Stochastic population models

SBM

February 10, 2018

Today we are going to introduce stochastic population models. In our previous population models, we assumed that the dynamics were deterministic. This means that if you know the current population size and the dynamics, you know the population size at all other times. But it is rarely the case that things are so simple - unpredictable influences ranging from the weather to asteroids can influence birth and death rates. In addition, even in constant, benign environments individual reproductive output and survival fluctuate.

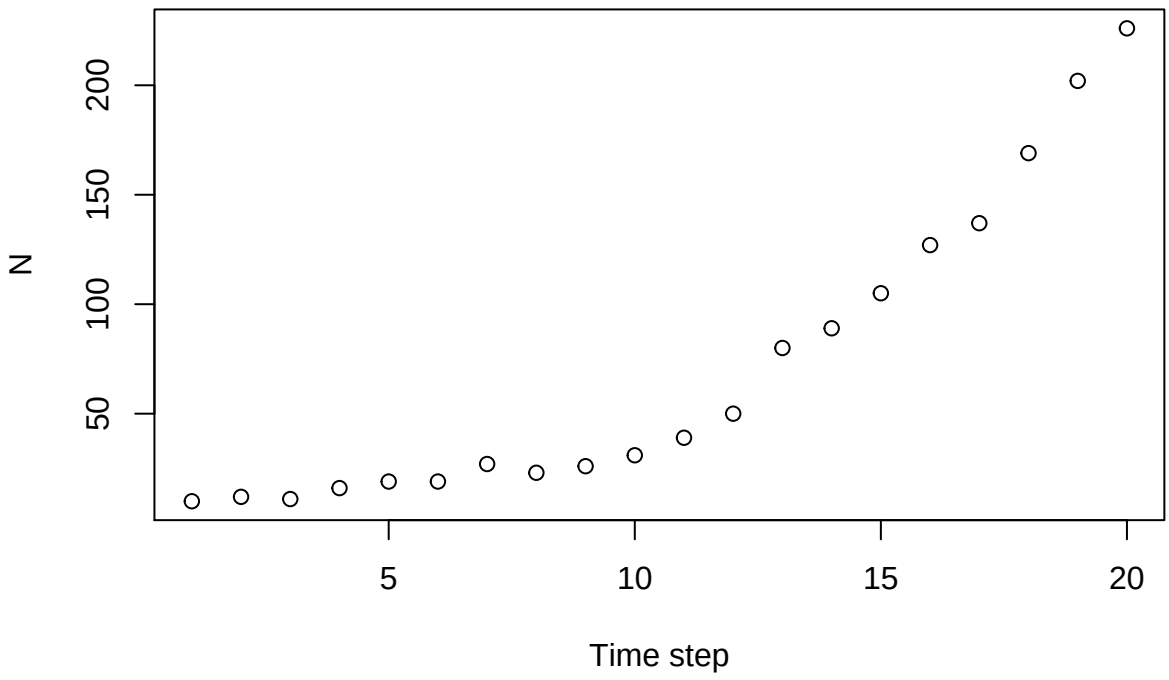
One approach to this apparent indeterminism would be to develop ever more elaborate models of individual physiological states, including individuals from multiple species, and coupling these dynamics to global climate models in the hopes of finding exactly the right description of nature. Another approach is to keep the model formulation relatively simple, but treat the unspecified variation as random. That's the approach we will take today.

The kinds of questions we typically address with stochastic population models are a. What is the probability of extinction? b. How long will it take for a population to go extinct? c. What is the long-run distribution of population sizes? d. How likely is a new species to successfully invade?

A simlation to start with Before jumping into the math, I thought it might be fun to start off with a simulation to try to make our ideas concrete. Imagine that we have an initial population of $N_0 = 10$ individuals. At each time step these individuals give birth to some random number of offspring, say $b_i = \{0, 1, \text{or } 2\}$, each with probability 1/3 and that each previously extant individual dies with probability 0.8. To do this in a simulation model we might set up a loop over time steps, inside the loop, we draw random births, draw random deaths, and tally up the total population size before moving on to the next step. To draw the random births from our list, we can use the *sample()* function. For deaths, we can use the following trick. Draw a number, call it u that is anywhere betwee 0 and 1, with equal probability. We can generate u using *runif()* The chance that u is less than 0.8 is... 0.8! So we say the inividual dies if $u < 0.8$ and lives otherwise. Note that using $u < 0.8$ returns a 1 if true and a 0 if false.

Here's a preliminary version of this simulation model-

```
T=20 #the total number of time steps
N0=10 #initial population size
bmax=2 #maximum number of offspring
pmort=0.8 #probability of dying

b<-c(0:bmax) #make list of possible births
N=array(data=0,T) #array for population sizes
N[1]=N0

#start loop
i=1
while ((i<T)&(N[i]>0)) {
  births<-sum(sample(b,N[i],replace=T))
  deaths<-sum(runif(N[i],0,1)<pmort)
  N[i+1]<-N[i]-sum(deaths)+sum(births)
  i<-i+1
}

plot(c(1:T),N, xlab="Time step",ylab="N")
```



I used a *while* loop to avoid sampling zeros over and over once things went extinct. For kicks, you should try running this a few times to see how different it ends up. What did you learn from this?

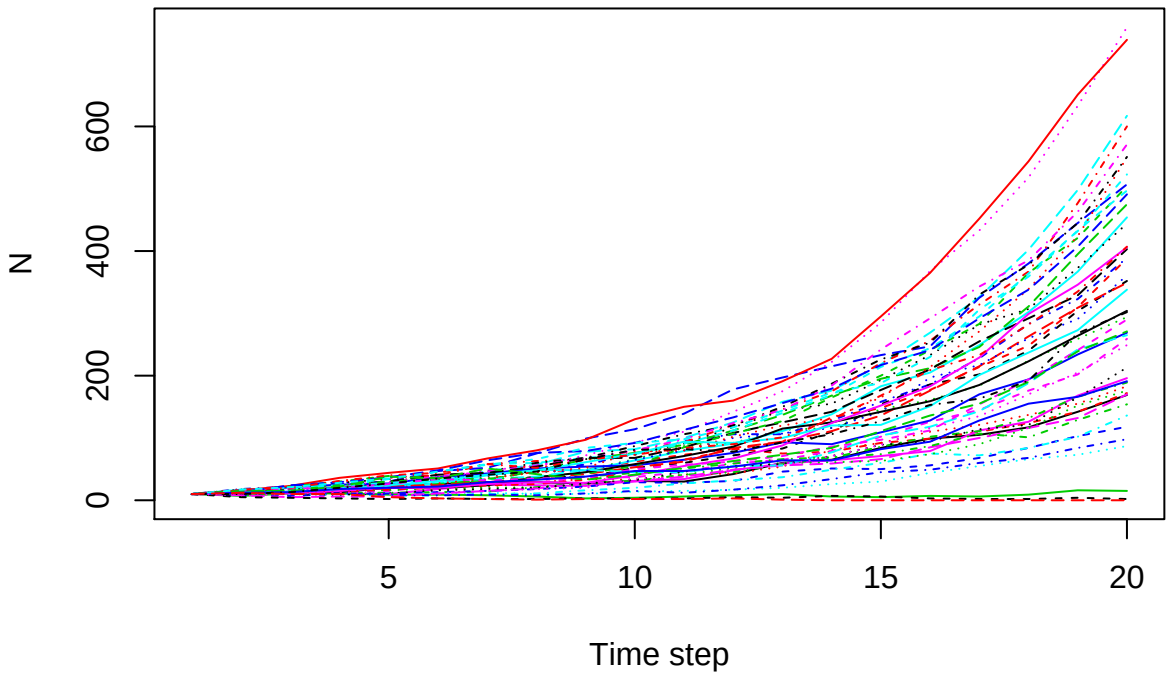
One thing that ought to be apparent pretty fast is that you dont get the same path twice. After running this a few times, I also noticed that some trajectories really take off and look smoothly exponential, while others look bouncy and stay fairly small.

