Mathematical Modeling of Infectious Diseases, UCSF

Lecture 5

Version 1.0—modified 31 Jan 2015

Negative binomial

Let's quickly review some things about the negative binomial and about generating functions.

We found last time that the negative binomial could be represented as a sum of geometrics. We wrote the generating function as

$$g^*(u) = \frac{p^k}{(1-u(1-p))^k}$$

And we know that if Y has the negative binomial with parameters p and k, $EY = g^{*\prime}(1)$. Let's do this using a computer algebra package called `Sage` (freely available from `sagemath.org`): Set it up:

```
sage: u=var('u')
sage: p=var('p')
sage: k=var('k')
```

Now:

```
sage: diff( p^k/(1-(1-p)*u)^k, u).subs(u=1).simplify()
-(p - 1)*k/p
```

which is just $(1-p)k/p$, from last time.

Let's find a few of the probabilities using Sage. We know

$$g^*(u) = u^0 P(Y=0) + u^1 P(Y=1) + u^2 P(Y=2) + \cdots$$

so $g^*(0) = P(Y=0)$. Well, $P(Y=0)$ is not too bad; put $u=0$ in the generating function and you get $P(Y=0) = p^k$.

To get $P(Y=1)$ we need the coefficient of u in the expansion. If we were to differentiate this once, we'd have

$$g^{*\prime}(u) = u^0 P(Y=1) + 2u^1 P(Y=2) + 3u^2 P(Y=3) + \cdots \tag{1}$$

Substituting $u=0$ in blows all that stuff away that is to the right of the first `+` sign, so we have $P(Y=1) = g^{*\prime}(0)$. The derivative evaluated AT ZERO gives us the chance of $Y=1$. Help, `Sage`:

```
sage: diff( p^k/(1-(1-p)*u)^k, u).subs(u=0).simplify()
-(p - 1)*p^k*k
```

So, $P(Y = 1) = k(1 - p)p^k$.

How about another one?

$$g^{*\prime\prime}(u) = 2u^0 P(Y = 2) + 3 \cdot 2u^1 P(Y = 3) + 4 \cdot 3u^2 P(Y = 4) + \cdot \tag{2}$$

Substituting $u = 0$ again blows a lot of noise away, and we get

$$P(Y = 2) = \frac{1}{2} g^{*\prime\prime}(0).$$

Play with this a while and you can convince yourself

$$P(Y = i) = \frac{1}{i!} g^{*(i)}(0).$$

where I mean take the i-th derivative, substitute 0 in, and divide by $i!$ (i factorial). We just derived the Taylor series expansion of the generating function. Let's at least do one more with the negative binomial:

```
sage: (diff( p^k/(1-(1-p)*u)^k, u, 2).subs(u=0)/2).simplify().factor()
1/2*(p - 1)^2*(k + 1)*p^k*k
```

So it is $\frac{1}{2}k(1 - p)^2(1 + k)p^k$.

Before we go much further: there are some important things to look at very closely. The first is that ANY probability generating function—no matter what it is—is always increasing; the slope is NEVER negative. You can see this from Equation 1; the probabilities are always positive and the first derivative is a sum of positive terms. The only partial exception is if every $P(Y = i) = 0$ for $i > 0$ and $P(Y = 0) = 1$; then the first derivative is zero; you have the dreary random variable that is 0 with probability 1.

The next is that the second derivative is never negative. Typically, it is positive; look at Equation . If any of the probabilities of $Y = 2$ or above are positive, the second derivative is always positive and the function is *convex*. Only if you have Bernoulli trials—random variables supported on $\{0, 1\}$, do you have a vanishing second derivative.

Exercise 0. What is the generating function of a Bernoulli trial with success probability p? Use the generating function to find its mean and variance. If you add up fixed N of them (independent, identical), what is the generating function? ∎

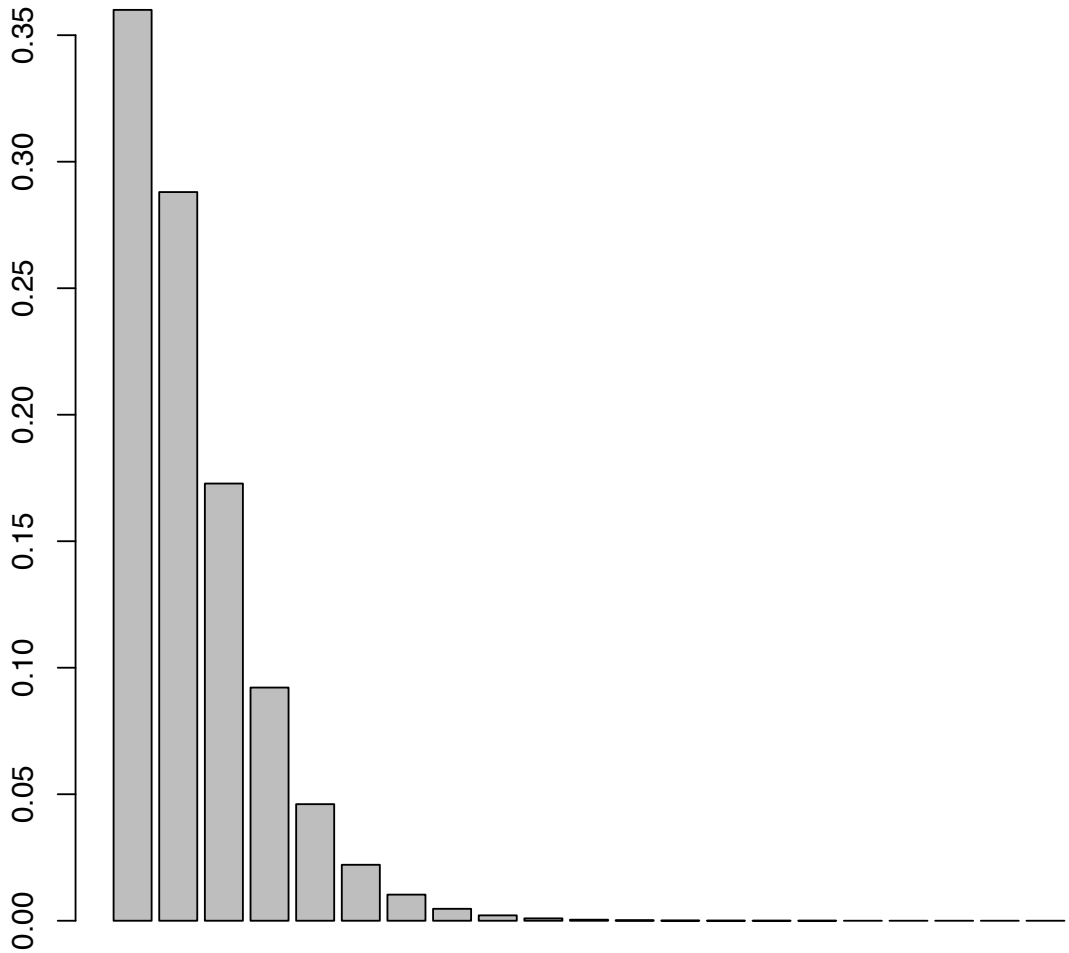
One of the things you need to know to deal with the literature and software out there is that different people parameterize the standard distributions in different ways. There's no fix for it; we just have to live with it. The program R has functions for the negative binomial. The function `dnbinom` generates the probabilities for the negative binomial, and the help file for it tells us:

```
dnbinom(x, size, prob, mu, log = FALSE)
```

