Introduction to Numerical Simulation (Fall 2003)
Problem Set #4 - due October 3rd

1) In this question we will compare the GCR algorithm implementations in *sgcr.m* and *tgcr.m*.

a) What is the difference between the two algorithms? Which is better and why?

b) Suppose you were trying to generate an example for which one of the gcr algorithms convergences faster than the other. Why would you avoid using a symmetric test example?

c) Please find an example for which one GCR algorithm converges faster than the other.

2) Suppose the GCR algorithm is applied to solving $Mx = b$, where M is symmetric and has all but two of its eigenvalues in the interval $[0.5, 3]$, the other two eigenvalues being 20 and 200.

a) Show that the residual produced by the third iteration of the GCR algorithm satisfies the inequality

$$\frac{\|r^3\|}{\|r^0\|} \leq 3\frac{\sqrt{6}-1}{\sqrt{6}+1}.$$

(Hint: think about constructing a third-order polynomial of the form $p_3(x) = 1 + ax + bx^2 + cx^3$.)

b) What is the best bound you can find for

$$\frac{\|r^k\|}{\|r^0\|}.$$

c) Experimentally verify parts (a) and (b) by constructing an example 100×100 diagonal matrix with the above eigenproperties. Be sure to pick an example that has a sufficient number of distinct eigenvalues, and plot $\frac{\|r^k\|}{\|r^0\|}$ as a function of k on a log-linear plot to verify your bounds.

3) Numerical simulation of ocean flow dynamics is used in weather prediction and in research on global warming. The commonly used simulation techniques are based on repeated solution of a Poisson equation for hydrostatic pressures. It is quite common to compute hydrostatic pressures by solving the Poisson equation on a domain that covers the entire Atlantic ocean, resulting in millions of unknowns. The boundary conditions for that Poisson equation then depend on the selected physical model. In this problem, you will examine solving a discretized two-dimensional Poisson equation problem associated with modeling the ocean, and examine the interactions between the choice of physical model (hence boundary conditions) and the rate of GCR convergence.

The ocean is an unusual domain because it is much wider than deep. To model this, consider solving a two-dimensional Poisson equation,

$$\frac{\partial^2 p(x,y)}{\partial x^2} + \frac{\partial^2 p(x,y)}{\partial y^2} = b(x,y),$$

where $x \in [0, 100]$ is the horizontal position, $y \in [0, 1]$ is the vertical position or depth, and $p(x, y)$ is the hydrostatic pressure at position x, y. Here $b(x, y)$ is a right-hand side which models Coriolis, convective, or other induced forces. The boundary conditions are, for the pure Neumann model,

$$\frac{\partial p(x,y)}{\partial y} = 0 \quad y = 0 \;\; or \;\; y = 1$$

$$\frac{\partial p(x,y)}{\partial x} = 0 \quad x = 0 \;\; or \;\; x = 100,$$

or, for the partly Dirichlet model,

$$p(x,y) = 0 \quad y = 0, \qquad \frac{\partial p(x,y)}{\partial y} = 0 \quad y = 1$$

$$\frac{\partial p(x,y)}{\partial x} = 0 \quad x = 0 \;\; or \;\; x = 100.$$

a) We have written a matlab function *pois2d(x,y,M,N,'condition')* which will generate the sparse matrices associated with discretizing both the pure Neumann and partly Dirichlet problems. The script is such that that discretization can be adjusted, with $\delta x = x/M$ and $\delta y = y/N$, where M and N are integers. Generate the matrices for the case $x = 100$, $y = 1$, $M = 100$ and $N = 20$. Note, when requested to generate the matrix associated with the purely Neumann problem, the script *pois2d* generates a matrix which forces the potential in one corner of the problem domain to be zero. Why is that?

b) The matrices generated in part (a) are symmetric and can be solved with GCR using a truncated back orthogonalization. Implement the truncated back orthogonalization in the *tgcr.m* routine and compare the time required to carry out 100 GCR iterations on the generated matrices with complete and truncated back orthogonalizations (just use your watch to time).

c) Apply your modified GCR algorithm (set the tolerance to 10^{-2}) to solving the discretized pure Neumann and partly Dirichlet problems, using any random vector for b. Plot $\frac{\|r^k\|}{\|r^0\|}$ ($r^k \equiv b - Ax^k$) as a function of k on a log-linear plot for both problems. How would you analyze why the partly Dirichlet problem converges faster?

d) (Optional!) Can you find a good preconditioner for the partly Dirichlet problem?