Bayes assignment on regression

Sean van der Merwe

27 May 2020

Currently marking student 2012345678

Instructions

Your individual task with this assignment is to construct and fit a regression of the observed log distances travelled by a set of animals, based on the time of day. The animals should be considered randomly selected from a general population and that the general population is of interest (predictions should be for a random future animal).

Please approach the problem iteratively as follows:

1. Find the sheet in the given Excel file that corresponds with your student number and load only the data on that sheet.
2. Scatter Plot *LnDistance* (y axis) vs *TimeOfDay* (x axis) and make a visual judgement of whether you think there is any relationship between them. Colour each animal differently. [10]
3. Do an ordinary regression of *LnDistance* on *TimeOfDay* and *Animal*, and comment on the significance of each term in the model. Ignore the unstable variance for now. Note that to get full marks you must analyse the ANCOVA summary, not the ordinary summary. [10]
4. Apply some sort of transformation (continuous, piecewise, circular, polynomial, anything) to *TimeOfDay* and show that you end up with a better regression model according to BIC. You are replacing TimeOfDay with one or more new variables based on it for a better model. Try not to go overboard. [10]
5. Fit the same regression model in stan and show that you get the same coefficient and p-value for any **one** of your parameters. [15] Minor deviations due to simulation error and prior choice are fine. Tip: Use a command like summary(StanFit)$summary to get the simulation error (reported as se_mean).
6. Adapt your model to make *Animal* a random effect as it should be. [5] Check your result by comparing with the ML mixed model results (maybe using lmer in lme4 package). [5]
7. Add an estimated fit line [5] and 95% prediction interval lines for a random future animal of the same type to your scatter plot. [10]
8. Adapt your stan model to also include a regression model for the standard deviation. Keep it sensible. Give a trace plot and histogram for any of the new parameters you've added. [10]
9. Add the new prediction interval to your scatter plot to show that you've improved the fit. [5]

The last 15 marks will be awarded competitively. The best submission will get +15, second best +14,..., second worst +1, worst +0. NB: Don't go way out of bounds to get these marks. Things that will get you up the list is *loo* comparison of models, assessment of simulation effectiveness and simulation error, neat graphs (and interesting bonus graphs like circular plots for time of day), and **detailed comments**. Things that will not get rewarded are fitting exotic non-Bayes models and obvious overfitting. Lastly, it's possible for everyone to get full marks: if after the marks are out you feel you did better than someone who got more bonus marks than you, then send in a logical argument why you say that and you'll get their bonus marks on top of yours (max. mark is 100 and they keep their marks too).

Generate samples that are different for each student

This is the code used to generate the data, for interest only.

```
library(openxlsx)
students <- c('2013105875', '2014142356', '2014180588', '2015040484', '2015045582', '2
015061803', '2015093376', '2015109206', '2015150408', '2015289065', '2016073500', '201
6114015', '2016152452', '2016156008', '2016161057', '2016264323', '2012345678', '21234
56789', '9876543210')
nn <- length(students)
n <- 600
datasets <- vector('list',nn)
for (i in 1:nn) {
nA <- ceiling(runif(1,4,6))
As <- paste0('A',1:nA)
gi <- sample(1:nA,n,T)
g <- As[gi]
Aeffs <- runif(nA, -0.5, 0.5)
tod <- rep(0:23, times=n/24)
mufunc <- c(4,3.8,3.6,3.3,3,5.3, 6,5.9,5.9,5.8,5.8,5.7, 5.6,5.6,5.6,5.7,5.8,6, 5,3.8,3
.9,4,4,4.2)
sigmafunc <- c(seq(2,1,-0.25),seq(1,2,l=19))
x1 <- mufunc[tod+1]
x2 <- Aeffs[gi]
y <- x1 + x2 + rnorm(n)*sigmafunc[tod+1]
datasets[[i]] <- data.frame(LnDistance=y, TimeOfDay=tod, Animal=g)
}
names(datasets) <- paste0('St__',students)
write.xlsx(datasets, file = "BayesAssignmentAnimalReg.xlsx")
rm('students','nn','datasets','i','n','g','gi','x1','x2','y','nA','As','Aeffs','tod','
mufunc','sigmafunc')
```

