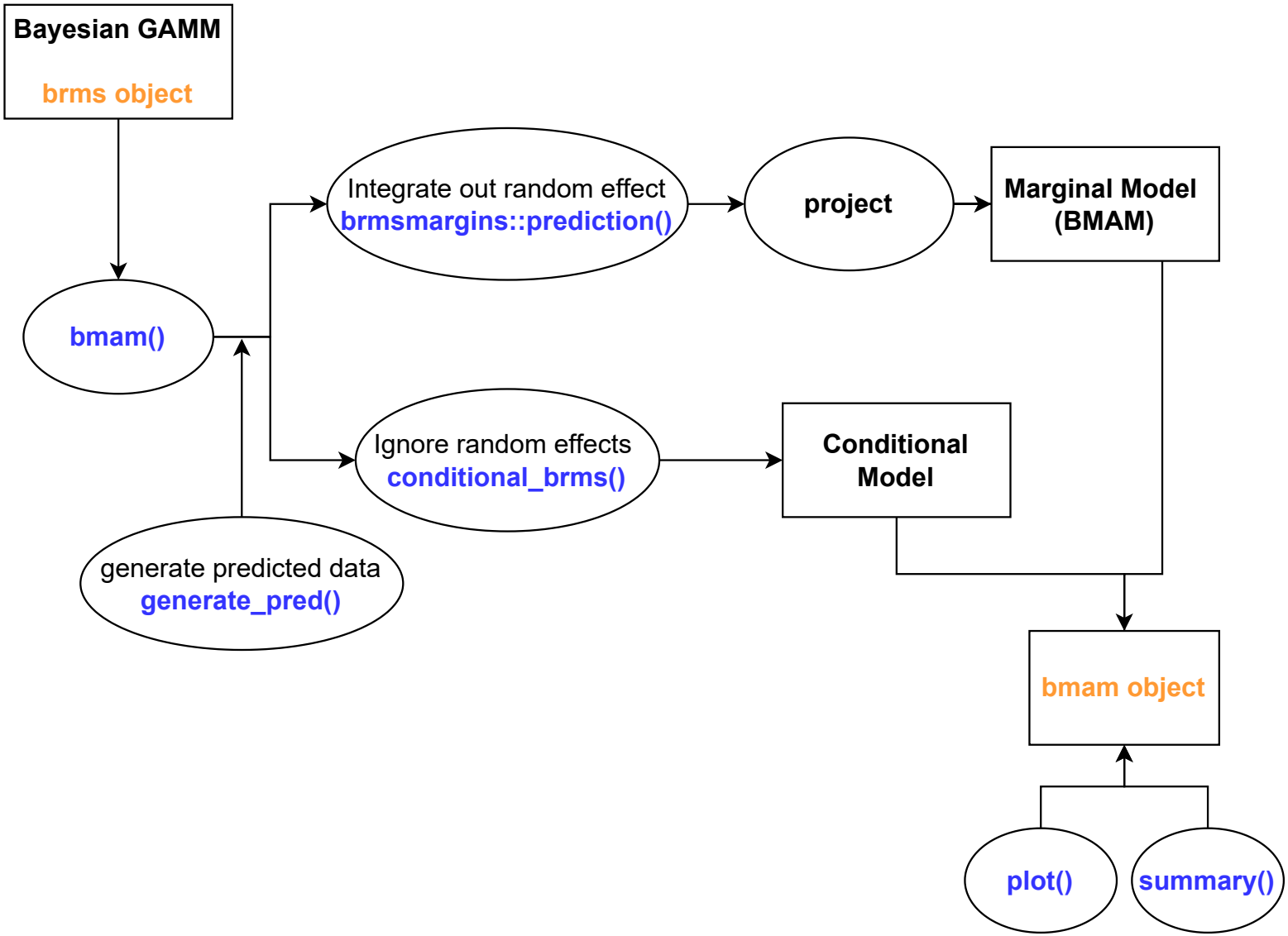
R functions: BMAM

Tianyi Pan

2022-05-30

Diagram


Bayesian GAMM
brms object
bmam()
Integrate out random effect
brmsmargins::prediction()
project
Marginal Model
(BMAM)
Ignore random effects
conditional_brms()
Conditional
Model
generate predicted data
generate_pred()
bmam object
plot()
summary()

