Package: refundBayes (via r-universe)

October 8, 2026

Encoding UTF-8

Type Package

Title Bayesian Regression with Functional Data

Version 0.6.0

Depends R (>= 3.6.0)

Imports ggplot2 (>= 2.0.0), mgcv, refund, rstan (>= 2.29.0), splines2, stats

Description Bayesian regression with functional data, including regression with scalar, survival, or functional outcomes. The package allows regression with scalar and functional predictors. Methods are described in Jiang et al. (2025) ``Tutorial on Bayesian Functional Regression Using Stan'' <doi:10.1002/sim.70265>.

LazyData true

NeedsCompilation no

BugReports https://github.com/ZirenJiang/refundBayes/issues

License GPL-2

URL https://zirenjiang.github.io/refundBayes/, https://github.com/ZirenJiang/refundBayes

RoxygenNote 7.3.3

Suggests knitr, rmarkdown

VignetteBuilder knitr

Config/pak/sysreqs cmake make

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

Date/Publication 2026-05-11 19:39:05 UTC

RemoteUrl https://github.com/zirenjiang/refundbayes

RemoteRef HEAD

RemoteSha 6b7bc72f6dffc9d1b34cd7ed4389b002c6f13f96

Contents

fcox_bayes 2
fofr_bayes 4
fosr_bayes 7
fpca_bayes 10
plot.refundBayes 12
sofr_bayes 13
summary.refundBayes 16

Index **17**

`fcox_bayes` *Bayesian Functional Cox Regression*

Description

Fit the Bayesian Functional Cox Regression model using Stan.

Usage

```
fcox_bayes(
  formula,
  data,
  cens,
  joint_FPCA = NULL,
  intercept = FALSE,
  runStan = TRUE,
  niter = 3000,
  nwarmup = 1000,
  nchain = 3,
  ncores = 1
)
```

Arguments

`formula`	Functional regression formula, with the same syntax as that in the R mgcv package.
`data`	A data frame containing data of all scalar and functional variables used in the model.
`cens`	A vector indicating censoring status (1 = event observed, 0 = censored). Must be the same length as the number of observations.
`joint_FPCA`	A True/False vector of the same length of the number of functional predictors, indicating whether to jointly model the functional predictor via FPCA together with the survival model. When the entry is `TRUE`, the corresponding observed functional predictor is replaced by an FPCA representation, and the FPC scores are sampled jointly with the regression coefficients, following Section 4 of Jiang

	et al. (2025). Default to NULL (no joint FPCA, equivalent to rep(FALSE, n_func)).
intercept	True/False variable for whether include an intercept term in the linear predictor. Default to FALSE.
runStan	True/False variable for whether to run the Stan program. If False, the function only generates the Stan code and data.
niter	Total number of Bayesian iterations. Default to 3000.
nwarmup	Number of warmup (burnin) iterations for posterior sampling. Default to 1000.
nchain	Number of chains for posterior sampling. Default to 3.
ncores	Number of cores to use when executing the chains in parallel. Default to 1.

Details

The Bayesian Functional Cox model extends the scalar-on-function regression framework to survival outcomes with right censoring. The model is specified using similar syntax as in the R mgcv package.

Value

A list containing:

stanfit	The Stan fit object.
spline_basis	Basis functions used to reconstruct the functional coefficients from posterior samples.
stancode	A character string containing the code to fit the Stan model.
standata	A list containing the data to fit the Stan model.
int	A vector containing posterior samples of the intercept term (NULL for Cox models by default).
scalar_coef	A matrix containing posterior samples of scalar coefficients, where each row is one sample and each column is one variable.
func_coef	A list containing posterior samples of functional coefficients. Each element is a matrix, where each row is one sample and each column is one location of the functional domain.
baseline_hazard	Posterior samples of baseline hazard parameters.
family	Family type: "Cox".

Author(s)

Erjia Cui <ecui@umn.edu>, Ziren Jiang <jian0746@umn.edu>

References

Jiang, Z., Crainiceanu, C., and Cui, E. (2025). Tutorial on Bayesian Functional Regression Using Stan. *Statistics in Medicine*, 44(20-22), e70265.

