Notes - Statistical Rethinking

Benjamin Wee

Lecture 01

Need to understand the internal processes of a model to use them properly. Construction from bottom up.

Eventually you will make a statistical model which will misbehave. But it will only misbehave according to your expectations. It is behaving *exactly* according to its design.

We construct models to do things which are difficult for us. We design them to be good at things we are bad at. But they are fantastically bad at the things people find easy. They are blind to our intentions.

"Animated by truth"

Frequentist tests are made for experimental designs (or factorial design models?).

Flasifying null models are insufficient for understanding how the world works.

"Statistical expectation"

Bayesian inference lays bare all the imperfections of logical inference.

Information critera are a cross validation metric

Anything which looks random to you is because you can't predict the outcome. Ignorance in the face of uncertainty.

Everything in Bayes is pseudo random, there is a deterministic process (probability distributions which produce random numbers). We taught computers to generate pseudo random numbers (we taught them a deterministic way to generate "randomness")

Propositional logic. What is a valid deduction. Bayes extends this to continuous logic.

We want to understand "the golem's belief", not the scientist's.

Bayesian stats -> Laplacian stats. Father of applied stats. Epistemic rather than ontological.

Coins themselves are not random. Our inability to predict their side which makes them random. They are a good randomisation device.

FLipping coins fairly is a chaotic system. So sensitive to initial conditions that they appear unpredictable, and hence randoom.

randomness is a property of bugs in our knowldege.

Based on our assumptions/conjecture, how different ways could we observe the data?

Probability is merely counting the number of ways the data could appear given our conjecture GIVEN all the POSSIBLE paths that could be generated from our conjecture. This is where the multiplication rule of probability comes from. We are multiplying all the possible "paths" of observing the data from our assumptions (an efficient way of counting).

Lecture 02

Extension of propositional logic to continuous probability

```r
# Counts to probability
ways <- c(3 ,8, 9)
ways/sum(ways)
```

```
## [1] 0.15 0.40 0.45
```

Probability is just a shortcut for counting probabilties. It cannot tell us what is true, nothing can. It's simply, make some assumptions, and see what the logical consequences are.

Applied statistics as a type of engineering. Building a bridge across a river of ignorance.

Design > *Condition* > *Evaluate* Story of the small world, executing in the small world, does the model make sense?

Translating data story into probabilty statements:

1) Law of Total Probability: Make sure the probabilities sum to 1 Two to remember to construct things:
2) Sum rule: Things that are alternative: add
3) Product rule: Things that happen together: multiply

The prior and posterior are one in the same. The prior summarises all the past datapoints AND the sample size. May want the raw data IF you want to fit a different model. Other models may find other relevant information.

Bayesian learning is optimal in our small world. Counting up the logical ways data could arise.

Bayes: Translate a prior state of information (before data arrives) into posterior state of information (after data arrives)

Prior: We have prior counts of the plausibilities of each conjecture (e.g. philisophy or previous data). What is the logical change of belief. Prior: assigning information states. Joint prior over data and parameters. Prior predictive distribution. Idea of simulated dataset. Terrible predictions before you got data. **This is the way to know what the prior actually means**

Posterior predictive distribution: hopefully better predictions after data is observed. Good way to test what the golem thinks its learned from the data.

Density: "Rate of change" – It's continuous. Relative values matters, not absolute.

Overfitting - Need to build in conservatism into golems. Otherwise it will think the sample is "everything"

Prior as a state of information, there's no "true" prior. Important to embody your golem with information it to guard against *inferential risk*. Truth isn't what's relevant here. It's the risk of taking action from the model that matters.

$$Posterior = \frac{Likelihood \times prior}{AverageLikelihood}$$

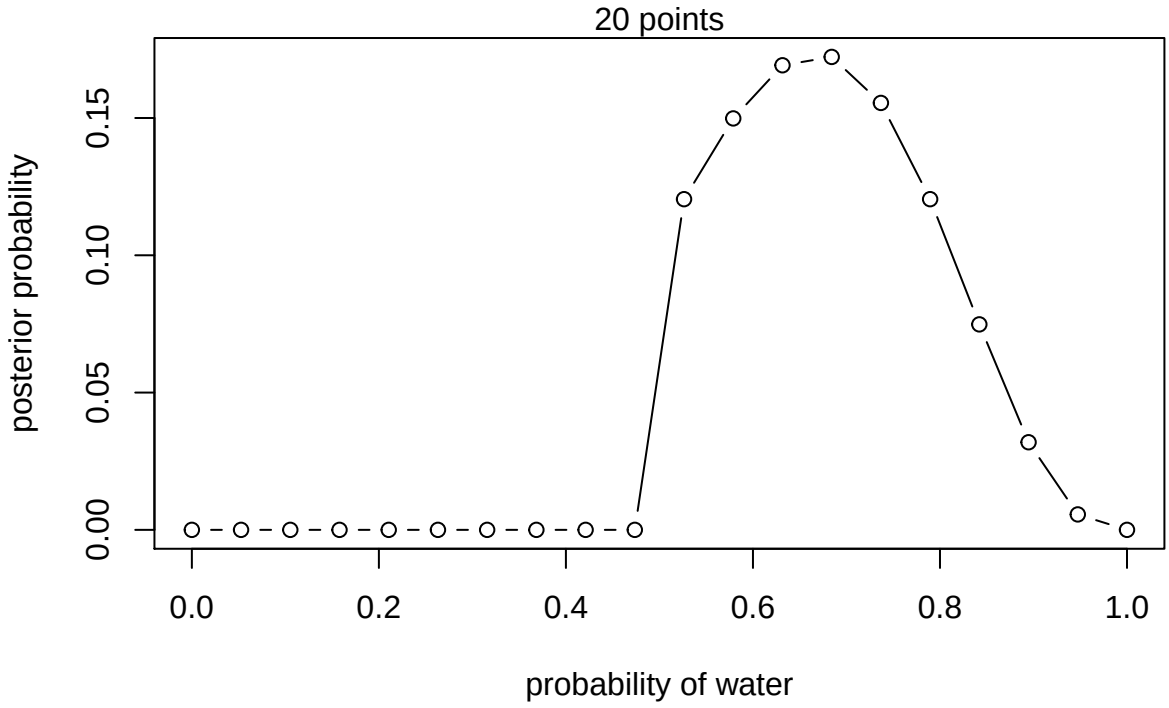
Where the average likelihood normalises the posterior so that it integrates to 1. The posterior is the *standardised product of the likelihood and proior

Textbook Code

Chapter 2

```r
# Page 40
# Define Grid
p_grid <- seq(from = 0, to =1, length.out = 20)
```

