Computational Thinking Cancer Genomics - report 2

Robin Hofmeister

June 8, 2018

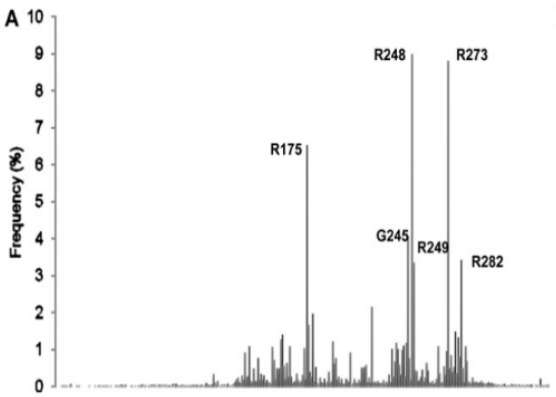
Question 1 (day1/2)

To design a computational approach aimed at identifying cancer functional mutations, which features should be taken into account and why?

To define whether a mutation is functional, an approach is to look for similar features with known functional mutations. These features are:

- **recurrence**: the frequency of alterations may be indicative of their functional importance. A mutation that happens frequently in the population is more likely to be important than a mutations that occurs in a single individual. In the *Figure 1.A* for instance is shown the frequency of several mutations across individuals (N=19). Mutations in amino acids that occurs the more frequently across individuals are indicated by their positions and reported in *Figure 1.B*. These mutations are more likely to be functional than mutations with a weak frequency.
- **mutation Hotspot**s: Even if the frequency of mutations is weak, those mutations may be functional if they occur in the same codon or in the same functional or structural domain. For instance, we can see in the *Figure 1.C* that two alterations impact the residue 245: G245S and G245D. These alterations will have the same impact on the transcript function because they occur in the same codon.
- **amino-acid conservation**: Region of ortholog genes that are well conserved among different species are regions that are likely to be critical for the protein structure or function. For instance in *Figure 2* is represented an alignment of RASA1 sequence around the PH domain among several species. We can see that sequences are well conserved across species and that five mutations that are identified in melanomas are localized in the conserved PH domain.
- **prior biological knowledge**: "Rational behind clinical sequencing". With the news sequencing technologies, there is ever larger data available about more and more diseases. These data can serve as a basis for comparing new mutations identified. The knowledge acquired in the scientific field should serve as a basis on which new research should be based.

Reference Fig. 1, Reference Fig. 2



B

Amino Acid Residue	Frequency of Alteration
248	7.0%
273	6.7%
175	5.1%
245	3.3%
249	2.9%
282	2.9%

C

Mutation	Overall Frequency	Wild-Type Codon	Mutant Codon	CpG	Class
R175H	4.6%	CGC	CAC	Yes	Conformation
R248Q	3.5%	CGG	CAG/CAA	Yes	DNA Contact
R273H	3.1%	CGT	CAT	Yes	DNA Contact
R248W	2.8%	CGG	TGG	Yes	DNA Contact
G245S	2.8%	GGC	AGC	Yes	Conformation
R273C	2.7%	CGT	TGT	Yes	DNA Contact
R282W	2.4%	CGG	TGG	Yes	DNA Contact
R249S	1.8%	AGG	AGT	No	Conformation
G245D	0.68%	GGC	GAC	No	Conformation

Figure 1: Tp53 mutations spectrum in human cancers. (A) Distribution of TP53 missens mutations.(B) Table of the mutational frequency of six amino acid 'hot spot' residues. (C) Table of most common alteration at 'hot spot' residues.

Figure 2: Alignment of RASA1 PH domain among differents species with a representation of five mutations located in the PH domain.

Question 2:

Using the complete set of gene expression signatures (GES) from (Gatza et al.) (see Day4/data/signatures_gatza.RData):

Part a) compute the naive GES score for each signature: The naive GES score consist in the average expression of the genes in the gene expression signature. First, the expression of genes are transformed with log2 and standardized. Second, the naive GES score is computed with the function "mean(x)" applied on columns. The data we used for this exercise is *signatures_gatza.RData*. It contains several sets of genes expression signatures (GES). To investigate the GES score for each signature, we used the same methods as for the data *PAM50_signature.RData*, which contain one single signature, and we apply this method to each signature contained in *signatures_gatza.RData* with a for loop.

Part b) for each GES, test whether breast cancer subtypes have significantly different scores: For each GES score previously computed, we used a wilcoxon test to identify whether the differents tumor subtypes (LumA, LumB, Her2, Normal, Basal) have significant different GES score. We created a list (subtypes_names) to go through all subtypes with a for loop and compute a wilcoxon test for each combination of subtypes in each signature.

Part c) report significant associations: We created a matrix with 'signature names' as rows and 'combination of subtypes' as columns. We fill the matrix with the corresponding pvalue resulted from the wilcoxon test between subtypes. We added a threshold (0.05) to keep only significant association in the matrix. The non-significant association are indicated by 'X'. Because of its size, the matrix of results is not shown but can be visualised by running the associated Rmarkdown . To summarize results, we can see in *Table 1* the number of significant signature scores between subtypes. We can notice that each combination of subtypes have more than the half of signatures that exhibits significant associations. For example, subtypes Lum_A and Lum_B (*La* and *Lb*, respectively) have 38 signatures with significant different scores.