Before we move on, you might try making pmort lower or bmax higher. The math is still fine, but you might see smoke come out of your computer. See if you can figure out why and what you could do about it. Suffice it to say that this code is probably fine for a few time steps and when N is small, but it is awfully slow when N is big.

To get some more general things out of this code we are going to want to generate more 'sample paths' (that is, run it a bunch more times). To do this, we could add an outer loop that repeats everything inside a whole bunch of times.

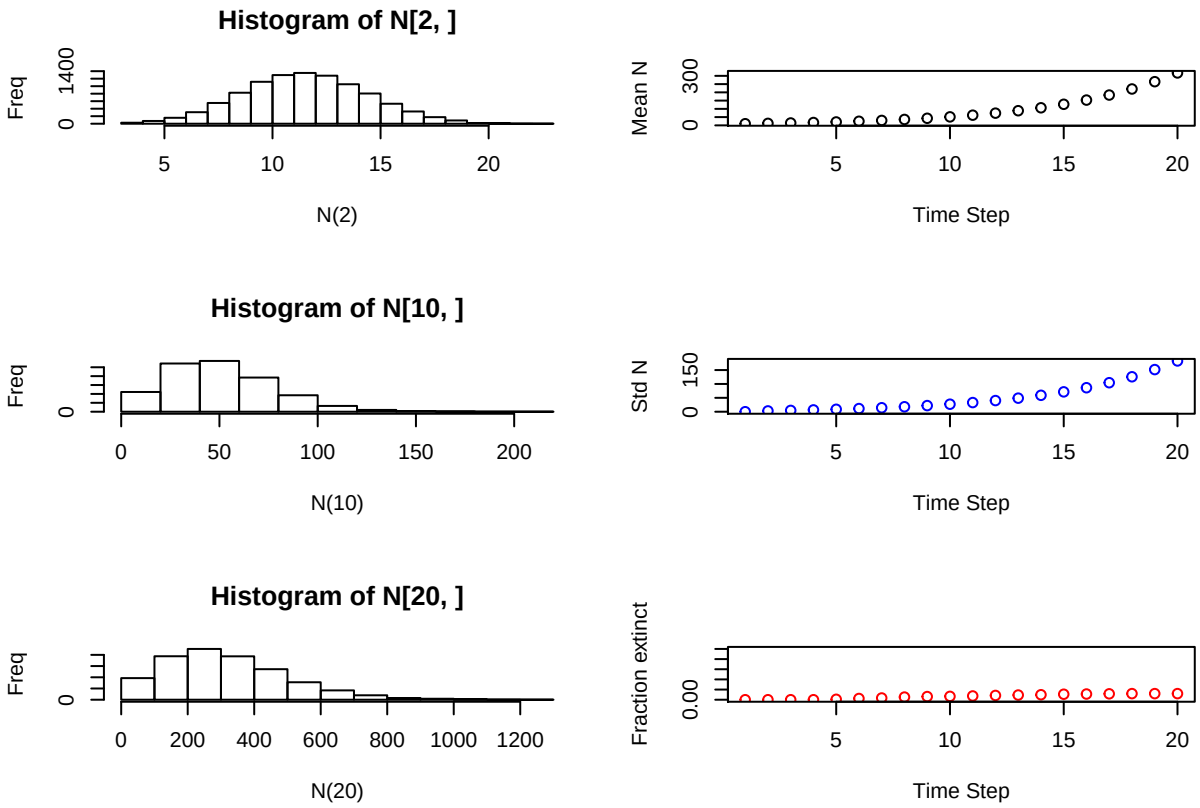
```
nsamples=10000 # the total number of sample paths we will generate
T=20 #the total number of time steps
N0=10 #initial population size
bmax=2 #maximum number of offspring
pmort=0.8 #probability of dying

b<-c(0:bmax) #make list of possible births
N=matrix(data=0,T,nsamples) #array for population sizes
N[1,]=N0

#outer loop
for (j in 1:nsamples){
```

```
#inner loop
for (i in 1:(T-1)) {
  births<-sum(sample(b,N[i,j],replace=T))
  deaths<-sum(runif(N[i,j],0,1)<pmort)
  N[i+1,j]<-N[i,j]-sum(deaths)+sum(births)

}}

matplot(c(1:T),N[,1:50], xlab="Time step",ylab="N",type="l")
```



Now, let's think about summarizing the results. We might want to look at a histogram of the final population size, calculate the mean and the variance in population size through time, or we might want to know the probability of extinction.

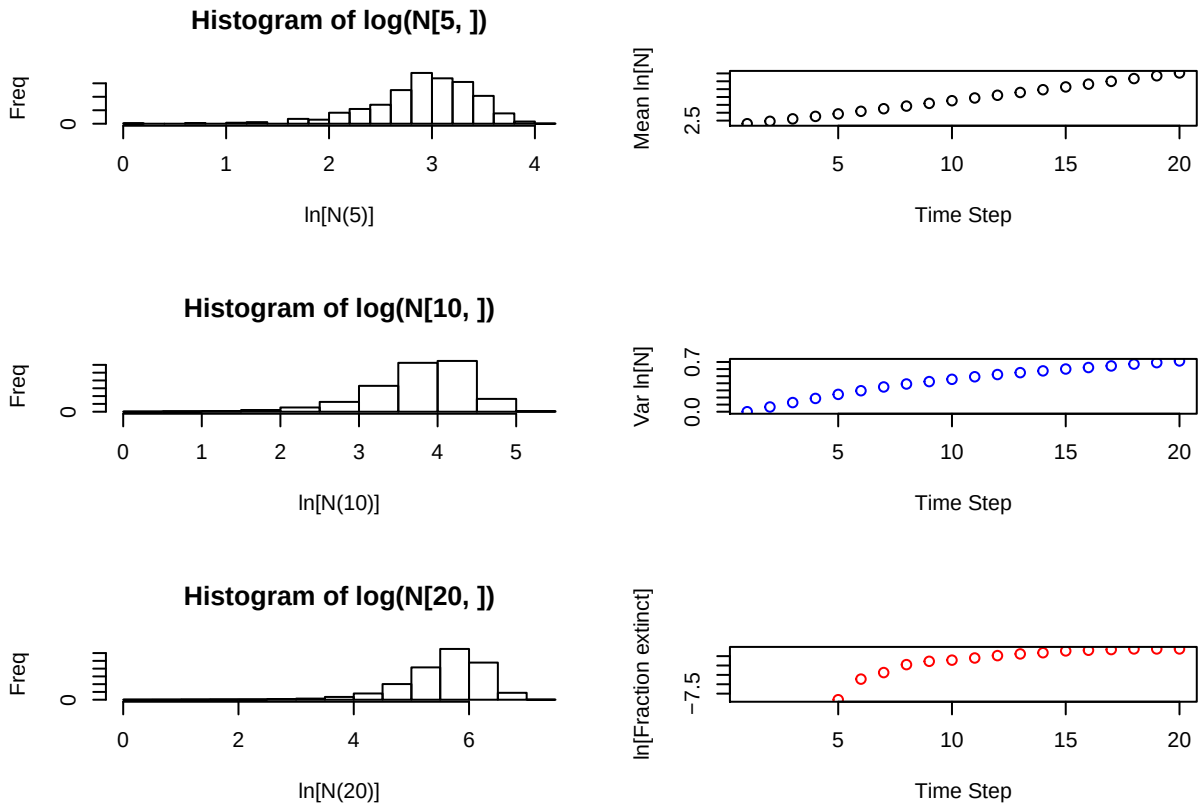
```
M<-apply(N,1,mean)
V<-apply(N,1,var)
S<-apply(N>0,1,mean)

par(mfrow=c(3,2))
hist(N[2,],xlab="N(2)",ylab="Freq")
plot(c(1:T),M,xlab="Time Step",ylab="Mean N")
hist(N[10,],xlab="N(10)",ylab="Freq")
plot(c(1:T),sqrt(V),xlab="Time Step",ylab="Std N",col="blue")
hist(N[20,],xlab="N(20)",ylab="Freq")
plot(c(1:T),1-S,xlab="Time Step",ylab="Fraction extinct",ylim=c(0,.05),col="red")
```



