# Package 'COCOA'

October 16, 2019

**Version** 1.2.0

**Date** 2019-3-06

**Title** Coordinate Covariation Analysis

**Description** COCOA is a method for understanding variation among samples and can be used with data that includes genomic coordinates such as DNA methylation. On a high level, COCOA uses a database of ``region sets'' and principal component analysis (PCA) of your data to identify sources of variation among samples. A region set is a set of genomic regions that share a biological annotation, for instance, transcription factor binding regions, histone modification regions, or open chromatin regions. COCOA works in both supervised (known groups of samples) and unsupervised (no groups) situations and can be used as a complement to ``differential'' methods that find discrete differences between groups. COCOA can identify biologically meaningful sources of variation between samples and increase understanding of variation in your data.

**Depends** R (>= 3.5), GenomicRanges

**Imports** BiocGenerics, S4Vectors, IRanges, data.table, ggplot2, Biobase, stats, methods, ComplexHeatmap, MIRA, tidyr, grid, grDevices

**Suggests** knitr, parallel, testthat, BiocStyle, rmarkdown, AnnotationHub, LOLA

**VignetteBuilder** knitr

**License** GPL-3

**biocViews** PrincipalComponent, GenomicVariation, DNAMethylation, GeneRegulation, GenomeAnnotation, SystemsBiology, FunctionalGenomics, ChIPSeq, MethylSeq, Sequencing, Epigenetics, ImmunoOncology

**URL** http://code.databio.org/COCOA/

**BugReports** https://github.com/databio/COCOA

**RoxygenNote** 6.1.0

**git_url** https://git.bioconductor.org/packages/COCOA

**git_branch** RELEASE_3_9

**git_last_commit** ddecaed

**git_last_commit_date** 2019-05-02

**Date/Publication** 2019-10-15

**Author** John Lawson [aut, cre],
Nathan Sheffield [aut] (http://www.databio.org)

**Maintainer** John Lawson <jtl2hk@virginia.edu>

# R topics documented:



---

`aggregateLoadings` *Use PCA loadings to score a region set*

---

## Description

First, this function identifies which loadings are within the region set. Then the loadings are used to score the region set according to the ‘scoringMetric‘ parameter.

## Usage

```
aggregateLoadings(loadingMat, signalCoord, regionSet,
  PCsToAnnotate = c("PC1", "PC2"), scoringMetric = "regionMean",
  verbose = FALSE)
```

## Arguments

| | |
|---|---|
| `loadingMat` | matrix of loadings (the coefficients of the linear combination that defines each PC). One named column for each PC. One row for each original dimension/variable (should be same order as original data/signalCoord). The x$rotation output of prcomp(). |

`signalCoord` a GRanges object or data frame with coordinates for the genomic signal/original data (eg DNA methylation) included in the PCA. Coordinates should be in the same order as the original data and the loadings (each item/row in signalCoord corresponds to a row in loadingMat). If a data.frame, must have chr and start columns. If end is included, start and end should be the same. Start coordinate will be used for calculations.

`regionSet` A genomic ranges (GRanges) object with regions corresponding to the same biological annotation. Must be from the same reference genome as the coordinates for the actual data/samples (signalCoord).

`PCsToAnnotate` A character vector with principal components to include. eg c("PC1", "PC2") These should be column names of loadingMat.

`scoringMetric` A character object with the scoring metric. "regionMean" is a weighted average of the absolute value of the loadings with no normalization (recommended). First loadings are averaged within each region, then all the regions are averaged. With "regionMean" score, be cautious in interpretation for region sets with low number of regions that overlap signalCoord. The "simpleMean" method is just the unweighted average of all absolute loadings that overlap the given region set. Wilcoxon rank sum test ("rankSum") is also supported but is skewed toward ranking large region sets highly and is significantly slower than the "regionMean" method. For the ranksum method, the absolute loadings for loadings that overlap the given region set are taken as a group and all the loadings that do not overlap the region set are taken as the other group. Then p value is then given as the score. It is a one sided test, with the alternative hypothesis that the loadings in the region set will be greater than the loadings not in the region set.

`verbose` A "logical" object. Whether progress of the function should be shown, one bar indicates the region set is completed. Useful when using aggregateLoadings with 'apply' to do many region sets at a time.

