Proceedings of the
44th IEEE Conference on Decision and Control, and
the European Control Conference 2005
Seville, Spain, December 12-15, 2005

Reliable Tracking Algorithms for Principal and Minor Eigenvector Computations

Markus Baumann, Uwe Helmke, Jonathan H. Manton

Abstract— **Many problems in control and signal processing require the tracking of certain eigenvectors of a time-varying matrix; the eigenvectors associated with the largest eigenvalues are called the principal eigenvectors and those with the smallest eigenvalues the minor eigenvectors. This paper presents a novel algorithm for tracking minor eigenvectors. One interesting feature, inherited from a recently proposed minor eigenvector flow upon which part of this work is based, is that the algorithm can be used also for tracking principal eigenvectors simply by changing the sign of the matrix whose eigenvectors are being tracked. The other key feature is that the algorithm has a guaranteed accuracy. Indeed, the algorithm is based on a flow which can be interpreted as the combination of a homotopy method and a Newton method, the purpose of the latter to compensate for discretisation errors.**

I. Introduction

In statistical analysis, the principal components of a covariance matrix C are projections of the data vectors on the directions of the principal eigenvectors of C. Therefore, the major task in principal component analysis (PCA) is the determination of the principal eigenvectors of the covariance matrix.

In this paper, we consider the more general task of determining the minor and principal eigenvectors of a matrix, which are the eigenvectors associated with the smallest and largest eigenvalues, respectively.

For a fixed matrix, the computation of the principal or minor eigenvectors is essentially a problem in numerical linear algebra. For time-varying matrices though, the problem takes on a new dimension as it becomes challenging to develop recursive algorithms which are computationally inexpensive yet are guaranteed to produce results accurate to within a prescribed tolerance. In fact, even though many algorithms have been developed, fueled by the large number of applications in control and signal processing [8], [9], [12], [13], [14], [15], [16], [17], [19], [21], we believe the algorithm proposed in this paper is the first to have a guaranteed accuracy.

Two ingredients are used in the development of the algorithm. The first is a recently proposed flow which converges to the principal or minor eigenvectors of a constant matrix [8]. Unlike all previously proposed principal or minor eigenvector flows, this flow evolves in Euclidean space, does not require any projection operators, yet remains stable even if the matrix is indefinite. This means that simply changing the sign of the matrix causes the same flow to converge to either the principal or minor eigenvectors.

The second ingredient is a modified homotopy method [2], [5], [6], the modification being to add a stabilizing term which makes the method stable even in the presence of discretisation errors. Indeed, this modification can be thought of as a Newton flow, always pulling the trajectory back on course. Although Newton flows have their disadvantages, such as inherent lack of global convergence and the need to invert the Hessian, in this particular case, these disadvantages are mollified. In tracking applications, global convergence is not important as long as it can be shown that the trajectory always remains in the domain of attraction. The need to invert the Hessian is overcome by first showing it suffices to invert only approximately the Hessian, and then deriving a suitable approximation which can be computed relatively cheaply.

The paper is organized as follows. The following section introduces a modified homotopy method for tracking a zero of a time varying cost function. Next, a cost function appropriate for minor eigenvector analysis is given. Combining these two ingredients leads to the proposed algorithm for tracking the minor eigenvectors of a time varying matrix. To reduce the computational complexity, a certain approximation to the algorithm is introduced. Simulation results confirm the correct operation of the algorithm.

II. The Newton flow and its discretization

In this section we consider the problem of finding a time-varying root of a smooth map $F : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$, $(x, t) \mapsto F(x, t)$, i.e. we want to determine a smooth curve $t \mapsto x_*(t)$, which satisfies

$$F(x_*(t), t) = 0$$

for all t. This problem arises in many important applications, including e.g. the inverse kinematics problem in robotics or parametric optimization tasks. A well-known approach for solving this problem is via homotopy methods, i.e. by solving an associated differential equation whose solutions track the roots of F. Specifically, we consider the implicit differential equation

$$DF(x, t)\dot{x} + \frac{\partial}{\partial t}F(x, t) = \mathcal{M}F(x, t), \qquad (1)$$

for an arbitrary linear map $\mathcal{M}$, whose eigenvalues all have negative real part. Here DF denotes the $n \times n$ matrix of

Research partially supported by DAAD PPP Australia, Germany, under grant D/0243869

M. Baumann and U. Helmke are with the Mathematisches Institut, Universität Würzburg,97074 Würzburg, Germany {baumann, helmke}@mathematik.uni-wuerzburg.de

J. Manton is with the Department of Information Engineering, Research School of Information Sciences and Engineering, The Australian National University Canberra, Australia j.manton@ieee.org

0-7803-9568-9/05/$20.00 ©2005 IEEE

partial derivatives of F with respect to the variable x. Under suitable invertibility conditions on DF, system (1) can thus be written in explicit form as

$$\dot{x} = DF(x,t)^{-1}\left(\mathcal{M}F(x,t) - \frac{\partial}{\partial t}F(x,t)\right). \qquad (2)$$

We refer to (1) as the (implicit) time-varying Newton Flow. The reason for this terminology is that in the time-invariant case and for $\mathcal{M} = -I$, equation (1) becomes equivalent to the classical Newton flow

$$\dot{x} = -DF(x)^{-1}F(x) \qquad (3)$$

for root finding of a nonlinear map F. The main reason for inserting $\mathcal{M}$ into the differential equation is to stabilize the dynamics around the desired solution $x_*(t)$. Note, that the use of the implicit Newton flow (1) for time-varying root finding is closely related to the dynamic inversion technique introduced by [6].