Code question 2:

```
a=load('BRCA.rnaseq_ge.RData')
b = load('signatures_gatza.RData')
c = load('BRCA_subtypes.RData')
subtypes = BRCA_subtypes[colnames(ge),'class']
names(subtypes) = colnames(ge)
signatures_gatza = lapply(signatures_gatza, function(x) x[which(x %in% rownames(ge))])
```

- **Matrix to store pvalues from wilcoxon test:**

```
names_col=c("LumA/LumB", "LumA/Her2", "LumA/Normal", "LumA/Basal",
            "LumB/Her2", "LumB/Normal","LumB/Basal", "Her2/Normal",
            "Her2/Basal", "Basal/Normal")
matrix_pvalues=matrix('X',nrow=length(signatures_gatza), ncol=length(names_col))
rownames(matrix_pvalues)=names(signatures_gatza) # each row is a signature
colnames(matrix_pvalues)=names_col # each column is an association of subtypes
subtypes_names=c('BRCA_LumA','BRCA_LumB','BRCA_Her2','BRCA_Normal','BRCA_Basal')
```

- **Loop to to comute the naive GES score for each signature (part A), to test subtypes associations (part B) and to store pvalues in the matrix (part C):**

```
c=1
for (i in signatures_gatza){

  #Part a): compute the naive GES score for each signature
```

```
  signature=i
  signatures_ge=ge[signature,]
  signatures_ge=log2(signatures_ge+1)
  ge_standardized = t(apply(signatures_ge,1,function(x) (x - mean(x))/sd(x)))
  ges_signatures = round(apply(ge_standardized, 2, function(x) {mean(x)}),3)

  # Part b): for each GES, test whether beast cancer subtypes have
  # significantly different scores

  for (i in 1:length(subtypes_names)){
    for (j in i:length(subtypes_names)){
      if (i!=j){

        pvalue=wilcox.test(ges_signatures[which(subtypes == subtypes_names[i])],
                           ges_signatures[which(subtypes == subtypes_names[j])])$p.value

        # Part c): report significant associations

        if (pvalue<0.05){
          if (i==1){
            matrix_pvalues[c,j-i]=pvalue
          }
          if (i==2){
            matrix_pvalues[c,i+j]=pvalue
          }
          if (i==3){
            matrix_pvalues[c,i+j+1]=pvalue
          }
          if (i==4){
            matrix_pvalues[c,i+j+1]=pvalue
          }

        }
      }
    }
  }
  c=c+1 # go to the next row in the matrix for the next subtype
}
# print(matrix_pvalues) # this matrix represents only significant pvalues (<0.05).
#Not shown here because of its size.
```

- **To summarize the matrix of pvalues, we also compute the number of signatures with significant associations for a each combination of subtypes.**

```
nbr_significant=matrix(nrow = 1, ncol=dim(matrix_pvalues)[2])
colnames(nbr_significant)=c("La/Lb", "La/H2", "La/N", "La/B", "Lb/H2",
                            "Lb/N","Lb/B", "H2/N", "H2/B", "B/N")
rownames(nbr_significant)='Number of signature'

for (j in 1:dim(matrix_pvalues)[2]){
  sum=0
  for (i in 1:dim(matrix_pvalues)[1]){
    if (matrix_pvalues[i,j]=='X'){
      sum=sum+1
```

```
    }
  }
  nbr_significant[j]=dim(matrix_pvalues)[1]-sum
}
panderOptions("table.split.table", Inf)
pander(nbr_significant)
```

	La/Lb	La/H2	La/N	La/B	Lb/H2	Lb/N	Lb/B	H2/N	H2/B	B/N
Number of signature	38	44	29	45	34	36	41	35	38	41

Table 1: Number of signature with significant different score between subtypes. *(La=Lum_A; Lb=Lum_B; H2=Her_2; N=Normal; B=Basal)*

Question 3:

Assess which probability distributions best fits the total number of mutations and total number of copy number alterations in breast cancer (Day 3/mutations_brca.txt and copy_number_brca.txt):

To assess the probability distribution that best fits the data, we use the the library *fitdistrplus*. It provides a function with which we can chose the distribution to fit (*fitdist*) and a function to statistically test which distribution probability best fits the data with different test (Kolmogorov-Smirnov test, Cramer-von Mises test, Anderson-Darling test).

a)We used in a first time the data *mutation_brca.txt*: We begin by plotting the empirical distribution of data (Figure 3), and we can notice that it seems not normally distributed. We then use the fitdist function to fit the data with probability distributions suggested (normal, log-normal, poisson). We can notice on Figures 4 and 5 that none of these probability distributions seem to fit well the data, even if the log-normal distribution probability is the better among these three distributions.

To go further, we use the library *actuar*, that provides more probability distributions to fit. The fitting of these distribution is shown on Figures 6 and 7. We can notice this time that the Burr probability distribution fits well the data. Finally, we use the built-in function to test whether the probability distribution is rejected or accepted. Results of the test are shown in table 2. The only distribution that is not rejected is the Burr distribution with the Kolmogorov-Smirnov test, which confirm the expectations from plots.

b)The same method is applied on the data *copy_number_brca.txt* (code not shown, can be seen in the Rmarkdown). The empirical distribution of the data is shown is Figure 8. Figure 9,10 represent the fit with normal, log-normal and poisson distrubution. Figure 11 and 12 represent the fit with Burr, Weibull, Pareto and loglogistic. The results of the statistical test is shown in table 3. For this data, none probability distribution is accepted. However, it can be noticed that the distribution of copy_number seems weel fits the lognormal distribution in Figure 10 and 12.

