MATLAB

Equation Solving & Curve Fitting

CS101 Lecture #24

2017-12-19

Administrivia

Administrivia

- HW9 due this Friday 6pm.
- Q/A on Wednesday's lab sessions
- Final exam location update:

29/12/2017	09:00-12:00	CS 101	40	A-414
			40	A-418
			64	RC-1 101

Random numbers

Random number generator

- Matlab supports a variety of RNG:
  - `rand`, uniform distribution [0,1)
  - `randi`, random integers [1,n]
  - `randn`, *standard* normal (Gaussian) distribution

rand, randi

```
rand(5)                  %generate a 5x5 matrix
rand(5,1)                %generate a 5x1 column vector

randi(5)                 %generate a number from [1,2,3,4,5]
randi(5,2)               %generate a 2x2 matrix
randi([-1,1],10,1)       %generate a 10x1 matrix from [-1,0,1]
```

randn

```
randn();          %a single normal number
randn(5);         %generate a 5x5 matrix
a + b*rand();     %a random number drawn from a
                  %normal distribution with
                  %center ‘a’ and std ‘b’
```

rng(seed)

```
rng('default');   %restore the settings as restart
rng(1);           %seeds the rng using a nonnegative integer
                  %so that rand,randi,randn produces a
                  %predictable sequence of random numbers

x = linspace(0,10*pi,1001)';
y = sin(x)./x + 0.02*randn(1001,1);
clf;      %clear current figure window
plot(x,y,'.');
```

Equation Solving

System of linear equations

- A classical linear algebra problem:

A x = b

$$\begin{pmatrix} 2 & 3 \\ 1 & 2 \end{pmatrix} x = \begin{pmatrix} 1 \\ 0 \end{pmatrix}$$

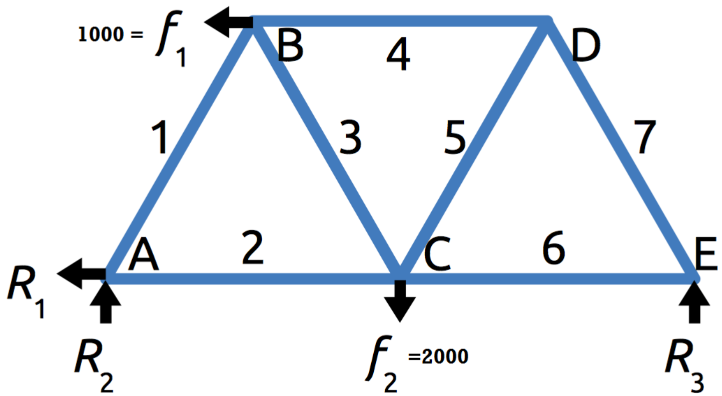
```
A = [2 3; 1 2];
b = [1 0]';
x = A\b;
```

'\': the magic backslash!

- Also called 'left division'

System of linear equations

- Consider a truss problem:



$$
\begin{aligned}
0.5x_1 + x_2 &= R_1 = f_1 \\
0.866x_1 &= -R_2 = -0.5f_2 - 0.433f_1 \\
-0.5x_1 + 0.5x_3 + x_4 &= -f_1 \\
0.866x_1 + 0.866x_3 &= 0 \\
-x_2 - 0.5x_3 + 0.5x_5 + x_6 &= 0 \\
0.866x_3 + 0.866x_5 &= f_2 \\
-x_4 - 0.5x_5 + 0.5x_7 &= 0
\end{aligned}
$$

System of linear equations

$$\begin{pmatrix} 0.5 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0.866 & 0 & 0 & 0 & 0 & 0 & 0 \\ -0.5 & 0 & 0.5 & 1 & 0 & 0 & 0 \\ 0.866 & 0 & 0.866 & 0 & 0 & 0 & 0 \\ 0 & -1 & -0.5 & 0 & 0.5 & 1 & 0 \\ 0 & 0 & 0.866 & 0 & 0.866 & 0 & 0 \\ 0 & 0 & 0 & -1 & -0.5 & 0 & 0.5 \end{pmatrix} \underline{x} = \begin{pmatrix} 1000 \\ -1433 \\ -1000 \\ 0 \\ 0 \\ 2000 \\ 0 \end{pmatrix}$$

