PSQ9 Data Simmulation

Lee Panter

Description

This is an exploration of the problem/problem posed by Alan Malik. In the following, the goal of establishing several data sets over which Alan's Neural Net analysis can be replicated will be approached by:

1. simmulating a random sample of IID depression risk scores, to which we will associate a subject
2. simmulating responses to each question of the PSQ9 based upon the depression risk score assigned
3. calculating the "algorithmic sum" (the sum of all question responses) for each subject, and assigning depression classification as outlined by partition in presentation
4. assigning random classification outcome pertubations in proportion to proximity to classification boundaries (assume pertubed classification to represent "expert-evaluated truth")

The final data set(s) will be analyzed using a Convolutional Neural Net classification in the next script. **Scritp-Name.Rmd**

Part (1)

We will simmulate three different depression risk scores distributions for a subject population of $N = 1,000$ surveys.

Suppose that the parameter θ_n represents the n^{th} subjec's depression score rating for $n = 1, \ldots, 1000$. We will be assuming that θ_n is distributed in one of three different ways, i.e.:

$$\theta_{n1} \sim U[0,1]$$

or

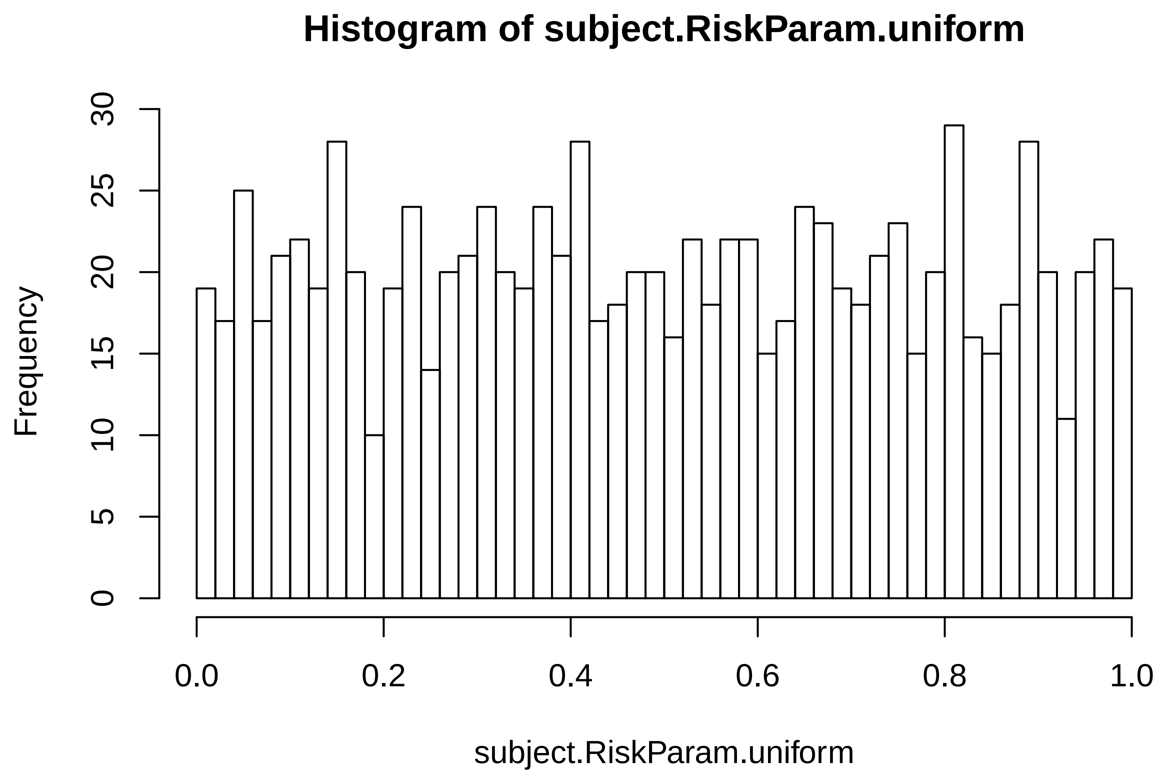
$$\theta_{n2} \sim beta\,(2,2)$$

or

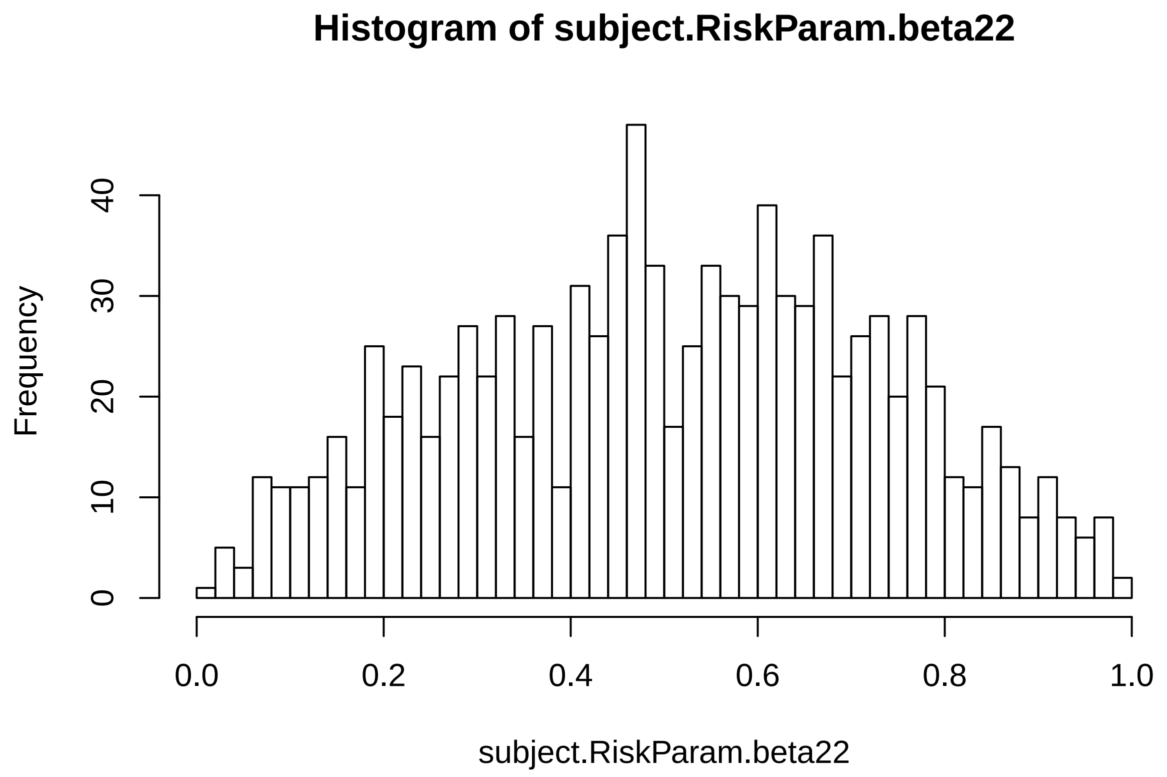
$$\theta_{n3} \sim beta(2,5)$$

each of the above parameters will be referrenced as θ_n, unless the context needs to be specified, and it is unclear.

