LECTURES IN LEAST SQUARES

Charles Van Loan

TR 76-279

May 1976

Department of Computer Science
Cornell University
Ithaca, New York 14853

LECTURES IN LEAST SQUARES

by

Charles Van Loan

Introduction

These lecture notes arose out of the numerical analysis seminar given at Cornell University in the Spring of 1976. The goal of the seminar was to acquaint a variety of researchers and undergraduates with the field of least squares computations. I hope the presentation will neither bore the expert nor discourage the layman.

Acknowledgements

Special mention should be made of the book, Solving Least Squares Problems by C.L. Lawson and R.J. Hanson. Lectures 1,2,3,4,6, and 7 drew heavily upon material from this fine book.

I am also indebted to Alan Cline and Gene Golub for discussions relating to the LSQI problem of Lecture 5. These discussions were so stimulating that I was forced to abandon my four page-per-lecture quota!

Finally, I am obliged to all the people who attended the seminar. They kept me honest with their questions and pleased with their interest.

LECTURE	TITLE
1.	Introduction to Linear Least Squares.
2.	Solving the Linear Least Square Problem.
3.	Post Decomposition Operations.
4.	Weighting and Equality Constraints.
5.	Some Modified Least Square Problems.
6.	Inequality Constraints.
7.	Algorithms for Linear Problems with Inequality Constraints.
8.	Some Sources of Least Square Problems.
9.	Basic Algorithms for Nonlinear Least Square Problems.
10.	Variations of the Gauss-Newton Iteration.
11.	Updating Methods.
12.	Introduction to Separable Nonlinear Least Squares.
13.	Variable Projection Algorithms.
14.	Constrained and Sparse Nonlinear Least Squares.

LECTURE 1. INTRODUCTION TO LINEAR LEAST SQUARES.

Notation.

R^{mxn} = real matrices having m rows and n columns

R^n = real n - vectors

I_n = nxn identity matrix

A^T = "A transpose" $A = (a_{ij}) \in R^{mxn} \rightarrow A^T = (a_{ji}) \in R^{nxm}$

$\|x\| = (x^Tx)^{1/2} = (x_1^2 + \ldots + x_n^2)^{1/2} \quad x \in R^n$

$\|A\| = \max\limits_{\|x\|=1} \|Ax\| \quad A \in R^{mxn}, \; x \in R^n$

$\|A\|_F = \left[\sum_{i=1}^{m} \sum_{j=1}^{n} a_{ij}^2\right]^{1/2} \quad A \in R^{mxn}$

The Linear Least Square Problem.

Given $A \in R^{mxn}$ $(m \geq n)$, and $b \in R^m$, find a vector $\hat{x} \in R^n$ such that

$$\|A\hat{x} - b\| = \min_{x \in R^n} \|Ax - b\|$$

Orthogonal Matrices.

The analysis and solution of the least square (LS) problem demands an expertise with orthogonal matrices:

$$Q \in R^{nxn} \text{ orthogonal } \Leftrightarrow Q^TQ = I_n$$

Orthogonal matrices preserve norms:

$$\|QAZ\| = \|A\| \qquad \|QAZ\|_F = \|A\|_F \qquad \|Qy\| = \|y\| \qquad \|QAx - Qb\| = \|Ax - b\|$$

Here, $A \in R^{mxn}$, $Q \in R^{mxm}$ (orthogonal), $Z \in R^{nxn}$ (orthogonal), $x \in R^n$, $b, y \in R^m$. Many of the algorithms we plan to discuss involve using special orthogonal matrices to introduce zeros in various arrays and vectors. These special orthogonal matrices come in two flavors:

(a) Householder Reflections.

Let $x \in R^n$ be an arbitrary nonzero vector. Define $v \in R^n$ by $v^T = (x_1 + \mathrm{sgn}(x_1)\|x\|, x_2, \ldots, x_n)$. Then $H = I_n - \frac{2}{\|v\|^2} vv^T$ is orthogonal and $(Hx)^T = (-\mathrm{sgn}(x_1)\|x\|, 0, 0, \ldots, 0)$

(b) Jacobi Rotations.

Let $x = (x_1, \ldots, x_n)^T$. For any (i,j) , $i < j$, we can find real numbers c and s so that $c^2 + s^2 = 1$ and $-sx_i + cx_j = 0$. Thus, the nxn matrix

$$Q = \begin{array}{cc} \left[\begin{array}{cccccc} 1 & 0 & \cdots & & \cdots & 0 \\ 0 & 1 & \cdots & & \cdots & 0 \\ \vdots & & \ddots & & & \vdots \\ \vdots & & c & & s & \vdots \\ \vdots & & & \ddots & & \vdots \\ \vdots & & -s & & c & \vdots \\ \vdots & & & & & \ddots \\ 0 & 0 & \cdots & & \cdots & 1 \end{array}\right] & \begin{array}{c} \\ \\ \\ i \\ \\ j \\ \\ \\ \end{array} \\ \begin{array}{cccccc} & & i & & j & \end{array} & \end{array}$$

is orthogonal and the j-th component of Qx is zero.

The Singular Value Decomposition.

A most penetrating analysis of the LS problem is possible through the application of the singular value decomposition (SVD) [20,22,25,32].

Theorem [14,19,25]

Let $A \in R^{m\times n}$ and assume $m \geq n$. There exist orthogonal $U \in R^{m\times m}$ and orthogonal $V \in R^{n\times n}$ such that

$$U^T A V = \operatorname{diag}(\mu_i) = \begin{bmatrix} \mu_1 & \cdots & 0 \\ \vdots & \mu_2 & 0 \\ & \ddots & \vdots \\ 0 & 0 & \mu_n \\ \vdots & \vdots & \vdots \\ 0 & 0 \cdots & 0 \end{bmatrix} \in R^{m\times n}$$

The μ_i are called the singular values of A and we shall assume throughout that they are ordered $\mu_1 \geq \mu_2 \geq \cdots \geq \mu_r > \mu_{r+1} = \cdots = \mu_n = 0$

Corollaries of the SVD.

1. $\operatorname{rank}(A) = r$
2. $\mu_n \|x\| \leq \|Ax\| \leq \mu_1 \|x\| \qquad \forall x \in R^n$.
3. If u_i and v_i denote the respective i-th columns of U and V , then

$$A = \sum_{i=1}^{n} \mu_i \, u_i v_i^T \quad \text{and} \quad A_k = \sum_{i=1}^{k} \mu_i u_i v_i^T \quad \text{is the closest}$$

rank k matrix to A measured in either $\|\cdot\|$ or $\|\cdot\|_F$. $(k \leq r)$

4. If $\mathcal{S}$ is the set of all rank deficient mxn matrices, then

$$\mu_n = \min_{S \in \mathcal{S}} \|A - S\|$$

5. The squares of the singular values are the eigenvalues of A^TA.

6. If $A + E$ has singular values $\hat{\mu}_1, \ldots, \hat{\mu}_n$, then $\sum_{i=1}^{n} (\mu_i - \hat{\mu}_i)^2 \leq \|E\|_F^2$

<u>Some Analysis of the LS Problem.</u>

Define the set X as follows:

$$X = \{ x \mid \|Ax - b\| \leq \|Az - b\| \quad \forall z \in R^n \}$$

1. $x \in X \to A^T(Ax - b) = 0$. (The residual $r = Ax - b$ is orthogonal to the columns of A, i.e. $r \in \text{range}(A)^{\perp}$)

   <u>Proof</u>. Let $x \in X$ and set $w = A^T(Ax - b)$. For all $\alpha \in R$ we must have $\|Ax - b\|^2 \leq \|A(x+\alpha w) - b\|^2 = \|Ax - b\|^2 + \alpha[\alpha \|Aw\|^2 + 2\|w\|^2]$

   If $w \neq 0$, there would be a violation of this inequality for small negative α. (The equations $A^T(Ax - b) = 0$ are called the "normal equations.")

2. X is convex and has a unique element of minimal norm $\hat{x}$:

$$\|\hat{x}\| < \|x\| \qquad \forall x \in X - \{\hat{x}\}$$

3. If $A = UDV^T$ is the SVD of A, then the LS sol'n $\hat{x} = \sum_{i=1}^{r} \frac{(u_i^Tb)}{\mu_i} v_i$

   <u>Proof</u>. Defining $y = V^Tx$ we find

$$\|Ax - b\|^2 = \|UDV^Tx - b\|^2 = \|Dy - U^Tb\|^2 = \sum_{i=1}^{r} (\mu_i y_i - u_i^Tb)^2 + \sum_{i=r+1}^{m} (u_i^Tb)$$

   It is clear that to minimize this sum of squares with a vector $\hat{y}$ of minimal norm, we must set $\hat{y}_i = u_i^Tb/\mu_i$ for $i = 1, \ldots, r$ and $\hat{y}_i = 0$ for $i = r+1, \ldots, n$. This in turn implies that $\hat{x} = V\hat{y} = \sum_{i=1}^{r} \hat{y}_i \hat{v}_i$

4. Let $A = UDV^T$ be the SVD of A and define the $n \times m$ diagonal matrix D^+ by

$$D^+ = \mathrm{diag}\left(\frac{1}{\mu_1}, \ldots, \frac{1}{\mu_r}, 0, \ldots, 0\right)$$

If $A^+ = VD^+U^T$ then $x = A^+b$ and A^+ is the unique solution of minimal $\|\cdot\|_F$ to

$$\min_{Y \in R^{n \times m}} \| AY - I_m \|_F$$

A^+ is called the "pseudo-inverse" of A. It is an obvious generalizati of A^{-1} .

5. If x and y are in X , then $A(x - y) = 0$. Thus, if $\mathrm{rank}(A) = n$, then $X = \{\hat{x}\}$.

6. $\min \|Ax - b\|$ = distance of b to the range space of A .

<u>Sensitivity of the LS Problem</u>. [25,p.50]

Consider the following two least square problems:

$$\min \| Ax - b \|$$

$$\min \| (A+\Delta A)x - (b+\Delta b) \| \qquad \|\Delta A\| \leqslant \varepsilon \|A\| \qquad \|\Delta b\| < \varepsilon \|b\|$$

If these have solutions $\hat{x}$ and $\hat{y}$ respectively and if $\mathrm{rank}(A) = \mathrm{rank}(A+\Delta A) = n$, then by ignoring second order terms and small constants we have:

$$\| \hat{x} - \hat{y} \| \leqslant \varepsilon \left[\mathrm{cond}(A) \| \hat{x} \| + \|A^+\| \|b\| + \mathrm{cond}(A)^2 \| r \| \right]$$

where

2 $$\mathrm{cond}(A) = \frac{\mu_1}{\mu_n} \qquad\qquad r = b - A\hat{x}$$

As in the case of linear equations (see Forsythe and Moler[14]), the LS problem is sensitive to changes in A and b when the condition number is large. But more than that, we see that when $r \neq 0$, the $\mathrm{cond}(A)^2$ factor is dominant in the above upper bound. As an illustration of extreme sensitivit when the condition is large and the residual nonzero, consider:

$$A = \begin{pmatrix} 1 & 0 \\ 0 & \varepsilon \\ 0 & 0 \end{pmatrix} \quad \Delta A = \begin{pmatrix} 0 & 0 \\ 0 & -\varepsilon \\ 0 & \varepsilon \end{pmatrix} \quad b = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \quad \Delta b = \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix} \quad \hat{x} = \begin{pmatrix} 0 \\ 0 \end{pmatrix} \quad \hat{y} = \begin{pmatrix} 0 \\ 1/\varepsilon \end{pmatrix} \quad \mathrm{cond}(A) = \frac{1}{\varepsilon} \quad r = ($$

LECTURE 2. SOLVING THE LINEAR LEAST SQUARE PROBLEM

SVD Method

We have already seen that the solution to the LS problem can be written down in terms of the output of the SVD of A:

$$(1) \qquad \hat{x} = \sum_{i=1}^{r} \frac{(u_i^T b)}{\mu_i} v_i \qquad \begin{aligned} A &= U \operatorname{diag}(\mu_i) V^T \\ U &= [u_1 | \cdots | u_m] \\ V &= [v_1 | \cdots | v_n] \end{aligned}$$

One can thus use the Golub-Reinsch SVD algorithm [22] to solve the LS problem. Some details of this practical process should be mentioned. To begin with, rarely will a computed singular value be flat zero. It is thus a difficult problem to determine how many terms to take in the summation (1). A standard practice is to adopt some tolerance $\delta > 0$ and thereafter regard a computed singular value as zero if it is less than δ . The computed solution to the LS problem thus has the form

$$\tilde{\hat{x}} = \sum_{i=1}^{k} \frac{\tilde{u}_i^T \tilde{b}}{\tilde{\mu}_i} \qquad \left(\tilde{\mu}_i \leqslant \delta \quad \forall \quad i > k \right)$$

(The squiggles denote computed quantities.) Roughly speaking, the bigger δ is the larger will be the residual $r = b - Ax$ and the smaller will be $\tilde{\hat{x}}$. The choice of δ can obviously have a profound effect on the computed solution when A is (nearly) rank deficient. But as we have seen, there is an inherent sensitivity of the LS problem in this case implying that any method of solution will have problems. The merit of the SVD approach is that the user is supplied with a great deal of information about the structure of A upon which sensible decisions as to rank can be based. One final point, through a rather complicated overwriting scheme, one needs no more than the original mxn array containing A.

QR Methods.

Let us first assume that the matrix $A \in R^{m\times n}$ has rank n. We will show that there exists an orthogonal Q such that $QA = \begin{pmatrix} R \\ 0 \end{pmatrix}$ where R is an nxn upper triangular matrix. The relevance of this decomposition to the LS problem is derived from the identity

$$\|Ax - b\|^2 = \|Q^TAx - Q^Tb\|^2 = \left\| \begin{pmatrix} R \\ 0 \end{pmatrix} x - \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \right\|^2 = \|Rx - b_1\|^2 + \|b_2\|^2$$

Here $Q^Tb = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$, $b_1 \in R^n$, $b_2 \in R^{m-n}$. Since rank(A) = n by assumption, R^{-1} exists and thus $\|Ax - b\|$ is minimized by selecting $\hat{x} = R^{-1}b_1$. In other words, $\hat{x}$ is the solution to the triangular system $Rx = b_1$.

The algorithm which determines Q is based upon the application of Householder reflections. (See Lecture 1.) It goes as follows

<u>Algorithm QR</u>

For k = 1,2,...,n (or n-1 if m=n)

(a) Find Householder reflection $\tilde{Q}_k$ of order (m-k+1) such that

$$\tilde{Q}_k \begin{pmatrix} a_{kk} \\ \vdots \\ a_{mk} \end{pmatrix} = \begin{pmatrix} * \\ 0 \\ \vdots \\ 0 \end{pmatrix} \Big\} \text{ m-k zeros}$$

(b) Update A:

$$A \leftarrow \begin{pmatrix} I_{k-1} & 0 \\ 0 & \tilde{Q}_k \end{pmatrix} A \equiv Q_k A$$

One can show that $(Q_nQ_{n-1} \cdots Q_2Q_1)A = \begin{pmatrix} R \\ 0 \end{pmatrix}$. To see why this is so, consider the k-th step $A \quad Q_kA$. Before this update is executed, A has the following form

$$A = \begin{matrix} x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & \boxed{x} & x \\ 0 & 0 & \boxed{x} & x \\ 0 & 0 & \boxed{x} & x \\ 0 & 0 & \boxed{x} & x \end{matrix} \qquad (m=6,\ n=4,\ k=3)$$

After finding $\tilde{Q}_3$ so $\tilde{Q}_3 \begin{pmatrix} x \\ x \\ x \\ x \end{pmatrix} = \begin{pmatrix} x \\ 0 \\ 0 \\ 0 \end{pmatrix}$ we update and obtain

$$A \leftarrow \begin{pmatrix} I_2 & 0 \\ 0 & \tilde{Q}_3 \end{pmatrix} A = \begin{matrix} x & x & x & x \\ 0 & x & x & x \\ 0 & 0 & \boxed{x} & x \\ 0 & 0 & \boxed{0} & x \\ 0 & 0 & \boxed{0} & x \\ 0 & 0 & \boxed{0} & x \end{matrix}$$

In the final analysis, we find that A is upper triangular ($a_{ij}=0$, $i > j$) after n such steps and that $Q = (Q_1 Q_2 \cdots Q_n)$. Of course, the Q_k's do not have to be accumulated but rather just applied to b.

A more accurate and illuminating variation of this process results if we do column pivoting. This means at the k-th stage we examine the norms of $(a_{kj}, \ldots, a_{mj})$ for $j=k,\ldots,n$. If the max occurs for subscript p, we interchange columns p and k of the current array and proceed as in the original form of the algorithm. The resulting decomposition is

$$Q^T A P = \begin{pmatrix} R \\ 0 \end{pmatrix} \qquad \text{Q Orthogonal, P permutation}$$

and, if rank(A) =n , $\hat{x} = PR^{-1}b_1$ where $Q^T b = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix}$ as before. In exact arithmetic, rank deficiency can be detected by the column pivoting algorithm. If rank(A) = r < n then after r steps the array will have the following form:

$$\begin{pmatrix} R_{11} & R_{12} \\ 0 & 0 \end{pmatrix} \begin{matrix} \}\ r \text{ rows} \\ \}\ m-r \text{ rows} \end{matrix} \qquad R_{11} = \nabla \in R^{r \times r}$$

One can do subsequent column operations to solve for $\hat{x}$. However, in the (nearly) rank deficient case, one should use SVD because rounding errors can pervert the rank detection ability of the QR+pivoting algorithm.

<u>Normal Equations Method</u>

If rank(A) = n , then $\hat{x} = (A^T A)^{-1} A^T b$. This follows from the fact that any solution to the LS problem satisfies the normal equations $A^T(Ax-b)=0$. In more detail, this technique involves

(a) Forming $C = A^T A$ (symmetric positive definite)

(b) Factoring $C = LL^T$ ($L = \Delta$ = Cholesky triangle)

(c) Solving $Ly = A^T b$ and $L^T x = y$.

Notice that $\text{cond}(A^TA) = \text{cond}(A)^2$. This doubling of the condition number can render our problem insoluble in a floating point environment and is what makes the normal equations method unsatisfactory for poorly conditione examples. To partially compensate for this it customary to do all the suggested computations in double precision.

