Vignette for `CINdex` package

Lei Song, Krithika Bhuvaneshwar, Yue Wang*, Yuanjian Feng*, Ie-Ming Shih**, Subha Madhavan, Yuriy Gusev
Innovation Center for Biomedical Informatics, Georgetown University Medical Center
*Virginia Polytechnic Institute and State University
**Johns Hopkins University School of Medicine
Correspondance email: `yg63@georgetown.edu`

May 15, 2016

1 Overview

The `CINdex` package calculates the chromosome instability (CIN) index that allows to quantitatively characterize genome-wide copy number alterations as a measure of chromosome instability. The algorithm for this method will be described in a paper (in preparation).

Genomic instability is known to be a fundamental trait in the development of tumors; and most human tumors exhibit this instability in structural and numerical alterations: deletions, amplifications, inversions or even losses and gains of whole chromosomes or chromosomes arms. The chromosome instability indicated by these copy number alternations is associated with various events in the development or the severity of tumors in terms of clinical outcome.

To mathematically and quantitatively describe these alternations we first locate their genomic positions and measure their ranges. Such algorithms are referred to as "segmentation algorithms". An example of such an algorithm is Fused Margin Regression (FMR)[1].

The CINdex package accepts segmentation results from any segmentation algorithm and calculates the genomic instability across a chromosome for a global view (referred to as "Chromosome CIN"), and the genomic instability across cytobands regions for higher resolution (referred to as "Cytobands CIN"). This allows to assess the impacts of copy number alternations on various biological events or clinical outcomes by studying the association of CIN indices with those events.

This package allows the automated processing of the experimental copy number data generated by Affymetrix SNP 6.0 arrays or similar high throughput technologies. An older version of this algorithm that shows overall instability has been integrated into G-DOC web portal and made available for users as part of the G-DOC *Plus* analysis tools at `https://gdoc.georgetown.edu`[2]. This package calculates not only overall instability, but also gains and losses at the chromosome and cytoband level.

To use the package in R, load it as follows:

```
#source("http://bioconductor.org/biocLite.R")
#biocLite("CINdex")
library(CINdex)
```

2 Preparation of input data

This section has been covered in detail in the second vignette provided as part of this package titled "Prepare input data for CINdex". Load example segment data into the workspace. It is a GRangesList object

```
data("grl.data")
```

2.1 Probe annotation file input

The platform annotation file will be used in the CIN algorithm to obtain location of the cytobands. The cytoband probes and the SNP probes along with the cytoband information must be a GRanges object.

```
# loading the example file
data("cnvgr.18.auto")
data("snpgr.18.auto")
```

2.2 Reference annotation file input

Load hg18 file into workspace. This must be a GRanges object.

```
# loading the example file
data("hg18.ucsctrack")
```

2.3 Clinical data input

Load the example clinical data into the workspace.

```
data("clin.crc")
```

3 Performing an end to end analysis workflow - an example

3.1 Run Chromosome CIN (also called Standard or Regular CIN)

The first step is to call the function that will run Chromosome CIN. That can be done as shown below:

```
run.cin.chr(grl.seg = grl.data)
```

In this example, the rest of the input settings for this function have been kept default. The function allows users to input their own gain and the loss thresholds, although we recommend users to use the default settings.

The code will calculate CIN for the following default settings:

- A combination of gain and loss threshold (18 combinations total)
  - gain threshold 2.5 and loss threshold 1.5
  - gain threshold 2.25 and loss threshold 1.75
  - gain threshold 2.10 and loss threshold 1.90
- For each of these threshold settings, this function will calculate CIN for gains, losses, and a combination of gains and losses (referred to as 'sum')
- For each of the above settings, CIN is calculated with normalization (referred to by the number 2) and without normalization(referred to by the number 3)

