Math 401 - Groupwork #10
Matrix Exponentials and Rotations

Instructions: Work through the problems below in order. Write down all of your answers on paper, to be turned in at the end of class. I do not want you to turn in any MATLAB code. Make sure that the name of every student in the group is on the paper.

Here are some MATLAB remarks:

- The command to compute the matrix exponential e^A is `expm(A)`.
- DO NOT accidentally use the command `exp(A)`. This command DOES NOT give the matrix exponential e^A. Instead, it just exponentiates each entry of the matrix (this is very different from a matrix exponential.)
- If you declare a symbolic variable `syms t`, MATLAB can usually handle evaluating `expm(t*A)` and presenting it as a function of `t`.
- Sometimes, MATLAB tries too hard to give exact symbolic expressions and it becomes unreadable. Sometimes it is best to use `vpa(expm(t*A),4)`, which gives decimal approximations.
- In the past, you (or I) may have used an aprostrophe `A'` to compute a transpose. It may be better to use `transpose(A)` instead. The apostrophe command computes the *conjugate transpose* (takes the transpose and complex conjugates everything.) There's no difference if everything is real, but you may want to be careful if there are complex numbers (or symbolic variables) present.

1. Consider the matrix $A = \begin{bmatrix} 1 & -2 \\ 1 & 3 \end{bmatrix}$.

   (a) Enter a symbolic variable `t` into MATLAB and execute the command `expm(t*A)`. Note that MATLAB presents the answer using complex exponentials. This is not desirable because e^{tA} is a real matrix. Diagonalize $A = PDP^{-1}$ in MATLAB, and then compute e^{tD} by hand. Simplify e^{tD} using Euler's formula so that there are no complex exponents. (Note: $e^{a+bi} = e^a e^{bi}$, then you can use Euler's formula on e^{bi}.)

   (b) Enter your e^{tD} manually into MATLAB and use it to compute e^{tA}. There should not be any i's in the result.

   (c) The reason why MATLAB is a bit stubborn about complex exponentials in this case is related to the fact that it is making no assumptions on what kind of number the symbolic variable `t` is (e.g real or complex). We actually get much better results if we tell MATLAB that we want to assume that `t` is a real number. This can be done with the command `assume(t,'real')` After doing this, try executing `expm(t*A)` again. Write down the result (it should be the same as above.)

2. (a) Use MATLAB to give the solution $\mathbf{x}(t)$ to

$$\mathbf{x}'(t) = \begin{bmatrix} 1 & -1 \\ 0.4 & 0.6 \end{bmatrix} \mathbf{x}(t), \qquad \mathbf{x}(0) = \begin{bmatrix} 1 \\ 2 \end{bmatrix}.$$

   As in the previous part, set `t` as a real variable, so that there are no complex numbers in your solution.

   (b) Describe what happens to your solution $\mathbf{x}(t)$ as t goes to infinity.

3. Use matrix exponentials in MATLAB to give the rotation matrix which rotates by θ radians about the line through the origin which goes in the direction of the vector $\mathbf{v} = \begin{bmatrix} 1 \\ 2 \\ -2 \end{bmatrix}$.

Download the script `rotate_cube.m` from ELMS. The script contains the cube (from the previous 3-d computer graphics groupwork). In the script, you can enter a 3×3 matrix A. The script applies e^{tA} to the cube for values of t between 0 and 1 and applies perspective projection. The script creates an animation as t goes from 0 to 1, so that the end of the script is e^A applied to the cube. This allows us to visualize what the matrix exponential e^A does dynamically.

4. The script is initially set to rotate about the z-axis.
   (a) By what angle does it rotate?
   (b) Change it so that it rotates around by $9\pi/4$ radians, so it goes around one and an eighth times. (Only edit A, don't edit the loop in the script.)

5. Experiment with the script by entering the following matrices in place of A. For each one, write down the eigenvalues of A and save an image of the final frame of the animation as `exp-5x.bmp` where `x` is the part you did.

   (a) $A = \begin{bmatrix} 0 & 1 & .5 \\ 0 & 0 & .75 \\ 0 & 0 & 0 \end{bmatrix}$ (shear)

   (b) $A = \begin{bmatrix} -2 & 1 & 1 \\ 1 & -2 & 1 \\ 1 & 1 & -2 \end{bmatrix}$ (heat flow on three vertices of a triangle)

   (c) $A = \begin{bmatrix} -3 & -11 & 1 \\ 8 & -3 & -3 \\ -6 & 4 & 2 \end{bmatrix}$ (something else)

6. Find a unit vector that points in the direction of a diagonal of the cube. Then give the corresponding skew-symmetric matrix which can be used to rotate about the diagonal. Have the MATLAB script produce an animation that rotates about this diagonal two and a half times. Save a picture of the final image as `exp-6.bmp`

7. Consider the line that connects opposite corners of the top square of the cube. (Note that this line does not go through the origin!) Indicate how you can rotate about this line by $7\pi/6$ radians. Then have MATLAB create an animation of this rotation. Save a picture of the final image as `exp-7.bmp`