Package: BayesGP (via r-universe)

June 7, 2026

Type Package

Title Efficient Implementation of Gaussian Process in Bayesian Hierarchical Models

Version 0.1.3

Description Implements Bayesian hierarchical models with flexible Gaussian process priors, focusing on Extended Latent Gaussian Models and incorporating various Gaussian process priors for Bayesian smoothing. Computations leverage finite element approximations and adaptive quadrature for efficient inference. Methods are detailed in Zhang, Stringer, Brown, and Stafford (2023) <doi:10.1177/09622802221134172>; Zhang, Stringer, Brown, and Stafford (2024) <doi:10.1080/10618600.2023.2289532>; Zhang, Brown, and Stafford (2023) <doi:10.48550/arXiv.2305.09914>; and Stringer, Brown, and Stafford (2021) <doi:10.1111/biom.13329>.

License GPL (>= 3)

Encoding UTF-8

LazyData true

RoxygenNote 7.3.2

LinkingTo TMB (>= 1.9.7), RcppEigen

Imports TMB (>= 1.9.7), numDeriv, rstan, sfsmisc, Matrix (>= 1.6.3), aghq (>= 0.4.1), fda, tmbstan, LaplacesDemon, methods

Depends R (>= 3.6.0)

Suggests rmarkdown, knitr, survival, testthat (>= 3.0.0)

VignetteBuilder knitr

NeedsCompilation yes

Config/testthat/edition 3

Config/pak/sysreqs make

Repository https://bayes-gp.r-universe.dev

Date/Publication 2024-12-05 02:36:30 UTC

RemoteUrl https://github.com/bayes-gp/bayesgp

RemoteRef HEAD

RemoteSha 7db1d65825a795f0e74d25fc95d053ce96a0aff4

Contents

ccData . . . 2
compute_d_step_sgpsd . . . 3
compute_post_fun_iwp . . . 3
compute_post_fun_sgp . . . 4
compute_weights_precision . . . 6
compute_weights_precision_helper . . . 6
covid_canada . . . 7
custom_template . . . 7
dummy . . . 8
extract_mean_interval_given_samps . . . 8
f . . . 9
get_default_option_list_MCMC . . . 10
global_poly_helper . . . 10
global_poly_helper_sgp . . . 11
local_poly_helper . . . 12
model_fit . . . 12
model_fit_loop . . . 14
para_density . . . 15
para_samps . . . 16
PEN_death . . . 16
post_table . . . 17
predict.FitResult . . . 17
prior_conversion_iwp . . . 18
prior_conversion_sgp . . . 19
sample_fixed_effect . . . 19
sample_sd . . . 20
sd_density . . . 21
sd_plot . . . 21

Index **23**

`ccData` *A simulated dataset from the case-crossover model.*

Description

A simulated dataset from the case-crossover model.

Usage

```
ccData
```

Format

'ccData' A data frame with 3596 rows and 6 columns.

`compute_d_step_sgpsd` *Compute the SD correction factor for sgp*

Description

Compute the SD correction factor for sgp

Usage

```
compute_d_step_sgpsd(d, a)
```

Arguments

`d`	A numeric value for the prediction step.
`a`	The frequency parameter of the sgp.

Value

The correction factor c that should be used to compute the d-step PSD as c*SD.

`compute_post_fun_iwp` *Computing the posterior samples of the function or its derivative using the posterior samples of the basis coefficients for iwp*

Description

Computing the posterior samples of the function or its derivative using the posterior samples of the basis coefficients for iwp

Usage

```
compute_post_fun_iwp(
  samps,
  global_samps = NULL,
  knots,
  refined_x,
  p,
  degree = 0,
  intercept_samps = NULL
)
```

Arguments

samps	A matrix that consists of posterior samples for the O-spline basis coefficients. Each column represents a particular sample of coefficients, and each row is associated with one basis function. This can be extracted using ‘sample_marginal‘ function from ‘aghq‘ package.
global_samps	A matrix that consists of posterior samples for the global basis coefficients. If NULL, assume there will be no global polynomials and the boundary conditions are exactly zero.
knots	A vector of knots used to construct the O-spline basis, first knot should be viewed as "0", the reference starting location. These k knots will define (k-1) basis function in total.
refined_x	A vector of locations to evaluate the O-spline basis
p	An integer value indicates the order of smoothness
degree	The order of the derivative to take, if zero, implies to consider the function itself.
intercept_samps	A matrix that consists of posterior samples for the intercept parameter. If NULL, assume the function evaluated at zero is zero.