Examples

```
## Not run:
# Simulate survival data with a functional predictor
set.seed(1)
n  <- 150  # number of subjects
L  <- 50   # number of functional domain points
Lindex <- seq(0, 1, length.out = L)       # functional domain grid
X_func <- matrix(rnorm(n * L), nrow = n)  # functional predictor (n x L)
age    <- rnorm(n)                         # scalar predictor

# True functional effect and linear predictor
beta_true <- cos(pi * Lindex)
lp <- X_func %*% beta_true / L + age

# Generate survival times from an exponential baseline hazard
time  <- rexp(n, rate = exp(lp))
cens_time <- runif(n, min = 0.5, max = 3)
obs_time  <- pmin(time, cens_time)
cens_ind  <- as.integer(time <= cens_time)  # 1 = event, 0 = censored

dat <- data.frame(obs_time = obs_time, age = age)
dat$X_func <- X_func
dat$Lindex <- matrix(rep(Lindex, n), nrow = n, byrow = TRUE)

# Fit the Bayesian Functional Cox model
fit_cox <- fcox_bayes(
  formula = obs_time ~ age + s(Lindex, by = X_func, bs = "cr", k = 10),
  data    = dat,
  cens    = cens_ind,
  niter   = 2000,
  nwarmup = 1000,
  nchain  = 3
)

# Summarise scalar coefficients and plot functional coefficient
summary(fit_cox)
plot(fit_cox)

## End(Not run)
```

`fofr_bayes` *Bayesian Function-on-Function Regression*

Description

Fit the Bayesian Function-on-Function Regression (FoFR) model using Stan.

Usage

```
fofr_bayes(
  formula,
  data,
  joint_FPCA = NULL,
  runStan = TRUE,
  niter = 3000,
  nwarmup = 1000,
  nchain = 3,
  ncores = 1,
  spline_type = "bs",
  spline_df = 10
)
```

Arguments

`formula`	Functional regression formula, with the same syntax as that in the R mgcv package. The response must be a matrix (functional response) and at least one `s(..., by = ...)` term must be present for the functional predictor(s).
`data`	A data frame containing data of all scalar and functional variables used in the model.
`joint_FPCA`	A True/False vector of the same length of the number of functional predictors, indicating whether jointly modeling FPCA for the functional predictors. Default to NULL.
`runStan`	True/False variable for whether to run the Stan program. If False, the function only generates the Stan code and data.
`niter`	Total number of Bayesian iterations.
`nwarmup`	Number of warmup (burnin) iterations for posterior sampling.
`nchain`	Number of chains for posterior sampling. Default to 3.
`ncores`	Number of cores to use when executing the chains in parallel. Default to 1.
`spline_type`	Type of spline basis for modelling the residual process and the response-domain component of the bivariate coefficient. Default to "bs".
`spline_df`	Degrees of freedom for the spline basis for modelling the response-domain component. Default to 10.

Details

The Bayesian FoFR model extends the function-on-scalar regression (FoSR) framework by allowing functional predictors in addition to (optional) scalar predictors. The bivariate coefficient function $\beta(s,t)$ is represented using a tensor product of the predictor-domain spline basis and the response-domain spline basis. The model is specified using the same syntax as in the R mgcv package.

Value

A list containing:

`stanfit` The Stan fit object.

`spline_basis` Basis functions used to reconstruct the functional coefficients from posterior samples.

`stancode` A character string containing the code to fit the Stan model.

`standata` A list containing the data to fit the Stan model.

`scalar_func_coef`
A 3-d array (n_samples x P x M) containing posterior samples of scalar predictor coefficient functions. Each slice `[,p,]` is the coefficient function for the p-th scalar predictor. NULL if no scalar predictors.

`bivar_func_coef`
A list of 3-d arrays. Each element corresponds to one functional predictor and is an array of dimension (n_samples x S_grid x M), representing posterior samples of the bivariate coefficient function $\beta(s, t)$.

`func_coef` A 3-d array for the scalar predictor coefficient functions, stored for compatibility with the `plot.refundBayes` method. Same as `scalar_func_coef`.

`family` Model family: "fofr".

Author(s)

Erjia Cui <ecui@umn.edu>, Ziren Jiang <jian0746@umn.edu>

References

Jiang, Z., Crainiceanu, C., and Cui, E. (2025). Tutorial on Bayesian Functional Regression Using Stan. *Statistics in Medicine*, 44(20-22), e70265.

