International University
School of Electrical Engineering

Introduction to Computers for Engineers

Dr. Hien Ta

Lecturely Topics

- Lecture 1 - Basics – variables, arrays, matrices
- Lecture 2 - Basics – matrices, operators, strings, cells
- Lecture 3 - Functions & Plotting
- Lecture 4 - User-defined Functions
- Lecture 5 - Relational & logical operators, if, switch statements
- Lecture 6 - For-loops, while-loops
- Lecture 7 - Review on Midterm Exam
- Lecture 8 - Solving Equations & Equation System (Matrix algebra)
- Lecture 9 - Data Fitting & Integral Computation
- Lecture 10 - Representing Signal and System
- Lecture 11 - Random variables & Wireless System
- Lecture 12 - Review on Final Exam

References: H. Moore, *MATLAB for Engineers*, 4/e, Prentice Hall, 2014

G. Recktenwald, *Numerical Methods with MATLAB*, Prentice Hall, 2000

A. Gilat, *MATLAB, An Introduction with Applications*, 4/e, Wiley, 2011

Data Fitting

- data fitting with polynomials – **`polyfit, polyval`**
- examples: Moore’s law,
- Hank Aaaron,
- US census data
- nonlinear fits – **`nlinfit,lsqcurvefit,fminsearch`**
- least-squares polynomial regression
- least-squares with other basis functions
- examples: exponential models
- trigonometric basis functions
- trigonometric with polynomial trends (CO2 data)

key methods & concepts

For Exam-3, you will need to be able to apply the following:

1. least-squares fits with polyfit, polyval
2. least-squares fits with basis functions method
3. transform data before applying polyfit or basis functions
4. least-squares fits with nlinfit
5. do-it-yourself L1 and L2 criteria with fminsearch
6. interpolation functions, interp1, interp2

These are covered in week-11 and week-12 lecture notes.

Polynomial data fitting - review

polyfit, polyval

$$P(x) = p_1 x^M + p_2 x^{M-1} + \cdots + p_M x + p_{M+1}$$

$$\mathbf{p} = [p_1,\ p_2, \ldots,\ p_M,\ p_{M+1}]$$

```
>> doc polyfit
>> doc polyval
```

$$P(x) = 5x^4 - 2x^3 + x^2 + 4x + 3$$

$$\mathbf{p} = [5,\ -2,\ 1,\ 4,\ 3]$$

polynomial $P(x)$ is represented by its coefficients $\mathbf{p}$

Given N data points $\{x_i, y_i\}$, $i=1,2,\ldots,N$, find an M-th degree polynomial that best fits the data – **(polyfit)**

```
% design procedure:

xi = [x1,x2,...,xN];
yi = [y1,y2,...,yN];

p = polyfit(xi,yi,M);

y = polyval(p,x);
```

