QUADRATIC AND CUBIC MAJORIZATION

JAN DE LEEUW

Abstract. Majorization algorithms generalize the EM algorithm. In this paper we discuss and compare various quadratic and cubic majorization methods, and apply them to logistic regression.

1. Majorization

Majorization algorithms [De Leeuw, 1994; Heiser, 1995; Lange et al., 2000] are used with increasing frequency in statistical computation. They generalize the EM algorithm to a much broader class of problems and they can usually be tailored to handle very high-dimensional problems.

The general idea is simple. If we want to minimize f over $X \subseteq \mathbb{R}^n$, we construct a *majorization function* g on $X \times X$ such that

$$f(x) \leq g(x, y) \qquad \forall x, y \in X,$$
$$f(x) = g(x) \qquad \forall x \in X.$$

The majorization algorithm corresponding with this majorization function g updates x at iteration k by

$$x^{(k+1)} \in \underset{x \in X}{\textbf{argmin}}\, g(x, x^{(k)}),$$

unless we already have

$$x^{(k)} \in \underset{x \in X}{\textbf{argmin}}\, g(x, x^{(k)}),$$

in which case we stop. Convergence follows, under some additional simple conditions, from the *sandwich inequality*, which says that if we do not stop

Date: March 23, 2006.

2000 *Mathematics Subject Classification.* 90C30.

Key words and phrases. Mathematical programming, Nonlinear Programming.

at iteration k, then

$$f(x^{(k+1)}) \leq g(x^{(k+1)}, x^{(k)}) < g(x^{(k)}, x^{(k)}) = f(x^{(k)}).$$

2. Quadratic Majorization

Majorization functions can be constructed by using classic inequalities such as Cauchy-Schwartz or Jensen, by using convexity, or by using Taylor's theorem. Taylor's theorem usually lead to *quadratic majorization algorithms* [Böhning and Lindsay, 1988], which we explain next.

By Taylor's theorem

$$f(x) = f(y) + (x - y)'\partial f(y) + \frac{1}{2}(x - y)'\partial^2 f(\xi)(x - y)$$

for some ξ on the line between x and y. Now suppose there is a matrix B such that $\partial^2 f(\xi) \lesssim B$, in the sense that $B - \partial^2 f(\xi)$ is positive semi-definite for all ξ. Then clearly

$$g(x, y) = f(y) + (x - y)'\partial f(y) + \frac{1}{2}(x - y)'B(x - y)$$

is a majorization function for f. We also write $g_y(x)$ for $g(x, y)$ to emphasize that g_y is a function of x that majorizes f at y. Observe that g_y has both the same function value and the same derivative as f at y.

By defining the current *target*

$$z = y - B^{-1}\partial f(y),$$

and by completing the square, we see that

$$g(x, y) = f(y) + \frac{1}{2}(x - z)'B(x - z) - \frac{1}{2}\partial f(y)'B^{-1}\partial f(y).$$

Thus step k of the majorization algorithm solves the least squares problem

$$\min_{x \in X}(x - z^{(k)})'B(x - z^{(k)}).$$

We can choose the matrix B to be scalar, for instance by using an upper bound for the largest eigenvalue of $\partial^2 f(\xi)$. In that case computing the target simplifies, and all majorization subproblems are unweighted least squares problems.

In the case in which X is all of $\mathbb{R}^n$ the quadratic majorization algorithm simply becomes

$$x^{(k+1)} = x^{(k)} - B^{-1}\partial f(x^{(k)}). \tag{1}$$

By Ostrowski's theorem [Ortega and Rheinboldt, 1970, page 300-301] if algorithm (1) converges to x^∞, then it will in general have a linear convergence rate $1 - \lambda(x^\infty)$, where $\lambda(x^\infty)$ is the smallest eigenvalue of $B^{-1}\partial^2 f(x^\infty)$. A smaller B will give a more rapid convergence rate, but in general we cannot expect to see anything faster than linear convergence. If our bound B is really bad, then we may see very slow linear convergence.

