

Geometrical inverse matrix approximation for least-squares problems and acceleration strategies

Jean-Paul Chehab* Marcos Raydan †

February 21, 2019

Abstract

We extend the geometrical inverse approximation approach for solving linear least-squares problems. For that we focus on the minimization of $1 - \cos(X(A^T A), I)$, where A is a given rectangular coefficient matrix and X is the approximate inverse. In particular, we adapt the recently published simplified gradient-type iterative scheme MinCos to the least-squares scenario. In addition, we combine the generated convergent sequence of matrices with well-known acceleration strategies based on recently developed matrix extrapolation methods, and also with some deterministic and heuristic acceleration schemes which are based on affecting, in a convenient way, the steplength at each iteration. A set of numerical experiments, including large-scale problems, are presented to illustrate the performance of the different accelerations strategies.

Key words: Inverse approximation, cones of matrices, matrix acceleration techniques, gradient-type methods.

1 Introduction

The development of inverse matrix approximation strategies for solving linear least-squares problems is an active research area since they play a key role in a wide variety of science and engineering applications involving ill-conditioned large matrices (sparse or dense); see e.g., [8, 9, 10, 13, 14, 15, 19, 24, 30, 32, 34, 35].

In this work, for a given real rectangular $m \times n$ ($m \geq n$) matrix A , we will obtain inverse approximations based on minimizing the positive-scaling-invariant function $\hat{F}(X) = 1 - \cos(X(A^T A), I)$ on a suitable closed and bounded subset of the cone of symmetric and positive semidefinite matrices (PSD). Therefore, our inverse approximations will remain in the PSD cone, in sharp contrast with the standard approach of minimizing the Frobenius norm of the residual $(I - X(A^T A))$, for which a symmetric and positive definite approximation cannot be guaranteed; see e.g., [16, 22].

For the minimization of $\hat{F}(X)$ we will extend and adapt the simplified gradient-type scheme MinCos, introduced in [12], to the linear least-squares scenario. Moreover, we will adapt and apply some

*LAMFA, UMR 7352, Université de Picardie Jules Verne, 33 rue Saint Leu, 80039 Amiens France(Jean-Paul.Chehab@u-picardie.fr)

†Departamento de Cómputo Científico y Estadística, Universidad Simón Bolívar, Ap. 89000, Caracas 1080-A, Venezuela (mraydan@usb.ve)

well-known modern matrix acceleration strategies to the generated convergent sequences. In particular, we will focus our attention on the use of the simplified topological ε -algorithms [4, 5], and also on the extension of randomly-chosen steplength acceleration strategies [31], as well as the extension of some recent nonmonotone gradient-type choices of steplengths [20, 36].

The rest of the document is organized as follows. In Section 2, we recall the MinCos method for matrices in the *PSD* cone, and briefly describe its most important properties. In Section 3, we develop the extended and adapted version of the MinCos method for solving linear least-squares problems. In Section 4, we describe the different adapted acceleration strategies to hopefully observe a faster convergence of the generated sequences. In Section 5, we present experimental numerical results to illustrate the performance of the adapted algorithm for least-squares problems, and also to illustrate the advantages of using acceleration techniques over a set of problems, including large-scale matrices.

2 The MinCos method

Let us recall the MinCos method for the minimization of $F(X) = 1 - \cos(XA, I)$, when A is an $n \times n$ real symmetric and positive definite matrix.

Algorithm 1 : MinCos (simplified gradient approach on $F(X) = 1 - \cos(XA, I)$)

- 1: Given $X^{(0)} \in PSD$ (commuting with A)
 - 2: **for** $k = 0, 1, \dots$ until a stopping criterion is satisfied, **do**
 - 3: **Set** $w_k = \langle X^{(k)}A, I \rangle$
 - 4: **Set** $\widehat{D}_k = -\frac{1}{n} \left(\frac{w_k}{n} X^{(k)}A - I \right)$
 - 5: **Set** $\alpha_k = \left| \frac{n \langle \widehat{D}_k A, I \rangle - w_k \langle X^{(k)}A, \widehat{D}_k A \rangle}{\langle \widehat{D}_k A, I \rangle \langle X^{(k)}A, \widehat{D}_k A \rangle - w_k \|\widehat{D}_k A\|_F^2} \right|$
 - 6: **Set** $Z^{(k+1)} = X^{(k)} + \alpha_k \widehat{D}_k$
 - 7: **Set** $X^{(k+1)} = s \sqrt{n} \frac{Z^{(k+1)}}{\|Z^{(k+1)}A\|_F}$, where $s = 1$ if $\text{trace}(Z^{(k+1)}A) > 0$, $s = -1$ else
 - 8: **end for**
-

This method has been successfully introduced in [12], and can be seen as an improved version of the Cauchy Method applied to the minimization of the merit function

$$F(X) = 1 - \cos(XA, I) = 1 - \frac{\langle XA, I \rangle}{\|XA\|_F \|I\|_F},$$

where I is the identity matrix, $\langle A, B \rangle = \text{trace}(A^T B)$ is the Frobenius inner product in the space of matrices and $\|\cdot\|_F$ is the associated Frobenius norm. In here PSD refers to the positive semi-definite closed cone of square matrices which possesses a rich geometrical structure; see, e.g., [1, 11].

In Remark 2.1 we summarize the most important properties of Algorithm 2 (see [12]).

Remark 2.1 1. *The minimum of $F(X)$ is reached at X such that $AX = \alpha I$. But if we impose $\|AX\|_F = \|I\|_F = \sqrt{n}$, we have $\alpha = \pm 1$. If in addition we impose $\text{trace}(XA) \geq 0$, we have $XA = I$.*

2. By construction $\|X^{(k)}A\|_F = \sqrt{n}$, for all $k \geq 1$. If we choose $X^{(0)}$ such that $\text{trace}(X^{(0)}A) = \langle X^{(0)}A, I \rangle > 0$ then by construction all the iterates remain in the PSD cone. Moreover, if in addition $X^{(0)}A = AX^{(0)}$, then $X^{(k)}A = AX^{(k)}$ and $Z^{(k)}A = AZ^{(k)}$, for all $k \geq 0$.
3. Unless we are at the solution, the search direction \widehat{D}_k is a descent direction for the function F at X . The steplength $\alpha_k > 0$ is the optimal choice, i.e., the positive steplength that (exactly) minimizes the function $F(X)$ along the direction \widehat{D}_k . Furthermore, $Z^{(k)}$, $X^{(k)}$, and $X^{(k)}A$ in the MinCos Algorithm are symmetric matrices for all k , which are also uniformly bounded away from zero, and so the algorithm is well-defined.