This is kind of cool! On the first pass, I set the number of samples to 100, which is kind of low for this sort of thing. My histograms were chunky, the mean was wiggly, and the extinction probability was really small. So, I went back and reset the number of samples to 10000. Doing so really smooths out the histograms, mean and variance, and provides a more reasonable estimate of the extinction probability (which is still pretty darned low)

Despite my predisposition to look down my nose at simulations, I have to admit that simulating things is kind of fun. What else could we want to get out of this? Well, the current model only has variation in the individual births and deaths, but the averages are constant and independent of the population size. So how might we update the code to chage this? There are lots (and lots!) of ways to do this. Think of a few.

Let's look at those figures one more time, but this time with N on a log scale.



Lets think about the histograms first. Early in the series, the histogram of $\ln N$ is pretty skewed, but by 20 steps it is fairly symmetric, almost Gaussian. Why might this be? We'll come back to this in a little bit. The panels on the right indicate that on a log scale both the mean and the standard deviation increase linearly in time. Maybe we expected exponential growth for the mean, given our earlier analysis with constant birth and death rates, but the standard deviation is definitely new. Any ideas why these things both grow exponentially?

Some mathematical preliminaries

We are going to need a few things in our toolkit to make sense of stochastic models. Feel free to skip this section if you have a background in probability. The first things we need are the ideas of random variables and probability distributions. A discrete random variable is one that takes values in a well-defined set of values. For example we might have a variable that takes on values 0 or 1. Or, if we are rolling a die, the random variable might take values 1,2,3,4,5,6. The set might also be infinite, but 'countably' so, e.g. 0,1,2,3,…., on up to infinity. For discrete variables we describe the probability that they take on a particular value with a probability distribution (aka a 'probability mass function'). In the [0,1] example, the values would be p_0, p_1 and in the die rolling example, we would have $p_1, p_2, p_3, p_4, p_5, p_6$. Regardless of how we've defined the possible outcomes, the sum of the probabilities over all possibilities must be 1, which is usually written as $\sum_i p_i = 1$. We can think of the probability of a particular outcome as the number of times it occurs out of the total in a very large number of trials. That is, if n_i is the number out of N that take on the value i, then

$$p_i = \lim_{N\to\infty} \frac{n_i}{N}$$

The next things we'll need are the mean and variance. We have probably all seen the definitions for the sample mean, $\bar{x}$ and variance, s^2, given by

(1)
$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i s^2 = \frac{1}{N-1}\sum_{i=1}^{N}(x_i - \bar{x})^2$$

The analogs for discrete random variables are very similar. But rather than treat each observation separately, we can count up the number in each class and use these to calculate the mean. For example in the die rolling example, if we rolled the die N times, we could use (1) to calculate the mean directly, OR we could count up the number of times we got a 1, 2, etc, and use $n_1, n_2, ...$ to represent these counts. Since it doesnt matter what order we sum things up, the following are the same

$$\bar{x} = \frac{1}{N}\sum_{i=1}^{N} x_i = \frac{1}{N}\sum_{i=1}^{6} i n_i = \sum_{i=1}^{6} i\frac{n_i}{N}$$

And in the limit as $N \to \infty$, the mean converges on $E(x) = \sum_{i=1}^{6} i p_i$. Here, the $E(x)$ denotes the 'expected value' of the random variable which is what we get when we calculate the average using the probabilities. This usage of 'expectation' extends to functions as well, e.g. $E(f(x)) = \sum_i f(i)p_i$. We can do the same trick for the variance, which could be written as $E[(x - E(x)^2] = \sum_{i=1}^{6}(i - E(x))^2 p_i$. For more general discrete random variables, the indices differ (e.g. i=0,1 or i=0,1,2,...), but the forms remain the same.

When a random variable doesnt take on values from a nice tidy set, we refer to it as continuous. For example, if we said that X could be any real number between 0 and 1, how many possible values are there? Too many. For a continuous random variable there are infinite possible outcomes. So rather than thinking about probabilities associated with particular values, we think about the probability of being in some sub-interval, e.g. $X \in [0, 1/4]$. If we let $F(x)$ be the probability that our random variable, X is less than the specific value x, then the probability that $P[X \in (a, b)] = F(b) - F(a)$. If we take the limit as the interval becomes infintessimal, $P[X \in (a, a + da)] = F(a + da) - F(a)$, which is approximately $F'(a)da$ (since $F'(a) = \lim_{da\to 0}[F(a + da) - F(a)]/da$). And this is the quantity we usually work with, also known as the 'probability density function' (i.e. the probability per unit area) which is defined as $f(x) = F'(x)$. By analogy with the discrete random variables, the mean and variance are given by $E(x) = \int x f(x)dx$ and $Var(x) = \int (x - E(x))^2 f(x)dx$.

The most commonly used probability density function is the Normal or Gaussian density which is given by

(2)
$$f(x) = \frac{1}{\sqrt{2\pi v}} e^{-\frac{1}{2v}(x-m)^2}$$

where the mean m and variance v are the only two parameters in the model. It is called the 'normal' distribution because it is the most commonly observed and lots of other distributions are well approximated by it (under certain limits). Most importantly for our purposes is the fact that a sum of a large number of independent draws from *any* distribution with finite variance is approximately normally distributed. This result is the 'Central Limit Theorem.'

Closely related to the normal is the log-normal density. It is called this because something that is normally distributed on a log scale is log-normal on the original scale. Just like the normal, the density for the log-normal has only two parameters

(3)
$$f(x) = \frac{1}{x\sqrt{2\pi v}} e^{-\frac{1}{2v}(\ln x - m)^2}$$

which is defined only for $x > 0$ and where m and v are the mean and variance for $\ln x$. Since the normal distribution is the limiting distribution for the sum of a large number of random variables, the log-normal is the limiting distribution for products (of strictly positive random variables).

Handy facts

Here are a few handy facts about means and variances that will be important for understanding our simulation results. Let's say that X and Y are random variables, and a and b are constants. The mean of ax is $E(aX) = aE(X)$ which follows directly from the definition of the mean. (Try it for the die rolling example: $E(aX) = \sum_{i=1}^{6}(ai)p_i = a\sum_{i=1}^{6} ip_i$).

