ON THE CONVERGENCE OF SIPEX: A SIMULTANEOUS PRINCIPAL COMPONENTS EXTRACTION ALGORITHM

Deniz Erdogmus[1], *Yadunandana Rao*[1], *M. Can Ozturk*[1], *Luis Vielva*[2], *Jose C. Principe*[1]

[1] CNEL, Electrical and Computer Eng. Dept., University of Florida, Gainesville, FL 32611, USA
[2] GTAS, DICOM, Universidad de Cantabria, Santander, Spain

ABSTRACT

We have previously proposed SIPEX as a fast-converging and accurate principal components algorithm. Its superiority in terms of data efficiency and solution accuracy was demonstrated through Monte Carlo simulations. In this paper, we focus on the convergence properties of the original gradient-based algorithm as well as two modified versions of SIPEX based on approximations to the Hessian of the cost function. We provide practical bounds on the step sizes of these algorithms and compare their convergence properties.

1. INTRODUCTION

Principal components analysis (PCA) is an important statistical tool that has found place in many important signal processing applications. The first on-line PCA algorithms started with Oja's [1] and Sanger's [2] rules. These can be grouped into two main categories: gradient-based and fixed-point algorithms. Regardless of category, almost all PCA algorithms use the deflation-normalization procedure to guarantee the orthonormality of the solution. Exceptions include LMSER [3], and APEX [4]. The weight matrix of LMSER is not restricted to be orthonormal. Fixed-point algorithms, e.g., power rule [5] converge fast, but they still need to use deflation, which is undesirable due to accuracy considerations.

Recently, we have proposed a gradient-based algorithm for simultaneous principal component extraction (SIPEX-G), which has been demonstrated to outperform benchmark PCA algorithms [6,7]. In these publications, we did not investigate the stability conditions. Clearly, any update rule using a step size can be made to converge to the vicinity of the solution. Once in this neighborhood, it is important to select a proper step size to converge stably to the optimal solution.

In this paper, we will determine the stability conditions for SIPEX-G and two other variants, named SIPEX-L and SIPEX-H. The latter is motivated by the fact that the Hessian of the SIPEX criterion evaluated at a solution becomes diagonal, and hence, a Newton-type optimization algorithm becomes feasible. Monte Carlo simulations using synthetic data are performed to compare the convergence performances of these three algorithms.

2. AN OVERVIEW OF SIPEX-G

Consider a PCA network **y=Rx**, where $\mathbf{x},\mathbf{y}\in\Re^{nx1}$ are the zero-mean input and output vectors, respectively, and $\mathbf{R}\in D\subset\Re^{nxn}$, is the weight matrix restricted to the subset *D* of orthonormal matrices. It was shown in [6] that if **R** is parameterized in terms of Givens rotations, then all stationary points of

$$J=\sum_{o=1}^{n-1}\gamma_o Var(\mathbf{y}_o)=\sum_{o=1}^{n-1}\gamma_o\sum_{i=1}^{n}\sum_{j=1}^{n}\mathbf{R}_{oi}\mathbf{R}_{oj}\mathbf{\Sigma}_{ij} \qquad (1)$$

correspond to PCA solutions, where the rows of **R** are all the eigenvectors of the covariance matrix Σ=E[$\mathbf{xx}^T$]. The gains γ_o satisfy $\gamma_1>\gamma_2>...>\gamma_{n-1}>0$. In (1), the subscripts indicate the entry of the vector/matrix. SIPEX-G updates the Givens rotation angles, which are collected in a vector $\boldsymbol{\theta}$, using steepest ascent/descent. Since all stationary points are PCA solutions, any local minima, maxima, or saddle points are equally acceptable. In the previous papers, we have presented extensive comparisons of SIPEX-G with Sanger's rule, APEX, and LMSER that clearly demonstrated the superiority of SIPEX-G in speed and accuracy [6,7]. We also experimented with SIPEX-G in various problems including subspace Wiener filtering and direction of arrival estimation. SIPEX-G performed well in all these problems.