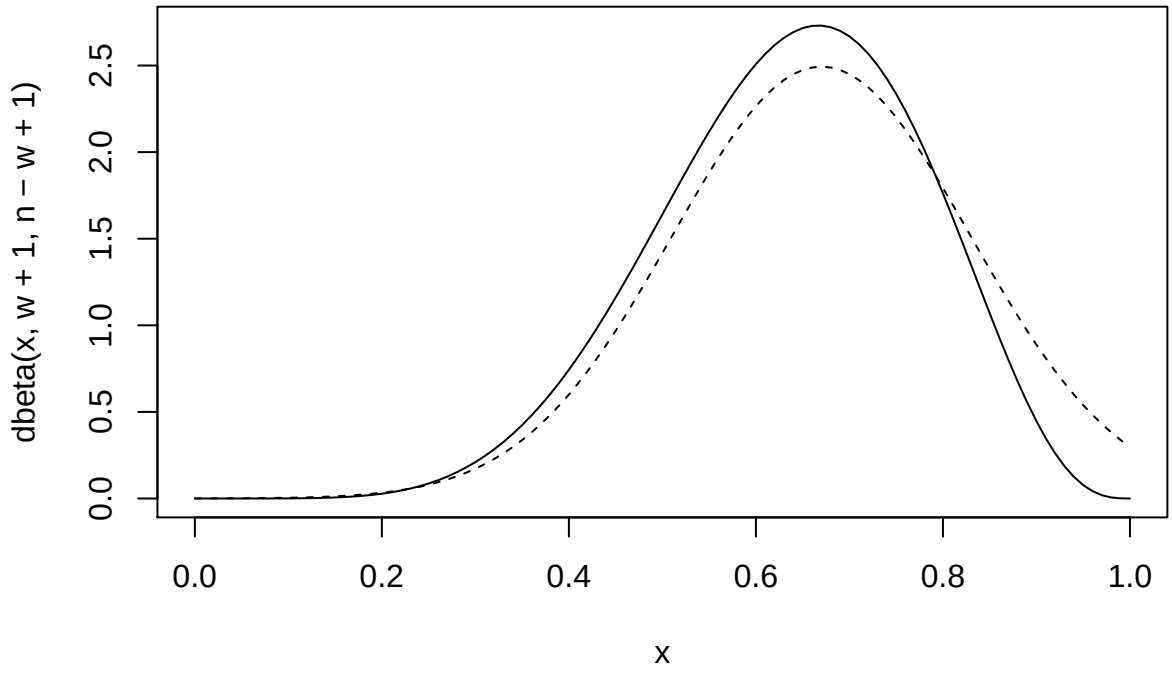
```r
# Define prior
#prior <- rep(1, 20)
prior <- ifelse(p_grid<0.5, 0, 1)
#prior <- exp(-5*abs(p_grid - 0.5))

# Compute likelihood at each value in grid
likelihood <- dbinom(6, size = 9, prob = p_grid)

# Compute product of likelihood and prior
unstd.posterior <- likelihood * prior

# Standardise the posterior so it sums to 1
posterior <- unstd.posterior / sum(unstd.posterior)

# Visualise posterior
plot( p_grid , posterior , type="b" ,
xlab="probability of water" , ylab="posterior probability" )
mtext( "20 points" )
```




```r
# Page 42
library(rethinking)
globe.qa <- map(
  alist(
    w ~ dbinom(9, p), # Binomial Likelihood
    p ~ dunif(0,1)    # Uniform Prior
  ),
  data = list(w = 6)
)

# Display summary of quadratic approximation
precis(globe.qa)
```

```
##   Mean StdDev 5.5% 94.5%
## p 0.67   0.16 0.42  0.92
```

```
# Page 43
# Analytical Calculation
w <- 6
n <- 9
curve(dbeta(x, w+1, n-w+1), from = 0, to = 1)

# Quadratic approximation
curve(dnorm(x, 0.67, 0.16), lty = 2, add = TRUE)
```



Chapter 3 Sampling the imaginary

Bayesian inference is distinguised by a broad view of probability. Not by use of Bayes' therorem.

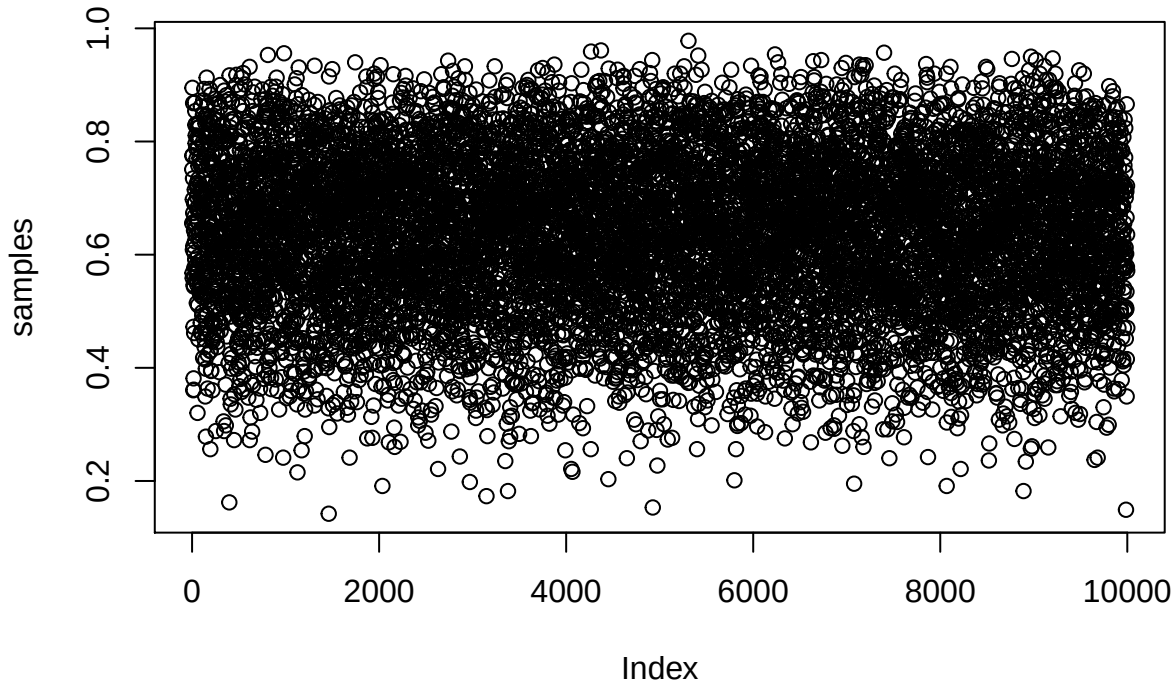
3.1 Sampling from a grid-approximate posterior

```
# Page 49
# Conditional Probability Calculation

PrPV <- 0.95
PrPM <- 0.01
PrV <- 0.001
PrP <- PrPV*PrV + PrPM*(1-PrV)
( PrVP <- PrPV*PrV / PrP )
```

