DERIVATIVES OF GENERALIZED EIGEN SYSTEMS WITH APPLICATIONS

JAN DE LEEUW

ABSTRACT. In this note we compute derivatives of generalized eigenvalues and singular values, and of the corresponding eigen and singular vectors. These formulas have been around for a long time in various places, and we collect them mostly for reference purposes. We discuss some applications to multivariate analysis, and include R routines for doing the computations.

Date: December 22, 2007.

2000 *Mathematics Subject Classification.* 15A18,62H25.

Key words and phrases. Generalized Eigenvalue Decomposition, Generalized Singular Value Decomposition, Derivatives.

Contents

1. Introduction

The *generalized eigen problem* for a pair of real symmetric matrices A and B is defined by the system of equations

$$Ay = \lambda By, \tag{1a}$$

$$y'By = 1. \tag{1b}$$

Any solution (y, λ) of (1) is a *generalized eigen pair*, with y the *generalized eigen vector* and λ the *generalized eigen value*.

The *generalized singular value problem* for a triple of matrices P, Q and R is defined by the system of equations

$$Rz = \lambda Px, \tag{2a}$$

$$R'x = \gamma Qz, \tag{2b}$$

$$x'Px = 1, \tag{2c}$$

$$z'Qz = 1. \tag{2d}$$

The matrices P and Q are square and symmetric, and R is rectangular. Any solution (x, z, γ) of (2) is a *generalized singular triple*, with x the *generalized left singular vector*, z the *generalized right singular vector* and γ the *generalized singular value*.

Now suppose A and B, or P, Q and R, are matrix valued functions of a parameter θ. Then Equations (1) define y and λ implicitly as functions of θ. Similarly, Equations (2) define (x, z, γ) as implicit functions. Suppose $\hat{\theta}$ is a point where B is positive definite, the eigenvalue λ is real and simple, or a point where P and Q are positive definite and γ is simple. Also suppose the matrix-valued functions A and B, or P, Q, and R, are two times continuously differentiable at $\hat{\theta}$. Then the implicit function theorem guarantees that the eigen or singular values and vectors are differentiable at $\hat{\theta}$. Studying the partial derivatives goes back more than 100 years to Lord

Rayleigh's work on vibrating strings and Schrödinger's on quantum mechanics. The Rayleigh-Schrödinger perturbation theory entered the mathematical literature with the work of Rellich in 1937. See [Kato, 1976; Baumgärtel, 1985] for an historical overview, and for many generalizations.

Derivatives of eigenvalues and singular values are of major importance in physics, in engineering for studying structural optimization, in multivariate statistics, and in optimization theory. The literature, especially in physics and engineering, is gigantic and cannot possibly be reviewed in an article like this. We give some selected references in perturbation theory, structural engineering and applied mathematics that emphasize generalized inverses, like we do, and discuss computational implications [Sun, 1985; Chu, 1990; Sun, 1990; Wang and Wang, 1994; Wang, 1996; Andrew and Tan, 1998; McCartin, 2003; Zhang and Zhang, 2006]. They contain many additional references, as well as generalizations to multiple eigenvalues and asymmetric matrices.

In statistics the emphasis has always been on eigenvalue problems for symmetric matrices. In the papers, for example, by Kollo and Neudecker (1997a; 1997b) the algebra of differentiating eigenvalues and eigenvectors of symmetric matrices is treated, and the application of these results to the asymptotics for sample covariance and correlation matrices is discussed. They review earlier work of Girshick, Anderson, Waternaux, Fang, Krishnaiah, Fukikoshi, and Konishi. Also see Magnus and Neudecker [1998, Chapter 8]. Many other examples can be found in the multivariate analysis literature. We are interested, however, in treating the more general case of eigenvalue and singular value problems with positive definite weight matrices, where the weight matrices also depend on the parameters. And we will try to avoid various specialized matrix constructs such as Kronecker products and $\mathbf{vec}()$ operators.

In optimization theory there is a large and growing literature dealing with *eigenvalue optimization*, reviewed, for example, by Lewis and Overton [1996] and Lewis [2003]. This literature concentrates on the symmetric case, on second derivatives, and it emphasizes methods based on convexity and on directional derivatives or subdifferentials of non-smooth functions. Although our results can be used for eigenvalue and singular value optimization, they are clearly intended for statistical applications, more specifically as input for the delta method of computing standard errors and confidence regions [Van Der Vaart, 1998, Chapter 3].

2. Computing Partial Derivatives

2.1. **Generalized Eigen Value Problems.** We first compute the derivatives of the generalized eigenvalues. Remember we do this for a parameter value for which the eigenvalue is simple, i.e. has multiplicity one, and for which B is positive definite. Generalization to B positive semi-definite is fairly straightforward, but dealing with multiple eigenvalues requires a more delicate theory.

Differentiate (1a) to get

$$\frac{\partial A}{\partial\theta}y + A\frac{\partial y}{\partial\theta} = \lambda B\frac{\partial y}{\partial\theta} + \lambda\frac{\partial B}{\partial\theta}y + \frac{\partial\lambda}{\partial\theta}By, \tag{3a}$$

and collecting terms gives

$$(A - \lambda B)\frac{\partial y}{\partial\theta} = -(\frac{\partial A}{\partial\theta} - \lambda\frac{\partial B}{\partial\theta})y + \frac{\partial\lambda}{\partial\theta}By. \tag{3b}$$

Premultiplying both sides by y', and using (1a), gives the desired result

$$\frac{\partial\lambda}{\partial\theta} = y'(\frac{\partial A}{\partial\theta} - \lambda\frac{\partial B}{\partial\theta})y. \tag{4}$$

Of course Equation (4) is written for a single eigenvalue and for a single parameter. But by introducing suitable subscripts it extends immediately to multiple parameters and to more than one simple eigenvalue. This is done in the R code in Appendix A. Another easy extension would be if A depended on one set of parameters and B depended on another. This can be handled easily by concatenating the two sets of parameters and setting some of the partials equal to zero.