evaluate $P(x)$ at a given vector x

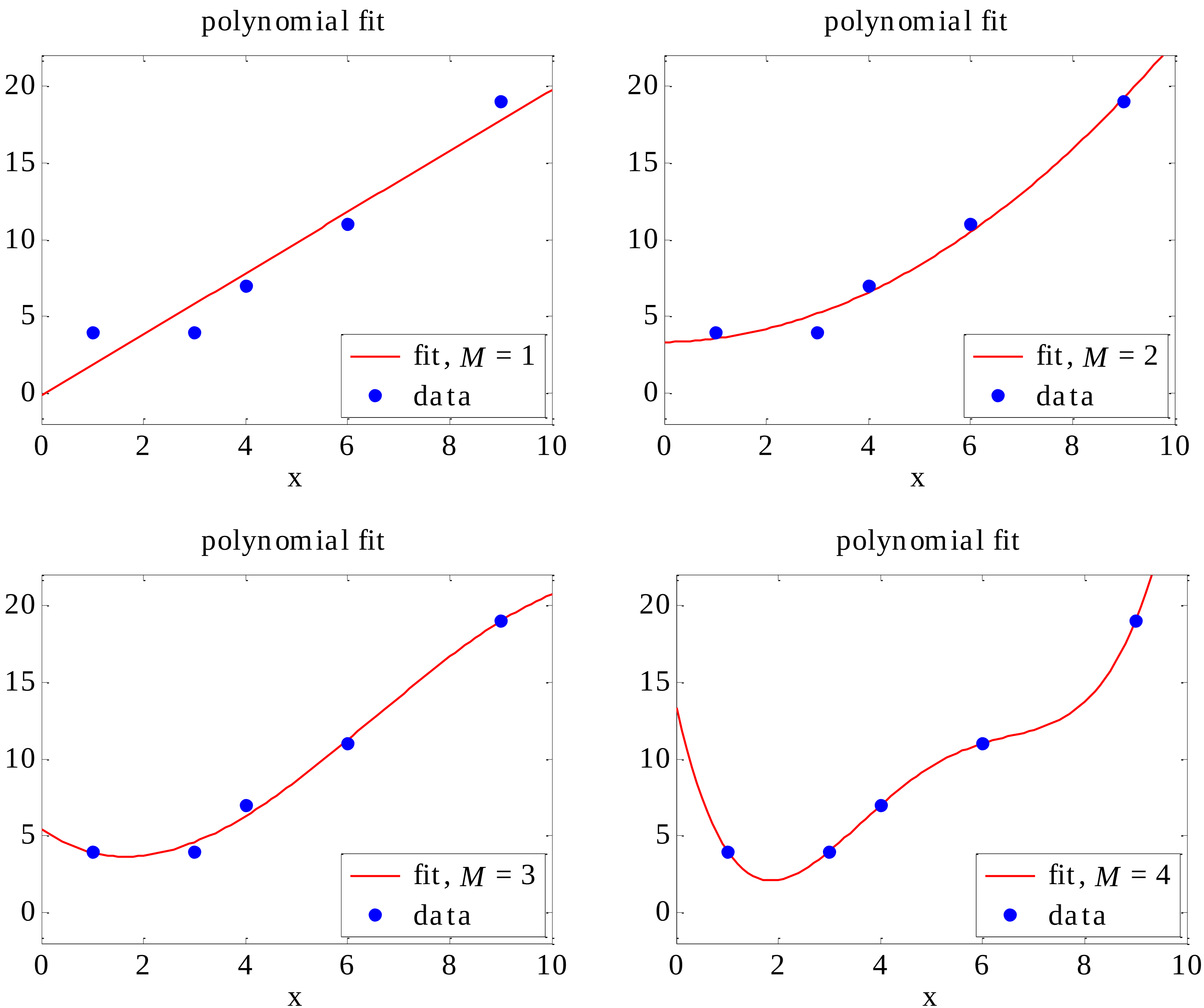
M = polynomial order

if $N = M+1$, the polynomial interpolates the data