The negative binomial distribution with `size` $= n$ and `prob` $= p$ ... represents the number of failures which occur in a sequence of Bernoulli trials before a target number of successes is reached. The mean is $n(1-p)/p$...

It's clear that their n is our k, and their p is our p. Careful—sometimes their p might be our $1-p$ and so on. Let's look at one:

```
barplot(dnbinom(0:20,size=2,prob=0.6))
```

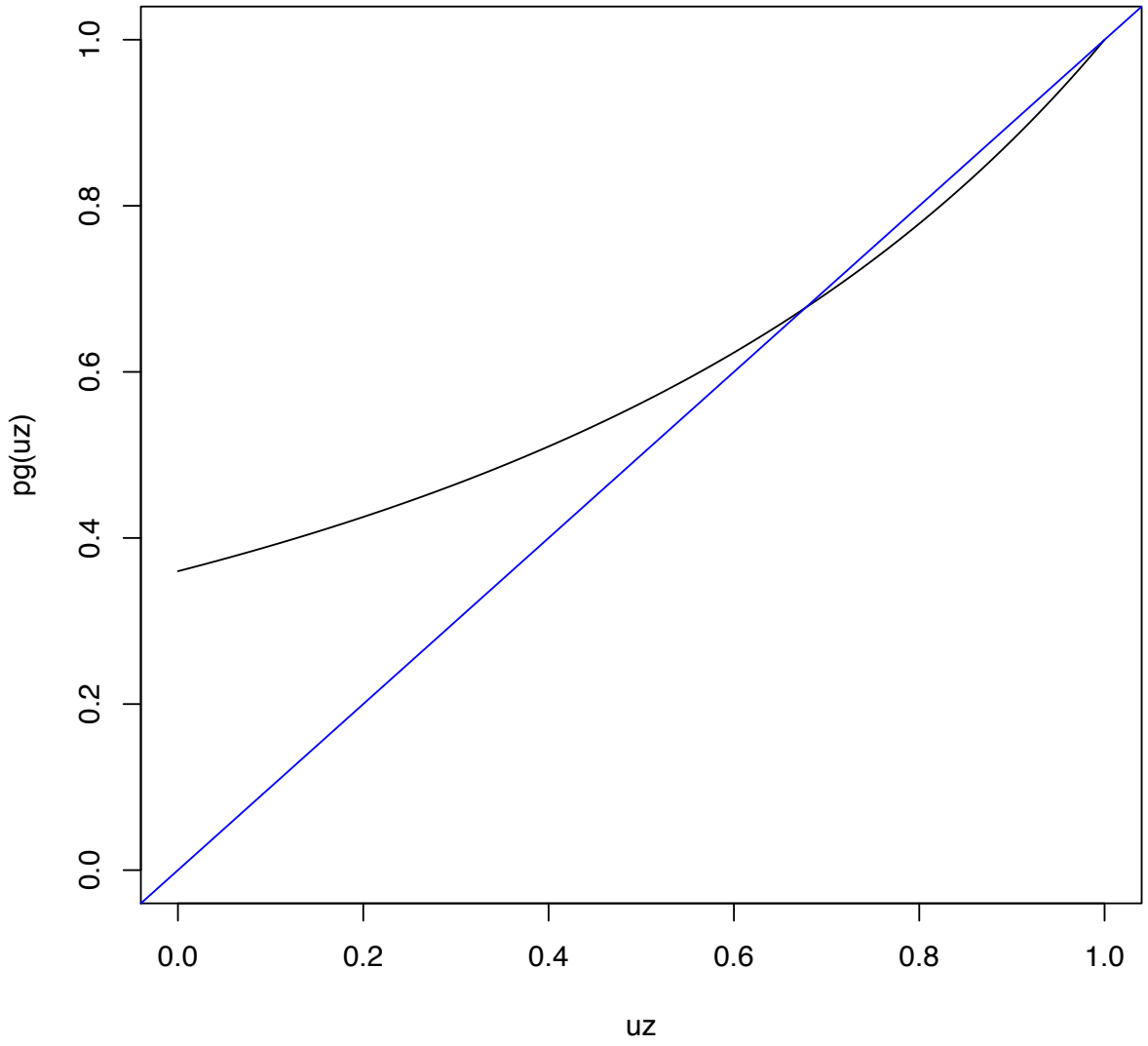


Here, $k = 2$ and $p = 0.6$, and we went out to 20. What is the mean for this one? It is $2*(1-0.6)/0.6 = 4/3$.

