**Page 113: #2**

The following table gives the elongation *e* in inches per inch (in./in.) for a given stress *S* on a steel wire measured in pounds per square in (lb/in.^2). Test the model *e = cS* by plotting the data. Estimate c graphically.

Estimating C: (255-57)/(70-20)= C = 3.96

So *e = 3.96S*

The plotted model is ok, but clearly does not sit directly on top of the data set.

|  |  |  |
| --- | --- | --- |
| S(x10^-3) | e(x10^5) | e = 3.96S |
| 5 | 0 | 19.8 |
| 10 | 19 | 39.6 |
| 20 | 57 | 79.2 |
| 30 | 94 | 118.8 |
| 40 | 134 | 158.4 |
| 50 | 173 | 198 |
| 60 | 216 | 237.6 |
| 70 | 256 | 277.2 |
| 80 | 297 | 316.8 |
| 90 | 343 | 356.4 |
| 100 | 390 | 396 |

**Page 121: #2.a**

For each of the following data sets, formulate the mathematical model that minimizes the largest deviation between the data and the line y = ax+b. If a computer is available, solve for the estimates of a and b.

This would be the Chebyshev approximation criterion.

|  |  |  |
| --- | --- | --- |
| x | y | r |
| 1 | 3.6 | 1 |
| 2.3 | 3 | 2 |
| 3.7 | 3.2 | 3 |
| 4.2 | 5.1 | 4 |
| 6.1 | 5.3 | 5 |
| 7 | 6.8 | 6 |

Minimize *r (largest residual)* subject to:

**Page 127: #10**

Fit the data with the models given, using least squares.

Model:

=

Calculations:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Body | distance (x) | period (y) | |  | | --- | | /Users/andrew/Library/Group Containers/UBF8T346G9.Office/msoclip1/01/clip_image001.png | | |  | | --- | | /Users/andrew/Library/Group Containers/UBF8T346G9.Office/msoclip1/01/clip_image001.png | | y=ax^(3/2) |
| Mercury | 57900000000 | 7600000 | 1.05884E+23 | 1.94105E+32 | 7607582.623 |
| Venus | 1.08E+11 | 19400000 | 6.88553E+23 | 1.25971E+33 | 19380477.16 |
| Earth | 1.5E+11 | 31600000 | 1.83579E+24 | 3.375E+33 | 31722376.9 |
| Mars | 2.28E+11 | 59400000 | 6.46679E+24 | 1.18524E+34 | 59447158.77 |
| Jupiter | 7.79E+11 | 374000000 | 2.57145E+26 | 4.72729E+35 | 375435225.9 |
| Saturn | 1.43E+12 | 935000000 | 1.59888E+27 | 2.92421E+36 | 933754888.3 |
| Uranus | 2.87E+12 | 2640000000 | 1.28359E+28 | 2.36399E+37 | 2654921456 |
| Neptune | 4.5E+12 | 5220000000 | 4.98298E+28 | 9.1125E+37 | 5212518422 |

Able to fit the data pretty tightly !

**Page 136: #7**

In the following data, W represents the weight of a fish (bass) and *l* represents its length. Fit the model *W=kl^3* to the data using the least squares criterion.

1. **Model:**

=

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| length | weight | /Users/andrew/Library/Group Containers/UBF8T346G9.Office/msoclip1/01/clip_image001.png   |  | | --- | |  | | /Users/andrew/Library/Group Containers/UBF8T346G9.Office/msoclip1/01/clip_image001.png   |  | | --- | |  | |  |
| 14.5 | 27 | 82312.875 | 9294114.391 |  |
| 12.5 | 17 | 33203.125 | 3814697.266 |  |
| 17.25 | 41 | 210451.0781 | 26347207.78 |  |
| 14.5 | 26 | 79264.25 | 9294114.391 |  |
| 12.625 | 17 | 34209.21289 | 4049378.016 |  |
| 17.75 | 49 | 274025.6094 | 31274483.38 |  |
| 14.125 | 23 | 64817.63867 | 7942015.658 |  |
| 12.625 | 16 | 32196.90625 | 4049378.016 |  |
|  | sum: | 810480.6953 | 96065388.9 | 0.008436761 |

1. **Model:**

=

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| length | weight | girth | wl^2g | l^4g^2 |  |
| 14.5 | 27 | 9.75 | 55348.3125 | 4202243.754 |  |
| 12.5 | 17 | 8.375 | 22246.09375 | 1712417.603 |  |
| 17.25 | 41 | 11 | 134200.6875 | 10713756.41 |  |
| 14.5 | 26 | 9.75 | 53298.375 | 4202243.754 |  |
| 12.625 | 17 | 8.5 | 23031.94531 | 1835538.079 |  |
| 17.75 | 49 | 12.5 | 192975.7813 | 15510059.2 |  |
| 14.125 | 23 | 9 | 41299.73438 | 3224325.254 |  |
| 12.625 | 16 | 8.5 | 21677.125 | 1835538.079 |  |
|  |  | sum: | 544078.0547 | 43236122.14 | 0.012583877 |

c. Which of the two models fits the data better? Justify fully. Which model do you prefer? Why?

It appears that fits better than based on residuals (SSE of 3.4 vs 30.5 respectively) The chart also shows the model that includes girth has more overlap than without.

