Package 'roptions'

October 14, 2022

Title Option Strategies and Valuation

Version 1.0.3

Description Collection of tools to develop options strategies, value option contracts using the Black-Scholes-Merten option pricing model and calculate the option Greeks. Hull, John C. ``Options, Futures, and Other Derivatives" (1997, ISBN:0-13-601589-1). Fischer Black, Myron Scholes (1973) ``The Pricing of Options and Corporate Liabilities" <doi:10.1086/260062>.

License GPL-3

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

Imports purrr, ggplot2, plotly, stats

Depends R (>= 2.10)

NeedsCompilation no

Author Anurag Agrawal [aut, cre] (<https://orcid.org/0000-0003-2272-8273>)

Maintainer Anurag Agrawal <agrawalanurag1999@gmail.com>

Repository CRAN

Date/Publication 2020-05-11 11:10:06 UTC

R topics documented:

`box.spread` *Box Spread Strategy Function*

Description

This function can be used to develop a box spread strategy for options. A box spread is an options arbitrage strategy that combines buying a bull call spread with a matching bear put spread

Usage

```
box.spread(
  k_long_call,
  k_short_call,
  k_long_put,
  k_short_put,
  c1,
  c2,
  p1,
  p2,
  llimit = 20,
  ulimit = 20
)
```

Arguments

k_long_call	Excercise Price of Long call Option
k_short_call	Excercise Price of Short call Option
k_long_put	Excercise Price of Long Put Optioon
k_short_put	Excercise Price of Short Put Option
c1	Premium of Long Call Option
c2	Premium of Short Call Option
p1	Premium of Long Put Option
p2	Premium of Short Put Option
llimit	Lower limit of stock price at Expiration., Default: 20
ulimit	Upper Limit of Stock Price at Expiration, Default: 20

Details

To construct a box spread, a trader buys an in-the-money (ITM) call, sells an out-of-the-money (OTM) call, buys an ITM put and sells an OTM put. In other words, buy an ITM call and put and then sell an OTM call and put.

Value

Returns the profit/loss generated from the strategy along with the profit/loss of individual contract and an interactive graph for the same.

Examples

```
box.spread(100, 105, 95, 110, 3.2, 2.6, 1.1, 2.4)
```

---

butterfly.call *Butterfly Call Spread Strategy Function*

Description

This function can be used to develop a Butterfly call Spread Strategy.

Usage

```
butterfly.call(
  k1,
  k2,
  k3,
  c1,
  c2,
  c3,
  spread = c("long", "short"),
  llimit = 20,
  ulimit = 20
)
```

Arguments

k1	Excercise Price of 1st Long call Option (Long Spread)/ Excercise Price of 1st Short call Option (Short Spread)
k2	Excercise Price of Short call Option (Long Spread) / Excercise Price of Long call Option (Short Spread)
k3	Excercise Price of 2nd Long call Option (Long Spread) / Excercise Price of 2nd Short call Option (Short Spread)
c1	Premium of 1st Long call Option (Long Spread)/ Premium of 1st Short call Option (Short Spread)
c2	Premium of Short call Option (Long Spread) / Premium of Long call Option (Short Spread)
c3	Premium of 2nd Long call Option (Long Spread) / Premium of 2nd Short call Option (Short Spread)
spread	Type of Spread, Default: c("long", "short")
llimit	Lower limit of stock price at Expiration., Default: 20
ulimit	Upper Limit of Stock Price at Expiration, Default: 20

Details

The long butterfly call spread is created by buying one in-the-money call option with a low strike price, writing two at-the-money call options, and buying one out-of-the-money call option with a higher strike price. The short butterfly spread is created by selling one in-the-money call option with a lower strike price, buying two at-the-money call options, and selling an out-of-the-money call option at a higher strike price.

Value

OUTPUT_DESCRIPTION Returns the profit/loss generated from the strategy along with the profit/loss of individual contract and an interactive graph for the same.

