Sample Contamination

Lai Ping Wong laiping.wong@roswellpark.org

2017-08-16

Contents

1 Introduction

sampleCont R package predicts sample contamination base on variant allele frequency (VAF) discovered from genome sequencing of cancer samples. It requires one configuration file and two input data files

- configuration file
- variant allele frequency, n x m data frame (VAF.out)
- variant coverage, n x m data frame (VAFcov.out)

An example of VAF, VAFcov data and its configuration file are distributed with this package under inst/extdata/.

2 Installation

To install this package:

```
install_github("sampleCont", quick=TRUE)
library(sampleCont)
```

3 Quick start single R command line execution

```
data_path <-  system.file('extdata',package='sampleCont')
output_path <- 'output_express'
config_file <- system.file("extdata", 'config.txt',
    package = "sampleCont", mustWork = TRUE)

run_sampleContamination(data_path = data_path,
                        output_path = output_path,
                        config_file = config_file)
```

4 Step by step execution

The purpose of this section is to provide users a step by step workflow of the sample contamination prediction using test data distributed with the package.

4.1 Setup parameters

```
config_file <-  system.file("extdata", 'config.txt',
    package = "sampleCont", mustWork = TRUE)
data_path <- system.file('extdata',package='sampleCont')
output_path <- paste0(getwd() ,'/output')

# set parameters
setParameterConfig(config_file)

# set VAF and COV data file
vaf_file <- paste0(data_path,"/VAF.out")
cov_file <- paste0(data_path,"/VAF_cov.out")
```

4.2 Load VAF data

```
VAFdata <- inputData(vaf_file) # VAF
head(VAFdata)
```

mutationID	CAP001-Nb-K	CAP001-Td1c-K	CAP002-Nb	CAP002-Td1b-K
1.103468336.A.C	0	0	0.2667	0
1.10510320.G.A	0	0	0.4643	0.3333
1.109391557.G.A	0	0	0.5405	0.4699
1.109428144.G.C	0	0	0	0
1.109477466.G.T	0	0	0	0
1.110033736.G.A	0.4651	0.5703	0	0.05556

CAP003-Nb-K	CAP003-Td1a	CAP003-Td1b	publicDB	freq
0	0	0.08	unknown	2/7
0.1236	0	0	unknown	3/7
0.07407	0	0	unknown	3/7
0	0.15	0.09574	unknown	2/7
0	0	0.1522	unknown	1/7
0	0	0	unknown	3/7

4.3 Load COV data

```
VAFcov <- inputData(cov_file) # COV
head(VAFcov)
```

mutationID	CAP001-Nb-K	CAP001-Td1c-K	CAP002-Nb	CAP002-Td1b-K
1.103468336.A.C	38	30	30	26
1.10510320.G.A	52	60	56	72
1.109391557.G.A	112	93	74	83
1.109428144.G.C	102	89	63	73
1.109477466.G.T	83	88	60	117
1.110033736.G.A	129	128	104	144

CAP003-Nb-K	CAP003-Td1a	CAP003-Td1b
46	27	50
89	44	38
108	75	86
107	60	94
106	66	46
157	90	50

4.4 Calculate number of SNPs for each sample

```
SNPcount <- numMutationperSample(VAFdata,VAF_cutoff,n_sample)
SNPcount
```

	numMut
CAP001-Nb-K	206
CAP001-Td1c-K	244
CAP002-Nb	205
CAP002-Td1b-K	292
CAP003-Nb-K	235
CAP003-Td1a	101
CAP003-Td1b	135

4.5 Count number of common SNPs for pairwise sample

```
SNPshare<- pairShare(VAFdata,VAF_cutoff,n_sample)
SNPshare
```

	CAP001-Nb-K	CAP001-Td1c-K	CAP002-Nb	CAP002-Td1b-K
CAP001-Nb-K	206	140	18	118
CAP001-Td1c-K	140	244	76	201
CAP002-Nb	18	76	205	149
CAP002-Td1b-K	118	201	149	292
CAP003-Nb-K	32	75	166	147
CAP003-Td1a	43	1	1	1
CAP003-Td1b	58	1	33	4

