Homework 06

Simulation

Your Name

September 2, 2017

Discrete probability simulation:

suppose that a basketball player has a 60% chance of making a shot, and he keeps taking shots until he misses two in a row. Also assume his shots are independent (so that each shot has 60% probability of success, no matter what happened before).

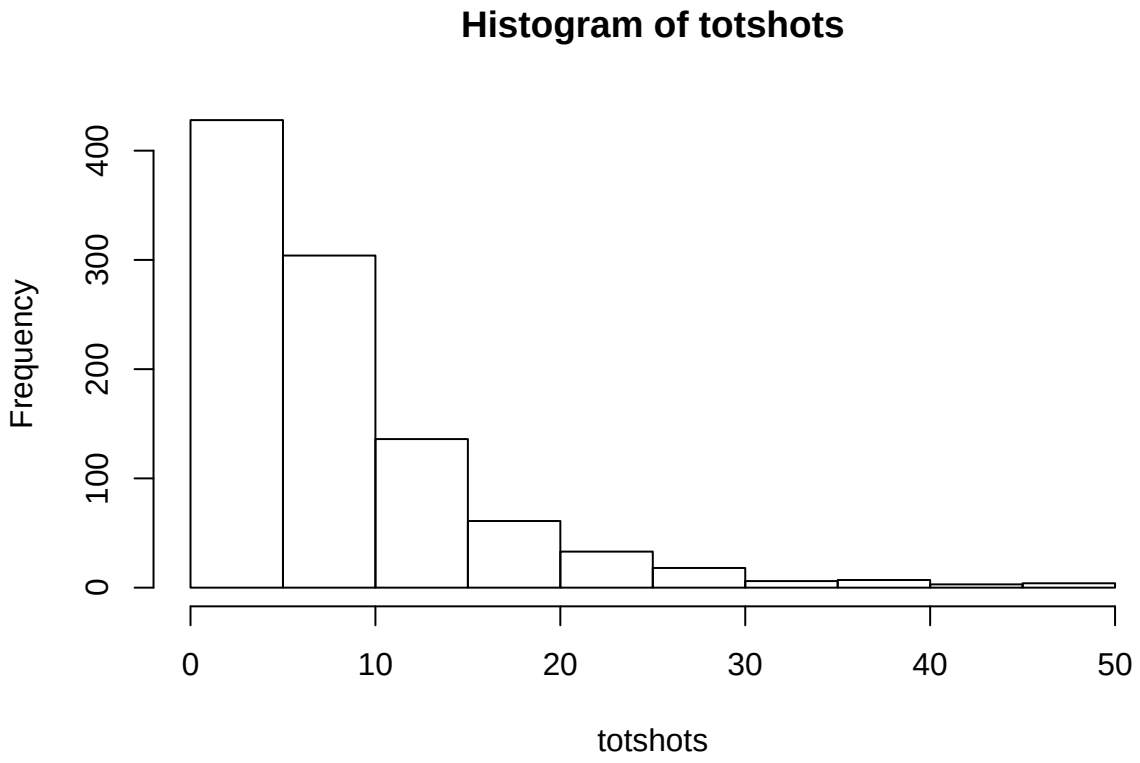
1. Write an R function to simulate this process.

```
sim_shots<-function(){
  ee <- TRUE # flag for checking
  shots<- rbinom(1, 1, 0.6) # first shot
  i=1
  while( ee ) {
    i = i + 1
    ashot<- rbinom(1, 1, 0.6) # subsequent shots
    if(shots[i-1]==0 && ashot==0){ ee=FALSE } # two miss in a row
    shots <- c(shots,ashot)
  }
  return(shots)
}
```

2. Put the R function in a loop to simulate the process 1000 times. Use the simulation to estimate the mean, standard deviation, and distribution of the total number of shots that the player will take.

```
n.sims<-1000
totshots<-rep(NA,n.sims)
propshots<-rep(NA,n.sims)

for(i in 1:n.sims){
  simshots<-sim_shots()
  totshots[i]<-length(simshots)
  propshots[i]<-mean(simshots)
}
hist(totshots)
```




```
mean(totshots)
```

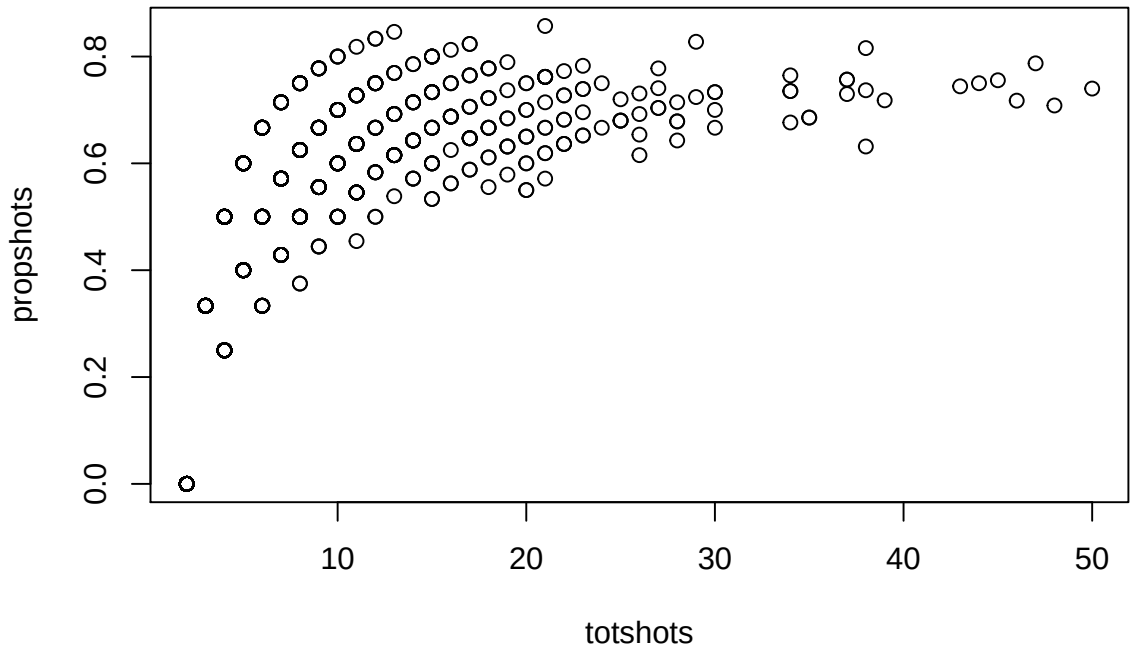
```
## [1] 8.62
```

```
sd(totshots)
```

```
## [1] 7.358379
```

3. Using your simulations, make a scatterplot of the number of shots the player will take and the proportion of shots that are successes.

```
plot(totshots,propshots)
```

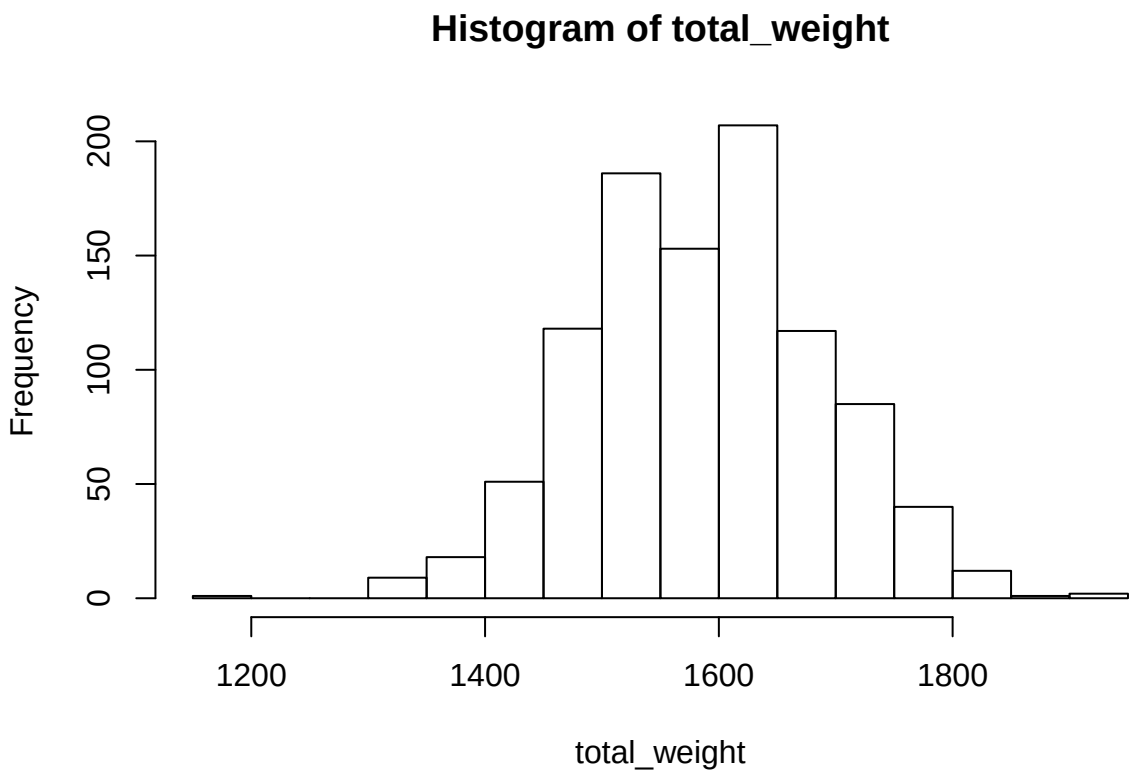


Continuous probability simulation:

the logarithms of weights (in pounds) of men in the United States are approximately normally distributed with mean 5.13 and standard deviation 0.17; women with mean 4.96 and standard deviation 0.20. Suppose 10 adults selected at random step on an elevator with a capacity of 1750 pounds. What is the probability that the elevator cable breaks?

We will use the fact that the ratio betwee men and woman in US is about 0.49:0.51

```
n.sims<-1000
total_weight<- rep(NA,n.sims)
for(i in 1:n.sims){
  male<- rbinom(10,1,0.49) # number of males in a sample
  male_weight<-rnorm(sum(male),5.13,0.17) # sample of male weights
  nfem <- 10-sum(male); # number of females in a sample
  if(nfem>0){
    female_weight<-rnorm(nfem,4.96,0.2) # sample of female weights
  }else {
    female_weight<-0 # No female no weights
  }
  total_weight[i]<-sum(c(exp(male_weight),exp(female_weight)))
}
hist(total_weight)
```



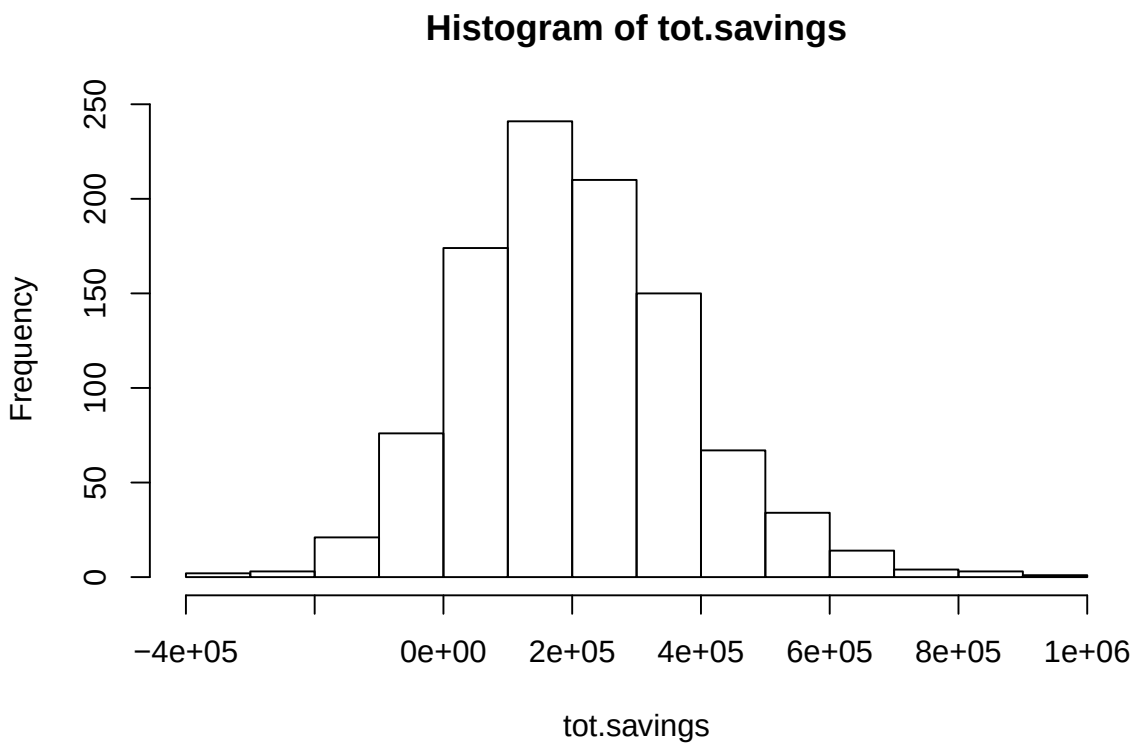

```
mean(total_weight>1750)
```

```
## [1] 0.055
```

Propagation of uncertainty:

we use a highly idealized setting to illustrate the use of simulations in combining uncertainties. Suppose a company changes its technology for widget production, and a study estimates the cost savings at $5 per unit, but with a standard error of $4. Furthermore, a forecast estimates the size of the market (that is, the number of widgets that will be sold) at 40,000, with a standard error of 10,000. Assuming these two sources of uncertainty are independent, use simulation to estimate the total amount of money saved by the new product (that is, savings per unit, multiplied by size of the market).

```
savings.mean <- 5
savings.se   <- 4
market.size  <- 40000
market.se    <- 10000
n.sims       <- 1000
tot.savings  <- rep(NA, n.sims)
for (t in 1:n.sims) {
    market  <- rnorm(n=1, mean=market.size,  sd=market.se)
    savings <- rnorm(n=1, mean=savings.mean, sd=savings.se)
    tot.savings[t] <- market * savings
}
hist(tot.savings)
```



Predictive simulation for linear regression:

take one of the models from previous exercise that predicts course evaluations from beauty and other input variables. You will do some simulations.

1. Instructor A is a 50-year-old woman who is a native English speaker and has a beauty score of 1. Instructor B is a 60-year-old man who is a native English speaker and has a beauty score of - .5. Simulate 1000 random draws of the course evaluation rating of these two instructors. In your simulation, account for the uncertainty in the regression parameters (that is, use the `sim()` function) as well as the predictive uncertainty.

```
prof <- read.csv("http://www.stat.columbia.edu/~gelman/arm/examples/beauty/ProfEvaltnsBeautyPublic.csv")

# convert into factors
prof$profnumber <- as.factor(prof$profnumber)
prof$female <- as.factor(prof$female)

# convert dummy `class*` variables into a factor
dummies <- prof[, 18:47]
prof$class <- factor(apply(dummies, FUN=function(r) r %*% 1:30, MARGIN=1))

# remove dummy variables
prof <- prof[-c(18:47)]

# normalise and centre professor evaluation (all other predictors are binary)
prof$c.profevaluation <- prof$profevaluation - mean(prof$profevaluation) / (2 * sd(prof$profevaluation))

# fit linear model
m1 <- lm(courseevaluation ~ female + onecredit + c.profevaluation*nonenglish, data=prof)
```

```r
display(m1)
```

