R documentation

of all in
'/Users/mdelattre/Documents/R/MsdeParEst/MsdeParEst/man'

August 11, 2017

R topics documented:

`MsdeParEst-package` *Parametric estimation in mixed-effects stochastic differential equations*

Description

Parametric estimation in mixed-effects stochastic differential equations

Details

This package is dedicated to parametric estimation in the following mixed-effects stochastic differential equations:

$$dX_j(t) = (\alpha_j - \beta_j X_j(t))dt + \sigma_j \; a(X_j(t))dW_j(t),$$

$j = 1, \ldots, M$, where the $(W_j(t))$ are independant Wiener processes and the $(X_j(t))$ are observed without noise. The volatility function $a(x)$ is known and can be either $a(x) = 1$ (Ornstein-Uhlenbeck process) or $a(x) = \sqrt{x}$ (Cox-Ingersoll-Ross process).

Different estimation methods are implemented depending on whether there are random effects in the drift and/or in the diffusion coefficient:

1. The diffusion coefficient is fixed $\sigma_j \equiv \sigma$ and the parameters in the drift are Gaussian random variables:

(a) either $\alpha_j \equiv \alpha$ and $\beta_j \sim N(\mu, \Omega), j = 1, \ldots, M$,
(b) or $\beta_j \equiv \beta$ and $\alpha_j \sim N(\mu, \Omega), j = 1, \ldots, M$,
(c) or $(\alpha_j, \beta_j) \sim N(\mu, \Omega), j = 1, \ldots, M$.

μ, Ω and potentially the fixed effects σ, α, β are estimated as proposed in [1] and [4]. The extension to mixtures of Gaussian distributions is also implemented by following [3].

2. The coefficients in the drift are fixed $\alpha_j \equiv \alpha$ and $\beta_j \equiv \beta$ and the diffusion coefficient $1/\sigma_j^2$ follows a Gamma distribution $1/\sigma_j^2 \sim \Gamma(a, \lambda), j = 1, \ldots, M$. a, λ, and potentially the fixed effects α and β are estimated by the method published in [2].
3. There are random effects in the drift and in the diffusion, such that $1/\sigma_j^2 \sim \Gamma(a, \lambda)$ and

(a) either $\alpha_j \equiv \alpha$ and $\beta_j | \sigma_j \sim N(\mu, \sigma_j^2 \Omega)$,
(b) or $\beta_j \equiv \beta$ and $\alpha_j | \sigma_j \sim N(\mu, \sigma_j^2 \Omega)$,
(c) or $(\alpha_j, \beta_j) | \sigma_j \sim N(\mu, \sigma_j^2 \Omega)$.

a, λ, μ, Ω and potentially the fixed effects α and β are estimated by following [5].

References

[1] Maximum Likelihood Estimation for Stochastic Differential Equations with Random Effects, Delattre, M., Genon-Catalot, V. and Samson, A. *Scandinavian Journal of Statistics 40(2) 2012* **322-343**.

[2] Estimation of population parameters in stochastic differential equations with random effects in the diffusion coefficient, Delattre, M., Genon-Catalot, V. and Samson, A. *ESAIM:PS 19 2015* **671-688**.

[3] Mixtures of stochastic differential equations with random effects: application to data clustering, Delattre, M., Genon-Catalot, V. and Samson, A. *Journal of Statistical Planning and Inference 173 2016* **109-124**.

[4] Parametric inference for discrete observations of diffusion processes with mixed effects, Delattre, M., Genon-Catalot, V. and Laredo, C. *Stochastic Processes and their Applications*. To appear. (Available pre-publication hal-0133263, 2016).

[5] Estimation of the joint distribution of random effects for a discretely observed diffusion with random effects, Delattre, M., Genon-Catalot, V. and Laredo, C. *hal-01446063 2017*.

