Package 'varycoef'

October 11, 2019

Type Package

Title Varying Coefficients

Version 0.2.9

Depends spam

Imports fields, methods, sp, RandomFields

Suggests tmap, knitr, rmarkdown, microbenchmark

Description Gives maximum likelihood estimation (MLE) method to estimate and predict spatially varying coefficient (SVC) Models. It supports covariance tapering by Furrer et al. (2006) <doi:10.1198/106186006X132178> to allow MLE on large data.

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

VignetteBuilder knitr

NeedsCompilation no

Author Jakob Dambon [aut, cre],
Fabio Sigrist [ctb],
Reinhard Furrer [ctb]

Maintainer Jakob Dambon <jakob.dambon@math.uzh.ch>

Repository CRAN

Date/Publication 2019-10-11 09:20:02 UTC

R topics documented:

`coef.SVC_mle` *Extact Mean Effects*

Description

Method to extract the mean effects from an `SVC_mle` object.

Usage

```
## S3 method for class 'SVC_mle'
coef(object, ...)
```

Arguments

`object`	`SVC_mle` object
`...`	further arguments

Value

named vector with mean effects, i.e. μ from `SVC_mle`

Author(s)

Jakob Dambon

cov_par *Extact Covariance Parameters Function to extract the covariance parameters from an SVC_mle object.*

Description

Extact Covariance Parameters

Function to extract the covariance parameters from an SVC_mle object.

Usage

```
cov_par(object, ...)
```

Arguments

object	SVC_mle object
...	further arguments

Value

vector with covariance parameters and attributes what kind

- "GRF", charachter, describing the GRF used, see SVC_mle_control.
- "tapering", either NULL if no tapering is applied of the taper range.

Author(s)

Jakob Dambon

fitted.SVC_mle *Extact Model Fitted Values*

Description

Method to extract the fitted values from an SVC_mle object. This is only possible if save.fitted was set to TRUE in the control of the function call

Usage

```
## S3 method for class 'SVC_mle'
fitted(object, ...)
```

Arguments

object	SVC_mle object
...	further arguments

Value

data frame, fitted values to given data, i.e. the SVC as well as the response and their locations

Author(s)

Jakob Dambon

`fullSVC_reggrid` *Sample Function for SVCs*

Description

Samples SVC on a regular grid. The SVC have all mean 0.

Usage

```
fullSVC_reggrid(m, p, cov_pars, nugget, seed = 123)
```

Arguments

`m`	integer. square root number of observations, in total the function will sample m^2 locations on a regular grid.
`p`	integer. number of SVC
`cov_pars`	data.frame including the covariance parameters of SVCs, using an exponential covariance function. The columns must have the names `"var"` and `"range"`.
`nugget`	scalar. variance of the nugget / error term.
`seed`	integer. seed for sampling

Value

object of class `SpatialPointsDataFrame` (see `SpatialPointsDataFrame-class`) of the sampled SVC including the nugget.

`logLik.SVC_mle` *Extact the Likelihood*

Description

Method to extract the computed (penalized) log (profile) Likelihood from an `SVC_mle` object.

Usage

```
## S3 method for class 'SVC_mle'
logLik(object, ...)
```

Arguments

`object`	`SVC_mle` object
`...`	further arguments

Value

an object of class `logLik` with attributes

- `"penalized"`, logical, if the likelihood (`FALSE`) or some penalized likelihood (`TRUE`) was optimized.
- `"profileLik"`, logical, if the optimization was done using the profile likelihood (`TRUE`) or not.
- `"nobs"`, integer of number of observations
- `"df"`, integer of how many parameters were estimated. **Note**: This includes only the covariance parameters if the profile likelihood was used.

Author(s)

Jakob Dambon

`nlocs` *Extact Number of Unique Locations Function to extract the number of unique locations in the data set used in an MLE of the* `SVC_mle` *object.*

Description

Extact Number of Unique Locations

Function to extract the number of unique locations in the data set used in an MLE of the `SVC_mle` object.

Usage

```
nlocs(object)
```

Arguments

object	SVC_mle object

Value

integer with the number of unique locations

Author(s)

Jakob Dambon

`nobs.SVC_mle` *Extact Number of Observations*

Description

Method to extract the number of observations used in MLE for an SVC_mle object.

Usage

```
## S3 method for class 'SVC_mle'
nobs(object, ...)
```

Arguments

object	SVC_mle object
...	further arguments

Value

an integer of number of observations

Author(s)

Jakob Dambon

plot.SVC_mle *Plotting Residuals of* SVC_mle *model*

Description

Method to plot the residuals from an SVC_mle object. For this, save.fitted has to be TRUE in SVC_mle_control.

