Package ‘fExpressCertificates’

January 10, 2019

Type Package
Title Structured Products Valuation for ExpressCertificates/Autocallables
Version 1.3
Date 2019-01-08
Author Stefan Wilhelm
Maintainer Stefan Wilhelm <stefan.wilhelm@financial.com>
Depends R (>= 2.15.0), tmvtnorm, fCertificates
Imports mvtnorm, Matrix, fExoticOptions, fOptions
Encoding latin1
Description Provides pricing by duplication and Monte Carlo methods for Express Certificates products (also known as Autocallables).
License GPL (>= 2)
LazyLoad yes
URL https://www.r-project.org
NeedsCompilation no
Repository CRAN
Date/Publication 2019-01-10 00:10:03 UTC

R topics documented:
calcBMProbability .................................................. 2
calcGBMProbability .................................................. 4
Distribution of the Brownian Bridge Minimum ...................... 6
Express Certificates Redemption Probabilities .................... 8
ExpressCertificate.Classic ....................................... 10
GeometricBrownianMotion ........................................ 11
getRedemptionTime ............................................... 12
MonteCarlo.ExpressCertificate.Classic ............................ 13
payoffExpress ...................................................... 14
calcBMProbability

*Calculates probabilities for the Arithmetic Brownian Motion*

**Description**

This method is a compilation of formulas for some (joint) probabilities for the Arithmetic Brownian Motion $B_t = B(t)$ with drift parameter $\mu$ and volatility $\sigma$ and its minimum $m_t = m(t)$ or maximum $M_t = M(t)$.

**Usage**

```r
calcBMProbability(
  Type = c(
    "P(M_t >= a)",
    "P(M_t <= a)",
    "P(m_t <= a)",
    "P(m_t >= a)",
    "P(M_t >= a, B_t <= z)",
    "P(m_t <= a, B_t >= z)",
    "P(a <= m_t, M_t <= b)",
    "P(M_s >= a, B_t <= z | s < t)",
    "P(m_s <= a, B_t >= z | s < t)",
    "P(M_s >= a, B_t <= z | s > t)",
    "P(m_s <= a, B_t >= z | s > t)"
  ),
  a, z, t, mu, sigma, s)
```

**Arguments**

- **Type**: Type of probability to be calculated, see details.
- **a**: level
- **z**: level
- **t**: point in time, $t > 0$
- **mu**: Brownian Motion drift term $\mu$
- **sigma**: Brownian Motion volatility $\sigma$
- **s**: Second point in time, used by some probabilities like $P(M_s >= a, B_t <= z | s < t)$
Details

Let \( M_t = \max(B_t) \) and \( m_t = \min(B_t) \) for \( t > 0 \) be the running maximum/minimum of the Brownian Motion up to time \( t \) respectively.

- \( P(M_t \geq a) \) (\( P(M_t \leq a) \)) is the probability of the maximum \( M_t \) exceeding (staying below) a level \( a \) up to time \( t \). See Chuang (1996), equation (2.3).
- \( P(m_t \leq a) \) (\( P(m_t \geq a) \)) is the probability of the minimum \( m_t \) to fall below (rise above) a level \( a \) up to time \( t \).
- \( P(M_t \geq a, B_t \leq z) \) is the joint probability of the maximum exceeding level \( a \) while the Brownian Motion is below level \( z \) at time \( t \). See Chuang (1996), equation (2.1), p.82.
- \( P(m_t \leq a, B_t \geq z) \) is the joint probability of the minimum to be below level \( a \), while the Brownian Motion is above level \( z \) at time \( t \).
- \( P(M_s \geq a, B_t \leq z | s < t) \) See Chuang (1996), equation (2.7), p.84 for the joint probability \((M_s, B_t)\) of the maximum \( M_s \) and the Brownian Motion \( B_t \) at different times \( s < t \)
- \( P(m_s \leq a, B_t \geq z | s < t) \) See Chuang (1996), equation (2.7), p.84 for the joint probability \((m_s, B_t)\) of the minimum \( m_s \) and the Brownian Motion \( B_t \) at different times \( s < t \).
- \( P(M_s \geq a, B_t \leq z | s > t) \) See Chuang (1996), equation (2.9), p.85 for the joint probability \((M_s, B_t)\) of the maximum \( M_s \) and the Brownian Motion \( B_t \) at different times \( s > t \).
- \( P(m_s \leq a, B_t \geq z | s > t) \) See Chuang (1996), equation (2.9), p.85 for the joint probability \((m_s, B_t)\) of the minimum \( m_s \) and the Brownian Motion \( B_t \) at different times \( s > t \). Adapted this formula for the minimum \((m_s, B_t)\)

Some identities:

For \( s < t \):

\[
P(M_s \leq a, M_t \geq a, B_t \leq z) = P(M_t \geq a, B_t \leq z) - P(M_s \geq a, B_t \leq z)
\]

\[
P(M_s \geq a, B_t \leq z) = P(M_s \geq a) - P(M_s \geq a, B_t \geq z)
\]

\[
P(X \leq -x, Y \leq -y) = P(-X \geq x, -Y \geq y) = 1 - P(-X \leq x) - P(-Y \leq y) + P(-X \leq x, -Y \leq y)
\]

Changing from maximum \( M_t \) of \( B_t \) to minimum \( m_t^* \) of \( B_t^* = -B_t \):

\( P(M_t \geq z) \) becomes \( P(m_t^* \leq -z) \).

Value

The method returns a vector of probabilities, if used with vector inputs.

