CHAPTER 3: UNCONSTRAINED OPTIMIZATION

W. Erwin Diewert May 2003

1. First and Second Order Conditions for A Local Min or Max.

Consider the problem of maximizing or minimizing a function of N variables, $f(x_1, . . ., x_N) = f(x)$. From Chapter 1, we found that the first order necessary conditions for x^0 to be a local minimizer or maximizer for f were:

(1) $D_v f(x^0) = 0$ for all directions $v \neq 0_N$

where the first order directional derivative of f in the direction v evaluated at the point x is defined as

(2) $D_v f(x) \equiv \lim_{t \to 0} [f(x + tv) - f(x)] / t$.

In the case where the first order partial derivatives of f exist and are continuous around x^0, we found that conditions (1) were equivalent to the following N first order necessary conditions for x^0 to be a local minimizer or maximizer for f:

(3) $f_1(x^0) = 0$; $f_2(x^0) = 0$; . . . ; $f_N(x^0) = 0$

where the ith first order partial derivative of f is defined as

(4) $f_i(x) \equiv \lim_{t \to 0}[f(x + te_i) - f(x)]/t$; $i = 1, . . ., N$

where e_i is the ith unit vector.

It is convenient to introduce a symbol to denote the *vector* of first order partial derivatives of f evaluated at the point x:

(5) $\nabla f(x) \equiv [f_1(x), f_2(x), . . ., f_N(x)]^T$.

Note that we have defined $\nabla f(x)$ (called the *gradient vector* of f evaluated at x) to be column vector.

Using the notation (5), the system of first order conditions (3) can be written more efficiently as:

(6) $\nabla f(x^0) = 0_N$.

Also using (5), it can be seen that our old First Order Directional Derivative Theorem can be written as:

(7) $D_v f(x^0) = \sum_{i=1}^{N} v_i f_i(x^0) = v^T \nabla f(x^0)$.

Recall that we required the first order partial derivative functions $f_i(x)$ to exist and be continuous around x^0, in order to derive the formula (7).

It is also convenient to introduce a notation for the N by N matrix of second order partial derivatives of f evaluated at the point x:

(8) $$\nabla^2 f(x) \equiv \begin{bmatrix} f_{11}(x), & \ldots, & f_{1N}(x) \\ \vdots & & \vdots \\ f_{N1}(x), & \ldots, & f_{NN}(x) \end{bmatrix}$$

where the ijth element in $\nabla^2 f(x)$ is defined as

(9) $f_{ij}(x) \equiv \lim_{t \to 0}[f_i(x + te_j) - f_i(x)]/t$,

where $f_i(x)$ is the ith first order partial derivative of f evaluated at the point x. The N by N matrix $\nabla^2 f(x)$ is called the *Hessian matrix* of f evaluated at x.

Recall that the directional derivative of the function $D_v f(x)$ evaluated at x in the direction $u \neq 0_N$ is defined as:

(10) $D_{vu} f(x) \equiv \lim_{t \to 0}[D_v f(x + tu) - D_v f(x)]/t$.

Using (8), it can be seen that our old Second Order Directional Derivative Theorem can be written as follows:

(11) $D_{vu} f(x) = v^T \nabla^2 f(x) u$.

Recall that in order to prove (11), we required the existence and continuity of the second order partial derivative functions $f_{ij}(x)$ around the point x.

Armed with formula (11), we can state *second order sufficient conditions* for x^0 to be a *strict local minimizer* of f: in addition to the first order conditions (6), we require the following *second order conditions*:

(12) $D_{vv} f(x^0) = v^T \nabla^2 f(x^0)\, v > 0$ for all $v \neq 0_N$.

If it is convenient, we can replace $v \neq 0_N$ in (12) by $v^T v = 1$ in order to obtain an equivalent set of conditions.

Since conditions (6) are equivalent to conditions (1), it can be seen that conditions (6) and (12) are analogues to our single variable calculus conditions for a strict local minimum, except that these univariate conditions now have to hold for all possible directions v.

