Package: NBKP (via r-universe)

June 17, 2026

Type Package

Title Beta Kernel Process Modeling with Negative Binomial Response

Version 0.1.0

Description An extension of the Beta Kernel Process model designed to handle negative binomial responses for count data modeling, building upon Zhao, Qing and Xu (2025) <doi:10.48550/arXiv.2508.10447>.

License GPL-3

Encoding UTF-8

RoxygenNote 7.3.3

Depends R (>= 4.0.0)

Imports stats,graphics,utils,dirmult,gridExtra,lattice,optimx,tgp,lhs

NeedsCompilation no

Author Xueqin Li [aut], Xingyue Fa [aut], Shanqing Yin [aut, cre], Yi Zhuo [aut]

Maintainer Shanqing Yin <2191983548@qq.com>

Repository https://cran.r-universe.dev

Date/Publication 2026-06-17 19:33:16 UTC

RemoteUrl https://github.com/cran/NBKP

RemoteRef HEAD

RemoteSha 275d6db5845ee1aa695750cd4626ab62b67e12b8

Contents

`fit_NBKP` *Fit a Negative Binomial Kernel Process (NBKP) Model*

Description

Fits a Negative Binomial Kernel Process (NBKP) model to count response data using local kernel smoothing. The method constructs a flexible latent mean surface by updating Gamma priors with kernel-weighted observations.

Usage

```
fit_NBKP(
  X,
  y,
  Xbounds = NULL,
  prior = c("noninformative", "fixed", "adaptive"),
  r0 = 0.1,
  mu0 = mean(y),
  kernel = c("gaussian", "matern52", "matern32"),
  loss = c("mse", "nll"),
  n_multi_start = NULL,
  theta = NULL
)
```

Arguments

`X`	A numeric input matrix of size $n \times d$, where each row corresponds to a covariate vector.
`y`	A numeric vector of observed counts (length `n`).
`Xbounds`	Optional $d \times 2$ matrix specifying the lower and upper bounds of each input dimension. Used to normalize inputs to $[0, 1]^d$. If `NULL`, inputs are assumed to be pre-normalized, and default bounds $[0, 1]^d$ are applied.
`prior`	Global prior type: `"noninformative"` (default), `"fixed"`, or `"adaptive"`.
`r0`	Global prior precision (used when `prior = "fixed"` or `"adaptive"`).
`mu0`	Global prior mean (used when `prior = "fixed"`). Default is `mean(y)`.
`kernel`	Kernel function for local weighting: `"gaussian"` (default), `"matern52"`, or `"matern32"`.

loss	Loss function for kernel hyperparameter tuning: "mse" (default) or "nll".
n_multi_start	Number of random initializations for multi-start optimization. Default is $10 \times d$.
theta	Optional. A positive scalar or numeric vector of length d specifying kernel lengthscale parameters directly. If NULL (default), lengthscales are optimized using multi-start L-BFGS-B to minimize the specified loss.

Value

A list of class "NBKP" containing the fitted NBKP model, including:

theta_opt	Optimized kernel hyperparameters (lengthscales).
kernel	Kernel function used, as a string.
loss	Loss function used for hyperparameter tuning.
loss_min	Minimum loss achieved during optimization, or NA if theta was user-specified.
X	Original input matrix ($n \times d$).
Xnorm	Normalized input matrix scaled to $[0, 1]^d$.
Xbounds	Normalization bounds for each input dimension ($d \times 2$).
y	Observed counts.
phi	Estimated negative binomial dispersion parameter.
prior	Type of prior used.
r0	Prior precision parameter.
mu0	Prior mean (for fixed priors).
alpha0	Prior Gamma shape parameter $\alpha_0(\mathbf{x})$.
beta0	Prior Gamma rate parameter $\beta_0(\mathbf{x})$.
alpha_n	Posterior shape parameter $\alpha_n(\mathbf{x})$.
beta_n	Posterior rate parameter $\beta_n(\mathbf{x})$.

Author(s)

Xueqin Li

References

Zhao J, Qing K, Xu J (2025). BKP: An R Package for Beta Kernel Process Modeling.