In order to obtain a reasonable tracking algorithm for the time-varying roots of F, one needs to discretize the differential equations. Of course, there are many possible choices. In [2] a stable discretization of (1) has been proposed using Euler step discretization. In the sequel, we need a slightly more refined result than in [2] that we now describe.

Thus we discretize (2) at equidistant times $t_k := kh$, for $k \in \mathbb{N}_0$ and a fixed stepsize $h > 0$. Depending on the stepsize we choose $\mathcal{M} := -\frac{1}{h}I$. This choice of $\mathcal{M}$ will be crucial for our subsequent analysis. The single-step Euler discretization of the time-varying Newton flow is then given as

$$\begin{aligned} x_{k+1} &= x_k - \\ DF(x_k,t_k)^{-1}&\left(F(x_k,t_k) + h\frac{\partial F}{\partial t}(x_k,t_k)\right). \end{aligned} \qquad (4)$$

In order to implement this algorithm one needs however an exact formula for the partial derivatives $\frac{\partial F}{\partial t}(x_k,t_k)$. Often this is a restriction, as the precise values may not be available or corrupted by noise. Thus one has to replace the partial derivative by suitable higher order Taylor approximations $H(x,t,h)$ of $\frac{\partial F}{\partial t}(x,t)$. To this end we derive an mth-order approximation formula for the partial derivative $\frac{\partial F}{\partial t}(x,t)$ by evaluating F at preceding points $(x,t),\dots,(x_k,t-mh)$, $m \in \mathbb{N}$.

Hence, we get the following list of mth-order approximations $H_m(x,t)$ for $\frac{\partial F}{\partial t}(x,t)$:

1) $H_0(x,t) = 0$
2) $H_1(x,t) = \frac{1}{h}\Big(F(x,t) - F(x,t-h)\Big)$
3) $H_2(x,t) = \frac{1}{2h}\Big(3F(x,t) - 4F(x,t-h) + F(x,t-2h)\Big)$
4) $H_3(x,t) = \frac{1}{30h}\Big(37F(x,t) - 45F(x,t-h) + 9F(x,t-2h) - F(x,t-3h)\Big)$

Using such approximations, equation (4) turns into

$$\begin{aligned} x_{k+1} &= x_k - \\ DF(x_k,t_k)^{-1}&\Big(F(x_k,t_k) + hH_m(x_k,t_k)\Big). \end{aligned} \qquad (5)$$

This is the main discrete time system that we will explore subsequently. The next theorem shows the stability of this update scheme.

Theorem 1: Let $F : \mathbb{R}^n \times \mathbb{R} \to \mathbb{R}^n$, $(x,t) \mapsto F(x,t)$ be C^2 in (x,t) and H_m as above. Let $t \mapsto x_*(t)$ be a continuous root of F, i.e. $F(x_*(t),t) = 0$ for all t. Assume there exist constants $c_1, c_2, c_3, c_4, c_5, c_6, c_7, R > 0$ such that

(i) $\|DF(x_*(t),t)\| \leq c_1$, $\|\frac{\partial}{\partial t}F(x_*(t),t)\| \leq c_2$, $\|DF(x_*(t),t)^{-1}\| \leq c_3$ for all $t > 0$,
(ii) $\|D^2F(x,t)\| \leq c_4$, $\|\frac{\partial^2}{\partial t^2}F(x,t)\| \leq c_5$, $\|\frac{\partial}{\partial t}DF(x,t)\| \leq c_6$ for all $t > 0$ and $x \in B_R(x_*(t))$.
(iii) $\|H_m(x,t) - \frac{\partial F}{\partial t}(x,t)\| \leq c_7 h$, for all $t > 0$ and $x \in B_R(x_*(t))$.

Then the following statements hold

1) There exist $0 < r < R$ and $c_8, c_9 > 0$ such that for $t \in \mathbb{R}$

$$\|x - x_*(t)\| \leq c_8\|F(x,t)\|, \qquad (6)$$

and

$$\|DF(x,t)^{-1}\| \leq c_9 \qquad (7)$$

where $x \in B_r(x_*(t))$. In particular, the root $x_*(t)$ of F is isolated, i.e. for any $t \in \mathbb{R}$ and any $x \in B_r(x_*(t))$ holds: $F(x,t) = 0$ if and only if $x = x_*(t)$.