One straightforward way to improve slow linear convergence in quadratic majorization was first suggested by De Leeuw and Heiser [1980]. They show that global convergence is preserved if we use *over-relaxation* and upgrade by

$$x^{(k+1)} = x^{(k)} - 2B^{-1}\partial f(x^{(k)}), \tag{2}$$

i.e. we make the step twice a large. If the convergence rate for (1) is $1 - \lambda$, then that for (2) is $1 - 2\lambda$. Thus for small λ it is approximately $(1 - \lambda)^2$. The number of iterations will be approximately half of what is was before.

2.1. **Example.** We use maximum likelihood logistic regression as an example because it leads to relatively simple computations and clearly illustrates the principles of algorithm construction. In logistic regression we have an $n \times p$ matrix Z with regressors, and we solve for the p regression coefficients β.

The negative log-likelihood is

$$f(\beta) = -\sum_{i=1}^n n_i z_i'\beta - \sum_{i=1}^n N_i \log(1 - \pi_i(\beta)),$$

where

$$\pi_i(\beta) = \frac{1}{1 + \exp(-z_i'\beta)}.$$

Since

$$\partial\pi_i(\beta) = \pi_i(\beta)(1 - \pi_i(\beta))x_i$$

we have

$$\partial f(\beta) = \sum_{i=1}^{n}(N_i\pi_i(\beta) - n_i)z_i,$$

and

$$\partial^2 f(\beta) = \sum_{i=1}^{n} N_i\pi_i(\beta)(1 - \pi_i(\beta))\, z_i \otimes z_i,$$

where $\otimes$ is used for the outer product. If we collect the $N_i\pi_i(\beta)(1 - \pi_i(\beta))$ in a diagonal matrix $V(\beta)$, then the Hessian is simply $X'V(\beta)X$. Observe that the Hessian is positive semi-definite, and thus the negative log-likelihood is convex (strictly convex if Z is of full column-rank). The gradient is $Z'u(\beta)$, where $u(\beta)$ has elements $N_i\pi_i(\beta) - n_i$.

The most straightforward algorithm to minimize f is Newton's method, which is simply

$$\beta^{(k+1)} = \beta^{(k)} - (Z'V(\beta^{(k)})Z)^{-1}Z'u(\beta^{(k)})$$

This requires solving a linear system in each iteration, but it will give super-linear, in fact quadratic, convergence to the unique minimum in most cases.

Quadratic majorization can be based on $\pi_i(\beta)(1 - \pi_i(\beta)) \leq \frac{1}{4}$, from which $V(\beta) \leq \frac{1}{4}N$, where N is the diagonal matrix with the N_i. Thus

$$\beta^{(k+1)} = \beta^{(k)} - 4(Z'NZ)^{-1}Z'u(\beta^{(k)})$$

which only requires a single matrix inversion. It does still require a matrix multiplication in each iteration.

If K is a bound for the largest eigenvalue of $Z'NZ$, i.e. if $b'Z'NZb \leq Kb'b$ for all b, then

$$\beta^{(k+1)} = \beta^{(k)} - \frac{4}{K}Z'u(\beta^{(k)})$$

will have slower convergence, but will require fewer multiplications per iteration.

2.2. **Computation.** Consider the data from Maxwell [1961, page 64] in Table 2.2. They indicate the number of boys in a clinic classified as inveterate liars by the resident psychiatrist. We do a simple logistic regression on age, which means Z has a column of ones and a column with the

numbers one to five. The maximum likelihood solution for the intercept is $\beta_0 \approx -1.1971$, while that for the slope is $\beta_1 \approx 0.2737$.