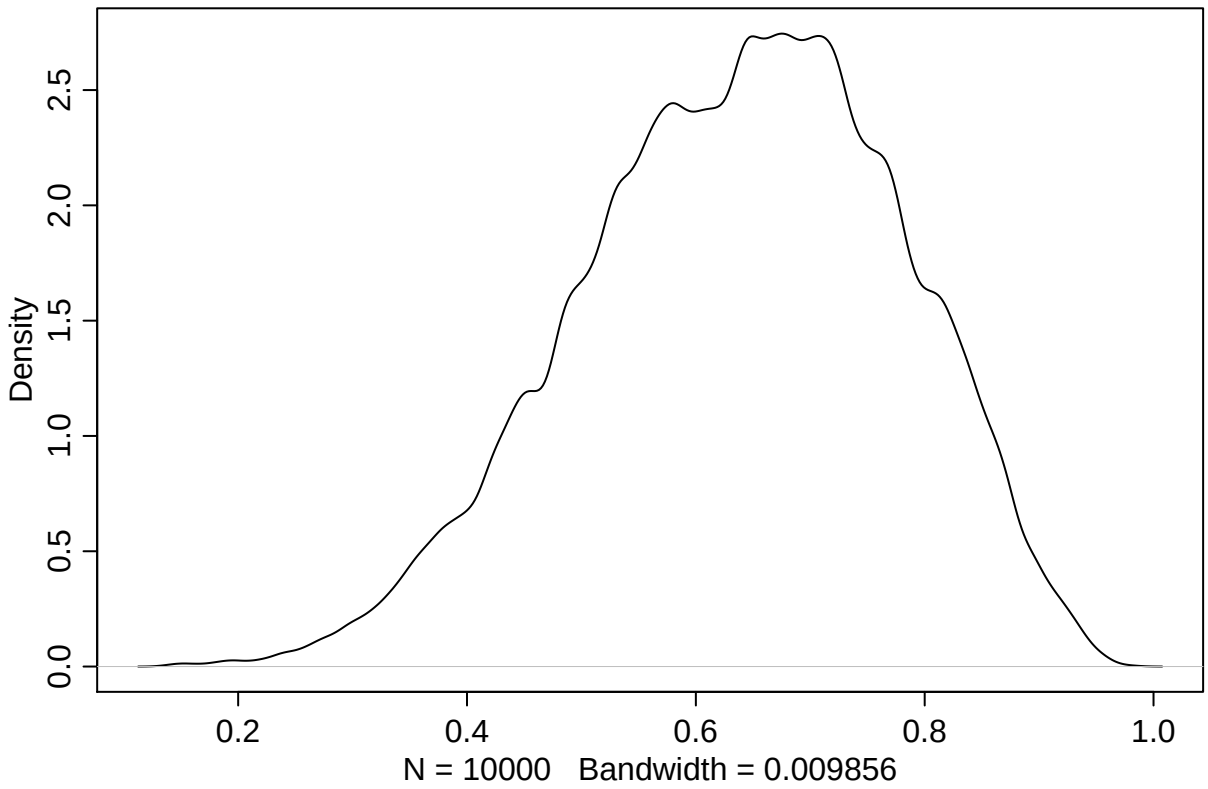
```
## [1] 0.08683729
```

```
# Page 52
# Sampling from a grid approximate posterior
p_grid <- seq( from=0 , to=1 , length.out=1000 )
```

```r
prior <- rep( 1 , 1000 )
likelihood <- dbinom( 6 , size=9 , prob=p_grid )
posterior <- likelihood * prior
posterior <- posterior / sum(posterior)

samples <- sample( p_grid , prob=posterior , size=1e4 , replace=TRUE)
plot(samples)
```




```r
# Density approximiation of samples
library(rethinking)
dens(samples)
```



3.2 Sampling to summarise

3.2.1. Intervals of defined boundries

```
# add up posterior probability where p < 0.5
sum(posterior[ p_grid < 0.5 ]) # So about 17% of posterior probability is under 0.5
```

```
## [1] 0.1718746
```

```
# Performing same calculation from samples of the posterior
sum(samples<0.5) / 1e4 # About the same
```

```
## [1] 0.168
```

```
# See how much probability is between 0.5 and 0.75
sum( samples > 0.5 & samples < 0.75 ) / 1e4
```

```
## [1] 0.6082
```

3.2.2. Intervals of defined mass

It's easy to keep track of what's being summarised as long as you pay attention to how the model is defined. We can use sampling to get the boundries of the *lower* 80% posterior probability.

```
# R code 3.9
quantile(samples, 0.8)
```

```
##       80%
## 0.7607608
```

Similarly the middle 80% interval lies between the 10th and 90th percentile. Using sampling:

```
# R code 3.10
quantile(samples, c(0.1, 0.9))
```

```
##       10%       90%
## 0.4504505 0.8148148
```

These are known as **percentile intervals (PI)** – good for summarising symmertical distributions. They are limited for assymetrical distributions and are *not* perfect for for supporting inferences about which parameters are consistent with the data.

The posterior distribution is consistent with observing 3 waters in 3 tosses with a uniform prior. This is highly skewed with a maximum boundry balue of $p = 1$. Computing this with grid approximation:

```
# R code 3.11 - Grid approximation of posterior + sampling
p_grid <- seq(from = 0, to = 1, length.out = 1000)
prior <- rep(1, 1000)
likelihood <- dbinom(3, size = 3, prob = p_grid)
posterior <- likelihood * prior
posterior <- posterior / sum(posterior)
samples <- sample(p_grid, size = 1e4, replace = TRUE, prob = posterior)
```

Computing the 50% percentile intervals:

```
# R code 3.12: The interval assigns 25% probability mass above and below the interval.
PI(samples, prob = 0.5)
```

```
##       25%       75%
## 0.7077077 0.9289289
```

But this example excludes the most probably parameter value, $p = 1$. Therefore, the PI is misleading in describing the shape of this posterior.

The Highest Posterior Density Interval (HPDI) – narrowest interval containing the specified probability mass.

```
# R code 3.13 - HPDI - smallest interval with 50% probability mass
HPDI(samples, prob = 0.5)
```

```
##      |0.5      0.5|
## 0.8428428 0.9989990
```

HPDI has some advantages over PI, but in many cases they are similar. In this case, the intervals are different due to a *highly skewed distribution*. Using different probability masses:

```
HPDI(samples, prob = 0.8)
```

```
##      |0.8      0.8|
## 0.6676677 1.0000000
```

```
HPDI(samples, prob = 0.95)
```

```
##     |0.95     0.95|
## 0.4764765 1.0000000
```

Disadvantages of HPDI – More computationally intensive than PI and greater simulation variance (sensitive to how many samples are drawn from the posterior).

Overall, if the choice of interval type makes a big difference, then we *shouldn't* be using intervals to summarise the posterior. Better of plotting the estimate of the posterior distribution.

3.2.3. Point Estimates

Maximum a posteriori (MAP) estimate – parameter with the highest posterior probability.

```
# R code 3.14 -- MAP
p_grid[which.max(posterior)]
```

```
## [1] 1
```

You can approximate the point if you have samples from the posterior:

```
# R code 3.15
chainmode(samples, adj = 0.01)
```

```
## [1] 0.9970614
```

Could also report posterior mean and median:

```
# R code 3.16
mean(samples)
```

```
## [1] 0.8015076
```

```
median(samples)
```

```
## [1] 0.8438438
```

But how do we choose between MAP, median and mean?