For completeness, we state the convergence result concerning the MinCos method (for the proof see [12]).

Theorem 2.1 *The sequence $\{X^{(k)}\}$ generated by the MinCos Algorithm converges to A^{-1} .*

3 The MinCos method for least-squares problems

Let us now consider linear systems involving the real rectangular $m \times n$ ($m \geq n$) matrix A , for which solutions do not exist. An interesting and always robust available option is to use the least-squares approach, i.e., to solve instead the normal equations, which involve solving a linear system with the square matrix $A^T A$ that belongs to the PSD cone. Let us assume that A is full column rank, i.e., that $A^T A$ is symmetric and positive definite. In that case, it is always an available (default) option, although not recommendable, to apply Algorithm 1 directly on the matrix $A^T A$. Nevertheless, to avoid multiplications with the matrix A^T (which is usually not available for practical applications), and also to avoid unnecessary and numerically risky calculations, we will adapt each one of the steps of the MinCos algorithm. For that we first need to recall that, using properties of the trace operator, for any matrices W_1 , W_2 , and W_3 with the proper sizes, it follows that

$$\langle W_1, W_2 W_3 \rangle_F = \langle W_2^T W_1, W_3 \rangle_F = \langle W_1 W_3^T, W_2 \rangle_F. \quad (1)$$

For any given matrix Y for which Y^T is available, using (1), we obtain that

$$\langle Y A^T A, I \rangle = \langle (A Y^T)^T A, I \rangle = \langle A, A Y^T \rangle = \langle A Y^T, A \rangle. \quad (2)$$

Hence, using (2) and the fact that \widehat{D}_k is symmetric, it follows that $\langle \widehat{D}_k A^T A, I \rangle = \langle A \widehat{D}_k, A \rangle$, and since $X^{(k)}$ is symmetric, $\langle X^{(k)} A^T A, I \rangle = \langle A X^{(k)}, A \rangle$. Moreover,

$$\langle X^{(k)} A^T A, \widehat{D}_k A^T A \rangle = \langle (A X^{(k)})^T A, (A \widehat{D}_k)^T A \rangle.$$

Similarly, we obtain that $\|Y A^T A\|_F^2 = \|(A Y^T)^T A\|_F^2$. Summing up we obtain the following extended version of the MinCos algorithm for minimizing $\widehat{F}(X) = 1 - \cos(X(A^T A), I)$, where A is a given rectangular matrix.

Algorithm 2 : MinCos for minimizing $\widehat{F}(X) = 1 - \cos(X(A^T A), I)$

- 1: Given $X^{(0)} \in PSD$ (commuting with $A^T A$)
 - 2: **for** $k = 0, 1, \dots$ until a stopping criterion is satisfied, **do**
 - 3: **Set** $C_k = AX^{(k)}$ and $w_k = \langle C_k, A \rangle$
 - 4: **Set** $\widehat{D}_k = -\frac{1}{n} (\frac{w_k}{n} C_k^T A - I)$, $B_k = A\widehat{D}_k$, and $\mu_k = \langle B_k^T A, C_k^T A \rangle$
 - 5: **Set** $\beta_k = \langle B_k, A \rangle$ and $\alpha_k = \left| \frac{n\beta_k - w_k\mu_k}{\beta_k\mu_k - w_k\|B_k^T A\|_F^2} \right|$
 - 6: **Set** $Z_{temp} = X^{(k)} + \alpha_k \widehat{D}_k$ and $Z^{(k+1)} = (Z_{temp} + Z_{temp}^T)/2$
 - 7: **Set** $X^{(k+1)} = s \frac{\sqrt{n} Z^{(k+1)}}{\|(AZ^{(k+1)})^T A\|_F}$, where $s = 1$ if $trace((AZ^{(k+1)})^T A) > 0$,
 $s = -1$ else
 - 8: **end for**
-

We note that for taking advantage of the adapted formulas obtained above, which are based mainly on (2), it is important to maintain the symmetry of all the iterates in Algorithm 2. For that, let us observe that at Step 6, $Z^{(k+1)}$ is actually obtained as the closest symmetric matrix to the original one given by $Z^{(k+1)} = X^{(k)} + \alpha_k \widehat{D}_k$. This additional calculation represents an irrelevant computational cost as compared to the rest of the steps in the algorithm, but at the same time it represents a safety procedure to avoid the numerical loss of symmetry that might occur when A is large-scale and ill-conditioned. We also note that for any matrix A , $A^T A$ is in the PSD cone, and so all the results presented in Section 2 apply for algorithm 2, in particular since A is full column rank then by Theorem 2.1 the sequence $\{X^{(k)}\}$ converges to $(A^T A)^{-1}$.

4 Acceleration strategies

The sequence of matrices $\{X^{(k)}\} \subset \mathbb{R}^{n \times n}$ generated either by Algorithm 1 or by Algorithm 2 can be viewed as simplified and improved versions of the CauchyCos algorithm developed in [12], which is a specialized version of the Cauchy (steepest descent) method for minimizing $F(X)$. Nevertheless, both algorithms are gradient-type methods, and as a consequence they can be accelerated using some well-known acceleration strategies which extend effective scalar and vector modern sequence acceleration techniques; see, e.g., [2, 3]. We note that the sequence $\{X^{(k)}\}$, generated by Algorithm 1 or by Algorithm 2, converges to the limit point A^{-1} or $(A^T A)^{-1}$, respectively.

For our first acceleration strategy, we will focus on the matrix version of the so-called simplified topological ε -algorithms, which belongs to the general family of acceleration schemes that transform the original sequence to produce a new one that hopefully will converge faster to the same limit point; see e.g., [4, 7, 23, 25, 26, 27, 28]. For a full historical review on this topic as well as some other related issues we recommend [6]. To use the simplified topological ε -algorithms, we take advantage of the recently published Matlab package EPSfun¹ [5], that effectively implements the most advanced options of that family, including the matrix sequence versions. In particular, we focus on the matrix versions of the specific simplified topological ε -algorithms 1 (STEAl) and the simplified topological ε -algorithms 2 (STEAl2) using the restarted option (RM), which have been implemented in the EPSfun

¹The Matlab package EPSfun is freely available at <http://www.netlib.org/numeralgo/>

package, including four possible variants. The details of all the options that can be used in the EPSfun package are fully described in [5].

