Package: stochcorr (via r-universe)

June 4, 2026

Title Stochastic Correlation Modelling via Circular Diffusion

Version 0.0.1

Description Performs simulation and inference of diffusion processes on circle. Stochastic correlation models based on circular diffusion models are provided. For details see Majumdar, S. and Laha, A.K. (2024) ``Diffusion on the circle and a stochastic correlation model'' <doi:10.48550/arXiv.2412.06343>.

License GPL (>= 3)

Encoding UTF-8

RoxygenNote 7.3.2

Depends R (>= 3.5.0)

LinkingTo Rcpp, RcppArmadillo

Imports Rcpp, nloptr, progress, foreach, doSNOW, snow

Suggests ggplot2

Author Sourav Majumdar [aut, cre] (<https://orcid.org/0000-0002-7487-6449>)

Maintainer Sourav Majumdar <souravm@iitk.ac.in>

NeedsCompilation yes

Repository https://cran.r-universe.dev

Date/Publication 2025-04-02 17:20:01 UTC

RemoteUrl https://github.com/cran/stochcorr

RemoteRef HEAD

RemoteSha dab13767f4c0e6e2f35bdb99171fd3760cc4765b

Contents

`circ.bootstrap` *Bootstrap for circdiff*

Description

`stoch.bootstrap` returns the Bootstrap confidence interval for estimated parameters from `circdiff`.

Usage

```
circ.bootstrap(est, theta, boot_iter=500, p=0.05, seed = NULL)
```

Arguments

`est`	object of class `cbm` or `vmp`
`theta`	data of the discretely observed diffusion
`boot_iter`	number of bootstrap iteration (Default is 500)
`p`	1-p% confidence interval (Default is 0.05)
`seed`	(optional) seed value

Details

This function returns a (1-p)% confidence interval for estimated parameters from `circdiff` using parametric bootstrap. See section 4 of Majumdar and Laha (2024) doi:10.48550/arXiv.2412.06343.

Value

Returns a matrix of the bootstrap (1-p)% confidence interval for the parameters. The first row is the lower bound and the second row is the upper bound.

See Also

`circdiff()`

Examples

```
library(stochcorr)


data(wind)

if(requireNamespace("ggplot2")){
library(ggplot2)
ggplot2::ggplot(wind, aes(x = Date, y = Dir)) +
  geom_line() +
  labs(title = "Sotavento Wind Farm",
       x = "Date",
       y = "Wind Direction") +
  scale_x_datetime(date_labels = "%d-%b", date_breaks = "2 day") +
  theme_test() +
  theme(
    text = element_text(size = 15),
    axis.text.x = element_text(angle = 90, hjust = 1)
  )
}

a<-circdiff(wind$Dir,10/1440,"vmp")
a

b<-circdiff(wind$Dir,10/1440,"cbm")
b



estimates_vmp<-circ.bootstrap(a,wind$Dir, seed = 100)
estimates_vmp

estimates_cbm<-circ.bootstrap(b,wind$Dir, seed = 100)
estimates_cbm
```

---

circdiff — *Estimation of circular diffusion models*

Description

`circdiff` returns the estimates of parameters of a discretely observed von-Mises process or Circular Brownian motion

Usage

```
circdiff(theta, dt, corr_process, iter=2000, lambda=0, lambda_1=0, lambda_2=0)
```

Arguments

theta	data of the discretely observed diffusion
dt	time step Δt
corr_process	vmp for von-Mises process; cbm for circular brownian motion
iter	number of iterations (Default is 2000)
lambda	regularization parameter for cbm (Default is 0)
lambda_1	regularization parameter for vmp (Default is 0)
lambda_2	regularization parameter for vmp (Default is 0)

Details

Let $\theta_0, \theta_{\Delta t}, \theta_{2\Delta t}, \ldots, \theta_{n\Delta t}$ be a discretely observed circular diffusion at time step Δt. The circular diffusion could either be von-Mises process,

$$d\theta_t = -\lambda \sin(\theta_t - \mu)dt + \sigma dW_t$$

or the circular brownian motion,

$$d\theta_t = \sigma dW_t$$

under periodic boundary condition. The function returns the MLE of λ, σ, μ for von-Mises process or σ for circular brownian motion.

We provide an option to perform penalised MLE using an iterative optimization procedure as following, we maximise for von Mises process, where n is the number of observations,

$$\text{Log-lik} - n\lambda_1 \sum (1 - cos(\theta_{i+1} - \theta_i)) - \lambda_2 \frac{2\lambda}{\sigma^2}$$

For Circular Brownian motion we maximise,

$$\text{Log-lik} - n\lambda \sum (1 - cos(\theta_{i+1} - \theta_i))$$

See section 3 of Majumdar and Laha (2024) doi:10.48550/arXiv.2412.06343.

