Linear Regression and Non-Linear Regression

Andrew Long

February 27, 2020

Abstract

There are two fundamental ideas at work here. The first is finding a best-fit to an over-determined problem (over-determined means that we have too many constraints for our ability to fit a given model).

The second essential idea is that of using a linear approximation or method to attempt a solution to a non-linear problem. We will rely on linear approximations, and iteration, hoping to get a good approximation to the true solution of the non-linear problem.

1 Linear regression

Consider a set of data values of the form $\{x_i, y_i\}_{i=1,\ldots,n}$. We think of y as a function of x, i.e. $y = f(x;\theta)$, and seek to estimate the optimal parameters θ of the model $f(x;\theta)$. For example, f might be parameterized by a slope m and an intercept b, as in

$$y = f(x;\theta) = mx + b$$

Then θ would be the vector

$$\theta = \left[\begin{array}{c} m \\ b \end{array}\right]$$

We anticipate the presence of error, often assumed to be of the form

$$y_i = mx_i + b + \epsilon_i$$

The upshot is that the error makes the data straddle the line (rather than fit it exactly).

We generally try to find the parameters using the principle of “least squares”: that is, we try to minimize the “sum of the squared errors”, or the function

$$S(\left[\begin{array}{c} m \\ b \end{array}\right]) = \sum_{i=1}^{n} \epsilon_i^2 = \sum_{i=1}^{n} (y_i - (mx_i + b))^2$$

If we take partial derivatives of this expression with respect to the parameters, and set them to zero, we obtain two equations:

$$-2\sum_{i=1}^{n}(y_i - (mx_i + b))x_i = 0 \text{ and } \quad -2\sum_{i=1}^{n}(y_i - (mx_i + b)) = 0$$

They look a lot simpler in vector form, however:

$$[\mathbf{x}\ \mathbf{1}]'\mathbf{y} = [\mathbf{x}\ \mathbf{1}]'[\mathbf{x}\ \mathbf{1}]\left[\begin{array}{c} m \\ b \end{array}\right]$$

If we define

$$X = [\mathbf{x}\ \mathbf{1}]$$

and write the vector of parameters as θ, then we can write the system more succinctly as

$$X'\mathbf{y} = X'X\theta$$

With any luck, the matrix product $X'X$ is invertible, so, formally, the parameters are estimated to be

$$\theta = (X'X)^{-1}X'\mathbf{y}$$

This form readily generalizes, of course, to the case where there are p independent predictor variables, rather than the single variable x. If we include the "one vector" $\mathbf{1}$, then we will have an intercept term in the linear model; otherwise, no.

2 Non-Linear Regression

2.1 Newton's Method

We consider a variation of non-linear regression, which is essentially a multivariate form of Newton's method; so we begin there. The idea behind Newton's method is an important one: we attempt to solve a non-linear problem by successive linear approximations. That is, we will solve a linear problem to approach the solution of the non-linear problem; then we do it again, and again, and again – until satisfied.

Newton's method is specifically a procedure designed to iteratively approach the root of a non-linear function. Specifically, we attempt to find the root of a function $f(x)$ by

a. starting with a good guess x_0, and

b. iteratively improving that guess.

So how does one "improve iteratively"? We use the linearization of f about our initial guess x_0:

$$y - f(x_0) = f'(x_0)(x - x_0)$$

Set $y = 0$, and solve for x:

$$x = x_0 - \frac{f(x_0)}{f'(x_0)}$$

This is an iterative scheme for successive improvement of our initial guess. It might converge to a true solution of the non-linear problem, which is our hope.

2.2 Non-linear Regression using Taylor Series Expansion

The linearization

$$y - f(x_0) = f'(x_0)(x - x_0)$$

can be brought to bear in our regression problem, as follows: we seek a fit to the data using the model form given by the function f, with parameters θ. That is, for a given data location i, we have

$$y_i = f(\mathbf{x}_i; \theta) + \epsilon_i$$

Once again our objective is to minimize a sum of squared errors over n data locations:

$$S(\theta) = \sum_{i=1}^{n} \epsilon_i^2 = \sum_{i=1}^{n} \left(y_i - f(\mathbf{x}_i; \theta)\right)^2$$

If we take partials of S with respect to the p parameters, we obtain p equations, such as

$$\frac{\partial S(\theta)}{\partial \theta_j} = -2 \sum_{i=1}^{n} \left(y_i - f(\mathbf{x}_i; \theta)\right) \frac{\partial f(\mathbf{x}_i; \theta)}{\partial \theta_j}$$

We then set them equal to zero and **hope** to find a global minimum (there is no guarantee).