Examples

```
butterfly.call(100, 95, 105, 2.3, 1.25, 3.2, spread = 'long')
```

---

butterfly.put *Butterfly Put Spread Strategy Function*

Description

This function can be used to develop a Butterfly Put Spread Strategy

Usage

```
butterfly.put(
  k1,
  k2,
  k3,
  p1,
  p2,
  p3,
  spread = c("long", "short"),
  llimit = 20,
  ulimit = 20
)
```

Arguments

k1	Excercise Price of 1st Long Put Option (Long Spread)/ Excercise Price of 1st Short Put Option (Short Spread)
k2	Excercise Price of Short Put Option (Long Spread) / Excercise Price of Long Put Option (Short Spread)
k3	Excercise Price of 2nd Long Put Option (Long Spread) / Excercise Price of 2nd Short Put Option (Short Spread)
p1	Premium of 1st Long Put Option (Long Spread)/ Premium of 1st Short Put Option (Short Spread)
p2	Premium of Short Put Option (Long Spread) / Premium of Long Put Option (Short Spread)
p3	Premium of 2nd Long Put Option (Long Spread) / Premium of 2nd Short Put Option (Short Spread)
spread	Type of Spread, Default: c("long", "short")
llimit	Lower limit of stock price at Expiration., Default: 20
ulimit	Upper Limit of Stock Price at Expiration, Default: 20

Details

The long put butterfly spread is created by buying one put with a lower strike price, selling two at-the-money puts, and buying a put with a higher strike price. Net debt is created when entering the position. The short put butterfly spread is created by writing one out-of-the-money put option with a low strike price, buying two at-the-money puts, and writing an in-the-money put option at a higher strike price.

Value

OUTPUT_DESCRIPTION Returns the profit/loss generated from the strategy along with the profit/loss of individual contract and an interactive graph for the same.

Examples

```
butterfly.put(100, 105, 95, 2.2, 3.2, 1.25, spread = 'long')
```

call.delta *Call Delta*

Description

Calculate the Delta (Option Greek) of a Contract

Usage

```
call.delta(s, k, t, sd, r, d = 0)
```

Arguments

s	Spot Price of Underlying Asset
k	Exercise Price of Contract
t	Time to Expiration
sd	Volatality
r	Risk free rate of return
d	Divident Yield (use cont.rate()), Default: 0

Details

Delta represents the rate of change between the option's price and a $1 change in the underlying asset's price.

Value

Oupput gives the delta of a Option Contract.

Examples

```
call.delta(100, 105, 0.25, 0.35, 0.0488)
```

call.estimate *Option Greek and Estimated Premium of Call Option*

Description

Calculate the Option Greek of a Contract and Estimated Premium of Contract

Usage

```
call.estimate(s, k, t, sd, r, d = 0)
```

Arguments

s	Spot Price of Underlying Asset
k	Exercise Price of Contract
t	Time to Expiration
sd	Volatality
r	Risk free rate of return
d	Divident Yield (use cont.rate()), Default: 0

Details

"Greeks" is a term used in the options market to describe the different dimensions of risk involved in taking an options position. These Greeks are calculated in this function along with the premium of the option contract using the BSM Model.

Value

Output gives the Option Greek of a Option Contract. Also the Premium of the contract is estimated.

Examples

```
call.estimate(100, 105, 0.25, 0.35, 0.0488)
```

---

call.gamma *Call Gamma*

Description

Calculate the Gamma (Option Greek) of a Contract

Usage

```
call.gamma(s, k, t, sd, r, d = 0)
```

Arguments

s	Spot Price of Underlying Asset
k	Exercise Price of Contract
t	Time to Expiration
sd	Volatality
r	Risk free rate of return
d	Divident Yield (use cont.rate()), Default: 0

Details

Gamma represents the rate of change between an option's delta and the underlying asset's price.

Value

Output gives the Gamma of a Option Contract.

Examples

```
call.gamma(100, 105, 0.25, 0.35, 0.0488)
```

`call.greek` *Specified Call Option Greek*

Description

Calculate the Specified Option Greek of a Contract

Usage

```
call.greek(
  greek = c("delta", "gamma", "theta", "vega", "rho"),
  s,
  k,
  t,
  sd,
  r,
  d = 0
)
```

Arguments

`greek`	Character String of the greek to be calculated
`s`	Spot Price of Underlying Asset
`k`	Exercise Price of Contract
`t`	Time to Expiration
`sd`	Volatality
`r`	Risk free rate of return
`d`	Divident Yield (use cont.rate()), Default: 0

Details

Delta represents the rate of change between the option's price and a $1 change in the underlying asset's price. Theta represents the rate of change between the option price and time, or time sensitivity - sometimes known as an option's time decay. Gamma represents the rate of change between an option's delta and the underlying asset's price. Vega represents the rate of change between an option's value and the underlying asset's implied volatility. Rho represents the rate of change between an option's value and a 1% change in the interest rate.

