PROJECT 9

Math 441

Due: Thursday, December 11

All "by hand" calculations require that you SHOW YOUR WORK. All MATLAB code and output must be turned in.

1. **Sensitivity of the Eigenvalue Problem:** Download **A10x10.mat** from the course web site, then in MATLAB type **load A10x10** and **whos**. You will see a 10×10 non-symmetric matrix A in memory. Type **imagesc(A); colorbar** to see what A "looks like."

   (a) Calculate the eigenpairs of A by **[Wtrue eigtrue]=eig(A)**. Put these eigenvalues into a 10×1 vector in increasing order, **eigtrue = sort(diag(eigtrue))**.

   (b) Suppose that the matrix A is perturbed by δA such that $||\delta A||_2 = 10^{-8}$, and $\hat{\lambda}$ is an eigenvalue of the new matrix $\hat{A} = A + \delta A$. Give the theoretical upper bound on how much $\hat{\lambda}$ can differ from λ_{true}, an eigenvalue of A. Justify your answer.

   (c) Run your QR algorithm, from project 8, at least 100 iterations to estimate all of the eigenvalues of A, and call the estimates you get $\hat{\lambda}$. In MATLAB, put these estmates in a 10×1 vector **eighat**. Now sort them in increasing order by typing **eighat = sort(eighat)**. Recall that the matrix in the QR algorithm will converge to an upper triangular matrix T, and that the estimates of the eigenvalues of A will be on the diagonal. Turn in your code and print out of the matrix T.

   (d) Print out **[eighat eigtrue]** and calculate the ∞-norm of the difference **eighat-eigtrue**. Use this result and the Bauer-Fike bound to estimate the size of the perturbation $||\delta A||_2$ introduced by your algorithm.

2. **Lanczos Eigensolving:** You will use the Lanczos algorithm to estimate a handful of the eigenvalues of the 100×100 matrix A that you can download from the course web site at **A100x100.mat** and then enter **load A100x100**. When you type **whos**, you'll see the matrix A in memory. Type **imagesc(A); colorbar** to see what the symmetric matrix A "looks like."

   (a) Download **Lanczos.m** from the course web site and run the code ONLY 12 iterations on the matrix A. You do not need to modify the code (ie look at the input parameters). Give a printout of the resulting 12×12 tridiagonal matrix T.

   (b) For this problem, use MATLAB to verify that $T \approx Q^T AQ$ and to confirm that Q is orthogonal.

   (c) Calculate the eigenvalues $\lambda_1, ..., \lambda_{100}$ eigenvalues of A and plot them using **semilogy(eig(A),'.')**. Turn in this plot.

   (d) Calculate $\theta_1, ..., \theta_{12}$, the eigenvalues of T and compare them to $\lambda_1, ..., \lambda_{100}$. Which eigenvalues of A are being estimated by the θ_i? Explain why these λ_i being estimated, and not some other ones (see #2c).

   (e) Now run Lanczos on A for 15 iterations, and check out the eigenvalues of this new 15×15 tridiagonal T. Which eigenvalues of A are being estimated? What are the three new eigenvalues (in addition to the 12×12 case) estimating?

3. Give two scenarios when you would want to use the Lanczos Eigensolver (**eigs**) instead of the QR algorithm (**eig**) to estimate the eigenvalues of a matrix.

4. **Natural Logs:** Consider estimating $\int_1^R \frac{1}{x} dx$.

   (a) Let $R = 2$ and consider 3 nodes. Give the values of the nodes and the weights for each of the following numerical integration methods: Riemann sums, the Trapezoid Rule, and Simpsons Rule.

   (b) The nodes and weights for Gaussian Quadrature are computed from a particular Lanczos tridiagonal matrix. For $R = 2$ and 3 nodes, give this matrix.

   (c) How are the weights and nodes calculated from the Lanczos matrix? Give these values for 3 nodes.

   (d) Create an m-file which evaluates $\frac{1}{x}$ for any given x. Turn in this code.

   (e) For $R = 2$ and 101 nodes, use the numerical integration program **numint.m** from the class web site to estimate the integral $\int_1^R \frac{1}{x} dx$. Run **numint.m** 4 times, once using Riemann sums, then the Trapezoid Rule, then Simpsons, and finally using Gaussian Quadrature.

   (f) For each run, consider the absolute error **abs(estimate - log(2))**. Which of the 4 methods is most accurate?

   (g) Keep track of the cpu time for each of the 4 methods. Which is least costly? Which is most costly?

5. **Square Roots:** (known by Heron ca. 250-150BC, about 200 years after Pythagorus) Use Newton's Method to find the zeros of $f(\beta) = \beta^2 - R$.

