Department of Mathematics
The University of Southwestern Louisiana
Ph.D. Qualifying Examination: Numerical Analysis
August, 1990

Instructions: This portion of the exam is open-book, open notes; it is recommended that you have the Golub / van Loan and Ames texts with you. You need only complete three of the four problems, but you must do what you do well. Make sure to organize and check your work.

1. Suppose that we wanted to compute a solution to the linear boundary value problem

$$a(x,y)u_{xx} + b(x,y)u_{yy} = -1, \qquad (x,y) \in [0,1]\times[0,1]$$
$$u(x,y) = 0 \quad \text{for } (x,y) \in \partial([0,1]\times[0,1]),$$

using central differences with N internal mesh points in each spatial direction. The conjugate gradient method as embodied in Algorithm 10.2.1 on p. 523 of Golub and van Loan (**Matrix Computations**, second edition) would be appropriate if it were implemented properly. Write a Fortran subroutine to evaluate the vector Ap_k in that algorithm. The subroutine should not contain storage for the matrix A, and should call function subroutines $a(x,y)$ and $b(x,y)$. The routine should run well on a scalar processor.

2. Discuss whether the subroutine given in Problem 1 would run well on a vector processor. If not, then rewrite it so that it will. *(Hint: you may compute all of the values of a and b simultaneously.)*

3. Consider the nonlinear boundary value problem

$$u_{xx} = -u^2 \qquad u(0) = 1, \quad u(1) = 1.$$

   (a) Using central differences on $[0,1]$ with 3 equally spaced internal mesh points, write down a nonlinear system of equations which approximates this nonlinear boundary value problem. Do an iteration of Newton's method, starting with $u_1 = u_2 = u_3 = 1$, on this system, and compute the resulting residual.

   (b) Discretize the problem using a Galerkin method with basis functions $\phi_1(x) \equiv 1$ and $\phi_2(x) \equiv \sin(\pi x)$. Do an iteration of Newton's method, starting with $a_1 = 1$ and $a_2 = 0$, where the approximate solution is $u_2(x) = a_1\phi_1(x) + a_2\phi_2(x)$.

4. If A is an m by n matrix, with $m \gg n$, then we may wish to find a least squares solution to the (usually) overdetermined system of equations $Ax = b$. A common technique is the following algorithm.

   (i) Form a QR factorization, i.e., to write $A = QR$, where $Q'Q = I$ and R is right triangular.

   (ii) form $w = Q'b$,

   (iii) For $i = n$ to 1 by -1:

$$x_i \leftarrow \frac{w_i - \sum_{j=i+1}^{n} r_{i,j}x_j}{r_{i,i}}.$$

Note that Q in the above algorithm is m by m; in certain instances, m may be so large that it is undesirable to store Q. An alternative is to factor A in the form $A = LQ$, where L is *left* or lower triangular, and Q is a (different) n by n orthogonal matrix (i.e. $Q'Q = I$). In such instances, only Q and $L'L$ are stored. Write an algorithm as above, assuming you are able to obtain Q and $L'L$. *(Hint: In this case, you may need to work with a more ill-conditioned system.)*