

# Package ‘orclus’

March 17, 2018

**Version** 0.2-6

**Date** 2018-03-17

**Title** Subspace Clustering Based on Arbitrarily Oriented Projected  
Cluster Generation

**Author** Gero Szepannek

**Maintainer** Gero Szepannek <gero.szepannek@web.de>

**Description** Functions to perform subspace clustering and classification.

**License** GPL (>= 2)

**NeedsCompilation** no

**Repository** CRAN

**Date/Publication** 2018-03-17 22:55:34 UTC

## R topics documented:

orclass . . . . .	1
orclus . . . . .	4
predict.orclass . . . . .	7
predict.orclus . . . . .	9

<b>Index</b>	<b>11</b>
--------------	-----------

---

orclass	<i>Subspace clustering based local classification using ORCLUS.</i>
---------	---

---

## Description

Function to perform local classification where the subclasses are concentrated in different subspaces of the data.

**Usage**

```

orclass(x, ...)
## Default S3 method:
orclass(x, grouping, k, l, k0, a = 0.5, prior = NULL, inner.loops = 1,
        predict.train = "nearest", verbose = TRUE, ...)
## S3 method for class 'formula'
orclass(formula, data = NULL, ...)

```

**Arguments**

x	A matrix or data frame containing the explanatory variables. The method is restricted to numerical data.
grouping	A factor specifying the class for each observation.
formula	A formula of the form $\text{grouping} \sim x_1 + x_2 + \dots$ . That is, the response is the grouping factor and the right hand side specifies the (non-factor) discriminators.
data	Data frame from which variables specified in formula are to be taken.
k	Prespecifies the final number of clusters.
l	Prespecifies the dimension of the final cluster-specific subspaces (equal for all clusters).
k0	Initial number of clusters (that are computed in the entire data space). Must be greater than k. The number of clusters is iteratively decreased by factor a until the final number of k clusters is reached.
a	Prespecified factor for the cluster number reduction in each iteration step of the algorithm.
prior	Argument for optional specification of class prior probabilities if different from the relative class frequencies.
inner.loops	Number of repetitive iterations (i.e. recomputation of clustering and cluster-specific subspaces) while the number of clusters and the subspace dimension are kept constant.
predict.train	Character specifying whether prediction of training data should be pursued. If "nearest" the class distribution in orclus cluster assignment is used for classification.
verbose	Logical indicating whether the iteration process should be displayed.
...	Currently not used.

**Details**

For each cluster the class distribution is computed.

**Value**

Returns an object of class `orclass`.

`orclus.res` Object of class `orclus` containing the resulting clusters.

<code>cluster.posterior</code> s	Matrix of clusterwise class posterior probabilities where clusters are rows and classes are columns.
<code>cluster.priors</code>	Vector of relative cluster frequencies weighted by class priors.
<code>purity</code>	Statistics indicating the discriminability of the identified clusters.
<code>prior</code>	Vector of class prior probabilities.
<code>predict.train</code>	Prediction of training data if specified.
<code>orclass.call</code>	(Matched) function call.

**Author(s)**

Gero Szepannek

**References**

Aggarwal, C. and Yu, P. (2000): *Finding generalized projected clusters in high dimensional spaces*, Proceedings of ACM SIGMOD International Conference on Management of Data, pp. 70-81.

**See Also**

[predict.orclass](#), [orclus](#), [predict.orclus](#)

