Introduction to `BNPTSclust`

Martell-Juarez, David A.   Nieto-Barajas, Luis E.

April 19th, 2015

Abstract

This document explains detailedly how to use the functions from the `BNPTSclust` package, which perform the Bayesian Nonparametric Algorithm for Time Series Clustering developed by Nieto-Barajas & Contreras-Cristan (2014). Each function implements a Gibbs Sampler to approximate the posterior distribution of the parameters that will define the final clustering. Section 1 gives a brief introduction to the problem that wants to be solved and its relevance. Section 2 outlines some theoretical results about how the clustering is obtained through the algorithm. Section 3 explains how to use each of the functions from the package and shows the results of some worked examples.

Contents

1 Introduction

Data classification is an important aspect to consider while studying a sample because it allows us to identify the heterogeneity present in the data. The explanation of such source of heterogeneity might give us more information about the data set which we might not have considered at first. Therefore, it is relevant to have data classification methods. In particular, a time series clustering algorithm would let us understand which variables have the same or different behavior through time.

Statistics has developed solutions for data classification during the course of its development. Nevertheless, many of them depend on the researcher's own criteria to determine the final groupings of the data.

The Bayesian Nonparametric Approach proposed by Nieto-Barajas and Contreras-Cristan (2014) provides an objective time series clustering scheme, because the classification mechanism is decided by the model itself. The algorithm is computer intensive, but technological development has given feasibility to the bayesian nonparametric tools it employs, which have had an acceptable performance so far.

2 Theoretical bases of the model

Bayesian analysis carries out statistical inference by combining the likelihood function of a sample with some prior information of the variable of interest. This prior information is typically modeled by assuming that the variable of interest follows a probability distribution, which is known as a *prior distribution.* This distribution is incorporated into the analysis through Bayes' Theorem to obtain the *posterior distribution* of the variable of interest. Inference is made once the *posterior distribution* is obtained.

Let $\mathbf{y_i} = \{y_{it} : t = 1, ..., T\}, i = 1, ..., n$, denote a time series and $\mathbf{y} = \{\mathbf{y_1}, ..., \mathbf{y_n}\}$ denote the sample of time series under consideration. Each observation is modeled as if it followed a dynamic linear model of the form:

$$y_{it} = F_{it}\theta_{it} + \epsilon_{it} \tag{1}$$

$$\theta_{it} = \rho\theta_{it-1} + \nu_{it} \tag{2}$$

where $\epsilon_{it} \sim N\left(0, \sigma^2_{\epsilon_i}\right)$ and $\nu_{it} \sim N\left(0, \sigma^2_{\theta}\right)$.

Equation (2) is known as the evolution equation and it represents an autoregressive process of order 1. Every θ_{it} accounts for time dependencies in the observations.

To accomodate level, trends, seasonal and temporal components, equation (1) is defined as:

$$E[y_{it}] = \mu_i + \omega_i^{'} g(t) + \upsilon_i^{'} h(t) + \theta_{it} \tag{3}$$

where μ_i denotes the level of the series; $\omega_i^{'} g(t)$ denotes a polynomial trend of the form: $\omega_{1i} t + \omega_{2i} t^2 + ... + \omega_{ni} t^n$ and $\upsilon_i^{'} h(t)$ denotes seasonal components like: $\upsilon_{1i} I(January) + ... + \upsilon_{11i} I(November)$.

The time series will be clustered according to the parameters that determine the mean level of the series, i.e. the parameters in vector $\eta_{\mathbf{i}} = (\mu_i, \omega_{\mathbf{i}}, \nu_{\mathbf{i}}, \theta_{\mathbf{i}})$. However, not all parameters might be useful for clustering, since two series that share the same trend and seasonalities may be desired to belong of the same cluster, despite their level differences.

Therefore, we can separate $\eta_{\mathbf{i}} = (\alpha_{\mathbf{i}}, \beta_{\mathbf{i}}, \theta_{\mathbf{i}})$ where $\alpha_{\mathbf{i}}$ will contain the parameters *not* considered for clustering and $\gamma_i = (\beta_{\mathbf{i}}, \theta_{\mathbf{i}})$ will contain the parameters used for clustering.

Given all of these specifications, the general sampling model for each time series will be defined as:

$$\mathbf{y_i} = \mathbf{Z}\alpha_{\mathbf{i}} + \mathbf{X}\beta_{\mathbf{i}} + \theta_{\mathbf{i}} + \epsilon_{\mathbf{i}}, i = 1, ..., n. \tag{4}$$

where $\mathbf{Z}$ and $\mathbf{X}$ have dimensions $T \times p$ and $T \times d$ respectively. p represents the number of parameters that will not be considered for clustering and d is the number of parameters that will be taken into account for clustering. $\alpha_{\mathbf{i}}$ is a $T \times 1$ vector, $\beta_{\mathbf{i}}$ is a $d \times 1$ dimensional vector and θ_i has dimension $T \times 1$. $\epsilon_i \sim N_T(0, \sigma^2_{\epsilon_i} I)$.

2.1 Prior specification on $\gamma_{\mathbf{i}}$

This prior specification is one of the most important for the model, because it defines the classification mechanism that will produce the time series clusters.

The whole vector $\gamma = \{\gamma_{\mathbf{1}}, ..., \gamma_{\mathbf{n}}\}$ is assumed to come from a *Poisson-Dirichlet* process denoted by $\mathcal{PD}(a, b, G_0)$. The parameters a and b characterize the process completely but the centering measure G_0 can be specified externally. For the model, it is assumed that $G_0 = N_d(\beta|\mathbf{0}, \mathbf{\Sigma}_\beta) \times N_T(\theta|\mathbf{0}, \mathbf{R})$ with $\mathbf{\Sigma}_\beta = \mathbf{diag}(\sigma^{\mathbf{2}}_{\beta_{\mathbf{1}}}, ..., \sigma^{\mathbf{2}}_{\beta_{\mathbf{n}}})$ and $R_{jk} = \sigma^2_\theta \rho^{|j-k|}$.

The *Poisson-Dirichlet* process is almost surely a discrete random measure and it has the property that each $\gamma_{\mathbf{i}}$ is a copy from an element in the vector $\gamma_{\mathbf{-i}} = \{\gamma_{\mathbf{1}}, ..., \gamma_{\mathbf{i-1}}, \gamma_{\mathbf{i+1}}, ..., \gamma_{\mathbf{n}}\}$ with certain probability and it is a drawing from the density g_0 associated to G_0 with another probability, i.e.:

$$f(\gamma_{\mathbf{i}}|\gamma_{-\mathbf{i}}) = \frac{b + am_i}{b + n - 1} g_0(\gamma_{\mathbf{i}}) + \sum_{j=1}^{m_i} \frac{n^*_{j,i} - a}{b + n - 1} I_{\gamma^*_{\mathbf{j,i}}}(\gamma_{\mathbf{i}}), i = 1, ..., n \tag{5}$$

where $(\gamma^*_{\mathbf{1,i}}, ..., \gamma^*_{\mathbf{m_i,i}})$ denotes the unique values in $\gamma_{-\mathbf{i}}$ which occur with frequency $n^*_{j,i}, j = 1, ..., m_i$.

