Efficient and Robust Classification for Positive Definite Matrices with Wasserstein Metric

by

Jian Cui

A thesis submitted to the Graduate Faculty of
Auburn University
in partial fulfillment of the
requirements for the Degree of
Master of Science

Auburn, Alabama
May 1, 2021

Keywords: Riemannian Manifold, Rie-mannian Metric, Bures-Wasserstein Distance, Bures-Wasserstein Mean

Approved by

Jingyi Zheng, Chair, Assistant Professor of Mathematics and Statistics
Huajun Huang, Associate Professor of Mathematics and Statistics
Peng Zeng, Associate Professor of Mathematics and Statistics

Abstract

Riemannian geometry methods are widely used to classify SPD (Symmetric Positives-Definite) matrices, such as covariances matrices of brain-computer interfaces. Common Riemannian geometry classification methods are based on Riemannian distance to compute the mean of matrices. The purpose of this paper is to propose different algorithms based on Bures-Wasserstein distance for computing the mean of SPD matrices. Combining two purposed BW algorithms, Inductive mean and Cheap mean, with the most common simple projection algorithm based on Riemannian distance, there are 6 kinds of mean algorithms tested. The results obtained in this paper include that Bures-Wasserstein simple projection mean algorithm has a better efficient and robust performance than the others.

Acknowledgments

I would like to thank my supervisor Dr. Jingyi Zheng, who guided me to throughout the whole project and greatly improved my understanding of academic writing and taught me a lot of specific research skills.

I also would like to thank Dr. Huajun Huang, who provided me with a lot of theoretical supports and ideas for experiments.

In addition, I would like to thank my parents for their mental and financial support.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

BW	Bures-Wasserstein
SPD	Symmetric Positive Definite

Chapter 1

Introduction

Btrain-computer interface (BCI) is a computer-based system that can obtain and analyzes brain signals by connecting brain and external devices. There are kinds of devices used to collect brain signals, Invasive and Non-invasive devices. The Invasive devices will implant electrodes under scalp, which can obtain brain signals clearly. For the non-invasive devices, signals have more noise, but people don't need to do surgery on their head. This advantage makes more people can take part in experiment. With the cost of having more noise, collecting data becomes easier. Among different noninvasive devices, EEG-based BCIs, which is our focus in this proposal, have been widely adopted in various applications, especially in helping users with severe motor impairments (e.g. brainstem stroke, spinal cord injury) to communicate and control external devices [1, 2, 3, 4, 5, 6, 7]. For example, Niels et al [8] used EEG-based BCI to help severely paralyzed people to control a computer cursor. Based on motor imagination, Bin et al [9] controlled a virtual helicopter flying in 3-dimensional space by EEG-based BCI.

All researchers are trying to interpret brain signals to commands or situations accurately. But applications of EEG-based BCIs still cannot be applied in practice widely. Because there are too much noise and information in signals. The main purposes of researching in this area is to find the specific signals among brain signals and interpret them. Also, since the brain signals are non-stationary and non-linear signals with neural oscillations. For getting enough EEG data, users have to repeat the specific actions such as moving, thinking. In the process of collecting, it is impossible to control users mind. If they are distracted by other minds, data would be contaminated seriously and discarded. Therefore, improving the accuracy of interpretation will be a long-term research topic in this area.

In the past, the research work of process can be summarized into two pipelines. The most common pipeline of dealing with EEG-based BCI data has steps as follows.

1. source extraction via spatial filtering (e.g. using common spatial pattern method [10, 11, 12, 13, 14]),

2. feature extraction from the source signals (e.g. extracting power spectral density [15, 16, 17, 18, 19, 20]),

3. classification via vector-based models (e.g. using linear discriminant analysis, SVM, etc [21, 22, 23, 24, 25, 26, 27]).

In this pipeline, researchers could use different methods in every step. The results of different methods also are various for different problems. Besides, the other pipeline shown [28, 29, 30, 31, 32] merges source extraction and feature extraction into one step, which classifies the covariance matrices directly. The upside of this pipeline save more the spatial and temporal structure of the BCI system than the others by direct classification on covariance matrices. In the Euclidean space, analyzing matrices will lead to biases, but these biases do not exist in matrix space. For building classification model on the manifold, it is necessary to build a metric system based on Riemannian Geometry. In a metric system, the distances and angles can be measured. With a metric system, the classification model on manifold can be calculate. Now, the most common distance function are used to measure distance between matrices is Riemannian Distance. There are also many classifiers that developed based on metric of Riemannian distance.[33, 28, 34, 35, 36, 37]. Metric based on Riemannian distance has good performance on accuracy of classification and more simple steps. But, the process of computing riemannian distance also has some potential issues, such as time consuming and unstable, which means the classification processes based on riemannian distance have these disadvantages too.

Chapter 2

Background Knowledge

$\mathbb{P}_n$ can be viewed as a Riemannian manifold. For simplicity of computation, Euclidean metrics was found that it can be used on the classification of Riemannian Geometry. When matrices on $\mathbb{P}_n$ are projected to tangent space of the manifold $\mathbb{P}_n$, euclidean metrics can be used to find mean of matrices on the tangent space. Then, This mean can be projected back to manifold, which can be the estimation of mean of matrices on the manifold. The projection, also named retractions, between a manifold and its tangent space contains two parts, Logarithmic map and Exponential map. Logarithmic map function used to project the points on the manifold to the related tangent, Exponential map function used to project points from the tangent space to the manifold, while Exponential map function used to project points from the tangent space back to the manifold as illustrated in Figure 2.1. Therefore, the definition of the exponential and logarithmic maps are important and they depend on our geometric view of the manifold $\mathbb{P}_n$. In this thesis, we can consider two metrics based on different distance methods respectively:

- The metric based on Riemannian distance
- The metric based on Bures-Wasserstein distance

2.1 The metric based on Riemannian distance

Based on the metric of Riemanian distance, The key function for measuring distance between A and B on the mannifold is Riemannian distance. The function of Riemannian distance is as follows,

$$\delta_R(A, B) = \big(\sum_{i=1}^{n} \log^2 \lambda_i(A^{-1}B) \big)^{1/2}, \tag{2.1}$$

