On gradient based descent algorithms for joint diagonalization of matrices

Erik Troedsson, Marcus Carlsson, Herwig Wendt

On gradient based descent algorithms for joint diagonalization of matrices

Erik Troedsson
Centre for Mathematical Sciences
Lund University, Lund, Sweden
erik.troedsson@math.lu.se

Marcus Carlsson
Centre for Mathematical Sciences
Lund University, Lund, Sweden
marcus.carlsson@math.lu.se

Herwig Wendt
CNRS, IRIT
Université de Toulouse, Toulouse, France
herwig.wendt@irit.fr

***Abstract*—Joint diagonalization of collections of matrices, i.e. the problem of finding a joint set of approximate eigenvectors, is an important problem that appears in many applicative contexts. It is commonly formulated as finding the minimizer, over the set of all possible bases, for a certain non-convex functional that measures the size of off-diagonal elements. Many approaches have been studied in the literature, some of the most popular ones working with approximations of this cost functional. In this work, we deviate from this philosophy and instead propose to directly attempt to find a minimizer making use of the gradient and Hessian of the original functional. Our main contributions are as follows. First, we design and study gradient descent and conjugate gradient algorithms. Second, we show that the intricate geometry of the functional makes it beneficial to change basis at each iteration, leading to faster convergence. Third, we conduct large sets of numerical experiments that indicate that our proposed descent methods yield competitive results when compared to popular methods such as WJDTE.**

***Index Terms*—matrix diagonalization, simultaneous diagonalization, joint eigen-decomposition, gradient descent, conjugate gradient**

I. Introduction

Context. Let $\mathcal{A} = \{A_1, \dots, A_K\}$ be collection of $n \times n$ real matrices. The problem of finding a nonsingular matrix U that approximately diagonalizes all matrices in $\mathcal{A}$ is known as Joint Diagonalization (JD), and appears in numerous applications in various fields such as blind source separation and independent component analysis [1], [2], canonical polyadic decomposition [3], [4], or multidimensional harmonic analysis [5], [6], to name but a few. A common denominator in most works is the desire to minimize the functional

$$f_{\mathcal{A}}(U) = \frac{1}{2} \sum_{k=1}^{K} \sum_{i \neq j} \left| (U^{-1} A_k U)_{ij} \right|^2 , \qquad (1)$$

defined for nonsingular matrices U. Throughout this work we shall for simplicity assume $\mathcal{A}$ and U to be real, but our results have a natural extension to the complex case.

Related work. Many different approaches have been studied for solving the Joint Diagonalization problem, (also known as "Joint Eigen-Decomposition" or "Simultaneous Diagonalization"), sometimes in combination with additional assumptions such as $U^{-1} = U^*$, which naturally appears if the matrices A_k are self-adjoint. These approaches rely on, for instance, LU decompositions [7], optimization formulated on manifolds [8]–[10], random methods [11], as well as approximations of (1) [12], to name a few. This latter approach recently yielded a fast algorithm, termed WJDTE, that leads to state-of-the-art results, and will serve as our benchmark in this work. Just like [12], our approach focuses on the unconstrained case but does not require approximations. Instead, it relies on finding critical points of (1) based on the computation of its gradient, aided by the Hessian to pick suitable step-lengths. The expression for the Hessian is to our best knowledge new, whereas the gradient appears in earlier works such as [13]. However, these articles have been largely overlooked and, to our knowledge, no operational and competitive gradient-based method has been implemented or studied thoroughly.