Usage

```
## S3 method for class 'SVC_mle'
plot(x, which = 1:3, legend.pos = "bottomright", ...)
```

Arguments

x	SVC_mle object
which	numeric, indicating which of the 3 plots should be plotted
legend.pos	character describing the position of the legend in the spatial residual plot, see legend
...	further arguments

Value

a maximum 3 plots

- Tukey-Anscombe plot, i.e. residuals vs. fitted
- QQ-plot
- spatial residuals

Author(s)

Jakob Dambon

See Also

legend SVC_mle

Examples

```
#' ## ---- toy example ----
## sample data
# setting seed for reproducibility
set.seed(123)
m <- 7
# number of observations
n <- m*m
# number of SVC
```

```
p <- 3
# sample data
y <- rnorm(n)
X <- matrix(rnorm(n*p), ncol = p)
# locations on a regular m-by-m-grid
locs <- expand.grid(seq(0, 1, length.out = m),
                    seq(0, 1, length.out = m))

## preparing for maximum likelihood estimation (MLE)
# controls specific to MLE
control <- SVC_mle_control(
  # initial values of optimization
  init = rep(0.1, 2*p+1),
  # using profile likelihood
  profileLik = TRUE
)

# controls specific to optimization procedure, see help(optim)
opt.control <- list(
  # number of iterations (set to one for demonstration sake)
  maxit = 1,
  # tracing information
  trace = 6
)

## starting MLE
fit <- SVC_mle(y = y, X = X, locs = locs,
               control = control,
               optim.control = opt.control)

## output: convergence code equal to 1, since maxit was only 1
summary(fit)

## plot residuals
# only QQ-plot
plot(fit, which = 2)

# all three plots next to each other
oldpar <- par(mfrow = c(1, 3))
plot(fit)
par(oldpar)
```

`predict.SVC_mle` *Prediction of SVC (and response variable)*

Description

Prediction of SVC (and response variable)

Usage

```
## S3 method for class 'SVC_mle'
predict(object, newlocs = NULL, newX = NULL,
  newW = NULL, compute.y.var = FALSE, ...)
```

Arguments

object	output of SVC_mle
newlocs	matrix of dimension n' x 2. These are the new locations the SVCs are predicted for. If NULL, the locations from the SVC_mle (i.e. locs) are considered.
newX	optional matrix of dimension n' x pX. If provided, besides the predicted SVC, the function also returns the predicted response variable.
newW	optional matrix of dimension n' x pW.
compute.y.var	logical. If y will be estimated and TRUE, the standard deviation of each estimate will be computed.
...	further arguments

Value

returns a data frame of n' rows and with columns

- SVC_1, ..., SVC_p, i.e. the predicted SVC at locations newlocs
- y.pred, if newX and newW are provided
- y.var, if newX and newW are provided and compute.y.var is set to TRUE.
- loc_x, loc_y, the locations of the predictions

Author(s)

Jakob Dambon

See Also

SVC_mle

Examples

```
## ---- toy example ----
## sample data
# setting seed for reproducibility
set.seed(123)
m <- 7
# number of observations
n <- m*m
# number of SVC
p <- 3
# sample data
y <- rnorm(n)
X <- matrix(rnorm(n*p), ncol = p)
```

```
# locations on a regular m-by-m-grid
locs <- expand.grid(seq(0, 1, length.out = m),
                    seq(0, 1, length.out = m))

## preparing for maximum likelihood estimation (MLE)
# controls specific to MLE
control <- SVC_mle_control(
  # initial values of optimization
  init = rep(0.1, 2*p+1),
  # using profile likelihood
  profileLik = TRUE
)

# controls specific to optimization procedure, see help(optim)
opt.control <- list(
  # number of iterations (set to one for demonstration sake)
  maxit = 1,
  # tracing information
  trace = 6
)

## starting MLE
fit <- SVC_mle(y = y, X = X, locs = locs,
               control = control,
               optim.control = opt.control)

## output: convergence code equal to 1, since maxit was only 1
summary(fit)

## prediction
# new location
newlocs <- matrix(0.5, ncol = 2, nrow = 1)

# new data
X.new <- matrix(rnorm(p), ncol = p)

# predicting SVCs
predict(fit, newlocs = newlocs)

# predicting SVCs and calculating response
predict(fit, newlocs = newlocs,
        newX = X.new, newW = X.new)

# predicting SVCs, calculating response and predictive variance
predict(fit, newlocs = newlocs,
        newX = X.new, newW = X.new,
        compute.y.var = TRUE)
```

---

print.summary.SVC_mle *Printing Method for* summary.SVC_mle

Description

Printing Method for `summary.SVC_mle`

Usage

```
## S3 method for class 'summary.SVC_mle'
print(x, digits = max(3L, getOption("digits") -
  3L), ...)
```

Arguments

`x`	summary.SVC_mle
`digits`	the number of significant digits to use when printing.
`...`	further arguments

Value

The printed output of the summary in the console.

