kMeans

Version 1.1, user's manual.

Software for Amova-based clustering

of population genetic data

Patrick Meirmans

I.B.E.D. Universiteit van Amsterdam

p.g.meirmans@uva.nl

http://www.patrickmeirmans.com/software/

Introduction

K-Means clustering divides a number of objects into a a priori assigned number (*k*) of groups in such a way that the among-groups Sum of Squares is maximised. This program can perform the clustering on genetic marker data either based on the allele frequencies or using an Analysis of Molecular Variance.

This manual starts with some background information on the subject. If you don't like reading manuals (who does?), you can skip the background and proceed to the "Input file" and "Simple Mode" sections. If you are familiar with the command line and wish to try out more esoteric settings, you should read the section "Command line Mode".

If you are using this on a Mac, I suggest you use my program GenoDive (Meirmans & Van Tienderen 2004) instead, which can perform the exact same clustering with a more user-friendly interface. GenoDive can be downloaded from my website: http://www.patrickmeirmans.com/software/.

If you have any questions about the program or the method, bug-reports, or praise, please don't hesitate to contact me at: p.g.meirmans@uva.nl.

Good luck, Patrick Meirmans

K-Means clustering (background)

The method uses a pairwise matrix of distances between all observations. Given a certain clustering into *k* groups, for every group the within-group Sum of Squares is calculated by taking the sum of the squared within-group distances. When the distances are Euclidean, this is equivalent to calculating the sum of the squared distances from the points to the group's centroid. The Error Sum of Squares is then found by summing over groups. The amount of variance explained by the grouping is then calculated by dividing the Error Sum of Squares by the Total Sum of Squares.

Because the method uses a matrix of distances, it is very flexible as it can be used with many different types of distance indices. For genetic data, a very straightforward method is to use it with a matrix of Euclidean distances based on within-population or within-individual allele frequencies. On the other hand, the method of calculating the Sums of Squares is very similar to that used by an Analysis of Molecular Variance (Excoffier et al., 1992), so it is also possible to use the Sums of Squares from an AMOVA to perform the clustering. Because of it flexibility, the method can be performed either at the individual-level or at the populations level. At the individual level, individuals are clustered into putative populations, equivalent to maximising F_{st}. At the population level, populations are clustered into groups, equivalent maximising F_{ct}.

Using hill-climbing

The classical method to perform K-Means clustering was first developed by MacQueen (1967), and is implemented in many statistical programs. The analysis starts by assigning every observation at random to one of the *k* groups and then calculates the Error Sum of Squares. A new clustering is then made by removing one by one each observation and placing it into the group to which centroid it is closest. This process is repeated and every iteration the new clustering will have a smaller Error Sum of Squares, until at some point convergence is reached. The problem with this method is that it can only climb uphill, so it is very likely to get stuck at a local optimum. Therefore the whole procedure is repeated a number of times (say 100) and the best solution is taken from among those repeats. For small datasets, this is a very fast and useful method. However, for very complex problems such as a dataset of many populations genotyped at many highly variable loci, the size of the solution space is huge and may contain many millions of local optima. The chance that the overall minimum is found is then very small, even when a large number of random starts is used.

Using Simulated Annealing

Simulated annealing uses a Monte Carlo Markov Chain (MCMC) that prevents the clustering from getting stuck in local optima. The algorithm has many similarities with the hill-climbing approach, but has some important differences. It again starts by randomly assigning observations to the *k* clusters. From that clustering, the MCMC is then started which lasts a certain number of user-defined steps. In every step of the MCMC, a random observation is selected and placed in a different, randomly picked, cluster. If the clustering is better than before, i.e. the Error Sum of Squares decreases, the new situation is accepted. However, there is also a chance that the new situation is accepted if the clustering is worse than it was before. This chance depends on the difference in the Error Sum of Squares and on what is called the "temperature" of the chain. This temperature gradually decreases during the run. Initially, the temperature is high and almost all new clusterings are accepted. In the end, when the temperature reaches zero, only the changes that improve the clustering are accepted. The real power of the method lies in the middle, where most of the climbing is uphill, but every now and then a valley is crossed.