We now solve (3b) for the partials of the generalized eigen vector y. Suppose Y is a complete set of generalized eigen vectors, i.e. Y is a non-singular matrix with $AY = BY\Lambda$ and $Y'BY = I$. The diagonal matrix Λ contains the generalized eigen values, and y is one of the columns of Y, say column s. So $Y^{-1}y = e_s$, a unit vector with all elements equal to zero, except the s^{th} element which is equal to one.

For our computations we define the symmetric generalized inverse $(A - \lambda B)^- = Y(\Lambda - \lambda I)^+ Y'$, where $(\Lambda - \lambda I)^+$ is the Moore-Penrose inverse. Thus $(\Lambda - \lambda I)^+ e_s = (\Lambda - \lambda I)e_s = 0$. It is easy to check that $(A - \lambda B)^-$ satisfies the first two Penrose conditions

$$(A - \lambda B)^-(A - \lambda B)(A - \lambda B)^- = (A - \lambda B)^-, \tag{5a}$$

$$(A - \lambda B)(A - \lambda B)^-(A - \lambda B) = (A - \lambda B), \tag{5b}$$

but the third and fourth Penrose conditions are generally not satisfied. In fact

$$\begin{aligned}(A - \lambda B)^-(A - \lambda B) &= (A - \lambda B)(A - \lambda B)^- \\ &= Y(I - e_s e_s')Y^{-1} = I - yy'B,\end{aligned} \tag{6}$$

where the last equality follows from $Y'BY = I$, which gives $Y^{-1} = Y'B$. We also see that

$$\begin{aligned}(A - \lambda B)^- By &= Y(\Lambda - \lambda I)^+ Y'(Y')^{-1}Y^{-1}y = \\ &= Y(\Lambda - \lambda I)^+ e_s = 0.\end{aligned} \tag{7}$$

Now premultiply both sides of (3b) by $(A - \lambda B)^-$. Then, using both (6) and (7),

$$(I - yy'B)\frac{\partial y}{\partial \theta} = -(A - \lambda B)^-(\frac{\partial A}{\partial \theta} - \lambda \frac{\partial B}{\partial \theta})y. \tag{8}$$

Differentiate (1b) to get

$$y'\frac{\partial B}{\partial\theta}y+2y'B\frac{\partial y}{\partial\theta}=0, \tag{9a}$$

or

$$y'B\frac{\partial y}{\partial\theta}=-\frac{1}{2}y'\frac{\partial B}{\partial\theta}y. \tag{9b}$$

Using (9b) in (8) gives

$$\frac{\partial y}{\partial\theta}=-(A-\lambda B)^{-}(\frac{\partial A}{\partial\theta}-\lambda\frac{\partial B}{\partial\theta})y-\frac{1}{2}(y'\frac{\partial B}{\partial\theta}y)y. \tag{10}$$

2.1.1. *Symmetric Eigen Problems.* Equations (4) and (10) are our results for the first partials. We specialize them to ordinary eigenvalue problems, If B does not depend on θ then

$$\frac{\partial\lambda}{\partial\theta}=y'\frac{\partial A}{\partial\theta}y, \tag{11a}$$

$$\frac{\partial y}{\partial\theta}=-(A-\lambda B)^{-}\frac{\partial A}{\partial\theta}y. \tag{11b}$$

If in addition $B=I$, i.e. if we have an eigen problem for the matrix A, then

$$\frac{\partial y}{\partial\theta}=-(A-\lambda I)^{+}\frac{\partial A}{\partial\theta}y, \tag{12}$$

where the generalized inverse is now a Moore-Penrose inverse. These are the classical results that can be found in the books of Wilkinson [1965] or Magnus and Neudecker [1998]. Again they extend easily to parameter vectors θ and to considering several eigenvectors simultaneously.

2.2. **Second Partials.** Although second partial derivatives are not needed for computations of confidence intervals and standard errors, they can be used in bias correction and in eigenvalue optimization. If we differentiate (3b) again we find

$$\left(\frac{\partial A}{\partial \xi}-\lambda\frac{\partial B}{\partial \xi}\right)\frac{\partial y}{\partial \theta}+\left(\frac{\partial A}{\partial \theta}-\lambda\frac{\partial B}{\partial \theta}\right)\frac{\partial y}{\partial \xi}+(A-\lambda B)\frac{\partial^2 y}{\partial \xi\partial \theta}= \\ \frac{\partial \lambda}{\partial \xi}B\frac{\partial y}{\partial \theta}+\frac{\partial \lambda}{\partial \theta}B\frac{\partial y}{\partial \xi}+\frac{\partial \lambda}{\partial \xi}\frac{\partial B}{\partial \theta}y+\frac{\partial \lambda}{\partial \theta}\frac{\partial B}{\partial \xi}y+ \\ \frac{\partial^2 \lambda}{\partial \xi\partial \theta}-\left(\frac{\partial^2 A}{\partial \xi\partial \theta}-\lambda\frac{\partial^2 B}{\partial \xi\partial \theta}\right)y. \tag{13}$$

This can be solved for the second partials of the generalized eigenvalues. After some simplification we find

$$\frac{\partial^2 \lambda}{\partial \xi\partial \theta}=y'\left(\frac{\partial^2 A}{\partial \xi\partial \theta}-\lambda\frac{\partial^2 B}{\partial \xi\partial \theta}\right)y+ \\ -2y'\left(\frac{\partial A}{\partial \xi}-\lambda\frac{\partial B}{\partial \xi}\right)(A-\lambda B)^-\left(\frac{\partial A}{\partial \theta}-\lambda\frac{\partial B}{\partial \theta}\right)y- \\ y'\left(\frac{\partial \lambda}{\partial \xi}\frac{\partial B}{\partial \theta}+\frac{\partial \lambda}{\partial \theta}\frac{\partial B}{\partial \xi}\right)y. \tag{14}$$

It is also possible, although somewhat painful, to solve (13) for the second partials of the eigenvectors. In this case the simplifications if B is a constant, or the identity, are quite dramatic.

