Package ‘QFRM’

January 20, 2025

Type Package

Title Pricing of Vanilla and Exotic Option Contracts

Version 1.0.1

Date 2015-06-28

Maintainer Oleg Melnikov <XisReal@gmail.com>

Description Option pricing (financial derivatives) techniques mainly following textbook 'Options, Futures and Other Derivatives', 9ed by John C.Hull, 2014. Prentice Hall. Implementations are via binomial tree option model (BOPM), Black-Scholes model, Monte Carlo simulations, etc.
This package is a result of Quantitative Financial Risk Management course (STAT 449 and STAT 649) at Rice University, Houston, TX, USA, taught by Oleg Melnikov, statistics PhD student, as of Spring 2015.

Repository CRAN

License GPL (>= 2)

URL http://Oleg.Rice.edu

NeedsCompilation no

Depends R (>= 2.14.0)

LazyLoad yes

LazyData yes

Imports stats,methods,graphics

Author Oleg Melnikov [aut, cre],
Max Lee [ctb],
Robert Abramov [ctb],
Richard Huang [ctb],
Liu Tong [ctb],
Jake Kornblau [ctb],
Xinnan Lu [ctb],
Kiryl Novikau [ctb],
Tongyue Luo [ctb],
Le You [ctb],
Jin Chen [ctb],
Chengwei Ge [ctb],
Jiayao Huang [ctb],
Kim Raath [ctb]

Date/Publication 2015-07-28 00:48:19

Contents

as.OptPos . . . 3
AsianBS . . . 4
AsianMC . . . 5
AverageStrikeMC . . . 6
BarrierBS . . . 7
BarrierLT . . . 8
BarrierMC . . . 10
BinaryBS . . . 11
BinaryMC . . . 12
Binary_BOPM . . . 13
BOPM . . . 14
BOPM_Eu . . . 15
BS . . . 16
BS_Simple . . . 17
ChooserBS . . . 18
ChooserLT . . . 19
ChooserMC . . . 20
CompoundBS . . . 21
CompoundLT . . . 22
DeferredPaymentLT . . . 24
ForeignEquityBS . . . 25
ForwardStartBS . . . 26
ForwardStartMC . . . 27
GapBS . . . 28
GapLT . . . 29
GapMC . . . 30
HolderExtendibleBS . . . 31
is.Opt . . . 32
is.OptPos . . . 33
is.OptPx . . . 33
LadderMC . . . 34
LookbackBS . . . 35
LookbackMC . . . 36
Opt . . . 38
OptPos . . . 39
OptPx . . . 39
pbnorm . . . 40
PerpetualBS . . . 41
Profit . . . 42
QuotientBS . . . 43
QuotientMC . . . 44
RainbowBS . . . 45
ShoutFD . . . 46
ShoutLT . . . 47

ShoutLTVectorized 48
ShoutMC 49
VarianceSwapBS 50
VarianceSwapMC 51

Index **53**

`as.OptPos` *Coerce an argument to* `OptPos` *class.*

Description

Coerce an argument to `OptPos` class.

Usage

```
as.OptPos(o = Opt(), Pos = c("Long", "Short"), Prem = 0)
```

Arguments

`o`	A `Opt` or `OptPx` object
`Pos`	Specify position direction in your portfolio. `Long` indicates that you own security (it's an asset). `Short` that you shorted (short sold) security (it's a liability).
`Prem`	Option premium, i.e. cost of an option purchased or to be purchased.

Value

An object of class `OptPos`.

Author(s)

Oleg Melnikov

Examples

```
as.OptPos(Opt())
```

`AsianBS` *Asian option valuation via Black-Scholes (BS) model*

Description

Price Asian option using BS model

Usage

```
AsianBS(o = OptPx(Opt(Style = "Asian")))
```

Arguments

`o`	An object of class `OptPx`

Details

This pricing algorithm assumes average price is calculated continuously.

Value

A list of class `AsianBS` consisting of the original `OptPx` object and the option pricing parameters `M1`, `M2`, `F0`, and `sigma` as well as the computed option price `PxBS`.

Author(s)

Xinnan Lu, Department of Statistics, Rice University, Spring 2015

References

Hull, John C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, http://www-2.rotman.utoronto.ca/~hull/ofod/index.html pp.609-611.

Examples

```
(o = AsianBS())$PxBS #Price = ~4.973973,  using default values

 o = Opt(Style='Asian',S0=100,K=90,ttm=3)
 (o = AsianBS(OptPx(o,r=0.03,q=0,vol=0.3)))$PxBS

 o = Opt(Style='Asian',Right='P',S0=100,K=110,ttm=0.5)
 (o = AsianBS(OptPx(o,r=0.03,q=0.01,vol=0.3)))$PxBS

 #See J.C.Hull, OFOD'2014, 9-ed, ex.26.3, pp.610. The price is 5.62.
 o = Opt(Style='Asian',Right='Call',S0=50,K=50,ttm=1)
 (o = AsianBS(OptPx(o,r=0.1,q=0,vol=0.4)))$PxBS
```

`AsianMC` *Asian option valuation with Monte Carlo (MC) simulation.*

Description

Calculates the price of an Asian option using Monte Carlo simulations to determine expected payout.
Assumptions:
The option follows a General Brownian Motion (BM),
$ds = mu * S * dt + sqrt(vol) * S * dW$ where $dW\ N(0, 1)$.
The value of mu (the expected price increase) is `o$r`, the risk free rate of return (RoR).
The averaging period is the life of the option.

Usage

```
AsianMC(o = OptPx(o = Opt(Style = "Asian"), NSteps = 5), NPaths = 5)
```

Arguments

`o`	The `OptPx` Asian option to price.
`NPaths`	The number of simulation paths to use in calculating the price,

Value

The option `o` with the price in the field `PxMC` based on MC simulations.

Author(s)

Jake Kornblau, Department of Statistics and Department of Computer Science, Rice University, 2016

References

Hull, John C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8,
http://www-2.rotman.utoronto.ca/~hull/ofod/index.html
http://www.math.umn.edu/~spirn/5076/Lecture16.pdf

Examples

```
(o = AsianMC())$PxMC #Price = ~5.00,  using default values

  o = OptPx(Opt(Style='Asian'), NSteps = 5)
  (o = AsianMC(o, NPaths=5))$PxMC #Price = ~$5

  (o = AsianMC(NPaths = 5))$PxMC # Price = ~$5

  o = Opt(Style='Asian', Right='Put',S0=10, K=15)
```

```
 o = OptPx(o, r=.05, vol=.1, NSteps = 5)
 (o = AsianMC(o, NPaths = 5))$PxMC # Price = ~$4

 #See J.C.Hull, OFOD'2014, 9-ed, ex.26.3, pp.610.
o = Opt(Style='Asian',S0=50,K=50,ttm=1)
o = OptPx(o,r=0.1,q=0,vol=0.4,NSteps=5)
(o = AsianBS(o))$PxBS   #Price is 5.62.
(o = AsianMC(o))$PxMC
```

AverageStrikeMC *Average Strike option valuation via Monte Carlo (MC) simulation*

Description

Calculates the price of an Average Strike option using Monte Carlo simulations by determining the determine expected payout. Assumes that the input option follows a General Brownian Motion $ds = mu * S * dt + sqrt(vol) * S * dz$ where dz $N(0, 1)$ Note that the value of mu (the expected price increase) is assumped to be o$r, the risk free rate of return. Additionally, the averaging period is assumed to be the life of the option.

Usage

```
AverageStrikeMC(o = OptPx(o = Opt(Style = "AverageStrike")), NPaths = 5)
```

Arguments

o	The AverageStrike OptPx option to price.
NPaths	the number of simulations to use in calculating the price,

Value

The original option object o with the price in the field PxMC based on the MC simulations.

Author(s)

Jake Kornblau, Department of Statistics and Department of Computer Science, Rice University, Spring 2015

References

Hull, John C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, http://www-2.rotman.utoronto.ca/~hull/ofod/index.html Also, http://www.math.umn.edu/~spirn/5076/Lecture16.pdf

Examples

```
(o = AverageStrikeMC())$PxMC   #Price =~ $3.6

  o = OptPx(o = Opt(Style='AverageStrike'), NSteps = 5)
  (o = AverageStrikeMC(o))$PxMC # Price =~ $6

  (o = AverageStrikeMC(NPaths = 20))$PxMC  #Price =~ $3.4

  o = OptPx(o = Opt(Style='AverageStrike'), NSteps = 5)
  (o = AverageStrikeMC(o, NPaths = 20))$PxMC  #Price =~ $5.6
```

`BarrierBS` *Barrier option pricing via Black-Scholes (BS) model*

Description

This function calculates the price of a Barrier option. This price is based on the assumptions that the probability distribution is lognormal and that the asset price is observed continuously.

Usage

```
BarrierBS(o = OptPx(Opt(Style = "Barrier")), dir = c("Up", "Down"),
  knock = c("In", "Out"), H = 40)
```

Arguments

`o`	The `OptPx` option object to price. See `OptBarrier()`, `OptPx()`, and `Opt()` for more information.
`dir`	The direction of the option to price. Either Up or Down.
`knock`	Whether the option goes In or Out when the barrier is reached.
`H`	The barrier level

Details

To price the barrier option, we need to know whether the option is Up or Down | In or Out | Call or Put. Beyond that we also need the S0, K, r, q, vol, H, and ttm arguments from the object classes defined in the package.

Value

The price of the barrier option `o`, which is based on the BSM-adjusted algorithm (see references).