Code question 3: Mutation_brca.txt

```
library(fitdistrplus)
library(actuar)
mutations<- read.table('mutation_brca.txt')
```

- Empirical distribution:

```
plotdist(mutations$V1, histo=TRUE, demp=TRUE) #Figure 3
```

- Test to fit the three probability distributions suggested and plots:

```
norm_mut<-fitdist(mutations$V1,"norm")
poi_mut<- fitdist(mutations$V1, "pois")
log_mut<-fitdist(mutations$V1, "lnorm")

denscomp(list(norm_mut,poi_mut,log_mut),  #Figure 4
         legendtext=c("normal","poisson","lognormal"))
```

```
cdfcomp(list(norm_mut,poi_mut,log_mut) , xlogscale=TRUE, #Figure 5
        ylogscale=TRUE, legendtext=c("normal","poisson","lognormal"))
```

- Test more distributions to fit in the library actuar:

```
wei_mut<- fitdist(mutations$V1, "weibull")
gamma_mut<- fitdist(mutations$V1, "gamma")
llogis_mut<- fitdist(mutations$V1,"llogis", start = list(shape = 1, scale = 500))
```

```
pareto_mut<- fitdist(mutations$V1, "pareto",start = list(shape = 1, scale = 500))
burr_mut<- fitdist(mutations$V1, "burr", start = list(shape1 = 0.3, shape2 = 1, rate = 1))

denscomp(list(norm_mut, poi_mut, log_mut, wei_mut, #Figure 6
              gamma_mut, llogis_mut,pareto_mut, burr_mut),
         legendtext= c("normal","poisson","lognormal","weibull",
                       "gamma", "loglogistic", "Pareto", "Burr"))
```

```
cdfcomp(list(norm_mut, poi_mut, log_mut, wei_mut, #Figure 7
             gamma_mut, llogis_mut,pareto_mut, burr_mut),
        xlogscale=TRUE, ylogscale = TRUE, legendtext=
          c("normal","poisson","lognormal","weibull", "gamma","loglogistic",
            "pareto", "Burr"))
```

- Test statistic:

```
results<-gofstat(list(norm_mut, poi_mut, log_mut, wei_mut, gamma_mut,
                      llogis_mut,pareto_mut, burr_mut))
result_cvm<-results$cvmtest
result_ad<-results$adtest
result_ks<-results$kstest
results<-c(result_cvm,result_ad,result_ks)
r<-matrix(results,ncol=8,nrow=3, byrow =TRUE)
rownames(r)<-c("cvm","ad","ks")
colnames(r)<-c("normal","poisson","lognormal","weibull", "gamma", "loglogistic",
               "Pareto", "Burr")
pander(r) # Table 2
```

	normal	poisson	lognormal	weibull	gamma	loglogistic	Pareto	Burr
cvm	not computed	not computed	not computed	rejected	rejected	not computed	not computed	not computed
ad	not computed	not computed	not computed	rejected	rejected	not computed	not computed	not computed
ks	rejected	rejected	rejected	rejected	rejected	rejected	rejected	not rejected

Table 2: Choice to reject or not a probability distribution for the data *mutation_brca.txt*.

	normal	poisson	lognormal	weibull	gamma	loglogistic	Pareto	Burr
cvm	not computed	not computed	not computed	rejected	rejected	not computed	not computed	not computed
ad	not computed	not computed	not computed	rejected	rejected	not computed	not computed	not computed
ks	rejected	rejected	rejected	rejected	rejected	rejected	rejected	rejected

Table 3: Choice to reject or not a probability distribution for the data *copy_number_brca.txt*.