The counterpart to the univariate *second order necessary conditions* for x^0 to be a *local minimizer* for f are conditions (6) plus the following second order conditions:

(13) $D_{vv}f(x^0) = v^T\nabla^2 f(x^0)\, v \geq 0$ for all $v \neq 0_N$.

Obviously, there are analogous *sufficient conditions* for x^0 to be a *strict local maximizer* for f: in addition to (6), we require

(14) $D_{vv}f(x^0) = v^T\nabla^2 f(x^0)\, v < 0$ for all $v \neq 0_N$.

Finally, the analogous *necessary conditions* for x^0 to be a *local maximizer* for f are (6) and the following second order conditions:

(15) $D_{vv}f(x^0) = v^T\nabla^2 f(x^0)\, v \leq 0$ for all $v \neq 0_N$.

Notice that if $N \geq 2$, then the second order conditions (12) - (15) involve checking an infinite number of inequalities. In Chapter 1, we have shown how this task can be accomplished in the case where N = 2. In section 4 below, we will show how to do this checking of inequalities for a general N. However, before we do this, it is useful to develop a few properties of quadratic functions.

2. Taylor's Theorem and Quadratic Approximations

Taylor's Theorem: Let f(x) be a function of one variable defined over the interval $x^0 \leq x \leq x^1$ where $x^0 < x^1$. Suppose the n-1 derivative of f, $f^{(n-1)}(x)$, exists and is continuous over this interval and suppose that the nth derivative of f, $f^{(n)}(x)$, exists for x such that $x^0 < x < x^1$. Define the "remainder" R by the following equation:

(16) $f(x^1) = f(x^0) + \sum_{k=1}^{n-1} \frac{(x^1-x^0)^k}{k!} f^{(k)}(x^0) + R.$

Then there exists a point x* such that $x^0 < x^* < x^1$ and

(17) $R = (x^1 - x^0)^n f^{(n)}(x^*) \,/\, n!.$

Proof: Define the number M by the following equation:

(18) $f(x^1) = f(x^0) + \sum_{k=1}^{n-1} \frac{(x^1-x^0)^k}{k!} f^{(k)}(x^0) + M\frac{(x^1-x^0)^n}{n!}.$

Define the function F(x) by:

(19) $F(x) \equiv -f(x^1) + f(x) + \sum_{k=1}^{n-1} [(x - x^0)^k/k!]\, f^{(k)}(x) + M(x - x^0)^n/n!\,.$

It can be seen that $F(x^1) = 0$ and by using (18), it can be seen that $F(x^0) = 0$ as well. Thus the function $F(x)$ is continuous for x such that $x^0 \leq x \leq x^1$ and F is such that $F(x^0) = F(x^1)$. Thus the function F must attain a local min or a local max for at least one x* such that $x^0 < x^* < x^1$. The first order necessary conditions for a min or a max of F(x) must hold at $x = x^*$ so we have

$$\begin{aligned} 0 &= F'(x^*) \\ &= f'(x^*) + \sum_{k=1}^{n-1} k \frac{(x^1-x^*)^{k-1}}{k!}(-1)f^{(k)}(x^*) + \sum_{k=1}^{n-1} \frac{(x^1-x^*)^{k}}{k!} f^{(k+1)}(x^*) \\ &\qquad + Mn\frac{(x^1-x^*)^{n-1}}{n!}(-1) \end{aligned}$$

differentiating the F defined by (19)

$$\begin{aligned} &= f'(x^*) - f'(x^*) - (x^1 - x^*)f^{(2)}(x^*) - \frac{(x^1-x^*)^2}{2!} f^{(3)}(x^*) - \ldots - \frac{(x^1-x^*)^{n-2}}{(n-2)!} f^{(n-1)}(x^*)2! \\ &\qquad + (x^1 - x^*)f^{(2)}(x^*) + \frac{(x^1-x^*)^2}{(n-2)!} f^{(3)}(x^*) + \ldots + \frac{(x^1-x^*)^{n-1}}{(n-1)!} f^{(n)}(x^*) - \\ &\qquad \frac{(x^1-x^*)^{n-1}}{(n-1)!} M \end{aligned}$$

(20) $= (x^1 - x^*)^{n-1}[f^{(n)}(x^*) - M]/(n-1)!$ cancelling terms.