Author(s)

Kiryl Novikau, Department of Statistics, Rice University, Spring 2015

References

Hull, John C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8. http://www-2.rotman.utoronto.ca/~hull/ofod/index.html. pp.606-607

Examples

```
(o = BarrierBS())$PxBS # Option with default arguments is valued at $9.71

 #Down-and-In-Call
 o = Opt(Style='Barrier', S0=50, K=50, ttm=1, Right="Call", ContrSize=10)
 o = OptPx(o,  r = .05, q = 0, vol = .25)
 o = BarrierBS(o, dir = "Down", knock = 'In', H = 40)

 #Down-and-Out Call
 o = Opt(Style='Barrier', S0=50, K=50, ttm=1, Right="Call", ContrSize=10)
 o = OptPx(o, r = .05, q = .02, vol = .25)
 o = BarrierBS(o, dir = "Down", knock = 'Out', H = 40)

 #Up-and-In Call
 o = Opt(Style='Barrier', S0=50, K=50, ttm=1, Right="Call", ContrSize=1)
 o = OptPx(o, r = .05, q = .02, vol = .25)
 o = BarrierBS(o, dir = "Up", knock = 'In', H = 60)

 #Up-and-Out Call
 o = Opt(Style='Barrier', S0 = 50, K = 50, ttm = 1, Right="Call", ContrSize=1)
 o = OptPx(o, r = .05, q = .02, vol = .25)
 o = BarrierBS(o, dir = "Up", knock = 'Out', H = 60)

 #Down-and-In Put
 o = Opt(Style='Barrier', S0=50, K=50, ttm=1, Right="Put", ContrSize=1)
 o = OptPx(o, r = .05, q = .02, vol = .25)
 o = BarrierBS(o, dir = "Down", knock = 'In', H = 40)

 #Down-and-Out Put
 o = Opt(Style='Barrier', S0=50, K=50, ttm=1, Right="Put", ContrSize=1)
 o = OptPx(o, r = .05, q = .02, vol = .25)
 o = BarrierBS(o, dir = "Down", knock = 'Out', H = 40)

 #Up-and-In Put
 o = Opt(Style='Barrier', S0=50, K=50, ttm=1, Right="Put", ContrSize=1)
 o = OptPx(o, r = .05, q = .02, vol = .25)
 o = BarrierBS(o, dir = "Up", knock = 'In', H = 60)

 #Up-and-Out Put
 o = Opt(Style='Barrier', S0=50, K=50, ttm=1, Right="Put", ContrSize=1)
 o = OptPx(o, r = .05, q = .02, vol = .25)
 o = BarrierBS(o, dir = "Up", knock = 'Out', H = 60)
```

BarrierLT *Barrrier option valuation via lattice tree (LT)*

Description

Use Binomial Tree to price barrier options with relatively large NSteps (NSteps > 100) steps. The price may be not as percise as BSM function cause the convergence speed for Binomial Tree is kind of slow.

Usage

```
BarrierLT(o = OptPx(Opt(Style = "Barrier"), vol = 0.25, r = 0.05, q = 0.02,
  NSteps = 5), dir = c("Up", "Down"), knock = c("In", "Out"), H = 60)
```

Arguments

`o`	An object of class `OptPx`
`dir`	A direction for the barrier, either `'Up'` or `'Down'` Default=`'Up'`
`knock`	The option is either a knock-in option or knock-out option. Default=`'In'`
`H`	The barrier level. `H` should less than `S0` if `'Up'`, `H` should greater than `S0` if `'Down'` Default=60.

Value

A list of class `BarrierLT` consisting of the input object `OptPx` and the appended new parameters and option price.

Author(s)

Tong Liu, Department of Statistics, Rice University, Spring 2015

References

Hull, John C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, http://www-2.rotman.utoronto.ca/~hull/ofod/index.html
p.467-468. Trinomial Trees, p.604-606: Barrier Options.

Examples

```
# default Up and Knock-in Call Option with H=60, approximately 7.09
(o = BarrierLT())$PxLT

#Visualization of price changes as Nsteps change.
o = Opt(Style="Barrier")
visual=sapply(10:200,function(n) BarrierLT(OptPx(o,NSteps=n))$PxLT)

c=(10:200)
plot(visual~c,type="l",xlab="NSteps",ylab="Price",main="Price converence with NSteps")

# Down and Knock-out Call Option with H=40
o = OptPx(o=Opt(Style="Barrier"))
BarrierLT(o,dir="Down",knock="Out",H=40)

# Down and Knock-in Call Option with H=40
```

```
o = OptPx(o=Opt(Style="Barrier"))
BarrierLT(o,dir="Down",knock="In",H=40)

# Up and Knock-out Call Option with H=60
o = OptPx(o=Opt(Style="Barrier"))
BarrierLT(o,dir='Up',knock="Out")

# Down and Knock-out Put Option with H=40
o = OptPx(o=Opt(Style="Barrier",Right="Put"))
BarrierLT(o,dir="Down",knock="Out",H=40)

# Down and Knock-in Put Option with H=40
o = OptPx(o=Opt(Style="Barrier",Right="Put"))
BarrierLT(o,dir="Down",knock="In",H=40)

# Up and Knock-out Put Option with H=60
o = OptPx(o=Opt(Style="Barrier",Right="Put"))
BarrierLT(o,dir='Up',knock="Out")

# Up and Knock-in Put Option with H=60
BarrierLT(OptPx(o=Opt(Style="Barrier",Right="Put")))
```

`BarrierMC` *Barrier option valuation via Monte Carlo (MC) simulation.*

Description

Calculates the price of a Barrier Option using 10000 Monte Carlo simulations. The helper function BarrierCal() aims to calculate expected payout for each stock prices.

Important Assumptions: The option follows a General Brownian Motion (GBM) $ds = mu * S * dt + sqrt(vol) * S * dW$ where $dW\ N(0, 1)$. The value of mu (the expected percent price increase) is assumed to be `o$r-o$q`.

Usage

```
BarrierMC(o = OptPx(o = Opt(Style = "Barrier")), knock = c("In", "Out"),
  B = 60, NPaths = 5)
```

Arguments

`o`	The `OptPx` Barrier option to price.
`knock`	Defines the Barrier option to be "`In`" or "`Out`"
`B`	The Barrier price level
`NPaths`	The number of simulation paths to use in calculating the price

Value

The option `o` with the price in the field `PxMC` based on MC simulations and the Barrier option properties set by the users themselves

Author(s)

Huang Jiayao, Risk Management and Business Intelligence at Hong Kong University of Science and Technology, Exchange student at Rice University, Spring 2015

References

Hull, John C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, `http://www-2.rotman.utoronto.ca/~hull/ofod/index.html`. Also, `http://stackoverflow.com/questions/25946852/r-monte-carlo-simulation-price-path-converging-volatility-iss`

Examples

```
(o = BarrierMC())$PxMC #Price =~ $11

 o = OptPx(o=Opt(Style='Barrier'),NSteps = 10)
 (o = BarrierMC(o))$PxMC #Price =~ $14.1

 (o = BarrierMC(NPaths = 5))$PxMC # Price =~ $11

 (o = BarrierMC(B=65))$PxMC # Price =~ $10

 (o = BarrierMC(knock="Out"))$PxMC #Price =~ $1
```

`BinaryBS` *Binary option valuation with Black-Scholes (BS) model*

Description

S3 object pricing model for a binary option. Two types of binary options are priced: `'cash-or-nothing'` and `'asset-or-nothing'`.

Usage

```
BinaryBS(o = OptPx(Opt(Style = "Binary")), Q = 1,
  Type = c("cash-or-nothing", "asset-or-nothing"))
```

Arguments

`o`	An object of class `OptPx`
`Q`	A fixed amount of payoff
`Type`	Binary option type: 'Cash or Nothing' or 'Asset or Nothing'. Partial names are allowed, eg. `'C'` or `'A'`

Value

A list of class `Binary.BS` consisting of the input object `OptPx` and the appended new parameters and option price.

Author(s)

Xinnan Lu, Department of Statistics, Rice University, Spring 2015

References

Hull, John C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, http://www-2.rotman.utoronto.ca/~hull/ofod/index.html. pp.606-607

Examples

```
(o = BinaryBS())$PxBS

#This example should produce price 4.33 (see Derivagem, DG201.xls)
o = Opt(Style="Binary", Right='Call', S0=50, ttm=5/12, K=52)
o = OptPx(o, r=.1, vol=.40, NSteps=NA)
(o = BinaryBS(o, Q = 10, Type='cash-or-nothing'))$PxBS

BinaryBS(OptPx(Opt(Style="Binary"), q=.01), Type='asset-or-nothing')
BinaryBS(OptPx(Opt(Style="Binary", S0=100, K=80),q=.01))
o = Opt(Style="Binary", Right="Put", S0=50, K=60)
BinaryBS(OptPx(o,q=.04), Type='asset-or-nothing')
```

BinaryMC *Binary option valuation via Monte-Carlo (via) simulation.*

Description

Binary option valuation via Monte-Carlo (via) simulation.

Usage

```
BinaryMC(o = OptPx(Opt(Style = "Binary")), Q = 25,
  Type = c("cash-or-nothing", "asset-or-nothing"), NPaths = 5)
```

Arguments

o	An OptPx object
Q	A fixed numeric amount of payoff
Type	Binary option type: 'cash-or-nothing' or 'asset-or-nothing'.
NPaths	The number of simulation paths to use in calculating the price Partial names are allowed, eg. 'c' or 'a'

Details

Two types of binary options are priced: 'cash-or-nothing' and 'asset-or-nothing'.

Value

The original input object o with added parameters and option price PxMC

Author(s)

Tongyue Luo, Rice University, Spring 2015.

References

Hull, John C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, http://www-2.rotman.utoronto.ca/~hull/ofod/index.html. pp.606-607.

Examples

```
(o = BinaryMC())$PxMC

o = OptPx(Opt(Style="Binary"))
(o = BinaryMC(o, Type="cash"))$PxMC

o = OptPx(Opt(Style="Binary"),q=0.01)
(o = BinaryMC(o, Type="asset"))$PxMC

o = OptPx(Opt(Style="Binary", S0=100, K=80),q=0.01)
(o = BinaryMC(o, Type="cash"))$PxMC

o = OptPx(Opt(Style="Binary", Right="Put", S0=50, K=60),q=0.04)
(o = BinaryMC(o, Type="asset"))$PxMC
```

Binary_BOPM *Binary option valuation vialattice tree (LT) implementation*

Description

Compute option price via binomial option pricing model (recombining symmetric binomial tree)

Usage

```
Binary_BOPM(o = OptPx(Opt(Style = "Binary")), Type = c("cash-or-nothing",
  "asset-or-nothing"), Q = 1000, IncBT = FALSE)
```

Arguments

o	OptPx object
Type	Binary option type: 'cash-or-nothing' or 'asset-or-nothing'
Q	A fixed amount of payoff
IncBT	TRUE/FALSE, indicates whether to include the full binomial tree in the returned object

Value

original `OptPx` object with `Px.BOPM` property and (optional) binomial tree IncBT = FALSE: option price value (type double, class numeric) IncBT = TRUE: binomial tree as a list (of length (o$n+1) of numeric matrices (2 x i). Each matrix is a set of possible i outcomes at time step i columns: (underlying prices, option prices)

Examples

```
(o = Binary_BOPM())$PxBT

o = OptPx(o=Opt(Style='Binary'))
(o = Binary_BOPM(o, Type='cash', Q=100, IncBT=TRUE))$PxBT

o = OptPx(Opt(Style='Binary'), r=0.05, q=0.02, rf=0.0, vol=0.30, NSteps=5)
(o = Binary_BOPM(o, Type='cash', Q=1000, IncBT=FALSE))$PxBT

o = OptPx(o=Opt(Style='Binary'), r=0.15, q=0.01, rf=0.05, vol=0.35, NSteps=5)
(o = Binary_BOPM(o,Type='asset',Q=150, IncBT=FALSE))$PxBT

o = OptPx(o=Opt(Style='Binary'), r=0.025, q=0.001, rf=0.0, vol=0.10, NSteps=5)
(o = Binary_BOPM(o, Type='cash', Q=20, IncBT=FALSE))$PxBT
```

`BOPM` *Binomial option pricing model*

Description

Compute option price via binomial option pricing model (recombining symmetric binomial tree). If no tree requested for European option, vectorized algorithm is used.

