survClip package (Version 0.2.3)

Paolo Martini

November 17, 2017

1 survClip: finding prognostic modules exploiting pathway topology

When working with survival analysis, the most important things are a good (big enought) batch of patients and an accurate annotation of events and other covariates. Many cancer datasets have these features, expecially those collected from TCGA project. In R bioconductor, we can find TCGA data about OV cancer in a package called *curatedOvarianData*. In this brief example, we are going to give an overview of survClip package.

We start by loading the library and the dataset. We used the microarray dataset because RNASeq data row counts are not available. For an example with RNASeq data please refer to our online example at *romauldi.bio.unipd.it*.

```
> library(curatedOvarianData)
> data(TCGA_eset)
> TCGA_eset

ExpressionSet (storageMode: lockedEnvironment)
assayData: 13104 features, 578 samples
  element names: exprs
protocolData: none
phenoData
  sampleNames: TCGA.20.0987 TCGA.23.1031 ...
    TCGA.13.1819 (578 total)
  varLabels: alt_sample_name unique_patient_ID ...
    uncurated_author_metadata (31 total)
  varMetadata: labelDescription
featureData
  featureNames: A1CF A2M ... ZZZ3 (13104 total)
  fvarLabels: probeset gene
  fvarMetadata: labelDescription
experimentData: use 'experimentData(object)'
  pubMedIds: 21720365
Annotation: hthgu133a
```

This dataset consist of 13000 genes measured over 578 patients. All the patients have associated clinical data that include relapse events, vital status and survival rate. In the following chunck of code, we format the clinical data of the phenoData to get them suitable to use in survClip.

```
> names(phenoData(TCGA_eset)@data)

 [1] "alt_sample_name"
 [2] "unique_patient_ID"
 [3] "sample_type"
 [4] "histological_type"
 [5] "primarysite"
 [6] "arrayedsite"
 [7] "summarygrade"
 [8] "summarystage"
 [9] "tumorstage"
[10] "substage"
[11] "grade"
[12] "age_at_initial_pathologic_diagnosis"
[13] "pltx"
[14] "tax"
[15] "neo"
[16] "days_to_tumor_recurrence"
[17] "recurrence_status"
[18] "days_to_death"
[19] "vital_status"
[20] "os_binary"
[21] "relapse_binary"
[22] "site_of_tumor_first_recurrence"
[23] "primary_therapy_outcome_success"
[24] "debulking"
[25] "percent_normal_cells"
[26] "percent_stromal_cells"
[27] "percent_tumor_cells"
[28] "batch"
[29] "flag"
[30] "flag_notes"
[31] "uncurated_author_metadata"

> annot <- phenoData(TCGA_eset)@data
> pid <- annot$unique_patient_ID
> days.recurrence <- annot$days_to_tumor_recurrence
> status.recurrence <- annot$recurrence_status
> days <- annot$days_to_death
> status <- annot$vital_status
> table(status.recurrence)

status.recurrence
norecurrence   recurrence
         279          299

> table(status)

status
deceased   living
     290      270

> status[status=="living"] <- 0
> status[status=="deceased"] <- 1
```

```
> status.recurrence[status.recurrence=="norecurrence"] <- 0
> status.recurrence[status.recurrence=="recurrence"] <- 1
> survAnnot.os <- data.frame(status=as.numeric(status), days=as.numeric(days),
+                            row.names=pid, stringsAsFactors=F)
> survAnnot.pfs <- data.frame(status=as.numeric(status.recurrence), days=as.numeric(days.recurrence),
+                             row.names=pid, stringsAsFactors=F)
```

As results, we build two data.frames that represent the minimal information to run survClip analysis: vital status (status) and the days to death or last follow up (days) for each patient. In this example, we analyze the overall survival. We remove NAs and we sort the samples and the expression matrix according to the survival annotation.

```
> survAnnot <- na.omit(survAnnot.os)
> exp <- exprs(TCGA_eset)
> samples <- colnames(exp)
> samples <- gsub('.', replacement = '-', fixed = T, x = samples)
> colnames(exp)<- samples
> samples <- intersect(samples, row.names(survAnnot))
> survAnnot <- survAnnot[samples,]
> exp <- exp[, samples, drop=F]
```