Gram-Schmidt Methods [3]

These methods compute the factorization $A = GS$ where $G \in R^{m\times n}$ has orthogonal columns and $S \in R^{n\times n}$ is upper triangular matrix with ones along the diagonal. (We are assuming that rank(A)=n.) It is straight forward to verify that the solution to the LS problem is given by $x = S^{-1}D^{-1}G^Tb$ where $D = \text{diag}(\|g_i\|^2) = G^TG$. There are two ways of computing the $A = GS$ factorization- Classical Gram-Schmidt (CGS) which finds S column at a time and Modified Gram-Schmidt (MGS) which finds S row at a time. CGS breaks down when the columns of A are close to dependent whereas MGS has stabilit properties similar to the QR factorization above. CGS should never be preferred and MGS holds practically no advantage over QR techniques. Nevertl less,we describe these processes denoting the i-th columns of A and G by a_i and g_i respectively:

CGS

$g_1 = a_1$

For j=2,3,...,n

$s_{ij} = a_j^Tg_i/g_i^Tg_i \quad (i=1,..,j-1)$

$g_j = a_j - \sum_{i=1}^{j-1} s_{ij}g_i$

MGS

For j=1,..,n $a_j^{(1)} = a_j$

For i=1,..,n

$g_i = a_i^{(i)} \qquad d_i^2 = g_i^Tg_i$

For j=i+1,..,n

$s_{ij} = g_i^Ta_j^{(i)}/d_i^2$

$a_j^{(i+1)} = a_{ij}^{(i)} - s_{ij}g_i$

Comparison of Work

If we count multiplicative operations in the above algorithms for the LS problem, we find (ignoring low order terms)

SVD	$2mn^2 + 4n^3$
QR	$mn^2 - n^3/3$
NE	$mn^2/2 + n^3/6$
MGS	mn^2

LECTURE 3. POST DECOMPOSITION OPERATIONS

Iterative Improvement in Linear Equations [14]

In the linear equations problem $Ax = b$, an approximate, computed solution $x^{(1)}$ can, under appropriate conditions, be improved. The process is called "iterative improvement" and if implemented with Gaussian elimination with pivoting, goes like this:

1. Compute $PA = LU$ = ◺◹
2. Solve $Ly = Pb$; $Ux^{(1)} = y$
3. For $k = 1,2,\ldots$
   (a) Compute $r^{(k)} = b - Ax^{(k)}$ in double the machine precision used in step 1.
   (b) Solve $Ly^{(k)} = Pr^{(k)}$; $Uz^{(k)} = y^{(k)}$
   (c) Set $x^{(k+1)} = x^{(k)} + z^{(k)}$
   } single precision

Suppose the computer arithmetic used has base β and that $\text{cond}(A) \cong \beta^p$ If step 1 is executed using s-digit floating point arithmetic, then a heuristic argument shows that $x^{(k)}$ will have about $k(s-p)$ correct base β digits. This is the meaning of the statement "iterative improvement works provided A is not too badly conditioned". Clearly, A is badly conditioned relative to the machine precision if $\text{cond}(A) \cong \beta^s$. A nice side benefit of doing iterative improvement is that an estimate of cond(A) can be obtained from the heuristic equation $\text{cond}(A) = \beta^s \|z^{(1)}\| / \|x^{(1)}\|$.

Least Square Iterative Improvement I. [23]

We discuss this first method assuming an approximate solution $x^{(1)}$ to $\min \|Ax - b\|$ has been obtained through computation of $A = Q\binom{R}{0}$ in single precision. We also assume that A has rank n . The improvement of $x^{(1)}$ is performed as follows:

For k=1,2,..

(a) Form $r^{(k)} = b - Ax^{(k)}$ in double precision.

(b) Solve $Rz^{(k)} = y^{(k)}$ where $Q^T r^{(k)} = \begin{pmatrix} y^{(k)} \\ w^{(k)} \end{pmatrix} \begin{matrix} \}n \\ \}m-n \end{matrix}$

(c) $x^{(k+1)} = x^{(k)} + z^{(k)}$

Notice that $z^{(k)}$ minimizes $\|Az - r^{(k)}\|$. It can be shown that the success of this technique depends upon how small $\text{cond}(A)^2 \min\|Ax - b\|$ is . The sputtering of this algorithm on nonzero residual problems leads one to consider an alternative improvement process.

<u>Least Square Iterative Improvement II</u>. [4]

This method is based upon the fact that if

$$\begin{pmatrix} I & A \\ A^T & 0 \end{pmatrix} \begin{pmatrix} r \\ x \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix}$$

then x minimizes $\|Ax - b\|$. (Proof: $A^T r = 0$; $r = b-Ax$; $A^T(b-Ax) = 0$.)

Suppose $A = Q\binom{R}{0}$. Then $A^TA = R^TR$ and

$$\begin{pmatrix} I & A \\ A^T & 0 \end{pmatrix} = \begin{pmatrix} I & 0 \\ A^T & R^T \end{pmatrix} \begin{pmatrix} I & A \\ 0 & -R \end{pmatrix} =$$

Thus, it is easy to solve linear systems involving the above augmented matrix once we have the QR factorization of A. Notice that one does not need an $(m+n)\times(m+n)$ array to do this. With these facts in mind we present a second LS iterative improvement algorithm. Suppose $x^{(1)}$ is an approximate solution and $r^{(1)} = b - Ax^{(1)}$. We then improve the accuracy of $x^{(1)}$ as follows:

For $k = 1,2,...$

(a) Form $\begin{pmatrix} f^{(k)} \\ g^{(k)} \end{pmatrix} = \begin{pmatrix} b \\ 0 \end{pmatrix} - \begin{pmatrix} I & A \\ A^T & 0 \end{pmatrix} \begin{pmatrix} r^{(k)} \\ x^{(k)} \end{pmatrix}$ (double precision;

(b) Solve: $\begin{pmatrix} I & A \\ A^T & 0 \end{pmatrix} \begin{pmatrix} \Delta r^{(k)} \\ \Delta x^{(k)} \end{pmatrix} = \begin{pmatrix} f^{(k)} \\ g^{(k)} \end{pmatrix}$

(c) Update: $\begin{pmatrix} r^{(k+1)} \\ x^{(k+1)} \end{pmatrix} = \begin{pmatrix} r^{(k)} \\ x^{(k)} \end{pmatrix} + \begin{pmatrix} \Delta r^{(k)} \\ \Delta x^{(k)} \end{pmatrix}$

The convergence of this process depends on $\text{cond}(A)$, not $\text{cond}(A)^2$. Furthermore, the success of the iteration does not depend upon the LS problem having a zero residual. The superiority of this algorithm to the previous o is derived from the fact fact that the residual, and not just the solution, are improved. Method 2 will give full single precision accuracy if cond(A) is not too much bigger than β^s . One can fairly well conclude that if the first correction vector is large, then cond(A) is big -- valuable information.

In conclusion we mention that both improvement shemes (a) require storage in addition to the original mxn array and (b) can be implemented with factorizations other than Q-R.

Column Alteration.

Adding columns to the current matrix A is a problem which arises when one is doing curve fitting (see Lecture 8) with polynomials of ever increasing degree. Column addition/deletion transpires in certain algorithms for constrained least squares (see Lecture 7).

Adding Columns with Normal Equations.

Suppose $A^TA = LL^T = \Delta\nabla$. If $Lw = A^Tv$, $c^2 = v^Tv - w^Tw$, and $\bar{A} = [A|v]$, then

$$\bar{A}^T\bar{A} = \begin{pmatrix} A^T \\ \hline v^T \end{pmatrix} [A|v] = \begin{pmatrix} A^TA & A^Tv \\ v^TA & v^Tv \end{pmatrix} = \begin{pmatrix} L & 0 \\ w^T & c \end{pmatrix} \begin{pmatrix} L^T & w \\ 0 & c \end{pmatrix} \equiv \bar{L}\,\bar{L}^T$$

Adding and Deleting Columns with QR.

If $Q^TA = \begin{pmatrix} R \\ 0 \end{pmatrix}$ then $Q^T[A|v] = \begin{pmatrix} R & w \\ 0 & y \end{pmatrix}$ where $v \in R^m$; $w \in R^n$; and $y \in R^{m-n}$. If Z is an m-n Householder reflection which zeros all but the first entry in y, then

$$\left[\begin{pmatrix} I_n & 0 \\ 0 & Z \end{pmatrix} Q^T\right] [A|v] = \begin{pmatrix} R & w \\ 0 & d \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} \bar{R} \\ 0 \end{pmatrix}$$

is the desired factorization of $[A|v]$. This technique normally requires storage of Q in some form or other. In certain problems this can be avoided. For example, when the columns of the "A" matrix are taken from a known "pool" of vectors $\{v_1, \ldots, v_p\}$, one can update this pool with the various orthogonal matrices as they arise. This is actually what happens in the nonnegative least square algorithm NNLS (see Lecture 7).

Now suppose $Q^T A = Q^T [a_1 | a_2 | \cdots | a_n]$ and that we want to minimize $\|\bar{A}z - b\|$ where $\bar{A} = [a_1 | \cdots | a_{k-1} | a_{k+1} | \cdots | a_n]$. Notice that $Q^T \bar{A}$ is upper triangular except for subdiagonal elements in columns $k, \ldots, n-1$. However, these can be dealt with by applying a sequence of Jacobi rotations:

$$Q^T \bar{A} = \begin{matrix} x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & 0 & \boxed{x} & x & x \\ 0 & 0 & 0 & \boxed{x} & x & x \\ 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & x \\ 0 & 0 & 0 & 0 & 0 & 0 \end{matrix} \rightarrow \begin{matrix} x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & \boxed{x} & x \\ 0 & 0 & 0 & 0 & \boxed{x} & x \\ 0 & 0 & 0 & 0 & 0 & x \\ 0 & 0 & 0 & 0 & 0 & 0 \end{matrix} \rightarrow \begin{matrix} x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & \boxed{x} \\ 0 & 0 & 0 & 0 & 0 & \boxed{x} \\ 0 & 0 & 0 & 0 & 0 & 0 \end{matrix} \rightarrow \begin{matrix} x & x & x & x & x & x \\ 0 & x & x & x & x & x \\ 0 & 0 & x & x & x & x \\ 0 & 0 & 0 & x & x & x \\ 0 & 0 & 0 & 0 & x & x \\ 0 & 0 & 0 & 0 & 0 & x \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{matrix}$$

Adding Rows With QR.

These problems arise when A has too many rows to store at once and/or it is desireable to process observations sequentially. To illustrate the basic idea, suppose we want to minimize $\|Ax - b\|$ where

$$A = \begin{pmatrix} C \\ D \end{pmatrix} \begin{matrix} \} \ m_1 \text{ rows} \\ \} \ m_2 \text{ rows} \end{matrix} \qquad b = \begin{pmatrix} c \\ d \end{pmatrix} \begin{matrix} \} m_1 \\ \} m_2 \end{matrix}$$

1. Apply QR to $\{C, c\}$:

$$Q^T C = \begin{pmatrix} R \\ 0 \end{pmatrix} \qquad Q^T c = \begin{pmatrix} c_1 \\ c_2 \end{pmatrix} \begin{matrix} \} n \\ \} m_1 - n \end{matrix}$$

Do not store Q. Notice we have equipment to minimize $\|Cz - c\|$.

2. Minimize $\left\| \begin{pmatrix} R \\ D \end{pmatrix} x - \begin{pmatrix} c_1 \\ d \end{pmatrix} \right\|$ using QR. Thus we want an orthogonal Z so

$$Z \begin{pmatrix} R \\ D \end{pmatrix} = \begin{pmatrix} \bar{R} \\ 0 \end{pmatrix} \begin{matrix} \} \ n \text{ rows} \\ \} \ m_2 \text{rows} \end{matrix}$$

. Z can be given as a sequence of Jacobi rotations (although Householder reflections are used in practice):

$$\begin{pmatrix} R \\ D \end{pmatrix} = \begin{matrix} x & x & x \\ 0 & x & x \\ 0 & 0 & x \\ \boxed{x} & x & x \\ \boxed{x} & x & x \end{matrix} \rightarrow \begin{matrix} \boxed{x} & x & x \\ 0 & x & x \\ 0 & 0 & x \\ \boxed{x} & x & x \\ 0 & x & x \end{matrix} \rightarrow \begin{matrix} x & x & x \\ 0 & x & x \\ 0 & 0 & x \\ 0 & \boxed{x} & x \\ 0 & \boxed{x} & x \end{matrix} \rightarrow \begin{matrix} x & x & x \\ 0 & \boxed{x} & x \\ 0 & 0 & x \\ 0 & \boxed{x} & x \\ 0 & 0 & x \end{matrix} \rightarrow \begin{matrix} x & x & x \\ 0 & x & x \\ 0 & 0 & x \\ 0 & 0 & \boxed{x} \\ 0 & 0 & \boxed{x} \end{matrix} \rightarrow \begin{matrix} x & x & x \\ 0 & x & x \\ 0 & 0 & \boxed{x} \\ 0 & 0 & \boxed{x} \\ 0 & 0 & 0 \end{matrix} \rightarrow \begin{matrix} x & x & x \\ 0 & x & x \\ 0 & 0 & x \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{matrix}$$

The solution to the problem in 2 is the solution to the original problem because

$$\|Ax - b\| = \left\| \begin{pmatrix} R \\ 0 \\ D \end{pmatrix} x - \begin{pmatrix} c_1 \\ c_2 \\ d \end{pmatrix} \right\| = \left\| \begin{pmatrix} R \\ D \\ 0 \end{pmatrix} x - \begin{pmatrix} c_1 \\ d \\ c_2 \end{pmatrix} \right\|$$

We refer the reader to Lawson and Hanson [25] for these and other updating techniques.

LECTURE 4. WEIGHTING AND EQUALITY CONSTRAINTS.

Row Weighting. [25,p.183]

Consider the problem of minimizing $\|G(Ax - b)\|$ where G is an mxm nonsingular matrix. The solution depends upon G in a non-trivial way. However, if $G = \mathrm{diag}(g_1,\ldots,g_m)$ then

$$\|G(Ax-b)\|^2 = \sum_{i=1}^{m} g_i^2(a_i^T x - b_i)^2 \qquad a_i^T = \text{i-th row of A}$$

showing that $(a_i^T x - b_i)$ is more likely to be small when g_i is big. In statistical applications, it is typical to choose G so that the uncertainties in the transformed observation vector Gb are about the same. A simple illustration would be to set $G = \mathrm{diag}(\frac{1}{\sigma_1},\ldots,\frac{1}{\sigma_m})$ where σ_i is the standard deviation of the uncertainty in b_i. (Little uncertainty $\Rightarrow \frac{1}{\sigma_i}$ large $\Rightarrow$ $a_i^T x - b_i$ small .) When the uncertainties in b are correlated, a more complicated G may be involved.

In other settings, one may be interested in achieving near consistancy in some of the rows of Ax-b and appropriate selection of the weights g_i can help accomplish this.

The solution to min $\|G(Ax-b)\|$ may be computed by applying any of our previous techniques with GA as our matrix and Gb as our right hand side.

Column Weighting. [25,p.185]

Consider the problem of minimizing $\|AHx - b\|$ where H is a nonsingular nxn matrix. This is nothing more than a change of variables. One might choose H to make $\mathrm{cond}(AH) < \mathrm{cond}(A)$. Another point of view is to choose $H = \mathrm{diag}(h_1,\ldots,h_n)$ so as to even out the uncertainties in the columns of A.

Weighting in General.

Let P be an nxn positive definite matrix. Let $\|x\|_P = (x^T Px)^{1/2}$. Define Q to be P-orthogonal if $Q^T PQ = I_n$. The general weighted least squares problem can be regarded as that of minimizing $\|Ax - b\|_S$ such that $\|x\|_T$ is min. Here S and T are mxm and nxn positive definite matrices respectively. The solution of the generalized weighted least squares problem can be given in terms of a generalized singular value decomposition. To wit, it can be shown [34] there exists an S-orthogonal U and a T-orthogonal V such that

$$U^{-1}AV = \mathrm{diag}(a_1,\ldots,a_r,0,\ldots,0) \qquad (a_i > 0)$$

and that the solution to $\min \|Ax-b\|_S$ $(\|x\|_T = \min)$ is given by

$$\hat{x} = V\,\mathrm{diag}(\frac{1}{a_1},\ldots,\frac{1}{a_r},0,\ldots,0)\; U^{-1}b$$

The QR factorization admits a similar generalization. Algorithms which exploit "Fast Givens" transformations can be used to compute these factorizations when S and T are diagonal [15].

The B-Singular Value Decomposition [34]

We now look at another generalization of the SVD which is useful in examining a class of least squares problems.

Theorem (BSVD)

Let $A \in R^{m\times n}$ $(m \geq n)$ and $B \in R^{p\times n}$. There exist orthogonal U (mxm) and V (pxp) and an invertible X (nxn) such that

$$U^TAX = D_A = \mathrm{diag}(\alpha_1,\ldots,\alpha_n) \qquad (\alpha_i \geq 0)$$

$$V^TBX = D_B = \mathrm{diag}(\beta_1,\ldots,\beta_r,0,\ldots,0)$$

where $\beta_1 \geq \beta_2 \geq \cdots \geq \beta_r > 0$ and $r = \mathrm{rank}(B) \leq q = \min(p,q)$

Proof.

We prove this theorem constructively for the case $\mathrm{rank}(A) = n$. Some slight modifications are necessary when this is not the case.

1. Let $Q\begin{pmatrix}A\\B\end{pmatrix}Z = \begin{pmatrix}D\\0\end{pmatrix}$ $D = \mathrm{diag}(c_i) \in R^{n\times n}$ be the SVD of $\begin{pmatrix}A\\B\end{pmatrix}$.
   Set $A_1 = AZD^{-1}$ and $B_1 = BZD^{-1}$.
2. Let $V^TB_1Z_1 = \mathrm{diag}(\beta_1,\beta_2,\ldots,\beta_r,0,\ldots,0) = D_B$ be the SVD of B_1.
3. Let $U^T(A_1Z_1) = \begin{pmatrix}R\\0\end{pmatrix}$ be the QR factorization of A_1Z_1.