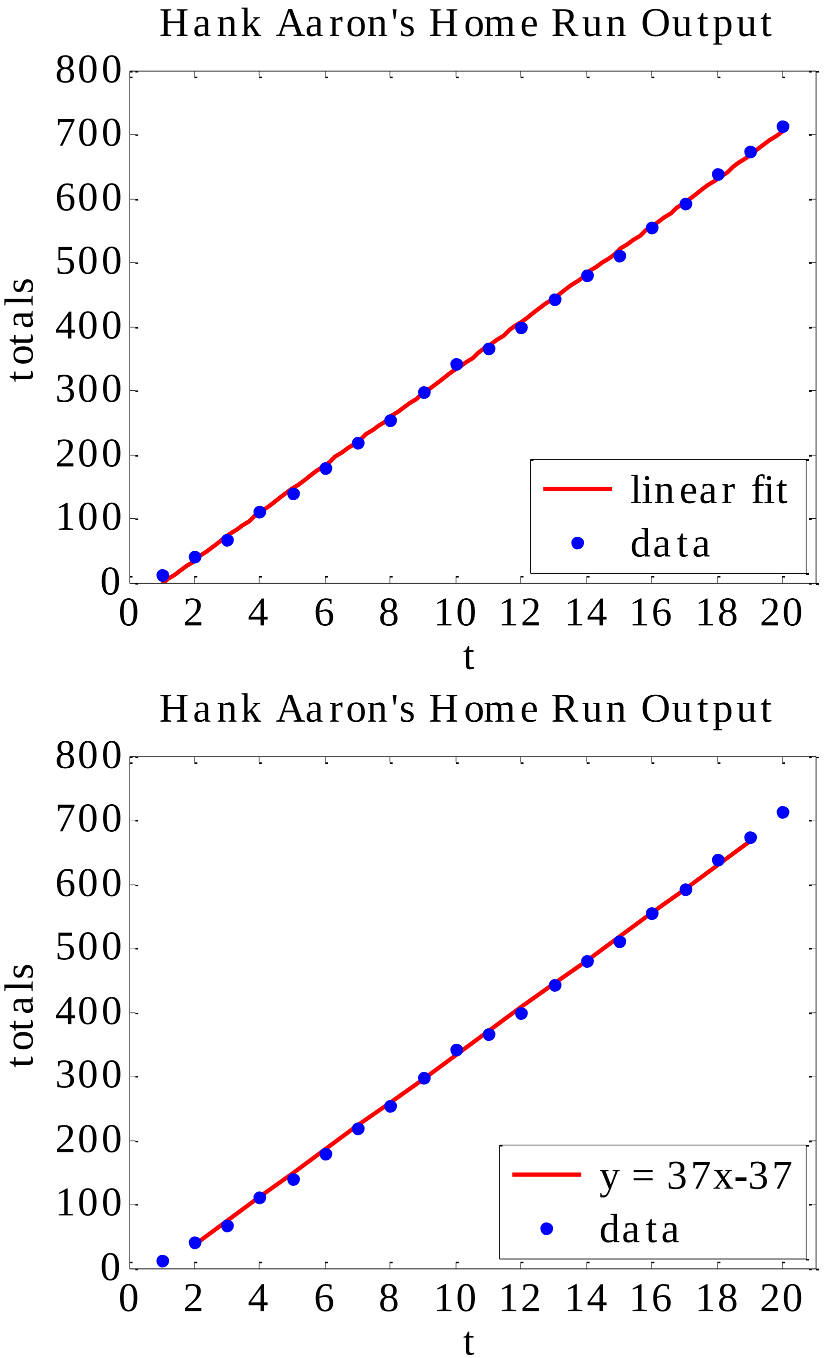
if $N > M+1$, the polynomial provides the best fit in a least-squares sense

