ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

Extending a two-variable mean to a multi-variable mean

Estelle M. Massart, Julien M. Hendrickx, P.-A. Absil [*]

Université catholique de Louvain - ICTEAM Institute
B-1348 Louvain-la-Neuve - Belgium

Abstract. We consider the problem of extending any two-variable mean $M(\cdot, \cdot)$ to a multi-variable mean, using no other tool than $M(\cdot, \cdot)$ itself. Pálfia proposed an iterative procedure that consists in evaluating successively two-variable means according to a cyclic pattern. We propose here a variant of his procedure to improve the convergence speed. Our approach consists in re-ordering the iterates after each iteration in order to speed up the transfer of information between successive iterates.

1 Introduction

Efficient averaging tools are of major importance in many applications. For example, a common way to denoise data coming from sensors consists in repeating the measure and averaging the values obtained, which justifies the need to develop averaging tools for many data types. Another well-known application of mean computation is the k-means clustering algorithm.

Means have been widely studied on Euclidean spaces, but they have also been considered on several non-Euclidean spaces. The Riemannian barycenter (often termed Karcher mean) has for example been introduced to average data belonging to non-Euclidean spaces endowed with a Riemannian manifold structure. On several manifolds of interest, a closed-form expression exists for the Karcher mean of two data points (which corresponds to the midpoint of the minimizing geodesic—when it exists—between these two data points), but computing a multi-variable Karcher mean becomes more challenging, and usually requires solving an optimization problem on the manifold. We consider here the problem of extending the definition of two-variable means to several variables: a multi-variable mean is computed based on successive evaluations of two-variable means. One of the motivations behind this problem is that if this multi-variable mean is close to the Karcher mean while being cheaper to compute, it could replace the Karcher mean in some applications. Given a set $\mathcal{S}$ endowed with a two-variable mean $M(\cdot, \cdot) : \mathcal{S} \times \mathcal{S} \to \mathcal{S}$, the goal is thus to extend M to a multi-variable mean using no other tools than the $M(\cdot, \cdot)$ operator. We require this procedure to be consistent, meaning that if $\mathcal{S}$ is a vector space and $M(\cdot, \cdot)$ is the two-variable arithmetic mean, then the procedure must converge to the multi-variable arithmetic mean.

[*]This paper presents research results of the Belgian Network DYSCO (Dynamical Systems, Control, and Optimization), funded by the Interuniversity Attraction Poles Programme initiated by the Belgian Science Policy Office.

In [1], Pálfia proposed an iterative algorithm to extend two-variable means to several variables. We propose here a variant of this algorithm aiming at improving the convergence speed. We then use our variant to average symmetric positive definite (SPD) matrices, and we show that it indeed achieves a better accuracy than comparable state-of-the-art methods when run with a small number of iterations (by *comparable methods*, we mean methods that do not require other tools than the ones needed by our method).

We present the algorithm proposed by Pálfia in Section 2 and we describe our variant in Section 3. We then conclude with numerical experiments in Section 4.

2 The Cyclic mean

We recall here the procedure proposed by Pálfia [1], which we term *Cyclic mean.* Let $(A_1, \ldots, A_N)$ be an ordered collection of elements belonging to a given set $\mathcal{S}$, and let $M(\cdot,\cdot) : \mathcal{S}\times\mathcal{S} \to \mathcal{S}$ be a two-variable mean. Let also $\pi = \big(\pi^1, \pi^2, \ldots\big)$ be a sequence of permutations, i.e., $\pi^l = \big(\pi^l(1), \pi^l(2), \ldots, \pi^l(N)\big)$ is a permutation of $(1, \ldots, N)$ for all l. The Cyclic mean $M_\pi(A_1, \ldots, A_N)$ associated to the sequence of permutations π is computed according to Algorithm 1.

Algorithm 1 Cyclic mean

```
   Data: $A_1, ..., A_N \in \mathcal{S}$ and $\pi = \big(\pi^1, \pi^2, \ldots\big)$ a sequence of permutations of $(1, \ldots, N)$.
1: $A_i^0 := A_i \quad \forall i = 1, \ldots, N; \qquad l = 1;$
2: while not converged
3:     for $i = 1, ..., N$
4:         $A_i^l = M(A^{l-1}_{\pi^l(i)}, A^{l-1}_{\pi^l(i+1)}) \quad$ where $\quad \pi^l(N+1) := \pi^l(1)$
5:     $l := l + 1;$
6: return $A_1^l =: M_\pi(A_1, \ldots, A_N)$
```

The first two iterations of this algorithm are illustrated for $N = 5$ and $\pi^1, \pi^2 = (1, \ldots, N)$ on Fig. 1. On this figure, iterates are represented as nodes, and two nodes are linked by an edge if their mean is evaluated during the iteration.

