Justin's MATH246 Guide and Project 2.

Due by Monday 5 August 2019

Contents

What to Submit

For this project you will need to create an m-file which does all the tasks listed in this project in order. You should then publish the m-file and print.

Each task should be within its own Matlab code section so that the result of each task appears directly after that task, not all at the end. If you're not sure how to properly mark up an m-file for publication like this then ask me.

Matrices and Determinants

Matrices can be stored in Matlab using bracket notation with semicolons separating rows and either commas or spaces separating columns within those rows:

```
A = [1 2 3;-4 0 7]
```

```
A =
     1     2     3
    -4     0     7
```

If a matrix is square then the determinant can be found using the `det` command:

```
B = [1 -1 0 ; -7 -1 5 ; 8 1 0];
det(B)
```

```
ans =
   -45
```

Task 1: Find the determinant of the matrix

$$\begin{bmatrix} 3 & 2 & 8 & -1 \\ 0 & -1 & -3 & 5 \\ 2 & 2 & -2 & 7 \\ 5 & 1 & 0 & 1 \end{bmatrix}$$

If the matrix has (symbolic) variables in it this is still okay:

```
syms a;
A = [a 2 -1; 0 3 a; a a 5];
det(A)
```

```
ans =
- a^3 + 2*a^2 + 18*a
```

Task 2: Declare the variable b as symbolic, and find where the determinant of the following matrix is zero using `det` wrapped in `solve`.

$$\begin{bmatrix} 3 & b & 2 \\ b & b & 5 \\ 1 & -1 & 5 \end{bmatrix}$$

Factoring Polynomials

When hunting for fundamental sets we often have to factor the characteristic polynomial. Matlab can make this process easy once we understand what the output looks like. Here is the factorization of $x^3 - x^2 - 8x + 12$:

```
syms x
factor(x^3-x^2-8*x+12)
```

```
ans =
[ x + 3, x - 2, x - 2]
```

We see that Matlab gives us the factors, with multiplicity if appropriate, as a list. Notice that in standard mode Matlab will not factor out complex terms:

```
factor(x^2+4)
```

```
ans =
x^2 + 4
```

If we wish it to do so then we should tell it:

```
factor(x^2+4,'FactorMode','Complex')
```

```
ans =
[ x + 2.0i, x - 2.0i]
```

Task 3: Factor the polynomial $x^2 - 7x + 10$.

Task 4: Factor the polynomial $x^8 - 18x^6 - 4x^5 - 51x^4 + 612x^3 - 216x^2 + 648x - 4860$ and include complex terms.

Solving Higher Order Homogeneous Linear Differential Equations

Matlab can easily handle higher order linear differential equations. The key thing to know is to set the unknown function up properly and use the `dsolve` command. For a simple differential equation like $y'' - 3y' - 4y = 0$ we simply do something like:

```
syms y(t)
dsolve(diff(y,2)-3*diff(y)-4*y==0)
```

```
ans =
C1*exp(-t) + C2*exp(4*t)
```

If we want to give it some initial values it's notationally easier to prepare the derivatives before using `dsolve` and it makes things look prettier. Notice below that Matlab will happily accept the definitions of `Dy` and `D2y` as derivatives even though `y` itself is symbolic and not known. Here's the above DE with the conditions $y(1) = -1$ and $y'(1) = 3$ added to make it an IVP.

```
syms y(t)
Dy = diff(y);
D2y = diff(y,2);
dsolve(D2y-3*Dy-4*y==0,y(1)==-1,Dy(1)==3)
```

```
ans =
(2*exp(4*t)*exp(-4))/5 - (7*exp(-t)*exp(1))/5
```

Task 5: Solve the initial value problem $y'' - 3y' + 10y = 0$ with $y(0) = 1$ and $y'(0) = 17$.

Task 6: Solve the differential equation $D^5y - 9D^4y + 36D^3y - 108D^2y + 243Dy - 243y = 0$.

Nonhomogeneous

It makes no difference to Matlab whether the system is nonhomogeneous. For example here's the solution to the differential equation $y'' - y' - 2y = 3$:

```
syms y(t);Dy=diff(y);D2y=diff(y,2);
dsolve(D2y-Dy-2*y==3)
```

```
ans =
C5*exp(-t) + C6*exp(2*t) - 3/2
```

Task 7: Solve the differential equation $y'' - y' - 2y = \cos(t)$.

Task 8: Solve the initial value problem $y''' + 3y'' - 4y' = t + \cos(t)$ with $y(0) = 0$ and $y'(0) = 1$ and $y''(0) = 3$.

Laplace Transforms and Inverse Laplace Transforms

Matlab can calculate Laplace Transforms easily using the `laplace` command, as long as you understand exactly what to plug in. Basically we have to give it the function and both the "input" and "output" variables. We usually go from t to s and so we'd do something like this:

```
clear all
syms s t
laplace(exp(3*t),t,s)
```

```
ans =
1/(s - 3)
```

So if something it not in our table it's still easy. Here I wrapped the answer in `simplify` because it made the result nicer:

```
simplify(laplace(t^3*sin(t)*exp(3*t),t,s))
```

```
ans =
(24*(s^3 - 9*s^2 + 26*s - 24))/(s^2 - 6*s + 10)^4
```

Task 9: Find and simplify $\mathcal{L}\{t^3 \sin(2t)\}$.

Task 10: Find and simplify $\mathcal{L}\{(2t^2 - t)\cos(5t)\}$.

Matlab can also take inverse Laplace Transforms with the `ilaplace` command. For example suppose you know that $\mathcal{L}\{y\} = \frac{s+1}{s^2+9}$ and you wish to know y. Notationally you could think of this as $y = \mathcal{L}^{-1}\left\{\frac{s+1}{s^2+9}\right\}$ and Matlab can compute this with:

```
clear all
syms s t
ilaplace((s+1)/(s^2+9),s,t)
```

```
ans =
cos(3*t) + sin(3*t)/3
```

Task 11: Find the function y which satisfies $\mathcal{L}\{y\} = \frac{1}{(s-2)^3}$.

Task 12: Find the function y which satisfies $\mathcal{L}\{y\} = \frac{s^2+s+1}{s^3-5s^2-2s}$.