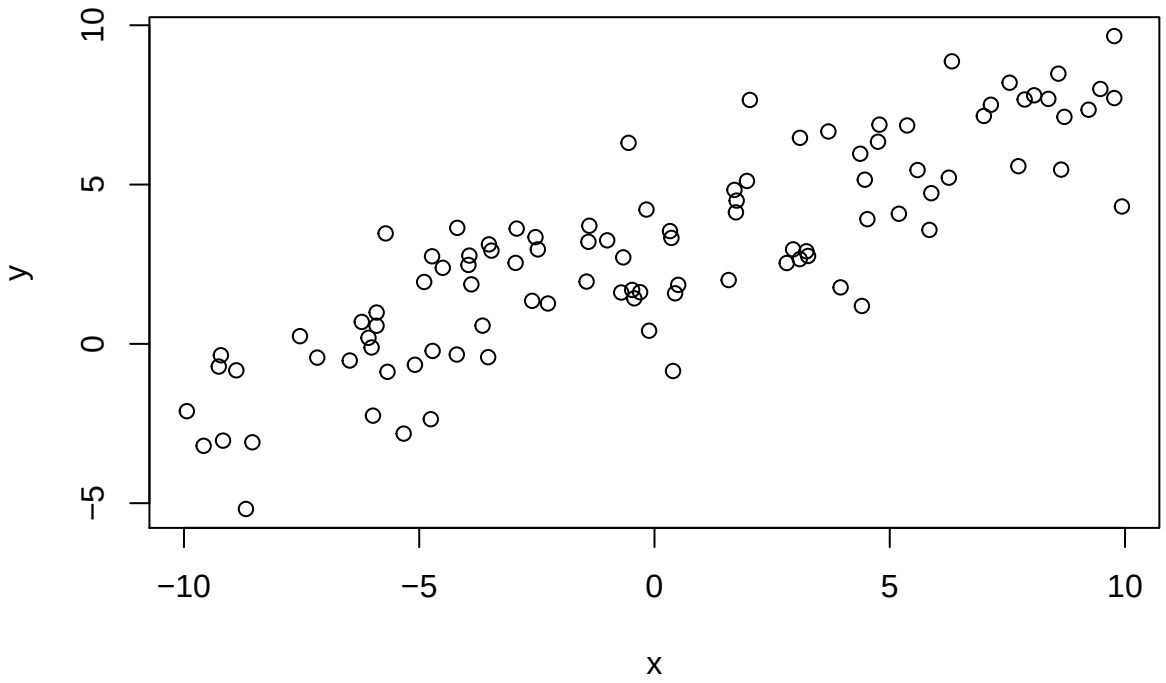
Learning Maximum Likelihood & Bayesian Inference

Let's try to understand how we can estimate statistical parameters using Maximum Likelihood and Bayesian Inference.