See Also

predict.NBKP, plot.NBKP, summary.NBKP

Examples

```
# -------------------------- 1D Example --------------------------
set.seed(123)

true_mu_fun <- function(x) {
  exp(sin(x) + 0.5)
}

n <- 30
Xbounds <- matrix(c(-2, 2), nrow=1)
X <- matrix(seq(-2, 2, length.out = n))
true_mu <- true_mu_fun(X)
y <- rnbinom(n, size = 1, mu = true_mu)

model1 <- fit_NBKP(X, y, Xbounds=Xbounds)
print(model1)
```

`fitted.NBKP` *Extract NBKP Model Fitted Values*

Description

Compute the posterior fitted values from a fitted `NBKP` object. For a `NBKP` object, this returns the posterior mean count from the Gamma-Negative Binomial conjugate posterior.

Usage

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

Arguments

`object`	An object of class `NBKP`, typically the result of a call to `fit_NBKP`.
`...`	Additional arguments (currently unused).

Details

For a `NBKP` model, the fitted values correspond to the posterior mean count, computed from the Gamma Kernel Process posterior distribution.

Value

A numeric vector containing posterior mean count estimates at the training inputs.

References

Zhao J, Qing K, Xu J (2025). *BKP: An R Package for Beta Kernel Process Modeling*. arXiv. https://doi.org/10.48550/arxiv.2508.10447.

Examples

```
set.seed(123)

# Define true mean function
true_mu_fun <- function(x) {
  exp(sin(x) + 0.5)
}

n <- 30
Xbounds <- matrix(c(-2, 2), nrow = 1)
X <- tgp::lhs(n = n, rect = Xbounds)
true_mu <- true_mu_fun(X)
y <- rnbinom(n, size = 1, mu = true_mu)

# Fit NBKP model
model <- fit_NBKP(X, y, Xbounds = Xbounds)

# Extract fitted values
fitted(model)
```

---

get_prior *Construct Prior Parameters for BKP/DKP Models*

Description

Computes prior parameters for Beta Kernel Process (BKP, binary) or Dirichlet Kernel Process (DKP, multi-class) models. Supports three prior strategies: noninformative, fixed, adaptive.

Usage

```
get_prior(
  prior = c("noninformative", "fixed", "adaptive"),
  model = c("BKP", "DKP"),
  r0 = 2,
  p0 = NULL,
  y = NULL,
  m = NULL,
  Y = NULL,
  K = NULL
)
```

Arguments

`prior`	Character string; prior type. One of: ‘"noninformative"‘, ‘"fixed"‘, ‘"adaptive"‘.
`model`	Character string; model type. One of: ‘"BKP"‘ (binary), ‘"DKP"‘ (multi-class).
`r0`	Numeric; prior precision (positive scalar, default = 2).

p0	Numeric; global prior mean. BKP: scalar in (0,1); DKP: vector summing to 1.
y	Numeric vector; observed success counts (BKP only).
m	Numeric vector; number of trials (BKP only, same length as ‘y‘).
Y	Numeric matrix; observed class counts (DKP only, n × q).
K	Numeric matrix; precomputed kernel matrix.

Details

Prior strategies: * ‘noninformative‘: flat prior (Beta(1,1) or Dirichlet(1,...,1)). * ‘fixed‘: global constant prior. * ‘adaptive‘: kernel-smoothed local prior, estimated from observed data.

Value

For BKP: a list with ‘alpha0‘ and ‘beta0‘. For DKP: a matrix ‘alpha0‘ of prior Dirichlet parameters.

Examples

```
# BKP example
set.seed(123)
n <- 10
X <- matrix(runif(n*2), ncol = 2)
y <- rbinom(n, size = 5, prob = 0.6)
m <- rep(5, n)
K <- matrix(1, n, n)
prior_bkp <- get_prior(
  model = "BKP", prior = "adaptive",
  r0 = 2, y = y, m = m, K = K
)
```

---

kernel_matrix *Compute Kernel Matrix Between Input Locations*

Description

Computes the kernel matrix between two sets of input locations using a specified kernel function. Supports both isotropic and anisotropic lengthscales. Available kernels include the Gaussian, Matérn 5/2, and Matérn 3/2.

