Seattle, Bioconductor Workshop, Feb 29th 2012

Aedin Culhane

February 28, 2012

1 Exercise 1

Coinertia analysis has been applied to the cross-platform comparison (meta-analysis) of microarray gene expression datasets [3]. CIA is a multivariate method that identifies trends or co-relationships in multiple datasets which contain the same samples. That is either the rows or the columns of a matrix must be "matchable". CIA can be applied to datasets where the number of variables (genes) far exceeds the number of samples (arrays) such is the case with microarray analyses. `cia` calls `coinertia` in the R package *ade4*.

Lets examine two gene expression datasets of the same 60 cell lines. The NCI60 cells lines are a set of 60 cell lines with different tumour phenotypes (eg Breast, Colon, Leukemia, Prostate, CNS, lung cancer, ovarian, renal cancer etc). The gene expression of these cell lines have been examined by a number of groups [4],[5].

The same 60 cell lines were analysed by different labs on differnt microarray platforms. We will compare one from Affymetrix (Staunton et al., 2001) and one that was obtained using Stanford spotted cDNA arrays (Ross et al., 2000) using `cia`. These 2 datasets were analyzed using `cia` by Culhane et al., 2003 [3].

These 2 datasets are available in the *made4* data object *NCI60*. The Ross dataset contains 1375 genes, and the affy dataset contains 1517. There is little overlap betwen the genes represented on these platforms. CIA allows visualisation of genes with similar expression patterns across platforms.

1.1 Correspondence Analysis

As CIA uses non symetric correspondence analysis in its calculation, lets first look at the results of NSC.

The function `ord` simplifies the running of ordination methods such as principal component, correspondence or non-symmetric correspondence analysis. It provides a wrapper which can call each of these methods. To run a correspondence analysis [1] on this dataset.

```
> require(made4)
> data(NCI60)
> summary(NCI60)
```

```
        Length Class      Mode
Ross     60    data.frame list
Affy     60    data.frame list
classes 120    -none-     character
Annot     4    data.frame list
```

```
> names(NCI60)
```

```
[1] "Ross"    "Affy"    "classes" "Annot"
```

```
> NCI60$classes[1:3,]
```

```
     Sample          Class
[1,] "BREAST_BT549"  "BREAST"
[2,] "BREAST_HS578T" "BREAST"
[3,] "BREAST_MCF7"   "BREAST"
```

```
> table(NCI60$classes[,2])

  BREAST      CNS    COLON     LEUK    MELAN    NSCLC
       8        6        7        6        8        9
    OVAR PROSTATE    RENAL
       6        2        8
```

```
> data.coa <-ord(NCI60$Ross, type="nsc")
```

Have a look at data.coa. The ordination results are in $ord. The row, column coordinates are $li and $co respectively. The eigenvalues are in $eig.

```
> data.coa$ord
```

```
Duality diagramm
class: nsc dudi
$call: dudi.nsc(df = data.tr, scannf = FALSE, nf = ord.nf)

$nf: 59 axis-components saved
$rank: 59
eigen values: 0.007109 0.004584 0.003671 0.002675 0.001829 ...
  vector length mode    content
1 $cw    60     numeric column weights
2 $lw    144    numeric row weights
3 $eig   59     numeric eigen values

  data.frame nrow ncol content
1 $tab       144  60   modified array
2 $li        144  59   row coordinates
3 $l1        144  59   row normed scores
4 $co        60   59   column coordinates
5 $c1        60   59   column normed scores
other elements: N
```

```
> data.coa$ord$co[1:2,1:3]

                   Comp1       Comp2       Comp3
BREAST_BT549  -0.1027503 -0.05317834 -0.03500724
BREAST_HS578T -0.1810595 -0.09919005 -0.03431942
```