Also useful is the fact that

(4)
$$Var(x) = E[(x - E(x))^2] = E(x^2) - E(x)^2$$

This sometimes simplifies calculating the variance alot. For instance, it makes it easier to see that the variance of aX is

(5)
$$Var(aX) = a^2 Var(X)$$

.

Now let's say we are interested in some combination of X and Y. Let's define a new random variable Z as $Z = aX + bY$. Then the mean for Z is

(6)
$$E(Z) = E(aX + bY) = aE(X) + bE(Y)$$

.

In addition, if X and Y are independent, the variance for Z is

(7)
$$Var(Z) = Var(aX + bY) = a^2 Var(X) + b^2 Var(Y)$$

This extends immediately to more than 2 random variables. If we define another random variable W as $W = XY$, then if X and Y are independent, the mean for W is

(8)
$$E(W) = E(XY) = E(X)E(Y)$$

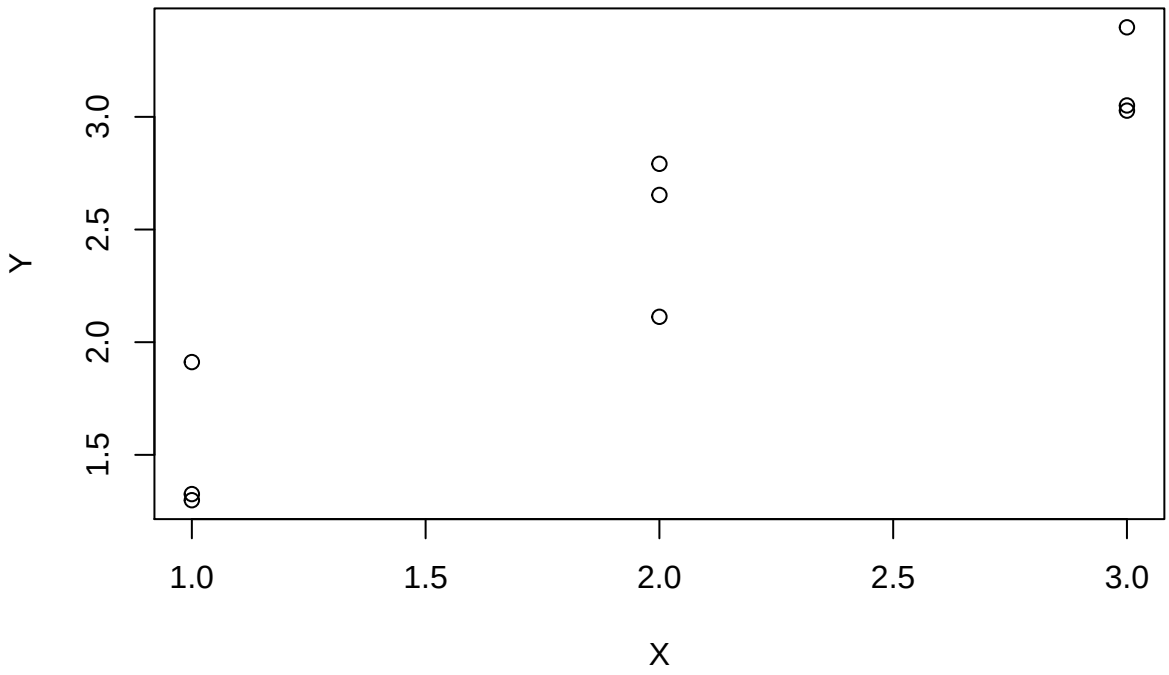
and the variance is

(9)
$$Var(W) = Var(XY) = Var(x)Var(y) + E(x)^2 Var(y) + Var(x)E(y)^2$$

Which is somewhat less pretty, but still pretty handy. The last two are useful when we have more than one random variable in which case it may be easier to find the mean or variance for one of the variables in subsets corresponding to the other random variable. I will give the formulas first, then a picture to make this more concrete

(10)
$$E(Y) = E_X[E(Y|X)]Var(Y) = E_X[Var(Y|X)] + Var_X[E(Y|X)]$$

In words, these say that the average for Y can be found by dividing into groups based on variable X, taking the mean within groups and then averaging these group means. The formula for the variance says that the variance in Y can be otbained by averaging the within-group variances and adding on the variance in the group means. A picture might help



Here, we have three values of X and 9 values for Y. We can find the average for Y directly, which gives us 2.4. OR, we could take the average of Y at each value of X, which gives {1.3, 2.5, 3.4} and then average these, which gives... 2.4!

Stochastic population dynamics

In our first models of population dynamics, we had bN births and dN deaths so that $N_{t+1} = (1+b-d)N_t = \lambda N_t$. So $\lambda = N_{t+1}/Nt$ which was constant for all t. Now, in keeping with our simulation, let's imagine that each individual makes *some* offspring, the exact number of which varies across individuals (let's say that individual i makes b_i) so that the total number of new individuals being born is $\sum_{i=1}^{N_t} b_i$. For deaths, let's give each individual a 'death indicator,' d_i which takes the value 1 if the individual is dead and 0 otherwise. Putting these things together, the total number of individuals next time is $N(t+1) = N(t) + \sum_{i=1}^{N_t} b_i - \sum_{i=1}^{N_t} d_i = \sum_{i=1}^{N_t}(b_i + s_i)$ where s_i is 1 if the individual survived and 0 otherwise. This is, of course, exactly how we calculated N_{t+1} in the simulation. But, what is λ now? To find that, we can just divide by N_t, to get

(11)

$$\lambda_t = \frac{N_{t+1}}{N_t} = \frac{N_t + \sum_{i=1}^{N_t} b_i - \sum_{i=1}^{N_t} d_i}{N_t} = 1 + \frac{1}{N_t}\sum_{i=1}^{N_t} b_i - \frac{1}{N_t}\sum_{i=1}^{N_t} d_i$$

The right hand side is just 1 plus the average number of births and deaths, which is pretty intuitive. But we've written λ with a subscript t because if we had a different sample of births/deaths or different weather, the average might be different.

Let's see if we can use our handy facts about means and variances to shed some light on the simulation results.

For the mean and variance of N_t we set up recursions as follows. For the mean, we note that $E(N_{t+1}) = E(\lambda_t N_t)$ and apply (8) to find that $E(N_{t+1}) = E(\lambda_t)E(N_t)$ which is the sort of thing we've been solving all quarter -

(12)
$$E(N_t) = E(\lambda)^t N_0 = E(b+s)^t N_0$$

The variance is a little harder. Since $N_{t+1} = \sum_{i=1}^{N_t}(b_i + s_i)$, and both $(b_i + s_i)$ and N_t are random, we need (10) to get the variance. To do so, we will use N_t as the 'grouping' variable. Plugging in, we get

$$Var(N_{t+1}) = E_{N_t}[Var(\sum_{i=1}^{N_t}(b_i + s_i)|N_t)] + Var_{N_t}[E(\sum_{i=1}^{N_t}(b_i + s_i)|N_t)]$$