$$Tx = f$$

Curve fitting

Polynomials

- The polynomial form

$$f(x) = a_1 x^3 + a_2 x^2 + a_3 x + a_4$$

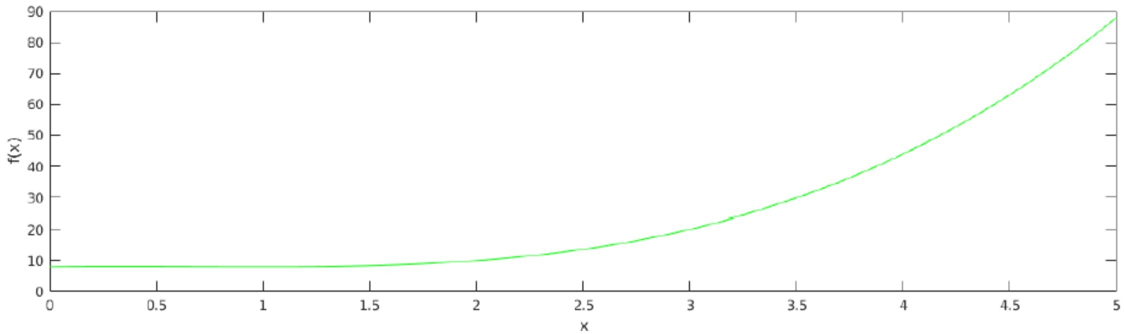
- (note the numbering is a bit odd)
- How MATLAB represents polynomials?

Polynomials

- The polynomial form

$$f(x) = a_1x^3 + a_2x^2 + a_3x + a_4$$

$[a_1\ a_2\ a_3\ a_4]$



$$f(x) = x^3 - 2x^2 + x + 8$$

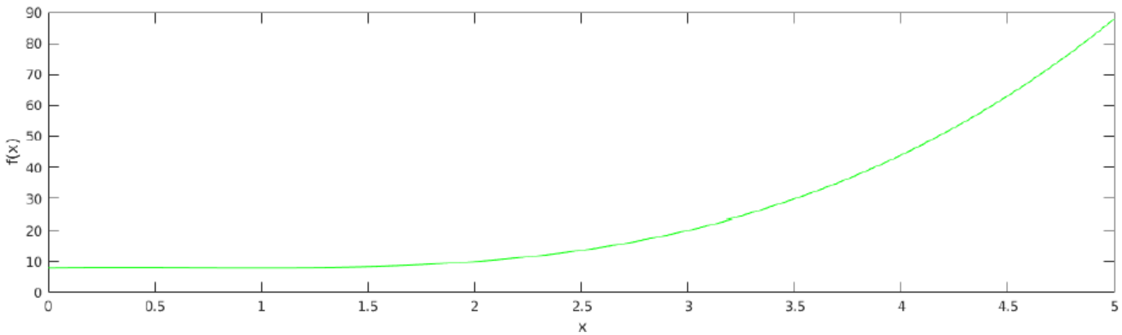
$[1\ -2\ \ 1\ \ 8]$

Polynomials

- How to evaluate a polynomial?

$$f(x) = x^3 - 2x^2 + x + 8$$

$[1 \ -2 \ \ 1 \ \ 8]$




```
y = polyval([1 -2 1 8], 0:0.01:5);
plot(0:0.01:5, y, 'g-')
```

