clusterlab

Christopher R John

2018-10-03

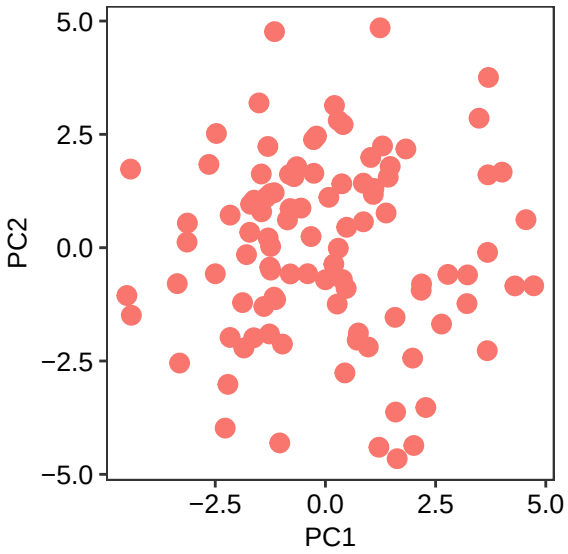
Clustering is a central task in big data analyses and clusters are often Gaussian or near Gaussian. However, a flexible Gaussian cluster simulation tool with precise control over the size, variance, and spacing of the clusters in NXN dimensional space does not exist. This is why we created clusterlab. The algorithm first creates X points equally spaced on the circumference of a circle in 2D space. These form the centers of each cluster to be simulated. Additional samples are added by adding Gaussian noise to each cluster center and concatenating the new sample co-ordinates. Then if the feature space is greater than 2D, the generated points are considered principal component scores and projected into N dimensional space using linear combinations using fixed eigenvectors. Through using vector rotations and scalar multiplication clusterlab can generate complex patterns of Gaussian clusters and outliers. Clusterlab is highly customizable and well suited to testing class discovery tools across a range of fields.

Contents

1. Simulating a single cluster

Here we simulate a 100 sample cluster with the default number of features (500). The standard deviation is left to default which is 1.

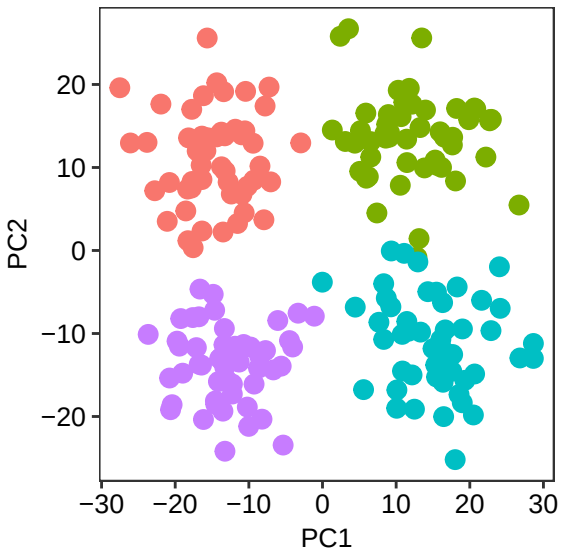
```
library(clusterlab)
synthetic <- clusterlab(centers=1,numbervec=100,pcafontsize=10)
#> ***clusterlab***
#> mode: circle
#> simulating clusters...
#> user has not set standard deviation of clusters, setting automatically...
#> user has not set alphas of clusters, setting automatically...
#> finished.
```



2. Simulating four clusters with equal variances

Next, we simulate a 4 cluster dataset with a radius of 8 for the circle on which the centers are placed. Then the standard deviations of the cluster are the same, 2.5. We set the alphas to 1, which is the value the clusters are pushed apart from one another. So there are two ways to seperate the clusters, either by the radius of the circle, or by the alpha parameter.