Exercise 1. Use R's function `dnbinom` to plot some negative binomial plots; pick $k = 0.1$, $k = 1$, $k = 2$, $k = 10$; try $p = 0.1$, $p = 0.5$, $p = 0.8$; go out further than 20 if you have to (plot out far enough that the tiny probabilities don't show up any more on barplot. Play with it; turn 2 or 3 plots in. ∎

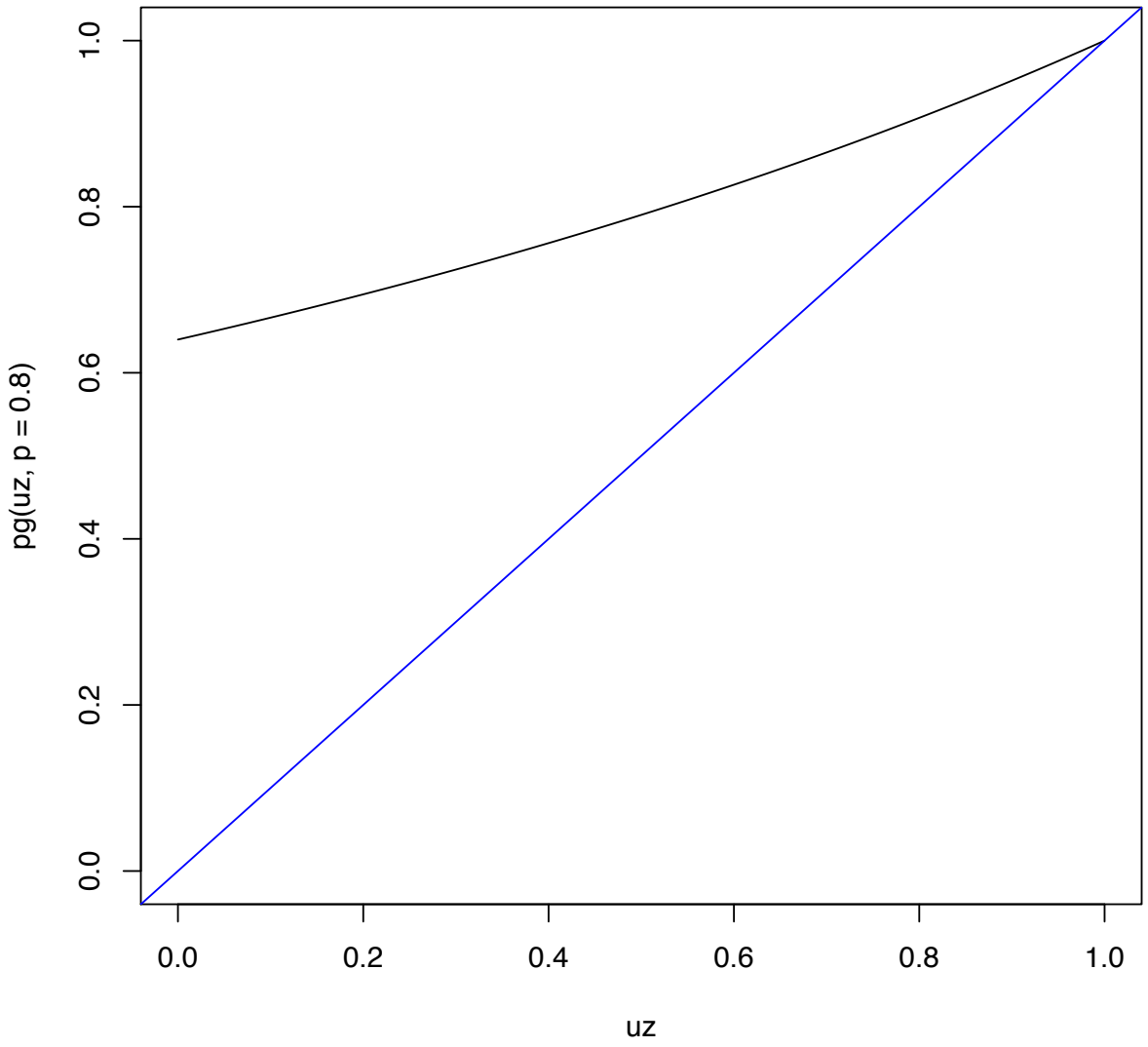
Now, let's plot the probability generating function for it (and add a line for where the value on the x-axis equals the value on the y-axis in blue):

```
uz <- seq(0,1,by=1/1024)
pg <- function(u,p=0.6,k=2){p^k/((1-u*(1-p))^k)}
plot(uz,pg(uz),type="l",xlim=c(0,1),ylim=c(0,1))
abline(a=0,b=1,col="blue")
```



