Package ‘Sim.DiffProc’

February 12, 2011

Type Package

Title Simulation of Diffusion Processes

Version 2.0

Date 2011-02-09

Author Prof. BOUKHETALA Kamal <kboukhetala@usthb.dz>, Mr. GUIDOUM Arsalane <starsalane@gmail.com>

Maintainer BOUKHETALA Kamal <kboukhetala@usthb.dz>

Depends R (>= 2.11.0), tcltk, tcltk2, stats4, rgl

Description Simulation of diffusion processes and numerical solution of stochastic differential equations. Analysis of discrete-time approximations for stochastic differential equations (SDE) driven by Wiener processes,in financial and actuarial modeling and other areas of application for example modelling and simulation of dispersion in shallow water using the attractive center (K.BOUKHETALA,1996). Simulation and statistical analysis of the first passage time (FPT) and M-samples of the random variable X(v) given by a simulated diffusion process.

License GPL (>= 2)

URL http://www.r-project.org

Repository CRAN

LazyLoad yes

R topics documented:

`Sim.DiffProc-package` *Simulation of Diffusion Processes.*

Description

Simulation of diffusion processes and numerical solution of stochastic differential equations. Analysis of discrete-time approximations for stochastic differential equations (SDE) driven by Wiener processes,in financial and actuarial modeling and other areas of application for example modelling and simulation of dispersion in shallow water using the attractive center (K.BOUKHETALA,1996).

Simulation and statistical analysis of the first passage time (FPT) and M-samples of the random variable X(v) given by a simulated diffusion process.

Details

Package:	Sim.DiffProc
Type:	Package
Version:	2.0
Date:	2011-02-09
License:	GPL (>= 2)
LazyLoad:	yes

Author(s)

BOUKHETALA Kamal <kboukhetala@usthb.dz>, GUIDOUM Arsalane <starsalane@gmail.com>.

Maintainer: BOUKHETALA Kamal <kboukhetala@usthb.dz>

References

1. Franck Jedrzejewski. Modeles aleatoires et physique probabiliste, Springer, 2009.
2. K.Boukhetala, Estimation of the first passage time distribution for a simulated diffusion process, Maghreb Math.Rev, Vol.7, No 1, Jun 1998, pp. 1-25.
3. K.Boukhetala, Simulation study of a dispersion about an attractive centre. In proceedings of 11th Symposium Computational Statistics, edited by R.Dutter and W.Grossman, Wien , Austria, 1994, pp. 128-130.
4. K.Boukhetala,Modelling and simulation of a dispersion pollutant with attractive centre, Edited by Computational Mechanics Publications, Southampton ,U.K and Computational Mechanics Inc, Boston, USA, pp. 245-252.
5. K.Boukhetala, Kernel density of the exit time in a simulated diffusion, les Annales Maghrebines De L ingenieur, Vol , 12, N Hors Serie. Novembre 1998, Tome II, pp 587-589.
6. T. Rolski, H. Schmidli, V. Schmidt and J. Teugels, Stochastic Processes for Insurance and Finance,John Wiley & Sons, 1998.
7. Fima C Klebaner. Introduction to stochastic calculus with application (Second Edition), Imperial College Press (ICP), 2005.

8. LAWRENCE C.EVANS. An introduction to stochastic differential equations (Version 1.2), Department of Mathematics (UC BERKELEY).
9. Hui-Hsiung Kuo. Introduction to stochastic integration, Springer, 2006.
10. E.Allen. Modeling with Ito stochastic differential equations, Springer, 2007.
11. Peter E.Kloeden, Eckhard Platen, Numerical solution of stochastic differential equations, Springer, 1995.
12. Douglas Henderson, Peter Plaschko, Stochastic differential equations in science and engineering,World Scientific, 2006.
13. A.Greiner, W.Strittmatter,and J.Honerkamp, Numerical Integration of Stochastic Differential Equations, Journal of Statistical Physics, Vol. 51, Nos. 1/2, 1988.
14. YOSHIHIRO SAITO, TAKETOMO MITSUI, SIMULATION OF STOCHASTIC DIFFERENTIAL EQUATIONS, Ann.Inst.Statist.Math, Vol. 45, No.3,419-432 (1993).
15. FRANCOIS-ERIC RACICOT, RAYMOND THEORET, Finance computationnelle et gestion des risques, Presses de universite du Quebec, 2006.
16. Avner Friedman, Stochastic differential equations and applications, Volume 1, ACADEMIC PRESS, 1975.

Examples

```
demo(Sim.DiffProc)
```

`ABM` *Creating Arithmetic Brownian Motion Model*

Description

Simulation of the arithmetic brownian motion model.

Usage

```
ABM(N, t0, T, x0, theta, sigma, output = FALSE)
```

Arguments

`N`	size of process.
`t0`	initial time.
`T`	final time.
`x0`	initial value of the process at time `t0`.
`theta`	constant (`Coefficient of drift`).
`sigma`	constant positive (`Coefficient of diffusion`).
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

The function `ABM` returns a trajectory of the Arithmetic Brownian motion starting at `x0` at time `t0`, than the Discretization `dt = (T-t0)/N`.

The stochastic differential equation of the Arithmetic Brownian motion is :

$$dX(t) = theta * dt + sigma * dW(t)$$

with `theta :drift coefficient` and `sigma :diffusion coefficient`,`W(t)` is Wiener process.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`ABMF` creating flow of the arithmetic brownian motion model.

Examples

```
## Arithmetic Brownian Motion Model
## dX(t) = 3 * dt + 2 * dW(t) ; x0 = 0 and t0 = 0
ABM(N=1000,t0=0,T=1,x0=0,theta=3,sigma=2)
```

`ABMF` *Creating Flow of The Arithmetic Brownian Motion Model*

Description

Simulation flow of the arithmetic brownian motion model.

Usage

```
ABMF(N, M, t0, T, x0, theta, sigma, output = FALSE)
```

Arguments

`N`	size of process.
`M`	number of trajectories.
`t0`	initial time.
`T`	final time.
`x0`	initial value of the process at time `t0`.
`theta`	constant (`Coefficient of drift`).
`sigma`	constant positive (`Coefficient of diffusion`).
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

The function `ABMF` returns a flow of the Arithmetic Brownian motion starting at `x0` at time `t0`, than the discretization `dt = (T-t0)/N`.

The stochastic differential equation of the Arithmetic Brownian motion is :

$$dX(t) = theta * dt + sigma * dW(t)$$

With `theta :drift coefficient` and `sigma :diffusion coefficient`, `W(t)` is Wiener process.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`ABM` creating the arithmetic brownian motion model.

Examples

```
## Flow of Arithmetic Brownian Motion Model
## dX(t) = 3 * dt + 2 * dW(t) ; x0 = 0 and t0 = 0
ABMF(N=1000,M=5,t0=0,T=1,x0=0,theta=3,sigma=2)
```

`Ajdbeta` *Adjustment By Beta Distribution*

Description

Adjusted your sample by the beta law, estimated these parameters using the method of maximum likelihood, and calculating the Akaike information criterion for one or several fitted model objects for which a log-likelihood value can be obtained, according to the formula -2*log-likelihood + k*npar, where npar represents the number of parameters in the fitted model, and k = 2 for the usual AIC, and computes confidence intervals for one or more parameters in a fitted model (Law).

Usage

```
Ajdbeta(X, starts = list(shape1 = 1, shape2 = 1), leve = 0.95)
```

Arguments

`X`	a numeric vector of the observed values.
`starts`	named list. Initial values for optimizer.
`leve`	the confidence level required.

Details

The optim optimizer is used to find the minimum of the negative log-likelihood. An approximate covariance matrix for the parameters is obtained by inverting the Hessian matrix at the optimum.

For more detail consulted mle,confint,AIC.

R has the [dqpr]beta functions to evaluate the density, the quantiles, and the cumulative distribution or generate pseudo random numbers from the beta distribution.

Value

coef	Coefficients extracted from the model.
AIC	A numeric value with the corresponding AIC.
vcov	A matrix of the estimated covariances between the parameter estimates in the linear or non-linear predictor of the model.
confint	A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labelled as (1-level)/2 and 1 - (1-level)/2.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

Ajdexp Adjustment By Exponential Distribution,Ajdf Adjustment By F Distribution, Ajdgamma Adjustment By Gamma Distribution,Ajdlognorm Adjustment By Log Normal Distribution, Ajdnorm Adjustment By Normal Distribution,Ajdt Adjustment By Student t Distribution, Ajdweibull Adjustment By Weibull Distribution,Ajdchisq Adjustment By Chi-Squared Distribution.

Examples

```
## X <- rbeta(1000,shape1 = 1, shape2 = 3)
## Ajdbeta(X, starts = list(shape1 = 1, shape2 = 1), leve = 0.95)
```

Ajdchisq *Adjustment By Chi-Squared Distribution*

Description

Adjusted your sample by the chi-squared law, estimated these parameters using the method of maximum likelihood, and calculating the Akaike information criterion for one or several fitted model objects for which a log-likelihood value can be obtained, according to the formula -2*log-likelihood + k*npar, where npar represents the number of parameters in the fitted model, and k = 2 for the usual AIC, and computes confidence intervals for one or more parameters in a fitted model (Law).

Usage

```
Ajdchisq(X, starts = list(df = 1), leve = 0.95)
```

Arguments

X	a numeric vector of the observed values.
starts	named list. Initial values for optimizer.
leve	the confidence level required.

Details

The optim optimizer is used to find the minimum of the negative log-likelihood. An approximate covariance matrix for the parameters is obtained by inverting the Hessian matrix at the optimum.

For more detail consulted mle,confint,AIC.

R has the [dqpr]chisq functions to evaluate the density, the quantiles, and the cumulative distribution or generate pseudo random numbers from the chi-squared distribution.

Value

coef	Coefficients extracted from the model.
AIC	A numeric value with the corresponding AIC.
vcov	A matrix of the estimated covariances between the parameter estimates in the linear or non-linear predictor of the model.
confint	A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labelled as (1-level)/2 and 1 - (1-level)/2.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

Ajdexp Adjustment By Exponential Distribution,Ajdf Adjustment By F Distribution, Ajdgamma Adjustment By Gamma Distribution,Ajdlognorm Adjustment By Log Normal Distribution, Ajdnorm Adjustment By Normal Distribution,Ajdt Adjustment By Student t Distribution, Ajdweibull Adjustment By Weibull Distribution,Ajdbeta Adjustment By Beta Distribution.

Examples

```
X <- rchisq(1000,df = 20)
Ajdchisq(X, starts = list(df = 1), leve = 0.95)
```

Ajdexp *Adjustment By Exponential Distribution*

Description

Adjusted your sample by the exponential law, estimated these parameters using the method of maximum likelihood, and calculating the Akaike information criterion for one or several fitted model objects for which a log-likelihood value can be obtained, according to the formula -2*log-likelihood + k*npar, where npar represents the number of parameters in the fitted model, and k = 2 for the usual AIC, and computes confidence intervals for one or more parameters in a fitted model (Law).

Usage

```
Ajdexp(X, starts = list(lambda = 1), leve = 0.95)
```

Arguments

X	a numeric vector of the observed values.
starts	named list. Initial values for optimizer.
leve	the confidence level required.

Details

The `optim` optimizer is used to find the minimum of the negative log-likelihood. An approximate covariance matrix for the parameters is obtained by inverting the Hessian matrix at the optimum.

For more detail consulted `mle`,`confint`,`AIC`.

`R` has the `[dqpr]exp` functions to evaluate the density, the quantiles, and the cumulative distribution or generate pseudo random numbers from the exponential distribution.

Value

coef	Coefficients extracted from the model.
AIC	A numeric value with the corresponding AIC.
vcov	A matrix of the estimated covariances between the parameter estimates in the linear or non-linear predictor of the model.
confint	A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labelled as (1-level)/2 and 1 - (1-level)/2.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`Ajdchisq` Adjustment By Chi-Squared Distribution,`Ajdf` Adjustment By F Distribution, `Ajdgamma` Adjustment By Gamma Distribution,`Ajdlognorm` Adjustment By Log Normal Distribution, `Ajdnorm` Adjustment By Normal Distribution,`Ajdt` Adjustment By Student t Distribution, `Ajdweibull` Adjustment By Weibull Distribution,`Ajdbeta` Adjustment By Beta Distribution.

Examples

```
X <- rexp(100,15)
Ajdexp(X, starts = list(lambda = 1), leve = 0.95)
```

`Ajdf` *Adjustment By F Distribution*

Description

Adjusted your sample by the F law, estimated these parameters using the method of maximum likelihood, and calculating the Akaike information criterion for one or several fitted model objects for which a log-likelihood value can be obtained, according to the formula -2*log-likelihood + k*npar, where npar represents the number of parameters in the fitted model, and k = 2 for the usual AIC, and computes confidence intervals for one or more parameters in a fitted model (Law).

Usage

```
Ajdf(X, starts = list(df1 = 1, df2 = 1), leve = 0.95)
```

Arguments

`X`	a numeric vector of the observed values.
`starts`	named list. Initial values for optimizer.
`leve`	the confidence level required.

Details

The `optim` optimizer is used to find the minimum of the negative log-likelihood. An approximate covariance matrix for the parameters is obtained by inverting the Hessian matrix at the optimum.

For more detail consulted `mle`,`confint`,`AIC`.

`R` has the `[dqpr]f` functions to evaluate the density, the quantiles, and the cumulative distribution or generate pseudo random numbers from the F distribution.

Value

`coef`	Coefficients extracted from the model.
`AIC`	A numeric value with the corresponding AIC.
`vcov`	A matrix of the estimated covariances between the parameter estimates in the linear or non-linear predictor of the model.
`confint`	A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labelled as (1-level)/2 and 1 - (1-level)/2.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`Ajdchisq` Adjustment By Chi-Squared Distribution,`Ajdexp` Adjustment By Exponential Distribution, `Ajdgamma` Adjustment By Gamma Distribution,`Ajdlognorm` Adjustment By Log Normal Distribution, `Ajdnorm` Adjustment By Normal Distribution,`Ajdt` Adjustment By Student t Distribution, `Ajdweibull` Adjustment By Weibull Distribution,`Ajdbeta` Adjustment By Beta Distribution.

Examples

```
X <- rf(100,df1=5,df2=5)
Ajdf(X, starts = list(df1 = 1, df2 = 1), leve = 0.95)
```

`Ajdgamma` *Adjustment By Gamma Distribution*

Description

Adjusted your sample by the gamma law, estimated these parameters using the method of maximum likelihood, and calculating the Akaike information criterion for one or several fitted model objects for which a log-likelihood value can be obtained, according to the formula -2*log-likelihood + k*npar, where npar represents the number of parameters in the fitted model, and k = 2 for the usual AIC, and computes confidence intervals for one or more parameters in a fitted model (Law).

Usage

```
Ajdgamma(X, starts = list(shape = 1, rate = 1), leve = 0.95)
```

Arguments

`X`	a numeric vector of the observed values.
`starts`	named list. Initial values for optimizer.
`leve`	the confidence level required.

Details

The `optim` optimizer is used to find the minimum of the negative log-likelihood. An approximate covariance matrix for the parameters is obtained by inverting the Hessian matrix at the optimum.

For more detail consulted `mle`,`confint`,`AIC`.

`R` has the `[dqpr]gamma` functions to evaluate the density, the quantiles, and the cumulative distribution or generate pseudo random numbers from the gamma distribution.

Value

`coef`	Coefficients extracted from the model.
`AIC`	A numeric value with the corresponding AIC.
`vcov`	A matrix of the estimated covariances between the parameter estimates in the linear or non-linear predictor of the model.
`confint`	A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labelled as (1-level)/2 and 1 - (1-level)/2.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`Ajdchisq` Adjustment By Chi-Squared Distribution,`Ajdexp` Adjustment By Exponential Distribution, `Ajdf` Adjustment By F Distribution,`Ajdlognorm` Adjustment By Log Normal Distribution, `Ajdnorm` Adjustment By Normal Distribution,`Ajdt` Adjustment By Student t Distribution, `Ajdweibull` Adjustment By Weibull Distribution,`Ajdbeta` Adjustment By Beta Distribution.

Examples

```
##X <- rgamma(100,shape=1,rate=0.5)
## gamma(1,0.5)~~ exp(0.5)~~ weibull(1,2)
##Ajdgamma(X, starts = list(shape = 1, rate = 1), leve = 0.95)
##Ajdexp(X)
##Ajdweibull(X)
```

`Ajdlognorm` *Adjustment By Log Normal Distribution*

Description

Adjusted your sample by the log normal law, estimated these parameters using the method of maximum likelihood, and calculating the Akaike information criterion for one or several fitted model objects for which a log-likelihood value can be obtained, according to the formula -2*log-likelihood + k*npar, where npar represents the number of parameters in the fitted model, and k = 2 for the usual AIC, and computes confidence intervals for one or more parameters in a fitted model (Law).

Usage

```
Ajdlognorm(X, starts = list(meanlog = 1, sdlog = 1), leve = 0.95)
```

Arguments

`X`	a numeric vector of the observed values.
`starts`	named list. Initial values for optimizer.
`leve`	the confidence level required.

Details

The `optim` optimizer is used to find the minimum of the negative log-likelihood. An approximate covariance matrix for the parameters is obtained by inverting the Hessian matrix at the optimum.

For more detail consulted `mle`,`confint`,`AIC`.

`R` has the `[dqpr]lnorm` functions to evaluate the density, the quantiles, and the cumulative distribution or generate pseudo random numbers from the log normal distribution.

Value

`coef`	Coefficients extracted from the model.
`AIC`	A numeric value with the corresponding AIC.
`vcov`	A matrix of the estimated covariances between the parameter estimates in the linear or non-linear predictor of the model.
`confint`	A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labelled as (1-level)/2 and 1 - (1-level)/2.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

Ajdchisq Adjustment By Chi-Squared Distribution,Ajdexp Adjustment By Exponential Distribution, Ajdf Adjustment By F Distribution,Ajdgamma Adjustment By Gamma Distribution, Ajdnorm Adjustment By Normal Distribution,Ajdt Adjustment By Student t Distribution, Ajdweibull Adjustment By Weibull Distribution,Ajdbeta Adjustment By Beta Distribution.

Examples

```
##X <- rlnorm(1000,3,1)
##Ajdlognorm(X, starts = list(meanlog = 1, sdlog = 1), leve = 0.95)
```

Ajdnorm *Adjustment By Normal Distribution*

Description

Adjusted your sample by the normal law, estimated these parameters using the method of maximum likelihood, and calculating the Akaike information criterion for one or several fitted model objects for which a log-likelihood value can be obtained, according to the formula -2*log-likelihood + k*npar, where npar represents the number of parameters in the fitted model, and k = 2 for the usual AIC, and computes confidence intervals for one or more parameters in a fitted model (Law).

Usage

```
Ajdnorm(X, starts = list(mean = 1, sd = 1), leve = 0.95)
```

Arguments

X	a numeric vector of the observed values.
starts	named list. Initial values for optimizer.
leve	the confidence level required.

Details

The `optim` optimizer is used to find the minimum of the negative log-likelihood. An approximate covariance matrix for the parameters is obtained by inverting the Hessian matrix at the optimum.

For more detail consulted mle,confint,AIC.

`R` has the `[dqpr]norm` functions to evaluate the density, the quantiles, and the cumulative distribution or generate pseudo random numbers from the normal distribution.

Value

coef	Coefficients extracted from the model.
AIC	A numeric value with the corresponding AIC.
vcov	A matrix of the estimated covariances between the parameter estimates in the linear or non-linear predictor of the model.
confint	A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labelled as (1-level)/2 and 1 - (1-level)/2.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

Ajdchisq Adjustment By Chi-Squared Distribution,Ajdexp Adjustment By Exponential Distribution, Ajdf Adjustment By F Distribution,AjdgammaAdjustment By Gamma Distribution, Ajdlognorm Adjustment By Log Normal Distribution,Ajdt Adjustment By Student t Distribution, Ajdweibull Adjustment By Weibull Distribution,Ajdbeta Adjustment By Beta Distribution.

Examples

```
##X <- rnorm(1000,4,0.5)
##Ajdnorm(X, starts = list(mean = 1, sd = 1), leve = 0.95)
```

Ajdt *Adjustment By Student t Distribution*

Description

Adjusted your sample by the student t law, estimated these parameters using the method of maximum likelihood, and calculating the Akaike information criterion for one or several fitted model objects for which a log-likelihood value can be obtained, according to the formula -2*log-likelihood + k*npar, where npar represents the number of parameters in the fitted model, and k = 2 for the usual AIC, and computes confidence intervals for one or more parameters in a fitted model (Law).

Usage

```
Ajdt(X, starts = list(df = 1), leve = 0.95)
```

Arguments

X	a numeric vector of the observed values.
starts	named list. Initial values for optimizer.
leve	the confidence level required.

Details

The optim optimizer is used to find the minimum of the negative log-likelihood. An approximate covariance matrix for the parameters is obtained by inverting the Hessian matrix at the optimum.

For more detail consulted mle,confint,AIC.

R has the [dqpr]t functions to evaluate the density, the quantiles, and the cumulative distribution or generate pseudo random numbers from the student t distribution.

Value

`coef`	Coefficients extracted from the model.
`AIC`	A numeric value with the corresponding AIC.
`vcov`	A matrix of the estimated covariances between the parameter estimates in the linear or non-linear predictor of the model.
`confint`	A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labelled as (1-level)/2 and 1 - (1-level)/2.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`Ajdchisq` Adjustment By Chi-Squared Distribution,`Ajdexp` Adjustment By Exponential Distribution, `Ajdf` Adjustment By F Distribution,`Ajdgamma` Adjustment By Gamma Distribution, `Ajdlognorm` Adjustment By Log Normal Distribution,`Ajdnorm` Adjustment By Normal Distribution, `Ajdweibull` Adjustment By Weibull Distribution,`Ajdbeta` Adjustment By Beta Distribution.

Examples

```
X <- rt(1000,df=2)
Ajdt(X, starts = list(df = 1), leve = 0.95)
```

`Ajdweibull` *Adjustment By Weibull Distribution*

Description

Adjusted your sample by the weibull law, estimated these parameters using the method of maximum likelihood, and calculating the Akaike information criterion for one or several fitted model objects for which a log-likelihood value can be obtained, according to the formula -2*log-likelihood + k*npar, where npar represents the number of parameters in the fitted model, and k = 2 for the usual AIC, and computes confidence intervals for one or more parameters in a fitted model (Law).

Usage

```
Ajdweibull(X, starts = list(shape = 1, scale = 1), leve = 0.95)
```

Arguments

`X`	a numeric vector of the observed values.
`starts`	named list. Initial values for optimizer.
`leve`	the confidence level required.

Details

The `optim` optimizer is used to find the minimum of the negative log-likelihood. An approximate covariance matrix for the parameters is obtained by inverting the Hessian matrix at the optimum.

For more detail consulted `mle`,`confint`,`AIC`.

`R` has the `[dqpr]weibull` functions to evaluate the density, the quantiles, and the cumulative distribution or generate pseudo random numbers from the weibull distribution.

