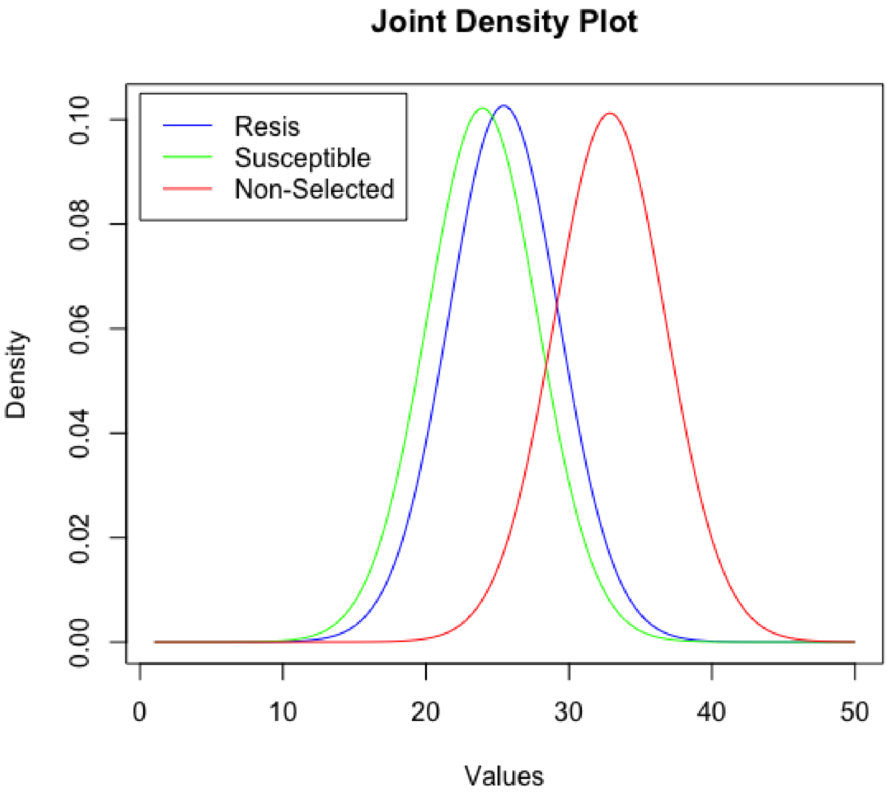
CSI-674 Jericho McLeod
Final Exam G00986513

Problem 1:

The joint posterior distributions for the thetas are normal-gamma distributions with the following hyperparameters:

	μ	k	α	β
θ_R	25.43971	32.78494	33.16422	0.0007224692
θ_S	23.95000	32.78494	33.16442	0.0007160803
θ_N	32.86629	32.78494	33.16442	0.0007025716

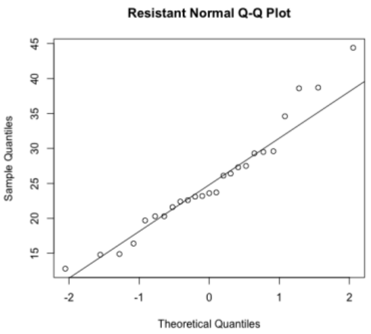
The density plot of the joint distributions is shown below.

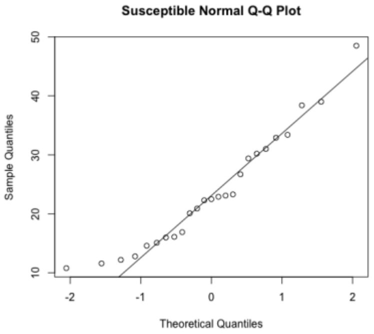


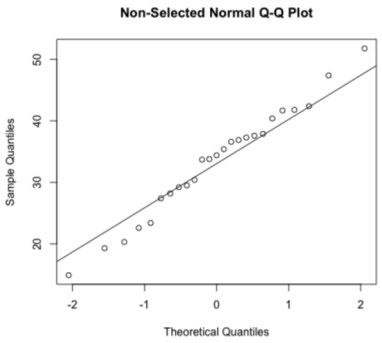
The 95% confidence intervals are:

	0.025	0.975
θ_R	17.71129	33.16813
θ_S	16.18718	31.71282
θ_N	25.02919	40.70339

Based on the Q-Q plots, the data appears to be nearly normal. There are some fluctuations







Primarily, the bottom end of the susceptible class of fruit flies appears to fan out some; this suggest the data may be skewed or otherwise not-normal to some degree. Using my judgement, I would accept this as close enough.