Usage

```
kernel_matrix(
  X,
  Xprime = NULL,
  theta = 0.1,
  kernel = c("gaussian", "matern52", "matern32"),
  anisotropic = TRUE
)
```

Arguments

X	A numeric matrix (or vector) of input locations with shape $n \times d$.
Xprime	An optional numeric matrix of input locations with shape $m \times d$. If NULL (default), it is set to X, resulting in a symmetric matrix.
theta	A positive numeric value or vector specifying the kernel lengthscale(s). If a scalar, the same lengthscale is applied to all input dimensions. If a vector, it must be of length d, corresponding to anisotropic scaling.
kernel	A character string specifying the kernel function. Must be one of "gaussian", "matern32", or "matern52".
anisotropic	Logical. If TRUE (default), theta is interpreted as a vector of per-dimension lengthscales. If FALSE, theta is treated as a scalar.

Details

Let $\mathbf{x}$ and $\mathbf{x}'$ denote two input points. The scaled distance is defined as

$$r = \left\| \frac{\mathbf{x} - \mathbf{x}'}{\boldsymbol{\theta}} \right\|_2 .$$

The available kernels are defined as:

- **Gaussian:**
$$k(\mathbf{x}, \mathbf{x}') = \exp(-r^2)$$
- **Matérn 5/2:**
$$k(\mathbf{x}, \mathbf{x}') = \left(1 + \sqrt{5}r + \frac{5}{3}r^2\right) \exp(-\sqrt{5}r)$$
- **Matérn 3/2:**
$$k(\mathbf{x}, \mathbf{x}') = \left(1 + \sqrt{3}r\right) \exp(-\sqrt{3}r)$$

The function performs consistency checks on input dimensions and automatically broadcasts theta when it is a scalar.

Value

A numeric matrix of size $n \times m$, where each element K_{ij} gives the kernel similarity between input X_i and X'_j.

References

Zhao J, Qing K, Xu J (2025). *BKP: An R Package for Beta Kernel Process Modeling*. arXiv. https://doi.org/10.48550/arxiv.2508.10447.

Rasmussen, C. E., & Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. MIT Press.

Examples

```
# Basic usage with default Xprime = X
X <- matrix(runif(20), ncol = 2)
K1 <- kernel_matrix(X, theta = 0.2, kernel = "gaussian")

# Anisotropic lengthscales with Matérn 5/2
K2 <- kernel_matrix(X, theta = c(0.1, 0.3), kernel = "matern52")

# Isotropic Matérn 3/2
K3 <- kernel_matrix(X, theta = 1, kernel = "matern32", anisotropic = FALSE)

# Use Xprime different from X
Xprime <- matrix(runif(10), ncol = 2)
K4 <- kernel_matrix(X, Xprime, theta = 0.2, kernel = "gaussian")
```

---

loss_fun *Loss Function for Kernel Process Model Tuning*

Description

Computes prediction loss for kernel process models (BKP binary / DKP multi-class) during kernel hyperparameter tuning. Supports Brier score (MSE) and log-loss (cross-entropy) with leave-one-out cross-validation (LOOCV).

Usage

```
loss_fun(
  gamma,
  Xnorm,
  y = NULL,
  m = NULL,
  Y = NULL,
  model = c("BKP", "DKP"),
  prior = c("noninformative", "fixed", "adaptive"),
  r0 = 2,
  p0 = NULL,
  loss = c("brier", "log_loss"),
  kernel = c("gaussian", "matern52", "matern32")
)
```

Arguments

gamma	Numeric vector; log10-transformed kernel lengthscale parameters.
Xnorm	Numeric matrix; normalized input matrix (values in [0,1]).
y	Numeric vector; observed success counts (BKP model only).

m	Numeric vector; number of trials (BKP model only).
Y	Numeric matrix; observed class counts (DKP model only).
model	Character string; model type: "BKP" (binary) or "DKP" (multi-class).
prior	Character string; prior type: "noninformative", "fixed", "adaptive".
r0	Numeric scalar; prior precision (default = 2).
p0	Numeric; global prior mean (for fixed prior).
loss	Character string; loss type: "brier" (MSE) or "log_loss" (cross-entropy).
kernel	Character string; kernel function: "gaussian", "matern52", "matern32".

