MSE 310/ECE 340

Instructor: Bill Knowlton

Examples of Plotting Functions

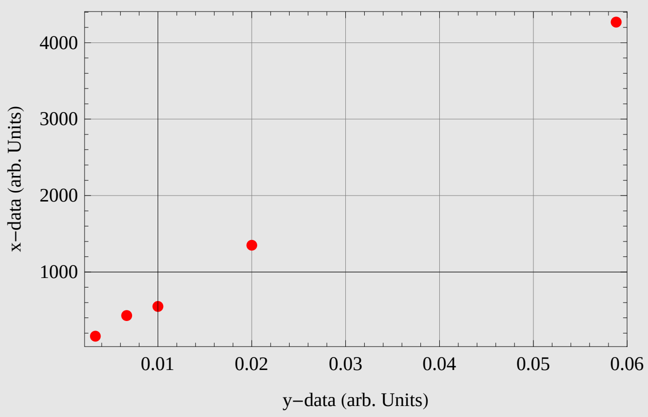
- **Example 1: Using the command ListPlot[ ] to plot data in a scatter plot**

In[1]:=

```
(*Initialize data*)
Clear[someData, somePlot]

(*Place data in list*)
someData =
 {{1/17, 4270}, {1/50, 1350}, {1/100, 550}, {1/150, 430}, {1/300, 160}}

(*Create scatter plot of data*)
somePlot = ListPlot[someData, Frame → True, GridLines → Automatic,
  PlotStyle → {RGBColor[1, 0, 0], PointSize[0.02`]},
  FrameLabel → {"y-data (arb. Units)", "x-data (arb. Units)"}]
```

Out[2]= $\left\{\left\{\frac{1}{17}, 4270\right\}, \left\{\frac{1}{50}, 1350\right\}, \left\{\frac{1}{100}, 550\right\}, \left\{\frac{1}{150}, 430\right\}, \left\{\frac{1}{300}, 160\right\}\right\}$

Out[3]=



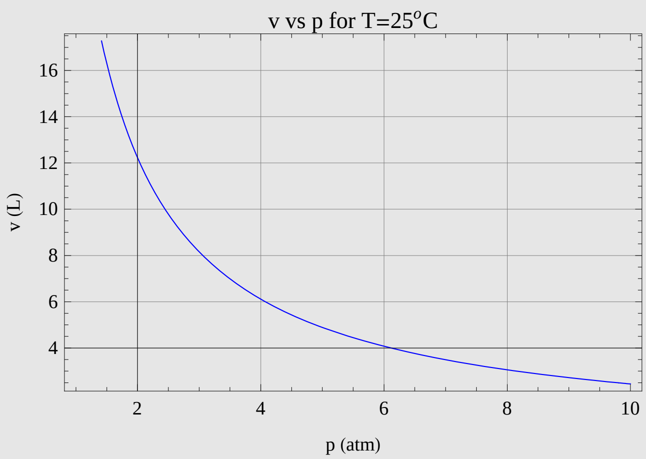
Example 2: Using the command Plot[ ] to plot a function

- In the Input Cell below, I use the ideal gas law showing a direct relation with temperture, T, and an inverse relation with pressure and is given by:
  v[p_, n_, T_] := $\frac{n R T}{p}$

  Note that there are other variables, but they are not included in the bracket [ ]. The variables included in the [ ] are call Arguments allows us to vary or give a value directly for each variable in the [ ]. For more information, see Help –> Master Index –> [ ].