In a COA analysis the total $eig will be equivalent to the total chi-sq of the table. To get the % of variance explained by each axis.

```
> data.coa$ord$eig*100/sum(data.coa$ord$eig)
```

```
 [1] 17.91785574 11.55239173  9.25136783  6.74070884
 [5]  4.60913722  4.27981008  3.71199288  3.37734886
 [9]  3.05385196  2.69593057  2.45194969  2.09441162
```

```
[13]  1.92911407  1.80687648  1.69231686  1.63652062
[17]  1.58822127  1.41338463  1.30014741  1.24317392
[21]  1.16263483  1.07582551  1.04825608  0.93870215
[25]  0.83803403  0.81431918  0.78425986  0.70268453
[29]  0.68077905  0.64506503  0.59717333  0.51091894
[33]  0.49257055  0.46244680  0.42976922  0.39696042
[37]  0.35951726  0.33868435  0.31112555  0.29291998
[41]  0.26716341  0.25724510  0.24508236  0.21441937
[45]  0.20487311  0.20177249  0.18092929  0.16961007
[49]  0.15406772  0.14811757  0.12143088  0.11132035
[53]  0.11016981  0.09184747  0.08661265  0.08011274
[57]  0.05875530  0.03962683  0.02768457
```

The cumulative variance is given by

```
> cumsum(data.coa$ord$eig*100/sum(data.coa$ord$eig))
```

```
 [1]  17.91786  29.47025  38.72162  45.46232  50.07146
 [6]  54.35127  58.06326  61.44061  64.49447  67.19040
[11]  69.64235  71.73676  73.66587  75.47275  77.16506
[16]  78.80159  80.38981  81.80319  83.10334  84.34651
[21]  85.50915  86.58497  87.63323  88.57193  89.40996
[26]  90.22428  91.00854  91.71123  92.39201  93.03707
[31]  93.63425  94.14516  94.63774  95.10018  95.52995
[36]  95.92691  96.28643  96.62511  96.93624  97.22916
[41]  97.49632  97.75357  97.99865  98.21307  98.41794
[46]  98.61971  98.80064  98.97025  99.12432  99.27244
[51]  99.39387  99.50519  99.61536  99.70721  99.79382
[56]  99.87393  99.93269  99.97232 100.00000
```

Therefore almost 57% of the variance is captured by the first 2 components.

1.2 Correspondence Analysis- Visualization of Results

There are many functions in *made4* for visualizing results from ordination analysis. The simplest way to view results from `ord` is to use the function `plot`. This will draw a plot of the eigenvalues, along with plots of the variables (genes) and a plot of the cases (microarray samples).

1.3 Q1

Perform the same analysis, but on the Staunton Dataset

1.4 CIA

Now lets perform a CIA to visualize the relationship between the 2 datasets

```
> data(NCI60)
> coin <- cia(NCI60$Ross, NCI60$Affy)
> names(coin)
```

```
[1] "call"      "coinertia" "coa1"      "coa2"
```

```
> coin$coinertia
```

```
> plot(data.coa)
```

