al. (2008a) (orange solid triangles). Figure 5.2 also reports the simulation results of the application of the two correlations to the cases with refrigerant and nanolubricant mixture (red solid squares and green solid diamonds). The simulation results were able to predict most of the experimental data within 40%. If refrigerant R410A was modeled, then the heat transfer coefficients were predicted with an uncertainty of 20% for Hamilton et al. correlation and of 30% for Hu et al. correlation. These uncertainties were consistent with the ones reported in the original studies from which the correlations were developed.

The correlation by Hamilton et al. (2008) described flow boiling of refrigerants and refrigerants blends inside a horizontal microfin tube. This correlation was built upon the theory of the law of corresponding states and it is only applicable for mass fluxes between 70 kg/m^2s and

370 kg/m^2s and for a quality range of 0 to 0.7. The work by Sawant et al. (2007) proved the applicability of Hamilton et al. correlation to mixtures of R410A and POE oil with 20% error. The oil used in their work had about same viscosity as the one used for this work. The same authors also stated that the relative heat transfer coefficient of the R410A and POE mixture ranged from 20% up to +42% compared to that of refrigerant R410A only heat transfer coefficient. Hu et al. (2008a) developed another correlation to describe the flow boiling of R410A and lubricant mixtures in a microfin tube with a 7 mm outside diameter. Their correlation accounted for both convective and nucleate boiling contributions to the heat transfer and was validated with a deviation from experimental data of 30%. The oil used in their experiments was slightly more viscous than the one used for this work. Although the correlation for kinematic viscosity used in their heat transfer correlation provided values of kinematic viscosity that are almost one order of magnitude higher with respect to other sources. Hu et al. (2008a) observed that for qualities lower than 0.4, the heat transfer was enhanced in presence of oil, while for qualities higher than 0.65, the heat transfer decreased drastically.

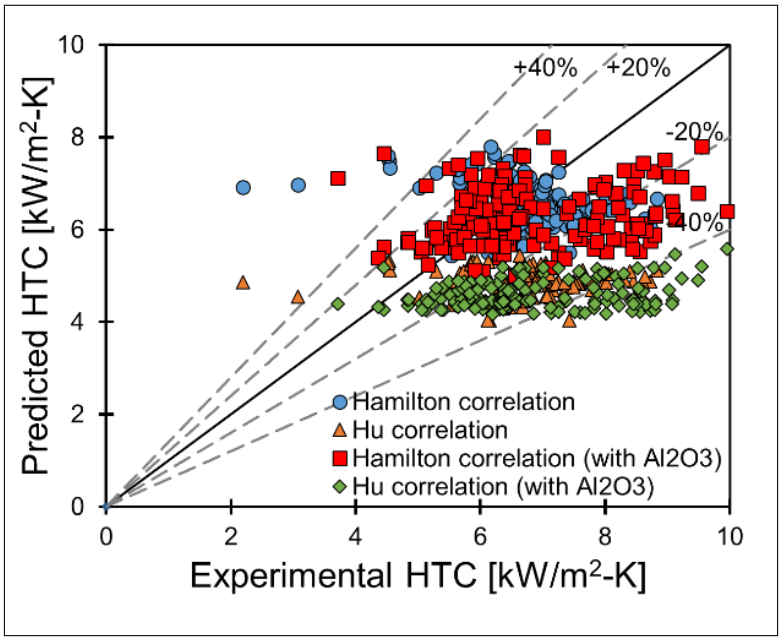


Figure 5.2: Comparison of predicted heat transfer coefficients (HTC) vs. experimental data.

5.2.3 Simulation Results and Discussion

Figures 5.3 and 5.4 show the pressure drop and heat transfer coefficients for the case of 250 kg/m^2s mass flux and 0 to 3% oil mass fraction, with and without nanoparticles. The plots are given with refrigerant thermodynamic quality on the x-axis and for both heat transfer and pressure drop correlations used in this work. Error bars are reported for only one data point but are representative of the uncertainty of all data points (see also Figure 4.13 and Section 4.3).

Different series of experimental data are also reported, showing the behavior of the mixtures when the quality increases. The experimental series reported are for the following refrigerant mixtures: at 0% POE -oil-free case- (in blue solid circles); at 3% POE (in green solid triangles); at 10 and 20% nanoparticle mass concentration in 3% POE oil (respectively, in purple solid squares and red solid diamonds). The predicted results are summarized by blue and orange solid lines for the oil-free cases. For the cases with POE oil and nanolubricants the predicted results are on the top of the oil-free case solid lines, that is, the predicted pressure drops and heat transfer coefficients when oil and nanolubricants were present did not vary appreciably to be distinguished in Figures 5.3 and 5.4 as separated individual lines.

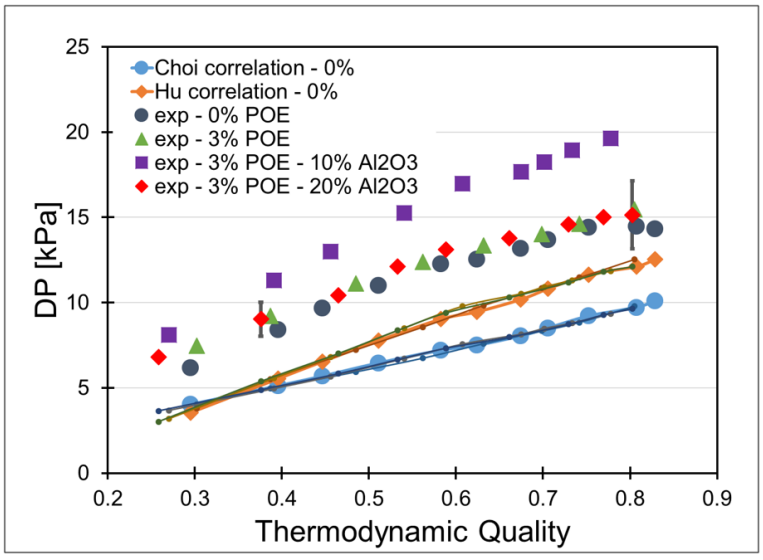


Figure 5.3: Experimental and simulation trends of different refrigerant/lubricant mixtures for pressure drop.

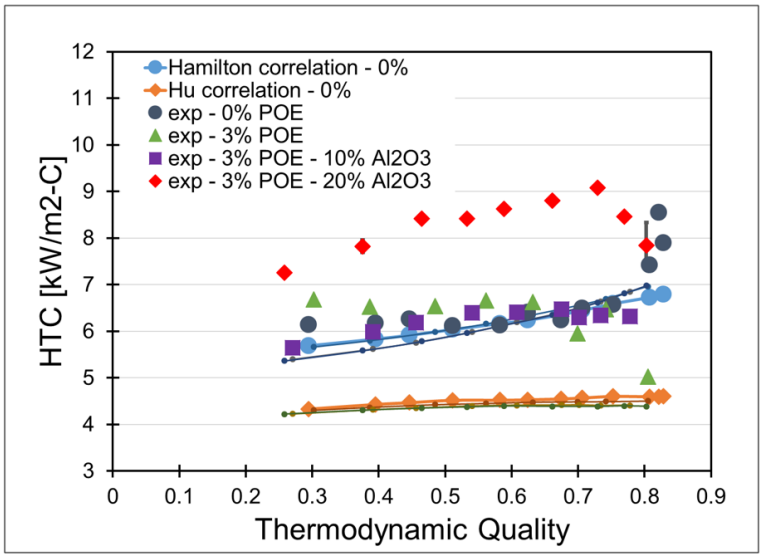


Figure 5.4: Experimental and simulation trends of different refrigerant/lubricant mixtures for heat transfer coefficient.

Table 5.1 shows the results of a sensitivity analysis of the existing correlations used to estimate kinematic viscosity and thermal conductivity, whose estimated values were varied by 25% in a parametric fashion. The error was calculated as difference of the simulation results minus the experimental data, in percentage, and for two representative qualities. The comparison was conducted for the case of refrigerant-nanolubricant mixture at 3% POE oil OMF and 20% nanoparticles concentration in oil (see row 1 in the Table 5.1). The variation of the nanolubricant kinematic viscosity and thermal conductivity by 25% did not decrease the error, as shown in rows 2 and 3. A slight reduction of few percentages was observed for the predicted heat transfer coefficients at quality of 0.75, as indicated in the last column of row 3.

While a variation of the refrigerant R410A and nanolubricant mixture kinematic viscosity had small effects, an increase of thermal conductivity of the refrigerant R410A and nanolubricant liquid phase mixture of +25% increased the predicted heat transfer coefficient significantly, and reduced the error to 6 and 15%, as shown in row 5 of Table 5.1. The thermal conductivity correlation developed by Kedzierski et al. (2017) for nanolubricant considered the effect of surfactant and was designed for the cases of aluminum oxide and zinc oxides dispersed in the same POE oil used in this work. However, for the case of an evaporative two-phase flow, this correlation was still not able to provide the increase of thermal conductivity in the mixture liquid phase required to support the measured heat transfer enhancement. This observation

suggests that other phenomena at nanoscale level might occur and contribute to the variation of the two-phase flow heat transfer coefficient.

Table 5.1: Sensitivity analysis for kinematic viscosity and thermal conductivity.

		x = 0.5		x = 0.75	
		DP error [%]	HTC error [%]	DP error [%]	HTC error [%]
1	R410A - 3% POE - 20% Al2O3	-45.1	-29.1	-38.1	-19.5
2	Nanolubricant ν of +25% / -25%	-45.0 / -45.1	-29.2 / -29.0	-38.0 / -38.2	-19.5 / -19.5
3	Nanolubricant k_{th} of +25% / -25%	-45.1 / -45.1	-28.8 / -29.5	-38.1 / -38.1	-18.8 / -20.1
4	R410A-nanolubricant ν of +25% / -25%	-43.9 / -46.4	-30.8 / -26.9	-36.9 / -39.7	-19.3 /-19.7
5	R410A-nanolubricant k_{th} of +25% / -25%	-45.1 / -45.1	**-15.2** / -43.8	-38.1 / -38.1	**-6.0** / -34.0

5.2.4 Discussion of the Simulation Results for Two-Phase Flow Pressure Drop of Nanolubricants

For the case of 250 kg/m^2s mass flux and 0 to 3% oil mass fraction, Figure 5.5 shows that the pressure drop tended to increase if the quality increases. The lubricant had over 10 times higher viscosity than liquid refrigerant and thus it significantly increased the viscosity of the refrigerant/lubricant mixture liquid phase. This generally resulted in higher frictional pressure drops of the refrigerant and lubricant mixture compared to refrigerant only. However, Figure 5.5 shows that for both the simulation results of the present model and the experimental data used to verify the model, the pressure drop penalization due to the presence of oil was small.

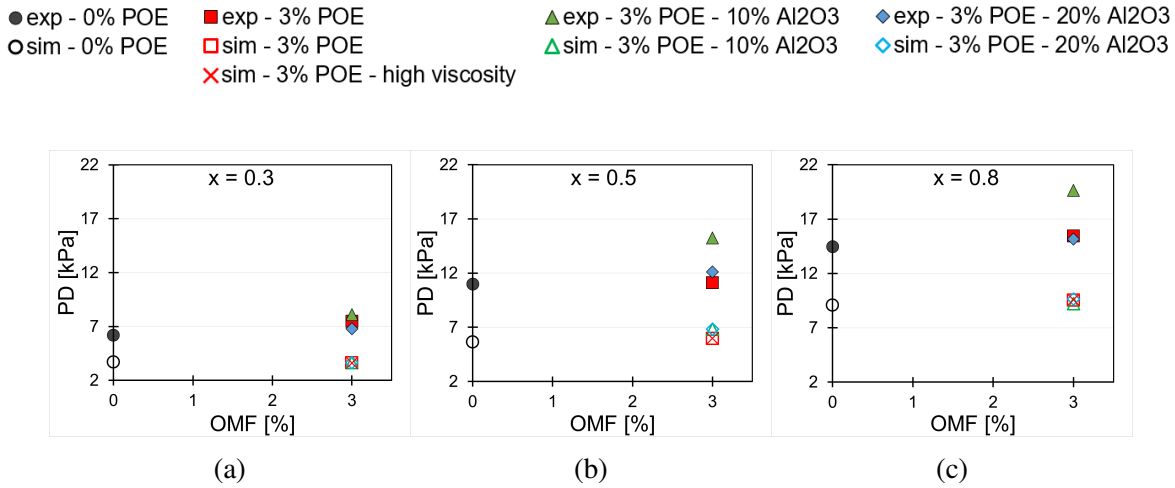
The simulations results indicated that at 3% OMF, both POE lubricant and Al_2O_3 nanolubricant had estimated pressure losses that were just slightly higher than that of refrigerant R410A. The data showed similar trends for POE, while higher pressure drop were measured for the Al_2O_3 nanolubricant at medium quality (see Figure 5.5(b)) and high quality (see Figure 5.5(c)). An increase of the frictional losses became evident only at higher qualities, as shown in Figure 5.5(c). Similar findings were also observed in the literature (Nidegger et al., 1997; Zuercher et al., 1998).

According to the aforementioned correlations for mixtures of nanofluids and assuming that the nanoparticles remained well dispersed in the POE and refrigerant mixture liquid phase, the nanolubricants must have higher viscosity than that of liquid refrigerant and POE oil mixture. Thus, the highest pressure losses were expected for the 3% POE oil OMF and 10% and 20% Al_2O_3 nanoparticle concentration case in Figure 5.5. This was more or less the case in the experimental data. The work of Deokar et al. (2016) confirmed that at low quality the pressure losses of lubricant and nanolubricant were very close to each other while nanolubricants tended to have slightly higher pressure losses at medium and high qualities, as shown by the solid red square data points for POE at 3% OMF experimental data with respect to the solid green triangles data points for the Al_2O_3 based nanolubricant at 3% OMF and 10% nanoparticle mass concentration. The model predicted this trend well at medium quality while at both low and high qualities the difference of the pressure drop between POE oil at 3% OMF case (see void red square simulation results points with the sim 3% POE legend in Figure 5.5) and Al_2O_3 based nanolubricant at 3% OMF and 10% nanoparticle mass concentration case (see void green triangles simulation results with the sim 3% POE - 10% Al_2O_3 legend) were very small. Similar observations could be made for the case of Al_2O_3 based nanolubricant at 3% OMF and 20% nanoparticle mass concentration, at medium quality, where the solid blue diamond showed a slightly higher pressured drop than the solid red square of the 3% POE. The simulation pressure drop of the 20% nanolubricant case was slightly higher than the 10% nanolubricant case both at medium and high qualities.

The model seemed to capture trends similar to the experimental data, and closer to the 3% POE (comparison between void blue diamonds and void red squares). In order to investigate

these results, a sensitivity analysis of the viscosity was conducted by increasing the viscosity value up to 10 times (reported in Figure 5.5 as a red cross) for the case of 3% POE. Interestingly, the model did not seem to be affected by a higher viscosity as the new pressure drop indicated by the red cross did not move from the void red square of the base 3% POE case. More recent investigations on pool boiling of non-Newtonian fluids and Al_2O_3 nanolubricants (Soltani et al., 2010; Kedzierski, 2011) showed how even a 1.4 to 1.6% nanoparticle volume fraction can drastically enhance the heat transfer of the base fluid, thank to the interaction of the nanoparticles with the bubbles formation process. However, other works on nanofluids also observed a share-rate dependency of the viscosity, arguing the possibility of a transition from a Newtonian to a non-Newtonian behavior (Venerus et al., 2010). Aladag et al. (2012) studied nanofluids with nanoparticles of different shapes and reported a shear-thickening behavior for Al_2O_3-water nanofluid over a wide range of shear rates and for temperatures between 2°C and 10°C.

The pressure drop correlations used in this work lack information on the change of the fluid behavior when nanoparticles are added, as well as a dependency from the flow rate and the nanoparticles material, shape, size and dispersion. It might be possible that a similar situation to the one described by Aladag et al. (2012) is occurring for the flow regime of this work and this aspect requires further investigation in future follow up research of this work.



(a) (b) (c)

Figure 5.5: Pressure drop for (a) low quality, (b) medium quality and (c) high quality of different refrigerant-lubricant mixtures at test conditions of 250 kg/m^2s and 12 W/m^2 (the simulation data in this figure were obtained from application of Choi et al. (1999) correlation)

5.2.5 Discussion of the Simulation Results for Two-Phase Flow Heat Transfer Coefficients of Nanolubricants

For the case of 250 kg/m2-s mass flux and 0 (refrigerant only) to 3% oil mass fraction, the experimental results by Deokar et al. (2016) in Figure 5.6 show that the oil-free case slightly increased heat transfer coefficient if the quality increase from 0.3 (low quality in Figure 5.6(a)) up to 0.8 (high quality in Figure 5.6(c)). For 3% oil mass fraction, the heat transfer coefficient was higher than the oil-free case at lower and medium qualities, but it dropped at higher qualities. This behavior was unexpected but similar to what observed in the experimental work of Hu et al. (2008a). The review paper by Bandarra Filho et al. (2009) on flow boiling of refrigerant/lubricant mixtures reported other literature studies for microfin tubes where the presence of oil increased the heat transfer coefficient. Compared to the liquid phase of most refrigerants, generally lubricants have lower density (that, at constant mass flux, can increase the fluid velocity and promote more uniform mixture), higher thermal conductivity, higher specific heat, higher surface tension (increasing the wettability), higher bubble temperature and higher viscosity, which greatly affects both pressure drop and heat transfer, especially at higher qualities. Oil might induce some foaming at the liquid-vapor interface.

The internal geometry of a microfin tube also affects the flow patterns, promoting annual type flow regime. The effect of these phenomena on the heat transfer coefficient are not properly captured by the heat transfer correlations used in the present model, as shown by the discrepancy between simulation results for POE (void red squares) and nanolubricant (void green triangles) mixtures and experimental data (solid red square and solid green triangles) in Figure 5.6(a) and 5.5(b).

A sensitivity analysis of these results with respect to the mixture thermophysical properties, suggested that at higher qualities the increase in viscosity was much faster and it could affect greatly the Reynolds numbers used to estimate the heat transfer coefficients. Thus, a steeper increase of viscosity could lead to a sudden decrease of heat transfer coefficient. For the case of nanolubricants, even if nanoparticles enhance thermal conductivity, they could also

further increase the viscosity by promoting a shear-thickening behavior, typical of some non-Newtonian fluids. The existing viscosity models in the literature used for nanolubricants did not include non-Newtonian behaviors, which affect the flow development of the liquid phase of the mixture. The localized thickening and thinning of the liquid film thickness around the inner walls of the tube can alter the film local convective thermal resistance. This mechanism could explain the discrepancy between the simulation results of the present and the experimental data. However, this behavior was not properly captured by the existing two-phase flow boiling heat transfer coefficient correlations that were implemented in the present heat transfer model from the state-of-the-art literature and that are commonly used for predicting heat transfer performance of refrigerant and POE oil mixtures during flow boiling in micro-fin tubes. Similarly to what was observed for pressure drops case, the heat transfer correlations were not able to predict the nanolubricant behavior. Information on the change of the fluid flow behavior in presence of nanoparticles should be added in future work to the present model.

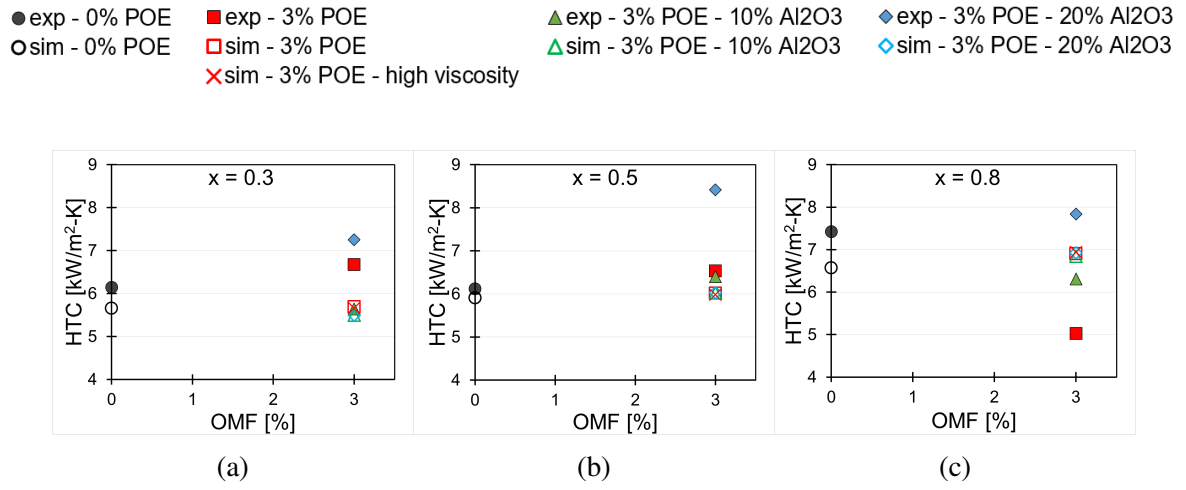


Figure 5.6: Heat transfer coefficient for (a) low quality, (b) medium quality and (c) high quality of different refrigerant-lubricant mixtures at test conditions of 250 $\frac{kg}{m2-s}$ and 12 $\frac{W}{m2}$ (the simulation data in this figure were obtained from application of Hamilton et al. (2008) correlation)

5.3 Fundamental Approach

Recently, the study of colloidal solutions and dispersed particles in a fluid flow became of interest because of experimental observations showing intriguing heat transfer enhancement. The understanding of these results promoted the investigation of different mechanisms governing

the particles behavior within the base fluid, with the intention to assess what is the particles specific contribution to the heat transfer phenomenon.

Researchers pointed out that the sole increase in thermal conductivity is not enough to justify the observed enhancement in convective heat transfer. Starting from the analysis conducted by Buongiorno (2006), different slip mechanisms were considered as possible contributions to the nanoparticles' behavior and interactions within the liquid layer: particle rotation and translation, inertia, Brownian diffusion, thermophoresis, diffusiophoresis, Magnus effect, fluid drainage, gravity. More authors conducted investigation using a similar approach (He and Ahmadi, 1998; Phillips et al., 1992; Hwang et al., 2009; Wen et al., 2009; Savithiri et al., 2011; Cremaschi, 2012; Mahdavi et al., 2017) and, while studies were reported on the nanoparticles lubrication effects (Ghaednia et al., 2016), no study was found on the thermal behavior of nanoparticle dispersions in high-viscosity fluids.

The following paragraphs investigate a number of mechanisms that the nanoparticles could be subjected to when dispersed in a liquid flow. The investigation is carried out with the purpose to understand which mechanism affects more the nanoparticle distribution, both for a laminar and a turbulent flow regime. The mechanisms analyzed are characteristic of the study of fouling and particle precipitation (Lister, 1980) and they are applied here to the case of nanoparticles in a two-phase flow.

5.3.1 Analysis of Slip Mechanisms

In this section, different slip mechanisms will be described and their relevance will be estimated for the case of Al_2O_3 nanoparticles dispersed in a mixture of refrigerant R410A and POE oil. In order to understand the relevance of each mechanisms, the "diffusion time", t_D, was calculated, that is, the time a particle takes to diffuse a length equivalent to its diameter, when affected by that mechanism ($D_{nano}/v_{mechanism}$). Because the magnitude of slip mechanisms can change depending on the fluid thermophysical properties and operational conditions (e.g. heat flux and mass flux), the analysis of the slip mechanisms was conducted for a set of conditions chosen from the experimental data, because naturally more representative to describe this specific study. The thermophysical properties were estimated using the simulation model developed

for this work and the set of tests were chosen in order to cover the largest range of conditions. In particular, the mechanisms were evaluated at the lower and higher mass fluxes tested (180 kg/m^2s and 425 kg/m^2s), at low and high oil mass fraction (1 and 3%), and at low and high thermodynamic quality (x ~ 0.2 and x ~ 0.8). Nanoparticle mass fraction (20%), saturation temperature (4°C) and heat flux (12 kW/m^2) were kept constant. Table 5.2 summarizes the thermophysical properties calculated for each set of conditions:

Table 5.2: Fluid thermophysical properties for different sets of operational conditions (at NMF = 20%, $T_{sat} = 4°C$, q" = 12 kW/m^2).

OMF	MASS FLUX [$\frac{kg}{m^2s}$]	x	$\mu\left[\frac{kg}{m-s}\right]$	$k\left[\frac{W}{m-K}\right]$	$\rho\left[\frac{kg}{m^3}\right]$	$c_p[\frac{J}{kg-K}]$
1%	180	0.18	$1.71{\cdot}10^{-4}$	0.1012	1151.76	1540.47
		0.79	$2.21{\cdot}10^{-4}$	0.1021	1148.97	1538.83
	425	0.11	$1.69{\cdot}10^{-4}$	0.1010	1151.22	1541.39
		0.79	$2.22{\cdot}10^{-4}$	0.1018	1146.82	1541.7
3%	180	0.14	$2.03{\cdot}10^{-4}$	0.1018	1150.54	1538.52
		0.76	$3.94{\cdot}10^{-4}$	0.1046	1142.44	1535.66
	425	0.10	$2.00{\cdot}10^{-4}$	0.1016	1149.06	1540.86
		0.75	$3.73{\cdot}10^{-4}$	0.1041	1141.76	1537.79

5.3.2 Continuum Assumption

A first assumption pointed out by Buongiorno (2006) was to consider the fluid surrounding the nanoparticles as a continuum. In order to justify this assumption, the Knudsen number, Kn, was calculated according to its definition:

$$Kn = \frac{\lambda}{D_{nano}} \tag{5.48}$$

where λ is the fluid molecule mean free path and D_{nano} is the nanoparticle diameter (in this work, equal to about $60nm$ for Al_2O_3). The fluid molecule mean free path is the average distance that a particle can travel between collisions with other moving particles and, when the

particles' velocities are assumed to have a Maxwell distribution, the formula is (Chapman and Cowling, 1970):

$$\lambda = \alpha \frac{1}{\pi D_{molecule}^2 n_V} \tag{5.49}$$

where $\alpha = 1/\sqrt{2}$ is a correction factor accounting for the molecules average relative velocity, $D_{molecule}$ is the molecular diameter, and n_V is the number of molecules per unit volume, calculated as:

$$n_V = \frac{N_A}{M_W/\rho} \tag{5.50}$$

being, N_A, the Avogadro number ($6.022 \cdot 10^{23}$), M_W, the molecular weight, and ρ, the fluid density. Table 5.3 reports the molecular formulas, M_W and ρ for both refrigerant R410A (a near-azeotropic blend (50/50%) of difluoromethane (CH_2F_2, called R-32) and pentafluoroethane (CHF_2CF_3, called R-125) and POE oil.

Table 5.3: Fluids chemical characterization.

FLUID	FORMULA	M_W	ρ	n_V
R410A	R125(50%)-R32(50%)	$72.6 g/mol$	$\sim 1.04 g/cm^3$	$8.626 \cdot 10^{21}$
	R125 - C_2HF_5	$120 g/mol$	$\sim 1.24 g/cm^3$	$6.223 \cdot 10^{21}$
	R32 - CH_2F_2	$52 g/mol$	$\sim 1.1 g/cm^3$	$1.274 \cdot 10^{22}$
POE oil	-	$570 g/mol$	$\sim 1 g/cm^3$	$1.056 \cdot 10^{21}$

The molecular diameter of refrigerant R410A and POE oil were calculated estimating the length of the molecular bonds (Table 5.4) and assuming a quasi-spherical shape.

Table 5.4: Carbon-to-element bond length (Weast, 1984).

ELEMENT BONDED	BOND LENGTH $\left[\mathring{A} = 10^{-1}nm\right]$
C	$\sim 1.2 - 1.54$
H	$\sim 1.06 - 1.12$
F	~ 1.34
O	$\sim 1.43 - 2.15$

Table 5.5 reports the estimates of the Knudsen number for both the refrigerant and the POE oil.

Table 5.5: Estimation of the Knudsen number.

FLUID	$D_{molecule}[cm]$	$\lambda[\frac{cm}{mol}]$	Kn
R125	$\sim$4·10^{-8}	$\sim$ 2.26·10^{-8}	$\sim$3.77 ·10^{-3}
R32	$\sim$2.68·10^{-8}	$\sim$ 2.46·10^{-8}	$\sim$4.1 ·10^{-3}
POE oil	$\sim$1.6·10^{-7}	$\sim$ 8.34·10^{-9}	$\sim$1.39 ·10^{-3}

In liquids, generally the molecule mean free path is of the order of the molecule diameter. The exact chemical composition of the ISO VG 32 POE oil used in this research is unknown, although the molecular structure of similar oils can sometimes be found in public domain literature (Matsuo and Itoh, 1998). In this analysis it was estimated that the diameter of POE oil molecule is still smaller than the nanoparticle diameter. Therefore, in this work, the Knudsen number was found to be less than one both with respect to refrigerant R410A and POE oil, confirming the assumption that the fluid around the nanoparticles is a continuum.

5.3.3 Particle Rotation and Translation

In the study of laminar flow of suspensions of polystyrene spheres (diameter range: $\sim 50 - 100\mu m$), Ahuja (1975b) observed that an increase in shear rate, particle concentration and size, and tube size induced an increase in the fluid effective thermal conductivity, without increasing the friction factor. At higher sphere concentrations of about $20wt\%$, Ahuja measured a flattening of the typical Poiseuille flow parabolic velocity profile. The author suggested that this result

is in agreement with the "σ phenomenon" referred to by Goldsmith et al. (1967) and differently called by different authors as either "hydrodynamic wall effect" or "mechanical wall effect" or "radial particles' migration". In all cases, the phenomenon was described as the formation of a particle-free low-viscosity layer near the tube wall because of the migration of the rigid spheres to the center of the flow. The development of a particle-free layer induces a lubricating effect that causes a drop in pressure gradient at the wall and a change in the velocity profile, exhibiting the characteristic of a plug flow, where the core flow has a higher viscosity than the liquid film surrounding it. Because of the flattening of the velocity gradient, the spheres stop rotating as it moves away from the wall. Based on this knowledge, Ahuja (1975a) proposed a theoretical model suggesting that the enhancement in the thermal conductivity of the flowing suspension might be the result of both a rotational (induced by the shear field) and a translational (radial migration by effect of the inertia) motion of particles.

Rotation

Under the effect of a shear stress, a spherical particle immersed in a continuum fluid starts rotating about an axis perpendicular to the shear stress. If the particle rotates fast enough, the inertia induces an outflow of fluid close to the particle "equator" and an inflow close to the particle "poles", creating a three-dimensional hydrodynamic boundary layer. The increase in thermal conductivity depends also on the kind of fluid and on the time, t_{diff}, the heat takes to diffuse in this fluid by conduction ($t_{diff} \sim D^2_{particle}/\alpha_{fluid}$). Therefore, depending on the fluid thermal diffusivity, α_{fluid}, an augmentation is expected if the particle rotation time, $1/\omega$ (where ω is the particle angular velocity), is of the same order of magnitude of t_{diff}, that is: $\frac{D^2_{particle}\omega}{\alpha_{fluid}} \sim 1$. The ratio $\frac{D^2_{particle}\omega}{\alpha_{fluid}}$ represents the Peclet number, Pe, which is the ratio between the heat transported by convection to the heat transported by conduction. Because the particle angular velocity varies linearly from the tube axis to the tube wall, Ahuja (1975a) suggested assuming ω to be of the same order of magnitude of the wall share rate ($\omega \sim \gamma/4$, where $\gamma = \tau_{wall}/\mu_{fluid}$). Therefore the Peclet number can be written as:

$$Pe = \frac{R^2_{particle}}{\alpha_{fluid}} \frac{\tau_{wall}}{\mu_{fluid}} \tag{5.51}$$

where τ_{wall} can be calculated according to Equation 5.52 for the single phase case.

$$\tau_{wall} = \frac{f}{8}\rho\overline{v}^2 \tag{5.52}$$

where f is the friction factor and $\overline{v}$ is the fluid average axial velocity. However, for the case of two-phase annular flow, τ_{wall} was calculated from a first estimate of the liquid-vapor interfacial shear stress $\tau_i = \frac{1}{2}f_i\rho_v u_{core}^2$, and more details on this calculations are provided in Section 5.4. The estimations of the rotation effect are reported in Table 5.6.

Table 5.6: Peclet number estimates.

OMF	MASS FLUX [$\frac{kg}{m^2 s}$]	x	τ_{wall} [Pa]	Pe
1%	180	0.18	0.346	$3.20{\cdot}10^{-5}$
		0.79	0.765	$5.39{\cdot}10^{-5}$
	425	0.11	0.682	$6.36{\cdot}10^{-5}$
		0.79	2.240	$1.58{\cdot}10^{-4}$
3%	180	0.14	0.318	$2.45{\cdot}10^{-5}$
		0.76	0.977	$3.74{\cdot}10^{-5}$
	425	0.1	0.714	$5.61{\cdot}10^{-5}$
		0.75	2.852	$1.16{\cdot}10^{-4}$

In all cases, the Peclet number is very small, meaning that spherical particles of 60 nm will not rotate. This result is in agreement with Ahuja (1975a) who concluded his study observing that rotation is less likely to start any time the fluid viscousity and thermal diffusivity are high, the tube diameter is small, and the particles are small.

Translation

Segr and Silberberg (1962) proposed the following empirical formula to estimate the radial velocity, u_{radial}, of neutrally buoyant particles in a Poiseuille flow moving from the wall towards an equilibrium position at about 0.6 radii from the tube axis ($r^\star/R_{tube} = 0.6$):

$$u_r = 0.34 \frac{2R_{tube}u_{fluid}^2}{\nu_{fluid}} \left(\frac{R_p}{R_{tube}}\right)^{2.84} \frac{r}{R_{tube}} \left(\frac{r^\star}{R_{tube}} - \frac{r}{R_{tube}}\right) \tag{5.53}$$

(valid for tube Reynolds number $Re \leq 30$).

In the present case study, the Reynolds number of the liquid film was generally larger than 30. Equation 5.53 was only used to estimate the translational velocity order of magnitude. The value of r was assumed to be equal to the thickness of the liquid film and the equilibrium position was assumed to be at about 0.6 of the same thickness. The results for the translational effect are reported in Table 5.7.

Table 5.7: Estimation of the translational velocity and corresponding diffusion time.

OMF	MASS FLUX [$\frac{kg}{m^2 s}$]	x	u_r [$\frac{m}{s}$]	t_D [sec]
1%	180	0.18	-2.91·10^{-13}	2.06·10^5
		0.79	-1.37·10^{-14}	4.37·10^6
	425	0.11	-1.98·10^{-12}	3.03·10^4
		0.79	-6.91·10^{-14}	8.68·10^5
3%	180	0.14	-2.71·10^{-13}	2.21·10^5
		0.76	-1.02·10^{-14}	5.91·10^6
	425	0.1	-1.66·10^{-12}	3.60·10^4
		0.75	-6.63·10^{-14}	9.05·10^5

where the negative value signifies that the velocity is directed from the wall, towards the center of the tube. The translational velocity is negligible and the diffusion time is very large, making this slip mechanism ineffective. This result is also in accordance with Ahuja (1975a).

5.3.4 Inertia

An aspect of interest in the study of the behavior of nanoparticles is to understand if it is reasonable to think that nanoparticles can move with respect to the fluid at a relative velocity. In the case of laminar regime, the nanoparticle is assumed to be undisturbed. However, in the case of turbulent flow, given that nanoparticles and the fluid have different densities, it could

be assumed that, in the presence of turbulent eddies, a nanoparticle motion could be disturbed and the nanoparticle could start moving at a relative or slip velocity, $\overline{u}$, with respect to the fluid, until the inertial force was dissipated and the nanoparticle relative velocity "relaxed" back to the velocity of the surrounding fluid, u_{eddies}. The relative or slip velocity can be estimated from a force balance between the second law of motion and the Stokes's law, representing the viscous drag force acting opposite to the relative motion of the nanoparticle:

$$\frac{\pi}{6} D_{nano}^3 \rho_p \frac{du}{dt} = -3\pi D_p \mu_{fluid} \overline{u} \tag{5.54}$$

where u is the velocity in the flow direction. The integral between the turbulent eddies velocity u_{eddies} and $\overline{u}$ leads to the expression:

$$\overline{u} = u_{eddies} e^{-t/\tau_p} \quad \text{where} \quad \tau_p = \frac{\rho_p D_p^2}{18\mu_{fluid}} \tag{5.55}$$

τ_p is called "relaxation time". The stopping distance S_p covered by the nanoparticle traveling at $\overline{u}$ is found integrating Equation 5.55 between $\overline{u} = 0$ and u_{eddies}:

$$S_p = \frac{\rho_p D_p^2}{18\mu_{fluid}} u_{eddies} \tag{5.56}$$

and u_{eddies} can be estimated as in Equation 5.57, being in the same order of magnitude as the shear velocity:

$$u_{eddies} \sim \sqrt{\tau_{wall}/\rho_{fluid}} \tag{5.57}$$

Table 5.8 reports the values of liquid film Reynolds number (calculated as $G_{flux}(1 - x)D_h/\mu_b$), the interfacial velocity, u_i (calculated as in Equation 3.32), and the thickness of the liquid film, δ_f (the procedure to estimate the thickness of the liquid film is discussed in Section 5.4.2).

Table 5.8: Liquid film Reynolds number, superficial velocity, and thickness.

OMF	MASS FLUX [$\frac{kg}{m^2 s}$]	x	Re_{liquid}	u_i [$\frac{m}{sec}$]	δ_f [m]
1%	180	0.18	4700.8	1.03	$5.70 \cdot 10^{-4}$
		0.79	928.77	0.69	$2.11 \cdot 10^{-4}$
	425	0.11	12133.8	2.35	$6.51 \cdot 10^{-4}$
		0.79	2068.24	1.78	$1.84 \cdot 10^{-4}$
3%	180	0.14	4103.58	0.93	$6.70 \cdot 10^{-4}$
		0.76	589.28	0.63	$2.69 \cdot 10^{-4}$
	425	0.1	10160.3	2.21	$6.91 \cdot 10^{-4}$
		0.75	1541.33	1.75	$2.39 \cdot 10^{-4}$

It can be observed that in this analysis, the flow regimes are generally laminar ($Re_{liquid} < 2300$) at higher thermodynamic qualities, and turbulent at lower qualities. It is therefore possible to speculate that at low qualities, and closer to the liquid-vapor interface, the liquid could experience turbulence and nanoparticles could be disturbed by the local formation of eddies. For the turbulent cases, the relaxation time, eddies velocity, and stopping distance were calculated according to Equations 5.55, 5.56, 5.57, and the results are reported in Table 5.9.

Table 5.9: Estimation of relaxation time, flow velocity, and stopping distance.

OMF	MASS FLUX [$\frac{kg}{m^2 s}$]	x	τ_p [sec]	u_{eddies} [$\frac{m}{sec}$]	S_p [m]
1%	180	0.18	$4.21 \cdot 10^{-9}$	0.0173	$7.29 \cdot 10^{-11}$
	425	0.11	$4.25 \cdot 10^{-9}$	0.0243	$1.03 \cdot 10^{-10}$
3%	180	0.14	$3.54 \cdot 10^{-9}$	0.0166	$5.87 \cdot 10^{-11}$
	425	0.10	$3.59 \cdot 10^{-9}$	0.0248	$8.94 \cdot 10^{-11}$

In order to understand the magnitude of relative velocity that nanoparticles can acquire in presence of turbulent eddies, it is relevant to remember that, according to the theory of energy cascade presented by Kolmogorov (Pope, 2000), the turbulent kinetic energy is transferred from larger eddies to smaller eddies until it is dissipated by viscous forces into heat. According to

Equations 5.58, the ratio between small and large eddies is used to scale the rate of magnitude change over length, l, and time, t:

$$\frac{l_{large}}{l_{small}} = Re^{3/4} \quad \text{and} \quad \frac{t_{large}}{t_{small}} = Re^{1/2} \tag{5.58}$$

where l_{large} is equivalent to the characteristic length (i.e. the tube diameter, D), and t_{large} is the flow time scale (i.e. $D/\overline{U}$). In this work, the characteristic length was assumed to be equivalent to the thickness of the liquid film and turbulent average velocity $\overline{U}$ was assumed to be equivalent to the liquid-vapor interfacial velocity.

Equations 5.58 were applied to estimate the length and time scales of small eddies for the turbulent cases of Table 5.9. The results are summarized in Table 5.10.

Table 5.10: Length and time scales of small turbulent eddies.

OMF	MASS FLUX [$\frac{kg}{m^2 s}$]	x	$l_{small}\ [m]$	$t_{small}\ [s]$
1%	180	0.18	$1.00{\cdot}10^{-6}$	$8.05{\cdot}10^{-6}$
	425	0.11	$5.63{\cdot}10^{-7}$	$2.51{\cdot}10^{-6}$
3%	180	0.14	$1.31{\cdot}10^{-6}$	$1.13{\cdot}10^{-5}$
	425	0.10	$6.83{\cdot}10^{-7}$	$3.11{\cdot}10^{-6}$

The comparison of these values with those of the stopping distance and relaxation time of Table 5.9, shows that even the smaller eddies have larger length and time scales than those estimated for nanoparticles. Therefore it is possible to conclude that nanoparticles are dominated by the turbulence and transported homogeneously with the fluid.

Another parameter that was calculated to describe the nanoparticles inertial behavior is the Stokes number, Stk. The Stokes number is used in Stokes flows (when the particle Reynolds number is less than 1) to estimate the particles tendency to follow the surrounding fluid streamlines and it is defined as the ratio of a particle characteristic time to the flow characteristic time.

$$Stk = \frac{\tau_p u_f}{D_{tube}} \tag{5.59}$$

where τ_p is the particle relaxation time (Equation 5.55), u_f is the fluid velocity, and D_{tube} is the tube diameter (or the diameter of an obstacle in the fluid stream). If the Stokes number is less than one, then it can be concluded that the particle is dominated by the fluid viscosity and its trajectory follows the fluid streamlines. For the case of a straight pipe, near-wall turbulence can induce non-uniform inertial particle dispersion (Noorani et al., 2016). It is therefore more meaningful to calculate the dimensionless Stokes number, Stk^+, as a function of the friction velocity, $u_f^+ = \sqrt{\tau_{wall}/\rho_f}$.

$$Stk^+ = \frac{\tau_p (u_f^+)^2}{\nu_f} \tag{5.60}$$

For Al_2O_3 nanoparticles dispersed in oil-refrigerant mixture, τ_p is of the order of 10^{-9} and the Stokes number is found to be always smaller than one. The Stokes number could start increasing with increasing friction velocity (i.e. higher wall shear stress) or in the case of the presence of smaller obstacles in the fluid stream. In this particular study, the presence of microfins on the internal wall of a tube could represent obstacles able to increase the Stokes number. The analysis of the impact of microfins was not performed in this work, and it is left as a suggestion for possible future investigations.

5.3.5 Brownian Diffusion

Brownian motion is generally defined as the random diffusion of microscopic particles dispersed in a fluid, as the result of continuous collisions with molecules of the surrounding fluid. Diffusion of microscopic particles is a thermal transport mechanism that eventually results in a homogeneous particles' distribution. The Brownian diffusion coefficient, D_B, describing one particle diffusing in a liquid medium is defined by Einstein's equation:

$$D_B = \frac{k_B T}{f} \tag{5.61}$$

where k_B is the Boltzmann's constant, T is the medium temperature (the product $k_B T$ representing thermal energy), and f is the particle friction coefficient. For the case of a spherical

particle of diameter D_p diffusing in a Newtonian medium of viscosity μ_{fluid}, the friction coefficient for translational motion is defined by the Stokes friction factor:

$$f = 3\pi\mu_{fluid}D_p \tag{5.62}$$

resulting in the Einstein-Stokes' equation:

$$D_B = \frac{k_B T}{3\pi\mu_{fluid}D_p} \tag{5.63}$$

Table 5.11 reports the brownian diffusion coefficient and corresponding diffusion time.

Table 5.11: Brownian diffusion coefficient and corresponding diffusion time.

OMF	MASS FLUX [$\frac{kg}{m^2 s}$]	x	$D_B[\frac{m^2}{s}]$	$t_D[s]$
1%	180	0.18	$3.96{\cdot}10^{-11}$	$9.10{\cdot}10^{-5}$
		0.79	$3.06{\cdot}10^{-11}$	$1.18{\cdot}10^{-4}$
	425	0.11	$3.99{\cdot}10^{-11}$	$9.01{\cdot}10^{-5}$
		0.79	$3.05{\cdot}10^{-11}$	$1.18{\cdot}10^{-4}$
3%	180	0.14	$3.33{\cdot}10^{-11}$	$1.08{\cdot}10^{-4}$
		0.76	$1.72{\cdot}10^{-11}$	$2.10{\cdot}10^{-4}$
	425	0.10	$3.39{\cdot}10^{-11}$	$1.06{\cdot}10^{-4}$
		0.75	$1.81{\cdot}10^{-11}$	$1.98{\cdot}10^{-4}$

Given the small diffusion time, the Brownian effect can have an impact on the distribution of nanoparticles within the laminar sublayer.

5.3.6 Thermophoresis

Thermophoresis is a phenomenon by which a particle can diffuse because of the temperature gradient applied to the surrounding medium (Brenner and Bielenberg, 2005). Under the effect of thermophoresis, a particle moves opposite to the temperature gradient (away from the hot source) at a thermophoretic velocity, u_T, defined as:

$$u_T = -\beta \frac{\mu_{fluid}}{\rho_{fluid}} \frac{\nabla T}{T} \tag{5.64}$$

where ∇T is the temperature gradient ($\nabla T = -q_{flux}/k_{fluid}$), and β is a proportionality factor whose expression was taken from the work of McNab and Meisen (1973) to be equal to: $0.26\frac{k_{fluid}}{2k_{fluid}+k_p}$. This expression was developed for particles of about $1\mu m$ of diameter dispersed in water. A more recent work by Michaelides (2015) suggested different values of proportionality factor for different kinds of nanoparticles dispersed in liquids. Among others, a coefficient was proposed for Al_2O_3 nanoparticles dispersed in refrigerant R134a, β_{ref}, and in engine oil, β_{oil}. These coefficient are respectively reported in Equations 5.65 and 5.66 and they were considered in the model developed for this dissertation to describe the thermophoretic behavior of nanoparticles dispersed in a mixture of refrigerant and lubricant.

$$\beta_{ref} = 6270 R_{nano}^{-1.819} \tag{5.65}$$

$$\beta_{oil} = 7.1026 R_{nano}^{-1.579} \tag{5.66}$$

where R_{nano} is the nanoparticle radius in nm. Another recent research was presented by Corcione et al. (2015) and Quintino et al. (2017) for the study of natural convection of nanofluids. In this work a different correlation for β_{ref} was proposed and, although it was not used in this work, its use should be investigated in the future developments of this dissertation. Table 5.12 reports the thermophoretic velocity and corresponding diffusion time.

Table 5.12: Thermophoretic velocity and corresponding diffusion time.

OMF	MASS FLUX [$\frac{kg}{m^2 s}$]	x	$u_T[\frac{m}{s}]$	$t_D[s]$
1%	180	0.18	$4.21 \cdot 10^{-8}$	1.43
		0.79	$5.42 \cdot 10^{-8}$	1.11
	425	0.11	$4.18 \cdot 10^{-8}$	1.44
		0.79	$5.46 \cdot 10^{-8}$	1.10
3%	180	0.14	$4.97 \cdot 10^{-8}$	1.21
		0.76	$9.47 \cdot 10^{-8}$	0.63
	425	0.10	$4.91 \cdot 10^{-8}$	1.22
		0.75	$9.00 \cdot 10^{-8}$	0.67

It should be noted that the results of Table 5.12 are obtained using the McNab and Meisen (1973) correlation for beta. Equations 5.65 and 5.66 are not function of thermodynamic properties and the estimated values of beta are: $\beta_{ref} = 45.5$ and $\beta_{oil} = 0.0987$. Both values are larger than the one calculated with McNab and Meisen (1973) and therefore they estimate a higher thermophoretic velocity and a smaller diffusion time. Therefore, thermophoresis can have an impact on the distribution of nanoparticles within the laminar sublayer.

5.3.7 Diffusiophoresis

Diffusiophoresis is a phenomenon by which a particle can diffuse because of a macroscopic concentration gradient of a molecular solute interacting with the surface of the particle and pushing it in the direction of the solute lower concentration (Anderson and Prieve, 1984). Because the refrigerant-oil mixture is assumed to be homogeneous and have no concentration gradient, it was concluded that the nanoparticles will not be affected by this phenomenon.

5.3.8 Magnus Effect

The Magnus effect is the lateral lift force experienced by a spinning sphere or particle in a shear flow. When moving from a turbulent layer to a laminar viscous sublayer close to the wall, depending on the relative velocity between the particle (u_p) and the fluid (u_{fluid}), the lift force

will push the particle either towards the wall ($u_p > u_{fluid}$) or away from it ($u_p < u_{fluid}$). This behavior was very well described in the work by Rouhiainen and Stachiewicz (1970) on small particles' deposition from turbulent streams. The authors questioned the particles' deposition models based solely on the concept of Stokes stopping distance, generally valid only in the case of particles moving through a stagnant viscous fluid and in the absence of external forces. The Stokes stopping distance is derived similarly to what done in Subsection 5.3.4 and is formalized as in Equation 5.56. However, in the case of particles moving through a laminar sublayer, the Stokes drag force is not sufficient to describe the effects of a velocity gradient on particles trajectory. Rouhiainen and Stachiewicz (1970) proposed the use of the lift force, F_L, derived by Saffman (1965) to model small spheres moving in an unbounded viscous shear flow:

$$F_{LIFT} = 81.2 \frac{\overline{u}_{LIFT} R_p^2 \mu_{fluid}}{\nu_{fluid}^{1/2}} \left(\frac{du}{dy} \right)^{1/2} \tag{5.67}$$

where $\overline{u}_{LIFT}$ is the relative, or slip, velocity between the particle and the fluid, R_p is the particle radius, and $\dfrac{du}{dy}$ (equal to $\dfrac{u_{average}^2}{\nu_{fluid}} \dfrac{f}{2}$, where f is the friction factor) is the fluid velocity gradient in the shear flow. The equation is applicable when

$$Re_{\overline{u}_{LIFT}} \ll Re_{\frac{du}{dy}}^{1/2}; \quad Re_{\frac{du}{dy}} \ll 1; \quad Re_{\Omega} \ll 1 \tag{5.68}$$

where $Re_{\overline{u}_{LIFT}}$, $Re_{\frac{du}{dy}}$ and Re_{Ω} are defined as:

$$Re_{\overline{u}_{LIFT}} = \frac{\overline{u}_{LIFT} R_p}{\nu_{fluid}}; \quad Re_{\frac{du}{dy}} = \frac{R_p^2}{\nu_{fluid}} \frac{du}{dy}; \quad Re_{\Omega} = \frac{\Omega R_p^2}{\nu_{fluid}} \tag{5.69}$$

(Ω being the particle rotational speed equal to $\frac{l}{2} \frac{du}{dy}$ for free rotation).

In order to compare the magnitude of the lift force with respect to the Stokes drag force in the radial direction ($F_{DRAG,y} = 6\pi R_p \mu_{fluid} u_{p,y}$), the following ratio was calculated:

$$\frac{|F_{LIFT}|}{|F_{DRAG,y}|} = \frac{81.2}{6\pi} \left(\frac{R_p^2}{\nu_{fluid}} \frac{du}{dy} \right)^{1/2} \frac{\overline{u}_{LIFT}}{u_{p,y}} \simeq 4.3 Re_{\frac{du}{dy}}^{1/2} \frac{\overline{u}_{LIFT}}{u_{p,y}} \tag{5.70}$$

$\overline{u}_{LIFT}$ represents the axial relative velocity, while $u_{p,y}$ is the particle radial velocity. It was assumed that when a particle travels through a laminar sublayer, in the proximity of the wall $u_{p,y}$ becomes very small compared to $\overline{u}_{LIFT}$ and the lift force could become relevant. A system of two equations was proposed to describe the particle motion through the laminar sublayer in a horizontal pipe:

$$\begin{cases} \frac{\pi}{6}D_p^3\rho_p\frac{du_x}{dt} = -3\pi D_p\mu_{fluid}\overline{u} \qquad \text{(similar to Eq. 5.54)} & \text{x-direction} \\ \frac{\pi}{6}D_p^3\rho_p\frac{du_y}{dt} = -3\pi D_p\mu_{fluid}u_{p,y} - 81.2\frac{\overline{u}R_p^2\mu_{fluid}}{\nu_{fluid}^{1/2}}\left(\frac{du}{dy}\right)^{1/2} & \text{y-direction} \end{cases}$$

where $\overline{u}$ is equal to $(u_{p,x} - u_{fluid}) = \overline{u}_{LIFT}$. The first equation was integrated between the turbulent velocity $u_{eddies,x}$ and $u_{p,x}$; the second equation was integrated between an initial radial velocity $u_{eddies,y}$ and $u_{p,y}$.

$$\begin{cases} u_{p,x} = u_{fluid}\left(1 - e^{-t/\tau_p}\right) + u_{eddies,x}e^{-t/\tau_p} & \text{x-direction} \\ u_{p,y} = -4.3\frac{\overline{u}R}{\nu_{fluid}^{1/2}}\left(\frac{du}{dy}\right)^{1/2}\left(1 - e^{-t/\tau_p}\right) + u_{eddies,y}e^{-t/\tau_p} & \text{y-direction} \end{cases}$$

where τ_p is the "relaxation time" $\left(\frac{\rho_p D_p^2}{18\mu_{fluid}}\right)$.

The discussion by Rouhiainen and Stachiewicz (1970) concluded that the particle initial radial velocity $u_{p,y}$ causes the particle to move toward the wall. However, only if $u_{p,y}$ is high enough, the particle will enter a region near the wall where $u_{p,x} > u_{fluid}$, causing the lift force to change direction and accelerate the particle toward the wall; otherwise, the particle will be "bounced" away from the wall. The effect of the lift force becomes generally more measurable at higher Reynolds number (bigger velocity gradients du/dx), for larger particles (bigger radii, R_p), and for lighter particles (smaller densities, ρ_p). However, because the particle inertial relaxation time is of the order of 10^{-9}, similarly to what observed for inertial forces, small particles dissipate their inertia very quickly and have no time to penetrate the laminar sublayer towards the wall. Therefore, the Magnus effect was found irrelevant.

5.3.9 Gravity

Nanoparticles's change of velocity due to gravity, u_g, was calculated from a balance between buoyancy and viscous forces (represented by the Stokes's law):

$$\frac{\pi}{6} D_{nano}^3 \left(\rho_{nano} - \rho_{fluid}\right) g = -3\pi D_{nano} \mu_{fluid} u_g \tag{5.71}$$

where the value of u_g can be obtained as:

$$u_g = \frac{D_{nano}^2 \left(\rho_{nano} - \rho_{fluid}\right) g}{18 \mu_{fluid}} \tag{5.72}$$

Table 5.13 reports the gravity velocity and the corresponding diffusion time.

Table 5.13: Gravity velocity and corresponding diffusion time.

OMF	MASS FLUX [$\frac{kg}{m^2 s}$]	x	$u_g[\frac{m}{s}]$	$t_D[s]$
1%	180	0.18	$2.81 \cdot 10^{-8}$	2.14
		0.79	$2.17 \cdot 10^{-8}$	2.76
	425	0.11	$2.84 \cdot 10^{-8}$	2.12
		0.79	$2.17 \cdot 10^{-8}$	2.77
3%	180	0.14	$2.37 \cdot 10^{-8}$	2.53
		0.76	$1.22 \cdot 10^{-8}$	4.91
	425	0.10	$2.41 \cdot 10^{-8}$	2.49
		0.75	$1.29 \cdot 10^{-8}$	4.64

Given the high diffusion time, the gravity effect is considered to have a very small impact on the distribution of nanoparticles within the laminar sublayer.

5.3.10 Drainage

When a bubble or a particle approaches a fluid plane, a resistance develops in the fluid between the particle's surface and the fluid's surface (Charles and Mason, 1960), called "draining fluid". Such resistance is represented by a force proportional to the increasing pressure between the

surfaces. For the case of a sphere of radius R_p approaching a plane, the drainage force, F_D, is calculated as:

$$F_D = 6\pi R_p^2 \mu_{fluid} \frac{u_p}{h} \tag{5.73}$$

where h is the thickness of the draining fluid film, and $u_p = dh/dt$ is the sphere's approach velocity normal to the fluid's plane. However, since this force becomes relevant only when $R_p \gg h$, the contribution of this force was disregarded in the study of spherical nanoparticles approaching the liquid laminar sublayer.

5.3.11 Wall Lubrication

When a spherical particle of radius R_p traveling in a fluid flow approaches a wall perpendicular to the flow direction, the particle is affected by a lubrication force, F_L, defined as:

$$F_L = 6\pi R_p^2 \mu_{fluid} \frac{u_p}{h} \tag{5.74}$$

where h is the thickness of the draining fluid film, and $u_p = dh/dt$ is the sphere's approach velocity normal to the fluid's plane (Marston et al., 2010). The lubrication force is defined similarly to the drainage force. However, this force becomes relevant only when h, is of the same order of magnitude of the particle diameter $2R_p$, meaning that the contribution of this force is measurable only within a very small fluid thickness, near the wall. For this reason, the contribution of the lubrication force was disregarded in the study of spherical nanoparticles approaching the liquid laminar sublayer.

5.3.12 Conclusions

The study of slip mechanisms helps to determine which mechanism will affect nanoparticles distribution. Considering Buongiorno's model, any mechanism that is found able to affect the nanoparticles behavior, should be added to the nanoparticle diffusion mass flux (Equation 3.10). One important finding of the analysis conducted here was the observation that nanoparticles are dominated by eddies in case of turbulent flow, making any of the mechanisms analyzed ineffective. In a laminar flow however, of all the slip mechanisms analyzed, three are considered to

have a potential to affect the distribution of nanoparticles within the laminar sublayer: Brownian motion, thermophoresis, and gravity. However, the diffusion time of the gravity effect is too high for Al_2O_3 nanoparticles in a high viscosity liquid, and for this reason this effect will not be considered in the following analysis. Thermophoresis will be considered in the following analysis, although it should be observed that the impact of this mechanism is very sensitive to the prediction of the diffusion coefficient β, for which very little information can be found in the literature. Other mechanisms could affect the nanoparticle distribution, but they were not considered in this dissertation. Some are already known, such as the effect of electromagnetic forces (Lister, 1980); others could be considered specific to boiling during two-phase flow. In particular, two effects are suggested for future investigation: (i) at different flow regimes, the vapor velocity on the surface of the liquid layer could induce the formation of ripples and waves that could disturb the distribution of nanoparticles at the liquid-vapor interface; (ii) in case of boiling, the bubbles growth and departure could either push the nanoparticles away from the wall surface, or induce "micro-suction" effect from the depression caused by the bubble departure from the nucleation site.

5.4 Correlation Development

The fundamental analysis of nanoparticles slip mechanisms conducted in Section 5.3.1 will be used as a starting point for an investigation on the nanoparticles distribution within the liquid film of a refrigerant-nanolubricant mixture during two-phase flow boiling in a microfin tube. The objective was to provide a physical explanation of the experimental findings presented in Section 4.4 for two-phase flow heat transfer and pressure drop, in light of the data collected on thermophysical properties. An attempt to developing a new correlation was made following the approach presented by Chen (1966), briefly described in Appendix A. For this purpose, a convective flow correlation and a nucleate pool boiling correlations are needed. The convective flow correlation selected was the one developed by Buongiorno (2006) for its more fundamental approach; the nucleate pool boiling correlation selected was the one described by Kedzierski (2012) because it described pool boiling of mixtures of refrigerant and Al_2O_3 laden lubricant on a finned surface (reported in Section 3.4.4). A second approach based on the superposition

model was proposed by Sawant (2012) for convective boiling of mixtures of lubricant and refrigerant R410A in a micro-fin tube. Because of the similarity with the case studied in this work (similar fluid mixture and similar tube geometry), this model approach was also investigated in this dissertation.

5.4.1 Single Phase Radial Analysis and Buongiorno Model

The approach to the study of nanoparticles behavior and their impact on heat transfer performances presented by Buongiorno (2006) was firstly applied in this dissertation to describe the single-phase heat transfer of the refrigerant-nanolubricant mixture. The theoretical approach described by Buongiorno (2006) was extended to study the behavior of nanoparticles in a high viscosity liquid flow. On top of the assumptions already made by Buongiorno and listed in Section 3.5.1, the following assumptions were made:

1. Uniform refrigerant-oil mixture
2. Nanoparticles are always in solution with oil
3. Surfactant has a negligible impact on performances, compared to the impact of nanoparticles motion

Assumption number one finds its justification in the fact that refrigerant and oil are always miscible in the range of temperatures tested in this work. The reason for assumption number two was that, although no information was provided by the manufacturer on the nature of the surfactant, it was speculated that the surfactant chains would bond with the POE oil molecules more than they would with the refrigerant molecules. Finally, because experimental results showed both cases of heat transfer enhancement and degradation, using the same nanoparticles with the same surfactant, it was speculated that the impact of surfactant was less important than the effect of the nanoparticles motion (assumption number three).

Buongiorno's model is a physically based model and the only parameter that needed to be determined empirically is the adimensional thickness of the laminar sublayer, δ_v^+. Because in absence of nanoparticles ($\phi = 0$) this model converges to the Dittus-Boelter model for single

phase flow, it was possible to estimate the value of δ_v^+ for the case of pure refrigerant. This result was obtained by converging the heat transfer coefficient calculated with Buongiorno's model, to the value obtained applying Dittus-Boelter correlation. The value of δ_v^+ estimated for pure refrigerant R410A convective flow was ~ 10 and it was later maintained constant for all tests with oil or nanolubricant.

In addition to the strict implementation of Buongiorno's model, a single phase radial analysis of the laminar sublayer was implemented in the simulation, with the scope to observe the distribution of nanoparticles and the effect on both thermophysical properties, and shear stress and flow velocity, as a function of the parameter N_{BT}. Because at different thermodynamic qualities, the mass ratio of oil and liquid refrigerant changes, the thermophoretic coefficient, β, was calculated as a mass weighted average of the local concentration of oil in liquid refrigerant, according to Equation 5.75:

$$\beta = \beta_{ref} \cdot (1 - \omega_{local}) + \beta_{oil} \cdot \omega_{local} \tag{5.75}$$

where ω_{local} represents the local oil mass fraction, and β_{ref} and β_{oil} are the thermophoretic coefficients for refrigerant and oil proposed by Michaelides (2015) and previously discussed in Section 5.3.6.

Radial Analysis of the Laminar Sublayer

The region of the liquid layer that is near a wall (no-slip boundary condition) is dominated by viscous forces that induce a laminar behavior (Hewitt and Hall-Taylor, 1970), where the profiles of shear stress and velocity vary as a function of the nanoparticle volume concentration. In order to study the different distributions, the laminar sublayer was internally subdivided in a number n of smaller layers of equivalent thickness $\frac{\delta_v}{n}$. The thickness of the laminar (or viscous) sublayer δ_v was calculated from δ_v^+ using Equation 5.76:

$$\delta_v = \delta_v^+ \frac{\mu_v / \rho_v}{\sqrt{\tau_{wall} / \rho_b}} \tag{5.76}$$

where the wall shear stress τ_{wall} was estimated using the thermophysical properties of the bulk fluid, i.e. the thermophysical properties of the liquid mixture of refrigerant, oil and nanoparticles at volume concentration $\phi_{bulk} = \phi_{nominal}$. For this purpose, a traditional friction factor correlation for turbulent flow f_{turb} was used to calculate the wall shear stress:

$$f_{turb} = \frac{C}{Re_{bulk}^{n}} \tag{5.77}$$

where $Re_{bulk} = \frac{\rho_{bulk} u_{bulk} D_h}{\mu_{bulk}}$, $u_{bulk} = \frac{\dot{m}_{TOT}}{\rho_{bulk} A_{sec}}$, $A_{sec} = \frac{\pi D_h^2}{4}$, $C = 0.184$, and $n = 0.2$.

$$\tau_{wall} = \frac{f_{turb}}{8} \rho_{bulk} u_{bulk}^2 \tag{5.78}$$

A schematic representation of the radial segmentation is given in Figure 5.8 where $n = 3$.

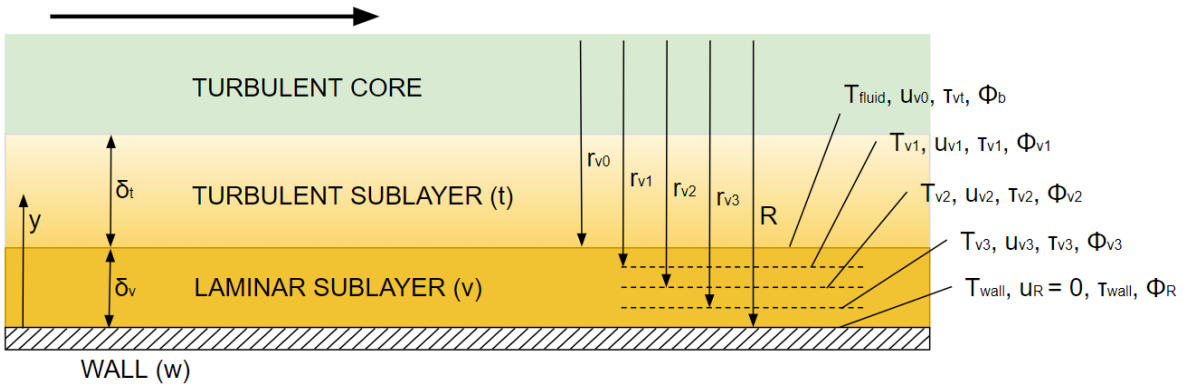


Figure 5.7: Schematic representation of the segmentation of the laminar sublayer in a single phase flow ($n = 3$).

For each layer, the nanoparticle volume concentration was estimated according to Equation 3.21, reported here again:

$$\phi_{v,j} = \phi_{bulk} e^{-(\frac{1}{N_{BT}})(1-\frac{y}{\delta_v})} \tag{5.79}$$

where the value of y was now changed with the radius distance of each layer j, with j that goes from 0 to n ($j = 0$ at the laminar-turbulent interface and $j = n$ at the tube wall where $r = R$). It was assumed that temperature varied linearly between the wall and the laminar-turbulent interface. In Buongiorno's model, the wall temperature, T_{wall}, was found as

a convergence parameter and then it was used in this analysis to determine the temperature of each segment, according to Equation 5.80

$$T_j = T_{fluid} + (T_{wall} - T_{fluid}) \cdot \frac{j}{n} \tag{5.80}$$

Based on the local value of volume concentration $\phi_{v,j}$, and on the local temperature,T_j, the local thermophysical properties were calculated. The local shear stress and velocity were calculated using respectively Equation 3.28 and Equation 3.32 (being $\frac{dp}{dz} = \frac{-2 \cdot L}{D_h \tau_{wall}}$). The boundary conditions are listed here:

1. T_{vt} = T_{fluid}
2. ϕ_{vt} = ϕ_b
3. u_R = 0

5.4.2 Two-Phase Radial Analysis and Radermacher-Cremaschi Model

A subroutine was added to the model in order to evaluate the radial distribution of properties, shear stress and velocity within the liquid film as a function of the local nanoparticle concentration. The approach followed is very similar to the one described in Section 3.5.2, although in this case the unknown was the thickness of the liquid film.

Liquid Film Thickness

The first step of the subroutine was to estimate the thickness of the liquid film. According to the procedure described in Carey (1992), the liquid mass flow rate was first calculated as in Equation 5.81:

$$\dot{m}_L = \frac{\pi D_h^2}{4} G_{flux} \left(1 - x_{in}\right) \tag{5.81}$$

where G_{flux} is the total mass flux and x_{in} is the mixture quality at the segment inlet. Then, based on a first guess of the liquid layer thickness $\delta_{L,guess}$, the void fraction α was calculated as:

$$\alpha = \left(\frac{D_h - 2\delta_{L,guess}}{D_h} \right)^2 \tag{5.82}$$

In sequence, the following quantities were calculated: the radius of the interfacial thickness r_i, the velocity of the vapor phase u_{core}, and the interfacial shear stress τ_i:

$$r_i = \frac{D_h}{2} - \delta_{L,guess} \tag{5.83}$$

$$u_{core} = \frac{G_{flux} x_{in}}{\rho_v \alpha} \tag{5.84}$$

$$\tau_i = \frac{1}{2} f_i \rho_v u_{core}^2 \tag{5.85}$$

The friction factor f_i was estimated as suggested by Hewitt and Hall-Taylor (1970) using the following equation:

$$f_i = f_g \left(1 + 300 \frac{\delta_f}{D_h} \right) \tag{5.86}$$

where $f_g = 0.079(Re_{core})^{-1/2}$ is a friction factor for the gas core. The pressure drop was calculated accounting for both friction and momentum contributions:

$$\frac{dp}{dz} = -\frac{4\tau_i}{D_h} - \frac{2xG_{flux}^2}{\alpha^2 \rho_{core}} \frac{4q"}{G_{flux} D_h h_{LV}} \tag{5.87}$$

The liquid mass flow rate was then calculated according to Equation 3.33 and the result compared to the one obtained from Equation 5.81. By changing the value of $\delta_{L,guess}$, the process was iterated until Equations 5.81 and 3.33 converged, that is until the correct value of film thickness δ_L was found. This value of δ was then used during the two-phase flow radial analysis, in place of δ_v representing the thickness of the laminar sublayer in the case of a single phase convective flow.

Radial Analysis of the Liquid Film

From the analysis conducted in Section 5.3.1, it can be assumed that the liquid film close the wall is generally laminar. Similarly to what observed for single phase flow, within the laminar sublayer, the profiles of shear stress and velocity vary as a function of the nanoparticle volume concentration and temperature. In order to study the different distributions, the laminar sublayer was internally subdivided in a number n of smaller layers of equal thickness $\frac{\delta_v}{n}$. A schematic representation is given in Figure 5.8 where $n = 3$.

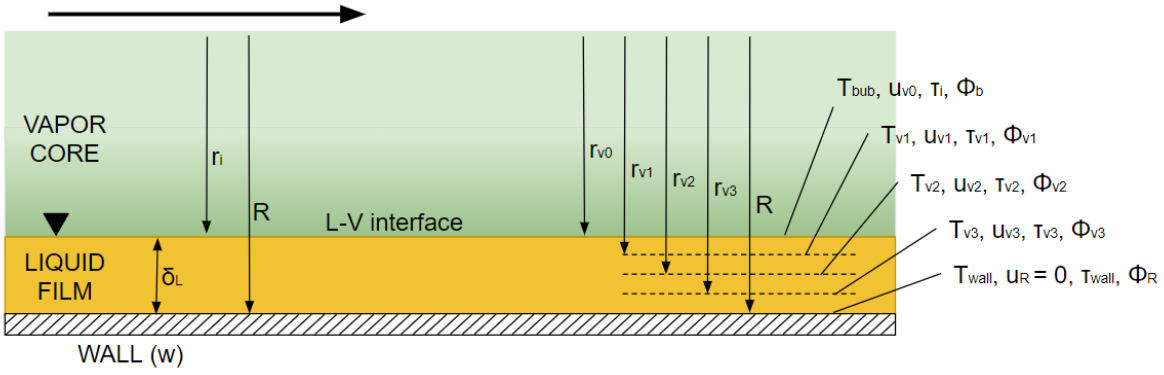


Figure 5.8: Schematic representation of the segmentation of the liquid film in a two-phase flow (in this schematic, the sublayer is assumed to be all laminar and $n = 3$).

For each layer, the nanoparticle volume concentration was estimated according to Equation 3.21, reported here again (where δ_v was replaced by the thickness of the liquid film δ_L:

$$\phi_{v,j} = \phi_b e^{-(\frac{1}{N_{BT}})(1-\frac{y}{\delta_L})} \tag{5.88}$$

where the value of y was now changed with the radius distance of each layer j, with j that goes from 0 to n ($j = 0$ at the laminar-turbulent interface and $j = n$ at the tube wall where $r = R$). Based on the local value of $\phi_{v,j}$ and temperature, the local thermophysical properties were calculated for each segment. In this case the wall temperature was already a simulation input variable, obtained directly from the experimental data set. The temperature of each segment was calculated according to Equation 5.80. The local shear stress and velocity were calculated using respectively Equation 3.28 and Equation 3.32 (being $\frac{dp}{dz}$ estimated as in Equation 5.87). The boundary conditions are listed here:

1. $T_i = T_{bub}$

2. $\phi_i = \phi_b$

3. $u_R = 0$

4. $u_i = u_{core}$

5.4.3 Friction Factor of Microfins

The experimental data presented in Section 4.4.7 was collected for refrigerant-nanolubricant mixture during two-phase flow boiling in a microfin tube. The pool boiling correlation proposed by Kedzierski (2012) was chosen in this work because it estimates the nucleate boiling heat transfer coefficient of nanolubricant-refrigerant mixtures on a finned surface. However, the correlation for nanofluid single phase convective flow presented by Buongiorno (2006) assumed the tube was smooth. Therefore, in order to model the effect of microfins on the internal walls of the tube, an appropriate friction factor correlation was selected from the literature. The works by Wang and Rose (2004) and Meyer and Olivier (2011) suggested the use of the correlation by Jensen and Vlakancic (1999) designed for turbulent heat transfer and fluid flow in internally finned tubes ($f_{turb,finned}$). The correlation is function of geometrical parameters such as the tube envelope diameter (or root diameter, D_{root}), the number of microfins (N_{fins}), the fins' height (h_{fin}), helix angle (γ), and apex angle (α). The correlation is reported here in Equation 5.89:

$$\frac{f_{turb,finned}}{f_{turb,smooth}} = \left(\frac{L_c}{D_{root}}\right)^{-1.25}\left(\frac{A_{root}}{A_c}\right)^{1.75} - \frac{0.0151}{f_{turb,smooth}}\left[\left(\frac{L_c}{D_{root}}\right)^{-1.25}\left(\frac{A_{root}}{A_c}\right)^{1.75} - 1\right] exp\left(\frac{-Re}{6780}\right) \quad (5.89)$$

where A_c is the actual tube cross sectional area, A_{root} is the nominal cross sectional area based on the root diameter, $f_{turb,smooth}$ is a smooth tube friction factor, and the expression for

the characteristic length $\frac{L_c}{D_{root}}$ is the one suggested by Wang and Rose (2004), reported here in Equation 5.90:

$$\frac{L_c}{D_{root}} = 1 - 1.577 \left(\frac{N_{fins} sin\gamma}{\pi} \right)^{0.64} \left(\frac{2 \cdot h_{fin}}{D_{root}} \right)^{0.53} \left[\left(\frac{\pi}{N_{fins}} - \frac{s}{D_{root}} \right) cos\gamma \right]^{0.28} \quad (5.90)$$

The value of the fin average width (s) was estimated according to the formulation used by Meyer and Olivier (2011) ($s = 4/3 \cdot h_{fin} \cdot tan(\alpha/2)$). For the smooth tube friction factor ($f_{turb,smooth}$), Meyer and Olivier (2011) suggested the use of the Darcy-Weisbach expression, while Wang and Rose (2004) suggested the correlation by Filonenko (1954). In order to be consistent with Buongiorno's model, in this work $f_{turb,smooth}$ was estimated using Equation 5.77.