For our second acceleration strategy, we will adapt a procedure proposed and analyzed in [31] for the minimization of convex quadratics, and that can be viewed as a member of the family for which the acceleration is generated by the method itself, i.e., at once and dynamically only one accelerated sequence is generated; see, e.g., [10]. This approach can take advantage of the intrinsic characteristics of the method that generates the original sequence, which in some cases has proved to produce more effective accelerations than the standard approach that transforms the original one to produce an independent accelerated one [18]. For our specific algorithms, the second strategy is obtained by relaxing the optimal descent parameter α_k as

$$\alpha_k \leftarrow \theta_k \alpha_k,$$

where θ_k is at each step randomly chosen in (a, b) , preferably $(a, b) = (1 - \eta, 1 + \eta)$ for $0 < \eta < 1$, following a uniform distribution. In order to study the interval in which $F(X^{(k+1)})$ attains a value less than or equal to $F(X^{(k)})$ (see [31]), let us define

$$\phi_k(t) = F(X^{(k)} + t\alpha_k \widehat{D}_k) - F(X^{(k)})$$

where \widehat{D}_k is the descent direction and α_k the optimal parameter given in Algorithm 1. Notice that all the results that follow for F and Algorithm 1 also applies automatically to \widehat{F} and Algorithm 2.

Proposition 4.1 *For all k , it holds that $\phi_k(0) = 0$, $\phi'_k(0) < 0$, and $\phi_k(1) < 0$.*

Proof. $\phi_k(0) = 0$ by construction and $\phi'_k(0) < 0$ since \widehat{D}_k is a descent direction. Now $\phi_k(1) < 0$ because α_k minimizes $F(X^{(k)} + \alpha \widehat{D}_k)$. ■

Our next result is concerned with the right extreme value of the interval.

Proposition 4.2 *If there exists $t_k^* > 1$ such that $\phi_k(t_k^*) = 0$, then we have*

$$t_k^* = \frac{2 \left(\langle X^{(k)} A, \widehat{D}_k A \rangle \langle X^{(k)} A, I \rangle^2 - n \langle \widehat{D}_k A, I \rangle \right)}{\alpha_k \left(\|\widehat{D}_k A\|_F^2 \langle X^{(k)} A, I \rangle^2 - n \langle \widehat{D}_k A, I \rangle^2 \right)}$$

Proof. Forcing $\phi_k(t) = 0$ implies that

$$\begin{aligned} & \|X^{(k)} A\|_F^2 \left(\langle X^{(k)} A, I \rangle + t\alpha_k \langle \widehat{D}_k A, I \rangle \right)^2 \\ &= \langle X^{(k)} A, I \rangle^2 \left(\|X^{(k)} A\|_F^2 + 2\alpha_k t \langle X^{(k)} A, \widehat{D}_k A \rangle + \alpha_k^2 t^2 \|\widehat{D}_k A\|_F^2 \right), \end{aligned}$$

and we obtain

$$\begin{aligned} & t^2 \alpha_k^2 \left(\|X^{(k)} A\|_F^2 \langle \widehat{D}_k A, I \rangle^2 - \langle X^{(k)} A, I \rangle^2 \|\widehat{D}_k A\|_F^2 \right) + \\ & + 2t\alpha_k \left(\langle \widehat{D}_k A, I \rangle \|X^{(k)} A\|_F^2 - \langle X^{(k)} A, \widehat{D}_k A \rangle \langle X^{(k)} A, I \rangle^2 \right) = 0. \end{aligned}$$

Now, dividing by $\alpha_k t \neq 0$ and using $\|X^{(k)} A\|_F = \sqrt{n}$, it follows that

$$t\alpha_k \left(n \langle \widehat{D}_k A, I \rangle^2 - \langle X^{(k)} A, I \rangle^2 \|\widehat{D}_k A\|_F^2 \right) = 2 \left(\langle \widehat{D}_k A, I \rangle n - \langle X^{(k)} A, \widehat{D}_k A \rangle \langle X^{(k)} A, I \rangle^2 \right),$$

and the result is established. ■

Remark 4.1 *At each iteration k we can compute t_k^* when it exists and relax randomly α_k with θ_k uniformly randomly chosen in $(0, t_k^*)$. Notice that the computation of t_k^* is obtained for free: all the terms defining t_k^* have been previously computed to obtain α_k . Nevertheless, as we will discuss in our next section, the uniformly random choice $\theta_k \simeq U([1/2, 3/2])$ is an efficient practical option, which clearly represents a heuristic proposal.*

Since \widehat{D}_k is a gradient-type descent direction, for our third acceleration strategy, we will adapt the steplength associated with the recently developed ABBmin low-cost gradient method [20, 36], which has proved to be very effective in the solution of general nonlinear unconstrained optimization problems [20, 33]. It is worth mentioning that the ABBmin method is a nonmonotone scheme for which convergence to local minimizers has been established [33]. As in the case of the randomly relaxed acceleration, this approach can also be viewed as a member of the family for which the acceleration is generated by the method itself. For our specific algorithms, the third strategy is obtained by substituting the optimal steplength α_k by

$$\widehat{\alpha}_k = \begin{cases} \min\{\alpha_j^{BB2} : j = \max\{1, k - M\}, \dots, k\}, & \text{if } \alpha_k^{BB2}/\alpha_k^{BB1} < \tau; \\ \alpha_k^{BB1}, & \text{otherwise} \end{cases} \quad (3)$$

where $\tau \in (0, 1)$ (in practice $\tau \approx 0.8$), M is a small nonnegative integer, and the involved parameters are given by

$$\alpha_k^{BB1} = \frac{\|S^{(k-1)}\|_F^2}{\langle S^{(k-1)}, Y^{(k-1)} \rangle} \quad \text{and} \quad \alpha_j^{BB2} = \frac{\langle S^{(j-1)}, Y^{(j-1)} \rangle}{\|Y^{(j-1)}\|_F^2} \quad \text{for all } 1 \leq j \leq k,$$

where $S^{(j-1)} = X^{(j)} - X^{(j-1)}$ and $Y^{(j-1)} = \widehat{D}_j - \widehat{D}_{j-1}$, for all j .

A geometrical as well as an algebraic motivation for the choice $\widehat{\alpha}_k$ in (3) can be found in [20, 33, 36]. In particular, they establish an interesting connection between α_k^{BB1} , α_k^{BB2} , and the ratio $\alpha_k^{BB2}/\alpha_k^{BB1}$, with the eigenvalues (and eigenvectors) of the underlying Hessian of the objective function. In general, the relationship between the choice of steplength, in gradient-type methods, and the eigenvalues and eigenvectors of the underlying Hessian of the objective function is well-known, and for nonmonotone methods can be traced back to [21, pp. 117-118]; see also [17, 31].