We could use a **loss function**. This is a rule which gives the cost associated with using any point estimate. *Different loss functions imply different point estimates.* Calculating the loss for any point estimate means using the posterior to average over our uncertainty in the true value. If we arbitrarily choose 0.5 as our point estimate, the *expected loss* will be:

```
# R code 3.17
# Sum of posterior probability multiplied by deviation of point estimate from all other possible parame

# Goal is to *minimise* this value (minimising the weighted average loss).
sum(posterior * abs(0.5 - p_grid))
```

```
## [1] 0.3128752
```

```
# sum(weight_i * abs(point_estimate - parametervalues_i))
```

We can repeat this calculation for each decision and choose the smallest value.

```
# R code 3.18.
loss <- sapply(p_grid, function(d) sum(posterior*abs(d-p_grid)))
```

```
# R code 3.19 - Find parameter value which minimises the loss
p_grid[which.min(loss)] # same as posterior median in this case.
```

```
## [1] 0.8408408
```

In principle, the details of the applied context may demand unique loss functions. Some loss functions may be highly assymetric. Rather, point estimates should help *describe* the shape of the posterior distribution.

3.3 Sampling to simulate predction

1) *Model checking* – Simulating implied observations to check if the model fit is correct and to investigate model behaviour.
2) *Software validation* – Simulate observations under a known model and then attempt to recovr the values of the parameters the data were simulated under.

3) *Research design* – If you can simulate observations from your hypothesis, then you can evaluate whether the research design can b effective. In a narrow sense, this means doing *power analysis*, but the possibilties are much broader.

4) *Forecasting* – Estimates can be used to simulate new predictions, for new cases and future observations. These forecasts can be useful as applied prediction, but also for model criticism and revision.

3.3.1. Dummy Data

A fixed 'true' proportion of water p exists, and that is the target of our inference. So far, our assumptions have allowed us to infer the plausilibilty of each possible value of p after observations. Using the same assumptions, we can simulate the observations that the model implies, since the likelihood works in both directions.

Given a realised observation, the likelihood function says how plausible an observation is.

Given only the parameters, the likelihood defines a distribution of possible observations that we can sample from – to simulate observations.

Therefore, bayesian models are *always* **generative** – capable of simulating predictions.

Using a binomial likelihood:

$$Pr(w|n,p) = \frac{n!}{w!(n-w)!}p^w(1-p)^{n-w} n = 2, p = 0.7$$

Computing the likelihood:

```
# R code 3.20
dbinom(0:2, size = 2, prob = 0.7)
```

```
## [1] 0.09 0.42 0.49
```

9% chance of observing $w = 0$, 42% chance of observing $w = 1$, 42% chance of $w = 2$.

Simulating observations from the likelihood:

```
#R code 3.21
rbinom(1, size = 2, prob = 0.7)
```

```
## [1] 2
```

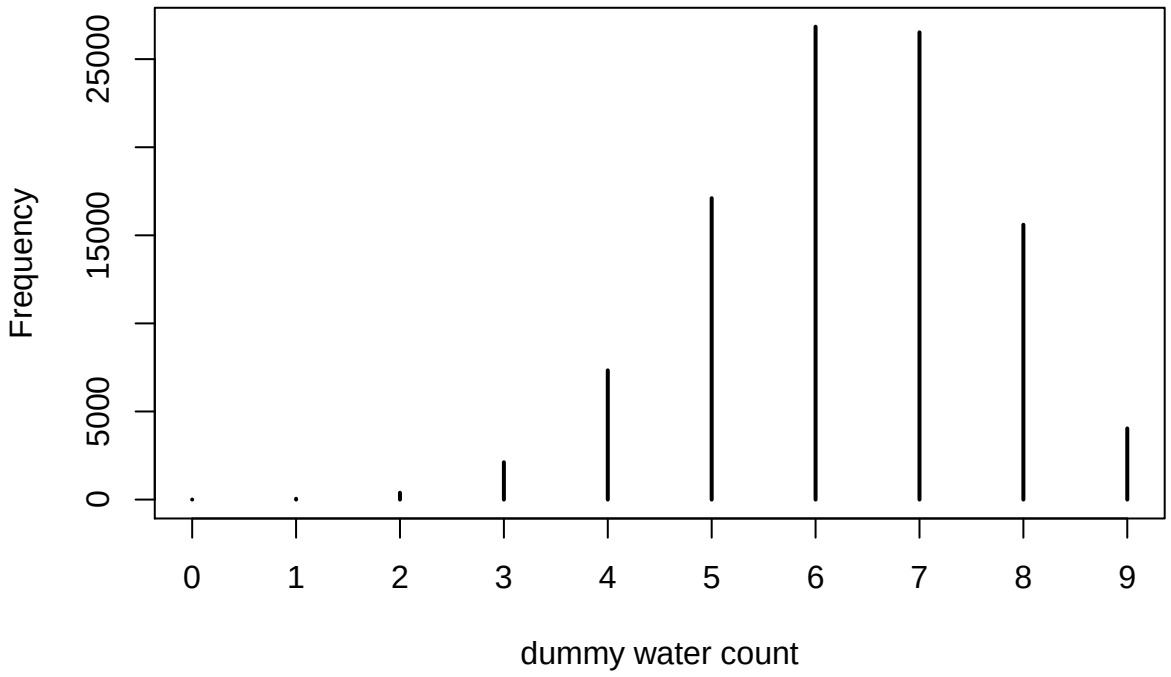
```
#R code 3.22
rbinom(10, size = 2, prob = 0.7)
```

```
##  [1] 2 1 2 1 1 2 1 2 1 1
```

```
# R code 3.23
dummy_w <- rbinom(1e5, size = 2, prob = 0.7)
table(dummy_w)/1e5
```

```
## dummy_w
##       0       1       2
## 0.09005 0.41889 0.49106
```

```
# R code 3.24 -- Extending this to 9 tosses
dummy_w <- rbinom(1e5, size = 9, prob = 0.7)
simplehist(dummy_w, xlab = "dummy water count")
```



3.3.2. Model Checking

Model checking means:

1) Ensuring model fitting worked correctly
2) Evaluating the adequacy of a model for some purpose

Bayesian models are *generative*: able to simulate observaions and estimate parameters from observations. Once the model is conditioned on data, you can simulate to examine the model's *empirical expectations*.

3.3.2.1. *Did the software work?* – Check implied predictions and the data used to fit the model (retrodictions). How well does the model reproduce the data used to educate it. There should be *some* but not *exact* correspondence. Otherwise this may suggest something wrong with the software. This is a simple check to catch silly mistakes since there's no sure way to check that the software is working correctly.

3.3.2.2. *Is the model adequate?* – Looking for aspects of the data that are not described well by the model's expectations. The goal is *not* to test whether the model's assumptions are "true", because all models are false. Rather, the goal is to assess exactly how the model fails to describe the data, as a path towards model comprehension, revision and improvement. All models will fail in some aspect – use own judgement and others – to decide whether particular failure is important or not. Sample from entire posterior distribution, not just a point estimate. Posterior contains lots of infromation about uncertainty. Stripping this away leads to overconfidence.

Need to be aware of uncertainty from the implied predictions of the model.