TABLE 1. Boys' Ratings on a Lie Scale

age	n	N-n	N
5-7	6	15	21
8-9	18	31	49
10-11	19	31	50
12-13	27	32	59
14-15	25	19	44

We now use the code in the Appendix to test various iterative methods, iterating in all cases until the sup-norm of the gradient is less than $1e-6$. We always start with all elements of β equal to one. Newton's method does not converge if we start relatively far from the solution, for instance at $(1,1)$. This is because the Hessian is numerically singular. The algorithm could easily be fixed by building in some safeguards. If we start closer Newton typically converges in less than 10 iterations.

The majorization method using $\frac{1}{4}Z'NZ$ as the upper bound for the Hessian typically also uses less than 10 iterations, and it converges from any starting point. If we start very far away, for example in $(10,10)$, then majorization takes about 30 iterations. Asymptotically majorization converges with a rate of about 0.1192, which is very fast and implies an additional decimal of precision in each iteration.

In this example majorization looks particulary good, because the $\pi_i(\beta)$ are all fairly close to $\frac{1}{2}$, which implies that the bound we use is quite sharp. If we use the much poorer bound $\frac{1}{4}KI$, using the largest eigenvalue of $Z'NZ$ for K, then the convergence rate becomes 0.9917, we need about 275 iterations for an additional decimal of precision, and the algorithm typically takes around 2000 iterations.

Alternatively, we can use the non-uniform bound $Z'W(\beta)Z$, with $W(\beta)$ the diagonal matrix with elements

$$w_i(\beta) = N_i \frac{2\pi_i(\beta) - 1}{2z_i'\beta}.$$

These non-uniform bounds for the logit were discovered, independently, by Jaakkola and Jordan [2000] and Groenen et al. [2003]. The resulting quadratic majorization algorithm

$$\beta^{(k+1)} = \beta^{(k)} - (Z'W(\beta^{(k)})Z)^{-1}Z'u(\beta^{(k)})$$

converges in less than 10 iterations, even if we start at $(10, 10)$, and has a rate of 0.0810.

This example illustrates the main points we want to make. Newton's method without safeguards can go astray, while majorization methods will converge but can be very slow if the bound for the Hessian is imprecise. Logistic regression is, in a sense, atypical because we have a convex negative log-likelihood, and thus a unique optimum. Newton's method is very well behaved, and difficult to improve upon. Also, our example has only two parameters. More complicated functions will no doubt behave quite differently.

For completeness, we also study the quite different Cancer Remission data of Lee [1974], given in Table 2.2. These data have a binary outcome (given in the last column), indicating in which of 27 patients remission occurred. There are six variables (plus the intercept), indicating the results of several medical tests.

We cannot get the un-safeguarded Newton's method to converge from our usual starting point. If we use quadratic majorization with the uniform bound $\frac{1}{4}$, we need 1475 iterations and the convergence rate is .9929. Over-relaxation leads to 731 iterations, with rate .9858. Using the non-uniform bound needs 278 iterations with rate .9600, and over-relaxed 115 iterations with rate 0.9200. The scalar bounds needs more than 100,000 iterations, at a rate equal to one in six decimals – so for all practical purposes we have sub-linear convergence in that case.

