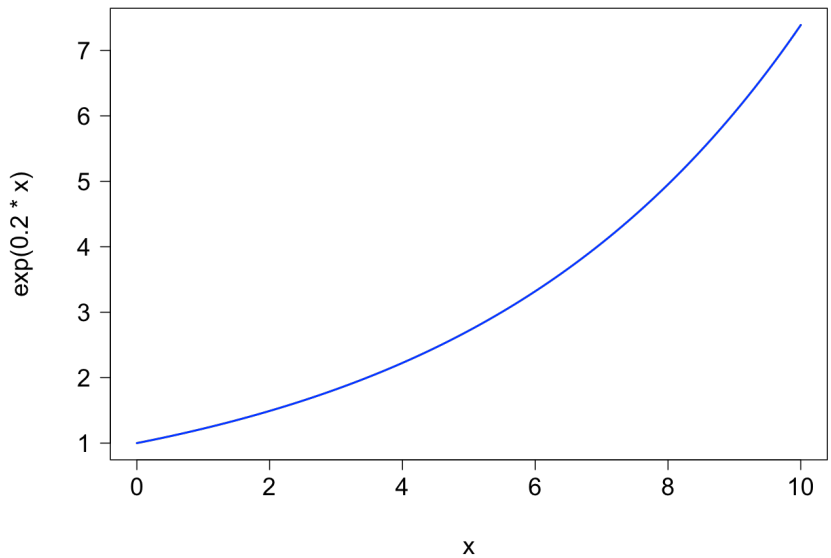
Lecture 3. Nonlinear functions

One of the challenges of analyzing biological data is appropriately modeling the shape of quantitative relationships. Linear relationships are fairly common, and are easily modeled using linear models (i.e., lm()). However, many biological processes are fundamentally nonlinear, and the common tools for modeling non-normal response variables (i.e., generalized linear models) involve non-linear functions that are important to understand. In this lecture we'll think about common nonlinear functions and how to understand their shape and parameterization.

Plotting curves. Plotting nonlinear functions is a great way to develop intuition for what they look like and what the parameters do. Also, when modeling your data you'll want to plot fitted curves to compare to the data they're representing. The curve() function is easy to use, you just feed it the equation for the function with 'x' used to represent the x-axis:

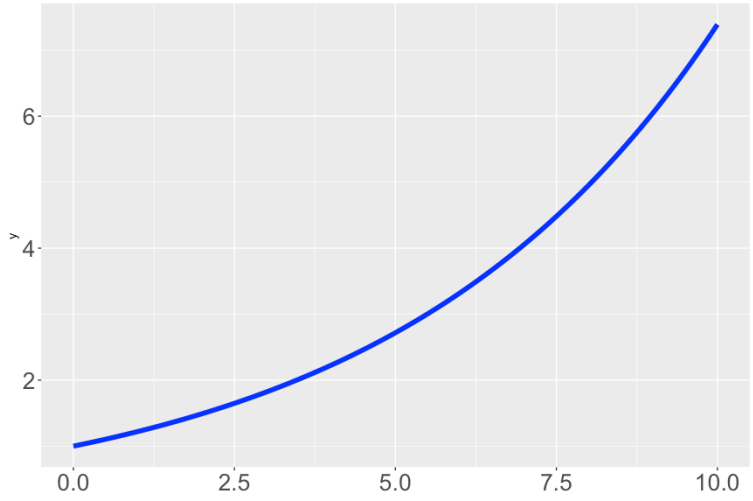
```
par(cex.lab = 1.5, cex.main = 1.5, cex.axis= 1.5, las = 1, mar = c(5,7.
5,2,1), mgp = c(4,1,0))


curve(exp(0.2*x), from = 0, to = 10, col = 'blue', lwd = 2)
```



Here I've plotted an exponential function, *a**exp(*b**x), with the parameters *a* = 1 and *b* = 0.2. The par() function sets some of the plot formatting for the graphics device; *cex* increases the size of all the labels, and *las*=1 makes the numbers on the axes horizontal. Within the curve() function, *from* and *to* set the range on the x-axis.

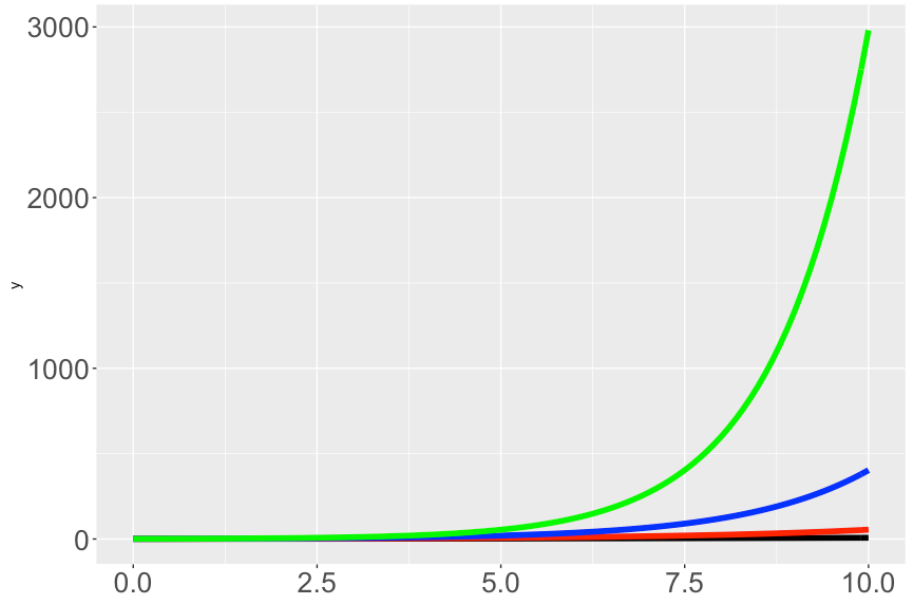
In ggplot2 we can use the geom 'function' to draw curves:

```
ggplot() + xlim(0, 10) + geom_function(fun = ~ exp(0.2*.x), col = 'blue
', size = 2)
```