Author(s)

Stefan Wilhelm <wilhelm@financial.com>

References

Examples

```r
# Example 1: Maximum M_t of Brownian motion

# simulate 1000 discretized paths from Brownian Motion B_t
B <- matrix(NA,1000,101)
for (i in 1:1000) {
  B[i,] <- BrownianMotion(S0=100, mu=0.05, sigma=1, T=1, N=100)
}

# get empirical Maximum M_t
M_t <- apply(B, 1, max, na.rm=TRUE)
plot(density(M_t, from=100))

# empirical CDF of M_t
plot(ecdf(M_t))
a <- seq(100, 103, by=0.1)
P(M_t <= a)
# 1-cdf.m_t(a-100, t=1, mu=0.05, sigma=1)
p <- calcBMPProbability(Type = "P(M_t <= a)", a-100, t = 1, 
  mu = 0.05, sigma = 1)
lines(a, p, col="red")
```

# Example 2: Minimum m_t of Brownian motion

# Minimum m_t : Drift ändern von 0.05 auf -0.05
m_t <- apply(B, 1, min, na.rm=TRUE)
a <- seq(97, 100, by=0.1)
# cdf.m_t(a-100, t=1, mu=0.05, sigma=1)
p <- calcBMPProbability(Type = "P(m_t <= a)", a-100, t = 1, mu = 0.05, sigma = 1)
plot(ecdf(m_t))
lines(a, p, col="blue")
```

**calcGBMProbability**

*Calculates probabilities for the Geometric Brownian Motion*

**Description**

This method is a compilation of formulas for some (joint) probabilities for the Geometric Brownian Motion $S_t = S(t)$ with drift parameter $\mu$ and volatility $\sigma$ and its minimum $m_t = m(t) = \min_{0 \leq \tau \leq t} S(\tau)$ and its maximum $M_t = M(t) = \max_{0 \leq \tau \leq t} S(\tau)$. 
Usage

calculateProbabilityGeometricBrownianMotion(
    Type = c("P(S_t <= X)",
    "P(S_t >= X)",
    "P(S_t >= X, m_t >= B)",
    "P(M_t <= B)",
    "P(M_t >= B)",
    "P(m_t <= B)",
    "P(m_t >= B)"), S0 = 100, X, B, t = 1, mu = 0, sigma = 1)

Arguments

    Type            Type of probability to be calculated, see details.
    S0              Start price
    X               strike level
    B               barrier level
    t               time
    mu              drift term
    sigma           volatility in % p.a.

Details

Let \( M_t = \max(S_t) \) and \( m_t = \min(S_t) \) for \( t > 0 \) be the running maximum/minimum of the Geometric Brownian Motion \( S \) up to time \( t \) respectively.

- \( P(S_t \leq X) \) is the probability of the process being below \( X \) at time \( t \).
  Possible Application: shortfall risk of a plain-vanilla call option at maturity
- \( P(M_t \geq B) \) is the probability of the maximum exceeding a barrier level \( B \).
- \( P(M_t \leq B) \) is the probability of the maximum staying below a barrier level \( B \) up to time \( t \).
- \( P(m_t \leq B) \) is the probability of the minimum to fall below a barrier level \( B \).
- \( P(m_t \geq B) \) is the probability of the minimum to stay above barrier level \( B \).

Value

    a vector of probabilities

Author(s)

    Stefan Wilhelm <wilhelm@financial.com>

References

Distribution of the Brownian Bridge Minimum

See Also

calcBMProbability for probabilities of the standard Brownian Motion

Examples

```r
# Simulate paths for Geometric Brownian Motion and compute barrier probabilities
N=400
S <- matrix(NA, 1000, N+1)
for (i in 1:1000) {
  S[i,] <- GBM(S0=100, mu=0.05, sigma=1, T=1, N=N)
}

# a) Maximum M_t
M_t <- apply(S, 1, max, na.rm=TRUE)
S0 <- 100
B <- seq(100, 1000, by=1)
p1 <- calcGBMProbability(Type="P(M_t <= B)", S0=S0, B=B, t=1, mu=0.05, sigma=1)

# or via arithmetic Brownian Motion and drift mu - sigma^2/2
p2 <- calcBMProbability(Type="P(M_t <= a)", a=log(B/S0), t=1, mu=0.05-1/2, sigma=1)

plot(ecdf(M_t))
lines(B, p1, col="red", lwd=2)
lines(B, p2, col="green")

# b) Minimum m_t
m_t <- apply(S, 1, min, na.rm=TRUE)
B <- seq(0, 100, by=1)
p3 <- calcGBMProbability(Type="P(m_t <= B)", S0=S0, B=B, t=1, mu=0.05, sigma=1)
p4 <- calcBMProbability(Type="P(m_t <= a)", a=log(B/S0), t=1, mu=0.05-1/2, sigma=1)

