CBB 752 - Project 1.3

Extract mutations

```
getMutations <- function(str) {
  (str <- sub(";.*", "", str)); (str <- sub(",.*", "", str))
  (str <- sub("\\d+_\\d+_\\d+_", "", str))
  (from <- sub("->.", "", str));
  (to <- sub(".->", "", str))
  data.frame(from=from, to=to)
}

mutations <- getMutations(x[,"change"])
mutations$syn <- x$syn

table(mutations$from)
```

```
##
##   A   C   D   E   F   G   H   I   K   L   M   N   P   Q   R   S   T   V
## 238  85 127 113  87 174 129 225 132 163 157 146 228 137 412 286 296 340
##   W   Y
##  10  48
```

```
table(mutations$to)
```

```
##
##   *   A   C   D   E   F   G   H   I   K   L   M   N   P   Q   R   S   T
##   3 286  86 148 144  65 175 139 189 102 243 148 139 167 143 315 304 317
##   V   W   Y
## 315  40  65
```

Charge Change

We score each change in the following manner:

```
charge.chage.table = matrix(c(0,1,1,1,0,2,1,2,0), nrow = 3, ncol=3, byrow = TRUE)
row.names(charge.chage.table) <- c("Neutral", "Hydrophilic", "Hydrophobic" )
colnames(charge.chage.table) <- c("Neutral", "Hydrophilic", "Hydrophobic" )
kable(charge.chage.table, caption = "Amino Acid Hydrophobicity Change Table")
```

Table 1: Amino Acid Hydrophobicity Change Table

	Neutral	Hydrophilic	Hydrophobic
Neutral	0	1	1
Hydrophilic	1	0	2
Hydrophobic	1	2	0

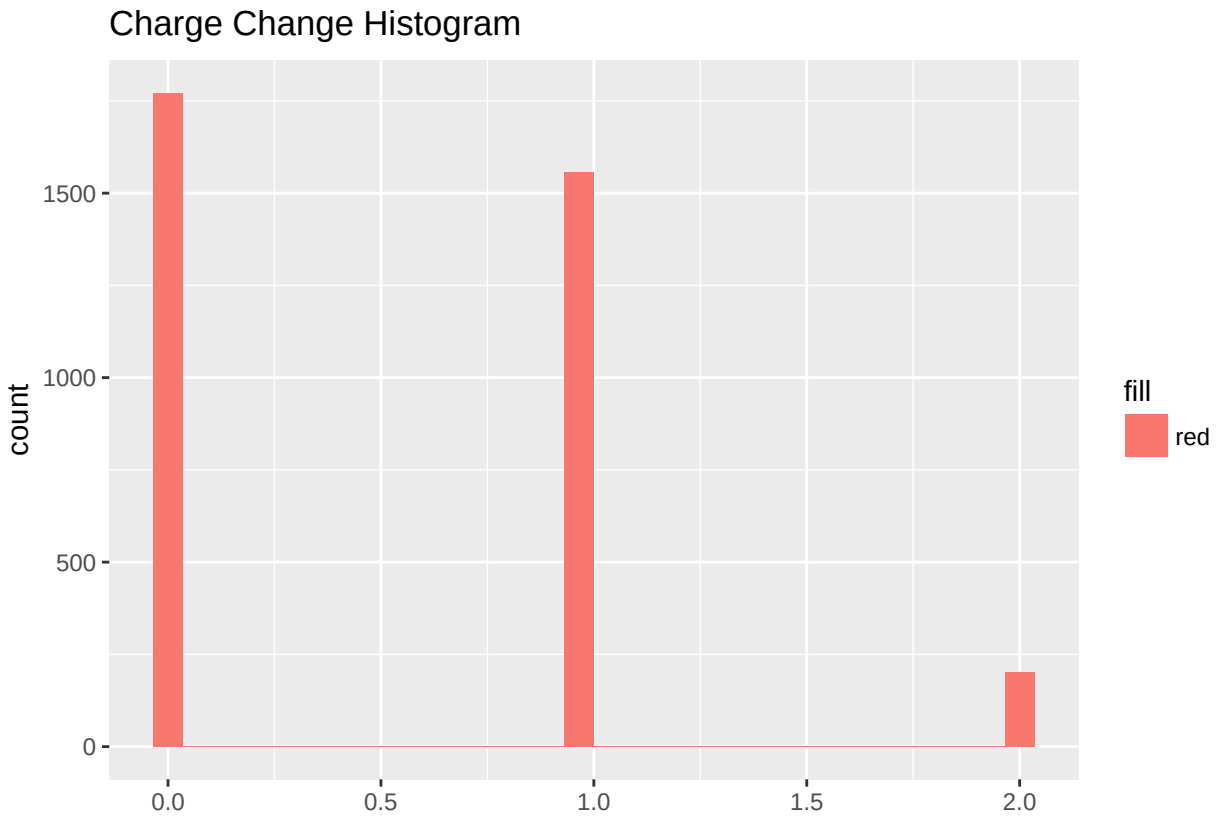
Let's calculate the score according to this scoring system