If the mean $M(A, B)$ corresponds to the midpoint of the geodesic between A and B, it has been shown under some conditions on the space $\mathcal{S}$ that Algorithm 1 converges for all π, see [1]; that is, there exists A_* in $\mathcal{S}$ such that $\lim_{l\to\infty} A_i^l = A_*$ for all i.

It has also been pointed out that using $\pi^l = (1, \ldots, N)$ for all l leads to a slow convergence [2]. Different ways to choose π have been proposed to improve the convergence speed of Algorithm 1. For example, in [2], the authors suggest to choose a sequence of random permutations. In [3] a new permutation is built at each iteration, based on the distance between the current iterates; this requires a distance on $\mathcal{S}$ that is cheap to compute, since $N(N-1)/2$ pairwise distances need to be computed each time l is incremented.

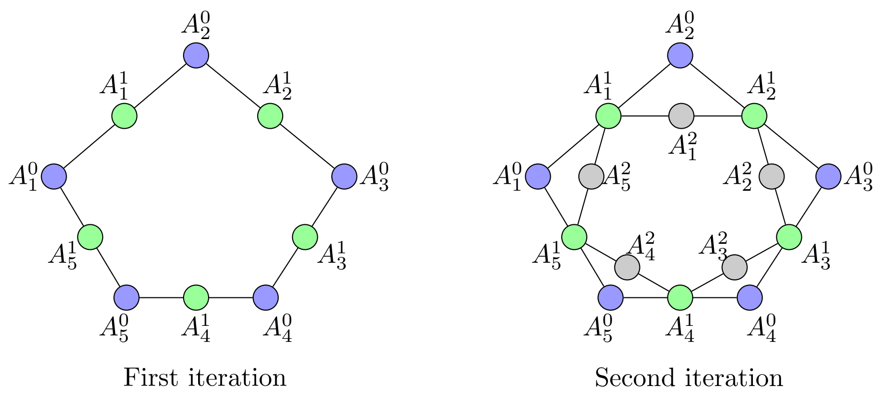

Fig. 1: First steps of Algorithm 1, for $N = 5$ and $\pi_1 = \pi_2 = (1, \ldots, N)$

3 Improving the convergence speed

We build here another sequence of permutations to make the convergence faster. In comparison with the different approaches mentioned in the previous section, the sequence we propose is deterministic and does not require a distance definition on $\mathcal{S}$. The idea is to build this sequence of permutations in order to favor the transfer of information between the iterates, from one iteration to another. This approach is justified by the observation, on Fig. 1, that iterate A_1^2 depends only on the inputs A_1^0, A_2^0 and A_3^0 (the dependence in A_2^0 being considerably greater than the one in A_1^0, A_3^0). Similar observations can be made for iterates $A_i^2,\ i = 1, \ldots, 5$. Actually, using this sequence of permutations, we will have to wait until the 4th iteration to be sure that each iterate contains information from all the inputs. This fact could however be avoided: in the case of Fig. 1, choosing $\pi^1 = (1, 2, 3, 4, 5)$ and $\pi^2 = (1, 3, 5, 2, 4)$ would ensure that four of the five inputs already impact the iterates $A_i^2,\ i = 1, \ldots, N$.

Our goal will be to find at each iteration a set of pairs of most complementary nodes (nodes that contain mostly information related to different inputs, such as nodes A_1^1 and node A_3^1 on Fig. 1) and to choose a permutation leading to the averaging of those pairs of nodes. We expect this to improve the communication between the iterates, hence the convergence speed.

We measure the complementarity between a pair of nodes by computing the weights that have been given to the inputs for their computation, since the beginning of the algorithm. The nodes with the most different weights are the most complementary. These weights are computed and updated from one iteration to another using the incidence matrices associated to the cycles along which the means are computed at each iteration (such as those represented on Fig. 1).