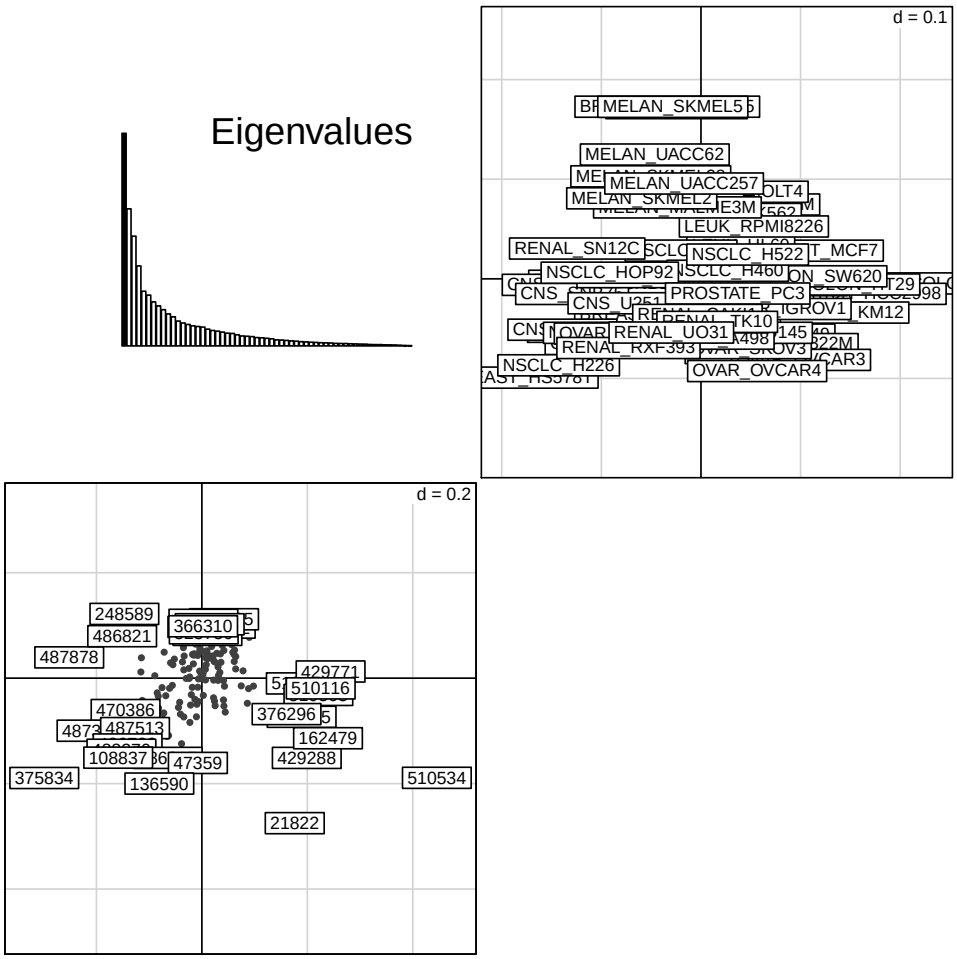


Figure 1: Correspondence analysis plot. A. plot of the eigenvalues, B. projection of microarray samples (colored by Donor) C. projection of genes (gray filled circles) and D. biplot showing both genes and samples. Samples and genes with a strong associated are projected in the same direction from the origin. The greater the distance from the origin the stronger the association

```
Coinertia analysis
call: coinertia(dudiX = t.dudi(coa1), dudiY = t.dudi(coa2), scannf = cia.scan,
    nf = cia.nf)
class: coinertia dudi

$rank (rank)     : 59
$nf (axis saved) : 2
$RV (RV coeff)   : 0.7859656

eigen values: 2.266e-05 9.904e-06 4.342e-06 2.335e-06 1.576e-06 ...

  vector length mode    content
1 $eig   59     numeric eigen values
2 $lw    144    numeric row weigths (crossed array)
3 $cw    144    numeric col weigths (crossed array)

   data.frame nrow ncol content
1  $tab       144  144  crossed array (CA)
2  $li        144  2    Y col = CA row: coordinates
3  $l1        144  2    Y col = CA row: normed scores
4  $co        144  2    X col = CA column: coordinates
5  $c1        144  2    X col = CA column: normed scores
6  $lX        60   2    row coordinates (X)
7  $mX        60   2    normed row scores (X)
8  $lY        60   2    row coordinates (Y)
9  $mY        60   2    normed row scores (Y)
10 $aX        2    2    axis onto co-inertia axis (X)
11 $aY        2    2    axis onto co-inertia axis (Y)
```

The RV coefficient $RV which is 0.786 in this instance, is a measure of "global" similarity between the datasets. The closer to 1, in the scale 0-1 the greater the correlation between the two datasets.

```
> coin$coinertia$RV

[1] 0.7859656
```

To visually examine the cell lines that have similar or different gene expression profiles in these datasets, use `plotarrays`.

```
> plotarrays(coin, classvec=NCI60$classes[,2],
+             lab="", cpoint=3)
```