So at the end of the function, 18 files will be created inside folder "output_chr" (folder name customizable), each a combination of settings specifically underlined above

```
    dataMatrix_2.5_1.5_normalized_sum.RData
    #gain threshold=2.5, loss threshold=1.5,normalized, showing sum (gains and losses)

    dataMatrix_2.5_1.5_unnormalized_sum.RData
    #gain threshold=2.5, loss threshold=1.5, unnormalized,showing sum (gains and losses)

    dataMatrix_2.5_1.5_normalized_amp.RData
    #gain threshold=2.5, loss threshold=1.5, normalized, showing amplifications

    dataMatrix_2.5_1.5_unnormalized_amp.RData
    #gain threshold=2.5, loss threshold=1.5, unnormalized, showing amplifications

    dataMatrix_2.5_1.5_normalized_del.RData
    #gain threshold=2.5, loss threshold=1.5, normalized, showing deletions

    dataMatrix_2.5_1.5_unnormalized_del.RData
    # gain threshold=2.5, loss threshold=1.5, unnormalized,showing deletions

    dataMatrix_2.25_1.75_normalized_sum.RData
    dataMatrix_2.25_1.75_unnormalized_sum.RData
    dataMatrix_2.25_1.75_normalized_amp.RData
    dataMatrix_2.25_1.75_unnormalized_amp.RData
    dataMatrix_2.25_1.75_normalized_del.RData
    dataMatrix_2.25_1.75_unnormalized_del.RData
    dataMatrix_2.1_1.9_normalized_sum.RData
    dataMatrix_2.1_1.9_unnormalized_sum.RData
    dataMatrix_2.1_1.9_normalized_amp.RData
    dataMatrix_2.1_1.9_unnormalized_amp.RData
    dataMatrix_2.1_1.9_normalized_del.RData
    dataMatrix_2.1_1.9_unnormalized_del.RData
```

Alternatively, if a user is interested in running Chromosome CIN for only one setting, it can be done as shown below. This code runs Chromosome CIN for a treshold gain of 2.25, threshold loss of 1.75, unnormalized (indicated by V.def=3), showing overall (gains and losses) CIN (indicated by V.mode="sum')

```
run.cin.chr(grl.seg =  grl.data, thr.gain=2.25, thr.loss=1.75, V.def=3, V.mode="sum")
```

3.2 Run Cytoband CIN

The second step is to call the function that will calculate CIN at the cytoband level, shown below.

```
run.cin.cyto(grl.seg = grl.data, cnvgr=cnvgr.18.auto, snpgr=snpgr.18.auto,
             genome.ucsc = hg18.ucsctrack)
```

This command calculates CIN at cytoband level for each of the 18 thresholds. Due to this, it may take several minutes to complete execution. Thank you for your patience.

Similar to the algorithm for chromosome above, the algorithm for cytobands CIN outputs:

- 18 files starting with "dataMatrix." inside folder "output_cyto" (folder name is customizable). Each file contains an object "dataMatrix" that contains CIN values saved as a matrix.
- 18 files starting with "cytobands.cin" inside folder "output_cyto". (folder name is customizable). Each file contains an object the same CIN values as above, saved as a group of 22 lists (one list for each chromosome)

Alternatively, if a user is interested in running Cytoband CIN for only one setting, it can be done as shown below. In this example, we only run cytoband CIN on copy number probes in chromosome 22, treshold gain of 2.25, threshold loss of 1.75, unnormalized (indicated by V.def=3), showing overall (gains and losses) CIN (indicated by V.mode="sum')

```
run.cin.cyto(grl.seg = grl.data, cnvgr=cnvgr.18.auto, snpgr=snpgr.18.auto,
             genome.ucsc = hg18.ucsctrack,thr.gain = 2.25,thr.loss = 1.75,
             V.def = 3, V.mode="sum")
```

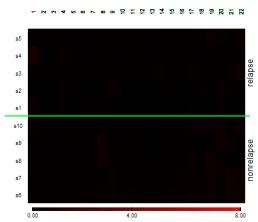
3.3 Call function to plot heatmap for chromosome level results

NOTE: The purpose of generating these many output files with all combination of settings, is that this allows the user to pick the threshold/setting that show the best contrast between two groups of interest. This can be done by visual check of the heatmaps. In this section, we describe how to plot heatmaps for each setting