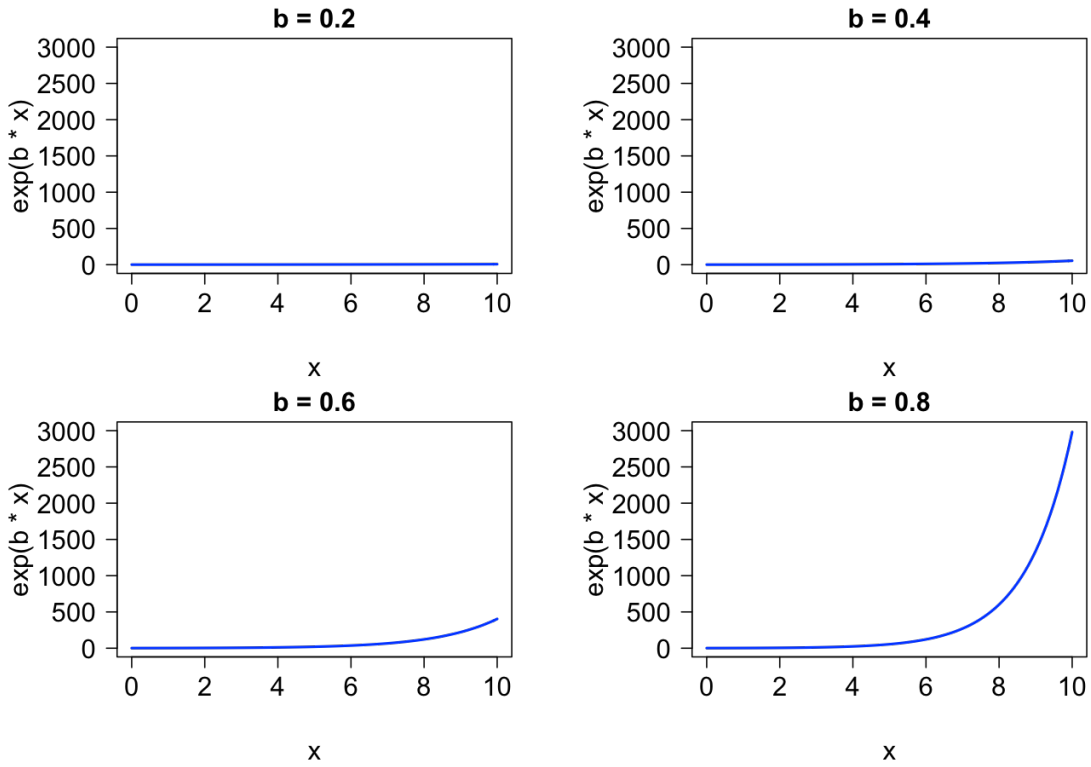
To ask how the shape of the curve changes when we change the parameters of the curve, we need to iterate curve drawing. A simple way is to just add layers, though with many layers this will become inefficient:

```
ggplot() + xlim(0, 10) + geom_function(fun = ~ exp(0.2*.x), col = 'black
', size = 2) +  geom_function(fun = ~ exp(0.4*.x), col = 'red', size =
2) + geom_function(fun = ~ exp(0.6*.x), col = 'blue', size = 2) + geom_f
unction(fun = ~ exp(0.8*.x), col = 'green', size = 2) + theme(axis.text
= element_text(size = 20))
```



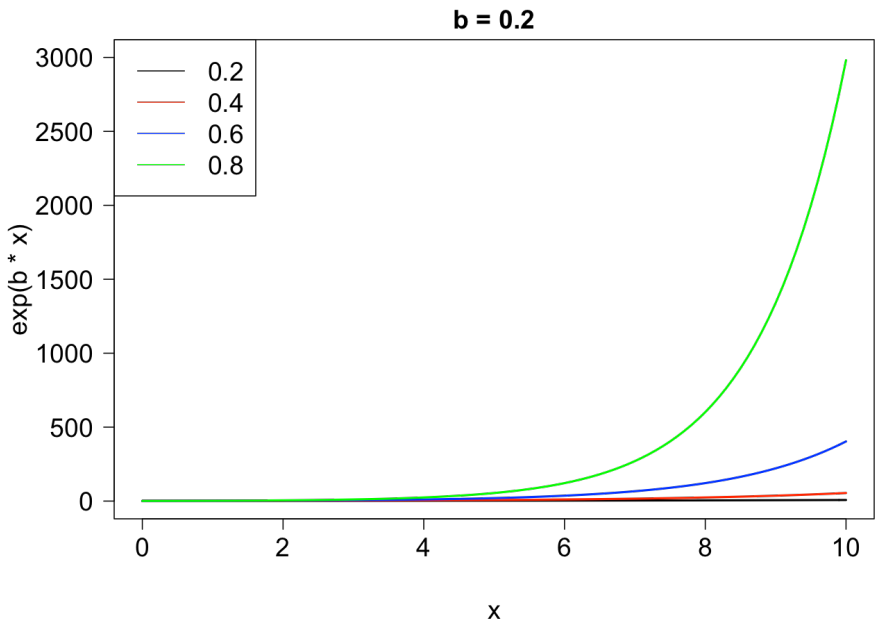
I've also used the paste() function so that each plot gets an appropriate title, by pasting the string "b =" to the currently define value of *b*. When we compare the four plots above, the range on the y-axis is very different between the plots. This is because the curve() function by default plots only the range of the y-axis produced by the function. If we want to compare these four plots more easily, we can give them all the same range on the y-axis with the option *ylim*:

```
#define 4 values of 'b' to look at
b.values = c(0.2,0.4,0.6,0.8)

#set up to make a 2 by 2 grid of plots
par(mfrow = c(2,2))

#use a for loop to draw a curve for each b.value
for (i in 1:length(b.values)) {

  b = b.values[i]

  curve(exp(b*x), from = 0, to = 10, col = 'blue', main = paste("b =",
b), ylim = c(0, 3000), lwd = 2)

  }
```



Now it is much clearer that increasing the value of *b* causes the curve to rise more steeply. You can also plot many curves on one plot, using add=TRUE:

```r
#define 4 values of 'b' to look at
b.values = c(0.2,0.4,0.6,0.8)
b.colors = c('black', 'red', 'blue', 'green')

#reset the graphics device to plot just one plot
par(mfrow = c(1,1))

#use a for loop to draw a curve for each b.value
for (i in 1:length(b.values)) {

  if (i == 1) add.the.curve = FALSE
  if (i > 1) add.the.curve = TRUE

  b = b.values[i]

  curve(exp(b*x), from = 0, to = 10, col = b.colors[i], main = paste("b
=", b), ylim = c(0, 3000), add = add.the.curve, lwd = 2)

  }

legend('topleft', lty = 1, col = b.colors, legend = b.values, cex = 1.5)
```



Here we have to do a little more work to make the 4 curves distinguishable. I defined a color for each curve, and then used the legend() function to add a legend that plots lines (lty = 1) with the right colors (col = b.colors) and the corresponding labels (legend = b.values). I also added something new to the for loop, a variable *add.the.curve*, which is set to FALSE when i = 1, to make the initial plot, and set to TRUE for i>1, to add the subsequent curves to the first plot.

Thinking about the shape of curves. Plotting curves is an essential tool for understanding the relationships they represent. However, the part of the curve we see will depend on the range of x-values over which we plot it, and so it's good to have some rubrics for looking at a formula and understanding what it implies. The most important things to pay attention to are:

- *what happens at the limits*? as $x \to \infty$ or $x \to -\infty$ (or for functions defined for $x > 0$, as $x \to 0$), what does the function do? And how do the parameters of the function affect this behavior? For most functions useful in biological statistics, the function will either diverge (to $+\infty$ or $-\infty$) or converge to an asymptote.
- *what happens in the middle?* For intermediate values of the function, what is its shape and how do the parameters of the function affect this?

Now we'll go through a set of functions commonly used in biology to develop a sense for these behaviors.

Exponential function. The exponential function looks like this:

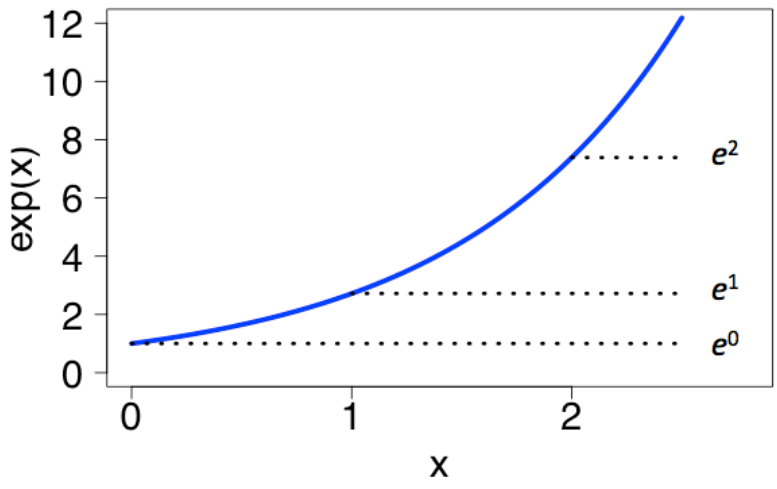
$$y = ae^{bx}$$

b is a slope parameter that determines how quickly the curve rises or falls, and a is an intercept parameter that determines the value of the function when $x = 0$. Because exp(0) = 1, when $x = 0$ then a*exp(b*x) = a, which is why a is the intercept.