As observed by Meyer and Olivier (2011), compared to a smooth tube, the presence of microfin induces an early transition from laminar to turbulent flow. A "secondary" transition (increase in friction factor) was also caused by the flow rotation favored by the fins. It was concluded that only the height of the fins (or of the roughness) influenced the transition.

5.4.4 Superposition Model

According to the superposition model for the calculation of heat transfer coefficients during flow boiling, the total heat transfer coefficient can be approximated as the summation of a nucleate boiling heat transfer coefficient and a convective heat transfer coefficient. This model was utilized in this work because believed to be a simple and comprehensive approach to first modeling the two-phase flow performances of refrigerant-nanolubricant mixtures. The convective heat transfer coefficient, h_L, was obtained from the application of Buongiorno's model (Buongiorno, 2006) to the case of refrigerant-nanolubricant mixture. The nucleate boiling heat transfer coefficient, h_{nb}, was obtained from the application of Kedzierski's correlation (Kedzierski, 2012). Chen (1966) suggested that flow velocity can suppress nucleate boiling and for this reason he proposed the use of a "suppression factor", S, as a corrective multiplier

for the nucleate boiling ($S \cdot h_{nb}$). The two-phase convective evaporation heat transfer is calculated as a function of the liquid phase convective heat transfer (h_L), multiplied by a "two-phase convection multiplier", F ($h_{cv} = F \cdot h_L$). Therefore:

$$h = S \cdot h_{nb} + F \cdot h_L \tag{5.91}$$

The two-phase multiplier, F, is described by Chen (1966) as the ratio of a two-phase Reynolds number, Re_{tp}, to the liquid Reynolds number, Re_L. Being this ratio a flow parameter, it was assumed that, by analogy to momentum transfer in two-phase flow, F was also a function of the Martinelli parameter, X_{tt}:

$$X_{tt} = \left(\frac{1-x}{x}\right)^{0.9} \left(\frac{\rho_v}{\rho_L}\right)^{0.5} \left(\frac{\mu_L}{\mu_v}\right)^{0.1} \tag{5.92}$$

The two-phase multiplier was represented as in Equation 5.93, so that it would collapse to 1 in case of single phase flow ($X_{tt} = 0$):

$$F = 1 + f(X_{tt}) \tag{5.93}$$

The suppression factor, S, is defined as the ratio of the effective superheat to the total superheat at the wall, and it was represented as a function of the two-phase Reynolds number, Re_{tp}. S collapses to 1 in case of no flow, and it goes to zero in case of infinite flow. For this reason it is represented as in Equation 5.94:

$$S = \frac{1}{1 + f(Re_{tp})} \tag{5.94}$$

A recent work by Chen and Fang (2014) analyzed different expressions of F and S available in the literature and indicated that the correlation that best represents the two-phase multiplier F obtained by Chen (1966) is provided in the work of Bergles et al. (1981) and reported here in Equation 5.95:

$$F = \begin{cases} 2.35(\frac{1}{X_{tt}} + 0.213)^{0.736}, & \text{if } 1/X_{tt} > 0.1 \\ 1, & \text{if } 1/X_{tt} \leq 0.1 \end{cases} \tag{5.95}$$

Among many parametric equations used to describe the suppression factor S, one that is simple is the one reported in the work by Orian et al. (2010) and shown in Equation 5.96:

$$S = \frac{1}{1 + 2.53 \cdot 10^{-6}(Re_L F^{1.25})^{1.17}} \tag{5.96}$$

where the quantity $Re_L F^{1.25}$ represents the two-phase Reynolds number Re_{tp}.

The procedure suggested by Chen is described in detail in Appendix A, while the results of its application to the experimental data collected for this work are reported in Section 6.5.

Recently, a correlation based on the superposition model was proposed by Sawant (2012) for convective boiling of mixtures of lubricant and refrigerant R410A in a micro-fin tube. Sawant (2012) suggested an approach similar to the one used by Gungor and Winterton (1986) for flow boiling in tubes and annuli. Based on observations of two-phase flow velocities, quality, and boundary layer thickness it was argued that the two-phase multiplier, F, and the suppression factor, S, could be functions of the two-phase Reynolds number and of quality. In particular, the functions representing F and S were chosen to have the functional form of Equations 5.97 and 5.98, respectively:

$$F = \frac{A_1}{X_{tt} \cdot Re \cdot x} \tag{5.97}$$

$$S = e^{-A_0 \cdot Re \cdot x} \tag{5.98}$$

It was not specified what formulation of the Reynolds number was used, and here it is assumed that $Re = \frac{G \cdot D_h}{\mu_L}$. The parameters A_0 and A_1 were calculated based on an iterative procedure that was not described in the work by Sawant (2012). However, it was mentioned that for S, the iterative procedure had to convergence to the value obtained from Equation 5.99, assuming F to be equal to unity.

$$S = \frac{h_{exp} - h_L}{h_{nb}} \tag{5.99}$$

Once S was obtained, the value of F in Equation 5.97 was estimated by converging to the value of Equation 5.100:

$$F = \frac{h_{exp} - S \cdot h_{nb}}{h_L} \tag{5.100}$$

The final form of the correlation presented by Sawant (2012) for two-phase flow of oil-refrigerant mixtures in a microfin tube is reported here in Equation 5.101:

$$h_{tp} = 13.7 \cdot e^{-0.00132 \cdot Re \cdot x} \cdot h_{nb} + 1.685 \cdot 10^{13} \left(\frac{1}{X_{tt} \cdot Re \cdot x} \right)^{4.419} \cdot h_L \tag{5.101}$$

Because of the similarity with the geometry and type of fluid utilized to develop this correlation, the same procedure was followed in this work to describe the two-phase heat transfer coefficient of refrigerant-nanolubricant mixtures. The procedure followed in this work is reported in Appendix A, while the results of its application to the experimental data are reported in Section 6.5.

Chapter 6

Results and Discussion

In this section the simulation results will be discussed with particular focus to the radial analysis of the laminar liquid film.

6.1 Buongiorno Validation

The correlation developed by Buongiorno (2006) described single-phase convective flow of Al_2O_3 nanoparticles dispersed in water. This model was implemented in a subroutine that followed step-by-step the algorithm described to calculate the Nusselt number. A bulk Reynolds number (Re_b), bulk Prandtl number (Pr_b), and a laminar sublayer Prandtl number (Pr_v) were calculated as a function of the nanoparticle concentration in the laminar sublayer. The correctness of the implementation of this model in the simulation tool developed for this work was directly verified by comparison of the simulation results with the results presented by Buongiorno in the same paper, for the case of a single phase water flow, at two different nanoparticle concentrations ($\phi = 0.01$ and $\phi = 0.03$). Figure 6.1 shows the trends of Nusselt number at different Reynolds number for different correlations. The plotting matches successfully the results reported by Buongiorno for water and Al_2O_3 nanoparticles of 13 nm in diameter.

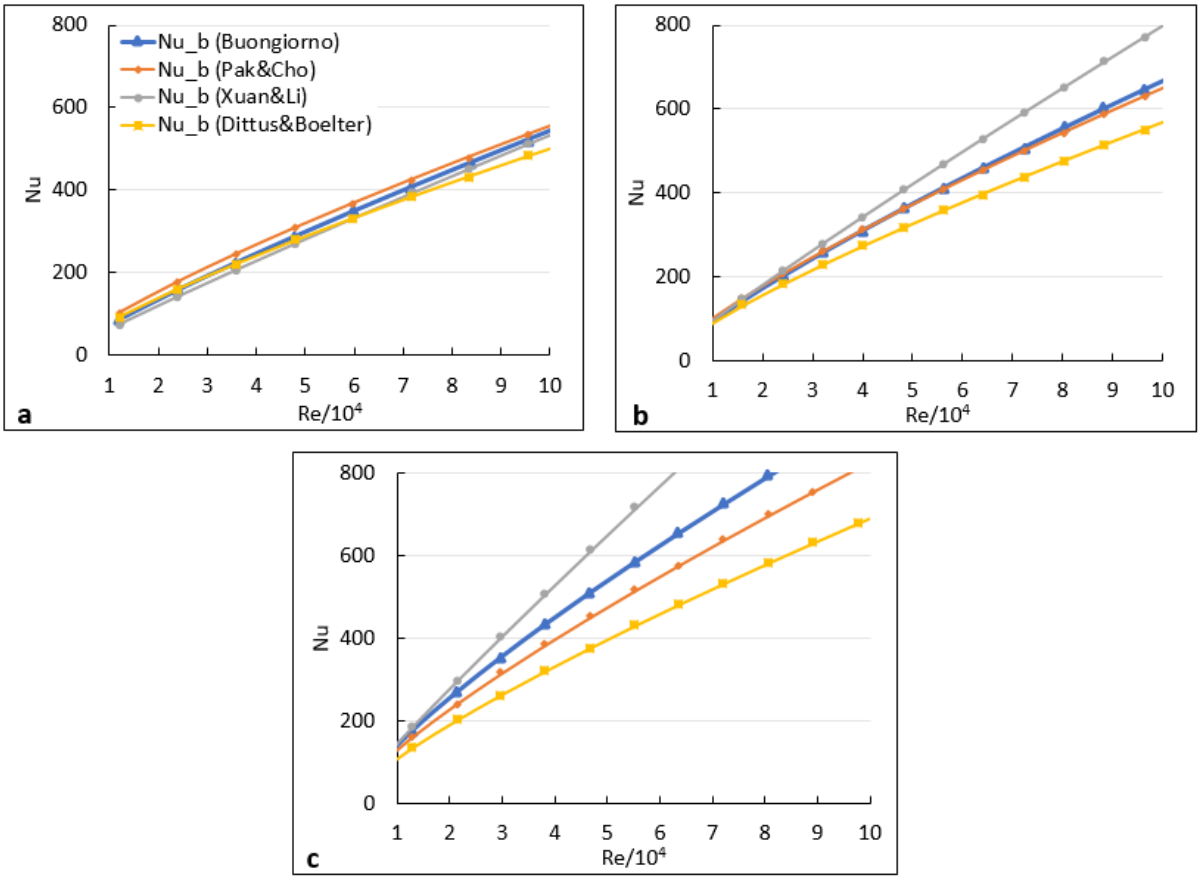


Figure 6.1: Validation of the correct implementation of Buongiorno's model at (a) $\phi = 0$, (b) $\phi = 0.01$, and (c) $\phi = 0.03$.

The same validation also proved that the simulation tool was working correctly. After changing the routines for the calculation of the thermophysical properties, the same model was then applied to the cases of (i) single phase pure refrigerant R410A, (ii) refrigerant and oil mixture, and (iii) refrigerant and nanolubricant mixture. This analysis was conducted to test the sensitivity of the simulation tool to different mixtures, although no actual validation of these results was possible because no single phase tests were previously collected, and no literature database was found for single phase flow of refrigerant and oil mixtures. Nonetheless, in this work an attempt was made to extrapolate single phase data of pure refrigerant and mixtures of refrigerant and oil (or nanolubricant), from the data recorded at the preheater of the experimental facility previously described. Because the preheater is the section of the experimental setup used to prepare the conditions of the fluid at the inlet of the test section, it is reasonable to assume that, for those experimental series collected at low thermodynamic quality, a large

part of the preheater was occupied by single phase flow. The pressure and temperature of the all-liquid mixture measured at the inlet of the preheater were used to calculate the preheater inlet enthalpy. According to the thermodynamic quality desired at the inlet of the test section, a different heat flux was provided by water counter-flowing in the preheater jacket. From an analysis of the preheater energy balance and knowing the geometry details of the preheater section, the single phase heat transfer coefficient of the water flowing in the jacket annulus, and of the refrigerant-nanolubricant mixture were estimated using the Dittus-Boelter correlation. These results were then compared to the Nusselt numbers obtained from the application of Buongiorno's model to the cases of mixtures of refrigerant and oil (or nanolubricant). The result of this comparison are represented in Figure 6.2.

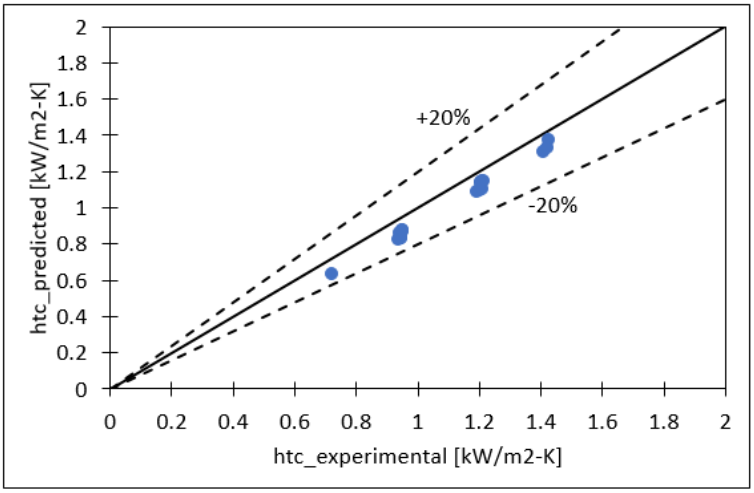


Figure 6.2: Verification of Buongiorno's model against extrapolated single phase data.

Although this analysis cannot be considered a validation of the applicability of Buongiorno's model to single phase mixtures of refrigerant and oil, it was however interpreted as a second confirmation of the correctness of Buongiorno's model implementation, in that it converged to the Dittus-Boelter correlation as suggested by Buongiorno.

6.2 Comparison with Literature Correlations

This section presents a comparison of the experimental data with literature two-phase flow heat transfer correlations. The correlations chosen are those presented previously in Section 5.4.4 by Sawant (2012) and by Chen (1966). The correlation by Gungor and Winterton (1986)

is another form of superposition model (where $E = 1 + 24000Bo^{1.16} + 1.37(\frac{1}{X_{tt}})^{0.86}$, $S = 1/(1 + 1.15 \cdot 10^{-6} \cdot E^2 \cdot Re_L^{1.17})$, and $Re_L = \frac{G \cdot (1-x) \cdot D_h}{\mu_L}$) and it was also chosen for comparison with the experimental data of this work.

The values of h_{nb} and h_L were estimated applying the models by Kedzierski (2012) and Buongiorno (2006). The friction factor for finned surface ($f_{turb,finned}$) was estimated according to Jensen and Vlakancic (1999), as described in Section 5.4.3. Figures 6.3, 6.4, and 6.5 show the comparison with the three aforementioned correlations.

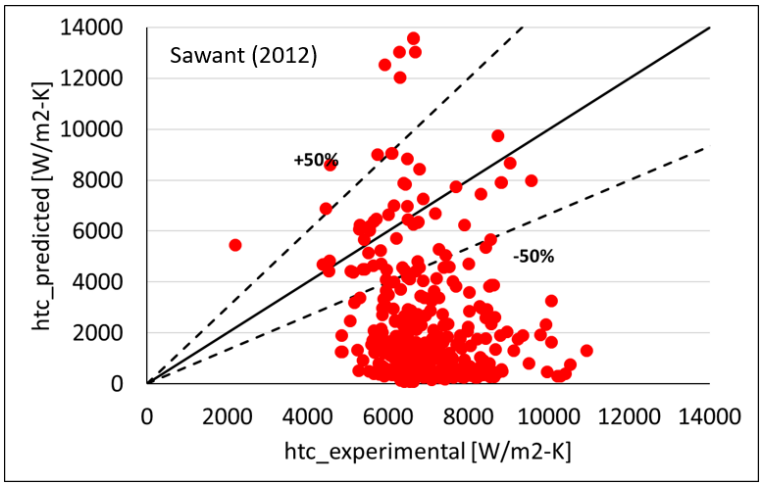


Figure 6.3: Comparison of the experimental data with the correlation by Sawant (2012).

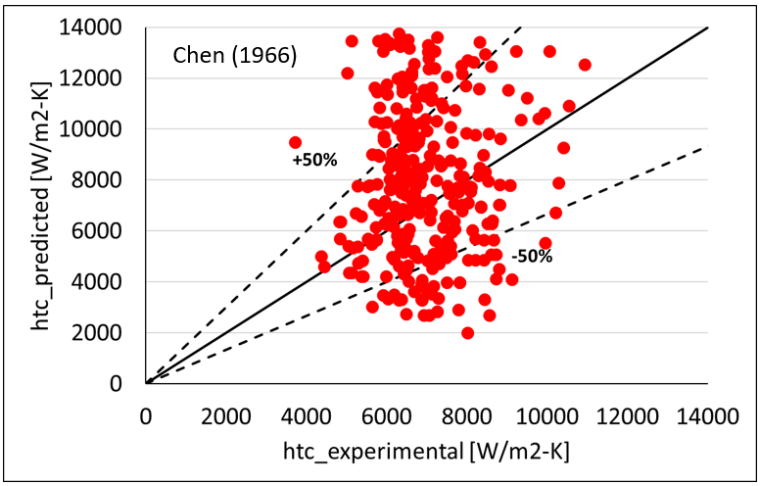


Figure 6.4: Comparison of the experimental data with the correlation by Chen (1966).

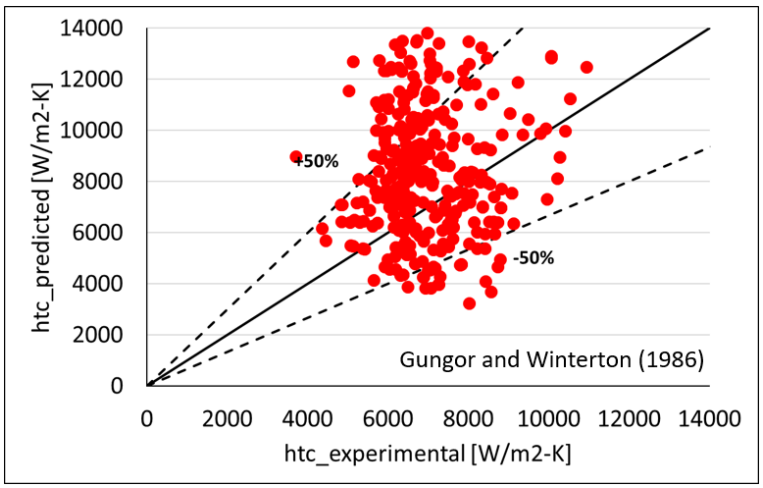


Figure 6.5: Comparison of the experimental data with the correlation by Gungor and Winterton (1986).

The comparison of Figures 6.3, 6.4, and 6.5 shows that the models proposed by Chen (1966) and Gungor and Winterton (1986) were able to predict the majority of the experimental data of this work with an uncertainty of about $\pm 50\%$. However, the model by Sawant (2012) generally underpredicted the data.

In this work the model was originally chosen because of the similarity with the geometry and type of fluid investigated, and according to Sawant (2012), the model was developed following an approach similar to the one used by Gungor and Winterton (1986) for flow boiling in tubes and annuli. However, since the correlation by Gungor and Winterton (1986) proved to provide better predictions, it is unclear why the application of the model by Sawant (2012) resulted in such a large deviation.

It was concluded that the correlation by Sawant (2012) is not suitable to describe the experimental data of this work.

6.3 N_{BT} Sensitivity Analysis

A sensitivity analysis was conducted to observe the distribution of nanoparticles in the liquid laminar layer. In particular, this analysis focused on $N_{BT} = (D_B T_b \rho)/(\beta \mu \frac{q_w \delta_v}{k})$ (previously introduced in Equation 3.22) representing the ratio between the Brownian and the thermophoretic diffusion coefficients.

N_{BT} is relevant because, according to Equation 3.21 (reported here: $\phi = \phi_b e^{-(\frac{1}{N_{BT}})(1-\frac{y}{\delta_v})}$), it describes the distribution of nanoparticles in the laminar sublayer, as represented in Figure 6.6. The larger N_{BT} is, the more uniformly the nanoparticles will distribute because of the stronger effect of Brownian motion; the smaller N_{BT} is, the more the nanoparticles will move away from the wall because of the stronger effect of thermophoresis.

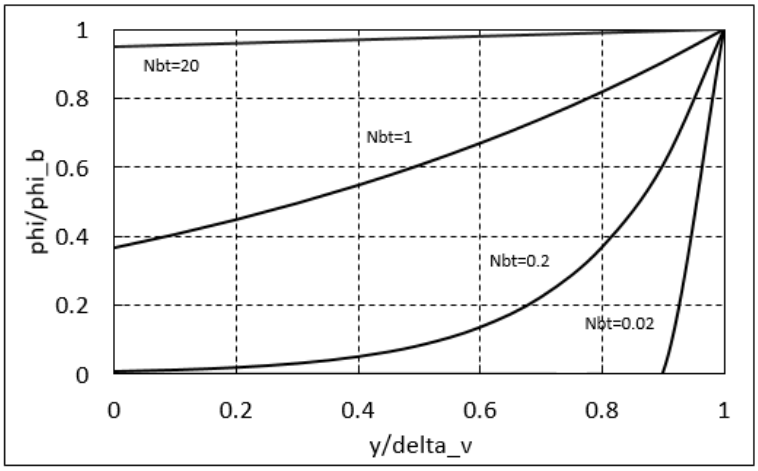


Figure 6.6: Distribution of nanoparticle concentration within the laminar sublayer as a function of N_{BT} (adapted from Buongiorno (2006)).

It should be observed that many variables converge into the parameter N_{BT} and for simplicity they will be listed here:

- wall heat flux, q_{wall}"
- thickness of the laminar liquid layer, δ_v (also representative of mass flux)
- thermophoretic coefficient, β
- saturation (or bubble) temperature, T_b
- thermophysical properties, ρ, μ, k
- nanoparticle diameter, D_{nano} (present in the Brownian diffusion coefficient)

It can be expected that at higher heat flux, thermophoretic coefficient, viscosity, and nanoparticle diameter, N_{BT} will get smaller. Similarly, at higher mass flux, saturation temperature, density, and thermal conductivity, N_{BT} will increase.

6.3.1 Single Phase Radial Analysis

In this section the results of the radial analysis for a single phase convective flow are presented. In particular, observations are made with respect to the change of thermophysical properties, and for the impact on the velocity profile within the laminar sublayer. In this analysis, the diameter was chosen to be equal to the equivalent length ($D_e = 8.8\ mm$, as reported in Table 4.1).

Single Phase Thermophysical Properties Gradient

Figure 6.7 shows trends of thermophysical properties as a function of N_{BT}, calculated with the set of correlations presented by Buongiorno (2006). The thermophysical properties are (a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat, and they are calculated for a refrigerant R410A based nanofluid with Al_2O_3 nanoparticles in bulk volume concentration of about 0.01 (concentration similar to the one used in Buongiorno's paper for water). The y-axes represents the height of the liquid film thickness from the wall. It can be observed that when N_{BT} is high (marked by grey triangles), the ratio ϕ_{local}/ϕ_{bulk} is very close to unity and therefore all thermophysical properties show a linear trend that is representative of the temperature change across the layer (assumed to be linear in the simulation); this condition describes the case of uniform nanoparticles distribution induced by Brownian motion. When N_{BT} is low (marked by blue squares), the ratio ϕ_{local}/ϕ_{bulk} is closer to zero for a large portion of the laminar sublayer; this condition signifies that nanoparticles migrate away from the wall, towards the center of the tube by effect of thermophoresis. Therefore, in this case all thermophysical properties closer to the wall will be closer to the properties of the nanoparticle-free fluid.

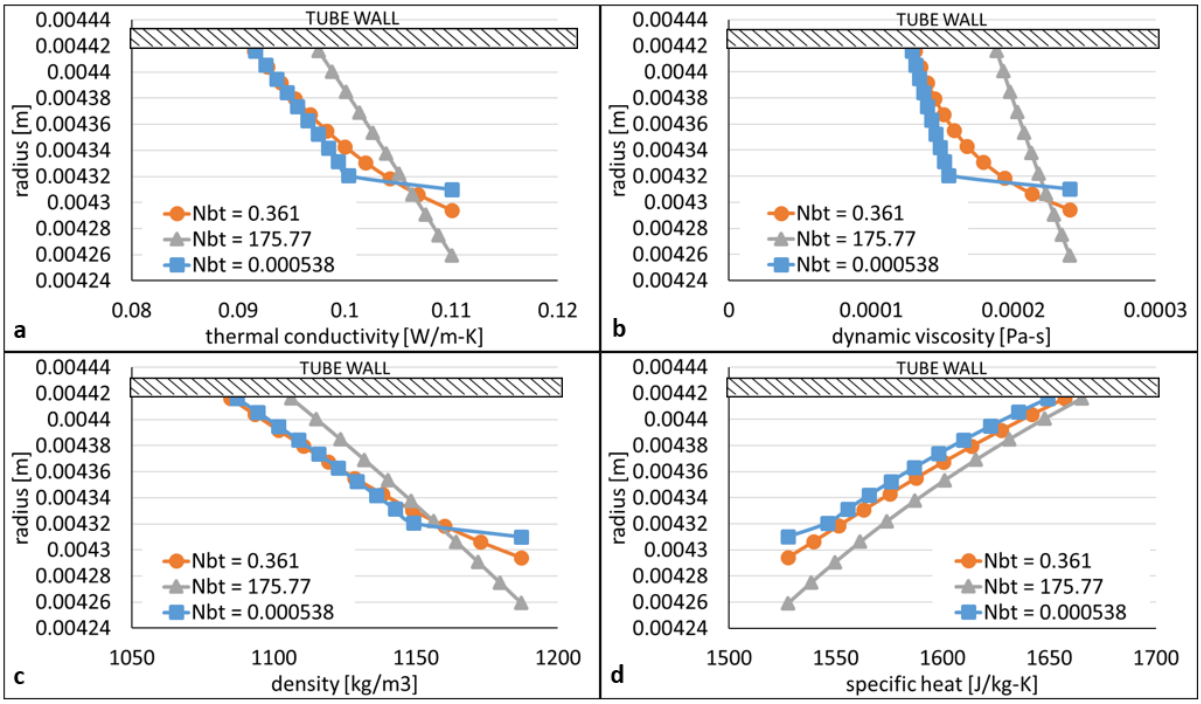


Figure 6.7: Thermophysical properties ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) within the laminar sublayer as a function of N_{BT}, for the case of a R410A based nanofluid (ϕ_b = 0.01).

Figures 6.8 and 6.9 report trends of thermophysical properties change in the laminar sublayer as a function of N_{BT}, for the case of a single phase mixture of refrigerant R410A and nanolubricant at 1 wt.% and 3 wt.% mass fraction, and 20% Al_2O_3 nanoparticle mass fraction in oil. In this case, the thermophysical properties were calculated according to the set of correlations presented in Section 5.1.

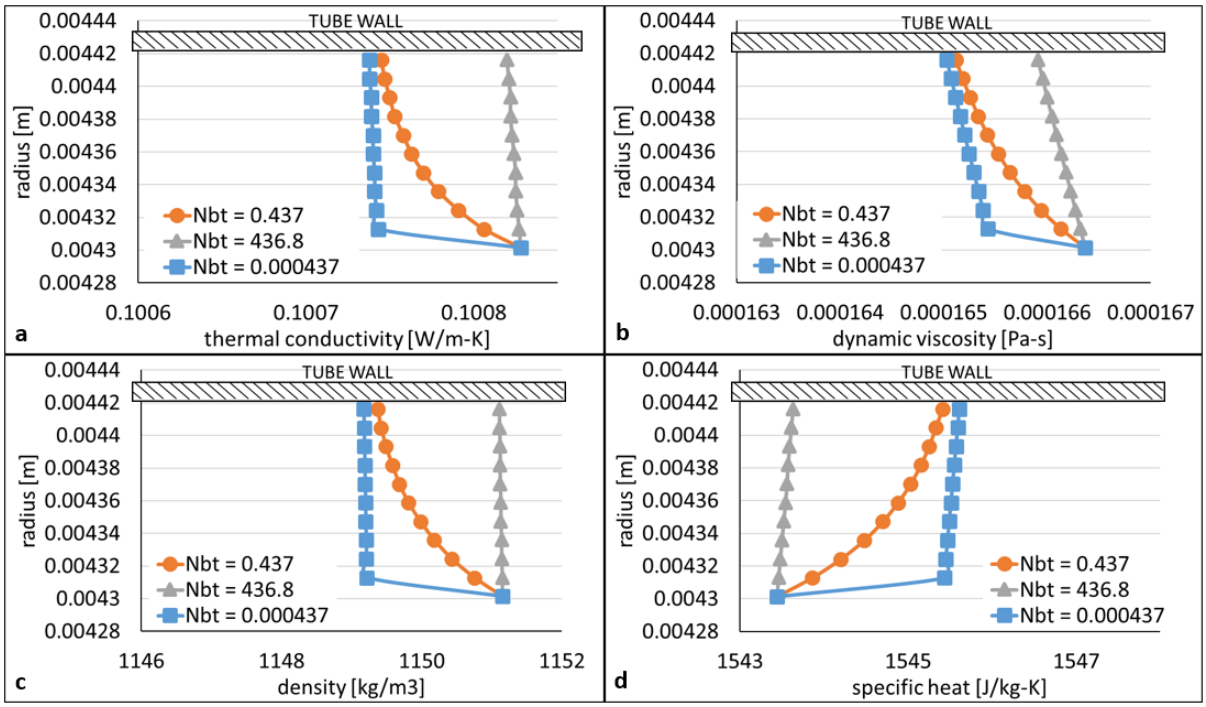


Figure 6.8: Thermophysical properties ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) within the laminar sublayer as a function of N_{BT}, for the case of a R410A-lubricant based nanofluid (OMF = 1%, NMF = 20%)

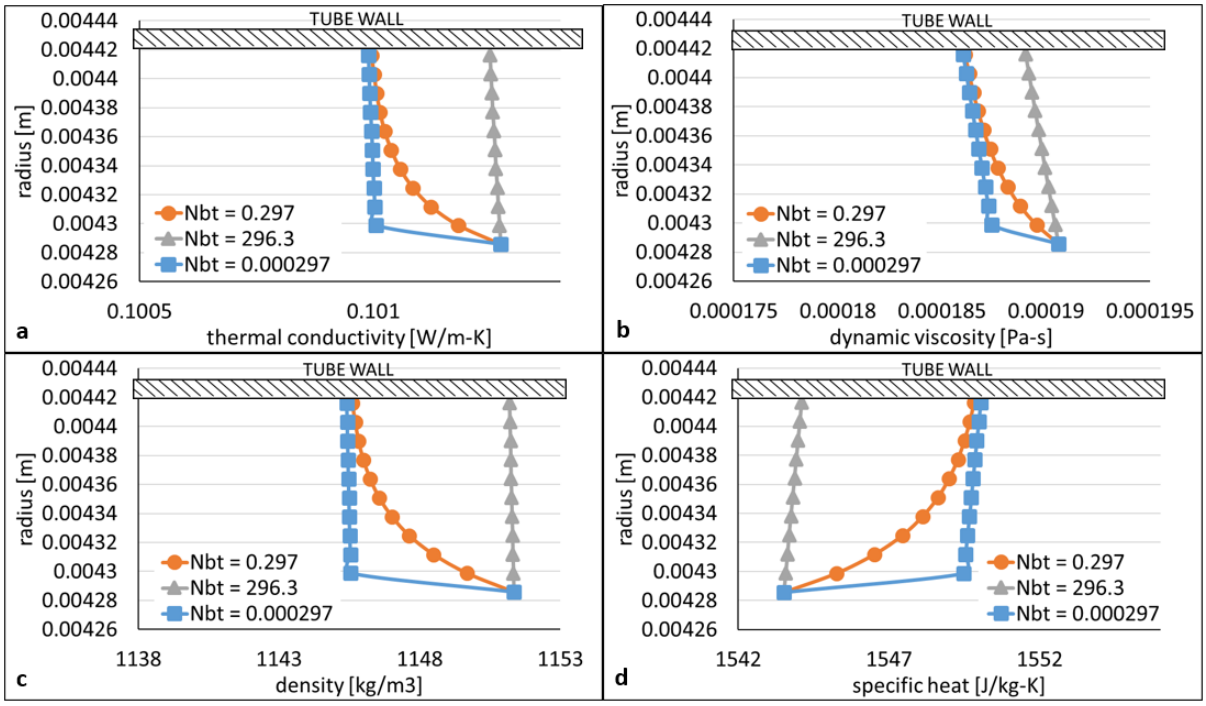


Figure 6.9: Thermophysical properties ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) within the laminar sublayer as a function of N_{BT}, for the case of a R410A-lubricant based nanofluid (OMF = 3%, NMF = 20%)

The trends of Figures 6.8 and 6.9 are similar, although all properties show slightly higher values for the 3% oil mass fraction case. It should also be observed that the range of variation of the calculated properties is very small compared to the pure refrigerant based nanofluid of Figure 6.7. This behavior was justified by the relatively smaller overall nanoparticle concentration in the refrigerant-nanolubricant case. In fact, if in Figure 6.7 the volume concentration (ϕ_b = 0.01) is relative to the base fluid, in Figures 6.8 and 6.9 the nanoparticle mass concentration is 20% in oil, where oil is only 1 or 3% of the refrigerant mass. The overall volume concentration with respect to the refrigerant-lubricant mixture was estimated to be about 0.00064 for 1% OMF, and 0.0019 for 3% OMF. Therefore in presence of oil, nanoparticles are very diluted and their estimated impact on thermophysical properties is small.

Single Phase Velocity Profile

The results of the radial analysis of the laminar sublayer for the single phase case are first presented for water based nanofluid and pure refrigerant based nanofluid. A second analysis is reported for oil-refrigerant based nanofluid.

The velocity profile for turbulent flow can be described by the von Karman equations (Hewitt and Hall-Taylor, 1970) and it is often referred to as the universal velocity profile. This profile is defined by three equations for turbulent region, buffer layer and laminar layer, respectively reported here:

$$u^+ = y^+ \qquad y^+ < 5 \qquad (laminar\ \ layer) \tag{6.1}$$

$$u^+ = -3.05 + 5lny^+ \qquad 5 < y^+ < 30 \qquad (buffer\ \ layer) \tag{6.2}$$

$$u^+ = y^+ \qquad y^+ > 30 \qquad (turbulent) \tag{6.3}$$

where $u^* = \sqrt{\tau_{wall}/\rho}$ is friction velocity, $u^+ = u/u^*$ is a dimensionless velocity parameter, and $y^+ = u^*\rho y/\mu$ is a dimensionless friction distance parameter. The representation

of a conventional velocity profile is represented in Figure 6.10, where the three regions are represented by blue squares (laminar layer), grey triangles (buffer layer), and orange circles (turbulent region).

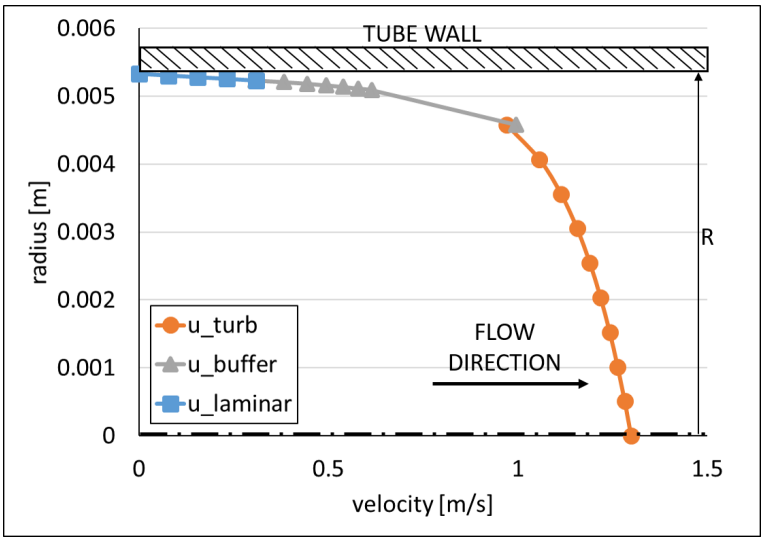


Figure 6.10: Conventional velocity profile of a single phase fluid inside a tube.

In Buongiorno's analysis the laminar and buffer layers are considered as one, with a direct transition to turbulent flow. It should be noticed that Figure 6.10 is also representative of the velocity profile of a nanofluid, when Equations 6.1, 6.2, and 6.3 are calculated as a function of bulk properties (i.e. $\phi_{nano} = \phi_{bulk} = constant$). However, when analyzing the nanoparticles radial distribution, properties were calculated as a function of the local nanoparticle concentration (ϕ_{local}), and the velocity profile resulting from the radial analysis changed as reported in Figure 6.11(a) (represented by dark blue circles). A zoom in on the velocity profile within the laminar sublayer is represented in Figure 6.11(b).

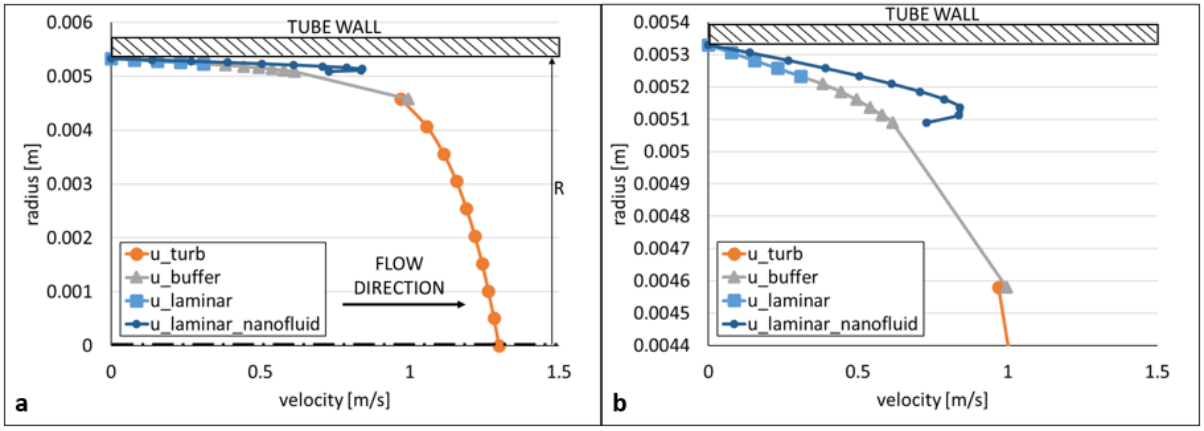


Figure 6.11: (a) Velocity profile obtained from radial analysis in case of presence of nanoparticles and (b) zoom into the laminar region.

The velocity profile of Figure 6.11 shows how the distribution of nanoparticles impacts the local thermophysical properties and therefore the fluid behavior. In particular, depending on the value of the parameter N_{BT}, nanoparticles tend to move away from the wall and increase their concentrations at the laminar-turbulent interface. For this reason, the viscosity of the fluid near the wall (where $\phi_{local} < \phi_{bulk}$) will be lower than the viscosity of the bulk fluid, and according to Equation 3.32, the fluid velocity will be higher.

The velocity profile described in Figure 6.11 should not be interpreted as the actual velocity profile of the fluid (that is, the fluid is not expected to actually slow down), but rather as the mathematical confirmation of the impact that nanoparticles can have within a laminar region. This result is also in agreement with what previously reported (see Section 5.3.3) about the study of laminar flow of suspensions of polystyrene spheres (Ahuja, 1975b,a) where it was measured a flattening of the typical Poiseuille flow parabolic velocity profile. Ahuja suggested the formation of a particle-free low-viscosity layer near the tube wall because of the migration of the rigid spheres to the center of the flow. A particle-free layer induces a lubricating effect that causes a drop in pressure gradient at the wall and a change in the velocity profile, exhibiting the characteristic of a plug flow, where the core flow has a higher viscosity than the liquid film surrounding it. In this study, the migration of nanoparticles is not induced by rotation or translation (as for Ahuja's case), but rather by thermophoresis. For these reasons, this result was considered explanatory of the behavior observed experimentally, regarding the relatively small

increase in pressure drop when testing nanofluids, compared to the pressure drop measured for the same nanoparticle-free fluid.

Figure 6.12 reports the effect of the variation of Al_2O_3 nanoparticle concentration on the velocity profile of a single phase liquid flow of refrigerant R410A. The velocity profile obtained by the radial analysis is represented by the green line. In Figure 6.12 (a) and (b), the nanoparticles concentration was increased respectively from $\phi = 0.01$ to $\phi = 0.03$. The value of N_{BT} reported in Figure (b) changes slightly because of the change in viscosity. In Figure 6.12 (c) and (d), while maintaining constant $\phi = 0.01$, N_{BT} was respectively increased ($N_{BT} = 175.77$) and reduced ($N_{BT} = 0.0005381$).

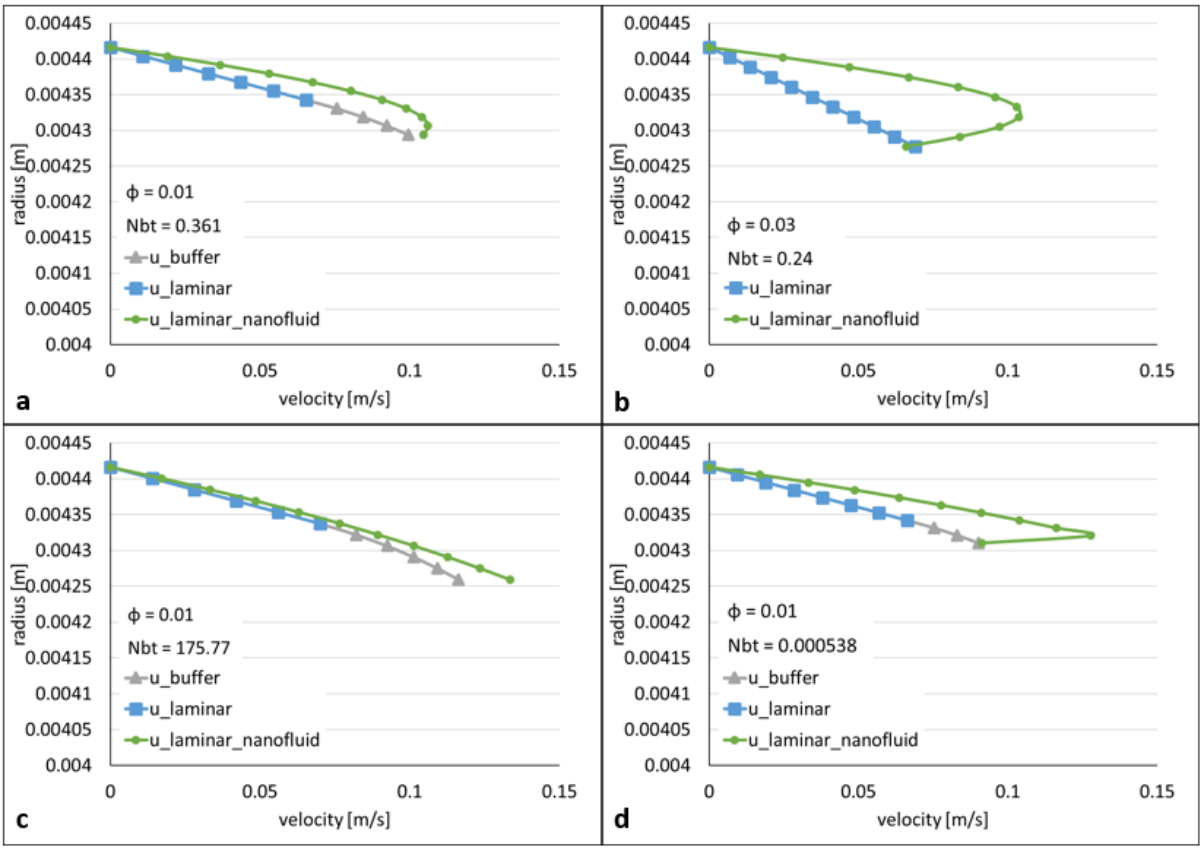


Figure 6.12: R410A nanofluid velocity profile variation at different concentrations ((a) $\phi = 0.01$, (b) $\phi = 0.03$) and at different values of N_{BT} ((c) $N_{BT} = 175$, (d) $N_{BT} = 0.0005381$).

From the behavior represented in Figures 6.12(a) and (b) it can be said that a decrease in nanoparticle volume concentration will cause the velocity profile to collapse to the linear velocity profile calculated at bulk properties, but also representative of a fluid without nanoparticles.

Figures 6.12(c) and (d) resemble the trends observed in Figure 6.7 for thermophysical properties: at high values of N_{BT} the distribution of nanoparticles will be uniform and the profile will be again similar to the profile of a nanoparticle-free fluid; at low values of N_{BT} nanoparticles will tend to move away from the wall, creating a low concentration sublayer before the fluid enters the buffer-turbulent region where the bulk properties cause a sudden change in velocity.

In Figures 6.13 and 6.14 are reported the velocity profiles for a single phase Al_2O_3 nanolubricant-refrigerant mixture, at 1 and 3% OMF. The velocity profile obtained by the radial analysis is represented by the green line. In Figures (a) and (b), the nanoparticles mass concentration in oil was increased respectively from $NMF = 20\%$ to $NMF = 40\%$. The corresponding mixture volume concentrations for 1% OMF changed from $\phi = 0.00065$ to $\phi = 0.00128$; for 3% OMF changed from $\phi = 0.0019$ to $\phi = 0.0038$. In Figures (c) and (d), N_{BT} was respectively increased and reduced, while ϕ was maintained constant.

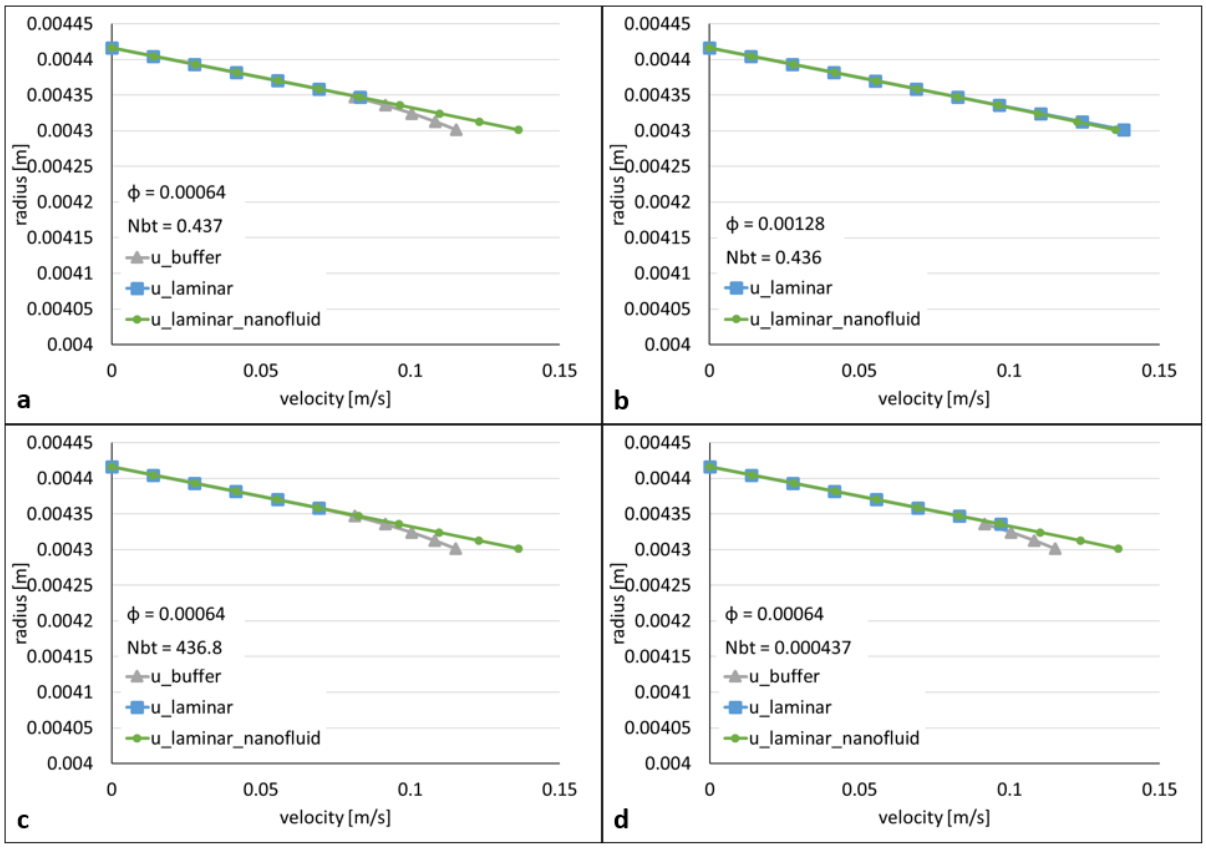


Figure 6.13: 1% OMF nanolubricant-refrigerant velocity profile variation at different concentrations ((a) $\phi = 0.00065$, (b) $\phi = 0.00128$) and at different values of N_{BT} ((c) $N_{BT} = 436.8$, (d) $N_{BT} = 0.000437$).

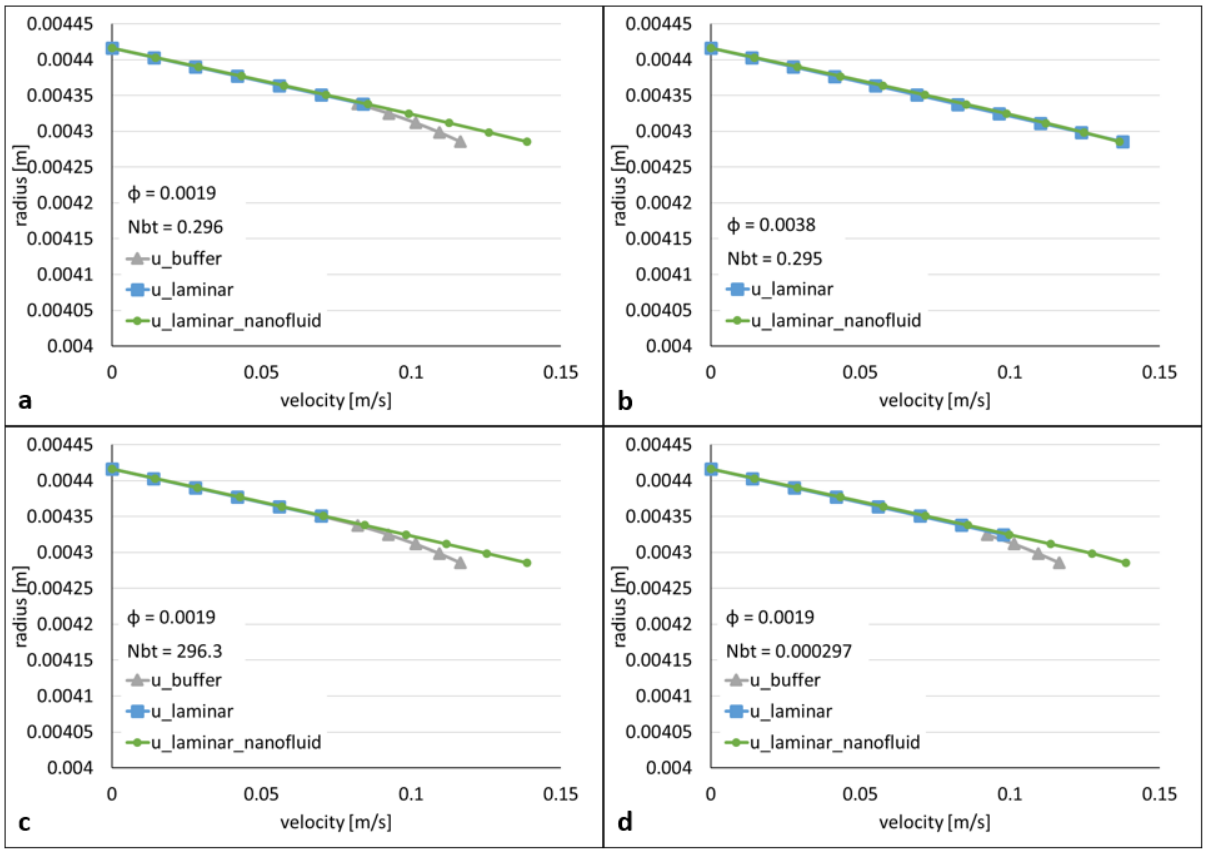


Figure 6.14: 3% OMF nanolubricant-refrigerant velocity profile variation at different concentrations ((a) $\phi = 0.0019$, (b) $\phi = 0.0038$) and at different values of N_{BT} ((c) $N_{BT} = 296.3$, (d) $N_{BT} = 0.000297$).

The velocity profiles calculated in Figures 6.13 and 6.14 are all flat to the velocity profile calculated at bulk thermophysical properties and radial analysis does not show a change in gradient when nanoparticles change their distribution or concentration. The reason for this behavior is found in the low concentration of nanoparticles having a small impact on thermophysical properties, as previously observed in Figures 6.8 and 6.9. This is also the reason why the value of N_{BT} does not change between Figures (a) and (b) when the nanoparticle mass fraction in oil is increased from 20% to 40%.

In order to exercise the simulation, an additional analysis was conducted to understand what is the minimum nanoparticle concentration in oil needed to observe a change in the thermophysical properties of the mixture. Figure 6.15 reports the outcome of this analysis. In this case the oil mass fraction was 10% and the nanoparticle mass fraction in oil was 50% (corresponding to a mixture volume concentration of $\phi = 0.016$).

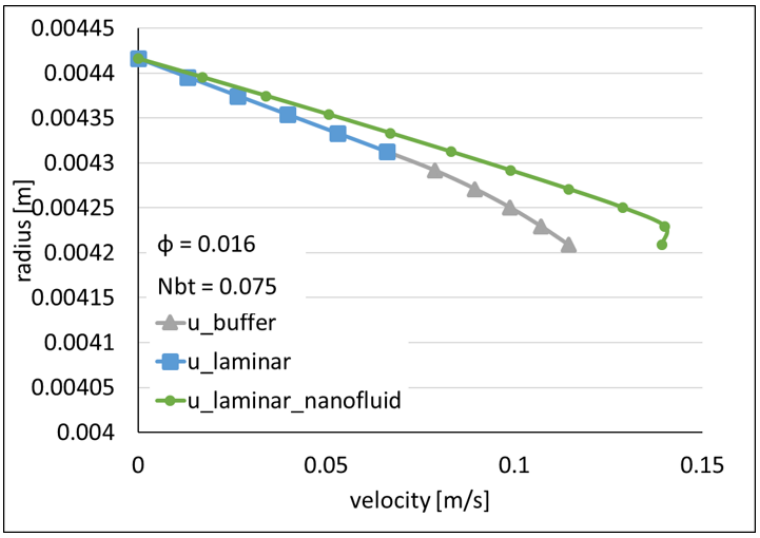


Figure 6.15: 10% OMF nanolubricant-refrigerant velocity profile variation at $\phi = 0.016$ and $N_{BT} = 0.075$.

From a comparison with the pure refrigerant based nanofluid of Figure 6.12 it could be speculated that Al_2O_3 nanoparticles have a measurable effect on thermophysical properties when their bulk volume concentration (ϕ) is about 0.01 or higher. However, according to the analysis conducted, and for the case of nanolubricant-refrigerant mixtures studied in this dissertation, in order to achieve a mixture bulk volume concentration of about 0.01, the oil mass fraction should be about 10%. This value of oil concentration is beyond the range of oil concentrations studied in this work.

6.3.2 Two-Phase Radial Analysis

In this section the results of the radial analysis for a two-phase flow are presented. Similarly to what discussed for the case of a single phase flow, the investigation was made with respect to the change of thermophysical properties, and the impact on the velocity profile within the liquid film. The diameter was chosen to be equal to the equivalent length ($D_e = 8.8\ mm$, as reported in Table 4.1).

Two-Phase Thermophysical Properties Gradient

A sensitivity analysis was performed to observe the way thermophysical properties change within the liquid film as a function of the distribution of nanoparticles. For this analysis, a

set of tests was chosen to be more meaningful to the particular investigation conducted in this dissertation. In particular, thermophysical properties were estimated at a lower and a higher mass flux (250 kg/m^2s and 373 kg/m^2s), and at low and high thermodynamic quality (x ~ 0.2 and x ~ 0.8). Saturation temperature (4°C) and heat flux (12 kW/m^2) were kept constant. The investigation was conducted first for a oil-free case, and the thermophysical properties were estimated with the set of correlations presented by Buongiorno (2006). A second analysis is then reported for oil-refrigerant based nanofluid, with tests at low and high oil mass fraction (1 and 3%), and at low and high nanoparticle mass fraction (20% and 70%). In this case, the thermophysical properties were estimated with the set of correlations presented in Section 5.1.

For the liquid film of a R410A based nanofluid charged with Al_2O_3 nanoparticles, Figures 6.16 and 6.17 present the estimated trends of thermophysical properties ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat), respectively at thermodynamic quality of 0.2 and 0.8.

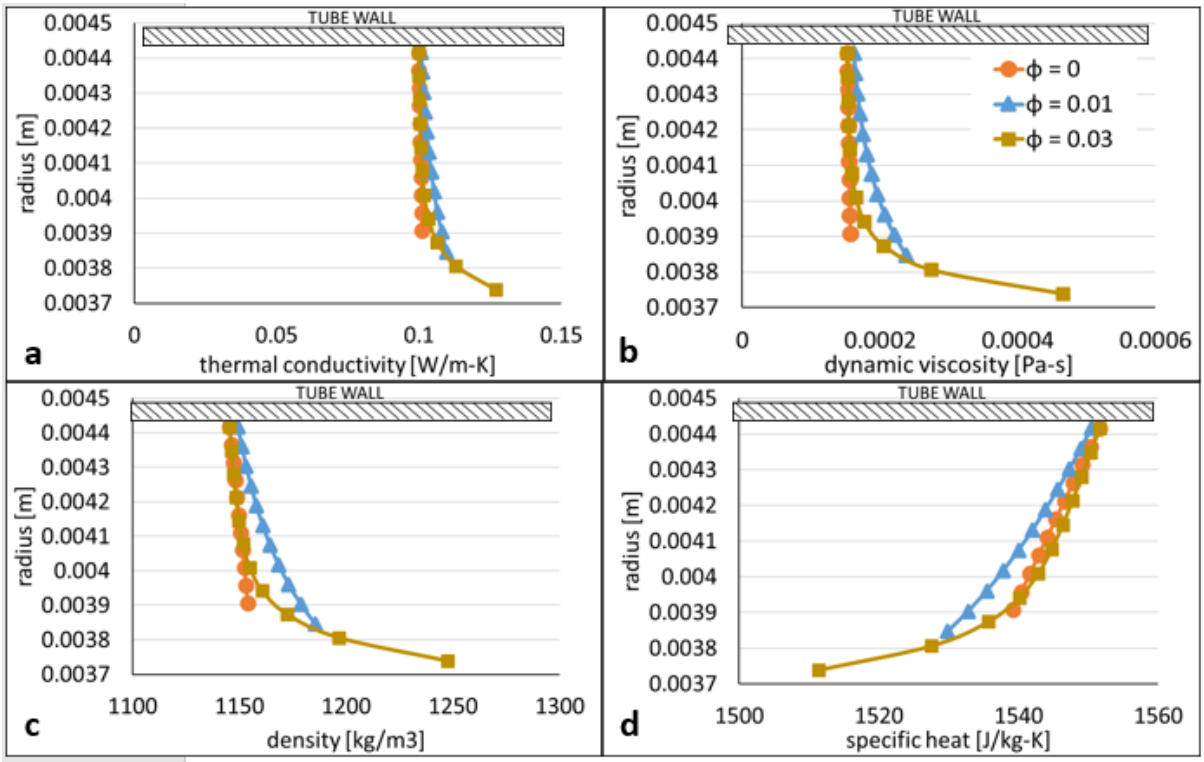


Figure 6.16: Thermophysical properties of a R410A based nanofluid ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) calculated at different volume concentrations (ϕ = 0, 0.01, and 0.03), at x = 0.2, and 250 kg/m^2s mass flux.

The y-axes represents the height of the liquid film thickness from the wall. The results are calculated at 250 kg/m^2s mass flux, and they are presented for different volume concentrations (ϕ = 0, 0.01, and 0.03, respectively represented by orange circles, blue triangles, and yellow squares), and for a limited range of N_{BT} ($0.13 < N_{BT} < 3.5$).

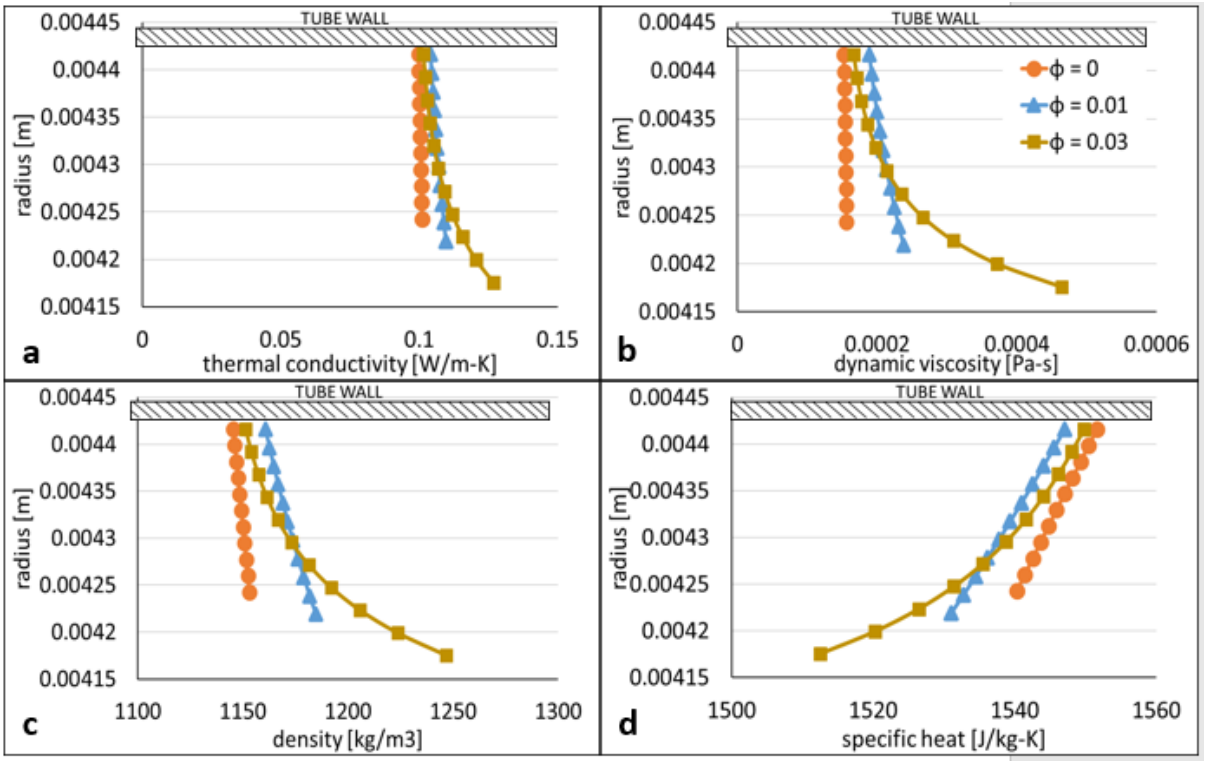


Figure 6.17: Thermophysical properties of a R410A based nanofluid ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) calculated at different volume concentrations (ϕ = 0, 0.01, and 0.03), at x = 0.8, and 250 kg/m^2s mass flux.

Figures 6.16 and 6.17 show that when ϕ = 0, the thermophysical properties trends are linear, as they change only in function of the temperature gradient through the liquid film, and that was assumed to be always linear in this work. The two-phase flow experiments collected also showed an average temperature difference between the wall and the saturation (or bubble) temperature of only 2°C. It is therefore expected that the base fluid properties will not change considerably by effect of the temperature gradient only. In both figures there is clear difference between the blue curve representing $\phi = 0.01$ and the yellow curve representing a higher volume concentration ($\phi = 0.03$).

In particular, it could be observed how at lower quality (x = 0.2, Figure 6.16) the yellow curve tends to overlap with the orange curve (representing the particle-free fluid) more than

the blue curve. This result was interpreted as the consequence of the combination of at least two facts: (i) that a higher nanoparticles concentration has a larger impact on thermophysical properties, and consequently (ii) that the thickness of the liquid film varies as a function of the liquid film mass flow rate.

It should be noted here that, while the literature agrees on the fact that in annular flows the film thickness grows smaller at higher mass fluxes, there are contradicting results regarding the effect of viscosity on the film thickness. Some authors reported a thinning of the film with increasing viscosity (Asali et al., 1985), while others observed the opposite (Hori et al., 1979; Furukawa and Fukano, 2001). Finally, other authors stated that the change in film thickness is dependent on both viscosity and surface tension (Yoshinaga et al., 2014). In this study, the simulation calculated an increase in thickness when the nanoparticle concentration increased (see Figure 6.16). Consequently, due to both larger thickness and higher viscosity, the parameter N_{BT} will be smaller, meaning that nanoparticles will move away from the wall more effectively, leaving behind an almost "particle-free" liquid. This effect is not visible when $\phi = 0.01$ because the initial nanoparticle concentration might be too small.

As the evaporation process continues and the thermodynamic quality increases, the thickness of the liquid layer decreases. Assuming that nanoparticles only remain in the liquid phase, if the liquid evaporates, the relative nanoparticle concentration increases. However, because the effect of the decrease of thickness is stronger than the relative increase of viscosity, N_{BT} will become larger, causing the nanoparticles to be homogeneously distributed. This result is observed in the straight blue line of Figure 6.17, while the yellow line starts behaving like the blue line when the quality was at x = 0.2.

The results reported in Figures 6.18 and 6.19 are those for a higher mass flux (G_{flux} = 373 kg/m^2s). At higher mass flux there was no sensible change in trends with respect to Figures 6.16 and 6.17, except for an expected small reduction of the film thickness.

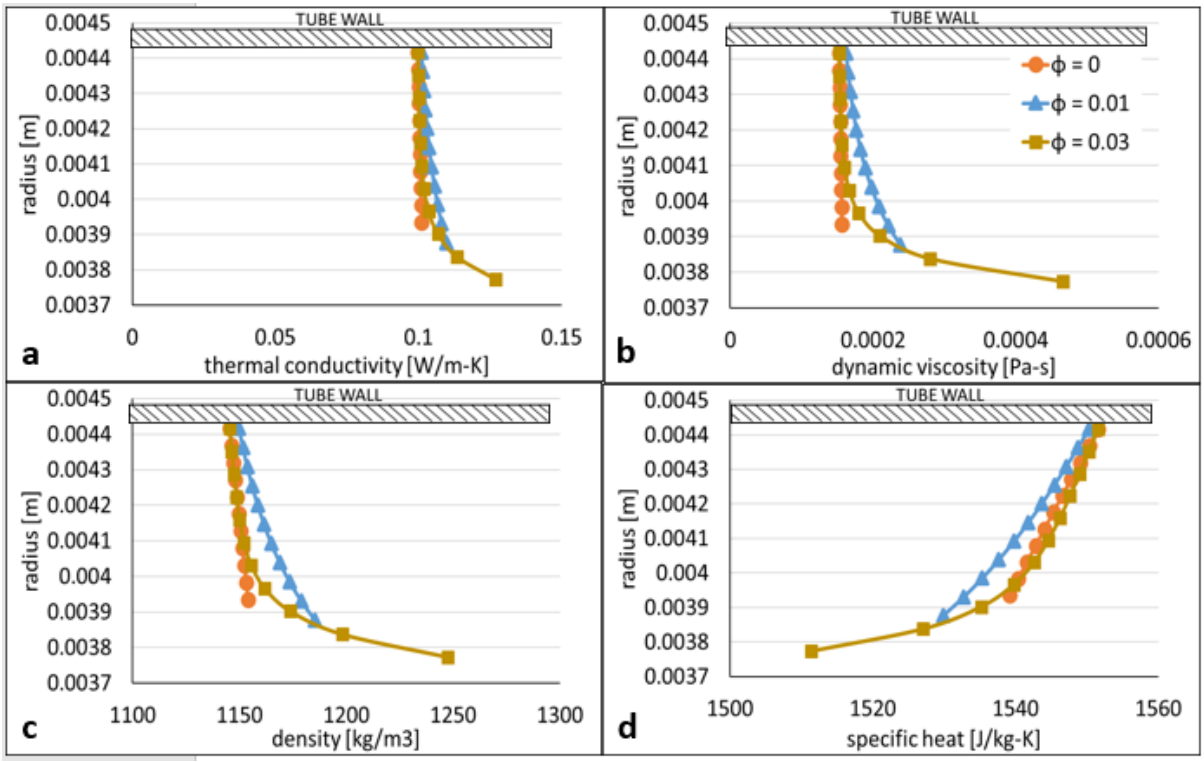


Figure 6.18: Thermophysical properties of a R410A based nanofluid ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) calculated at different volume concentrations (ϕ = 0, 0.01, and 0.03), at x = 0.2, and 373 kg/m^2s mass flux.

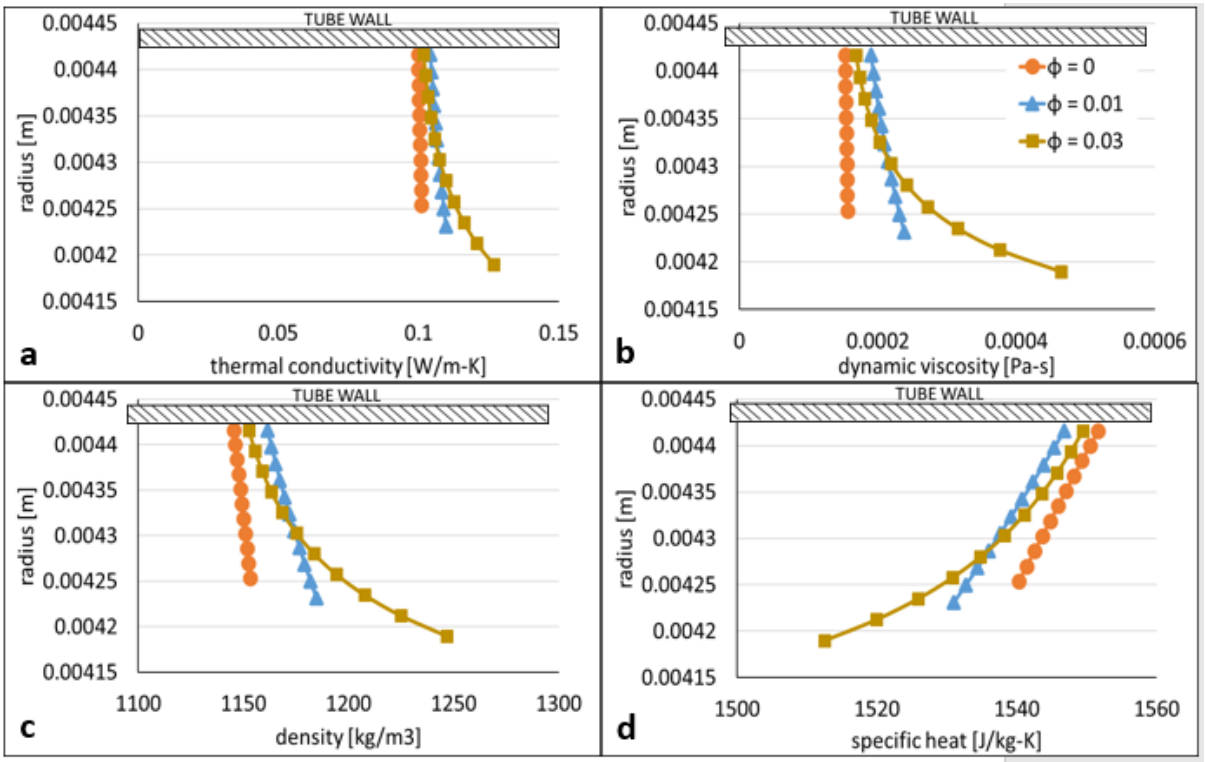


Figure 6.19: Thermophysical properties of a R410A based nanofluid ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) calculated at different volume concentrations (ϕ = 0, 0.01, and 0.03), at x = 0.8, and 373 kg/m^2s mass flux.

The two-phase radial analysis was conducted also for a refrigerant-nanolubricant mixture. The simulation tests were conducted for a 250 kg/m^2s mass flux, and oil mass fraction of 1% and 3%. The nanoparticle mass fraction was tested at 0%, 20% and 70%. Figures 6.20 and 6.21 report the estimated trends of thermophysical properties, respectively at thermodynamic quality of 20% and 80%. In these figures, the colored lines represent the three mass fractions tested (in orange circles is the 0%, in blue triangles is the 20% and in yellow squares is the 70%). It should be noted that in this case the concentrations are reported in mass fraction and not in volume fractions as in previous figures. This was done for easiness of reading, given that the volume concentration changes with oil mass fraction and with quality (being larger at higher OMF and at higher qualities). The range of N_{BT} was: $0.10 < N_{BT} < 0.26$.

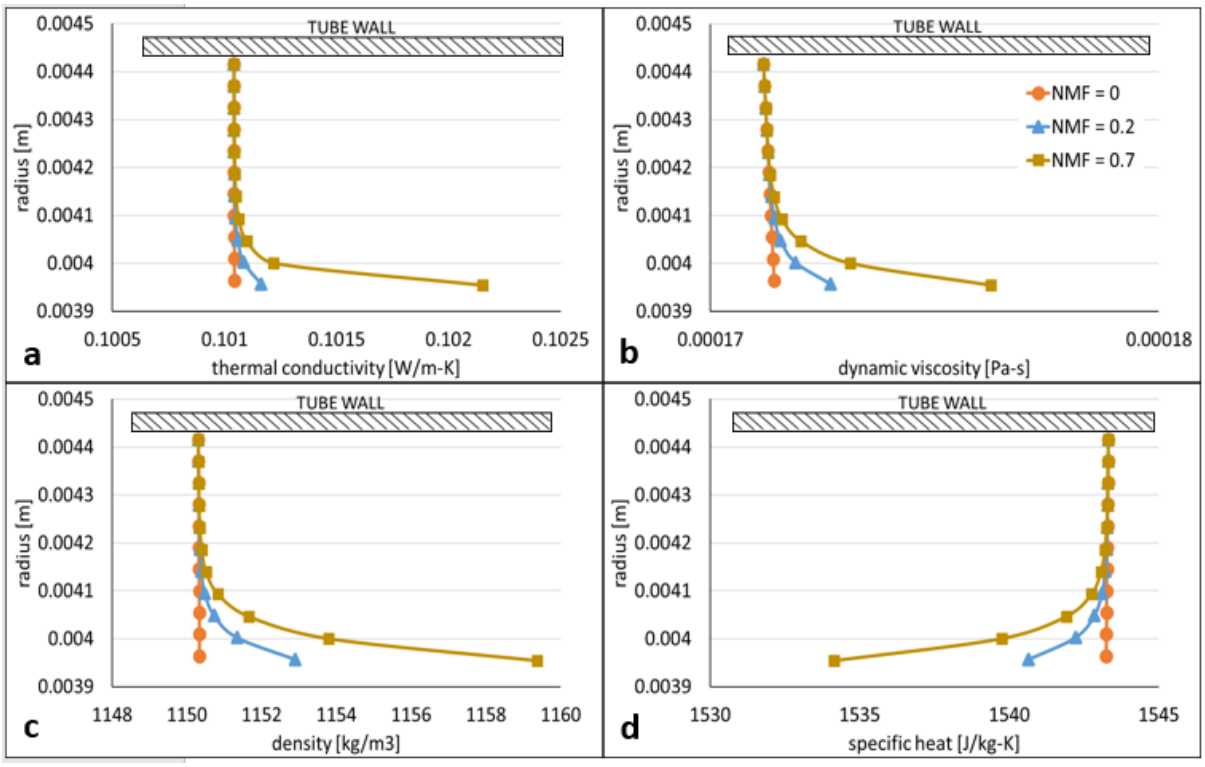


Figure 6.20: Thermophysical properties of a R410A based nanofluid ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) calculated for OMF = 1%, at different mass concentrations (NMF = 0, 0.2, and 0.7), at x = 0.2, and 250 kg/m^2s mass flux.