$$J = \sum_{i=1}^{N} \big(P(x_i) - y_i\big)^2 = \min$$

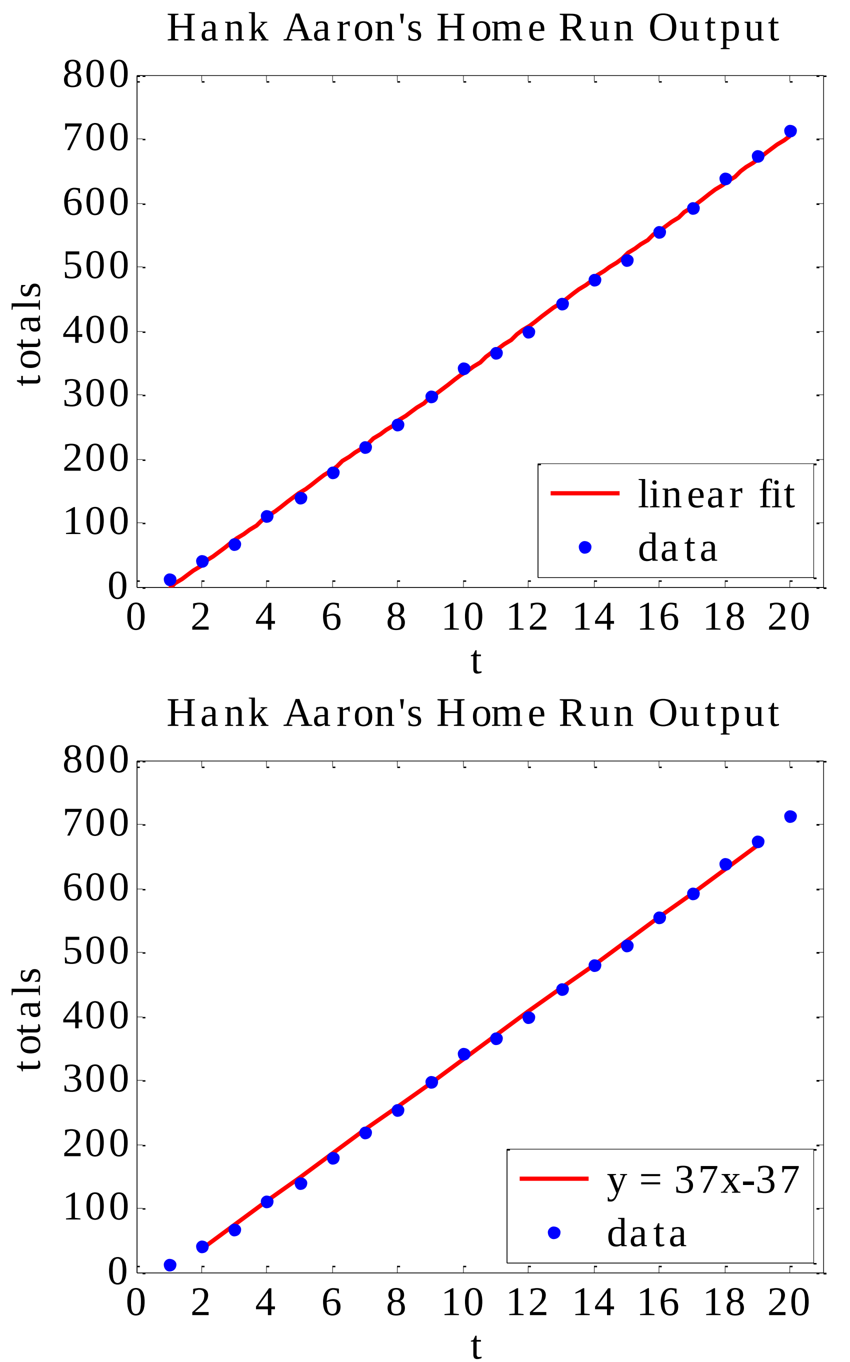
```
xi = [1, 3, 4, 6, 9];
yi = [4, 4, 7, 11, 19];

x = linspace(0,10,101);

for M = [1,2,3,4]
  p = polyfit(xi,yi,M);

  y = polyval(p,x);

  figure;
  plot(x,y,'r-', xi,yi,'b.', 'markersize',25);
  yaxis(-2,22,0:5:20); xaxis(0,10,0:2:10);
  xlabel('x'); title('polynomial fit');
  legend([' fit, {\itM} = ',num2str(M)],...
         ' data', 'location','se');
end
```


polynomial fit
fit, M = 1
data
x
polynomial fit
fit, M = 2
data
x
polynomial fit
fit, M = 3
data
x
polynomial fit
fit, M = 4
data
x
0
5
10
15
20
0
2
4
6
8
10

```
% year      ti    H
% ----------------
  1954       1   13
  1955       2   27
  1956       3   26
  1957       4   44
  1958       5   30
  1959       6   39
  1960       7   40
  1961       8   34
  1962       9   45
  1963      10   44
  1964      11   24
  1965      12   32
  1966      13   44
  1967      14   39
  1968      15   29
  1969      16   44
  1970      17   38
  1971      18   47
  1972      19   34
  1973      20   40
```



```
A = load('aaron.dat');

ti = A(:,2); H = A(:,3);
yi = cumsum(H);

p = polyfit(ti,yi,1)

% p =
%     37.2617  -39.8474

t = linspace(1,20, 101);
y = polyval(p,t);

plot(t,y,'r-', ...
     ti,yi,'b.', ...
     'markersize', 18);
```