`Fit.class-class` *S4 class for the estimation results in the mixed SDE with random effects in the drift, in the diffusion or both*

Description

S4 class for the estimation results in the mixed SDE with random effects in the drift, in the diffusion or both

Slots

`model` 'OU' or 'CIR' (character)

`drift.random` 0, 1, 2, or c(1,2) (numeric)

`diffusion.random` 0 or 1 (numeric)

`gridf` matrix of values on which the estimation of the density of the random effects in the drift is done (matrix) `gridg` matrix of values on which the estimation of the density of the random effects in the diffusion is done (matrix)

`mu` estimator of the mean mu of the drift random effects (numeric)

`omega` estimator of the variance of the drift random effects (numeric)

`a` estimator of the shape of the Gamma distribution for the diffusion random effect (numeric)

`lambda` estimator of the scale of the Gamma distribution for the diffusion random effect (numeric)

`sigma2` estimated value of σ^2 if the diffusion coefficient is not random (numeric)

`index` index of the valid trajectories for the considered model (numeric)

`indexestim` index of the trajectories used for the estimation (numeric)

`estimphi` matrix of the estimator of the drift random effects (matrix)

`estimpsi2` vector of the estimator of the diffusion random effects σ_j^2 (numeric)

`estimf` estimator of the (conditional) density of the drift random effects (numeric)

`estimg` estimator of the density of σ_j^2 (numeric)

`estim.drift.fix` 1 if the user asked for the estimation of fixed parameter in the drift (numeric)

`estim.diffusion.fix` 1 if the user asked for the estimation of fixed diffusion coefficient (numeric)

`discrete` 1 if the estimation is based on the likelihood of discrete observations, 0 otherwise (numeric)

`bic` bic (numeric)

`aic` aic (numeric)

`times` vector of observation times, storage of input variable (numeric)

`X` matrix of observations, storage of input variable (matrix)

`Mixture.fit.class-class` *S4 class for the estimation results when the random effects in the drift follow mixture of normal distributions*

Description

S4 class for the estimation results when the random effects in the drift follow mixture of normal distributions

Slots

`model` character 'OU' or 'CIR'

`drift.random` numeric 1, 2, or c(1,2)

`gridf` matrix of values on which the estimation of the density of the random effects is done

`mu` array estimated value of the mean of the drift random effects at each iteration of the EM algorithm (Niter x nb.mixt x 2)

`omega` array estimated value of the standard deviation of the drift random effects at each iteration of the EM algorithm (Niter x nb.mixt x 2)

`mixt.prop` matrix estimated value of the mixing proportions at each iteration of the EM algorithm (Niter x nb.mixt) `sigma2` numeric estimated value of σ^2

`index` index of the valid trajectories for the considered model (numeric)

`indexestim` index of the trajectories used for the estimation (numeric)

`estimphi` matrix of the estimator of the drift random effects

`probindi` matrix of posterior component probabilities

`estimf` matrix estimator of the density of the drift random effects

`estim.drift.fix` numeric 1 if the user asked for the estimation of fixed parameter in the drift

`bic` numeric bic

`aic` numeric aic

`times` vector of observation times, storage of input variable

`X` matrix of observations, storage of input variable

`msde.fit` *Estimation Of The Random Effects In Mixed Stochastic Differential Equations*

Description

Parametric estimation of the joint density of the random effects in the mixed SDE

$$dX_j(t) = (\alpha_j - \beta_j X_j(t))dt + \sigma_j\, a(X_j(t))dW_j(t),$$

$j = 1, \ldots, M$, where the $(W_j(t))$ are independant Wiener processes and the $(X_j(t))$ are observed without noise. There can be random effects either in the drift (α_j, β_j) or in the diffusion coefficient σ_j or both $(\alpha_j, \beta_j, \sigma_j)$.

Usage

```
msde.fit(times, X, model = c("OU", "CIR"), drift.random = c(1, 2),
  drift.fixed = NULL, diffusion.random = 0, diffusion.fixed = NULL,
  nb.mixt = 1, Niter = 10, discrete = 1, valid = 0, level = 0.05,
  newwindow = FALSE)
```

Arguments

`times`	vector of observation times
`X`	matrix of the M trajectories (each row is a trajectory with as much columns as observations)
`model`	name of the SDE: 'OU' (Ornstein-Uhlenbeck) or 'CIR' (Cox-Ingersoll-Ross)
`drift.random`	random effects in the drift: 0 if only fixed effects, 1 if one additive random effect, 2 if one multiplicative random effect or c(1,2) if 2 random effects. Default to c(1,2)
`drift.fixed`	NULL if the fixed effect(s) in the drift is (are) estimated, value of the fixed effect(s) otherwise. Default to NULL
`diffusion.random`	1 if σ is random, 0 otherwise. Default to 0

