

# Package ‘ConsensusClusterPlus’

April 30, 2026

**Type** Package

**Imports** Biobase, ALL, graphics, stats, utils, cluster

**Title** ConsensusClusterPlus

**Version** 1.77.0

**Date** 2020-4-24

**Author** Matt Wilkerson <mdwilkerson@outlook.com>, Peter Waltman  
<waltman@soe.ucsc.edu>

**Maintainer** Matt Wilkerson <mdwilkerson@outlook.com>

**Description** algorithm for determining cluster count and membership by  
stability evidence in unsupervised analysis

**License** GPL version 2

**biocViews** Software, Clustering

**git\_url** <https://git.bioconductor.org/packages/ConsensusClusterPlus>

**git\_branch** devel

**git\_last\_commit** 61c8cc6

**git\_last\_commit\_date** 2026-04-28

**Repository** Bioconductor 3.24

**Date/Publication** 2026-04-30

## Contents

ConsensusClusterPlus . . . . .	1
<b>Index</b>	<b>5</b>

---

ConsensusClusterPlus    *run ConsensusClusterPlus*

---

## Description

ConsensusClusterPlus function for determining cluster number and class membership by stability evidence. calcICL function for calculating cluster-consensus and item-consensus.

**Usage**

```

ConsensusClusterPlus(
d=NULL, maxK = 3, reps=10, pItem=0.8, pFeature=1, clusterAlg="hc", title="untitled_consensus_cluste
innerLinkage="average", finalLinkage="average", distance="pearson", ml=NULL,
tmyPal=NULL, seed=NULL, plot=NULL, writeTable=FALSE, weightsItem=NULL, weightsFeature=NULL, verbose=F,

calcICL(res, title="untitled_consensus_cluster", plot=NULL, writeTable=FALSE)

```

**Arguments**

d	data to be clustered; either a data matrix where columns=items/samples and rows are features. For example, a gene expression matrix of genes in rows and microarrays in columns, or ExpressionSet object, or a distance object (only for cases of no feature resampling)
maxK	integer value. maximum cluster number to evaluate.
reps	integer value. number of subsamples.
pItem	numerical value. proportion of items to sample.
pFeature	numerical value. proportion of features to sample.
clusterAlg	character value. cluster algorithm. 'hc' hierarchical (hclust), 'pam' for partitioning around medoids, 'km' for k-means upon data matrix, or a function that returns a clustering. See example and vignette for more details.
title	character value for output directory. Directory is created only if plot is not NULL or writeTable is TRUE. This title can be an absolute or relative path.
innerLinkage	hierarchical linkage method for subsampling.
finalLinkage	hierarchical linkage method for consensus matrix.
distance	character value. 'pearson': (1 - Pearson correlation), 'spearman' (1 - Spearman correlation), 'euclidean', 'binary', 'maximum', 'canberra', 'minkowski' or custom distance function.
ml	optional. prior result, if supplied then only do graphics and tables.
tmyPal	optional character vector of colors for consensus matrix
seed	optional numerical value. sets random seed for reproducible results.
plot	character value. NULL - print to screen, 'pdf', 'png', 'pngBMP' for bitmap png, helpful for large datasets.
writeTable	logical value. TRUE - write output and log to csv.
weightsItem	optional numerical vector. weights to be used for sampling items.
weightsFeature	optional numerical vector. weights to be used for sampling features.
res	result of consensusClusterPlus.
verbose	boolean. If TRUE, print messages to the screen to indicate progress. This is useful for large datasets.
corUse	optional character value. specifies how to handle missing data in correlation distances 'everything', 'pairwise.complete.obs', 'complete.obs' see cor() for description.

## Details

ConsensusClusterPlus implements the Consensus Clustering algorithm of Monti, et al (2003) and extends this method with new functionality and visualizations. Its utility is to provide quantitative stability evidence for determining a cluster count and cluster membership in an unsupervised analysis.

ConsensusClusterPlus takes a numerical data matrix of items as columns and rows as features. This function subsamples this matrix according to `pItem`, `pFeature`, `weightsItem`, and `weightsFeature`, and clusters the data into 2 to `maxK` clusters by `clusterArg` clusteringAlgorithm. Agglomerative hierarchical (`hclust`) and `kmeans` clustering are supported by an option see above. For users wishing to use a different clustering algorithm for which many are available in R, one can supply their own clustering algorithm as a simple programming hook - see the second commented-out example that uses divisive hierarchical clustering.

For a detailed description of usage, output and images, see the vignette by: `openVignette()`.

## Value

ConsensusClusterPlus returns a list of length `maxK`. Each element is a list containing `consensusMatrix` (numerical matrix), `consensusTree` (`hclust`), `consensusClass` (consensus class assignments). ConsensusClusterPlus also produces images.

`calcICL` returns a list of two elements `clusterConsensus` and `itemConsensus` corresponding to cluster-consensus and item-consensus. See Monti, et al (2003) for formulas.

## Author(s)

Matt Wilkerson [mdwilkerson@outlook.com](mailto:mdwilkerson@outlook.com) Peter Waltman [waltman@soe.ucsc.edu](mailto:waltman@soe.ucsc.edu)

## References

Please cite the ConsensusClusterPlus publication, below, if you use ConsensusClusterPlus in a publication or presentation: Wilkerson, M.D., Hayes, D.N. (2010). ConsensusClusterPlus: a class discovery tool with confidence assessments and item tracking. *Bioinformatics*, 2010 Jun 15;26(12):1572-3.

Original description of the Consensus Clustering method: Monti, S., Tamayo, P., Mesirov, J., Golub, T. (2003) Consensus Clustering: A Resampling-Based Method for Class Discovery and Visualization of Gene Expression Microarray Data. *Machine Learning*, 52, 91-118.

## Examples

```
# obtain gene expression data
library(Biobase)
data(geneData)
d=geneData
#median center genes
dc = sweep(d,1, apply(d,1,median))

# run consensus cluster, with standard options
rcc = ConsensusClusterPlus(dc,maxK=4, reps=100, pItem=0.8, pFeature=1, title="example", distance="pearson", clu

# same as above but with pre-computed distance matrix, useful for large datasets (>1,000's of items)
dt = as.dist(1-cor(dc,method="pearson"))
rcc2 = ConsensusClusterPlus(dt,maxK=4, reps=100, pItem=0.8, pFeature=1, title="example2", distance="pearson", clu

# k-means clustering
rcc3 = ConsensusClusterPlus(d,maxK=4, reps=100, pItem=0.8, pFeature=1, title="example3", distance="euclidean", clu
```

```
### partition around medoids clustering with manhattan distance
rcc4 = ConsensusClusterPlus(d,maxK=4, reps=100, pItem=0.8, pFeature=1, title="example3", distance="manhattan", c

## example of custom distance function as hook:
myDistFunc = function(x){ dist(x,method="manhattan")}
rcc5 = ConsensusClusterPlus(d,maxK=4, reps=100, pItem=0.8, pFeature=1, title="example3", distance="myDistFunc", c

##example of clusterAlg as hook:
#library(cluster)
#dianaHook = function(this_dist,k){
# tmp = diana(this_dist,diss=TRUE)
# assignment = cutree(tmp,k)
# return(assignment)
#}
#rcc6 = ConsensusClusterPlus(d,maxK=6, reps=25, pItem=0.8, pFeature=1, title="example", clusterAlg="dianaHook")

## ICL
resICL = calcICL(rcc,title="example")
```

# Index

## \* **methods**

ConsensusClusterPlus, [1](#)

calcICL (ConsensusClusterPlus), [1](#)

ConsensusClusterPlus, [1](#)