One problem with this method is choosing a suitable starting temperature, if this is either too high or too low, the middle part gets very short and the clustering is suboptimal. As a workaround, kMeans implements an adaptive starting temperature. A very high value is used as initial starting temperature, the as long as the acceptance rate is too high, the starting temperature is lowered iteratively until a suitable value is reached. Despite the improved performance relative to the hill-climbing algorithm, multiple repeats may be necessary for the simulated annealing in order to find the overall optimum.

Which method to use?

The hill-climbing algorithm is usually much faster than the simulated annealing and works rather well for small datasets. However, for datasets with a large number of individuals or populations (depending on what is being clustered), the simulated annealing approach usually returns better results, provided that a suitably large number of steps is used (at least 50.000 is a minimum) and the method is repeated several times. In general, I would use both and see which gives the best results. Also try several runs to see whether these give wildly different results.

Optimal value of k

Most often, the number of clusters will not be known a priori. In that case, it is possible to set a range of values for *k* and perform the clustering for all values. However, this introduces the problem of determining the best clustering among all values of *k*. The percentage of explained variance is unsuitable for this task as it tends to increase with increasing *k*. The kMeans programs provides four different statistics that can determine the number of clusters. The first is the Calinski-Harabasz (1974) pseudo-F-statistic; the optimal clustering is the one with the highest value for the pseudo-f statistic. The second is the Akaike Information Criterion (AIC, Akaike, 1974); here, the optimal clustering is the one with the lowest value of AIC. The third is the Bayesian Information Criterion (BIC, Schwarz, 1978); again, the optimal clustering is the one with the lowest value. The fourth is the Gap-statistic (Tibshirani et al. 2002), which uses a number of generated datasets to obtain a baseline to help comparing which number of clusters is the best.

Simulations have revealed that BIC and pseudo-F work well for clustering populations and individuals, especially when there is random mating within populations (Meirmans *in prep*., Jombart et al. 2010). However, pseudo-F works slightly better for clustering individuals and when there is non-random mating. On the other hand, BIC can be calculated for a single cluster (k=1), whereas pseudo-F can only be calculated for two or more clusters ($k \geq 2$). Therefore, BIC has the benefit that it can be used to determine whether there actually is any population structure at all. Both AIC and the Gap statistic give much worse results than pseudo-F and BIC, so I advice against their use

Input file

kMeans can only cluster genetic data of an allelic nature, for example microsatellites, AFLPs, or SNPs. The data needs to be presented in the same format as the program Fstat (Goudet, 1995). This should be a tab-delimited plain text file, with the following format:

- The first line contains four numbers: 1) The number of populations, 2) The number of loci, 3) The maximum number used to code an allele (this is required by Fstat, but not used by kMeans), 4) The number of digits used to code alleles (1, 2, or 3).
- The following lines contain the locus names, one line per locus. The names should not contain any spaces.
- The following lines contains the individual data: first the population (as a number) then per locus the diploid genotype at that locus. The genotype should be coded with the specified number of digits, using zeroes to fill in any gaps. So an individual heterozygous for the alleles 3 and 4 can be coded as 34, 0304, or 003004, depending on whether 1, 2, or 3 digits are used per allele. Missing alleles should be coded as 0 (or 00, or 000).

Example input file

```
2    3    99   2
loc-1
loc-2
loc-3
1    2424 2424 2727
1    2424 4502 0101
1    2424 4502 2701
1    2424 0232 0000
2    2446 0202 2701
2    2424 1717 2727
2    2424 1702 2701
2    5050 0202 0127
```

Simple mode

For normal purposes, kMeans is pretty simple to use, despite the lack of a graphical interface. Just double-click the executable and you will be asked to give the name of the input file. This name should be the *absolute path* of the file, so including the full location on the hard drive. On a Mac you can simply drag the file from the Finder onto the terminal. When the filename is valid, a few questions more questions are asked, and then you will be asked to give the name of the output-file (also here, use an absolute path). Then the clustering will start, and all results will be written to the outfile, with very minimal information provided via the terminal.

Settings to use

For most uses, both for clustering individuals or populations, I recommend the following settings:

- A maximum number of clusters of 15 (the program will automatically scale this down if this number is not possible).
- AMOVA for calculating the distances.
- Pseudo-F for determining the optimal number of clusters.
- Simulated annealing as a clustering algorithm.
- 50000 steps for the simulated annealing chain.
- 10 repeats of the algorithm (and do it again with many more repeats if the clustering is fast enough).