Code used for Problem 1:

```
library(data1.table)
data1 <- fread('https://www2.stat.duke.edu/courses/Spring03/sta113/Data/Hand/fruitfly.dat')
head(data1)
R <- data1$V1
S <- data1$V2
N <- data1$V3

all <- c(R,S,N)

len <- length(R)

qqnorm(R,main='Resistant Normal Q-Q Plot')
qqline(R)
qqnorm(S,main='Susceptible Normal Q-Q Plot')
qqline(S)
qqnorm(N,main='Non-Selected Normal Q-Q Plot')
qqline(N)

qqnorm(all,main='All Fruit Flies Normal Q-Q Plot')
qqline(all)

sumR <- sum(R)
sumS <- sum(S)
sumN <- sum(N)

# Estimate Mean
mu_0 <- mean(c(mean(R),mean(S),mean(N)))

# Estimate the sample precisions
pR0 <- 1/var(R)
pS0 <- 1/var(S)
```

```
pN0 <- 1/var(N)

# Estimate mean of aB
ab_mean0 <- mean(c(pR0, pS0, pN0))

# Estimate the variance aB^2
ab2_var0 <- var(c(pR0, pS0, pN0))

# Solving for alpha and beta (shape = sh, scale=sc)
sh <- ab_mean0^2/ab2_var0
sc <- ab2_var0/ab_mean0

# Estimate K
# Sample means
mean_R <- mean(R)
mean_S <- mean(S)
mean_N <- mean(N)

# Sample means Var
means_var <- var(c(mean_R,mean_S,mean_N))

# Precision of Means
means_prec <- 1/means_var

# Estimate K
k <- means_prec / mean(c(pR0,pS0,pN0))

print(c(mu_0,k,sh,sc))

# Now that we have Mu, K, Alpha, and Beta empirical Bayesian Estimates,
# we need to update them for the samples (Unit 5 page 26/27)

r_alpha1 <- sh + n/2
s_alpha1 <- sh + n/2
n_alpha1 <- sh + n/2
r_beta1 <- 1 / (1/sc + 0.5*sum(R-mean_R)^2 + ((n*k)/(2*(n+k)))*(mean_R -
mu_0)^2 )
s_beta1 <- 1 / (1/sc + 0.5*sum(S-mean_S)^2 + ((n*k)/(2*(n+k)))*(mean_S -
mu_0)^2 )
n_beta1 <- 1 / (1/sc + 0.5*sum(N-mean_N)^2 + ((n*k)/(2*(n+k)))*(mean_N -
mu_0)^2 )
r_mu1 <- (k*mu_0 + n*mean_R) / (k+n)
s_mu1 <- (k*mu_0 + n*mean_S) / (k+n)
n_mu1 <- (k*mu_0 + n*mean_N) / (k+n)
r_k1 <- k+n
s_k1 <- k+n
n_k1 <- k+n

# Printing off the results:
print(c(r_mu1,r_k1,r_alpha1,r_beta1))
print(c(s_mu1,s_k1,s_alpha1,s_beta1))
print(c(n_mu1,n_k1,n_alpha1,n_beta1))

# Creating a density plot to review the results
# First, calculating the spread for each theta
r_spread <- sqrt(1/(k*r_alpha1*r_beta1))
s_spread <- sqrt(1/(k*s_alpha1*s_beta1))
```

```
n_spread <- sqrt(1/(k*n_alpha1*n_beta1))

# Then updating the input array using the mean and spread
vals <- seq(1,50,length=500)
stdval_r <- (vals-r_mu1)/r_spread
stdval_s <- (vals-s_mu1)/s_spread
stdval_n <- (vals-n_mu1)/n_spread

# Calculating density across inputs
theta_r <- dt(stdval_r,2*r_alpha1)/r_spread
theta_s <- dt(stdval_s,2*s_alpha1)/s_spread
theta_n <- dt(stdval_n,2*n_alpha1)/n_spread

# Plotting the results
plot(vals,theta_r,type='l',col='blue',main="Joint Density
Plot",xlab='Values',ylab='Density')
lines(vals,theta_s,col='green')
lines(vals,theta_n,col='red')
legend(0,0.105,c("Resis",'Susceptible','Non-Selected'),
       col=c('blue','green','red'),lty=c(1,1,1))

# And finally, calculating the 90
proba <- c(0.025,0.975)

r_ci <- (qt(proba,2*r_alpha1)*r_spread)+r_mu1
s_ci <- (qt(proba,2*s_alpha1)*s_spread)+s_mu1
n_ci <- (qt(proba,2*n_alpha1)*n_spread)+n_mu1

print(r_ci)
print(s_ci)
print(n_ci)
```