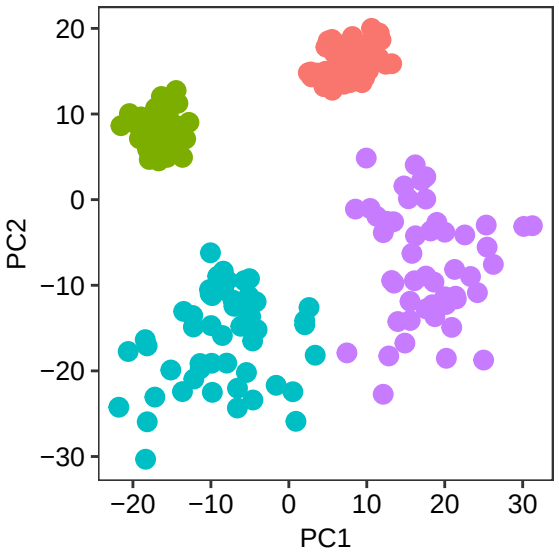
```
library(clusterlab)
synthetic <- clusterlab(centers=4,r=8,sdvec=c(2.5,2.5,2.5,2.5),
                        alphas=c(1,1,1,1),centralcluster=FALSE,
                        numbervec=c(50,50,50,50),pcafontsize=10)
#> ***clusterlab***
#> mode: circle
#> simulating clusters...
#> finished.
```



3. Simulating four clusters with unequal variances

The same as above, but 2 clusters have different variances to the other 2.

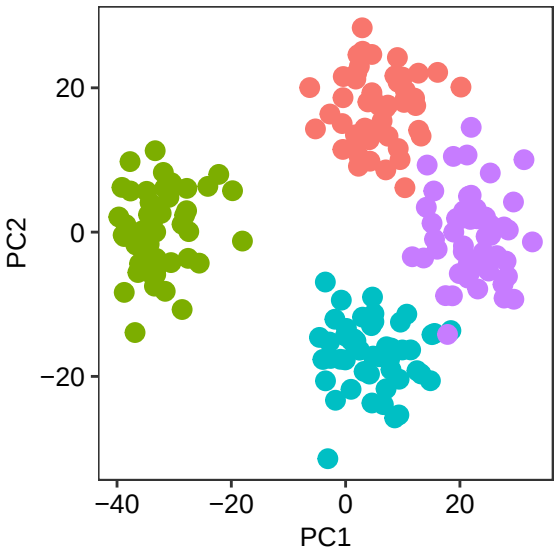
```
library(clusterlab)
synthetic <- clusterlab(centers=4,r=8,sdvec=c(1,1,2.5,2.5),
                        alphas=c(1,1,1,1),centralcluster=FALSE,
                        numbervec=c(50,50,50,50),pcafontsize=10)
#> ***clusterlab***
#> mode: circle
#> simulating clusters...
#> finished.
```



4. Simulating four clusters with one cluster pushed to the outside

The alpha parameter allows any number of clusters to be pushed away from the others. Here 1 cluster is pushed away slightly.

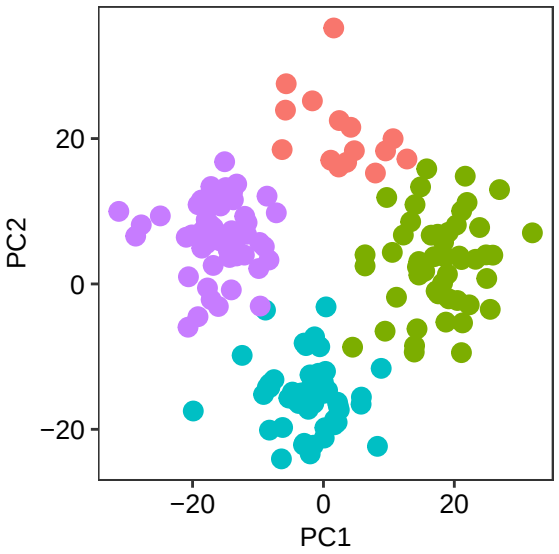
```
library(clusterlab)
synthetic <- clusterlab(centers=4,r=8,sdvec=c(2.5,2.5,2.5,2.5),
                        alphas=c(1,2,1,1),centralcluster=FALSE,
                        numbervec=c(50,50,50,50),pcafontsize=10)
#> ***clusterlab***
#> mode: circle
#> simulating clusters...
#> finished.
```



5. Simulating four clusters with one small cluster

Here we change the number vec entry for 1 cluster to a smaller value, therefore lowering the number of samples in the specified cluster.