1) Observation uncertainty – For each unique parameter value p, there is a unique implied pattern of observations that the model expects. There is uncertainty in the predicted observations, since even if you know p with certainty, you won't know the next globe toss with certainty.
2) Uncertainty about the parameter p. The posterior over p embodies this uncertainty. Since there is uncertainty about p, there is uncertainty about everything that depends upon p. The uncertainty in p will interact with the sampling variation, when we try to assess what the model tells us about outcomes.

We want to *propagate* parameter uncertainty – carriy it forward – as we evaluate the implied predictions. For each value of parameter p, there is an implied distributions of outcomes. We can average all these predictive

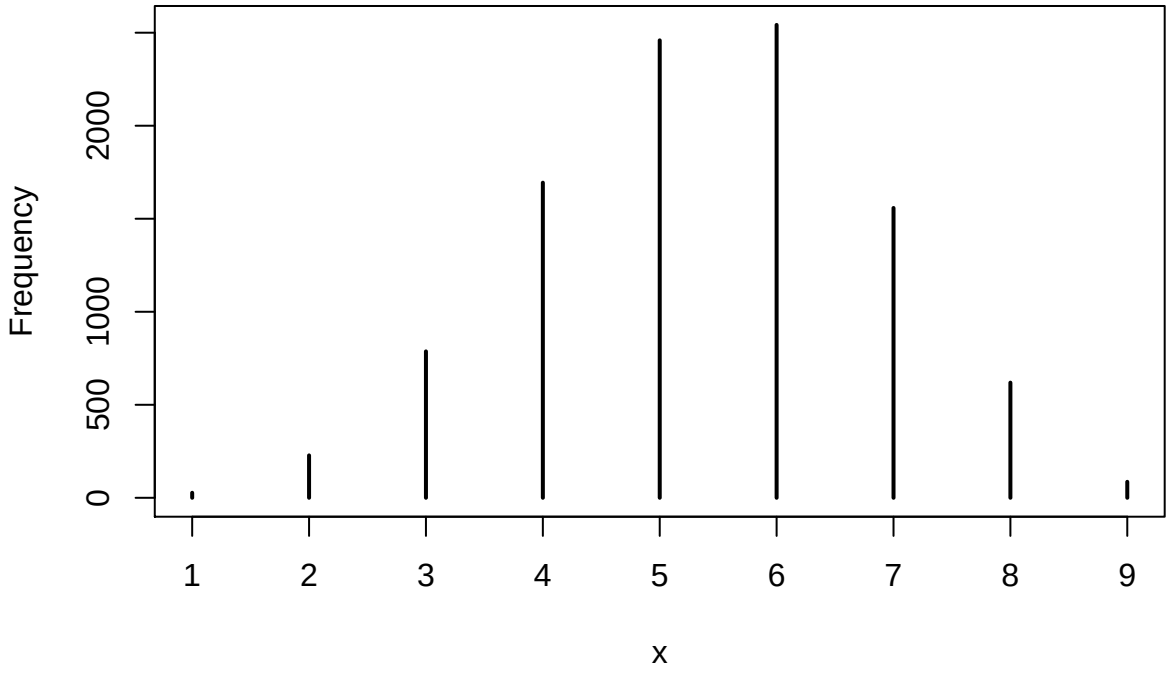
distributions together using the posterior probabilities of each value of p to get a **posterior predictive distribution**.

The resulting distributions is for predictions, but it incorperates all of the uncertainty embodied in the posterior distribution for the parameter p. As a result, it is honest.

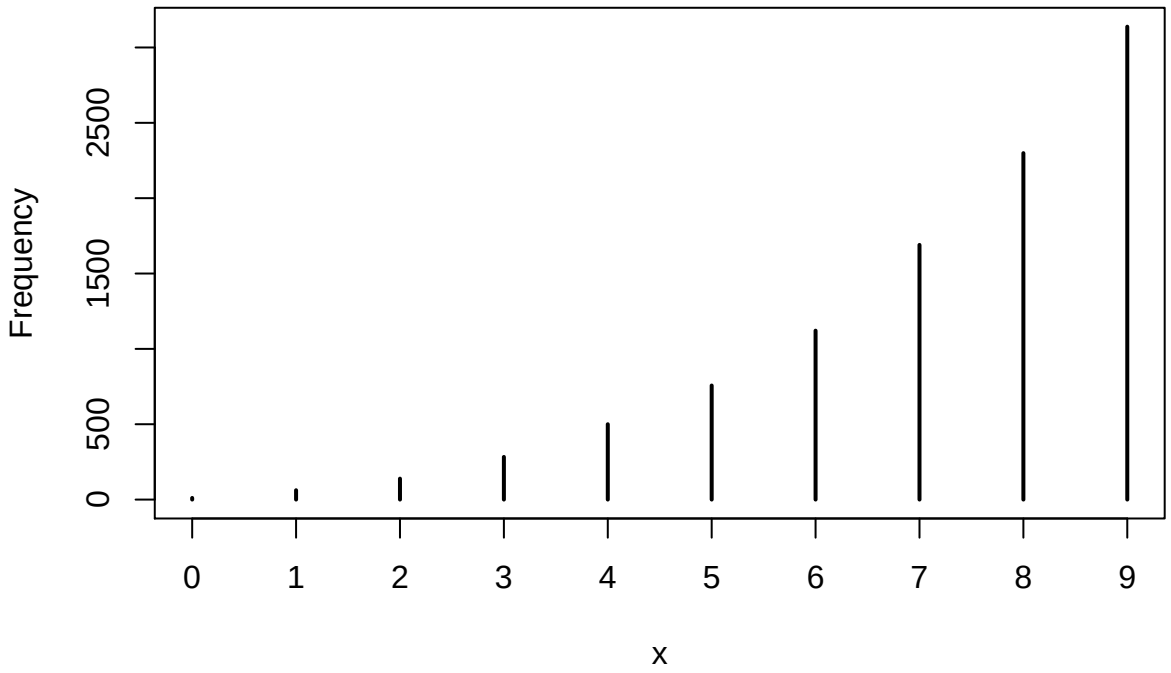
If you use only a single parameter value to compute implied predictions, say the most probable value at the peak of the distribution, you'd produce an *overconfident distribution of predictions*. This will lead you to believe the model is more consistent with the data than it really is – predictions will cluster around the obesrvations more tightly. This illusion arises from tossing away uncertainty about the parameters.

To see this:

```
# R code 3.25
# Generates 10,000 simulated predictions of 9 globe tosses assuming p = 0.6
w <- rbinom(1e4, size = 9, prob = 0.6)
simplehist(w)
```




```
# R code 3.26
# Sampled values appear in proportion to their posterior probabilities -- the resulting simulated obser
w <- rbinom(1e4, size = 9, prob = samples)
simplehist(w)
```



Can also consider distributions of longest run of waters or number of switches within 9 globe tosses.

Models are always wrong, in some sense *mis-specified*. But whether the mis-specification should lead us to try other models will depend on our specific interests.