Value

A list containing the estimates of the model if corr_process=vmp then it returns

- dt time step Δt
- lambda_vm the drift parameter of the von Mises process
- sigma_vm the volatility parameter of the von Mises process
- mu_vm the mean direction of the von Mises process
- lambda_1, lambda_2 value of the regularization parameters used for estimation

else if corr_process=cbm then it returns

- dt time step Δt
- sigma_cbm the volatility parameter of the circular Brownian motion
- lambda value of the regularization parameter used for estimation

See Also

circ.bootstrap()

Examples

```
library(stochcorr)

data(wind)
if(requireNamespace("ggplot2")){
library(ggplot2)
ggplot2::ggplot(wind, aes(x = Date, y = Dir)) +
  geom_line() +
  labs(title = "Sotavento Wind Farm",
       x = "Date",
       y = "Wind Direction") +
  scale_x_datetime(date_labels = "%d-%b", date_breaks = "2 day") +
  theme_test() +
  theme(
    text = element_text(size = 15),
    axis.text.x = element_text(angle = 90, hjust = 1)
  )
 }

a<-circdiff(wind$Dir,10/1440,"vmp")
a

b<-circdiff(wind$Dir,10/1440,"cbm")
b
```

---

dcbm — *Probability transition density function for Circular Brownian Motion*

Description

dcbm evaluates the transition density for Circular Brownian Motion

Usage

```
dcbm(theta, t, theta_0, sigma)
```

Arguments

theta	vector of data
t	time step Δt
theta_0	initial value
sigma	sigma

Details

See section 2 of Majumdar and Laha (2024) doi:10.48550/arXiv.2412.06343.

Value

dcbm gives the transition density function of the circular Brownian motion

---

dvmp — *Probability transition density function for von Mises process*

Description

dvmp evaluates the transition density for von Mises process

Usage

```
dvmp(theta, t, theta_0, lambda, sigma, mu)
```

Arguments

theta	vector of data
t	time step Δt
theta_0	initial value
lambda	lambda
sigma	sigma
mu	mu

Details

See section 2 of Majumdar and Laha (2024) doi:10.48550/arXiv.2412.06343.

Value

dvmp gives the transition density function of the von Mises process

ftse2020 *2020 USD/GBP and FTSE250 data*

Description

End of the day closing price for USD/GBP and FTSE250 in the year 2020.

Usage

```
data("ftse2020")
```

Format

A data frame with 224 rows and 3 columns:

`USD/GBP` EOD USD/GBP rate

`FTSE250` EOD FTSE250

`Date` Date

Source

Yahoo Finance

nikkei2020 *2020 USD/JPY and Nikkei data*

Description

End of the day closing price for USD/JPY and Nikkei in the year 2020.

Usage

```
data("nikkei2020")
```

Format

A data frame with 213 rows and 3 columns:

`USD/JPY` EOD USD/JPY rate

`Nikkei` EOD Nikkei

`Date` Date

Source

Yahoo Finance

nse2020 — *2020 USD/INR and NIFTY data*

Description

End of the day closing price for USD/INR and NIFTY in the year 2020.

Usage

```
data("nse2020")
```

Format

A data frame with 250 rows and 3 columns:

`USD/INR` EOD USD/INR rate

`Nifty` EOD Nifty

`Date` Date

Source

Yahoo Finance

rtraj.cbm — *Simulate circular Brownian motion*

Description

`rtraj.cbm` returns a simulated path of a circular Brownian motion for given parameters

Usage

```
rtraj.cbm(n, theta_0, dt, sigma, burnin=1000)
```

Arguments

`n`	number of steps in the simulated path
`theta_0`	initial point
`dt`	Time step
`sigma`	volatility parameter
`burnin`	number of initial samples to be rejected (Default is 1000)

Details

Let θ_t evolve according to a circular Brownian motion given by,

$$d\theta_t = \sigma dW_t$$

We simulate θ_t by simulating from its transition density.

Value

A vector of length `n` of the simulated path from circular Brownian motion

---

`rtraj.vmp` *Simulate von Mises process*

Description

`rtraj.vmp` returns a simulated path of a von Mises process for given parameters

Usage

```
rtraj.vmp(n, theta_0, dt, mu, lambda, sigma)
```

Arguments

`n`	number of steps in the simulated path
`theta_0`	initial point
`dt`	Time step
`mu`	mean parameter
`lambda`	drift parameter
`sigma`	volatility parameter

Details

Let θ_t evolve according to a von Mises process given by,

$$d\theta_t = -\lambda \sin(\theta_t - \mu)dt + \sigma dW_t$$

We simulate θ_t by the Euler-Maruyama discretization of the above SDE.