Since all individuals are assumed to draw births and survivals from the same distribution, we can use (7) to simplify the variance in the first term to $Var(\sum_{i=1}^{N_t}(b_i + s_i)|N_t) = N_t Var(b+s)$ and (6) to simplify the mean in the second term to $E(\sum_{i=1}^{N_t}(b_i + s_i)|N_t) = N_t E(b+s)$. This gets us to

$$Var(N_{t+1}) = E_{N_t}[N_t Var(b+s)] + Var_{N_t}[N_t E(b+s)]$$

We can then use (5) and (6) to simplify this down to

$$Var(N_{t+1}) = E(N_t)Var(b+s) + Var(N_t)E(b+s)^2$$

This recursion is solvable by direct iteration and has the solution

$$Var(N_{t+1}) = N_0 Var(b+s)\frac{E(b+s)^{2t} - E(b+s)^t}{E(b+s)^2 - E(b+s)}$$

Assuming that $E(b+s) > 1$, this is approximately

(13)
$$Var(N_{t+1}) = N_0 Var(b+s)\frac{E(b+s)^{2t}}{E(b+s)^2 - E(b+s)}$$

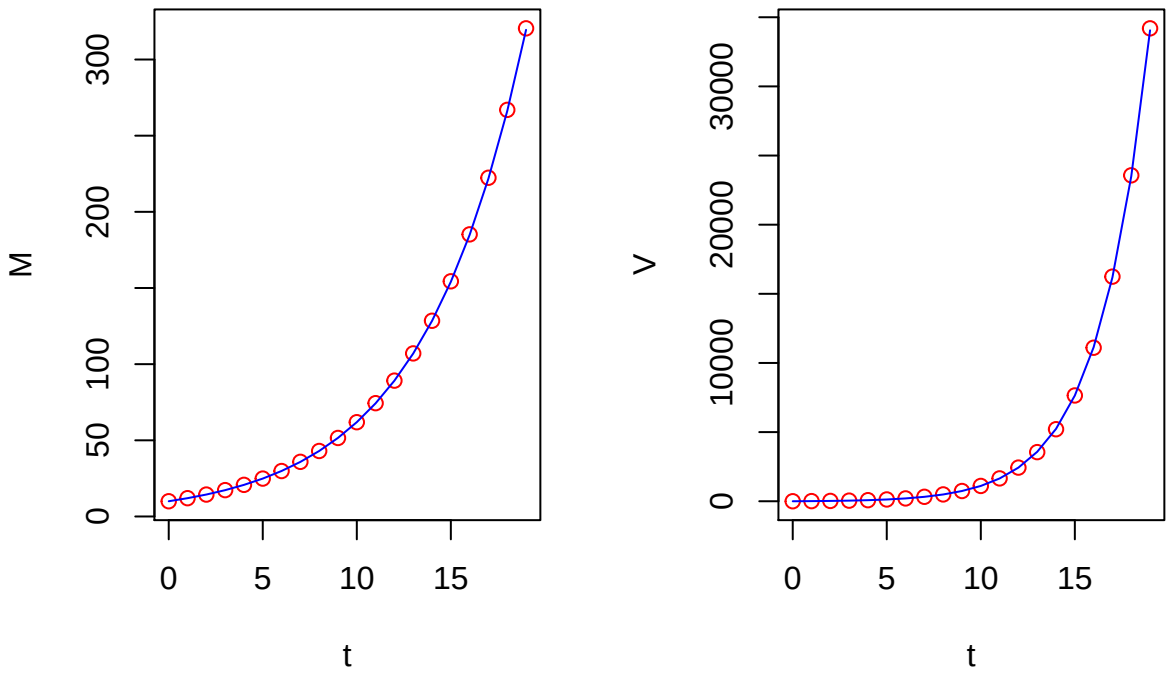
From this we can see that - as in the simulations - the mean and variance both grow exponentially. Moreover, the variance grows roughly 2x as fast as the mean. The second (and far more important) thing to take from this is that the specific shapes of the distributions dont really affect the mean and the variance in population size. If these are the only things we want to know, then all we need are the mean and variance in the births and survivals. In addition, the Central Limit Theorem tells us that sums of independent, identically distributed random variables are approximately Normally distributed. Which suggests that for big enough t, $\ln N_t$ should be approximately normally distributed. Similarly, N_t should be approximately log-normal. Since the probability density for these distributions is determined solely by the values for the mean m and variance v - and we have already calculated these, we can just plug them in to make predictions about the distribution of future population size.

Before we continue with stochastic population models, I thought it would be fun to see whether these hard-earned formulae are worth anything by comparing them to our simulation results

Application to the simulation

To apply our expressions for the means and variances to the simulations, we need to calculate a few things from our simulation set up. Specifically we need $E(b+s)$, $V(b+s)$ which I leave to you to figure out. Here's a plot Of the mean (left) and variance (right) for the simulation (dots) and the theory from (12) and (13) (lines)

```
nsamples=10000 # the total number of sample paths we will generate
T=20 #the total number of time steps
N0=10 #initial population size
bmax=2 #maximum number of offspring
pmort=0.8 #probability of dying

b<-c(0:bmax) #make list of possible births
N=matrix(data=0,T,nsamples) #array for population sizes
N[1,]=N0

#outer loop
for (j in 1:nsamples){
#inner loop
for (i in 1:(T-1)) {
  births<-sum(sample(b,N[i,j],replace=T))
  deaths<-sum(runif(N[i,j],0,1)<pmort)
  N[i+1,j]<-N[i,j]-sum(deaths)+sum(births)

}}

#summary stats from the simulation
M<-apply(N,1,mean)
V<-apply(N,1,var)
S<-apply(N>0,1,mean)

#theoretical predictions
Eb=sum(b)/(bmax+1)
Vb=sum(b^2)/(bmax+1)-Eb^2
Es = 1-pmort
Vs=pmort*(1-pmort)

Ebs=Eb+Es
Vbs=Vb+Vs

t<-c(0:(T-1))
EN=N0*(Ebs)^t
VN=N0*Vbs*(Ebs^(2*t)-Ebs^t)/(Ebs^2-Ebs)

par(mfrow=c(1,2))
plot(t,M,col="red")
lines(t,EN,col="blue")

plot(t,V,col="red")
lines(t,VN,col="blue")
```



Demographic versus Environmental Stochasticity

One distinction that is commonly made in the literature is between 'demographic' and 'environmental' stochasticity. When we think about year-to-year variation in λ_t, let's imagine that we can separate the parts that are due to the envrionment (common to everyone) and to individual variation. That is, let's write $b_i = \bar{b}_t + \delta_i$ and $\mu_i = \bar{\mu}_t + \gamma_i$. Plugging this into (11) we get

(14)

$$\lambda_t = \bar{b}_t - \bar{\mu}_t + \frac{1}{N_t}\sum_{i=1}^{N_t}(\delta_i - \gamma_i)$$

Now, we know from our previous work on population dynamics that $N_t = \Pi_i \lambda_i N_0$ and the long run growth rate of the population is given by the geometric mean of λ which is - in the notation introduced above - $exp[E(\ln \lambda)]$. If we pretend for a moment that $\lambda_t = \bar{\lambda} + \epsilon_t$, we can approximate $E(\ln \lambda_t)$ as