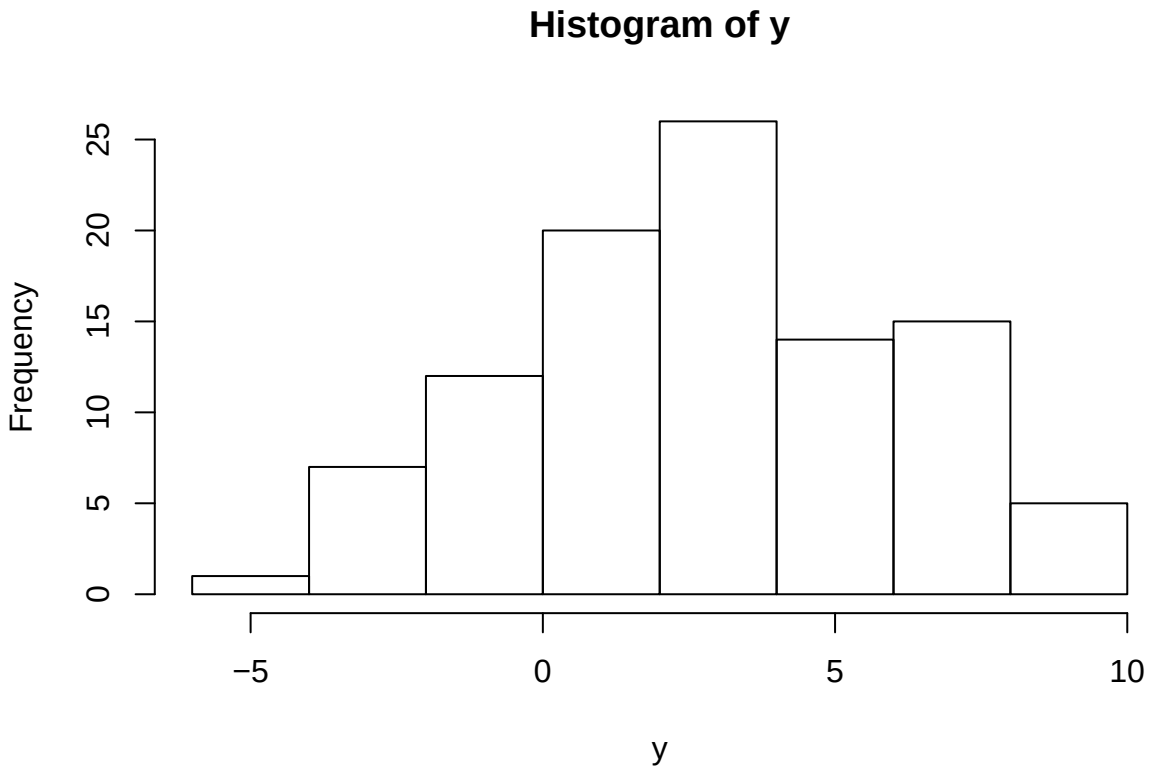
Now let's turn to a linear regression. Let's first generate data that follows a standard linear regression model, which should be familiar to you as $y = m * x + b + \epsilon$. x can take on any set of values, while m and b are regression parameters. ϵ is a random error term that we will model as a normal distribution with mean $\mu = 0$ and σ^2, which will be estimated from the data. Let's begin by simulating the data.

```
n <- 100 # The number of data points to simulate
m <- 0.5 #Pick a value for m, this is the slope of the regression
b <- 3 #Pick a vlue for b, this is the intercept
sig2 <- 3 #Pick a [positive] value for sigma^2, this is the variance of the normal error
x <- runif(n, -10, 10)
y <- m*x + b + rnorm(n, 0, sqrt(sig2))
plot(x,y)
```



Now, we could estimate this regression using R's built in tools, but we want to learn how to do it ourselves using Maximum Likelihood. What is a likelihood? It is simply the probability of observing the data given certain parameters $P(data|parameters)$. Our goal for maximizing the likelihood is to simply search for the parameters that make $P(data|parameters)$ as big as it can possibly be. How do we calculate that probability? Let's start by assuming that $m = 0$ and $b = 0$, i.e. that the data are simple draws from a normal distribution. Then we could use the R function *dnorm*, which provides the *probability density* of a particular data point.

```
#?dnorm ##Get help
## Calculate the probability of observing y for sigma^2 = 1
hist(y)
```




```
L <- dnorm(y, 0, 1)
L
```

```
##   [1] 1.968028e-01 7.430369e-14 5.735864e-04 7.252980e-10 1.622833e-05
##   [6] 1.579863e-02 1.271658e-07 6.829591e-07 9.135800e-03 4.907784e-03
##  [11] 5.780191e-14 3.358157e-18 8.379880e-07 2.498099e-11 1.438141e-01
##  [16] 3.386543e-01 3.771959e-01 6.969338e-02 7.173492e-02 3.917688e-01
##  [21] 5.040784e-15 5.848809e-03 4.757427e-14 3.480361e-01 9.563725e-05
##  [26] 1.457858e-03 5.259654e-04 1.378206e-07 6.049545e-02 3.635752e-01
##  [31] 3.091601e-12 3.299940e-10 2.433945e-14 2.825298e-01 3.742542e-01
##  [36] 3.406185e-06 5.897238e-02 6.703208e-14 7.555031e-03 1.071449e-01
##  [41] 2.775956e-01 5.509873e-03 1.875383e-04 2.340354e-13 6.662860e-04
##  [46] 8.891602e-03 3.152337e-02 2.178273e-21 1.594624e-02 3.379980e-03
##  [51] 9.093364e-10 1.161479e-02 3.655917e-05 1.601417e-01 7.456974e-13
##  [56] 4.916336e-07 9.581198e-17 3.963871e-01 3.216372e-01 2.441534e-02
##  [61] 3.664042e-01 3.966378e-03 3.095056e-01 1.856000e-02 4.926393e-03
##  [66] 1.030497e-15 4.133670e-04 5.358683e-02 7.032813e-08 1.133380e-01
##  [71] 2.356254e-03 2.457889e-01 2.097318e-11 7.581119e-04 8.975203e-11
##  [76] 1.586794e-03 5.512339e-05 9.792437e-04 3.043588e-03 2.028162e-03
##  [81] 3.800859e-12 1.788365e-01 4.279931e-02 7.388063e-09 3.656569e-01
##  [86] 3.874825e-01 3.893134e-01 1.087583e-01 3.147724e-01 7.962970e-05
##  [91] 5.875070e-07 9.522646e-02 1.003413e-02 3.393205e-01 2.381989e-03
##  [96] 8.293893e-02 5.539265e-06 8.598743e-03 2.310789e-02 2.714840e-01
```