Since $x^0 < x^* < x^1$ and hence $x^1 - x^* > 0$, we see that (20) implies

(21) $f^{(n)}(x^*) = M.$

Now substitute (21) into (18) and we obtain (16) where R is defined by (17).

Q.E.D.

Note that Taylor's Theorem reduces to the Mean Value Theorem if we set n = 1.

Where n = 2, Taylor's Theorem becomes, letting x^1 be replaced by x:

(22) $f(x) = f(x^0) + (x - x^0)f'(x^0) + R.$

If we drop the remainder term R on the right hand side of (22), what is left is called the *linear approximation* to f around the point x^0, i.e., define

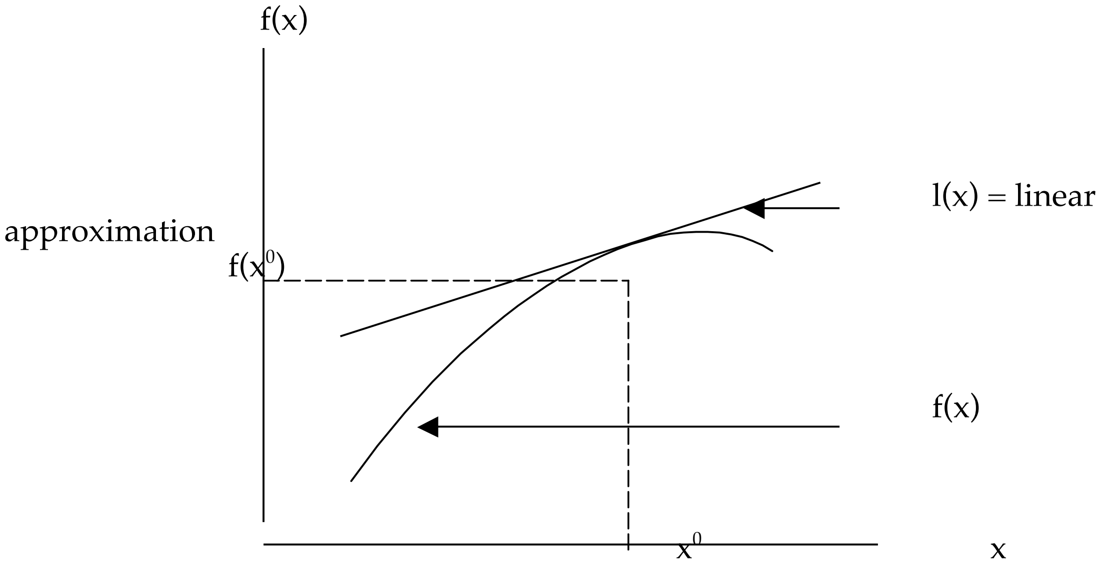
(23) $l(x) \equiv f(x^0) + (x - x^0)f'(x^0).$

Then for x "reasonably" close to x^0, l(x) will approximate f(x) "reasonably" well:

(24) $f(x) \approx f(x^0) + (x - x^0)f'(x^0).$

Note that $l(x^0) = f(x^0)$ and $l'(x^0) = f'(x^0)$; i.e., the linear approximation to f around the point x^0 has the same level and first derivative as f when evaluated at $x = x^0$.