plot(ecdf(m_t))
lines(B, p3, col="red", lwd=2)
lines(B, p4, col="green", lty=2)
```

Distribution of the Brownian Bridge Minimum

Distribution of the Minimum of a Brownian Bridge

Description

Density function and random generation of the minimum \( m_T = \min_{t_0 \leq t \leq T} \) of a Brownian Bridge \( B_t \) between time \( t_0 \) and \( T \).
Usage

\texttt{rBrownianBridgeMinimum(n = 100, t0 = 0, T = 1, a = 0, b = 0, sigma = 1)}
\texttt{dBrownianBridgeMinimum(x, t0 = 0, T = 1, a = 0, b = 0, sigma = 1)}

Arguments

- \texttt{n} the number of samples to draw
- \texttt{x} a vector of minimum values to calculate the density for
- \texttt{t0} start time
- \texttt{T} end time
- \texttt{a} start value of the Brownian Bridge (B(t0)=a)
- \texttt{b} end value of the Brownian Bridge (B(T)=b)
- \texttt{sigma} volatility p.a., e.g. 0.2 for 20%

Details

\texttt{rBrownianBridgeMinimum()} simulates the minimum \( m(T) \) for a Brownian Bridge \( B(t) \) between \( t_0 \leq t \leq T \), i.e. a Brownian Motion \( W(t) \) constraint to \( W(t_0) = a \) and \( W(T) = b \).

The simulation algorithm uses the conditional density \( f(m(T) = x|B(t_0) = a, B(T) = b) \) and is based on the exponential distribution given by Beskos et al. (2006), pp.1082–1083, which we generalized to the \( \sigma^2 \neq 1 \) case.

The joint density function \( m(T) \) and \( W(T) \) is (see Beskos2006, pp.1082–1083 and Karatzas2008, p.95):

\[
f_{m(T),W(T)}(b,a) = \frac{2 \cdot (a - 2b)}{\sqrt{2\pi} \sigma^3 \sqrt{T^3}} \cdot \exp \left\{ -\frac{(a - 2b)^2}{2\sigma^2 T} \right\}
\]

With the density of \( W(T) \)

\[
f_{W(T)}(a) = \frac{1}{\sqrt{2\pi} \sigma \sqrt{T}} \cdot \exp \left\{ -\frac{a^2}{2\sigma^2 T} \right\}
\]

it follows for the conditional density of the minimum \( m(T)|W(T) = a \)

\[
f_{m(T)|W(T)=a}(b) = \frac{2 \cdot (a - 2b)}{\sigma^2 T} \cdot \exp \left\{ -\frac{(a - 2b)^2}{2\sigma^2 T} + \frac{a^2}{2\sigma^2 T} \right\}
\]

Value

\texttt{simBrownianBridgeMinimum()} returns a vector of simulated minimum values of length \( n \).
\texttt{densityBrownianBridgeMinimum} returns a vector of length \( \text{length}(x) \) with density values

Author(s)

Stefan Wilhelm <wilhelm@financial.com>
References


Examples

```
# simulate 1000 samples from minimum distribution
m <- rBrownianBridgeMinimum(n = 1000, t0 = 0, T = 1, a = 0.2, b = 0, sigma = 2)

# and compare against the density
x <- seq(-6, 0, by=0.01)
dm <- dBrownianBridgeMinimum(x, t0 = 0, T = 1, a = 0.2, b = 0, sigma = 2)

plot(density(m))
lines(x, dm, lty=2, col="red")
```

Express Certificates Redemption Probabilities

Redemption Probabilities for Express Certificates

Description

Calculates the stop probabilities/early redemption probabilities for express certificates using the multivariate normal distribution or determines stop probabilities with Monte Carlo simulation.

Usage

```r
calcRedemptionProbabilities(S, X, T, r, r_d, sigma)
simRedemptionProbabilities(S, X, T, r, r_d, sigma, mc.steps=1000, mc.loops=20)
```

Arguments

- **S**: the asset price, a numeric value
- **X**: a vector of early exercise prices ("Bewertungsgrenzen"), vector of length \((n-1)\)
- **T**: a numeric vector of evaluation times measured in years ("Bewertungstage"): \(T = (t_1, ..., t_n)\)', vector of length \(n\)
- **r**: the annualized rate of interest, a numeric value; e.g. 0.25 means 25% pa.
- **r_d**: the annualized dividend yield, a numeric value; e.g. 0.25 means 25% pa.
- **sigma**: the annualized volatility of the underlying security, a numeric value; e.g. 0.3 means 30% volatility pa.
- **mc.steps**: Monte Carlo steps in one path
- **mc.loops**: Monte Carlo loops (iterations)
Express Certificates Redemption Probabilities

Details

Calculates the stop probabilities/early redemption probabilities for Express Certificates at valuation dates \((t_1, ..., t_n)'\) using the multivariate normal distribution of log returns of a Geometric Brownian Motion. The redemption probability \(p(t_i)\) at \(t_i < t_n\) is

\[
p(t_i) = P(S(t_i) \geq X(t_i), \forall j < i S(t_j) < X(t_j))
\]

i.e.

\[
p(t_i) = P(S(t_i) \geq X(t_i), S(t_1) \leq X(t_1), ..., S(t_{i-1}) \leq X(t_{i-1}))
\]

for \(i = 1, \ldots, (n - 1)\) and

\[
p(t_n) = P(S(t_1) \leq X(t_1), ..., S(t_{n-1}) \leq X(t_{n-1}))
\]

for \(i = n\).

Value

a vector of length \(n\) with the redemption probabilities at valuation dates \((t_1, ..., t_n)'\).

Author(s)

Stefan Wilhelm <wilhelm@financial.com>

References


Examples

```r
# Monte Carlo simulation of redemption probabilities
# p(t_i) = P(S(t_i) >= X(t_i), \forall j < i S(t_j) < X(t_j))
m.c. loops <- 5000
probs <- simRedemptionProbabilities(S=100, X=c(100,100,100), T=c(1,2,3,4),
  r=0.045, r_d=0, sigma=0.3, mc.steps=3000, mc.loops=5000)
table(probs$stops)/mc.loops