Given N data points $\{x_i, y_i\}$, i=1,2,…,N, the following data models can be reduced to linear fits using an appropriate transformation of the data:

linear:	$y = a\,x + b$		
exponential:	$y = b\,e^{ax}$	$\Rightarrow$	$\log(y) = ax + \log(b)$
exponential:	$y = b\,2^{ax}$	$\Rightarrow$	$\log_2(y) = ax + \log_2(b)$
exponential:	$y = b\,x\,e^{ax}$	$\Rightarrow$	$\log(y/x) = ax + \log(b)$
power:	$y = b\,x^{a}$	$\Rightarrow$	$\log(y) = a\,\log(x) + \log(b)$

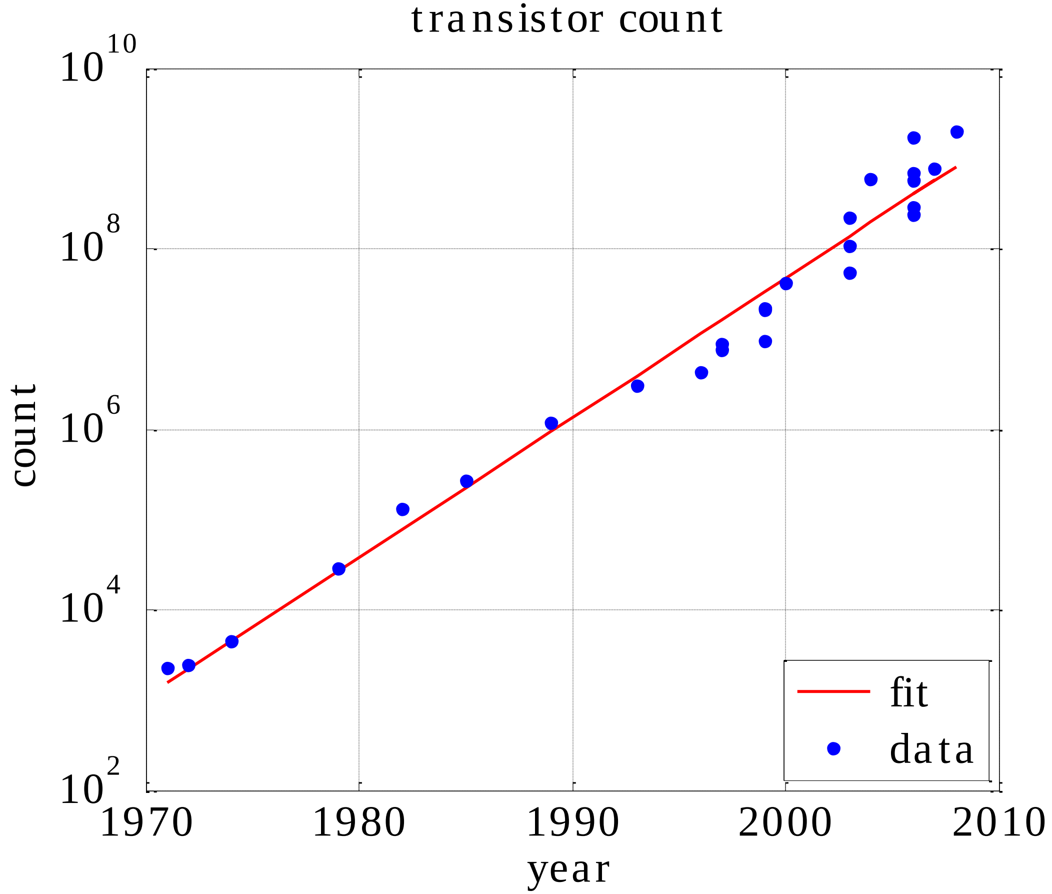
```
p = polyfit(xi,log(yi),1);   % exponential
y = exp(polyval(p,x));          % y=exp(a*x+log(b))
a = p(1);                     % y = exp(p(1)*x+p(2))
b = exp(p(2));              % so that y = b*exp(a*x)
```

