Package ‘HMMCont’

February 19, 2015

Type Package

Title Hidden Markov Model for Continuous Observations Processes

Version 1.0

Date 2014-02-11

Author Mikhail A. Beketov

Maintainer Mikhail A. Beketov <mikhail.beketov@gmx.de>

Description The package includes the functions designed to analyse continuous observations processes with the Hidden Markov Model approach. They include Baum-Welch and Viterbi algorithms and additional visualisation functions. The observations are assumed to have Gaussian distribution and to be weakly stationary processes. The package was created for analyses of financial time series, but can also be applied to any continuous observations processes.

LazyData yes

License GPL-3

NeedsCompilation no

Repository CRAN

Date/Publication 2014-02-11 17:15:51

R topics documented:

<table>
<thead>
<tr>
<th>Function</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>HMMCont-package</td>
<td>2</td>
</tr>
<tr>
<td>baumwelchcont</td>
<td>3</td>
</tr>
<tr>
<td>hmmcontSimul</td>
<td>4</td>
</tr>
<tr>
<td>hmmsetcont</td>
<td>6</td>
</tr>
<tr>
<td>logreturns</td>
<td>7</td>
</tr>
<tr>
<td>Prices</td>
<td>8</td>
</tr>
<tr>
<td>statesDistributionsPlot</td>
<td>9</td>
</tr>
<tr>
<td>viterbicont</td>
<td>10</td>
</tr>
</tbody>
</table>

Index 11
Description

The package includes the functions designed to analyse continuous observations processes with the Hidden Markov Model approach. They include Baum-Welch and Viterbi algorithms and additional visualisation functions. The observations are assumed to have Gaussian distribution and to be weakly stationary processes. The package was created for analyses of financial time series, but can also be applied to any continuous observations processes.

Details

A Hidden Markov Model (HMM) is a statistical model in which the process being modelled is assumed to be a Markov process with unobserved, i.e. hidden states. This unobserved Markov process can be revealed from an observable process that is dependent on the states of the underlying Markov process. The HMMCont package compiles the functions that can analyse the continuous observable processes (i.e. continuous in space, discrete in time) and identify the underlying two-states Markov processes. The observable process should be weakly stationary (e.g. in case of financial time series the returns, but not the prices should be analysed). The state-dependent probabilities of the observations are modelled with Gaussian probability density functions (Rabiner, 1989). The implemented analysis procedure includes: (i) setting the initial model parameters and loading the data (function hmmsetcont), repeated execution of the Baum-Welch algorithm (function baumwelchcont), and execution of the Viterbi algorithm (viterbicont). The function baumwelchcont allows to control the model parameters after each Baum-Welch iteration, and accumulates the information on the model evolution. The model object can be analysed with tailored print, summary, and plot functions (S3 methods). For details on HMMs see the publications by Viterbi (1967), Baum et al (1970), and Rabiner (1989).

Author(s)

Mikhail A. Beketov

Maintainer: Mikhail A. Beketov <mikhail.beketov@gmx.de>

References


See Also

Functions: `hmmsetcont`, `baumwelchcont`, `viterbicont`, `statesDistributionsPlot`, and `logreturns`.

Examples

```r
# Step-by-step analysis example.
Returns<-logreturns(Prices) # Getting a stationary process
Returns<-Returns*10 # Scaling the values

hmmm<-hmmsetcont(Returns) # Creating a HMM object
print(hmmm) # Checking the initial parameters

for(i in 1:6){hmmm<-baumwelchcont(hmmm)} # Baum-Welch is # executed 6 times and results are accumulated
print(hmmm) # Checking the accumulated parameters
summary(hmmm) # Getting more detailed information

hmmcomplete<-viterbicont(hmmm) # Viterbi execution

statesDistributionsPlot(hmmcomplete, sc=10) # PDFs of # the whole data set and two states are plotted
par(mfrow=c(2,1))
plot(hmmcomplete, Prices, ylab="Price")
plot(hmmcomplete, ylab="Returns") # the revealed # Markov chain and the observations are plotted
```

baumwelchcont  

**Baum-Welch Algorithm**

Description

The function performs Baum-Welch algorithm with Gaussian PDFs (Baum et al., 1970; Rabiner, 1989). It allows to control the model parameters after each iteration, and accumulates the information on the model evolution. The intended use is to perform repeated executions and to save the returned object into the argument object (see examples below).

