

# Package ‘SEMPLR’

April 13, 2026

**Title** SNP Effect Matrix Pipeline in R

**Version** 0.99.2

**Description** SEMPLR computes transcription factor binding affinity scores for genomic positions and genetic variants. Scores are computed from SNP Effect Matrices (SEMs) produced by SEMpl. 223 pre-computed SEMs are included with the package or custom sets can be provided. Enrichment can be tested among sets of genomic positions to determine if transcription factor binding events occur more often than expected. Comparing binding affinity scores between alleles can reveal differences in transcription factor binding with genetic variation. This package also includes several visualization functions to view scores both on the motif and variant/position level.

**Imports** BiocGenerics, Biostrings, GenomeInfoDb, AnnotationDbi, ggplot2, ggrepel, VariantAnnotation, GenomicRanges, GenomicFeatures, data.table, methods, scales, S4Vectors, stats, rlang, stringi, universalmotif, Rcpp, ggtree

**Encoding** UTF-8

**Roxygen** list(markdown = TRUE)

**RoxygenNote** 7.3.3

**Suggests** knitr, rmarkdown, BiocStyle, devtools, testthat (>= 3.0.0), IRanges, BSgenome.Hsapiens.UCSC.hg38, BSgenome.Hsapiens.UCSC.hg19, TxDb.Hsapiens.UCSC.hg38.knownGene, org.Hs.eg.db

**VignetteBuilder** knitr

**License** MIT + file LICENSE

**Config/testthat/edition** 3

**URL** <https://github.com/grkenney/SEMPLR>,  
<https://grkenney.github.io/SEMPLR>

**BugReports** <https://www.github.com/grkenney/SEMPLR/issues>

**biocViews** MotifAnnotation, Transcription, SNP, GenomicVariation

**LazyData** true

**LinkingTo** Rcpp**git\_url** <https://git.bioconductor.org/packages/SEMPLR>**git\_branch** devel**git\_last\_commit** 5d0ce71**git\_last\_commit\_date** 2026-03-09**Repository** Bioconductor 3.23**Date/Publication** 2026-04-12

**Author** Grace Kenney [aut, cre] (ORCID:  
<https://orcid.org/0009-0009-6308-3150>),  
 Douglas Phanstiel [aut],  
 NIH NIGMS [fnd],  
 NSF GRFP [fnd]

**Maintainer** Grace Kenney <kenney.grace6@gmail.com>**Contents**

SEMPLR-package . . . . .	3
convertSEMsToPPMs . . . . .	3
enrichmentSets . . . . .	4
enrichSEMs . . . . .	6
getBaseline . . . . .	7
getRanges . . . . .	8
getRangeSeqs . . . . .	9
getSEM . . . . .	10
getSEMId . . . . .	10
getSEMs . . . . .	11
loadSEMCollection . . . . .	12
plotEnrich . . . . .	13
plotSEM . . . . .	14
plotSEMMotifs . . . . .	16
plotSEMVariants . . . . .	17
scoreBinding . . . . .	18
scores . . . . .	19
scoreVariants . . . . .	20
SEMC . . . . .	21
semData . . . . .	22
SEMScores . . . . .	23
SEMScores-class . . . . .	24
show,SEMScores-method . . . . .	25
show,SNPEffectMatrix-method . . . . .	25
show,SNPEffectMatrixCollection-method . . . . .	26
SNPEffectMatrix . . . . .	26
SNPEffectMatrix-class . . . . .	27
SNPEffectMatrixCollection . . . . .	27
SNPEffectMatrixCollection-class . . . . .	28

**Description**

SEMPLR computes transcription factor binding affinity scores for genomic positions and genetic variants. Scores are computed from SNP Effect Matrices (SEMs) produced by SEMpl. 223 pre-computed SEMs are included with the package or custom sets can be provided. Enrichment can be tested among sets of genomic positions to determine if transcription factor binding events occur more often than expected. Comparing binding affinity scores between alleles can reveal differences in transcription factor binding with genetic variation. This package also includes several visualization functions to view scores both on the motif and variant/position level.

**Author(s)**

**Maintainer:** Grace Kenney <kenney.grace6@gmail.com> ([ORCID](#))

Authors:

- Douglas Phanstiel

Other contributors:

- NIH NIGMS [funder]
- NSF GRFP [funder]

**See Also**

Useful links:

- <https://github.com/grkenney/SEMPLR>
- <https://grkenney.github.io/SEMPLR>
- Report bugs at <https://www.github.com/grkenney/SEMPLR/issues>

**Description**

Converts the SNP Effect Matrix in a SNPEffectMatrix or SNPEffectMatrixCollection object to a position probability matrix (PPM) where each entry is the probability of a base at each position in the motif.