Suppose that we have an initial guess for the parameters, $\theta_{\mathbf{0}}$, and are interested in improving it. The trick to make use of this result to find an improvement to $\theta_{\mathbf{0}}$. Once again, the trick is to use the linearization, and to use our guess $\theta_{\mathbf{0}}$.

We replace $f(\mathbf{x}_i; \theta)$ in the summation by the linearization of f **with respect to the** p **parameters** θ_j of θ:

$$f(\mathbf{x}_i; \theta) = f(\mathbf{x}_i; \theta_{\mathbf{0}}) + \sum_{k=1}^{p} \frac{\partial f(\mathbf{x}_i; \theta)}{\partial \theta_k}|_{\theta=\theta_{\mathbf{0}}}(\theta_k - \theta_{k0})$$

Then

$$0 = \sum_{i=1}^{n} \left(y_i - f(\mathbf{x}_i; \theta_{\mathbf{0}}) - \sum_{k=1}^{p} \frac{\partial f(\mathbf{x}_i; \theta)}{\partial \theta_k}|_{\theta=\theta_{\mathbf{0}}}(\theta_k - \theta_{k0}) \right) \frac{\partial f(\mathbf{x}_i; \theta)}{\partial \theta_j}|_{\theta=\theta_{\mathbf{0}}}$$

There are p equations (one for each of the p parameters). The only things unknown in these systems of equations are the θ_j (that is, the vector θ). This leads to a linear system of the form

$$[Z_i]'(y_i - f(\mathbf{x}_i; \theta_{\mathbf{0}}) = [Z_i]'[Z_i](\theta - \theta_{\mathbf{0}})$$

where

$$[Z_i] = \left[\frac{\partial f(\mathbf{x}_i;\theta)}{\partial\theta_1}, \ldots, \frac{\partial f(\mathbf{x}_i;\theta)}{\partial\theta_p}\right]_{\theta=\theta_{\mathbf{0}}}$$

the row-vector of partials evaluated at the i^{th} data location and using the parameter estimates $\theta_{\mathbf{0}}$.

When we combine these systems for each of the n data locations, we end up with the linear system

$$Z'(\mathbf{y} - \mathbf{f}(.,\theta_{\mathbf{0}})) = Z'Z(\theta - \theta_{\mathbf{0}})$$

where by $\mathbf{f}(.,\theta_{\mathbf{0}})$ we mean the model form evaluated at the n data locations, with the current best parameter estimates $\theta_{\mathbf{0}}$.

Our revised estimate for the parameters is thus given formally as

$$\theta = (Z'Z)^{-1}Z'(\mathbf{y} - \mathbf{f}(.,\theta_{\mathbf{0}})) + \theta_{\mathbf{0}}$$

An alternative way to derive this same system of equations (again based on the linearization) is as follows: assuming that

$$y_i = f(\mathbf{x}_i;\theta),$$

we have that

$$w_i \equiv y_i - f(\mathbf{x}_i;\theta_{\mathbf{0}}) = \sum_{i=1}^{p} \frac{\partial f(\mathbf{x}_i;\theta)}{\partial\theta_i}|_{\theta=\theta_{\mathbf{0}}}(\theta_i - \theta_{i0})$$

which is better written in matrix form as

$$\mathbf{w} = Z\beta$$

where Z is a constant matrix, and $\beta = \theta - \theta_{\mathbf{0}}$.

This is just a linear regression problem, which we solve for β:

$$\beta = (Z'Z)^{-1}Z'\mathbf{w}$$

and then our next estimate for θ is given by

$$\theta = \beta + \theta_{\mathbf{0}}$$

Now iterate, as long as we're converging....

