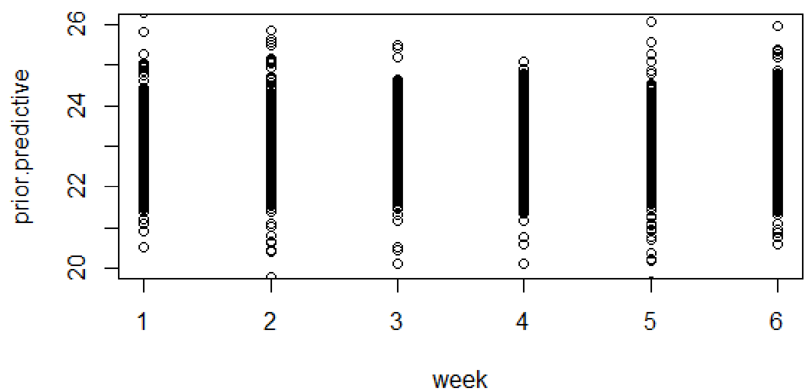
a) I set a prior of:

beta0=matrix(c(23,0.001),nrow=2,ncol=1)

Sigma0=matrix(c(0.2,0,0,0.0001),nrow=2,ncol=2); gamma0=2; sigma20=0.01

The corresponding prior predictive values of y are:



Week	1	3	5	7	9	11
Mean	22.96400	23.03614	23.01940	22.99436	23.00404	23.00219
95% Interval	(22.07806 23.92952)	(22.06302 23.94010)	(22.06648 23.94304)	(22.05332 23.95180)	(22.04757 23.94575)	(22.07460 23.97269)

We can see the result is close to our knowledge that for the age group the general time is between 22 seconds to 24 seconds. And thus the prior can be held for further analysis.

b) Using posterior predictive distribution, we can use MCMC to simulate the swimming time of each of the 4 individual in week 13. The simulated result is that

Swimmer	1	3	5	7
Mean	22.62978	23.56570	22.89240	23.37976
95% Interval	(22.59534 22.66661)	(23.52597 23.60469)	(22.84136 22.94213)	(23.34021 23.41743)

Using LLN, we can calculate out the probability that individual 1 will almost be 100% quicker than all the other 3 ones. And thus, we should choose individual 1 according to the linear regression model.

However, we know that the model is not that good because we are not given information on how the error is distributed. And we only considered week number as a variable to estimate the result.

##CODE

```
library(mvtnorm)
######## 1 ########33
#Informative Prior

beta0=matrix(c(23,0.001),nrow=2,ncol=1)
Sigma0=matrix(c(0.2,0,0,0.0001),nrow=2,ncol=2)
gamma0=2
sigma20=0.01

#Prior Predictive
S=5000
prior.predictive <- error <- sigma2 <- matrix(0,nrow=S,ncol=6)
set.seed(1)
for (s in 1:S)
{
   for (t in 1:6)
  {
   beta<- rmvnorm(1,beta0,Sigma0)
   sigma2[s,t] <- 1/rgamma(1,gamma0/2,gamma0*sigma20/2)
   error[s,t] <- rnorm(1,0,sigma2[s,t])
   prior.predictive[s,t] <- beta[1]+(t*2-1)*beta[2]+error[s,t]
  }
}

week=cbind(matrix(1,nrow=S,ncol=1),matrix(2,nrow=S,ncol=1),matrix(3,nrow=S,ncol=1),
     matrix(4,nrow=S,ncol=1),matrix(5,nrow=S,ncol=1),matrix(6,nrow=S,ncol=1))
plot(week,prior.predictive,ylim=c(20,26))
colMeans(prior.predictive)
quantile(prior.predictive[,1],c(0.025,0.975))
quantile(prior.predictive[,2],c(0.025,0.975))
quantile(prior.predictive[,3],c(0.025,0.975))
quantile(prior.predictive[,4],c(0.025,0.975))
quantile(prior.predictive[,5],c(0.025,0.975))
quantile(prior.predictive[,6],c(0.025,0.975))

############ 2 ############

Y <- read.table("http://www.stat.washington.edu/hoff/Book/Data/hwdata/swim.dat")

#set initials
sigma2 <- 1/rgamma(1,gamma0/2,gamma0*sigma20/2)
Y.pred <- matrix(nrow=S,ncol=4)

#Posterior Predictive#
for ( s in 1:S)
{
```

```
  for (j in 1:4)
  {
    X <- matrix(c(1,1,1,1,1,1,1,3,5,7,9,11),nrow=6,ncol=2)
    Sigma.star <- solve(solve(Sigma0)+(t(X)%*%X)/sigma2)
    y <- unlist(matrix(Y[j,],nrow=6,ncol=1))
    mu.star <- Sigma.star%*%(solve(Sigma0)%*%beta0+t(X)%*%y/sigma2)
    beta <- t(rmvnorm(1,mu.star,Sigma.star))

    SSR <- t(y-X%*%beta)%*%(y-X%*%beta)
    sigma2 <- 1/rgamma(1,(gamma0+n)/2,(gamma0*sigma20+SSR)/2)

    Y.pred[s,j]<- c(1,13)%*%beta+rnorm(1,0,sigma2.t)
  }
}

count=matrix(0,nrow=4,ncol=1)
for (s in 1:S)
{
  for (j in 1:4)
  {
    count[j]=count[j]+as.numeric(Y.pred[s,j]==min(Y.pred[s,]))
  }
}

colMeans(Y.pred)
quantile(Y.pred[,1],c(0.025,0.975))
quantile(Y.pred[,2],c(0.025,0.975))
quantile(Y.pred[,3],c(0.025,0.975))
quantile(Y.pred[,4],c(0.025,0.975))
```