Chap 14 TOPICS WE DO

- SOLVE LINEAR ALGEBRAIC EQNS ✓
- FIT DATA TO POLYNOMIAL EQN
  FIND COEFFICIENTS OF POLYNOMIAL
  TO GET "BEST FIT" BETWEEN
  VALUES OF POLYNOMIAL EQN & DATA VALUES
- SYMBOLIC MATH

For practical examples of solving systems of linear algebraic equations,
we looked at “Notes - Material Balances” at ReactorLab.net, Resources, Matlab, Solving math problems with Matla
http://reactorlab.net/resources/matlab/examples-solving-math-problems-matlab/

Material Balances and System of Linear Equations

In most cases in this section, we will get linear algebraic equations when we write material balances.

Linear algebraic equations are those in which unknown variables do not appear in a product or fraction with themselves or any other unknown variable, and also do not appear as exponents or in logarithms. That is, with unknown variables x and y and known constants a and b, we will see ax and by but NOT any of the following: ax^2 , axy , ax/y , e^{ax}, $\log(by)$.

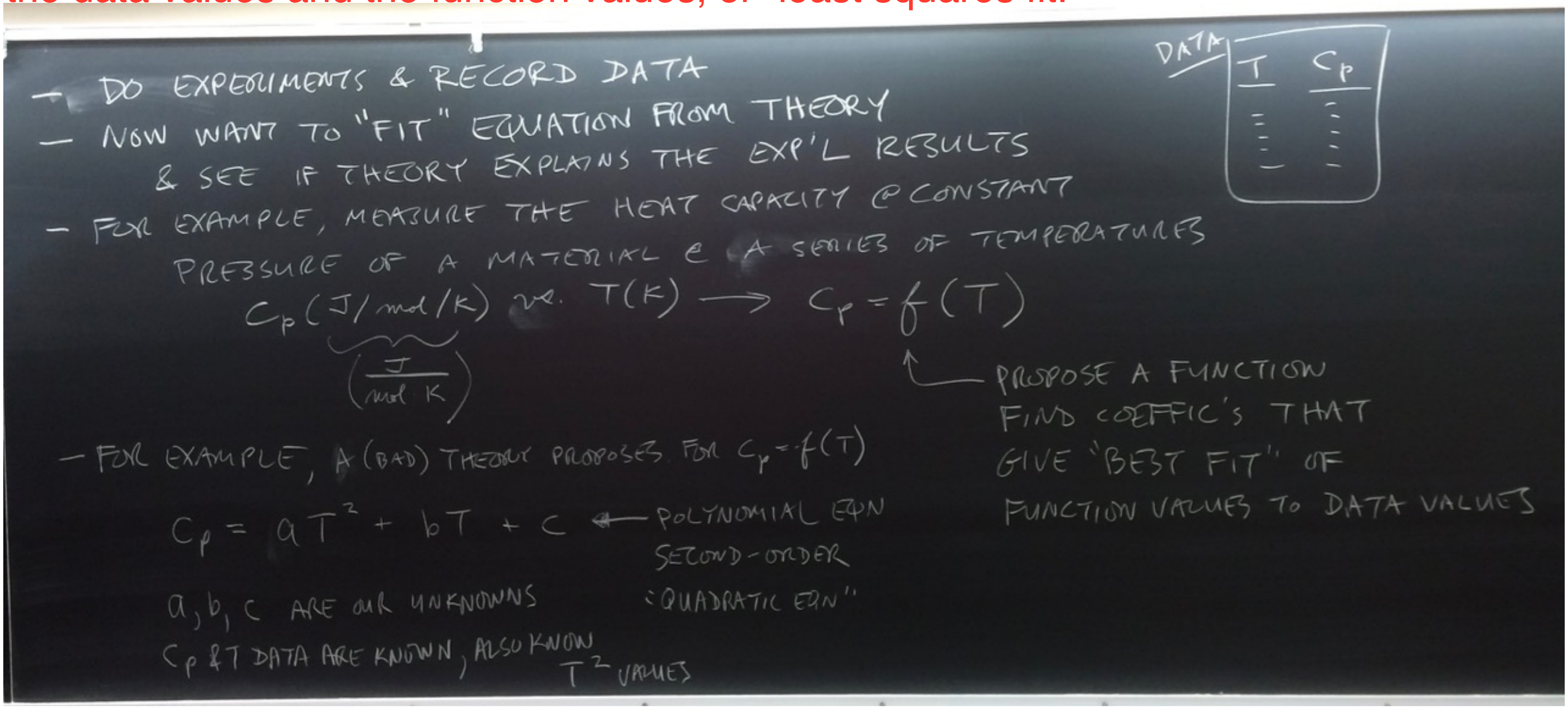
Linear equations can be solved by the methods of "linear algebra."