Examples

```
## Not run:
# Simulate data for a Function-on-Function Regression model
set.seed(1)
n  <- 100  # number of subjects
L  <- 30   # number of functional predictor domain points
M  <- 30   # number of functional response domain points
sindex <- seq(0, 1, length.out = L)  # predictor domain grid
tindex <- seq(0, 1, length.out = M)  # response domain grid

# Functional predictor
X_func <- matrix(rnorm(n * L), nrow = n)

# Scalar predictor
age <- rnorm(n)

# True bivariate coefficient beta(s, t)
beta_true <- outer(sin(2 * pi * sindex), cos(2 * pi * tindex))

# True scalar coefficient function
alpha_true <- sin(pi * tindex)
```

```
# Generate functional response: Y(t) = age * alpha(t) + integral X(s) beta(s,t) ds + error
Y_mat <- outer(age, alpha_true) + X_func %*% beta_true / L +
          matrix(rnorm(n * M, sd = 0.3), nrow = n)

dat <- data.frame(age = age)
dat$Y_mat  <- Y_mat
dat$X_func <- X_func
dat$sindex <- matrix(rep(sindex, n), nrow = n, byrow = TRUE)

# Fit the Bayesian FoFR model
fit_fofr <- fofr_bayes(
  formula    = Y_mat ~ age + s(sindex, by = X_func, bs = "cr", k = 10),
  data       = dat,
  spline_type = "bs",
  spline_df  = 10,
  niter      = 2000,
  nwarmup    = 1000,
  nchain     = 3
)

# Examine the bivariate coefficient beta(s, t) (posterior mean)
beta_est <- apply(fit_fofr$bivar_func_coef[[1]], c(2, 3), mean)
image(sindex, tindex, beta_est,
      xlab = "s (predictor domain)", ylab = "t (response domain)",
      main = expression(hat(beta)(s, t)))

# Fit a FoFR model with functional predictors only (no scalar predictors)
fit_fofr2 <- fofr_bayes(
  formula    = Y_mat ~ s(sindex, by = X_func, bs = "cr", k = 10),
  data       = dat,
  niter      = 2000,
  nwarmup    = 1000,
  nchain     = 3
)

## End(Not run)
```

`fosr_bayes` *Bayesian Function-on-Scalar Regression*

Description

Fit the Bayesian Function-on-Scalar Regression (FOSR) model using Stan.

Usage

```
fosr_bayes(
  formula,
  data,
```

```
  joint_FPCA = NULL,
  runStan = TRUE,
  niter = 3000,
  nwarmup = 1000,
  nchain = 3,
  ncores = 1,
  spline_type = "bs",
  spline_df = 10
)
```

Arguments

formula	Functional regression formula, with the same syntax as that in the R mgcv package.
data	A data frame containing data of all scalar and functional variables used in the model.
joint_FPCA	A True/False vector of the same length of the number of functional predictors, indicating whether jointly modeling FPCA for the functional predictors. Default to NULL.
runStan	True/False variable for whether to run the Stan program. If False, the function only generates the Stan code and data.
niter	Total number of Bayesian iterations.
nwarmup	Number of warmup (burnin) iterations for posterior sampling.
nchain	Number of chains for posterior sampling. Default to 3.
ncores	Number of cores to use when executing the chains in parallel. Default to 1.
spline_type	Type of spline basis for modelling the residual process.
spline_df	Degrees of freedom for the spline basis for modelling the residual process.

Details

The Bayesian FOSR model is implemented following the tutorial by Jiang et al., 2025. The model is specified using the same syntax as in the R mgcv package.

Value

A list containing:

stanfit	The Stan fit object.
spline_basis	Basis functions used to reconstruct the functional coefficients from posterior samples.
stancode	A character string containing the code to fit the Stan model.
standate	A list containing the data to fit the Stan model.
int	A vector containing posterior samples of the intercept term.
scalar_coef	A matrix containing posterior samples of scalar coefficients, where each row is one sample and each column is one variable.

func_coef	A list containing posterior samples of functional coefficients. Each element is a matrix, where each row is one sample and each column is one location of the functional domain.
family	Distribution of the outcome variable.

Author(s)

Erjia Cui <ecui@umn.edu>, Ziren Jiang <jian0746@umn.edu>

References

Jiang, Z., Crainiceanu, C., and Cui, E. (2025). Tutorial on Bayesian Functional Regression Using Stan. *Statistics in Medicine*, 44(20-22), e70265.