Usage

```
BOPM(o = OptPx(), IncBT = TRUE)
```

Arguments

`o`	An `OptPx` object
`IncBT`	Values `TRUE` or `FALSE` indicating whether to include a list of all option tree values (underlying and derivative prices) in the returned `OptPx` object.

Value

An original `OptPx` object with `PxBT` field as the binomial-tree-based price of an option and (an optional) the fullly-generated binomial tree in `BT` field.

- `IncBT` = `FALSE`: option price value (type `double`, class `numeric`)
- `IncBT` = `TRUE`: binomial tree as a list (of length (`o$NSteps+1`) of numeric matrices (2 x `i`)

Each matrix is a set of possible i outcomes at time step i columns: (underlying prices, option prices)

Author(s)

Oleg Melnikov, Department of Statistics, Rice University, Spring 2015

References

Hull, J.C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, `http://www-2.rotman.utoronto.ca/~hull/ofod`. `http://amzn.com/0133456315`

#See Fig.13.11, Hull/9e/p291. #Create an option and price it o = Opt(Style='Eu', Right='C', S0 = 808, ttm = .5, K = 800) o = BOPM(OptPx(o, r=0.05, q=0.02, vol=0.2, NSteps=2), IncBT=TRUE) o$PxBT #print added calculated price to PxBT field

#Fig.13.11, Hull/9e/p291: o = Opt(Style='Eu', Right='C', S0=810, ttm=.5, K=800) BOPM(OptPx(o, r=0.05, q=0.02, vol=0.2, NSteps=2), IncBT=TRUE)$PxBT

#DerivaGem diplays up to 10 steps: o = Opt(Style='Am', Right='C', 810, .5, 800) BOPM(OptPx(o, r=0.05, q=0.02, vol=0.2, NSteps=20), IncBT=TRUE)

#DerivaGem computes up to 500 steps: o = Opt(Style='American', Right='Put', 810, 0.5, 800) BOPM(OptPx(o, r=0.05, q=0.02, vol=0.2, NSteps=1000), IncBT=FALSE)

See Also

`BOPM_Eu` for European option via vectorized approach.

`BOPM_Eu` *European option valuation (vectorized computation).*

Description

A helper function to price European options via a vectorized (fast, memory efficient) approach.

Usage

```
BOPM_Eu(o = OptPx())
```

Arguments

`o`	An `OptPx` object

Value

A list of class `OptPx` with an element `PxBT`, which is an option price value (type `double`, class `numeric`)

Author(s)

Oleg Melnikov, Department of Statistics, Rice University, Spring 2015 Code adopted Gilli & Schumann's R implementation to `Opt*` objects

References

Gili, M. and Schumann, E. (2009) *Implementing Binomial Trees*, COMISEF Working Papers Series

See Also

http://papers.ssrn.com/sol3/papers.cfm?abstract_id=1341181 for original paper, BOPM for American option pricing.

Examples

```
#Fig.13.11, Hull/9e/p291:
o = Opt(Style='European', Right='Call', S0=810, ttm=.5, K=800)
(o <- BOPM_Eu( OptPx(o, r=.05, q=.02, vol=.2, NSteps=2)))$PxBT

o = Opt('Eu', 'C', 0.61, .5, 0.6, SName='USD/AUD')
o = OptPx(o, r=.05, q=.02, vol=.12, NSteps=2)
(o <- BOPM_Eu(o))$PxBT
```

BS *Black-Scholes (BS) pricing model*

Description

a wrapper function for BS_Simple; uses `OptPx` object as input.

Usage

```
BS(o = OptPx())
```

Arguments

o	An `OptPx` object

Value

An original `OptPx` object with `BS` list as components of Black-Scholes formular. See `BS_Simple`.

Author(s)

Oleg Melnikov, Department of Statistics, Rice University, Spring 2015

References

Hull, J.C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, http://www-2.rotman.utoronto.ca/~hull/ofod. http://amzn.com/0133456315

Examples

```
#See Hull, p.338, Ex.15.6. #Create an option and price it
o = Opt(Style='Eu', Right='Call', S0 = 42, ttm = .5, K = 40)
o = BS( OptPx(o, r=.1, vol=.2, NSteps=NA))
o$PxBS #print call option price computed by Black-Scholes pricing model
o$BS$Px$Put #print put option price computed by Black-Scholes pricing model
```

`BS_Simple` *Black-Scholes formula*

Description

Black-Scholes (aka Black-Scholes-Merton, BS, BSM) formula for simple parameters

Usage

```
BS_Simple(S0 = 42, K = 40, r = 0.1, q = 0, ttm = 0.5, vol = 0.2)
```

Arguments

`S0`	The spot price of the underlying security
`K`	The srike price of the underlying (same currency as S0)
`r`	The annualized risk free interest rate, as annual percent / 100 (i.e. fractional form. 0.1 is 10 percent per annum)
`q`	The annualized divididend yield, same units as `r`
`ttm`	The time to maturity, fraction of a year (annualized)
`vol`	The volatility, in units of standard deviation.

Details

Uses BS formula to calculate call/put option values and elements of BS model

Value

a list of BS formula elements and BS price, such as `d1` for d_1, `d2` for d_2, `Nd1` for $N(d_1)$, `Nd2` for $N(d_2)$, `NCallPxBS` for BSM call price, `PutPxBS` for BSM put price

Author(s)

Robert Abramov, Department of Statistics, Rice University, Spring 2015

References

Hull, J.C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, `http://www-2.rotman.utoronto.ca/~hull/ofod`. `http://amzn.com/0133456315` `http://www.theresearchkitchen.com/archives/106`

Examples

```
#See Hull p.339, Ex.15.6.
(o <- BS_Simple(S0=42,K=40,r=.1,q=0,ttm=.5,vol=.2))$Px$Call #returns 4.759422
o$Px$Put # returns 0.8085994 as the price of the put

BS_Simple(100,90,0.05,0,2,0.30)
BS_Simple(50,60,0.1,.2,3,0.25)
BS_Simple(90,90,0.15,0,.5,0.20)
BS_Simple(15,15,.01,0.0,0.5,.5)
```

`ChooserBS` *Chooser option valuation via Black-Scholes (BS) model*

Description

Compute an exotic option that allow the holder decide the option will be a call or put option at some predetermined future date. In a simple case, both put and call option are plain vanilla option. The value of the simple chooser option is $\max C(S, K, t_1), P(S, K, t_2)$. The plain vanilla option is calculated based on the BS model.

Usage

```
ChooserBS(o = OptPx(Opt(Style = "Chooser")), t1 = 9/12, t2 = 3/12)
```

Arguments

`o`	An object of class `OptPx`
`t1`	The time to maturity of the call option, measured in years.
`t2`	The time to maturity of the put option, measured in years.

Value

A list of class `SimpleChooserBS` consisting of the original `OptPx` object and the option pricing parameters `t1`, `t2`, as well as the computed price `PxBS`.

Author(s)

Le You, Department of Statistics, Rice University, spring 2015

References

- Hull, John C.,*Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8. `http://www-2.rotman.utoronto.ca/~hull/ofod/index.html`
- Huang Espen G., *Option Pricing Formulas*, 2ed. `http://down.cenet.org.cn/upfile/10/20083212958160.pdf`
- Wee, Lim Tiong, MFE5010 *Exotic Options,Notes for Lecture 4 Chooser option*. `http://www.stat.nus.edu.sg/~stalimtw/MFE5010/PDF/L4chooser.pdf`
- Humphreys, Natalia A., ACTS 4302 Principles of Actuarial Models: Financial Economics. *Lesson 14: All-or-nothing, Gap, Exchange and Chooser Options*.

Examples

```
(o = ChooserBS())$PxBS

o = Opt(Style='Chooser',Right='Other',S0=50, K=50)
(o = ChooserBS(OptPx(o, r=0.06, q=0.02, vol=0.2),9/12, 3/12))$PxBS

o = Opt(Style='Chooser',Right='Other',S0=50, K=50)
(o = ChooserBS (OptPx(o,r=0.08, q=0, vol=0.25),1/2, 1/4))$PxBS

o = Opt(Style='Chooser',Right='Other',S0=100, K=50)
(o = ChooserBS(OptPx(o,r=0.08, q=0.05, vol=0.3),1/2, 1/4))$PxBS
```

ChooserLT *Chooser option valuation via Lattice Tree (LT) Model*

Description

Calculates the price of a Chooser option using a recombining binomial tree model. Has pricing capabilities for both simple European Chooser options as well as American Chooser Options, where exercise can occur any time as a call or put options.

Usage

```
ChooserLT(o = OptPx(Opt("Chooser", ttm = 1)), t1 = 0.5, t2 = 0.5,
  IncBT = FALSE)
```

Arguments

o	The `OptPx` option object to price.
t1	The time to maturity of the call option, measured in years.
t2	The time to maturity of the put option, measured in years.
IncBT	`TRUE/FALSE` Choice of including the lattice tree simulation in the output. Input `FALSE` yields faster computation and fewer calculated results to store in memory.

Details

The American chooser option is interpreted as exercise of option being available at any point in time during the life of the option.

Value

An original `OptPx` object with `PxLT` field as the price of the option and user-supplied `ttc`, `IncBT` parameters attached.

Author(s)

Richard Huang, Department of Statistics, Rice University, spring 2015

References

Hull, J.C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, http://www-2.rotman.utoronto.ca/~hull/ofod/index.html
Thomas S.Y. Ho et al., *The Oxford Guide to Financial Modeling : Applications for Capital Markets*. . .