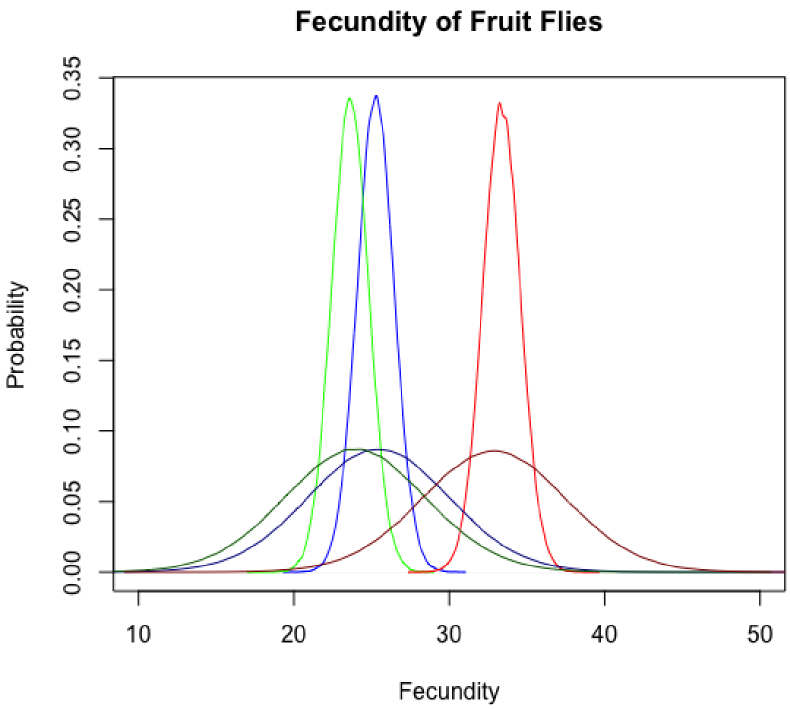
Jericho McLeod
G00986513

Problem 2:

Posterior probability that Non-Selected fruit-flies have a higher fecundity than selected fruit flies: 95.207%

The posterior probability that a Non-Selected fruit fly chosen at random will have greater fecundity than both a randomly selected Resistant and a randomly selected Susceptible fruit fly is much lower at 81.826%.

The important distinction between the two is that the former percentage is an estimate of the means, while in the second, it is among a randomly drawn sample of size 1 for each category. The probabilities for means are calculated from normal distributions, while in the latter calculation, a T distribution is used for each group to marginalize out the unknown precision; this widens the distribution relative to the sample drawn. Since the sample size is 1, it is much wider. This is visualized in the chart below.



The distributions for means are in lighter colors, with red being the non-selected group. The distributions for single fruit-flies are in darker colors with again red being for the non-selected fruit flies.

Code used for Problem 2:

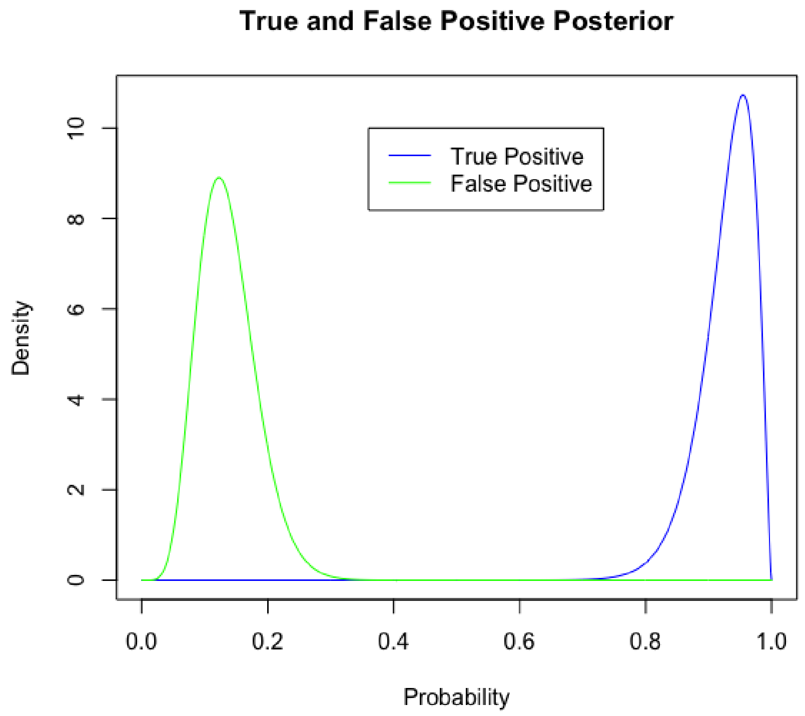
```
print(c(r_mu1,r_k1,r_alpha1,r_beta1))
print(c(s_mu1,s_k1,s_alpha1,s_beta1))
print(c(n_mu1,n_k1,n_alpha1,n_beta1))
# Part 1: draw MC samples
numSim <- 100000
```

```
r_rho  <- rgamma(numSim,shape=r_alpha1,scale=r_beta1)
r_sigma <- 1/sqrt(r_rho)
r_theta <- rnorm(numSim,mean=mean_R,sd=r_sigma/sqrt(n))

s_rho  <- rgamma(numSim,shape=s_alpha1,scale=s_beta1)
s_sigma <- 1/sqrt(s_rho)
s_theta <- rnorm(numSim,mean=mean_S,sd=s_sigma/sqrt(n))

n_rho  <- rgamma(numSim,shape=n_alpha1,scale=n_beta1)
n_sigma <- 1/sqrt(n_rho)
n_theta <- rnorm(numSim,mean=mean_N,sd=n_sigma/sqrt(n))

# Plot densities
plot(density(r_theta),col='blue',xlim=c(18,38),main="Mean Fecundity of Fruit
Flies",xlab='Fecundity',ylab='Probability')
lines(density(s_theta),col='green')
lines(density(n_theta),col='red')

posterior_nonselected <- sum(n_theta>max(r_theta,s_theta))/length(n_theta)

# Part 2: Draw MC samples from Gamma -> T distributions
numSim <- 1000000

# Calculate the spreads
r_spread <- 1/sqrt((k/k+1)*r_alpha1*r_beta1)
s_spread <- 1/sqrt((k/k+1)*s_alpha1*s_beta1)
n_spread <- 1/sqrt((k/k+1)*n_alpha1*n_beta1)

# dMC
r_theta_t <- r_mu1+rt(numSim,df=2*r_alpha1)*r_spread
s_theta_t <- s_mu1+rt(numSim,df=2*s_alpha1)*s_spread
n_theta_t <- n_mu1+rt(numSim,df=2*n_alpha1)*n_spread

# Plotting Densities
plot(density(r_theta_t),col='blue',xlim=c(5,50),main="Fecundity of Single
Fruit Flies",xlab='Fecundity',ylab='Probability')
lines(density(s_theta_t),col='green')
lines(density(n_theta_t),col='red')

# Calculate exact percentages.
posterior_nonselected_1 <-
sum((n_theta_t>s_theta_t)&(n_theta_t>r_theta_t))/length(n_theta_t)
n_r<-sum(n_theta_t>r_theta_t )/length(n_theta_t)
n_s<-sum(n_theta_t>s_theta_t)/length(n_theta_t)

# Plot collected Densities
plot(density(r_theta),col='blue',xlim=c(10,50),main="Fecundity of Fruit
Flies",xlab='Fecundity',ylab='Probability')
lines(density(s_theta),col='green')
lines(density(n_theta),col='red')
lines(density(r_theta_t),col='darkblue')
lines(density(s_theta_t),col='darkgreen')
lines(density(n_theta_t),col='darkred')
```