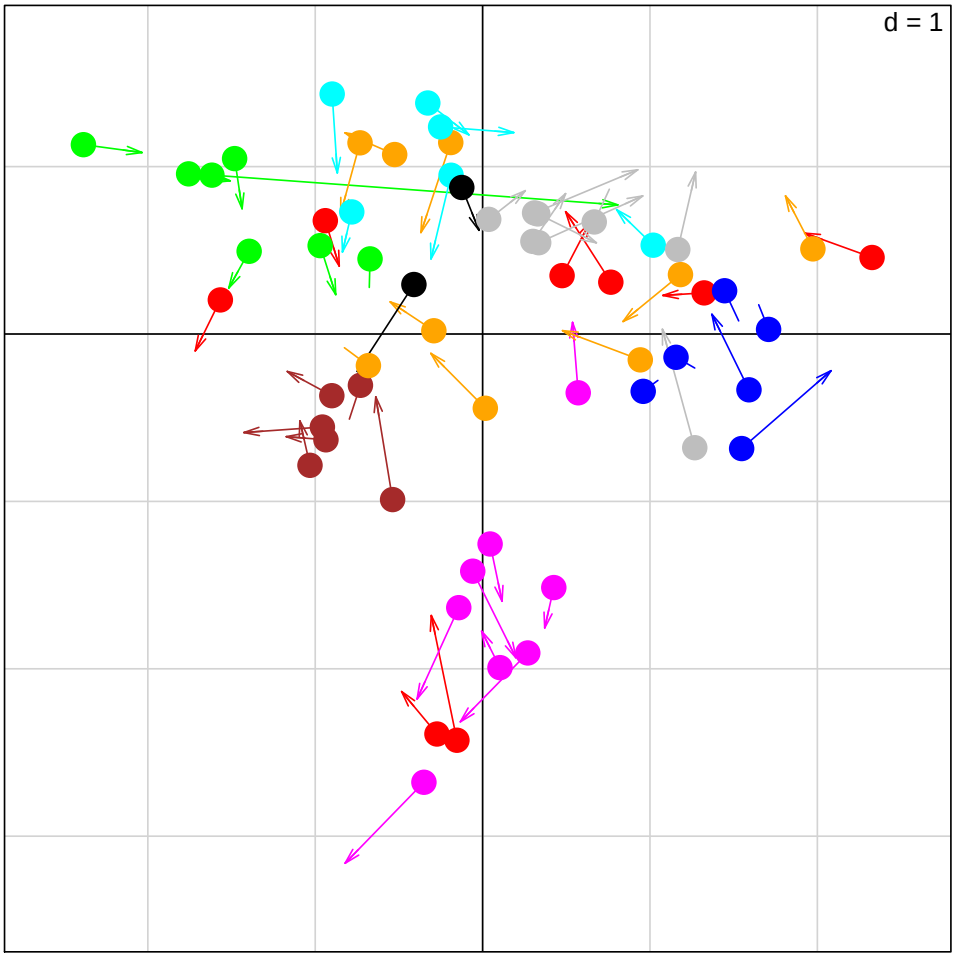


Figure 2: Coinertia analysis of NCI 60 cell line Spotted and Affymetrix gene expression dataset. Each cell lines is colored by its phenotype (eg colon are green,breast are red, melanoma are pink etc). For each of the 60 cell lines, there are two coordinates ($coinertia$mX and $coinertia$mY). On the plot, these are visually shown as a closed circle and an arrow. These are joined by a line. If the profiles are similar they will be projected close together in the new space (ie joined by a short line). For more information see Culhane et al., BMC bioinformatics 2003.

If `plot` is used, the above plot together with the plots of the gene projections from each dataset can be visualized.

Coinertia analysis be applied to other types of data including the integration of gene expression and transcription factor binding site data [6] or to the analysis of gene and protein expression data [7].

```
> plot(coin, classvec=NCI60$classes[,2])
```

