CHAPTER 14

14.1 The data can be ordered and tabulated as

i	y	$(y_i - \bar{y})^2$
1	0.9	0.524755
2	1.05	0.329935
3	1.27	0.125599
4	1.3	0.105235
5	1.32	0.092659
6	1.35	0.075295
7	1.35	0.075295
8	1.42	0.041779
9	1.47	0.023839
10	1.47	0.023839
11	1.55	0.005535
12	1.63	3.14E-05
13	**1.65**	**0.000655**
14	1.66	0.001267
15	1.71	0.007327
16	1.74	0.013363
17	1.78	0.024211
18	1.82	0.038259
19	1.85	0.050895
20	1.92	0.087379
21	1.95	0.106015
22	1.96	0.112627
23	2.06	0.189747
24	2.14	0.265843
25	2.29	0.443023
Σ	**40.61**	**2.764416**

(a) $\bar{y} = \dfrac{40.61}{25} = 1.6244$

(b) 1.65

(c) There are two values that occur most frequently: 1.35 and 1.47.

(d) range = maximum – minimum = 2.29 – 0.9 = 1.39

(e) $s_y = \sqrt{\dfrac{2.764416}{25-1}} = 0.339388$

(f) $s_y^2 = 0.339388^2 = 0.115184$

(g) c.v. $= \dfrac{0.339388}{1.6244} \times 100\% = 20.89\%$

Here is how the problem would be answered using MATLAB's built-in functions:

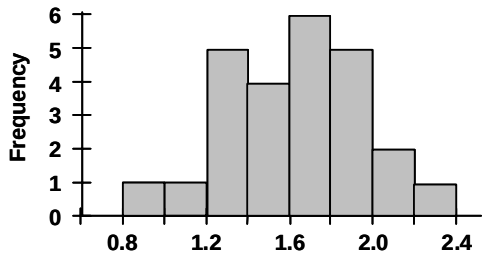
```
clear, clc
y=[0.9 1.32 1.96 1.85 2.29 1.42 1.35 1.47 1.74 1.82 ...
  1.3 1.47 1.92 1.65 2.06 1.55 1.95 1.35 1.78 2.14 ...
  1.63 1.66 1.05 1.71 1.27];
>> m=mean(y)
m =
    1.6244
>> median(y)
ans =
    1.6500
```

```
>> mode(y)
ans =
    1.3500
>> range=max(y)-min(y)
range =
    1.3900
>> s=std(y)
s =
    0.3394
>> var(y)
ans =
    0.1152
>> cv=s/m
cv =
    0.1152
```

14.2 The data can be sorted and then grouped. We assume that if a number falls on the border between bins, it is placed in the lower bin.

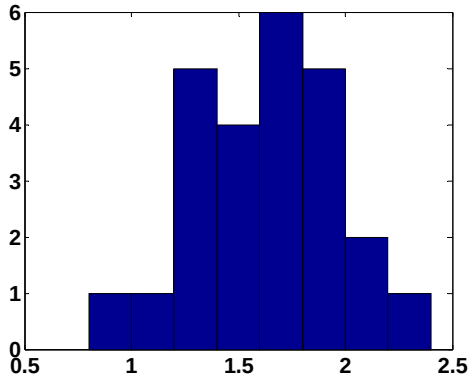
lower	upper	Frequency
0.8	1	1
1	1.2	1
1.2	1.4	5
1.4	1.6	4
1.6	1.8	6
1.8	2	5
2	2.2	2
2.2	2.4	1

The histogram can then be constructed as



Here is how the problem would be answered using MATLAB's built-in functions:

```
clear, clc
y=[0.9 1.32 1.96 1.85 2.29 1.42 1.35 1.47 1.74 1.82 ...
  1.3 1.47 1.92 1.65 2.06 1.55 1.95 1.35 1.78 2.14 ...
  1.63 1.66 1.05 1.71 1.27];
binmids=[0.9:0.2:2.3];
hist(y,binmids)
```



14.3 The data can be sorted in ascending order and tabulated as

i	y	$(y_i - \bar{y})^2$
1	28.15	2.829605
2	28.55	1.64389
3	28.65	1.397462
4	28.75	1.171033
5	29.05	0.611747
6	29.15	0.465319
7	29.25	0.33889
8	29.25	0.33889
9	29.35	0.232462
10	29.35	0.232462
11	29.45	0.146033
12	29.65	0.033176
13	29.65	0.033176
14	**29.65**	**0.033176**
15	**29.65**	**0.033176**
16	29.75	0.006747
17	29.75	0.006747
18	29.85	0.000319
19	30.15	0.101033
20	30.15	0.101033
21	30.25	0.174605
22	30.45	0.381747
23	30.45	0.381747
24	30.55	0.515319
25	30.65	0.66889
26	30.85	1.036033
27	31.25	2.010319
28	33.65	14.57603
Σ	**835.3**	**29.50107**

(a) $\bar{y} = \dfrac{835.3}{28} = 29.83214$

(b) median = 29.65

(c) mode = 29.65

(d) range = maximum – minimum = 33.65 – 28.15 = 5.5

(e) $s_y = \sqrt{\dfrac{29.50107}{28-1}} = 1.045291$

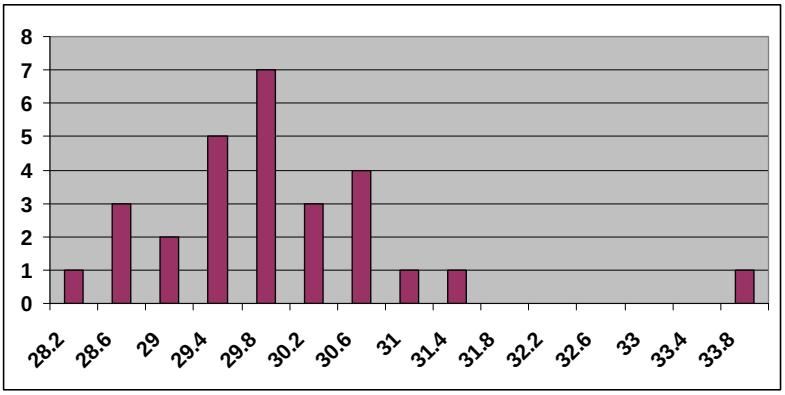
(f) $s_y^2 = 1.045291^2 = 1.092632$

(g) $\text{c.v.} = \dfrac{1.045291}{29.83214} \times 100\% = 3.50\%$

(h) The data can be sorted and grouped.

Lower	Upper	Midpoint	Frequency
28	28.4	28.2	1
28.4	28.8	28.6	3
28.8	29.2	29	2
29.2	29.6	29.4	5
29.6	30	29.8	7
30	30.4	30.2	3
30.4	30.8	30.6	4
30.8	31.2	31	1
31.2	31.6	31.4	1
31.6	32	31.8	
32	32.4	32.2	
32.4	32.8	32.6	
32.8	33.2	33	
33.2	33.6	33.4	
33.6	34	33.8	1

The histogram can then be constructed as

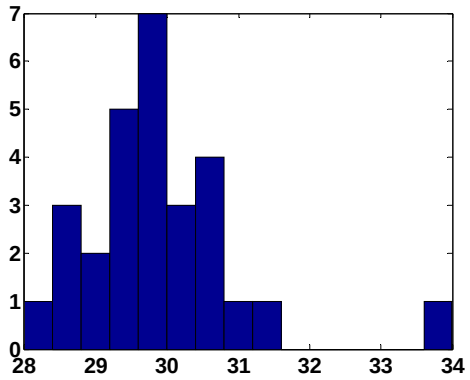


(i) 68% of the readings should fall between $\bar{y} - s_y$ and $\bar{y} + s_y$. That is, between 29.83214 – 1.04529 = 28.78685 and 29.83214 + 1.04529 = 30.87743. Twenty-two values fall between these bounds which is equal to 22/28 = 78.6% of the values which is somewhat higher than the expected value of 68% for the normal distribution.