2) There exists constants $c_{10}, c_{11} > 0$ such that the solution (x_k) of (5) satisfies

$$\begin{aligned} &\|x_{k+1} - x_*(t_{k+1})\| \leq \\ &c_{10}\|x_k - x_*(t_k)\|^2 + c_{11}h^2, \end{aligned} \qquad (8)$$

for $k \in \mathbb{N}_0$ with $x_k \in B_r(x_*(t_k))$.

3) Let $c > 0$ be a constant and $h \leq \frac{c}{c^2 c_{10} + c_{11}}$. For any initial condition x_0 with $\|x_0 - x_*(0)\| < ch$ we have

$$\|x_k - x_*(t_k)\| \leq ch$$

for all $k \in \mathbb{N}_0$. Thus the update scheme (5) is well defined and produces estimates for $x_*(t_k)$, whose accuracy can be controlled by the step size.

Proof: [Sketch of proof]

1) Due to condition (i) and (ii), there exists a $r > 0$ such that for any $t \in \mathbb{R}$ the map $x \mapsto F(x,t)$ is a local diffeomorphism on $B_r(x_*(t))$. This shows (6) and (7)

2) Use Taylor's Theorem to consider

$$\begin{aligned} F(x_{k+1},t_{k+1}) &= F(x_k,t_k) + \\ \frac{\partial F}{\partial t}(x_k,t_k)h &+ DF(x_k,t_k)(x_{k+1} - x_k) + R, \end{aligned} \qquad (9)$$

where R satisfies $\|R\| \leq c_4\Delta^2 + c_5h^2 + c_6\Delta h$ for $\Delta := \|x_{k+1} - x_k\|$

Take the update scheme (5) to replace x_{k+1} in the above equation and obtain

$$F(x_{k+1},t_{k+1}) = -hH(x_k,t_k) + \frac{\partial F}{\partial t}(x_k,t_k)h + R.$$

Therefore

$$\|F(x_{k+1},t_{k+1})\| \leq \left(c_4 + \frac{c_6}{2}\right)\Delta^2 + \left(c_5 + \frac{c_6}{2} + c_7\right)h^2 \qquad (10)$$

Noting (5), it is easily shown that

$$\Delta \leq c_9(\|F(x_k,t_k)\| + h(c_2 + c_7 h))/(1 - hc_9c_6),$$

provided $h < \frac{1}{c_6c_9}$. Plugging this result into (10) yields

$$\|F(x_{k+1},t_{k+1})\| \leq a\|F(x_k,t_k)\|^2 + bh\|F(x_k,t_k)\| + ch^2 \qquad (11)$$

for some constants $a, b, c > 0$. From this estimate (8) easily follows.

3) Suppose

$$\|x_k - x_*(t_k)\| \leq ch$$

holds for some k. By the estimate (8) then

$$\|x_{k+1} - x_*(t_{k+1})\| \leq c_{10}\|x_k - x_*(t_k)\|^2 + c_{11}h^2 \leq c_{10}c^2h^2 + c_{11}h^2 \leq ch, \quad (12)$$

for $h < \frac{c}{c_{10}c^2 + c_{11}}$. ■

Standard estimates for the discretization error imply the above result 3) only for a finite number of iterates x_k. The new interesting feature here is that the error estimate can be guaranteed to hold for all $k \in \mathbb{N}_0$. This is due to the fact, that we have chosen $\mathcal{M} = -\frac{1}{h}I$. Without that choice we would not be able to prove a similar estimate.

Remark 1: Assume, one has an approximation $G(x,t)$ for $DF(x,t)^{-1}$ such that for some $c, R > 0$,

$$\|(I - G(x,t)DF(x,t))(F(x,t) - H(x,t))\| \leq c\|F(x,t)\|, \qquad (13)$$

for all $t \in \mathbb{R}$ and $x \in B_R(x_*(t))$.

Then it can be shown that under the conditions of the previous theorem, the sequence

$$x_{k+1} = x_k + hG(x_k,t_k)\left(\mathcal{M}F(x_k,t_k) - H(x_k,t_k)\right). \qquad (14)$$

has a similar stability property as the sequence (5). Particularly, for $c > 0$ holds for all $k \in \mathbb{N}$

$$\|x_k - x_*(t_k)\| \leq ch,$$

provided $\|x_0 - x_*(0)\| < ch$ and $h > 0$ is sufficiently small.

As it will turn out, exploiting this fact leads to algorithms, which are computationally much cheaper.

III. Time-varying minor and principal component analysis

In this section, we consider the task of determining for $t \in \mathbb{R}$ the minor and principal eigenvectors of a time-varying symmetric matrix $A(t) \in \mathbb{R}^{n\times n}$.

