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

Jean-Paul Chehab\textsuperscript{1} · Marcos Raydan\textsuperscript{2}

Received: 21 February 2019 / Accepted: 3 December 2019 / Published online: 8 January 2020
© Springer Science+Business Media, LLC, part of Springer Nature 2020

Abstract
We extend the geometrical inverse approximation approach to the linear least-squares scenario. For that, we focus on the minimization of $1 - \cos(X(A^T A), I)$, where $A$ is a full-rank matrix of size $m \times n$, with $m \geq n$, and $X$ is an approximation of the inverse of $A^T A$. In particular, we adapt the recently published simplified gradient-type iterative scheme MinCos to the least-squares problem. 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 line search acceleration schemes which are based on selecting an appropriate steplength at each iteration. A set of numerical experiments, including large-scale problems, are presented to illustrate the performance of the different accelerations strategies.

Keywords Inverse approximation · Matrix acceleration techniques · Gradient-type methods

1 Introduction
The development of inverse matrix approximation strategies for the 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, sparse, or dense matrices; see, e.g., [9–11, 14–16, 21, 26, 35, 37, 39, 41].
In this work, for a given real rectangular \(m \times n\) \((m \geq n)\) full-rank matrix \(A\), we will obtain inverse approximations of \(A^T A\) based on minimizing the positive-scaling-invariant function

\[
\hat{F}(X) = 1 - \cos(X(A^T A), I) = 1 - \frac{\langle X(A^T A), I \rangle}{\|X(A^T A)\|_F \|I\|_F}
\]

on a suitable closed and bounded subset of the positive semi-definite (PSD) cone of square matrices which has a rich geometrical structure; see, e.g., [2, 12]. 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., [17, 24]. In here, \(I\) 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.

For the minimization of \(\hat{F}(X)\), we will extend and adapt the simplified gradient-type scheme MinCos, introduced in [13], to the linear least-squares scenario. Moreover, we will adapt and apply some 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 \(\varepsilon\)-algorithms [5, 6], and also on the extension of randomly chosen steplength acceleration strategies [36], as well as the extension of some recent nonmonotone gradient-type choices of steplenghts [22, 42]. It is worth noticing that these acceleration strategies are quite different, in the sense that the simplified topological \(\varepsilon\)-algorithms are extrapolation schemes, whereas the other two are based on line search strategies which are usually applied for solving numerical optimization problems. Nevertheless, it is also interesting to observe that both families of accelerations can be effectively applied and combined to solve nonlinear matrix problems.

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 a version of the MinCos method to approximate the inverse of \(A^T A\). In Section 4, we describe and analyze different acceleration strategies aimed at obtaining a faster convergence of the sequence generated by the new version of the MinCos method. 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.
This method has been successfully introduced in [13], and can be seen as an improved version of the Cauchy method applied to the minimization of the merit function as follows:

\[
F(X) = 1 - \frac{1}{\|X\|_F} \cdot \frac{\|I\|_F}{\|I\|_F}.
\]

In Remark 2.1, we summarize the most important properties of Algorithm 2 (see [13]).

**Remark 2.1**

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

2. By construction \(\|X(k)\|_F = \sqrt{n}\), for all \(k \geq 1\). If we choose \(X(0)\) such that \(\text{trace}(X(0)A) = \langle X(0), I \rangle > 0\) 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 \(\tilde{D}_k\) 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 \(\tilde{D}_k\). Furthermore, \(Z(k)\), \(X(k)\), and \(X(k)A\) in the MinCos Algorithm are symmetric matrices for all \(k\), whose Frobenius norms are uniformly bounded away from zero (([13, Lemma 2.10]), and so the algorithm is well-defined.

For completeness, we state the convergence result concerning the MinCos method (for the proof see [13]).

**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 may 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 in 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

$$
\langle W_1, W_2 W_3 \rangle_F = \langle W_2^T W_1, W_3 \rangle_F = \langle W_1 W_3^T, W_2 \rangle_F.
$$

(1)

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

$$
\langle YA^T A, I \rangle = \langle (AY)^T A, I \rangle = \langle A, AY^T \rangle = \langle AY^T, A \rangle.
$$

(2)

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

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

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

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

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

We note that for any matrix $A$, $A^T A$ is in the PSD cone, and so all the results presented in Section 2 apply to Algorithm 2, in particular, since $A$ is full column rank, then by Theorem 2.1, the sequence $\{X(k)\}$ converges to $(A^T A)^{-1}$. 

© Springer
4 Acceleration strategies

The sequence of matrices $\{X^{(k)}\} \subset IR^{n \times n}$ generated either by Algorithm 1 or by Algorithm 2 can be viewed as simplified and improved versions of the CauchyCos algorithm developed in [13], 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., [3, 4].

Simplified topological $\varepsilon$-algorithms (STEA)

For our first acceleration strategy, we will focus on the matrix version of the so-called simplified topological $\varepsilon$-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., [5, 8, 25, 30–33]. For a full historical review on this topic as well as some other related issues we recommend [7]. To use the simplified topological $\varepsilon$-algorithms, we take advantage of the recently published Matlab package EPSfun$^1$ [6], 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 $\varepsilon$-algorithms 1 (STEA1) and the simplified topological $\varepsilon$-algorithms 2 (STEA2) using the restarted method (RM) option, which have been implemented in the EPSfun package, including four possible variants.