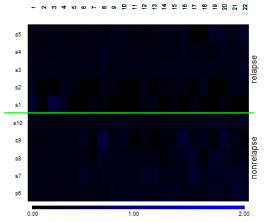
Calling function to plot heatmap for chromosome level results:

```
comp.heatmap(R_or_C="Regular", clinical.inf=clin.crc, genome.ucsc=hg18.ucsctrack,
             in.folder.name="output_chr_cin", out.folder.name="output_chr_plots")
```

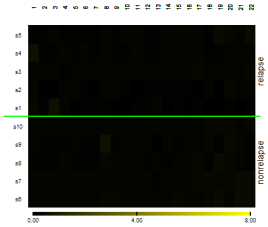
This function uses the CIN objects created in the folder "output_chr_cin" and creates a total of 18 images in a folder "output_chr_plots" (folder name customizable), one for each setting, showing the instability index for each chromosome. All the images generated also get consolidated into one pdf file, as shown in Figure 1.



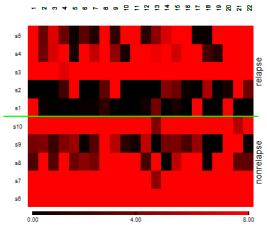
output_chr_plots/dataMatrix_2.1_1.9_normalized_amp



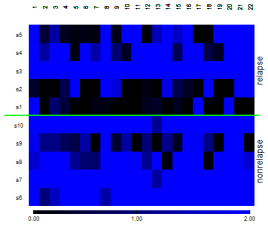
output_chr_plots/dataMatrix_2.1_1.9_normalized_del



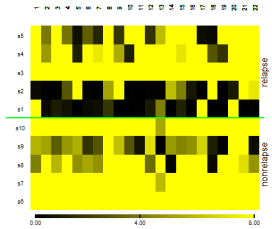
output_chr_plots/dataMatrix_2.1_1.9_normalized_sum



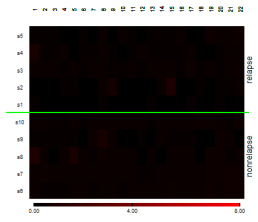
output_chr_plots/dataMatrix_2.1_1.9_unnormalized_amp



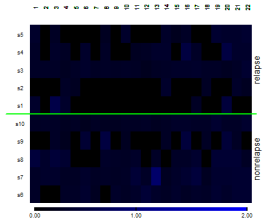
output_chr_plots/dataMatrix_2.1_1.9_unnormalized_del



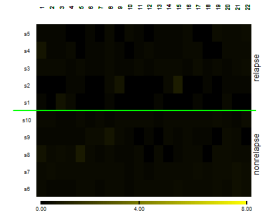
output_chr_plots/dataMatrix_2.1_1.9_unnormalized_sum



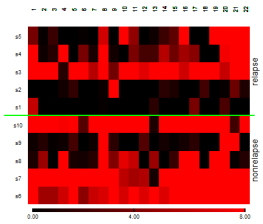
output_chr_plots/dataMatrix_2.25_1.75_normalized_amp



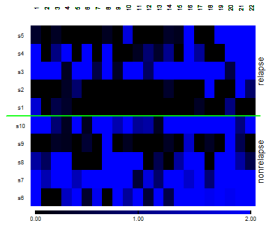
output_chr_plots/dataMatrix_2.25_1.75_normalized_del



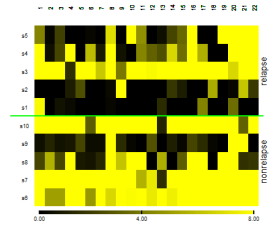
output_chr_plots/dataMatrix_2.25_1.75_normalized_sum



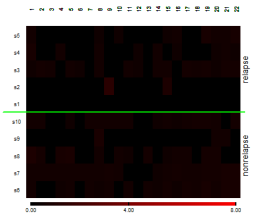
output_chr_plots/dataMatrix_2.25_1.75_unnormalized_amp



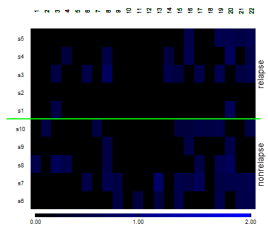
output_chr_plots/dataMatrix_2.25_1.75_unnormalized_del



