Package 'armaOptions'

July 22, 2025

Type Package

Title ARMA Models to Value Stock Options

Version 1.0.0

Maintainer Brian MacCarvill <brianmaccarvills@gmail.com>

Description Providing ways to estimate the value of European stock options
given historical stock price data. It includes functions for calculating option values
based on autoregressive–moving-
average (ARMA) models and generates information about these models.
This package is make to be easy to understand and for financial analysis capabilities.

License GPL-3

Encoding UTF-8

Imports forecast, stats

RoxygenNote 7.3.1

NeedsCompilation no

Author Brian MacCarvill [aut, cre]

Repository CRAN

Date/Publication 2025-07-11 12:40:06 UTC

Contents

`CallOptionsOverStrikePrices` *Call Option Values for Differently Priced European Call Options*

Description

This function calculates the value of the a European call option for a list of strike price / buy values, given stock price data and a given future time.

Usage

```
CallOptionsOverStrikePrices(
  stock_data,
  future_time,
  buy_values,
  max.p = 5,
  max.q = 5,
  method = "CSS-ML"
)
```

Arguments

`stock_data`	Numeric vector of stock prices data.
`future_time`	Numeric constant of the future time
`buy_values`	Numeric vector of the buy values at which to calculate the call option values
`max.p`	The maximum order of the Auto Regressive part of the ARMA model (default is set to 5)
`max.q`	The maximum order of the Moving Average part of the ARMA model (default is set to 5)
`method`	The way that the ARMA model is calculated, accepted values are "ML", "CSS-ML" and "CSS"

Value

Estimated values of a European call option at different buy values

Examples

```
library(stats)
library(forecast)

# Create simulated data
n = 100
set.seed(42)
arma_values = arima.sim(n = n, model = list(ar = c(0.6), ma = c(0.5, -0.5)))
linear_model = 5 + 1:n
```

```
stock_data = arma_values + linear_model

future_time = 3
buy_values = seq(90, 110, length.out = 5)

CallOptionsOverStrikePrices(stock_data, future_time, buy_values)
```

`CallOptionsOverTime` *Time Sensitivity Analysis for European Call Option*

Description

This function calculates the value of the a European call option for a list of future time values, given stock price data and a given buy value.

Usage

```
CallOptionsOverTime(
  stock_data,
  future_times,
  buy_value,
  max.p = 5,
  max.q = 5,
  method = "CSS-ML"
)
```

Arguments

`stock_data`	Numeric vector of stock prices data.
`future_times`	Numeric vector of the future times
`buy_value`	Numeric value representing the buy value
`max.p`	The maximum order of the autoregressive part of the ARMA model (default is 5).
`max.q`	The maximum order of the moving average part of the ARMA model (default is 5).
`method`	The way that the ARMA model is calculated, accepted values are "ML", "CSS-ML" and "CSS"

Value

Estimated values of a European call option at different future times

Examples

```
library(stats)
library(forecast)

# Create simulated data
n = 100
set.seed(42)
arma_values = arima.sim(n = n, model = list(ar = c(0.6), ma = c(0.5, -0.5)))
linear_model = 5 + 1:n
stock_data = arma_values + linear_model

future_times = c(1,3,5)
buy_value = 105


CallOptionsOverTime(stock_data, future_times, buy_value)
```

`europeanCallOptionValue` *Estimate European Call Option Value*

Description

This function calculates the value of a European call option based on stock data, a future time value, and a buy value

Usage

```
europeanCallOptionValue(
  stock_data,
  future_time,
  buy_value,
  max.p = 5,
  max.q = 5,
  method = "CSS-ML"
)
```

Arguments

`stock_data`	Numeric vector of stock prices data
`future_time`	Numeric constant of the future time
`buy_value`	The numeric buy value of the European call option
`max.p`	The maximum order of the autoregressive part of the ARMA model (default is set to 5)
`max.q`	The maximum order of the moving average part of the ARMA model (default is set to 5)

method	The way that the ARMA model is calculated, accepted values are "ML", "CSS-ML" and "CSS"

Value

Estimate the value of a European call option, determine the probability of making profits, and model an appropriate ARMA model for the given stock data