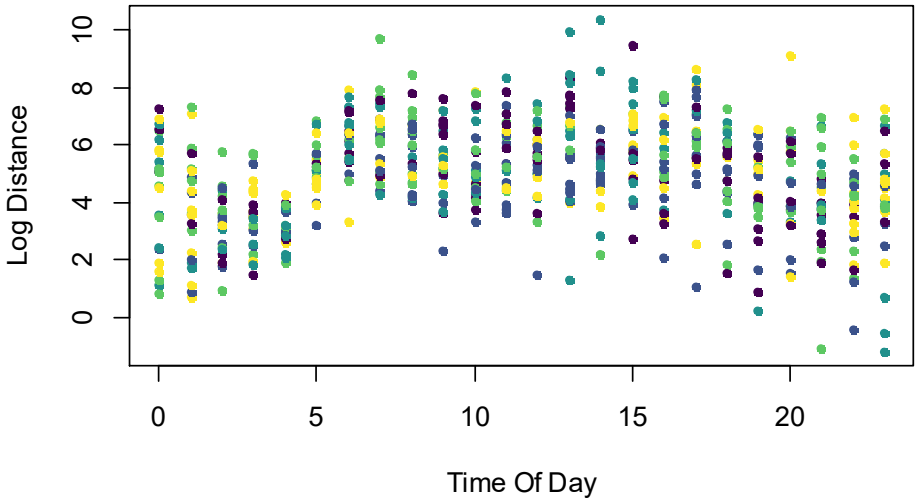
Memorandum

Read data

```
library(openxlsx)
mydata <- read.xlsx('BayesAssignmentAnimalReg.xlsx',paste0('St__',st))
names(mydata)

## [1] "LnDistance" "TimeOfDay"  "Animal"

attach(mydata)
```

Scatter plot

```
AnFac <- factor(Animal)
AnNum <- c(AnFac)
AnCount <- nlevels(AnFac)
AnNames <- levels(AnFac)
library(viridisLite)
cols <- viridis(AnCount)
plot(TimeOfDay, LnDistance, main = '', ylab = 'Log Distance', xlab = 'Time Of Day', co
l=cols[AnNum], pch=20)
```



|| 10 marks total: 5 marks for loading data and getting graph correct. 2 marks for colouring by animal. 3 marks for stating that the distances are not stable across time of day. ||

Regression

```
anova(m1 <- lm(LnDistance~TimeOfDay+AnFac))

## Analysis of Variance Table
##
## Response: LnDistance
##             Df  Sum Sq Mean Sq F value Pr(>F)
## TimeOfDay    1    4.28  4.2849  1.3431 0.2470
## AnFac        4   19.83  4.9565  1.5536 0.1852
## Residuals  594 1895.03  3.1903
```

|| 4 marks for doing a single regression including all the correct terms. 2 marks for giving the ANCOVA summary. 4 marks for a discussing the significance of the terms (probably not significant). ||

Add time of day transformation

There are many options and not one correct answer here. I'm giving a simple option for now.

Proposal: One average distance from 6:00 to 18:59 (calling it Daytime) and a different average distance from 19:00 to 5:59.

```
Daytime <- (TimeOfDay>=6)&(TimeOfDay<19)
anova(m2 <- lm(LnDistance~Daytime+AnFac))

## Analysis of Variance Table
##
## Response: LnDistance
##             Df  Sum Sq Mean Sq  F value    Pr(>F)
## Daytime      1  463.35  463.35 193.9963 < 2.2e-16 ***
## AnFac        4   37.04    9.26   3.8766  0.004058 **
## Residuals  594 1418.75    2.39
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```
BIC(m1,m2)
```

```
##    df      BIC
## m1  7 2437.541
## m2  7 2263.866
```

|| 5 marks for implementing a sensible transformation. 5 marks for showing that BIC goes down. ||