**Usage**

```
convertSEMstoPPMs(x)
```

**Arguments**

x                    A SNPEffectMatrix or SNPEffectMatrixCollection object

**Value**

a list of matrices

**Examples**

```
# Load default SEMs

# From a SNPEffectMatrixCollection
convertSEMsToPPMs(SEMC)

# From a single SNPEffectMatrix
convertSEMsToPPMs(getSEMs(SEMC, "JUN"))
```

---

enrichmentSets

*Run the Full Differential-Expression Motif Enrichment Pipeline*

---

**Description**

enrichmentSets() is a one-stop wrapper that takes a vector of user-provided IDs from a differential expression analysis and returns promoter regions plus matched background sets for downstream motif enrichment. It performs:

1. ID mapping
2. Optional biotype filtering
3. Promoter coordinate extraction
4. Background sampling within that same call

**Usage**

```
enrichmentSets(
  txdb,
  orgdb,
  id_type,
  foreground_ids,
  background_ids = NULL,
  transcript = FALSE,
  threshold = 0.9,
  stripVersions = TRUE,
  inflateThresh = 1,
  geneType = NULL,
  overlapMinGap = 0,
  onePromoterPerGene = FALSE,
```

```

n_ratio = 1,
promoterWindow = c(upstream = 300, downstream = 50),
standardChroms = TRUE,
reduceOverlaps = TRUE
)

```

## Arguments

<code>txdb</code>	A TxDb object. (e.g. TxDb.Hsapiens.UCSC.hg38.knownGene).
<code>orgdb</code>	An OrgDb object. (e.g. org.Hs.eg.db).
<code>id_type</code>	Type of identifier supplied in foreground and background IDs. See the keytypes(orgdb) for available id type options for each genome.
<code>foreground_ids</code>	Character vector of gene or transcript IDs (e.g. Ensembl, RefSeq, gene symbols) to analyze.
<code>background_ids</code>	Character vector of gene or transcript IDs to use as background set.
<code>transcript</code>	Logical; TRUE to treat inputs as transcript-level, FALSE for gene-level.
<code>threshold</code>	Numeric in range 0 to 1. Min fraction of IDs that must map to pick a keytype (default 0.9).
<code>stripVersions</code>	Logical; strip version suffixes (e.g. ".1") from Ensembl/RefSeq IDs.
<code>inflateThresh</code>	Numeric in range 0 to 1; max allowed transcript:gene inflation before auto-collapsing (default 1).
<code>geneType</code>	Optional character; biotype filter (e.g. "protein-coding").
<code>overlapMinGap</code>	Numeric; minimum gap when reducing overlapping promoters.
<code>onePromoterPerGene</code>	Logical; if TRUE, pick only one promoter per gene.
<code>n_ratio</code>	Numeric; ratio of ranges to retain in the background set. The number of background ranges will be equal to n_ratio multiplied by the number of foreground ranges.
<code>promoterWindow</code>	Numeric named vector c(upstream, downstream); promoter flank widths in bp (default c(300, 50)).
<code>standardChroms</code>	Logical; restrict to standard chromosomes.
<code>reduceOverlaps</code>	Logical; merge overlapping promoter windows.

## Value

A list containing:

**backgroundElements** GRanges of sampled background promoters

**foregroundElements** GRanges of foreground promoters

**backgroundUniverse** GRanges of the pruned promoter pool

**Examples**

```

library(TxDb.Hsapiens.UCSC.hg38.knownGene)
library(org.Hs.eg.db)

txdb <- TxDb.Hsapiens.UCSC.hg38.knownGene
orgdb <- org.Hs.eg.db

# Note that this is a minimal set for demonstration only
genes <- c("ENSG00000139618", "ENSG00000157764")
background <- c("ENSG00000111640", "ENSG00000175624")

# Minimal run with defaults:
results <- enrichmentSets(genes,
  background,
  txdb = txdb,
  orgdb = orgdb,
  id_type = "ENSEMBL"
)

```

---

enrichSEMs

*Calculates binding enrichment for motif(s)*


---

**Description**

Perform a binomial test to determine if SNP Effect Matrices are bound more often than expected.

**Usage**

```
enrichSEMs(x, sem, background = NULL, seqs = NULL, nFlank = 0, genome = NULL)
```

**Arguments**

x	The scoring table produced by scoreBinding
sem	A SNPEffectMatrix or SNPEffectMatrixCollection object
background	A GRanges object or a list of DNA sequences to use as a background set for the binomial test. The length of each sequence must match the length of sequences in x. By default, will scramble the provided sequences.
seqs	The sequences scored in scoreBinding
nFlank	Number of flanking nucleotides added to the sequences. Defaults to the length of the longest motif. If no flanks were added (ie, sequences were scored rather than a GRanges), use nFlank = 0.
genome	A BSGenome object. Required if background isn't specified.