Random MinCos method

For our second acceleration strategy, we will adapt a procedure proposed and analyzed in [36] for the minimization of convex quadratics, and that can be viewed as a member of the line search family for which the acceleration is generated by the method itself, i.e., at once and dynamically only one accelerated sequence is generated; see also [11]. 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 [19]. For our specific algorithms, the second strategy is obtained by relaxing the optimal descent parameter $\alpha_k$ as follows:

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

where $\theta_k$ is at each step randomly chosen in $(a, b)$, preferably $(a, b) = (1 - \eta, 1 + \eta)$ for some $0 < \eta < 1$, 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 [36]), let us define the following:

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

---

$^1$The Matlab package EPSfun is freely available at http://www.netlib.org/numeralgo/
where $\hat{D}_k$ is the descent direction and $\alpha_k$ the optimal parameter given in Algorithm 1. Notice that all the following results concerning $F$ and Algorithm 1 also apply automatically to $\hat{F}$ 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 $\hat{D}_k$ is a descent direction. Now $\phi_k(1) < 0$ because $\alpha_k$ minimizes $F(X^{(k)} + \alpha \hat{D}_k)$.

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 as follows:

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

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

$$\| X^{(k)}A \|^2_F (\langle X^{(k)}A, I \rangle + t \alpha_k \langle \hat{D}_k A, I \rangle)^2 = \langle X^{(k)}A, I \rangle^2 (\| X^{(k)}A \|^2_F + 2 \alpha_k t \langle X^{(k)}A, \hat{D}_k A \rangle + \alpha_k^2 t^2 \| \hat{D}_k A \|^2_F),$$

and we obtain the following:

$$t^2 \alpha_k^2 (\| X^{(k)}A \|^2_F \langle \hat{D}_k A, I \rangle^2 - \langle X^{(k)}A, I \rangle^2 \| \hat{D}_k A \|^2_F) + 2 t \alpha_k \langle \hat{D}_k A, I \rangle \| X^{(k)}A \|^2_F - \langle X^{(k)}A, \hat{D}_k A \rangle \langle X^{(k)}A, I \rangle^2 = 0.$$

Now, dividing by $\alpha_k t \neq 0$ and using $\| X^{(k)}A \|^2_F = \sqrt{n}$, it follows that

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

and the result is established.

At each iteration $k$, we can compute $t^*_k$ using (4), and then relax $\alpha_k$ using (3), where $\theta_k$ is chosen uniformly random in $(0, t^*_k)$. By the definition of $\phi_k(t)$ and Propositions 4.1 and 4.2, it follows that $F(X^{(k+1)}) < F(X^{(k)})$, which is a desired result but not enough to guarantee that the combined scheme has the same global convergence of the original MinCos method. For that, we adapt and apply the permissive line search backtracking globalization strategy, developed in [20]. The idea is to obtain the next iterate as $X^{(k+1)} = X^{(k)} + \tau_k \alpha_k \hat{D}_k$, where $\tau_k \in (0, 1]$ guarantees a sufficient reduction in the objective function value. Recalling from [13] that $\nabla F(X^{(k)}) = -\hat{D}_k A$, the acceptance condition is given by the following:

$$F(X^{(k)} + \tau_k \alpha_k \hat{D}_k) \leq \max_{0 \leq j \leq \min \{k, M\}} F(X^{(k-j)}) - \gamma \alpha_k \tau_k^2 \langle \hat{D}_k A, \hat{D}_k \rangle + \eta_k,$$

where $\hat{M} \geq 0$ is a fixed integer, $0 < \gamma < 1$ is a small number, $\alpha_k$ is the relaxed steplength given by (3), and $\tau_k > 0$ is obtained after a backtracking process that
must start at \( \tau_k = 1 \) to promote the acceptance of the randomly relaxed steplength. Notice that for the Random MinCos method, since \( F(X^{(k+1)}) < F(X^{(k)}) \) for all \( k \), we can set \( \hat{M} = 0 \), and the first term on the right-hand side of (5) is reduced to \( F(X^{(k)}) \). At each backtracking step, we reduce \( \tau_k = \rho \tau_k \), where \( \rho \in (0, 1) \), until (5) is satisfied. The parameter \( \rho \in (0, 1) \) can be chosen differently at every step, but for our acceleration schemes, the conservative fixed choice \( \rho = 1/2 \) works effectively.

To take advantage of the convergence results in [20], the sequence \( \eta_k > 0 \) must be chosen such that:

\[
0 < \sum_k \eta_k = \eta < \infty. \tag{6}
\]

**Theorem 4.1** The Random MinCos iterations are well-defined, and the sequence \( \{X^{(k)}\} \) that the method generates converges to \( A^{-1} \).

**Proof** For every \( k \), \( \hat{D}_k \) is a descent direction and \( \eta_k > 0 \), so the number of cycles at the backtracking process will be finite, until a sufficiently small \( \tau_k > 0 \) is obtained such that (5) is satisfied. As a consequence, the iterations are well-defined.

