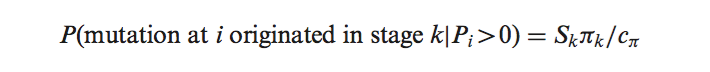
CancerTiming

分析原理：

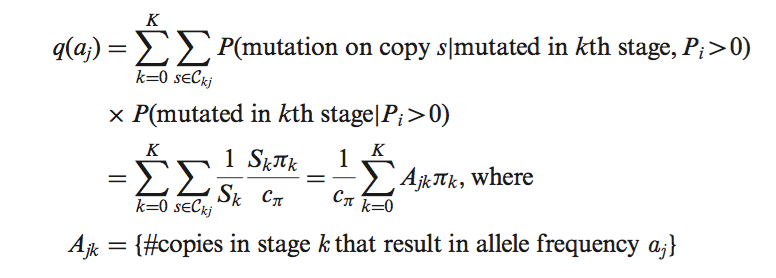
理论假设：

1. 每个突变i历史上只发生过一次
2. 如果一个突变i发生在k阶段，那么它发生在k阶段每个拷贝上的概率(可能性)是相同的
3. 一个突变i发生在k阶段的概率假设与k阶段存在的时间(πk)以及k阶段存在的拷贝数(Sk)正相关。



i : 突变位点 *P*i : 突变位点i的等位基因频率

C π : 



model Xi，固定A，估算q，进而估算π

（一）makeEventHistory *Create the event history matrix*

Description

Create the event history matrix needed for the event timing algorithm

代码：

makeEventHistory(type="gain",totalCopy=8)

makeEventHistory(type="gain",copies=c(1,4),totalCopy=5,onlyIdentifiable=FALSE)

> LOH is only an option for totalCopy <3. S=2 type为LOH

makeEventHistory(type="LOH",totalCopy=2,onlyIdentifiable=FALSE)

$`1MLineage`

[,1] [,2]

[1,] 0 2

[2,] 1 0

> Cannot have 'gain' and totalCopy<3 S>2 type为gain

makeEventHistory(type="gain",totalCopy=3,onlyIdentifiable=FALSE)

（二）eventTiming *Estimate the time of events in tumor data*

Description

Estimate the proportion of time spent between different chromosomal abnormalities based on the allele frequencies of mutated locations.

代码：

data(mutData)

ACNLOH<-matrix(c(0,2,1,0),ncol=2,nrow=2,byrow=TRUE)

onlyMuts<-subset(mutData,is.na(rsID) & position <= 1.8E7)

onlyMuts$t\_depth<-onlyMuts$t\_ref\_count+onlyMuts$t\_alt\_count

x<-eventTiming(x=onlyMuts$t\_alt\_count,m=onlyMuts$t\_depth,

history=ACNLOH,totalCopy=2,type="CNLOH",normCont=0.22)

x$pi #estimate of time of stages

x$q #estimate of the multinomial (likelihood of each of the alleles)

x$call$alleleSet #possible set of alleles after

#adjusting for normal contamination

（三）eventTimingOverList *eventTiming for multiple samples and regions*

Description

eventTimingOverList is a wrapper to eventTiming that allows for timing of common events over several regions of a sample and/or multiple samples. getPi0Summary gets summary information about ⇡0 (the first event) from the output of that function and returns a simple dataframe of the estimate of ⇡0 for every region and sample combination.

if(require(GenomicRanges)){

#fix up mutation data to right format

data(mutData)

colnames(mutData)[1]<-c("chr")

colnames(mutData)[grep("t\_alt\_count",colnames(mutData))]<-"nMutAllele"

colnames(mutData)[grep("t\_ref\_count",colnames(mutData))]<-"nRefAllele"

mutData$nReads<-mutData$nMutAllele+mutData$nRefAllele

mutData$mutationId<-1:nrow(mutData)

#add segmentation annotation -- second region is bogus, only for illustration

segs<-data.frame(chr=c(17,17),start=c(1,1.8e7+100),end=c(1.8e7,81195210),

normCont=0.22,segId=c("Seg1","Seg2"),type=c("CNLOH","SingleGain"))

##Create Trivial segmentation annotation for example

mutId<-mut2Seg(mutData,segs)

eventOut<-eventTimingOverList(dfList=list(Sample1=mutId),normCont=0.22)

getPi0Summary(eventOut)

}

＃＃＃＃

eventOut<-eventTimingOverList(dfList=list(Sample1=mutId,Sample2=mutId),normCont=c(0.22,0.20))