# Analytic calculation of redemption probabilities
probs2 <- calcRedemptionProbabilities(S=100, X=c(100,100,100), T=c(1,2,3,4),
  r=0.045, r_d=0, sigma=0.3)
probs2
```
ExpressCertificate.Classic

Analytical and numerical pricing of Classic Express Certificates

Description

Pricing of Classic Express Certificates using the truncated multivariate normal distribution (early stop probabilities) and numerical integration of the one-dimensional marginal return distribution at maturity

Usage

ExpressCertificate.Classic(S, X, T, K, g = function(S_T) {S_T},
   r, r_d, sigma, ratio = 1)

Arguments

S        the asset price, a numeric value
X        a vector of early exercise prices ("Bewertungsgrenzen"), vector of length (n-1)
T        a vector of evaluation times measured in years ("Bewertungstage"), vector of length n
K        vector of fixed early cash rebates in case of early exercise, length (n-1)
g        a payoff function at maturity, by default g(S_T)=S_T
r        the annualized rate of interest, a numeric value; e.g. 0.25 means 25% pa.
r_d      the annualized dividend yield, a numeric value; e.g. 0.25 means 25% pa.
sigma    the annualized volatility of the underlying security, a numeric value; e.g. 0.3 means 30% volatility pa.
ratio    ratio, number of underlyings one certificate refers to, a numeric value; e.g. 0.25 means 4 certificates refer to 1 share of the underlying asset

Details

The principal feature inherent to all express certificates is the callable feature with pretermined valuation dates \((t_1 < \ldots < t_n)\) prior to final maturity \(t_n\). Express certificates are typically called, if the underlying price on the valuation date is above a strike price (call level): \(S(t_i) > X(t_i)\).

The payoff of an express classic certificate at maturity is the underlying performance itself. So the payoff function at maturity takes the simple form of \(g(S(t_n)) = S(t_n)\).

We compute early redemption probabilities via the truncated multivariate normal distribution and integrate the one-dimensional marginal distribution for the expected payoff \(E[g(S(t_n))] = E[S(t_n)]\).

Value

a vector of length n with certificate prices
Author(s)
Stefan Wilhelm <wilhelm@financial.com>

References

See Also
MonteCarlo.ExpressCertificate.Classic and MonteCarlo.ExpressCertificate for Monte Carlo evaluation with similar payoff functions

Examples
ExpressCertificate.Classic(S=100, X=c(100),
T=c(1, 2), g = function(S) { S },
K=142.5, r=0.01, r_d=0, sigma=0.3, ratio = 1)

ExpressCertificate.Classic(S=100, X=c(100),
T=c(1, 2), g = function(S) { max(S, 151) },
K=142.5, r=0.01, r_d=0, sigma=0.3, ratio = 1)

GeometricBrownianMotion

Simulate paths from a Arithmetic or Geometric Brownian Motion

Description
Simulate one or more paths for an Arithmetic Brownian Motion \( B(t) \) or for a Geometric Brownian Motion \( S(t) \) for \( 0 \leq t \leq T \) using grid points (i.e. Euler scheme).

Usage
BM(S0, mu=0, sigma=1, T, N)
GBM(S0, mu, sigma, T, N)
GeometricBrownianMotionMatrix(S0, mu, sigma, T, mc.loops, N)

Arguments

S0 start value of the Arithmetic/Geometric Brownian Motion, i.e. \( S(0)=S0 \) or \( B(0) = S0 \)
mu the drift parameter of the Brownian Motion
sigma the annualized volatility of the underlying security, a numeric value; e.g. 0.3 means 30% volatility pa.
T time
mc.loops number of Monte Carlo price paths
N number of grid points in price path
Value

A vector of length \( N+1 \) with simulated asset prices at \((i \cdot T/N), i = 0, \ldots, N\).

Author(s)

Stefan Wilhelm <wilhelm@financial.com>

References


Examples

```r
# Simulate three trajectories of the Geometric Brownian Motion S(t)
T <- 1
mc.steps <- 100
dt <- T/mc.steps
t <- seq(0, T, by=dt)
S_t <- GBM(S0=100, mu=0.05, sigma=0.3, T=T, N=mc.steps)
plot(t, S_t, type="l", main="Sample paths of the Geometric Brownian Motion")
for (i in 1:2) {
  S_t <- GBM(S0=100, mu=0.05, sigma=0.3, T=T, N=mc.steps)
  lines(t, S_t, type="l")
}
```

---

getRedemptionTime | Redemption times

Description

Return redemption index

Usage

```r
getRedemptionTime(S, n, X)
getRedemptionTimesForMatrix(S, n, X)
```

Arguments

- **S** A \((n \times 1)\) vector of prices at valuation dates or a \((N \times n)\) matrix.
- **n** number of valuation dates; integer value.
- **X** A vector of call levels (length \((n - 1)\)).

Details

For a price vector of \( n \) prices at valuation dates \((S(t_1), \ldots, S(t_n))'\), determine the first redemption index \( i \) such as \( S(t_i) \geq X(t_i), \forall j \leq i, S(t_j) \leq X(t_j) (i = 1, \ldots, (n - 1) \) or \( i = n \) if \( S(t_1) \leq X(t_1), \ldots, S(t_{n-1}) \leq X(t_{n-1}) \)
Value

getRedemptionTime returns a scalar; getRedemptionTimesForMatrix returns a $N \times 1$ vector.

Author(s)

Stefan Wilhelm

See Also

calcRedemptionProbabilities and simRedemptionProbabilities

Examples

```r
S <- c(90, 95, 110, 120)
X <- c(100, 100, 100)
getRedemptionTime(S, n=4, X)
# 3
```

Monte Carlo valuation methods for Express Classic Certificates using the Euler scheme or sampling from conditional densities

Usage

