Week 3 assignment: Bayesian inference

Example solutions

December 20, 2015

Problem 1

Maure et al. (2015) conducted experiments to see how ladybird beetle diet affected interactions with parasitoid wasps. Individual beetles were randomly assigned to one of five diet treatments:

1. 20 aphids per day
2. 2 aphids per day
3. 20 aphids per day + pollen
4. 2 aphids per day + pollen
5. pollen only

Each beetle was exposed to a female parisitoid wasp for 3 hours, which deposited eggs in the beetle, and the beetles were then fed each day and the outcome of the interaction monitored. The authors wanted to know whether diet affected the probability of surviving the parasitoid attack. Your task is to build a Bayesian model with Stan for beetle survival as a function of experimental treatment, estimating the probability of survival for each treatment. To keep things simple, consider the treatments to be categorical and unordered. The data are archived in Dryad here.

Important: do not manipulate the original raw data file! This is is error prone, leaves no written record, encourages spreadsheet farming, and is not reproducible. And, in this case, the data are already well organized. Read in the .xls file using the `readxl` package, then use R to massage the data as necessary.

```
## libraries
library(readxl) # use the read_excel() function to load the data
library(ggplot2)
library(rstan)
```

```
## rstan (Version 2.9.0, packaged: 2016-01-05 16:17:47 UTC, GitRev: 05c3d0058b6a)
```

```
## For execution on a local, multicore CPU with excess RAM we recommend calling
## rstan_options(auto_write = TRUE)
## options(mc.cores = parallel::detectCores())
```