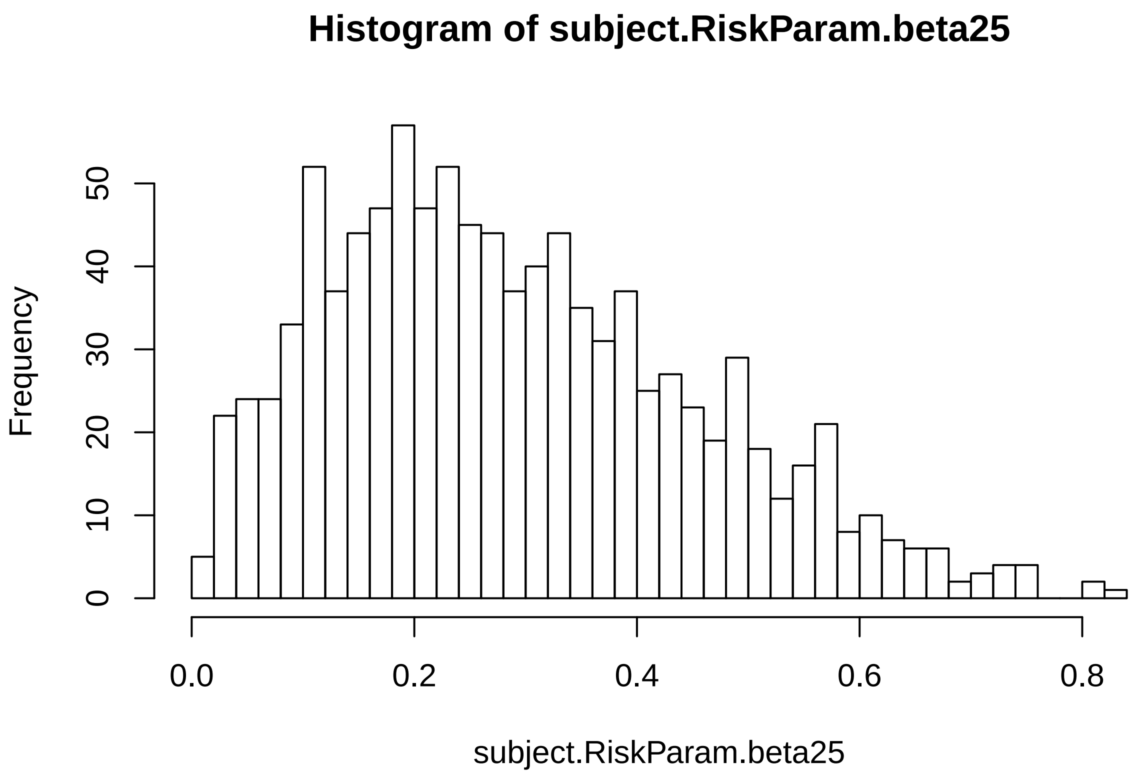
```r
set.seed(123)

N=1000

subject.RiskParam.uniform=runif(N)
subject.RiskParam.beta22=rbeta(N, 2,2)
subject.RiskParam.beta25=rbeta(N, 2,5)

hist(subject.RiskParam.uniform, breaks = 50)
```



```r
hist(subject.RiskParam.beta22, breaks = 50)
```



```r
hist(subject.RiskParam.beta25, breaks = 50)
```



Part (2)

In order to simmulate responses to each question of the PSQ9 which will depend on the user's depression score rating (DSR), we will create a function which will randomly sample responses from a categorical distribution, where each categorical outcome has an associated probability determined by the DSR. We will create this function so that higher values of DSR tend to have higher weighted probabilities for higher valued categorical responses.

Suppose that the value of Y_{ni} is the response given by subject $n \in \{1, \ldots, 1000\}$, on question $i \in \{1, \ldots, 9\}$, then:

$$P\left(Y_{ni} = j\right) = \begin{cases} f_{ni0}\left(\theta_n\right) & \text{for} \quad j = 0 \\ f_{ni1}\left(\theta_n\right) & \text{for} \quad j = 1 \\ f_{ni2}\left(\theta_n\right) & \text{for} \quad j = 2 \\ f_{ni3}\left(\theta_n\right) & \text{for} \quad j = 3 \end{cases}$$

We will partition the values of f_{nij} into three distinct definitions, according to the magnitude of the DSR variable.

Cases:

1. $0 \leq \theta_n < 0.25$
2. $0.25 \leq \theta_n \leq 0.75$
3. $0.75 < \theta_n \leq 1$

Case 1

We will define the question response distribution (function) as:

$$\begin{bmatrix} f_{ni0}\left(\theta_n\right) \\ f_{ni1}\left(\theta_n\right) \\ f_{ni2}\left(\theta_n\right) \\ f_{ni3}\left(\theta_n\right) \end{bmatrix} = \begin{bmatrix} \frac{1-\theta_n}{2} \\ \frac{1-\theta_n}{3} \\ \frac{3}{24}\left(1+5\theta_n\right) \\ \frac{1}{24}\left(1+5\theta_n\right) \end{bmatrix}$$