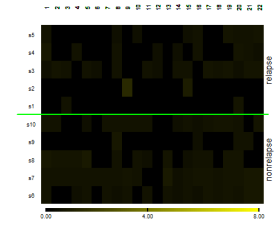
output_chr_plots/dataMatrix_2.25_1.75_unnormalized_sum



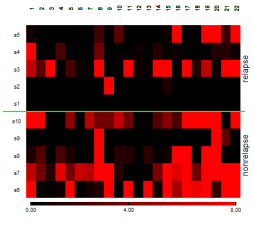
output_chr_plots/dataMatrix_2.5_1.5_normalized_amp



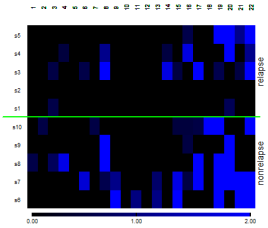
output_chr_plots/dataMatrix_2.5_1.5_normalized_del



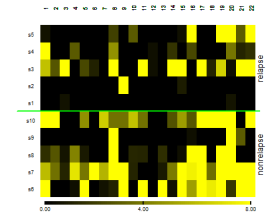
output_chr_plots/dataMatrix_2.5_1.5_normalized_sum



output_chr_plots/dataMatrix_2.5_1.5_unnormalized_amp



output_chr_plots/dataMatrix_2.5_1.5_unnormalized_del



output_chr_plots/dataMatrix_2.5_1.5_unnormalized_sum

Figure 1: Chromosome CIN overview for all settings

In the image, the color gradient reflects the value of the CIN index. Instability in terms of amplifications are shown in red color, losses are shown in blue color. Overall instability (referred to as "sum") shows both losses and gains in yellow color. **Black/white color refers to no instability, and brighter the color, the higher the instability.**

Alternatively, if a user is interested in plotting heatmap for only one setting, it can be done as shown below. This code will plot chromosome level heatmap for a treshold gain of 2.25, threshold loss of 1.75, unnormalized (indicated by V.def=3), showing overall (gains and losses) CIN (indicated by V.mode="sum')

```
comp.heatmap(R_or_C="Regular", clinical.inf=clin.crc, genome.ucsc=hg18.ucsctrack,
             thr.gain = 2.25, thr.loss = 1.75,V.def = 3,V.mode = "sum")
```

3.4 Calling function to plot heatmap for cytoband level results

Calling function to plot heatmap for cytoband level results:

```
comp.heatmap(R_or_C="Cytobands", clinical.inf=clin.crc, genome.ucsc=hg18.ucsctrack,
             in.folder.name="output_cyto_cin", out.folder.name="output_cyto_plots")
```

For each setting, this function loads the CIN objects from "output_cyto_cin" and creates cytoband level heatmaps in "output_cyto_plots" (folder name customizable). Within the output folder, it creates one folder for each setting where it saves an image for each chromosome (showing cytoband level information). In this example, we have data on 22 chromosomes.

Alternatively, if a user is interested in plotting heatmap for only one setting, it can be done as shown below. This code will plot cytoband level heatmap for a treshold gain of 2.25, threshold loss of 1.75, unnormalized (indicated by V.def=3), showing overall (gains and losses) CIN (indicated by V.mode="sum')

```
comp.heatmap(R_or_C="Cytobands", clinical.inf=clin.crc, genome.ucsc=hg18.ucsctrack,
             in.folder.name="output_cyto_cin", out.folder.name="output_cyto_plots",
             thr.gain=2.25, thr.loss=1.75,V.def=3,V.mode='sum')
```

An example image for (gainTreshold=2.25, lossThreshold=1.75, unnormalized) for chromosome 20 is shown in Figure 2. In Figure 2, you can see the non-relapse group having more yellow colored bands compared to the relapse group. As mentioned previously, the yellow color indicates overall chromosome instability in terms of losses and gains.

3.5 Selecting appropriate gain and loss thresholds

After the heatmap images are generated at the chromosome and cytoband level, it is up to the user to examine them and select the best setting (of gain treshold, loss treshold, normalized/un-normalized) in his/her opinion.