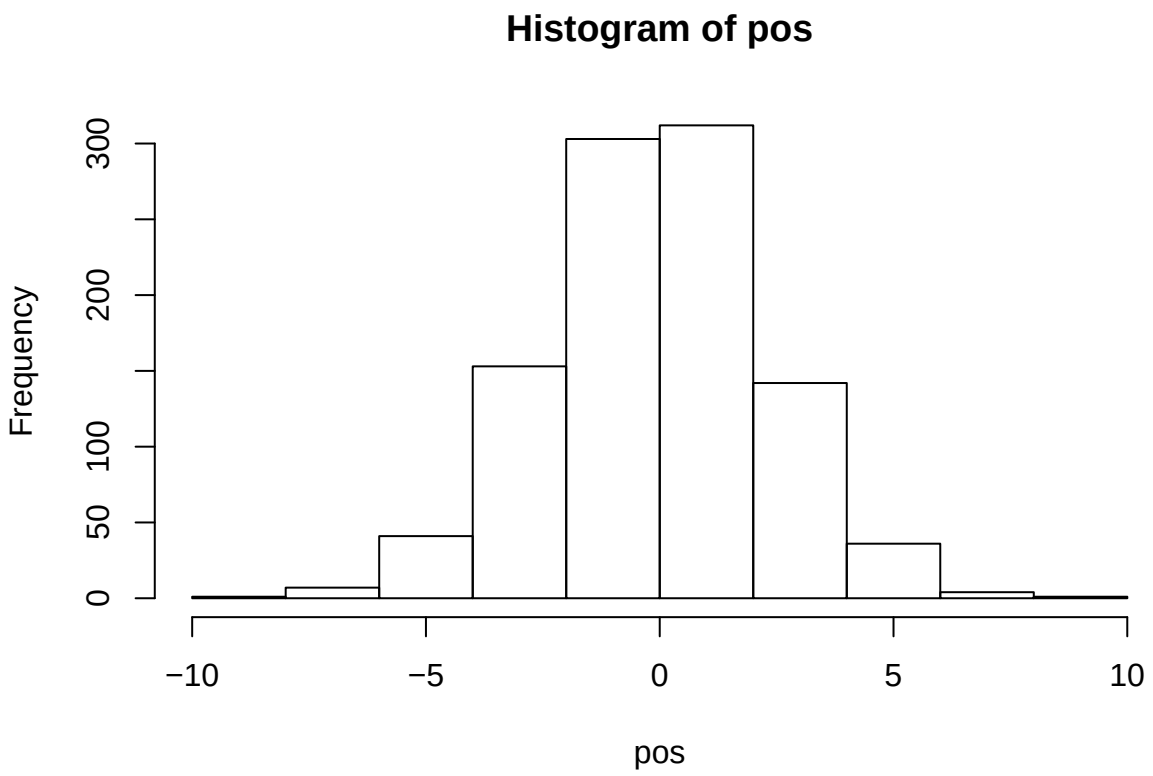
Chapter 4 Linear models

Linear regression is a descriptive model that corresponds to many different process models. Not universally useful, but a strong claim to being foundational. Once you learn to build and interpret linear regression models, you can quickly move on to other types of regressions which are less normal

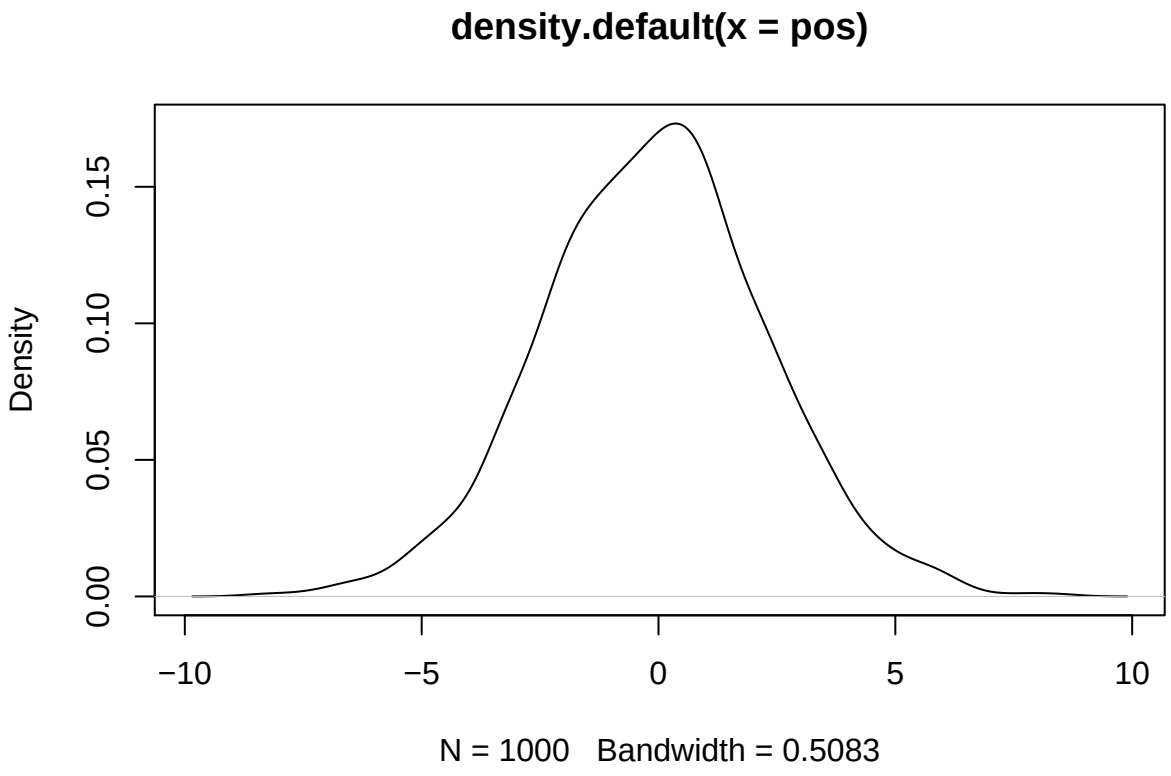
4.1 Why are normal distributions normal

4.1.1 Normal by addition

```
# R code 4.1
# Simulating coin flipping. Heads/tails -> +1/-1 step from the centre of a football pitch.

# Simulate 1000 random variables. 16 steps. Final position looks normally distributed.
pos <- replicate(1000 , sum( runif(16, -1, 1)))
hist(pos)
```




```
plot(density(pos))
```



Any process that adds together random values from the same distribution converges to normal

Conceptually, whatever the average calue of the source distribution, each sample can be thought of as a fluctuation from that average value. When we add these values together, they start to cancel each other out.

A large positive fluctuation will cancel a large negative one. The more we sum, the more chances for each deviaion from this mean will be canclled out.