See Also

summary.SVC_mle SVC_mle

`print.SVC_mle` *Print Method for* `SVC_mle`

Description

Method to print an `SVC_mle` object.

Usage

```
## S3 method for class 'SVC_mle'
print(x, digits = max(3L, getOption("digits") - 3L),
  ...)
```

Arguments

`x`	`SVC_mle` object
`digits`	numeric, number of digits to be plotted
`...`	further arguments

Author(s)

Jakob Dambon

residuals.SVC_mle *Extact Model Residuals*

Description

Method to extract the residuals from an SVC_mle object. This is only possible if save.fitted was set to TRUE in the control of the function call

Usage

```
## S3 method for class 'SVC_mle'
residuals(object, ...)
```

Arguments

object	SVC_mle object
...	further arguments

Value

numeric, residuals of model

Author(s)

Jakob Dambon

summary.SVC_mle *Summary Method for* SVC_mle

Description

Method to construct a summary.SVC_mle object out of a SVC_mle object.

Usage

```
## S3 method for class 'SVC_mle'
summary(object, ...)
```

Arguments

object	SVC_mle object
...	further arguments

Value

object of class summary.SVC_mle with summarized values of the MLE.

Author(s)

Jakob Dambon

See Also

SVC_mle

SVC_mle *MLE of SVC model*

Description

Calls MLE of the SVC model defined as:

$$y(s) = X\mu + W\eta(s) + \epsilon(s)$$

where:

- y is the response (vector of length n)
- X is the data matrix for the fixed effects covariates
- μ is the vetor containing the fixed effects
- W is the data matrix for the SVCs represented by zero mean GRF
- η are the SVCs represented by zero mean GRF
- ϵ is the nugget effect

The MLE is done by calling the function `optim`.

Usage

```
SVC_mle(...)

## Default S3 method:
SVC_mle(y, X, locs, W = NULL, control = NULL,
  optim.control = list(), ...)

## S3 method for class 'formula'
SVC_mle(formula, data, RE_formula = NULL, locs,
  control, optim.control = list(), ...)
```

Arguments

...	further arguments
y	numeric response vector of dimension n.
X	matrix of covariates of dimension n x pX. Intercept has to be added manually.
locs	matrix of locations of dimension n X 2. May contain multiple observations at single location which (may) cause a permutation of y, X, W and locs.
W	Optional matrix of covariates with fixed effects, i.e. non-SVC, of dimension n x pW
control	list of control paramaters, usually given by SVC_mle_control
optim.control	list of control arguments for optimization function, see Details in optim
formula	Formula describing the fixed effects in SVC model. The response, i.e. LHS of the formula, is not allowed to have functions such as sqrt() or log().
data	data frame containing the observations
RE_formula	Formula describing the random effects in SVC model. Only RHS is considered. If NULL, the same RHS of argument formula for fixed effects is used.

Value

Object of class SVC_mle

Author(s)

Jakob Dambon

See Also

predict.SVC_mle

Examples

```
## ---- toy example ----
## sample data
# setting seed for reproducibility
set.seed(123)
m <- 7
# number of observations
n <- m*m
# number of SVC
p <- 3
# sample data
y <- rnorm(n)
X <- matrix(rnorm(n*p), ncol = p)
# locations on a regular m-by-m-grid
locs <- expand.grid(seq(0, 1, length.out = m),
                    seq(0, 1, length.out = m))

## preparing for maximum likelihood estimation (MLE)
```

```
# controls specific to MLE
control <- SVC_mle_control(
  # initial values of optimization
  init = rep(0.1, 2*p+1),
  # using profile likelihood
  profileLik = TRUE
)

# controls specific to optimization procedure, see help(optim)
opt.control <- list(
  # number of iterations (set to one for demonstration sake)
  maxit = 1,
  # tracing information
  trace = 6
)

## starting MLE
fit <- SVC_mle(y = y, X = X, locs = locs,
               control = control,
               optim.control = opt.control)

## output: convergence code equal to 1, since maxit was only 1
summary(fit)


## ---- real data example ----
require(sp)
## get data set
data("meuse", package = "sp")

# construct data matrix and response, scale locations
y <- log(meuse$cadmium)
X <- model.matrix(~1+dist+lime+elev, data = meuse)
locs <- as.matrix(meuse[, 1:2])/1000


## starting MLE
# the next call takes a couple of seconds
fit <- SVC_mle(y = y, X = X, locs = locs,
               # has 4 fixed effects, but only 3 random effects (SVC)
               # elev is missing in SVC
               W = X[, 1:3],
               control = SVC_mle_control(
                 # inital values for 3 SVC
                 # 7 = (3 * 2 covariance parameters + nugget)
                 init = c(rep(c(0.4, 0.2), 3), 0.2),
                 profileLik = TRUE
               ))

## summary and residual output
summary(fit)
plot(fit)
```

```
## predict
# new locations
newlocs <- expand.grid(
  x = seq(min(locs[, 1]), max(locs[, 1]), length.out = 30),
  y = seq(min(locs[, 2]), max(locs[, 2]), length.out = 30))
# predict SVC for new locations
SVC <- predict(fit, newlocs = as.matrix(newlocs))
# visualization
sp.SVC <- SVC
coordinates(sp.SVC) <- ~loc_x+loc_y
spplot(sp.SVC, colorkey = TRUE)
```

