

Least Squares as a tool for regression

Suppose that we have a set of experimental data that we would like to fit to a curve. Generally, the idea would be to choose a functional form for the curve based on some known theory and then adjust the coefficients to minimize the error. We might be doing a transport experiment where dimensional analysis predicts a power-law relation, $f \sim 1/Re$ or we could be searching for the value of a universal constant, say g .

How about for the sake of an initial example in this course, we choose a known function to generate our data, do the fitting, and then check the result.

Here are some data. The independent variable is:

```
t = Table[i, {i, 0, 2, .5}]
```

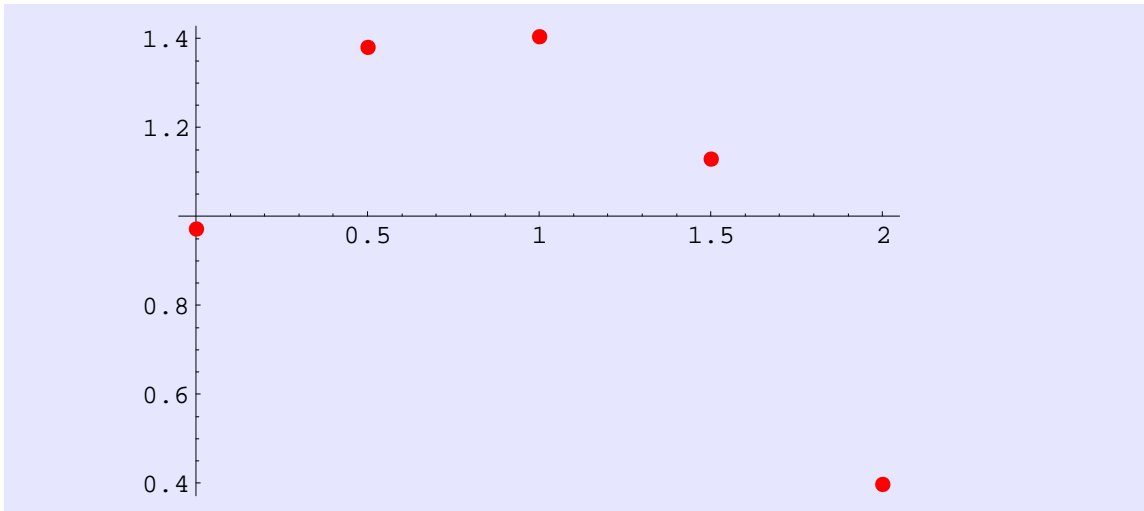
```
{0, 0.5, 1., 1.5, 2.}
```

We make some fake data that has some noise

```
x = Table[Cos[i] + Sin[i] + Random[Real, {- .1, .1}], {i, 0, 2, .5}]
```

```
{0.971397, 1.38058, 1.40375, 1.12953, 0.397554}
```

```
plot1 = ListPlot[Transpose[{t, x}],
  PlotStyle -> {PointSize[.02], RGBColor[1, 0, 0]}]
```



- Graphics -

Now we need to choose a function that will be fit, how about a polynomial.

$$f[t] = a_0 + a_1 t + a_2 t^2$$

If we assume that we know the values of t precisely, then the error for each point, compared to the fit, is given by the residual, $r[i] = x[i] - a_0 + a_1 t[i] + a_2 t[i]^2$.

The "goodness of fit" needs some quantification. Why not use something that looks reasonable, the χ^2 function, $\chi^2 = \text{Sum}[r[i]^2, \{i, 1, n\}]$.

The idea is to choose $\{a_0, a_1, a_2\}$ so that χ^2 has the smallest possible value.

How do we do this? We need to minimize, χ^2 , with respect to the 3 coefficients.

Take the partial derivatives and set the three equations to 0.

```
chisquar =
  Sum[ ((-xx[i] + a0 + a1 tt[i] + a2 tt[i]^2))^2, {i, 0, 2, .5}]
```

$$(a2 \text{tt}(0)^2 + a1 \text{tt}(0) + a0 - \text{xx}(0))^2 + (a2 \text{tt}(0.5)^2 + a1 \text{tt}(0.5) + a0 - \text{xx}(0.5))^2 + \\ (a2 \text{tt}(1.)^2 + a1 \text{tt}(1.) + a0 - \text{xx}(1.))^2 + (a2 \text{tt}(1.5)^2 + a1 \text{tt}(1.5) + a0 - \text{xx}(1.5))^2 + \\ (a2 \text{tt}(2.)^2 + a1 \text{tt}(2.) + a0 - \text{xx}(2.))^2$$

Here we take the derivative with respect to a0.

```
d0 = FullSimplify[D[chisquar, a0]]
```

$$2(5 a0 + a1 (\text{tt}(0) + \text{tt}(0.5) + \text{tt}(1.) + \text{tt}(1.5) + \text{tt}(2.)) + \\ a2 (\text{tt}(0)^2 + \text{tt}(0.5)^2 + \text{tt}(1.)^2 + \text{tt}(1.5)^2 + \text{tt}(2.)^2) - \text{xx}(0) - \text{xx}(0.5) - \text{xx}(1.) - \text{xx}(1.5) - \text{xx}(2.))$$

Here we take the derivative with respect to a1.

```
d1 = FullSimplify[D[chisquar, a1]]
```

$$2(\text{tt}(0) (a0 + \text{tt}(0) (a1 + a2 \text{tt}(0)) - \text{xx}(0)) + \text{tt}(0.5) (a0 + \text{tt}(0.5) (a1 + a2 \text{tt}(0.5)) - \text{xx}(0.5)) + \\ \text{tt}(1.) (a0 + \text{tt}(1.) (a1 + a2 \text{tt}(1.)) - \text{xx}(1.)) + \text{tt}(1.5) (a0 + \text{tt}(1.5) (a1 + a2 \text{tt}(1.5)) - \text{xx}(1.5)) + \\ \text{tt}(2.) (a0 + \text{tt}(2.) (a1 + a2 \text{tt}(2.)) - \text{xx}(2.)))$$