Concerning the backtracking parameter \( \tau_k \), our search direction is given by \( \alpha_k \hat{D}_k \), and hence the line search strategy (5), imposed at every iteration, can be seen as a special case of the one used in the model algorithm in [20]. As a consequence, from Theorem 1 in [20], we have that for any convergent subsequence \( \{X^{(k)}\}_{k \in K} \) \((K \subseteq \mathbb{N})\), \( \lim_{k \to \infty} \langle \nabla F(X^{(k)}), \hat{D}_k \rangle \geq 0 \), which implies that

\[
\lim_{k \to \infty} \langle \hat{D}_k A, \hat{D}_k \rangle \leq 0. \tag{7}
\]

On the other hand, since \( A \) is symmetric and positive definite, it has a square root \( A^{1/2} \) which is also symmetric and positive definite, and so for all \( k \),

\[
\langle \hat{D}_k A, \hat{D}_k \rangle = \langle \hat{D}_k A^{1/2}, \hat{D}_k A^{1/2} \rangle = \| \hat{D}_k A^{1/2} \|_F \geq 0. \tag{8}
\]

Combining (7) and (8), we obtain that \( \lim_{k \to \infty} \| \hat{D}_k A^{1/2} \|_F = 0 \). Therefore,

\[
\lim_{k \to \infty} \nabla F(X^{(k)}) = \lim_{k \to \infty} \langle \hat{D}_k A^{1/2} \rangle A^{1/2} = 0,
\]

which implies that \( \lim_{k \to \infty} X^{(k)} = A^{-1} \) (see the proof of Theorem 2.1 in [13]), and by continuity that \( \lim_{k \to \infty} F(X^{(k)}) = 0 \). Now, the entire sequence \( \{X^{(k)}\} \) belongs to a closed and bounded set (see [13, Section 2] for details); hence, there exist limit points in that set. Moreover, since \( F \) is bounded below and the whole sequence \( F(X^{(k)}) \) is decreasing, then the whole sequences \( F(X^{(k)}) \) and \( \nabla F(X^{(k)}) \) converge to zero. Once again, from the proof of Theorem 2.1 in [13], we have that \( \nabla F(\hat{X}) = 0 \) if and only if \( \hat{X} = A^{-1} \), i.e., the only possible stationary point is \( A^{-1} \). Therefore, by continuity the whole sequence \( X^{(k)} \) converges to \( A^{-1} \).

In practice, the parameters associated with the line search strategy are chosen to reduce the number of backtracking steps as much as possible, i.e., to accept the first relaxed steplength given by (3) as frequently as possible, while keeping the convergence properties of the method. In particular, we set \( \gamma = 10^{-4} \), the parameter \( \eta_0 \) is
chosen as a large number, e.g., $\eta_0 = 10 \times F(X^{(0)})$, and then it is reduced as slow as possible making sure that (6) holds; e.g., $\eta_k = \eta_0 / k^{1.1}$.

**Remark 4.1** Notice that the computation of $t^*_k$ is obtained for free: all the terms defining $t^*_k$ in (4) have been previously computed to obtain $\alpha_k$. Moreover, in all our experiments and for all $k$, $t^*_k \in (1, 2)$. Nevertheless, as illustrated in our next section, the uniformly random choice $\theta_k \simeq U([1/2, 3/2])$ combined with the permissive line search strategy (5) is a robust and efficient practical option whose global convergence is guaranteed by Theorem 4.1.

**ABBmin method**

Since $\hat{D}_k$ 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 [22, 42], which has proved to be very effective in the solution of general nonlinear unconstrained optimization problems [22, 38]. It is worth mentioning that the ABBmin method is a nonmonotone scheme for which convergence has been established for strictly convex quadratics [38]. As in the case of the randomly relaxed acceleration, this approach can also be viewed as a member of the line search 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 $\alpha_k$ by the following:

$$\hat{\alpha}_k = \begin{cases} \min\{\alpha_{BB2}^j : j = \max\{1, k - M\}, \ldots, k\}, & \text{if } \alpha_k^{BB2}/\alpha_k^{BB1} < \tilde{\tau}; \\ \alpha_k^{BB1}, & \text{otherwise} \end{cases} (9)$$

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

$$\alpha_k^{BB1} = \max\{\alpha_{\min}, \min\left\{\frac{\|S^{(k-1)}\|_F^2}{\langle S^{(k-1)}, Y^{(k-1)} \rangle}, \alpha_{\max}\right\}\}$$

and

$$\alpha_j^{BB2} = \max\{\alpha_{\min}, \min\left\{\frac{\langle S^{(j-1)}, Y^{(j-1)} \rangle}{\|Y^{(j-1)}\|_F^2}, \alpha_{\max}\right\}\}$$

for all $1 \leq j \leq k$, where $S^{(j-1)} = X^{(j)} - X^{(j-1)}$ and $Y^{(j-1)} = \hat{D}_j - \hat{D}_{j-1}$, for all $j$. The safety parameters $0 < \alpha_{\min} \ll \alpha_{\max}$ are chosen to keep the steplengths in a large closed and bounded positive interval.