## Value

a data.table with one row and the following columns: one column for each item of PCsToAnnotate with names given by PCsToAnnotate. These columns have scores for the region set for each PC. Other columns: cytosine_coverage which has number of cytosines that overlapped with regionSet (or in the general case, coordinates from signalCoord that overlapped regionSet) region_coverage which has number of regions from regionSet that overlapped any coordinates from signalCoord, total_region_number that has number of regions in regionSet, mean_region_size that has average size in base pairs of regions in regionSet, the average is based on all regions in regionSet and not just ones that overlap.

## Examples

```
data("brcaMCoord1")
data("brcaLoadings1")
data("esr1_chr1")
rsScores <- aggregateLoadings(loadingMat=brcaLoadings1,
                              signalCoord=brcaMCoord1,
                              regionSet=esr1_chr1,
                              PCsToAnnotate=c("PC1", "PC2"),
                              scoringMetric="regionMean")
```

## atf3_chr1 *Atf3 binding regions.*

### Description

Binding regions for Atf3. hg38 genome version. Only includes regions in chr1 to keep the example data small.

### Usage

```
data(atf3_chr1)
```

### Format

A GRanges object

## brcaLoadings1 *A matrix with loadings*

### Description

This object contains loadings for PCA of DNA methylation data. DNA methlyation data is Illumina 450k microarray data from breast cancer patients from The Cancer Genome Atlas (TCGA-BRCA, https://portal.gdc.cancer.gov/). Each row corresponds to one cytosine and the coordinates for these cytosines are in the object brcaMCoord1, (data("brcaMCoord1"), hg38 genome). Only cytosines on chr1 are included to keep the example data small.

### Usage

```
data(brcaLoadings1)
```

### Format

A matrix object

## brcaMCoord1 *A data.frame object with coordinates for cytosines from chr1 included in the PCA.*

### Description

Corresponds to the rows of brcaLoadings1 and brcaMethylData1. DNA methlyation data is Illumina 450k microarray data from breast cancer patients from The Cancer Genome Atlas (TCGA-BRCA, https://portal.gdc.cancer.gov/). Coordinates correspond to the hg38 genome version. Only cytosines on chr1 are included to keep the example data small.

### Usage

```
data(brcaMCoord1)
```

### Format

A data.frame object

---

`brcaMethylData1` *A matrix with DNA methylation levels from chromosome 1 for four patients.*

---

### Description

This object contains methylation levels (0 to 1) for cytosines in chromosome 1 that were covered by the DNA methylation microarray (Illumina 450k microarray). Each row corresponds to one cytosine and the coordinates for these cytosines are in the object brcaMCoord1, (data("brcaMCoord1"), hg38 genome). Only cytosines on chr1 are included to keep the example data small. Columns are patients, with TCGA patient identifiers as column names. The patients were selected based on their PC1 score from the PCA of DNA methylation on all chromosomes. The patients with the two highest PC1 scores and the two lowest PC1 scores are included (see data("brcaPCScores") for the actual scores). DNA methlyation data is Illumina 450k microarray data from breast cancer patients from The Cancer Genome Atlas (TCGA-BRCA, https://portal.gdc.cancer.gov/).

### Usage

```
data(brcaMethylData1)
```

### Format

A matrix object

---

`brcaPCScores` *A matrix with principal component scores for PCs 1-4 for four breast cancer patients.*

---

### Description

This object contains PC scores for four patients for PCs 1-4. Columns are PCs. Rows are patients, with TCGA patient identifiers as row names. The patients were selected based on their PC1 score from the PCA of DNA methylation on all chromosomes. The patients with the two highest PC1 scores and the two lowest PC1 scores are included. DNA methlyation data is Illumina 450k microarray data from breast cancer patients from The Cancer Genome Atlas (TCGA-BRCA, https://portal.gdc.cancer.gov/), hg38 genome.

