Lecture 7.2: Basic linear regression

* *This lecture is based on chapter 3 of Statistical Rethinking by Richard McElreath.*

As before, we need to load some packages and set some options prior to running any models:

```
library(rstan)
library(shinystan)
library(car)
rstan_options(auto_write = TRUE)
options(mc.cores = parallel::detectCores())
source("../utilityFunctions.R")
```

Data story

Dictyota menstrualis is a brown seaweed that produces > 250 dichyol terpenes as chemical deterrents against marine herbivores and biofouling microbes.

- When herbivore pressure is high, having higher terpene concentrations should lead to higher lifetime biomass. At least that's my story!

I have saved this data as a `.csv` file called `algae.csv`. Also, below is a function, `seaweedSim` with the code to simulate your own data if you want to play around with it.

```
# You give it the total number of desired observations, the intercept,
# the slope, and the standard deviation and it makes a dataframe for you.
# Play with it, changing parameters as you wish to see how the model
# differs. Also, in the script, terpenes is the x variable. I didn't
# include arguments to change that, but it would be easy by changing
# the mean=50 & sd=3 to whatever you want.

seaweedSim <- function(nObs, alpha, beta, sigma) {
  terpenes <- round(rnorm(nObs, mean=50, sd=3), digits=2)
  error <- rnorm(nObs, mean=0, sd=sigma)
  biomass <- alpha + beta * terpenes + error
  out <- data.frame(terpenes, biomass)
  out <- out[order(terpenes),]
  return(out)
}

set.seed(20)
algae <- seaweedSim(nObs=50, alpha=0, beta=3, sigma=12)
write.csv(algae, file = "algae.csv", row.names = FALSE)
```

```
algae <- read.csv("algae.csv")
head(algae)
```

```
  terpenes  biomass
1    41.33 139.9417
2    42.58 111.4138
3    44.42 125.6067
4    45.37 112.4937
5    45.44 150.6760
6    45.58 141.4026
```

Modeling the dependent variable with a Gaussian distribution

We will begin with a single measurement variable, `biomass`, to model as a normal distribution. There are two parameters describing the distribution's shape:

1. μ: the mean describing the central location
2. σ: the standard deviation describing the spread

Bayes and MCMC will allow us to explore a number of the most plausible distributions, each with their own μ and σ and rank them by their posterior plausability.

To define our model for biomass as normally distributed with mean μ and standard deviation σ, we need to define a prior $\Pr(\mu, \sigma)$—the *joint prior probability* for the parameters.

- For many purposes, priors are specified independently for each parameter (as we have done previously). Thus we assume $\Pr(\mu, \sigma) = \Pr(\mu)\Pr(\sigma)$.

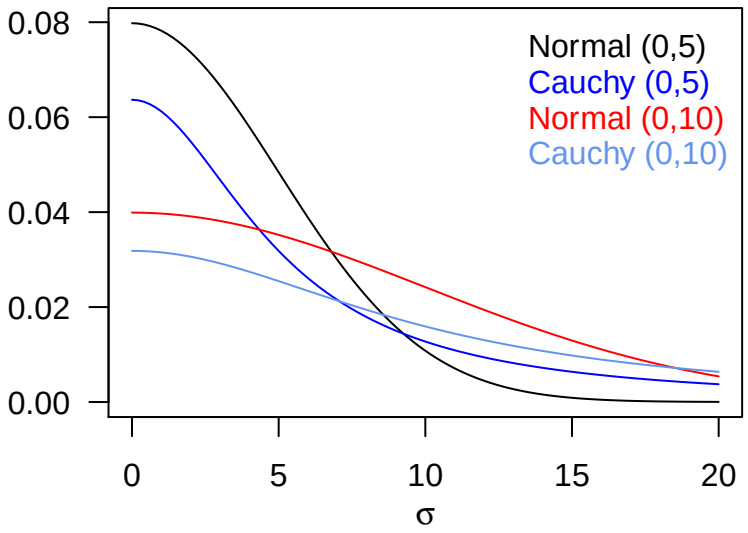
$$\begin{aligned} BM_i &\sim \text{Normal}(\mu, \sigma) \\ \mu &\sim \text{Normal}(150, 30) \\ \sigma &\sim \text{Cauchy}^{+}(0, 10) \end{aligned} \tag{1}$$

I have set the priors as follows:

- The prior for μ is weakly informative and centered on the mean of `biomass` with 95% probability the average is between 150 ± 60.
  - Later we will play with more restrictive *regularizing priors*
- The prior for σ must be positive. A uniform distibution distribution for this usually doesn't sample well from $\text{U}(0, \infty)$, and setting upper bounds on uniforms can cause issues.