Here we take the derivative with respect to a2.

```
d2 = FullSimplify[D[chisquar, a2]]
```

$$2((a0 + \text{tt}(0) (a1 + a2 \text{tt}(0)) - \text{xx}(0)) \text{tt}(0)^2 + \text{tt}(0.5)^2 (a0 + \text{tt}(0.5) (a1 + a2 \text{tt}(0.5)) - \text{xx}(0.5)) + \\ \text{tt}(1.)^2 (a0 + \text{tt}(1.) (a1 + a2 \text{tt}(1.)) - \text{xx}(1.)) + \text{tt}(1.5)^2 (a0 + \text{tt}(1.5) (a1 + a2 \text{tt}(1.5)) - \text{xx}(1.5)) + \\ \text{tt}(2.)^2 (a0 + \text{tt}(2.) (a1 + a2 \text{tt}(2.)) - \text{xx}(2.)))$$

However, we need to see what is going on here to make some sense,

Coefficient [ExpandAll [d0], a0]

10

Coefficient [ExpandAll [d0], a1]

$2 \text{tt}(0) + 2 \text{tt}(0.5) + 2 \text{tt}(1.) + 2 \text{tt}(1.5) + 2 \text{tt}(2.)$

Coefficient [ExpandAll [d0], a2]

$2 \text{tt}(0)^2 + 2 \text{tt}(0.5)^2 + 2 \text{tt}(1.)^2 + 2 \text{tt}(1.5)^2 + 2 \text{tt}(2.)^2$

Coefficient [ExpandAll [d1], a0]

$2 \text{tt}(0) + 2 \text{tt}(0.5) + 2 \text{tt}(1.) + 2 \text{tt}(1.5) + 2 \text{tt}(2.)$

Coefficient [ExpandAll [d1], a1]

$2 \text{tt}(0)^2 + 2 \text{tt}(0.5)^2 + 2 \text{tt}(1.)^2 + 2 \text{tt}(1.5)^2 + 2 \text{tt}(2.)^2$

Coefficient [ExpandAll [d1], a2]

$2 \text{tt}(0)^3 + 2 \text{tt}(0.5)^3 + 2 \text{tt}(1.)^3 + 2 \text{tt}(1.5)^3 + 2 \text{tt}(2.)^3$

Coefficient [ExpandAll [d2], a0]

$2 \text{tt}(0)^2 + 2 \text{tt}(0.5)^2 + 2 \text{tt}(1.)^2 + 2 \text{tt}(1.5)^2 + 2 \text{tt}(2.)^2$

```
Coefficient [ExpandAll [d2], a1]
```

$$2 \text{tt}(0)^3 + 2 \text{tt}(0.5)^3 + 2 \text{tt}(1.)^3 + 2 \text{tt}(1.5)^3 + 2 \text{tt}(2.)^3$$

```
Coefficient [ExpandAll [d2], a2]
```

$$2 \text{tt}(0)^4 + 2 \text{tt}(0.5)^4 + 2 \text{tt}(1.)^4 + 2 \text{tt}(1.5)^4 + 2 \text{tt}(2.)^4$$

It is fairly clear that there is a pattern. Note also that the xx 's all appear as a linear sum. Someone has figured out the pattern, you could do this if you knew there was one and had enough time.

Here is the problem that needs to be solved to get the coefficients

We start with the data, $x[t[i]], t[i]$.

We want to fit the coefficients, $\{a_0, a_1, a_2\}$ from the relation,

$$f[t] = a_0 + a_1 t(i) + a_2 t(i)^2$$

We construct a T matrix which is the table of the values of the independent variable. The functional relation for the independent variable can be nonlinear. The original data, $x[t[i], t[i]]$ are then related by the residual relations.

$$T \cdot a = x$$

If the number of data points (i.e., rows) was equal to the number of columns of T , then there would be a unique solution and no minimization would be necessary. However, we have more data and thus want to minimize the χ^2 function.

So starting with T .

```
T = Table[{1, tt[i], tt[i]^2}, {i, 0, 2, .5}]
```

$$\begin{pmatrix} 1 & \text{tt}(0) & \text{tt}(0)^2 \\ 1 & \text{tt}(0.5) & \text{tt}(0.5)^2 \\ 1 & \text{tt}(1.) & \text{tt}(1.)^2 \\ 1 & \text{tt}(1.5) & \text{tt}(1.5)^2 \\ 1 & \text{tt}(2.) & \text{tt}(2.)^2 \end{pmatrix}$$

It turns out that to reproduce the pattern generated from taking the derivative above we need the following operation.

$$\mathbf{T}^T \cdot \mathbf{T} \cdot \mathbf{a} = \mathbf{T}^T \mathbf{x}$$

which gives

$$\mathbf{a} = (\mathbf{T}^T \cdot \mathbf{T})^{-1} \mathbf{T}^T \mathbf{x}$$

to get \mathbf{a} .

Here are some checks,

```
test1 = (Transpose[T] . T) . {a0, a1, a2}
```

```
{5 a0 + a1 (tt(0) + tt(0.5) + tt(1.) + tt(1.5) + tt(2.)) + a2 (tt(0)^2 + tt(0.5)^2 + tt(1.)^2 + tt(1.5)^2 + tt(2.)^2),
 a0 (tt(0) + tt(0.5) + tt(1.) + tt(1.5) + tt(2.)) + a1 (tt(0)^2 + tt(0.5)^2 + tt(1.)^2 + tt(1.5)^2 + tt(2.)^2) +
 a2 (tt(0)^3 + tt(0.5)^3 + tt(1.)^3 + tt(1.5)^3 + tt(2.)^3),
 a0 (tt(0)^2 + tt(0.5)^2 + tt(1.)^2 + tt(1.5)^2 + tt(2.)^2) +
 a1 (tt(0)^3 + tt(0.5)^3 + tt(1.)^3 + tt(1.5)^3 + tt(2.)^3) +
 a2 (tt(0)^4 + tt(0.5)^4 + tt(1.)^4 + tt(1.5)^4 + tt(2.)^4)}
```

