Package ‘varycoef’

April 14, 2020

Type Package

Title Modeling Spatially Varying Coefficients

Version 0.2.11

Description Implements a maximum likelihood estimation (MLE) method for estimation and prediction in spatially varying coefficient (SVC) models (Dambon et al. (2020) <arXiv:2001.08089>). Covariance tapering (Furrer et al. (2006) <doi:10.1198/106186006X132178>) can be applied such that the method scales to large data.

License GPL-2

URL https://github.com/jakobdambon/varycoef

BugReports https://github.com/jakobdambon/varycoef/issues

Depends spam

Imports RandomFields, fields, optimParallel (>= 0.8-1), sp

Suggests gstat, knitr, microbenchmark, parallel, rmarkdown, R.rsp

VignetteBuilder knitr, R.rsp

Encoding UTF-8

LazyData true

RoxygenNote 7.1.0

NeedsCompilation no

Author Jakob A. Dambon [aut, cre] (<https://orcid.org/0000-0001-5855-2017>), Fabio Sigrist [ctb] (<https://orcid.org/0000-0002-3994-2244>), Reinhard Furrer [ctb] (<https://orcid.org/0000-0002-6319-2332>)

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

Repository CRAN

Date/Publication 2020-04-14 10:20:02 UTC
**R topics documented:**

- `coef.SVC_mle` .......................................................... 2
- `cov_par` ................................................................. 3
- `d.Lq` ................................................................. 3
- `d.SCAD` .............................................................. 4
- `fitted.SVC_mle` .......................................................... 5
- `fullSVC_reggrid` ......................................................... 5
- `logLik.SVC_mle` .......................................................... 6
- `Lq` ................................................................. 7
- `nlocs` ................................................................. 8
- `nobs.SVC_mle` ............................................................ 8
- `plot.SVC_mle` ........................................................... 9
- `predict.SVC_mle` .......................................................... 11
- `print.summary.SVC_mle` ................................................... 13
- `print.SVC_mle` ............................................................ 14
- `residuals.SVC_mle` ........................................................ 14
- `SCAD` ................................................................. 15
- `summary.SVC_mle` ......................................................... 16
- `SVC_mle` ............................................................... 16
- `SVC_mle_control` ......................................................... 19
- `varycoef` .............................................................. 22

**Index** ................................................................ 23

| coef.SVC_mle | Extract Mean Effects |

**Description**

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

**Usage**

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

**Arguments**

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

**Value**

named vector with mean effects, i.e. \( \mu \) from `SVC_mle`

**Author(s)**

Jakob Dambon
**cov_par**

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

**Description**

Extract Covariance Parameters
Function to extract the covariance parameters from an SVC_mle object.

**Usage**

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

**Arguments**

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

**Value**

vector with covariance parameters and attributes what kind
- "GRF", character, describing the GRF used, see SVC_mle_control.
- "tapering", either NULL if no tapering is applied or the taper range.

**Author(s)**

Jakob Dambon

---

**d.Lq**

*Derivative of L^q Norm Penalty*

**Description**

derivative of L_q, which is not differentiable at x == 0.

**Usage**

```r
d.Lq(x, lambda = 1, q = 1, d.side = "both")
```

**Arguments**

- `x` numeric.
- `lambda` non-negative scalar, shrinkage parameter.
- `q` non-negative scalar, norm parameter.
- `d.side` side of derivative at origin. Default value is "both", returning NA for x == 0. If set to "RHS", then returns RHS derivative, i.e., λ, and −λ with "LHS".
**Value**

\[ L^q(x) = q\lambda|x|^{q-1} \]

, for \( x = 0 \) return value is NA.

**Author(s)**

Jakob Dambon

**Examples**

```r
d.Lq(-5:5)
d.Lq(-2:2, d.side = "LHS")
curve(d.Lq, from = -5, to = 5)
```

---

**d.SCAD**

*Derivative of Smoothly Clipped Absolute Deviation Penalty*

**Description**

derivative of SCAD, which is not differentiable at \( x = 0 \).