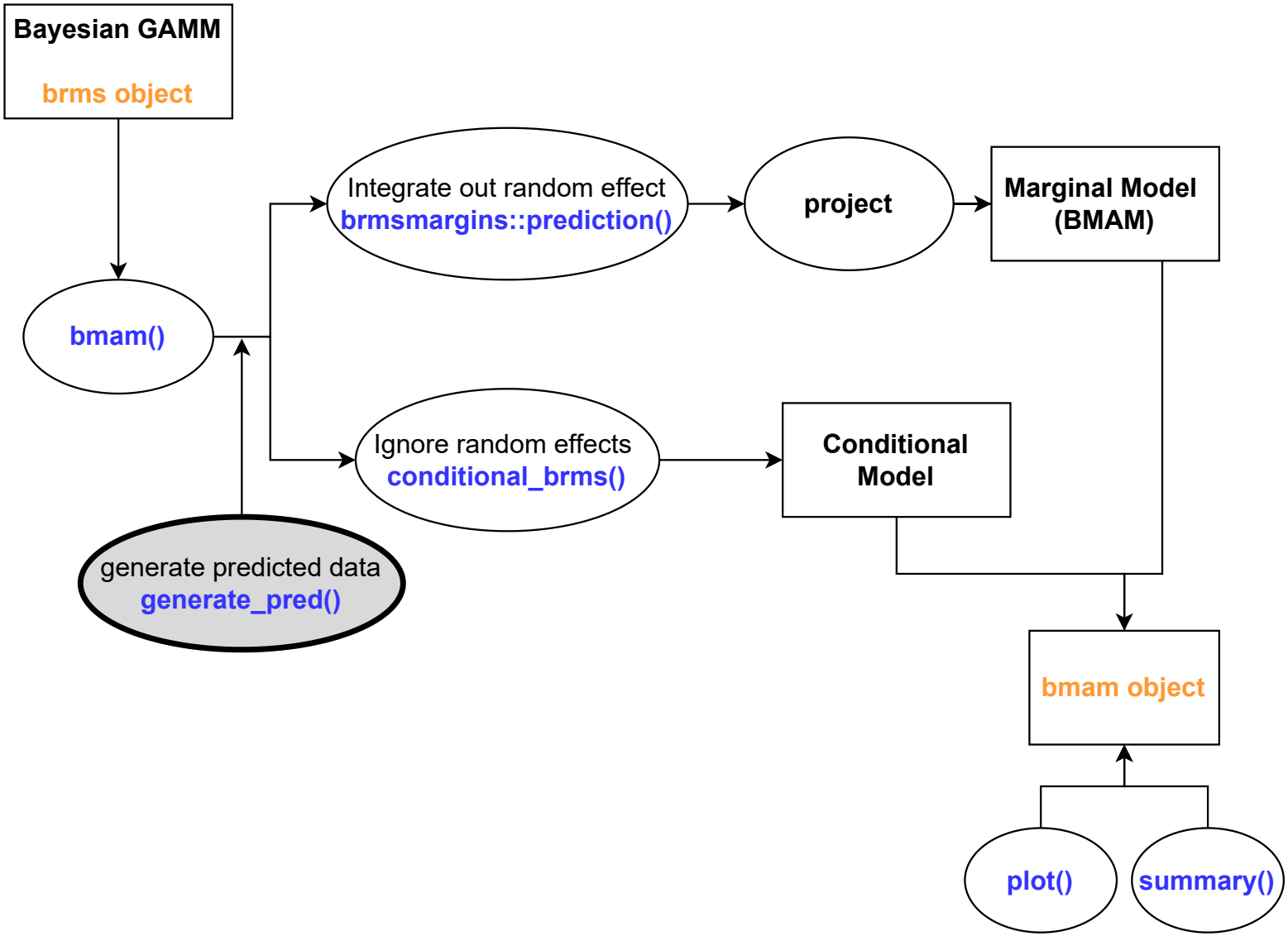
bmam()