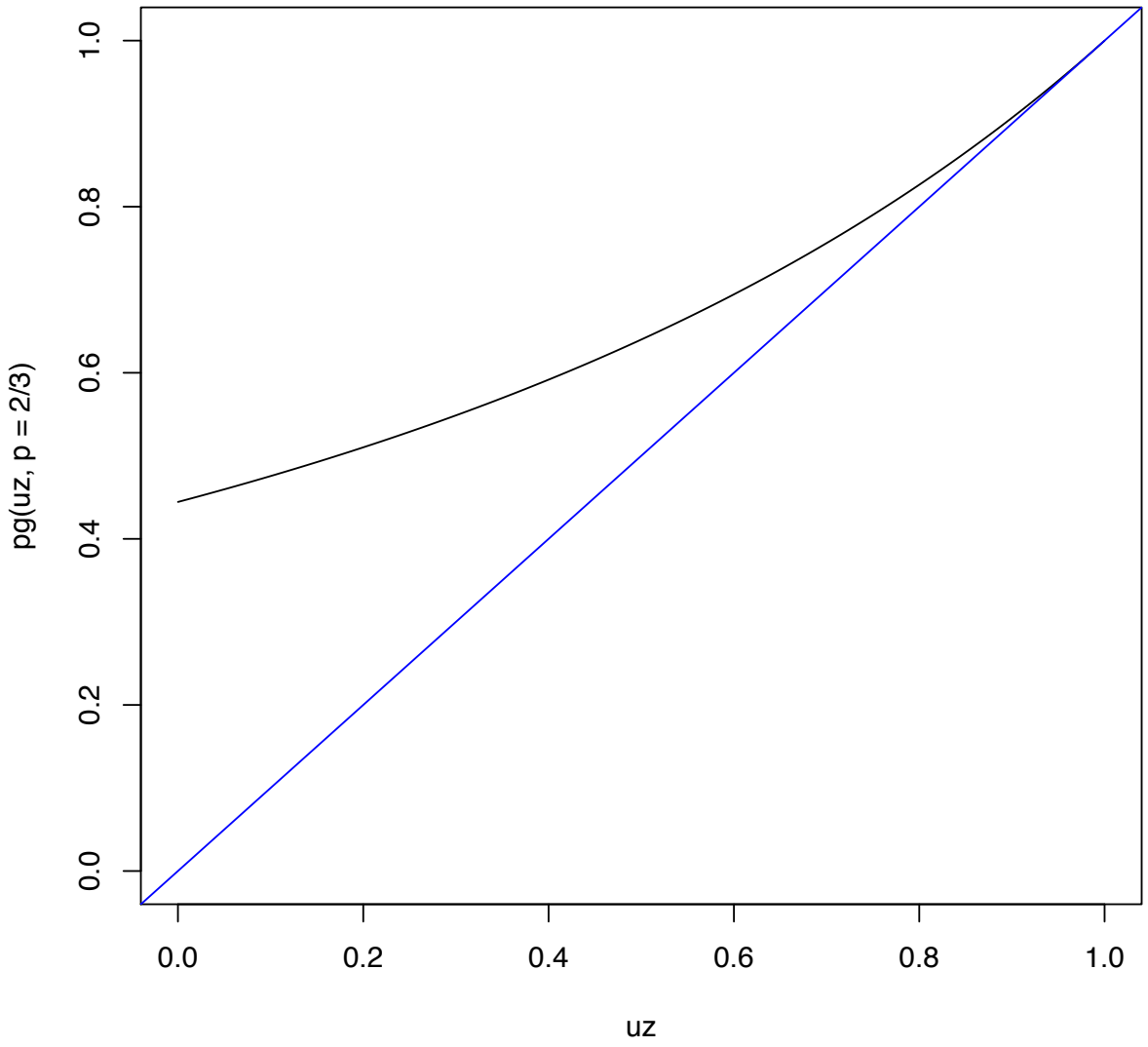
Now, let's keep $k = 2$ but make $p = 0.8$. The mean now is $2 * 0.2/0.8 = 1/2$. Here is the generating function:

```r
plot(uz,pg(uz,p=0.8),type="l",xlim=c(0,1),ylim=c(0,1))
abline(a=0,b=1,col="blue")
```



And what about $k = 2$ and $p = 2/3$? The mean is $2 * (1/3)/(2/3) = 1$:

```
plot(uz,pg(uz,p=2/3),type="l",xlim=c(0,1),ylim=c(0,1))
abline(a=0,b=1,col="blue")
```



Look at that—the slope at $u = 1$ is exactly 1, and the curve is just tangent to the 45 degree line right there. Of course the slope is 1 at $u = 1$ since the mean is 1; the slope at $u = 1$ is the mean.

Detour on Convexity

Geometrically, a function $f(u)$ is *convex* on an interval $[u_0, u_1]$ if, and only if, for every $u_0 \leq a < b \leq u_1$, the segment joining the point $(a, f(a))$ on the left with $(b, f(b))$ on the right lies over the curve. If the straight line lies over the curve, the curve has to start out increasing more slowly and then get steeper and steeper to catch up. The slope of the curve has to be increasing.

Imagine we have a convex function f, and we have a random variable X. The mean of X is EX, whereever that is. Imagine we write $f(X)$. If a function is convex, we know it lies below every one of its secants. But it lies above all its tangent lines. Let's pick the tangent line that crosses the function right at the mean. Let m be the slope right there at the mean, and use the point-slope form: $f(EX)+m(X-EX)$ is a line that touches the function $f(X)$ right at where X's mean is. By convexity, $f(X) > f(EX) + m(X - EX)$. Now, take the expectation of both sides: $E[f(X)] > E[f(EX)] + mE[X - EX]$, which gives us $E[f(X)] > f(EX) + m \cdot 0$, so that $E[f(X)] > f(E[X])$. If f is convex, the expected transform beats the transformed expectation. This comes up all over the place, and is known as **Jensen's Inequality**.

Note $f(x) = x^2$ is convex because the second derivative is 2, which is positive. So $E[X^2] > (EX)^2$, which we already knew. What if we consider $f(x) = 1/x$, on $0 < x < \infty$ for example? (No zero.) Then $f'(x) = (-1)x^{-2}$ and $f''(x) = -(-2)x^{-3} > 0$. So now we know $E[1/X] > 1/EX$ for $X > 0$. Or this: let $f(x) = -\log(x)$ on $x > 0$. Here, $f''(u) = 1/u^2 > 0$. So minus the log is convex, so $E[-\log(X)] > -\log(EX)$, or $\log(EX) > E[\log(X)]$.

The Galton-Watson Process

We will now analyze the Galton-Watson process as a model of a simple epidemic. The naturalness of the branching process approach for epidemic analysis has been known for a long time, and was emphasized by Bartoszynski long ago.

The basic model is simplicity itself. We will let X_0 be the number of index cases, and we will usually make this equal to 1. The index cases each independently give rise to secondary cases. The number of secondary cases caused by a single case follows a distribution with probability generating function $g^*(u)$. The next generation consists of all the cases produced by the previous generation.