3 Case Study

When we die, our bodies become rigid (*rigor mortis* sets in). Niderkorn's (1872) observations on 113 bodies provides the main reference database for the development of *rigor mortis* and is commonly cited in textbooks. Via a series of log transformations I was able to fit a lovely model to this somewhat unlovely data, for the proportion $p(t)$ of bodies in *rigor* *mortis* after t hours. It is illustrated in the graph below: the general two-parameter model that I tried is

$$p(t) = e^{\left(-\alpha/t^{\beta}\right)}$$

I don't know how I thought of this, except that we'd been looking at models of compositions with exponentials, and I had a feeling that this sort of model might work. I wanted a model with an asymptote of 1, for sure. I also wanted the function to have a zero derivative at the origin, and to remain flat for awhile. Model building is a mysterious and black art. What model might you propose?

Models are often compared against each other based on the number of parameters they require, how much predictive power they have, whether they make intuitive sense, whether the parameters are interpretable, etc. When you find your model bring it over to my house, and we'll race them!

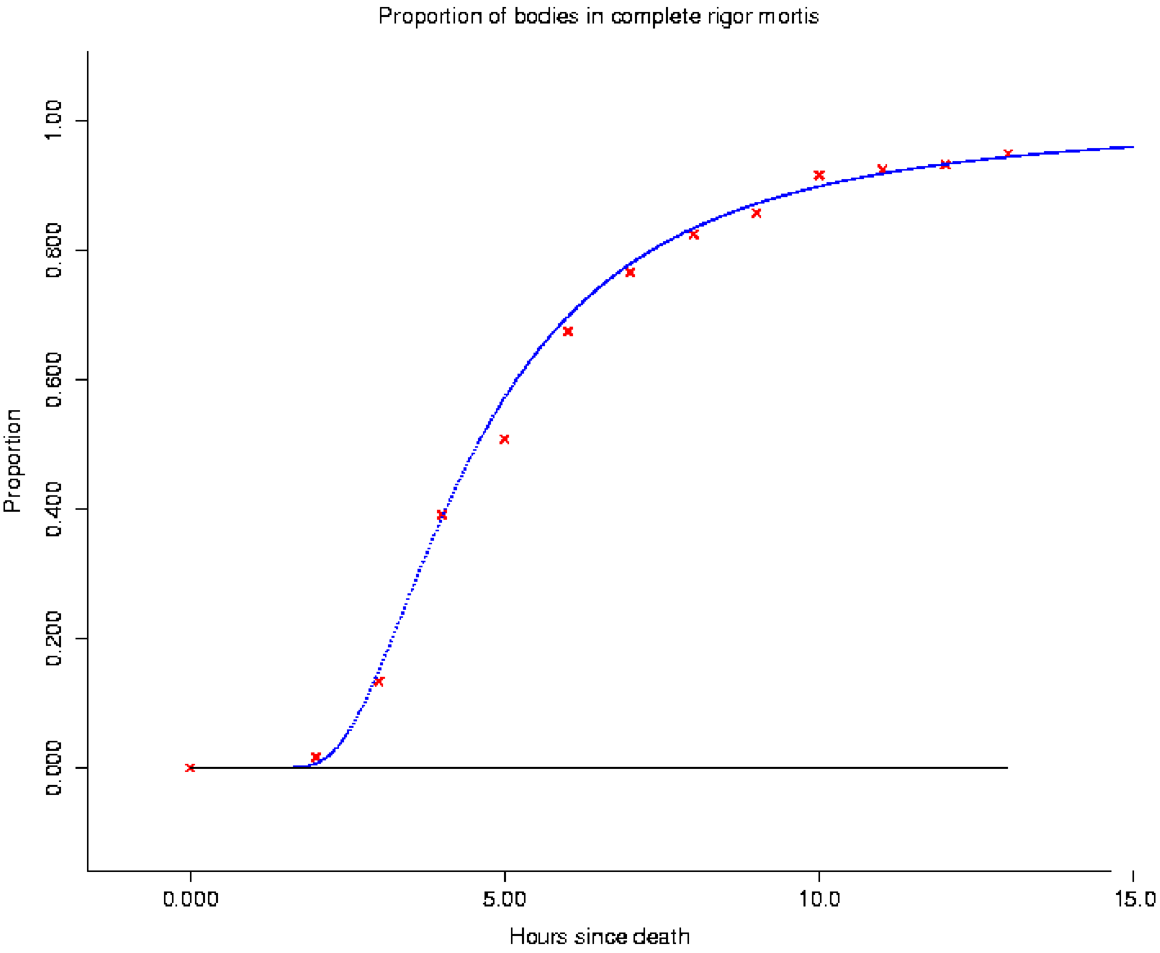

Figure 1: The linear regression results.