Here is a script showing how the problem would be answered using MATLAB's built-in functions:

```
clear,clf,clc,format compact
y=[29.65 28.55 28.65 30.15 29.35 29.75 29.25 30.65 28.15 29.85 ...
29.05 30.25 30.85 28.75 29.65 30.45 29.15 30.45 33.65 29.35 ...
29.75 31.25 29.45 30.15 29.65 30.55 29.65 29.25];
meany=mean(y)
mediany=median(y)
modey=mode(y)
range=max(y)-min(y)
stddevy=std(y)
variancey=var(y)
cv=stddevy/meany
```

```
ny=(34-28)/0.4-1;
binmids=linspace(28+0.4,34-0.4,ny);
hist(y,binmids)

meany =
   29.8321
mediany =
   29.6500
modey =
   29.6500
range =
    5.5000
stddevy =
    1.0453
variancey =
    1.0926
cv =
    0.0350
```



14.4 The sum of the squares of the residuals for this case can be written as

$$S_r = \sum_{i=1}^{n} \left(y_i - a_1 x_i \right)^2$$

The partial derivative of this function with respect to the single parameter a_1 can be determined as

$$\frac{\partial S_r}{\partial a_1} = -2 \sum \left[(y_i - a_1 x_i) x_i \right]$$

Setting the derivative equal to zero and evaluating the summations gives

$$0 = \Sigma y_i x_i - a_1 \Sigma x_i^2$$

which can be solved for

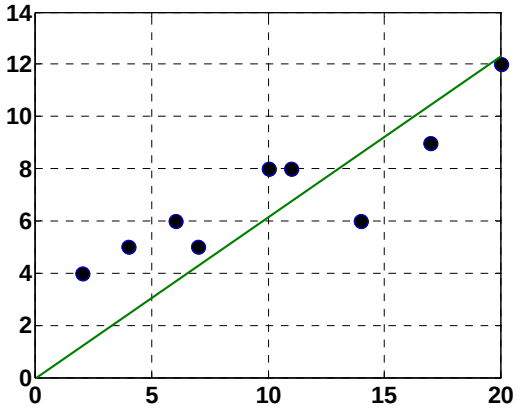
$$a_1 = \frac{\sum y_i x_i}{\sum x_i^2}$$

So the slope that minimizes the sum of the squares of the residuals for a straight line with a zero intercept is merely the ratio of the sum of the dependent variables (y) times the sum of the independent variables (x) over the sum of the independent variables squared (x^2).

The following M-file determines the best-fit slope and plots the resulting line,

```
function [a] = linregrzero(x,y)
% linregrzero: linear regression curve fitting with zero intercept
%   [a, r2] = linregr(x,y): Least squares fit of straight
%             line to data with zero intercept
% input:
%   x = independent variable
%   y = dependent variable
% output:
%   a = slope

n = length(x);
if length(y)~=n, error('x and y must be same length'); end
x = x(:); y = y(:);    % convert to column vectors
sx2 = sum(x.*x); sxy = sum(x.*y);
a = sxy/sx2;
% create plot of data and best fit line
xp = linspace(0,max(x),2);
yp = a*xp;
plot(x,y,'o',xp,yp)
grid on
```

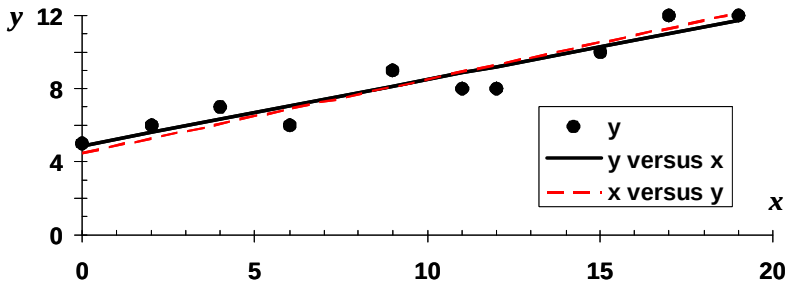
```
>> x=[2 4 6 7 10 11 14 17 20];
>> y=[4 5 6 5 8 8 6 9 12];
>> a1=linregrzero(x,y)

a1 =
    0.6144
```



14.5 The results can be summarized as

	y* versus *x	***x* versus *y***
Best fit equation	$y = 4.88812 + 0.3591455\,x$	$x = -11.1349 + 2.486137\,y$
Standard error	0.851097	2.23927
Correlation coefficient	0.892885	0.892885

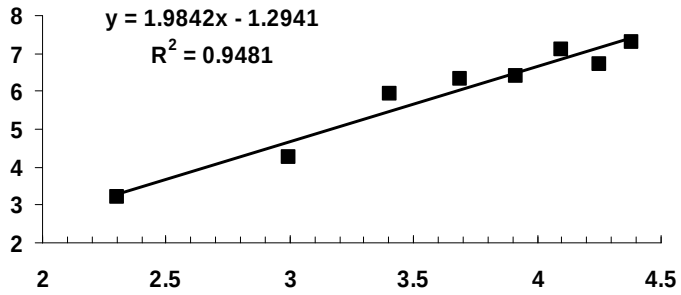
We can also plot both lines on the same graph



Thus, the "best" fit lines and the standard errors differ. This makes sense because different errors are being minimized depending on our choice of the dependent (ordinate) and independent (abscissa) variables. In contrast, the correlation coefficients are identical since the same amount of uncertainty is explained regardless of how the points are plotted.

14.6 The data can be transformed, plotted and fit with a straight line

v, m/s	F, N	ln v	ln F
10	25	2.302585	3.218876
20	70	2.995732	4.248495
30	380	3.401197	5.940171
40	550	3.688879	6.309918
50	610	3.912023	6.413459
60	1220	4.094345	7.106606
70	830	4.248495	6.721426
80	1450	4.382027	7.279319



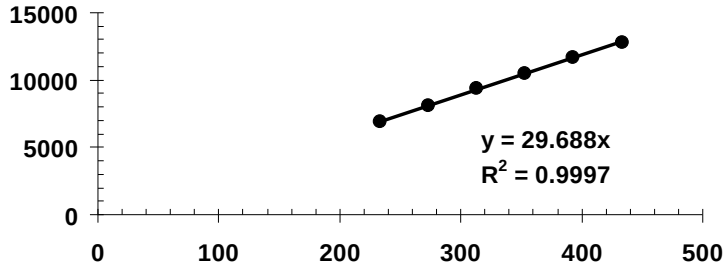
The least-squares fit is

$$\ln y = 1.9842 \ln x - 1.2941$$

The exponent is 1.9842 and the leading coefficient is $e^{-1.2941} = 0.274137$. Therefore, the result is the same as when we used common or base-10 logarithms:

$$y = 0.274137 x^{1.9842}$$

14.7 Linear regression with a zero intercept gives [note that T(K) = T(°C) + 273.15].



Thus, the fit is

$$p = 29.688T$$

Using the ideal gas law

$$R = \left(\frac{p}{T}\right)\frac{V}{n}$$

For our fit, $p/T = 29.688$. For nitrogen,

$$n = \frac{1 \text{ kg}}{28 \text{ g/mole}}$$

Therefore,

$$R = 29.728\left(\frac{10}{10^3 / 28}\right) = 8.31264$$

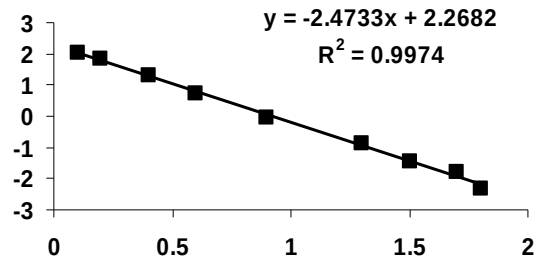
This is close to the standard value of 8.314 J/gmole.

14.8 The function can be linearized by dividing it by x and taking the natural logarithm to yield

$$\ln(y / x) = \ln \alpha_4 + \beta_4 x$$

Therefore, if the model holds, a plot of $\ln(y/x)$ versus x should yield a straight line with an intercept of $\ln\alpha_4$ and a slope of β_4.