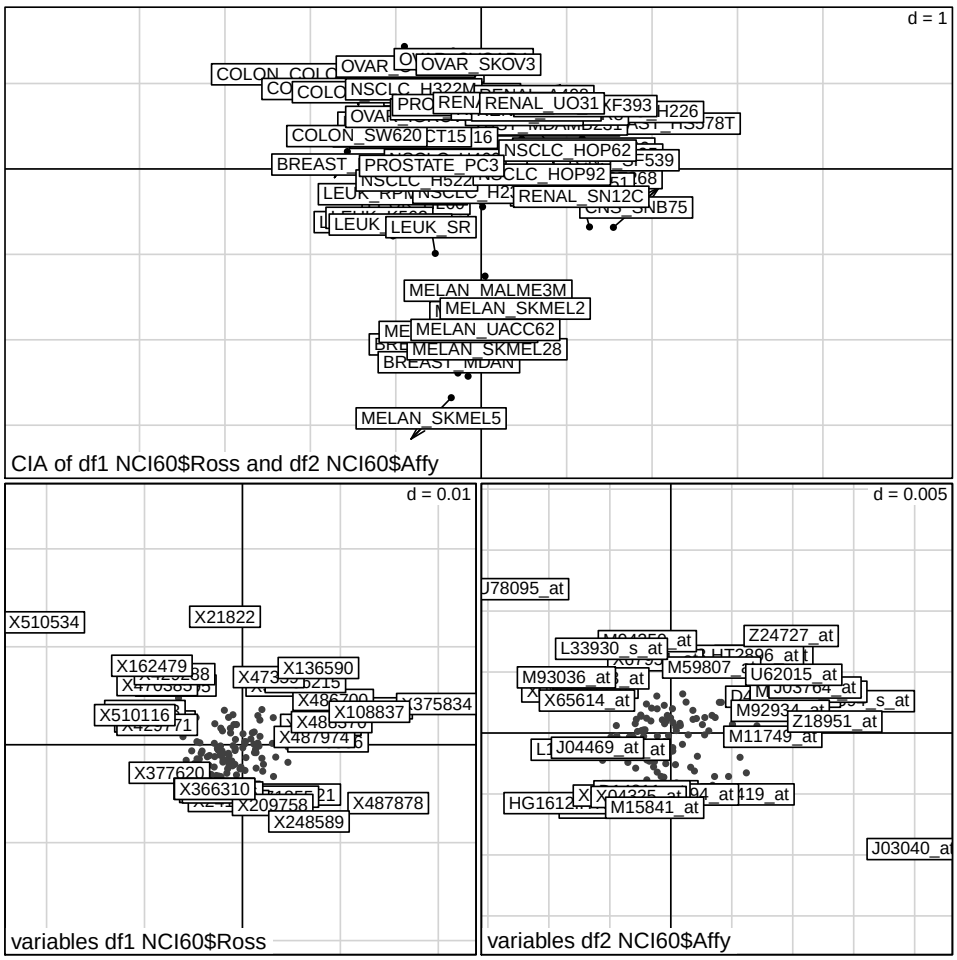


Figure 3: Coinertia analysis of NCI 60 cell line Spotted and Affymetrix gene expression dataset. A) shows a plot of the 60 microarray samples projected onto the one space. The 60 circles represent dataset 1 (Ross) and the 60 arrows represent dataset 2 (affy). Each circle and arrow are joined by a line, the length of which is proportional to the divergence between that samples in the two datasets. The samples are coloured by cell type. B)The gene projections from datasets 1 (Ross), C) the gene projections from dataset 2 (Affy). Genes and samples projected in the same direction from the origin show genes that are expressed in those samples.

2 Exercise 2: Gene Sets

Gene Set Enrichment Analysis uses two different types of data: a gene expression dataset and a list of gene sets. The gene expression dataset is provided to you in the data package breastCancerVDX file.

In this example we use GeneSets from GeneSigDB, but I enclose the code to use those from MSigDB, which can be downloaded from MSigDB（http://www.broadinstitute.org/gsea/downloads.jsp).

```
> Burl<-"http://www.broadinstitute.org/gsea/msigdb/download_file.jsp?
+ filePath=/resources/msigdb/3.0/msigdb_v3.0.xml"
> gsc <- getBroadSets(Burl)
> types <- sapply(gsc, function(elt) bcCategory(collectionType(elt)))
> ## C2 contains the curated Gene Sets
> c2gsc1 <- gsc[types == "c2"]

> #download.file("http://bcb.dfci.harvard.edu/~aedin/
> #courses/Seattle2012/GeneSigDB_GS.RData", "GeneSigDB.RData")
> #load("GeneSigDB.RData")
>
> library(GeneSigDB)
> data(genesigdbEntrez)
> length(genesigdbEntrez)

[1] 2951

> genesigdbEntrez[[1]]

setName: Lymphoma_Golub99_50genes
geneIds: 26999, 6929, ..., 4067 (total: 48)
geneIdType: EntrezId (org.Hs.eg.db)
collectionType: Null
details: use 'details(object)'

> names(genesigdbEntrez)<-sapply(genesigdbEntrez, setName)
```