TABLE 2. Cancer Remission Data

	A	B	C	D	E	F	
1	0.80	0.83	0.66	1.9	1.100	0.996	1
1	0.90	0.36	0.32	1.4	0.740	0.992	1
1	0.80	0.88	0.70	0.8	0.176	0.982	0
1	1.00	0.87	0.87	0.7	1.053	0.986	0
1	0.90	0.75	0.68	1.3	0.519	0.980	1
1	1.00	0.65	0.65	0.6	0.519	0.982	0
1	0.95	0.97	0.92	1.0	1.230	0.992	1
1	0.95	0.87	0.83	1.9	1.354	1.020	0
1	1.00	0.45	0.45	0.8	0.322	0.999	0
1	0.95	0.36	0.34	0.5	0.000	1.038	0
1	0.85	0.39	0.33	0.7	0.279	0.988	0
1	0.70	0.76	0.53	1.2	0.146	0.982	0
1	0.80	0.46	0.37	0.4	0.380	1.006	0
1	0.20	0.39	0.08	0.8	0.114	0.990	0
1	1.00	0.90	0.90	1.1	1.037	0.990	0
1	1.00	0.84	0.84	1.9	2.064	1.020	1
1	0.65	0.42	0.27	0.5	0.114	1.014	0
1	1.00	0.75	0.75	1.0	1.322	1.004	0
1	0.50	0.44	0.22	0.6	0.114	0.990	0
1	1.00	0.63	0.63	1.1	1.072	0.986	1
1	1.00	0.33	0.33	0.4	0.176	1.010	0
1	0.90	0.93	0.84	0.6	1.591	1.020	0
1	1.00	0.58	0.58	1.0	0.531	1.002	1
1	0.95	0.32	0.30	1.6	0.886	0.988	0
1	1.00	0.60	0.60	1.7	0.964	0.990	1
1	1.00	0.69	0.69	0.9	0.398	0.986	1
1	1.00	0.73	0.73	0.7	0.398	0.986	0

3. CUBIC MAJORIZATION

Next we show that majorization can be used to obtain super-linear convergence under some conditions. Alternatively, we think of this particular use of majorization as a way to safeguard Newton's method. The technique and most of the analysis we use is due to Nesterov and Polyak [2003], although our implementation is quite different.

Let us go back to the general problem of minimizing f on $\mathbb{R}^n$. We can write, in obvious notation,

$$f(x) = f(y) + (x-y)'\partial f(y) + \frac{1}{2}(x-y)'\partial^2 f(y)(x-y) + \\ + \frac{1}{6}\partial^3 f(\xi)(x-y)\otimes(x-y)\otimes(x-y)$$

Now if we can find $K > 0$ such that

$$\partial^3 f(\xi)(x-y)\otimes(x-y)\otimes(x-y) \leq K\|x-y\|^3$$

then

$$g(x,y) = f(y) + (x-y)'\partial f(y) + \frac{1}{2}(x-y)'\partial^2 f(y)(x-y) + \frac{1}{6}K\|x-y\|^3$$

is a majorization function of f. Observe that for each y the function g_y is twice continuously differentiable, and has the same first and second derivatives at y as f. This immediately implies super-linear convergence, and in fact quadratic convergence in regular cases [Ortega and Rheinboldt, 1970, page 304].

There is another way to see this. Differentiating g shows that the algorithm can be written as

$$x^{(k+1)} = x^{(k)} - \left[\partial^2 f(x^{(k)}) + \frac{1}{2}K\|x^{(k)} - x^{(k+1)}\|\right]^{-1}\partial f(x^{(k)}). \tag{3}$$

This clearly shows cubic majorization is a regularization of Newton's method. It also shows that at a solution the derivative of the algorithmic map is zero, which implies super-linear convergence.

It may appear, at first sight, that minimizing the majorization function is a complicated non-convex problem, But we shall show, following Nesterov and Polyak [2003], that it can actually be solved by finding the solution of a univariate equation.

3.1. **Example.** Before we discuss how to implement the majorization algorithm associated with this majorization function, we first derive some

bounds K for the logistic negative log- likelihood. We find

$$\partial^3 f(\beta) = -\sum_{i=1}^{n} N_i \pi_i(\beta)(1-\pi_i(\beta))(1-2\pi_i(\beta)) z_i \otimes z_i \otimes z_i,$$

and since

$$\pi_i(\beta)(1-\pi_i(\beta))(1-2\pi_i(\beta)) \leq \frac{1}{18}\sqrt{3}$$

we can choose

$$K_T = \frac{1}{18}\sqrt{3}\sum_{i=1}^{n} N_i \|z_i\|^3.$$

A more precise bound is

$$K_S = \frac{1}{18}\sqrt{3}\|H\|_\infty.$$

where $\|H\|_\infty$ is the largest eigenvalue of

$$H = \sum_{i=1}^{n} N_i \|z_i\| z_i z_i'.$$

Observe that K_T is $\frac{1}{18}\sqrt{3}$ times the trace of H, and thus $K_T \geq K_S$.