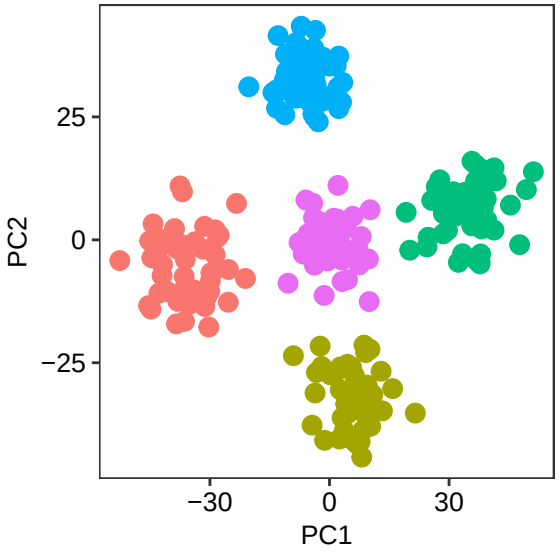
```
library(clusterlab)
synthetic <- clusterlab(centers=4,r=8,sdvec=c(2.5,2.5,2.5,2.5),
                        alphas=c(1,1,1,1),centralcluster=FALSE,
                        numbervec=c(15,50,50,50),pcafontsize=10)
#> ***clusterlab***
#> mode: circle
#> simulating clusters...
#> finished.
```



6. Simulating five clusters with one central cluster

In this case we change the centralcluster parameter to TRUE, in order to make a central cluster as well as those placed on the circumference.

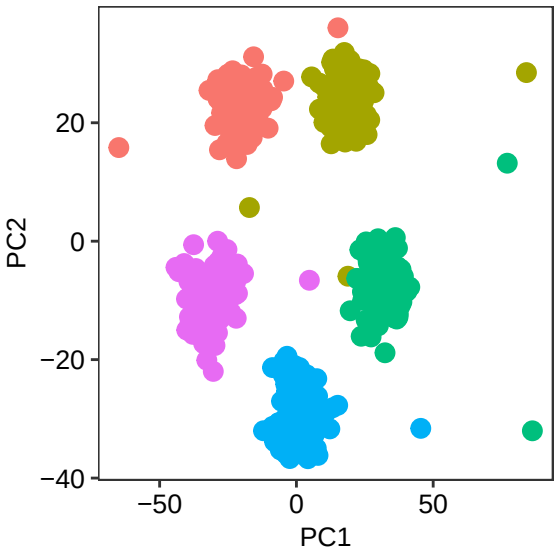
```
library(clusterlab)
synthetic <- clusterlab(centers=5,r=8,sdvec=c(2.5,2.5,2.5,2.5,2.5),
                        alphas=c(2,2,2,2,2),centralcluster=TRUE,
                        numbervec=c(50,50,50,50,50),pcafontsize=10)
#> ***clusterlab***
#> mode: circle
#> simulating clusters...
#> finished.
```



7. Simulating five clusters with ten outliers

Here we add ten outliers using the outliers parameter and a distance to move them by of 50. The angle chosen to transform the original coordinates is randomly generated by clusterlab internally.

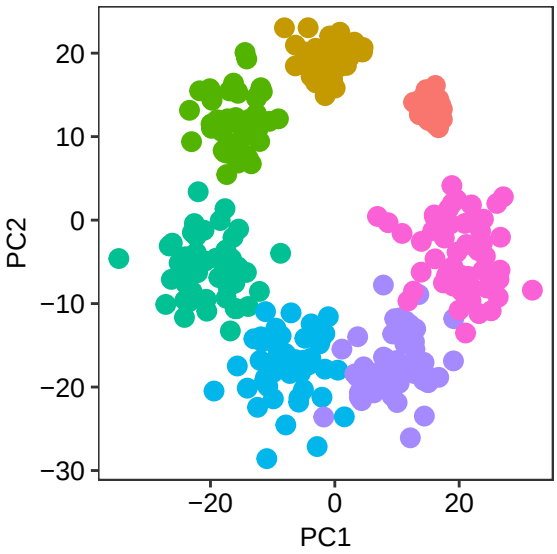
```
library(clusterlab)
synthetic <- clusterlab(centers=5,r=7,sdvec=c(2,2,2,2,2),
                        alphas=c(2,2,2,2,2),centralcluster=FALSE,
                        numbervec=c(50,50,50,50), seed=123, outliers=10,
                        outlierdist=20, pcafontsize=10)
#> ***clusterlab***
#> mode: circle
#> simulating clusters...
#> user has not set length of numbervec equal to number of clusters, setting automatically...
#> we are generating outliers...
#> finished.
```



8. Simulating six clusters with different variances

Setting the variance here with the sdvec parameter.