I also prefer the model that includes girth, as it takes into account the fatness of the fish, which as we can see, is not entirely proportional to length.

|  |  |  |  |
| --- | --- | --- | --- |
| length | weight | sse .008l^3 | sse .0126l^2g |
| 14.5 | 27 | 6.817321 | 1.37074337 |
| 12.5 | 17 | 1.890625 | 0.261856079 |
| 17.25 | 41 | 0.004048141 | 0.058642676 |
| 14.5 | 26 | 2.595321 | 0.02916837 |
| 12.625 | 17 | 0.812786768 | 0.005003573 |
| 17.75 | 49 | 18.15718627 | 0.387311743 |
| 14.125 | 23 | 0.206783352 | 0.140571099 |
| 12.625 | 16 | 0.009693018 | 1.146475448 |
|  | totals: | 30.49376454 | 3.399772359 |

Page 146: #5

Solve problems 1-4 with the model V = m(logP) + b. Compare the errors with those computed in Problem 4. Compare the two models. Which is better?

1. Fit the model V = CP^a to the “pace of life” data in Table 4.5 Use the transformation logV = alogP + logC. Plot logV versus logP. Does the relationship seem reasonable?

|  |  |  |  |
| --- | --- | --- | --- |
| pop p | mean velocity v (ft/sec) | logP | logV |
| 341,948 | 4.81 | 5.533960068 | 0.682145076 |
| 1092759 | 5.88 | 6.038524392 | 0.769377326 |
| 5491 | 3.31 | 3.739651444 | 0.519827994 |
| 49375 | 4.9 | 4.693507109 | 0.69019608 |
| 1340000 | 5.62 | 6.127104798 | 0.749736316 |
| 365 | 2.76 | 2.562292864 | 0.440909082 |
| 2500 | 2.27 | 3.397940009 | 0.356025857 |
| 78200 | 3.85 | 4.893206753 | 0.58546073 |
| 867023 | 5.21 | 5.938030618 | 0.716837723 |
| 14000 | 3.7 | 4.146128036 | 0.568201724 |
| 23700 | 3.27 | 4.374748346 | 0.514547753 |
| 70700 | 4.31 | 4.849419414 | 0.63447727 |
| 304500 | 4.42 | 5.483587297 | 0.645422269 |
| 138000 | 4.39 | 5.139879086 | 0.64246452 |
| 2602000 | 5.05 | 6.415307292 | 0.703291378 |

* 1. Make a table of logP versus logV
  2. Construct a scatter plot of your log-log data
  3. Eyeball a line onto your scatterplot
  4. Estimate the slope and the intercept.

**Slope: .5/2.5 = .12**

**Intercept: .05**

* 1. Find the linear equation that relates logV and logP

Log v = .12 LogP + .05

* 1. Find equation of the form …that expresses V in terms of P

4. Average of Bornstein errors = .479625. I don’t remember using average absolute error (it’s usually error squared?), but it suggests that its an adequate fit, with potentially more room to improve.

|  |  |  |  |
| --- | --- | --- | --- |
| pop p | mean velocity v (ft/sec) | 1.122P^12 | Obs - Pred |
| 341,948 | 4.81 | 5.176879606 | 0.366879606 |
| 1092759 | 5.88 | 5.951354002 | 0.071354002 |
| 5491 | 3.31 | 3.153196459 | 0.156803541 |
| 49375 | 4.9 | 4.104060081 | 0.795939919 |
| 1340000 | 5.62 | 6.09881462 | 0.47881462 |
| 365 | 2.76 | 2.277550423 | 0.482449577 |
| 2500 | 2.27 | 2.869099818 | 0.599099818 |
| 78200 | 3.85 | 4.336882614 | 0.486882614 |
| 867023 | 5.21 | 5.788373294 | 0.578373294 |
| 14000 | 3.7 | 3.527997114 | 0.172002886 |
| 23700 | 3.27 | 3.758050904 | 0.488050904 |
| 70700 | 4.31 | 4.2847273 | 0.0252727 |
| 304500 | 4.42 | 5.105324277 | 0.685324277 |
| 138000 | 4.39 | 4.642782591 | 0.252782591 |
| 2602000 | 5.05 | 6.604344696 | 1.554344696 |
|  |  |  | 0.479625003 |

**Now using V=m(logP) + B:**

Looks like intercept goes through origin, with slope of about .9, so

**V = .9(logP)**

Comparing the average of the absolute errors, it looks like this second model (.374 avg abs error) fits better than the original (.480)