Figure 1: The Linear Approximation to f at x^0



When n = 3, Taylor's Theorem becomes, letting x^1 be replaced by x:

(25) $f(x) = f(x^0) + (x - x^0)f'(x^0) + (1/2)(x - x^0)^2\ f''(x^0) + R.$

If we drop the remainder term R on the right hand side of (25), what is left is called the *quadratic approximation* to f around the point x^0:

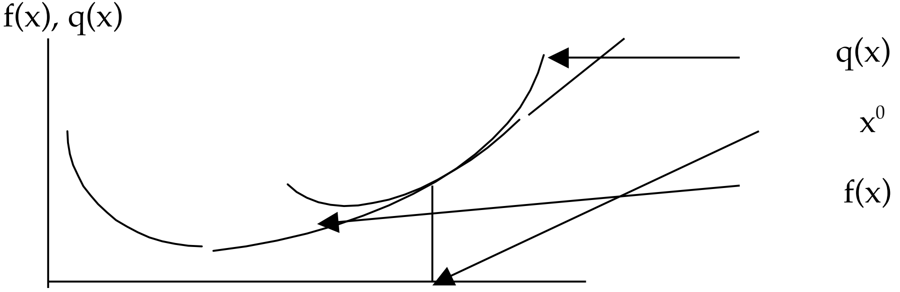
(26) $q(x) \equiv f(x^0) + (x - x^0)f'(x^0) + (1/2)(x - x^0)^2 f''(x^0).$

Note that the quadratic approximation to f(x) around the point x^0 will have the same level and first and second derivatives evaluated at $x = x^0$; i.e., we have

(27) $q(x^0) = f(x^0)\ ;\quad q'(x^0) = f'(x^0)\ ;\quad q''(x^0) = f''(x^0).$

The quadratic approximation to f around the point x^0 will generally more closely approximate f around x^0 than the corresponding linear approximation.

Figure 2: The Quadratic Approximation to f at x^0



The concepts of linear and quadratic approximations to general nonlinear functions can be extended to functions of N variables using the univariate analysis developed above. Let $f(x) = f(x_1, x_2, . . ., x_N)$ be a function of N variables with continuous first and second order partial derivatives. Now use f in order to define the following function of a single variable t:

(28) $g(t) \equiv f(x^0 + t(x - x^0))$; $0 \le t \le 1$.

Thus we have:

(29) $g(0) = f(x^0)$ and $g(1) = f(x)$.

Now apply the linear approximation idea to g around the point t = 0. Thus we have:

$$g(t) \mid\mid g(0) + g\square(0)(t\text{-}0)$$

$$= f(x^0) + \square_{i=1}^{N} f_{\,i}(x^0)(x_i - x^0)(t\text{-}0)$$

differentiating (28) with respect to t and evaluating the derivatives at t = 0

(30) $$= f(x^0) + t\ \square^T f(x^0)(x - x^0)$$

rearranging terms.

Letting t = 1 and using (29), (30) becomes:

(31) $f(x) \mid\mid f(x^0) + \square^T f(x^0)(x - x^0)$

and the right hand side of (31) can be regarded as the *linear approximation* to f(x) around $x = x^0$.

In order to calculate the quadratic approximation to g(t) around t = 0, we need to calculate the first and second derivatives of g(t). Differentiating (28) with respect to t, we obtain:

(32) $g\square(t) = \square_{i=1}^{N} f_{\,i}(x^0 + t(x - x^0))(x_i - x_i^0)$

$$= \square^T f\,(x^0 + t(x - x^0))(x - x^0);$$

(33) $g\square\square(t) = \square_{i=1}^{N}\square_{j=1}^{N} f_{ij}(x^0 + t(x \square x^0))(x_i \square x_i^0)(x_j \square x_j^0)$

$$= (x - x^0)^T\ \square^2 f(x^0 + t(x - x^0))(x - x^0).$$

Thus the quadratic approximation to g around the point t = 0 is:

(34) $g(t) \approx g(0) + g'(0)\,(t - 0) + (1/2)g''(0)(t - 0)^2$

(35) $= f(x^0) + t\,\nabla^T f(x^0)(x - x^0) + (1/2)t^2(x - x^0)^T\nabla^2 f(x^0)(x - x^0)$

where (35) follows from (34) using (29), (32) and (33). Now evaluate (35) at t = 1 and using (29), we have:

(36) $f(x) \approx f(x^0) + \nabla^T f(x^0)(x - x^0) + (1/2)(x - x^0)^T\nabla^2 f(x^0)(x - x^0).$

Note that the right hand side of (36) is a quadratic function of x; it is called the *quadratic approximation* to f(x) around the point $x = x^0$. Note that Young's Theorem ($f_{ij}(x^0) = f_{ji}(x^0)$) for all $i \neq j$) implies that the N by N matrix $\nabla^2 f(x^0)$ in (36) will be symmetric.

Linear and quadratic approximations to general nonlinear functions of N variables are widespread in economics, science, engineering, statistics and business.

3. Rules for Differentiating Linear and Quadratic Functions.

Suppose f(x) is a linear function of N variables; i.e.,

(37) $f(x) \equiv a + \sum_{i=1}^{N} b_i x_i = a + b^T x$

where $b^T \equiv [b_1, b_2, \ldots, b_N]$. Partially differentiating the f(x) defined by (37) with respect to the components of x yields:

(38) $f_i(x) = \partial f(x)/\partial x_i = b_i; \qquad i = 1, 2, \ldots, N.$

Obviously, equations (38) can be rewritten as:

(39) $\nabla f(x) = b \qquad \text{if} \qquad f(x) \equiv a + b^T x;$ *Rule 1.*

If we further differentiate the $f_i(x)$ defined by (38) with respect to the components of x, we obtain:

(40) $f_{ij}(x) \equiv \partial^2 f(x)/\partial x_i \partial x_j = 0; \qquad 1 \le i, j \le N.$

The equations (40) can be written more compactly as:

(41) $\nabla^2 f(x) = 0_{N\times N} \qquad \text{if} \qquad f(x) \equiv a + b^T x;$ *Rule 2.*

Now suppose f(x) is the following (homogeneous) quadratic function of N variables; i.e.,

(42) $f(x) \equiv \sum_{i=1}^{N} \sum_{j=1}^{N} a_{ij}\, x_i\, x_j$
$= x^T Ax$ where $A = A^T$.

Note that we are assuming that the matrix of coefficients $A \equiv [a_{ij}]$ in (42) is *symmetric*; i.e., we have $a_{ij} = a_{ji}$ for all $i \neq j$. We want to calculate the first and second order partial derivatives of the f defined by (42). Let us first consider the case $N = 2$. In this case, taking into account the fact that $a_{12} = a_{21}$, we have:

(43) $f(x_1, x_2) = a_{11}\, x_1^2 + 2a_{12}x_1x_2 + a_{22}\, x_2^2.$

The first order partial derivatives of (43) are:

(44) $f_1(x_1, x_2) = 2a_{11}x_1 + 2a_{12}x_2;$

$f_2(x_1, x_2) = 2a_{12}x_1 + 2a_{22}x_2.$

Equations (44) can be written as:

(45) $\nabla f(x) = 2Ax$ if $f(x) \equiv x^TAx,\ A = A^T;$ *Rule 3.*

If we partially differentiate the $f_i(x_1, x_2)$ in (44) with respect to x_1 and x_2, we obtain the following second order derivatives:

(46) $f_{11}(x_1, x_2) = 2a_{11};$ $f_{12}(x_1, x_2) = 2a_{12};$
$f_{21}\,(x_1, x_2) = 2a_{12};$ $f_{22}(x_1, x_2) = 2a_{22}.$

Using matrices, equations (46) can be rewritten as:

(47) $\nabla^2 f(x) = 2A$ if $f(x) \equiv x^TAx,\ A = A^T;$ *Rule 4.*

It can be verified that Rules 3 and 4 hold for a general N and not only the cases $N = 1$ and $N = 2$.

Rules 1 to 4 are extremely useful and should be memorized.

