SOMClustering Documentation

Module name: SOMClustering
Description: Self-Organizing Maps algorithm
Author: Keith Ohm (Broad Institute), gp-help@broad.mit.edu
Date: 10/28/03
Release: 1.0

Summary:
The Self Organizing Map (SOM) is a clustering algorithm where a grid of 2D nodes (clusters) is iteratively adjusted to reflect the global structure in the expression dataset. With the SOM, the geometry of the grid is randomly chosen (e.g., a 3 x 2 grid) and mapped to the k-dimensional gene expression space. The mapping is then iteratively adjusted to reflect the natural structure of the data. Resulting clusters are organized in a 2D grid where similar clusters lie near to each other and provide an automatic "executive" summary of the dataset. This module is a standard implementation of the SOM algorithm that can be used to cluster genes or samples (or just about any data, i.e. stocks, mutual funds, spectral peaks, etc).

References:

- Tamayo, P., Slonim, D., Mesirov, J., Zhu, Q., Dmitrovsky, E., Lander, E.S., Golub, T.R. (1999) Interpreting gene expression with self-organizing maps: Methods and application to hematopoeitic differentiation. *Proc. Natl. Acad. Sci. USA* 96:2907–2912.

Parameters:

Name	**Description**
dataset.filename:	Dataset (res, gct, or odf dataset)
cluster.range:	Range of values can be entered and the program will automatically run the algorithm for each number of clusters in the range. For example, range 2-6 will produce clusters of (1 x 2), (1 x 3), (1 x 4), (1 x 5), (1 x 6), (2 x 2) and (2 x 3).
iterations:	How many times the algorithm should try to refine the clusters. Initially, this value can be set low for faster exploration, but should be set high (e.g., 50,000 or 500,000) for good convergence.
seed.range:	The seed for the random number generator, is exposed to allow the user to recreate a given session at a later time, (as opposed to allowing the program to generate a random initial seed which could result in different outcomes even if all the other parameters are identical).
cluster.by	Whether to cluster by rows or columns.
som.rows	setting this and *som.cols* to a non zero value will override *cluster.range* and the computation will be forthe specified geometry
som.cols	setting this and *som.rows* to a non zero value will override *cluster.range* and the computation will be forthe specified geometry

initialization	The SOM algorithm starts from a set of random centroids. These centroids can be initialized by: Random_Vectors(new vectors are randomly generated) or Random_Datapoints(actual datapoints are randomly selected to use as the initial centroids)
neighborhood	The neighborhood function determines how centroids near to the target centroid are updated. Gaussian; all centroids get updated and they are weighted by a Gaussian centered on the target centroid, with a standard deviation of sigma. Bubble; centroids within sigma get a full update and centroids outside of sigma get no update.
alpha.initial	The initial learning weights. Centroid updates are weighted by the learning rate.
alpha.final	The final learning weights. Centroid updates are weighted by the learning rate.
sigma.initial	The initial sigma that determine the size of the update neighborhood around the target centroid.
sigma.final	The final sigma that determine the size of the update neighborhood around the target centroid.

Return Value:

1. SOM Cluster results files (one file or more depending on the *cluster.range* specified)
2. Stdout.txt: the"stdout" text output from running the program.

Platform dependencies:

Task type:	Clustering
CPU type:	any
OS:	any
Java JVM level:	1.3
Language:	Java