2.2.1. *Generalized Singular Value Problems.* The generalized singular value problem solves a system of equations of the form (2). Here R is a real $n\times m$ matrix, while P symmetric of order n and Q symmetric of order m. We can suppose without loss of generality that $n\geq m$.

Define

$$A = \begin{bmatrix} \emptyset & R \\ R' & \emptyset \end{bmatrix},$$

$$B = \begin{bmatrix} P & \emptyset \\ \emptyset & Q \end{bmatrix},$$

$$y = \frac{1}{2}\sqrt{2}\begin{bmatrix} x \\ z \end{bmatrix},$$

and, using these, the generalized eigenvalue problem $Ay = \lambda By$, with $y'By = 1$. Then solving the generalized singular value problem can be done by solving the generalized eigenvalue problem, and vice versa.

More precisely, suppose

$$RZ = PX\Gamma, \tag{15a}$$

$$R'X = QZ\Gamma, \tag{15b}$$

$$X'PX = I, \tag{15c}$$

$$Z'QZ = I, \tag{15d}$$

where Z and Λ are $m \times m$, Λ is non-negative, and X is $n \times m$.

Define

$$Y = \begin{bmatrix} \frac{1}{2}\sqrt{2}X & \frac{1}{2}\sqrt{2}X & X_\perp \\ \frac{1}{2}\sqrt{2}Z & -\frac{1}{2}\sqrt{2}Z & \emptyset \end{bmatrix}, \tag{16a}$$

where $X_\perp$ is the $m \times (n - m)$ matrix satisfying $X'PX_\perp = \emptyset$ and $X_\perp'PX_\perp = I$, and define

$$\Lambda = \begin{bmatrix} \Gamma & \emptyset & \emptyset \\ \emptyset & -\Gamma & \emptyset \\ \emptyset & \emptyset & \emptyset \end{bmatrix}. \tag{16b}$$

Then $AY = BY\Lambda$ with $Y'BY = I$.

From our previous results on generalized eigen problems

$$\begin{bmatrix} -\gamma P & R \\ R' & -\gamma Q \end{bmatrix} \begin{bmatrix} \frac{\partial x}{\partial \theta} \\ \frac{\partial z}{\partial \theta} \end{bmatrix} = = - \begin{bmatrix} -\gamma \frac{\partial P}{\partial \theta} & \frac{\partial R}{\partial \theta} \\ \frac{\partial R'}{\partial \theta} & -\gamma \frac{\partial Q}{\partial \theta} \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} + \frac{\partial \gamma}{\partial \theta} \begin{bmatrix} P & \emptyset \\ \emptyset & Q \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix}. \tag{17}$$

It follows that

$$\frac{\partial \gamma}{\partial \theta} = x' \frac{\partial R}{\partial \theta} z - \frac{1}{2} \gamma \left(x' \frac{\partial P}{\partial \theta} x + z' \frac{\partial Q}{\partial \theta} z \right), \tag{18}$$

as well as

$$\begin{bmatrix} \frac{\partial x}{\partial \theta} \\ \frac{\partial z}{\partial \theta} \end{bmatrix} = - \begin{bmatrix} -\gamma P & R \\ R' & -\gamma Q \end{bmatrix}^{-} \begin{bmatrix} -\gamma \frac{\partial P}{\partial \theta} & \frac{\partial R}{\partial \theta} \\ \frac{\partial R'}{\partial \theta} & -\gamma \frac{\partial Q}{\partial \theta} \end{bmatrix} \begin{bmatrix} x \\ z \end{bmatrix} + - \frac{1}{4} (x' \frac{\partial P}{\partial \theta} x + z' \frac{\partial Q}{\partial \theta} z) \begin{bmatrix} x \\ z \end{bmatrix}. \tag{19}$$

Instead of using the generalized inverse of an $(n + m) \times (n + m)$ matrix in (19) it is computationally more sensible to use (16) and use smaller matrices. We find

$$\begin{bmatrix} -\gamma P & R \\ R' & -\gamma Q \end{bmatrix}^{-} = \begin{bmatrix} \gamma X (\Gamma^2 - \gamma^2 I)^{+} X' - \frac{1}{\gamma} X_{\perp} X_{\perp}' & X\Gamma(\Gamma^2 - \gamma^2 I)^{+} Z' \\ Z\Gamma(\Gamma^2 - \gamma^2 I)^{+} X' & \gamma Z (\Gamma^2 - \gamma^2 I)^{+} Z' \end{bmatrix}. \tag{20}$$

Unfortunately, no matter how we manipulate the equations, they do not really simplify beyond this. But writing a computer program is relatively simple (see Appendix A).

2.2.2. *Singular Value Problems.* If P and Q are constant, and equal to the identity I, then X and Z are orthonormal. In that case

$$\begin{bmatrix} -\gamma P & R \\ R' & -\gamma Q \end{bmatrix}^{-} = \begin{bmatrix} \gamma (RR' - \gamma^2 I)^{+} & R(R'R - \gamma^2 I)^{+} \\ (R'R - \gamma^2 I)^{+} R' & \gamma (R'R - \gamma^2 I)^{+} \end{bmatrix}. \tag{21}$$

Consequently

$$\frac{\partial \gamma}{\partial \theta} = x' \frac{\partial R}{\partial \theta} z, \tag{22a}$$

$$\frac{\partial x}{\partial \theta} = \gamma (RR' - \gamma^2 I)^+ \frac{\partial R}{\partial \theta} z + R(R'R - \gamma^2 I)^+ \frac{\partial R'}{\partial \theta} x, \tag{22b}$$

$$\frac{\partial z}{\partial \theta} = (R'R - \gamma^2 I)^+ R' \frac{\partial R}{\partial \theta} z + \gamma (R'R - \gamma^2 I)^+ \frac{\partial R'}{\partial \theta} x. \tag{22c}$$

This agrees with the results given for the singular value decomposition, for example, in Papadopulo and Lourakis [2000] and O'Neil [2005], although their notation is slightly different.