In [8] a method was introduced, which was able to extract the minor and principal eigenvectors of a constant matrix $A \in \mathbb{R}^{n\times n}$. This was achieved by finding the minimum or maximum of a suitable cost function. The minimization of this function leads to a method, which determines the minor eigenvectors, hence the eigenvectors associated to the smallest eigenvalues. We follow this approach in order to derive a minor/principal eigenvector flow for time-varying matrices $A(t)$.

In principal component analysis for time-varying data one is concerned with the associated eigenvector estimation task for a time-varying covariance matrix $A(t)$. There are at least two interpretations.

1) Let $x(\tau) \in \mathbb{R}^n$ a curve of data points for $\tau \geq 0$. Then define for $t \geq 0$

$$A(t) := \frac{1}{t}\int_0^t x(\tau)x(\tau)^T d\tau.$$

2) Let $x_1(t), ..., x_m(t) \in \mathbb{R}^n$ curves of data points for $t \geq 0$. Then

$$A(t) := \frac{1}{m}\sum_{i=1}^m x_i(t)x(t)^T.$$

The subsequent analysis will not depend on any such statistical interpretation of $A(t)$. Thus, the developed eigenvector tracking techniques are also suitable to compute the time-varying principal components of any covariance matrix $A(t)$.

Consider the smooth cost function $f : \mathbb{R}^{n\times p} \times \mathbb{R} \to \mathbb{R}$

$$f(X,t) = \frac{1}{2}\mathrm{tr}\left(A(t)XNX^T\right) + \frac{\mu}{4}\|N - X^TX\|^{,} \qquad (15)$$

where $N \in \mathbb{R}^{p\times p}$, $\mu \in \mathbb{R}$. To ensure that the extrema $X_*(t)$ of f correspond to the minor/principal eigenvectors of $A(t)$, we make the following assumptions, cf. [8].

A1 The scalar μ is strictly positive
A2 $A(t) \in \mathbb{R}^{n\times n}$ is symmetric for all $t \in \mathbb{R}$
A3 $N = \mathrm{diag}\,(n_1, ..., n_p) \in \mathbb{R}^{p\times p}$ with $n_1 > ... > n_p > 0$.
A4 $A(t)$ has distinct eigenvalues for all $t \in \mathbb{R}$
A5 μ does not equal any eigenvalue of $A(t)$ for all $t \in \mathbb{R}$

Assumption A1 implies that the cost function (15) has compact sublevel sets and therefore a global minimum $X_*(t)$ exists for any $t \in \mathbb{R}$. Furthermore, for fixed $t \in \mathbb{R}$, assumptions A1-A3 imply that each column of a critical point $X_* = [x_1...x_p]$ of f is either the null-vector or an eigenvector of $A(t)$ with eigenvalue λ_i, $i = 1, ..., p$. The following lemma holds.

Lemma 1: [8] Assume A1-A5 hold. For $t \in \mathbb{R}$, let $\lambda_1(t) < ... < \lambda_p(t) < \lambda_{p+1}(t) \leq \cdots \leq \lambda_n(t)$ be the eigenvalues of $A(t)$ in ascending order and let $v_1(t), ..., v_n(t)$ be the corresponding normalized eigenvectors. Then for any $t \in \mathbb{R}$, $X_*(t) = [x_1(t) \; ... \; x_p(t)]$ is a local minimum of $f(X,t)$ if and only if

$$x_i(t) = \pm\gamma_i(t)v_i(t) \qquad (16)$$

where

$$\gamma_i(t) = \begin{cases} \sqrt{N_{ii}(1 - \lambda_i(t)/\mu)} & \text{if } \lambda_i(t) < \mu \\ 0 & \text{otherwise} \end{cases}$$

for $i = 1, ..., p$.

Let M denote an upper bound for $\lambda_i(t)$ for all $t \in \mathbb{R}$ and $i = 1, ..., p$. If $\mu > M$, then the previous lemma shows, that for any $t \in \mathbb{R}$, $X_* = [x_1 ... x_p]$ is a minimum of $f(X, t)$ if and only if its columns are the non-trivial eigenvectors of $A(t)$, which correspond to the p smallest eigenvalues. Thus the global minimum fo f gives the minor eigenvectors of A. In contrast, the maxima of f correspond to the principal eigenvectors of A, i.e. the minima of the cost function f_- obtained by replacing A by $-A$. Thus, by replacing A by $-A$ in the subsequent formulas, all results about minor eigenvectors are immediately reformulated into equivalent results about principal eigenvectors. Thus, from now on, we restrict ourselves to the minor eigenvector case. This duality between MCA and PCA does not hold for the previously proposed cost functions for principal eigenvector analysis and motivates our choice of the specific cost function (15).

Proposition 1: Let $t \mapsto A(t)$ be a smooth matrix valued function satisfying A1-A5. Assume that $\|A(t)\|$, $\|\dot{A}(t)\|$ and $|\lambda_i(t) - \lambda_j(t)|^{-1}$ for $i \neq j$ are uniformly bounded on $\mathbb{R}$. Then there exists a smooth family $t \mapsto X_*(t) \in \mathbb{R}^{n \times p}$, which is a continuous isolated minimum of $f(X, t)$. Moreover, if $\mu > M$, then the rows of $X_*(t)$ are the p non-trivial eigenvectors, associated to the p smallest eigenvalues.