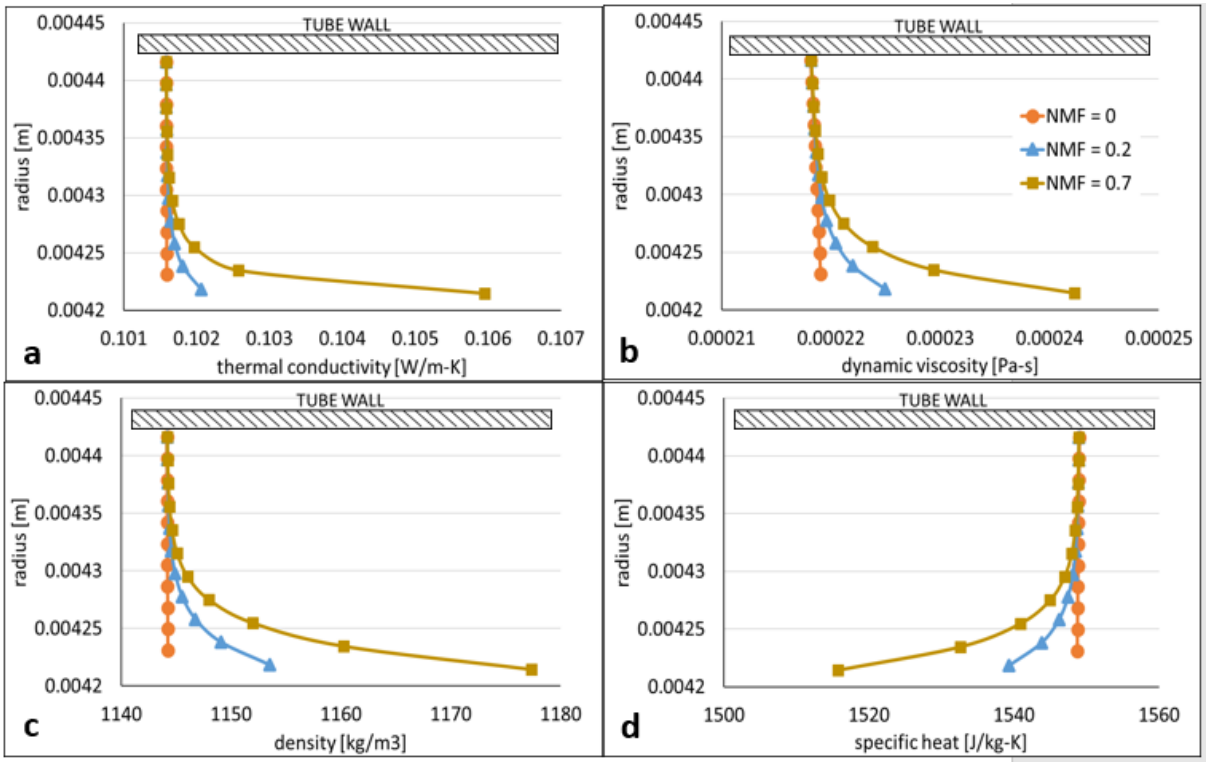


Figure 6.21: Thermophysical properties of a R410A based nanofluid ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) calculated for OMF = 1%, at different mass concentrations (NMF = 0, 0.2, and 0.7), at x = 0.8, and 250 kg/m^2s mass flux.

These results are in line with those of Figure 6.8for a single phase mixture of refrigerant and nanolubricant. The reason of this behavior is to be found in the relatively smaller overall nanoparticle volume concentration in the refrigerant-nanolubricant case, estimated to be about 0.00086 and 0.00312, respectively at lower and higher quality, for the case of NMF = 20%; and 0.003 and 0.011, respectively at lower and higher quality, for the case of NMF = 70%. Therefore in presence of oil, nanoparticles are very diluted and their estimated impact on thermophysical properties is small.

For values of 3% oil mass fraction the estimated thermophysical properties are represented in Figures 6.22 and 6.23 for low and high quality. Similarly to what observed for the two-phase 1% OMF case, and for the single phase mixture of refrigerant and nanolubricant of Figure 6.9, the thermophysical properties seem to be affected by the presence of nanoparticles only far away from the wall where the concentration becomes higher (in these figures the range of N_{BT} was slightly smaller than the case presented for 1%OMF).

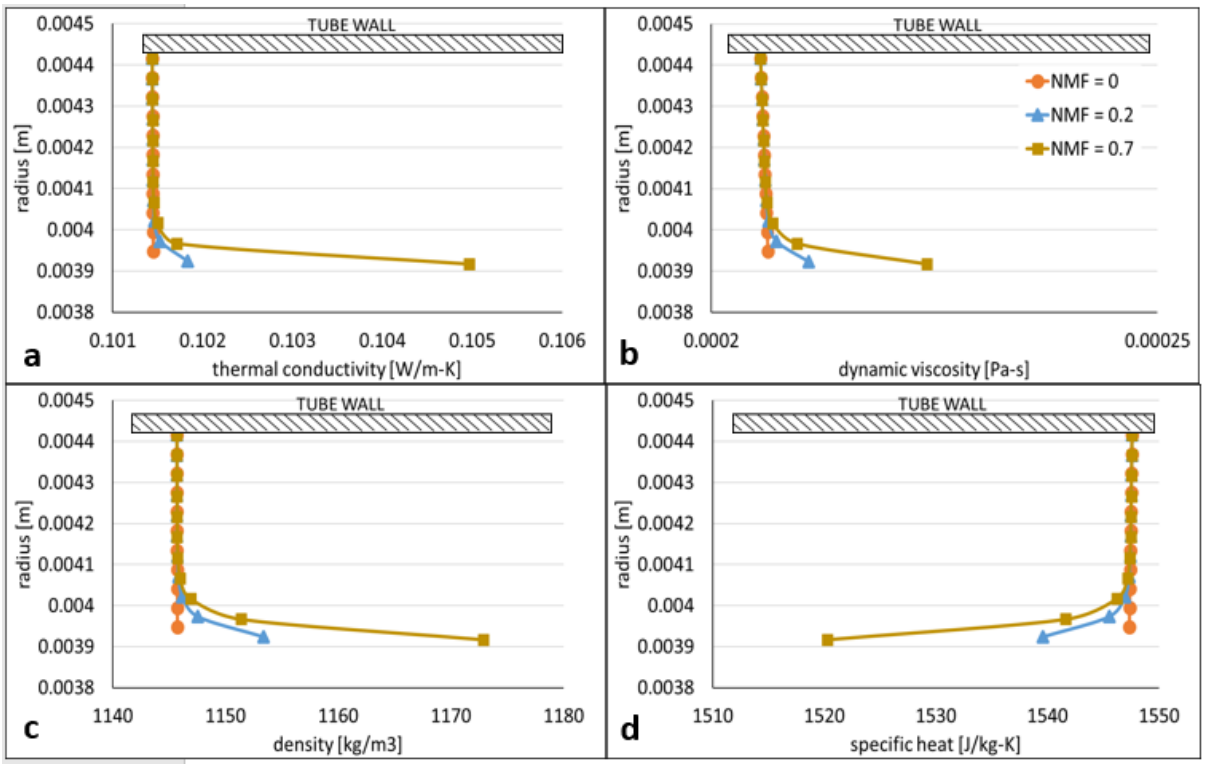


Figure 6.22: Thermophysical properties of a R410A based nanofluid ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) calculated for OMF = 3%, at different mass concentrations (NMF = 0, 0.2, and 0.7), at x = 0.2, and 250 kg/m^2s mass flux.

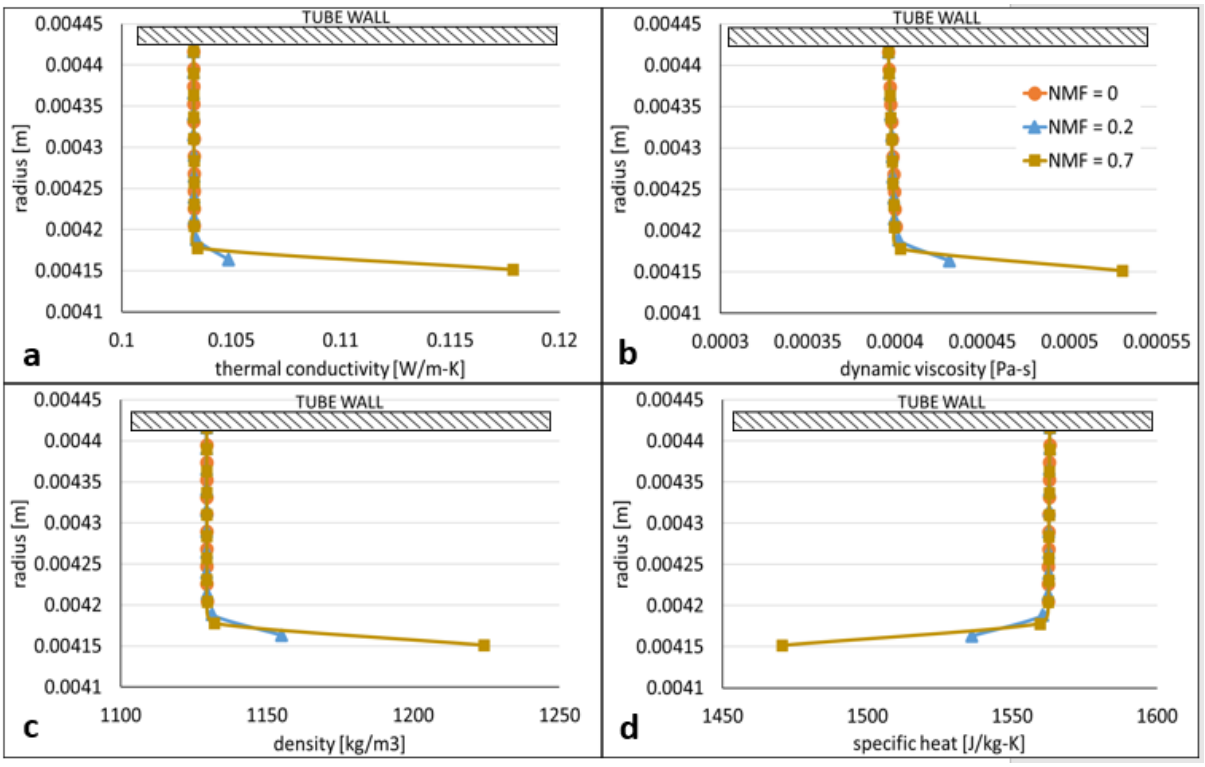


Figure 6.23: Thermophysical properties of a R410A based nanofluid ((a) thermal conductivity, (b) dynamic viscosity, (c) density, (d) specific heat) calculated for OMF = 3%, at different mass concentrations (NMF = 0, 0.2, and 0.7), at x = 0.8, and 250 kg/m^2s mass flux.

Simulations were performed also at a higher mass flux (373 kg/m^2s) but the results are not reported here for conciseness, being the outcomes very similar to the ones already described for the lower mass flux. The only notable difference observed was a small reduction of the film thickness.

Two-Phase Velocity Profile

The results of the radial analysis on the velocity profile are reported first for the pure refrigerant based nanofluid. Figure 6.24 reports the velocity profiles estimated at thermodynamic quality of 0.2 and 0.8. The results are calculated at 250 kg/m^2s mass flux, and they are presented for three volume concentrations: 0, 0.01, and 0.03, respectively represented by orange circles, blue triangles, and yellow squares. Finally, the impact of different ranges of the parameter N_{BT} are observed ((a) and (b): $1.35 < N_{BT} < 35$; (c) and (d): $0.13 < N_{BT} < 3.5$, (e) and (f):$0.0013 < N_{BT} < 0.035$).

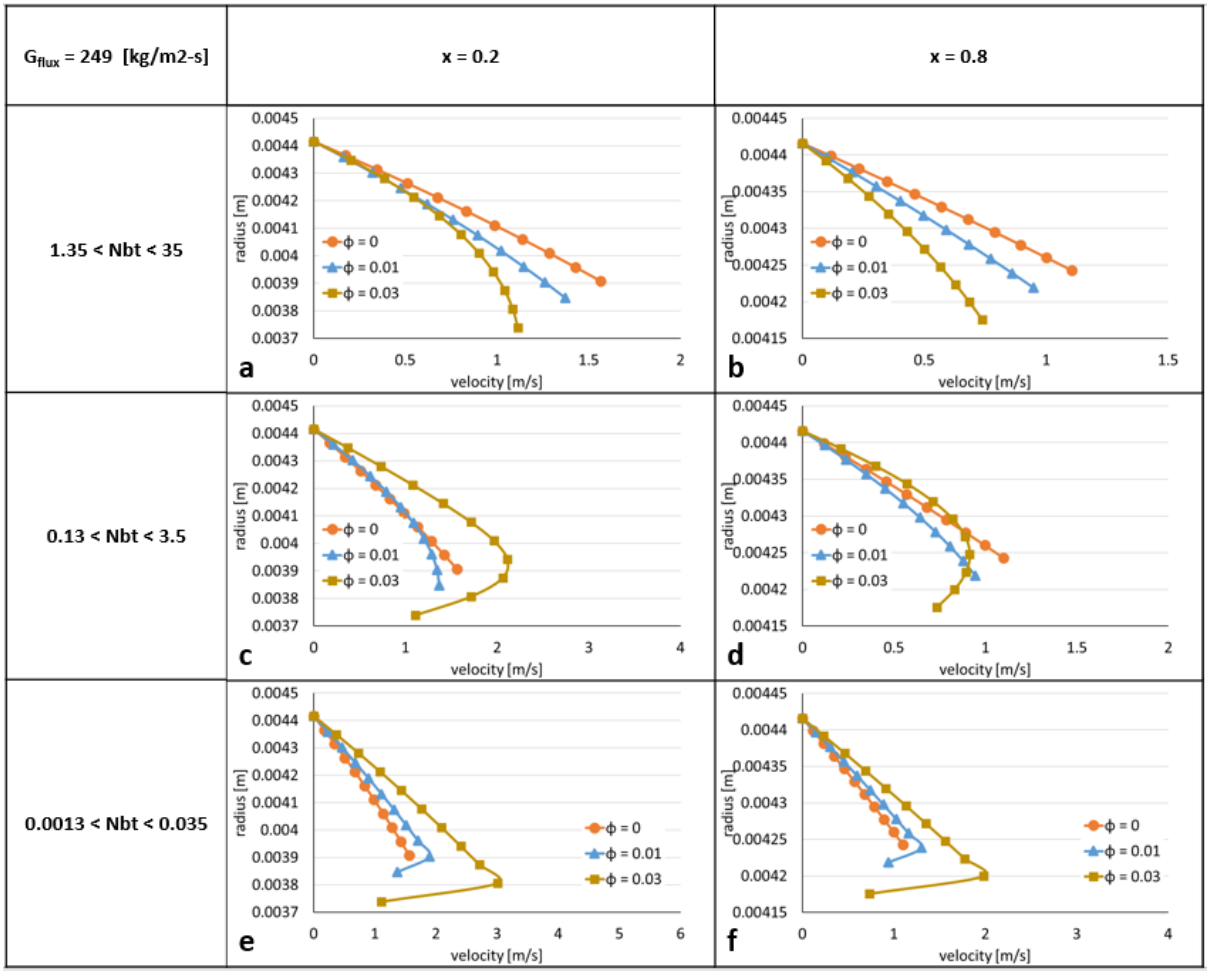


Figure 6.24: Velocity profiles of a R410A based nanofluid for different N_{BT} ranges ((a) and (b): $1.35 < N_{BT} < 35$; (c) and (d): $0.13 < N_{BT} < 3.5$, (e) and (f):$0.0013 < N_{BT} < 0.035$), calculated at different volume concentrations (ϕ = 0, 0.01, and 0.03), at x = 0,2 and 0.8, and 250 kg/m^2s mass flux.

In Figure 6.24, the nanofluids with lower and higher concentrations (marked with blue triangles and orange circles) show different trends depending on the value of N_{BT}. In particular, for values of N_{BT} smaller than 3.5 (Figures (c), (d), (e) and (f)) the simulation predicted higher velocities at higher concentrations, especially at lower qualities. Similarly to what observed for the single phase case, the velocity profile described in Figure 6.24 should not be interpreted as the actual velocity profile of the fluid (that is, the fluid is not expected to slow down), but rather as the mathematical confirmation of the impact that nanoparticles can have within the liquid film.

The migration of nanoparticles could create a particle-free layer, inducing a lubricating effect that causes a drop in pressure gradient at the wall and a change in the velocity profile. Interestingly however, when N_{BT} is greater than 1.36 (Figures (a) and (b)) the trends are inverted and the higher concentration fluid is slower. Considering the observations made in the previous section on the effect of thermophysical properties, the velocity profile is also affected by the nanoparticle concentration: when the concentration is small (the blue lines), the velocity profile will not change much compared to the particle-free case. Also, when N_{BT} is higher than unity, the particle distribution will be uniform, showing no gradient in the change of thermophysical properties. However, because viscosity is more dependent on ϕ, the overall velocity profile will show a degradation.

For higher mass flux, the results are found in Figure 6.25. No sensible change is found in velocity trends with respect to Figures 6.24, except for an expected reduction of the film thickness.

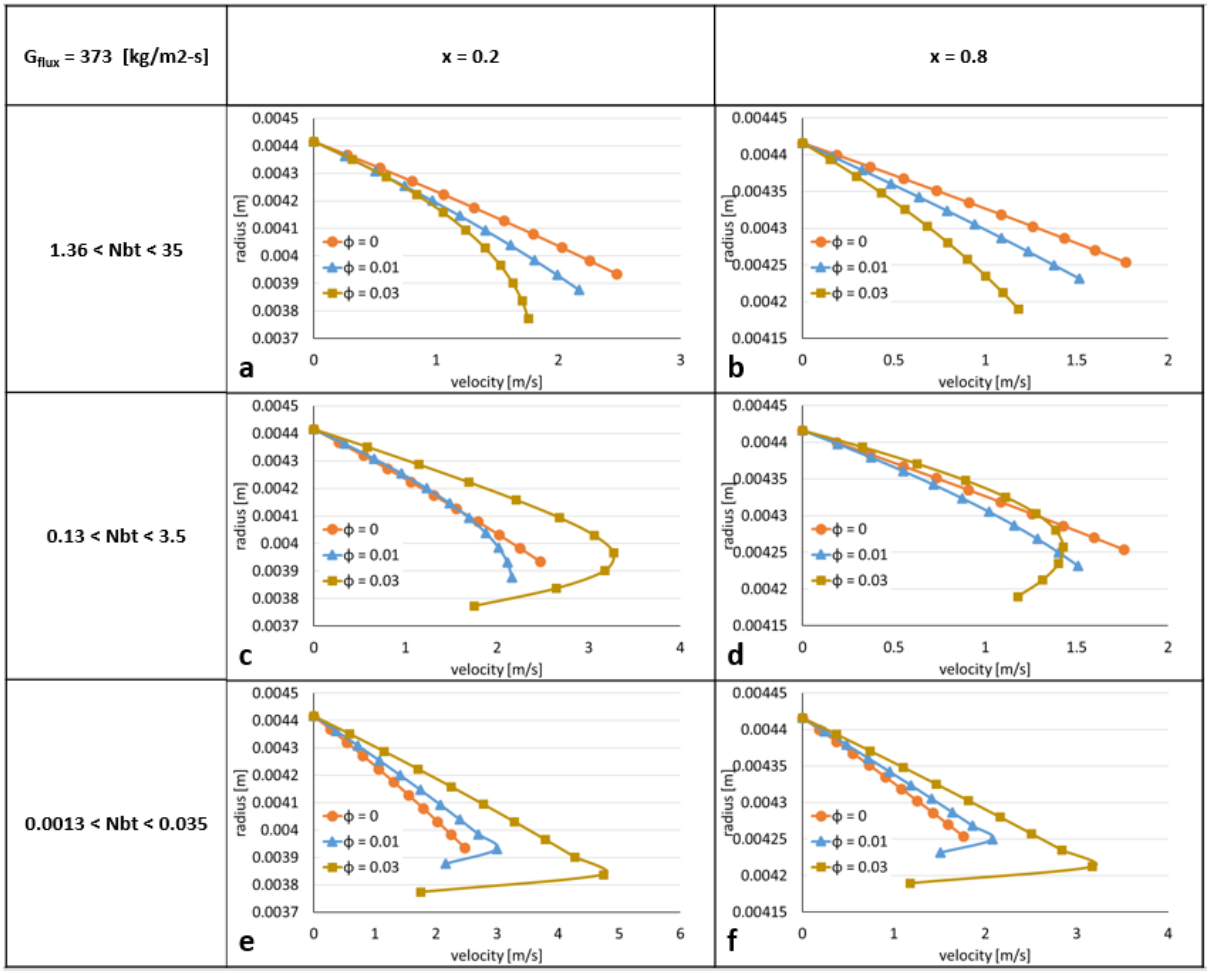


Figure 6.25: Velocity profiles of a R410A based nanofluid for different N_{BT} ranges ((a) and (b): $1.35 < N_{BT} < 35$; (c) and (d): $0.13 < N_{BT} < 3.5$, (e) and (f):$0.0013 < N_{BT} < 0.035$), calculated at different volume concentrations (ϕ = 0, 0.01, and 0.03), at x = 0.2 and 0.8, and 373 kg/m^2s mass flux.

The velocity profiles for a 250 kg/m^2s mass flux two-phase flow of Al_2O_3 nanolubricant-refrigerant mixture are shown in Figure 6.26. The profiles are plotted for two thermodynamic qualities (x = 0.2 and x = 0.8), and different nanoparticle mass fractions: in orange circles is NMF = 0%, in blue triangles is NMF = 20%, and in yellow squares is NMF = 70%. In Figures (a) and (b) are represented the estimated profiles for 1% OMF (N_{BT} was between 0.1 and 0.25). The velocity gradient changed slightly only at higher quality where the volume concentration is higher. Figures (c) and (d) show the velocity profiles at 3% OMF (and for N_{BT} between 0.03 and 0.13). Once again, because the nanoparticles volume concentration in the oil-refrigerant mixture increases when the refrigerant evaporates, the larger impact of nanoparticles on the flow velocity was found to be larger at higher qualities.

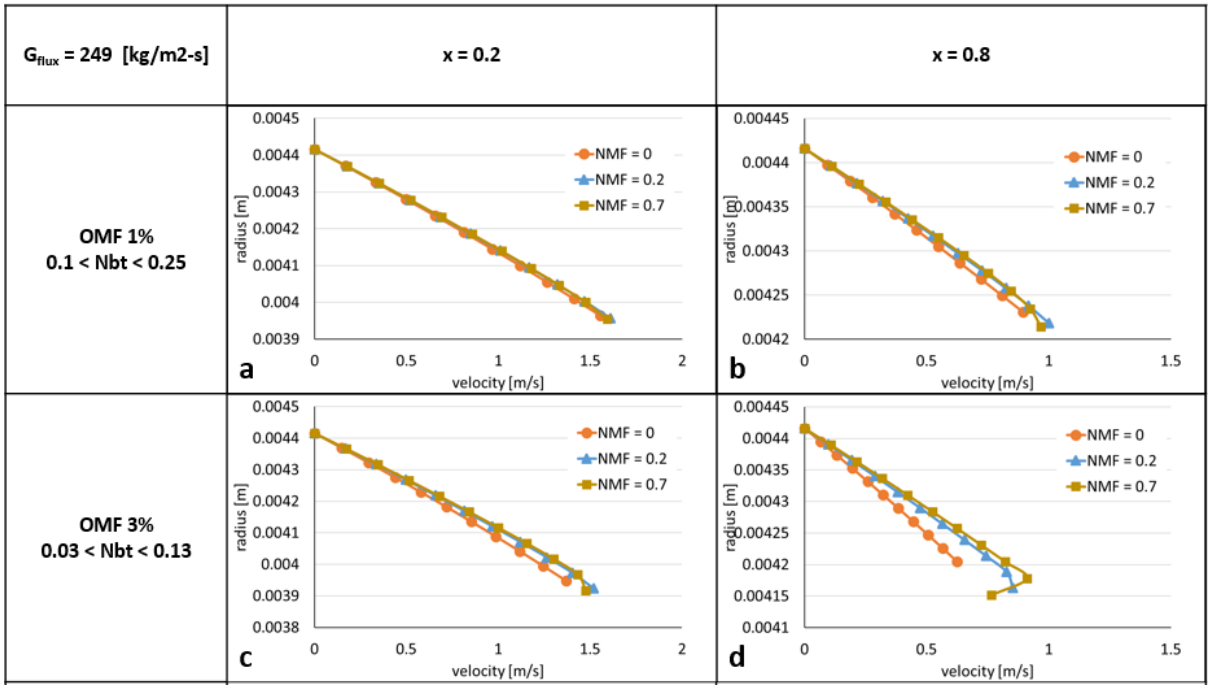


Figure 6.26: Velocity profiles of a 250 kg/m^2s mass flux two-phase flow of R410A-oil mixture at different oil concentrations ((a) and (b): 1% OMF, $0.1 < N_{BT} < 0.25$; (c) and (d): 3% OMF, $0.03 < N_{BT} < 0.13$), calculated at different nanoparticle mass concentrations (NMF = 0, 0.2, and 0.7), at x = 0.2 and x = 0.8

Results are reported only for low mass flux because at higher mass flux the results were found to be analogous. The ranges of N_{BT} chosen for the previous discussion were considered to more meaningful in the analysis of radial behavior. When N_{BT} reaches higher or lower values than the ones presented, the profiles will collapse to the cases observed previously in Figures 6.13 and 6.14 for single phase.

6.3.3 Convective Heat Transfer Analysis

According to Buongiorno's model, the Nusselt number of a nanofluid convective flow can be calculated with Equation 3.27 introduced in Section 3.5.1 and reported here again:

$$Nu = \frac{\frac{f}{8}\left(Re_b - 1000\right) Pr_b}{1 + \delta_v^+ \sqrt{\frac{f}{8}} \left(Pr_v^{2/3} - 1\right)} \tag{6.4}$$

This expression of the Nusselt number is dependent on the ratio of two different values of the Prandtl number, one calculated at bulk properties (Pr_b, at the numerator) and one calculated at the laminar sublayer properties (Pr_v, at the denominator). As also observed by Buongiorno,

there are at least two main reasons why generally Pr_v tends to be smaller than Pr_b: (i) in a heating configuration, the wall temperature will be naturally higher than the temperature of the bulk fluid. Viscosity and thermal conductivity are temperature-dependent and will be smaller in the laminar sublayer where the temperature is higher; (ii) viscosity and thermal conductivity of the nanofluid will be smaller in the proximity of the wall because of the lower nanoparticle concentration (assuming nanoparticles migrate away from the wall). Because of these effects, the ratio of Prandtl numbers will be generally higher than unity. This observation explains mathematically why Equation 6.4 predicts an enhancement when nanoparticles are dispersed in a fluid.

This conclusion is accepted here, although a sensitivity analysis of the effects of the variation of the volume concentration, ϕ, and of the parameter N_{BT}, on the calculation of the Nusselt number could provide more insights. It could be argued that there are conditions where the heat transfer enhancement in presence of nanoparticles might not be as large: (i) N_{BT} is larger than unity when Brownian diffusion is more effective than thermophoresis. In this case, the distribution within the laminar sublayer tends to be more homogeneous, and the difference between Pr_v and Pr_b is smaller; (ii) when the nanoparticles volume concentration is small, the impact of nanoparticles on the fluid thermophysical properties is also small. Therefore, Pr_v and Pr_b will be almost equal, even when N_{BT} is lower than unity.

In order to investigate these speculations, a convective heat transfer sensitivity analysis was conducted assuming a R410A based nanofluid in single phase. The results of this analysis are reported in Figure 6.27, where the Nusselt number (Figures (a), (c) and (e)) and the ratio between Pr_b and Pr_v (Figures (b), (d) and (f)) are represented as a function of the Reynolds number, for different Al_2O_3 nanoparticle volume concentrations and for different values of N_{BT}, increasing from top to bottom.

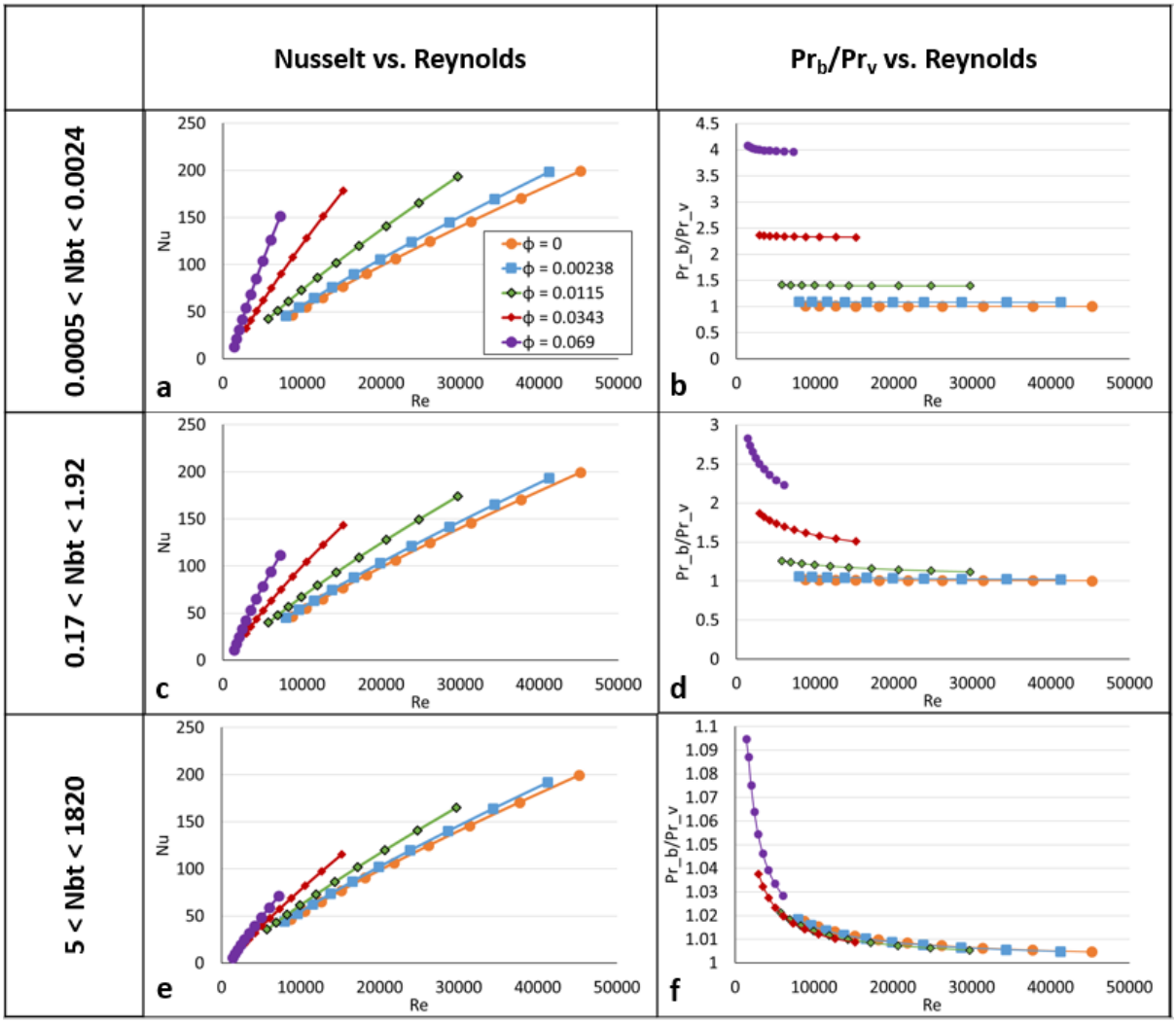


Figure 6.27: Single phase R410A based nanofluid: Nu vs. Re ((a), (c) and (e)), and Pr_b/Pr_v vs. Re ((b), (d) and (f)) for different nanoparticles concentrations and for increasing values of N_{BT}.

By looking at Figures 6.27, it is clear that a low nanoparticle volume concentration (marked by a blue line) does not affect the performances much, compared to the particle-free case (orange line), and this is true at all values of N_{BT}. A minimum concentration threshold for observable change in performances could be fixed at about 1% in volume fraction (green line). Similar conclusions could be drawn from the ratio of Prandtl numbers of Figure (b), (d) and (f), where the blue line (ϕ = 0.00238) is always on top of the orange line, representing the particle-free case. Because the concentration is low, thermophysical properties will not be very much affected, at all values of N_{BT}.

In Figure (a), (c) and (e), the Nusselt number tends to increase with increasing nanoparticle concentration, where the impact on thermophysical properties is higher. In particular, the increase is higher for lower values of N_{BT}, that is when the nanoparticles tend to move away from the wall leaving a less viscous fluid behind. At higher N_{BT} (Figure (e)) the nanoparticles will be more homogeneously distributed in the laminar sublayer and the concentration closer to the bulk concentration. In this situation, the nanofluid behaves like a pure fluid with higher viscosity and higher thermal conductivity and its heat transfer performances will be closer to a particle-free fluid. Because there is no migration of particles, the small increase in Nusselt number observable in Figure (c), is mainly the consequence of the temperature gradient within the laminar sublayer.

In Figures (b), (d) and (f) the ratio between Pr_b and Pr_v shows that the difference between the Prandtl numbers calculated at the bulk and at the laminar sublayer properties, gets smaller as N_{BT} increases, that is when the laminar sublayer properties are closer to the bulk fluid properties. At lower Reynolds number, the curve describing the ratio tended to increase, meaning that Pr_v started becoming smaller. It was observed that at lower Reynolds number, the value of N_{BT} was also smaller and it was speculated that the reason why the ratio starts growing is because the thickness of the laminar sublayer and the temperature gradient are larger at lower flow regimes.

For the way the model was described, this ratio between Pr_b and Pr_v will never be smaller than unity, and therefore no heat transfer degradation could be calculated. Possible ways a degradation could be obtained with this model are if the temperature gradient is inverted, causing the particles to move towards the wall, or if another mechanism started acting on particles, causing them to accumulate along the tube wall. A suggestion for future work in the study of nanofluids in a two-phase flow, is that the boundary conditions of the radial analysis could be improved. In particular, the boundary condition stating that the interface particle concentration is equal to the bulk concentration ($\phi_i = \phi_b$) might not be accurate for a two-phase flow. In fact, differently from the single phase case, when nanoparticles migrate from the wall, they can only travel as far away as the liquid-vapor interface. It is therefore possible to speculate that during the evaporation process, nanoparticles will increase the concentration at the liquid-vapor

interface beyond the starting value of ϕ_b. This phenomenon could impact on the liquid surface velocity, as well as on the bubbles formation and departure.

As already observed, in presence of oil and nanoparticles, the thermophysical properties calculated according to the correlations presented in Section 5.1 did not show a measurable difference with respect to other cases with higher nanoparticles concentrations, or with respect to the the nanoparticle-free case. This result was also confirmed by the heat transfer analysis for a oil-refrigerant mixture. For conciseness only one case is reported here, for a mixture of refrigerant R410A and oil at 3% OMF. Figure 6.28 shows the trends of Nusselt number for a particle-free case (orange line). The lines at higher volume concentrations (blue, green and red) show a small increase of the Nusselt number but the variation with concentration is insignificant. These results were observed also at 1% OMF and for higher and lower values of N_{BT}.

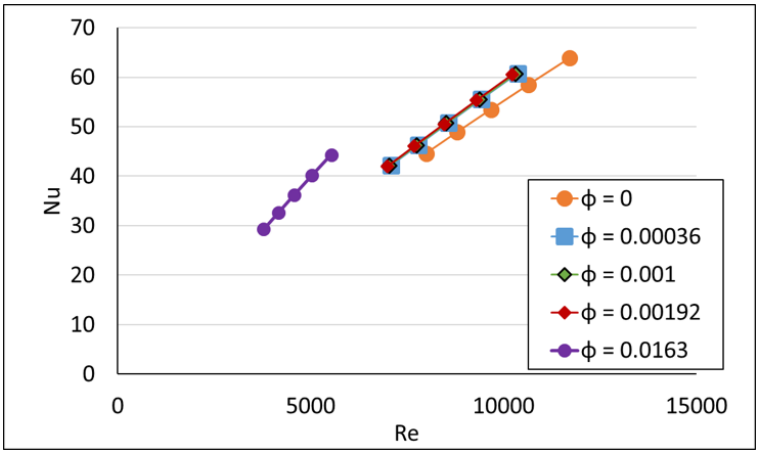


Figure 6.28: Single phase R410A and 3% oil: Nu vs. Re for a refrigerant-nanolubricant mixture at different nanoparticles concentrations.

Similarly to what done in Figure 6.15, in order to exercise the simulation, a case with high oil mass fraction (10%) and high nanoparticle mass fraction in oil (50%) was tested. This case is represented in Figure 6.28 by the purple line and it corresponds to a mixture volume concentration of about 0.0163. The previous observation that nanoparticles have a measurable effect on thermophysical properties when their bulk volume concentration (ϕ) is about 0.01 or higher seems to be confirmed also in this case.

The study of a heat transfer correlation describing nanolubricant-refrigerant mixture two-phase flow will be object of the next Section.

6.4 Effect of Microfins

The use of a friction factor describing the effect of microfins' roughness was introduced in Section 5.4.3. Depending on flow conditions and on geometrical parameters, microfins not only increase the effective heat transfer surface area, but they also promote the formation of fluid turbulence at earlier stages, when compared to a smooth tube. The effect of the use of a friction factor correlation for finned tubes on the prediction of the heat transfer coefficient is reported in Figure 6.29 for one test at 3% OMF, 20% NMF, 250 kg/m^2s mass flux, 12 kW/m^2 heat flux. It can be observed that when the finned surface friction factor is used (represented by green squares), the prediction of heat transfer coefficient increases with respect to the case of a smooth tube (represented by yellow triangles).

The heat transfer coefficient reported in Figure 6.29 is the single phase convective heat transfer predicted applying the correlation of Buongiorno (2006), calculated at different qualities with thermodynamic properties that were function of the increasing value of local oil concentration (ω_{local}).

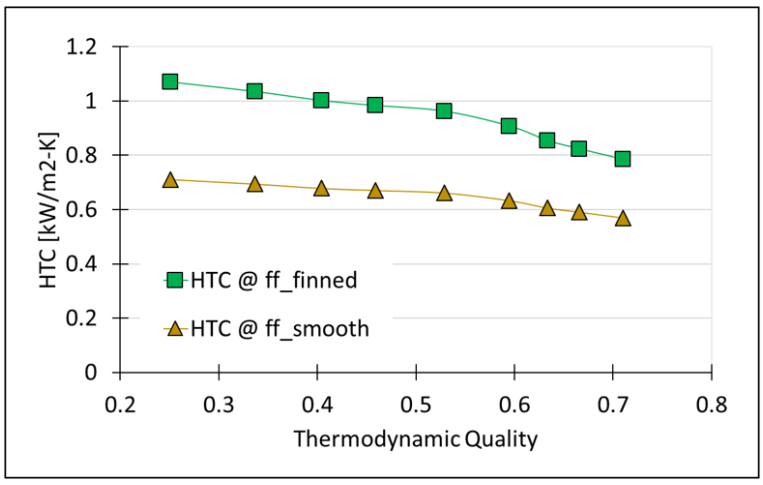


Figure 6.29: Effect of the use of a friction factor correlation for finned tubes on the prediction of the heat transfer coefficient.

It is not clear what effect the presence of microfins might have on the distribution of nanoparticles within the liquid film. As the refrigerant evaporates, the thickness of the liquid

decreases and it could be speculated that at some point the height of microfins could exceed the thickness of the liquid film. This effect should be expected at higher qualities, higher mass fluxes and lower values of OMF.

Figure 6.30 reports the trends of change of liquid film thickness at different qualities, and for different values of OMF. The mass flux was kept constant at 425 kg/m^2s because generally at higher mass fluxes the thickness is smaller. The predictions were obtained from the simulation model developed in this work. Figure 6.30 shows that for increasing quality the thickness decreases and becomes equal to the fin height around 0.7 quality or higher. This is observed for all values of OMF (0%, 1%, and 3%), although the thickness is slightly higher for higher values of OMF.

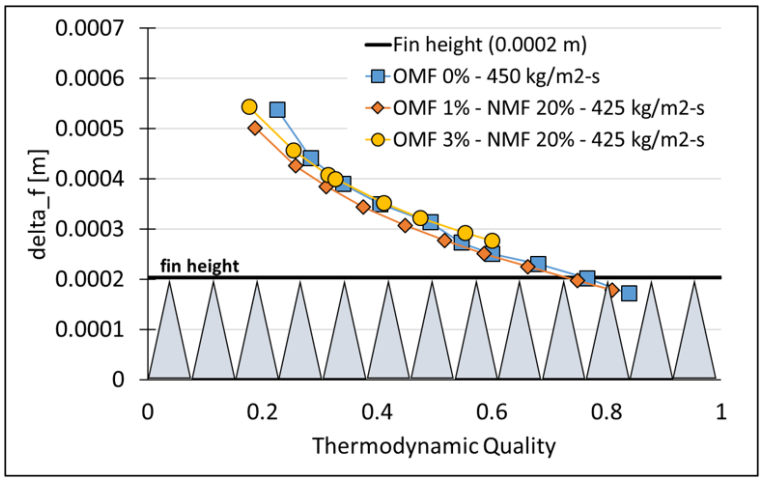


Figure 6.30: Trend of liquid film thickness for simulation tests at 425 kg/m^2s mass flux, at different values of OMF.

6.5 Correlation for Two-Phase Heat Transfer

The investigation conducted in this work offered many relevant insights about the behavior of nanoparticles when dispersed in fluid undergoing a heating process. However, in the specific study of nanoparticle laden lubricants, the chosen modeling approach seemed to reveal at least two limitations: (i) the chosen correlations implemented so far in the simulation tool to describe the thermophysical properties do not seem to be sensitive to different nanoparticle concentrations. This observations could also be justified by the fact that the actual nanoparticle concentration used in this work is very small when compared to the total liquid mass of oil

and refrigerant; (ii) the boundary conditions to the radial analysis were chosen in such a way that the ratio of Pr_b and Pr_v will never be smaller than one, and therefore there will be no possibility for degradation.

These limitations are to be considered here because the experimental data set for two-phase flow heat transfer and pressure drop discussed in Section 4.4.7, did include cases in which the heat transfer performances showed a degradation.

Also, with the current modeling approach, there still is a limitation on the enhancement that can be predicted, since as it is, the simulation starts to predict an enhancement only when the nanoparticle volume concentration is much higher than the one used during the experiments.

6.5.1 Approach for Correlation Development

In order to address the first limitation, a more representative estimation of the convective and nucleate boiling heat transfer coefficients was obtained by calculating the thermophysical properties as a function of the local oil concentration (ω_{local}) rather than the constant value of oil mass fraction (OMF). In fact, because the nanoparticles are dispersed in oil and liquid refrigerant, as the refrigerant evaporates, the nanoparticles' concentration (ϕ) increases. It is therefore reasonable to estimate the liquid thermophysical properties as a function of the local value of oil mass fraction (defined as in Equation 5.12 and reported here: $\frac{m_{oil}}{m_{oil}+m_{ref,L}}$). At higher qualities, the increase in local oil mass fraction determines an increase in ϕ such that the impact of nanoparticles dispersed in an oil-refrigerant mixture becomes more significant.

In the attempt to describe the experimental data using the model presented, correlations were developed using only the experimental data that showed neither an enhancement nor a degradation with respect to the pure refrigerant case. The parameters obtained in this way were then used to predict also the series that did show an enhancement or a degradation.

As a first approach to modeling the two-phase flow performances of refrigerant-nanolubricant mixtures, the experimental data set was subdivided in three sub-sets, based on the different oil mass fraction concentrations (i.e. 0 wt.%, 1 wt.%, and 3 wt.%). This was done in order to simplify the investigation, by reducing the number of variables accounted for in each

subset. The range of applicability of the correlations is: $180 < G_{flux} < 425\ kg/m^2s$, $12 < q" < 15\ kW/m^2$, and $0.2 < x < 0.8$.

Chen Modeling Approach

The procedure described by Chen (1966) and presented in Section 5.4.4 was applied here to the three sub-sets of experimental data. Therefore, three sets of two-phase multiplier and of suppression factor were obtained and they are listed here:

- $F_{0\%} = 1 + 3.809 \left(\frac{1}{X_{tt}}\right)^{-0.0671}$; $S_{0\%} = \dfrac{1}{1 + 10135.94 \cdot Re_{tp}^{0.268}}$
- $F_{1\%} = 1 + 3.085 \left(\frac{1}{X_{tt}}\right)^{0.0425}$; $S_{1\%} = \dfrac{1}{1 + 0.05 \cdot Re_{tp}^{1.289}}$
- $F_{3\%} = 1 + 4.417 \left(\frac{1}{X_{tt}}\right)^{0.0322}$; $S_{3\%} = \dfrac{1}{1 + 1.996 \cdot Re_{tp}^{2.14}}$

Figure 6.31 reports the predictions of the heat transfer coefficients obtained using the parameters F and S. Figure (a), (b) and (c) represent subsets at 0 wt.%, 1 wt.%, and 3 wt.% respectively. The points marked with red circles are the data that was used to develop F and S. The blue triangles mark the series where a degradation higher than 15% was observed. The green squares represent the series where an enhancement was observed.

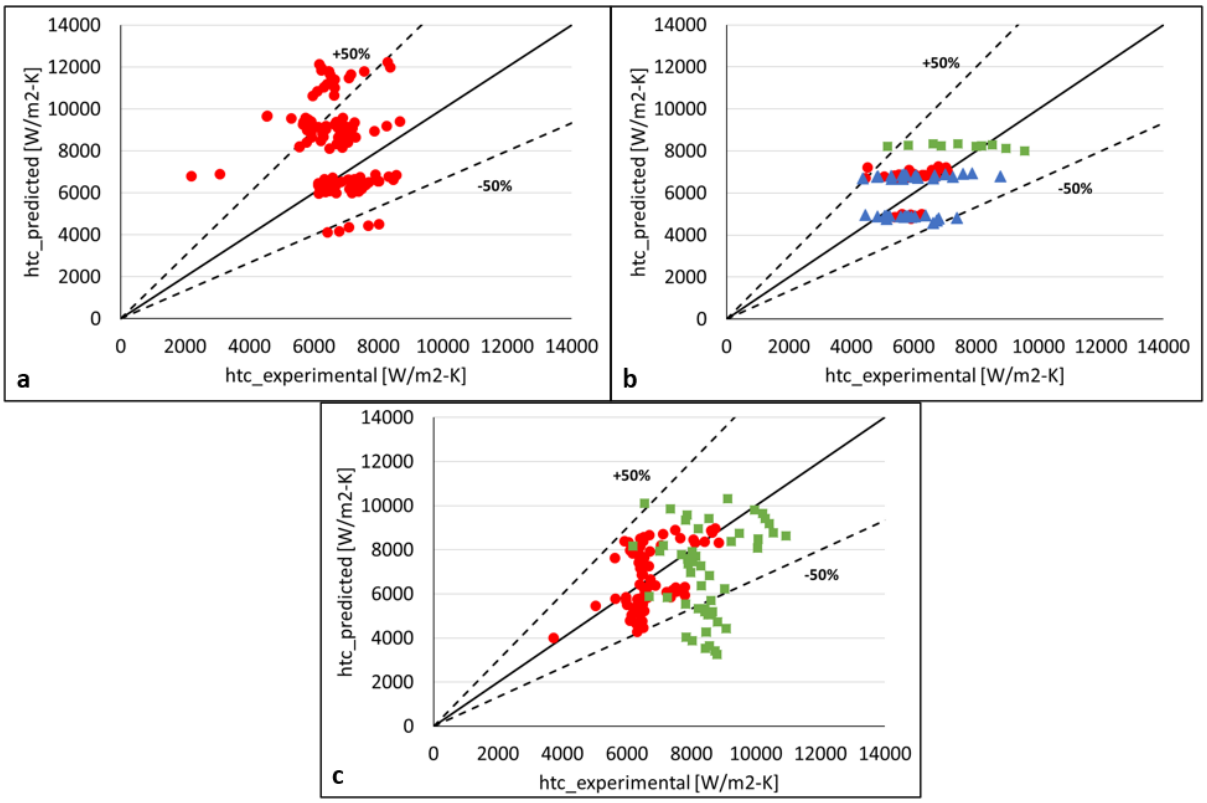


Figure 6.31: Correlation predictions for (a) OMF = 0, (b) OMF = 1%, and (3) OMF = 3 %, using Chen method.

The predictions are scattered with an uncertainty of about $\pm 50\%$ for all oil mass concentrations. While the blue triangles (representing a series were degradation was measured) fall within this uncertainty range (Figure (b)). Because the green squares represent enhancement and because they were not accounted for when developing the parameters F and S, it is reasonable to observe that the correlation underpredicts their values (Figure (c)). A lower uncertainty was expected for the series in red circles and it is not clear whether an error was made when following the procedure to develop S and F, or rather if there is an uncertainty in some of the experimental data used for this purpose.

Sawant Modeling Approach

The procedure described by Sawant (2012) and presented in Section 5.4.4 was also applied here because of the similarity with the geometry and type of fluid utilized by Sawant (2012) to develop their correlation. The same procedure was followed in this work to the three sub-sets of

experimental data. Three sets of two-phase multiplier and of suppression factor were obtained and they are listed here:

- $F_{0\%} = 13665 \left(\frac{1}{X_{tt} Re \cdot x}\right)^{1.334}$; $S_{0\%} = 2.334 \cdot e^{-1.57 \cdot 10^{-5} Re \cdot x}$
- $F_{1\%} = 1100035 \left(\frac{1}{X_{tt} Re \cdot x}\right)^{1.98}$; $S_{1\%} = 2.617 \cdot e^{5.81 \cdot 10^{-5} Re \cdot x}$
- $F_{3\%} = 5.9 \cdot 10^{19} \left(\frac{1}{X_{tt} Re \cdot x}\right)^{7.17}$; $S_{3\%} = 14.65 \cdot e^{2.57 \cdot 10^{-5} Re \cdot x}$

Figure 6.32 shows the correlation predictions for the three sub-sets using the estimated values of F and S.

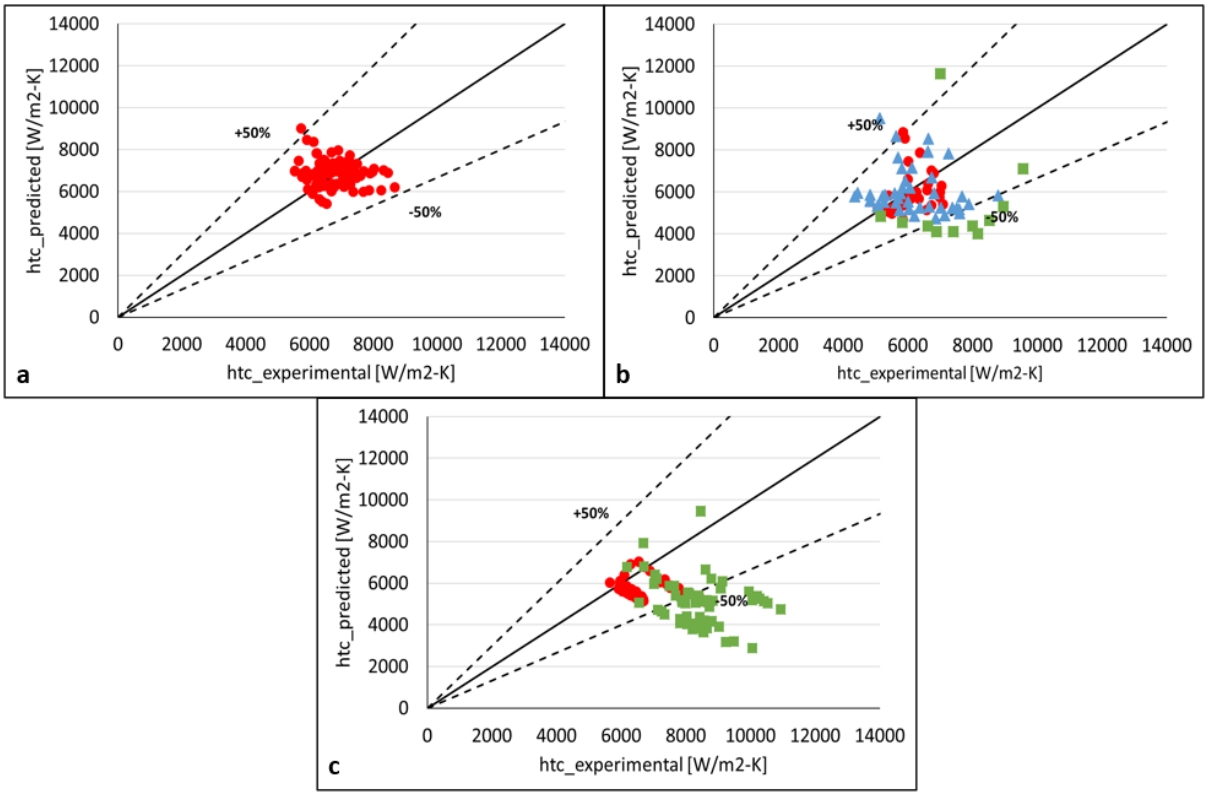


Figure 6.32: Correlation predictions for (a) OMF = 0, (b) OMF = 1%, and (3) OMF = 3 %, using Sawant method.

The general trends of prediction are similar to the ones found with Chen's method. Points are scattered with an uncertainty of about $\pm 50\%$ for all oil mass concentrations. The blue triangles at 1% wt. (representing a series with degradation) fall within the same $\pm 50\%$ uncertainty range (Figure (b)). The green squares representing enhancement are underpredicted (Figure (b) and (c)) and this is expected as discussed for Chen's method.

An investigation on Sawant's model, led to the observation that the predictions could be improved if the total heat transfer coefficient was calculated according to Equation 6.5:

$$h = C - S \cdot h_{nb} + F \cdot h_L \tag{6.5}$$

where C is a constant value equal to 10000 $\frac{W}{m^2 K}$ and the coefficients S and F are the same reported before. The results of this modification are reported in Figure 6.33

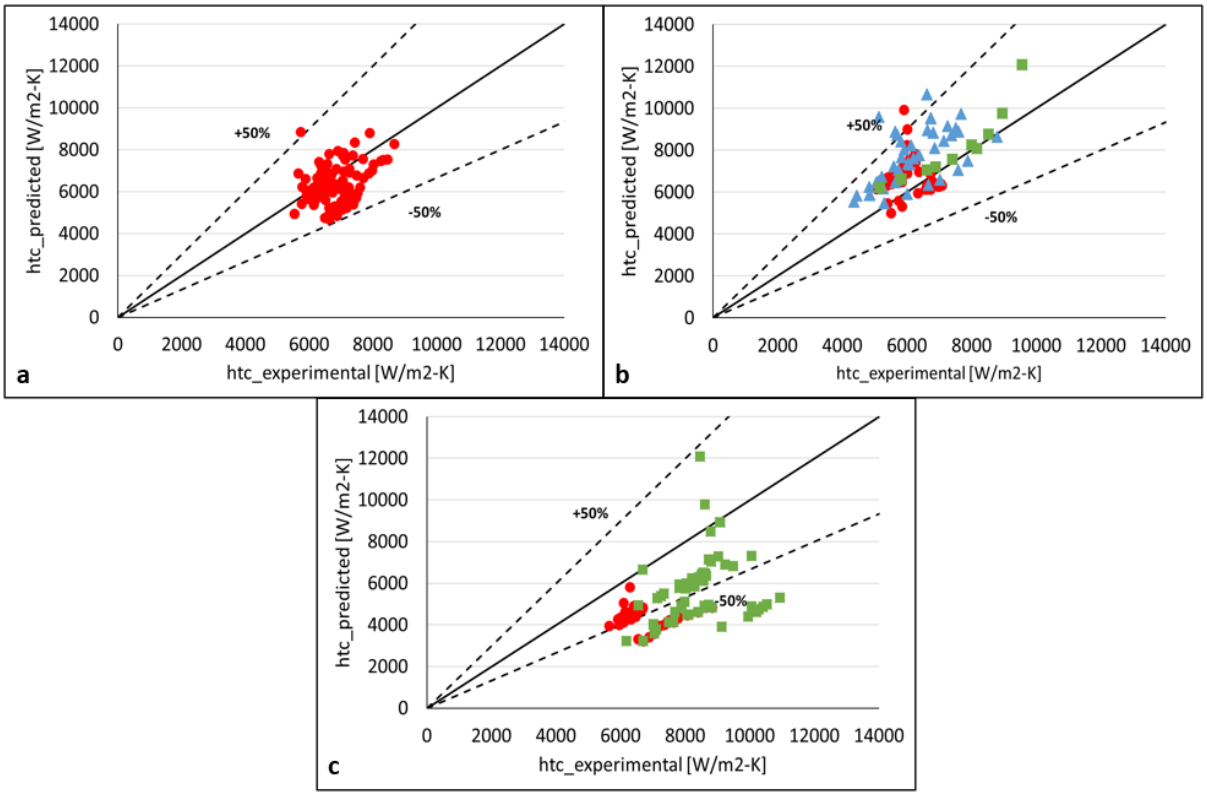


Figure 6.33: Correlation predictions for (a) OMF = 0, (b) OMF = 1%, and (3) OMF = 3 %, using a modified Sawant method.

This correction is arbitrary and is reported here only to suggest that the Sawant correlation can be improved to better predict the experimental data of this work. This observation is left as a suggestion and recommendation for future investigation.

Chapter 7

Conclusions and Recommendations

This field of research is still in its infancy and both the experimental and the modeling work conducted for this work provided useful insights into the understanding of the behavior of nanoparticle dispersion in highly viscous fluids. These insights can be used in future work to better assess the potentiality and convenience of the use of nanofluids.

The objective of this dissertation was to investigate, both experimentally and theoretically, the thermophysical properties and thermal performances of mixtures of refrigerant and nanolubricants, during two-phase flow boiling inside a microfin tube (with hydraulic diameter of 5.45 mm).

This study focused on Alumina nanoparticles (Aluminium oxide, Al_2O_3) with a 40÷60 nm nominal diameter and spherical shape. The nanoparticles were stabilized using a surfactant and they were dispersed at different mass concentrations (0%, 10%, and 20%) in an ISO VG 32 POE oil with density of 0.981 g/ml at 20°C and kinematic viscosity of 31.2 cSt and 5.6 cSt respectively at 40°C and 100°C. The base fluid was refrigerant R410A and the oil concentration ranged between 0% and 3%. The experimental operational conditions chosen for mass flux (180 kg/m^2s to 425 kg/m^2s), heat flux (12 kW/m^2), and saturation temperature (4°C) were similar to those of a real case scenario.

7.1 Conclusions of the Experimental Work

Experiments were conducted to measure the degree of potential sedimentation and agglomeration of the nanoparticles. This was sometimes cited as an operational challenge associated with

the storage and usage of nanolubricants in vapor compression systems. Tests were also conducted to measure the thermal conductivity, specific heat, and viscosity of the nanolubricants at various nanoparticles concentration and using different surfactants and dispersion methods. Solubility and miscibility of refrigerant R410A with two types of nanolubricants that shared the same nanoparticles but had different surfactants, were measured for temperature ranging from 0°C to 45°C.

The results showed that surfactants play a critical role in preventing agglomeration and sedimentation of the nanoparticles dispersed in the POE oil. Two out of three surfactants used in this work were successful to prevent agglomeration while one type of surfactant was ineffective and large clusters were observed few hours after ultrasonic mixing. The specific heats of the nanolubricants were lower than that of POE oil at temperature from 0°C to 20°C while they were similar at 40°C. Thermal conductivity ranged from 1.5 times higher at 5°C to 2 times higher at 40°C than that of POE lubricant. The viscosity was about 2.6 higher at 5°C while it was similar to that of POE lubricant at 40°C. The nanolubricants had also lower refrigerant R410A solubility with respect to POE oil and surfactants affected slightly the thermal conductivity, viscosity, and solubility properties of the nanolubricants.

By adding POE based nanolubricants to refrigerant R410A during two phase flow boiling at saturation temperature of 3.5°C $\pm$ 0.9°C, the effects on the heat transfer coefficient for an horizontal 9.5mm micro-fin tube were marked, and more important measurable, with respect to the effects on the two phase flow pressure drop. Al_2O_3 based nanolubricants provided an enhancement of the heat transfer coefficients with no or very small penalization of the two phase flow pressure drop. When charging nanoparticles in the POE oil at 20% mass concentration and with oil concentration of 3% in the two phase flow, the heat transfer coefficient increased by 15% and up to 40%. This result suggested that the spherical shaped Al_2O_3 nanoparticles created a preferential path for heat transfer exchange across the liquid phase of the mixture but did not add any additional resistance to the flow of refrigerant and lubricant mixture inside the micro-fin tube.

7.2 Conclusions of the Simulation Work

The simulation work provided a comparison between experimental results of two-phase flow boiling in a microfin tube of refrigerant R410A, R410A-lubricant mixture and R410A-Al_2O_3 nanolubricant mixture, and models available in the literature for estimating pressure drop and heat transfer coefficient. The comparison was made by using a newly developed simulation tool that included literature correlations for predicting thermophysical properties of lubricants, nanolubricants and refrigerant/lubricant and nanolubricant mixtures. For high nanoparticle concentrations and in some flow conditions, the refrigerant and nanolubricant mixture showed higher heat transfer coefficient than that of both the refrigerant R410A-POE oil mixture and the refrigerant R410A only. However, the enhancements were dependent on quality, mass flux, and heat flux, and in some case, the data showed a degradation of heat transfer coefficient. Similar findings were documented in the literature and it was reported that some lubricants could enhance the thermophysical properties of the refrigerant liquid phase during evaporation.

The findings of this work showed that, within their reported uncertainty, the correlations in the literature were generally able to predict the experimental data for the cases of refrigerant R410A and refrigerant-lubricant mixture of this work but they were inadequate to describe the behavior of the refrigerant-nanolubricant mixtures. An increase of thermal conductivity of the refrigerant R410A and nanolubricant liquid phase mixture of +25% increased the predicted heat transfer coefficient significantly, and in this case, the error between the simulation results and the data was within 15%. However, none of the existing correlations resulted in such increase of thermal conductivity of the liquid phase of the refrigerant R410A and nanolubricant mixture. It was speculated, as observed in available work on nanofluids research, that Al_2O_3 nanoparticles could induce a change in the nature of the mixture depending on the local dispersion concentration and promote a transition to non-Newtonian behavior. For this reason, different approaches to model these types of nanolubricants mixture were considered.

A more fundamental approach was followed to understand the magnitude of different mechanisms governing the particles behavior within the base fluid, and to assess what is the particles' specific contribution to the heat transfer phenomenon. Starting from the analysis

conducted by Buongiorno (2006) for a nanofluid single phase convective flow, different slip mechanisms were considered and the ones that were found to have larger impact were the particle Brownian diffusion and thermophoresis.

Based on a study of the Navier-Stokes equations of continuity, momentum, and energy balance, a radial analysis was implemented in the simulation tool, to observe the distribution of nanoparticles within the liquid film of a two-phase flow. The simulation was able to describe the impact of nanoparticle distribution on thermophysical properties and on the velocity profile.

It was concluded that, when the Brownian diffusion is predominant, the distribution of nanoparticles will be uniform and the concentration in the liquid film will be closer to the bulk concentration. In this situation, the nanofluid behaves like a pure fluid with higher viscosity and higher thermal conductivity and its heat transfer performances will be closer to a particle-free fluid. A small increase in Nusselt number is mainly the consequence of the temperature gradient within the laminar sublayer.

A sensitivity analysis of the effect of nanoparticle distribution on heat transfer showed that a low nanoparticle volume concentration does not affect the performances much, compared to the particle-free case. A minimum concentration threshold for observable change in performances could be fixed at about 1% in volume fraction.

When thermophoresis is the predominant diffusion mechanism, nanoparticles will move away from the wall, creating a low concentration sublayer. The formation of a particle-free low-viscosity layer near the tube wall could induce a lubricating effect that causes a drop in pressure gradient at the wall and a change in the velocity profile. This result was considered explanatory of the behavior observed experimentally, regarding the relatively small increase in pressure drop when testing nanofluids, compared to the pressure drop measured for the same nanoparticle-free fluid.

In the case of an evaporative two-phase flow, the increase in pressure drop found at higher qualities during the experimental campaign could be justified by the fact that the nanoparticles volume concentration in the oil-refrigerant mixture increases when the refrigerant evaporates, reducing the thickness of the liquid layer and therefore making the distribution of the particles more uniform.

Similar conclusions were expected in presence of a refrigerant-nanolubricant mixture. However, the thermophysical properties calculated according to the correlations presented in this work did not show a measurable difference in presence of nanoparticles. The reason for this behavior could be found in the very low nanoparticle concentration having a very small impact on thermophysical properties of the refrigerant-oil mixture.

The model described by Buongiorno (2006) was used in this work to estimate the single phase convective flow heat transfer of a heated liquid. This model relies on the calculation of a ratio between two different values of the Prandtl number, calculated with bulk properties, and with local properties of the laminar sublayer. The difference between the Prandtl numbers is smaller when the distribution of nanoparticle is uniform. However, for the way the model was described, the ratio between Pr_b and Pr_v will never be smaller than unity, and therefore no heat transfer degradation could be predicted.

Using the superposition model, the development of correlations to describe the evaporative two-phase flow of a refrigerant-nanolubricant mixture was attempted using the correlations developed by Buongiorno (2006) for convective heat transfer, and by Kedzierski (2012) for pool boiling heat transfer. The predictions of the experimental data were spread on a $\pm$ 50% uncertainty range and the cause of this low accuracy might be found in the limitations of the model to describe the effect of the presence of nanoparticles in the lubricant-refrigerant mixture.

7.3 Recommendations for Future Work

The direct observation of the nanoparticles' behavior is ultimately the best way to verify the hypothesis and assumptions of this and other works on nanofluids. Technology advancements are making it possible to design instrumentation able to perform direct measurements of nanoparticles' distribution. The use of such technology is therefore recommended in future works.

With regard to the present work, in order to verify the correctness of the simulation tool and the applicability of Buongiorno's model to a highly viscous fluid, it is suggested that tests are collected for a single phase convective flow of different refrigerant-nanolubricant mixtures. The range of investigation could be expanded with more tests on both a finned and a smooth

tube, and for different types of nanoparticles, preferably those with higher thermal conductivity to favor thermophoresis and make the measurement more certain.

The enhancement reported in some experiments on two-phase flow cannot be justified by this model and it was argued that other mechanisms might play a role on nanoparticle distribution and behavior. In particular, the effect of the presence of microfins on the tube walls needs to be investigated in future work, with particular attention to the ratio between fin height and liquid film thickness.

Regarding the use of surfactant, its effect could be better described both in the model and with direct experimental measurements.

Possible ways a degradation could be obtained with the model presented are if the temperature gradient is inverted, causing the particles to move towards the wall, or if another mechanism started acting on particles, causing them to accumulate along the tube wall. For this reason a set of condensation tests could provide interesting insights.

A suggestion for future improvement of the simulation of two-phase flow of nanofluids, is a change in the boundary conditions of the radial analysis. In particular, the boundary condition stating that the interface particle concentration is equal to the bulk concentration ($\phi_i = \phi_b$) might not be accurate for a two-phase flow. In fact when nanoparticles migrate from the wall, they can only travel as far away as the liquid-vapor interface. It is therefore possible to speculate that during the evaporation process, nanoparticles will increase the concentration at the liquid-vapor interface beyond the starting value of ϕ_b. This phenomenon could impact on the liquid surface velocity, as well as on the mechanisms of bubbles formation and departure.

References

Ahuja, A. S. (1975a). Augmentation of heat transport in laminar flow of polystyrene suspensions. I. Experiments and results. *Journal of Applied Physics*, 46(8):3408–3416.

Ahuja, A. S. (1975b). Augmentation of heat transport in laminar flow of polystyrene suspensions. II. Analysis of the data. *Journal of Applied Physics*, 46(8):3417–3425.

Aladag, B., Halelfadl, S., Doner, N., Mar, T., Duret, S., and Estell, P. (2012). Experimental investigations of the viscosity of nanofluids at low temperatures. *Applied Energy*, 97:876–880.

Anderson, J. L. and Prieve, D. C. (1984). Diffusiophoresis: Migration of Colloidal Particles in Gradients of Solute Concentration. *Separation and Purification Methods*, 13(1):67–103.

Angayarkanni, S. A. and Philip, J. (2015). Review on thermal properties of nanofluids: Recent developments. *Advances in Colloid and Interface Science*, 225:146–176.

Asali, J. C., Hanratty, T. J., and Andreussi, P. (1985). Interfacial drag and film height for vertical annular flow. *AIChE Journal*, 31(6):895–902.

ASHRAE (2010). ASHRAE Handbook - Refrigeration. Technical report, American Society of Heating, Refrigerating and Air Conditioning Engineers, Inc., Atlanta.

Bandarra Filho, E. P., Cheng, L., and Thome, J. R. (2009). Flow boiling characteristics and flow pattern visualization of refrigerant/lubricant oil mixtures. *International Journal of Refrigeration*, 32(2):185–202.

Bang, I. C. and Heung Chang, S. (2004). Boiling heat transfer performance and phenomena of Al2o3water nanofluids from a plain surface in a pool. *International Journal of Heat and Mass Transfer*, 48(12):2407–2419.

Baqeri, S., Akhavan-Behabadi, M. A., and Ghadimi, B. (2014). Experimental investigation of the forced convective boiling heat transfer of R-600a/oil/nanoparticle. *International Communications in Heat and Mass Transfer*, (55):71–76.

Bartelt, K., Park, Y., Liu, L., and Jacobi, A. (2008). Flow boiling of R-134a/POE/CuO nanofluids in a horizontal tube. West Lafayette, IN (USA). Purdue University. Paper 2278.

Batchelor, G. (1977). The effect of Brownian motion on the bulk stress in a suspension of spherical particles. *Journal of Fluid Mechanics*, 83(01):97–117.

Bell, I. H., Wronski, J., Quoilin, S., and Lemort, V. (2014). Pure and Pseudo-pure Fluid Thermophysical Property Evaluation and the Open-Source Thermophysical Property Library CoolProp. *Industrial & Engineering Chemistry Research*, 53(6):2498–2508.

Bergles, A. E., Collier, J. G., Delhaye, J. M., Hewitt, G. F., and Mayinger, F. (1981). *Two-phase flow and heat transfer in the power and process industries*. Hemisphere New York, New York.

Bergman, T. L., Incropera, F. P., DeWitt, D. P., and Lavine, A. S. (2011). *Fundamentals of Heat and Mass Transfer*. John Wiley & Sons, New York, 7th edition. Google-Books-ID: vvyIoXEywMoC.

Bird, R. B., Stewart, W. E., and Lightfoot, E. N. (2002). *Transport Phenomena*. John Wiley & Sons, 2nd edition.

Bobbo, S., Fedele, L., Fabrizio, M., Barison, S., Battiston, S., and Pagura, C. (2010). Influence of nanoparticles dispersion in POE oils on lubricity and R134a solubility. *International Journal of Refrigeration*, 33(6):1180–1186.

Brenner, H. and Bielenberg, J. R. (2005). A continuum approach to phoretic motions: Thermophoresis. *Physica A: Statistical Mechanics and its Applications*, 355(2):251–273.

Briefing, U. S. (2013). International Energy Outlook 2013. *US Energy Information Administration*.

Buongiorno, J. (2006). Convective Transport in Nanofluids. *Journal of Heat Transfer*, 128(3):240–250.

Buongiorno, J., Venerus, D. C., Prabhat, N., McKrell, T., Townsend, J., Christianson, R., Tolmachev, Y. V., Keblinski, P., Hu, L.-w., Alvarado, J. L., Bang, I. C., Bishnoi, S. W., Bonetti, M., Botz, F., Cecere, A., Chang, Y., Chen, G., Chen, H., Chung, S. J., Chyu, M. K., Das, S. K., Paola, R. D., Ding, Y., Dubois, F., Dzido, G., Eapen, J., Escher, W., Funfschilling, D., Galand, Q., Gao, J., Gharagozloo, P. E., Goodson, K. E., Gutierrez, J. G., Hong, H., Horton, M., Hwang, K. S., Iorio, C. S., Jang, S. P., Jarzebski, A. B., Jiang, Y., Jin, L., Kabelac, S., Kamath, A., Kedzierski, M. A., Kieng, L. G., Kim, C., Kim, J.-H., Kim, S., Lee, S. H., Leong, K. C., Manna, I., Michel, B., Ni, R., Patel, H. E., Philip, J., Poulikakos, D., Reynaud, C., Savino, R., Singh, P. K., Song, P., Sundararajan, T., Timofeeva, E., Tritcak, T., Turanov, A. N., Vaerenbergh, S. V., Wen, D., Witharana, S., Yang, C., Yeh, W.-H., Zhao, X.-Z., and Zhou, S.-Q. (2009). A benchmark study on the thermal conductivity of nanofluids. *Journal of Applied Physics*, 106(9):094312.

Carey, V. P. (1992). Liquid-vapor phase-change phenomena.

Chapman, S. and Cowling, T. G. (1970). *The mathematical theory of non-uniform gases: an account of the kinetic theory of viscosity, thermal conduction and diffusion in gases*. Cambridge university press, 3rd edition.

Charles, G. E. and Mason, S. G. (1960). The coalescence of liquid drops with flat liquid/liquid interfaces. *Journal of Colloid Science*, 15(3):236–267.