Let $X = ZD^{-1}Z_1$. We see that $V^TBX = D_B$. Several observations are necessar to show that U^TAX is also diagonal. Notice from (1) that $A_1^TA_1 + B_1^TB_1 = I_n$ Thus $(A_1Z_1)^T(A_1Z_1) = Z_1^T(I_n - B_1^TB_1)Z_1 = I - D_B^TD_B = \diagdown$ so the columns of AZ_1

and hence R are orthogonal. But an invertible, upper triangular matrix with orthogonal columns must be diagonal. Hence, $U^TAX = U^TA_1Z_1 = \binom{R}{0}$ is diagonal as promised. When rank(A) $<$ n, one must do column pivoting in (3). When rank$\binom{A}{B} < n$, one must determine, and then remove, the common null space of A and B in step (1).

<u>Relevance of the BSVD to the Equality Constrained LS Problem</u>

Let $A \in R^{m \times n}$ $(m \geq n)$, $B \in R^{p \times n}$ $(p \leq n)$, $b \in R^m$, and $d \in R^p$. Consider the problem

$$\text{(LSE)} \qquad \min_{Bx=d} \|Ax - b\|$$

This problem has a solution provided there exists a y so $By = d$. It has a unique solution provided the null spaces of A and B have a trivial intersection. Let $U^TAX = D_A = \text{diag}(\alpha_i)$ and $V^TBX = D_B = \text{diag}(\beta_i)$ be the BSVD. Then

$$\min_{Bx=d} \|Ax-b\| \equiv \min_{V^TBXX^{-1}x=V^Td} \|U^TAXX^{-1}x - U^Tb\| \equiv \min_{D_By=\tilde{d}} \|D_Ay - \tilde{b}\|$$

where $\tilde{b} = U^Tb$, $\tilde{d} = V^Td$, and $y = X^{-1}x$. It is easy to see what y must be to solve the transformed problem:

$$\hat{y}_i = \begin{cases} \tilde{d}_i/\beta_i & i = 1,\dots,r \\ \tilde{b}_i/\alpha_i & \alpha_i \neq 0 \quad i=r+1,\dots,n \\ 0 & \alpha_i = 0 \quad \text{or } i > n \text{ .} \end{cases}$$

Thus $\hat{x} = X\hat{y}$ solves the original problem. We see that the sensitivity of the problem depends upon the smallness of $\beta_1,\dots,\beta_r$ and $\alpha_r,\dots,\alpha_n$. Although the BSVD can be used to solve the LSE problem, there are much faster methods which are perfectly satisfactory for well-conditioned $\binom{A}{B}$.

<u>Other Methods for Solving LSE</u> [25]

(a) As a weighted Least Squares Problem.

By our remarks which attended row weighting, we see that for small ε, the $x(\varepsilon)$ which minimizes $\left\|\begin{pmatrix}\varepsilon A \\ B\end{pmatrix}x - \begin{pmatrix}\varepsilon b \\ d\end{pmatrix}\right\|$ will approximately solve LSE.

The BSVD can be used to measure the exact difference between $x(\varepsilon)$ and the exact solution to LSE. The advantage of treating LSE as a weighted LS problem is that no special subroutines are needed. On the other hand, if high accuracy is required, there may be numerical problems because of the small ε typically required.

(b) <u>Elimination of Variables</u> [25]

This method is based upon the fact that we are only allowed to minimize $\|Ax - b\|$ by vectors x of the form $x = u + v$ where $Bu = d$ and $Bv = 0$. We will assume that $\text{rank}(A) = n$ and $\text{rank}(B) = p$. Suppose $BQ = [L|O]$ where Q is an $n \times n$ orthogonal matrix and L is a $p \times p$ (invertible) lower triangular matrix. This is just the transpose of our ordinary Q-R factorization of B^T. Now all solutions to $Bx = d$ are of the form

$$x = Q\begin{pmatrix} L^{-1}d \\ y_2 \end{pmatrix} \qquad y_2 \text{ arbitrary } n\text{-}p \text{ vector}$$

because

$$d = Bx = BQQ^Tx = [L|O]\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \qquad Q^Tx = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \begin{matrix} \}p \\ \}n\text{-}p \end{matrix}$$

Partitioning $AQ = [A_1|A_2]$ $(A_1 \in R^{m\times p}, A_2 \in R^{m\times(n-p)})$ we find

$$\|Ax - b\| = \|AQQ^Tx - b\| = \left\| [A_1|A_2]\begin{pmatrix} L^{-1}d \\ y_2 \end{pmatrix} - b \right\|$$

$$= \|A_2y_2 - (b - A_1L^{-1}d)\|$$

Hence, y_2 is determined by the minimization of an unconstrained LS problem. In summary, with the same notation as above:

1. Factor $BQ = [L|O]$ and set $AQ = [A_1|A_2]$, $y_1 = L^{-1}d$.
2. Let y_2 minimize $\|A_2z - (b - A_1y_1)\|$
3. $\hat{x} = Q\begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$ solves LSE.

<u>A Note on Work</u>

If we count multiplicative operations and throw away low order terms, then the weighted LS method requires $n^2(m-\frac{n}{3}) + n^2p$ operations while the elimination of variables demands $n^2(m-\frac{n}{3}) + n^2p + p^2m$.

LECTURE 5. SOME MODIFIED LEAST SQUARE PROBLEMS

Lagrange Multipliers

It is worthwhile to review this important tool from advanced calculus. Suppose we want to find the stationary points of a function $f(x)$ subject to $g(x) = 0$. Here $f: R^n \to R$ and $g: R^n \to R^p$. The method of Lagrange multipliers for this problem is roughly as follows:

(a) Set $h(x) = f(x) + \lambda^T g(x) \qquad \lambda \in R^p$

(b) Solve for x by manipulating the equations

$$g(x) = 0$$
$$\frac{\partial h}{\partial x_i} = 0 \qquad i = 1,\ldots,n$$

By way of illustration we apply this technique to

$$\text{(LSE)} \qquad \min_{Bx=d} \|Ax - b\| \qquad A \in R^{m\times n},\ b \in R^m,\ B \in R^{p\times n},\ d \in R^p$$

We are thus led to the functional

$$h(x) = \frac{1}{2}\|Ax - b\|^2 + \lambda^T(Bx - d)$$

The "1/2" is just to simplify the subsequent derivatives. In vector notation the n equations $\frac{\partial h}{\partial x_i} = 0$ become

$$A^TAx - A^Tb + B^T\lambda = 0$$

By combining this result with $Bx = d$ and the definition $r = b - Ax$ we see that

$$\begin{pmatrix} 0 & 0 & B \\ 0 & I_m & A \\ B^T & A^T & 0 \end{pmatrix} \begin{pmatrix} \lambda \\ r \\ x \end{pmatrix} = \begin{pmatrix} d \\ b \\ 0 \end{pmatrix}$$

Provided $NS(A) \cap NS(B) = \{0\}$, the above system is invertible. (NS = null space.) So much for this illustration of Lagrange multipliers. However, we remark that the above augmented system figures in an iterative improvement scheme for problem LSE [39].

Least Squares with Quadratic Constraints

Suppose $A \in R^{m\times n}$, $B \in R^{p\times n}$, $b \in R^m$, $d \in R^p$ and that m and p are both larger than n . We consider imposing a quadratic inequality constraint on the LS problem

$$\text{(LSQI)} \qquad \min_{\|Bx-d\|\leqslant\alpha} \|Ax - b\|$$

If $U^TAX = D_A = \text{diag}(\alpha_i)$ and $V^TBX = D_B = \text{diag}(\beta_i)$ $(\beta_1 \geqslant \quad \geqslant \beta_r > \beta_{r+1} = \quad = \beta_n = 0)$ is the BSVD of A and B then LSQI transforms to

$$(1) \qquad \min_{\|D_By-\hat{d}\|\leqslant\alpha} \|D_Ay - \hat{b}\| \qquad \text{where} \qquad \begin{aligned} \hat{b} &= U^Tb \\ \hat{d} &= V^Td \\ y &= X^{-1}x \end{aligned}$$

The diagonal problem (1) will assist us in the analysis and solution of (LSQI).

To begin with, (LSQI) has a solution if and only if $\min \|Bx - d\| = \sum_{i=r+1}^{p} \hat{d}_i^2 \leqslant \alpha^2$. If we have equality in this expression then the solution to (1) has the form $\hat{y} = \left(\frac{\hat{d}_1}{\beta_1}, \ldots, \frac{\hat{d}_r}{\beta_r}, z_{r+1}, \ldots, z_n\right)$. The unknown z_i's are easily resolved by minimizing $\|D_A\hat{y} - \hat{b}\|$. Of course, if rank(B) = r = n , there are no z_i's to resolve! In light of these remarks we hereafter assume

$$(2) \qquad \min \|Bx - d\|^2 = \sum_{i=r+1}^{p} \hat{d}_i^2 < \alpha^2$$

The solution to (LSQI) is not unique if and only if the null spaces of A and B have a nontrivial intersection. This is equivalent to having some $\alpha_k = 0$ for some $k > r$. In this case the unique solution of minimal norm to (1) can be obtained by setting $y_k = 0$ for all $k > r$ where $\alpha_k = 0$. The remaining y_i can be resolved by solving a deflated version of (1). The deflated diagonal matrices of this new problem have non-intersecting null spaces, hence, we hereafter assume without loss of generality that

$$(3) \qquad NS(A) \cap NS(B) = \{0\} \quad \equiv \quad \alpha_i \neq 0 \quad i=r+1,\ldots,n$$

It may be that A^+b, the solution of $\min\|Ax - b\|$, is also feasible. That is, we may have

$$\|BA^+b - d\|^2 = \sum_{\substack{i=1 \\ \alpha_i \neq 0}}^{n} \left[\beta_i \frac{\hat{b}_i}{\alpha_i} - \hat{d}_i\right]^2 \leqslant \alpha^2$$

In this case we just solve the unconstrained version of (LSQI). Hence we may assume

(4) $$\|BA^+b - d\|^2 = \sum_{\substack{i=1 \\ \alpha_i \neq 0}}^{n} \left[\beta_i \frac{\hat{b}_i}{\alpha_i} - \hat{d}_i\right]^2 > \alpha^2$$

This assumption means that the solution to (LSQI) occurs on the boundary of the feasible set. Hence, our remaining goal is to solve

$$\min_{\|Bx-d\|=\alpha} \|Ax - b\| \equiv \min_{\|D_By-\hat{d}\|=\alpha} \|D_Ay - \hat{b}\|$$

assuming that (2),(3), and (4) hold.

Following the method of Lagrange multipliers we equate the partials of $h(y) = \|D_Ay - \hat{b}\|^2 + \lambda[\|D_By - \hat{d}\|^2 - \alpha^2]$ to zero. This leads to

$$[D_A^TD_A + \lambda D_B^TD_B]y = D_A^Tb + \lambda D_B^Td$$

implying a solution $y^{(\lambda)}$ where

(5) $$y_i^{(\lambda)} = \frac{\alpha_i\hat{b}_i + \lambda\beta_i\hat{d}_i}{\alpha_i^2 + \lambda\,\beta_i^2} \qquad i = 1,\ldots,n$$

Plugging $y^{(\lambda)}$ into $\|D_By - \hat{d}\| = \alpha^2$ gives the important equation

(6) $$\alpha^2 = \sum_{i=1}^{r} \left[\alpha_i \frac{\beta_i\hat{b}_i - \alpha_i\hat{d}_i}{\alpha_i^2 + \lambda\beta_i^2}\right]^2 + \sum_{i=r+1}^{p} \hat{d}_i^2$$

Our aim is to find a λ so (6) is satisfied and so $\|D_Ay^{(\lambda)} - \hat{b}\|$ is minimum. Assumption (2) implies that there is a real solution to (6) and (4) implies it must be positive. This also guarantees the $y_i^{(\lambda)}$ in (5) are well defined. The monotonicity of (6) tells us that there is only one positive solution

λ^* and this is the one we want. It can be computed using any standard root finding technique. We summarize these ideas in the following algorith

Algorithm LSQI

(a) Compute the BSVD $U^TAX = \mathrm{diag}(\alpha_i)$, $V^TBX = \mathrm{diag}(\beta_i)$ where $\beta_1 \geq \cdots \geq \beta_r > \beta_{r+1} = \cdots = \beta_n = 0$. Save X, $\hat{b} = U^Tb$, and $\hat{d} = V^Td$.

(b) If $\sum_{i=r+1}^{p} \hat{d}_i^2 > \alpha^2$ then quit since there is no solution.

(c) If $\sum_{i=r+1}^{p} \hat{d}_i^2 = \alpha^2$ then define $\hat{y}$ by

$$\hat{y}_i = \begin{cases} \hat{d}_i/\beta_i & i = 1,\ldots,r \\ \hat{b}_i/\alpha_i & i = r+1,\ldots,n \quad \alpha_i \neq 0 \\ 0 & i = r+1,\ldots,n \quad \alpha_i = 0 \end{cases}$$

and go to (g).

(d) If $\alpha_i = 0$ for some $i > r$, reorder indices r+1 through n so that $\alpha_{r+1} \geq \quad \geq \alpha_k > \alpha_{k+1} = \quad = \alpha_n = 0$; set $\hat{y}_{k+1} = \ldots = \hat{y}_n = 0$; replace n by k; and continue.

(e) If $\sum_{i=1}^{n} \left(\beta_i \frac{\hat{b}_i}{\alpha_i} - \hat{d}_i \right)^2 \leq \alpha^2$ define $\hat{y}$ by

$$\hat{y}_i = \begin{cases} \hat{b}_i/\alpha_i & i = 1,\ldots,n \quad \alpha_i \neq 0 \\ 0 & i = 1,\ldots,n \quad \alpha_i = 0 \end{cases}$$

and go to (g).

(f) Find the unique positive root λ to (6) and define $\hat{y}$ by

$$\hat{y}_i = \frac{\alpha_i \hat{b}_i + \lambda \beta_i \hat{d}_i}{\alpha_i^2 + \lambda \beta_i^2} \qquad i = 1,\ldots,n$$

(g) $\hat{x} = X\hat{y}$

Quadratically constrained LS problems are mentioned in Golub[17] where particular attention is paid to the special case $\min_{\|x\| = \alpha} \|Ax - b\|$.

Related material may be found in [25,p.189].

Total Least Squares [17]

In the ordinary LS problem we sought a vector x satisfying $Ax=b+r$ such that $\|r\|$ was minimum. This approach is consistent in fitting problems where errors are present in the observation b but not in the data A. When this is not the case it has been suggested that the total least squares (TLS) minimization is appropriate. To describe this new problem assume we are given $A \in R^{m\times n}$ $(m > n)$, $b \in R^m$, and square positive diagonal matrices $S = \mathrm{diag}(s_1,\ldots,s_m)$ and $T = \mathrm{diag}(t_1,\ldots,t_{n+1})$. The TLS problem involves finding a vector x satisfying

$$(1) \qquad (A + E)x = b + r \qquad E \in R^{m\times n} , r \in R^m$$

such that

$$(2) \qquad \| S \ [E|r] \ T \|_F = \min$$

Our analysis of the TLS problem begins by fixing x and observing that there are many $[E|r]$ for which (1) holds. We'll use Lagrange multipliers to find the one with the smallest weighted F-norm. By defining $B = [A|b]$, $G = [E|r]$, $y = \begin{pmatrix} x \\ -1 \end{pmatrix}$ (1) and (2) transform to

$$By + Gy = 0 \qquad \| S \ G \ T \|_F = \min$$

By equating the appropriate partials of $h(G) = \frac{1}{2} \|SGT\|_F^2 + \lambda^T(By+Gy)$ to zero we find

$$\frac{\partial h}{\partial g_{ij}} = s_i^2 \, g_{ij} \, t_j^2 + \lambda_i y_j = 0$$

implying $S^2 G T^2 + \lambda y^T = 0$. By manipulating this equation and $By+Gy = 0$ we find

$$\lambda = \frac{S^2 By}{\|T^{-1}y\|^2} \qquad G = - \frac{Byy^T T^{-2}}{\|T^{-1}y\|^2}$$

Our goal is to pick y , and hence x, so that

$$\| SGT\|_F = \frac{\|SByy^T T^{-1}\|_F}{\|T^{-1}y\|^2} = \frac{\|SBy\|}{\|T^{-1}y\|} = \frac{\|SBT(T^{-1}y)\|}{\|T^{-1}y\|}$$

is as small as possible. If $\mu_{min}(C)$ denotes the smallest singular value of a matrix C, it is clear we want $T^{-1}y$ to be a multiple of $w \in R^{n+1}$ where $(SBT)w = \mu_{min}(SBT)w$. These ideas lead to the following algorithm:

Algorithm TLS

1. Form $C = S[A|b]T$ and compute its SVD: $U^TCV = \text{diag}(\gamma_1, \ldots, \gamma_{n+1})$
   $\gamma_1 \geqslant \cdots \geqslant \gamma_{k-1} > \gamma_k = \cdots = \gamma_{n+1} \geqslant 0$
2. Find $z \in \text{span}\langle v_k, \ldots, v_{n+1}\rangle$ so $z_{n+1} \neq 0$. (v_i = i-th column of V)
   If no such z exists, the problem has no solution.
3. $\hat{x}$ defined by $\begin{pmatrix} \hat{x} \\ -1 \end{pmatrix} = \dfrac{1}{t_{n+1}z_{n+1}} Tz$ solves the TLS problem.

Remarks.

TLS may have more than one solution if the smallest singular value of $S[A|b]T$ is repeated.

Failure can occur in step 2 for a number of reasons. For example, TLS has no solution if $\text{rank}(A) < n$ and $b \notin \text{range}(A)$. To see why set $E = \varepsilon \dfrac{bw^T}{\|b\|\|w\|}$ where $Aw = 0$ and $\varepsilon > 0$. One can show that $(A+E)\dfrac{\|b\|}{\varepsilon\|w\|}w = b$ and that $\|S[E|0]T\|_F \leqslant \varepsilon \|S\|\|T\|$. Since ε was arbitrary, but not zero, we see that there is no "smallest" $[E|r]$.