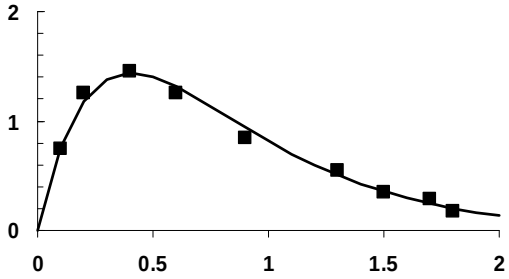
x	y	$\ln(y/x)$
0.1	0.75	2.014903
0.2	1.25	1.832581
0.4	1.45	1.287854
0.6	1.25	0.733969
0.9	0.85	-0.05716
1.3	0.55	-0.8602
1.5	0.35	-1.45529
1.7	0.28	-1.80359
1.8	0.18	-2.30259



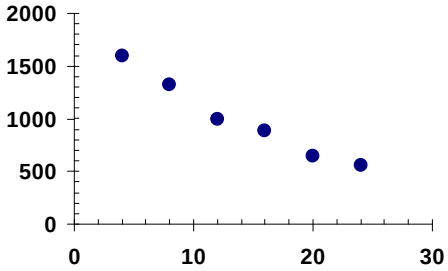
Therefore, $\beta_4 = -2.4733$ and $\alpha_4 = e^{2.2682} = 9.661786$, and the fit is

$$y = 9.661786xe^{-2.4733x}$$

This equation can be plotted together with the data:



14.9 (*a*) The data can be plotted

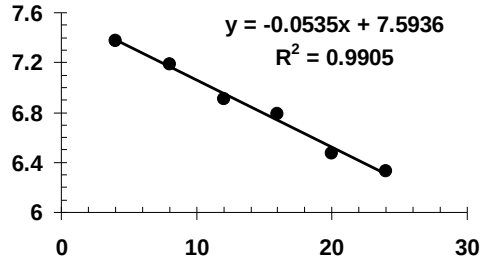


The plot indicates that the data is somewhat curvilinear. An exponential model (i.e., a semi-log plot) is the best choice to linearize the data. This conclusion is based on

- A power model does not result in a linear plot
- Bacterial decay is known to follow an exponential model
- The exponential model by definition will not produce negative values.

The exponential fit can be determined as

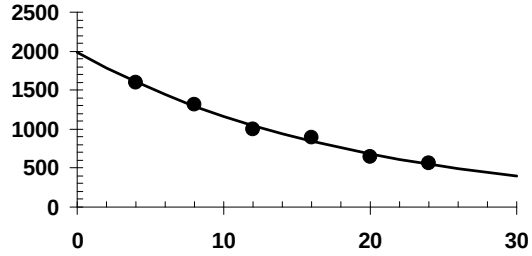
t (hrs)	*c* (CFU/100 mL)	ln *c*
4	1600	7.377759
8	1320	7.185387
12	1000	6.907755
16	890	6.791221
20	650	6.476972
24	560	6.327937



Therefore, the coefficient of the exponent (β_1) is –0.0535 and the lead coefficient (α_1) is $e^{7.5936}$ = 1985.437, and the fit is

$$c = 1985.437e^{-0.0535t}$$

Consequently the concentration at t = 0 is 1978.63 CFU/100 ml. Here is a plot of the fit along with the original data:



(*b*) The time at which the concentration will reach 200 CFU/100 mL can be computed as

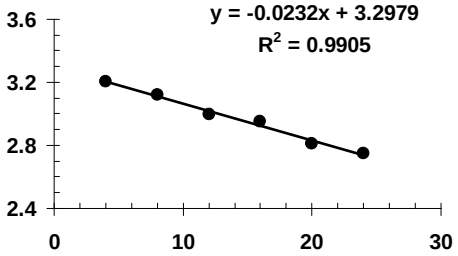
$$200 = 1985.437\, e^{-0.0535t}$$

$$\ln\left(\frac{200}{1985.437}\right) = -0.0535t$$

$$t = \frac{\ln\left(\frac{200}{1985.437}\right)}{-0.0535} = 42.8973 \text{ d}$$

14.10 (a) The exponential fit can be determined with the base-10 logarithm as

t **(hrs)**	***c*** **(CFU/100 mL)**	**log *c***
4	1600	3.20412
8	1320	3.120574
12	1000	3
16	890	2.94939
20	650	2.812913
24	560	2.748188



Therefore, the coefficient of the exponent (β_5) is –0.0232 the lead coefficient (α_5) is $10^{3.2979}$ = 1985.437, and the fit is

$c = 1985.437\ (10)^{-0.0232t}$

Consequently the concentration at t = 0 is 1985.437 CFU/100 ml.

(b) The time at which the concentration will reach 200 CFU/100 mL can be computed as

$$200 = 1985.437(10)^{-0.0232t}$$

$$\log_{10}\left(\frac{200}{1985.437}\right) = -0.0232t$$

$$t = \frac{\log_{10}\left(\frac{200}{1985.437}\right)}{-0.0232} = 42.8973 \text{ d}$$

Thus, the results are identical to those obtained with the base-e model. The relationship between β_1 and β_5 can be developed as in

$$e^{-\beta_1 t} = 10^{-\beta_5 t}$$

Take the natural log of this equation to yield

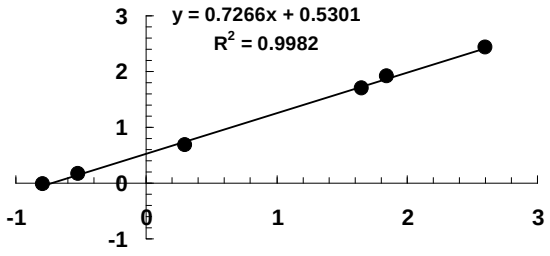
$$-\beta_1 t = -\beta_5 t \ln 10$$

or $\beta_1 = 2.302585\ \beta_5$.

14.11 A power fit can be determined by taking the common logarithm of the data,

Animal	Mass (kg)	Metabolism (watts)	log(Mass)	log(Met)
Cow	400	270	2.6021	2.4314
Human	70	82	1.8451	1.9138
Sheep	45	50	1.6532	1.6990
Hen	2	4.8	0.3010	0.6812
Rat	0.3	1.45	-0.5229	0.1614
Dove	0.16	0.97	-0.7959	-0.0132

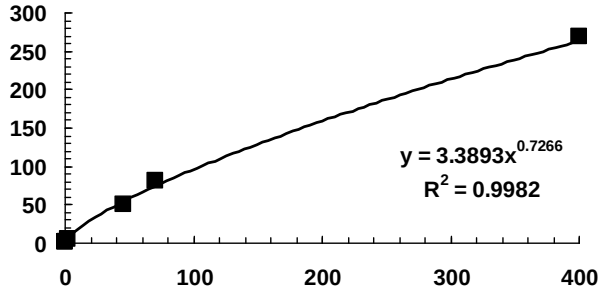
Linear regression gives



Therefore, the power is b = 0.7266 and the lead coefficient is $a = 10^{0.5301} = 3.389$, and the fit is

Metabolism = 3.389 Mass$^{0.7266}$

Here is a plot of the fit along with the original data:

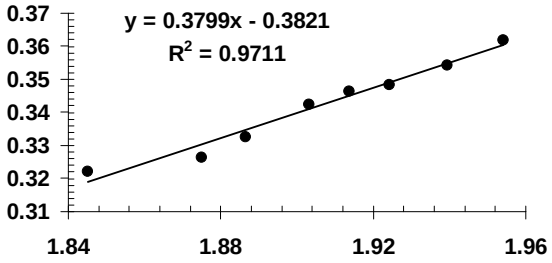


The value of the metabolism for a 200-kg tiger can be estimated as

$$\text{Metabolism} = 3.389\ (200)^{0.7266} = 159.2047\ \text{W}$$

14.12 The power fit can be determined as

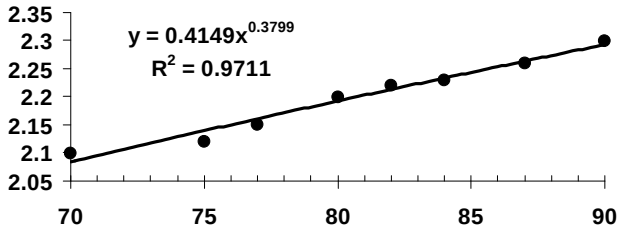
W (kg)	A (m^2)	log W	log A
70	2.10	1.845098	0.322219
75	2.12	1.875061	0.326336
77	2.15	1.886491	0.332438
80	2.20	1.90309	0.342423
82	2.22	1.913814	0.346353
84	2.23	1.924279	0.348305
87	2.26	1.939519	0.354108
90	2.30	1.954243	0.361728



Therefore, the power is $b = 0.3799$ and the lead coefficient is $a = 10^{-0.3821} = 0.4149$, and the fit is

$$A = 0.4149W^{0.3799}$$

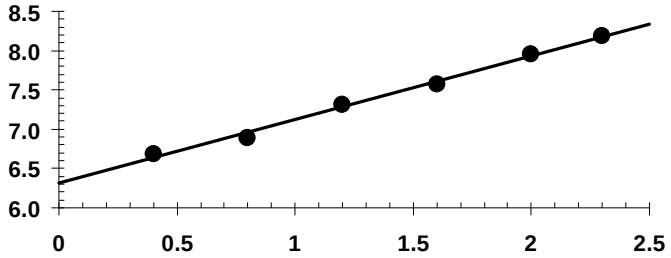
Here is a plot of the fit along with the original data:



The value of the surface area for a 95-kg person can be estimated as

$$A = 0.4149(95)^{0.3799} = 2.34\ \text{m}^2$$

14.13 A semi-log plot can be developed by plotting the natural log versus *x*. As expected, both the data and the best-fit line are linear when plotted in this way.