- Fit Bayesian Marginal Additive Model (BMAM)
- Modifying `brmsmargins::marginalcoef` function

bmam()

```
bmam <- function(object, preddat, length = 100,
                 summarize = TRUE, posterior = TRUE,
                 backtrans = c("response", "linear", "identity",
                               "invlogit", "exp", "square", "inverse"),
                 centered = FALSE, k = 100, ...) {
  ...
```

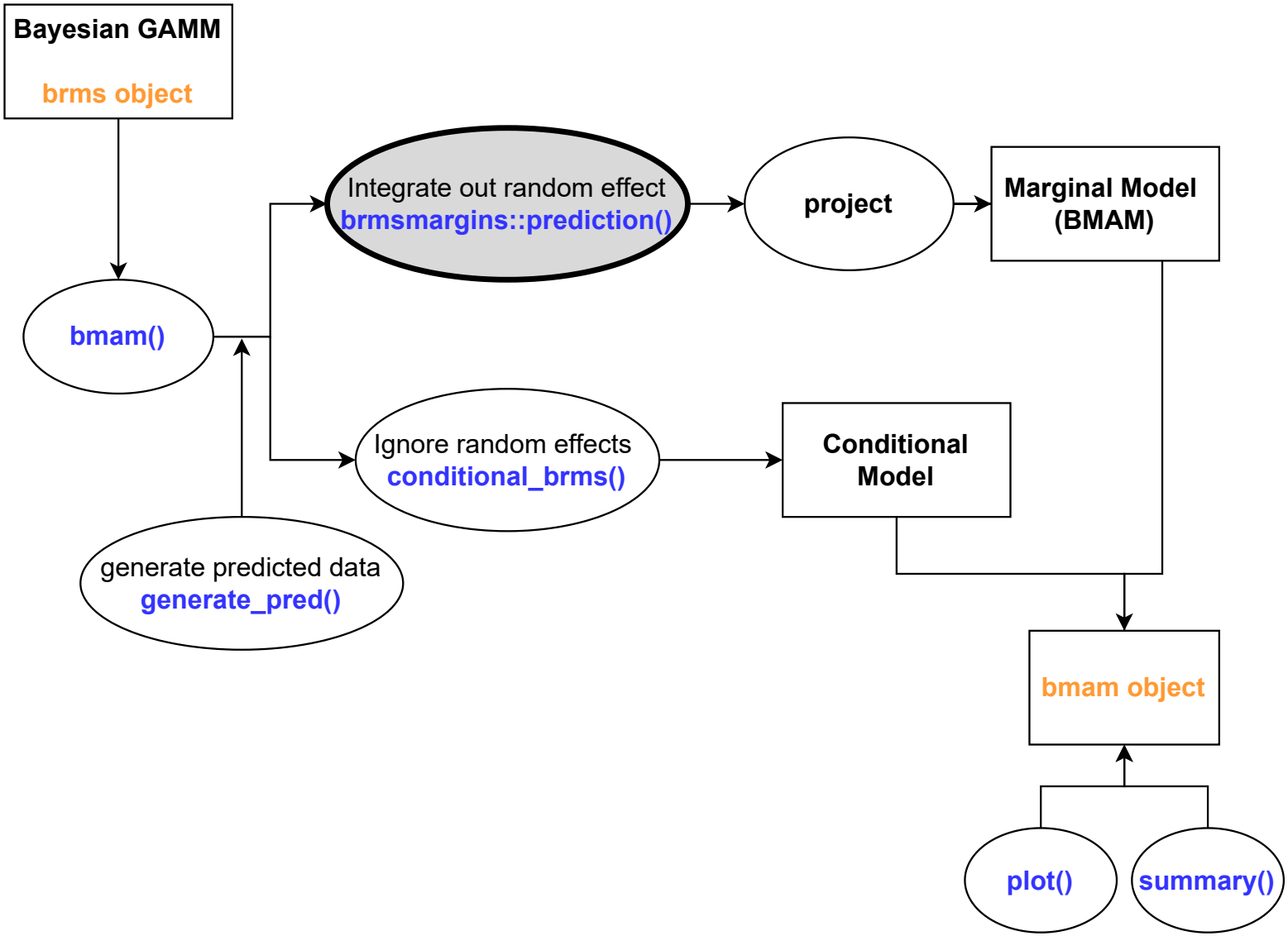
- **object** A fitted brms model object
- **preddat** A data frame containing covariates at which predictions are required
- **length** The number of observations in the generated predicted data
- **summarize** Whether or not to calculate summaries of the posterior predictions
- **posterior** Whether or not to save and return the posterior samples
- **backtrans** The type of back transformation to be applied. default: response meaning to use the response scale
- **centered** Whether or not return centered smooths
- **k** The number of random draws to use for integrating out the random effects (Monte Carlo)