When A has full rank, it is rather hard to characterize failure in step 2 in terms of the various subspaces and vectors around. However, one ca show that the algorithm will fail if and only if

$$\mu_{min}(SA\hat{T}) = \mu_{min}(SBT)$$

where $\hat{T} = \text{diag}(t_1, \ldots, t_n)$. Since $SBT = [SA\hat{T}|t_{n+1}Sb]$ we can choose $t_1, \ldots, t_{n+1}$ so that under the assumption $\text{rank}(A) = n$

$$t_{n+1}\|Sb\| < \mu_{min}(SAT)$$

Since $\mu_{min}(SBT) < t_{n+1}\|Sb\|$, for this choice of T, the algorithm will succeed.

LECTURE 6. INEQUALITY CONSTRAINTS.

The Problem.

We now want to consider the problem

$$\text{(LSI)} \qquad \min_{Bx \geqslant d} \|Ax - b\|$$

where $A \in R^{m\times n}$, $b \in R^m$, $B \in R^{p\times n}$, $d \in R^p$. We assume that $m \geqslant n$. To say that $u \geqslant v$ where $u,v \in R^k$ means that $u_i \geqslant v_i$ for $i=1,\ldots,k$. Inequality constraints rule out the possibility of doing simplifying row operations on B because $Bx \geqslant d$ does not imply $QBx \geqslant Qd$

The Geometry.

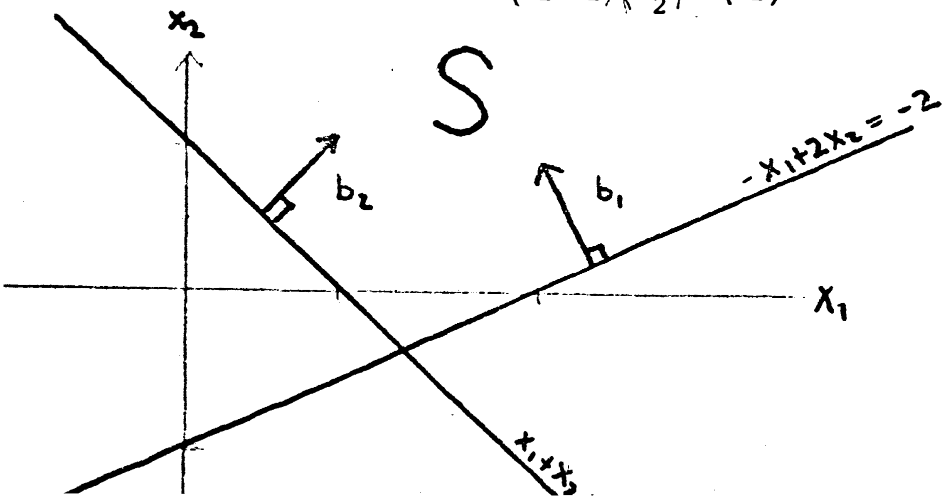
It is convenient to define

$$\begin{aligned} \phi(x) &= \tfrac{1}{2}\|Ax - b\|^2 \\ S &= \{x \mid Bx \geqslant d\} \\ g_x &= A^TAx - A^Tb \end{aligned}$$

Thus, $\min_{Bx \geqslant d} \|Ax - b\| \equiv \min_{x \in S} \phi(x)$ and so we are in the business of minimizing a convex function $\phi(x)$ $\left[(\phi(\lambda x+(1-\lambda)y) \leqslant \lambda\phi(x)+(1-\lambda)\phi(y) \quad \lambda\in[0,1] \right]$ over a convex set S $\left[x,y \in S \Rightarrow \lambda x+(1-\lambda)y \in S \quad \lambda\in[0,1] \right]$. Things could be a lot worse.

(a) What S looks like in detail.

Let b_i^T be the i-th row of B . The sides of S are hyperplanes with normals b_i pointing into S (also known as the feasible region). For example, if our constraint is $\begin{pmatrix} -1 & 2 \\ 1 & 1 \end{pmatrix}\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \geqslant \begin{pmatrix} -2 \\ 1 \end{pmatrix}$ then



(b) How many solutions?

- if $S = \emptyset$, none!
- if y and z both solve LSI $A(y-z) = 0$. (Proof: use convexity of ϕ, S
- if A has full column rank, at most one solution.
- if A lacks full column rank, we <u>may</u> still have only one solution:

$$A = \begin{pmatrix} 2 & 1 \\ 4 & 2 \end{pmatrix} , \ b = \begin{pmatrix} 3 \\ 6 \end{pmatrix} , \ B = \begin{pmatrix} 1 & -1 \\ 1 & 1 \end{pmatrix} , \ d = \begin{pmatrix} 0 \\ 2 \end{pmatrix} \qquad x = \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

<u>The Math</u>.

The basic game we will play is the following:

1. find $x \in S$ (quit if ca
2. find y so $x+y \in S$ and $\phi(x+y) < \phi(x)$ (quit if ca
3. $x := x+y$ go to 2 .

The first theorem tells us that y cannot be in any old direction.

<u>Theorem 1</u>.

If $g_x = A^TAx - A^Tb$ and $y^Tg_x < 0$, then there exists an $\varepsilon > 0$ so that $\phi(x+\varepsilon y) < \phi(x)$. If $y^Tg_x > 0$, then $\phi(x+\varepsilon y) > \phi(x)$ for all $\varepsilon > 0$.

<u>Proof</u>.

$$\phi(x+\varepsilon y) = \frac{1}{2}\|(Ax-b) + \varepsilon Ay\|^2 = \phi(x) + \varepsilon(\, y^Tg_x + \frac{\varepsilon}{2}\|Ay\|^2 \,)$$

The quantity $\varepsilon(y^Tg_x+\frac{\varepsilon}{2}\|Ay\|^2)$ can be made negative for small enough ε provided $y^Tg_x < 0$. If $y^Tg_x \geq 0$, the same quantity will always be non-negative for all $\varepsilon \geq 0$ ∎

The next theorem tells us when a vector x solves LSI .

<u>Theorem 2</u>.

Suppose $x \in S$. If for every y such that $x+y \in S$ we have $y^Tg_x \geq 0$, then x solves LSI.

<u>Proof</u>.

Suppose $\hat{x}$ solves LSI and set $y = \hat{x} - x$. Then

$\phi(\hat{x}) = \phi(x+y) = \phi(x) + y^Tg_x + \frac{1}{2}\|Ay\|^2 \geq \phi(x)$. But $\phi(\hat{x}) \leq \phi(x)$ so

$\phi(\hat{x}) = \phi(x)$. ∎

Theorem 2 suggests, but not in a very practical way, how to ascertain whether a vector $x \in S$ solves LSI. For certain problems, the next theorem gives a more practical characterization of the solution.

Theorem 3. (Kuhn-Tucker)

Suppose $x \in S$ and define $\mathcal{E}_x = \{i \mid b_i^T x = d_i\}$. ($\mathcal{E}_x$ is the set of indices where we have equality in $Bx \geq d$.) Then

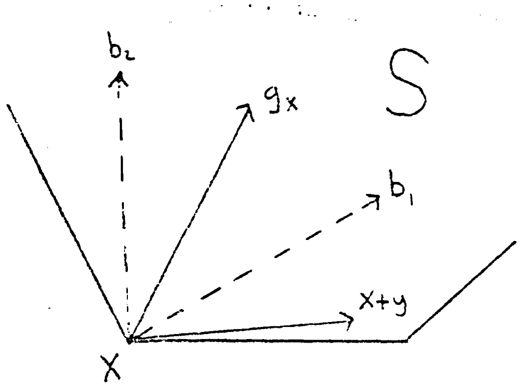
$$x \text{ solves LSI} \iff \begin{cases} \exists z \in R^p \text{ so } g_x = \sum_{i=1}^{p} z_i b_i \\ z_i = 0 \quad i \notin \mathcal{E}_x \ ; \ z_i \geq 0 \quad i \in \mathcal{E}_x \end{cases}$$

Proof.

($\Leftarrow$) Suppose $x+y \in S$. For each $i \in \mathcal{E}_x$ $b_i^T x = d_i$ and so $d_i \leq b_i^T(x+y)$ implies $b_i^T y \geq 0$. But from Theorem 2 this says x solves LSI because $y^T g_x = \sum_{i=1}^{p} z_i y^T b_i \geq 0$

($\Rightarrow$) For clarity assume $\mathcal{E}_x = \{1,\ldots,k\}$. Without loss of generality we may also assume that $\{b_1,\ldots,b_r\}$ $(r \leq k)$ is a basis for span$\langle b_1,\ldots,b_k \rangle$. Write $g_x = u + v$ where $u \perp v$ and $u \in \text{span}\langle b_1,\ldots,b_k\rangle$. It is easy to verify that for small $\varepsilon > 0$ $(x - \varepsilon v) \in S$. Since x solves LSI we have from Theorem 2 that $-\varepsilon v^T g_x \geq 0$. But this implies $v = 0$ because $-\varepsilon v^T g_x = -\varepsilon \|v\|^2$ There are thus scalars $z_1,\ldots,z_r$ so $g_x = \sum_{i=1}^{r} z_i b_i$. We must show that each z_i is nonnegative. For every i $(1 \leq i \leq r)$ there exists a y so $y^T b_i > 0$ and $y^T b_j = 0$ $(j=1,\ldots,k,\ j \neq i)$. This follows from the independence of $b_1,\ldots,b_r$ and the fact that these vectors span $\{b_1,\ldots,b_k\}$. It is easy to show that for small enough $\varepsilon > 0$, $x + \varepsilon y \in S$. Since x solves LSI we must have $\varepsilon y^T g_x \geq 0$. But $\varepsilon y^T g_x = \varepsilon\, z_i\, y^T b_i$ implying $z_i \geq 0$. ∎

R^2 Example



x is a solution because we cannot find a y more than 90° away from g_x so $x+y \in S$. Notice g_x is a positive linear combination of b_1 and b_2 .

Problems with Equality and Inequality Constraints

Before we consider algorithms for LSI problems we shall dispense with the following more general problem:

$$\text{(LSEI)} \qquad \min_{\substack{Bx \geq d \\ Cx=f}} \|Ax - b\| \qquad \begin{array}{lll} A \in R^{m\times n}, & b \in R^m & (m \geq n) \\ B \in R^{p\times n}, & d \in R^p & \\ C \in R^{q\times n}, & f \in R^q & (q < n) \end{array}$$

We give an algorithm similar to that used in the reduction of (LSE) to (LS) which leaves us with an LSI problem of smaller dimension:

Algorithm LSEI

1. Let $Q \in R^{n\times n}$ be orthogonal so $CQ = [L \mid 0]$ $L = \triangle \in R^{q\times q}$

   Set $AQ = [\underbrace{A_1}_{q} \mid \underbrace{A_2}_{n-q}]$, $BQ = [\underbrace{B_1}_{q} \mid \underbrace{B_2}_{n-q}]$, and $Q^T x = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \begin{matrix} \}q \\ \}n-q \end{matrix}$

2. Determine y_1 from $Ly_1 = f$ and y_2 from
$$\min_{B_2 y_2 \geq d - B_1 L^{-1} f} \|A_2 y_2 - (b - A_1 L^{-1} f)\|$$

3. $\hat{x} = Q \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}$ solves (LSEI).

We have assumed that C has full row rank. (If not one formulates a successful version of this algorithm by doing an SVD of C in step 1.) The reason this reduction of (LSEI) to (LSI) works is the following:

$$\text{(LSEI)} \equiv \min_{\substack{BQQ^T x \geq d \\ CQQ^T x = f}} \|AQQ^T x - b\| \equiv \min_{\substack{B_1 y_1 + B_2 y_2 \geq d \\ Ly_1 = f}} \|A_1 y_1 + A_2 y_2 - b\|$$

Nonnegative Least Squares

An important special case of (LSI) is the nonnegative least square problem:

$$\text{(NNLS)} \qquad \min_{x \geq 0} \|Ax - b\| \qquad A \in R^{m\times n}, \quad b \in R^m$$

Applied to this problem, Theorem 3 says

$$x \text{ solves (NNLS)} \iff \begin{cases} g_i \geq 0 & \text{when } x_i = 0 \\ g_i = 0 & \text{when } x_i > 0 \\ g = A^T Ax - A^T b & \end{cases}$$

LECTURE 7. ALGORITHMS FOR INEQUALITY CONSTRAINED LEAST SQUARE PROBLEMS

Algorithm NNLS [25,p.161]

0. Set $\mathcal{E} = \{1,2,\ldots,n\}$ $\quad P = \emptyset \quad$ $x = 0$
1. Compute $g = A^T(Ax - b)$.
2. If $g_i \geqslant 0$ for every $i \in \mathcal{E}$, x solves (NNLS) so quit.
3. Find index r in $\mathcal{E}$ such that $g_r = \min\limits_{i \in \mathcal{E}} g_i$. Move r from $\mathcal{E}$ to P .
4. Let a_i denote the i-th column of A. Form $A_P = [a_{i_1}|a_{i_2}|\cdots|a_{i_k}]$ where $P = \{i_1,\ldots,i_k\}$ and let $w^T = (w_{i_1},\ldots,w_{i_k})$ be such that $\|A_P w - b\|$ is minimum.
5. Define $z \in R^n$ by $z_q = w_{i_j}$ if $q = i_j$, $z_q = 0$ if $q \in \mathcal{E}$. If $z_q > 0$ for all q in P , set $x = z$ and go to 1.
6. Set $x = x + \alpha(z-x)$ where $\alpha = \min\{ \frac{x_k}{x_k - z_k} \mid k \in P , \ z_k \leqslant 0 \}$
7. Move all the indices j in P to $\mathcal{E}$ if $x_j = 0$. Go to 4.

This algorithm is not only important because it solves NNLS, but because it has a central role to play in other algorithms which follow.

Comments on NNLS

Step 0. Throughout the algorithm, $\mathcal{E}$ (P) specifies which components of the current x are zero (positive).

Step 1. Here the gradient of the current x is computed.

Step 2. This is a quitting test based on the Kuhn-Tucker criteria. The reason we only have to test the g_i where $i \in \mathcal{E}$, is that $g_i = 0$ for all $i \in P$. This follows by looking at the normal equations associated with the LS problem in step 4. From these equations we see that $g_i = a_i^T(Ax - b) = 0$ for all i in P .

Step 3. The gradient g makes the biggest angle with the inward normal of constraining hyperplane r. This means that of all the x_i which are currently zero, allowing x_r to be positive will probably diminish $\|Ax - b\|$ the most.

Step 4. This is the most time consuming step in the algorithm as it involves solving an unconstrained LS problem. However, each LS problem is just like its predecessor except one column may be added (if we enter from Step 3) or several columns may be deleted (if we enter from Step 7). If we solve these problems using the QR method, then the updating schemes of Lecture 3 are applicable. We mention that the various orthogonal matrices which arise are not accumulated but are just applied to b and all of the columns of A (whether or not they are associated with P).

What goes on in the remaining steps is imbedded in the analysis of why NNLS works. To begin with, suppose we enter Step 4 from Step 3. Let P denote the old set of positive component indices and define P_1 by $P_1 = P \cup \{r\}$. From Steps 2 and 3 , $g_r = a_r^T(Ax-b) < 0$ implying $a_r \notin \operatorname{span}_{i \in P} \langle a_i \rangle$ since each vector in this span is orthogonal to $Ax - b$. If e_r is the r-th column of I, then $e_r^T g = g_r < 0$ and so $\| A(x+\epsilon e_r) - b \|$ can be made strictly smaller than $\|Ax - b\|$. This implies that $a_r^T b \neq 0$ and thus, $\min \|A_{P_1} u - b\| < \min \|A_P v - b\|$. But the vector z of Step 5 satisfies $\|Az - b\| = \min \|A_{P_1} u - b\|$ while $\|Ax - b\| = \min \|A_P v - b\|$ and so

$$(1) \qquad \| Az - b\| < \|Ax - b\|$$

Hence, if we exit to Step 2 from Step 5, our new (feasible) x yields a strictly smaller sum of squares than the old x. Suppose, however, we go on to Step 6. The choice of α is always in (0,1) and is such that $x + \alpha(z-x)$ is nonnegative. If $\tilde{x} = x + \alpha(z - x)$ then from inequality (1) we have

$$(2) \qquad \| A\tilde{x} - b\| = \| A[x+\alpha(z-x)]-b\| < (1-\alpha)\| Ax-b\| + \alpha\|Az-b\| < \|Ax - b\|$$

If we repeat the loop 45674567... then it can be shown by an inductive argument that all subsequent x's give a sum of squares no bigger than $\|A\tilde{x} - b\|^2$. Conclusion: every time we reach Step 2 , $\|Ax - b\|$ is strictly smaller. It is clear that the loop 45674567... cannot go on forever because each time through this cycle the set P is strictly diminished in Step 7. P can never be "emptied" because this would imply $x = 0$ and the current x gives a sum of squares strictly smaller than $\|A0 - b\|^2$. The algorithm is finite because every time we reach step 2, P is different. This follows because at Step 2, $\|Ax - b\| = \min_{\substack{y_i = 0 \\ i \in \mathcal{E}}} \|Ay - b\|$ and as we have shown, the sum of squares at this step is strictly decreasing. In practice one finds the critical Step 4 executed about .5n times. This implies that the overall algorithm is $O(mn^2)$.

Least Distance Programming

The least distance programming problem is defined as follows:

$$\text{(LDP)} \qquad \min_{Bx \geq d} \|x\| \qquad x \in R^n\ , \ B \in R^{p\times n}\ , \quad d \in R^p\ .$$

The algorithm for solving this involves setting up a related nonnegative least square problem which is subsequently solved by NNLS. The output is then manipulated to give the solution to the original LDP problem if it exists.

Algorithm LDP [25,p.165]

1. Use NNLS to find the solution $\hat{u}$ to $\min_{u \geq 0} \left\| \begin{pmatrix} B^T \\ d^T \end{pmatrix} u - \begin{pmatrix} 0 \\ 1 \end{pmatrix} \right\|$
2. Define $r = \begin{pmatrix} B^T \\ d^T \end{pmatrix} u - \begin{pmatrix} 0 \\ 1 \end{pmatrix} \in R^{n+1}$.
3. If $r_{n+1} = 0$, then no x satisfies $Bx \geq d$ and LDP has no solution.
   If $r_{n+1} \neq 0$, then $\hat{x}$ solves LDP where $\hat{x}_i = -r_i/r_{n+1}$ (i=1,...,n).