An even sharper bound K_E can be computed by maximizing $\sum_{i=1}^n N_i |z_i' x|^3$ over $x'x = 1$. This can be done by linear majorization, because the function $|x|^3$ is convex and differentiable. The iterative algorithm for the bound computes the update for x by normalizing $\sum_{i=1}^n N_i (z_i' x)^2 \mathbf{sign}(z_i' x) z_i$ to unit length. Convergence is generally fast. For the Maxwell example, we have $K_T = 1169.280$, $K_S = 1163.663$, and $K_E = 1160.792$. Using smaller bounds will mean faster overall convergence, in particular in regions where quadratic approximation, and consequently Newton's method, is poor.

3.2. **Implementation.** Let us now look in more detail at minimizing the cubic majorization function in iteration k. Abbreviate

$$\begin{aligned} g &= \partial f(x^{(k)}), \\ H &= \partial^2 f(x^{(k)}), \\ d &= x^{(k+1)} - x^{(k)}, \end{aligned}$$

and rewrite Equation (3) as

$$d = -[H + \frac{1}{2}K\|d\|I]^{-1}g. \tag{4}$$

Define $\theta = \|d\|$. It follows from (4) that

$$g'[H + \frac{1}{2}K\theta I]^{-2}g = \theta^2. \tag{5}$$

The code in the Appendix first finds a solution of (5), and then computes d from (4). Equations of the form (5) have been studied in great detail. A good overview is in Conn et al. [2000, Chapter 7]. The left-hand side of (5) is plotted in Figure 3.2 for two cases, together with the function θ^2. The poles of the function are located at points where $-\frac{1}{2}K\theta$ is equal to one of the eigenvalues of H. In particular the branch on the right starts at $+\infty$ at the point corresponding with $-\lambda_{min}(H)$, the smallest eigenvalues, and then decreases (as a convex function) to zero. Thus it always intersects the quadratic θ^2, either once or twice. In there are two intersections, we want the positive one.

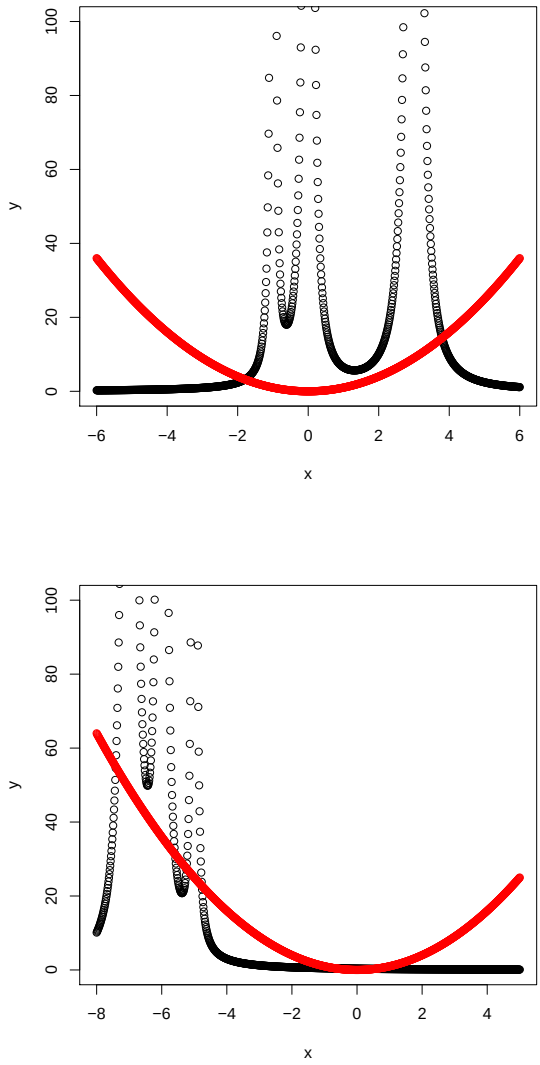
We use a straightforward version of Newton's method to solve (5), using a complete eigen-decomposition of H, and the information that the root must satisfy

$$\frac{K}{2}\theta \geq \max(0, -\lambda_{min}(H)).$$

Better methods are discussed by Melman [1997, 1998]. If the problems become larger than our examples, then methods based on explicit eigen-decomposition may no longer be practical. Several alternatives have been proposed in the literature.