  Lastly, to add a comment in the Input Cell that *Mathematica* will ignore, use (* comment *). I use this to document what I'm up to in the program.

```
Clear[v, R, T, p]
v[p_, n_, T_] := n R T / p
(*Constants*)
R = 0.08206;(*atm L /(mol K)*)

(*Plots*)
Plot[v[p, 1, 298], {p, 1, 10}, Frame → True, GridLines → Automatic,
 PlotStyle → {RGBColor[0, 0, 1]}, FrameLabel → {"p (atm)", "v (L)"},
 PlotLabel -> "v vs p for T=25oC"]
Print["v: ", v[1, 1, 298], "L at T=25oC"]
```




```
v: 24.4539L at T=25oC
```

Example 3: Plotting multiple plots of a funtion by creating a table of output using the command Table[ ]

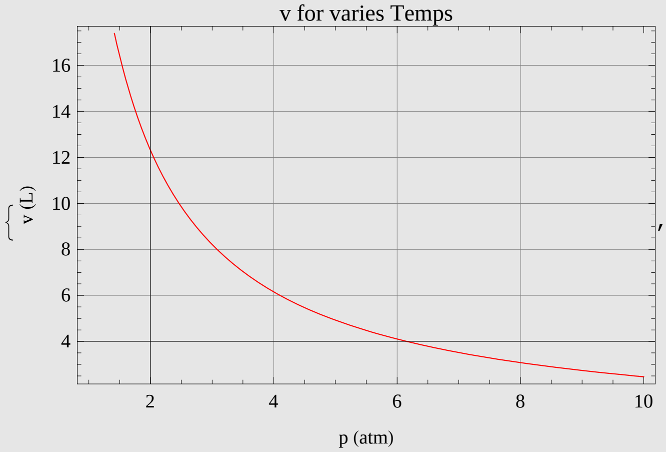
- **Plot your function over 2 variables. If I want to graphically examine a function in 2D but with 2 variables, one way I do so is shown below.**

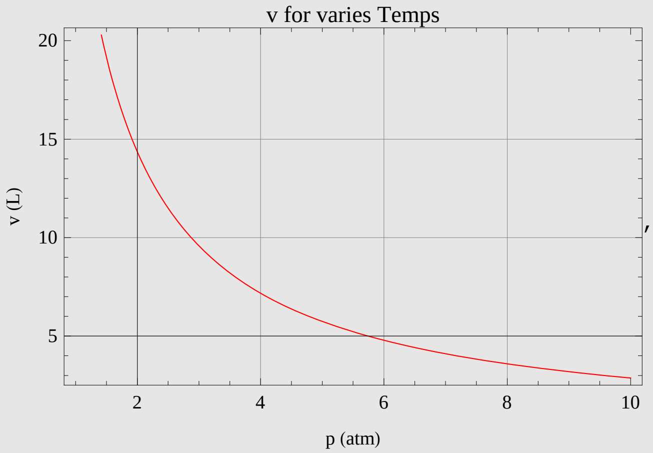
  The easiest way to show all of the graphs together would be to use the Table[ ] command instead of using a "For loop" or "Do loop" approach. The Table is called gasplots, and contains the same Print and Plot commands as in the For loop. The second argument {T,300,750,50}, iterates T from 300K to 750K in steps of 50, which returns the same result as the For loop.

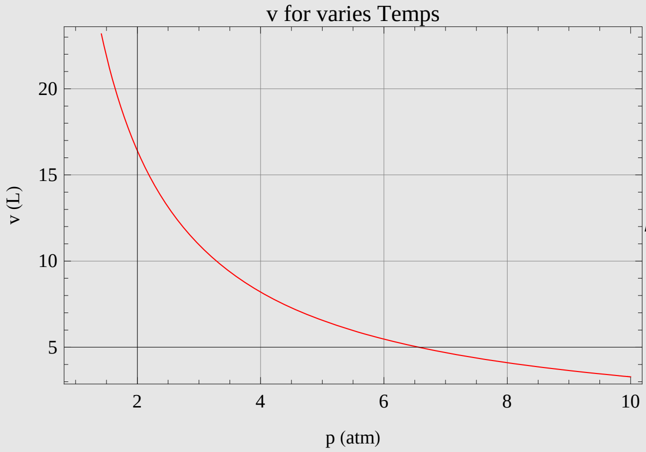
  Note that this is essentially a program because it includes a For loop type functionality and a Print command and a Plot command and a Table command. Note that there are multiple [ ]s.