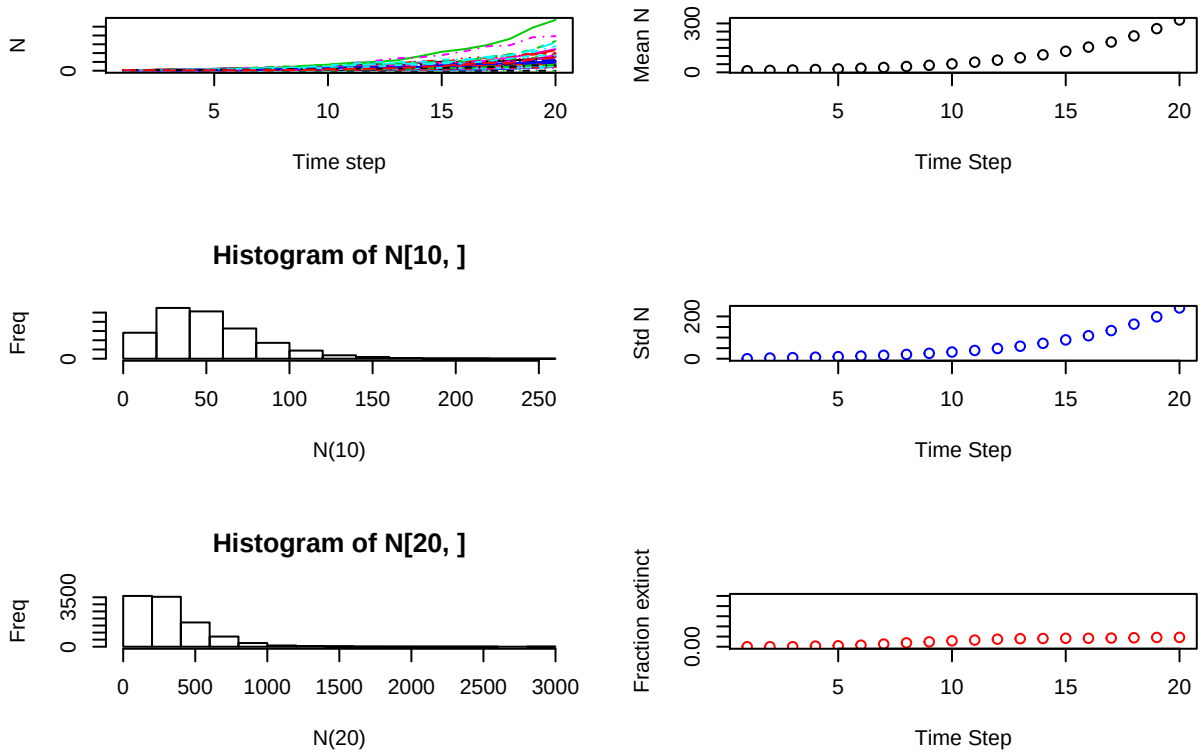
(15)

$$E(\ln \lambda_t) = E\left(\ln\left[\bar{\lambda} + \epsilon_t\right]\right) \approx \log \bar{\lambda} + \frac{1}{\bar{\lambda}}E(\epsilon_t) - \frac{1}{2}\frac{1}{\bar{\lambda}^2}E(\epsilon_t^2) = \log \bar{\lambda} - \frac{1}{2}\frac{Var(\epsilon_t)}{\bar{\lambda}^2}$$

That is, average value of $\ln \lambda_t$ is approximately the log of the average λ_t discounted by a factor that increases with the variance and decreases with the square of the mean.

For now, let's say that the environment changes the *probabilities* of births and deaths, rather than the range of values they can take on. For instance, we might say that μ isnt constant, but takes on random values between μ_{low} and μ_{hi} Let's see what this does.

```
#model with 'environmental' noise in mortality
nsamples=10000 # the total number of sample paths we will generate
T=20 #the total number of time steps
N0=10 #initial population size
bmax=2 #maximum number of offspring
pmort_lo=0.6 #lower bound on probability of dying
pmort_hi=1 #upper bound on probability of dying

b<-c(0:bmax) #make list of possible births
N=matrix(data=0,T,nsamples) #array for population sizes
N[1,]=N0

#outer loop
for (j in 1:nsamples){
#inner loop
for (i in 1:(T-1)) {
  pmort<-runif(1,pmort_lo,pmort_hi)
  births<-sum(sample(b,N[i,j],replace=T))
  deaths<-sum(runif(N[i,j],0,1)<pmort)
  N[i+1,j]<-N[i,j]-sum(deaths)+sum(births)

}}

Me<-apply(N,1,mean)
Ve<-apply(N,1,var)
Se<-apply(N>0,1,mean)

par(mfrow=c(3,2))
matplot(c(1:T),N[,1:50], xlab="Time step",ylab="N",type="l")
plot(c(1:T),Me,xlab="Time Step",ylab="Mean N")
hist(N[10,],xlab="N(10)",ylab="Freq")
plot(c(1:T),sqrt(Ve),xlab="Time Step",ylab="Std N",col="blue")
hist(N[20,],xlab="N(20)",ylab="Freq")
plot(c(1:T),1-Se,xlab="Time Step",ylab="Fraction extinct",ylim=c(0,.05),col="red")
```



We picked the range for our random draws of μ_t so that the mean was the same, but it looks like the extinction risk has gotten bigger. And maybe the distribution for N_20 is a bit more spread out. Otherwise, our general observations with constant μ still hold.

So far all of these models have assumed that births and deaths happen independent of population size. To allow for density density dependence, we could modify the model in many ways, e.g. changing the range of possible litter sizes, etc. But for now, what we will do is have density modify the birth *probabilities* keeping the possibilities the same. Again there are lots of ways to do this, and many of them will be smarter than what I have below, but this is an easy way to get going.

Let's say that instead of equally likely, we make $P(littersize = b)$ a function of n.

$$P(b|n) = \begin{cases} 2/3g(n) & b = 0 \\ 1/3 & b = 1 \\ 2/3(1 - g(n)) & b = 2 \end{cases}$$

where $g(n)$ is some function that starts near 0 when $n = 0$ and goes to 1 as n gets big. Let's just use $g(n) = n/(2 * K)$ for simplicity. This makes our simulation model a stochastic analog of the discrete logistic model (can you see why?).

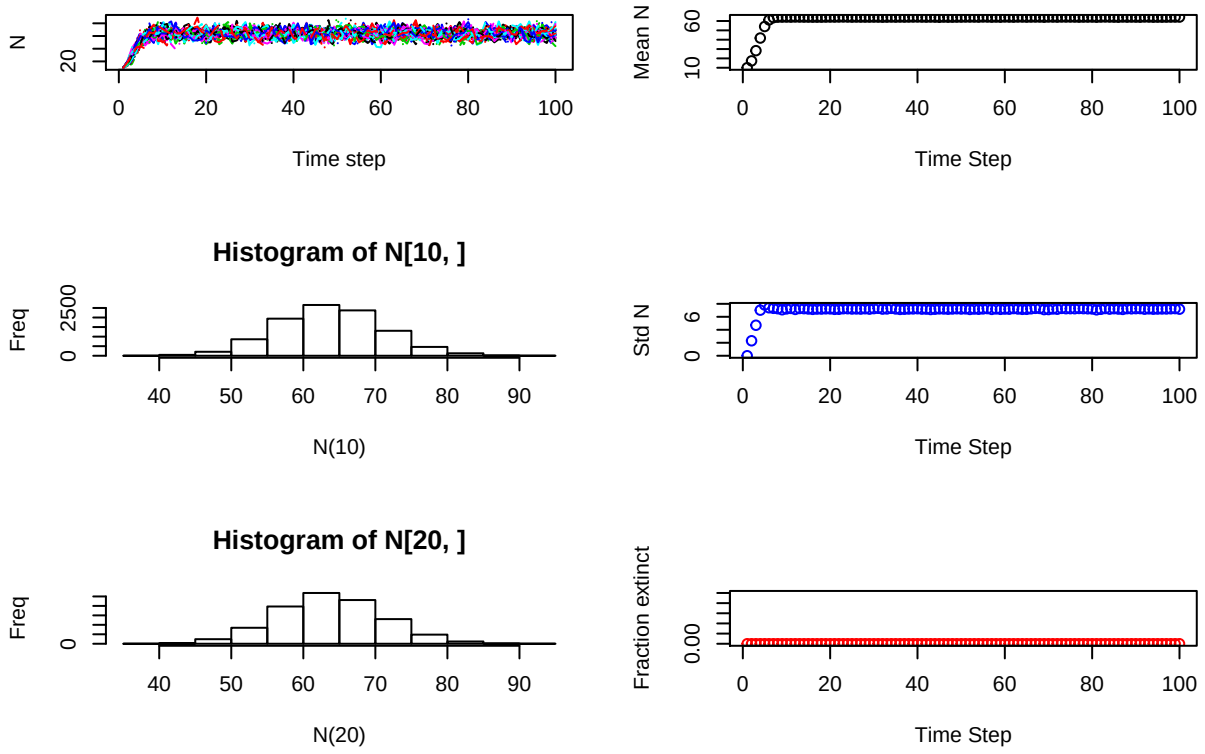
The main difference here is that, instead of increasing without bound, the sample paths now hover near some constant value. And although the population size is still random, the distribution of population sizes stays constant. That is, introducing density dependence has allowed us to obtain a *stationary distribution* of population sizes.

