Vignette 1 - Understanding the Results of MCIA

Max Mattessich Joaquin Reyna Anna Konstorum

9/1/2022

Part 1: Interpreting Global Factor Scores

The main MCIA function `nipals_multiblock()` outputs a global factor score matrix $\mathbf{F}$ that is $n \times r$, where n is the number of samples and r is the number of factors chosen by the user with the `num_PCs = r` argument. Each column of this matrix represents one of the orders of global factors computed, i.e.

$$\mathbf{F} = \begin{pmatrix} | & | & & | \\ \mathbf{f}^{(1)} & \mathbf{f}^{(2)} & \dots & \mathbf{f}^{(r)} \\ | & | & & | \end{pmatrix} \in \mathbb{R}^{n \times r}$$

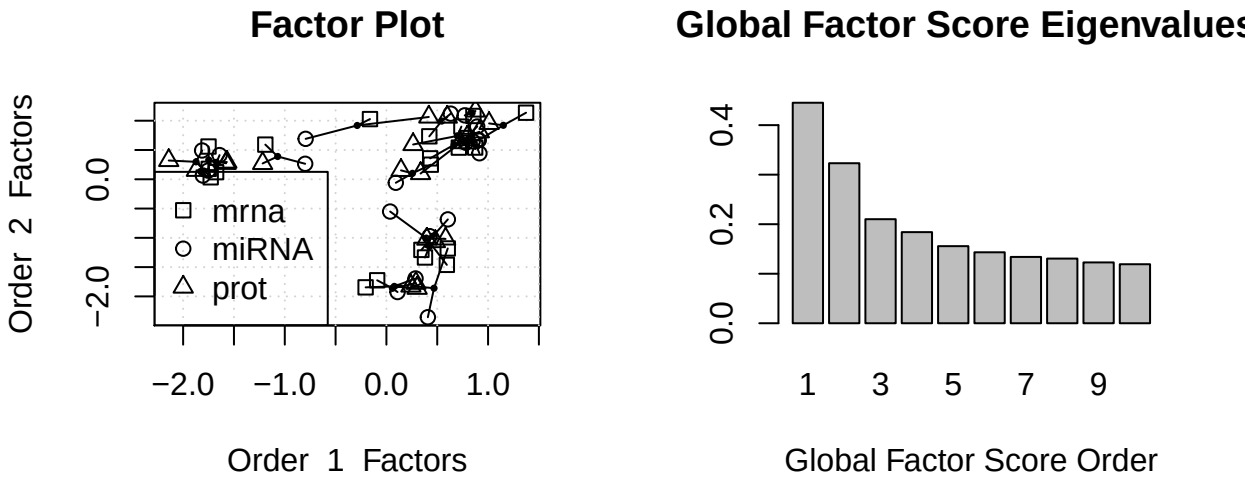
This matrix encodes a low-dimensional representation of the dataset, with the i-th row representing a set of r-dimensional coordinates for the i-th sample.

Running MCIA on NCI60 Data and Basic Visualization

`NIPALS-MCIA` includes a sample multi-omics dataset modified from data collected on the NCI-60 cancer cell lines [CITE: Meng, 2016]. This can be used to illustrate low-dimensional plotting with the global factors.

First, we compute the first 10 global factors for the dataset:

```
data(NCI60) # this creates the dataset as `data_blocks`
mcia_results <- nipals_multiblock(data_blocks, preprocMethod='colprofile', num_PCs = 10, tol=1e-12)
```



By default, `nipals_multiblock()` returns a plot of the first two global factors, with the block factors plotted as shapes connected to the global factors. If a block (in this case, one omics type) is plotted far from its corresponding global factor, this is an indication that the block does not agree with the whole-dataset trend. This may indicate batch effects in data collection, or indicate some underlying difference between blocks.