Notice that many numbers are VERY SMALL. Computationally, it works better if we take the log of the probability density

```
logLs <- log(L)
```

Now we take the sum of this number to get the log-Likelihood of the data given m=0, b=0 and sigma^2 = 1.

```
logL <- sum(logLs)
logL
```

```
## [1] -1018.863
```

However, we know these are not the parameters we used to generate these data. What's the log likelihood under those parameters? Well we simply need to define for each data point, it's expectation and variance. In other words, we need to calculate the residuals from the model, then plug them into dnorm.

```
expected_values <- m*x + b
residuals <- y - expected_values
logL <- sum(dnorm(residuals, 0, sqrt(sig2), log=TRUE))
logL
```

```
## [1] -193.6599
```

Notice that while the log-likelihood is still small, it is substantially larger than it was previously. But is it the maximum? Let's make a function of the likelihood to streamline calculating the likelihood for a set of parameters.

```
logL_fx <- function(parameters, x, y){
  m <- parameters[1]
  b <- parameters[2]
  sig2 <- parameters[3]
  expected_values <- m*x + b
  residuals <- y - expected_values
  logL <- sum(dnorm(residuals, 0, sqrt(sig2), log=TRUE))
  return(logL)
}
```

Verify that it returns the same value as before:

```
logL_fx(c(m, b, sig2), x, y)
```

```
## [1] -193.6599
```

Now we can use computational tools to *optimize* the best likelihood. These are "hill-climbing" algorithms.

```
starting_values <- c(0, 0, 1) #Our initial guesses for the parameters
optim(par=starting_values, fn=logL_fx, control=list(fnscale=-1), x=x, y=y)
```

```
## $par
## [1] 0.4873369 2.9284027 2.7997773
##
## $value
## [1] -193.3759
##
## $counts
## function gradient
##      126       NA
##
## $convergence
## [1] 0
##
## $message
## NULL
```

Compare to R's built-in regression estimation:

```
lm1 <- lm(y~x)
summary(lm1)
```

```
##
## Call:
## lm(formula = y ~ x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.9713 -1.2358  0.1781  1.2113  3.7409
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.92762    0.16903   17.32   <2e-16 ***
## x            0.48736    0.03075   15.85   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.69 on 98 degrees of freedom
## Multiple R-squared:  0.7193, Adjusted R-squared:  0.7165
## F-statistic: 251.2 on 1 and 98 DF,  p-value: < 2.2e-16
```

```
sqrt(sig2)
```

```
## [1] 1.732051
```

Likely, the biggest difference will be in the residual standard error. This is because Maximum Likelihood estimators are biased estimators, whereas the algorithm used in R is an unbiased estimator of the variance. Nevertheless, this bias will get smaller and smaller as more data are added.

Congratulations! That's all there is to ML estimation. In phylogenetics, we do the same thing, except instead of m, b and σ^2, we are estimating τ (the topology of the tree), ν (a vector of branch lengths), and Q (a matrix of transition probabilities) and our data are gene sequences or phenotypes.

What about Bayesian analyses?

Suppose you're an alien visiting earth for the first time. Not knowing any borders or habits of the people who live there, you randomly chose a particular human to suck up and analyze. You find that this particular human is wearing what they call a cowboy hat. You also learn of a people known as "Texans" that often wear cowboy hats. You hypothesize that given the observed data (human X is wearing a cowboy hat) that human X is a Texan. So long as $P(Cowboyhat|Texan) > P(Cowboyhat|anyotherstate)$, then the Maximum Likelihood conclusion that you should make is that the human you obtained is in fact, a Texan (even if Wyomingites and Montanans are only slightly less likely to wear cowboy hats). Suppose that the rate of cowboy hat wearing in Texas is 10%.