```
# There are many scales for AA hydrophobiciy. They following was adapted from:
# http://www.imgt.org/IMGTeducation/Aide-memoire/_UK/aminoacids/IMGTclasses.html
aa.hydrophobic <- c("A", "C", "I", "L", "M", "F", "W", "V")
aa.hydrophilic <- c("R", "N", "D", "Q", "E", "K")
aa.neutral     <- c("G", "H", "P", "S", "T", "Y")
charge_change <- function(values) {
  from <- values[1];  to <- values[2]; cc <- NA
  if(from %in% aa.neutral) {
    if(to %in% aa.neutral) { cc <- charge.chage.table["Neutral", "Neutral"] }
    if(to %in% aa.hydrophilic) { cc <- charge.chage.table["Neutral", "Hydrophilic"] }
    if(to %in% aa.hydrophobic) { cc <- charge.chage.table["Neutral", "Hydrophobic"] }
  }
  if(from %in% aa.hydrophilic) {
    if(to %in% aa.hydrophilic) { cc <- charge.chage.table["Hydrophilic", "Hydrophilic"] }
    if(to %in% aa.neutral) { cc <- charge.chage.table["Hydrophilic", "Neutral"] }
    if(to %in% aa.hydrophobic) { cc <- charge.chage.table["Hydrophilic", "Hydrophobic"] }
  }
  if(from %in% aa.hydrophobic) {
    if(to %in% aa.hydrophobic) { cc <- charge.chage.table["Hydrophobic", "Hydrophobic"] }
    if(to %in% aa.neutral) { cc <- charge.chage.table["Hydrophobic", "Neutral"] }
    if(to %in% aa.hydrophilic) { cc <- charge.chage.table["Hydrophobic", "Hydrophilic"] }
  }
  return(cc)
}

mutations$cc <- apply(mutations, 1, charge_change)
```