- instead we will use a *half-Cauchy* distribution. This is equivalent to a folded t-distribution with $df = 1$.
  - It centers most of the probability mass around zero and therefore credible values, but has fat tails for extremes. You can play with the Cauchy with the `dcauchy` function.

```
curve(dnorm(x, 0, 5), from=0, to=20, las=1)
curve(dcauchy(x, 0, 5), add=TRUE, col="blue")
curve(dnorm(x, 0, 10), add=TRUE, col="red")
curve(dcauchy(x, 0, 10), add=TRUE, col="cornflowerblue")

mtext(text = expression(bold(sigma)), side=1, line = 2)
text(13.5, 0.075, "Normal (0,5)", font=1,cex=1, col="black", adj=c(0, 0.5))
text(13.5, 0.0675, "Cauchy (0,5)", font=1,cex=1, col="blue", adj=c(0, 0.5))
text(13.5, 0.06, "Normal (0,10)", font=1,cex=1, col="red", adj=c(0, 0.5))
text(13.5, 0.0525, "Cauchy (0,10)", font=1,cex=1, col="cornflowerblue",
  adj=c(0, 0.5))
```



We set up our model (`modMean.stan`) similarly to how we set up the simple binomial models previously, except that our data are now part of a vector.

```
data {
  int<lower=0> nObs;          // No. obs.
  vector<lower=0>[nObs] BM;   // biomass observations
  real<lower=0> muMean;       // mean of prior mu
  real<lower=0> muSD;         // SD of prior mu
  real<lower=0> sigmaSD;      // scale for sigma
}

parameters {
  real mu;
```

```
  real<lower=0> sigma;
}

model {
  mu ~ normal(muMean, muSD);
  sigma ~ cauchy(0, sigmaSD);

  BM ~ normal(mu, sigma);
}
```

Lets set up the data and look at the simplest model first:

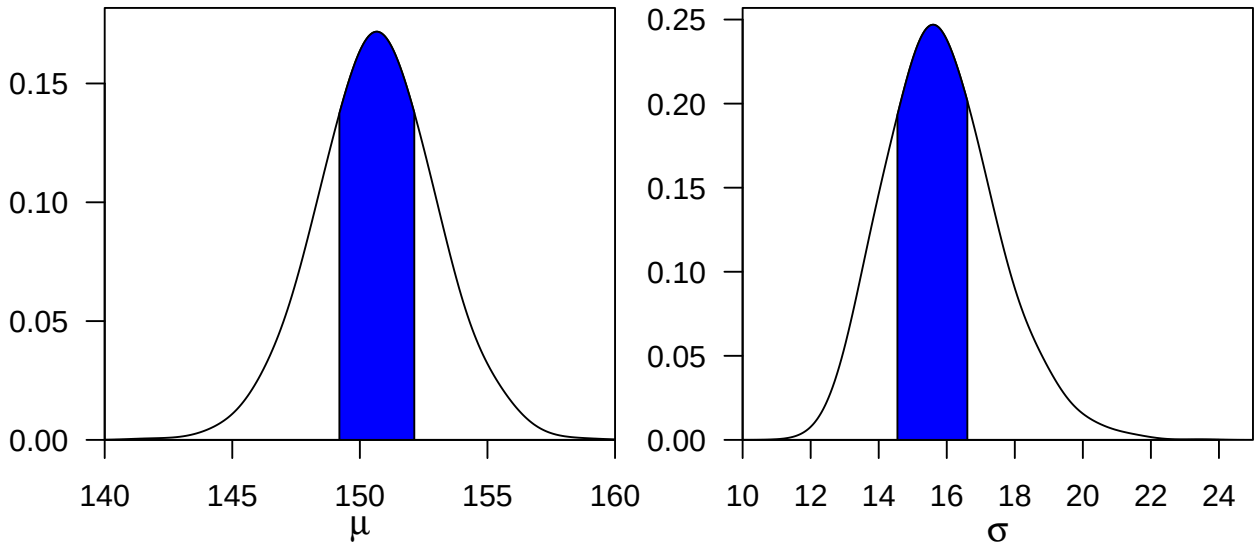
```
dat <- list(nObs=dim(algae)[1], BM=algae$biomass, muMean=150,
            muSD=30, sigmaSD=10)

intMod <- stan(file="modMean.stan", data=dat, iter=2000, chains=4, seed=3)

parMod <- as.data.frame(intMod, pars=c("mu", "sigma"))
```

```
print(intMod, pars=c("mu", "sigma"), digits.summary=2)
```

```
Inference for Stan model: modMean.
4 chains, each with iter=2000; warmup=1000; thin=1;
post-warmup draws per chain=1000, total post-warmup draws=4000.

        mean se_mean   sd   2.5%    25%    50%    75%  97.5% n_eff Rhat
mu    150.65    0.04 2.30 146.13 149.17 150.65 152.17 155.23  3244    1
sigma  15.86    0.03 1.59  13.14  14.74  15.74  16.83  19.29  3082    1

Samples were drawn using NUTS(diag_e) at Thu Feb 23 13:33:36 2017.
For each parameter, n_eff is a crude measure of effective sample size,
and Rhat is the potential scale reduction factor on split chains (at
convergence, Rhat=1).
```