Details

This function is used internally to optimize kernel lengthscales. It converts log10 hyperparameters gamma to lengthscales theta, computes the kernel matrix, applies LOOCV (diag(K)=0), and calculates loss between estimated and empirical probabilities.

Value

Numeric scalar; loss value to minimize during optimization.

References

Zhao J, Qing K, Xu J (2025). *BKP: An R Package for Beta Kernel Process Modeling*. arXiv. https://doi.org/10.48550/arXiv.2508.10447

See Also

get_prior, kernel_matrix

Examples

```
# -------------------------- BKP Example --------------------------
set.seed(123)
n <- 10
Xnorm <- matrix(runif(2 * n), ncol = 2)
m <- rep(10, n)
y <- rbinom(n, size = m, prob = runif(n))
loss_fun(gamma = rep(0, 2), Xnorm = Xnorm, y = y, m = m, model = "BKP")

# -------------------------- DKP Example --------------------------
set.seed(123)
n <- 10
q <- 3
Xnorm <- matrix(runif(2 * n), ncol = 2)
Y <- matrix(rmultinom(n, size = 10, prob = rep(1/q, q)), nrow = n, byrow = TRUE)
loss_fun(gamma = rep(0, 2), Xnorm = Xnorm, Y = Y, model = "DKP")
```

`parameter` *Extract Model Parameters from a Fitted NBKP Model*

Description

Retrieve the key model parameters from a fitted `NBKP` object. This includes the optimized kernel hyperparameters, posterior Gamma parameters, and the negative-binomial dispersion parameter.

Usage

```
parameter(object, ...)

## S3 method for class 'NBKP'
parameter(object, ...)
```

Arguments

`object`	An object of class `NBKP`, typically the result of a call to `fit_NBKP`.
`...`	Additional arguments (currently unused).

Value

A named list containing:

- `theta`: Estimated kernel hyperparameters.
- `phi`: Negative-binomial dispersion parameter.
- `alpha_n`: Posterior Gamma α parameters.
- `beta_n`: Posterior Gamma β parameters.

References

Zhao J, Qing K, Xu J (2025). *BKP: An R Package for Beta Kernel Process Modeling*. arXiv. https://doi.org/10.48550/arxiv.2508.10447.

See Also

`fit_NBKP` for fitting NBKP models.

Examples

```
set.seed(123)

# Define true mean function
true_mu_fun <- function(x) {
  exp(sin(x) + 0.5)
}

n <- 30
```

```
Xbounds <- matrix(c(-2, 2), nrow = 1)
X <- tgp::lhs(n = n, rect = Xbounds)
true_mu <- true_mu_fun(X)
y <- rnbinom(n, size = 1, mu = true_mu)

# Fit NBKP model
model <- fit_NBKP(X, y, Xbounds = Xbounds)

# Extract posterior and kernel parameters
parameter(model)
```

`plot.NBKP` *Plot Fitted NBKP Models*

Description

Visualizes fitted NBKP (count) models according to the input dimensionality. For 1D inputs, it shows predicted mean counts with credible intervals and observed count data. For 2D inputs, it generates contour plots of posterior summaries. For higher-dimensional inputs, users must specify which dimensions to plot.

Usage

```
## S3 method for class 'NBKP'
plot(x, only_mean = FALSE, n_grid = 80, dims = NULL, ...)
```

Arguments

x	An object of class "NBKP", typically returned by fit_NBKP.
only_mean	Logical. If TRUE, only the predicted mean surface is plotted for 2D inputs. Default is FALSE.
n_grid	Positive integer specifying the number of grid points per dimension for constructing the prediction grid. Larger values produce smoother and more detailed surfaces, but increase computation time. Default is 80.
dims	Integer vector indicating which input dimensions to plot. Must have length 1 (for 1D) or 2 (for 2D). If NULL (default), all dimensions are used when their number is <= 2.
...	Additional arguments passed to internal plotting routines (currently unused).