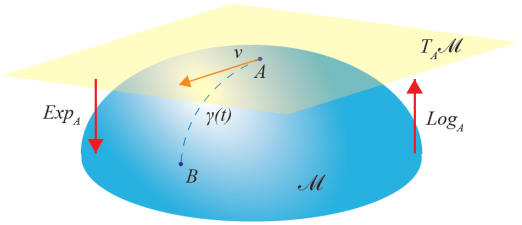


Figure 2.1: Manifold $\mathbb{P}_n$ and its tangent space at point A

where $\lambda_i(A^{-1}B), i = 1, \ldots, n$ is the eigenvalues of $A^{-1}B$. There is a unique geodesic connecting A and B on the manifold $\mathbb{P}_n$ (i.e. A and B) shown as follows.

$$\gamma(t) = A^{1/2}(A^{-1/2}BA^{-1/2})^t A^{1/2}, \quad t \in [0, 1]. \tag{2.2}$$

When $t = 1/2$, The $\gamma(t)$ is the midpoint of two matrices A and B, and denoted as $A\#B$.

$$\gamma(1/2) = A^{1/2}(A^{-1/2}BA^{-1/2})^{1/2}A^{1/2}. \tag{2.3}$$

The logarithmic and exponential functions based on Riemannian distance in projection process based are [36]:

$$X = \log_A(B) = A^{1/2}log(A^{-1/2}BA^{-1/2})A^{1/2} \tag{2.4}$$

$$B = \exp_A(B) = A^{1/2}exp(A^{-1/2}XA^{-1/2})A^{1/2} \tag{2.5}$$

Riemannian distance retains many good properties, which makes it widely used in BCIs and other applications. However, computing Riemannian distance involves matrix inverse and eigenvalue decomposition as shown in (2.1), which are time consuming and computationally unstable especially for large matrices. Therefore, we consider the manifold in a different geometric view.

2.2 The metric based on Bures-Wasserstein distance

There was a different metric system based on Bures-Wasserstein distance proposed by Dr.Huajun Huang. The Bures-Wasserstein distance function is

$$d_{BW}(A,B) = \left[\mathrm{tr}\,(A+B) - 2\mathrm{tr}\,(AB)^{1/2}\right]^{1/2}$$

where $A, B \in \mathbb{P}_n$. The geodesic of A to B on the manifold $\mathbb{P}_n$ is

$$\gamma(t) = (1-t)^2 A + t^2 B + t(1-t)[(AB)^{1/2} + (BA)^{1/2}], t \in [0,1] \tag{2.6}$$

The logarithmic and exponential functions based on Bures-Wasserstein distance are:

$$X = \log_A(B) = (AB)^{1/2} + (BA)^{1/2} - 2A. \tag{2.7}$$

$$B = \exp_A(X) = U[W \circ (U^*XU + 2\Lambda)]\Lambda[W \circ (U^*XU + 2\Lambda)]U^*.$$

Where $\circ$ denote the Hadamard product, $A = U\Lambda U^*$,$\Lambda = \mathrm{diag}\,(\lambda_1, \cdots, \lambda_n)$ and

$$W = \left(\frac{1}{\lambda_i + \lambda_j}\right)_{n\times n}. \tag{2.8}$$

From the equations above, Bures-Wasserstein distance seems has some potential advantages compared with Riemannian distance. It could be more robust and easy to calculate. Because there is no logarithm in Bures-Wasserstein computing process.

Chapter 3

Methodology

3.1 Inductive mean algorithm with BW distance

From the idea of Fréchet mean, we estimate the 'center' of the matrices by the Fréchet edition's mean of SPD matrices with BW distance. The function can be defined as .

$$\text{Fréchet Mean: } \bar{A}(A_1, \ldots, A_m) = \underset{X \in \mathbb{P}_n}{\operatorname{argmin}} \sum_{i=1}^{m} d^2(A_i, X) \tag{3.1}$$

where d is a distance function on the manifold. The Fréchet mean is also called the barycenter or Karcher mean in literatures (e.g. [38, 30]). The barycenter of matrices provides an estimation for mean of matrices on manifold, and plays an important role in building the Gaussian distribution for matrices.

To calculate the Fréchet mean for matrices on $\mathbb{P}_n$, there are two types of methods was proposed, an inductive type and a cheap mean type. They use different iteration steps to converge matrices to the Fréchet mean.

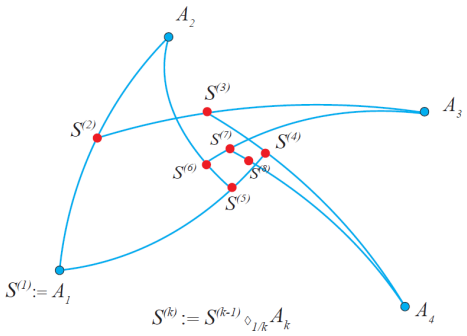


Figure 3.1: Illustration of the inductive mean algorithm.

The inductive mean method described in [39] is proved to be valid to find the Fréchet mean of points on Hadamard spaces like $(\mathbb{P}_n, \delta_R)$. Even though $(\mathbb{P}_n, d_{BW})$ is not a Hadamard space, the inductive mean method still works in almost all situations. We propose the **Inductive Mean Algorithm**, illustrated in Figure 3.1, to find the Fréchet mean of m matrices $A_1, \ldots, A_m \in \mathbb{P}_n$:

1. Define the sequence $\{A_k\}_{k\in\mathbb{N}}$ such that $A_k = A_{k+m} = A_{k+2m} = \cdots$ for all $k \in \mathbb{N}$.

2. Let $S^{(1)} := A_1$. For $k = 2, 3, \ldots,$ let

$$\begin{aligned} S^{(k)} &:= S^{(k-1)} \diamond_{\frac{1}{k}} A_k \\ &= \frac{(k-1)^2}{k^2} S^{(k-1)} + \frac{1}{k^2} A_k + \frac{k-1}{k^2} \left[(S^{(k-1)} A_k)^{1/2} + (A_k S^{(k-1)})^{1/2} \right]. \end{aligned}$$

3. The limit of $\{S^{(k)}\}_{k\in\mathbb{N}}$ will be the Fréchet mean of $\{A_1, \ldots, A_m\}$ with d_{BW}:

$$\lim_{k\to\infty} S^{(k)} = \overline{A}(A_1, \ldots, A_m). \tag{3.2}$$

The Inductive Mean Algorithm has simple computation process, since it doesn't has projection steps containing logarithms and exponents .