**Value**

a list of matrices

**Examples**

```
# load SEMs

# note that this is a small example for demonstration purposes
# in actual enrichment analyses sets of 100+ ranges are recommended

# create a GRanges object
gr <- GenomicRanges::GRanges(
  seqnames = "chr12",
  ranges = 94136009
)

# calculate binding propensity
sb <- scoreBinding(gr, SEMC, BSgenome.Hsapiens.UCSC.hg19::Hsapiens)

enrichSEMs(sb, SEMC)
```

---

getBaseline	<i>Access baseline from a SNPEffectMatrix object</i>
-------------	--

---

**Description**

Access baseline from a SNPEffectMatrix object

**Usage**

```
getBaseline(x)

## S4 method for signature 'SNPEffectMatrix'
getBaseline(x)
```

**Arguments**

x                   SNPEffectMatrix object

**Value**

The numeric baseline value is returned

**Examples**

```
# Isolate a single SNPEffectMatrix object from the default
# SNPEffectMatrixCollection
sm <- getSEMs(SEMC)[[1]]

# Access the baseline
getBaseline(sm)
```

---

getRanges	<i>Access ranges slot in a SEMScores object</i>
-----------	---

---

**Description**

Access ranges slot in a SEMScores object

**Usage**

```
getRanges(x)

## S4 method for signature 'SEMScores'
getRanges(x)
```

**Arguments**

x                    a SEMScores object

**Value**

A GRanges or VRanges object

**Examples**

```
library(VariantAnnotation)

# load default SEMs

# create a VRanges object
vr <- VRanges(
  seqnames = "chr12",
  ranges = 94136009,
  ref = "G", alt = "C"
)

# calculate binding propensity
s <- scoreVariants(vr, SEMC, BSgenome.Hsapiens.UCSC.hg19::Hsapiens)

getRanges(s)
```

---

`getRangeSeqs`*Get sequence for genomic ranges and variants*

---

### Description

Query sequences associated with a given range in a reference genome and optionally include nucleotides up and downstream of the range of interest sequences. Store the results in the metadata of the range object.

### Usage

```
getRangeSeqs(x, genome, up = 0, down = 0, refCol = NULL, altCol = NULL)
```

### Arguments

<code>x</code>	A VRanges or GRanges object with one or more variants. seqnames and ranges fields are required.
<code>genome</code>	A BSgenome object for the genome build to use.
<code>up</code>	Numeric, number of bases to return upstream of variant
<code>down</code>	Numeric, number of bases to return downstream of variant
<code>refCol</code>	Column name in meta data storing the ref allele
<code>altCol</code>	Column name in meta data storing the alt allele

### Value

a x with added meta data columns. By default, only a "sequence" column is added to the meta data. If variant information is supplied, either in a VRanges object or through the "allele" parameter, "ref\_seq" and "alt\_seq" columns are added to the meta data.

### Examples

```
# set reference genome
b <- BSgenome.Hsapiens.UCSC.hg19::Hsapiens

# Query sequence for a GRanges object
gr <- GenomicRanges::GRanges(
  seqnames = "chr12",
  ranges = IRanges::IRanges(94136009, 94136019)
)
getRangeSeqs(gr, genome = b)

# Query variant sequences for a VRanges object with 10bp flanks
vr <- VariantAnnotation::VRanges(
  seqnames = "chr12",
  ranges = IRanges::IRanges(94136009),
  ref = "G", alt = "C"
)
```

```
getRangeSeqs(vr, genome = b, up = 10, down = 10)
```

---

getSEM	<i>Access the SEM from a SNPEffectMatrix object</i>
--------	---

---

### Description

Access the SEM from a SNPEffectMatrix object

### Usage

```
getSEM(x)

## S4 method for signature 'SNPEffectMatrix'
getSEM(x)
```

### Arguments

x                   SNPEffectMatrix object

### Value

A position (rows) by nucleic acid (columns) data . table is returned

### Examples

```
# Isolate a single SNPEffectMatrix object from the default
# SNPEffectMatrixCollection
sm <- getSEMs(SEMC)[[1]]

# Access the matrix
getSEM(sm)
```

---

getSEMId	<i>Access semId from a SNPEffectMatrix object</i>
----------	---

---

### Description

Access semId from a SNPEffectMatrix object

### Usage

```
getSEMId(x)

## S4 method for signature 'SNPEffectMatrix'
getSEMId(x)
```

**Arguments**

x                   SNPEffectMatrix object

**Value**

The character id is returned

**Examples**

```
# Isolate a single SNPEffectMatrix object from the default
# SNPEffectMatrixCollection
sm <- getSEMs(SEMC)[[1]]

# Access the SEM id
getSEMid(sm)
```

---

getSEMs

*Access sems matrix from a SNPEffectMatrixCollection object*

---

**Description**

Access sems matrix from a SNPEffectMatrixCollection object

**Usage**

```
getSEMs(x, semId = NULL)

## S4 method for signature 'SNPEffectMatrixCollection'
getSEMs(x, semId = NULL)
```

**Arguments**

x                   SNPEffectMatrixCollection object

semId               optional character corresponding to an SEM in the SNPEffectMatrixCollection object. See semIds with names(getSEMs(x)). Defaults to returning all SEMs.