Goals, contributions and outline. The present work aims at filling this gap and proposes, as key novelties: 1) an operational gradient descent algorithm and 2) an efficient *conjugate gradient* algorithm for finding a local minimizer, or at least a stationary point, of $f_{\mathcal{A}}$. More precisely, we first give expressions for the gradient and Hessian of $f_{\mathcal{A}}$. Second, we show that due to the intricate geometry of the highly non-convex functional $f_{\mathcal{A}}$, a certain multiplicative change of basis at each iteration is beneficial in order for good convergence of related algorithms. Third, we design and study a gradient descent algorithm, with an efficient Newton step-size selection based on the Hessian (see Sec. II). Fourth, we detail and implement a conjugate gradient method, which requires a non-standard and novel update step due to the change of basis that we discuss in detail (cf., Sec. III). Finally, in Sec. IV, we provide a large set of numerical results that indicate that both our gradient descent and conjugate gradient method lead to better solutions than the state-of-the-art method WJDTE, and that our conjugate gradient method is moreover competitive in terms of computation time.

II. Gradient, Hessian and change of basis

A. Gradient and Hessian

We first compute the gradient and the Hessian of (1). Since we are working over a linear vector space (of matrices), recall that the gradient $\nabla f_{\mathcal{A}}$ evaluated at U is the matrix $\nabla f_{\mathcal{A}}|_U$ such that

$$f_{\mathcal{A}}(U + Z) = f_{\mathcal{A}}(U) + \langle \nabla f_{\mathcal{A}}|_U, Z \rangle + o(||Z||),$$

Work funded by Swedish Research Council Grant 2022-04917.

where the norm (and scalar product) refer to the Frobenius norm and $o(||Z||)$ denotes "little ordo". Analogously, the Hessian $H_{\mathcal{A}}|_U$ at the point U can be defined as the unique symmetric linear operator, taking matrices into matrices, such that the ordo term above can be written

$$\frac{1}{2}\langle Z, H_{\mathcal{A}}|_U(Z)\rangle + o(||Z||^2).$$

The expressions for $\nabla f_{\mathcal{A}}|_U$ and $H_{\mathcal{A}}|_U$ are given in Theorem 1, whose proof we postpone to a forthcoming paper for space reasons.

Theorem 1: The gradient and Hessian of (1) are given by

$$\nabla f_{\mathcal{A}}|_U = \sum_{k=1}^{K} (U^{-1})^T \left[D_k^T, (D_k \diamond J)\right], \tag{2}$$

$$\begin{aligned} H_{\mathcal{A}}|_U(Z) = \sum_{k=1}^{K} (U^{-1})^T \Big(&\left[D_k^T, J \diamond (D_k U^{-1} Z)\right] \\ &+ \left[J \diamond (U^{-1} Z D_k), D_k^T\right] \\ &+ \left[Z^T (U^{-1})^T (J \diamond D_k), D_k^T\right] \\ &+ (J \diamond D_k)\left[D_k^T, Z^T (U^{-1})^T\right]\Big), \end{aligned} \tag{3}$$

where $D_k \triangleq U^{-1} A_k U$, $[X, Y] = XY - YX$ is the matrix commutator, J is the matrix with zeros on the diagonal and ones elsewhere, and $\diamond$ denotes Hadamard-multiplication of matrices.

B. Multiplicative basis change

A basic gradient descent algorithm starts with an initial point U_0 and then iteratively uses the gradient as search direction to compute the next point as

$$U_{m+1} = U_m - \lambda_m \nabla f_{\mathcal{A}}|_{U_m} \tag{4}$$

where $\lambda_m > 0$ is the step size parameter. In this setting the matrix U_m is updated additively, while not changing the initial matrix tuple $\mathcal{A}$. However, since

$$f_{\mathcal{A}}(U_m V) = f_{U_m^{-1}\mathcal{A}U_m}(V)$$

we may instead consider computing the gradient of $f_{U_m^{-1}\mathcal{A}U_m}$ at the identity and update via

$$U_{m+1} = U_m(I - \lambda_m \nabla f_{U_m^{-1}\mathcal{A}U_m}|_I).$$

Setting $A_{k,m} = U_m^{-1} A_k U_m$ and writing $\mathcal{A}_m$ for $\{A_{1,m}, \ldots, A_{K,m}\}$, this leads to the following scheme (with $\mathcal{A}_0 = \mathcal{A}$ and initializing at $U_0 = I$):