3.2 Simple projection mean algorithm with BW distance

The main idea of simple projection algorithm with BW distance is from simple projection algorithm with Riemannian distance [36]. In riemannian metric, this algorithm is the most common algorithm to find mean of SPD matrices on manifold. So, we developed simple projection algorithm with BW distance. To the Fréchet mean of m matrices $A_1, \ldots, A_m \in \mathbb{P}_n$, the main steps of simple projection mean algorithm with BW distance are

1. Initialise $P^{(1)} = \frac{1}{m} \sum_{i=1}^{m} (A_i)$

2. Project all A_i onto the tangent space at $P^{(k)}$. For k = 1,2,3,...

$$X_i = (P^{(k)} A_i)^{1/2} + (A_i P^{(k)})^{1/2} - 2P^{(k)}$$

3. Compute the arithmetic mean of the projection vectors,

$$S^{(k)} = \frac{1}{m}\sum_{i=1}^{m} X_i$$

4. Project S back to manifold, the exponential of S at P is

$$P^{(k+1)} = U_i[W_i \circ (U_i{}^* S^{(k)} U_i + 2\Lambda_i)]\Lambda_i[W_i \circ (U_i{}^* S^{(k)} U_i + 2\Lambda_i)]U_i{}^*.$$

5. The limit of $P^{(k)}{}_{k\in\mathbb{N}}$ will be the simple projection mean of $A_1, ...A_m$ with BW distance.

$$\lim_{k\to\infty} P^{(k)} = \overline{A}(A_1, \ldots, A_m). \tag{3.3}$$

3.3 Cheap mean algorithm with BW distance

The third algorithm we will consider and compare with is the Cheap Mean Algorithm originally developed on $(\mathbb{P}_n, \delta_R)$ in [40]. Since simple projection mean algorithms have the same key projection step, it can be considered as a complex edition of simple projection algorithm. The iteration process of our **Cheap Mean Algorithm** on $(\mathbb{P}_n, d_{BW})$ is as follow:

1. Let $A_k^{(0)} = A_k$ for $k = 1, \ldots, m$.

2. Suppose $A_1^{(\ell)}, \ldots, A_m^{(\ell)}$ are known for some $\ell \in \mathbb{N}$. For each $k \in \{1, \ldots, m\}$, we project the geodesic curves connecting $A_k^{(\ell)}$ to $A_1^{(\ell)}, \ldots, A_m^{(\ell)}$ onto the tangent space at $A_k^{(\ell)}$. Then find the arithmetic mean of the projection vectors. The exponential of this arithmetic mean at $A_k^{(\ell)}$ is denoted by $A_k^{(\ell+1)}$:

   (a) By (2.7), the projection of geodesic from $A_k^{(\ell)}$ to $A_j^{(\ell)}$ onto the tangent space at $A_k^{(\ell)}$ is

$$X_{kj}^{(\ell)} := (A_k^{(\ell)} A_j^{(\ell)})^{1/2} + (A_j^{(\ell)} A_k^{(\ell)})^{1/2} - 2A_k^{(\ell)}. \tag{3.4}$$

(b) The arithmetic mean of the projection vectors is

$$X_k^{(\ell)} := \frac{1}{m}\sum_{j=1}^{m} X_{kj}^{(\ell)} = \frac{1}{m}\sum_{j=1}^{m}\left[(A_k^{(\ell)}A_j^{(\ell)})^{1/2} + (A_j^{(\ell)}A_k^{(\ell)})^{1/2}\right] - 2A_k^{(\ell)}. \tag{3.5}$$

(c) Find the spectral decomposition

$$A_k^{(\ell)} = U_k^{(\ell)}\Lambda_k^{(\ell)}U_k^{(\ell)^*}, \quad U_k^{(\ell)} \in \mathrm{U(n)},\ \Lambda_k^{(\ell)} = \mathrm{diag}\,(\lambda_{k1}^{(\ell)}, \lambda_{k2}^{(\ell)}, \cdots, \lambda_{km}^{(\ell)}). \tag{3.6}$$

Denote the matrix

$$W_k^{(\ell)} := \left(\frac{1}{\lambda_{ki}^{(\ell)} + \lambda_{kj}^{(\ell)}}\right)_{n\times n}. \tag{3.7}$$

By (2.8), the exponential of $X_k^{(\ell)}$ at $A_k^{(\ell)}$ is

$$A_k^{(\ell+1)} := U_k^{(\ell)}[W_k^{(\ell)} \circ (U_k^{(\ell)^*}XU_k^{(\ell)} + 2\Lambda_k^{(\ell)})]\Lambda_k^{(\ell)}[W_k^{(\ell)} \circ (U_k^{(\ell)^*}XU_k^{(\ell)} + 2\Lambda_k^{(\ell)})]U_k^{(\ell)^*}. \tag{3.8}$$

3. All sequences $\{A_k^{(\ell)}\}_{\ell\in\mathbb{N}}$ for $k = 1, \ldots, m$ will converge to the same limit, which is called the Cheap Mean $\overline{A}'(A_1, \ldots, A_m)$:

$$\overline{A}'(A_1, \ldots, A_m) := \lim_{\ell\to\infty} A_1^{(\ell)} = \lim_{\ell\to\infty} A_2^{(\ell)} = \cdots = \lim_{\ell\to\infty} A_m^{(\ell)}. \tag{3.9}$$

From functions of Cheap man algorithm with BW distance, we assume that the computation complexity of cheap algorithm is low and the convergence speed of cheap algorithm is fast. Since the ideas of Inductive mean and Cheap mean are from the Fréchet mean, it is necessary to compare the results and properties between inductive and cheap algorithms.

Chapter 4

Experiments

We proposed three mean algorithms with BW distance, Bures-Wasserstein simple projection mean, Bures-Wasserstein inductive mean, Bures-Wasserstein cheap mean. At this part, we extend the range of experiments to compare more algorithms. Thus, we find the Riemannian distance editions of these algorithms. The experiments part will test 6 different algorithms in terms of efficiency and robustness in Table 4.1.