Value

`coef`	Coefficients extracted from the model.
`AIC`	A numeric value with the corresponding AIC.
`vcov`	A matrix of the estimated covariances between the parameter estimates in the linear or non-linear predictor of the model.
`confint`	A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labelled as (1-level)/2 and 1 - (1-level)/2.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`Ajdchisq` Adjustment By Chi-Squared Distribution,`Ajdexp` Adjustment By Exponential Distribution, `Ajdf` Adjustment By F Distribution,`Ajdgamma` Adjustment By Gamma Distribution, `Ajdlognorm` Adjustment By Log Normal Distribution,`Ajdnorm` Adjustment By Normal Distribution, `Ajdt` Adjustment By Student t Distribution,`Ajdbeta` Adjustment By Beta Distribution.

Examples

```
##X <- rweibull(100,2,1)
##Ajdweibull(X, starts = list(shape = 1, scale = 1), leve = 0.95)
```

`AnaSimFPT` *Simulation The First Passage Time FPT For A Simulated Diffusion Process*

Description

Simulation M-samples of the first passage time (FPT) by a simulated diffusion process with a fixed the threshold v.

Usage

```
AnaSimFPT(N, M, t0, Dt, T = 1, X0, v, drift, diff,
          ELRENA=c("No","Yes","Mean","Median"),
          Output = FALSE, Methods = c("Euler", "Milstein",
          "MilsteinS","Ito-Taylor", "Heun", "RK3"), ...)
```

Arguments

N	size of the diffusion process.
M	size of the FPT.
t0	initial time.
Dt	time step of the simulation (discretization).
T	final time.
X0	initial value of the process at time t0.
v	threshold (Risk).
drift	drift coefficient: an expression of two variables t and x.
diff	diffusion coefficient: an expression of two variables t and x.
ELRENA	if ELRENA = "No" not eliminate NA (Not Available),and if ELRENA="Yes" eliminate NA (Not Available), or replace NA by : mean(FPT) ,median(FPT).
Output	if Output = TRUE write a Output to an Excel 2007.
Methods	method of simulation ,see details snssde.
...	

Details

The stochastic differential equation of is :

$$dX(t) = a(t, X(t)) * dt + b(t, X(t)) * dW(t)$$

with a(t,X(t)) :drift coefficient and b(t,X(t)) :diffusion coefficient, W(t) is Wiener process.

We take interest in the random variable tau "first passage time", is defined by :

$$tau = inf(t >= 0\ X(t) <= vORX(t) >= v)$$

with v is the threshold.

For more detail consulted References.

Value

Random variable tau "FPT".

Note

Time of Calculating

The Ornstein-Uhlenbeck Process (example) drift <- expression(-5*x) diff <- expression(1)

system.time(AnaSimFPT(N=1000, M=30, t0=0, Dt=0.001, T = 1, X0=10, v=0.05,drift, diff, ELRENA ="No", Output = FALSE))

utilisateur systeme ecoule

1.89 0.55 2.62

system.time(AnaSimFPT(N=1000, M=100, t0=0, Dt=0.001, T = 1, X0=10, v=0.05,drift, diff, ELRENA ="No", Output = FALSE))

utilisateur systeme ecoule

5.74 1.64 7.78

system.time(AnaSimFPT(N=1000, M=500, t0=0, Dt=0.001, T = 1, X0=10, v=0.05,drift, diff, ELRENA ="Mean", Output = FALSE))

utilisateur systeme ecoule

26.07 7.78 37.93

system.time(AnaSimFPT(N=1000, M=500, t0=0, Dt=0.001, T = 1, X0=10, v=0.05,drift, diff, ELRENA ="Mean", Output = FALSE,Methods="RK3"))

utilisateur systeme ecoule

125.64 8.90 150.85

Author(s)

boukhetala Kamal, guidoum Arsalane.

References

1. K.Boukhetala, Estimation of the first passage time distribution for a simulated diffusion process, Maghreb Math.Rev, Vol.7, No 1, Jun 1998, pp. 1-25.
2. K.Boukhetala, Simulation study of a dispersion about an attractive centre. In proceedings of 11th Symposium Computational Statistics, edited by R.Dutter and W.Grossman, Wien , Austria, 1994, pp. 128-130.
3. K.Boukhetala,Modelling and simulation of a dispersion pollutant with attractive centre, Edited by Computational Mechanics Publications, Southampton ,U.K and Computational Mechanics Inc, Boston, USA, pp. 245-252.
4. K.Boukhetala, Kernel density of the exit time in a simulated diffusion, les Annales Maghrebines De L ingenieur, Vol , 12, N Hors Serie. Novembre 1998, Tome II, pp 587-589.

See Also

AnaSimX Simulation M-Samples of Random Variable X(v[t]) For A Simulated Diffusion Process, tho_M1 Simulation The FPT For Attractive Model(S = 1,Sigma), tho_M1 Simulation The FPT For Attractive Model(S >= 2,Sigma), tho_02diff Simulation FPT For Attractive Model for 2-Diffusion Processes.

Examples

```
## Example 1
## tau = inf(t>=0 \ X(t) <= v
## Ornstein-Uhlenbeck Process  or Gaussian Diffusion Models
## v = 0.05
drift <- expression(5*(-2-x))
diff <- expression(1)
AnaSimFPT(N=1000, M=30, t0=0, Dt=0.001, T = 1, X0=10, v=0.05, drift,
          diff,ELRENA ="No", Output = FALSE)
summary(tau)
hist(tau)
plot(density(tau,kernel ="gaussian"),col="red")
## v = -0.05
AnaSimFPT(N=1000, M=30, t0=0, Dt=0.001, T = 1, X0=10, v=-0.05, drift,
          diff,ELRENA ="No", Output = FALSE)
summary(tau)
hist(tau)
plot(density(tau,kernel ="gaussian"),col="red")
```

```
## Attention
## v = -3
## AnaSimFPT(N=1000, M=30, t0=0, Dt=0.001, T = 1, X0=10, v=-3, drift,
##           diff,ELRENA ="No", Output = FALSE)

## Example 2
## tau = inf(t>=0 \ X(t) >= v )
## v = 1
drift <- expression(2*(3-x))
diff <- expression(0.1)
AnaSimFPT(N=1000, M=30, t0=0, Dt=0.001, T = 1, X0=-5, v=1, drift,
          diff,ELRENA ="No", Output = FALSE)
summary(tau)
hist(tau)
plot(density(tau,kernel ="gaussian"),col="red")
## v = 3
AnaSimFPT(N=1000, M=30, t0=0, Dt=0.01, T = 1, X0=-5, v=3, drift,
          diff,ELRENA ="No", Output = FALSE)
summary(tau)
hist(tau)
plot(density(tau,kernel ="gaussian"),col="red")
## v = 3.1
##AnaSimFPT(N=1000, M=30, t0=0, Dt=0.01, T = 1, X0=-5, v=3.1, drift,
##          diff,ELRENA ="No", Output = FALSE)
## Remplaced NA by mean(tau) or median(tau)
##AnaSimFPT(N=1000, M=30, t0=0, Dt=0.01, T = 1, X0=-5, v=3.1, drift,
##          diff,ELRENA ="Yes", Output = FALSE)
##AnaSimFPT(N=1000, M=30, t0=0, Dt=0.01, T = 1, X0=-5, v=3.1, drift,
##          diff,ELRENA ="Mean", Output = FALSE)
##AnaSimFPT(N=1000, M=30, t0=0, Dt=0.01, T = 1, X0=-5, v=3.1, drift,
##          diff,ELRENA ="Median", Output = FALSE)
```

`AnaSimX` *Simulation M-Samples of Random Variable X(v[t]) For A Simulated Diffusion Process*

Description

Simulation M-samples of the random variable X(v(t)) by a simulated diffusion process with a fixed the time v , `v = k * Dt` with `k integer, 1 <= k <= N .`

Usage

```
AnaSimX(N, M, t0, Dt, T = 1, X0, v, drift, diff, Output = FALSE,
        Methods = c("Euler", "Milstein", "MilsteinS", "Ito-Taylor",
                    "Heun", "RK3"), ...)
```

Arguments

`N`	size of the diffusion process.
`M`	size of the random variable.
`t0`	initial time.
`Dt`	time step of the simulation `(discretization).`

`T`	final time.
`X0`	initial value of the process at time `t0`.
`v`	moment (time) between `t0` and `T` , `v = k * Dt` with `k integer, 1 <= k <= N` .
`drift`	drift coefficient: an expression of two variables `t` and `x`.
`diff`	diffusion coefficient: an expression of two variables `t` and `x`.
`Output`	if `Output = TRUE` write a `Output` to an Excel 2007.
`Methods`	method of simulation ,see details `snssde`.
`...`	

Details

The stochastic differential equation of is :

$$dX(t) = a(t, X(t)) * dt + b(t, X(t)) * dW(t)$$

with `a(t,X(t)) :drift coefficient` and `b(t,X(t)) :diffusion coefficient`, `W(t)` is Wiener process.

We take interest in the random variable `X(v)`, is defined by :

$$X = (t >= 0 \; X = X(v))$$

with `v` is the time between `t0` and `T` , `v = k * Dt` with `k integer, 1 <= k <= N` .

Value

Random variable "X(v(t))".

Note

Time of Calculating

The Ornstein-Uhlenbeck Process (example) drift <- expression(-5*x) diff <- expression(1)

system.time(AnaSimX(N=1000,M=30,t0=0,Dt=0.001,T=1,X0=0, v=0.5,drift,diff,Output=FALSE))

utilisateur systeme ecoule

1.88 0.56 2.59

system.time(AnaSimX(N=1000,M=30,t0=0,Dt=0.001,T=1,X0=0, v=0.5,drift,diff,Output=FALSE,Methods="RK3"))

utilisateur systeme ecoule

8.64 0.72 9.24

Author(s)

boukhetala Kamal, guidoum Arsalane.

References

1. K.Boukhetala, Estimation of the first passage time distribution for a simulated diffusion process, Maghreb Math.Rev, Vol.7, No 1, Jun 1998, pp. 1-25.
2. K.Boukhetala, Simulation study of a dispersion about an attractive centre. In proceedings of 11th Symposium Computational Statistics, edited by R.Dutter and W.Grossman, Wien , Austria, 1994, pp. 128-130.
3. K.Boukhetala,Modelling and simulation of a dispersion pollutant with attractive centre, Edited by Computational Mechanics Publications, Southampton ,U.K and Computational Mechanics Inc, Boston, USA, pp. 245-252.
4. K.Boukhetala, Kernel density of the exit time in a simulated diffusion, les Annales Maghrebines De L ingenieur, Vol , 12, N Hors Serie. Novembre 1998, Tome II, pp 587-589.

See Also

`AnaSimFPT` Simulation The First Passage Time FPT For A Simulated Diffusion Process, `tho_M1` Simulation The FPT For Attractive Model(S = 1,Sigma), `tho_M1` Simulation The FPT For Attractive Model(S >= 2,Sigma), `tho_02diff` Simulation FPT For Attractive Model for 2-Diffusion Processes.

Examples

```
## Example 1:  BM
## v = k * Dt with k integer , 1 <= k <= N .
## k = 500 nombre for discretization
## Dt = 0.001 ===> v = 500 * 0.001 = 0.5
drift <- expression(0)
diff <- expression(1)
AnaSimX(N=1000,M=30,t0=0,Dt=0.001,T=1,X0=0,v=0.5,drift,diff,Output=FALSE,Methods="Euler")
summary(X)
hist(X)
v=0.5
plot(density(X,kernel ="gaussian"),col="red")
x <- seq(min(X),max(X),length=1000)
curve(dnorm(x,0,v), col = 3, lwd = 2, add = TRUE,
      panel.first=grid(col="gray"))


## Example 2: BMG or BS
## v = k * Dt with k integer , 1 <= k <= N .
## k = 800 nombre for discretization
## Dt = 0.001 ===> v = 800 * 0.001 = 0.8
drift <- expression(2*x)
diff <- expression(x)
AnaSimX(N=1000,M=30,t0=0,Dt=0.001,T=1,X0=1,v=0.8,drift,diff,Output=FALSE,Methods="Euler")
summary(X)
hist(X)
plot(density(X,kernel ="gaussian"),col="red")
```

Asys *Evolution a Telegraphic Process in Time*

Description

Simulation the evolution of the telegraphic process (the availability of a system).

Usage

```
Asys(lambda, mu, t, T)
```

Arguments

`lambda`	the rate so that the system functions.
`mu`	the rate so that the system is broken down.
`t`	calculate the matrix of transition `p(t)` has at the time `t`.
`T`	final time of evolution the process `[0,T]`.

Details

Calculate the matrix of transition `p(t)` at time `t`, the space states of the telegraphic process is `(0,1)` with `0 : the system is broken down` and `1 : the system functions`, the initial distribution at time `t = 0` of the process is `p(t=0)=(1,0)` or `p(t=0)=(0,1)`.

Value

matrix `p(t)` at time `t`, and plot of evolution the process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`Telegproc` simulation a telegraphic process.

Examples

```
## evolution a telegraphic process in time [0 , 5]
## calculate the matrix of transition p(t = 10)
Asys(0.5,0.5,10,5)
```

BB *Creating Brownian Bridge Model*

Description

Simulation of brownian bridge model.

Usage

```
BB(N, t0, T, x0, y, output = FALSE)
```

Arguments

`N`	size of process.
`t0`	initial time.
`T`	final time.
`x0`	initial value of the process at time `t0`.
`y`	terminal value of the process at time `T`.
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

The function returns a trajectory of the brownian bridge starting at `x0` at time `t0` and ending at `y` at time `T`.

It is defined as :

$$Xt(t0, x0, T, y) = x0 + W(t - t0) - (t - t0/T - t0) * (W(T - t0) - y + x0)$$

This process is easily simulated using the simulated trajectory of the Wiener process `W(t)`.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BBF` simulation flow of brownian bridge Model, `diffBridge` Diffusion Bridge Models, `BMN` simulation brownian motion by the Normal Distribution , `BMRW` simulation brownian motion by a Random Walk, `GBM` simulation geometric brownian motion, `ABM` simulation arithmetic brownian motion, `snssde` Simulation Numerical Solution of SDE.

Examples

```
##brownian bridge model
##starting at x0 =0 at time t0=0 and ending at y =3 at time T =1.
BB(N=1000,t0=0,T=1,x0=0,y=3)
```

BBF	*Creating Flow of Brownian Bridge Model*

Description

Simulation flow of brownian bridge model.

Usage

```
BBF(N, M, t0, T, x0, y, output = FALSE)
```

Arguments

`N`	size of process.
`M`	number of trajectories.
`t0`	initial time.
`T`	final time.
`x0`	initial value of the process at time `t0`.
`y`	terminal value of the process at time `T`.
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

The function `BBF` returns a flow of the brownian bridge starting at `x0` at time `t0` and ending at `y` at time `T`.

It is defined as :

$$Xt(t0, x0, T, y) = x0 + W(t - t0) - (t - t0/T - t0) * (W(T - t0) - y + x0)$$

This process is easily simulated using the simulated trajectory of the Wiener process `W(t)`.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BB` simulation brownian bridge Model, `diffBridge` Diffusion Bridge Models, `BMN` simulation brownian motion by the Normal Distribution , `BMRW` simulation brownian motion by a Random Walk, `GBM` simulation geometric brownian motion, `ABM` simulation arithmetic brownian motion, `snssde` Simulation Numerical Solution of SDE.

Examples

```
## flow of brownian bridge model
## starting at x0 =1 at time t0=0 and ending at y = -2 at time T =1.
BBF(N=1000,M=5,t0=0,T=1,x0=1,y=-2)
```

Besselp *Creating Bessel process (by Milstein Scheme)*

Description

Simulation Besselp process by milstein scheme.

Usage

```
Besselp(N, M, t0, T, x0, alpha, output = FALSE)
```

Arguments

N	size of process.
M	number of trajectories.
t0	initial time.
T	final time.
x0	initial value of the process at time t0.
alpha	constant positive alpha >=2.
output	if output = TRUE write a output to an Excel 2007.

Details

The stochastic differential equation of Bessel process is :

$$dX(t) = (alpha - 1)/(2 * X(t)) * dt + dW(t)$$

with (alpha-1)/(2*X(t)) :drift coefficient and 1 :diffusion coefficient, W(t) is Wiener process, and the discretization dt = (T-t0)/N.

Constraints: alpha >= 2 and x0 =! 0.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

CEV Constant Elasticity of Variance Models, CIR Cox-Ingersoll-Ross Models, CIRhy modified CIR and hyperbolic Process, CKLS Chan-Karolyi-Longstaff-Sanders Models, DWP Double-Well Potential Model, GBM Model of Black-Scholes, HWV Hull-White/Vasicek Models, INFSR Inverse of Feller s Square Root models, JDP Jacobi Diffusion Process, PDP Pearson Diffusions Process, ROU Radial Ornstein-Uhlenbeck Process, diffBridge Diffusion Bridge Models, snssde Simulation Numerical Solution of SDE.

Examples

```
## Bessel Process
## alpha = 4
## dX(t) = 3/(2*x) * dt + dW(t)
## One trajectorie
Besselp(N=1000,M=1,t0=0,T=100,x0=1,alpha=4,output=FALSE)
```

BMcov *Empirical Covariance for Brownian Motion*

Description

Calculate empirical covariance of the Brownian Motion.

Usage

```
BMcov(N, M, T, C)
```

Arguments

N	size of process.
M	number of trajectories.
T	final time.
C	constant positive (`if C = 1 it is standard brownian motion`).

Details

The brownian motion is a process with increase independent of function the covariance `cov(BM) = C * min(t,s)`, If `t > s` than `cov(BM) = C * s` else `cov(BM) = C * t`.

Value

contour of the empirical covariance for brownian motion.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMN` simulation brownian motion by the Normal Distribution , `BMRW` simulation brownian motion by a Random Walk, `BMinf` brownian motion property(Time tends towards the infinite), `BMIrt` brownian motion property(invariance by reversal of time), `BMscal` brownian motion property (invariance by scaling).

Examples

```
## empirical covariance of 200 trajectories brownian standard
BMcov(N=100,M=250,T=1,C=1)
```

BMinf *Brownian Motion Property*

Description

Calculated the limit of standard brownian motion `limit(W(t)/t,0,T)`.

Usage

```
BMinf(N,T)
```

Arguments

N	size of process.
T	final time.

Details

Calculated the limit of standard brownian motion if the time tends towards the infinite,i.e the `limit(W(t)/t,0,T) = 0`.

Value

plot of `limit(W(t)/t)`.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMN` simulation brownian motion by the Normal Distribution , `BMRW` simulation brownian motion by a Random Walk, `BMIrt` brownian motion property(invariance by reversal of time), `BMscal` brownian motion property (invariance by scaling), `BMcov` empirical covariance for brownian motion.

Examples

```
BMinf(N=1000,T=10^5)
```

BMIrt *Brownian Motion Property (Invariance by reversal of time)*

Description

Brownian motion is invariance by reversal of time.

Usage

```
BMIrt(N, T)
```

Arguments

N	size of process.
T	final time.

Details

Brownian motion is invariance by reversal of time,i.e `W(t) = W(T-t) - W(T)`.

Value

plot of `W(T-t) - W(T)`.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMN` simulation brownian motion by the Normal Distribution , `BMRW` simulation brownian motion by a Random Walk, `BMinf` Brownian Motion Property (time tends towards the infinite), `BMscal` brownian motion property (invariance by scaling), `BMcov` empirical covariance for brownian motion.

Examples

```
BMIrt(N=1000,T=1)
```

BMIto1 *Properties of the stochastic integral and Ito Process [1]*

Description

Simulation of the Ito `integral(W(s)dW(s),0,t)`.

Usage

```
BMIto1(N, T, output = FALSE)
```

Arguments

N	size of process.
T	final time.
output	if `output = TRUE` write a `output` to an Excel 2007.

Details

However the Ito integral also has the peculiar property, amongst others, that :

$$integral(W(s)dW(s), 0, t) = 0.5 * (W(t)^2 - t)$$

from classical calculus for Ito integral with `w(0) = 0`.

The follows from the algebraic rearrangement :

$$integral(W(s)dW(s), 0, t) = sum(W(t) * (W(t+1) - W(t)), 0, t)$$

Value

data frame(time,Ito,sum.Ito) and plot of the Ito integral.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMIto2` simulation of the Ito integral[2], `BMItoC` properties of the stochastic integral and Ito processes[3], `BMItoP` properties of the stochastic integral and Ito processes[4], `BMItoT` properties of the stochastic integral and Ito processes[5].

Examples

```
##
BMIto1(N=1000,T=1)
## comparison with BMIto2
system.time(BMIto1(N=10^4,T=1))
system.time(BMIto2(N=10^4,T=1))
```

BMIto2 *Properties of the stochastic integral and Ito Process [2]*

Description

Simulation of the Ito `integral(W(s)dW(s),0,t)`.

Usage

```
BMIto2(N, T, output = FALSE)
```

Arguments

N	size of process.
T	final time.
output	if `output = TRUE` write a `output` to an Excel 2007.

Details

However the Ito integral also has the peculiar property, amongst others, that :

$$integral(W(s)dW(s), 0, t) = 0.5 * (W(t)^2 - t)$$

from classical calculus for Ito integral with `w(0) = 0`.

The follows from the algebraic rearrangement :

$$integral(W(s)dW(s), 0, t) = sum(W(t) * (W(t + 1) - W(t)), 0, t)$$

Value

data frame(time,Ito,sum.Ito) and plot of the Ito integral.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMIto1` simulation of the Ito integral[1], `BMItoC` properties of the stochastic integral and Ito processes[3], `BMItoP` properties of the stochastic integral and Ito processes[4], `BMItoT` properties of the stochastic integral and Ito processes[5].

Examples

```
##
BMIto2(N=1000,T=1)
## comparison with BMIto1
system.time(BMIto2(N=10^4,T=1))
system.time(BMIto1(N=10^4,T=1))
```

BMItoC *Properties of the stochastic integral and Ito Process [3]*

Description

Simulation of the Ito `integral(alpha*dW(s),0,t)`.

Usage

```
BMItoC(N, T, alpha, output = FALSE)
```

Arguments

`N`	size of process.
`T`	final time.
`alpha`	constant.
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

However the Ito integral also has the peculiar property, amongst others, that :

$$integral(alpha * dW(s), 0, t) = alpha * W(t)$$

from classical calculus for Ito integral with `w(0) = 0`.

The follows from the algebraic rearrangement :

$$integral(alpha * dW(s), 0, t) = sum(alpha * (W(t + 1) - W(t)), 0, t)$$

Value

data frame(time,Ito,sum.Ito) and plot of the Ito integral.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMIto1` simulation of the Ito integral[1], `BMIto2` simulation of the Ito integral[2], `BMItoP` properties of the stochastic integral and Ito processes[4], `BMItoT` properties of the stochastic integral and Ito processes[5].

Examples

```
##
BMItoC(N=1000,T=1,alpha=2)
```

BMItoP — *Properties of the stochastic integral and Ito Process [4]*

Description

Simulation of the Ito `integral(W(s)^n*dW(s),0,t)`.

Usage

```
BMItoP(N, T, power, output = FALSE)
```

Arguments

`N`	size of process.
`T`	final time.
`power`	constant.
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

However the Ito integral also has the peculiar property, amongst others, that :

$integral(W(s)^n * dW(s), 0, t) = W(t)^(n+1)/(n+1) - (n/2) * integral(W(s)^n - 1 * ds, 0, t)$

from classical calculus for Ito integral with `w(0) = 0`.
The follows from the algebraic rearrangement :

$$integral(W(s)^n * dW(s), 0, t) = sum(W(t)^n * (W(t+1) - W(t)), 0, t)$$

Value

data frame(time,Ito,sum.Ito) and plot of the Ito integral.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMIto1` simulation of the Ito integral[1], `BMIto2` simulation of the Ito integral[2], `BMItoC` properties of the stochastic integral and Ito processes[3], `BMItoT` properties of the stochastic integral and Ito processes[5].