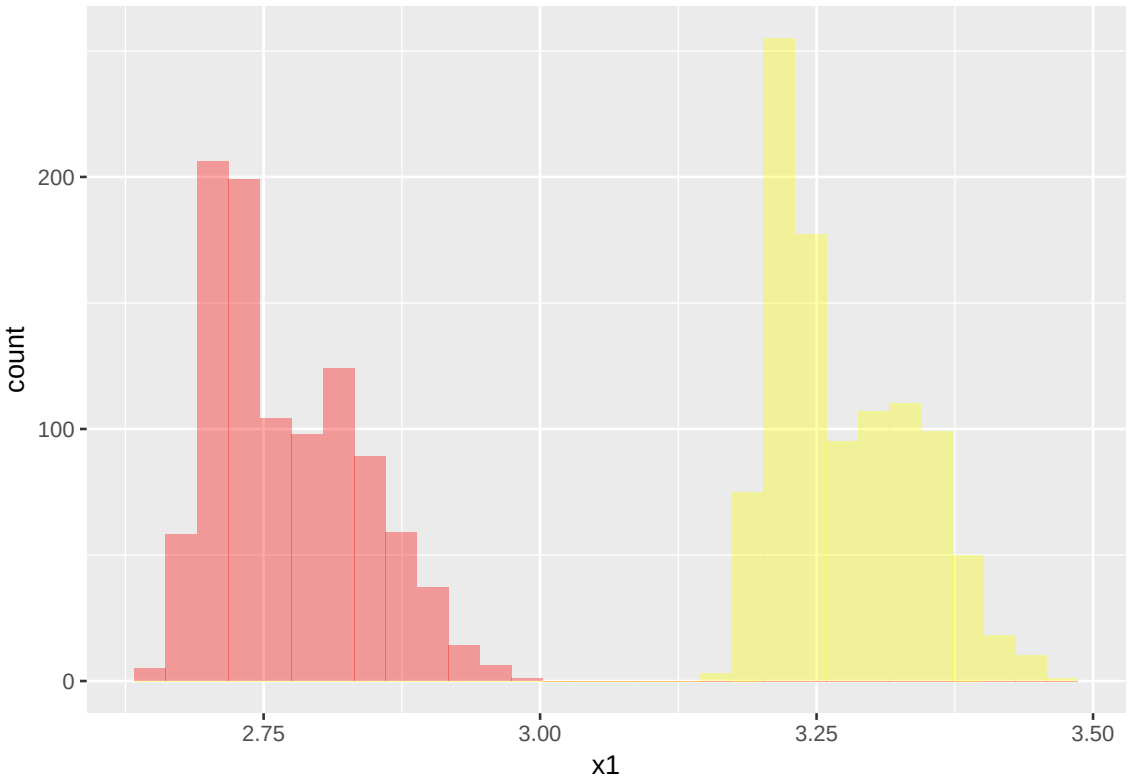
```
## lm(formula = courseevaluation ~ female + onecredit + c.profevaluation *
##     nonenglish, data = prof)
##                              coef.est coef.se
## (Intercept)                   3.69     0.01
## female1                      -0.03     0.02
## onecredit                     0.10     0.04
## c.profevaluation              0.94     0.02
## nonenglish                   -0.08     0.04
## c.profevaluation:nonenglish -0.17     0.09
## ---
## n = 463, k = 6
## residual sd = 0.19, R-Squared = 0.88
```

```r
n.sims <- 1000
sim1 <- sim(m1, n.sims)
# Instructor A: 50-year-old woman who is a native English speaker and has a beauty score of - 1
sim.outcome <- coef(sim1)[, 1] + 1 * coef(sim1)[, 2] +
    rep(c(0,1), 500) * coef(sim1)[, 3] + -1 *coef(sim1)[, 4] +
    0 * coef(sim1)[, 5] + -1 * 0 * coef(sim1)[, 6]

# Instructor B: 60-year-old man who is a native English speaker and has a beauty score of 0.5
sim.outcome2 <- coef(sim1)[, 1] + 0 * coef(sim1)[, 2] +
    rep(c(0,1), 500) * coef(sim1)[, 3] + -.5 * coef(sim1)[, 4] +
    0 * coef(sim1)[, 5] + -.5 * 0 * coef(sim1)[, 6]

ggplot(data=data.frame(x1=sim.outcome, x2=sim.outcome2)) +
    geom_histogram(aes(x=x1), fill="red", alpha=.35) +
    geom_histogram(aes(x=x2), fill="yellow", alpha=.35)
```

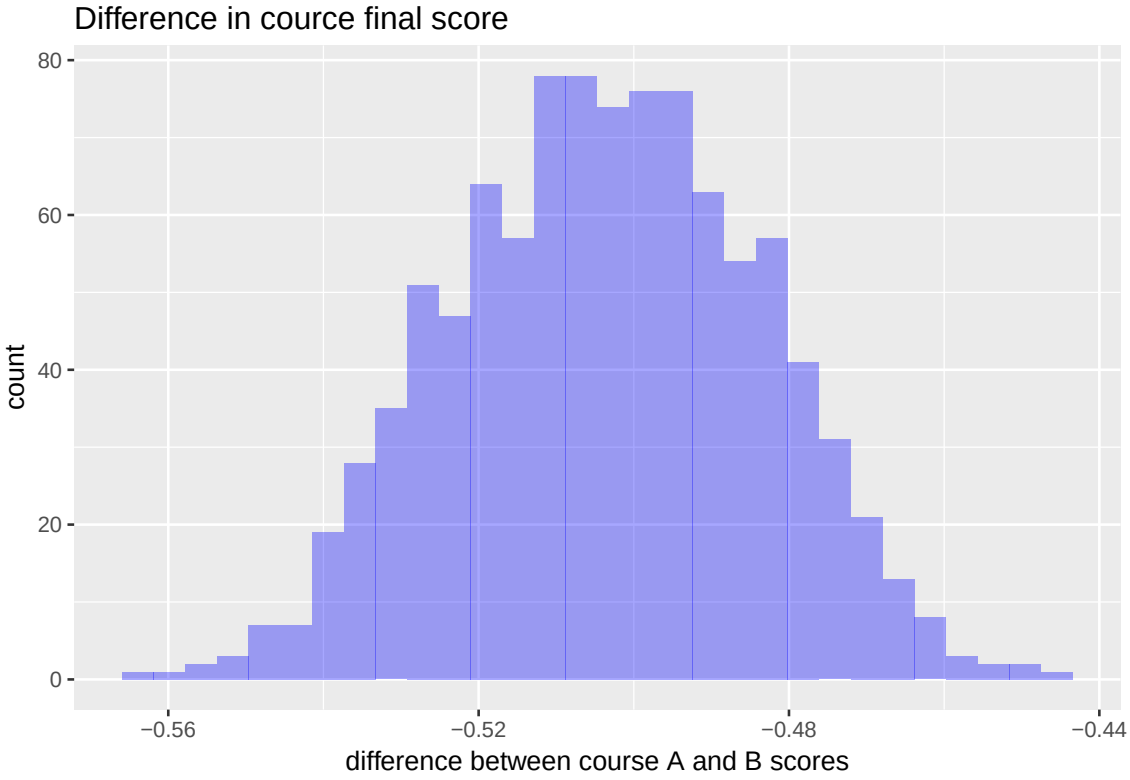
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



2. Make a histogram of the difference between the course evaluations for A and B. What is the probability that A will have a higher evaluation?

```
sim.diff <- sim.outcome - sim.outcome2
ggplot(data=data.frame(x=sim.diff)) +
  geom_histogram(aes(x=x), fill="blue", alpha=.35) +
    labs(title="Difference in cource final score",
         x="difference between course A and B scores")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



Predictive simulation for linear regression:

using data of interest to you, fit a linear regression model. Use the output from this model to simulate a predictive distribution for observations with a particular combination of levels of all the predictors in the regression.

I'll use the example from mortality rates and various environmental factors from 60 U.S. metropolitan areas from McDonald, G.C. and Schwing, R.C. (1973) 'Instabilities of regression estimates relating air pollution to mortality', Technometrics, vol.15, 463-482.

Variables, in order:

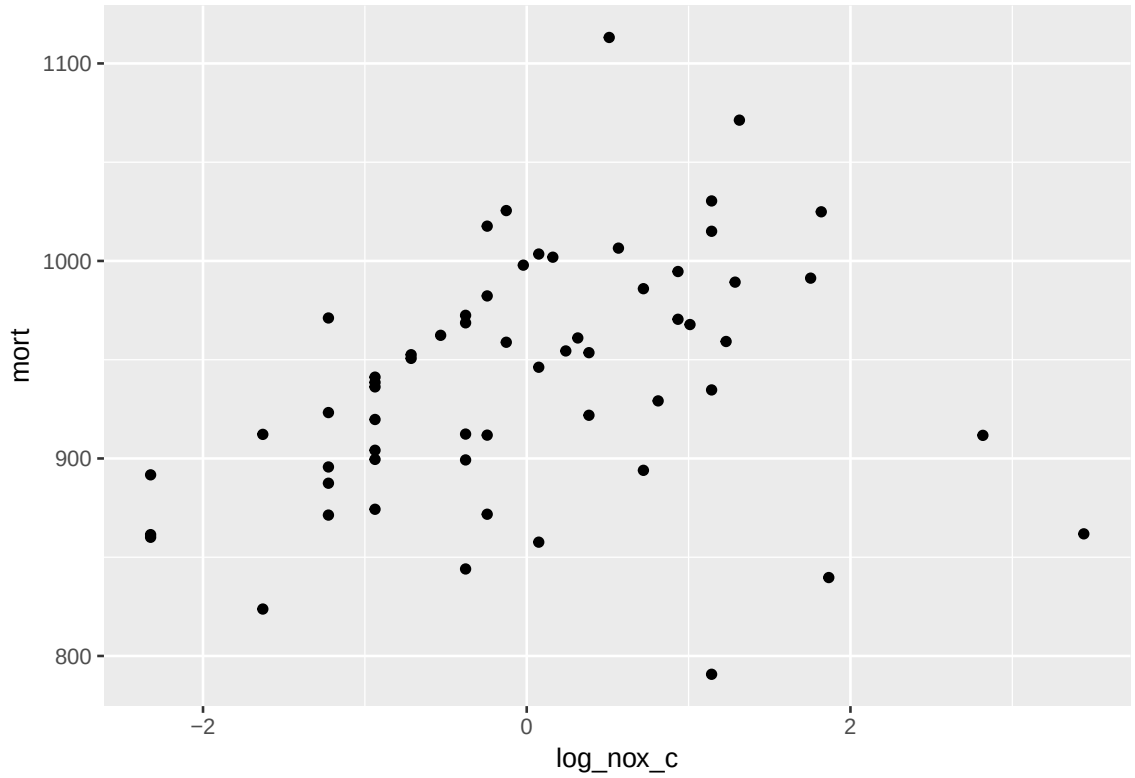
- PREC Average annual precipitation in inches
- JANT Average January temperature in degrees F
- JULT Same for July
- OVR65 % of 1960 SMSA population aged 65 or older
- POPN Average household size
- EDUC Median school years completed by those over 22
- HOUS % of housing units which are sound & with all facilities
- DENS Population per sq. mile in urbanized areas, 1960
- NONW % non-white population in urbanized areas, 1960
- WWDRK % employed in white collar occupations
- POOR % of families with income < $3000
- HC Relative hydrocarbon pollution potential
- NOX Same for nitric oxides
- SO@ Same for sulphur dioxide
- HUMID Annual average % relative humidity at 1pm
- MORT Total age-adjusted mortality rate per 100,000

In homework 2 the final model used NOX, SO2, and HC as the predictor for the mortality rate. I'll do a predictive simulation to see how good the final model fit was to the data.

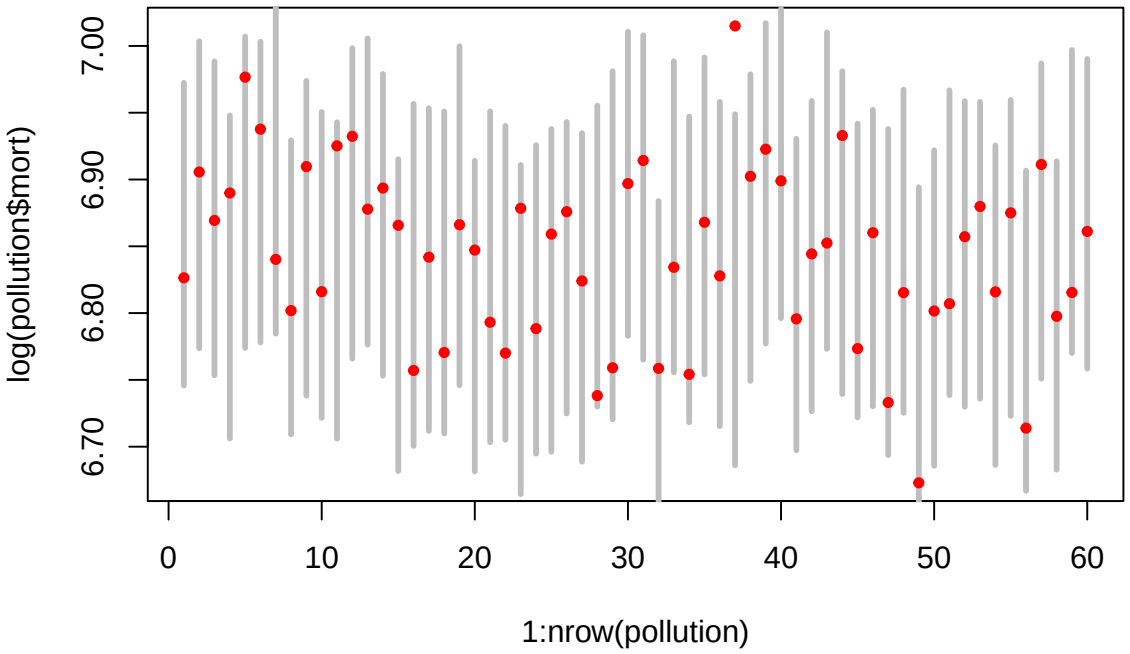
```
gelman_dir <- "http://www.stat.columbia.edu/~gelman/arm/examples/"
pollution <- data.table(read.dta (paste0(gelman_dir,"pollution/pollution.dta")))

pollution<-pollution[,log_nox_c:=scale(log(nox),center=TRUE,scale=FALSE)]
pollution<-pollution[,log_hc_c:=scale(log(hc),center=TRUE,scale=FALSE)]
pollution<-pollution[,log_so2_c:=scale(log(so2),center=TRUE,scale=FALSE)]

ggplot(pollution)+geom_point()+aes(x=log_nox_c,y=mort)
```



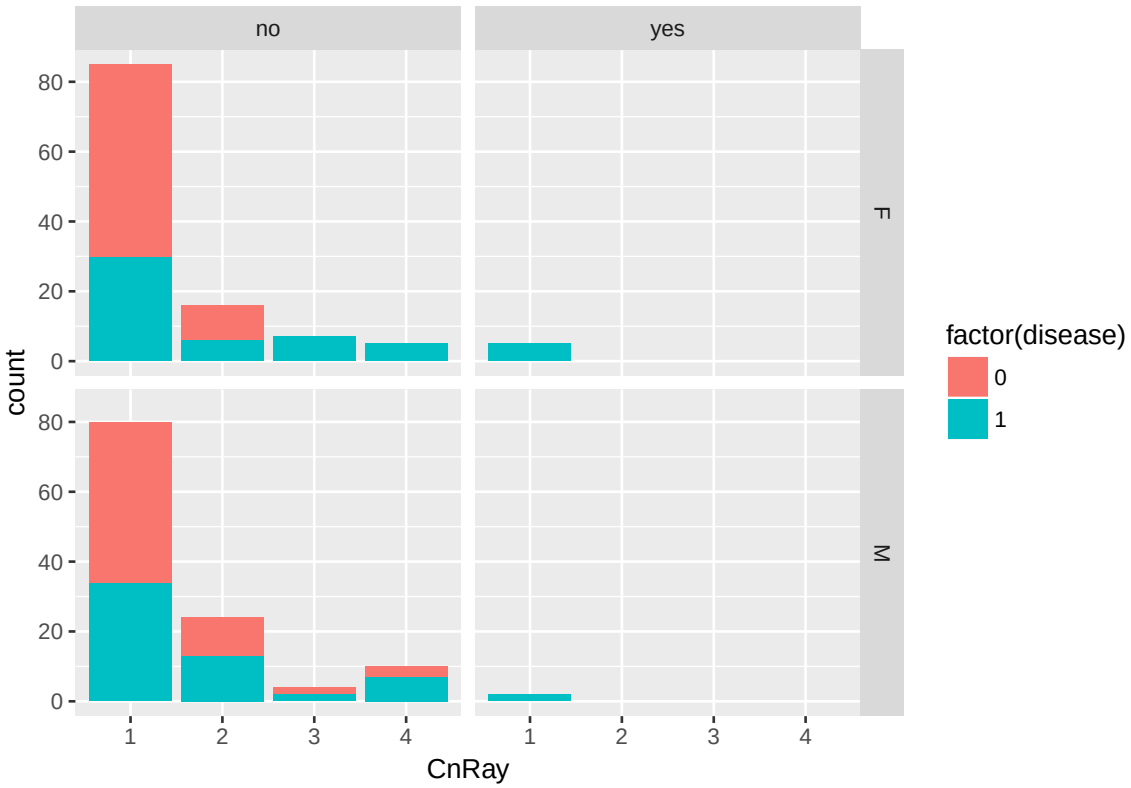

```
fit_mort_04<-pollution[,lm(log(mort)~log_nox_c+log_hc_c+log_so2_c)]
ssms<-sim(fit_mort_04,1000)
yhatt<-as.matrix(pollution[,list(1,log_nox_c,log_hc_c,log_so2_c)])%*%t(ssms@coef)
ytilde<-matrix(NA,nrow(pollution),1000)
for(i in 1:1000) ytilde[,i]<-rnorm(nrow(pollution),yhatt[,i],ssms@sigma[i])
qtt<-t(apply(ytilde,1,quantile,c(0.025,0.5,0.975)))
colnames(qtt)<-c("ymin","y","ymax")
plot(1:nrow(pollution),log(pollution$mort),type="n")
for( i in 1:nrow(pollution)){
 lines( c(i,i),c(qtt[i,"ymin"],qtt[i,"ymax"]),lwd=3,col="grey")
 points(i, log(pollution$mort)[i],pch=20,col="red")
}
```



Repeat the previous exercise using a logistic regression example.

We will look at Match pair study for AML and Xray link. A matched case control study was carried out to investigate the connection between X-ray usage and acute myeloid leukemia in childhood. The pairs are matched by age, race and county of residence.