IV. Minor Component tracking

Let $A(t)$ and f as in the previous section, where $X_*(t)$ denote the time-varying minimum of f, i.e. $f(X_*(t), t)$ is the minimum of f for any t. We want to track this minimum by using the time-varying Newton Flow. Hence, we need to determine the "X-gradient" and "X-Hessian" of f, i.e. we consider for fixed t the function $X \mapsto f(X, t)$ and determine its gradient, which is given as

$$\nabla f(X, t) = A(t)XN - \mu XN + \mu XX^T X. \quad (17)$$

Moreover, the Hessian (w.r.t. X) acts on arbitrary $\xi \in \mathbb{R}^{n \times p}$ in the following manner

$$H_f(X, t) \cdot \xi = \\ A(t)\xi N - \mu \xi N + \mu(\xi X^T X + X\xi^T X + XX^T \xi). \quad (18)$$

In order to use the Newton Flow for the tracking of the root of the gradient, we have to compute an inverse of the Hessian operator.

In the special case $p = 1$, X and ξ are vectors of length n and (18) can be rewritten as

$$H_f(X, t) \cdot \xi = A(t)\xi - \mu\xi + \mu(X^T X + 2XX^T)\xi, \quad (19)$$

where we choose $N = 1$. Hence, the matrix representation of the Hessian is just given as $A(t) - \mu I_n + \mu(X^T X + 2XX^T)$.

But if $p > 1$, one needs more effort to determine the Hessian operator and we show two different approaches.

A. Vectorizing the matrix differential equation

In this paragraph, we vectorize ξ and $\nabla f(X, t)$ by using the vec operation [7] and computing a matrix representation for H_f by employing the Kronecker product, denoted by $\otimes$. Thus

$$H_f(X, t) = N \otimes (A(t) - \mu I_n) + \\ \mu(X^T X \otimes I_n + (X^T \otimes X)\pi_T + I_p \otimes XX^T), \quad (20)$$

where π_T is such that vectorized matrices are mapped onto vectors, which equal the vectorized transposed matrix, i.e.

$$\pi_T \cdot \mathrm{vec(X)} = \mathrm{vec(X^T)}, \quad \mathrm{X \in \mathbb{R}^{n \times p}}.$$

Therefore, π_T is a permutation matrix, which is for $p = 1$ given as $\pi_T = I_n$.

Once having determined the Hessian, we arrive at the tracking algorithm for vector-valued X_k, $k \in \mathbb{N}$.

Algorithm 1: Choose a starting point X_0 close to the exact minimum $X_*(0)$ of $f(\cdot, 0)$, $t_k := kh$. For $k \in \mathbb{N}$, the new point X_{k+1}, which approximates the minimum of f at t_{k+1} is determined by

$$X_{k+1} = \\ X_k - H_f^{-1}\left(\nabla f(X_k, t_k) + hH_m(X_k, t_k)\right) \quad (21)$$

Here H_m denotes an mth-order approximation to $\frac{\partial}{\partial t}\nabla f(X_k, t_k)$. Some valid choices for this are given in section II.

The following proposition gives necessary conditions for the applicability of this algorithm.

Proposition 2: Let $f(X, t)$ as defined in (15) and assume A1-A5. Let further $t \mapsto A(t)$ be a smooth function and let $\|A(t)\|$, $\|\dot{A}(t)\|$, $\|\ddot{A}(t)\|$, $|\lambda_i(t) - \lambda_j(t)|^{-1}$ be uniformly bounded on $\mathbb{R}$ for $i \neq j$ and $i \leq p \vee j \leq p$ and let $\mu > \sup\{\lambda_i(t)| \ 1 \leq i \leq p, t \in \mathbb{R}\}$.

Then there exists a continuous isolated root $X_*(t)$ of $\nabla f(X, t)$, whose columns are the eigenvectors of $A(t)$, associated with the p smallest values of the eigenvalues.

Moreover, for any sufficiently small step-size h and $c > 0$ with $\|X_0 - X_*(0)\| \leq ch$, the sequence X_k defined by Algorithm 1 satisfies for all $k \in \mathbb{N}$

$$\|X_k - X_*(t_k)\| \leq ch. \quad (22)$$

Remark 2: Note that the matrix representation of $H_f(X, t)$ is of dimension $np \times np$, which shows, that standard way to employ the Newton flow by inverting this matrix might not be practical for large n and p. This motivates to look for an other way to invert H_f.