3. Problems Linear in the Parameters

Consider a generalized eigenvalue problem of the form $Ay = \lambda By$, where

$$A = \sum_{k=1}^{K} p_k A_k,$$

and

$$B = \sum_{k=1}^{K} p_k B_k.$$

Since the matrices are linear in the parameters computing the derivatives is relatively simple. We find

$$\frac{\partial \lambda_s}{\partial p_k} = y_s'(A_k - \lambda_s B_k)y_s, \tag{23a}$$

$$\frac{\partial y_s}{\partial p_k} = -(A - \lambda_s B)^{-}(A_k - \lambda_s B_k)y_s - (\frac{1}{2}y_s' B_k y_s)y_s. \tag{23b}$$

Observe

$$\sum_{k=1}^{k} p_k \frac{\partial \lambda_s}{\partial p_k} = 0.$$

This also follows from the fact that the eigenvalues do not change if we multiply p by a constant, i.e. eigenvalues are homogeneous of degree zero and we can apply Euler's Theorem. In the same way

$$\sum_{k=1}^{k} p_k \frac{\partial y_s}{\partial p_k} = -\frac{1}{2}y,$$

which also follows from Euler's Theorem if we realize that normalized eigenvectors are of degree $-\frac{1}{2}$.

3.1. Asymptotic Statistics.

In a statistical context, suppose we have a discrete random variable that takes as values the matrix pairs (A_k, B_k), with probabilities π_k. The p_k are then the observed proportions in a sequence of N independent trials, and thus they are asymptotically normal. More precisely, we assume there is a sequence of random variables $\underline{p}_i$, with p_N as a realization of $\underline{p}_N$, such that

$$\sqrt{N}(\underline{p}_N - \pi) \overset{\mathcal{L}}{\Rightarrow} \mathcal{N}(0, \Pi - \pi\pi'),$$

where Π is the diagonal matrix with the elements of π on the diagonal.

First look at the eigenvalues, for which the situation is much simpler than for the eigenvectors. Using hats for observed valued, we estimate the dispersion matrix by

$$N\widehat{\mathbf{ACov}}(\hat\lambda_s,\hat\lambda_t)=\sum_{k=1}^{k}\hat p_k\left.\frac{\partial\lambda_s}{\partial p_k}\right|_{p_k=\hat p_k}\left.\frac{\partial\lambda_t}{\partial p_k}\right|_{p_k=\hat p_k}.$$

This means, from the computational point of view, we construct the $K\times p$ matrix $\hat U$ with elements

$$\hat u_{ks}=\hat y_s(A_k-\hat\lambda_s B_k)\hat y_s,$$

and then $n\;\widehat{\mathbf{ACov}}(\hat\lambda_s,\hat\lambda_t)=\hat U'\hat P\hat U.$

For the eigenvectors the formulas are, as usual, more complicated. We have

$$N\widehat{\mathbf{ACov}}(\hat y_{is},\hat y_{it})=\sum_{k=1}^{k}\hat p_k\left.\frac{\partial y_{is}}{\partial p_k}\right|_{p_k=\hat p_k}\left.\frac{\partial y_{it}}{\partial p_k}\right|_{p_k=\hat p_k}-\frac{1}{4}\hat y_{is}\hat y_{it}.$$

For computation we can construct the $K\times n$ matrices $\hat T_s$ with elements

$$\{\hat T_s\}_{kj}=\{(\hat A-\hat\lambda_s\hat B)^-(A_k-\hat\lambda_s B_k)\hat y_s\}_j.$$

Put the p matrices $\hat T_s$ that we need next to each other in a $K\times np$ matrix $\hat T$, then the asymptotic covariances we are looking for can be selected from the covariance matrix computed from the columns of $\hat T$, i.e. from $\hat T'(\hat P-\hat p\hat p')\hat T$.

Observe, however, that in order to compute $\hat T$ we need to compute all eigenvalues and eigenvectors, which we need for $(\hat A-\hat\lambda_s\hat B)^-$. Much of the literature we reviewed in earlier sections deals with ways to avoid having to compute a complete decomposition, for example by using iteration methods.

3.2. **Reindexing.** It is of some importance that instead of indexing the matrices A_k and B_k, and weighting them by the proportions p_k, we can also index them by observation i and use obvious identities such as

$$A = \sum_{k=1}^{K} p_k A_k = \frac{1}{n} \sum_{i=1}^{n} A_i.$$

This type of indexing may be preferable for computational purposes, for example if the number of observations n is much smaller than $K = \prod_{j=1}^{m} k_j$, the number of possible profiles.

3.3. **Interesting Special Cases.**

3.3.1. *Multiple Correspondence Analysis.* In multiple correspondence analysis (MCA), for instance, the A_k are of the form $g_k g_k'$, where the g_k are binary profile vectors for m categorical variables. Thus the g_k are m concatenated unit vectors, each of length k_j, where k_j is the number of categories of variable j. The vectors g_k have $\sum_{j=1}^{m} k_j$ elements. The B_k are defined as $G_k = \mathbf{diag}(g_k g_k')$. Thus A is the *Burt matrix* [Burt, 1950], and B is its diagonal.