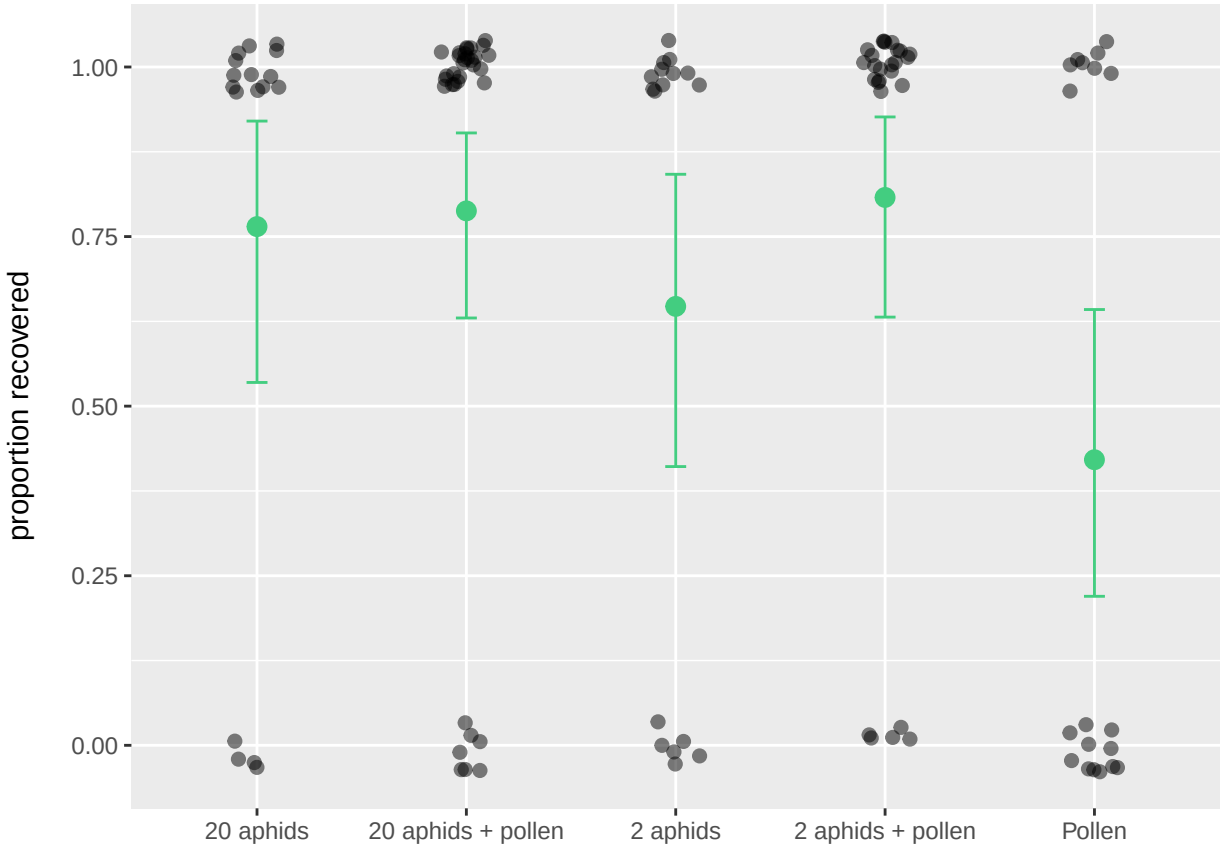
```
# tell stan to find all of my processors when fitting models
options(mc.cores = parallel::detectCores())

# load the dataset
options(warn = -1)
dat <- read_excel("data/diet_data.xlsx")
options(warn = 0)

## Cleaning the data
# manually replace spaces with underscores and capitalize column names
names(dat)[names(dat) == "Ladybird emergence date"] <- "ladybird_emergence_date"
names(dat)[names(dat) == "Parasitoid oviposition date"] <- "parasitoid_oviposition_date"
```

```r
  # ugh...there has to be a quicker way to do this!

# replace spaces with underscores using grep
names(dat) <- gsub(pattern = " ", replacement = "_", x = names(dat))

# change capitals to lowercase
names(dat) <- tolower(names(dat))
  # much better

# also, that pound sign means something else in R
names(dat)[names(dat) == "#"] <- "ID"

# lets remove the rows from the data we won't use here
dat_use <- dat[is.na(dat$ladybird_recovery) == FALSE,]

# our response variable is recovery, so make a new binary version
dat_use[dat_use$ladybird_recovery == "YES", "ladybird_recovery"] <- 1
dat_use[dat_use$ladybird_recovery == "NO", "ladybird_recovery"] <- 0
dat_use$ladybird_recovery <- as.numeric(dat_use$ladybird_recovery)

# finally, let's change diet treatment to a factor
dat_use$diet <- factor(dat_use$diet)

## Working with the data

# calculate the proportion that recovered for each diet treatments
recov_summ <- data.frame(diet = levels(dat_use$diet),