We note that:

$$\begin{aligned} \sum_{j=0}^{3} f_{nij}\left(\theta_n\right) &= \frac{1-\theta_n}{2} + \frac{1-\theta_n}{3} + \frac{3}{24}\left(1+5\theta_n\right) + \frac{1}{24}\left(1+5\theta_n\right) \\ &= \left(1-\theta_n\right)\left[\frac{1}{2}+\frac{1}{3}\right] + \left(1+5\theta_n\right)\left[\frac{3}{24}+\frac{1}{24}\right] \\ &= \left(1-\theta_n\right)\left[\frac{5}{6}\right] + \left(1+5\theta_n\right)\left[\frac{4}{24}\right] \\ &= \left(1-\theta_n\right)\left[\frac{5}{6}\right] + \left(1+5\theta_n\right)\left[\frac{1}{6}\right] \\ &= \left[\frac{1}{6}\right]\left[5-5\theta_n+1+5\theta_n\right] \\ &= \left[\frac{1}{6}\right]\left[6\right] \\ &= 1 \end{aligned}$$

We also note that:

$$\begin{aligned} \theta_n \in [0, 0.25) &\implies \frac{1-\theta_n}{2} \in (0.375, 0.5] \\ &\implies \frac{1-\theta_n}{3} \in (0.25, 0.3333] \\ &\implies \frac{3}{24}\left(1+5\theta_n\right) \in (0.125, 0.281] \\ &\implies \frac{1}{24}\left(1+5\theta_n\right) \in (0.0417, 0.0937] \end{aligned}$$

Which is approximately the behavior we wish to observe.

Case 2

We will define the question response distribution (function) using:

$$\theta_n^* = min(\theta_n + 0.25, 0.45)$$

$$\begin{bmatrix} f_{ni0}(\theta_n) \\ f_{ni1}(\theta_n) \\ f_{ni2}(\theta_n) \\ f_{ni3}(\theta_n) \end{bmatrix} = \begin{bmatrix} \frac{\theta_n^*}{2} \\ \frac{1-\theta_n^*}{2} \\ \frac{1-\theta_n^*}{2} \\ \frac{\theta_n^*}{2} \end{bmatrix}$$

Case 3

The last case is defined comparably to case 1, but with the weighting replected so that higher values of the response are more probable.

$$\begin{bmatrix} f_{ni0}(\theta_n) \\ f_{ni1}(\theta_n) \\ f_{ni2}(\theta_n) \\ f_{ni3}(\theta_n) \end{bmatrix} = \begin{bmatrix} \frac{1}{24}(1+5\theta_n) \\ \frac{3}{24}(1+5\theta_n) \\ \frac{1-\theta_n}{3} \\ \frac{1-\theta_n}{2} \end{bmatrix}$$

We create a function that creates these probabilities, and evaluates them for each sample parameter.

```
subject.response.function=function(subject.theta.value){
  out.vec=c()
  if(0 <= subject.theta.value & subject.theta.value< 0.25){
    out.vec[1]=(1-subject.theta.value)/2
    out.vec[2]=(1-subject.theta.value)/3
    out.vec[3]=(3/24)*(1+5*subject.theta.value)
    out.vec[4]=(1/24)*(1+5*subject.theta.value)
  }
  else if(0.25 <= subject.theta.value & subject.theta.value <= 0.75){
    theta.star=min(subject.theta.value, 0.35)
    out.vec[1]=theta.star/2
    out.vec[2]=(1-theta.star)/2
    out.vec[3]=(1-theta.star)/2
    out.vec[4]=theta.star/2
  }
  else{
    out.vec[1]=(1/24)*(1+5*subject.theta.value)
    out.vec[2]=(3/24)*(1+5*subject.theta.value)
    out.vec[3]=(1-subject.theta.value)/3
    out.vec[4]=(1-subject.theta.value)/2
  }
  return(out.vec)
}


subject.question.response.probabilites.uniform=list()
subject.question.response.probabilites.beta22=list()
subject.question.response.probabilites.beta25=list()

for(i in 1:N)
{
  subject.question.response.probabilites.uniform[[i]]=subject.response.function(subject.RiskParam.uni
  subject.question.response.probabilites.beta22[[i]]=subject.response.function(subject.RiskParam.beta
  subject.question.response.probabilites.beta25[[i]]=subject.response.function(subject.RiskParam.beta
}
```

We now draw a random sample from the distribution defined at the beginning of part two for each of the subject-specific parameters estimated in the step above. Please note that the following analysis may not be immediately replicable since no seed has been set for the iterative sampling process within subjects over question numbers.

```
response.labels=c("0", "1", "2", "3")
response.labels.ordered=factor(response.labels, ordered = TRUE,
                               levels = c("0", "1", "2", "3"))
response.labels.numerical=c(0,1,2,3)

subject.question.response.uniform = list()
subject.question.response.beta22 = list()
subject.question.response.beta25 = list()

for(i in 1:N)
{
   subject.question.response.uniform[[i]] = extraDistr::rcat(9,
                                                         response.labels.ordered)
   subject.question.response.beta22[[i]] = extraDistr::rcat(9,
                                                           response.labels.ordered)
   subject.question.response.beta25[[i]] = extraDistr::rcat(9,
                                    subject.question.response.probabilites.beta25[[i]],
                                                         response.labels.ordered)
}

for(i in 1:10){
   print(subject.question.response.uniform[[i]])
}
```

```
## [1] 2 2 1 2 2 2 2 2 0
## Levels: 0 1 2 3
## [1] 1 1 0 1 1 0 3 1 0
## Levels: 0 1 2 3
## [1] 0 2 0 1 1 0 1 3 1
## Levels: 0 1 2 3
## [1] 1 1 1 1 0 0 1 1 1
## Levels: 0 1 2 3
## [1] 1 1 0 2 1 1 0 1 0
## Levels: 0 1 2 3
## [1] 0 2 0 1 0 1 0 1 2
## Levels: 0 1 2 3
## [1] 1 3 1 0 2 3 1 1 0
## Levels: 0 1 2 3
## [1] 1 0 3 1 1 1 1 1 0
## Levels: 0 1 2 3
## [1] 1 2 2 1 3 1 2 2 0
## Levels: 0 1 2 3
## [1] 3 0 0 3 2 1 0 3 0
## Levels: 0 1 2 3
```

```
for(i in 1:10){
   print(subject.question.response.beta22[[i]])
}
```

```
## [1] 2 2 1 1 1 2 2 1 1
## Levels: 0 1 2 3
## [1] 0 1 0 1 0 1 1 1 0
## Levels: 0 1 2 3
## [1] 0 0 1 1 1 1 1 1 1
## Levels: 0 1 2 3
## [1] 1 2 1 0 1 3 1 2 1
## Levels: 0 1 2 3
## [1] 2 0 2 1 0 0 0 2 0
## Levels: 0 1 2 3
## [1] 3 1 0 1 3 3 2 3 2
## Levels: 0 1 2 3
## [1] 2 2 2 1 2 1 1 3 2
## Levels: 0 1 2 3
## [1] 1 0 2 2 1 0 2 2 2
## Levels: 0 1 2 3
## [1] 2 0 2 3 1 2 0 1 1
## Levels: 0 1 2 3
## [1] 2 3 3 2 2 3 2 1 2
## Levels: 0 1 2 3
```

```
for(i in 1:10){
   print(subject.question.response.beta25[[i]])
}
```

```
## [1] 0 0 0 1 0 0 2 0 0
## Levels: 0 1 2 3
## [1] 0 0 0 0 1 0 2 1 2
## Levels: 0 1 2 3
## [1] 1 1 1 2 2 0 1 2 2
## Levels: 0 1 2 3
## [1] 0 2 1 2 1 2 3 0 2
## Levels: 0 1 2 3
## [1] 0 2 0 1 0 0 0 1 3
## Levels: 0 1 2 3
## [1] 2 1 2 2 0 2 2 0 3
## Levels: 0 1 2 3
## [1] 3 0 1 1 1 1 0 1 2
## Levels: 0 1 2 3
## [1] 0 3 0 3 2 0 3 2 2
## Levels: 0 1 2 3
## [1] 1 2 1 1 2 0 0 0 0
## Levels: 0 1 2 3
## [1] 2 2 0 1 1 1 2 2 2
## Levels: 0 1 2 3
```

