Math 224: Scientific Computing I

Homework 7.1 **Assigned: Friday, Oct 22, 2004** **Due: Friday, Nov 5, 2004**

Eigenvalue Problems

0. Reading: Atkinson 9.1, 9.2, 9.6, Numerical Recipes 11.7, Trangenstein 7.7.

1. [X-matrix^2, the sequel] Let M be a $n \times n$ matrix (n must be an **EVEN** integer), whose entries are all zeroes, except for:

$$m_{i,i} = 4i \qquad m_{i,n-i+1} = 2i, \qquad i = 1, 2, \cdots, n$$

If you enter "`format +`" in `matlab` or `octave` you will see the structure of this matrix is an "X".
Let $n = 6$. (But write all of your work so it could be generalized to any value.)
We will consider several questions related to finding the eigenvalues of M:

(a) Based on the $||M||_1$, $||M||_\infty$ matrix norms and the Gershgorin circle theorem, give estimates for what is known about the eigenvalues of this matrix.

(b) Lets return to `randvec()` and Monte Carlo methods from Homework 5. Use $k_{\max} = 100,000$ Monte Carlo trials of `randvec()` to find the max and min of the Rayleigh quotient,

$$R(\vec{\mathbf{x}}) = \frac{\vec{\mathbf{x}} \cdot M\vec{\mathbf{x}}}{\vec{\mathbf{x}} \cdot \vec{\mathbf{x}}},$$

similar to problem 3 on Homework 5.

(c) Monte Carlo version 2.0: Lets create a better set of random direction vectors. The random vectors could be "better" if: (i) they really were unit direction vectors, so we don't need to worry about their lengths, and (ii) if their directions were really uniformly distributed. We can use hyper-spherical polar coordinates to create uniformly distributed directions vectors in $\mathbb{R}^n$:

- Create $n-1$ random angles $\theta_i \in [0, 2\pi)$ for $i = 1, 2, \cdots, n-1$ (create them one at a time).[1]
- Write a routine to construct the random vector based on the following schematic (for $\vec{\mathbf{x}} \in \mathbb{R}^5$)

$$\begin{aligned}
x_1 &= \cos\theta_1 \\
x_2 &= \sin\theta_1 \cdot \cos\theta_2 \\
x_3 &= \sin\theta_1 \cdot \sin\theta_2 \cdot \cos\theta_3 \\
x_4 &= \sin\theta_1 \cdot \sin\theta_2 \cdot \sin\theta_3 \cdot \cos\theta_4 \\
x_5 &= \sin\theta_1 \cdot \sin\theta_2 \cdot \sin\theta_3 \cdot \sin\theta_4
\end{aligned}$$

Hopefully, you see the pattern. Notice that $||\vec{\mathbf{x}}||_2^2 = 1$.

Use this approach to write `randvec2()` and use it to repeat part (a).
Which choice of random vectors gives better results?

(d) Implement the Power method to find the largest eigenvalue of M, $\lambda_{\max} = \max |\lambda(M)|$. How many iterations does it take to converge to an answer?

(e) Implement the Inverse Power method (inverse iteration) to find the smallest eigenvalue of M, $\lambda_{\min} = \min |\lambda(M)|$. How many iterations does it take to converge to an answer?

(f) Hopefully, you have noticed that there is a contradiction between your results in the previous parts. Explain what the apparent conflict is in the λ values from (b,c) and (d,e).

[1] `M_PI` is `math.h`'s definition of π.

This is not a real conflict – only a limitation of the Rayleigh quotient approach. Break up the matrix M into symmetric and anti-symmetric parts,

$$M = S + A, \qquad S = \frac{1}{2}(M + M^T), \qquad A = \frac{1}{2}(M - M^T)$$

How are the Rayleigh quotients for these different matrices, R_M, R_S, R_A related?[2]

Repeat parts (d,e) for matrix S to explain parts (b,c).

(g) Write a routine to implement a stabilized version of the optimal inverse-iteration, shifted Rayleigh quotient method for finding the eigenvalue of a matrix M that is closest to a given value μ:

```
double eigen(double **M, int n,double *x,double mu);
```

That is, iterate the scheme described in class for $k = 1, 2, \cdots, 30$, but reset the value of λ_k to the original initial guess $\lambda_k =$ `mu` if $k < 20$. Note: you will need your `LUfactor()`, `LUsolve()`, and `productAxb()` routines and you will need to allocate memory for another matrix – DO NOT overwrite the original matrix, you will need it! Have your routine return the value of the eigenvalue for this function, and fill the entries of `x[]` with the eigenvector.

(h) Use (a,g) to find all of the eigenvalues of M to AT LEAST 6 digits of accuracy. You will need good initial guesses for `mu` in order to converge.[3]

2. [Banded matrices, the eigenvalues] Similar to 1(g), for banded matrices, write

```
double bandeigen(double **B,int n,int m1,int m2,double *x,double mu);
```

(a) [The warm-up] Solve the system $M\vec{\mathbf{y}} = \lambda\vec{\mathbf{y}}$ with

$$m_{i,i} = \frac{2}{\Delta x^2} \qquad i = 1, 2, \cdots, n$$

$$m_{i,i+1} = m_{i+1,i} = -\frac{1}{\Delta x^2} \qquad i = 1, 2, \cdots, n-1$$

and else $m_{i,j} = 0$ with

$$\Delta x = \frac{1}{n+1}$$

For $n = 4999$, find the value of the THREE SMALLEST λ (to at least 6 digits of accuracy).
Plot y_i vs $x_i = i\Delta x$ for each.
Hint: This is a numerical approximation of the ODE problem

$$-y'' = \lambda y \qquad y(0) = y(1) = 0, \qquad y_k(x) = \sin(k\pi x), \qquad \lambda_k = k^2\pi^2$$

(b) [Bessel's equation] Similarly, solve the system $M\vec{\mathbf{y}} = \lambda\vec{\mathbf{y}}$ with

$$m_{i,i} = \frac{2}{\Delta x^2} \qquad i = 1, 2, \cdots, n$$

$$m_{i,i+1} = -\frac{1}{\Delta x^2}\left(\frac{1 + [i + \frac{1}{2}]\Delta x}{1 + i\Delta x}\right) \qquad m_{i+1,i} = -\frac{1}{\Delta x^2}\left(\frac{1 + [i + \frac{1}{2}]\Delta x}{1 + [i+1]\Delta x}\right) \qquad i = 1, 2, \cdots, n-1$$

and else $m_{i,j} = 0$ with

$$\Delta x = \frac{1}{n+1}$$

For $n = 4999$, find the value of the THREE SMALLEST λ (to at least 6 digits of accuracy).
Plot y_i vs $x_i = 1 + i\Delta x$ for each.
Hint: This is a numerical approximation of a problem for Bessel's equation

$$-(xy')' = \lambda xy \qquad y(1) = y(2) = 0$$

[2] Hint: Think about the transposes of products of matrices.
[3] You can check you answers with octave or matlab.