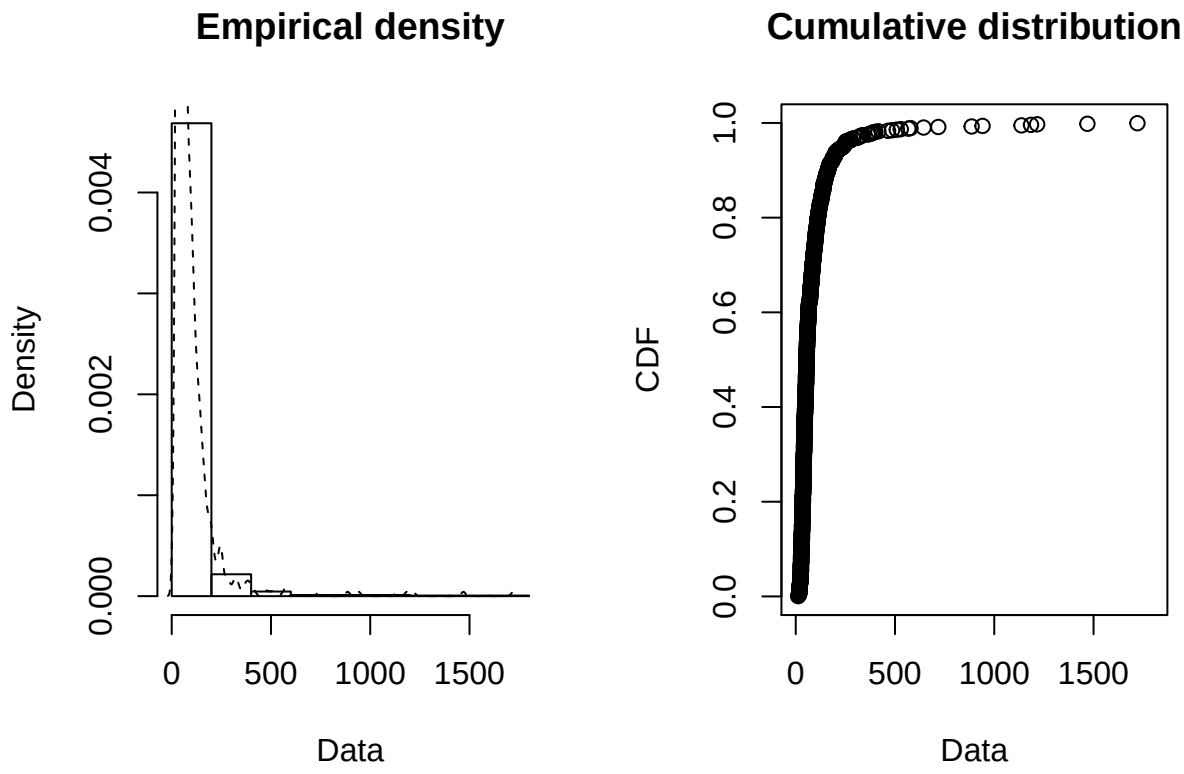


Figure 3: Mutations: Empirical distribution of mutations

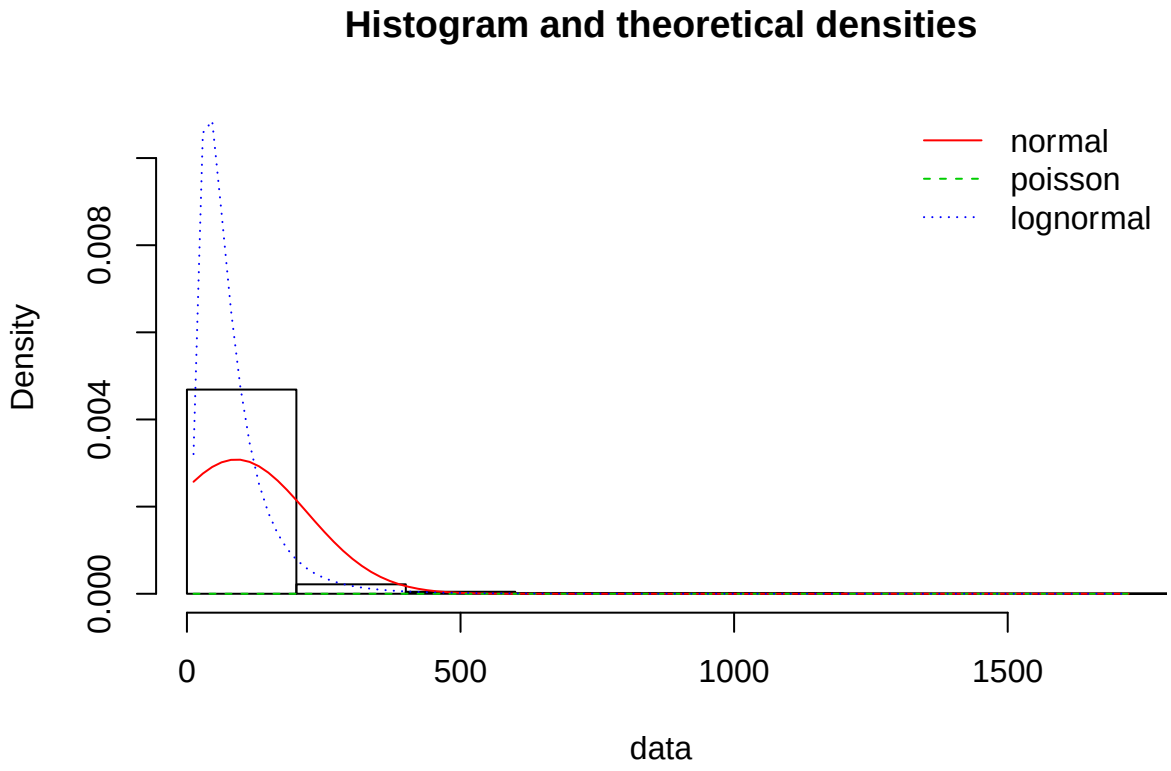


Figure 4: Mutations: Fitting distributions with density function: normal, poisson and log-normal distributions