Part (3)

We will now structure our data into a data frame, and give numerical interpretations to the values encoded as factors.

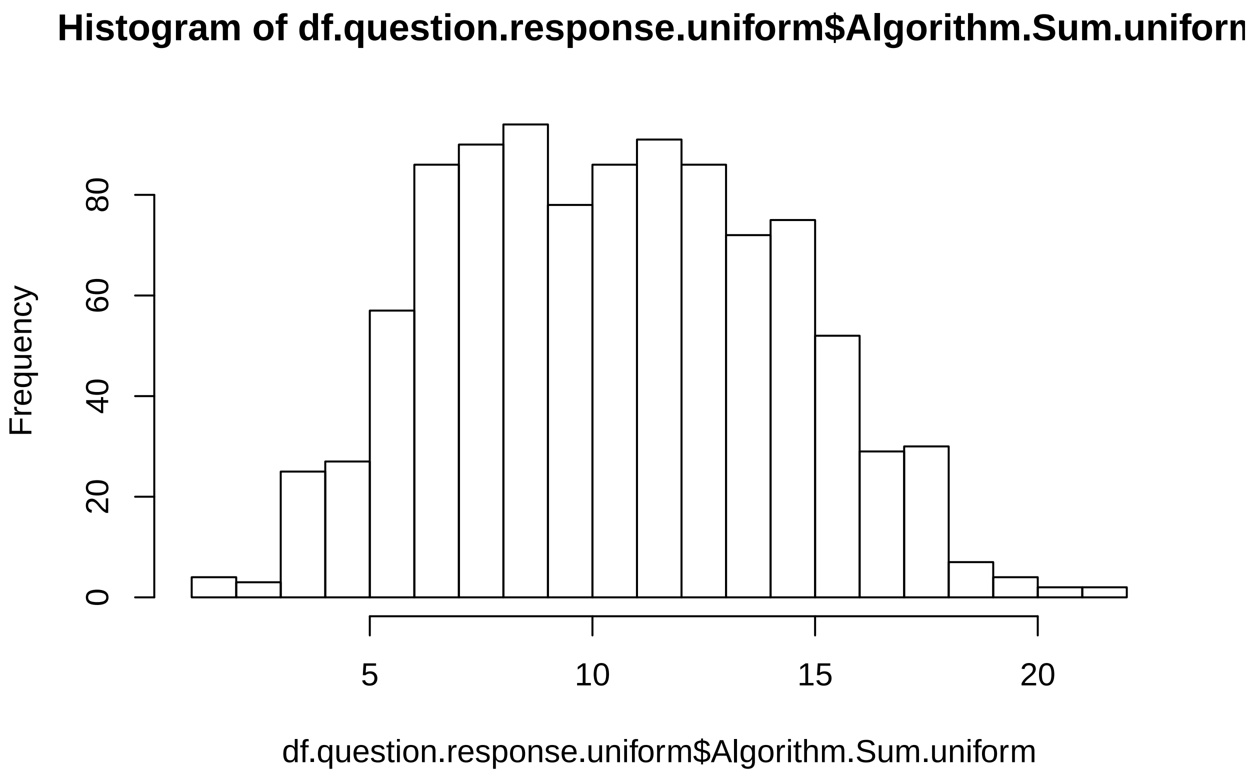
```r
matrix.question.response.uniform = matrix(NA, nrow=1000, ncol=9)
matrix.question.response.beta22 = matrix(NA, nrow=1000, ncol=9)
matrix.question.response.beta25 = matrix(NA, nrow=1000, ncol=9)

for(i in 1:N)
{
  matrix.question.response.uniform[i,]=subject.question.response.uniform[[i]]
  matrix.question.response.beta22[i,]=subject.question.response.beta22[[i]]
  matrix.question.response.beta25[i,]=subject.question.response.beta25[[i]]
}

df.question.response.uniform=as.data.frame(matrix.question.response.uniform)
df.question.response.beta22=as.data.frame(matrix.question.response.beta22)
df.question.response.beta25=as.data.frame(matrix.question.response.beta25)
colnames(df.question.response.uniform)=c("Q1","Q2","Q3","Q4","Q5","Q6","Q7","Q8","Q9")
colnames(df.question.response.beta22)=c("Q1","Q2","Q3","Q4","Q5","Q6","Q7","Q8","Q9")
colnames(df.question.response.beta25)=c("Q1","Q2","Q3","Q4","Q5","Q6","Q7","Q8","Q9")


df.question.response.uniform$Q1.numeric=df.question.response.uniform$Q1-1
df.question.response.uniform$Q2.numeric=df.question.response.uniform$Q2-1
df.question.response.uniform$Q3.numeric=df.question.response.uniform$Q3-1
df.question.response.uniform$Q4.numeric=df.question.response.uniform$Q4-1
df.question.response.uniform$Q5.numeric=df.question.response.uniform$Q5-1
df.question.response.uniform$Q6.numeric=df.question.response.uniform$Q6-1
df.question.response.uniform$Q7.numeric=df.question.response.uniform$Q7-1
df.question.response.uniform$Q8.numeric=df.question.response.uniform$Q8-1
df.question.response.uniform$Q9.numeric=df.question.response.uniform$Q9-1


df.question.response.beta22$Q1.numeric=df.question.response.beta22$Q1-1
df.question.response.beta22$Q2.numeric=df.question.response.beta22$Q2-1
df.question.response.beta22$Q3.numeric=df.question.response.beta22$Q3-1
df.question.response.beta22$Q4.numeric=df.question.response.beta22$Q4-1
df.question.response.beta22$Q5.numeric=df.question.response.beta22$Q5-1
df.question.response.beta22$Q6.numeric=df.question.response.beta22$Q6-1
df.question.response.beta22$Q7.numeric=df.question.response.beta22$Q7-1
df.question.response.beta22$Q8.numeric=df.question.response.beta22$Q8-1
df.question.response.beta22$Q9.numeric=df.question.response.beta22$Q9-1

df.question.response.beta25$Q1.numeric=df.question.response.beta25$Q1-1
df.question.response.beta25$Q2.numeric=df.question.response.beta25$Q2-1
df.question.response.beta25$Q3.numeric=df.question.response.beta25$Q3-1
df.question.response.beta25$Q4.numeric=df.question.response.beta25$Q4-1
df.question.response.beta25$Q5.numeric=df.question.response.beta25$Q5-1
df.question.response.beta25$Q6.numeric=df.question.response.beta25$Q6-1
df.question.response.beta25$Q7.numeric=df.question.response.beta25$Q7-1
df.question.response.beta25$Q8.numeric=df.question.response.beta25$Q8-1
df.question.response.beta25$Q9.numeric=df.question.response.beta25$Q9-1
```