We can do some checks,

```
Coefficient[test1[[1]], a0]
```

```
5
```

```
Coefficient[test1[[1]], a1]
```

```
tt(0) + tt(0.5) + tt(1.) + tt(1.5) + tt(2.)
```

```
Coefficient[test1[[1]], a2]
```

```
tt(0)^2 + tt(0.5)^2 + tt(1.)^2 + tt(1.5)^2 + tt(2.)^2
```

```
Coefficient[test1[[2]], a0]
```

```
tt(0) + tt(0.5) + tt(1.) + tt(1.5) + tt(2.)
```

```
Coefficient[test1[[2]], a1]
```

```
tt(0)2 + tt(0.5)2 + tt(1.)2 + tt(1.5)2 + tt(2.)2
```

```
Coefficient[test1[[2]], a2]
```

```
tt(0)3 + tt(0.5)3 + tt(1.)3 + tt(1.5)3 + tt(2.)3
```

```
Coefficient[test1[[3]], a0]
```

```
tt(0)2 + tt(0.5)2 + tt(1.)2 + tt(1.5)2 + tt(2.)2
```

```
Coefficient[test1[[3]], a1]
```

```
tt(0)3 + tt(0.5)3 + tt(1.)3 + tt(1.5)3 + tt(2.)3
```

```
Coefficient[test1[[3]], a2]
```

```
tt(0)4 + tt(0.5)4 + tt(1.)4 + tt(1.5)4 + tt(2.)4
```

So we see that that sans a factor of 2, we have got it. Check the xx's

```
test2 = Transpose [aa] . {xx[0], xx[.5], xx[1], xx[1.5], xx[2]}
```

```
{xx(0) + xx(0.5) + xx(1) + xx(1.5) + xx(2),
 tt(0) xx(0) + tt(0.5) xx(0.5) + tt(1.) xx(1) + tt(1.5) xx(1.5) + tt(2.) xx(2),
 xx(0) tt(0)2 + tt(0.5)2 xx(0.5) + tt(1.)2 xx(1) + tt(1.5)2 xx(1.5) + tt(2.)2 xx(2)}
```

```
Coefficient [d0, xx[.5]]
```

```
-2
```

```
test2[[1]]
```

```
xx(0) + xx(0.5) + xx(1) + xx(1.5) + xx(2)
```

```
Coefficient [d1, xx[1.]]
```

```
-2 tt(1.)
```

```
test2[[2]]
```

```
tt(0) xx(0) + tt(0.5) xx(0.5) + tt(1.) xx(1) + tt(1.5) xx(1.5) + tt(2.) xx(2)
```

```
Coefficient [d2, xx[1.]]
```

```
-2 tt(1.)2
```

```
test2[[3]]
```

```
xx(0) tt(0)2 + tt(0.5)2 xx(0.5) + tt(1.)2 xx(1) + tt(1.5)2 xx(1.5) + tt(2.)2 xx(2)
```

Thus the factor of 2 is still there but the set of equations to be solved is

```
eqs = (Transpose[T] . T) . {a0, a1, a2} -
      Transpose[T] . {xx[0], xx[.5], xx[1.], xx[1.5], xx[2.]}
```

```
{5 a0 + a1 (tt(0) + tt(0.5) + tt(1.) + tt(1.5) + tt(2.)) +
  a2 (tt(0)^2 + tt(0.5)^2 + tt(1.)^2 + tt(1.5)^2 + tt(2.)^2) - xx(0) - xx(0.5) - xx(1.) - xx(1.5) - xx(2.),
 a0 (tt(0) + tt(0.5) + tt(1.) + tt(1.5) + tt(2.)) + a1 (tt(0)^2 + tt(0.5)^2 + tt(1.)^2 + tt(1.5)^2 + tt(2.)^2) +
  a2 (tt(0)^3 + tt(0.5)^3 + tt(1.)^3 + tt(1.5)^3 + tt(2.)^3) -
  tt(0) xx(0) - tt(0.5) xx(0.5) - tt(1.) xx(1.) - tt(1.5) xx(1.5) - tt(2.) xx(2.),
 -xx(0) tt(0)^2 + a0 (tt(0)^2 + tt(0.5)^2 + tt(1.)^2 + tt(1.5)^2 + tt(2.)^2) +
  a1 (tt(0)^3 + tt(0.5)^3 + tt(1.)^3 + tt(1.5)^3 + tt(2.)^3) +
  a2 (tt(0)^4 + tt(0.5)^4 + tt(1.)^4 + tt(1.5)^4 + tt(2.)^4) - tt(0.5)^2 xx(0.5) - tt(1.)^2 xx(1.) -
  tt(1.5)^2 xx(1.5) - tt(2.)^2 xx(2.)}
```

Here we check this, just to be sure!!

```
FullSimplify[d0 - 2 Flatten[eqs[[1]]]]
```

```
0
```

```
FullSimplify[d1 - 2 Flatten[eqs[[2]]]]
```

```
0
```

```
FullSimplify[d2 - 2 Flatten[eqs[[3]]]]
```

```
0
```

Now let's solve the first problem. In terms of the original data we have

```
t * t
```

```
{0, 0.25, 1., 2.25, 4.}
```

```
t
```

```
{0, 0.5, 1., 1.5, 2.}
```

```
x
```

```
{0.971397, 1.38058, 1.40375, 1.12953, 0.397554}
```

Here is the a matrix which is the matrix of the t values for each of the data points. This is one place to use `*` for a vector operation!!

Note that this matrix has a number columns equal to the number of coefficients to fit. However, the number of rows is equal to the number of data points. We better have more data points than coefficients!!

```
Tx = Transpose[{Table[1, {i, 0, 2, .5}], t, t*t}]
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 0.5 & 0.25 \\ 1 & 1. & 1. \\ 1 & 1.5 & 2.25 \\ 1 & 2. & 4. \end{pmatrix}$$

```
?? LinearSolve
```

LinearSolve[m, b] finds an x which solves the matrix equation $m.x==b$.

```
Attributes[LinearSolve] = {Protected}
```

```
Options[LinearSolve] = {Method -> Automatic, Modulus -> 0, ZeroTest -> (#1 == 0 &)}
```

The set of equations is

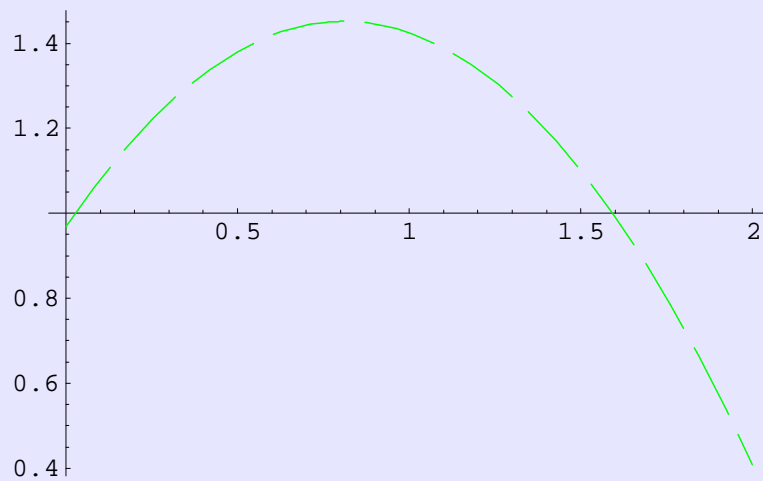
```
solv1 = LinearSolve[(Transpose[Tx] . Tx), Transpose[Tx] . x]
```