### Usage

```
data(brcaPCScores)
```

### Format

A matrix object

## brcaPCScores657 *A data.frame with principal component scores for PCs 1-4 for 657 breast cancer patients as well as a column with estrogen receptor status.*

### Description

This object contains PC scores for 657 patients for PCs 1-4. Columns are PCs as well as a column with estrogen receptor status. Rows are patients, with TCGA patient identifiers as row names. Patients were selected from all BRCA patients in TCGA based on having complete metadata information for estrogen receptor status and progesterone receptor status as well as having 450k microarray data. PCA was done on the Illumina 450k DNA methlyation microarray data (TCGA-BRCA, https://portal.gdc.cancer.gov/), hg38 genome.

### Usage

```
data(brcaPCScores657)
```

### Format

A data.frame object

## COCOA *Coordinate Covariation Analysis (COCOA)*

### Description

COCOA is a method for understanding variation among samples. COCOA can be used with data that includes genomic coordinates such as DNA methylation. To describe the method on a high level, COCOA uses a database of "region sets" and principal component analysis (PCA) of your data to identify sources of variation among samples. A region set is a set of genomic regions that share a biological annotation, for instance transcription factor (TF) binding regions, histone modification regions, or open chromatin regions. In contrast to some other common techniques, COCOA is unsupervised, meaning that samples do not have to be divided into groups such as case/control or healthy/disease, although COCOA works in those situations as well. Also, COCOA focuses on continuous variation between samples instead of having cutoffs. Because of this, COCOA can be used as a complementary method alongside "differential" methods that find discrete differences between groups of samples and it can also be used in situations where there are no groups. COCOA can identify biologically meaningful sources of variation between samples and increase understanding of variation in your data.

### Author(s)

John Lawson

Nathan Sheffield

### References

http://github.com/databio

## esr1_chr1 *Estrogen receptor alpha binding regions.*

### Description

Binding regions for estrogen receptor alpha (ESR1). hg 38 genome version. Only includes regions in chr1 to keep the example data small.

### Usage

```
data(esr1_chr1)
```

### Format

A GRanges object

## gata3_chr1 *Gata3 binding regions.*

### Description

Binding regions for gata3. hg38 genome version. Only includes regions in chr1 to keep the example data small.

### Usage

```
data(gata3_chr1)
```

### Format

A GRanges object

## getLoadingProfile *Create a "meta-region" loading profile*

### Description

This loading profile can show enrichment of genomic signals with high loading values in region set but not in surrounding genome, suggesting that variation is linked specifically to that region set.

### Usage

```
getLoadingProfile(loadingMat, signalCoord, regionSet,
  PCsToAnnotate = c("PC1", "PC2"), binNum = 25, verbose = TRUE)
```

## Arguments

| | |
|---|---|
| loadingMat | matrix of loadings (the coefficients of the linear combination that defines each PC). One named column for each PC. One row for each original dimension/variable (should be same order as original data/signalCoord). Given by prcomp(x)$rotation. |
| signalCoord | a GRanges object or data frame with coordinates for the genomic signal/original data (eg DNA methylation) included in the PCA. Coordinates should be in the same order as the original data and the loadings (each item/row in signalCoord corresponds to a row in loadingMat). If a data.frame, must have chr and start columns. If end is included, start and end should be the same. Start coordinate will be used for calculations. |
| regionSet | A genomic ranges (GRanges) object with regions corresponding to the same biological annotation. Must be from the same reference genome as the coordinates for the actual data/samples (signalCoord). |
| PCsToAnnotate | A character vector with principal components to include. eg c("PC1", "PC2") These should be column names of loadingMat. |
| binNum | Number of bins to split each region into when making the aggregate loading profile. More bins will give a higher resolution but perhaps more noisy profile. |
| verbose | A "logical" object. Whether progress of the function should be shown, one bar indicates the region set is completed. Useful when using ‘lapply‘ to get the loading profiles of many region sets. |

## Details

All regions in a given region set are combined into a single aggregate profile. Regions should be expanded on each side to include a wider area of the genome around the regions of interest (see example and vignettes). To make the profile, first we take the absolute value of the loadings. Then each region is split into ‘binNum‘ bins. All loadings in each bin are averaged to get one value per bin. Finally, corresponding bins from the different regions are averaged (eg all bin1’s averaged with each other, all bin2’s averaged with each other, etc.) to get a single "meta-region" loading profile. Since DNA strand information is not considered, the profile is averaged symmetrically around the center. A peak in the middle of this profile suggests that variability is specific to the region set of interest and is not a product of the surrounding genome. A region set can still be significant even if it does not have a peak. For example, some histone modification region sets may be in large genomic blocks and not show a peak, despite having variation across samples.