We then load the Veridex breast cancer dataset

```
> ## load data package
> require("breastCancerVDX")
> ## load the dataset
> data(vdx)

> ## GSA using camera in limma package
> require(limma)
> ## first create a 1,0 index for each genelist
> Gind<-lapply(genesigdbEntrez[1:100], function(x) fData(vdx)$EntrezGene.ID%in%geneIds
> GSAres<-camera(Gind, vdx, model.matrix(~vdx$er))
> GSAres[1:5,]

                                     NGenes Correlation
Lymphoma_Golub99_50genes                100  0.02479444
Breast_Sgroi99_16genes                   18  0.01901641
Breast_Rinehart-Kim00_21genes_MCF7cells  30  0.09690196
Colon_Ben-Dor00_93genes                 143  0.01790732
```

```
Leukemia_Ben-Dor00_143genes                243  0.03835053
                                              Down
Lymphoma_Golub99_50genes                0.04747673
Breast_Sgroi99_16genes                  0.65981212
Breast_Rinehart-Kim00_21genes_MCF7cells 0.53335955
Colon_Ben-Dor00_93genes                 0.04565837
Leukemia_Ben-Dor00_143genes             0.05871325
                                               Up
Lymphoma_Golub99_50genes                0.9525233
Breast_Sgroi99_16genes                  0.3401879
Breast_Rinehart-Kim00_21genes_MCF7cells 0.4666404
Colon_Ben-Dor00_93genes                 0.9543416
Leukemia_Ben-Dor00_143genes             0.9412868
                                          TwoSided
Lymphoma_Golub99_50genes                0.09495346
Breast_Sgroi99_16genes                  0.68037577
Breast_Rinehart-Kim00_21genes_MCF7cells 0.93328089
Colon_Ben-Dor00_93genes                 0.09131675
Leukemia_Ben-Dor00_143genes             0.11742649
```

3 Gene Set Enrichment Analysis

Or we get the GSEAlm package from Bioconductor and load it into R:

```
> require(GSEAlm)
```

Now we run the GSEAlm using the gene expression dataset and the gene sets. We also have to specify the column of the clinical information data.frame we want to use and the number of permutations that are used to estimate the null distribution of the enrichment scores. In this case we use 1000 permutations, which can take a while to run. If this takes too long, try 100 instead of 1000:

```
> Gind<-t(sapply(genesigdbEntrez[1:100], function(x)fData(vdx)$EntrezGene.ID%in%geneIds
> pVals=gsealmPerm(vdx, ~er, Gind,nperm=100)
> head(pVals)
```

Since we test 825 gene sets at the same time we have to correct for multiple testing. In this case we use the method from Benjamini-Hochberg:

```
> pVals<-apply(pVals,2,p.adjust,method='BH',n=nrow(pVals))
> head(pVals)
```

To show the results we set a significance threshold:

```
> THRESHOLD<-0.05
```

And finally, the first 20 gene sets that are down regulated in Non-BLC breast cancer:

```
> downRegulated<-data.frame(sort(pVals[pVals[,1]<THRESHOLD,1])[1:20])
> names(downRegulated)<-'lower'
> downRegulated
```

and the gene sets that are up regulated in Non-BLC breast cancer:

```
> upRegulated<-data.frame(sort(pVals[pVals[,2]<THRESHOLD,2]))
> names(upRegulated)<-'upper'
> upRegulated
```

4 iBBiG

To demonstrate iBBiG, we will use a simulated binaru dataset of 400 rows x 400 columns, in 1 indicates a positive association (or p<0.05).

```
> library(iBBiG)
> binMat<-makeArtificial()
```

```
[1] "***** Summary of Design Matrix ******"
   Rows Cols DensityLow DenistyHigh
M1  250   25        0.4         0.9
M2   75  175        0.4         0.8
M3   50   50        0.5         0.8
M4   40   40        0.4         0.9
M5   30   30        0.4         0.8
M6   20   20        0.6         0.9
M7   40   40        0.5         0.6

Cluster sizes in new iBBiG (Biclust) data object
Number of Modules: 7
Rows        250        75        50        40        30        20        40
Columns      25       175        50        40        30        20        40
```