Examples

```
## if power = 1
## integral(W(s) * dW(s),0,t) = W(t)^2/2 - 1/2 * t
BMItoP(N=1000,T=1,power =1)
## if power = 2
## integral(W(s)^2 * dW(s),0,t) = W(t)^3/3 - 2/2 * integral(W(s)*ds,0,t)
BMItoP(N=1000,T=1,power =2)
```

BMItoT *Properties of the stochastic integral and Ito Process [5]*

Description

Simulation of the Ito `integral(s*dW(s),0,t)`.

Usage

```
BMItoT(N, T, output = FALSE)
```

Arguments

`N`	size of process.
`T`	final time.
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

However the Ito integral also has the peculiar property, amongst others, that :

$$integral(s * dW(s), 0, t) = t * W(t) - integral(W(s) * ds, 0, t)$$

from classical calculus for Ito integral with `w(0) = 0`.

The follows from the algebraic rearrangement :

$$integral(s * dW(s), 0, t) = sum(t * (W(t + 1) - W(t)), 0, t)$$

Value

data frame(time,Ito,sum.Ito) and plot of the Ito integral.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMIto1` simulation of the Ito integral[1], `BMIto2` simulation of the Ito integral[2], `BMItoC` properties of the stochastic integral and Ito processes[3], `BMItoP` properties of the stochastic integral and Ito processes[4].

Examples

```
##
BMItoT(N=1000,T=1)
```

BMN *Creating Brownian Motion Model (by the Normal Distribution)*

Description

Simulation of the brownian motion model by the normal distribution.

Usage

```
BMN(N, t0, T, C, output = FALSE)
```

Arguments

`N`	size of process.
`t0`	initial time.
`T`	final time.
`C`	constant positive (`if C = 1 it is standard brownian motion`).
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

Given a fixed time increment `dt = (T-t0)/N`, one can easily simulate a trajectory of the Wiener process in the time interval `[t0,T]`. Indeed, for `W(dt)` it holds true that `W(dt) = W(dt) - W(0) ~> N(0,dt) ~> sqrt(dt) * N(0,1)`,`N(0,1)` normal distribution.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMRW` simulation brownian motion by a random walk, `BMNF` simulation flow of brownian motion by the normal distribution, `BMRWF` simulation flow of brownian motion by a random walk, `BB` Simulation of brownian bridge model, `GBM` simulation geometric brownian motion Model.

Examples

```
##
BMN(N=1000,t0=0,T=1,C=1)
BMN(N=1000,t0=0,T=1,C=10)
```

BMN2D *Simulation Two-Dimensional Brownian Motion (by the Normal Distribution)*

Description

simulation 2-dimensional brownian otion in plane (O,X,Y).

Usage

```
BMN2D(N, t0, T, x0, y0, Sigma, Step = FALSE, Output = FALSE)
```

Arguments

`N`	size of process.
`t0`	initial time.
`T`	final time.
`x0`	initial value of `BM1(t)` at time `t0`.
`y0`	initial value of `BM2(t)` at time `t0`.
`Sigma`	constant positive.
`Step`	if `Step = TRUE` ploting step by step.
`Output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

see , BMN

Value

data.frame(time,W1(t),W2(t)) and plot of process 2-D.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

BMN3D Simulation Three-Dimensional Brownian Motion.

Examples

```
BMN2D(N=5000, t0=0, T=1, x0=0, y0=0,Sigma=0.2,
     Step = FALSE, Output = FALSE)
```

BMN3D — *Simulation Three-Dimensional Brownian Motion (by the Normal Distribution)*

Description

simulation 3-dimensional brownian otion in (O,X,Y,Z).

Usage

```
BMN3D(N, t0, T, X0, Y0, Z0, Sigma, Output = FALSE)
```

Arguments

`N`	size of process.
`t0`	initial time.
`T`	final time.
`X0`	initial value of `BM1(t)` at time `t0`.
`Y0`	initial value of `BM2(t)` at time `t0`.
`Z0`	initial value of `BM3(t)` at time `t0`.
`Sigma`	constant positive.
`Output`	if `output = TRUE` write a `output` to an Excel 2007

Details

see , `BMN`

Value

data.frame(time,W1(t),W2(t),W3(t)) and plot of process 3-D.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMRW3D` Simulation Three-Dimensional Brownian Motion.

Examples

```
BMN3D(N=500, t0=0, T=1, X0=0.5, Y0=0.5, Z0=0.5,
     Sigma=0.3, Output = FALSE)
```

BMNF *Creating Flow of Brownian Motion (by the Normal Distribution)*

Description

Simulation flow of the brownian motion model by the normal distribution.

Usage

```
BMNF(N, M, t0, T, C, output = FALSE)
```

Arguments

`N`	size of process.
`M`	number of trajectories.
`t0`	initial time.
`T`	final time.
`C`	constant positive (`if C = 1 it is standard brownian motion`).
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

Given a fixed time increment `dt = (T-t0)/N`, one can easily simulate a flow of the Wiener process in the time interval `[t0,T]`. Indeed, for `W(dt)` it holds true that `W(dt) = W(dt) - W(0) ~> N(0,dt) ~> sqrt(dt) * N(0,1)`,`N(0,1)` normal distribution.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMRW` simulation brownian motion by a random walk, `BMN` simulation of brownian motion by the normal distribution, `BMRWF` simulation flow of brownian motion by a random walk, `BB` Simulation of brownian bridge model, `GBM` simulation geometric brownian motion Model.

Examples

```
##
BMNF(N=1000,M=5,t0=0,T=1,C=1)
BMNF(N=1000,M=5,t0=0,T=1,C=10)
```

`BMP` *Brownian Motion Property (trajectories brownian between function (+/-)2*sqrt(C*t))*

Description

trajectories Brownian lies between the two curves `(+/-)2*sqrt(C*t)`.

Usage

```
BMP(N, M, T, C)
```

Arguments

`N`	size of process.
`M`	number of trajectories.
`T`	final time.
`C`	constant positive (`if C = 1 it is standard brownian motion`).

Details

A flow of brownian motion lies between the two curves `(+/-)2*sqrt(C*t)`,`W(dt) - W(0) ~~> N(0,dt)`,`N(0,dt)` normal distribution.

Value

plot of the flow.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMscal` brownian motion property (invariance by scaling), `BMinf` brownian motion Property (time tends towards the infinite), `BMcov` empirical covariance for brownian motion, `BMIrt` brownian motion property(invariance by reversal of time).

Examples

```
##
BMP(N=1000,M=10,T=1,C=1)
```

BMRW *Creating Brownian Motion Model (by a Random Walk)*

Description

Simulation of the brownian motion model by a Random Walk.

Usage

```
BMRW(N, t0, T, C, output = FALSE)
```

Arguments

`N`	size of process.
`t0`	initial time.
`T`	final time.
`C`	constant positive (`if C = 1 it is standard brownian motion`).
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

One characterization of the Brownian motion says that it can be seen as the limit of a random walk in the following sense.

Given a sequence of independent and identically distributed random variables `X1, X2, . . . , Xn`, taking only two values `+1` and `-1` with equal probability and considering the partial sum, `Sn = X1+ X2+ . . . + Xn`. then, as `n --> lnf`,`P(Sn/sqrt(N) < x) = P( W(t) < x)`.

Where `[x]` is the integer part of the real number `x`. Please note that this result is a refinement of the central limit theorem that, in our case, asserts that `Sn/sqrt(n) ~~> N(0,1)`.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMN` simulation brownian motion by the normal distribution, `BMNF` simulation flow of brownian motion by the normal distribution, `BMRWF` simulation flow of brownian motion by a random walk, `BB` Simulation of brownian bridge model, `GBM` simulation geometric brownian motion Model.

Examples

```
##
BMRW(N=1000,t0=0,T=1,C=1)
BMRW(N=1000,t0=0,T=1,C=10)
```

BMRW2D	*Simulation Two-Dimensional Brownian Motion (by a Random Walk)*

Description

simulation 2-dimensional brownian otion in plane (O,X,Y).

Usage

```
BMRW2D(N, t0, T, x0, y0, Sigma, Step = FALSE, Output = FALSE)
```

Arguments

`N`	size of process.
`t0`	initial time.
`T`	final time.
`x0`	initial value of `BM1(t)` at time `t0`.
`y0`	initial value of `BM2(t)` at time `t0`.
`Sigma`	constant positive.
`Step`	if `Step = TRUE` ploting step by step.
`Output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

see , `BMRW`

Value

data.frame(time,W1(t),W2(t)) and plot of process 2-D.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMRW3D` Simulation Three-Dimensional Brownian Motion.

Examples

```
BMRW2D(N=5000, t0=0, T=1, x0=0, y0=0,Sigma=0.2,
     Step = FALSE, Output = FALSE)
```

BMRW3D *Simulation Three-Dimensional Brownian Motion (by a Random Walk)*

Description

simulation 3-dimensional brownian otion in (O,X,Y,Z).

Usage

```
BMRW3D(N, t0, T, X0, Y0, Z0, Sigma, Output = FALSE)
```

Arguments

`N`	size of process.
`t0`	initial time.
`T`	final time.
`X0`	initial value of `BM1(t)` at time `t0`.
`Y0`	initial value of `BM2(t)` at time `t0`.
`Z0`	initial value of `BM3(t)` at time `t0`.
`Sigma`	constant positive.
`Output`	if `output = TRUE` write a `output` to an Excel 2007

Details

see , `BMRW`

Value

data.frame(time,W1(t),W2(t),W3(t)) and plot of process 3-D.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMN3D` Simulation Three-Dimensional Brownian Motion.

Examples

```
BMRW3D(N=500, t0=0, T=1, X0=0.5, Y0=0.5, Z0=0.5,
     Sigma=0.3, Output = FALSE)
```

BMRWF *Creating Flow of Brownian Motion (by a Random Walk)*

Description

Simulation flow of the brownian motion model by a Random Walk.

Usage

```
BMRWF(N, M, t0, T, C, output = FALSE)
```

Arguments

`N`	size of process.
`M`	number of trajectories.
`t0`	initial time.
`T`	final time.
`C`	constant positive (`if C = 1 it is standard brownian motion`).
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

One characterization of the Brownian motion says that it can be seen as the limit of a random walk in the following sense.

Given a sequence of independent and identically distributed random variables `X1, X2, . . . , Xn`, taking only two values `+1` and `-1` with equal probability and considering the partial sum, `Sn = X1+ X2+ . . . + Xn`. then, as `n --> lnf`,`P(Sn/sqrt(N) < x) = P( W(t) < x)`.

Where `[x]` is the integer part of the real number `x`. Please note that this result is a refinement of the central limit theorem that, in our case, asserts that `Sn/sqrt(n) ~~> N(0,1)`.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMN` simulation brownian motion by the normal distribution, `BMRW` simulation brownian motion by a random walk, `BB` Simulation of brownian bridge model, `GBM` simulation geometric brownian motion Model.

Examples

```
##
BMRWF(N=1000,M=5,t0=0,T=1,C=1)
BMRWF(N=1000,M=5,t0=0,T=1,C=10)
```

`BMscal` *Brownian Motion Property (Invariance by scaling)*

Description

Brownian motion with different scales.

Usage

```
BMscal(N, T, S1, S2, S3, output = FALSE)
```

Arguments

`N`	size of process.
`T`	final time.
`S1`	constant (`scale 1`).
`S2`	constant (`scale 2`).
`S3`	constant (`scale 3`).
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

Brownian motion is invariance by change the scales,i.e `W(t) = (1/S) * W(S^2 * t)`, `S` is scale.

Value

data.frame(w1,w2,w3) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMinf` brownian motion Property (time tends towards the infinite), `BMcov` empirical covariance for brownian motion, `BMIrt` brownian motion property(invariance by reversal of time).

Examples

```
##
BMscal(N=1000,T=10,S1=1,S2=1.1,S3=1.2)
```

BMStra *Stratonovitch Integral [1]*

Description

Simulation of the Stratonovitch `integral(W(s) o dW(s),0,t).`

Usage

```
BMStra(N, T, output = FALSE)
```

Arguments

`N`	size of process.
`T`	final time.
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

Stratonovitch integral as defined :

$$integral(f(t)odW(s),0,t) = lim(sum(0.5*(f(t[i])+f(t[i+1]))*(W(t[i+1])-W(t[i]))))$$

calculus for Stratonovitch integral with `w(0) = 0`:

$$integral(W(s)odW(s),0,t) = 0.5*W(t)^2$$

The discretization `dt = T/N`, and `W(t)` is Wiener process.

Value

data frame(time,Stra) and plot of the Stratonovitch integral.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMStraC` Stratonovitch Integral [2], `BMStraP` Stratonovitch Integral [3], `BMStraT` Stratonovitch Integral [4].

Examples

```
##
BMStra(N=1000, T=1, output = FALSE)
```

BMStraC *Stratonovitch Integral [2]*

Description

Simulation of the Stratonovitch `integral(alpha o dW(s),0,t)`.

Usage

```
BMStraC(N, T, alpha, output = FALSE)
```

Arguments

`N`	size of process.
`T`	final time.
`alpha`	constant.
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

Stratonovitch integral as defined :

$integral(f(t)odW(s), 0, t) = lim(sum(0.5*(f(t[i]) + f(t[i+1]))*(W(t[i+1]) - W(t[i]))))$

calculus for Stratonovitch integral with `w(0) = 0`:

$$integral(alphaodW(s), 0, t) = alpha*W(t)$$

The discretization `dt = T/N`, and `W(t)` is Wiener process.

Value

data frame(time,Stra) and plot of the Stratonovitch integral.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMStra` Stratonovitch Integral [1], `BMStraP` Stratonovitch Integral [3], `BMStraT` Stratonovitch Integral [4].

Examples

```
##
BMStraC(N=1000, T=1, alpha = 2,output = FALSE)
```

BMStraP	*Stratonovitch Integral [3]*

Description

Simulation of the Stratonovitch `integral(W(s)^n o dW(s),0,t)`.

Usage

```
BMStraP(N, T, power, output = FALSE)
```

Arguments

`N`	size of process.
`T`	final time.
`power`	constant.
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

Stratonovitch integral as defined :

$$integral(f(t)odW(s),0,t) = lim(sum(0.5*(f(t[i])+f(t[i+1]))*(W(t[i+1])-W(t[i]))))$$

calculus for Stratonovitch integral with `w(0) = 0`:

$$integral(W(s)^n odW(s),0,t) = lim(sum(0.5*(W(t[i])^(n-1)+W(t[i+1])^(n-1))*(W(t[i+1])^2-W(t[i])^2)))$$

The discretization `dt = T/N`, and `W(t)` is Wiener process.

Value

data frame(time,Stra) and plot of the Stratonovitch integral.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMStra` Stratonovitch Integral [1], `BMStraC` Stratonovitch Integral [2], `BMStraT` Stratonovitch Integral [4].

Examples

```
##
BMStraP(N=1000, T=1, power = 2,output = FALSE)
```

BMStraT *Stratonovitch Integral [4]*

Description

Simulation of the Stratonovitch `integral(s o dW(s),0,t)`.

Usage

```
BMStraT(N, T, output = FALSE)
```

Arguments

N	size of process.
T	final time.
output	if `output = TRUE` write a `output` to an Excel 2007.

Details

Stratonovitch integral as defined :

$$integral(f(t)odW(s),0,t) = lim(sum(0.5*(f(t[i])+f(t[i+1]))*(W(t[i+1])-W(t[i]))))$$

calculus for Stratonovitch integral with `w(0) = 0`:

$$integral(sodW(s),0,t) = lim(sum(0.5*(t[i]*(W(t[i+1])-W(t[i]))+t[i+1]*(W(t[i+1])-W(t[i])))))$$

The discretization `dt = T/N`, and `W(t)` is Wiener process.

Value

data frame(time,Stra) and plot of the Stratonovitch integral.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`BMStra` Stratonovitch Integral [1], `BMStraC` Stratonovitch Integral [2], `BMStraC` Stratonovitch Integral [3].

Examples

```
BMStraT(N=1000, T=1,output = FALSE)
```

`CEV` *Creating Constant Elasticity of Variance (CEV) Models (by Milstein Scheme)*

Description

Simulation constant elasticity of variance models by milstein scheme.

Usage

```
CEV(N, M, t0, T, x0, mu, sigma, gamma, output = FALSE)
```

Arguments

`N`	size of process.
`M`	number of trajectories.
`t0`	initial time.
`T`	final time.
`x0`	initial value of the process at time `t0`.
`mu`	constant (`mu * X(t) :drift coefficient`).
`sigma`	constant positive (`sigma * X(t)^gamma :diffusion coefficient`).
`gamma`	constant positive (`sigma * X(t)^gamma :diffusion coefficient`).
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

The Constant Elasticity of Variance `(CEV)` model also derives directly from the linear drift class, the discretization `dt = (T-t0)/N`.

The stochastic differential equation of `CEV` is :

$$dX(t) = mu * X(t) * dt + sigma * X(t)^g amma * dW(t)$$

with `mu * X(t) :drift coefficient` and `sigma * X(t)^gamma :diffusion coefficient`, `W(t)` is Wiener process.

This process is quite useful in modeling a skewed implied volatility. In particular,for `gamma < 1`, the skewness is negative, and for `gamma > 1` the skewness is positive. For `gamma = 1`, the `CEV` process is a particular version of the `geometric Brownian motion`.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`CIR` Cox-Ingersoll-Ross Models, `CIRhy` modified CIR and hyperbolic Process, `CKLS` Chan-Karolyi-Longstaff-Sanders Models, `DWP` Double-Well Potential Model, `GBM` Model of Black-Scholes, `HWV` Hull-White/Vasicek Models, `INFSR` Inverse of Feller s Square Root models, `JDP` Jacobi Diffusion Process, `PDP` Pearson Diffusions Process, `ROU` Radial Ornstein-Uhlenbeck Process, `diffBridge` Diffusion Bridge Models, `snssde` Simulation Numerical Solution of SDE.

Examples

```
## Constant Elasticity of Variance Models
## dX(t) = 0.3 *X(t) *dt + 2 * X(t)^1.2 * dW(t)
## One trajectorie
CEV(N=1000,M=1,t0=0,T=1,x0=0.1,mu=0.3,sigma=2,gamma=1.2)
```

`CIR` *Creating Cox-Ingersoll-Ross (CIR) Square Root Diffusion Models (by Milstein Scheme)*

Description

Simulation cox-ingersoll-ross models by milstein scheme.

Usage

```
CIR(N, M, t0, T, x0, theta, r, sigma, output = FALSE)
```

Arguments

`N`	size of process.
`M`	number of trajectories.
`t0`	initial time.
`T`	final time.
`x0`	initial value of the process at time `t0`.
`theta`	constant positive (`(r - theta * X(t)) :drift coefficient`).
`r`	constant positive (`(r - theta * X(t)) :drift coefficient`).
`sigma`	constant positive (`sigma * sqrt(X(t)) :diffusion coefficient`).
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

Another interesting family of parametric models is that of the Cox-Ingersoll-Ross process. This model was introduced by Feller as a model for population growth and became quite popular in finance after Cox, Ingersoll, and Ross proposed it to model short-term interest rates. It was recently adopted to model nitrous oxide emission from soil by Pedersen and to model the evolutionary rate variation across sites in molecular evolution.

The discretization `dt = (T-t0)/N`, and the stochastic differential equation of CIR is :

$$dX(t) = (r - theta * X(t)) * dt + sigma * sqrt(X(t)) * dW(t)$$

With `(r - theta *X(t)) :drift coefficient` and `sigma*sqrt(X(t)) :diffusion coefficient`, `W(t)` is Wiener process.

Constraints: `2*r > sigma^2`.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

CEV Constant Elasticity of Variance Models, CIRhy modified CIR and hyperbolic Process, CKLS Chan-Karolyi-Longstaff-Sanders Models, DWP Double-Well Potential Model, GBM Model of Black-Scholes, HWV Hull-White/Vasicek Models, INFSR Inverse of Feller s Square Root models, JDP Jacobi Diffusion Process, PDP Pearson Diffusions Process, ROU Radial Ornstein-Uhlenbeck Process, diffBridge Diffusion Bridge Models, snssde Simulation Numerical Solution of SDE.

Examples

```
## Cox-Ingersoll-Ross Models
## dX(t) = (0.1 - 0.2 *X(t)) *dt + 0.05 * sqrt(X(t)) * dW(t)
## One trajectorie
CIR(N=1000,M=1,t0=0,T=1,x0=0.2,theta=0.2,r=0.1,sigma=0.05)
```

CIRhy *Creating The modified CIR and hyperbolic Process (by Milstein Scheme)*

Description

Simulation the modified CIR and hyperbolic process by milstein scheme.

Usage

```
CIRhy(N, M, t0, T, x0, r, sigma, output = FALSE)
```

Arguments

N	size of process.
M	number of trajectories.
t0	initial time.
T	final time.
x0	initial value of the process at time t0.
r	constant (-r * X(t) :drift coefficient).
sigma	constant positive (sigma * sqrt(1+X(t)^2) :diffusion coefficient).
output	if output = TRUE write a output to an Excel 2007.

Details

The stochastic differential equation of the modified CIR is :

$$dX(t) = -r * X(t) * dt + sigma * sqrt(1 + X(t)^2) * dW(t)$$

With `-r*X(t) :drift coefficient` and `sigma*sqrt(1+X(t)^2) :diffusion coefficient`, `W(t)` is Wiener process, the discretization `dt = (T-t0)/N`.

Constraints: `r + (sigma^2)/2 > 0` (this is needed to make the process positive recurrent).

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`CEV` Constant Elasticity of Variance Models, `CIR` Cox-Ingersoll-Ross Models , `CKLS` Chan-Karolyi-Longstaff-Sanders Models, `DWP` Double-Well Potential Model, `GBM` Model of Black-Scholes, `HWV` Hull-White/Vasicek Models, `INFSR` Inverse of Feller s Square Root models, `JDP` Jacobi Diffusion Process, `PDP` Pearson Diffusions Process, `ROU` Radial Ornstein-Uhlenbeck Process, `diffBridge` Diffusion Bridge Models, `snssde` Simulation Numerical Solution of SDE.

Examples

```
## The modified CIR and hyperbolic Process
## dX(t) = - 0.3 *X(t) *dt + 0.9 * sqrt(1+X(t)^2) * dW(t)
## One trajectorie
CIRhy(N=1000,M=1,T=1,t0=0,x0=1,r=0.3,sigma=0.9)
```

`CKLS` *Creating The Chan-Karolyi-Longstaff-Sanders (CKLS) family of models (by Milstein Scheme)*

Description

Simulation the chan-karolyi-longstaff-sanders models by milstein scheme.

Usage

```
CKLS(N, M, t0, T, x0, r, theta, sigma, gamma, output = FALSE)
```

Arguments

`N`	size of process.
`M`	number of trajectories.
`t0`	initial time.
`T`	final time.
`x0`	initial value of the process at time `t0`.

r	constant ((r + theta *X(t)) :drift coefficient).
theta	constant ((r + theta *X(t)) :drift coefficient).
sigma	constant positive (sigma * X(t)^gamma :diffusion coefficient).
gamma	constant positive (sigma * X(t)^gamma :diffusion coefficient).
output	if output = TRUE write a output to an Excel 2007.