**Value**

A position (rows) by nucleic acid (columns) data.table is returned

**Examples**

```
## Create example SEM
df <- data.frame(
  A = c(1, 2, 3),
  C = c(1, 2, 3),
  G = c(1, 2, 3),
  T = c(1, 2, 3)
)
s <- SNPEffectMatrix(df, 1.205, "motif_id")
sc <- SNPEffectMatrixCollection(s)

## Access count matrix
getSEMs(sc)
```

---

loadSEMCollection      *Load .sem files and meta data into a SNPEffectMatrixCollection*

---

**Description**

Load .sem files and meta data into a SNPEffectMatrixCollection

**Usage**

```
loadSEMCollection(
  semFiles,
  semMetaData = NULL,
  semMetaKey = "",
  semIds = NULL,
  bls = NULL
)
```

**Arguments**

semFiles	A list of paths to .sem files. Expects header of .sem files to be in format #BASELINE:{bl} where bl is the numeric baseline value. If matrix does not include baseline header, must be specified in bl.
semMetaData	A data.table with meta data on each SEM
semMetaKey	The name of a column in semData that matches the semIds in the sems list as a character. If column entries have a .sem suffix, a new column named SEM_KEY will be created without the .sem suffixes.
semIds	Unique id for the sem as a character vector in same order as sems. Defaults to semFile file name without the extension.
bls	numeric vector or baseline values for the SEMs. Overrides baseline specified in semFile header.

**Value**

A SNPEffectMatrix object

**Examples**

```
# write a tmp file to hold a SEM
m <- matrix(rnorm(16), nrow = 4)
colnames(m) <- c("A", "C", "G", "T")
tf <- tempfile()
write.table(m, tf, quote = FALSE, sep = "\t", row.names = FALSE)

# build a meta data table
md <- data.table::data.table(
  transcription_factor = c("tf1"),
  cell_type = c("HepG2"),
  sem_id = c("sem_id")
)

loadSEMCollection(tf,
  semMetaData = md, semIds = "sem_id",
  semMetaKey = "sem_id", bls = 1
)
```

---

plotEnrich

*Plot the results of enrichSEMs*

---

**Description**

Generates a circular dendrogram, clustering SNP Effect Matrices on similarity and a heatmap representing the  $-\log_{10}$  transformed adjusted p-value of a SEMPLR enrichment.

**Usage**

```
plotEnrich(
  e,
  sem,
  label = "transcription_factor",
  method = "WPCC",
  threshold = 0.05,
  lineWidth = 0.5,
  textCols = c("darkgrey", "black"),
  textCex = 1,
  heatmapCols = c("white", "red"),
  pvalRange = c(0, 20)
)
```

**Arguments**

e	The resulting data.table from enrichSEMs
sem	A SNPEffectMatrixCollection object
label	Column in semData(sem) to use for tree labels
method	Method to use for SEM comparison. See ?universalmotif::compare_motifs for options.
threshold	The adjusted p-value threshold for coloring SEMs
lineWidth	A numeric specifying the dendrogram line width
textCols	A vector of two colors to label non-significant and significant SEMs respectively.
textCex	Text size of SEM labels.
heatmapCols	A vector of two colors to use for the heatmap, ordered low to high $-\log_{10}(\text{padj})$ .
pvalRange	A vector of 2 numerics to use as the scale range for the heatmap of $-\log_{10}(\text{padj})$ .

**Value**

a ggtree object

**Examples**

```
# load SEMs

# note that this is a small example for demonstration purposes
# in actual enrichment analyses sets of 100+ ranges are recommended

# create a GRanges object
gr <- GenomicRanges::GRanges(
  seqnames = "chr12",
  ranges = 94136009
)

# calculate binding propensity
sb <- scoreBinding(gr, SEMC, BSgenome.Hsapiens.UCSC.hg19::Hsapiens)

e <- enrichSEMs(sb, SEMC)
plotEnrich(e, SEMC)
```

---

plotSEM

*Plot SEM scores for each nucleic acid in each position of the motif*


---

**Description**

Plot SEM scores for each nucleic acid in each position of the motif

**Usage**

```
plotSEM(
  sem,
  motif = NULL,
  motifSeq = NULL,
  rc = FALSE,
  cols = c("lightgrey", "dodgerblue"),
  size = 7,
  hindex = NULL,
  hcol = "dodgerblue",
  halpha = 0.1,
  hwidth = 15,
  lcol = "#d7dbdd",
  lwidth = 1
)
```