Problem 3:

The posterior distributions are beta distributions with the following parameters and 95% credible intervals:

	Alpha	Beta	0.025	0.975
True Positive	32.5	2.5	0.8244175	0.9875756
False Positive	7.5	47.5	0.0599024	0.2377323



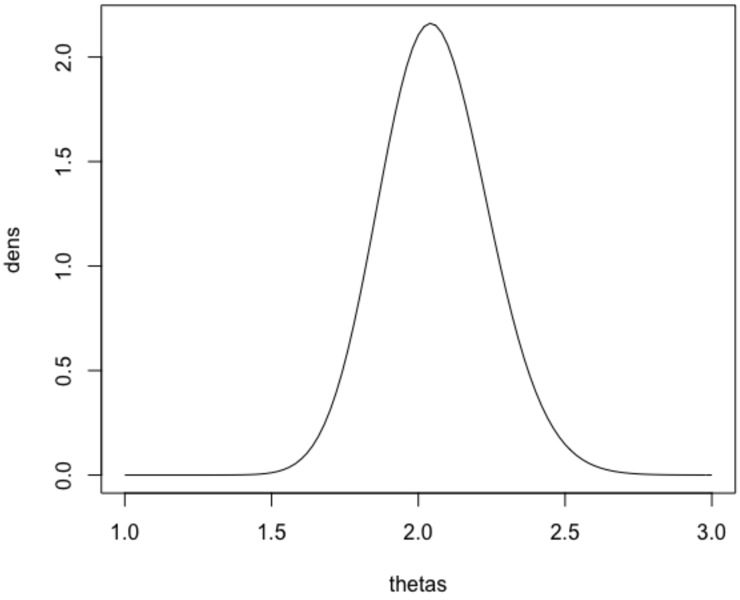
Code used for Problem 3:

```
require(rmutil)
acc <- 0.9
tp_alpha0 <- 4.5
tp_beta0 <- 0.5
fp_alpha0 <- 0.5
fp_beta0 <- 4.5

tp_alpha1 <- tp_alpha0 + 28
tp_beta1 <- tp_beta0 +(30-28)

fp_alpha1 <- fp_alpha0 + 7
fp_beta1 <- fp_beta0 + (50-7)

thetas <- seq(0,1,length=1001)

tp_p <- tp_alpha1 / (tp_alpha1+tp_beta1)
tp_m <- tp_alpha1+tp_beta1
```

```
fp_p <- fp_alpha1 / (fp_alpha1+fp_beta1)
fp_m <- fp_alpha1+fp_beta1

tp_dens <- dbeta(thetas,tp_alpha1,tp_beta1)
fp_dens <- dbeta(thetas,fp_alpha1,fp_beta1)

plot(thetas,tp_dens,type='l',col='blue',main='True and False Positive
Posterior',xlab='Probability',ylab='Density')
lines(thetas,fp_dens,col='green')
legend(0.36,10,c("True Positive","False
Positive"),col=c("blue","green"),lty=c(1,1))

pv <- c(0.025, 0.975)
tp_ci <- qbeta(pv,tp_alpha1,tp_beta1)
fp_ci <- qbeta(pv,fp_alpha1,fp_beta1)
tp_ci
fp_ci
```

Problem 4:

The posterior distribution of Lambda is a gamma distribution with shape 123.5 and scale 0.016667.



Code used for Problem 4:

```
alpha0 <- 0.5
beta0 <- Inf
alpha1 <- alpha0 + 123
beta1 <- 1/(1/beta0 + 60)

thetas <- seq(1,3,length=101)
dens <- dgamma(thetas,shape=alpha1,scale=beta1)
plot(thetas,dens,type='l')
```

Problem 5:

Based on the above posterior distribution for Lambda, the probability that 40 or more calls will arrive during a 15 minute mid-morning period is 0.06648101.