If for two series i, j it is the case that $\gamma_{\mathbf{i}} = \gamma_{\mathbf{j}}$, then it will be considered that both belong to the same group. However, we need the posterior distribution of every $\gamma_{\mathbf{i}}$ to be able to determine the final clustering.

Such posterior distribution cannot be expressed analytically and it must be calculated numerically as explained by Nieto-Barajas, L.E. and Contreras-Cristan, A. (2014) through a process called *Gibbs sampling*. This is the objective of the functions from this package. The user supplements the file with the time series information and the functions carry out the Gibbs sampling to approximate the posterior distribution of γ.

2.2 Prior specification of the rest of the parameters

To approximate correctly the posterior distribution of γ we need to compute the posterior distribution of the rest of the parameters in the model too. These distributions are approximated through *Gibbs sampling* as well in the functions of the package. The prior distributions assigned to the rest of the parameters are the following:

- $\alpha_{\mathbf{i}} \overset{iid}{\sim} N_p(\mathbf{0}, \mathbf{\Sigma}_\alpha), i = 1, ..., n$
- $\sigma^2_{\epsilon_i} \sim IGa(c^\epsilon_0, c^\epsilon_1), i = 1, ..., n$
- $\sigma^2_{\beta_j} \sim IGa(c^\beta_0, c^\beta_1), j = 1, ..., d$
- $\sigma^2_{\alpha_k} \sim IGa(c^\alpha_0, c^\alpha_1), k = 1, ..., p$
- $f(\sigma^2_\theta, \rho) \propto (\sigma^2_\theta)^{-1} \frac{\sqrt{1+\rho^2}}{1-\rho^2}$
- $f(a) = \pi I_0(a) + (1 - \pi) Be(a|q^a_0, q^a_1)$
- $f(b|a) = Ga(b + a|q^b_0, q^b_1)$

3 Functions description and examples

The functions from the package can be divided into three categories:

- **Clustering functions**. These are: `tseriescm`, `tseriescq` and `tseriesca`, which perform time series clustering for monthly, quarterly and annual data respectively. They implement a Gibbs sampler to approximate the posterior distribution of the model parameters.

- **Plotting functions**. These are: `clusterplots` and `diagplots`, which generate the plots of the time series clustering provided by the model and the diagnostic plots to assess the convergence of the Gibbs sampler.

- **Auxiliary functions**. These are: `comp11`, `desingmatrices` and `scaleandperiods`, which are for internal use and will not be explained here.

3.1 Clustering functions

3.1.1 Arguments

- `data.` This is a data frame that contains the time series information to be clustered. It is assumed that the periods of the series appear as the row names of the file.

- `maxiter.` *Gibbs sampling* is an iterative process and it requires a several number of repetitions to achieve convergence and approximate correctly the posterior distribution of the model parameters. `maxiter` is the maximum number of iterations that the Gibbs sampling carries out. The default is 2000 and the user can increase the number of iterations under the consideration that it will also take more time for the program to finish, since it is computer intensive.

- `burnin.` It takes several iterations for the Gibbs sampler to become stable once the process begins. Therefore, these initial iterations are usually discarded and are considered as a *burn-in* period. The default is to establish this as 10% of the maximum number of iterations desired. This can be modified, but it must always be the case that `maxiter` is greater than `burnin`.

- `thinning.` Gibbs sampling works better when only one in every few iterations is taken into account instead of all the iterations as a whole. This procedure is known as *thinning* and the default is to take one in every five iterations as valid during Gibbs sampling.

- `level, trend` and `seasonality`. Remember that the observation equation of the model defines the time series as a function of their level, trend, seasonal and temporal components, but that not all of them may be taken into account for clustering. The temporal components are always considered for this objective, but the rest of them are left to the choice of the user. `level, trend` and `seasonality` are just flag variables that signal if such components will be considered for clustering. If they are equal to `FALSE`, they are left out and if they are equal to `TRUE`, they are included as clustering components. The default is: `level = FALSE`, `trend=seasonality=TRUE`.

- `deg`. This variable defines the degree of the trend polynomial desired for the model. The default is 2.

- `c0eps,c1eps,c0beta,c1beta,c0alpha,c1alpha`. These are the parameters of the inverse gamma prior on $\sigma^2_\epsilon, \sigma^2_\beta, \sigma^2_\alpha$. Their value must be always positive and they affect the number of groups that will be in the final clustering. The default values are: `c0eps = c0beta = c0alpha = 2` and `c1eps = c1beta = c1alpha = 1` which tend to generate a small number of clusters. If the number of groups wants to be increased, then these parameters should be given values close to zero, e.g. `c0eps = c0beta = c0alpha = 0.001` and `c1eps = c1beta = c1alpha = 0.001`.

- `priora` and `priorb`. These arguments indicate whether a prior distribution on parameters `a` and `b` should be assigned or not. If their value is `FALSE`, then no prior is assigned on the parameters and their value is fixed throughout the algorithm. If their value is `TRUE`, then the posterior distribution of these parameters is approximated by the functions. The default value is `priora = priorb = FALSE`. Of course, both arguments need not take the same value.

- `pia, q0a, q1a, q0b` and `q1b`. This are the parameters of the prior distribution proposed for `a` and `b`. These arguments will only play a role if `priora` or `priorb` are set to 1. `pia` must be a number in the interval $(0, 1)$ and `q0a, q1a, q0b, q1b` must be positive numbers. The default is `pia = 0.5` and `q0a=q1a=q0b=q1b = 1`

- `a` and `b`. If `priora` or `priorb` are zero, `a` and `b` are the fixed values the parameters take throughout the algorithm. If `priora` or `priorb` are one, then `a` and `b` are the initial values that the parameters take for the algorithm. Please note that `a` and `b` must always satisfy that: $0 \leq \texttt{a} < 1$ and $\texttt{a} + \texttt{b} > 0$.