The exponential has some important properties that we need to keep in mind. In particular,

$$e^{c+ax} = e^c e^{ax}$$

or in other words, when we add a number (c) to the exponent, that is equivalent to multiplying the original number (e^{ax}) by e^c. One way to think of this is that the exponential function turns addition into multiplication. This is a good way to think about the exponential curve:



Starting at $x = 0$, if we add 1 to x, then the curve increases by a factor of e, going from e^0 to e^1.

I like to think about the exponential curve in terms of exponential growth, because that draws a connection to biology and makes the parameters more intuitive. Exponential relationships arise commonly in biology because they result from any compounding process, where the amount of something increases or decreases at a rate that is proportional to the current amount. For example, the growth of a population at low density is often fit by the curve:

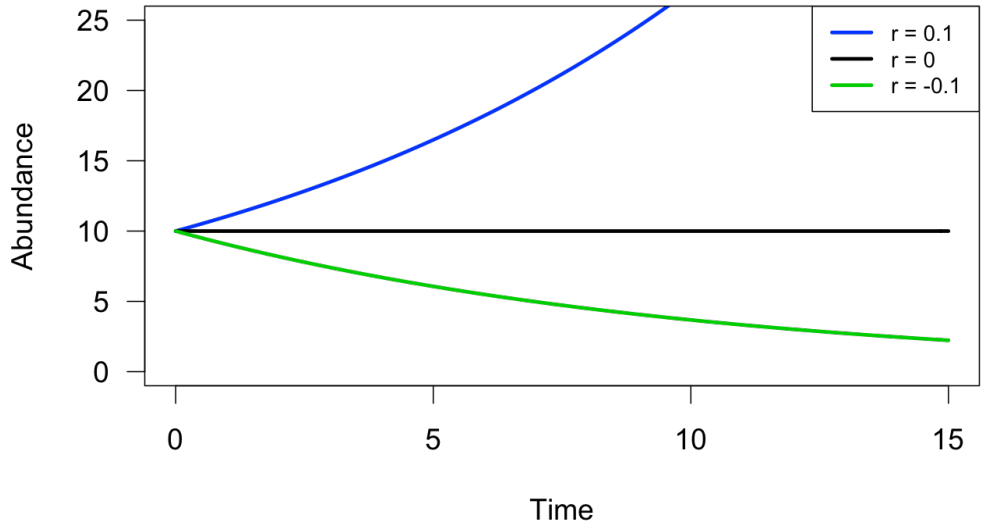
$$N(t) = N(0)e^{rt}$$

Where N(t) is abundance at time (t), N(0) is abundance at time 0, and r is the exponential growth rate, which determines how quickly the curve rises or falls. By comparison with the previous equation, you can see that here N(0) is a and r is b. Population abundance over time increases (or decreases) exponentially because the rate at which new organisms are produced is assumed to be directly proportional to how many organisms there currently are:

$$\frac{dN}{dt} = rN$$

When this equation is integrated to solve for $N(t)$, the solution is $N(t) = N(0)e^{rt}$. You can see from this equation that if r is positive, the population will increase (dN/dt will be positive), and if r is negative the population will decline. If $r = 1$ then the population will not change. This helps us understand what the exponential function looks like:

```
#exponential growth or decay
par(cex = 1.5)
r = 0.1
curve(10*exp(r*x), from = 0, to = 15, col = 'blue', lwd = 3, las = 1, yl
im = c(0, 25), xlab = 'Time', ylab = 'Abundance')
r = 0
curve(10*exp(r*x), col = 'black', lwd = 3, add = T)
r = -0.1
curve(10*exp(r*x),col = 'green3', lwd = 3, add = T)
legend('topright', lty = 1, lwd = 3, col = c('blue', 'black', 'green3'),
 legend = c('r = 0.1', 'r = 0', 'r = -0.1'), cex = 0.75)
```



When the slope parameter of the exponential function is positive, the curve goes to infinity, at an ever increasing rate. When the parameter of the exponential function is negative, the curve declines asymptotically to zero, declining at an ever slower rate. It's important to notice that *the exponential function is never negative*. Even as the curve declines towards zero, it never actually reaches zero (that's what it means to approach a number asympototically: the function is headed towards it but only gets there in the limit of x->Infinity). We can also see that the exponential function is *monotonic*, which means that it is either always increasing (positive *r*) or always decreasing (negative *r*), or is constant ($r = 0$), but it never changes direction (that's what monotonic means, the function never changes direction). Likewise, if the curve is increasing (positive *r*) that also means that for very negative values of *x*, the curve declines towards zero (which you can see in the earlier plots in this lecture). Here's an example of exponential growth in nature, the increase of *Spartina*, an invasive intertidal cordgrass in Washington:

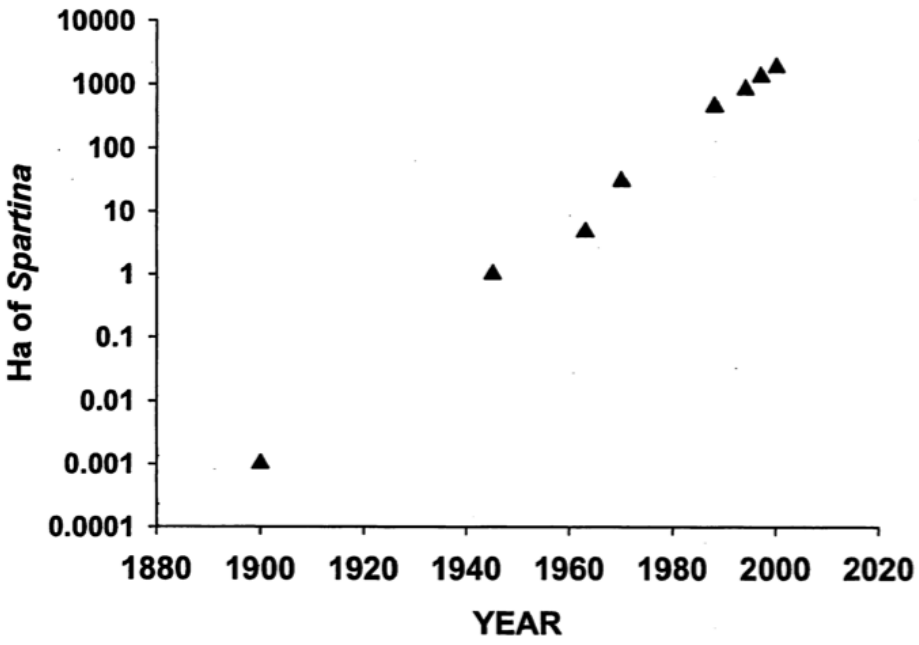
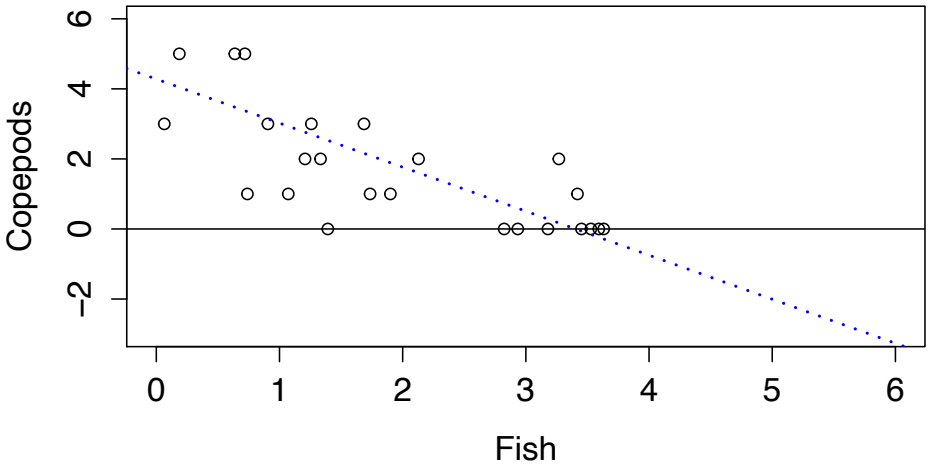


Figure 2.9. The intertidal cordgrass Spartina alterniflora has spread exponentially in Willapa Bay, Wa. since introduction in the early 20th century.