Stan

```
library(parallel)
library(rstan)
mycores <- max(1,floor(detectCores(logical = FALSE)*0.75))
options(mc.cores = mycores)
rstan_options(auto_write = TRUE)

// This Stan block defines a regression model with one grouping variable and one expla
natory variable, by Sean van der Merwe, UFS
data {
  int<lower=0> n;                         // number of observations
  real y[n];              // observations
  int<lower=0> k;                         // number of groups
  int<lower=0, upper=k> g[n];     // group membership
  real x1[n];              // Explanatory variable 1
}
// The parameters of the model
parameters {
  real<lower=0> sigma;            // standard deviation
  real animal[k];                 // per animal intercept
  real beta1;                     // slope of x1
}
transformed parameters {
  real mu[n];                     // the expected values (linear predictor)
  for (i in 1:n) {                // every observation has a different model
    mu[i] = animal[g[i]]+beta1*x1[i];     // mean model
  }
}

// The model to be estimated.
model {
  y ~ normal(mu, sigma);       // fit the data pattern
}

saveRDS(AnimalModel1, file = 'AnimalModel1.Rds')

n <- length(LnDistance)
stan_data <- list(n=n, y=LnDistance, k=AnCount, g=AnNum, x1=Daytime+0)
ModelFit1 <- sampling(AnimalModel1, stan_data, pars=c('sigma','animal','beta1'), iter
= 20000, chains = mycores)
draws1 <- extract(ModelFit1)
summary(ModelFit1)$summary
```

```
##                  mean      se_mean         sd        2.5%         25%
## sigma        1.548633 0.0001527839 0.04523724    1.462363    1.517753
## animal[1]    4.069946 0.0006638590 0.16479339    3.748720    3.958031
## animal[2]    3.494313 0.0006731530 0.15851657    3.185025    3.386913
## animal[3]    3.991583 0.0006713272 0.16654979    3.664849    3.879631
## animal[4]    4.183634 0.0005892860 0.14787558    3.892770    4.084373
```

```
## animal[5]    4.086244 0.0005912318 0.15190247    3.787294    3.983559
## beta1        1.815861 0.0006455683 0.12808242    1.562914    1.729940
## lp__      -561.263320 0.0113828893 1.87750388 -565.786720 -562.287897
##                  50%         75%       97.5%    n_eff      Rhat
## sigma        1.547403    1.578301    1.640343 87667.22 0.9999902
## animal[1]    4.070283    4.181900    4.391956 61620.87 0.9999426
## animal[2]    3.493893    3.600598    3.804940 55452.58 0.9999745
## animal[3]    3.991697    4.104184    4.315287 61548.81 1.0000557
## animal[4]    4.183021    4.283752    4.472730 62971.00 0.9999603
## animal[5]    4.086826    4.188029    4.382939 66010.66 0.9999681
## beta1        1.815592    1.902121    2.066260 39363.58 0.9999765
## lp__      -560.939878 -559.880644 -558.605248 27205.50 1.0002113
```

```
pvalfunc <- function(sims,target=0) { 2*min(mean(sims<target),mean(sims>target)) }
cat('\nThe coefficient for DayTime for the ordinary fit is about',round(coef(m2)[2],4)
,'(p-value about',round(coef(summary(m2))[2,4],4),') while the same coefficient from t
he Stan fit is',round(mean(draws1$beta1),4),'(p-value about',round(pvalfunc(draws1$bet
a1),4),').\n')
```

```
##
## The coefficient for DayTime for the ordinary fit is about 1.8155 (p-value about 0 )
while the same coefficient from the Stan fit is 1.8159 (p-value about 0 ).
```

Random effects model

In the random effects model we still have a different intercept for each animal but now they are assumed to vary randomly around a general intercept, according to some unknown variance.