Code used for Problem 5:

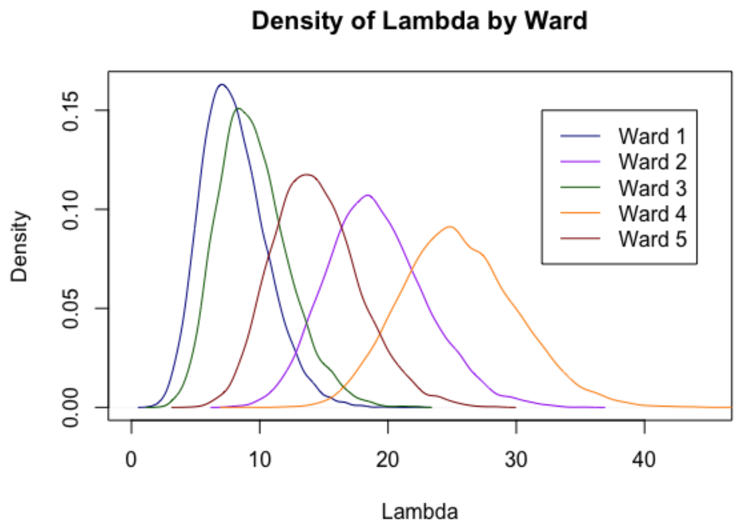
```
x <- dnbinom(seq(0,100,length=101),size=alpha1,prob=1/(1+15*beta1))
plot(seq(0,100,length=101),x,typ='l')

p <- 1-pnbinom(40,size=alpha1,prob=1/(1+15*beta1))
print(p)
```

Problem 6:

The resulting Lambda values, or estimates of the unknown rate parameters for the Poisson distributions of crime rates across 5 wards, have the following 95% credible interval after a 1000 example Gibbs Sampled Monte Carlo simulation:

	0.025	0.975
Λ_1	3.927	13.597
Λ_2	12.222	27.442
Λ_3	4.982	15.715
Λ_4	17.520	35.028
Λ_5	8.578	21.841

The amounts of overlap between Wards is much easier to understand as a density plot:



We can clearly see that Ward 4 probably has a higher crime rate than Ward 1, but we would be less likely to say Ward 4 has a higher crime rate than Ward 2. This uncertainty comes from the inferenced distribution of the rates, rather than relying on a single observation, making it more credible in its reporting by adding in the uncertainty.

Code used for Problem 6:

```
# find conditional for lambdas given A and B
# then find posterior and sample from it

nW <- 5
nC <- c(9,23,11,31,17)

numSim <- 10000
```

```
aGrid <-  seq(1,50,length=50)       # Alpha
aPrior <- 1/aGrid
aPrior <- aPrior/sum(aPrior)
mGrid <-  seq(4,40,length=50)       # M
mPrior <- seq(0.02,0.02,length=50)

aMC <- sample(aGrid,1,prob=aPrior)
mMC <- sample(mGrid,1,prob=mPrior)

lam <- array(0, dim=c(numSim,nW))   # Allocate space for simulated CRIMES
RATES --- Lambda
lam[1,] <- rgamma(nW,shape=aMC,scale=mMC/aMC)

for (k in 2:numSim) {

  # This doesn't seem to work right; getting roughly the same lambdas at the
end.
  # Trying instead to sample from posteriors given data individually by
Lambda
  #alpha1 = aMC[k-1] + (sum(nC))
  #beta1  = 1/(1/(mMC[k-1] / aMC[k-1]) + nW)

  # sample new lambdas from alpha and beta priors
  #lam[k,] <- rgamma(nW,shape=alpha1,scale=beta1)


  # Sample new lambdas from individually updated gamma distributions
  for (j in 1:nW){
    alpha1 <- aMC[k-1] + nC[j]
    beta1  <- 1/(1/(mMC[k-1] / aMC[k-1]) + 1)
    lam[k,j] <- rgamma(1,shape=alpha1,scale=beta1)
  }

  # from new lambdas get alpha likelihoods
  aLik <- 1
  for (j in 1:nW) {
    aLik <- aLik * dgamma(lam[k-1,j],aMC[k-1]*aPrior,(mMC[k-
1]*mPrior)/(aMC[k-1]*aPrior))
  }
  # find conditional for lambdas given A and B
  # then find posterior and sample from it
  aPost <- aPrior*aLik/sum(aPrior*aLik)
  aMC[k] <- sample(aGrid,1,prob=aPost)

  # from new lambdas get beta likelihoods
  mLik <- 1
  for (j in 1:nW) {
    mLik <- mLik * dgamma(lam[k-1,j],aMC[k]*aPrior,(mMC[k-
1]*mPrior)/(aMC[k]*aPrior))
  }
  # find conditional for lambdas given A and B
  # then find posterior and sample from it
  mPost <- mPrior*mLik/sum(mPrior*mLik)
  mMC[k] <- sample(mGrid,1,prob=mPost)

}
```

```
quantile(lam[,1],c(0.025, 0.975))
quantile(lam[,2],c(0.025, 0.975))
quantile(lam[,3],c(0.025, 0.975))
quantile(lam[,4],c(0.025, 0.975))
quantile(lam[,5],c(0.025, 0.975))


plot(density(lam[,1]),col='dark blue',xlim=c(0,45),main='Density of Lambda by
Ward',ylab='Density','xlab'='Lambda')
lines(density(lam[,2]),col='purple')
lines(density(lam[,3]),col='dark green')
lines(density(lam[,4]),col='dark orange')
lines(density(lam[,5]),col='dark red')
legend(32,0.15,c("Ward 1",'Ward 2','Ward 3','Ward 4','Ward 5'),
       col=c('dark blue','purple','dark green','dark orange','dark
red'),lty=c(1,1,1,1,1))
```