Fitting data with a polynomial

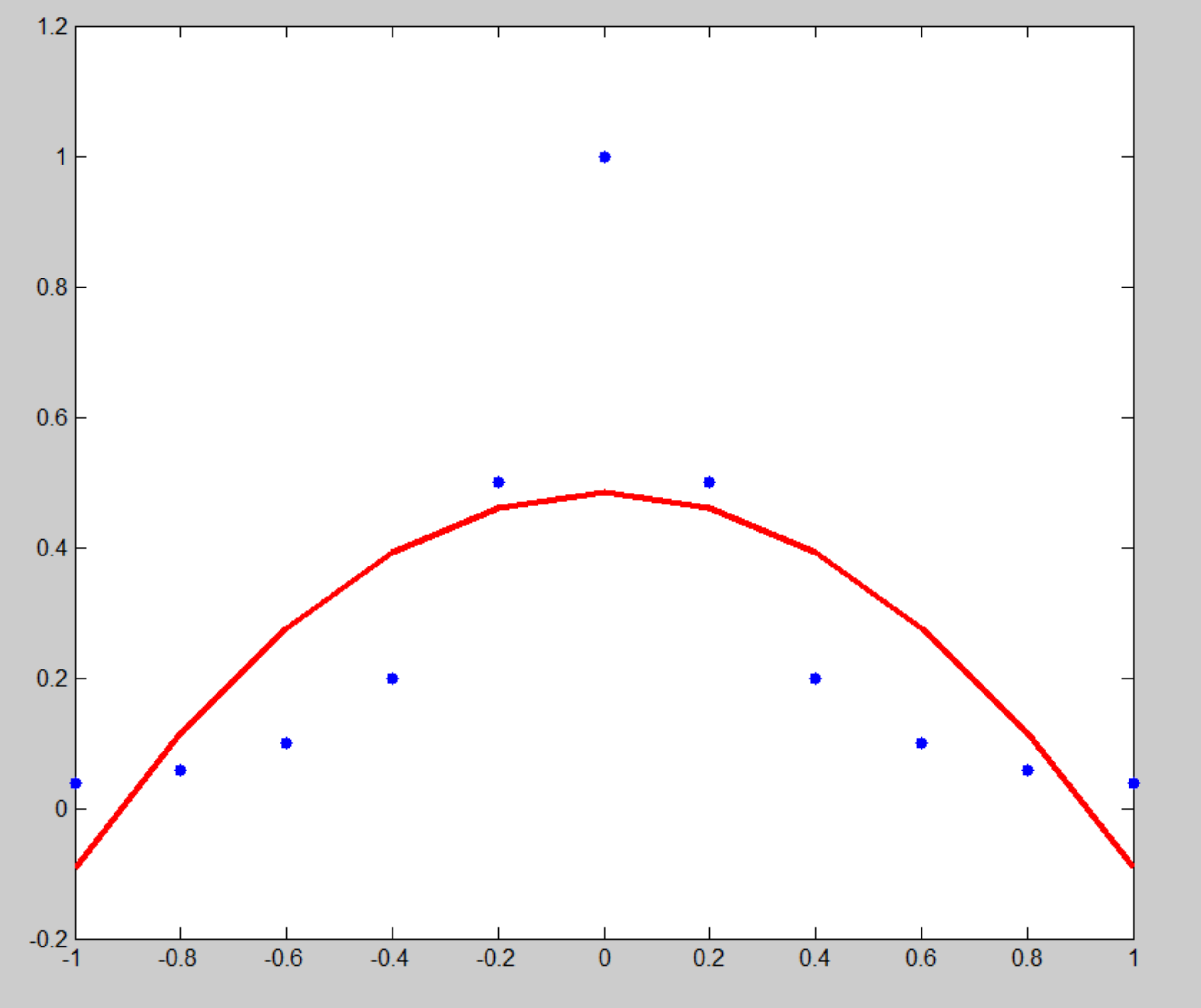
- **Function:** `polyfit(x,y,n)`

```
x = linspace(-1,1,11);
y = [ 0.038 0.058 0.1 0.2 0.5 1 0.5 0.2 0.1 0.058 0.038 ];

coefs = polyfit(x,y,2);  %fit data with a polynomial of degree 2
yfit = polyval(coefs, x);

plot(x,y,'.', x, yfit, 'r-');
```