- `indlpml`. This argument indicates if the LPML (Logarithm of Pseudo-Marginal Likelihood) for the model is to be computed. This is a goodness of fit measure and larger positive values indicate a better fit. If `indlpml` is `FALSE`, then LPML is not calculated. If `indlpml` is `TRUE`, then the LPML is computed, but this will make the functions run slower. The default value is `indlpml = FALSE`, since the LPML value is not essential for the clustering outcome.

3.1.2 Output

While the function is running, every 50 iterations a message like this will be displayed:

```
                              ⋮
 Iteration Number:  1500.  Progress:  75%
Iteration Number:  1550.  Progress:  77.5%
 Iteration Number:  1600.  Progress:  80%
                              ⋮
```

Clearly, progress percentage depends on the maximum number of iterations desired. When they finish, the functions will return the following variables as a list:

- `mstar`. This variable contains the number of groups in the chosen cluster configuration.
- `gnstar`. Each time series is identified by a number according to its column in the file provided by the user. `gnstar` will contain the group number to which each time series belongs in the final cluster configuration. For example, if the file contains 10 time series and group 1 consists of the series 1, 4, 7 and 10, group 2 consists of the series 2, 3, 5, 6 and 8 and group 3 consists only of series 9, then `gnstar` would be defined as:

```
> gnstar
[1] 1 2 2 1 2 2 1 2 3 1
```

- `HM`. This is the *Heterogeneity Measure* of the final cluster configuration. The larger the value of HM, the more heterogeneous a clustering is. A clustering with small HM and a small number of groups is preferable.
- `arrho`, `ara` and `arb`. At each iteration, *Gibbs sampling* produces a simulation of the posterior distribution of the model parameters. However, the parameters ρ, a and b, require an extra step in which the simulated values are not always accepted as a sample of their posterior distribution. Hence, the variables `arrho`, `ara` and `arb` contain the *Acceptance Rate* of the simulations for each parameter. The *Acceptance Rate* is simply the number times a simulation was accepted over the total number of iterations.
- `sig2epssample`, `sig2alphasample`, `sig2betasample`, `sig2thesample`, `rhosample`, `asample`, `bsample` and `msample`. The first three variables are matrices that contain in their columns the posterior distribution sample of parameters $\sigma^2_{\epsilon_i}$, $\sigma^2_{\alpha_j}$ and $\sigma^2_{\beta_k}$, $\forall i, j, k$. The rest of the variables are vectors with the posterior distribution sample of $\sigma^2_\theta, \rho, a, b$ and the sample of the number groups at each Gibbs sampling iteration saved.
- `lpml`. The variable contains the value of LPML calculated for the model if the argument `indlpml` was set to `TRUE`.

3.2 Plotting functions

.

Once the clustering functions are finished, the user can visualize the output and assess algorithm convergence through the plotting functions.

3.2.1 Arguments

- `L`. List that contains the output of the clustering functions.

  Besides, the function `clusterplots` must receive as argument the data frame that contains the time series information.

3.2.2 Output

The `clusterplots` function returns the graphs of the time series that belong to each cluster, for all clusters. This makes it easier to visualize and to interpret the reason for which the algorithm determined the final cluster configuration.

The `diagplots` function returns three different sets of graphs:

- *Trace plots.* These are the first eight plots returned and they assess the convergence of the Gibbs sampler. They are simply the plot of the simulated values of the posterior distribution of each parameter at every iteration. If the plots are stationary, then the Gibbs sampler achieved convergence and the final clustering configuration is acceptable.

- *Histograms.* These are the following eight and they give an idea of the frequency in which each simulated value appears as a sample of the posterior distribution of each parameter.

- *Ergodic mean plots.* The next eight plots are the ergodic mean of the samples of the posterior distributions of the model parameters at every iteration. The ergodic mean is just the sum of the simulated values up to a certain iteration divided by the total number of iterations at that point. If the plot converges to a value as the iterations increase, then it can be considered that the Gibbs sampler achieved convergence and the final clustering configuration is acceptable.

These are some examples of the output that each function produces. Please note that the examples might have slight differences from what you get, due to the probabilistic nature of the functions.

3.3 `tseriescm` example

This function performs the clustering algorithm for monthly time series data. This example comes from the monthly adjusted closing prices of 58 shares from the Mexican stock exchange market. This is the database used by Nieto-Barajas, L.E. and Contreras-Cristan, A. (2014). The following output was obtained by running this code:

```
> data(stocks)
> L <- tseriescm(stocks,maxiter = 4000,level = FALSE,trend = TRUE,seasonality
= TRUE,a = 0,b = 1)
> clusterplots(L,stocks)
> diagplots(L)
```

These arguments imply that the prior on γ will be a Poisson-Dirichlet Process. The ouput is the following:

- *Console output*

```
Number of groups of the chosen cluster configuration:  9
Time series in group 1 : 1 2 4 5 7 10 12 13 19 21 22 25 29 30 31 33 34 40 41 42
43 44 46 47 48 49 52 57 58
Time series in group 2 : 3 6 8 9 11 14 15 17 18 26 27 28 32 35 36 37 38 45 50 51
53 56
Time series in group 3 : 16
Time series in group 4 : 20
Time series in group 5 : 23
Time series in group 6 : 24
Time series in group 7 : 39
Time series in group 8 : 54
Time series in group 9 : 55
HM Measure:  199.2226
```

Figure 1: Console output for the function `tseriescm`.

- *Trace plots*

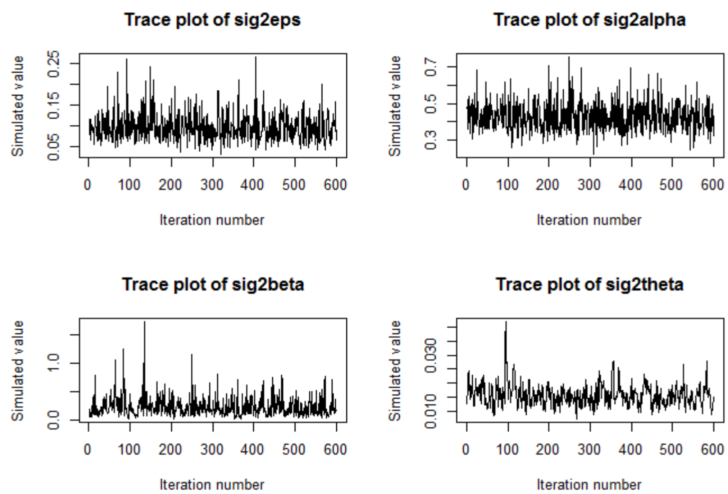

Figure 2: Trace plots for the function `tseriescm`.

