The data file `tuna.R` is a table of data for southern bluefin tuna found at `http://www.ccsbt.org/docs/data.html`. It can be read in as follows (cross fingers here)

```
> f = "http://www.math.csi.cuny.edu/verzani/classes/MTH804/computer/tuna.R"
> tuna = read.table(file=url(f),header = T)
```

The variable `tuna` contains estimates for the global catch of southern bluefin tuna. We want to model the total estimate against time.

A deterministic model used by Myers and Worm (Nature 423, 280-283 (2003)) to model decline of predatory fish of species i is

$$N_i(t) = N_i(0)((1-\delta_i)e^{-r_i t} + \delta_i).$$

For a single species the subscripts i can be dropped.

Suppose the process has observation uncertainty of the type $N_{obs,t} = N_t + W_t$ where W_t is assumed to be normal with mean 0 and unknown variance.

1. Plot the data. Do all the data points appear to be valid? If not, argue that they can be dropped.
2. Write down the negative log likelihood function for the model. Assume, you know N_0 from the graph.
3. Use the method of non-linear least squares to estimate the parameters r and d.
4. Look at the negative log likelihood as a function of σ. Can you minimize this function as a function of δ, r? What equation do these satisfy at $\hat{\sigma}^2$?
5. Can you compare the submodel $N(0)\delta = 20000$ to the model δ is in $[0,1]$ using the likelihood ratio test? What about the AIC criteria.

1 Analysis

First, we can plot the data with this command

```
> plot(Total ~ year, data = tuna)
```

Looking at the plot (not shown) we extract just the years after 1980 with the `subset` command:

```
> tuna = subset(tuna, subset = year > 1980, select=c("year","Total"))
```

The `subset()` command has the argument `subset =` which allows a logical condition to select the rows. The `select=` argument allows you to choose which columns. So the above assigns to `tuna` just the rows when `year` is 1981 or more, and just the two columns `year` and `Total`.

A new plot is done with the same command as before, only now as `tuna` is reduced the plot will be different:

```
  plot(Total ~ year, data = tuna)
```

Now, let's try to fit this data. Previous attempts at fitting the range from 1960 forward failed as the suggested curve wants to have $d < 0$ and r close to 0 which essentially means the best fit is a straight line. Here we do better.

As is typical, we set up some functions. Notice, we subtract 1981 from the year so that time 0 corresponds to the first data point.

```
> f = function(t,N0,r,d) N0*((1-d)*exp(- r*(t-1981)) + d)
```

Let's try to predict good starting values. First, `N0` should be the initial amount so we guess that $N_0 = 45000$ should be a good value. Next $N_0 \cdot d$ is the leveling off amount. As this is about 15,000 we guess that $d = 15,000/45,000 = 0.333$ so we start with $d = 1/3$ for roundness. Next, `r` is the initial descent. We can just guess a value for r by plotting. Try $r = 1$ and see it works:

```
> curve(f(x,N0=45000,r=1,d=1/3), add=T)
```

The plot declines too quickly, slowing up `r` by setting `r=.1` is a little better as this shows

```
> curve(f(x,N0=45000,r=.1,d=1/3),add=T)
```

We now try to fit the data with `nls()`:

```
> res=nls(Total ~ f(year,N0,r,d),data=tuna,start=list(N0=45000,r=.1,d=1/3))
> res
Nonlinear regression model
  model:  Total ~ f(year, N0, r, d)
   data:  tuna
       N0         r         d
4.981e+04 1.968e-01 2.669e-01
 residual sum-of-squares:  213219573
> curve(f(x,N0=49810,r=.1968,d=.2669),add=T,col="red")
> AIC(res)
[1] 404.4
```

It converges now and the curve shows an okay fit.

2 Why did we fit with least squares?

Good question. We have a model that says

$$N(t) = N(0)\left((1-\delta)e^{-rt} + \delta\right)$$

(as a deterministic model) with *additive* process uncertainty

$$N_{obs}(t) = N(t) + W_t$$

where we assume W_t are iid normals with mean 0 and variance σ^2.

As we observe the data, we have a model

$$N_{obs}(t) = N(0)\left((1-\delta)e^{-rt} + \delta\right) + W_t$$

which has negative log likelihood function

$$-\log L(\Theta) = -\log L(N_0, r, \delta, \sigma) = \log\sigma + \frac{1}{2\sigma^2}\sum (N_{obs}(t) - f(t|\Theta)^2$$

As in class, if we look for maximum likelihood estimators, then we first differentiate in σ to see that at the optimal σ, $\hat{\sigma}$ we have

$$\hat{\sigma}^2 = \frac{1}{n}\sum (N_{obs}(t) - f(t|\Theta)^2$$

Putting this back into the equation, gives

$$-\log L(N_0, r, \delta, \hat{\sigma}) = \frac{1}{2}\log(\frac{1}{n}\sum (N_{obs}(t) - f(t|\Theta)^2) + \frac{n}{2}$$

Minimizing this is the same as minimizing the non-linear least squares problem.