A geometrical as well as an algebraic motivation for the choice $\hat{\alpha}_k$ in (9) can be found in [22, 38, 42]. In particular, they establish an interesting connection between $\alpha_k^{BB1}, \alpha_k^{BB2}$, 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 [23, pp. 117-118]; see also [18, 36].
The function $F$ is not a quadratic function, and so the convergence of the ABB-min method is not guaranteed. In order to establish a global convergence result, we apply the same permissive line search backtracking globalization strategy given by (5), using $\hat{\alpha}_k$ given by (9) instead of $\alpha_k$ given by (3), and now choosing $\hat{M} \geq 1$ to allow its nonmonotone behavior, for example, $\hat{M} > 5$ is a recommendable practical choice. Notice that the first and third terms on the right-hand side of (5) are responsible for the sufficiently nonmonotone behavior of the sequence $\{F(X^{(k)})\}$. The other parameters in (5) are chosen as before, and the backtracking procedure, reducing $\tau_k > 0$ until (5) holds, is identical to the one described above for the Random MinCos method.

**Theorem 4.2** The ABBmin iterations are well-defined, and the method generates a subsequence that converges to $A^{-1}$.

**Proof** For every $k$, $\hat{D}_k$ is the same descent direction used for the Random MinCos method, and so repeating the same arguments shown in the first part of the proof of Theorem 4.1, it follows that the iterations are well-defined. Moreover, the entire sequence $\{X^{(k)}\}$ belongs to a closed and bounded set (see [13, Section 2] for details); hence, there exist limit points in that set. The proof follows identical to the second part of the proof in Theorem 4.1, up to the point in which it is established that for any limit point, say $\hat{X}$, we have that $\nabla F(\hat{X}) = 0$ and so $\hat{X} = A^{-1}$.

**Remark 4.2** Since the function values $F(X^{(k)})$ in the ABBmin method do not have a monotonic behavior as in the Random MinCos method, the convergence result in Theorem 4.2 is weaker than the one in 4.1, in the sense that in Theorem 4.2 only the convergence of a subsequence to the solution $A^{-1}$ is established.

**5 Illustrative numerical examples**

To give further insight into the behavior of the MinCos method for least-squares problems, and the described acceleration strategies, we present the results of some numerical experiments. Taking advantage of the fact that the STEA family represents extrapolation algorithms, in addition to the three discussed strategies in Section 4 (STEA, Random MinCos, and ABBmin), we consider the application of the STEA acceleration schemes not only to the sequence generated by the MinCos method but also to the sequence generated by the Random MinCos monotone line search acceleration.

All computations were performed in Matlab 7.3 (R2019a), using double precision, which has unit roundoff $\mu \approx 1.1 \times 10^{-16}$. All the runs were carried out on an Intel Core i5-2450M at 2.7 GHz with 16 GB of RAM. Our initial guess is chosen a $X^{(0)} = \beta I$, 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. To take advantage of the adapted formulas obtained in Section 3, which are based mainly on (2), it is important to maintain the symmetry of all the iterates.
in Algorithm 2. For that, in our implementation, \( Z^{(k+1)} \) at Step 6 is actually obtained as the closest symmetric matrix to the original one, as follows:

6 : Set \( Z_{\text{temp}} = X^{(k)} + \alpha_k \hat{D}_k \) and \( Z^{(k+1)} = (Z_{\text{temp}} + Z_{\text{temp}}^T)/2 \).

We note that this additional calculation represents a negligible computational cost as compared to the other steps in Algorithm 2, but at the same time it represents a safety procedure to avoid the numerical loss of symmetry that we have observed in some experiments. We stop the process when the merit function \( F(X^{(k)}) \) (Algorithm 1) or \( \tilde{F}(X^{(k)}) \) (Algorithm 2) is less than or equal to \( \epsilon = 0.5 \times 10^{-15} \) or when a given number of maximum iterations (MaxIt) is reached. We set a different value of MaxIt for each experiment depending on the characteristics of the involved matrix (dimension, sparsity, and condition number) which affect the required CPU time. Once we stop the process, for each method, we report the number of iterations (Iter), the required CPU time in seconds (Time), and the value of the merit function (Residual).

Concerning the package EPSfun, we use the STEA2 option which has proved to be more effective than STEA1 in all our experiments, with different choices of the key parameters MAXCOL and NCY. The details of all the options that can be used in the EPSfun package are fully described in [6]. For the Random MinCos acceleration, the parameters \( \theta_k \) is randomly chosen in the interval \([1/2, 3/2]\) as indicated in Remark 4.1, and \( \tilde{M} = 0 \). Concerning the ABBmin method, we set \( \bar{r} = 0.8 \), \( \alpha_{\min} = 10^{-10} \), \( \alpha_{\max} = 10^{10} \), and \( \tilde{M} = M = 10 \) in all cases. Summing up, we will report results with the following accelerations applied to the sequence generated by the MinCos method: Random MinCos (R-MinCos), ABBmin, STEA2 (STEA2-MinCos); and also with the STEA2 acceleration applied to the sequence generated by the Random MinCos scheme (STEA2-R-MinCos).

