Flags

/Z Does not generate an error if the specified window does not exist.

See Also

The **DoWindow** operation.

KMeans

KMeans [***flags***] ***populationWave***

The KMeans operation analyzes the clustering of data in *populationWave* using an iterative algorithm. The result of KMeans is a specification of the classes which is saved in the wave M_KMClasses in the current data folder. Optional results include the distribution of class members (W_KMMembers) and the inter-class distances. *populationWave* is a 2D wave in which columns correspond to members of the population and rows contain the dimensional information.

Flags

/CAN Analyzes the clustering by computing Euclidean distances between the means of the resulting classes. The resulting distances are stored in an NxN square matrix where N is the number of classes. Self distances (along the diagonal) or distances involving classes that did not survive the iterations are filled with NaN. Also saves the wave W_KMDispersion, which contains the sum of the distances between the center of each class and all its members. Distances are evaluated using the method specified by /DIST.

/DEAD=*method* Specifies how the algorithm should handle "dead" classes, which are those that lose all members in a given iteration.

- *method*=1: Remove the class if it looses all members.
- *method*=2: Default; keeps the last value of the mean vector in case the class might get new members in a subsequent iteration.
- *method*=3: Assigns the class a random mean vector.

/DIST=*mode* Specifies how the class distances are evaluated.

- *mode*=1: Distance is evaluated as the sum of the absolute values (also known as Manhattan distance).
- *mode*=2: Default; distance is evaluated as Euclidean distance.

/INIT=*method* Specifies the initialization method.

- *method*=1: Random assignment of members of the population to a class.
- *method*=2: User-specified mean values (/INW).
- *method*=3: Default; initialize classes using values of a random selection from the population.

/INW=*iWave* Sets the initial classes. The number of rows of *iWave* equals the dimensionality of the class and the number of columns of *iWave* is the number of classes. For example, if we want to initialize 5 classes in a problem that involves position in two dimensions then *iWave* must have 2 rows and 5 columns. The number of rows must also match the number of rows in *populationWave*.

/NCLS=*num* Sets the number of classes in the data. If the initialization method uses specific means (/INIT=2) then the number of columns of *iWave* (see /INW) must match *num*. The default number of classes is 2.

/OUT=*format* Specifies the format for the results.

format=1: Output only the specification of the classes in the 2D wave M_KMClasses (default). Each column in M_KMClasses represents a class. The number of rows in M_KMClasses is equal to the number of rows in *populationWave*+1. The last row contains the number of class members. The remaining rows represent the center of the class. For example, if *populationWave* has two rows then the dimensionality of the problem is 2 and M_KMClasses has 3 rows with the first row containing the first components of each class center, the second row containing the second components of each class center and the third row containing the number of elements in each class.

format=2: Output (in addition to M_KMClasses) the class membership in the wave W_KMMembers. The rows in this 1D wave correspond to sequential members of *populationWave* and the entries correspond to the (zero based) column number in M_KMClasses.

/SEED=*val* Sets the seed for a new sequence in the pseudo-random number generator that is used by the operation. *val* must be an integer greater than zero.

By changing the sequence you may be able to find new solutions or just make the process converge at a different rate.

/TER=*method* Determines when the iterations stop.

method=1: User-specified number of iterations (/TERN).

method=2: Default; continue iterating until no more than a fixed number of elements change classes in one iteration (TERN).

/TERN=*num* Specifies the termination number. The meaning of the number is determined by /TER above. By default, the termination *method*=2 and the default value of the maximum number of elements that change classes in one iteration is 5% of the size of the population.

/Z No error reporting. If an error occurs, sets V_flag to -1 but does not halt function execution.

Details

KMeans uses an iterative algorithm to analyze the clustering of data. The algorithm is not guaranteed to find a global optimum (maximum likelihood solution) so the operation provides various flags to control when the iterations terminate. You can determine if the operation iterates a fixed number of times or loops until at most a specified maximum number of elements change class membership in a single iteration. If you are computing KMeans in more than one dimension you should pay attention to the relative magnitudes of the data in each dimension. For example, if your data is distributed on the interval [0,1] in the first dimension and on the interval [0,1e7] in the second dimension, the operation will be biased by the much larger magnitude of values in the second dimension.

Examples

Create data with 3 classes:

```
Make/O/N=(1,128) jack=4+gnoise(1)
jack[0][15,50]+=10
jack[0][60,]+=20
```

Perform KMeans looking for 5 classes:

```
KMeans/init=1/out=1/ter=1/dead=1/tern=1000/ncls=5 jack
Print M_KMClasses

  M_KMClasses[0][0]= {24.1439,68}
  M_KMClasses[0][1]= {14.1026,36}
  M_KMClasses[0][2]= {4.01537,24}
```

See Also

The **FPClustering** function.