Examples

```
(o = ChooserLT())$PxLT    #Default Chooser option price. (See Ho pg 234 in references)

o = Opt('Eu', S0=100, ttm=1, K=100)
o = OptPx(o, r=0.10, q=0, vol=0.1, NSteps=5)
(o = ChooserLT(o, t1 = .5, t2 =.5, IncBT=TRUE))$PxLT

#American Chooser, higher price than European equivalent
o = Opt('Am', S0=100, ttm=1, K=100)
o = OptPx(o, r=0.10, q=0, vol=0.1, NSteps=5)
ChooserLT(o,t1=.5, t2=.5,IncBT=FALSE)$PxLT

o = Opt('Eu', S0=50, ttm=1, K=50)
o = OptPx(o, r=0.05, q=0.02, vol=0.25, NSteps=5)
ChooserLT(o, t1 = .75, t2 = .75, IncBT=FALSE)$PxLT

o = Opt('Eu', S0=50, ttm=1, K=50)
o = OptPx(o, r=0.05, q=0.5, vol=0.25, NSteps=5)
ChooserLT(o, t1 = .75, t2 = .75, IncBT=FALSE)$PxLT
```

ChooserMC — *Chooser option valuation via Monte Carlo (MC) simulations*

Description

Price chooser option using Monte Carlo (MC) simulation.

Usage

```
ChooserMC(o = OptPx(Opt(Style = "Chooser")), isEu = TRUE, T1 = 1,
  NPaths = 5, plot = FALSE)
```

Arguments

`o`	An object of class `OptPx`
`isEu`	Values `TRUE` or `FALSE` indicating if the chooser is an European or American style option
`T1`	The time when the choice is made whether the option is a call or put
`NPaths`	The number of Monte Carol simulation paths
`plot`	Values `TRUE` or `FALSE` indicating whether to include a comparison plot of option price versus number of paths

Details

A chooser option (sometimes referred to as an as you like it option) has the feature that, after a specified period of time, the holder can choose whether the option is a call or a put. In this algorithm, we can price chooser options when the underlying options are both European or are both American. When the underlying is an American option, the option holder can exercise before and after T1.

Value

A list of class `ChooserMC` consisting of original `OptPx` object, option pricing parameters `isEu`, `NPaths`, and `T1`, as well as the computed price `PxMC` for the chooser option.

Author(s)

Xinnan Lu, Department of Statistics, Rice University, Spring 2015

References

Hull, John C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, `http://www-2.rotman.utoronto.ca/~hull/ofod/index.html`. p.603.

Examples

```
(o = ChooserMC())$PxMC

o = OptPx(Opt(Right='Call',Style="Chooser"))
 ChooserMC(o,isEu=TRUE,NPaths=5, plot=TRUE)

 o = OptPx(Opt(Right='Put',Style="Chooser"))
 ChooserMC(o,isEu=TRUE,NPaths=5, plot=TRUE)

 o = Opt(Right='C',S0=100,K=110,ttm=4,Style="Chooser")
 o = OptPx(o,vol=0.2,r=0.05,q=0.04)
 ChooserMC(o,isEu=TRUE,T1=2,NPaths=5)

 o = Opt(Right='P',S0=110,K=100,ttm=4,Style="Chooser")
 o = OptPx(o,vol=0.2,r=0.05,q=0.04)
 ChooserMC(o,isEu=TRUE,T1=2,NPaths=5)

 o = Opt(Right='C',S0=50,K=50,ttm=0.5,Style="Ch")
 o = OptPx(o,vol=0.25,r=0.08,q=0.1)
 ChooserMC(o,isEu=FALSE,T1=0.25,NPaths=5)
```

`CompoundBS` *Compound option valuation with Black-Scholes (BS) model*

Description

Compound option valuation with Black-Scholes (BS) model

Usage

```
CompoundBS(o = OptPx(Opt(Style = "Compound")), K1 = 10, T1 = 0.5,
  Type = c("cc", "cp", "pp", "pc"))
```

Arguments

o	= OptPx object
K1	The first Strike Price (of the option on the option)
T1	The time of first expiry (of the option on the option)
Type	Possible choices are cc - call option on call option cp - call on put pc - put on call pp - put on put

Value

A list of object 'OptCompound' containing the option parameters binomial tree parameters and compound option parameters

Author(s)

Robert Abramov

Examples

```
(o <- CompoundBS())$PxBS #price compound option with default parameters

o = OptPx(Opt(Style='Compound'), r=0.05, q=0.0, vol=0.25)
CompoundBS(o,K1=10,T1=0.5)

o = Opt(Style='Compound', S0=50, K=52, ttm=1)
CompoundBS(o=OptPx(o, r=.05, q=0, vol=.25),K1=6,T1=1.5)

o = Opt(Style='Compound', S0=90, K=100, ttm=1.5)
CompoundBS(o=OptPx(o, r=.05, q=0, vol=.25),K1=15,T1=1)

o = Opt(Style='Compound', S0=15, K=15, ttm=0.25)
CompoundBS(o=OptPx(o, r=.05, q=0, vol=.25),K1=3,T1=1.5)
```

CompoundLT *Compound option valuation via lattice tree (LT) model*

Description

CompoundLT prices a compound option using the binomial tree (BT) method. The inputs it takes are two OptPx objects. It pulls the S from the o2 input which should be the option with the greater time to maturity.

Usage

```
CompoundLT(o1 = OptPx(Opt(Style = "Compound")), o2 = OptPx(Opt(Style =
  "Compound")))
```

Arguments

`o1`	The `OptPx` object with the shorter time to maturity
`o2`	The `OptPx` object with the longer time to maturity

Value

User-supplied `o1` option with fields `o2` and `PxLT`, as the second option and calculated price, respectively.

Author(s)

Kiryl Novikau, Department of Statistics, Rice University, Spring 2015

References

Hull, John C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, http://www-2.rotman.utoronto.ca/~hull/ofod/index.html.

Examples

```
(o = CompoundLT())$PxLT # Uses default arguments

#Put option on a Call:
o = Opt(Style="Compound", S0=50, ttm=.5, Right="P", K = 50)
o1 = OptPx(o, r = .1, vol = .4, NSteps = 5)
o = Opt(Style="Compound", S0=50, ttm=.75, Right="C", K = 60)
o2 = OptPx(o, r = .1, vol = .4, NSteps = 5)
(o = CompoundLT(o1, o2))$PxLT

#Call option on a Call:
o = Opt(Style = "Compound", S0 = 50, ttm= .5, Right = "Call", K = 50)
o1 = OptPx(o, r = .1, vol = .4, NSteps = 5)
o = Opt(Style = "Compound", S0 = 50, ttm= .75, Right = "Call", K = 5)
o2 = OptPx(o, r = .1, vol = .4, NSteps = 5)
(o = CompoundLT(o1, o2))$PxLT

#Put option on a Put:
o = Opt(Style = "Compound", S0 = 50, ttm= .5, Right = "Put", K = 40)
o1 = OptPx(o, r = .1, vol = .4, NSteps = 5)
o = Opt(Style = "Compound", S0 = 50, ttm= .75, Right = "Put", K = 50)
o2 = OptPx(o, r = .1, vol = .4, NSteps = 5)
(o = CompoundLT(o1, o2))$PxMC

#Call option on a Put:
o = Opt(Style = "Compound", S0 = 50, ttm= .5, Right = "Call", K = 30)
o1 = OptPx(o, r = .1, vol = .4, NSteps = 5)
```

```
o = Opt(Style = "Compound", S0 = 50, ttm= .75, Right = "Put", K = 80)
o2 = OptPx(o, r = .1, vol = .4, NSteps = 5)
(o = CompoundLT(o1, o2))$PxLT
```

DeferredPaymentLT *DeferredPaymentLT*

Description

A binomial tree pricer of a Deferred Payment option. An American option that has payment at expiry no matter when exercise, causing differences in present value (PV) of a payoff.

Usage

```
DeferredPaymentLT(o = OptPx(Opt(Style = "DeferredPayment")))
```

Arguments

o	An object of class `OptPx`

Value

An object of class `OptPx` with price included

Author(s)

Max Lee, Department of Statistics, Rice University, Spring 2015

References

Hull, J.C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, http://www-2.rotman.utoronto.ca/~hull/ofod/index.html

Examples

```
(o = DeferredPaymentLT())$PxLT

o = Opt(Style='DeferredPayment', Right="Call", S0=110,ttm=.5,K=110)
(o = DeferredPaymentLT(OptPx(o,r=.05,q=.04,vol=.2,NSteps=5)))$PxLT

o = Opt(Style='DeferredPayment', Right="Put", S0 = 50, ttm=2,K=47)
(o = DeferredPaymentLT(OptPx(o,r=.05,q=.04,vol=.25,NSteps=3)))$PxLT
```

ForeignEquityBS — *ForeignEquity option valuation via Black-Scholes (BS) model*

Description

ForeignEquity Option via Black-Scholes (BS) model

Usage

```
ForeignEquityBS(o = OptPx(Opt(Style = "ForeignEquity")), I1 = 1540,
  I2 = 1/90, sigma1 = 0.14, sigma2 = 0.18, g1 = 0.02, rho = -0.3,
  Type = c("Foreign", "Domestic"))
```

Arguments

`o`	An object of class `OptPx`
`I1`	A spot price of the underlying security 1 (usually I1)
`I2`	A spot price of the underlying security 2 (usually I2)
`sigma1`	a vector of implied volatilities for the associated security 1
`sigma2`	a vector of implied volatilities for the associated security 2
`g1`	is the payout rate of the first stock
`rho`	is the correlation between asset 1 and asset 2
`Type`	ForeignEquity option type: 'Foreign' or 'Domestic'

Details

Two types of ForeignEquity options are priced: `'Foreign'` and `'Domestic'`. See "Exotic Options", 2nd, Peter G. Zhang for more details.

Value

A list of class `ForeignEquityBS` consisting of the original `OptPx` object and the option pricing parameters `I1`,`I2`, `Type`, `isForeign`, and `isDomestic` as well as the computed price `PxBS`.

Author(s)

Chengwei Ge, Department of Statistics, Rice University, 2015

References

Zhang, Peter G. *Exotic Options*, 2nd, 1998.