1) Given $\mathcal{A}_m$, compute $\mathcal{A}_{m+1}$ by setting

$$A_{k,m+1} = (I - \lambda_m \nabla f_{\mathcal{A}_m}|_I)^{-1} A_{k,m} (I - \lambda_m \nabla f_{\mathcal{A}_m}|_I)$$

2) Update U_m according to

$$U_{m+1} = U_m(I - \lambda_m \nabla f_{\mathcal{A}_m}|_I). \tag{5}$$

The above algorithm functions by a multiplicative update of U_m as opposed to an additive one. Multiplicative updates have been used before in, e.g, [12], without deeper analysis or comparison. Our numerical results suggest that this alternative increases performance by a large amount, in terms of convergence speed. Moreover, the gradient (2) and the Hessian (3) at the identity matrix $U = I$ are cheaper to compute since no matrix inverse needs to be computed. In Sec. III we will further develop this idea into a "multiplicative" version of conjugate gradient descent.

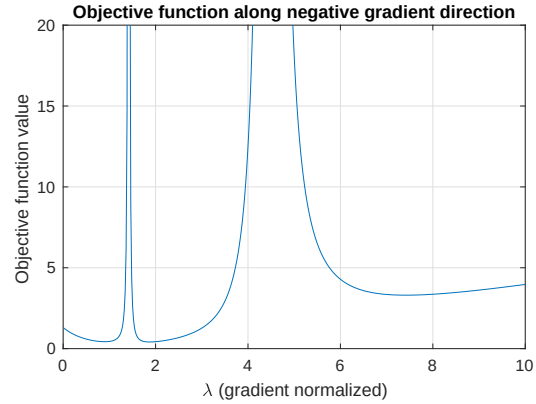

Fig. 1. The objective function $f_{\mathcal{A}}$ (1) for a random matrix tuple $\mathcal{A}$ of five 5×5 matrices plotted along the line $I + \lambda S$ where $S = -\nabla f_{\mathcal{A}}|_I / ||\nabla f_{\mathcal{A}}|_I||$.

C. Non-convexity of $f_{\mathcal{A}}$

The functional $f_{\mathcal{A}}$ is highly non-convex and its analysis is intricate, see Fig. 1 for an example of how it can look in the gradient direction. Why the multiplicative basis change works better for all the algorithms we have evaluated remains unclear theoretically, but possibly the basis change somehow improves the local geometry of the graph. Empirically the effect is substantial, see Sec. IV-B and Fig. 2.

The non-convexity also makes choosing step-size a delicate matter. Given a fixed direction S at some point U, our proposed strategy is to use the Hessian to find a reasonable step size λ. Noting that $f_{\mathcal{A}}(U + \lambda S)$ approximately equals

$$f_{\mathcal{A}}(U) + \lambda \langle \nabla f_{\mathcal{A}}|_U, S\rangle + \frac{1}{2}\lambda^2 \langle S, H_{\mathcal{A}}|_U(S)\rangle,$$

and assuming that $\langle S, H_{\mathcal{A}}|_U(S)\rangle > 0$, we can minimize this over λ to get

$$\lambda = -\frac{\langle \nabla f_{\mathcal{A}}|_U, S\rangle}{\langle S, H_{\mathcal{A}}|_U(S)\rangle}. \tag{6}$$

However, there is no guarantee for the denominator to be positive, leading to spurious objective function increase and possible divergence. An alternative is to further study the denominator $\langle S, H_{\mathcal{A}}|_U(S)\rangle$ in (6) which after expanding and simplifying can be written as

$$\sum_{k=1}^{K} \|J \diamond [D_k, U^{-1}S]\|^2 + 2\langle J \diamond D_k, [U^{-1}S, U^{-1}SD_k]\rangle$$

