GKROM

Carol Alexander Wei Wei Xi Chen

class GKROM (M, path = None, folder_name = None, simuSize = None, match4 = False, fixedOmega = True, initStyle = 'uniform', maxT_col2 = 30, maxT_incr = 4, edf = 2, xtol = None, maxiter = None, max_RO = 20)

This class generate a simulated sample that matches a given sample 'M' in its mean, variance and Kollo skewness (match4 = False), as well as Kollo kurtosis if required (match4 = True).

Dependencies:

```
import numpy as np
import math as math
import random as random
import time as time
import os
from scipy.optimize import root
from scipy.linalg import cholesky
```

Parameters: **M :** ***ndarray***

A target sample to match.

path : ***Nonetype or str, optional***

The path under which the simulated sample is saved. The default value is None.

If None, no path is provided, and the simulated sample will not be exported.

If path is assigned by a string identifying the path to save the results, the simulated sample will be saved into a folder under the provided path.

folder_name : ***Nonetype or str, optional***

The name of the folder under the given path where the simulated sample is saved. The default value is None, indicating no folder name is provided.

When None and yet the simulated sample is still required exporting (path is given a string), a folder called "simulated_path" will be created under the given path for the simulated sample recorded in csv.

simuSize : ***Nonetype or int, optional***

The number of observations per dimension in the simulated sample. The default value is None.

If None, the simulated sample will be set with the same number of observations as the target sample.

match4 : ***Nonetype or bool, optional***

The default value is False.

If False, and only the first three moments are matched.

If True, all first four moments are matched.

fixedOmega : ***bool, optional***

The default value is True.

If True, and a fixed rotation matrix (see paper page 9) is applied.

If False, a random normal rotation matrix is applied.

*** The parameters below are only relevant to matching all first four moments (match4 = True) ***

initStyle : ***str, optional***

The method of generating initial guesses for solving the equation system. The default value is 'uniform'.

If 'uniform', the initial guesses are drawn from a uniform (-1,1) distribution.

If 'equal', the initial guesses are set equal across all dimensions.

maxT_col2 : ***int or float, optional***

Time limit (in seconds) for solving the second column. The default value is 30.

When this time limit is breached, the code is bounced back to regenerate column 1 and search for a feasible column 2.

maxT_incr : ***int or float, optional***

Incremental multiplier on the time limit for solving the next column. The default value is 4.

For example, when maxT_incr = 4 and maxT_col2 = 30, the time limit for solving the third column is $30 \times 4 = 120$ seconds, and the time limit for solving the fourth column is $120 \times 4 = 480$ seconds.

When the time limit is breached, the code is bounced back to regenerate column 1, 2, 3 . . . etc.

edf : ***int, optional***

Extra number of orthogonal vectors need for each column. The default value is 2.

xtol : ***Nonetype or float, optional***

Tolerance for termination when searching for a real solution of the equation system. The default value is None.

If None, the tolerance is set as default given by the scipy solver.

maxiter: ***Nonetype or int, optional***

Maximum iterations for finding a real solution of the equation system. The default value is None.

If None the maximum iterations is set as default given by the scipy solver.

max_RO: ***int or float, optional***

Time limit (in seconds) for finding a set of proper orthogonal vectors. The default value is 20.

Generating a simulated sample (Example):

```
# Import GKROM class from GKROM_core file
%run ''xxx/GKROM.ipynb'' import GKROM
# xxx is str of the path where the class file is saved; it does not have to be the same as the path
    where the simulated sample is exported to (as .csv)

# Launch function with pre−defined parameters
result = GKROM(M, path, folder name, simuSize, match4, fixedOmega)

# Compute simulated sample
result.run()
```

Output:

result.output : ***ndarray***

The matrix of a simulated sample that matches the target sample.

When exported, it is saved in a file named "Simulated_sample_xxx.csv" where xxx is an integer generated from the time stamp at which the file is exported.

result.stdized_output: ***ndarray***

The matrix of a standardized simulated sample that matches the target sample.

result.m : ***int***

The number of observations in the simulated sample.

result.mu : ***ndarray***

The mean vector of the target sample.

result.A : ***ndarray***

The matrix of the Cholesky decomposition of the target covariance.

result.b: ***ndarray***

The matrix of the target Kollo skewness.

result.L_o: ***ndarray***

The matrix of the standardized target sample.

result.path: ***Nonetype or str***

The path under which the simulated sample is saved.

result.math4 : ***bool***

False: only the first three moments are matched.

True: all first four moments are matched.

result.fixedOmega : ***bool***

True, and a fixed rotation matrix (see paper page 9) is applied.

False, a random normal rotation matrix is applied.

*** The output below are only meaningful to matching all first four moments ***

result.K: ***ndarray***

The matrix of the target Kollo kurtosis.

result.initStyle : ***str***

The way of generating initial guesses for solving the equation system.
'uniform': the initial guesses are drawn from a uniform (-1,1) distribution.
'equal': the initial guesses are set equal across all dimensions.

result.maxT_col2 : ***int or float***

Time limit (in seconds) for solving the second column.

result.maxT_incr : ***int or float***

Incremental multiplier on the time limit for solving the next column.

result.edf: ***int***

Extra number of orthogonal vectors need for each column.

result.xtol: ***Nonetype or float***

Tolerance for termination when searching for a real solution of the equation system.
None when the tolerance is set as default given by the scipy solver.

result.maxiter: ***Nonetype or int***

Maximum iterations for finding a real solution of the equation system.
None when the tolerance is set as default given by the scipy solver.

result.max_RO : ***int or float***

Time limit (in seconds) for finding a set of proper orthogonal vectors.