   (a) You will need to create an m-file which returns $f(\beta) = \beta^2 - R$ and $f'(\beta)$ for any given β. Turn in this code.

   (b) By hand, explicitly write out the Newton step $\beta_{k+1} = ...$ (of course, newton.m will calculate this for you automatically for each k).

   (c) For $R = 2$, run **newton.m** to estimate the root $\hat{\beta}$. Turn in all output, including the graph in figure 1. After how many iterations does Newton's method converge?

   (d) If you want the positive square root, you must set $\beta_0 > 0$. Explain why this is true.

   (e) Why does Newton's Method fail if you set $\beta_0 = 0$?

   (f) Calculate the absolute error $|\hat{\beta} - \sqrt{2}|$. Is Newton's Method comparable to MATLAB's **sqrt**?

6. **Optimization:** Use Newton's method to minimize the scalar function $f\begin{pmatrix} \beta_1 \\ \beta_2 \end{pmatrix} = \beta_1 \ln(\beta_2)$.

   (a) By hand, calculate the 2×1 gradient $\nabla_\beta f$ and the 2×2 Hessian matrix of second derivatives $d^2_\beta f$.

   (b) You will need to create an m-file which returns the scalar f, the gradient $\nabla_\beta f$ and the Hessian matrix of second derivatives $d^2_\beta f$ for a given value of β. Turn in this code (see **rosen.m** and **NewtonForRosenbrock.m**, available at the class web site).

   (c) By hand, explicitly write out the Newton step $\beta_{k+1} = ...$

   (d) Modify **NewtonForRosenbrock.m** to optimize f using Newton's Method. Initialize at $\beta_0 = \begin{pmatrix} 1.5 \\ 1.5 \end{pmatrix}$. Turn in your code.

   (e) Run your code to get $\hat{\beta}$, an estimate of the minimizer of $f(\beta)$. Turn in ALL output to your code, including the graphs in figures 2 and 3.

   (f) Check whether $\hat{\beta}$ is a minimizer of $f(\beta)$ by doing the second derivative test.

   (g) Rerun your code, this time initializing at $\beta_0 = \begin{pmatrix} 1.5 \\ 5 \end{pmatrix}$. Explain why Newton's Method fails to converge in this case.

7. **Non-linear least squares:** Use the Gauss-Newton Method to fit a logistic model to the number of cases of AIDS in the US from 1981 to 1992. The data is available from the course web site, where the first column of the data matrix is in years, $t = 1$ corresponds to 1981, and $t = 12$ corresponds to 1992. The second column is the number of new AIDS cases each year, in the thousands. The least squares problem you will solve is $\min_\beta f(\beta) = \min_\beta ||y - \phi(\beta)||_2^2$ where

$$\phi(\beta) = \frac{\beta_1 \beta_2}{\beta_1 + (\beta_2 - \beta_1)e^{-\beta_3 t}}.$$

Note that β_3 is the exponential rate of change of f from β_1 to β_2.

   (a) Show that β_1 is the predicted number of AIDS cases in 1980, and that β_2 is the asymptotic number of cases as $t \to \infty$.

   (b) You will need to write an m-file which accepts a 3×1 vector of β parameters and a 12×1 vector of times t, and outputs the value of $\phi(\beta, t)$ (see **GNexpfunc.m** at the course web site for an example). Turn your code in.

   (c) By hand, explicitly write out the Gauss-Newton step $\beta_{k+1} = ...$

   (d) Modify **GaussNewtonExample.m**, available at the course web site, then run it to find $\hat{\beta}$, the least squares solution to f. Initialize at $\beta_0 = \begin{pmatrix} .26 \\ 47.6 \\ 1 \end{pmatrix}$. Turn in your code.

   (e) Turn in all output to your code, including the figure. Also, write out the function $\phi(\hat{\beta}) = \frac{\hat{\beta}_1 \hat{\beta}_2}{\hat{\beta}_1 + (\hat{\beta}_2 - \hat{\beta}_1)e^{-\hat{\beta}_3 t}}$ where $\hat{\beta}$ is the least squares solution.

   (f) Does the model predict that the number of cases of AIDS in 1980 was more than 100? Do the statistics (the standard error and confidence interval) support your answer? Why or why not?

   (g) Does the model predict that the number of cases of AIDS in 1993 was less than 100,000? Do the statistics (the standard error and confidence interval) support your answer? Why or why not?

   (h) In fact, there were 106,618 cases of AIDS in 1993. Does this mean that the model you have fit is a bad model? Explain.