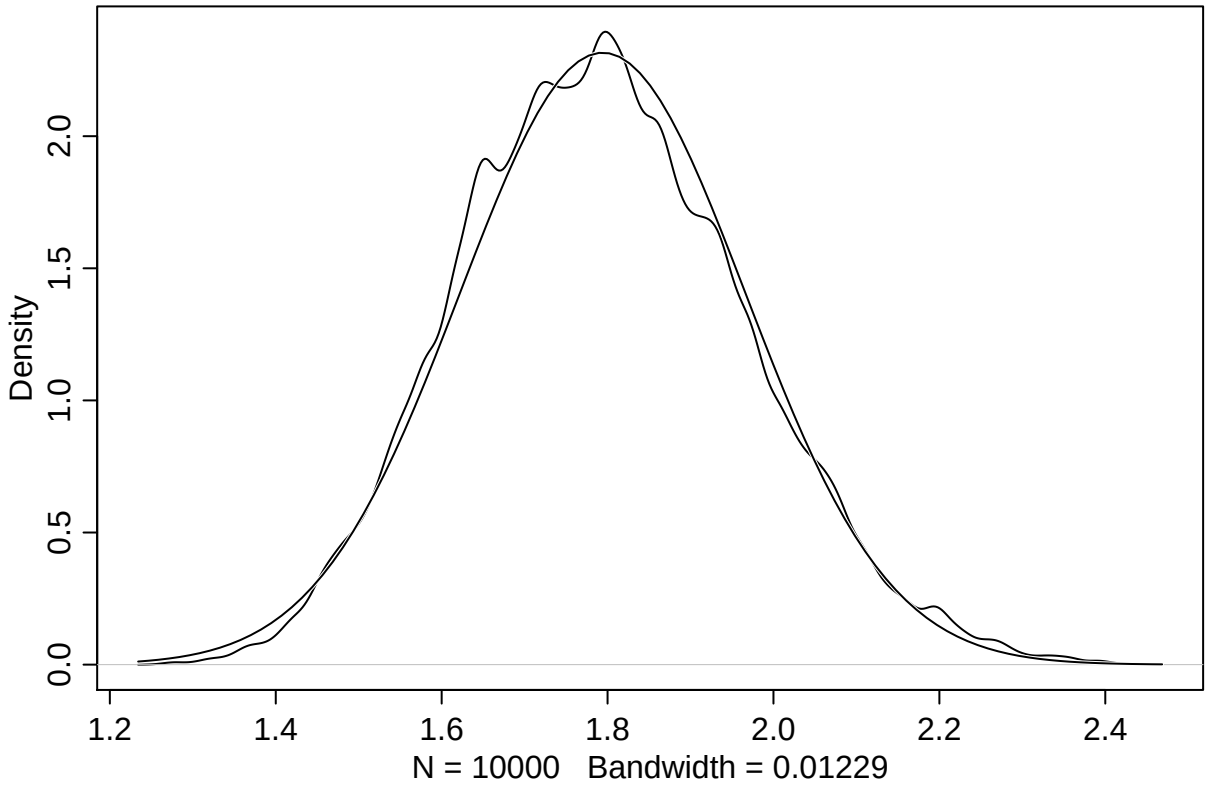
This holds for any distribution (in general). *Technically*, the distribution of sums converges to normal when the original distribution has finite variance. Practically, this means the magnitude of a newly sampled value cannot be so big as to overwhelm the previous values.

4.1.2. Normal by multiplication

```
# R code 4.2
# Taking the product of growth rates
prod(1 + runif(12, 0, 0.1))
```

```
## [1] 1.735413
```

```
# R code 4.3
# Repeat this 10,000 times
growth <- replicate( 10000 , prod( 1 + runif(12,0,0.1) ) )
dens( growth , norm.comp=TRUE ) # Approximately normal
```



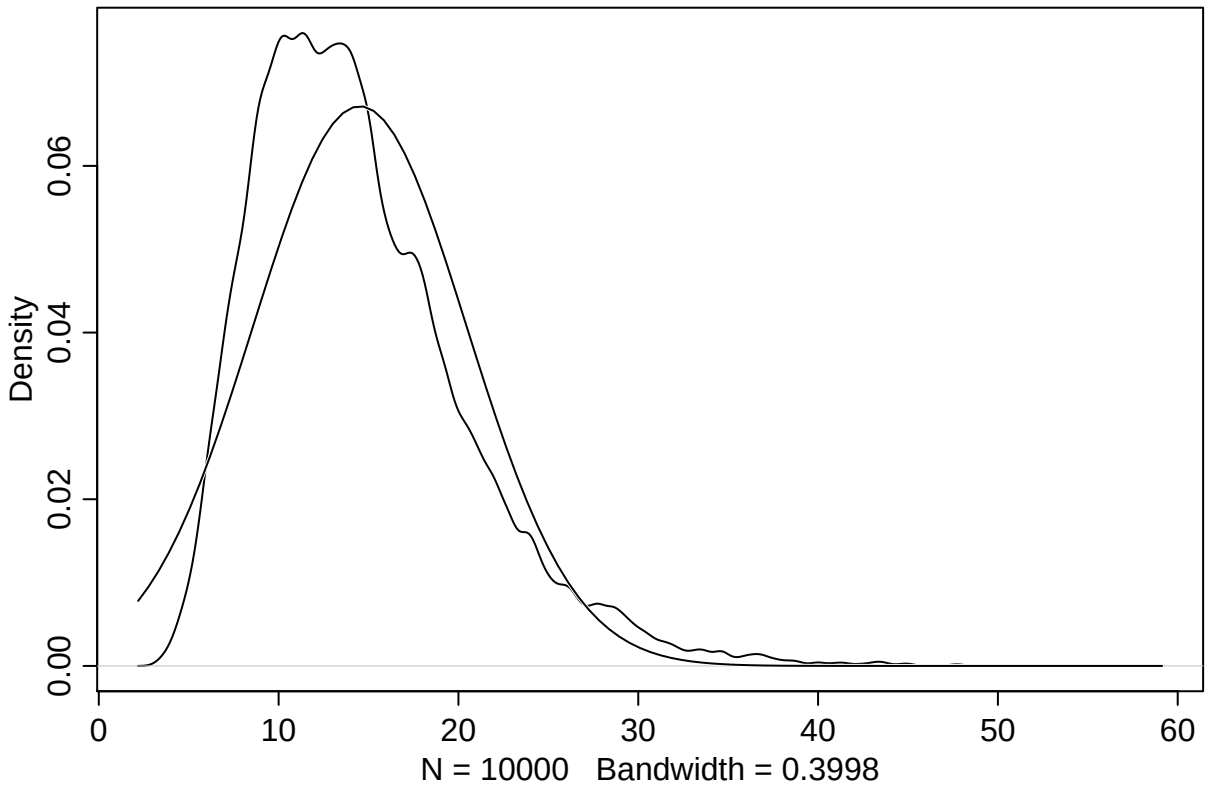
Multiplying small numbers is approximately the same as addition

$$1.1 \times 1.1 = 1.211.1 \times 1.1 = (1 + 0.1)(1 + 0.1) = 1 + 0.2 + 0.01 \approx 1.2$$