Examples

```
## Not run:
# Simulate data for a Function-on-Scalar Regression model
set.seed(1)
n  <- 100   # number of subjects
M  <- 50    # number of functional response observation points
tindex <- seq(0, 1, length.out = M)  # response functional domain grid

# Scalar predictors
age <- rnorm(n)
sex <- rbinom(n, 1, 0.5)

# True coefficient functions
beta_age <- sin(2 * pi * tindex)
beta_sex <- cos(2 * pi * tindex)

# Generate functional response (n x M matrix)
epsilon  <- matrix(rnorm(n * M, sd = 0.3), nrow = n)
Y_mat    <- outer(age, beta_age) + outer(sex, beta_sex) + epsilon

dat <- data.frame(age = age, sex = sex)
dat$Y_mat <- Y_mat

# Fit the Bayesian FoSR model
fit_fosr <- fosr_bayes(
  formula    = Y_mat ~ age + sex,
  data       = dat,
  spline_type = "bs",
  spline_df  = 10,
  niter      = 2000,
  nwarmup    = 1000,
  nchain     = 3
)

# Plot estimated coefficient functions
plot(fit_fosr)
```

```
## End(Not run)
```

`fpca_bayes` *Bayesian Functional Principal Component Analysis*

Description

Fit the Bayesian Functional Principal Component Analysis (FPCA) model using Stan.

Usage

```
fpca_bayes(
  formula,
  data,
  npc = NULL,
  runStan = TRUE,
  niter = 3000,
  nwarmup = 1000,
  nchain = 3,
  ncores = 1,
  spline_type = "bs",
  spline_df = 10
)
```

Arguments

`formula`	Functional formula of the form `Y_mat ~ 1`, where `Y_mat` is the functional response variable stored as a matrix column of `data`.
`data`	A data frame containing the functional response variable used in the model.
`npc`	Number of functional principal components. If NULL, it is selected automatically by the initial `refund::fpca.sc()` fit based on the percentage of variance explained. Default to NULL.
`runStan`	True/False variable for whether to run the Stan program. If False, the function only generates the Stan code and data.
`niter`	Total number of Bayesian iterations.
`nwarmup`	Number of warmup (burnin) iterations for posterior sampling.
`nchain`	Number of chains for posterior sampling. Default to 3.
`ncores`	Number of cores to use when executing the chains in parallel. Default to 1.
`spline_type`	Type of spline basis for modelling the population mean function. Default to "bs".
`spline_df`	Degrees of freedom for the spline basis for modelling the population mean function. Default to 10.

Details

The Bayesian FPCA model is implemented following the tutorial by Jiang et al., 2025. The model decomposes a dense functional response $Y_i(t)$ into a smooth population mean function $\mu(t)$ and a sum of functional principal components,

$$Y_i(t) = \mu(t) + \sum_{k=1}^{K} \xi_{ik}\phi_k(t) + \epsilon_i(t),$$

where $\phi_k(t)$ are orthonormal eigenfunctions obtained from an initial frequentist FPCA fit (used as a fixed basis), and the mean function $\mu(t)$, the FPC scores ξ_{ik}, the eigenvalues λ_k, and the residual variance are estimated via posterior sampling. The population mean function is modelled with a penalized spline using the same syntax as in the R mgcv package.

Value

A list containing:

`stanfit`	The Stan fit object.
`stancode`	A character string containing the code to fit the Stan model.
`standata`	A list containing the data to fit the Stan model.
`mu`	A matrix of posterior samples of the population mean function, where each row is one sample and each column is one location of the functional domain.
`efunctions`	A matrix of the (fixed) eigenfunctions from the initial FPCA used as the FPC basis. Each column is one eigenfunction.
`scores`	A 3-d array of posterior samples of FPC scores with dimensions (n_samples x n_subjects x npc).
`evalues`	A matrix of posterior samples of FPC eigenvalue standard deviations, where each row is one sample and each column is one principal component.
`sigma`	A vector of posterior samples of the residual standard deviation.
`family`	Family type: "fpca".

Author(s)

Erjia Cui <ecui@umn.edu>, Ziren Jiang <jian0746@umn.edu>

References

Jiang, Z., Crainiceanu, C., and Cui, E. (2025). Tutorial on Bayesian Functional Regression Using Stan. *Statistics in Medicine*, 44(20-22), e70265.