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| pop p | mean velocity v (ft/sec) | M1 1.122P^12 | M1 Obs - Pred | M2 V =.9(logP) | M2 Obs - Pred |
| 341,948 | 4.81 | 5.176879606 | 0.366879606 | 4.980564061 | 0.170564061 |
| 1092759 | 5.88 | 5.951354002 | 0.071354002 | 5.434671953 | 0.445328047 |
| 5491 | 3.31 | 3.153196459 | 0.156803541 | 3.365686299 | 0.055686299 |
| 49375 | 4.9 | 4.104060081 | 0.795939919 | 4.224156398 | 0.675843602 |
| 1340000 | 5.62 | 6.09881462 | 0.47881462 | 5.514394319 | 0.105605681 |
| 365 | 2.76 | 2.277550423 | 0.482449577 | 2.306063578 | 0.453936422 |
| 2500 | 2.27 | 2.869099818 | 0.599099818 | 3.058146008 | 0.788146008 |
| 78200 | 3.85 | 4.336882614 | 0.486882614 | 4.403886078 | 0.553886078 |
| 867023 | 5.21 | 5.788373294 | 0.578373294 | 5.344227557 | 0.134227557 |
| 14000 | 3.7 | 3.527997114 | 0.172002886 | 3.731515232 | 0.031515232 |
| 23700 | 3.27 | 3.758050904 | 0.488050904 | 3.937273511 | 0.667273511 |
| 70700 | 4.31 | 4.2847273 | 0.0252727 | 4.364477472 | 0.054477472 |
| 304500 | 4.42 | 5.105324277 | 0.685324277 | 4.935228567 | 0.515228567 |
| 138000 | 4.39 | 4.642782591 | 0.252782591 | 4.625891178 | 0.235891178 |
| 2602000 | 5.05 | 6.604344696 | 1.554344696 | 5.773776563 | 0.723776563 |
|  |  |  | 0.479625003 |  | 0.374092419 |

    Page 157: #4

In the following data, X represents the diameter of a ponderosa pine measured at breast height, and Y is a measure of volume—number of board feet divided by 10. Make a scatterplot of the data. Discuss the appropriateness of using a 13th-degree polynomial that passes through the data points as an empirical model. If you have a computer available, fit a polynomial to the data and graph the results.

There’s 14 data points, so a 13th degree polynomial is the maximum amount allowed, but it sounds like overkill, especially since the data appears to be following a linear or more basic exponential curve.

Work in r to get the coefficients:

xDia <- c(17,19,20,22,23,25,31,32,33,36,37,38,39,41)

yVol <- c(19,25,32,51,57,71,141,123,187,192,205,252,248,294)

> fit <- lm(yVol ~ poly(xDia, 13, raw=TRUE))

> fit

> summary(fit)

Call:

lm(formula = yVol ~ poly(xDia, 13, raw = TRUE))

Residuals:

1 2 3 4 5 6 7 8 9 10 11

-0.0777 1.2350 -2.4560 3.2869 -1.7901 -0.6550 13.4945 -29.2786 18.3321 0.6008 -9.5549

12 13 14

10.2363 -3.5314 0.1583

Coefficients: (3 not defined because of singularities)

Estimate Std. Error t value Pr(>|t|)

(Intercept) 1.670e+07 2.823e+07 0.592 0.596

poly(xDia, 13, raw = TRUE)1 -6.202e+06 1.031e+07 -0.602 0.590

poly(xDia, 13, raw = TRUE)2 1.018e+06 1.666e+06 0.611 0.584

poly(xDia, 13, raw = TRUE)3 -9.699e+04 1.563e+05 -0.621 0.579

poly(xDia, 13, raw = TRUE)4 5.906e+03 9.383e+03 0.629 0.574

poly(xDia, 13, raw = TRUE)5 -2.380e+02 3.732e+02 -0.638 0.569

poly(xDia, 13, raw = TRUE)6 6.309e+00 9.774e+00 0.646 0.565

poly(xDia, 13, raw = TRUE)7 -1.038e-01 1.590e-01 -0.653 0.561

poly(xDia, 13, raw = TRUE)8 8.663e-04 1.315e-03 0.659 0.557

poly(xDia, 13, raw = TRUE)9 NA NA NA NA

poly(xDia, 13, raw = TRUE)10 -4.741e-08 7.078e-08 -0.670 0.551

poly(xDia, 13, raw = TRUE)11 NA NA NA NA

poly(xDia, 13, raw = TRUE)12 2.555e-12 3.768e-12 0.678 0.546

poly(xDia, 13, raw = TRUE)13 NA NA NA NA

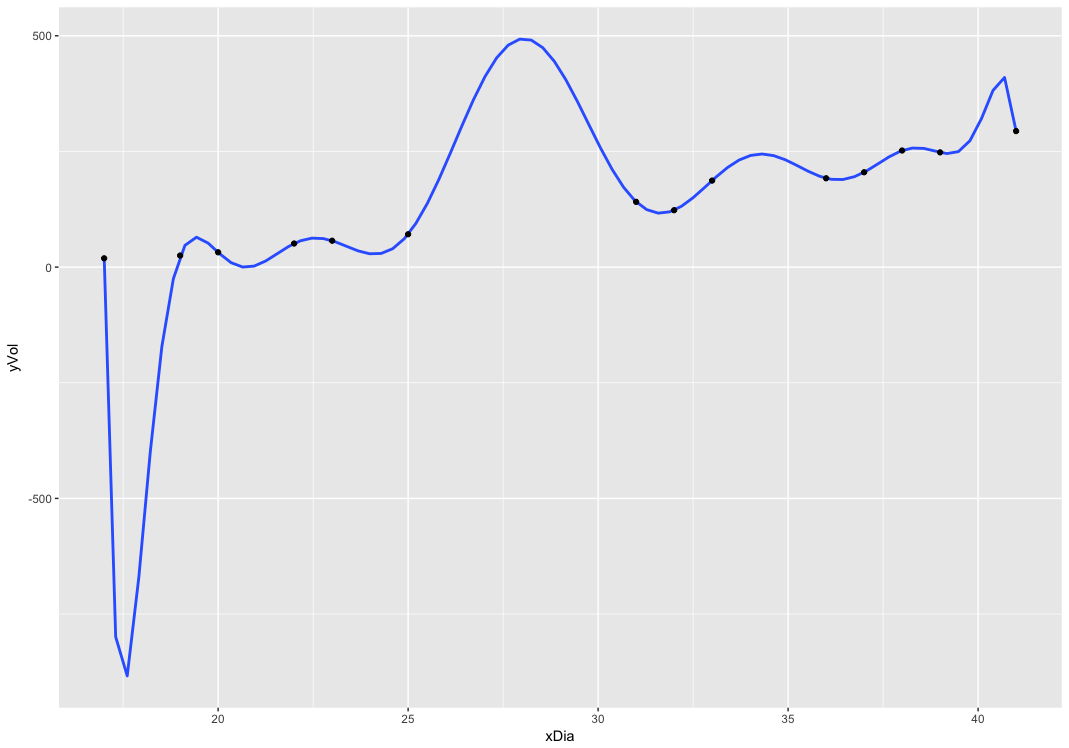
Residual standard error: 23.14 on 3 degrees of freedom