`diffusion.fixed` NULL if σ is estimated (if fixed), value of σ otherwise. Default to NULL

`nb.mixt` number of mixture components for the distribution of the random effects in the drift. Default to 1 (no mixture)

`Niter` number of iterations for the EM algorithm if the random effects in the drift follow a mixture distribution. Default to 10

`discrete` 1 for using a contrast based on discrete observations, 0 otherwise. Default to 1

`valid` 1 if test validation, 0 otherwise. Default to 0

`level` alpha for the predicion intervals. Default 0.05

`newwindow` logical(1), if TRUE, a new window is opened for the plot. Default to FALSE

Details

Parametric estimation of the random effects density from M independent trajectories of the SDE:

$$dX_j(t) = (\alpha_j - \beta_j X_j(t))dt + \sigma_j \; a(X_j(t))dW_j(t),$$

$j = 1, \ldots, M$, where the $(W_j(t))$ are independant Wiener processes and the $(X_j(t))$ are observed without noise.

Specification of the random effects:

The drift includes no, one or two random effects:

1. if drift.random = 0: $\alpha_j \equiv \alpha$ and $\beta_j \equiv \beta$ are fixed
2. if drift.random = 1: $\beta_j \equiv \beta$ is fixed and α_j is random
3. if drift.random = 2: $\alpha_j \equiv \alpha$ is fixed and β_j is random
4. if drift.random = c(1,2): α_j and β_j are random

The diffusion includes either a fixed effect or a random effect:

1. if diffusion.random = 0: $\sigma_j \equiv \sigma$ is fixed
2. if diffusion.random = 1: σ_j is random

Distribution of the random effects

If there is no random effect in the diffusion (diffusion.random = 0), there is at least on random effect in the drift that follows

1. a Gaussian distribution (nb.mixt=1): $\alpha_j \sim N(\mu, \Omega)$ or $\beta_j \sim N(\mu, \Omega)$ or $(\alpha_j, \beta_j) \sim N(\mu, \Omega)$,
2. or a mixture of Gaussian distributions (nb.mixt=K, K>1): $\alpha_j \sim \sum_{k=1}^K p_k N(\mu_k, \Omega_k)$ or $\beta_j \sim \sum_{k=1}^K p_k N(\mu_k, \Omega_k)$ or $(\alpha_j, \beta_j) \sim \sum_{k=1}^K p_k N(\mu_k, \Omega_k)$, where $\sum_{k=1}^K p_k = 1$.

If there is one random effect in the diffusion (diffusion.random = 1), $1/\sigma_j^2 \sim \Gamma(a, \lambda)$, and the coefficients in the drift are conditionally Gaussian: $\alpha_j|\sigma_j \sim N(\mu, \sigma_j^2\Omega)$ or $\beta_j|\sigma_j \sim N(\mu, \sigma_j^2\Omega)$ or $(\alpha_j, \beta_j)|\sigma_j \sim N(\mu, \sigma_j^2\Omega)$, or they are fixed $\alpha_j \equiv \alpha, \beta_j \equiv \beta$.

SDEs

Two diffusions are implemented:

1. the Ornstein-Uhlenbeck model (OU) $a(X_j(t)) = 1$
2. the Cox-Ingersoll-Ross model (CIR) $a(X_j(t)) = \sqrt{X_j(t)}$

Estimation

- If discrete = 0, the estimation is based on the exact likelihood associated with continuous observations ([1],[3]). This is only possible if diffusion.random = 0 and σ is not estimated by maximum likelihood but empirically by means of the quadratic variations.
- If discrete = 1, the likelihood of the Euler scheme of the mixed SDE is computed and maximized for estimating all the parameters.
- If nb.mixt > 1, an EM algorithm is implemented and the number of iterations of the algorithm must be specified with Niter.
- If valid = 1, two-thirds of the sample trajectories are used for estimation, while the rest is used for validation. A plot is then provided for visual comparison between the true trajectories of the test sample and some predicted trajectories simulated under the estimated model.