5 Illustrative numerical examples

To give further insight into the behavior of the MinCos method for least-squares problems and the three discussed acceleration strategies, we present the results of some numerical experiments. All computations were performed in Matlab, using double precision, which has unit roundoff $\mu \approx 1.1 \times 10^{-16}$. Our initial guess is chosen as $X^{(0)} = \beta I$, where $\beta > 0$ is fixed to satisfy the scaling performed at Step 7 in Algorithm 1 and also in Algorithm 2, i.e., $\beta = \sqrt{n}/\|A\|_F$ for Algorithm 1 and $\beta = \sqrt{n}/\|A\|_F^2$ for Algorithm 2. In every experiment, we stop the process when the merit function $F(X^{(k)})$ (Algorithm 1) or $\widehat{F}(X^{(k)})$ (Algorithm 2) is less than or equal to ϵ , for some small $\epsilon > 0$. Concerning the package EPSfun, we use the STEA2 option which has proved to be more effective than STEA1 for our experiments, with different choices of the key parameters NCYLCE and MCOL.

We notice that when using the STEA2 strategy, the number of reported iterations is given by NCYLCE times MCOL. Concerning the ABBmin method, we set $\tau = 0.8$ and $M = 10$ in all cases. We consider test matrices from the Matlab gallery (Poisson2D, Poisson3D, Wathen, Lehmer, and normal), and also from the Matrix Market [29].

5.1 PSD matrices using Algorithm 1

For our first set of experiments we consider symmetric positive definite matrices that are not badly conditioned as the ones obtained in a 2D or 3D discretization of Poisson equations with Dirichlet boundary conditions. In Figures 1 and 2 we report the convergence behavior of the MinCos method (Algorithm 1), the Random Mincos acceleration, and the STEA2 acceleration for different values of NCYLCE and MCOL, when applied to the gallery matrices Poisson2D with $n = 100$ and Poisson3D with $n = 1000$, respectively. We notice that in both cases the two acceleration schemes need significantly less iterations than the MinCos method to achieve the required accuracy. We also note, in Figure 1, that STEA2 outperforms the Random Mincos acceleration. However, we can notice in Figure 2 that when the size of the matrix increases, as well as the condition number, then the Random Mincos acceleration outperforms the STEA2 scheme.

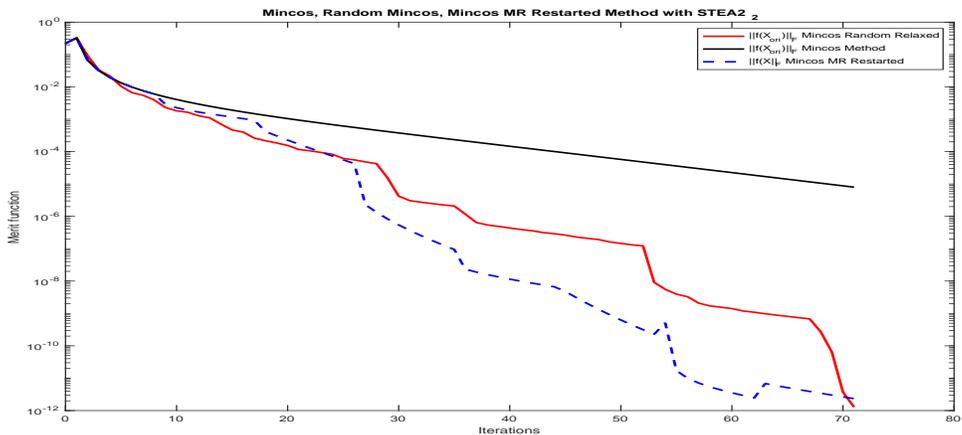


Figure 1: Convergence history of MinCos, Random Mincos and STEA2 for the 2D Poisson matrix for $n = 100$, NCYLCE=8, and MCOL=8.

For the next experiments either the matrix is sparse and of large size or the matrix is dense. In these cases, we will focus our attention on the Random Mincos and the ABBmin acceleration strategies, which are well suited for large problems, since they only require the storage of the direction matrix D_k and very low additional computational cost per iteration. These results are reported in Figure 3 (for the Poisson 2D matrix with $n = 900$) and in Figure 4 (for the Lehmer matrix with $n = 20$). We note that the Lehmer matrices, from the Matlab Gallery, are dense. We can observe that the ABBmin acceleration represents an aggressive option that sometimes outperforms the Random Mincos scheme (for example in Figure 3), but it shows a highly nonmonotone behavior that can produce unstable calculations. The highly nonmonotone performance of the ABBmin scheme can be noticed in Figure

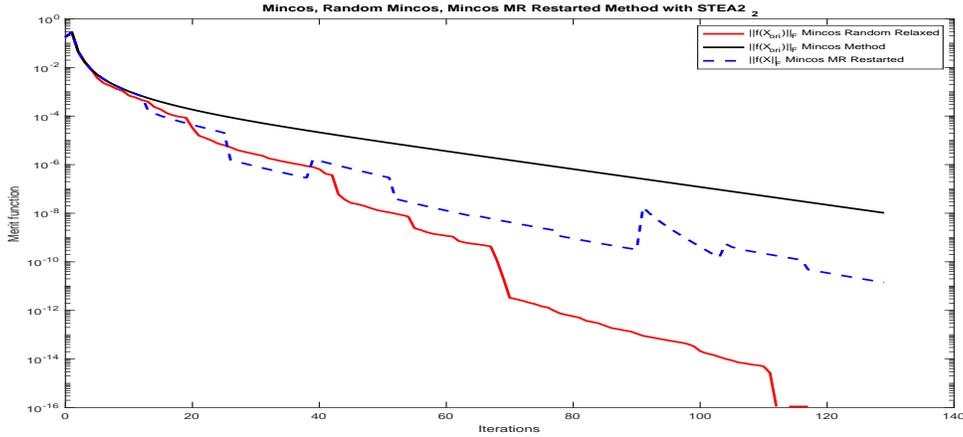


Figure 2: Convergence history of MinCos, Random Mincos and STEA2 for the 3D Poisson matrix for $n = 1000$, NCYLCE=10, and MCOL=12.

4, in which the Random Mincos shows a better acceleration with a numerically trustable monotone behavior.