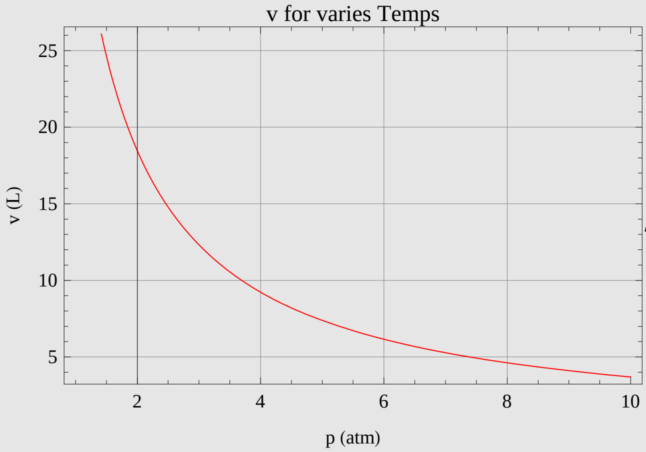
```
gasplots = Table[
  Print["T= ", T, " K"];
  Plot[v[p, 1, T], {p, 1, 10}, Frame → True, GridLines → Automatic,
   PlotStyle → {RGBColor[1, 0, 0]}, FrameLabel → {"p (atm)", "v (L)"},
   PlotLabel -> "v for varies Temps"], {T, 300, 750, 50}]
```

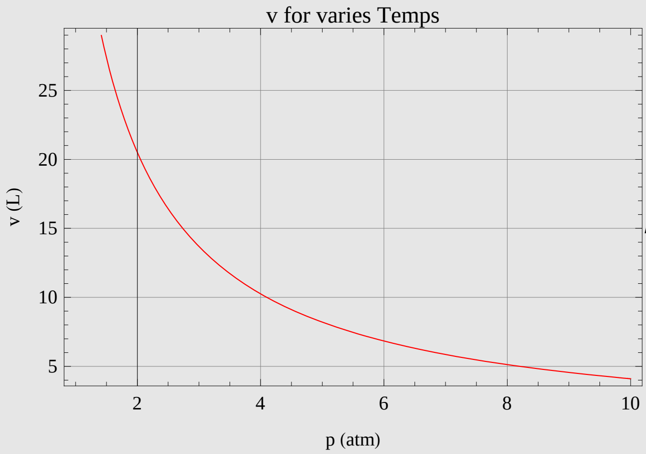
```
T= 300 K
T= 350 K
T= 400 K
T= 450 K
T= 500 K
T= 550 K
T= 600 K
T= 650 K
T= 700 K
T= 750 K
```


v for varies Temps
v (L)
16
14
12
10
8
6
4
2
4
6
8
10
p (atm)

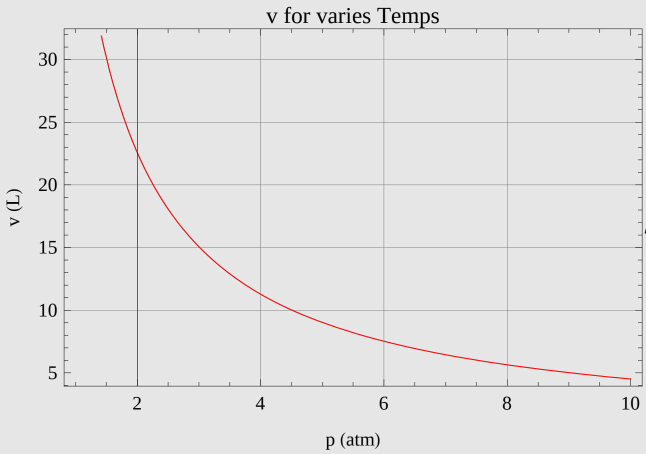

v for varies Temps
v (L)
20
15
10
5
2
4
6
8
10
p (atm)