B. Approximately solving the implicit differential equation

In order to derive a practical inversion formula for H_f,we consider the following equation for ξ

$$H_f(X, t)\xi := \\ A(t)\xi N - \mu\xi N + \mu(\xi X^T X + X\xi^T X + XX^T\xi) = R, \quad (23)$$

where $R \in \mathbb{R}^{n\times p}$. If $X_*(t)$ denotes a minimum of f, we determine $Q \in O(n)$ such that $QD = X_*$, where $D \in \mathbb{R}^{n\times p}$ satisfying $D_{ij} = 0$ for $i \neq j$. Such matrices Q, D exist, as the columns of X_* are pairwise orthogonal. Hence, $D_{ii} = \pm\|x_i\|$, and $q_i = x_i/\|x_i\|$ where x_i, q_i denotes the ith column of X_*, Q, respectively for $i = 1, ..., p$.

The remaining $n-p$ columns Y of Q have to be chosen such that Q is an orthogonal matrix, and the columns of Y span the eigenspace of the $n-p$ principal eigenvectors.

This is a quite restrictive assumption, however, if the $n-p$ principal eigenvectors are all equal, then $Y \in \mathbb{R}^{n\times(n-p)}$ can be any matrix with orthonormal columns such that $X^TY = 0$. Note further, that in the case of $p = n$, this assumption is not a restriction, either.

By defining $G := Q^T\xi$, we can rewrite the above equation as

$$\begin{aligned}(A(t) - \mu I_n)QGN + \mu(QGD^TQ^TQD +\\ QDG^TQ^TQD + QDD^TQ^TQG) = R\end{aligned} \tag{24}$$

Multiply both sides with Q^T. Then

$$\begin{aligned}Q^T(A(t) - \mu I_n)QGN +\\ \mu Q^T(QGD^TD + QDG^TD + QDD^TG) = Q^TR,\end{aligned} \tag{25}$$

which is equivalent to

$$\begin{aligned}Q^T(A(t) - \mu I_n)QGN +\\ \mu(GD^TD + DG^TD + DD^TG) = Q^TR\end{aligned} \tag{26}$$

This is a linear equation which we want to solve for G. As we have assumed that X_* is a minimum of $f(\cdot, t)$, the matrix $Q^T(A(t) - \mu I_n)Q = Q^TA(t)Q - \mu I_n$ is diagonal with distinct eigenvalues. Let $K \in \mathbb{R}^{n\times n}$ denote this diagonal matrix. Thus, we can solve equation 26 by considering the entries on position (i, j) and (j, i) for $1 \leq i, j \leq p$. We have

$$c_1G_{ij} + c_2G_{ji} = (Q^TR)_{ij}$$

and

$$c_3G_{ij} + c_4G_{ij} = (Q^TR)_{ji}$$

Here $c_1 = K_{ii}N_{jj} + \mu(D_{jj}^2 + D_{ii}^2)$, $c_2 = D_{ii}D_{jj}$, $c_3 = K_{jj}N_{ii} + \mu(D_{ii}^2 + D_{jj}^2)$, $c_2 = D_{jj}D_{ii}$.

If $i > p$ then $D_{i,j} = 0$ for all $j = 1, ..., p$. Hence, $(DG^TD)_{ij} = 0$ and we have only to consider one equation to determine G_{ij}:

$$c_1G_{ij} = (Q^TR)_{ij}$$

Therefore, we get the following formula for G:

$$G_{ij} = \begin{cases} \left((Q^TR)_{ij} - \frac{c_2}{c_3}(Q^TR)_{ji}\right)\left(c_1 - \frac{c_2c_4}{c_3}\right)^{-1} \\ \qquad \text{for } 1 \leq i, j \leq p \\ \\ \frac{(Q^TR)_{ij}}{c_1}, \text{ for } 1 \leq j \leq p, \ p+1 \leq i \leq n \end{cases} \tag{27}$$

Noting that $\xi = QG$ shows that we have now found an explicit form of (23).

If X is not the exact minimum of f, then $Q^T(A(t) - \mu I_n)Q$ is not diagonal and hence, K is only approximation for that term. However, using this approximation, one obtains a similar linear equation, whose solutions in G are also given by (27). Moreover, the approximation leads to results, which are not far away from the exact solution, as the original system is well-conditioned and the approximation is of order $c\|X - X_*(t)\|$, where $X_*(t)$ denotes the locally unique minimum of f for $t \in \mathbb{R}$. This leads to the following tracking algorithm. Note that the implementation of this algorithm is considerably cheaper than the previous one, as there is no need to compute the exact inverse of the Hessian.

Algorithm 2: 1. Choose a starting point X_0 close to the exact minimum $X_*(0)$ of $f(\cdot, 0)$ and use a sufficiently small stepsize h and $t_k := kh$.
2. For $k \in \mathbb{N}$, suppose that X_k is given. Choose a matrix $Q_k \in \mathbb{R}^{n\times n}$, whose first p columns result from the normalization of the columns of X_k. The rest $n-p$ normalized columns of Q have to be chosen such that they are normalized and span the eigenspace of the $n-p$ principal eigenvectors.
3. The new point X_{k+1}, which approximates the minimum of f at t_{k+1} is defined by

$$X_{k+1} = X_k - Q_kG, \tag{28}$$

where $G \in \mathbb{R}^{n\times p}$ is as defined in (27) for $R = \nabla f(X_k, t_k) + hH_m(X_k, t_k)$. Here H_m denotes the mth-order approximation to $\frac{\partial}{\partial t}\nabla f(X_k, t_k)$.