```
library(faraway)
library(data.table)
data(amlxray)
amlxray_dt<-data.table(amlxray)
ggplot(amlxray)+geom_bar()+aes(x=CnRay,fill=factor(disease))+facet_grid(Sex~downs)
```

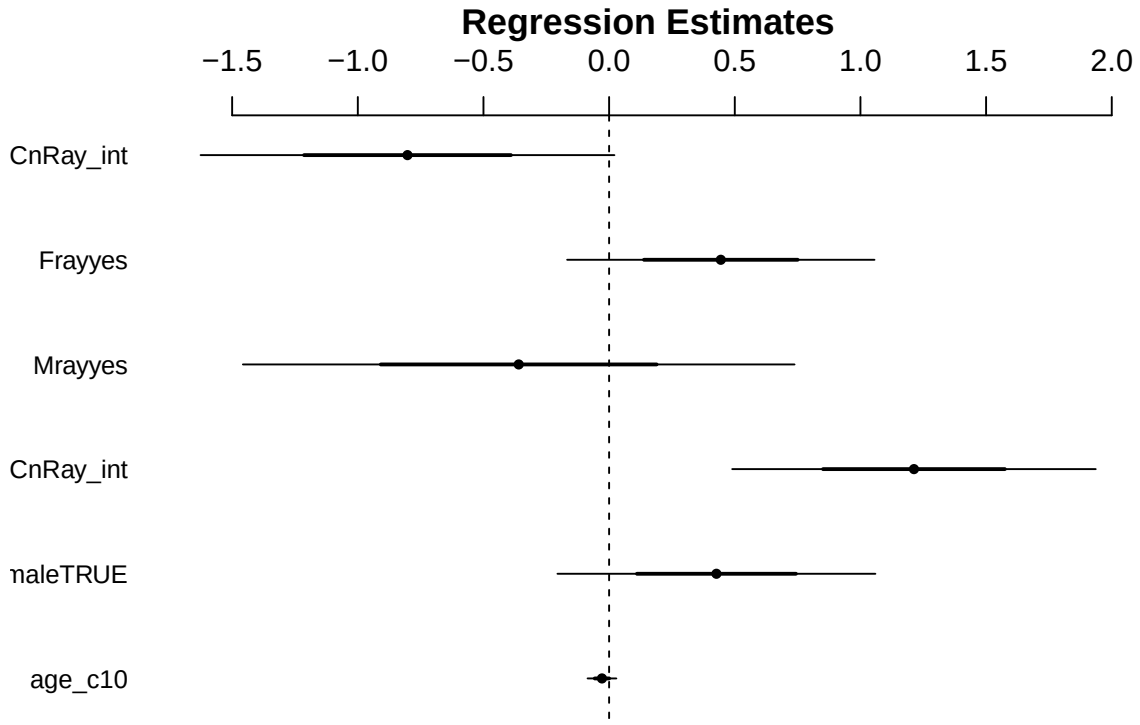


The down syndrome is a known risk factor in AML and will be removed from the following analysis since it will cause numerical problem.

```
amlxray$CnRay_int<- as.integer(amlxray$CnRay)-1
amlxray$age_c10<- amlxray$age-10
amlxray$male<- amlxray$Sex=="M"
amlxray_glm<-glm(disease~age_c10+male*CnRay_int+Mray+Fray,family=binomial(),
                 data=amlxray ,subset =amlxray$downs=="no")
display(amlxray_glm)
```

```
## glm(formula = disease ~ age_c10 + male * CnRay_int + Mray + Fray,
##     family = binomial(), data = amlxray, subset = amlxray$downs ==
##         "no")
##                     coef.est coef.se
## (Intercept)         -0.92     0.28
## age_c10             -0.03     0.03
## maleTRUE             0.43     0.32
## CnRay_int            1.21     0.36
## Mrayyes             -0.36     0.55
## Frayyes              0.44     0.31
## maleTRUE:CnRay_int -0.80     0.41
## ---
##   n = 231, k = 7
##   residual deviance = 295.2, null deviance = 317.9 (difference = 22.8)
```

```
coefplot(amlxray_glm)
```



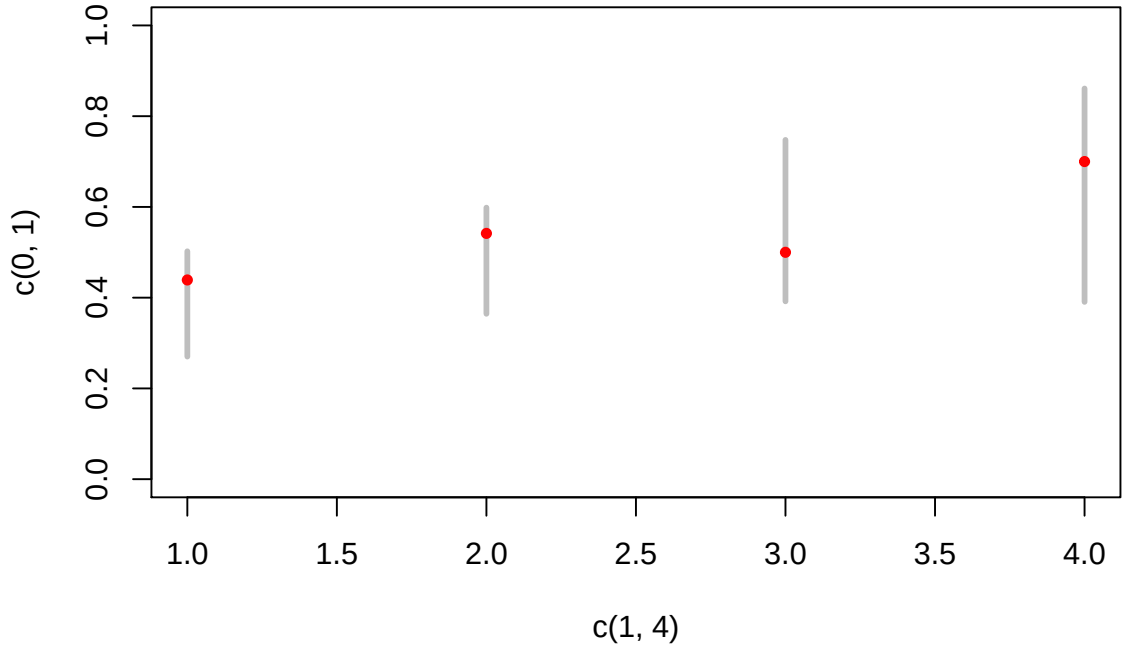

```
n.sim=1000
sim_amlxray_glm<-sim(amlxray_glm,n.sims=n.sim)
amlxray[amlxray$downs=="no" &amlxray$Mray=="no" & amlxray$Fray=="no"&amlxray$CnRay==4,]
```

```
##       ID disease Sex downs age Mray MupRay MlowRay Fray Cray CnRay
## 45  7036       1   M    no  13   no     no       no   no  yes     4
## 76  7060       1   F    no  15   no     no       no   no  yes     4
## 102 7084       1   M    no   3   no     no       no   no  yes     4
## 103 7084       0   M    no   3   no     no       no   no  yes     4
## 120 7101       1   M    no  10   no     no       no   no  yes     4
## 129 7109       0   M    no  14   no     no       no   no  yes     4
## 137 7118       1   F    no   5   no     no       no   no  yes     4
## 141 7120       1   M    no  15   no     no       no   no  yes     4
## 206 7186       1   M    no  14   no     no       no   no  yes     4
## 214 7193       1   F    no  13   no     no       no   no  yes     4
##     CnRay_int age_c10  male
## 45          3       3  TRUE
## 76          3       5 FALSE
## 102         3      -7  TRUE
## 103         3      -7  TRUE
## 120         3       0  TRUE
## 129         3       4  TRUE
## 137         3      -5 FALSE
## 141         3       5  TRUE
## 206         3       4  TRUE
## 214         3       3 FALSE
```

For a male, age 10, with 1:4 or more xray exposure with mother and father not having exposed.

```
predx<-as.matrix(expand.grid(1,0,1,0:3,0,0))
predx=cbind(predx,0:3)
py<-invlogit(sim_amlxray_glm@coef%*%t(predx))
```

```
dtobs<-amlxray_dt[,mean(disease),by=list(Sex,CnRay)][order(Sex,CnRay)][5:8]
dtobs$CnRay<-as.integer(dtobs$CnRay)
qqt<-apply(py,2,quantile,c(0.5,0.025,0.975))
qqtd<-t(qqt)
colnames(qqtd)=c("y","ymin","ymax")
qqtdt<-data.frame(CnRay=1:4,qqtd)
plot(c(1,4),c(0,1),type="n")
for( i in 1:4){
 lines( c(i,i),c(qqtdt[i,"ymin"],qqtdt[i,"ymax"]),lwd=3,col="grey")
 points(i, dtobs$V1[i],pch=20,col="red")
}
```




```
apply(py,2,mean)
```

```
## [1] 0.3795538 0.4778082 0.5761184 0.6626998
```

```
n.sims<-1000
ysim<-rbinom(n.sims*11,1,py[,4])
mean(ysim)
```

```
## [1] 0.6662727
```

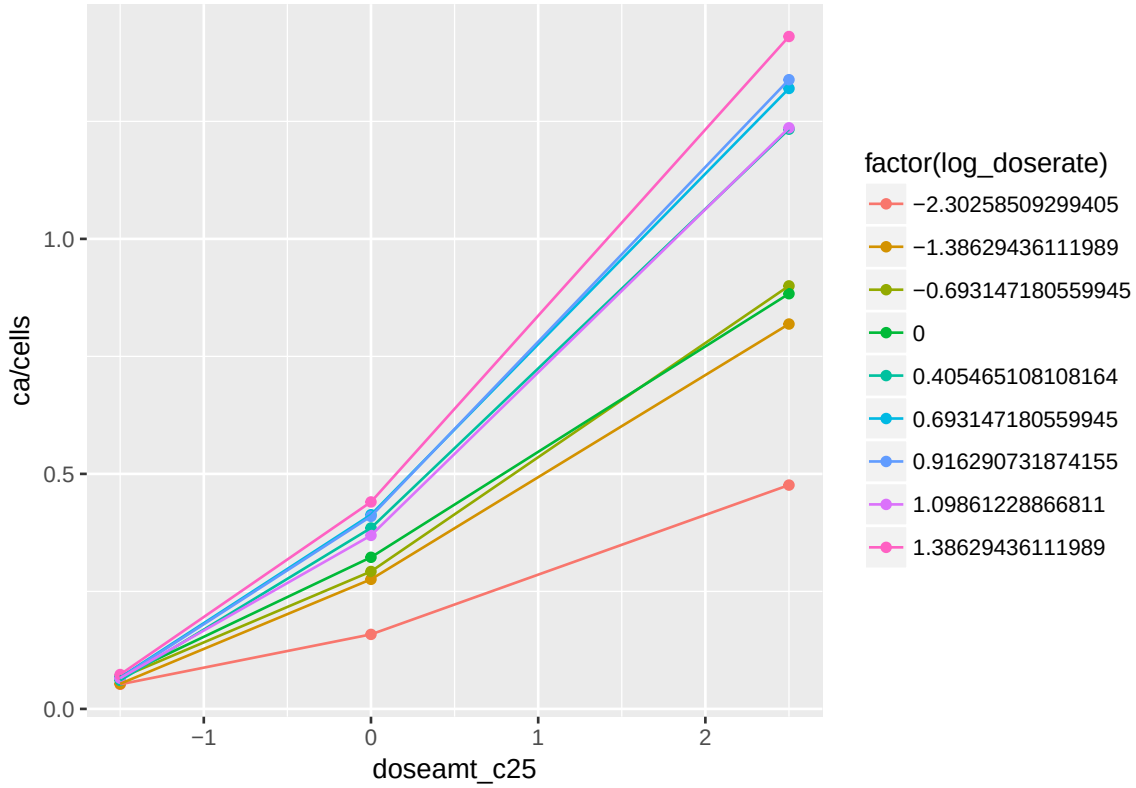
```
mean(amlxray[amlxray$downs=="no" &amlxray$Mray=="no" & amlxray$Fray=="no"&amlxray$CnRay==4&amlxray$male
```

```
## [1] 0.7142857
```

Repeat the previous exercise using a Poisson regression example.

An data from experiment conducted by Purott and Reeder (1976) to determine the effect of gamma radiation on the numbers of chromosomal abnormalities (ca) observed. The number (cells), in hundreds of cells exposed in each run, differs. The dose amount (doseamt) and the rate (doserate) at which the dose is applied are the predictors of interest.

```
data(dicentric)
dicentric$log_doserate<-log(dicentric$doserate)
dicentric$log_doserate_c <-dicentric$log_doserate
dicentric$doseamt_c25<-dicentric$doseamt-2.5
ggplot(dicentric)+geom_point()+
  aes(x=doseamt_c25,y=ca/cells,color=factor(log_doserate),group=factor(log_doserate))+geom_line()
```




```
rmod <- glm(ca ~ log_doserate_c*doseamt_c25,offset= log(cells),
  family=poisson,dicentric)
display(rmod)
```

```
## glm(formula = ca ~ log_doserate_c * doseamt_c25, family = poisson,
##     data = dicentric, offset = log(cells))
##                            coef.est coef.se
## (Intercept)                -1.56     0.02
## log_doserate_c              0.18     0.02
## doseamt_c25                 0.66     0.01
## log_doserate_c:doseamt_c25  0.03     0.01
## ---
##   n = 27, k = 4
##   residual deviance = 221.4, null deviance = 4753.0 (difference = 4531.6)
```

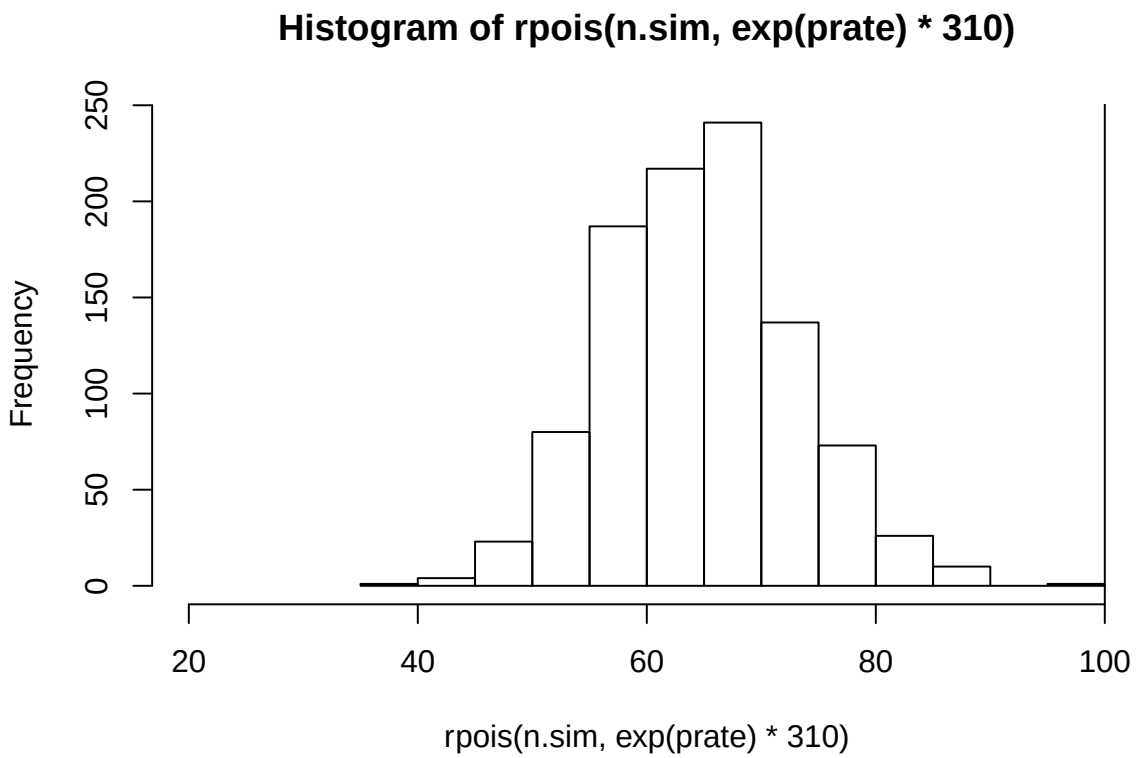
The predicted result show under prediction of the rate.