v for varies Temps
v (L)
20
15
10
5
2
4
6
8
10
p (atm)
,

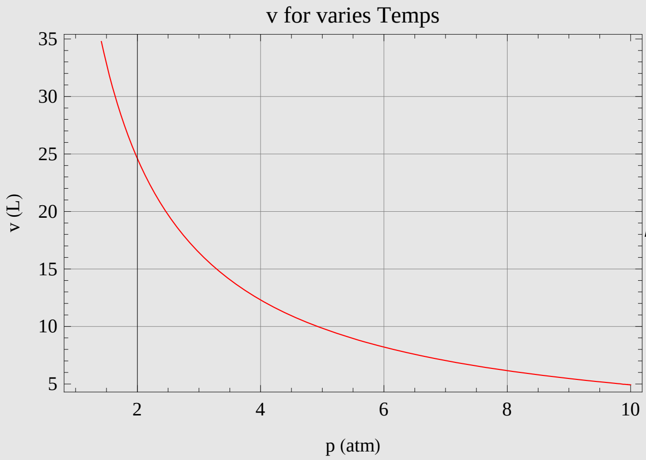

v for varies Temps
v (L)
25
20
15
10
5
2
4
6
8
10
p (atm)
,


v for varies Temps
v (L)
25
20
15
10
5
2
4
6
8
10
p (atm)

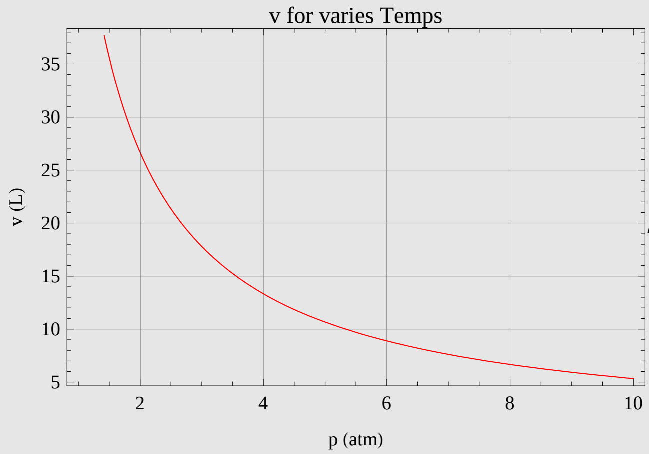
,


v for varies Temps
v (L)
30
25
20
15
10
5
2
4
6
8
10
p (atm)