Value

A data.frame that contains different samples of the function or its derivative, with the first column being the locations of evaluations x = refined_x.

Examples

```
knots <- c(0, 0.2, 0.4, 0.6)
samps <- matrix(rnorm(n = (3 * 10)), ncol = 10)
result <- compute_post_fun_iwp(samps = samps, knots = knots, refined_x = seq(0, 1, by = 0.1), p = 2)
plot(result[, 2] ~ result$x, type = "l", ylim = c(-0.3, 0.3))
for (i in 1:9) {
  lines(result[, (i + 1)] ~ result$x, lty = "dashed", ylim = c(-0.1, 0.1))
}
global_samps <- matrix(rnorm(n = (2 * 10), sd = 0.1), ncol = 10)
```

---

compute_post_fun_sgp *Computing the posterior samples of the function using the posterior samples of the basis coefficients for sGP*

Description

Computing the posterior samples of the function using the posterior samples of the basis coefficients for sGP

Usage

```
compute_post_fun_sgp(
  samps,
  global_samps = NULL,
  k,
  refined_x,
  a,
  region,
  boundary = TRUE,
  m,
  intercept_samps = NULL,
  initial_location = NULL
)
```

Arguments

samps	A matrix that consists of posterior samples for the O-spline basis coefficients. Each column represents a particular sample of coefficients, and each row is associated with one basis function. This can be extracted using 'sample_marginal' function from 'aghq' package.
global_samps	A matrix that consists of posterior samples for the global basis coefficients. If NULL, assume there will be no global polynomials and the boundary conditions are exactly zero.
k	The number of the sB basis.
refined_x	A vector of locations to evaluate the sB basis
a	The frequency of sGP.
region	The region to define the sB basis
boundary	A boolean variable to indicate whether the boundary condition should be considered in the prediction.
m	The number of harmonics to consider
intercept_samps	A matrix that consists of posterior samples for the intercept parameter. If NULL, assume there is no intercept samples to adjust.
initial_location	The initial location of the sGP.

Value

A data.frame that contains different samples of the function, with the first column being the locations of evaluations x = refined_x.

`compute_weights_precision` *Constructing the precision matrix given the knot sequence*

Description

Constructing the precision matrix given the knot sequence

Usage

```
compute_weights_precision(object)
```

Arguments

`object`	A fitted model object.

Value

A precision matrix of the corresponding basis function, should be diagonal matrix with size (k-1) by (k-1).

`compute_weights_precision_helper` *Constructing the precision matrix given the knot sequence (helper)*

Description

Constructing the precision matrix given the knot sequence (helper)

Usage

```
compute_weights_precision_helper(x)
```

Arguments

`x`	A vector of knots used to construct the O-spline basis, first knot should be viewed as "0", the reference starting location. These k knots will define (k-1) basis function in total.

Value

A precision matrix of the corresponding basis function, should be diagonal matrix with size (k-1) by (k-1).

Examples

```
compute_weights_precision_helper(x = c(0,0.2,0.4,0.6,0.8))
```

covid_canada *The COVID-19 daily death data in Canada.*

Description

A subset of the the COVID-19 daily death data collected between 2020 to 2022. The data is obtained from COVID-19 Data Repository by the Center for Systems Science and Engineering (CSSE) at Johns Hopkins University (Dong et al., 2020).

Usage

```
covid_canada
```

Format

'covid_canada' A data frame with 787 rows and 5 columns:

Date The date of the measurement.

new_deaths The number of new deaths at that date.

t The converted numerical value of 'Date'.

weekdays 1-6 Coded as 1 if the date is the corresponding weekday, -1 else if the date is on Sunday, and 0 otherwise.

index The index of that observation.

Source

Dong, E., H. Du, and L. Gardner (2020). An interactive web-based dashboard to track covid-19 in real time. *The Lancet infectious diseases 20 (5)*, 533–534.

custom_template *Custom Template Function*

Description

This function allows for the dynamic modification of a C++ template within the BayesGP package. Users can specify custom content for the log-likelihood as well as the log-prior of the SD parameter in the template.