Type of Distance	Inductive mean	Cheap mean	Simple projection mean
BW distance	BW Inductive mean	BW cheap mean	BW Simple projection mean
Rie distance	Rie Inductive mean	Rie cheap mean	Rie Simple projection mean

Table 4.1: Table of 6 Algorithms

4.1 Compare results of algorithms

At first, some SPD matrices were generated randomly. Then, 6 different algorithms were used to calculate the mean of these SPD matrices. In the table 1, there are results of these algorithms with 10 random 5X5 SPD matrices. These values above the mean matrices are the difference between inductive and the others, which can be computed by:

$$value = \frac{||M_{ij} - M_{i1}||_2}{||M_{i1}||_2} \tag{4.1}$$

Where i = 1,2 and j =2,3, M_{ij} means the i_{th} row and j_{th} column mean matrix in the table.

The means of inductive algorithm with precision parameter of algorithm eps = 1e-6 of Riemannian and Bures-Wasserstein are nearly the same as results of simple projection algorithm.

But they are slightly different from results of cheap mean algorithms. In aspect of difference of Riemannian and Bures-Wasserstein, the difference between of three means of BW algorithms and three means of Riemannian algorithms are obviously larger than the difference within every group. It also implies the results of BW algorithms are different from Riemannian algorithms. With comparing table a and table b, it shows that the results of inductive algorithms with high precision parameter eps = 1e-6 are closer to the results of simple projection algorithms than eps = 1e-2. But since the result differences between eps =1e-6 and eps = 1e-2 are subtle, the results of inductive algorithms with eps = 1e-2 are still reliable.

4.2 Efficiency test

In order to evaluate the efficiency of Riemannian and Bures-Wasserstein algorithms, we compare two of them every time. All data are some SPD matrices generated randomly. We computed mean of these matrices by 6 algorithms. Under the condition of the constant number of dimensions, we measured the change of running time of algorithms with the increase of number of SPD matrices. After testing different constant number of dimensions, Bures-Wasserstein algorithms show better performance on efficiency in the condition that the number of dimensions are low(The number of dimensions of matrices was set to 10). In the Figure 4.2, the green line shows that the ratio of running time of BW and Riemannian cheap mean algorithm are constant(0.3) as the increase of number of matrices. But in Figure 4.3,as the number of matrices rise, the ratio of running time of BW and Riemannian cheap mean algorithm increased.

The simple projection and cheap mean algorithms based on Bures-Wasserstein distance used less time to obtain results than their counterparts based on Riemannian distance. In Figure 4.3 , their distances between BW and Riemannian means are close. Thus, simple and cheap algorithms based on Bures-Wasserstein distance have significant advantage of efficiency on dealing with low number of dimensions and high number of matrices. In contrast to Simple projection and cheap mean algorithms, Inductive algorithms of mean consume a large amount of time at the high precision in Figure 4.4 Therefore, inductive mean algorithms with high precision are not suitable to be applied in practice.

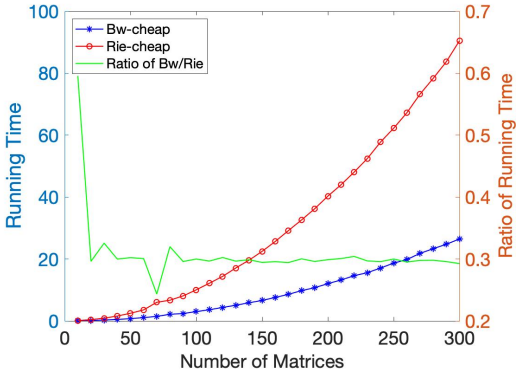
	Inductive(1e-2)					Cheap					Simple Projection				
BW						0.0651					0.0103				
	4.1208	-0.2750	0.4653	-0.8494	-0.1499	3.8898	-0.2633	0.4783	-0.8709	-0.0665	4.1274	-0.2823	0.4731	-0.8567	-0.1366
	-0.2750	3.5848	-0.2542	0.5078	-0.1412	-0.2633	3.5015	-0.2356	0.4877	-0.1904	-0.2823	3.5882	-0.2362	0.4991	-0.1388
	0.4653	-0.2542	2.0736	-0.9125	-0.1850	0.4783	-0.2356	1.9450	-0.8943	-0.1437	0.4731	-0.2362	2.0664	-0.8861	-0.1995
	-0.8494	0.5078	-0.9125	2.5577	-0.5170	-0.8709	0.4877	-0.8943	2.3898	-0.5512	-0.8567	0.4991	-0.8861	2.5785	-0.5391
	-0.1499	-0.1412	-0.1850	-0.5170	3.7303	-0.0665	-0.1904	-0.1437	-0.5512	3.5066	-0.1366	-0.1388	-0.1995	-0.5391	3.7259
Rie						0.0155					0.0049				
	1.9862	-0.2556	0.1536	-0.4926	0.2708	1.9482	-0.2664	0.1539	-0.5105	0.2840	1.9791	-0.2581	0.1546	-0.4959	0.2653
	-0.2556	2.4721	-0.1720	0.2623	-0.3131	-0.2664	2.4921	-0.1690	0.2428	-0.3211	-0.2581	2.4764	-0.1681	0.2550	-0.3131
	0.1536	-0.1720	1.3480	-0.5222	0.1619	0.1539	-0.1690	1.3680	-0.5225	0.1568	0.1546	-0.1681	1.3479	-0.5181	0.1574
	-0.4926	0.2623	-0.5222	1.4409	-0.5309	-0.5105	0.2428	-0.5225	1.4474	-0.5363	-0.4959	0.2550	-0.5181	1.4326	-0.5283
	0.2708	-0.3131	0.1619	-0.5309	2.1871	0.2840	-0.3211	0.1568	-0.5363	2.1766	0.2653	-0.3131	0.1574	-0.5283	2.1920