Next, we calculate the “algorithmic sum” as defined by the sum of the response values, i.e:

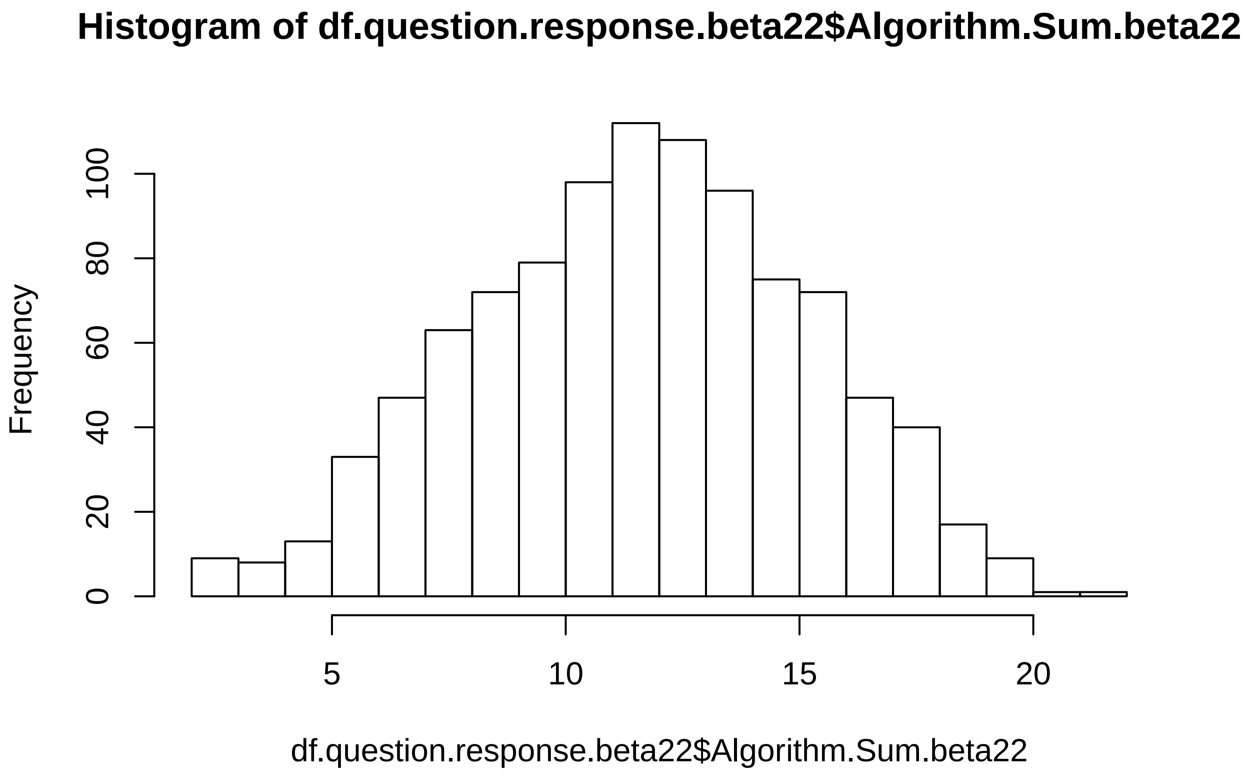
$$Algorithmic\ Sum_n = \sum_{i=1}^{9} Y_{ni}$$