```
dicentric[dicentric$doseamt_c25==0&dicentric$log_doserate_c==0,]
```

```
##    cells  ca doseamt doserate log_doserate log_doserate_c doseamt_c25
## 13   310 100     2.5        1            0              0           0
```

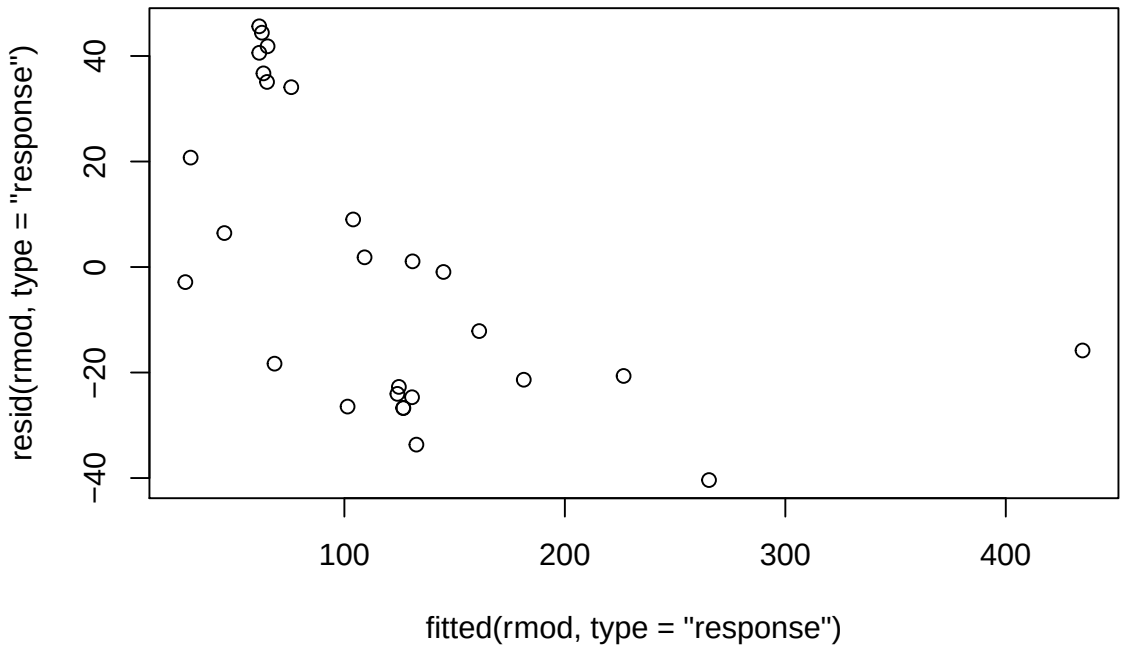
```
n.sim<-1000
prate<-sim(rmod,n.sims=n.sim)@coef%*%c(1,0,0,0)
```

```
hist(rpois(n.sim,exp(prate)*310),xlim=c(20,100)); abline(v=100)
```

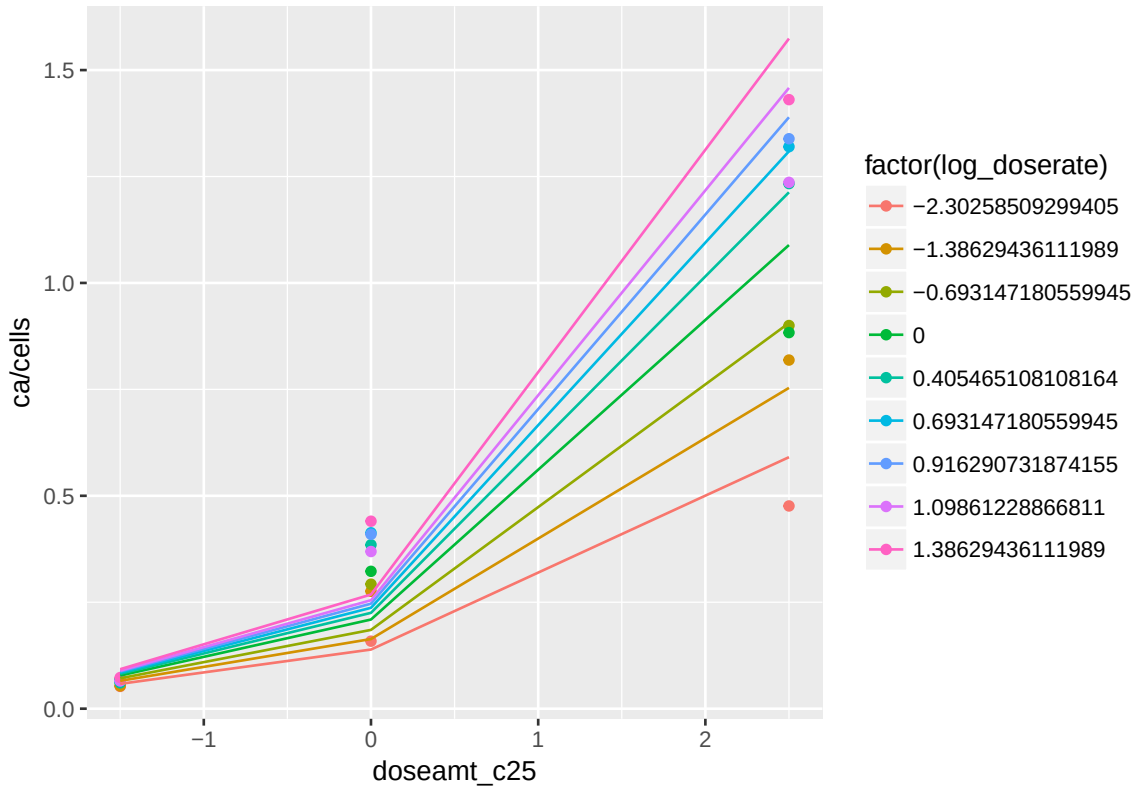


This can also be seen from the residuals.

```
dicentric$pred<-predict(rmod,type="response")
plot(fitted(rmod,type = "response"),resid(rmod,type = "response"))
```




```
ggplot(dicentric)+geom_point()+
  aes(x=doseamt_c25,y=ca/cells,color=factor(log_doserate),group=factor(log_doserate))+geom_line(aes(y=pr
```



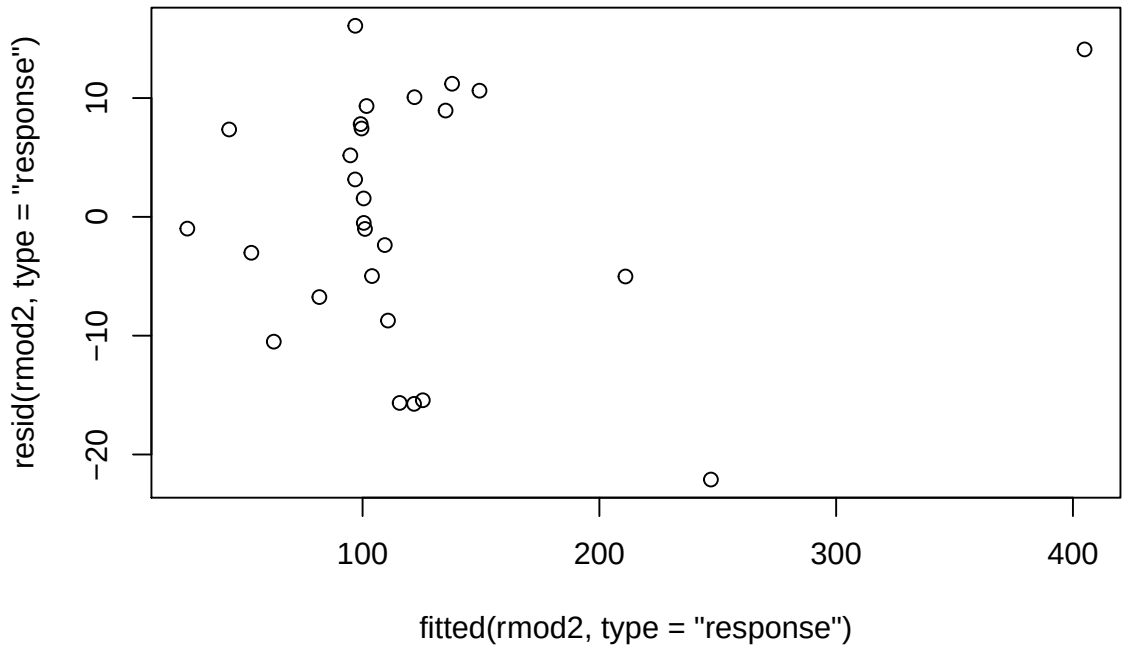
The result shows a hint of non linearity.

We can add quadratic term to the dose amount to account for the non linearity.

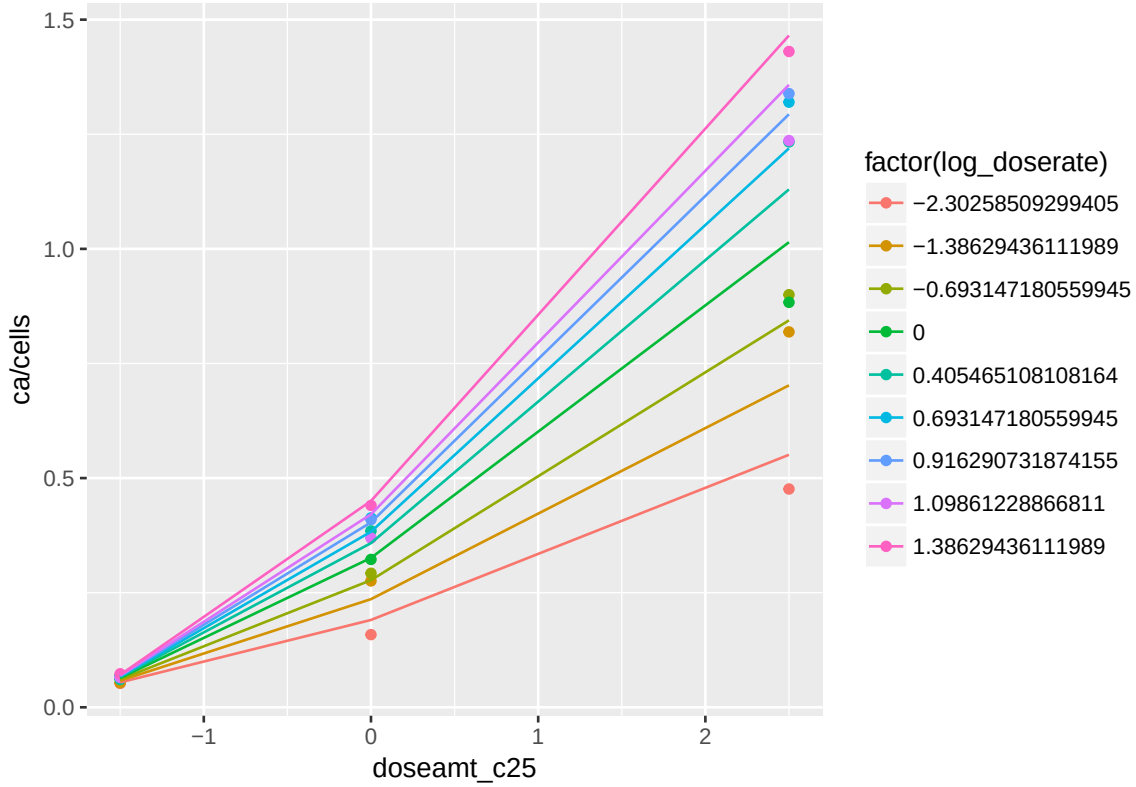
```
rmod2 <- glm(ca ~ log_doserate_c*poly(doseamt_c25,2),offset= log(cells),
  family=poisson,dicentric)
display(rmod2)
```

```
## glm(formula = ca ~ log_doserate_c * poly(doseamt_c25, 2), family = poisson,
##     data = dicentric, offset = log(cells))
##                                     coef.est coef.se
## (Intercept)                         -1.28     0.02
## log_doserate_c                       0.19     0.02
## poly(doseamt_c25, 2)1                5.71     0.09
## poly(doseamt_c25, 2)2               -1.43     0.10
## log_doserate_c:poly(doseamt_c25, 2)1  0.38     0.09
## log_doserate_c:poly(doseamt_c25, 2)2 -0.21     0.10
## ---
##   n = 27, k = 6
##   residual deviance = 21.7, null deviance = 4753.0 (difference = 4731.3)
```

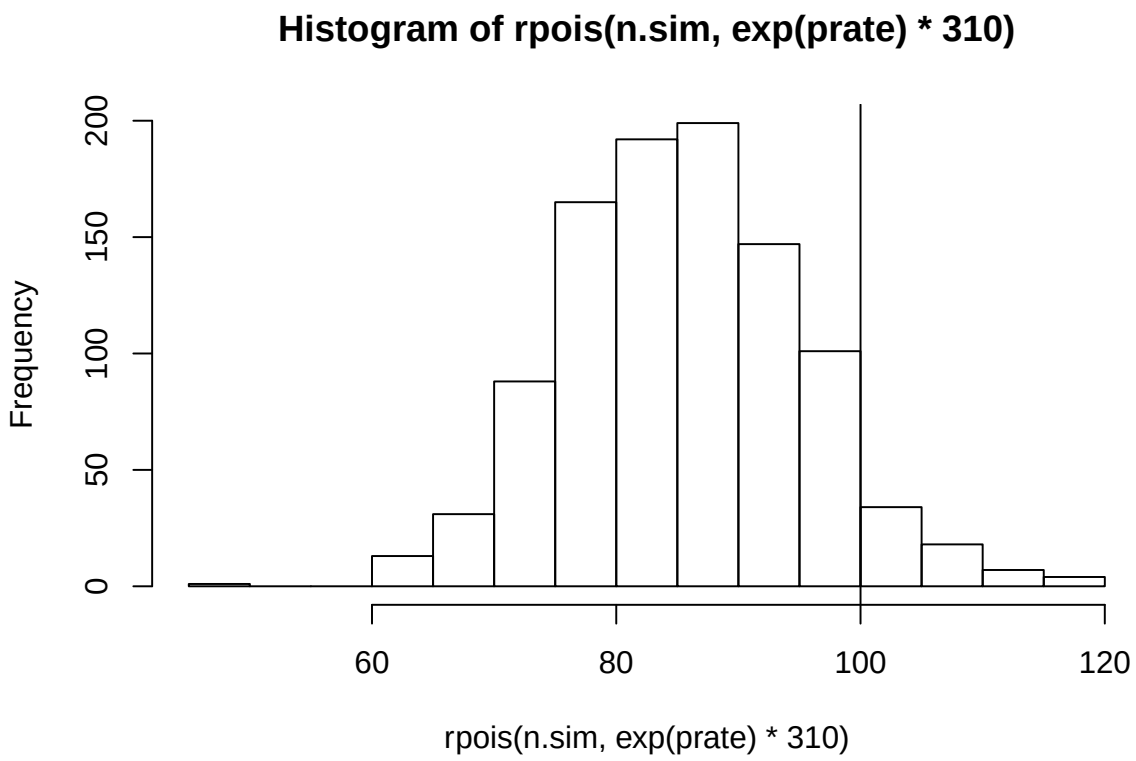
```
dicentric$pred<-predict(rmod2,type="response")
plot(fitted(rmod2,type = "response"),resid(rmod2,type = "response"))
```




```
ggplot(dicentric)+geom_point()+
  aes(x=doseamt_c25,y=ca/cells,color=factor(log_doserate),group=factor(log_doserate))+geom_line(aes(y=p
```