(a) Inductive mean with the precision eps = 1e-2

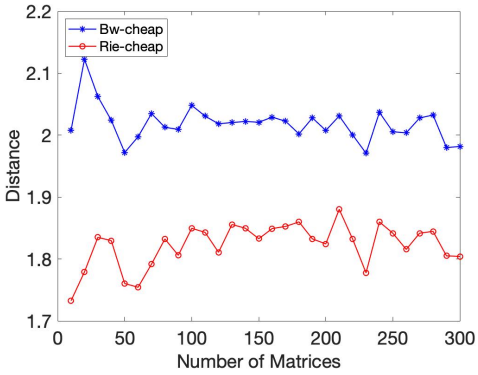
	Inductive(1e-6)					Cheap					Simple Projection				
BW						0.0635					1.3525e-5				
	4.1274	-0.2823	0.4731	-0.8567	-0.1366	3.8898	-0.2633	0.4783	-0.8709	-0.0665	4.1274	-0.2823	0.4731	-0.8567	-0.1366
	-0.2823	3.5882	-0.2362	0.4991	-0.1388	-0.2633	3.5015	-0.2356	0.4877	-0.1904	-0.2823	3.5882	-0.2362	0.4991	-0.1388
	0.4731	-0.2362	2.0664	-0.8861	-0.1995	0.4783	-0.2356	1.9450	-0.8943	-0.1437	0.4731	-0.2362	2.0664	-0.8861	-0.1995
	-0.8567	0.4991	-0.8861	2.5785	-0.5391	-0.8709	0.4877	-0.8943	2.3898	-0.5512	-0.8567	0.4991	-0.8861	2.5785	-0.5391
	-0.1366	-0.1388	-0.1995	-0.5391	3.7259	-0.0665	-0.1904	-0.1437	-0.5512	3.5066	-0.1366	-0.1388	-0.1995	-0.5391	3.7259
Rie						0.0139					5.1324e-7				
	1.9791	-0.2581	0.1546	-0.4959	0.2653	1.9482	-0.2664	0.1539	-0.5105	0.2840	1.9791	-0.2581	0.1546	-0.4959	0.2653
	-0.2581	2.4764	-0.1681	0.2550	-0.3131	-0.2664	2.4921	-0.1690	0.2428	-0.3211	-0.2581	2.4764	-0.1681	0.2550	-0.3131
	0.1546	-0.1681	1.3479	-0.5181	0.1574	0.1539	-0.1690	1.3680	-0.5225	0.1568	0.1546	-0.1681	1.3479	-0.5181	0.1574
	-0.4959	0.2550	-0.5181	1.4326	-0.5283	-0.5105	0.2428	-0.5225	1.4474	-0.5363	-0.4959	0.2550	-0.5181	1.4326	-0.5283
	0.2653	-0.3131	0.1574	-0.5283	2.1920	0.2840	-0.3211	0.1568	-0.5363	2.1766	0.2653	-0.3131	0.1574	-0.5283	2.1920

(b) Inductive mean with the precision eps = 1e-6

Figure 4.1: An example of results of all 6 algorithms. 10 random SPD matrices with number dimensions 5 are computed.

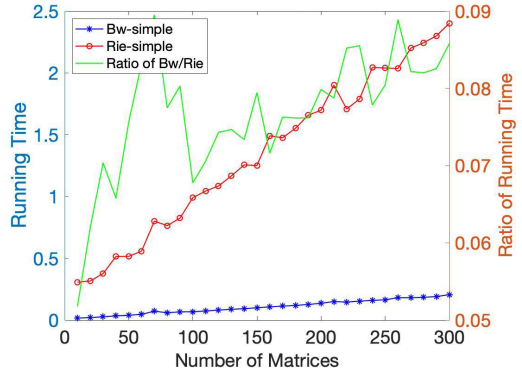


(a) Comparison for Running Time of Cheap Algorithms

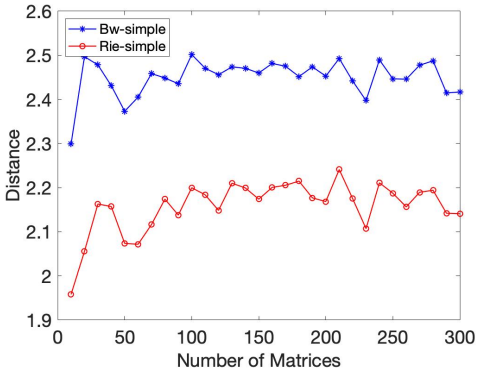


(b) Comparison for two Mean Matrices of Cheap Algorithm

Figure 4.2: Generated different number of 10×10 random SPD matrices and applied BW and Rie cheap mean algorithms to compute means.(a) shows the running time of the two methods. (b) shows the distance, measured by BW distance and Rie distance separately, between the two mean matrices obtained using the two methods. Time unit: Second



(a) Comparison for Running Time of Simple Algorithms



(b) Comparison for two Mean Matrices of Simple Algorithm

Figure 4.3: Generated different number of 10×10 random SPD matrices and applied BW and Rie simple projection mean algorithms to compute means.(a) shows the running time of the two methods. (b) shows the distance, measured by BW distance and Rie distance separately, between the two mean matrices obtained using the two methods. Time unit: Second

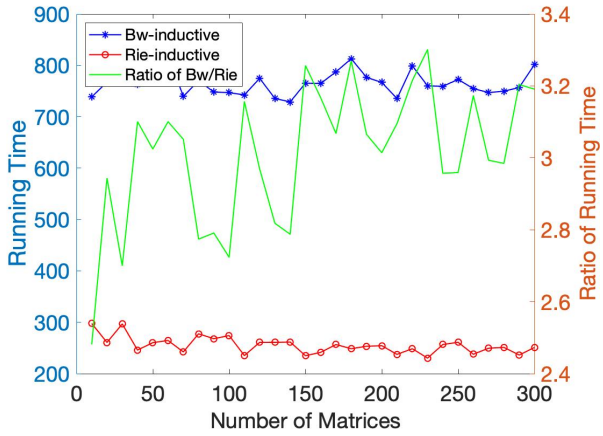


Figure 4.4: Running time of inductive mean algorithms with different number of matrices. Time unit: Second