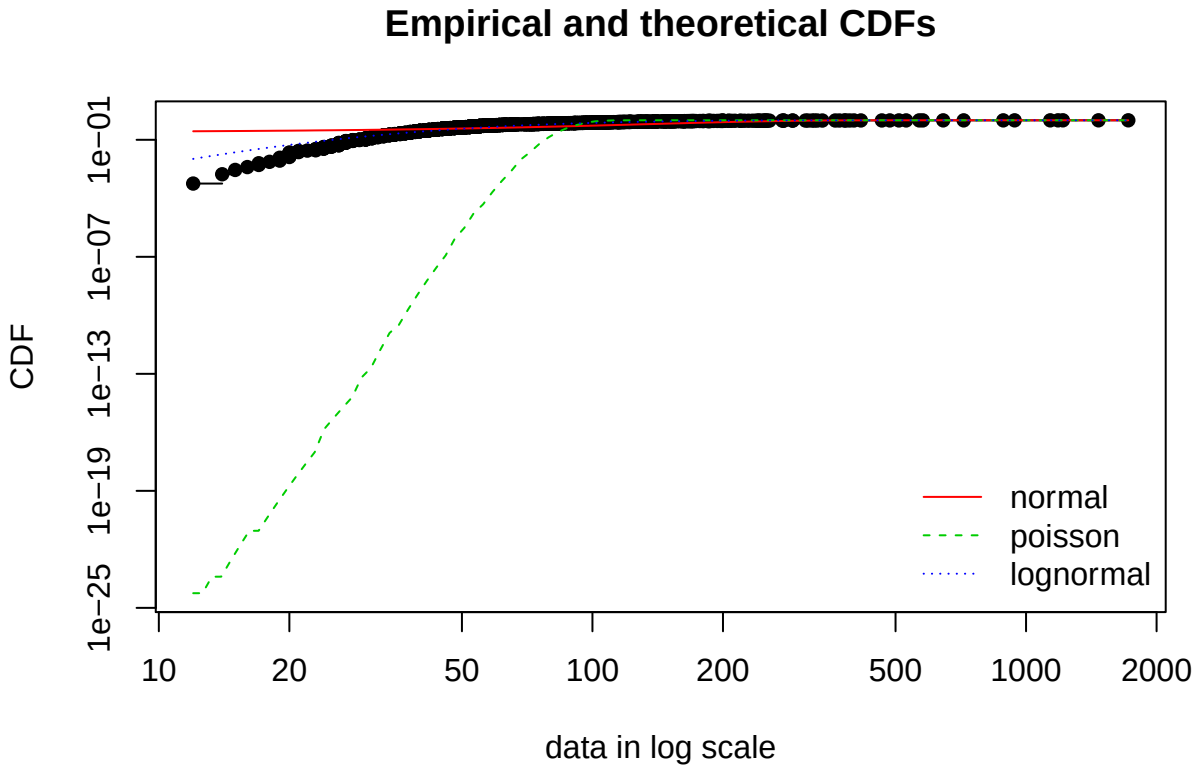


Figure 5: Mutations: Fitting distributions with cumulative distribution function: normal, poisson and log-normal distributions

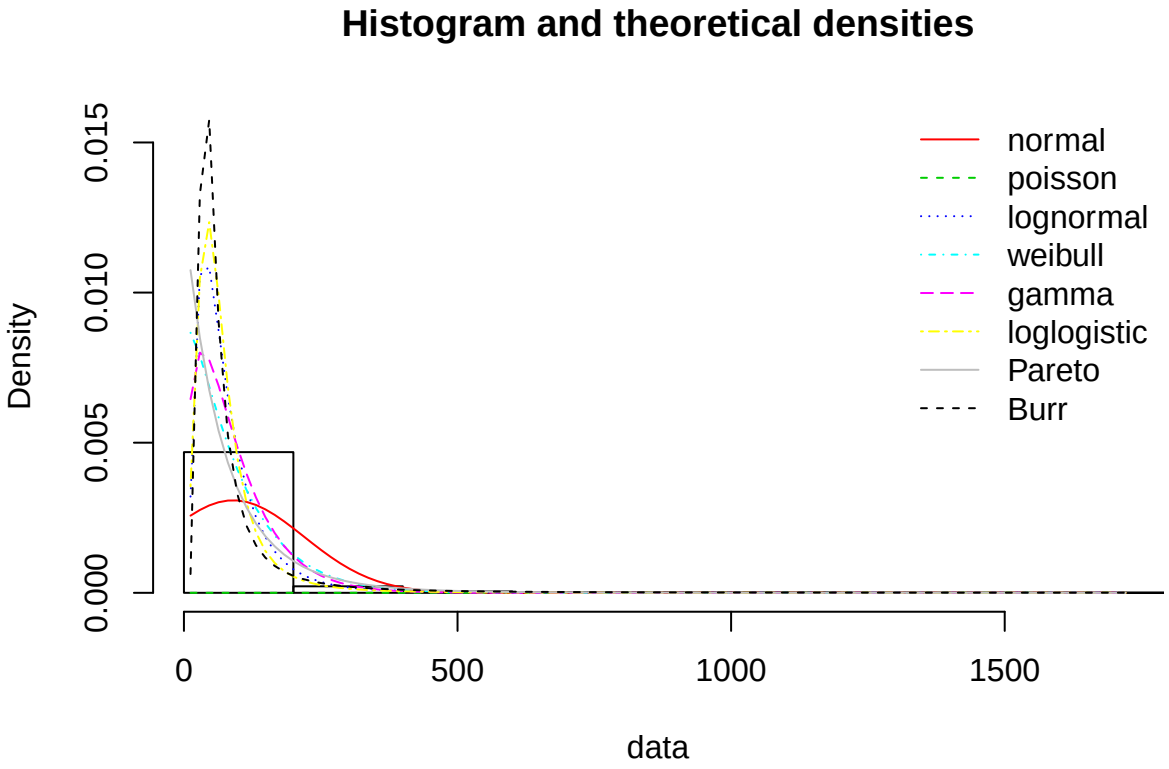


Figure 6: Mutations: Fitting distributions with density function: normal, poisson, log-normal, weibull, gamma, loglogistic, pareto