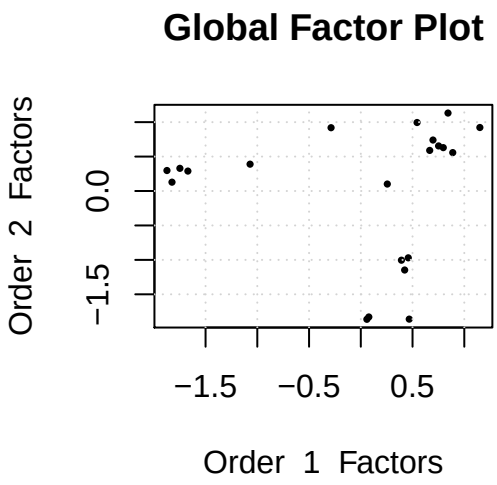
The second plot returned is a scree plot of factor singular values, where higher values indicate a given order of factor is more important. This can be interpreted exactly the same as an eigenvalue scree plot in principal

component analysis.

Visualizing only global factor scores

For clustering, it is useful to only look at global factors (Without block factors). The `MCIA_plots()` function can be used to generate this plot with the `projection_global` argument:

```
MCIA_plots(mcia_results,'projection_global',orders = c(1,2), legend_loc = "bottomleft")
```



It may be helpful to color points by some external label. The `MCIA_plots()` function can do this with a `metadata` argument. The `metadata` object is a dataframe of labels for each sample, possibly containing multiple types of external data. For instance, each of the 21 samples in the NCI-60 dataset relates to one of three cancer types: CNS, Leukemia, or Melanoma. Thus we create the metadata object with the `cancerType` column:

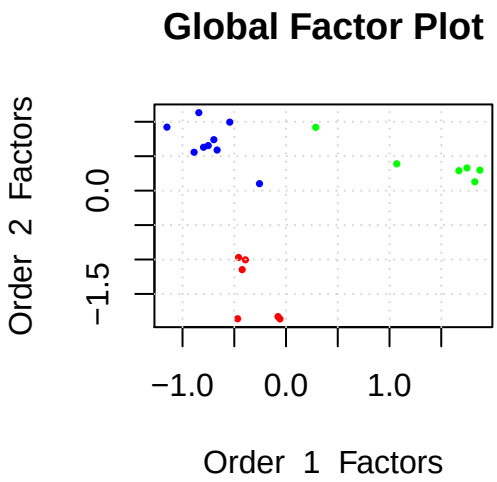
```
CNS = 1:6; LEU = 7:12; ME = 13:21;
nameslist <- list(1:21)
nameslist[CNS] <- "CNS"
nameslist[LEU] <- "Leukemia"
nameslist[ME] <- "Melanoma"
metadata_NCI60 <- data.frame(cancerType = unlist(nameslist))
row.names(metadata_NCI60) <- rownames(data_blocks[[1]])

#View(metadata_NCI60)
```

This object can be passed into the `metadata` argument of `nipals_multiblock()` or added directly to `mcia_results`.

```
mcia_results <- nipals_multiblock(data_blocks, preprocMethod='colprofile',
                                  metadata = metadata_NCI60,
                                  plots = 'none',num_PCs = 10, tol=1e-12)
# Alternative method for adding metadata:
mcia_results$metadata <- metadata_NCI60
```

The `coloring` argument of `MCIA_plots()` can be used to determine which column of `metadata` is used for coloring the projection plots. In this case the column is `cancerType`:

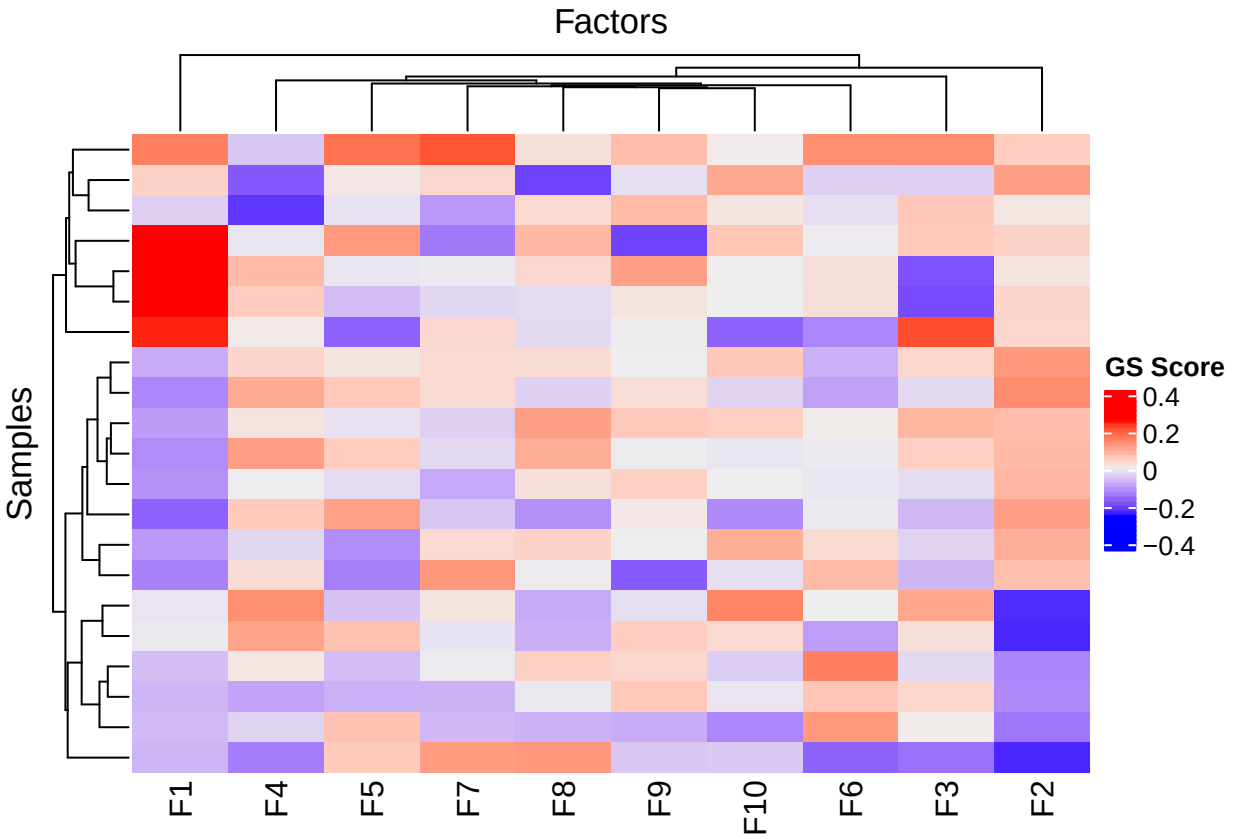
```
MCIA_plots(mcia_results,'projection_global',orders = c(1,2),coloring = 'cancerType',
           legend_loc = "bottomleft")
```



Clusters of samples can be computed from the global factors using any method, such as k-means clustering.

Visualizing the clustering of samples by factor scores

```
gs_scores = mcia_results$global_scores
colnames(gs_scores) = paste0('F', seq(1, ncol(mcia_results$global_scores)))
p = ComplexHeatmap::Heatmap(gs_scores,
    name = "GS Score",
    column_title = "Factors",
    row_title = "Samples",
    row_names_gp = grid::gpar(fontsize = 7),
    show_column_names = T,
    show_row_names = T,
    row_names_side = "right"
)
p
```