## Value

A data.frame with the binned loading profile, one row per bin. columns: binID and one column for each PC in PCsToAnnotate. The function will return NULL if there is no overlap between regionSet and signalCoord.

## Examples

```
data("brcaMCoord1")
data("brcaLoadings1")
data("esr1_chr1")
esr1_chr1_expanded <- resize(esr1_chr1, 14000, fix="center")
getLoadingProfile(loadingMat=brcaLoadings1,
                  signalCoord=brcaMCoord1,
                  regionSet=esr1_chr1_expanded,
                  PCsToAnnotate=c("PC1", "PC2"),
                  binNum=25)
```

## nrf1_chr1 *Nrf1 binding regions.*

### Description

Binding regions for Nrf1. hg38 genome version. Only includes regions in chr1 to keep the example data small.

### Usage

```
data(nrf1_chr1)
```

### Format

A GRanges object

## regionQuantileByPC *Visualize how individual regions are associated with principal components*

### Description

Visualize how much each region in a region set is associated with each PC. For each PC, the average absolute loading is calculated for each region in the region set. Then for a given PC, the average loading is converted to a percentile/quantile based on the distribution of all loadings for that PC. These values are plotted in a heatmap.

### Usage

```
regionQuantileByPC(loadingMat, signalCoord, regionSet, rsName = "",
  PCsToAnnotate = paste0("PC", 1:5), maxRegionsToPlot = 8000,
  cluster_rows = TRUE, row_title = "Region", column_title = rsName,
  column_title_side = "top", cluster_columns = FALSE,
  name = "Percentile of Loading Scores in PC", col = c("skyblue",
  "yellow"), ...)
```

### Arguments

| | |
|---|---|
| loadingMat | matrix of loadings (the coefficients of the linear combination that defines each PC). One named column for each PC. One row for each original dimension/variable (should be same order as original data/signalCoord). The x$rotation output of prcomp(). |
| signalCoord | a GRanges object or data frame with coordinates for the genomic signal/original data (eg DNA methylation) included in the PCA. Coordinates should be in the same order as the original data and the loadings (each item/row in signalCoord corresponds to a row in loadingMat). If a data.frame, must have chr and start columns. If end is included, start and end should be the same. Start coordinate will be used for calculations. |

`regionSet` A genomic ranges object with regions corresponding to the same biological annotation. These are the regions that will be visualized. Must be from the same reference genome as the coordinates for the actual data (signalCoord).

`rsName` character vector. Names of the region sets in the same order as GRList. For use as a title for each heatmap.

`PCsToAnnotate` A character vector with principal components to include. eg c("PC1", "PC2") These should be column names of loadingMat.

`maxRegionsToPlot` how many top regions from region set to include in heatmap. Including too many may slow down computation and increase memory use. If regionSet has more regions than maxRegionsToPlot, a number of regions equal to maxRegionsToPlot will be randomly sampled from the region set and these regions will be plotted. Clustering rows is a major limiting factor on how long it takes to plot the regions so if you want to plot many regions, you can also set cluster_rows to FALSE.

`cluster_rows` "logical" object, whether to cluster rows or not (may increase computation time significantly for large number of rows)

`row_title` character object, row title

`column_title` character object, column title

`column_title_side` character object, where to put the column title: "top" or "bottom"

`cluster_columns` "logical" object, whether to cluster columns. It is recommended to keep this as FALSE so it will be easier to compare PCs (with cluster_columns = FALSE, they will be in the same specified order in different heatmaps)

`name` character object, legend title

`col` a vector of colors or a color mapping function which will be passed to the ComplexHeatmap::Heatmap() function. See ?Heatmap (the "col" parameter) for more details.

`...` optional parameters for ComplexHeatmap::Heatmap()

## Value

a heatmap. Columns are PCs, rows are regions. This heatmap allows you to see if some regions are associated with certain PCs but not others. Also, you can see if a subset of regions in the region set are associated with PCs while another subset are not associated with any PCs To color each region, first the absolute loading values within that region are averaged. Then this average is compared to the distribution of absolute loading values for all individual genomic signal values to get a quantile/percentile for that region. Colors are based on this quantile/percentile. The output is a Heatmap object (ComplexHeatmap package).