Multiple R-squared: 0.9862, Adjusted R-squared: 0.9401

F-statistic: 21.4 on 10 and 3 DF, p-value: 0.01419

g <- ggplot(p154, aes(xDia, yVol))

g + stat\_smooth(method="lm", formula = y ~ poly(x,13)) + geom\_point()



So yes, its very clearly overkill and overfitted, here.

   Page 169: #11

The following data represent the length of a bass fish and its weight.

Construct a scatterplot of the given data.

Is there a trend in the data?

There appears to be an upward sloping nearly linear pattern

Are any of the data points outliers?

No obvious outliers, but an argument can be made for the point on the far right

Construct a divided difference table.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| length (in) | weight (oz) | 1st order | 2nd order | 3rd order | 4th order | 5th order |
| 12.5 | 17 |  |  |  |  |  |
|  |  | -4 |  |  |  |  |
| 12.625 | 16.5 |  | 5.128205128 |  |  |  |
|  |  | 4.333333333 |  | -1.230769231 |  |  |
| 14.125 | 23 |  | 2.666666667 |  | 0.078577392 |  |
|  |  | 9.333333333 |  | -0.857526618 |  | 0.064067223 |
| 14.5 | 26.5 |  | -1.299393939 |  | 0.414930313 |  |
|  |  | 5.272727273 |  | 1.268991239 |  |  |
| 17.25 | 41 |  | 3.300699301 |  |  |  |
|  |  | 16 |  |  |  |  |
| 17.75 | 49 |  |  |  |  |  |

Is smoothing with a low-order polynomial appropriate? If so, choose an appropriate polynomial and fit using the least-squares criterion of best fit.

I can’t quite tell if soothing is appropriate here, since the 5th order divided difference is still small, but not negligible. So, I suppose I shall fit both a 4th and 5th polynomial!

4th order polynomial:

> summary(fit4)

Call:

lm(formula = yWei ~ poly(xLen, 4, raw = TRUE))

Residuals:

1 2 3 4 5 6

-0.15828 0.19243 -0.15486 0.12759 -0.01699 0.01012

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) 1.846e+04 3.041e+03 6.070 0.1040

poly(xLen, 4, raw = TRUE)1 -4.967e+03 8.049e+02 -6.170 0.1023

poly(xLen, 4, raw = TRUE)2 4.985e+02 7.939e+01 6.279 0.1005

poly(xLen, 4, raw = TRUE)3 -2.211e+01 3.459e+00 -6.393 0.0988 .

poly(xLen, 4, raw = TRUE)4 3.663e-01 5.619e-02 6.518 0.0969 .