Details

The plotting behavior depends on the dimensionality of the input covariates:

- **1D inputs**:

  - The function plots the posterior mean curve with a shaded 95 interval, overlaid with the observed counts.
- **2D inputs**:
  - The function generates contour plots over a 2D prediction grid.
  - Users can choose to plot only the predictive mean surface (`only_mean = TRUE`) or a set of four summary plots (`only_mean = FALSE`):
    1. Predictive mean
    2. 97.5th percentile (upper bound of 95
    3. Predictive variance
    4. 2.5th percentile (lower bound of 95
- **Input dimensions greater than 2**:
  - The function does not automatically support visualization and will terminate with an error.
  - Users must specify which dimensions to visualize via the `dims` argument (length 1 or 2).

Value

This function is called for its side effects and does not return a value. It produces plots visualizing the predicted counts, credible intervals, and posterior summaries.

References

Zhao J, Qing K, Xu J (2025). *BKP: An R Package for Beta Kernel Process Modeling*. arXiv. https://doi.org/10.48550/arxiv.2508.10447.

See Also

`fit_NBKP` for fitting NBKP models; `predict.NBKP` for generating predictions from fitted NBKP models.

Examples

```
# -------------------------- 1D Example --------------------------
set.seed(123)

# Define true mean function
true_mu_fun <- function(x) {
  exp(sin(x) + 0.5)
}

n <- 30
Xbounds <- matrix(c(-2, 2), nrow=1)
X <- tgp::lhs(n = n, rect = Xbounds)
true_mu <- true_mu_fun(X)
y <- rnbinom(n, size = 1, mu = true_mu)

# Fit NBKP model
model1 <- fit_NBKP(X, y, Xbounds=Xbounds)
```

```
# Plot results
plot(model1)

# -------------------------- 2D Example --------------------------
set.seed(123)

# Define 2D latent function and mean transformation
true_mu_fun <- function(X) {
  if(is.null(nrow(X))) X <- matrix(X, nrow=1)
  x1 <- 4*X[,1] - 2
  x2 <- 4*X[,2] - 2
  f <- sin(2*pi*x1) * cos(2*pi*x2)
  return(exp(f))
}

n <- 100
Xbounds <- matrix(c(0, 0, 1, 1), nrow = 2)
X <- tgp::lhs(n = n, rect = Xbounds)
true_mu <- true_mu_fun(X)
y <- rnbinom(n, size = 0.5, mu = true_mu)

# Fit NBKP model
model2 <- fit_NBKP(X, y, Xbounds=Xbounds)

# Plot results
plot(model2, n_grid = 50)
```

`predict.NBKP` *Predict Method for NBKP Objects*

Description

Generate predictions from a fitted NBKP model.

Usage

```
## S3 method for class 'NBKP'
predict(object, newdata = NULL, ci_level = 0.95, ...)
```

Arguments

`object`	Fitted NBKP model object.
`newdata`	Matrix of new input points. If NULL, uses training data.
`ci_level`	Credible interval level.
`...`	Additional arguments.

Value

A list containing predictions, credible intervals, and variance.

`print.NBKP` *Print Methods for NBKP Objects*

Description

Provides formatted console output for fitted NBKP model objects and their predictions. The following specialized methods are supported:

- `print.NBKP` – display fitted NBKP model objects.
- `print.predict_NBKP` – display posterior predictive results.

Usage

```
## S3 method for class 'NBKP'
print(x, ...)

## S3 method for class 'predict_NBKP'
print(x, ...)
```

Arguments

`x`	An object of class `"NBKP"` or `"predict_NBKP"`.
`...`	Additional arguments passed to the generic `print` method (currently unused; included for S3 consistency).

Value

Invisibly returns the input object. Called for the side effect of printing human-readable summaries to the console.