For example, let us examine the chromosome level plots in Figure 1. For an easy start, we can look at the "sum' plots as it gives an overall picture of CIN index. From this examination, you can see that the normalized plots for all settings are nearly all-black (or all-white); and the images for "gainTreshold=2.1,lossThreshold=1.9, unnormalized" seem too bright colored, which would make it hard to glean any information from it. So we reject these settings.

Ideally, the best setting would have enough black/white and colored areas allowing users to easily contrast two groups of interest and extract meaningful information. Based on this, the images for the settings(gainTreshold=2.5,lossThreshold=1.5, unnormalized) and (gainTreshold=2.25, lossThreshold=1.75, unnormalized) look appropriate and we will short list them.

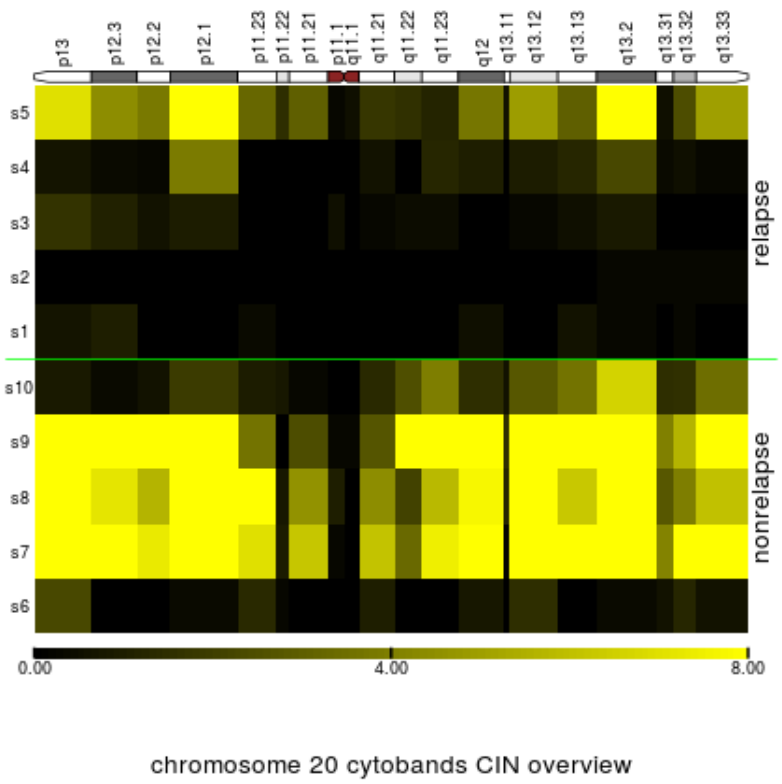


Figure 2: Cytoband CIN heatmap for chromosome 20

Let us now examine the cytoband level plots. Comparing the "sum' plots for the short listed settings and applying the same rule of thumb as above, we select this setting: (gainTreshold=2.25, lossThreshold=1.75, unnormalized).

NOTE: The documentation for the package essentially ends here. In the following sections, we show one possible way of downstream analysis using the CIN results.

3.6 Perform T test on cytoband level data to find differentially changed cytobands

Once we have selected the setting for the CIN calculation, we can now perform T test at cytoband level. The user will have to provide the CIN object obtained from previous step for the appropriate setting as an input to this section. The function will use this data and perform T test at cytoband level data to find cytobands that have statistically significant differential change in CIN index.

Let us first load the CIN object from the chosen setting into the work-space. Once loaded, this object will be seen with the name "cytoband.cin" in the environment.

```
load('output_cyto_cin/cytobands.cin_2.25_1.75_unnormalized_sum.RData')
```

Call function to run T- test.

```
ttest.cyto.cin.heatmap(cytobands.cin.obj = cytobands.cin,
                       clinical.inf = clin.crc, genome.ucsc = hg18.ucsctrack,
                       file.ext='unnormalized')
```

This function will output the following files in an output folder "output_ttest" (the folder name is customizable):