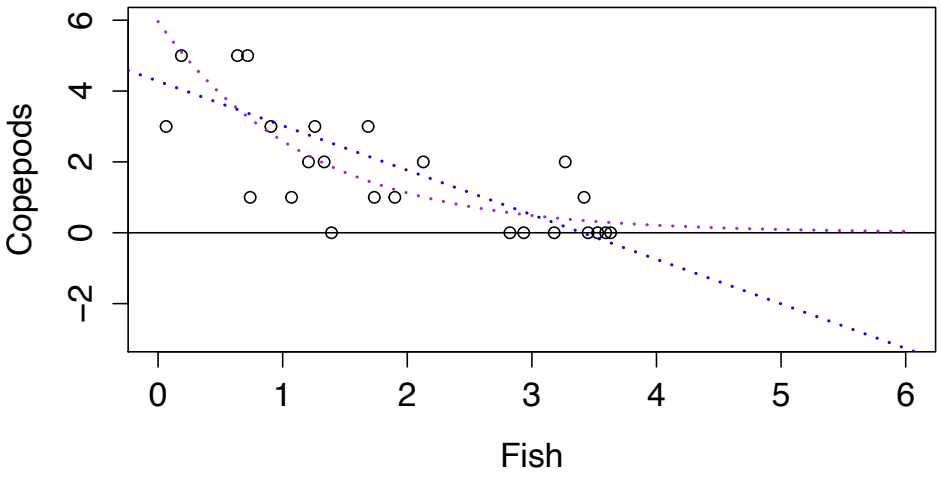
Most of your use of the exponential curve in statistics will not be related to population growth, although it is a helpful way to understand the curve. When it comes to modeling data, the properties of the exponential function mean that it will be useful for modeling data that is always positive (like count data), and when the relationship is monotonic. I'm belaboring the properties of the exponential function here because when we get to generalized linear models, the exponential function will be the default function to use with count data, in combination with a Poisson or negative binomial distribution. It's useful to compare what would happen if we tried to model Poisson-distributed data with a linear function instead of the exponential. For example, let's say that we're sampling copepods along a transect in the ocean, and that we want to test whether the abundance of copepods decreases as the abundance of planktivorous fish increases. The copepod abundances are count data, so we want our function to predict the mean of the Poisson distribution:

$$\mu_i = a + b*\text{Planktivores}_i$$
$$\text{Copepod}_i = \text{Poisson}(\mu_i)$$

Here μ_i is the mean number of copepods when planktivore abundance = Planktivores_i. The problem with using a linear model is that as the predictor gets small (for positive b) or large (for negative b), eventually the line goes below zero:



When Fish is above ~3.6, the line for Copepods goes negative. Predicting negative copepods is a problem, both logically and statistically. Because the mean of the Poisson function (lambda) is by definition > 0, you cannot use a Poisson error distribution to fit the blue line. If the number of copepods were always large, i.e. not close to zero, then using a linear model with a normal error distribution might work fine. But when dealing with smaller numbers that are appropriately modeled as counts (with the Poisson), an exponential function will typically work well:

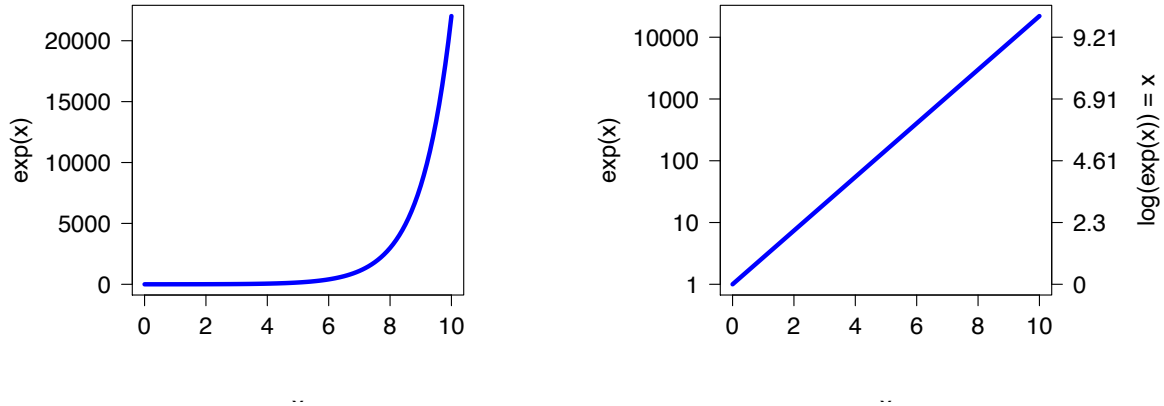


The exponential curve (in purple) declines to zero as planktivorous fish become abundant.

The logarithm. Like the exponential function, the logarithm shows up a lot in statistical modeling, so we will spend a little time unpacking it. As I mentioned when

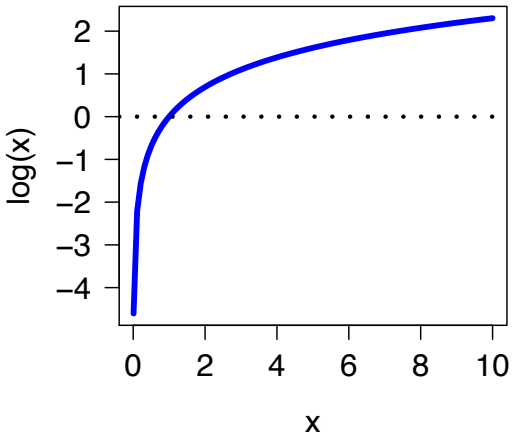
introducing the lognormal distribution in lecture 3, there are many multiplicative processes in nature (e.g. exponential growth), and those processes tend to result in lognormally distributed variation. The lognormal distribution has that name because if *X* is lognormally distributed, then log(*X*) is normally distributed.

Another way to look at the logarithm is that *the logarithm turns multiplication into addition*. The rules for logarithms say that log(a*b) = log(a) + log(b). This fact is actually the reason that logarithms were discovered in the first place, by Napier in 1614. Back then it was time-consuming to multiply or divide large numbers, e.g. for use in astronomy and navigation. With the discover of the logarithm, to multiply a*b you could instead add log(a)+log(b) to get log(a*b), and then look up the antilog of log(a*b) to get a*b. This was a much quicker calculation. Napier and others spent years making tables of logarithms to facilitate these calculations. The fact that logarithms turn multiplication into addition is also the reason why it's often useful to plot (or analyze) variables on log-axes:



On the left is exp(x) on the original scale, and on the right the y-axis is on a log scale. As you can see from the axis labels on this plot, multiplying by a factor of ten (e.g. from 10 to 100, or from 100 to 1000) is equivalent to adding one unit.