But this is odd. What we really want to know is what is the probability that our human is in fact, a Texan. Instead, ML has given us "What are the most likely people to wear cowboy hats?". Mathematically, the rules of conditional probability relate these two quantities for us:

$$P(Cowboyhat|Texan)P(Texan) = P(Texan|Cowboyhat)P(Cowboyhat)$$

So we figured out how to maximize $P(Cowboyhat|Texan)$, but we really want $P(Texan|Cowboyhat)$. We can do some algebra to figure out what we need. It turns out it is: $P(Texan|Cowboyhat) = \frac{P(Texan)P(Cowboyhat|Texan)}{P(Cowboyhat)}$

This is Bayes Formula.

Let's go through each of these terms. $P(Texan)$ is the probability that a random human from the world is Texan. We know this value, it is approximately 28.3 million/7.6 billion or ~0.003724. What's the Probability of someone wearing a cowboy hat? That's a harder number to estimate. But we could go around estimating it for each state/country, e.g. $P(cowboyhat|Virginian)$ and $P(cowboyhat|Norwegian)$ wouldn't be too big, while $P(cowboyhat|Montanan)$, $P(cowboyhat|Argentinian)$ and $P(cowboyhat|Wyomingite)$ might be a bit bigger. Of course for each one, we would have to also weight them by their population.

In the end, we'd get this: $P(Texan|Cowboyhat) = \frac{0.003724}{P(cowboyhat|Montanan)*P(Montanan)+P(cowboyhat|Virginian)*P(Virginian)+P(cow}$

Notice that the denominator is a daunting quantity to calculate. Let's instead limit our world to a few regions. The alien knows that the human was sucked up from either Texas, Montana, California, or Virginia with populations of 28.3 million, 1.1 million, 39.5 million and 8.5 million respectively. Furthermore, let's assume they have cowboy hat wearing at rates of 10%, 9%, 1% and 0.01%. Under ML, it's clear that Texans are most likely to wear cowboy hats. However, let's plug it into Bayes Formula.

```
pop <- c("Texas"=28.3, "Montana"=1.1, "California"=39.5, "Virginia"=8.5)
totalpop <- sum(pop)
pState <- pop/totalpop
cat("\n Prior probabilities of being an inhabitant of each state\n")
```

```
##
##  Prior probabilities of being an inhabitant of each state
```

```
pState
```

```
##      Texas    Montana California   Virginia
## 0.36563307 0.01421189 0.51033592 0.10981912
```

Now calculate the Likelihoods for each state.

```
liks <- c("Texas"=10, "Montana"=9, "California"=1, "Virginia"=0.01)/100
```

Followed by the denominator of Bayes Formula, or the marginal probability of wearing a cowboy hat.

```
denom <- pState * liks
```

Put it all together in Bayes Formula:

```
post <- (liks['Texas']*pState['Texas'])/sum(denom)
post
```

```
##     Texas
## 0.8511662
```

This is the *posterior probability* that someone is a Texan if you observe they are wearing a cowboy hat. We can see that our certainty of them being Texan went up from our prior probability:

```
#Prior probability
pState["Texas"]
```

```
##     Texas
## 0.3656331
```

```
## Increase in posterior over prior
post/pState["Texas"]
```

```
##    Texas
## 2.327925
```

Furthermore, our human being is still the most probably from Texas. However, suppose that ALL Montanans wear Cowboy hats. What is our posterior probability of our person being a Cowboy then? Under Maximum

Likelihood, we would definitely conclude that our human was a Montanan. However, under Bayesian Inference we would STILL conclude that they were most likely a Texan

```
liks["Montana"] <- 1
denom <- sum(pState*liks)
post <- pState['Texas']*liks['Texas']/denom
post
```

```
##     Texas
## 0.6542067
```

Why? Well because we knew beforehand that they were very unlikely to be a Montanan! Compare the prior and posterior of being from each state:

```
posts <- pState*liks/denom
results <- data.frame("prior"=round(pState,2), "likelihood"=round(liks,2), "posterior"=round(posts,2))
results
```

```
##            prior likelihood posterior
## Texas       0.37       0.10      0.65
## Montana     0.01       1.00      0.25
## California  0.51       0.01      0.09
## Virginia    0.11       0.00      0.00
```

The posterior probabilities went up a lot for us having a Montanan! 25x greater. However, even if every Montanan wears a cowboy hat, we know there are so few of them that it is more likely they come from another, more populous state.

