Week 3 Class Problems-Answers

John Vinson

8/29/2017

1. For many species of fish, the weight W (in g) is a function of the length L (in mm) that can be expressed by

$$W = W(L) = kL^3 \tag{1}$$

where k is a constant. For a particular species of fish $k = 0.02$ and the length of this species is a function of the number of years that the fish has been alive, t, and is given by

$$L = L(t) = 50 - \frac{(t-20)^2}{10} \tag{2}$$

with no fish living longer than 20 years.

- Determine how the weight of the fish changes with their age (time that they've been alive).

This question is asking to express W as a function of t. So we must plug L from equation 2 into the W expression from equation 1.

$$W(t) = k(50 - \frac{(t-20)^2}{10})^3 \tag{3}$$

- What are all the possible values of k?

For an orgnanism to exist it must have mass which would mean that both W (weight) and L (length) must be positive and nonzero. Since W and L both have to be greater than 0 then $k > 0$.

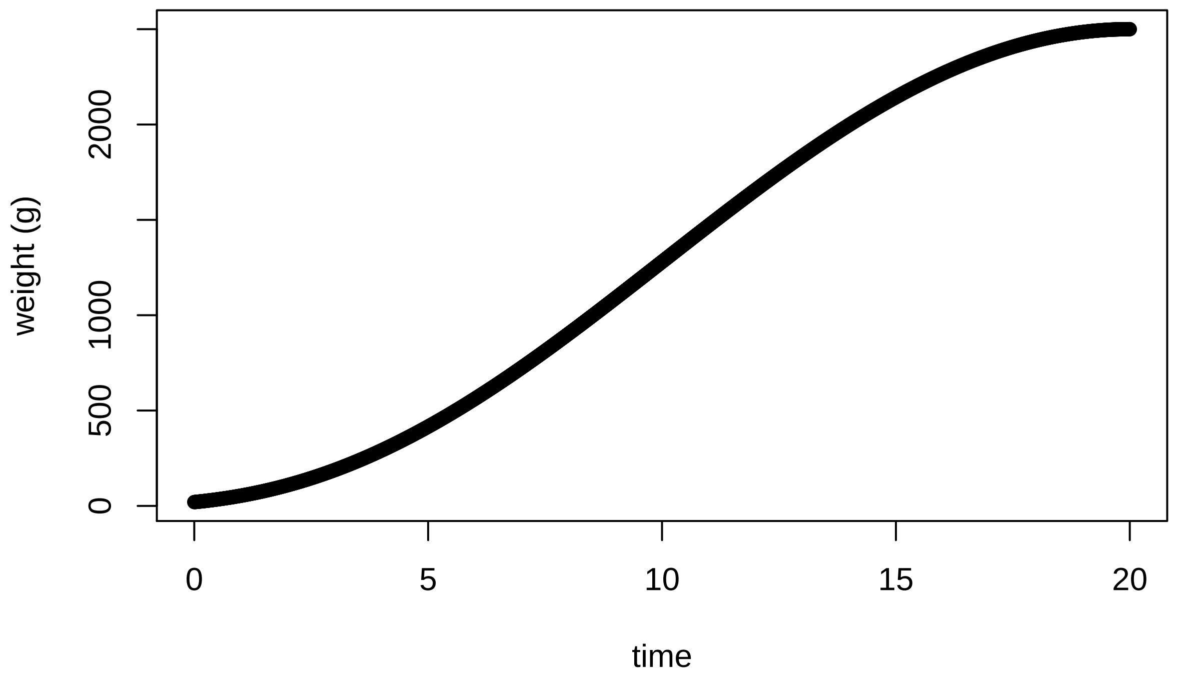
- Plot how the weight of the fish changes through their lifetime.

```r
#Since we already solved for W(t) in part one we can write this as a function
W = function(t){
  return(0.02*(50 - ((t-20)^2)/10)^3)
}

t.vec = seq(0, 20, 0.01)
#create a vector of time points from 0 to 20 going by 0.01.
#Our vector will look like {0, 0.01, 0.02, 0.03, ..., 20.0}

W.vec = W(t.vec)
#create a new vector where the time vector is plugged into the W function.

plot(t.vec, W(t.vec), xlab="time", ylab="weight (g)", pch=16)
```



```
#plot the two vectors, change the x and y axis labels/titles and change how the points appear
```

- Mathematically find the age of the fish for which the weight will exceed 450 g.