## Examples

```
data("brcaLoadings1")
data("brcaMCoord1")
data("esr1_chr1")
data("brcaPCScores")
regionByPCHM <- regionQuantileByPC(loadingMat = brcaLoadings1,
                                   signalCoord = brcaMCoord1,
                                   regionSet = esr1_chr1,
```

```
                                rsName = "Estrogen Receptor Chr1",
                                PCsToAnnotate=paste0("PC", 1:2),
                                maxRegionsToPlot = 8000,
                                cluster_rows = TRUE,
                                cluster_columns = FALSE,
                                column_title = rsName,
                                name = "Percentile of Loading Scores in PC")
```

---

`rsRankingIndex` *Get indices for top scored region sets*

---

## Description

For each PC, get index of original region sets but ordered by rsScores ranking for each PC. The original index refers to that region set's position in the 'GRList' param given to 'runCOCOA' which is also that region set's row index in the COCOA output. The first number in a given column of this function's output will be the original index of the region set ranked first for that PC. Second row for a column will be the original index of the region set that ranked second for that PC, etc. You can use this function to make it easier when you want to select the top region sets for further analysis or just for sorting the results. Region set scores are sorted in decreasing order so if you have p values they should be log transformed: -log(pval, 10)

## Usage

```
rsRankingIndex(rsScores, PCsToAnnotate)
```

## Arguments

| | |
|---|---|
| `rsScores` | a data.frame with scores for each region set from the main COCOA function. Each row is a region set. Columns are PCs and info on region set overlap with DNA methylation data. Should be in the same order as GRList (the list of region sets used to create it.) |
| `PCsToAnnotate` | a character vector. PCs in rsScores for which you want the indices of the original region sets (must be column names of rsScores) eg c("PC1", "PC2") |

## Value

A data.frame with columns PCsToAnnotate. Each column has been sorted by score for region sets for that PC (decreasing order). Original indices for region sets that were used to create rsScores are given. Region sets with a score of NA are counted as having the lowest scores and indices for these region sets will be at the bottom of the returned data.frame (na.last=TRUE in sorting)

## Examples

```
data("rsScores")
rsRankInd = rsRankingIndex(rsScores=rsScores,
                           PCsToAnnotate=c("PC1", "PC2"))
# region sets sorted by score for PC1
rsScores[rsRankInd$PC1, ]
# region sets sorted by score for PC2
rsScores[rsRankInd$PC2, ]
```

`rsScoreHeatmap` *Heatmap of region set scores*

## Description

Heatmap of the ranking of region set scores across PCs A visualization of the rank of region sets in each PC, allowing the user to see if a region set is ranked highly in all PCs or only a subset. Region sets will be ranked from highest scoring to lowest based on their score for 'orderByPC'. The ComplexHeatmap package is used and additional parameters for the ComplexHeatmap::Heatmap function may be passed to this function to modify the heatmap.

## Usage

```
rsScoreHeatmap(rsScores, PCsToAnnotate = paste0("PC", 1:5),
  orderByPC = "PC1", rsNameCol = "rsName", topX = 20,
  col = c("red", "#EEEEEE", "blue"), row_title = "Region Set",
  column_title = "Principal Component", column_title_side = "bottom",
  cluster_rows = FALSE, cluster_columns = FALSE,
  show_row_names = TRUE, row_names_max_width = unit(10000, "mm"),
  name = "Rank within PC", ...)
```