SVC_mle_control *Set Parameters for* SVC_mle

Description

Function to set up control parameters for SVC_mle

Usage

```
SVC_mle_control(...)

## Default S3 method:
SVC_mle_control(cov.name = c("exp", "sph"),
  tapering = NULL, cl = NULL, init = NULL, lower = NULL,
  upper = NULL, save.fitted = TRUE, profileLik = FALSE,
  mean.est = c("GLS", "OLS"), pc.prior = NULL, ...)

## S3 method for class 'SVC_mle'
SVC_mle_control(object, ...)
```

Arguments

...	further parameters yet to be implemented
cov.name	name of the covariance function defining the covariance matrix of the GRF. Currently, only "exp" for the exponential and "exp" for spherical covariance functions are supported.
tapering	if NULL, no tapering is applied. If a scalar is given, covariance tapering with this taper range is applied, for all GRF modelling the SVC.
cl	cluster for parallelization. Currently not supported.
init	numeric. Initial values for optimization procedure. The vector consists of p-times (alternating) scale and variance, the nugget variance and the p + p.fix mean effects
lower	lower bound for optim, default NULL sets the lower bounds to 1e-6 for covariance parameters and -Inf for mean parameters.

upper	upper bound for optim, default NULL sets the upper bounds to Inf for covariance and mean parameters.
save.fitted	logical. If TRUE, calculates the fitted values and residuals after MLE and saves them.
profileLik	logical. If TRUE, MLE is done over profile Likelihood of covariance parameters.
mean.est	if profileLik is TRUE, the means have to be estimated seperately. "GLS" uses the generalized least square estimate while "OLS" uses the ordinary least squares estiamte.
pc.prior	takes vector of $\rho_0, \alpha_\rho, \sigma_0, \alpha_\sigma$ to compute penalized complexity priors. This regulates the optimization process. Currently, only supported for Gaussian random fields of Matérn class. Based on the idea Simpson and Fulgstad.
object	An object of class SVC_mle. The function then extracts the control settings from the particular function call used to compute object.

Value

A list with which SVC_mle can be controlled

Author(s)

Jakob Dambon

See Also

SVC_mle

Examples

```
control <- SVC_mle_control(init = rep(0.3, 10))
# or
control <- SVC_mle_control()
control$init <- rep(0.3, 10)
```

---

varycoef *varycoef: An R Package for Varying Coefficient Models.*

Description

This package offers functions to work with varying coefficient models. Currently, it can model, estimate and predict spatially varying coefficient (SVC) models. Briefly described, one generalizes a linear regression equation such that the coefficients are no longer constant, but have the possibility to vary spatially. This is enabled by modelling the coefficients by Gaussian random fields with either an exponential or spherical covariance function. The advantages of such SVC models are that they are usually quite easy to interpret, yet they offer a very highe level of flexibility.

Estimation and Prediction

The ensemble of the function SVC_mle and the method predict estimates the defined SVC model and gives predictions of the SVC as well as the response for some pre-defined locations. This concept should be rather familiar as it is the same for the classical regression (lm) or local polynomial regression (loess), to name a couple. As the name suggests, we are using a MLE approach in order to estimate the model and following the empirical best linear unbiased predictor to give location-specifc predictions. A detailed tutorial with examples is given in a vignette; call vignette("example",package = "varycoef").

Methods

With the before mentioned SVC_mle function one gets an object of class SVC_mle. And like the method predict for predictions, there are several more methods in order to diagnose the model, see methods(class = "SVC_mle").

Examples

```
vignette("example", package = "varycoef")
methods(class = "SVC_mle")
```

Index