Some of the problems may be simple such that we can solve them in a step-wise manner and not involve explicitly the methods of linear algebra. An example is Himmelblau & Riggs, Workbook problem 7.3.

Other problems will require that we write matrices and use the methods of linear algebra. An example is Himmelblau & Riggs, Workbook problem 7.4.

These problems are solved on the following slides.

Another example of solving linear algebraic equations is fitting polynomial functions to experimental data. The goal is to find values of the unknown coefficients in the polynomial function in order to get the best fit of the predictions of the polynomial to the data. “Best fit” often means a minimum value of the sum of the squared errors between the data values and the function values, or “least squares fit.”



IF WE HAVE 3 DATA PTS $\longrightarrow$ 3 EQNS IN 3 UNKNOWNS

$$\underbrace{\begin{bmatrix} T_1^2 & T_1^1 & T_1^0 \\ T_2^2 & T_2^1 & T_2^0 \\ T_3^2 & T_3^1 & T_3^0 \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} a \\ b \\ c \end{bmatrix}}_{x} = \underbrace{\begin{bmatrix} C_{p_1} \\ C_{p_2} \\ C_{p_3} \end{bmatrix}}_{y}$$

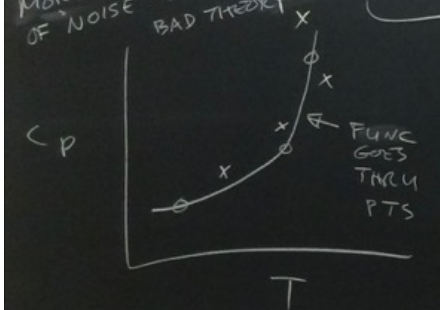
$$x = A^{-1} y$$

$$x = inv(A) * y$$

NO DOT

$$C_{CALC} = A * x$$

BUT FUNC DOESN'T GO THRU MORE EXP'S BECAUSE OF "NOISE" IN DATA OR BAD THEORY



- BECAUSE ALL EXP'S HAVE SOME LEVEL OF NOISE WANT TO DO MORE THAN MINIMUM # OF EXPS.
- DO 1 MORE EXP, NOW A IS 4X3, y IS 4X1

BUT CAN NOT TAKE INVERSE OF NON-SQUARE MATRIX A!

- $\left[A^T_{4\times3} A_{4\times3}\right] = A^*_{3\times3}$

$$A^*_{3\times3}\, x = A^T_{4\times3}\, y$$

$$\underbrace{A^{*-1} A^*}_{I} x = A^{*-1} A^T y$$

ALSO WORKS WITH, e.g., 20 MORE PTS

$$x = A^{*-1} A^T y \longrightarrow$$

IN MATLAB

$A_{ST} = A^T * A$

$x = inv(A_{ST}) * A^T * y$

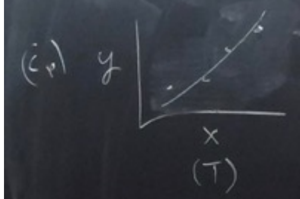
BETTER SOLN IN MATLAB IS

$$x = A \backslash y$$

x has a different meaning here than on previous board
here x is the independent variable (which was T on previous board)
on previous board, x was the vector of unknown coefficient values
on both boards, y are the function values

– WE ARE USING A POLYNOMIAL AS EXAMPLE OF A LINEAR ALGEBRAIC EQN
BUT METHOD WORKS FOR ANY LINEAR ALG. EQN.