Problems:

1. Verify Rules 3 and 4 for the case $N = 3$.

2. Consider the following system of equations:

(i) $y = Xx + e$

where y and e are M dimensional vectors, X is an M by N matrix and x is an N dimensional vector. Define the function f(x) as

$$f(x) \equiv e^Te = \sum_{i=1}^{M} e_i^2$$

$$= (y - Xx)^T(y - Xx) \qquad \text{using (i) to solve for e}$$

$$= (y^T - x^TX^T)(y - Xx)$$

$$= y^Ty - y^TXx - x^TX^Ty + x^TX^TXx$$

(ii) $$= y^Ty - 2y^TXx + x^TX^TXx \qquad \text{using } y^TXx = [x^TX^Ty]^T.$$

Assume that $(X^TX)^{-1}$ exists.

(a) Show that $\hat{x} \equiv (X^TX)^{-1} X^Ty$ satisfies the system of first order conditions for minimizing f(x):

(iii) $\nabla f(\hat{x}) = 0_N$.

[In statistics, $\hat{x}$ is known as the *least squares estimator* for the vector of parameters x].

(b) Show that $\nabla^2 f(x)$ does not depend on x.

(c) Show that

(iv) $v^T\nabla^2 f(\hat{x})\, v > 0$ for every $v \neq 0_N$

and so $\hat{x}$ is in fact a local minimizer for f(x). [This part of the problem is difficult]. This problem shows that the least squares estimator $\hat{x}$ actually does minimize the sum of squared errors e^Te with respect to the vector of coefficients x.

4. Quadratic Forms and Definite Matrices

Let A be an N by N *symmetric* matrix and consider the following definitions:

(48) A is *positive definite* iff $x^TAx > 0$ for all $x \neq 0_N$;

(49) A is *negative definite* iff $x^TAx < 0$ for all $x \neq 0_N$;

(50) A is *positive semidefinite* iff $x^TAx \geq 0$ for all $x \neq 0_N$;

(51) A is *negative semidefinite* iff $x^TAx \leq 0$ for all $x \neq 0_N$;

(52) A is *indefinite* iff it is none of (48) - (51).

Recall the second order conditions (12) - (15) that were discussed in section 1 above. If we let the A matrix in this section equal $\nabla^2 f(x^0)$ in section 1, it can be seen that (48) corresponds to conditions (12) for a strict local minimum, (49) corresponds to conditions (14) for a strict local maximum, (50) corresponds to the second order necessary conditions (13) for a local minimum and (51) corresponds to the second order necessary conditions (15) for a local maximum.

In the following section, we show how the Gaussian triangularization procedure can be adapted to determine whether a symmetric matrix A has any of the definiteness properties (48) - (52).

Problem:

3. Let D be an N by N diagonal matrix with main diagonal elements d_{ii} for i = 1, 2, . . ., N. Determine what restrictions the d_{ii} must satisfy in order for D to be: (i) positive definite; (ii) negative definite; (iii) positive semidefinite; (iv) negative semidefinite and (v) indefinite; (assume $N \geq 2$ for this case).

5. The Method of Lagrange and Gauss for Diagonalizing a Symmetric Matrix

Recall the Gaussian triangularization procedure that was discussed in section 3 of Chapter 2 on Elementary Matrix Algebra. If A is a symmetric N by N matrix, then this algorithm can readily be modified to transform A into a diagonal matrix.

Consider Stage 1 of our old algorithm where we added multiples of one row of A to other rows of A to create zeros below the first component of the first column of A. We again apply Stage 1 of our old algorithm, but before we proceed to Stage 2, we now add multiples of the final Stage 1 first column to the remaining columns of the transformed A matrix to create zeros in the remainder of row 1. In other words, we repeat the sequence of elementary row operations that we used to accomplish Stage 1 of the algorithm but now we apply the same sequence to the columns as well.