Bayesian GAMM
brms object
bmam()
Integrate out random effect
brmsmargins::prediction()
project
Marginal Model
(BMAM)
Ignore random effects
conditional_brms()
Conditional
Model
generate predicted data
generate_pred()
bmam object
plot()
summary()

generate_pred()

```
  ## bmam()
  if (missingArg(preddat)) preddat <- generate_pred(object, length)
```

```
generate_pred <- function(object, length = 100){
  mf <- model.frame(object) # data in object
  ## smooth term
  smterm <- brmsterms(object$formula)$dpars$mu$sm # smooth term
  stopifnot(!is.null(smterm)) # check
  smvariable <- lapply(smterm[[2]][-1], function(term.) term.[[2]]) # varia
  sm_pred <- lapply(smvariable, function(var){
    x <- mf[[var]]
    x <- x[which(x!=0)] # remove 0
    x.max <- max(x)
    x.min <- min(x)
    seq(x.min, x.max, length = length)
  })
  names(sm_pred) <- smvariable
  sm_pred <- do.call("cbind.data.frame", sm_pred)
  ## fix effect
  ...
```


Bayesian GAMM
brms object
bmam()
Integrate out random effect
brmsmargins::prediction()
project
Marginal Model
(BMAM)
Ignore random effects
conditional_brms()
Conditional
Model
generate predicted data
generate_pred()
bmam object
plot()
summary()

bmam() Marginal Model

Integrate out random effect (brmsmargins::prediction)

Equation 4 in Hedeker's paper

$$\hat{\pi}_{ij}^{pa} = \int_{\theta} g^{-1}\left(\boldsymbol{x}_{ij}'\hat{\boldsymbol{\beta}}^{ss} + \boldsymbol{z}_{ij}'\hat{\boldsymbol{T}}\boldsymbol{\theta}_i\right) dF(\boldsymbol{\theta}_i)$$

For the h^{th} draw from posterior distribution, $h = 1, 2, \cdots, H$ (H is the number of sampling, for example, 4×1000)

- Calculate $\boldsymbol{x}\hat{\boldsymbol{\beta}}^{ss,h}$, and $\boldsymbol{z}\hat{\boldsymbol{T}}^{h}$
- Generate K numbers from $N(0, I)$, denoted as $\theta^k, k = 1, \cdots, K$.
- Calculate $g^{-1}\left(\boldsymbol{x}_{ij}\hat{\boldsymbol{\beta}}^{ss,h} + \boldsymbol{z}_{ij}\hat{\boldsymbol{T}}^{h}\boldsymbol{\theta}_i^k\right), k = 1, \cdots, K$
- Average over k. $\hat{\pi}_{ij}^{pa,h} = \sum_{k=1}^{K} g^{-1}\left(\boldsymbol{x}_{ij}\hat{\boldsymbol{\beta}}^{ss,h} + \boldsymbol{z}_{ij}\hat{\boldsymbol{T}}^{h}\boldsymbol{\theta}_i^k\right)/K$

bmam() Marginal Model

Integrate out random effect (brmsmargins::prediction)