**Examples**

```
# definition of a function for parameterized data simulation
sim.orclus <- function(k = 3, nk = 100, d = 10, l = 4,
                      sd.cl = 0.05, sd.rest = 1, locshift = 1){
  ### input parameters for data generation
  # k          number of clusters
  # nk         observations per cluster
  # d          original dimension of the data
  # l          subspace dimension where the clusters are concentrated
  # sd.cl      (within cluster subspace) standard deviations for data generation
  # sd.rest    standard deviations in the remaining space
  # locshift   parameter of a uniform distribution to sample different cluster means

  x <- NULL
  for(i in 1:k){
    # cluster centers
    apts <- locshift*matrix(runif(1*k), ncol = 1)
    # sample points in original space
    xi.original <- cbind(matrix(rnorm(nk * l, sd = sd.cl), ncol=1) + matrix(rep(apts[i,], nk),
                                                                    ncol = 1, byrow = TRUE),
                        matrix(rnorm(nk * (d-l), sd = sd.rest), ncol = (d-l)))
    # subspace generation
    sym.mat <- matrix(nrow=d, ncol=d)
    for(m in 1:d){
      for(n in 1:m){
        sym.mat[m,n] <- sym.mat[n,m] <- runif(1)
      }
    }
  }
}
```

```

    }
    subspace <- eigen(sym.mat)$vectors
    # transformation
    xi.transformed <- xi.original %*% subspace
    x <- rbind(x, xi.transformed)
  }
  clids <- rep(1:k, each = nk)
  result <- list(x = x, cluster = clids)
  return(result)
}

# simulate data of 2 classes where class 1 consists of 2 subclasses
simdata <- sim.orclus(k = 3, nk = 200, d = 15, l = 4,
                    sd.cl = 0.05, sd.rest = 1, locshift = 1)

x <- simdata$x
y <- c(rep(1,400), rep(2,200))

res <- orclus(x, y, k = 3, l = 4, k0 = 15, a = 0.75)
res

# compare results
table(res$predict.train$class, y)

```

---

orclus

*Arbitrarily ORiented projected CLUster generation*


---

### Description

Function to perform subspace clustering where the clusters are concentrated in different cluster specific subspaces of the data.

### Usage

```

orclus(x, ...)
## Default S3 method:
orclus(x, k, l, k0, a = 0.5, inner.loops = 1, verbose = TRUE, ...)

```

### Arguments

x	A matrix or data frame containing the explanatory variables. The method is restricted to numerical data.
k	Prespecifies the final number of clusters.
l	Prespecifies the dimension of the final cluster-specific subspaces (equal for all clusters).
k0	Initial number of clusters (that are computed in the entire data space). Must be greater than k. The number of clusters is iteratively decreased by factor a until the final number of k clusters is reached.

<code>a</code>	Prespecified factor for the cluster number reduction in each iteration step of the algorithm.
<code>inner.loops</code>	Number of repetitive iterations (i.e. recomputation of clustering and cluster-specific subspaces) while the number of clusters and the subspace dimension are kept constant.
<code>verbose</code>	Logical indicating whether the iteration process should be displayed.
<code>...</code>	Currently not used.

### Details

The function performs ORCLUS subspace clustering (Aggarwal and Yu, 2000). Simultaneously both cluster assignments as well as cluster specific subspaces are computed. Cluster assignments have minimal euclidean distance from the cluster centers in the corresponding subspaces. As an extension to the originally proposed algorithm initialization in the full data space is done by calling `kmeans` for  $k_0$  clusters. Further, by `inner.loops` a number of repetitions during the iteration process for each number of clusters and subspace dimension can be specified. An outlier option has not been implemented. Even though increasing the initialization parameter  $k_0$  most strongly effects the computation time it should be chosen as large as possible (at least several times greater than  $k$ ).

### Value

Returns an object of class `orclus`. Its structure is similar to objects resulting from calling `kmeans`.

<code>cluster</code>	Returns the final cluster labels.
<code>centers</code>	A matrix where each row corresponds to a cluster center (in the original space).
<code>size</code>	The final number of observations in each cluster.
<code>subspaces</code>	List of matrices for projection of the data onto the cluster-specific subspaces by post-multiplication.
<code>subspace.dimension</code>	Dimension of the final subspaces.
<code>within.projenss</code>	Corresponds to <code>withinss</code> of <code>kmeans</code> objects: projected within cluster energies for each cluster.
<code>sparsity.coefficient</code>	Sparsity coefficient of the clustering result. If its value is close to 1 the subspace dimension may have been chosen too large. A small value close to 0 can be interpreted as a hint that a strong cluster structure has been found.
<code>orclus.call</code>	(Matched) function call.

### Author(s)

Gero Szepannek

### References

Aggarwal, C. and Yu, P. (2000): *Finding generalized projected clusters in high dimensional spaces*, Proceedings of ACM SIGMOD International Conference on Management of Data, pp. 70-81.