Value

`index`	is the vector of subscript in 1,...,M used for the estimation. Most of the time index=1:M, except for the CIR that requires positive trajectories.
`estimphi`	matrix of estimators of the drift random effects $\hat{\alpha}_j$, or $\hat{\beta}_j$ or $(\hat{\alpha}_j, \hat{\beta}_j)$
`estimpsi2`	vector of estimators of the squared diffusion random effects $\hat{\sigma}_j^2$
`gridf`	grid of values for the plots of the random effects distribution in the drift, matrix form
`gridg`	grid of values for the plots of the random effects distribution in the diffusion, matrix form
`estimf`	estimator of the density of α_j, β_j or (α_j, β_j). Matrix form.
`estimg`	estimator of the density of σ_j^2. Matrix form.
`mu`	estimator of the mean of the random effects normal density
`omega`	estimator of the standard deviation of the random effects normal density
`a`	estimated value of the shape of the Gamma distribution
`lambda`	estimated value of the scale of the Gamma distribution
`sigma2`	value of the diffusion coefficient if it is fixed
`bic`	BIC criterium
`aic`	AIC criterium
`model`	initial choice
`drift.random`	initial choice
`diffusion.random`	initial choice
`drift.fixed`	initial choice
`estim.drift.fix`	1 if the fixed effects in the drift are estimated, 0 otherwise.
`estim.diffusion.fixed`	1 if the fixed effect in the diffusion is estimated, 0 otherwise.
`discrete`	initial choice
`times`	initial choice
`X`	initial choice

For mixture distributions in the drift:

`mu`	estimated value of the mean at each iteration of the algorithm. Niter x N x 2 array.

omega	estimated value of the standard deviation at each iteration of the algorithm. Niter x N x 2 array.
mixt.prop	estimated value of the mixture proportions at each iteration of the algorithm. Niter x N matrix.
probindi	posterior component probabilites. M x N matrix.

References

See

[1] Maximum Likelihood Estimation for Stochastic Differential Equations with Random Effects, Delattre, M., Genon-Catalot, V. and Samson, A. *Scandinavian Journal of Statistics 40(2) 2012* **322-343**

[2] Estimation of population parameters in stochastic differential equations with random effects in the diffusion coefficient, Delattre, M., Genon-Catalot, V. and Samson, A. *ESAIM:PS 19 2015* **671-688**

[3] Mixtures of stochastic differential equations with random effects: application to data clustering, Delattre, M., Genon-Catalot, V. and Samson, A. *Journal of Statistical Planning and Inference 173 2016* **109-124**

[4] Parametric inference for discrete observations of diffusion processes with mixed effects, Delattre, M., Genon-Catalot, V. and Laredo, C. *hal-01332630 2016*

[5] Estimation of the joint distribution of random effects for a discretely observed diffusion with random effects, Delattre, M., Genon-Catalot, V. and Laredo, C. *hal-01446063 2017*

Examples

```
## Not run:
# Example : One random effect in the drift and one random effect in the diffusion

# -- Simulation

sim <- msde.sim(M = 100, T = 5, N = 5000, model = 'OU',
                drift.random = 2, drift.param = c(0,0.5,0.5),
                diffusion.random = 1, diffusion.param = c(8,1/2))

# -- Fixed effect in the drift estimated
res <- msde.fit(times = sim$times, X = sim$X, model = 'OU', drift.random = 2,
diffusion.random = 1)

# ----- Fixed effect in the drift known and not estimated
resbis <- msde.fit(times = sim$times, X = sim$X, model = "OU", drift.random = 2,
                    diffusion.random = 1, drift.fixed = 0)
summary(resbis)

# Example : one mixture of two random effects in the drift, and one fixed effect in
# the diffusion coefficient

sim <- msde.sim(M = 100, T = 5, N = 5000, model = 'OU', drift.random = c(1,2),
                diffusion.random = 0,
                drift.param = matrix(c(0.5,1.8,0.25,0.25,1,2,0.25,0.25),nrow=2,byrow=F),
                diffusion.param = 0.1, nb.mixt = 2, mixt.prop = c(0.5,0.5))

# -- Estimation without validation
```

```
res <- msde.fit(times = sim$times, X = sim$X, model = 'OU', drift.random = c(1,2),
                nb.mixt=2, Niter = 10)

summary(res)
plot(res)

# -- Estimation with prediction
res.valid <- msde.fit(times = sim$times, X = sim$X, model = 'OU', drift.random = c(1,2),
                      nb.mixt=2, Niter = 10, valid = 1)

summary(res.valid)
plot(res.valid)

## End(Not run)
```