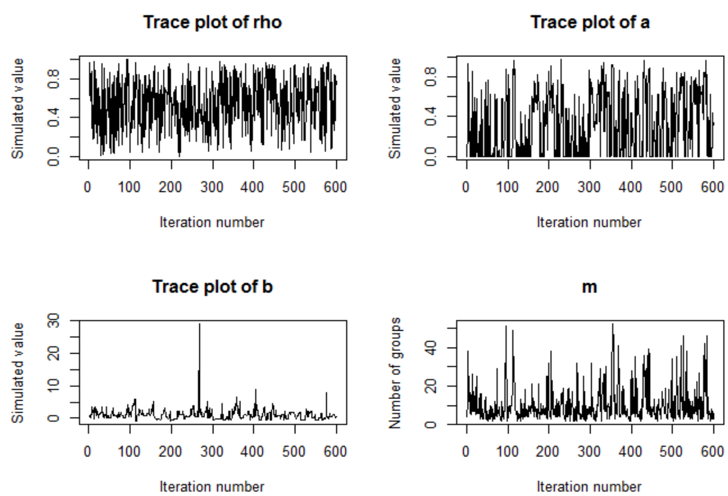

Figure 3: Trace plots for the function `tseriescm`.

- *Histograms*

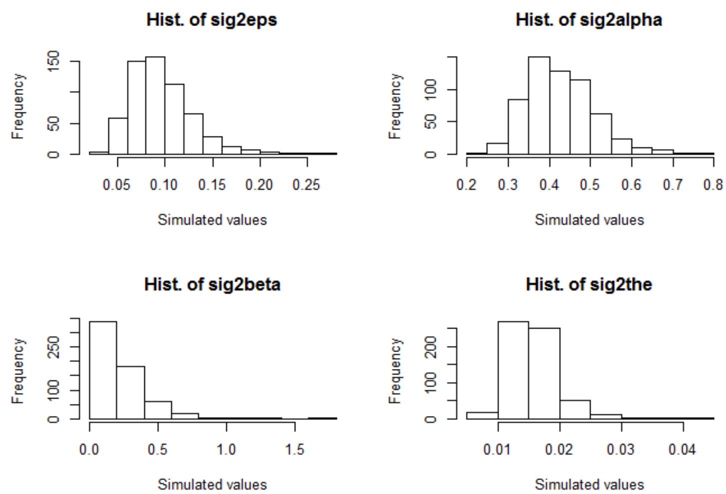

Figure 4: Histograms for the function `tseriescm`.

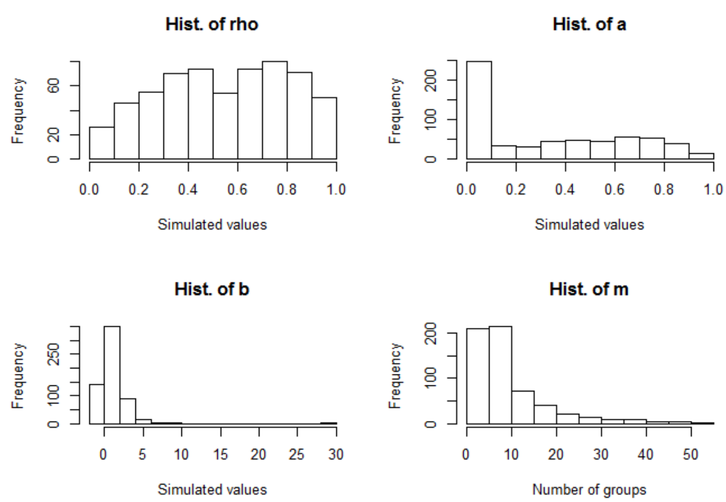

Figure 5: Histograms for the function `tseriescm`.

- *Ergodic Mean Plots*

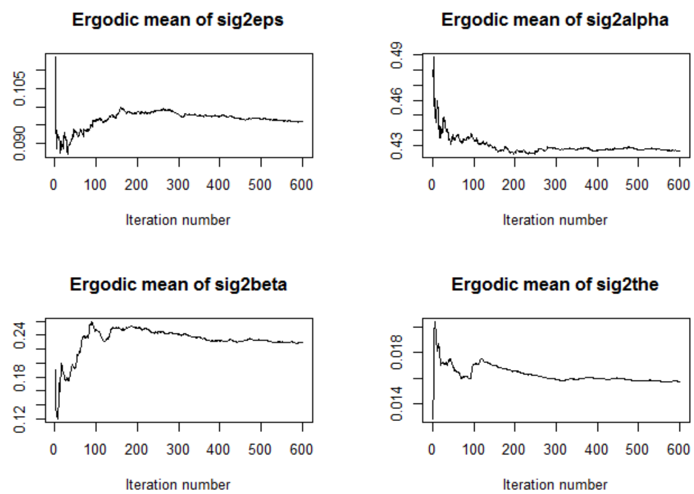

Figure 6: Ergodic Mean plots for the function `tseriescm`.

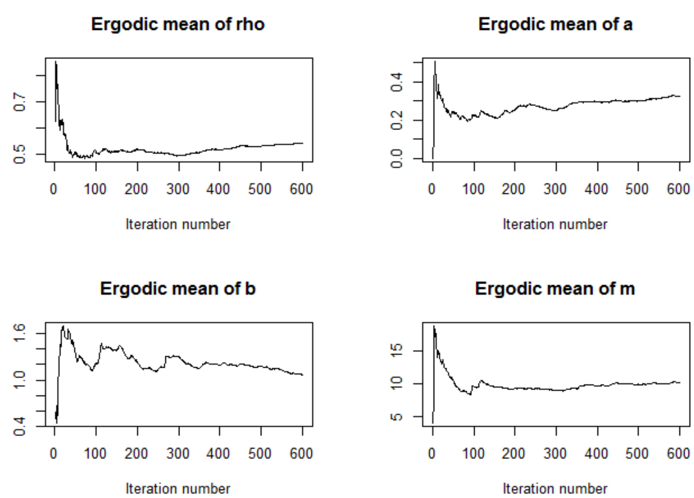

Figure 7: Ergodic Mean plots for the function `tseriescm`.