Details

The Chan-Karolyi-Longstaff-Sanders (CKLS) family of models is a class of parametric stochastic differential equations widely used in many finance applications, in particular to model interest rates or asset prices.

The CKLS process solves the stochastic differential equation :

$$dX(t) = (r + theta * X(t)) * dt + sigma * X(t)^gamma * dW(t)$$

With (r + theta * X(t)) :drift coefficient and sigma* X(t)^gamma :diffusion coefficient, W(t) is Wiener process, the discretization dt = (T-t0)/N.

This CKLS model is a further extension of the Cox-Ingersoll-Ross model and hence embeds all previous models.

The CKLS model does not admit an explicit transition density unless r = 0 or gamma = 0.5. It takes values in (0, + Inf) if r,theta > 0, and gamma > 0.5. In all cases, sigma is assumed to be positive.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

CEV Constant Elasticity of Variance Models, CIR Cox-Ingersoll-Ross Models, CIRhy modified CIR and hyperbolic Process, DWP Double-Well Potential Model, GBM Model of Black-Scholes, HWV Hull-White/Vasicek Models, INFSR Inverse of Feller s Square Root models, JDP Jacobi Diffusion Process, PDP Pearson Diffusions Process, ROU Radial Ornstein-Uhlenbeck Process, diffBridge Diffusion Bridge Models, snssde Simulation Numerical Solution of SDE.

Examples

```
## Chan-Karolyi-Longstaff-Sanders Models
## dX(t) = (0.3 + 0.01 *X(t)) *dt + 0.1 * X(t)^0.2 * dW(t)
## One trajectorie
CKLS(N=1000,M=1,T=1,t0=0,x0=1,r=0.3,theta=0.01,sigma=0.1,gamma=   0.2)
```

DATA1 *Observation of Ornstein-Uhlenbeck Process*

Description

Simulation the observation of Ornstein-Uhlenbeck Process by function OU.

Examples

```
data(DATA1)
plot(ts(DATA1,delta=0.001),type="l")
```

DATA2 *Observation of Geometric Brownian Motion Model*

Description

Simulation the observation of Geometric Brownian Motion Model by function GBM.

Examples

```
data(DATA2)
plot(ts(DATA2,delta=0.001),type="l")
```

DATA3 *Observation of Arithmetic Brownian Motion*

Description

Simulation the observation of Arithmetic Brownian Motion by function ABM.

Examples

```
data(DATA3)
plot(ts(DATA3,delta=0.001),type="l")
```

`diffBridge` *Creating Diffusion Bridge Models (by Euler Scheme)*

Description

Simulation of diffusion bridge models by euler scheme.

Usage

```
diffBridge(N, t0, T, x, y, drift, diffusion, Output = FALSE)
```

Arguments

`N`	size of process.
`t0`	initial time.
`T`	final time.
`x`	initial value of the process at time `t0`.
`y`	terminal value of the process at time `T`.
`drift`	drift coefficient: an expression of two variables `t` and `x`.
`diffusion`	diffusion coefficient: an expression of two variables `t` and `x`.
`Output`	if `Output = TRUE` write a `Output` to an Excel 2007.

Details

The function `diffBridge` returns a trajectory of the diffusion bridge starting at `x` at time `t0` and ending at `y` at time `T`, the discretization `dt = (T-t0)/N`.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`CEV` Constant Elasticity of Variance Models, `CKLS` Chan-Karolyi-Longstaff-Sanders Models, `CIR` Cox-Ingersoll-Ross Models, `CIRhy` modified CIR and hyperbolic Process, `DWP` Double-Well Potential Model, `GBM` Model of Black-Scholes, `HWV` Hull-White/Vasicek Models, `INFSR` Inverse of Feller s Square Root models, `JDP` Jacobi Diffusion Process, `PDP` Pearson Diffusions Process, `ROU` Radial Ornstein-Uhlenbeck Process, `snssde` Simulation Numerical Solution of SDE.

Examples

```
## example 1 : Ornstein-Uhlenbeck Bridge Model (x0=1,t0=0,y=3,T=1)
drift     <- expression( (3*(2-x)) )
diffusion <- expression( (2) )
diffBridge(N=1000,t0=0,T=1,x=1,y=1,drift,diffusion)


## example 2 : Brownian Bridge Model (x0=0,t0=0,y=1,T=1)
drift     <- expression( 0)
diffusion <- expression( 1 )
diffBridge(N=1000,t0=0,T=1,x=0,y=0,drift,diffusion)

## example 3 : Geometric Brownian Bridge Model (x0=1,t0=1,y=3,T=3)
drift     <- expression( (3*x) )
diffusion <- expression( (2*x) )
diffBridge(N=1000,t0=0,T=10,x=1,y=1,drift,diffusion)

## example 4 : sde\ dX(t)=(0.03*t*X(t)-X(t)^3)*dt+0.1*dW(t) (x0=0,t0=0,y=2,T=100)
drift     <- expression( (0.03*t*x-x^3) )
diffusion <- expression( (0.1) )
diffBridge(N=1000,t0=0,T=100,x=1,y=1,drift,diffusion)
```

`DWP` *Creating Double-Well Potential Model (by Milstein Scheme)*

Description

Simulation double-well potential model by milstein scheme.

Usage

```
DWP(N, M, t0, T, x0, output = FALSE)
```

Arguments

`N`	size of process.
`M`	number of trajectories.
`t0`	initial time.
`T`	final time.
`x0`	initial value of the process at time `t0`.
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

This model is interesting because of the fact that its density has a bimodal shape.

The process satisfies the stochastic differential equation :

$$dX(t) = (X(t) - X(t)^3) * dt + dW(t)$$

With `(X(t) - X(t)^3) :drift coefficient` and `1 is diffusion coefficient`, `W(t)` is Wiener process,and the discretization `dt = (T-t0)/N`.

This model is challenging in the sense that the Milstein approximation.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`CEV` Constant Elasticity of Variance Models, `CIR` Cox-Ingersoll-Ross Models, `CIRhy` modified CIR and hyperbolic Process, `CKLS` Chan-Karolyi-Longstaff-Sanders Models, `GBM` Model of Black-Scholes, `HWV` Hull-White/Vasicek Models, `INFSR` Inverse of Feller s Square Root models, `JDP` Jacobi Diffusion Process, `PDP` Pearson Diffusions Process, `ROU` Radial Ornstein-Uhlenbeck Process, `diffBridge` Diffusion Bridge Models, `snssde` Simulation Numerical Solution of SDE.

Examples

```
## Double-Well Potential Model
## dX(t) = (X(t) - X(t)^3) * dt + dW(t)
## One trajectorie
DWP(N=1000,M=1,T=1,t0=0,x0=1)
```

`fctgeneral` *Adjustment the Empirical Distribution of Random Variable X*

Description

Adjusted your empirical distribution of Random Variable X.

Usage

```
fctgeneral(Data, Law = c("exp", "GAmma", "chisq", "Beta", "fisher",
                      "student", "weibull", "Normlog", "Norm"))
```

Arguments

`Data`	a numeric vector of the observed values.
`Law`	distribution function with Adjusted. see details `Distributions` (R >= 2.12.1)

Details

calculating the empirical distribution `F[i]= (1/n)*Sum(V[i])` with `V[i]= 1 if x[i] <= X else V[i] = 0`.

And ajusted with the Distribution c("pexp","pgamma","pchisq", "pbeta","pf","pt","pweibull","plnorm","pnorm")

Value

Plot the empirical distribution with Adjustment and Estimation.

Note

Choose your best distribution with minimum AIC.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

hist_general Histograms Methods,Kern_general Kernel Methods.

Examples

```
## Example
## X <- rgamma(100,1,4)
## par(mfrow=c(2,2))
## fctgeneral(Data=X,Law=("exp"))
## fctgeneral(Data=X,Law=("GAmma"))
## fctgeneral(Data=X,Law=("weibull"))
## fctgeneral(Data=X,Law=("Normlog"))
```

---

fctrep_Meth — *Calculating the Empirical Distribution of Random Variable X*

Description

Calculating your empirical distribution of random variable X.

Usage

```
fctrep_Meth(X)
```

Arguments

X	a numeric vector of the observed values.

Details

calculating the empirical distribution `F[i]= (1/n)*Sum(V[i])` with `V[i]= 1 if x[i] <= X else V[i] = 0`.

Value

Plot the empirical distribution.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

hist_meth Histograms,Kern_meth Kernel Density.

Examples

```
X <- rexp(1000,2)
Y <- rgamma(1000,1,2)
Z <- rweibull(1000,1,1)
G <- rnorm(1000,mean(X),sd(X))
par(mfrow=c(2,2))
fctrep_Meth(X)
fctrep_Meth(Y)
fctrep_Meth(Z)
fctrep_Meth(G)
```

GBM — *Creating Geometric Brownian Motion (GBM) Models*

Description

Simulation geometric brownian motion or Black-Scholes models.

Usage

```
GBM(N, t0, T, x0, theta, sigma, output = FALSE)
```

Arguments

`N`	size of process.
`t0`	initial time.
`T`	final time.
`x0`	initial value of the process at time `t0` (`x0 > 0`).
`theta`	constant (`theta is the constant interest rateand` and `theta * X(t) :drift coefficient`).
`sigma`	constant positive (`sigma is volatility of risky activities` and `sigma * X(t):diffusion coefficient`).
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

This process is sometimes called the Black-Scholes-Merton model after its introduction in the finance context to model asset prices.

The process is the solution to the stochastic differential equation :

$$dX(t) = theta * X(t) * dt + sigma * X(t) * dW(t)$$

With `theta * X(t) :drift coefficient` and `sigma * X(t) : diffusion coefficient`, `W(t)` is Wiener process, the discretization `dt = (T-t0)/N`.

`sigma > 0`, the parameter `theta` is interpreted as the constant interest rate and `sigma` as the volatility of risky activities.

The explicit solution is :

$$X(t) = x0 * exp((theta - 0.5 * sigma^2) * t + sigma * W(t))$$

The conditional density function is `log-normal`.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

GBMF Flow of Geometric Brownian Motion, PEBS Parametric Estimation of Model Black-Scholes, snssde Simulation Numerical Solution of SDE.

Examples

```
## Black-Scholes Models
## dX(t) = 4 * X(t) * dt + 2 * X(t) *dW(t)
GBM(N=1000,T=1,t0=0,x0=1,theta=4,sigma=2)
```

GBMF *Creating Flow of Geometric Brownian Motion Models*

Description

Simulation flow of geometric brownian motion or Black-Scholes models.

Usage

```
GBMF(N, M, t0, T, x0, theta, sigma, output = FALSE)
```

Arguments

N	size of process.
M	number of trajectories.
t0	initial time.
T	final time.
x0	initial value of the process at time t0 (x0 > 0).
theta	constant (theta is the constant interest rateand and theta * X(t) :drift coefficient).
sigma	constant positive (sigma is volatility of risky activities and sigma * X(t):diffusion coefficient).
output	if output = TRUE write a output to an Excel 2007.

Details

This process is sometimes called the Black-Scholes-Merton model after its introduction in the finance context to model asset prices.

The process is the solution to the stochastic differential equation :

$$dX(t) = theta * X(t) * dt + sigma * X(t) * dW(t)$$

With `theta * X(t) :drift coefficient` and `sigma * X(t) : diffusion coefficient`, `W(t)` is Wiener process, the discretization `dt = (T-t0)/N`.

`sigma > 0`, the parameter `theta` is interpreted as the constant interest rate and `sigma` as the volatility of risky activities.

The explicit solution is :

$$X(t) = x0 * exp((theta - 0.5 * sigma^2) * t + sigma * W(t))$$

The conditional density function is `log-normal`.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`GBM` Geometric Brownian Motion, `PEBS` Parametric Estimation of Model Black-Scholes, `snssde` Simulation Numerical Solution of SDE.

Examples

```
## Flow of Black-Scholes Models
## dX(t) = 4 * X(t) * dt + 2 * X(t) *dW(t)
GBMF(N=1000,M=5,T=1,t0=0,x0=1,theta=4,sigma=2)
```

`hist_general` *Adjustment the Density of Random Variable X by Histograms Methods*

Description

Adjusted your density of random variable X by histograms methods with Different number of cells.

Usage

```
hist_general(Data, Breaks, Law = c("exp", "GAmma", "chisq", "Beta",
             "fisher", "student", "weibull", "Normlog", "Norm"))
```

Arguments

`Data`	a numeric vector of the observed values.
`Breaks`	one of: o a vector giving the breakpoints between histogram cells. o a single number giving the number of cells for the histogram. o a function to compute the number of cells. o Breaks = c('scott','Sturges','FD') or manual.
`Law`	distribution function with Adjusted. see details `Distributions` (R >= 2.12.1)

Details

Ajusted the density for random variable X by histograms methods with Different number of cells see details `nclass.scott`, ajusted with the Distribution c("dexp","dgamma", "dchisq","dbeta","df","dt","dweibull", "dlnorm","dnorm").

Value

plot.histogram with Adjustment and Estimation.

Note

Choose your best distribution with minimum AIC.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`fctgeneral` empirical distribution, `Kern_general` Kernel Methods.

Examples

```
##
X <- rexp(1000,2)
par(mfrow=c(2,2))
hist_general(Data=X, Breaks='FD', Law="exp")
hist_general(Data=X, Breaks='scott', Law="exp")
hist_general(Data=X, Breaks='Sturges', Law="exp")
hist_general(Data=X, Breaks=60, Law="exp")
```

`hist_meth` *Histograms of Random Variable X*

Description

The generic function hist_meth computes a histogram of the given data values.

Usage

```
hist_meth(X, Breaks, Prob = c("TRUE", "FALSE"))
```

Arguments

- `X` a numeric vector of the observed values.
- `Breaks` one of: o a vector giving the breakpoints between histogram cells. o a single number giving the number of cells for the histogram. o a function to compute the number of cells. o Breaks = c('scott','Sturges','FD') or manual.
- `Prob` logical; if TRUE, the histogram graphic is a representation of frequencies, the counts component of the result; if FALSE, probability densities, component density, are plotted (so that the histogram has a total area of one). Defaults to TRUE if and only if breaks are equidistant (and probability is not specified).

Details

The definition of histogram differs by source (with country-specific biases). R's default with equispaced breaks (also the default) is to plot the counts in the cells defined by breaks. Thus the height of a rectangle is proportional to the number of points falling into the cell, as is the area provided the breaks are equally-spaced.

Value

plot.histogram for the random variable X.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`Kern_meth` Kernel Density,`fctrep_Meth` Empirical Distribution.

Examples

```
##
X <- rexp(1000,2)
X11()
hist_meth(X, Breaks='scott', Prob ="TRUE")
curve(dexp(x, 2), col = 2, lwd = 2, add = TRUE)
X11()
hist_meth(X, Breaks='FD', Prob ="TRUE")
curve(dgamma(x,1, 2), col = 2, lwd = 2, add = TRUE)
X11()
hist_meth(X, Breaks=100, Prob ="TRUE")
curve(dweibull(x,1, 0.5),col=2, lwd = 2, add = TRUE)
```

HWV *Creating Hull-White/Vasicek (HWV) Gaussian Diffusion Models*

Description

Simulation the Hull-White/Vasicek or gaussian diffusion models.

Usage

```
HWV(N, t0, T, x0, theta, r, sigma, output = FALSE)
```

Arguments

`N`	size of process.
`t0`	initial time.
`T`	final time.
`x0`	initial value of the process at time `t0`.
`theta`	constant (`theta is the long-run equilibrium value of the process` and `r*(theta -X(t)) :drift coefficient`).
`r`	constant positive (`r is speed of reversion` and `r*(theta -X(t)):drift coefficient`).
`sigma`	constant positive (`sigma (volatility) :diffusion coefficient`).
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

The Hull-White/Vasicek `(HWV)` short rate class derives directly from SDE with mean-reverting drift:

$$dX(t) = r * (theta - X(t)) * dt + sigma * dW(t)$$

With `r *(theta- X(t)) :drift coefficient` and `sigma : diffusion coefficient`, `W(t)` is Wiener process, the discretization `dt = (T-t0)/N`.

The process is also ergodic, and its invariant law is the `Gaussian density`.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`HWVF` Flow of Gaussian Diffusion Models, `PEOUG` Parametric Estimation of Hull-White/Vasicek Models, `snssde` Simulation Numerical Solution of SDE.

Examples

```
## Hull-White/Vasicek Models
## dX(t) = 4 * (2.5 - X(t)) * dt + 1 *dW(t)
HWV(N=1000,t0=0,T=1,x0=10,theta=2.5,r=4,sigma=1)
## if theta = 0 than "OU" = "HWV"
## dX(t) = 4 * ( 0 - X(t)) * dt + 1 *dW(t)
system.time(OU(N=10^4,t0=0,T=1,x0=10,r=4,sigma=1))
system.time(HWV(N=10^4,t0=0,T=1,x0=10,theta=0,r=4,sigma=1))
```

HWVF *Creating Flow of Hull-White/Vasicek (HWV) Gaussian Diffusion Models*

Description

Simulation flow of the Hull-White/Vasicek or gaussian diffusion models.

Usage

```
HWVF(N, M, t0, T, x0, theta, r, sigma, output = FALSE)
```

Arguments

`N`	size of process.
`M`	number of trajectories.
`t0`	initial time.
`T`	final time.
`x0`	initial value of the process at time `t0`.
`theta`	constant (`theta is the long-run equilibrium value of the process` and `r*(theta -X(t)) :drift coefficient`).
`r`	constant positive (`r is speed of reversion` and `r*(theta -X(t)):drift coefficient`).
`sigma`	constant positive (`sigma (volatility) :diffusion coefficient`).
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

The Hull-White/Vasicek `(HWV)` short rate class derives directly from SDE with mean-reverting drift:

$$dX(t) = r * (theta - X(t)) * dt + sigma * dW(t)$$

With `r *(theta- X(t)) :drift coefficient` and `sigma : diffusion coefficient`, `W(t)` is Wiener process, the discretization `dt = (T-t0)/N`.

The process is also ergodic, and its invariant law is the `Gaussian density`.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`HWV` Hull-White/Vasicek Models, `PEOUG` Parametric Estimation of Hull-White/Vasicek Models, `snssde` Simulation Numerical Solution of SDE.

Examples

```
## flow of Hull-White/Vasicek Models
## dX(t) = 4 * (2.5 - X(t)) * dt + 1 *dW(t)
HWVF(N=1000,M=10,t0=0,T=1,x0=10,theta=2.5,r=4,sigma=1)
## if theta = 0 than "OUF" = "HWVF"
## dX(t) = 4 * ( 0 - X(t)) * dt + 1 *dW(t)
system.time(HWVF(N=1000,M=10,t0=0,T=1,x0=10,theta=0,r=4,sigma=1))
system.time(OUF(N=1000,M=5,t0=0,T=1,x0=10,r=4,sigma=1))
```

Hyproc — *Creating The Hyperbolic Process (by Milstein Scheme)*

Description

Simulation hyperbolic process by milstein scheme.

Usage

```
Hyproc(N, M, t0, T, x0, theta, output = FALSE)
```

Arguments

`N`	size of process.
`M`	number of trajectories.
`t0`	initial time.
`T`	final time.
`x0`	initial value of the process at time `t0`.
`theta`	constant positive.
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

A process `X` satisfying :

$$dX(t) = (-theta * X(t)/sqrt(1 + X(t)^2)) * dt + dW(t)$$

With `(-theta*X(t)/sqrt(1+X(t)^2)) :drift coefficient` and `1 :diffusion coefficient`, `W(t)` is Wiener process, discretization `dt = (T-t0)/N`.

Constraints: `theta > 0`.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`Hyprocg` General Hyperbolic Diffusion, `CIRhy` modified CIR and hyperbolic Process, `snssde` Simulation Numerical Solution of SDE.

Examples

```
## Hyperbolic Process
## dX(t) = (-2*X(t)/sqrt(1+X(t)^2)) *dt + dW(t)
## One trajectorie
Hyproc(N=1000,M=1,T=100,t0=0,x0=3,theta=2)
```

`Hyprocg` *Creating The General Hyperbolic Diffusion (by Milstein Scheme)*

Description

Simulation the general hyperbolic diffusion by milstein scheme.

Usage

```
Hyprocg(N, M, t0, T, x0, beta, gamma, theta, mu, sigma, output = FALSE)
```

Arguments

`N`	size of process.
`M`	number of trajectories.
`t0`	initial time.
`T`	final time.
`x0`	initial value of the process at time `t0`.
`beta`	constant `(0.5*sigma^2*(beta-(gamma*X(t))/sqrt(theta^2+(X(t)-mu)^2)):drift coefficient)`.
`gamma`	constant positive `(0.5*sigma^2*(beta-(gamma*X(t))/sqrt(theta^2+(X(t)-mu)^2)):drift coefficient)`.
`theta`	constant positive `(0.5*sigma^2*(beta-(gamma*X(t))/sqrt(theta^2+(X(t)-mu)^2)):drift coefficient)`.
`mu`	constant `(0.5*sigma^2*(beta-(gamma*X(t))/sqrt(theta^2+(X(t)-mu)^2)):drift coefficient)`.
`sigma`	constant positive `( sigma :diffusion coefficient)`.
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

A process `X` satisfying :

$$dX(t) = (0.5 * sigma^2 * (beta - (gamma * X(t))/sqrt(theta^2 + (X(t) - mu)^2)) * dt + dW(t)$$

With `(0.5*sigma^2*(beta-(gamma*X(t))/sqrt(theta^2+(X(t)-mu)^2)):drift coefficient` and `sigma :diffusion coefficient`, `W(t)` is Wiener process, discretization `dt = (T-t0)/N`.

The parameters gamma > 0 and 0 <= abs(beta) < gamma determine the shape of the distribution, and theta >= 0, and mu are, respectively, the scale and location parameters of the distribution.

Constraints: gamma > 0,0 <= abs(beta) < gamma,theta >= 0,sigma > 0.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

Hyproc Hyperbolic Process, CIRhy modified CIR and hyperbolic Process, snssde Simulation Numerical Solution of SDE.

Examples

```
## Hyperbolic Process
## dX(t) = 0.5 * (2)^2*(0.25-(0.5*X(t))/sqrt(2^2+(X(t)-1)^2)) *dt + 2* dW(t)
## One trajectorie
Hyprocg(N=1000,M=1,T=100,t0=0,x0=-10,beta=0.25,gamma=0.5,theta=2,mu=1,sigma=2)
```

INFSR *Creating Ahn and Gao model or Inverse of Feller Square Root Models (by Milstein Scheme)*

Description

Simulation the inverse of feller square root model by milstein scheme.

Usage

```
INFSR(N, M, t0, T, x0, theta, r, sigma, output = FALSE)
```

Arguments

N	size of process.
M	number of trajectories.
t0	initial time.
T	final time.
x0	initial value of the process at time t0.
theta	constant (X(t)*(theta-(sigma^3-theta*r)*X(t)) :drift coefficient).
r	constant (X(t)*(theta-(sigma^3-theta*r)*X(t)) :drift coefficient).
sigma	constant positive (sigma * X(t)^(3/2) :diffusion coefficient).
output	if output = TRUE write a output to an Excel 2007.