Examples

```
o = OptPx(Opt(Style = 'ForeignEquity', Right = "Put"), r= 0.03)
ForeignEquityBS(o, I1=1540, I2=1/90, g1=.02, sigma1=.14,sigma2=0.18, rho=.03,Type='Foreign')

o = OptPx(Opt(Style = 'ForeignEquity',  Right = "Put", ttm=9/12, K=1600), r=.03)
ForeignEquityBS(o, I1=1540, I2=1/90, g1=.02, sigma1=.14,sigma2=0.18, rho=0.03,Type='Foreign')

o = OptPx(Opt(Style = 'ForeignEquity', Right = "C", ttm=9/12, K=1600), r=.03)
ForeignEquityBS(o, I1=1540, I2=1/90, g1=.02, sigma1=.14,sigma2=0.18, rho=0.03,Type='Foreign')

o = OptPx(Opt(Style = 'ForeignEquity', Right = "C", ttm=9/12, K=1600), r=.03)
ForeignEquityBS(o, I1=1540, I2=1/90, g1=.02, sigma1=.14,sigma2=0.18, rho=0.03,Type='Domestic')

o = OptPx(Opt(Style = 'ForeignEquity', Right = "P", ttm=9/12, K=1600), r=.03)
ForeignEquityBS(o, I1=1540, I2=1/90, g1=.02, sigma1=.14,sigma2=0.18, rho=0.03,Type='Domestic')
```

`ForwardStartBS` *ForwardStart option valuation via Black-Scholes (BS) model*

Description

Compute the price of Forward Start options using BSM. A forward start option is a standard European option whose strike price is set equal to current asset price at some prespecified future date. Employee incentive options are basically forward start option

Usage

```
ForwardStartBS(o = OptPx(Opt(Style = "ForwardStart")), tts = 0.1)
```

Arguments

`o`	an `OptPx` object including basic information of an option
`tts`	Time to start of the option (in years)

Details

A standard European option starts at a future time tts.

Value

The original user-supplied `OptPX` object with price field `PxBS` and any other provided user-supplied parameters.

Author(s)

Tongyue Luo, Department of Statistics, Rice University, Spring 2015

References

Hull, John C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8.http://www-2.rotman.utoronto.ca/~hull/ofod/index.html. p.602

Examples

```
(o = ForwardStartBS())$PxBS

o = OptPx(Opt(Style='ForwardStart', Right='Put'))
(o = ForwardStartBS(o))$PxBS
```

`ForwardStartMC` *Forward Start option valuation via Monte-Carlo (MC) simulation*

Description

S3 object pricing model for a forward start European option using Monte Carlo simulation

Usage

```
ForwardStartMC(o = OptPx(Opt(Style = "ForwardStart")), tts = 0.1,
  NPaths = 5)
```

Arguments

`o`	An object of class `OptPx`
`tts`	Time to start of the option, in years.
`NPaths`	The number of MC simulation paths.

Details

A standard European option starts at a future time tts.

Value

A list of class `ForwardStartMC` consisting of the input object `OptPx` and the appended new parameters and option price.

Author(s)

Tongyue Luo, Rice University, Spring 2015.

References

Hull, John C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, http://www-2.rotman.utoronto.ca/~hull/ofod/index.html.
http://investexcel.net/forward-start-options/

Examples

```
(o = ForwardStartMC())$PxMC

o = OptPx(Opt(Style='ForwardStart'), q = 0.03, r = 0.1, vol = 0.15)
(o = ForwardStartMC(o, tts=0.25))$PxMC

ForwardStartMC(o = OptPx(Opt(Style='ForwardStart', Right='Put')))$PxMC
```

GapBS *Gap option valuation via Black-Scholes (BS) model*

Description

S3 object constructor for price of gap option using BS model

Usage

```
GapBS(o = OptPx(Opt(Style = "Gap", Right = "Put", S0 = 5e+05, K = 4e+05, ttm =
  1, ContrSize = 1, SName =
  "Insurance coverage example #26.1, p.601, OFOD, J.C.Hull, 9ed."), r = 0.05, q
  = 0, vol = 0.2), K2 = 350000)
```

Arguments

`o`	An object of class `OptPx`
`K2`	Strike price that determine if the option pays off.

Value

An original `OptPx` object with `PxBS` field as the price of the option and user-supplied `K2` parameter

Author(s)

Tong Liu, Department of Statistics, Rice University, Spring 2015

References

Hull, John C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8. http://www.mathworks.com/help/fininst/gapbybls.html

Examples

```
#See J.C.Hull, OFOD'2014, 9-ed, Example 26.1, p.601
(o <- GapBS())$PxBS

GapBS(o=OptPx(Opt(Style='Gap',Right='Put',K=57)))

#See http://www.mathworks.com/help/fininst/gapbybls.html
```

```
o = Opt(Style='Gap',Right='Put',K=57,ttm=0.5,S0=52)
o = GapBS(OptPx(o,vol=0.2,r=0.09),K2=50)

o = Opt(Style='Gap',Right='Put',K=57,ttm=0.5,S0=50)
(o <- GapBS(OptPx(o,vol=0.2,r=0.09),K2=50))$PxBS
```

`GapLT` *Gap option valuation via lattice tree (LT) model*

Description

A binomial tree pricer of Gap options that takes the average results for given step sizes in NSteps. Large step sizes should be used for optimal accuracy but may take a minute or so.

Usage

```
GapLT(o = OptPx(Opt(Style = "Gap")), K2 = 60, on = c(100, 200))
```

Arguments

`o`	An object of class `OptPx`
`K2`	A numeric strike price above used in calculating if option is in the money or not, known as trigger.
`on`	A vector of number of steps to be used in binomial tree averaging, vector of positive intergers.

Value

An onject of class `OptPx` including price

Author(s)

Max Lee, Department of Statistics, Rice University, Spring 2015

References

Hull, John C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8. http://www-2.rotman.utoronto.ca/~hull/ofod/index.html.
Humphreys, Natalia. University of Dallas.

Examples

```
(o = GapLT())$PxLT

o = Opt(Style="Gap",Right='Put',S0 = 500000, ttm = 1,K = 400000)
o = OptPx(o,r = .05, q=0, vol =.2)
(o = GapLT(o,K2 = 350000,on=c(498,499,500,501,502)))$PxLT

o = Opt(Style="Gap", Right='Call',S0 = 65, ttm = 1,K = 70)
o = OptPx(o,r = .05, q=.02,vol =.1)
```

GapMC *Gap option valuation via Monte Carlo (MC) simulation*

Description

GapMC prices a gap option using the MC method. The call payoff is $S_T - K$ when $S_T > K2$, where K_2 is the trigger strike. The payoff is increased by $K_2 - K$, which can be positive or negative. The put payoff is $K - S_T$ when $S_T < K_2$. Default values are from policyholder-insurance example 26.1, p.601, from referenced OFOD, 9ed, text.

Usage

```
GapMC(o = OptPx(Opt(Style = "Gap", Right = "Put", S0 = 5e+05, K = 4e+05, ttm =
  1, ContrSize = 1, SName =
  "Insurance coverage example #26.1, p.601, OFOD, J.C.Hull, 9ed."), r = 0.05, q
  = 0, vol = 0.2), K2 = 350000, NPaths = 5)
```

Arguments

o	The OptPx object (See OptPx() constructor for more information)
K2	The trigger strike price.
NPaths	The number of paths (trials) to simulate.

Value

An OptPx object. The price is stored under o$PxMC.

Author(s)

Kiryl Novikau, Department of Statistics, Rice University, Spring 2015

References

Hull, John C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8. http://www-2.rotman.utoronto.ca/~hull/ofod/index.html. p.601

Examples

```
(o = GapMC())$PxMC  #example 26.1, p.601

o = Opt(Style='Gap', Right='Call', S0=50, K=40, ttm=1)
o = OptPx(o, vol=.2, r=.05, q = .02)
(o = GapMC(o, K2 = 45, NPaths = 5))$PxMC

o = Opt(Style='Gap', Right='Call', S0 = 50, K = 60, ttm = 1)
o = OptPx(o, vol=.25,r=.15, q = .02)
(o = GapMC(o, K2 = 55, NPaths = 5))$PxMC
```

```
o = Opt(Style='Gap', Right = 'Put', S0 = 50, K = 57, ttm = .5)
o = OptPx(o, vol = .2, r = .09, q = .2)
(o = GapMC(o, K2 = 50, NPaths = 5))$PxMC

o = Opt(Style='Gap', Right='Call', S0=500000, K=400000, ttm=1)
o = OptPx(o, vol=.2,r=.05, q = 0)
(o = GapMC(o, K2 = 350000, NPaths = 5))$PxMC
```

HolderExtendibleBS   *Holder Extendible option valuation via Black-Scholes (BS) model*

Description

Computes the price of exotic option (via BS model) which gives the holder the right to extend the option's maturity at an additional premium.

Usage

```
HolderExtendibleBS(o = OptPx(Opt(Style = "HolderExtendible")), k = 105,
  t1 = 0.5, t2 = 0.75, A = 1)
```

Arguments

`o`	An object of class `OptPx`
`k`	The exercise price of the option at t2, a numeric value.
`t1`	The time to maturity of the call option, measured in years.
`t2`	The time to maturity of the put option, measured in years.
`A`	The corresponding asset price has exceeded the exercise price X.

Value

The original `OptPx` object and the option pricing parameters `t1`, `t2`,k,`A`, and computed price `PxBS`.

Author(s)

Le You, Department of Statistics, Rice University, Spring 2015

References

Hull, J.C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, http://www-2.rotman.utoronto.ca/~hull/ofod/index.html
Haug, Espen G.,*Option Pricing Formulas*, 2ed.

Examples

```
(o = HolderExtendibleBS())$PxBS

o = Opt(Style='HolderExtendible',Right='Call', S0=100, ttm=0.5, K=100)
o = OptPx(o,r=0.08,q=0,vol=0.25)
(o = HolderExtendibleBS(o,k=105,t1=0.5,t2=0.75,A=1))$PxBS

o = Opt("HolderExtendible","Put", S0=100, ttm=0.5, K=100)
o = OptPx(o,r=0.08,q=0,vol=0.25)
(o = HolderExtendibleBS(o,k=90,t1=0.5,t2=0.75,A=1))$PxBS
```

is.Opt *Is an object* `Opt`*?*

Description

Tests the argument for the specific class type.

Usage

```
is.Opt(o)
```

Arguments

o	Any object

Value

TRUE if and only if an argument is of `Opt` class.

Author(s)

Oleg Melnikov

Examples

```
is.Opt(Opt())  #verifies that Opt() returns an object of class \code{Opt}
is.Opt(1:3)    #verifies that code{1:3} is not an object of class \code{Opt}
```

`is.OptPos` *Is an object* `OptPos`*?*

Description

Tests the argument for the specific class type.