```
{0.967779, 1.19438, -0.737063}
```

```
fit = solv1 . {1, ty, ty^2}
```

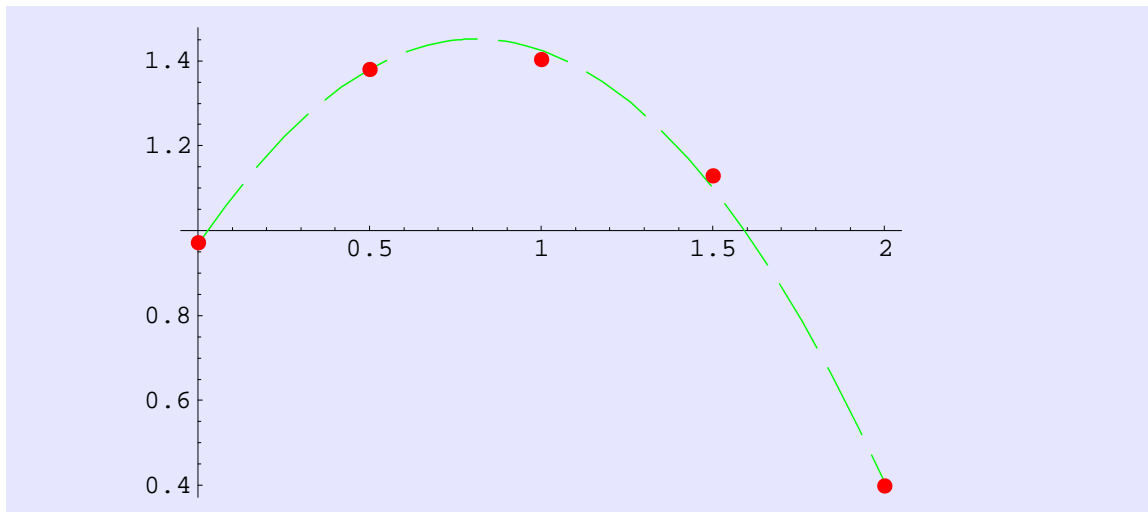
```
-0.737063 ty2 + 1.19438 ty + 0.967779
```

```
plot2 = Plot[fit, {ty, 0, 2},  
  PlotStyle -> {Dashing[ {.1, .03}], RGBColor[0, 1, 0]}]
```



- Graphics -

```
Show[plot2, plot1]
```



- Graphics -

We see that the result is not too bad. However, check this against the real expression

```
Series[Cos[z] + Sin[z], {z, 0, 2}]
```

$$1 + z - \frac{z^2}{2} + O(z^3)$$

```
fit
```

$$-0.737063 \text{ ty}^2 + 1.19438 \text{ ty} + 0.967779$$

- Try alot more points.

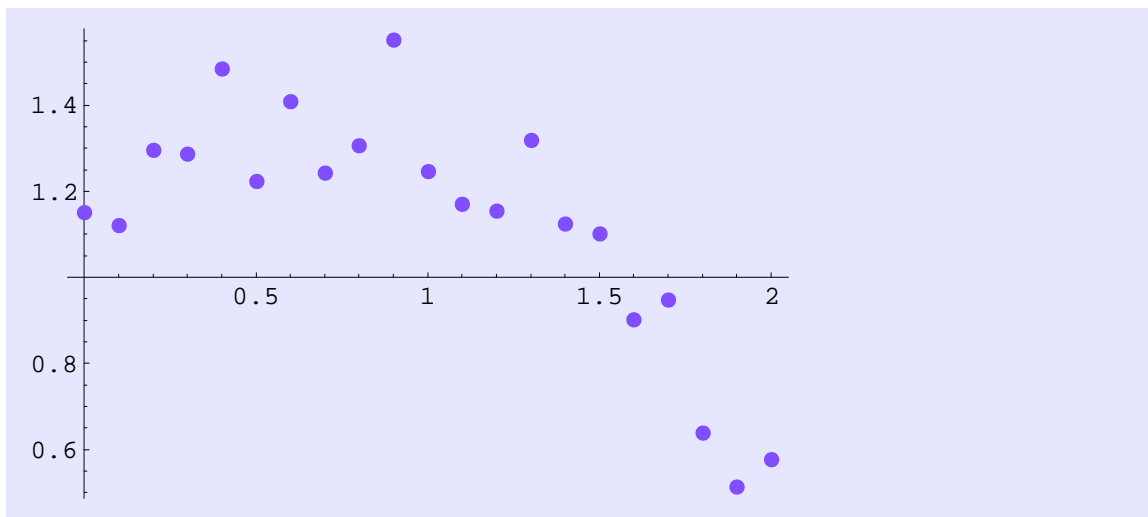
```
t = Table[i, {i, 0, 2, .1}]
```

```
{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1., 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.}
```

```
x = Table[Cos[i] + Sin[i] + Random[Real, {-.2, .2}], {i, 0, 2, .1}]
```

```
{1.15028, 1.12034, 1.29572, 1.28579, 1.48375, 1.22315, 1.4084, 1.24187, 1.30547,  
1.55142, 1.24571, 1.16907, 1.1544, 1.31732, 1.12324, 1.10008, 0.901399, 0.947385,  
0.637676, 0.512065, 0.57703}
```

```
plot3 = ListPlot[Transpose[{t, x}],  
PlotStyle -> {PointSize[.02], RGBColor[.5, .3, 1]}]
```



- Graphics -