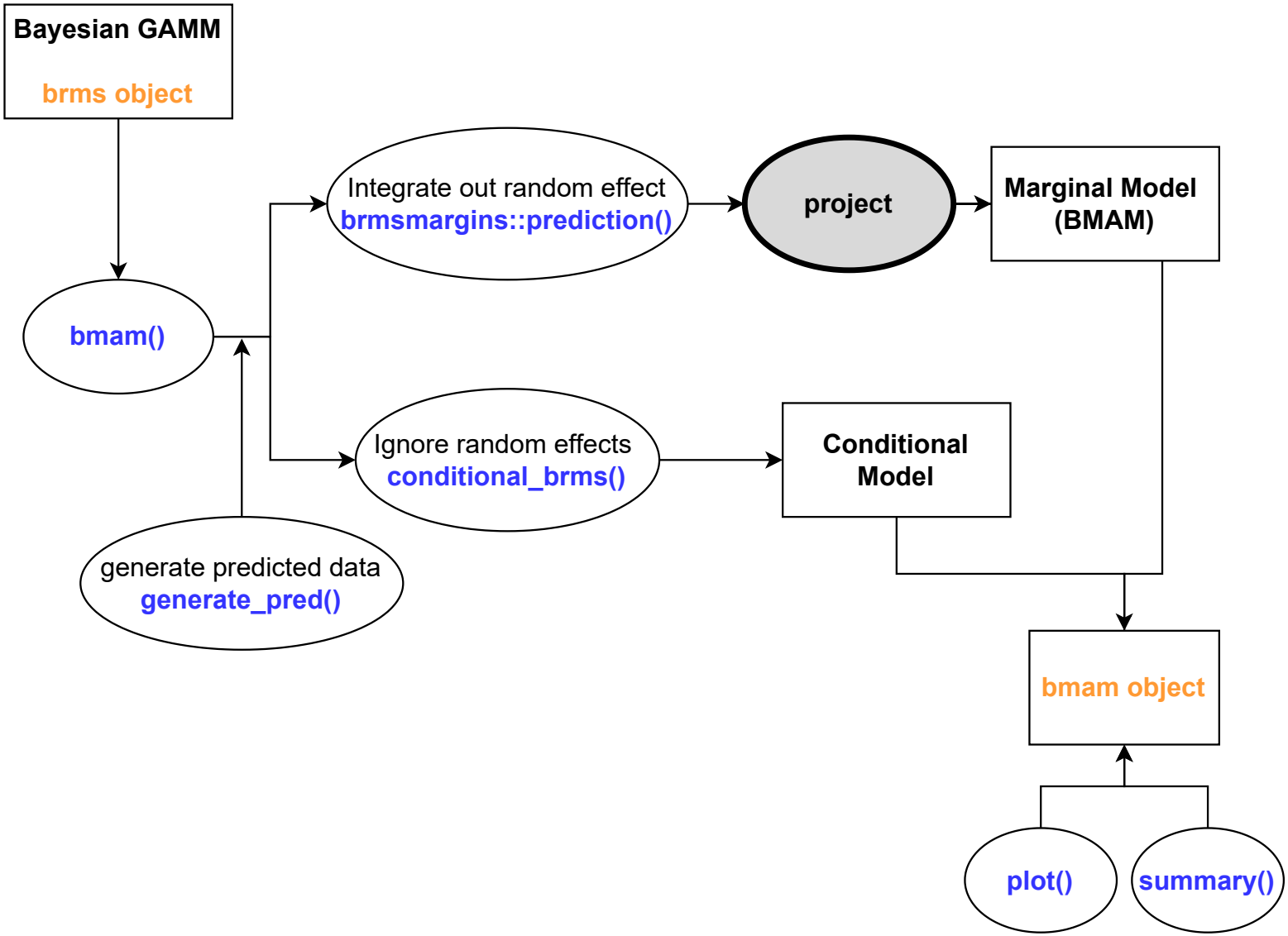
Equation 4 in Hedeker's paper

$$\hat{\pi}_{ij}^{pa} = \int_{\theta} g^{-1}\left(\boldsymbol{x}_{ij}'\hat{\beta}^{ss} + \boldsymbol{z}_{ij}'\hat{\boldsymbol{T}}\boldsymbol{\theta}_i\right) dF(\boldsymbol{\theta}_i)$$