yi	ti
2.300e+003	1971
2.500e+003	1972
4.500e+003	1974
2.900e+004	1979
1.340e+005	1982
2.750e+005	1985
1.200e+006	1989
3.100e+006	1993
4.300e+006	1996
7.500e+006	1997
8.800e+006	1997
9.500e+006	1999
2.130e+007	1999
2.200e+007	1999
4.200e+007	2000
5.430e+007	2003
1.059e+008	2003
2.200e+008	2003
5.920e+008	2004
2.410e+008	2006
2.910e+008	2006
5.820e+008	2006
6.810e+008	2006
7.890e+008	2007
1.700e+009	2006
2.000e+009	2008

Moore's law

fitted model $f(t) = b\,2^{a(t-t_1)}$

$$\log_2 f(t) = \log_2 b + a(t - t_1)$$



```
Y = load('transistor_count.dat');  % from textbook

y = Y(:,1);  t = Y(:,2);

t1 = t(1);

p = polyfit(t-t1, log2(y), 1);

% p =
%   0.5138  10.5889   % b = 2^p(2) = 1.5402e+003

f = 2.^(polyval(p,t-t1));

semilogy(t,f,'r-', t,y,'b.', 'markersize',18)
```

fitted model:

```
f(t) = b * 2.^(a*(t-t1)) = 2.^(a*(t-t1)+log2(b));

% a = p(1), log2(b) = p(2) --> b = 2^(p(2))
```

The April 2015 issue of the IEEE Spectrum marks the 50th anniversary of Moore's law.

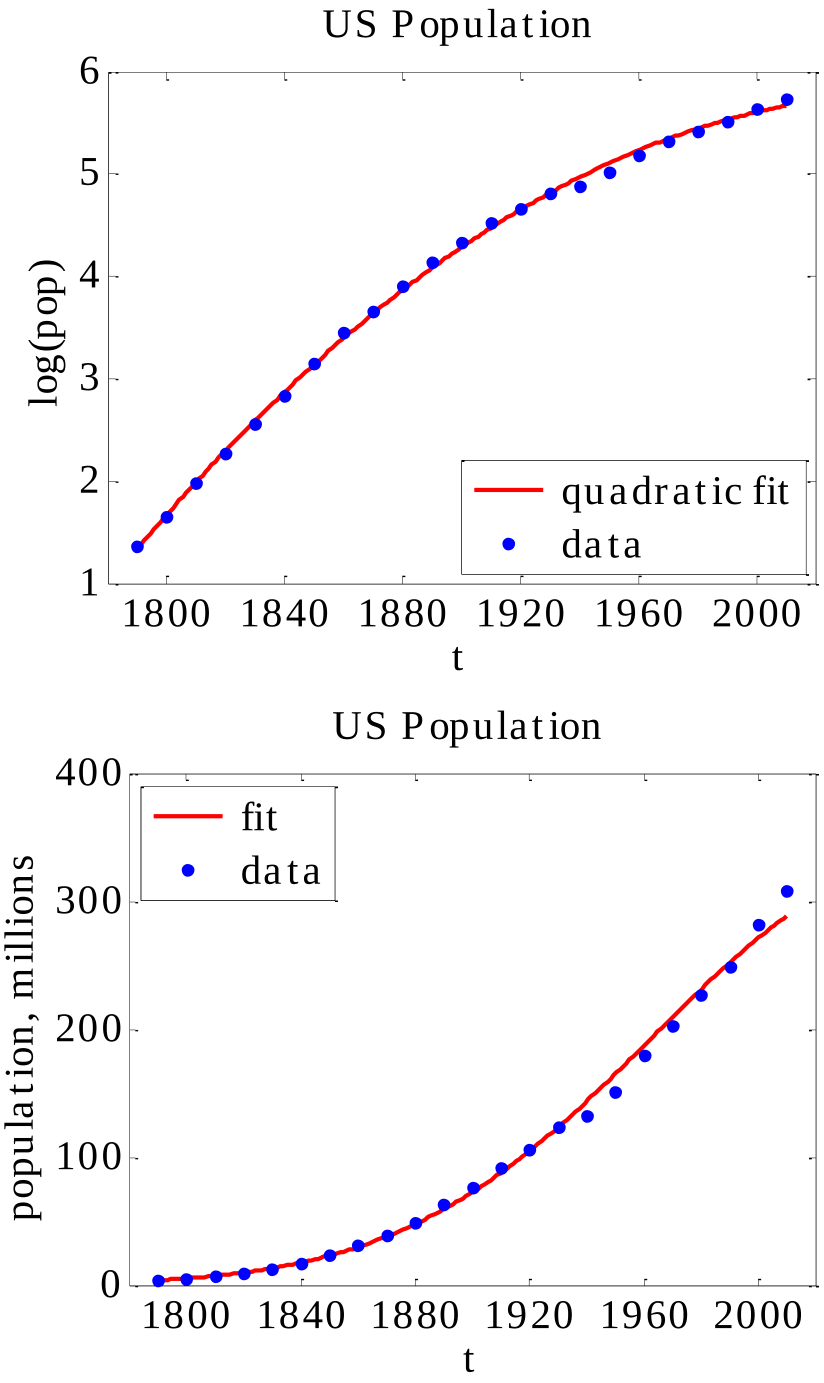
Please see the articles below (accessible from within Rutgers), including an interview with Moore, as well as a link to his original 1965 article.