Usage

```r
baumwelchcont(hmm)
```

Arguments

- `hmm` An object of the class ContObservHMM.
Value

An object of the class ContObservHMM (see section on the function hmmsetcont). After sufficient number of iterations the object can be used to derive the Markov states sequence by the Viterbi algorithm (function viterbicont). The object can be analysed with the class-specific functions print, summary, and plot.

Author(s)

Mikhail A. Beketov

References


See Also

Functions: hmmsetcont, viterbicont, and statesDistributionsPlot.

Examples

```r
Returns<-logreturns(Prices) # Getting a stationary process
Returns<-Returns*10 # Scaling the values
hmm<-hmmsetcont(Returns) # Creating a HMM object
print(hmm) # Checking the initial parameters

hmm<-baumwelchcont(hmm) # First iteration
print(hmm) # Inspecting

for(i in 1:5){hmm<-baumwelchcont(hmm)} # Subsequent iterations
print(hmm) # Inspecting

hmmcomplete<-viterbicont(hmm) # Viterbi execution
par(mfrow=c(2,1))
plot(hmmcomplete, Prices, ylabel="Price")
plot(hmmcomplete, ylabel="Returns") # the revealed
# Markov chain and the observations are plotted
```

\hspace{1cm} \textbf{hmmcontSimul} \hspace{2cm} \textit{Simulation of an observation and underlying Markov processes according to a given model}
**Description**

The function simulates (i) two observation processes that correspond to the distributions of the two states in an HMM, (ii) the underlying Markov process, and (iii) an observation process that correspond to an HMM in terms of both the underlying Markov and observations processes. The function uses the last-iteration parameters, when the HMM-object was modified by function `baumwelchcont` previously.

**Usage**

```r
hmmcontSimul(hmm, n)
```

**Arguments**

- `hmm`:
  - An object of the class `ContObservHMM`
- `n`:
  - Number of observations to be simulated.

**Value**

The function returns an object of the class `SimulContHMM` that is a list comprising the two observations processes (each corresponds to the distribution of one of the two states), the Markov chain (i.e. the underlying process consisting of two states), and the observation process that correspond to that Markov chain and the two distributions. Hence, the latter is the process simulated according a given HMM.

**Author(s)**

Mikhail A. Beketov

**See Also**

Functions: `hmmsetcont`, `baumwelchcont`, and `viterbicont`.

**Examples**

```r
Returns<-logreturns(Prices)*10
hmm<-hmmsetcont(Returns)
for(i in 1:6){hmm<-baumwelchcont(hmm)}
hmmcomplete<-viterbicont(hmm)
sim<-hmmcontSimul(hmmcomplete, n=100) # simulating the processes

plot(sim$StateProcess1, type="l", ylab="State 1 Process")
plot(sim$StateProcess2, type="l", ylab="State 2 Process")
plot(sim$MarkovChain, type="l", ylab="Markov chain")
plot(sim$SimulatedObservation, type="l", ylab="Full HMM Process")
```
hmmsetcont  Setting an initial HMM object

Description

The function sets an initial Hidden Markov Model object with initial set of model parameters. It returns the object of class ContObservHMM that can be analysed with Baum-Welch (function baumwelchcont) and Viterbi algorithms (viterbicont).