**Arguments**

sem	A SNPEffectMatrix or SNPEffectMatrixCollection object.
motif	sem id to plot. If providing a SNPEffectMatrixCollection object, must match a name in the list. If providing a single SNPEffectMatrix object, this parameter is ignored and the SNPEffectMatrix's semId is used for plotting.
motifSeq	Character sequence to color on plot
rc	Boolean indicating whether to plot the reverse complement orientation of the SEM
cols	A vector of two colors to color the nucleotides not included and included in the provided motifSeq respectively
size	Font size of nucleotides in plot
hindex	Index of nucleotide to highlight in the motifSeq
hcol	The color of the vertical highlight bar
halpha	The alpha transparency of the vertical highlight bar
hwidth	Width of highlight bar
lcol	Color of the horizontal endogenous and background binding lines
lwidth	Width of the horizontal endogenous and background binding lines

**Value**

a ggplot with sem scores for each nucleic acid per position

**Examples**

```
library(VariantAnnotation)

# Given a SNPEffectMatrix Collection
plotSEM(SEMC, motif = "JUN")
```

```
# Given a single SNPEffectMatrix
sem <- getSEMs(SEMC, "JUN")
plotSEM(sem)

# color by sequence
plotSEM(sem, motifSeq = "TGAGTCA", hindex = 2)
```

---

plotSEMMotifs

*Plot non-alt versus alt binding propensity for a single variant*


---

### Description

Plot non-alt versus alt binding propensity for a single variant

### Usage

```
plotSEMMotifs(
  s,
  variant,
  label = "transcription_factor",
  rc = TRUE,
  labsize = 4,
  cols = c("#F8766D", "dodgerblue2"),
  ptsize = 1
)
```

### Arguments

s	a SEMScores object with scores populated
variant	variant id to plot
label	column in sem_metadata slot of simplObj to use for point labels
rc	label SEM orientations
labsize	numeric size of the point labels
cols	vector of length 2 with colors to use for plotting gained and lost motifs respectively
ptsized	numeric size of the ggplot points

### Value

a ggplot scatter plot of non-alt binding propensity versus alt binding propensity

**Examples**

```

library(VariantAnnotation)

# create a VRanges object
vr <- VRanges(
  seqnames = "chr12",
  ranges = 94136009,
  ref = "G", alt = "C"
)

# calculate binding propensity
s <- scoreVariants(vr, SEMC, BSgenome.Hsapiens.UCSC.hg19::Hsapiens)

plotSEMMotifs(s, "chr12:94136009:G>C", label = "transcription_factor")

```

---

plotSEMVariants      *Plot non-alt versus alt binding propensity for a single motif*

---

**Description**

Plot non-alt versus alt binding propensity for a single motif

**Usage**

```

plotSEMVariants(
  s,
  sem,
  label = "varId",
  labsize = 4,
  cols = c("#F8766D", "dodgerblue2"),
  ptsize = 1
)

```

**Arguments**

s	a SEMScores object with scores populated
sem	a single character vector matching a SEM in the sempObj
label	column in scores slot of sempObj to use for point labels
labsize	numeric size of the point labels
cols	vector of length 2 with colors to use for plotting gained and lost motifs respectively
ptsized	numeric size of the ggplot points

**Value**

a ggplot scatter plot of non-alt binding propensity versus alt binding propensity

**Examples**

```

library(VariantAnnotation)

# create a VRanges object
vr <- VRanges(
  seqnames = "chr12",
  ranges = 94136009,
  ref = "G", alt = "C"
)

# calculate binding propensity
s <- scoreVariants(vr, SEMC, BSgenome.Hsapiens.UCSC.hg19::Hsapiens)

plotSEMVariants(s, "IKZF1_HUMAN.GM12878")

```

---

scoreBinding	<i>Calculate binding propensity for all SEM motifs and genomic positions provided</i>
--------------	---

---

**Description**

Calculate binding propensity for all SEM motifs and genomic positions provided

**Usage**

```
scoreBinding(x, sem, genome, nFlank = NULL, seqId = NULL, rc = TRUE)
```

**Arguments**

x	GRanges object or a vector of DNA sequences
sem	A SNPEffectMatrix or SNPEffectMatrixCollection object
genome	A BSgenome object for the genome build to use. ie. BSgenome.Hsapiens.UCSC.hg19::Hsapiens. Required if providing a GRanges object. Ignored if providing a vector of sequences.
nFlank	Number of flanking nucleotides to add to provided range. By default will add flank equal to the length of the longest motif. Ignored if providing a vector of sequences.
seqId	Column in GRanges object to use for unique id. By default, ids will be generated from the seqnames and ranges. Ignored if not providing a GRanges object.
rc	plot the reverse complement SEMs