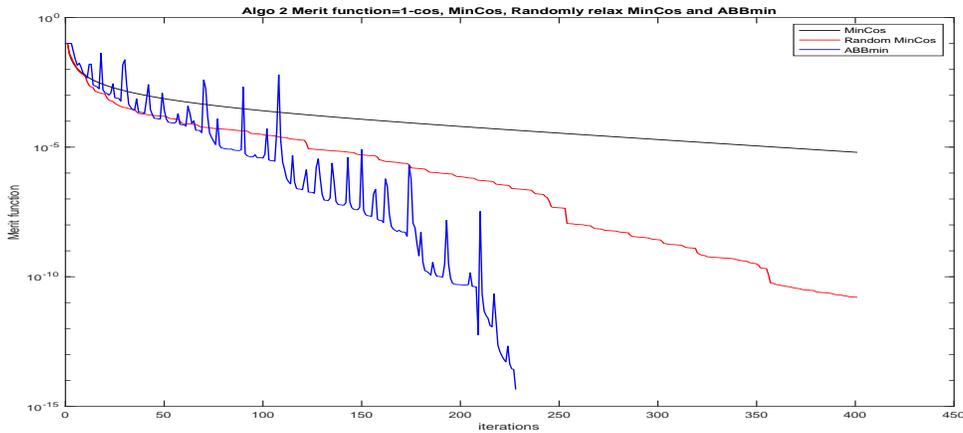


Figure 3: Convergence history of MinCos, Random Mincos and ABBmin for the 2D Poisson matrix for $n = 900$ and $\epsilon = 10^{-14}$, and maxiter=400.

For our next experiments we use the Wathen matrix and the 2D Poisson matrix, both from the Matlab gallery, with different large dimensions, and we compare the convergence history of the MinCos method and the Random Mincos acceleration. In Figures 5 and 6 we can observe the significant acceleration obtained by the Random Mincos for Wathen(30) of size $n = 2821$ ($\epsilon = 10^{-6}$, maxiter=900), and Wathen(50) of size $n = 7701$ ($\epsilon = 10^{-8}$, maxiter=300), respectively. As a consequence we note

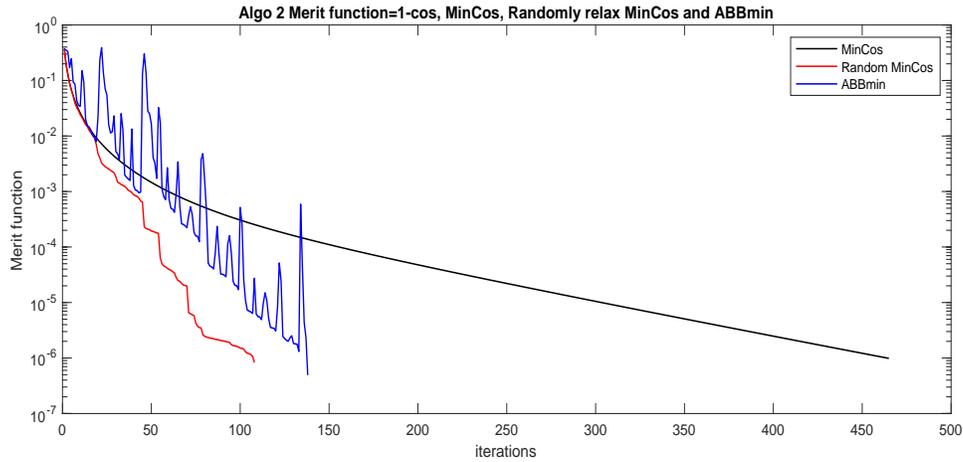


Figure 4: Convergence history of MinCos, Random Mincos and ABBmin for the Lehmer matrix for $n = 20$, $\epsilon = 10^{-6}$, and $\text{maxiter}=450$.

that the Random Mincos scheme exhibits in both cases a clear reduction in the required number of iterations when compared with the MinCos method. Let us recall that since the inverse of these matrices is dense, we are dealing with n^2 unknowns for all the considered problems, hence in the specific case of the wathen matrix ($n = 7701$) it is a very large number of variables. Similarly, in Figure 7 we can notice the clear acceleration and reduction in number of iterations of the Random Mincos acceleration for large-scale problems, as compared to the MinCos method.

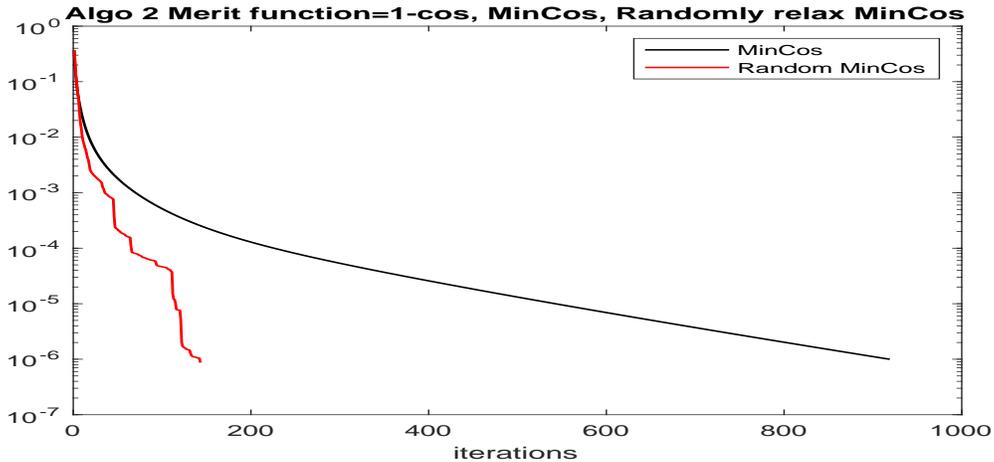


Figure 5: Convergence history of MinCos and the Random MinCos for the Whaten(30) matrix for $n = 2821$ ($3 \times 30^2 + 4 \times 30 + 1$), $\epsilon = 10^{-6}$, and $\text{maxiter}=1000$.