    prop_recov = tapply(dat_use$ladybird_recovery, dat_use$diet,mean))
rownames(recov_summ) <- seq(1:nrow(recov_summ)) # cleaner

# for kicks, lets calculate maximum likelihood estimates and 95% CIs
ml <- glm(ladybird_recovery ~ 0 + diet, data = dat_use, family = "binomial")

# now adding it back to our summary data frame
recov_summ <- data.frame(recov_summ,
  ml = plogis(ml$coef),
  lwr = plogis(suppressMessages(confint(ml)[,"2.5 %"])),
  upr = plogis(suppressMessages(confint(ml)[,"97.5 %"])))
recov_summ
```

```
##                 diet prop_recov        ml       lwr       upr
## 1          20 aphids  0.7647059 0.7647059 0.5349128 0.9202509
## 2 20 aphids + pollen  0.7878788 0.7878788 0.6299712 0.9027663
## 3           2 aphids  0.6470588 0.6470588 0.4109249 0.8418251
## 4  2 aphids + pollen  0.8076923 0.8076923 0.6312839 0.9263602
## 5             Pollen  0.4210526 0.4210526 0.2196991 0.6424414
```

```r
  # good, the ml estimates match the calculate averages

## Visualizing the data

# now we'll plot the mean proportions in each category along with the raw data
ggplot(data = dat_use, aes(diet,ladybird_recovery)) +
```

```
  geom_point(size = 2,
    alpha = 1/2,
    position = position_jitter(width = 0.3, height = 0.1)) +
  geom_point(data = recov_summ,
      aes(diet, ml),
      size = 3,
      color = "seagreen3") +
  geom_errorbar(data = recov_summ,
    aes(y = ml, ymin = lwr, ymax = upr),
    color = "seagreen3",
    width = 0.1) +
  labs(y = "proportion recovered\n") +
  theme(axis.title.x = element_blank())
```