Chen, J. C. (1966). Correlation for boiling heat transfer to saturated fluids in convective flow. *Industrial & engineering chemistry process design and development*, 5(3):322–329.

Chen, W. and Fang, X. (2014). A note on the Chen correlation of saturated flow boiling heat transfer. *International Journal of Refrigeration*, 48:100–104.

Cheng, L., Bandarra Filho, E. P., and Thome, J. R. (2008). Nanofluid Two-Phase Flow and Thermal Physics: A New Research Frontier of Nanotechnology and Its Challenges. *Journal of Nanoscience and Nanotechnology*, 8(7):3315–3332.

Choi, J. Y., Kedzierski, M. A., and Domanski, P. A. (1999). *A generalized pressure drop correlation for evaporation and condensation of alternative refrigerants in smooth and micro-fin tubes*. National Institute of Standards and Technology Internal Report 6333.

Choi, S. U. (2009). Nanofluids: from vision to reality through research. *Journal of Heat Transfer*, 131(3):033106.

Chongrungreong, S. and Sauer, H. J., J. (1980). Nucleate Boiling Performance of Refrigerants and Refrigerant-Oil Mixtures. *Journal of Heat Transfer*, 102(4):701–705.

Cieliski, J. and Targaski, W. (2007). Horizontal flow boiling of R22, R134a and their mixtures with oil in smooth and enhanced tubes. *Archives of Thermodynamics*, 28:19–40.

Clary, D. R. and Mills, G. (2011). Preparation and Thermal Properties of CuO Particles. *The Journal of Physical Chemistry C*, 115(5):1767–1775.

Conti, J., Holtberg, P., Diefenderfer, J., LaRose, A., Turnure, J. T., and Westfall, L. (2016). International Energy Outlook 2016 With Projections to 2040. Technical report, USDOE Energy Information Administration (EIA), Washington, DC (United States). Office of Energy Analysis.

Corcione, M., Cianfrini, M., and Quintino, A. (2015). Enhanced natural convection heat transfer of nanofluids in enclosures with two adjacent walls heated and the two opposite walls cooled. *International Journal of Heat and Mass Transfer*, 88:902–913.

Cremaschi, L. (2012). A Fundamental View of the Flow Boiling Heat Transfer Characteristics of Nano-Refrigerants. In *Proceedings of the ASME 2012 International Mechanical Engineering Congress and Exposition*, pages 2779–2792, Houston, TX.

Cremaschi, L., Hwang, Y., and Radermacher, R. (2005a). Experimental investigation of oil retention in air conditioning systems. *International Journal of Refrigeration*, 28(7):1018–1028.

Cremaschi, L., Hwang, Y., and Radermacher, R. (2005b). Experimental investigation of oil retention in air conditioning systems. *International Journal of Refrigeration*, 28(7):1018–1028.

Das, S. K., Putra, N., and Roetzel, W. (2003). Pool boiling characteristics of nano-fluids. *International Journal of Heat and Mass Transfer*, 46(5):851–862.

Deokar, P., Cremaschi, L., Wong, T., and Criscuolo, G. (2016). Effect of Nanoparticles Aspect Ratio on the Two Phase Flow Boiling Heat Transfer Coefficient and Pressure Drop of Refrigerant and Nanolubricants Mixtures in a 9.5 mm Micro-fin Tube. page Paper No. 2098, West Lafayette, IN (USA). Purdue University.

Filippov, L. P. and Novoselova, N. S. (1955). Thermal conductivity of normal liquid solutions. *Vest. Mask. Gos. Univ., Ser. Fiz.*, (3):37–40.

Filonenko, G. (1954). HYDRAULIC RESISTANCE IN PIPES. *Teploergetica*, 1(4):40–44.

Furukawa, T. and Fukano, T. (2001). Effects of liquid viscosity on flow patterns in vertical upward gasliquid two-phase flow. *International Journal of Multiphase Flow*, 27(6):1109–1126.

Ghadimi, A., Saidur, R., and Metselaar, H. S. C. (2011). A review of nanofluid stability properties and characterization in stationary conditions. *International Journal of Heat and Mass Transfer*, 54(17):4051–4068.

Ghaednia, H., Hossain, M. S., and Jackson, R. L. (2016). Tribological Performance of Silver NanoparticleEnhanced Polyethylene Glycol Lubricants. *Tribology Transactions*, 59(4):585–592.

Godson, L., Raja, B., Mohan Lal, D., and Wongwises, S. (2010). Enhancement of heat transfer using nanofluidsAn overview. *Renewable and Sustainable Energy Reviews*, 14(2):629–641.

Goldsmith, H. L., Mason, S. G., and Eirich, F. (1967). *Rheology: Theory and Applications*, volume 4. Academic Press Inc., New York.

Gungor, K. E. and Winterton, R. H. S. (1986). A general correlation for flow boiling in tubes and annuli. *International Journal of Heat and Mass Transfer*, 29(3):351–358.

Hamilton, L. J., Kedzierski, M. A., and Kaul, M. P. (2008). Horizontal Convective Boiling of Pure and Mixed Refrigerants within a Micro-Fin Tube. *Journal of Enhanced Heat Transfer*, 15(3):211–226.

He, C. and Ahmadi, G. (1998). Particle Deposition with Thermophoresis in Laminar and Turbulent Duct Flows. *Aerosol Science and Technology*, 29(6):525–546.

Hetsroni, G., Zakin, J. L., Lin, Z., Mosyak, A., Pancallo, E. A., and Rozenblit, R. (2001). The effect of surfactants on bubble growth, wall thermal patterns and heat transfer in pool boiling. *International Journal of Heat and Mass Transfer*, 44(2):485–497.

Hewitt, G. and Hall-Taylor, N. S. (1970). *Annular two-phase flow*. Pergamon Press Inc., 1st edition.

Hori, K., Nakazatomi, M., Nishikawa, K., and Sekoguchi, K. (1979). On Ripple of Annular Two-Phase Flow : 3.Effect of Liquid Viscosity on Characteristics of Wave and Interfacial Friction Factor. *Bulletin of JSME*, 22(169):952–959.

Hu, H., Ding, G., and Wang, K. (2008a). Heat transfer characteristics of R410aoil mixture flow boiling inside a 7mm straight microfin tube. *international journal of refrigeration*, 31(6):1081–1093.

Hu, H.-t., Ding, G.-l., and Wang, K.-j. (2008b). Measurement and correlation of frictional two-phase pressure drop of R410a/POE oil mixture flow boiling in a 7mm straight micro-fin tube. *Applied Thermal Engineering*, 28(1112):1272–1283.

Hwang, K. S., Jang, S. P., and Choi, S. U. S. (2009). Flow and convective heat transfer characteristics of water-based Al2o3 nanofluids in fully developed laminar flow regime. *International Journal of Heat and Mass Transfer*, 52(1):193–199.

Jain, S., Patel, H. E., and Das, S. K. (2009). Brownian dynamic simulation for the prediction of effective thermal conductivity of nanofluid. *Journal of Nanoparticle Research*, 11(4):767–773.

Jensen, M. and Jackman, D. (1984). Prediction of nucleate pool boiling heat transfer coefficients of refrigerant-oil mixtures. *Journal of heat transfer*, 106(1):184–190.

Jensen, M. K. and Vlakancic, A. (1999). Technical Note Experimental investigation of turbulent heat transfer and fluid flow in internally finned tubes. *International Journal of Heat and Mass Transfer*, 42(7):1343–1351.

Kakac, S. and Pramuanjaroenkij, A. (2009). Review of convective heat transfer enhancement with nanofluids. *International Journal of Heat and Mass Transfer*, 52(1314):3187–3196.

Kedzierski, M. A. (2002). Effect of bulk lubricant concentration on the excess surface density during R123 pool boiling. *International Journal of Refrigeration*, 25(8):1062–1071.

Kedzierski, M. A. (2003). A semi-theoretical model for predicting refrigerant/lubricant mixture pool boiling heat transfer. *International Journal of Refrigeration*, 26(3):337–348.

Kedzierski, M. A. (2009). Effect of CuO Nanoparticle Concentration on R134a/Lubricant Pool-Boiling Heat Transfer. *Journal of Heat Transfer*, 131(4):043205–043205.

Kedzierski, M. A. (2011). Effect of Al2o3 nanolubricant on R134a pool boiling heat transfer. *International Journal of Refrigeration*, 34(2):498–508.

Kedzierski, M. A. (2012). R134a/Al2o3 nanolubricant mixture pool boiling on a rectangular finned surface. *Journal of Heat Transfer*, 134(12):121501.

Kedzierski, M. A. (2013). Viscosity and density of aluminum oxide nanolubricant. *International Journal of Refrigeration*, 36(4):1333–1340.

Kedzierski, M. A. (2018). Private Communications. In *NIST*, Gaithersburg, MD.

Kedzierski, M. A., Brignoli, R., Quine, K. T., and Brown, J. S. (2017). Viscosity, density, and thermal conductivity of aluminum oxide and zinc oxide nanolubricants. *International Journal of Refrigeration*, 74:3–11.

Kedzierski, M. A. and Gong, M. (2009). Effect of CuO nanolubricant on R134a pool boiling heat transfer. *International Journal of Refrigeration*, 32(5):791–799.

Kedzierski, M. A. and Kaul, M. P. (1998). Horizontal Nucleate Flow Boiling Heat Transfer Coefficient Measurements and Visual Observations for R12, R134a and R134a/Ester Lubricant Mixtures. *International Journal of Fluid Mechanics Research*, 25(1-3):386–399.

Kutateladze, S. S. (1961). Boiling heat transfer. *International Journal of Heat and Mass Transfer*, 4:31–45.

Lin, L., Peng, H., and Ding, G. (2015). Dispersion stability of multi-walled carbon nanotubes in refrigerant with addition of surfactant. *Applied Thermal Engineering*, 91:163–171.

Lin, L., Peng, H., and Ding, G. (2016). Experimental research on particle aggregation behavior in nanorefrigerantoil mixture. *Applied Thermal Engineering*, 98:944–953.

Lister, D. H. (1980). *Corrosion products in power generating systems*. Chalk River Nuclear Laboratories.

Lotfi, R., Saboohi, Y., and Rashidi, A. M. (2010). Numerical study of forced convective heat transfer of Nanofluids: Comparison of different approaches. *International Communications in Heat and Mass Transfer*, 37(1):74–78.

Lottin, O., Guillemet, P., and Lebreton, J.-M. (2003). Effects of synthetic oil in a compression refrigeration system using R410a. Part I: modelling of the whole system and analysis of its response to an increase in the amount of circulating oil. *International Journal of Refrigeration*, 26(7):772–782.

Mahdavi, M., Sharifpur, M., and Meyer, J. P. (2017). A novel combined model of discrete and mixture phases for nanoparticles in convective turbulent flow. *Physics of Fluids*, 29(8):082005.

Marston, J. O., Yong, W., and Thoroddsen, S. T. (2010). Direct verification of the lubrication force on a sphere travelling through a viscous film upon approach to a solid wall. *Journal of Fluid Mechanics*, 655:515–526.

Matsuo, T. and Itoh, M. (1998). A New Synthetic Hybrid Refrigeration Oil for R410a and R407c. volume 1, page Paper No. 441, West Lafayette, IL. Purdue University.

Maxwell, J. C. (1881). *A treatise on electricity and magnetism*, volume 1. Clarendon press, Oxford.

McNab, G. S. and Meisen, A. (1973). Thermophoresis in liquids. *Journal of Colloid and Interface Science*, 44(2):339–346.

Meyer, J. P. and Olivier, J. A. (2011). Transitional flow inside enhanced tubes for fully developed and developing flow with different types of inlet disturbances: Part I Adiabatic pressure drops. *International Journal of Heat and Mass Transfer*, 54(7):1587–1597.

Michaelides, E. E. (2015). Brownian movement and thermophoresis of nanoparticles in liquids. *International Journal of Heat and Mass Transfer*, 81:179–187.

Morrell, R. (1987). *Handbook of Properties of Technical & Engineering Ceramics: Part 2: Data Reviews: Section 1: High-Alumina Ceramics: Part 2*. Stationery Office Books, London.

Murshed, S. M. S. (2011). Determination of effective specific heat of nanofluids. *Journal of Experimental Nanoscience*, 6(5):539–546.

Murshed, S. M. S., Leong, K. C., and Yang, C. (2008). Investigations of thermal conductivity and viscosity of nanofluids. *International Journal of Thermal Sciences*, 47(5):560–568.

Murshed, S. M. S., Nieto de Castro, C. A., Lourenco, M. J. V., Lopes, M. L. M., and Santos, F. J. V. (2011). A review of boiling and convective heat transfer with nanofluids. *Renewable and Sustainable Energy Reviews*, 15(5):2342–2354.

Nidegger, E., Thome, J. R., and Favrat, D. (1997). Flow boiling and pressure drop measurements for R-134a/oil mixtures. Part 1: Evaporation in a microfin tube. *HVAC and R Research*, 3(1):38–53.

Noorani, A., Sardina, G., Brandt, L., and Schlatter, P. (2016). Particle transport in turbulent curved pipe flow. *Journal of Fluid Mechanics*, 793:248–279.

Oezerinc, S., Kakac, S., and Yazicioglu, A. G. (2010). Enhanced thermal conductivity of nanofluids: a state-of-the-art review. *Microfluidics and Nanofluidics*, 8(2):145–170.

Orian, G., Jelinek, M., and Levy, A. (2010). Flow boiling of binary solution in horizontal tube. *Energy*, 35(1):35–44.

Pak, B. C. and Cho, Y. I. (1998). Hydrodynamic and heat transfer study of dispersed fluids with submicron metallic oxide particles. *Experimental Heat Transfer an International Journal*, 11(2):151–170.

Peng, H., Ding, G., and Hu, H. (2011). Effect of surfactant additives on nucleate pool boiling heat transfer of refrigerant-based nanofluid. *Experimental Thermal and Fluid Science*, 35(6):960–970.

Peng, H., Ding, G., Hu, H., and Jiang, W. (2010). Influence of carbon nanotubes on nucleate pool boiling heat transfer characteristics of refrigerantoil mixture. *International Journal of Thermal Sciences*, 49(12):2428–2438.

Phillips, R. J., Armstrong, R. C., Brown, R. A., Graham, A. L., and Abbott, J. R. (1992). A constitutive equation for concentrated suspensions that accounts for shearinduced particle migration. *Physics of Fluids A: Fluid Dynamics (1989-1993)*, 4(1):30–40.

Pope, S. B. (2000). *Turbulent flows*. Cambridge University Press, Cambridge.

Puliti, G., Paolucci, S., and Sen, M. (2011). Nanofluids and their properties. *Applied Mechanics Reviews*, 64(3):030803.

Quintino, A., Ricci, E., Habib, E., and Corcione, M. (2017). Natural convection from a pair of differentially-heated horizontal cylinders aligned side by side in a nanofluid-filled square enclosure. *Energy Procedia*, 126:26–33.

Radermacher, R., Cremaschi, L., and Schwentker, R. A. (2006). Modeling of Oil Retention in the Suction Line and Evaporator of Air-Conditioning Systems. *HVAC&R Research*, 12(1):35–56.

Rouhani, S. Z. and Axelsson, E. (1970). Calculation of void volume fraction in the subcooled and quality boiling regions. *International Journal of Heat and Mass Transfer*, 13(2):383–393.

Rouhiainen, P. O. and Stachiewicz, J. W. (1970). On the deposition of small particles from turbulent streams. *Journal of Heat Transfer*, 92(1):169–177.

Saffman, P. G. (1965). The lift on a small sphere in a slow shear flow. *Journal of Fluid Mechanics*, 22(2):385–400.

Sarkar, J. (2011). A critical review on convective heat transfer correlations of nanofluids. *Renewable and Sustainable Energy Reviews*, 15(6):3271–3277.

Sarkas, H. (2014). Private Communications. In *Nanophase Technologies Corporation*, Romeoville, IL.

Savithiri, S., Pattamatta, A., and Das, S. K. (2011). Scaling analysis for the investigation of slip mechanisms in nanofluids. *Nanoscale Research Letters*, 6(1):471.

Sawant, N. N. (2012). *Influence of Lubricant on Horizontal Convective Boiling in a Micro-fin Tube*. PhD thesis, The Catholic University of America, Washington D. C.

Sawant, N. N., Kedzierski, M. A., and Brown, J. S. (2007). *Effect of Lubricant on R410A Horizontal Flow Boiling*. National Institute of Standards and Technology Internal Report 7456.

Segr, G. and Silberberg, A. (1962). Behaviour of macroscopic rigid spheres in Poiseuille flow Part 2. Experimental results and interpretation. *Journal of Fluid Mechanics*, 14(1):136–157.

Shah, M. M. (1976). A new correlation for heat transfer during boiling flow through pipes. *Ashrae Trans.*, 82(2):66–86.

Smith, J. R. and Cremaschi, L. (2014). Two Phase Flow Boiling Heat Transfer and Pressure Drop of Two New LGWP Developmental Refrigerants Alternative to R-410a. In *International Refrigeration and Air Conditioning Conference*, volume 1, page Paper No. 2332, West Lafayette, IN (USA). Purdue University.

Soltani, S., Etemad, S. G., and Thibault, J. (2010). Pool boiling heat transfer of non-Newtonian nanofluids. *International Communications in Heat and Mass Transfer*, 37(1):29–33.

Stephan, K. (1964). Influence of oil on heat transfer of boiling refrigerant 12 and refrigerant 22. In *Proceedings of the 11th International Congress of Refrigeration*, volume 1, pages 369–380.

Taylor, R., Coulombe, S., Otanicar, T., Phelan, P., Gunawan, A., Lv, W., Rosengarten, G., Prasher, R., and Tyagi, H. (2013). Small particles, big impacts: A review of the diverse applications of nanofluids. *Journal of Applied Physics*, 113(1):011301.

Thome, J. R. (1995). Comprehensive thermodynamic approach to modeling refrigerant-lubricating oil mixtures. *HVAC&R Research*, 1(2):110–125.

Thome, J. R. (2004). Engineering data book III. *Wolverine Tube Inc*, 2010.

Touloukian, Y. S. (1970). *Specific heat: nonmetallic solids*. Springer, New York, 1 edition edition.

Vajjha, R. S. and Das, D. K. (2009). Specific Heat Measurement of Three Nanofluids and Development of New Correlations. *Journal of Heat Transfer*, 131(7):071601–071601.

Venerus, D., Buongiorno, J., Christianson, R., Townsend, J., Bang, I. C., Chen, G., Chung, S. J., Chyu, M., Chen, H., and Ding, Y. (2010). Viscosity measurements on colloidal dispersions (nanofluids) for heat transfer applications. *Applied rheology*, 20(4):1–7.

Venerus, D. C. and Jiang, Y. (2011). Investigation of thermal transport in colloidal silica dispersions (nanofluids). *Journal of Nanoparticle research*, 13(7):3075–3083.

Wang, H. S. and Rose, J. W. (2004). Prediction of effective friction factors for single-phase flow in horizontal microfin tubes. *International Journal of Refrigeration*, 27(8):904–913.

Weast, R. C. (1984). *Handbook of Chemistry and Physics*. CRC Press, 65th edition.

Webb, R. L. and Gupte, N. S. (1992). A Critical Review of Correlations for Convective Vaporization in Tubes and Tube Banks. *Heat Transfer Engineering*, 13(3):58–81.

Wen, D. and Ding, Y. (2004). Experimental investigation into convective heat transfer of nanofluids at the entrance region under laminar flow conditions. *International Journal of Heat and Mass Transfer*, 47(24):5181–5188.

Wen, D. and Ding, Y. (2005). Effect of particle migration on heat transfer in suspensions of nanoparticles flowing through minichannels. *Microfluidics and Nanofluidics*, 1(2):183–189.

Wen, D., Zhang, L., and He, Y. (2009). Flow and migration of nanoparticle in a single channel. *Heat and Mass Transfer*, 45(8):1061–1067.

Yokozeki, A. (1994). Solubility and viscosity of refrigerant-oil mixtures. pages 335–340, West Lafayette, IN (USA). Purdue University.

Yoshinaga, Y., Peng, H., Dang, C., and Hihara, E. (2014). Experimental Study on Liquid Film Thickness of Annular Flow in Microchannels. volume 1, page Paper 2594, West Lafayette, IN (USA). Purdue University.

Youbi-Idrissi, M., Bonjour, J., Marvillet, C., and Meunier, F. (2003). Impact of refrigerantoil solubility on an evaporator performances working with R-407c. *International Journal of Refrigeration*, 26(3):284–292.

Zhao, Y., Molki, M., Ohadi, M. M., Franca, F. H., and Radermacher, R. (2002). Flow boiling of CO2 with miscible oil in microchannels/Discussion. *ASHRAE transactions*, 108:135.

Zuercher, O., Thome, J. R., and Favrat, D. (1998). In-Tube Flow Boiling of R-407c and R-407c/Oil Mixtures Part I: Microfin Tube. *HVAC&R Research*, 4(4):347–372.

Appendices

Appendix A

Brief Review on Correlations for Convective Vaporization

For practical purposes, the literature presents correlations developed to describe the phenomenology of fluids flowing in enclosed systems, undergoing a change of behavior due to the effect of energy addition or removal. The first available studies focused on single phase convective heat transfer and famous is the correlation developed by Dittus and Boelter (Bergman et al., 2011). The study of flow boiling is concerned with the description of fluids subjected to a change of phase (evaporation or condensation) and it originates its theory from a combination of the available models describing pool (or nucleate) boiling and single phase convective heat transfer. Convective vaporization is described in literature through a number of correlations that vary depending on the geometry, the fluid, and the range of applicability. Webb and Gupte (1992) reviewed several models and grouped them in three main categories: superposition, asymptotic, and enhancement models.

The "superposition" model estimates the convective vaporization heat transfer (h) as the summation of nucleate boiling heat transfer (h_{nb}) and single phase convective evaporation heat transfer (h_{cv}). Chen (1966) suggested that flow velocity can suppress nucleate boiling and for this reason he proposed the use of a "suppression factor", S, as a corrective multiplier for the nucleate boiling ($S \cdot h_{nb}$). The two-phase convective evaporation heat transfer is calculated as a function of the liquid phase convective heat transfer (h_L), multiplied by a "two-phase convection multiplier", F ($h_{cv} = F \cdot h_L$). The plot reported in Figure A.1 exemplifies the superposition of the two heat transfer contributions. Therefore:

$$h = S \cdot h_{nb} + F \cdot h_L \tag{A.1}$$

The plot reported in Figure A.1 exemplifies the superposition of the two heat transfer contributions. It is interesting to observe that the vaporization curve (h) is asymptotic to h_{cv} at low heat fluxes, and it is asymptotic to $S \cdot h_{nb}$ at high heat fluxes.

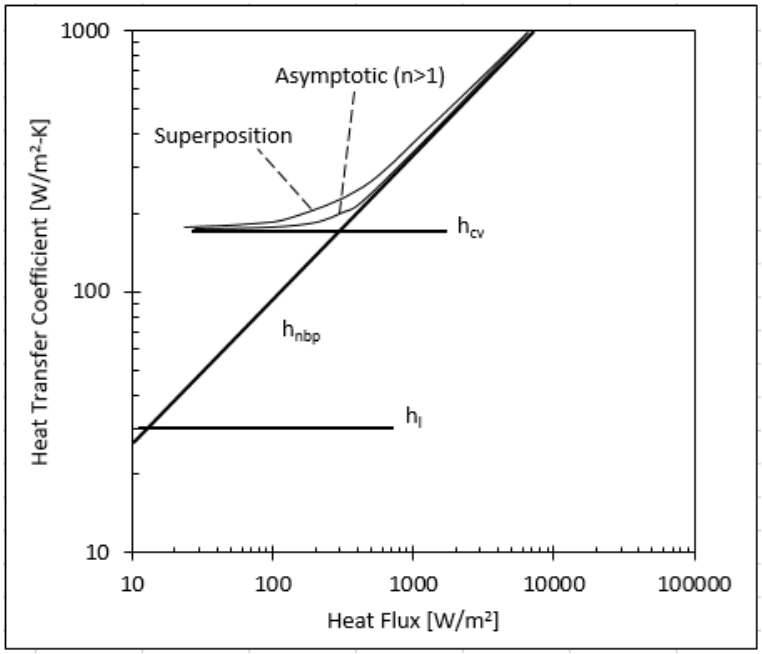


Figure A.1: Superposition model (figure adapted from Webb and Gupte (1992)).

Rearranging Equation A.1, F can be determined directly from experimental data (see Equation A.2):

$$F = \frac{h - S \cdot h_{nb}}{h_L} \tag{A.2}$$

The "asymptotic" model was first proposed by Kutateladze (1961) and it introduces the use of an exponent, n, chosen according to best fit. When n increases, the curved region describing h in Figure A.1 contracts. Equation A.3 describes h:

$$h^n = (S \cdot h_{nb})^n + (F \cdot h_L)^n \tag{A.3}$$

and F is similarly obtained from :

$$F = \frac{(h^n - (S \cdot h_{nb})^n)^{1/n}}{h_L} \tag{A.4}$$

Finally, the "enhancement" model introduced by Shah (1976) calculates h as in Equation A.5:

$$h = E \cdot h_L \tag{A.5}$$

where E is an enhancement factor, function of the boiling number (Bo), of the convection number (Co), and of the Freude number (Fr).

A.1 Procedure to Estimate the Coefficients S and F

This section explains the procedure of how the coefficients S and F were estimated in this work, based on the the algorithm described by Chen (1966) and by Sawant (2012). The values of h_{nb} and h_L were previously estimated applying the models by Kedzierski (2012) and Buongiorno (2006). The friction factor for finned surface ($f_{turb,finned}$) was estimated according to Jensen and Vlakancic (1999), as described in Section 5.4.3. The experimental two-phase flow heat transfer coefficient (h_{exp}) was obtained from the measurements described in Section 4.4.7 and reported in Appendix B.

A.1.1 Chen Correlation

For a series of data, a first estimate of $F_{n=1}$ is obtained from plotting h_{exp}vs$\frac{1}{X_{tt}}$, where X_{tt} is the Martinelli parameter calculated as: $\left(\frac{1-x}{x}\right)^{0.9}\left(\frac{\rho_v}{\rho_L}\right)^{0.5}\left(\frac{\mu_L}{\mu_v}\right)^{0.1}$. The Reynolds number is Re = $\frac{G(1-x)D_h}{\mu_L}$.

In the order, and starting from $n = 1$, the next steps are followed:

$$h_{macro,n} = F_n \cdot h_L$$

$$h_{micro,n} = h_{exp} - h_{macro,n}$$

$$Re_{tp} = Re \cdot F_n^{1.25}$$

At this point, the first estimate of $S_{n=1}$ is obtained similarly to $F_{n=1}$ by plotting $\frac{h_{micro}}{h_{nb}}$ vs Re_{tp}. From the $S_{n=1}$, the next guess of $F_{n=2}$ is obtained as:

$$F_{n=2} = \frac{h_{exp} - S_{n=1} \cdot h_{nb}}{h_L}$$

This procedure is repeated n times until the functions for F and S remain relatively constant. The parameterization of F_{par} and S_{par} was chosen as follows:

$$F_{par} = 1 + a \cdot \left(\frac{1}{X_{tt}}\right)^b$$
$$S_{par} = \frac{1}{1 + c \cdot (Re_{tp})^d}$$

where a, b, c, and d are coefficients found by minimizing the difference between the values of F_n and F_{par}, and S_n and S_{par} (using the Least Squares method).

A.1.2 Sawant Correlation

In the original work by Sawant (2012) it was not specified what formulation of the Reynolds number was used, and here it was assumed that $Re = \frac{G \cdot D_h}{\mu_L}$. For a series of data, the value of S_{saw} was obtained from Equation A.6, assuming F to be equal to unity.

$$S_{saw} = \frac{h_{exp} - h_L}{h_{nb}} \tag{A.6}$$

The value of F_{saw} was then estimated from Equation A.7:

$$F_{saw} = \frac{h_{exp} - S_{saw} \cdot h_{nb}}{h_L} \tag{A.7}$$

The parameterization of S_{par} and F_{par} was chosen as follows:

$$S_{par} = a \cdot e^{-b \cdot Re \cdot x}$$

$$F_{par} = c \cdot \left(\frac{1}{X_{tt} \cdot Re \cdot x} \right)^{d}$$

where a, b, c, and d are coefficients found by minimizing the difference between the values of F_{saw} and F_{par}, and S_{saw} and S_{par} (using the Least Squares method).

Appendix B

Two-Phase Flow Experimental Data Set

The data set reported below represents the series of tests conducted during the experimental campaign on flow performances during two-phase flow. At the beginning of each test series, the table lists the fluid tested. For each test, the conditions at the inlet of the preheater and of the test section of the experimental setup presented in Section 4.2.5 are provided, together with the measured heat transfer coefficient and pressure drop.

				PREHEATER			TEST SECTION					
Fluid	**OMF**	**NMF**	**Mass Flow Rate [kg/s]**	**Q [W]**	**P_in [kPa]**	**T_in [°C]**	**Q [W]**	**P_ave [kPa]**	**h_in [kJ/kg]**	**T_wall [°C]**	**HTC [kW/m²K]**	**ΔP [kPa]**
R410A	0	0	0.01517	510.92	930.63	2.56	1004.45	926.59	237.57	6.01	6.14	6.18
	0	0	0.01517	848.31	933.03	2.07	1008.62	925.38	259.07	5.99	6.17	8.40
	0	0	0.01521	1028.07	935.33	1.71	1007.07	925.91	270.19	6.00	6.26	9.68
	0	0	0.01525	1258.78	932.58	0.93	1010.93	920.69	283.98	5.81	6.12	10.99
	0	0	0.01524	1524.11	934.73	0.06	987.81	920.79	300.13	5.78	6.13	12.26
	0	0	0.01508	1650.58	937.87	-0.45	987.35	923.33	308.79	5.84	6.40	12.53
	0	0	0.01505	1826.15	936.23	-1.21	987.47	919.55	319.52	5.72	6.24	13.16
	0	0	0.01521	1965.41	940.59	-1.63	986.85	922.33	326.76	5.79	6.49	13.67
	0	0	0.01535	2163.21	939.73	-2.69	993.94	918.72	336.87	5.61	6.57	14.41
	0	0	0.01526	2443.12	940.62	-7.63	987.71	918.61	348.68	5.40	7.43	14.47
	0	0	0.01542	2793.52	942.44	-18.55	984.37	922.42	353.78	5.49	7.90	14.31
	0	0	0.01539	2534.65	949.78	-8.25	978.39	928.02	352.41	5.64	8.55	14.24
R410A	0	0	0.01521	534.47	930.30	2.43	1199.64	927.95	238.86	6.14	7.17	6.76
	0	0	0.01506	991.80	930.84	1.51	1206.61	924.01	268.16	6.02	7.10	9.78
	0	0	0.01506	1297.02	934.11	0.54	1207.29	924.09	286.96	6.01	7.25	11.16
	0	0	0.01506	1567.00	934.01	-0.67	1205.89	920.92	303.05	5.90	7.16	12.26
	0	0	0.01529	1753.03	936.76	-1.42	1204.36	921.55	312.50	5.92	7.24	12.90
	0	0	0.01532	1896.32	936.43	-2.15	1207.68	919.67	320.54	5.88	7.12	13.18
	0	0	0.01501	2015.24	936.18	-2.91	1209.13	918.62	329.91	5.74	7.49	13.02
	0	0	0.01515	2174.94	942.59	-4.43	1205.43	923.91	336.89	5.78	8.45	13.27
	0	0	0.01545	2288.86	938.41	-6.69	1212.50	919.04	338.17	5.57	8.31	13.93
R410A	0	0	0.02083	549.25	932.30	1.99	1215.81	925.96	229.40	6.13	6.87	8.30
	0	0	0.02068	1007.71	935.65	1.28	1217.22	923.59	250.69	6.03	6.92	11.90
	0	0	0.02088	1353.73	935.00	0.58	1215.81	917.90	265.71	5.82	6.75	14.79
	0	0	0.02076	1724.65	943.29	-0.23	1210.15	920.37	282.72	5.88	7.01	16.93
	0	0	0.02114	2214.82	949.33	-1.12	1210.35	918.71	303.07	5.77	7.20	20.45
	0	0	0.02080	2678.59	954.96	-2.52	1205.34	915.62	325.00	5.74	6.84	22.86
	0	0	0.02055	3022.03	956.26	-3.90	1219.22	911.75	341.16	5.86	6.13	23.60
	0	0	0.02046	3204.98	960.07	-4.64	1216.39	913.36	349.68	6.07	5.84	23.52
	0	0	0.02033	3395.96	962.01	-5.87	1215.10	913.29	358.28	6.06	5.89	23.55
	0	0	0.02041	3464.06	962.75	-6.40	1214.47	913.02	360.11	6.02	5.95	23.77
R410A	0	0	0.02088	593.26	930.09	2.10	969.29	923.95	231.60	6.00	5.54	7.64
	0	0	0.02083	586.19	929.95	2.10	972.11	923.92	231.34	6.01	5.54	7.50
	0	0	0.02039	1223.85	936.17	1.07	972.26	922.47	261.65	5.90	5.76	12.41
	0	0	0.02062	1641.63	942.38	0.54	970.38	921.90	280.46	5.83	5.92	15.38
	0	0	0.02072	2043.92	945.96	-0.38	968.99	918.85	298.08	5.71	5.90	18.06
	0	0	0.02081	2382.12	948.49	-1.29	971.21	914.24	312.54	5.55	5.78	20.85
	0	0	0.02092	2718.32	956.26	-2.20	972.85	915.59	326.60	5.60	5.89	22.69

	0	0	0.02085	3032.61	957.42	-3.20	970.95	910.82	340.65	5.46	5.66	23.79
	0	0	0.02095	3280.70	961.72	-3.97	969.49	911.24	350.59	5.49	5.67	25.00
	0	0	0.02110	3545.12	964.79	-5.13	966.90	910.20	360.34	5.60	5.29	25.69
	0	0	0.02105	3875.37	967.29	-6.99	970.10	908.90	373.63	5.93	4.55	25.84
R410A	0	0	0.02696	948.18	938.22	1.25	1214.20	922.60	237.07	6.05	6.62	13.67
	0	0	0.02685	1700.01	948.33	0.19	1214.17	918.36	263.60	5.87	6.63	19.74
	0	0	0.02732	2217.64	953.96	-0.79	1210.86	916.43	279.98	5.79	6.63	22.81
	0	0	0.02716	3118.44	968.53	-2.62	1223.73	910.20	310.86	5.59	6.50	30.09
	0	0	0.02691	3796.06	978.68	-4.62	1223.76	907.86	334.14	5.52	6.46	33.41
	0	0	0.02690	3802.95	978.84	-4.65	1223.56	907.84	334.39	5.53	6.44	33.45
	0	0	0.02687	4002.88	977.02	-5.46	1227.40	901.72	340.78	5.31	6.23	34.73
	0	0	0.02697	4228.09	981.77	-6.07	1217.16	901.79	347.69	5.31	6.22	35.64
	0	0	0.02694	4672.01	988.37	-7.55	1210.57	900.08	362.11	5.25	6.16	36.69
R410A	0	0	0.02083	888.00	933.75	1.76	1199.13	924.54	245.30	6.07	6.82	9.92
	0	0	0.02068	1007.71	935.65	1.28	1217.22	924.38	250.68	6.03	7.04	13.45
	0	0	0.02089	1354.44	935.00	0.58	1215.81	918.64	265.74	5.82	6.87	16.28
	0	0	0.02086	1611.83	939.37	-0.07	1207.62	919.55	277.17	5.89	6.79	15.12
	0	0	0.02073	2532.74	949.84	-2.99	1212.96	914.91	317.65	5.67	6.97	20.04
	0	0	0.02072	2824.66	953.44	-3.21	1209.01	913.05	331.49	5.57	7.05	21.08
	0	0	0.02100	3153.01	957.66	-4.30	1211.93	911.40	343.65	5.45	7.25	22.31
	0	0	0.02064	1985.92	944.00	-1.22	1221.35	918.07	294.37	5.77	7.11	17.39
	0	0	0.02105	3795.63	978.22	-8.77	1180.59	927.29	367.24	7.45	4.54	22.36
R410A	0	0	0.02072	3425.89	963.42	-6.13	969.75	914.98	356.17	5.35	6.74	22.49
	0	0	0.02086	3713.51	957.69	-7.99	972.51	904.77	366.06	5.14	5.81	23.16
	0	0	0.02083	3914.65	963.23	-9.47	968.56	909.38	373.78	5.40	5.73	23.21
	0	0	0.02091	1000.65	933.58	1.60	975.24	923.38	250.28	5.78	6.20	10.23
	0	0	0.02088	1424.18	939.02	0.79	968.47	922.87	269.39	5.74	6.29	13.56
	0	0	0.02091	1771.13	942.80	-0.15	965.40	921.17	284.48	5.68	6.26	15.91
	0	0	0.02078	2169.05	947.25	-1.28	954.90	919.41	302.44	5.59	6.27	18.32
	0	0	0.02075	2585.26	951.34	-2.93	962.41	916.33	320.15	5.45	6.36	20.39
	0	0	0.02082	2897.91	955.76	-3.97	958.42	915.01	333.22	5.40	6.36	21.65
	0	0	0.02075	3185.11	958.92	-5.05	971.57	913.65	345.91	5.30	6.63	22.39
R410A	0	0	0.01519	615.70	927.37	2.61	967.75	926.07	244.52	5.86	6.36	6.34
	0	0	0.01504	998.80	928.85	1.68	961.10	923.79	268.95	5.78	6.30	8.67
	0	0	0.01504	998.80	928.85	1.68	961.10	923.79	268.95	5.78	6.30	8.67
	0	0	0.01514	1332.84	931.90	0.58	971.70	923.32	288.93	5.73	6.56	10.25
	0	0	0.01509	1609.64	933.35	-0.73	970.64	921.79	305.54	5.64	6.70	11.34
	0	0	0.01521	1770.85	936.26	-1.20	968.58	922.77	314.57	5.66	6.84	11.82
	0	0	0.01507	1977.76	936.48	-2.21	973.19	921.80	327.95	5.57	7.13	11.84
	0	0	0.01514	2130.35	937.31	-3.10	970.98	920.35	336.02	5.52	7.05	12.34
	0	0	0.01513	2179.80	937.36	-3.45	970.90	919.99	338.88	5.50	7.07	12.35

	0	0	0.01513	2229.84	937.74	-3.75	969.28	920.06	341.74	5.72	6.31	12.30
	0	0	0.01517	2750.17	946.77	-6.70	953.73	925.12	371.27	7.97	3.08	12.22
R410A	0	0	0.01530	650.25	927.08	2.14	973.10	925.77	245.78	5.82	6.47	6.41
	0	0	0.01519	644.20	929.72	2.41	1211.81	927.41	246.07	6.09	7.38	6.92
	0	0	0.01511	1266.26	932.61	0.39	1205.02	924.66	284.41	5.96	7.52	10.26
	0	0	0.01511	957.16	931.91	1.55	1198.31	927.17	265.69	6.06	7.47	8.87
	0	0	0.01514	1487.87	933.27	-0.65	1203.42	922.87	297.30	5.86	7.59	10.97
	0	0	0.01511	1846.27	933.80	-2.45	1207.29	919.85	318.47	5.72	7.70	11.67
	0	0	0.01508	1983.57	938.89	-2.71	1202.07	923.76	327.44	5.86	7.98	11.63
	0	0	0.01500	2113.87	941.58	-3.49	1191.76	925.07	335.69	5.91	8.02	11.73
	0	0	0.01517	2239.49	937.94	-5.17	1207.63	920.41	339.91	5.90	7.25	12.02
	0	0	0.01533	2439.41	936.04	-6.12	1190.15	916.25	349.97	10.42	2.19	12.20
R410A + POE	0.01	0	0.02115	945.75	936.57	1.49	1204.14	925.43	246.99	6.26	6.57	10.62
	0.01	0	0.02077	1374.74	938.03	0.27	1206.38	922.23	266.61	6.09	6.71	13.87
	0.01	0	0.02056	1729.10	943.73	-0.88	1205.60	923.28	282.78	6.06	7.08	15.65
	0.01	0	0.02066	2049.89	944.79	-2.00	1202.38	918.98	296.21	5.89	6.97	17.51
	0.01	0	0.02074	2283.77	947.78	-2.80	1203.61	917.56	305.89	5.84	6.99	18.95
	0.01	0	0.02081	2557.53	952.86	-3.72	1203.11	917.45	317.28	5.85	7.03	20.55
	0.01	0	0.02091	2881.97	957.22	-4.71	1195.54	915.58	330.73	5.86	6.79	21.96
	0.01	0	0.02045	3283.85	956.51	-6.71	1203.58	911.05	350.55	6.08	5.85	22.02
	0.01	0	0.02081	3582.64	953.33	-8.76	1206.39	902.45	359.09	6.46	4.52	22.98
R410A + POE	0.03	0	0.02110	969.95	935.44	1.10	1204.20	924.73	247.64	5.76	8.73	12.39
	0.03	0	0.02091	966.00	933.78	1.21	1201.59	923.09	248.03	5.70	8.60	11.48
	0.03	0	0.02101	1355.71	938.95	0.00	1213.50	923.15	264.53	5.75	8.64	13.81
	0.03	0	0.02060	1728.71	936.45	-1.65	1211.40	914.86	281.43	5.51	8.06	16.35
	0.03	0	0.02033	1725.77	936.09	-1.71	1211.88	915.00	282.31	5.51	8.11	16.15
	0.03	0	0.02066	2050.79	944.23	-2.78	1216.67	917.84	295.08	5.64	8.40	17.87
	0.03	0	0.02067	2212.28	950.49	-3.35	1212.90	921.51	302.01	5.76	8.83	18.73
	0.03	0	0.02100	916.27	914.74	0.80	1220.48	904.23	244.84	5.02	7.49	11.44
	0.03	0	0.02115	1573.86	917.36	-1.61	1215.09	897.18	271.99	4.83	7.10	16.04
	0.03	0	0.02115	2173.84	933.93	-3.64	1214.68	904.14	297.29	5.12	7.64	19.85
	0.03	0	0.02111	2802.94	934.35	-6.58	1220.16	892.02	322.94	4.80	7.04	23.44
	0.03	0	0.02091	3103.68	938.35	-7.78	1216.19	890.70	336.81	4.95	6.69	24.24
R410A + POE	0.03	0	0.01495	449.12	911.16	2.07	1214.11	909.79	233.19	5.22	7.79	7.11
	0.03	0	0.01544	1380.27	916.67	-2.04	1211.85	906.21	286.34	5.28	7.41	12.32
	0.03	0	0.01526	1183.94	917.95	-1.08	1212.45	909.74	275.96	5.38	7.59	11.34
	0.03	0	0.01532	1184.03	917.39	-1.08	1212.21	909.18	275.66	5.36	7.57	11.49
	0.03	0	0.01538	1503.23	918.34	-2.79	1212.06	906.29	293.56	5.29	7.52	12.99
	0.03	0	0.01521	760.80	911.20	0.87	1205.78	908.49	251.35	5.26	7.50	9.21
	0.03	0	0.01517	1245.01	913.85	-1.43	1209.50	906.30	279.92	5.32	7.21	11.46
	0.03	0	0.01528	1740.23	921.48	-4.18	1209.36	906.85	307.61	5.32	7.78	13.45

	0.03	0	0.01528	1957.58	925.72	-5.16	1200.72	909.01	320.39	5.61	7.35	13.81
	0.03	0	0.01551	2196.81	926.67	-6.90	1197.93	907.00	331.34	5.86	6.54	14.30
	0.03	0	0.01546	778.64	900.15	0.15	1272.69	896.14	250.61	4.93	6.88	9.31
R410A + POE	0.03	0	0.01527	660.85	915.09	1.62	973.33	912.60	245.74	5.24	6.68	7.48
	0.03	0	0.01525	971.07	913.97	0.22	970.46	909.09	264.00	5.15	6.53	9.22
	0.03	0	0.01513	1315.68	916.17	-1.31	971.50	907.57	284.99	5.13	6.54	11.12
	0.03	0	0.01513	1605.70	920.32	-2.90	965.62	908.33	301.79	5.20	6.65	12.38
	0.03	0	0.01523	1886.66	921.99	-4.47	963.82	906.53	317.17	5.18	6.63	13.36
	0.03	0	0.01523	2143.00	924.35	-5.94	959.87	905.63	331.80	5.44	5.96	14.03
	0.03	0	0.01527	2316.69	927.12	-6.99	957.72	905.64	341.29	5.38	6.47	14.62
	0.03	0	0.01543	2581.28	926.68	-8.13	958.71	901.03	355.18	5.96	5.02	15.46
R410A + POE	0.03	0	0.02056	1011.55	921.81	0.94	956.82	911.45	250.64	5.14	6.69	11.79
	0.03	0	0.02072	1428.41	924.51	-0.61	965.36	907.95	268.03	5.07	6.52	14.93
	0.03	0	0.02081	1735.03	928.61	-1.70	955.84	906.44	280.82	5.04	6.41	17.12
	0.03	0	0.02101	2002.85	930.67	-2.65	961.42	904.14	291.31	4.96	6.43	18.78
	0.03	0	0.02091	2129.78	933.48	-3.11	959.93	904.93	297.16	5.00	6.50	19.43
	0.03	0	0.02088	2614.41	939.24	-5.30	968.51	900.89	317.27	4.92	6.40	22.60
	0.03	0	0.02090	2968.03	940.45	-6.67	967.06	894.95	332.01	4.80	6.08	24.34
	0.03	0	0.02088	2950.67	944.61	-6.36	963.57	899.81	331.77	4.96	6.34	23.98
	0.03	0	0.02087	3168.14	945.95	-7.41	961.72	896.95	340.72	4.93	6.17	24.83
	0.03	0	0.02081	3440.03	949.76	-9.07	962.21	898.39	351.75	5.33	5.62	25.24
	0.03	0	0.02076	3801.62	952.64	-11.45	959.52	894.52	366.14	5.15	6.67	26.41
R410A	0	0	0.02065	2670.01	951.73	-3.98	1197.84	913.09	323.30	5.82	6.22	22.52
	0	0	0.02065	2670.01	951.73	-3.98	1197.84	913.09	323.30	5.82	6.22	22.52
	0	0	0.02113	3199.35	957.23	-4.88	1193.29	907.62	344.06	5.69	5.91	24.81
	0	0	0.02096	3222.29	963.09	-4.44	1193.57	913.19	347.08	5.68	6.68	24.47
	0	0	0.02092	2560.18	955.14	-2.35	1207.60	916.32	318.81	5.81	6.73	22.65
	0	0	0.02081	2852.43	958.58	-3.34	1227.40	915.22	332.01	5.75	6.94	23.04
	0	0	0.02119	1185.00	939.86	1.29	1240.20	924.97	257.90	6.18	6.83	13.83
R410A + POE + Al2O3	0.01	0.02	0.02076	971.79	927.45	1.11	1207.04	915.92	248.50	6.12	5.73	12.02
	0.01	0.02	0.02092	545.25	924.93	2.19	1201.14	918.33	229.40	6.36	5.39	8.78
	0.01	0.02	0.02076	1410.26	934.08	0.78	1204.50	915.98	269.12	5.91	6.33	15.34
	0.01	0.02	0.02076	2017.33	942.09	0.27	1208.08	912.53	297.58	5.71	6.58	19.88
	0.01	0.02	0.02071	2286.24	945.59	-0.17	1205.18	909.80	310.12	5.58	6.61	22.07
	0.01	0.02	0.02059	2699.99	951.86	-0.68	1207.04	908.04	330.12	5.52	6.71	23.75
	0.01	0.02	0.02085	2980.98	957.08	-1.12	1206.63	907.36	341.27	5.64	6.36	24.88
	0.01	0.02	0.02095	3393.25	957.00	-2.04	1190.16	899.56	358.88	6.31	4.45	25.97
R410A + POE + Al2O3	0.01	0.02	0.02090	779.05	927.49	1.69	955.64	919.22	239.86	6.00	5.06	9.87
	0.01	0.02	0.02088	1137.47	931.58	1.03	954.52	918.43	256.04	5.84	5.36	12.63

	0.01	0.02	0.02090	1451.09	933.85	0.73	954.09	915.71	270.55	5.67	5.52	15.26
	0.01	0.02	0.02095	1757.14	938.14	0.54	953.60	914.42	284.69	5.54	5.77	17.73
	0.01	0.02	0.02087	2027.67	943.41	0.37	951.83	914.18	297.73	5.51	5.87	20.08
	0.01	0.02	0.02089	2259.27	948.44	0.15	958.47	913.61	308.36	5.47	6.01	21.93
	0.01	0.02	0.02082	2606.10	953.70	-0.27	953.44	911.86	324.77	5.33	6.27	23.84
	0.01	0.02	0.02074	3051.66	958.18	-0.96	955.94	907.42	345.67	5.31	5.85	25.49
	0.01	0.02	0.02072	3244.10	961.03	-1.16	954.20	907.27	354.78	5.47	5.50	25.66
R410A + POE + Al2O3	0.01	0.02	0.01523	548.83	925.18	2.37	960.86	923.67	239.65	6.09	5.41	6.61
	0.01	0.02	0.01519	550.60	925.26	2.43	961.09	923.77	239.95	6.11	5.37	6.71
	0.01	0.02	0.01530	1312.23	930.23	1.72	958.08	919.25	288.37	5.77	5.89	11.85
	0.01	0.02	0.01547	923.13	928.32	1.92	961.17	922.84	262.58	6.01	5.62	9.46
	0.01	0.02	0.01514	1538.00	932.00	1.60	964.16	918.04	303.99	5.70	6.08	12.59
	0.01	0.02	0.01542	1705.59	935.70	1.50	965.30	919.31	312.91	5.73	6.25	13.43
	0.01	0.02	0.01528	1940.80	935.79	1.16	955.40	916.13	328.74	5.68	6.00	14.35
	0.01	0.02	0.01526	2150.68	937.03	0.78	958.21	914.41	342.16	5.64	6.00	14.85
	0.01	0.02	0.01516	2308.33	937.74	0.41	956.45	913.43	352.91	5.68	5.90	14.77
R410A + POE + Al2O3	0.03	0.02	0.02086	538.06	924.41	1.97	955.28	917.72	228.79	5.60	6.04	9.00
	0.03	0.02	0.02088	961.39	929.72	0.89	954.56	918.39	247.40	5.59	6.30	12.28
	0.03	0.02	0.02081	1272.53	933.58	0.61	955.18	917.19	262.10	5.56	6.32	14.95
	0.03	0.02	0.02089	1261.29	933.84	0.60	952.29	917.56	261.29	5.59	6.27	14.50
	0.03	0.02	0.02076	1664.00	939.02	0.32	954.70	915.02	280.64	5.48	6.39	18.58
	0.03	0.02	0.02081	2001.56	942.94	-0.18	957.77	912.36	295.94	5.41	6.36	20.95
	0.03	0.02	0.02085	2355.58	948.89	-0.92	959.91	910.00	311.58	5.34	6.41	23.85
	0.03	0.02	0.02097	2708.64	954.24	-1.57	957.91	906.94	326.83	5.24	6.48	26.17
	0.03	0.02	0.02090	3078.59	959.88	-2.28	953.51	904.20	343.83	5.25	6.38	27.75
	0.03	0.02	0.02092	3453.79	966.57	-3.18	952.91	902.61	360.30	5.57	5.96	29.14
R410A + POE + Al2O3	0.03	0.02	0.01522	670.46	925.04	1.23	959.50	922.14	245.92	5.81	6.31	8.14
	0.03	0.02	0.01527	669.73	925.63	1.55	959.12	922.62	246.23	5.81	6.37	8.28
	0.03	0.02	0.01522	1040.56	928.19	1.49	958.64	920.40	270.62	5.73	6.48	10.48
	0.03	0.02	0.01520	1257.31	929.19	1.39	959.06	918.35	284.84	5.69	6.42	11.92
	0.03	0.02	0.01525	1453.12	928.98	1.02	959.77	915.35	296.82	5.63	6.26	13.11
	0.03	0.02	0.01516	1664.85	932.49	0.51	959.90	915.81	310.61	5.76	6.13	13.83
	0.03	0.02	0.01523	1968.39	934.81	-0.14	959.38	913.75	329.07	5.79	6.08	15.16
	0.03	0.02	0.01516	2062.03	936.43	-0.27	957.59	913.94	335.64	5.80	6.28	15.42
R410A + POE + Al2O3	0.03	0.02	0.01527	669.73	925.63	1.55	959.12	922.62	246.23	5.81	6.37	8.28
	0.03	0.02	0.01521	1036.59	928.09	1.50	957.30	920.31	270.44	5.73	6.45	10.51
	0.03	0.02	0.01520	1257.31	929.19	1.39	959.06	918.35	284.84	5.69	6.42	11.92
	0.03	0.02	0.01526	1494.21	932.39	1.06	958.17	918.37	299.55	5.71	6.54	13.21

	0.03	0.02	0.01515	1662.82	932.37	0.51	958.96	915.81	310.53	5.76	6.14	13.89
	0.03	0.02	0.01523	1967.94	934.84	-0.14	959.25	913.79	329.05	5.79	6.09	15.16
	0.03	0.02	0.01508	2157.22	937.98	-0.42	955.52	913.74	342.45	5.81	6.50	15.73
	0.03	0.02	0.01510	2431.14	933.96	-0.69	945.10	905.08	360.01	7.18	3.71	16.32
R410A + POE + Al2O3	0.01	0.1	0.01545	601.89	926.10	2.03	959.02	922.81	242.05	6.53	4.45	8.40
	0.01	0.1	0.01513	943.63	928.99	1.94	960.46	920.59	265.32	6.25	4.84	11.44
	0.01	0.1	0.01514	1224.45	931.64	1.73	958.76	918.81	283.52	6.00	5.23	13.84
	0.01	0.1	0.01507	1484.18	934.62	1.43	959.18	918.00	300.68	5.85	5.58	15.28
	0.01	0.1	0.01517	1716.16	936.36	1.03	964.04	916.12	314.67	5.71	5.82	16.34
	0.01	0.1	0.01510	1856.41	939.38	0.77	964.23	916.58	324.15	5.73	5.92	17.15
	0.01	0.1	0.01521	2035.59	938.87	0.26	964.49	912.48	334.24	5.59	5.82	18.25
	0.01	0.1	0.01524	2148.90	942.34	0.02	958.53	913.71	341.05	5.71	5.70	18.84
	0.01	0.1	0.01518	2364.94	943.55	-0.53	962.92	911.66	354.97	5.93	5.13	19.04
R410A + POE + Al2O3	0.01	0.1	0.02078	606.45	926.38	2.01	958.02	917.94	232.25	6.30	4.36	10.05
	0.01	0.1	0.02071	1053.08	933.13	1.08	958.86	918.07	252.50	6.08	4.83	14.54
	0.01	0.1	0.02078	1053.53	933.41	1.11	959.08	917.92	252.39	6.06	4.85	14.61
	0.01	0.1	0.02074	1445.20	937.50	0.78	962.72	914.71	270.87	5.74	5.27	18.65
	0.01	0.1	0.02078	1761.42	943.61	0.47	964.00	914.47	285.47	5.61	5.64	21.52
	0.01	0.1	0.02085	2045.61	947.36	0.05	963.43	911.24	298.16	5.45	5.70	24.70
	0.01	0.1	0.02076	2280.75	951.68	-0.37	960.03	908.71	309.28	5.31	5.76	27.36
	0.01	0.1	0.02081	2626.32	958.15	-0.98	961.16	906.01	324.73	5.11	6.07	30.23
	0.01	0.1	0.02071	3038.25	963.53	-1.53	957.95	901.77	344.36	4.93	6.13	32.04
	0.01	0.1	0.02089	3364.78	970.63	-2.46	960.46	901.47	357.40	5.15	5.64	33.25
R410A + POE + Al2O3	0.03	0.1	0.01517	557.29	921.26	1.98	953.50	918.32	239.76	5.81	5.64	8.10
	0.03	0.1	0.01512	956.65	924.59	1.58	955.17	916.47	265.69	5.66	5.99	11.30
	0.03	0.1	0.01513	1177.64	927.75	1.29	954.57	916.30	279.79	5.63	6.19	12.99
	0.03	0.1	0.01530	1489.17	932.02	0.60	956.32	915.66	298.28	5.61	6.40	15.25
	0.03	0.1	0.01540	1738.84	934.91	0.07	953.11	913.98	313.06	5.58	6.41	16.97
	0.03	0.1	0.01516	1936.91	937.00	-0.47	955.33	912.76	327.03	5.60	6.47	17.68
	0.03	0.1	0.01517	2028.88	936.49	-0.75	956.76	910.42	332.66	5.58	6.30	18.23
	0.03	0.1	0.01513	2137.88	939.97	-0.97	949.84	911.40	339.84	5.68	6.34	18.94
	0.03	0.1	0.01512	2285.93	941.23	-1.33	953.83	909.30	349.20	5.75	6.31	19.64
R410A + POE + Al2O3	0.03	0.1	0.02067	1245.53	935.97	0.65	958.15	916.67	261.25	5.52	6.37	16.75
	0.03	0.1	0.02105	583.83	926.75	1.76	955.04	918.20	230.41	5.67	5.92	10.05
	0.03	0.1	0.02081	1608.83	940.45	0.27	956.24	913.24	277.74	5.35	6.53	21.34
	0.03	0.1	0.02098	1894.41	944.63	-0.15	957.45	910.11	290.08	5.27	6.38	24.44
	0.03	0.1	0.02080	2209.04	950.30	-0.86	956.61	907.69	304.90	5.17	6.46	27.58
	0.03	0.1	0.02072	2574.84	956.13	-1.49	956.52	903.65	322.00	5.05	6.45	30.45

	0.03	0.1	0.02096	2886.89	962.04	-2.28	951.94	901.58	334.31	4.94	6.72	32.53
R410A + POE + Al2O3	0.01	0.2	0.01463	2457.39	924.36	-10.65	963.08	914.04	352.10	6.16	6.62	13.23
	0.01	0.2	0.01484	2262.85	922.44	-8.97	965.18	912.50	339.12	6.07	6.74	13.78
	0.01	0.2	0.01496	2055.45	922.59	-7.34	965.97	913.73	326.38	6.03	6.80	13.40
	0.01	0.2	0.01502	1866.64	912.73	-6.47	966.90	905.45	314.56	5.75	7.38	12.89
	0.01	0.2	0.01524	1575.72	919.54	-3.80	966.62	914.01	297.70	6.09	6.37	11.80
	0.01	0.2	0.01512	1302.79	916.49	-2.04	966.09	913.06	283.07	6.15	6.04	10.71
	0.01	0.2	0.01524	1037.43	913.68	-0.41	966.39	912.18	267.47	6.21	5.74	9.42
	0.01	0.2	0.01547	589.85	912.12	1.60	964.04	913.82	240.57	6.42	5.12	6.89
R410A + POE + Al2O3	0.03	0.2	0.01534	528.49	909.85	1.61	957.43	911.42	236.89	5.71	7.25	6.79
	0.03	0.2	0.01525	953.55	911.24	-0.17	953.76	910.60	262.26	5.63	7.82	9.03
	0.03	0.2	0.01518	1312.40	912.00	-3.42	954.68	909.48	281.33	5.55	8.42	10.42
	0.03	0.2	0.01528	1603.42	914.07	-5.70	951.32	909.06	296.38	5.58	8.42	12.11
	0.03	0.2	0.01534	1830.55	916.34	-7.23	950.13	909.64	308.52	5.59	8.62	13.10
	0.03	0.2	0.01514	2084.42	917.41	-9.03	944.21	910.13	324.16	5.61	8.81	13.78
	0.03	0.2	0.01509	2342.02	917.73	-11.00	942.05	908.49	338.86	5.61	9.08	14.59
	0.03	0.2	0.01517	2509.30	919.98	-12.07	946.45	908.92	347.50	5.77	8.46	15.00
	0.03	0.2	0.01511	2617.30	920.41	-12.50	944.59	908.60	354.67	5.93	7.84	15.15
R410A	0	0	0.01511	606.78	916.23	2.53	965.33	918.36	244.03	5.93	6.68	6.70
	0	0	0.01530	1543.49	920.45	-1.51	957.48	914.56	298.60	5.98	6.65	11.49
	0	0	0.01530	1966.59	922.27	-4.30	961.12	914.21	322.05	5.85	7.36	12.39
R410A	0	0	0.02063	591.17	920.62	1.72	1192.17	915.76	231.27	6.35	6.48	9.07
	0	0	0.02037	1115.97	924.08	0.04	1198.62	916.84	254.84	6.36	6.70	12.31
	0	0	0.02077	1510.37	924.11	-0.63	1191.98	913.10	271.76	6.14	7.28	15.27
	0	0	0.02072	2343.54	931.29	-4.01	1189.45	911.91	307.06	6.04	7.86	20.06
	0	0	0.02093	2807.61	938.14	-5.33	1201.86	913.37	326.16	6.03	8.25	22.53
	0	0	0.02105	3273.89	942.62	-7.15	1194.79	913.61	344.82	5.98	8.66	23.68
	0	0	0.02101	3709.30	947.13	-9.86	1198.35	916.94	361.81	6.51	6.89	23.28
R410A	0	0	0.01075	455.20	911.01	2.68	949.80	914.40	246.43	5.95	6.42	4.41
	0	0	0.01068	666.61	907.60	1.85	964.36	909.54	265.24	5.85	6.77	5.32
	0	0	0.01073	853.55	913.08	0.99	965.32	914.36	281.08	5.97	6.71	5.90
	0	0	0.01078	1052.73	913.28	-0.37	962.53	914.27	297.10	5.90	7.07	6.47
	0	0	0.01086	1202.18	912.36	-1.16	955.41	912.71	308.98	5.75	7.68	6.83
	0	0	0.01090	1333.50	914.64	-2.53	945.67	914.79	318.46	5.72	8.00	6.90
R410A	0	0	0.02677	1021.37	922.26	0.93	963.01	913.76	239.58	6.04	5.95	13.58
	0	0	0.02667	1485.65	928.66	0.07	962.99	914.37	255.81	6.03	6.10	16.68
	0	0	0.02674	1865.62	932.84	-0.89	959.21	914.04	268.42	5.98	6.30	19.97
	0	0	0.02678	2240.88	938.54	-1.87	956.20	914.58	280.84	5.98	6.40	22.64
	0	0	0.02672	2648.35	944.68	-2.86	953.86	914.75	294.78	5.97	6.53	25.74
	0	0	0.02666	3203.32	951.18	-4.35	951.29	913.12	313.63	5.84	7.07	29.67

	0	0	0.02675	3587.67	957.60	-5.82	948.05	916.44	325.39	5.89	7.13	31.68
	0	0	0.02680	3975.43	963.38	-7.65	949.43	918.03	336.90	5.85	7.55	33.80
	0	0	0.02674	4525.73	966.81	-10.05	942.54	915.13	354.27	5.68	8.36	35.93
	0	0	0.02670	5104.06	972.61	-12.38	951.79	916.35	372.77	5.77	8.28	37.27
R410A + POE	0.005	0	0.01510	2561.22	930.26	-9.11	962.28	917.70	356.05	6.15	6.77	14.14
	0.005	0	0.01499	2282.98	927.74	-6.99	971.13	916.93	341.79	6.01	7.19	13.95
	0.005	0	0.01484	2068.93	925.14	-5.78	971.73	916.01	330.70	6.04	6.92	13.41
	0.005	0	0.01502	1823.02	923.46	-4.06	971.77	915.89	315.28	6.10	6.55	12.93
	0.005	0	0.01513	1487.64	923.17	-1.97	962.50	918.29	295.39	6.23	6.05	11.67
	0.005	0	0.01546	1121.78	920.60	-0.04	963.80	918.77	272.51	6.32	5.68	10.07
	0.005	0	0.01518	595.06	916.29	2.26	960.85	918.53	242.64	6.41	5.28	6.98
R410A + POE	0.005	0	0.02071	749.12	920.50	1.85	1183.40	918.59	239.00	6.61	5.94	10.82
	0.005	0	0.02051	1363.86	925.23	-0.25	1201.18	917.09	266.13	6.51	6.39	14.80
	0.005	0	0.02038	1742.94	928.72	-1.98	1198.81	916.67	282.53	6.45	6.61	16.63
	0.005	0	0.02044	2142.70	932.70	-3.83	1205.29	916.25	299.09	6.38	6.96	19.18
	0.005	0	0.02061	2532.11	939.68	-5.35	1202.45	918.47	314.86	6.42	7.03	21.47
	0.005	0	0.02048	2910.14	944.05	-6.93	1194.65	918.89	331.73	6.36	7.34	23.04
	0.005	0	0.02067	3249.82	944.38	-9.06	1193.49	915.99	343.68	6.16	8.04	24.32
	0.005	0	0.02048	3429.89	943.19	-10.50	1192.75	913.71	351.87	6.04	8.56	24.78
	0.005	0	0.02067	3589.25	947.33	-11.67	1186.83	916.50	356.25	6.11	8.44	25.00
R410A + POE + Al2O3	0.01	0.2	0.01095	470.74	907.89	1.95	955.75	911.77	245.94	6.00	6.19	4.56
	0.01	0.2	0.01078	659.99	903.57	0.88	956.34	906.89	262.58	5.78	6.86	5.32
	0.01	0.2	0.01077	863.76	904.54	-0.67	953.29	907.23	279.17	5.75	7.13	6.14
	0.01	0.2	0.01083	1044.70	905.38	-2.27	955.45	907.55	293.00	5.69	7.60	6.78
	0.01	0.2	0.01098	1182.22	907.37	-3.58	952.95	909.01	302.26	5.74	7.51	7.10
	0.01	0.2	0.01082	1375.19	907.56	-5.88	944.68	908.77	318.30	5.73	7.67	7.00
R410A + POE + Al2O3	0.01	0.2	0.01463	2457.39	924.36	-10.65	964.34	913.88	352.10	6.16	6.62	12.91
	0.01	0.2	0.01484	2262.85	922.44	-8.97	966.46	912.34	339.12	6.07	6.74	13.46
	0.01	0.2	0.01496	2055.45	922.59	-7.34	967.26	913.57	326.38	6.03	6.80	13.08
	0.01	0.2	0.01502	1866.64	912.73	-6.47	968.20	905.29	314.56	5.75	7.38	12.57
	0.01	0.2	0.01524	1575.72	919.54	-3.80	967.92	913.85	297.70	6.09	6.37	11.50
	0.01	0.2	0.01512	1302.79	916.49	-2.04	967.39	912.90	283.07	6.15	6.04	10.38
	0.01	0.2	0.01524	1037.43	913.68	-0.41	967.69	912.02	267.47	6.21	5.74	9.11
	0.01	0.2	0.01543	585.67	913.44	1.82	964.14	915.08	240.72	6.47	5.06	6.50
R410A + POE + Al2O3	0.01	0.2	0.02080	533.45	912.13	1.67	1204.12	910.83	228.19	6.84	5.29	9.15
	0.01	0.2	0.02073	1054.09	916.05	-0.05	1211.86	910.70	250.77	6.56	6.01	12.26
	0.01	0.2	0.02077	1442.24	920.02	-1.62	1212.21	911.20	266.98	6.35	6.67	14.57
	0.01	0.2	0.02086	1800.68	923.54	-3.76	1209.80	911.15	280.65	6.27	7.01	16.66

	0.01	0.2	0.02092	2209.61	926.52	-5.74	1203.71	909.66	296.99	6.11	7.59	19.51
	0.01	0.2	0.02100	2684.93	932.72	-8.11	1200.87	911.18	315.76	6.09	7.87	21.73
	0.01	0.2	0.02077	3211.36	937.22	-11.09	1208.02	910.90	338.13	5.96	8.78	23.50
	0.01	0.2	0.02103	3742.52	952.06	-13.85	1187.01	921.63	357.42	6.50	7.25	24.38
	0.01	0.2	0.02088	3776.84	944.97	-14.87	1192.76	914.01	358.85	6.60	6.63	24.03
R410A + POE + Al2O3	0.01	0.2	0.02566	708.37	917.52	1.47	964.43	912.94	229.84	6.34	5.16	10.88
	0.01	0.2	0.02539	1207.26	920.04	0.01	968.08	910.14	247.57	6.08	5.83	14.51
	0.01	0.2	0.02518	1638.80	916.76	-1.62	971.17	901.52	262.64	5.67	6.89	17.60
	0.01	0.2	0.02533	2009.23	930.53	-3.04	968.93	911.80	274.75	5.91	6.62	20.15
	0.01	0.2	0.02525	2428.83	931.73	-4.93	957.01	907.48	288.79	5.66	7.41	23.15
	0.01	0.2	0.02497	2863.23	935.86	-6.70	952.14	906.99	304.65	5.53	8.17	26.40
	0.01	0.2	0.02489	3309.67	946.59	-8.75	947.32	912.56	319.90	5.67	8.01	28.77
	0.01	0.2	0.02526	3862.08	954.24	-12.04	967.27	913.55	335.01	5.66	8.52	31.93
	0.01	0.2	0.02502	4347.54	959.73	-14.92	964.30	914.71	351.68	5.65	8.95	33.53
	0.01	0.2	0.02540	4992.38	961.68	-17.71	959.09	909.66	370.44	5.55	9.55	35.03
	0.01	0.2	0.02562	5437.06	968.60	-19.54	959.84	912.54	383.49	6.12	7.01	35.22
R410A + POE + Al2O3	0.03	0.2	0.01086	377.38	900.90	1.97	967.25	906.21	237.76	5.52	8.02	4.93
	0.03	0.2	0.01069	656.81	904.08	0.45	968.16	908.66	262.14	5.52	8.56	6.00
	0.03	0.2	0.01075	906.51	907.00	-1.82	965.66	910.90	281.61	5.62	8.43	6.73
	0.03	0.2	0.01095	1220.18	906.94	-5.18	963.78	909.61	303.67	5.62	8.73	7.48
	0.03	0.2	0.01090	1378.20	904.98	-7.22	971.57	906.77	315.65	5.62	8.79	7.55
R410A + POE + Al2O3	0.03	0.2	0.01534	528.49	909.85	1.61	958.75	911.30	236.89	5.71	7.25	6.54
	0.03	0.2	0.01525	953.60	911.09	-0.15	955.05	910.32	262.30	5.63	7.82	8.75
	0.03	0.2	0.01521	1307.17	913.82	-3.13	955.58	911.19	281.25	5.61	8.22	10.12
	0.03	0.2	0.01528	1608.04	912.52	-5.81	952.79	907.31	296.53	5.52	8.66	11.90
	0.03	0.2	0.01534	1827.39	917.02	-7.19	951.36	910.25	308.34	5.62	8.51	12.94
	0.03	0.2	0.01514	2084.59	917.40	-9.03	945.50	909.99	324.17	5.61	8.81	13.51
	0.03	0.2	0.01509	2341.96	917.72	-11.00	943.18	908.30	338.84	5.61	9.07	14.23
	0.03	0.2	0.01517	2509.23	919.98	-12.07	947.63	908.73	347.50	5.77	8.45	14.65
	0.03	0.2	0.01511	2617.44	920.41	-12.50	945.73	908.43	354.68	5.93	7.84	14.80
R410A + POE + Al2O3	0.03	0.2	0.02079	632.57	907.80	1.13	971.06	906.20	232.15	5.65	7.13	8.99
	0.03	0.2	0.02077	1109.22	910.13	-0.68	971.72	904.50	252.38	5.49	8.02	11.67
	0.03	0.2	0.02078	1549.74	916.17	-2.99	969.48	905.98	270.06	5.52	8.14	14.47
	0.03	0.2	0.02073	1935.59	922.64	-4.94	964.60	908.82	285.96	5.62	8.04	16.89
	0.03	0.2	0.02079	2308.75	923.98	-7.14	960.31	907.06	300.38	5.58	8.28	19.35
	0.03	0.2	0.02074	2841.13	931.29	-9.98	953.78	908.47	322.12	5.61	8.55	22.25
	0.03	0.2	0.02065	3395.48	937.11	-13.65	965.34	909.43	344.20	5.65	9.03	24.43
	0.03	0.2	0.02047	3707.97	934.70	-16.24	958.69	905.07	357.12	5.72	8.60	24.93

R410A + POE + Al2O3	0.03	0.2	0.02060	595.65	910.75	1.17	1204.65	908.78	230.70	6.44	6.18	9.62
	0.03	0.2	0.02072	1089.70	915.43	-0.90	1208.58	909.86	251.23	6.22	7.01	12.51
	0.03	0.2	0.02087	1553.74	918.09	-3.46	1209.30	907.60	269.24	6.04	7.69	15.10
	0.03	0.2	0.02086	1910.35	923.39	-5.44	1201.90	909.48	283.41	6.04	7.88	17.50
	0.03	0.2	0.02090	2292.06	927.63	-7.46	1198.61	910.56	298.51	6.09	7.90	19.97
	0.03	0.2	0.02088	2844.05	936.05	-10.52	1210.97	912.60	320.53	6.18	7.98	23.14
	0.03	0.2	0.02078	3393.40	939.68	-14.18	1196.37	911.38	342.31	6.16	8.30	24.98
	0.03	0.2	0.02061	3704.72	940.43	-16.51	1186.71	910.70	355.34	6.66	6.69	25.43
R410A + POE + Al2O3	0.03	0.2	0.02531	709.07	918.23	1.14	968.15	914.60	229.75	5.91	6.55	10.91
	0.03	0.2	0.02535	1190.35	919.91	-0.49	969.43	910.88	246.22	5.69	7.34	13.85
	0.03	0.2	0.02532	1708.78	924.74	-2.68	964.94	909.84	263.46	5.60	7.87	17.25
	0.03	0.2	0.02544	2135.40	926.38	-4.68	963.30	906.14	276.90	5.45	8.53	20.85
	0.03	0.2	0.02533	2207.04	934.29	-4.72	960.84	913.57	280.03	5.70	7.81	21.21
	0.03	0.2	0.02518	2760.33	941.64	-7.22	960.51	913.95	298.80	5.68	8.21	24.77
	0.03	0.2	0.02543	3236.86	940.64	-9.79	974.87	906.44	312.70	5.43	9.49	28.22
	0.03	0.2	0.02560	3851.68	954.17	-13.57	962.65	913.34	330.35	5.60	9.23	31.73
	0.03	0.2	0.02568	4212.13	955.59	-15.72	961.18	910.49	340.79	5.49	10.06	33.73
R410A + POE + Al2O3	0.03	0.2	0.02566	733.64	918.50	0.36	1615.13	912.76	229.14	6.27	9.13	14.16
	0.03	0.2	0.02514	1253.82	919.95	-1.65	1619.42	909.20	247.40	6.08	9.95	16.76
	0.03	0.2	0.02525	1708.87	924.50	-3.97	1610.18	908.93	261.71	6.05	10.21	20.07
	0.03	0.2	0.02517	2135.96	929.76	-5.93	1597.33	909.36	275.97	6.06	10.28	23.41
	0.03	0.2	0.02523	2621.22	935.71	-8.21	1602.06	909.13	291.63	6.08	10.41	26.74
	0.03	0.2	0.02507	3164.72	943.48	-10.94	1611.71	910.39	309.94	6.14	10.53	30.18
	0.03	0.2	0.02557	3699.19	948.31	-14.22	1610.60	908.51	323.61	6.09	10.93	33.51
	0.03	0.2	0.02555	3864.61	955.79	-15.16	1611.67	914.68	328.84	6.36	10.06	33.89