Moore's 1965 article
Moore's 1975 revision
Moore's 1995 retrospect

The Law That's Not A Law – Moore Interview

The Multiple Lives of Moore's Law
(including some incredible facts about transistors)

```
% source: Wikipedia
% US population in millions
%
%  ti         yi
% --------------
  1790      3.929
  1800      5.237
  1810      7.240
  1820      9.638
  1830     12.866
  1840     17.069
  1850     23.192
  1860     31.443
  1870     38.558
  1880     49.371
  1890     62.980
  1900     76.212
  1910     92.229
  1920    106.022
  1930    123.202
  1940    132.165
  1950    151.326
  1960    179.323
  1970    203.212
  1980    226.546
  1990    248.710
  2000    281.422
  2010    308.746
```

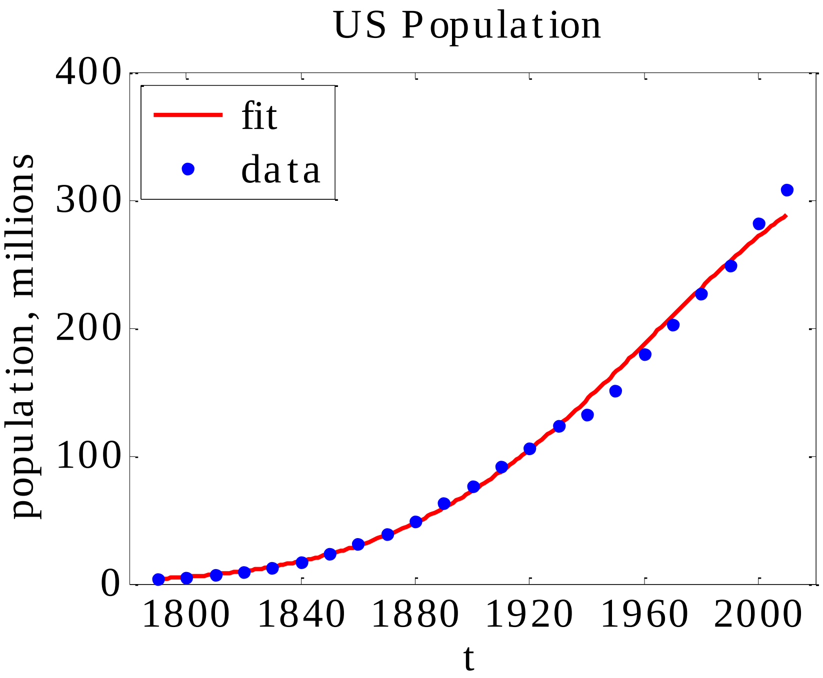


```
A = load('uspop.dat');

ti = A(:,1); yi = A(:,2);

p = polyfit(ti,log(yi),2)     % quadratic fit

% p =
%  -0.0001    0.2653 -266.4672

t = linspace(1790, 2010, 201);
y = exp(polyval(p,t));

figure; plot(t, log(y), 'r-', ...
             ti,log(yi),'b.','markersize',18);

figure; plot(t, y,'r-', ...
             ti,yi,'b.','markersize',18);
```