```
n.sim<-1000
prate<-sim(rmod2,n.sims=n.sim)@coef%*%c(1,0,0,0,0,0)
hist(rpois(n.sim,exp(prate)*310)); abline(v=100)
```



Inference for the ratio of parameters:

a (hypothetical) study compares the costs and effectiveness of two different medical treatments. - In the first part of the study, the difference in costs between treatments A and B is estimated at $600 per patient, with a standard error of $400, based on a regression with 50 degrees of freedom. - In the second part of the study, the difference in effectiveness is estimated at 3.0 (on some relevant measure), with a standard error of 1.0, based on a regression with 100 degrees of freedom. - For simplicity, assume that the data from the two parts of the study were collected independently.

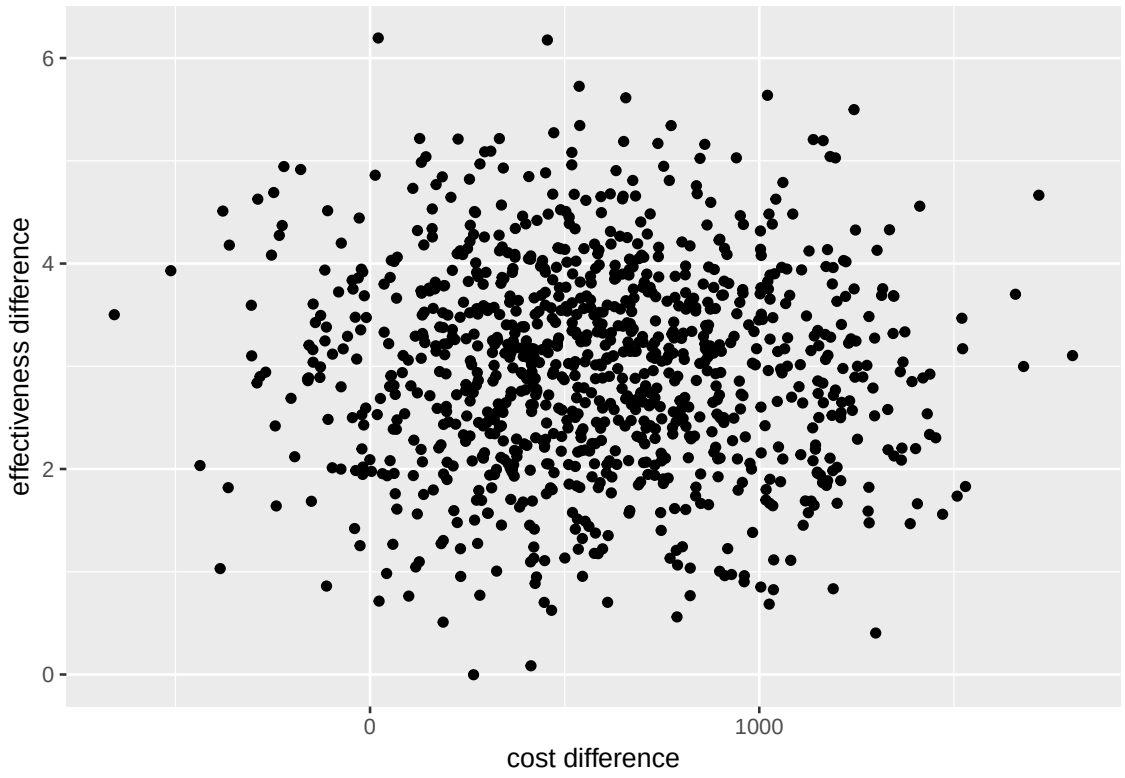
Inference is desired for the incremental cost-effectiveness ratio: the difference between the average costs of the two treatments, divided by the difference between their average effectiveness. (This problem is discussed further by Heitjan, Moskowitz, and Whang, 1999.)

1. Create 1000 simulation draws of the cost difference and the effectiveness difference, and make a scatterplot of these draws.

```
n.sims      <- 1000
cost.mean   <-  600
cost.sd     <-  400
effect.mean <-    3.0
effect.sd   <-    1.0

costs   <- rnorm(n=n.sims, mean=cost.mean,   sd=cost.sd  )
effects <- rnorm(n=n.sims, mean=effect.mean, sd=effect.sd)

ggplot(data.frame(cost=costs, effect=effects))+
  aes(x=cost, y=effect) +
    geom_point() +
  labs(x="cost difference", y="effectiveness difference")
```



2. Use simulation to come up with an estimate, 50% interval, and 95% interval for the incremental cost-effectiveness ratio.

```
cost.effectiveness <- costs/effects

mean(cost.effectiveness)
```

```
## [1] 119.5378
```

```
quantile(cost.effectiveness, c(.25, .75))
```

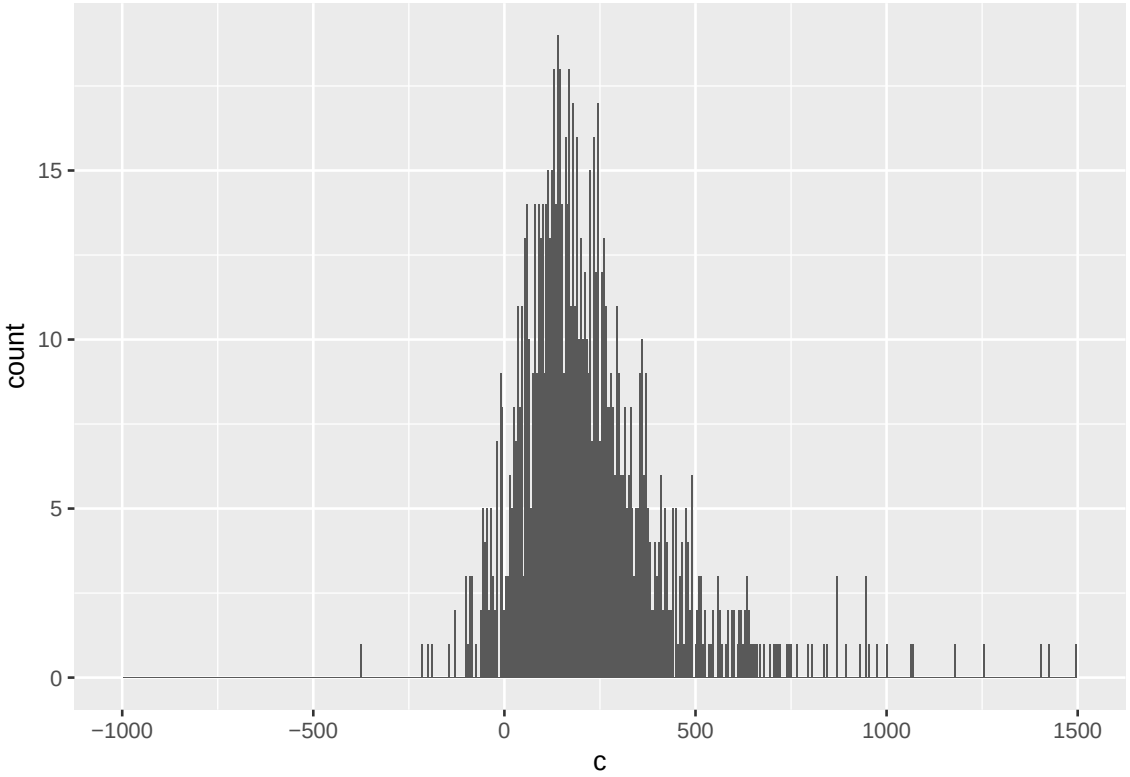
```
##      25%      75%
## 101.8544 304.4406
```

```
quantile(cost.effectiveness, c(.025, .975))
```

```
##      2.5%     97.5%
## -52.77796 750.44633
```

```
ggplot(data=data.frame(c=cost.effectiveness), aes(x=c)) +
    geom_histogram(binwidth=5) + xlim(c(-1000, 1500))
```

```
## Warning: Removed 3 rows containing non-finite values (stat_bin).
```



3. Repeat this problem, changing the standard error on the difference in effectiveness to 2.0.

```
effects <- rnorm(n=n.sims, mean=effect.mean, sd=2.0)
cost.effectiveness <- costs/effects

mean(cost.effectiveness)
```

```
## [1] 290.355
```

```
quantile(cost.effectiveness, c(.25, .75))
```

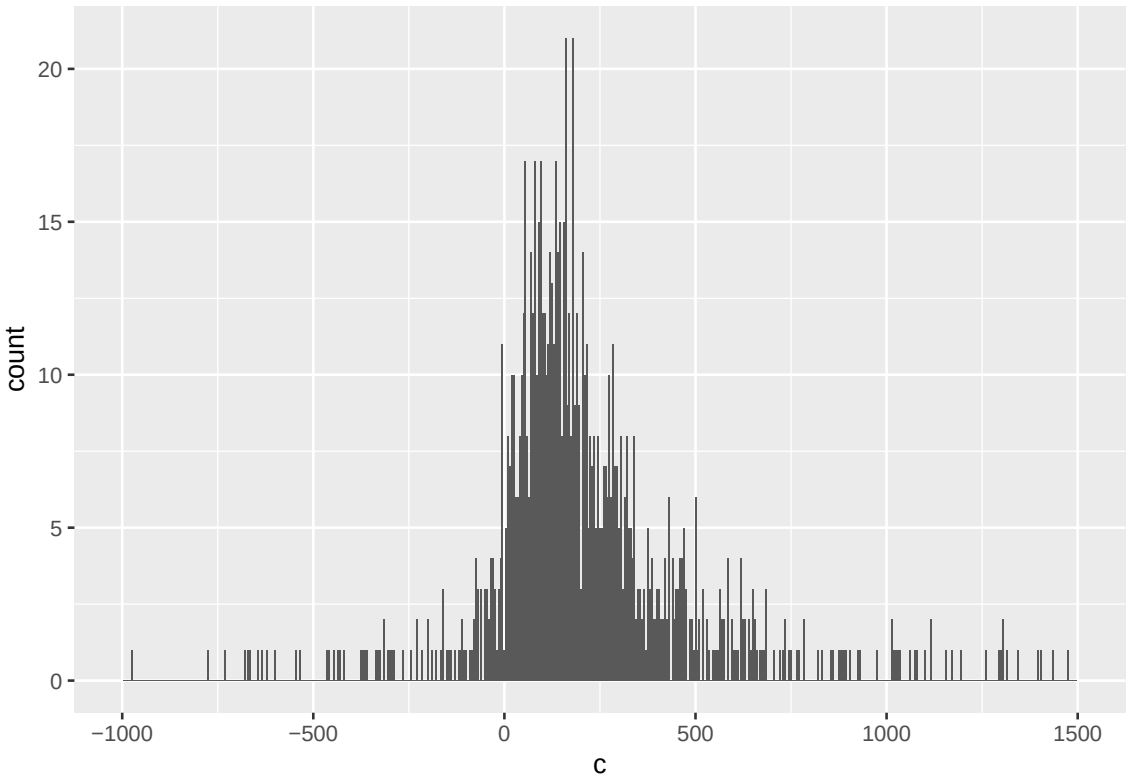
```
##       25%       75%
##  69.29057 309.72589
```

```
quantile(cost.effectiveness, c(.025, .975))
```

```
##      2.5%     97.5%
## -1076.410  1317.365
```

```
ggplot(data.frame(c=cost.effectiveness))+ aes(x=c) +
    geom_histogram(binwidth=5) + xlim(c(-1000, 1500))
```

```
## Warning: Removed 47 rows containing non-finite values (stat_bin).
```



Summarizing inferences and predictions using simulation:

Exercise 6.5 used a Tobit model to fit a regression with an outcome that had mixed discrete and continuous data. In this exercise you will revisit these data and build a two-step model: (1) logistic regression for zero earnings versus positive earnings, and (2) linear regression for level of earnings given earnings are positive. Compare predictions that result from each of these models with each other.

```
nsw <- read.dta("http://www.stat.columbia.edu/~gelman/arm/examples/lalonde/NSW.dw.obs.dta")

# create factor variables
nsw$sample <- factor(nsw$sample, labels=c("NSW", "CPS", "PSID"))
nsw$black  <- factor(nsw$black)
nsw$hisp   <- factor(nsw$hisp)
nsw$nodegree  <- factor(nsw$nodegree)
nsw$married   <- factor(nsw$married)
nsw$treat     <- factor(nsw$treat)
nsw$educ_cat4 <- factor(nsw$educ_cat4, labels=c("less than high school", "high school", "sm college", "

nsw$c_age <- scale(nsw$age, center=TRUE)
nsw$c_educ<- scale(nsw$educ,center=TRUE)
nsw$c_re74<- scale(nsw$re74,center=TRUE)
nsw$c_re75<- scale(nsw$re75,center=TRUE)

# create a dummy variable to represent when re78 is greater than 0
nsw$earn.pos <- ifelse(nsw$re78>0, 1, 0)
```

fit the models;

```
fit1.a <- glm(earn.pos ~ c_age + c_educ + c_re75 + black + married, family=binomial(link="logit"), data=
fit1.b <- lm(re78 ~ c_age + c_educ + c_re75 + black + married, data=nsw, subset=earn.pos>0)

fit_tobit <-vglm(re78 ~ c_age + c_educ + c_re75, tobit(Lower=0, Upper=25563), data=nsw, subset=re78<255
summary(fit_tobit)
```

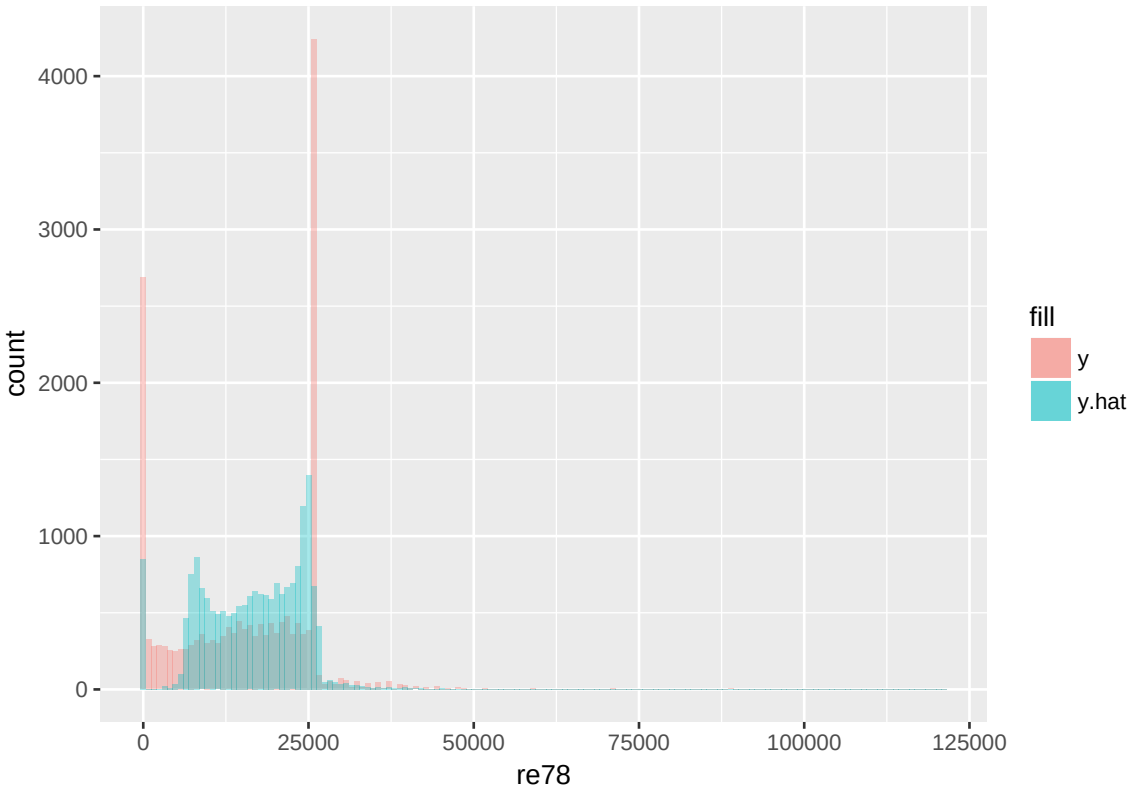
```
##
## Call:
## vglm(formula = re78 ~ c_age + c_educ + c_re75, family = tobit(Lower = 0,
##     Upper = 25563), data = nsw, subset = re78 < 25564)
##
##
## Pearson residuals:
##                 Min      1Q  Median     3Q    Max
## mu       -137.6002 -0.7610  0.1368 0.6831  2.034
## loge(sd)   -0.9974 -0.7217 -0.4189 0.2442 72.432
##
## Coefficients:
##                 Estimate Std. Error  z value Pr(>|z|)
## (Intercept):1  1.237e+04  7.933e+01  155.976  < 2e-16 ***
## (Intercept):2  9.027e+00  7.283e-03 1239.450  < 2e-16 ***
## c_age         -1.654e+03  7.666e+01  -21.575  < 2e-16 ***
## c_educ        -3.270e+02  7.551e+01   -4.331 1.49e-05 ***
## c_re75         6.808e+03  9.540e+01   71.368  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Number of linear predictors:  2
##
## Names of linear predictors: mu, loge(sd)
##
## Log-likelihood: -118527.5 on 27221 degrees of freedom
##
## Number of iterations: 5
```

```
# make predictions using training data
y.hat <- ifelse(predict(fit1.a, newdata=nsw, type="response")<0.5, 0, predict(fit1.b, newdata=nsw))
```

```
# compute RMSE
y <- nsw$re78
print(paste0("RMSE: ", sprintf("%.2f", sqrt(mean((y-y.hat)**2)))))
```

```
## [1] "RMSE: 7907.55"
```

```
ggplot(data=data.frame(cbind(nsw, y.hat=y.hat))) +
    geom_histogram(aes(x=re78, fill="y"), alpha=.35, binwidth=(range(nsw$re78)[2] - range(nsw$re78)[1])
    geom_histogram(aes(x=y.hat, fill="y.hat"), alpha=.35, binwidth=(range(nsw$re78)[2] - range(nsw$re78
```



How many simulation draws are needed:

take the model from previous exercise that predicts course evaluations from beauty and other input variables. Use display() to summarize the model fit. Focus on the estimate and standard error for the coefficient of beauty.