Details

A process `X` satisfying :

$$dX(t) = X(t) * (theta - (sigma^3 - theta * r) * X(t)) * dt + sigma * X(t)^(3/2) * dW(t)$$

With `X(t)*(theta-(sigma^3-theta*r)*X(t)) :drift coefficient` and `sigma * X(t)^(3/2) :diffusion coefficient`, `W(t)` is Wiener process, discretization `dt = (T-t0)/N`.

The conditional distribution of this process is related to that of the Cox-Ingersoll-Ross `(CIR)` model.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`CEV` Constant Elasticity of Variance Models, `CIR` Cox-Ingersoll-Ross Models, `CIRhy` modified CIR and hyperbolic Process, `CKLS` Chan-Karolyi-Longstaff-Sanders Models, `DWP` Double-Well Potential Model, `GBM` Model of Black-Scholes, `HWV` Hull-White/Vasicek Models, `JDP` Jacobi Diffusion Process, `PDP` Pearson Diffusions Process, `ROU` Radial Ornstein-Uhlenbeck Process, `diffBridge` Diffusion Bridge Models, `snssde` Simulation Numerical Solution of SDE.

Examples

```
## Inverse of Feller Square Root Models
## dX(t) = X(t)*(0.5-(1^3-0.5*0.5)*X(t)) * dt + 1 * X(t)^(3/2) * dW(t)
## One trajectorie
INFSR(N=1000,M=1,T=50,t0=0,x0=0.5,theta=0.5,r=0.5,sigma=1)
```

`JDP` *Creating The Jacobi Diffusion Process (by Milstein Scheme)*

Description

Simulation the jacobi diffusion process by milstein scheme.

Usage

```
JDP(N, M, t0, T, x0, theta, output = FALSE)
```

Arguments

N	size of process.
M	number of trajectories.
t0	initial time.
T	final time.
x0	initial value of the process at time t0.
theta	constant positive.
output	if output = TRUE write a output to an Excel 2007.

Details

The Jacobi diffusion process is the solution to the stochastic differential equation :

$$dX(t) = -theta * (X(t) - 0.5) * dt + sqrt(theta * X(t) * (1 - X(t))) * dW(t)$$

With -theta * (X(t) - 0.5) :drift coefficient and sqrt(theta*X(t)*(1-X(t))) :diffusion coefficient, W(t) is Wiener process, discretization dt = (T-t0)/N.

For theta > 0. It has an invariant distribution that is uniform on [0,1].

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

CEV Constant Elasticity of Variance Models, CIR Cox-Ingersoll-Ross Models, CIRhy modified CIR and hyperbolic Process, CKLS Chan-Karolyi-Longstaff-Sanders Models, DWP Double-Well Potential Model, GBM Model of Black-Scholes, HWV Hull-White/Vasicek Models, INFSR Inverse of Feller s Square Root models, PDP Pearson Diffusions Process, ROU Radial Ornstein-Uhlenbeck Process, diffBridge Diffusion Bridge Models, snssde Simulation Numerical Solution of SDE.

Examples

```
## Jacobi Diffusion Process
## dX(t) = -0.05 * (X(t)-0.5)* dt + sqrt(0.05*X(t)*(1-X(t))) * dW(t),
## One trajectorie
JDP(N=1000,M=1,T=100,t0=0,x0=0,theta=0.05)
```

Kern_general *Adjustment the Density of Random Variable by Kernel Methods*

Description

kernel density estimates. Its default method does so with the given kernel and bandwidth for univariate observations, and adjusted your density with distributions.

Usage

```
Kern_general(Data, bw, k, Law = c("exp", "GAmma", "chisq", "Beta",
            "fisher", "student", "weibull", "Normlog", "Norm"))
```

Arguments

Data	a numeric vector of the observed values.
bw	the smoothing bandwidth to be used. The kernels are scaled such that this is the standard deviation of the smoothing kernel. bw=c('Irt','scott','Ucv','Bcv','SJ') or manual, see details `bw.nrd0`
k	a character string giving the smoothing kernel to be used. This must be one of "gaussian", "rectangular", "triangular", "epanechnikov", "biweight", "cosine" or "optcosine"
Law	distribution function with Adjusted. see details `Distributions` (R >= 2.12.1)

Details

see details `density`

Value

plot.density estimated with Adjustment.

Note

- bw='Irt' ===> bw= bw.nrd0(X), implements a rule-of-thumb for choosing the bandwidth of a Gaussian kernel density estimator.
- bw='scott' ===> bw= bw.nrd(X) ,is the more common variation given by Scott.
- bw='Ucv' ===> bw= bw.ucv(X) , implement unbiased cross-validation.
- bw='Bcv' ===> bw= bw.bcv(X) , implement biased cross-validation.
- bw='SJ' ===> bw= bw.SJ(X) , implements the methods of Sheather & Jones.
- Choose your best distribution with minimum AIC.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`fctgeneral` empirical distribution,`hist_general` Histograms Methods.

Examples

```
##
X <- rexp(1000,1)
par(mfrow=c(2,2))
Kern_general(Data=X, bw='Irt', k="gaussian", Law = c("exp"))
Kern_general(Data=X, bw='scott', k="gaussian", Law = c("exp"))
Kern_general(Data=X, bw='Ucv', k="gaussian", Law = c("exp"))
Kern_general(Data=X, bw=0.3, k="gaussian", Law = c("exp"))
```

`Kern_meth` *Kernel Density of Random Variable X*

Description

kernel density estimates. Its default method does so with the given kernel and bandwidth for univariate observations.

Usage

```
Kern_meth(X, bw, k)
```

Arguments

`X`	a numeric vector of the observed values.
`bw`	the smoothing bandwidth to be used. The kernels are scaled such that this is the standard deviation of the smoothing kernel. bw=c('Irt','scott','Ucv','Bcv','SJ') or manual, see details `bw.nrd0`
`k`	a character string giving the smoothing kernel to be used. This must be one of "gaussian", "rectangular", "triangular", "epanechnikov", "biweight", "cosine" or "optcosine"

Details

see details `plot.density`

Value

plot.density for your data.

Note

- bw='Irt' ===> bw= bw.nrd0(X), implements a rule-of-thumb for choosing the bandwidth of a Gaussian kernel density estimator.
- bw='scott' ===> bw= bw.nrd(X) ,is the more common variation given by Scott.
- bw='Ucv' ===> bw= bw.ucv(X) , implement unbiased cross-validation.
- bw='Bcv' ===> bw= bw.bcv(X) , implement biased cross-validation.
- bw='SJ' ===> bw= bw.SJ(X) , implements the methods of Sheather & Jones.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`hist_meth` Histograms,`fctrep_Meth` Empirical Distribution.

Examples

```
## Example 1
## fixed bw with different kernel
X <- rbeta(1000,1,2)
par(mfrow=c(2,2))
Kern_meth(X, bw='Ucv', k="rectangular")
Kern_meth(X, bw='Ucv',k="triangular")
Kern_meth(X, bw='Ucv',k="epanechnikov")
Kern_meth(X, bw='Ucv',k="cosine")

## Example 2
## fixed kernel with different bw
Y <- rlnorm(1000)
par(mfrow=c(2,2))
Kern_meth(Y, bw='Irt', k="epanechnikov")
Kern_meth(Y, bw='Ucv',k="epanechnikov")
Kern_meth(Y, bw='scott',k="epanechnikov")
Kern_meth(Y, bw=0.4,k="epanechnikov")
```

MartExp — *Creating The Exponential Martingales Process*

Description

Simulation the exponential martingales.

Usage

```
MartExp(N, t0, T, sigma, output = FALSE)
```

Arguments

`N`	size of process.
`t0`	initial time.
`T`	final time.
`sigma`	constant positive (`sigma is volatility`).
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

That is to say `W(t)` a Brownian movement the following processes are continuous martingales :

1. `X(t) = W(t)^2 - t.`
2. `Y(t) = exp( integral(f(s)dW(s),0,t)- 0.5 * integral(f(s)^2 ds,0,t) ).`

Value

data.frame(time,x,y) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

Examples

```
## Exponential Martingales Process
MartExp(N=1000,t0=0,T=1,sigma=2)
```

`OU` *Creating Ornstein-Uhlenbeck Process*

Description

Simulation the ornstein-uhlenbeck or Hull-White/Vasicek model.

Usage

```
OU(N, t0, T, x0, r, sigma, output = FALSE)
```

Arguments

`N`	size of process.
`t0`	initial time.
`T`	final time.
`x0`	initial value of the process at time `t0`.
`r`	constant positive (`r is speed of reversion` and `-r * X(t) :drift coefficient`).
`sigma`	constant positive (`sigma (volatility) :diffusion coefficient`).
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

The Ornstein-Uhlenbeck or Vasicek process is the unique solution to the following stochastic differential equation :

$$dX(t) = -r * X(t) * dt + sigma * dW(t)$$

With `-r * X(t) :drift coefficient` and `sigma : diffusion coefficient`, `W(t)` is Wiener process, the discretization `dt = (T-t0)/N`.

Please note that the process is stationary only if `r > 0`.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

OUF Flow of Ornstein-Uhlenbeck Process, PEOU Parametric Estimation of Ornstein-Uhlenbeck Model, PEOUexp Explicit Estimators of Ornstein-Uhlenbeck Model, snssde Simulation Numerical Solution of SDE.

Examples

```
## Ornstein-Uhlenbeck Process
## dX(t) = -2 * X(t) * dt + 1 *dW(t)
OU(N=1000,t0=0,T=10,x0=10,r=2,sigma=1)
```

OUF *Creating Flow of Ornstein-Uhlenbeck Process*

Description

Simulation flow of ornstein-uhlenbeck or Hull-White/Vasicek model.

Usage

```
OUF(N, M, t0, T, x0, r, sigma, output = FALSE)
```

Arguments

N	size of process.
M	number of trajectories.
t0	initial time.
T	final time.
x0	initial value of the process at time t0.
r	constant positive (r is speed of reversion and -r * X(t) :drift coefficient).
sigma	constant positive (sigma (volatility) :diffusion coefficient).
output	if output = TRUE write a output to an Excel 2007.

Details

The Ornstein-Uhlenbeck or Vasicek process is the unique solution to the following stochastic differential equation :

$$dX(t) = -r * X(t) * dt + sigma * dW(t)$$

With -r * X(t) :drift coefficient and sigma : diffusion coefficient, W(t) is Wiener process, the discretization dt = (T-t0)/N.

Please note that the process is stationary only if r > 0.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`OU` Ornstein-Uhlenbeck Process, `PEOU` Parametric Estimation of Ornstein-Uhlenbeck Model, `PEOUexp` Explicit Estimators of Ornstein-Uhlenbeck Model, `snssde` Simulation Numerical Solution of SDE.

Examples

```
## Flow of Ornstein-Uhlenbeck Process
## dX(t) = -2 * X(t) * dt + 1 *dW(t)
OUF(N=1000,M=5,t0=0,T=1,x0=10,r=2,sigma=1)
```

PDP *Creating Pearson Diffusions Process (by Milstein Scheme)*

Description

Simulation the pearson diffusions process by milstein scheme.

Usage

```
PDP(N, M, t0, T, x0, theta, mu, a, b, c, output = FALSE)
```

Arguments

`N`	size of process.
`M`	number of trajectories.
`t0`	initial time.
`T`	final time.
`x0`	initial value of the process at time `t0`.
`theta`	constant positive.
`mu`	constant.
`a`	constant.
`b`	constant.
`c`	constant.
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

A class that further generalizes the Ornstein-Uhlenbeck and Cox-Ingersoll-Ross processes is the class of Pearson diffusion, the pearson diffusions process is the solution to the stochastic differential equation :

$$dX(t) = -theta * (X(t) - mu) * dt + sqrt(2 * theta * (a * X(t)^2 + b * X(t) + c)) * dW(t)$$

With `-theta *(X(t)-mu) :drift coefficient` and `sqrt( 2*theta*(a*X(t)^2 + b *X(t)+ c)) :diffusion coefficient`, `W(t)` is Wiener process, discretization `dt = (T-t0)/N`.

With `theta > 0` and `a`, `b`, and `c` such that the diffusion coefficient is well-defined i.e., the square root can be extracted for all the values of the state space of `X(t)`.

1. When the `diffusion coefficient = sqrt(2*theta*c)` i.e, `(a=0,b=0)`, we recover the Ornstein-Uhlenbeck process.
2. For `diffusion coefficient = sqrt(2*theta*X(t))` and `0 < mu <= 1` i.e, `(a=0,b=1,c=0)`, we obtain the Cox-Ingersoll-Ross process, and if `mu > 1` the invariant distribution is a `Gamma law` with scale parameter `1` and shape parameter `mu`.
3. For `a > 0` and `diffusion coefficient = sqrt(2*theta*a*(X(t)^2+1))` i.e, `(b=0,c=a)`, the invariant distribution always exists on the real line, and for `mu = 0` the invariant distribution is a scaled `t distribution` with `v=(1+a^(-1))` degrees of freedom and scale parameter `v^(-0.5)`, while for `mu =! 0` the distribution is a form of skewed `t distribution` that is called Pearson type `IV` distribution.
4. For `a > 0`, `mu > 0`, and `diffusion coefficient = sqrt(2*theta*a*X(t)^2)` i.e, `(b=0,c=0)`, the distribution is defined on the positive half line and it is an `inverse Gamma distribution` with shape parameter `1 + a^-1` and scale parameter `a/mu`.
5. For `a > 0`, `mu >= a`, and `diffusion coefficient = sqrt(2*theta*a*X(t)*(X(t)+1))` i.e, `(b=a,c=0)`, the invariant distribution is the scaled `F distribution` with `(2*mu)/a` and `(2/a)+2` degrees of freedom and scale parameter `mu / (a+1)`. For `0 < mu < 1`, some reflecting conditions on the boundaries are also needed.
6. If `a < 0` and `mu > 0` are such that `min(mu,1-mu) >= -a` and `diffusion coefficient = sqrt(2*theta*a*X(t)*(X(t)-1))` i.e, `(b=-a,c=0)`, the invariant distribution exists on the interval `[0,1]` and is a `Beta distribution` with parameters `-mu/a` and `(mu-1)/a`.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`CEV` Constant Elasticity of Variance Models, `CIR` Cox-Ingersoll-Ross Models, `CIRhy` modified CIR and hyperbolic Process, `CKLS` Chan-Karolyi-Longstaff-Sanders Models, `DWP` Double-Well Potential Model, `GBM` Model of Black-Scholes, `HWV` Hull-White/Vasicek Models, `INFSR` Inverse of Feller s Square Root models, `JDP` Jacobi Diffusion Process, `ROU` Radial Ornstein-Uhlenbeck Process, `diffBridge` Diffusion Bridge Models, `snssde` Simulation Numerical Solution of SDE.

Examples

```
## example 1
## theta = 5, mu = 10, (a=0,b=0,c=0.5)
## dX(t) = -5 *(X(t)-10)*dt + sqrt( 2*5*0.5)* dW(t)
PDP(N=1000,M=1,T=1,t0=0,x0=1,theta=5,mu=10,a=0,b=0,c=0.5)

## example 2
## theta = 0.1, mu = 0.25, (a=0,b=1,c=0)
## dX(t) = -0.1 *(X(t)-0.25)*dt + sqrt( 2*0.1*X(t))* dW(t)
PDP(N=1000,M=1,T=1,t0=0,x0=1,theta=0.1,mu=0.25,a=0,b=1,c=0)

## example 3
## theta = 0.1, mu = 1, (a=2,b=0,c=2)
## dX(t) = -0.1*(X(t)-1)*dt + sqrt( 2*0.1*(2*X(t)^2+2))* dW(t)
PDP(N=1000,M=1,T=1,t0=0,x0=1,theta=0.1,mu=1,a=2,b=0,c=2)

## example 4
## theta = 0.1, mu = 1, (a=2,b=0,c=0)
## dX(t) = -0.1*(X(t)-1)*dt + sqrt( 2*0.1*2*X(t)^2)* dW(t)
PDP(N=1000,M=1,T=1,t0=0,x0=1,theta=0.1,mu=1,a=2,b=0,c=0)

## example 5
## theta = 0.1, mu = 3, (a=2,b=2,c=0)
## dX(t) = -0.1*(X(t)-3)*dt + sqrt( 2*0.1*(2*X(t)^2+2*X(t)))* dW(t)
PDP(N=1000,M=1,T=1,t0=0,x0=0.1,theta=0.1,mu=3,a=2,b=2,c=0)

## example 6
## theta = 0.1, mu = 0.5, (a=-1,b=1,c=0)
## dX(t) = -0.1*(X(t)-0.5)*dt + sqrt( 2*0.1*(-X(t)^2+X(t)))* dW(t)
PDP(N=1000,M=1,T=1,t0=0,x0=0.1,theta=0.1,mu=0.5,a=-1,b=1,c=0)
```

PEABM — *Parametric Estimation of Arithmetic Brownian Motion(Exact likelihood inference)*

Description

Parametric estimation of Arithmetic Brownian Motion

Usage

```
PEABM(X, delta, starts = list(theta= 1, sigma= 1), leve = 0.95)
```

Arguments

X	a numeric vector of the observed time-series values.
delta	the fraction of the sampling period between successive observations.
starts	named list. Initial values for optimizer.
leve	the confidence level required.

Details

This process solves the stochastic differential equation :

$$dX(t) = theta * dt + sigma * dW(t)$$

The conditional density `p(t,.|x)` is the density of a `Gaussian law` with `mean = x0 + theta * t` and `variance = sigma^2 * t`.

`R` has the `[dqpr]norm` functions to evaluate the density, the quantiles, and the cumulative distribution or generate pseudo random numbers from the normal distribution.

Value

`coef`	Coefficients extracted from the model.
`AIC`	A numeric value with the corresponding AIC.
`vcov`	A matrix of the estimated covariances between the parameter estimates in the linear or non-linear predictor of the model.
`confint`	A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labelled as (1-level)/2 and 1 - (1-level)/2.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`PEOU` Parametric Estimation of Ornstein-Uhlenbeck Model, `PEOUexp` Explicit Estimators of Ornstein-Uhlenbeck Model, `PEOUG` Parametric Estimation of Hull-White/Vasicek Models, `PEBS` Parametric Estimation of model Black-Scholes.

Examples

```
## Parametric estimation of Arithmetic Brownian Motion.
## t0 = 0 ,T = 100
data(DATA3)
res <- PEABM(DATA3,delta=0.1,starts=list(theta=1,sigma=1),leve = 0.95)
res
ABMF(N=1000,M=10,t0=0,T=100,x0=DATA3[1],theta=res$coef[1],sigma=res$coef[2])
points(seq(0,100,length=length(DATA3)),DATA3,type="l",lwd=3,col="red")
```

`PEBS` *Parametric Estimation of Model Black-Scholes (Exact likelihood inference)*

Description

Parametric estimation of model Black-Scholes.

Usage

```
PEBS(X, delta, starts = list(theta= 1, sigma= 1), leve = 0.95)
```

Arguments

X	a numeric vector of the observed time-series values.
delta	the fraction of the sampling period between successive observations.
starts	named list. Initial values for optimizer.
leve	the confidence level required.

Details

The Black and Scholes, or geometric Brownian motion model solves the stochastic differential equation:

$$dX(t) = theta * X(t) * dt + sigma * X(t) * dW(t)$$

The conditional density function `p(t,.|x)` is `log-normal` with `mean = x * exp(theta*t)` and `variance = x^2 * exp(2*theta*t)*(exp(sigma^2 *t ) -1 ).`

`R` has the `[dqpr]lnorm` functions to evaluate the density, the quantiles, and the cumulative distribution or generate pseudo random numbers from the lognormal distribution.

Value

coef	Coefficients extracted from the model.
AIC	A numeric value with the corresponding AIC.
vcov	A matrix of the estimated covariances between the parameter estimates in the linear or non-linear predictor of the model.
confint	A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labelled as (1-level)/2 and 1 - (1-level)/2.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`PEABM` Parametric Estimation of Arithmetic Brownian Motion, `PEOU` Parametric Estimation of Ornstein-Uhlenbeck Model, `PEOUexp` Explicit Estimators of Ornstein-Uhlenbeck Model, `PEOUG` Parametric Estimation of Hull-White/Vasicek Models.

Examples

```
## Parametric estimation of model Black-Scholes.
## t0 = 0 ,T = 1
data(DATA2)
res <- PEBS(DATA2,delta=0.001,starts=list(theta=2,sigma=1))
res
GBMF(N=1000,M=10,T=1,t0=0,x0=DATA2[1],theta=res$coef[1],sigma=res$coef[2])
points(seq(0,1,length=length(DATA2)),DATA2,type="l",lwd=3,col="red")
```

PEOU *Parametric Estimation of Ornstein-Uhlenbeck Model (Exact likelihood inference)*

Description

Parametric estimation of Ornstein-Uhlenbeck Model.

Usage

```
PEOU(X, delta, starts = list(r= 1, sigma= 1), leve = 0.95)
```

Arguments

`X`	a numeric vector of the observed time-series values.
`delta`	the fraction of the sampling period between successive observations.
`starts`	named list. Initial values for optimizer.
`leve`	the confidence level required.

Details

This process solves the stochastic differential equation :

$$dX(t) = -r * X(t) * dt + sigma * dW(t)$$

It is ergodic for `r > 0`. We have also shown its exact conditional and stationary densities. In particular, the conditional density `p(t,.|x)` is the density of a `Gaussian law` with `mean = x0 * exp(-r*t)` and `variance = ((sigma^2)/(2*r))*(1-exp(-2*r*t))`.

`R` has the `[dqpr]norm` functions to evaluate the density, the quantiles, and the cumulative distribution or generate pseudo random numbers from the normal distribution.

Value

`coef`	Coefficients extracted from the model.
`AIC`	A numeric value with the corresponding AIC.
`vcov`	A matrix of the estimated covariances between the parameter estimates in the linear or non-linear predictor of the model.
`confint`	A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labelled as (1-level)/2 and 1 - (1-level)/2.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`PEABM` Parametric Estimation of Arithmetic Brownian Motion, `PEOUexp` Explicit Estimators of Ornstein-Uhlenbeck Model, `PEOUG` Parametric Estimation of Hull-White/Vasicek Models, `PEBS` Parametric Estimation of model Black-Scholes.

Examples

```
## Parametric estimation of Ornstein-Uhlenbeck Model.
## t0 = 0 ,T = 10
data(DATA1)
res <- PEOU(DATA1,delta=0.01,starts=list(r=2,sigma=1),leve = 0.90)
res
OUF(N=1000,M=10,t0=0,T=10,x0=40,r=0.1979284,sigma=3.972637)
points(seq(0,10,length=length(DATA1)),DATA1,type="l",lwd=3,col="red")
```

PEOUexp — *Parametric Estimation of Ornstein-Uhlenbeck Model (Explicit Estimators)*

Description

Explicit estimators of Ornstein-Uhlenbeck Model.

Usage

```
PEOUexp(X, delta)
```

Arguments

`X`	a numeric vector of the observed time-series values.
`delta`	the fraction of the sampling period between successive observations.

Details

This process solves the stochastic differential equation :

$$dX(t) = -r * X(t) * dt + sigma * dW(t)$$

It is ergodic for `r > 0`.