For example, imagine a single case of measles introduced into some county in California. Suppose that this person causes three secondary cases. The initial generation has 1 case (the index case). The second generation consists of three cases. Now, suppose the first case in the second generation causes 1 case themselves before recovering. Suppose the second case produces no new cases, and finally, suppose the third case produces two cases. The third generation then has 1 + 0 + 2 cases. And so on it would go.

We are modeling this as a random process, so the number of cases in each generation will be treated as a random variable. Even if we insist that X_0 is always one, it is still random. It is a kind of degenerate special case of randomness where it takes the value 1 with probability 1. The generating function of this is

$$g_0(u) = u^0 P(X_0 = 0) + u^1 P(X_1 = 1) + u^2 P(X_2 = 2) + \cdots$$

which is

$$g_0(u) = 0 + u + 0 + \cdots = u.$$

This might seem kind of cheap, but it works.

Remember the expected value of a random variable is $E[X] = g'(1)$? Let's try it with X_0 having $g_0(u) = u$.

$$g_0'(u) = \frac{d}{du} u = 1.$$

This is just one all the time, no matter what u is, so for $u = 1$ it is also 1. This tells us that the mean of a random variable which takes the value 1 with probability 1 is just 1 (as it really had better be!)

What about the variance? We know $E[X(X-1)] = g''(1)$, so $E[X^2] - E[X] = g''(1)$. Then

$$E[X^2] - (E[X])^2 = E[X] + g''(1) - (E[X])^2,$$

so $\mathrm{var}(X) = g''(1) - (g'(1))^2 + g'(1)$. Here, $g''(u) = \frac{d}{du} 1 = 0$, so we have the variance equals $0 - 1^2 + 1 = 0$. Again not too suprising; a constant random variable has variance 0.

Exercise 2. Suppose that a random variable takes the value 2 with probability 1. Write the generating function, and from it, find the mean and variance. ∎

Now. Imagine we are on generation i. The number of cases is X_i; this is, in general, random. Each of these randomly generates new cases, according to the secondary case distribution $g^*(u)$.

If $X_i = 0$, then we have zero secondary cases in the next generation, with certainty. If $X_i = 0$, then the distribution of the next generation looks like 0 with probability 1, and that's it. The generating function of that is going to be just 1 for all u: $u^0 P(X_i = 0) + u^1 P(X_i = 1) + \cdots = 1$ if $P(X_i = 0) = 1$.

If $X_i = 1$, then what is the size of the next generation? We just take $g^*(u)$, the secondary case distribution.

If $X_i = 2$, then we have to add up the secondary cases generated by two people. And here we will now *assume* they are independent. In a real epidemic they might not be, but as a model, let's start here. So we need to add up two independent random variables. We have already learned the slick trick for this—multiplying the generating function; the generating function for the number of secondary cases in generation $i+1$ conditional on there being two cases in generation i is $(g^*(u))^2$.

If $X_i = 3$, for the same reason, we will have $(g^*(u))^3$ and so on.

If we wrote:

$$E[u^{X_{i+1}}] = E[E[u^{X_{i+1}}|X_i] = E[u^{X_{i+1}}|X_i = 0]P(X_i = 0) + \cdots$$

$$E[u^{X_{i+1}}] = E[u^{X_{i+1}}|X_i = 0]P(X_i = 0) + E[u^{X_{i+1}}|X_i = 1]P(X_i = 1) + E[u^{X_{i+1}}|X_i = 2]P(X_i = 2) + \cdots + E[u^{X_{i+1}}|X_i =$$

This becomes

$$E[u^{X_{i+1}}] = 1 \cdot P(X_i = 0) + (g^*(u))P(X_i = 1) + (g^*(u))^2P(X_i = 2) + \cdots + (g^*(u))^jP(X_i = j) + \cdots$$

Suppose we just let $v = g^*(u)$. Then this whole thing is

$$E[u^{X_{i+1}}] = v^0 \cdot P(X_i = 0) + (v)^1P(X_i = 1) + (v)^2P(X_i = 2) + \cdots + (v)^jP(X_i = j) + \cdots$$

Of course we know what that is. It is $E[v^{X_i}] = g_i(v)$, the generating function of X_i evaluated at v. So the generating function of X_{i+1}, which is $g_{i+1}(u)$, is the generating function of X_i evaluated at v. But v is $g^*(u)$, so

$$g_{i+1}(u) = g_i(g^*(u)). \tag{3}$$

Let's stop here for a minute. Let's look at the mean. Remember: differentiate.

$$\frac{d}{du}g_{i+1}(u) = g_i'(g^*(u))\frac{d}{du}g^*(u)$$

evaluated at 1. First of all, what happens if you evaluate a generating function at $u = 1$?

$$g(u) = P(X = 0) + uP(X = 1) + u^2P(X = 2) + u^3P(X = 3) + \cdots.$$

At 1, this is just

$$g(1) = P(X = 0) + P(X = 1) + P(X = 2) + P(X = 3) + \cdots = 1.$$

So $g_i'(g^*(1)) = g_i'(1) = E[X_i]$. And if we evaluate the derivative of $g^*(u)$ at 1, we get the mean number of secondary cases per case. On the left we have the derivative of $g_{i+1}(u)$ evaluated at $u = 1$, which is the mean of the next generation. Let R equal the mean number of secondary cases per case. Then:

$$E[X_{i+1}] = RE[X_i].$$

If $R < 1$, this just gets smaller and smaller, and closer and closer to 0 with time. If $R > 1$, on average this just gets bigger and bigger, exponentially/geometrically with time. And: if $R = 1$, the mean is constant over time.