```
beauty <- read.csv("http://www.stat.columbia.edu/~gelman/arm/examples/beauty/ProfEvaltnsBeautyPublic.cs
fit1 <- lm(btystdave ~ courseevaluation + female + age, data=beauty)
```

1. Use sim() with n.sims = 10000. Compute the mean and standard deviations of the 1000 simulations of the coefficient of beauty, and check that these are close to the output from display.

```
display(fit1)
```

```
## lm(formula = btystdave ~ courseevaluation + female + age, data = beauty)
##                   coef.est coef.se
## (Intercept)       -0.17     0.34
## courseevaluation   0.27     0.06
## female             0.12     0.07
## age               -0.02     0.00
## ---
## n = 463, k = 4
## residual sd = 0.74, R-Squared = 0.12
```

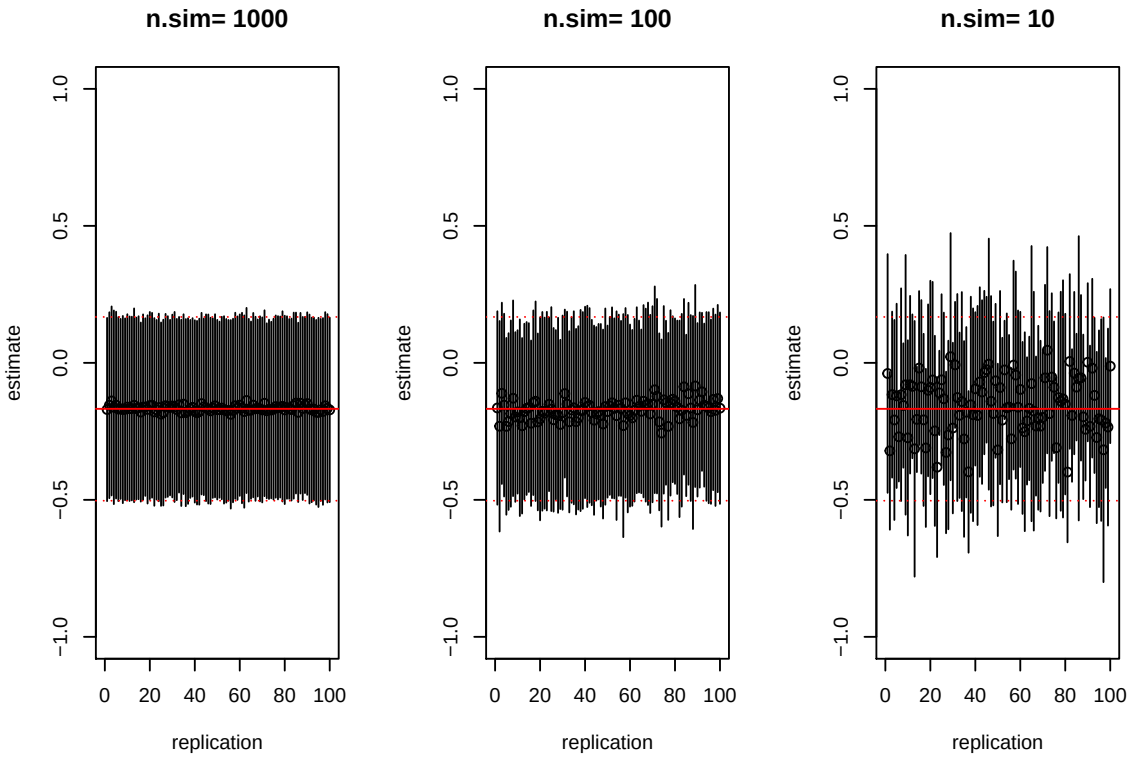
```
sim.fit1<-sim(fit1,n.sims=10000)
round(data.frame(coef.est=colMeans(sim.fit1@coef),
                 coef.se=apply(sim.fit1@coef,2,sd)),2)
```

```
##                   coef.est coef.se
```

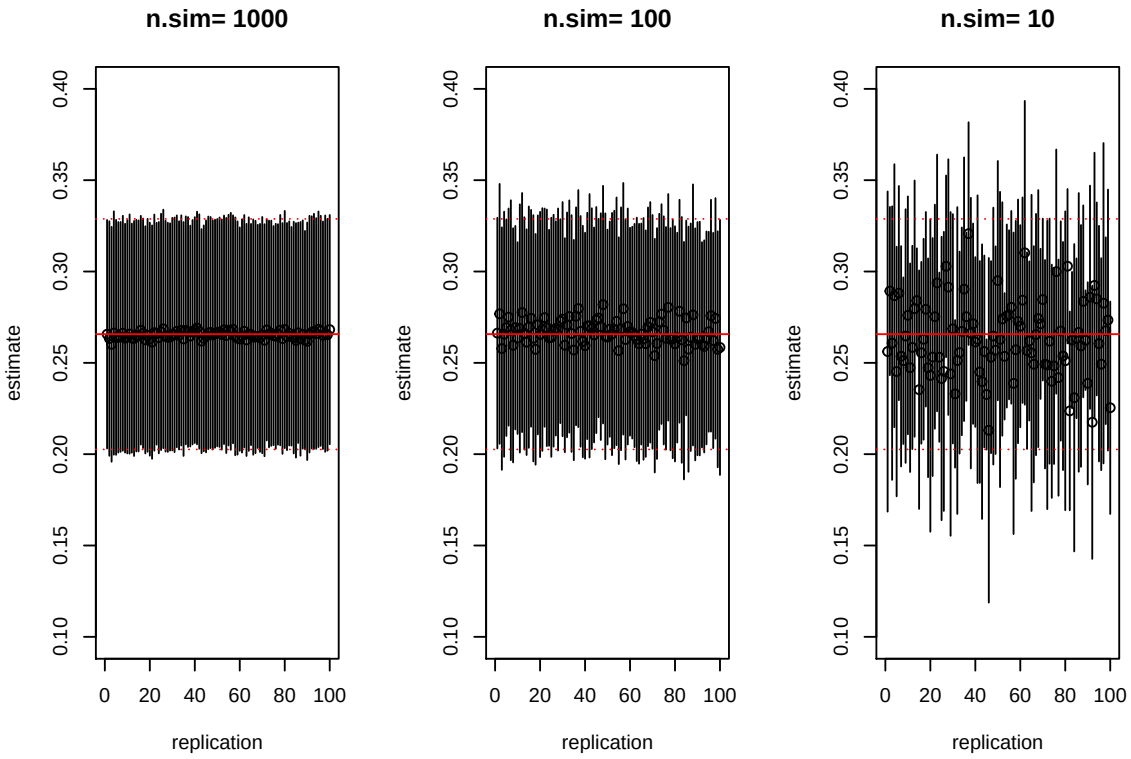
```
## (Intercept)         -0.17    0.34
## courseevaluation     0.27    0.06
## female               0.12    0.07
## age                 -0.02    0.00
```

2. Repeat with n.iter = 1000, n.iter = 100, and n.iter = 10. Do each of these a few times in order to get a sense of the simulation variability.

```
reslist<- vector("list",3)
for(i in 1:3) reslist[[i]] <- array(NA,c(4,2,100))
for(k in 1:100){
  for( ii in 1:3){
    sim.fit1<-sim(fit1,n.sims=c(1000,100,10)[ii])
    reslist[[ii]][,,k]<-cbind(coef.est=colMeans(sim.fit1@coef),
                              coef.se=apply(sim.fit1@coef,2,sd))
  }
}
par(mfrow=c(1,3))
for( ii in 1:3){
plot(0,0,type="n",xlim=c(0,100),
     xlab="replication",ylab="estimate",
     main=paste("n.sim=",c(1000,100,10)[ii]))
points(1:100,reslist[[ii]][1,1,])
for(jj in 1:100) lines(c(jj,jj),
                       c(reslist[[ii]][1,1,jj]+reslist[[ii]][1,2,jj],
                       reslist[[ii]][1,1,jj]-reslist[[ii]][1,2,jj]))
abline(h=summary(fit1)$coefficients[1,1],col='red')
abline(h=summary(fit1)$coefficients[1,1]+
         summary(fit1)$coefficients[1,2],col='red',lty=3)
abline(h=summary(fit1)$coefficients[1,1]-
         summary(fit1)$coefficients[1,2],col='red',lty=3)
}
```




```
par(mfrow=c(1,3))
for( ii in 1:3){
plot(0,0,type="n",xlim=c(0,100),ylim=c(0.1,0.4),
     xlab="replication",ylab="estimate",
     main=paste("n.sim=",c(1000,100,10)[ii]))
points(1:100,reslist[[ii]][2,1,])
for(jj in 1:100) lines(c(jj,jj),
                       c(reslist[[ii]][2,1,jj]+reslist[[ii]][2,2,jj],
                       reslist[[ii]][2,1,jj]-reslist[[ii]][2,2,jj]))
abline(h=summary(fit1)$coefficients[2,1],col='red')
abline(h=summary(fit1)$coefficients[2,1]+
         summary(fit1)$coefficients[2,2],col='red',lty=3)
abline(h=summary(fit1)$coefficients[2,1]-
         summary(fit1)$coefficients[2,2],col='red',lty=3)
}
```



3. How many simulations were needed to give a good approximation to the mean and standard error for the coefficient of beauty?

about 1000 should be enough.

Fitting the wrong model:

suppose you have 100 data points that arose from the following model: $y = 3 + 0.1x_1 + 0.5x_2 + error$, with errors having a t distribution with mean 0, scale 5, and 4 degrees of freedom. We shall explore the implications of fitting a standard linear regression to these data.

1. Simulate data from this model. For simplicity, suppose the values of `x_1` are simply the integers from 1 to 100, and that the values of `x_2` are random and equally likely to be 0 or 1. In R, you can define `x_1 <- 1:100`, simulate `x_2` using `rbinom()`, then create the linear predictor, and finally simulate the random errors in `y` using the `rt()` function. Fit a linear regression (with normal errors) to these data and see if the 68% confidence intervals for the regression coefficients (for each, the estimates ±1 standard error) cover the true values.

```
x_1       <- 1:100
x_2       <- rbinom(100,1,0.5)
er        <- rt(100, df=4)*5
y         <- 3+ 0.1*x_1 + 0.5*x_2 + er
stmp      <- summary(lm(y~x_1+x_2))$coef
cbind( stmp[,1]-stmp[,2],stmp[,1]+stmp[,2])
```

```
##                    [,1]       [,2]
## (Intercept)  1.89955310 6.77980408
## x_1          0.01656991 0.09235194
## x_2         -0.28871947 4.10040694
```

2. Put the above step in a loop and repeat 1000 times. Calculate the confidence coverage for the 68% intervals for each of the three coefficients in the model.

```
# result intervals
resci<-array(NA,c(3,2,1000))
for( i in 1:1000){
  x_1       <- 1:100
  x_2       <- rbinom(100,1,0.5)
  er        <- rt(100, df=4)*5
  y         <- 3+ 0.1*x_1 + 0.5*x_2 + er
  stmp      <- summary(lm(y~x_1+x_2))$coef
  resci[,,i]<- cbind( stmp[,1]-stmp[,2],stmp[,1]+stmp[,2])
}
mean(  3 > resci[1,1,]   &   3 < resci[1,2,])
```

```
## [1] 0.681
```

```
mean(0.1 > resci[2,1,]   & 0.1 < resci[2,2,])
```

```
## [1] 0.679
```

```
mean(0.5 > resci[3,1,]   & 0.5 < resci[3,2,])
```

```
## [1] 0.684
```

3. Repeat this simulation, but instead fit the model using t errors (see Exercise 6.6).

```
library(hett)
```

```
## Loading required package: lattice
```

```
##
## Attaching package: 'lattice'
```

```
## The following object is masked from 'package:faraway':
##
##     melanoma
```

```
resci<-array(NA,c(3,2,1000))
rtci<-array(NA,c(3,2,1000))
for( i in 1:1000){
  x_1       <- 1:100
  x_2       <- rbinom(100,1,0.5)
  er        <- rt(100, df=4)*5
  y         <- 3+ 0.1*x_1 + 0.5*x_2 + er
  ctmp <- summary(hett::tlm( y~x_1+x_2, start = list(dof = 4) ),
                  obs = F)$loc.summary$coefficients
 rtci[,,i] <- cbind( ctmp[,1]-ctmp[,2],ctmp[,1]+ctmp[,2])
}

mean(  3 > rtci[1,1,] &   3 < rtci[1,2,])
```

```
## [1] 0.658
```

```
mean(0.1 > rtci[2,1,] & 0.1 < rtci[2,2,])
```

```
## [1] 0.67
```

```
mean(0.5 > rtci[3,1,] & 0.5 < rtci[3,2,])
```

```
## [1] 0.653
```

Predictive checks:

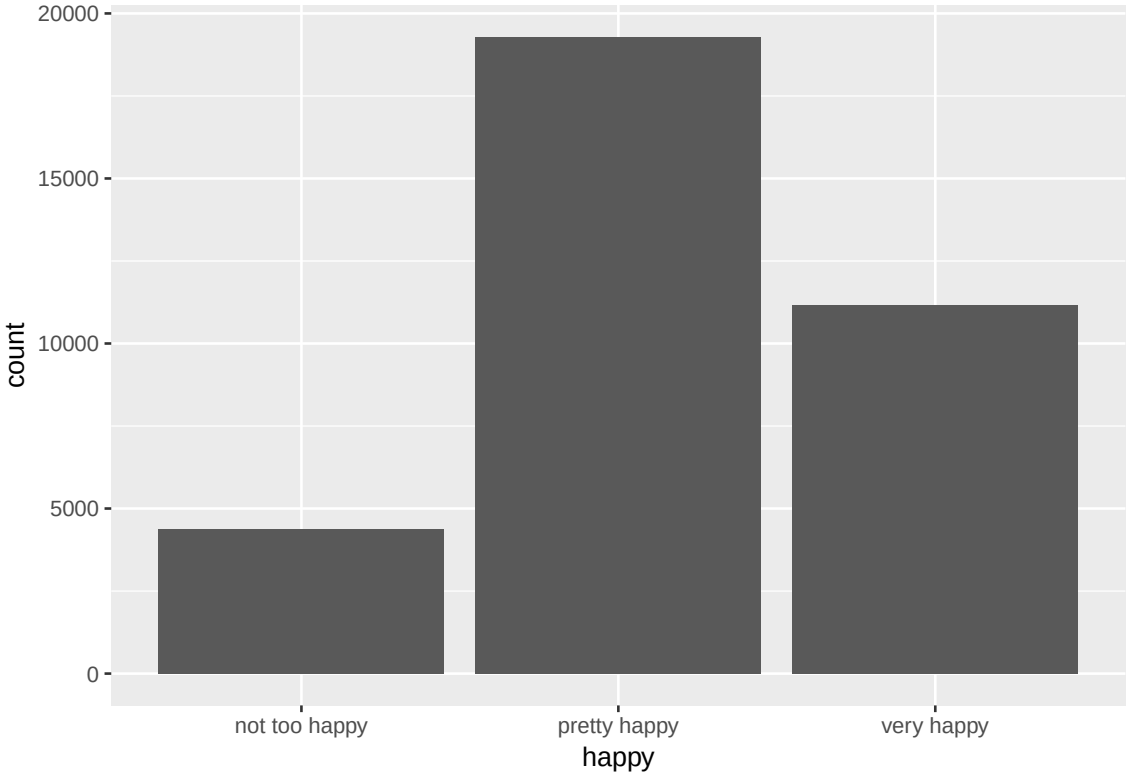
using data of interest to you, fit a model of interest. 1. Simulate replicated datasets and visually compare to the actual data.