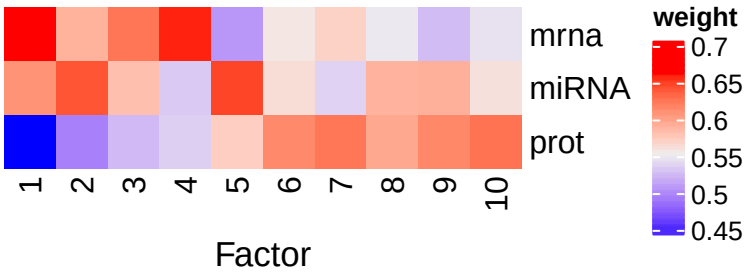


Part 2: Interpreting Global Loadings

In addition to the global scores matrix, MCIA also calculates a global loadings matrix that is $(m_1 + ... + m_j + ... + m_R) \times k$ where m_j is the number of features within the omics matrix X^j and K is the number of factors calculated. This second matrix provides information as to the contribution

Pseudoeigenvalues representing the contribution of each omic to the global factor score

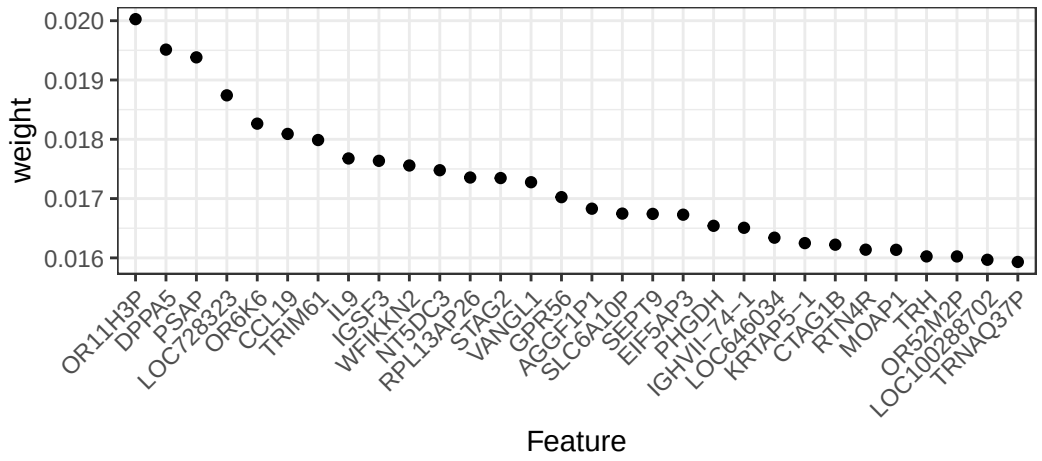
```
## Should make into a function
bs_weights<-as.matrix(data.frame(mcia_results$block_score_weights))
colnames(bs_weights)<-c(1:10)
Heatmap(bs_weights, cluster_columns=FALSE, cluster_rows=FALSE, name = "weight",
                        column_title_side = "bottom", column_title = "Factor")
```



Scree Plot: Visualizing the top features per factor

```
# make into function
gl<-mcia_results$global_loadings
```

```
omic_list<-gsub("^.*_", "", rownames(gl))
factor<-4
gl_f<-data.frame(gl[,factor])
gl_f$omic<-omic_list
colnames(gl_f)<-c("weight",'omic')
gl_f_ord<-gl_f[order(gl_f$weight, decreasing=TRUE),]

# look at mRNA (need to filter since issue with miRNA omic name)
gl_f_ord_mRNA<-gl_f_ord[gl_f_ord$omic=="mrna",]
omic_name<-sub("_.*", "",rownames(gl_f_ord_mRNA) )
gl_f_ord_mRNA$omic_name<-sub("_.*", "",rownames(gl_f_ord_mRNA) )
ggplot(gl_f_ord_mRNA[0:30,], aes(x=factor(omic_name,level=omic_name), y=weight))+
      geom_point()+
      theme_bw()+
      xlab('Feature')+
      theme(axis.text.x=element_text(angle=45, hjust=1))
```



Pathway analysis for the top factors using data from gene-centric omics blocks

. . . (to be continued)