- A csv file with the output of the T test showing all cytobands and its p-value. Users will be able to open this file and sort based on p-value to find the top differentially changed cytobands (eg. ttest.result.cytobands.cin.2.25_1.75_unnormalized.csv)
- A pdf file that shows a heatmap between the two outcome groups for the top differentially changed cytobands with p-value less than 0.05 . (eg. CIN relapse VS nonrelapse for 2.25_1.75 _unnormalized _dendrogram.pdf). The heatmap shown in Figure 3 shows a nice separation between the two relapse and non-relapse groups.
- A csv file that has the underlying cytoband CIN values used in the heatmap. This file essentially contains the cytoband CIN values for the top differentially changed cytobands with p-value less than 0.05. (eg. cyto.cin4heatmap.2.25._1.75_unnormalized.csv)
- An RData object version of the csv file listed above containing CIN values used in the heatmap (eg. cyto.cin4heatmap.2.25_1.75_unnormalized.RData)

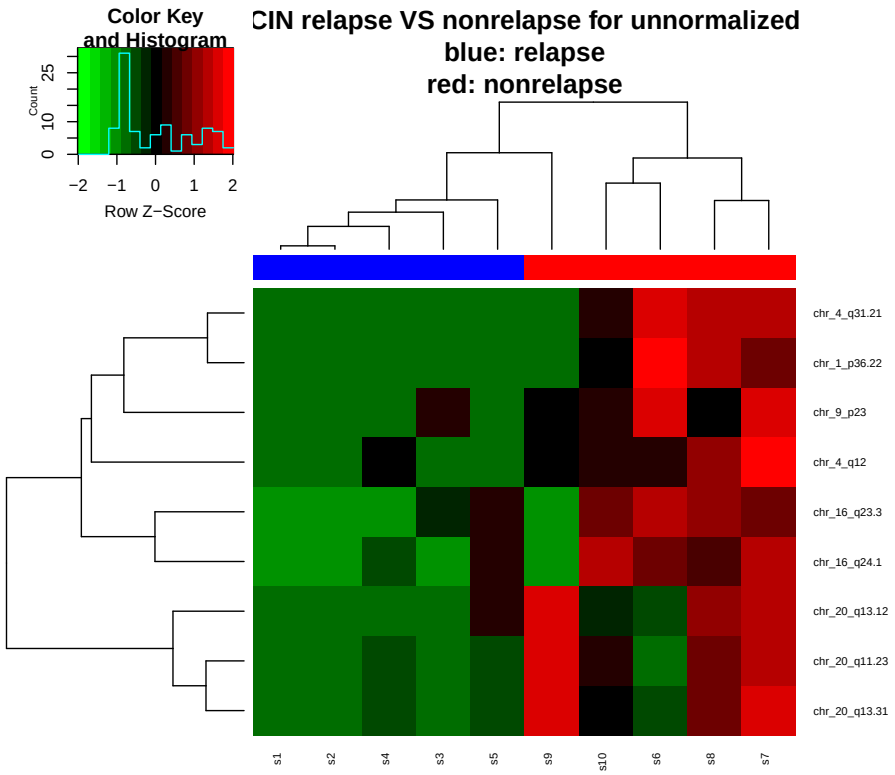


Figure 3: Heatmap showing statistically significant differentially changed cytobands between two groups

3.7 Calling function to find genes present in the cytoband regions

Once we get the list of differentially changed cytobands, it would be interesting to see which genes fall in these cytoband regions.

To be able to use this function, a CDS gene annotation file is required. A sample gene annotation file is provided (detailed explanation on how this file was created is present in the file titled "Prepare input data for CINdex"). Let us load the file.

```
#Loading the data into the workspace
data("geneAnno")
```

Load heatmap object created in the above step (T-test output) into the workspace.

```
#Loading the heatmap object into the workspace
load('output_ttest/cyto.cin4heatmap.unnormalized.RData')
#object loaded is called "cyto.cin4heatmap"
```

Calling the function:

```
extract.genes.in.cyto.regions(cyto.cin4heatmapObj = cyto.cin4heatmap,
                              genome.ucsc=hg18.ucsctrack, gene.annotations = geneAnno)
```

This function will output the following files in an output folder "output_genename" (the folder name is customizable):

- A csv file "cytoband_genes.csv" that contains gene symbols and the cytoband name where it is located
- An RData object of the above file "cytoband_genes.RData"
- A csv file "out.genes.csv" that contains the same gene symbols as above, but also includes the CDS start and end position

3.8 Performing pathway enrichment on gene list

In this last step of the workflow, we take the gene symbols obtained in the previous step and perform pathway enrichment using the Reactome database within Bioconductor.