- *Cluster plots*

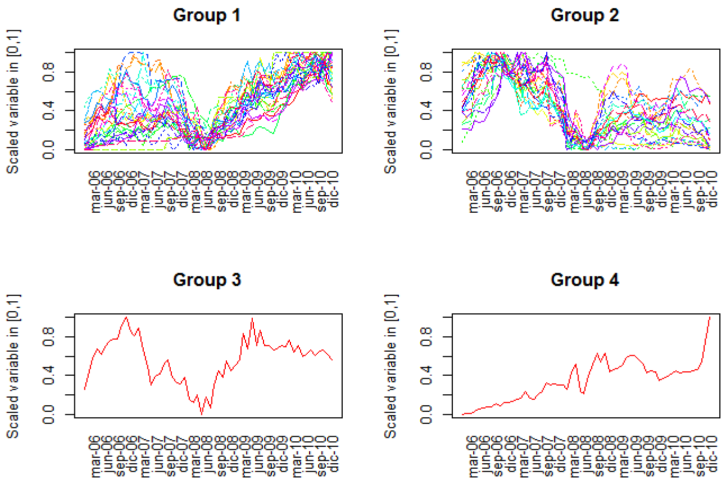

Figure 8: Cluster plots for the function `tseriescm`.

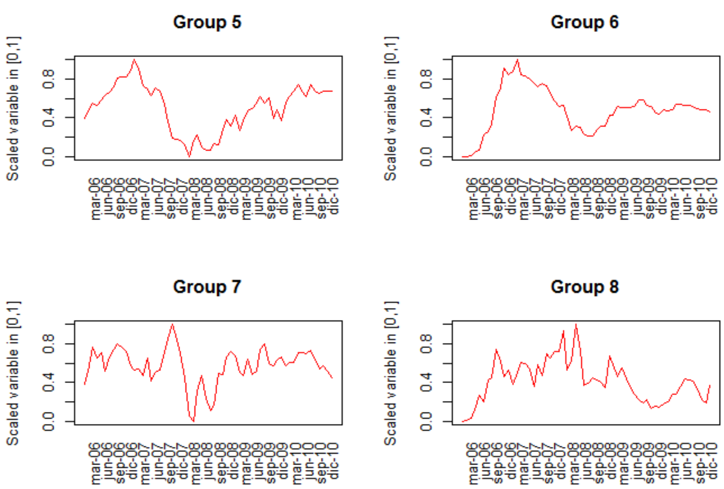

Figure 9: Cluster plots for the function `tseriescm`.

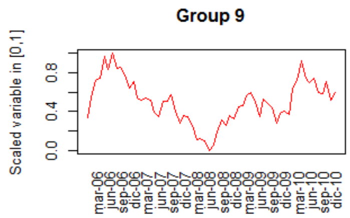

Figure 10: Cluster plots for the function `tseriescm`.

3.4 tseriescq example

This function performs the clustering algorithm for quarterly time series data. This example comes from quarterly average house price data in Scotland from 2003 to 2014. The output was obtained by running the following code:

```
> data(houses)
> L <- tseriescq(houses,maxiter = 4000,priora = 1)
> clusterplots(L,houses)
> diagplots(L)
```

These arguments imply that the prior on γ will be a Normalized Stable Process. The ouput is the following:

- *Console output*

```
Number of groups of the chosen cluster configuration:  9
Time series in group 1 : 1
Time series in group 2 : 2 3 4 5 6 7 9 10 11 12 13 15 16 17 18 19 20 21 25 26 27
29 30 31 33
Time series in group 3 : 8 23
Time series in group 4 : 14
Time series in group 5 : 22
Time series in group 6 : 24
Time series in group 7 : 28
Time series in group 8 : 32
Time series in group 9 : 34
HM Measure:  126.9543
```

Figure 11: Console output for the function tseriescq.

- *Trace plots*

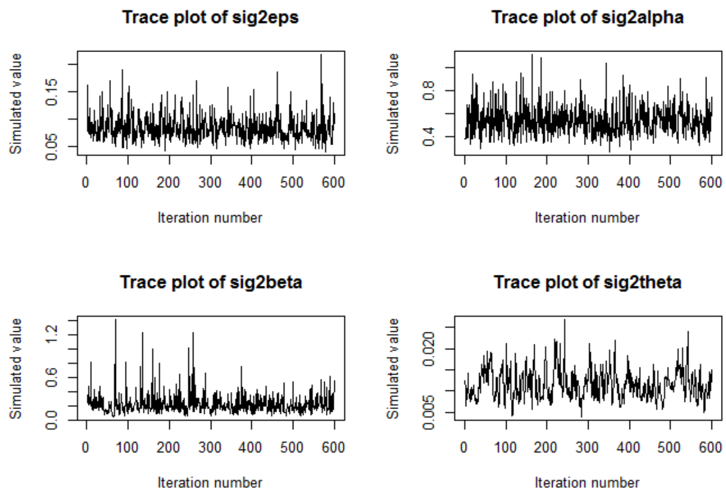

Figure 12: Trace plots for the function `tseriescq`.

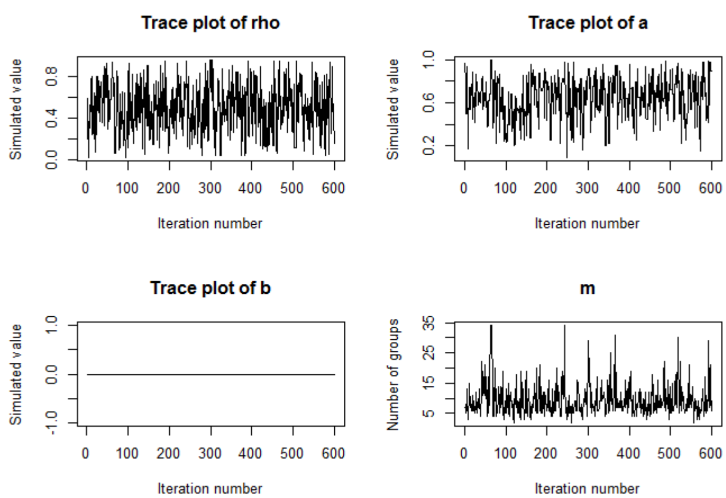

Figure 13: Trace plots for the function `tseriescq`.

• *Histograms*

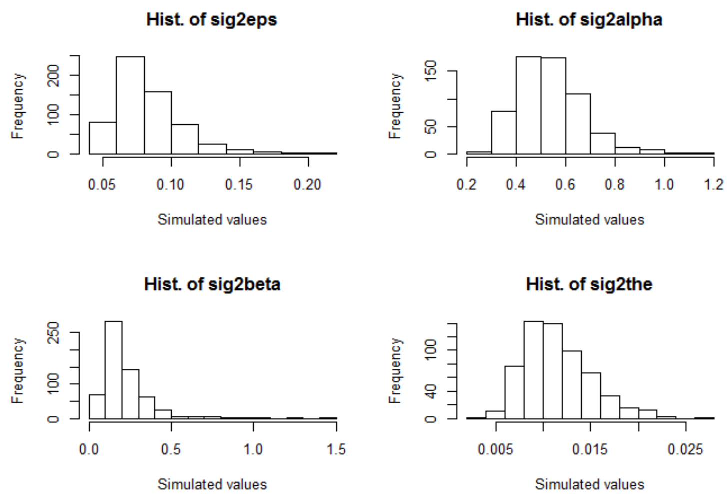

Figure 14: Histograms for the function `tseriescq`.