Value

Output gives the Specified Greek of a Option Contract.

Examples

```
call.greek('delta', 100, 105, 0.25, 0.35, 0.0488)
```

`call.minorgreek` *Specified Minor Option Greek*

Description

Calculate the Specified Minor Option Greek of a Contract

Usage

```
call.minorgreek(minorgreek = c("lambda", "vomma"), s, k, t, sd, r, d = 0)
```

Arguments

`minorgreek`	Character String of the minor greek to be calculated
`s`	Spot Price of Underlying Asset
`k`	Exercise Price of Contract
`t`	Time to Expiration
`sd`	Volatality
`r`	Risk free rate of return
`d`	Divident Yield (use cont.rate()), Default: 0

Details

Vomma is the rate at which the vega of an option will react to volatility in the market. In options trading, Lamba is the Greek letter assigned to variable which tells the ratio of how much leverage an option is providing as the price of that option changes.

Value

Output gives the Specified Minor Greek of a Option Contract.

Examples

```
call.minorgreek('lambda', 100, 105, 0.25, 0.35, 0.0488)
```

`call.premium.est` *Estimated Premium of Option Contract*

Description

Calculate the Estimated Premium of Option Contract

Usage

```
call.premium.est(s, k, t, sd, r, d = 0)
```

Arguments

`s`	Spot Price of Underlying Asset
`k`	Exercise Price of Contract
`t`	Time to Expiration
`sd`	Volatality
`r`	Risk free rate of return
`d`	Divident Yield (use cont.rate()), Default: 0

Details

Estimate is calculated based on Black-Scholes Model. The Black Scholes model, also known as the Black-Scholes-Merton (BSM) model, is a mathematical model for pricing an options contract.

Value

Output gives the Estimated Premium of a Option Contract.

Examples

```
call.premium.est(100, 105, 0.25, 0.35, 0.0488)
```

`call.rho` *Call Rho*

Description

Calculate the Rho (Option Greek) of Option Contract

Usage

```
call.rho(s, k, t, sd, r, d = 0)
```

Arguments

s	Spot Price of Underlying Asset
k	Exercise Price of Contract
t	Time to Expiration
sd	Volatality
r	Risk free rate of return
d	Divident Yield (use cont.rate()), Default: 0

Details

Rho represents the rate of change between an option's value and a 1% change in the interest rate.

Value

Output gives the Rho of a Option Contract.

Examples

```
call.rho(100, 105, 0.25, 0.35, 0.0488)
```

---

call.spread *Bull/Bear Call Spread Strategy Function*

Description

This function can be used to develop a Bull/Bear Call Strategy.

Usage

```
call.spread(k1, k2, c1, c2, llimit = 20, ulimit = 20)
```

Arguments

k1	Excercise Price of Long call Option
k2	Excercise Price of Short Call Option
c1	Premium of Long call Option
c2	Premium of Short Call Option
llimit	Lower limit of stock price at Expiration., Default: 20
ulimit	Upper Limit of Stock Price at Expiration, Default: 20

Details

Bull Call Spread uses two call options to create a range consisting of a lower strike price and an upper strike price.bear call spread is achieved by purchasing call options at a specific strike price while also selling the same number of calls with the same expiration date, but at a lower strike price.

Value

OUTPUT_DESCRIPTION Returns the profit/loss generated from the strategy along with the profit/loss of individual contract and an interactive graph for the same.

Examples

```
call.spread(1.2, 3.2, 100, 105)
```

`call.theta` *Call Theta*

Description

Calculate the Theta (Option Greek) of Option Contract

Usage

```
call.theta(s, k, t, sd, r, d = 0)
```

Arguments

`s`	Spot Price of Underlying Asset
`k`	Exercise Price of Contract
`t`	Time to Expiration
`sd`	Volatality
`r`	Risk free rate of return
`d`	Divident Yield (use cont.rate()), Default: 0

Details

Theta represents the rate of change between the option price and time, or time sensitivity - sometimes known as an option's time decay.

Value

Output gives the Theta of a Option Contract.