The proof that $\hat{x}$ solves LDP involves hooking up the Kuhn-Tucker criteria for the NNLS problem in Step 1 with those of the original LDP problem. The details may be found in [25].

The General LSI Problem [25]

We now look at the general inequality constrained LS problem. Assume that A has rank n and that $A = Q\begin{pmatrix} R \\ 0 \end{pmatrix}$ is its QR factorization. If

$$Q^T b = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \begin{matrix} \}n \\ \}m-n \end{matrix} \qquad \text{and} \qquad z = Rx - b_1$$

then

$$\min_{Bx \geq d} \|Ax - b\| \equiv \min_{Bx \geq d} \|Q^T Ax - Q^T b\| \equiv \min_{Bx \geq d} \|Rx - b_1\| \qquad \min_{BR^{-1}(z+b_1) \geq d} \|z\|$$

This shows how the LSI problem can be converted into LDP problem.

Algorithm LSI (Full rank case)

1. Compute $Q^T A = \begin{pmatrix} R \\ 0 \end{pmatrix} = \begin{pmatrix} \nabla \\ 0 \end{pmatrix}$, $Q^T b = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \begin{matrix} \} n \\ \} m-n \end{matrix}$

2. Use Algorithm LDP to find the solution $\hat{z}$ to $\min_{\tilde{B}z \geq \tilde{d}} \|z\|$ where $\tilde{B} = BR^{-1}$ and $\tilde{d} = d - BR^{-1}b_1$.

   If no solution to this problem exists, the original LSI problem has no solution.

3. $\hat{x} = R^{-1}(\hat{z} + b_1)$ solves LSI.

Complications arise when rank(A) < n . Suppose

$$U^T AV = \begin{array}{c} \begin{pmatrix} D & 0 \\ 0 & 0 \end{pmatrix} \begin{matrix} \}r \\ \}m-r \end{matrix} \\ \begin{matrix} r & n-r \end{matrix} \end{array} \qquad D = \text{diag}(\mu_i) \ , \ \mu_i > 0$$

is the SVD of A. By defining $y = V^T x = \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} \begin{matrix} \} r \\ \} n-r \end{matrix}$, $BV = [B_1 | B_2]$,

$U^T b = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \begin{matrix} \} r \\ \} m-r \end{matrix}$ and $z = Dy_1 - b_1$, we have

$$\min_{Bx \geq d} \|Ax - b\| \equiv \min_{B_1 y_1 + B_2 y_2 \geq d} \|Dy_1 - b_1\| \equiv \min_{B_1 D^{-1}(z+b_1) + B_2 y_2 \geq d} \|z\|$$

The problem on the right is a "modified" LDP problem and has been studied and solved by Cline [10]. One might think of setting $y_2 = 0$ in the above for then we would have an ordinary LDP problem to solve. However, the act of setting $y_2 = 0$ may make the LDP insoluble even though the original LSI is not: $\min_{y \geq 1} \left\| \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} - \begin{pmatrix} 2 \\ 3 \end{pmatrix} \right\|$.

LECTURE 8. SOME SOURCES OF LEAST SQUARES PROBLEMS.

Linear Regression.

Suppose one wishes to predict the height b of an adult plant based upon the concentrations of various nutrients in the soil. A possible linear model for height might be

$$b(a_1,\ldots,a_n) \;=\; x_0 + a_1x_1 + a_2x_2 + \ldots + a_nx_n + e$$

where a_j is the concentration of nutrient j and e is a normally distributed random variable with mean 0 and variance σ^2. This model depends upon the parameters $x_0,\ldots,x_n$ and σ^2 . For a given nutrient vector $(a_1,\ldots,a_n)$ the expected height is thus $x_o + a_1x_1+\ldots+a_nx_n$.

One is now faced with the problem of determining the parameters which define the model. This is done by performing m experiments the i-th of which involves setting the concentration of nutrient j at a_{ij} and observing the resulting height b_i . Now

$$b_i = x_0 + a_{i1}x_1 + \ldots + a_{in}x_n + e$$

is itself a normally distributed random variable with mean $x_0 + \sum_{j=1}^{n} a_{ij}x_j$ and variance σ^2. It has the following probability density function:

$$f_i(z) \;=\; (2\pi\sigma^2)^{-\frac{1}{2}} \exp\left[-\frac{(z - x_0 - a_{i1}x_1 - \ldots - a_{in}x_n)^2}{2\sigma^2}\right]$$

Now the liklihood of observing the heights $b_1,\ldots,b_m$ is defined by

$$L(x_0,x_1,\ldots,x_n,\sigma) \equiv \prod_{i=1}^{m} f_i(b_i) = (2\pi\sigma^2)^{-\frac{m}{2}} \exp\left[-\frac{\|Ax-b\|^2}{2\sigma^2}\right]$$

where $b = (b_1,\ldots,b_m)^T$ and $A = \begin{pmatrix} 1 & a_{11} & \cdot & a_{1n} \\ \vdots & \vdots & & \vdots \\ 1 & a_{m1} & \cdot\cdot & a_{mn} \end{pmatrix}$. Now the "best" model

is the one which most likely predicts b given A. Mathwise, we want to choose $x_0,\ldots,x_n,$ so that L is maximized. Taking the necessary partial derivatives of L shows us that we want $\|Ax - b\| = \min$ and $\sigma^2 = \frac{\|Ax - b\|^2}{m}$

Curve Fitting

In this application we are given coordinates $(x_1,y_1),\ldots,(x_m,y_m)$ and basis functions $p_0(x),\ldots,p_n(x)$ and are asked to determine constants $a_0,\ldots,a_n$ so that if

$$p(x) = a_0p_0(x) + a_1p_1(x) + \ldots + a_np_n(x)$$

then

$$\phi(a_0,\ldots,a_n) = \sum_{i=1}^{m} [y_i - p(x_i)]^2$$

is minimum. We can phrase this in matrix language:

$$\text{minimize } \phi(a_0,\ldots,a_n) \quad \left\| \begin{bmatrix} p_0(x_1) & \cdots & p_n(x_1) \\ p_o(x_2) & \cdots & p_n(x_2) \\ \vdots & & \vdots \\ p_0(x_m) & \cdots & p_n(x_m) \end{bmatrix} \begin{pmatrix} a_o \\ a_1 \\ \\ a_n \end{pmatrix} - \begin{pmatrix} y_1 \\ y_2 \\ \\ y_m \end{pmatrix} \right\|$$

$$\equiv \| Pa - y \|^2$$

Suppose in addition we require that the slope of our approximation $p(x)$ is nonnegative at a given set of points $\{z_1,\ldots,z_q\}$. This translates into the constraint

$$Ba \geqslant 0 \equiv \begin{pmatrix} p_0'(z_1) & & p_n'(z_1) \\ \\ p_0'(z_q) & & p_n'(z_q) \end{pmatrix} \begin{pmatrix} a_0 \\ \\ a_n \end{pmatrix} \geqslant \begin{pmatrix} 0 \\ \\ 0 \end{pmatrix}$$

Thus, we are led to the problem $\min\limits_{Ba \geqslant 0} \| Pa - y \|$.

The basis functions above may be polynomials, splines, exponentials or trigonometric functions. The term "basis" is meant to imply that these functions are independent in some sense. When we assume that the basis functions are given, then we arrive at a linear least square problem. The basis functions, however, may have unknown parameters in them. (e.g. splines with variable knots or exponentials with unknown exponents.) In this case a nonlinear least squares problem results.

As an example of this type of nonlinear curve fitting problem, suppose we wish to fit $g(t) = a_1 \exp[\alpha_1 t] + a_2 \exp[\alpha_2 (t - \alpha_3)^2]$ to the data $\{(t_i, b_i)\}_{i=1}^{m}$. This leads to the minimization of

$$\left\| \begin{pmatrix} \exp[\alpha_1 t_1] & \exp[\alpha_2 (t_1 - \alpha_3)^2] \\ \vdots & \vdots \\ \exp[\alpha_1 t_m] & \exp[\alpha_2 (t_m - \alpha_3)^3] \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} - \begin{pmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{pmatrix} \right\|$$

Integral Equations of the First Kind

The continuous version of this problem involves finding a function **f(x)** defined on $[a,b]$ so

(1) $$\int_a^b K(y,x) f(x)\,dx = g(y)$$

holds for all $y \in [c,d]$. K and g are given and do not have to be very remarkable for (1) to lack a solution. However, we shall assume (1) is solvable and proceed immediately to discretize. Replacing the integral with a quadrature involving weights $\omega_1, \ldots, \omega_n$ and abscissae $x_1, \ldots, x_n$ we obtain

(2) $$\sum_{j=1}^{n} \omega_j K(y, x_j) f(x_j) = g(y)$$

If we now try to satisfy (2) at $y_1, \ldots, y_m$ $(m \geq n)$ we are led to an overdetermined system of linear equations in unknowns $f(x_j)$:

(3) $$\sum_{j=1}^{n} \omega_j K(y_i x_j) f(x_j) = g(y_i) \qquad i=1,\ldots,m$$

Solving this in the least squares sense gives the following problem:

(4) $$\min_f \|Kf - g\|$$

where $K \in R^{m\times n}$, $k_{ij} = K(y_i, x_j)\omega_j$, $f^T = (f(x_1), \ldots, f(x_n))$, and $g^T = (g(y_1), \ldots, g(y_m))$.

The ill-conditioned nature of (1) spills over into the least square problem (4) making it advisable to "calm" the solution in some sensible fashion. A way of tempering (1) is to require

$$(5) \qquad \int_a^b [f''(x)]^2\,dx \leq \alpha^2$$

By approximating this integral with the same quadrature rule as above and estimating $f''(x_i)$ with $[f(x_{i-1}), f(x_i), f(x_{i+1})]$ (second order divided difference) we find that (5) becomes

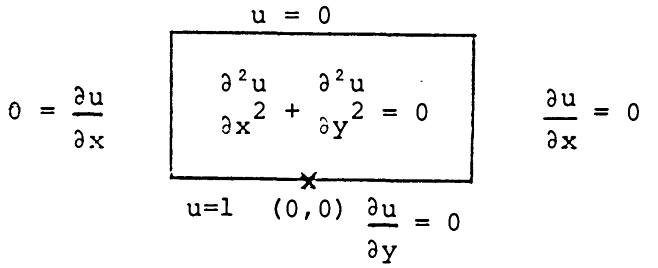
$$(6) \qquad \|Bf\|^2 \leq \alpha^2$$

where B is an nxn tridiagonal matrix. ($f(x_{-1})$ and $f(x_{n+1})$ require speci handling.) Coupling the constraint (6) with (4) renders the following LSQI problem

$$\min_{\|Bf\| \leq \alpha} \|Kf - g\|$$

<u>Collocation</u>

It so happens that $u_n(r,\theta) = 1 + \sum_{j=1}^{n} a_j\, r^{(j-.5)} \cos[(j-\tfrac{1}{2})\theta]$ satisfi everything in



$x = r\cos(\theta)$
$y = r\sin(\theta)$

except the top and left and right boundary conditions. In collocation we choose $a_1, \ldots a_n$ by placing m points P_i around the top, right, and left boundary of the above rectangle and solving the following overdetermined system in the least squares sense:

$$\frac{\partial u_n(P_i)}{\partial x} = 0 \quad (P_i \text{ on left or right boundary})$$

$$u_n(P_i) = 0 \quad (P_i \text{ on the top boundary})$$

LECTURE 9. BASIC ALGORITHMS FOR NONLINEAR LEAST SQUARE PROBLEMS

Introduction

We now turn our attention to the nonlinear least square problem:

$$\text{(NLS)} \qquad \min_{x \in R^n} \phi(x) \qquad \text{where} \quad \phi(x) = \frac{1}{2} \sum_{i=1}^{m} f_i(x)^2 \qquad f_i : R^n \to R$$

When $f_i(x) = b_i - \sum_{j=1}^{n} a_{ij} x_j$ we get our old linear least square problem : $\phi(x) = \frac{1}{2} \|Ax - b\|^2$.

The Taylor Expansion of $\phi(x)$

When $\theta(x)$ is a sufficiently differentiable function from R^n to R it is possible to do a Taylor expansion of θ about a point x . Defining the gradient $\nabla\theta(x) \in R^n$ by

$$\nabla\theta(x)^T = [\frac{\partial\theta(x)}{\partial x_1}, \ldots, \frac{\partial\theta(x)}{\partial x_n}]$$

and the Hessian $\nabla^2\theta(x) \in R^{n \times n}$ by

$$[\nabla^2\theta(x)]_{ij} = \frac{\partial^2\theta(x)}{\partial x_i \partial x_j}$$

we have

$$\text{(1)} \qquad \theta(x+y) = \theta(x) + \nabla\theta(x)^T y + \frac{1}{2} y^T [\nabla^2\theta(x)] y + O(\|y\|^3)$$

For our sum of squares functional $\phi(x)$ it is easy to verify that

$$\text{(2)} \qquad \nabla\phi(x) = J(x)^T f(x) \qquad \nabla^2\phi(x) = J(x)^T J(x) + \sum_{i=1}^{m} f_i(x) \nabla^2 f_i(x)$$

where

$$J(x) = \begin{bmatrix} \frac{\partial f_1(x)}{\partial x_1} & \cdots & \frac{\partial f_1(x)}{\partial x_n} \\ \vdots & & \vdots \\ \frac{\partial f_m(x)}{\partial x_1} & \cdots & \frac{\partial f_m(x)}{\partial x_n} \end{bmatrix} \in R^{m \times n} \qquad f(x) = \begin{bmatrix} f_1(x) \\ \vdots \\ f_m(x) \end{bmatrix} \in R^m$$

Hence, the Taylor expansion for $\phi(x)$ can be written

$$(3) \qquad \phi(x+y) = \frac{1}{2} f^T f + f^T J y + \frac{1}{2} y^T J^T J y + \frac{1}{2} \sum_{i=1}^{m} f_i(x) y^T \nabla^2 f_i(x) y + O$$

where $f \equiv f(x)$ and $J \equiv J(x)$

From Taylor Expansions to Algorithms

Two basic strategies for minimizing $\phi(x)$ can be deduced by studying the expansion (3).

(a) Newton Methods

Looking at $\phi(x+y) \cong \phi(x) + \nabla\phi(x)^T y + \frac{1}{2} y^T \nabla^2\phi(x) y$ we ask, what y makes the right hand side as small as possible? By taking appropriate partial derivatives we find that the y we are looking for satisfies

$$[\nabla^2\phi(x)]y = -\nabla\phi(x) \quad ,$$

a symmetric system of linear equations. Hopefully, the "new" solution x+y renders a smaller sum of squares than x. The process can obviously be repeated and the result is the Newton iteration:

$$(N) \qquad x^{k+1} = x^k + y^k \quad \text{where } y^k \text{ solves } [\nabla^2\phi(x^k)]y = -\nabla\phi(x^k)$$

The gradient and Hessian are of course specified by (2).

There is another way this iteration can be deduced. Substituting ϕ for θ and $-\varepsilon\nabla\phi(x)$ for y in (1) we find

$$\phi[x - \varepsilon\nabla\phi(x)] = \phi(x) - \varepsilon \|\nabla\phi(x)\|^2 + O(\varepsilon^2)$$

If $\nabla\phi(x) \neq 0$, then by choosing $\varepsilon > 0$ small enough $x - \varepsilon\nabla\phi(x)$ will give a smaller sum of squares than x . Hence we are in the market for zeros of the gradient $\nabla\phi(x)$. The Taylor expansion for the gradient looks like $\nabla\phi(x+y) \simeq \nabla\phi(x) + \nabla^2\phi(x)y$. By equating the right hand side of this to zero we are once again led to (N).

(b) Gauss-Newton Methods

The expansion (3) can be written

$$(4) \qquad \phi(x+y) \approx \frac{1}{2}\|Jy + f\|^2 + \frac{1}{2} y^T [\sum_{i=1}^{m} f_i(x)\nabla^2 f_i(x)] y$$

In Gauss-Newton it is argued that a good correction y is obtained by minimizing $\|Jy + f\|$, a linear least square problem. The iteration of this idea gives

$$\text{(GN)} \qquad x^{k+1} = x^k + y^k \quad \text{where } y^k \text{ minimizes } \|J(x^k)y + f(x^k)\|$$

Nonlinear Lingo

In this section we summarize the pros and cons of the (N) and (GN) iterations. This will involve a concurrent review of some basic "nonlinear" terminology. The material for this section has been gathered from the survey articles by Powell[31], Dennis[11], and Dennis and Moré[12].

We say that a vector z is a descent direction for ϕ at x if $\nabla\phi(x)^T z < 0$. If this is the case then there exists an $\varepsilon>0$ such that $\phi(x + \varepsilon z) < \phi(x)$. This follows by looking at the Taylor expansion for ϕ. One cannot guarantee that the Newton correction is a descent direction unless the Hessian is positive definite for then we have $\nabla\phi(x)^T y = -\nabla\phi(x)^T[\nabla^2\phi(x)]^{-1}\nabla\phi(x) < 0$. Positive definiteness of the Hessian is usually present in the vicinity of a local minimum (see below). On the other hand, it is easy to show that the Gauss-Newton correction $-J(x^k)^+f(x^k)$ is always a descent direction so long as neither $f(x^k)$ nor $\nabla\phi(x^k)$ are zero.

For both the (N) and (GN) iterations, the zeros of $\nabla\phi(x)$ are fixed points. These zeros are the stationary values of ϕ and may be maxima, minima, or "saddle points". Which they are depends upon the Hessian. In the case of interest, x^* is a local minimum if $\nabla\phi(x^*) = 0$ and $\nabla^2\phi(x^*) = H$ is positive semidefinite ($z^T Hz \geq 0$ all z). This means that no other vector in an open neighborhood of x^* can give a smaller sum of squares. This does not imply x^* is a global minimum, i.e. the "best" x in all of R^n from the point of view of minimizing ϕ. Without additional assumptions, such as convexity, or additional knowledge about the (NLS) problem in hand, it is impossible to conclude that a local minimum x^* is in fact a global minimum.