More explicitly, consider the 3 cases for Stage 1 of our old algorithm. In case (i), we had $a_{11} \neq 0$, and at the of Stage 1, the transformed A matrix had the following form (the E_n represent elementary row operation matrices that add multiples of the first row of A to the remaining rows of A):

$$(53) \quad \begin{bmatrix} a_{11}, & a_{12}, & \ldots, & a_{1N} \\ 0_{N-1}, & & A^{(2)} & \end{bmatrix} = E_N E_{N\text{-}1} \ldots E_2 A.$$

Now add $-a_{12}/a_{11}$ times the first column of (53) to the second column of (53); add $-a_{13}/a_{11}$ times the first column of (53) to the 3rd column of (53); . . .; add $-a_{1N}/a_{11}$ times the first column of (53) to the Nth column of (53). It can be verified that these elementary column operations can be performed by multiplying (53) on the right by $E_2^T E_3^T \ldots E_N^T$; i.e., the transposes of the sequence of row operation matrices $E_2, E_3, \ldots, E_N$ sweep out $a_{12} = a_{21}, a_{13} = a_{31}, \ldots, a_{1N} = a_{N1}$. Thus we have

$$(54) \quad E_N E_{N-1} \ldots E_2 A E_2^T E_3^T \ldots E_N^T = \begin{bmatrix} a_{11}, & 0_{N-1}^T \\ 0_{N-1}, & A^* \end{bmatrix}$$

at the end of our new Stage 1 Algorithm for case (i) where $a_{11} \neq 0$.

If we take transposes of both sides of (54), we deduce that the matrix on the left hand side of (54) is symmetric. Hence A^* on the right hand side of (54) must also be symmetric. Hence, we can now apply the next stage of our modified algorithm to the N-1 by N-1 symmetric matrix A^*.

Now suppose that at Stage 1 of our old algorithm, case (iii) occurred, i.e., $a_{i1} = 0$ for $i = 1, 2, \ldots, N$. But since A is now assumed to be symmetric, we have $a_{1j} = 0$ as well for $j = 1, 2, \ldots, N$. Thus in case (iii), A has the following form:

$$(55) \quad A = \begin{bmatrix} 0, & 0_{N-1}^T \\ 0_{N-1}, & A^* \end{bmatrix}$$

which is the required form for the next stage of our modified algorithm.

Finally, suppose that at Stage 1 of our old algorithm, case (ii) occurred; i.e., $a_{11} = 0$ but $a_{i1} \neq 0$ for some $i > 1$. Recall that in our old algorithm, we added row i of A to the first row of A and then applied the case (i) operations to the transformed matrix. In the present algorithm, we not only add row i of A to row 1, we then immediately add column i of the transformed matrix to column 1. The resulting matrix will be symmetric with the element $2a_{i1} + a_{ii}$ in the northwest corner of the transformed matrix. We now need to consider 2 cases:

Case (a): $2a_{i1} + a_{ii} \neq 0$

In this case, we can now apply our new case (i) algorithm on the previous page to this transformed matrix. If we denote E_1 as the elementary row matrix that adds row i of A to row 1, then we have the following decomposition at the end of Stage 1 of our new algorithm:

$$(56) \quad E_N E_{N-1} \ldots E_2 E_1 A E_1^T E_2^T \ldots E_N^T = \begin{bmatrix} 2a_{i1} + a_{ii}, & 0_{N-1}^T \\ 0_{N-1}, & A^* \end{bmatrix}$$

i.e., we have again reduced A into block diagonal form where A^* is a symmetric N-1 by N-1 matrix.

Case (b): $2a_{i1} + a_{ii} = 0$.

