In this programming exercise, we will develop a preconditioned conjugate gradient (PCG) routine (Algorithm 2 of Class Notes 12) and apply it to the 5 crs files of the previous assignments. We denote \( A \) to be the matrix coming from the CRS file and \( B \) to be the preconditioner. The PCG routine involves only 5 vectors, \( x, r, tr := \tilde{r}, p \) and \( Ap \). The interface to the routine is of the form:

\[
function[x] = PCG(x,tr,nmax,epsilon).
\]

Here \( x \) is the initial iterate and becomes the final iterate after execution. \( tr = \tilde{r}_0 \), \( nmax \) is the maximum number of iterations, \( epsilon := \epsilon \) is an accuracy threshold. Note that the calling routine needs to compute \( tr = b - Ax \) \((x = x_0 \text{ at this time})\).

The routine iterates until either the max number of iterations is achieved or

\[
\eta_i := \left( \frac{(Ap_i, p_i)}{(Ap_0, p_0)} \right)^{1/2} < \epsilon,
\]

which ever comes first.

The flow of the routine should be as follows:

(Start up:) Compute \( r = p = B tr \) and then \( Ap \). Then, \( e0s = (Ap, p) \).

(General step:) (a) Compute \( ((Ap, p)/e0s)^{1/2} \) and check for convergence.

(b) Compute \( aphai \).

(c) Update \( x \).

(d) Update \( tr \).

(e) Update \( r = B tr \).

(f) Compute \( betai \).

(g) Update \( p \).

(h) Compute \( Ap \).

**Problem 1.** Run the 5 problems using the identity as a preconditioner. Specifically, set \( x_0 \) to be the zero vector and \( b \) to be the vector of ones. Use \( \epsilon = .001 \) and report the number of iterations and the value of \( \eta_i \) when the threshold \( \eta_i < \epsilon \) is achieved. You should be able to see the conjugate gradient acceleration, i.e. the number of iterations should grow like the parameter for
the problem (in contrast to a growth like the parameter squared for Jacobi or Gauss Seidel).

**Problem 2.** Run the same 5 problems (same initial iterate, right hand side, and stopping criteria) but using the preconditioner of the last programming assignment. Report the number of iterations needed to make $\eta_i < .001$. Your iteration numbers should be smaller than the case of preconditioned steepest descent but not drastically different.

The above two problems illustrate that the conjugate gradient method gives rise to significant savings when applied to a poorly conditioned problem but only modest savings when applied to a well conditioned system. If your problem is well conditioned, any properly formulated preconditioned iterative method will work quite well.