We can solve the equation from part 1 for t.

$$W = k(50 - \frac{(t-20)^2}{10})^3 \tag{4}$$
$$\frac{W}{k} = (50 - \frac{(t-20)^2}{10})^3 \tag{5}$$
$$(\frac{W}{k})^{\frac{1}{3}} = 50 - \frac{(t-20)^2}{10} \tag{6}$$
$$\frac{(t-20)^2}{10} = 50 - (\frac{W}{k})^{\frac{1}{3}} \tag{7}$$
$$(t-20)^2 = 500 - 10(\frac{W}{k})^{\frac{1}{3}} \tag{8}$$
$$t - 20 = (500 - 10(\frac{W}{k})^{\frac{1}{3}})^{\frac{1}{2}} \tag{9}$$
$$t = (500 - 10(\frac{W}{k})^{\frac{1}{3}})^{\frac{1}{2}} + 20 \tag{10}$$

Now we just need to plug in our values for W (450) and k (0.02).

$$t = (500 - 10(\frac{450}{0.02})^{\frac{1}{3}})^{\frac{1}{2}} + 20 \tag{11}$$

This provides two solutions:

$$t_1 = (500 - 10(\frac{450}{0.02})^{\frac{1}{3}})^{\frac{1}{2}} + 20 \approx 34.75429 \tag{12}$$
$$t_2 = -(500 - 10(\frac{450}{0.02})^{\frac{1}{3}})^{\frac{1}{2}} + 20 \approx 5.245706 \tag{13}$$

*However, since $t \in [0, 20]$ the solution we are looking for is $t_2 \approx 5.25$.

- Numerically find the age of the fish for which the weight will exceed 450 g.

```
t.450.vec = which(W.vec>=450)
#returns a vector of the index of all the values of W.vec that are greater than or equal to 450

t.450.min = min(t.450.vec)
#this provides the index of the first t.vec value that that is greater than or equal to 450

t.450.out = t.vec[t.450.min]
#returns the t.vec value that is the first time that is greater than or equal to 450

t.450.out
```

```
## [1] 5.25
```

2. For fiddler crabs, data gathered by Thompson show that the relationship between the weight C of the claw and the weight of the body W is given by

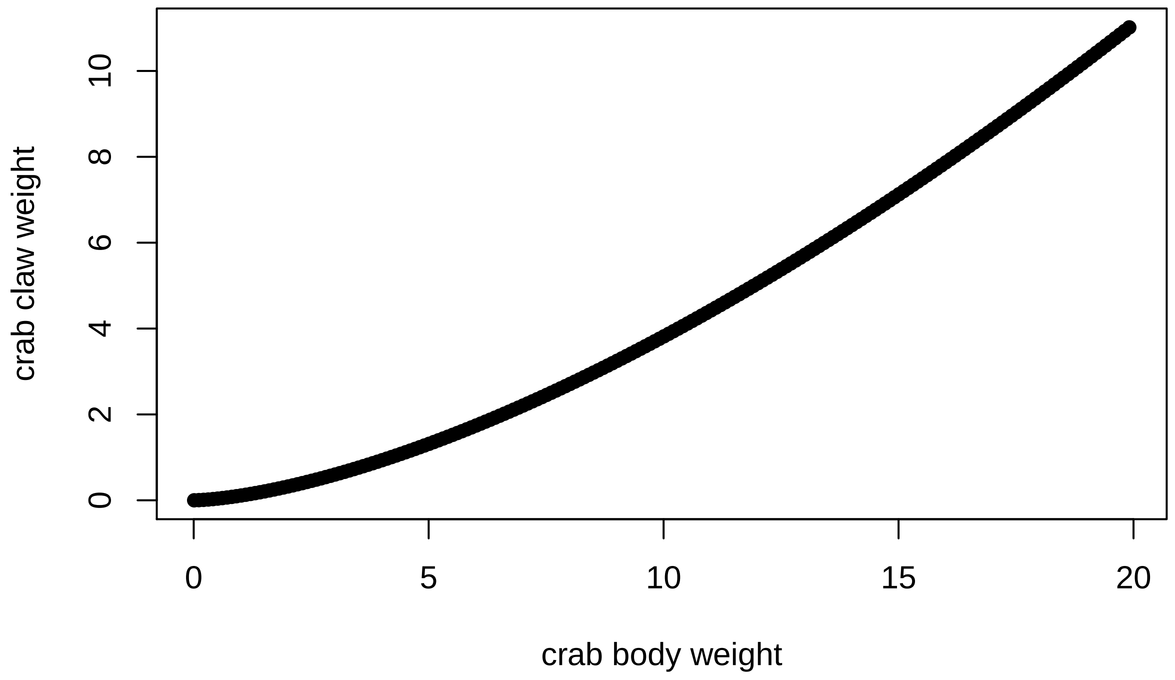
$$C = 0.11W^{1.54} \tag{14}$$

where W and C are in grams.