For the h^{th} draw from posterior distribution, $h = 1, 2, \cdots, H$ (H is the number of sampling, for example, 4×1000)

- Calculate $\boldsymbol{x}\hat{\beta}^{ss,h}$, and $\boldsymbol{z}\hat{\boldsymbol{T}}^{h}$
- Generate K numbers from $N(0, I)$, denoted as $\theta^k, k = 1, \cdots, K$.
- Calculate $g^{-1}\left(\boldsymbol{x}_{ij}\hat{\beta}^{ss,h} + \boldsymbol{z}_{ij}\hat{\boldsymbol{T}}^{h}\boldsymbol{\theta}_i^k\right), k = 1, \cdots, K$
- Average over k. $\hat{\pi}_{ij}^{pa,h} = \sum_{k=1}^{K} g^{-1}\left(\boldsymbol{x}_{ij}\hat{\beta}^{ss,h} + \boldsymbol{z}_{ij}\hat{\boldsymbol{T}}^{h}\boldsymbol{\theta}_i^k\right)/K$

```
mu <- brmsmargins::prediction(
  object, data = model.frame(object),
  summarize = FALSE, posterior = TRUE, effects = "integrateoutRE",
  backtrans = backtrans, k = k, raw = TRUE)
```


Bayesian GAMM
brms object
bmam()
Integrate out random effect
brmsmargins::prediction()
project
Marginal Model
(BMAM)
Ignore random effects
conditional_brms()
Conditional
Model
generate predicted data
generate_pred()
bmam object
plot()
summary()

bmam() Marginal Model

Projection

$$f^{\mathrm{M}}(\mathbf{X}) = \boldsymbol{B}^{\mathrm{M}}\boldsymbol{\alpha}^{\mathrm{M}}$$

$$\widehat{\boldsymbol{\alpha}}^{\mathrm{M}} = \operatorname{argmin} \left\| \widehat{\boldsymbol{\lambda}}^{\mathrm{M}} - \boldsymbol{B}^{\mathrm{M}}\boldsymbol{\alpha}^{\mathrm{M}} \right\| = \left(\boldsymbol{B}^{\mathrm{M}\top}\boldsymbol{B}^{\mathrm{M}} \right)^{-1} \boldsymbol{B}^{\mathrm{M}\top}\widehat{\boldsymbol{\lambda}}^{\mathrm{M}}.$$

1. Design Matrix $\boldsymbol{B}^{\mathrm{M}}$

```
standata <- make_standata(formula(object), data = model.frame(object))
## subtract the variables
if(smooth){
  data_names <- names(standata) # get the names of data
  ## Zs: basis function for smooth term (ncol = k-2)
  Zs_name <- data_names[grep(pattern = "Zs_\\d_\\d", data_names)]
  Zs <- do.call(cbind, standata[Zs_name])
  ## set names for Zs
  Zs_name_list <- mapply(function(i,j)paste(j,seq_len(i),sep="_alpha_"),
                         lapply(standata[Zs_name],ncol),
                         Zs_name, SIMPLIFY = FALSE)
  colnames(Zs) <- as.character(unlist(Zs_name_list))
```

bmam() Marginal Model

1. Design Matrix B^{M}

```
  ## Xs: basis function for smooth term, without penalty (ncol = 1)
  Xs_name <- data_names[grep(pattern = "Xs", data_names)]
  Xs <- do.call(cbind, standata[Xs_name])
  ## set names for Xs
  Xs_name_list <- mapply(function(i,j)paste(j,seq_len(i),"alpha",sep="_"),
                         lapply(standata[Xs_name],ncol),
                         Xs_name)
  colnames(Xs) <- as.character(unlist(Xs_name_list))

  ## X: linear term, for example intercept + x1 + x2 + x1:x2
  X <- standata$X

  ## design matrix
  B <- cbind(X, Xs, Zs)
}
```