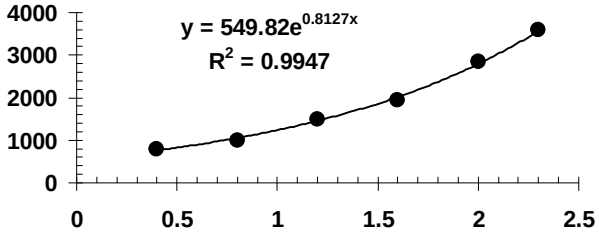
We regress ln(*y*) versus *x* to give

$$\ln y = 6.309582 + 0.812728\,x$$

Therefore, $\alpha_1 = e^{6.309582} = 549.8153$ and $\beta_1 = 0.812728$, and the exponential model is

$$y = 549.8153e^{0.812728x}$$

The model and the data can be plotted as



14.14 The equation can be linearized by inverting it to yield

$$\frac{1}{k} = \frac{c_s}{k_{\max}}\frac{1}{c^2} + \frac{1}{k_{\max}}$$

Consequently, a plot of $1/k$ versus $1/c^2$ should yield a straight line with an intercept of $1/k_{\max}$ and a slope of $c_s/k_{\max}$

c, mg/L	*k*, /d	$1/c^2$	$1/k$	$1/c^2 \times 1/k$	$(1/c^2)^2$
0.5	1.1	4.000000	0.909091	3.636364	16
0.8	2.5	1.562500	0.400000	0.625000	2.441406
1.5	5.3	0.444444	0.188679	0.083857	0.197531
2.5	7.6	0.160000	0.131579	0.021053	0.0256
4	8.9	0.062500	0.112360	0.007022	0.003906
	Sum →	6.229444	1.741709	4.373296	18.66844

The slope and the intercept can be computed as

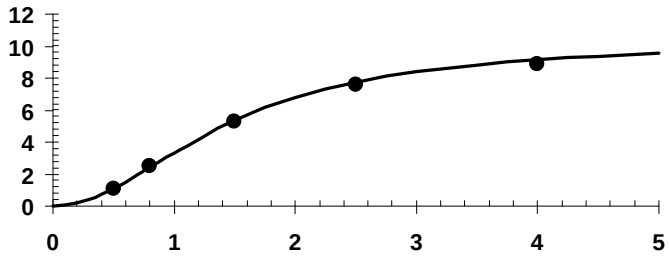
$$a_1 = \frac{5(4.373296) - 6.229444(1.741709)}{5(18.66844) - (6.229444)^2} = 0.202005$$

$$a_0 = \frac{1.741709}{5} - 0.202005\frac{6.229444}{5} = 0.096666$$

Therefore, k_{max} = 1/0.096666 = 10.34493 and c_s = 10.34493(0.202005) = 2.08973, and the fit is

$$k = \frac{10.34493c^2}{2.08973 + c^2}$$

This equation can be plotted together with the data:



The equation can be used to compute

$$k = \frac{10.34493(2)^2}{2.08973 + (2)^2} = 6.795002$$

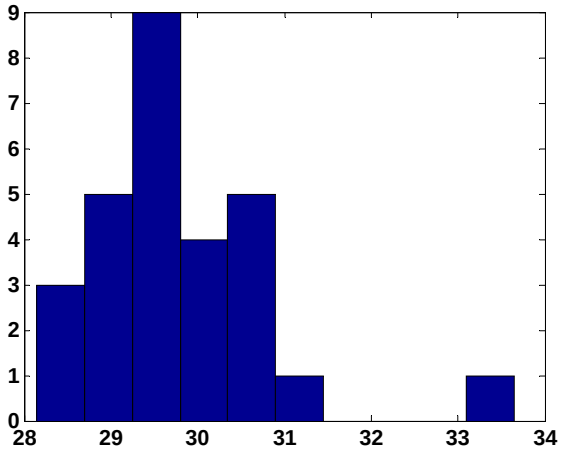
14.15

```
function stats(x)
% stats: simple descriptive statistics and histogram
%   stats(x): computes simple descriptive statistics
%             and generates a histogram for the values
%             in a vector
% input:
%   x = vector of values
% output:
%   the function does not return any values, but displays the
%   following statistics: number of values, mean, median, mode,
%   range, standard deviation, variance,
%          and coefficient of variation
if length(x)<=1, error('Vector must hold at least 2 values'); end
fprintf('number = %5d\n', length(x))
fprintf('mean = %8.4g\n', mean(x))
fprintf('median = %8.4g\n', median(x))
fprintf('mode = %8.4g\n', mode(x))
fprintf('range = %8.4g\n', max(x)-min(x))
fprintf('standard deviation = %8.4g\n', std(x))
fprintf('variance = %8.4g\n', var(x))
fprintf('coefficient of variation = %8.4g\n', std(x)/mean(x))
hist(x)
```

Script using stats to generate the solution to Prob. 14.3:

```
clear,clf,clc,format compact
y=[29.65 28.55 28.65 30.15 29.35 29.75 29.25 30.65 28.15 29.85 ...
29.05 30.25 30.85 28.75 29.65 30.45 29.15 30.45 33.65 29.35 ...
29.75 31.25 29.45 30.15 29.65 30.55 29.65 29.25];
stats(y)
```

```
number =    28
mean =    29.83
median =    29.65
mode =    29.65
range =      5.5
standard deviation =    1.045
variance =    1.093
coefficient of variation =  0.03504
```



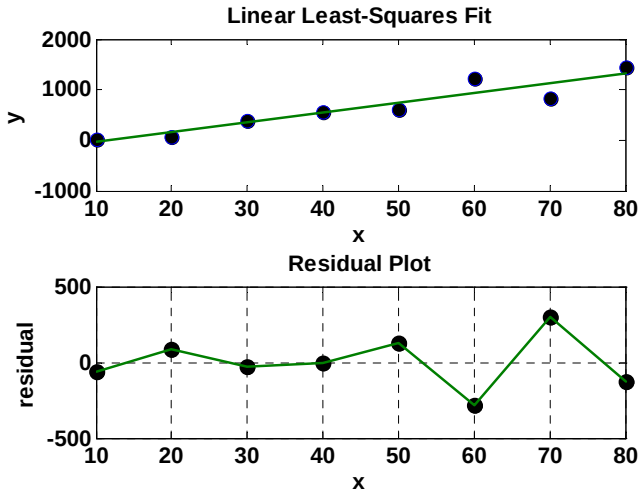
14.16

```
function [a, r2, syx] = linregr2(x,y)
% [a, r2, syx] = linregr(x,y):
%   Least squares fit of a straight line to data
%   by solving the normal equations.
% input:
%   x = independent variable
%   y = dependent variable
% output:
%   a = vector of slope, a(1), and intercept, a(2)
%   r2 = coefficient of determination
%   syx = standard error of the estimate

n = length(x);
if length(y)~=n, error('x and y must be same length'); end
x = x(:); y = y(:);    % convert to column vectors
sx = sum(x); sy = sum(y);
sx2 = sum(x.*x); sxy = sum(x.*y); sy2 = sum(y.*y);
a(1) = (n*sxy-sx*sy)/(n*sx2-sx^2);
a(2) = sy/n-a(1)*sx/n;
syx=sqrt(sum((y-a(1)*x-a(2)).^2)/(n-2));
r2 = ((n*sxy-sx*sy)/sqrt(n*sx2-sx^2)/sqrt(n*sy2-sy^2))^2;
% create plot of data and best fit line
xp = linspace(min(x),max(x),2);
yp = a(1)*xp+a(2);
res=a(1)*x+a(2)-y;
subplot(2,1,1);plot(x,y,'o',xp,yp)
xlabel('x'),ylabel('y'),title('Linear Least-Squares Fit')
subplot(2,1,2);plot(x,res,'o',x,res)
xlabel('x'),ylabel('residual'),title('Residual Plot')
grid on
```