There are some nice analytical tricks we can do with these, too. But the nonlinearity makes them a bit harder. The easiest things to do are to approximate the stationary mean and variance. For the mean, the trick is to find the population size for which the average births and deaths are equal. In this model, that

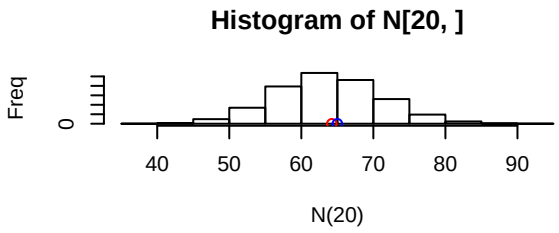
means that $E(b+s)=1$. Here $E(b)=0*[2/3g(n)]+1*[1/3]+2*[2/3(1-g(n))]=[5-4g(n)]/3$. Setting this equal to the death rate and solving for n, we get $n_{eq}\approx 2K(5-3\mu)/4$. The solution for the variance involves a linear approximation of the model near steady state. But it's frankly too much to type here - but it is in the code below as 'Vs'

```
#with density dependent births
nsamples=10000 # the total number of sample paths we will generate
T=100 #the total number of time steps
N0=10 #initial population size
bmax=2 #maximum number of offspring
pmort=0.8 #lower bound on probability of dying
K=50

b<-c(0:bmax) #make list of possible births
pi_n<-function(n) min(1,n/(2*K))
dpi<-function(n) 1/(2*K)
d2pi<-function(n) 0

birth_prob<-function(n) c(2/3*pi_n(n), 1/3,2/3*(1-pi_n(n)))
dbirth_prob<-function(n) c(2/3*dpi(n), 0,-2/3*dpi(n))
d2birth_prob<-function(n) c(2/3*d2pi(n), 0,-2/3*d2pi(n))

mb<-function(n) sum(b*birth_prob(n))
dmb<-function(n) sum(b*dbirth_prob(n))
d2mb<-function(n) sum(b*d2birth_prob(n))
vb<-function(n) sum(b^2*birth_prob(n))-mb(n)^2
dvb<-function(n) sum(b^2*dbirth_prob(n))-2*mb(n)*dmb(n)
d2vb<-function(n) sum(b^2*d2birth_prob(n))-2*dmb(n)^2-2*mb(n)*d2mb(n)

vs<-pmort*(1-pmort)
mg<-function(n) n*(mb(n)+1-pmort)
vg<-function(n) vb(n)+vs

dmg<-function(n) n*dmb(n)+mb(n)
dvg<-function(n) dvb(n)
d2vg<-function(n) d2vb(n)

sdg<-function(n) (n*vg(n))^(1/2)
d_sdg<-function(n) (1/2)*(n*vg(n))^(-1/2)*(n*dvg(n)+vg(n))
d2_sdg<-function(n) -(1/4)*(n*vg(n))^(-3/2)*(n*dvg(n)+vg(n))^2+(1/2)*(n*vg(n))^(-1/2)*(2*dvg(n)+n*d2vg(n


Ns=2*K*(5-3*pmort)/4  #steady state for pi_n = N/2K
den<-dmg(Ns)^2+d_sdg(Ns)^2+sdg(Ns)*d2_sdg(Ns)
Vs=sdg(Ns)^2/(1-den)


N=matrix(data=0,T,nsamples) #array for population sizes
N[1,]=N0

#outer loop
for (j in 1:nsamples){
#inner loop
for (i in 1:(T-1)) {
```

```
  births<-sum(sample(b,N[i,j],replace=T,prob=birth_prob(N[i,j])))
  deaths<-sum(runif(N[i,j],0,1)<pmort)
  N[i+1,j]<-N[i,j]-sum(deaths)+sum(births)

}}

M<-apply(N,1,mean)
V<-apply(N,1,var)
S<-apply(N>0,1,mean)

par(mfrow=c(3,2))
matplot(c(1:T),N[,1:50], xlab="Time step",ylab="N",type="l")
plot(c(1:T),M,xlab="Time Step",ylab="Mean N")
hist(N[10,],xlab="N(10)",ylab="Freq")
plot(c(1:T),sqrt(V),xlab="Time Step",ylab="Std N",col="blue")
hist(N[20,],xlab="N(20)",ylab="Freq")
plot(c(1:T),1-S,xlab="Time Step",ylab="Fraction extinct",ylim=c(0,.05),col="red")
```




```
hist(N[20,],xlab="N(20)",ylab="Freq")
points(M[20],0,col="red")
points(Ns,0,col="blue")
```

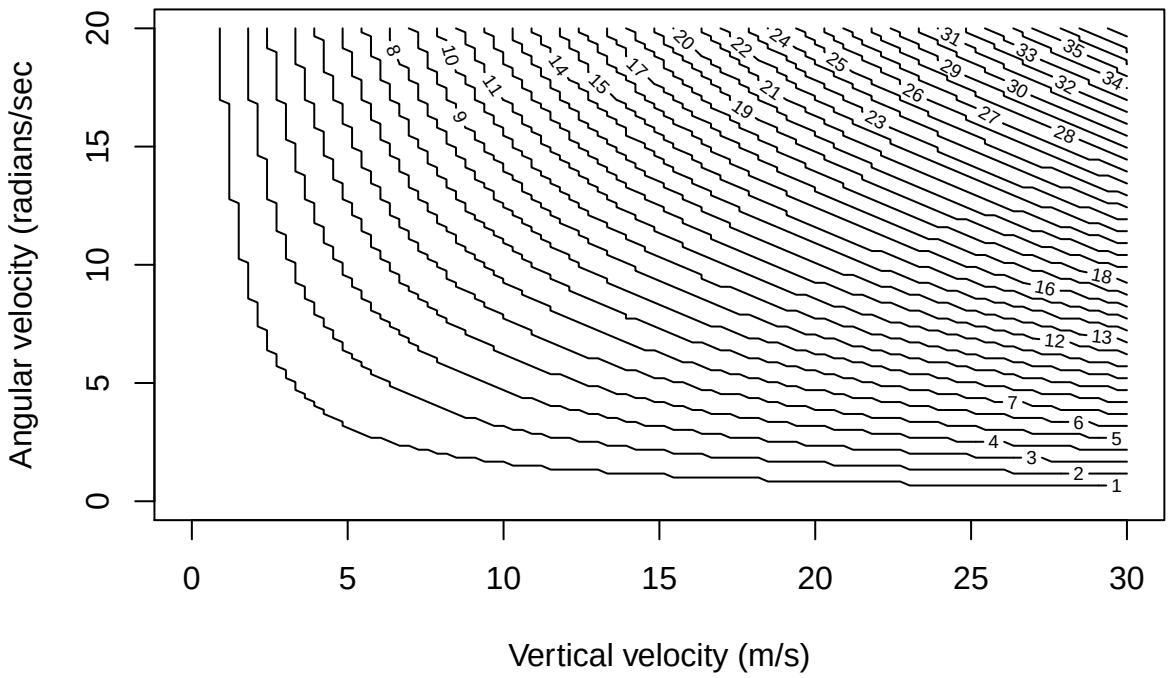


There are, of course, lots more things we could do with these models. I hope this was enough of a taste of what is possible to get you going.