Value

A vector of length `n` of the simulated path from von Mises process

s&p2020 — *2020 EUR/USD and S&P500 data*

Description

End of the day closing price for EUR/USD and S&P500 in the year 2020.

Usage

```
data("s&p2020")
```

Format

A data frame with 224 rows and 3 columns:

`EUR/USD` EOD EUR/USD rate

`S&P500` EOD S&P500

`Date` Date

Source

Yahoo Finance

stoch.bootstrap — *Bootstrap for stochcorr*

Description

`stoch.bootstrap` returns the Bootstrap confidence interval for estimated rho from `stochcorr`.

Usage

```
stoch.bootstrap(est, S_1, S_2, boot_iter=500, p=0.05, seed = NULL)
```

Arguments

`est`	object of class `cbm` or `vmp`
`S_1`	historical price of the first asset
`S_2`	historical price of the second asset
`boot_iter`	number of bootstrap iteration (Default is 500)
`p`	1-p% confidence interval (Default is 0.05)
`seed`	(optional) seed value

Details

This function returns a p% confidence interval for estimated rho from `stochcorr` using parametric bootstrap. See section 4 of Majumdar and Laha (2024) doi:10.48550/arXiv.2412.06343.

Value

Returns a matrix for the bootstrap (1-p)% confidence interval for rho. The first row of the matrix is the lower bound and the second row is the upper bound.

See Also

`stochcorr()`

Examples

```
library(stochcorr)

data("nse2020")

## using von Mises process as the correlation process

a <- stochcorr(nse2020$`USD/INR`, nse2020$Nifty, 1 / 250, corr_process = "vmp")
b <- stoch.bootstrap(a, nse2020$`USD/INR`, nse2020$Nifty, seed = 100)

rho_data <- as.data.frame(cbind(a$rho, nse2020$Date))
rho_data[, 2] <- as.Date(rho_data[, 2], origin = "1970-01-01")
colnames(rho_data) <- c("Correlation", "Time")

if(requireNamespace("ggplot2")){
library(ggplot2)
ggplot2::ggplot(rho_data, aes(x = Time, y = Correlation)) +
  theme_test() +
  theme(
    text = element_text(size = 15),
    axis.text.x = element_text(angle = 90, hjust = 1)
  ) +
  geom_line() +
  geom_ribbon(aes(ymin = b[1, ], ymax = b[2, ]), fill = "blue", alpha = 0.15) +
  scale_y_continuous(breaks = round(seq(-1, 1, by = 0.05), 1)) +
  scale_x_date(breaks = "1 month", date_labels = "%B %Y")
  }

## using Circular Brownian Motions as the correlation process

a <- stochcorr(nse2020$`USD/INR`, nse2020$Nifty, 1 / 250, corr_process = "cbm")
b <- stoch.bootstrap(a, nse2020$`USD/INR`, nse2020$Nifty, seed = 100)

rho_data <- as.data.frame(cbind(a$rho, nse2020$Date))
rho_data[, 2] <- as.Date(rho_data[, 2], origin = "1970-01-01")
colnames(rho_data) <- c("Correlation", "Time")

if(requireNamespace("ggplot2")){
```

```
library(ggplot2)
ggplot2::ggplot(rho_data, aes(x = Time, y = Correlation)) +
  theme_test() +
  theme(
    text = element_text(size = 15),
    axis.text.x = element_text(angle = 90, hjust = 1)
  ) +
  geom_line() +
  geom_ribbon(aes(ymin = b[1, ], ymax = b[2, ]), fill = "blue", alpha = 0.15) +
  scale_y_continuous(breaks = round(seq(-1, 1, by = 0.05), 1)) +
  scale_x_date(breaks = "1 month", date_labels = "%B %Y")}
```

stochcorr — *Estimate a stochastic correlation model*

Description

`stochcorr` returns the estimates of the instantaneous correlation and other model parameters.

Usage

```
stochcorr(S_1, S_2, dt, corr_process, iter=2000, lambda=4, lambda_1=10, lambda_2=0)
```

Arguments

`S_1`	Historical price of the first asset
`S_2`	Historical price of the second asset
`dt`	Time step
`corr_process`	specify the correlation process, `vmp` for von Mises process or `cbm` for Circular Brownian Motion
`iter`	Number of iteration (Default is 2000)
`lambda`	regularization parameter for circular Brownian motion (Default is 4)
`lambda_1`	(Default is 10)
`lambda_2`	0 (Default)

Details

Let S_t^1 and S_t^2 be two discretely observed geometric Brownian motions observed at a time step of dt.

$$dS_t^1 = \mu_1 S_t^1 dt + \sigma_1 dW_t^1 dS_t^2 = \mu_2 S_t^2 dt + \sigma_2(\rho_t dW_t^1 + \sqrt{1-\rho_t^2} dW_t^2)$$

where $\rho_t = \cos\theta_t$, with θ_t being specified by the von Mises Process ($d\theta_t = \lambda\sin(\theta_t - \mu) + \sigma dW_t^3$) or the Circular Brownian Motion,

$$d\theta = \sigma dW_t^3$$

with periodic boundary conditions. Here W_t^1, W_t^2, W_t^3 are mutually independent Brownian Motions.