Usage

```
custom_template(
  setup = NULL,
  log_likelihood = NULL,
  log_prior = NULL,
  compile_template = FALSE
)
```

Arguments

setup	A character string or vector containing the lines of C++ code to be inserted before the computation of log-likelihood.
log_likelihood	A character string or vector containing the lines of C++ code to be inserted in the log-likelihood section of the template. Should be NULL if no changes are to be made to this section.
log_prior	A character string or vector containing the lines of C++ code to be inserted in the log-prior (of the SD parameter) section of the template. Should be NULL if no changes are to be made to this section.
compile_template	A indicator of whether the new template should be compiled. default is FALSE.

Value

A string representing the path to the temporary .so (or .dll) file containing the compiled custom C++ code.

---

dummy *Roxygen commands*

Description

Roxygen commands

Usage

```
dummy()
```

---

extract_mean_interval_given_samps *Construct posterior inference given samples*

Description

Construct posterior inference given samples

Usage

```
extract_mean_interval_given_samps(samps, level = 0.95, quantiles = NULL)
```

Arguments

samps	Posterior samples of f or its derivative, with the first column being evaluation points x. This can be yielded by 'compute_post_fun_iwp' function.
level	The level to compute the pointwise interval. Ignored when quantiles are provided.
quantiles	A numeric vector of quantiles to be computed.

Value

A dataframe with a column for evaluation locations x, and posterior mean and pointwise intervals at that set of locations.

---

f *Function defined to enhance the usability for users on IDEs.*

Description

Function defined to enhance the usability for users on IDEs.

Usage

```
f(
  smoothing_var,
  model = "iid",
  sd.prior = NULL,
  boundary.prior = NULL,
  initial_location = c("median", "min", "max")
)
```

Arguments

smoothing_var	The variable name of the smoothing variable.
model	The name of the smoothing model.
sd.prior	A list/vector that specifies the prior of the sd parameter.
boundary.prior	A list/vector that specifies the prior of the boundary parameter.
initial_location	A character/number that specifies the initial location of the smoothing variable.

Value

A call object that can be used in the model_fit formula to indicate a smooth term or random effect.

get_default_option_list_MCMC *Get default options for MCMC implementation*

Description

This function takes an optional list of options and fills in any missing values with a set of default MCMC options.

Usage

```
get_default_option_list_MCMC(option_list = list())
```

Arguments

option_list	A list of options to be passed to the MCMC. If some options are missing, the function will use default values.

Value

A list containing the complete set of options with defaults where necessary.

Examples

```
# Example: Get the default option list
options <- get_default_option_list_MCMC()
print(options)
```

global_poly_helper *Constructing and evaluating the global polynomials, to account for boundary conditions (design matrix)*

Description

Constructing and evaluating the global polynomials, to account for boundary conditions (design matrix)

Usage

```
global_poly_helper(x, p = 2)
```

Arguments

x	A vector of locations to evaluate the global polynomials
p	An integer value indicates the order of smoothness

Value

A matrix with i,j componet being the value of jth basis function value at ith element of x, the ncol should equal to p, and nrow should equal to the number of elements in x

Examples

```
global_poly_helper(x = c(0, 0.2, 0.4, 0.6, 0.8), p = 2)
```

`global_poly_helper_sgp` *Constructing and evaluating the global polynomials, to account for boundary conditions (design matrix) of sgp*

Description

Constructing and evaluating the global polynomials, to account for boundary conditions (design matrix) of sgp

Usage

```
global_poly_helper_sgp(refined_x, a, m, initial_location = NULL)
```

Arguments

`refined_x`	A vector of locations to evaluate the sB basis
`a`	The frequency of sgp.
`m`	The number of harmonics to consider
`initial_location`	The value of the initial location. If NULL, the minimum value of refined_x will be used.

Value

A matrix with i,j componet being the value of jth basis function value at ith element of x, the ncol should equal to (2*m), and nrow should equal to the number of elements in x

local_poly_helper *Constructing and evaluating the local O-spline basis (design matrix)*

Description

Constructing and evaluating the local O-spline basis (design matrix)

Usage

```
local_poly_helper(knots, refined_x, p = 2, neg_sign_order = 0)
```

Arguments

knots	A vector of knots used to construct the O-spline basis, first knot should be viewed as "0", the reference starting location. These k knots will define (k-1) basis function in total.
refined_x	A vector of locations to evaluate the O-spline basis
p	An integer value indicates the order of smoothness
neg_sign_order	An integer value N such that D = ((-1)^N)*D for the splines at negative knots. Default is 0.

Value

A matrix with i,j component being the value of jth basis function value at ith element of refined_x, the ncol should equal to number of knots minus 1, and nrow should equal to the number of elements in refined_x.