By discarding the second term inside the sum above we obtain a nonnegative Gauss-Newton approximation of the Hessian,

which leads to the following step-size method that can be used in cases when (6) gives a negative answer:

$$\lambda_{GN} = -\frac{\langle \nabla f_{\mathcal{A}}|_U, S\rangle}{\|J \diamond [D_k, U^{-1}S]\|^2}. \tag{7}$$

Finally, since $f_{\mathcal{A}}$ is nonconvex, care must be taken so that the step-size does not become too large. A good step size method will approximately minimize $f_{\mathcal{A}}$ over the line $U+\lambda S$. The behavior of $f_{\mathcal{A}}$ along such a line is quite peculiar, see Fig. 1. However, we have the following result.

Theorem 2: $f_{\mathcal{A}}(U+\lambda S)$ has no singularity on the interval $0 \leq \lambda < \|U^{-1}S\|^{-1}$.

Proof. Since the inverse $(U+\lambda S)^{-1}$ is present in the defining formula (1), $f_{\mathcal{A}}(U+\lambda S)$ will typically have poles at the singularities of $(U+\lambda S)^{-1}$, which are precisely the reciprocal of the non-zero eigenvalues of $U^{-1}S$ after change of sign. Thus, the first singularity of $f_{\mathcal{A}}(U+\lambda S)$ will be at distance $1/\rho(U^{-1}S)$ from the origin, where $\rho(U^{-1}S)$ is the spectral radius of $U^{-1}S$. Since we always have $\rho(U^{-1}S) \leq \|U^{-1}S\|$, a lower bound on this first singularity is given by $\|U^{-1}S\|^{-1}$ which can be used as a maximally allowed step size in order to avoid hitting a singularity with the methods (6) or (7).

III. Multiplicative conjugate gradient descent

The standard gradient descent method has infamously slow convergence. One possible improvement is to use the conjugate gradient method instead. Armed with Theorem 1, and the step size methods (6) and (7), the gradient descent is straightforward to implement. However, it becomes a bit unclear how to interpret conjugate gradient descent in combination with the everchanging basis, in the multiplicative update setting, which we now discuss.

A. Multiplicative Conjugate Gradient

Recall that the idea of the conjugate gradient method in the classical setting is to replace the gradient directions $-\nabla f_{\mathcal{A}}(U_m)$ with search directions S_m and ensure that these directions are orthogonal with respect to the inner product induced by the Hessian operator (3)

$$\langle S_m, H_{\mathcal{A}}|_{U_m}(S_{m-1})\rangle = 0, \tag{8}$$

for all m. In the multiplicative update setting, we end up with a curious situation where after the matrix tuple $\mathcal{A}_{m-1}$ is updated to $\mathcal{A}_m$ we are effectively at the point $U_m = I$ and are now seeking to minimize the new functional $f_{\mathcal{A}_m}$, where

$$\mathcal{A}_m = (I+\lambda_{m-1}S_{m-1})^{-1}\mathcal{A}_{m-1}(I+\lambda_{m-1}S_{m-1}). \tag{9}$$

The above transformation of $\mathcal{A}_{m-1}$ corresponds to a change of basis on the underlying matrix space we are optimizing over, where objects V in step $m-1$ get transformed by $(I+\lambda_{m-1}S_{m-1})^{-1}V$. Thus, the old direction S_{m-1} will, with respect to the new tuple $\mathcal{A}_m$, correspond to

$$\tilde{S}_{m-1} \triangleq (I+\lambda_{m-1}S_{m-1})^{-1}S_{m-1}. \tag{10}$$

We now take the new search direction S_m to be of the form

$$S_m = -\nabla f_{\mathcal{A}_m}|_I + \beta_m \tilde{S}_{m-1}. \tag{11}$$

Inserting the above into the conjugate criterion (8), we obtain

$$0 = -\langle \nabla f_{\mathcal{A}_m}|_I, H_{\mathcal{A}_m}|_I(\tilde{S}_{m-1})\rangle + \beta_m \langle \tilde{S}_{m-1}, H_{\mathcal{A}_m}|_I(\tilde{S}_{m-1})\rangle,$$