Let us first load the cytoband_genes.Rdata object into the workspace

```
load("output_genename/cytoband_genes.Rdata")
```

Consider only unique gene symbols for the next step

```
inputGenes <- unique(cytoband_genes[,"geneNames"])
```

Convert gene symbols to Entrez Gene id

```
#Loading the package
#source("http://bioconductor.org/biocLite.R")
#biocLite("org.Hs.eg.db")
library(org.Hs.eg.db)
######
entrez <- mapIds(x = org.Hs.eg.db,keys = inputGenes, keytype = "SYMBOL",
                 column = "ENTREZID")
entrez <- na.omit(entrez)
```

We now use the ReactomePA Bioconductor package for pathway enrichment.

```
#Loading the package
#biocLite("ReactomePA")
library(ReactomePA)

#Calling function to perform enrichment
enrich <- enrichPathway(gene = entrez, pvalueCutoff = 0.05, readable = TRUE)
#Examining and saving the results
head(enrich@result,5)
#viewing top pathway names
head(enrich@result$Description,15)
#viewing top pathway names and its p-value
head(cbind(enrich@result$Description,enrich@result$pvalue),15)
#results not displayed in this document due to its large size
```

```
#saving results into a file for easy viewing
write.csv(enrich@result, "enrichment.csv",row.names = FALSE)
```

Viewing top pathways as a bar plot. Figure 4 shows the results of pathway enrichment.

```
#View results as a bar plot
barplot(enrich, showCategory=8)
```

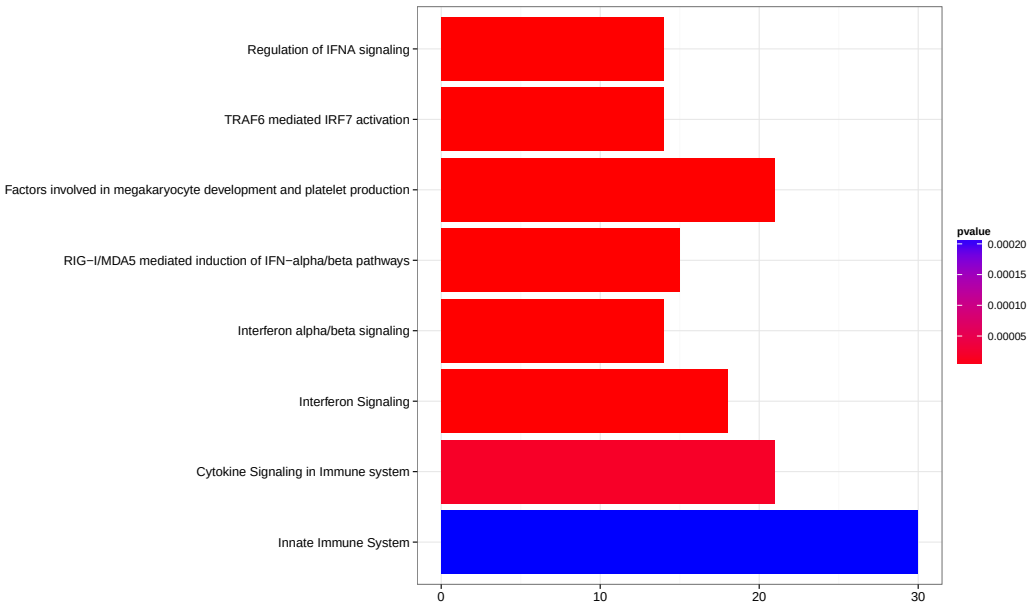


Figure 4: Top pathways enriched

4 Conclusion

We hope this example workflow has been useful in doing an end to end workflow starting with CIN calculation, plotting the CIN heatmaps, identification of the appropriate CIN settings and downstream analysis.