Simple Regression

This discussion requires learning some new Lisp techniques. I'll explain as the need arises. Consider the example in class, in which we're trying to use average daily temperature (degrees Farenheit) to estimate pounds of steam used monthly. I have gained some more information about this data set, and can tell you it comes from a study of a steam plant at "a large industrial concern". The study also looked at things like wind velocity, pounds of real fatty acid in storage per month, number of operating days per month, etc. We'll use temperature as the independent variable and pounds of steam as the response variable.
The first task is to enter the data, which, since they consist of two short lists, I entered directly into the xlisp-stat environment:

```
> (def temp (list 35.3 29.7 30.8 58.8 61.4 71.3 74.4 76.7 70.7 57.5
 46.4 28.9 28.1 39.1 46.8 48.5 59.3 70 70 74.5 72.1 58.1 44.6 33.4 28.6))
TEMP
> (def pounds (list 10.98 11.13 12.51 8.4 9.27 8.73 6.36 8.5 7.82 9.14
8.24 12.19 11.88  9.57 10.94 9.58 10.09  8.11  6.83  8.88  7.68  8.47
8.86  10.36 11.08))
POUNDS
>
> (length temp)
25
> (length pounds)
25
```

Checking the length of the lists is one technique for detecting data-entry errors. Before beginning, you should plot the data. "(plot-points temp pounds)" will do, but we want to save the work, so instead type "(def ourplot (plot-points temp pounds)". Next, we run the regression and give the output the name "steam". The ":print nil" command is not necessary, but it supresses the output. You can leave this off if you wish.

```
>> (def steam (regression-model temp pounds :print nil))
STEAM
> (send steam :coef-estimates)
(13.622989268172683 -0.07982869331126773)
> (send steam :coef-standard-errors)
(0.5814634941477705 0.010523580966436095)
```

The "send" command is a strange one. Think of *steam* as an object that will tell you about itself only if you send it messages instructing it to tell you something.

So the line (send stream :coef-estimates) can be translated to read "tell me what your estimates of the coefficients of this regression model are." Perhaps you have never spoken to a data structure before, and this will take some getting used to. Naturally, there are only certain things the two of you can talk about. To see what they are, send a help message to *steam*:

```
> (send steam :help)
loading in help file information - this will take a minute ...done
REGRESSION-MODEL-PROTO
Normal Linear Regression Model
Help is available on the following:

ADD-METHOD ADD-SLOT BASIS CASE-LABELS COEF-ESTIMATES COEF-STANDARD-ERRORS COMPUTE COOKS-DIS
NIL
```

Some of these messages require additional inputs. To learn more about any of them, type "(send steam :help :display)" returns this:

```
> (send steam :help :display)
loading in help file information - this will take a minute ...done
DISPLAY
Message args: ()
Prints the least squares regression summary. Variables not used in the fit
are marked as aliased.
NIL

> (send steam :display)

Least Squares Estimates:

Constant                  13.6230      (0.581463)
Variable 0               -7.982869E-2  (1.052358E-2)

R Squared:               0.714438
Sigma hat:               0.890125
Number of cases:               25
Degrees of freedom:            23
```

You'll notice that the display is only partially helpful. It doesn't provide us with any t-tests. But you can get everything you need to make your own. And if necessary, one can easily write a program to provide them. Most computer outputs do this automatically, but xlispstat apparently does not.
Our next task is to add the regression line to our plot. Because we have saved the plot as an object called "ourplot", we can send it messages. (Different messages than we could send to the regression-model object. Use the help message to see what we can do.

```
> (send ourplot :abline 13.6230 -7.98E-2)
NIL
> (send steam :plot-residuals temp)
#<Object: 1236312, prototype = SCATTERPLOT-PROTO>
>
```

"abline" requires an intercept and slope as input. It then superimposes the line with that intercept and slope onto the graph. "plot-residuals" automatically plots the residuals versus whatever variable you type afterwards.