and thus we have the following formula for β_m,

$$\beta_m = \frac{\langle \nabla f_{\mathcal{A}_m}|_I, H_{\mathcal{A}_m}|_I(\tilde{S}_{m-1})\rangle}{\langle \tilde{S}_{m-1}, H_{\mathcal{A}_m}|_I(\tilde{S}_{m-1})\rangle}. \tag{12}$$

The proposed conjugate gradient descent method hence consists of the following steps:

Initialization ($m = 0$): Set $S_{m-1} = -\nabla f_{\mathcal{A}}|_I$ and perform a gradient descent step.

1) Given S_{m-1}, $\mathcal{A}_m$ and U_m, compute $\tilde{S}_{m-1}$ via (10).
2) Compute the new search direction S_m according to (11) and (12).
3) Compute the step-size λ_m with (6) or, if negative, with (7) (using $\mathcal{A} = \mathcal{A}_m$, $S = S_m$ and $U = I$).
4) Compute $\mathcal{A}_{m+1}$ via (9) (using $m+1$ in place of m).
5) Update the eigenvectors with
$$U_{m+1} = U_m(I+\lambda_m S_m).$$
6) Repeat steps 1-5 until convergence.

It should be noted that the conjugate gradient method is only proven to converge when the optimization problem it is applied on is (at least) convex. Nevertheless, conjugate gradient methods are still frequently applied on non-convex problems and, as we show in the next section, it works well on the functional (1). However, when far from a local minimum, it may happen that the Hessian is negative at a given step, and hence there is no guarantee that the conjugate directions S_m will even be descent directions. In this case one can simply take the gradient as descent direction instead, which can be achieved by setting β to 0 in the above algorithm when (12) returns a negative number.

IV. Numerical Experiments and Results

Our proposed multiplicative gradient descent and conjugate gradient algorithms for joint diagonalization will be denoted JD-MG and JD-MCG, respectively.

A. Monte Carlo simulations

We study the performance of the proposed multiplicative update descent algorithms by applying them to a large number of realizations ($N_{MC} = 1000$) of collections $\mathcal{A} = \{A_1, \ldots, A_K\}$ of K $n \times n$ matrices, to which centered white Gaussian noise is added for several signal to noise ratios (SNR). Each matrix $A_k, k = 1, \ldots, K$ is constructed as $A_k = Z\Delta_k Z^{-1}$, where Z is an $n \times n$ matrix whose entries are i.i.d. zero mean Gaussian random variables, normalized to yield unit norm columns, and Δ_k are diagonal matrices with uniform random variables on the interval $[0, 1]$. We compare our method with the state-of-the-art algorithm WJDTE [12]. If not mentioned otherwise, all algorithms are stopped after $M = 1000$ iterations, without other stopping criterion. Results reported here were obtained for the case $n = 10$ and $K = 6$; similar results are obtained for other values for n and K and not reported here for space reasons.

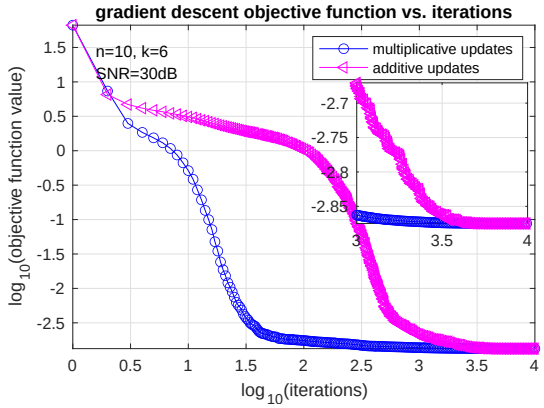

Fig. 2. **Additive vs. multiplicative updates.** Median of the objective function values versus iterations for gradient descent with additive (4) and multiplicative (5) updates.