In MCA we are mostly interested in the asymptotic covariances of the eigenvalues and the asymptotic covariances of corresponding i^{th} elements from different eigenvectors. The covariance matrix of these eigenvector elements can be used to draw confidence ellipsoids around the points in the the biplots, as in Gifi [1990, Section 12.1.1]. The formulas will be incorporated in the next version of De Leeuw and Mair [2008a]. The implementation is in Appendix A.3

3.3.2. *Correspondence Analysis.* Derivatives of the singular values and vectors in ordinary correspondence analysis have been considered by O'Neill [1978a,b]; Kuriki [2005]. We can either use the generalized eigenvalue formulation for the Burt matrix, with only two blocks, or we can use the generalized singular value decomposition formulas. In both cases the matrices are linear in the proportions, and basically the same techniques apply as discussed above.

The singular value method is incorporated in De Leeuw and Mair [2008b]. The code is in Appendix A.2.

3.3.3. *Covariances and Correlations.* If a discrete random variable takes vector values s_k with probabilities π_k, then the sample covariance is

$$A = \sum_{k=1}^{K} p_k(x_k - m)(x_k - m)',$$

where

$$m = \sum_{k=1}^{K} p_k x_k.$$

Of course m depends on the p_k and this should be taken into account when computing derivatives. But for first order asymptotic statistics we can treat m as if it was a vector of constants. The eigenvalues of the correlation matrix are the generalized eigenvalues of the pair A and $B = \mathbf{diag}(A)$. Basically, the same formulas and the same computations apply as in the correspondence analysis case. If it is desirable, we can switch again from indexing by profiles x_k to indexing by observations x_i.

3.4. **Scaling the Vectors.** If we make plots of the eigenvectors of generalized eigenvalue problems, we often scale them by using $\tilde{y} = \sqrt{\lambda}y$, so that $\tilde{y}'B\tilde{y} = \lambda$. Of course

$$\frac{\partial\tilde{y}}{\partial\theta} = \frac{1}{2}\frac{1}{\sqrt{\lambda}}\frac{\partial\lambda}{\partial\theta}y + \sqrt{\lambda}\frac{\partial y}{\partial\theta},$$

and these partials can be readily computed from the ones we already have.

References

A.L. Andrew and R.C. Tan. Computation of Derivatives of Repeated Eigenvalues and the Corresponding Eigenvectors of Symmetric Matrix Pencils. *SIAM Journal Matrix Analysis and Applications*, 20:78–100, 1998.

H. Baumgärtel. *Analytic Perturbation Theory for Matrices and Operators*. Birkhauser, Basel, Boston, Stuttgart, 1985.

C. Burt. The Factorial Analysis of Qualitative Data. *British Journal of Statistical Psychology*, 3:166–185, 1950.

K.-W. E. Chu. On Multiple Eigenvalues of Matrices Depending on Several Parameters. *SIAM Journal of Numerical Analysis*, 27: 1368–1385, 1990.

J. De Leeuw and P. Mair. Homogeneity Analysis in R: The package homals. *Journal of Statistical Software*, (forthcoming), 2008a.

J. De Leeuw and P. Mair. Simple and Canonical Correspondence Analysis Using the R Package anacor. *Journal of Statistical Software*, (forthcoming), 2008b.

A. Gifi. *Nonlinear Multivariate Analysis*. Wiley, Chichester, England, 1990.

T. Kato. *Perturbation Theory for Linear Operators*. Springer-Verlag, Berlin, Heidelberg, New York, second edition, 1976.

T. Kollo and H. Neudecker. The Derivative of an Orthogonal matrix of Eigenvectors of a Symmetric Matrix. *Linear Algebra and Its Applications*, 264:489–493, 1997a.

T. Kollo and H. Neudecker. Asymptotics of Eigenvalues and Unit-Length Eigenvectors of Sample Variance aand Correlation Matrices. *Journal of Multivariate Analysis*, 47:283–300, 1997b.

S. Kuriki. Asymptotic Distribution of Inequality-restricted Canonical Correlation wih Application to Tests for Independence in Ordered Contingency Tables. *Journal of Multivariate Analysis*, 94: 420–449, 2005.

A.S. Lewis. The Mathematics of Eigenvalue Optimization. *Mathematical Programming, Series B*, 97:155–176, 2003.

A.S. Lewis and M.L. Overton. Eigenvalue Optimization. *Acta Numerica*, 5:149–190, 1996.

J.R. Magnus and H. Neudecker. *Matrix Differential Calculus, with Applications in Statistics and Economatrix*. Wiley, New York, 1998.

B.J. McCartin. Pseudoinverse Formulation of Rayleigh-Schrödinger Perturbation Theory for the Symmetric Eigenvalue Problem. *Journal of Applied Mathematics*, 9:459–485, 2003.

K. A. O'Neil. Critical Points of the Singular Value Decomposition. *SIAM Journal Matrix Analysis and Applications*, 27:459–473, 2005.

M.E. O'Neill. Distributional Expansions for Canonical Correlations from Contingency Tables. *Journal of the Royal Statistical Society B*, 40:303–312, 1978a.

M.E. O'Neill. Asymptotic Distribution of the Canonical Correlations from Contingency Tables. *Australian Journal of Statistics*, 20:75–82, 1978b.

T. Papadopulo and M.I.A. Lourakis. Estimating the Jacobian of the Singular Value Decomposition: Theory and Applications. Rapport de Recherche 3961, INRIA, 2000.

J.-G. Sun. Eigenvalues and Eigenvectors of a Matrix Dependent on Several Parameters. *Journal of Computational Mathematics*, 3:727–764, 1985.

J.-G. Sun. Multiple Eigenvalue Sensitivity Analysis. *Linear Algebra and Its Applications*, 137:183–211, 1990.

A.W. Van Der Vaart. *Asymptotic Statistics*. Cambridge Series in Statistical and Probabilistic Mathematics. Cambridge University Press, 1998.

R. Wang and W. Wang. On the Application of Generalized Inverse for Computing Eigen-Sensitivity. *Acta Mechanica Solida Sinica*, 7:292–305, 1994.

W. Wang. A Family of Modal Methods for Computing Eigenvector Derivatives with Repeated Roots. *Acta Mathematica Sinica (English Series)*, 12:158–168, 1996.