Examples

```
## Not run:
# Simulate functional data with two underlying principal components
set.seed(1)
n  <- 100   # number of subjects
M  <- 50    # number of functional observation points
```

```
tindex <- seq(0, 1, length.out = M)

# True mean function and eigenfunctions
mu_true <- sin(pi * tindex)
phi1    <- sqrt(2) * sin(2 * pi * tindex)
phi2    <- sqrt(2) * cos(2 * pi * tindex)

# Simulate scores and noisy observations
xi1 <- rnorm(n, 0, sqrt(2))
xi2 <- rnorm(n, 0, sqrt(0.5))
Y_mat <- matrix(rep(mu_true, n), nrow = n, byrow = TRUE) +
         outer(xi1, phi1) + outer(xi2, phi2) +
         matrix(rnorm(n * M, sd = 0.3), nrow = n)

dat <- data.frame(inx = 1:n)
dat$Y_mat <- Y_mat

# Fit the Bayesian FPCA model
fit_fpca <- fpca_bayes(
  formula     = Y_mat ~ 1,
  data        = dat,
  spline_type = "bs",
  spline_df   = 10,
  niter       = 2000,
  nwarmup     = 1000,
  nchain      = 3
)

# Posterior mean of the mean function
mu_est <- apply(fit_fpca$mu, 2, mean)
plot(tindex, mu_est, type = "l", ylab = expression(hat(mu)(t)))

# Posterior means of the FPC scores and eigenvalues
scores_est  <- apply(fit_fpca$scores, c(2, 3), mean)
evalues_est <- apply(fit_fpca$evalues, 2, mean)

## End(Not run)
```

`plot.refundBayes` *Plot the estimated functional coefficients with the corresponding credible interval(s).*

Description

Produces coefficient plots tailored to the model family:

- `sofr_bayes()`, `fcox_bayes()`: one curve plot per functional predictor coefficient $\beta(s)$, with pointwise and/or CMA credible bands.
- `fosr_bayes()`: one curve plot per scalar predictor coefficient function $\alpha_p(t)$.

- `fofr_bayes()`: curve plots for scalar predictor coefficient functions $\alpha_p(t)$ (if any), followed by heatmap plots for each bivariate coefficient surface $\beta_q(s,t)$ (posterior mean) from the functional predictors.
- `fpca_bayes()`: posterior mean function $\mu(t)$ with a pointwise credible band; a combined plot of the (fixed) FPC eigenfunctions $\phi_j(t)$; a point-and-error-bar plot of the posterior of the eigenvalue SDs λ_j; and a histogram of the residual-SD posterior σ_ϵ.

Usage

```
## S3 method for class 'refundBayes'
plot(x = NULL, ..., prob = 0.95, include = "both")
```

Arguments

`x`	A fitted object returned by `sofr_bayes()`, `fosr_bayes()`, `fofr_bayes()`, `fcox_bayes()`, or `fpca_bayes()`.
`...`	Other parameters
`prob`	Coverage probability for the credible interval(s). Defaults to 0.95.
`include`	Type of interval to include. `"pointwise"` produces pointwise credible intervals; `"CMA"` produces the CMA credible band; `"both"` produces both. Defaults to `"both"`. Only used for `sofr_bayes()` / `fcox_bayes()` curve plots.

Value

A named list of `ggplot` objects. For FoFR, scalar-predictor curves are named `scalar_<p>` and bivariate-predictor heatmaps are named `bivar_<q>`. For FPCA, the plots are named `mu`, `efunctions`, `evalues`, and `sigma`.

`sofr_bayes` *Bayesian Scalar-on-Function Regression*

Description

Fit the Bayesian Scalar-on-Function Regression (SoFR) model using Stan.