V. Numerical results

All simulations were performed in Matlab. We used step size $h = 0.05$, $n = 10$, $p = 3$ and $A(t) = \Theta(t)K(t)\Theta(t)^T$, where $K(t) = \operatorname{diag}(a_1\sin(t), ..., a_{10}\sin(10t))$, $\Theta(t) = R^T\begin{bmatrix}\cos(t) & \sin(t) & 0\\ -\sin(t) & \cos(t) & 0\\ 0 & 0 & I_8\end{bmatrix}R$. Here, $R \in O(10)$ is a fixed random orthogonal matrix and $a_i := 2.5i$ for $i = 1, ..., 10$. We moreover set $N = \operatorname{diag}(p, ..., 1)$.

In the first simulation, we check the tracking ability of algorithm 1 and we used approximations H_m for $\frac{\partial}{\partial t}\nabla f$ of order 2 (cf. section 2). Figure 1 shows the computed (dashed) and exact (solid) 3 minor time-varying eigenvalues. As it can be seen in the corresponding error plot (Fig 2.), where $\|X_k - X_*(t_k)\|$ is depicted, we did not use perfect initial conditions but the computed values converged fast towards the exact solution.

As we have seen in the previous section, it is of much less effort to compute an approximation instead of the exact inverse of the Hessian H_f, which, however, is not practical in the general case. Thus, we replaced in the simulation $K(t)$ by $\tilde{K}(t) = \operatorname{diag}(a_1\sin(t), ..., a_3\sin(3t), a_4\sin(4t)), ..., a_4\sin(4t) \in \mathbb{R}^{10\times 10}$.

Table I shows the mean error of both algorithms (21) and (28) for different choices of $H_m(x, t)$. Here the mean error

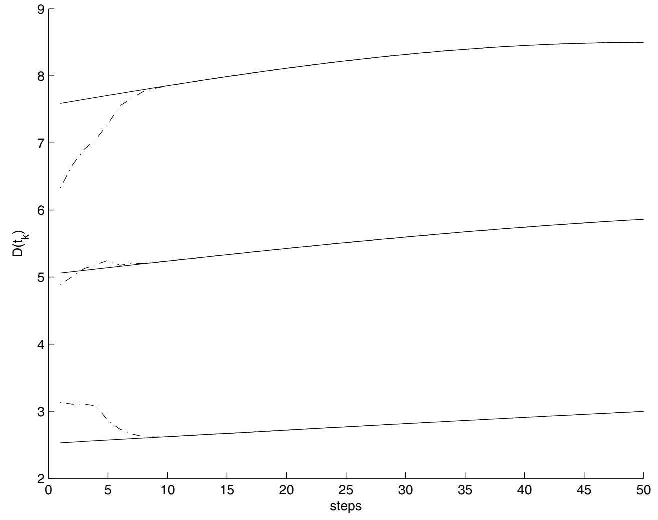

Fig. 1. The evolution of the minor eigenvalues. Solid: exact, dotted: computed values.

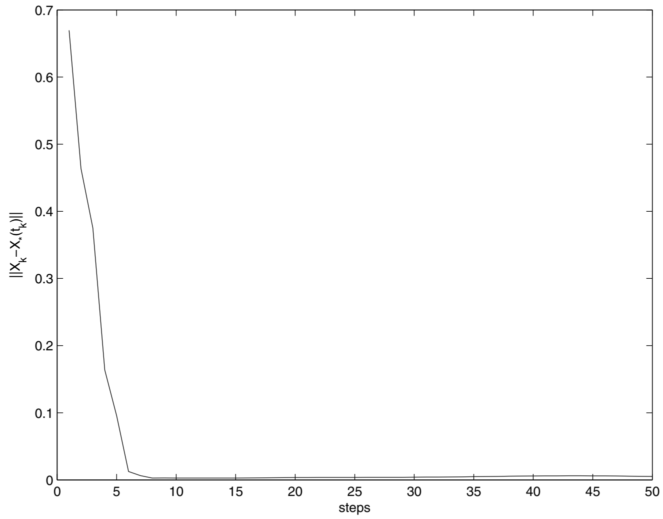

Fig. 2. The error plot, corresponding to Figure 1.