We have also shown its exact conditional and stationary densities. In particular, the conditional density `p(t,.|x)` is the density of a `Gaussian law` with `mean = x0 * exp(-r*t)` and `variance = ((sigma^2)/(2*r))*(1-exp(-2*r*t))`, the maximum likelihood estimator of `r` is available in explicit form and takes the form :

$$r = -(1/dt) * log(sum(X(t) * X(t-1))/sum(X(t-1)^2))$$

which is defined only if `sum(X(t)*X(t-1)) > 0`, this estimator is consistent and asymptotically Gaussian.

The maximum likelihood estimator of :

$$sigma^2 = (2 * r)/(N * (1 - exp(-2 * dt * r))) * sum(X(t) - X(t-1) * *exp(-dt * r))^2$$

Value

`r`	Estimator of speed of reversion.
`sigma`	Estimator of volatility.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

PEABM Parametric Estimation of Arithmetic Brownian Motion, PEOU Parametric Estimation of Ornstein-Uhlenbeck Model, PEOUG Parametric Estimation of Hull-White/Vasicek Models, PEBS Parametric Estimation of model Black-Scholes.

Examples

```
## t0 = 0 ,T = 10
data(DATA1)
res <- PEOUexp(DATA1,delt=0.01)
res
OUF(N=1000,M=10,t0=0,T=10,x0=DATA1[1],r=res$r,sigma=res$sigma)
points(seq(0,10,length=length(DATA1)),DATA1,type="l",lwd=3,col="red")
```

PEOUG — *Parametric Estimation of Hull-White/Vasicek (HWV) Gaussian Diffusion Models(Exact likelihood inference)*

Description

Parametric estimation of Hull-White/Vasicek Model.

Usage

```
PEOUG(X, delta, starts = list(r= 1, theta= 1, sigma= 1), leve = 0.95)
```

Arguments

X	a numeric vector of the observed time-series values.
delta	the fraction of the sampling period between successive observations.
starts	named list. Initial values for optimizer.
leve	the confidence level required.

Details

the Vasicek or Ornstein-Uhlenbeck model solves the stochastic differential equation :

$$dX(t) = r * (theta - X(t)) * dt + sigma * dW(t)$$

It is ergodic for `r > 0`. We have also shown its exact conditional and stationary densities. In particular, the conditional density `p(t,.|x)` is the density of a `Gaussian law` with `mean = theta+(x0-theta)*exp(-r*t)` and `variance = (sigma^2/(2*r))*(1-exp(-2*r*t))`.

`R` has the `[dqpr]norm` functions to evaluate the density, the quantiles, and the cumulative distribution or generate pseudo random numbers from the normal distribution.

Value

coef	Coefficients extracted from the model.
AIC	A numeric value with the corresponding AIC.
vcov	A matrix of the estimated covariances between the parameter estimates in the linear or non-linear predictor of the model.
confint	A matrix (or vector) with columns giving lower and upper confidence limits for each parameter. These will be labelled as (1-level)/2 and 1 - (1-level)/2.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

PEABM Parametric Estimation of Arithmetic Brownian Motion, PEOUexp Explicit Estimators of Ornstein-Uhlenbeck Model, PEOU Parametric Estimation of Ornstein-Uhlenbeck Model, PEBS Parametric Estimation of model Black-Scholes.

Examples

```
## example 1
## t0 = 0 ,T = 10
data(DATA1)
res <- PEOUG(DATA1,delta=0.01,starts=list(r=2,theta=0,sigma=1))
res
HWVF(N=1000,M=10,t0=0,T=10,x0=40,r=0.9979465,theta=16.49602,sigma=3.963486)
points(seq(0,10,length=length(DATA1)),DATA1,type="l",lwd=3,col="red")
```

PredCorr *Predictor-Corrector Method For One-Dimensional SDE*

Description

Predictor-Corrector method of simulation numerical solution of one dimensional stochastic differential equation.

Usage

```
PredCorr(N, M, T = 1, t0, x0, Dt, alpha = 0.5,
         mu = 0.5, drift, diffusion, output = FALSE)
```

Arguments

N	size of process.
M	number of trajectories.
T	final time.
t0	initial time.
x0	initial value of the process at time t0.
Dt	time step of the simulation (discretization).

alpha	weight alpha of the predictor-corrector scheme.
mu	weight mu of the predictor-corrector scheme.
drift	drift coefficient: an expression of two variables t and x.
diffusion	diffusion coefficient: an expression of two variables t and x.
output	if output = TRUE write a output to an Excel 2007.

Details

The function returns a trajectory of the process; i.e., x0 and the new N simulated values if M = 1. For M > 1, an mts (multidimensional trajectories) is returned, which means that M independent trajectories are simulated. If Dt is not specified, then Dt = (T-t0)/N. If Dt is specified, then N values of the solution of the sde are generated and the time horizon T is adjusted to be T = N * Dt.

The method we present here just tries to approximate the states of the process first. This method is of weak convergence order 1.

The predictor-corrector algorithm is as follows. First consider the simple approximation (the predictor), Then choose two weighting coefficients alpha and mu in [0,1] and calculate the corrector.

Value

data.frame(time,x) and plot of process.

Note

- Note that the predictor-corrector method falls back to the standard Euler method for alpha = mu = 0.
- The function by default implements the predictor corrector method with alpha = mu = 0.5.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

diffBridge Creating Diffusion Bridge Models.snssde numerical solution of one-dimensional SDE . snssde2D numerical solution of two-dimensional SDE. PredCorr2D predictor-corrector method for two-dimensional SDE.

Examples

```
## example 1
## Hull-White/Vasicek Model
## T = 1 , t0 = 0 and N = 1000 ===> Dt = 0.001
drift     <- expression( (3*(2-x)) )
diffusion <- expression( (2) )
PredCorr(N=1000, M=1, T = 1, t0=0, x0=10, Dt=0.001, alpha = 0.5,
         mu = 0.5, drift, diffusion, output = FALSE)
## Multiple trajectories of the OU process by Euler Scheme
PredCorr(N=1000,M=5,T=1,t0=0,x0=10,Dt=0.001,alpha = 0.5,
         mu = 0.5,drift,diffusion,output=FALSE)
```

PredCorr2D *Predictor-Corrector Method For Two-Dimensional SDE*

Description

Predictor-Corrector method of simulation numerical solution of Two dimensional stochastic differential equation.

Usage

```
PredCorr2D(N, T = 1, t0, x0, y0, Dt, alpha = 0.5, mu = 0.5, driftx,
           drifty, diffx, diffy, Step = FALSE, Output = FALSE)
```

Arguments

N	size of process.
T	final time.
t0	initial time.
x0	initial value of the process X(t) at time t0.
y0	initial value of the process Y(t) at time t0.
Dt	time step of the simulation (discretization).
alpha	weight alpha of the predictor-corrector scheme.
mu	weight mu of the predictor-corrector scheme.
driftx	drift coefficient of process X(t): an expression of three variables t , x and y.
drifty	drift coefficient of process Y(t): an expression of three variables t , x and y.
diffx	diffusion coefficient of process X(t): an expression of three variables t , x and y.
diffy	diffusion coefficient of process Y(t): an expression of three variables t , x and y.
Step	if Step = TRUE ploting step by step.
Output	if output = TRUE write a output to an Excel 2007.

Details

the system for stochastic differential equation Two dimensional is :

$$dX(t) = ax(t, X(t), Y(t)) * dt + bx(t, X(t), Y(t)) * dWx(t)$$

$$dY(t) = ay(t, X(t), Y(t)) * dt + by(t, X(t), Y(t)) * dWy(t)$$

with driftx=ax(t,X(t),Y(t)), drifty=ay(t,X(t),Y(t)) and diffx=bx(t,X(t),Y(t)), diffy=by(t,X(t),Y(t)).

The method we present here just tries to approximate the states of the process first. This method is of weak convergence order 1. dW1(t) and dW2(t) are brownian motions independent.

The predictor-corrector algorithm is as follows. First consider the simple approximation (the predictor), Then choose two weighting coefficients alpha and mu in [0,1] and calculate the corrector.

Value

data.frame(time,X(t),Y(t)) and plot of process 2-D.

Note

- Note that the predictor-corrector method falls back to the standard Euler method for `alpha = mu = 0`.
- The function by default implements the predictor corrector method with `alpha = mu = 0.5`.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`diffBridge` Creating Diffusion Bridge Models. `snssde` numerical solution of one-dimensional SDE. `snssde2D` numerical solution of Two-dimensional SDE. `PredCorr` predictor-corrector method for one-dimensional SDE.

Examples

```
## Example 1
driftx <- expression(cos(t*x*y))
drifty <- expression(cos(t))
diffx <- expression(0.1)
diffy <- expression(0.1)
PredCorr2D(N=5000, T = 1, t0=0, x0=0, y0=0, Dt=0.001, alpha = 0.5,
           mu = 0.5, driftx, drifty, diffx, diffy, Step = FALSE,
           Output = FALSE)
## ploting Step by Step
##PredCorr2D(N=5000, T = 1, t0=0, x0=0, y0=0, Dt=0.001, alpha = 0.5,
##           mu = 0.5, driftx, drifty, diffx, diffy, Step = TRUE,
##           Output = FALSE)

## Example 2
## BM 2-D
driftx <- expression(0)
drifty <- expression(0)
diffx <- expression(1)
diffy <- expression(1)
PredCorr2D(N=5000, T = 1, t0=0, x0=0, y0=0, Dt=0.001, alpha = 0.5,
           mu = 0.5, driftx, drifty, diffx, diffy, Step = FALSE,
           Output = FALSE)
## ploting Step by Step
##PredCorr2D(N=5000, T = 1, t0=0, x0=0, y0=0, Dt=0.001, alpha = 0.5,
##           mu = 0.5, driftx, drifty, diffx, diffy, Step = TRUE,
##           Output = FALSE)


## Example 3
driftx <- expression(0.03*t*x-x^3)
drifty <- expression(0.03*t*y-y^3)
diffx <- expression(0.1)
diffy <- expression(0.1)
```

```
PredCorr2D(N=5000, T = 1, t0=0, x0=0, y0=0, Dt=0.001, alpha = 0.5,
           mu = 0.5, driftx, drifty, diffx, diffy, Step = FALSE,
           Output = FALSE)
## ploting Step by Step
## PredCorr2D(N=5000, T = 1, t0=0, x0=0, y0=0, Dt=0.001, alpha = 0.5,
##            mu = 0.5, driftx, drifty, diffx, diffy, Step = FALSE,
##            Output = FALSE)
```

RadialP2D_1 *Two-Dimensional Attractive Model Model(S = 1,Sigma)*

Description

Simulation 2-dimensional attractive model (S = 1).

Usage

```
RadialP2D_1(N, t0, Dt, T = 1, X0, Y0, v, K, Sigma, Output = FALSE)
```

Arguments

N	size of process.
t0	initial time.
Dt	time step of the simulation (discretization).
T	final time.
X0	initial value of the process X(t) at time t0.
Y0	initial value of the process Y(t) at time t0.
v	threshold. 0 < v < sqrt(X0^2 + Y0 ^2)
K	constant K > 0.
Sigma	constant Sigma > 0.
Output	if Output = TRUE write a Output to an Excel 2007.

Details

The attractive models is defined by the system for stochastic differential equation Two-dimensional :

$$dX(t) = (-K * X(t)/(sqrt(X(t)^2 + Y(t)^2))^(S + 1)) * dt + Sigma * dW1(t)$$

$$dY(t) = (-K * Y(t)/(sqrt(X(t)^2 + Y(t)^2))^(S + 1)) * dt + Sigma * dW2(t)$$

dW1(t) and dW2(t) are brownian motions independent.

If S = 1 (ie M(S=1,Sigma)) the system SDE is :

$$dX(t) = (-K * X(t)/(X(t)^2 + Y(t)^2)) * dt + Sigma * dW1(t)$$

$$dY(t) = (-K * Y(t)/(X(t)^2 + Y(t)^2)) * dt + Sigma * dW2(t)$$

For more detail consulted References.

Value

data.frame(time,X(t),Y(t)) and plot of process 2-D.

Note

- `2*K > Sigma^2`.

Author(s)

boukhetala Kamal, guidoum Arsalane.

References

1. K.Boukhetala, Estimation of the first passage time distribution for a simulated diffusion process, Maghreb Math.Rev, Vol.7, No 1, Jun 1998, pp. 1-25.
2. K.Boukhetala, Simulation study of a dispersion about an attractive centre. In proceedings of 11th Symposium Computational Statistics, edited by R.Dutter and W.Grossman, Wien , Austria, 1994, pp. 128-130.
3. K.Boukhetala,Modelling and simulation of a dispersion pollutant with attractive centre, Edited by Computational Mechanics Publications, Southampton ,U.K and Computational Mechanics Inc, Boston, USA, pp. 245-252.
4. K.Boukhetala, Kernel density of the exit time in a simulated diffusion, les Annales Maghrebines De L ingenieur, Vol , 12, N Hors Serie. Novembre 1998, Tome II, pp 587-589.

See Also

`snssde2D`, `PredCorr2D`, `RadialP2D_1PC`, `RadialP3D_1`, `tho_M1`, `fctgeneral`, `hist_general`, `Kern_meth`.

Examples

```
RadialP2D_1(N=1000, t0=0, Dt=0.001, T = 1, X0=2, Y0=1, v=0.3,
            K=3, Sigma=0.2, Output = FALSE)
```

`RadialP2D_1PC` *Two-Dimensional Attractive Model in Polar Coordinates Model(S = 1,Sigma)*

Description

Simulation 2-dimensional attractive model (S = 1) in polar coordinates.

Usage

```
RadialP2D_1PC(N, R0, t0, T, ThetaMax, K, sigma, output = FALSE)
```

Arguments

N	size of process.
R0	initial valueR0 > 0 at time t0.
t0	initial time.
T	final time.
ThetaMax	polar coordinates, example ThetaMax = 2*pi.
K	constant K > 0.
sigma	constant sigma > 0.
output	if Output = TRUE write a Output to an Excel 2007.

Details

The attractive models is defined by the system for stochastic differential equation Two-dimensional :

$$dX(t) = (-K * X(t)/(sqrt(X(t)^2 + Y(t)^2))^(S + 1)) * dt + Sigma * dW1(t)$$

$$dY(t) = (-K * Y(t)/(sqrt(X(t)^2 + Y(t)^2))^(S + 1)) * dt + Sigma * dW2(t)$$

dW1(t) and dW2(t) are brownian motions independent.

Using Ito transform, it is shown that the Radial Process R(t) with R(t)=||(X(t),Y(t))|| is a markovian diffusion, solution of the stochastic differential equation one-dimensional:

$$dR(t) = ((0.5 * Sigma^2 * R(t)^(S - 1) - K)/R(t)^S) * dt + Sigma * dW(t)$$

If S = 1 (ie M(S=1,Sigma)) the R(t) is :

$$dR(t) = ((0.5 * Sigma^2 - K)/R(t)) * dt + Sigma * dW(t)$$

Where ||.|| is the Euclidean norm and dW(t) is a determined brownian motions.

R(t)=sqrt(X(t)^2 + Y(t)^2) it is distance between X(t) and Y(t), then X(t)=R(t)*cos(theta(t)) and Y(t)=R(t)*sin(theta(t)),

For more detail consulted References.

Value

data.frame(time,R(t),theta(t)) and plot of process 2-D in polar coordinates.

Note

- 2*K > Sigma^2.

Author(s)

boukhetala Kamal, guidoum Arsalane.

References

1. K.Boukhetala, Estimation of the first passage time distribution for a simulated diffusion process, Maghreb Math.Rev, Vol.7, No 1, Jun 1998, pp. 1-25.
2. K.Boukhetala, Simulation study of a dispersion about an attractive centre. In proceedings of 11th Symposium Computational Statistics, edited by R.Dutter and W.Grossman, Wien , Austria, 1994, pp. 128-130.
3. K.Boukhetala,Modelling and simulation of a dispersion pollutant with attractive centre, Edited by Computational Mechanics Publications, Southampton ,U.K and Computational Mechanics Inc, Boston, USA, pp. 245-252.
4. K.Boukhetala, Kernel density of the exit time in a simulated diffusion, les Annales Maghrebines De L ingenieur, Vol , 12, N Hors Serie. Novembre 1998, Tome II, pp 587-589.

See Also

`snssde2D`, `PredCorr2D`, `RadialP2D_2PC`, `RadialP3D_1`, `tho_M1`, `fctgeneral`, `hist_general`, `Kern_meth`.

Examples

```
RadialP2D_1PC(N=1000, R0=3, t0=0, T=1, ThetaMax=4*pi, K=2, sigma=1,
              output = FALSE)
```

RadialP2D_2 — *Two-Dimensional Attractive Model Model(S >= 2,Sigma)*

Description

Simulation 2-dimensional attractive model (S >= 2).

Usage

```
RadialP2D_2(N, t0, Dt, T = 1, X0, Y0, v, K, s, Sigma, Output = FALSE)
```

Arguments

`N`	size of process.
`t0`	initial time.
`Dt`	time step of the simulation (`discretization`).
`T`	final time.
`X0`	initial value of the process `X(t)` at time `t0`.
`Y0`	initial value of the process `Y(t)` at time `t0`.
`v`	threshold. `0 < v < sqrt(X0^2 + Y0 ^2)`
`K`	constant `K > 0`.
`s`	constant `s >= 2`.
`Sigma`	constant `Sigma > 0`.
`Output`	if `Output = TRUE` write a `Output` to an Excel 2007.

Details

The attractive models is defined by the system for stochastic differential equation Two-dimensional :

$$dX(t) = (-K * X(t)/(sqrt(X(t)^2 + Y(t)^2))^(S + 1)) * dt + Sigma * dW1(t)$$

$$dY(t) = (-K * Y(t)/(sqrt(X(t)^2 + Y(t)^2))^(S + 1)) * dt + Sigma * dW2(t)$$

`dW1(t)` and `dW2(t)` are brownian motions independent.

For more detail consulted `References`.

Value

data.frame(time,X(t),Y(t)) and plot of process 2-D.

Note

- `2*K > Sigma^2`.

Author(s)

boukhetala Kamal, guidoum Arsalane.

References

1. K.Boukhetala, Estimation of the first passage time distribution for a simulated diffusion process, Maghreb Math.Rev, Vol.7, No 1, Jun 1998, pp. 1-25.
2. K.Boukhetala, Simulation study of a dispersion about an attractive centre. In proceedings of 11th Symposium Computational Statistics, edited by R.Dutter and W.Grossman, Wien , Austria, 1994, pp. 128-130.
3. K.Boukhetala,Modelling and simulation of a dispersion pollutant with attractive centre, Edited by Computational Mechanics Publications, Southampton ,U.K and Computational Mechanics Inc, Boston, USA, pp. 245-252.
4. K.Boukhetala, Kernel density of the exit time in a simulated diffusion, les Annales Maghrebines De L ingenieur, Vol , 12, N Hors Serie. Novembre 1998, Tome II, pp 587-589.

See Also

`snssde2D`, `PredCorr2D`, `RadialP2D_1PC`, `RadialP3D_1`, `tho_M1`, `fctgeneral`, `hist_general`, `Kern_meth`.

Examples

```
RadialP2D_2(N=1000, t0=0, Dt=0.001, T = 1, X0=2, Y0=3, v=0.5, K=16,
             s=2,Sigma=0.2, Output = FALSE)
```

`RadialP2D_2PC` *Two-Dimensional Attractive Model in Polar Coordinates Model(S >= 2,Sigma)*

Description

Simulation 2-dimensional attractive model (S >= 2) in polar coordinates.

Usage

```
RadialP2D_2PC(N, R0, t0, T, ThetaMax, K, s, sigma, output = FALSE)
```

Arguments

`N`	size of process.
`R0`	initial value`R0 > 0` at time `t0`.
`t0`	initial time.
`T`	final time.
`ThetaMax`	polar coordinates, example `ThetaMax = 2*pi`.
`K`	constant `K > 0`.
`s`	constant `s >= 2`.
`sigma`	constant `sigma > 0`.
`output`	if `Output = TRUE` write a `Output` to an Excel 2007.

Details

see details `RadialP2D_1PC`, and for more detail consulted `References`.

Value

data.frame(time,R(t),theta(t)) and plot of process 2-D in polar coordinates.

Note

- `2*K > Sigma^2`.

Author(s)

boukhetala Kamal, guidoum Arsalane.

References

1. K.Boukhetala, Estimation of the first passage time distribution for a simulated diffusion process, Maghreb Math.Rev, Vol.7, No 1, Jun 1998, pp. 1-25.
2. K.Boukhetala, Simulation study of a dispersion about an attractive centre. In proceedings of 11th Symposium Computational Statistics, edited by R.Dutter and W.Grossman, Wien , Austria, 1994, pp. 128-130.

3. K.Boukhetala,Modelling and simulation of a dispersion pollutant with attractive centre, Edited by Computational Mechanics Publications, Southampton ,U.K and Computational Mechanics Inc, Boston, USA, pp. 245-252.
4. K.Boukhetala, Kernel density of the exit time in a simulated diffusion, les Annales Maghrebines De L ingenieur, Vol , 12, N Hors Serie. Novembre 1998, Tome II, pp 587-589.

See Also

snssde2D, PredCorr2D, RadialP2D_1PC, RadialP3D_1, tho_M1, fctgeneral, hist_general, Kern_meth.

Examples

```
RadialP2D_2PC(N=1000, R0=3, t0=0, T=1, ThetaMax=2*pi, K=2, s=2,
               sigma=0.2,output = FALSE)
```

RadialP3D_1 *Three-Dimensional Attractive Model Model(S = 1,Sigma)*

Description

Simulation 3-dimensional attractive model (S = 1).

Usage

```
RadialP3D_1(N, t0, Dt, T = 1, X0, Y0, Z0, v, K, Sigma,
             Output = FALSE)
```

Arguments

N	size of process.
t0	initial time.
Dt	time step of the simulation (discretization).
T	final time.
X0	initial value of the process X(t) at time t0.
Y0	initial value of the process Y(t) at time t0.
Z0	initial value of the process Z(t) at time t0.
v	threshold. 0 < v < sqrt(X0^2 + Y0 ^2 + Z0^2)
K	constant K > 0.
Sigma	constant Sigma > 0.
Output	if Output = TRUE write a Output to an Excel 2007.

Details

The attractive models is defined by the system for stochastic differential equation three-dimensional :

$$dX(t) = (-K * X(t)/(sqrt(X(t)^2 + Y(t)^2 + Z(t)^2))^(S + 1)) * dt + Sigma * dW1(t)$$

$$dY(t) = (-K * Y(t)/(sqrt(X(t)^2 + Y(t)^2 + Z(t)^2))^(S + 1)) * dt + Sigma * dW2(t)$$

$$dZ(t) = (-K * Z(t)/(sqrt(X(t)^2 + Y(t)^2 + Z(t)^2))^(S + 1)) * dt + Sigma * dW3(t)$$

`dW1(t)`, `dW2(t)` and `dW3(t)` are brownian motions independent.

If `S = 1` (ie `M(S=1,Sigma)`) the system SDE is :

$$dX(t) = (-K * X(t)/(X(t)^2 + Y(t)^2 + Z(t)^2)) * dt + Sigma * dW1(t)$$

$$dY(t) = (-K * Y(t)/(X(t)^2 + Y(t)^2 + Z(t)^2)) * dt + Sigma * dW2(t)$$

$$dZ(t) = (-K * Z(t)/(X(t)^2 + Y(t)^2 + Z(t)^2)) * dt + Sigma * dW3(t)$$

For more detail consulted `References`.

Value

data.frame(time,X(t),Y(t),Z(t)) and plot of process 3-D.