**Value**

If a GRanges object is provided, return a SEMScores object. If a list of sequences is provided, just return the scoring table. The scoring table will contain the following columns:

- seqId: a unique identifier for the sequence scored
- SEM: the identifier for the SEM
- rc: the orientation of the sequence (fwd or rev)
- score: the unnormalized SEM score
- scoreNorm: the SEM score normalized to it's corresponding baseline
- index: the index of the motif with the highest SEM score within the sequence scored
- seq: the motif sequence with the highest SEM scores within the sequence scored

**Examples**

```
# load SEMs

# create a GRanges object
gr <- GenomicRanges::GRanges(
  seqnames = "chr12",
  ranges = 94136009
)

# calculate binding propensity
scoreBinding(gr, SEMC, BSgenome.Hsapiens.UCSC.hg19::Hsapiens)
```

---

scores

*Accessor scores slot in a SEMScores object*

---

**Description**

Accessor scores slot in a SEMScores object

**Usage**

```
scores(x)

## S4 method for signature 'SEMScores'
scores(x)
```

**Arguments**

x                    a SEMScores object

**Examples**

```

library(VariantAnnotation)

# load default SEMs

# create a VRanges object
vr <- VRanges(
  seqnames = "chr12",
  ranges = 94136009,
  ref = "G", alt = "C"
)

# calculate binding propensity
s <- scoreVariants(vr, SEMC, BSgenome.Hsapiens.UCSC.hg19::Hsapiens)

scores(s)

```

---

scoreVariants	<i>Calculate risk/non-risk binding propensity for all SEM motifs and variants provided</i>
---------------	--

---

**Description**

Calculate risk/non-risk binding propensity for all SEM motifs and variants provided

**Usage**

```

scoreVariants(
  x,
  sem,
  genome,
  refCol = NULL,
  altCol = NULL,
  varId = NULL,
  rc = TRUE
)

```

**Arguments**

x	VRanges object
sem	a list of SNPEffectMatrix objects
genome	A BSgenome object for the genome build to use. ie. BSgenome.Hsapiens.UCSC.hg19::Hsapiens
refCol	If providing a GRanges, the meta data column name with the reference (ref) allele. Ignored if providing a VRanges object.
altCol	If providing a GRanges, the meta data column name with the alternative (alt) allele. Ignored if providing a VRanges object.

varId	A column name in the meta data of x to use as a unique id.
rc	plot the reverse complement SEMs

**Value**

a SEMScores object with slots for the ranges of the provided variants with an added sequence column with the sequence scored, the SEM metadata, and the resulting scoring table. These slots are accessible with the `getRanges()`, `semData()`, and `scores()` accessor functions respectively.

The scoring table will contain the following columns:

- varId: a unique identifier for the sequence scored
- SEM: the identifier for the SEM
- rc: the orientation of the sequence (fwd or rev)
- score: the unnormalized SEM score
- scoreNorm: the SEM score normalized to it's corresponding baseline
- index: the index of the motif with the highest SEM score within the sequence scored
- seq: the motif sequence with the highest SEM scores within the sequence scored

**Examples**

```
library(VariantAnnotation)

# load default SEMs

# create a VRanges object
x <- VRanges(
  seqnames = "chr12",
  ranges = 94136009,
  ref = "G", alt = "C"
)

# calculate binding propensity
scoreVariants(x, SEMC, BSgenome.Hsapiens.UCSC.hg19::Hsapiens)
```

---

SEMC

*Default SNP Effect Matrix Data Collection*


---

**Description**

A collection of pre-computed SNP Effect Matrix objects to be used for motif scoring

**Usage**

```
SEMC
```

**Format**

SEMC:

A SNPEffectMatrixCollection object containing 223 SEMs as SNPEffectMatrix objects and a data frame with 223 rows and 13 columns containing meta data:

**transcription\_factor** Transcription factor name

**ensembl\_id** Ensembl id

**ebi\_complex\_ac**

**uniprot\_ac** Uniprot accession id

**PWM\_id** Position weighted matrix id

**SEM** SNP Effect Matrix file

**SEM\_baseline** SNP Effect Matrix baseline

**cell\_type** Cell Type

**neg\_log10\_pval** -log10(p value) from SEMpl calculation

**chip\_ENCODE\_accession** ENCODE accession for ChIP data used in SEMpl

**dnase\_ENCODE\_accession** ENCODE accession for DNase data used in SEMpl

**PWM\_source** Position weighted matrix source ...

**Source**

[https://data.igvf.org/multireport/?type=ModelSet&software\\_versions.software.title=SEMpl](https://data.igvf.org/multireport/?type=ModelSet&software_versions.software.title=SEMpl)