All algorithms for the (NLS) problem are iterative and we thus need a way of quantifying their speed of convergence. If $x^k \to x^*$ we say that convergence is

<u>linear</u> if for all $k \geq k_0$ $\|x_{k+1} - x^*\| \leq c\|x_k - x^*\|$ $c \in (0,1)$

<u>superlinear</u> if $\|x_{k+1} - x^*\| = \alpha_k \|x_k - x^*\|$ and $\lim \alpha_k = 0$

<u>quadratic</u> if for all $k \geq k_0$, $\|x_{k+1} - x^*\| \leq C\|x_k - x^*\|^2$

Suppose the (N) and (GN) iterations converge to x^* and that the following Lipschitz conditions hold for all x and y in a neighborhood of x^* :

$$\|\nabla^2 f_i(x) - \nabla^2 f_i(y)\| \leq K_1\|x - y\| \qquad \|J(x) - J(y)\| \leq K_2\|x - y\|$$

One can show that the Newton iteration converges quadratically provided $\nabla^2(x^*)$ is nonsingular. On the otherhand, Gauss-Newton is quadratic if $\phi(x^*) = 0$, but linear if $\phi(x^*) \neq 0$.

The conditional quadratic convergence of the (GN) iteration stems from the neglecting of the "second half" of the Hessian

$$K(x) = \sum_{i=1}^{m} f_i(x)\nabla^2 f_i(x)$$

If $\phi(x^*) = 0$ then $K(x^*) = 0$ and the (N) and (GN) iterations will be indistinguishable in the vicinity of x^*. However, if $\phi(x^*)$ is not zero, we can hardly expect second order convergence from a method which ignores second order terms like $K(x)$. When the ϕ- surface is highly nonlinear in the neighborhood of x^*, the component Hessians $\nabla^2 f_i$ will be large intensifying the harmful effect that a nonzero $\phi(x^*)$ has. The (N) iteration is free from these limitations because $K(x)$ was incorporated in the formulatio of the Newton algorithm.

The distinction between the zero and nonzero residual problems figures in the <u>local convergence</u> behavior of the (GN) iteration. We ask, if x^k is sufficiently close to x^*, will Gauss-Newton converge? It turns out that convergence cannot be guaranteed unless $K_2\phi(x^*)^{1/2} < \mu_{min}[J(x^*)]^2$ showing that a poorly conditioned $J(x^*)$ coupled with a nonzero residual may prove disasterous for (GN). (N) is locally convergent independent of $\phi($

On the practical side, (N) involves many more partial derivatives to evaluate than (GN) in a given step. One also has to worry about singular Hessians whereas the LS problem in (GN) is always well-defined.

LECTURE 10. VARIATIONS OF THE GAUSS-NEWTON ITERATION

Limiting the Gauss-Newton Correction

Just because the vector y^k in the (GN) iteration is a descent direction doesn't mean $\phi(x^k + y^k) < \phi(x^k)$. In fact, if $J(x^k)$ is near rank degeneracy, y^k will most likely be large and poorly determined. As a result, $\phi(x^k + y^k)$ may be much larger than $\phi(x^k)$. For this reason, the basic Gauss-Newton step should be modified. One way of doing this is to add only a portion of y^k to x^k .

$$\text{(MGN)} \qquad x^{k+1} = x^k + \alpha_k y^k \qquad \text{where:} \quad \begin{array}{l} y^k \text{ minimizes } \|J(x^k)y + f(x^k)\| \\ \alpha_k > 0 \text{ chosen so } \phi(x^k+\alpha_k y^k) < \phi(x^k) \end{array}$$

We know that the scalar α_k can be found because y^k is a descent direction. The one dimensional problem of finding an α so $\phi(x^k+\alpha y^k) < \phi(x^k)$ can be approached in several different ways. A discussion of these ways and appropriate references can be found in Powell[31].

The Levenberg-Marquardt Algorithm [26]

There is another shortcoming of the (GN) iteration not readily apparant from the defining equations. We know that y^k is a descent direction because it is less than 90° from the negative gradient $-J(x^k)^T f(x^k)$. When this angle is close to 90° we correctly suspect that ϕ diminishes very slowly along y^k . The point we wish to make is that it is very easy to have this state of affairs when J is poorly conditioned. To see why, let θ be the angle between $-J^T f \equiv -J(x^k)^T f(x^k)$ and $y \equiv y^k = -J^+ f$. We then have

$$\cos(\theta) = \frac{f^T J J^+ f}{\|J^T f\| \|J^+ f\|}$$

Let $J = UDV^T$, $D = \text{diag}(\mu_i)$ be the SVD of J. If u_i denotes the i-th column of U and neither μ_n nor $u_n^T f$ are zero, then

$$\cos(\theta) = \frac{\sum_{i=1}^{n} (u_i^T f)^2}{\|J^T f\| \left\{ \sum_{i=1}^{n} \left(\frac{u_i^T f}{\mu_i}\right)^2 \right\}^{1/2}} \leq \frac{\mu_n}{|u_n^T f|} \frac{\|f\|}{\|J^T f\|}$$

From this inequality it is easy to see that θ has a good chance of being close to 90° when μ_n is small and J poorly conditioned. This leads to slow convergence because of the mildness of the descents.

The Levenberg-Marquardt algorithm attempts to overcome this limitation of the (GN) correction vector.

$$\text{(LM)} \qquad x^{k+1} = x^k + y^k \qquad \text{where} \quad \begin{array}{l} y^k \text{ minimizes } \|J(x^k)y + f(x^k)\|^2 + \lambda_k^2 \|y\|^2 \\ \lambda_k \geq 0 \end{array}$$

We have two things to explain: the effects of the λ's have and how they are chosen.

<u>The Effect of the Levenberg-Marquardt Parameter</u>

Denote the solution to

$$\min \|Jy + f\|^2 + \lambda^2 \|y\|^2 \equiv \min \left\| \begin{pmatrix} J \\ \lambda I \end{pmatrix} y + \begin{pmatrix} f \\ 0 \end{pmatrix} \right\|^2$$

by $y(\lambda)$. Of course, $y(0)$ is the (GN) correction. If $\lambda > 0$ then it is not hard to see that $\begin{pmatrix} J \\ \lambda I \end{pmatrix}$ has full rank. In fact, if $\mu_1 \geq \cdots \geq \mu_n$ are the singular values of J, then

$$\text{cond}\left[\begin{pmatrix} J \\ \lambda I \end{pmatrix}\right] = \sqrt{\frac{\mu_1^2 + \lambda^2}{\mu_n^2 + \lambda^2}}$$

It is clear that the condition of the augmented matrix $\begin{pmatrix} J \\ \lambda I \end{pmatrix}$ decreases wit increasing λ. This prompts many people to determine $y(\lambda)$ via the normal equations:

$$[J^TJ + \lambda^2 I]y(\lambda) = J^Tf \qquad\qquad y(\lambda) = [J^TJ + \lambda^2 I]^{-1}J^Tf$$

From this formula for $y(\lambda)$ it can be deduced that

(a) $y(\lambda)$ is a descent direction

(b) $\lim_{\lambda\to\infty} \|y(\lambda)\| = 0$

(c) $\lim_{\lambda\to\infty} \dfrac{y(\lambda)}{\|y(\lambda)\|} = -\dfrac{J^Tf}{\|J^Tf\|}$

A Lagrange multiplier argument also shows that if $\|y(\lambda)\| = \beta$, then $y(\lambda)$ solves $\min_{\|y\|=\beta} \|Jy + f\|$. This shows $y(\lambda)$ is the "best" y of length β.

Choosing the λ_k

If the swift minimization of ϕ is our goal, then we must be sensible about the selection of each λ_k . In this section we sketch and comment upon some methods which have been suggested.

Marquardt[26] sets $\lambda_k = \nu^p \lambda_{k-1}$. Here, ν is a fixed constant greater than 1 and p is chosen to be the smallest integer greater or equal to -1 for which $\phi[x^k + y(\nu^p \lambda_{k-1})] < \phi[x^k]$. Properties (a), (b), and (c) above guarantee that such a p exists. ($\exists\ \lambda \geq 0$ so $\phi[x+y(\lambda)] < \phi(x)$) This process is expensive because each $y(\lambda)$ costs about mn^2 operations. However, Marquardt argues that his λ_k allow for nearly optimal interpolation between steepest descent ($\lambda = \infty$) and Gauss-Newton ($\lambda = 0$).

Brown and Dennis[7] determine the λ_k on the basis of the current residual $\|f(x^k)\|$:

$$\lambda_k^2 = c\,\|f(x^k)\| \qquad \text{where} \quad c = \begin{cases} 10 & \text{if} \quad 10 \leq \|f(x^k)\| \\ 1 & \text{if} \quad 1 \leq \|f(x^k)\| < 10 \\ .01 & \text{if} \quad \|f(x^k)\| < 1 \end{cases}$$

Behind these criteria is the authors' result that the (LM) iteration has the same local convergence properties as Gauss Newton provided $\lambda_k^2 = O(\|f(x^k)\|)$.

Fletcher[33] adjusts the λ_k from iteration to iteration on the basis of the following ratio:

$$r = \frac{\phi(x^k) - \phi(x^k + y^k)}{\phi(x^k) - \frac{1}{2}\|J(x^k)y^k + f(x^k)\|^2} = \frac{\text{actual reduction in } \phi}{\text{predicted reduction in } \phi}$$

When r is greater than .75 (say), the linear model is good and reduction in λ_k is called for. When r is less than .25 (say), the linear model is poor and λ_k is increased. Otherwise, λ_k is "OK". When we say that the linear model is "good" ("bad") we mean that the higher order terms in the Taylor expansion which Gauss-Newton neglects are inconsequential (important).

Finally, Bartels, Golub, and Saunders [2] suggest that y^k solve $\min_{\|y\| \leq \alpha_k} \|J(x^k)y + f(x^k)\|$. We won't detail how the α_k are selected but simply call attention to fact (d) above which connects this approach to the standard LM procedures already outlined.

Implementing the Levenberg-Marquardt Iteration

Thus far we haven't paid too much attention to the volume of computation which attends an (LM) step. The work involves the setting up and solution of a linear LS problem. On the setting up score, it doesn't take much expertise to realize that thecoding up and evaluation of the mn partials which make up J is an error prone and time consuming proposition. We are thus led to the possibility of using divided differences instead of analytic derivatives. Brown and Dennis[7] have discussed the following derivative-free Levenberg-Marquardt algorithm:

(DFLM) $x^{k+1} = x^k + y^k$ where y^k minimizes $\|Gy + f\|^2 + \lambda_k^2 \|y\|^2$ $\lambda_k > 0$

Here $g_{ij} = [f_i(x^k + h_j e_j) - f_i(x^k)]/h_j$, e_j = j-th column of I_n and h_j is the smaller of $\|f(x^k)\|$ and δ_j where δ_j is a small constant based upon machine precision and $|x_j^k|$.

The authors' found that the performance of (LM) and (DFLM) was similar i: that the algorithms seemed to succeed and fail together.

After J is evaluated or approximated, we still must solve a linear least square problem involving the matrix $\begin{pmatrix} J \\ \lambda I \end{pmatrix}$. Because of the special structure of this matrix, economies can be made if its QR factorization is sought [20]. Essentially, one computes the QR factorization of J and then "wipes out" the λI by a judicious choice of Householder transformations.

Prelude to Update Methods

Using differences instead of derivatives makes coding up and using Levenberg-Marquardt a lot easier. Nevertheless, the volume of computatio per step remains cubic.Furthermore, the (LM) iteration sputters on large residual problems. This sets the stage for sundry Newton methods which involve a minimum of setting up and solution costs per step.

LECTURE 11. UPDATING METHODS

Updating Formulae and Background [13]

Theorem 1.

If $B \in R^{m\times n}$, $y \in R^m$, and $s \in R^n$ (nonzero), then $\hat{E} = \frac{(y - Bs)s^T}{\|s\|^2}$ solves $\min_{(B+E)s=y} \|E\|_F$.

Proof.

Using the method of Lagrange multipliers we form $h(E) = \frac{1}{2}\|E\|_F^2 + \lambda^T[(B+E)s-y]$. Upon differentiation we find $\frac{\partial h}{\partial e_{ij}} = e_{ij} + \lambda_i s_j = 0 \Rightarrow$

$E = -\lambda s^T \Rightarrow y = (B+E)s = Bs - \|s\|^2 \lambda \Rightarrow \hat{\lambda} = (Bs - y)/\|s\|^2 \Rightarrow \hat{E} = -\hat{\lambda}s^T = (y - Bs)s^T/\|s\|^2$. ▮

This theorem says that $\bar{B} = B + \hat{E}$ is the closest matrix to B which sends s to y . Also notice that $\bar{B}z = Bz$ if $z^Ts = 0$. The following theorem can also be established using Lagrange multipliers but we delete the proof.

Theorem 2.

If $B \in R^{n\times n}$, $B^T = B$, $y \in R^n$, and $s \in R^n$ (nonzero), then

$$\hat{E} = \frac{(y - Bs)^Ts + s(y - Bs)^T}{\|s\|^2} - \frac{(y - Bs)^Ts}{\|s\|^4} ss^T$$ solves the problem

$$\min_{\substack{E^T=E \\ (B+E)s=y}} \|E\|_F \quad .$$

In this case, $\bar{B} = B + \hat{E}$ is the closest symmetric matrix to B which sends s to y. Again notice that $\bar{B}z = Bz$ if $z^Ts = 0$.

Now let $g: R^n \to R^m$ be twice differentiable so it has the following Taylor expansion about $\bar{x}$:

$$g(x) = g(\bar{x}) + T(\bar{x})(x - \bar{x}) + O(\|x - \bar{x}\|^2)$$

Here $T(\bar{x}) \in R^{m\times n}$ is the Jacobian of g at $\bar{x}$ ($[T(\bar{x})]_{ij} = \frac{\partial g_i(\bar{x})}{\partial x_j}$).

Suppose we have an approximation B to T(x) and seek an approximation $\bar{B}$ to $T(\bar{x})$ subject to the following constraints:

(1) $g(x) = g(\bar{x}) + \bar{B}(x - \bar{x})$ $\qquad \|\bar{B} - B\|_F = \min$

An application of Theorem 1 tells us that

(2) $$\bar{B} = B + \frac{(y - Bs)s^T}{\|s\|^2} \quad \text{where} \quad \begin{aligned} s &= \bar{x} - x \\ y &= g(\bar{x}) - g(x) \end{aligned}$$

This is Broyden's rank one update formula.

Now suppose $m = n$ and that the Jacobian T is always symmetric. Assume we have a symmetric approximate B to $T(x)$ and seek an approximate $\bar{B}$ to $T(\bar{x})$ subject to the following constraints:

(3) $g(x) = g(\bar{x}) + \bar{B}(x - \bar{x})$, $\|\bar{B} - B\|_F = \min$, $\bar{B}^T = \bar{B}$

In this case Theorem 2 informs us that

(4) $$\bar{B} = B + \frac{(y-Bs)s^T + s(y-Bs)^T}{\|s\|^2} - \frac{(y-Bs)^T s}{\|s\|^4} ss^T \qquad \begin{aligned} s &= \bar{x} - x \\ y &= g(\bar{x}) - g(x \end{aligned}$$

This is the Powell symmetric Broyden update.

The symmetric formulae of this section are important because when $g(x) = \nabla\phi(x)$, $T(x) = \nabla^2\phi(x)$ is symmetric. Equations (2) and (4) are examples of "quasi-Newton" updates.

<u>Application: Broyden's Method</u> [39]

Suppose we seek a zero x^* of $g: R^n \to R^n$. One approach to finding x^* is the Newton iteration : $x^{k+1} = x^k - T(x^k)^{-1}g(x^k)$. (T = Jacobian of g). But if we have an approximation x^0 to x^* , and an approximation B_0 to $T(x^0)$, then the following "quasi-Newton" iteration suggests itself:

(B) $x^{k+1} = x^k + y^k$ where y^k solves $B_k y = -g(x^k)$ and

$$B_k = B_{k-1} + \frac{(y - B_{k-1}s)s^T}{\|s\|^2} \quad , \; s = x^k - x^{k-1}, \; y = g(x^k) - g(x^{k-1})$$

This is Broyden's method. It is a practical improvement over the classical Newton iteration because we have no matrix of functions to evaluate each step. Furthermore, exploitation of $(B+uv^T)^{-1} = B^{-1} - \dfrac{(B^{-1}u)(v^T B^{-1})}{1 - v^T B^{-1} u}$

enables us to solve the linear system in (B) in $O(n^2)$ instead of $O(n^3)$. Local and superlinear convergence can also be established with suitable assumptions [13].

Updating in Gauss-Newton Algorithms

We consider now how our least squares Jacobian $J(x)$ can be updated. Suppose B is an approximation to $J(x)$ and that we want an approximation $\bar{B}$ to $J(\bar{x})$. By applying (1) with $g(x) \equiv f(x)$ we find

$$\bar{B} = B + \frac{(y - Bs)s^T}{\|s\|^2} \qquad \begin{array}{l} s = \bar{x} - x \\ y = f(\bar{x}) - f(x) \end{array}$$

This leads to the "quasi-Gauss-Newton" iteration:

$$\text{(QGN)} \qquad x^{k+1} = x^k + y^k \quad \text{where } y^k \text{ minimizes } \|B_k y + f(x^k)\| \quad \text{and}$$

$$B_k = B_{k-1} + \frac{(y - B_{k-1}s)s^T}{\|s\|^2} \qquad \begin{array}{l} s = x^k - x^{k-1} \\ y = f(x^k) - f(x^{k-1}) \end{array}$$

If one has a factorization $QB_{k-1}Z = \begin{pmatrix} R \\ 0 \end{pmatrix}$ (Q, Z orthogonal, $R = \nabla$) then this factorization can be updated, and the LS problem in (QGN) solved in $O(mn)$ operations.