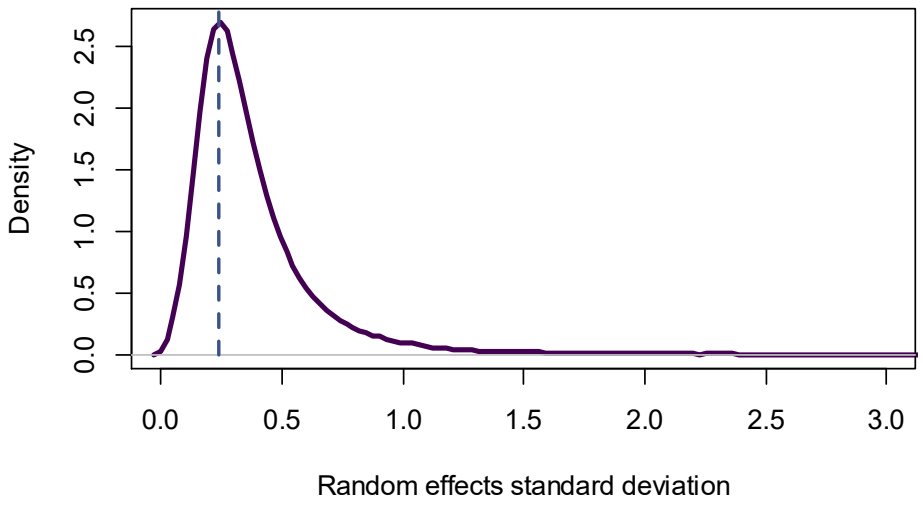
```
// This Stan block defines a mixed effects model with one grouping variable and one ex
planatory variable, by Sean van der Merwe, UFS
data {
  int<lower=0> n;                         // number of observations
  real y[n];              // observations
  int<lower=0> k;                         // number of groups
  int<lower=0, upper=k> g[n];     // group membership
  real x1[n];              // Explanatory variable 1
}
// The parameters of the model
parameters {
  real<lower=0> sigma;            // standard deviation
  real animal[k];                 // per animal intercept
  real beta1;                     // slope of x1
  real beta0;                     // general intercept
  real<lower=0> sigmaanimal;      // between animal sd
}
transformed parameters {
  real mu[n];                     // the expected values (linear predictor)
  for (i in 1:n) {                // every observation has a different model
    mu[i] = animal[g[i]]+beta1*x1[i];     // mean model
  }
}

// The model to be estimated.
model {
  y ~ normal(mu, sigma);       // fit the data pattern
  animal ~ normal(beta0, sigmaanimal); // Random effects pattern
}
```

```r
saveRDS(AnimalModel2, file = 'AnimalModel2.Rds')

ModelFit2 <- sampling(AnimalModel2, stan_data, pars=c('sigma','animal','beta1','beta0'
,'sigmaanimal'), iter = 20000, chains = mycores, control=list(adapt_delta=0.99))

## Warning: There were 56 divergent transitions after warmup. Increasing adapt_delta a
bove 0.99 may help. See
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup

## Warning: Examine the pairs() plot to diagnose sampling problems

draws2 <- extract(ModelFit2)
mysummary2 <- summary(ModelFit2)$summary

print(mysummary2)

##                     mean      se_mean         sd          2.5%
## sigma          1.5491681 0.0001996973 0.04526991    1.46338718
## animal[1]      4.0520233 0.0008430339 0.15314410    3.75534003
## animal[2]      3.6002954 0.0011121695 0.17108164    3.26642319
## animal[3]      3.9925602 0.0008465240 0.15259605    3.69313851
## animal[4]      4.1453247 0.0007862838 0.14121756    3.87358350
## animal[5]      4.0660251 0.0007487536 0.14321833    3.78755698
## beta1          1.8042613 0.0008080723 0.12771811    1.55376307
## beta0          3.9727739 0.0020354535 0.24038856    3.52438020
## sigmaanimal    0.3863855 0.0027899335 0.33110027    0.09405414
## lp__        -558.8101831 0.0186070797 2.41577190 -564.49575685
##                     25%          50%         75%       97.5%    n_eff
## sigma          1.5178871    1.5481182    1.578965    1.640940 51389.54
## animal[1]      3.9490201    4.0502534    4.152896    4.357260 32999.74
## animal[2]      3.4840840    3.5987033    3.714257    3.939494 23662.73
## animal[3]      3.8904272    3.9926984    4.094922    4.293075 32494.36
## animal[4]      4.0490612    4.1436191    4.240266    4.426567 32256.61
## animal[5]      3.9695928    4.0644963    4.161662    4.351283 36586.38
## beta1          1.7180614    1.8043977    1.890483    2.055616 24980.70
## beta0          3.8556820    3.9730030    4.088540    4.425844 13947.78
## sigmaanimal    0.2163004    0.3107977    0.455651    1.117185 14084.17
## lp__        -560.1668909 -558.4546452 -557.054172 -555.200441 16856.03
##                  Rhat
## sigma       1.0000248
## animal[1]   1.0000994
## animal[2]   1.0001722
## animal[3]   1.0000153
## animal[4]   1.0001428
## animal[5]   0.9999823
## beta1       1.0000611
## beta0       1.0002028
## sigmaanimal 1.0003225
## lp__        1.0006231

library(lme4)
(s2 <- summary(lmer(LnDistance~Daytime+(1|AnFac))))

## Linear mixed model fit by REML ['lmerMod']
## Formula: LnDistance ~ Daytime + (1 | AnFac)
##
## REML criterion at convergence: 2234.4
##
## Scaled residuals:
```

```
##     Min      1Q  Median      3Q     Max
## -3.3760 -0.6330 -0.0206  0.6890  3.2890
##
## Random effects:
##  Groups   Name        Variance Std.Dev.
##  AnFac    (Intercept) 0.05589  0.2364
##  Residual             2.38826  1.5454
## Number of obs: 600, groups:  AnFac, 5
##
## Fixed effects:
##             Estimate Std. Error t value
## (Intercept)   3.9720     0.1414   28.10
## DaytimeTRUE   1.8024     0.1276   14.13
##
## Correlation of Fixed Effects:
##             (Intr)
## DaytimeTRUE -0.490
```