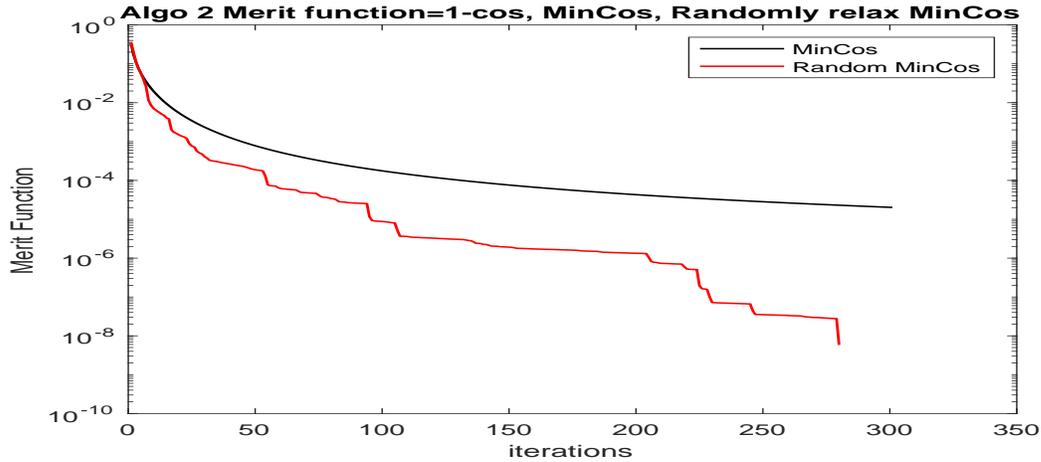


Figure 6: Convergence history of MinCos and the Random MinCos for the Whaten(50) matrix for $n = 7701$ ($3 \times 50^2 + 4 \times 50 + 1$), $\epsilon = 10^{-8}$, and $\text{maxiter}=300$.

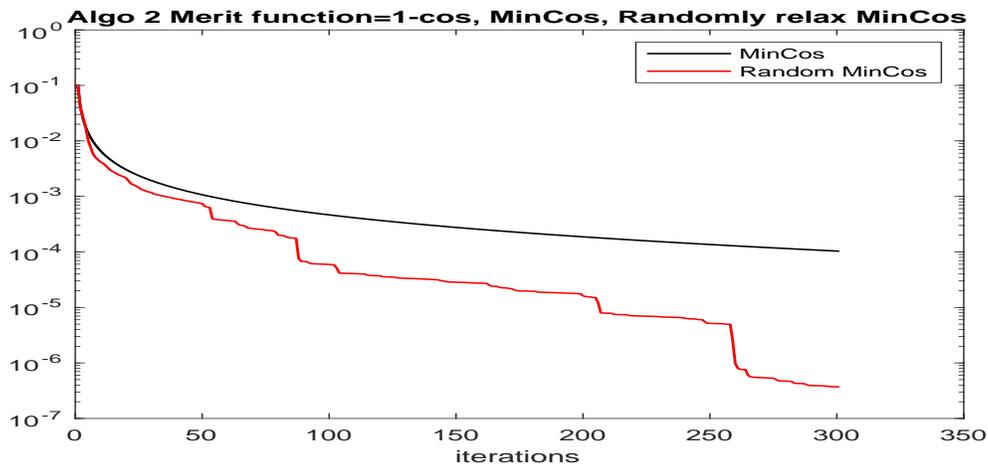


Figure 7: Convergence history of MinCos and the Random MinCos for the 2D Poisson matrix for $n = 3969$, $\epsilon = 10^{-7}$, and $\text{maxiter}=300$.

5.2 Rectangular matrices for Least-square problems using Algorithm 2

We will now consider rectangular matrices A from the Matlab gallery and also from the Matrixmarket [29] (reported in Table 1), and apply Algorithm 2. We note that in all these experiments, the matrix $A^T A$ is very ill-conditioned. In figures 8 and 9 we report the convergence behavior of the MinCos method (Algorithm 2), the Random Mincos acceleration, and the STEA2 acceleration for different values of NCYLCE and MCOL, when applied to the Matlab random normal matrices (seed=1) for $n = 100$ ($m = 80$, MAXCOL=8, and NCYCLE=30) and $n = 200$ ($m = 160$, MAXCOL=10, and

NCYCLE=25), respectively. We notice, in Figure 8 that the two acceleration schemes show a much better performance as compared to the MinCos method, requiring significantly less iterations to achieve the same accuracy. We can also notice in Figure 9 that when the size of the matrix increases, as well as the condition number, then the Random Mincos acceleration clearly outperforms the STEA2 scheme.

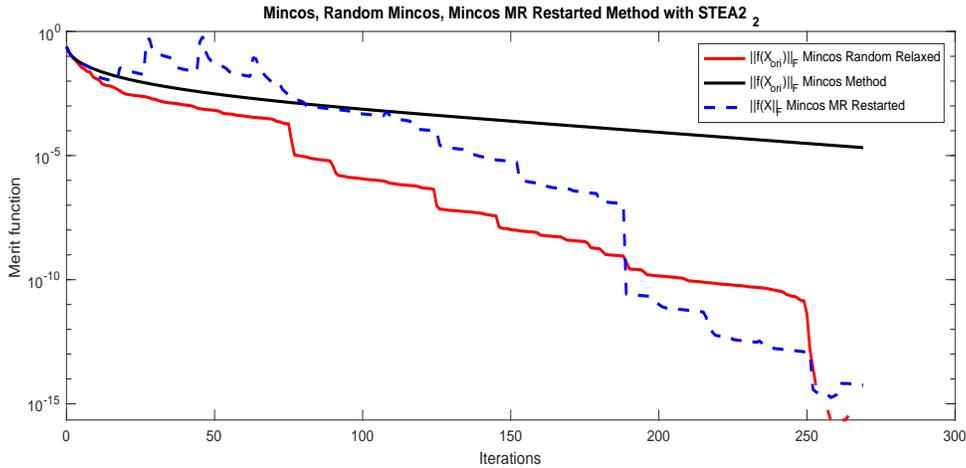


Figure 8: Convergence history of MinCos, Random Mincos and STEA2 for the random normal matrix for $n = 100$, $m = 80$, NCYLCE=30, and MCOL=8.

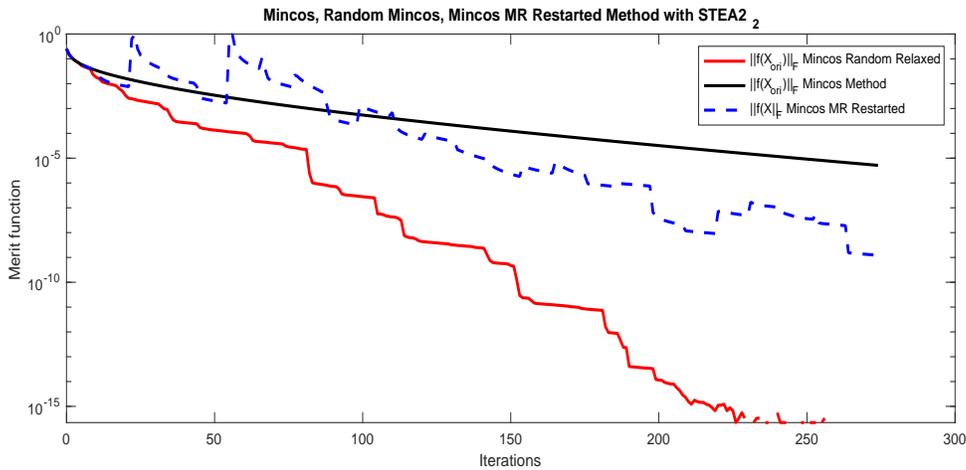


Figure 9: Convergence history of MinCos, Random Mincos and STEA2 for the random normal matrix for $n = 200$, $m = 160$, NCYLCE=25, and MCOL=10.