J.H. Wilkinson. *The Algebraic Eigenvalue Problem*. Clarendon Press, Oxford, England, 1965.

Z.-Y. Zhang and H.-S. Zhang. Calculation of Eigenvalue and Eigenvector Derivatives of a Defective Matrix. *Applied Mathematics and Computation*, 176:7–26, 2006.

Appendix A. Code

The two major functions in the code below are `gevdDer()`and `gsvdDer()`. The other functions are support and help functions, and some extensions.

`gevdDer()` does a generalized eigenvalue analysis of a pair $(A, B$ at a point θ. The function has four required function arguments that the user needs to provide. The first two `a()` and `b()` return the value of the matrices at the parameter values, the second two `da()` and `db()` return the partial derivatives at the parameter value.

Note that if A and B are of order n and we have K parameters and look at s eigenpairs, then `da()` and `db()` must return arrays of dimension $n \times n \times K$. `gevdDer()` returns a list with the generalied eigenvalues in `gd`, the generalized eigenvectors in `gv`, the partial derivatives of the s eigenvalues we have selected in the $s \times K$ matrix `dl`, and the partial derivatives of the eigenvectors in the $n \times s \times K$ array `dy`.

`gsvdDer()` does a generalized singular value decomposition of the triple (P, Q, R). It needs six function arguments to compute the matrices and their derivatives, and it returns the three components of the singular value decomposition with their three blocks of derivatives.

The two additional function `gevdScal()`and `gsvdScal()` scale the length of the eigenvectors relative to the eigenvalues. For the function `gevdScal()` we use $\tilde{y} = \sqrt{\lambda}y$. The function also returns the derivatives of the scaled eigenvectors.

For the function `gsvdScal()` we can choose between four options. In Benzécri scaling we set $\tilde{x} = xy$ and $\tilde{z} = zy$. In Goodman scaling this becomes $\tilde{x} = x\sqrt{y}$ and $\tilde{z} = z\sqrt{y}$. In row-column scaling we use $\tilde{x} = x$ and $\tilde{z} = zy$, while in column-row scaling it is $\tilde{x} = xy$ and

$\tilde{z} = z$. For all four cases the function returns the scaled singular vectors, together with their partial derivatives.

The use of `gsvdDer()` and `gsvdScal()` is illsutrated in a second code chunk, which gives a correspondence analysis program with asymptotic dispersion matrices and plots of 95% confidence ellipses. This is spaghetti code, a much improved version will be published as a `CRAN` package.