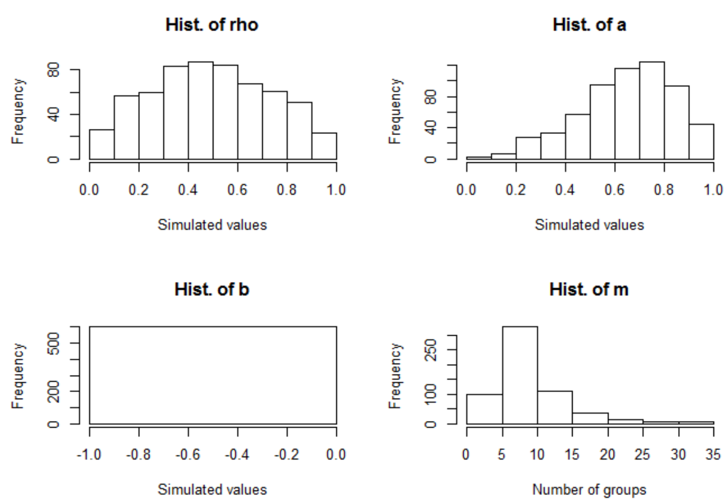

Figure 15: Histograms for the function `tseriescq`.

- *Ergodic Mean Plots*

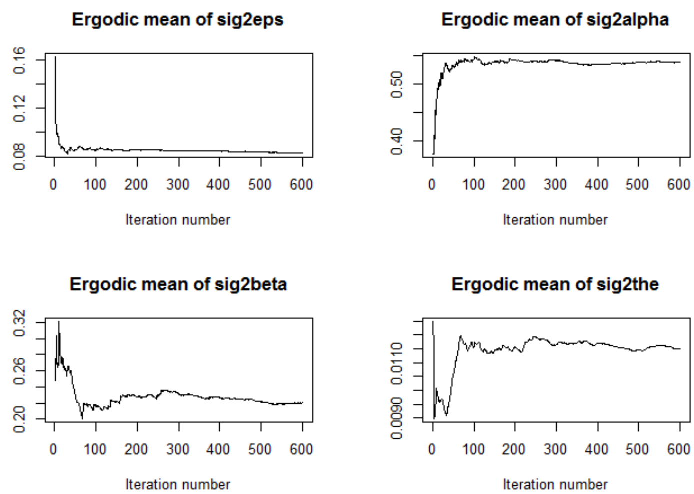

Figure 16: Ergodic Mean plots for the function `tseriescq`.

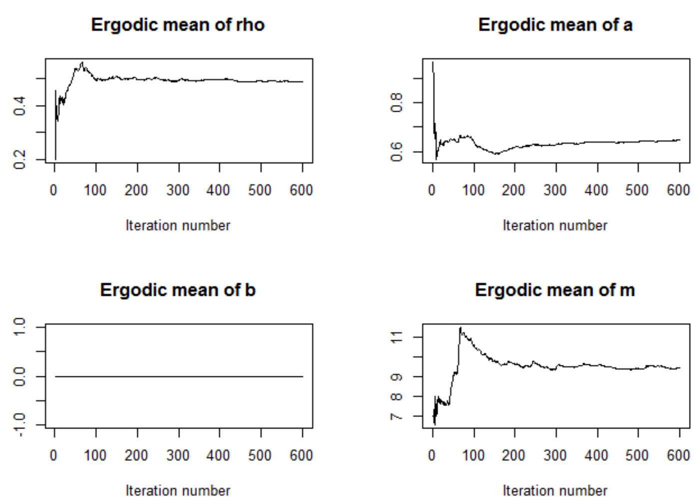

Figure 17: Ergodic Mean plots for the function `tseriescq`.

- *Cluster plots*

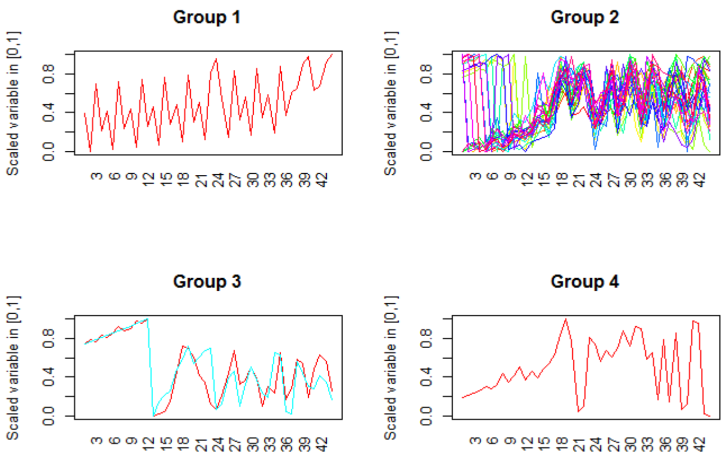

Figure 18: Cluster plots for the function `tseriescq`.

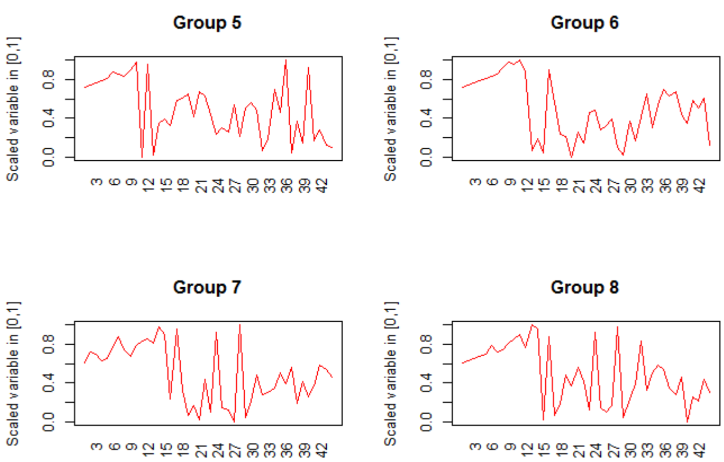

Figure 19: Cluster plots for the function `tseriescq`.

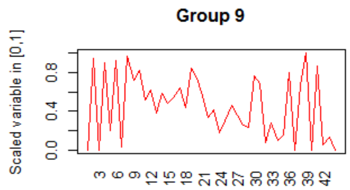

Figure 20: Cluster plots for the function `tseriescq`.