Test application to data from Examples 14.2 and 14.3,

```
>> x=[10 20 30 40 50 60 70 80];
```

```
>> y=[25 70 380 550 610 1220 830 1450];
>> [a, r2, syx] = linregr2(x,y)

a =
   19.4702 -234.2857
r2 =
    0.8805
syx =
  189.7885
```



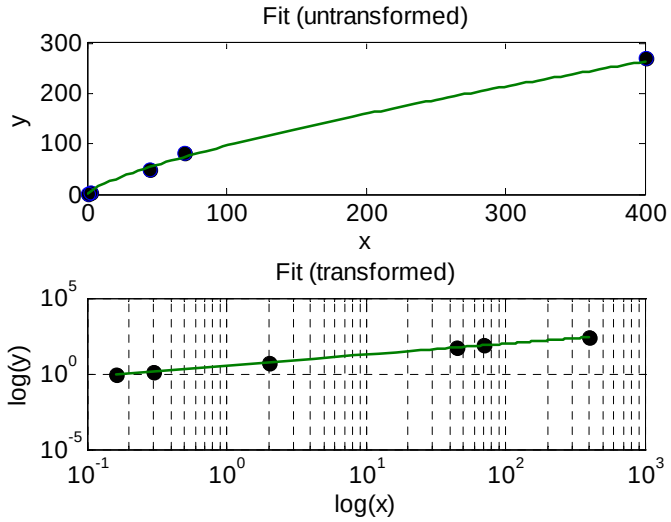
14.17

```
function [b, r2] = powerfit(x,y)
% linregr: linear regression curve fitting
%   [b, r2] = powerfit(x,y): Least squares fit of straight
%             line to log-transformed data
% input:
%   x = independent variable
%   y = dependent variable
% output:
%   b = vector of coefficient, b(1), and exponent, b(2)
%   r2 = coefficient of determination

n = length(x);
if length(y)~=n, error('x and y must be same length'); end
x = x(:); y = y(:);
xl=log10(x);yl=log10(y);
sx = sum(xl); sy = sum(yl);
sx2 = sum(xl.*xl); sxy = sum(xl.*yl); sy2 = sum(yl.*yl);
% compute slope and intercept
a(1) = (n*sxy-sx*sy)/(n*sx2-sx^2);
a(2) = sy/n-a(1)*sx/n;
% compute coefficient and exponent
b(1) = 10^a(2);
b(2) = a(1);
St=sum((mean(y)-y).^2)
ypred = b(1)*x.^b(2);
Sr=sum((ypred-y).^2)
r2 = (St - Sr)/St;
% create plots of data and best fit line
xp = linspace(min(x),max(x));
yp = b(1)*xp.^b(2);
subplot(2,1,1);plot(x,y,'o',xp,yp)
plot(x,y,'o',xp,yp)
```

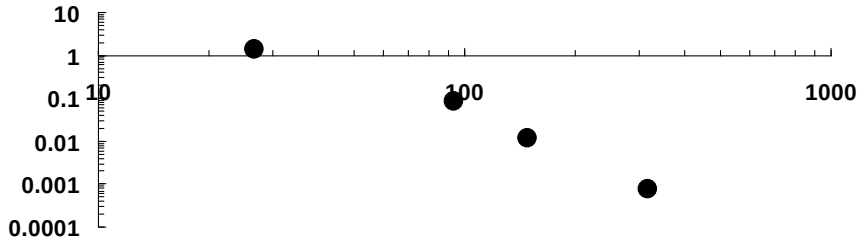
```
xlabel('x'),ylabel('y'),title('Fit (untransformed)')
subplot(2,1,2);loglog(x,y,'o',xp,yp)
xlabel('log(x)'),ylabel('log(y)'),title('Fit (transformed)')
grid on
```

Application to Prob. 14.11.

```
>> mass=[400 70 45 2 0.3 0.16];
>> metabolism=[270 82 50 4.8 1.45 0.97];
>> [b,r2]=powerfit(mass,metabolism)

b =
    3.3893    0.7266
r2 =
    0.9978
```



14.18 A log-log plot of μ versus T suggests a linear relationship.



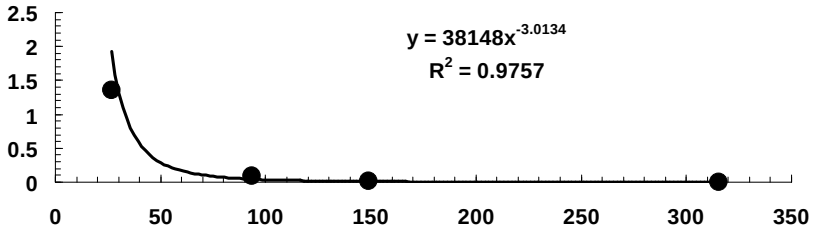
We regress $\log_{10}\mu$ versus $\log_{10}T$ to give

$$\log_{10}\mu = 4.581471 - 3.01338\log_{10}T \qquad (r^2 = 0.975703)$$

Therefore, $\alpha_2 = 10^{4.581471} = 38{,}147.94$ and $\beta_2 = -3.01338$, and the power model is

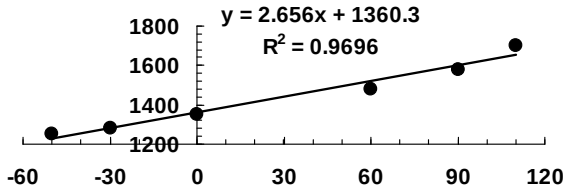
$$\mu = 38{,}147.94T^{-3.01338}$$

The model and the data can be plotted on untransformed scales as



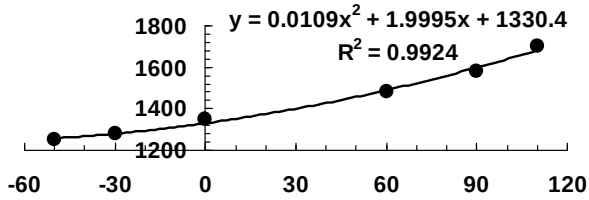
14.19 We can first try a linear fit:

```
clear,clc,clf,format short g
T=[-50 -30 0 60 90 110];
c=[1250 1280 1350 1480 1580 1700];
a = polyfit(T,c,1)
Tp=linspace(min(T),max(T));
cp = polyval(a, Tp);
plot(Tp,cp,T,c,'o')
```



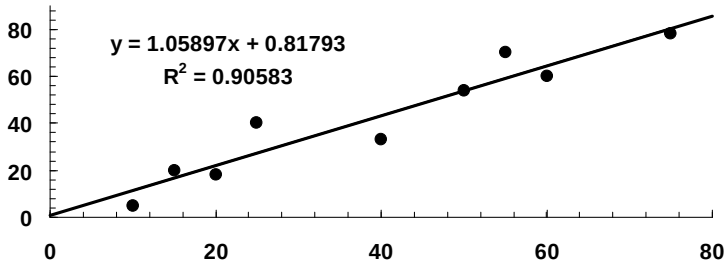
As shown, the fit line is somewhat lacking. Therefore, we can use polynomial regression to fit a parabola

```
a = polyfit(T,c,2)
cp = polyval(a, Tp);
plot(Tp,cp,T,c,'o')
```



This fit seems adequate in that it captures the general trend of the data. Note that a slightly better fit can be attained with a cubic polynomial, but the improvement is marginal.