It is clear that all parameter estimates agree within two standard errors of simulation, except for the standard deviation between animal intercepts. This parameter has a skew distribution, which means that the mode, median and mean of the posterior are apart from each other. The ML estimate would be close to the the posterior mode estimate (check given below).

```
plot(density(draws2$sigmaanimal), lwd=3, col=cols[1], xlim=c(0,3), main='', xlab='Rand
om effects standard deviation')
lines(rep(attr(s2$varcor$AnFac,'stddev'),2),c(0,3),lwd=2,col=cols[2],lty=2)
```

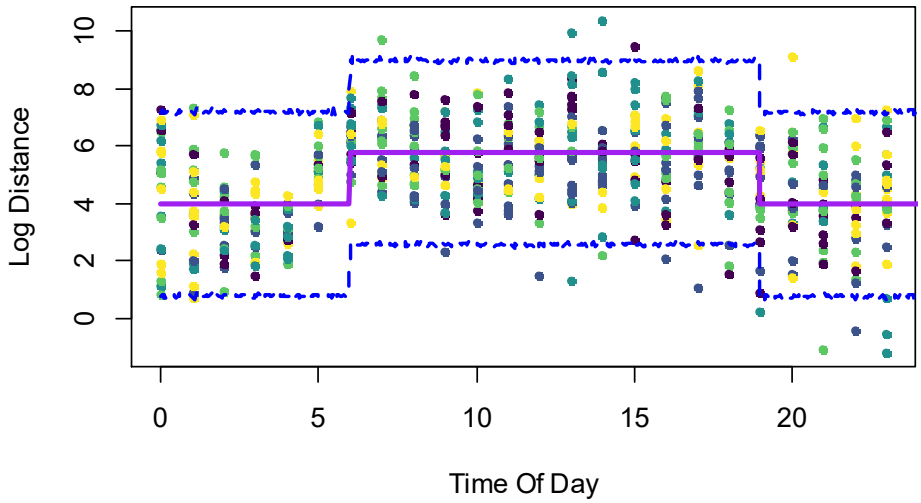


|| 5 marks for correctly implementing random effects. 5 marks for comparing summary to ML summary. ||

Predictive density

```
shortestinterval <- function(postsims,alpha=0.05) { # Coded by Sean van der Merwe, UFS
sorted.postsims <- sort(postsims)
nsims <- length(postsims)
gap <- round(nsims*(1-alpha))
widths <- diff(sorted.postsims,gap)
interval <- sorted.postsims[c(which.min(widths),(which.min(widths) + gap))]
```

```
return(interval) }


numx <- 601
xpoints <- seq(0,24,l=numx)
is.x.day <- (xpoints>=6)&(xpoints<19)
fitline2 <- mean(draws2$beta0) + mean(draws2$beta1)*(is.x.day+0)
nsims <- length(draws2$beta0)
ints <- matrix(1,2,numx)
for (i in 1:numx) {
  random.animal <- rnorm(nsims, draws2$beta0, draws2$sigmaanimal)
  daytime.adjustment <- draws2$beta1*is.x.day[i]
  random.lndistance <- rnorm(nsims, random.animal+daytime.adjustment, draws2$sigma)
  ints[,i] <- shortestinterval(random.lndistance,0.05)
}

plot(TimeOfDay, LnDistance, main = '', ylab = 'Log Distance', xlab = 'Time Of Day', co
l=cols[AnNum], pch=20)
lines(xpoints, fitline2, lwd=3, lty=1, col='purple')
lines(xpoints, ints[1,], lwd=2, lty=2, col='blue')
lines(xpoints, ints[2,], lwd=2, lty=2, col='blue')
```



|| 5 marks for fit line using general intercept. 10 marks for interval that captures correct variances. ||