This is the heart of Bayesian Analyses, with the main difference being that we include a prior. Sometimes those priors are well-justified. For example, our priors here are based on knowledge of human population sizes. However, if you didn't have that available, the results would end up looking almost identical to Maximum Likelihood analysis:

```
pState_uninformative <- rep(0.25,4)
denom2 <- sum(pState_uninformative*liks)
posts2 <- pState_uninformative*liks/denom2
results_uninformative <- data.frame("prior"=pState_uninformative, "likelihood"=round(liks,2), "posterio
results_uninformative
```

```
##            prior likelihood posterior
## Texas       0.25       0.10      0.09
## Montana     0.25       1.00      0.90
## California  0.25       0.01      0.01
## Virginia    0.25       0.00      0.00
```

So Bayesian analysis does a better job of answering what questions we probably want answers to, but can be harder to apply for two reasons: 1) The prior may be hard to justify and 2) the denominator of Bayes' formula is hard to figure out! For (1), we can either try to justify our prior using data, or we can see whether our conclusions are *prior sensitive* by trying a range of plausible prior values. If our results are unaffected by our choice of prior, then we don't have to worry! For (2), we often make use a computational trick called Markov Chain Monte Carlo.

A Simple MCMC

```
#We're going to travel randomly between all 4 states. Let's start in Texas.
gens <- 10000 # Pick how many times you want to try moving between states
states <- names(pState)
state <- rep(NA, gens)
state[1] <- "Texas"
```

```
lik <- rep(NA, gens)
prior <- rep(NA, gens)
lik[1] <- liks[state[1]]
prior[1] <- pState[state[1]]

for(i in 2:gens){
  proposed_state <- sample(states, 1) #Propose a new state at random
  num_post_state_old <- prior[i-1]*lik[i-1] # get the posterior numerator for the old state
  num_post_state_new <- pState[proposed_state]*liks[proposed_state] # get the posterior numerator for t
  if(num_post_state_new > num_post_state_old){ #If the new state has a higher posterior probability, ac
    state[i] <- proposed_state
    lik[i] <- liks[proposed_state]
    prior[i] <- pState[proposed_state]
  } else{ #If it's not higher, accept it anyway with probability proportional to the ratio of the proba
    u <- runif(1)
    a <- num_post_state_new/num_post_state_old
    if(u < a){
      state[i] <- proposed_state
      lik[i] <- liks[proposed_state]
      prior[i] <- pState[proposed_state]
    } else {
      state[i] <- state[i-1]
      lik[i] <- lik[i-1]
      prior[i] <- prior[i-1]
    }
  }
}
```

```
MCMCestimates <- table(factor(state,states))/gens
```

```
data.frame(results, "mcmcEst"=round(as.vector(MCMCestimates),2))
```

```
##            prior likelihood posterior mcmcEst
## Texas       0.37       0.10      0.65    0.66
## Montana     0.01       1.00      0.25    0.26
## California  0.51       0.01      0.09    0.09
## Virginia    0.11       0.00      0.00    0.00
```

In phylogenetics, we're just doing the same exact thing as we've done above. Instead of P(state | cowboy hat) we have P(tree | sequence/phenotypic data). Tree space is a lot bigger than 4 states, it's enormous, and the likelihoods are a bit trickier to calculate, but doable. What priors do we use in a Bayesian analysis? That's a tricky question and the subject of much debate. Ideally, we'd use informative priors. In practice, people usually use software defaults. This is bad. Don't do this. Learn about what priors mean. And when you review papers, make people 1) specify what priors they used for every parameter and 2) make people justify their use of those priors.

Breast Cancer Testing

Why are you not recommended to get a mammogram until you're under 40?

Research on breast cancer has determined the following statistics (I don't think these are actually accurate, but they convey the idea):

1% of women have breast cancer 80% of mammograms detect breast cancer when breast cancer is present 9.6% of mammograms detect breast cancer when breast cancer IS ABSENT

Imagine you received a positive mammogram test stating that breast cancer is detected. How worried should you be?

Make a statement using conditional probability that represents the probability you want to determine.

Then use Bayes' Formula to determine that conditional probability. Compare to the answer from Maximum Likelihood.