The expression data are almost ready. We go rapidly through a step of normalization with limma.

```
> library(limma)
> expN <- normalizeQuantiles(exp)
```

Now we can analyze this subset of patients with survClip. First, we need to load pathways. The source of pathway we choose is KEGG from graphite Bioconductor package.

```
> library(graphite)
> kegg<- pathways("hsapiens", "kegg")
```

Then, we need to convert the identifier in geneSymbol since our matrix has been summarized by gene symbols.

```
> cancerPathways <- names(kegg)[grep("cancer", names(kegg))]
> kegg <- convertIdentifiers(kegg[cancerPathways], "symbol")
```

At this stage, we have all the ingriedents needed to perform the analysis with survClip: an expression matrix, survival annotations and a graph. Let's do it! To speed up analysis we are going to extract a selection of cancer related pathways. In the following, you will find how to run whole pathway survival analysis. To improve readability, I reformat results in a table.

```
> library(survClip)
> row.names(expN) <- paste0("SYMBOL:",row.names(expN))
> cancerRelated<-lapply(cancerPathways, function(p) {
+   graph <- pathwayGraph(kegg[[p]])
+   pathwaySurvivalTest(expN, survAnnot, graph,
+                       pcsSurvCoxMethod = "topological", maxPCs=5)
+ })
> names(cancerRelated) <- cancerPathways
> pvalues <-sapply(cancerRelated, function(cr) {
```

```
+   cr@pvalue
+ })
> names(pvalues)<- cancerPathways
> pvalues
```

```
                     Pathways in cancer
                           0.0010633622
Transcriptional misregulation in cancer
                           0.4781860467
                Proteoglycans in cancer
                           0.0442486141
                    MicroRNAs in cancer
                           0.0510126971
                      Colorectal cancer
                           0.2518070165
                      Pancreatic cancer
                           0.4682307685
                     Endometrial cancer
                           0.5400407919
                        Prostate cancer
                           0.0121228746
                       Thyroid cancer
                           0.3259142273
                       Bladder cancer
                           0.4503073811
                 Small cell lung cancer
                           0.7223864431
            Non-small cell lung cancer
                           0.5012585743
                          Breast cancer
                           0.0007908927
    Central carbon metabolism in cancer
                           0.0628361093
           Choline metabolism in cancer
                           0.0049718530
```

Among the other, "Breast cancer" pathway is particularly significan. Let's try to decompose the pathway and see the survival modules. Please note that it is not mandatory to perform whole pathway test in advance.

```
> pathName = "Breast cancer"
> graph <- pathwayGraph(kegg[[pathName]])
> ct <- cliqueSurvivalTest(expN, survAnnot, graph, pcsSurvCoxMethod = "sparse", maxPCs=5)
> getTopLoadGenes(ct)
```

```
        feature clId           geneLoad whichPC
1  SYMBOL:FZD10   13 -0.986669893113836     PC1
2   SYMBOL:FZD1   13  0.622352100170717     PC5
3   SYMBOL:FZD3   13 -0.603012106772038     PC5
4  SYMBOL:FZD10   14  0.993520586336023     PC1
5   SYMBOL:FZD1   14   -0.7152215867918     PC5
6   SYMBOL:FZD7   14 -0.644920046564349     PC5
```

```
7   SYMBOL:FZD5   15 -0.870705558033241     PC3
8  SYMBOL:WNT7A   15 -0.775667314223227     PC5
9    SYMBOL:FOS   23  -0.98918942775966     PC1
10   SYMBOL:APC   35                  1     PC1
```

Calling the function "getTopLoadGenes" we inspect every significant cliques to get the main driver genes.