– MATLAB HAS SPECIAL FUNCTIONS FOR POLYNOMIAL EQNS: polyfit & polyval



coeffic = polyfit(x, y, n) % RETURNS COEFFICIENTS OF y = polynomial function of x of ORDER n

NOTE: A STRAIGHT LINE IS POLYNOMIAL OF ORDER 1
$y = m x^1 + b$

– FOR OUR HEAT CAPAC. DATA

NAME OF FUNCTION

ANY VARIABLE NAME

coeffic = polyfit(T, Cp, 2) % RETURNS VECTOR OF a, b, c VALUES

% NEED n+1 <u>OR</u> <u>MORE</u> VALUES OF x, y

Ccalc = polyval(coeffic, T) → BETTER Ccalc = polyval(coeffic, Tp)

plot(T, Cp, 'o', T, Ccalc)

Tp = T(1) : 0.01 : T(length(T)) % e.g., FOR 7 DATA @ 1K+ INCREMENTS

OR T(end)

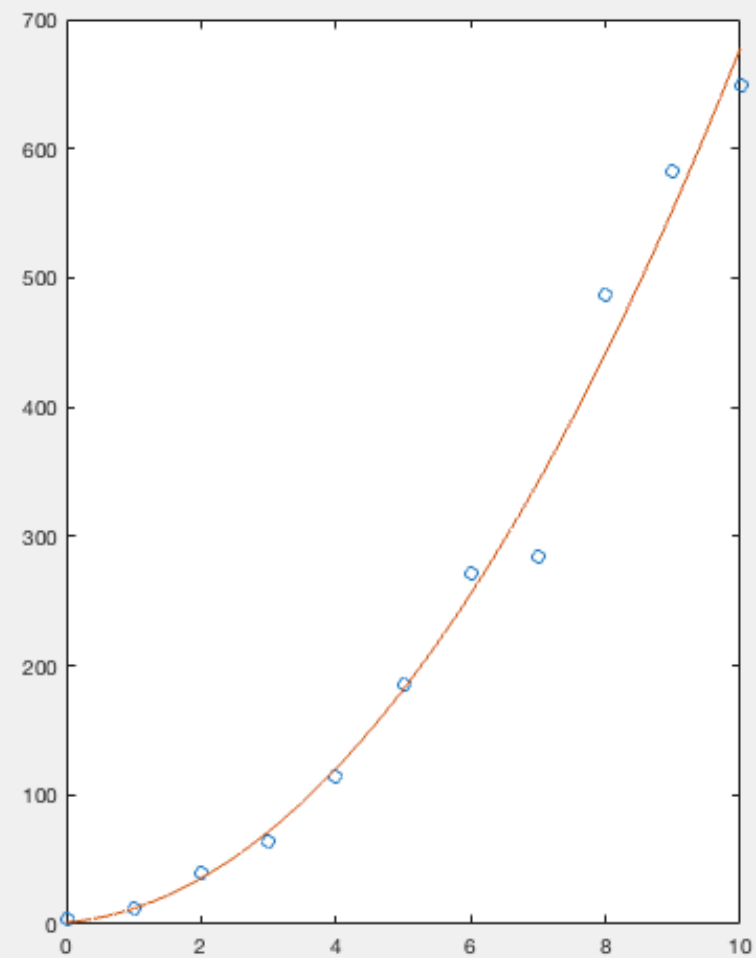
TO CALC FUNCTION @ MORE "X" VALUES plot(T, Cp, 'o', Tp, Ccalc) poly01.m

```
%% polyfit soln

x = 0:10;
y = 5*x.^2 + 4*x + 3;
% add some random scatter to y
rf = 0.4;
y = y + y .* (rf*rand(1,length(y)));
c = polyfit(x,y,2)
xp = 0:0.1:10;
yp = polyval(c,xp);
plot(x,y,'o',xp,yp)

%% matrix soln

A = [x'.^2 x' ones(length(x),1)];
b = y'
u = A\b
```

Command Window

```
c =

    6.3143    4.5038    1.3343
```

Figure 1



partial pseudo-code for possible exam 2 problem

last value of vector X = X(end)

PSEUDO-CODE

FIND MIN & MAX VALUES OF ARRAY a <u>NOT</u> USING MATLAB's min & max FUNCTIONS!

LOWER-CASE

PICK INITIAL VALUES, e.g., MIN = inf, MAX = -inf ?

VARIABLE NAMES UPPER-CASE

```
REPEAT FOR EACH row r OF a
    REPEAT FOR EACH COL c OF a
        IF a(r,c) < MIN
        ASSIGN a(r,c) TO BE NEW VALUE MIN
        IF a(r,c) > MAX
        ASSIGN a(r,c) TO BE NEW VALUE MAX
    END
END
DISPLAY MIN & MAX VALUES
```

CHECK IF YOU AGREE WITH MATLAB FUNCTIONS min & max