Another way to define log(x) is to say that it's the inverse of the exponential function. That means that log(exp(x)) = x. This will be important to remember when we get into generalized linear models. Just like exp(x) can take any x (positive or negative) and always spits out a positive number, log(x) can only take positive numbers and spits out numbers ranging from –Inf to Inf:



The fact that log(0) does not exist will come up when we consider data transformation.

Saturating functions

Functions that increase and then saturate at some asymptotic value are important in biology because can represent dynamic processes that increase with the supply of some resource, and then saturate due to limitation by some rate-limiting factor. A classic example is the *Michaelis-Menten* model:

$$v(S) = \frac{V_{\max} S}{K + S}$$

Here $v(S)$ is the rate of an enzymatic reaction as a function of the substrate concentration S, V_{max} is the maximum rate of reaction, and K is the half-saturation constant. Here is an example of the equation fit to the rate of phosphate uptake by phytoplankton cultures:

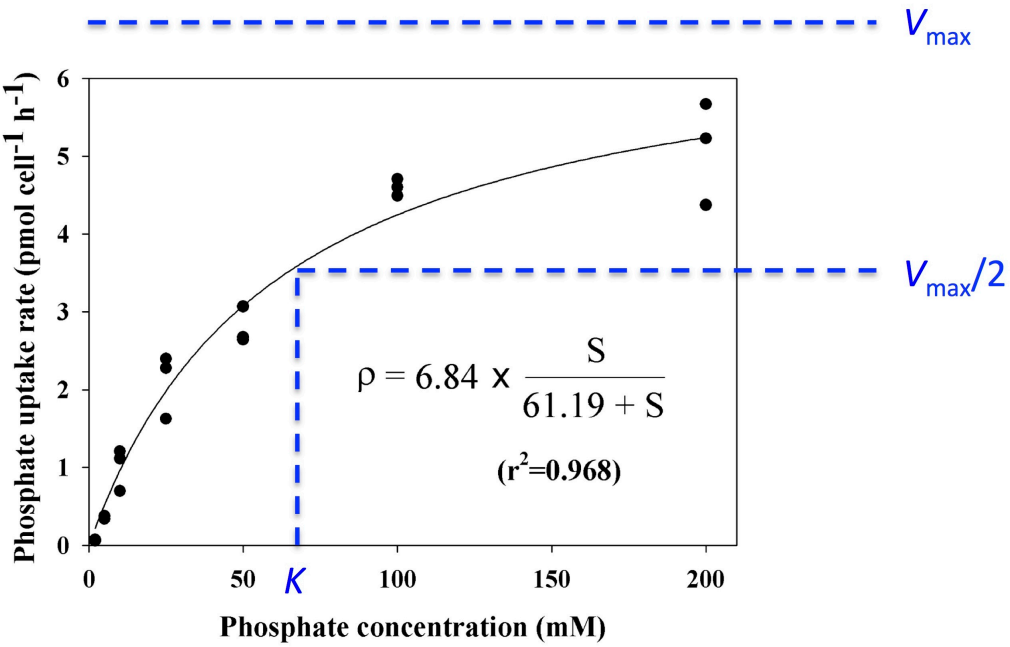


Fig. 2. Phosphate uptake rate of *Nitzschia* sp. as a function of ambient phosphate concentration. (Yamamoto et al. 2012)

I have labeled the curve with the parameters defined above. The curve starts at the origin (0,0), and at first phosphate uptake increases linearly as more phosphate is supplied. The rate of increase starts to slow until it ultimately approaches an asymptote, V_{max}. K is the substrate concentration (phosphate) at which the uptake rate is at half of the maximum, $V_{max}/2$. The role of K is to determine how quickly the curve approaches the asymptote; for a small K, the curve will rise steeply and approach the asymptote quickly.

For this experiment the curve fits the data very well, and this is often found when measuring the physiological kinetics. Why does the curve work? There is a mechanistic derivation from first principles that makes some approximations to arrive at this simple function. But I think it's fairly intuitive why a curve of this shape is necessary. When phosphate is at low concentration, the rate of reaction is determined by how often a phosphate molecule in the medium bumps into an uptake transporter. This means that as more phosphate is supplied, the rate of collisions will increase in direct proportion (linearly). However, as phosphate supply increases, not all transporters will be available at all times, because some will be busy transporting capture phosphate molecules. This causes the curve to start to bend. Eventually, at very high phosphate supply, the rate at which transporters encounter phosphate is no longer limiting. Rather, the limiting step is the rate at which each transporter can bring a phosphate molecule into the cell. This sets the maximum uptake rate, V_{max}.

The Michaelis-Menten curve is commonly used in ecophysiology and other fields, but the same equation, or similar saturating curves, come up in other scenarios as well. As common example from ecology is the Type II functional response:

$$f(R) = \frac{aR}{1+ahR}$$

Here $f(R)$ is the per capita feeding rate of an animal as a function of the concentration of its prey or resource (R); a is the attack rate, and h is the handling time. Here's an example from field data on wolves eating caribou:

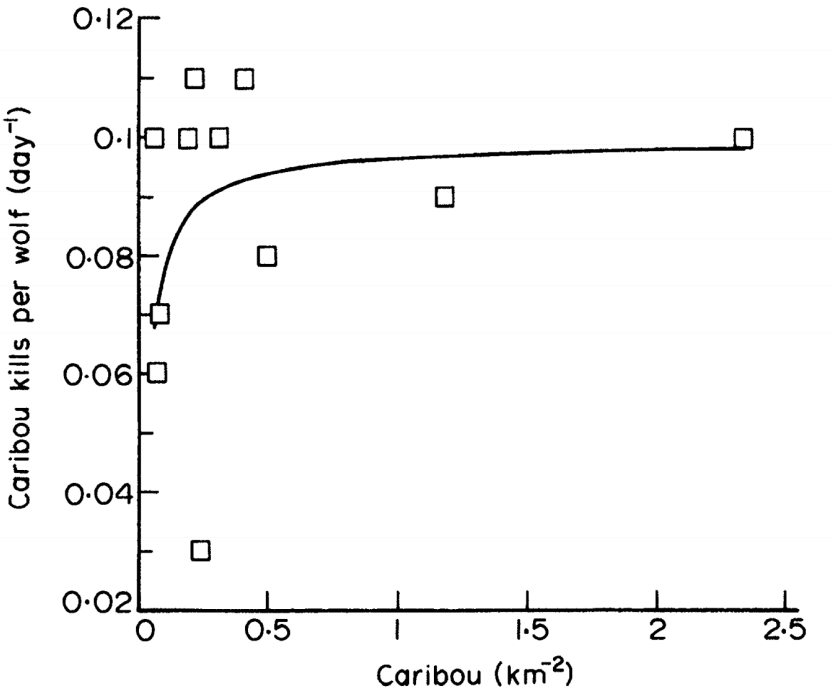


Fig. 2. Observed daily winter kill rate estimates fitted by Holling's (1965) disc equation ($R^2 = 0{\cdot}80$, the assumption that kill rate = 0 at 0 caribou per km^2 was necessary to fit equation 1, $Y = (3{\cdot}616 * N * 1)/[1 + (3{\cdot}616 * 10{\cdot}066 * N)]$.