Next we compare the performance of the MinCos method, the Random Mincos acceleration and the ABBmin acceleration on the matrix well1850. In Figure 10 we notice that the ABBmin scheme shows a similar nonmonotone acceleration as before, up to an accuracy of 10^{-2} , and after that it

becomes unstable and cannot reach the required precision. In sharp contrast, the Random Mincos acceleration represents a trustable option that clearly outperforms the MinCos method.

| Matrix | Size (m, n) | $Cond(A^T A)$ |
|----------|---------------|---------------|
| illc1850 | (1850,712) | 1.4033e+07 |
| Well1850 | (1850,712) | 1.2309e+05 |

Table 1: Rectangular Matrices form Matrix market

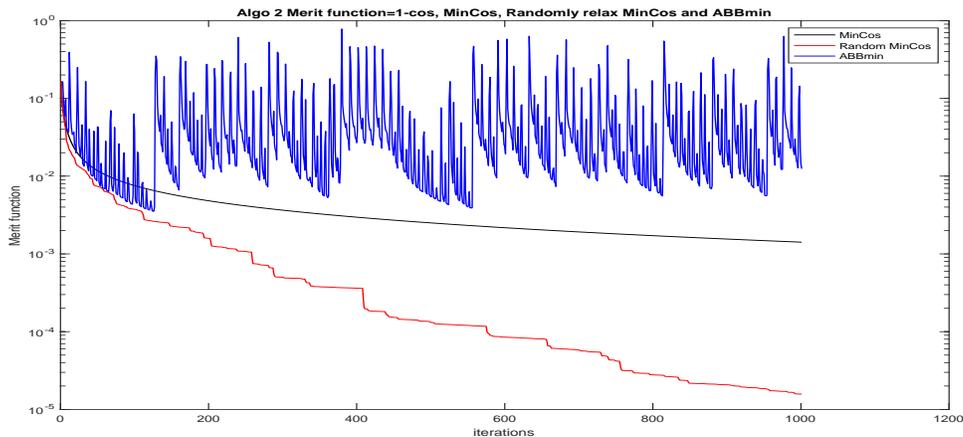


Figure 10: Convergence history of MinCos, the Random MinCos, and ABBmin for the matrix well1850, with $\epsilon = 10^{-5}$ and maxiter=1000.

In Figure 11 we report the convergence history of the MinCos method and the random Mincos acceleration for the harder illc1850 matrix. We note that the Random Mincos scheme shows a significant acceleration, and so it needs less iterations than the MinCos method to achieve the same accuracy.

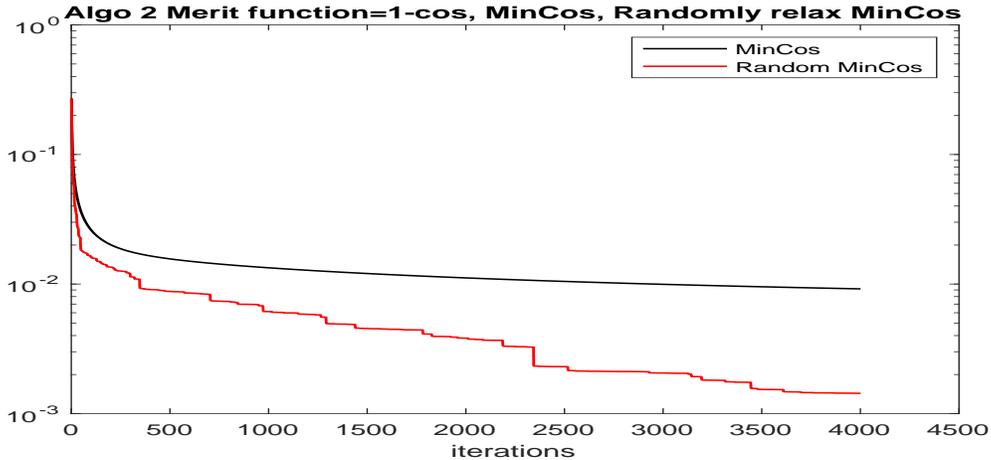


Figure 11: Convergence history of MinCos and the Random MinCos for the matrix illc1850, with $\epsilon = 10^{-3}$ and maxiter=4000.

6 Conclusions

We have extended the MinCos iterative method, originally developed in [12] for symmetric and positive definite matrices, to approximate the inverse of the matrices associated with linear least-squares problems, and we have also described and adapted three different possible acceleration schemes to the generated convergent matrix sequences.

Our experiments have shown that the geometrical MinCos scheme is also a robust option to approximate the inverse of matrices of the form $A^T A$, associated with least-squares problems, without requiring the explicit knowledge of A^T . They also show that both schemes (Algorithms 1 and 2) can be significantly accelerated by the three discussed strategies. In particular, the STEA2 scheme, from the simplified topological ϵ -algorithms family, and the Random MinCos scheme are clearly the most effective and most stable options. Our conclusion is that for small to medium size problems which are not ill-conditioned, the STEA2 scheme represents a good option with a strong mathematical support. For large-scale and ill-conditioned problems, our conclusion is that the inexpensive and numerically trustable Random MinCos acceleration is the method of choice.

An interesting application of inverse approximation techniques is the development of preconditioning strategies, for which in many real problems a sparse approximation is required. In that case, a suitable dropping or filtering strategy can be adapted, as it was done and extensively discussed in [12]. Therefore, the MinCos method has been already combined with dropping strategies showing a convenient performance to obtain sparse inverse approximations. Finally, we note that in our results we have compared the different schemes using $\hat{F}(X) = 1 - \cos(X(A^T A), I)$ as the merit function. Similar results can also be obtained using as the merit function the norm of the residual, i.e., $\|(I - X(A^T A))\|_F$, as it was already reported for the MinCos method for symmetric and positive definite matrices in [12].

Acknowledgments. The second author was supported by CNRS throughout a 3 months stay *Poste*

Rouge, in the LAMFA Laboratory (UMR 7352) at Université de Picardie Jules Verne, Amiens, France, from february to april, 2019.