3.5 tseriesca example

This function performs the clustering algorithm for annual time series data. This example comes from the yearly GDP per worker for 121 countries from 1993 to 2012. The output was obtained by running the following code:

```
> data(gdp)
> ts <- tseriesca(gdp,maxiter = 4000,c0eps = 0.001,c1eps = 0.001,
c0beta = 0.001,c1beta = 0.001,c0alpha = 0.001,c1alpha = 0.001,priorb
= 1,a = 0,b = 0.1)
> clusterplots(L,gdp)
> diagplots(L)
```

These arguments imply that the prior on γ will be a Dirichlet Process. The ouput is the following:

- *Console output*

```
Number of groups of the chosen cluster configuration:  13
Time series in group 1 : 1 111
Time series in group 2 : 2 8
Time series in group 3 : 3 4 5 6 7 10 11 12 13 14 15 16 17 18 19 20 21 22 24 25
26 28 29 30 31 32 33 34 35 36 37 38 40 41 42 43 44 45 46 47 49 50 51 52 55 56 57
58 59 61 62 63 65 67 68 69 70 71 74 75 76 77 78 79 80 81 82 83 84 85 86 89 91 92
93 94 95 96 97 100 101 102 103 104 105 106 107 108 109 110 113 114 117 118 120
Time series in group 4 : 9 23 48 54 60 87
Time series in group 5 : 27
Time series in group 6 : 39
Time series in group 7 : 53 73 88
Time series in group 8 : 64
Time series in group 9 : 66 98 112
Time series in group 10 : 72
Time series in group 11 : 90 116 119 121
Time series in group 12 : 99
Time series in group 13 : 115
HM Measure:  99.50627
```

Figure 21: Console output for the function `tseriesca`.

- *Trace plots*

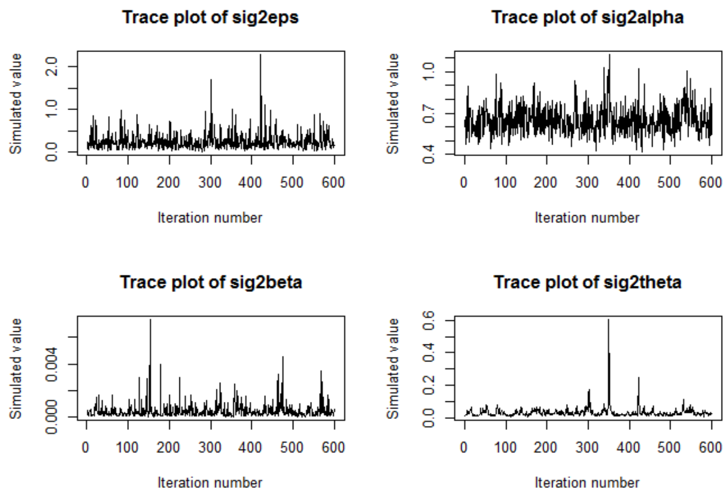

Figure 22: Trace plots for the function `tseriesca`.

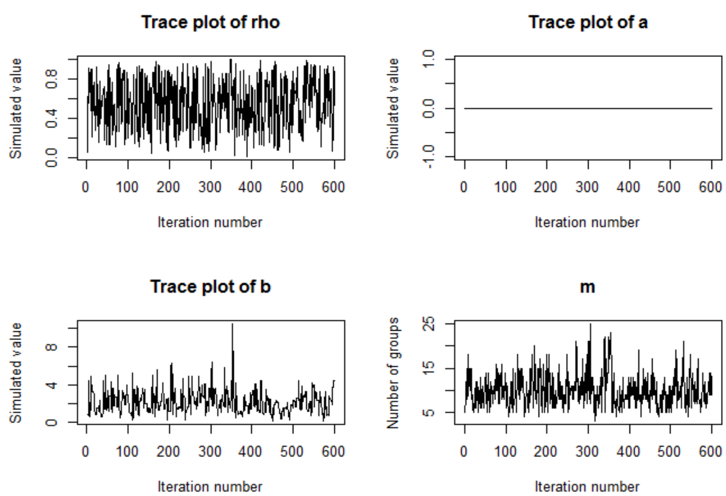

Figure 23: Trace plots for the function `tseriesca`.

- *Histograms*

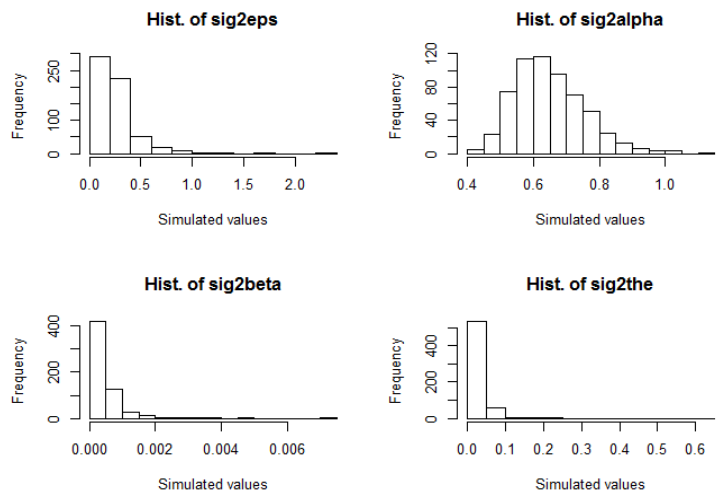

Figure 24: Histograms for the function `tseriesca`.

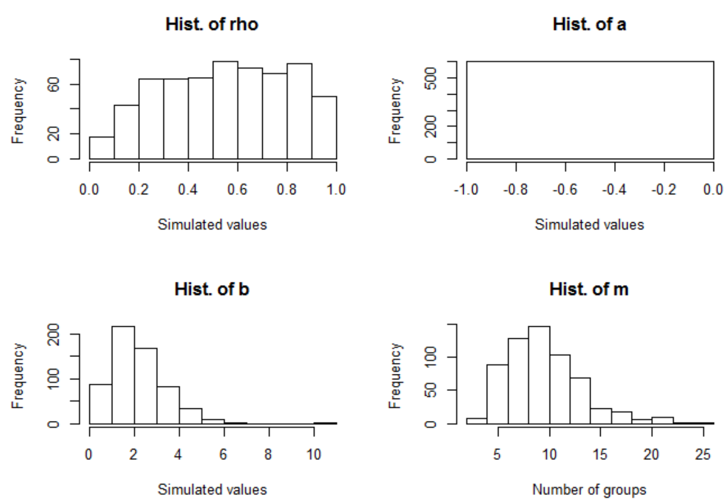

Figure 25: Histograms for the function `tseriesca`.