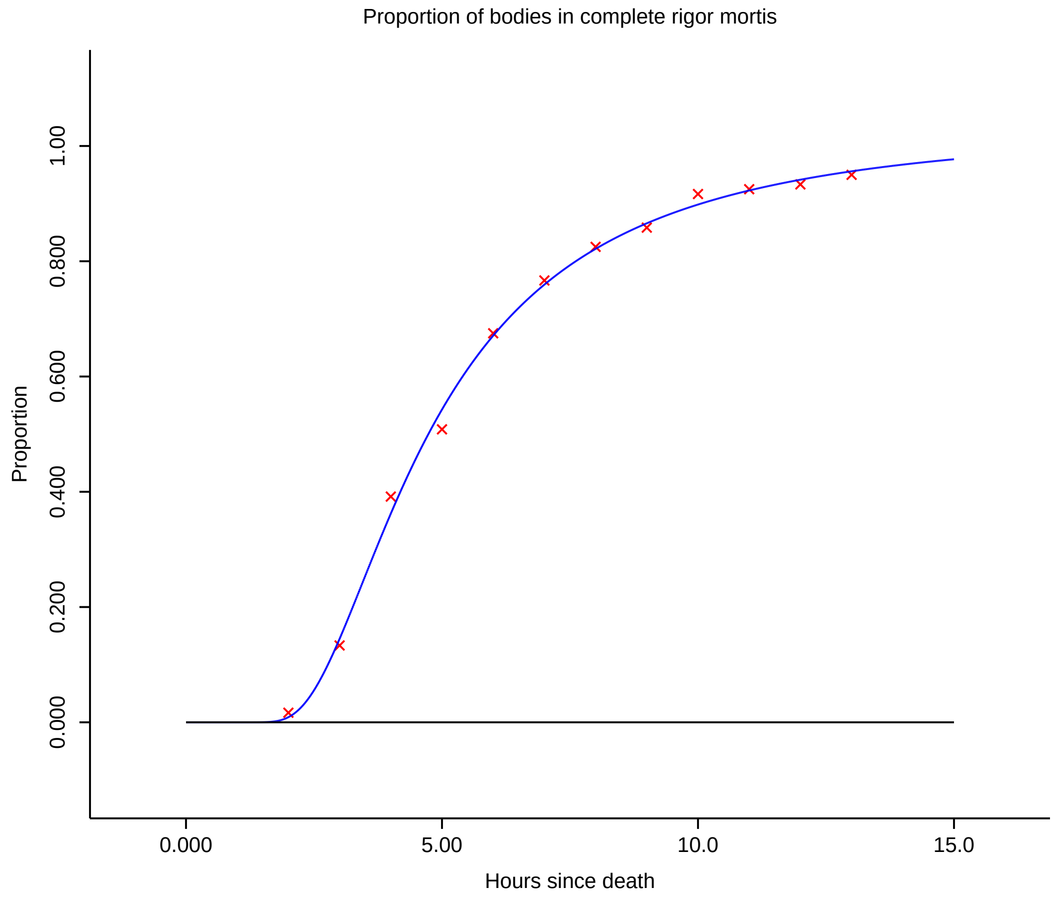

Figure 2: The non-linear regression results appear to be a slight improvement.

There are log transformations that **nearly** permit our problem to be resolved using linear regression. Let's proceed as follows:

$$\ln(p(t)) = -\alpha/t^{\beta}$$

and then

$$\ln(-(\ln(p(t)))) = \ln(\alpha) - \beta \ln(t)$$

(where we take the negative to the other side because the proportion $p(t)$ itself is between 0 and 1, which means that the first log gives a negative – and the second log would not be defined otherwise for negatives).