3. SIPEX-L AND SIPEX-H

In SIPEX-L, we use the matrix $(\mathbf{I}+\mathbf{G}_k\mathbf{G}_k^T)$ similar to the Levenberg algorithm, where $\mathbf{G}_k$ is the current gradient, and in SIPEX-H we use $\tilde{\mathbf{H}}_k=abs(diag(\mathbf{H}_k))$, where $\mathbf{H}_k$ is the current Hessian matrix. The *diag*(.) operator generates a diagonal matrix from the diagonal entries of the argument and *abs*(.) takes elementwise absolute values. In SIPEX, the current gradient and Hessian are defined as

$$\mathbf{G}_k = \frac{\partial J(\boldsymbol{\theta}_k)}{\partial \boldsymbol{\theta}_k} \quad \text{and} \quad \mathbf{H}_k = \frac{\partial^2 J(\boldsymbol{\theta}_k)}{\partial \boldsymbol{\theta}_k^2} \tag{2}$$

With these definitions, all three algorithms become

$$\begin{aligned} &\text{SIPEX-G}: \quad \boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta_k \mathbf{G}_k \\ &\text{SIPEX-L}: \quad \boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta_k \left(\mathbf{I} + \mathbf{G}_k \mathbf{G}_k^T\right)^{-1} \mathbf{G}_k \\ &\text{SIPEX-H}: \quad \boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta_k \tilde{\mathbf{H}}_k^{-1} \mathbf{G}_k \end{aligned} \tag{3}$$

Using the matrix inversion lemma, the matrix inverse in SIPEX-L is found as $\left(\mathbf{I} - \mathbf{G}_k \mathbf{G}_k^T / (1 + \mathbf{G}_k^T \mathbf{G}_k)\right)$, therefore this algorithm becomes

$$\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k - \eta_k \mathbf{G}_k \Big/ \left(1 + \mathbf{G}_k^T \mathbf{G}_k\right) \tag{4}$$

This is similar to the normalized LMS algorithm. SIPEX-L is a normalized gradient update rule. On the other hand, in SIPEX-H, the update direction is different from the gradient. The Hessian of (1) will be shown to be diagonal at the solutions. Therefore SIPEX-H essentially becomes a Newton update near the solutions.

4. LOCAL STABILITY CONDITIONS FOR SIPEX

In order to guarantee stable convergence to the solution once the weights are in its neighborhood, the step size must satisfy some inequality.

Theorem 1. The following is the upper bound on the step size of SIPEX-G that guarantees stable convergence to the solution nearby the current weights.

$$\eta_k \le \frac{1}{\max\limits_{\substack{p=1,\dots,n-1 \\ q=p+1,\dots,n}} \left| \gamma_p \bar{\lambda}_q + \gamma_q \bar{\lambda}_p - \gamma_p \bar{\lambda}_p - \gamma_q \bar{\lambda}_q \right|} \tag{5}$$

where $\bar{\lambda}_j$ is the eigenvalue corresponding to the eigenvector in the j^{th} row of $\mathbf{R}$ at this nearby solution.

Proof. From MSE adaptation, we know that gradient update stability requires $\eta \le 2/|\lambda_{\max}|$, where $\lambda_{\max}$ is the maximum eigenvalue of the Hessian evaluated at the solution of interest. After tedious calculations, the Hessian of (1) with respect to the Givens angles is interestingly found to be diagonal (discussion in Appendix). Thus, the maximum eigenvalue is immediately obtained as the maximum diagonal entry. Using the chain rule, the Hessian can be expressed as

$$\frac{\partial^2 J}{\partial \bar{\theta}_{pq}^2} = \left(\frac{\partial \bar{\mathbf{r}}}{\partial \bar{\theta}_{pq}}\right)^T \frac{\partial^2 J}{\partial \bar{\mathbf{r}}^2} \left(\frac{\partial \bar{\mathbf{r}}}{\partial \bar{\theta}_{pq}}\right) + \frac{\partial J}{\partial \bar{\mathbf{r}}} \frac{\partial^2 \bar{\mathbf{r}}}{\partial \bar{\theta}_{pq}^2} \tag{6}$$