```
library(GGally)
```

```
##
## Attaching package: 'GGally'
```

```
## The following object is masked from 'package:faraway':
##
##     happy
```

```
data(happy,package = "GGally")
happy<- happy[complete.cases(happy),]
happyfit<-polr(factor(happy)~age+finrela+degree+health+marital+sex,data=happy)
ggplot(happy)+geom_bar()+aes(x=happy)
```




```
simfit<-sim(happyfit)
```

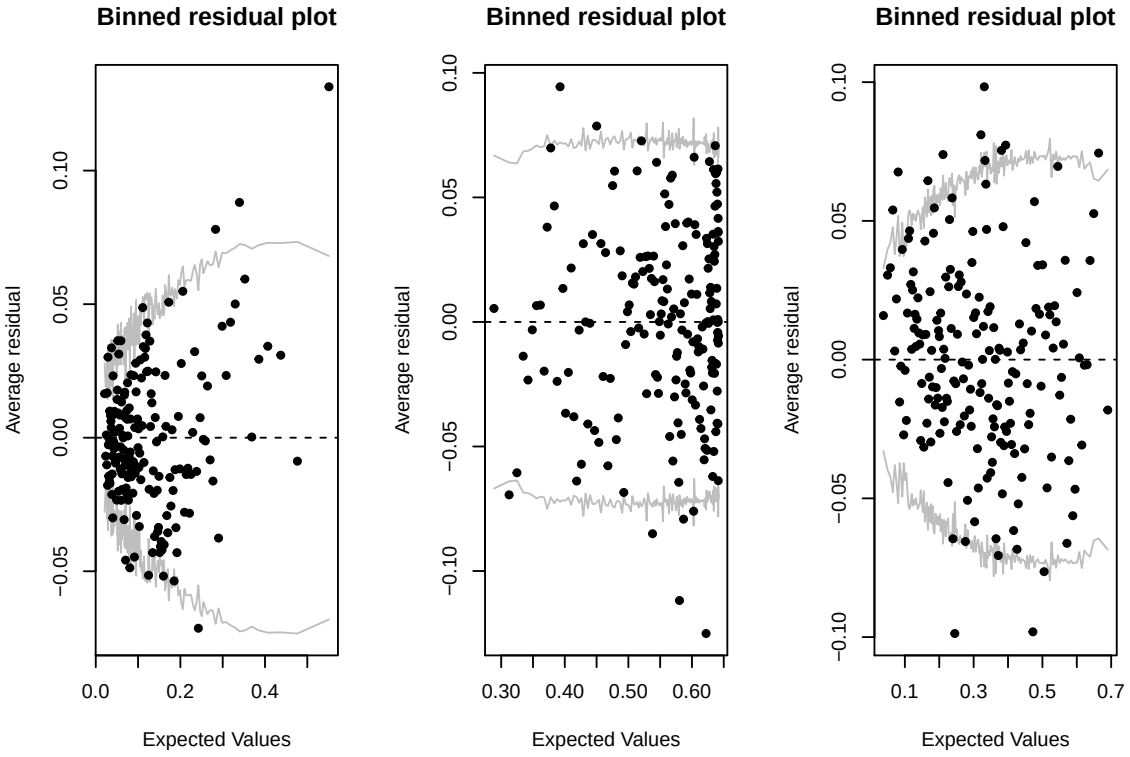
```
##
## Re-fitting to get Hessian
```

```
xx<-model.matrix(~-1+age+finrela+degree+health+marital+sex,data=happy)
xb<-xx[,colnames(simfit@coef)]%*%t(simfit@coef)

reslist<-vector("list",100)
resmat<-matrix(NA,nrow(happy),100)
for(iter in 1:100){
  pa<-invlogit(outer(-xb[,iter],simfit@zeta[iter,],"+"))
```

```
  pp<-cbind( pa[,1], pa[,2]-pa[,1],1-pa[,2])
  aa=apply(pp,1,function(x)rmultinom(1,1,prob=x))
  resmat[,iter]<-t(aa)%*%c(1,2,3)
  #reslist[[iter]]<-resd
}
xt<-apply(resmat,2,table)
prx<-(t(xt)/colSums(xt))

predhap<-predict(happyfit,type="probs")
par(mfrow=c(1,3))
residhap<-model.matrix(~-1+happy,data=happy)-predhap
for(i in 1:3)
binnedplot(predhap[,i],residhap[,i])
```

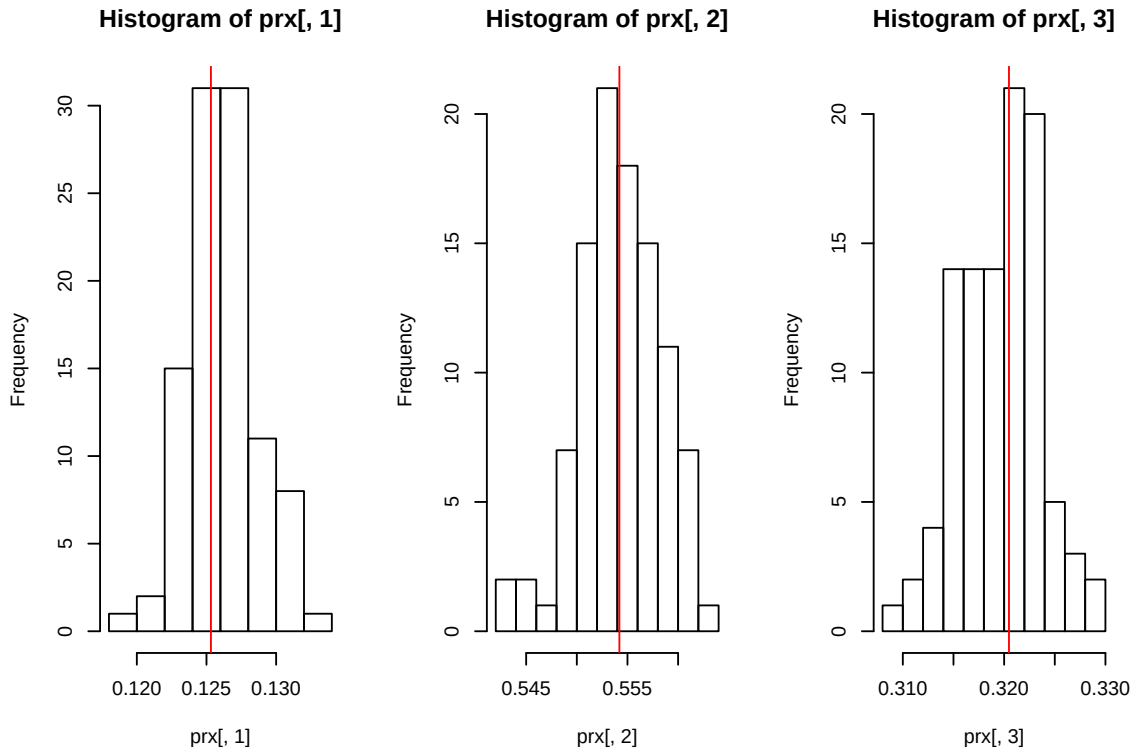


2. Summarize the data by a numerical test statistic, and compare to the values of the test statistic in the replicated datasets.

```
par(mfrow=c(1,3))
hist(prx[,1])
abline(v=mean(happy$happy=="not too happy"),col="red")

hist(prx[,2])
abline(v=mean(happy$happy=="pretty happy"),col="red")

hist(prx[,3])
abline(v=mean(happy$happy=="very happy"),col="red")
```



Using simulation to check the fit of a time-series model:

find time-series data and fit a first-order autoregression model to it. Then use predictive simulation to check the fit of this model as in GH Section 8.4.

I'll use Global mean land (only) temperature deviations measured in degrees centigrade, for the years 1880-2015.

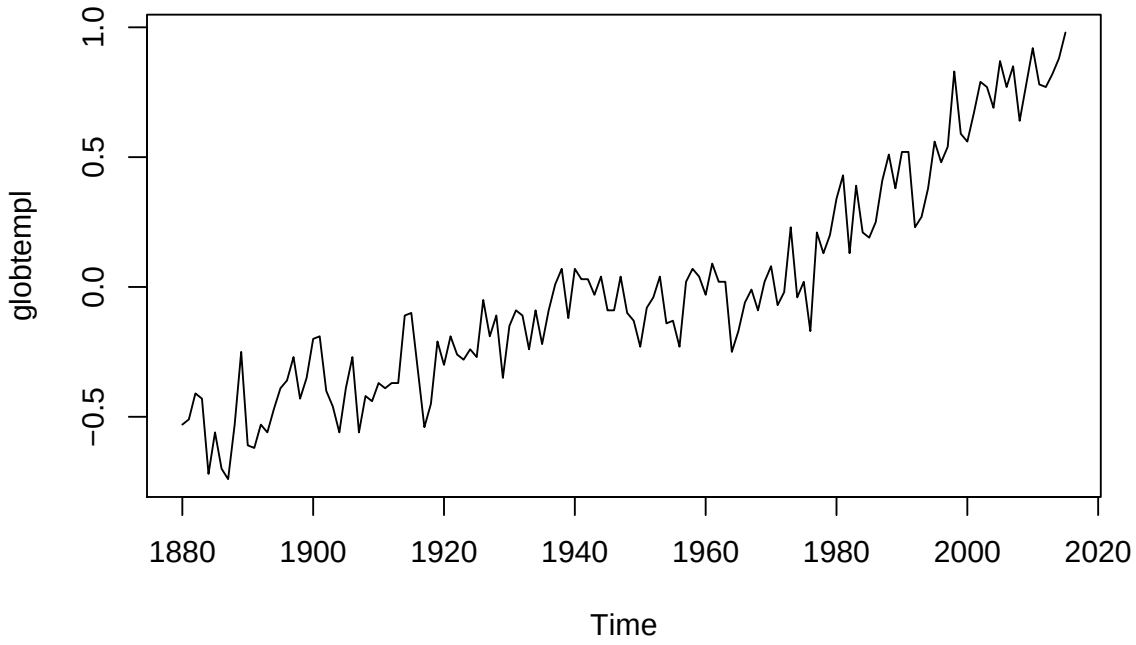
```
library(astsa)
```

```
##
## Attaching package: 'astsa'
```

```
## The following object is masked _by_ '.GlobalEnv':
##
##     jj
```

```
## The following object is masked from 'package:faraway':
##
##     star
```

```
data(globtempl)
?globtempl
plot(globtempl)
```




```
y <- as.vector(globtempl)
#n<- length(yt)
#yt.lag<- c(NA,yt[1:(n-1)])
#lm_lag<-lm(yt~yt.lag)
year<-1880:2015
```

Fitting a 1st-order autogregression

```
n <- length (y)
y.lag <- c (NA, y[1:(n-1)])
lm.lag <- lm (y ~ y.lag)
display (lm.lag)
```

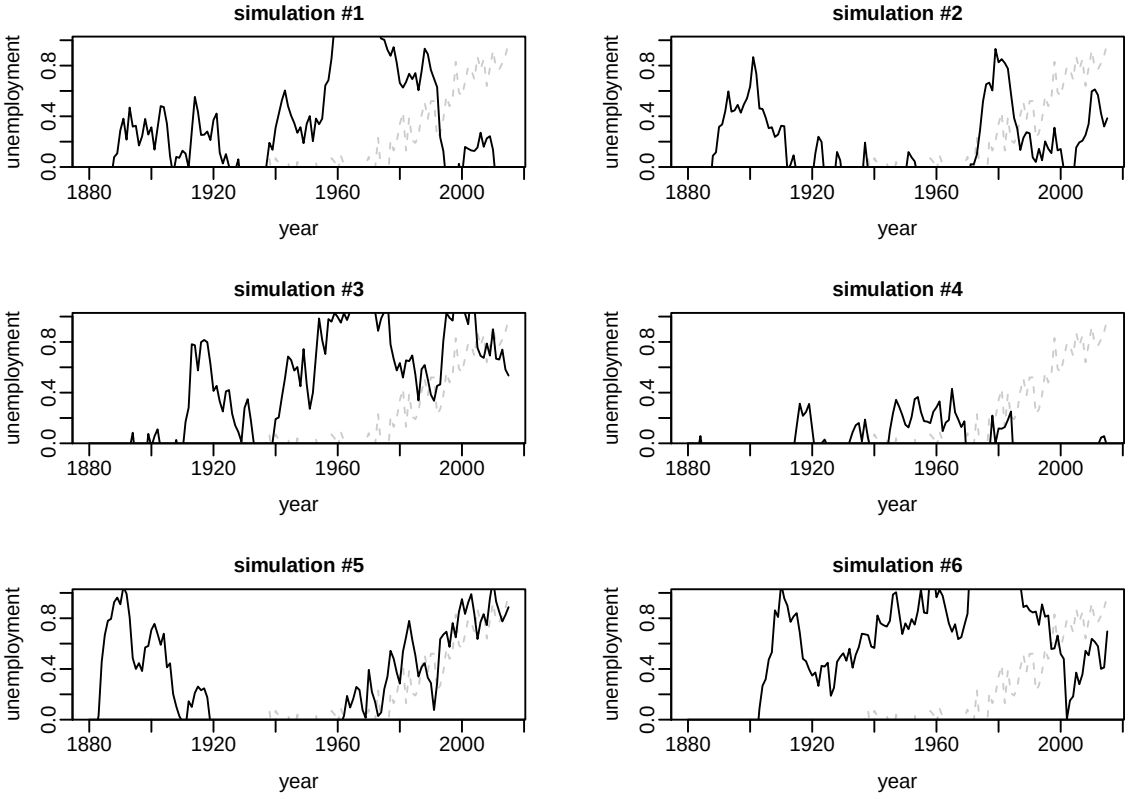
```
## lm(formula = y ~ y.lag)
##             coef.est coef.se
## (Intercept) 0.01     0.01
## y.lag       0.95     0.03
## ---
## n = 135, k = 2
## residual sd = 0.14, R-Squared = 0.88
```

Simulating replicated datasets

```
b.hat <- coef (lm.lag)         # vector of 2 regression coefs
s.hat <- sigma.hat (lm.lag)    # residual sd

n.sims <- 1000
y.rep <- array (NA, c(n.sims, n))
for (s in 1:n.sims){
  y.rep[s,1] <- y[1]
  for (t in 2:n){
    prediction <- c (1, y.rep[s,t-1]) %*% b.hat
    y.rep[s,t] <- rnorm (1, prediction, s.hat)
  }
}
```

```
par (mfrow=c(3,2), mar=c(4,4,2,2))
for (s in 1:6){
  plot (as.integer(year), y.rep[s,], type="l",
        ylab="unemployment", xlab="year", yaxs="i",
         ylim=c(0, max(y)*1.05), yaxt="n", mgp=c(2,.5,0),
        main=paste("simulation #", s, sep=""), cex.main=0.95)
  lines(year,y,col=rgb(0,0,0,alpha=0.2),lty=2)
  axis (2, mgp=c(2,.5,0))
}
```



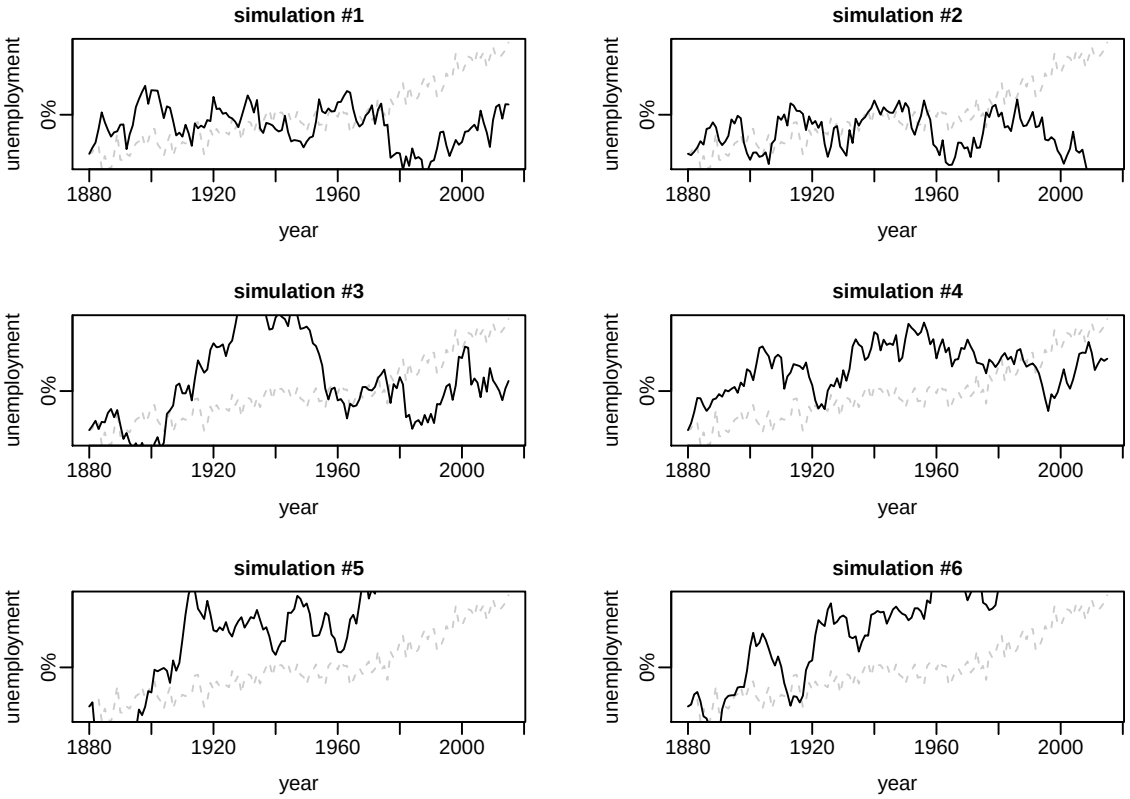
Including uncertainty in the estimated parameters

