EGR 141: Vectors and Matrices

Summary: The goal of this lab is to use vectors and matrices. You should not use any MATLAB commands or concepts that are discussed in future chapters and sections.

- Inside your script, solve each of the given problems. In between each problem, type *pause;* Clearly indicate where the code for each problem begins by using a comment block. Start each new problem with a *clear .*
- If my example output "lines up nicely" then your output should as well.
- All output statements should output variables, not pre-computed constants. For example, if I ask you to output $r/2$ when $r = 3$, then you should set r to be three then output as *fprintf(' r/2 = %f ',r/2);* and not *fprintf('r/2 =1.5')* or *fprintf(r/2 = %f',3/2)*.
- For each problem, the script file should be called something appropriate, such as *Lab2_1_yourName.m*
- Note that example output for each problem is not necessarily correct output (I intentionally change numbers so my answers will not always match your answers).

1. For each of the following

$$\text{product: } \frac{4}{3}\cdot\frac{9}{8}\cdot\frac{16}{15}\cdot\frac{25}{24}\cdot\frac{36}{35}\cdots$$
$$\text{sum:} 1+\frac{1}{3}+\frac{1}{6}+\frac{1}{10}+\frac{1}{15}+\frac{1}{21}+\cdots$$

   find a pattern for the numerator and a corresponding pattern for the denominator, each for a generic number of terms, n. Set and output each variable:

   (a) $sum100$ is given sum with $n = 100$ terms
   (b) $sum1000$ is given sum with $n = 1000$ terms
   (c) $prod100$ is the given product with $n = 100$ terms
   (d) $prod1000$ is the given product with $n = 1000$ terms

   Note that none of the above should be hard-coded (you should use vectors, not type term by term).

```
Lab 2 - Product & Sum
Using 100 terms
  1.5765 = 1/2+1/4+1/7+1/11+...
  0.0099 = 1/2*2/3*3/4*4/5*5/6+...
Using 1000 terms
  1.5943 = 1/2+1/4+1/7+1/11+...
  0.0010 = 1/2*2/3*3/4*4/5*5/6+...
```

2. The differential equation $y'' = xe^x$, $y(0) = 0$, $y(1) = 2$ can be approximated on an interval $x \in [0, 1]$ by first creating a uniform grid of x values

$$x_0 = 0,\ x_1 = \frac{1}{N},\ x_2 = \frac{2}{N}, \ldots, x_{N-1} = \frac{(N-1)}{N},\ x_N = 1.$$

   We then define the distance between the $N+1$ grid points as $\Delta x = \frac{1}{N}$.Then, using an approximation to the 2nd derivative:

$$y''(x_k) \approx \frac{y_{k+1} - 2y_k + y_{k-1}}{\Delta x^2} = \frac{y_{k+1} - 2y_k + y_{k-1}}{\Delta x^2}$$

we want

$$y''(x_k) = x_k e^{x_k} \rightarrow \frac{y_{k+1} - 2y_k + y_{k-1}}{\Delta x^2} = x_k e^{x_k}.$$

Moving Δx^2 to the right, we have

$$y_{k+1} - 2y_k + y_{k-1} = \Delta x^2 x_k e^{x_k}$$

Putting as a matrix and incorporating boundary conditions, we write:

$$\begin{bmatrix} -2 & 1 & 0 & \dots & 0 \\ 1 & -2 & 1 & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & \\ 0 & \dots & 1 & -2 & 1 \\ 0 & \dots & & 1 & -2 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{N-2} \\ y_{N-1} \end{bmatrix} + \begin{bmatrix} y_0 \\ 0 \\ \vdots \\ 0 \\ y_N \end{bmatrix} = \begin{bmatrix} x_1 e^{x_1} \Delta x^2 \\ x_2 e^{x_2} \Delta x^2 \\ \vdots \\ x_{N-2} e^{x_{N-2}} \Delta x^2 \\ x_{N-1} e^{x_{N-1}} \Delta x^2 \end{bmatrix}.$$

Moving the boundary conditions to the right hand side, one finds:

$$\begin{bmatrix} -2 & 1 & 0 & \dots & 0 \\ 1 & -2 & 1 & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & \\ 0 & \dots & 1 & -2 & 1 \\ 0 & \dots & & 1 & -2 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{N-2} \\ y_{N-1} \end{bmatrix} = \begin{bmatrix} x_1 e^{x_1} \Delta x^2 - y_0 \\ x_2 e^{x_2} \Delta x^2 \\ \vdots \\ x_{N-2} e^{x_{N-2}} \Delta x^2 \\ x_{N-1} e^{x_{N-1}} \Delta x^2 - y_N \end{bmatrix}.$$

which can be written as $A\boldsymbol{y} = \boldsymbol{b}$. Here, A is an $N-1 \times N-1$ "tri-diagonal form" with a -2 on the diagonal and a 1 on the upper and lower diagonals:

$$A = \begin{bmatrix} -2 & 1 & 0 & \dots & 0 \\ 1 & -2 & 1 & \dots & 0 \\ 0 & \ddots & \ddots & \ddots & \\ 0 & \dots & 1 & -2 & 1 \\ 0 & \dots & & 1 & -2 \end{bmatrix}$$

and $\boldsymbol{b}$ is given by

$$\mathbf{b} = \begin{bmatrix} x_1 e^{x_1} \Delta x^2 - y_0 \\ x_2 e^{x_2} \Delta x^2 \\ \vdots \\ x_{N-2} e^{x_{N-2}} \Delta x^2 \\ x_{N-1} e^{x_{N-1}} \Delta x^2 - y_N \end{bmatrix} = \begin{bmatrix} x_1 e^{x_1} \Delta x^2 - 0 \\ x_2 e^{x_2} \Delta x^2 \\ \vdots \\ x_{N-2} e^{x_{N-2}} \Delta x^2 \\ x_{N-1} e^{x_{N-1}} \Delta x^2 - 2 \end{bmatrix}.$$