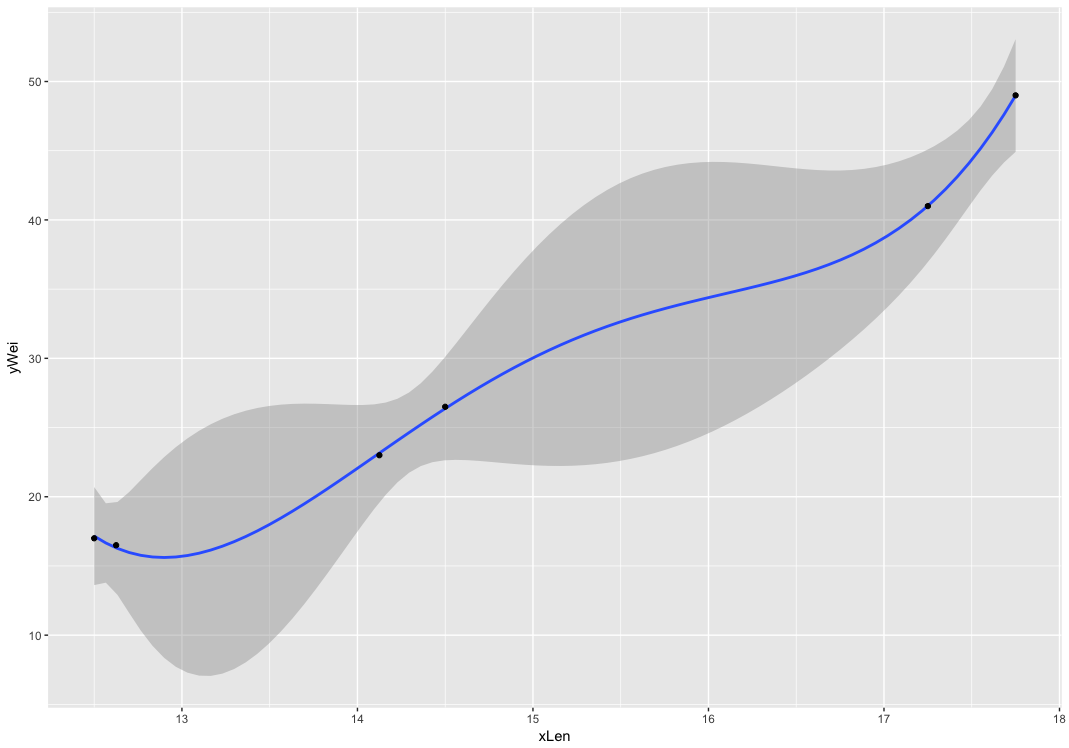
---

Signif. codes: 0 ‘\*\*\*’ 0.001 ‘\*\*’ 0.01 ‘\*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.3205 on 1 degrees of freedom

Multiple R-squared: 0.9999, Adjusted R-squared: 0.9994

F-statistic: 2157 on 4 and 1 DF, p-value: 0.01615



5th order:

> summary(fit5)

Call:

lm(formula = yWei ~ poly(xLen, 5, raw = TRUE))

Residuals:

ALL 6 residuals are 0: no residual degrees of freedom!

Coefficients:

Estimate Std. Error t value Pr(>|t|)

(Intercept) -2.956e+04 NA NA NA

poly(xLen, 5, raw = TRUE)1 1.123e+04 NA NA NA

poly(xLen, 5, raw = TRUE)2 -1.676e+03 NA NA NA

poly(xLen, 5, raw = TRUE)3 1.233e+02 NA NA NA

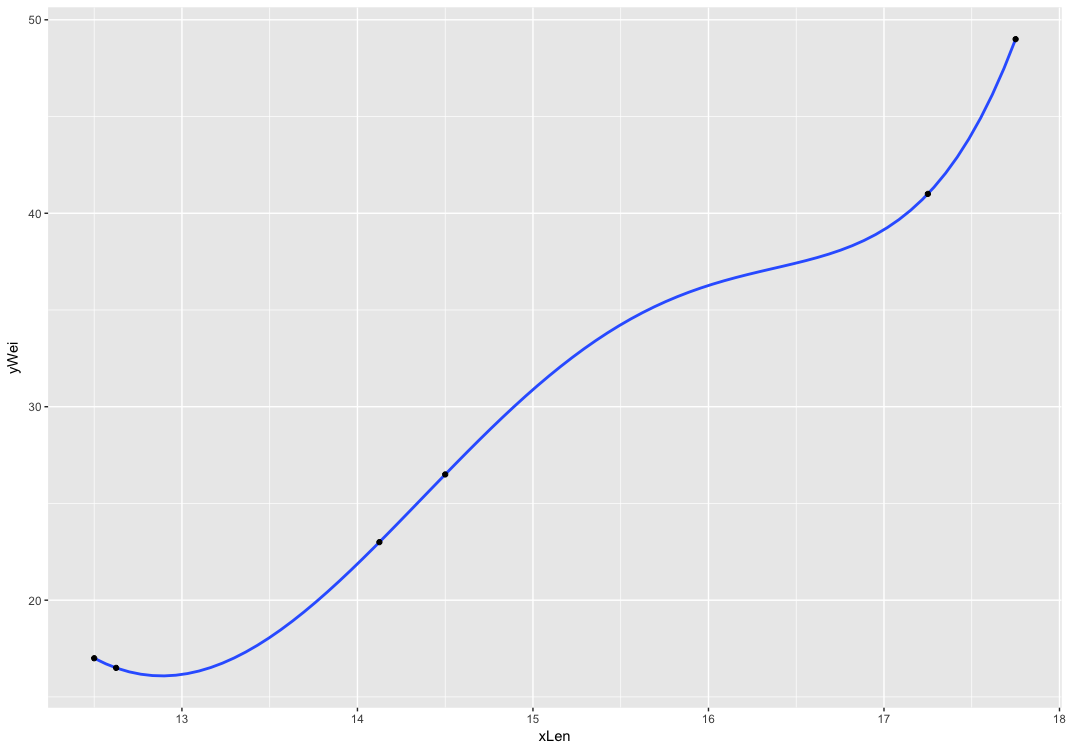
poly(xLen, 5, raw = TRUE)4 -4.470e+00 NA NA NA

poly(xLen, 5, raw = TRUE)5 6.407e-02 NA NA NA

Residual standard error: NaN on 0 degrees of freedom

Multiple R-squared: 1, Adjusted R-squared: NaN

F-statistic: NaN on 5 and 0 DF, p-value: NA



Analyze the goodness of fit by examining appropriate indicators and graphing the model, the data points, and the deviations.

Graphs are above. They both fit the data points well, although not sure what’s wrong with the 5th order data calculations in r.