Plot of AA chagne in charge

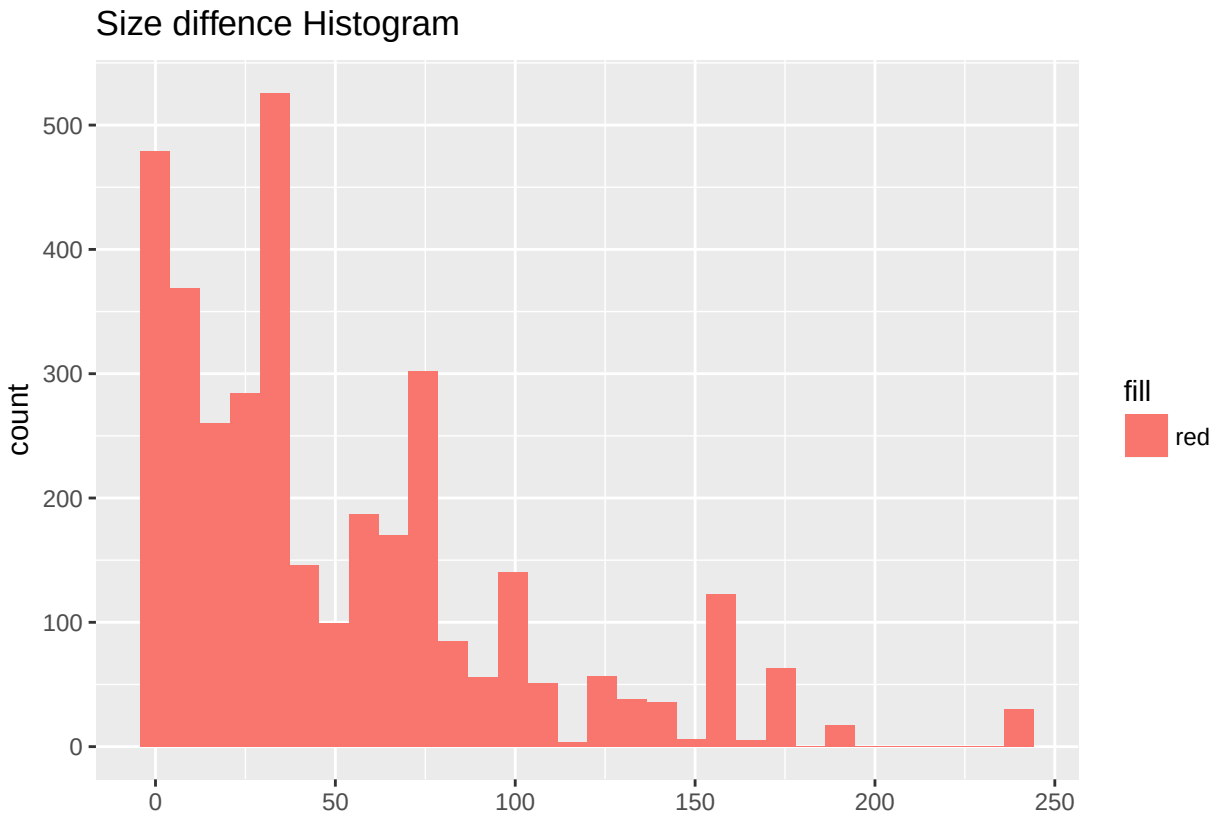
```
qplot(mutations$cc,
      geom="histogram",
      main="Charge Change Histogram", xlab="",
      border="blue", fill="red")
```



Size Change

As as estimate for size, we used the average volume of buried residues, calculated from the surface area of the side chain (Richards 1977; Baumann et al. 1989) Adapted from: http://www.proteinsandproteomics.org/content/free/tables_1/table08.pdf

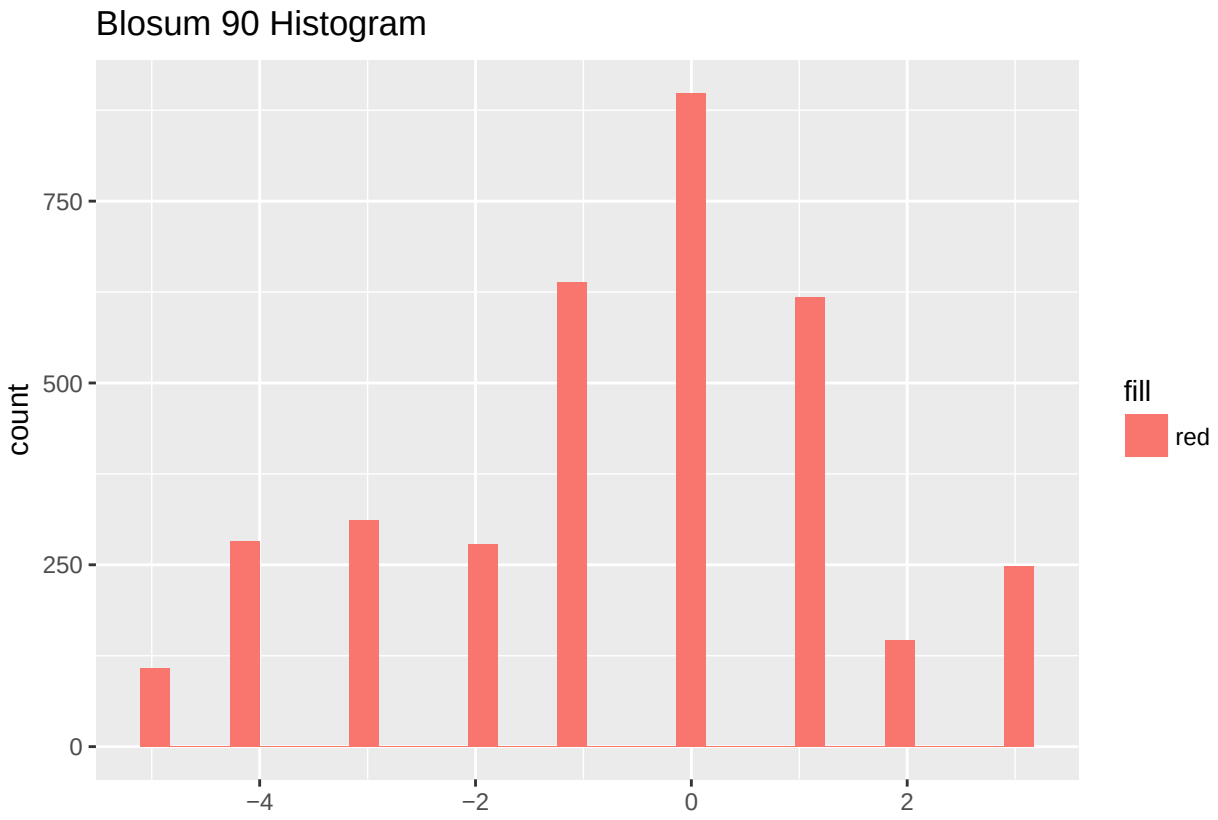
```
aa.sizes <- list("A"=92 , "C"=106, "I"=169, "L"=168, "M"=171, "F"=203,
                 "W"=240, "V"=142, "R"=255, "N"=135, "D"=125, "Q"=161,
                 "E"=155, "K"=171, "G"=66 , "H"=167, "P"=129, "S"=99 ,
                 "T"=122, "Y"=203, "*"=0)
mutations$size <- abs(as.numeric(unlist(aa.sizes[mutations$from])) -
                      as.numeric(unlist(aa.sizes[mutations$to])))
qplot(mutations$size,
      geom="histogram",
      main="Size diffence Histogram", xlab="",
      border="blue", fill="red")
```