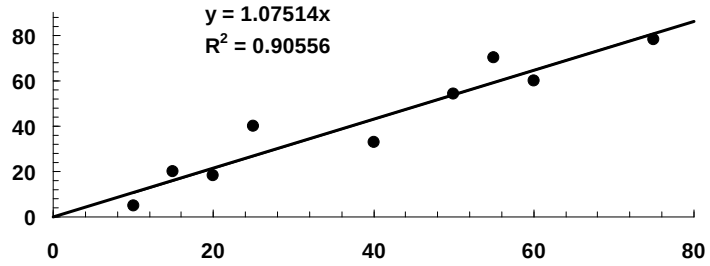
14.20 (a) The linear fit is



The tensile strength at $t = 32$ can be computed as

$$y = 1.05897(32) + 0.81793 = 34.7048913$$

(b) A straight line with zero intercept can be fit as



For this case, the tensile strength at $t = 32$ can be computed as

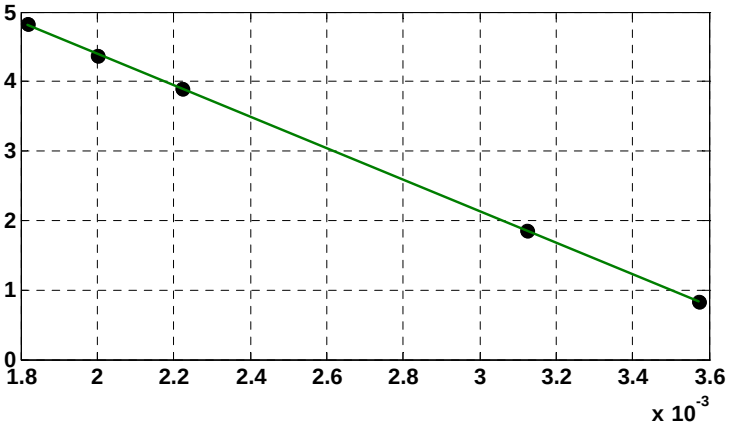
$$y = 1.07514(32) = 34.40452$$

14.21 The equation can be linearized,

$$\ln\left(\frac{-dA/dt}{A}\right) = \ln k_{01} - \frac{E_1}{RT}$$

This indicates that a plot of $\ln(-(dA/dt)/A)$ versus $1/T$ should have an intercept of $\ln k_{01}$ and a slope of E_1/R.

```
clear,clc,clf,format short g
dAdt=[460 960 2485 1600 1245];
A=[200 150 50 20 10];
T=[280 320 450 500 550];
y=log(dAdt./A);
TI=1./T;
[a,r2]=linregr(TI,y)
k01=exp(a(2))
E1=-a(1)*0.00198

a =
      -2268.3       8.9386
r2 =
      0.99995
k01 =
       7620.5
E1 =
       4.4913
```



Therefore, k_{01} = 7,620.5 and E_1 = 4.4913.

14.22 The standard errors can be computed via Eq. 14.19

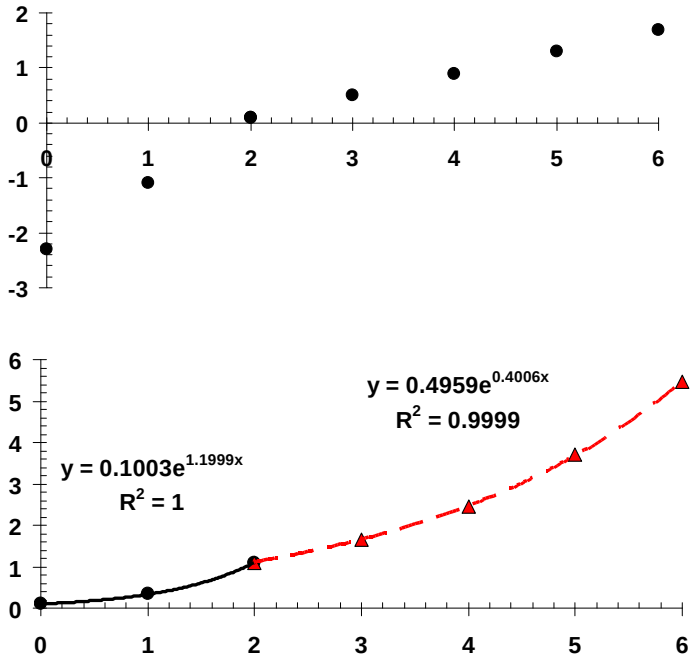
$$s_{y/x} = \sqrt{\frac{S_r}{n-p}}$$

n = 15

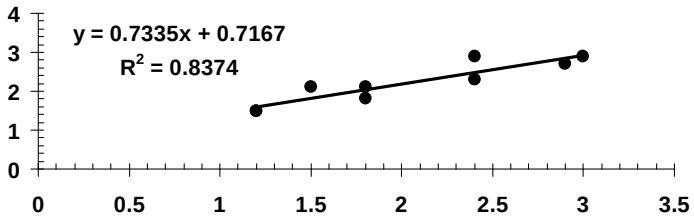
	Model A	Model B	Model C
S_r	135	105	100
Number of model parameters fit (p)	2	3	5
$s_{y/x}$	3.222517	2.95804	3.162278

Thus, Model B seems best because its standard error is lower.

14.23 A plot of the natural log of cells versus time indicates two straight lines with a sharp break at 2. Each range can be fit separately with the exponential model as shown in the second plot.



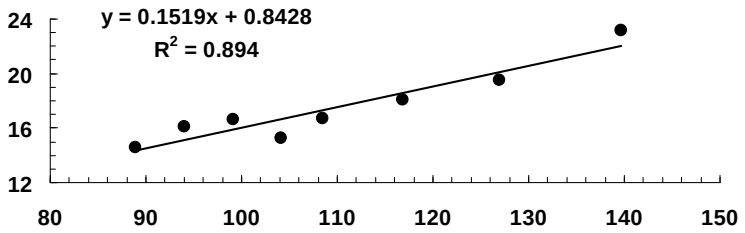
14.24 (a) and (b) Simple linear regression can be applied to yield the following fit



(c) The minimum lane width corresponding to a bike-car distance of 2 m can be computed as

$$y = 0.7335(1.8) + 0.7167 = 2.037 \text{ m}$$

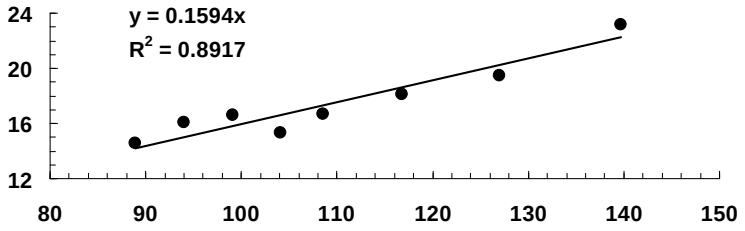
14.25 (a) and (b) Simple linear regression can be applied to yield the following fit



(c) The flow corresponding to the precipitation of 120 cm can be computed as

$$Q = 0.1519(120) + 0.8428 = 19.067$$

(d) We can redo the regression, but with a zero intercept



Thus, the model is

$$Q = 0.1594P$$

where Q = flow and P = precipitation. Now, if there are no water losses, the maximum flow, Q_m, that could occur for a level of precipitation should be equal to the product of the annual precipitation and the drainage area. This is expressed by the following equation.

$$Q_m = A(\text{km}^2)P\left(\frac{\text{cm}}{\text{yr}}\right)$$

For an area of 1,100 km^2 and applying conversions so that the flow has units of m^3/s

$$Q_m = 1{,}100\ \text{km}^2 \times P\left(\frac{\text{cm}}{\text{yr}}\right)\frac{10^6\,\text{m}^2}{\text{km}^2}\frac{1\ \text{m}}{100\ \text{cm}}\frac{\text{d}}{86{,}400\ \text{s}}\frac{\text{yr}}{365\ \text{d}}$$

Collecting terms gives

$$Q_m = 0.348808P$$

Using the slope from the linear regression with zero intercept, we can compute the fraction of the total flow that is lost to evaporation and other consumptive uses can be computed as

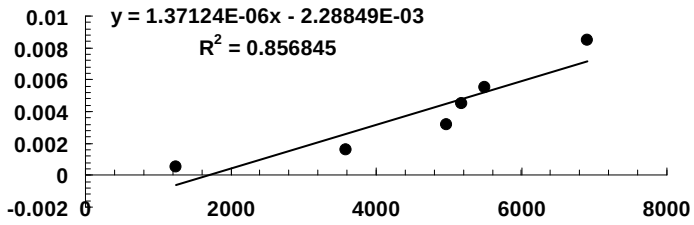
$$F = \frac{0.348808 - 0.1594}{0.348808} = 0.543$$