Problem 7:

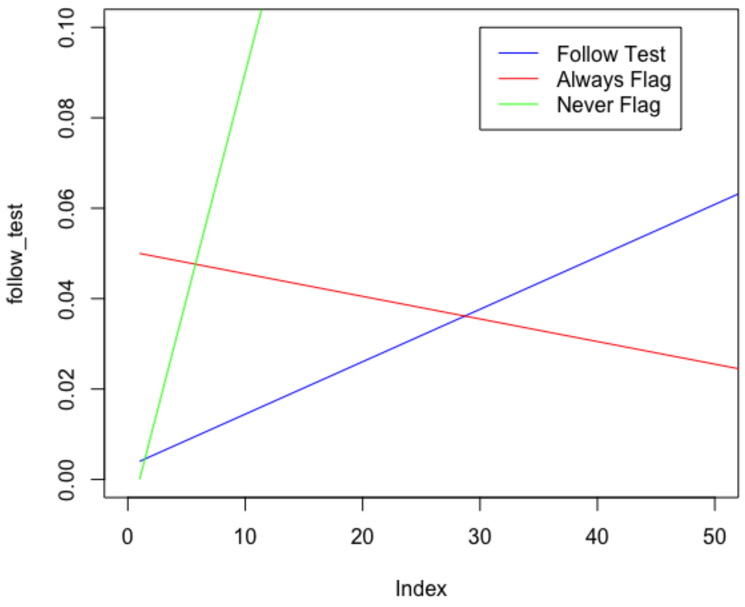
Table of initial probabilities and costs:

Actual	Test	Action	Loss	Probability
OK	OK	None	0	0.92*(1-P)
OK	Def	Flag	0.05	0.08*(1-P)
Def	Ok	None	1	0.12*P
Def	Def	Flag	0	0.88*P

As functions of P, the loss calculations are:

Follow Test: $f(p) = p*0.12*1 + (1-p)*0.08*0.05$
Always Flag: $f(p) = (1-p)*0.05$
Never Flag: $f(p) = p*1$

If P is zero, it is best to not flag any items as defective. When P > 0.01 (my smallest checked value), it is best to follow the test until P > 0.29, at which time it becomes the least costly to always flag every item as defective.



These results depend heavily on the fact that there is no cost for correct answers, and that the cost for a missed defective item being very high.

Code for Problem 7:

```
cost_fp <- 0.05
cost_fn <- 1

p <- seq(0,1,length=101)

# Actual  Test  Action  Loss  Probability
# Ok      Ok    None    0     0.92*(1-P)
# Ok      Def   flag    0.05  0.08*(1-P)
# Def     Ok    None    1     0.12*P
# Def     Def   flag    0     0.88*P

follow_test <- p*0.12*1 + (1-p)*0.08*0.05
always_flag <- (1-p)*0.05
never_flag  <- p*1

plot(follow_test,type='l',col='blue',ylim=c(0,.1),xlim=c(0,50))
lines(always_flag,col='red')
lines(never_flag,col='green')
legend(30,.10,c("Follow Test","Always Flag",'Never
Flag'),col=c("blue","red","green"),lty=c(1,1,1))

print(follow_test)
print(never_flag)
print(always_flag)
```

Problem 8:

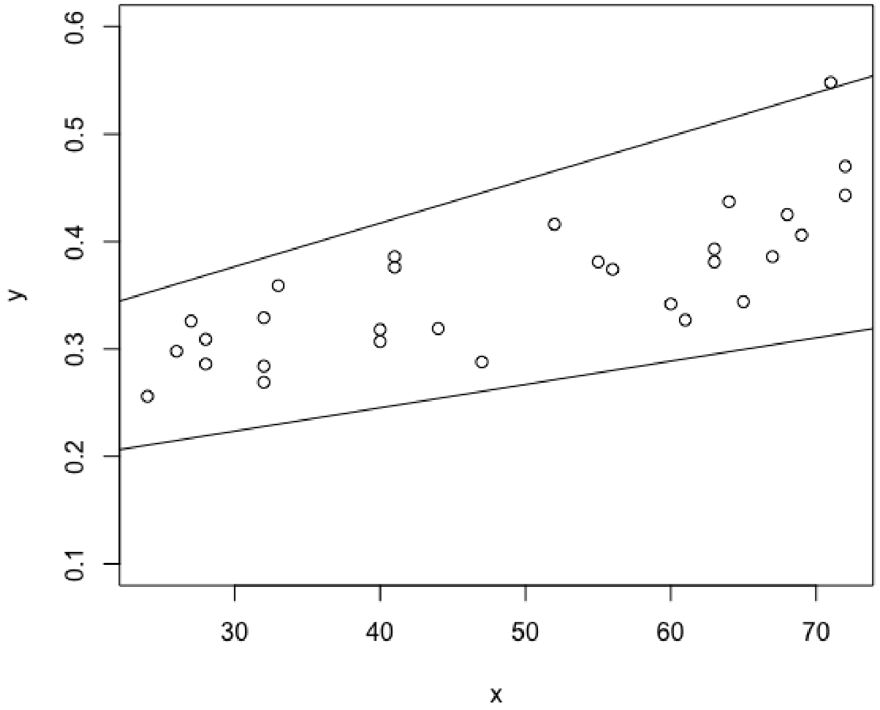
Joint Posterior Distribution:
The precision, Rho, is from a gamma distribution with shape (n-2)/2 = 14 and scale 2/S_{ee} = 39.99254