(Dale et al. 1994)

This equation can be derived from various mechanistic models of how animals forage and eat. Mathematically it is the same equation as the Michaelis-Menten equation, although it looks different at first. The equation can be rearranged and written in terms of an asymptote $\frac{1}{h}$ and a half saturation constant $\frac{1}{ah}$. The fact that the Type II functional response and the Michaelis-Menten equation have the same mathematical form is actually pretty intuitive. In both cases we're modeling consumption, either consumption of a chemical by an enzyme, or consumption of prey by an animal. In both cases the consumption rate at low density is determined by how often the two components encounter each other, and so consumption rate

rises proportional to prey/substrate density. Likewise, in both cases when the supply rate is high the rate of consumption is limited by how fast the predator/enzyme can process the prey/substrate. In Holling's original derivation of the functional response this limiting rate is the "handling time", i.e. how long it takes the predator to capture/subdue/eat its prey.

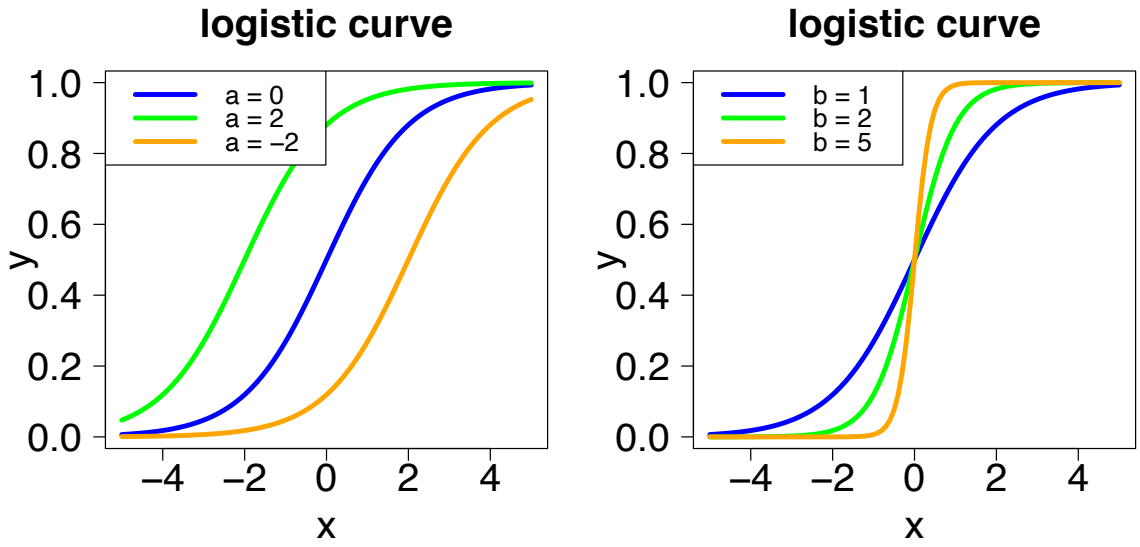
Yet another equation in biology that has the same mathematical form is the Beverton-Holt model that is common in fisheries modeling. This model assumes that the number of recruits produced in year $t+1$ is a saturating function of the number of adults (the stock) in year t.

Sigmoidal functions

A sigmoidal function is a function that has an "S" shape, and the most common one you will encounter is the *logistic* curve:

$$y = \frac{\exp(a + bx)}{1 + \exp(a + bx)}$$

There are a couple different ways to write this curve that you may encounter, but we'll use this one because it relates most clearly to statistical modeling. What is the behavior of this curve? When x is a large negative number, the $\exp(a + bx)$ parts become approximately zero, because as we discussed above the exponential function goes to zero for very negative x. When x is a large positive number, $\exp(a + bx)$ becomes much larger than one; that means the numerator and the denominator of the function are both approximately equal to $\exp(a + bx)$, and so $y \approx 1$. In other words, the function ranges from an asymptote at zero to an asymptote at one:



From this plot you can see that the parameter *a* moves the curve to the left or right. For this reason a parameter like this is often called the *location parameter* of the curve, and it plays a similar role to the intercept of a straight line. The parameter *b* determines how steeply the curve rises from 0 to 1. For this reason a parameter like *b* is often called a *scale parameter*, and it plays a role similar to the slope of a straight line. Because the logistic curve starts out as an accelerating curve (for small *x* it is approximately equal to $\exp(a + bx)$), and eventually becomes a decelerating curve (as it approaches 1), that means it has an *inflection point* where it moves from decelerating to accelerating. The inflection point occurs at $x = -\frac{a}{b}$, which also happens to be the half-maximum point.

The logistic function is commonly used in biology to model data that can take on one of two states, such as alive/dead. The two states are coded as 0 and 1, and the logistic curve models the probability that the response variable is 0 or 1. This means *the logistic is the natural curve to use for binomially distributed data*, and we will use it frequently for this purpose. Here's an example of logistic regression from a study of terrestrial isopods (genus *Balloniscus*). The authors studied the tendency of the isopods to "play dead" when they are squeezed or dropped, which simulates attempts by predators to eat them. They code the results as either "responsive" (played dead) or not responsive, and tested whether responsiveness varied with body size:

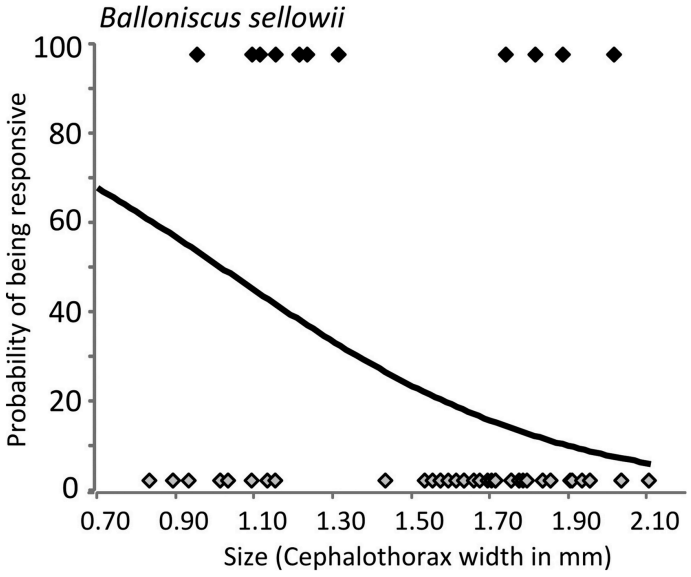
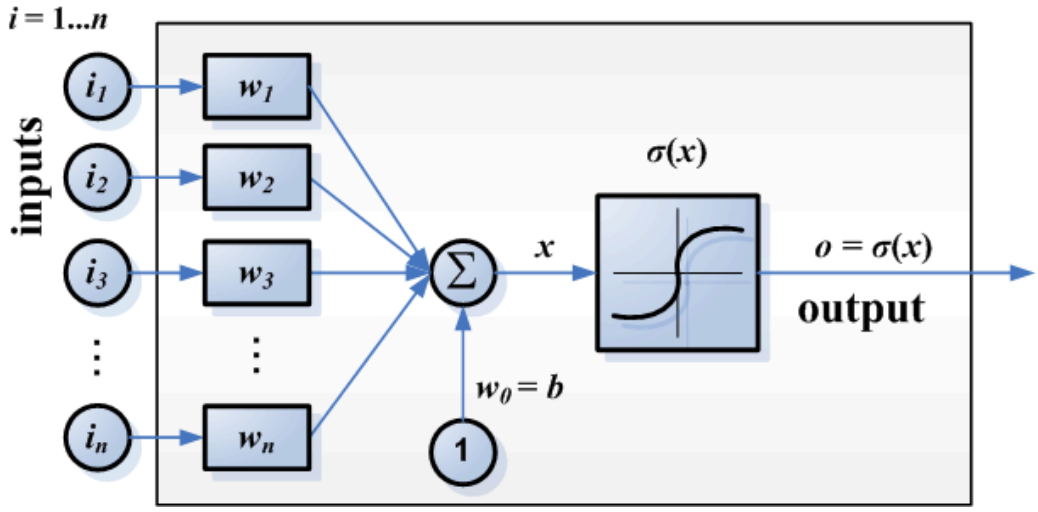


Figure 5.
Responsiveness of *Balloniscus sellowii* individuals in relation to size. The line models the probability of being responsive according to the individual size (after a logistic regression). The black and grey symbols show the responsive and non-responsive individuals, respectively.