We can use our data and linear regression to estimate the parameters, if we define "new data” from the old via

$$\begin{aligned} y &= \ln(-(\ln(p(t)))) \\ b &= \ln(\alpha) \\ m &= -\beta \\ x &= \ln(t) \end{aligned}$$

So we log-transform the hours, and do two log-transforms of the cumulative proportion data[1], after which linear regression is performed to get the estimated parameters. This process is encoded in xlispstat as follows:

```
(setq
  hours ’(2 3 4 5 6 7 8 9 10 11 12 13)
  bodies ’(2 14 31 14 20 11 7 4 7 1 1 2)
  ;; there are a total of 114 bodies
  cumulatives ’(2 16 47 61 81 92 99 103 110 111 112 114)
  proportions (/ cumulatives 120)
  ;; new we perform linear regression (with intercept):
  reg (regress (list (ln hours)) (ln (- (ln proportions))))
  ;; and ask reg for its coeficients:
  coefficients (send reg :coef-estimates)
  )
```

which resulted in the coefficients (and hence the model)

$$p(t) = e^{\left(-26.28/t^{2.39}\right)}$$

Now we’d like to compare our result with that obtained using non-linear regression. One of the advantages of non-linear regression is that the *ad hoc* choice of 120 as the number of bodies to divide by is eliminated by estimation of a multiplicative constant in front of the model:

$$c(t) = \gamma e^{\left(-\alpha/t^{\beta}\right)}$$

where c is the cumulative number of bodies in *rigor mortis* at time t, and γ is a multiplicative constant (which we’ll guess is about 120, to start). The other parameters are estimated by their values from linear regression.

Here is the xlispstat code which implements this procedure:

[1] which we obtain by the “trick” of dividing the cumulative number of bodies by 120, which is slightly more than the total number of bodies. If we divided the cumulative bodies by the total number of bodies (114), then our last cumulative value would give rise to a proportion of 1, which will lead to a zero after the first log transformation, and hence the second log would not be defined.

```
;; initial values (solutions of linearization)
(setq
 gamma 120
 alpha (exp (first coefficients))
 beta (- (second coefficients))
 nreg (nreg-model
      (lambda(theta)
        (let (
              (gamma (elt theta 0))
              (alpha (elt theta 1))
              (beta (elt theta 2))
              )
          (* gamma (exp (- (/ alpha (^ hours beta)))))
          )
       )
      (cumsum bodies)
      (list gamma alpha beta)
      )
 )

Residual sum of squares: 47.4054
Coefficients: (124.382 19.424650502192684 2.127669245585588)
Residual sum of squares: 40.9828
Coefficients: (124.42338726758184 21.334959045370884 2.1745553351139515)
Residual sum of squares: 40.7887
Coefficients: (124.3834661473006 21.518819388090982 2.177378350279522)
Residual sum of squares: 40.7887
Coefficients: (124.3819389982478 21.522880354505403 2.1774842047896654)
Residual sum of squares: 40.7887
Coefficients: (124.38193962238628 21.522890934820094 2.1774844349107663)

Least Squares Estimates:

Parameter 0                124.382      (2.87169)
Parameter 1                21.5229      (3.99613)
Parameter 2                2.17748      (0.140507)

Sigma hat:                 2.12887
Number of cases:                12
Degrees of freedom:              9
```

3.1 Non-linear regression as multivariate Newton’s Method in Lisp

Now we’ll implement the scheme described above to carry out non-linear regression as a succession of linear regressions in this case study. We begin by defining the model form (called f below). Following that, we compute the partial derivatives (and name them). In xlispstat I can do that as follows:

```
(defun f(x gamma alpha beta)
  (* gamma (exp (- (/ alpha (^ x beta)))))
  )
(derfunc fpg f gamma)
(derfunc fpa f alpha)
(derfunc fpb f beta)
```

Now we’ll use the model form, and the data, to compute estimates for the responses (the cumulatives minus the estimates). We use the three vectors of the partials evaluated at the data (hours) and at the current best parameter values (the vector containing $\gamma\alpha\beta$).

```
;; find parameters for cumulatives versus hours:
(let* (
       (n (length hours))
       (gammas (repeat gamma n))
       (alphas (repeat alpha n))
       (betas (repeat beta n))
       (estimates (mapcar #'f hours gammas alphas betas))
       reg new-coefs coefs
       )
  (setq reg (regress (list
                      ;; these are vectors of partial derivative values
                      (mapcar #'fpg hours gammas alphas betas)
                      (mapcar #'fpa hours gammas alphas betas)
                      (mapcar #'fpb hours gammas alphas betas)
                      )
                     (-
                      (cumsum bodies)
                      estimates
                      )
                     :intercept nil
                     )
        new-coefs (send reg :coef-estimates)
        coefs (+ (list gamma alpha beta) new-coefs)
        gamma (first coefs)
        alpha (second coefs)
        beta (third coefs)
        )
  (list gamma alpha beta)
  )
```