Write out your model using LaTeX:

$y_i \sim Bernoulli(p_j[i])$
$p_j \sim Beta(1, 1)$

This should model the recovery (yes/no) as a bernoulli random variable with an underlying probability p_j, which varies across the J treatments. The probabilities themselves are given a uniform prior between zero and one, reflecting my complete lack of a prior knowledge about ladybird recovery

Paste your Stan model statement in the code block below, and ensure that your written model matches the notation in your Stan file:

```
data {
```

```
  // integer inputs
  int n;                          // the number of samples
  int n_treat;                    // the number of treatments

  // integer vector inputs
  int<lower=0, upper=1> y[n];   // probability of recovery (zeros and ones)
  int<lower=1, upper=n_treat> treatment[n];  // vector of treatments (1 to n_treat)

}

parameters {

  // probabilities (p) of recovery
  vector<lower=0, upper=1>[n_trt] p; // p is a real number between 0 and 1
}

model {

  // define prior for p
  p ~ beta(1, 1);

  // define the likelihood
  for (i in 1:n)
    y[i] ~ bernoulli(p[treatment[i]]); // calculate y for each element of y[n]

}
```

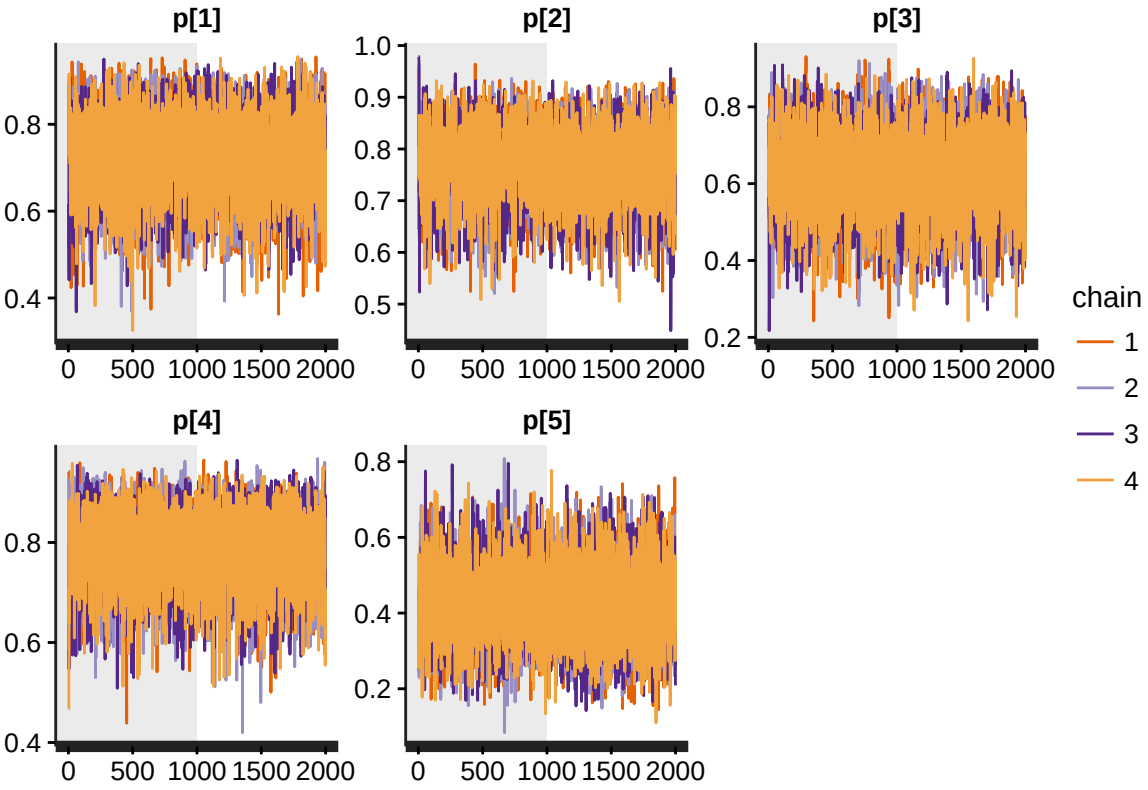
Now, use `rstan` to fit your model. Evaluate convergence by inspecting the $\hat{R}$ statistic and the traceplots.

```
# make a list that mirrors the input in the stan model
input <- list(n = nrow(dat_use),
              n_treat = length(levels(dat_use$diet)),
              y = dat_use$ladybird_recovery,
              treatment = as.numeric(dat_use$diet))
  # don't forget the we defined treatment as a number in the stan model!

# fit the model
mod <- stan('bern_mod_will.stan',
  data = input)

# check Rhat using model ouput
mod
```

```
## Inference for Stan model: bern_mod_will.
## 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
## p[1]   0.73    0.00 0.10   0.53   0.67   0.74   0.80   0.90  3139    1
## p[2]   0.77    0.00 0.07   0.62   0.72   0.77   0.82   0.89  2877    1
## p[3]   0.63    0.00 0.11   0.40   0.56   0.64   0.71   0.83  2925    1
## p[4]   0.78    0.00 0.08   0.62   0.73   0.79   0.84   0.91  2918    1
## p[5]   0.43    0.00 0.11   0.22   0.35   0.43   0.50   0.64  3046    1
## lp__ -73.77    0.04 1.61 -77.70 -74.60 -73.43 -72.58 -71.61  1664    1
```

```
##
## Samples were drawn using NUTS(diag_e) at Fri Feb  5 19:22:03 2016.
## 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).
```

```
  # looks good (Rhat all < 1.1)

# check traceplots including warmup samples
traceplot(mod, inc_warmup = TRUE)
```




```
  # that's what convergence looks like
```