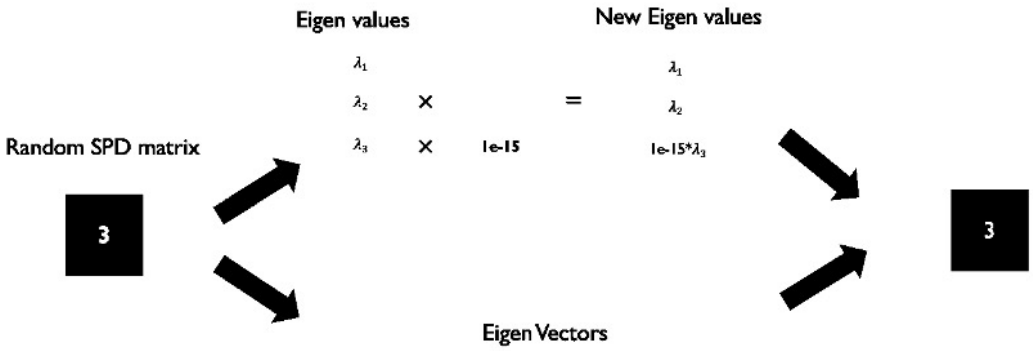


Figure 4.5: Process of generating a matrix that has an eigen value close to zero

4.3 Robustness test

With the comparing the computing formulas of 6 algorithms, logarithm in Riemannian algorithms could be factor that leads to instability of results. Then, we designed a robustness test for two kinds of algorithms. In this case, we selected simple projection algorithms of Bures-Wasserstein and Riemannian as an example to test. Data are 3 SPD matrices. 2 of them are randomly matrices. The third one has 1 eigen value(λ_3) that is close to zero. The third matrix was generated randomly as follow process Figure 4.5. After eigen value decomposition of an SPD matrix, the third eigen value multiply by a parameter that is close to zero to make itself close to zero. Then, the inverse eigenvalue decomposition was conducted with new eigen values and original eigen vectors. Therefore, the new matrix has an eigen value close to zero. In the next step, the mean of these three matrices was compute by a computing mean algorithm

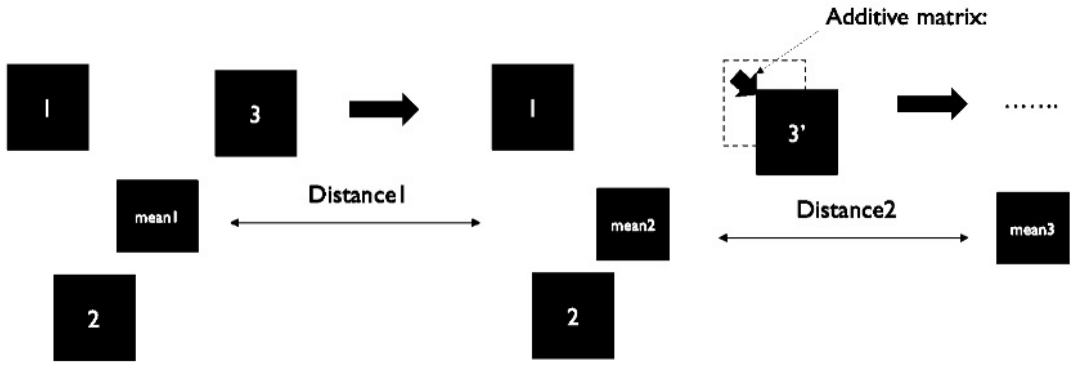


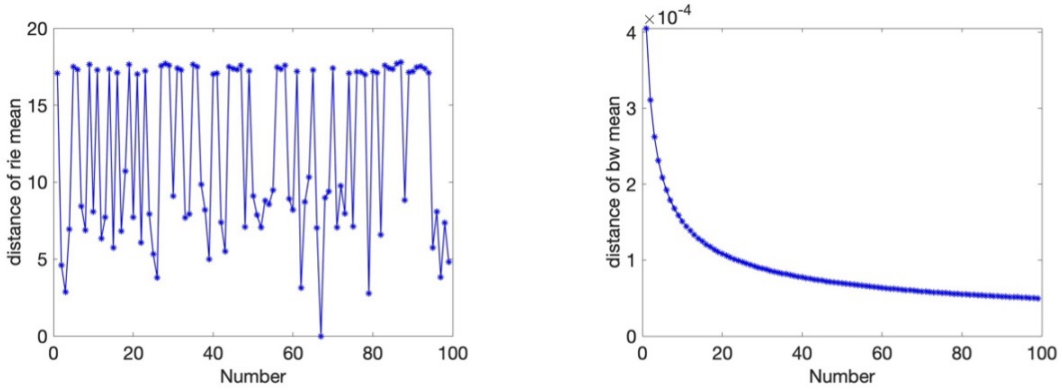
Figure 4.6: The design of robustness test

and denoted by mean1. Then the third matrix will be moved a little by changing the first row and first column element value a little bit. In order to do that, we add a matrix T(4.2) to the third matrix.

$$T = \begin{pmatrix} 1e-5 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix} \tag{4.2}$$

After that, the mean of two SPD matrices with the modified matrix was calculated and denoted by mean2. Repeating this process, we will have means from mean1 to mean n. Next, the distances between every two means that are close will be calculated in Figure 4.6. In the theory, these distances should be stable and become less and less. When we applied simple Bures-Wasserstein and Riemannian mean algorithm on this test and plot distances between every two nearest means in order. Figure 4.7 implies that the Bures-Wasserstein mean algorithm is stable when the SPD matrices contain matrices that have eigen values close to zero($1e-10$). The distances of BW are low (lower than 4e-4) and decreasing continuously. But The algorithm based on Riemannian distance is unstable to cope with this situation. The distances are complex and show a drastic fluctuation from 0 to 17.

In order to find out the reason of instability of Riemannian mean algorithm, we used two kinds of matrices samples. One sample has three random SPD matrices. The other one has two random SPD matrices and one matrix that has an eigen value close to zero to calculate Riemannian mean. For the first sample, the Riemannian mean algorithm stopped in the 23_{rd}



(a) Plot for Riemannian simple projection algorithm

(b) Plot for Bures-Wasserstein simple projection algorithm

Figure 4.7: Distance Plots of robustness test for Rie and BW algorithms

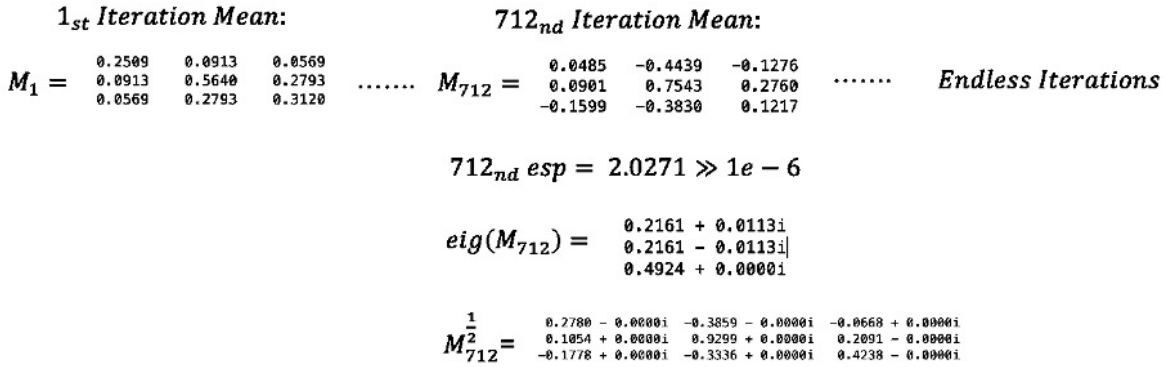


Figure 4.8: Diagram of instability of Riemannian simple mean algorithm

iteration with meeting the condition of $eps < 1e-6$. In this iteration, the mean result was computed.