The first six iterates are given in the appendix. They agree with the results obtained using the non-linear regression program used above.

If we redo the linear regression analysis using the value of 124.38 rather than 120 (which was *ad hoc*), we see that the linear regression results improve a little (a smidgen, but then they didn't have much room for improvement – they're already excellent):

```
Linear Regression:        Estimate            SE              Prob

Constant                 2.96654       (8.280051E-2)       0.00000
Variable 0               -2.12767      (4.222013E-2)       0.00000

R Squared:               0.996078
Sigma hat:               8.198243E-2
Number of cases:                12
Degrees of freedom:             10
```

versus

```
Linear Regression:        Estimate            SE              Prob

Constant                 3.26870       (0.129672)          0.00000
Variable 0               -2.39415      (6.612010E-2)       0.00000

R Squared:               0.992431
Sigma hat:               0.128391
Number of cases:                12
Degrees of freedom:             10
```

4 Appendix

Non-linear regression results, obtained by iterating an approximating linear system. Observe how the estimates are tending towards zero, and how the SEs are converging to the values given by the non-linear regression program.

```
Linear Regression:        Estimate            SE              Prob

Variable 0               4.138723E-2    (2.91773)           0.98899
Variable 1               1.91031        (3.50720)           0.59921
Variable 2               4.688609E-2    (0.138536)          0.74279

R Squared:               0.119552
Sigma hat:               2.12964
Number of cases:               12
Degrees of freedom:             9

(124.42338723437011 21.334959074739725 2.174555336396219)

Linear Regression:        Estimate            SE              Prob

Variable 0               -3.992119E-2   (2.86921)           0.98920
Variable 1               0.183860       (3.94951)           0.96389
Variable 2               2.823019E-3    (0.140285)          0.98438

R Squared:               3.655313E-3
Sigma hat:               2.12887
Number of cases:               12
Degrees of freedom:             9

(124.38346604819884 21.51881954326433 2.1773783557662942)

Linear Regression:        Estimate            SE              Prob

Variable 0               -1.527102E-3   (2.87184)           0.99959
Variable 1               4.060895E-3    (3.99513)           0.99921
Variable 2               1.058519E-4    (0.140502)          0.99942

R Squared:               0.000000
Sigma hat:               2.12887
Number of cases:               12
Degrees of freedom:             9

(124.38193894645649 21.522880438343165 2.1774842077153984)

Linear Regression:        Estimate            SE              Prob

Variable 0               7.135111E-7    (2.87169)           1.00000
Variable 1               1.043788E-5    (3.99612)           1.00000
Variable 2               2.251319E-7    (0.140507)          1.00000

R Squared:               0.000000
Sigma hat:               2.12887
Number of cases:               12
Degrees of freedom:             9

(124.38193965996763 21.522890876219947 2.177484432847311)

Linear Regression:        Estimate            SE              Prob

Variable 0               -2.201135E-8   (2.87169)           1.00000
Variable 1               6.411643E-8    (3.99613)           1.00000
Variable 2               1.922862E-9    (0.140507)          1.00000
```

```
R Squared:               0.000000
Sigma hat:               2.12887
Number of cases:               12
Degrees of freedom:             9

(124.38193963795628 21.522890940336374 2.177484434770173)

Linear Regression:        Estimate            SE              Prob

Variable 0               1.405989E-10   (2.87169)           1.00000
Variable 1               -5.322782E-11  (3.99613)           1.00000
Variable 2               -3.853513E-12  (0.140507)          1.00000

R Squared:               0.000000
Sigma hat:               2.12887
Number of cases:               12
Degrees of freedom:             9

(124.38193963809688 21.522890940283148 2.1774844347663196)
```