Conditional on Rho, the intercept, Eta, is a normal distribution with mean h= Ybar = 0.3594 and precision n*rho = 30*Rho.

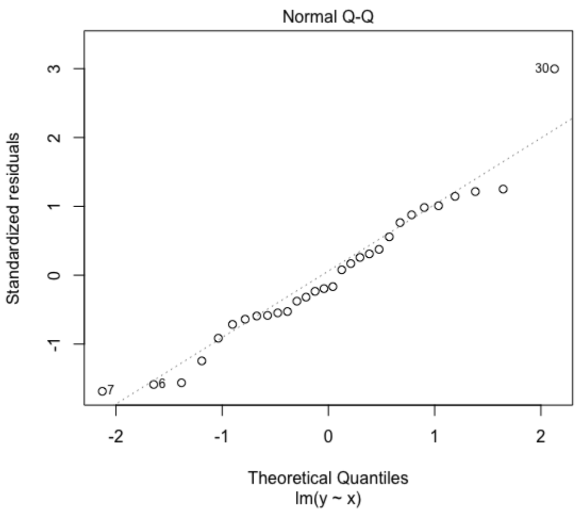
Given Rho, the slope, Beta, is also a normal distribution with mean b = S_{xy}/S_{xx} = 0.003107 and precision S_{xx}*Rho = 7820.7*Rho.

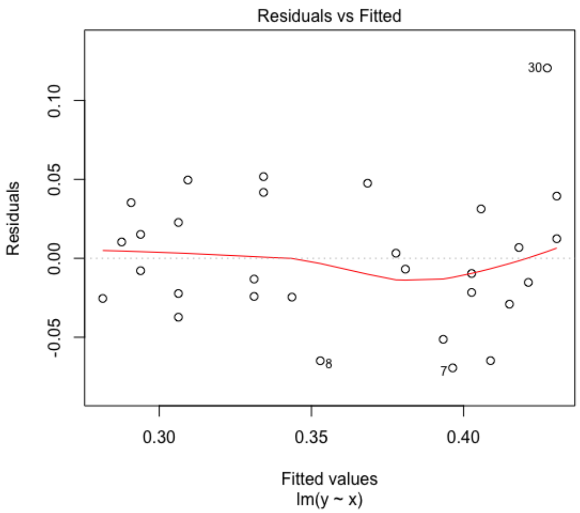
Thus, the joint density function for (eta, beta, rho) is the product of a gamma density function for rho with shape 14 and scale 39.99254, a normal density function with mean 0.3594 and precision 30*rho for eta, and a normal density function with mean 0. 003107 and precision 7820.7*rho for beta.

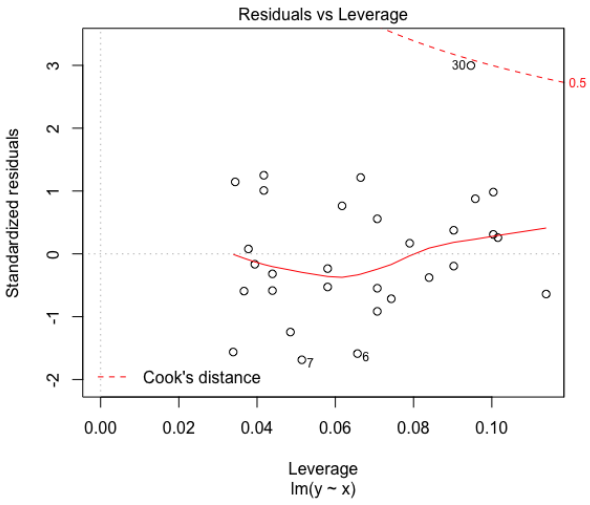
The 95% credible interval for slope is 0.00217 to 0.004044, and the same for the intercept is 0.15845 to 0.25527. Plotting this credible interval for the data reveals the following:



The Q-Q plot shots a single outlier, which appears again in the residuals to fitted plot and the residuals to leverage plot. This value may need to be excluded.