```
aaa = Transpose[{Table[1, {i, 0, 2, .1}], t, t*t}]
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 0.1 & 0.01 \\ 1 & 0.2 & 0.04 \\ 1 & 0.3 & 0.09 \\ 1 & 0.4 & 0.16 \\ 1 & 0.5 & 0.25 \\ 1 & 0.6 & 0.36 \\ 1 & 0.7 & 0.49 \\ 1 & 0.8 & 0.64 \\ 1 & 0.9 & 0.81 \\ 1 & 1. & 1. \\ 1 & 1.1 & 1.21 \\ 1 & 1.2 & 1.44 \\ 1 & 1.3 & 1.69 \\ 1 & 1.4 & 1.96 \\ 1 & 1.5 & 2.25 \\ 1 & 1.6 & 2.56 \\ 1 & 1.7 & 2.89 \\ 1 & 1.8 & 3.24 \\ 1 & 1.9 & 3.61 \\ 1 & 2. & 4. \end{pmatrix}$$

The set of equations is

```
ans1 = LinearSolve[(Transpose[aaa] . aaa), Transpose[aaa] . x]
```

```
{1.13158, 0.686211, -0.502511}
```

```
fit1 = ans1 . {1, ty, ty^2}
```

```
-0.502511 ty^2 + 0.686211 ty + 1.13158
```

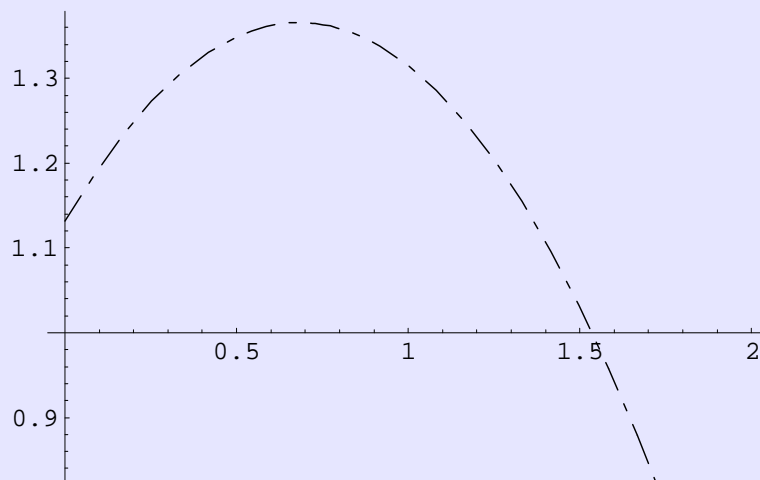
```
fit
```

```
-0.737063 ty^2 + 1.19438 ty + 0.967779
```

```
Series[Cos[z] + Sin[z], {z, 0, 2}]
```

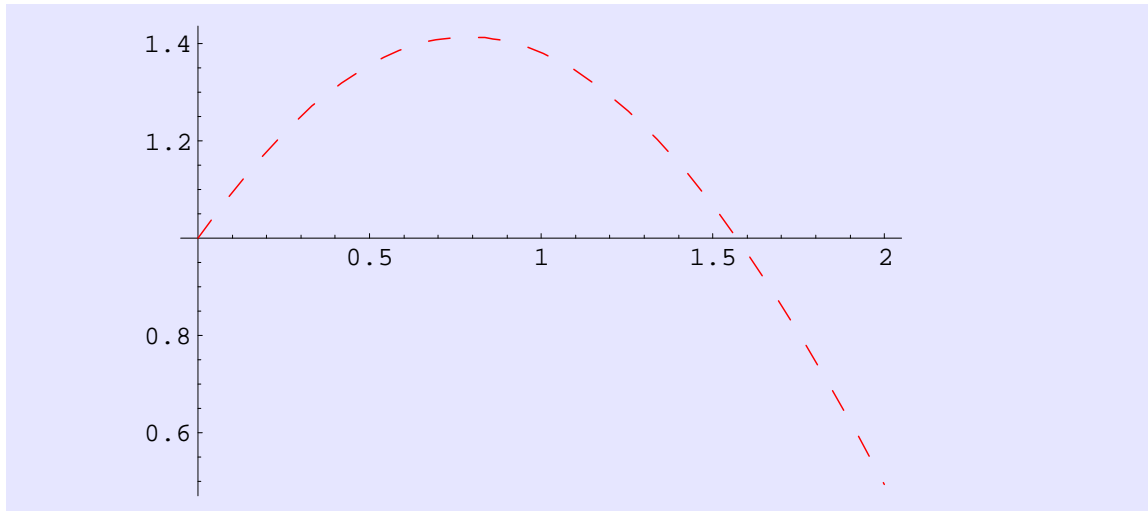
$$1 + z - \frac{z^2}{2} + O(z^3)$$

```
plot4 = Plot[{fit1}, {ty, 0, 2},  
  PlotStyle -> {Dashing[ {.04, .02, .01, .02} ]}]
```



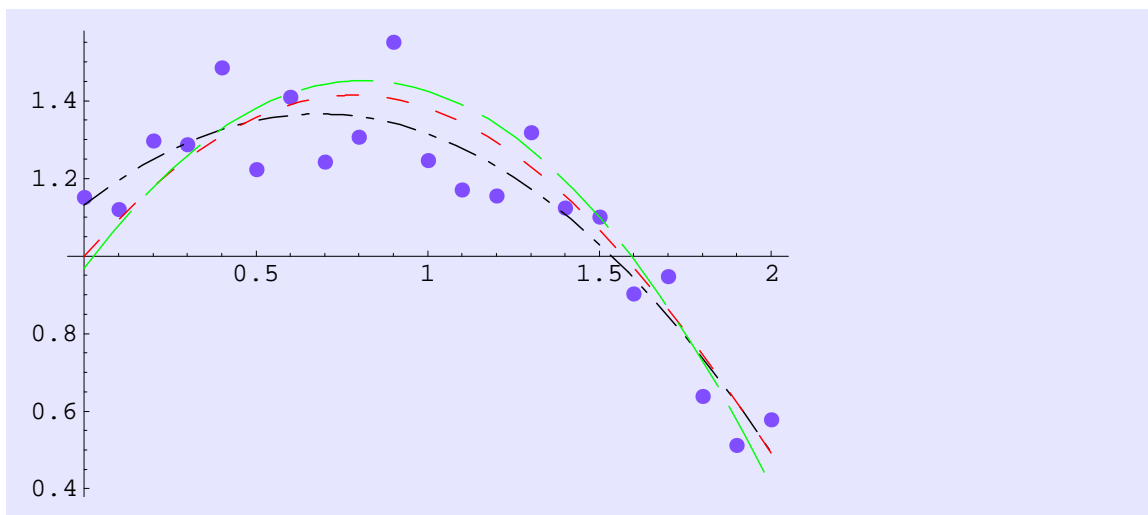
- Graphics -

```
plot5 = Plot[Cos[ty] + Sin[ty], {ty, 0, 2},  
PlotStyle -> {Dashing[ {.03, .04}], RGBColor[1, 0, 0]}]
```



- Graphics -

```
Show[plot3, plot4, plot5, plot2]
```



- Graphics -

We see that the formalism that allows least squares fitting is a linear algebra problem, but the function does not have to be linear! Note that we have not answered how accurate the coefficients are! Check out *Numerical Recipes* or *Applied Linear algebra*.