To obtain (6), we defined $\mathbf{R} = \bar{\mathbf{R}} \mathbf{Q}_{\mathbf{x}}^T$, where $\mathbf{Q}_{\mathbf{x}}$ is the orthonormal eigenvector matrix of $\boldsymbol{\Sigma}$ in descending order and $\bar{\mathbf{r}} = vec(\bar{\mathbf{R}})$. Evaluating (6) at the solution of interest yields (at this point $\bar{\mathbf{R}} = \mathbf{P}$, $\mathbf{P}$ being a permutation matrix, i.e. all Givens angles $\bar{\theta}_{pq}$ are integer multiples of $\pi/2$).

$$\left.\frac{\partial^2 J}{\partial \bar{\theta}_{pq}^2}\right|_{\bar{\boldsymbol{\theta}}_*} = 2(\gamma_p - \gamma_q)(\bar{\lambda}_q - \bar{\lambda}_p) \tag{7}$$

where the eigenvalues are reordered using $\bar{\boldsymbol{\lambda}} = \mathbf{P}\boldsymbol{\lambda}$, where $\boldsymbol{\lambda}$ is the vector of eigenvalues in descending order. Thus, the result in the theorem is obtained. □

SIPEX-L: This result immediately applies to the SIPEX-L algorithm if we regard the term $\eta / 1 + \mathbf{G}_k^T \mathbf{G}_k$ as the effective step size of this algorithm.

$$\eta_k \le \frac{(1 + \mathbf{G}_k^T \mathbf{G}_k)}{\max\limits_{p,q} \left| (\gamma_p - \gamma_q)(\bar{\lambda}_q - \bar{\lambda}_p) \right|} \tag{8}$$

In practice, the eigenvalues can be approximated by output variances. In addition, γ can be manipulated to facilitate stability. In particular, selecting γ close to each other, we can increase the upper bound for the step size. Thus, we can traverse the transition region in the weight space faster to converge to the solution.

SIPEX-H: Assume that the weights are sufficiently near a solution so the performance surface is approximated by a quadratic function around this solution. Let this solution be $\boldsymbol{\theta}_*$. The gradient is accurately approximated by a first order Taylor expansion at the solution.

$$\mathbf{G}_k = \mathbf{G}_* + \mathbf{H}_*(\boldsymbol{\theta}_k - \boldsymbol{\theta}_*) + O((\boldsymbol{\theta}_k - \boldsymbol{\theta}_*)^3) \tag{9}$$

Here $\mathbf{G}_* = \mathbf{0}$ and the higher order terms are negligible. Subtracting $\boldsymbol{\theta}_*$ from both sides of the SIPEX-H update equation in (3), assuming $\tilde{\mathbf{H}}_k \approx \tilde{\mathbf{H}}_*$, and defining $\boldsymbol{\varepsilon}_k = \boldsymbol{\theta}_k - \boldsymbol{\theta}_*$ we get

$$\boldsymbol{\varepsilon}_{k+1} = \boldsymbol{\varepsilon}_k - \eta \tilde{\mathbf{H}}_*^{-1} \mathbf{H}_* \boldsymbol{\varepsilon}_k = (\mathbf{I} - \eta \tilde{\mathbf{H}}_*^{-1} \mathbf{H}_*) \boldsymbol{\varepsilon}_k \tag{10}$$

We mentioned that the Hessian becomes diagonal at the stationary points. Therefore, in the vicinity of a stationary point, the error dynamics are $\boldsymbol{\varepsilon}_{k+1} = (1 - \eta)\mathbf{I}\boldsymbol{\varepsilon}_k$. Choosing $\eta = 1$ results in immediate convergence if the surface is actually quadratic, which is the motivation behind SIPEX-H. In practice, this is not the case, so the error dynamics are roughly governed by $(\mathbf{I} - \eta \tilde{\mathbf{H}}_k^{-1} \mathbf{H}_k)$. For this error to decay to zero, according to the Lyapunov theory, the eigenvalues of this matrix must be in the unit circle. Let the Hessian be $\mathbf{H}_k = [h_{ij}]$. Then, $(\mathbf{I} - \eta \tilde{\mathbf{H}}_k^{-1} \mathbf{H}_k)$ is