Your goal is to solve the resulting linear algebra equation $A\boldsymbol{y} = \boldsymbol{b}$ by

(a) Setting N to be 1000

(b) Setting x to be a linearly spaced vector of $N-1$ points between 0 and 1 NOT including 0 or 1.

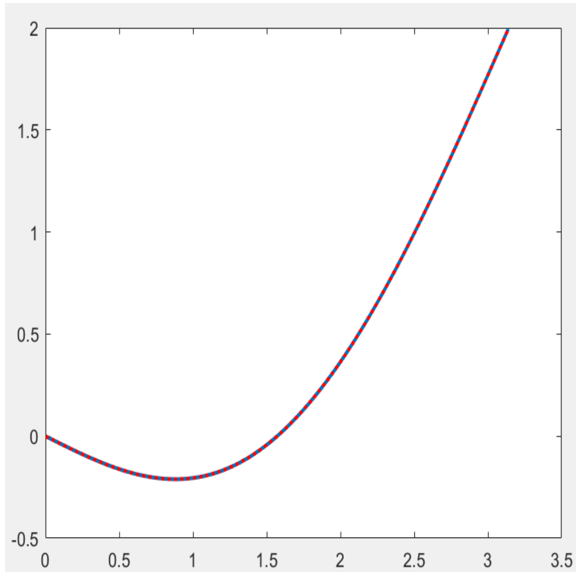
(c) Setting the $N-1 \times N-1$ matrix A, using the *diags* function.

(d) Setting the $\boldsymbol{b}$ vector, as defined above.

(e) Solving $A\boldsymbol{y} = \boldsymbol{b}$.

(f) Check your answer by using the following MATLAB command (yes, it is skipping ahead a bit), where x and y are assumed to be as defined above:
*plot(x,x*exp(1)-2*exp(x)+x.*exp(x)+2,x,y,'r:','LineWidth',2);*
Note that $y = xe - 2e^x + xe^x + 2$ is the analytical solution to the differential equation, and your solution should line up almost perfectly on top.

```
Lab 2 - An ODE
  Solution to y''=sin(x), y(0)=0,y(pi)=2 is
  y = 2*x/pi-sin(x)
```



3. My grading scheme in a mathematics class works as follows:

   - Each homework assignment is out of 10 points. The lowest homework grade is dropped. The homework average is 25% of the final grade
   - The final exam is worth 100 points. It is worth 25% of the final grade.
   - Each regular (not final) exam is worth 100 points. The exam average is 50% of the final grade. If a user receives a higher score on their final exam than their lowest regular exam, the low regular exam grade is replaced with their final exam grade (if the regular exams are a 74, 64, 88 and the final exam is an 80, then we replace the 64 with an 80 and their regular exams average is (74+80+88)/3).

   Given the following, compute the final average in my class. Output only the student's grade to the screen:

HW:	4	8	1	5	1	6
	2	3	4	2	10	9
Exams:	92	65	70	80		
Final Exam:	89					

   All calculations should be done via MATLAB. Do not manually drop the lowest HW, manually replace the Exam grade, etc. Input ALL values, store them in vectors, and compute the average using vector operations.

   ```
   Lab 2 - Math Grades
    Your class average is 85.58%
   ```

4. Recall that, to solve a matrix problem $A\boldsymbol{x} = \boldsymbol{b}$ we traditionally apply an inverse on each side

   $$\boldsymbol{x} = A^{-1}\boldsymbol{b}.$$

   From precalculus, we learned that an inverse exists under only certain circumstances, the most basic of which requires A to be a square matrix.

   We would like to fit a line to the following data, which is clearly non linear.

x	*0*	*1*	*2*	*3*	*4*	*6*	*8*
y	*3.50*	*1.25*	*-2.25*	*-1.75*	*-2.50*	*-6.50*	*-9.00*

Thus, we want to find c and d so that

$$cx_i + d = y_i$$

for each data point (x_i, y_i) (which is, of course, not possible).

(a) Turn our problem (finding c and d which fits the given data) into a matrix problem of the form

$$A \begin{bmatrix} c \\ d \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \boldsymbol{y}$$

where A is an $n \times 2$ matrix (up to you to figure out what is in the matrix A).

(b) This matrix does not have a single solution. Instead, we multiply both sides of our equation by A^T, setting up a square-matrix problem (often called the *normal* equation)

$$A^T A \begin{bmatrix} c \\ d \end{bmatrix} = A^T \boldsymbol{y}.$$

(c) Solve this normal equation *without explicitly computing the inverse of* $A^T A$, outputting the equation of the line you find ($y = cx + d$) to the terminal.

```
Lab 2 - Linear Regression
 Line of best fit is y = -3.85x + 11.67
```