■ Now try a lot less noise

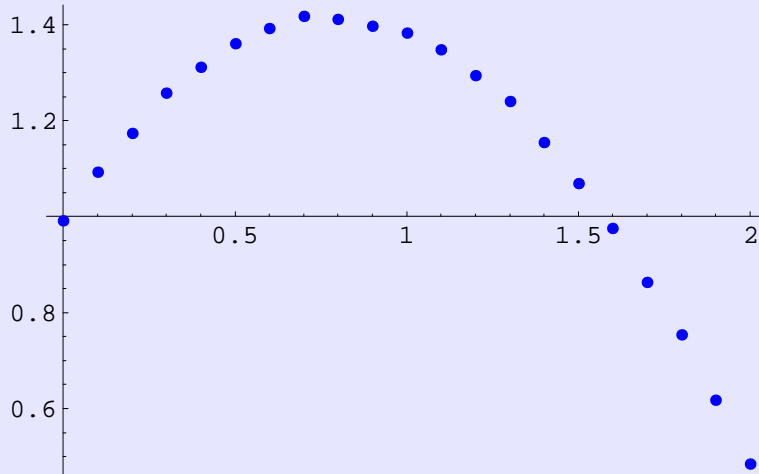
```
t = Table[i, {i, 0, 2, .1}]
```

```
{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1., 1.1, 1.2, 1.3, 1.4, 1.5, 1.6, 1.7, 1.8, 1.9, 2.}
```

```
x = Table[Cos[i] + Sin[i] + Random[Real, {-.01, .01}], {i, 0, 2, .1}]
```

```
{0.99203, 1.09326, 1.17275, 1.25754, 1.31123, 1.35958, 1.39224, 1.41708, 1.41151,  
1.39659, 1.3824, 1.34825, 1.29452, 1.23951, 1.15483, 1.06868, 0.976184, 0.862884,  
0.754465, 0.616905, 0.484733}
```

```
plot3 = ListPlot[Transpose[{t, x}],  
PlotStyle -> {PointSize[.015], RGBColor[0, 0, 1]}]
```



- Graphics -

```
aaa = Transpose[{Table[1, {i, 0, 2, .1}], t, t*t}]
```

$$\begin{pmatrix} 1 & 0 & 0 \\ 1 & 0.1 & 0.01 \\ 1 & 0.2 & 0.04 \\ 1 & 0.3 & 0.09 \\ 1 & 0.4 & 0.16 \\ 1 & 0.5 & 0.25 \\ 1 & 0.6 & 0.36 \\ 1 & 0.7 & 0.49 \\ 1 & 0.8 & 0.64 \\ 1 & 0.9 & 0.81 \\ 1 & 1. & 1. \\ 1 & 1.1 & 1.21 \\ 1 & 1.2 & 1.44 \\ 1 & 1.3 & 1.69 \\ 1 & 1.4 & 1.96 \\ 1 & 1.5 & 2.25 \\ 1 & 1.6 & 2.56 \\ 1 & 1.7 & 2.89 \\ 1 & 1.8 & 3.24 \\ 1 & 1.9 & 3.61 \\ 1 & 2. & 4. \end{pmatrix}$$

The set of equations is

```
ans2 = LinearSolve[(Transpose[aaa] . aaa), Transpose[aaa] . x]
```

```
{1.00224, 1.02219, -0.645499}
```

```
fit2 = ans2 . {1, ty, ty^2}
```

```
-0.645499 ty^2 + 1.02219 ty + 1.00224
```

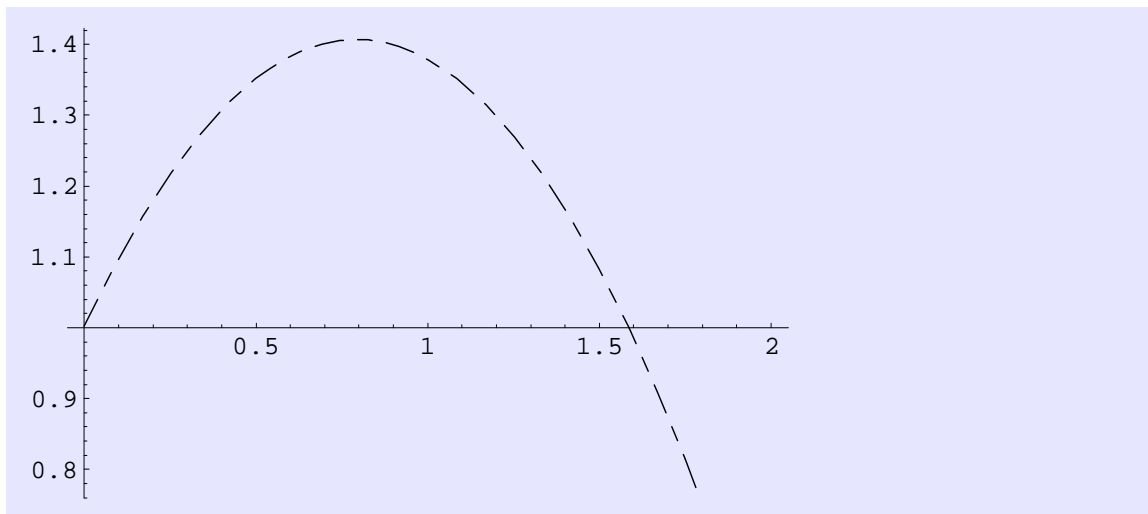
```
fit
```

```
-0.737063 ty^2 + 1.19438 ty + 0.967779
```

```
Series[Cos[z] + Sin[z], {z, 0, 2}]
```