```r
MonteCarlo.ExpressCertificate.Classic(S, X, T, K, r, r_d, sigma, ratio = 1, mc.steps = 1000, mc.loops = 20)
Conditional.MonteCarlo.ExpressCertificate.Classic(S, X, T, K, r, r_d, sigma, ratio = 1, mc.loops = 20, conditional.random.generator = "rnorm")
MonteCarlo.ExpressCertificate(S, X, T, K, B, r, r_d, sigma, mc.steps = 1000, mc.loops = 20, payoff.function)
```

Arguments

- `S` the asset price, a numeric value
- `X` a vector of early exercise prices ("Bewertungsgrenzen"), vector of length (n-1)
- `T` a vector of evaluation times measured in years ("Bewertungstage"), vector of length n
- `K` vector of fixed early cash rebates in case of early exercise, length (n-1)
- `B` barrier level
- `r` the annualized rate of interest, a numeric value; e.g. 0.25 means 25% pa.
- `r_d` the annualized dividend yield, a numeric value; e.g. 0.25 means 25% pa.
payoffExpress

sigma  the annualized volatility of the underlying security, a numeric value; e.g. 0.3 means 30% volatility pa.
ratio  ratio, number of underlyings one certificate refers to, a numeric value; e.g. 0.25 means 4 certificates refer to 1 share of the underlying asset
mc.steps  Monte Carlo steps in one path
mc.loops  Monte Carlo Loops (iterations)
conditional.random.generator  A pseudo-random or quasi-random (Halton-Sequence, Sobol-Sequence) generator for the conditional distributions, one of "rnorm", "rnorm.halton", "rnorm.sobol"
payoff.function  payoff function

Details

The conventional Monte Carlo uses the Euler scheme with mc.steps steps in order to approximate the continuous-time stochastic process.
The conditional Monte Carlo samples from conditional densities \( f(x_{i+1} | x_i) \) for \( i = 0, \ldots, (n-1) \), which are univariate normal distributions for the log returns of the Geometric Brownian Motion and Jump-diffusion model: \( f(x_1, x_2, \ldots, x_n) = f(x_n | x_{n-1}) \cdots f(x_2 | x_1) \cdot f(x_1 | x_0) \). The conditional Monte Carlo does not need the mc.steps points in between and has a much better performance.

Value

returns a list of
stops  stops
prices  vector of prices, length mc.loops
p  Monte Carlo estimate of the price = mean(prices)
S_T  vector of underlying prices at maturity

Author(s)

Stefan Wilhelm <wilhelm@financial.com>

payoffExpress  Defining payoff functions for Express Certificates

Description

Defining common or particular payoff functions for Express Certificates

Usage

payoffExpressClassic(i, n, S, m, K)
payoffExpressML0AN5(i, n, S, m, K, B, S0)
payoffExpressCappedBonusType1(i, n, S, m, K, B)
payoffExpressBonusType1(i, n, S, m, K, B)
Arguments
i  The redemption date (i = 1, ..., n)
n  The number of valuation dates
S  A vector of length n for the prices at the valuation dates, i.e. S(t_1), ..., S(t_n)
m  A vector of length n for the running minimum at the valuation dates, i.e. m(t_1), ..., m(t_n)
K  A vector of fixed cash rebates at early redemption times
B  A barrier level to be monitored
S0  underlying start price

Details
Payoff structure of express certificates can be either path independent or path dependent, while monitoring a barrier B.
Path independent payoffs:
The function `payoffExpressClassic` implements the following payoff at t_i:
\[
p(t_i) = K(t_i) \quad \text{for } i < n, \quad \text{else } S(t_n)
\]
Path dependent payoffs:
The function `payoffExpressCappedBonusType1` implements the following payoff:
\[
\begin{align*}
p(t_i) &= K(t_i) \quad \text{for } i < n \\
S(t_n) &\quad \text{for } i = n \text{ and } m(t_n) \leq B \\
K(t_n) &\quad \text{for } i = n \text{ and } m(t_n) > B
\end{align*}
\]
In case the barrier has not been hit during the lifetime, a fixed bonus payment K(t_n) is payed and the payoff is therefore capped.

The function `payoffExpressBonusType1` implements the following payoff:
\[
\begin{align*}
p(t_i) &= K(t_i) \quad \text{for } i < n \\
S(t_n) &\quad \text{for } i = n \text{ and } m(t_n) \leq B \\
\max (K(t_n), S(t_n)) &\quad \text{for } i = n \text{ and } m(t_n) > B
\end{align*}
\]
Unlike in the `payoffExpressCappedBonusType1`, this payoff is not capped for the case (S(t_n) > K(t_n))

The function `payoffExpressML0AN5` is an example of an quite complicated payoff including path dependence and coupon payments. See also the certificate prospectus `../inst/doc/ML0AN5.pdf`.

Value
returns the certificate payoff (Not discounted payoff!) for the given inputs at time i
print.express.certificate

Author(s)

Stefan Wilhelm <wilhelm@financial.com>

See Also

See also the generic pricing function SimulateGenericExpressCertificate

Description

Print method for express certificates objects

Usage

## S3 method for class 'express.certificate'
print(x, digits = max(3,getOption("digits") - 3), ...)

Arguments

x
An object of S3 class "express.certificate"

digits
Number of digits for printing the object "express.certificate" in method print.express.certificate

... further arguments passed to or from other methods

Details

The method print.express.certificate can be used for pretty printing of express certificates properties.

Author(s)

Stefan Wilhelm <wilhelm@financial.com>
Description

Simulates from the joint distribution of finite-dimensional distribution \((S(t_1), \ldots, S(t_n))\) and the minimum \(m(t_n)\) of a Geometric Brownian motion by either using simple grid approach or using the multivariate normal distribution of the returns and the conditional distribution of a minimum of a Brownian Bridge given the returns.

Usage

\[
\text{simPricesAndMinimumFromGBM}(N = 100, S, T, \mu, \sigma, \log = \text{FALSE}, m = \text{Inf})
\]

\[
\text{simPricesAndMinimumFromGBM2}(N = 10000, S, T, \mu, \sigma, \text{mc.steps} = 1000)
\]

Arguments

- \(N\) number of samples to draw
- \(S\) start value of the Arithmetic/Geometric Brownian Motion, i.e. \(S(0) = S0\) or \(B(0) = S0\)
- \(T\) Numeric vector of valuation times (length \(n\)). \(T = (t_1, \ldots, t_n)\)'
- \(\mu\) the drift parameter of the Geometric Brownian Motion
- \(\sigma\) volatility p.a., e.g. 0.2 for 20%
- \(\log\) logical, if true the returns instead of prices are returned
- \(m\) Possible prior minimum value.
- \(\text{mc.steps}\) Number of gridpoints

Details

grid-approach
The \text{simPricesAndMinimumFromGBM2} method uses the Monte Carlo Euler Scheme, the stepsize is \(\delta t = t_n/\text{mc.steps}\). The method is quite slow.

multivariate-normal distribution approach

The method \text{simPricesAndMinimumFromGBM} draws from the multivariate normal distribution of returns. For the \(n\) valuation times given by \(T = (t_1, \ldots, t_n)\)' we simulate from the joint distribution \((S(t_1), \ldots, S(t_n), m(t_1), \ldots, m(t_n))\) of the finite-dimensional distribution \((S(t_1), \ldots, S(t_n))\) and the running minimum \(m(t_i) = \min_{0 \leq t \leq t_i}(S(t))\) of a Geometric Brownian motion. This is done by using the multivariate normal distribution of the returns of a GBM and the conditional distribution of a minimum of a Brownian Bridge (i.e. in-between valuation dates).
First we simulate $(S(t_1), \ldots, S(t_n))$ from a multivariate normal distribution of the returns with mean vector 

$$(\mu - \sigma^2 / 2) T$$

and covariance matrix

$$(\Sigma)_{ij} = \min (t_i, t_j) \cdot \sigma^2$$

Next, we simulate the period minimum $m(t_{i-1}, t_i) = \min_{t_{i-1} \leq t \leq t_i} S(t)$ between two times $t_{i-1}$ and $t_i$ for all $i = 1, \ldots, n$. This minimum $m(t_{i-1}, t_i) | S(t_{i-1}), S(t_i)$ is the minimum of a Brownian Bridge between $t_{i-1}$ and $t_i$.

The global minimum is the minimum of all period minima given by $m(t_n) = \min(m(0,1), m(1,2), \ldots, m(n-1,n)) = \min m(t_{i-1}, t_i)$ for all $i = 1, \ldots, n$.

**Value**

A matrix $(N \times 2n)$ with rows $(S(t_1), \ldots, S(t_n), m(t_1), \ldots, m(t_n))$

**Note**

Since we are considering a specific path for the prices and are interested in the minimum given the specific trajectory (i.e. $m(t_n) | S(t_1), \ldots, S(t_n)$), it is not sufficient to sample from the bivariate density $(S(t_n), m(t_n))$, for which formulae is given by Karatzas/Shreve and others. Otherwise we could face the problem that some of the $S(t_1), \ldots, S(t_{n-1})$ are smaller than the simulated $m(t_n)$. However, both approaches yield the same marginal density for $m(t_n)$.

**Author(s)**

Stefan Wilhelm <wilhelm@financial.com>

**References**


**See Also**

The method `simPricesAndMinimumFromGBM` returns the same, but using the Euler Scheme.

See also `calcGBMProbability` for the CDF of the minimum $m_t$ (i.e. Type="P(m_t <= B)"

**Examples**

