This project is based on a thesis by Philippe Grosjean available at

`http://www.sciviews.org/_phgrosjean/`

The 2-second summary is to fit a growth model to a dataset which measures the growth of sea urchins. The actual dataset and some additional functions are loaded into your session by grabbing them from the course website as follows

```
> f = "http://www.math.csi.cuny.edu/verzani/classes/MTH804/computer/urchin.R"
> source(file=url(f))
```

The data is stored in the variable `urchin.freqs`. It records a count of individuals of a specified size at several time intervals.

The first thing to do is to expand the data so that we have pairs of data of the type (y_i, x_i) where y_i is the size and x_i the age. The function `expandFreq` does the dirty work

```
x = expandFreq(ages,urchin.freqs)
```

The vector `ages` is also loaded in when you did source.

Peek at the data to get a sense of what it is

```
> fix(x)
```

(The command `fix()` is an oddly named function which is similar to `edit()`, but does not need its return value assigned to make changes.)

You see a large data frame with two columns `age` and `size`. We can attach the data frame so that the variable names are accessible. We then plot

```
> attach(x)
> plot(size ~ age)              # size is response variable
> plot(jitter(size) ~ jitter(age)) # What does jitter do?
```

1 Growth models

Our initial goal is to try to fit different types of models to this data using the method of least squares. According to the author, the following models are popular when fitting sea urchin data. Let Y_i be the size and x_i the age of the ith observation. We formulate the following statistical models for size of an individual as

$$y_i = f(x_i|\beta_0, \beta_1, \ldots, \beta_j) + W_i$$

where the β_i are the parameters and W_i is randomness in the size. It may make sense to also attribute error to the age measurement (process error) but we don't do that here.

Gompertz model This model states that the rate of change (dY/dt) decreases proportionally to the logarithm of survival. In a statistical model formula it looks like

$$f(t|Y_\infty, a, k) = Y_\infty a^{e^{-kt}}$$

The parameter Y_∞ is the carrying capacity as when $t \to \infty$ we get $f(t) \to Y_\infty$.

von Bertalanffy models This model introduces non-linear terms into the rate of growth, for example

$$dY/dt = aY^{2/3} - bY$$

The solution gives

$$f(t|Y_\infty, k, t_0) = Y_\infty(1 - e^{-k(t-t_0)})^3$$

This is the von Bertalaffy model 3, the von Bertalaffy model 1 is

$$f(t|Y_\infty, k, t_0) = Y_\infty(1 - e^{-k(t-t_0)})$$

Richards model The Richards model is an easy generalization of the von Bertalaffy models. It is

$$f(t|Y_\infty, k, t_0, m) = Y_\infty(1 - e^{-k(t-t_0)})^m$$

If $m = -1$ the Richards model becomes the logistic model, and as $m \to \infty$ the Gompertz model.

Weibull model The Weibull model for population growth is not the same as the Weibull density, but rather the integral of that. In our formulation we have

$$f(t|Y_\infty, Y_0, k, m) = Y_\infty - (Y_\infty - Y_0)e^{-kt^m}$$

Jolicouer curve This is like the logistic model with different time scale

$$f(t|Y_\infty, b, m) = \frac{Y_\infty}{1 + bt^{-m}}.$$

Johnson model Another similar curve, which is like exponential growth, but the t is in a different place

$$f(t|Y_\infty, k, t_0) = Y_\infty e^{\frac{1}{k(t-t_0)}}$$

Preece-Banes 1 model A model with growth spurts governed by two exponential growth phases

$$f(t|Y_\infty, d, k_1, k_2, t_0) = Y_\infty - \frac{2(Y_\infty - d)}{e^{k_1(t-t_0)} + e^{k_2(t-t_0)}}$$