- What values can C and W take?

Since weight must be a postive, nonzero value both C and W must be greater than 0.

- Plot C for $W \in (0, 20]$.



- Write the function where the weight of the crab body is a function of claw weight.

We simply need to solve for W.

$$\begin{aligned} C &= 0.11W^{1.54} && (15) \\ 0.11W^{1.54} &= C && (16) \\ W^{1.54} &= \frac{C}{0.11} && (17) \\ W &= (\frac{C}{0.11})^{\frac{1}{1.54}} && (18) \end{aligned}$$

- For what values of the claw weight does the crab body weight exceed 6 g?

There are two ways we can solve this. The most easiest is to plug in $W = 6$ into the C equation given.

$$C = 0.11W^{1.54} = 0.11 * 6^{1.54} \approx 1.74g \quad (19)$$

The second way is to numerically find this.

```
W.6.vec = which(W.vec>6)
#determine the index of the weight values that are greater than 6

W.6.min = min(W.6.vec)
#pull out the minimum index
#(which is the first that has a weight greater than 6)

C.6 = C.vec[W.6.min]
#pull out the claw weight that corresponds to that weight

C.6
```

```
## [1] 1.741243
```

3. The following table give the number of AIDS-related deaths in the New York Prison System.

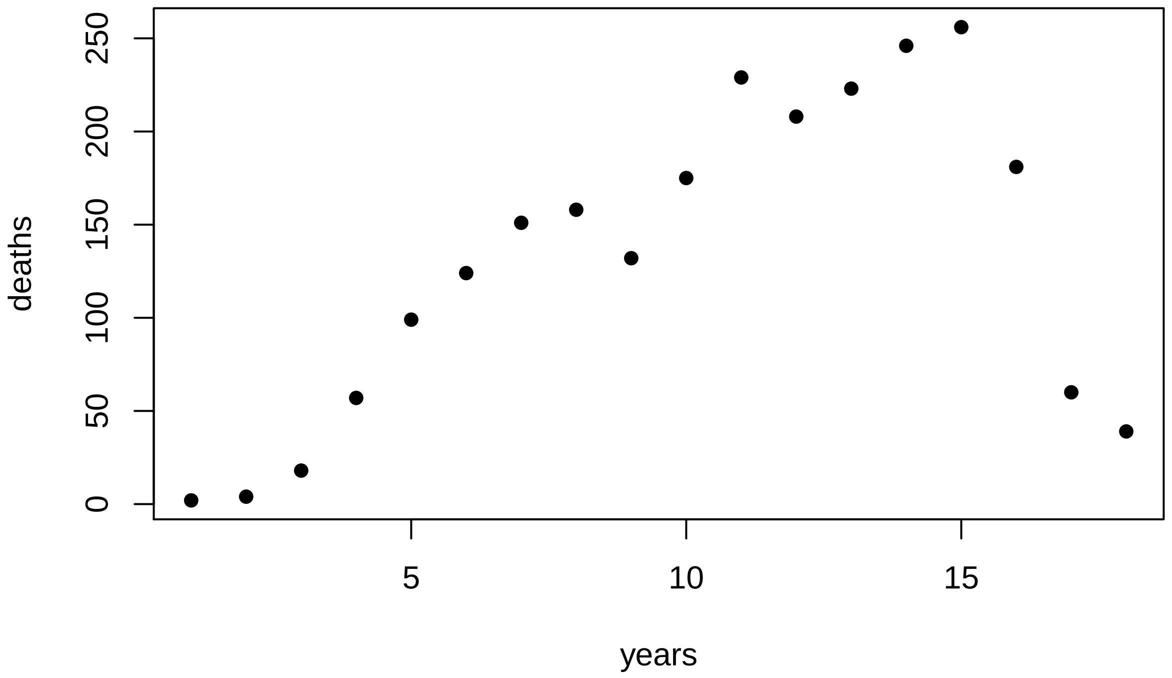
year	deaths	year	deaths
1981	2	1990	175
1982	4	1991	229
1983	18	1992	208
1984	57	1993	223
1985	99	1994	246
1986	124	1995	256
1987	151	1996	181
1988	158	1997	60
1989	132	1998	39

- Create a scatter plot for the data in the table. Let the x-axis represent the number of years since 1980.

```
years = seq(1981,1998)-1980
#our vector of years since 1980
deaths = c(2,4,18,57,99,124,151,158, 132,175,229,208,223,246,256,181,60,39)
#the deaths

aids.data = data.frame(years, deaths)
#create a dataframe of the data
```

```r
plot(deaths~years, data=aids.data, pch=16)#plot the data.
```



- Determine an appropriate function that can be used to model the data. Why did you choose this function? Estimate the parameter values. **We can rule out linear functions because it would not capture the behavior in the right tail (after 15 years) of the data.**

A quadratic function would be better than linear however the the behavior in the both tails would be lost.