Calculate posterior credible intervals, medians, means, and modes for the survival probabilities for each treatment. Hint: posterior draws can be extracted with the `rstan::extract` function, which returns a list of arrays.

```
library(modeest)  # allows me to calculate the mode of a continous distibution
```

```
##
## This is package 'modeest' written by P. PONCET.
## For a complete list of functions, use 'library(help = "modeest")' or 'help.start()'.
```

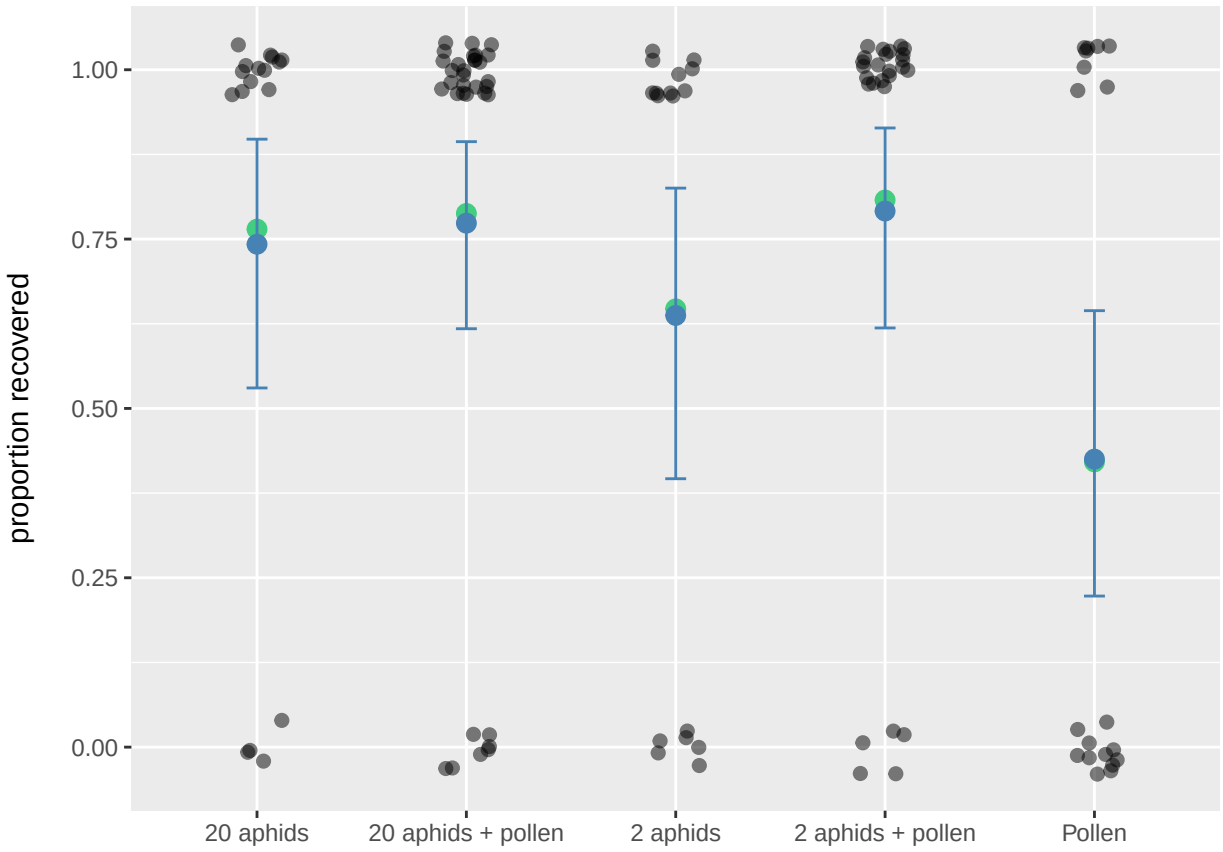
```
# extract the posterior samples
posts <- extract(mod)
```

```
# pull posteriors for probabilities into data frame
prob_posts <- as.data.frame(posts$p)
colnames(prob_posts) <- levels(dat_use$diet)

# create summary table
posterior_summary <- c()   # empty table

for(i in 1:ncol(prob_posts)){
  posterior_summary <- rbind(posterior_summary,
      data.frame(diet = colnames(prob_posts)[i],
        lwr = quantile(prob_posts[,i], prob = 0.025),
        upr = quantile(prob_posts[,i], prob = 0.975),
        mean = mean(prob_posts[,i]),
        median = median(prob_posts[,i]),
        mode = mlv(prob_posts[,i], method = "mfv")$M
      ),
    make.row.names = FALSE
  )
}

# show the summary
posterior_summary
```