Usage

hmmsetcont(Observations, Pi1 = 0.5, Pi2 = 0.5, A11 = 0.7, A12 = 0.3, A21 = 0.3, A22 = 0.7, Mu1 = 5, Mu2 = -5, Var1 = 10, Var2 = 10)

## S3 method for class 'ContObservHMM'
print(x, ...)

## S3 method for class 'ContObservHMM'
summary(object, ...)

## S3 method for class 'ContObservHMM'
plot(x, Series=x$Observations, ylabel="Observation series", xlabel="Time", ...)

Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Observations</td>
<td>Vector of observations (class &quot;numeric&quot;), a weakly stationary process (e.g. returns time series).</td>
</tr>
<tr>
<td>Pi1</td>
<td>Initial probability of state 1.</td>
</tr>
<tr>
<td>Pi2</td>
<td>Initial probability of state 2.</td>
</tr>
<tr>
<td>A11</td>
<td>Initial transition probability from state 1 to state 1.</td>
</tr>
<tr>
<td>A12</td>
<td>Initial transition probability from state 1 to state 2.</td>
</tr>
<tr>
<td>A21</td>
<td>Initial transition probability from state 2 to state 1.</td>
</tr>
<tr>
<td>A22</td>
<td>Initial transition probability from state 2 to state 2.</td>
</tr>
<tr>
<td>Mu1</td>
<td>Initial mean for Gaussian PDF for state 1.</td>
</tr>
<tr>
<td>Mu2</td>
<td>Initial mean for Gaussian PDF for state 2.</td>
</tr>
<tr>
<td>Var1</td>
<td>Initial variance for Gaussian PDF for state 1.</td>
</tr>
<tr>
<td>Var2</td>
<td>Initial variance for Gaussian PDF for state 2.</td>
</tr>
<tr>
<td>x</td>
<td>An object returned by the function hmmsetcont.</td>
</tr>
<tr>
<td>object</td>
<td>An object returned by the function hmmsetcont.</td>
</tr>
<tr>
<td>Series</td>
<td>Observations time series to be plotted along the Markov states.</td>
</tr>
<tr>
<td>ylabel</td>
<td>Y axis label.</td>
</tr>
<tr>
<td>xlabel</td>
<td>X axis label.</td>
</tr>
<tr>
<td>...</td>
<td>Not used.</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
logreturns

Value

The function returns an object of the class ContObservHMM that is a list comprising the observations, tables accumulating the model parameters and results after each Baum-Welch iterations (i.e. after each execution of the function baumwelchcont), table for the state sequence derived by the Viterbi algorithm (function viterbicont), and table of the b-probabilities. The object can be analysed with the class-specific functions print, summary, and plot.

Author(s)

Mikhail A. Beketov

See Also

Functions: baumwelchcont, viterbicont, and statesDistributionsPlot.

Examples

```r
Returns<-logreturns(Prices) # Getting a stationary process
Returns<-Returns*10 # Scaling the values

hmm<-hmmsetcont(_returns) # Creating a HMM object
print(hmm) # Checking the initial parameters

for(i in 1:6){hmm<-baumwelchcont(hmm)} # Baum-Welch is executed 6 times and results are accumulated
hmmcomplete<-viterbicont(hmm) # Viterbi execution
print(hmm) # Checking the accumulated parameters
summary(hmm) # Getting more detailed information
par(mfrow=c(2,1))
plot(hmmcomplete, Prices, ylabel="Price")
plot(hmmcomplete, ylabel="Returns") # the revealed Markov chain and the observations are plotted
```

logreturns

Calculating Log-returns

Description

Simple function that calculates log-returns from prices time series.

Usage

logreturns(x)

Arguments

x Vector of prices or an index values (class "numeric").
**Value**

Vector of log-returns (class "numeric").

**Author(s)**

Mikhail A. Beketov

**See Also**

Functions: `hmmsetcont`, `baumwelchcont`, `viterbicont`, and `statesDistributionsPlot`.

**Examples**

```r
Returns<-logreturns(Prices)
par(mfrow=c(2,1))
plot(Prices, type="l")
plot(Returns, type="l")
```

---

**Prices**

*A dummy data set of prices.*

**Description**

A dummy data set of prices or a non-stationary random process.

**Usage**

data(Prices)

**Format**

The format is: num [1:168] 1394 1366 1499 1452 1421 ...