`msde.sim` *Simulation Of A Mixed Stochastic Differential Equation*

Description

Simulation of M independent trajectories of a mixed stochastic differential equation (SDE) with linear drift

$$dX_j(t) = (\alpha_j - \beta_j X_j(t))dt + \sigma_j a(X_j(t))dW_j(t), j = 1, ..., M.$$

There can be up to two random effects (α_j, β_j) in the drift and one random effect σ_j in the diffusion coefficient.

Usage

```
msde.sim(M, T, N = 100, model, drift.random, diffusion.random, drift.param,
  diffusion.param, nb.mixt = 1, mixt.prop = 1, t0 = 0, X0 = 0.01,
  delta = T/N, op.plot = 0, add.plot = FALSE)
```

Arguments

`M`	number of trajectories.
`T`	horizon of simulation.
`N`	number of simulation steps, default Tx100.
`model`	name of the SDE: 'OU' (Ornstein-Uhlenbeck) or 'CIR' (Cox-Ingersoll-Ross).
`drift.random`	random effects in the drift: 0 if no random effect, 1 if one additive random effect, 2 if one multiplicative random effect or c(1,2) if 2 random effects.
`diffusion.random`	random effect in the diffusion coefficient: 0 if no random effect, 1 if one multiplicative random effect.
`drift.param`	vector (not mixture) or matrix (mixture) of values of the fixed effects and/or the parameters of the distribution of the random effects in the drift (see details).
`diffusion.param`	diffusion parameter if the diffusion coefficient is fixed, vector of parameters $c(a, \lambda)$ of the distribution of the diffusion random effect otherwise.

`nb.mixt`	number of mixture components if the drift random effects follow a mixture distribution, default nb.mixt=1.
`mixt.prop`	vector of mixture proportions if the drift random effects follow a mixture distribution, default mixt.prop=1.
`t0`	time origin, default 0.
`X0`	initial value of the process, default X0=0.001.
`delta`	time step of the simulation (T/N).
`op.plot`	1 if a plot of the trajectories is required, default 0.
`add.plot`	1 for add trajectories to an existing plot

Details

Simulation of N discrete observations on time interval [t0,T] of M independent trajectories of the SDE

$$dX_j(t) = (\alpha_j - \beta_j X_j(t))dt + \sigma_j\ a(X_j(t))dW_j(t),$$

$j = 1, \ldots, M$, where the $(W_j(t))$ are independant Wiener processes.

Specification of α, β, σ

The diffusion includes either a fixed effect or a random effect:

1. if diffusion.random = 0: $\sigma_j \equiv \sigma$ is fixed, and diffusion.param = σ. In this case, the drift includes no, one or two random effects:
   (a) if drift.random = 0: $\alpha_j \equiv \alpha$ and $\beta_j \equiv \beta$ are fixed, and drift.param=c(α, β)
   (b) if drift.random = 1: α_j is random with distribution $N(\mu_\alpha, \omega_\alpha^2)$ whereas $\beta_j \equiv \beta$ is fixed, and drift.param=c$(\mu_\alpha, \omega_\alpha^2, \beta)$
   (c) if drift.random = 2: $\alpha_j \equiv \alpha$ is fixed and β_j is random with distribution $N(\mu_\beta, \omega_\beta^2)$, and drift.param=c$(\alpha, \mu_\beta, \omega_\beta^2)$
   (d) if drift.random = c(1,2): α_j and β_j are random with distributions $N(\mu_\alpha, \omega_\alpha^2)$ and $N(\mu_\beta, \omega_\beta^2)$ respectively, and drift.param = c$(\mu_\alpha, \omega_\alpha^2, \mu_\beta, \omega_\beta^2)$
2. if diffusion.random = 1: σ_j is random such that $1/\sigma_j^2 \sim \Gamma$, and drift.param=c(a, λ). In this case, the drift includes at least one random effect:
   (a) if drift.random = 1: α_j is random with distribution $N(\mu_\alpha, \sigma_j^2\omega_\alpha^2)$ whereas $\beta_j \equiv \beta$ is fixed, and drift.param=c$(\mu_\alpha, \omega_\alpha^2, \beta)$
   (b) if drift.random = 2: $\alpha_j \equiv \alpha$ is fixed and β_j is random with distribution $N(\mu_\beta, \sigma_j^2\omega_\beta^2)$, and drift.param=c$(\alpha, \mu_\beta, \omega_\beta^2)$
   (c) if drift.random = c(1,2): α_j and β_j are random with distributions $N(\mu_\alpha, \sigma_j^2\omega_\alpha^2)$ and $N(\mu_\beta, \sigma_j^2\omega_\beta^2)$ respectively, and drift.param = c$(\mu_\alpha, \omega_\alpha^2, \mu_\beta, \omega_\beta^2)$