We pose (QGN) not as a serious algorithm but merely as an illustration of how quasi-newton updates can reduce the amount of work per step by an order of magnitude. Powell's VA05A [41] is an example of a "serious" Gauss-Newton type algorithm which incorporates single rank updates of J. In this algorithm, y^k is obtained by doing a judicious linear combination of $-B_k f(x^k)$ (an approximate steepest descent vector) and $-[B_k^T B_k + \lambda_k^2 I]^{-1} B_k^T f(x^k)$ (a Levenberg-Marquardt correction).

Updating in Least Square Newton Algorithms

Applied to our NLS problem, the classical Newton iteration computes $\bar{x} = x + s$ where s solves

$$(5) \qquad [J(x)^T J(x) + K(x)]y = -J(x)^T f(x)$$

and

$$K(x) = \sum_{i=1}^{m} f_i(x)\nabla^2 f_i(x)$$

We now consider determining y from $[J(x)^T J(x) + B]y = - J(x)^T f(x)$ where B is an approximation to K(x) and is corrected each iteration by using the symmetric update formula (4).

(a) Method of Brown and Dennis [6]

Here $B = \sum_{i=1}^{m} f_i(x)B_i$ where each B_i is an approximation to $\nabla^2 f_i(x)$ and is updated to $\bar{B}_i$ by setting $B \equiv B_i$, $y = \nabla f_i(\bar{x}) - \nabla f(x)$, $s = \bar{x} - x$ in (4). Standard assumptions allow one to prove local and superlinear convergence. The required storage , $O(mn^2)$, can be prohibitive. However, there are curve fitting problems where the component Hessians are sparse in which case the $O(mn^2)$ figure is not applicable.

(b) Method of Broyden and Dennis [11]

The approximation $[J(\bar{x})^T J(\bar{x}) + K(\bar{x})](\bar{x} - x) \cong J(\bar{x})^T f(\bar{x}) - J(x)^T f(x)$ leads us to ask $[J(\bar{x})^T J(\bar{x}) + \bar{B}](\bar{x} - x) = J(\bar{x})^T f(\bar{x}) - J(x)^T f(x)$ of our update $\bar{B}$. This suggests obtaining $\bar{B}$ through formula (4) with $s = \bar{x} - x$ and $y = J(\bar{x})^T f(\bar{x}) - J(x)^T f(x) - J(\bar{x})^T J(\bar{x})s$. The success of this algorthm has not been documented.

(c) Method of Dennis [12]

Here the approximation $K(\bar{x})(\bar{x} - x) = \sum_{i=1}^{m} f_i(\bar{x})\nabla^2 f_i(\bar{x})(\bar{x} - x) \cong \sum_{i=1}^{m} f_i(\bar{x})[\nabla f_i(\bar{x}) - \nabla f_i(x)] = [J(\bar{x}) - J(x)]^T f(\bar{x})$ is considered. By requiring $\bar{B}$ to satisfy $\bar{B}(\bar{x} - x) = [J(\bar{x}) - J(x)]^T f(\bar{x})$ we see that B can be computed via (4) with $s = \bar{x} - x$ and $y = [J(\bar{x}) - J(x)]^T f(\bar{x})$. Again, the performance of this algorithm has not been recorded but some interesting comments are made by Nazareth [42].

Conclusions

We have sketched some of the ideas which arise in connection with the application of quasi-Newton techniques to the NLS problem. We have been vague not only because of limited space but because this is presently an area of active research making it impossible to draw hard conclusi

LECTURE 12. INTRODUCTION TO SEPARABLE NONLINEAR LEAST SQUARES

The Problem

We shall now consider nonlinear least square problems in which some of the parameters are involved only linearly. The example of exponential curve fitting in Lecture 8 is a problem of this type. Denote the linear parameters by $a^T = (a_1,\ldots,a_n)$ and their nonlinear counterparts by $\alpha^T = (\alpha_1,\ldots,\alpha_k)$. We are thus trying to minimize a sum of squares of the form

$$\phi(x) = \phi(\alpha,a) = \frac{1}{2}\sum_{i=1}^{m} f_i(\alpha,a)^2 \qquad f_i : R^{n+k} \to R$$

We shall assume that $m \geqslant n+k$. Because the a_i's are involved linearly, each f_i has the form

$$f_i(\alpha,a) = \sum_{j=1}^{n} a_j f_{ij}(\alpha) - b_i \qquad f_{ij} : R^k \to R$$

We are thus required to solve

$$\text{(SNLS)} \qquad \min_{\alpha,a} \phi(\alpha,a) = \min_{\alpha,a} \frac{1}{2}\|b - F(\alpha)a\|^2$$

where $[F(\alpha)]_{ij} = f_{ij}(\alpha)$ and $b^T = (b_1,\ldots,b_m)$.

Treating All Variables as Nonlinear

Let us look at the form the Gauss-Newton iteration takes when applied to $\phi(\alpha,a)$. If we identify $x^T = (x_1,\ldots,x_{n+k})$ with $(\alpha,a) = (\alpha_1,\ldots,\alpha_k,a_1,\ldots,a_n)$ then our old notation gives

$$f(\alpha,a) = F(\alpha)a - b$$

and

$$J(\alpha,a) = [\ H(\alpha,a) \mid F(\alpha)\ ]$$

where $H(\alpha,a) = [\ \frac{\partial F(\alpha)}{\partial \alpha_k} a \mid \cdots\cdots \mid \frac{\partial F(\alpha)}{\partial \alpha_k} a\ ] \in R^{m\times k}$

Now, if $y \in R^k$ and $w \in R^n$, then

$$\left\| J(\alpha,a)\binom{y}{w} + f(\alpha,a)\right\| = \| H(\alpha,a)y + F(\alpha)(w + a) - b \|$$

Thus, a (GN) update of (α^k, a^k) looks like

$$(1)\quad \begin{aligned} \alpha^{k+1} &= \alpha^k + y^k \\ a^{k+1} &= a^k + w^k \end{aligned} \quad \text{where } \begin{pmatrix} y^k \\ w^k \end{pmatrix} \text{ minimizes } \| H(\alpha^k, a^k)y + F(\alpha^k)(w+a^k) - b \|$$

We've applied the Gauss-Newton iteration to the (SNLS) problem because it is easiest to describe, but of course any NLS algorithm could be used to minimize $\phi(\alpha, a)$. However, practical experience has shown that it is often worth exploiting the linear/nonlinear structure of ϕ [20],[35].

Barham and Drane's Method [35]

For each fixed α, $a(\alpha) = F(\alpha)^+ b$ is the solution of the linear problem

$$\min_a \| F(\alpha)a - b \|$$

Thus, each vector of nonlinear parameters α has a "best" set of linear "brothers" in $a(\alpha)$. We can incorporate this idea in the (GN) iteration (1) by setting $a^{k+1} = F(\alpha^k)^+ b$. This yields the method of Barham and Drane[35]:

$$(\mathrm{BD})\quad a^{k+1} \text{ minimizes } \| F(\alpha^k)a - b \| \; ; \text{ and } \alpha^{k+1} = \alpha^k + y^k$$
$$\text{where } y^k \text{ minimizes } \| H(\alpha^k, a^k)y - [I - F(\alpha^k)F(\alpha^k)^+]b \|$$

If we use the QR factorization to solve the linear least square problems in (1) and (BD) then the latter requires about $nk[2m - n - k]$ fewer operations. (Setting up costs are about the same for each algorithm.) Of course, this is an empty advantage if more iterations are required when (BD) is used. This appears not to be the case [35].

Separation and Variable Projection Algorithms

If only candidate minimizers for $\phi(\alpha, a)$ of the form $(\alpha, a(\alpha))$ are considered, we are led to the functional

$$\tilde{\phi}(\alpha) \equiv \phi(\alpha, a(\alpha)) = \frac{1}{2} \| b - F(\alpha)F(\alpha)^+ b \|^2 = \frac{1}{2} \| [I - F(\alpha)F(\alpha)^+]b \|^2$$

This leads us to consider the following "variable projection" algorithm:

(VP)

(a) Find $\hat{\alpha}$ to minimize $\tilde{\phi}(\alpha) = \frac{1}{2} \|[I - F(\alpha)F(\alpha)^+]b\|^2$

(b) Find $\hat{a}$ to minimize $\phi(\hat{\alpha},a) = \frac{1}{2} \|F(\hat{\alpha})a - b\|^2$

Under appropriate conditions, $(\hat{\alpha},\hat{a})$ will minimize $\phi(\alpha,a)$.(Of course, we should be talking about local minimums.) These conditions involve certain "constant rank" assumptions on the matrix function $F(\alpha)$ which are needed to guarantee the differentiability of $F(\alpha)^+$ and thus, $\tilde{\phi}(\alpha)$. A detailed analysis may be found in Golub and Pereyra[20].

(VP) shows what the term "separable" in "separable nonlinear least squares" means. The linear parameters are separated from the problem leaving only α_i's to find in step (a). After this is accomplished, the desired linear parameters are obtained by solving a linear least square problem in (b).

There are a number of variable projection algorithms we wish to consider, each differing in how the step (a) above is carried out. Since the properties of the "variable projector" $I - F(\alpha)F(\alpha)^+$ will be critically exploited by these algorithms, we shall spend some time time studying it.

<u>The Matrix $I - FF^+$</u>

Let $F \in R^{mxn}$ be of rank r and suppose

(2) $$F = Q \begin{pmatrix} R & 0 \\ 0 & 0 \end{pmatrix} Z^T$$

where $Q \in R^{mxm}$ (orthogonal) , $Z \in R^{nxn}$ (orthogonal) , and $R \in R^{rxr}$ (invertible). Factorization (2) may represent any of our standard orthogonal decomposions. Recalling the definition of the pseudoinverse from Lecture 1, we have

(3) $$F^+ = Z \begin{pmatrix} R^{-1} & 0 \\ 0 & 0 \end{pmatrix} Q^T \qquad \in R^{nxm}$$

and so

$$(4) \qquad F F^{+} = Q \begin{pmatrix} I_r & 0 \\ 0 & 0 \end{pmatrix} Q^T$$

$$(5) \qquad I - F F^{+} = Q \begin{pmatrix} 0 & 0 \\ 0 & I_{m-r} \end{pmatrix} Q^T$$

If $G = F F^{+}$, equations (2),(3), and (4) can be used to verify that

$$(6) \qquad G F = F$$

$$(7) \qquad G^T = G$$

$$(8) \qquad G^2 = G$$

One can also show that

$$x \in \text{ran}(F) \rightarrow (I - FF^{+})x = 0$$

$$x \in \text{ran}(F)^{\perp} \rightarrow (I - FF^{+})x = x$$

These properties tell us that $I - FF^{+}$ is the orthogonal projector onto the orthogonal complement of the range of F.

Differentiation of Matrix Functions

In the next lecture we shall have occasion to take various partial derivatives of $G(\alpha) = F(\alpha)F(\alpha)^{+}$. To prepare for this we will now conduct a little review of matrix calculus. Suppose $A(\alpha)$ is an mxn matrix whose entries are real functions of $\alpha \varepsilon R^k$: $[A(\alpha)]_{ij} = a_{ij}(\alpha)$. The derivative of $A(\alpha)$ with respect to α_p is another mxn matrix $\frac{\partial A(\alpha)}{\partial \alpha_p}$ whose (i,j) entry is $\frac{\partial a_{ij}(\alpha)}{\partial \alpha_p}$. The usual rules of calculus hold. For example:

$$\frac{\partial [A(\alpha)B(\alpha)]}{\partial \alpha_p} = A(\alpha)\frac{\partial B(\alpha)}{\partial \alpha_p} + \frac{\partial A(\alpha)}{\partial \alpha_p} B(\alpha)$$

As an application of these definitions and results, let us see how partial derivatives of $g(\alpha) = x^T[A(\alpha)]^{-1}x$ might be evaluated. (Assume $A(\alpha)$ is positive definite for all $\alpha \varepsilon R^k$ and that $x \varepsilon R^n$ is fixed.) Since $I = A(\alpha)A(\alpha)^{-1}$ we have $0 = \frac{\partial A(\alpha)}{\partial \alpha_p} A(\alpha)^{-1} + A(\alpha)\frac{\partial [A(\alpha)^{-1}]}{\partial \alpha_p}$ so $\frac{\partial g}{\partial \alpha_p} = -x^T A^{-1}[\frac{\partial A(\alpha)}{\partial \alpha_p}] A^{-1}x$.
The right hand side of this last expression is readily computed.

LECTURE 13. VARIABLE PROJECTION ALGORITHMS

Derivative-Free Methods

In this lecture we examine methods for minimizing $\tilde{\phi}(\alpha) = \frac{1}{2}\|[I - F(\alpha)F(\alpha)^{+}]b\|^2$ recalling that once this is accomplished, the complete solution to the (SNLS) problem is easily obtained. Because differentiation of $\tilde{\phi}(\alpha)$ appears so formidable (α is buried inside a pseudo-inverse) we might be tempted to use a derivative free minimizer. These routines, a discussion of which may be found in Brent[37], require that the user supply a subroutine capable of evaluating $\tilde{\phi}(\alpha)$. Fortunately, this can be done although at a cost of $O(mn^2)$. Suppose for clarity that $F(\alpha)$ has rank n. If $F(\alpha) = Q\begin{pmatrix} R \\ 0 \end{pmatrix}$ is the QR factorization of $F(\alpha)$, then

$$I - F(\alpha)F(\alpha)^{+} = Q\begin{pmatrix} 0 & 0 \\ 0 & I_{m-n} \end{pmatrix} Q^T$$

If $Q^T b = \begin{pmatrix} b_1 \\ b_2 \end{pmatrix} \begin{matrix} \}n \\ \}m-n \end{matrix}$ then $\tilde{\phi}(\alpha) = \frac{1}{2}\|b_2\|^2$. In implementing QR to evaluate $\tilde{\phi}(\alpha)$, we see that the Q matrix need never be stored but only applied to the vector b.

The Jacobian of $\tilde{\phi}(\alpha)$

To apply Gauss-Newton or Levenberg-Marquardt to $\tilde{\phi}(\alpha)$, we must have access to the Jacobian $J(\alpha)$ of $\tilde{\phi}(\alpha)$. By defining $r(\alpha)$

$$r(\alpha) = [I - F(\alpha)F(\alpha)^{+}]b$$

we find that

$$(1) \qquad J(\alpha) = \left[\frac{\partial r(\alpha)}{\partial \alpha_1} \Big| \ldots \Big| \frac{\partial r(\alpha)}{\partial \alpha_k} \right] \in R^{m \times k}$$

Right away we see that $J(\alpha)$ cannot be set up unless we know how to differentiate $F(\alpha)F(\alpha)^{+}$.

Differentiation of $F(\alpha)F(\alpha)^+$ [20]

Define

$$G(\alpha) = F(\alpha)\, F(\alpha)^+$$

We will show that

$$(2) \qquad \frac{\partial G}{\partial \alpha_j} = \left[(I - G) \frac{\partial F}{\partial \alpha_j} F^+ \right] + \left[(I - G) \frac{\partial F}{\partial \alpha_j} F^+ \right]^T$$

To prove this we shall exploit the following three results established in the previous lecture:

(a) $GF = F$

(b) $G^T = G$

(c) $G^2 = G$

By differentiating (a) we find

$$(3) \qquad \frac{\partial G}{\partial \alpha_j} F + G \frac{\partial F}{\partial \alpha_j} = \frac{\partial F}{\partial \alpha_j} \quad \text{or} \quad \frac{\partial G}{\partial \alpha_j} F = (I - G) \frac{\partial F}{\partial \alpha_j}$$

By using (c) and then (b) we obtain

$$\frac{\partial G}{\partial \alpha_j} = \frac{\partial [G^2]}{\partial \alpha_j} = \frac{\partial G}{\partial \alpha_j} G + G \frac{\partial G}{\partial \alpha_j} = \frac{\partial G}{\partial \alpha_j} G + \left[\frac{\partial G}{\partial \alpha_j} G \right]^T$$

Equation (2) follows because from (3)

$$\frac{\partial G}{\partial \alpha_j} G = \frac{\partial G}{\partial \alpha_j} FF^+ = (I - G) \frac{\partial F}{\partial \alpha_j} F^+$$

The Efficient Computation of J

From the definition of $r(\alpha)$ and equation (1) we find that the j-th column of $J(\alpha)$ is given by

$$(4) \qquad \frac{\partial r}{\partial \alpha_j} = - \left\{ (I - G) \frac{\partial F}{\partial \alpha_j} F^+ + \left[(I - G) \frac{\partial F}{\partial \alpha_j} F^+ \right]^T \right\} b$$

Everything in this expression is computable. The question, however, is how the computations can be done efficiently. Let us assume $F(\alpha)$ has full rank and that its QR factorization is given by

$$F = Q \begin{pmatrix} R \\ 0 \end{pmatrix} \qquad Q^T Q = I_m \quad , \quad R = \nabla \in R^{n \times n}$$

Substitution of $F^+ = [\, R^{-1} \mid 0]$ and $(I - G) = Q \begin{pmatrix} 0 & 0 \\ 0 & I_{m-n} \end{pmatrix} Q^T$ into (4) gives

$$\frac{\partial r}{\partial \alpha_j} = - Q \, (C_j + C_j^T) \, Q^T b$$

where

$$C_j = \begin{pmatrix} 0 & 0 \\ 0 & I_{m-n} \end{pmatrix} Q^T \left(\frac{\partial F}{\partial \alpha_j} \right) [\, R^{-1} \mid 0 \,]$$

Let us put these results together to see what the Gauss-Newton least square problem looks like:

$$\begin{aligned} \| J(\alpha) y + r(\alpha) \| &= \left\| \left[\frac{\partial r}{\partial \alpha_1} \mid \cdots \mid \frac{\partial r}{\partial \alpha_k} \right] y + (I - G) b \right\| \\ &= \left\| [(C_1 + C_1^T) Q^T b \mid \cdots \mid (C_k + C_k^T) Q^T b] \, y - \begin{pmatrix} 0 & 0 \\ 0 & I_{m-n} \end{pmatrix} Q^T b \right\| \end{aligned}$$

It is clear from this result that the setting up of this least square problem involved heavy use of the QR factorization of $F(\alpha)$. Still, the volume of work seems massive, especially when one considers the C_j. But it is important to remember (and exploit) the fact that the derivative matrices $\frac{\partial F}{\partial \alpha_j}$ are often sparse. To see this, consider the problem of fitting the function $u(t) = \sum_{j=1}^{n} a_j \exp[\alpha_j t]$ to the data $\{(t_i, b_i)\}_{i=1}^{m}$.