(Quadros et al. 2012)

You may notice that over the range of the size data, the logistic curve is nearly linear. Nonetheless it is important to use an equation like the logistic because the curve is modeling a probability (of being responsive), and probabilities vary between zero and one.

An interesting use of the logistic curve that you may encounter is in artificial neural networks. We can think of the logistic curve as taking a continuous range of inputs and converting those into an output that ranges from zero to one. This is analogous to how a neuron receives electrical impulses from a variety of attached neurons, and in response the neuron can itself fire (or not), at different frequencies.



Artificial neuron, from http://www.ro.feri.uni-mb.si/predmeti/int_reg/Predavanja/Eng/2.Neural%20networks/_05.html

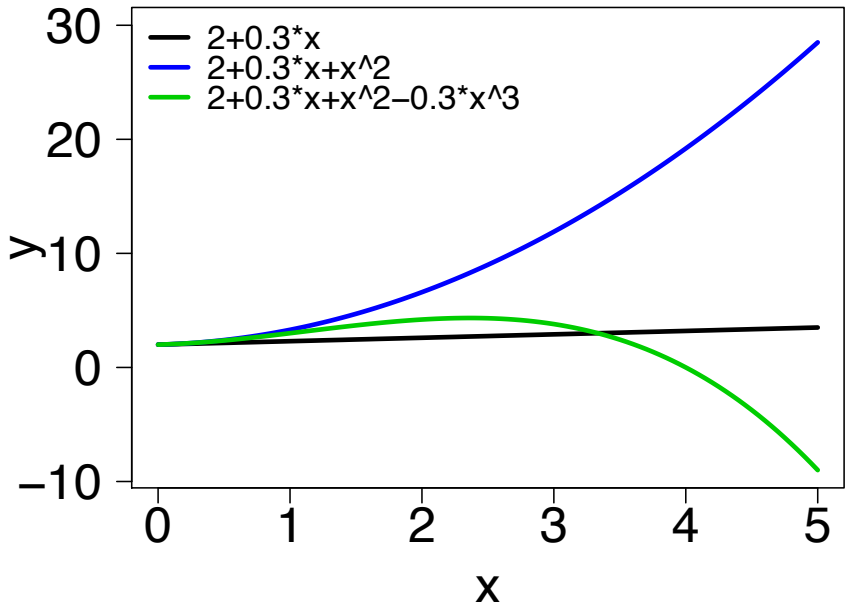
By creating a network where many artificial neurons are each receiving input from many neurons, processing that input through a logistic (or similar) function, and sending that output to many neurons, one can fit a very complex and nonlinear model that "learns" from the data you train it with. For example, when Netflix or Amazon are trying to predict what things you'll like based on how you rated other things, artificial neural networks would be one way to do this (I don't know what methods they actually use).

Polynomials

A polynomial function is a function that has the form:

$$y = a + b_1 x + b_2 x^2 + b_3 x^3 + \cdots$$

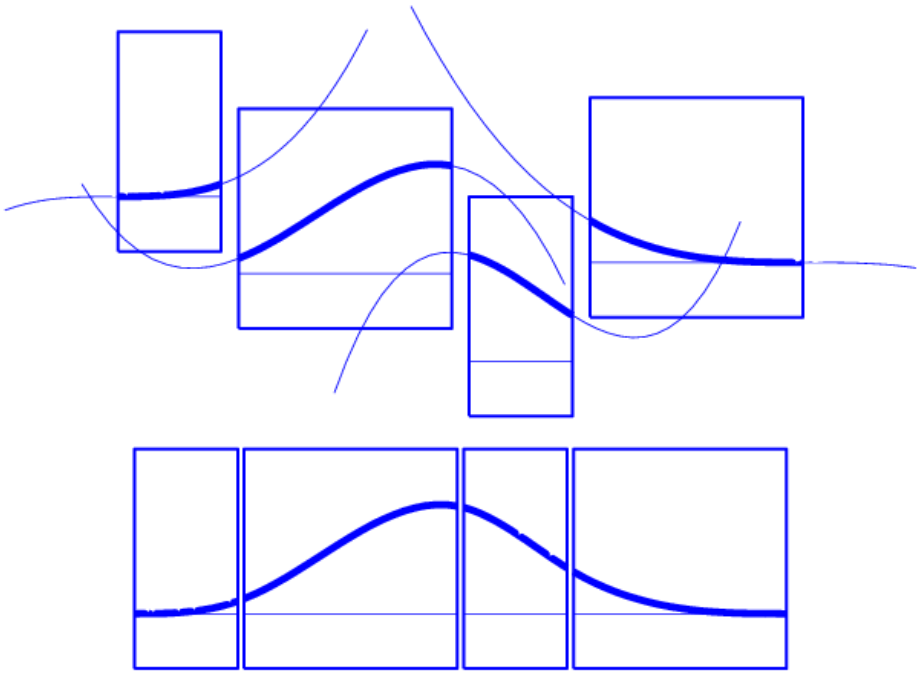
so this includes lines, quadratic curves (parabolas), cubic curves, etc.



Some example polynomials of different degree.

Although polynomials are in some ways simple to work with, they are not that common in biological statistics, perhaps because every polynomial function always diverges to either ∞ or -∞ on each end. The highest-order term in the polynomial will determine this ultimate behavior, because as x gets very big then x^2 becomes much bigger than x, and x^3 becomes much bigger than x^2, etc. So for the green curve plotted above, $-0.3*x^3$ is the highest-order term, which means that as x gets very big the function will diverge to -∞ (and when x is a large negative number $-0.3*x^3$ goes to positive infinity). In contrast, many of the nonlinear functions that we've looked at so far have an asymptote on either one end (exponential function, michaelis-menten function) or both ends (logistic function), for various reasons.

Nonetheless, polynomials will become important when we start using generalized additive models (GAMs). GAMs are useful when you want to model the fact that y is some function of x, but you don't know and/or don't care what that function is. This can be achieved using a *smoothing spline*, which is a series of polynomial functions (usually cubic polynomials) pasted together:



Cubic spline example, from http://www.mathworks.com/help/curvefit/the-b-form.html.