That is, the `nls()` procedure above is finding maximum likelihood estimators for the model.

3 Model selection using AIC, likelihood ratios

From the model we have the following numbers

```
> AIC(res)
[1] 404.4
> 1/2* log(sum(resid(res)^2)) + length(year)/2
[1] 20.09
```

The latter being the negative log-likelihood.

To fit the model using $d = 1/3$ as known, and not a parameter can be done. This is an example to compare nested models with likelihood.

We need a new function. Note the body of the function is identical to `f` and we just cut and paste:

```
> f
function(t,N0,r,d) N0*((1-d)*exp(- r*(t-1981)) + d)
> f.1 = function(t,N0,r) N0*((1-d)*exp(- r*(t-1981)) + d)
```

This function will find the value of `d` in the environment. We set it explicitly with

```
> d = 1/3
```

Now repeat the fitting above

```
> res = nls(Total ~ f.1(year,N0,r),data=tuna,start=list(N0=45000,r=.1))
> res
Nonlinear regression model
  model:  Total ~ f.1(year, N0, r)
   data:  tuna
       N0         r
4.759e+04 2.219e-01
 residual sum-of-squares:  264918454
```

We can compare fits as follows

```
> plot(Total ~ year,tuna)
> curve(f(x,N0=49810,r=.1968,d=.2669),add=T,col="red")
> curve(f.1(x,N0=47590,r=.2219),add=T,col="blue")
```

The two fits are different as expected. The red line fits "better" as measured by sum of squares, but it has an extra parameter.

We have two ways to compare models here. The AIC comparison gives

```
> AIC(res)
[1] 407.0
```

which we compare to the 404.4 from the red line. The lower one wins, so this would select the model where δ is a parameter.

The log likelihood computes the statistic

$$2(-\log(L(\Theta|M_A)) + -\log(L(\Theta|M_B))$$

The negative log likelihood for the second fit is found as above with

```
> 1/2* log(sum(resid(res)^2)) + length(year)/2
[1] 20.20
```

So the statistic is

$$2(20.20 - 20.09) = 0.22$$

There is 1 extra parameter. Under the null hypothesis assumption that the extra parameter is not needed, we have the p-value of

$$P(\chi^2_1 > 0.22)$$

which is computed with

```
> 1 - pchisq(0.22,df=1)
[1] 0.639
```

Indicating that the extra parameter is not needed by the likelihood ratio test.

4 Multiplicative uncertainty

How would this analysis change if the observed error was modeled in a multiplicative manner:

$$N_{obs}(t) = N_t W_t,$$

where W_t is log-normal:

$$W_t = \exp(Z_t\sigma - \sigma^2/2)$$

Now the model would be

$$N_{obs}(t) = W_t \cdot N(0)((1-\delta)e^{-rt} + \delta).$$

We would take logarithms first to get

$$-\log L(\Theta) = -\log L(N_0, r, t, \sigma) = n\log\sigma + \frac{1}{2\sigma^2}\sum(\log N_{obs}(t) - f(t|N_0, r, \delta) + \frac{\sigma^2}{2})^2$$

Notice, we can't use calculus as before as their is a σ^2 inside the sum so the result is not nearly as simple.

We can do a direct optimization though and hope it converges. For this, we use the `optim()` function. This expects the function to be a function of a single vector containing the parameters. For instance we define `g()` using the previous definition for `f`

```
> f
function(t,N0,r,d) N0*((1-d)*exp(- r*(t-1981)) + d)
> g = function(x) {
+ n=length(year)
+ n*log(x[4]) + 1/(2*x[4]^2)*sum((log(Total) -
+ log(f(year,x[1],x[2],x[3])) + x[4]^2/2)^2)
+ }
```

(The vector `x` contains the parameters in this order N_0, r, δ, σ. For example, `x[4]`$= \sigma$.)

We need good guesses for the initial quantities. We use the ones from before which gave

$$N0 = 49810, r = .1968, d = .2669$$

As for σ we note that the variation is on the order of a few percent, so we guess σ should be fairly small, say around 1.

Now we can try `optim()` it is called with the initial guesses, then the function name

```
> attach(tuna)                  # needed as no data= argument
> optim(c(N0=49810,r=.1968,d=.2669,sigma=1), g)
$par
       N0         r         d     sigma
7.094e+04 2.152e+05 2.929e-01 3.859e-01
... note the warnings ...
```

The results are returned after the `par`. Notice a warning which can be viewed with `warnings()`. Something about `NaN`'s indicating that something may have given a singularity.

We should check that the values are reasonable. Plotting them gives us

```
>  curve(f(x,N0=70940,r=21520,d=.2929),add=T,col="green")
```

That `r` value is a big problem!

