1. [42 points: 7 points per matrix]

At its \( k \)th step, an iterative method for solving \( Ax = b \) constructs an approximate solution \( x_k \). With this iterate we associate the residual vector

\[ r_k = b - Ax_k. \]

A successful method for solving \( Ax = b \) will drive \( r_k \to 0 \) as quickly as possible.

Lewis Fry Richardson proposed that \( x_k \) be constructed as follows Let \( x_0 = 0 \) so that \( r_0 = b \). Then for a fixed constant \( \gamma \), define

\[ x_{k+1} = x_k + \gamma r_k. \]

(a) Show that you can write \( r_k = (I - \gamma A)^k b \).

(b) Suppose that \( A \) is diagonalizable, \( A = V \Lambda V^{-1} = \sum_{j=1}^{n} \lambda_j v_j v_j^* \).

Show that \( I - \gamma A \) has the same eigenvectors as \( A \). What are the eigenvalues of \( I - \gamma A \)?

(c) Suppose further that all eigenvalues \( \lambda \) of \( A \) satisfy \( 1 \leq \lambda \leq 2 \).

For what real numbers \( \gamma \) will \( r_k \to 0 \)?

What optimal choice of \( \gamma \) will drive \( r_k \to 0 \) most rapidly (knowing only that \( 1 \leq \lambda \leq 2 \))?

For the rest of the problem, consider this \( 500 \times 500 \) matrix \( A \) and accompanying right-hand side:

\[ A = \frac{1}{4} \begin{bmatrix} 6 & 1 & & & \\ 1 & 6 & & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 6 & \\ & & & & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ \vdots \\ 1 \end{bmatrix}, \]

where the unspecified entries of \( A \) are zero. The eigenvalues of this matrix (a modification of one you studied on Problem Set 4) satisfy \( 1 \leq \lambda \leq 2 \).

(d) Produce a semilogy plot showing \( \|r_k\| \) versus \( k \) for the optimal \( \gamma \) you determined in part (c).

(e) The polynomial \( \psi_k(z) = (1 - \gamma z)^k \) should be small on the interval \( 1 \leq z \leq 2 \), and satisfy \( \psi_k(0) = 1 \). Confirm these properties by producing a plot showing \( \phi_k(z) \) versus \( z \) for \( -1 \leq z \leq 3 \) for \( k = 1, 2, 3, 4, 5 \). (Show all five graphs on the same plot.) Use \( \text{axis([-3 3 -.5 1.5])} \) to crop the axes to the most relevant region.

(f) Now replace Richardson’s method with the optimal method, GMRES.

On the same plot you generated in part (d), show \( \|r_k\| \) versus \( k \) for GMRES.

You can generate the \( \|r_k\| \) values by calling MATLAB’s built-in GMRES code:

\[ [x, \text{flag}, \text{relres}, \text{iter}, \text{resvec}] = \text{gmres}(A, b, [], 1e-10, 100); \]

The values of \( \|r_k\| \) (starting with \( \|r_0\| = \|b\| \)) are stored in the vector \( \text{resvec} \).
2. [30 points: (a)=6 points; (b)=12 points; (c)=12 points; (d) = 4 bonus points]

In the last problem, you are dealing with a fairly small matrix. Iterative methods excel when working with larger matrices. To appreciate the challenges of solving large systems, consider this example.

Tim Davis at the University of Florida maintains a large collection of test matrices. We will experiment with one of them, a FEMLAB model of the Navier Stokes equations in a 3 dimensional domain. This model is still relatively small (the matrix has dimension $n = 20,414$); this size is large enough to be interesting, but small enough to conduct experiments in MATLAB. For details about the matrix, see: http://www.cise.ufl.edu/research/sparse/matrices/FEMLAB/ns3Da.html

(a) Download the matrix from ns3Da.mat from this link:
http://www.cise.ufl.edu/research/sparse/mat/FEMLAB/ns3Da.mat
Load this file into MATLAB (load ns3Da).
Extract the matrix $A$ and right-hand $b$: ($A = \text{Problem.A}$; $b = \text{Problem.b}$).
What percentage of the entries in $A$ are nonzero? ($\text{nnz}(A)$ counts the nonzero entries.)

(b) Solve this system using GMRES, as described in part (f) of Problem 1.
Use tolerance $1e-10$, and set the maximum number of iterations to 2000.
(Type help gmres to learn about the arguments the gmres command takes.)
Produce a semilogy plot showing $\|r_k\|$ versus $k$.
Use the tic and toc commands to time how long GMRES takes to run. Note that this command might take a couple minutes to run, and use a bit of memory.