```r
# Comparison of sampling of GBM Minimum m_t via finite dimensional approach +
# Brownian Bridges vs. crude Monte Carlo

# naive grid-based approach
X0 <- simPricesAndMinimumFromGBM2(N=5000, S=100, T=c(1,2,3), mu = 0.05, sigma=0.3,
mc.steps=1000)

# Simulation of minimums m_t via prices at valuation dates
```
**simPricesAndMinimumFromTruncatedGBM**

Simulation of the joint finite-dimensional distribution of a restricted Geometric Brownian Motion and its minimum

**Description**

Simulates from the joint distribution of finite-dimensional distributions \((S(t_1), ..., S(t_n))\) and the minimum \(m(t_n)\) of a restricted Geometric Brownian motion by using the truncated multivariate normal distribution of the returns and the conditional distribution of a minimum of a Brownian Bridge given the returns.

**Usage**

```r
simPricesAndMinimumFromTruncatedGBM(N = 100, S, T, mu, sigma,
lowerX = rep(0, length(T)),
upperX = rep(+Inf, length(T)),
log = FALSE, m=Inf)
```

# (S(t_1), S(t_2), ..., S(t_n)) and Brownian Bridges in-between
X1 <- simPricesAndMinimumFromGBM(N=5000, S=100, T=c(1,2,3), mu=0.05, sigma=0.3)
m1 <- X1[,4]

# Monte Carlo simulation of m_t via gridpoints (m2)
mc.loops <- 5000
mc.steps <- 2000
S <- matrix(NA, mc.loops, mc.steps + 1)
for (j in 1:mc.loops) {
  S[j,] <- GBM(S0=100, mu=0.05, sigma=0.3, T=3, N=mc.steps)
}
m2 <- apply(S, 1, min) # minimum for each price path

# Compare probability density function and CDF for m_t against each other
# and against theoretical CDF.
par(mfrow=c(2,2))
# a) pdf of GBM minimum m_t at maturity for both approaches
plot(density(m1, to=100), col="black")
lines(density(m2, to=100), col="blue")

# b) compare empirical CDFs for m_t with theoretical probability \(P(m_t < B)\)
B <- seq(0, 100, by=1)
p3 <- calcGBMProbability(Type="P(m_t < B)",
  S0=100, B=B, t=3, mu=0.05, sigma=0.3)
plot(ecdf(m1), col="black", main="Sampling of GBM minimum m_t")
lines(ecdf(m2), col="blue")
lines(B, p3, col="red")
legend("topleft", legend=c("Finite-dimensions and Brownian Bridge",
  "MC Euler scheme", "Theoretical value"),
  col=c("black","blue","red"), lwd=2)
```
Arguments

- **N** number of samples to draw
- **S** start value of the Arithmetic/Geometric Brownian Motion, i.e. \( S(0) = S_0 \) or \( B(0) = S_0 \)
- **T** Numeric vector of \( n \) valuation times \( T = (t_1, \ldots, t_n)' \)
- **mu** the drift parameter of the Geometric Brownian Motion
- **sigma** volatility p.a., e.g. 0.2 for 20%
- **lowerX** Numeric vector of \( n \) lower bounds for the Geometric Brownian Motion, zeros are permitted, default is `rep(0,length(T))`
- **upperX** Numeric vector of \( n \) upper bounds for the Geometric Brownian Motion, \( K_{\infty} \) are permitted, default is `rep(K_{\infty},length(T))`
- **log** logical, if true the returns instead of prices are returned
- **m** Possible prior minimum value.