3.3. **Computation.** If we use analyze the Maxwell example with third order majorization, we indeed find global convergence from any starting point. The number of iterations is around 30, which is not particularly impressive. In this example quadratic approximation is much better. This is even more true in the cancer remission example, in which cubic majorization takes 7296 iterations.

FIGURE 1. Solving the Secular Equation



One reason is that none of our bounds K is very good, unfortunately, and we find the algorithm still converges (and much faster) with a much smaller value of K. Better bounds may be possible, and convergence can possibly be improved by using non-uniform cubic majorization along the lines of the Jaakola-Jordan method. In the cancer remission example using the same type of over-relaxation which makes sense in the quadratic case leads to

3648 iterations. In any case, clearly research into possible improvement of cubic majorization is needed, and its performance on more complicated functions needs to be evaluated.

REFERENCES

D. Böhning and B.G. Lindsay. Monotonicity of Quadratic-approximation Algorithms. *Annals of the Institute of Statistical Mathematics*, 40(4): 641–663, 1988.

Andrew R. Conn, Nicholas I.M. Gould, and Philippe L. Toint. *Trust-Region Methods*. MPS-SIAM Series on Optimization. SIAM, Philadelphia, PA, 2000.

J. De Leeuw. Block Relaxation Methods in Statistics. In H.H. Bock, W. Lenski, and M.M. Richter, editors, *Information Systems and Data Analysis*, Berlin, 1994. Springer Verlag.

J. De Leeuw and W. J. Heiser. Multidimensional Scaling with Restrictions on the Configuration. In P.R. Krishnaiah, editor, *Multivariate Analysis, Volume V*, pages 501–522, Amsterdam, The Netherlands, 1980. North Holland Publishing Company.

P.J.F. Groenen, P. Giaquinto, and H.L Kiers. Weighted Majorization Algorithms for Weighted Least Squares Decomposition Models. Technical Report EI 2003-09, Econometric Institute, Erasmus University, Rotterdam, Netherlands, 2003.

W.J. Heiser. Convergent Computing by Iterative Majorization: Theory and Applications in Multidimensional Data Analysis. In W.J. Krzanowski, editor, *Recent Advanmtages in Descriptive Multivariate Analysis*, pages 157–189. Oxford: Clarendon Press, 1995.

T.S. Jaakkola and M. I. Jordan. Bayesian Parameter Estimation via Variational Methods. *Statistics and Computing*, 10:25–37, 2000.

K. Lange, D.R. Hunter, and I. Yang. Optimization Transfer Using Surrogate Objective Functions. *Journal of Computational and Graphical Statistics*, 9:1–20, 2000.

E.T. Lee. A Computer Program for Linear Logistic Regression Analysis. *Computer Programs in Biomedicine*, 4:80–92, 1974.

A.E. Maxwell. *Analysing Qualitative Data*. Chapman & Hall, London, GB, 1961.

A. Melman. A Unifying Convergence Analysis of Second-Order Methods for Secular Equations. *Mathematics of Computation*, 66:333–344, 1997.

A. Melman. Analysis of Third-order Methods for Secular Equations. *Mathematics of Computation*, 67:271–286, 1998.

Yu. Nesterov and B.T. Polyak. Cubic Regularization of a Newton Scheme and its Global Performance. CORE Discussion Paper 41, Catholic University of Louvain, 2003.