We estimate the model by maximising penalized MLE, using an iterative optimization procedure. In case of the von Mises process as the correlation process we maximise,

$$\text{Log-lik} - \lambda_1 \sum (\rho_{i+1} - \rho_i)^2 - \lambda_2 \frac{2\lambda}{\sigma^2}$$

For Circular Brownian motion as the correlation process we maximise,

$$\text{Log-lik} - \lambda \sum (\rho_{i+1} - \rho_i)^2$$

See section 4 of Majumdar and Laha (2024) doi:10.48550/arXiv.2412.06343.

Value

A list containing the estimates of the model including the asset price parameters, instantaneous correlations and estimates of the correlation process

- `rho` Estimated instantaneous correlations
- `mu_1` Drift of the first asset
- `mu_2` Drift of the second asset
- `sigma_1` Volatility of the first asset
- `sigma_2` Volatility of the second asset

if `corr_process=vmp` then it additionally returns

- `lambda_vm` the drift parameter of the von Mises process
- `sigma_vm` the volatility parameter of the von Mises process
- `mu_vm` the mean direction of the von Mises process
- `lambda_1, lambda_2` value of the regularization parameters used for estimation

else if `corr_process=cbm` then it returns

- `sigma_cbm` the volatility parameter of the circular Brownian motion
- `lambda` value of the regularization parameter used for estimation

See Also

`stoch.bootstrap()`

Examples

```
data("nse2020")

## using von Mises process as the correlation process

a <- stochcorr(nse2020$`USD/INR`, nse2020$Nifty, 1 / 250, corr_process = "vmp")

## using Circular Brownian Motions as the correlation process

a <- stochcorr(nse2020$`USD/INR`, nse2020$Nifty, 1 / 250, corr_process = "cbm")
```

stochcorr.sim *Simulate stochastic correlation model*

Description

`stochcorr.sim` returns the paths of stock price under a stochastic correlation model

Usage

```
stochcorr.sim(m=500, n, dt, S1_0, S2_0, mu1, sigma1, mu2, sigma2,
mu, lambda, sigma, corr_process)
```

Arguments

`m`	number of paths (Default is 500)
`n`	number of steps in each simulated path
`dt`	time step
`S1_0`	initial price of the first asset
`S2_0`	initial price of the second asset
`mu1`	drift of the first asset
`sigma1`	volatility of the first asset
`mu2`	drift of the second asset
`sigma2`	volatility of the second asset
`mu`	mean direction of the correlation process (if `corr_process=vmp`)
`lambda`	drift of the correlation process (if `corr_process=vmp`)
`sigma`	volatility of the correlation process (if `corr_process=vmp` or `corr_process=cbm`)
`corr_process`	specify the correlation process, `vmp` for von Mises process or `cbm` for Circular Brownian Motion

Details

This function returns the simulated paths of two stock prices following a stochastic correlation model. See `stochcorr()` details of the stochastic correlation model

Value

Returns a list with prices of two assets `S1` and `S2` under the stochastic correlation model

Examples

```
library(stochcorr)
# Generate 500 paths of two geometric Brownian motions, S1 and S2, of length 100 each
# following the von Mises process with mu=pi/2, lambda=1 and sigma =1

a<-stochcorr.sim(m=500,100,0.01,100,100,0.05,0.05,0.06,0.1,pi/2,1,1,"vmp")
t<-seq(0,100*0.01-0.01,0.01)

# Plot the first realization of S1 and S2

plot(t,a$S1[1,], ylim=c(min(a$S1[1,],a$S2[1,]),max(a$S1[1,],a$S2[1,])),type="l")
lines(t,a$S2[1,], col="red",type="l")
legend(0.01,max(a$S1[1,],a$S2[1,]), legend = c("S1","S2"), col = c("black", "red"), lty=1)
```

`wind` *Wind direction data*

Description

Wind direction data from Sotavento Wind farm from 1 November 2024 to 30 November 2024 at a 10 minute frequency.

Usage

```
data("wind")
```

Format

A data frame with 4320 rows and 2 columns:

`Dir` Wind Direction data

`Date` Date

Source

https://www.sotaventogalicia.com/en/technical-area/real-time-data/historical/

Index