akmeans-package

Adaptive K-means

Description

This package provides two things. At first, K-means based on cosine distance is provided as a component of this package. Then, based on existing 'kmeans' function and the new kmeans with cosine distance, adaptive part is implemented using various thresholds.

Details

- akmeans - the main function, adaptive kmeans algorithm
- norm.sim.ksc - kmeans based on cosine distance

Author(s)

Jungsuk Kwac <kwjusu1@stanford.edu>

Usage

akmeans(x, ths1 = 0.2, ths2 = 0.2, ths3 = 0.7, ths4 = 0.2, min.k = 5, max.k = 100, iter.max = 100, nstart = 1, mode = 1, d.metric = 1, verbose = TRUE)
akmeans

Arguments

- **x**: data matrix n by p: all elements should be numeric
- **ths1**: threshold to decide whether to increase k or not: check $\sum((\text{sample-assigned center})^2) < \text{ths1} \sum(\text{assigned center}^2)$
- **ths2**: threshold to decide whether to increase k or not: check all components of $|\text{sample-assigned center}| < \text{ths2}$
- **ths3**: threshold to decide whether to increase k or not: check inner product of $(\text{sample,assigned center}) > \text{ths3}$, this is only for cosine distance metric
- **ths4**: threshold to decide whether to increase k or not: check all components of $\sum(\text{abs(sample-assigned center)}) < \text{ths4}$
- **min.k**: minimum number of clusters, starting k
- **max.k**: maximum number of clusters
- **iter.max**: will be delivered to kmeans function
- **nstart**: will be delivered to kmeans function
- **mode**: 1: use ths1, 2: use ths2, 3: use ths3
- **d.metric**: 1: use euclidean distance metric, otherwise use cosine distance metric
- **verbose**: print the messages or not

Details

```
## ths1: threshold to decide whether to increase k or not: check $\sum((\text{sample-assigned center})^2) < \text{ths1} \sum(\text{assigned center}^2)$
## ths2: threshold to decide whether to increase k or not: check all components of $|\text{sample-assigned center}| < \text{ths2}$
## ths3: threshold to decide whether to increase k or not: check inner product of $(\text{sample,assigned center}) > \text{ths3}$, this is only for cosine distance metric
## ths4: threshold to decide whether to increase k or not: check all components of $\sum(\text{abs(sample-assigned center)}) < \text{ths4}$
```

Value

if d.metric=1, it will return the same result as 'kmeans' function. if d.metric is not 1, a list will be returned with components: cluster: A vector of integers indicating the cluster to which each point is allocated. centers: A matrix of cluster centres size: The number of points in each cluster

Author(s)

Jungsuk Kwac

Examples

- `x = matrix(rnorm(1000),100,10)`
  - `akmeans(x) # euclidean distance based`
  - `akmeans(x,d.metric=2,ths3=0.8,mode=3) # cosine distance based`
Description

On the assumption that the two samples are already normalized to have L2 norm as 1, cosine distance is defined as 1 - inner product of the two samples.

Usage

\[
\text{norm.sim.ksc}(A, k, \text{init.cen} = \text{NULL}, \text{init.mem} = \text{NULL}, \text{iter.max} = 100)
\]

Arguments

- **A**: n by p matrix, each row is a sample
- **k**: the number of clusters
- **init.cen**: initial cluster centers
- **init.mem**: initial cluster member assignment
- **iter.max**: the maximum number of iteration

Value

A list will be returned with components: cluster: A vector of integers indicating the cluster to which each point is allocated. centers: A matrix of cluster centres size: The number of points in each cluster

Author(s)

Jungsuk Kwac

Examples

```
###############
## test code
## T classes: a1, a2, a3, a4
## for each class, R0 samples
###############
n = R0; p = 3R
a1 = 10*sin(0.1*(1:p))
a2 = 10*cos(0.1*(1:p))+10
a3 = c(1:(p/2),(p/2):1)
a4 = c((p/2):1,1:(p/2))
A = c()
for (i in 1:n){
  A = rbind(A,a1+rnorm(p),a2+rnorm(p),a3+rnorm(p),a4+rnorm(p))
}
res = norm.sim.ksc(quick.norm(A,1),4)
```
**norm.sim.ksc.center.update**

*Internal function in norm.sim.ksc*

**Description**

Internal function in norm.sim.ksc

**Usage**

```r
norm.sim.ksc.center.update(mem, A, k, cur.center = NULL)
```

**Arguments**

- `mem`
- `A`
- `k`
- `cur.center`

**Author(s)**

Jungsuk Kwac

---

**quick.norm**

*normalization function*

**Description**

it is normalizing each row to have L2 norm as 1 or sum as 1

**Usage**

```r
quick.norm(A, mod = 2)
```

**Arguments**

- `A` Input matrix, n by p
- `mod` 1: make each row has L2 norm as 1 2: make each row has sum as 1

**Value**

A normalized n by p matrix will be returned
Author(s)
Jungsuk Kwac

Examples
quick.norm(matrix(rnorm(9),3,3))
quick.norm(matrix(rnorm(9),3,3), mod=1)
Index

*Topic adaptive kmeans
  akmeans, 2
  akmeans, 2
  akmeans-package, 2
  norm.sim.ksc, 4
  norm.sim.ksc.center.update, 5
  quick.norm, 5