Appendix C

Code Script

The script is composed of nine files (.cpp) referred to here as modules:

1. get-input.cpp, module used to read and acquire the input data
2. main.cpp, module where the simulation ”use mode” is selected
3. tube-calorimeter.cpp, module simulating a two-phase flow in a pipe
4. test-unit.cpp, module used as a ”gym” to try new correlations or simulation environments
5. radial.cpp, module used for the two-phase radial analysis of the liquid film
6. ht-coef.cpp, module listing heat transfer correlations
7. p-drop.cpp, module listing pressure drop and friction factor correlations
8. void-fraction.cpp, module listing void fraction correlations and an inventory subroutine
9. fluid-props.cpp, module listing thermophysical properties correlations

In addition, there are two input files;

- in.txt, text file mainly used to define the ”use mode”, the fluid, and the tube geometry
- in.csv, comma separated value file used to list the input conditions of each test

```
1
2   #define _USE_MATH_DEFINES
3   #include <cmath>
4   #include "CoolProp.h"
5   #include "HumidAirProp.h"
6   #include <iostream>
7   #include <sstream>
8   #include <stdlib.h>
9   #include <fstream>
10  #include <string>
11  #include <cstddef>
12
13
14  using namespace CoolProp;
15  using namespace std;
16
17
18  double number[30];
19  string name[30];
20  string line;
21  ifstream input_file;
22
23
24  void ReadInputLines();
25
26  void GetInput(string &mode, string &ent_ref, string &fluid, double &A0, double &A1,
    double &A2, double &a0, double &b0,
27      string &oil, string &nano_mater, string &nano_shape, double &D_nano, string &
        tube_type, string &tube_mater,
28      string &orientation, double &tube_roughness, double &nSeg, double &nRad, double &
        lengthTube,
29      double &DPlengthTube, double &Dh, double &Do, double &Dr, double &tw,
30      double &heightFin, double &pitchFin, double &beta, double &alpha, double &nFins,
        double &Sp)
31  {
32      //use: read and acquire data from input file
33      //
34      //source:
35      //
36      //
37      //author: Andrea Bigi
38      //date: 06/2016
39      //-----------------------------------------------------------------
40
41      ////local variables
42      //string ent_ref;     //enthalpy reference (IIR, ASHRAE, NBP, DEF)
43      //string fluid;       //base fluid
44      //string oil;         //oil type (EMKARATE POE RL32-3MAF, ...)
45      //string nano_mater;  //nanoparticle material
46      //double D_nano;      //nanoparticle equivalent diameter, nm
47      //string nano_shape;  //nanoparticle shape
48
49      //string tube_type;   //tube type (smooth, microfin,...)
50      //string orientation; //tube orientation (horizontal, vertical, inclined)
51      //string tube_mater;  //tube material (copper, steel, ...)
52      //double tube_roughness;    //tube roughness, m
53      //double lengthTube;  //tube length, m
54      //double DPlengthTube;//pressure drop tube length, m
55      //double Dh;          //hydraulic diameter, m
56      //double nSeg;        //number of segments, -
57      //double Do;          //tube outside diameter, m
58      //double Dr;          //maximum inside diameter of micro-fin tube, m
59      //double tw;          //tube wall thickness at fin root, m
60      //double heightFin;   //fin height, m
61      //double pitchFin;    //fin pitch, m
62      //double beta;        //helix angle, deg
```

```
63      //double alpha;       //fin angle, deg
64      //double nFins;       //number of fins, -
65      //double Sp;          //perimeter of one fin and channel, m
66
67
68      //flow
69
70      ////BEGIN INPUT DATA ACQUISITION
71      //ifstream input_file;
72      input_file.open("in.txt");
73      while (getline(input_file, line))
74      {
75          if (line == "\"mode\"")
76          {
77
78              ReadInputLines();
79
80              mode = name[1];
81          }
82
83          else if (line == "\"fluid\"")
84          {
85
86              ReadInputLines();
87
88              ent_ref = name[1];
89              fluid = name[2];
90              A0 = number[1];
91              A1 = number[2];
92              A2 = number[3];
93              a0 = number[4];
94              b0 = number[5];
95              oil = name[3];
96              nano_mater = name[4];
97              nano_shape = name[5];
98              D_nano = number[6];
99          }
100
101         else if (line == "\"geometry\"")
102         {
103
104             ReadInputLines();
105
106             tube_type = name[1];
107             tube_mater = name[2];
108             orientation = name[3];
109             tube_roughness = number[1];
110             nSeg = number[2];
111             nRad = number[3];
112             lengthTube = number[4];
113             DPlengthTube = number[5];
114             Dh = number[6];
115             Do = number[7];
116             Dr = number[8];
117             tw = number[9];
118             heightFin = number[10];
119             pitchFin = number[11];
120             beta = number[12];
121             alpha = number[13];
122             nFins = number[14];
123             Sp = number[15];
124         }
125
126         else
127         {
128             exit;
```

```
        }
    }
    input_file.close();
    ////END INPUT DATA ACQUISITION
}

void ReadInputLines()
{
    //use: read input file and separate digits from strings
    //
    //source:
    //
    //
    //author: Andrea Bigi & Pratik Deokar
    //date: 06/2016
    //-----------------------------------------------------------------------

    //local variables

    double value;
    size_t found = 0;
    int i = 1;
    int j = 1;

    do
    {
        getline(input_file, line);
        found = line.find_first_of(",;");
        //------------------------------
        bool has_only_digits = true;
        for (size_t n = 0; n < found; n++)
        {
            if (!isdigit(line[n]))
            {
                has_only_digits = false;
                if (line[n] == '.')
                {
                    has_only_digits = true;
                }
            }
        }
        //------------------------------
        if (has_only_digits == true)
        {
            value = stod(line.substr(0, found), NULL);
            number[i] = value;
            i++;
            found = line.find_first_of(";");
        }
        else
        {
            name[j] = line.substr(0, found);
            j++;
            found = line.find_first_of(";");
        }

    } while (found == string::npos);
}
```

```
/*
title: Tube Calorimeter Simulation Platform
author: Andrea Bigi
date: 10/2015
*/

#define _USE_MATH_DEFINES
#include <cmath>
#include "CoolProp.h"
#include "HumidAirProp.h"
#include <iostream>
#include <sstream>
#include <stdlib.h>
#include <fstream>
#include <string>
#include <cstddef>


using namespace CoolProp;
using namespace std;

//list of functions
void GetInput(string&, string&, string&, double&, double&, double&, double&, double&, string&, string&, string&, double&,
                string&, string&, string&, double&, double&, double&, double&, double&, double&, double&, double&, double&,
                double&, double&, double&, double&, double&, double&);
void TubeCalorimeter(string, string, double, double, double, double, double, string, string, string,
                double, string, string, string, double, double, double, double, double, double, double, double, double,
                double, double, double, double, double);
void TestUnit(string, string, double, double, double, double, double, string, string, string,
                double, string, string, string, double, double, double, double, double, double, double, double, double,
                double, double, double, double, double);


int main()
{   /* description
    Main solver
    */
    //author: Andrea Bigi
    //date: 07/2016
    //---------------------------------------------------------------

    //// variable definition

    //mode
    string mode;

    //fluid definition
    string ent_ref;     //enthalpy reference (IIR, ASHRAE, NBP, DEF)
    string fluid;       //base fluid
    double A0;    //correlation coefficient for refrigerant-oil mixture bubble temperature (Sawant correlation), 1/K
    double A1;    //correlation coefficient for refrigerant-oil mixture bubble temperature (Sawant correlation), 1/K
    double A2;    //correlation coefficient for refrigerant-oil mixture bubble temperature (Sawant correlation), 1/K
    double a0;    //correlation coefficient for refrigerant-oil mixture bubble temperature (Sawant correlation)
    double b0;    //correlation coefficient for refrigerant-oil mixture bubble temperature (Sawant correlation)
    string oil;         //oil type (EMKARATE POE RL32-3MAF, ...)
```

```
56      string nano_mater;  //nanoparticle material
57      string nano_shape;  //nanoparticle shape
58      double D_nano;      //nanoparticle equivalent diameter, nm
59
60
61
62      //geometry of tube
63      string tube_type;   //tube type (smooth, microfin,...)
64      string tube_mater;  //tube material (copper, steel, ...)
65      string orientation; //tube orientation (horizontal, vertical, inclined)
66      double tube_roughness;  //tube roughness, m
67      double nSeg;        //number of segments, -
68      double nRad;        //number of radial segments of the laminar sublayers, -
69      double lengthTube;  //heat transfer tube length, m
70      double DPlengthTube;//pressure drop tube length, m
71      double Dh;          //hydraulic diameter, m
72      double Do;          //tube outside diameter, m
73      double Dr;          //maximum inside diameter of micro-fin tube, m
74      double tw;          //tube wall thickness at fin root, m
75      double heightFin;   //fin height, m
76      double pitchFin;    //fin pitch, m
77      double beta_deg;    //helix angle, deg
78      double alpha_deg;   //fin top angle, deg
79      double nFins;       //number of fins, -
80      double Sp;          //perimeter of one fin and channel, m
81
82
83      // flow
84
85      ////BEGIN INPUT DATA ACQUISITION
86      GetInput(mode, ent_ref, fluid, A0, A1, A2, a0, b0, oil, nano_mater, nano_shape,
        D_nano,
87          tube_type, tube_mater, orientation, tube_roughness, nSeg, nRad, lengthTube,
            DPlengthTube,
88          Dh, Do, Dr, tw, heightFin, pitchFin, beta_deg, alpha_deg, nFins, Sp);
89
90
91      if (mode == "tube calorimeter")
92      {
93          TubeCalorimeter(ent_ref, fluid, A0, A1, A2, a0, b0, oil, nano_mater, nano_shape,
             D_nano,
94              tube_type, tube_mater, orientation, tube_roughness, nSeg, nRad, lengthTube,
                DPlengthTube,
95              Dh, Do, Dr, tw, heightFin, pitchFin, beta_deg, alpha_deg, nFins, Sp);
96      }
97      else if (mode == "test unit")
98      {
99          TestUnit(ent_ref, fluid, A0, A1, A2, a0, b0, oil, nano_mater, nano_shape, D_nano,
100             tube_type, tube_mater, orientation, tube_roughness, nSeg, nRad, lengthTube,
                DPlengthTube,
101             Dh, Do, Dr, tw, heightFin, pitchFin, beta_deg, alpha_deg, nFins, Sp);
102     }
103
104
105     //system("pause");
106     return 0;
107
108 }
```

```
#define _USE_MATH_DEFINES
#include <cmath>
#include "CoolProp.h"
#include "HumidAirProp.h"
#include <iostream>
#include <sstream>
#include <stdlib.h>
#include <fstream>
#include <string>
#include <cstddef>


using namespace CoolProp;
using namespace std;

void CalcProps(string, double, double, double, double, double,
    string, string, string, double, double, double, double, double, double,
    double&, double&, double&, double&, double&, double&, double&, double&, double&,
    double&, double&, double&, double&, double&, double&, double&, double&, double&,
    double&, double&, double&, double&, double&, double&, double&, double&, double&,
    double&);

void Inventory(string, string, double, double, double, double, double, double, double,
double,
    double&, double&, double&, double&, double&, double&, double&, double&, double&);

void Radial(double, int, ofstream&, string, double, double, double, double, double,
string, string, string,
    double, double, double, double, double, double, double, double, double,
    double, double, double, double, double, double, double, double, double,
    double, double, double, double, double, double, double, double, double,
    double, double, double, double, double, double, double, double, double,
    double, double, double, double, string, double, double, double, double, double&,
    double&);

double FrictionFactor(string, string, double, double, double, double, double,
    double, double, double, double, double, double, double);

double RouhaniAxelsson(string, double, double, double, double, double);
double Buongiorno2006(double, int, ofstream&, ofstream&, string, double, double, double,
 double, double, string, string, string,
    double, double, double, double, double, double, double, double, double,
    double, double, double, double, double, double, double, double, double,
    double, double, double, double, double, double, double, double,
    double, double, double, double, double, double, double, double, double, double,
    double, double, double, double, double, double, double, double, double&, double&);

double singlePhase_L(double, double, double, double, double, double, double);
double twoPhaseMomentumDp(string, double, double, double, double, double, double, double
, double, double);
double twoPhaseDp_Choi1999(string, double, double, double, double, double, double,
    double, double, double, double, double, double);
double twoPhaseDp_HuDing2008(string, double, double, double, double, double, double,
double);
double twoPhaseDp_DingHu2009(string, double, double, double, double, double, double,
double);
double twoPhaseHTC_Hamilton2005(string, double, double, double, double, double,
    double, double, double, double, double, double);
double twoPhaseHTC_HuDing2008(string, double, double, double, double, double, double,
    double, double, double, double, double, double, double, double);
double twoPhaseHTC_Zou2010(string, double, double, double, double, double, double,
    double, double, double, double);
double twoPhaseHTC_Sawant2012(string, string, double, double, double, double, double,
double, double, double,
    double, double, double, double, double, double, double, double, double, double);
double PoolHTC_Kedzierski2012(string, string, double, double, double, double, double,
```

```
58        double, double, double, double, double, double, double, double);
59
60    double LocalOilMassFraction(double, double);
61    double OilDensity(double);
62    double NanoVolumeFraction(string, double, double);
63    double NanoDensity(string);
64    double OilMixtureBubbleTemp(double, double, string);
65    double OilMixtureSpecificHeat(double, double, double);
66    double OilSpecificEnthalpy(double);
67    double OilMixtureEnthalpy(double, double, double, double, double, string);
68    double OilMixtureDensity(double, double, double);
69    double OilMixtureSurfTension(double, double, double);
70    double NanoOilMixtureDensity(string, double, double, double, double);
71    double NanoOilMixtureSurfTension(double, double, double);
72
73
74    void TubeCalorimeter(string ent_ref, string fluid, double A0, double A1, double A2,
      double a0, double b0,
75        string oil, string nano_mater, string nano_shape, double D_nano, string tube_type,
          string tube_mater,
76        string orientation, double tube_roughness, double nSeg, double nRad, double
          lengthTube, double DPlengthTube, double Dh, double Do, double Dr,
77        double tw, double heightFin, double pitchFin, double beta_deg, double alpha_deg,
          double nFins, double Sp)
78    {
79        //use: calculate radial nanoparticle distribution
80        //
81        //GEOMETRY SOURCES:
82        //source: Choi, Kedzierski, Domanski - 1999 -
83        //          A Generalized Pressure Drop Correlation for Evaporation and Condensation
84        //          of Alternative Refrigerants in Smooth and Micro-fin Tubes NISTIR 6333
85        //
86        //          Choi, Kedzierski, Domanski -
87        //          A Generalized Pressure Drop Correlation for Evaporation and Condensation
88        //          of Alternative Refrigerants in Smooth and Micro-fin Tubes
89        //          IIF - IIR - Commission B1- Paderbom, Germany - 2001/5
90        //
91        //
92        //author: Andrea Bigi
93        //date: 06/2017
94        //-----------------------------------------------------------------
95
96
97        //local variables
98        double value;
99        double number[30];
100       int count;
101       int i;
102       string line;
103
104       //preheater inlet
105       double p_pre_in;   //preheater inlet pressure, kPa
106       double t_pre_in;   //preheater inlet temperature, C
107       double Q_pre;      //preheater capacity, W
108       double h_pre_f;    //preheater pure fluid inlet enthalpy, kJ/kg
109       double h_pre_in;   //preheater fluid mixture inlet enthalpy, kJ/kg
110
111       //inlet and outlet properties
112       double OMF;         //oil mass fraction in base fluid, -
113       double NMF;         //nanoparticle mass fraction in oil, -
114       double m_dot_fluid; //fluid mass flow rate, kg/s
115       double p_in;        //inlet pressure, kPa
116       double h_in;        //inlet enthalpy, kJ/kg
117       double Q;           //test section capacity, W
118       double t_wall;      //wall temperature, C
119
```

```
120       double t_in;         //inlet temperature, C
121       double x_in;         //inlet quality, -
122       double rho_in;       //inlet density, kg/m3
123       double v_in;         //inlet specific volume, m3/kg
124       double mu_in;        //inlet dynamic viscosity, N-s/m2, kg/m-s, Pa-s
125       double cp_in;        //inlet specific heat, kJ/kg-K
126       double k_in;         //inlet thermal conductivity, W/m-K
127       double sigma_in;     //inlet surface tension, N/m
128
129       double p_out;        //outlet pressure, kPa
130       double t_out;        //outlet temperature, C
131       double h_out;        //outlet enthalpy, kJ/kg
132       double x_out;        //outlet quality, -
133       double rho_out;      //outlet density, kg/m3
134       double v_out;        //outlet specific volume, m3/kg
135       double mu_out;       //outlet viscosity, kg/m-s
136       double cp_out;       //inlet specific heat, kJ/kg-K
137       double k_out;        //inlet thermal conductivity, W/m-K
138       double sigma_out;    //outlet surface tension, N/m
139
140       //saturated properties
141       double t_sat_f;      //saturated liquid temperature, C
142       double h_f;          //saturated liquid enthalpy, kJ/kg
143       double rho_f;        //saturated liquid density, kg/m3
144       double v_f;          //saturated liquid specific volume, m3/kg
145       double cp_f;         //saturated liquid specific heat, kJ/kg-K
146       double mu_f;         //saturated liquid dynamic viscosity, N-s/m2, kg/m-s, Pa-s
147       double k_f;          //saturated liquid thermal conductivity, W/m-C
148       double sigma_f;      //saturated liquid surface tension, N/m
149
150       double t_sat_g;      //saturated vapor temperature, C
151       double h_g;          //saturated vapor enthalpy, kJ/kg
152       double rho_g;        //saturated vapor density, kg/m3
153       double v_g;          //saturated vapor specific volume, m3/kg
154       double cp_g;         //saturated vapor specific heat, kJ/kg-K
155       double mu_g;         //saturated vapor dynamic viscosity, N-s/m2, kg/m-s, Pa-s
156       double k_g;          //saturated vapor thermal conductivity, W/m-C
157       double sigma_g;      //saturated vapor surface tension, N/m
158
159       double h_fg;         //vaporization enthalpy, kJ/kg
160       double dh_x;         //delta enthalpy, kJ/kg
161       double t_sat;        //saturation temperature, C
162
163       //auxiliary properties   (for printing purposes, later they might be expanded and
          substitute the "saturated properties")
164       double rho_f_in;     //inlet saturated liquid density, kg/m3
165       double rho_f_out;    //outlet saturated liquid density, kg/m3
166       double v_f_in;       //inlet saturated liquid specific volume, kg/m3
167       double v_f_out;      //outlet saturated liquid specific volume, kg/m3
168       double sigma_f_in;   //inlet saturated liquid surface tension, N/m
169       double sigma_f_out; //outlet saturated liquid surface tension, N/m
170
171       double x_out_p;      //predicted outlet quality
172       double p_out_p;      //predicted outlet pressure, kPa
173       double h_out_p;      //predicted outlet enthalpy, kJ/kg
174       double t_out_p;      //predicted outlet temperature, C
175       double cp_f_p;       //predicted saturated liquid specific heat, kJ/kg-K
176       double h_f_p;        //predicted saturated liquid enthalpy, kJ/kg
177       double cp_g_p;       //predicted saturated vapor specific heat, kJ/kg-K
178       double h_g_p;        //predicted saturated vapor enthalpy, kJ/kg
179       double t_sat_g_p;    //predicted saturated vapor temperature, C
180       double h_fg_p;       //predicted vaporization enthalpy, kJ/kg
181
182       //oil properties
183       double w_local;      //local oil mass fraction
184       double w_local_p;    //local OMF, used in pressure routine
```

```
185       double h_oil;       //oil enthalpy, kJ/kg
186
187       //correlations
188       double QSeg;        //segment heat capacity, W
189       double q_flux;      //tube heat flux, W/m2
190       double G_flux;      //mass flux, kg/m2-s
191
192       double htc;         //heat transfer coefficient, W/m2-K
193       double htc_corr;    //heat transfer coefficient from correlation, W/m2-K
194       double htc_corr_Ham;
195       double htc_corr_Hu;
196       double htc_corr_Zuo;
197       double htc_corr_Saw;
198       double htc_corr_Buong;
199       double htc_radial;
200       double htc_nb;      //nucleate boiling heat transfer coefficient, W/m2-K
201       double DpSeg_mom;   //segment momentum pressure drop, kPa
202       double DpSeg_fric;  //segment frictional pressure drop, kPa
203       double DpSeg_fric2008;  //segment frictional pressure drop, kPa
204       double DpSeg_fric2009;
205       double DpSeg_grav;  //segment gravitational pressure drop, kPa
206       double DpSeg_tot;   //segment total pressure drop, kPa
207       double DpSeg_tot_Choi;
208       double DpSeg_tot_Hu;
209       double DpSeg_Lsp;   //segment liquid single phase pressure drop, kPa
210       double ff;      //friction factor
211
212       //heat transfer
213       //double alpha; //void fraction
214       double Xtt;     //Martinelli parameter
215       //double a, m, cc, ex;  //correlation coefficients
216       double delta_f; //liquid thickness, m
217       double u_f;     //liquid layer velocity, m/s
218       double Pr_b;
219       double Re_b;
220       //double htc_TPmix; //two-phase mixture heat transfer coefficient, W/m2-K
221       double htc_Liq;     //liquid heat transfer coefficient, W/m2-K
222
223       double N_bt;    //ratio of Brownian and thermophoretic diffusivities
224
225       //geometry variables
226
227       //geometry of tube
228       double lengthSeg;   //heat transfer segment length, m
229       double DPlengthSeg; //pressure drop segment length, m
230       double baseFin;     //fin base, m
231       double sideFin;     //fin hypotenuse, m
232       double sigma_deg;   //fin base angle, deg
233       double De;          //equivalent diameter, m
234       double Af;          //cross sectional area associated with one fin, m2
235       double SectA;       //section area, m2
236       double SurfA;       //tube surface area, m2
237       double SurfASeg;    //segment surface area, m2
238
239       //inventory variables
240       double epsilon;     //void fraction
241       double VSeg;        //segment volume, m3
242       double m_ref_f_Seg; //mass of liquid refrigerant inside the segment, kg
243       double m_ref_g_Seg; //mass of vapor refrigerant inside the segment, kg
244       double m_ref_Seg;   //total mass of refrigerant inside the segment, kg
245       double m_oil_Seg;   //mass of oil inside the segment, kg
246       double m_nanooil_Seg;   //mass of oil inside the segment, kg
247       double m_np_Seg;    //mass of nanoparticles inside the segment, kg
248       double n_np_Seg;    //number of nanoparticles inside the segment, -
249       double V_nano;      //nanopartcile volume, m3
250       double rho_nano;    //nanoparticle density, kg/m3
```

```
251         double phi;         //nanoparticle volume fraction in oil
252
253         double Re;  //Reynolds number, -
254         double Pr;   //Prandtl number, -
255
256         int index_Nbt = 0;
257
258         double err;         //error difference
259
260         //output variables
261
262         //flow
263
264         set_reference_stateS(fluid, ent_ref);   //set enthalpy reference for properties
            calculation
265
266         ////OPEN OUTPUT FILE
267         ofstream output_file;
268         output_file.open("out_tc.csv");
269         //writing legend
270         output_file << "OMF, NMF, m_dot [kg/s], G_flux [kg/m2-s], q_flux [W/m2], x_in [-],
            x_out [-],"
271             "h_in [kJ/kg], h_out [kJ/kg], p_in [kPa], p_out [kPa], t_in [C], t_out [C],
                h_fg [kJ/kg],"
272             "rho_f [kg/m3], v_f [m3/kg], cp_f [kJ/kg-K], mu_f [N-s/m2], k_f [W/m-K],
                sigma_f [N/m],"
273             "rho_g [kg/m3], v_g [m3/kg], cp_g [kJ/kg-K], mu_g [N-s/m2], k_g [W/m-K],
                sigma_g [N/m],"
274             "DpSeg_tot_Choi [kPa], DpSeg_tot_Hu [kPa], DpSeg_fric2008 [kPa], DpSeg_fric2009
                [kPa], DpSeg_mom [kPa],"
275             "htc [W/m2-K], htc_corr_Ham [W/m2-K], htc_corr_Hu [W/m2-K], htc_corr_Zuo
                [W/m2-K], htc_corr_Saw [W/m2-K], htc_corr_Buong [W/m2-K],"
276             "m_ref_f_Seg [kg], m_ref_g_Seg [kg], m_oil_Seg [kg], m_np_Seg [kg], n_np_Seg,
                t_wall [C]" << endl;
277
278         ofstream radial_output_file;
279         radial_output_file.open("out_tc_radial.csv");
280
281         ofstream Buongiorno_output_file;
282         Buongiorno_output_file.open("out_Buongiorno.csv");
283         //writing legend
284         Buongiorno_output_file << "G_flux [kg/m2-s], q_flux [W/m2], phi, N_bt, phi_v, ff,
            tau_w [Pa],"
285             "D_B [m2/s], grad_T [K/m], V_T [m2/s], delta_v [m], u_ave [m/s], t_wall [C],
                Re_b, Pr_b, Pr_v, Nu_b, htc,"
286             "rho_b [kg/m3], cp_b [J/kg-K], mu_b [N-s/m2], k_b [W/m-K],"
287             "rho_v [kg/m3], cp_v [J/kg-K], mu_v [N-s/m2], k_v [W/m-K]" << endl;
288
289         ofstream Buongiorno_radial_output_file;
290         Buongiorno_radial_output_file.open("out_Buongiorno_radial.csv");
291
292
293         ////OPEN INPUT FILE WITH TEST DATA
294         ifstream input_file_data;
295         input_file_data.open("in.csv");
296         getline(input_file_data, line);  //read first line: file legend
297         while (getline(input_file_data, line))
298         {
299             istringstream ss(line);
300             string token;
301             vector<string> tokens;
302
303             while (getline(ss, token, ','))
304                 tokens.push_back(move(token));
305
306             m_dot_fluid = stof(tokens[0]);
```

```
307         Q_pre = stof(tokens[1]);
308         p_pre_in = stof(tokens[2]);
309         t_pre_in = stof(tokens[3]);
310         Q = stof(tokens[4]);
311         p_in = stof(tokens[5]);
312         h_in = stof(tokens[6]);
313         t_wall = stof(tokens[7]);
314         OMF = stof(tokens[8]);
315         NMF = stof(tokens[9]);
316
317         ////////////////////////////////////////////////////////
318         ////geometry calculations
319         if (tube_type == "smooth")
320         {
321             SectA = M_PI*pow(Dh, 2) / 4;
322             SurfA = M_PI*Dh*lengthTube;
323
324             q_flux = Q / SurfA;
325             G_flux = m_dot_fluid / SectA;
326
327             ////segmentation
328             lengthSeg = lengthTube / nSeg;
329             DPlengthSeg = DPlengthTube / nSeg;
330             SurfASeg = M_PI*Dh*lengthSeg;
331             QSeg = q_flux * SurfASeg;    //or also: QSeg = Q / nSeg;
332
333             De = Dh;    //for the case of a smooth tube (De is used in the radial
                subroutine)
334         }
335         else if (tube_type == "microfin")
336         {
337             sigma_deg = 180 - 90 - alpha_deg / 2;
338             baseFin = 2 * heightFin / tan(sigma_deg * M_PI / 180);
339             sideFin = heightFin / sin(sigma_deg * M_PI / 180);
340             Af = (baseFin * heightFin / 2) + tw * (baseFin + Sp / cos(beta_deg * M_PI /
                180) - 2 * sideFin);
341
342             SectA = M_PI / 4 * pow(Do, 2) - nFins * Af;
343             De = sqrt(4 * SectA / M_PI);
344             Dh = 4 * SectA * cos(beta_deg * M_PI / 180) / (nFins * Sp);
345
346             SurfA = nFins * Sp / cos(beta_deg * M_PI / 180) * lengthTube;
347
348             q_flux = Q / SurfA;
349             G_flux = m_dot_fluid / SectA;
350
351             ////segmentation
352             lengthSeg = lengthTube / nSeg;
353             DPlengthSeg = DPlengthTube / nSeg;
354             SurfASeg = nFins * Sp / cos(beta_deg * M_PI / 180) * lengthSeg;
355             QSeg = q_flux * SurfASeg;    //or also: QSeg = Q / nSeg;
356         }
357
358         VSeg = M_PI / 4 * pow(Dh, 2)* lengthSeg;    //should this be the actual section
            area, rather than the area based on Dh?
359
360         ////////////////////////////////////////////////////////
361         ////calculations
362
363         ////preheater calculations (all liquid)
364         if (Q_pre > 0)
365         {
366             h_oil = OilSpecificEnthalpy(t_pre_in);
367             h_pre_f = PropsSI("H", "P", p_pre_in * 1000, "T", t_pre_in + 273.15, fluid)
                / 1000;
368             h_pre_in = (1 - OMF) * h_pre_f + OMF * h_oil;
```

```
369
370             //calculate correct test section inlet enthalpy, given the preheater capacity
371             h_in = h_pre_in + Q_pre / m_dot_fluid / 1000;
372         }
373
374
375         ////segment by segment analysis
376         for (i = 1; i <= nSeg; i++)
377         {
378
379             ////initialization
380             htc_corr_Ham = 0;
381             htc_corr_Hu = 0;
382             htc_corr_Zuo = 0;
383             htc_corr_Saw = 0;
384             htc_corr_Buong = 0;
385             DpSeg_mom = 0;
386             DpSeg_fric = 0;
387             DpSeg_fric2008 = 0;
388             DpSeg_fric2009 = 0;
389             DpSeg_grav = 0;
390             DpSeg_tot = 0;
391             DpSeg_tot_Choi = 0;
392             DpSeg_tot_Hu = 0;
393             DpSeg_Lsp = 0;
394
395             ////calculation of inlet fluid properties
396             CalcProps(fluid, A0, A1, A2, a0, b0, oil, nano_mater, nano_shape,
397                 D_nano, OMF, NMF, m_dot_fluid, p_in, h_in,
398                 t_in, x_in, rho_in, v_in, cp_in, mu_in, k_in, sigma_in,
399                 t_sat_f, h_f, rho_f, v_f, cp_f, mu_f, k_f, sigma_f,
400                 t_sat_g, h_g, rho_g, v_g, cp_g, mu_g, k_g, sigma_g, h_fg, dh_x, t_sat,
                    w_local);
401
402             ////segment inventory, based on the hydraulic diameter
403             epsilon = RouhaniAxelsson(fluid, G_flux, sigma_f, rho_g, rho_f, x_in);
404             Inventory(fluid, nano_mater, D_nano, SectA, lengthSeg, epsilon, w_local, NMF
                , rho_g, rho_f,
405                 VSeg, m_ref_f_Seg, m_ref_g_Seg, m_ref_Seg, m_nanooil_Seg, m_np_Seg,
                    m_oil_Seg, V_nano, n_np_Seg);
406
407             ////Friction factor calculation
408             ff = FrictionFactor(tube_type, tube_mater, tube_roughness, nFins, heightFin,
                 beta_deg, alpha_deg,
409                 G_flux, m_dot_fluid, SectA, De, Dr, rho_f, mu_f);
410
411             ////heat transfer coefficient (theoretical formulation)
412             htc = q_flux / (t_wall - t_sat);
413
414             ////heat transfer correlation
415             //two-phase flow
416             if (x_in > 0 && x_in < 1)
417             {
418                 //in theory some of these correlations are designed for cases where OMF
                    > 0
419                 htc_corr_Ham = twoPhaseHTC_Hamilton2005(fluid, Dh, OMF, G_flux, q_flux,
                    t_sat, p_in, x_in, mu_f, cp_f, k_f, h_fg);
420                 htc_corr_Hu = twoPhaseHTC_HuDing2008(fluid, Dr, heightFin, pitchFin,
                    beta_deg, G_flux, q_flux, t_sat,
421                     p_in, x_in, rho_f, cp_f, mu_f, k_f, h_fg);
422                 htc_corr_Zuo = twoPhaseHTC_Zou2010(fluid, Dh, G_flux, q_flux, t_sat,
                    p_in, x_in, rho_f, cp_f, mu_f, k_f);
423
424                 //pass the local oil mass fraction (w_local) instead of the oil mass
                    fraction (OMF)
425                 htc_corr_Saw = twoPhaseHTC_Sawant2012(fluid, nano_mater, D_nano, w_local
```

```
                , NMF, n_np_Seg, SurfASeg, Dh, G_flux, q_flux,
426                 t_sat, t_wall, p_in, x_in, rho_f, cp_f, mu_f, k_f, sigma_f, h_fg);
427
428             }
429             //single phase flow - liquid
430             if (x_in <= 0)
431             {
432                 //need more correlations for single phase
433
434                 //Dittus-Boelter correlation
435                 Re = G_flux * Dh / mu_f;
436                 Pr = mu_f*cp_f * 1000 / k_f;
437                 htc = 0.023 * (k_f / Dh) * pow(Re, 0.8) * pow(Pr, 0.4);
438
439                 if (NMF > 0)
440                 {
441                     htc_corr_Buong = Buongiorno2006(nRad, index_Nbt,
                    Buongiorno_output_file, Buongiorno_radial_output_file,
442                         fluid, A0, A1, A2, a0, b0, oil, nano_mater, nano_shape, ff,
443                         D_nano, OMF, NMF, n_np_Seg, m_dot_fluid, p_in, h_in,
444                         t_in, x_in, rho_in, v_in, cp_in, mu_in, k_in, sigma_in,
445                         t_sat_f, h_f, rho_f, v_f, cp_f, mu_f, k_f, sigma_f,
446                         t_sat_g, h_g, rho_g, v_g, cp_g, mu_g, k_g, sigma_g, h_fg,
447                         dh_x, t_sat, w_local, Dh, De, SectA, lengthSeg, t_wall, G_flux,
                        q_flux, delta_f,
448                         N_bt, phi);
449
450                     t_wall = t_in + q_flux / htc_corr_Buong;    //Newton's law of
                    cooling (to calculate t_wall, it should be better to use the
                    segment average temperature (t_in+t_out)/2, instead of t_in
451                 }
452             }
453
454             ////LVPCP pag.611, eq.12.36b
455             //double Re_le;
456             //double Pr_l;
457             //double h_le;
458             //double gamma;
459             //double x_sup;
460
461             //Re_le = G_flux*Dh/mu_f;
462             //Pr_l = mu_f*cp_f/k_f;
463             //h_le = 0.023*(k_f/Dh)*pow(Re_le, 0.8)*pow(Pr_l, 0.4);
464             //gamma = pow(rho_g / rho_f, 0.56)*pow(mu_f / mu_g,
                0.11)*pow((q_flux*k_f*h_fg*rho_g) / (98 * sigma_f*(t_sat+273.15)*pow(h_le,
                2)), 1.11);
465             //x_sup = gamma/(1+gamma);
466
467             ////calculate outlet enthalpy
468             //h_out = h_in + (htc*(t_wall - t_sat)*SurfASeg / m_dot_fluid)/1000;
469             h_out = h_in + QSeg / 1000 / m_dot_fluid;
470
471             ////calculate outlet quality
472             if (OMF == 0)
473             {
474                 x_out = (h_out - h_f) / h_fg;
475             }
476             else
477             {
478                 //x_out = OilMixtureEnthalpy(t_sat, OMF, x_in, p_in, h_out-h_in,
                fluid);   //not sure it is working correctly, needs to be verified
479                 //x_out = (h_out - h_f) / h_fg;
480                 x_out = PropsSI("Q", "P", p_in * 1000, "H", h_out * 1000, fluid);
481                 x_out = x_out * (1 - OMF);
482             }
483
```

```
484             //calcDP();
485
486             ////calculate outlet pressure
487             rho_f_in = rho_f;
488             sigma_f_in = sigma_f;
489
490             //first estimate
491             p_out_p = p_in;
492             x_out_p = x_out;
493             count = 0;
494             do
495             {
496                 count++;   //this is added in order to cycle at least one time after
                    the first estimate
497
498                 //calculate outlet properties (first time based on inlet pressure)
499                 rho_out = PropsSI("D", "P", p_out_p * 1000, "H", h_out * 1000, fluid);
500                 v_out = 1 / rho_out;
501
502                 //calculate saturated outlet properties
503                 rho_f_out = PropsSI("D", "P", p_out_p * 1000, "Q", 0, fluid);
504                 sigma_f_out = PropsSI("I", "P", p_out_p * 1000, "Q", 0, fluid);
505                 if ((OMF > 0) && (NMF == 0))
506                 {
507                     sigma_f_out = OilMixtureSurfTension(t_sat, w_local, sigma_f_out);
508                     rho_f_out = OilMixtureDensity(t_sat, w_local, rho_f_out);
509                     v_f_out = 1 / rho_f_out;
510                     v_out = (1 - x_out_p)*v_f_out + x_out_p*v_g;
511                 }
512                 if ((OMF > 0) && (NMF > 0))
513                 {
514                     sigma_f_out = NanoOilMixtureSurfTension(t_sat, w_local, sigma_f_out);
515                     rho_f_out = NanoOilMixtureDensity(nano_mater, NMF, t_sat, w_local,
                        rho_f_out);
516                     v_f_out = 1 / rho_f_out;
517                     v_out = (1 - x_out_p)*v_f_out + x_out_p*v_g;
518                 }
519
520
521                 if (x_in > 0 && x_in < 1)
522                 {
523                     //pressure drop correlations (gravitational, momentum, frictional)
524                     if (orientation == "horizontal")
525                     {
526                         DpSeg_grav = 0;
527                     }
528                     else if (orientation == "vertical")
529                     {
530                         DpSeg_grav = -9.81*rho_f_in*lengthSeg;  ///verify this!!!!
531                     }
532                     DpSeg_mom = twoPhaseMomentumDp(fluid, G_flux, sigma_f_in,
                        sigma_f_out, p_in, p_out_p, rho_f_in, rho_f_out, x_in, x_out_p);
533                     DpSeg_fric = twoPhaseDp_Choi1999(fluid, OMF, Dh, DPlengthSeg, G_flux
                        , m_dot_fluid, p_in, v_in, v_out, x_in, x_out_p, mu_f, h_fg);
534                     //only for Choi1999, the Dp_fric calculated also accounts for the
                        momentum  pressure drop
535                     DpSeg_tot_Choi = DpSeg_grav + DpSeg_fric;
536
537                     DpSeg_fric2008 = twoPhaseDp_HuDing2008(fluid, Dh, DPlengthSeg,
                        G_flux, p_in, x_in, rho_f, mu_f);
538                     DpSeg_fric2009 = twoPhaseDp_DingHu2009(fluid, Dh, DPlengthSeg,
                        G_flux, p_in, x_in, rho_f, mu_f);
539                     DpSeg_tot_Hu = DpSeg_grav + DpSeg_mom + DpSeg_fric2008;
540
541                     p_out = p_in - DpSeg_tot_Choi;
542                 }
```

```
543                 //single phase flow - liquid
544                 if (x_in <= 0)
545                 {
546                     DpSeg_Lsp = singlePhase_L(G_flux, m_dot_fluid, SectA, Dh, lengthSeg,
                         rho_f, mu_f);
547
548                     p_out = p_in - DpSeg_Lsp;
549                 }
550
551                 //calculate outlet quality and enthalpy based on newly calculated
                    outlet pressure
552                 t_out_p = PropsSI("T", "P", p_out * 1000, "H", h_out * 1000, fluid) -
                    273.15;
553                 x_out_p = PropsSI("Q", "P", p_out * 1000, "H", h_out * 1000, fluid);
554                 //h_out_p = PropsSI("H", "P", p_out * 1000, "Q", x_out_p, fluid) / 1000;
555
556                 if (OMF > 0)
557                 {
558                     cp_f_p = PropsSI("C", "P", p_out * 1000, "Q", 0, fluid) / 1000;
559                     h_f_p = PropsSI("H", "P", p_out * 1000, "Q", 0, fluid) / 1000;
560
561                     cp_g_p = PropsSI("C", "P", p_out * 1000, "Q", 1, fluid) / 1000;
562                     h_g_p = PropsSI("H", "P", p_out * 1000, "Q", 1, fluid) / 1000;
563                     t_sat_g_p = PropsSI("T", "P", p_out * 1000, "Q", 1, fluid) - 273.15;
564                     h_fg_p = h_g_p - h_f_p;
565
566                     w_local_p = LocalOilMassFraction(OMF, x_out_p);
567                     t_out_p = OilMixtureBubbleTemp(p_out * 1000, w_local_p, fluid);
568                     x_out_p = x_out_p * (1 - OMF);
569                     cp_f_p = OilMixtureSpecificHeat(t_out_p, w_local_p, cp_f_p);
570
571                     ////see Thome(1995)
572                     h_fg_p = h_fg_p*(x_out_p - 0) + (1 - x_out_p)*cp_f_p*(t_out_p -
                        t_sat_g_p) + x_out_p*cp_g_p*(t_out_p - t_sat_g_p);
573                     h_out_p = h_in - dh_x + h_fg_p;
574                 }
575
576                 //error calculation
577                 //err = (h_out - h_out_p) / h_out_p;  //in case h_out was not certain,
                    this could be another convergence condition
578                 err = (p_out - p_out_p) / p_out_p;
579
580                 if (abs(err) > 0.001 || count == 1)
581                 {
582                     p_out_p = p_out;
583                 }
584                 else
585                 {
586                     x_out = x_out_p;
587                     t_out = t_out_p;
588                 }
589
590             } while (abs(err) > 0.001 || count == 1);
591
592
593             //radial analysis for nanoparticle distribution
594             if (NMF > 0 && nRad > 0)
595             {
596                 Radial(nRad, index_Nbt, radial_output_file,
597                     fluid, A0, A1, A2, a0, b0, oil, nano_mater, nano_shape, D_nano, ff,
598                     OMF, NMF, n_np_Seg, m_dot_fluid, p_in, h_in,
599                     t_in, x_in, rho_in, v_in, cp_in, mu_in, k_in, sigma_in,
600                     t_sat_f, h_f, rho_f, v_f, cp_f, mu_f, k_f, sigma_f,
601                     t_sat_g, h_g, rho_g, v_g, cp_g, mu_g, k_g, sigma_g, h_fg, dh_x,
                        t_sat, w_local,
602                     Dh, De, SectA, lengthSeg, orientation, t_wall, G_flux, q_flux,
```

```
                        DpSeg_tot_Choi, delta_f, htc_radial);
603             }
604
605
606
607             ////WRITE ON OUTPUT FILE
608             output_file << OMF << "," << NMF << "," << m_dot_fluid << "," << G_flux <<
                "," << q_flux << "," << x_in << "," << x_out << ","
609                 << h_in << "," << h_out << "," << p_in << "," << p_out << "," << t_in <<
                     "," << t_out << "," << h_fg << ","
610                 << rho_f << "," << v_f << "," << cp_f << "," << mu_f << "," << k_f <<
                    "," << sigma_f << ","
611                 << rho_g << "," << v_g << "," << cp_g << "," << mu_g << "," << k_g <<
                    "," << sigma_g << ","
612                 << DpSeg_tot_Choi << "," << DpSeg_tot_Hu << "," << DpSeg_fric2008 << ","
                     << DpSeg_fric2009 << "," << DpSeg_mom << ","
613                 << htc << "," << htc_corr_Ham << "," << htc_corr_Hu << "," <<
                    htc_corr_Zuo << "," << htc_corr_Saw << "," << htc_corr_Buong << ","
614                 << m_ref_f_Seg << "," << m_ref_g_Seg << "," << m_nanooil_Seg << "," <<
                    m_np_Seg << "," << n_np_Seg << "," << t_wall << endl;
615
616             ////pass segment output variables as input to next segment
617             p_in = p_out;
618             h_in = h_out;
619
620         }  //end of segment by segment analysis
621
622     }
623
624     input_file_data.close();
625     output_file.close();
626     radial_output_file.close();
627     Buongiorno_output_file.close();
628     Buongiorno_radial_output_file.close();
629
630 }
```

```
#define _USE_MATH_DEFINES
#include <cmath>
#include "CoolProp.h"
#include "HumidAirProp.h"
#include <iostream>
#include <sstream>
#include <stdlib.h>
#include <fstream>
#include <string>
#include <cstddef>


using namespace CoolProp;
using namespace std;

void CalcProps(string, double, double, double, double, double,
    string, string, string, double, double, double, double, double, double,
    double&, double&, double&, double&, double&, double&, double&, double&, double&,
    double&, double&, double&, double&, double&, double&, double&, double&, double&,
    double&, double&, double&, double&, double&, double&, double&, double&, double&,
    double&);

void Inventory(string, string, double, double, double, double, double, double, double,
double,
    double&, double&, double&, double&, double&, double&, double&, double&, double&);

void Radial(double, int, ofstream&, string, double, double, double, double, double,
string, string, string,
    double, double, double, double, double, double, double, double, double,
    double, double, double, double, double, double, double, double, double,
    double, double, double, double, double, double, double, double, double,
    double, double, double, double, double, double, double, double, double,
    double, double, double, double, string, double, double, double, double, double&,
    double&);

double FrictionFactor(string, string, double, double, double, double, double,
    double, double, double, double, double, double, double);

double RouhaniAxelsson(string, double, double, double, double, double);
double Buongiorno2006(double, int, ofstream&, ofstream&, string, double, double, double,
 double, double, string, string, string,
    double, double, double, double, double, double, double, double, double,
    double, double, double, double, double, double, double, double, double,
    double, double, double, double, double, double, double, double,
    double, double, double, double, double, double, double, double, double, double,
    double, double, double, double, double, double, double, double, double&, double&);

double twoPhaseHTC_Hamilton2005(string, double, double, double, double, double,
    double, double, double, double, double, double);
double twoPhaseHTC_HuDing2008(string, double, double, double, double, double, double,
    double, double, double, double, double, double, double, double);
double twoPhaseHTC_Zou2010(string, double, double, double, double, double, double,
    double, double, double, double);
double twoPhaseHTC_Sawant2012(string, string, double, double, double, double, double,
double, double, double,
    double, double, double, double, double, double, double, double, double, double);

double PoolHTC_ForsterZuber(string, double, double);
double PoolHTC_JensenJackman(string, double, double, double);
double PoolHTC_Kedzierski2003(string, double, double, double, double, double, double);
double PoolHTC_Kedzierski2012(string, string, double, double, double, double, double,
    double, double, double, double, double, double);

double LocalOilMassFraction(double, double);
double OilDensity(double);
double NanoVolumeFraction(string, double, double);
```

```
61  double NanoDensity(string);
62  double NanoConductivty(string, double);
63  double NanoSpecificHeat(string, double);
64  double OilMixtureBubbleTemp(double, double, string);
65  double OilMixtureBubbleTemp_Sawant(double, double, double, string);
66  double OilMixtureSpecificHeat(double, double, double);
67  double OilSpecificEnthalpy(double);
68  double OilMixtureEnthalpy(double, double, double, double, double, string);
69  double OilMixtureDensity(double, double, double);
70  double OilMixtureSurfTension(double, double, double);
71  double NanoOilMixtureDensity(string, double, double, double, double);
72  double NanoOilMixtureDynViscosity(string, double, double, double, double, double);
73  double NanoOilMixtureConductivity(string, double, double, double, double);
74  double NanoOilMixtureSpecificHeat(string, double, double, double, double);
75  double NanoOilMixtureSurfTension(double, double, double);
76  double NanoOilDynViscosity(string, double, double, double);
77  double NanoOilSpecificHeat(string, double, double);
78  double NanoOilConductivity(string, double, double);
79
80
81  void TestUnit(string ent_ref, string fluid, double A0, double A1, double A2, double a0,
    double b0,
82      string oil, string nano_mater, string nano_shape, double D_nano, string tube_type,
        string tube_mater,
83      string orientation, double tube_roughness, double nSeg, double nRad, double
        lengthTube, double DPlengthTube, double Dh, double Do, double Dr,
84      double tw, double heightFin, double pitchFin, double beta_deg, double alpha_deg,
        double nFins, double Sp)
85  {
86      //use: test platform
87      //
88      //
89      //author: Andrea Bigi
90      //date: 07/2017
91      //---------------------------------------------------------------------
92
93
94      //// variable definition
95      double value;
96      double number[30];
97      int count;
98      int i;
99      string line;
100     size_t found;
101
102     //fluid definition
103
104     //preheater inlet
105     double p_pre_in;   //preheater inlet pressure, kPa
106     double t_pre_in;   //preheater inlet temperature, C
107     double Q_pre;      //preheater capacity, W
108     double h_pre_f;    //preheater pure fluid inlet enthalpy, kJ/kg
109     double h_pre_in;   //preheater fluid mixture inlet enthalpy, kJ/kg
110
111     //inlet and outlet properties
112     double OMF;         //oil mass fraction in base fluid, -
113     double NMF;         //nanoparticle mass fraction in oil, -
114     double m_dot_fluid; //fluid mass flow rate, kg/s
115     double p_in;        //inlet pressure, kPa
116     double h_in;        //inlet enthalpy, kJ/kg
117
118     double t_in;        //inlet temperature, C
119     double x_in;        //inlet quality, -
120     double rho_in;      //inlet density, kg/m3
121     double v_in;        //inlet specific volume, m3/kg
122     double mu_in;       //inlet dynamic viscosity, N-s/m2, kg/m-s, Pa-s
```

```
123         double cp_in;       //inlet specific heat, kJ/kg-K
124         double k_in;        //inlet thermal conductivity, W/m-K
125         double sigma_in;    //inlet surface tension, N/m
126
127         double p_out;       //outlet pressure, kPa
128         double t_out;       //outlet temperature, C
129         double h_out;       //outlet enthalpy, kJ/kg
130         double x_out;       //outlet quality, -
131         double rho_out;     //outlet density, kg/m3
132         double v_out;       //outlet specific volume, m3/kg
133         double mu_out;      //outlet viscosity, kg/m-s
134         double cp_out;      //inlet specific heat, kJ/kg-K
135         double k_out;       //inlet thermal conductivity, W/m-K
136         double sigma_out;   //outlet surface tension, N/m
137
138         //saturated properties
139         double t_sat_f;     //saturated liquid temperature, C
140         double h_f;         //saturated liquid enthalpy, kJ/kg
141         double rho_f;       //saturated liquid density, kg/m3
142         double v_f;         //saturated liquid specific volume, m3/kg
143         double cp_f;        //saturated liquid specific heat, kJ/kg-K
144         double mu_f;        //saturated liquid dynamic viscosity, N-s/m2, kg/m-s, Pa-s
145         double k_f;         //saturated liquid thermal conductivity, W/m-C
146         double sigma_f;     //saturated liquid surface tension, N/m
147
148         double t_sat_g;     //saturated vapor temperature, C
149         double h_g;         //saturated vapor enthalpy, kJ/kg
150         double rho_g;       //saturated vapor density, kg/m3
151         double v_g;         //saturated vapor specific volume, m3/kg
152         double cp_g;        //saturated vapor specific heat, kJ/kg-K
153         double mu_g;        //saturated vapor dynamic viscosity, N-s/m2, kg/m-s, Pa-s
154         double k_g;         //saturated vapor thermal conductivity, W/m-C
155         double sigma_g;     //saturated vapor surface tension, N/m
156
157         double h_fg;        //vaporization enthalpy, kJ/kg
158         double dh_x;        //delta enthalpy, kJ/kg
159         double t_sat;       //saturation temperature, C
160
161         //auxiliary properties   (for printing purposes, later they might be expanded and
            substitute the "saturated properties")
162         double rho_f_in;    //inlet saturated liquid density, kg/m3
163         double rho_f_out;   //outlet saturated liquid density, kg/m3
164         double v_f_in;      //inlet saturated liquid specific volume, kg/m3
165         double v_f_out;     //outlet saturated liquid specific volume, kg/m3
166         double sigma_f_in;  //inlet saturated liquid surface tension, N/m
167         double sigma_f_out; //outlet saturated liquid surface tension, N/m
168
169         double x_out_p;     //predicted outlet quality
170         double p_out_p;     //predicted outlet pressure, kPa
171         double h_out_p;     //predicted outlet enthalpy, kJ/kg
172         double t_out_p;     //predicted outlet temperature, C
173         double cp_f_p;      //predicted saturated liquid specific heat, kJ/kg-K
174         double h_f_p;       //predicted saturated liquid enthalpy, kJ/kg
175         double cp_g_p;      //predicted saturated vapor specific heat, kJ/kg-K
176         double h_g_p;       //predicted saturated vapor enthalpy, kJ/kg
177         double t_sat_g_p;   //predicted saturated vapor temperature, C
178         double h_fg_p;      //predicted vaporization enthalpy, kJ/kg
179
180         //oil properties
181         double w_local;     //local oil mass fraction
182         double h_oil;       //oil enthalpy, kJ/kg
183
184         //correlations
185         double Q;           //test section heat capacity, W
186         double QSeg;        //segment heat capacity, W
187         double q_flux;      //tube heat flux, W/m2
```

```
188       double G_flux;      //mass flux, kg/m2-s
189       double t_wall;      //wall temperature, C
190
191
192       //geometry variables
193
194       //geometry of tube
195       double baseFin;     //fin base, m
196       double sideFin;     //fin hypotenuse, m
197       double sigma_deg;   //fin base angle, deg
198       double De;          //equivalent diameter, m
199       double lengthSeg;   //segment length, m
200       double DPlengthSeg; //pressure drop segment length, m
201       double Af;          //cross sectional area associated with one fin, m2
202       double SectA;       //section area, m2
203       double SurfA;       //tube surface area, m2
204       double SurfASeg;    //segment surface area, m2
205
206       //inventory variables
207       double epsilon;     //void fraction
208       double VSeg;        //segment volume, m3
209       double m_ref_f_Seg; //mass of liquid refrigerant inside the segment, kg
210       double m_ref_g_Seg; //mass of vapor refrigerant inside the segment, kg
211       double m_oil_Seg;   //mass of oil inside the segment, kg
212       double m_np_Seg;    //mass of nanoparticles inside the segment, kg
213       double n_np_Seg;    //number of nanoparticles inside the segment, -
214       double V_nano;      //nanopartcile volume, m3
215       double rho_oil;     //oil density, kg/m3
216       double rho_nano;    //nanoparticle density, kg/m3
217       //double k_nano;      //nanoparticle thermal conductivity, W/m-K
218       double phi;         //nanoparticle volume fraction in oil
219
220       double err;         //error difference
221
222       ////
223       double Kn;          //Knudsen number
224       double lambda;      //molecule mean free path, m
225       double alpha;       //thermal diffusivity, m2/s
226       //double tau_w;     //shear stress at the wall, Pa
227       double Pe_r;        //rotational Peclet number, -
228       double C, n;        //friction factor coefficients
229       double tau_p;       //nanoparticle relaxation time, s
230       double Vel_eo;      //turbulent eddies velocity, m/s
231       double Vel_e;       //nanoparticle/fluid slip velocity due to turbulent eddies, m/s
232       double k_B = 1.38064852e-23;    //Boltzmann constant, J/K
233       double D_B;         //Brownian diffusion coefficient, m2/s
234       double V_T;         //thermophoretic diffusion coefficient, m2/s
235       double Vel_g;       //gravity velocity, m/s
236
237       //double beta;      //thermophoresis proportionality factor
238       double gradT;       //temperature gradient, K/m
239       double Vel_t;       //thermophoresis velocity, m/s
240       double Vel_rad;
241       double Vel_Brown;
242
243       double t_inert;
244       double t_Brown;
245       double t_thermoph;
246       double t_grav;
247
248       double delta_u_inert;       //inertial slip velocity, m/s
249       double delta_u_Brown;       //Brownian motion slip velocity, m/s
250       double delta_u_thermoph;    //thermophoresis slip velocity, m/s
251       double delta_u_grav;        //gravitational slip velocity, m/s
252
253       double t_diff;
```

```
254
255       //Buongiorno variables
256       double k_nano;  //nanoparticle thermal conductivity, W/m-K
257       double cp_nano; //nanoparticle specific heat, kJ/kg-K
258       double beta;    //thermophoretic coefficient
259       double Re;      //Reynolds number
260       double tau_w;   //shear stress at the wall, Pa
261       double ff;      //friction factor
262       double u_ave;   //mean axial velocity or bulk velocity, m/s
263       double phi_v_guess;     //volume fraction initial guess
264       double phi_v;   //nanoparticle volume fraction in laminar sublayer
265       double delta_v; //thickness of the laminar sublayer, m
266       double delta_v_plus;    //non-dimensional thickness of the laminar sublayer
267       double N_bt;    //ratio of Brownian and thermophoretic diffusivities
268       double grad_T;  //film temperature gradient, K/m
269
270       double NMF_v;   //nanoparticle mass fraction in laminar sublayer
271       double t_v;     //temperature
272       double mu_v;    //dynamic viscosity in laminar sublayer, Pa*s
273       double rho_v;
274       double cp_v;
275       double k_v;
276       double mu_b;    //dynamic viscosity in turbulent sublayer, Pa*s
277       double rho_b;
278       double cp_b;
279       double k_b;
280       double sigma_b;
281       double Pr_v;
282       double Pr_b;
283       double Re_b;
284       double Nu_b;    //Nusselt bulk
285       double htc_SP;     //single phase heat transfer coefficient, W/m2-C
286       double t_wall_new;
287       double S_p;
288
289       //double Y;     //wall film thickness, m
290
291       double F_rotat;
292       double F_drag;
293       double F_Brown;
294       double F_thermoph;
295       double F_grav;
296       double F_inert;
297       double F_lift;
298       double gamma;   //shear rate
299       double F_drain;
300
301       //double u_v;       //velocity of laminar sublayer, m/s
302       double u_nano;  //velocity of nanoparticle sublayer, m/s
303       double delta_u; //slip velocity: u_v-u_nano, m/s
304       double Re_v;    //Reynolds number based on slip velocity, -
305       double A_p;     //projected area of the body in the direction of flow, m2
306       double C_d;     //drag coefficient
307
308       double C_wl;
309       double F_wl;
310
311       ////////
312
313       double mu_v_Rad[100];   //laminar sublayer viscosity, N-s/m2, kg/m-s, Pa-s
314       double k_v_Rad[100];    //laminar sublayer conductivity, W/m-K
315       double cp_v_Rad[100];   //laminar sublayer specific heat, kJ/kg-K
316       double rho_v_Rad[100];  //laminar sublayer density, kg/m3
317       double phi_v_Rad[100];  //laminar sublayer nanoparticle volume fraction
318       double NMF_v_Rad[100];  //laminar sublayer nanoparticle mass fraction
319       double grad_T_Rad[100]; //laminar sublayer temperature gradient, K/m
```

```
320       double Vol_rate_Rad[100];//laminar sublayer volumetric flow rate per unit wetted
          perimeter, m2/s
321
322       double ff_i;    //interfacial friction factor, -
323       double tau_v[100];//laminar sublayer shear stress, Pa
324       double tau_i;   //interfacial shear stress, Pa
325       double tau_vt;  //laminar-turbulent interface shear stress, Pa
326
327       double q_v[100];//laminar sublayer heat flux, W/m2
328       double q_wall;  //wall heat flux, W/m2
329       double q_i;     //liquid-vapor interface heat flux, W/m2
330       double q_vt;    //laminar-turbulent interface heat flux, W/m2
331
332       double u_v[100];//laminar sublayer  velocity, m/s
333       double u_g;     //gas core velocity, m/s
334       double u_f;     //liquid layer velocity, m/s
335       double u_i;     //liquid-vapor interface velocity, m/s
336       double u_vt;    //laminar-turbulent interface velocity, m/s
337
338       double r_v[100];//laminar sublayer radius, m
339       double r_i;     //interfacial radius, m
340       double r_vt;    //laminar-turbulent interface radius, m
341
342       double t_v_Rad[100];//laminar sublayer temperature, C
343       double t_g;     //gas core temperature, C
344       double t_i;     //liquid-vapor interface temperature, C
345       double t_vt;    //laminar-turbulent interfacetemperature, C
346
347       double htc_v[100];
348       double htc_rv;
349       double htc_radial;
350       double HH;
351
352       //pure refrigerant saturation properties used to recalculate
          nanolubricant-refrigerant mixture
353       double mu_f_pure;
354       double rho_f_pure;
355       double cp_f_pure;
356       double k_f_pure;
357
358       double VolSeg;      //segment volume, m3
359       double OilSeg;      //oil mass per segment, kg
360       double OilNanoSeg;  //nanooil mass per segment, kg
361       double NanoSeg;     //nanoparticles mass per segment, kg
362       double RefSegLIQ;   //liquid refrigerant mass per segment, kg
363       double RefSegVAP;   //vapor refrigerant mass per segment, kg
364       double RefSeg;      //total refrigerant mass per segment, kg
365       double N_nano_Seg;  //number of nanoparticles inside the segment, -
366
367       double DpSeg_tot_;
368       double Nu_radial;
369       double Pr_v_Rad[100];
370       double t_wall_radial;
371       double Re_tp[100];  //two-phase Reynolds number, -
372       double Re_fo;       //Reynolds number, liquid only, -
373       double Pr_fo;       //Prandtl number, liquid only,  -
374       double Re_af;       //Reynolds number, all fluid, -
375       double Pr_af;       //Prandtl number, all fluid, -
376
377       int j;      //iteration index
378
379       double u_f_star;
380       double y_plus_i;
381       double y_plus_ave;
382       double y_plus[100];
383
```

```
384         double a, m, cc, ex;    //correlation coefficients
385         double htc_nb;      //nucleate boiling heat transfer coefficient, W/m2-C
386         double htc_TPmix;   //two-phase mixture heat transfer coefficient, W/m2-K
387         double htc_tot;
388         double C0, C1, M0, M1;
389         double E;       //two-phase convection multiplier
390         //double S;     //boiling suppression factor
391
392         double delta_f;     //thickness of liquid layer, m
393         double DpSeg_tot_Choi;
394
395         double t_wall_iter;
396
397         double htc;      //heat transfer coefficient, W/m2-C
398         double htc_cb;
399         double htc_exp;     //experimental heat transfer coefficient, W/m2-C
400         double htc_corr_Ham;
401         double htc_corr_Hu;
402         double htc_corr_Zuo;
403         double htc_corr_Saw;
404
405         ////
406         double Xtt;             //Martinelli parameter
407
408         double ratioF[100];
409         //double Xtt[100];      //Martinelli parameter
410
411         double F[100];
412         double S[100];
413
414         double htc_mac[100];
415         double htc_mic[100];
416
417         int index;
418
419
420
421         ////////
422
423
424         // flow
425
426         set_reference_stateS(fluid, ent_ref);
427
428
429         ////OPEN OUTPUT FILE
430         ofstream output_file;
431         output_file.open("out_tu.csv");
432         //writing legend
433         output_file << "htc_exp, htc_nb, htc_cb, Re_fo, Pr_fo, Xtt, G_flux, q_flux,
            delta_f, phi, phi_v, N_bt,"
434                     "x_in, rho_g, rho_f, mu_g, mu_f, cp_g, cp_f, k_g, k_f" << endl;
435
436         ofstream radial_output_file;
437         radial_output_file.open("out_tu_radial.csv");
438
439         ofstream Buongiorno_output_file;
440         Buongiorno_output_file.open("out_Buongiorno.csv");
441         //writing legend
442         Buongiorno_output_file << "G_flux [kg/m2-s], q_flux [W/m2], phi, N_bt, phi_v, ff,
            tau_w [Pa],"
443             "D_B [m2/s], grad_T [K/m], V_T [m2/s], delta_v [m], u_ave [m/s], t_wall [C],
            Re_b, Pr_b, Pr_v, Nu_b, htc,"
444             "rho_b [kg/m3], cp_b [J/kg-K], mu_b [N-s/m2], k_b [W/m-K],"
445             "rho_v [kg/m3], cp_v [J/kg-K], mu_v [N-s/m2], k_v [W/m-K]" << endl;
446
```

```
447         ofstream Buongiorno_radial_output_file;
448         Buongiorno_radial_output_file.open("out_Buongiorno_radial.csv");
449
450         ofstream test_props_file;
451         test_props_file.open("test_props.csv");
452
453         int index_Nbt = 0;
454         //this loop is added only to run a sensitivity analysis on Nbt, inside the
            Buongiorno routine
455         while (index_Nbt <= 0)
456         {
457
458             index = 0;
459
460             ////OPEN INPUT FILE WITH TEST DATA
461             ifstream input_file_data;
462             input_file_data.open("in.csv");
463             getline(input_file_data, line);  //read first line: file legend
464             while (getline(input_file_data, line))
465             {
466                 index = index + 1;
467
468                 istringstream ss(line);
469                 string token;
470                 vector<string> tokens;
471
472                 while (getline(ss, token, ','))
473                     tokens.push_back(move(token));
474
475                 m_dot_fluid = stof(tokens[0]);
476                 Q_pre = stof(tokens[1]);
477                 p_pre_in = stof(tokens[2]);
478                 t_pre_in = stof(tokens[3]);
479                 Q = stof(tokens[4]);
480                 p_in = stof(tokens[5]);
481                 h_in = stof(tokens[6]);
482                 t_wall = stof(tokens[7]);
483                 OMF = stof(tokens[8]);
484                 NMF = stof(tokens[9]);
485                 htc_exp = stof(tokens[10]) * 1000;
486
487                 ////geometry calculations
488                 if (tube_type == "smooth")
489                 {
490                     SectA = M_PI*pow(Dh, 2) / 4;
491                     SurfA = M_PI*Dh*lengthTube;
492
493                     q_flux = Q / SurfA;
494                     G_flux = m_dot_fluid / SectA;
495
496                     ////segmentation
497                     lengthSeg = lengthTube / nSeg;
498                     DPlengthSeg = DPlengthTube / nSeg;
499                     SurfASeg = M_PI*Dh*lengthSeg;
500                     QSeg = q_flux * SurfASeg;    //or also: QSeg = Q / nSeg;
501
502                     De = Dh;    //for the case of a smooth tube (De is used in the radial
                        subroutine)
503                 }
504                 else if (tube_type == "microfin")
505                 {
506                     sigma_deg = 180 - 90 - alpha_deg / 2;
507                     baseFin = 2 * heightFin / tan(sigma_deg * M_PI / 180);
508                     sideFin = heightFin / sin(sigma_deg * M_PI / 180);
509                     Af = (baseFin * heightFin / 2) + tw * (baseFin + Sp / cos(beta_deg *
                        M_PI / 180) - 2 * sideFin);
```

```
510
511                 SectA = M_PI / 4 * pow(Do, 2) - nFins * Af;
512                 De = sqrt(4 * SectA / M_PI);
513                 Dh = 4 * SectA * cos(beta_deg * M_PI / 180) / (nFins * Sp);
514
515                 SurfA = nFins * Sp / cos(beta_deg * M_PI / 180) * lengthTube;
516
517                 q_flux = Q / SurfA;
518                 G_flux = m_dot_fluid / SectA;
519
520                 ////segmentation
521                 lengthSeg = lengthTube / nSeg;
522                 DPlengthSeg = DPlengthTube / nSeg;
523                 SurfASeg = nFins * Sp / cos(beta_deg * M_PI / 180) * lengthSeg;
524                 QSeg = q_flux * SurfASeg;    //or also: QSeg = Q / nSeg;
525             }
526
527             ////calculations
528
529             N_bt = 0;
530             phi = 0;
531             phi_v = 0;
532             DpSeg_tot_Choi = 0;     //initialized to zero, although not actually used
                in Radial
533             SurfASeg = M_PI*De*lengthSeg;
534
535             //calculate bulk properties for the case with water and nanoparticles
536             if (OMF == 0 && NMF > 0)
537             {
538                 //calculation of the saturated properties before nanoparticles (NMF = 0)
539                 CalcProps(fluid, A0, A1, A2, a0, b0, oil, nano_mater, nano_shape, D_nano
                    , 0, 0, m_dot_fluid, p_in, h_in,
540                     t_in, x_in, rho_in, v_in, cp_in, mu_in, k_in, sigma_in,
541                     t_sat_f, h_f, rho_f, v_f, cp_f, mu_f, k_f, sigma_f,
542                     t_sat_g, h_g, rho_g, v_g, cp_g, mu_g, k_g, sigma_g, h_fg, dh_x,
                        t_sat, w_local);
543
544                 rho_nano = 3880;    // NanoDensity(nano_mater);     //3880;
545                 k_nano = NanoConductivty(nano_mater, t_in);     //36;   40
546                 cp_nano = NanoSpecificHeat(nano_mater, t_in) * 1000;
547
548                 //calculation of the volume concentration in water
549                 phi = NMF / (NMF + (1 - NMF)*rho_nano / rho_f);
550
551                 rho_f = phi*rho_nano + (1 - phi)*rho_f;     //Buongiorno
552                 mu_f = mu_f*(1 + 39.11*phi + 533.9*pow(phi, 2));    //Buongiorno
553                 k_f = k_f*(1 + 7.47*phi);                   //Buongiorno
554                 cp_f = phi*cp_nano + (1 - phi)*cp_f * 1000;     //Buongiorno
555                 sigma_f = sigma_f;
556
557             }
558             else
559             {
560                 ////calculation of inlet fluid properties
561                 CalcProps(fluid, A0, A1, A2, a0, b0, oil, nano_mater, nano_shape,
562                     D_nano, OMF, NMF, m_dot_fluid, p_in, h_in,
563                     t_in, x_in, rho_in, v_in, cp_in, mu_in, k_in, sigma_in,
564                     t_sat_f, h_f, rho_f, v_f, cp_f, mu_f, k_f, sigma_f,
565                     t_sat_g, h_g, rho_g, v_g, cp_g, mu_g, k_g, sigma_g, h_fg, dh_x,
                        t_sat, w_local);
566
567                 cp_f = cp_f * 1000;
568
569             }
570
571             Re_af = G_flux * Dh / mu_f;
```

```
572             Pr_af = mu_f*cp_f / k_f;
573             Re_fo = G_flux * (1 - x_in) * Dh / mu_f;
574             Pr_fo = Pr_af;
575
576             ////Martinelli parameter
577             Xtt = pow((1 - x_in) / x_in, 0.9) * pow(rho_g / rho_f, 0.5) * pow(mu_f /
                mu_g, 0.1);
578
579
580             ////////////////////////////segment inventory, based on section
                area/////////////////////////////
581             alpha = RouhaniAxelsson(fluid, G_flux, sigma_f, rho_g, rho_f, x_in);
582             Inventory(fluid, nano_mater, D_nano, SectA, lengthSeg, alpha, w_local, NMF,
                rho_g, rho_f,
583                 VolSeg, RefSegLIQ, RefSegVAP, RefSeg, OilNanoSeg, NanoSeg, OilSeg,
                    V_nano, N_nano_Seg);
584
585             ////Friction factor calculation
586             ff = FrictionFactor(tube_type, tube_mater, tube_roughness, nFins, heightFin,
                 beta_deg, alpha_deg,
587                 G_flux, m_dot_fluid, SectA, De, Dr, rho_f, mu_f);
588
589
590             ////pool boiling heat transfer coefficient
591
592             SurfASeg = M_PI*De*lengthSeg;
593             if (OMF > 0 && NMF > 0)
594             {
595                 //volume concentration in lubricant only
596                 phi = NanoVolumeFraction(nano_mater, NMF, t_in);
597
598                 htc_nb = PoolHTC_Kedzierski2012(fluid, nano_mater, D_nano, OMF, phi,
                    N_nano_Seg, SurfASeg, t_sat,
599                     t_wall, x_in, sigma_f, h_fg, q_flux);
600             }
601             else if (OMF > 0 && NMF == 0)
602             {
603
604                 htc_nb = PoolHTC_Kedzierski2003(fluid, OMF, t_sat, t_wall, x_in, h_fg,
                    q_flux);
605             }
606             else if (OMF == 0 && NMF == 0)
607             {
608                 htc_nb = PoolHTC_ForsterZuber(fluid, t_sat, t_wall);
609             }
610             else if (OMF == 0 && NMF > 0)
611             {
612                 //need a pool boiling correlation for nanofluid without oil
613                 htc_nb = 1;
614             }
615
616
617
618             ////radial analysis
619             if (x_in > 0 && x_in < 1)
620             {
621
622                 Radial(nRad, index_Nbt, radial_output_file,
623                     fluid, A0, A1, A2, a0, b0, oil, nano_mater, nano_shape, D_nano, ff,
624                     OMF, NMF, N_nano_Seg, m_dot_fluid, p_in, h_in,
625                     t_in, x_in, rho_in, v_in, cp_in, mu_in, k_in, sigma_in,
626                     t_sat_f, h_f, rho_f, v_f, cp_f, mu_f, k_f, sigma_f,
627                     t_sat_g, h_g, rho_g, v_g, cp_g, mu_g, k_g, sigma_g, h_fg, dh_x,
                        t_sat, w_local,
628                     Dh, De, SectA, lengthSeg, orientation, t_wall, G_flux, q_flux,
                        DpSeg_tot_Choi, delta_f, htc_radial);
```

```
            }
            else
            {
                delta_f = 0;
            }

            ////convective heat transfer coefficient, from Buongiorno
            ////it converges to Dittus-Boelter in case of NMF = 0
            if (NMF >= 0)
            {

                htc_cb = Buongiorno2006(nRad, index_Nbt, Buongiorno_output_file,
                Buongiorno_radial_output_file,
                    fluid, A0, A1, A2, a0, b0, oil, nano_mater, nano_shape, D_nano, ff,
                    OMF, NMF, N_nano_Seg, m_dot_fluid, p_in, h_in,
                    t_in, x_in, rho_in, v_in, cp_in, mu_in, k_in, sigma_in,
                    t_sat_f, h_f, rho_f, v_f, cp_f, mu_f, k_f, sigma_f,
                    t_sat_g, h_g, rho_g, v_g, cp_g, mu_g, k_g, sigma_g, h_fg,
                    dh_x, t_sat, w_local, Dh, De, SectA, lengthSeg, t_wall, G_flux,
                    q_flux, delta_f,
                    N_bt, phi);

                phi_v = phi*N_bt*(1 - exp(-1 / N_bt));

            }
            else if (NMF == 0)
            {
                //Dittus-Boelter correlation
                htc_cb = 0.023 * (k_f / Dh) * pow(Re_af, 0.8) * pow(Pr_af, 0.4);
            }



            output_file << htc_exp << "," << htc_nb << "," << htc_cb << "," << Re_fo <<
            "," << Pr_fo << "," << Xtt <<
                "," << G_flux << "," << q_flux << "," << delta_f << "," << phi << "," <<
                 phi_v << "," << N_bt <<
                "," << x_in << "," << rho_g << "," << rho_f << "," << mu_g << "," <<
                mu_f <<
                "," << cp_g << "," << cp_f << "," << k_g << "," << k_f << endl;


        }



    input_file_data.close();

    index_Nbt++;
    Buongiorno_output_file << endl;

    }

    output_file.close();
    radial_output_file.close();
    Buongiorno_output_file.close();
    Buongiorno_radial_output_file.close();
    test_props_file.close();

}
```

```
#define _USE_MATH_DEFINES
#include <cmath>
#include "CoolProp.h"
#include "HumidAirProp.h"
#include <iostream>
#include <stdlib.h>
#include <fstream>
#include <string>
#include <cstddef>


using namespace CoolProp;
using namespace std;

void CalcProps(string, double, double, double, double, double,
                string, string, string, double, double, double, double, double, double,
                double&, double&, double&, double&, double&, double&, double&, double&,
                double&,
                double&, double&, double&, double&, double&, double&, double&, double&,
                double&,
                double&, double&, double&, double&, double&, double&, double&, double&,
                double&, double&);

void Inventory(string, string, double, double, double, double, double, double, double,
double,
                double&, double&, double&, double&, double&, double&, double&, double&,
                double&);

double RouhaniAxelsson(string, double, double, double, double, double);
double NanoDensity(string);
double NanoConductivty(string, double);
double NanoSpecificHeat(string, double);
double OilMixtureDensity(double, double, double);
double OilMixtureConductivity(double, double, double);
double NanoVolumeFraction(string, double, double);
double NanoOilDensity(string, double, double);
double NanoOilConductivity(string, double, double);
double NanoOilDynViscosity(string, double, double, double);
double NanoOilSpecificHeat(string, double, double);
double NanoOilMixtureDensity(string, double, double, double, double);
double NanoOilMixtureSpecificHeat(string, double, double, double, double);
double NanoOilMixtureDynViscosity(string, double, double, double, double, double);
double NanoOilMixtureConductivity(string, double, double, double, double);

double PoolHTC_Kedzierski2012(string, string, double, double, double, double, double,
    double, double, double, double, double, double);

void Radial(double nRad, int index_Nbt, ofstream &radial_output_file, string fluid,
double A0, double A1, double A2, double a0, double b0,
    string oil, string nano_mater, string nano_shape, double D_nano, double ff,
    double OMF, double NMF, double n_np_Seg, double m_dot, double p_in, double h_in,
    double t_in, double x_in, double rho_in, double v_in, double cp_in, double mu_in,
    double k_in, double sigma_in,
    double t_sat_f, double h_f, double rho_f, double v_f, double cp_f, double mu_f,
    double k_f, double sigma_f,
    double t_sat_g, double h_g, double rho_g, double v_g, double cp_g, double mu_g,
    double k_g, double sigma_g, double h_fg,
    double dh_x, double t_sat, double w_local, double Dh, double De, double SectA,
    double lengthSeg, string orientation,
    double t_wall, double G_flux, double q_flux, double DpSeg_tot_Choi, double &delta_f,
     double &htc_radial)
{
    //use: calculate radial nanoparticle distribution
    //
    //source: L. Cremaschi (2012). A Fundamental View of the Flow Boiling Heat Transfer
    Characteristics of Nano-Refrigerants, ASME
```

```
55      //
56      //        Wen, Ding (2005). Effect of Particle Migration on Heat Transfer in
        Suspensions
57      //                  of Nanoparticles Flowing Through Minichannels, MN
58      //
59      //        V. Carey (2008). Liquid-Vapor Phase-Change Phenomena (LVPCP), Second
        edition, CRC press
60      //
61      //        G. F. Hewitt (1970). Annular Two-Phase Flow (ATPF), First edition,
        Pergamon Press
62      //
63      //author: Andrea Bigi
64      //date: 07/2017
65      //-----------------------------------------------------------------
66
67      //local variables
68
69      double VolSeg;      //segment volume, m3
70      double OilSeg;      //oil mass per segment, kg
71      double OilNanoSeg;  //nanooil mass per segment, kg
72      double NanoSeg;     //nanoparticles mass per segment, kg
73      double RefSeg;      //total refrigerant mass per segment, kg
74      double RefSegLIQ;   //liquid refrigerant mass per segment, kg
75      double RefSegVAP;   //vapor refrigerant mass per segment, kg
76      double NanoOilRefRad;   //liquid refrigerant and nanooil mass, inside each radius
        of the laminar sublayer, kg
77      double OilRad;      //oil mass, inside each radius of the laminar sublayer, kg
78      double OilNanoRad;  //nanooil mass, inside each radius of the laminar sublayer, kg
79      double NanoRad;     //nanoparticles mass, inside each radius of the laminar
        sublayer, kg
80      double V_nano;      //nanopartcile volume, m3
81      double N_nano_Seg;  //number of nanoparticles inside the segment, -
82
83      double mu_v_Rad[100];   //laminar sublayer viscosity, N-s/m2, kg/m-s, Pa-s
84      double k_v_Rad[100];    //laminar sublayer conductivity, W/m-K
85      double cp_v_Rad[100];   //laminar sublayer specific heat, kJ/kg-K
86      double rho_v_Rad[100];  //laminar sublayer density, kg/m3
87      double phi_v_Rad[100];  //laminar sublayer nanoparticle volume fraction
88      double NMF_v_Rad[100];  //laminar sublayer nanoparticle mass fraction (for
        nanolubricant-refrigerant mixture)
89      double NMF_Rad[100];    //laminar sublayer nanoparticle mass fraction (for
        nanolubricant only)
90      double grad_T_Rad[100]; //laminar sublayer temperature gradient, K/m
91      double Vol_rate_Rad[100];//laminar sublayer volumetric flow rate per unit wetted
        perimeter, m2/s
92
93      double m_dot_f; //total film mass flow rate, kg/s
94      double m_dot_f_calc;    //calculated total film mass flow rate, kg/s
95      double m_dot_f_calc_star;//dimensionless calculated total film mass flow rate, -
96
97      double alpha;   //void fraction
98      double Xtt;     //Martinelli parameter
99      //double delta_f;   //liquid thickness, m
100     double delta_f_star;//dimensionless liquid thickness, -
101     double delta_v;     //laminar sublayer thickness, m
102     double SurfASeg;    //segment surface area, m2
103
104     double ff_i;    //interfacial friction factor, -
105     double tau_v[100];//laminar sublayer shear stress, Pa
106     double tau_i;   //interfacial shear stress, Pa
107     double tau_vt;  //laminar-turbulent interface shear stress, Pa
108
109     double q_v[100];//laminar sublayer heat flux, W/m2
110     double q_wall;  //wall heat flux, W/m2
111     double q_i;     //liquid-vapor interface heat flux, W/m2
112     double q_vt;    //laminar-turbulent interface heat flux, W/m2
```

```
113
114         double u_v[100];//laminar sublayer  velocity, m/s
115         double u_g;     //gas core velocity, m/s
116         double u_f;     //liquid layer velocity, m/s
117         double u_i;     //liquid-vapor interface velocity, m/s
118         double u_vt;    //laminar-turbulent interface velocity, m/s
119
120         double r_v[100];//laminar sublayer radius, m
121         double r_i;     //interfacial radius, m
122         double r_vt;    //laminar-turbulent interface radius, m
123
124         double t_v_Rad[100];//laminar sublayer temperature, C
125         double t_g;     //gas core temperature, C
126         double t_i;     //liquid-vapor interface temperature, C
127         double t_vt;    //laminar-turbulent interfacetemperature, C
128
129         double DxSeg;   //segment delta quality, -
130         double DpSeg_fric;  //segment frictional pressure drop, Pa
131         double DpSeg_mom;   //segment momentum pressure drop, Pa
132         double DpSeg_grav;  //segment gravitational pressure drop, Pa
133         double DpSeg_tot_;  //segment delta pressure, Pa
134
135         double HH;          //heat transfer parameter
136         double htc[100];
137         double htc_tot;         //heat transfer coefficient, W/m2-C
138         double htc_nb;      //nucleate boiling heat transfer coefficient, W/m2-C
139         double htc_TPmix;   //two-phase mixture heat transfer coefficient, W/m2-K
140         double htc_Liq;     //liquid heat transfer coefficient, W/m2-K
141         double htc_rv;
142         double htc_core;
143
144         int j;      //iteration index
145
146         //variables for Buongiorno analysis
147         //Buongiorno variables
148         double C, n;    //friction factor coefficients
149         double beta;    //thermophoretic coefficient
150         double beta_oil;    //thermophoresis coefficient / proportionality factor for oil
151         double beta_ref;    //thermophoresis coefficient / proportionality factor for
        refrigerant
152         double Re;      //Reynolds number
153         double Re_g;    //Reynolds number of vapor phase
154         double Re_f;    //Reynolds number of liquid phase
155         double tau_w;   //shear stress at the wall, Pa
156         //double ff;        //friction factor
157         double u_ave;   //mean axial velocity or bulk velocity, m/s
158         double k_B = 1.38064852e-23;    //Boltzmann constant, J/K
159         double D_B;     //Brownian diffusion coefficient, m2/s
160         double V_T;     //thermophoretic diffusion coefficient, m2/s
161         double phi_v_guess;     //nanoparticle volume fraction initial guess
162         double phi_v;   //nanoparticle volume fraction in laminar sublayer
163         double phi_b;   //nanoparticle volume fraction in oil and refrigerant
164         double NMF_b;   //nanoparticle mass fraction in oil and refrigerant
165         double delta_v_plus;    //non-dimensional thickness of the laminar sublayer
166         double N_bt;    //ratio of Brownian and thermophoretic diffusivities
167         double grad_T;  //film temperature gradient, K
168         double t_wall_new;  //new wall temperature, K
169
170         //variables for beta, the thermophoretic coefficient
171         double Cm;
172         double Cs;
173         double Ct;
174         double Kratio;
175         double Knudsen;
176
177         //properties variables
```

```
178       double NMF_v;   //nanoparticle mass fraction in refrigerant and oil, in laminar
          sublayer
179       double NMF_v_oil;   //nanoparticle mass fraction in oil, in laminar sublayer
180       double t_sat_gv;    //vapor saturation temperature calculated with the laminar
          sublayer temperature
181       double mu_fv;
182       double rho_fv;
183       double cp_fv;
184       double k_fv;
185       double t_inv;   //fluid temperature calculated with the laminar sublayer temperature
186       double p_inv;   //fluid pressure calculated with the laminar sublayer temperature
187       double t_v;     //temperature
188       double mu_v;    //dynamic viscosity in laminar sublayer, Pa*s
189       double rho_v;
190       double cp_v;
191       double k_v;
192       double mu_b;    //dynamic viscosity in turbulent sublayer, Pa*s
193       double rho_b;
194       double cp_b;
195       double k_b;
196       double sigma_b;
197       double Pr_v;
198       double Pr_b;
199       double Re_b;
200       double Re_vt;   //Reynolds number of liquid phase
201
202       //nanoparticles properties
203       double k_nano;  //nanoparticle thermal conductivity, W/m-K
204       double cp_nano; //nanoparticle specific heat, kJ/kg-K
205       double rho_nano;//nanoparticle density, kg/m3
206
207       //pure refrigerant saturation properties used to recalculate
          nanolubricant-refrigerant mixture
208       double mu_f_pure;
209       double rho_f_pure;
210       double cp_f_pure;
211       double k_f_pure;
212
213       //oil-refrigerant properties without nanoparticles
214       double rho_f_N_v_Rad[100];
215       double rho_f_N;
216       double k_f_N;
217
218
219       double htc_cond;
220       double rho_g_star;
221       double h_fg_star;
222       double lengthSeg_calc;
223
224       double f_g;     //friction factor for the gas core flowing in the absence of the film
225
226       double u_g_star;
227       double u_f_star;
228       double y_plus[100];
229       double y_plus_i;
230
231       double Pr_v_Rad[100];
232
233       double delta_f_new;
234       double Re_fo;   //Reynolds number, fluid only, -
235       double C0, C1;
236       double E;       //two-phase convection multiplier
237       double S;       //boiling suppression factor
238
239       double phi;
240       double htc_cb;  //convective boiling heat transfer coefficient, W/m2-K
```

```
241
242         //output variables
243
244         //flow
245
246         htc_radial = 0;
247         htc_nb = 0;
248         htc_tot = 0;
249
250         if (NMF == 0)
251         {
252             rho_nano = 0;
253             k_nano = 0;
254             cp_nano = 0;
255         }
256
257         //properties definition
258
259         rho_f_pure = PropsSI("D", "P", p_in * 1000, "Q", 0, fluid);
260         k_f_pure = PropsSI("L", "P", p_in * 1000, "Q", 0, fluid);
261         cp_f_pure = PropsSI("C", "P", p_in * 1000, "Q", 0, fluid) / 1000;
262         mu_f_pure = PropsSI("V", "P", p_in * 1000, "Q", 0, fluid);
263
264         rho_b = rho_f;
265         k_b = k_f;
266         cp_b = cp_f * 1000;
267         mu_b = mu_f;
268         sigma_b = sigma_f;
269
270         k_f_N = OilMixtureConductivity(t_in, w_local, k_f_pure);
271
272
273         //initialization
274         DpSeg_fric = 0;
275         DpSeg_mom = 0;
276         DpSeg_grav = 0;
277
278
279         //mass flow rate in the liquid film
280         m_dot_f = SectA*G_flux*(1 - x_in);
281
282         ////////////////////////////loop to find liquid film thickness delta_f - LVPCP pag.
            519-522
283         //first guess of the overall film thickness
284         delta_f = De/2 * 0.1;
285         while (TRUE)
286         {
287             //Xtt = pow((1 - x_in) / x_in, 0.9) * pow(rho_g / rho_b, 0.5) * pow(mu_b /
                mu_g, 0.1);
288             //alpha = pow(1 + pow(Xtt, a), m);      //Martinelli, taken from Cremaschi paper
289             //alpha = RouhaniAxelsson(fluid, G_flux, sigma_b, rho_g, rho_b, x_in);
290             alpha = pow((De - 2 * delta_f), 2) / pow(De, 2);    //theoretical definition
291
292             //actual/mean velocity of the vapor core - ATPF, pag.25 or 56
293             u_g = G_flux*x_in / (rho_g*alpha);
294             ////superficial velocity of the vapor core - LVPCP pag. 480-481
295             //u_g = G_flux*x_in / (rho_g);
296
297             //Re_g = G_flux*x_in*(De - 2 * delta_f) / mu_g;
298             Re_g = G_flux*x_in*De / mu_g;
299
300             //actual/mean velocity across the liquid film - ATPF, pag.25
301             u_f = G_flux*(1 - x_in) / (rho_b*(1 - alpha));
302             ////superficial velocity across the liquid film - LVPCP pag. 480-481
303             //u_f = G_flux*(1 - x_in) / (rho_b);
304
```

```
305         ////annulus liquid Reynolds number
306         //Re_f = G_flux*(1 - x_in) * (2 * delta_f) / mu_b;  //Byrd, Stewart, Transport
            Phenomena, pag.56
307         //liquid film Reynolds number
308         Re_f = 4 * u_f * delta_f *rho_b / mu_b;     //ATPF, pag.79
309         Re_f = 4 * G_flux*(1 - x_in) * delta_f / (mu_b*(1 - alpha));
            //equivalent to ATPF, pag.79
310         Re_f = 4 * m_dot_f / (mu_b*M_PI*Dh);    //this Reynolds number is equivalent
            to: G_flux * (1 - x_in) * Dh / mu_f
311                                                 //the value however might change
                                                    depending on the diameter used in case
                                                    of microfin geometry
312
313         r_i = De / 2 - delta_f;
314         //ff_i = 0.005*(1 + 300 * delta_f / De);
315         f_g = 0.079*pow(Re_g, -0.5);
316         ff_i = f_g*(1 + 300 * delta_f / De);    //ATPF pag.93
317         tau_i = 0.5*ff_i*rho_g*pow(u_g, 2);
318
319         ////this formula can be used if DpSeg is already known - ATPF pag.57
320         //tau_i = -r_i / 2 * -DpSeg_tot_ - pow(De / 2 / r_i, 2) * 2 * u_g * q_flux /
            (h_fg * 1000);
321
322         //quality change
323         DxSeg = 4 * q_flux / (G_flux * De * h_fg * 1000);
324
325         //pressure drop
326         DpSeg_fric = -4 * tau_i / De;       //LVPCP; pressure drop in the gas core is
            assumed to be equal to the pressure drop in the liquid film
327         DpSeg_mom = -(2 * x_in * pow(G_flux, 2)) / (pow(alpha, 2) * rho_g) * DxSeg;
328         if (orientation == "horizontal")
329         {
330             DpSeg_grav = 0;
331         }
332         else if (orientation == "vertical")
333         {
334             DpSeg_grav = -9.81*rho_g;
335         }
336         DpSeg_tot_ = DpSeg_fric+DpSeg_mom+DpSeg_grav;
337
338         //DpSeg_tot_ = DpSeg_fric;  //acceleration and gravitational effects can be
            ignored in the gas core
339
340
341         //laminar film mass flow rate - ATPF eq.4.43
342         m_dot_f_calc = 2 * M_PI*rho_b / mu_b*((tau_i*r_i + 0.5*DpSeg_tot_*pow(r_i, 2))*
343             (0.25*(pow(De / 2, 2) - pow(r_i, 2)) - 0.5*pow(r_i, 2)*log((De / 2) / r_i)) -
344             (DpSeg_tot_*pow(pow(De / 2, 2) - pow(r_i, 2), 2)) / 16);
345
346
347         ////turbulent film mass flow rate - ATPF pag.61
348         //u_f_star = sqrt(tau_i / rho_b);
349         //delta_f_star = delta_f*u_f_star*rho_b / mu_b;
350
351         //if (delta_f_star > 0 && delta_f_star < 5)
352         //{
353         //  m_dot_f_calc_star = (delta_f_star / 2) * ((3 * tau_i - 2 * rho_b*delta_f) /
            (3 * tau_i - 3 * rho_b*delta_f));
354         //}
355         //else if (delta_f_star >= 5 && delta_f_star < 30)
356         //{
357         //  m_dot_f_calc_star = (-8.05*delta_f_star + 5 *
            delta_f_star*log(delta_f_star) + 12.45) * ((3 * tau_i - 2 * rho_b*delta_f) / (3
            * tau_i - 3 * rho_b*delta_f));
358         //}
359         //else if (delta_f_star >= 30)
```

```
360             //{
361             //  m_dot_f_calc_star = (8 * delta_f_star + 2.5 *
                delta_f_star*log(delta_f_star) - 214) * ((3 * tau_i - 2 * rho_b*delta_f) / (3 *
                tau_i - 3 * rho_b*delta_f));
362             //}
363
364             //m_dot_f_calc = m_dot_f_calc_star*(De*M_PI*mu_b);
365
366
367             //convergence condition
368             if (abs(m_dot_f_calc - m_dot_f) < 0.00001)  //I guess there are better methods
                for convergence but this works for now...
369             {
370                 break;
371             }
372             else if (m_dot_f_calc > m_dot_f)
373             {
374                 delta_f = delta_f*0.95;
375             }
376             else if (m_dot_f_calc < m_dot_f)
377             {
378                 delta_f = delta_f*1.15;
379             }
380
381         }
382
383         //delta_f = delta_f/2;
384
385         ////////////////////////////segment inventory, based on section
            area//////////////////////////
386         Inventory(fluid, nano_mater, D_nano, SectA, lengthSeg, alpha, w_local, NMF, rho_g,
            rho_f,
387             VolSeg, RefSegLIQ, RefSegVAP, RefSeg, OilNanoSeg, NanoSeg, OilSeg, V_nano,