## Arguments

| | |
|---|---|
| `rsScores` | a data.frame with scores for each region set from main COCOA function 'runCOCOA'. Each row is a region set. Columns are scores, one column for each PCs Also can have columns with info on region set overlap with the original data. Should be in the same order as GRList (the list of region sets used to create it.) |
| `PCsToAnnotate` | A character vector with principal components to include. eg c("PC1", "PC2"). Must be column names of rsScores. |
| `orderByPC` | a character object. PC to order by in heatmap (arranged in decreasing order for scores so p values should be -log transformed). Must be the name of a column in rsScores. |
| `rsNameCol` | character. Name of the column in rsScores that has the names/identifiers for the region sets so these can be included in the plot as row names. |
| `topX` | Number of top region sets to include in the heatmap |
| `col` | a vector of colors or a color mapping function which will be passed to the ComplexHeatmap::Heatmap() function. See ?Heatmap (the "col" parameter) for more details. "#EEEEEE" is the code for a color similar to white. |
| `row_title` | character object, row title |
| `column_title` | character object, column title |
| `column_title_side` | character object, where to put the column title: "top" or "bottom" |
| `cluster_rows` | "logical" object, whether rows should be clustered. This should be kept as FALSE to keep the correct ranking of region sets. |
| `cluster_columns` | "logical" object, whether to cluster columns. It is recommended to keep this as FALSE so it will be easier to compare PCs (with cluster_columns = FALSE, they will be in the same specified order in different heatmaps) |

| | |
|---|---|
| show_row_names | "logical" object, display row names (ie region set names) |
| row_names_max_width | "unit" object. The amount of room to allocate for row names. See ?grid::unit for object type. |
| name | character object, legend title |
| ... | optional parameters for ComplexHeatmap::Heatmap(). |

## Value

A heatmap of region set scores across. Each row is a region set, each column is a PC. The color corresponds to the relative rank of a region set's score for a given PC out of all tested region sets.

## Examples

```
data("rsScores")
scoreHeatmap <- rsScoreHeatmap(rsScores,
          PCsToAnnotate=paste0("PC", 1:2), orderByPC = "PC2")
```

---

# rsScores *Example COCOA Results (made up)*

---

## Description

A data.frame with example COCOA results. 5 region sets with names given by rsScores$rsName. Each region set has a score for each PC. Scores for real region sets would normally be orders of magnitude smaller.

## Usage

```
data(rsScores)
```

## Format

A data.frame object

---

# runCOCOA *Do COCOA with many region sets*

---

## Description

This function will give each region set a score for each PC in 'PCsToAnnotate' based on the 'scoringMetric' parameter. Based on these scores, you can determine which region sets out of a region set database (given by GRList) are most associated with the top PCs. See the vignette "Introduction to Coordinate Covariation Analysis" for help interpreting your results.

## Usage

```
runCOCOA(loadingMat, signalCoord, GRList, PCsToAnnotate = c("PC1",
  "PC2"), scoringMetric = "regionMean", verbose = TRUE)
```

## Arguments

| | |
|---|---|
| loadingMat | matrix of loadings (the coefficients of the linear combination that defines each PC). One named column for each PC. One row for each original dimension/variable (should be same order as original data/signalCoord). The x$rotation output of prcomp(). |
| signalCoord | a GRanges object or data frame with coordinates for the genomic signal/original data (eg DNA methylation) included in the PCA. Coordinates should be in the same order as the original data and the loadings (each item/row in signalCoord corresponds to a row in loadingMat). If a data.frame, must have chr and start columns. If end is included, start and end should be the same. Start coordinate will be used for calculations. |
| GRList | GRangesList object. Each list item is a distinct region set to test (region set: regions that correspond to the same biological annotation). The region set database. Must be from the same reference genome as the coordinates for the actual data/samples (signalCoord). |
| PCsToAnnotate | A character vector with principal components to include. eg c("PC1", "PC2") These should be column names of loadingMat. |
| scoringMetric | A character object with the scoring metric. "regionMean" is a weighted average of the absolute value of the loadings with no normalization (recommended). First loadings are averaged within each region, then all the regions are averaged. With "regionMean" score, be cautious in interpretation for region sets with low number of regions that overlap signalCoord. The "simpleMean" method is just the unweighted average of all absolute loadings that overlap the given region set. Wilcoxon rank sum test ("rankSum") is also supported but is skewed toward ranking large region sets highly and is significantly slower than the "regionMean" method. For the ranksum method, the absolute loadings for loadings that overlap the given region set are taken as a group and all the loadings that do not overlap the region set are taken as the other group. Then p value is then given as the score. It is a one sided test, with the alternative hypothesis that the loadings in the region set will be greater than the loadings not in the region set. |
| verbose | A "logical" object. Whether progress of the function should be shown, one bar indicates the region set is completed. |

## Value

data.frame of results, one row for each region set. One column for each PC in PCsToAnnotate with score for that PC for a given region set (specific score depends on "scoringMetric" parameter). Rows will be in the same order as region sets in GRList "cytosine_coverage" column has number of cytosines that overlapped with the given region set (or in the general case, coordinates from signalCoord that overlapped regionSet). "region_coverage" column has number of regions that overlapped any coordinates from signalCoord. "total_region_number" column has total number of regions. "mean_region_size" has average region size (average of all regions, not just those that overlap a cytosine).