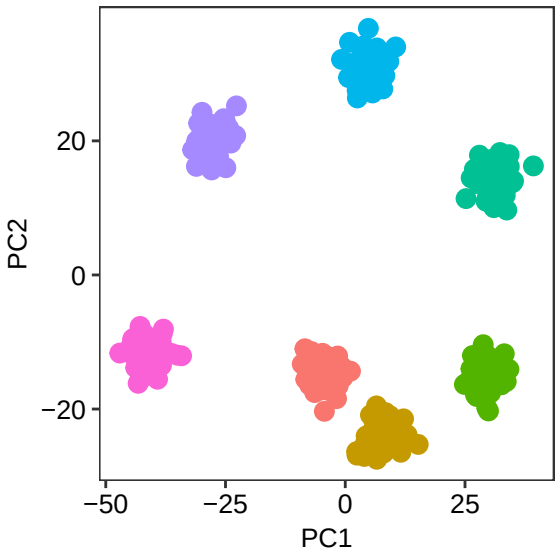
```
library(clusterlab)
synthetic <- clusterlab(centers=7,r=9,sdvec=c(0.5,1,1.5,1.75,1.85,1.95,2.05),
                        numbervec=c(50,50,50,50,50,50,50), seed=123,
                        pcafontsize=10)
#> ***clusterlab***
#> mode: circle
#> simulating clusters...
#> user has not set alphas of clusters, setting automatically...
#> finished.
```



9. Simulating six clusters with different push apart degrees

We set the push apart degree with the alphas parameter.

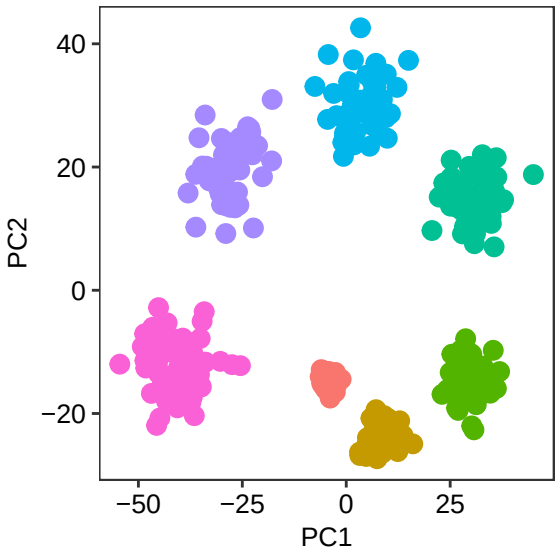
```
library(clusterlab)
synthetic <- clusterlab(centers=7,r=9,alphas=c(0.5,1,1.5,1.75,1.85,1.95,2.05),
                        numbervec=c(50,50,50,50,50,50,50), seed=123,
                        pcafontsize=10)
#> ***clusterlab***
#> mode: circle
#> simulating clusters...
#> user has not set standard deviation of clusters, setting automatically...
#> finished.
```



10. Simulating six clusters with different push apart degrees and variances

Setting the push apart degree and variance of the clusters allows a more complex structure.