Examples

```
local_poly_helper(knots = c(0, 0.2, 0.4, 0.6, 0.8), refined_x = seq(0, 0.8, by = 0.1), p = 2)
```

model_fit *Model fitting with random effects/fixed effects*

Description

Fitting a hierarchical model based on the provided formula, data and parameters such as type of method and family of response. Returning the S4 objects for the random effects, concatenated design matrix for the intercepts and fixed effects, fitted model, indexes to partition the posterior samples.

Usage

```
model_fit(
  formula,
  data,
  method = "aghq",
  family = "gaussian",
  control.family,
  control.fixed,
  aghq_k = 4,
  size = NULL,
  cens = NULL,
  weight = NULL,
  strata = NULL,
  M = 3000,
  customized_template = NULL,
  Customized_RE = NULL,
  option_list = list(),
  envir = parent.frame(),
  extra_theta_num = NULL
)
```

Arguments

formula	A formula that contains one response variable, and covariates with either random or fixed effect.
data	A dataframe that contains the response variable and other covariates mentioned in the formula.
method	The inference method used in the model. By default, the method is set to be "aghq".
family	The family of response used in the model. By default, the family is set to be "gaussian".
control.family	Parameters used to specify the priors for the family parameters, such as the standard deviation parameter of Gaussian family. For example control.family = 1 in the Gaussian family corresponds to an Exponential prior to the standard deviation parameter of the Gaussian noise with median 1. When left unspecified, the default prior is an Exponential prior with median 1.
control.fixed	Parameters used to specify the priors for the fixed effects. For example control.fixed = list(intercept = list(prec = 0.001, mean = 0)) will setup the prior N(0,1/0.001) for the intercept parameter. When left unspecified, all fixed effect parameters will be assigned independent N(0,1/0.001) priors.
aghq_k	An integer to specify the number of quadrature points used in the aghq method. By default, the value is 4.
size	The name of the size variable, should be one of the variables in ‘data‘. The default value is "NULL", corresponding to a vector of 1s. This is only used for the Binomial family, and denotes the number of binomial trails.

`cens` The name of the right-censoring indicator, should be one of the variables in 'data'. The default value is "NULL". This is only used for the CoxPH family.

`weight` The name of the weight variable, should be one of the variables in 'data'. The default value is "NULL", corresponding to a vector of 1s. This is only used for the Case-Crossover family, and denotes the weight of each observation.

`strata` The name of the strata variable, should be one of the variables in 'data'. The default value is "NULL". This is only used for the Case-Crossover family, and denotes the strata of each observation.

`M` The number of posterior samples to be taken, by default is 3000.

`customized_template`
The name of the customized cpp template that the user wants to use instead. By default this is NULL, and the cpp template 'BayesGP' will be used.

`Customized_RE` The list that contains the compute_B and compute_P functions for the customized random effect. By default, this is NULL and there is not customized random effect in the model.

`option_list` A list that controls the details of the inference algorithm, by default is an empty list.

`envir` The environment in which the formula and other expressions are to be evaluated. Defaults to 'parent.frame()', which refers to the environment from which the function was called. This allows the function to access variables that are defined in the calling function's scope.

`extra_theta_num`
An integer number to indicate the extra number of parameters required in the 'theta' vector. Should only be specified when using customized template.

Value

A list that contains following items: the S4 objects for the random effects (instances), concatenated design matrix for the fixed effects (design_mat_fixed), fitted aghq (mod) and indexes to partition the posterior samples (boundary_samp_indexes, random_samp_indexes and fixed_samp_indexes).

`model_fit_loop` *Repeated fitting Bayesian Hierarchical Models for a sequence of values of the looping variable.*

Description

Performs repeated model fitting over a sequence of values for a specified variable within a hierarchical model. This function repeatedly fits a model for each value of the looping variable, compiles the log marginal likelihoods, and calculates the posterior probabilities for the variable's values.

Usage

```
model_fit_loop(
  loop_holder = "LOOP",
  loop_values,
  prior_func = function(x) {
     1
 },
  parallel = FALSE,
  cores = (parallel::detectCores() - 1),
  ...
)
```

Arguments

loop_holder	A string specifying the name of the variable to loop over. The default value is 'LOOP'.
loop_values	A numeric vector containing the values to loop over for the specified variable.
prior_func	A function that takes the specified loop_values and returns the values of the prior for the loop variable. By default, it is a uniform prior which returns a constant value, indicating equal probability for all values.
parallel	Logical, indicating whether or not to run the model fitting in parallel (default is FALSE).
cores	The number of cores to use for parallel execution (default is detected cores - 1).
...	Additional arguments passed to the model fitting function 'model_fit'.