A.1. **Package.**

```
#
#   diffEigen package
#   Copyright (C) 2007  Jan de Leeuw <deleeuw@stat.ucla.edu>
#   UCLA Department of Statistics, Box 951554, Los Angeles, CA 90095-1554
#
#   This program is free software; you can redistribute it and/or modify
#   it under the terms of the GNU General Public License as published by
#   the Free Software Foundation; either version 2 of the License, or
#   (at your option) any later version.
#
#   This program is distributed in the hope that it will be useful,
#   but WITHOUT ANY WARRANTY; without even the implied warranty of
#   MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
#   GNU General Public License for more details.
#
#   You should have received a copy of the GNU General Public License
#   along with this program; if not, write to the Free Software
#   Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.
#
###################################################################
#
# version 0.0.1, 2007-12-15   Initial Alpha Release
# version 0.0.2, 2007-12-16   Added scaled versions of vectors
#


#  gevd compute the generalized eigenvalue
# decomposition for (a,b)

gevd<-function(a,b=diag(nrow(a))) {
        bs<-mfunc(b,function(x) ginvx(sqrt(x)))
```

```
        ev<-eigen(bs%*%a%*%bs)
        return(list(gvalues=ev$values,gvectors=bs%*%ev$vectors))
}

# gsvd computes the generalized singular value
# decomposition for (r,p,q)

gsvd<-function(r,p=diag(nrow(a)),q=diag(ncol(a))) {
        ps<-mfunc(p,function(x) ginvx(sqrt(x)))
        qs<-mfunc(q,function(x) ginvx(sqrt(x)))
        sv<-svd(ps%*%r%*%qs)
        return(list(gd=sv$d,gu=ps%*%sv$u,gv=qs%*%sv$v))
}

# ginvgevd: the (1,2) inverse needed for derivatives of
# generalized eigenvectors

ginvgevd<-function(ge,ind) {
        y<-ge$gvectors; v<-ge$gvalues-(ge$gvalues[ind])
        return(tcrossprod(y%*%diag(ginvx(v)),y))
}

# ginvgsvd: the (1,2) inverse needed for derivatives of
# generalized singular vectors

ginvgsvd<-function(gs,p,ind) {
        x<-gs$gu; z<-gs$gv; gm<-gs$gd; gi<-gm[ind]
        iv1<-(ginvx(gm-gi)+ginvx(-(gm+gi)))/2
        iv2<-(ginvx(gm-gi)-ginvx(-(gm+gi)))/2
        n<-nrow(x); m<-nrow(z)
        a<-matrix(0,n+m,n+m)
        b<-tcrossprod(z%*%diag(iv2),x)
        a[1:n,n+(1:m)]<-t(b); a[n+(1:m),1:n]<-b
        a[n+(1:m),n+(1:m)]<-tcrossprod(z%*%diag(iv1),z)
        a[1:n,1:n]<-tcrossprod(x%*%diag(iv1),x)
        a[1:n,1:n]<-a[1:n,1:n]-(solve(p)-tcrossprod(x))/gi
        return(a)
}

# gevdDer: generalized eigenvalue decomposition plus derivatives
# needs four functions to compute a, b, da, and db

gevdDer<-function(par,aPar,bPar,daPar,dbPar,ind=1) {
```

```
    a<-aPar(par); b<-bPar(par); da<-daPar(par); db<-dbPar(par)
    nord<-nrow(a); neval<-length(ind); npars<-length(par)
    ge<-gevd(a,b); gv<-ge$gvectors; gd<-ge$gvalues
    dl<-matrix(0,neval,npars); dy<-array(0,c(nord,neval,npars))
    for (i in 1:neval) {
        j<-ind[i]; y<-gv[,j]; lb<-gd[j]
        dl[i,]<-apply(da-lb*db,3,function(x) y%*%x%*%y)
        aux0<-ginvgevd(ge,j)
        aux1<-apply(da-lb*db,3,function(x) aux0%*%x%*%y)
        aux2<-outer(y,apply(db,3,function(x) y%*%x%*%y))/2
        dy[,i,]<- -(aux1+aux2)
        }
    return(list(gd=gd,gv=gv,dl=dl,dy=dy,ind=ind))
}

# gsvdDer: generalized singular value decomposition plus derivatives
# needs six functions to compute p, q, r, dp, dq, and dr


gsvdDer<-function(par,pPar,qPar,rPar,dpPar,dqPar,drPar,ind=1) {
    p<-pPar(par); q<-qPar(par); r<-rPar(par)
    dp<-dpPar(par); dq<-dqPar(par); dr<-drPar(par)
    nrows<-nrow(p); ncols<-nrow(q); neval<-length(ind); npars<-length(par)
    gs<-gsvd(r,p,q); gu<-gs$gu; gv<-gs$gv; gd<-gs$gd
    dl<-matrix(0,neval,npars)
    dx<-array(0,c(nrows,neval,npars)); dz<-array(0,c(ncols,neval,npars))
    for (i in 1:neval) {
        j<-ind[i]; x<-gu[,j]; z<-gv[,j]; gi<-gd[j]
        xz<-rbind(cbind(x),cbind(z))
        dpx<-apply(dp,3,function(d) x%*%d%*%x)
        dqz<-apply(dq,3,function(d) z%*%d%*%z)
        dl[i,]<-apply(dr,3,function(d) x%*%d%*%z)-gi*(dpx+dqz)/2
        aux0<-ginvgsvd(gs,p,j)
        kv<-array(0,c(nrows+ncols,nrows+ncols,npars))
        kv[1:nrows,1:nrows,]<- -gi*dp
        kv[1:nrows,nrows+(1:ncols),]<-dr
        kv[nrows+(1:ncols),1:nrows,]<-aperm(dr,c(2,1,3))
        kv[nrows+(1:ncols),nrows+(1:ncols),]<- -gi*dq
        aux1<-apply(kv,3,function(d) aux0%*%d%*%xz)
        aux2<-drop(outer(xz,(dpx+dqz)/4))
        dxz<- -(aux1+aux2)
        dx[,i,]<-dxz[1:nrows,]
```

```
                dz[,i,]<-dxz[nrows+(1:ncols),]
                }
        return(list(gd=gd,gu=gu,gv=gv,dl=dl,dx=dx,dz=dz,ind=ind))
}

# gevdScal: generalized eigen value decomposition with eigen vector
# length scaled to generalized eigenvalue

gevdScal<-function(ge) {
        gd<-ge$gd; gv<-ge$gv; dl<-ge$dl; dy<-ge$dy
        dys<-dy; gvs<-gv; ind<-ge$ind; neval<-length(ind)
        for (i in 1:ncol(gv))
                gvs[,i]<-gv[,i]*sqrt(gd[i])
        for (i in 1:neval) {
                j<-ind[i]
                dys[,i,]<-1/(2*sqrt(gd[j]))*outer(gv[,j],dl[i,])+sqrt(gd[j])
                    *dy[,i,]
                }
return(list(gd=gd,gv=gvs,dl=dl,dy=dys,ind=ind))
}

# gsvdScal: generalized singular value decomposition with singular vector
# length scaled to generalized singular value (in four ways)

gsvdScal<-function(gs,scal="be") {
        gd<-gs$gd; gu<-gs$gu; gv<-gs$gv
        dl<-gs$dl; dx<-gs$dx; dz<-gs$dz
        dxs<-dx; dzs<-dz; gus<-gu; gvs<-gv
        ind<-gs$ind; neval<-length(ind)
        if (scal=="be") {
                for (i in length(gd)) {
                        gus[,i]<-gu[,i]*gd[i]
                        gvs[,i]<-gv[,i]*gd[i]
                        }
                for (i in 1:neval) {
                        j<-ind[i]
                        dxs[,i,]<-outer(gu[,j],dl[i,])+gd[j]*dx[,i,]
                        dzs[,i,]<-outer(gv[,j],dl[i,])+gd[j]*dz[,i,]
                        }
                }
        if (scal=="go") {
                for (i in length(gd)) {
                        gus[,i]<-gu[,i]*sqrt(gd[i])
```

```
                        gvs[,i]<-gv[,i]*sqrt(gd[i])
                        }
                for (i in 1:neval) {
                        j<-ind[i]
                        dxs[,i,]<-1/(2*sqrt(gd[j]))*outer(gu[,j],dl[i,])+
                            sqrt(gd[j])*dx[,i,]
                        dzs[,i,]<-1/(2*sqrt(gd[j]))*outer(gv[,j],dl[i,])+
                            sqrt(gd[j])*dz[,i,]
                        }
                }
        if (scal=="rc") {
                for (i in length(gd)) {
                        gvs[,i]<-gv[,i]*gd[i]
                        }
                for (i in 1:neval) {
                        j<-ind[i]
                        dzs[,i,]<-outer(gv[,j],dl[i,])+gd[j]*dz[,i,]
                        }
                }
        if (scal=="cr") {
                for (i in length(gd)) {
                        gus[,i]<-gu[,i]*gd[i]
                        }
                for (i in 1:neval) {
                        j<-ind[i]
                        dxs[,i,]<-outer(gu[,j],dl[i,])+gd[j]*dx[,i,]
                        }
                }
return(list(gd=gd,gu=gus,gv=gvs,dl=dl,dx=dxs,dz=dzs,ind=ind))
}

# ginvx is a helper to compute reciprocals

ginvx<-function(x) {ifelse(x==0,0,1/x)}

# mfunc is a helper to compute matrix functions

mfunc<-function(a,fn=sqrt) {
        e<-eigen(a); y<-e$vectors; v<-e$values
        return(tcrossprod(y%*%diag(fn(v)),y))
}
```