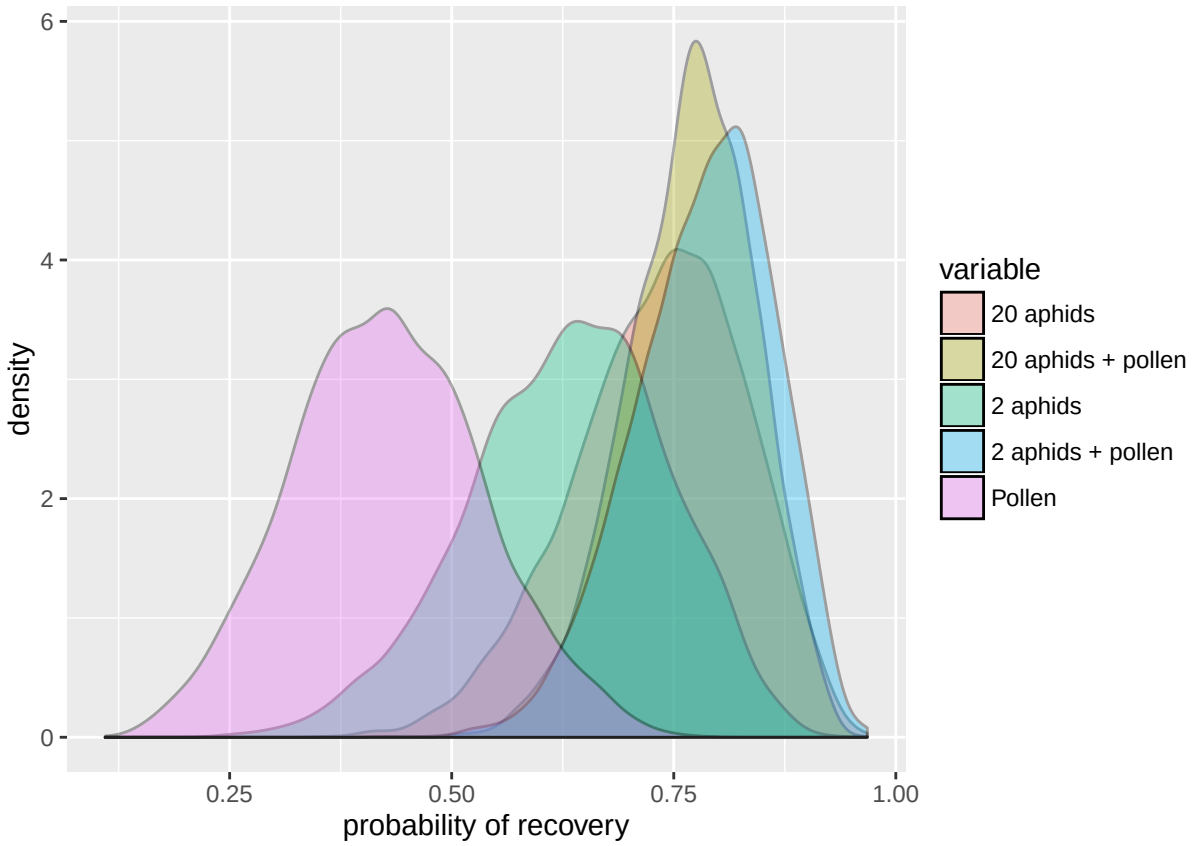
```
##                 diet       lwr       upr      mean    median      mode
## 1          20 aphids 0.5302281 0.8974886 0.7339815 0.7424182 0.7263726
## 2 20 aphids + pollen 0.6176724 0.8938449 0.7691671 0.7734920 0.7895543
## 3           2 aphids 0.3962065 0.8252377 0.6308819 0.6372536 0.6187099
## 4  2 aphids + pollen 0.6188090 0.9140066 0.7841815 0.7913828 0.7564225
## 5             Pollen 0.2230171 0.6443308 0.4266231 0.4251711 0.3791365
```

Generate a plot that shows all of the raw data along with the posterior probability distributions of recovery for each treatment:

```
## plot of raw data
ggplot(data = dat_use, aes(diet,ladybird_recovery)) +
  geom_point(size = 2, alpha = 1/2,
    position = position_jitter(width = 0.3, height = 0.1)) +
  geom_point(data = recov_summ, aes(diet, ml), size = 3, color = "seagreen3") +
  geom_point(data = posterior_summary,
    aes(diet,median),
    color = "steelblue",
    size = 3) +
  geom_errorbar(data = posterior_summary,
    aes(y = mode, ymin = lwr, ymax = upr),
    color = "steelblue",
    width = 0.1) +
  labs(y = "proportion recovered\n") +
  theme(axis.title.x = element_blank())
```




```
## plot the posterior distributions overlaid
library(reshape) # for melt function
over <- suppressMessages(melt(prob_posts))

ggplot(data = over, aes(value)) +
  geom_density(aes(fill = variable), color = "black", alpha = 1/3) +
  labs(x = "probability of recovery")
```



The authors reported statistically significant differences in ladybird beetle recovery between the diet treatments. What is your conclusion for the effect of diet on ladybird beetle recovery?

There doesn't appear to be substantial differences between treatments in the probability of recovery. There is a suggestion that having at least 2 aphids increases the probability of recovery, but the posterior distributions still overlap quite a bit. We would need more data to say one way or the other.

Inspecting the methods, results, and figure 2 (dynamite!) of Maure et al. (2015), it would appear that they actually achieved roughly the same estimates of proportions as we did. However, they used a frequentist approach of first asking whether including treatment as a predictor improved model fit versus a model of the mean only (it did, barely, at $P = 0.04$ when using a log-likelihood ratio test). Then they "performed pairwise comparisons to reveal differences among means" though they don't report what method they use to "reveal" nor its associated statistics, and only report the results with "a" and "b" lettering on figure 2.

This is a classical example of how ANOVA or analysis of deviance methods can be misleading. I like our way comparing estimated proportions much better.