2 Fitting the models using least squares

We can use the `nls()` function to find the least squares estimators for the unknown parameters. Recall, we define the a function in terms of the data and the parameters that give the right hand side of the model. For example, these commands will fit the Weibull model. Recall we need to give an initial guess which was found here by guessing, plotting and perhaps reguessing.

```
> attach(x)
> plot(jitter(size) ~ jitter(age))
> weibull = function(age,Y,Y0,k,m) Y - (Y-Y0) * exp(-k*ageˆm)
> curve(weibull(age,65,0,1,1),add=T) # oops! curve wants x there
Error in curve(weibull(age, 65, 0, 1, 1), add = T) :
        'expr' must be a function or an expression containing 'x'
> curve(weibull(x,65,0,1,1),add=T) # this looks like a good start
> nls(size ~ weibull(age,Y,Y0,k,m), start = list(Y=65,Y0=0,k=1,m=1))
Nonlinear regression model
  model:  size ~ weibull(age, Y, Y0, k, m)
   data:  parent.frame
      Y      Y0       k       m
56.4528  0.4850  0.2264  1.6444
 residual sum-of-squares:  111706
> curve(weibull(x,56.45,0.48,.2264,1.6444),add=T, col="blue")
> res.weibull = nls(size ~ weibull(age,Y,Y0,k,m),
+ start = list(Y=65,Y0=0,k=1,m=1))
```

The guess was used to feed starting value into the `nls()` function, which returned with the least-squares estimates. We then plot the results in blue and store them for later in the variable `res.weibull`.

Next, we do the trickier Preece-Banes model. In this example, we write the function to use a single vector valued parameter. This is perhaps less clear, but makes it easier to manipulate with the computer.

```
> preece = function(age,beta) {
+ top = 2*(beta[1]-beta[2])
+ bottom = exp(beta[3]*(age-beta[5])) + exp(beta[4]*(age-beta[5]))
+ beta[1] - top/bottom
+ }
> plot(jitter(size) ~ jitter(age))
> curve(preece(x,c(Y=65,d=10,k1=.1,k2=.2,t0=0)),add=T)
> nls(size ~ preece(age,beta),
+ start = list(beta=c(Y=65,d=10,k1=.1,k2=.2,t0=0)),
+ trace=T)
511437 :  65.0 10.0  0.1  0.2  0.0
Error in numericDeriv(form[[3]], names(ind), env) :
        Missing value or an Infinity produced when evaluating the
        model
```

The extra argument to `nls()`, `trace=T` turns on tracing of the algorithm which we see wasn't happy with these starting points. Some of these models have "self-starting" implementations which eliminate this hunting for parameters, but not in this case. Thinking about the role of the *k*'s (controlling growth), by making one of them larger we speed up a growth spurt. Trying this gives

```
> nls(size ~ preece(age,beta),
+ start = list(beta=c(Y=65,d=10,k1=.1,k2=.5,t0=0)),
+ trace=T)
455919 :  65.0 10.0  0.1  0.5  0.0
441222 :  6.099e+01 2.690e+01 9.013e-04 6.475e-01 1.206e+00
211513 :  56.8809 40.8957  0.1802  0.9713  2.9797
140728 :  55.1321 34.5503  0.1936  1.2056  2.1669
```

```
130220 :  55.4060 40.2570  0.2592  1.3036  2.6889
113087 :  55.5012 43.2264  0.2922  1.4911  3.0979
111071 :  55.4265 42.1463  0.2832  1.5088  2.9817
111041 :  55.4201 42.3294  0.2863  1.5219  3.0040
111041 :  55.4146 42.3529  0.2868  1.5249  3.0060
111041 :  55.4134 42.3596  0.2870  1.5255  3.0066
111041 :  55.413 42.361  0.287  1.526  3.007
Nonlinear regression model
  model:  size ~ preece(age, beta)
   data:  parent.frame
 beta1  beta2  beta3  beta4  beta5
55.413 42.361  0.287  1.526  3.007
 residual sum-of-squares:  111041
> res.preece = nls(size ~ preece(age,beta),
+ start = list(beta=c(Y=65,d=10,k1=.1,k2=.5,t0=0)))
> coef(res.preece)
 beta1  beta2  beta3  beta4  beta5
55.413 42.361  0.287  1.526  3.007
> curve(preece(x,coef(res.preece)),add=T,col="blue")
```

With these starting points we get an estimate. Notice, we store the results into `res.preece()`. From this we can extract the coefficients with the `coef()` function. By writing the function to accepts a vector of coefficients we can easily use the found coefficients in the `curve()` command.

Exercise 0.1 Try to fill in this table with estimated values for the model. In each case, once you have fit the model, save the results as above.

Model	parameters	estimates
Gompertz	y_∞, a, k	
von Bertalaffy 3	Y_∞, k, t_0	
Richards	Y_∞, k, t_0, m	
Weibull	Y_∞, Y_0, k, m	56.4528, 0.4850, 0.2264, 1.6444
Jolicouer	Y_∞, b, m	
Johnson	Y_∞, k, t_0	
Preece-Banes 1	$Y_\infty, d, k_1, k_2, t_0$	55.413, 42.361, 0.287, 1.526, 3.007

Exercise 0.2 Try to make a graph with all of the different models. Visually which one fits best?