We define thus for each iteration l a matrix $B^{(l)} \in \{0, 1/2\}^{N \times N}$, such that $B^{(l)}(i, j) = 1/2$ if $\pi^l(j) = i$ or $\pi^l(j+1) = i$, and $B^{(l)}(i, j) = 0$ otherwise. Matrix

$B^{(l)}$ corresponds thus to the incidence matrix, normalized to become row- and column-stochastic, of the cycle considered at iteration l (and corresponding to permutation π^l, see Fig. 1). Let now $P^{(l)}$ be the product of the transposed matrices $B^{(l)T}, \dots, B^{(1)T}$:

$$P^{(l)} = B^{(l)T} \cdot B^{(l-1)T} \cdot \dots \cdot B^{(1)T} \qquad l = 1, 2, \dots$$

We can see that $P^{(l)}$ is a row– and column–stochastic matrix in which row i contains the weights given to the inputs $A_1^0, \dots, A_N^0$ in the computation of iterate A_i^l. We can then define a complementarity matrix at each iteration l:

$$C^{(l)}(i,j) = \left(\sum_{k=1}^{N} \left(P^{(l)}(i,k) - P^{(l)}(j,k) \right)^2 \right)^{1/2}, \; i,j = 1, \dots, N \tag{1}$$

The complementarity between two iterates A_i^l and A_j^l is thus defined as the Euclidean distance between the corresponding rows of matrix $P^{(l)}$. At each iteration l, we want to find the cycle maximizing the total complementarity along its edges. This amounts to finding a maximum-length cycle in a complete graph, in which the length of an edge (A_i^l, A_j^l) is given by $C^{(l)}(i,j)$. This problem is NP-hard, but we compute an approximate solution. The whole procedure to generate the sequence of permutation π is given in Algorithm 2.

Algorithm 2 Construction of the sequence π to give as input to Algorithm 1

```
1: Choose $\pi^1 := (1, \dots, N)$.
2: for $l = 1, 2, \dots$ do
3:     Compute matrices $B^{(l)}$, $P^{(l)}$ and $C^{(l)}$ as explained above.
4:     Find $(\pi^{l+1}(1), \pi^{l+1}(2))$ the two nodes incident to the longest edge of the
       complete graph whose edge lengths are contained in matrix $C^{(l)}$ ($\pi^{l+1}(1)$
       and $\pi^{l+1}(2)$ are ordered according to a lexicographic order).
5:     for $i = 3, \dots, N$ do
6:         Find $\pi^{l+1}(i)$ the most distant node to $\pi^{l+1}(i-1)$ such that
           $\pi^{l+1}(i) \neq \pi^{l+1}(i-2), \dots, \pi^{l+1}(1)$. If different nodes are possible, a
           lexicographic order is again used.
7: return $\pi = (\pi^1, \pi^2, \dots)$
```

4 Numerical experiments

As in [1], we focus on a particular mean computation problem: the computation of a mean of a collection of symmetric positive definite (SPD) matrices. This task appears in various applications, e.g., in medical imaging, elasticity and radar processing.

The Karcher mean is usually used to average SPD matrices, and is defined as:

$$M_{\mathrm{K}}(A_1, \dots, A_N) = \underset{X \in \mathbb{S}_+^n}{\operatorname{argmin}} \sum_{i=1}^{N} \delta(A_i, X)^2, \tag{2}$$

where $\delta(A,B) = ||\log(A^{-1/2}BA^{-1/2})||_F$ is the affine-invariant Riemannian distance between A and B and $\mathbb{S}_+^n$ is the cone of SPD matrices. When $N = 2$, the Karcher mean can be expressed explicitly as

$$M_{\mathrm{K}}(A_1, A_2) = A_1^{\frac{1}{2}}(A_1^{-\frac{1}{2}}A_2A_1^{-\frac{1}{2}})^{\frac{1}{2}}A_1^{\frac{1}{2}},$$

but no closed-form expression is known for the Karcher mean of more than two SPD matrices, which then has to be computed by an iterative process (see [4]).

We apply the Cyclic mean to extend the definition of the two-variable Karcher mean to more variables. If it results in a good estimate of the exact Karcher mean, then it makes a valuable substitute for the Karcher mean in practical applications for which the computation of the exact Karcher mean is too expensive. We therefore compare the results obtained by the Cyclic mean, using different sequences of permutations π, to a reference value for the Karcher mean obtained using a gradient descent, taking as initial guess the arithmetic mean of the inputs and choosing automatically the step-length as proposed in [4]. We measure the estimation error between a result $M_\pi(A_1, \ldots, A_N)$ of the extension method and the Karcher mean $M_{\mathrm{K}}(A_1, \ldots, A_N)$ according to:

$$E_{\mathrm{rel}} = \frac{\delta(M_\pi(A_1, \ldots, A_N), M_{\mathrm{K}}(A_1, \ldots, A_N))}{\frac{1}{N}\sum_{i=1}^{N}\delta(A_i, M_{\mathrm{K}}(A_1, \ldots, A_N))} \tag{3}$$