Let's compute the variance.

$$\frac{d^2}{du^2}g_{i+1}(u) = \frac{d}{du}\left(g_i'(g^*(u))\right)g^*(u) + g_i'(g^*(u))\frac{d}{du}g^*(u)$$

$$\frac{d^2}{du^2}g_{i+1}(u) = g_i''(g^*(u))g^{*\prime}(u)g^*(u) + g_i'(g^*(u))g^{*\prime}(u)$$

At 1:

$$\frac{d^2}{du^2}g_{i+1}(u)|_{u=1} = g_i''(g^*(1))g^{*\prime}(1)g^*(1) + g_i'(g^*(1))g^{*\prime}(1)$$

$$\frac{d^2}{du^2}g_{i+1}(u)|_{u=1} = E[X_i(X_i-1)]R + E[X_i]R$$

$$E[X_{i+1}(X_{i+1}-1)] = E[X_i(X_i-1)]R + E[X_i]R$$

$$E[(X_{i+1})^2] - E[X_{i+1}] = (E[X_i^2] - E[X_i])R + E[X_i]R$$

$$E[(X_{i+1})^2] = E[X_i^2]R - E[X_i]R + E[X_i]R + E[X_{i+1}]$$

$$E[(X_{i+1})^2] = E[X_i^2]R + RE[X_i]$$

Now: $\text{var}(Y) = E[Y^2] - (EY)^2$, so $E[Y^2] = \text{var}(Y) + (EY)^2$.

$$\text{var}(X_{i+1}) + (E[X_{i+1}])^2 = \text{var}(X_i)R + E[X_i]^2R + RE[X_i]$$

$$\text{var}(X_{i+1}) = \text{var}(X_i)R + E[X_i]^2R + RE[X_i] - R^2E[X_i]^2$$

$$\text{var}(X_{i+1}) = \text{var}(X_i)R + RE[X_i]$$

We can show that if $R < 1$, this actually goes to 0. But if $R = 1$, this keeps growing even though $EX_i = 1$! More on this in a moment.

The next move the mathematicians came up with is even slicker. Suppose now we start with $X_0 = 1$, and $g_0(u) = u$. Then,

$$g_1(u) = g_0(g^*(u)).$$

But what is $g_0(u)$? It is u. It is the thing that maps its input to itself. It gives you back what you put in. So $g_0(g^*(u)) = g^*(u)$.

$$g_1(u) = g^*(u).$$

And then

$$g_2(u) = g_1(g^*(u)) = g^*(g^*(u)).$$

In general the n generation is the n-fold composition of g^* with itself.

$$g_i(u) = g^*(g^* \cdots (u)) \cdots)).$$

We could just as well write

$$g_{i+1}(u) = g^*(g_i(u))$$

as long as we start with just one person at the beginning.

One of the questions we have is what happens in the long run. If the epidemic ever hits 0 cases, it stays there forever. Zero is a so-called *absorbing state*. The state space is $\{0, 1, 2, \ldots\}$. This is an infinite set, and there is a lot of room at the top. In this model, the epidemic might keep going and *never* come back.

We can find the probability of extinction on a certain generation by looking at $P(X_i = 0)$, which is just the generating function evaluated at 0: $g_i(0) = P(X_i = 0)$. So

$$g_{i+1}(0) = g^*(g_i(0)).$$

If we start with 1 case, we have $g_1(0) = g^*(0)$. It is the chance of extinction in the first round of transmission, i.e. the chance a case just produces no secondary cases.

Then, $g_2(0) = g^*(g^*(0))$. Now, the long run extinction probability is the long run limit of this iteration. We know a generating function is convex. Say the slope at $u = 1$ is less than 1. It can then never touch the 45 degree line. Every time you iterate, you get a bigger number, and geometrically you must converge to 1—certainty of ultimate extinction.

The same thing is true if the slope at $u = 1$ is exactly equal to 1! So even if on average $R = 1$, the epidemic goes extinct with probability 1 in the long run. But strangely, the mean always equals 1, and the variance keeps increasing. What is happening is that if you imagine an ensemble of these epidemics happening at the same time, the average stays the same. More and more of them hit 0 and go extinct as you keep watching. But those that don't go extinct get bigger and bigger.

Finally, what if the slope at $u = 1$ is greater than 1? Now, the graph of the generating function has another intersection with the identity line. In the long run, the iterates converge to that intersection, and there is in general a nonzero chance the epidemic will escape to infinity and never die out. But a nonzero chance the epidemic will die out is something we see as well. So the simplicity of the recurrence for the mean conceals what is really happening. A certain number of these epidemics die out. Infinity isn't part of epidemiology, so what we really think is that when the number of cases gets large enough, eventually we can't use the same secondary case distribution; we run out of susceptibles. The approximation we made in the Galton-Watson process no longer applies. We think of the Galton-Watson process as a model of the beginning of an epidemic ... as a model of invasion.

Here, the slope at 1 is the basic reproduction number, the mean of the secondary case distribution. We see that when the basic reproduction number is less than OR EQUAL TO 1, the process dies out. Strictly speaking, we have to say that it converges in probability to 0. We also found that the intersection of the

generating function with the identity line gives the chance of ultimate extinction (starting from 1) when the basic reproduction number is greater than 1.

Exercise 3. Make barplots of the negative binomial distribution with mean 2 for $k = 20$, $k = 2$, $k = 1$, and $k = 0.1$. Holding the mean constant, how can you have more superspreading by changing k? ∎

Worked Exercise. Assume a negative binomial distribution with $k = 20$ and mean 2. Roughly calculate the extinction probability; find that place where the pgf crosses the 45 degree line by trial and error, or by reading it approximately off of a graph. Try it for $k = 2$, $k = 1$, and $k = 0.1$.