Maybe our value of σ is too large. Let's try $\sigma = .1$ and see

```
> optim(c(N0=49810,r=.1968,d=.2669,sigma=.1), g)$par
       N0         r         d     sigma
5.560e+04 2.707e-01 2.640e-01 1.000e-01
There were 34 warnings (use warnings() to see them)
```

This gives us the same warning. Looking at the help page for `optim` we see there are other algorithms. The one labeled `"L-BFGS-B"` allows us to put constraints on the values. Trying this first without constraints, gives us convergence though:

```
> optim(c(N0=49810,r=.1968,d=.2669,sigma=.1), g,method="L-BFGS-B")$par
       N0         r         d     sigma
4.981e+04 2.400e-01 2.926e-01 1.560e-01
```

Now we can plot to view

```
> curve(f(x,N0=49810,r=.2400,d=.2926),add=T,col="green")
```

This gives a reasonable fit.

5 Confidence interval for δ.

The value of δ inidicates how much the stocks are down. In this case,as δ is around .3, we can say that stocks are down 70% from 1980 levels.

As this is a statistic, it would be nice to understand its sampling variation, or to give a 95% confidence interval for it.

Oneway to approach that problem is through the likelihood function. This is more math intensive. An alternate is the bootstrap method. This is more computer intensive. We illustrate this one next.

The basic non-parametric bootstrap method uses the following scheme to find out about the sampling distribution of $\hat{\theta}$:

- Sample from the data with replacement to form a new sample
- Find the parameter estimates for the new sample. Call this $\hat{\theta}^*$.
- Repeat many times to find the sampling distribution of $\hat{\theta}^*$.
- Under many assumptions, the quantiles of the above simulation of $\hat{\theta}^*$ form confidence intervals for $\hat{\theta}$.

The mechanics involve the following.

- Sampling with replacement. This is done with the `sample()` command:

  ```
  > n = length(tuna$year)
  > new.indices = sample(1:n,n, replace=T)
  > df = tuna[new.indices,]
  ```

  The new indices are a sample with replacement from the old indices. As replacement is used, there may be repeats. These indices are then used to extract the data, storing it above in a variable `df`.

- Automating the model fitting. We see that sometimes the model fitting dies on us. We can catch these times with the `try()` function so that our repeating part doesn't stop. The format to use is

  ```
  x = try(something.that.may.fail)
  if(inherits(x,"try-error"))     what.to.do.if.fail   else     what.to.do.if.success
  ```

- Looping to repeat. A `for()` loop will loop over values in a vector. For example `for(i in 1:m)` will loop over the values in the vector $1, 2, \cdots,$`m` setting `i` to the value within the loop.

We apply this to the model with additive observation uncertainty. We could use `nls()` on this model. We use a function to implement the bootstrap:

```
"d.boot" <-
  function(m=100) {
    res = c()
    N0=49810;r=.2400;d=.2926;sigma=.156
    n = length(tuna$year)
    for(i in 1:m) {
      new.indices = sample(1:n,n,replace=T)
      df = tuna[new.indices,]
      res.nls= try(nls(Total ~ f(year,N0,r,d),data = df,
           start=c(N0=N0,r=r,d=d)))
      if(!inherits(res.nls, "try-error")) {
        res[i] = coef(res.nls)[3]
      } else {
        res[i] = NA
      }
    }

    res
  }
```

(You might wish to cut and paste this into a file from the pdf file on the web, then source that file in to define the function.)

When this is run, it outputs a vector with simulations. For example,

```
> res = d.boot(1000)
Error in nls(Total ~ f(year, N0, r, d), data = df, start = c(N0 = N0,  :
        singular gradient
Error in nls(Total ~ f(year, N0, r, d), data = df, start = c(N0 = N0,  :
        number of iterations exceeded maximum of 50
Error in nls(Total ~ f(year, N0, r, d), data = df, start = c(N0 = N0,  :
        number of iterations exceeded maximum of 50
Error in nls(Total ~ f(year, N0, r, d), data = df, start = c(N0 = N0,  :
        number of iterations exceeded maximum of 50
```

The errors are when the `nls()` function failed. We can plot the distribution with a histogram as follows

```
> hist(res)
```

We can find 95% confidence intervals using the `quantile()` function as

```
> quantile(res,c(.025,.975),na.rm=T)
  2.5%  97.5%
0.1588 0.3140
```

Notice, the `na.rm=T` to remove and `NA` values from consideration.

Thus the 95% bootstrap confidence interval on δ is $(.159, .314)$.

6 summary

We are trying to model the abundance of bluefin tuna, using the amount caught as some measure of total abundance. We employ a model with exponential decay to some baseline. This baseline allows us to say that from this analysis the tuna stocks are down 70% since 1980 with a 95% CI of $(69\%, 84\%)$.