Examples

```
call.theta(100, 105, 0.25, 0.35, 0.0488)
```

call.vega *Call Vega*

Description

Calculate the Vega (Option Greek) of Option Contract

Usage

```
call.vega(s, k, t, sd, r, d = 0)
```

Arguments

s	Spot Price of Underlying Asset
k	Exercise Price of Contract
t	Time to Expiration
sd	Volatality
r	Risk free rate of return
d	Divident Yield (use cont.rate()), Default: 0

Details

Vega represents the rate of change between an option's value and the underlying asset's implied volatility.

Value

Output gives the Vega of a Option Contract.

Examples

```
call.vega(100, 105, 0.25, 0.35, 0.0488)
```

cont.rate *Continous Rate*

Description

Converts nominal rate into Continously compounded Rate

Usage

```
cont.rate(r, t)
```

Arguments

r	rate (nominal)
t	number of compounding period

Details

Generates Continuously Compounded Rate

Value

Generates Continuously Compounded Rate

Examples

```
cont.rate(0.025, 4)
```

---

iron.condour *Iron Condour Strategy Function*

Description

This function can be used to develop a Iron Condour Strategy.

Usage

```
iron.condour(
  k_long_call,
  k_short_call,
  k_long_put,
  k_short_put,
  c1,
  c2,
  p1,
  p2,
  llimit = 20,
  ulimit = 20
)
```

Arguments

k_long_call	Excercise Price of Long call Option
k_short_call	Excercise Price of Short call Option
k_long_put	Excercise Price of Long Put Option
k_short_put	Excercise Price of Short Put Option
c1	Premium of Long call Option

c2	Premium of Short call Option
p1	Premium of Long Put Option
p2	Premium of Short Put Option
llimit	Lower limit of stock price at Expiration., Default: 20
ulimit	Upper Limit of Stock Price at Expiration, Default: 20

Details

An Iron condor is an options strategy created with four options consisting of two puts (one long and one short) and two calls (one long and one short), and four strike prices, all with the same expiration date.

Value

OUTPUT_DESCRIPTION Returns the profit/loss generated from the strategy along with the profit/loss of individual contract and an interactive graph for the same.

Examples

```
iron.condour(100, 95, 105, 102, 2.3, 1.25, 3.2, 2.3)
```

---

put.delta *Put Delta*

Description

Calculate the Delta (Option Greek) of a Contract

Usage

```
put.delta(s, k, t, sd, r, d = 0)
```

Arguments

s	Spot Price of Underlying Asset
k	Exercise Price of Contract
t	Time to Expiration
sd	Volatality
r	Risk free rate of return
d	Divident Yield (use cont.rate()), Default: 0

Details

Delta represents the rate of change between the option's price and a $1 change in the underlying asset's price.

Value

Oupput gives the delta of a Option Contract.

Examples

```
put.delta(100, 105, 0.25, 0.35, 0.0488)
```

`put.estimate` *Option Greek and Estimated Premium of Put Option*

Description

Calculate the Option Greek of a Contract and Estimated Premium of Contract

Usage

```
put.estimate(s, k, t, sd, r, d = 0)
```

Arguments

`s`	Spot Price of Underlying Asset
`k`	Exercise Price of Contract
`t`	Time to Expiration
`sd`	Volatality
`r`	Risk free rate of return
`d`	Divident Yield (use cont.rate()), Default: 0

Details

"Greeks" is a term used in the options market to describe the different dimensions of risk involved in taking an options position. These Greeks are calculated in this function along with the premium of the option contract using the BSM Model.

Value

Output gives the Option Greek of a Option Contract. Also the Premium of the contract is estimated.

Examples

```
put.estimate(100, 105, 0.25, 0.35, 0.0488)
```

put.gamma *Put Gamma*

Description

Calculate the Gamma (Option Greek) of a Contract

Usage

```
put.gamma(s, k, t, sd, r, d = 0)
```

Arguments

s	Spot Price of Underlying Asset
k	Exercise Price of Contract
t	Time to Expiration
sd	Volatality
r	Risk free rate of return
d	Divident Yield (use cont.rate()), Default: 0

Details

Gamma represents the rate of change between an option's delta and the underlying asset's price.

Value

Output gives the Gamma of a Option Contract.