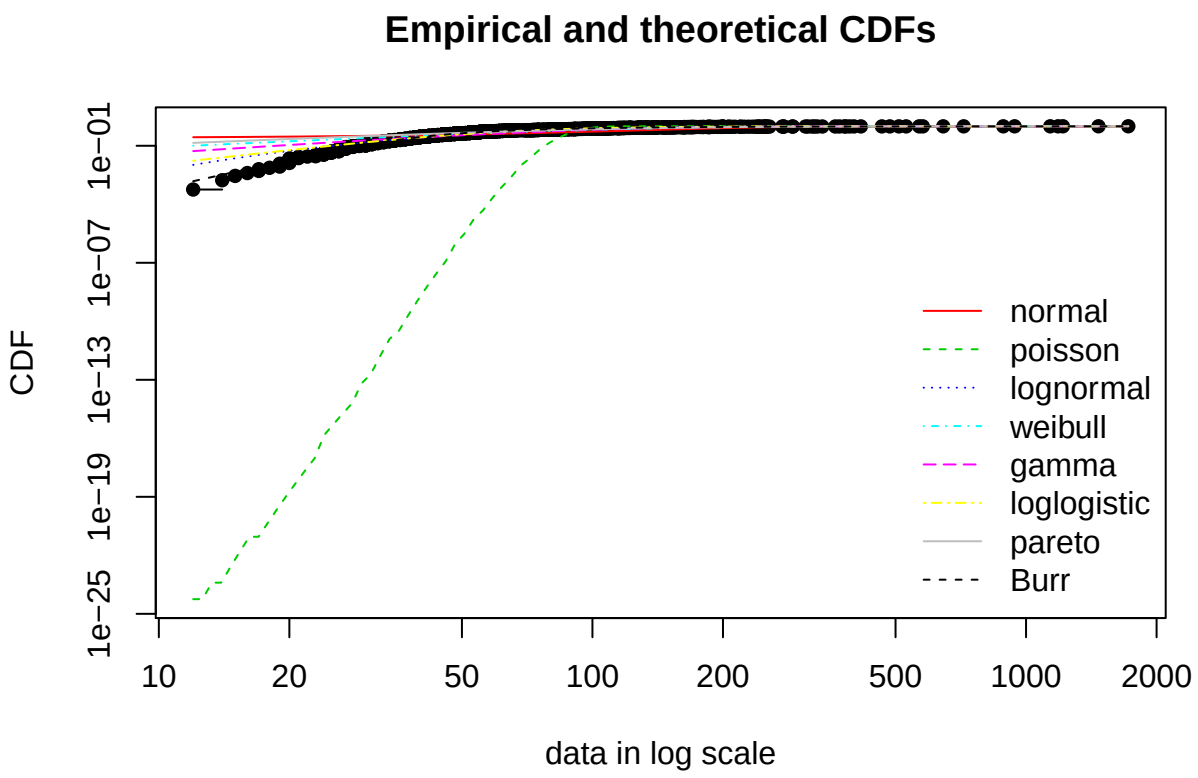


Figure 7: Mutations: Fitting distributions with cumulative distribution function: normal, poisson, log-normal, weibull, gamma, loglogistic, pareto