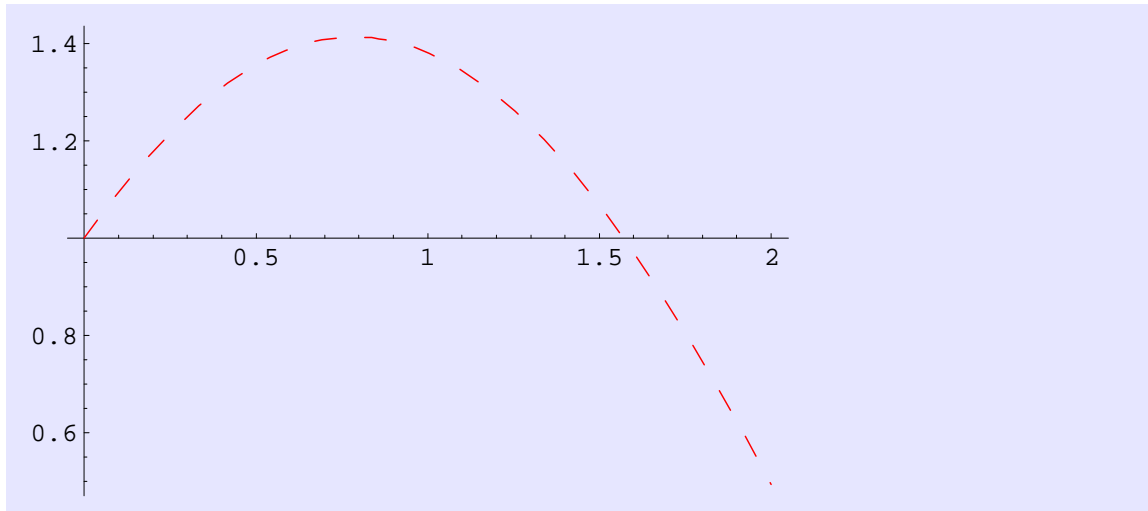
$$1 + z - \frac{z^2}{2} + O(z^3)$$

```
plot4 = Plot[{fit2}, {ty, 0, 2},  
PlotStyle -> {Dashing[ {.04, .02, .02, .02} ]}]
```



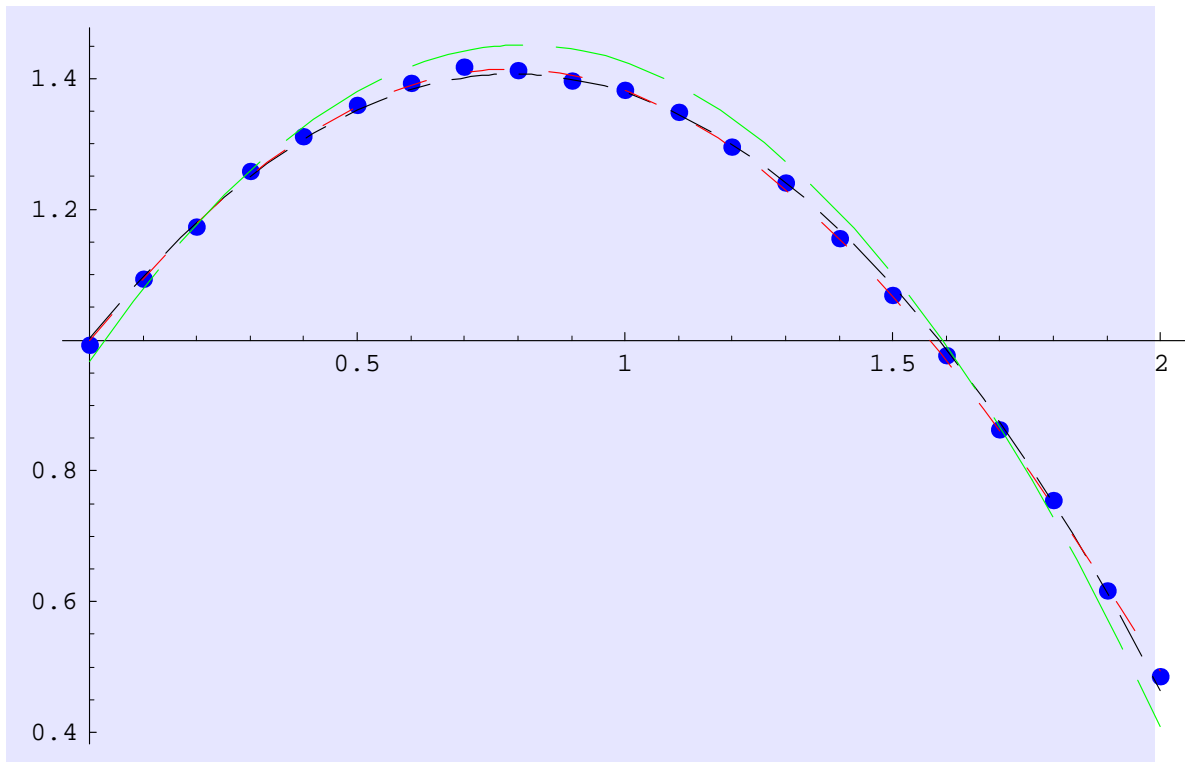
- Graphics -

```
plot5 = Plot[Cos[ty] + Sin[ty], {ty, 0, 2},  
  PlotStyle -> {Dashing[ {.03, .04}], RGBColor[1, 0, 0]}]
```



- Graphics -

```
Show[plot3, plot4, plot5, plot2]
```



- Graphics -

We see that the formalism that allows least squares fitting is a linear algebra problem, but the function does not have to be linear!! Note that we have not answered how accurate the coefficients are! Check out Numerical Recipes or Applied Linear algebra.