Why treat things as random?

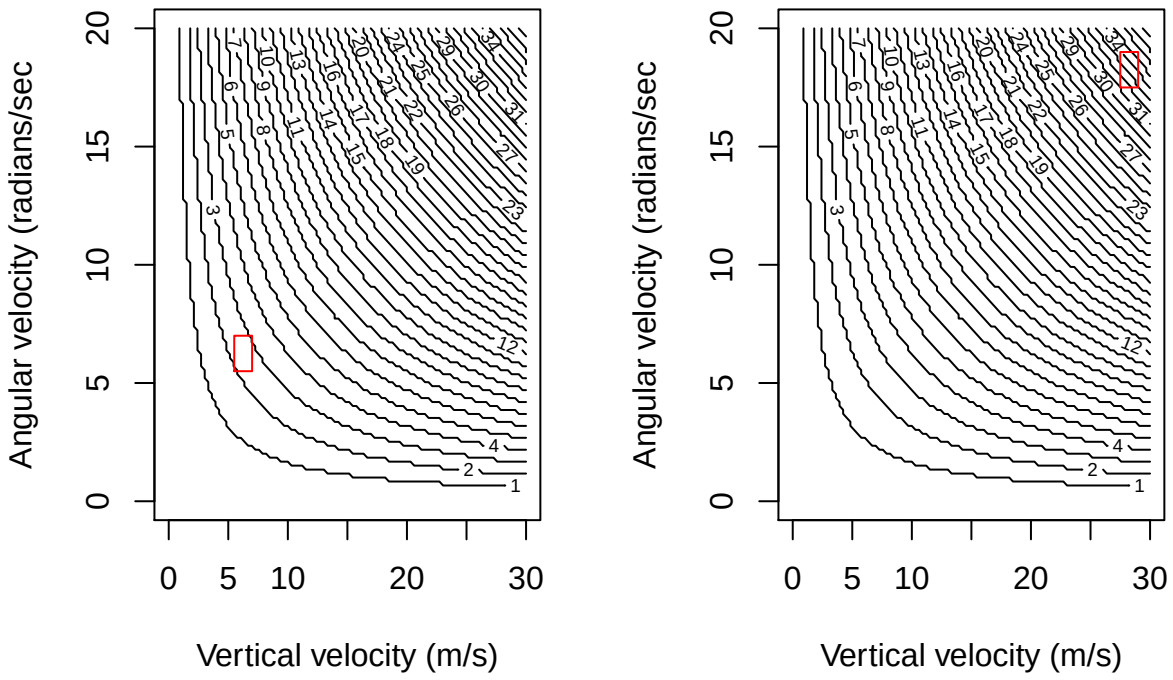
Let's think a little bit harder than usual about what it means for something to be 'random.' Take, for instance, the classic example of a random event that we were all taught in grade school, tossing a coin. We start with the coin, flip it into the air so that it rotates reasably fast, and check which side is facing up when the coin eventually comes to rest. Our usual assumption is that the coin is 'fair', i.e. that both sides are equally likely to face up. But what makes this random? The physics of the problem are actually relatively simple - the coin leaves from a fixed height, x_0 (m) with a given vertical velocity, v_0 (m/s) and gravity draws it back to earth with constant acceleration $-g$ (m/s^2). Since the height at time t is given by $x(t) = x_0 + v_0 t - 1/2gt^2$, the time aloft (i.e. before the height is again s_0) is set by $t = 2v_0/g$. If we neglect air resistance, we can assume that the angular velocity stays constant through time (at rate ν) so that the angle at time t is $\theta(t) = \theta_0 + \nu t$. Let's say that we started it horizontally with heads up and we'll call this $\theta_0 = 0$. So, if it lands at time $t = 2v_0/g$, the angle will be $\theta = 2\nu v_0/g$. So, where did we ever get the idea that this was random??!

At this point we could try to allow for bouncing etc, but that's really hard and totally unncessary. Let's just pretend that if it lands with θ between 0 and π, we'll call it 'heads' and if it lands with $\theta \in [\pi, 2\pi]$ we'll call it 'tails'. Of course, there's no reason for $\theta = 2\nu v_0/g$ to be in $[0, 2\pi]$, so we need to be a little more careful. If we kept on going into the interval $\theta \in [2\pi, 3\pi]$ we'd have 'heads' again and for $\theta \in [3\pi, 4\pi]$ we'd have 'tails.' We can keep going like this, so really what we want to say is that 'heads' corresponds to $\theta \in [2n\pi, (2n+1)\pi]$ where n is any integer, starting from 0. Similarly, 'tails' means that $\theta \in [(2n+1)\pi, 2(n+1)\pi]$. So, let's make a plot of the outcome of our coin toss, as a function of our initial vertical and angular velocities.



The countours here indicate a switch from heads to tails and vice-versa, starting from heads. That is between the axes and the coutnour labeled "1" we have heads, between 1 and 2, we have tails, etc. Down in the bottom left, where both the rotation and vertical velocity are low, there are big gaps between contours. This agrees with our experience on the playground as kids where if we tossed a coin slow enough we could predict with good accuracy which side it would land on. However, as we increase the vertical and angular velocities (i.e. heading toward the upper right side of the plot), the spacing between successive contours becomes very fine, such that it is hard to know which size will come up. Again, this agrees with our practical exuperience of coin tossing as children.

So, why do we treat coin tosses as random? If we know the initial and angular velocities with infinite precision, there is no uncertainty. But in practice, our precision is always finite. As a consequence we can alway increase the velocity to a point where the contours are finer than our precision will allow us to differentiate, effectively making it impossible to tell which side will come out 'up'. You can see this effect in the figure below, where the red boxes represent the same degree of uncertainty around two different sets of conditions. For the one at the bottom right, we can be nearly sure that, despite our uncertainty, we will get heads. But in the upper right box, the same degree of uncertainty means we have almost equal chances of heads and tails.



This puts me in mind of times insisting that my playground pals "flip the coin fast enough so it's random." But, really, the 'randomness' in the coin toss - our archtypal example of a random experiment - comes entirely from our lack of information about the initial conditions.

Let's take a step back. At this point we've said that we might as well treat the outcome of a coin toss as random because we can't know the initial conditions precisely enough. . . and this is for a situation where the physics is absurdly simple! Imagine what happens when the dynamics are more complex. The contours stop being nice smooth curves and start being fractals, so that the areas where we 'know' the outcome become much more fine grained.