Value

A data frame containing the values of the looping variable, their corresponding log marginal likelihoods, and posterior probabilities.

---

para_density *Obtain the posterior and prior density of all the parameters in the fitted model*

Description

Obtain the posterior and prior density of all the parameters in the fitted model

Usage

```
para_density(object, variables = NULL)
```

Arguments

object	The fitted object from the function 'model_fit'.
variables	The name of the parameter that you want to show the posterior density. By default this is 'NULL', indicating all the parameters are of interest.

Value

A list of data frames, each data frame contains the posterior density of the corresponding parameter.

`para_samps` *Extract the posterior samples from the fitted model for the target parameters.*

Description

Extract the posterior samples from the fitted model for the target parameters.

Usage

```
para_samps(model_fit, variables, M = NULL, h = NULL)
```

Arguments

`model_fit`	The result from model_fit().
`variables`	A vector of names of the target parameters to sample.
`M`	The number of samples to draw from the posterior distribution.
`h`	The unit of predictive step to consider, by default is set to 'NULL', indicating the result is using the same 'h' as in the model fitting.

Value

A matrix with columns being the posterior samples of the target parameters.

`PEN_death` *The monthly all-cause mortality for male with age less than 40 in Pennsylvania.*

Description

The monthly all-cause mortality for male with age less than 40 in Pennsylvania.

Usage

```
PEN_death
```

Format

'PEN_death' A data frame with 299 rows of observations.

post_table *Obtain the posterior summary table for all the parameters in the fitted model*

Description

Obtain the posterior summary table for all the parameters in the fitted model

Usage

```
post_table(object, quantiles = c(0.025, 0.975), digits = 3)
```

Arguments

object	The fitted object from the function ‘model_fit‘.
quantiles	The specified quantile to display the posterior summary, default is c(0.025, 0.975).
digits	The significant digits to be kept in the result, default is 3.

Value

A data frame that contains the posterior summary of all the parameters in the fitted model.

predict.FitResult *To predict the GP component in the fitted model, at the locations specified in ‘newdata‘.*

Description

To predict the GP component in the fitted model, at the locations specified in ‘newdata‘.

Usage

```
## S3 method for class 'FitResult'
predict(
  object,
  newdata = NULL,
  variable,
  deriv = 0,
  include.intercept = TRUE,
  only.samples = FALSE,
  quantiles = c(0.025, 0.5, 0.975),
  boundary.condition = "Yes",
  ...
)
```

Arguments

object	The fitted object from the function ‘model_fit‘.
newdata	The dataset that contains the locations to be predicted for the specified GP. Its column names must include ‘variable‘.
variable	The name of the variable to be predicted, should be in the ‘newdata‘.
deriv	The degree of derivative that the user specifies for inference. Only applicable for a GP in the ‘iwp‘ type.
include.intercept	A logical variable specifying whether the intercept should be accounted when doing the prediction. The default is TRUE. For Coxph model, this variable will be forced to FALSE.
only.samples	A logical variable indicating whether only the posterior samples are required. The default is FALSE, and the summary of posterior samples will be reported.
quantiles	A numeric vector of quantiles that predict.FitResult will produce, the default is c(0.025, 0.5, 0.975).
boundary.condition	A string specifies whether the boundary.condition should be considered in the prediction, should be one of c("yes", "no", "only"). The default option is "Yes".
...	Other arguments to be passed to the function.

Value

A data.frame that contains the posterior mean and pointwise intervals (or posterior samples) at the locations specified in ‘newdata‘.

---

prior_conversion_iwp *Construct prior based on d-step prediction SD (for iwp)*

Description

Construct prior based on d-step prediction SD (for iwp)

Usage

```
prior_conversion_iwp(d, prior, p)
```

Arguments

d	A numeric value for the prediction step.
prior	A list that contains alpha and u. This specifies the target prior on the d-step SD $\sigma(d)$, such that $P(\sigma(d) > u) = alpha$.
p	An integer for the order of iwp.

Value

A list that contains alpha and u. The prior for the smoothness parameter σ such that $P(\sigma > u) = alpha$, that yields the ideal prior on the d-step SD.