Usage

```
is.OptPos(o)
```

Arguments

`o`	Any object

Value

TRUE if and only if an argument is of `OptPos` class.

Author(s)

Oleg Melnikov

Examples

```
is.OptPos(OptPos())
```

`is.OptPx` *Is an object* `OptPx`*?*

Description

Tests the argument for the specific class type.

Usage

```
is.OptPx(o)
```

Arguments

`o`	Any object

Value

TRUE if and only if an argument is of `OptPx` class.

Author(s)

Oleg Melnikov

Examples

```
is.OptPx(OptPx(Opt(S0=20), r=0.12))
```

LadderMC — *Ladder option valuation via Monte Carlo (MC) simulation.*

Description

Calculates the price of a Ladder Option using 5000 Monte Carlo simulations. The helper function LadderCal() aims to calculate expected payout for each stock prices.

Important Assumptions: The option o follows a General Brownian Motion (BM) $ds = mu * S * dt + sqrt(vol) * S * dW$ where $dW\ N(0,1)$. The value of mu (the expected price increase) is assumed to be `o$r`, the risk free rate of return.

Usage

```
LadderMC(o = OptPx(o = Opt(Style = "Ladder"), NSteps = 5), NPaths = 5,
  L = c(60, 80, 100))
```

Arguments

`o`	The `OptPx` Ladder option object to price.
`NPaths`	The number of simulation paths to use in calculating the price
`L`	A series of ladder strike price.

Value

The option `o` with the price in the field `PxMC` based on MC simulations and the ladder strike price `L` set by the users themselves

Author(s)

Huang Jiayao, Risk Management and Business Intelligence at Hong Kong University of Science and Technology, Exchange student at Rice University, Spring 2015

References

`http://stackoverflow.com/questions/25946852/r-monte-carlo-simulation-price-path-converging-volatil`

Examples

```
(o = LadderMC())$PxMC #Price = ~12.30

 o = OptPx(o=Opt(Style='Ladder'), NSteps = 5)
 (o = LadderMC(o))$PxMC        #Price = ~11.50

 o = OptPx(Opt(Style='Ladder', Right='Put'))
 (o = LadderMC(o, NPaths = 5))$PxMC   # Price = ~12.36

 (o = LadderMC(L=c(55,65,75)))$PxMC   # Price = ~10.25
```

LookbackBS — *Lookback option valuation with Black-Scholes (BS) model*

Description

Calculates the price of a lookback option using a BSM-adjusted algorithm; Carries the assumption that the asset price is observed continuously.

Usage

```
LookbackBS(o = OptPx(Opt(Style = "Lookback")), Smax = 50, Smin = 50,
  Type = c("Floating", "Fixed"))
```

Arguments

o	An object of class OptPx.
Smax	The maximum asset price observed to date.
Smin	The minimum asset price observed to date.
Type	Specifies the Lookback option as either Floating or Fixed- default argument is Floating.

Details

To price the lookback option, we require the Smax/Smin, S0, r, q, vol, and ttm arguments from the object classes defined in the package. An example of a complete OptLookback option object can be found in the examples.

Value

An original `OptPx` object with `PxBS` field as the price of the option and user-supplied `Smin`, `Smax`, and `Type` lookback parameters attached.

Author(s)

Richard Huang, Department of Statistics, Rice University, Spring 2015

References

Hull, J.C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, http://www-2.rotman.utoronto.ca/~hull/ofod/index.html.

Examples

```
(o = LookbackBS())$PxBS
  LookbackBS(OptPx(Opt(Style = 'Lookback'))) #Uses default arguments

  # See Hull 9e Example 26.2, p.608; gives price of 7.79
  o = Opt(Style = 'Lookback', S0 = 50, ttm= .25, Right = "Put")
  o = OptPx(o,r = .1, vol = .4)
  o = LookbackBS(o, Type = "Floating")

  # See Hull 9e Example 26.2, p.608; gives price of 8.04
  o = Opt(Style = 'Lookback', S0 = 50, ttm= .25, Right = "Call")
  o = OptPx(o, r = .1, vol = .4)
  o = LookbackBS(o, Type = "Floating")

  # Price = 17.7129
  o = Opt(Style = 'Lookback', S0 = 50, ttm= 1, Right = "Put", K = 60)
  o = OptPx(o,r = .05, q = .02, vol = .25)
  o = LookbackBS(o, Type = "Fixed")

  # Price = 8.237
  o = Opt(Style = 'Lookback', S0 = 50, ttm= 1, Right = "Call", K = 55)
  o = OptPx(o,r = .1, q = .02, vol = .25)
  o = LookbackBS(o, Type = "Fixed")
```

LookbackMC — *Lookback option valuation via Monte Carlo (MC) simulation*

Description

Calculates the price of a lookback option using a Monte Carlo (MC) Simulation. Carries the assumption that the asset price is observed continuously. Assumes that the the option o followes ds = mu * S * dt + sqrt(vol) * S * dz where dz is a Wiener Process. Assume that without dividends, mu are default to be r.

Usage

```
LookbackMC(o = OptPx(Opt(Style = "Lookback"), r = 0.05, q = 0, vol = 0.3),
  NPaths = 5, div = 1000, Type = c("Floating", "Fixed"))
```

Arguments

`o`	The `OptPx` option object to price. See `OptPx` and `Opt` for more information.
`NPaths`	How many time of the simulation are applied. Coustomer defined.

`div`	number to decide length of each interval
`Type`	Specifies the Lookback option as either Floating or Fixed- default argument is Floating.

Details

To price the lookback option, we require the S0, K, and ttm arguments from object `Opt` and r, q, vol from object OptPx defined in the package. The results of simulation would unstable without setting seeds.

Value

A list of class `LookbackMC` consisting of the input object `OptPx` and the price of the lookback option based on Monte Carlo Simulation (see references).

Author(s)

Tong Liu, Department of Statistics, Rice University, Spring 2015

References

Hull, John C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, `http://www-2.rotman.utoronto.ca/~hull/ofod`

Examples

```
(o = LookbackMC())$PxMC   #Use default arguments, Output: approximately 16.31.

 # Floating & Put
 o=OptPx(Opt(S0=50,K=50,ttm=0.25,Right='Put',Style="Lookback"),r=0.1,vol=.4)
 LookbackMC(o,NPaths=5,div=1000) #Output: 7.79 from Hull 9e Example 26.2 Pg 608.

 # Floating & Call
 o=OptPx(Opt(S0=50,K=50,ttm=0.25,Right='Call',Style="Lookback"),r=0.1,vol=.4)
 LookbackMC(o,NPaths=5,div=1000) #Output: 8.04 from Hull 9e Example 26.2 Pg 608

 # Fixed & Put
 o=OptPx(Opt(S0=50,K=60,ttm=1,Right='Put',Style="Lookback"),r=0.05,q=0.02,vol=.25)
 LookbackMC(o,Type="Fixed",NPaths=5,div=1000)

 # Fixed & Call
 o=OptPx(Opt(S0=50,K=55,ttm=1,Right='Call',Style="Lookback"),r=0.1,vol=.25)
 LookbackMC(o,Type="Fixed",NPaths=5,div=1000)
```

Opt — Opt *object constructor*

Description

An S3 object constructor for an option contract (financial derivative)

Usage

```
Opt(Style = c("European", "American", "Asian", "Binary", "AverageStrike",
  "Barrier", "Chooser", "Compound", "DeferredPayment", "ForeignEquity",
  "ForwardStart", "Gap", "HolderExtendible", "Ladder", "Lookback", "MOPM",
  "Perpetual", "Quotient", "Rainbow", "Shout", "SimpleChooser", "VarianceSwap"),
  Right = c("Call", "Put", "Other"), S0 = 50, ttm = 2, K = 52,
  Curr = "$", ContrSize = 100, SName = "A stock share", SSymbol = "")
```

Arguments

`Style`	An option style: `European` or `American`. Partial names are allowed, eg. `E` or `A`
`Right`	An option right: `Call` or `Put`. Partial names are allowed.
`S0`	A spot price of the underlying security (usually, today's stock price, S_0)
`ttm`	A time to maturity, in units of time matching r units; usually years
`K`	A strike price
`Curr`	An optional currency units for monetary values of the underlying security and an option
`ContrSize`	A contract size, i.e. number of option shares per contract
`SName`	A (optional) descriptful name of the underlying. Eg. *Microsoft Corp*
`SSymbol`	An (optional) official ticker of the underlying. Eg. *MSFT*

Value

A list of class `Opt`

Author(s)

Oleg Melnikov, Department of Statistics, Rice University, Spring 2015

Examples

```
Opt()  #Creates an S3 object for an option contract
Opt(Right='Put')   #See J. C. Hull, OFOD'2014, 9-ed, Fig.13.10, p.289
```

`OptPos`	`OptPos` *object constructor*

Description

S3 object constructor for lattice-pricing specs of an option contract. Inherits `Opt` object.

Usage

```
OptPos(o = Opt(), Pos = c("Long", "Short"), Prem = 0)
```

Arguments

`o`	An object of class `Opts`
`Pos`	A position direction (to the holder) with values `Long` for owned option contract and `Short` for shorted contract.
`Prem`	A option premim (i.e. market cost or price), a non-negative amount to be paid for the option contract being modeled.

Value

A list of class `OptPx`

Author(s)

Oleg Melnikov, Department of Statistics, Rice University, Spring 2015

Examples

```
OptPos()  # Creates an S3 object for an option contract
OptPos(Opt(Right='Put'))  #See J.C.Hull, OFOD'2014, 9-ed, Fig.13.10, p.289
```

`OptPx`	`OptPx` *object constructor*

Description

An S3 object constructor for lattice-pricing specifications for an option contract. `Opt` object is inhereted.