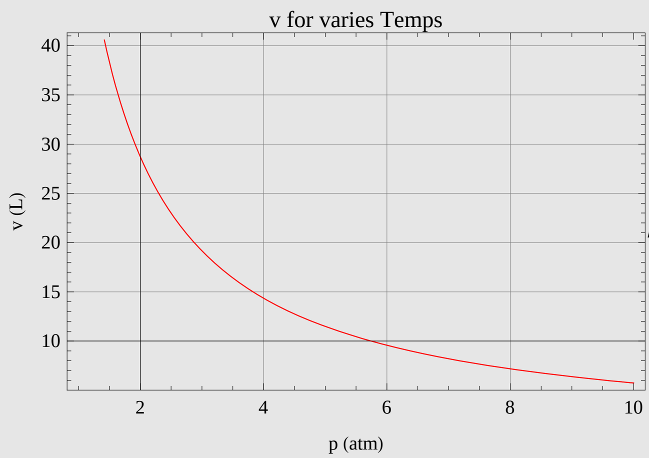
,


v for varies Temps
v (L)
p (atm)
35
30
25
20
15
10
5
2
4
6
8
10

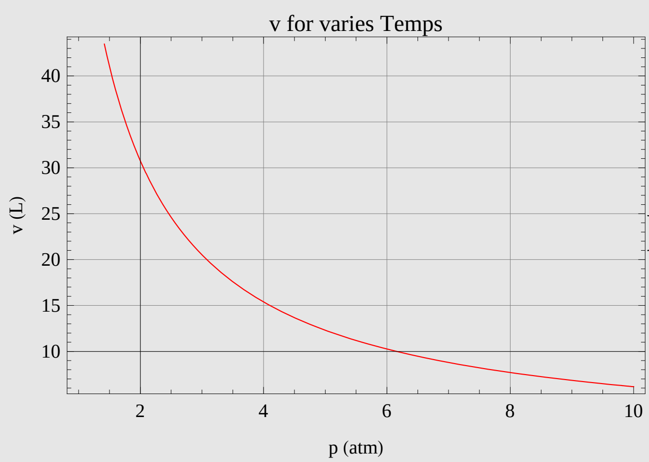
,



,



,



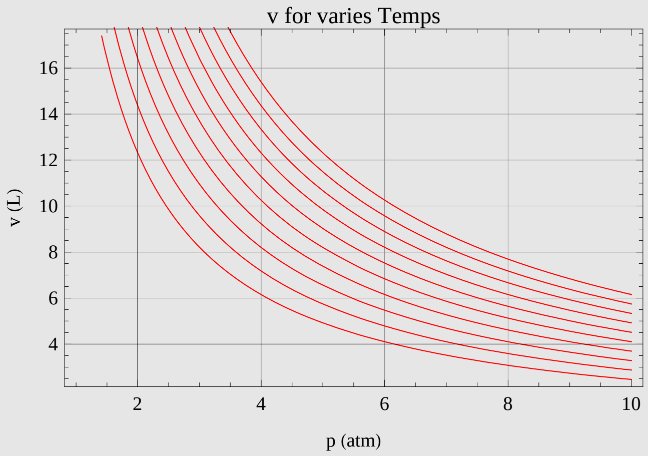
}

■ Example 4: Combining all the plots to show one plot using the command Show[ ]

- Plot the above plots in one graph.

One way to plot all the data in one graph is to use the Show[ ] command.

```
Show[gasplots]
```



Examples of Plotting and Fitting Data Using the Commands:
1. LinearFitModel[ ]
2. NonlinearFitModel[ ]

- **LinearModelFit** returns a symbolic **FittedModel** object to represent the linear model it constructs. The properties and diagnostics of the model can be obtained from *model*["*property*"].

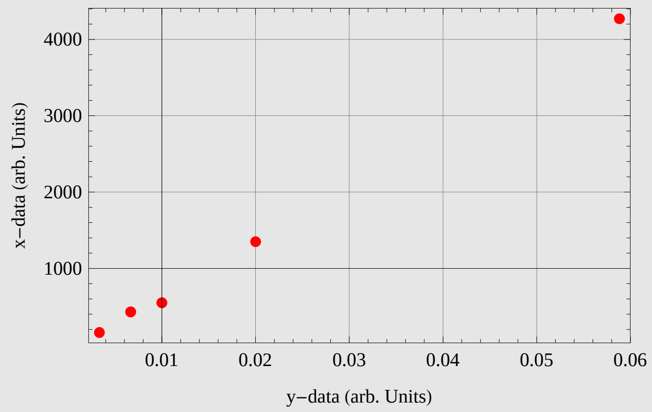
  NonlinearModelFit returns a symbolic **FittedModel** object to represent the linear model it constructs. The properties and diagnostics of the model can be obtained from *model*["*property*"].

