HW1

1.

My stan model:

```
data {
  real<lower=0> r; //ball radius
  real<lower=0> R; //hole radius
  int<lower=0> I; //#data points
  real<lower=R-r> x[I]; // distance(feet)
  int<lower=0> n[I]; // #tries
  int<lower=0> y[I]; // #successes
}
parameters {
  real<lower=0> sigma;
}
transformed parameters {
  real<lower=0,upper=1> theta[I];
  real as[I];
  for (i in 1:I)
    as[i] <- asin((R - r) / x[i])  / sigma;
  for (i in 1:I)
    theta[i] <- 2 * normal_cdf(as[i], 0, 1) - 1;
}
model {
  y ~ binomial(n, theta);
}
```

And code for running it:

```
library ("rstan")
setwd("~/Documents/BDA/Homework 1")

## Sebastian's function to run stan with caching of compiled Stan models
stan_run <- function(stanModel, ...) {
  if(class(stanModel) == "stanfit") {
    stanExe <- stanModel
  } else {
    stanModel.rda <- gsub("stan$", "rda", stanModel)
    if(!file.exists(stanModel.rda) || file.info(stanModel.rda)$mtime < file.info(stanModel)$mtime) {
      cat("Model",stanModel,"needs recompilation.\n")
      args <- modifyList(list(...), list(file=stanModel, iter=0, warmup=0, chains=0))
      stanExe <- do.call(stan, args)
      saveRDS(stanExe, file=stanModel.rda)
    } else {
      cat("Loading cached stan model", stanModel, ".\n")
      stanExe = readRDS(stanModel.rda)
    }
  }
  # This bit with the seed is for debugging purposes; once we figure out why Stan is crashing R we can
  seed <- sample.int(.Machine$integer.max, 1)
```

```
  write (seed, file="stan_seed.txt")
  stan(fit=stanExe, seed=seed, ...)
}

putting_data <- read.table ("putting_data.txt", header=TRUE)
x <- as.vector(putting_data[[1]])
n <- as.vector(putting_data[[2]])
y <- as.vector(putting_data[[3]])
I <- nrow(putting_data)
r <- 1.68/24
R <- 4.25/24
data <- c("I","x","n","y","R","r")

fit <- stan_run("putting_model.stan", data=data, chains=4, iter=2000)
```

```
print(fit, pars = "sigma")
```

```
## Inference for Stan model: putting_model.
## 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
## sigma 0.03       0  0 0.03 0.03 0.03 0.03  0.03  1320    1
##
## Samples were drawn using NUTS(diag_e) at Fri Sep 18 09:10:24 2015.
## 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).
```

2. (a) A noninformative prior would be uniform over [-100, 100] since that is the entire space of possible changes in success probability\ (b) My best knowledge suggests that improvements will be positive and small. This could maybe be captured by a truncated gamma with appropriate parameters?\ (c) A uniform prior over [0, 100] since we expect practice to improve ability

3. (a)

```
sim_doc_office <- function() {
  patients <- rexp(1, rate = 1/10)
  while(sum(patients) < 540)
  {
    patients <- append(patients, rexp(1, rate = 1/10))
  }
  patients <- patients[-length(patients)]
  num_patients <- length(patients)
  wait_times <- vector(mode = "numeric", length = num_patients)
  doctors <- c(0, 0, 0)
  arrival <- 0
  for(p in 1:num_patients)
  {
    arrival <- arrival + patients[p]
    d <- which.min(doctors)
    wait_times[p] <- max(doctors[d] - arrival, 0)
```

```
    doctors[d] <- arrival + wait_times[p] + runif(1, min = 5, max = 20)
  }
  num_patients <- length(patients)
  num_wait <- sum(wait_times > 0)
  mean_wait <- mean(wait_times)
  close <- max(max(doctors) - 540, 0)
  return(list("num_patients" = num_patients, "num_wait" = num_wait, "mean_wait" = mean_wait, "close" = c
}
results <- sim_doc_office()
cat("Number of patients is: ", results$num_patients)
```

```
## Number of patients is:  52
```

```
cat("Number of patients who had to wait is: ", results$num_wait)
```

```
## Number of patients who had to wait is:  5
```

```
cat("Average wait time in minutes: ", results$mean_wait)
```

```
## Average wait time in minutes:  0.2351623
```

```
cat("Time the doctors office closed: ", results$close, " minutes after 4PM")
```

```
## Time the doctors office closed:  6.212267  minutes after 4PM
```

```
samples <- 100
num_patients <- vector(mode = "numeric", length = samples)
num_waits <- vector(mode = "numeric", length = samples)
mean_waits <- vector(mode = "numeric", length = samples)
closes <- vector(mode = "numeric", length = samples)
for(i in 1:samples)
{
  results <- sim_doc_office()
  num_patients[i] <- results$num_patients
  num_waits[i] <- results$num_wait
  mean_waits[i] <- results$mean_wait
  closes[i] <- results$close
}
cat("Median number of patients: ", median(num_patients))
```

```
## Median number of patients:  54.5
```

```
cat("50% interval is: ", c(quantile(num_patients)[2], quantile(num_patients)[4]))
```

```
## 50% interval is:  48 58
```

```
cat("Median number of waits is : ", median(num_waits))
```

```
## Median number of waits is :  7
```

```r
cat("50% interval is: ", c(quantile(num_waits)[2], quantile(num_waits)[4]))
```

```
## 50% interval is:  5 10
```

```r
cat("Median average wait time is : ", median(mean_waits))
```

```
## Median average wait time is :  0.5011502
```

```r
cat("50% interval is: ", c(quantile(mean_waits)[2], quantile(mean_waits)[4]))
```

```
## 50% interval is:  0.2653582 0.787862
```

```r
cat("Median closing time: ", median(closes), " minutes after 4PM")
```

```
## Median closing time:  6.216906  minutes after 4PM
```

```r
cat("50% interval is: ", c(quantile(closes)[2], quantile(closes)[4]))
```

```
## 50% interval is:  0 10.58991
```