References

Zhao J, Qing K, Xu J (2025). *BKP: An R Package for Beta Kernel Process Modeling*. arXiv. https://doi.org/10.48550/arxiv.2508.10447.

See Also

`fit_NBKP` for model fitting; `predict.NBKP` for posterior prediction.

Examples

```
# -------------------------- 1D Example --------------------------
set.seed(123)

# Define true mean function
true_mu_fun <- function(x) {
  exp(sin(x) + 0.5)
}

n <- 30
Xbounds <- matrix(c(-2, 2), nrow=1)
X <- tgp::lhs(n = n, rect = Xbounds)
true_mu <- true_mu_fun(X)
y <- rnbinom(n, size = 1, mu = true_mu)

# Fit NBKP model
model1 <- fit_NBKP(X, y, Xbounds=Xbounds)
print(model1) # fitted object

pred1 <- predict(model1)
print(pred1) # predictions

# -------------------------- 2D Example --------------------------
set.seed(123)

# Define 2D latent function and mean transformation
true_mu_fun <- function(X) {
  if(is.null(nrow(X))) X <- matrix(X, nrow=1)
  x1 <- 4*X[,1] - 2
  x2 <- 4*X[,2] - 2
  f <- sin(2*pi*x1) * cos(2*pi*x2)
  return(exp(f))
}

n <- 100
Xbounds <- matrix(c(0, 0, 1, 1), nrow = 2)
X <- tgp::lhs(n = n, rect = Xbounds)
true_mu <- true_mu_fun(X)
y <- rnbinom(n, size = 0.5, mu = true_mu)

# Fit NBKP model
model2 <- fit_NBKP(X, y, Xbounds=Xbounds)
print(model2)

pred2 <- predict(model2)
print(pred2)
```

---

quantile.NBKP *Posterior Quantiles from a Fitted NBKP Model*

Description

Compute posterior quantiles from a fitted NBKP model. This returns posterior quantiles of the latent mean count from the Gamma posterior distribution.

Usage

```
## S3 method for class 'NBKP'
quantile(x, probs = c(0.025, 0.5, 0.975), ...)
```

Arguments

x	An object of class NBKP, typically the result of a call to fit_NBKP.
probs	Numeric vector of probabilities specifying which posterior quantiles to return. Defaults to c(0.025, 0.5, 0.975).
...	Additional arguments (currently unused).

Details

For a NBKP model, posterior quantiles are computed from the Gamma Kernel Process posterior distribution.

Value

A numeric vector (if length(probs) = 1) or numeric matrix (if length(probs) > 1) of posterior quantiles. Rows correspond to observations, and columns correspond to the requested probabilities.

References

Zhao J, Qing K, Xu J (2025). *BKP: An R Package for Beta Kernel Process Modeling*. arXiv. https://doi.org/10.48550/arxiv.2508.10447.

See Also

fit_NBKP for model fitting.

Examples

```
set.seed(123)

# Define true mean function
true_mu_fun <- function(x) {
  exp(sin(x) + 0.5)
}

n <- 30
Xbounds <- matrix(c(-2, 2), nrow = 1)
X <- tgp::lhs(n = n, rect = Xbounds)
true_mu <- true_mu_fun(X)
y <- rnbinom(n, size = 1, mu = true_mu)
```

```
# Fit NBKP model
model <- fit_NBKP(X, y, Xbounds = Xbounds)

# Extract posterior quantiles
quantile(model)
```

`simulate.NBKP` *Simulate from a Fitted NBKP Model*

Description

Generates random draws from the posterior predictive distribution of a fitted `NBKP` model at specified input locations.

For NBKP models, posterior samples are generated from Gamma distributions characterizing latent mean count values for negative-binomial observations.

Usage

```
## S3 method for class 'NBKP'
simulate(object, nsim = 1, seed = NULL, Xnew = NULL, ...)
```

Arguments

`object`	An object of class `"NBKP"`, typically returned by `fit_NBKP`.
`nsim`	Number of posterior samples to generate (default `1`).
`seed`	Optional integer seed for reproducibility.
`Xnew`	A numeric matrix or vector of new input locations at which simulations are generated.
`...`	Additional arguments (currently unused).