3 Model selection using sum of squares

In modeling, there is a balance between using more parameters which makes a better fit, but also creates a more complicated model. A simple criteria for "selecting" a model takes this into account. Models are weighed according to the criteria

$$\frac{SSQ(m)}{n-2m}$$

A model with m parameters has a sum of squares (the residuals squared). As you add parameters, this goes down. However, we divide by a smaller number so the ratio may go up. These values for the two models fit above are found with the `resid()` command to find the residuals, and `sum()` to add them up.

```
> sum(resid(res.weibull)^2)/(length(age) - 2*4)
[1] 27.87
> sum(resid(res.preece)^2)/(length(age) - 2*5)
[1] 27.72
```

By this criteria, the Preece model fits better.

4 Model assessment when a distribution on W_i is assumed

In the linear regression model, when we assume the errors are iid normals then we can say more about the estimates for the coefficients (they too are normal, confidence intervals can be found with *t*-statistics, etc.). Let's investigate the distributions of the data to see if they are normally distributed.

A simple test can be to plot boxplots for each value of the age. Boxplots for normal data have a few close outliers and are symmetric. The boxplots can be made with

```
> boxplot(size ~ factor(age))
```

The data isn't exactly normal and in some cases seems to have a long tail.

If we proceed as though the data were normal, then we can use the likelihood function to perform model selection.

5 Model selection based on AIC

The statistical models

$$y_i = f(x_i|\beta) + W_i,$$

where W_i are assumed to be normal with mean 0 and variance σ^2 specify a model which can be analyzed using maximum likelihood. To do so, we use the likelihood function

$$L(\beta,\sigma^2) = f(x_1,\ldots,x_n,y_1,\ldots,y_n|\beta,\sigma^2).$$

(Here f is the density function.) As the samples are assumed to be iid the density is a product, and we *assume* the randomness is that $y_i - f(x_i|\beta)$ has the normal distribution we have

$$\begin{aligned} L(\beta,\sigma^2) &= \prod (1/2\pi\sigma^2)^{1/2} e^{\frac{1}{2\sigma^2}(y_i - f(x_i|\beta))^2} \\ &= (1/2\pi)^{n/2}(1/\sigma^2)^{n/2} e^{-\frac{1}{2\sigma^2}\sum (y_i - f(x_i|\beta))^2}. \end{aligned}$$

So the negative log-likelihood function is

$$-\log L(\beta,\sigma^2) = n/2\log(2\pi) + n/2\log(\sigma^2) + \frac{1}{2\sigma^2}\sum (y_i - f(x_i|\beta))^2.$$

To maximize this, if the function is nice, all the partial derivatives will be zero. First, taking a derivative in σ^2, we get $\sigma^2 = 1/n\sum (y_i - f(x_i|\beta)^2$ at the minimum value. We can put this into our equation, and then observe when this value holds we need to minimize (set $A = \sum (y_i - f(x_i|\beta)^2)$)

$$\log L(\beta,\sigma^2) = n/2\log(2\pi) + n/2\log(A) + (n/2)$$

This is smallest, when A is smallest, but that is just the least squares problem in disguise.

In summary, the statistical model has a likelihood function, the negative log likelihood function yields maximum likelihood estimators for σ^2 and the β which coincide with the least-squares estimates. Furthermore, the models can be compared using the AIC criteria which applies to likelihood based approaches. The AIC for a given model with m parameters is

$$AIC = -2\log L(\widehat{\beta}, \hat{\sigma}^2) - 2m.$$

For comparison, the Schwarz Bayesian criterion for a model with m parameters and n observations is

$$BIC = -2\log L(\widehat{\beta}, \hat{\sigma}^2) - 2m\log(n).$$

The results of `nls()` have an `AIC()` function which computes the AIC. For the two models we fit above

```
> AIC(res.preece)
[1] 24738
> AIC(res.weibull)
[1] 24760
```

The Preece model fits "better" by this criteria.

As an aside, we can combine our results into a list, and apply the `AIC()` function this way

```
> res = list(preece=res.preece,weibull=res.weibull)
> sapply(res,AIC)
 preece weibull
  24738   24760
```

which has advantages when all the models are to be compared.

Exercise 0.3 Figure out the AIC values for each of the models above. Which is smallest?