Solution. First, the mean is given by $k(1-p)/p$, so if $k = 20$ and the mean is 2, we have $20(1-p)/p = 2$; $p = 10/11$. Let's do this using R, numerically:

```
g <- function(u,p,k) {p^k/(1-(1-p)*u)^k}
uniroot(function(u)g(u,p=10/11,k=20)-u,c(0.0001,0.9999))

## $root
## [1] 0.2245248
##
## $f.root
## [1] 3.629481e-06
##
## $iter
## [1] 6
##
## $init.it
## [1] NA
##
## $estim.prec
## [1] 6.103516e-05
```

If you look at the value of `$root` in the output of `uniroot`, we find out the answer is about 22%. It might be cleaner if we coded this up to solve for the value of p for us. The point is that we are changing k but keeping the mean constant.

```
g2 <- function(u,mu,k) {
  p <- k/(k+mu)
  g(u,p,k)
```

```
}
uniroot(function(u)g2(u,mu=2,k=20)-u,c(0.0001,0.9999))$root

## [1] 0.2245248

uniroot(function(u)g2(u,mu=2,k=2)-u,c(0.0001,0.9999))$root

## [1] 0.3819707

uniroot(function(u)g2(u,mu=2,k=1)-u,c(0.0001,0.9999))$root

## [1] 0.500014

uniroot(function(u)g2(u,mu=2,k=0.1)-u,c(0.0001,0.9999))$root

## [1] 0.8904393
```

What is happening to the extinction probability? Why?

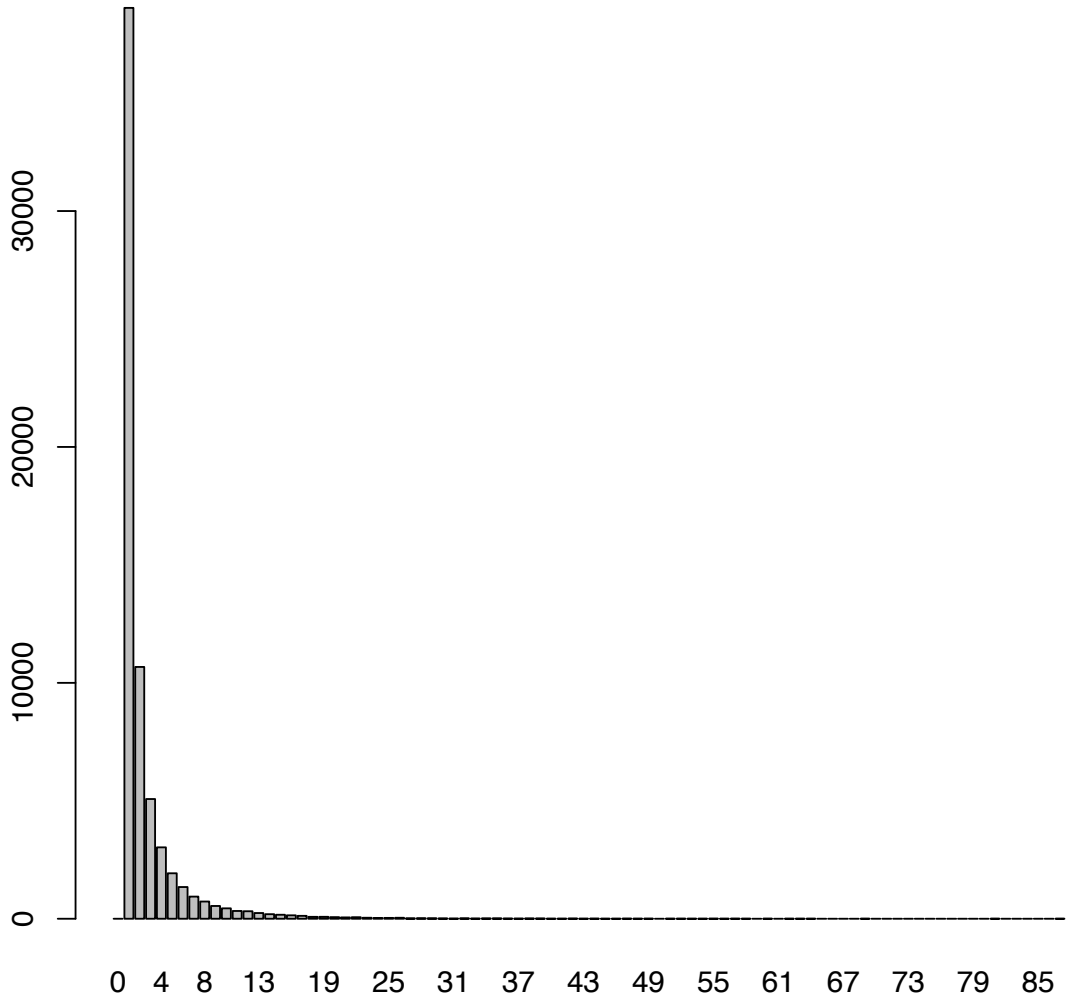
Let's simulate a Galton-Watson process in R with a negative binomial secondary case parameter. Call with `n` the number of index cases, `cumul` set to 0, `rr` the mean we assume, `agg` the assumed aggregation parameter (k).

```
gwsim <- function(n,cumul,rr,agg,curit=0,maxit=1024,mxclus=4096) {
  if (rr<0) {
    error("invalid r")
  }
  if (curit>maxit || cumul>=mxclus) {
    cumul
  } else if (n <= 0) {
    cumul
  } else {
    nc <- sum(rnbinom(n,size=agg,mu=rr))
    gwsim(nc,cumul+n,rr,agg,curit+1,maxit)
  }
}
gwsim(1,0,0.6,2)

## [1] 1
```

