HW7 due **FRIDAY** March 1 (7.1-7.2 on paper, 7.3 **online**)

7.1 Lagrange Polynomials

4 pts

(t)	(H)
0	0
1	1
2	7
3	8

You just watched some poor kid let go of her balloon in a 8-m high mall food court. The four data points at right represent the height of a balloon (H, in meters) as a function of time (t, in seconds).

(a) Create the 3rd-order Lagrange polynomial that *exactly* passes through all $H(t)$ data, and use it to estimate (*extrapolate*) the balloon height at **time = 4** seconds. *Show your work!!*

(b) Create the 3rd-order Lagrange polynomial to interpolate the height of the balloon at **time = 2.5** seconds (before hitting the ceiling).

(c) Create a 1st-order Lagrange polynomial to interpolate the height of the balloon at **time = 2.5** seconds. This is basically doing a *LINEAR SPLINE* to interpolate h at t = 2.5 sec.

(d) Make a **plot** by-hand of the data (on the paper you're handing in), with the three interpolated points above super-imposed, and then **comment** on **concerns** using the Lagrange polynomials above for answering (a) – (c). Do your results above make "common-sense"? Do you "trust" them, or are these just "bogus mathematical outputs"?

7.2 Making a Cubic Spline (by-hand)

8 pts

(t)	(H)
1	1
2	7
3	8

Take just the **last three points** from the $H(t)$ data above (re-printed at right), and develop a cubic-spline *BY-HAND* that matches that data, and use it to estimate the height at ***t* = 1.5** and ***t* = 2.5 seconds**. For the two end points (t = 1 and t = 3) apply "natural spline" (y'' = 0) conditions.

When I say "by-hand" what I mean is (and *show your work!!!*)

- write out the form of the cubic interpolating polynomials, $F_i(x) = a_i + b_i\, x + c_i\, x^2 + d_i\, x^3$, in both intervals (i = 1 and 2),
- apply all matching conditions to create a system of 8 equations with 8 unknowns,
- re-write these equations in matrix form ($Ax=b$),
- use MATLAB to solve for the 8 unknown coefficients, [**OK** to use MATLAB **for just this step!**]
- Finally, by-hand again, use the resulting functions to interpolate the force required at both t = 1.5 **and** t = 2.5.

Then feel free to compare the interpolated value at t = 2.5 from the spline with the value you got with the cubic and linear polynomials in 7.1. What do you think is "best"?

7.3 Using MATLAB to Explore Two Interpolating Methods

The data below represents the percent of cars (R_i) needing repairs as a function of time in years (t_i) after the model is released:

9 pts

t_i (yr)	**0**	**2**	**4**	**6**	**8**	**10**	**15**	**20**
R_i (%)	**3**	**4**	**7**	**9**	**12**	**13**	**19**	**21**

Develop a short MATLAB script that calculates and plots the interpolation of this repair data over a finer range of every year from t = 0 to 20. Make sure ALL your steps and calculations below are included in your script so we can evaluate your methodology and answers (*i.e.* don't do things off-line on paper that we can't see).

Continued next page

HW7 due **FRIDAY** March 1 (7.1-7.2 on paper, 7.3 **online**)

7.3 (continued …)

Start by creating a new vector of time: `tnew = [0:20]`. Now create three new vectors that interpolate the data over `tnew` using each of three different methods:

a) Exact Polynomial: Use `polyfit` to calculate the *single* polynomial that *exactly* fits all eight data points, and `polyval` to evaluate it over the `tnew` vector.

b) Linear Spline: Use `interp1` with the "linear" option.

c) Cubic Spline: Use `interp1` with the "spline" option (which defaults to a cubic spline with "not-a-knot" end conditions).

Make a single plot with the raw data (as **circles**) plus all three interpolated lines, where exact polynomial line (a) is dashed **red**, linear spline (b) is solid **black**, cubic spline (c) is dash-dot **blue**, maybe using something similar to the command below:

```
plot(ti,Ri,'ok', tnew,Ra,'--r', tnew,Rb,'-k', tnew,Rc,'-.b')
```

Constrain the plot over 0 ≤ *Repair %* ≤ 25 and 0 ≤ *time* ≤ 20 with the command `axis([0 20 0 25])`.

Finally, interrogate the plot and the interpolated vectors over the range in `tnew` to answer the following questions:

i. What does <u>each</u> model say the *interpolated* repair rate is at t = 3 years?

ii. What does <u>each</u> model say the *interpolated* repair rate is at t = 18 years?

iii. If you trust the data 100%, use the plot and your engineering judgment to determine which model you think did the best job at *interpolating* the data over the entire range.

Please submit the following **online** in Carmen for problem **7.3 only**

- Your **commented** final script (**HW7_3.m**) that does all the above
- The **.pdf** of your final (labeled!) plot.
- In the comment box, enter your answers for questions (i) through (iii) above. Be sure to write values accurate to at least one decimal, and clearly say what answer goes with what question. For example, you could write:
  (i) Rate at t=3: (a) 4.5%, (b) 7.6%, (c) 12.9%
  (ii) Rate at t=18: (a) 20.0%, (b) -14.3%, (c) 1.1%
  (iii) I have no opinion – just tell me what I should like best.

Please don't write anything from 7.3 on paper. Your code should be complete and show all your steps,