---

semData

*Accessor semData slot in a SEMScores object*

---

**Description**

Accessor semData slot in a SEMScores object

Access semData from a SNPEffectMatrixCollection object

**Usage**

```
semData(x)
```

```
## S4 method for signature 'SEMScores'
semData(x)
```

```
## S4 method for signature 'SNPEffectMatrixCollection'
semData(x)
```

**Arguments**

x SNPEffectMatrixCollection object

**Value**

A data.table is returned

**Examples**

```
library(VariantAnnotation)

# load default SEMs

# create a VRanges object
vr <- VRanges(
  seqnames = "chr12",
  ranges = 94136009,
  ref = "G", alt = "C"
)

# calculate binding propensity
s <- scoreVariants(vr, SEMC, BSgenome.Hsapiens.UCSC.hg19::Hsapiens)

semData(s)

## Create example SEM
df <- data.frame(
  A = c(1, 2, 3),
  C = c(1, 2, 3),
  G = c(1, 2, 3),
  T = c(1, 2, 3)
)
sem_data <- data.frame(SEM = "motif_id", TF = "tf_id")
s <- SNPEffectMatrix(df, 1.205, "motif_id")
sc <- SNPEffectMatrixCollection(s, semData = sem_data, semKey = "SEM")

## Access count matrix
semData(sc)
```

---

SEMScores

*SEMScores object and constructor*

---

**Description**

Constructs a SEMScores class object.

**Usage**

```
SEMScores(ranges = NULL, semData = NULL, scores = NULL)
```

**Arguments**

**ranges** A GRanges or VRanges object to hold one or more variants  
**semData** A named list of SNPEffectMatrix objects  
**scores** (optional) A data.table object for motif information and binding scores

**Value**

a SEMScores object

**Examples**

```

# load default SEMs

# create a VRanges object
vr <- VariantAnnotation::VRanges(
  seqnames = c("chr12", "chr19"),
  ranges = c(94136009, 10640062),
  ref = c("G", "T"), alt = c("C", "A")
)

SEMScores(ranges = vr, semData = semData(SEMC))

```

---

SEMScores-class	<i>Class for storing SEM motif binding calculations for multiple genomic ranges or variants</i>
-----------------	---

---

**Description**

Class for storing SEM motif binding calculations for multiple genomic ranges or variants

**Slots**

**ranges** A GRanges or VRanges object to hold one or more genomic ranges  
**semData** A data.table object of metadata for the SEMs with one row for each SEM  
**scores** A data.table object for motif information and binding scores

---

show,SEMScores-method *Show method for SEMScores objects*

---

**Description**

Prints information about the number of variants, SEM meta data columns, and the scoring table if scoreVariants has been run.

**Usage**

```
## S4 method for signature 'SEMScores'  
show(object)
```

**Arguments**

object            a SEMScores object

**Value**

An invisible NULL

---

show,SNPEffectMatrix-method  
*Show for SNPEffectMatrix*

---

**Description**

Show for SNPEffectMatrix

**Usage**

```
## S4 method for signature 'SNPEffectMatrix'  
show(object)
```

**Arguments**

object            SNPEffectMatrix object.

**Value**

An invisible NULL

---

show, SNPEffectMatrixCollection-method

*Show method for SNPEffectMatrixCollection objects*

---

### Description

Prints information about the number of SEMs and meta data columns included in the object.

### Usage

```
## S4 method for signature 'SNPEffectMatrixCollection'
show(object)
```

### Arguments

object            a SNPEffectMatrixCollection object

### Value

An invisible NULL

---

SNPEffectMatrix

*SNPEffectMatrix object and constructor*

---

### Description

Constructs a SNPEffectMatrix class object.

### Usage

```
SNPEffectMatrix(sem, baseline, semId)
```

### Arguments

sem            A data.table object of format base position (rows) by nucleic acid (columns). Column names must include A, C, T, and G; other columns will be ignored.

baseline      A numeric scrambled baseline, representing the binding score of randomly scrambled kmers of the same length.

semId         A character unique identifier for the SEM.

### Value

a SNPEffectMatrix object

**Examples**

```
# build a matrix for a motif of length 4
m <- matrix(rnorm(16), nrow = 4)
colnames(m) <- c("A", "C", "G", "T")

# build a SNPEffectMatrix object
SNPEffectMatrix(m, baseline = 1, semId = "sem_id")
```

---

SNPEffectMatrix-class *SNP Effect Matrix (SEM)*

---

**Description**

A SNP Effect Matrix (SEM) estimates the binding affinity of every possible mutation in a particular transcription factor (TF) binding motif. Read more about SEMs here: <https://doi.org/10.1093/bioinformatics/btz612>  
This class contains three slots: the matrix, the baseline value, and a unique id

**Slots**

`sem` The SEM itself as a data table Rows represent sequence position (variable length), columns represent effects due to each nucleotide base A, C, G, T (fixed length: 4)  
`baseline` A scrambled baseline, representing the binding score of randomly scrambled kmers of the same length. This is the binding cutoff for a TF  
`semId` basename of the sem file

**Examples**

```
# build a SEM example matrix
m <- matrix(rnorm(16), nrow = 4)
colnames(m) <- c("A", "C", "G", "T")

SNPEffectMatrix(sem = m, baseline = 1, semId = "sem_id")
```

---

SNPEffectMatrixCollection

*SNPEffectMatrixCollection object and constructor*

---

**Description**

Constructs a SNPEffectMatrixCollection class object.

