CS/MATH 375, Fall 2024 — HOMEWORK # 11
Due : Nov. 15th at 10:00pm on Canvas

Instructions

- **Report:** In general, your report needs to read coherently. That is, start off by answering question 1. Fully answer the question, and provide all the information needed to understand your answer. If Matlab code or output is part of the question, include that code or output (e.g., screenshot) alongside your narrative answer. If discussion is required for a question, include that. Overall, your report is your narrative explanation of what was done, your answers to the specific questions, and how you arrived at your answers. *Your report should include your Matlab scripts, code output, and any figures.*
- **What to hand in:** Submission must be one **single PDF** document, containing your entire report, submitted on Canvas.
- **Partners:** You are allowed to (even encouraged) to **work in pairs**. If you work with a partner, only one member of the group should need to submit a report. On Ca nvas, both partners should join a group (numbered 1 through 15). Then either member can upload the report for the entire group. Groups of more than 2 students are not allowed.
- **Typesetting:** If you write your answers by hand, then make sure that your handwriting is readable. Otherwise, I cannot grade it.
- **Plots:** All plots/figures in the report must be generated in Matlab or Python and not hand drawn (unless otherwise specified in the homework question).

  In general, make sure to (1) title figures, (2) label both axes, (3) make the curves nice and thick to be easily readable, and (4) include a legend for the plotted data sets. The font-size of all text in your figures must be large and easily readable.

---

1. Since we spent a week or two learning to solve nonlinear equations, one might assume that finding the characteristic polynomial of a matrix and then using one of our root-finding methods would be a good approach for finding the eigenvalues. In this problem, we illustrate one reason why this is generally a bad idea.

   Assume that
   $$A = \begin{bmatrix} 1 & \epsilon \\ \epsilon & 1 \end{bmatrix}.$$

   (a) What is the characteristic polynomial of A?

   (b) Calculate the eigenvalues and eigenvectors of A (by hand).

(c) Now let $\epsilon = \sqrt{\epsilon_m}/4$, where ϵ_m is machine epsilon. Verify that $\epsilon >> \epsilon_m$. Use the Matlab command for finding the coefficients of the characteristic polynomial, `charpoly`, and the Matlab command for finding the roots of a polynomial, `roots`, to find the eigenvalues. Report the eigenvalues found this way.

For help with using `charpoly` and `roots`, type `help charpoly` and `help roots` at the Matlab prompt. The function `charpoly` returns the coefficients of the characteristic polynomial in the correct order for `roots`, i.e., it returns $[a_n, a_{n-1}, \ldots, a_0]$, where a_i is the coefficient for λ^i in the characteristic polynomial.

(d) Do your answers from parts a and b disagree? If so, explain why (use `format long` so that you are able to see by how much they disagree).

(e) Write a Matlab function `[eval, evec] = power_method(A, x, tol)` which takes in a matrix `A`, initial vector for power iteration `x`, and a tolerance `tol`, and returns an approximation to the largest eigenvalue `eval` and the corresponding normalized eigenvector `evec` using the normalized power method. Use the following criterion for the convergence of power method iteration:

$$|\lambda^{(k)} - \lambda^{(k-1)}| < \texttt{tol}$$

$$\left\| x^{(k)} - x^{(k-1)} \right\|_2 < \texttt{tol},$$

where $x^{(k)}$ is the approximation to the eigenvector after the kth iteration. where $\lambda^{(k)}$ is the approximation to the eigenvalue after the kth iteration.

(f) Keeping $\epsilon = \sqrt{\epsilon_m}/4$ use your function `power_method` and the initial guess `x=[3;4]` to find the largest eigenvalue of A. Report your approximate eigenvalue. Try doing this with `tol` $= 10^{-8}, 10^{-9}, 10^{-10}$. Depending on the tolerance, running your script could take a few minutes (not for credit: what does the ratio of the true eigenvalues have to do with it?).

2. So far we've only discussed using the power method to find the *largest* eigenvalue of a matrix. In this problem, we will see how to find the *next largest* eigenvalue.

(a) Assume the a symmetric positive definite matrix A has distinct eigenvalues ($\lambda_i \neq \lambda_j$). Let λ_1 denote the largest eigenvalue and $\vec{v}_1$ be an associated eigenvector. Show that

$$B = A - \frac{\lambda_1}{(\vec{v}_1)^T \vec{v}_1} \vec{v}_1 (\vec{v}_1)^T$$

has the same eigenvalues and eigenvectors of A, except that λ_1 is replaced by 0, i.e., $B\vec{v}_1 = 0$.

Hint: It can be shown that the eigenvectors of A are orthogonal, i.e.,

$$(\vec{v}_i)^T \vec{v}_j = 0$$

when $i \neq j$. You can assume this fact, but make sure to explain how you are using it.

Hint 2: Be careful with order of vector products. The term $(\vec{v}_1)^T \vec{v}_1$ is just a dot product and produces a scalar. However $\vec{v}_1 (\vec{v}_1)^T$ is an outer product and produces a matrix.

(b) Use the previous formula to modify your function `power_method` to find the second largest eigenvalue of the A from the previous problem,

$$A = \begin{bmatrix} 1 & \epsilon \\ \epsilon & 1 \end{bmatrix}.$$

Note, the largest eigenvalue of B will be the second largest eigenvalue of A.

3. Numeric differentiation

(a) Use a Taylor series expansion to show that

$$\frac{f(a+h) - f(a-h)}{2h} = f'(a) + \mathcal{O}(h^2).$$

(b) Let $f(x) = \cos(1.5x)$ and $a = 1$. Plot (using the appropriate scaling, i.e. choose the best option between `plot`, `semilogy` or `loglog`), the absolute error in the approximation of $f'(a)$ by the finite difference in part (a), for `h=logspace(-1,-16,16)`. Explain your results.

(c) The error in (b) arises from two sources of error: truncation error due to the finite Taylor series approximation and rounding errors due to floating point operations.

i. Show that the truncation error is bounded by $C_1 h^2$ for some constant C_1.

ii. Show that the rounding error is bounded by $C_2 \epsilon / h$, where ϵ is the machine precision or machine epsilon and C_2 is some constant.

iii. The total error is the sum of the two errors. Add them and write out the expression for the error in your finite difference approximation.

(d) Use calculus to show that the optimal h to use in the approximation in (b) (i.e. the one that gives the smallest error) is $h_{\text{opt}} \approx C\epsilon^{1/3}$, where C is some constant. (Hint: Minimize the expression for the total error in (c)-iii.)

(e) Does this result agree with the data you plotted in (b)? Plot a line at $\epsilon^{1/3}$ by using the Matlab command:

```
line([power(eps,1/3) power(eps,1/3)],[get(gca,'ylim')])
```