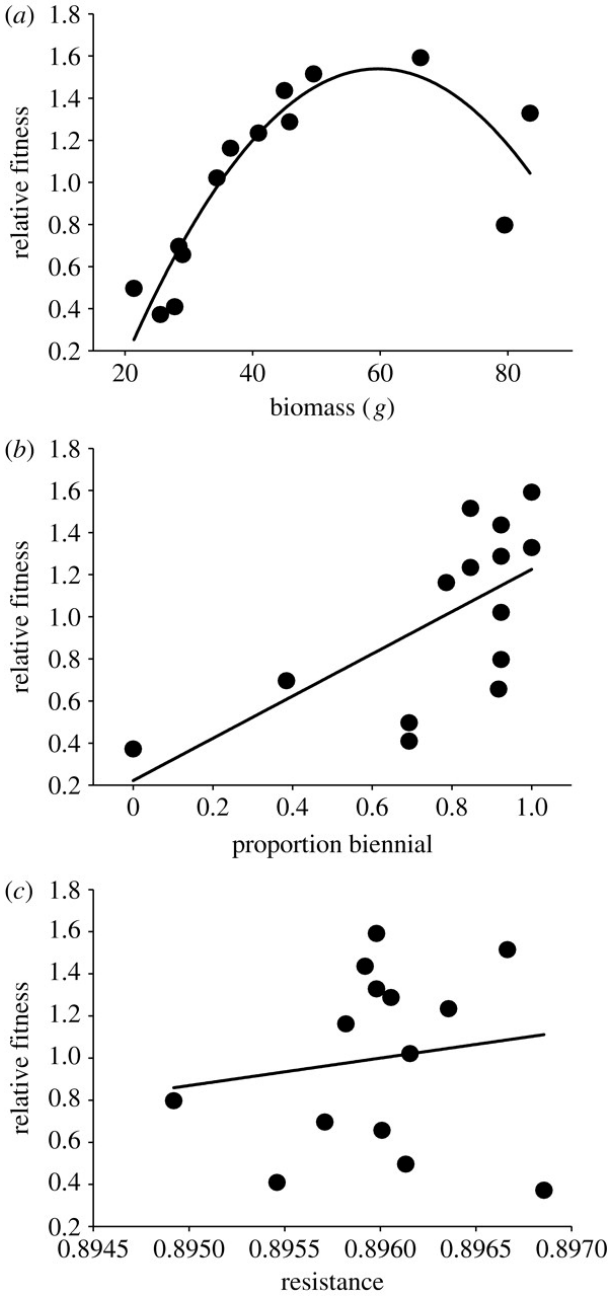
On the top this image shows four different cubic polynomials (the curved lines).the bottom, the pieces of the polynomials highlighted in the boxes are pasted together to form a single smooth curve. Luckily we won't have to do this pasting ourselves, it is automated by the software. However, it is useful to understand why using polynomials makes this a (somewhat) tractable technique to use in these kinds of models. An important reason is that *a polynomial can be estimated as a linear model*. Why is the apparently nonlinear polynomial a linear model? It's because the coefficients can be estimated using the standard linear model formula:

response = coefficient_1*predictor_1 + coefficient_2*predictor_2 + …

For example, to use a quadratic curve in a regression, $y = a + b_1x + b_2x^2$, the coefficient b_2 can be estimated by calculating x^2 and putting it into lm() as a predictor.

The relative ease of estimating polynomial coefficients has been employed for a while in quantitative genetic studies of natural selection. Here the response variable is some measure of individual fitness (e.g. fecundity), and the predictor is a trait of

that organism (e.g. body size). If the regression of fecundity vs. size has a significant coefficient for the linear term, that is evidence for directional selection, and if there is a signficant quadratic term then that may be evidence for stabilizing selection:



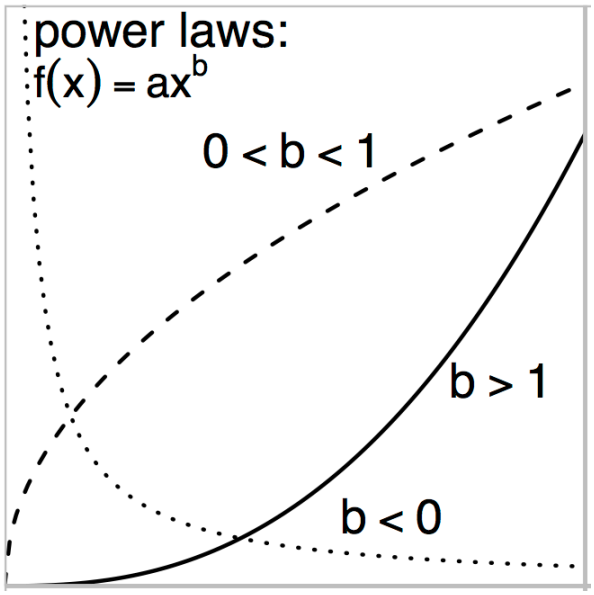
Examples of stabilizing and directional selection for the evening primrose *Oenothera biennis*, from Johnson et al. 2009.

Power functions

The last kind of function I'll discuss is a special kind of polynomial, the power function:

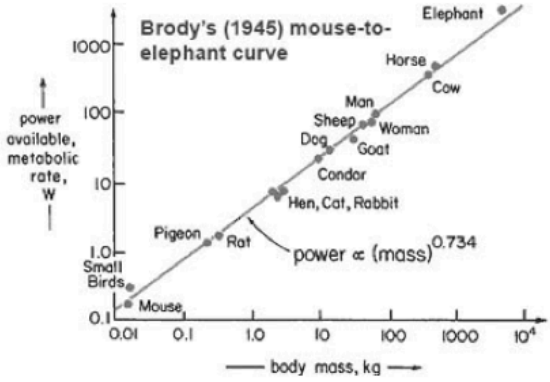
$$y = ax^b$$

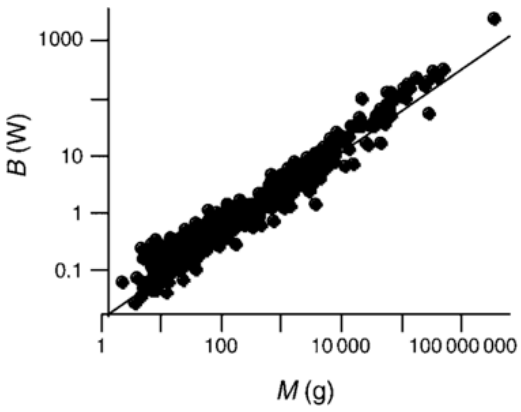
The shape of this relationship depends on the exponent b:



If $b > 1$, the curve increases at an accelerating rate, similar to the exponential function. If b is between zero and one, the curve increases at a decelerating rate with no asymptote, similar to log(x). If $b < 0$ the curve decreases at decelerating rate, with an asymptote at 0.

Power laws appear with some frequency in biology when studying scaling relationships such as the size of butterfly wings relative to body mass, or the number of species on an island as a function of island area. A classic example is the scaling of metabolic rate with body mass, which I described in the notes for lecture 1:





Basal metabolic rate (watts = joules per second) vs. individual body mass. On the

left, a classic plot by Brody going from mice to elephants. One the rigght, a modern compilation of a wide variety of mammals. The estimate slope from a regression on the log-transformed variables is 0.724 (95% CI: [0.706, 0.742]). From Duncan et al. 2007, *Ecology*, "Testing the metabolic theory of ecology: allometric scaling exponents in mammals".

The slope for this relationship is about ¾, which means metabolic rate increases with body size, but at a decelerating rate. However, in the plots this relationship is shown as a straight line because both axes are on a log scale:

$$y = ax^b$$
$$\log(y) = \log(ax^b) = \log(a) + b * \log(x)$$

Table of common nonlinear functions for analyzing biological data

Function	**Range**	**At the ends**	**Middle**	**Common Use**
Exponential	{-∞,∞}	From 0 (asymptote) to ∞	Accelerating (positive slope)	Regression with counts
Logarithm	{0, ∞}	From -∞ to ∞	Decelerating	Transformation
Michaelis-Menten	{0, ∞}	From 0 (intersects origin) to a positive asymptote	Saturating	Consumption rates
Logistic	{-∞,∞}	From 0 to 1 (both asymptotic)	Sigmoid	Binomial regression
Polynomial	{-∞,∞}	From -∞ to ∞	Depends	Splines
Power law	{0, ∞}	From 0 to ∞ (positive power) From an intercept to 0 (neg. power)	Accelerating or decelerating	Scaling relationships