Blosum 90

```
data("blosum90")
blosum <- function(values) {
  b <- NA
  from <- values[1];  to <- values[2];
  if( from != "*" & to != "*") { b <- blosum90[from,to] }
  return(b)
}

mutations$blosum <- apply(mutations, 1, blosum)
qplot(mutations$blosum,
      geom="histogram",
      main="Blosum 90 Histogram", xlab="",
      border="blue", fill="red")
```

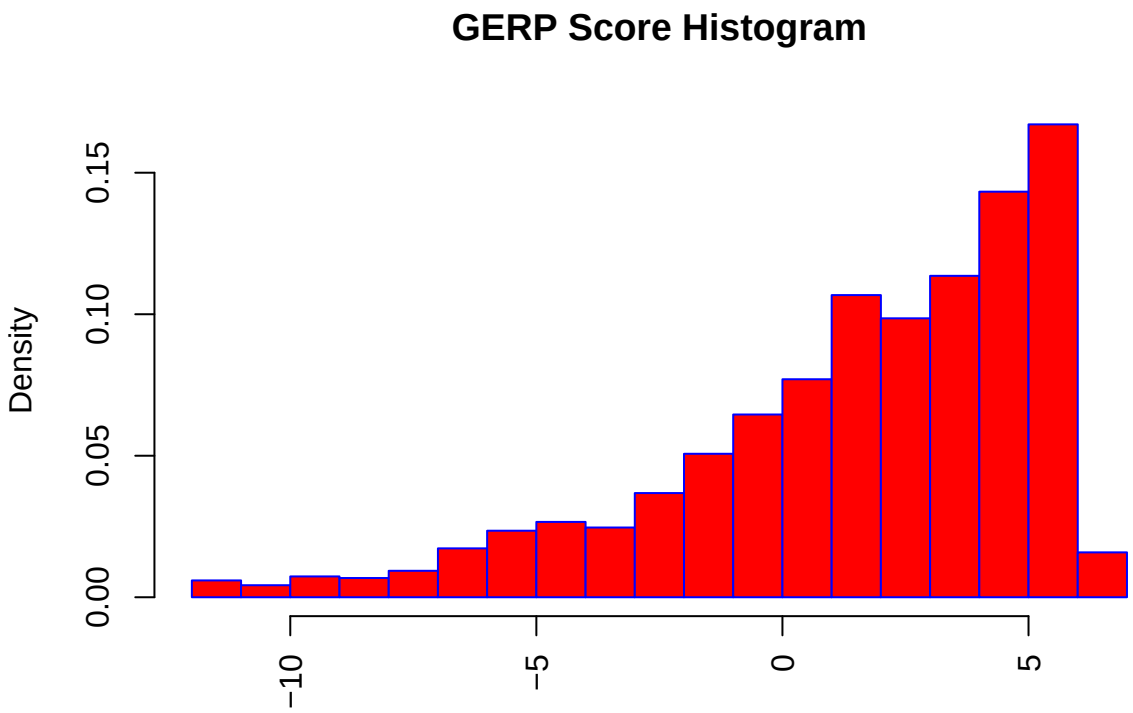


GERP Score

```
gerp <- function(str) {
  tmp <- sub("GERP=", "", str)
  tmp <- sub(";.*", "", tmp)
  return(as.numeric(tmp))
}

mutations$gerp <- gerp(x$GERP)

hist(mutations$gerp, probability = TRUE, las=3,
     main="GERP Score Histogram", xlab="",
     border="blue", col="red")
```



Premature Stop

We have sevral premature stop cases. Let's see how short is the read compared to the reference.

```
pre.l <- which(mutations$syn == "prematureStop")
pre <- x[pre.l, c("mutant", "wildtype")]
pre$missing.len <- as.numeric(nchar(pre$wildtype) - regexpr("\\*", pre$mutant))
```

We can see that we have 3 missing nonsense mutations One with length of 15, and two other with lenght of 101 We can assume taht the longer the deletion is, the more deleterious it is.

Summary Statistics

Let's see what is the frequency of each mutation

```
mutations.l <- array()
for(i in 1:nrow(mutations)) {
  m <- paste(mutations$from[i],mutations$to[i], sep="")
  mutations.l <- c(mutations.l, m)
}

sort(table(mutations.l))
```

```
## mutations.l
```

```
##  FI  LW  MR  CS  DV  EV  HL  HP  IF  LI  PQ  RI  SY  DY  GC  HN  LR  RM
##   1   1   1   2   2   2   2   2   2   2   2   2   2   3   3   3   3   3
##  VD  Y*  CF  NY  QP  RL  SI  YF  LH  LQ  AG  EA  HD  MK  SF  YN  AD  FC
##   3   3   4   4   4   4   4   4   5   5   6   6   6   6   6   6   7   7
##  FY  PH  QK  SC  TK  DA  FV  NH  TR  IN  IS  LM  NK  RT  TN  VG  YH  AE
##   7   7   7   7   7   8   8   8   8   9   9   9   9   9   9   9   9  10
##  DH  WR  KT  ML  NT  QH  VF  YD  CW  IL  QL  EQ  GV  KQ  MI  PR  YC  AP
##  10  10  11  11  11  11  11  11  12  12  14  15  15  15  15  15  15  16
##  EG  LS  IM  AS  RP  CY  GD  KN  QE  HQ  DG  SL  SR  CG  CR  FS  RW  ST
##  16  16  17  18  18  19  19  19  20  21  22  22  22  23  25  26  27  27
##  EK  PA  GE  HY  LV  TP  GS  PT  RG  SA  GA  DN  FL  GR  LF  KE  RS  SP
##  29  29  30  30  31  32  33  35  35  35  36  38  38  38  38  40  42  42
##  DE  RK  ED  IT  KR  TS  PS  VL  LP  RC  SN  ND  NS  TI  MV  TM  VM  SG
##  44  44  45  47  47  49  52  52  53  54  55  57  57  57  58  59  60  64
##  HR  MT  AV  TA  QR  RQ  PL  RH  VA  VI  AT  IV
##  65  66  70  75  81  85  88  89  97 108 111 129
```