Examples

```
put.gamma(100, 105, 0.25, 0.35, 0.0488)
```

put.greek *Put Greeks*

Description

Calculate the Specified Option Greek of a Contract

Usage

```
put.greek(
  greek = c("delta", "gamma", "theta", "vega", "rho"),
  s,
  k,
  t,
  sd,
  r,
  d = 0
)
```

Arguments

greek	Character String of the greek to be calculated
s	Spot Price of Underlying Asset
k	Exercise Price of Contract
t	Time to Expiration
sd	Volatality
r	Risk free rate of return
d	Divident Yield (use cont.rate()), Default: 0

Details

Delta represents the rate of change between the option's price and a $1 change in the underlying asset's price. Theta represents the rate of change between the option price and time, or time sensitivity - sometimes known as an option's time decay. Gamma represents the rate of change between an option's delta and the underlying asset's price. Vega represents the rate of change between an option's value and the underlying asset's implied volatility. Rho represents the rate of change between an option's value and a 1% change in the interest rate.

Value

Output gives the Specified Greek of a Option Contract.

Examples

```
put.greek('delta', 100, 105, 0.25, 0.35, 0.0488)
```

`put.minorgreek` *Specified Minor Option Greek*

Description

Calculate the Specified Minor Option Greek of a Contract

Usage

```
put.minorgreek(minorgreek = c("lambda", "vomma"), s, k, t, sd, r, d = 0)
```

Arguments

`minorgreek`	Character String of the minor greek to be calculated
`s`	Spot Price of Underlying Asset
`k`	Exercise Price of Contract
`t`	Time to Expiration
`sd`	Volatality
`r`	Risk free rate of return
`d`	Divident Yield (use cont.rate()), Default: 0

Details

Vomma is the rate at which the vega of an option will react to volatility in the market. In options trading, Lamba is the Greek letter assigned to variable which tells the ratio of how much leverage an option is providing as the price of that option changes.

Value

Output gives the Specified Minor Greek of a Option Contract.

Examples

```
put.minorgreek('lambda', 100, 105, 0.25, 0.35, 0.0488)
```

put.premium.est *Estimated Premium of Put Option*

Description

Calculate the Estimated Premium of Option Contract

Usage

```
put.premium.est(s, k, t, sd, r, d = 0)
```

Arguments

s	Spot Price of Underlying Asset
k	Exercise Price of Contract
t	Time to Expiration
sd	Volatality
r	Risk free rate of return
d	Divident Yield (use cont.rate()), Default: 0

Details

Estimate is calculated based on Black-Scholes Model. The Black Scholes model, also known as the Black-Scholes-Merton (BSM) model, is a mathematical model for pricing an options contract.

Value

Output gives the Estimated Premium of a Option Contract.

Examples

```
put.premium.est(100, 105, 0.25, 0.35, 0.0488)
```

put.rho *Put Rho*

Description

Calculate the Rho (Option Greek) of Option Contract

Usage

```
put.rho(s, k, t, sd, r, d = 0)
```

Arguments

s	Spot Price of Underlying Asset
k	Exercise Price of Contract
t	Time to Expiration
sd	Volatality
r	Risk free rate of return
d	Divident Yield (use cont.rate()), Default: 0

Details

Rho represents the rate of change between an option's value and a 1% change in the interest rate.

Value

Output gives the Estimated Premium of a Option Contract.

Examples

```
put.rho(100, 105, 0.25, 0.35, 0.0488)
```

---

put.spread *Bull/Bear Put Spread Strategy Function*

Description

This function can be used to develop a Bull/Bear Put Strategy.

Usage

```
put.spread(k1, k2, long_put, short_put, llimit = 20, ulimit = 20)
```

Arguments

k1	Excercise Price of Long Put Option
k2	Excercise Price of Short Put Option
long_put	Premium of Long Put Option
short_put	Premium of Short Put Option
llimit	Lower limit of stock price at Expiration., Default: 20
ulimit	Upper Limit of Stock Price at Expiration, Default: 20

Details

The strategy uses two put options to form a range consisting of a high strike price and a low strike price.

Value

OUTPUT_DESCRIPTION Returns the profit/loss generated from the strategy along with the profit/loss of individual contract and an interactive graph for the same.

Examples

```
put.spread(1.2, 3.2, 100, 105)
```

`put.theta` *Put Theta*

Description

Calculate the Theta (Option Greek) of Option Contract

Usage

```
put.theta(s, k, t, sd, r, d = 0)
```

Arguments

`s`	Spot Price of Underlying Asset
`k`	Exercise Price of Contract
`t`	Time to Expiration
`sd`	Volatality
`r`	Risk free rate of return
`d`	Divident Yield (use cont.rate()), Default: 0

Details

Theta represents the rate of change between the option price and time, or time sensitivity - sometimes known as an option's time decay.