for each $n \in \{1, \ldots, 1000\}$

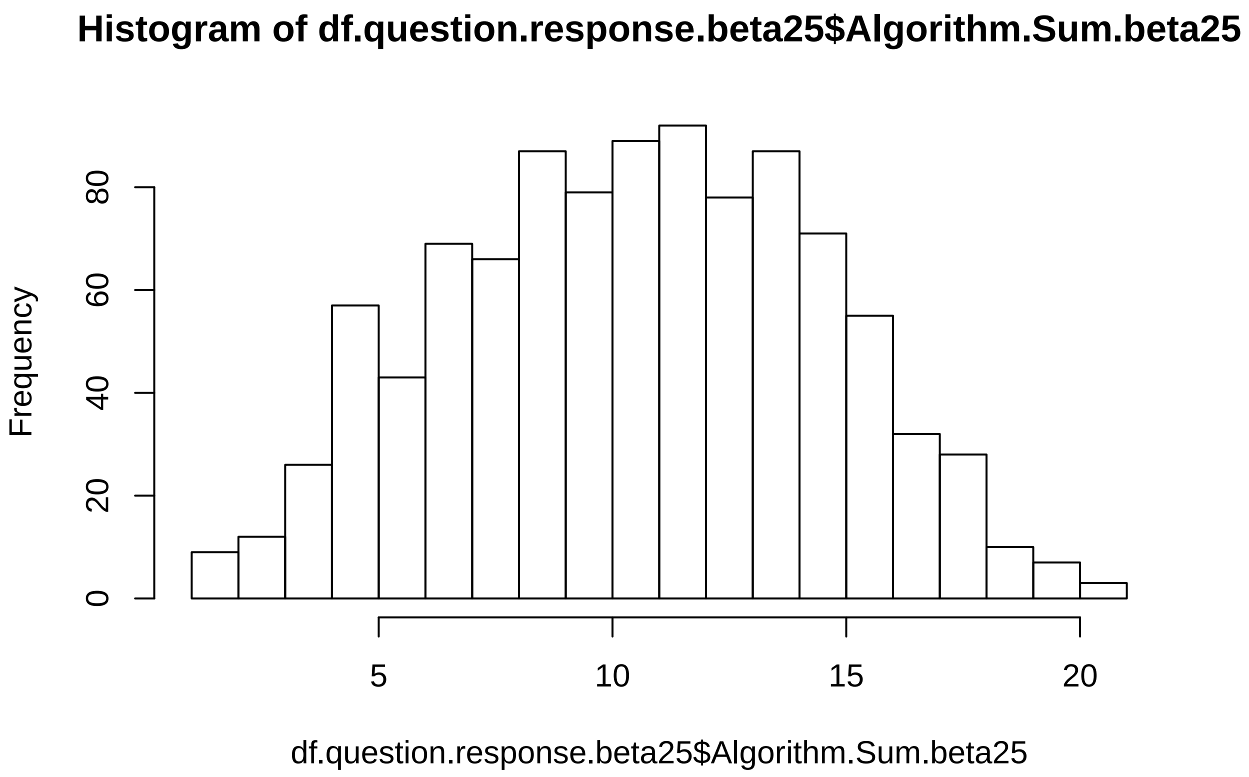
```r
Algorithm.Sum.uniform=c()
Algorithm.Sum.beta22=c()
Algorithm.Sum.beta25=c()

for(i in 1:N)
{
  Algorithm.Sum.uniform[i]=sum(df.question.response.uniform[i,10:18])
  Algorithm.Sum.beta22[i]=sum(df.question.response.beta22[i,10:18])
  Algorithm.Sum.beta25[i]=sum(df.question.response.beta25[i,10:18])
}

df.question.response.uniform$Algorithm.Sum.uniform=Algorithm.Sum.uniform
df.question.response.beta22$Algorithm.Sum.beta22=Algorithm.Sum.beta22
df.question.response.beta25$Algorithm.Sum.beta25=Algorithm.Sum.beta25

hist(df.question.response.uniform$Algorithm.Sum.uniform, breaks = 20)
```



```r
hist(df.question.response.beta22$Algorithm.Sum.beta22, breaks = 20)
```



```r
hist(df.question.response.beta25$Algorithm.Sum.beta25, breaks = 20)
```



We also need to assign an algorithmic depression classification according to the definition:

$$\text{Catigorical Depression Rating}_n = \begin{cases} \text{Not Clinically Depressed (Cat A)} & Alg.Sum_n \in \{0, 1, 2, \ldots, 6\} \\ \text{Sub-Threshold (Cat B)} & Alg.Sum_n \in \{7, 8, 9\} \\ \text{Major Depression (Cat C)} & Alg.Sum_n \in \{10, 11, \ldots, 27\} \end{cases}$$

```r
Algorithm.Category.uniform=c()
Algorithm.Category.beta22=c()
Algorithm.Category.beta25=c()

Algorithm.classification.function=function(in.value){
  out.value=NA_character_
  if(in.value <= 6){out.value <- "A"}
  else if(6 < in.value & in.value < 10){out.value <- "B"}
  else if(10 <= in.value ){out.value <- "C"}
  return(out.value)
}

for(i in 1:N)
{
  Algorithm.Category.uniform[i]=Algorithm.classification.function(df.question.response.uniform$Algori
  Algorithm.Category.beta22[i]=Algorithm.classification.function(df.question.response.beta22$Algorith
  Algorithm.Category.beta25[i]=Algorithm.classification.function(df.question.response.beta25$Algorith
}

df.question.response.uniform$Algorithm.Category.uniform=Algorithm.Category.uniform
df.question.response.beta22$Algorithm.Category.beta22=Algorithm.Category.beta22
df.question.response.beta25$Algorithm.Category.beta25=Algorithm.Category.beta25
```

Part (4)

We will be assigning random classification pertubations according to the following rules: (please note that each of the listed permutations is a further permutaion of the preceeding set)

1. approximately 25% of the observations originally assigned to:

a. cat B because they had an Algorithmic score of 7 will be assigned a cat A status
b. cat A because they had an Algorithmic score of 6 will be assigned a cat B status
c. cat B because they had an Algorithmic score of 9 will be assigned a cat C status
d. cat C because they had an Algorithmic score of 10 will be assigned a cat B status

2. approximately 12.5% of the observations originally assigned to:

a. cat B because they had an Algorithmic score of 8 will be assigned a cat A status
b. cat A because they had an Algorithmic score of 5 will be assigned a cat B status
c. cat B because they had an Algorithmic score of 8 will be assigned a cat C status
d. cat C because they had an Algorithmic score of 11 will be assigned a cat B status