                N_nano_Seg);
388
389
390         ////////////////////////////begin of the analysis according to Buongiorno
            (2006)//////////////////////////////
391
392         //calculate bulk properties for the case with water and nanoparticles
393         if (OMF == 0 && NMF > 0)
394         {
395             //calculation of the saturated properties before nanoparticles (NMF = 0)
396             CalcProps(fluid, A0, A1, A2, a0, b0, oil, nano_mater, nano_shape, D_nano, 0, 0,
                m_dot, p_in, h_in,
397                 t_in, x_in, rho_in, v_in, cp_in, mu_in, k_in, sigma_in,
398                 t_sat_f, h_f, rho_f, v_f, cp_f, mu_f, k_f, sigma_f,
399                 t_sat_g, h_g, rho_g, v_g, cp_g, mu_g, k_g, sigma_g, h_fg, dh_x, t_sat,
                    w_local);
400
401             rho_nano = 3880;    // NanoDensity(nano_mater);     //3880;
402             k_nano = NanoConductivty(nano_mater, t_in);     //36;   40
403             cp_nano = NanoSpecificHeat(nano_mater, t_in) * 1000;
404
405             //calculation of the volume concentration in water
406             phi_b = NMF / (NMF + (1 - NMF)*rho_nano / rho_f);
407
408             rho_b = phi_b*rho_nano + (1 - phi_b)*rho_f;     //Buongiorno
409             mu_b = mu_f*(1 + 39.11*phi_b + 533.9*pow(phi_b, 2));    //Buongiorno
410             k_b = k_f*(1 + 7.47*phi_b);                  //Buongiorno
411             cp_b = phi_b*cp_nano + (1 - phi_b)*cp_f * 1000;     //Buongiorno
412             sigma_b = sigma_f;
413
414         }
415         ////calculate bulk properties for the case with nanolubricant and refrigerant
416         else
```

```
417         {
418             //calculation of the saturated properties before nanoparticles (NMF = 0)
419             CalcProps(fluid, A0, A1, A2, a0, b0, oil, nano_mater, nano_shape, D_nano, OMF, 0
                , m_dot, p_in, h_in,
420                 t_in, x_in, rho_in, v_in, cp_in, mu_in, k_in, sigma_in,
421                 t_sat_f, h_f, rho_f, v_f, cp_f, mu_f, k_f, sigma_f,
422                 t_sat_g, h_g, rho_g, v_g, cp_g, mu_g, k_g, sigma_g, h_fg, dh_x, t_sat,
                    w_local);
423
424             //calculation of inlet fluid bulk properties
425             if (NMF > 0)
426             {
427                 CalcProps(fluid, A0, A1, A2, a0, b0, oil, nano_mater, nano_shape, D_nano,
                    OMF, NMF, m_dot, p_in, h_in,
428                     t_in, x_in, rho_in, v_in, cp_in, mu_in, k_in, sigma_in,
429                     t_sat_f, h_f, rho_b, v_f, cp_b, mu_b, k_b, sigma_b,
430                     t_sat_g, h_g, rho_g, v_g, cp_g, mu_g, k_g, sigma_g, h_fg, dh_x, t_sat,
                        w_local);
431
432                 rho_nano = NanoDensity(nano_mater);     //3880;
433                 k_nano = NanoConductivty(nano_mater, t_in);     //36;   40
434                 cp_nano = NanoSpecificHeat(nano_mater, t_in) * 1000;
435
436                 //in case of oil, NMF from the input file is intended to be the mass
                    fraction of nanoparticles in oil only,
437                 //therefore it is necessary to recalculate NMF as the mass fraction of the
                    oil-refrigerant mixture
438                 NMF_b = NanoSeg / (NanoSeg + OilSeg + RefSegLIQ);
439                 phi_b = NMF_b / (NMF_b + (1 - NMF_b)*rho_nano / rho_f);
440
441             }
442             else
443             {
444                 rho_b = rho_f;
445                 cp_b = cp_f;
446                 mu_b = mu_f;
447                 k_b = k_f;
448             }
449
450             cp_b = cp_b * 1000;
451         }
452
453         if (NMF == 0)
454         {
455             phi_b = 0;
456         }
457
458
459         //(i)
460         C = 0.184;
461         n = 0.2;
462         u_ave = m_dot / (rho_b * SectA);
463         Re_b = rho_b * u_ave * Dh / mu_b;
464
465         //passed as input parameter
466         //ff = 0.184 / pow(Re_b, 0.2);      //McAdams friction factor correlation
467         //ff = 0.046 / pow(Re_b, 0.2);      //Taitel&Dukler friction factor correlation
468
469         tau_w = ff / 8 * rho_b * pow(u_ave, 2);
470         //tau_w = C / 8 * pow(mu_b, 2)*pow(Re_b, 2 - n) / (rho_b*pow(Dh, 2));
471
472         //for annular flow
473         tau_w = tau_i*r_i / (De / 2) + 0.5*DpSeg_tot_*(pow(r_i, 2) - pow((De / 2), 2)) / (De
             / 2);
474
475         //(ii)
```

```
476         phi_v_guess = 0;    // 0.001;   //guess value of nanoparticle volume fraction in
            laminar sublayer
477         delta_v_plus = 0;   // 15.5;  // 8.7
478
479         //delta_v = delta_f*3/4;
480
481         //(iii) - to find the nanoparticle volumetric concentration in the laminar sublayer
482         while (TRUE)
483         {
484             ////////////////////////////////////////////////////////////////////////////
485
486             alpha = pow((De - 2 * delta_f), 2) / pow(De, 2);    //theoretical definition
487
488             //actual/mean velocity of the vapor core - ATPF, pag.56
489             u_g = G_flux*x_in / (rho_g*alpha);
490             ////superficial velocity of the vapor core
491             //u_g = G_flux*x_in / (rho_g);
492
493             Re_g = G_flux*x_in*(De - 2 * delta_f) / mu_g;
494             Re_g = G_flux*x_in*De / mu_g;
495
496             //actual/mean velocity across the liquid film
497             u_f = G_flux*(1 - x_in) / (rho_b*(1 - alpha));
498             ////superficial velocity across the liquid film
499             //u_f = G_flux*(1 - x_in) / (rho_b);
500
501             //annulus liquid Reynolds number
502             Re_f = G_flux*(1 - x_in) * (2 * delta_f) / mu_b;
503             //Re_f = G_flux*(1 - x_in) * De / mu_b;
504
505             r_i = De / 2 - delta_f;
506
507             ////////////////////////////segment inventory, based on section
                area//////////////////////////
508             ///this inventory routine is repeated in case delta_f changes (causing alpha to
                change)///
509             Inventory(fluid, nano_mater, D_nano, SectA, lengthSeg, alpha, w_local, NMF,
                rho_g, rho_f,
510                 VolSeg, RefSegLIQ, RefSegVAP, RefSeg, OilNanoSeg, NanoSeg, OilSeg, V_nano,
                    N_nano_Seg);
511
512
513             ////////////////////////////////////////////////////////////////////////////
514
515             if (fluid == "water")
516             {
517
518                 //McNab&Meisen, suggested by Buongiorno
519                 beta = 0.26*(k_f_pure / (2 * k_f_pure + k_nano));
520
521                 ////alternative correlation for beta - (IJHMT - 2015 - Michaelides -
                    Brownian movement and thermophoresis of nanoparticles in liquids)
522                 //beta = 1227 * pow(D_nano / 2 / 1, -1.434);        //water
523             }
524             else if (fluid != "water" && OMF == 0)
525             {
526                 //alternative correlation for beta - (IJHMT - 2015 - Michaelides - Brownian
                    movement and thermophoresis of nanoparticles in liquids)
527                 beta = 6270 * pow(D_nano / 2 / 1, -1.819);      //R134a
528             }
529             else if (fluid != "water" && OMF > 0)
530             {
531                 //alternative correlation for beta - (IJHMT - 2015 - Michaelides - Brownian
                    movement and thermophoresis of nanoparticles in liquids)
532                 beta_oil = 7.1026*pow(D_nano / 2 / 1, -1.579);      //engine oil
533                 beta_ref = 6270 * pow(D_nano / 2 / 1, -1.819);      //R134a
```

```
534
535             //weighted beta for oil-refrigerant mixture //Bigi
536             beta = beta_oil * w_local + beta_ref * (1 - w_local);
537         }
538
539         //added for sensitivity analysis on Nbt
540         beta = beta / pow(10, 3 * index_Nbt);
541
542         //(iv - v - vi)
543         rho_v = rho_b;
544         cp_v = cp_b;
545         mu_v = mu_b;
546         k_v = k_b;
547
548         phi_v = phi_b;
549         delta_v = delta_f;
550
551         D_B = k_B*(t_sat_f + 273.15) / (3 * M_PI*mu_b*D_nano*1e-9);
552         grad_T = q_flux*delta_v / k_b;
553         V_T = beta*mu_b*grad_T / (rho_b*(t_sat_f + 273.15));
554         N_bt = D_B / V_T;
555
556         ////////////////////
557
558         ////////////////////////guess laminar sublayer thickness
559         //delta_v = delta_f / 2;        //guess to be changed according to some
            correlation???!!!
560         //while (TRUE)
561         //{
562
563         ////vapor core max velocity for turbulent flow
564         //u_g_star = sqrt(tau_w/rho_g);
565         //u_g = u_g_star*(5.5+2.5*log(u_g_star*rho_g*De/2/mu_g));
566
567         ////vapor-liquid interface
568         //u_i = sqrt(tau_w / rho_f);        //ATPF, pag.61
569
570         u_i = 1 / mu_b * ((tau_i*r_i + 0.5*DpSeg_tot_*pow(r_i, 2))*log(De / 2 / r_i) -
571             0.25*DpSeg_tot_*(pow(De / 2, 2) - pow(r_i, 2)));        //ATPF eq.4.42, or
                see notes;
572
573         //u_i = tau_i / mu_b * (De/2 - r_i);
574
575         //turbulent-laminar (vt) interface
576         r_vt = De / 2 - delta_v;
577
578         //u_vt = 1 / mu_v * ((tau_i*r_i + 0.5*DpSeg_tot_*pow(r_i, 2))*log(De / 2 /
            r_vt) -
579         //  0.25*DpSeg_tot_*(pow(De / 2, 2) - pow(r_vt, 2)));       //ATPF eq.4.42, or
            see notes;
580         //u_vt = u_i - 1 / mu_v * ((tau_i*r_i + 0.5*DpSeg_tot_*pow(r_i, 2))*log(r_vt /
            r_i) -
581         //  0.25*DpSeg_tot_*(pow(r_vt, 2) - pow(r_i, 2)));      //ATPF eq.4.42, or see
            notes;
582         u_vt = u_i;     //or, see notes
583
584         Re_vt = rho_b*u_vt*(2 * delta_f) / mu_b;
585
586         tau_vt = tau_i*r_i / r_vt + 0.5*DpSeg_tot_*(pow(r_i, 2) - pow(r_vt, 2)) / r_vt;
587         //tau_vt = tau_i;
588
589         q_i = q_flux*De / 2 / r_i;
590         q_vt = q_flux*De / 2 / r_vt;
591
592
593         t_i = t_in; //t_wall - grad_T * delta_v;    //interface temperature should be
```

```
        saturation temperature, t_sat_
594
595         //r_v[0] = r_vt;
596         //u_v[0] = u_vt;
597         //q_v[0] = q_vt;
598
599
600         //(vii)
601         t_wall_new = t_wall;
602         while (TRUE)
603         {
604             //average liquid layer temperature
605             t_v = (t_wall_new + t_in) / 2;
606             //p_inv = PropsSI("P", "T", t_v + 273.15, "Q", 0, fluid) / 1000;
607
608             //calculation of average properties without nanoparticles, function of t_v
609             rho_fv = OilMixtureDensity(t_v, w_local, rho_f_pure);
610
611             //laminar sublayer segmentation
612             for (j = 0; j <= nRad; j++)
613             {
614
615                 t_v_Rad[j] = t_i + ((t_wall_new - t_in) / delta_v) * delta_v / nRad*j;
616
617                 if (OMF == 0)
618                 {
619                     //calculate local NMF in oil AND liquid refrigerant
620                     phi_v_Rad[j] = phi_b*exp(-(1 - ((delta_v - delta_v / nRad*j) /
                        delta_v)) / N_bt);
621
622                     rho_f = PropsSI("D", "T", t_v_Rad[j] + 273.15, "Q", 0, fluid);
623                     mu_f = PropsSI("V", "T", t_v_Rad[j] + 273.15, "Q", 0, fluid);
624                     k_f = PropsSI("L", "T", t_v_Rad[j] + 273.15, "Q", 0, fluid);
625                     cp_f = PropsSI("C", "T", t_v_Rad[j] + 273.15, "Q", 0, fluid);
626                     cp_nano = NanoSpecificHeat(nano_mater, t_v_Rad[j]) * 1000;
627
628                     rho_v_Rad[j] = phi_v_Rad[j] *rho_nano + (1 - phi_v_Rad[j])*rho_f;
                                //Buongiorno
629                     mu_v_Rad[j] = mu_f*(1 + 39.11*phi_v_Rad[j] + 533.9*pow(phi_v_Rad[j],
                         2));   //Buongiorno
630                     k_v_Rad[j] = k_f*(1 + 7.47*phi_v_Rad[j]);
                                                  //Buongiorno
631                     cp_v_Rad[j] = phi_v_Rad[j] *cp_nano + (1 - phi_v_Rad[j])*cp_f;
                                  //Buongiorno
632                 }
633                 ////for nanolubricants
634                 else
635                 {
636                     rho_f_N_v_Rad[j] = OilMixtureDensity(t_v_Rad[j], w_local, rho_f_pure
                        );
637
638                     //calculate local NMF in oil AND liquid refrigerant
639                     phi_v_Rad[j] = phi_b*exp(-(1 - ((delta_v - delta_v / nRad*j) /
                        delta_v)) / N_bt);
640                     NMF_v_Rad[j] = (phi_v_Rad[j] * rho_nano / rho_f_N_v_Rad[j]) / (1 -
                        phi_v_Rad[j] + phi_v_Rad[j] * rho_nano / rho_f_N_v_Rad[j]);
641
642                     //laminar sublayer inventory
643                     NanoOilRefRad = ((OilNanoSeg + RefSegLIQ)*delta_v / delta_f) / nRad;
644                     OilNanoRad = (OilNanoSeg *delta_v / delta_f) / nRad;
645                     NanoRad = NMF_v_Rad[j] * NanoOilRefRad;
646
647                     //NMF in oil only - needed in the thermophysical properties routine
648                     //NMF_Rad[j] = NMF_v_Rad[j] * NanoOilRefRad / (NanoRad + OilNanoRad);
649                     NMF_Rad[j] = NMF_v_Rad[j] * NanoOilRefRad / (OilNanoRad);
650
```

```
651                     if (NMF_Rad[j] >= 1)
652                     {
653                         system("pause");
654                     }
655 
656                     ////calculation of properties with nanoparticles, at the value of
                        phi_v_Rad[j] AND at the pressure corresponding to t_v
657                     rho_v_Rad[j] = NanoOilMixtureDensity(nano_mater, NMF_Rad[j], t_v_Rad
                        [j], w_local, rho_f_pure);
658                     cp_v_Rad[j] = NanoOilMixtureSpecificHeat(nano_mater, NMF_Rad[j],
                        t_v_Rad[j], w_local, cp_f_pure) * 1000;
659                     mu_v_Rad[j] = NanoOilMixtureDynViscosity(nano_mater, D_nano, NMF_Rad
                        [j], t_v_Rad[j], w_local, mu_f_pure);
660                     if (mu_v_Rad[j] < 0)
661                     {
662                         system("pause");
663                     }
664                     k_v_Rad[j] = NanoOilMixtureConductivity(nano_mater, NMF_Rad[j],
                        t_v_Rad[j], w_local, k_f_pure);
665                 }
666 
667             Pr_v_Rad[j] = mu_v_Rad[j] * cp_v_Rad[j] / k_v_Rad[j];
668 
669             }
670 
671             for (j = 0; j <= nRad; j++)
672             {
673                 //calculations
674                 r_v[j] = r_vt + delta_v / nRad*j;
675 
676                 if (j == 0)
677                 {
678                     //u_v[j] = u_i - 1 / mu_v_Rad[j] * ((tau_i*r_i +
                        0.5*DpSeg_tot_*pow(r_i, 2))*log(r_v[j] / r_i) -
679                     //  0.25*DpSeg_tot_*(pow(r_v[j], 2) - pow(r_i, 2)));        //ATPF
                        eq.4.42, or see notes
680                     tau_v[j] = tau_i*r_i / r_v[j] + 0.5*DpSeg_tot_*(pow(r_i, 2) - pow(
                        r_v[j], 2)) / r_v[j];
681                 }
682                 else
683                 {
684                     //u_v[j] = u_v[j - 1] - 1 / mu_v_Rad[j] * ((tau_i*r_i +
                        0.5*DpSeg_tot_*pow(r_i, 2))*log(r_v[j] / r_v[j-1]) -
685                     //  0.25*DpSeg_tot_*(pow(r_v[j], 2) - pow(r_v[j-1], 2)));
                        //ATPF eq.4.42, or see notes
686                     tau_v[j] = tau_v[j - 1] * r_v[j - 1] / r_v[j] + 0.5*DpSeg_tot_*(pow(
                        r_v[j - 1], 2) - pow(r_v[j], 2)) / r_v[j];
687                 }
688 
689                 q_v[j] = q_flux * (De / 2) / r_v[j];
690 
691             }
692 
693             for (j = nRad; j >= 0; j--)
694             {
695                 if (j == nRad)
696                 {
697                     u_v[j] = 0;
698                 }
699                 else
700                 {
701                     //u_v[j] = u_v[j + 1] + 1 / mu_v_Rad[j + 1] * ((tau_v[j] * r_v[j] +
                        0.5*DpSeg_tot_*pow(r_v[j], 2))*log(r_v[j + 1] / r_v[j]) -
702                     //  0.25*DpSeg_tot_*(pow(r_v[j + 1], 2) - pow(r_v[j], 2)));
                        //ATPF eq.4.42, or see notes
703                     //u_v[j] = 1 / mu_v_Rad[j] * ((tau_v[j] * r_v[j] +
```

```
                    0.5*DpSeg_tot_*pow(r_v[j], 2))*log(De / 2 / r_v[j]) -
//                  0.25*DpSeg_tot_*(pow(De / 2, 2) - pow(r_v[j], 2)));     //ATPF eq.4.42, or see notes
                    u_v[j] = 1 / mu_v_Rad[j] * ((tau_i * r_i + 0.5*DpSeg_tot_*pow(r_i, 2))*log(De / 2 / r_v[j]) -
                        0.25*DpSeg_tot_*(pow(De / 2, 2) - pow(r_v[j], 2)));     //ATPF eq.4.42, or see notes
                }

            }


            //(viii-ix)
            ////////////////////////////heat transfer coefficient calculation
            //as from Buongiorno's analysis
            HH = 0;
            //film laminar region
            for (j = nRad; j >= 0; j--)
            {
                if (j == 0)
                {
                    htc[j] = De / 2 / r_v[j] / tau_v[j] * mu_v_Rad[j] / k_v_Rad[j] * (u_i - u_v[j]);
                }
                else
                {
                    htc[j] = De / 2 / r_v[j] / tau_v[j] * mu_v_Rad[j] / k_v_Rad[j] * (u_v[j - 1] - u_v[j]);
                }

                HH = HH + htc[j];
            }

            //film turbulent region
            //htc_rv = De / 2 / r_vt / tau_vt / cp_b * (u_i - u_vt);
            //htc_rv = De / 2 / r_vt / tau_vt * mu_b / k_b * (u_i - u_vt);
            //HH = HH + htc_rv;

            ////vapor core
            //htc_core = De / 2 / r_i / tau_i / cp_g * (u_g - u_i);
            //HH = HH + htc_core;

            htc_radial = 1 / HH;

            ////////////////////////////end of Buongiorno analysis

            break;

        }


        break;

    }

    ////////////////////////find f_g knowing delta_f////////////////////////////

    //while (TRUE)
    //{
    //  delta_f_new = (ff_i / f_g - 1) * De / 300;
    //  if (abs(delta_f_new - delta_f) < 0.0001)
    //  {
    //      break;
    //  }
    //  else if (delta_f_new < delta_f)
    //  {
```

```
764         //      f_g = f_g * 0.95;
765         //  }
766         //  else if (delta_f_new > delta_f)
767         //  {
768         //      f_g = f_g * 1.05;
769         //  }
770         //}
771
772         ////////////////////////////////////////////////////////////////////
773
774         ////////////////////////////WRITE ON OUTPUT FILE
775         //writing legend
776         radial_output_file << "VolSeg, OilSeg, NanoSeg, RefSegLIQ, RefSegVAP, RefSeg,
            delta_f, delta_v, delta_v_plus,"
777             "phi_b, phi_v, D_B, V_T, N_bt, mu_b, k_b, rho_b, cp_b, tau_i, tau_vt,"
778             "r_i, r_vt, q_i, q_vt, q_wall, u_g, u_f, u_i, u_vt, Re_f, Re_g, htc_tot,
                htc_nb, htc_radial" << endl;   //, t_g, t_i, t_vt" << endl;
779
780         radial_output_file << VolSeg << "," << OilSeg << "," << NanoSeg << "," << RefSegLIQ
            << "," << RefSegVAP << "," << RefSeg << ","
781             << delta_f << "," << delta_v << "," << delta_v_plus << "," << phi_b << "," <<
                phi_v << "," << D_B << "," << V_T << "," << N_bt << ","
782             << mu_b << "," << k_b << "," << rho_b << "," << cp_b << "," << tau_i << "," <<
                tau_vt << ","
783             << r_i << "," << r_vt << "," << q_i << "," << q_vt << "," << q_flux << ","
784             << u_g << "," << u_f << "," << u_i << "," << u_vt << "," << Re_f << "," << Re_g
                << ","
785             << htc_tot << "," << htc_nb << "," << htc_radial << endl << endl;       //<<
                t_g << "," << t_i << "," << t_vt << endl;
786
787         //writing legend
788         radial_output_file << "nRad, t_v_Rad[j], mu_v_Rad[j], k_v_Rad[j], rho_v_Rad[j],
            cp_v_Rad[j],"
789             "NMF_v_Rad[j], phi_v_Rad[j], r_v[j], tau_v[j], q_v[j], u_v[j], Pr_v_Rad[j],
                htc[j]" << endl;
790
791         for (j = 0; j <= nRad; j++)
792         {
793             radial_output_file << j << "," << t_v_Rad[j] << "," << mu_v_Rad[j] << "," <<
                k_v_Rad[j] << "," << rho_v_Rad[j] << "," << cp_v_Rad[j] << ","
794                 << NMF_v_Rad[j] << "," << phi_v_Rad[j] << "," << r_v[j] << "," << tau_v[j]
                    << "," << q_v[j] << "," << u_v[j] << "," << Pr_v_Rad[j] << ","
795                 << htc[j] << endl;
796         }
797
798         radial_output_file << endl;
799
800     }
801
```

```
1
2   #define _USE_MATH_DEFINES
3   #include <cmath>
4   #include "CoolProp.h"
5   #include "HumidAirProp.h"
6   #include <iostream>
7   #include <stdlib.h>
8   #include <fstream>
9   #include <string>
10  #include <cstddef>
11
12
13  using namespace CoolProp;
14  using namespace std;
15
16  void CalcProps(string, double, double, double, double, double,
17      string, string, string, double, double, double, double, double, double,
18      double&, double&, double&, double&, double&, double&, double&, double&, double&,
19      double&, double&, double&, double&, double&, double&, double&, double&, double&,
20      double&, double&, double&, double&, double&, double&, double&, double&, double&,
        double&);
21
22  void Inventory(string, string, double, double, double, double, double, double, double,
    double,
23      double&, double&, double&, double&, double&, double&, double&, double&, double&);
24
25  double OilDensity(double);
26  double OilConductivity(double);
27  double OilKinViscosity(double);
28  double NanoConductivty(string, double);
29  double NanoDensity(string);
30  double NanoSpecificHeat(string, double);
31  double NanoVolumeFraction(string, double, double);
32  double NanoOilMassFraction(double, double, double, double);
33  double LocalOilMassFraction(double, double);
34  double OilMixtureDensity(double, double, double);
35  double OilMixtureConductivity(double, double, double);
36  double OilMixtureSurfTension(double, double, double);
37  double PoolHTC_ForsterZuber(string, double, double);
38  double PoolHTC_JensenJackman(string, double, double, double);
39  double PoolHTC_Kedzierski2003(string, double, double, double, double, double, double);
40  double PoolHTC_Kedzierski2011(string, string, double, double, double, double,
41          double, double, double, double, double, double);
42  double PoolHTC_Kedzierski2012(string, string, double, double, double, double, double,
43          double, double, double, double, double, double);
44  double PoolHTC_PengDing2011(string, string, double, double, double, double, double,
45          double, double, double, double, double, double, double, double);
46  double PoolHTC_HuPeng2013(string, string, double, double, double, double, double,
47          double, double, double, double, double, double, double, double, double);
48  double RouhaniAxelsson(string, double, double, double, double, double);
49
50
51  double NanoOilMixtureDensity(string, double, double, double, double);
52  double NanoOilMixtureSpecificHeat(string, double, double, double, double);
53  double NanoOilMixtureDynViscosity(string, double, double, double, double, double);
54  double NanoOilMixtureConductivity(string, double, double, double, double);
55
56
57  double twoPhaseHTC_Hamilton2005(string fluid, double Dh, double OMF, double G_flux,
    double q_flux,
58      double t_sat, double p_in, double x_in, double mu_f, double cp_f, double k_f, double
         h_fg)
59  {
60      //use: calculate boiling heat transfer coefficient in a finned tube
61      //
62      //source: NIST - 2005 - Hamilton, Kedzierski, Kaul -
```

```
63        //        Horizontal Convective Boiling of Refrigerants and Refrigerants
64        //        Mixtures within a Micro-Fin Tube
65        //
66        //        NIST - 2007 - Sawant, Kedzierski, Brown -
67        //        Effect of Lubricant on R410A Horizontal Flow Boiling
68        //
69        //author: Andrea Bigi
70        //date: 11/2015
71        //-----------------------------------------------------------------
72
73        //local variables
74        double p_c;     //critical pressure, kPa
75        double Mw;      //molecular weight, g/mol
76        double Re_fo;   //Reynolds number, fluid only, -
77        double Pr_fo;   //Prandtl number, fluid only, -
78        double Bo;      //boiling number, -
79        double Nusselt; //Nusselt number, -
80        double C1, C2, C3, C4, C5, C6;  //coefficients for Nusselt correlation
81
82        //double h_f;
83        //double h_g;
84
85        //output variables
86        double htc;      //heat transfer coefficient, W/m2-C
87
88        //flow
89
90        ////!!!!!!!!!!!!!!!!!!!as by original paper, only viscosity and density account for
          presence of oil
91        //cp_f = PropsSI("C", "P", p_in * 1000, "Q", 0, fluid) / 1000;
92        //k_f = PropsSI("L", "P", p_in * 1000, "Q", 0, fluid);
93        //h_f = PropsSI("H", "P", p_in * 1000, "Q", 0, fluid) / 1000;
94        //h_g = PropsSI("H", "P", p_in * 1000, "Q", 1, fluid) / 1000;
95        //h_fg = h_g - h_f;
96        ////!!!!!!!!!!!!!!!!!!!
97
98        p_c = PropsSI("Pcrit", "T", t_sat + 273.15, "Q", 0, fluid) / 1000;
99        Mw = PropsSI("M", "T", t_sat + 273.15, "Q", 0, fluid) * 1000;
100
101       Re_fo = G_flux*Dh / mu_f;
102       Pr_fo = mu_f*cp_f*1000/k_f;
103       Bo = q_flux / (G_flux*h_fg*1000);
104
105       ////!!!!!!!!!!!!!!!!!!!
106       //C1 = 0.51*(x_in/(1 - OMF));
107       //C2 = 5.57*(x_in / (1 - OMF)) - 5.21*pow((x_in / (1 - OMF)), 2);
108       //C3 = 0.54 - 1.56*(x_in / (1 - OMF)) + 1.42*pow((x_in / (1 - OMF)), 2);
109       //C4 = -0.81 + 12.56*(x_in / (1 - OMF)) - 11 * pow((x_in / (1 - OMF)), 2);
110       //C5 = 0.25 - 0.035*pow((x_in / (1 - OMF)), 2);
111       ////!!!!!!!!!!!!!!!!!!!
112
113       C1 = 0.51*x_in;
114       C2 = 5.57*x_in - 5.21*pow(x_in, 2);
115       C3 = 0.54 - 1.56*x_in + 1.42*pow(x_in, 2);
116       C4 = -0.81 + 12.56*x_in - 11 * pow(x_in, 2);
117       C5 = 0.25 - 0.035*pow(x_in, 2);
118
119       if (OMF == 0)
120       {
121           C6 = 0; //C6 equal to zero in case of pure refrigerant
122       }
123       else
124       {
125           C6 = 0.15; //for R410A only, otherwise: (t_lv - t_mv)*(279.8*(x_v - x_l) - 4298
          * (t_d - t_b) / t_sat) / t_sat;
126       }
```

```
127
128         Nusselt = 482.18*pow(Re_fo, 0.3)*pow(Pr_fo, C1)*pow(p_in / p_c, C2)*
129             pow(Bo, C3)*pow(-log10(p_in / p_c), C4)*pow(Mw, C5)*pow(1.1, C6);
130
131         htc = Nusselt*k_f / Dh;
132
133         return htc;
134     }
135
136     double twoPhaseHTC_HuDing2008(string fluid, double D_f, double e_f, double l_f, double
    beta_f,
137                                   double G_flux, double q_flux, double t_sat, double p_in, double
                                       x_in,
138                                   double rho_f, double cp_f, double mu_f, double k_f, double h_fg)
139     {
140         //use: calculate boiling heat transfer coefficient in a finned tube
141         //
142         //source: Hu, H., G. Ding, et al. (2008). Heat transfer characteristics
143         //        of R410A-oil mixture flow boiling inside a 7 mm straight microfin tube.
144         //        International Journal of Refrigeration 31(6): 1081-1093.
145         //
146         //        Gungor, K. E. and R. H. S. Winterton (1986).
147         //        A general correlation for flow boiling in tubes and annuli.
148         //        International Journal of Heat and Mass Transfer 29(3): 351-358.
149         //
150         //        Ravigururajan, T.S., Bergles, A.E., (1985). General correlations for
151         //        pressure drop and heat transfer for single - phase turbulent flow in
        internally ribbed tubes.
152         //        Augmentation of Heat Transfer in Energy Systems 52, 9–20.
153         //
154         //author: Andrea Bigi
155         //date: 04/2016
156         //-----------------------------------------------------------------
157
158         //local variables
159         double p_c;     //critical pressure, kPa
160         double p_r;     //reduced pressure, -
161         double Mw;      //molecular weight, g/mol
162         double rho_g;   //saturated vapor density, kg/m3
163         double mu_g;    //saturated vapor dynamic viscosity, N-s/m2, kg/m-s, Pa-s
164         double Xtt;     //Martinelli parameter
165         double Bo;      //boiling number, -
166         double Re_fo;   //Reynolds number, fluid only, -
167         double Pr_fo;   //Prandtl number, fluid only, -
168         double E;       //two-phase convection multiplier
169         double S;       //boiling suppression factor
170         double E_rb;    //ribbed tube enhancement factor
171         double alfa_DB; //Dittus_Boelter flow boiling heat transfer coefficient, W/m2-C
172         double alfa_l;  //liquid component heat transfer coefficient, W/m2-C
173         double alfa_nb; //nucleate boling heat transfer coefficient, W/m2-C
174         double a, b, c, d; //correlation parameters
175
176         double h_f;
177         double h_g;
178
179         //output variables
180         double htc;      //heat transfer coefficient, W/m2-C
181
182         //flow
183
184         a = 33686.87;
185         b = 1.169;
186         c = 2.53e-6;
187         d = 1.489;
188
189         ////!!!!!!!!!!!!!!!!!!
```

```
190         //h_f = PropsSI("H", "P", p_in * 1000, "Q", 0, fluid) / 1000;
191         //h_g = PropsSI("H", "P", p_in * 1000, "Q", 1, fluid) / 1000;
192         //h_fg = h_g - h_f;
193         ////!!!!!!!!!!!!!!!!!!!
194
195         p_c = PropsSI("Pcrit", "T", t_sat + 273.15, "Q", 0, fluid) / 1000;
196         Mw = PropsSI("M", "T", t_sat + 273.15, "Q", 0, fluid) * 1000;
197         rho_g = PropsSI("D", "T", t_sat + 273.15, "Q", 1, fluid);
198         mu_g = PropsSI("V", "T", t_sat + 273.15, "Q", 1, fluid);
199
200         p_r = p_in / p_c;
201
202         Xtt = pow((1 - x_in) / x_in, 0.9) * pow(rho_g / rho_f, 0.5) * pow(mu_f / mu_g, 0.1);
               //this is the correct formula
203         Bo = q_flux / (G_flux*h_fg*1000);
204         Re_fo = G_flux * (1-x_in) * D_f / mu_f;
205         Pr_fo = mu_f*cp_f*1000 / k_f;
206
207         E = 1 + a * pow(Bo, 1.16) + b * pow(Xtt, -0.86);
208         S = 1 / (1 + c * pow(E, d) * pow(Re_fo, 1.17));   //this is the correct formula,
            reported wrong on Hu(2008). See Gungor(1986) for correct one
209
210         E_rb = pow(1 + pow((2.64 * pow(Re_fo, 0.036) * pow(Pr_fo, 0.024) *     //this is
            the correct formula, reported wrong on Hu(2008). See Ravigururajan(1985) for
            correct one
211                 pow(e_f / D_f, 0.212) * pow(l_f / D_f, -0.21) * pow(beta_f / 90, 0.29)), 7),
                    1/7);
212         alfa_DB = 0.023 * (k_f/D_f) * pow(Re_fo, 0.8) * pow(Pr_fo, 0.4);
213
214         alfa_l = E_rb * alfa_DB;
215         alfa_nb = 55 * pow(p_r, 0.12) * pow(-log10(p_r), -0.55) * pow(Mw, -0.5) * pow(q_flux
            , 0.67);  //Cooper correlation (1984)
216
217         htc = E * alfa_l + S * alfa_nb;
218
219         return htc;
220     }
221
222     double twoPhaseHTC_Zou2010(string fluid, double Dh, double G_flux, double q_flux, double
         t_sat, double p_in, double x_in,
223         double rho_f, double cp_f, double mu_f, double k_f)
224     {
225         //use: calculate boiling heat transfer coefficient in a smooth tube
226         //
227         //source: X. Zou, M.Q. Gong et al. (2010). Experimental study on saturated flow
            boiling
228         //        heat transfer of R170/R290 mixtures in a horizontal tube.
229         //        International Journal of Refrigeration 33(2): 371-380.
230         //
231         //author: Andrea Bigi
232         //date: 02/2017
233         //-----------------------------------------------------------------
234
235         //local variables
236         double p_c;     //critical pressure, kPa
237         double p_r;     //reduced pressure, -
238         double Mw;      //molecular weight, g/mol
239         double rho_g;   //saturated vapor density, kg/m3
240         double Re_fo;   //Reynolds number, fluid only, -
241         double Pr_fo;   //Prandtl number, fluid only, -
242         double E;       //two-phase convection multiplier
243         double S;       //boiling suppression factor
244         double alfa_DB; //Dittus_Boelter forced convection heat transfer coefficient, W/m2-C
245         double alfa_nb; //nucleate boling heat transfer coefficient, W/m2-C
246
247         //output variables
```

```
248         double htc;      //heat transfer coefficient, W/m2-C
249
250         //flow
251
252         p_c = PropsSI("Pcrit", "T", t_sat + 273.15, "Q", 0, fluid) / 1000;
253         Mw = PropsSI("M", "T", t_sat + 273.15, "Q", 0, fluid) * 1000;
254         rho_g = PropsSI("D", "T", t_sat + 273.15, "Q", 1, fluid);
255
256         p_r = p_in / p_c;
257
258         Re_fo = G_flux * Dh / mu_f;  //all liquid Reynolds number
259         Pr_fo = mu_f*cp_f * 1000 / k_f;
260
261         E = pow(1 + x_in*Pr_fo*(rho_f / rho_g - 1), 0.35);
262         S = 1 / (1 + 0.055*pow(E, 0.1) * pow(Re_fo, 0.16));
263
264         alfa_DB = 0.023 * (k_f / Dh) * pow(Re_fo, 0.8) * pow(Pr_fo, 0.4);
265         alfa_nb = 55 * pow(p_r, 0.12) * pow(-log10(p_r), -0.55) * pow(Mw, -0.5) * pow(q_flux
        , 0.67);  //Cooper correlation (1984)
266
267         htc = sqrt(pow(E * alfa_DB, 2) + pow(S * alfa_nb, 2));
268
269         return htc;
270     }
271
272     double twoPhaseHTC_Sawant2012(string fluid, string nano_mater, double D_nano, double OMF
        , double NMF, double n_np_Seg, double SurfASeg,
273         double Dh, double G_flux, double q_flux, double t_sat, double t_wall, double p_in,
            double x_in,
274         double rho_f, double cp_f, double mu_f, double k_f, double sigma_f, double h_fg)
275     {
276         //use: calculate boiling heat transfer coefficient in a finned tube with
            lubricant-refrigerant mixture
277         //
278         //source: Sawant (2012). Influence of Lubricant on Horizontal Convective Boiling in
            a Micro-fin Tube,
279         //        The Catholic University of America, Ph.D. thesis
280         //
281         //author: Andrea Bigi
282         //date: 02/2017
283         //-----------------------------------------------------------------
284
285         //local variables
286         double rho_g;   //saturated vapor density, kg/m3
287         double mu_g;    //saturated vapor dynamic viscosity, N-s/m2, kg/m-s, Pa-s
288         double Xtt;     //Martinelli parameter
289         double Re_fo;   //Reynolds number, fluid only, -
290         double Pr_fo;   //Prandtl number, fluid only, -
291         double E;       //two-phase convection multiplier
292         double S;       //boiling suppression factor
293         double alfa_c;  //convective heat transfer coefficient, W/m2-C
294         double alfa_nb; //nucleate boiling heat transfer coefficient, W/m2-C
295         double phi;     //nanoparticle volume fraction
296         double A0, A1;  //correlation parameters
297
298         double Bo;      //Boiling number, -
299         double Nu;      //NUssetlt Number, -
300         double p_c;     //critical pressure, kPa
301         double t_c;     //critical temperature, K
302         double p_r;     //reduced pressure, -
303         double t_r;     //reduced temperature, -
304         double Mw;      //molecular weight, g/mol
305
306         //output variables
307         double htc;      //heat transfer coefficient, W/m2-C
308
```

```
309         //flow
310 
311         A0 = 1;     //variable initialization
312         A1 = 1;     //variable initialization
313 
314         rho_g = PropsSI("D", "T", t_sat + 273.15, "Q", 1, fluid);
315         mu_g = PropsSI("V", "T", t_sat + 273.15, "Q", 1, fluid);
316 
317         Xtt = pow((1 - x_in) / x_in, 0.9) * pow(rho_g / rho_f, 0.5) * pow(mu_f / mu_g, 0.1);
318         Re_fo = G_flux * (1 - x_in) * Dh / mu_f;      //liquid phase Reynolds number
319 
320         Pr_fo = mu_f*cp_f * 1000 / k_f;
321 
322         alfa_c = 0.023 * (k_f / Dh) * pow(Re_fo, 0.8) * pow(Pr_fo, 0.4);
            //Dittus-Boelter correlation
323 
324         alfa_nb = PoolHTC_JensenJackman("R134a", 0.5, 4, 10);
325         alfa_nb = PoolHTC_Kedzierski2003(fluid, OMF, t_sat, t_wall, x_in, h_fg, q_flux);
326 
327         if (NMF > 0)
328         {
329             //calculation of the volume concentration in water
330             phi = NanoVolumeFraction(nano_mater, NMF, t_sat);
331 
332             //alfa_nb = PoolHTC_Kedzierski2011(fluid, nano_mater, D_nano, OMF, phi, t_sat,
333             //  t_wall, x_in, k_f, rho_f, mu_f, sigma_f, h_fg, q_flux);
334             alfa_nb = PoolHTC_Kedzierski2012(fluid, nano_mater, D_nano, OMF, phi, n_np_Seg,
                SurfASeg, t_sat,
335                 t_wall, x_in, sigma_f, h_fg, q_flux);
336         }
337 
338 
339         p_c = PropsSI("Pcrit", "T", t_sat + 273.15, "Q", 0, fluid) / 1000;
340         t_c = PropsSI("Tcrit", "T", t_sat + 273.15, "Q", 0, fluid);
341         Mw = PropsSI("M", "T", t_sat + 273.15, "Q", 0, fluid) * 1000;
342         rho_g = PropsSI("D", "T", t_sat + 273.15, "Q", 1, fluid);
343 
344         p_r = p_in / p_c;
345         t_r = (t_sat + 273.15)/t_c;
346         Re_fo = G_flux * Dh / mu_f;  //all liquid Reynolds number
347         Bo = q_flux / (G_flux*h_fg*1000);
348 
349 
350         A0 = 0.00132;
351         A1 = 1;
352         S = exp(-A0*Re_fo*x_in);
353         E = A1 / (Xtt*Re_fo*x_in);
354 
355         htc = 13.7 * S * alfa_nb + 1.685e13 * pow(E, 4.419) * alfa_c;
356 
357         Nu = htc * Dh / k_f;
358 
359         return htc;
360     }
361 
362     double PoolHTC_ForsterZuber(string fluid, double t_sat, double t_wall)
363     {
364         //use: calculate pool boiling heat transfer coefficient for refrigerant/lubricant
            mixture
365         //
366         //source: Forster, Zuber. (1955). Dynamic of Vapor Bubbles and Boiling Heat
            Transfer,
367         //          AIChE Journal, Vol.1 (4): 531-535
368         //
369         //          Jensen, Jackman. (1984). Prediction of Nucleate Pool Boiling Heat
            Transfer Coefficients
```

```
370       //              of Refrigerant-Oil Mixtures, Transactions of the ASME Vol.106:
          184-190
371       //
372       //author: Andrea Bigi
373       //date: 11/2017
374       //--------------------------------------------------------------------
375
376       //local variables
377       double rho_g;   //saturated vapor density, kg/m3
378       double nu_f;    //saturated liquid kinematik viscosity, m2/s
379       double rho_f;   //saturated liquid density, kg/m3
380       double mu_f;    //saturated liquid dynamic viscosity, N-s/m2, kg/m-s, Pa-s
381       double cp_f;    //saturated liquid specific heat, kJ/kg-K
382       double k_f;     //saturated liquid thermal conductivity, W/m-C
383       double sigma_f; //saturated liquid surface tension, N/m
384       double h_fg;    //vaporization enthalpy, kJ/kg
385       double h_f;     //saturated liquid enthalpy, kJ/kg
386       double h_g;     //saturated vapor enthalpy, kJ/kg
387       double DT_s;    //wall superheat Twall-ETXTsat, C
388       double DP_s;    //p_wall-ETXp_sat, Pa
389       double p_wall;  //saturation pressure at t_wall, Pa
390       double p_sat;   //saturation pressure, Pa
391
392       //output variables
393       double htc;      //heat transfer coefficient, W/m2-C
394
395       //flow
396
397       rho_g = PropsSI("D", "T", t_sat + 273.15, "Q", 1, fluid);
398       rho_f = PropsSI("D", "T", t_sat + 273.15, "Q", 0, fluid);
399       mu_f = PropsSI("V", "T", t_sat + 273.15, "Q", 0, fluid);
400       nu_f = mu_f / rho_f;
401       cp_f = PropsSI("C", "T", t_sat + 273.15, "Q", 0, fluid);
402       k_f = PropsSI("L", "T", t_sat + 273.15, "Q", 0, fluid);
403       sigma_f = PropsSI("I", "T", t_sat + 273.15, "Q", 0, fluid);
404       h_f = PropsSI("H", "T", t_sat + 273.15, "Q", 0, fluid);
405       h_g = PropsSI("H", "T", t_sat + 273.15, "Q", 1, fluid);
406       h_fg = h_g - h_f;
407
408       p_wall = PropsSI("P", "T", t_wall + 273.15, "Q", 0, fluid);
409       p_sat = PropsSI("P", "T", t_sat + 273.15, "Q", 0, fluid);
410
411       DT_s = t_wall - t_sat;
412       DP_s = p_wall - p_sat;
413
414       htc = 0.00122*((pow(k_f, 0.79)*pow(cp_f, 0.45)*pow(rho_f, 0.49))
415           / (pow(sigma_f, 0.5)*pow(mu_f, 0.29)*pow(h_fg, 0.24)*pow(rho_g, 0.24)))
416           *pow(DT_s, 0.24)*pow(DP_s, 0.75);
417
418       return htc;
419   }
420
421   double PoolHTC_JensenJackman(string fluid, double OMF, double t_sat, double t_wall)
422   {
423       //use: calculate pool boiling heat transfer coefficient for refrigerant/lubricant
          mixture
424       //
425       //source: Jensen, Jackman. (1984). Prediction of Nucleate Pool Boiling Heat
          Transfer Coefficients
426       //          of Refrigerant-Oil Mixtures, Transactions of the ASME Vol.106: 184-190
427       //
428       //author: Andrea Bigi
429       //date: 10/2017
430       //--------------------------------------------------------------------
431
432       //local variables
```

```
433       double rho_g;   //saturated vapor density, kg/m3
434       double nu_f;    //saturated liquid kinematik viscosity, m2/s
435       double rho_f;   //saturated liquid density, kg/m3
436       double mu_f;    //saturated liquid dynamic viscosity, N-s/m2, kg/m-s, Pa-s
437       double cp_f;    //saturated liquid specific heat, kJ/kg-K
438       double k_f;     //saturated liquid thermal conductivity, W/m-C
439       double sigma_f; //saturated liquid surface tension, N/m
440       double h_fg;    //vaporization enthalpy, kJ/kg
441       double h_f;     //saturated liquid enthalpy, kJ/kg
442       double h_g;     //saturated vapor enthalpy, kJ/kg
443       double OMF_eff; //effective oil concentration
444       double DT_s;    //wall superheat Twall-ETXTsat, C
445       double DP_s;    //p_wall-ETXp_sat, Pa
446       double p_wall;  //saturation pressure at t_wall, Pa
447       double p_sat;   //saturation pressure, Pa
448       double htc_ForsterZuber;    //Forster and Zuber (1955) pool boiling htc
449
450       //output variables
451       double htc;      //heat transfer coefficient, W/m2-C
452
453       //flow
454
455       //Forster Zuber correlation according to Chen (1962)
456       htc_ForsterZuber = PoolHTC_ForsterZuber(fluid, t_sat, t_wall);
457
458       DT_s = t_wall - t_sat;
459       OMF_eff = OMF*(1 + 0.0317*pow(DT_s, 0.753));
460
461       htc = htc_ForsterZuber * exp(-4.095*OMF_eff - 55.11*pow(OMF_eff, 2));
462
463       return htc;
464   }
465
466
467   double PoolHTC_Kedzierski2003(string fluid, double OMF, double t_sat, double t_wall,
      double x_in, double h_fg, double q_flux)
468   {
469       //use: calculate pool boiling heat transfer coefficient for refrigerant/lubricant
          mixture
470       //
471       //source: [1] Kedzierski, M. (2003). A semi-theoretical model for predicting
          refrigerant/lubricant mixture
472       //         pool boiling heat transfer, International Journal of Refrigeration 26:
          337-348
473       //
474       //      [2] Kedzierski, M. (2003). Improved thermal boundary layer parameter for
          semi-theoretical
475       //          refrigerant-lubricant pool boiling model, International Congress of
          Refrigeration,
476       //          Washington, D.C., ICR0504
477       //
478       //          Geller, V.Z., Lapardin, N.I. (2016). SOLUBILITY AND MISCIBILITY OF
          REFRIGERANTS R407C AND R410A WITH SYNTHETIC
479       //          COMPRESSOR OILS, Refrigeration technology and technology, 52 (3), UDC
          532.739.2; 536.423.15; 532.77-2; 536.444; 532.133
480       //
481       //author: Andrea Bigi
482       //date: 02/2017
483       //-----------------------------------------------------------------
484
485       //local variables
486       double rho_g;   //saturated vapor density, kg/m3
487       double nu_f;    //saturated liquid kinematik viscosity, m2/s
488       double rho_f;   //saturated liquid density, kg/m3
489       double mu_f;    //saturated liquid dynamic viscosity, N-s/m2, kg/m-s, Pa-s
490       double sigma_f; //saturated liquid surface tension, N/m
```

```
491         double rho_b;   //bulk liquid density, kg/m3
492         double sigma_b; //bulk liquid surface tension, N/m
493         double rho_oil; //oil density, kg/m3
494         double k_oil;   //oil thermal conductivity, W/m-C
495         double nu_oil;  //oil kinematic viscosity, mm2/s (cSt)
496         double w_local; //local oil mass fraction
497         double qm_qp;   //ratio of the refrigerant/lubricant heat flux (qm) to that of the
        pure refrigerant (qp)
498         double DT_s;    //wall superheat Twall-ETXTsat, C
499         double t_c;     //refrigerant/lubricant lower critical solution temperature (LCST), C
500         double lambda;  //thermal boundary constant
501         double gamma;   //oil excess surface density, kg/m2
502         double l_e;     //thickness of excess layer, m
503         double r_b;     //bubble departure radius, m
504         double A0;      //constant
505
506         //output variables
507         double htc;      //heat transfer coefficient, W/m2-C
508
509         //flow
510
511         rho_g = PropsSI("D", "T", t_sat + 273.15, "Q", 1, fluid);
512         rho_f = PropsSI("D", "T", 40 + 273.15, "Q", 0, fluid);
513         mu_f = PropsSI("V", "T", 40 + 273.15, "Q", 0, fluid);
514         nu_f = mu_f / rho_f;
515         sigma_f = PropsSI("I", "T", t_sat + 273.15, "Q", 0, fluid);
516
517         rho_oil = OilDensity(t_sat);
518         k_oil = OilConductivity(t_sat);
519         nu_oil = OilKinViscosity(40) / 1000000;
520
521         w_local = LocalOilMassFraction(OMF, x_in);
522
523         rho_b = OilMixtureDensity(t_sat, w_local, rho_f);
524         sigma_b = OilMixtureSurfTension(t_sat, w_local, sigma_f);
525
526         t_c = -40;  //Geller, Lapardin; Kedzierski [1], not sure LCST = -40C
527         //t_c = -3.15;  //Kedzierski [2]
528
529         DT_s = t_wall - t_sat;
530         r_b = 18.75e-10 * rho_oil * (1 - OMF) / (OMF * rho_g);
531
532         l_e = OMF*(t_sat + 273.15)*sigma_b / (5.9e-7*(1 - OMF)*rho_oil*h_fg*1000*DT_s);
533         gamma = l_e * (rho_oil - rho_b * OMF);
534
535         //qm_qp = 1.25 - OMF*(91.9 - ((nu_oil - nu_f) / nu_f)*(0.529 - 1.92*((t_sat - t_c)
        / (t_sat + 273.15)))
536         //  - 211 * ((t_sat - t_c) / (t_sat + 273.15)));
537         qm_qp = 1.27 - OMF*(99.1 - ((nu_oil - nu_f) / nu_f)*(0.578 - 2.09*((t_sat - t_c) / (
        t_sat + 273.15)))
538             - 226 * ((t_sat - t_c) / (t_sat + 273.15)));
539
540         //Kedzierski [1]
541         lambda = 0.27 + 10700 * r_b * qm_qp;
542
543         ////Kedzierski [2]
544         //A0 = 5e-5;    //(q_flux * OMF) / (gamma * h_fg * 1000 * (1 - OMF) * exp(DT_s));
545         //lambda = (OMF*pow(gamma, 2)*h_fg * 1000 / l_e) /
546         //  ((k_oil*DT_s*(rho_f - rho_b * OMF)*OMF) / (A0 * exp(DT_s) * (1 - OMF)) - 0.62 *
        pow(gamma, 2) * h_fg * 1000);
547
548         htc = 5.9e-7*(1 - OMF)*rho_oil*h_fg*1000*DT_s*k_oil*(1 - exp(-lambda*l_e / r_b)) / (
        OMF*(t_sat + 273.15)*sigma_b);
549
550         return htc;
551     }
```

```
552
553
554 double PoolHTC_Kedzierski2011(string fluid, string nano_mater, double D_nano, double OMF
    , double phi,
555     double t_sat, double t_wall, double x_in, double sigma_f, double h_fg, double q_flux)
556 {
557     //use: calculate pool boiling heat transfer coefficient for
        refrigerant/nanolubricant mixture
558     //
559     //source: Kedzierski, M. (2011). Effect of Al2O3 nanolubricant on R134a pool
        boiling heat
560     //          transfer, International Journal of Refrigeration (34): 348-508
561     //
562     //author: Andrea Bigi
563     //date: 02/2017
564     //---------------------------------------------------------------------
565
566     //local variables
567     double rho_g;   //saturated vapor density, kg/m3
568     double rho_oil; //oil density, kg/m3
569     double k_oil;   //oil thermal conductivity, W/m-C
570     double nu_oil;  //oil kinematic viscosity, mm2/s (cSt)
571     double rho_nano;//nanoparticle density, kg/m3
572     double DT_s;    //wall superheat Twall-ETXTsat, C
573     double qnp;     //refrigerant/nanolubricant mixture heat flux, W/m2
574     double qPL;     //refrigerant/pure lubricant mixture heat flux, W/m2
575     double qnp_qPL; //ratio of the refrigerant/nanolubricant heat flux (qnp) to that of
        refrigerant/pure lubricant (qpL)
576
577     //output variables
578     double htc;      //heat transfer coefficient, W/m2-C
579
580     //flow
581
582     rho_g = PropsSI("D", "T", t_sat + 273.15, "Q", 1, fluid);
583
584     rho_oil = OilDensity(t_sat);
585     k_oil = OilConductivity(t_sat);
586     nu_oil = OilKinViscosity(t_sat) / 1000000;
587
588     rho_nano = NanoDensity(nano_mater);
589
590     DT_s = t_wall - t_sat;
591     D_nano = D_nano*1e-9;   //from nm to m
592
593     qPL = DT_s * PoolHTC_Kedzierski2003(fluid, OMF, t_sat, t_wall, x_in, h_fg, q_flux);
594
595     qnp_qPL = 1 + (3.45e-9*phi*sigma_f*nu_oil*rho_g*pow(OMF, 2)) / (pow(D_nano,4)*pow(
        qPL,3/2)*rho_oil*(rho_nano-rho_oil)*9.81*pow((1-OMF),2));
596
597     qnp = qnp_qPL*qPL;
598
599     htc = qnp / DT_s;
600
601     return htc;
602 }
603
604
605 double PoolHTC_Kedzierski2012(string fluid, string nano_mater, double D_nano, double OMF
    , double phi, double n_np_Seg, double SurfASeg,
606     double t_sat, double t_wall, double x_in, double sigma_f, double h_fg, double q_flux)
607 {
608     //use: calculate pool boiling heat transfer coefficient for
        refrigerant/nanolubricant mixture on a finned surface
609     //
610     //source: Kedzierski, M. (2012). R134a/Al2O3 Nanolubricant Mixture Pool Boiling on a
```

```
611         //          Rectangular Finned Surface, ASME Vol. 134
612         //
613         //author: Andrea Bigi
614         //date: 02/2017
615         //-----------------------------------------------------------------------
616
617         //local variables
618         double rho_g;   //saturated vapor density, kg/m3
619         double rho_oil; //oil density, kg/m3
620         double k_oil;   //oil thermal conductivity, W/m-C
621         double nu_oil;  //oil kinematic viscosity, mm2/s (cSt)
622         double rho_nano;//nanoparticle density, kg/m3
623         double DT_s;    //wall superheat Twall-[illegible]Tsat, C
624         double qnp;     //refrigerant/nanolubricant mixture heat flux, W/m2
625         double qPL;     //refrigerant/pure lubricant mixture heat flux, W/m2
626         double qnp_qPL; //ratio of the refrigerant/nanolubricant heat flux (qnp) to that of
            refrigerant/pure lubricant (qpL)
627         double Nnp_As;  //nanoparticle surface density, 1/m2
628         double Nnp_As_G;//geometry dependent nanoparticle surface density, 1/m2
629
630         //output variables
631         double htc;      //heat transfer coefficient, W/m2-C
632
633         //flow
634
635         rho_g = PropsSI("D", "T", t_sat + 273.15, "Q", 1, fluid);
636
637         rho_oil = OilDensity(t_sat);
638         k_oil = OilConductivity(t_sat);
639         nu_oil = OilKinViscosity(t_sat) / 1000000;
640
641         rho_nano = NanoDensity(nano_mater);
642
643         DT_s = t_wall - t_sat;
644         D_nano = D_nano*1e-9;   //from nm to m
645
646         qPL = DT_s * PoolHTC_Kedzierski2003(fluid, OMF, t_sat, t_wall, x_in, h_fg, q_flux);
647
648         Nnp_As = n_np_Seg / SurfASeg;
649         Nnp_As_G = 4.15e8 * pow(qPL, 2.53)*pow(Nnp_As*1e-20, 1.47);
650
651         qnp_qPL = 1 + (3.45e-9*phi*sigma_f*nu_oil*rho_g*pow(OMF, 2)) / (pow(D_nano, 4)*pow(
            qPL, 3 / 2)*rho_oil*(rho_nano - rho_oil)*9.81*pow((1 - OMF), 2));
652         qnp_qPL = 1 + (1.45e-9*Nnp_As_G*sigma_f*nu_oil*rho_g*OMF) / (D_nano*pow(qPL, 3 / 2)*
            rho_oil*(rho_nano - rho_oil)*9.81*pow((1 - OMF), 2));
653
654         qnp = qnp_qPL*qPL;
655
656         htc = qnp / DT_s;
657
658         return htc;
659     }
660
661
662     double PoolHTC_PengDing2011(string fluid, string nano_mater, double D_nano, double NMF,
        double OMF, double phi, double q_flux,
663         double t_sat, double x_in, double k_f, double rho_f, double cp_f, double mu_f,
            double sigma_f, double h_fg)
664     {
665         //use: calculate pool boiling heat transfer coefficient for refrigerant-based
            nanofluids, with surfactant additives
666         //
667         //source: Peng, H., Ding, G., Hu, H. (2011). Effect of surfactant additives on
            nucleate pool boiling heat transfer
668         //           of refrigerant-based nanofluid, Experimental Thermal and Fluid
            Sciences (35): 960-970
```

```
669         //
670         //author: Andrea Bigi
671         //date: 02/2017
672         //---------------------------------------------------------------------
673
674         //local variables
675         double rho_g;   //saturated vapor density, kg/m3
676         double nu_f;    //saturated liquid kinematik viscosity, m2/s
677         double alpha_f; //saturated liquid thermal diffusivity, m2/s
678         double rho_oil; //oil density, kg/m3
679         double k_nano;  //nanoparticle thermal conductivity, W/m-K
680         double rho_nano;//nanoparticle density, kg/m3
681         double cp_nano; //nanoparticle specific heat, J/kg-K
682         double r_b;     //bubble departure radius, m
683         double C_surf;  //surfactant concentration
684         double M_surf;  //surfactant molecular weight, g/mol
685         double m1, m2, m3, m4, m5, n1, n2;   //correlation fitted coefficients
686         double SER;     //surfactant enhancement ratio
687         double NER;     //nanoparticle enhancement ratio
688
689
690         //output variables
691         double htc;      //heat transfer coefficient, W/m2-C
692
693         //flow
694
695         m1 = -2691;
696         m2 = 27.1;
697         m3 = 3517;
698         m4 = 0.5;
699         m5 = -1290;
700         n1 = 0.69;
701         n2 = 0.25;
702
703         rho_g = PropsSI("D", "T", t_sat + 273.15, "Q", 1, fluid);
704         nu_f = mu_f / rho_f; // m2/s
705         alpha_f = k_f/(rho_f*cp_f*1000);
706         rho_oil = OilDensity(t_sat);
707         k_nano = NanoConductivty(nano_mater, t_sat);
708         rho_nano = NanoDensity(nano_mater);
709         cp_nano = NanoSpecificHeat(nano_mater, t_sat)*1000;
710
711         C_surf = 1;   //!!!
712         M_surf = 1;   //!!!
713         r_b = 0.0146 * 35 * sqrt(2*sigma_f/(9.81*(rho_f-rho_g)));
714         //r_b = 18.75e-10 * rho_oil * (1 - OMF) / (OMF * rho_g);    //Kedzierski2003
715
716         SER = exp((m1*pow(C_surf, 2)+m2*C_surf)*m3/pow(q_flux*M_surf*NMF, n1));
717         NER = 1 + pow(NMF, n2)*(m4*k_nano / k_f + m5*(rho_nano*cp_nano) / (rho_f*cp_f));
718         //Stephan and Abdelsalam correlation (1980)
719         htc = 207 * k_f / r_b * pow(q_flux*r_b / (k_f*t_sat), 0.745) * pow(rho_g / rho_f,
            0.581) * pow(nu_f / alpha_f, 0.533);
720
721         htc = SER*NER*htc;
722
723         return htc;
724     }
725
726
727     double PoolHTC_HuPeng2013(string fluid, string nano_mater, double D_nano, double NMF,
        double OMF, double phi, double q_flux,
728         double m_dot_fluid, double t_sat, double x_in, double k_f, double rho_f, double cp_f
            , double mu_f, double sigma_f, double h_fg)
729     {
730         //use: calculate pool boiling heat transfer coefficient for refrigerant-based
            nanofluids, with surfactant additives
```

```
731         //
732         //source: Hu, H., Peng, H., Ding, G. (2013). Nucleate pool boiling heat transfer
            characteristics of
733         //          refrigerant / nanolubricant mixture with surfactant, International
            Journal of Refrigeration (36): 1045-1055
734         //
735         //        Peng, H., Ding, G., Hu, H., Jiang, W. (2011) Effect of nanoparticle size
            on nucleate pool boiling heat transfer
736         //          of refrigerant / oil mixture with nanoparticles, International Journal
            of Heat and Mass Transfer (54): 1839–1850
737         //
738         //        Peng, H., Ding, G., Hu, H. et al. (2010) Nucleate pool boiling heat
            transfer characteristics of
739         //          refrigerant / oil mixture with diamond nanoparticles, International
            Journal of Refrigeration (33): 347-358
740         //
741         //author: Andrea Bigi
742         //date: 03/2017
743         //-----------------------------------------------------------------
744
745         //local variables
746         double rho_g;   //saturated vapor density, kg/m3
747         double rho_oil; //oil density, kg/m3
748         double OMFnano; //nanolubricant mass fraction in refrigerant AND nanolubricant
            mixture
749         double C_surf;  //surfactant concentration
750         double M_surf;  //surfactant molecular weight, g/mol
751         double DT;      //temperature difference, C
752         double Csf;     //Rohsenow coefficient
753         double n;       //Rohsenow coefficient
754         double D0;      //benchmark nanoparticle size, nm
755         double a, b, c, d;      //correlation fitted coefficients
756         double m1, m2, m3, n1;  //correlation fitted coefficients
757         double SIF;     //surfactant impact factor
758
759
760         //output variables
761         double htc;      //heat transfer coefficient, W/m2-C
762
763         //flow
764
765         D0 = 100;
766         a = 0.0093;
767         b = 0.00356;
768         c = -0.0048;
769         d = 0.0025;
770
771         m1 = -1395;
772         m2 = 14;
773         m3 = 2400;
774         n1 = 0.48;
775
776         rho_g = PropsSI("D", "T", t_sat + 273.15, "Q", 1, fluid);
777         rho_oil = OilDensity(t_sat);
778         OMFnano = NanoOilMassFraction(OMF, NMF, m_dot_fluid, t_sat);
779
780         C_surf = 1;   //!!!
781         M_surf = 1;   //!!!
782
783         //Rohsenow correlation (1952)
784         Csf = a + b*D_nano / D0 + c*OMFnano + d*NMF;
785         n = 1.3068;
786         DT = (h_fg*Csf / cp_f*pow(q_flux / (mu_f*h_fg)*sqrt(sigma_f / (9.81*(rho_f - rho_g
            ))), 0.33)*pow(cp_f * 1000 * mu_f / k_f, n));
787         htc = q_flux / DT;
788
```

```
789         SIF = exp((m1*pow(C_surf, 2) + m2*C_surf)*m3 / pow(q_flux*M_surf*NMF, n1));
790
791         htc = SIF*htc;
792
793         return htc;
794     }
795
796
797     double Buongiorno2006(double nRad, int index_Nbt, ofstream &Buongiorno_output_file,
        ofstream &Buongiorno_radial_output_file,
798         string fluid, double A0, double A1, double A2,  double a0, double b0, string oil,
            string nano_mater, string nano_shape,
799         double D_nano, double ff, double OMF, double NMF, double n_np_Seg, double
            m_dot_fluid, double p_in, double h_in,
800         double t_in, double x_in, double rho_in, double v_in, double cp_in, double mu_in,
            double k_in, double sigma_in,
801         double t_sat_f, double h_f, double rho_f, double v_f, double cp_f, double mu_f,
            double k_f, double sigma_f,
802         double t_sat_g, double h_g, double rho_g, double v_g, double cp_g, double mu_g,
            double k_g, double sigma_g, double h_fg,
803         double dh_x, double t_sat, double w_local, double Dh, double De, double SectA,
            double lengthSeg, double t_wall,
804         double G_flux, double q_flux, double delta_f, double &N_bt, double &phi)
805     {
806         //use: calculate single phase convective heat transfer in nanofluids
807         //
808         //source: Buongiorno, J. (2006). Convective transport in nanofluids, Transactions
            of the ASME (128): 240-250
809         //
810         //author: Andrea Bigi
811         //date: 10/2017
812         //-----------------------------------------------------------------
813
814         //local variables
815
816         //inventory
817         double epsilon;     //void fraction,
818         double VolSeg;      //segment volume, m3
819         double OilSeg;      //oil mass per segment, kg
820         double OilNanoSeg;  //nanooil mass per segment, kg
821         double NanoSeg;     //nanoparticles mass per segment, kg
822         double RefSeg;      //total refrigerant mass per segment, kg
823         double RefSegLIQ;   //liquid refrigerant mass per segment, kg
824         double RefSegVAP;   //vapor refrigerant mass per segment, kg
825         double NanoRad;     //nanoparticles mass, inside each radius of the laminar
            sublayer, kg
826         double V_nano;      //nanopartcile volume, m3
827         double N_nano_Seg;  //number of nanoparticles inside the segment, -
828
829         //
830         double Kn;          //Knudsen number
831         double lambda;      //molecule mean free path, m
832         double alpha;       //thermal diffusivity, m2/s
833         double Re_fo;       //Reynolds number, fluid only, -
834         double Pe_r;        //rotational Peclet number, -
835         double C, n;        //friction factor coefficients
836
837         double k_B = 1.38064852e-23;    //Boltzmann constant, J/K
838         double D_B;         //Brownian diffusion coefficient, m2/s
839         double V_T;         //thermophoretic diffusion coefficient, m2/s
840         //double N_bt;      //ratio of Brownian and thermophoretic diffusivities
841         double grad_T;      //film temperature gradient, K
842
843         double tau_p;       //nanoparticle relaxation time, s
844         double beta;        //thermophoresis coefficient / proportionality factor
845         double beta_oil;    //thermophoresis coefficient / proportionality factor for oil
```

```
846         double beta_ref;    //thermophoresis coefficient / proportionality factor for
            refrigerant
847         double gradT;       //temperature gradient, K/m
848         double Vel_eo;      //turbulent eddies velocity, m/s
849         double Vel_e;       //nanoparticle/fluid slip velocity due to turbulent eddies, m/s
850         double Vel_t;       //thermophoresis velocity, m/s
851         double Vel_Brown;   //Brownian velocity, m/s
852         double Vel_g;       //gravity velocity, m/s
853
854         double t_inert;
855         double t_Brown;
856         double t_thermoph;
857         double t_grav;
858
859         double delta_u_inert;       //inertial slip velocity, m/s
860         double delta_u_Brown;       //Brownian motion slip velocity, m/s
861         double delta_u_thermoph;    //thermophoresis slip velocity, m/s
862         double delta_u_grav;        //gravitational slip velocity, m/s
863
864         double F_rotat;
865         double F_Brown;
866         double F_thermoph;
867         double F_grav;
868         double F_inert;
869         double gamma;       //shear rate
870
871         double tau_w;       //shear stress at the wall, Pa
872         //double ff;            //friction factor
873         double u_ave;       //mean axial velocity or bulk velocity, m/s
874         double u_f;         //liquid layer velocity, m/s
875         double u_i;         //liquid-vapor interface velocity, m/s
876
877         double phi_v_guess; //volume fraction initial guess
878         double phi_v;       //nanoparticle volume fraction in laminar sublayer
879         double delta_v;     //thickness of the laminar sublayer, m
880         double delta_v_plus;//non-dimensional thickness of the laminar sublayer
881         double NMF_v_guess; //mass fraction initial guess
882         double NMF_v;       //nanoparticle mass fraction in refrigerant and oil, in laminar
            sublayer
883         double NMF_v_oil;   //nanoparticle mass fraction in oil, in laminar sublayer
884
885         //properties
886         double rho_nano;    //nanoparticle density, kg/m3
887         double k_nano;      //nanoparticle thermal conductivity, W/m-K
888         double cp_nano;     //nanoparticle specific heat, kJ/kg-K
889
890         double t_v;         //temperature in laminar sublayer, C
891         double mu_v;        //dynamic viscosity in laminar sublayer, Pa*s
892         double rho_v;
893         double cp_v;
894         double k_v;
895         double mu_b;        //dynamic viscosity in turbulent sublayer, Pa*s
896         double rho_b;
897         double cp_b;
898         double k_b;
899         double sigma_b;
900
901         //pure refrigerant saturation properties used to recalculate
            nanolubricant-refrigerant mixture
902         double mu_f_pure;
903         double rho_f_pure;  //density of pure refrigerant, kg/m3
904         double cp_f_pure;
905         double k_f_pure;
906         double rho_f_N;     //density of refrigerant-oil mixture, kg/m3
907         double k_f_N;
908
```

```
909       double NMF_b;       //nanoparticle mass fraction in oil and refrigerant
910
911       //
912       double Pr_v;
913       double Pr_b;
914       double Re_b;
915       double Nu_b;        //Nusselt bulk
916       double t_wall_new;
917       double S_p;
918
919       double r_vt;        //laminar-turbulent interface radius, m
920       double u_f_star;
921       double y_plus_i;
922
923       int count;          //convergence counter
924       int count_NMF;      //NMF convergence counter
925       int iter;           //number of iterations
926
927       //variables needed for radial analysis of the laminar sublayer
928       double ff_i;        //interfacial friction factor, -
929       double tau_v[100];  //laminar sublayer shear stress, Pa
930       double u_v[100];    //laminar sublayer  velocity, m/s
931       double r_v[100];    //laminar sublayer radius, m
932       double t_v_Rad[100];//laminar sublayer temperature, C
933       double Pr_v_Rad[100];
934       double mu_v_Rad[100];   //laminar sublayer viscosity, N-s/m2, kg/m-s, Pa-s
935       double k_v_Rad[100];    //laminar sublayer conductivity, W/m-K
936       double cp_v_Rad[100];   //laminar sublayer specific heat, kJ/kg-K
937       double rho_v_Rad[100];  //laminar sublayer density, kg/m3
938       double phi_v_Rad[100];  //laminar sublayer nanoparticle volume fraction
939       double NMF_v_Rad[100];  //laminar sublayer nanoparticle mass fraction (for
          nanolubricant-refrigerant mixture)
940       double NMF_Rad[100];    //laminar sublayer nanoparticle mass fraction (for
          nanolubricant only)
941       double grad_T_Rad[100]; //laminar sublayer temperature gradient, K/m
942       double Vol_rate_Rad[100];//laminar sublayer volumetric flow rate per unit wetted
          perimeter, m2/s
943       double tau_i;       //interfacial shear stress, Pa
944       double tau_vt;      //laminar-turbulent interface shear stress, Pa
945       double t_i;         //liquid-vapor interface temperature, C
946       double u_vt;        //laminar-turbulent interface velocity, m/s
947       double r_i;         //interfacial radius, m
948       double q_v[100];    //laminar sublayer heat flux, W/m2
949       double HH;          //heat transfer parameter
950       double htc_[100];
951       double htc_rv;
952       double htc_radial;
953       double rho_f_N_v_Rad[100];
954       double NanoOilRefRad;   //liquid refrigerant and nanooil mass, inside each radius
          of the laminar sublayer, kg
955       double OilRad;      //oil mass, inside each radius of the laminar sublayer, kg
956       double OilNanoRad;  //nanooil mass, inside each radius of the laminar sublayer, kg
957       double DpSeg_tot_;  //segment delta pressure, Pa
958
959       int j;
960
961       //output variables
962       double htc;         //heat transfer coefficient, W/m2-C
963
964       //flow
965
966       count = 0;
967       count_NMF = 0;
968       iter = 50;
969
970       if (NMF == 0)
```

```
971         {
972             rho_nano = 0;
973             k_nano = 0;
974             cp_nano = 0;
975         }
976 
977         rho_f_pure = PropsSI("D", "P", p_in * 1000, "Q", 0, fluid);
978         k_f_pure = PropsSI("L", "P", p_in * 1000, "Q", 0, fluid);
979         cp_f_pure = PropsSI("C", "P", p_in * 1000, "Q", 0, fluid) / 1000;
980         mu_f_pure = PropsSI("V", "P", p_in * 1000, "Q", 0, fluid);
981 
982         k_f_N = OilMixtureConductivity(t_in, w_local, k_f_pure);
983 
984         ////////////////////////////segment inventory, based on section
            area////////////////////////////
985         epsilon = 0;     //single phase flow
986         Inventory(fluid, nano_mater, D_nano, SectA, lengthSeg, epsilon, w_local, NMF, rho_g,
             rho_f,
987             VolSeg, RefSegLIQ, RefSegVAP, RefSeg, OilNanoSeg, NanoSeg, OilSeg, V_nano,
            N_nano_Seg);
988 
989         //calculate bulk properties for the case with water and nanoparticles
990         if (OMF == 0 && NMF > 0)
991         {
992             //calculation of the saturated properties before nanoparticles (NMF = 0)
993             CalcProps(fluid, A0, A1, A2, a0, b0, oil, nano_mater, nano_shape, D_nano, 0, 0,
            m_dot_fluid, p_in, h_in,
994                 t_in, x_in, rho_in, v_in, cp_in, mu_in, k_in, sigma_in,
995                 t_sat_f, h_f, rho_f, v_f, cp_f, mu_f, k_f, sigma_f,
996                 t_sat_g, h_g, rho_g, v_g, cp_g, mu_g, k_g, sigma_g, h_fg, dh_x, t_sat,
                w_local);
997 
998             rho_nano = 3880;    // NanoDensity(nano_mater);     //3880;
999             k_nano = NanoConductivty(nano_mater, t_in);     //36;   40
1000            cp_nano = NanoSpecificHeat(nano_mater, t_in) * 1000;
1001 
1002            //calculation of the volume concentration in water
1003            phi = NMF / (NMF + (1 - NMF)*rho_nano / rho_f);
1004 
1005            rho_b = phi*rho_nano + (1 - phi)*rho_f;     //Buongiorno
1006            mu_b = mu_f*(1 + 39.11*phi + 533.9*pow(phi, 2));    //Buongiorno
1007            k_b = k_f*(1 + 7.47*phi);                   //Buongiorno
1008            cp_b = phi*cp_nano + (1 - phi)*cp_f*1000;       //Buongiorno
1009            sigma_b = sigma_f;
1010 
1011        }
1012        ////calculate bulk properties for the case with nanolubricant and refrigerant
1013        else
1014        {
1015            //calculation of the saturated properties before nanoparticles (NMF = 0)
1016            CalcProps(fluid, A0, A1, A2, a0, b0, oil, nano_mater, nano_shape, D_nano, OMF, 0
            , m_dot_fluid, p_in, h_in,
1017                t_in, x_in, rho_in, v_in, cp_in, mu_in, k_in, sigma_in,
1018                t_sat_f, h_f, rho_f, v_f, cp_f, mu_f, k_f, sigma_f,
1019                t_sat_g, h_g, rho_g, v_g, cp_g, mu_g, k_g, sigma_g, h_fg, dh_x, t_sat,
                w_local);
1020 
1021            //calculation of inlet fluid bulk properties
1022            if (NMF > 0)
1023            {
1024                CalcProps(fluid, A0, A1, A2, a0, b0, oil, nano_mater, nano_shape, D_nano,
                OMF, NMF, m_dot_fluid, p_in, h_in,
1025                    t_in, x_in, rho_in, v_in, cp_in, mu_in, k_in, sigma_in,
1026                    t_sat_f, h_f, rho_b, v_f, cp_b, mu_b, k_b, sigma_b,
1027                    t_sat_g, h_g, rho_g, v_g, cp_g, mu_g, k_g, sigma_g, h_fg, dh_x, t_sat,
                    w_local);
```

```
1028
1029                rho_nano = NanoDensity(nano_mater);     //3880;
1030                k_nano = NanoConductivty(nano_mater, t_in);     //36;   40
1031                cp_nano = NanoSpecificHeat(nano_mater, t_in) * 1000;
1032
1033                //in case of oil, NMF from the input file is intended to be the mass
                    fraction of nanoparticles in oil only,
1034                //therefore it is necessary to recalculate NMF as the mass fraction of the
                    oil-refrigerant mixture
1035                NMF_b = NanoSeg / (NanoSeg + OilSeg + RefSegLIQ);
1036                phi = NMF_b / (NMF_b + (1 - NMF_b)*rho_nano / rho_f);
1037
1038            }
1039            else
1040            {
1041                rho_b = rho_f;
1042                cp_b = cp_f;
1043                mu_b = mu_f;
1044                k_b = k_f;
1045            }
1046
1047            cp_b = cp_b * 1000;
1048        }
1049
1050        //
1051        u_ave = m_dot_fluid / (rho_b * SectA);
1052        Pr_b = mu_b*cp_b / k_b;
1053        Re_b = rho_b*u_ave*Dh / mu_b;   //G_flux*Dh / mu_b;
1054        C = 0.184;
1055        n = 0.2;
1056
1057        //
1058        //ff = C / pow(Re_b, n);    //passed as subroutine input
1059        tau_w = ff / 8 * rho_b * pow(u_ave, 2);
1060        //tau_w = C / 8 * pow(mu_b, 2)*pow(Re_b, 2 - n) / (rho_b*pow(Dh, 2));
1061
1062        ////////////////////////////////slip
            mechanisms//////////////////////////////////////
1063        if (NMF > 0)
1064        {
1065            //(o) Knudsen number
1066            lambda = 7.5*1e-9;  //7.5*1e-9; for nanolubricants  //0.3*1e-9; for water, from
                Buongiorno
1067            Kn = lambda / (D_nano*1e-9);
1068
1069            //(o) rotation
1070            alpha = k_b / (rho_b*cp_b);
1071            tau_w = ff / 8 * rho_b * pow(u_ave, 2);
1072            //tau_w = C*pow(mu_b, 2)*pow(Re_b, 2 - n) / (8 * rho_b*pow(Dh, 2));
1073            Pe_r = (tau_w / mu_b)*(pow(D_nano*1e-9, 2) / alpha);
1074            gamma = tau_w / mu_b;
1075            F_rotat = mu_b*rho_nano*gamma*M_PI / 4 * pow(D_nano*1e-9, 2) / rho_b;
1076
1077            //(o) inertia
1078            tau_p = rho_nano*pow(D_nano*1e-9, 2) / (18 * mu_b);         //relaxation time
1079            Vel_eo = sqrt(C / 8)*pow(Re_b, 1 - n / 2)*mu_b / (rho_b*Dh);
1080            S_p = tau_p*Vel_eo;
1081            t_inert = (D_nano*1e-9) / Vel_eo;
1082            delta_u_inert = Vel_eo*exp(-t_inert / tau_p);       //slip velocity
1083            F_inert = 3 * M_PI*D_nano*1e-9*mu_f*delta_u_inert;
1084
1085            //(o) Brownian diffusion [m2/s]
1086            D_B = k_B*(t_sat_f + 273.15) / (3 * M_PI*mu_b*D_nano*1e-9);
1087            t_Brown = pow(D_nano*1e-9, 2) / D_B;
1088            delta_u_Brown = Vel_eo*exp(-t_Brown / tau_p);       //slip velocity
1089            Vel_Brown = 2 * k_B*(t_sat_f + 273.15) / (M_PI*mu_b*pow(D_nano*1e-9, 2));
```

```
1090            F_Brown = -rho_nano*D_B*phi*(M_PI / 4 * pow(D_nano*1e-9, 2)*Vel_Brown);
                //not sure that grad_phi is correct
1091
1092            //(o) thermophoresis
1093            beta = 0.26*k_f / (2 * k_f + k_nano);       //k_f instead of k_b, according to
                McNab and Meisen, Thermophoresis in liquids (1973)
1094            gradT = q_flux / k_b;
1095            Vel_t = beta*mu_b*gradT / (rho_b*(t_sat_f + 273.15));
1096            t_thermoph = (D_nano*1e-9) / Vel_t;
1097            delta_u_thermoph = Vel_eo*exp(-t_thermoph / tau_p);     //slip velocity
1098            F_thermoph = rho_nano*phi*Vel_t*(M_PI / 4 * pow(D_nano*1e-9, 2)*Vel_t);
1099
1100            //(o) gravity
1101            Vel_g = pow(D_nano*1e-9, 2)*(rho_nano - rho_b)*9.81 / (18 * mu_b);
1102            t_grav = (D_nano*1e-9) / Vel_g;
1103            delta_u_grav = Vel_eo*exp(-t_grav / tau_p);     //slip velocity
1104            F_grav = M_PI / 6 * pow(D_nano*1e-9, 3)*(rho_nano - rho_f)*9.81;
1105
1106        }
1107        ////////////////////////////////////////////////////////////////////
1108
1109
1110        ////////////////////////////////////////////begin
                Buongiorno//////////////////////////////////////////////////
1111
1112        k_B = 1.38064852e-23;
1113        D_B = k_B*(t_sat_f + 273.15) / (3 * M_PI*mu_b*D_nano*1e-9);
1114
1115        if (fluid == "water")
1116        {
1117
1118            //McNab&Meisen, suggested by Buongiorno
1119            beta = 0.26*(k_f_pure / (2 * k_f_pure + k_nano));
1120
1121            ////alternative correlation for beta - (IJHMT - 2015 - Michaelides - Brownian
                movement and thermophoresis of nanoparticles in liquids)
1122            //beta = 1227 * pow(D_nano / 2 / 1, -1.434);        //water
1123        }
1124        else if (fluid != "water" && OMF == 0)
1125        {
1126            //alternative correlation for beta - (IJHMT - 2015 - Michaelides - Brownian
                movement and thermophoresis of nanoparticles in liquids)
1127            beta = 6270 * pow(D_nano / 2 / 1, -1.819);      //R134a
1128        }
1129        else if (fluid != "water" && OMF > 0)
1130        {
1131            //alternative correlation for beta - (IJHMT - 2015 - Michaelides - Brownian
                movement and thermophoresis of nanoparticles in liquids)
1132            beta_oil = 7.1026*pow(D_nano / 2 / 1, -1.579);      //engine oil
1133            beta_ref = 6270 * pow(D_nano / 2 / 1, -1.819);      //R134a
1134
1135            //weighted beta for oil-refrigerant mixture //Bigi
1136            beta = beta_oil * w_local + beta_ref * (1 - w_local);
1137        }
1138
1139        //added for sensitivity analysis on Nbt
1140        beta = beta / pow(10, 3*index_Nbt);
1141
1142        //(i)
1143        u_ave = m_dot_fluid / (rho_b * SectA);
1144        Re_b = rho_b * u_ave * Dh / mu_b;
1145
1146        //friction factor passed as subroutine input
1147        //ff = 0.184 / pow(Re_b, 0.2);      //McAdams friction factor correlation
1148        //ff = 0.046 / pow(Re_b, 0.2);      //Taitel&Dukler friction factor correlation
1149
```

```
1150        tau_w = ff / 8 * rho_b * pow(u_ave, 2);
1151        //tau_w = C / 8 * pow(mu_b, 2)*pow(Re_b, 2 - n) / (rho_b*pow(Dh, 2));
1152
1153        //(ii)
1154        //phi_v_guess = phi*0.001;  //estimate value of nanoparticle volume fraction in
            laminar sublayer
1155        NMF_v_guess = NMF;  //or estimate value of nanoparticle mass fraction in laminar
            sublayer
1156
1157        //(iii) loop to find the correct phi_v
1158        count = 0;
1159        while (TRUE && count_NMF < iter)
1160        {
1161            //for water, from Buongiorno
1162            if (OMF == 0 && NMF > 0)
1163            {
1164
1165                rho_f = PropsSI("D", "P", p_in * 1000, "Q", 0, fluid);
1166                cp_f = PropsSI("C", "P", p_in * 1000, "Q", 0, fluid);
1167                mu_f = PropsSI("V", "P", p_in * 1000, "Q", 0, fluid);
1168                k_f = PropsSI("L", "P", p_in * 1000, "Q", 0, fluid);
1169
1170                //calculation of the volume concentration in water
1171                phi_v_guess = NMF_v_guess / (NMF_v_guess + (1 - NMF_v_guess)*rho_nano /
                    rho_f);
1172
1173                rho_v = phi_v_guess*rho_nano + (1 - phi_v_guess)*rho_f;          //Buongiorno
1174                mu_v = mu_f*(1 + 39.11*phi_v_guess + 533.9*pow(phi_v_guess, 2));
                    //Buongiorno
1175                k_v = k_f*(1 + 7.47*phi_v_guess);                                //Buongiorno
1176                cp_v = phi_v_guess*cp_nano + (1 - phi_v_guess)*cp_f;
                    //Buongiorno
1177
1178            }
1179            ////for nanolubricants
1180            else
1181            {
1182                CalcProps(fluid, A0, A1, A2, a0, b0, oil, nano_mater, nano_shape, D_nano,
                    OMF, 0, m_dot_fluid, p_in, h_in,
1183                    t_in, x_in, rho_in, v_in, cp_in, mu_in, k_in, sigma_in,
1184                    t_sat_f, h_f, rho_f, v_f, cp_f, mu_f, k_f, sigma_f,
1185                    t_sat_g, h_g, rho_g, v_g, cp_g, mu_g, k_g, sigma_g, h_fg, dh_x, t_sat,
                        w_local);
1186
1187                if (NMF_v_guess > 0)
1188                {
1189                    CalcProps(fluid, A0, A1, A2, a0, b0, oil, nano_mater, nano_shape, D_nano
                        , OMF, NMF_v_guess, m_dot_fluid, p_in, h_in,
1190                        t_in, x_in, rho_in, v_in, cp_in, mu_in, k_in, sigma_in,
1191                        t_sat_f, h_f, rho_v, v_f, cp_v, mu_v, k_v, sigma_f,
1192                        t_sat_g, h_g, rho_g, v_g, cp_g, mu_g, k_g, sigma_g, h_fg, dh_x,
                            t_sat, w_local);
1193
1194                    phi_v_guess = NMF_v_guess / (NMF_v_guess + (1 - NMF_v_guess)*rho_nano /
                        rho_f);
1195
1196                }
1197                else
1198                {
1199                    rho_v = rho_f;
1200                    cp_v = cp_f;
1201                    mu_v = mu_f;
1202                    k_v = k_f;
1203                }
1204
1205                cp_v = cp_v * 1000;
```

```
1206
1207            }
1208
1209            ////given delta_f as input, then delta_v_plus can be simply calculated as:
1210            //if (x_in > 0 && x_in < 1)
1211            //{
1212            //  delta_v_plus = delta_f / (mu_v / rho_v / sqrt(tau_w / rho_b));
1213            //}
1214            //else
1215            //{
1216            //  delta_v_plus = 15.5;  // 8.7  15.5; //Buongiorno used 15.5 for his validation
1217            //}
1218
1219            if (fluid == "water")
1220            {
1221                delta_v_plus = 15.5;  // 8.7  15.5; //Buongiorno used 15.5 for his validation
1222            }
1223            else
1224            {
1225                delta_v_plus = 10.5;        //10.5 makes  Buongiorno converge to
                    Dittus-Boelter in the case of pure refrigerant
1226                                            //it is used here also for the case of
                                                refrigerant-oil mixtures
1227            }
1228
1229            //delta_v_plus = delta_v_plus / 2;
1230
1231            delta_v = delta_v_plus*(mu_v / rho_v / sqrt(tau_w / rho_b));    //mu_b or mu_v
            ??? according to Hewitt, Annular Two-Phase Flow, pag.126,
1232                                                                           //I think this
                                                                               form is
                                                                               correct, rather
                                                                               than:
                                                                               delta_v_plus*(mu_
                                                                               b / rho_b /
                                                                               sqrt(tau_w /
                                                                               rho_b))
1233            //(iv)
1234            D_B = k_B*(t_sat_f + 273.15) / (3 * M_PI*mu_v*D_nano*1e-9);     //function of
            laminar properties (from email correspondence with Dr Buongiorno)
1235            grad_T = q_flux*delta_v / k_v;                                  //function of
            laminar properties (from email correspondence with Dr Buongiorno)
1236            V_T = beta*mu_v*grad_T / (rho_v*(t_sat_f + 273.15));            //function of
            laminar properties (from email correspondence with Dr Buongiorno)
1237            N_bt = D_B / V_T;
1238
1239            //(v)
1240            phi_v = phi*N_bt*(1 - exp(-1 / N_bt));
1241
1242            NMF_v = (phi_v * rho_nano / rho_f) / (1 - phi_v + phi_v * rho_nano / rho_f);
1243
1244
1245            ////(vi)
1246            //if (abs(phi_v - phi_v_guess) < 0.0001)
1247            //{
1248            //  break;
1249            //}
1250            //else
1251            //{
1252            //  phi_v_guess = phi_v;
1253            //}
1254
1255            //(vi)
1256            if (abs(NMF_v - NMF_v_guess) < 1e-8)
1257            {
1258                break;
```

```
1259            }
1260            else
1261            {
1262                NMF_v_guess = (NMF_v + NMF_v_guess) / 2;    //NMF_v;
1263                count_NMF++;
1264            }
1265
1266        }
1267
1268        //(vii)
1269        r_vt = De / 2 - delta_v;
1270
1271        t_wall_new = t_wall;
1272
1273        while (TRUE && count < iter)
1274        {
1275            t_v = (t_wall_new + t_sat_f) / 2;
1276
1277            //for water, from Buongiorno
1278            if (OMF == 0 && NMF > 0)
1279            {
1280                rho_f = PropsSI("D", "T", t_v + 273.15, "Q", 0, fluid);
1281                mu_f = PropsSI("V", "T", t_v + 273.15, "Q", 0, fluid);
1282                k_f = PropsSI("L", "T", t_v + 273.15, "Q", 0, fluid);
1283                cp_f = PropsSI("C", "T", t_v + 273.15, "Q", 0, fluid);
1284                cp_nano = NanoSpecificHeat(nano_mater, t_v) * 1000;
1285
1286                rho_v = phi_v*rho_nano + (1 - phi_v)*rho_f;          //Buongiorno
1287                mu_v = mu_f*(1 + 39.11*phi_v + 533.9*pow(phi_v, 2));    //Buongiorno
1288                k_v = k_f*(1 + 7.47*phi_v);                          //Buongiorno
1289                cp_v = phi_v*cp_nano + (1 - phi_v)*cp_f;             //Buongiorno
1290            }
1291            ////for nanolubricants
1292            else
1293            {
1294                p_in = PropsSI("P", "T", t_v + 273.15, "Q", 0, fluid) / 1000;       //!!!
                    check that tsat changes when props are calculated
1295
1296                CalcProps(fluid, A0, A1, A2, a0, b0, oil, nano_mater, nano_shape, D_nano,
                    OMF, 0, m_dot_fluid, p_in, h_in,
1297                    t_in, x_in, rho_in, v_in, cp_in, mu_in, k_in, sigma_in,
1298                    t_v, h_f, rho_f, v_f, cp_f, mu_f, k_f, sigma_f,
1299                    t_sat_g, h_g, rho_g, v_g, cp_g, mu_g, k_g, sigma_g, h_fg, dh_x, t_sat,
                        w_local);
1300
1301                if (NMF_v > 0)
1302                {
1303                    CalcProps(fluid, A0, A1, A2, a0, b0, oil, nano_mater, nano_shape, D_nano
                        , OMF, NMF_v, m_dot_fluid, p_in, h_in,
1304                        t_in, x_in, rho_in, v_in, cp_in, mu_in, k_in, sigma_in,
1305                        t_v, h_f, rho_v, v_f, cp_v, mu_v, k_v, sigma_f,
1306                        t_sat_g, h_g, rho_g, v_g, cp_g, mu_g, k_g, sigma_g, h_fg, dh_x,
                            t_sat, w_local);
1307                }
1308                else
1309                {
1310                    rho_v = rho_f;
1311                    cp_v = cp_f;
1312                    mu_v = mu_f;
1313                    k_v = k_f;
1314                }
1315
1316                cp_v = cp_v * 1000;
1317            }
1318
1319            Pr_v = mu_v*cp_v / k_v;
```

```
1320
1321        ////////////////////////////////////ADDED FOR RADIAL ANALYSIS OF THE LAMINAR
            SUBLAYER////////////////////////////////////
1322
            ////////////////////////////////////////////////////////START////////////////////
            ////////////////////////////////////
1323
1324        DpSeg_tot_ = -2*lengthSeg/Dh*tau_w;
1325
1326        delta_f = delta_v;
1327        r_i = De / 2 - delta_v;
1328
1329        //tau_w = tau_i*r_i / (De / 2) + 0.5*DpSeg_tot_*(pow(r_i, 2) - pow((De / 2),
            2)) / (De / 2);
1330        tau_i = tau_w*(De / 2)/r_i - 0.5*DpSeg_tot_*(pow(r_i, 2) - pow((De / 2), 2)) /
            r_i;
1331
1332        u_i = 1 / mu_b * ((tau_i*r_i + 0.5*DpSeg_tot_*pow(r_i, 2))*log(De / 2 / r_i) -
1333            0.25*DpSeg_tot_*(pow(De / 2, 2) - pow(r_i, 2)));        //ATPF eq.4.42, or
                see notes;
1334
1335        t_i = t_in;
1336
1337        //laminar sublayer segmentation
1338        for (j = 0; j <= nRad; j++)
1339        {
1340
1341            t_v_Rad[j] = t_i + ((t_wall_new - t_in) / delta_v) * delta_v / nRad*j;
1342
1343            if (OMF == 0 && NMF > 0)
1344            {
1345                //calculate local NMF in oil AND liquid refrigerant
1346                phi_v_Rad[j] = phi*exp(-(1 - ((delta_v - delta_v / nRad*j) / delta_v)) /
                     N_bt);
1347
1348                rho_f = PropsSI("D", "T", t_v_Rad[j] + 273.15, "Q", 0, fluid);
1349                mu_f = PropsSI("V", "T", t_v_Rad[j] + 273.15, "Q", 0, fluid);
1350                k_f = PropsSI("L", "T", t_v_Rad[j] + 273.15, "Q", 0, fluid);
1351                cp_f = PropsSI("C", "T", t_v_Rad[j] + 273.15, "Q", 0, fluid);
1352                cp_nano = NanoSpecificHeat(nano_mater, t_v_Rad[j]) * 1000;
1353
1354                rho_v_Rad[j] = phi_v_Rad[j] * rho_nano + (1 - phi_v_Rad[j])*rho_f;
                            //Buongiorno
1355                mu_v_Rad[j] = mu_f*(1 + 39.11*phi_v_Rad[j] + 533.9*pow(phi_v_Rad[j], 2
                    ));   //Buongiorno
1356                k_v_Rad[j] = k_f*(1 + 7.47*phi_v_Rad[j]);
                    //Buongiorno
1357                cp_v_Rad[j] = phi_v_Rad[j] * cp_nano + (1 - phi_v_Rad[j])*cp_f;
                                //Buongiorno
1358            }
1359            ////for nanolubricants
1360            else
1361            {
1362                rho_f_N_v_Rad[j] = OilMixtureDensity(t_v_Rad[j], w_local, rho_f_pure);
1363
1364                //calculate local NMF in oil AND liquid refrigerant
1365                phi_v_Rad[j] = phi*exp(-(1 - ((delta_v - delta_v / nRad*j) / delta_v)) /
                     N_bt);
1366                NMF_v_Rad[j] = (phi_v_Rad[j] * rho_nano / rho_f_N_v_Rad[j]) / (1 -
                    phi_v_Rad[j] + phi_v_Rad[j] * rho_nano / rho_f_N_v_Rad[j]);
1367
1368                //laminar sublayer inventory
1369                NanoOilRefRad = ((OilNanoSeg + RefSegLIQ)*delta_v / delta_f) / nRad;
1370                OilNanoRad = (OilNanoSeg *delta_v / delta_f) / nRad;
1371                NanoRad = NMF_v_Rad[j] * NanoOilRefRad;
1372
```

```
1373                //NMF in oil only - needed in the thermophysical properties routine
1374                //NMF_Rad[j] = NMF_v_Rad[j] * NanoOilRefRad / (NanoRad + OilNanoRad);
1375                NMF_Rad[j] = NMF_v_Rad[j] * NanoOilRefRad / (OilNanoRad);
1376
1377                if (NMF_Rad[j] >= 1)
1378                {
1379                    system("pause");
1380                }
1381
1382                ////calculation of properties with nanoparticles, at the value of
                    phi_v_Rad[j] AND at the pressure corresponding to t_v
1383                rho_v_Rad[j] = NanoOilMixtureDensity(nano_mater, NMF_Rad[j], t_v_Rad[j],
                     w_local, rho_f_pure);
1384                cp_v_Rad[j] = NanoOilMixtureSpecificHeat(nano_mater, NMF_Rad[j], t_v_Rad
                    [j], w_local, cp_f_pure) * 1000;
1385                mu_v_Rad[j] = NanoOilMixtureDynViscosity(nano_mater, D_nano, NMF_Rad[j],
                     t_v_Rad[j], w_local, mu_f_pure);
1386                if (mu_v_Rad[j] < 0)
1387                {
1388                    system("pause");
1389                }
1390                k_v_Rad[j] = NanoOilMixtureConductivity(nano_mater, NMF_Rad[j], t_v_Rad[
                    j], w_local, k_f_pure);
1391            }
1392
1393            Pr_v_Rad[j] = mu_v_Rad[j] * cp_v_Rad[j] / k_v_Rad[j];
1394
1395        }
1396
1397        for (j = 0; j <= nRad; j++)
1398        {
1399            //calculations
1400            r_v[j] = r_vt + delta_v / nRad*j;
1401
1402            if (j == 0)
1403            {
1404                //u_v[j] = u_i - 1 / mu_v_Rad[j] * ((tau_i*r_i +
                    0.5*DpSeg_tot_*pow(r_i, 2))*log(r_v[j] / r_i) -
1405                //  0.25*DpSeg_tot_*(pow(r_v[j], 2) - pow(r_i, 2)));        //ATPF
                    eq.4.42, or see notes
1406                tau_v[j] = tau_i*r_i / r_v[j] + 0.5*DpSeg_tot_*(pow(r_i, 2) - pow(r_v[j
                    ], 2)) / r_v[j];
1407
1408            }
1409            else
1410            {
1411                //u_v[j] = u_v[j - 1] - 1 / mu_v_Rad[j] * ((tau_i*r_i +
                    0.5*DpSeg_tot_*pow(r_i, 2))*log(r_v[j] / r_v[j-1]) -
1412                //  0.25*DpSeg_tot_*(pow(r_v[j], 2) - pow(r_v[j-1], 2)));       //ATPF
                    eq.4.42, or see notes
1413                tau_v[j] = tau_v[j - 1] * r_v[j - 1] / r_v[j] + 0.5*DpSeg_tot_*(pow(r_v[
                    j - 1], 2) - pow(r_v[j], 2)) / r_v[j];
1414                //tau_v[j] = tau_i * r_i / r_v[j] + 0.5*DpSeg_tot_*(pow(r_i, 2) -
                    pow(r_v[j], 2)) / r_v[j];
1415            }
1416
1417            q_v[j] = q_flux * (De / 2) / r_v[j];
1418
1419        }
1420
1421        for (j = nRad; j >= 0; j--)
1422        {
1423            if (j == nRad)
1424            {
1425                u_v[j] = 0;
1426            }
```

```
1427            else
1428            {
1429                //u_v[j] = u_v[j + 1] + 1 / mu_v_Rad[j + 1] * ((tau_v[j] * r_v[j] +
                    0.5*DpSeg_tot_*pow(r_v[j], 2))*log(r_v[j + 1] / r_v[j]) -
1430                //  0.25*DpSeg_tot_*(pow(r_v[j + 1], 2) - pow(r_v[j], 2)));     //ATPF
                    eq.4.42, or see notes
1431                //u_v[j] = 1 / mu_v_Rad[j] * ((tau_v[j] * r_v[j] +
                    0.5*DpSeg_tot_*pow(r_v[j], 2))*log(De / 2 / r_v[j]) -
1432                //  0.25*DpSeg_tot_*(pow(De / 2, 2) - pow(r_v[j], 2)));     //ATPF
                    eq.4.42, or see notes
1433                u_v[j] = 1 / mu_v_Rad[j] * ((tau_i * r_i + 0.5*DpSeg_tot_*pow(r_i, 2))*
                    log(De / 2 / r_v[j]) -
1434                    0.25*DpSeg_tot_*(pow(De / 2, 2) - pow(r_v[j], 2)));     //ATPF
                        eq.4.42, or see notes
1435            }
1436
1437        }
1438
1439        //u_vt = u_v[0];
1440        //u_i = u_vt + 1 / mu_b * ((tau_i*r_i + 0.5*DpSeg_tot_*pow(r_i, 2))*log(r_vt /
            r_i) -
1441        //  0.25*DpSeg_tot_*(pow(r_vt, 2) - pow(r_i, 2)));
1442
1443
1444        ////////////////////////////////////velocity profile in turbulent
            flow////////////////////////////////////
1445
1446        //u_f_star = sqrt(tau_w / rho_b);
1447        //y_plus_i = u_f_star*rho_b / mu_b * (De / 2 - r_i);
1448        ////u_i = u_f_star*(5.5 + 2.5*log(y_plus_i));
1449        //u_i = u_f_star*y_plus_i;
1450
1451        //for (j = nRad; j >= 0; j--)
1452        //{
1453        //  y_plus[j] = u_f_star*rho_v_Rad[j] / mu_v_Rad[j] * (De/2 - r_v[j]);
1454
1455        //  //if (y_plus[j] >= 0 && y_plus[j] < 5)
1456        //  //{
1457        //      u_v[j] = u_f_star*y_plus[j];
1458        //  //}
1459        //  //else if (y_plus[j] >= 5 && y_plus[j] < 30)
1460        //  //{
1461        //  //  u_v[j] = u_f_star*(-3.05 + 5*log(y_plus[j]));
1462        //  //}
1463        //  //else if (y_plus[j] >= 30)
1464        //  //{
1465        //  //  u_v[j] = u_f_star*(5.5 + 2.5*log(y_plus[j]));
1466        //  //}
1467
1468        //}
1469
1470        ////////////////////////////////////////////////////////////////////////
1471
1472        //(viii-ix)
1473        ////////////////////////////heat transfer coefficient calculation
1474        //as from Buongiorno's analysis
1475        HH = 0;
1476        //film laminar region
1477        for (j = nRad; j >= 0; j--)
1478        {
1479            if (j == 0)
1480            {
1481                htc_[j] = De / 2 / r_v[j] / tau_v[j] * mu_v_Rad[j] / k_v_Rad[j] * (u_i -
                     u_v[j]);
1482            }
1483            else
```

```
1484            {
1485                htc_[j] = De / 2 / r_v[j] / tau_v[j] * mu_v_Rad[j] / k_v_Rad[j] * (u_v[j
                     - 1] - u_v[j]);
1486            }
1487
1488            HH = HH + htc_[j];
1489        }
1490
1491        //film turbulent region