References

- [1] R. Andreani, M. Raydan and P. Tarazaga [2013], On the geometrical structure of symmetric matrices, *Linear Algebra and its Applications*, 438, 1201–1214.
- [2] C. Brezinski [2000], Convergence acceleration during the 20th century, *J. Comput. Appl. Math.*, 122, 1–21. Numerical Analysis in the 20th Century Vol. II: Interpolation and Extrapolation.
- [3] C. Brezinski and M. Redivo-Zaglia [1991], *Extrapolation Methods. Theory and Practice*, North-Holland, Amsterdam.
- [4] C. Brezinski and M. Redivo-Zaglia [2014], The simplified topological ε -algorithms for accelerating sequences in a vector space, *SIAM J. Sci. Comput.*, 36, 2227–2247.
- [5] C. Brezinski and M. Redivo-Zaglia [2017], The simplified topological ε -algorithms: software and applications, *Numer. Algor.*, 74, 1237–1260.
- [6] C. Brezinski and M. Redivo-Zaglia [2019], The genesis and early developments of Aitkens process, Shanks transformation, the ε -algorithm, and related fixed point methods, *Numer. Algor.*, 84, 11–133.
- [7] C. Brezinski, M. Redivo-Zaglia, Y. Saad [2018], Shanks sequence transformations and Anderson acceleration, *SIAM Rev.*, 60, 646–669.
- [8] L. E. Carr, C. F. Borges, and F. X. Giraldo [2012], An element based spectrally optimized approximate inverse preconditioner for the Euler equations, *SIAM J. Sci. Comput.*, 34, 392–420.
- [9] J.-P. Chehab [2007], Matrix differential equations and inverse preconditioners, *Computational and Applied Mathematics*, 26, 95–128.
- [10] J.-P. Chehab [2016], Sparse approximations of matrix functions via numerical integration of ODEs, *Bull. Comput. Appl. Math.*, 4, 95–132.
- [11] J.P. Chehab and M. Raydan [2008], Geometrical properties of the Frobenius condition number for positive definite matrices, *Linear Algebra and its Applications*, 429, 2089–2097.
- [12] J.P. Chehab and M. Raydan [2016], Geometrical inverse preconditioning for symmetric positive definite matrices, *Mathematics*, 4(3), 46, doi:10.3390/math4030046.
- [13] K. Chen [2001], An analysis of sparse approximate inverse preconditioners for boundary integral equations, *SIAM J. Matrix Anal. Appl.*, 22, 1058–1078.
- [14] E. Chow and Y. Saad [1997], Approximate inverse techniques for block-partitioned matrices, *SIAM J. Sci. Comput.*, 18, 1657–1675.

- [15] J. Chung, M. Chung, and D. P. O’Leary [2015], Optimal regularized low rank inverse approximation, *Linear Algebra and its Applications*, 468, 260–269.
- [16] X. Cui and K. Hayami [2009], Generalized approximate inverse preconditioners for least squares problems, *Japan J. Indust. Appl. Math.*, 26, 1–14.
- [17] R. De Asmundis, D. di Serafino, F. Riccio, and G. Toraldo [2013], On spectral properties of steepest descent methods, *IMA J. Numer. Anal.*, 33, 1416–1435.
- [18] J. P. Delahaye and B. Germain-Bonne [1980], Résultats négatifs en accélération de la convergence, *Numer. Math.*, 35, 443–457.
- [19] K. Forsman, W. Gropp, L. Kettunen, D. Levine, and J. Salonen [1995], Solution of dense systems of linear equations arising from integral equation formulations, *Antennas and Propagation Magazine*, 37, 96–100.
- [20] G. Frassoldati, L. Zanni, and G. Zanghirati [2008], New adaptive stepsize selections in gradient methods, *J. Ind. Manag. Optim.*, 4, 299–312.
- [21] W. Glunt and T. L. Hayden and M. Raydan [1993], Molecular Conformations from Distance Matrices, *Journal of Computational Chemistry*, 14, 114–120.
- [22] N. I. M. Gould and J. A. Scott [1998], Sparse approximate-inverse preconditioners using norm-minimization techniques, *SIAM J. Sci. Comput.*, 19, 605–625.
- [23] P. R. Graves-Morris, P. R. Roberts, and A. Salam [2000], The epsilon algorithm and related topics, *J. Comput. Appl. Math.*, 122, 51–80.
- [24] J. Helsing [2006], Approximate inverse preconditioners for some large dense random electrostatic interaction matrices, *BIT Numerical Mathematics*, 46, 307–323.
- [25] K. Jbilou and A. Messaoudi [2016], Block extrapolation methods with applications, *Applied Numerical Mathematics*, 106, 154–164.
- [26] K. Jbilou and H. Sadok [2000], Vector extrapolation methods: applications and numerical comparison, *J. Comput. Appl. Math.*, 122, 149–165.
- [27] K. Jbilou and H. Sadok [2015], Matrix polynomial and epsilon-type extrapolation methods with applications, *Numer. Algor.*, 68, 107–119.
- [28] A. C. Matos [1992], Convergence and acceleration properties for the vector ϵ -algorithm, *Numer. Algor.* 3, 313–320.
- [29] Matrix Market, <http://math.nist.gov/MatrixMarket/>
- [30] G. Montero, L. González, E. Flórez, M. D. García, and A. Suárez [2002], Approximate inverse computation using Frobenius inner product, *Numerical Linear Algebra with Applications*, 9, 239–247.

- [31] M. Raydan and B. Svaiter [2002], Relaxed steepest descent and Cauchy-Barzilai-Borwein method, *Computational Optimization and Applications*, 21, 155–167.
- [32] A. M. Sajo-Castelli, M. A. Fortes, and M. Raydan [2014], Preconditioned conjugate gradient method for finding minimal energy surfaces on Powell-Sabin triangulations, *Journal of Computational and Applied Mathematics*, 268, 34–55.
- [33] D. di Serafino, V. Ruggiero, G. Toraldo, and L. Zanni [2018], On the steplength selection in gradient methods for unconstrained optimization, *Applied Mathematics and Computation*, 318, 176–195.
- [34] R. B. Sidje and Y. Saad [2011], Rational approximation to the Fermi–Dirac function with applications in density functional theory, *Numer. Algor.*, 56, 455–479.
- [35] J.-K. Wang and S.D. Lin [2014], Robust inverse covariance estimation under noisy measurements, in *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, 928–936.
- [36] B. Zhou, L. Gao, and Y. H. Dai [2006], Gradient methods with adaptive step-sizes, *Comput. Optim. Appl.*, 35, 69–86.