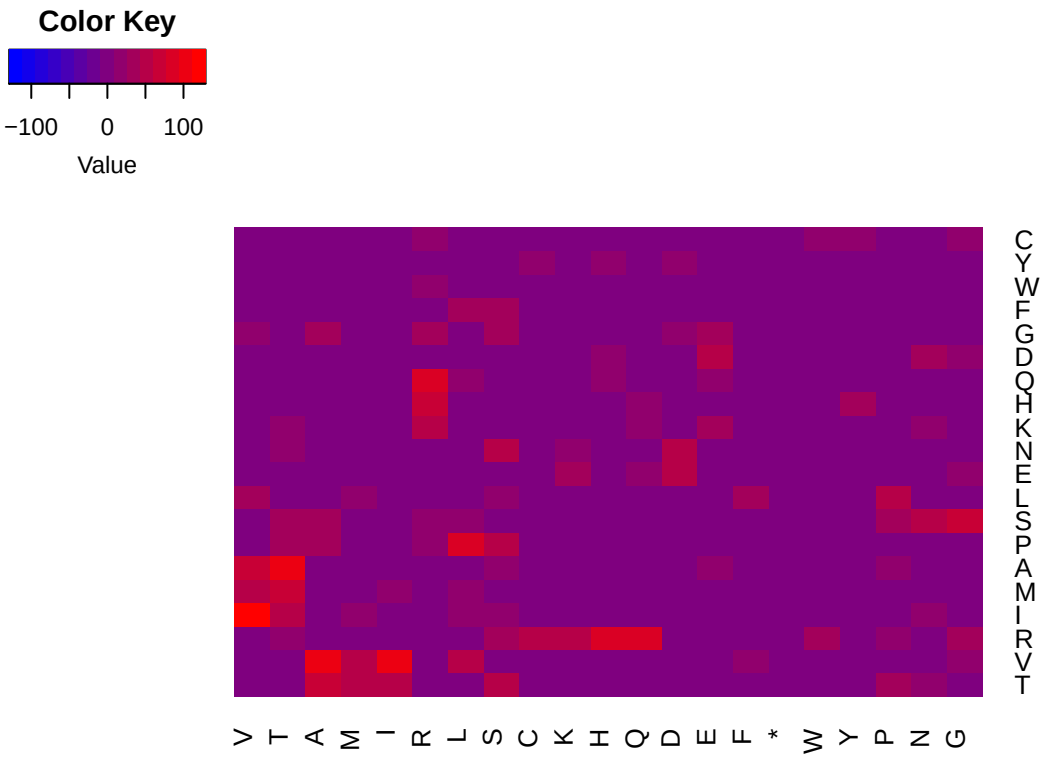
```
mutations.mat <- matrix(0, nrow = length(unique(mutations$from)),
                        ncol = length(unique(mutations$to)))
rownames(mutations.mat) = sort(unique(mutations$from))
colnames(mutations.mat) = sort(unique(mutations$to), decreasing = FALSE)

for(i in 1:nrow(mutations)) {
  mutations.mat[mutations$from[i], mutations$to[i]] <-
    1 + mutations.mat[mutations$from[i], mutations$to[i]]
}

mutations.mat
```

```
##    *  A  C  D  E  F  G  H   I  K  L  M  N  P  Q  R  S   T   V  W  Y
## A 0  0  0  7 10  0  6  0   0  0  0  0  0 16  0  0 18 111  70  0  0
## C 0  0  0  0  0  4 23  0   0  0  0  0  0  0  0 25  2   0   0 12 19
## D 0  8  0  0 44  0 22 10   0  0  0  0 38  0  0  0  0   0   2  0  3
## E 0  6  0 45  0  0 16  0   0 29  0  0  0  0 15  0  0   0   2  0  0
## F 0  0  7  0  0  0  0  0   1  0 38  0  0  0  0  0 26   0   8  0  7
## G 0 36  3 19 30  0  0  0   0  0  0  0  0  0  0 38 33   0  15  0  0
## H 0  0  0  6  0  0  0  0   0  0  2  0  3  2 21 65  0   0   0  0 30
## I 0  0  0  0  0  2  0  0   0  0 12 17  9  0  0  0  9  47 129  0  0
## K 0  0  0  0 40  0  0  0   0  0  0  0 19  0 15 47  0  11   0  0  0
## L 0  0  0  0  0 38  0  5   2  0  0  9  0 53  5  3 16   0  31  1  0
## M 0  0  0  0  0  0  0  0  15  6 11  0  0  0  0  1  0  66  58  0  0
## N 0  0  0 57  0  0  0  8   0  9  0  0  0  0  0  0 57  11   0  0  4
## P 0 29  0  0  0  0  0  7   0  0 88  0  0  0  2 15 52  35   0  0  0
## Q 0  0  0  0 20  0  0 11   0  7 14  0  0  4  0 81  0   0   0  0  0
## R 0  0 54  0  0  0 35 89   2 44  4  3  0 18 85  0 42   9   0 27  0
## S 0 35  7  0  0  6 64  0   4  0 22  0 55 42  0 22  0  27   0  0  2
## T 0 75  0  0  0  0  0  0  57  7  0 59  9 32  0  8 49   0   0  0  0
## V 0 97  0  3  0 11  9  0 108  0 52 60  0  0  0  0  0   0   0  0  0
## W 0  0  0  0  0  0  0  0   0  0  0  0  0  0  0 10  0   0   0  0  0
## Y 3  0 15 11  0  4  0  9   0  0  0  0  6  0  0  0  0   0   0  0  0
```

```
heatmap.2(mutations.mat, col=colorRampPalette(c("blue", "red")),
          density.info="none", trace="none", dendrogram="none",
           symm=F,symkey=T,symbreaks=T, scale="none")
```