```
lm.lag.sim <- sim (lm.lag, n.sims)       # simulations of beta and sigma
for (s in 1:n.sims){
  y.rep[s,1] <- y[1]
  for (t in 2:n){
    prediction <-  c (1, y.rep[s,t-1]) %*% lm.lag.sim@coef[s,]
    y.rep[s,t] <- rnorm (1, prediction, lm.lag.sim@sigma[s])
  }
}

## Plot of simulated unemployment rate series

par (mfrow=c(3,2), mar=c(4,4,2,2))
for (s in 1:6){
  plot (as.integer(year), y.rep[s,], type="l", ylab="unemployment", xlab="year", yaxs="i",
  ylim=c(min(y), max(y)*1.05), yaxt="n", mgp=c(2,.5,0),
        main=paste("simulation #", s, sep=""), cex.main=0.95)
```

```
    lines(year,y,col=rgb(0,0,0,alpha=0.2),lty=2)
  axis (2, c(0,5,10), paste (c(0,5,10), "%", sep=""), mgp=c(2,.5,0))
}
```



Model checking for count data:

the folder `risky.behavior` contains data from a study of behavior of couples at risk for HIV;

"sex" is a factor variable with labels "woman" and "man". This is the member of the couple that reporting sex acts to the researcher

The variables "couple" and "women_alone" code the intervention:

couple women_alone 0 0 control - no conselling 1 0 the couple was counselled together 0 1 only the woman was counselled

"bs_hiv" indicates whether the member reporting sex acts was HIV-positive at "baseline", that is, at the beginning of the study.

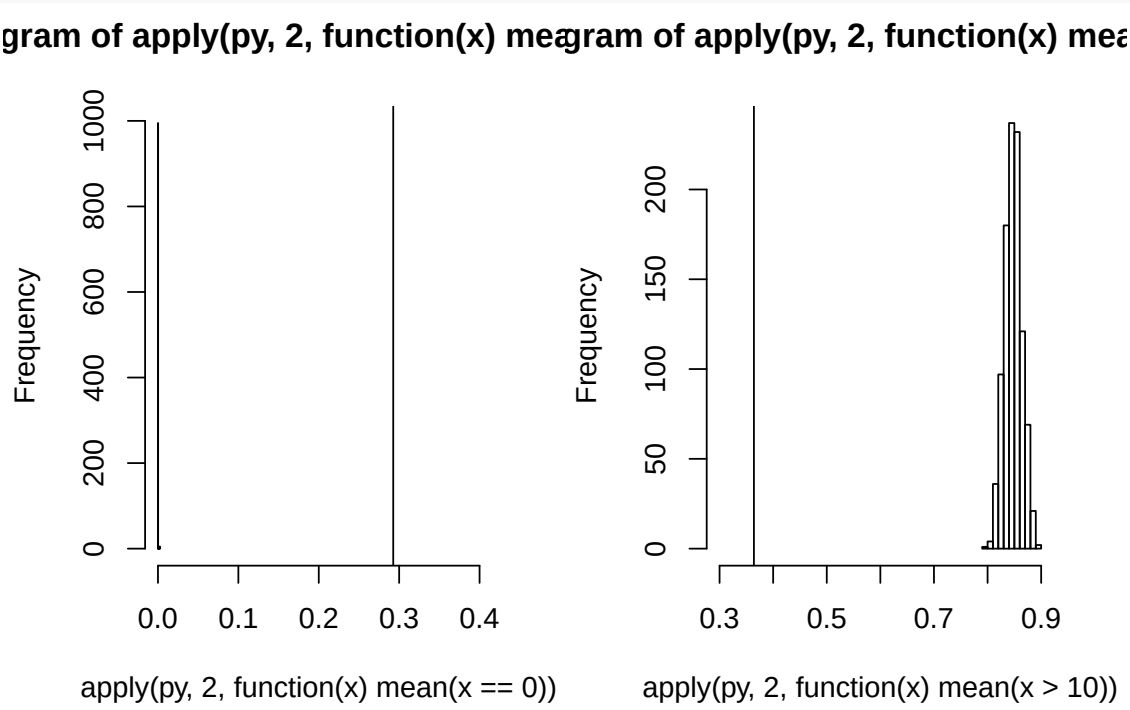
"bupacts" - number of unprotected sex acts reportied at "baseline", that is, at the beginning of the study

"fupacts" - number of unprotected sex acts reported at the end of the study (final report).

1. Fit a Poisson regression model predicting number of unprotected sex acts from baseline HIV status. Perform predictive simulation to generate 1000 datasets and record both the percent of observations that are equal to 0 and the percent that are greater than 10 (the third quartile in the observed data) for each. Compare these values to the observed value in the original data.

```
qpoi.reg.ext <- glm(round(fupacts) ~ bs_hiv,
                    family=poisson, data=risky_behaviors)
```

```
px<-model.matrix(~bs_hiv, data=risky_behaviors)
xb<-px%*%t(sim(qpoi.reg.ext,n.sims=1000)@coef)
py<-apply(xb,2,function(xx) rpois(nrow(risky_behaviors),exp(xx)))
par(mfrow=c(1,2))
hist(apply(py,2,function(x)mean(x==0)),xlim=c(0,0.4));abline(v=mean(risky_behaviors$fupacts==0))
hist(apply(py,2,function(x)mean(x>10)),xlim=c(0.3,0.9));abline(v=mean(risky_behaviors$fupacts>10))
```

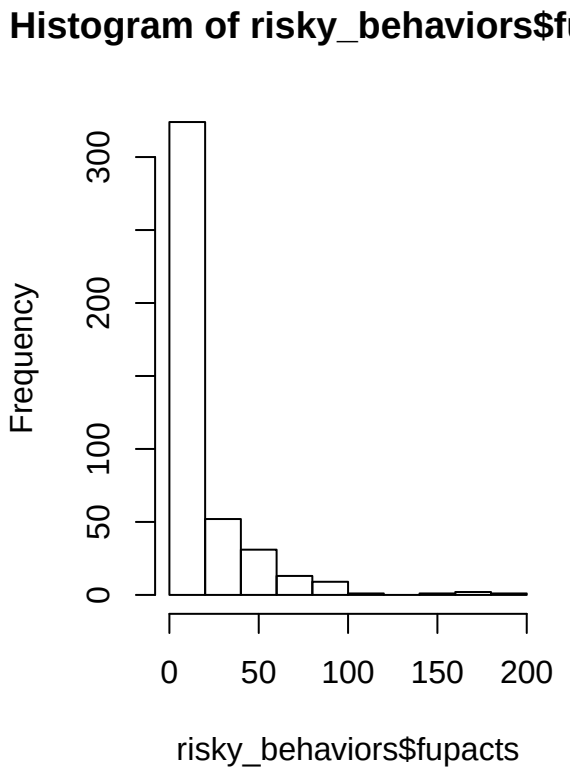



```
hist(risky_behaviors$fupacts)
mean(risky_behaviors$fupacts==0)
```

```
## [1] 0.2926267
```

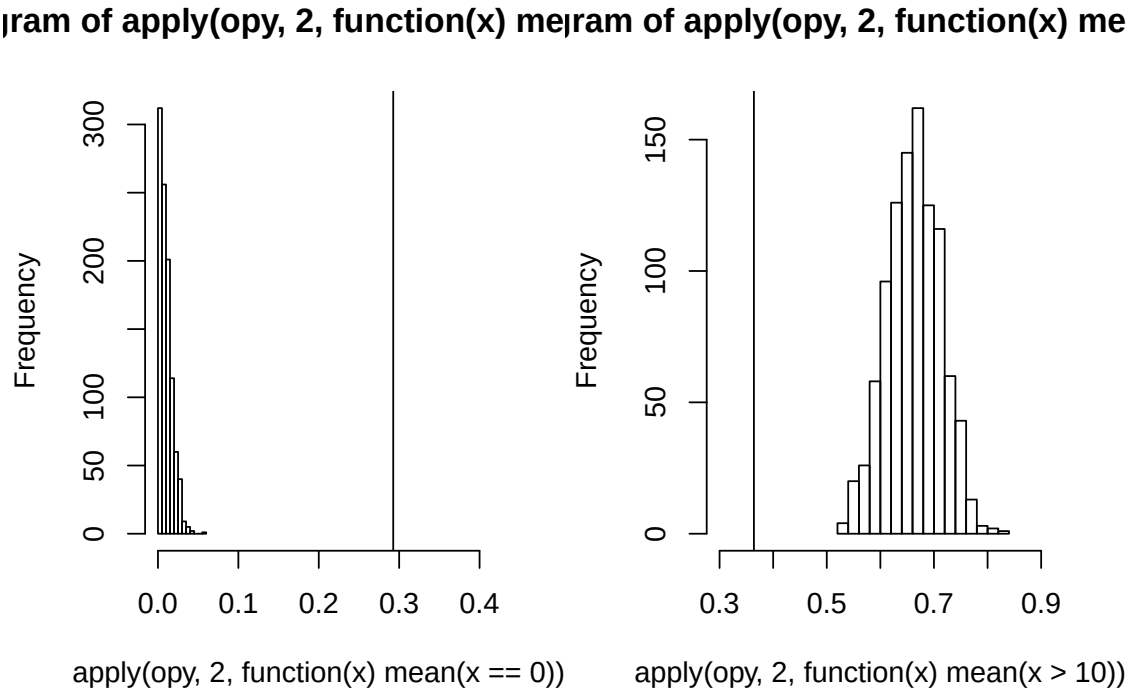
```
mean(risky_behaviors$fupacts>10)
```

```
## [1] 0.3640553
```



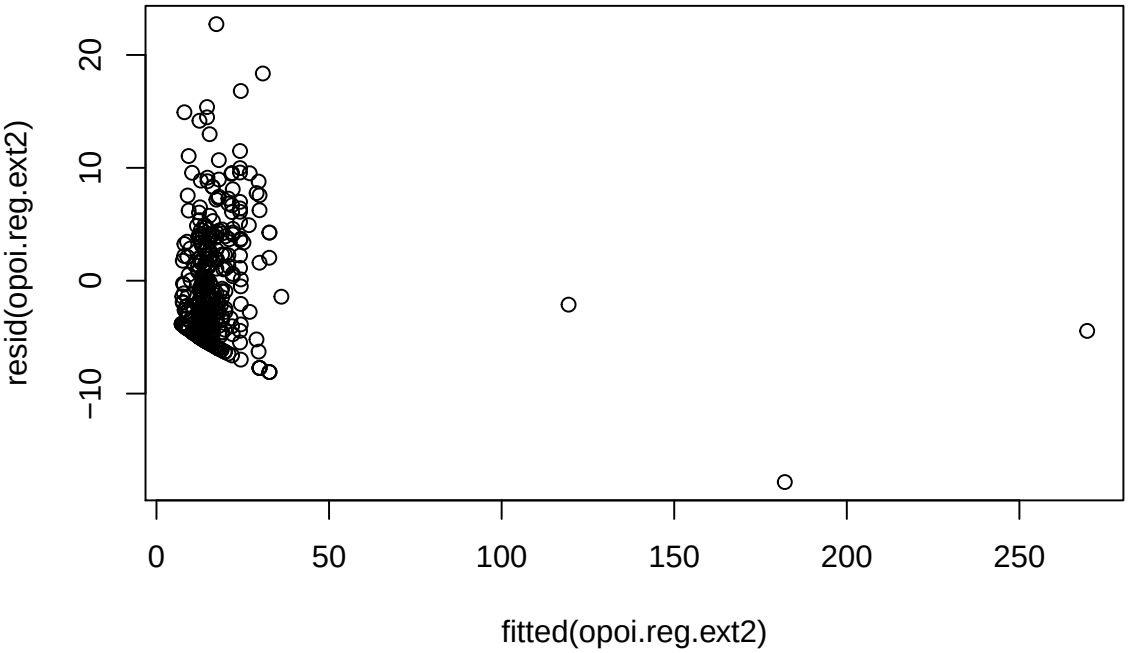
2. Repeat (1) using an overdispersed Poisson regression model.

```
opoi.reg.ext <-glm(round(fupacts) ~bs_hiv ,family=quasipoisson(),data=risky_behaviors)
sop<-sim(opoi.reg.ext,n.sims=1000)
xb<-px%*%t(sop@coef)
rqpois = function(n, lambda, phi) {
  mu = lambda
  k = mu/phi/(1-1/phi)
  return(rnbinom(n, mu = mu, size = k))
}
# https://www.r-bloggers.com/generating-a-quasi-poisson-distribution-version-2/
opy<-apply(xb,2,function(xx) rqpois(nrow(risky_behaviors),exp(xx),sop@sigma[1]))
par(mfrow=c(1,2))
hist(apply(opy,2,function(x)mean(x==0)),xlim=c(0,0.4));abline(v=mean(risky_behaviors$fupacts==0))
hist(apply(opy,2,function(x)mean(x>10)),xlim=c(0.3,0.9));abline(v=mean(risky_behaviors$fupacts>10))
```

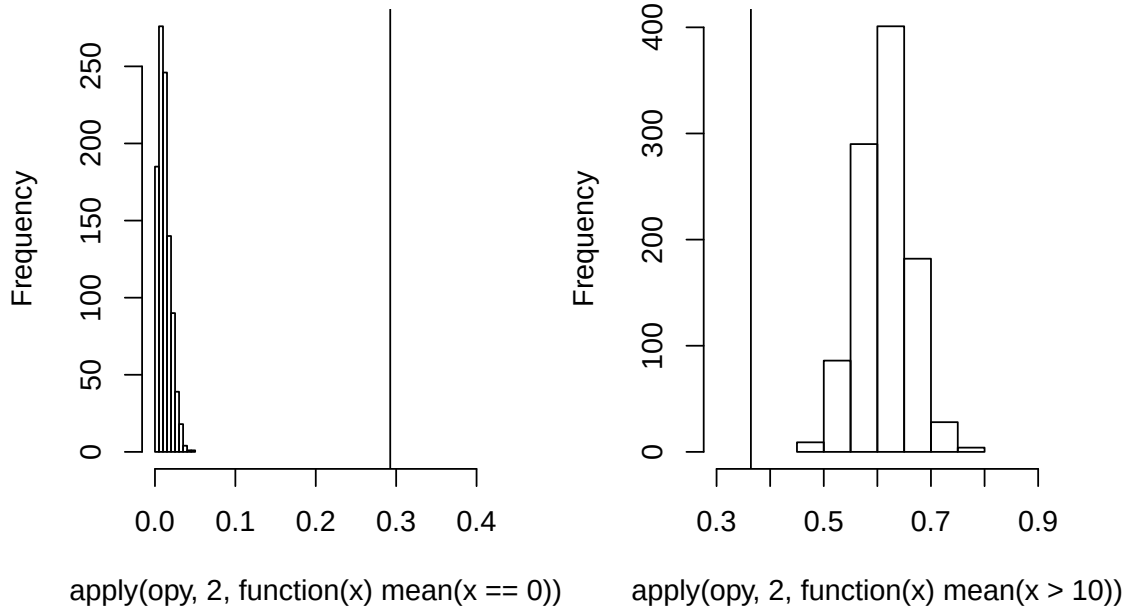


3. Repeat (2), also including gender and baseline number of unprotected sex acts as input variables.

```
opoi.reg.ext2<-glm(round(fupacts) ~ bupacts+bs_hiv+sex ,family=quasipoisson(),data=risky_behaviors)
sop2<-sim(opoi.reg.ext2,n.sims=1000)
px<-model.matrix(~bupacts+bs_hiv+sex, data=risky_behaviors)
xb<-px%*%t(sop2@coef)
plot(fitted(opoi.reg.ext2),resid(opoi.reg.ext2))
```




```
opy<-apply(xb,2,function(xx) rqpois(nrow(risky_behaviors),exp(xx),sop2@sigma[1]))
par(mfrow=c(1,2))
hist(apply(opy,2,function(x)mean(x==0)),xlim=c(0,0.4));abline(v=mean(risky_behaviors$fupacts==0))
hist(apply(opy,2,function(x)mean(x>10)),xlim=c(0.3,0.9));abline(v=mean(risky_behaviors$fupacts>10))
```


ıram of apply(opy, 2, function(x) me
ıram of apply(opy, 2, function(x) me
Frequency
0
50
100
150
200
250
0.0
0.1
0.2
0.3
0.4
apply(opy, 2, function(x) mean(x == 0))
Frequency
0
100
200
300
400
0.3
0.5
0.7
0.9
apply(opy, 2, function(x) mean(x > 10))