In this case

$$F(\alpha) = \begin{bmatrix} \exp[\alpha_1 t_1] & \cdots & \exp[\alpha_n t_1] \\ \vdots & & \vdots \\ \exp[\alpha_1 t_m] & \cdots & \exp[\alpha_n t_m] \end{bmatrix} \qquad \frac{\partial F}{\partial \alpha_j} = \begin{bmatrix} 0 & \cdots & t_1 \exp[\alpha_j t_1] & \cdots & 0 \\ \vdots & & \vdots & & \vdots \\ 0 & \cdots & t_m \exp[\alpha_j t_m] & \cdots & 0 \end{bmatrix}$$

Without going into details, it is obvious in this case that the setting up of $J(\alpha)$ will involve considerably less work than the general equations imply. These ideas are incorporated in the Golub-Pereyra algorithm "VARPRO" which is described in [20]. In their implementation Levenberg-Marquardt is used but our remarks above are still germane.

gorithm "VARPRO" which is described in [20]. In their implementation Levenberg-Marquardt is used but our remarks above are still relevant.

Computing $\nabla\tilde{\phi}(\alpha)$

One may choose to use a general variable metric routine [39] to minimize $\tilde{\phi}$. These algorithms require that the user be able to compute $\nabla\tilde{\phi}(\alpha)$. Since $\tilde{\phi}(\alpha) = \frac{1}{2} r(\alpha)^T r(\alpha)$ we find

$$[\nabla\phi(\alpha)]_j = \frac{\partial\phi(\alpha)}{\partial\alpha_j} = r(\alpha)^T \frac{\partial r}{\partial\alpha_j}$$

Hence, the vectors $\frac{\partial r}{\partial\alpha_j}$ must be computed and so the same techniques and comments we mentioned in connection with the evaluation of $J(\alpha)$ are applicable.

Handling Separable Equality Constraints

We briefly discuss how to minimze $\phi(\alpha,a)$ such that $H(\alpha)a = g(\alpha)$ where $H(\alpha)$ is a pxn $(p < n)$ matrix function. Dropping the "alphas" and assuming H has full row rank, let $HQ^T = (R \,|\, O)$ $(Q^T Q = I_n$, $R = \nabla \in R^{pxp}$

Thus,

$$Ha = g \qquad HQ^T Qa = g \qquad (R \mid O)\begin{pmatrix} c \\ z \end{pmatrix} = g \qquad a = Q^T \begin{pmatrix} R^{-1}g \\ \cdots \\ z \end{pmatrix} \qquad z = \text{anyt} \atop \text{in F}$$

Using the facts $H = (R \mid O)Q^T$, $H^+ = Q \begin{pmatrix} R^{-1} \\ O \end{pmatrix}$, it is possible to show

$$a = H^+ g + (I_n - H^+ H)w \qquad w = \text{anything in } R^n$$

Substituting this into our original functional gives

$$\begin{aligned} \| F(\alpha)a - b \| &= \| F(\alpha)[H^+(\alpha)g(\alpha) + (I_n - H^+(\alpha)H(\alpha))w] - b \| \\ &= \| [F(\alpha)(I_n - H^+(\alpha)H(\alpha))]w - [b - F(\alpha)H^+(\alpha)g(\alpha)] \| \\ &\equiv \| \tilde{F}(\alpha)w - \tilde{h}(\alpha) \| \end{aligned}$$

Using the kinds of techniques as above, we can take appropriatee derivative and use (say) Gauss-Newton to find $\hat{\alpha}$. Using $\hat{\alpha}$, we can minimize $\|\tilde{F}(\hat{\alpha})w -$ to obtain $\hat{w}$. Finally, $\hat{a} = H^+(\hat{\alpha})g(\hat{\alpha}) + (I - H^+(\hat{\alpha})H(\hat{\alpha}))\hat{w}$

LECTURE 14. CONSTRAINED AND SPARSE NONLINEAR LEAST SQUARES

(NLS) with Linear Equality Constraints [2]

In this and the next section we shall contemplate the minimization of $\phi(x) = \frac{1}{2}\sum_{i=1}^{m} f_i(x)^2$ when x satisfies some prescribed constraint. Our first case involves linear equality constraints:

(NLSE) $$\min_{Bx=d} \phi(x) \qquad B \in R^{p\times n} \quad (p \leqslant n)\ ,\ d \in R^p$$

Levenberg-Marquardt can be used to solve this problem in the following way:

(a) Find x^0 so $Bx^0 = d$. (Quit if this is impossible.)

(b) For $k = 0,1,\ldots$

$$x^{k+1} = x^k + y^k \quad \text{where } y^k \text{ solves } \min_{By=0} \left\| \begin{pmatrix} J(x^k) \\ \lambda_k I \end{pmatrix} y + \begin{pmatrix} f(x^k) \\ 0 \end{pmatrix} \right\|$$

Notice that $Bx^k = d$ for all k. The LSE problem in (b) can be solved using the elimination of variables method of Lecture 4. Observe that the first step of this algorithm, computing $BQ = [\ L \mid 0]$, need only be done once.

(NLS) with Linear Inequality Constraints

Constraints imposed on the solution of the NLS problem have a way of filtering down to the linear least square problem solved in each (LM) iteration. This was clear in the case of NLSE. Another example involves inequality constraints.

(NLSI) $$\min_{Bx \geqslant d} \phi(x) \qquad B \in R^{p\times n}\ ,\ d \in R^p$$

When (LM) is applied to this problem we have the following algorithm:

(a) Find x^0 so $Bx^0 > d$

(b) For $k = 0,1,\ldots$

$$x^{k+1} = x^k + y^k \quad \text{where } y^k \text{ solves } \min_{By \geqslant d-Bx^k} \left\| \begin{pmatrix} J(x^k) \\ \lambda_k I \end{pmatrix} y + \begin{pmatrix} f(x^k) \\ 0 \end{pmatrix} \right\|$$

Notice that $Bx^k \geq d$ for all k . The methods of Lecture 7 are applicable to the LSI problem in (b). We are not presently able to comment on the details or success of this iteration.

We remark in closing that if one is willing to forget that ϕ is a sum of squares, then general optimization techniques can be used to solve NLSI. (See [40].)

Sparse Nonlinear Least Squares.

To say that the $\min \frac{1}{2} \sum_{i=1}^{m} f_i(x)^2$ problem is "sparse" is to say that each of the (many) component functions f_i depends on only a few of the parameters $x_1,\ldots,x_n$. It is clear that the Jacobian $J(x)$ of such a system has many zero and few nonzero entries. When we apply (LM) or (GN) we are immediately confronted with a sparse linear LS problem.

Bidiagonalization.

Let $A \in R^{m\times n}$ $(m > n)$ and consider the following algorithm which generates unit vectors $u_i \in R^m$ and unit vectors $v_i \in R^n$:

Algorithm BD.

- Let $u_1 \in R^m$ with $\|u_1\| = 1$.
- For $i = 1,2,\ldots$

$$\alpha_i v_i = A^T u_i - \beta_i v_{i-1} \qquad (\alpha_i \text{ chosen so } \|v_i\| = 1)$$

$$\beta_{i+1} u_{i+1} = A v_i - \alpha_i u_i \qquad (\beta_{i+1} \text{ chosen so } \|u_{i+1}\| = 1)$$

$(\beta_1 v_0 = 0)$ For the moment we discount the possibility of either of the abo righthand sides being zero. If

$$L_k = \begin{pmatrix} \alpha_1 & & & \\ \beta_2 & \alpha_2 & & 0 \\ & \ddots & \ddots & \\ 0 & & \beta_k & \alpha_k \end{pmatrix} \qquad U_k = [u_1 \mid u_2 \mid \cdots \mid u_k] \in R^{m\times k}$$

$$V_k = [v_1 \mid v_2 \mid \cdots \mid v_k] \in R^{n\times k}$$

then the status of BD when i=k can be summarized as follows:

$$A^T U_k = V_k L_k^T$$

$$A V_k = U_k L_k + \beta_{k+1} u_{k+1} e_k^T$$

where e_k is the last column of the kxk identity. These equations can be used in an induction argument to show that

$$U_k^T U_k = V_k^T V_k = I_k$$

That is, the u_i and v_i form orthonormal sets. Furthermore, if u_1 is in the range of A, then the defining equations in BD tell us that all the u_i are in the range of A (an n or less dimensional subspace of R^m) This implies that $Av_n = \alpha_n u_n$ for otherwise u_{n+1} would be well defined by BD implying the inclusion of $u_1, \ldots, u_{n+1}$ in range(A). All these facts allow us to conclude:

$$AV_n = U_n L_n$$

assuming $u_1 \varepsilon$ range(A) and that all the steps in BD can be executed.

Theorem.

Let $A \in R^{mxn}$ and $b \in R^m$. If $L_n y = U_n^T b$ then $\hat{x} = V_n y$ minimizes $\|Ax - b\|$.

Proof.

By defining $y = V_n^T x$ we have $\|Ax - b\| = \|AV_n V_n^T x - b\| = \|U_n L_n y - b\|$

Hence, the minimizing y is given by the normal equations of the transformed problem:

$$(U_n L_n)^T (U_n L_n) y = (U_n L_n)^T b$$

$$L_n^T L_n y = L_n^T U_n^T b \qquad (U_n^T U_n = I)$$

$$L_n y = U_n^T b \qquad \text{(Assuming } L_n \text{ invertible } \equiv \text{ all the } \alpha_i \text{ nonzero)}$$

A Sparse Linear Least Square Algorithm

In understanding this routine, one should bear in mind that $x = V_n y = \sum_{j=1}^{n} y_j v_j$ and that $L_n y = U_n^T b$ which implies

$$y_1 = u_1^T b/\alpha_1 \qquad y_i = (u_i^T b - \beta_i y_{i-1})/\alpha_i \quad i=2,\ldots,n$$

Algorithm SLS

- let $u_1 \epsilon$ range(A) be such that $\| u_1 \| = 1$. (e.g. $u_1 = Az/\| Az \|$)
- $\alpha_1 v_1 = A^T u_1$ (α_1 chosen so $\| v_1 \| = 1$)
- set $y_1 = u_1^T b/\alpha_1$ and $x = y_1 v_1$
- For i=2,...,n

$$\begin{aligned} \beta_i u_i &= Av_{i-1} - \alpha_{i-1} u_{i-1} && \beta_i \text{ chosen so } \| u_i \| = 1 \\ \alpha_i v_i &= A^T u_i - \beta_i v_{i-1} && \alpha_i \text{ chosen so } \| v_i \| = 1 \\ y_i &= (u_i^T b - \beta_i y_{i-1})/\alpha_i \\ x &:= x + y_i v_i \end{aligned}$$

Notice that SLS does not require the storage of any arrays (just a few one dimensional vectors). All the user must supply is a subroutine that can do the matrix-vector multiplies Av and $A^T u$. For this reason SLS is well suited for large and sparse A. Notice that the singular values of A and L are the same and thus a stones throw away by applying the essence of SVD to the lower bidiagonal matrix L.

The successful implementation of SLS in a floating point setting depends upon a great number of details which may be found in Paige[29]. We have glossed over these in hopes of imparting the "general idea".

REFERENCES

[1] Y.Bard,"Comparison of Gradient Methods for the Solution of Nonlinear Parameter Estimation Problems", SIAM J.Num.Anal.,7, 1970, 157-186.

[2] R.H.Bartels, G.H.Golub, and M.A.Saunders, "Numerical Techniques in Mathematical Programming" in Nonlinear Programming, Academic Press, 1970.

[3] A.Bjork,"Solving Linear Least Squares Problems by Gram-Schmidt Orthogonalization", BIT, 7, 1967, 1-21.

[4] A.Bjork,"Iterative Refinement of Linear Least Squares Solutions I", BIT 7, 1967, 257-278.

[5] A.Bjork and G.H.Golub,"Iterative Refinement of Linear Least Squares Solution by Householder Transformations", BIT 7, 1967, 322-337.

[6] K.M.Brown and J.E.Dennis, "New Computational Algorithms for Minimizing a Sum of Squares of Nonlinear Functions", Report 71-6, Computer Science Dept.,Yale Univ., New Haven, Conn., 1971.

[7] K.M.Brown and J.E.Dennis, "Derivative-Free Analogues of the Levenberg-Marquardt and Gauss-Newton Algorithms for Nonlinear Least Squares Approximation", Numer.Math. 18, 1972,289-297.

[8] P.A.Businger, "Updating a Singular Value Decomposition", BIT 10, 1970, 376-385.

[9] D.Cantor and J.W.Evans, "On Approximation by Positive Sums of Powers", SIAM J.Appl.Math., 2, 1970, 380-388.

[10] A.K.Cline, "The Transformation of a Quadratic Programming Problem into Solvable Form , ICASE Report 75-14, NASA Langely Research Center, Hampton, Virginia, 1975.

[11] J.E.Dennis, "Some Computational Techniques for Nonlinear Least Squares Problems" in Numerical Solution of Systems of Nonlinear Algebraic Equations , G.DByrne and C.M.Hall (eds), Academic Press, N.Y., 1973.

[12] J.E.Dennis, "A Brief Survey of Convergence Results for Quasi-Newton Methods" in Nonlinear Programming, R.W.Cottle and C.E.Lemke (eds), AMS, Providence,R.I., 1976.

[13] J.E.Dennis and J.More, "Quasi-Newton Methods, Motivation and Theory", Computer Science Department Report 74-217, Cornell University, Ithaca, New York, 1974.

[14] G.E.Forsythe and C.B.Moler, Computer Solution of Linear Algebraic Systems , Prentice Hall, Engelwood Cliffs, 1967.

[15] W.M.Gentleman, "Least Squares Computations by Givens Transformation Without Square Roots", JIMA 12, 1973, 329-336.

[16] P.Gill, G.H.Golub, W.Murray, and M.A.Saunders, "Methods for Modifying Matrix Factorizations", Math.Comp. 28,1974,505-535.

[17] G.H.Golub,"Some Modified Matrix Eigenvalue Problems" , SIAM Review 15 , 1973, 318-334.

[18] G.H.Golub and P.A.Businger,"Linear Least Squares Solutions by Householder Transformations", Numer.Math. 7, 1965, 269-276.

[19] G.H.Golub and W.Kahan, "Calculating the Singular Values and Pseudo-inverse of a Matrix", J.SIAM Numer.Anal.Ser.B, 2, 1965, 206-216.

[20] G.H.Golub and V.Pereyra, "The Differentiation of Pseudoinverses and Nonlinear Least Squares Problems Whose Variables Separate", SIAM J.Num.Anal.,10, 1973, 413-432.

[21] G.H.Golub and V.Pereyra, "Differentiation of Pseudoinverses, Separable Nonlinear Least Squares Problems and Other Tales", Manuscript.

[22] G.H.Golub and C.Reinsch, "Singular Value Decomposition and Least Squares Solutions", Numer.Math. 14, 1970, 403-420.

[23] G.H.Golub and J.H.Wilkinson, "Note on the Iterative Refinement of Least Squares Solutions", Numer.Math. 9, 1967,139-148.

[24] L.Kaufman, "A Variable Projection Method for Solving SEparable Nonlinear Least Squares Problems", BIT 15, 1975, 49-57.

[25] C.L.Lawson and R.J.Hanson, Solving Least Squares Problems, Prentice-Hall, Engelwood Cliffs, 1974.

[26] D.W.Marquardt, "An Algorithm for Least Squares Estimation of Nonlinear Parameters", J.of SIAM, 11, 1963, 431-441.

[27] D.W.Marquardt, "Generalized Ridge Regression, Biased Linear Estimation and Nonlinear Estimation, Technometrics 12, 1970 591-612.

[28] L.Nazareth, "A Hybrid Least Squares Method", To Appear in Transactions on Mathematical Software.

[29] C.C,Paige, "Bidiagonalization of Matrices and Solution of Linear Equations", SIAM J.Numer.Anal. 11, 1974,197-209.

[30] M.J.D.Powell, "A Method for Minimizing a Sum of Squares of Nonlinear Functions without Calculating Derivatives", Comp.J. 1 , 1965, 303-307.

[31] M.J.D.Powell, "Problems Related to Unconstrained Optimization" in Numerical Methods for Unconstrained Optimization , W.Murray (ed.), Academic Press, N.Y., 1972.

[32] G.W.Stewart, Introduction to Matrix Computations, Academic Press, N.Y., 1973.

[33] J.Stoer, "On the Numerical Solution of Constrained Least Squares Problems", SIAM J.Numer.Anal. 8, 1971, 382-411.

[34] C.F.Van Loan, "Generalizing the Singular Value Decomposition", SIAM J.Numer.Anal. 13, 1976, 76-83.

[35] R.H.Barham and W.Drane, "An Algorithm for Least Squares Estimation of Nonlinear Parameters When Some of the Parameters are Linear", Technometrics 14, 1972, 757-766.

[36] A.Bjork, "Iterative Refinement of Linear Least Squares Solutions,II", BIT 8, 1968, 8-30.

[37] R.Brent, Algorithms for Minimization Without Derivatives , Prentice-Hall, Engelwood Cliffs, 1973.

[38] R.Fletcher, "A Modified Marquardt Subroutine for Nonlinear Least Squares, UKAEA Research Group Report AERE R6799, Harwell, Oxfordshire, England, 1971.

[39] C.G.Broyden, "A Class of Methods for Solving Nonlinear Simultaneous Equations", Math.Comp. 19, 1965, 577-593.

[40] S.LS.Jacoby, J.S.Kowalik, and J.T.Pizz0, Iterative Methods for Nonlinear Optimization Problems , Prentice Hall, 1972.

[41] M.J.D.Powell, AERE Harwell Library Subroutine VA05A, Harwell, Oxfordshire, England, 1971.

[42] L.Nazareth, "Some Recent Approaches to Solving Large Residual NOnlinea Least Squares Problems", Manuscript, 1976.

r