**See Also**

[predict.orclus](#)

**Examples**

```
# generate simple artificial example of two clusters
clus1.v1 <- runif(100)
clus2.v1 <- runif(100)
xample <- rbind(cbind(clus1.v1, 0.5 - clus1.v1), cbind(clus2.v1, -0.5 + clus2.v1))
plot(xample, col=rep(1:2, each=100))

# try standard kmeans clustering
kmeans.res <- kmeans(xample, 2)
plot(xample, col = kmeans.res$cluster)

# use orclus instead
orclus.res <- orclus(x = xample, k = 2, l = 1, k0 = 8, a = 0.5)
plot(xample, col = orclus.res$cluster)

# show data in cluster-specific subspaces
par(mfrow=c(1,2))
for(i in 1:length(orclus.res$size)) plot(xample %% orclus.res$subspaces[[i]],
    col = orclus.res$cluster, ylab = paste("Identified subspace for cluster",i))

### second 'more multivariate' example to play with...

# definition of a function for parameterized data simulation
sim.orclus <- function(k = 3, nk = 100, d = 10, l = 4,
    sd.cl = 0.05, sd.rest = 1, locshift = 1){
  ### input parameters for data generation
  # k          number of clusters
  # nk         observations per cluster
  # d          original dimension of the data
  # l          subspace dimension where the clusters are concentrated
  # sd.cl      (within cluster subspace) standard deviations for data generation
  # sd.rest    standard deviations in the remaining space
  # locshift   parameter of a uniform distribution to sample different cluster means

  x <- NULL
  for(i in 1:k){
    # cluster centers
    apts <- locshift*matrix(runif(1*k), ncol = 1)
    # sample points in original space
    xi.original <- cbind(matrix(rnorm(nk * l, sd = sd.cl), ncol=1) + matrix(rep(apts[i,], nk),
        ncol = 1, byrow = TRUE),
        matrix(rnorm(nk * (d-l), sd = sd.rest), ncol = (d-l)))
  }

  # subspace generation
  sym.mat <- matrix(nrow=d, ncol=d)
  for(m in 1:d){
    for(n in 1:m){
      sym.mat[m,n] <- sym.mat[n,m] <- runif(1)
    }
  }
}
```

```

    }
  }
  subspace <- eigen(sym.mat)$vectors
  # transformation
  xi.transformed <- xi.original %*% subspace
  x <- rbind(x, xi.transformed)
}
clids <- rep(1:k, each = nk)
result <- list(x = x, cluster = clids)
return(result)
}

# simulate data, you can play with different parameterizations...
simdata <- sim.orclus(k = 3, nk = 200, d = 15, l = 4,
                    sd.cl = 0.05, sd.rest = 1, locshift = 1)

# apply kmeans and orclus
kmeans.res2 <- kmeans(simdata$x, 3)
orclus.res2 <- orclus(x = simdata$x, k = 3, l = 4, k0 = 15, a = 0.75)
cat("SC: ", orclus.res2$sparsity.coefficient, "\n")

# compare results
table(kmeans.res2$cluster, simdata$cluster)
table(orclus.res2$cluster, simdata$cluster)

```

---

predict.orclass

*Subspace clustering based local classification using ORCLUS.*

---

## Description

Assigns clusters and distances and classes for new data according to the intrinsic subspace clusters of an [orclass](#) classification model.

## Usage

```
## S3 method for class 'orclass'
predict(object, newdata, type = "nearest", ...)
```

## Arguments

object	Model resulting from a call of <a href="#">orclass</a> .
newdata	A matrix or data frame to be clustered by the given model.
type	Default "nearest" computes relative class frequencies of nearest cluster as class posterior probabilities.
...	Currently not used.

**Details**

For prediction the class distribution of the "nearest" cluster is used. If `type = "fuzzywts"` cluster memberships (see e.g. Bezdek, 1981) are computed based on the cluster distances of cluster assignment by [predict.orclus](#). For orclass prediction the class distributions of the clusters are weighed using the cluster memberships of an observation.

**Value**

<code>class</code>	Vector of predicted class levels.
<code>posterior</code>	Matrix where columns contain class posterior probabilities.
<code>distances</code>	A matrix where columns are the distances to all cluster centers in the corresponding subspaces for the new data.
<code>cluster</code>	The resulting cluster labels for the new data.

**Author(s)**

Gero Szepannek

**References**

Aggarwal, C. and Yu, P. (2000): *Finding generalized projected clusters in high dimensional spaces*, Proceedings of ACM SIGMOD International Conference on Management of Data, pp. 70-81.

Bezdek, J. (1981): *Pattern recognition with fuzzy objective function algorithms*, Kluwer Academic, Norwell, MA.