For the other sample, in every iteration step, the errors produced by logarithm are accumulating in Figure 4.8. When it comes to the 712_{nd} iteration, the mean matrix is too asymmetric to obtain the real result. In the 713_{rd}, eigenvalues of M_{712} and $M_(712)^{1/2}$ will be computed, the results are complex. Then, M_{713} becomes complex. Meanwhile, the precision of result will never less than $1e-6$, which means the condition of stopping the iteration will never be met. Thus, with computing with matrix has eigen values close to zero, the instability of Riemannian algorithm is the result of the instability of logarithm in terms of dealing with matrices that have eigen values close to zero. In other words, accumulation of errors from logarithm lead to the

fail of algorithm. Similarly,Riemannian cheap mean also has the same disadvantage as simple projection. Because they have the same projection process.

In contrast to Riemannian simple projection and cheap mean algorithm, their counterparts of Bures-Wasserstein always have stable results.

Chapter 5

Conclusion

The purpose of the current thesis was to develop new methods of computing the mean of SPD matrices based on Bures-Wasserstein distance. In this case, Bures-Wasserstein inductive mean and cheap mean algorithms were developed theoretically. Then, combining with Riemannian simple projection algorithm, 6 kinds of mean algorithms were tested.

Inductive algorithms of BW and Riemannian have very similar mean results as the results of Simple project algorithms respectively, while cheap mean algorithms have different mean results from the others. In terms of Efficiency, at the same precision level, the simple projection mean has the best performance. Inductive algorithms are extremely time-consuming at high precision. When it comes to robustness, riemannian mean algorithms are failed to deal with the matrices containing extreme low eigen values. But BW mean algorithms run well. Overall, the best SPD matrices mean algorithms among the 6 different algorithms mentioned in the thesis is Simple Projection BW mean algorithm. It has the best performance in aspects of efficiency and robustness.

One of the limitations in the thesis is the lack of examples that these mean algorithms are applied in practice. The recommendations for future work is to confirm the accuracy of classification problem by applying these algorithms in EEG data sets. Maybe, cheap mean algorithms can bring better results on some certain problems. Besides, there is an interesting was found in the experiment, which is that inductive means are very close to simple projection mean. In the future, it is possible to prove the theory behind this situation.

References

[1] Fabien Lotte, Marco Congedo, Anatole Lécuyer, Fabrice Lamarche, and Bruno Arnaldi. A review of classification algorithms for eeg-based brain-computer interfaces. *Journal of Neural Engineering*, 4(2):24, 2007.

[2] G. Schalk, D.J. McFarland, T. Hinterberger, N. Birbaumer, and J.R. Wolpaw. Bci2000: a general-purpose brain-computer interface (bci) system. *IEEE Transactions on Biomedical Engineering*, 51(6):1034–1043, 2004.

[3] Luis Fernando Nicolas-Alonso and Jaime Gomez-Gil. Brain computer interfaces, a review. *Sensors*, 12(2):1211–1279, 2012.

[4] G. Pfurtscheller and C. Neuper. Motor imagery and direct brain-computer communication. *Proceedings of the IEEE*, 89(7):1123–1134, 2001.

[5] Eric C Leuthardt, Gerwin Schalk, Jonathan R Wolpaw, Jeffrey G Ojemann, and Daniel W Moran. A brain-computer interface using electrocorticographic signals in humans. *Journal of Neural Engineering*, 1(2):63–71, 2004.

[6] Janis J Daly and Jonathan R Wolpaw. Brain-computer interfaces in neurological rehabilitation. *Lancet Neurology*, 7(11):1032–1043, 2008.

[7] O. Friman, I. Volosyak, and A. Graser. Multiple channel detection of steady-state visual evoked potentials for brain-computer interfaces. *IEEE Transactions on Biomedical Engineering*, 54(4):742–750, 2007.

[8] Jonathan R Wolpaw, Niels Birbaumer, Dennis J McFarland, Gert Pfurtscheller, and Theresa M Vaughan. Brain-computer interfaces for communication and control. *Clinical Neurophysiology*, 113(6):767–791, 2002.

[9] Alexander J. Doud, John P. Lucas, Marc T. Pisansky, and Bin He. Continuous three-dimensional control of a virtual helicopter using a motor imagery based brain-computer interface. *PLOS ONE*, 6(10), 2011.

[10] H. Ramoser, J. Muller-Gerking, and G. Pfurtscheller. Optimal spatial filtering of single trial eeg during imagined hand movement. *international conference of the ieee engineering in medicine and biology society*, 8(4):441–446, 2000.

[11] Dennis J. McFarland, Lynn M. McCane, Stephen V. David, and Jonathan R. Wolpaw. Spatial filter selection for eeg-based communication. *Electroencephalography and Clinical Neurophysiology*, 103(3):386–394, 1997.

[12] Imran Khan Niazi, Ning Jiang, Olivier Tiberghien, Jørgen Feldbæk Nielsen, Kim Dremstrup, and Dario Farina. Detection of movement intention from single-trial movement-related cortical potentials. *Journal of Neural Engineering*, 8(6):66009, 2011.

[13] C. Gouy-Pailler, M. Congedo, C. Brunner, C. Jutten, and G. Pfurtscheller. Nonstationary brain source separation for multiclass motor imagery. *IEEE Transactions on Biomedical Engineering*, 57(2):469–478, 2010.

[14] Dongrui Wu, Jung-Tai King, Chun-Hsiang Chuang, Chin-Teng Lin, and Tzyy-Ping Jung. Spatial filtering for eeg-based regression problems in brain–computer interface (bci). *IEEE Transactions on Fuzzy Systems*, 26(2):771–781, 2018.

[15] Robert Jenke, Angelika Peer, and Martin Buss. Feature extraction and selection for emotion recognition from eeg. *IEEE Transactions on Affective Computing*, 5(3):327–339, 2014.