Value

A list with the following components:

`samples` A numeric matrix of size `nrow(Xnew)` × `nsim`, where each column corresponds to one posterior draw of latent mean counts.

`mean` A numeric vector of posterior mean count values at each `Xnew`.

`X` The training input matrix used to fit the NBKP model.

`Xnew` The new input locations at which simulations are generated.

References

Zhao J, Qing K, Xu J (2025). *BKP: An R Package for Beta Kernel Process Modeling*. arXiv. https://doi.org/10.48550/arxiv.2508.10447.

See Also

fit_NBKP for model fitting; predict.NBKP for posterior prediction.

Examples

```
set.seed(123)

# Define true mean function
true_mu_fun <- function(x) {
  exp(sin(x) + 0.5)
}

n <- 30
Xbounds <- matrix(c(-2, 2), nrow = 1)
X <- tgp::lhs(n = n, rect = Xbounds)
true_mu <- true_mu_fun(X)
y <- rnbinom(n, size = 1, mu = true_mu)

# Fit NBKP model
model <- fit_NBKP(X, y, Xbounds = Xbounds)

# Simulate 5 posterior draws of latent mean counts
Xnew <- matrix(seq(-2, 2, length.out = 5), ncol = 1)
simulate(model, Xnew = Xnew, nsim = 5)
```

---

summary.NBKP *Summary of a Fitted NBKP Model*

Description

Provides a structured summary of a fitted Negative Binomial Kernel Process (NBKP) model. This function reports the model configuration, prior specification, kernel settings, and key posterior quantities, giving users a concise overview of the fitting results.

Usage

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

Arguments

object	An object of class "NBKP" from fit_NBKP.
...	Additional arguments passed to the generic summary method (currently not used).

Value

A list containing key summaries of the fitted model:

`n_obs` Number of training observations.

`input_dim` Input dimensionality (number of columns in X).

`kernel` Kernel type used in the model.

`theta_opt` Estimated kernel hyperparameters.

`loss` Loss function type used in the model.

`loss_min` Minimum value of the loss function achieved.

`prior` Prior type used (e.g., "noninformative", "fixed", "adaptive").

`r0` Prior precision parameter.

`mu0` Prior mean parameter (for fixed prior).

`phi` Negative-binomial dispersion parameter.

`post_mean` Posterior mean count estimates at training points.

`post_var` Posterior variance estimates of latent mean counts.

References

Zhao J, Qing K, Xu J (2025). *BKP: An R Package for Beta Kernel Process Modeling*. arXiv. https://doi.org/10.48550/arxiv.2508.10447.

See Also

`fit_NBKP` for model fitting.

Examples

```
set.seed(123)

# Define true mean function
true_mu_fun <- function(x) {
  exp(sin(x) + 0.5)
}

n <- 30
Xbounds <- matrix(c(-2, 2), nrow = 1)
X <- tgp::lhs(n = n, rect = Xbounds)
true_mu <- true_mu_fun(X)
y <- rnbinom(n, size = 1, mu = true_mu)

# Fit NBKP model
model1 <- fit_NBKP(X, y, Xbounds = Xbounds)
summary(model1)

# 2D Example
set.seed(123)
true_mu_fun <- function(X) {
```

```
  if(is.null(nrow(X))) X <- matrix(X, nrow=1)
  x1 <- 4*X[,1] - 2
  x2 <- 4*X[,2] - 2
  f <- sin(2*pi*x1) * cos(2*pi*x2)
  return(exp(f))
}
n <- 100
Xbounds <- matrix(c(0, 0, 1, 1), nrow = 2)
X <- tgp::lhs(n = n, rect = Xbounds)
true_mu <- true_mu_fun(X)
y <- rnbinom(n, size = 0.5, mu = true_mu)
model2 <- fit_NBKP(X, y, Xbounds = Xbounds)
summary(model2)
```

Index