Permutation (1a)

```r
df.question.response.uniform$Algorithm.Category.Original.uniform=df.question.response.uniform$Algorit
df.question.response.beta22$Algorithm.Category.Original.beta22=df.question.response.beta22$Algorithm.
df.question.response.beta25$Algorithm.Category.Original.beta25=df.question.response.beta25$Algorithm.

index.B7.uniform=which(df.question.response.uniform$Algorithm.Category.uniform == "B" & df.question.r
index.B7.beta22=which(df.question.response.beta22$Algorithm.Category.beta22 == "B" & df.question.resp
index.B7.beta25=which(df.question.response.beta22$Algorithm.Category.beta22 == "B" & df.question.resp
length.sample.index.B7.uniform=round(length(index.B7.uniform)/4, digits = 0)
length.sample.index.B7.beta22=round(length(index.B7.beta22)/4, digits = 0)
length.sample.index.B7.beta25=round(length(index.B7.beta25)/4, digits = 0)

index.sample.B7.uniform=sample(index.B7.uniform,
                               length.sample.index.B7.uniform,
                               replace = FALSE)
index.sample.B7.beta22=sample(index.B7.beta22,
                               length.sample.index.B7.beta22,
                               replace = FALSE)
index.sample.B7.beta25=sample(index.B7.beta25,
                               length.sample.index.B7.beta25,
                               replace = FALSE)

df.question.response.uniform[index.sample.B7.uniform,20]="A"
df.question.response.beta22[index.sample.B7.beta22,20]="A"
df.question.response.beta25[index.sample.B7.beta25,20]="A"
```

Permutation (1b)

```
index.A6.uniform=which(df.question.response.uniform$Algorithm.Category.uniform == "A" & df.question.r
index.A6.beta22=which(df.question.response.beta22$Algorithm.Category.beta22 == "A" & df.question.resp
index.A6.beta25=which(df.question.response.beta22$Algorithm.Category.beta22 == "A" & df.question.resp

length.sample.index.A6.uniform=round(length(index.A6.uniform)/4, digits = 0)
length.sample.index.A6.beta22=round(length(index.A6.beta22)/4, digits = 0)
length.sample.index.A6.beta25=round(length(index.A6.beta25)/4, digits = 0)

index.sample.A6.uniform=sample(index.A6.uniform,
                               length.sample.index.A6.uniform,
                               replace = FALSE)
index.sample.A6.beta22=sample(index.A6.beta22,
                               length.sample.index.A6.beta22,
                               replace = FALSE)
index.sample.A6.beta25=sample(index.A6.beta25,
                               length.sample.index.A6.beta25,
                               replace = FALSE)

df.question.response.uniform[index.sample.A6.uniform,20]="B"
df.question.response.beta22[index.sample.A6.beta22,20]="B"
df.question.response.beta25[index.sample.A6.beta25,20]="B"
```

Permutation (1c)

```
index.B9.uniform=which(df.question.response.uniform$Algorithm.Category.uniform == "B" & df.question.r
index.B9.beta22=which(df.question.response.beta22$Algorithm.Category.beta22 == "B" & df.question.resp
index.B9.beta25=which(df.question.response.beta22$Algorithm.Category.beta22 == "B" & df.question.resp

length.sample.index.B9.uniform=round(length(index.B9.uniform)/4, digits = 0)
length.sample.index.B9.beta22=round(length(index.B9.beta22)/4, digits = 0)
length.sample.index.B9.beta25=round(length(index.B9.beta25)/4, digits = 0)

index.sample.B9.uniform=sample(index.B9.uniform,
                               length.sample.index.B9.uniform,
                               replace = FALSE)
index.sample.B9.beta22=sample(index.B9.beta22,
                               length.sample.index.B9.beta22,
                               replace = FALSE)
index.sample.B9.beta25=sample(index.B9.beta25,
                               length.sample.index.B9.beta25,
                               replace = FALSE)

df.question.response.uniform[index.sample.B9.uniform,20]="C"
df.question.response.beta22[index.sample.B9.beta22,20]="C"
df.question.response.beta25[index.sample.B9.beta25,20]="C"
```

Permutation (1d)

```
index.C10.uniform=which(df.question.response.uniform$Algorithm.Category.uniform == "C" & df.question.
index.C10.beta22=which(df.question.response.beta22$Algorithm.Category.beta22 == "C" & df.question.res
index.C10.beta25=which(df.question.response.beta22$Algorithm.Category.beta22 == "C" & df.question.res

length.sample.index.C10.uniform=round(length(index.C10.uniform)/4, digits = 0)
length.sample.index.C10.beta22=round(length(index.C10.beta22)/4, digits = 0)
length.sample.index.C10.beta25=round(length(index.C10.beta25)/4, digits = 0)

index.sample.C10.uniform=sample(index.C10.uniform,
                               length.sample.index.C10.uniform,
                               replace = FALSE)
index.sample.C10.beta22=sample(index.C10.beta22,
                               length.sample.index.C10.beta22,
                               replace = FALSE)
index.sample.C10.beta25=sample(index.C10.beta25,
                               length.sample.index.C10.beta25,
                               replace = FALSE)

df.question.response.uniform[index.sample.C10.uniform,20]="B"
df.question.response.beta22[index.sample.C10.beta22,20]="B"
df.question.response.beta25[index.sample.C10.beta25,20]="B"
```

Permutation (2a)

```
index.B8.uniform=which(df.question.response.uniform$Algorithm.Category.uniform == "B" & df.question.r
index.B8.beta22=which(df.question.response.beta22$Algorithm.Category.beta22 == "B" & df.question.resp
index.B8.beta25=which(df.question.response.beta22$Algorithm.Category.beta22 == "B" & df.question.resp

length.sample.index.B8.uniform=round(length(index.B8.uniform)/8, digits = 0)
length.sample.index.B8.beta22=round(length(index.B8.beta22)/8, digits = 0)
length.sample.index.B8.beta25=round(length(index.B8.beta25)/8, digits = 0)
```

```
index.sample.B8.uniform=sample(index.B8.uniform,
                               length.sample.index.B8.uniform,
                               replace = FALSE)
index.sample.B8.beta22=sample(index.B8.beta22,
                               length.sample.index.B8.beta22,
                               replace = FALSE)
index.sample.B8.beta25=sample(index.B8.beta25,
                               length.sample.index.B8.beta25,
                               replace = FALSE)

df.question.response.uniform[index.sample.B8.uniform,20]="A"
df.question.response.beta22[index.sample.B8.beta22,20]="A"
df.question.response.beta25[index.sample.B8.beta25,20]="A"
```