14.26 First, we can determine the stress

$$\sigma = \frac{25000}{10.65} = 2,347.418$$

We can then try to fit the data to obtain a mathematical relationship between strain and stress. First, we can try linear regression:

```
clear,clc,clf,format short g
strain=[0.0032 0.0045 0.0055 0.0016 0.0085 0.0005];
stress=[4970 5170 5500 3590 6900 1240];
a = polyfit(stress,strain,1)
stressp=linspace(min(stress),max(stress));
strainp = polyval(a, stressp);
plot(stressp,strainp,stress,strain,'o')
```

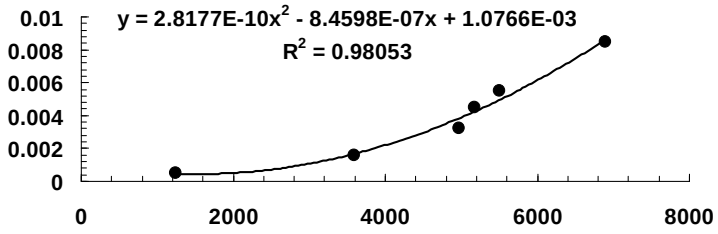


We can use this model to compute the strain and deflection

$\varepsilon = 1.37124\times10^{-6}(2347.4178) - 2.28849\times10^{-3} = 9.3038\times10^{-4}$
$\Delta L = 9.3038\times10^{-4}(9) = 0.0083735$ m

This is not a particularly good fit. In particular, the negative intercept is physically unrealistic. We therefore try a best-fit parabola:

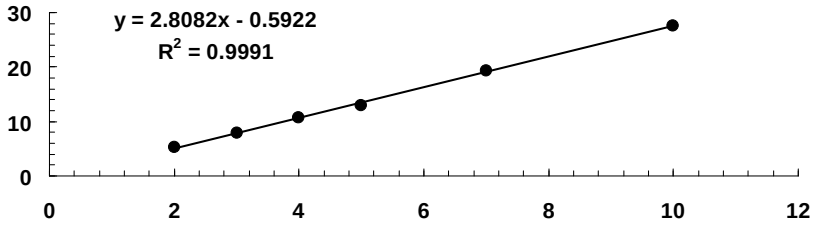
```
a = polyfit(stress,strain,2)
strainp = polyval(a, stressp);
plot(stressp,strainp,stress,strain,'o')
```



We can use this model to compute the strain and deflection

$\varepsilon = 2.8177\times10^{-10}(2347.4178)^2 - 8.4598\times10^{-7}(2347.4178) + 1.0766\times10^{-3} = 6.4341\times10^{-4}$
$\Delta L = 6.4341\times10^{-4}$ (9) = 0.0057907 m

14.27 (a) The linear fit is

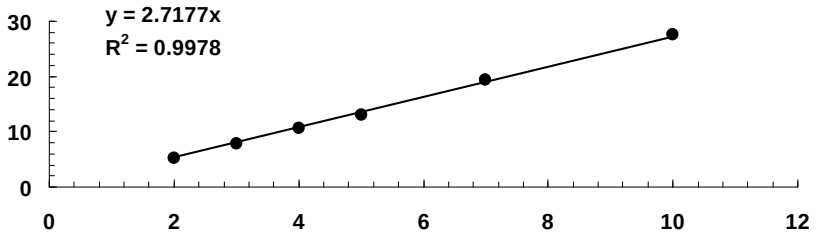


The current for a voltage of 3.5 V can be computed as

$y = 2.8082(3.5) - 0.5922 = 9.2364$

Both the graph and the r^2 indicate that the fit is good.

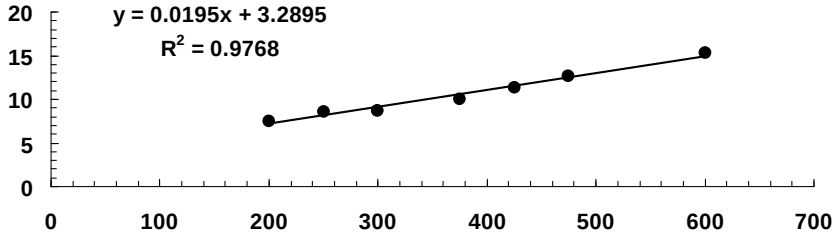
(b) A straight line with zero intercept can be fit as



For this case, the current at $V = 3.5$ can be computed as

$y = 2.7177(3.5) = 9.512$

14.28 Linear regression yields



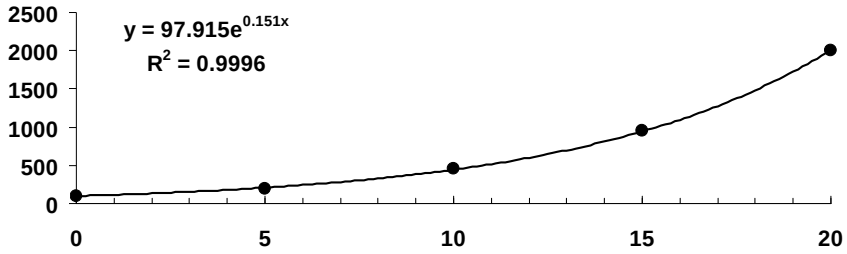
The percent elongation for a temperature of 400 can be computed as

$$\%\ \text{elongation} = 0.0195(400) + 3.2895 = 11.072$$

14.29 The fit of the exponential model can be developed with the following script:

```
clear,clc,clf,format short g
t=[0 5 10 15 20];
p=[100 200 450 950 2000];
a = polyfit(t,log(p),1)
coef=exp(a(2))
tp=linspace(min(t),max(t));
pp = coef*exp(a(1)*tp);
plot(tp,pp,t,p,'o')

a =
      0.15099       4.5841
```

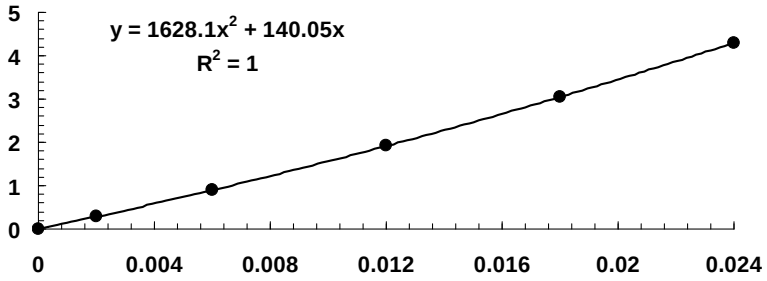
```
coef =
       97.915
```



The model can be used to predict the population 5 years in the future as

$p = 97.915e^{0.15099}(25) = 4{,}268$

14.30 We fit a number of curves to this data and obtained the best fit with a second-order polynomial with zero intercept



Therefore, the best-fit curve is

$u = 1628.1y^2 + 140.05y$

We can differentiate this function

$$\frac{du}{dy} = 3256.2y + 140.05$$

Therefore, the derivative at the surface is 140.05 and the shear stress can be computed as $1.8\times10^{-5}(140.05) = 0.002521$ N/m^2.

14.31 We can use transformations to linearize the model as

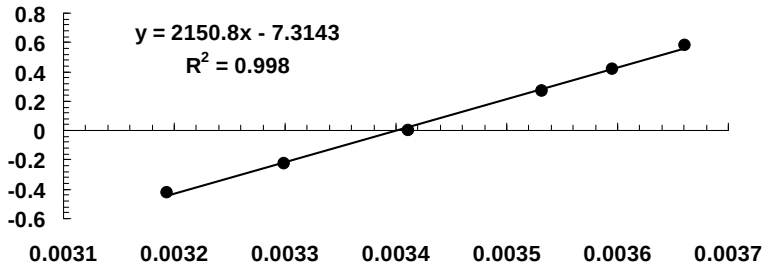
$$\ln \mu = \ln D + B\frac{1}{T_a}$$

Thus, we can plot the natural log of μ versus $1/T_a$ and use linear regression to determine the parameters. Here is the data showing the transformations.