Usage

```
OptPx(o = Opt(), r = 0.05, q = 0, rf = 0, vol = 0.3, NSteps = 3)
```

Arguments

`o`	An object of class `Opt`
`r`	A risk free rate (annualized)
`q`	A dividend yield (as annualized rate), Hull/p291
`rf`	A foreign risk free rate (annualized), Hull/p.292
`vol`	A volaility (as Sd.Dev, sigma)
`NSteps`	A number of time steps in BOPM calculation

Value

A list of class `OptPx` with parameters supplied to `Opt` and `OptPx` constructors

Author(s)

Oleg Melnikov, Department of Statistics, Rice University, Spring 2015

Examples

```
OptPx()  #Creates an S3 object for an option contract

#See J.C.Hull, OFOD'2014, 9-ed, Fig.13.10, p.289
OptPx(Opt(Right='Put'))

o = OptPx(Opt(Right='Call', S0=42, ttm=.5, K=40), r=.1, vol=.2)
```

`pbnorm` *Bivariate Standard Normal CDF*

Description

Bivariate Standard Normal CDF Calculator For Given Values of x, y, and rho

Usage

```
pbnorm(x = 0, y = 0, rho = 0)
```

Arguments

`x`	The x value (want probability under this value of x); values in $(-25, 25)$
`y`	The y value (want probability under this value of y); values in $(-25, 25)$
`rho`	The correlation between variables x and y; values in $[-1, 1]$

Details

This runs a bivariate standard normal pdf then calculates the cdf from that based on the input parameters

Value

Density under the bivariate standard normal distribution

Author(s)

Robert Abramov, Department of Statistics, Rice University, 2015

References

Adapted from "Bivariate normal distribution with R", Edouard Tallent's blog from Sep 21, 2012 https://quantcorner.wordpress.com/2012/09/21/bivariate-normal-distribution-with-r

Examples

```
pbnorm(1, 1, .5)
#pbnorm(2, 2, 0)
#pbnorm(-1, -1, .35)
#pbnorm(0, 0, 0)

ttl = 'cdf of x, at y=0'
X = seq(-5,5,1)
graphics::plot(X, sapply(X, function(x) pbnorm(0,x,0)), type='l', main=ttl)
```

PerpetualBS *Perpetual option valuation via Black-Scholes (BS) model*

Description

An exotic option is an option which has features making it more complex than commonly traded options. A perpetual option is non-standard financial option with no fixed maturity and no exercise limit. While the life of a standard option can vary from a few days to several years, a perpetual option (XPO) can be exercised at any time. Perpetual options are considered an American option. European options can be exercised only on the option's maturity date.

Usage

```
PerpetualBS(o = OptPx(Opt(Style = "Perpetual"), q = 0.1))
```

Arguments

`o`	AN object of class `OptPx`

Value

A list of class `Perpetual.BS` consisting of the input object `OptPx`

Author(s)

Kim Raath, Department of Statistics, Rice University, Spring 2015.

References

Chi-Guhn Lee, *The Black-Scholes Formula*, Courses, Notes, Note2, Sec 1.5 and 1.6 http://www.mie.utoronto.ca/courses/mie566f/materials/note2.pdf

Examples

```
#Perpetual American Call and Put
#Verify pricing with \url{http://www.coggit.com/freetools}
(o <- PerpetualBS())$PxBS # Approximately valued at $8.54

#This example should produce approximately $33.66
o = Opt(Style="Perpetual", Right='Put', S0=50, K=55)
o = OptPx(o, r = .03, q = 0.1, vol = .4)
(o = PerpetualBS(o))$PxBS

#This example should produce approximately $10.87
o = Opt(Style="Perpetual", Right='Call', S0=50, K=55)
o = OptPx(o, r = .03, q = 0.1, vol = .4)
(o <- PerpetualBS(o))$PxBS
```

`Profit` *Computes payout/profit values*

Description

Computes payout/profit values

Usage

```
Profit(o = OptPos(), S = o$S0)
```

Arguments

o	An object of class `Opt*`
S	A (optional) vector or value of stock price(s) (double) at which to compute profits

Value

A numeric matrix of size `[length(S), 2]`. Columns: stock prices, corresponding option profits

Author(s)

Oleg Melnikov, Department of Statistics, Rice University, Spring 2015

Examples

```
Profit(o=Opt())
graphics::plot( print( Profit(OptPos(Prem=2.5), S=40:60)), type='l'); grid()
```

`QuotientBS` *Quotient option valuation via Black-Scholes (BS) model*

Description

Quotient Option via Black-Scholes (BS) model

Usage

```
QuotientBS(o = OptPx(Opt(Style = "Quotient")), I1 = 100, I2 = 100,
  g1 = 0.04, g2 = 0.03, sigma1 = 0.18, sigma2 = 0.15, rho = 0.75)
```

Arguments

`o`	An object of class `OptPx`
`I1`	A spot price of the underlying security 1 (usually I1)
`I2`	A spot price of the underlying security 2 (usually I2)
`g1`	Payout rate of the first stock
`g2`	Payout rate of the 2nd stock
`sigma1`	a vector of implied volatilities for the associated security 1
`sigma2`	a vector of implied volatilities for the associated security 2
`rho`	is the correlation between asset 1 and asset 2

Value

A list of class `QuotientBS` consisting of the original `OptPx` object and the option pricing parameters `I1`,`I2`, `Type`, `isForeign`, and `isDomestic` as well as the computed price `PxBS`.

Author(s)

Chengwei Ge, Department of Statistics, Rice University, Spring 2015

References

Zhang Peter G., *Exotic Options*, 2nd, 1998. `http://amzn.com/9810235216`.

Examples

```
(o = QuotientBS())$PxBS

o = OptPx(Opt(Style = 'Quotient', Right = "Put"), r= 0.05)
(o = QuotientBS(o, I1=100, I2=100, g1=0.04, g2=0.03, sigma1=0.18,sigma2=0.15, rho=0.75))$PxBS

o = OptPx(Opt(Style = 'Quotient',  Right = "Put", ttm=1, K=1), r= 0.05)
QuotientBS(o, I1=100, I2=100, g1=0.04, g2=0.03, sigma1=0.18,sigma2=0.15, rho=0.75)

o = OptPx(Opt(Style = 'Quotient',  Right = "Call", ttm=1, K=1), r= 0.05)
QuotientBS(o, I1=100, I2=100, g1=0.04, g2=0.03, sigma1=0.18,sigma2=0.15, rho=0.75)
```

QuotientMC *Quotient option valuation via Monte Carlo (MC) model*

Description

Calculates the price of a Quotient option using Monte-Carlo simulations.

Usage

```
QuotientMC(o = OptPx(Opt(Style = "Quotient")), S0_2 = 100, NPaths = 5)
```

Arguments

`o`	The `OptQuotient` option object to price.
`S0_2`	The spot price of the second underlying asset.
`NPaths`	Number of monte-carlo simulations to run. Larger number of trials lower variability at the expense of computation time.

Details

The Monte-Carlo simulations assume the underlying price undergoes Geometric Brownian Motion (GBM). Payoffs are discounted at risk-free rate to price the option. A thorough understanding of the object class construction is recommended. Please see `OptPx`, `Opt` for more information.

Value

An original `OptPx` object with Px.MC field as the price of the option and user-supplied S0_2, NPaths parameters attached.

Author(s)

Richard Huang, Department of Statistics, Rice University, Spring 2015

References

http://www.investment-and-finance.net/derivatives/q/quotient-option.html

Examples

```
(o = QuotientMC())$PxMC #Default Quotient option price.

o = OptPx(Opt(S0=100, ttm=1, K=1.3), r=0.10, q=0, vol=0.1)
(o = QuotientMC(o, S0_2 = 180, NPaths=5))$PxMC

QuotientMC(OptPx(Opt()), S0_2 = 180, NPaths=5)

QuotientMC(OptPx(), S0_2 = 201, NPaths = 5)

QuotientMC(OptPx(Opt(S0=500, ttm=1, K=2)), S0_2 = 1000, NPaths=5)
```

RainbowBS — *Rainbow option valuation via Black-Scholes (BS) model*

Description

Rainbow Option via Black-Scholes (BS) model

Usage

```
RainbowBS(o = OptPx(Opt(Style = "Rainbow")), S1 = 100, S2 = 95, D1 = 0,
  D2 = 0, sigma1 = 0.15, sigma2 = 0.2, rho = 0.75, Type = c("Max",
  "Min"))
```

Arguments

`o`	An object of class `OptPx`
`S1`	A spot price of the underlying security 1 (usually S1)
`S2`	A spot price of the underlying security 2 (usually S2)
`D1`	A percent yield per annum from the underlying security 1
`D2`	A percent yield per annum from the underlying security 2
`sigma1`	a vector of implied volatilities for the associated security 1
`sigma2`	a vector of implied volatilities for the associated security 2
`rho`	is the correlation between asset 1 and asset 2
`Type`	Rainbow option type: ’Max’ or ’Min’.

Details

Two types of Rainbow options are priced: `'Max'` and `'Min'`.

Value

A list of class `RainbowBS` consisting of the original `OptPx` object and the option pricing parameters `S1`, `Type`, `isMax`, and `isMin` as well as the computed price `PxBS`.

Author(s)

Chengwei Ge,Department of Statistics, Rice University, Spring 2015

References

Zhang Peter G., *Exotic Options*, 2nd ed, 1998.

Examples

```
(o = RainbowBS())$PxBS

  o = OptPx(Opt(Style = 'Rainbow',  Right = "Put"), r = 0.08)
  RainbowBS(o, S1=100, S2=95, D1=0,D2=0,sigma1=0.15,sigma2=0.2, rho=0.75,Type='Min')

  o = OptPx(Opt(Style = 'Rainbow', K = 102, ttm = 1, Right = "Put"), r = 0.08)
  RainbowBS(o, S1=100, S2=95, D1=0,D2=0,sigma1=0.15,sigma2=0.2, rho=0.75,Type='Min')

  o=OptPx(Opt(Style = 'Rainbow', K = 102, ttm = 1, Right = "Put"), r = 0.08)
  RainbowBS(o, S1=100, S2=95, D1=0,D2=0,sigma1=0.15,sigma2=0.2, rho=0.75,Type='Max')

  o=OptPx(Opt(Style = 'Rainbow', K = 102, ttm = 1, Right = "Call"), r = 0.08)
  RainbowBS(o, S1=100, S2=95, D1=0,D2=0,sigma1=0.15,sigma2=0.2, rho=0.75,Type='Min')

  o=OptPx(Opt(Style = 'Rainbow', K = 102, ttm = 1, Right = "Call"), r = 0.08)
  RainbowBS(o, S1=100, S2=95, D1=0,D2=0,sigma1=0.15,sigma2=0.2, rho=0.75,Type='Max')
```

ShoutFD *Shout option valuation via finite differences (FD) method*

Description

Shout option valuation via finite differences (FD) method

Usage

```
ShoutFD(o = OptPx(Opt(Style = "Shout")), N = 100, M = 20, Smin = 0,
  Smax = 100)
```

Arguments

`o`	An object of class `OptPx`
`N`	The number of equally spaced intervals. Default is 100.
`M`	The number of equally spaced stock price. Default is 20.
`Smin`	similar to Smax
`Smax`	A stock price sufficiently high that, when it is reached, the put option has virtually no value. The level of Smax should be chosen in such a way that one of these equally spaced stock prices is the current stock price.