B. Multiplicative vs. additive updates

Fig. 2 shows the median over realizations of the objective function values versus iterations for gradient descent when either the standard additive update (4) or the multiplicative update (5) are used (SNR=30dB, $M = 10000$ iterations). Clearly, the algorithm with multiplicative update converges much faster than the one with additive update: Indeed, the latter requires more than 10 times more iterations to reach the same objective function value than the algorithm with multiplicative update. A possible reason for this could be that the objective function for the multiplicative update is *locally* better behaved: indeed, because of the update of $\mathcal{A}_m$ at each step, it in fact works at each step with a *different, local objective function* $f_{\mathcal{A}_m}$ evaluated at the point $U = I$, rather than a fixed objective function $f_{\mathcal{A}}$. A theoretical investigation of this phenomenon is left for future work.

C. Performance analysis

Fig. 3 provides a comparison of the performance of JD-MG, JD-MCG and WJDTE in terms of the objective function values that each method achieves. Investigation of the median (over realizations) objective function value at convergence as a function of noise level (Fig. 3, top) leads us to conclude that JD-MG and JD-MCG converge to the same minimum of the objective function, while WJDTE converges to minima with significantly larger objective function values. As expected, better results are obtained for larger SNR values for all methods. Fig. 3 (center) provides a more detailed analysis and plots, for each individual realization, the final objective function values for WJDTE against those for JD-MCG (SNR=20dB). It corroborates the conclusions from average performance results and shows that JD-MCG systematically yields a better objective function value than WJDTE (up to more than one order of magnitude). Finally, Fig. 3 (bottom) provides a complementary view on this result and plots the objective function values as a function of the condition number of the (random drawn) matrix Z for each realization. It shows that performance is essentially a function of this condition

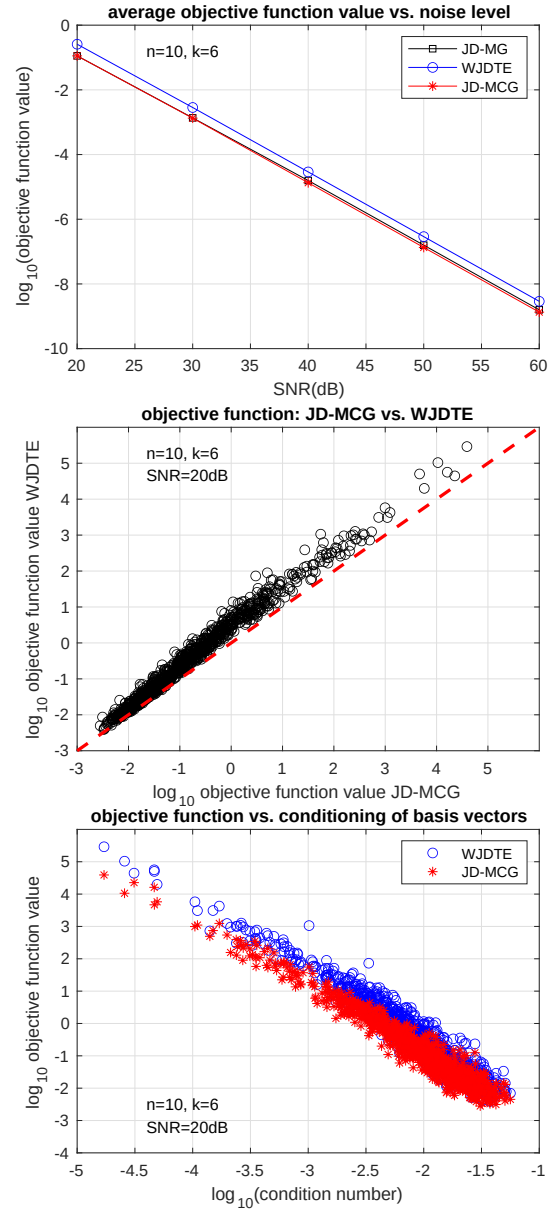

Fig. 3. **Performance comparisons.** Median of objective function values at convergence as a function of noise level (top); scatter plot of objective function values at convergence for conjugate gradient vs. WJDTE (center), and against the condition numbers of the matrices Z (bottom, SNR=20dB).