**Usage**

\[ d.SCAD(x, \lambda = 1, a = 3.7, d.side = "both") \]

**Arguments**

- **x** numeric.
- **lambda** non-negative scalar, shrinkage parameter.
- **a** scalar larger than 2. Fan & Li (2001) suggest \( a = 3.7 \).
- **d.side** side of derivative at origin. Default value is "both", returning NA for \( x = 0 \). If set to "RHS", then returns RHS derivative, i.e., \( \lambda \), and \( -\lambda \) with "LHS".

**Value**

derivative of \( SCAD(x) \), for \( x = 0 \) return value is NA.

**Author(s)**

Jakob Dambon

**Examples**

```r
d.SCAD(-5:5)
d.SCAD(-2:2, d.side = "LHS")
curve(d.SCAD, from = -5, to = 5)
```
fitted.SVC_mle

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 "scale".
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.

Examples

```r
# number of SVC
p <- 3
# sqrt of total number of observations
m <- 20
# covariance parameters
(pars <- data.frame(var = c(0.1, 0.2, 0.3),
                    scale = c(0.3, 0.1, 0.2)))
nugget.var <- 0.05

# function to sample SVCs
sp.SVC <- fullSVC_reggrid(m = m, p = p,
                          cov_pars = pars,
                          nugget = nugget.var)

library(sp)
# visualization of sampled SVC
spplot(sp.SVC, colorkey = TRUE)
```

---

**logLik.SVC_mle**

*Extract the Likelihood*

Description

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

Usage

```r
## 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

---

**Lq**

$L^q$ Norm Penalty

**Description**

Penalty function using the $L^q$ norm, i.e., $p_\lambda(x) = \lambda \|x\|^q$.

**Usage**

Lq(x, lambda = 1, q = 1)

**Arguments**

x numeric.

lambda non-negative scalar, shrinkage parameter.

q non-negative scalar, norm parameter.

**Value**

penalty for values of x.

**Author(s)**

Jakob Dambon

**Examples**

Lq(-5:5)

curve(Lq(x, q = 2), from = -5, to = 5)
nlocs

**Extract 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**

Extract 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

**Extract Number of Observations**

**Description**

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

**Usage**

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

**Arguments**

- **object**: SVC_mle object
- **...**: further arguments

**Value**

an integer of number of observations
plot.SVC_mle

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

```r
#' # ---- toy example ----
#' # sample data
#' # setting seed for reproducibility
#' set.seed(123)
#' m <- 7
#' # number of observations
#' n <- m^2
#' # 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

```r
## 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
predict.SVC_mle

See Also

SVC_mle

Examples

```r
## ---- 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 Method for SVC_mle

### Description

Method to print an `SVC_mle` object.

### Usage

```r
## 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

## Extract 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

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

### Arguments

- `object` : `SVC_mle` object
- `...` : further arguments

### Value

numeric, residuals of model

### Author(s)

Jakob Dambon
SCAD

Smoothly Clipped Absolute Deviation Penalty

Description


Usage

SCAD(x, lambda = 1, a = 3.7)

Arguments

x
numeric.

lambda
non-negative scalar, shrinkage parameter.

a
scalar larger than 2. Fan & Li (2001) suggest $a = 3.7$.

Value

penalty for values of x.

Author(s)

Jakob Dambon

References


Examples

SCAD(-5:5)
curve(SCAD, from = -5, to = 5)
summary.SVC_mle  Summary Method for SVC_mle

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

Usage
```r
## 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
- \( \mu \) is the vector containing the fixed effects
- W is the data matrix for the SVCs represented by zero mean GRF
- \( \eta \) are the SVCs represented by zero mean GRF
- \( \epsilon \) 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 parameters, 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 if control$extract_fun is FALSE, meaning that a MLE has been conducted. Otherwise, if control$extract_fun is TRUE, the function return a list with the objective function being used in the optimization (named obj_fun) and the arguments to call it (named args). For further details, see description of SVC_mle_control.