Problem 2

One of the biggest advantages of Bayesian approaches is the ease with which you can make inference on **derived parameters**. For example, we might want to know which diet treatment gives the highest survival probability. In one draw from the posterior distribution, we should have five estimated probabilities. The highest probability can be recorded and stored in an object (say `best`). We can do this for each posterior draw to produce a vector of the "best" treatments (from the beetle's perspective). To find the posterior probability that each particular treatment is best, count the frequency of each treatment in the `best` vector, and divide by the total number of posterior draws. Do this below using the results from problem 1, and report the posterior probabilities.

```r
# which treatment had the highest probability of recovery for each sample
best <- apply(prob_posts, 1, which.max)
worst <- apply(prob_posts, 1, which.min)

# convert to treatments
best <- factor(levels(dat_use$diet)[best])
worst <- factor(levels(dat_use$diet)[worst])

# make sure levels match in case any treatments have prob = 0
best <- factor(best, levels = levels(dat_use$diet))
worst <- factor(worst, levels = levels(dat_use$diet))

# proportion best
p_best <- round(table(best)/length(best), digits = 2)

# proportion worst(for kicks)
p_worst <- round(table(worst)/length(worst), digits = 2)

data.frame(diet = levels(dat_use$diet),
  proportion_best = as.numeric(p_best),
  proportion_worst = as.numeric(p_worst))
```

```
##                 diet proportion_best proportion_worst
## 1          20 aphids            0.22             0.01
## 2 20 aphids + pollen            0.31             0.00
## 3           2 aphids            0.04             0.09
## 4  2 aphids + pollen            0.43             0.00
## 5             Pollen            0.00             0.89
```

Which treatment was best? What is the probability of that treatment being the best, conditional on the data?

Well the "best" treatment was 2aphids + pollen, though the probability of that treatment being best was only ~ 0.4. On the other hand, the pollen only treatment did have a ~ 0.9 probability of being the worst, so there's more suggestive evidence that not having aphids in the diet is worse than having aphids.

Problem 3

Simulate data from a normal distribution for three groups, each with a different mean. You can assume that the standard deviations for each group are equal. In generating data, use a design matrix to acquire the expected value for your data (somewhere in your code there should be `X %*% beta`).

```r
# set up
n_groups <- 3       # nubmer of groups
n <- 200             # total number of individuals

# assign individuals to groups
groups <- sample(1:n_groups,n, replace = TRUE)

# create design matrix
X <- matrix(0,nrow = n, ncol = n_groups)
for(i in 1:nrow(X)){
```

```
   X[i,groups[i]] <- 1
}

# simulate y values
beta <- c(5,10,15)      # vector of means
sigma <- 3              # standard deviations
y <- rnorm(n, mean = X %*% beta,sd = sigma)
```

Write a Stan model statement for a linear model that you can use to estimate the parameters.

```
data {

  // integer inputs
  int n; // number of individuals
  int n_groups; // number of groups

  // vector inputs
  vector[n] y; // the observed data of length y

  // matrix inputs
  matrix[n, n_groups] X; // the n x n_groups design matrix

}
parameters {

  // vector of betas
  vector[n_groups] beta;

  // value for standard deviation
  real<lower=0> sigma; // must be greater than zero
}
model {

  // priors
  beta ~ normal(0, 5);
  sigma ~ normal(0, 5);

  // likelihood
  y ~ normal(X * beta, sigma);
}
```

Use Stan to estimate the parameters of your model.