Details

For the \( n \) valuation times given by \( T = (t_1, \ldots, t_n)' \) we simulate from the joint distribution \((S(t_1), \ldots, S(t_n), m(t_1), \ldots, m(t_n))\) of the finite-dimensional distribution \((S(t_1), \ldots, S(t_n))\) and the running minimum \( m(t_i) = \min_{0 \leq s \leq t_i} S_i \) of a restricted/truncated Geometric Brownian motion.

The Geometric Brownian Motion is conditioned at the \( n \) valuation dates \((t_1, \ldots, t_n)\) on \( lowerX_i \leq S(t_i) \leq upperX_i \) for all \( i = 1, \ldots, n \).

First we simulate \((S(t_1), \ldots, S(t_n))\) from a truncated multivariate normal distribution of the returns with mean vector \((\mu - \sigma^2 / 2) * T\)

and covariance matrix

\[
\Sigma = \begin{bmatrix}
\min (t_1, t_1) \sigma^2 & \min (t_1, t_2) \sigma^2 & \cdots & \min (t_1, t_n) \sigma^2 \\
\min (t_2, t_1) \sigma^2 & \min (t_2, t_2) \sigma^2 & \cdots & \min (t_2, t_n) \sigma^2 \\
\vdots & \vdots & \ddots & \vdots \\
\min (t_n, t_1) \sigma^2 & \cdots & \min (t_n, t_1) \sigma^2 & \min (t_n, t_2) \sigma^2
\end{bmatrix}
\]

and lower and upper truncation points `lower=log(lowerX/S)` and `upper=log(upperX/S)` respectively.

Given the realized prices \((S(t_1), \ldots, S(t_n))\) we simulate the global minimum as the minimum of several Brownian Bridges as described in Beskos (2006):

We simulate the period minimum \( m_{(i-1,i)} \) between two times \( t_{i-1} \) and \( t_i \) for all \( i = 1, \ldots, n \). This minimum \( m_{(i-1,i)} \mid S(t_{i-1}), S(t_i) \) is the minimum of a Brownian Bridge between \( t_{i-1} \) and \( t_i \).

The global minimum is the minimum of all period minima given by

\[
m_n = \min(m_{(0,1)}, m_{(1,2)}, \ldots, m_{(n-1,n)}) = \min(m_{(i-1,i)}) \text{ for all } i = 1, \ldots, n.
\]

Value

A \((N \times 2 * n)\) matrix with \( N \) rows and columns \((S(t_1), \ldots, S(t_n), m(t_1), \ldots, m(t_n))\)
Note

This function can be used to determine the barrier risk of express certificates at maturity, i.e. the probability that barrier $B$ has been breached given that we reach maturity: $P(m(t_n) \leq B \mid \forall i < n, S(t_i) < X(t_i))$

Author(s)

Stefan Wilhelm <wilhelm@financial.com>

See Also

See the similar method `simPricesAndMinimumFromGBM` for the unrestricted Geometric Brownian Motion (i.e. lowerX=rep(0,n) and upperX=rep(Inf,n)).

Examples

```r
# 1. Simulation of restricted GBM prices and minimums m_t
# finite-dimensional distribution and Brownian Bridge
X1 <- simPricesAndMinimumFromTruncatedGBM(N=5000, S=100, T=c(1,2,3),
  upperX=c(100,100,Inf), mu=0.05, sigma=0.3)
ml <- X1[,4]

# 2. Compare to distribution of unrestricted GBM minimums
X2 <- simPricesAndMinimumFromGBM(N=5000, S=100, T=c(1,2,3),
  mu=0.05, sigma=0.3)
m2 <- X2[,4]

plot(density(ml, to=100), col="black", main="Minimum m_t for Express Certificate
price paths at maturity")
lines(density(m2, to=100), col="blue")
legend("topleft", legend=c("Restricted GBM minimum","Unrestricted GBM minimum"),
  col=c("black","blue"), lty=1, bty="n")
```

SimulateExpressCertificate

*Monte Carlo Valuation of Express Certificates*

Description

Generic Monte Carlo Valuation of Express Certificates using the Euler scheme, multivariate normal distribution and truncated multivariate normal.

Usage

```r
SimulateGenericExpressCertificate(S, X, K, T, r, r_d, sigma, mc.loops = 10000,
  mc.steps = 1000, payoffFunction = payoffExpressClassic, ...)
SimulateExpressClassicCertificate(S, X, K, T, r, r_d, sigma, mc.loops = 10000,
  mc.steps = 1000)
```
SimulateExpressCertificate(S, X, B, K, T, r, r_d, sigma, mc.loops = 10000, mc.steps = 1000, barrierHit = FALSE)

simExpressPriceMVN(S, m = Inf, X, K, B, T, r, r_d, sigma, mc.loops = 100000, payoffFunction, ...)
simExpressPriceTMVN(S, m = Inf, X, K, B, T, r, r_d, sigma, mc.loops = 100000, payoffFunction, ...)