$$\begin{bmatrix} 1-\eta & -\eta \frac{h_{12}}{|h_{11}|} & \cdots & \cdots & -\eta \frac{h_{1m}}{|h_{11}|} \\ \vdots & \ddots & & & \vdots \\ -\eta \frac{h_{i1}}{|h_{ii}|} & \cdots & 1-\eta & \cdots & -\eta \frac{h_{im}}{|h_{ii}|} \\ \vdots & & & \ddots & \vdots \\ -\eta \frac{h_{m1}}{|h_{mm}|} & \cdots & \cdots & -\eta \frac{h_{m(m-1)}}{|h_{mm}|} & 1-\eta \end{bmatrix} \tag{11}$$

According to Gershgorin's theorem, the eigenvalues of an mxm matrix are always inside the union of m circular sets in the complex plane, where the centers of these circles are the diagonal elements of the matrix and the radii are the sum of the absolute values of off-diagonal elements in the corresponding row [5]. For (11), all circles are centered at (1-η) and the radius of the union set is $\max_i \eta \sum_{j=1, j\neq i}^{m} \left| h_{ij} / h_{ii} \right|$. This radius is equivalently $\eta\left(\left\| \tilde{\mathbf{H}}_k^{-1} \mathbf{H}_k \right\|_\infty - 1 \right)$, where $\|\cdot\|_\infty$ is the L_∞ matrix norm. Therefore, a sufficient stability condition for SIPEX-H reduces to $\left\| \tilde{\mathbf{H}}_k^{-1} \mathbf{H}_k \right\|_\infty < 2$ and $0 < \eta < 2 / \left\| \tilde{\mathbf{H}}_k^{-1} \mathbf{H}_k \right\|_\infty$. This bound only requires the evaluation of the Hessian entries. Hence, it is simpler to evaluate compared to a bound that based on the eigenvalues. In SIPEX γ could be used to manipulate the upper bound for stability.

The approach in SIPEX-H could, in fact, be applied to arbitrary topologies and criteria. The convergence results discussed above would still hold for these scenarios. The Hessian matrix, however, could be non-diagonal in general, in contrast to SIPEX.

5. NUMERICAL EXPERIMENTS

In order to test the convergence characteristics of the three SIPEX variants, we have performed a series of Monte Carlo simulations using 3-dimensional, synthetic, jointly Gaussian data. In each of the 200 simulations, the covariance matrix of the joint Gaussian density was selected randomly (as the product of a random matrix with its transpose), which usually resulted in highly large eigenspreads (on the order of tens and hundreds). All SIPEX variants were iterated using the same 10000 samples generated for each simulation and the convergence times and final eigenvector direction accuracies were recorded. We have defined the 10^o-convergence time and the 1^o-convergence time as the smallest iteration (sample) index such that all the estimated eigenvectors of the input covariance matrix are within the specified error threshold 99% of the time in the remaining iterations. As the reference eigenvectors, we have used the true eigenvectors of the true covariance matrix that is used to generate the data. The accuracy is measured by the root-mean-square (RMS) angle error, where the averaging is performed over 200 Monte Carlo simulations, three eigenvector angle errors, and the last 1000 iterations of each simulation.

For SIPEX-G,L,H, we used constant step sizes of $3\cdot10^{-2}$, $3\cdot10^{-2}$, and $3\cdot10^{-1}$, respectively. The results are shown in Figs. 1-3. The subplots in the right-columns show in detail the histograms of 10^o- and 1^o-convergence times using 100-length bins, and the RMS error histogram using 0.1^o-length bins. Fig. 1 shows that, almost surely, in 1000 iterations all three algorithms converge to within 10^o. However, SIPEX-G and SIPEX-H seem to be faster in this respect compared to SIPEX-L. In terms of the 1^o-