Fitting data with a polynomial



Fitting data with a polynomial

- **Function:** `polyfit(x,y,n)`

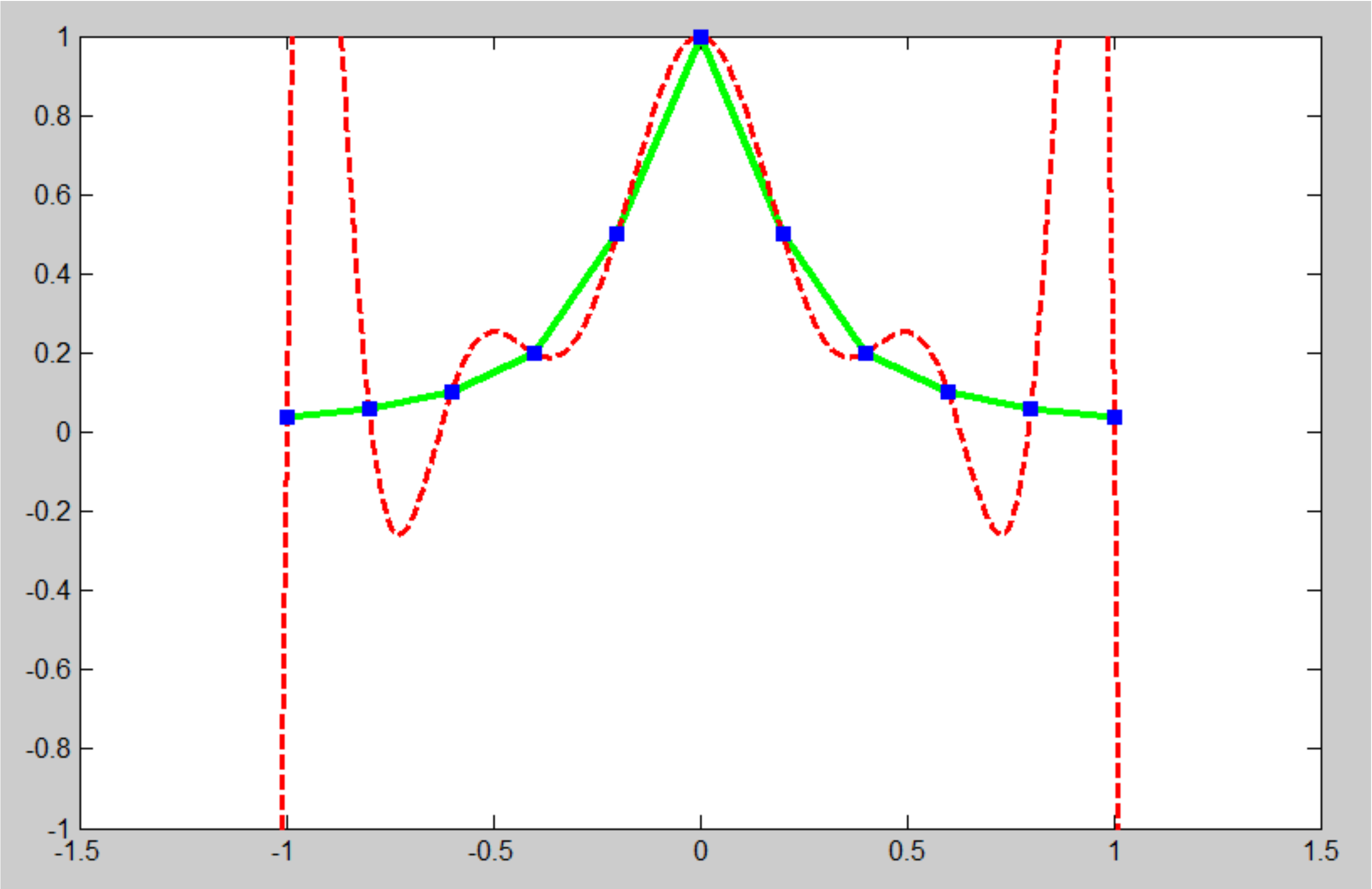
```
x = linspace(-1,1,11);
y = [ 0.038 0.058 0.1 0.2 0.5 1 0.5 0.2 0.1 0.058 0.038 ];

coefs = polyfit(x,y,10);  %fit data with a polynomial of degree 10
yfit = polyval(coefs, x);

plot(x,y,'.', x, yfit, 'g-');

hold on;
xnew = -1.5:0.01:1.5;
ypred = polyval(coefs, xnew);
plot(xnew, ypred, 'r--');
ylim([-1 1]);
```