• *Ergodic Mean Plots*

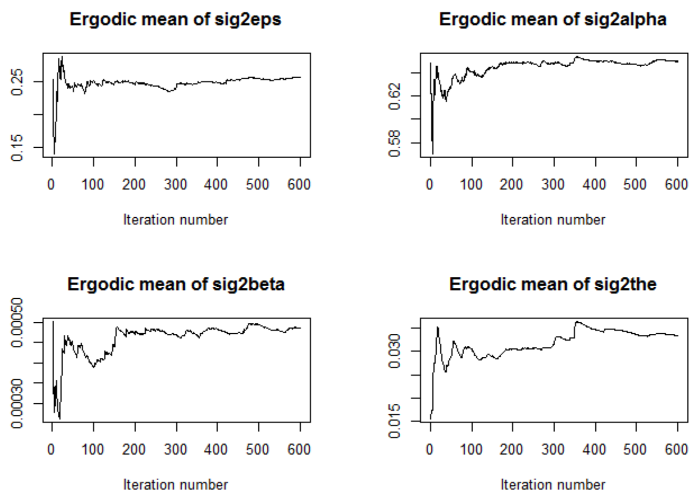

Figure 26: Ergodic Mean plots for the function `tseriescq`.

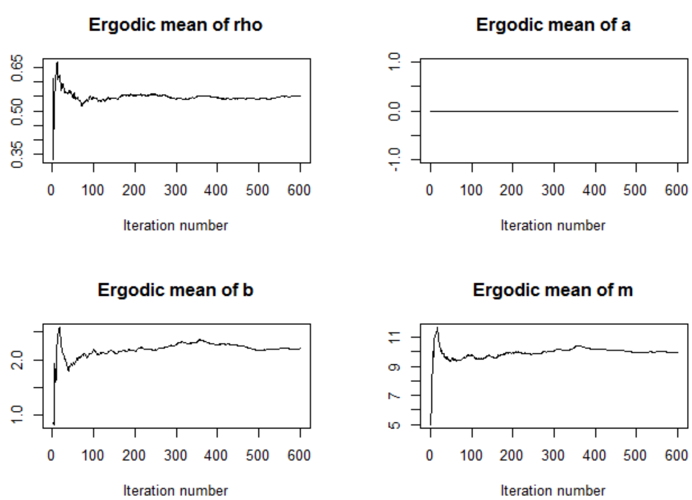

Figure 27: Ergodic Mean plots for the function `tseriescq`.

- *Cluster plots*

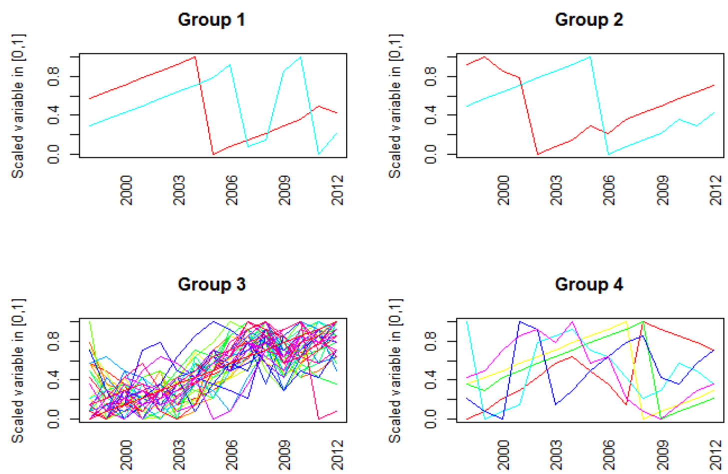

Figure 28: Cluster plots for the function `tseriesca`.

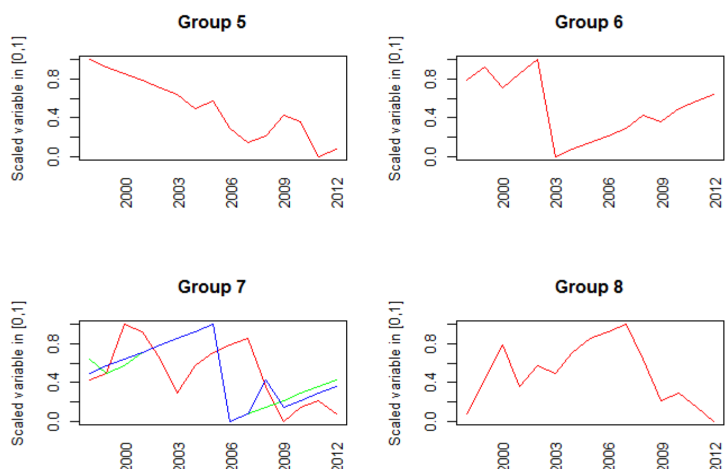

Figure 29: Cluster plots for the function `tseriesca`.

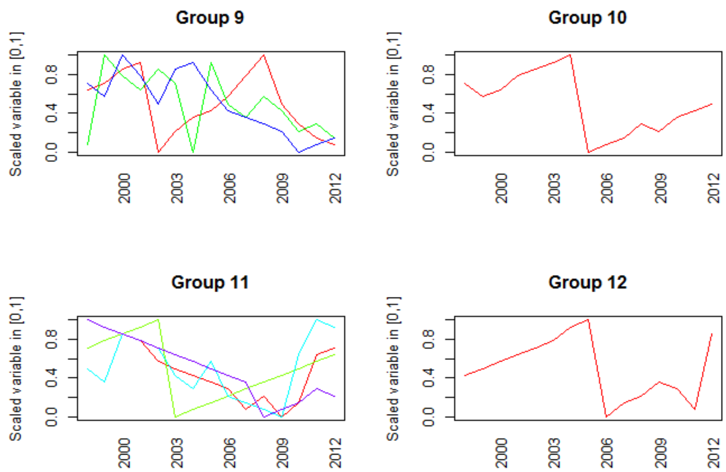

Figure 30: Cluster plots for the function `tseriesca`.

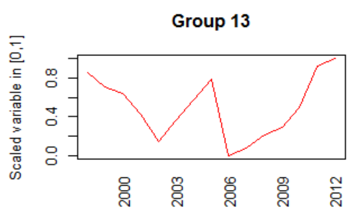

Figure 31: Cluster plots for the function `tseriesca`.

References

[1] Nieto-Barajas, L.E. & Contreras-Cristán A. (2014): *A Bayesian Nonparametric Approach for Time Series Clustering.* Bayesian Analysis, Vol. 9, No. 1 pp. 147-170.