see problem set-11 for fitting populations with logistic curves

```
A = load('uspop.dat');

ti = A(:,1); yi = A(:,2); t1 = ti(1);

p = polyfit(ti,yi,2)                 % quadratic fit
p1 = polyfit(ti-t1,yi,2)             % t1 = 1790

t = linspace(1790, 2010, 201);
y = polyval(p,t);
y1 = polyval(p1,t-t1);              % shifted origin

norm(y-y1)                          % = 7.7100e-011

plot(t, y,'r-', ti,yi,'b.','markersize',18);

>> num2str([p',p1'],'%12.2e')

ans =
  6.78e-003   6.78e-003
 -2.44e+001  -1.32e-001
  2.20e+004   6.51e+000
```

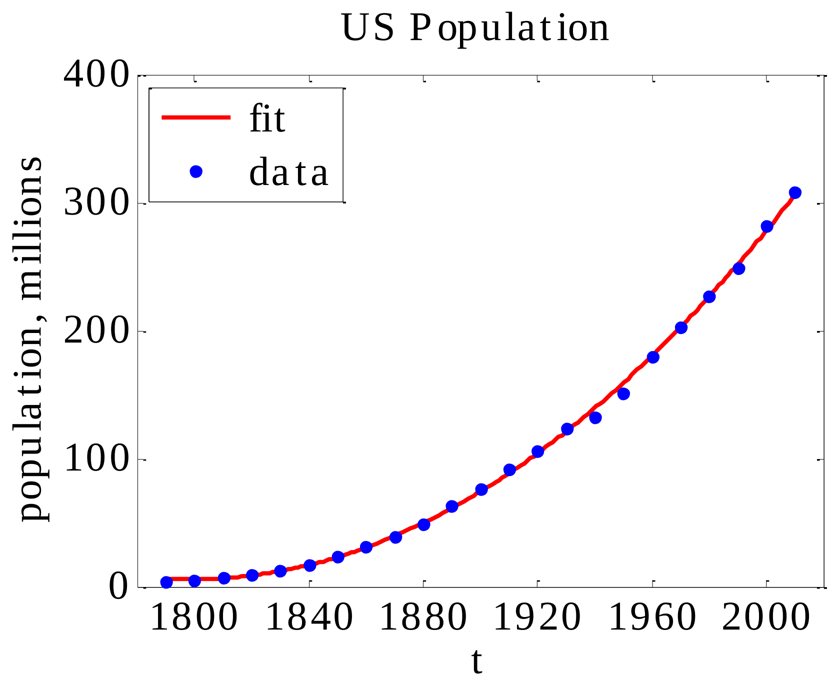
it is always beneficial to shift origin to $t = 0$



see also week-12 homework for logistic curve fits of the US and World populations

exponential fit

polynomial fit



thanks to Matt Brenner,
RU BME '15, Matlab class '11
for this idea

Integral Computing?
Following the lecture from Instructor