TABLE I

THE MEAN ERROR OF THE TWO ALGORITHMS, COMPUTED FOR DIFFERENT ORDER APPROXIMATIONS OF $\frac{\partial}{\partial t}\nabla f(X_k, t_k)$

Order m	Mean error algorithm 1	Mean error algorithm 2
0	$5.2 \cdot 10^{-2}$	$5.2 \cdot 10^{-2}$
1	$7.2 \cdot 10^{-3}$	$7.0 \cdot 10^{-3}$
2	$2.9 \cdot 10^{-3}$	$2.8 \cdot 10^{-3}$
3	$3.2 \cdot 10^{-3}$	$2.9 \cdot 10^{-3}$

is given by $\frac{1}{N}\sum_{i=1}^{N} \|X_k - X_*(t_k)\|$, where N denotes the number of steps.

Hence, using approximations for $\frac{\partial}{\partial t}F(X_k, t_k)$ of order $m > 1$ significantly improves the quality of the results in both algorithms. Note further, that the accuracy of the second algorithm is comparable to the accuracy of algorithm 1, although it only uses an approximatively inverted Hessian.

VI. CONCLUSION

This paper proposed a novel algorithm for tracking the minor eigenvectors of a time-varying symmetric matrix. Since there is no requirement for the matrix to be positive definite, the same algorithm can be used for tracking the principal eigenvectors simply by changing the sign of the matrix. The other key feature of the algorithm is that it has a guaranteed accuracy. Since the algorithm requires the inversion of a particular matrix, a suitable approximation was introduced to reduce the computational complexity. This reduced-complexity tracking algorithm also exhibits good performance.

REFERENCES

[1] M. Baumann, U. Helmke, Singular value decomposition of time-varying matrices, *Future Generation Computer Systems*, 19:353-361, 2003,

[2] M. Baumann, U. Helmke, C. Lageman, Newton-type algorithms for time-varying pose estimation, *Proceedings of the ISSNIP*, CD-ROM, 2004.

[3] R.W. Brockett, Dynamical systems that sort lists, diagonalize matrices, and solve linear programming problems, *Linear Algebra Appl.*, 146:79-91, 1991.

[4] P. Comon, G.H. Golub, Tracking a few extreme singular values and vectors in signal processing, *Proceedings of the IEEE*, 78(8):1327-1343, 1990.

[5] D. Davidenko, On a new method of numerically integrating a system of nonlinear equations, *Dokl. Akad. Nauk SSSR*, 88:601-604, 1953.

[6] N.H. Getz, J. Marsden, Dynamic methods of polar decomposition and inversion of matrices, *Linear Algebra and its Applications* 258:311-343, 1997.

[7] R.A. Horn, C.R. Johnson *Topics in Matrix Analysis*, Cambridge University Press, Cambridge 1991.

[8] J. H. Manton, U. Helmke, I.M.Y. Mareels, A dual purpose principal and minor component flow, *Systems and Control Letters* 54:759-769, 2005.

[9] Z. Kang, C. Chatterje, V.P. Roychowdhury, An adaptive quasi-Newton algorithm for eigensubspace estimation, *IEEE Transactions on Sig. Proc.*, 48(12):3328-3333, 2000.

[10] T. Kato, *Perturbation Theory for Linear Operators*, Springer-Verlag Berlin, 1966.

[11] G. Mathev, V.U. Reddy, S. Dasgupta, Adaptive estimation of eigenspace, *IEEE Transactions on Sig. Proc.*, 43(2):401-411, 1995.

[12] A.J. Laub, M.T. Heath, C.C. Paige, R.C. Ward, Computation of system balancing transformations and other applications of simultaneous diagonalization algorithms, *IEEE Transactions Automatic Control*, 32:115-121, 1987.

[13] E. Oja, H. Ogawa, J. Wangviwattana, Principal component analysis by homogenous neural networks, Part I: The weighted subspace criterion, *IEICE Transactions on Information and Systems*, 3:366-375, 1992.

[14] E. Oja, H. Ogawa, J. Wangviwattana, Principal component analysis by homogenous neural networks, Part II: Analysis and extensions of the learning algorithms, *IEICE Transactions on Information and Systems*, 3:376-382, 1992.

[15] D.J. Rabideau, Fast rank adaptive subspace tracking and applications, *IEEE Transactions on Signal Processing*, 44(9), 1996.

[16] E.C. Real, D.W. Tufts, J.W. Cooley, Two algorithms for fast approximative subspace tracking, *IEEE Transactions on Signal Processing*, 47(7), 1999.

[17] X. Wang, H.V. Poor, Blind multiuser detection: A subspace approach, *IEEE Transactions on Information Theory*, 44(2), 1998.

[18] L. Xu, Least mean square error recognition principle for self organizing neural nets, *Neural Networks*, 6, 1993.

[19] B. Yang, Projection approximation subspace tracking, *IEEE Transactions on Signal Processing*, 43(1), 1995.

[20] J.F. Yang, M. Kaveh, Adaptive eigenspace algorithms for direction or frequency estimation and tracking, *IEEE Transactions on Acoustics, Speech and Signal Processing*, 36(2), 1988.

[21] S. Yoshizawa, U. Helmke, K. Starkov, Convergence analysis for principal component flows, *International Journal of applied Mathematics and Computer Science*, 11, 2001.