The error is thus normalized according to the average error that would be obtained by taking as estimation for the Karcher mean one of the input matrices themselves. We compare the performances achieved by our method (termed *Cyclic_Cheap mean* in the rest of this discussion) to those obtained with two other comparable methods: the *Cyclic_Id mean*, corresponding to the choice $\pi^l = (1, \ldots, N)$ for all $l = 1, 2, \ldots$, and the *Cyclic_Random mean* which corresponds to permutations randomly chosen at each iteration. The performances obtained are presented on Fig. 2 and 3; they have been averaged on 100 sets of input matrices, randomly chosen from a Wishart distribution. These figures illustrate the slow convergence of the *Cyclic_Id mean*, and the fact that, in addition to being deterministic, our approach yields a better accuracy than the two other methods after a small number of iterations. Furthermore, if the computation cost of the Cyclic mean is driven by the number of evaluations of two-variable means (as it is the case when working with a few large-scale SPD matrices), then the smaller number of iterations translates in a smaller computational cost.

References

[1] Miklós Pálfia. Means in metric spaces and the center of mass. *Journal of Mathematical Analysis and Applications*, 381(1):383–391, 2011.

[2] Ben Jeuris and Raf Vandebril. Geometric mean algorithms based on harmonic and arithmetic iterations. In *Geometric Science of Information*, pages 785–793. Springer, 2013.

[3] Miklós Pálfia. The Riemann barycenter computation and means of several matrices. *Int. J. Comput. Math. Sci*, 3(3):128–133, 2009.

[4] Dario A Bini and Bruno Iannazzo. Computing the Karcher mean of symmetric positive definite matrices. *Linear Algebra and its Applications*, 438(4):1700–1710, 2013.

ESANN 2016 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 27-29 April 2016, i6doc.com publ., ISBN 978-287587027-8. Available from http://www.i6doc.com/en/.

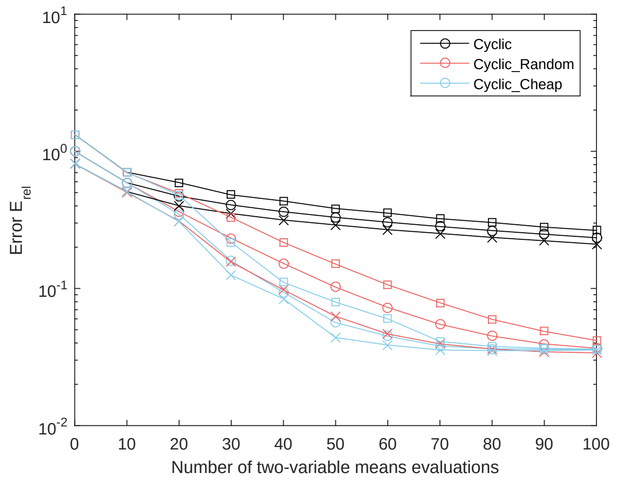

Fig. 2: Test on sets of 10 matrices of size 20×20, chosen from a Wishart distribution. The circles give the average error among the N iterates A_i^j, $i = 1, \ldots, N$, while the squares and crosses give respectively the minimum and maximum errors among those iterates.

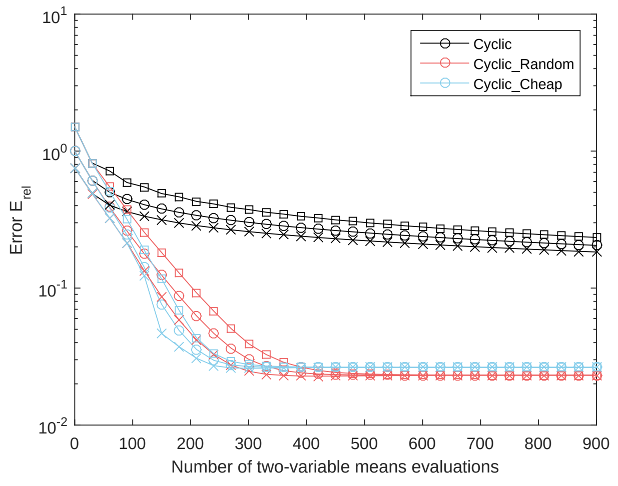

Fig. 3: Test on sets of 30 matrices of size 20×20, chosen from a Wishart distribution.