Just to go through the calculations, I’ll do the 4th order:

W(x) = 18460 – 4967l + .498.5l^2 – 22.11l^3 + .3663l^4

I tried to calculate this out, but it wasn’t accurate as I didn’t have the data to enough places.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| weight (oz) | length (in) | 18460 | -4967 | 498.5 | -22.11 | 0.3663 |  |
| 17 | 12.5 | 18460 | -62087.5 | 77890.625 | -43183.59375 | 8942.871094 | 22.40234375 |
| 16.5 | 12.625 | 18460 | -62708.375 | 79456.22656 | -44492.09982 | 9305.987521 | 21.73925935 |
| 23 | 14.125 | 18460 | -70158.875 | 99458.53906 | -62309.47787 | 14581.11532 | 31.3015074 |
| 26.5 | 14.5 | 18460 | -72021.5 | 104809.625 | -67405.09875 | 16192.31439 | 35.34064375 |
| 41 | 17.25 | 18460 | -85680.75 | 148334.9063 | -113489.5936 | 32433.46259 | 58.02524336 |
| 49 | 17.75 | 18460 | -88164.25 | 157058.6563 | -123647.0658 | 36360.54199 | 67.88246211 |

It has multiple r squared at .99999, so that’s a pretty tight fit!

Page 181: #5

*The cost of a postage stamp*—Consider the following data. Use the procudes in this chapter to capture the trend of the data if one exists.

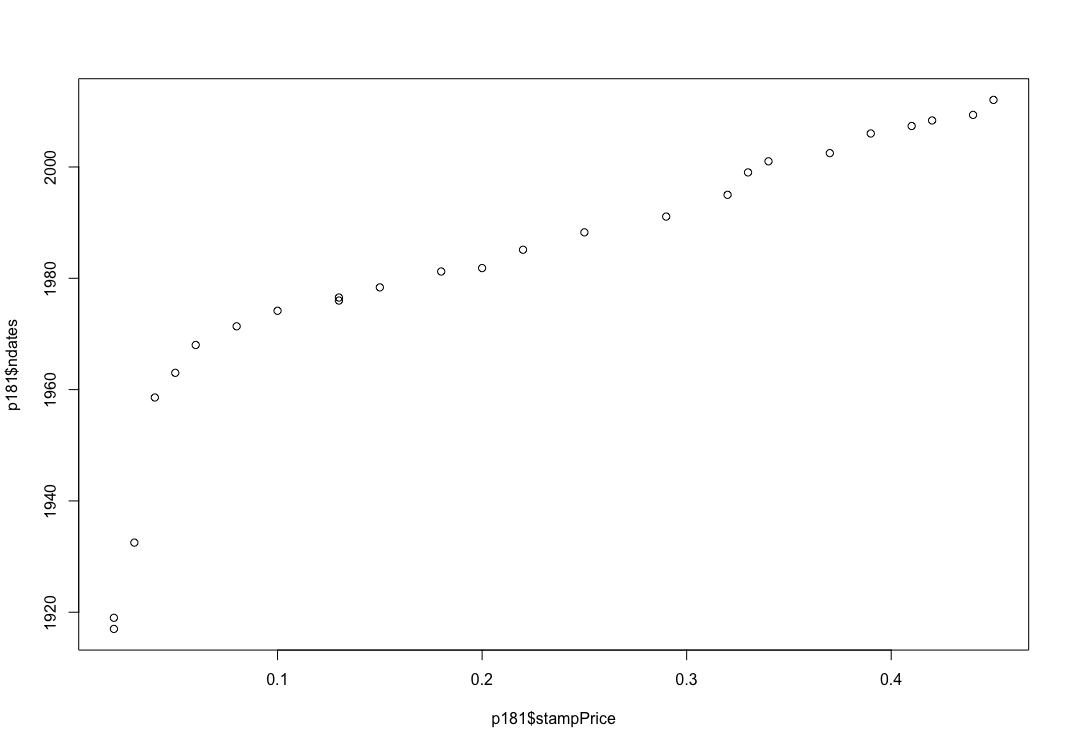
Taking a look at the scatter plot, there is a clear trend. The first three dates are kind of annoying though since they don’t have specific days and months, so I’m going to take them out.

dates <- c('1917', '1919', '06jul1932', '01aug1958', '07jan1963', '07jan1968', '16may1971', '02mar1974', '31dec1975', '18jul1976', '15may1978', '22mar1981', '01nov1981', '17feb1985', '03apr1988', '03feb1991', '01jan1995', '10jan1999', '07jan2001','30jun2002', '08jan2006', '14may2007', '12may2008', '11may2009', '22jan2012')

ndates <- as.Date(dates, "%d%B%Y")