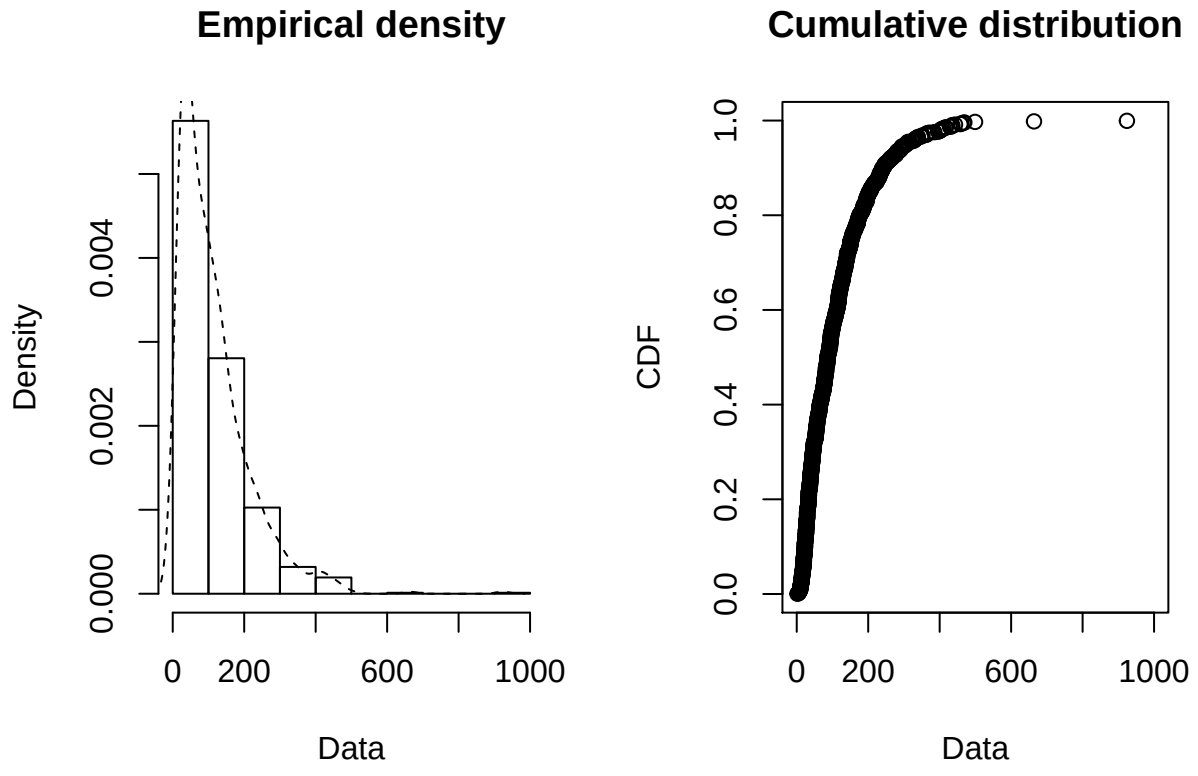


Figure 8: CNVs: Empirical distribution of copy_number

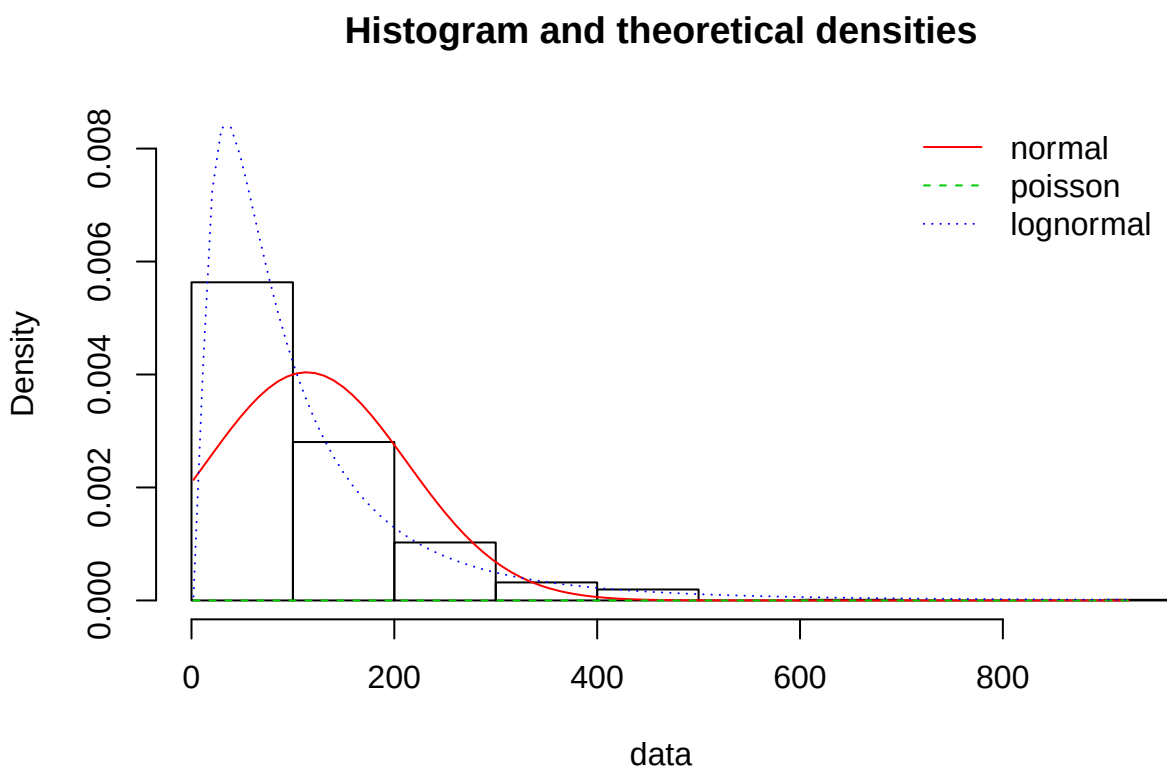


Figure 9: CNVs: Fitting distributions with density function: normal, poisson and log-normal distributions

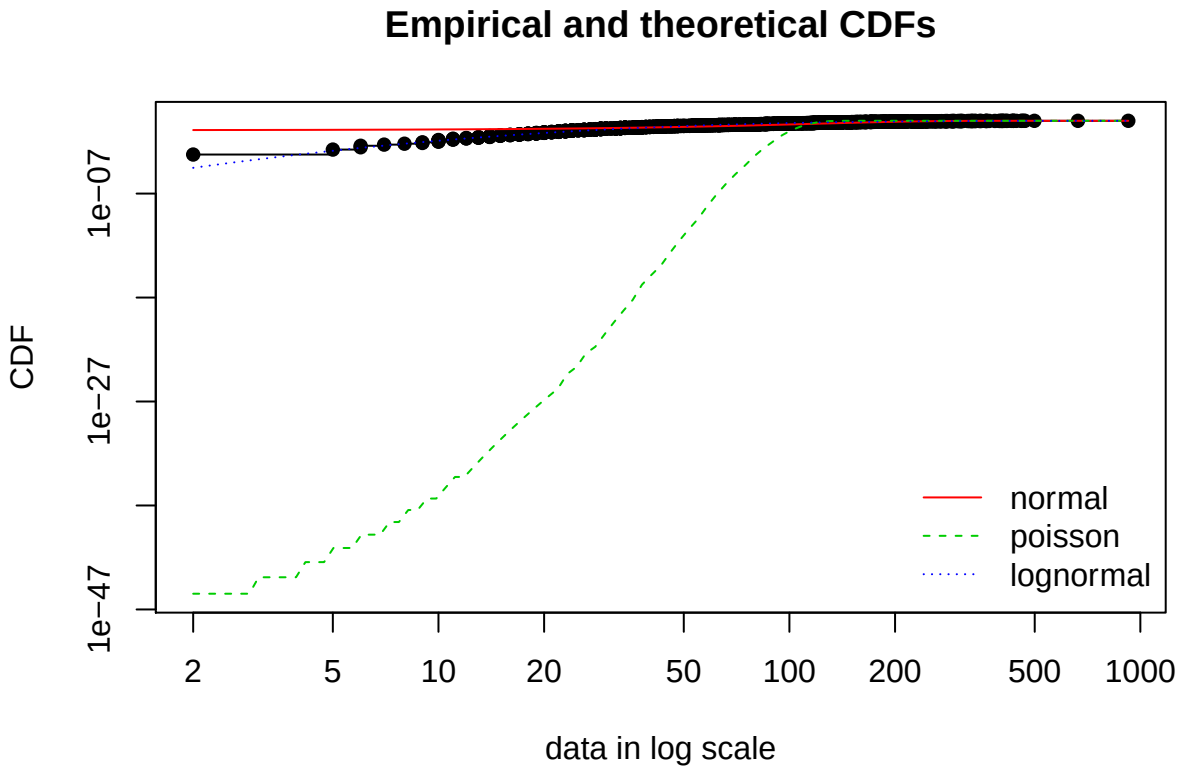


Figure 10: CNVs: Fitting distributions with cumulative distribution function: normal, poisson and log-normal distributions