Code used for Problem 8:

```
library(data.table)
data8 <-
fread('https://www2.stat.duke.edu/courses/Spring03/sta113/Data/Hand/icecream.
dat')
head(data8)

y <- data8$V2 # Consumption
x <- data8$V5 # Temperature F
x <- x[-length(x)]
y <- y[-length(y)]
n = length(x)

xbar <- mean(x)
```

```
ybar <- mean(y)

b <- sum( (x-xbar) * (y-ybar)) / sum( (x-xbar)^2 )
a <- ybar - b*xbar

sxx <- sum( (x-xbar)^2 )
syy <- sum( (y-ybar)^2 )
sxy <- sum( (x-xbar) * (y-ybar) )
see <- sum( (y-ybar-b*(x-xbar))^2)

#posteriors
rho_alpha <- (n-2)/2
rho_beta  <- 2/see

eta_center <- ybar
eta_spread <- (n*(n-2)/see)^-0.5
eta_degf   <- n-2

beta_center <- b
beta_spread <- (see*(n-2)/see)^-0.5
beta_degf   <- n-2

s2<- see/(n-2)
s <- sqrt(s2)

print(mean(y))
print(mean(x))

fit=lm(y ~ x)
summary(fit)  # Summary of regression results
plot(fit)     # Diagnostic plots from lm object

qnorm(c(0.025, 0.975),mean=0.2068621,sd=0.0247002) #intercept
qnorm(c(0.025, 0.975),mean=0.0031074,sd=0.0004779) #slope

temps <- seq(20,80,length=61)
min_vals <- sort(temps*0.002170733 + 0.1584506)
max_vals <- sort(temps*0.004044067 + 0.2552736)
plot(x,y,ylim=c(0.1,0.6))
lines(temps,min_vals)
lines(temps,max_vals)
print(max_vals)
```

Problem 9:

The posterior predictive intervals are shown in the table below:

	0.05	0.95
46 Degrees F	0.2667	0.4330
98 Degrees F	0.4282	0.5946

Of these, I would trust the 46 degree prediction to be right about 90% of the time, while I have trouble trusting the 98 degree prediction because it is well outside of the range of prior observations. Outliers are often different, and this is no different merely because it is unobserved.

Code used for Problem 9:

```
# Posterior predictive distribution for temps
xlow = 46
xhigh = 98

predlow.center = a + b*xlow
predlow.spread = sqrt(((xhigh-xbar)^2/sxx + 1/n + 1)*see/(n-2))
predlow.df = n-2

predhigh.center = a + b*xhigh
predhigh.spread = sqrt(((xhigh-xbar)^2/sxx + 1/n + 1)*see/(n-2))
predhigh.df = n-2

# Exact credible interval for prediction @ 19 mg
predlow.center + qt(c(0.05,0.95),predlow.df)*predlow.spread
predhigh.center + qt(c(0.05,0.95),predhigh.df)*predhigh.spread
```

Problem 10:

The vector of binomial probabilities can be shown as beta distributions. The uniform prior for a beta distribution is shape1 = 1 and shape 2 = 1.

Thus, initially, we have the following collection of beta distributions:
[beta_e(1,1), beta_d(1,1), beta_u(1,1)]

The posterior distributions for the betas after observing 30 tests would be:
[beta_e(10,22), beta_d(16,16), beta_u(7,25)]

The normalized expectation likelihoods:
[0.3020885, 0.4895282, 0.2083833]

The confidence intervals would then be:

essential_ci:	0.1866216	to	0.4519044
desirable_ci:	0.3565721	to	0.6434279
unnecessary_ci:	0.1110888	to	0.3466525

The Bayesian posteriors and confidence intervals are somewhat between the mean prior expected values based on the prior uniform distribution, meaning the high value for 'desirable' is pulled slightly while the essentially and unnecessary confidence intervals are slightly higher than the exact distribution of the observations. This is a typical feature of Bayesian statistics; the prior distribution and observations both influence the outcome.

This aside, the posterior is about what one would expect, and in all three cases would fit the observed data, indicating the prior was not bad, and that the model is probably a reasonable predictor for future votes.

If one were using these results to make a democratic decision, this would probably be sufficient to state that most individuals in the target population believe the asynchronous interaction capability to be either essential or desirable, as the lower bound of the confidence intervals for them account for more than half of the population.

Code used for Problem 10:

```
# Priors
beta_e <- c(1,1)
beta_d <- c(1,1)
beta_u <- c(1,1)

# Posteriors
beta_e <- c(10,22)
beta_d <- c(16,16)
beta_u <- c(7,25)
```

```
e_ci <- qbeta(c(0.05,0.95),shape1=beta_e[1],shape2=beta_e[2])
d_ci <- qbeta(c(0.05,0.95),shape1=beta_d[1],shape2=beta_d[2])
u_ci <- qbeta(c(0.05,0.95),shape1=beta_u[1],shape2=beta_u[2])
e_ci
d_ci
u_ci

e_ci <- qbeta(c(.5),shape1=beta_e[1],shape2=beta_e[2])
d_ci <- qbeta(c(.5),shape1=beta_d[1],shape2=beta_d[2])
u_ci <- qbeta(c(.5),shape1=beta_u[1],shape2=beta_u[2])
v <- c(e_ci,d_ci,u_ci)
v <- v/sum(v)
print(v)
```