T	μ	T_a	$1/T_a$	$\ln \mu$
0	1.787	273.15	0.003661	0.580538
5	1.519	278.15	0.003595	0.418052
10	1.307	283.15	0.003532	0.267734
20	1.002	293.15	0.003411	0.001998
30	0.7975	303.15	0.003299	-0.22627

40	0.6529	313.15	0.003193	-0.42633

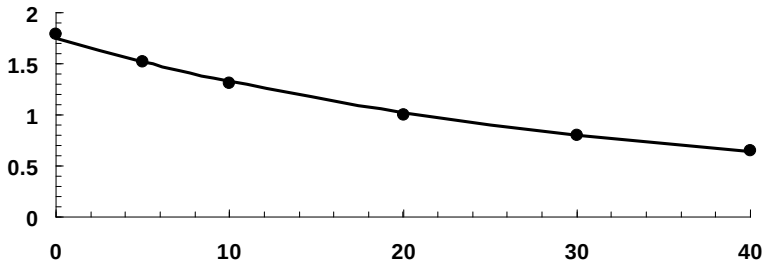
Here is the fit:



Thus, the parameters are estimated as $D = e^{-7.3143} = 6.65941\times10^{-4}$ and $B = 2150.8$, and the Andrade equation is

$$\mu = 6.65941\times10^{-4}\, e^{2150.8/Ta}$$

This equation can be plotted along with the data

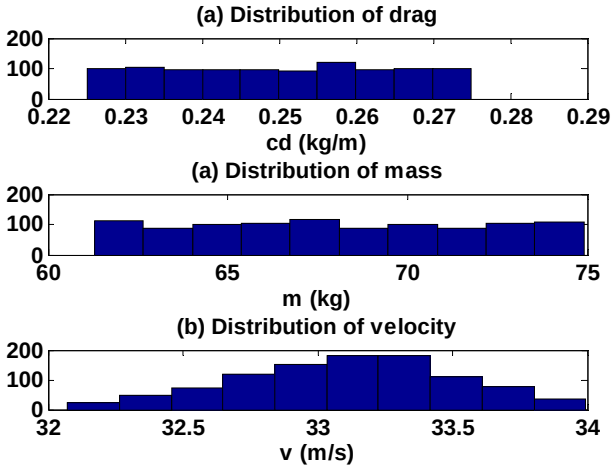


14.32 Script:

```
clear,clc,clf,format short g
n=1000;t=4;g=9.81;
cd=0.25;cdmin=cd-0.1*cd,cdmax=cd+0.1*cd
r=rand(n,1);
cdrand=cdmin+(cdmax-cdmin)*r;
meancd=mean(cdrand),stdcd=std(cdrand)
Deltacd=(max(cdrand)-min(cdrand))/meancd/2*100.
subplot(3,1,1)
hist(cdrand),title('(a) Distribution of drag')
xlabel('cd (kg/m)')
m=68.1;mmin=m-0.1*m,mmax=m+0.1*m
r=rand(n,1);
mrand=mmin+(mmax-mmin)*r;
meanm=mean(mrand),stdm=std(mrand)
Deltam=(max(mrand)-min(mrand))/meanm/2*100.
subplot(3,1,2)
hist(mrand),title('(a) Distribution of mass')
xlabel('m (kg)')

vrand=sqrt(g*mrand./cdrand).*tanh(sqrt(g*cdrand./mrand)*t);
meanv=mean(vrand)
Deltav=(max(vrand)-min(vrand))/meanv/2*100.
subplot(3,1,3)
hist(vrand),title('(b) Distribution of velocity')
xlabel('v (m/s)')
```

Output:

```
cdmin =
        0.225
cdmax =
        0.275
meancd =
      0.25011
stdcd =
     0.014467
Deltacd =
       9.9888
mmin =
        61.29
mmax =
        74.91
meanm =
        68.09
stdm =
       3.9809
Deltam =
        9.998
meanv =
       33.099
Deltav =
       2.9035
```



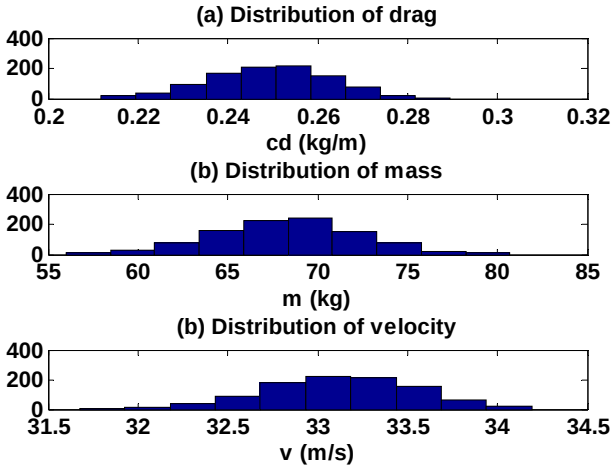
14.33 Script:

```
clear,clc,clf,format short g
n=1000;t=4;g=9.81;
cd=0.25;stdev=0.01443;
r=randn(n,1);
cdrand=cd+stdev*r;
meancd=mean(cdrand),stdevcd=std(cdrand)
cvcd=stdevcd/meancd*100.
subplot(3,1,1)
hist(cdrand),title('(a) Distribution of drag')
xlabel('cd (kg/m)')
m=68.1;stdev=0.05778*m;
r=randn(n,1);
mrand=m+stdev*r;
meanm=mean(mrand),stdevm=std(mrand)
cvm=stdevm/meanm*100.
subplot(3,1,2)
```

```
hist(mrand),title('(b) Distribution of mass')
xlabel('m (kg)')
vrand=sqrt(g*mrand./cdrand).*tanh(sqrt(g*cdrand./mrand)*t);
meanv=mean(vrand),stdevv=std(vrand)
cvv=stdevv/meanv*100.
subplot(3,1,3)
hist(vrand),title('(b) Distribution of velocity')
xlabel('v (m/s)')
```

Output:

```
meancd =
      0.24938
stdevcd =
     0.013615
cvcd =
       5.4594
meanm =
       68.279
stdevm =
       4.0578
cvm =
        5.943
meanv =
       33.125
stdevv =
      0.41512
cvv =
       1.2532
```



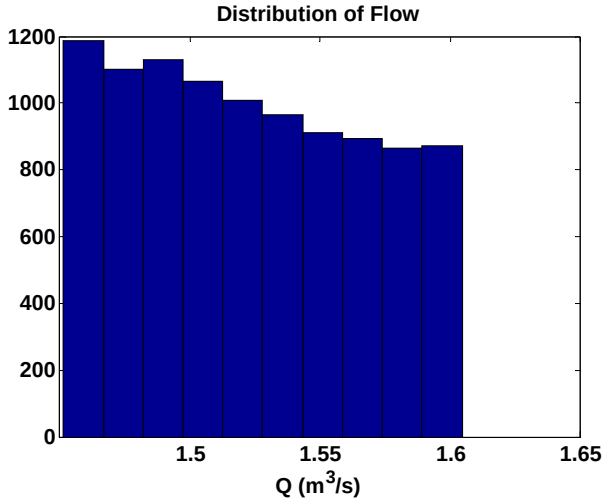
14.34 Script:

```
clear,clc,format short g
n=10000;B=20;H=0.3;
nM=0.03;nMmin=0.027;nMmax=0.033;
S=0.0003;Smin=0.00027;Smax=0.00033;
Q=1./nM.*(B.*H).^(5/3)./(B+2*H).^(2/3).*sqrt(S)
r=rand(n,1);
nMrand=nMmin+(nMmax-nMmin)*r;
Srand=Smin+(Smax-Smin)*r;
Qrand=1./nMrand.*(B.*H).^(5/3)./(B+2*H).^(2/3).*sqrt(Srand);
meanQ=mean(Qrand)
Qmin=min(Qrand)
Qmax=max(Qrand)
```

```
DeltaQ=(max(Qrand)-min(Qrand))/meanQ/2*100.
hist(Qrand),title('Distribution of Flow')
xlabel('Q (m^3/s)')
```

Output:

```
Q =
       1.5221
meanQ =
       1.5231
Qmin =
       1.4513
Qmax =
       1.6044
DeltaQ =
       5.0273
```



14.35 Script:

```
clear,clc,format short g
g=9.81;y0=1;v0=25;theta0=50*pi/180;
n=10000;
xmin=0;xmax=60;
r=rand(n,1);
xrand=xmin+(xmax-xmin)*r;
yrand=tan(theta0)*xrand-g/(2*v0^2*cos(theta0)^2)*xrand.^2+y0;
[ymax,i]=max(yrand)
xmax=xrand(i)
y=@(x) -(tan(theta0)*x-g/(2*v0^2*cos(theta0)^2)*x.^2+y0);
[xmax,fval]=fminbnd(y,0,60)
```

Output:

```
ymax =
       19.693
i =
        2349
xmax =
       31.373
xmax =
       31.371
fval =
      -19.693
```