GMRES uses quite a bit of memory to store the basis vectors for the Krylov subspace. An alternative algorithm, restarted GMRES, restricts the subspace dimension. For example, GMRES(10) computes 10 iterations of GMRES to build the iterate $x_{10}$, the best iterate from a 10-dimensional subspace. If $\|b - Ax_{10}\|$ is not sufficiently small, then use $x_{10}$ as an initial guess, and run 10 new steps of GMRES. (We seek to solve $Ax = b$. Let $x = x_{10} + y$. Then solving $Ax = b$ is equivalent to solving $Ay = b - Ax_{10}$. We hope that $\|b - Ax_{10}\|$ is quite a bit smaller than $\|b\|$.) Restarted GMRES is built into MATLAB; for example, to run GMRES($m$) for a maximum of 2000 iterations:

$$[x, \text{flag, relres, iter, resvec}] = \text{gmres}(A, b, 1e-10, 2000);$$

This method must take more iterations than standard GMRES (which is optimal), but it can be faster because, on average, the iterations take less time to execute. We shall explore this idea.

(c) Solve $Ax = b$ using GMRES(10), GMRES(20), GMRES(50), and GMRES(100) using MATLAB’s gmres command.
For each of these methods, plot $\|r_k\|$ versus $k$ on the same plot you generated for part (b).
Use tic and toc to record the run times for each of these algorithms.
Produce a table comparing these times to the run time of standard GMRES in part (b).

(d) (4 point bonus): Replace GMRES with the alternative (suboptimal) methods:

$$[x, \text{flag, relres, iter, resvec}] = \text{bicgstab}(A, b, 1e-10, 2000);$$
$$[x, \text{flag, relres, iter, resvec}] = \text{bicgstabl}(A, b, 1e-10, 2000);$$

Produce a plot showing $\|r_k\|$ for each of these methods, and report their execution times.
3. [28 points: (a)=10; (b)=4; (c)=4; (d)=4; (e)=6 points]

We have seen the matrix exponential \( e^{tA} \) as a tool for solving the dynamical system \( x'(t) = Ax(t) \). This problem explores a different modeling problem that also gives rise to \( e^{tA} \).

Suppose \( A \) is an adjacency matrix for an undirected graph with \( n \) nodes, i.e.,

\[
a_{j,k} = \begin{cases} 
1, & \text{an edge connects node } j \text{ to node } k; \\
0, & \text{otherwise.}
\end{cases}
\]

(This implies that the diagonal of \( A \) is zero.)

In class we say that the number of paths of length \( m \) from node \( j \) to node \( k \) is given by the \((j,k)\) entry of \( A^m \). Inspired by this method of path-counting, Estrada and Rodríguez-Velázquez (2005) proposed a way to gauge the importance of each node as a weighted sum of the number of paths from a given node back to itself. In particular, the subgraph centrality (or Estrada index) of node \( j \) is given by

\[
(e^A)_{j,j}.
\]

The higher this value, the more central a node is regarded. Estrada and Hatano (2007) extend this notion to measure the communicability of nodes \( j \) and \( k \) as

\[
(e^A)_{j,k}.
\]

(This work is surveyed and extended in a paper by Estrada and D. J. Higham, SIAM Review, 2010.)

We now wish to apply these ideas to real data. Analyzing a variety of media sources, Valdis Krebs has produced a graph of the 9/11 terrorist network, including both the hijackers and their accomplices; see http://www.orgnet.com/hijackers.html for details. We have written the routine \texttt{sept11.m}, which creates an adjacency matrix for Krebs’s graph, given an arbitrary node numbering. The graph contains 69 nodes, 19 of which correspond to the hijackers. (These nodes are identified at the top of \texttt{sept11.m}, with the pilots singled out.)

(a) Compute the subgraph centrality for all the nodes, and present a table showing the results for the top thirty nodes. Each row of this table should correspond to a node, and include:

i. subgraph centrality rank (in order, 1, \ldots, 30);

ii. node number, \( j \);

iii. subgraph centrality value, \((e^A)_{j,j}\);

iv. number of edges connected to node \( j \);

(b) In your table, clearly identify the four pilots (nodes 36, 44, 50, 65; see \texttt{sept11.m} for details).

(c) We might alternatively have ranked the nodes by their edge counts, rather than subgraph centrality. Does your table reveal any anomalous nodes that are ranked more highly than their edge counts would suggest, or nodes with large edge counts that have relatively low centrality scores?

(d) Now use \texttt{imagesc} in MATLAB (or similar) to visualize the magnitude of the entries in \( e^A \). This allows you to visually inspect the communicability of all nodes with one another at once. Designate the rows and columns corresponding to the hijackers of each plane. You may do this by hand, or using some slick command along the lines of, e.g.,

\[
\texttt{fill([0 70 70 0],[31.5 31.5 36.5 36.5],'v','facealpha',.3,'edgealpha',.3)}
\]

(e) Interpret your plot in part (d). Were the different crews of hijackers highly connected with one another? Do you notice anything about the crew of United Flight 93, which crashed in Pennsylvania as a result of passenger intervention?