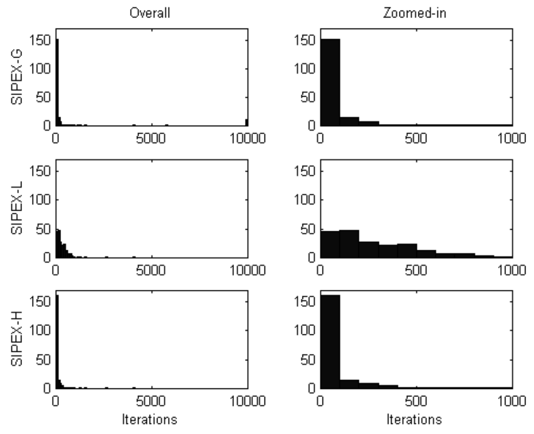

Figure 1. Histograms of 10^o-convergence times for SIPEX-G, SIPEX-L, and SIPEX-H (top to bottom).

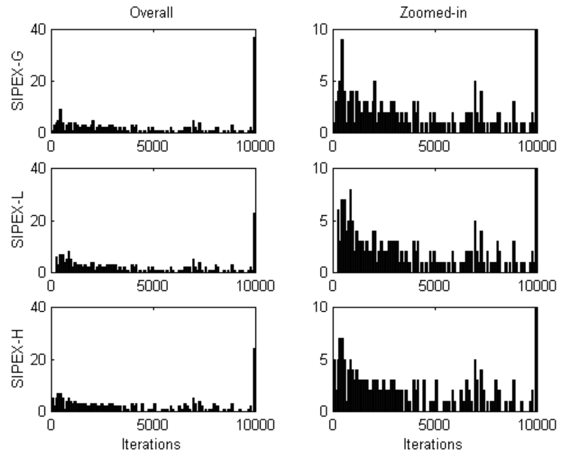

Figure 2. Histograms of 1^o-convergence times for SIPEX-G, SIPEX-L, and SIPEX-H (top to bottom).

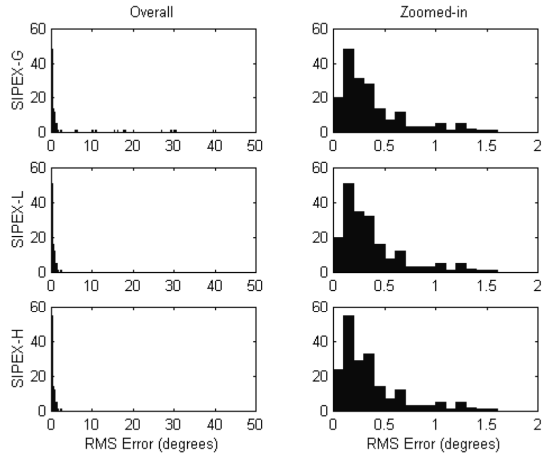

Figure 3. Histograms of RMS eigenvector direction angle errors in measured in degrees.

convergence times, all algorithms seem to be roughly equivalent. Likewise, the average final RMS angle errors of all algorithms are roughly identically distributed.

6. DISCUSSION

Recently, we have proposed SIPEX-G as a fast-converging and accurate PCA algorithm that identifies the principal components simultaneously, as opposed to the traditional methods that converge to the eigenvectors sequentially.

In this paper, we have proposed two variations for the SIPEX algorithm based on two different approximations of the Hessian matrix of the criterion. We have presented the stability conditions on the step size of all three algorithms. In Monte Carlo simulations (with $\boldsymbol{\gamma} = [n, n-1, ..., 2]^T$) whose results are not presented due to lack of space, we have observed that the original SIPEX-G and the new variant SIPEX-H performed similarly in terms of convergence speed. SIPEX-L, on the other hand, remained initially slower, due to the step size reducing effect of the normalization term. However, it took about the same time for it to converge to highly accurate solutions as the other two versions.

In simulations, it was observed that SIPEX-H could tolerate larger step sizes compared to the other two. This was expected from the theoretical stability analysis, since the upper bound of the SIPEX-H step size depends on the L_∞-norm of the diagonal-normalized Hessian matrix, whereas it depends on the maximum eigenvalue (L_2-norm) of the Hessian matrix for the others.