	CAP003-Nb-K	CAP003-Td1a	CAP003-Td1b
CAP001-Nb-K	32	43	58
CAP001-Td1c-K	75	1	1
CAP002-Nb	166	1	33
CAP002-Td1b-K	147	1	4
CAP003-Nb-K	235	54	54
CAP003-Td1a	54	101	83
CAP003-Td1b	54	83	135

4.6 Predict sample contamination

```
# set sample IDs
sampleID <- colnames(VAFcov)[-1]

# calculate pcomm
pcomm <- pairPCommon(VAFdata,VAF_cutoff,num_round_digit,n_sample)

# tabulate all pairwise samples
delimeter<-":"
```

```
pairs <- pairList(VAFdata,n_sample,delimeter)

# generate sample pairs info
pids <- t(sapply(as.matrix(pairs), function(i) unlist(strsplit(i, delimeter))))
colnames(pids) <- c("pid1", "pid2")
sample_pairs <- data.frame(pairID = pairs, pids)

# count number of mutation in 7 regions of VAF scatter plot
countPoint <- regionCountMutation(sample_pairs, VAFdata,
                                  SNPcount, SNPshare,
                                  VAF_cutoff, VAF_ignore, n_sample)
sample_pairs <- cbind(sample_pairs, countPoint)

# identify relation
rel <- pairRelation(sample_pairs,center_cutoff,source_cutoff,
                    target_cutoff,localPcomm_cutoff,region_cutoff,
                    num_round_digit,output_path)
sample_pairs <- cbind(sample_pairs, rel)


# eliminate multi-sources contamination by using Fisher test
final_rel <- as.data.frame(multipleSource(sample_pairs,VAFdata ,VAFcov,VAF_cutoff,
                                          VAF_cutoff1,p.val_cutoff,output_path))

# calculate mixing ratio for contamination pairs
mr <- as.data.frame(mixingRatio(VAFdata,VAFcov,sample_pairs,
                                final_rel,VAF_cutoff,VAF_ignore,FALSE,output_path))

# filter multi-source contamination by min distance to the predicted contamination level
# weighted with min coverage
mr_filt <- as.matrix(unique(cbind(source=mr$source,target=mr$target,
                                  predicted_contamination_perc=
                                    as.numeric(mr[,'lm_coeff'])*100)))

final.pred <- as.data.frame(filterMultiSources(mr_filt,output_path,
                                               VAFdata,VAFcov,uniq_both = 2))

# plot circos
same.subject <- sample_pairs[sample_pairs$rel=="00",]
mr_df <- as.data.frame(cbind(mr[,'source'],mr[,'target'],
                             as.numeric(mr[,'lm_coeff'])*100,
                             mr[,'rel'],mr[,'flip']))
colnames(mr_df) <- c('source','target','predicted_contamination_perc','rel','flip')

# attach relation
tmp = unique(merge(mr_df, final.pred, by=c('source','target'))[1:4])
tmp = tmp[order(tmp$rel,decreasing=T),]

contaminate <- tmp[!duplicated(tmp[,c('source','target',
                                      'predicted_contamination_perc.x')]),]
colnames(contaminate) <- c('source','target','link','rel')
contaminate$link <-as.numeric(contaminate$link)/10 # circos link weightage
contaminate[contaminate$rel=='01','rel'] <- '10'
```

```
if(nrow(same.subject)>0){
  same.subject.mr <- as.data.frame(mixingRatio(VAFdata,VAFcov,
                                               sample_pairs,final_rel,VAF_cutoff,
                                               VAF_ignore, ALL_flag = TRUE,
                                               sameSubject=TRUE))

  same.subject.out <- as.data.frame(cbind(source=same.subject.mr$source,
                                          target=same.subject.mr$target,
                                          link=round(10*
                                                     as.numeric(same.subject.mr$lm_coeff),
                                                     num_round_digit),
                                          rel=same.subject.mr$rel))
  circos.plot<- as.data.frame(rbind(same.subject.out,contaminate))
}else{
  circos.plot<- contaminate
}

plot_circos_link ( circos.plot,R=220,W=12,plotsize=800,titleStr="",
                   seg.lab.size = 0.7,fig.file="circos.pdf",
                   contaminatedOnly=TRUE,sameSubjectOnly=TRUE,allSamples=sampleID,
                   output_path )
```