For our experiments, we consider a variety of representative test matrices \( A \) with different characteristics ranging from small to large-scale size, from well-conditioned to very ill-conditioned, and with different sparsity patterns. We have observed, based on many additional results, that the behavior of each one of the acceleration schemes is quite similar when they are applied to another test matrices sharing the same characteristics of the matrices that have been considered. To be precise, from the Matlab gallery we consider the square symmetric PSD matrices: Wathen, Lehmer, and Poisson, which can be obtained with the Matlab commands \( A = \text{gallery}('wathen', N, N) \), \( A = \text{gallery}('lehmer', n) \), and \( A = \text{gallery}('poisson', N) \), respectively. We note that Wathen is a sparse \( n \times n \) matrix where \( n = 3N^2 + 4N + 1 \), Lehmer is a nonnegative dense matrix of size \( n \times n \), and Poisson is the banded \( n \times n \) (where \( n = N^2 \)) finite differences 2D discretization matrix of the negative Laplacian on the unit square with homogeneous Dirichlet boundary conditions. From the Matrix Market [34], we consider the Well1850 rectangular matrix (1850 \( \times \) 712), for which the condition numbers of \( A^T A \) is \( 1.2e+5 \). In addition, we consider the rectangular matrix that has random entries with a Gaussian distribution and can be obtained with the Matlab command \( A = \text{randn('normal', 0, 1, size(rand(m, n)))} \); in the following, this matrix is called Normal. We note that the inverses of all the considered matrices are dense, except the inverse of the Lehmer matrix which is tridiagonal.
5.1 Results with Algorithm 1 on PSD matrices

For our first set of experiments, we consider the symmetric and banded positive definite Poisson matrix with $n = 900$ ($N = 30$), which has spectral condition number $\kappa(A) = 565$. In Tables 1, 2, and 3, and corresponding Figs. 1, 2, and 3, we report the performance and the convergence history of the MinCos method (Algorithm 1), and all the different accelerations for different values of MaxIt, MAXCOL, and NCY. We notice that in all cases Random MinCos, ABBmin, and STEA2 produce significant accelerations when compared with the sequence generated by the MinCos method. In addition, we note that ABBmin and Random MinCos require less iterations than STEA2 to achieve the same accuracy. Furthermore, ABBmin is slightly better than Random MinCos in terms of iterations and computational time. We also note that the STEA2-R-MinCos is better than MinCos in terms of reducing the value of the merit function but it is not a competitive option when compared with the other three acceleration schemes. Finally, we can observe that STEA2 and STEA2-R-MinCos improve their performance when the parameter MAXCOL is increased.

For our second experiment, we consider the symmetric and sparse positive definite Wathen matrix with $n = 2821$ ($N = 30$), for which $\kappa(A) \approx 1490$. In Table 4 and Fig. 4, we report the performance and the convergence history of the MinCos method (Algorithm 1), and all the different accelerations for MaxIt = 370, MAXCOL = 30, and NCY = 12. Our first observation is that this is a hard problem for which all the acceleration schemes could only attain, after 370 iterations, a merit function value of approximately 1.e-4, while the MinCos method achieves a function value of 2.4e-3. Let us recall that since the inverses of these matrices are dense, we are dealing with $n^2$ unknowns for all the considered problems; hence, in the specific case of the Wathen matrix, it is a very large number of variables. We note, in Table 4, that

| Method          | Iter | Residual | Time |
|-----------------|------|----------|------|
| MinCos          | 540  | 1.3e-6   | 72.9 |
| R-MinCos        | 319  | 9.9e-16  | 42.5 |
| ABBmin          | 245  | 7.7e-16  | 27.6 |
| STEA2           | 540  | 8.1e-12  | 97.0 |
| STEA2-R-MinCos  | 540  | 9.4e-9   | 109.9|

| Method          | Iter | Residual | Time |
|-----------------|------|----------|------|
| MinCos          | 510  | 1.8e-6   | 70.6 |
| R-MinCos        | 322  | 8.8e-16  | 42.9 |
| ABBmin          | 245  | 7.7e-16  | 26.1 |
| STEA2           | 507  | 1.1e-16  | 118.4|
| STEA2-R-MinCos  | 510  | 2.7e-9   | 119.4|
ABBmin requires less CPU time than all the other options to achieve the same accuracy. However, we also note that the ABBmin acceleration represents an aggressive option that shows a highly nonmonotone behavior. We can also observe that the other accelerations show a similar performance with a numerically trustable behavior.

For the next experiment, we consider the nonnegative dense symmetric and positive definite Lehmer matrix with \( n = 20 \), whose condition number is of the order of \( n^2 \). In Table 5, we report the performance of the MinCos method (Algorithm 1), and all the different accelerations for \( \text{MaxIt} = 370, \text{MAXCOL} = 30, \text{NCY} = 12 \). Our first observation is that the ABBmin scheme, which in the previous experiments produced a competitive acceleration, is not capable of producing any acceleration whatsoever, exhibiting an erratic nonmonotone behavior. For this small size matrix,
Fig. 2 Convergence history of the MinCos method and the different accelerations for the Poisson matrix with $n = 900$, MaxIt$= 510$, MAXCOL$= 16$, and NCY$= 30$