The analysis presented in this paper clarified the relationship between the free γ parameters of the criterion and the eigenstructure of the performance surface near the solutions. Therefore, it is possible to manipulate these free parameters during adaptation to speed up convergence and to modify the stability characteristics of the algorithm.

Acknowledgments: This work was partially supported by the NSF grant ECS-9900394.

7. APPENDIX

In this appendix, due to space limitations, we briefly discuss how the Hessian matrix is calculated. We start by expressing the Hessian as in (6) using the chain rule and a change of optimization variables from $\boldsymbol{\theta}$ to $\bar{\boldsymbol{\theta}}$. The latter simplifies computations since the PCA solutions are represented by angles that are multiples of $\pi/2$ in this setup. The first and second derivatives with respect to the rotation matrix entries, which appear in (6), are easily found to be

$$\frac{\partial J}{\partial \bar{\mathbf{R}}_{kl}} = 2\gamma_k \bar{\lambda}_l \bar{\mathbf{R}}_{kl}, \frac{\partial^2 J}{\partial \bar{\mathbf{R}}_{pq} \partial \bar{\mathbf{R}}_{kl}} = 2\gamma_k \bar{\lambda}_l \delta_{kp} \delta_{lq} \quad \text{(A.1)}$$

where δ_{ab} denotes the Kronecker-delta.

The first and second derivatives of the rotation matrix entries with respect to the angles are determined after substituting the values of $\bar{\theta}_{pq}$ (p=1,…,n-1;q=p+1,…,n), which are all multiples of $\pi/2$. This leads to highly sparse matrices; therefore, most of these derivatives are zeros. Consequently, the vector matrix multiplications in (6), which can be written as summations of products of the entries of the considered gradients and Hessians, generate only a few non-zero terms. These terms, when added together, show that the Hessian matrix of interest is diagonal. In mathematical terms, we find that

$$\left. \frac{\partial^2 J}{\partial \bar{\theta}_{rs} \partial \bar{\theta}_{pq}} \right|_{\bar{\boldsymbol{\theta}}_*} = 2(\gamma_p - \gamma_q)(\bar{\lambda}_q - \bar{\lambda}_p)\delta_{pr}\delta_{qs} \quad \text{(A.2)}$$

where $\bar{\boldsymbol{\theta}}_*$ is the vector consisting of $\bar{\theta}_{pq}$ values corresponding to the solution of interest.

8. REFERENCES

[1] E. Oja, *Subspace Methods for Pattern Recognition*, Wiley, New York, 1983.

[2] T. D. Sanger, “Optimal Unsupevised Learning in a Single Layer Linear Feedforward Neural Network,” Neural Networks, vol. 2, no. 6, pp. 459-473, 1989.

[3] L. Xu, “Least Mean Square Error Reconstruction Principle for Self-Organizing Neural-Nets,” Neural Networks, vol. 6, pp. 627-648, 1993.

[4] S. Y. Kung, K. I. Diamantaras, J.S. Taur, “Adaptive Principal Component Extraction (APEX) and Applications,” IEEE Transactions on Signal Processing, vol. 42, May 1994.

[5] G. Golub, C.V. Loan, *Matrix Computation*, John Hopkins University Press, Baltimore, MD, 1993.

[6] D. Erdogmus, Y.N. Rao, J.C. Principe, J. Zhao, K.E. Hild II, “Simultaneous Extraction of Principal Components Using Givens Rotations and Output Variances,” Proc. ICASSP'02, vol. 1, pp. 1069-1072, Orlando, Florida, May 2002.

[7] D. Erdogmus, Y.N. Rao, J.C. Principe, O. Fontenla-Romero, L. Vielva, “An Efficient, Robust, and Fast Converging Principal Components Extraction Algorithm: SIPEX-G,” Proc. EUSIPCO'02, vol. 2, pp. 335-338, Toulouse, France, Sep 2002.