■ How to do error analysis for our favorite fluid mechanics experiment, flow in pipes

$f = 16/Re$, say is the expected answer

```
rex = {10, 20, 35, 45, 50, 70, 100, 200, 500}
```

```
{10, 20, 35, 45, 50, 70, 100, 200, 500}
```

```
ff = 16 / rex
```

$$\left\{ \frac{8}{5}, \frac{4}{5}, \frac{16}{35}, \frac{16}{45}, \frac{8}{25}, \frac{8}{35}, \frac{4}{25}, \frac{2}{25}, \frac{4}{125} \right\}$$

```
ffn = N[ff]
```

```
{1.6, 0.8, 0.457143, 0.355556, 0.32, 0.228571, 0.16, 0.08, 0.032}
```

```
fflogdata =
```

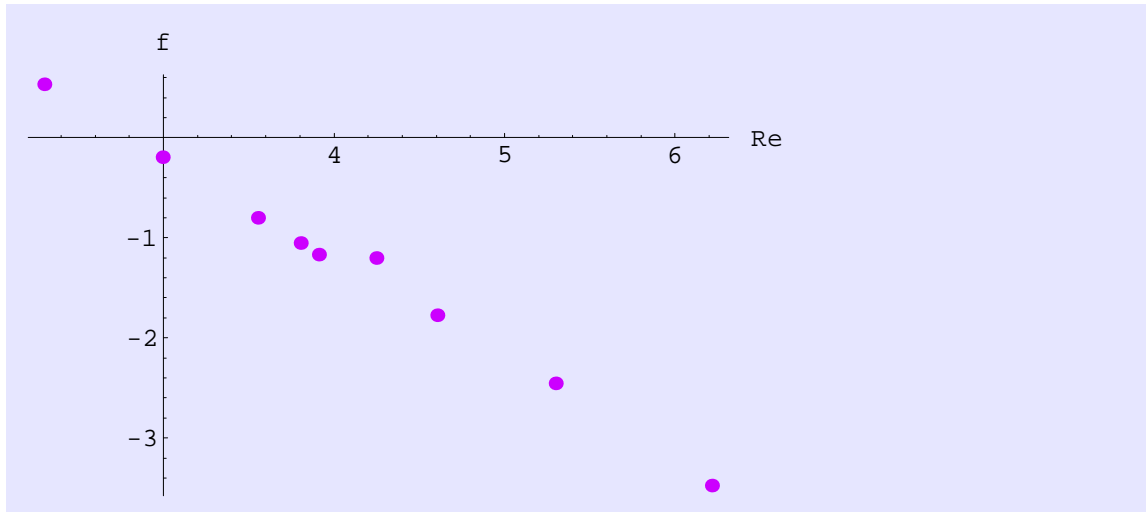
```
Log[{1.7, 0.82, 0.45, 0.35, 0.31, 0.3, 0.17, 0.086, 0.031}]
```

```
{0.530628, -0.198451, -0.798508, -1.04982, -1.17118, -1.20397, -1.77196,  
-2.45341, -3.47377}
```

```
pipedata = Transpose[{Log[rex], fflogdata}]
```

$$\begin{pmatrix} \log(10) & 0.530628 \\ \log(20) & -0.198451 \\ \log(35) & -0.798508 \\ \log(45) & -1.04982 \\ \log(50) & -1.17118 \\ \log(70) & -1.20397 \\ \log(100) & -1.77196 \\ \log(200) & -2.45341 \\ \log(500) & -3.47377 \end{pmatrix}$$

```
replot = ListPlot[ pipedata,
  PlotStyle -> {PointSize[.02], RGBColor[.8, 0, 1]},
  AxesLabel -> {"Re", "f"}]
```



- Graphics -

Here we make the matrix of independent variable values

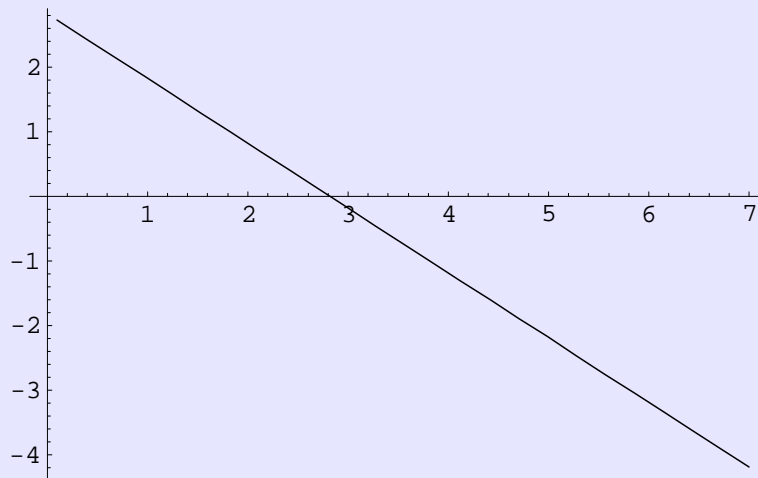
```
Tz = Transpose[{{1, 1, 1, 1, 1, 1, 1, 1, 1}, Log[rex]}]
```

$$\begin{pmatrix} 1 & \log(10) \\ 1 & \log(20) \\ 1 & \log(35) \\ 1 & \log(45) \\ 1 & \log(50) \\ 1 & \log(70) \\ 1 & \log(100) \\ 1 & \log(200) \\ 1 & \log(500) \end{pmatrix}$$

```
ans = Inverse[Transpose[Tz] . Tz] . Transpose[Tz] . fflogdata
```

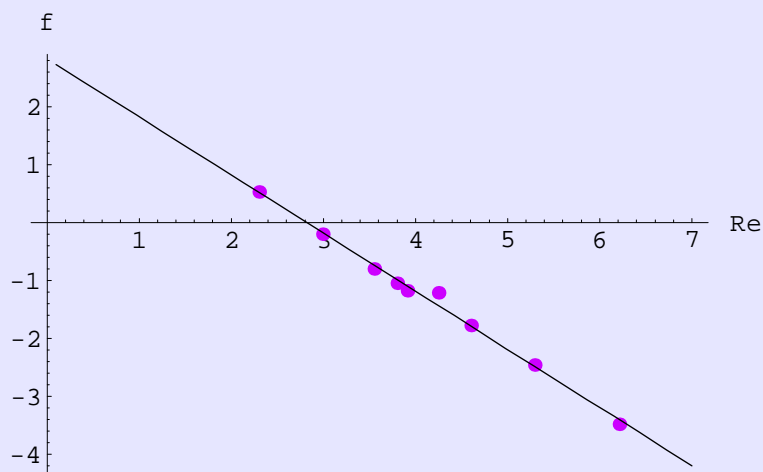
```
{2.82699, -1.00256}
```

```
replot2 = Plot[ans . {1, x}, {x, .1, 7}]
```



- Graphics -

```
Show[replot, replot2]
```



- Graphics -