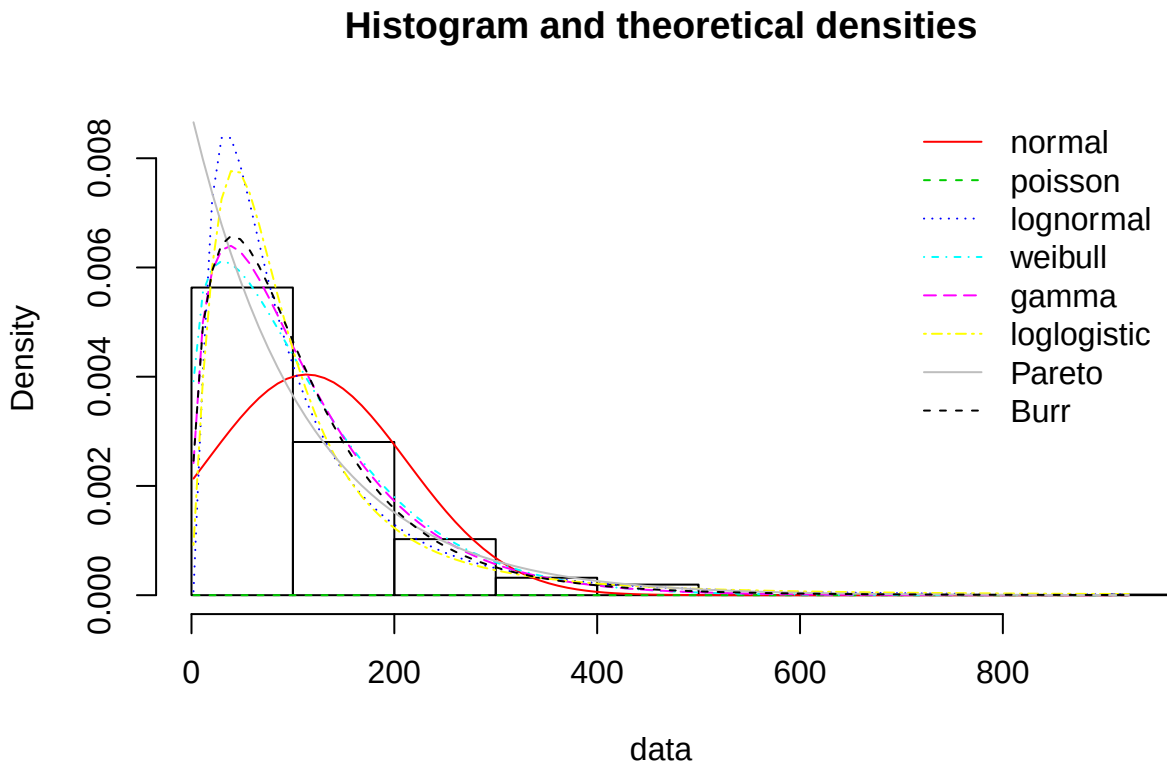


Figure 11: CNVs: Fitting distributions density function: normal, poisson, log-normal, weibull, gamma, loglogistic, pareto

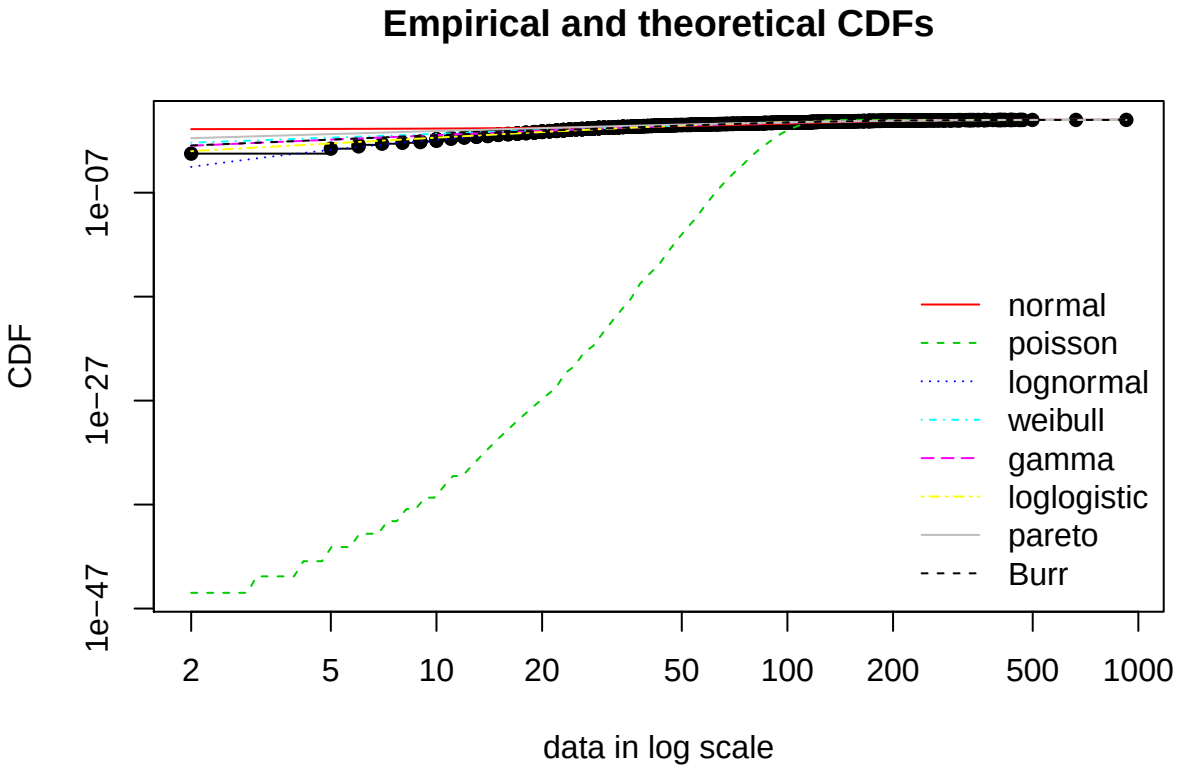


Figure 12: CNVs: Fitting distributions with cumulative distribution function: normal, poisson, log-normal, weibull, gamma, loglogistic, pareto