Standard deviation model

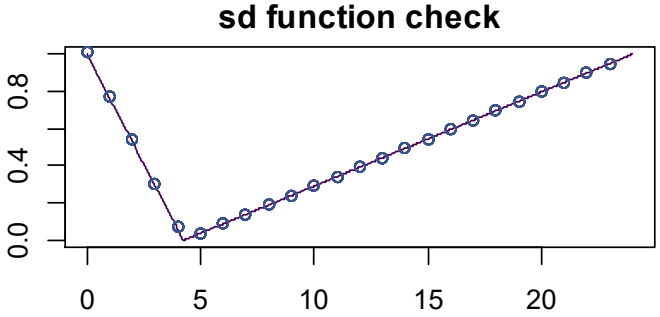
In order to make the standard deviation model simple but flexible at the same time, we say that there are minimum and maximum standard deviations which we don't know, and a function that weights them that we do know. Then we can play with the weights to get a good fit without having to recompile the model.

```
// This Stan block defines a mixed effects model with 1 group var, 1 explanatory var,
and a changing variance, by Sean van der Merwe, UFS
data {
  int<lower=0> n;                         // number of observations
  real y[n];              // observations
```

```
  int<lower=0> k;                         // number of groups
  int<lower=0, upper=k> g[n];     // group membership
  real x1[n];              // Explanatory variable 1
  real x2[n];              // Explanatory variable 2
}
// The parameters of the model
parameters {
  real<lower=0> sigma1;                     // base log standard deviation
  real animal[k];                 // per animal intercept
  real beta1;                     // slope of x1
  real beta0;                     // general intercept
  real<lower=0> sigmaanimal;      // between animal sd
  real<lower=0> sigma2;           // other log standard deviation
}
transformed parameters {
  real mu[n];                     // the expected values (linear predictor)
  real<lower=0> sigma[n];
  for (i in 1:n) {                // every observation has a different model
    mu[i] = animal[g[i]] + beta1 * x1[i];     // mean model
    sigma[i] = sigma1 * x2[i] + (sigma2 * (1 - x2[i]));  // standard deviation model
  }
}
// The model to be estimated.
model {
  y ~ normal(mu, sigma);       // fit the data pattern
  animal ~ normal(beta0, sigmaanimal); // Random effects pattern
}

saveRDS(AnimalModel3, file = 'AnimalModel3.Rds')
```

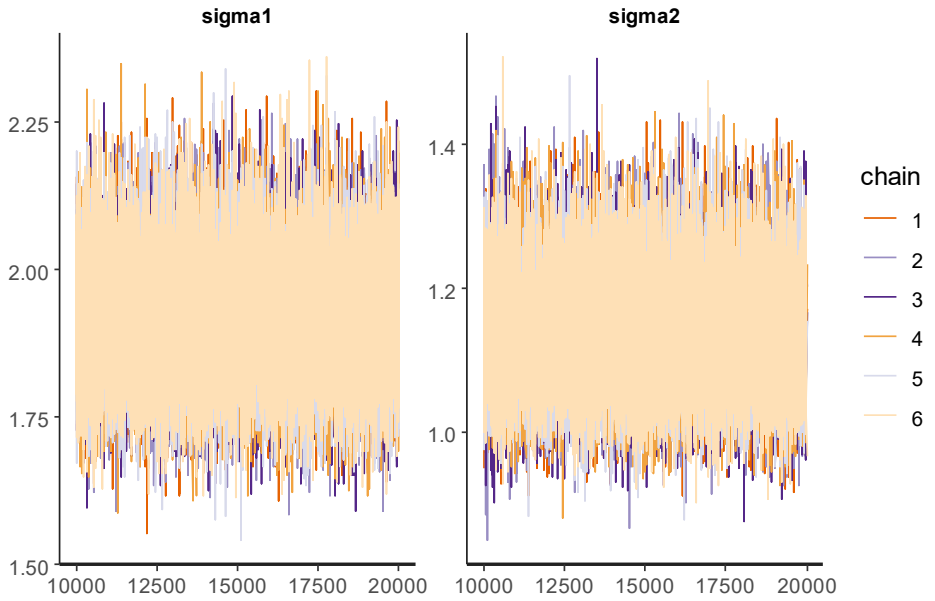
I'm going to say that the maximum sd is at midnight and call that sigma1, so midnight must have a weight of 1. Then I'm going to say the minimum sd is at 4:15 and call that sigma2, so 4:15 must have a weight of 0. I want to connect these with straight lines for simplicity.

```
cp <- 4.25 # Change Point
x2 <- rep(0,n)
x2[TimeOfDay<cp] <- 1.01 - (TimeOfDay[TimeOfDay<cp]/cp)
x2[TimeOfDay>=cp] <- (TimeOfDay[TimeOfDay>=cp]-cp)/(24-cp)
sdpoints <- rep(0,numx)
sdpoints[xpoints<cp] <- 1 - (xpoints[xpoints<cp]/cp)
sdpoints[xpoints>=cp] <- (xpoints[xpoints>=cp]-cp)/(24-cp)
par(mar=c(2,2,2,0.2))
plot(xpoints,sdpoints,type='l',main='sd function check',xlab='Time of day',ylab='',col
=cols[1])
points(TimeOfDay,x2,col=cols[2])
```