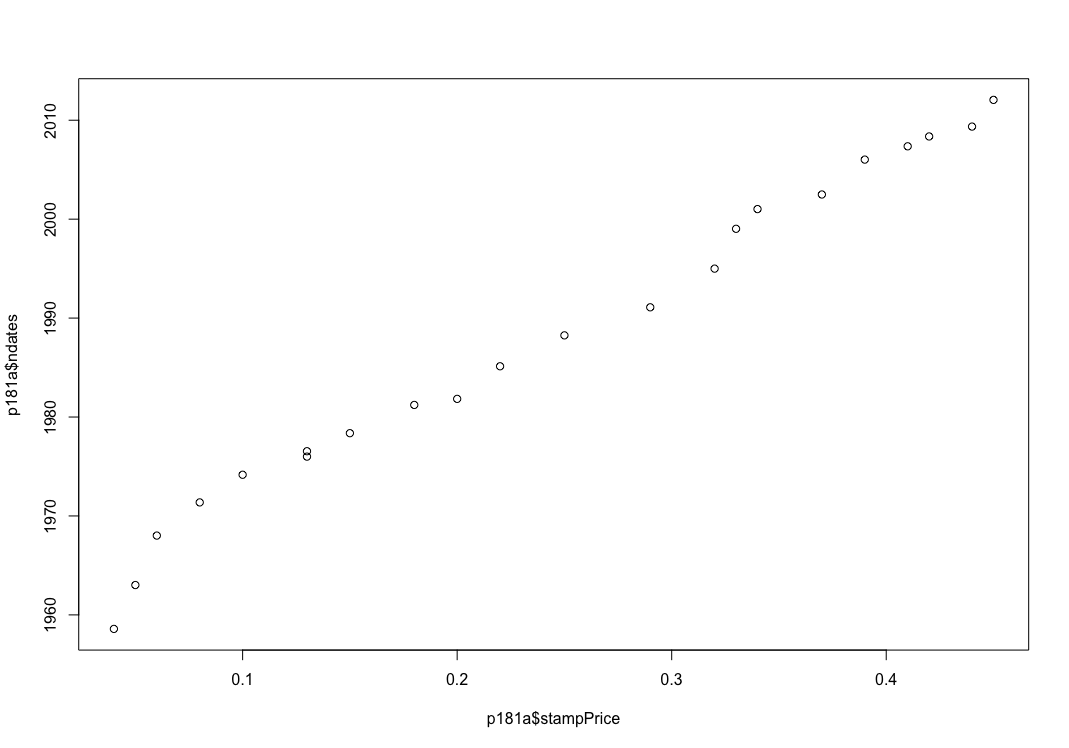
ndates[1] <- "1917-01-01"

ndates[2] <- "1919-01-01"

stampPrice <- c(.02, .02, .03, .04, .05, .06, .08, .10, .13, .13, .15, .18, .20, .22, .25, .29, .32, .33, .34, .37, .39, .41, .42, .44, .45)

p181 <- data.frame(ndates, stampPrice)

plot(p181$stampPrice, p181$ndates)



p181a <- p181[4:25,]

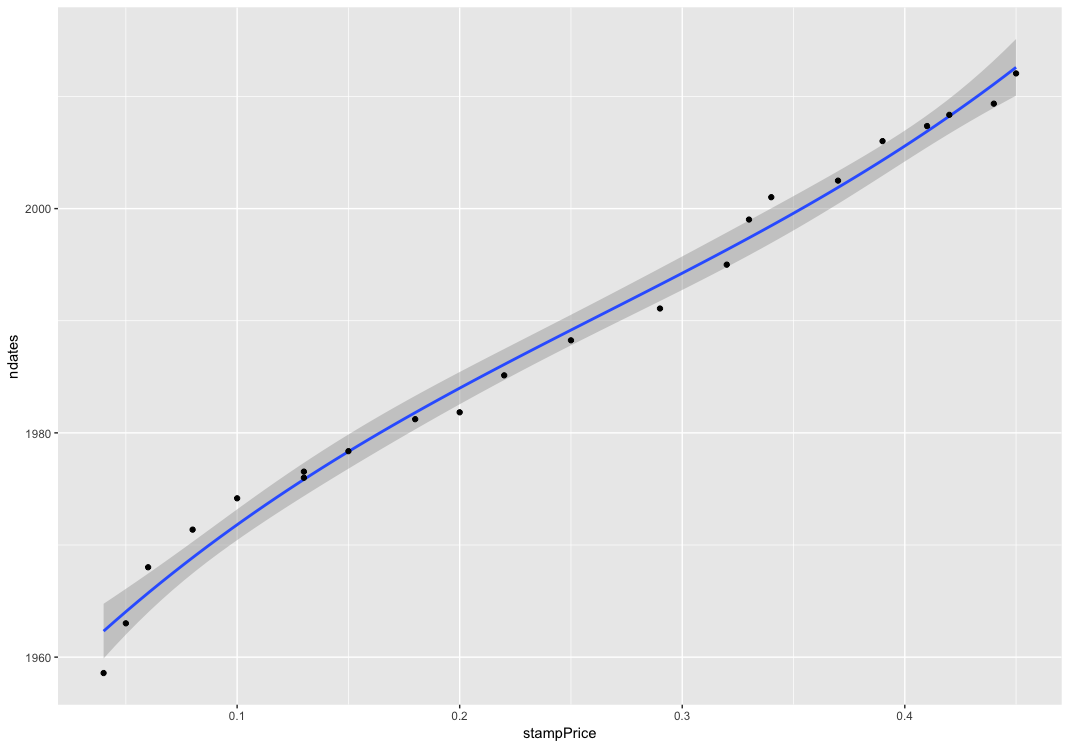
plot(p181a$stampPrice, p181a$ndates)

This looks like a more extended polynomial curve, more sophisticated than a quadratic. So I suppose I’ll play around with a few polynomials till I’m adequately satisfied.