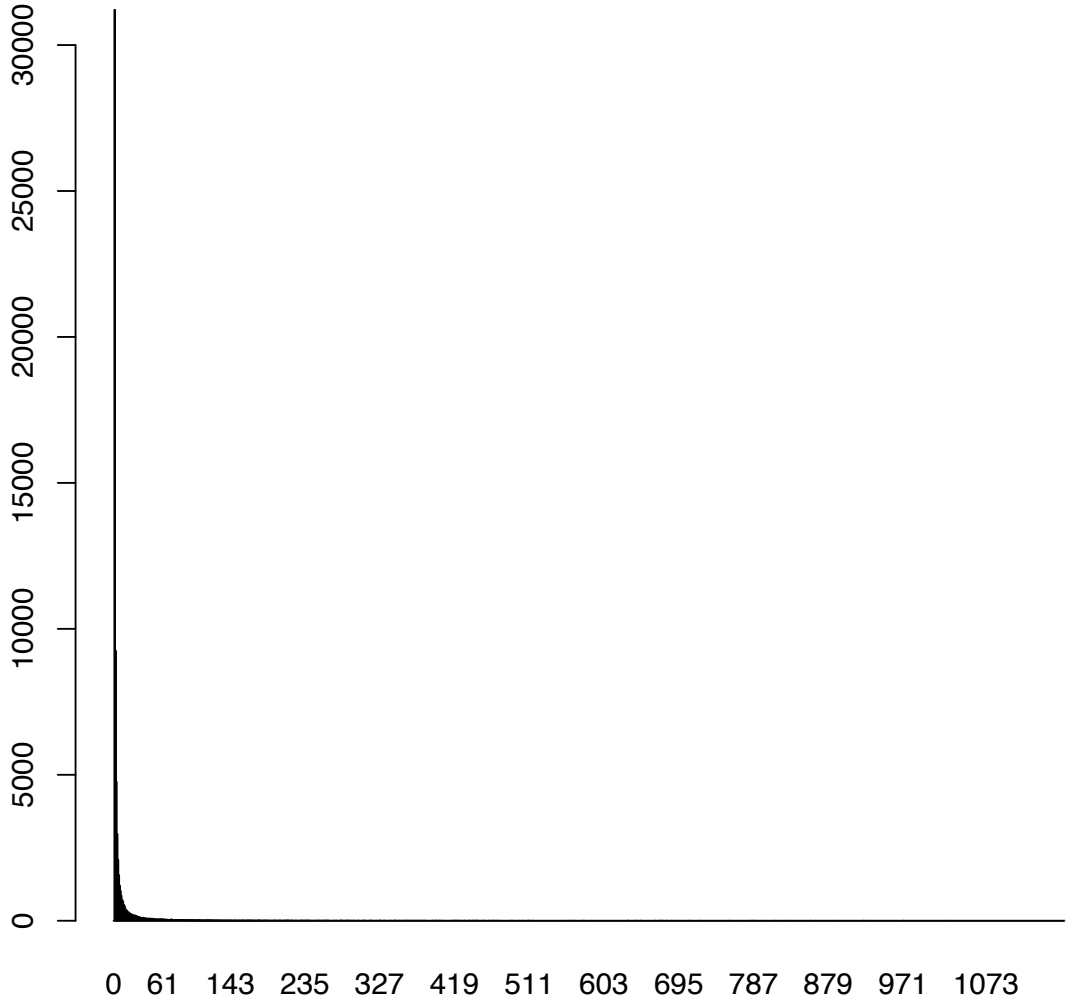
This gives you the cumulative outbreak size. But this is random, and we can't stop with just one run. Let's do a lot of runs and plot the distribution:

```
gwsim.rep <- function(reps,n,cumul,rr,agg) {
  values <- rep(NA,reps)
  for (ii in 1:reps) {
    values[ii] <- gwsim(n,cumul,rr,agg)
  }
  values
}
tbl <- function(aa) {
  t1 <- table(aa)
  nz <- setdiff(0:(max(aa)),unique(sort(aa)))
  t1a <- as.vector(t1)
  names(t1a) <- NULL
  t2 <- data.frame(s=as.numeric(names(t1)),num=t1a)
  t2a <- data.frame(s=nz,num=rep(0,length(nz)))
  t3 <- rbind(t2,t2a)
  t4 <- t3[order(t3$s),]
  t5 <- t4[,2]
  names(t5) <- t4[,1]
  t5
}
barplot(tbl(gwsim.rep(65536,1,0,0.6,2)))
```



Let's try some more; here with a mean of 0.9:

```
barplot(tbl(gwsim.rep(65536,1,0,0.9,2)))
```



Application

Let's apply this to a real epidemic. First, let's consider the winter measles outbreak centered on Disneyland in California from late 2014–2015. Full details are on the California department of Public Health website. We will assume that a total of 42 people were exposed in Disneyland from an unknown number of

index cases, and that the total outbreak size was 139.

Essentially, we had 97 transmissions from 139 total cases, so we could simply think of this as indicating $R = 97/139$. You get the same thing if you assume each generation is a factor of R smaller than the previous. The total cluster size would be

$$C = \sum_{i=0} R^i = \frac{1}{1-R}$$

counting each index case. If we solve for R, we would get $R = 1 - 1/C$. The mean cluster size C is $139/42 \approx 3.31$, which gives us $1 - 1/C = 0.69$, the same thing.

How could we get a confidence interval? It turns out there is a closed form likelihood for this problem if we assume the negative binomial distribution. But that is lucky and special. What would we do in general?

From prior work, we had reason to believe the aggregation parameter k was known; let's say $k = 0.21$. Now, we'd like a confidence interval. The question is, assuming this model, how big would the true R have to be before you had a 2.5% chance of seeing anything as small as (or smaller than) the real answer.

```
do.gwsim <- function(val,agg,reps=65536,nstart=42) {
  ans <- rep(0,reps)
  for (ii in 1:reps) {
    ans[ii] <- gwsim(nstart,0,rr=val,agg=agg)
  }
  sim.cluster.size <- ans/nstart
  rr = 1 - (1/sim.cluster.size)
  quantile(rr,c(0.025,0.975))
}
do.gwsim(1.10,0.27)

##      2.5%     97.5%
## 0.6842105 0.9909980
```

Based on these numbers, we get an upper bound of around 1.10 for our confidence interval. This is how large R can be with the LOWER confidence bound just touching our estimate. This program assumes a subcritical epidemic (unlike the formula), by the way; pay no attention here to the "upper bound". Now, let's see how low we can take R and have the estimated value just barely be on the edge of the confidence interval:

```
do.gwsim(0.50,0.27)

##      2.5%     97.5%
## 0.2222222 0.6934307
```

This technique is called *parametric bootstrap*.

Exercise 4. Assume now the aggregation parameter is 0.45, and compute the confidence interval (approximately). ∎

Exercise 5. How long did this assignment take you? ∎