**Usage**

```
SNPEffectMatrixCollection(sems, semData = NULL, semKey = "")
```

**Arguments**

sems	A list of SNPEffectMatrix objects
semData	A data.table containing meta data for each SEM
semKey	A column name in semData corresponding to the semIds in each SNPEffectMatrix object in the sems parameter

**Value**

a SNPEffectMatrixCollection object

**Examples**

```
#
# build two matrices for a motif of length 4
m1 <- matrix(rnorm(16), nrow = 4)
m2 <- matrix(rnorm(16), nrow = 4)
colnames(m1) <- c("A", "C", "G", "T")
colnames(m2) <- c("A", "C", "G", "T")

# build two SNPEffectMatrix objects
s1 <- SNPEffectMatrix(m1, baseline = 1, semId = "sem_id_1")
s2 <- SNPEffectMatrix(m2, baseline = 1, semId = "sem_id_2")

# build a meta data table
md <- data.table::data.table(
  transcription_factor = c("tf1", "tf2"),
  cell_type = c("HepG2", "K562"),
  sem_id = c("sem_id_1", "sem_id_2")
)

# build a SNPEffectMatrixCollection object
SNPEffectMatrixCollection(list(s1, s2), semData = md, semKey = "sem_id")
```

---

SNPEffectMatrixCollection-class

*A collection of SNPEffectMatrix objects and their corresponding meta data*

---

**Description**

A collection of SNPEffectMatrix objects and their corresponding meta data

**Slots**

sems A list of SNPEffectMatrix objects  
semData A data.table with meta data on each SEM

`semKey` The name of a column in `semData` that matches the `semIds` in the `sems` list as a character. If column entries have a `.sem` suffix, a new column named `SEM_KEY` will be created without the `.sem` suffixes.

### Examples

```
# build a SEM example matrix
m <- matrix(rnorm(16), nrow = 4)
colnames(m) <- c("A", "C", "G", "T")

# build a SNPEffectMatrix object
sm <- SNPEffectMatrix(sem = m, baseline = 1, semId = "sem_name")

# create a meta data table
md <- data.table::data.table(tf = "tf_name", sem = "sem_name")

# build a collection with 1 SEM
SNPEffectMatrixCollection(sems = list(sm), semData = md, semKey = "sem")
```

# Index

- \* **datasets**
  - SEMC, [21](#)
- \* **internal**
  - scores, [19](#)
  - SEMPLR-package, [3](#)
- convertSEMsToPPMs, [3](#)
- enrichmentSets, [4](#)
- enrichSEMs, [6](#)
- getBaseline, [7](#)
- getBaseline, SNPEffectMatrix-method (getBaseline), [7](#)
- getRanges, [8](#)
- getRanges, SEMScores-method (getRanges), [8](#)
- getRangeSeqs, [9](#)
- getSEM, [10](#)
- getSEM, SNPEffectMatrix-method (getSEM), [10](#)
- getSEMId, [10](#)
- getSEMId, SNPEffectMatrix-method (getSEMId), [10](#)
- getSEMs, [11](#)
- getSEMs, SNPEffectMatrixCollection-method (getSEMs), [11](#)
- loadSEMCollection, [12](#)
- plotEnrich, [13](#)
- plotSEM, [14](#)
- plotSEMMotifs, [16](#)
- plotSEMVariants, [17](#)
- scoreBinding, [18](#)
- scores, [19](#)
- scores, SEMScores-method (scores), [19](#)
- scoreVariants, [20](#)
- SEMC, [21](#)
- semData, [22](#)
- semData, SEMScores-method (semData), [22](#)
- semData, SNPEffectMatrixCollection-method (semData), [22](#)
- SEMPLR (SEMPLR-package), [3](#)
- SEMPLR-package, [3](#)
- SEMScores, [23](#)
- SEMScores-class, [24](#)
- show, SEMScores-method, [25](#)
- show, SNPEffectMatrix-method, [25](#)
- show, SNPEffectMatrixCollection-method, [26](#)
- SNPEffectMatrix, [26](#)
- SNPEffectMatrix-class, [27](#)
- SNPEffectMatrixCollection, [27](#)
- SNPEffectMatrixCollection-class, [28](#)