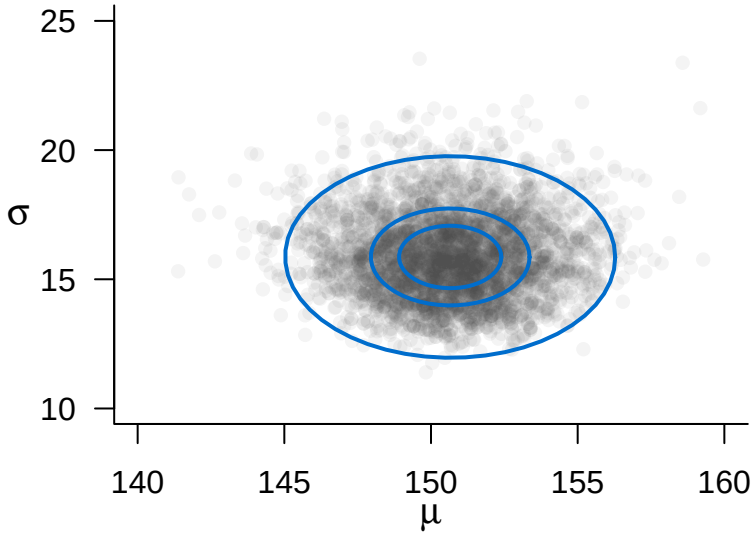
We can plot the marginal densities and 95% HDI's of μ & σ:

```
plotInterval(parMod$mu, HDI=TRUE, credMass=0.95, xlims=c(140, 160),
             col="blue", yOffset=0.01)
mtext(expression(paste(bold(mu))), side=1, line=2, cex=1.2)

plotInterval(parMod$sigma, HDI=TRUE, credMass=0.95, xlims=c(10, 25),
            col="blue", yOffset=0.01)
mtext(expression(paste(bold(sigma))), side=1, line=2, cex=1.2)
```



Or plot the joint posterior density $\Pr(\mu, \sigma)$:

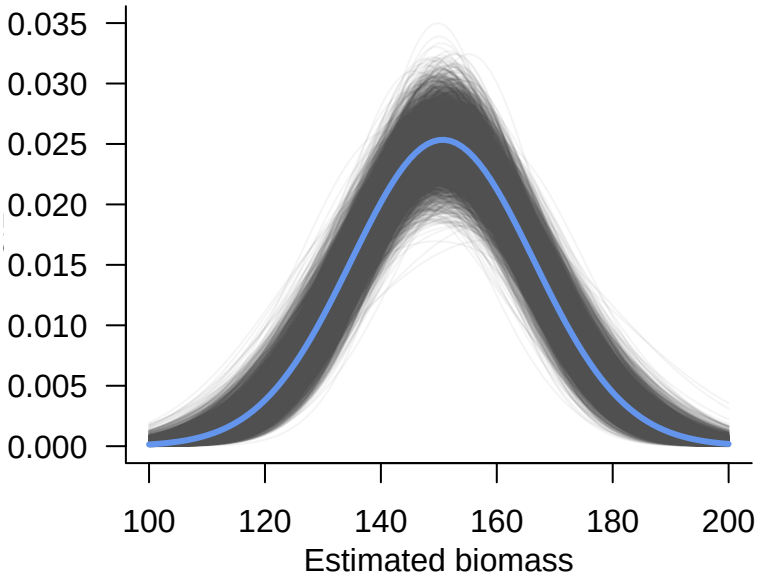
```
col <- "#50505010"
plot(parMod, pch=16, col=col, las=1, ylim=c(10,25),
      xlim=c(140,160), bty="l")
dataEllipse(as.matrix(parMod),level=c(0.25,0.5,0.95), add=TRUE, labels=FALSE,
            plot.points=FALSE, center.pch=FALSE, col=c(col,"#006DCC"))
mtext(text = expression(paste(sigma)), side=2, line=2.2, cex=1.2, las=1)
mtext(text = expression(paste(mu)), side=1, line=2, cex=1.2)
```