Examples

```
library(stats)
library(forecast)
# Create simulated data
n = 100
set.seed(42)
arma_values = arima.sim(n = n, model = list(ar = c(0.5), ma = c(0.5, -0.5)))
linear_model = 5 +  1:n
stock_data = arma_values + linear_model
buy_value = 105
future_time = 1
europeanCallOptionValue(stock_data = stock_data, future_time, buy_value, max.p = 5, max.q = 5)
```

---

europeanPutOptionValue

Estimate European Put Option Value

Description

This function calculates the value of a European put option based on stock data, a future time value, and a sell value

Usage

```
europeanPutOptionValue(
  stock_data,
  future_time,
  sell_value,
  max.p = 5,
  max.q = 5,
  method = "CSS-ML"
)
```

Arguments

stock_data	Numeric vector of stock prices data
future_time	Numeric constant of the future time
sell_value	The numeric sell value of the European put option.

max.p	The maximum order of the autoregressive part of the ARMA model (default is set to 5)
max.q	The maximum order of the moving average part of the ARMA model (default is set to 5)
method	The way that the ARMA model is calculated, accepted values are "ML", "CSS-ML" and "CSS"

Value

Estimate the value of a European put option, determine the probability of making profits, and model an appropriate ARMA model for the given stock data.

Examples

```
library(stats)
library(forecast)
# Create simulated data
n = 100
set.seed(42)
arma_values = arima.sim(n = n, model = list(ar = c(0.6), ma = c(0.5, -0.5)))
linear_model = 5 + 1:n
stock_data = arma_values + linear_model
europeanPutOptionValue(stock_data = stock_data,future_time = 5,sell_value = 110,max.p = 5,max.q = 5)
```

PutOptionsOverStrikePrices

Strike Price Sensitivity Analysis for European Put Option

Description

This function calculates the value of the a European put option for a list of strike price / sell values, given stock price data and a given future time.

Usage

```
PutOptionsOverStrikePrices(
  stock_data,
  future_time,
  sell_values,
  max.p = 5,
  max.q = 5,
  method = "CSS-ML"
)
```

Arguments

stock_data	Numeric vector of stock prices data.
future_time	Numeric constant of the future time
sell_values	Numeric vector of the sell values to calculate the put option values at
max.p	The maximum order of the autoregressive part of the ARMA model (default is 5)
max.q	The maximum order of the moving average part of the ARMA model (default is 5)
method	The way that the ARMA model is calculated, accepted values are "ML", "CSS-ML" and "CSS"

Value

Estimated values of a European put option at different sell values

Examples

```
library(stats)
library(forecast)

n = 100
set.seed(42)
arma_values = arima.sim(n = n, model = list(ar = c(0.6), ma = c(0.5, -0.5)))
linear_model = 5 + 1:n
stock_data = arma_values + linear_model


future_time = 2
sell_values = seq(90, 110, length.out = 5)

PutOptionsOverStrikePrices(stock_data, future_time, sell_values)
```

PutOptionsOverTime *Time Sensitivity Analysis for European Call Option*

Description

This function calculates the value of the a European put option for a list of future time values, given stock price data and a given buy value.

Usage

```
PutOptionsOverTime(
  stock_data,
  future_times,
  sell_value,
  max.p = 5,
  max.q = 5,
  method = "CSS-ML"
)
```

Arguments

stock_data	Numeric vector of stock prices data.
future_times	Numeric vector of the future times
sell_value	Numeric value representing the sell value
max.p	The maximum order of the autoregressive part of the ARMA model (default is 5).
max.q	The maximum order of the moving average part of the ARMA model (default is 5).
method	The way that the ARMA model is calculated, accepted values are "ML", "CSS-ML" and "CSS"

Value

Estimated values of a European put option at different future times

Examples

```
library(stats)
library(forecast)

# Create simulated data
n = 100
set.seed(42)
arma_values = arima.sim(n = n, model = list(ar = c(0.6), ma = c(0.5, -0.5)))
linear_model = 5 + 1:n
stock_data = arma_values + linear_model

sell_value = 110
future_times = c(1, 3, 5)

PutOptionsOverTime(stock_data = stock_data, future_times = future_times, sell_value = sell_value)
```

Index