Fig. 3 Convergence history of the MinCos method and the different accelerations for the Poisson matrix with $n = 900$, MaxIt$= 500$, MAXCOL$= 30$, and NCY$= 16$
Table 4 Performance of the MinCos method and the different accelerations for the Wathen matrix with $n = 2821$, MaxIt$= 370$, MAXCOL$= 30$, and NCY$= 12$

| Method          | Iter | Residual | Time   |
|-----------------|------|----------|--------|
| MinCos          | 370  | 2.4e-3   | 1.28e+03 |
| R-MinCos        | 370  | 1.1e-4   | 1.25e+03 |
| ABBmin          | 370  | 1.1e-4   | 0.77e+03 |
| STEA2           | 370  | 7.1e-4   | 1.6e+03  |
| STEA2-R-MinCos  | 370  | 1.1e-4   | 1.8e+03  |

the required CPU time for all considered methods is less than the one observed in our previous experiments. Nevertheless, the STEA2 and STEA2-R-MinCos acceleration requires additional CPU time when compared with the other line search gradient-type schemes that only require the storage of the direction matrix $D_k$ and very low additional computational cost per iteration. We also note that the STEA2 schemes, as well as the Random MinCos, produce an effective acceleration when compared with the MinCos method.

Fig. 4 Convergence history of the MinCos method and the different accelerations for the Wathen matrix with $n = 2821$, MaxIt$= 370$, MAXCOL$= 30$, and NCY$= 12$
Table 5 Performance of the MinCos method and the different accelerations for the Lehmer matrix with \( n = 20 \), MaxIt= 370, MAXCOL= 30, and NCY= 12

| Method      | Iter | Residual   | Time |
|-------------|------|------------|------|
| MinCos      | 370  | 2.1e-6     | 0.05 |
| R-MinCos    | 340  | 3.3e-16    | 0.04 |
| ABBmin      | 370  | 3.9e-1     | 0.05 |
| STEA2       | 370  | 8.4e-10    | 4.1  |
| STEA2-R-MinCos | 370 | 5.5e-11    | 5.4  |

5.2 Results with Algorithm 2 on rectangular matrices

We now consider rectangular matrices \( A \) and apply Algorithm 2. In all the forthcoming experiments, the matrix \( A^T A \) is ill-conditioned. In Tables 6 and 7, we report the performance of the MinCos method (Algorithm 2), and all the different accelerations when applied to the dense rectangular Normal matrix (\( n = 200 \) and \( m = 160 \)) for MaxIt= 270, NCY= 12, and setting MAXCOL= 20 and MAXCOL= 26, respectively. First we note that, in both cases, the ABBmin scheme is not capable of producing an acceleration, exhibiting once again an erratic nonmonotone behavior, and as a consequence it cannot reach an acceptable accuracy. We can also observe that Random MinCos and STEA2 produce significant accelerations when compared with the sequence generated by the MinCos method. However, the Random MinCos scheme achieves better accuracy with less iterations and less computational work. We also note that STEA2-R-MinCos is a competitive acceleration strategy when the parameter MAXCOL is increased, at the cost of a slight increase in CPU time.

For our last experiment, we consider the harder rectangular Well1850 matrix (1850 × 712), from the Matrix Market [34], for which \( A^T A \) is very ill-conditioned. In Table 8, we report the performance of the MinCos method (Algorithm 2), and all the different accelerations for MaxIt= 1000, MAXCOL= 40, and NCY= 25. We note that, as in previous experiments when dealing with larger and ill-conditioned problems, the ABBmin scheme is not capable of producing an acceleration, showing an erratic highly nonmonotone behavior. Hence, in Fig. 5, we report the convergence history of all the considered methods, except ABBmin which is not included to be able to observe and appreciate the convergence behavior of the other schemes. We note that, for this ill-conditioned matrix, the Random MinCos scheme is the only option that produces an effective acceleration, although it reached the maximum
Table 7 Performance of the MinCos method and the different accelerations for the Normal matrix with $n = 200$, $m = 160$, MaxIt = 270, MAXCOL = 26, and NCY = 12

| Method          | Iter | Residual | Time |
|-----------------|------|----------|------|
| MinCos          | 270  | 5.6e-6   | 0.47 |
| R-MinCos        | 223  | 8.8e-16  | 0.38 |
| ABBmin          | 270  | 6.7e-2   | 0.44 |
| STEA2           | 270  | 1.3e-13  | 4.92 |
| STEA2-R-MinCos  | 270  | 7.5e-13  | 5.21 |

Table 8 Performance of the MinCos method and the different accelerations for the Well1850 matrix with $n = 1850$, $m = 712$, MaxIt = 1000, MAXCOL = 40, and NCY = 25

| Method          | Iter | Residual | Time |
|-----------------|------|----------|------|
| MinCos          | 1000 | 2.4e-3   | 121.14 |
| R-MinCos        | 1000 | 3.1e-8   | 130.8 |
| ABBmin          | 1000 | 1.6e-2   | 75.4 |
| STEA2           | 1000 | 8.6e-3   | 273.4 |
| STEA2-R-MinCos  | 1000 | 7.8e-3   | 282.8 |

Fig. 5 Convergence history of the MinCos method, and the accelerations R-MinCos, STEA2, and STEA2-R-MinCos for the Well1850 matrix with $n = 1850$, $m = 712$, MaxIt = 1000, MAXCOL = 40, and NCY = 25
number of iterations without attaining the required demanding accuracy of 0.5e-15. Nevertheless, we can observe in Fig. 5 that Random MinCos maintains during the whole process a monotone and trustable accelerated convergence behavior. It is also worth noticing that the STEA2 scheme shows an acceleration tendency up to iteration 600 (as well as STEA2-R-MinCos up to iteration 450), and after that they both exhibit an erratic behavior. As a consequence, the STEA2 and STEA2-R-MinCos cannot achieve an acceptable precision when they reached the maximum number of iterations.