Usage

```
sofr_bayes(
  formula,
  data,
  family = gaussian(),
  joint_FPCA = NULL,
  intercept = TRUE,
  runStan = TRUE,
  niter = 3000,
  nwarmup = 1000,
```

```
  nchain = 3,
  ncores = 1
)
```

Arguments

formula	Functional regression formula, with the same syntax as that in the R mgcv package.
data	A data frame containing data of all scalar and functional variables used in the model.
family	Distribution of the outcome variable. Currently support "gaussian" and "binomial".
joint_FPCA	A True/False vector of the same length of the number of functional predictors, indicating whether to jointly model the functional predictor via FPCA together with the regression model. When the entry is TRUE, the corresponding observed functional predictor is replaced by an FPCA representation, and the FPC scores are sampled jointly with the regression coefficients, following Section 4 of Jiang et al. (2025). Default to NULL (no joint FPCA, equivalent to rep(FALSE, n_func)).
intercept	True/False variable for whether include an intercept term in the linear predictor. Default to TRUE.
runStan	True/False variable for whether to run the Stan program. If False, the function only generates the Stan code and data.
niter	Total number of Bayesian iterations.
nwarmup	Number of warmup (burnin) iterations for posterior sampling.
nchain	Number of chains for posterior sampling. Default to 3.
ncores	Number of cores to use when executing the chains in parallel. Default to 1.

Details

The Bayesian SoFR model is implemented following the tutorial by Jiang et al., 2025. The model is specified using the same syntax as in the R mgcv package.

Value

A list containing:

stanfit	The Stan fit object.
spline_basis	Basis functions used to reconstruct the functional coefficients from posterior samples.
stancode	A character string containing the code to fit the Stan model.
standate	A list containing the data to fit the Stan model.
int	A vector containing posterior samples of the intercept term.
scalar_coef	A matrix containing posterior samples of scalar coefficients, where each row is one sample and each column is one variable.

func_coef	A list containing posterior samples of functional coefficients. Each element is a matrix, where each row is one sample and each column is one location of the functional domain.
family	Distribution of the outcome variable.

Author(s)

Erjia Cui <ecui@umn.edu>, Ziren Jiang <jian0746@umn.edu>

References

Jiang, Z., Crainiceanu, C., and Cui, E. (2025). Tutorial on Bayesian Functional Regression Using Stan. *Statistics in Medicine*, 44(20-22), e70265.

Examples

```
## Not run:
# Simulate data for a Gaussian SoFR model
set.seed(1)
n  <- 100  # number of subjects
L  <- 50   # number of functional domain points
Lindex <- seq(0, 1, length.out = L)       # functional domain grid
X_func <- matrix(rnorm(n * L), nrow = n)  # functional predictor (n x L)
age    <- rnorm(n)                         # scalar predictor
beta_true <- sin(pi * Lindex)         # true functional coefficient
eta <- X_func %*% beta_true / L
Y <- eta + 0.5 * age + rnorm(n, sd = 0.5)

dat <- data.frame(Y = Y, age = age)
dat$X_func  <- X_func
dat$Lindex  <- matrix(rep(Lindex, n), nrow = n, byrow = TRUE)

# Fit Gaussian SoFR
fit_sofr <- sofr_bayes(
  formula = Y ~ age + s(Lindex, by = X_func, bs = "cr", k = 10),
  data    = dat,
  family  = "gaussian",
  niter   = 2000,
  nwarmup = 1000,
  nchain  = 3
)

# Summarise and plot estimated functional coefficient
summary(fit_sofr)
plot(fit_sofr)

# Fit binomial SoFR
prob <- plogis(X_func %*% beta_true / L)
Y_bin <- rbinom(n, 1, prob)
dat$Y_bin <- Y_bin
fit_bin <- sofr_bayes(
  formula = Y_bin ~ s(Lindex, by = X_func, bs = "cr", k = 10),
```

```
  data    = dat,
  family  = "binomial",
  niter   = 2000,
  nwarmup = 1000,
  nchain  = 3
)

## End(Not run)
```

`summary.refundBayes` *Generate the summary table for the Bayesian model*

Description

Generate the summary table for the Bayesian model

Usage

```
## S3 method for class 'refundBayes'
summary(object = NULL, ..., prob = 0.95)
```

Arguments

`object`	A fitted object returned by `sofr_bayes()`.
`...`	Other parameters
`prob`	Coverage probability for the reported confidence intervals. Defaults to 0.95.

Value

A list of two objects, the first is the summary table for the estimated scalar coefficients, the second is the plots for the estimated functional coefficients.

Index

fcox_bayes, 2
fofr_bayes, 4
fosr_bayes, 7
fpca_bayes, 10

plot.refundBayes, 12

sofr_bayes, 13
summary.refundBayes, 16