1492        tau_vt = tau_i;
1493        htc_rv = De / 2 / r_vt / tau_vt / cp_b * (u_ave - u_i);
1494        //htc_rv = De / 2 / r_vt / tau_vt / cp_b * (u_i - u_vt);
1495        //htc_rv = De / 2 / r_vt / tau_vt * mu_b / k_b * (u_i - u_vt);
1496        HH = HH + htc_rv;
1497
1498        ////vapor core
1499        //htc_core = De / 2 / r_i / tau_i / cp_g * (u_g - u_i);
1500        //HH = HH + htc_core;
1501
1502        htc_radial = 1 / HH;
1503
1504
        ////////////////////////////////////////////////////FINISH//////////////////////
        ////////////////////////////////////
1505
1506
1507        ////(viii)
1508        //Nu_b = 0.021*pow(Re_b, 0.8)*pow(Pr_b, 0.5);       //Pak & Cho (1998)
1509
1510        ////////////////////////////////////velocity profile in turbulent
        flow////////////////////////////////////
1511        //u_f_star = sqrt(tau_w / rho_b);
1512        //y_plus_i = u_f_star*rho_b / mu_b * (De / 2 - r_vt);
1513        //u_i = u_f_star*y_plus_i;
1514        //u_i = u_f_star*(-3.05 + 5 * log(y_plus_i));
1515
1516        ////u_i = (u_v[2] + u_v[3]) / 2;
1517        ////u_i = u_ave * 2.0 / 3.0;
1518        //Nu_b = (ff / 8 * Re_b * Pr_b) / (1 + u_i / u_ave * (Pr_v - 1));
        //Buongiorno, eq. 47
1519
1520        //htc = tau_w / (mu_v * u_i / k_v + (u_ave - u_i) / cp_b);      //Buongiorno,
        eq. 46
1521        //Nu_b = htc * Dh / k_b;
1522
1523        //Nu_b = (ff / 8 * Re_b * Pr_b) / (1 + 8.7 * sqrt(ff / 8) * (Pr_v - 1));
        //Buongiorno, eq. 48
1524
1525        if (Re_b < 2300)
1526        {
1527            Nu_b = 4.36;    //Nusselt number for laminar flow at constant heat flux
1528        }
1529        else
1530        {
1531            Nu_b = (ff / 8 * (Re_b - 1000) * Pr_b) / (1 + delta_v_plus*sqrt(ff / 8)*(pow
                (Pr_v, 2.0 / 3.0) - 1));        //Buongiorno, eq. 50
1532        }
1533
1534        //(ix)
1535        htc = Nu_b*k_b / Dh;
1536        t_wall = t_sat_f + q_flux / htc;    //Newton's law of cooling
1537
1538        if (abs(t_wall_new - t_wall) < 0.01)
1539        {
1540            break;
1541        }
```

```
1542            else
1543            {
1544                t_wall_new = (t_wall_new + t_wall) / 2; //t_wall;
1545                count++;
1546            }
1547
1548        }
1549
1550
1551        if (count == iter)
1552        {
1553            Buongiorno_output_file << "This test did not converge." << endl;
1554        }
1555        else
1556        {
1557            Buongiorno_output_file << G_flux << "," << q_flux << "," << phi << "," << N_bt <<
1558                "," << phi_v << "," << ff << "," << tau_w << "," << D_B <<
1559                "," << grad_T << "," << V_T << "," << delta_v << "," << u_ave << "," <<
                    t_wall <<
1560                "," << Re_b << "," << Pr_b << "," << Pr_v << "," << Nu_b << "," << htc <<
1561                "," << rho_b << "," << cp_b << "," << mu_b << "," << k_b <<
1562                "," << rho_v << "," << cp_v << "," << mu_v << "," << k_v << "," << count_NMF
                     << endl;
1563        }
1564
1565        Buongiorno_radial_output_file << "VolSeg, OilSeg, NanoSeg, RefSegLIQ, RefSegVAP,
            RefSeg, delta_f, delta_v, delta_v_plus,"
1566            "phi_b, phi_v, D_B, V_T, N_bt, mu_b, k_b, rho_b, cp_b, tau_i,"
1567            "r_i, r_vt, u_i, htc_radial" << endl;
1568
1569        Buongiorno_radial_output_file << VolSeg << "," << OilSeg << "," << NanoSeg << "," <<
             RefSegLIQ << "," << RefSegVAP << "," << RefSeg << ","
1570            << delta_f << "," << delta_v << "," << delta_v_plus << "," << phi << "," <<
                phi_v << "," << D_B << "," << V_T << "," << N_bt << ","
1571            << mu_b << "," << k_b << "," << rho_b << "," << cp_b << "," << tau_i << ","
1572            << r_i << "," << r_vt << "," << u_i << "," << htc_radial << endl << endl;
1573
1574        Buongiorno_radial_output_file << "nRad, t_v_Rad[j], mu_v_Rad[j], k_v_Rad[j],
            rho_v_Rad[j], cp_v_Rad[j],"
1575            "NMF_v_Rad[j], phi_v_Rad[j], r_v[j], tau_v[j], q_v[j], u_v[j], Pr_v_Rad[j],
                htc[j]" << endl;
1576
1577        for (j = 0; j <= nRad; j++)
1578        {
1579            Buongiorno_radial_output_file << j << "," << t_v_Rad[j] << "," << mu_v_Rad[j] <<
                 "," << k_v_Rad[j] << "," << rho_v_Rad[j] << "," << cp_v_Rad[j] << ","
1580                << NMF_v_Rad[j] << "," << phi_v_Rad[j] << "," << r_v[j] << "," << tau_v[j]
                    << "," << q_v[j] << "," << u_v[j] << "," << Pr_v_Rad[j] << ","
1581                << htc_[j] << endl;
1582        }
1583
1584        Buongiorno_radial_output_file << endl;
1585
1586        return htc;
1587    }
1588
```

```
1
2
3   #define _USE_MATH_DEFINES
4   #include <cmath>
5   #include "CoolProp.h"
6   #include "HumidAirProp.h"
7   #include <iostream>
8   #include <stdlib.h>
9   #include <fstream>
10  #include <string>
11  #include <cstddef>
12
13
14  using namespace CoolProp;
15  using namespace std;
16
17  double RouhaniAxelsson(string, double, double, double, double, double);
18
19
20  double FrictionFactor(string tube_type, string tube_mater, double tube_roughness, double
     nFins, double heightFin, double beta_deg,
21                        double alpha_deg, double G_flux, double m_dot_fluid, double SectA,
                          double De, double Dr, double rho_f, double mu_f)
22  {
23      //use: calculate friction factor
24      //
25      //source: Jensen, M. K., & Vlakancic, A. (1999). Technical Note Experimental
        investigation of turbulent heat transfer
26      //          and fluid flow in internally finned tubes. International Journal of
        Heat and Mass Transfer, 42(7), 1343-1351.
27      //
28      //      Wang, H. S., & Rose, J. W. (2004). Prediction of effective friction factors
        for single-phase flow in horizontal
29      //      microfin tubes. International journal of refrigeration, 27(8), 904-913.
30      //
31      //      Meyer, J. P., & Olivier, J. A. (2011). Transitional flow inside enhanced
        tubes for fully developed and developing flow
32      //      with different types of inlet disturbances: Part I-Adiabatic pressure drops.
33      //      International Journal of Heat and Mass Transfer, 54(7-8), 1587-1597.
34      //
35      //      Vicente, P. G., Garcıá, A., & Viedma, A. (2002). Experimental study of
        mixed convection and pressure drop in helically
36      //      dimpled tubes for laminar and transition flow. International Journal of
        Heat and Mass Transfer, 45(26), 5091-5105.
37      //
38      //
39      //
40      //author: Andrea Bigi
41      //date: 04/2018
42      //-------------------------------------------------------------------
43
44      //local variables
45      double u_ave;   //mean axial velocity or bulk velocity, m/s
46      double Re;      //Reynolds number, -
47      double C, n;    //friction factor coefficients
48      double ff_s;    //smooth tube friction factor
49      double s;       //average fin thickness, m
50      double Lcsw_Dr; //ratio of characteristic length modified for swirling flow and
        envelope (root) diamter, -
51      double SectA_root;  //Cross-sectional area based on root diameter, m2
52
53
54      //output variables
55      double ff;      //friction factor, -
56
57      //flow
```

```
58
59      u_ave = m_dot_fluid / (rho_f * SectA);
60      Re = rho_f * u_ave * De / mu_f;     //G_flux*Dh / mu_b;
61
62
63      //Blasius correlation
64      C = 0.316;
65      n = 0.25;
66      //McAdams correlation
67      C = 0.184;
68      n = 0.2;
69
70      if (tube_type == "smooth")
71      {
72          if (Re <= 2300)
73          {
74              //Poiseuille correlation
75              ff = 64 / Re;
76          }
77          else
78          {
79              ff = C / pow(Re, n);
80          }
81
82      }
83      else if (tube_type == "microfin")
84      {
85
86          if (Re <= 2300)
87          {
88              //Vicente correlation, corrected by Meyer et al. (2010)
89              ff = 64 / Re * (1 + 88 * pow(heightFin/Dr, 2.2) * pow(Re, 0.2));
90          }
91          else
92          {
93              //For ff_s, Meyer et al. (2010) suggests using directly Darcy-Weisbach
                friction factor. Not used here because it needs DP and tube length.
94              //Filonenko (1954), suggested by Wang et al. (2004) for smooth tube
                friction factor
95              ff_s = pow(1.58*log(Re) - 3.28, -2);
96
97              //McAdams correlation, chosen to be consistent with Buongiorno (2006)
98              ff_s = C / pow(Re, n);
99
100             s = 4/3 * heightFin * tan(alpha_deg * M_PI / 180 / 2);
101             //Meyer et al. (2010)
102             Lcsw_Dr = (1 - 0.994 * pow(nFins * sin(beta_deg * M_PI / 180) / M_PI, 0.89)
                * pow(2* heightFin / Dr, 0.44) * pow((M_PI/nFins - s/Dr) * cos(beta_deg *
                M_PI / 180), 0.41));
103             //Wang et al. (2004)
104             Lcsw_Dr = (1 - 1.577 * pow(nFins * sin(beta_deg * M_PI / 180) / M_PI, 0.64)
                * pow(2 * heightFin / Dr, 0.53) * pow((M_PI / nFins - s / Dr) * cos(beta_deg
                 * M_PI / 180), 0.28));
105
106             SectA_root = M_PI / 4 * pow(Dr, 2);
107
108             ff = ff_s * (pow(Lcsw_Dr, -1.25) * pow((SectA_root /SectA), 1.75) - 0.0151 /
                 ff_s * (pow(Lcsw_Dr, -1.25) * pow((SectA_root / SectA), 1.75)) * exp(-Re /
                6780));
109         }
110
111     }
112
113     return ff;
114 }
115
```

```
116 ////////////////////////////////SINGLE PHASE//////////////////////////////////////
117
118 double singlePhase_L(double G_flux, double m_dot_fluid, double SectA, double Dh, double
    length, double rho_f, double mu_f)
119 {
120     //use: calculate liquid single phase pressure drop
121     //
122     //source:
123     //
124     //author: Andrea Bigi
125     //date: 10/2017
126     //---------------------------------------------------------------------
127
128     //local variables
129     double C, n;        //friction factor coefficients
130     double u_ave;   //mean axial velocity or bulk velocity, m/s
131     double Re;      //Reynolds number, -
132     double ff;      //friction factor, -
133     double tau_w;   //shear stress at the wall, kPa
134
135
136     //output variables
137     double Dp_Lsp;      //pressure drop, kPa
138
139     //flow
140
141     //McAdams friction factor correlation
142     C = 0.184;
143     n = 0.2;
144
145     u_ave = m_dot_fluid / (rho_f * SectA);
146     Re = rho_f*u_ave*Dh / mu_f; //G_flux*Dh / mu_b;
147     ff = C / pow(Re, n);
148     tau_w = (ff / 8 * rho_f * pow(u_ave, 2)) / 1000;
149     tau_w = (C / 8 * pow(mu_f, 2)*pow(Re, 2 - n) / (rho_f*pow(Dh, 2))) / 1000;
150
151     Dp_Lsp = 4* tau_w * length / Dh;
152
153     return Dp_Lsp;
154 }
155
156 ////////////////////////////////TWO-PHASE//////////////////////////////////////
157
158 double twoPhaseMomentumDp(string fluid, double G_flux, double sigma_in, double sigma_out
    , double p_in, double p_out, double rho_f_in, double rho_f_out, double x_in, double
    x_out)
159 {
160     //use: calculate two-phase momentum pressure drop
161     //
162     //source:
163     //
164     //author: Andrea Bigi
165     //date: 03/2016
166     //---------------------------------------------------------------------
167
168     //local variables
169     double rho_g_in;      //inlet vapor density, kg/m3
170     double rho_g_out;     //outlet vapor density, kg/m3
171     double epsilon_in;    //inlet void fraction
172     double epsilon_out;   //outlet void fraction
173
174     //output variables
175     double Dp_mom;      //pressure drop, kPa
176
177     //flow
178
```

```
179         rho_g_in = PropsSI("D", "P", p_in * 1000, "Q", 1, fluid);
180         rho_g_out = PropsSI("D", "P", p_out * 1000, "Q", 1, fluid);
181         epsilon_in = RouhaniAxelsson(fluid, G_flux, sigma_in, rho_g_in, rho_f_in, x_in);
182         epsilon_out = RouhaniAxelsson(fluid, G_flux, sigma_out, rho_g_out, rho_f_out, x_out);
183
184         Dp_mom = pow(G_flux, 2) * ((pow(x_out, 2) / (epsilon_out*rho_g_out) + pow(1 - x_out,
             2) / ((1 - epsilon_out)*rho_f_out)) -
185                                    (pow(x_in, 2) / (epsilon_in*rho_g_in) + pow(1 - x_in, 2) /
                                        ((1 - epsilon_in)*rho_f_in))) / 1000;
186
187         return Dp_mom;
188     }
189
190
191     double twoPhaseDp_Choi1999(string fluid, double OMF, double Dh, double length, double
    G_flux,
192                                double m_dot_fluid, double p_in, double v_in, double v_out,
193                                double x_in, double x_out, double mu_f, double h_fg)
194     {
195         //use: calculate two-phase frictional pressure drop in a finned tube
196         //
197         //source: NIST - 1999 - Choi, Kedzierski, Domanski -
198         //        A Generalized Pressure Drop Correlation for Evaporation and Condensation
199         //        of Alternative Refrigerants in Smooth and Micro-fin Tubes
200         //
201         //author: Andrea Bigi
202         //date: 10/2015
203         //------------------------------------------------------------------
204
205         //local variables
206         double Re_fo;   //Reynolds number, all liquid, -
207         double Fn;      //two-phase friction factor
208         double Kf;      //two-phase number
209
210         //double h_f;
211         //double h_g;
212         //double rho_in;
213         //double rho_out;
214
215         //output variables
216         double Dp;     //frictional AND acceleration (or momentum) pressure drop, kPa
217
218         //flow
219
220         ////!!!!!!!!!!!!!!!!!!!as by original paper, only viscosity and quality account for
            presence of oil
221         //rho_in = PropsSI("D", "P", p_in * 1000, "Q", x_in / (1 - OMF), fluid);
222         //rho_out = PropsSI("D", "P", p_in * 1000, "Q", x_out / (1 - OMF), fluid);
223         //v_in = 1 / rho_in;
224         //v_out = 1 / rho_out;
225
226         //h_f = PropsSI("H", "P", p_in * 1000, "Q", 0, fluid) / 1000;
227         //h_g = PropsSI("H", "P", p_in * 1000, "Q", 1, fluid) / 1000;
228         //h_fg = h_g - h_f;
229         ////!!!!!!!!!!!!!!!!!!!
230
231         Re_fo = G_flux*Dh / mu_f;
232         Kf = (x_out - x_in)*h_fg*1000 / (length*9.81);
233         Fn = 0.00506*pow(Re_fo, -0.0951)*pow(Kf, 0.1554);
234
235         Dp = (((Fn*length*(v_out + v_in)) / Dh + (v_out - v_in))*pow(G_flux, 2)) / 1000;
236
237         return Dp;
238     }
239
240
```

```
241 double twoPhaseDp_HuDing2008(string fluid, double Dh, double length, double G_flux,
    double p_in,
242                             double x_in, double rho_f, double mu_f)
243 {
244     //use: calculate two-phase frictional pressure drop in a finned tube
245     //
246     //source: Hu, H., G. Ding, et al. (2008). Measurement and correlation
247     //        of frictional two-phase pressure drop of R410A-POE oil mixture
248     //        flow boiling in a 7mm straight micro-fin tube
249     //        Applied Thermal Engineering 28, 1272-1283
250     //
251     //author: Andrea Bigi
252     //date: 04/2016
253     //-----------------------------------------------------------------
254
255     //local variables
256     double rho_g;   //saturated vapor density, kg/m3
257     double mu_g;    //saturated vapor dynamic viscosity, N-s/m2, kg/m-s, Pa-s
258     double Re_go;   //Reynolds number, vapor only, -
259     double f_v;     //frictional coefficient for vapor single phase flow
260     double Dp_g;    //frictional pressure drop for vapor single phase flow
261     double Xtt;     //Martinelli parameter
262     double phi_g;   //vapor phase frictional multiplier
263     double a, n;    //correlation coefficients
264
265     //output variables
266     double Dp_fric;      //frictional pressure drop, kPa
267
268     //flow
269
270     a = 2.558;
271     n = 0.655;
272
273     rho_g = PropsSI("D", "P", p_in * 1000, "Q", 1, fluid);
274     mu_g = PropsSI("V", "P", p_in * 1000, "Q", 1, fluid);
275
276     Re_go = G_flux * x_in * Dh / mu_g;
277     f_v = 0.02 / pow(Re_go, 0.104);
278     Dp_g = 2 * length * f_v * pow(G_flux, 2) * pow(x_in, 2) / (rho_g * Dh);
279
280     Xtt = pow((1 - x_in) / x_in, 0.9) * pow(rho_g / rho_f, 0.5) * pow(mu_f / mu_g, 0.1);
        //this is the correct formula
281     phi_g = 1 + a * pow(Xtt, n);
282
283     Dp_fric = Dp_g * pow(phi_g, 2) / 1000;
284
285     return Dp_fric;
286 }
287
288 double twoPhaseDp_DingHu2009(string fluid, double Dh, double length, double G_flux,
    double p_in,
289     double x_in, double rho_f, double mu_f)
290 {
291     //use: calculate two-phase frictional pressure drop in a finned tube
292     //
293     //source: Ding, Hu, et al. (2009) - Experimental investigation and correlation
294     //      of two-phase frictional pressure drop of R410A-oil mixture flow boiling
295     //      in a 5 mm microfin tube, International Journal of Refrigeration 32, 150-161
296     //
297     //author: Andrea Bigi
298     //date: 04/2016
299     //-----------------------------------------------------------------
300
301     //local variables
302     double rho_g;   //saturated vapor density, kg/m3
303     double mu_g;    //saturated vapor dynamic viscosity, N-s/m2, kg/m-s, Pa-s
```

```
304         double Re_go;   //Reynolds number, vapor only, -
305         double f_v;     //frictional coefficient for vapor single phase flow
306         double Dp_g;    //frictional pressure drop for vapor single phase flow
307         double Xtt;     //Martinelli parameter
308         double phi_g;   //vapor phase frictional multiplier
309         double a, n;    //correlation coefficients
310
311         //output variables
312         double Dp_fric;      //frictional pressure drop, kPa
313
314         //flow
315
316         a = 1.892;
317         n = 0.587;
318
319         rho_g = PropsSI("D", "P", p_in * 1000, "Q", 1, fluid);
320         mu_g = PropsSI("V", "P", p_in * 1000, "Q", 1, fluid);
321
322         Re_go = G_flux * x_in * Dh / mu_g;
323         f_v = 0.128 / pow(Re_go, 0.267);
324         Dp_g = 2 * length * f_v * pow(G_flux, 2) * pow(x_in, 2) / (rho_g * Dh);
325
326         Xtt = pow((1 - x_in) / x_in, 0.9) * pow(rho_g / rho_f, 0.5) * pow(mu_f / mu_g, 0.1);
                //this is the correct formula
327         phi_g = 1 + a * pow(Xtt, n);
328
329         Dp_fric = Dp_g * pow(phi_g, 2) / 1000;
330
331         return Dp_fric;
332     }
```

```
1
2   #define _USE_MATH_DEFINES
3   #include <cmath>
4   #include "CoolProp.h"
5   #include "HumidAirProp.h"
6   #include <iostream>
7   #include <stdlib.h>
8   #include <fstream>
9   #include <string>
10  #include <cstddef>
11
12
13  using namespace CoolProp;
14  using namespace std;
15
16
17  double NanoDensity(string);
18
19
20  void Inventory(string fluid, string nano_mater, double D_nano, double SectA, double
    length,
21      double alpha, double w_local, double NMF, double rho_g, double rho_f,
22      double &Vol, double &RefLIQ, double &RefVAP, double &Ref, double &OilNano,
23      double &Nano, double &Oil, double &V_nano, double &N_nano)
24  {
25      //use: calculate inventory for the case of lubricant and refrigerant.
26      //      It needs to be mofified for the case when OMF = 0 && NMF != 0
27      //
28      //source:
29      //
30      //author: Andrea Bigi
31      //date: 02/2018
32      //-----------------------------------------------------------------
33
34      //local variables
35
36      //double Vol;       //volume, m3
37      //double RefLIQ;    //liquid refrigerant mass, kg
38      //double RefVAP;    //vapor refrigerant mass, kg
39      //double Ref;       //total refrigerant mass, kg
40      //double OilNano;   //nanooil mass, kg
41      //double Nano;  //nanoparticles mass, kg
42      //double Oil;       //oil mass, kg
43      //double V_nano;    //nanopartcile volume, m3
44      //double N_nano;    //number of nanoparticles, -
45      double rho_nano; //nanoparticle density, kg/m3
46
47      //output variables
48
49
50      //flow
51
52      rho_nano = NanoDensity(nano_mater);
53
54      Vol = SectA*length;
55      RefLIQ = Vol*(1 - alpha)*(1 - w_local)*rho_f;
56      RefVAP = Vol*alpha*rho_g;
57      Ref = RefLIQ + RefVAP;
58      OilNano = Vol*(1 - alpha)*w_local*rho_f;
59      Nano = OilNano*NMF;
60      Oil = OilNano*(1 - NMF);
61      V_nano = M_PI / 6 * pow(D_nano*1e-9, 3);
62      N_nano = Nano / (rho_nano*V_nano);
63
64  }
65
```

```
66
67  double RouhaniAxelsson(string fluid, double G_flux, double sigma, double rho_g, double
    rho_f, double x)
68  {
69      //use: calculate two-phase void fraction
70      //
71      //source: Rouhani, S. Z. and Axelsson, E. 1970.
72      // Calculation of void volume fraction in the subcooled and quality boiling regions.
73      // International Journal of Heat and Mass Transfer 13(2): 383-393.
74      //
75      //author: Andrea Bigi
76      //date: 03/2016
77      //-----------------------------------------------------------------
78
79      //local variables
80
81      //output variables
82      double epsilon;      //void fraction
83
84      //flow
85
86      epsilon = x / rho_g*pow((1 + 0.12*(1 - x))*(x / rho_g + (1 - x) / rho_f) +
87          ((1.18*(1 - x)*pow(9.81*sigma*(rho_f - rho_g), 0.25)) / (G_flux*pow(rho_f, 0.5
            ))), -1);
88
89      return epsilon;
90  }
```

```
1
2   #define _USE_MATH_DEFINES
3   #include <cmath>
4   #include "CoolProp.h"
5   #include "HumidAirProp.h"
6   #include <iostream>
7   #include <stdlib.h>
8   #include <fstream>
9   #include <string>
10  #include <cstddef>
11
12
13  using namespace CoolProp;
14  using namespace std;
15
16  double OilDensity(double);
17  double LocalOilMassFraction(double, double);
18  double OilMixtureBubbleTemp(double, double, string);
19  double OilMixtureBubbleTemp_Sawant(double, double, double, double, double, double,
    double, double, string);
20  double OilMixtureDensity(double, double, double);
21  double OilMixtureSpecificHeat(double, double, double);
22  double OilMixtureDynViscosity(double, double, double, string);
23  double OilMixtureConductivity(double, double, double);
24  double OilMixtureSurfTension(double, double, double);
25  double OilSpecificEnthalpy(double);
26  double NanoOilMassFraction(double, double, double, double);
27  double NanoOilDynViscosity(string, double, double, double);
28  double LocalNanoOilMassFraction(double, double);
29  double NanoOilMixtureDensity(string, double, double, double, double);
30  double NanoOilMixtureSpecificHeat(string, double, double, double, double);
31  double NanoOilMixtureDynViscosity(string, double, double, double, double, double);
32  double NanoOilMixtureConductivity(string, double, double, double, double);
33  double NanoOilMixtureSurfTension(double, double, double);
34
35  void CalcProps(string fluid, double A0, double A1, double A2, double a0, double b0,
36                 string oil, string nano_mater, string nano_shape, double D_nano, double
                   OMF, double NMF, double m_dot_fluid, double p_fluid, double h_fluid,
37                 double &t_fluid, double &x_fluid, double &rho_fluid, double &v_fluid,
                   double &cp_fluid, double &mu_fluid, double &k_fluid, double &sigma_fluid,
38                 double &t_sat_f, double &h_f, double &rho_f, double &v_f, double &cp_f,
                   double &mu_f, double &k_f, double &sigma_f,
39                 double &t_sat_g, double &h_g, double &rho_g, double &v_g, double &cp_g,
                   double &mu_g, double &k_g, double &sigma_g, double &h_fg,
40                 double &dh_x, double &t_sat, double &w_local)
41  {
42      //use: calculate fluid properties
43      //
44      //source: Thome, J. R. 1995. Comprehensive thermodynamic approach to modeling
45      // refrigerant-lubricating oil mixtures. HVAC&R Research 1(2): 110-125
46      //
47      // Youbi-Idrissi, M., Bonjour, J., Marvillet, C., & Meunier, F. (2003).
48      // Impact of refrigerant–oil solubility on an evaporator performances working with
        R-407C.
49      // International Journal of Refrigeration, 26(3), 284-292. Chicago
50      //
51      //author: Andrea Bigi
52      //date: 03/2016
53      //-----------------------------------------------------------------
54
55      //local variables
56      //double rho_fluid;    //inlet density, kg/m3
57      //double v_fluid;      //inlet specific volume, m3/kg
58      //double cp_fluid;     //inlet specific heat, kJ/kg-K
59      //double mu_fluid;     //inlet dynamic viscosity, N-s/m2, kg/m-s, Pa-s
60      double nu_fluid;       //inlet kinematik viscosity, mm2/s
```

```
61      //double k_fluid;      //inlet thermal conductivity, W/m-K
62      //double sigma_fluid;  //inlet surface tension, N/m
63      //double h_fg;         //vaporization enthalpy, kJ/kg
64      //double dh_x;         //delta enthalpy, kJ/kg
65
66      //double t_sat_f;     //saturated liquid temperature, C
67      //double h_f;         //saturated liquid enthalpy, kJ/kg
68      //double rho_f;       //saturated liquid density, kg/m3
69      //double v_f;         //saturated liquid specific volume, m3/kg
70      //double cp_f;        //saturated liquid specific heat, kJ/kg-K
71      //double mu_f;        //saturated liquid dynamic viscosity, N-s/m2, kg/m-s, Pa-s
72      double nu_f;          //saturated liquid kinematik viscosity, mm2/s
73      //double k_f;         //saturated liquid thermal conductivity, W/m-C
74      //double sigma_f;     //saturated liquid surface tension, N/m
75
76      //double t_sat_g;     //saturated vapor temperature, C
77      //double h_g;         //saturated vapor enthalpy, kJ/kg
78      //double rho_g;       //saturated vapor density, kg/m3
79      //double v_g;         //saturated vapor specific volume, m3/kg
80      //double cp_g;        //saturated vapor specific heat, kJ/kg-K
81      //double mu_g;        //saturated vapor dynamic viscosity, N-s/m2, kg/m-s, Pa-s
82      double nu_g;          //saturated vapor kinematik viscosity, mm2/s
83      //double k_g;         //saturated vapor thermal conductivity, W/m-C
84      //double sigma_g;     //saturated vapor surface tension, N/m
85
86      //double w_local;     //local oil mass fraction
87      double t_bub;         //mixture bubble temperature, C
88
89      double OMFnano;       //nanoparticles mass fraction in refrigerant-nanooil mixture
90
91
92      //output variables
93
94      //flow
95
96      w_local = 0;
97      p_fluid = p_fluid * 1000;
98      h_fluid = h_fluid * 1000;
99
100     //inlet conditions
101     x_fluid = PropsSI("Q", "P", p_fluid, "H", h_fluid, fluid);
102     t_fluid = PropsSI("T", "P", p_fluid, "H", h_fluid, fluid) - 273.15;
103     if (fluid == "R410A")
104     {
105         t_fluid = OilMixtureBubbleTemp_Sawant(A0, A1, A2, a0, b0, p_fluid, 0, x_fluid,
            fluid);  //uses thermodynamic quality as input
106     }
107     rho_fluid = PropsSI("D", "P", p_fluid, "H", h_fluid, fluid);
108     v_fluid = 1 / rho_fluid;
109     cp_fluid = PropsSI("C", "P", p_fluid, "H", h_fluid, fluid)/1000;
110     mu_fluid = PropsSI("V", "P", p_fluid, "H", h_fluid, fluid);
111     nu_fluid = mu_fluid / rho_fluid * 1000000;
112     k_fluid = PropsSI("L", "P", p_fluid, "H", h_fluid, fluid);
113     sigma_fluid = PropsSI("I", "P", p_fluid, "H", h_fluid, fluid);
114
115     //saturation conditions (saturated liquid)
116     t_sat_f = PropsSI("T", "P", p_fluid, "Q", 0, fluid) - 273.15;
117     h_f = PropsSI("H", "P", p_fluid, "Q", 0, fluid) / 1000;
118     rho_f = PropsSI("D", "P", p_fluid, "Q", 0, fluid);
119     v_f = 1 / rho_f;
120     cp_f = PropsSI("C", "P", p_fluid, "Q", 0, fluid) / 1000;
121     mu_f = PropsSI("V", "P", p_fluid, "Q", 0, fluid);
122     nu_f = mu_f / rho_f * 1000000;
123     k_f = PropsSI("L", "P", p_fluid, "Q", 0, fluid);
124     sigma_f = PropsSI("I", "P", p_fluid, "Q", 0, fluid);
125
```

```
126         //saturation conditions (saturated vapor)
127         t_sat_g = PropsSI("T", "P", p_fluid, "Q", 1, fluid) - 273.15;
128         h_g = PropsSI("H", "P", p_fluid, "Q", 1, fluid) / 1000;
129         rho_g = PropsSI("D", "P", p_fluid, "Q", 1, fluid);
130         v_g = 1 / rho_g;
131         cp_g = PropsSI("C", "P", p_fluid, "Q", 1, fluid) / 1000;
132         mu_g = PropsSI("V", "P", p_fluid, "Q", 1, fluid);
133         nu_g = mu_g / rho_g * 1000000;
134         k_g = PropsSI("L", "P", p_fluid, "Q", 1, fluid);
135         sigma_g = PropsSI("I", "P", p_fluid, "Q", 1, fluid);
136
137         h_fg = h_g - h_f;
138         //t_sat = (t_sat_f + t_sat_g)/2;
139         t_sat = t_fluid;
140
141         if (x_fluid < 0)
142         {
143             x_fluid = 0;
144         }
145
146         //calculate mixture LIQUID properties
147         if ((OMF > 0) && (NMF == 0))
148         {
149             w_local = LocalOilMassFraction(OMF, x_fluid);
150             t_fluid = OilMixtureBubbleTemp(p_fluid, w_local, fluid);
151             if (fluid == "R410A")
152             {
153                 t_fluid = OilMixtureBubbleTemp_Sawant(A0, A1, A2, a0, b0, p_fluid, w_local,
                    x_fluid, fluid);  //uses thermodynamic quality as input
154             }
155             x_fluid = x_fluid * (1 - OMF);  //x_mix
156             rho_f = OilMixtureDensity(t_fluid, w_local, rho_f);
157             v_f = 1 / rho_f;
158             cp_f = OilMixtureSpecificHeat(t_fluid, w_local, cp_f);
159             mu_f = OilMixtureDynViscosity(t_fluid, w_local, mu_f, fluid);
160             nu_f = mu_f / rho_f * 1000000;
161             k_f = OilMixtureConductivity(t_fluid, w_local, k_f);
162             sigma_f = OilMixtureSurfTension(t_fluid, w_local, sigma_f);
163
164             ////see Thome(1995) (given that (x_min_MAX=1 - OMF))
165             dh_x = h_fg*(x_fluid - 0) + (1 - x_fluid)*cp_f*(t_fluid - t_sat_g) + x_fluid*
                cp_g*(t_fluid - t_sat_g);
166             h_fg = h_fg*((1 - OMF) - 0) + (1 - (1 - OMF))*cp_f*(t_fluid - t_sat_g) + (1 -
                OMF)*cp_g*(t_fluid - t_sat_g);
167             h_f = (1 - OMF) * h_f + OMF * OilSpecificEnthalpy(t_fluid);
168             h_g = h_f + h_fg;
169
170             ////derived from Youbi-Idrissi(2003) but without solubility curve
171             //h_f = (1 - OMF) * h_f + OMF * OilSpecificEnthalpy(t_fluid);
172             //h_g = (1 - OMF) * h_g + OMF * OilSpecificEnthalpy(t_fluid);
173             //h_fg = h_g - h_f;
174
175             t_sat = t_fluid;
176
177             //calculation of new inlet conditions
178             v_fluid = (1 - x_fluid)*v_f + x_fluid*v_g;
179             //rho_fluid = 1 / v_fluid;
180         }
181         if ((OMF > 0) && (NMF > 0))
182         {
183             OMFnano = NanoOilMassFraction(OMF, NMF, m_dot_fluid, t_fluid);   //in the end,
                this should be same as the input OMF
184             w_local = LocalNanoOilMassFraction(OMFnano, x_fluid);
185             t_fluid = OilMixtureBubbleTemp(p_fluid, w_local, fluid);    //assumed same as
                case without nanoparticles
186             if (fluid == "R410A")
```

```
187         {
188             t_fluid = OilMixtureBubbleTemp_Sawant(A0, A1, A2, a0, b0, p_fluid, w_local,
                x_fluid, fluid);  //uses thermodynamic quality
189         }
190         x_fluid = x_fluid * (1 - OMFnano);  //x_mix
191         rho_f = NanoOilMixtureDensity(nano_mater, NMF, t_fluid, w_local, rho_f);
192         v_f = 1 / rho_f;
193         cp_f = NanoOilMixtureSpecificHeat(nano_mater, NMF, t_fluid, w_local, cp_f);
194         mu_f = NanoOilMixtureDynViscosity(nano_mater, D_nano, NMF, t_fluid, w_local,
            mu_f);
195         nu_f = mu_f / rho_f * 1000000;
196         k_f = NanoOilMixtureConductivity(nano_mater, NMF, t_fluid, w_local, k_f);
197         sigma_f = NanoOilMixtureSurfTension(t_fluid, w_local, sigma_f);
198
199         //see Thome(1995) (given that (x_min_MAX=1 - OMF))
200         dh_x = h_fg*(x_fluid - 0) + (1 - x_fluid)*cp_f*(t_fluid - t_sat_g) + x_fluid*
            cp_g*(t_fluid - t_sat_g);
201         h_fg = h_fg*((1 - OMFnano) - 0) + (1 - (1 - OMFnano))*cp_f*(t_fluid - t_sat_g) +
             (1 - OMFnano)*cp_g*(t_fluid - t_sat_g);
202         h_f = (1 - OMFnano) * h_f + OMFnano * OilSpecificEnthalpy(t_fluid);
203         h_g = h_f + h_fg;
204
205         ////derived from Youbi-Idrissi(2003) but without solubility curve
206         //h_f = (1 - OMFnano) * h_f + OMFnano * OilSpecificEnthalpy(t_fluid);
207         //h_g = (1 - OMFnano) * h_g + OMFnano * OilSpecificEnthalpy(t_fluid);
208         //h_fg = h_g - h_f;
209
210         t_sat = t_fluid;
211
212         //calculation of new inlet conditions
213         v_fluid = (1 - x_fluid)*v_f + x_fluid*v_g;
214         //rho_fluid = 1 / v_fluid;
215     }
216
217     if (x_fluid < 0)
218     {
219         x_fluid = 0;
220     }
221
222 }
223
224
225
226 //pure oil properties - START
227 //============================================================================//
228 double OilMolecMass = 570; // g/mol - for EMKARATE POE RL32-3MAF
229
230 double OilSurfTension(double temp)
231 {
232     //use: calculate pure oil surface tension, N/m
233     //
234     //source: Hu, Hai-tao, Guo-liang Ding, and Kai-jian Wang.
235     // Measurement and correlation of frictional two-phase pressure drop of R410A/POE
        oil mixture
236     // flow boiling in a 7mm straight micro-fin tube.
237     // Applied Thermal Engineering 28, no. 11 (2008): 1272-1283.
238     //
239     //author: Andrea Bigi
240     //date: 02/2016
241     //--------------------------------------------------------------------
242
243     //local variables
244
245     //output variables
246     double sigma_oil;    //oil surface tension, N/m
247
```

```
248         //flow
249
250         sigma_oil = (29 - 0.4*temp) / 1000;
251
252         return sigma_oil;
253     }
254
255     double OilConductivity(double temp)
256     {
257         //use: calculate pure oil thermal conductivity, W/m-C
258         //
259         //source: Lottin, O., Guillemet, P. and Lebreton, J.-M. 2003.
260         // Effects of synthetic oil in a compression refrigeration system using R410A.
261         // Part I: modelling of the whole system and analysis of its response to an increase
262         // in the amount of circulating oil. International Journal of Refrigeration 26(7):
            772-782
263         //
264         //author: Andrea Bigi
265         //date: 02/2016
266         //---------------------------------------------------------------------
267
268         //local variables
269         double temp_ref;  //reference temperature, C
270         double rho_oil;   //pure oil density, kg/m3
271         double rho_water; //water density, kg/m3
272
273         //output variables
274         double k_oil;    //oil thermal conductivity, W/m-C
275
276         //flow
277         temp_ref = 15.6;
278
279         rho_water = PropsSI("D", "T", temp_ref + 273.15, "Q", 0, "Water");
280         rho_oil = OilDensity(temp);
281
282         //from Lottin(2003)
283         k_oil = 0.1172*(1-0.0054 * temp) / (rho_oil/rho_water);
284
285         //correlation from experimental measurements at Oklahoma State University lab
286         k_oil = (6e-6)*pow(temp, 2) - 0.0006*temp + 0.1513;
287
288         return k_oil;
289     }
290
291     double OilDensity(double temp)
292     {
293         //use: calculate pure oil density, kg/m3
294         //
295         //source: Honeywell confidential document on EMKARATE POE RL32-3MAF properties
296         //
297         // Kedzierski, 2013, Viscosity and density of aluminum oxide nanolubricant,
298         // International Journal of Refrigeration (36), 1333-1340
299         //
300         // Kedzierski, Brignoli, Quine, Brown, 2017, Viscosity, density, and thermal
            conductivity of aluminum oxide
301         // and zinc oxide nanolubricants, International Journal of Refrigeration (74), 3-11
302         //
303         //author: Andrea Bigi
304         //date: 02/2016
305         //---------------------------------------------------------------------
306
307         //local variables
308         double d1, d2, d3; //constant from Honeywell confidential document
309
310         //output variables
311         double rho_oil;    //oil density, kg/m3
```

```
312
313         //flow
314
315         //from Honeywell confidential document (EMKARATE POE RL32-3MAF)
316         //%%%;
317
318         ////from Kedzierski(2013) (oil POE RL68H)
319         //rho_oil = 1/(0.7979e-3 + 0.7647e-6 * (temp + 273.15));
320
321         //from Kedzierski(2017) (oil POE RL32-3MAF)
322         rho_oil = 1/(0.7972e-3 + 0.2003e-3 * (temp + 273.15));
323
324         return rho_oil;
325     }
326
327     double OilKinViscosity(double temp)
328     {
329         //use: calculate pure oil kinematic viscosity, mm2/s (cSt)
330         //
331         //source: Honeywell confidential document on EMKARATE POE RL32-3MAF properties
332         //
333         // Hu, H., G. Ding, et al. (2008). Measurement and correlation
334         // of frictional two-phase pressure drop of R410A-POE oil mixture
335         // flow boiling in a 7mm straight micro-fin tube
336         // Applied Thermal Engineering 28, 1272–1283
337         //
338         // Kedzierski, 2013, Viscosity and density of aluminum oxide nanolubricant,
339         // International Journal of Refrigeration (36), 1333-1340
340         //
341         // Kedzierski, Brignoli, Quine, Brown, 2017, Viscosity, density, and thermal
            conductivity of aluminum oxide
342         // and zinc oxide nanolubricants, International Journal of Refrigeration (74), 3-11
343         //
344         //author: Andrea Bigi
345         //date: 02/2016
346         //---------------------------------------------------------------------
347
348         //local variables
349         double phi;        //constant from Honeywell confidential document
350         double a1, a2, a3; //constant from Honeywell confidential document
351         double HW, HWguess;//Honeywell equation parameters
352
353         //output variables
354         double nu_oil;    //oil kinematic viscosity, mm2/s (cSt)
355
356         //flow
357
358         //assuming linearity, for EMKARATE POE RL32-3MAF (from Emkarate property sheet)
359         nu_oil = -0.4233*temp + 48.133;
360
361         //from Honeywell confidential document (EMKARATE POE RL32-3MAF)
362         //%%%;
363
364         ////from Hu(2008)
365         //nu_oil = 1062.075 * exp(-temp/32.39)+4.90664;
366
367         ////from Kedzierski(2013) (oil POE RL68H)
368         //nu_oil = exp(-52.1912 + 58.8418 / ((temp + 273.15) / 273.15) + 36.8165 *
            log((temp + 273.15) / 273.15));
369
370         ////from Kedzierski(2017) (oil POE RL32-3MAF)
371         //nu_oil = exp(-45.0487 + 50.5360 / ((temp + 273.15) / 273.15) + 31.9522 *
            log((temp + 273.15) / 273.15));
372
373         return nu_oil;
374     }
```

```
375
376 double OilDynViscosity(double temp)
377 {
378     //use: calculate pure oil dynamic viscosity, N-s/m2, kg/m-s, Pa-s
379     //
380     //source:
381     //
382     //author: Andrea Bigi
383     //date: 02/2016
384     //---------------------------------------------------------------------
385
386     //local variables
387     double nu_oil;    //oil kinematic viscosity, mm2/s (cSt)
388     double rho_oil;    //oil density, kg/m3
389
390     //output variables
391     double mu_oil;    //oil dynamic viscosity, N-s/m2, kg/m-s, Pa-s
392
393     //flow
394
395     nu_oil = OilKinViscosity(temp);
396     rho_oil = OilDensity(temp);
397
398     mu_oil = nu_oil*1000000*rho_oil;
399
400     return mu_oil;
401 }
402
403 double OilSpecificHeat(double temp)
404 {
405     //use: calculate pure oil specific heat, kJ/kg-K
406     //
407     //source: Thome, J. R. 1995. Comprehensive thermodynamic approach to modeling
408     // refrigerant-lubricating oil mixtures. HVAC&R Research 1(2): 110-125
409     //
410     //author: Andrea Bigi
411     //date: 02/2016
412     //---------------------------------------------------------------------
413
414     //local variables
415     double temp_ref;  //reference temperature, C
416     double rho_oil;   //pure oil density, kg/m3
417     double rho_water; //water density, kg/m3
418
419     //output variables
420     double cp_oil;    //oil specific heat, kJ/kg-K
421
422     temp_ref = 15.56;
423
424     rho_water = PropsSI("D", "T", temp_ref+273.15, "Q", 0, "Water");
425     rho_oil = OilDensity(temp);
426     cp_oil = 4.186 * (0.388 + 0.00045 * (1.8 * temp + 32)) / sqrt(rho_oil / rho_water);
427
428     return cp_oil;
429 }
430
431 double OilSpecificEnthalpy(double temp)
432 {
433     //use: calculate pure oil specific enthalpy, kJ/kg
434     //
435     //source: Lottin, O., Guillemet, P. and Lebreton, J.-M. 2003.
436     // Effects of synthetic oil in a compression refrigeration system using R410A.
437     // Part I: modelling of the whole system and analysis of its response to an increase
438     // in the amount of circulating oil. International Journal of Refrigeration 26(7):
    772-782
439     //
```

```
440         //author: Andrea Bigi
441         //date: 02/2016
442         //---------------------------------------------------------------------
443
444         //local variables
445         double temp_ref;  //reference temperature, C
446         double rho_oil;   //pure oil density, kg/m3
447         double rho_water; //water density, kg/m3
448
449         //output variables
450         double h_oil;    //oil specific enthalpy, kJ/kg
451
452         temp_ref = 15.56;
453
454         rho_water = PropsSI("D", "T", temp_ref + 273.15, "Q", 0, "Water");
455         rho_oil = OilDensity(temp);
456         h_oil = 4.186 * (0.4024 * temp + 0.000405 * pow(temp, 2)) / sqrt(rho_oil / rho_water
            );
457
458         return h_oil;
459     }
460
461     //pure oil properties - FINISH
462     //=============================================================================//
463
464
465     //refrigerant-oil mixture properties - START
466     //=============================================================================//
467
468     double LocalOilMassFraction(double OMF, double x_fluid)
469     {
470         //use: calculate local oil mass fraction, given absolute oil mass fraction
471         //
472         //source:
473         //
474         //author: Andrea Bigi
475         //date: 11/2015
476         //---------------------------------------------------------------------
477
478         //local variables
479         double x_mix;        //oil-refrigerant mixture quality
480
481         //output variables
482         double w_local;      //local oil mass fraction
483
484         //flow
485
486         x_mix = x_fluid * (1 - OMF);
487         w_local = OMF / (1 - x_mix);
488
489         return w_local;
490     }
491
492     double OilMixtureSurfTension(double temp, double w_local, double sigma_f_ref)
493     {
494         //use: calculate refrigerant-oil mixture surface tension, N/m
495         //
496         //source: Jensen, M. and Jackman, D. 1984.
497         // Prediction of nucleate pool boiling heat transfer coefficients of
498         // refrigerant-oil mixtures. Journal of heat transfer 106(1): 184-190
499         //
500         //author: Andrea Bigi
501         //date: 11/2015
502         //---------------------------------------------------------------------
503
504         //local variables
```

```
505         double sigma_oil;   //oil surface tension, N/m
506
507         //output variables
508         double sigma_mix;   //refrigerant-oil mixture surface tension, N/m
509
510         //flow
511
512         sigma_oil = OilSurfTension(temp);
513         sigma_mix = sigma_f_ref + (sigma_oil - sigma_f_ref)*sqrt(w_local);
514
515         return sigma_mix;
516     }
517
518     double OilMixtureConductivity(double temp, double w_local, double k_f_ref)
519     {
520         //use: calculate refrigerant-oil mixture thermal conductivity, W/m-K
521         //
522         //source: Filippov, L. and Novoselova, N. 1955.
523         // The thermal conductivity of solutions of normal liquid
524         // Chem Abstr. 49: 37-40
525         //
526         //author: Andrea Bigi
527         //date: 02/2016
528         //-----------------------------------------------------------------
529
530         //local variables
531         double k_oil;   //oil thermal conductivity, W/m-K
532
533         //output variables
534         double k_mix;   //refrigerant-oil mixture thermal conductivity, W/m-K
535
536         //flow
537
538         k_oil = OilConductivity(temp);
539         k_mix = k_f_ref*(1 - w_local) + k_oil*w_local - 0.72*(k_oil - k_f_ref)*(1 - w_local
            )*w_local;
540
541         return k_mix;
542     }
543
544     double OilMixtureDensity(double temp, double w_local, double rho_f_ref)
545     {
546         //use: calculate refrigerant-oil mixture density, kg/m3
547         //
548         //source: Jensen, M. and Jackman, D. 1984.
549         // Prediction of nucleate pool boiling heat transfer coefficients of
550         // refrigerant-oil mixtures. Journal of heat transfer 106(1): 184-190
551         //
552         //author: Andrea Bigi
553         //date: 11/2015
554         //-----------------------------------------------------------------
555
556         //local variables
557         double rho_oil;   //oil density, kg/m3
558
559         //output variables
560         double rho_mix;   //refrigerant-oil mixture density, kg/m3
561
562         //flow
563
564         rho_oil = OilDensity(temp);
565         rho_mix = 1 / (w_local / rho_oil + (1 - w_local) / rho_f_ref);
566
567         return rho_mix;
568     }
569
```

```
570  double OilMixtureKinViscosity(double mu_mix, double rho_mix)
571  {
572      //use: calculate refrigerant-oil mixture kinematic viscosity, mm2/s (cSt)
573      //
574      //source:
575      //
576      //author: Andrea Bigi
577      //date:
578      //---------------------------------------------------------------
579
580      //local variables
581
582      //output variables
583      double nu_mix;   //refrigerant-oil mixture kinematic viscosity, mm2/s
584
585      //flow
586
587      nu_mix = mu_mix / rho_mix *1000000;
588
589      return nu_mix;
590  }
591
592  double OilMixtureDynViscosity(double temp, double w_local, double mu_f_ref, string fluid)
593  {
594      //use: calculate refrigerant-oil mixture dynamic viscosity, N-s/m2, kg/m-s, Pa-s
595      //
596      //source: Yokozeki, A. 1994. Solubility and viscosity of refrigerant-oil mixtures.
597      // 5th International Refrigeration Conference at Purdue.
598      // West Lafayette, IN (USA): 335-340
599      //
600      // Thome, J.R., 2004. Engineering Data Book III. Wolverine Tube Inc.
601      // http://www.wlv.com/products/databook/db3/DataBookIII.pdf (Chapters 15 and 16)
602      //
603      //author: Andrea Bigi
604      //date: 11/2015
605      //---------------------------------------------------------------
606
607      //local variables
608      double KK;    //empirical constant
609      double nu_oil;    //oil kinematic viscosity, mm2/s
610      double mu_oil;    //oil dynamic viscosity, N-s/m2, kg/m-s, Pa-s
611      double rho_oil;   //oil density, kg/m3
612      double molec_mass_oil;  //oil molecular mass, g/mol
613      double molec_mass_ref;  //refrigerant molecular mass, g/mol
614      double mole_frac_oil;  //oil mole fraction
615      double mole_frac_ref;  //refrigerant mole fraction
616      double Xi_oil;     //oil Yokozeki factor
617      double Xi_ref;     //liquid refrigerant Yokozeki factor
618
619      //output variables
620      double mu_mix;   //refrigerant-oil mixture dynamic viscosity, N-s/m2, kg/m-s, Pa-s
621
622      //flow
623
624      KK = 0.58;
625
626      molec_mass_oil = OilMolecMass;
627      molec_mass_ref = PropsSI("M", "T", temp, "Q", 0, fluid) * 1000;
628      mole_frac_oil = w_local*(molec_mass_ref / molec_mass_oil) / (1 - w_local + w_local*(
         molec_mass_ref / molec_mass_oil));
629      mole_frac_ref = 1 - mole_frac_oil;
630      Xi_oil = pow(molec_mass_oil, KK)*mole_frac_oil / (pow(molec_mass_ref, KK)*
         mole_frac_ref + pow(molec_mass_oil, KK)*mole_frac_oil);     //Oil Yokozeki Factor
631      Xi_ref = pow(molec_mass_ref, KK)*mole_frac_ref / (pow(molec_mass_ref, KK)*
         mole_frac_ref + pow(molec_mass_oil, KK)*mole_frac_oil);     //Liquid Refrigerant
         Yokozeki Factor
```

```
632
633         nu_oil = OilKinViscosity(temp);
634         rho_oil = OilDensity(temp);
635         mu_oil = nu_oil * rho_oil / 1000000;
636
637         mu_mix = exp(Xi_ref*log(mu_f_ref) + Xi_oil*log(mu_oil));
638
639         ////from Thome(2004)
640         //mu_mix = pow(mu_f_ref, 1 - w_local)*pow(mu_oil, w_local);
641
642         return mu_mix;
643
644     }
645
646     double OilMixtureSpecificHeat(double temp, double w_local, double cp_f_ref)
647     {
648         //use: calculate refrigerant-oil mixture specific heat, kJ/kg-K
649         //
650         //source: Jensen, M. and Jackman, D. 1984.
651         // Prediction of nucleate pool boiling heat transfer coefficients of
652         // refrigerant-oil mixtures. Journal of heat transfer 106(1): 184-190
653         //
654         //author: Andrea Bigi
655         //date: 11/2015
656         //-----------------------------------------------------------------
657
658         //local variables
659         double cp_oil;   //oil specific heat, kJ/kg-K
660
661         //output variables
662         double cp_mix;   //refrigerant-oil mixture specific heat, kJ/kg-K
663
664         //flow
665
666         cp_oil = OilSpecificHeat(temp);
667         cp_mix = w_local*cp_oil + (1 - w_local)*cp_f_ref;
668
669         return cp_mix;
670     }
671
672     double OilMixtureBubbleTemp(double p_sat, double w_local, string fluid)
673     {
674         //use: calculate refrigerant-oil mixture bubble temperature, C
675         //
676         //source: Thome, J. R. 1995. Comprehensive thermodynamic approach to modeling
677         // refrigerant-lubricating oil mixtures. HVAC&R Research 1(2): 110-125
678         //
679         //author: Andrea Bigi
680         //date: 11/2015
681         //-----------------------------------------------------------------
682
683         //local variables
684         //empirical coefficients
685         double a1 = 182.52;
686         double a2 = -724.21;
687         double a3 = 3868.0;
688         double a4 = -5268.9;
689         double b1 = -0.72212;
690         double b2 = 2.3914;
691         double b3 = -13.779;
692         double b4 = 17.066;
693
694         double a0;    //correlation coefficient
695         double b0;    //correlation coefficient
696         double AA;    //correlation coefficient
697         double BB;    //correlation coefficient
```

```
698         double Dp = 0.0001;    //delta pressure, MPa
699         double p_sat1;    //perturbed saturation pressure, MPa
700         double p_sat2;    //perturbed saturation pressure, MPa
701         double t_sat1;    //perturbed saturation temperature, K
702         double t_sat2;    //perturbed saturation temperature, K
703
704         //output variables
705         double Tbub_mix;    //refrigerant-oil bubble temperature, C
706
707         //flow
708
709         //Thome(1995)
710         p_sat1 = p_sat / 1000000 + Dp;
711         p_sat2 = p_sat / 1000000 - Dp;
712         t_sat1 = PropsSI("T", "P", p_sat1 * 1000000, "Q", 1, fluid);
713         t_sat2 = PropsSI("T", "P", p_sat2 * 1000000, "Q", 1, fluid);
714
715         b0 = (log(p_sat1)*(t_sat1 / t_sat2) - log(p_sat2)) / ((t_sat1 / t_sat2) - 1);
716         a0 = t_sat1*(log(p_sat1) - b0);
717         AA = a0 + a1*w_local + a2*pow(w_local, 3) + a3*pow(w_local, 5) + a4*pow(w_local, 7);
718         BB = b0 + b1*w_local + b2*pow(w_local, 3) + b3*pow(w_local, 5) + b4*pow(w_local, 7);
719
720         Tbub_mix = AA / (log(p_sat/1000000) - BB) - 273.15;
721
722         return Tbub_mix;
723     }
724
725     double OilMixtureBubbleTemp_Sawant(double A0, double A1, double A2, double a0, double b0
        , double p_sat, double w_local, double x_fluid, string fluid)
726     {
727         //use: calculate refrigerant-oil mixture bubble temperature, C
728         //
729         //source: Thome, J. R. 1995. Comprehensive thermodynamic approach to modeling
730         // refrigerant-lubricating oil mixtures. HVAC&R Research 1(2): 110-125
731         //
732         // Sawant, Kedzierski, Brown, 2007. Effect of Lubricant on R410A Horizontal
733         // Flow Boiling. NISTIR 7456
734         //
735         //author: Andrea Bigi
736         //date: 11/2015
737         //-----------------------------------------------------------------
738
739         //local variables
740         //empirical coefficients
741         double a1 = 182.52;
742         double a2 = -724.21;
743         double a3 = 3868.0;
744         double a4 = -5268.9;
745         double b1 = -0.72212;
746         double b2 = 2.3914;
747         double b3 = -13.779;
748         double b4 = 17.066;
749
750         double AA;    //correlation coefficient
751         double BB;    //correlation coefficient
752
753         //output variables
754         double Tbub_mix;    //refrigerant-oil bubble temperature, C
755
756         //flow
757
758         ////Sawant(2007)
759         //A0 = 0.658452e-2;
760         //A1 = 0.434741e-3;
761         //A2 = -0.129204e-5;
762         //a0 = -2300.2;
```

```
763         //b0 = 15.146;
764
765         ////from interpolation of experimental data of pure refrigerant R410A  (2015
            microfin tube series: 1-28)
766         //A0 = 0.00774789631085859;
767         //A1 = -0.000605742908554461;
768         //A2 = -4.20111057045546E-06;
769         //a0 = -1650.94903353027;
770         //b0 = 12.7910297482209;
771
772         ////from interpolation of experimental data of pure refrigerant R410A  (2015
            microfin tube series: 29-63)
773         //A0 = 0.00448285188977577;
774         //A1 = -0.000128240216365056;
775         //A2 = -4.20111057045546E-06;
776         //a0 = -5719.02974975048;
777         //b0 = 27.4591324204553;
778
779         AA = a0 + a1*w_local + a2*pow(w_local, 3) + a3*pow(w_local, 5) + a4*pow(w_local, 7);
               //the power values are the same of Thome(1995)
780         BB = b0 + b1*w_local + b2*pow(w_local, 3) + b3*pow(w_local, 5) + b4*pow(w_local, 7);
               //the power values are the same of Thome(1995)
781
782         Tbub_mix = AA / (log(p_sat/1000) - BB + A2 / A1 * x_fluid) - 273.15;
783
784         return Tbub_mix;
785     }
786
787     double OilMixtureEnthalpy(double temp, double OMF, double x_fluid, double p_sat, double
        delta_h, string fluid)
788     {
789         //use: calculate refrigerant-oil mixture enthalpy, kJ/kg
790         //
791         //source: Thome, J. R. 1995. Comprehensive thermodynamic approach to modeling
792         // refrigerant-lubricating oil mixtures. HVAC&R Research 1(2): 110-125
793         //
794         //Schwentker, R.A. (2005). Advances to computer model used in the
795         //simulation and optimization of heat exchangers. MS thesis.
796         //The Univeristy of Maryland, College Park
797         //
798         //author: Andrea Bigi
799         //date: 11/2015
800         //-----------------------------------------------------------------
801
802         //local variables
803         double x_mix_in;  //refrigerant-oil mixture quality at the inlet of the segment
804         double x_mix_out; //refrigerant-oil mixture quality at the outlet of the segment
805         double x_mix;     //refrigerant-oil mixture quality segment average
806         double x_min;    //minimum quality
807         double x_max;    //maximum quality
808         double w_local_in;  //local oil mass fraction at the inlet of the segment
809         double w_local_out;  //local oil mass fraction at the outlet of the segment
810         double w_local;   //local oil mass fraction segment average
811         double Tbub_in;   //bubble temperature at the inlet of the segment, C
812         double Tbub_out;    //bubble temperature at the outlet of the segment, C
813         double h_f;     //saturated liquid enthalpy, J/kg
814         double h_g;     //saturated vapor enthalpy, J/kg
815         double h_fg;    //vaporization enthalpy, kJ/kg
816         double cp_f;    //liquid refrigerant specific heat, kJ/kg-K
817         double cp_g;    //vapor refrigerant specific heat, kJ/kg-K
818         double cp_mix;    //refrigerant-oil mixture specific heat, kJ/kg-K
819         double delta_h_calc;  //calculated enthalpy difference accross the segment, kJ/kg
820
821         //output variables
822         double x_out;    //refrigerant quality at the outlet of the segment
823
```

```
824         //flow
825 
826         x_min = 0;
827         x_max = 1 - OMF;
828 
829         w_local_in = OMF / (1 - x_fluid);
830         Tbub_in = temp;
831         x_mix_in = x_fluid;
832 
833             do
834             {
835                 x_out = (x_min + x_max) / 2;  //first guess of outlet quality
836 
837                 w_local_out = LocalOilMassFraction(OMF, x_out);
838                 Tbub_out = OilMixtureBubbleTemp(w_local_out, p_sat, fluid);
839                 x_mix_out = x_out * (1 - OMF);
840 
841                 x_mix = (x_mix_in + x_mix_out) / 2;
842                 w_local = (w_local_in + w_local_out) / 2;
843 
844                 h_f = PropsSI("H", "P", p_sat, "Q", 0, fluid);
845                 h_g = PropsSI("H", "P", p_sat, "Q", 1, fluid);
846                 h_fg = (h_g - h_f) / 1000;
847                 cp_g = PropsSI("C", "P", p_sat, "Q", 1, fluid) / 1000;
848                 cp_f = PropsSI("C", "P", p_sat, "Q", 0, fluid) / 1000;
849                 cp_mix = OilMixtureSpecificHeat(temp, w_local, cp_f);
850 
851                 delta_h_calc = h_fg*(x_mix_out - x_mix_in) + (1 - x_mix)*cp_mix*(Tbub_out -
                    Tbub_in) + x_mix*cp_g*(Tbub_out - Tbub_in);
852 
853                 if (delta_h_calc > delta_h)   //comparison between the calculated delta_h
                    and the actual delta_h from the input variables
854                 {
855                     x_max = x_out;
856                 }
857                 else
858                 {
859                     x_min = x_out;
860                 }
861 
862             } while (abs(delta_h_calc - delta_h) > 0.01);
863 
864         return x_out;
865     }
866 
867     //refrigerant-oil mixture properties - FINISH
868     //============================================================================//
869 
870 
871     //nanoparticle properties - START
872     //============================================================================//
873 
874     double NanoDensity(string nano_mater)
875     {
876         //use: calculate nanoparticle density, kg/m3
877         //
878         //source: Kedzierski, 2013, Viscosity and density of aluminum oxide nanolubricant,
879         // International Journal of Refrigeration (36), 1333-1340
880         //
881         // Kedzierski M., Brignoli R., Quine K., Brown J., 2016, Viscosity, density and
            thermal conductivity
882         // of aluminum oxide and zinc oxide nanolubricants, International Journal of
            Refrigeration (),
883         //
884         //author: Andrea Bigi
885         //date: 08/2016
```

```
886         //-----------------------------------------------------------------
887
888         //local variables
889
890         //output variables
891         double rho_nano;        //nanoparticle density, kg/m3
892
893         //flow
894
895         if (nano_mater == "Al2O3")
896         {
897             rho_nano = 3600;    //3880
898         }
899         else if (nano_mater == "ZnO")
900         {
901             rho_nano = 5610;
902         }
903
904         return rho_nano;
905     }
906
907     double NanoConductivty(string nano_mater, double temp)
908     {
909         //use: calculate nanoparticle thermal conductivity, W/m-K
910         //
911         //source: Pertti Auerkari, 1996, "Mechanical and physical properties of engineering
912         // alumina ceramics", Espoo 1996, Technical Research Center of Finland,
913         // VTT Tiedotteita - Meddelanden - Research Notes 1792. 26 p.pag.8
914         //
915         // Touloukian, 1966, Thermophysical Properties of High Temperature Solid Materials,
916         // Vol 4., Pt 1, Sect 1, pp. 8 - 47
917         // http://www-ferp.ucsd.edu/LIB/PROPS/PANOS/al2o3.html
918         //
919         //author: Andrea Bigi
920         //date: 03/2016
921         //-----------------------------------------------------------------
922
923         //local variables
924
925         //output variables
926         double k_nano;          //nanoparticle thermal conductivity, W/m-K
927
928         //flow
929
930         if (nano_mater == "Al2O3")
931         {
932             //Pertti Auerkari
933             k_nano = 5.5 + 34.5*exp(-0.0033*(temp));
934
935             //Touloukian
936             //k_Al2O3 = 85.868 - 0.22972*(temp + 273.15) + 2.607e-4*pow(temp + 273.15, 2) -
937             //       1.3607e-7*pow(temp + 273.15, 3) + 2.7092e-11*pow(temp + 273.15, 4);
938         }
939         else if (nano_mater == "ZnO")
940         {
941             k_nano = 1.16; //to be reviewed
942         }
943
944         return k_nano;
945     }
946
947     double NanoSpecificHeat(string nano_mater, double temp)
948     {
949         //use: calculate nanoparticle specific heat, kJ/kg-K
950         //
951         //source: Pertti Auerkari, 1996, "Mechanical and physical properties of engineering
```

```
952       // alumina ceramics", Espoo 1996, Technical Research Center of Finland,
953       // VTT Tiedotteita - Meddelanden - Research Notes 1792. 26 p.pag.8
954       //
955       // Touloukian, 1966, Thermophysical Properties of High Temperature Solid Materials,
956       // Vol 4., Pt 1, Sect 1, pp. 8 - 47
957       // http://www-ferp.ucsd.edu/LIB/PROPS/PANOS/al2o3.html
958       //
959       //author: Andrea Bigi
960       //date: 03/2016
961       //-----------------------------------------------------------------
962
963       //local variables
964
965       //output variables
966       double cp_nano;         //nanoparticle specific heat, kJ/kg-K
967
968       //flow
969
970       if (nano_mater == "Al2O3")
971       {
972           //Pertti Auerkari
973           cp_nano = 1.0446 + 0.0001742*(temp + 273.15) - 27960 * pow((temp + 273.15), (-2
              ));
974
975           //Touloukian
976           //cp_Al2O3 = (-40.92 + 4.024*(temp + 273.15) - 0.0050048*pow(temp + 273.15, 2) +
977           //  0.0000028852*pow(temp + 273.15, 3) - 0.00000000062488*pow(temp + 273.15,
              4)) / 1000;
978       }
979       else if (nano_mater == "ZnO")
980       {
981           cp_nano = 41.086/81.38; //to be reviewed
982       }
983
984       return cp_nano;
985   }
986
987   //nanoparticle properties - FINISH
988   //================================================================================//
989
990
991   //nanolubricant properties - START
992   //================================================================================//
993
994   double NanoVolumeFraction(string nano_mater, double NMF, double temp)
995   {
996       //use: calculate nanoparticle volume fraction in oil
997       //
998       //source:
          https://sites.google.com/site/compositematerialsdesign/home/weight-and-volume-fractio
          ns
999       //
1000      //author: Andrea Bigi
1001      //date: 02/2016
1002      //-----------------------------------------------------------------
1003
1004      //local variables
1005      double rho_oil;      //oil density, kg/m3
1006      double rho_nano;     //nanoparticle density, kg/m3
1007
1008      //output variables
1009      double phi;          //nanoparticle volume fraction in oil
1010
1011      //flow
1012
1013      rho_oil = OilDensity(temp);
```

```
1014        rho_nano = NanoDensity(nano_mater);
1015
1016        phi = NMF / (NMF + (1-NMF)*rho_nano/rho_oil);
1017
1018        return phi;
1019    }
1020
1021    double NanoOilDensity(string nano_mater, double NMF, double temp)
1022    {
1023        //use: calculate nanolubricant density, kg/m3
1024        //
1025        //source: Peng, H., Ding, G., Hu, H. and Jiang, W. 2011.
1026        //Effect of nanoparticle size on nucleate pool boiling heat transfer of
1027        //refrigerant/oil mixture with nanoparticles.
1028        //International Journal of Heat and Mass Transfer 54(9–10): 1839-1850.
1029        //
1030        // B.C. Pak, Y.I. Cho, Hydrodynamic and heat transfer study of dispersed fluids
1031        // with submicron metallic oxide particles, Exp.Heat Transfer 11 (2) (1998) 151–170.
1032        //
1033        // Kedzierski, 2013, Viscosity and density of aluminum oxide nanolubricant,
1034        // International Journal of Refrigeration (36), 1333-1340
1035        //
1036        // Kedzierski M., Brignoli R., Quine K., Brown J., 2016, Viscosity, density and
            thermal conductivity
1037        // of aluminum oxide and zinc oxide nanolubricants, International Journal of
            Refrigeration (),
1038        //
1039        //author: Andrea Bigi
1040        //date: 02/2016
1041        //-----------------------------------------------------------------
1042
1043        //local variables
1044        double rho_oil;     //oil density, kg/m3
1045        double rho_nano;    //nanoparticle density, kg/m3
1046        double rho_s;       //surfactant/stabilizer density, kg/m3
1047        double phi;         //nanoparticle volume fraction in oil
1048        double x_nano;      //nanoparticle mass fraction in oil and surfactant
1049        double x_s;         //surfactant mass fraction
1050        double x_oil;       //oil mass fraction
1051        double B0, B1;      //paper correlation coefficients
1052
1053        //output variables
1054        double rho_nanooil;   //nanolubricant density, kg/m3
1055
1056        //flow
1057
1058        rho_oil = OilDensity(temp);   //prefer this correlation that is validated for a
            larger range of temperatures
1059        rho_nano = NanoDensity(nano_mater);
1060        phi = NanoVolumeFraction(nano_mater, NMF, temp);
1061
1062        //from Pak(1998)
1063        rho_nanooil = (1 - phi)*rho_oil + phi*rho_nano;
1064
1065        ////////////////////////////////////////////////////////////////
1066        ////from Kedzierski(2013) - POE RL68H
1067        //rho_oil = 1 / (0.7979e-3 + 0.7647e-6 * (temp + 273.15) / 273.15);   //from paper
            Table 1 but better use other implemented correlation
1068        //rho_s = 1/(0.8443e-3 + 0.7567e-6 * (temp + 273.15));
1069        //x_oil = 1 - NMF;
1070        //x_s = -0.2061*x_oil + 0.2048;    //from linear interpolation of paper data (check
            paper temperature range - no dependence from temperature or nanoparticle size is
            claimed)
1071        //x_nano = -0.7939*x_oil + 0.7952;    //from linear interpolation of paper data
            (check paper temperature range - no dependence from temperature or nanoparticle
            size is claimed)
```

```
1072        //rho_nanooil = 1 / (x_s / rho_s + x_nano / rho_nano + x_oil / rho_oil);
1073
1074        ////from Kedzierski(2013) - eq.3
1075        //rho_nanooil = 1 / ((7.647e-7 * (1 - x_nano) - 8.647e-9 * x_s)*(temp + 273.15) +
            7.979e-4 - 5.201e-4 * x_nano + 4.64e-5 * x_s);
1076
1077        ////////////////////////////////////////////////////////////////////
1078        //from Kedzierski(2016) POE RL32-3MAF
1079        //rho_oil = 1 / (0.7972e-3 + 0.2003e-3 * (temp + 273.15) / 273.15);   //from paper
            Table 1 but better use other implemented correlation
1080
1081        //rho_s = 1 / (0.0005840 + 0.0003240 * (temp + 273.15)/273.15);
1082        //x_oil = 1 - NMF;
1083        //if (nano_mater == "Al2O3")
1084        //{
1085        //  x_s = -0.2221*x_oil + 0.2155;    //from linear interpolation of paper data
            (check paper temperature range - no dependence from temperature or nanoparticle
            size is claimed)
1086        //  x_nano = -0.7779*x_oil + 0.7845;    //from linear interpolation of paper data
            (check paper temperature range - no dependence from temperature or nanoparticle
            size is claimed)
1087        //}
1088        //else if (nano_mater == "ZnO")
1089        //{
1090        //  x_s = -0.265*x_oil + 0.2557;    //from linear interpolation of paper data
            (check paper temperature range - no dependence from temperature or nanoparticle
            size is claimed)
1091        //  x_nano = -0.735*x_oil + 0.7443;    //from linear interpolation of paper data
            (check paper temperature range - no dependence from temperature or nanoparticle
            size is claimed)
1092        //}
1093        //
1094        //rho_nanooil = 1 / (x_s / rho_s + x_nano / rho_nano + x_oil / rho_oil);
1095
1096        ////from Table 1, coefficients interpolation for both Al2O3 and ZnO  !!!not working
1097        //B0 = 0.0004*x_oil + 0.0004;
1098        //B1 = 0.0002*x_oil + 0.00005;
1099        //rho_nanooil = 1 / (B0 + B1 * (temp + 273.15) / 273.15);
1100
1101        return rho_nanooil;
1102    }
1103
1104    double NanoOilConductivity(string nano_mater, double NMF, double temp)
1105    {
1106        //use: calculate nanolubricant thermal conductivity, W/m-K
1107        //
1108        //source: Wen, D. and Ding, Y., 2005, "Effect of particle migration on
1109        //heat transfer in suspensions of nanoparticles flowing through minichannels",
1110        //Microfluidics and Nanofluidics, vol. 1, pp. 183 - 189.
1111        //
1112        // Kedzierski M., Brignoli R., Quine K., Brown J., 2016, Viscosity, density and
            thermal conductivity
1113        // of aluminum oxide and zinc oxide nanolubricants, International Journal of
            Refrigeration (),
1114        //
1115        //author: Andrea Bigi
1116        //date: 03/2016
1117        //-----------------------------------------------------------------
1118
1119        //local variables
1120        double phi;      //nanoparticle volume fraction
1121        double k_oil;    //oil thermal conductivity, W/m-K
1122        double k_nano;   //nanoparticle thermal conductivity, W/m-K
1123        double psi;      //sphericity factor, -
1124
1125        //output variables
```

```
1126        double k_nanooil;   //nanolubricant thermal conductivity, W/m-K
1127
1128        //flow
1129
1130        phi = NanoVolumeFraction(nano_mater, NMF, temp);
1131        k_oil = OilConductivity(temp);
1132        k_nano = NanoConductivty(nano_mater, temp);
1133
1134        //Kedzierski(2016) - Maxwell-Garnett modified model (Hamilton and Crosser, 1962)
1135        if (nano_mater == "Al2O3")
1136        {
1137            psi = 1;
1138        }
1139        else if (nano_mater == "ZnO")
1140        {
1141            psi = 0.55;
1142        }
1143
1144        k_nanooil = k_oil*(k_nano + (3 / psi - 1) * k_oil - (3 / psi - 1) * phi*(k_oil -
            k_nano)) /
1145                            (k_nano + (3 / psi - 1) * k_oil + phi*(k_oil - k_nano));
1146
1147        return k_nanooil;
1148    }
1149
1150    double NanoOilKinViscosity(string nano_mater, double D_nano, double NMF, double temp)
1151    {
1152        //use: calculate nanolubricant kinematic viscosity, mm2/s (cSt)
1153        //
1154        //source: Kedzierski, 2013, Viscosity and density of aluminum oxide nanolubricant,
1155        // International Journal of Refrigeration (36), 1333-1340
1156        //
1157        // Kedzierski M., Brignoli R., Quine K., Brown J., 2016, Viscosity, density and
            thermal conductivity
1158        // of aluminum oxide and zinc oxide nanolubricants, International Journal of
            Refrigeration (),
1159        //
1160        //author: Andrea Bigi
1161        //date: 05/2016
1162        //-----------------------------------------------------------------
1163
1164        //local variables
1165        double nu_oil;      //oil kinematic viscosity, mm2/s
1166        double nu_nano;     //nanoparticle "pseudo" kinematic viscosity, mm2/s
1167        double nu_s;        //surfactant/stabilizer "pseudo" kinematic viscosity, mm2/s
1168        double x_nano;      //nanoparticle mass fraction in oil and surfactant
1169        double x_s;         //surfactant mass fraction
1170        double x_oil;       //oil mass fraction
1171        double A0, A1, A2;  //paper correlation coefficients
1172
1173        //output variables
1174        double nu_nanooil;   //nanolubricant kinematic viscosity, mm2/s
1175
1176        //flow
1177
1178        nu_oil = OilKinViscosity(temp);   //prefer this correlation that is validated for a
            larger range of temperatures
1179
1180        ////////////////////////////////////////////////////////////////////
1181        ////from Kedzierski(2013) - POE RL68H
1182        //nu_oil = exp(-52.1912 + 58.8418 / ((temp + 273.15) / 273.15) + 36.8165 *
            log((temp + 273.15) / 273.15));   //from paper Table 2
1183        //nu_nano = exp((1.426 - 0.0071 * D_nano) * (4.7356 + 1.4706 / (pow(((temp +
            273.15) / 273.15), 4.05) - 1.11)));
1184        //nu_s = exp(0.149 * D_nano - 87.2079 + 7.1353 / (pow(((temp + 273.15) / 273.15),
            -66.12) + 0.074));
```

```
1185        //x_oil = 1 - NMF;
1186        //x_s = -0.2061*x_oil + 0.2048;    //from linear interpolation of paper data (check
            paper temperature range - dependence from temperature or nanoparticle size is
            claimed)
1187        //x_nano = -0.7939*x_oil + 0.7952;    //from linear interpolation of paper data
            (check paper temperature range - dependence from temperature or nanoparticle size
            is claimed)
1188        //nu_nanooil = exp(pow(x_oil, 1.25) * log(nu_oil) + pow(x_nano, 1.25) *
            log(nu_nano) + pow(x_s, 1.25) * log(nu_s));   //seems like this correlation does
            not work
1189
1190        ////from paper Table 2, more viscous oil POE RL68H
1191        //nu_nanooil = exp(-60.8428 + 67.7102 / ((temp + 273.15) / 273.15) + 44.1411 *
            log((temp + 273.15) / 273.15));   //table interpolation works but need to correct
            fitted constants at different qualities
1192
1193        ////////////////////////////////////////////////////////////////////
1194        //from Kedzierski(2016) - POE RL32-3MAF
1195        //nu_oil = exp(-45.0487 + 50.5360 / ((temp + 273.15) / 273.15) + 31.9522 *
            log((temp + 273.15) / 273.15));   //from paper Table 2
1196
1197        if (nano_mater == "Al2O3")
1198        {
1199            D_nano = 127;       //according to Kedzierski, apparent diameter should be used
1200            nu_nano = exp((1.426 - 0.0071 * D_nano) * (4.7356 + 1.4706 / (pow(((temp +
                273.15) / 273.15), 4.05) - 1.11)));
1201            nu_s = exp(0.149 * D_nano + 10.431 - 0.396 / (0.0512 - pow(((temp + 273.15) /
                273.15), -66.12)));
1202            x_oil = 1 - NMF;
1203            x_s = -0.2221*x_oil + 0.2155;    //from linear interpolation of paper data
                (check paper temperature range - no dependence from temperature or nanoparticle
                size is claimed)
1204            x_nano = -0.7779*x_oil + 0.7845;    //from linear interpolation of paper data
                (check paper temperature range - no dependence from temperature or nanoparticle
                size is claimed)
1205        }
1206        else if (nano_mater == "ZnO")
1207        {
1208            D_nano = 135;       //according to Kedzierski, apparent diameter should be used
1209            nu_nano = exp((1.426 - 0.0071 * D_nano) * (4.7356 + 1.4706 / (pow(((temp +
                273.15) / 273.15), 4.05) - 1.11)));
1210            nu_s = exp(0.149 * D_nano + 10.431 - 0.396 / (0.0512 - pow(((temp + 273.15) /
                273.15), -66.12)));
1211            x_oil = 1 - NMF;
1212            x_s = -0.265*x_oil + 0.2557;    //from linear interpolation of paper data
                (check paper temperature range - no dependence from temperature or nanoparticle
                size is claimed)
1213            x_nano = -0.735*x_oil + 0.7443;    //from linear interpolation of paper data
                (check paper temperature range - no dependence from temperature or nanoparticle
                size is claimed)
1214        }
1215
1216        nu_nanooil = exp(pow(x_oil, 1.25) * log(nu_oil) + pow(x_nano, 1.25) * log(nu_nano) +
             pow(x_s, 1.25) * log(nu_s));   //seems like this correlation does not work
1217
1218        //from Table 1, coefficients interpolation for both Al2O3 and ZnO
1219        A0 = -18.401*x_oil - 29.739;
1220        A1 = 16.444*x_oil + 37.087;
1221        A2 = 16.855*x_oil + 17.823;
1222        nu_nanooil = exp(A0 + A1 / ((temp + 273.15) / 273.15) + A2 * log((temp + 273.15) /
            273.15));
1223
1224        return nu_nanooil;
1225    }
1226
1227    double NanoOilDynViscosity(string nano_mater, double D_nano, double NMF, double temp)
```

```
{
    //use: calculate nanolubricant dynamic viscosity, N-s/m2, kg/m-s, Pa-s
    //
    //source: Batchelor, G. 1977. The effect of Brownian motion on the bulk stress
    //in a suspension of spherical particles.
    //Journal of Fluid Mechanics 83(01): 97-117.
    //
    //author: Andrea Bigi
    //date: 03/2016
    //-----------------------------------------------------------------

    //local variables
    double k1, k2;      //empirical constants
    double nu_oil;      //oil kinematic viscosity, mm2/s
    double nu_nanooil;  //nanolubricant kinematic viscosity, mm2/s
    double rho_oil;     //oil density, kg/m3
    double rho_nanooil; //nanolubricant density, kg/m3
    double mu_oil;      //oil dynamic viscosity, N-s/m2, kg/m-s, Pa-s
    double phi;         //nanoparticle volume fraction in oil

    //output variables
    double mu_nanooil;   //nanolubricant dynamic viscosity, N-s/m2, kg/m-s, Pa-s

    //flow

    k1 = 2.5;
    k2 = 6.2;

    nu_oil = OilKinViscosity(temp);
    rho_oil = NanoOilDensity(nano_mater, NMF, temp);
    mu_oil = nu_oil*rho_oil / 1000000;
    phi = NanoVolumeFraction(nano_mater, NMF, temp);

    mu_nanooil = mu_oil*(1 + k1*phi + k2*pow(phi, 2));

    //from Kedzierski(2013, 2016)
    nu_nanooil = NanoOilKinViscosity(nano_mater, D_nano, NMF, temp);
    rho_nanooil = NanoOilDensity(nano_mater, NMF, temp);
    mu_nanooil = nu_nanooil * rho_nanooil / 1000000;

    return mu_nanooil;
}

double NanoOilSurfTension(double temp)
{
    //use: calculate nanolubricant surface tension, N/m
    //
    //source: Das, S. K., Putra, N. and Roetzel, W. 2003.
    //Pool boiling characteristics of nano-fluids.
    //International Journal of Heat and Mass Transfer 46(5): 851-862.
    //
    //author: Andrea Bigi
    //date: 02/2016
    //-----------------------------------------------------------------

    //local variables

    //output variables
    double sigma_nanooil;    //nanolubricant surface tension, N/m

    //flow

    sigma_nanooil = OilSurfTension(temp);

    return sigma_nanooil;
}
```

```
1294
1295  double NanoOilSpecificHeat(string nano_mater, double temp, double phi)
1296  {
1297      //use: calculate nanolubricant specific heat, kJ/kg-K
1298      //
1299      //source: Murshed, SM Sohel. Determination of effective specific heat of nanofluids.
1300      // Journal of Experimental Nanoscience 6, no. 5 (2011): 539-546.
1301      //
1302      //author: Andrea Bigi
1303      //date: 03/2016
1304      //-----------------------------------------------------------------
1305
1306      //local variables
1307      double rho_oil;     //oil density, kg/m3
1308      double rho_nano;    //nanoparticle density, kg/m3
1309      double cp_oil;        //lubricant specific heat, kJ/kg-K
1310      double cp_nano;       //nanoparticle specific heat, kJ/kg-K
1311
1312      //output variables
1313      double cp_nanooil;    //nanolubricant specific heat, kJ/kg-K
1314
1315      //flow
1316
1317      rho_oil = OilDensity(temp);
1318      rho_nano = NanoDensity(nano_mater);
1319      cp_oil = OilSpecificHeat(temp);
1320      cp_nano = NanoSpecificHeat(nano_mater, temp);
1321
1322      cp_nanooil = (phi*rho_nano*cp_nano + (1 - phi)*rho_oil*cp_oil) / (phi*rho_nano + (1
         - phi)*rho_oil);
1323
1324      return cp_nanooil;
1325  }
1326
1327  //nanolubricant properties - FINISH
1328  //=============================================================================//
1329
1330
1331  //refrigerant-nanolubricant mixture properties - START
1332  //=============================================================================//
1333
1334  double NanoOilMassFraction(double OMF, double NMF, double m_dot_fluid, double temp)
1335  {
1336      //use: calculate nanolubricant mass fraction in refrigerant AND nanolubricant mixture
1337      //
1338      //source:
1339      //
1340      //author: Andrea Bigi
1341      //date: 03/2016
1342      //-----------------------------------------------------------------
1343
1344      //local variables
1345      double m_dot_oil;           //oil mass flow rate, kg/s
1346      double v_dot_oil;           //oil volume flow rate, m3/s
1347      double v_dot_nanooil;       //nanolubricant volume flow rate, m3/s
1348      double m_dot_nanooil;       //nanolubricant mass flow rate, kg/s
1349      double rho_oil;             //oil density, kg/m3
1350      double rho_nano;            //nanoparticle density, kg/m3
1351      double phi;                 //nanoparticle volume fraction in oil
1352
1353      //output variables
1354      double OMFnano;      //local nanolubricant mass fraction
1355
1356      //flow
1357
1358      //rho_oil = OilDensity(temp);
```

```
1359        //rho_nano = rho_Al2O3;
1360        //phi = NanoVolumeFraction(NMF, temp);
1361
1362        m_dot_oil = (m_dot_fluid*OMF) / (1 - OMF);  //oil mass fraction already accounts
            for presence of nanoparticles inside it
1363                                                    //(it's just that they are defined
                                                        differently because OMFnano also
                                                        includes the mass of nanoparticles),
1364                                                    //so in the end OMFnano = OMF
1365        //v_dot_oil = m_dot_oil / rho_oil;
1366        //v_dot_nanooil = (v_dot_oil*phi) / (1-phi);
1367        //m_dot_nanooil = v_dot_nanooil*rho_nano;
1368
1369        OMFnano = m_dot_oil / m_dot_fluid;
1370        OMFnano = OMF;
1371
1372        return OMFnano;
1373    }
1374
1375    double LocalNanoOilMassFraction(double OMFnano, double x_fluid)
1376    {
1377        //use: calculate local nanolubricant mass fraction, given absolute nanolubricant
            mass fraction
1378        //
1379        //source:
1380        //
1381        //author: Andrea Bigi
1382        //date: 03/2016
1383        //---------------------------------------------------------------------
1384
1385        //local variables
1386        double x_nanomix;        //nanolubricant-refrigerant mixture quality
1387
1388        //output variables
1389        double w_local_nano;      //local nanolubricant mass fraction
1390
1391        //flow
1392
1393        x_nanomix = x_fluid * (1 - OMFnano);        //basically the same for the case
            without nanoparticles
1394
1395        w_local_nano = OMFnano / (1 - x_nanomix);
1396
1397
1398        return w_local_nano;
1399    }
1400
1401
1402    double NanoOilMixtureDensity(string nano_mater, double NMF, double temp, double
    w_local_nano, double rho_f_ref)
1403    {
1404        //use: calculate refrigerant-nanolubricant mixture density, kg/m3
1405        //
1406        //source: Jensen, M. and Jackman, D. 1984.
1407        // Prediction of nucleate pool boiling heat transfer coefficients of
1408        // refrigerant-oil mixtures. Journal of heat transfer 106(1): 184-190
1409        //
1410        //author: Andrea Bigi
1411        //date: 03/2016
1412        //---------------------------------------------------------------------
1413
1414        //local variables
1415        double rho_nanooil;   //nanolubricant density, kg/m3
1416
1417        //output variables
1418        double rho_nanomix;   //refrigerant-nanolubricant mixture density, kg/m3
```

```
1419
1420        //flow
1421
1422        rho_nanooil = NanoOilDensity(nano_mater, NMF, temp);
1423
1424        rho_nanomix = 1 / (w_local_nano / rho_nanooil + (1 - w_local_nano) / rho_f_ref);
1425
1426        return rho_nanomix;
1427    }
1428
1429    double NanoOilMixtureConductivity(string nano_mater, double NMF, double temp, double
        w_local_nano, double k_f_ref)
1430    {
1431        //use: calculate refrigerant-nanolubricant mixture thermal conductivity, W/m-K
1432        //
1433        //source: Filippov, L. and Novoselova, N. 1955.
1434        // The thermal conductivity of solutions of normal liquid
1435        // Chem Abstr. 49: 37-40
1436        //
1437        //author: Andrea Bigi
1438        //date: 03/2016
1439        //-----------------------------------------------------------------
1440
1441        //local variables
1442        double k_nanooil;   //nanolubricant density, kg/m3
1443
1444        //output variables
1445        double k_nanomix;   //refrigerant-nanolubricant mixture density, kg/m3
1446
1447        //flow
1448
1449        k_nanooil = NanoOilConductivity(nano_mater, NMF, temp);
1450
1451        k_nanomix = k_f_ref*(1 - w_local_nano) + k_nanooil*w_local_nano -
1452                    0.72*(k_nanooil - k_f_ref)*(1 - w_local_nano)*w_local_nano;
1453
1454        return k_nanomix;
1455    }
1456
1457    double NanoOilMixtureKinViscosity(string nano_mater, double D_nano, double NMF, double
        temp, double w_local_nano, double mu_f_ref, double rho_f_ref)
1458    {
1459        //use: calculate refrigerant-nanolubricant mixture kinematic viscosity, mm2/s (cSt)
1460        //
1461        //source:
1462        //
1463        //author: Andrea Bigi
1464        //date: 03/2016
1465        //-----------------------------------------------------------------
1466
1467        //local variables
1468        double mu_nanomix;   //refrigerant-nanolubricant mixture dynamic viscosity, N-s/m2,
            kg/m-s, Pa-s
1469        double rho_nanomix;  //refrigerant-nanolubricant mixture density, kg/m3
1470
1471        //output variables
1472        double nu_nanomix;   //refrigerant-nanolubricant mixture kinematic viscosity, mm2/s
1473
1474        //flow
1475
1476        mu_nanomix = NanoOilMixtureDynViscosity(nano_mater, D_nano, NMF, temp, w_local_nano,
             mu_f_ref);
1477        rho_nanomix = NanoOilMixtureDensity(nano_mater, NMF, temp, w_local_nano, rho_f_ref);
1478
1479        nu_nanomix = mu_nanomix / rho_nanomix * 1000000;
1480
```

```
1481        return nu_nanomix;
1482    }
1483
1484
1485    double NanoOilMixtureDynViscosity(string nano_mater, double D_nano, double NMF, double
        temp, double w_local_nano, double mu_f_ref)
1486    {
1487        //use: calculate refrigerant-nanolubricant mixture dynamic viscosity, N-s/m2,
            kg/m-s, Pa-s
1488        //
1489        //source: Kedzierski, M. A. and Kaul, M. P. 1993.
1490        // Horizontal nucleate flow boiling heat transfer coefficient measurements
1491        // and visual observations for R12, R134a, and R134a/ester lubricant mixtures,
1492        // National Institute of Standards and Technology, Building and Fire Research
            Laboratory
1493        //
1494        //author: Andrea Bigi
1495        //date: 03/2016
1496        //---------------------------------------------------------------------
1497
1498        //local variables
1499        double mu_nanooil;   //refrigerant-nanolubricant mixture kinematic viscosity, mm2/s
1500
1501        //output variables
1502        double mu_nanomix;    //refrigerant-nanolubricant mixture dynamic viscosity,
            N-s/m2, kg/m-s, Pa-s
1503
1504        //flow
1505
1506        mu_nanooil = NanoOilDynViscosity(nano_mater, D_nano, NMF, temp);
1507
1508        mu_nanomix = exp(w_local_nano*log(mu_nanooil) + (1 - w_local_nano)*log(mu_f_ref));
1509
1510        return mu_nanomix;
1511    }
1512
1513    double NanoOilMixtureSpecificHeat(string nano_mater, double NMF, double temp, double
        w_local_nano, double cp_f_ref)
1514    {
1515        //use: calculate refrigerant-nanolubricant mixture specific heat, kJ/kg-K
1516        //
1517        //source: Jensen, M. and Jackman, D. 1984.
1518        // Prediction of nucleate pool boiling heat transfer coefficients of
1519        // refrigerant-oil mixtures. Journal of heat transfer 106(1): 184-190
1520        //
1521        //author: Andrea Bigi
1522        //date: 03/2016
1523        //---------------------------------------------------------------------
1524
1525        //local variables
1526        double rho_oil;      //oil density, kg/m3
1527        double cp_oil;       //oil specific heat, kJ/kg-K
1528        double cp_nano;      //nanoparticle specific heat, kJ/kg-K
1529        double rho_nanooil;  //nanolubricant density, kg/m3
1530        double phi;          //nanoparticles volumetric fraction
1531        double cp_nanooil;   //nanolubricant specific heat, kJ/kg-K
1532
1533        //output variables
1534        double cp_nanomix;   //refrigerant-nanolubricant mixture specific heat, kJ/kg-K
1535
1536        //flow
1537
1538        rho_oil = OilDensity(temp);
1539        cp_oil = OilSpecificHeat(temp);
1540        cp_nano = NanoSpecificHeat(nano_mater, temp);
1541        rho_nanooil = NanoOilDensity(nano_mater, NMF, temp);
```

```
1542        phi = NanoVolumeFraction(nano_mater, NMF, temp);
1543        cp_nanooil = NanoOilSpecificHeat(nano_mater, temp, phi);
1544
1545        cp_nanomix = w_local_nano*cp_nanooil + (1 - w_local_nano)*cp_f_ref;
1546
1547        return cp_nanomix;
1548    }
1549
1550    double NanoOilMixtureSurfTension(double temp, double w_local_nano, double sigma_f_ref)
1551    {
1552        //use: calculate refrigerant-nanolubricant mixture surface tension, N/m
1553        //
1554        //source: Jensen, M. and Jackman, D. 1984.
1555        // Prediction of nucleate pool boiling heat transfer coefficients of
1556        // refrigerant-oil mixtures. Journal of heat transfer 106(1): 184-190
1557        //
1558        //author: Andrea Bigi
1559        //date: 03/2016
1560        //-----------------------------------------------------------------
1561
1562        //local variables
1563        double sigma_nanooil;
1564
1565        //output variables
1566        double sigma_nanomix;   //refrigerant-oil mixture surface tension, N/m
1567
1568        //flow
1569
1570        sigma_nanooil = NanoOilSurfTension(temp);
1571
1572        sigma_nanomix = sigma_f_ref + (sigma_nanooil - sigma_f_ref)*sqrt(w_local_nano);
1573
1574        return sigma_nanomix;
1575    }
1576
1577    //refrigerant-nanolubricant mixture properties - FINISH
1578    //=============================================================================//
```

```
1  //Tube Calorimeter - v0.1//
2  Input File
3
4  ////Sawant(2007)
5  0.00658452,     //A0, Sawant(2007)
6  0.000434741,    //A1, Sawant(2007)
7  -0.00000129204, //A2, Sawant(2007)
8  -2300.2,    //a0, Sawant(2007)
9  15.146,     //b0, Sawant(2007)
10
11 //from interpolation of experimental data of pure refrigerant R410A  (2015 microfin
   tube series: 1-28)
12 0.00774789631085859,    //A0, correlation coeff for ref-oil mixture Tbubble (Sawant)
   (2015 microfin tube series: 1-28)
13 -0.000605742908554461,  //A1, correlation coeff for ref-oil mixture Tbubble (Sawant)
   (2015 microfin tube series: 1-28)
14 -0.00000420111057045546,//A2, correlation coeff for ref-oil mixture Tbubble (Sawant)
   (2015 microfin tube series: 1-28)
15 -1650.94903353027,  //a0, correlation coeff for ref-oil mixture Tbubble (Sawant) (2015
   microfin tube series: 1-28)
16 12.7910297482209,   //b0, correlation coeff for ref-oil mixture Tbubble (Sawant) (2015
   microfin tube series: 1-28)
17
18 //from interpolation of experimental data of pure refrigerant R410A  (2015 microfin
   tube series: 29-63)
19 0.00448285188977577,    //A0, correlation coeff for ref-oil mixture Tbubble (Sawant)
   (2015 microfin tube series: 29-63)
20 -0.000128240216365056,  //A1, correlation coeff for ref-oil mixture Tbubble (Sawant)
   (2015 microfin tube series: 29-63)
21 -0.00000420111057045546,    //A2, correlation coeff for ref-oil mixture Tbubble
   (Sawant) (2015 microfin tube series: 29-63)
22 -5719.02974975048,  //a0, correlation coeff for ref-oil mixture Tbubble (Sawant) (2015
   microfin tube series: 29-63)
23 27.4591324204553,   //b0, correlation coeff for ref-oil mixture Tbubble (Sawant) (2015
   microfin tube series: 29-63)
24
25 "mode"
26 test unit;  //(tube calorimeter, test unit, ...)
27
28 "fluid"
29 IIR,    //enthalpy reference (IIR, ASHRAE, NBP, DEF)
30 R410A,  //base fluid
31 0.00448285188977577,    //A0, correlation coeff for ref-oil mixture Tbubble (Sawant)
   (2015 microfin tube series: 29-63)
32 -0.000128240216365056,  //A1, correlation coeff for ref-oil mixture Tbubble (Sawant)
   (2015 microfin tube series: 29-63)
33 -0.00000420111057045546,    //A2, correlation coeff for ref-oil mixture Tbubble
   (Sawant) (2015 microfin tube series: 29-63)
34 -5719.02974975048,  //a0, correlation coeff for ref-oil mixture Tbubble (Sawant) (2015
   microfin tube series: 29-63)
35 27.4591324204553,   //b0, correlation coeff for ref-oil mixture Tbubble (Sawant) (2015
   microfin tube series: 29-63)
36 EMKARATE POE RL32-3MAF, //oil type
37 Al2O3,  //nanoparticle material (Al2O3, ZnO,...)
38 spherical,  //nanoparticle shape
39 60;     //nanoparticle equivalent diameter, nm   (Al2O3: 60; ZnO: 40)
40
41 "geometry"
42 microfin, //tube type (smooth, microfin,...)
43 copper, //tube material (copper, steel,...)
44 horizontal,  //tube orientation (horizontal, vertical, inclined)
45 0.001, //tube roughness, m
46 1, //number of segments, -
47 10, //number of radial segments for laminar sublayer, -
48 1.83, //heat transfer tube length, m
49 2.4, //pressure drop tube length, m
```

```
0.00545, //tube hydraulic diameter, m
0.00952,  //tube outside diameter, m  (D_o)
0.00892,  //maximum inside diameter of micro-fin tube (at fin root), m  (D_o - 2*t_w)
0.0003,  //tube wall thickness at fin root, m  (t_w)
0.0002,  //fin height, m
0.00047,  //fin pitch, m
18,  //helix angle, deg  (beta)
50,  //fin angle, deg  (alpha)
60,  //number of fins, -
0.000707;  //perimeter of one fin and channel, m

"preheater inlet" //unless in.csv
949.78, //inlet pressure, kPa
-8.25, //inlet temperature, C
2534.65; //heat capacity, W

"inlet" //unless in.csv
0.0, //oil mass fraction (OMF)
0.0, //nanoparticle and surfactant mass fraction in oil (NMF)
0.0154, //mass flow rate, kg/s
935.00, //inlet pressure, kPa
352.41, //inlet enthalpy, kJ/kg
977.01, //heat capacity, W
5.64; //wall temperature, C