As before, if we want to estimate the biomass for the *Dictyota* population, we need to consider both the posterior mean and standard deviations.

```
# plot empty plot
plot(0:1,0:1, type="n", xlim=c(100, 200), ylim=c(0,0.035), las=1, bty="l")
mtext(text = "Estimated biomass", side=1, line = 2, cex=1)
```

```r
# Overlay posterior biomass densities
for (n in 1:nrow(parMod)) {
  curve(dnorm(x, parMod[n,1], parMod[n,2]), add=TRUE, col="#50505010")
}

# Overlay median posterior probability density
medBM <- apply(parMod,2,median)
curve(dnorm(x,medBM[1], medBM[2]), add=TRUE, col="cornflowerblue", lwd=3)
```



For this intercept only model, μ and σ are relatively uncorrelated.

```r
cor(parMod)
```

```
                mu       sigma
mu     1.00000000 -0.00359871
sigma -0.00359871  1.00000000
```

This can change though once we add a predictor.

Linear modeling

Modeling the average biomass in the population is interesting, but usually we are interested in how biomass changes as a function of some other variable(s), such as terpene concentration.

The simplest way to do this is with a linear model. In linear models, the predictor variable has a constant additive relationship with the mean of the outcome.

By adding terpenes (x_i), the model can be reformulated as:

$$
\begin{aligned}
BM_i &\sim \text{Normal}(\mu_i, \sigma) \\
\mu_i &= \alpha + \beta x_i \\
\alpha &\sim \text{Normal}(150, 100) \\
\beta &\sim \text{Normal}(0, 10) \\
\sigma &\sim \text{Cauchy}^{+}(0, 10)
\end{aligned} \tag{2}
$$

Now μ is no longer a paremter to be estimated but instead is a deterministic function of two parameters and a variable, where the two parameters are:

- α: the expected biomass when there are zero terpenes
- β: The expected change in biomass when terpene concentrations are increased by one unit.

We have specified a wide prior for α because it often can take a wide range of values.

The prior for beta specifies that we have no *a priori* expectation that β should be positive or negative, while accepting a relatively wide range of values.

In Stan, coding this should be easy. Try it and call the file `modLM.stan`.

```
data {
  int<lower=0> nObs;           // No. obs.
  vector<lower=0>[nObs] BM;    // biomass observations
  vector<lower=0>[nObs] terpenes;
  real<lower=0> aMean;        // mean of prior alpha
  real<lower=0> aSD;          // SD of prior alpha
  real<lower=0> bSD;           // SD of prior beta
  real<lower=0> sigmaSD;       // scale for sigma
}

parameters {
  real alpha;
  real beta;
  real<lower=0> sigma;
}

transformed parameters {
    // can be useful for plotting purposes
  vector[nObs] mu;
  mu = alpha + beta*terpenes;
}

model {
  alpha ~ normal(aMean, aSD);
```

```
  beta ~  normal(0, bSD);
  sigma ~ cauchy(0, sigmaSD);

  BM ~ normal(mu, sigma);
}
```

```
datLM <- list(nObs=dim(algae)[1], BM=algae$biomass, terpenes=algae$terpenes,
              aMean=150, aSD=100, bSD=10, sigmaSD=10)

modLM <- stan(file="modLM.stan", data=datLM, iter=2000, chains=4, seed=3)
```