**See Also**

[orclass](#), [orclus](#), [predict.orclus](#)

**Examples**

```
# definition of a function for parameterized data simulation
sim.orclus <- function(k = 3, nk = 100, d = 10, l = 4,
                      sd.cl = 0.05, sd.rest = 1, locshift = 1){
  ### input parameters for data generation
  # k          number of clusters
  # nk         observations per cluster
  # d          original dimension of the data
  # l          subspace dimension where the clusters are concentrated
  # sd.cl      (within cluster subspace) standard deviations for data generation
  # sd.rest    standard deviations in the remaining space
  # locshift   parameter of a uniform distribution to sample different cluster means

  x <- NULL
  for(i in 1:k){
    # cluster centers
    apts <- locshift*matrix(runif(1*k), ncol = 1)
    # sample points in original space
    xi.original <- cbind(matrix(rnorm(nk * l, sd = sd.cl), ncol=1) + matrix(rep(apts[i,], nk),
```



```

                                ncol = 1, byrow = TRUE),
                                matrix(rnorm(nk * (d-1), sd = sd.rest), ncol = (d-1)))
# subspace generation
sym.mat <- matrix(nrow=d, ncol=d)
for(m in 1:d){
  for(n in 1:m){
    sym.mat[m,n] <- sym.mat[n,m] <- runif(1)
  }
}
subspace <- eigen(sym.mat)$vectors
# transformation
xi.transformed <- xi.original %%% subspace
x <- rbind(x, xi.transformed)
}
clids <- rep(1:k, each = nk)
result <- list(x = x, cluster = clids)
return(result)
}

# simulate data of 2 classes where class 1 consists of 2 subclasses
simdata <- sim.orclus(k = 3, nk = 200, d = 15, l = 4,
                    sd.cl = 0.05, sd.rest = 1, locshift = 1)

x <- simdata$x
y <- c(rep(1,400), rep(2,200))

res <- orclass(x, y, k = 3, l = 4, k0 = 15, a = 0.75)
prediction <- predict(res, x)

# compare results
table(prediction$class, y)

```

---

predict.orclus

*Arbitrarily ORiented projected CLUster generation*


---

## Description

Assigns clusters and distances to cluster centers in the corresponding subspaces for new data according to a subspace clustering model of class [orclus](#).

## Usage

```
## S3 method for class 'orclus'
predict(object, newdata, ...)
```

## Arguments

object	Model resulting from a call of <a href="#">orclus</a> .
newdata	A matrix or data frame to be clustered by the given model.
...	Currently not used.

**Value**

`distances` A matrix where columns are the distances to all cluster centers in the corresponding subspaces for the new data.

`cluster` The resulting cluster labels for the new data.

**Author(s)**

Gero Szepannek

**References**

Aggarwal, C. and Yu, P. (2000): *Finding generalized projected clusters in high dimensional spaces*, Proceedings of ACM SIGMOD International Conference on Management of Data, pp. 70-81.

**See Also**

[orclus](#)

**Examples**

```
# generate simple artificial example of two clusters
clus1.v1 <- runif(100)
clus2.v1 <- runif(100)
xample <- rbind(cbind(clus1.v1, 0.5 - clus1.v1), cbind(clus2.v1, -0.5 + clus2.v1))

orclus.res <- orclus(x = xample, k = 2, l = 1, k0 = 8, a = 0.5)

# generate new data and predict it using the
newclus1.v1 <- runif(100)
newclus2.v1 <- runif(100)
true.clusterids <- rep(1:2, each = 100)
xample2 <- rbind(cbind(newclus1.v1, 0.5 - newclus1.v1),
                cbind(newclus2.v1, -0.5 + newclus2.v1))

orclus.prediction <- predict(orclus.res, xample2)
table(orclus.prediction$cluster, true.clusterids)
```

# Index

\*Topic **classif**

orclass, 1  
orclus, 4  
predict.orclass, 7  
predict.orclus, 9

\*Topic **cluster**

orclass, 1  
orclus, 4  
predict.orclass, 7  
predict.orclus, 9

\*Topic **multivariate**

orclass, 1  
orclus, 4  
predict.orclass, 7  
predict.orclus, 9

kmeans, 5

orclass, 1, 7, 8  
orclus, 3, 4, 8-10

predict.orclass, 3, 7  
predict.orclus, 3, 6, 8, 9  
print.orclass (orclass), 1  
print.orclus (orclus), 4