Note

- `2*K > Sigma^2`.

Author(s)

boukhetala Kamal, guidoum Arsalane.

References

1. K.Boukhetala, Estimation of the first passage time distribution for a simulated diffusion process, Maghreb Math.Rev, Vol.7, No 1, Jun 1998, pp. 1-25.
2. K.Boukhetala, Simulation study of a dispersion about an attractive centre. In proceedings of 11th Symposium Computational Statistics, edited by R.Dutter and W.Grossman, Wien , Austria, 1994, pp. 128-130.
3. K.Boukhetala,Modelling and simulation of a dispersion pollutant with attractive centre, Edited by Computational Mechanics Publications, Southampton ,U.K and Computational Mechanics Inc, Boston, USA, pp. 245-252.
4. K.Boukhetala, Kernel density of the exit time in a simulated diffusion, les Annales Maghrebines De L ingenieur, Vol , 12, N Hors Serie. Novembre 1998, Tome II, pp 587-589.

See Also

`RadialP3D_2`.

Examples

```
RadialP3D_1(N=1000, t0=0, Dt=0.001, T = 1, X0=1, Y0=0.5, Z0=0.5,
            v=0.2, K=3, Sigma=0.2, Output = FALSE)
```

`RadialP3D_2` *Three-Dimensional Attractive Model Model(S >= 2,Sigma)*

Description

Simulation 3-dimensional attractive model (S >= 2).

Usage

```
RadialP3D_2(N, t0, Dt, T = 1, X0, Y0, Z0, v, K, s, Sigma,
             Output = FALSE)
```

Arguments

`N`	size of process.
`t0`	initial time.
`Dt`	time step of the simulation (`discretization`).
`T`	final time.
`X0`	initial value of the process `X(t)` at time `t0`.
`Y0`	initial value of the process `Y(t)` at time `t0`.
`Z0`	initial value of the process `Z(t)` at time `t0`.
`v`	threshold. `0 < v < sqrt(X0^2 + Y0 ^2 + Z0^2)`
`K`	constant `K > 0`.
`s`	constant `s >= 2`.
`Sigma`	constant `Sigma > 0`.
`Output`	if `Output = TRUE` write a `Output` to an Excel 2007.

Details

The attractive models is defined by the system for stochastic differential equation three-dimensional :

$$dX(t) = (-K * X(t)/(sqrt(X(t)^2 + Y(t)^2 + Z(t)^2))^(S + 1)) * dt + Sigma * dW1(t)$$

$$dY(t) = (-K * Y(t)/(sqrt(X(t)^2 + Y(t)^2 + Z(t)^2))^(S + 1)) * dt + Sigma * dW2(t)$$

$$dZ(t) = (-K * Z(t)/(sqrt(X(t)^2 + Y(t)^2 + Z(t)^2))^(S + 1)) * dt + Sigma * dW3(t)$$

`dW1(t)`, `dW2(t)` and `dW3(t)` are brownian motions independent.

For more detail consulted `References`.

Value

data.frame(time,X(t),Y(t),Z(t)) and plot of process 3-D.

Note

- `2*K > Sigma^2`.

Author(s)

boukhetala Kamal, guidoum Arsalane.

References

1. K.Boukhetala, Estimation of the first passage time distribution for a simulated diffusion process, Maghreb Math.Rev, Vol.7, No 1, Jun 1998, pp. 1-25.
2. K.Boukhetala, Simulation study of a dispersion about an attractive centre. In proceedings of 11th Symposium Computational Statistics, edited by R.Dutter and W.Grossman, Wien , Austria, 1994, pp. 128-130.
3. K.Boukhetala,Modelling and simulation of a dispersion pollutant with attractive centre, Edited by Computational Mechanics Publications, Southampton ,U.K and Computational Mechanics Inc, Boston, USA, pp. 245-252.
4. K.Boukhetala, Kernel density of the exit time in a simulated diffusion, les Annales Maghrebines De L ingenieur, Vol , 12, N Hors Serie. Novembre 1998, Tome II, pp 587-589.

See Also

`RadialP3D_1`.

Examples

```
RadialP3D_2(N=1000, t0=0, Dt=0.001, T = 1, X0=1, Y0=0.5, Z0=0.5,
            v=0.2,K=3,s=2,Sigma=0.2, Output = FALSE)
```

`RadialP_1` *Radial Process Model(S = 1,Sigma) Or Attractive Model*

Description

Simulation the radial process one-dimensional (S = 1).

Usage

```
RadialP_1(N, t0, Dt, T = 1, R0, K, Sigma, Output = FALSE,
          Methods = c("Euler", "Milstein", "MilsteinS",
          "Ito-Taylor", "Heun", "RK3"), ...)
```

Arguments

`N`	size of process.
`t0`	initial time.
`Dt`	time step of the simulation (`discretization`).
`T`	final time.
`R0`	initial value of the process at time `t0` ,($R0 > 0$).

K	constant K > 0.
Sigma	constant Sigma > 0.
Output	if Output = TRUE write a Output to an Excel 2007.
Methods	method of simulation ,see details snssde.
...	

Details

The attractive models is defined by the system for stochastic differential equation two-dimensional :

$$dX(t) = (-K * X(t)/(sqrt(X(t)^2 + Y(t)^2))^(S + 1)) * dt + Sigma * dW1(t)$$

$$dY(t) = (-K * Y(t)/(sqrt(X(t)^2 + Y(t)^2))^(S + 1)) * dt + Sigma * dW2(t)$$

dW1(t) and dW2(t) are brownian motions independent.

Using Ito transform, it is shown that the Radial Process R(t) with R(t)=||(X(t),Y(t))|| is a markovian diffusion, solution of the stochastic differential equation one-dimensional:

$$dR(t) = ((0.5 * Sigma^2 * R(t)^(S - 1) - K)/R(t)^S) * dt + Sigma * dW(t)$$

If S = 1 (ie M(S=1,Sigma)) the R(t) is :

$$dR(t) = ((0.5 * Sigma^2 - K)/R(t)) * dt + Sigma * dW(t)$$

Where ||.|| is the Euclidean norm and dW(t) is a determined brownian motions.

For more detail consulted References.

Value

data.frame(time,R(t)) and plot of process R(t).

Note

- If methods is not specified, it is assumed to be the Euler Scheme.
- If T and t0 specified, the best discretization Dt = (T-t0)/N.
- 2*K > Sigma^2.

Author(s)

boukhetala Kamal, guidoum Arsalane.

References

1. K.Boukhetala, Estimation of the first passage time distribution for a simulated diffusion process, Maghreb Math.Rev, Vol.7, No 1, Jun 1998, pp. 1-25.
2. K.Boukhetala, Simulation study of a dispersion about an attractive centre. In proceedings of 11th Symposium Computational Statistics, edited by R.Dutter and W.Grossman, Wien , Austria, 1994, pp. 128-130.
3. K.Boukhetala,Modelling and simulation of a dispersion pollutant with attractive centre, Edited by Computational Mechanics Publications, Southampton ,U.K and Computational Mechanics Inc, Boston, USA, pp. 245-252.
4. K.Boukhetala, Kernel density of the exit time in a simulated diffusion, les Annales Maghrebines De L ingenieur, Vol , 12, N Hors Serie. Novembre 1998, Tome II, pp 587-589.

See Also

RadialP2D_1, RadialP2D_1PC, RadialP3D_1, tho_M1, fctgeneral, hist_general, Kern_meth.

Examples

```
## Example 1
RadialP_1(N=1000, t0=0, Dt=0.001, T = 1, R0=2, K=2,
          Sigma=0.5, Output = FALSE)
## Example 2
RadialP_1(N=1000, t0=0, Dt=0.001, T = 1, R0=3, K=3.5,
          Sigma=0.5, Output = FALSE,Methods="Heun")
```

RadialP_2 *Radial Process Model(S >= 2,Sigma) Or Attractive Model*

Description

Simulation the radial process one-dimensional (S >= 2).

Usage

```
RadialP_2(N, t0, Dt, T = 1, R0, K, s, Sigma, Output = FALSE,
          Methods = c("Euler", "Milstein", "MilsteinS",
          "Ito-Taylor", "Heun", "RK3"), ...)
```

Arguments

N	size of process.
t0	initial time.
Dt	time step of the simulation (discretization).
T	final time.
R0	initial value of the process at time t0 ,(R0 > 0).
K	constant K > 0.
s	constant s >= 2.
Sigma	constant Sigma > 0.
Output	if Output = TRUE write a Output to an Excel 2007.
Methods	method of simulation ,see details snssde.
...	

Details

The attractive models is defined by the system for stochastic differential equation two-dimensional :

$$dX(t) = (-K * X(t)/(sqrt(X(t)^2 + Y(t)^2))^(S + 1)) * dt + Sigma * dW1(t)$$

$$dY(t) = (-K * Y(t)/(sqrt(X(t)^2 + Y(t)^2))^(S + 1)) * dt + Sigma * dW2(t)$$

dW1(t) and dW2(t) are brownian motions independent.

Using Ito transform, it is shown that the `Radial Process R(t)` with `R(t)=||(X(t),Y(t))||` is a markovian diffusion, solution of the stochastic differential equation one-dimensional:

$$dR(t) = ((0.5 * Sigma^2 * R(t)^(S - 1) - K)/R(t)^S) * dt + Sigma * dW(t)$$

For more detail consulted `References`.

Value

data.frame(time,R(t)) and plot of process R(t).

Note

- If `methods` is not specified, it is assumed to be the `Euler Scheme`.
- If `T` and `t0` specified, the best discretization `Dt = (T-t0)/N`.
- `2*K > Sigma^2`.

Author(s)

boukhetala Kamal, guidoum Arsalane.

References

1. K.Boukhetala, Estimation of the first passage time distribution for a simulated diffusion process, Maghreb Math.Rev, Vol.7, No 1, Jun 1998, pp. 1-25.
2. K.Boukhetala, Simulation study of a dispersion about an attractive centre. In proceedings of 11th Symposium Computational Statistics, edited by R.Dutter and W.Grossman, Wien , Austria, 1994, pp. 128-130.
3. K.Boukhetala,Modelling and simulation of a dispersion pollutant with attractive centre, Edited by Computational Mechanics Publications, Southampton ,U.K and Computational Mechanics Inc, Boston, USA, pp. 245-252.
4. K.Boukhetala, Kernel density of the exit time in a simulated diffusion, les Annales Maghrebines De L ingenieur, Vol , 12, N Hors Serie. Novembre 1998, Tome II, pp 587-589.

See Also

`RadialP2D_2`, `RadialP2D_2PC`, `RadialP3D_2`, `tho_M2`, `fctgeneral`, `hist_general`, `Kern_meth`.

Examples

```
## Example 1
RadialP_2(N=1000, t0=0, Dt=0.001, T = 1, R0=2, K=2.5,s=2,
          Sigma=0.5, Output = FALSE)
## Example 2
RadialP_2(N=1000, t0=0, Dt=0.001, T = 1, R0=3, K=6,s=2,
          Sigma=0.5, Output = FALSE,Methods="Heun")
```

ROU *Creating Radial Ornstein-Uhlenbeck Process (by Milstein Scheme)*

Description

Simulation the radial ornstein-uhlenbeck process by milstein scheme.

Usage

```
ROU(N, M, t0, T, x0, theta, output = FALSE)
```

Arguments

`N`	size of process.
`M`	number of trajectories.
`t0`	initial time.
`T`	final time.
`x0`	initial value of the process at time `t0`.
`theta`	constant positive.
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Details

The radial Ornstein-Uhlenbeck process is the solution to the stochastic differential equation :

$$dX(t) = (theta * X(t)^-(1) - X(t)) * dt + dW(t)$$

With `(theta * X(t)^-1 - X(t)) :drift coefficient` and `1 :diffusion coefficient`, the discretization `dt = (T-t0)/N`, `W(t)` is Wiener process.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`CEV` Constant Elasticity of Variance Models, `CIR` Cox-Ingersoll-Ross Models, `CIRhy` modified CIR and hyperbolic Process, `CKLS` Chan-Karolyi-Longstaff-Sanders Models, `DWP` Double-Well Potential Model, `GBM` Model of Black-Scholes, `HWV` Hull-White/Vasicek Models, `INFSR` Inverse of Feller s Square Root models, `JDP` Jacobi Diffusion Process, `PDP` Pearson Diffusions Process, `diffBridge` Diffusion Bridge Models, `snssde` Simulation Numerical Solution of SDE.

Examples

```
## Radial Ornstein-Uhlenbeck
## dX(t) = (0.05*X(t)^(-1) - X(t)) *dt + dW(t)
## One trajectorie
ROU(N=1000,M=1,T=1,t0=0,x0=1,theta=0.05)
```

`showData` *Display a Data Frame in a Tk Text Widget*

Description

Show my data frame in Tk Text Widget.

Examples

```
##showData(data.frame(DATA1))
```

`snssde` *Numerical Solution of One-Dimensional SDE*

Description

Different methods of simulation of solutions to stochastic differential equations one-dimensional.

Usage

```
snssde(N, M, T = 1, t0, x0, Dt, drift, diffusion, Output = FALSE,
       Methods = c("SchEuler", "SchMilstein", "SchMilsteinS",
       "SchTaylor", "SchHeun", "SchRK3"), ...)
```

Arguments

`N`	size of process.
`M`	number of trajectories.
`T`	final time.
`t0`	initial time.
`x0`	initial value of the process at time `t0`.
`Dt`	time step of the simulation `(discretization)`.
`drift`	drift coefficient: an expression of two variables `t` and `x`.
`diffusion`	diffusion coefficient: an expression of two variables `t` and `x`.
`Output`	if `Output = TRUE` write a `Output` to an Excel 2007.
`Methods`	method of simulation ,`see details`.
`...`	

Details

The function `snssde` returns a trajectory of the process; i.e., `x0` and the new `N` simulated values if `M = 1`. For `M > 1`, an mts (multidimensional trajectories) is returned, which means that `M` independent trajectories are simulated. `Dt` the best discretization `Dt = (T-t0)/N`.

Simulation methods are usually based on discrete approximations of the continuous solution to a stochastic differential equation. The methods of approximation are classified according to their different properties. Mainly two criteria of optimality are used in the literature: the strong and the weak (orders of) convergence. The `methods` of simulation can be one among: `Euler Order 0.5` , `Milstein Order 1` , `Milstein Second-Order` , `Ito-Taylor Order 1.5` , `Heun Order 2` , `Runge-Kutta Order 3`.

Value

data.frame(time,x) and plot of process.

Note

- If `methods` is not specified, it is assumed to be the `Euler Scheme`.
- If `T` and `t0` specified, the best discretization `Dt = (T-t0)/N`.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`diffBridge` Creating Diffusion Bridge Models.`PredCorr` Predictor-Corrector Method for one-dimensional SDE. `snssde2D` numerical solution of two-dimensional SDE. `PredCorr2D` predictor-corrector method for two-dimensional SDE.

Examples

```
## example 1
## Hull-White/Vasicek Model
## T = 1 , t0 = 0 and N = 1000 ===> Dt = 0.001
drift     <- expression( (3*(2-x)) )
diffusion <- expression( (2) )
snssde(N=1000,M=1,T=1,t0=0,x0=10,Dt=0.001,
drift,diffusion,Output=FALSE)
## Multiple trajectories of the OU process by Euler Scheme
snssde(N=1000,M=5,T=1,t0=0,x0=10,Dt=0.001,
drift,diffusion,Output=FALSE)

## example 2
## Black-Scholes models
## T = 1 , t0 = 0 and N = 1000 ===> Dt = 0.001
drift     <- expression( (3*x) )
diffusion <- expression( (2*x) )
snssde(N=1000,M=1,T=1,t0=0,x0=10,Dt=0.001,drift,
diffusion,Output=FALSE,Methods="SchMilstein")

## example 3
## Constant Elasticity of Variance (CEV) Models
## T = 1 , t0 = 0 and N = 1000 ===> Dt = 0.001
drift     <- expression( (0.3*x) )
diffusion <- expression( (0.2*x^0.75) )
snssde(N=1000,M=1,T=1,t0=0,x0=1,Dt=0.001,drift,
diffusion,Output=FALSE,Methods="SchMilsteinS")

## example 4
## sde\ dX(t)=(0.03*t*X(t)-X(t)^3)*dt+0.1*dW(t)
## T = 100 , t0 = 0 and N = 1000 ===> Dt = 0.1
drift     <- expression( (0.03*t*x-x^3) )
diffusion <- expression( (0.1) )
snssde(N=1000,M=1,T=100,t0=0,x0=0,Dt=0.1,drift,
diffusion,Output=FALSE,Methods="SchTaylor")
```

```
## example 5
## sde\ dX(t)=cos(t*x)*dt+sin(t*x)*dW(t) by Heun Scheme
drift     <- expression( (cos(t*x)) )
diffusion <- expression( (sin(t*x)) )
snssde(N=1000,M=1,T=100,t0=0,x0=0,Dt=0.1,drift,
diffusion,Output=FALSE,Methods="SchHeun")

## example 6
## sde\ dX(t)=exp(t)*dt+tan(t)*dW(t) by Runge-Kutta Scheme
drift     <- expression( (exp(t)) )
diffusion <- expression( (tan(t)) )
snssde(N=1000,M=1,T=1,t0=0,x0=1,Dt=0.001,drift,
diffusion,Output=FALSE,Methods="SchRK3")
```

`snssde2D` *Numerical Solution of Two-Dimensional SDE*

Description

Different methods of simulation of solutions to stochastic differential equations Two-dimensional.

Usage

```
snssde2D(N, T = 1, t0, x0, y0, Dt, driftx, drifty, diffx, diffy,
         Step = FALSE, Output = FALSE, Methods = c("SchEuler",
         "SchMilstein", "SchMilsteinS", "SchTaylor", "SchHeun",
         "SchRK3"), ...)
```

Arguments

`N`	size of process.
`T`	final time.
`t0`	initial time.
`x0`	initial value of the process `X(t)` at time `t0`.
`y0`	initial value of the process `Y(t)` at time `t0`.
`Dt`	time step of the simulation (`discretization`).
`driftx`	drift coefficient of process `X(t)`: an expression of three variables `t` , `x` and `y`.
`drifty`	drift coefficient of process `Y(t)`: an expression of three variables `t` , `x` and `y`.
`diffx`	diffusion coefficient of process `X(t)`: an expression of three variables `t` , `x` and `y`.
`diffy`	diffusion coefficient of process `Y(t)`: an expression of three variables `t` , `x` and `y`.
`Step`	if `Step = TRUE` ploting step by step.
`Output`	if `output = TRUE` write a `output` to an Excel 2007.
`Methods`	method of simulation ,`see details`.
`...`	

Details

the system for stochastic differential equation Two dimensional is :

$$dX(t) = ax(t, X(t), Y(t)) * dt + bx(t, X(t), Y(t)) * dW1(t)$$

$$dY(t) = ay(t, X(t), Y(t)) * dt + by(t, X(t), Y(t)) * dW2(t)$$

with `driftx=ax(t,X(t),Y(t))`, `drifty=ay(t,X(t),Y(t))` and `diffx=bx(t,X(t),Y(t))`, `diffy=by(t,X(t),Y(t))`. `dW1(t)` and `dW2(t)` are brownian motions independent.

Simulation methods are usually based on discrete approximations of the continuous solution to a stochastic differential equation. The methods of approximation are classified according to their different properties. Mainly two criteria of optimality are used in the literature: the strong and the weak (orders of) convergence. The `methods` of simulation can be one among: `Euler Order 0.5` , `Milstein Order 1` , `Milstein Second-Order` , `Ito-Taylor Order 1.5` , `Heun Order 2` , `Runge-Kutta Order 3`.

Value

data.frame(time,X(t),Y(t)) and plot of process 2-D.

Note

- If `methods` is not specified, it is assumed to be the `Euler Scheme`.
- If `T` and `t0` specified, the best discretization `Dt = (T-t0)/N`.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`diffBridge` Creating Diffusion Bridge Models. `snssde` numerical solution of one-dimensional SDE. `snssde` numerical solution of one-dimensional SDE. `PredCorr` predictor-corrector method for one-dimensional SDE. `PredCorr2D` predictor-corrector method for Two-dimensional SDE.

Examples

```
## Example 1
driftx <- expression(cos(t*x))
drifty <- expression(cos(t*y))
diffx <- expression(sin(t*x))
diffy <- expression(sin(t*y))
snssde2D(N=1000, T = 1, t0=0, x0=0, y0=0, Dt=0.001, driftx,
         drifty, diffx, diffy, Step = FALSE, Output = FALSE,
         Methods="SchTaylor")

## Example 2
driftx <- expression(cos(t*x*y))
drifty <- expression(sin(t*y*y))
diffx <- expression(atan2(y, x))
diffy <- expression(atan2(y, x))
snssde2D(N=5000, T = 1, t0=0, x0=1, y0=1, Dt=0.001, driftx,
         drifty, diffx, diffy, Step = FALSE, Output = FALSE,
         Methods="SchHeun")
```

SRW *Creating Random Walk*

Description

Simulation random walk.

Usage

```
SRW(N, t0, T, p, output = FALSE)
```

Arguments

N	size of process.
t0	initial time.
T	final time.
p	probability of choosing `X = -1 or +1`.
output	if `output = TRUE` write a `output` to an Excel 2007.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`Stgamma` Stochastic Process The Gamma Distribution, `Stst` Stochastic Process The Student Distribution, `WNG` White Noise Gaussian.

Examples

```
## Random Walk
SRW(N=1000,t0=0,T=1,p=0.5)
SRW(N=1000,t0=0,T=1,p=0.25)
SRW(N=1000,t0=0,T=1,p=0.75)
```

Stgamma *Creating Stochastic Process The Gamma Distribution*

Description

Simulation stochastic process by a gamma distribution.

Usage

```
Stgamma(N, t0, T, alpha, beta, output = FALSE)
```

Arguments

N	size of process.
t0	initial time.
T	final time.
alpha	constant positive.
beta	an alternative way to specify the scale.
output	if `output = TRUE` write a `output` to an Excel 2007.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`SRW` Creating Random Walk, `Stst` Stochastic Process The Student Distribution, `WNG` White Noise Gaussian.

Examples

```
## Stochastic Process The Gamma Distribution
Stgamma(N=1000,t0=0,T=5,alpha=1,beta=1)
```

Stst — *Creating Stochastic Process The Student Distribution*

Description

Simulation stochastic process by a Student distribution.

Usage

```
Stst(N, t0, T, n, output = FALSE)
```

Arguments

`N`	size of process.
`t0`	initial time.
`T`	final time.
`n`	degrees of freedom (`> 0,non-integer`).
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`SRW` Creating Random Walk, `Stgamma` Stochastic Process The Gamma Distribution, `WNG` White Noise Gaussian.

Examples

```
## Stochastic Process The Student Distribution
Stst(N=1000,t0=0,T=1,n=2)
```

Telegproc — *Realization a Telegraphic Process*

Description

Simulation a telegraphic process.

Usage

```
Telegproc(t0, x0, T, lambda, output = FALSE)
```

Arguments

`t0`	initial time.
`x0`	state initial `(x0 = -1 or +1)`.
`T`	final time of the simulation.
`lambda`	exponential distribution with rate `lambda`.
`output`	if `output = TRUE` write a `output` to an Excel 2007.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`Asys` Evolution a Telegraphic Process.

