

Mathematica for Module 8.3 on "Empirical Models"
File: *EmpiricalModels.nb*

Introduction to Computational Science: Modeling and Simulation for the Sciences
Angela B. Shiflet and George W. Shiflet
Wofford College
© 2006 by Princeton University Press

Linear Empirical Model

Norris.dat

National Institute of Standards and Technology (NIST) "study involving calibration of ozone measurement of ozone concentration" and y is "the customer's measurement"
(<http://www.itl.nist.gov/div898/strd/ils/data/Norris.shtml>)

■ Some of the data

```
pts = {{0.2, 0.1}, {0.4, 0.3}, {0.3, 0.3}}
```

- Alternatively, data in two lists. Form a list of ordered pairs

```
xLst = {0.2, 0.4, 0.3, 0.3};  
yLst = {0.1, 0.3, 0.3, 0.6};
```

- Plot the points (Figure 8.3.1)

```
lp = ListPlot[pts,  
  PlotStyle -> {PointSize[.03]}, AxesL  
  PlotRange -> {{0, 0.62}, {0, 0.62}}]
```

- Fit function

```
fiteq = Fit[pts, {1, x}, x]
```

- Plot function (Figure 8.3.2)

```
pltfit = Plot[fiteq, {x, 0, .62}];
```

```
Show[lp, pltfit];
```

■ Plot point at (0.34, 0.365) (Figure 8.3.3)

```
plotPt = ListPlot[{{0.34, 0.365}},  
  PlotStyle -> {PointSize[.04]}, AxesL  
  PlotRange -> {{0, 0.62}, {0, 0.62}}]
```

```
Show[lp, pltfit, plotPt];
```

■ Complete set of data

```
x = {0.2, 337.4, 118.2, 884.6, 10.1, 226.5,  
  448.6, 777.0, 558.2, 0.4, 0.6, 775.5, 6  
  447.5, 11.6, 556.0, 228.1, 995.8, 887.  
  0.3, 556.8, 339.1, 887.2, 999.0, 779.0  
  229.2, 669.1, 448.9, 0.5};  
y = {0.1, 338.8, 118.1, 888.0, 9.2, 228.1,  
  449.1, 778.9, 559.2, 0.3, 0.1, 778.1, 6  
  448.9, 10.8, 557.7, 228.3, 998.0, 888.  
  0.6, 557.6, 339.3, 888.0, 998.5, 778.9  
  228.9, 668.4, 449.2, 0.2};
```

Non-Linear One-Term Model

DanWood.dat

The variable x is "the absolute emperature of the filament in 1000 degrees Kelvin," while y is "the carbon filament lamp per cm**2 per second."

(http://www.itl.nist.gov/div898/strd/nls/data/daniel_wood.shtml)

Reference: Data and model described in Daniel, C. and F. S. Wood (1980). "Fitting Equat
York, NY: John Wiley and Sons, pp. 428-431. Originally published in E.S.Keeping, "Intro
Nostrand Company, Princeton, NJ, 1962, p. 354.

■ Plot data

```
xLst = {1.309, 1.471, 1.490, 1.565, 1.611,  
yLst = {2.138, 3.421, 3.597, 4.340, 4.882,
```

```
pts = Transpose[{xLst, yLst}];
```

```
lp = ListPlot[pts, AxesLabel → {" x ", " y "  
PlotStyle → {PointSize[.03]}];
```

- Plotting data with a line through the first and last lines helps u (Figure 8.3.5).

```
ln =  
  Show[  
    Graphics[  
      Line[{{xLst[[1]], yLst[[1]]},  
           {xLst[[6]], yLst[[6]]}}]]];
```

```
Show[lp, ln];
```

```
lp = ListPlot[pts, AxesLabel → {" x ", " y "  
  PlotStyle → {PointSize[.03]}}];
```

- Squaring x coordinates does not seem quite enough (Figure 8.3

```
lp2 = ListPlot[Transpose[{xLst^2, yLst}  
  PlotStyle → {PointSize[.03]}}];
```

- Cubing x coordinates does not seem quite enough (Figure 8.3.7)

```
lpPower = ListPlot[Transpose[{xLst ^ 3, yLst},  
PlotStyle → {PointSize[.03]}];
```

- Raising x coordinates to the fourth power seems too much (Figure 8.3.8)

```
lpPower = ListPlot[Transpose[{xLst ^ 4, yLst},  
PlotStyle → {PointSize[.03]}];
```

- Raising x coordinates to the power 3.5 seems better (Figure 8.3.9)

```
lpPower = ListPlot[Transpose[{xLst ^ 3.5, yLst},  
PlotStyle → {PointSize[.03]}];
```

- Plot linear regression line with points (Figure 8.3.10)

```
fitData = Fit[Transpose[{xLst ^ 3.5, yLst},
```

```
plFit = Plot[fitData, {z, 0, 7.5}];
```

```
Show[lpPower, plFit];
```

- Always, eventually plot actual and predicted data together (Fig

```
Clear[f, x]  
f[x_] := fitData /. z -> x3.5
```

```
f[x]
```

```
plData = Plot[f[x], {x, 0, 1.7}];
```

```
Show[lp, plData];
```

```
plData = Plot[f[x], {x, 1.3, 1.7}];
```

```
Show[lp, plData];
```

Solving for y in a One-Term Model

Misra1a.dat

The data is from NIST dental research by D. Misra (1978) in monomolecular adsorption, volume.