---

prior_conversion_sgp *Construct prior based on d-step prediction SD (for sgp)*

Description

Construct prior based on d-step prediction SD (for sgp)

Usage

```
prior_conversion_sgp(d, prior, freq, period, a, m = 1)
```

Arguments

d	A numeric value for the prediction step.
prior	A list that contains alpha and u. This specifies the target prior on the d-step SD $\sigma(d)$, such that $P(\sigma(d) > u) = alpha$.
freq	The frequency of the sgp, ignored if a is provided.
period	The period of the sgp, ignored if a or freq is provided.
a	The frequency parameter of the sgp.
m	The number of harmonics that should be considered, by default m = 1 represents only the sgp.

Value

A list that contains alpha and u. The prior for the smoothness parameter σ such that $P(\sigma > u) = alpha$, that yields the ideal prior on the d-step SD.

---

sample_fixed_effect *Extract the posterior samples from the fitted model for the target fixed variables.*

Description

Extract the posterior samples from the fitted model for the target fixed variables.

Usage

```
sample_fixed_effect(model_fit, variables, M = NULL)
```

Arguments

model_fit	The result from model_fit().
variables	A vector of names of the target fixed variables to sample.
M	The number of samples to draw from the posterior distribution.

Value

A matrix with columns being the posterior samples of the target fixed effect variables.

---

sample_sd *Extract the posterior samples from the fitted model for the target standard deviation.*

Description

Extract the posterior samples from the fitted model for the target standard deviation.

Usage

```
sample_sd(model_fit, variables, M = NULL, h = NULL)
```

Arguments

model_fit	The result from model_fit().
variables	A vector of names of the target standard deviation to sample.
M	The number of samples to draw from the posterior distribution.
h	The unit of predictive step to consider, by default is set to 'NULL', indicating the result is using the same 'h' as in the model fitting.

Value

A matrix with columns being the posterior samples of the target sd variables.

sd_density *Obtain the posterior density of a SD parameter in the fitted model*

Description

Obtain the posterior density of a SD parameter in the fitted model

Usage

```
sd_density(
  object,
  component = NULL,
  h = NULL,
  theta_logprior = NULL,
  MCMC_samps_only = FALSE
)
```

Arguments

object	The fitted object from the function 'model_fit'.
component	The component of the SD parameter that you want to show. By default this is 'NULL', indicating the family.sd is of interest.
h	For PSD, the unit of predictive step to consider, by default is set to 'NULL', indicating the result is using the same 'h' as in the model fitting.
theta_logprior	The log prior function used on the selected SD parameter. By default is 'NULL', and the current Exponential prior will be used.
MCMC_samps_only	For model fitted with MCMC, whether only the posterior samples are needed.

Value

A data.frame that contains the posterior and prior densities of the SD parameter. The SD parameter will be converted into PSD if applicable.

sd_plot *Plot the posterior density of a SD parameter in the fitted model*

Description

Plot the posterior density of a SD parameter in the fitted model

Usage

```
sd_plot(object, component = NULL, h = NULL, theta_logprior = NULL)
```

Arguments

`object`	The fitted object from the function 'model_fit'.
`component`	The component of the SD parameter that you want to show. By default this is 'NULL', indicating the family.sd is of interest.
`h`	For PSD, the unit of predictive step to consider, by default is set to 'NULL', indicating the result is using the same 'h' as in the model fitting.
`theta_logprior`	The log prior function used on the selected SD parameter. By default is 'NULL', and the current Exponential prior will be used.

Value

A plot that shows the posterior and prior density of the SD parameter. The SD parameter will be converted into PSD if applicable.

Index

∗ **datasets**
ccData, 2
covid_canada, 7
PEN_death, 16

ccData, 2
compute_d_step_sgpsd, 3
compute_post_fun_iwp, 3
compute_post_fun_sgp, 4
compute_weights_precision, 6
compute_weights_precision_helper, 6
covid_canada, 7
custom_template, 7

dummy, 8

extract_mean_interval_given_samps, 8

f, 9

get_default_option_list_MCMC, 10
global_poly_helper, 10
global_poly_helper_sgp, 11

local_poly_helper, 12

model_fit, 12
model_fit_loop, 14

para_density, 15
para_samps, 16
PEN_death, 16
post_table, 17
predict.FitResult, 17
prior_conversion_iwp, 18
prior_conversion_sgp, 19

sample_fixed_effect, 19
sample_sd, 20
sd_density, 21
sd_plot, 21