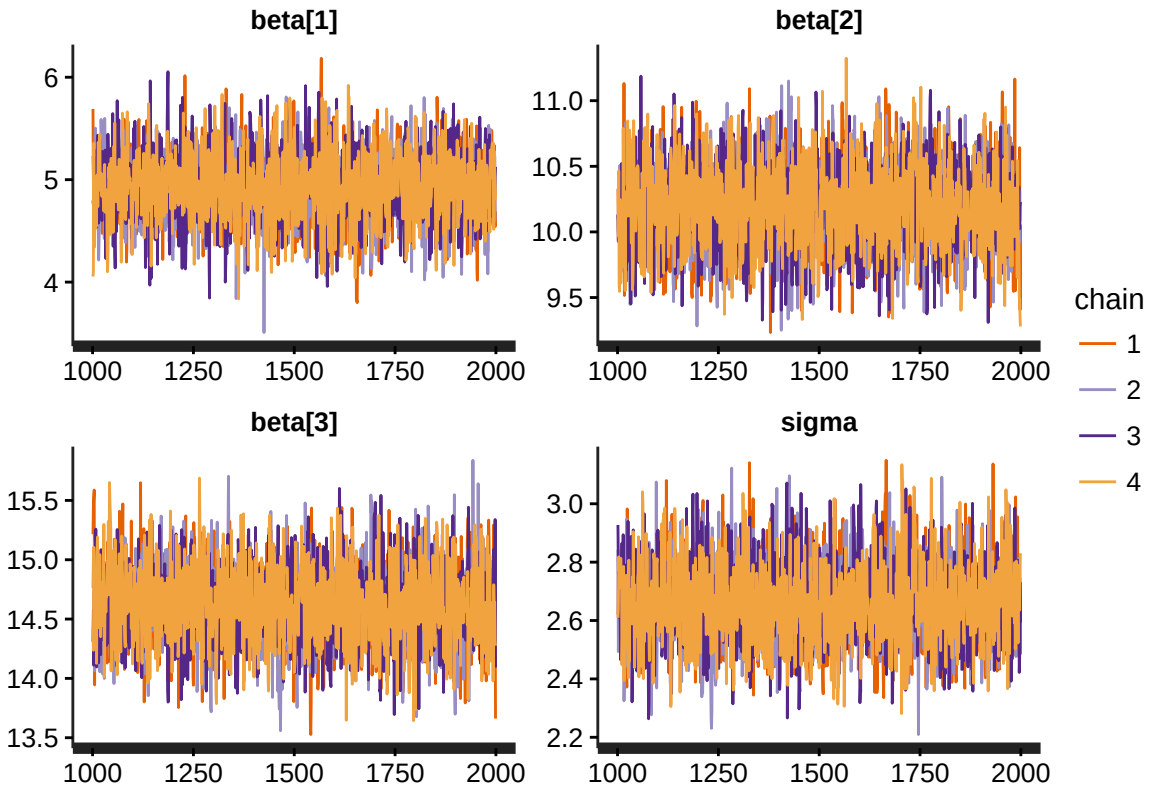
```
input <- list(n = n,
  groups = n_groups,
  X = X
)

mod2 <- stan("lm_fit_will.stan", data = input,open_progress = FALSE)
```

Assess convergence of the MCMC algorithm graphically and with the Rhat statistic.

```
mod2
```

```
## Inference for Stan model: lm_fit_will.
## 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
## beta[1]    4.93    0.01 0.33    4.26    4.70    4.92    5.15    5.58  3108
## beta[2]   10.20    0.01 0.32    9.57    9.98   10.20   10.42   10.82  2887
## beta[3]   14.63    0.01 0.33   13.98   14.41   14.63   14.85   15.26  2895
## sigma      2.66    0.00 0.14    2.40    2.56    2.65    2.75    2.95  2414
## lp__    -301.25    0.04 1.45 -304.94 -301.94 -300.94 -300.18 -299.41  1356
##         Rhat
## beta[1]    1
## beta[2]    1
## beta[3]    1
## sigma      1
## lp__       1
##
## Samples were drawn using NUTS(diag_e) at Fri Feb  5 19:22:43 2016.
## 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).
```

```
traceplot(mod2)
```



Plot the marginal posterior distributions for each parameter with a vertical line at the true value.

```
## extract posteriors
posts <- data.frame(b = extract(mod2)$beta,
  s = extract(mod2)$sigma)
names(posts) <- c("Group 1 mean","Group2 mean","Group 3 mean","SD")

## melt
posts_melted <- suppressWarnings(melt(posts))
```

```
## Using  as id variables
```

```
## convert parameters to data frame
params <- data.frame(variable = names(posts),
  value = c(beta,sigma))

## plot
ggplot(data = posts_melted, aes(value)) +
  geom_density(fill = "grey50") +
  facet_wrap(~variable) +
  geom_vline(data = params, aes(xintercept = value),
    color = "black", lty = "dashed")
```