In this case, if we look at row 1 and row i and column 1 and column i of the original A matrix, this 2 by 2 submatrix of A has the following form (using $a_{11} = 0$ and $a_{ii} = - 2a_{i1}$):

$$\begin{bmatrix} 0, & a_{i1} \\ a_{i1}, & -2a_{i1} \end{bmatrix}$$

After adding row i of the original matrix to row 1 and then adding column i of the original matrix to column 1, the above 2 by 2 submatrix is transformed into:

$$\begin{bmatrix} 0, & -a_{i1} \\ -a_{i1}, & -2a_{i1} \end{bmatrix}$$

so we have not succeeded in getting a nonzero element in the northwest corner of the transformed matrix. However, to solve this problem, all we have to do is add row i of the transformed matrix to row 1 and then add column i of the transformed matrix to column 1. Then the new transformed A matrix will be symmetric and have $- 4a_{i1} \neq 0$ in the top northwest corner. Hence in this case (b), we can again obtain a counterpart to (56) where $- 4a_{i1}$ will replace $2a_{i1} + a_{ii}$ in the northwest corner of the matrix on the right hand side of (56). Hence in both cases (a) and (b), we have again reduced A into block diagonal form where A* is an N-1 by N-1 symmetric matrix.

Hence, for all cases, at the end of Stage 1 of our new algorithm, we have reduced A into the following block diagonal form:

$$(57) \qquad \begin{bmatrix} d_{11}, & 0_{N-1}^T \\ 0_{N-1}, & A^* \end{bmatrix}$$

At Stage 2 of the algorithm, we apply the same type of elementary row and column operations to the symmetric matrix A* and at the end of Stage 2, we have reduced A* into the following form:

$$(58) \qquad A^* = \begin{bmatrix} d_{22}, & 0_{N-2}^T \\ 0_{N-2}, & A^{**} \end{bmatrix}$$

where A** is an N-2 by N-2 symmetric matrix.

Now further reduce A** into block diagonal form; etc.

Finally, at the end of Stage N, we have transformed A into diagonal form by means of a sequence of elementary row and column operations where we add multiples of one row to another row and then repeat the same operation to the corresponding columns. If we let the N by N matrix E denote the product of all of the elementary row matrices, then we have

(59) $EAE^T = D$

where $D = [d_{ij}]$ and $d_{ij} = 0$ if $i \neq j$.

Example: Let $A \equiv \begin{bmatrix} 1 & 2 & 3 \\ 2 & 1 & 0 \\ 3 & 0 & 1 \end{bmatrix}$

Stage 1: We are in case (i): $a_{11} = 1 \neq 0$. Hence take -2 times row 1 and add to row 2; take -3 times row 1 and add to row 3. We obtain the following matrix:

$$\begin{bmatrix} 1, & 2, & 3 \\ 0, & -3, & -6 \\ 0, & -6, & -8 \end{bmatrix}.$$

Now take -2 times column 1 and add to column 2; take -3 times column 1 and add to column 3; get:

$$\begin{bmatrix} 1, & 0, & 0 \\ 0, & -3, & -6 \\ 0, & -6, & -8 \end{bmatrix}.$$

Stage 2: Now take -2 times row 2 and add to row 3; get:

$$\begin{bmatrix} 1, & 0, & 0 \\ 0, & -3, & -6 \\ 0, & 0, & 4 \end{bmatrix}.$$

Finally, take -2 times column 2 and add to column 3; get:

(60) $D = \begin{bmatrix} 1, & 0, & 0 \\ 0, & -3, & 0 \\ 0, & 0, & 4 \end{bmatrix}$, a diagonal matrix.

The two elementary row matrices that we used at Stage 1 of the algorithm were:

(61) $E_1 \equiv \begin{bmatrix} 1, & 0, & 0 \\ -2, & 1 & 0 \\ 0, & 0, & 1 \end{bmatrix}$; $E_2 \equiv \begin{bmatrix} 1, & 0, & 0 \\ 0, & 1, & 0 \\ -3, & 0, & 1 \end{bmatrix}$.

The final elementary row matrix that we used at Stage 2 of the algorithm was:

(62) $E_3 = \begin{bmatrix} 1, & 0, & 0 \\ 0, & 1, & 0 \\ 0, & -2, & 1 \end{bmatrix}.$