**Source**

A dummy data set generated by the author.

**Examples**

```r
plot(Prices, type="l")
```
statesDistributionsPlot

Probability Density Functions of the States

Description

The function plots the Gaussian probability density functions from the means and variances of the whole data set, the two sub-sets corresponding to the two Markov chain states, and additionally from the HMM model (i.e. the means and variances taken form the last Baum-Welch iteration).

Usage

statesDistributionsPlot(hmm, sc = 1)

Arguments

hmm An object of the class ContObservHMM.
sc Scaling factor used when the initial HMM-object was set.

Value

Plot of the probability density functions.

Author(s)

Mikhail A. Beketov

See Also

Functions: hmmsetcont, baumwelchcont, and viterbicont.

Examples

Returns<-logreturns(Prices) # Getting a stationary process
Returns<-Returns*10 # Scaling the values
hmm<-hmmsetcont(Returns) # Creating a HMM object
for(i in 1:6){hmm<-baumwelchcont(hmm)} # Baum-Welch is # executed 6 times and results are accumulated
hmmcomplete<-viterbicont(hmm) # Viterbi execution

statesDistributionsPlot(hmmcomplete, sc=10) # PDFs of # the whole data set and two states are plotted
viterbicont

---

**Viterbi Algorithm**

**Description**

The function performs Viterbi algorithm (Viterbi, 1967). It can be applied to a ContObservHMM object after sufficient number of Baum-welch iterations (function baumwelchcont).

**Usage**

```r
viterbicont(hmm)
```

**Arguments**

- `hmm` An object of the class ContObservHMM.

**Value**

An object of the class ContObservHMM (see section on the function `hmmsetcont`). The object can be analysed with the class-specific functions `print`, `summary`, and `plot`.

**Author(s)**

Mikhail A. Beketov

**References**


**See Also**

Functions: `hmmsetcont`, `baumwelchcont`, and `statesDistributionsPlot`.

**Examples**

```r
Returns<-logreturns(Prices) # Getting a stationary process
Returns<-Returns*10 # Scaling the values
hmm<-hmmsetcont(Returns) # Creating a HMM object
for(i in 1:6){hmm<-baumwelchcont(hmm)} # Baum-Welch is executed 6 times and results are accumulated
hmmcomplete<-viterbicont(hmm) # Viterbi execution

par(mfrow=c(2,1))
plot(hmmcomplete, Prices, ylab="Price")
plot(hmmcomplete, ylab="Returns") # the revealed
# Markov chain and the observations are plotted
```
Index

*Topic **Baum-Welch**
  baumwelchcont, 3
  statesDistributionsPlot, 9

*Topic **Continuous observations**
  HMMCont-package, 2

*Topic **Econometrics**
  HMMCont-package, 2
  logreturns, 7

*Topic **Hidden Markov Model**
  baumwelchcont, 3
  HMMCont-package, 2
  hmmcontSimul, 4
  hmmsetcont, 6
  viterbicont, 10

*Topic **Quantitative Finance**
  HMMCont-package, 2

*Topic **Returns**
  logreturns, 7

*Topic **Simulation**
  hmmcontSimul, 4

*Topic **Time series**
  HMMCont-package, 2
  hmmsetcont, 6

*Topic **Viterbi**
  statesDistributionsPlot, 9
  viterbicont, 10

*Topic **datasets**
  Prices, 8

*Topic **HMMCont-package, 2**
  baumwelchcont, 3, 3, 5, 7–10
  HMMCont (HMMCont-package), 2
  HMMCont-package, 2
  hmmcontSimul, 4
  hmmsetcont, 3–5, 6, 8–10
  logreturns, 3, 7
  plot.ContObserveHMM (hmmsetcont), 6

Prices, 8
print.ContObserveHMM (hmmsetcont), 6
statesDistributionsPlot, 3, 4, 7, 8, 9, 10
summary.ContObserveHMM (hmmsetcont), 6
viterbicont, 3–5, 7–9, 10