(<http://www.itl.nist.gov/div898/strd/nls/data/Misra1a.shtml>)

■ Plot data (Figure 8.3.13)

```
xLst = {77.6, 114.9, 141.1, 190.8, 239.9,  
        378.4, 434.8, 477.3, 536.8, 593.1, 689  
yLst = {10.07, 14.73, 17.94, 23.93, 29.61,  
        44.82, 50.76, 55.05, 61.01, 66.40, 75.  
  
pts = Transpose[{xLst, yLst}];
```

```
lp = ListPlot[pts, AxesLabel → {"x", "y"},  
              PlotStyle → {PointSize[.03]}];
```

- Raising y coordinates to the power 6/5 seems linear (Figure 8.3)

```
lpPower = ListPlot[Transpose[{xLst, yLst}]];  
PlotStyle -> {PointSize[.03]}];
```

```
fitData = Fit[Transpose[{xLst, yLst6/5}]],
```

```
plFit = Plot[fitData, {z, 0, 800}];
```

```
Show[lpPower, plFit];
```

- Always, eventually plot actual and predicted data together. Sc

```
Clear[f, x]
```

```
f[x_] := (-5.48629 + 0.267869 x)5/6
```

```
f[x]
```

```
plData = Plot[f[x], {x, 77, 770}];
```

```
Show[lp, plData];
```

Multi-term Models

Filip.dat

The data is from NIST research.

(<http://www.itl.nist.gov/div898/strd/nls/data/filip.shtml>)

■ Plot data (Figure 8.3.16)

```
xLst = {-6.860120914, -4.324130045, -4.358426747, -6.955852379, -6.661145,
-6.355462942, -6.118102026, -7.115148,
-6.815308569, -6.519993057, -6.204119,
-5.853871964, -6.109523091, -5.798329,
-5.171791386, -4.851705903, -4.517126,
-4.143573228, -3.709075441, -3.499489,
-6.300769497, -5.953504836, -5.642065,
-5.031376979, -4.680685696, -4.329846,
-3.928486195, -8.56735134, -8.3632113,
-7.823908741, -7.522878745, -7.218819,
-6.920818754, -6.628932138, -6.323946,
-5.991399828, -8.781464495, -8.663140,
-8.473531488, -8.247337057, -7.971428,
-7.676129393, -7.352812702, -7.072065,
-6.774174009, -6.478861916, -6.159517,
-6.835647144, -6.53165267, -6.2240984,
-5.598599459, -5.290645224, -4.974284}
```

```
-4.290560426, -3.885055584, -3.408378  
-8.726767166, -8.66695597, -8.5110264  
-7.886056648, -7.588043762, -7.283412  
-6.995678626, -6.691862621, -6.392544  
-6.067374056, -6.684029655, -6.378719  
-6.065855188, -5.752272167, -5.132414  
-4.811352704, -4.098269308, -3.661742  
yLst = {0.8116, 0.9072, 0.9052, 0.9039, 0.  
0.8667, 0.8809, 0.7975, 0.8162, 0.8515  
0.8885, 0.8859, 0.8959, 0.8913, 0.8959  
0.9021, 0.909, 0.9139, 0.9199, 0.8692,  
0.891, 0.8977, 0.9035, 0.9078, 0.7675,  
0.7713, 0.7736, 0.7775, 0.7841, 0.7971  
0.8641, 0.8804, 0.7668, 0.7633, 0.7678  
0.77, 0.7749, 0.7796, 0.7897, 0.8131,  
0.8061, 0.846, 0.8751, 0.8856, 0.8919,  
0.894, 0.8957, 0.9047, 0.9129, 0.9209,  
0.7739, 0.7681, 0.7665, 0.7703, 0.7702  
0.7809, 0.7961, 0.8253, 0.8602, 0.8809  
0.8664, 0.8834, 0.8898, 0.8964, 0.8963  
0.9119, 0.9228};  
pts = Transpose[{xLst, yLst}];
```

```
lp = ListPlot[pts,  
PlotStyle → {PointSize[.02]}];
```

■ Model with fourth degree polynomial (Figure 8.3.17)

```
fitData = Fit[pts, {1, z, z2, z3, z4}, z
```

```
plFit = Plot[fitData, {z, -9, -3}];
```

```
Show[lp, plFit];
```

■ Model with tenth degree polynomial (Figure 8.3.18)

```
fitData =  
Fit[pts, {1, z, z2, z3, z4, z5, z6, z7, z
```

```
plFit = Plot[fitData, {z, -9, -3}];
```

```
Show[lp, plFit];
```

- Plot of tenth degree polynomial model inside and outside range

```
plFit = Plot[fitData, {z, -10, -2}];
```