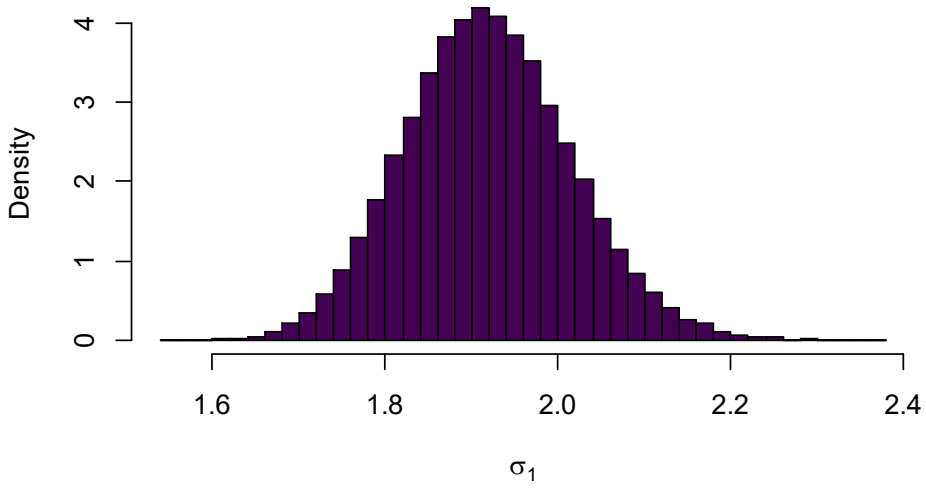



```
inits <- function() { list(sigma1=2, sigma2=1, beta0=4, beta1=1.8, sigmaanimal=0.3, an
imal=rep(4,AnCount), mu=rep(5,n), sigma=rep(1.5,n)) }
```

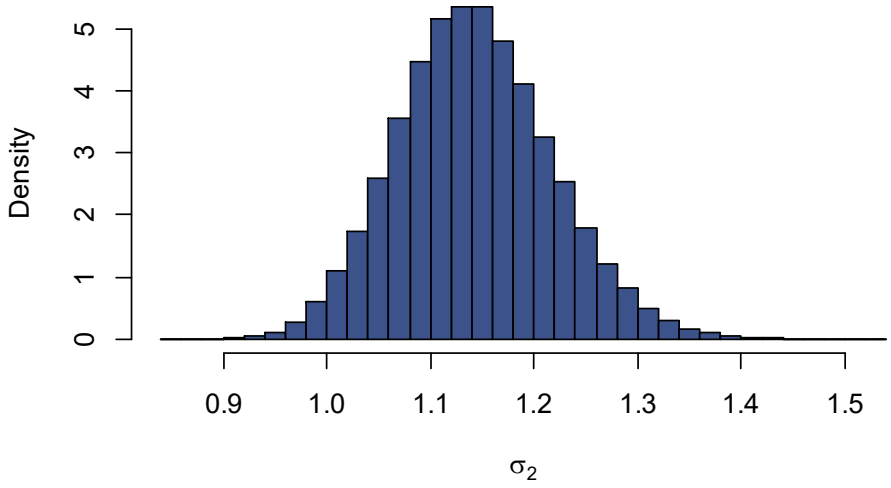
```
stan_data3 <- list(n=n, y=LnDistance, k=AnCount, g=AnNum, x1=Daytime+0, x2=x2)
ModelFit3 <- sampling(AnimalModel3, stan_data3, pars=c('sigma1','animal','beta1','beta
0','sigmaanimal','sigma2'), init=inits, iter = 20000, chains = mycores, control=list(a
dapt_delta=0.99))

## Warning: There were 43 divergent transitions after warmup. Increasing adapt_delta a
bove 0.99 may help. See
## http://mc-stan.org/misc/warnings.html#divergent-transitions-after-warmup

## Warning: Examine the pairs() plot to diagnose sampling problems

draws3 <- extract(ModelFit3)
traceplot(ModelFit3,c('sigma1','sigma2'))
```




```
hist(draws3$sigma1, 40, col=cols[1], main='', xlab = expression(sigma[1]), freq = FALS
E)
```