## Examples

```
data("brcaMCoord1")
data("brcaLoadings1")
data("esr1_chr1")
rsScores <- runCOCOA(loadingMat=brcaLoadings1,
                                  signalCoord=brcaMCoord1,
```

```
GRList=GRangesList(esr1_chr1),
PCsToAnnotate=c("PC1", "PC2"),
scoringMetric="regionMean")
```

---

`signalAlongPC` *Visualize how genomic signal in a region set changes along principal component axis*

---

## Description

Look at genomic signal (eg, DNA methylation values) in regions of interest across samples, with samples ordered according to score for PC of interest. The ComplexHeatmap package is used and additional parameters for the ComplexHeatmap::Heatmap function may be passed to this function to modify the heatmap.

## Usage

```
signalAlongPC(genomicSignal, signalCoord, regionSet, pcScores,
  orderByPC = "PC1", cluster_columns = FALSE, cluster_rows = FALSE,
  row_title = "Sample", column_title = "Genomic Signal",
  column_title_side = "bottom", name = "Genomic Signal Value",
  col = c("blue", "#EEEEEE", "red"), ...)
```

## Arguments

| | |
|---|---|
| `genomicSignal` | The genomic signal (eg DNA methylation levels) in matrix or data.frame. Rows are individual signal/feature values. Columns are samples. Must have sample names/IDs as column names, These same sample names must be row names of pcScores. |
| `signalCoord` | a GRanges object or data frame with coordinates for the genomic signal/original data (eg DNA methylation) included in the PCA. Coordinates should be in the same order as the original data and the loadings (each item/row in signalCoord corresponds to a row in loadingMat). If a data.frame, must have chr and start columns. If end is included, start and end should be the same. Start coordinate will be used for calculations. |
| `regionSet` | A genomic ranges object with regions corresponding to the same biological annotation. The regions where you will visualize the genomic signal. Must be from the same reference genome as the coordinates for the actual data (signalCoord). |
| `pcScores` | A matrix. The principal component scores for the samples (ie transformed methylation data). Must have sample names/IDs as row names, These same sample names must be column names of genomicSignal |
| `orderByPC` | a character object. PC to order samples by (order rows of heatmap by PC score, from high to low score). Must be the name of a column in pcScores. |
| `cluster_columns` | "logical" object, whether to cluster columns (the genomic signal, eg DNA methylation values for each CpG). |

| | |
|---|---|
| cluster_rows | "logical" object, whether rows should be clustered. This should be kept as FALSE to keep the correct ranking of samples/observations according to their PC score. |
| row_title | character object, row title |
| column_title | character object, column title |
| column_title_side | character object, where to put the column title: "top" or "bottom" |
| name | character object, legend title |
| col | a vector of colors or a color mapping function which will be passed to the ComplexHeatmap::Heatmap() function. See ?Heatmap (the "col" parameter) for more details. "#EEEEEE" is the code for a color similar to white. |
| ... | optional parameters for ComplexHeatmap::Heatmap() |

## Value

A heatmap of genomic signal values (eg DNA methylation levels) in regions of interest (regionSet), with rows ordered by PC score. Each row is a patient/sample and each column is an individual genomic signal value. Rows are ordered by PC score for 'orderByPC', high scores at top and low at the bottom.

## Examples

```
data("brcaMethylData1")
data("brcaMCoord1")
data("esr1_chr1")
data("brcaPCScores")
signalHM <- signalAlongPC(genomicSignal=brcaMethylData1,
                          signalCoord=brcaMCoord1,
                          regionSet=esr1_chr1,
                          pcScores=brcaPCScores,
                          orderByPC="PC1", cluster_columns=TRUE)
```

# Index