Permutation (2b)

```
index.A5.uniform=which(df.question.response.uniform$Algorithm.Category.uniform == "A" & df.question.r
index.A5.beta22=which(df.question.response.beta22$Algorithm.Category.beta22 == "A" & df.question.resp
index.A5.beta25=which(df.question.response.beta22$Algorithm.Category.beta22 == "A" & df.question.resp

length.sample.index.A5.uniform=round(length(index.A5.uniform)/8, digits = 0)
length.sample.index.A5.beta22=round(length(index.A5.beta22)/8, digits = 0)
length.sample.index.A5.beta25=round(length(index.A5.beta25)/8, digits = 0)

index.sample.A5.uniform=sample(index.A5.uniform,
                               length.sample.index.A5.uniform,
                               replace = FALSE)
index.sample.A5.beta22=sample(index.A5.beta22,
                               length.sample.index.A5.beta22,
                               replace = FALSE)
index.sample.A5.beta25=sample(index.A5.beta25,
                               length.sample.index.A5.beta25,
                               replace = FALSE)

df.question.response.uniform[index.sample.A5.uniform,20]="B"
df.question.response.beta22[index.sample.A5.beta22,20]="B"
df.question.response.beta25[index.sample.A5.beta25,20]="B"
```

Permutation (2c)

```
index.B82.uniform=which(df.question.response.uniform$Algorithm.Category.uniform == "B" & df.question.
index.B82.beta22=which(df.question.response.beta22$Algorithm.Category.beta22 == "B" & df.question.res
index.B82.beta25=which(df.question.response.beta22$Algorithm.Category.beta22 == "B" & df.question.res

length.sample.index.B82.uniform=round(length(index.B82.uniform)/8, digits = 0)
length.sample.index.B82.beta22=round(length(index.B82.beta22)/8, digits = 0)
length.sample.index.B82.beta25=round(length(index.B82.beta25)/8, digits = 0)

index.sample.B82.uniform=sample(index.B82.uniform,
                               length.sample.index.B82.uniform,
                                replace = FALSE)
index.sample.B82.beta22=sample(index.B82.beta22,
                                length.sample.index.B82.beta22,
                                replace = FALSE)
index.sample.B82.beta25=sample(index.B82.beta25,
                                length.sample.index.B82.beta25,
                                replace = FALSE)

df.question.response.uniform[index.sample.B82.uniform,20]="C"
df.question.response.beta22[index.sample.B82.beta22,20]="C"
df.question.response.beta25[index.sample.B82.beta25,20]="C"
```

Permutation (2d)

```
index.C11.uniform=which(df.question.response.uniform$Algorithm.Category.uniform == "C" & df.question.
index.C11.beta22=which(df.question.response.beta22$Algorithm.Category.beta22 == "C" & df.question.res
index.C11.beta25=which(df.question.response.beta22$Algorithm.Category.beta22 == "C" & df.question.res

length.sample.index.C11.uniform=round(length(index.C11.uniform)/8, digits = 0)
length.sample.index.C11.beta22=round(length(index.C11.beta22)/8, digits = 0)
length.sample.index.C11.beta25=round(length(index.C11.beta25)/8, digits = 0)

index.sample.C11.uniform=sample(index.C11.uniform,
                                length.sample.index.C11.uniform,
                                replace = FALSE)
index.sample.C11.beta22=sample(index.C11.beta22,
                                length.sample.index.C11.beta22,
                                replace = FALSE)
index.sample.C11.beta25=sample(index.C11.beta25,
                                length.sample.index.C11.beta25,
                                replace = FALSE)

df.question.response.uniform[index.sample.C11.uniform,20]="B"
df.question.response.beta22[index.sample.C11.beta22,20]="B"
df.question.response.beta25[index.sample.C11.beta25,20]="B"
```

Write data sets for use in other analyses

We will now create data frames that we wish to use to conduct further analyses, and saven them for later use.

```
df_uniform=data.frame(df.question.response.uniform$Q1,
                      df.question.response.uniform$Q2,
                      df.question.response.uniform$Q3,
                      df.question.response.uniform$Q4,
                      df.question.response.uniform$Q5,
                      df.question.response.uniform$Q6,
                      df.question.response.uniform$Q7,
                      df.question.response.uniform$Q8,
                      df.question.response.uniform$Q9,
                      df.question.response.uniform$Algorithm.Sum.uniform,
                      df.question.response.uniform$Algorithm.Category.Original.uniform,
                      df.question.response.uniform$Algorithm.Category.uniform)

colnames(df_uniform) = c("Q1",
                         "Q2",
                         "Q3",
                         "Q4",
                         "Q5",
                         "Q6",
                         "Q7",
                         "Q8",
                         "Q9",
                         "Sum",
                         "AlgCat",
                         "ResponseCat")

save(df_uniform, file = "df_uniform.Rdata")
```

```
df_beta22=data.frame(df.question.response.beta22$Q1,
                      df.question.response.beta22$Q2,
                      df.question.response.beta22$Q3,
                      df.question.response.beta22$Q4,
                      df.question.response.beta22$Q5,
                      df.question.response.beta22$Q6,
                      df.question.response.beta22$Q7,
                      df.question.response.beta22$Q8,
                      df.question.response.beta22$Q9,
                      df.question.response.beta22$Algorithm.Sum.beta22,
                      df.question.response.beta22$Algorithm.Category.Original.beta22,
                      df.question.response.beta22$Algorithm.Category.beta22)

colnames(df_beta22) = c("Q1",
                         "Q2",
                         "Q3",
                         "Q4",
                         "Q5",
                         "Q6",
                         "Q7",
                         "Q8",
                         "Q9",
                         "Sum",
                         "AlgCat",
                         "ResponseCat")

save(df_beta22, file = "df_beta22.Rdata")
```

```
df_beta25=data.frame(df.question.response.beta25$Q1,
                      df.question.response.beta25$Q2,
                      df.question.response.beta25$Q3,
                      df.question.response.beta25$Q4,
                      df.question.response.beta25$Q5,
                      df.question.response.beta25$Q6,
                      df.question.response.beta25$Q7,
                      df.question.response.beta25$Q8,
                      df.question.response.beta25$Q9,
                      df.question.response.beta25$Algorithm.Sum.beta25,
                      df.question.response.beta25$Algorithm.Category.Original.beta25,
                      df.question.response.beta25$Algorithm.Category.beta25)

colnames(df_beta25) = c("Q1",
                         "Q2",
                         "Q3",
                         "Q4",
                         "Q5",
                         "Q6",
                         "Q7",
                         "Q8",
                         "Q9",
                         "Sum",
                         "AlgCat",
                         "ResponseCat")

save(df_beta25, file = "df_beta25.Rdata")
```

Notes:

- add subsequent script name in description