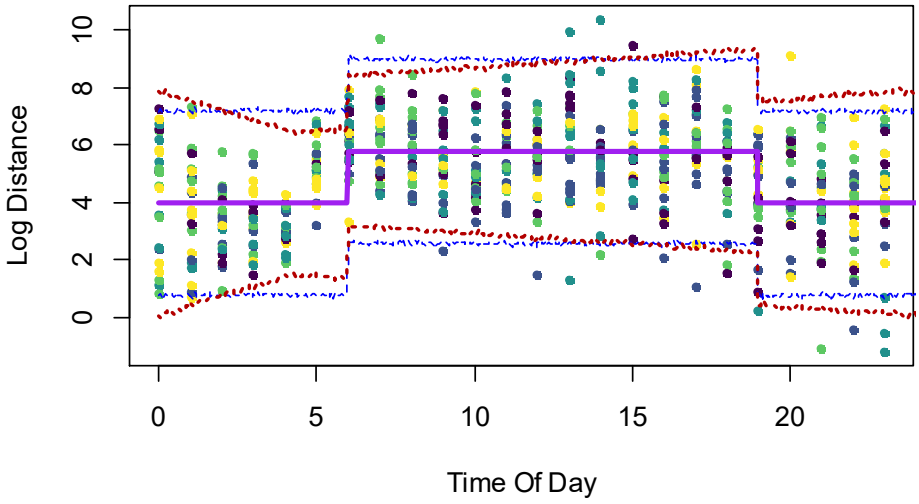
```
hist(draws3$sigma2, 40, col=cols[2], main='', xlab = expression(sigma[2]), freq = FALSE)
```



|| 7 marks for appropriately adapting model. 3 marks for trace plot and histogram. ||

Final fit

```
fitline3 <- mean(draws3$beta0) + mean(draws3$beta1)*(is.x.day+0)
nsims <- length(draws3$beta0)
ints3 <- matrix(1,2,numx)
for (i in 1:numx) {
  random.animal <- rnorm(nsims, draws3$beta0, draws3$sigmaanimal)
  daytime.adjustment <- draws3$beta1*is.x.day[i]
  sigmas <- draws3$sigma1 * sdpoints[i] + (draws3$sigma2 * (1 - sdpoints[i]));
  random.lndistance <- rnorm(nsims, random.animal+daytime.adjustment, sigmas)
  ints3[,i] <- shortestinterval(random.lndistance, 0.05)
}
plot(TimeOfDay, LnDistance, main = '', ylab = 'Log Distance', xlab = 'Time Of Day', col=cols[AnNum], pch=20)
lines(xpoints, fitline3, lwd=3, lty=1, col='purple')
lines(xpoints, ints[1,], lwd=1, lty=2, col='blue')
lines(xpoints, ints[2,], lwd=1, lty=2, col='blue')
lines(xpoints, ints3[1,], lwd=2, lty=3, col=rgb(0.7,0,0))
lines(xpoints, ints3[2,], lwd=2, lty=3, col=rgb(0.7,0,0))
```



|| 5 marks for final fit graph showing improved fit. ||

Clearly there is still some room for improvement, but hopefully that is for the reader to work on.