Examples

```
## Simulation a telegraphic process
Telegproc(t0=0,x0=1,T=1,lambda=0.5)
```

`test_ks_dbeta` *Kolmogorov-Smirnov Tests (Beta Distribution)*

Description

Performs one sample Kolmogorov-Smirnov tests.

Usage

```
test_ks_dbeta(X, shape1, shape2)
```

Arguments

`X`	a numeric vector of data values.
`shape1`	positive parameters of the Beta distribution.
`shape2`	positive parameters of the Beta distribution.

Details

see detail `ks.test`.

Value

A list with class "htest" containing the following components:

`statistic`	the value of the test statistic.
`p.value`	the p-value of the test.
`alternative`	a character string describing the alternative hypothesis.
`data.name`	a character string giving the name(s) of the data.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

ks.test

Examples

```
X <- rbeta(1000,1,1)
test_ks_dbeta(X, shape1=1, shape2=1)
test_ks_dbeta(X, shape1=1, shape2=2)
```

---

test_ks_dchisq *Kolmogorov-Smirnov Tests (Chi-Squared Distribution)*

Description

Performs one sample Kolmogorov-Smirnov tests.

Usage

```
test_ks_dchisq(X, df)
```

Arguments

X	a numeric vector of data values.
df	degrees of freedom (non-negative, but can be non-integer).

Details

see detail ks.test.

Value

A list with class "htest" containing the following components:

statistic	the value of the test statistic.
p.value	the p-value of the test.
alternative	a character string describing the alternative hypothesis.
data.name	a character string giving the name(s) of the data.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

ks.test

Examples

```
X <- rchisq(1000,15)
test_ks_dchisq(X, df=5)
test_ks_dchisq(X, df=10)
test_ks_dchisq(X, df=15)
test_ks_dchisq(X, df=20)
```

---

test_ks_dexp — *Kolmogorov-Smirnov Tests (Exponential Distribution)*

Description

Performs one sample Kolmogorov-Smirnov tests.

Usage

```
test_ks_dexp(X, lambda)
```

Arguments

`X`	a numeric vector of data values.
`lambda`	vector of rates.

Details

see detail `ks.test`.

Value

A list with class "htest" containing the following components:

`statistic`	the value of the test statistic.
`p.value`	the p-value of the test.
`alternative`	a character string describing the alternative hypothesis.
`data.name`	a character string giving the name(s) of the data.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`ks.test`

Examples

```
## Example 1
X <- rexp(1000,c(1,2,3))
test_ks_dexp(X, lambda=1)
test_ks_dexp(X, lambda=2)
test_ks_dexp(X, lambda=3)
## Example 2
X <- rexp(1000,3)
test_ks_dexp(X, lambda=3)
test_ks_dweibull(X, shape=1, scale=(1/3))
test_ks_dgamma(X, shape=1, rate=3)
```

---

test_ks_df *Kolmogorov-Smirnov Tests (F Distribution)*

Description

Performs one sample Kolmogorov-Smirnov tests.

Usage

```
test_ks_df(X, df1, df2)
```

Arguments

X	a numeric vector of data values.
df1	degrees of freedom. Inf is allowed.
df2	degrees of freedom. Inf is allowed.

Details

see detail `ks.test`.

Value

A list with class "htest" containing the following components:

statistic	the value of the test statistic.
p.value	the p-value of the test.
alternative	a character string describing the alternative hypothesis.
data.name	a character string giving the name(s) of the data.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`ks.test`

Examples

```
X <- rf(1000,10,20)
test_ks_df(X, df1=10, df2=20)
test_ks_df(X, df1=5, df2=15)
test_ks_df(X, df1=15, df2=25)
```

---

test_ks_dgamma *Kolmogorov-Smirnov Tests (Gamma Distribution)*

Description

Performs one sample Kolmogorov-Smirnov tests.

Usage

```
test_ks_dgamma(X, shape, rate)
```

Arguments

X	a numeric vector of data values.
shape	shape parameters. Must be positive, scale strictly.
rate	an alternative way to specify the scale.

Details

see detail `ks.test`.

Value

A list with class "htest" containing the following components:

statistic	the value of the test statistic.
p.value	the p-value of the test.
alternative	a character string describing the alternative hypothesis.
data.name	a character string giving the name(s) of the data.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`ks.test`

Examples

```
X <- rgamma(1000,1,6)
test_ks_dgamma(X, shape=1, rate=6)
test_ks_dexp(X, lambda=6)
test_ks_dweibull(X, shape=1, scale=(1/6))
```

`test_ks_dlognorm` *Kolmogorov-Smirnov Tests (Log Normal Distribution)*

Description

Performs one sample Kolmogorov-Smirnov tests.

Usage

```
test_ks_dlognorm(X, meanlog, sdlog)
```

Arguments

`X`	a numeric vector of data values.
`meanlog`	mean of the distribution.
`sdlog`	standard deviation of the distribution.

Details

see detail `ks.test`.

Value

A list with class "htest" containing the following components:

`statistic`	the value of the test statistic.
`p.value`	the p-value of the test.
`alternative`	a character string describing the alternative hypothesis.
`data.name`	a character string giving the name(s) of the data.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`ks.test`

Examples

```
X <- rlnorm(1000,1,1)
test_ks_dlognorm(X, meanlog=1, sdlog=1)
test_ks_dnorm(log(X), mean=1, sd=1)
```

`test_ks_dnorm` *Kolmogorov-Smirnov Tests (Normal Distribution)*

Description

Performs one sample Kolmogorov-Smirnov tests.

Usage

```
test_ks_dnorm(X, mean, sd)
```

Arguments

`X`	a numeric vector of data values.
`mean`	mean of the distribution.
`sd`	standard deviation of the distribution.

Details

see detail `ks.test`.

Value

A list with class "htest" containing the following components:

`statistic`	the value of the test statistic.
`p.value`	the p-value of the test.
`alternative`	a character string describing the alternative hypothesis.
`data.name`	a character string giving the name(s) of the data.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`ks.test`

Examples

```
## Example 1
X <- rnorm(1000,1,1)
test_ks_dnorm(X, mean=1, sd=1)
test_ks_dlognorm(exp(X), meanlog=1, sdlog=1)
## Example 2
X = c(runif(100),rt(200,20),rnorm(200))
X = sample(X)
test_ks_dnorm(X, mean=mean(X), sd=sd(X))
```

`test_ks_dt` *Kolmogorov-Smirnov Tests (Student t Distribution)*

Description

Performs one sample Kolmogorov-Smirnov tests.

Usage

```
test_ks_dt(X, df)
```

Arguments

`X`	a numeric vector of data values.
`df`	degrees of freedom (> 0, maybe non-integer). `df = Inf` is allowed.

Details

see detail `ks.test`.

Value

A list with class "htest" containing the following components:

`statistic`	the value of the test statistic.
`p.value`	the p-value of the test.
`alternative`	a character string describing the alternative hypothesis.
`data.name`	a character string giving the name(s) of the data.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`ks.test`

Examples

```
X <- rt(1000,15)
test_ks_dt(X, df=15)
test_ks_dt(X, df=10)
```

test_ks_dweibull *Kolmogorov-Smirnov Tests (Weibull Distribution)*

Description

Performs one sample Kolmogorov-Smirnov tests.

Usage

```
test_ks_dweibull(X, shape, scale)
```

Arguments

X	a numeric vector of data values.
shape	shape and scale parameters, the latter defaulting to 1.
scale	shape and scale parameters, the latter defaulting to 1.

Details

see detail `ks.test`.

Value

A list with class "htest" containing the following components:

statistic	the value of the test statistic.
p.value	the p-value of the test.
alternative	a character string describing the alternative hypothesis.
data.name	a character string giving the name(s) of the data.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`ks.test`

Examples

```
X <- rweibull(1000,1,4)
test_ks_dweibull(X, shape=1, scale=4)
test_ks_dexp(X, lambda=0.25)
test_ks_dgamma(X, shape=1, rate=0.25)
```

`tho_02diff` *Simulation The First Passage Time FPT For Attractive Model for Two-Diffusion Processes V(1) and V(2)*

Description

simulation M-sample for the first passage time "FPT" for attractive for 2-diffusion processes V(1)=c(X1(t),X2(t)) and V(2)=c(Y1(t),Y2(t)) or V(1)=c(X1(t),X2(t),X3(t)) and V(2)=c(Y1(t),Y2(t),Y3(t)).

Usage

```
tho_02diff(N, M, t0, Dt, T = 1, X1_0, X2_0, Y1_0, Y2_0,
           v, K, m, Sigma,Output=FALSE)
```

Arguments

`N`	size of the diffusion process V1(t) and V2(t).
`M`	size of the FPT.
`t0`	initial time.
`Dt`	time step of the simulation (`discretization`).
`T`	final time.
`X1_0`	initial value of the process `X1(t)` at time `t0`.
`X2_0`	initial value of the process `X2(t)` at time `t0`.
`Y1_0`	initial value of the process `Y1(t)` at time `t0`.
`Y2_0`	initial value of the process `Y2(t)` at time `t0`.
`v`	threshold. `see detail`
`K`	constant `K > 0`.
`m`	constant `m > 0`.
`Sigma`	constant `Sigma > 0`.
`Output`	if `Output = TRUE` write a `Output` to an Excel 2007.

Details

The 2-dimensional attractive models for 2-diffusion processes V(1)=(X1(t),X2(t)) and V(2)=c(Y1(t),Y2(t)) is defined by the Two (02) system for stochastic differential equation Two-dimensional :

$$dV1(t) = dV2(t) + Mu(m+1)(||D(t)||) * D(t) * dt + SigmaI(2*2) * dW1(t)$$

$$dV2(t) = Sigma * I(2*2) * dW2(t)$$

with:

$$D(t) = V1(t) - V2(t)$$

$$Mu(m)(||d||) = -K/||d||^m$$

Where ||.|| is the Euclidean norm and I(2*2) is identity matrix, `dW1(t)` and `dW2(t)` are brownian motions independent.

`D(t)=sqrt((X1(t)^2 - Y1(t)^2)+(X2(t)^2 - Y2(t)^2))` it is distance between `V1(t)` and `V2(t)`.

And the random variable `tau` "first passage time FPT", is defined by :

$$tau(V1(t),V2(t)) = inf(t >= 0 \; ||D(t)|| <= v)$$

with `v is the threshold`.

Value

Random variable tau "FPT".

Note

- `2*K > Sigma^2`.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

TowDiffAtra3D, TowDiffAtra2D, fctgeneral, hist_general, Kern_meth, AnaSimFPT Simulation The First Passage Time FPT For A Simulated Diffusion Process.

Examples

```
tho_02diff(N=500, M=50, t0=0, Dt=0.001, T = 1,
           X1_0=1, X2_0=1,Y1_0=0.5, Y2_0=0.5,v=0.05,
           K=4, m=0.2, Sigma=0.2)
summary(FPT)
hist(FPT,prob=TRUE)
plot(density(FPT,kernel="gaussian"),col="red")
```

`tho_M1` *Simulation The First Passage Time FPT For Attractive Model(S = 1,Sigma)*

Description

simulation M-sample for the first passage time "FPT" for attractive model(S = 1,Sigma).

Usage

```
tho_M1(N, M, t0, T, R0, v, K, sigma, Output = FALSE,
       Methods = c("Euler", "Milstein", "MilsteinS",
                   "Ito-Taylor", "Heun", "RK3"), ...)
```

Arguments

`N`	size of the diffusion process.
`M`	size of the FPT.
`t0`	initial time.
`T`	final time.
`R0`	initial value of the process at time `t0` ,(R0 > 0).
`v`	threshold.`see detail`.
`K`	constant `K > 0`.
`sigma`	constant `Sigma > 0`.
`Output`	if `Output = TRUE` write a `Output` to an Excel 2007.
`Methods`	method of simulation ,see details `snssde`.
`...`	

Details

Using Ito transform, it is shown that the `Radial Process R(t)` with `R(t)=||(X(t),Y(t))||` is a markovian diffusion, solution of the stochastic differential equation one-dimensional:

$$dR(t) = ((0.5 * Sigma^2 - K)/R(t)) * dt + Sigma * dW(t)$$

We take interest in the random variable `FPT` "first passage time", is defined by :

$$FPT = inf(t >= 0\ R(t) <= v)$$

with `v is the threshold`.

For more detail consulted `References`.

Value

M-sample for FPT.

Note

- `2*K > Sigma^2`.

o system.time(tho_M1(N=1000, M=100, t0=0, T=1, R0=2, v=0.05, K=3, sigma=0.3,Output = FALSE))

utilisateur systeme ecoule

5.64 0.10 6.08

o system.time(tho_M1(N=1000, M=100, t0=0, T=1, R0=2, v=0.05, K=3, sigma=0.3,Output = FALSE,Methods="RK3"))

utilisateur systeme ecoule

29.78 0.25 29.93

Author(s)

boukhetala Kamal, guidoum Arsalane.

References

1. K.Boukhetala, Estimation of the first passage time distribution for a simulated diffusion process, Maghreb Math.Rev, Vol.7, No 1, Jun 1998, pp. 1-25.
2. K.Boukhetala, Simulation study of a dispersion about an attractive centre. In proceedings of 11th Symposium Computational Statistics, edited by R.Dutter and W.Grossman, Wien , Austria, 1994, pp. 128-130.
3. K.Boukhetala,Modelling and simulation of a dispersion pollutant with attractive centre, Edited by Computational Mechanics Publications, Southampton ,U.K and Computational Mechanics Inc, Boston, USA, pp. 245-252.
4. K.Boukhetala, Kernel density of the exit time in a simulated diffusion, les Annales Maghrebines De L ingenieur, Vol , 12, N Hors Serie. Novembre 1998, Tome II, pp 587-589.

See Also

`AnaSimFPT` Simulation The First Passage Time FPT For A Simulated Diffusion Process.

Examples

```
tho_M1(N=1000, M=50, t0=0, T=1, R0=2, v=0.05, K=3,
       sigma=0.3, Output = FALSE)
```

`tho_M2` *Simulation The First Passage Time FPT For Attractive Model(S >= 2,Sigma)*

Description

simulation M-sample for the first passage time "FPT" for attractive model(S >= 2,Sigma).

Usage

```
tho_M2(N, M, t0, T, R0, v, K, s, Sigma, Output = FALSE,
       Methods = c("Euler", "Milstein", "MilsteinS",
                   "Ito-Taylor", "Heun", "RK3"), ...)
```

Arguments

`N`	size of the diffusion process.
`M`	size of the FPT.
`t0`	initial time.
`T`	final time.
`R0`	initial value of the process at time `t0` ,(R0 > 0).
`v`	threshold. `see detail`.
`K`	constant `K > 0`.
`s`	constant `s >= 2`.
`Sigma`	constant `Sigma > 0`.
`Output`	if `Output = TRUE` write a `Output` to an Excel 2007.
`Methods`	method of simulation ,see details `snssde`.
`...`	

Details

Using Ito transform, it is shown that the `Radial Process R(t)` with `R(t)=||(X(t),Y(t))||` is a markovian diffusion, solution of the stochastic differential equation one-dimensional:

$$dR(t) = ((0.5 * Sigma^2 * R(t)^(S - 1) - K)/R(t)^S) * dt + Sigma * dW(t)$$

We take interest in the random variable `FPT` "first passage time", is defined by :

$$FPT = inf(t >= 0\ R(t) <= v)$$

with `v is the threshold`.

For more detail consulted `References`.

Value

M-sample for FPTT.

Note

- `2*K > Sigma^2`.

o system.time(tho_M2(N=1000, M=100, t0=0, T=1, R0=2, v=0.05, K=3, s=2,Sigma=0.3,Output = FALSE,Methods="Euler"))

utilisateur systeme ecoule

9.58 0.14 9.74

o system.time(tho_M2(N=1000, M=100, t0=0, T=1, R0=2, v=0.05, K=3, s=2,Sigma=0.3,Output = FALSE,Methods="RK3"))

utilisateur systeme ecoule

51.29 0.36 52.79

Author(s)

boukhetala Kamal, guidoum Arsalane.

References

1. K.Boukhetala, Estimation of the first passage time distribution for a simulated diffusion process, Maghreb Math.Rev, Vol.7, No 1, Jun 1998, pp. 1-25.
2. K.Boukhetala, Simulation study of a dispersion about an attractive centre. In proceedings of 11th Symposium Computational Statistics, edited by R.Dutter and W.Grossman, Wien , Austria, 1994, pp. 128-130.
3. K.Boukhetala,Modelling and simulation of a dispersion pollutant with attractive centre, Edited by Computational Mechanics Publications, Southampton ,U.K and Computational Mechanics Inc, Boston, USA, pp. 245-252.
4. K.Boukhetala, Kernel density of the exit time in a simulated diffusion, les Annales Maghrebines De L ingenieur, Vol , 12, N Hors Serie. Novembre 1998, Tome II, pp 587-589.

See Also

`AnaSimFPT` Simulation The First Passage Time FPT For A Simulated Diffusion Process.

Examples

```
tho_M2(N=1000, M=50, t0=0, T=1, R0=2, v=0.05, K=3,s=2,
       Sigma=0.3,Output = FALSE,Methods="Euler")
```

`TowDiffAtra2D` *Two-Dimensional Attractive Model for Two-Diffusion Processes V(1) and V(2)*

Description

simulation 2-dimensional attractive model for 2-diffusion processes V(1)=(X1(t),X2(t)) and V(2)=c(Y1(t),Y2(t)).

Usage

```
TowDiffAtra2D(N, t0, Dt, T = 1, X1_0, X2_0, Y1_0, Y2_0,
              v, K, m, Sigma, Output = FALSE)
```

Arguments

`N`	size of process.
`t0`	initial time.
`Dt`	time step of the simulation (`discretization`).
`T`	final time.
`X1_0`	initial value of the process `X1(t)` at time `t0`.
`X2_0`	initial value of the process `X2(t)` at time `t0`.
`Y1_0`	initial value of the process `Y1(t)` at time `t0`.
`Y2_0`	initial value of the process `Y2(t)` at time `t0`.
`v`	threshold. `see detail`
`K`	constant `K > 0`.
`m`	constant `m > 0`.
`Sigma`	constant `Sigma > 0`.
`Output`	if `Output = TRUE` write a `Output` to an Excel 2007.

Details

The 2-dimensional attractive models for 2-diffusion processes V(1)=(X1(t),X2(t)) and V(2)=c(Y1(t),Y2(t)) is defined by the Two (02) system for stochastic differential equation Two-dimensional :

$$dV1(t) = dV2(t) + Mu(m+1)(||D(t)||) * D(t) * dt + SigmaI(2*2) * dW1(t)$$

$$dV2(t) = Sigma * I(2*2) * dW2(t)$$

with:

$$D(t) = V1(t) - V2(t)$$

$$Mu(m)(||d||) = -K/||d||^m$$

Where ||.|| is the Euclidean norm and I(2*2) is identity matrix, `dW1(t)` and `dW2(t)` are brownian motions independent.

`D(t)=sqrt((X1(t)^2 - Y1(t)^2)+(X2(t)^2 - Y2(t)^2))` it is distance between `V1(t)` and `V2(t)`.

And the random variable `tau` "first passage time", is defined by :

$$tau(V1(t), V2(t)) = inf(t >= 0 \; ||D(t)|| <= v)$$

with `v is the threshold`.

Value

data.frame(time,X1(t),X2(t),Y1(t),Y2(t),D(t)) and plot of process 2-D.

Note

- `2*K > Sigma^2`.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

`TowDiffAtra3D`, `tho_02diff`.

Examples

```
TowDiffAtra2D(N=2000, t0=0, Dt=0.001, T = 1, X1_0=0.5, X2_0=1,
              Y1_0=-0.5, Y2_0=-1,v=0.05, K=2, m=0.2,
              Sigma=0.1, Output = FALSE)
```

`TowDiffAtra3D` *Three-Dimensional Attractive Model for Two-Diffusion Processes V(1) and V(2)*

Description

simulation 3-dimensional attractive model for 2-diffusion processes V(1)=(X1(t),X2(t),X3(t)) and V(2)=c(Y1(t),Y2(t),Y3(t)).

Usage

```
TowDiffAtra3D(N, t0, Dt, T = 1, X1_0, X2_0, X3_0, Y1_0,
               Y2_0, Y3_0, v, K, m, Sigma, Output = FALSE)
```

Arguments

`N`	size of process.
`t0`	initial time.
`Dt`	time step of the simulation (`discretization`).
`T`	final time.
`X1_0`	initial value of the process `X1(t)` at time `t0`.
`X2_0`	initial value of the process `X2(t)` at time `t0`.
`X3_0`	initial value of the process `X3(t)` at time `t0`.
`Y1_0`	initial value of the process `Y1(t)` at time `t0`.
`Y2_0`	initial value of the process `Y2(t)` at time `t0`.
`Y3_0`	initial value of the process `Y3(t)` at time `t0`.
`v`	threshold. `see detail`
`K`	constant `K > 0`.
`m`	constant `m > 0`.
`Sigma`	constant `Sigma > 0`.
`Output`	if `Output = TRUE` write a `Output` to an Excel 2007.

Details

The 3-dimensional attractive models for 2-diffusion processes V(1)=(X1(t),X2(t),X3(t)) and V(2)=c(Y1(t),Y2(t),Y3(t)) is defined by the Two (02) system for stochastic differential equation three-dimensional :

$$dV1(t) = dV2(t) + Mu(m+1)(||D(t)||) * D(t) * dt + SigmaI(3*3) * dW1(t)$$

$$dV2(t) = Sigma * I(3*3) * dW2(t)$$

with:

$$D(t) = V1(t) - V2(t)$$

$$Mu(m)(||d||) = -K/||d||^m$$

Where ||.|| is the Euclidean norm and I(3*3) is identity matrix, `dW1(t)` and `dW2(t)` are brownian motions independent.

`D(t)=sqrt((X1(t)^2 - Y1(t)^2)+(X2(t)^2 - Y2(t)^2)+(X3(t)^2-Y3(t)^2))` it is distance between `V1(t)` and `V2(t)` .

And the random variable `tau` "first passage time", is defined by :

$$tau(V1(t), V2(t)) = inf(t >= 0\ ||D(t)|| <= v)$$

with `v is the threshold`.

Value

data.frame(time,X1(t),X2(t),X3(t),Y1(t),Y2(t),Y3(t),D(t)) and plot of process 3-D.

Note

- `2*K > Sigma^2`.

Author(s)

boukhetala Kamal, guidoum Arsalane.

See Also

TowDiffAtra2D, tho_02diff.

Examples

```
TowDiffAtra3D(N=500, t0=0, Dt=0.001, T = 1, X1_0=0.5, X2_0=0.25,
              X3_0=0.1,Y1_0=-0.5,Y2_0=-1, Y3_0=0.25, v=0.01, K=5,
              m=0.2, Sigma=0.1, Output = FALSE)
```

WNG — *Creating White Noise Gaussian*

Description

Simulation white noise gaussian.

Usage

```
WNG(N, t0, T, m, sigma2, output = FALSE)
```

Arguments

N	size of process.
t0	initial time.
T	final time.
m	mean.
sigma2	variance.
output	if `output = TRUE` write a `output` to an Excel 2007.

Value

data.frame(time,x) and plot of process.

Author(s)

boukhetala Kamal, guidoum Arsalane.

Examples

```
## White Noise Gaussian
WNG(N=1000,t0=0,T=1,m=0,sigma2=4)
```

Index