Arguments

S  the asset price, a numeric value
X a vector of early exercise prices/call levels ("Bewertungsgrenzen"), vector of length \(n-1\)
B  barrier level
K vector of fixed early cash rebates in case of early exercise, length \(n-1\) or \(n\) in case of a fixed rebate at maturity
T  a vector of evaluation times measured in years ("Bewertungstage"), vector of length \(n\)
r  the annualized rate of interest, a numeric value; e.g. 0.05 means 5% pa.
r_d  the annualized dividend yield, a numeric value; e.g. 0.25 means 25% pa.
sigma  the annualized volatility of the underlying security, a numeric value; e.g. 0.3 means 30% volatility pa.
mc.loops  Monte Carlo Loops (iterations)
mc.steps  Monte Carlo steps in one path
barrierHit  flag whether the barrier has already been reached/hit during the lifetime
payoffFunction  definition of a payoff function, see details below
m  The minimum price up to today for pricing during the lifetime.
...  Additional parameters passed to the payoff function

Details

TO BE DONE: Definition of payoff functions

Value

The methods return an object of class "express.certificate". An object of class "express.certificate" is a list containing at least the following components:

price  Monte Carlo estimate
prices  A vector of simulated discounted prices (length \(mc.loops\))
n  The number of valuation dates
redemptionTimes  A vector of redemption times \(i = 1..n\) (length \(mc.loops\))
S  the asset price, a numeric value
SimulateExpressCertificate

X early exercise prices/call levels
K vector of fixed early cash rebates in case of early exercise
T a vector of evaluation times measured in years ("Bewertungstage")

There is also a method `print.express.certificate` for pretty printing of `express.certificate` objects.

Author(s)

Stefan Wilhelm <wilhelm@financial.com>

See Also

Definition of several payoff functions in `payoffExpressClassic`, `payoffExpressCappedBonusType1` or `payoffExpressBonusType1`

`print.express.certificate` for pretty printing of `express.certificate` objects

Examples

```r
## Not run:
# Example CB7AXR on Deutsche Telekom on 10.12.2009
p <- SimulateExpressBonusCertificate(S=10.4/12.10*100, X=c(100,100,100), B=7/12.1*100,
   K=c(134, 142.5, 151),
   T=RLZ(c("16.12.2009","17.06.2010","17.12.2010"), start="10.12.2009"), r=0.01, r_d=0,
   sigma=0.23, mc.loops=10000, mc.steps=1000)

## End(Not run)
```
Index

*Topic **math**
calcBMPProbability, 2
calcGBMProbability, 4
Distribution of the Brownian Bridge Minimum, 6
Express Certificates Redemption Probabilities, 8
ExpressCertificate.Classic, 10
GeometricBrownianMotion, 11
MonteCarlo.ExpressCertificate.Classic, 13
simPricesAndMinimumFromTruncatedGBM, 19

*Topic **multivariate**
simPricesAndMinimumFromGBM, 17
simPricesAndMinimumFromTruncatedGBM, 19

ArtihmeticBrownianMotion
(GeometricBrownianMotion), 11

BM (GeometricBrownianMotion), 11
BrownianMotion
(GeometricBrownianMotion), 11
calcBMPProbability, 2, 6
calcBrownianMotionProbability
(calcBMProbability), 2
calcGBMProbability, 4, 18
calcRedemptionProbabilities, 13
calcRedemptionProbabilities (Express Certificates Redemption Probabilities), 8
calcStopProbabilities (Express Certificates Redemption Probabilities), 8
calculateProbabilityBrownianMotion
(calcBMProbability), 2
calculateProbabilityGeometricBrownianMotion
(calcGBMProbability), 4

(MonteCarlo.ExpressCertificate.Classic), 13
dBrownianBridgeMinimum (Distribution of the Brownian Bridge Minimum), 6
Distribution of the Brownian Bridge Minimum, 6
Express Certificates Redemption Probabilities, 8
ExpressCertificate.Classic, 10
GBM (GeometricBrownianMotion), 11
GeometricBrownianMotion, 11
GeometricBrownianMotionMatrix
(GeometricBrownianMotion), 11
getRedemptionTime, 12
getRedemptionTimesForMatrix
(getRedemptionTime), 12
MonteCarlo.ExpressCertificate.Classic, 11
MonteCarlo.ExpressCertificate
(MonteCarlo.ExpressCertificate.Classic), 13
MonteCarlo.ExpressCertificate.Classic, 11, 13
MonteCarloStopProbabilities (Express Certificates Redemption Probabilities), 8
payoffExpress, 14
payoffExpressBonusType1, 23
payoffExpressBonusType1
(payoffExpress), 14
payoffExpressCappedBonusType1, 23
payoffExpressCappedBonusType1
(payoffExpress), 14
payoffExpressClassic, 23
payoffExpressClassic(payoffExpress), 14
INDEX

payoffExpressML0AN5 (payoffExpress), 14
print.express.certificate, 16, 23

rBrownianBridgeMinimum (Distribution of the Brownian Bridge Minimum), 6

simExpressPriceMVN (SimulateExpressCertificate), 21
simExpressPriceTMVN (SimulateExpressCertificate), 21
simPricesAndMinimumFromGBM, 17, 21
simPricesAndMinimumFromGBM2, 18
simPricesAndMinimumFromGBM2 (simPricesAndMinimumFromGBM), 17
simPricesAndMinimumFromTruncatedGBM, 19
simRedemptionProbabilities, 13
simRedemptionProbabilities (Express Certificates Redemption Probabilities), 8
SimulateExpressBonusCertificate (SimulateExpressCertificate), 21
SimulateExpressCertificate, 21
SimulateExpressClassicCertificate (SimulateExpressCertificate), 21
SimulateGenericExpressCertificate, 16
SimulateGenericExpressCertificate (SimulateExpressCertificate), 21