6 Conclusions and perspectives

We have extended the MinCos iterative method, originally developed in [13] 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 several 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 $A^T A$ without requiring the explicit use of $A^T$. They also show that both schemes (Algorithms 1 and 2) can be significantly accelerated by the discussed strategies. However, as expected [19], there is no universal acceleration that will work effectively for all possible scenarios. In particular, the STEA2 scheme, from the simplified topological $\epsilon$-algorithms family, and the ABBmin and Random MinCos schemes, from the line search family, are clearly the most effective options. Our conclusion is that for small to medium size problems which are not ill-conditioned, the inexpensive ABBmin scheme represents an aggressive effective acceleration, and the STEA2 scheme represents a competitive 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 best considered option.

There exist some other promising extrapolation methods that do not belong to the topological $\epsilon$-algorithms family, like the Anderson acceleration method [1], as well as some other recently developed line search fast gradient-type schemes that have been properly adapted to a more general setting [29], including our specific case of minimizing a non-convex merit function over a non-convex feasible set (see Remark 2.1). Anderson’s acceleration, which has been recently applied on a wide variety of applications (see, e.g., [8, 27, 28, 40]), considers at each iteration a convex combination of a fixed number of previous iterates, and solves an auxiliary low-dimensional least-squares problem to find the coefficients of the convex combination. Adapting the Anderson acceleration strategy to our geometrical non-convex setting is an interesting challenge that requires additional understanding.

An interesting application of inverse approximation methods is the development of preconditioning techniques, for which in many real problems a sparse approximation is required. For future work, dropping or filtering schemes can be adapted to the extensions of the MinCos method to build suitable preconditioning strategies for solving linear least-squares problems, as it was done and extensively discussed.
in [13] for solving symmetric and positive definite large-scale linear systems of equations.

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 [13].

Acknowledgments We would like to thank two anonymous referees for their constructive comments and suggestions that helped us improve the final version of this paper.

Funding information This study is financially supported by CNRS throughout a 3-month *Poste Rouge* stay, in LAMFA Laboratory, (UMR CNRS 7352) at Université de Picardie Jules Verne, Amiens, France, from February 1 to April 30, 2019; and also by the Fundação para a Ciência e a Tecnologia (Portuguese Foundation for Science and Technology) through the project UID/MAT/00297/2019 (CMA), starting on May 1, 2019.