Command line mode

It is also possible to run kMeans from the command line. The settings should then be given as arguments. Every setting has its own flag, which should be followed by the value for that setting (see Table 1). Note that you always have to specify at least one setting, otherwise the program will start in Simple Mode as described above. The output will not be written to a file but to the terminal (which you can redirect to a file if you want, using >). For example, to perform a clustering of populations from the file “in.dat”, using the settings suggested above you can type into the terminal (Mac OS X & Linux):

```
./kMeans -f in.dat -a 1 -k 15 -n 50000 -r 100 > out.txt
```

Table 1. Command line arguments for kMeans settings

Flag	Type	Setting	Default	Description
-f	string	file name	in.dat	The name of the input file. The file should be a tab-delimited plain text file in Fstat format.
-m	int	optimality statistic	0	The statistic that is used for determining the number of clusters: 0 = Pseudo-F, 1 = AIC, 2 = BIC, 3 = Gap-statistic, 4 = PCA-based Gap-statistic
-l	int	level	1	The level at which to perform the clustering: 0 = individuals, 1 = populations
-a	int	distance	0	The method used for calculating the distances: 0 = frequencies, 1 = AMOVA
-s	int	start k	2	The minimum number of clusters to try, for AIC or BIC k=1 is anyway always done
-k	int	end k	0	The maximum number of clusters to try. When the value is too high or 0, the maximum will be set to half the number of objects to cluster
-u	int	algorithm	1	The algorithm to use for clustering: 0 = hill climbing, 1 = simulated annealing, 2 = both
-n	int	steps	10000	When the simulated annealing algorithm is used, this sets the number of steps in the chain
-r	int	repeats	100	The number of times the algorithm is repeated, not used for the Gap-statistic
-p	int	permutatio ns	100	The number of permutations for generating the Gap-statistic reference distribution
Advanced settings				
-t	double	start temp	10000	The initial starting temperature for the adaptive heating, should be very high.
-c	int	steps to check	20	The number of steps to look back for checking whether to lower the starting temperature.
-d	double	acceptance threshold	0.75	For adaptive heating, when the acceptance rate over the last steps is lower than this rate the starting temperature is lowered.
-i	double	reduction	0.5	The fraction by which the starting temperature will be lowered when the acceptance rate is too high
-b	double	burn-in	0	A fraction of the total chain length can be devoted to a short prerun, though there is no clear advantage to this (0.0)
-v	int	verbose output	0	Verbosity of the output 0 = all output, 1 = only the optimal k, 2 = only the table with Sums of Squares, 3 = only the cluster assignments (0)
-w	int	loci to use	-1	The number of loci to use, a value of -1 will simple include all loci.

References

Akaike H (1974). A new look at the statistical model identification. *IEEE transactions on automatic control,* **19:** 716-723.

Calinski R, Harabasz J (1974). A dendrite method for cluster analysis. *Communications in Statistics,* **3:** 1-27.

Excoffier L, Smouse PE, Quattro JM (1992). Analysis of molecular variance inferred from metric distances among DNA haplotypes - application to human mitochondrial-DNA restriction data. *Genetics,* **131:** 479-491.

Jombart T, Devillard S, Balloux, F (2010). Discriminant analysis of principal components: a new method for the analysis of genetically structured populations. *BMC Genetics* **11:**94.

Goudet J (1995). FSTAT (Version 1.2): A computer program to calculate F-statistics. *Journal of Heredity,* **86:** 485-486.

MacQueen JB (1967). Some methods for classification and analysis of multivariate observations. *Proceedings of the fifth Berkeley symposium on mathematics, statistics, and probability, Volume 1.* University of California Press, Berkeley, 281-297.

Meirmans PG, Van Tienderen PH (2004). GENOTYPE and GENODIVE: two programs for the analysis of genetic diversity of asexual organisms. *Molecular Ecology Notes,* **4:** 792-794.

Schwarz, GE (1978). Estimating the dimension of a model. *Annals of Statistics* **6:** 461–464.

Tibshirani R, Walther G, Hastie T (2001). Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society. Series B,* **63:** 411-423.