number: the better conditioned Z, the lower the obtained objective function values. Moreover, the performance gains yielded by JD-MCG as compared to WJDTE are larger for more difficult situations (smaller condition number) and more moderate for more easier situations (large condition number).

D. Convergence analysis

Fig. 4 (top) plots the median of the objective function values obtained by the different methods as a function of iteration number. It suggests that WJDTE gets stuck at minima with significantly larger objective function value than JD-MG and JD-MCG after a few iterations. JD-MCG reaches objective

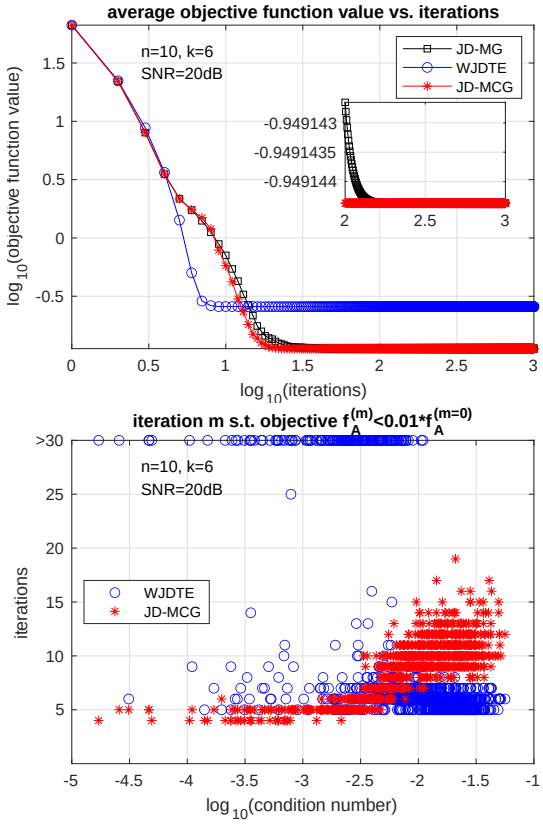

Fig. 4. Median of objective function values versus iterations (top) and scatter plots of objective function values at convergence (bottom); SNR=20dB.

function values close to its minimum at convergence in a small but slightly larger number of iterations, whereas JD-MG requires significantly more iterations but eventually finds the same optimal point as JD-MCG. Motivated by the analysis from the previous section, Fig. 4 (bottom) provides a more detailed analysis of the convergence for WJDTE and JD-MCG: It plots, for each individual realization, the number of iterations needed to decrease the objective function value to a fraction of $\frac{1}{100}$ of its initial value, as a function of the condition number of the matrix Z. It suggests that few iterations are needed for WJDTE (5-10 iterations) to reach this value when condition number is large ($\geq 10^{-2}$), while for smaller condition numbers many iterations ($\gg 30$) are needed, or the target value can not be reached at all with WJDTE. On the contrary, JD-MCG consistently converges below the target value in less than 15 iterations, regardless of the difficulty of the problem (i.e., the condition number of Z).

Overall, these results lead to the conclusion that the proposed descent algorithms are operational, effective and competitive. In particular, they find better solutions than the WJDTE method in terms of objective function values at convergence. Moreover, they lead to more consistent results than the WJDTE method (which in turn outperforms many preceding approaches, see [12] for extensive comparisons) across various levels of difficulty of the joint diagonalization problem, as quantified by the condition number of the ground truth matrix Z.