Value

Output gives the Theta of a Option Contract.

Examples

```
put.theta(100, 105, 0.25, 0.35, 0.0488)
```

put.vega *Put Vega*

Description

Calculate the Vega (Option Greek) of Option Contract

Usage

```
put.vega(s, k, t, sd, r, d = 0)
```

Arguments

s	Spot Price of Underlying Asset
k	Exercise Price of Contract
t	Time to Expiration
sd	Volatality
r	Risk free rate of return
d	Divident Yield (use cont.rate()), Default: 0

Details

Vega represents the rate of change between an option's value and the underlying asset's implied volatility.

Value

Output gives the Vega of a Option Contract.

Examples

```
put.vega(100, 105, 0.25, 0.35, 0.0488)
```

straddle.long *Long Straddle Strategy Function*

Description

This function can be used to develop a Long Straddle Strategy.

Usage

```
straddle.long(c, p, k, ulimit = 10, llimit = 10)
```

Arguments

c	Premium of Long call Option
p	Premium of Long Put Option
k	Excercise Price of Long call and Put Option
ulimit	Upper Limit of Stock Price at Expiration, Default: 20
llimit	Lower limit of stock price at Expiration., Default: 20

Details

A straddle is a neutral options strategy that involves simultaneously buying both a put option and a call option for the underlying security with the same strike price and the same expiration date.

Value

OUTPUT_DESCRIPTION Returns the profit/loss generated from the strategy along with the profit/loss of individual contract and an interactive graph for the same.

Examples

```
straddle.long(1.2, 3.2, 100)
```

---

straddle.short *Short Straddle Strategy Function*

Description

This function can be used to develop a Short Straddle Strategy.

Usage

```
straddle.short(c, p, k, ulimit = 10, llimit = 10)
```

Arguments

c	Premium of Short call Option
p	Premium of Short Put Option
k	Excercise Price of Short call and Put Option
ulimit	Upper Limit of Stock Price at Expiration, Default: 20
llimit	Lower limit of stock price at Expiration., Default: 20

Details

A straddle is a neutral options strategy that involves simultaneously selling both a put option and a call option for the underlying security with the same strike price and the same expiration date.

Value

OUTPUT_DESCRIPTION Returns the profit/loss generated from the strategy along with the profit/loss of individual contract and an interactive graph for the same.

Examples

```
straddle.short(1.2, 3.2, 100)
```

---

strangle.long *Long Strangle Strategy Function*

Description

This function can be used to develop a Long Strangle Strategy.

Usage

```
strangle.long(c, p, k_call, k_put, ulimit = 10, llimit = 10)
```

Arguments

c	Premium of Long call Option
p	Premium of Long Put Option
k_call	Excercise Price of Long call Option
k_put	Excercise Price of Long Put Option
ulimit	Upper Limit of Stock Price at Expiration, Default: 20
llimit	Lower limit of stock price at Expiration., Default: 20

Details

A strangle is an options strategy where the investor holds a position in both a call and a put option with different strike prices, but with the same expiration date and underlying asset.

Value

OUTPUT_DESCRIPTION Returns the profit/loss generated from the strategy along with the profit/loss of individual contract and an interactive graph for the same.

Examples

```
strangle.long(1.2, 3.2, 100, 105)
```

`strangle.short` *Short Strangle Strategy Function*

Description

This function can be used to develop a Short Strangle Strategy.

Usage

```
strangle.short(c, p, k_call, k_put, ulimit = 10, llimit = 10)
```

Arguments

`c`	Premium of Short call Option
`p`	Premium of Short Put Option
`k_call`	Excercise Price of Short call Option
`k_put`	Excercise Price of Short Put Option
`ulimit`	Upper Limit of Stock Price at Expiration, Default: 20
`llimit`	Lower limit of stock price at Expiration., Default: 20

Details

A strangle is an options strategy where the investor holds a position in both a call and a put option with different strike prices, but with the same expiration date and underlying asset.

Value

OUTPUT_DESCRIPTION Returns the profit/loss generated from the strategy along with the profit/loss of individual contract and an interactive graph for the same.

Examples

```
strangle.short(1.2, 3.2, 100, 105)
```

Index