Details

A shout option is a European option where the holder can 'shout' to the writer at one time during its life. At the end of the life of the option, the option holder receives either the usual payoff from a European option or the intrinsic value at the time of the shout, whichever is greater. An explicit finite difference method (Page 482 in Hull's book) is used here to price the shout put option. Similar to pricing American options, the value of the option is consolidated at each node of the grid to see if shouting would be optimal. The corresponding shout call option is priced using the Put-Call-Parity in the finite difference method .

Value

A list of class `OptPx`, including option pricing parameters `N`, `M`, `Smin`, and `Smax`, as well as the computed option price `PxFD`.

Author(s)

Xinnan Lu, Department of Statistics, Rice University, 2015

References

Hull, J.C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, http://www-2.rotman.utoronto.ca/~hull/ofod/index.html pp.609.

Examples

```
(o = ShoutFD(OptPx(Opt(Right="C", Style="Shout"))))$PxFD

 o = OptPx(Opt(Right="C", Style="Shout"))
 (o = ShoutFD(o, N=10))$PxFD # very differnt result for N=10

 (o = ShoutFD(OptPx(Opt(Right="P", Style="Shout"))))$PxFD

 o = Opt(Right='P', S0=100, K=110, ttm=0.5, Style='Shout')
 o = OptPx(o, vol=0.2, r=0.05, q=0.04)
 (o = ShoutFD(o,N=100,Smax=200))$PxFD

 o = Opt(Right="C", S0=110, K=100, ttm=0.5, Style="Shout")
 o = OptPx(o, vol=0.2, r=0.05, q=0.04)
 (o = ShoutFD(o,N=100,Smax=200))$PxFD
```

ShoutLT *Shout option valuation via lattice tree (LT)*

Description

A shout option is a European option where the holder can shout to the writer at one time during its life. At the end of the life of the option, the option holder receives either the usual payoff from a European option or the instrinsic value at the time of the shout, which ever is greater. $max(0, S_T - S_tau) + (S_tau - K)$

Usage

```
ShoutLT(o = OptPx(Opt(Style = "Shout")), IncBT = TRUE)
```

Arguments

`o`	An object of class `OptPx`
`IncBT`	TRUE/FALSE indicating whether to include binomial tree (list object) with output

Value

A list of class `ShoutLT` consisting of the original `OptPx` object, binomial tree step`BT` and the computed price `PxBS`.

Author(s)

Le You, Department of Statistics, Rice University, Spring 2015

References

Hull, J.C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, http://www-2.rotman.utoronto.ca/~hull/ofod. http://amzn.com/0133456315

Examples

```
(o = ShoutLT( OptPx(Opt(Style='Shout'))))$PxLT

o = Opt(Style='Shout', Right='Call', S0=60, ttm=.25, K=60)
ShoutLT( OptPx(o,r=.1, q=.02, vol=.45, NSteps=10))

o = Opt(Style='Shout', Right='Call', S0=60, ttm=.25, K=60)
```

`ShoutLTVectorized` *Shout option valuation via lattice tree (LT)*

Description

A shout option is a European option where the holder can shout to the writer at one time during its life. At the end of the life of the option, the option holder receives either the usual payoff from a European option or the instrinsic value at the time of the shout, which ever is greater. $max(0, S_T - S_tau) + (S_tau - K)$

Usage

```
ShoutLTVectorized(o = OptPx(o = Opt(Style = "Shout")))
```

Arguments

`o`	An object of class `OptPx`

Value

A list of class `ShoutLT` consisting of the original `OptPx` object, binomial tree step `BT` and the computed price `PxBS`.

Author(s)

Le You, Department of Statistics, Rice University, Spring 2015

References

Hull, J.C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, http://www-2.rotman.utoronto.ca/~hull/ofod. http://amzn.com/0133456315

Examples

```
(o = ShoutLTVectorized( OptPx(Opt(Style='Shout'))))$PxLT

o = Opt(Style='Shout')
(o = ShoutLTVectorized( OptPx(o, r=.1, q=.02, vol=.45, NSteps=10)))$PxLT
```

ShoutMC — *Shout option valuation via Monte Carlo (MC) simulations.*

Description

Calculates the price of a shout option using Monte Carlo simulations to determine expected payout. Assumes that the option follows a General Brownian Motion (GBM) process, $ds = mu * S * dt + sqrt(vol) * S * dW$ where $dW\ N(0, 1)$. Note that the value of mu (the expected price increase) is assumed to be `o$r`, the risk free rate of return.

Usage

```
ShoutMC(o = OptPx(o = Opt(Style = "Shout")), NPaths = 10)
```

Arguments

`o`	The `OptPx` Shout option to price.
`NPaths`	The number of simulation paths to use in calculating the price; must be >= 10

Value

The option object `o` with the price in the field `PxMC` based on the MC simulations.

Author(s)

Jake Kornblau, Department of Statistics, Rice University, 2015

References

Hull, J.C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, http://www-2.rotman.utoronto.ca/~hull/ofod/index.html.
Also: http://www.math.umn.edu/~spirn/5076/Lecture16.pdf

Examples

```
(o = ShoutMC())$PxMC # Approximately valued at $11

  o = Opt(Style='Shout')
  (o = ShoutMC(OptPx(o, NSteps = 5)))$PxMC # Approximately valued at $18.6

  o = Opt(Style='Shout',S0=110,K=100,ttm=.5)
  o = OptPx(o, r=.05, vol=.2, q=0, NSteps = 10)
  (o = ShoutMC(o, NPaths = 10))$PxMC
```

VarianceSwapBS *Variance Swap valuation via Black-Scholes (BS) model*

Description

Variance Swap valuation via Black-Scholes (BS) model

Usage

```
VarianceSwapBS(o = OptPx(Opt(Style = "VarianceSwap", Right = "Other", ttm =
  0.25, S0 = 1020), r = 0.04, q = 0.01), K = seq(800, 1200, 50),
  Vol = seq(0.2, 0.24, 0.005), notional = 10^8, varrate = 0.045)
```

Arguments

`o`	An object of class `OptPx`
`K`	A vector of non-negative strike prices
`Vol`	a vector of non-negative, less than zero implied volatilities for the associated strikes
`notional`	A numeric positive amount to be invested
`varrate`	A numeric positive varaince rate to be swapped

Value

An object of class `OptPx` with value included

Author(s)

Max Lee, Department of Statistics, Rice University, Spring 2015

References

Hull, J.C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, http://www-2.rotman.utoronto.ca/~hull/ofod.

Examples

```
(o = VarianceSwapBS())$PxBS

o = Opt(Style="VarianceSwap",Right="Other",ttm=.25,S0=1020)
o = OptPx(o,r=.04,q=.01)
Vol = Vol=c(.29,.28,.27,.26,.25,.24,.23,.22,.21)
(o = VarianceSwapBS(o,K=seq(800,1200,50),Vol=Vol,notional=10^8,varrate=.045))$PxBS

o = Opt(Style="VarianceSwap",Right="Other",ttm=.25,S0=1020)
o = OptPx(o,r=.04,q=.01)
Vol=c(.2,.205,.21,.215,.22,.225,.23,.235,.24)
(o =VarianceSwapBS(o,K=seq(800,1200,50),Vol=Vol,notional=10^8,varrate=.045))$PxBS

o = Opt(Style="VarianceSwap",Right="Other",ttm=.1,S0=100)
o = OptPx(o,r=.03,q=.02)
Vol=c(.2,.19,.18,.17,.16,.15,.14,.13,.12)
(o =VarianceSwapBS(o,K=seq(80,120,5),Vol=Vol,notional=10^4,varrate=.03))$PxBS
```

`VarianceSwapMC` *VarianceSwap option valuation via Monte Carlo (MC) simulation.*

Description

Calculates the price of a VarianceSwap Option using 500 Monte Carlo simulations.
Important Assumptions: The option o followes a General Brownian Motion $ds = mu * S * dt + sqrt(vol) * S * dW$ where $dW\ N(0,1)$. The value of mu (the expected price increase) is assumed to be `o$r-o$q`.

Usage

```
VarianceSwapMC(o = OptPx(o = Opt(Style = "VarianceSwap")), var = 0.2,
  NPaths = 5)
```

Arguments

`o`	The `OptPx` Variance Swap option to price.
`var`	The variance strike level
`NPaths`	The number of simulation paths to use in calculating the price,

Value

The option `o` with the price in the field `PxMC` based on MC simulations and the Variance Swap option properties set by the users themselves

Author(s)

Huang Jiayao, Risk Management and Business Intelligence at Hong Kong University of Science and Technology, Exchange student at Rice University, Spring 2015

References

Hull, J.C., *Options, Futures and Other Derivatives*, 9ed, 2014. Prentice Hall. ISBN 978-0-13-345631-8, http://www-2.rotman.utoronto.ca/~hull/ofod.
http://stackoverflow.com/questions/25946852/r-monte-carlo-simulation-price-path-converging-volatil

Examples

```
(o = VarianceSwapMC())$PxMC #Price = ~0.0245

 (o = VarianceSwapMC(NPaths = 5))$PxMC # Price = ~0.0245

 (o = VarianceSwapMC(var=0.4))$PxMC # Price = ~-0.1565
```

Index

as.OptPos, 3
AsianBS, 4
AsianMC, 5
AverageStrikeMC, 6

BarrierBS, 7
BarrierLT, 8
BarrierMC, 10
Binary_BOPM, 13
BinaryBS, 11
BinaryMC, 12
BOPM, 14, *16*
BOPM_Eu, *15*, 15
BS, 16
BS_Simple, 17

ChooserBS, 18
ChooserLT, 19
ChooserMC, 20
CompoundBS, 21
CompoundLT, 22

DeferredPaymentLT, 24

ForeignEquityBS, 25
ForwardStartBS, 26
ForwardStartMC, 27

GapBS, 28
GapLT, 29
GapMC, 30

HolderExtendibleBS, 31

is.Opt, 32
is.OptPos, 33
is.OptPx, 33

LadderMC, 34
LookbackBS, 35
LookbackMC, 36

Opt, 38
OptPos, 39
OptPx, 39

pbnorm, 40
PerpetualBS, 41
Profit, 42

QuotientBS, 43
QuotientMC, 44

RainbowBS, 45

ShoutFD, 46
ShoutLT, 47
ShoutLTVectorized, 48
ShoutMC, 49

VarianceSwapBS, 50
VarianceSwapMC, 51