References

1. Anderson, D.G.: Iterative procedures for nonlinear integral equations. J. ACM 12, 547–560 (1965)
2. Andreani, R., Raydan, M., Tarazaga, P.: On the geometrical structure of symmetric matrices. Linear Algebra Appl. 438, 1201–1214 (2013)
3. Brezinski, C.: Convergence acceleration during the 20th century. J. Comput. Appl. Math. 122, 1–21 (2000). Numerical Analysis in the 20th Century Vol. II: Interpolation and Extrapolation
4. Brezinski, C., Redivo-Zaglia, M.: Extrapolation methods theory and practice. North-Holland, Amsterdam (1991)
5. Brezinski, C., Redivo-Zaglia, M.: The simplified topological \( \varepsilon \)-algorithms for accelerating sequences in a vector space. SIAM J. Sci. Comput. 36, 2227–2247 (2014)
6. Brezinski, C., Redivo-Zaglia, M.: The simplified topological \( \varepsilon \)-algorithms: software and applications. Numer. Algor. 74, 1237–1260 (2017)
7. Brezinski, C., Redivo-Zaglia, M.: The genesis and early developments of Aitkens process, Shanks transformation, the \( \varepsilon \)-algorithm, and related fixed point methods. Numer. Algor. 84, 11–133 (2019)
8. Brezinski, C., Redivo-Zaglia, M., Saad, Y.: Shanks sequence transformations and Anderson acceleration. SIAM Rev. 60, 646–669 (2018)
9. Carr, L.E., Borges, C.F., Giraldo, F.X.: An element based spectrally optimized approximate inverse preconditioner for the Euler equations. SIAM J. Sci. Comput. 34, 392–420 (2012)
10. Chehab, J.-P.: Matrix differential equations and inverse preconditioners. Comput. Appl. Math. 26, 95–128 (2007)
11. Chehab, J.-P.: Sparse approximations of matrix functions via numerical integration of ODEs. Bull. Comput. Appl. Math. 4, 95–132 (2016)
12. Chehab, J.P., Raydan, M.: Geometrical properties of the Frobenius condition number for positive definite matrices. Linear Algebra Appl. 429, 2089–2097 (2008)
13. Chehab, J.P., Raydan, M.: Geometrical inverse preconditioning for symmetric positive definite matrices. Mathematics 4(3), 46 (2016). https://doi.org/10.3390/math4030046
14. Chen, K.: An analysis of sparse approximate inverse preconditioners for boundary integral equations. SIAM J. Matrix Anal. Appl. 22, 1058—1078 (2001)
15. Chow, E., Saad, Y.: Approximate inverse techniques for block-partitioned matrices. SIAM J. Sci. Comput. 18, 1657–1675 (1997)
16. Chung, J., Chung, M., O’Leary, D.P.: Optimal regularized low rank inverse approximation. Linear Algebra Appl. 468, 260–269 (2015)
17. Cui, X., Hayami, K.: Generalized approximate inverse preconditioners for least squares problems. J. Pan J. Indus. Appl. Math. 26, 1–14 (2009)
18. De Asmundis, R., di Serafino, D., Riccio, F., Toraldo, G.: On spectral properties of steepest descent methods. IMA J. Numer. Anal. 33, 1416–1435 (2013)
19. Delahaye, J.-P., Germain-Bonne, B.: Résultats négatifs en accélération de la convergence. Numer. Math. 35, 443–457 (1980)
20. Diniz-Ehrhardt, M.A., Martínez, J.M., Raydan, M.: A derivative-free nonmonotone line search technique for unconstrained optimization. J. Comput. Appl. Math. 219, 383–397 (2008)
21. Forsman, K., Gropp, W., Kettunen, L., Levine, D., Salonen, J.: Solution of dense systems of linear equations arising from integral equation formulations. Antennas and Propagation Magazine 37, 96–100 (1995)
22. Frassoldati, G., Zanni, L., Zanghirati, G.: New adaptive stepsize selections in gradient methods. J. Ind. Manag. Optim. 4, 299–312 (2008)
23. Glunt, W., Hayden, T.L.: M. Raydan molecular conformations from distance matrices. J. Comput. Chem. 14, 114–120 (1993)
24. Gould, N.I.M., Scott, J.A.: Sparse approximate-inverse preconditioners using norm-minimization techniques. SIAM J. Sci. Comput. 19, 605–625 (1998)
25. Graves-Morris, P.R., Roberts, P.R., Salam, A.: The epsilon algorithm and related topics. J. Comput. Appl. Math. 122, 51–80 (2000)
26. Helsing, J.: Approximate inverse preconditioners for some large dense random electrostatic interaction matrices. BIT Numer. Math. 46, 307–323 (2006)
27. Henderson, N.C., Varadhan, R.: Damped Anderson acceleration with restarts and monotonicity control for accelerating EM and EM like algorithms. Journal of Computational and Graphical Statistics. https://doi.org/10.1080/10618600.2019.1594835 (2019)
28. Higham, N., Strabić, N.: Anderson acceleration of the alternating projections method for computing the nearest correlation matrix. Numer. Algor. 72, 1021–1042 (2016)
29. Iannazzo, B., Porcelli, M.: The Riemannian Barzilai-Borwein method with nonmonotone line-search and the matrix geometric mean computation. IMA J. Numer. Anal. 38, 495–517 (2018)
30. Jbilou, K., Messaoudi, A.: Block extrapolation methods with applications. Appl. Numer. Math. 106, 154–164 (2016)
31. Jbilou, K., Sadok, H.: Vector extrapolation methods: applications and numerical comparison. J. Comput. Appl. Math. 122, 149–165 (2000)
32. Jbilou, K., Sadok, H.: Matrix polynomial and epsilon-type extrapolation methods with applications. Numer. Algor. 68, 107–119 (2015)
33. Matos, A.C.: Convergence and acceleration properties for the vector $\epsilon$-algorithm. Numer. Algor. 3, 313–320 (1992)
34. Matrix Market. http://math.nist.gov/MatrixMarket/
35. Montero, G., González, L., Flocrèz, E., García, M.D., Suárez, A.: Approximate inverse computation using Frobenius inner product. Numerical Linear Algebra with Applications 9, 239–247 (2002)
36. Raydan, M., Svaiter, B.: Relaxed steepest descent and Cauchy-Barzilai-Borwein method. Comput. Optim. Appl. 21, 155–167 (2002)
37. Sajo-Castelli, A.M., Fortes, M.A., Raydan, M.: Preconditioned conjugate gradient method for finding minimal energy surfaces on Powell-Sabin triangulations. J. Comput. Appl. Math. 268, 34–55 (2014)
38. di Serafino, D., Ruggiero, V., Toraldo, G., Zanni, L.: On the steplength selection in gradient methods for unconstrained optimization. Appl. Math. Comput. 318, 176–195 (2018)
39. Sidje, R.B., Saad, Y.: Rational approximation to the Fermi–Dirac function with applications in density functional theory. Numer. Algor. 56, 455–479 (2011)
40. Walker, H.F., Ni, P.: Anderson acceleration for fixed-point iterations. SIAM J. Numer. Anal. 49, 1715–1735 (2011)
41. Wang, J.-K., Lin, S.D.: Robust inverse covariance estimation under noisy measurements. In: Proceedings of the 31st International Conference on Machine Learning (ICML-14), 928–936 (2014)
42. Zhou, B., Gao, L., Dai, Y.H.: Gradient methods with adaptive step-sizes. Comput. Optim. Appl. 35, 69–86 (2006)

Publisher’s note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.