V. Conclusions

In this work, we studied gradient based descent algorithms for the joint diagonalization of collections of matrices and developed a novel conjugate gradient method for the problem. Our approach builds on three key elements: first, the use of a multiplicative change of basis at each iteration, which leads to significantly better convergence; second, analysis and evaluation of the Hessian as an efficient and effective means to determine step sizes; third, derivation of the non-standard conjugate gradient update required by the local change of basis. Our numerical results indicate that our proposed conjugate gradient approach JD-MCG yields competitive results when compared to the state of the art in terms of objective function values at convergence, number of iterations as well as robustness. Future work will focus on the theoretical analysis of the objective function and change of basis, and on applications to high dimensional harmonic retrieval problems.

References

[1] J.-F. Cardoso and A. Souloumiac, "Blind beamforming for non-gaussian signals," in *IEE proceedings F (radar and signal processing)*, vol. 140, no. 6, 1993, pp. 362–370.

[2] A. Ziehe, P. Laskov, G. Nolte, and K.-R. MÃžller, "A fast algorithm for joint diagonalization with non-orthogonal transformations and its application to blind source separation," *Journal of Machine Learning Research*, vol. 5, no. Jul, pp. 777–800, 2004.

[3] F. Roemer and M. Haardt, "A semi-algebraic framework for approximate cp decompositions via simultaneous matrix diagonalizations (secsi)," *Signal Process.*, vol. 93, no. 9, pp. 2722–2738, 2013.

[4] X. Luciani and L. Albera, "Canonical polyadic decomposition based on joint eigenvalue decomposition," *Chemometrics and Intelligent Laboratory Systems*, vol. 132, pp. 152–167, 2014.

[5] M. Haardt, F. Roemer, and G. Del Galdo, "Higher-order svd-based subspace estimation to improve the parameter estimation accuracy in multidimensional harmonic retrieval problems," *IEEE Trans. Signal Process.*, vol. 56, no. 7, pp. 3198–3213, 2008.

[6] F. Andersson and M. Carlsson, "Esprit for multidimensional general grids," *SIAM J. Matrix Analysis and Applications*, vol. 39, no. 3, pp. 1470–1488, 2018.

[7] X. Luciani and L. Albera, "Joint eigenvalue decomposition of non-defective matrices based on the lu factorization with application to ica," *IEEE Trans. Signal Process.*, vol. 63, no. 17, pp. 4594–4608, 2015.

[8] A. Mesloub, A. Belouchrani, and K. Abed-Meraim, "Efficient and stable joint eigenvalue decomposition based on generalized givens rotations," in *Proc. European Signal Process. Conf. (EUSIPCO)*, Rome, Italy, 2018.

[9] R. Iferroudjene, K. A. Meraim, and A. Belouchrani, "A new jacobi-like method for joint diagonalization of arbitrary non-defective matrices," *Applied Math. and Comput.*, vol. 211, no. 2, pp. 363–373, 2009.

[10] X.-F. Gong, K. Wang, and Q.-H. Lin, "Complex non-orthogonal joint diagonalization with successive givens and hyperbolic rotations," in *Proc. Int. Conf. Acoustics Speech and Signal Process. (ICASSP)*, Kyoto, Japan, 2012.

[11] H. He and D. Kressner, "Randomized joint diagonalization of symmetric matrices," *SIAM J. Matrix Analysis and Applications*, vol. 45, no. 1, pp. 661–684, 2024.

[12] R. André, X. Luciani, and E. Moreau, "Joint eigenvalue decomposition algorithms based on first-order taylor expansion," *IEEE Trans. Signal Process.*, vol. 68, pp. 1716–1727, 2020.

[13] G. Hori, "Joint diagonalization and matrix differential equations," in *Proc. Int. Symp. Nonlinear Theory and Applications (NOLTA)*, Hawaii, USA, 1999.