Fitting data with a polynomial



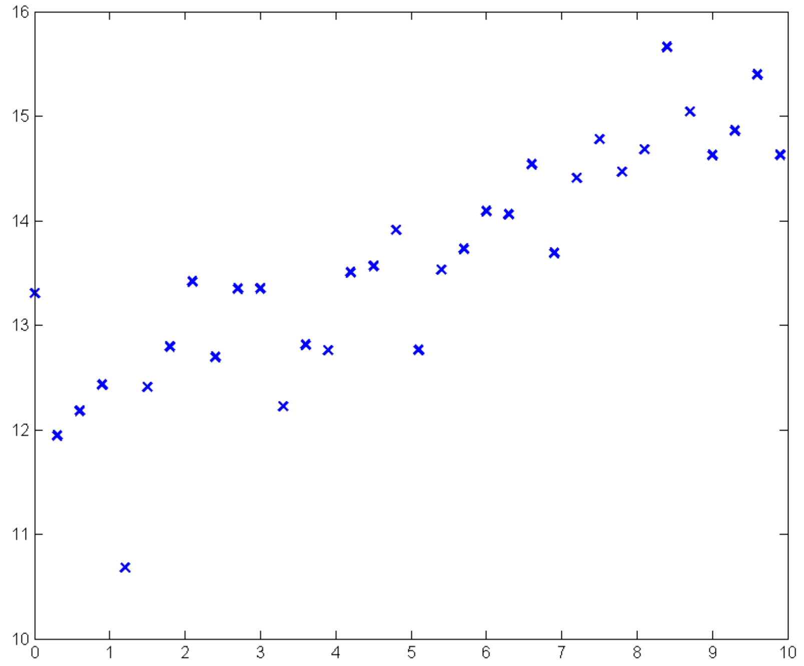
Line fitting

Fit a line to your data

Given $(x_1, y_1), (x_2, y_2), \dots (x_N, y_N), N > 2$, find a line

$$y \ = ax \ + b$$

such that $\hat{y}_i = ax_i + b$ fits to y_i for $i = 1,2, \dots N$

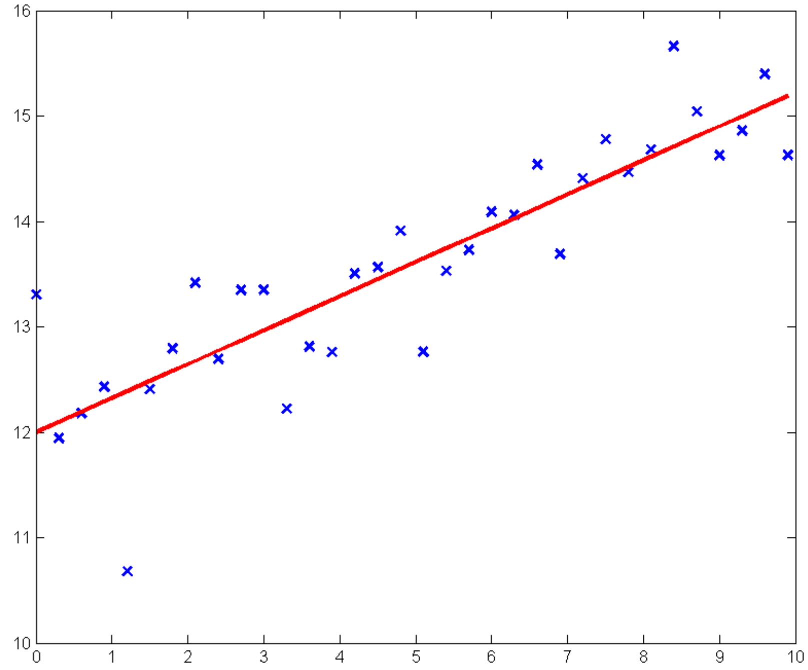


Fit a line to your data

Given $(x_1, y_1), (x_2, y_2), \ldots (x_N, y_N), N > 2$, find a line

$$y = ax + b$$

such that $\hat{y}_i = ax_i + b$ fits to y_i for $i = 1,2, \ldots N$

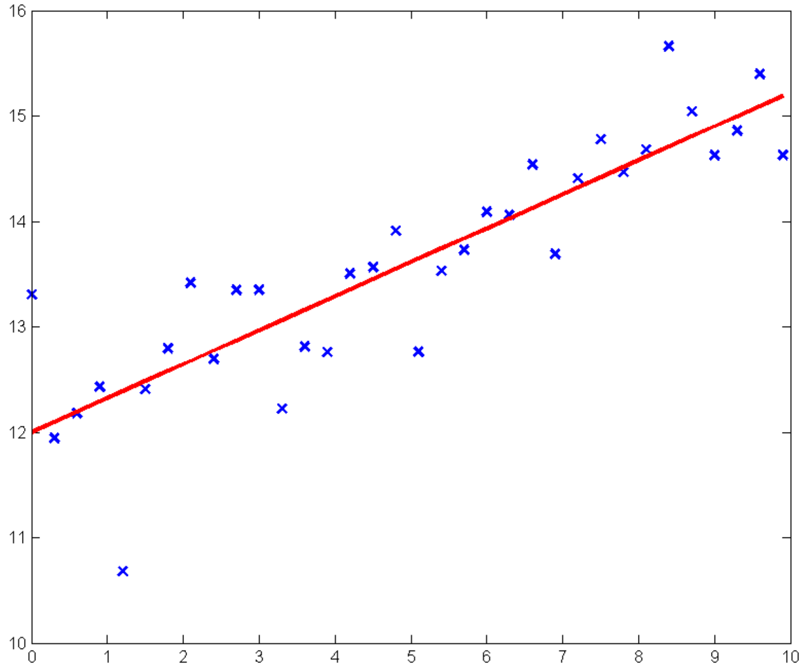


Fit a line to your data

Given $(x_1, y_1), (x_2, y_2), \ldots (x_N, y_N), N > 2$, find a line

$$y = ax + b$$

such that $\hat{y}_i = ax_i + b$ fits to y_i for $i = 1,2, \ldots N$



Goal:

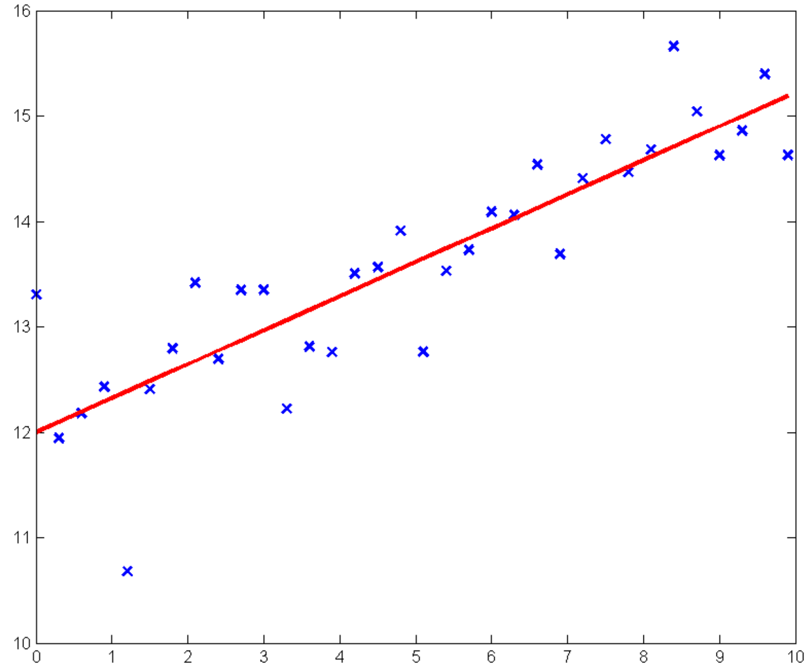
$$\begin{gathered} ax_1 + b = y_1 \\ ax_2 + b = y_2 \\ \ldots \\ ax_N + b = y_N \end{gathered}$$

Fit a line to your data

Given $(x_1, y_1), (x_2, y_2), \ldots (x_N, y_N), N > 2$, find a line

$$y = ax + b$$

such that $\hat{y}_i = ax_i + b$ fits to y_i for $i = 1,2, \ldots N$



Put these in matrix form:

$$\begin{bmatrix} x_1 & 1 \\ x_2 & 1 \\ \vdots & \vdots \\ x_N & 1 \end{bmatrix} \begin{bmatrix} a \\ b \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}$$

$$A * k = y$$

From an over-determined system of linear equations, to find a most likely solution k: the "least square" problem

Fit a line to your data

```
load('xy.mat');

%prepare the maxtrix A and rhs vector
A = [x ones(numel(x),1)];
rhs = y;

k = A\rhs;    %backslash again!

%compute fitted values
yfit = A*k;

figure; plot(x,y,'bx', 'linewidth', 2);
hold on;
plot(x, yfit, 'r-', 'linewidth',2);
plot(x, yfit, 'ms', 'linewidth',2);  %as scattered points
```

Least square for line fitting

The least square formulation for line fitting:

$$a, b = \underset{a,b}{\operatorname{argmin}} \sum_{i=1}^{N} \|y_i - (ax_i + b)\|^2$$

Why minimizing the sum of squares?

The MLE interpretation of least squares (after class reading):
http://people.math.gatech.edu/~ecroot/3225/maximum_likelihood.pdf