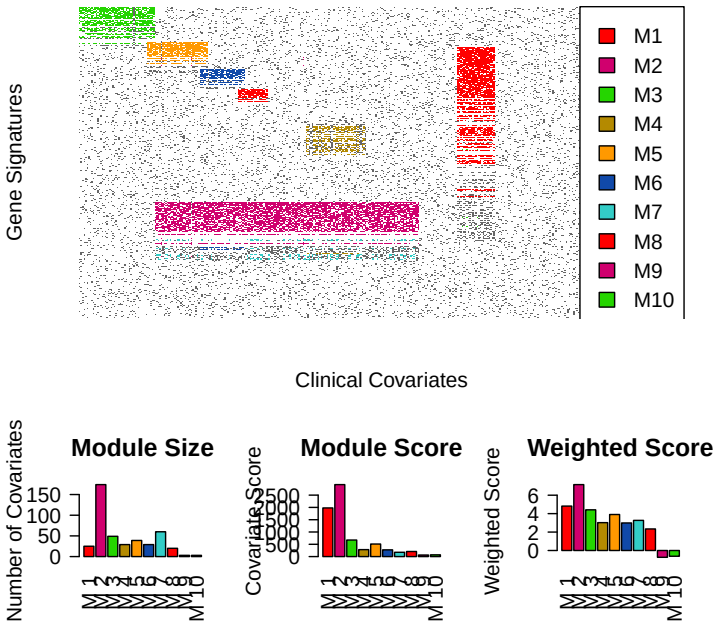
```
> #plot(binMat)
> res<- iBBiG(binMat@Seeddata, nModules=10)
```

```
Module:  1 ... done
Module:  2 ... done
Module:  3 ... done
Module:  4 ... done
Module:  5 ... done
Module:  6 ... done
Module:  7 ... done
Module:  8 ... done
Module:  9 ... done
Module:  10 ... done
```

```
> plot(res)
> statClust(res,binMat)
```

```
   Cluster    JI nRow nCol Covariate_accuracy
M1       2 1.000  129   25              1.000
M2       1 0.989   38  175              0.995
M3       3 0.980   32   49              0.998
M4       6 0.950   24   38              0.995
M5       4 0.900   19   27              0.992
M6       7 1.000   16   20              1.000
M7       5 0.675   27   27              0.968
   Covariate_sensitivity Covariate_specificity
M1                 1.000                 1.000
M2                 0.994                 0.996
M3                 0.980                 1.000
M4                 0.950                 1.000
```

```
M5                 0.900                 1.000
M6                 1.000                 1.000
M7                 0.675                 1.000
   Covariate_PPV Covariate_NPV GeneSet_accuracy
M1         1.000         1.000            0.698
M2         0.994         0.996            0.908
M3         1.000         0.997            0.955
M4         1.000         0.994            0.960
M5         1.000         0.992            0.962
M6         1.000         1.000            0.990
M7         1.000         0.965            0.952
   GeneSet_sensitivity GeneSet_specificity GeneSet_PPV
M1               0.516                1.000       1.000
M2               0.507                1.000       1.000
M3               0.640                1.000       1.000
M4               0.600                1.000       1.000
M5               0.567                0.995       0.895
M6               0.800                1.000       1.000
M7               0.600                0.992       0.889
   GeneSet_NPV
M1       0.554
M2       0.898
M3       0.951
M4       0.957
M5       0.966
M6       0.990
M7       0.957
```



```
> hclust2biClust<-function(seeddata, distMethod="binary",
+                          linkage="ward", k=8, plot=FALSE){
+   ## Rows (Signatures)
+   hcR<-hclust(dist(seeddata,method='binary'), method="ward")
+
+   ## Columns (covariates)
+   hcC <- hclust(dist(t(seeddata),method='binary'), method="ward")
+
+   hcc=cutree(hcC, k=k) # cut tree into k clusters
+   hcr=cutree(hcR, k=k)
+   if (!all(unique(hcc)==unique(hcr))) stop("error parsing")
+   cc<-unique(hcc)
+   NC<- t(sapply(cc, function(x) hcc%in%x))
+   RN<- sapply(cc, function(x) hcr%in%x)
+   nClust=length(cc)
+
+   if (inherits(seeddata, "data.frame")) {
+     seeddata<-as.matrix(seeddata)
+     new("Biclust", Parameters = list(seeddata=seeddata),
+       NumberxCol = NC, RowxNumber=RN, Number = nClust)
+   }
+ }
> hclustListk3.15<-lapply(3:15,function(x) hclust2biClust(seeddata=binMat@Parameters$se
> ## Run HC but cut at different K to determine the best K
> #names(hclustListk3.15) = paste("K", 3:15, sep="")
>
> ## Worked out 6 clusters is optimum
> # bC<-bestClust(hclustListk3.15, GS=tt)
>
>
>
> plotJI<-function(res=hclustListk3.15, GS=tt,
+                  margin="col", cols=c('#FF0000','#D0006E',
+                         '#25D500','#B48A00','#FF9900',
+                                    '#1049A9','#33CDC7')) {
+   ## Give a list of biclust results calculate the JI to another biclust object (TRUTH
+
+   HC.JI<-sapply(res, function(x) successRes(x, GS, margin=margin)[,"JI"])
+
+   meanJI<-apply(HC.JI, 2, mean)
+   if (margin=="both") yL=2 else yL=1.2
+   plot(HC.JI[1,],ylab="JI", xlab="Number of Clusters",
+        ylim=c(0,yL), pch=1, col=cols[1])
+   for (i in 2:nrow(HC.JI)) lines(HC.JI[i,], type="b",
+                                  pch=1, col=cols[i])
+   lines(meanJI, type="b", pch=15, col="black", lwd=2)
+   legend("bottomright", c(rownames(HC.JI), "mean"),
+          fil=c(cols, "black"), ncol=3)
+
+   title(main="Hierarchical Clustering: Jaccard Index",
+         sub=paste("(JI computd on ", margin, ")", sep=""),
```

```
+          cex.sub=0.7)
+ }
> plotJI()
> BC<-hclust2biClust(k=6)
> BCres<-statClust(resi=BC, GS=tt, parameters="default")
> write.table(BCres, file=paste("Hierarchical Clustering",
+                               outfile, ".csv", sep=""), sep=",")
> print(xtable(BCres, caption="Hierarchical Clustering"), file=paste("Hierarchical Clus
>
```