  Using this approach to fit data allows one to use the execute additional commands

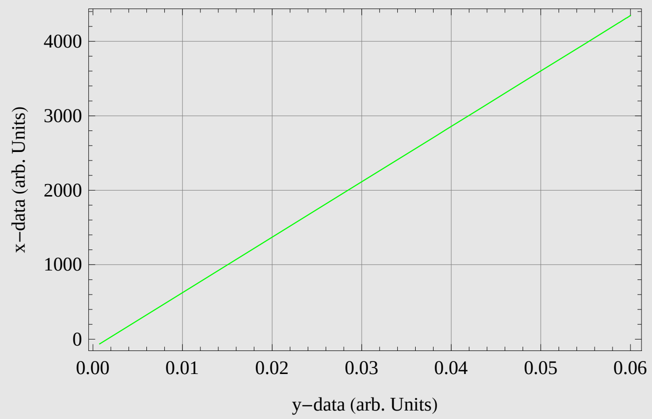
Example Using ListPlot[ ] to plot data and LinearModelFit[ ]

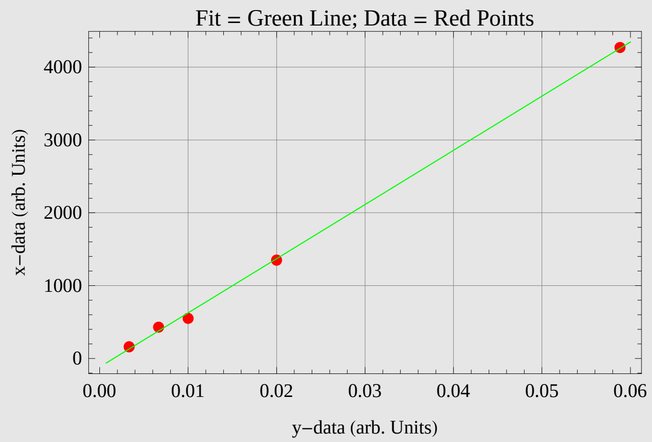
```
(*Initialize data*)
Clear[DifData, DifDataplot, DifDatafit, plotDifFit, diffit]

(*Place data in list*)
DifData =
  {{1 / 17, 4270}, {1/50, 1350}, {1/100, 550}, {1/150, 430}, {1/300, 160}}

(*Create scatter plot of data*)
DifDataplot = ListPlot[DifData, Frame → True, GridLines → Automatic,
  PlotStyle → {RGBColor[1, 0, 0], PointSize[0.02`]},
  FrameLabel → {"y-data (arb. Units)", "x-data (arb. Units)"}]

(*Perform fit and define the fitting function*)
Print["y = ", DifDatafit = Fit[DifData, {1, x}, x]]

plotDifFit = Plot[DifDatafit, {x, 0.00075, .06}, Frame → True,
  GridLines → Automatic, PlotStyle → {RGBColor[0, 1, 0]},
  FrameLabel → {"y-data (arb. Units)", "x-data (arb. Units)"}]
(*Show[ ] command to show the plots pplot and plotpfit
 on the same graph*) Show[DifDataplot, plotDifFit,
 PlotLabel → "Fit = Green Line; Data = Red Points"]

(* Use LinearModelFit[ ] command that is new to Mathematica 7*)
diffit = LinearModelFit[DifData, x, x]

(* Provides the adjusted R^2 value *)
Print["R^2 = ", diffit["RSquared"]] (* Provides the R^2 value *)
Print["Adjusted R^2 = ", diffit["AdjustedRSquared"]]

(* Provides the percent error *)
Print["Percent Error = ", (-24 246.5 - (-24 137))/(-24 246.5) 100, " %"]
```

```
{{1/17, 4270}, {1/50, 1350}, {1/100, 550}, {1/150, 430}, {1/300, 160}}
```



y = −118.625 + 74 406.6 x





FittedModel[−118.625 + 74 406.6 x]