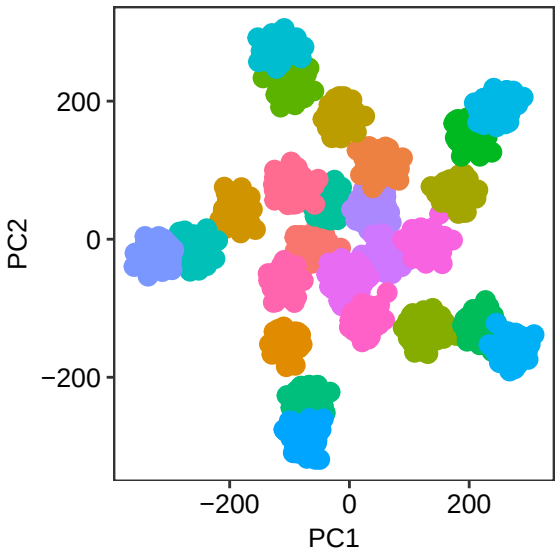
```
library(clusterlab)
synthetic <- clusterlab(centers=7,r=9,alphas=c(0.5,1,1.5,1.75,1.85,1.95,2.05),
                        sdvec=c(0.5,1,1.5,1.75,2,2.25,2.25),
                        numbervec=c(50,50,50,50,50,50,50), seed=123,
                        pcafontsize=10)
#> ***clusterlab***
#> mode: circle
#> simulating clusters...
#> finished.
```



11. Generating more complex multi-ringed structures

The ringthetas parameter may be used to rotate each ring individually. Through rotating the clusters complex patterns may be formed.

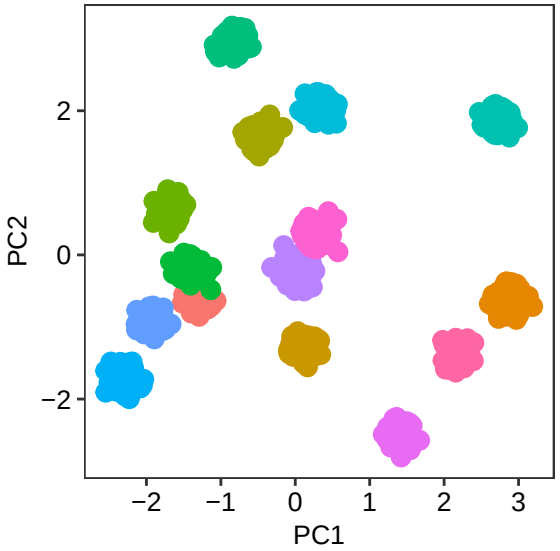
```
library(clusterlab)
synthetic <- clusterlab(centers=5,r=7,sdvec=c(6,6,6,6,6),
                        alphas=c(2,2,2,2,2),centralcluster=FALSE,
                        numbervec=c(50,50,50,50),rings=5,ringalphas=c(2,4,6,8,10,12),
                        ringthetas = c(30,90,180,0,0,0), seed=123,
                        pcafontsize=10) # for a six cluster solution)
#> ***clusterlab***
#> mode: circle
#> simulating clusters...
#> user has not set length of numbervec equal to number of clusters, setting automatically...
#> we are generating clusters arranged in rings...
#> finished.
```



12. Simulating randomly spaced Gaussian clusters

A simpler option is just to simulate randomly spaced Gaussian clusters without controlled spacing. A minimum distance can be specified, however, that uses a Monte Carlo method to space out the clusters. This is very similar to the Scikit-learn make.blobs function.

```
library(clusterlab)
synthetic <- clusterlab(mode='random',centers=15,pcafontsize=10)
#> ***clusterlab***
#> mode: random
#> simulating clusters...
#> user has not set standard deviation of clusters, setting automatically...
#> user has not set length of numbervec equal to number of clusters, setting automatically...
#> finished.
```



13. Keeping track of cluster allocations

Clusterlab also keeps track of the cluster allocations and gives each sample an unique ID. This may prove useful when scoring class discovery algorithms assignments.

```
head(synthetic$identity_matrix)
#>   sampleID cluster
#> 1     c1s1       1
#> 2     c2s1       2
#> 3     c3s1       3
#> 4     c4s1       4
#> 5     c5s1       5
#> 6     c6s1       6
```

14. Closing comments

We have seen how the clusterlab package may generate NXN Gaussian clusters in a flexible manner. Clusterlab was developed for the testing of class discovery algorithms on high dimensional genome wide expression data. For class discovery of this type of data we recommend clusterlab's sister package, M3C which was developed

in parallel. M3C has been extensively tested on high dimensional Gaussian clusters. M3C is available on the Bioconductor (https://bioconductor.org/packages/devel/bioc/html/M3C.html). Thanks for using clusterlab.