   If the random effects in the drift follow a mixture distribution (nb.mixt=K, K>1), drift.param is a matrix instead of a vector. Each line of the matrix contains, as above, the parameter values for each mixture component.

Value

`X`	matrix (M x (N+1)) of the M trajectories.
`times`	vector of the N+1 simulated observation times from t0 to T.
`phi`	vector (or matrix) of the M simulated random effects of the drift.
`psi`	vector of the M simulated values of σ_j.

References

This function mixedsde.sim is based on the package sde, function sde.sim. See

Simulation and Inference for stochastic differential equation, S.Iacus, *Springer Series in Statistics 2008* Chapter 2

See Also

http://cran.r-project.org/package=sde

Examples

```
 ## Not run:
 # Example : one random effect in the drift and one fixed effect in the diffusion coefficient
 sim <- msde.sim(M = 100, T = 5, N = 5000, model = 'OU', drift.random = 2,
             diffusion.random = 0, drift.param = c(0,1,sqrt(0.4/4)), diffusion.param = 0.5)

 # Example : two random effects in the drift and one random effect in the diffusion coefficient

 sim <- msde.sim(M = 100, T = 5, N = 5000, model = 'OU', drift.random = c(1,2),
             diffusion.random = 1, drift.param = c(1,0.5,0.5,0.5), diffusion.param = c(8,1/2))

 # Example : one fixed effect and one mixture random effect in the drift, and one fixed effect in
 # the diffusion coefficient

 sim <- msde.sim(M = 100, T = 5, N = 5000, model = 'OU',
              drift.random = 1, drift.param = matrix(c(0.5,1.8,0.25,0.25,1,1),nrow=2,byrow=F),
                 diffusion.random = 0, diffusion.param = 0.1,
                 nb.mixt = 2, mixt.prop = c(0.5,0.5))

## End(Not run)
```

---

plot,Fit.class,ANY-method

Plot method for the estimation class object

Description

Plot method for the S4 class Fit.class

Usage

```
## S4 method for signature 'Fit.class,ANY'
plot(x, newwindow = FALSE, ...)
```

Arguments

x	Fit.class class
newwindow	logical(1), if TRUE, a new window is opened for the plot
...	optional plot parameters

`plot,Mixture.fit.class,ANY-method`
Plot method for the mixture estimation class object

Description

Plot method for the S4 class Mixture.fit.class

Usage

```
## S4 method for signature 'Mixture.fit.class,ANY'
plot(x, newwindow = FALSE, ...)
```

Arguments

`x`	Mixture.fit.class class
`newwindow`	logical(1), if TRUE, a new window is opened for the plot
`...`	optional plot parameters

`summary,Fit.class-method`
Short summary of the results of class object Fit.class

Description

Method for the S4 class Fit.class

Usage

```
## S4 method for signature 'Fit.class'
summary(object)
```

Arguments

`object`	Fit.class class

`summary,Mixture.fit.class-method`
Short summary of the results of class object Mixture.fit.class

---

Description

Method for the S4 class Mixture.fit.class

Usage

```
## S4 method for signature 'Mixture.fit.class'
summary(object)
```

Arguments

`object`	Mixture.fit.class class

Index