A cubic function would be best as it would capture these behaviors.

$$death = a * years^3 + b * years^2 + c * years + d \tag{20}$$

Now that we've determined an appropiate function, we need to find the values of the coefficents (parameterize the the model). To do this we will minimize the sum of squared errors. We can define the sum of squared error as

$$SSE = \sum_{i}^{N}(D_i - E_i)^2 \tag{21}$$

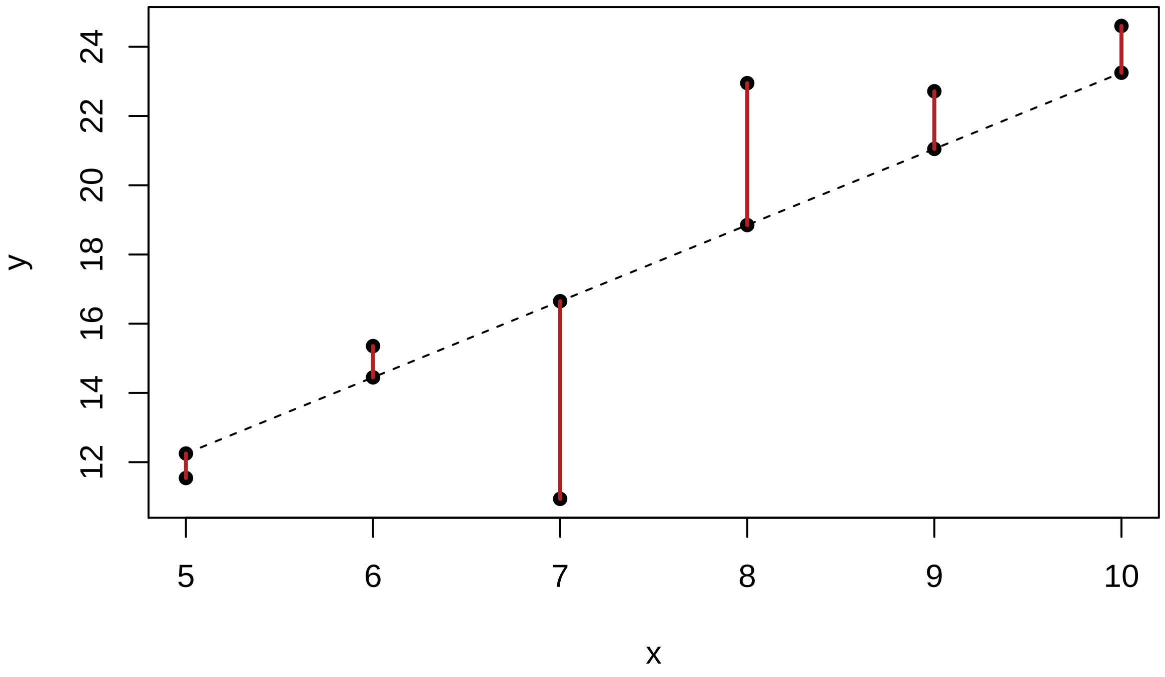
In this equation:

- D_i is the number of deaths from the data in year i.
- E_i is the number of estimated deaths from the model in year i. In our case, this will be the number of cases at year i from our cubic model.

- N is the total number of time points in our dataset. For our data, this is 18.

```r
sse = function(parms, data){
  out = sum((cub.fun(parms, data[,1])-data[,2])^2)
  return(out)
}
#the sse function that tells us how well our model fits our data
```

Essentially what we are doing is for each data point we are looking at the distance between the value predicted by the model with a specific parameter set and the actual number (red line segments in the figure below). In the SSE equation, we are squaring the difference to eliminate the negative values (which would occur for x values 6, 8, 9 and 10).



What we want to do is minimize that distance between the actual data and our predicted values from the model (make those red lines as short as possible). We could do a grid search where we iterate over all the possible values of $\{a, b, c, d\}$ and finding the parameter set that has the lowest SSE. Keep in mind that each time we plug in a new set of $\{a, b, c, d\}$ we will get a new dashed line and a corresponding SSE.