R^2 = 0.999133

Adjusted R^2 = 0.998844

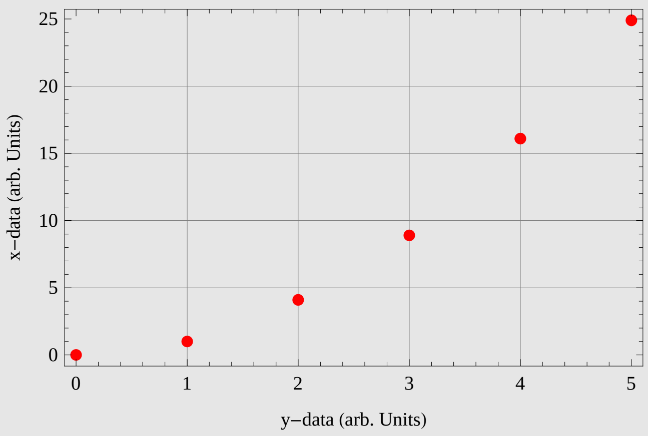
Percent Error = 0.451612 %

Example Using ListPlot[ ] to plot data and NonlinearModelFit[ ]

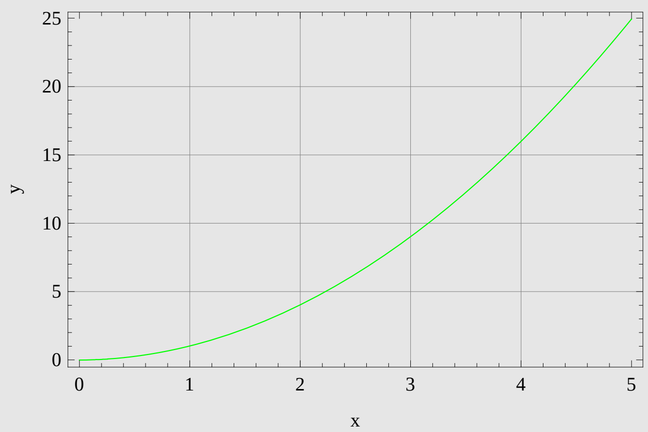
- For Non-linear curve fitting and output of R^2 , one can use the NonlinearModelFit[ ] function.

```
(*initialize data*)
Clear[SomeData, SomeDataplot, SomeDatafit, plotFit]

(*Place data in list*)
SomeData = {{0, 0}, {1, 1}, {2, 4.1}, {3, 8.9}, {4, 16.1}, {5, 24.9}}

(*Create scatter plot of data*)
SomeDataplot = ListPlot[SomeData, Frame → True, GridLines → Automatic,
  PlotStyle → {RGBColor[1, 0, 0], PointSize[0.02`]},
  FrameLabel → {"y-data (arb. Units)", "x-data (arb. Units)"}]

(*Perform fit and define the fitting function*)
Print["y = ", SomeDatafit =
  NonlinearModelFit[SomeData, a x^2 + b x + c, {a, b, c}, x]]

(*Plotting the NonlinearModelFit data. Note that SomeDatafit has
 to show that it is a function of x; I.e., SomeDatafit[x]*)
plotFit = Plot[SomeDatafit[x], {x, 0, 5},
  Frame → True, GridLines → Automatic,
  PlotStyle → {RGBColor[0, 1, 0]}, FrameLabel → {"x", "y"}]

(*showing the plots "pplot" and "plotpfit" on the same graph*)
Show[SomeDataplot, plotFit,
 PlotLabel → "Fit = Green Line; Data = Red Points"]

(* Provides the R^2 value *)
Print["R^2 = ", SomeDatafit["RSquared"]]
(* Provides the adjusted R^2 value *)
Print["Adjusted R^2 = ", SomeDatafit["AdjustedRSquared"]]
```

```
{{0, 0}, {1, 1}, {2, 4.1}, {3, 8.9}, {4, 16.1}, {5, 24.9}}
```



y = FittedModel[$-0.00714286 + 0.0421429\,x + 0.989286\,x^2$]



R^2 = 0.999966

Adjusted R^2 = 0.999932