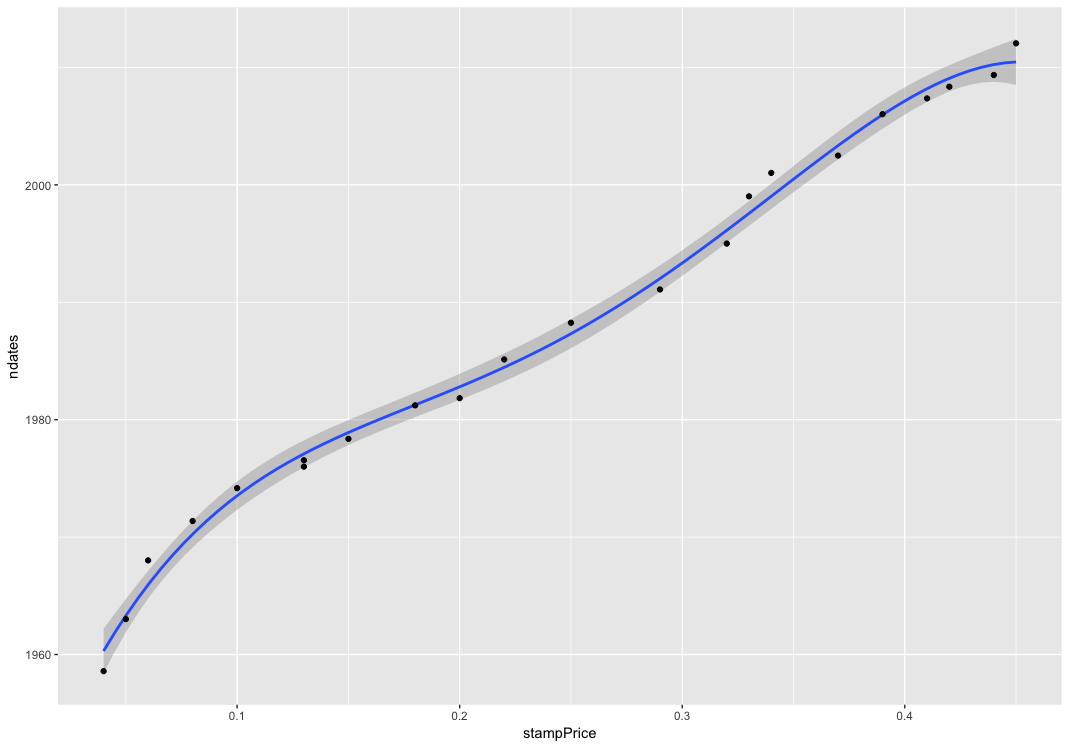
3rd order polynomial:

g <- ggplot(p181a, aes(stampPrice, ndates))

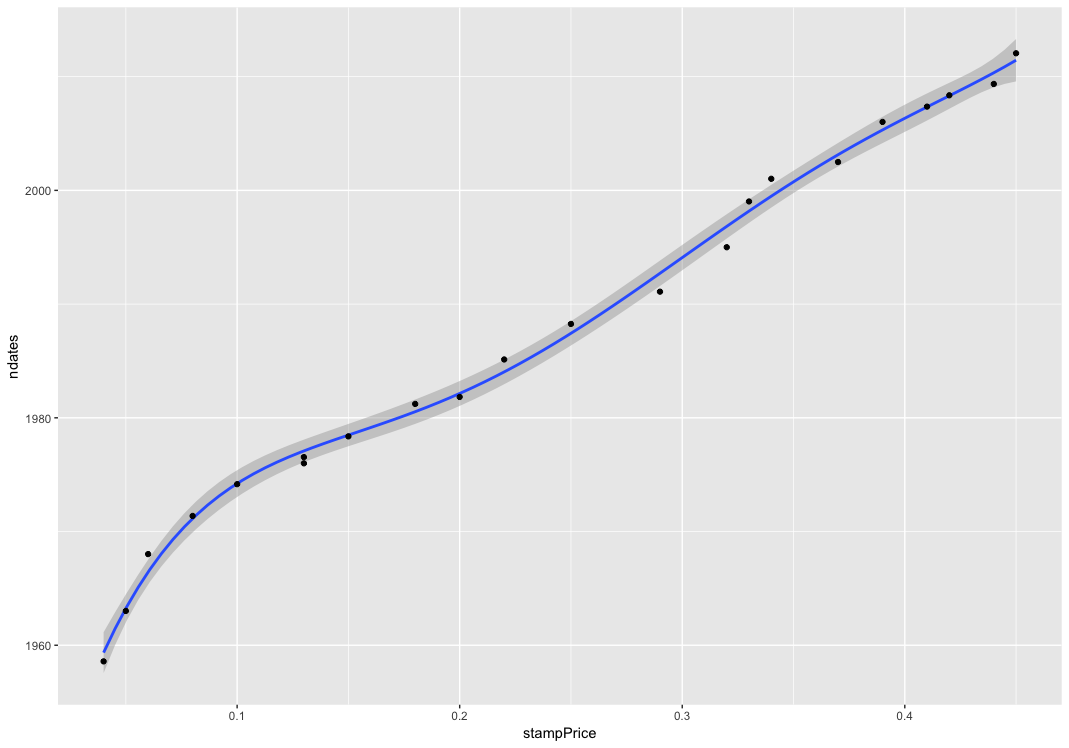
g + stat\_smooth(method="lm", formula = y ~ poly(x,3)) + geom\_point()



4th order:



5th order



The 5th order wasn’t a large improvement, so I’ll stay with 4th order for now.