But what we will do is use a function in R that is a more directed. The function *optim* will allow us to find the best fit values of $\{a, b, c, d\}$. What this function does is find the minimum value of takes in four arguments. First, we need to give it our best guess as to what $\{a, b, c, d\}$ are. For our model, we can start by "guessing" that all the values are 0. Next the function that we want optimized. Here this is the SSE function. Next, we need to tell it which dataset to use since that is the second argument of the SSE function. Last, the optimization method. Here we are using Nelder-Mead however there are other methods that could be used. See the help file for all of them.

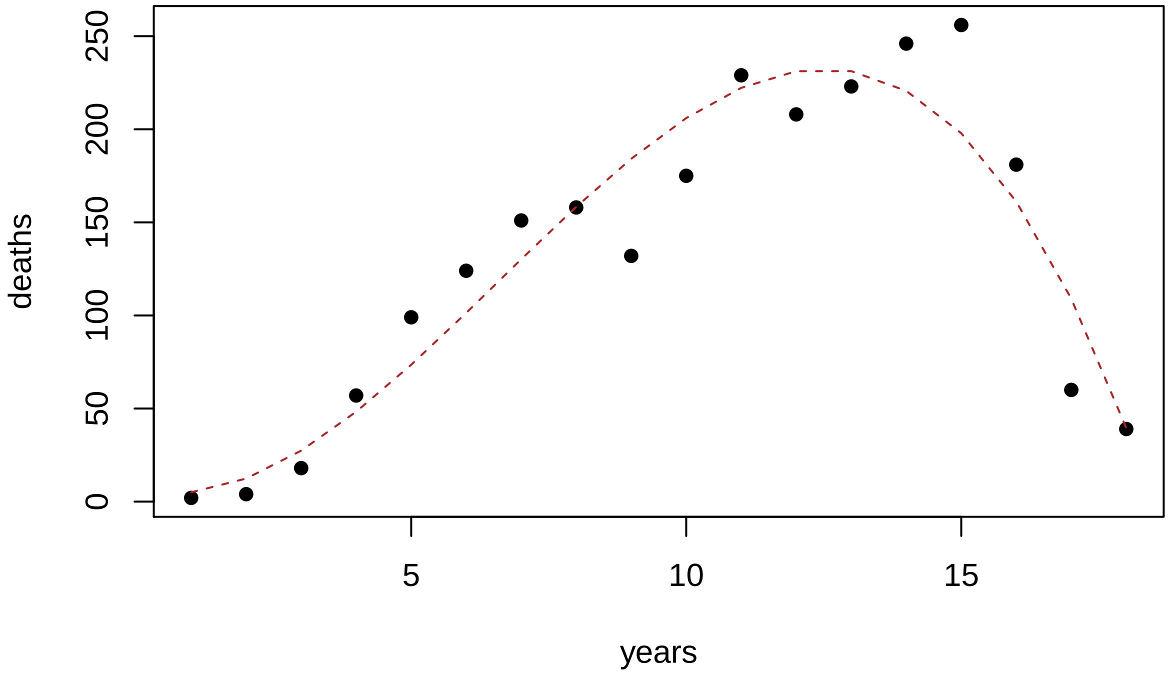
```r
out = optim(c(0,0,0,0), fn=sse, data=aids.data, method="Nelder-Mead")
#use the optimizer to find the best fit values
#i.e. the values that have the lowest SSE.
```

Now that we've found the best fit parameters (from *optim*) we need to pull them out of the estimated parameter values. We need to pull out the *par* element. Then we can plug that resulting vector into the cubic function and plot our results.

```r
fit.vals = out$par
#pull out the estimated values

best.mod = cub.fun(fit.vals, aids.data$years)
#plug in these values into our cubic function
```

```r
plot(deaths~years, data=aids.data, pch=16)
lines(aids.data$years, best.mod, col='firebrick', lty="dashed")#plot our results
```



The estimated values of $\{a, b, c, d\}$ are

```
## [1] -0.2753559  5.4474581 -7.0232443  6.8588047
```

- Using this model(function), determine what year the deaths was a maximum.

```r
death.max = max(best.mod)
#determine what the maximum value of deahts

death.max.index = which(best.mod==death.max)
#find the index of this value

death.max.year = years[death.max.index]
#pull out the corresponding year
```

The year that has the most deaths is

```
## [1] 13
```

with maximum deaths of

```
## [1] 231.2202
```