```

Flow Rate (	Q_pre (W)	P_in_pre (k	T_in_pre (	Q_test_sec	P_ave_test	H_in_test_	T_wall_tes	OMF	NMF	HTC (Twall center - Tref_sawant+t
0.010955	470.7431	907.893	1.948072	955.7524	914.0443	245.9402	6.00289	0.01	0	6.193457
0.010818	1375.191	907.5645	-5.88001	944.6802	912.2682	378.2967	5.732792	0.01	0	7.674864
0.025664	708.3709	917.5202	1.467133	964.4337	918.3754	229.836	6.341889	0.01	0	5.158957
0.02489	3309.674	946.5883	-8.75241	947.3195	926.9509	379.9047	5.673186	0.01	0	8.005268
0.010955	470.7431	907.893	1.948072	955.7524	914.0443	245.9402	6.00289	0.01	0.1	6.193457
0.010818	1375.191	907.5645	-5.88001	944.6802	912.2682	378.2967	5.732792	0.01	0.1	7.674864
0.025664	708.3709	917.5202	1.467133	964.4337	918.3754	229.836	6.341889	0.01	0.1	5.158957
0.02489	3309.674	946.5883	-8.75241	947.3195	926.9509	379.9047	5.673186	0.01	0.1	8.005268
0.010857	377.3763	900.8966	1.968719	967.2496	908.6794	237.7583	5.519875	0.03	0.2	8.020086
0.010898	1378.195	904.9805	-7.22188	971.5724	910.5479	375.6547	5.621894	0.03	0.2	8.794271
0.025306	709.0693	918.231	1.135838	968.1463	920.0545	229.7485	5.907358	0.03	0.2	6.549434
0.025429	3236.864	940.6439	-9.78648	974.8654	920.5543	372.7007	5.425539	0.03	0.2	9.490366

home)