5 Sample contamination prediction output

Output consists of two text files and a circos figure at the designated output_path

- allPairRelation.txt
- conPred.txt
- circos.pdf

Table 8: allPairRelation

source	target	rel
CAP001-Td1c-K	CAP001-Nb-K	0
CAP002-Td1b-K	CAP002-Nb	0
CAP003-Td1a	CAP003-Nb-K	0
CAP003-Td1b	CAP003-Nb-K	0
CAP003-Td1b	CAP003-Td1a	0
CAP002-Td1b-K	CAP001-Td1c-K	11
CAP001-Td1c-K	CAP002-Td1b-K	10
CAP002-Nb	CAP003-Nb-K	10
CAP003-Td1b	CAP001-Nb-K	10

Table 9: contPredict

source	target	predicted_contamination_perc
CAP003-Td1b	CAP001-Nb-K	9.387
CAP002-Td1b-K	CAP001-Td1c-K	4.640
CAP002-Nb	CAP003-Nb-K	21.285
CAP001-Td1c-K	CAP002-Td1b-K	10.001

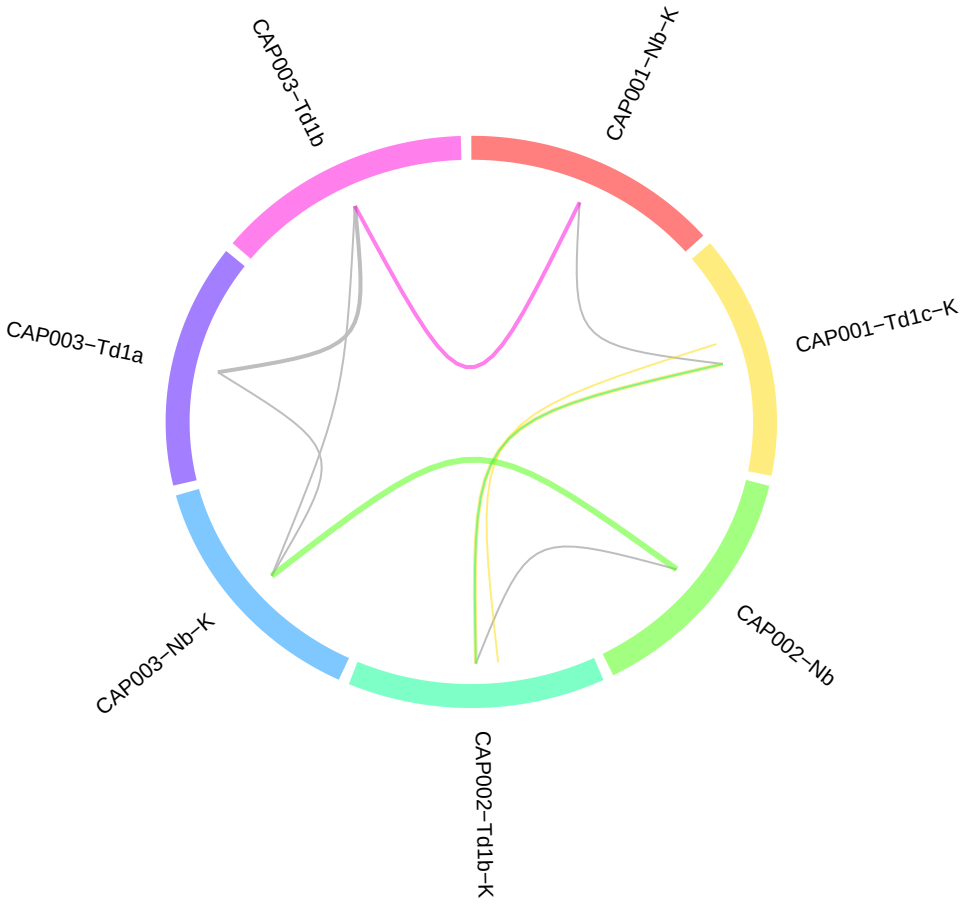

Figure 1: Circos. Each color section of the circos plot represents sample. Grey links show same subject pairs, source contamination sample color link to target sample. Thickness of link reflects the predicted contaminatoin level.