A.2. CA.

```
source("diffEigen.R")
library(car)

myAnaCor<-function(data,ind=2:3,scal="no") {
nr<-nrow(data); nc<-ncol(data); nn<-nr*nc
mOff<-array(0,c(nr,nc,nn))
mRow<-array(0,c(nr,nr,nn))
mCol<-array(0,c(nc,nc,nn))
freq<-rep(0,nn)
k<-1
for (i in 1:nr) for (j in 1:nc) {
        mOff[i,j,k]<-1
        mRow[i,i,k]<-1
        mCol[j,j,k]<-1
        freq[k]<-data[i,j]
        k<-k+1
}
N<-sum(freq)
prop<-freq/N
rpar<-function(p) {
        res<-matrix(0,nr,nc)
        for (k in 1:nn) res<-res+p[k]*mOff[,,k]
        return(res)
        }
ppar<-function(p) {
        res<-matrix(0,nr,nr)
        for (k in 1:nn) res<-res+p[k]*mRow[,,k]
        return(res)
        }
qpar<-function(p) {
        res<-matrix(0,nc,nc)
        for (k in 1:nn) res<-res+p[k]*mCol[,,k]
        return(res)
        }
drpar<-function(p) return(mOff)
dppar<-function(p) return(mRow)
dqpar<-function(p) return(mCol)
gs<-gsvdDer(prop,ppar,qpar,rpar,dppar,dqpar,drpar,ind)
if (!(scal=="no")) gs<-gsvdScal(gs,scal=scal)
gdl<-gs$dl
acovd<-gdl%*%(prop*t(gdl))/N
acovu<-array(0,c(2,2,nr))
```

```
gdx<-gs$dx
for (i in 1:nr) {
        gdxi<-gdx[i,,]
        gdxs<-drop(gdxi%*%prop)
        acovu[,,i]<-(gdxi%*%(prop*t(gdxi))-outer(gdxs,gdxs))/N
        }
acovv<-array(0,c(2,2,nc))
gdz<-gs$dz
for (i in 1:nc) {
        gdzi<-gdz[i,,]
        gdzs<-drop(gdzi%*%prop)
        acovv[,,i]<-(gdzi%*%(prop*t(gdzi))-outer(gdzs,gdzs))/N
        }
return(list(gs=gs,acovd=acovd,acovu=acovu,acovv=acovv,ind=ind))
}

myAnaPlot<-function(myAna,conf=.95){
rad<-sqrt(qchisq(conf,2))
gv<-myAna$gs$gv; gu<-myAna$gs$gu
nr<-nrow(gu); nc<-nrow(gv); ind<-myAna$ind
pdf("colPlot.pdf")
plot(gv[,ind],xlab="Dim 1",ylab="Dim 2",type="n")
for (i in 1:nc) ellipse(gv[i,ind],myAna$acovv[,,i],rad)
dev.off()
pdf("rowPlot.pdf")
plot(gu[,ind],xlab="Dim 1",ylab="Dim 2",type="n")
for (i in 1:nr) ellipse(gu[i,ind],myAna$acovu[,,i],rad)
dev.off()
}
```

A.3. **MCA.**

```

source("diffEigen.R")
source("preProcess.R")
library(car)

myAnaProf<-function(profiles,freq,ind=2:3,scal=TRUE) {
nr<-nrow(profiles); nn<-ncol(profiles)
mA<-array(0,c(nn,nn,nr))
mB<-array(0,c(nn,nn,nr))
k<-1
for (i in 1:nr) {
```

```
        tt<-profiles[i,]
        mA[,,k]<-outer(tt,tt)
        diag(mB[,,k])<-tt
        k<-k+1
}
N<-sum(freq)
prop<-freq/N
apar<-function(p) {
        res<-matrix(0,nn,nn)
        for (k in 1:nr) res<-res+p[k]*mA[,,k]
        return(res)
        }
bpar<-function(p) {
        res<-matrix(0,nn,nn)
        for (k in 1:nr) res<-res+p[k]*mB[,,k]
        return(res)
        }
dapar<-function(p) return(mA)
dbpar<-function(p) return(mB)
gs<-gevdDer(prop,apar,bpar,dapar,dbpar,ind)
names(gs)
if (scal) gs<-gevdScal(gs)
gdl<-gs$dl
acovd<-gdl%*%(prop*t(gdl))/N
acovy<-array(0,c(2,2,nn))
gdy<-gs$dy
for (i in 1:nn) {
        gdyi<-gdy[i,,]
        gdys<-drop(gdyi%*%prop)
        acovy[,,i]<-(gdyi%*%(prop*t(gdyi))-outer(gdys,gdys))/N
        }
return(list(gs=gs,acovd=acovd,acovy=acovy,ind=ind))
}

myProfPlot<-function(myMC,conf=.95){
rad<-sqrt(qchisq(conf,2))
gv<-myMC$gs$gv
nn<-nrow(gv); ind<-myMC$ind
pdf("profPlot.pdf")
plot(gv[,ind],xlab="Dim 1",ylab="Dim 2",type="n")
for (i in 1:nn) ellipse(gv[i,ind],myMC$acovy[,,i],rad)
dev.off()
}
```

Department of Statistics, University of California, Los Angeles, CA 90095-1554

E-mail address, Jan de Leeuw: `deleeuw@stat.ucla.edu`

URL, Jan de Leeuw: `http://gifi.stat.ucla.edu`