[16] Yu Zhang, Guoxu Zhou, Jing Jin, Xingyu Wang, and Andrzej Cichocki. Frequency recognition in ssvep-based bci using multiset canonical correlation analysis. *International Journal of Neural Systems*, 24(4):1450013, 2014.

[17] Teodiano Freire Bastos-Filho, Andre Ferreira, Anibal Cotrina Atencio, Sridhar Arjunan, and Dinesh Kumar. Evaluation of feature extraction techniques in emotional state recognition. In *2012 4th International Conference on Intelligent Human Computer Interaction (IHCI)*, pages 1–6, 2012.

[18] Ruo-Nan Duan, Jia-Yi Zhu, and Bao-Liang Lu. Differential entropy feature for eeg-based emotion classification. In *2013 6th International IEEE/EMBS Conference on Neural Engineering (NER)*, pages 81–84, 2013.

[19] Cristian Torres-Valencia, Mauricio Álvarez-López, and Álvaro Orozco-Gutiérrez. Svm-based feature selection methods for emotion recognition from multimodal data. *Journal on Multimodal User Interfaces*, 11(1):9–23, 2017.

[20] Boualem Boashash, Larbi Boubchir, and Ghasem Azemi. Time-frequency signal and image processing of non-stationary signals with application to the classification of newborn eeg abnormalities. In *2011 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)*, pages 120–129, 2011.

[21] Dean J Krusienski, Eric W Sellers, François Cabestaing, Sabri Bayoudh, Dennis J McFarland, Theresa M Vaughan, and Jonathan R Wolpaw. A comparison of classification techniques for the p300 speller. *Journal of Neural Engineering*, 3(4):299–305, 2006.

[22] B. Blankertz, K.-R. Muller, G. Curio, T.M. Vaughan, G. Schalk, J.R. Wolpaw, A. Schlogl, C. Neuper, G. Pfurtscheller, T. Hinterberger, M. Schroder, and N. Birbaumer. The bci competition 2003: progress and perspectives in detection and discrimination of eeg single trials. *IEEE Transactions on Biomedical Engineering*, 51(6):1044–1051, 2004.

[23] V. Bajaj and R. B. Pachori. Classification of seizure and nonseizure eeg signals using empirical mode decomposition. *international conference of the ieee engineering in medicine and biology society*, 16(6):1135–1142, 2012.

[24] Guohun Zhu, Yan Li, and Peng Paul Wen. Analysis and classification of sleep stages based on difference visibility graphs from a single-channel eeg signal. *IEEE Journal of Biomedical and Health Informatics*, 18(6):1813–1821, 2014.

[25] U. Rajendra Acharya, S. Vinitha Sree, Subhagata Chattopadhyay, Wenwei Yu, and Peng Chuan Alvin Ang. Application of recurrence quantification analysis for the automated identification of epileptic eeg signals. *International Journal of Neural Systems*, 21(3):199–211, 2011.

[26] Hafeez Ullah Amin, Aamir Saeed Malik, Rana Fayyaz Ahmad, Nasreen Badruddin, Nidal Kamel, Muhammad Hussain, and Weng-Tink Chooi. Feature extraction and classification for eeg signals using wavelet transform and machine learning techniques. *Australasian Physical & Engineering Sciences in Medicine*, 38(1):139–149, 2015.

[27] Alexander Craik, Yongtian He, and Jose L Contreras-Vidal. Deep learning for electroencephalogram (eeg) classification tasks: a review. *Journal of Neural Engineering*, 16(3):31001, 2019.

[28] Marco Congedo, Alexandre Barachant, and Rajendra Bhatia. Riemannian geometry for eeg-based brain-computer interfaces; a primer and a review. *Brain-Computer Interfaces*, 4(3):155–174, 2017.

[29] Chuong H Nguyen, George K Karavas, and Panagiotis Artemiadis. Inferring imagined speech using eeg signals: a new approach using riemannian manifold features. *Journal of Neural Engineering*, 15(1):16002, 2018.

[30] Florian Yger, Maxime Berar, and Fabien Lotte. Riemannian approaches in brain-computer interfaces: A review. *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, 25(10):1753–1762, 2017.

[31] Pedro Luiz Coelho Rodrigues, Christian Jutten, and Marco Congedo. Riemannian procrustes analysis: Transfer learning for brain–computer interfaces. *IEEE Transactions on Biomedical Engineering*, 66(8):2390–2401, 2019.

[32] S. Chevallier, E. K. Kalunga, Q. Barthélemy, and E. Monacelli. Review of riemannian distances and divergences, applied to ssvep-based bci. *Neuroinformatics*, pages 1–14, 2020.

[33] Alexandre Barachant, Stéphane Bonnet, Marco Congedo, and Christian Jutten. Multiclass brain–computer interface classification by riemannian geometry. *IEEE Transactions on Biomedical Engineering*, 59(4):920–928, 2012.

[34] Alexandre Barachant, StéPhane Bonnet, Marco Congedo, and Christian Jutten. Classification of covariance matrices using a riemannian-based kernel for bci applications. *Neurocomputing*, 112:172–178, 2013.

[35] Alexandre Barachant, Stéphane Bonnet, Marco Congedo, and Christian Jutten. Bci signal classification using a riemannian-based kernel. In *20th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2012)*, pages 97–102, 2012.

[36] Alexandre Barachant, Stéphane Bonnet, Marco Congedo, and Christian Jutten. Riemannian geometry applied to bci classification. In *LVA/ICA'10 Proceedings of the 9th international conference on Latent variable analysis and signal separation*, volume 6365, pages 629–636, 2010.

[37] Alexandre Barachant, Stephane Bonnet, Marco Congedo, and Christian Jutten. A brain-switch using riemannian geometry. In *5th International Brain-Computer Interface Conference 2011 (BCI 2011)*, pages 64–67, 2011.

[38] Rajendra Bhatia, Tanvi Jain, and Yongdo Lim. On the bures–wasserstein distance between positive definite matrices. *Expositiones Mathematicae*, 37(2):165–191, 2018.

[39] Jorge Antezana, Eduardo Ghiglioni, and Demetrio Stojanoff. Ergodic theorem in hadamard spaces in terms of inductive means. *arXiv preprint. arXiv:1808.02060*, 2018.

[40] Dario Andrea Bini and Bruno Iannazzo. A note on computing matrix geometric means. *Adv. Comput. Math.*, 35(2-4):175–192, 2011.