5 Session Info

- R Under development (unstable) (2012-02-23 r58468), `x86_64-unknown-linux-gnu`
- Locale: `C`
- Base packages: base, datasets, grDevices, graphics, grid, methods, stats, utils
- Other packages: AnnotationDbi~1.17.22, Biobase~2.15.3, BiocGenerics~0.1.7, BiocInstaller~1.3.7, GSEABase~1.17.2, GeneSigDB~1.0, KernSmooth~2.23-7, MASS~7.3-17, RColorBrewer~1.0-5, ade4~1.4-17, annotate~1.33.2, biclust~1.0.1, bitops~1.0-4.1, breastCancerVDX~1.0.2, caTools~1.12, colorspace~1.1-1, gdata~2.8.2, gplots~2.10.1, graph~1.33.0, gtools~2.6.2, iBBiG~1.0, lattice~0.20-0, limma~3.11.14, made4~1.29.0, mclust~3.4.11, scatterplot3d~0.3-33
- Loaded via a namespace (and not attached): DBI~0.2-5, IRanges~1.13.26, RSQLite~0.11.1, XML~3.9-4, tools~2.15.0, xtable~1.7-0

References

[1] Fellenberg, K., Hauser, N.C., Brors, B., Neutzner, A., Hoheisel, J.D., and Vingron, M. Correspondence analysis applied to microarray data. *Proc Natl Acad Sci U S A* **98**: 10781-10786. 2001.

[2] Thioulouse,J., Chessel,D., Dolédec,S., and Olivier,J.M ADE-4: a multivariate analysis and graphical display software. *Stat. Comput.*, **7**, 75-83. 1997.

[3] Culhane AC, Perriere G, Higgins DG. Cross platform comparison and visualisation of gene expression data using co-inertia analysis. *BMC Bioinformatics*.**4**:59. 2003.

[4] Ross DT, Scherf U, Eisen MB, Perou CM, Rees C, Spellman P, Iyer V, Jeffrey SS, Van de Rijn M, Waltham M, Pergamenschikov A, Lee JC, Lashkari D, Shalon D, Myers TG, Weinstein JN, Botstein D, Brown PO. Systematic variation in gene expression patterns in human cancer cell lines. *Nat Genet* **24**:227-235 2000,

[5] Staunton JE, Slonim DK, Coller HA, Tamayo P, Angelo MJ, Park J, Scherf U, Lee JK, Reinhold WO, Weinstein JN, Mesirov JP, Lander ES, Golub TR: Chemosensitivity prediction by transcriptional profiling. *Proc Natl Acad Sci U S A* **98**:10787-10792. 2001

[6] Jeffery IB,Madden SF,McGettigan PA,Perriere G, Culhane AC, Higgins DG Integrating transcription factor binding site information with gene expression datasets. *Bioinformatics* **23(3)**:298-305.2006.

[7] Fagan A, Culhane AC, Higgins DG. A multivariate analysis approach to the integration of proteomic and gene expression data. *Proteomics* Jun 5 2007.