Author(s)

Jakob Dambon
## Examples

```r
## ---- toy example ----
## sample data
# setting seed for reproducibility
set.seed(123)
m <- 7
# number of observations
n <- m^2
# 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)
class(fit)

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

## extract the optimization arguments, including objective function
control$extract_fun <- TRUE
opt <- SVC_mle(y = y, X = X, locs = locs,
               control = control)
```

See Also

`predict.SVC_mle`
# objective function and its arguments of optimization
class(opt$obj_fun)
class(opt$args)

# single evaluation with initial value
do.call(opt$obj_fun,
  c(list(x = control$init), opt$args))

## ---- 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(
    # initial 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)
Description

Function to set up control parameters for \texttt{SVC\_mle}

Usage

\texttt{SVC\_mle\_control(...)}

## Default S3 method:
\texttt{SVC\_mle\_control(}
\texttt{  cov.name = c("exp", "sph"),}
\texttt{  tapering = NULL,}
\texttt{  parallel = NULL,}
\texttt{  init = NULL,}
\texttt{  lower = NULL,}
\texttt{  upper = NULL,}
\texttt{  save.fitted = TRUE,}
\texttt{  profileLik = FALSE,}
\texttt{  mean.est = c("GLS", "OLS"),}
\texttt{  pc.prior = NULL,}
\texttt{  extract.fun = FALSE,}
\texttt{  hessian = FALSE,}
\texttt{  ...}
\texttt{)}

## S3 method for class 'SVC\_mle'
\texttt{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.

parallel list with arguments for parallelization, see documentation of \texttt{optimParallel}

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 separately. "GLS" uses the generalized least square estimate while "OLS" uses the ordinary least squares estimate.

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 by Fulgstad et al. (2018) doi: 10.1080/01621459.2017.1415907.

extract_fun logical. If TRUE, the function call of SVC_mle stops before the MLE and gives back the objective function of the MLE as well as all used arguments. If FALSE, regular MLE is conducted.

hessian logical, feault is FALSE. Gives back Hessian matrix, see optim.

object An object of class SVC_mle. The function then extracts the control settings from the particular function call used to compute object.

Details

The argument extract_fun is useful, when one wants to modify the objective function. Further, when trying to parallelize the optimization, it is useful to check whether a single evaluation of the objective function takes longer than 0.05 seconds, cf. Gerber and Furrer (2019) doi: 10.32614/RJ-2019030. Platform specific issues can be sorted out by the user by setting up their own optimization.

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)
Description

This package offers functions to 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 processes with (currently) 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 high level of flexibility.

Details

The underlying methodology is described in Dambon et al. (2020) https://arxiv.org/abs/2001.08089, where further details can be found.

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-specific 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").

Author(s)

Jakob Dambon

Examples

vignette("manual", package = "varycoef")
methods(class = "SVC_mle")
Index

c coef.SVC_mle, 2
cov_par, 3
d.Lq, 3
d.SCAD, 4
fitted.SVC_mle, 5
fullSVC_reggrid, 5
legend, 9
lm, 22
loess, 22
logLik.SVC_mle, 6
Lq, 3, 7
nlocs, 8
nobs.SVC_mle, 8
optim, 17, 21
optimParallel, 20
plot.SVC_mle, 9
predict.SVC_mle, 11, 18
print.summary.SVC_mle, 13
print.SVC_mle, 14
residuals.SVC_mle, 14
SCAD, 4, 15
summary.SVC_mle, 13, 16
SVC_mle, 2, 3, 5, 6, 8, 9, 11–14, 16, 16, 20–22
SVC_mle_control, 3, 9, 17, 19
varycoef, 22