J. M. Ortega and W. C. Rheinboldt. *Iterative Solution of Nonlinear Equations in Several Variables*. Academic Press, New York, N.Y., 1970.

Appendix A. code

```
logReg<-function(tab,x,b,vv="newt",third=FALSE,relax=
    FALSE,itmax=1000,eps=1e-6,ibnd=3,verbose=FALSE) {
nn<-rowSums(tab); n1<-tab[,1]; itel<-1; x<-as.matrix(x
    ); m<-dim(x)[2]
vf<-crossprod(x,nn*x)/4; vs<-((eigen(vf)$values)[1])
    *diag(m)
if (third) kk<-bounds(nn,x)[ibnd]
repeat {
        xb<-as.vector(x%*%b); l<-1/(1+exp(-xb)); jj
            <-((2*l)-1)/(2*xb)
        u<-nn*l-n1; g<-colSums(u*x); mg<-max(abs(g))
        f<--(sum(n1*xb)+sum(nn*log(1-l)))
        cat("Iteration: ",formatC(itel,digits=6,width
            =6),
                "Function: ",formatC(f,digits=6,width
                    =12,format="f"),
                "Maxgrad: ",formatC(mg,digits=6,width
                    =12,format="f"),
                "\n")
        if ((mg < eps) || (itel == itmax)) break
```

```
        vn<-crossprod(x,(nn*l*(1-l))*x)
        if (vv=="newt") v<-vn
        if (vv=="jajo") v<-crossprod(x,(nn*jj)*x)
        if (vv=="fixt") v<-vf
        if (vv=="scal") v<-vs
        if (!third) d<--solve(v,g)
                else d<-cubic(g,vn,kk,verbose=verbose)
        if (relax) b<-b+2*d else b<-b+d
        itel<-itel+1
        }
e<-max(1-eigen(solve(v,vn))$values)
return(list(b=b,g=g,v=v,l=l,e=e))
}

cubic<-function(b,a,k,off=1,eps=1e-6,itmax=100,verbose
    =FALSE) {
e<-eigen(a); lb<-e$values; ev<-e$vectors; bb<-colSums(
    b*ev); k2<-k/2; l2<-2/k
xold<-(l2*max(0,max(-lb)))+off; itel=1
repeat {
        ff<-sum((bb^2)/((lb+k2*xold)^2))-(xold^2)
        gg<--2*(k2*sum((bb^2)/((lb+k2*xold)^3))+xold)
        xnew<-xold-(ff/gg)
        if (verbose) cat("Secular **",
                "Function: ",formatC(ff,digits=6,width
                    =12,format="f"),
                "Gradient: ",formatC(gg,digits=6,width
                    =12,format="f"),
                "Oldval:    ",formatC(xold,digits=6,
                    width=12,format="f"),
                "Newval:    ",formatC(xnew,digits=6,
                    width=12,format="f"),
                "\n")
        if ((abs(ff) < eps) || (itel == itmax)) break
```

```
        xold<-xnew; itel<-itel+1
        }
if (xnew < (12*max(0,max(-lb)))) stop("trouble")
y<--bb/(lb+k2*xnew)
return(as.vector(ev%*%y))
}


bounds<-function(n,z,x=rep(1,dim(z)[2]),eps=1e-6,itmax
    =100) {
b1<-sum(n*rowSums(z*z)^1.5)
b2<-eigen(crossprod(z,(n*sqrt(rowSums(z*z)))*z))
   $values[1]
itel<-1; x<-x/sqrt(sum(x*x)); fold<-sum(n*abs(z%*%x)
    ^3)
repeat {
        h<-colSums(as.vector(n*((z%*%x)^2)*sign(z%*%x)
            )*z); xup<-h/sqrt(sum(h^2)); fup<-sum(n
            *abs(z%*%xup)^3)
        if (((fup - fold) < eps) || (itel == itmax))
            break
        itel<-itel+1; x<-xup; fold<-fup
        }
return(c(b1,b2,fup)*sqrt(3)/18)
}
```

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