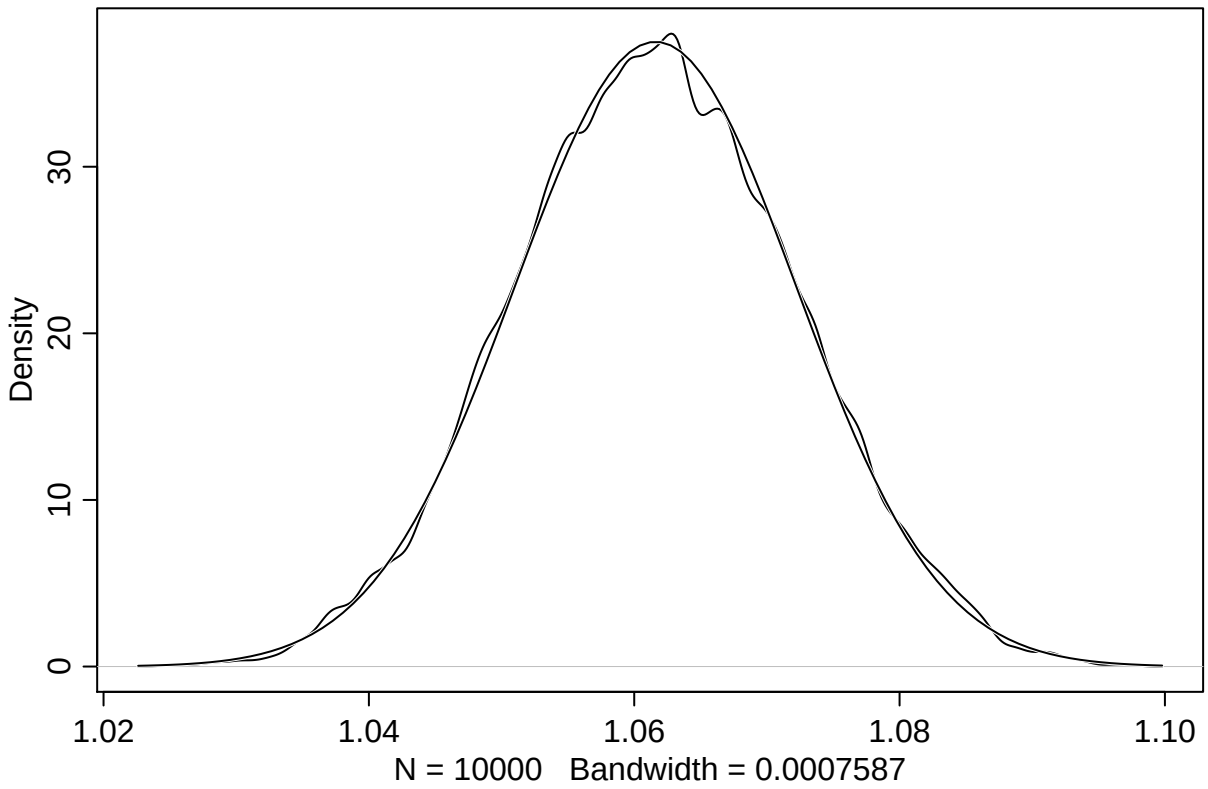
THe smaller the growth rate, the better the additive approximation will be. Small effects that multiply together are approximately additive,so they tend to stabilise on Gaussian distributions.

```
# R code 4.4
# Small is a better approximation of the normal distribution
big <- replicate( 10000 , prod( 1 + runif(12,0,0.5) ) )
```

```
small <- replicate( 10000 , prod( 1 + runif(12,0,0.01) ) )
dens(big , norm.comp=TRUE)
```



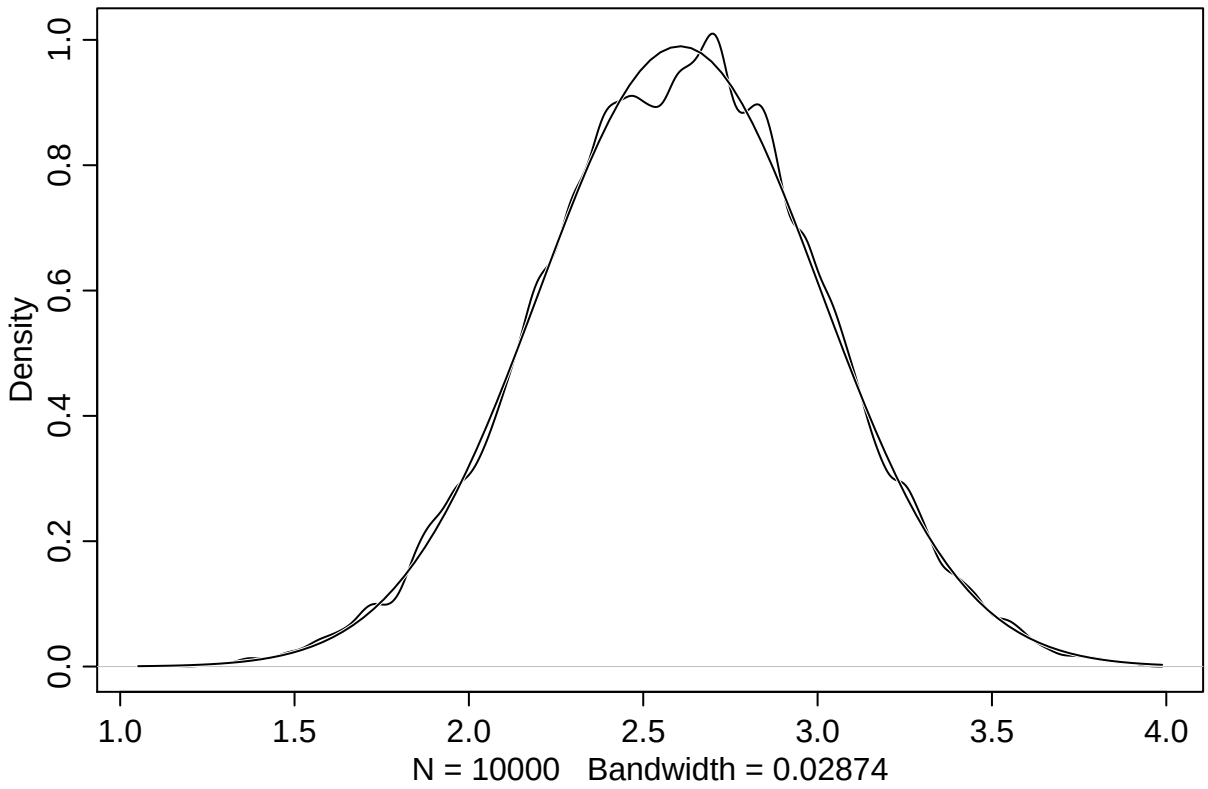

```
dens(small , norm.comp=TRUE)
```



4.1.3. Normal by log-multiplication

Large deviates that are multiplied together tend to produce Gaussian distributions on the log scale.

```
# R code 4.5
log.big <- replicate(10000, log(prod(1 + runif(12, 0, 0.5))))
dens(log.big, norm.comp = TRUE)
```



4.1.4. Using Gaussian distributions - Ontological and Epistemological justification

4.1.4.1 Ontological justification

Ontological - Knowledge that we observe Epistemological - Knowledge that we build/create

Quotes

Stripping away information away about uncertainty leads to over confidence.