Deleterious Score (DScore)

We define Dscore as following (for discussion see the writing section)

$$DScore = \alpha Charge + \beta Size + \gamma GERP + \delta Blosum$$

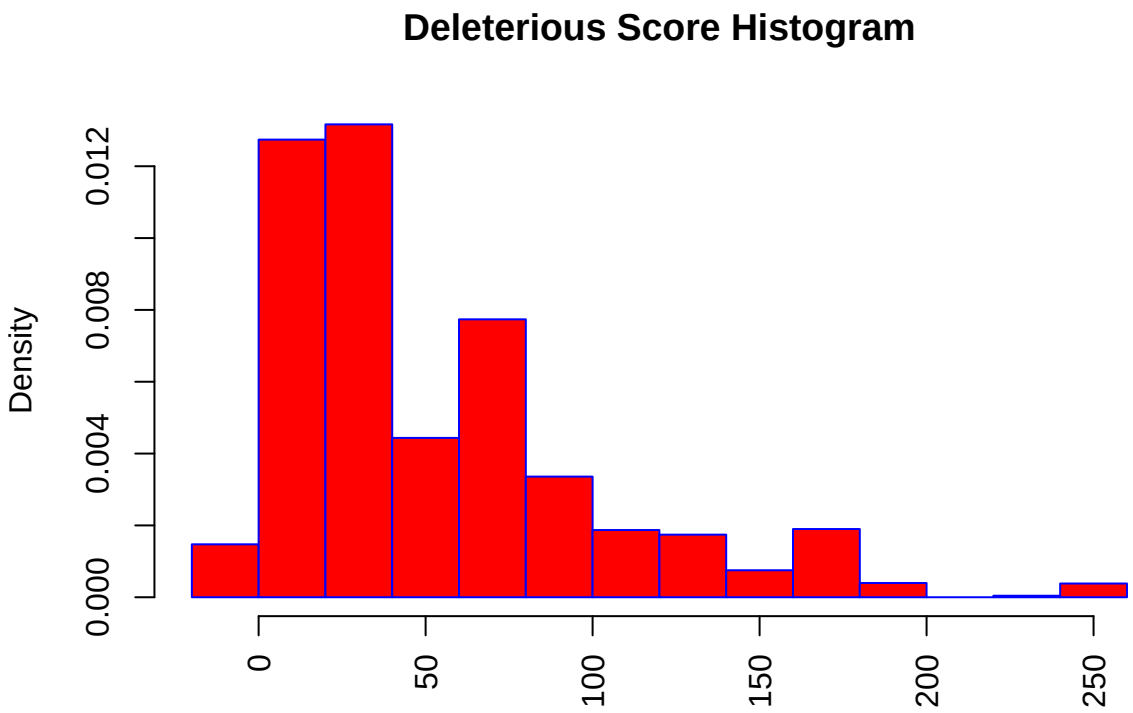
Each one of the coeffiecients can be determined by the user. We chose for example:

$$\alpha = \beta = \gamma = \delta = 1$$

```
coeff <- list(c=1, s=1, b=1, g=1) # This we can get as user input
mutations$score <- coeff$c*mutations$cc + coeff$s*mutations$size +
  coeff$b*mutations$blosum + coeff$g*mutations$gerp
m <- which.max(mutations$score)
print(paste("The most deleterious mutations is:", x$loc[m]))
```

```
## [1] "The most deleterious mutations is: chr3:183975365:C:T"
```

```
hist(mutations$score, probability = TRUE, las=3,
     main="Deleterious Score Histogram", xlab="",
     border="blue", col="red")
```



As one can see, the scoring histogram is right skewed: most mutations are meaningless, while very few might be actually deleterious