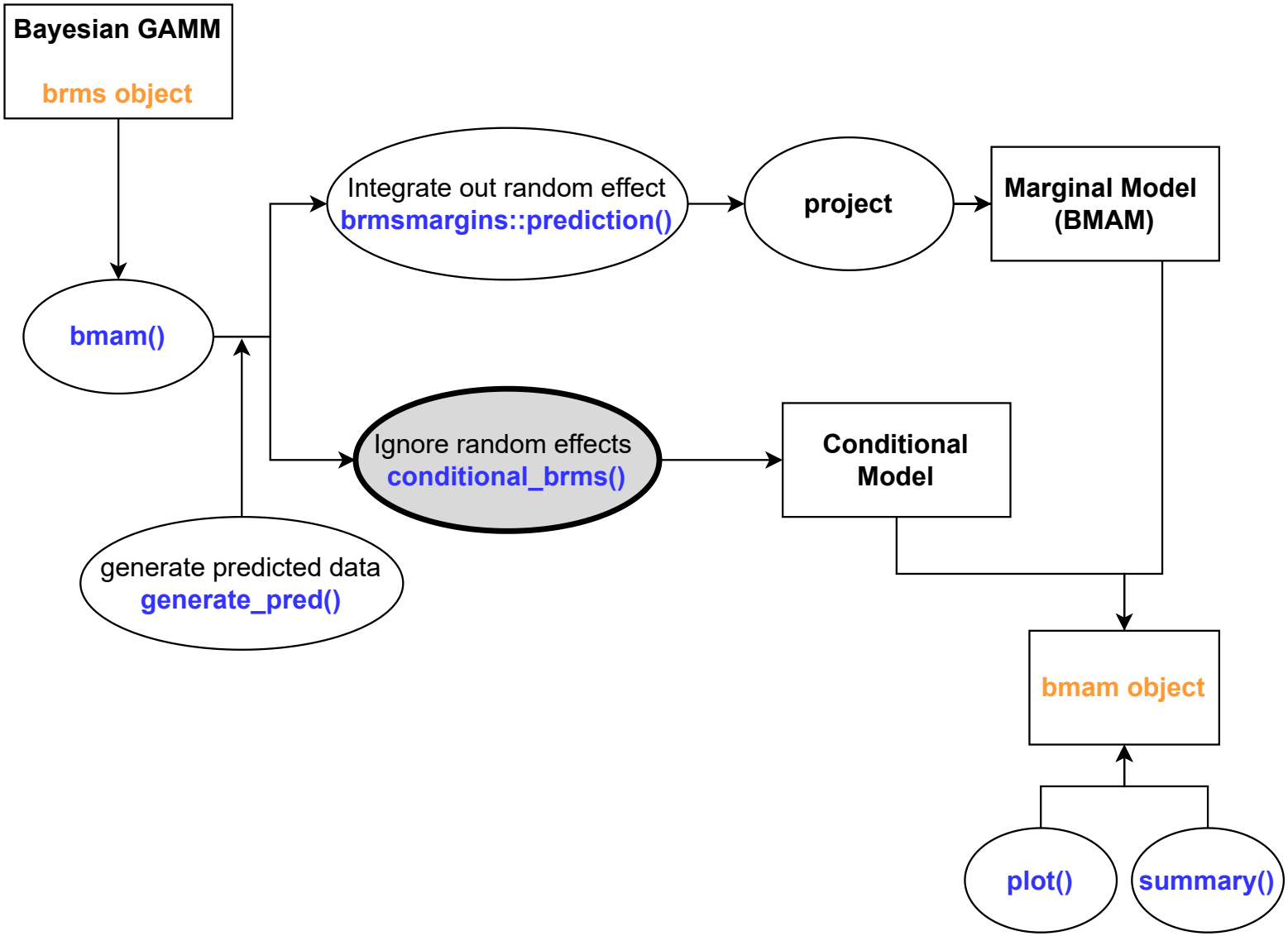
bmam() Marginal Model

2. $\widehat{\boldsymbol{\lambda}}^{\mathrm{M}}$

```
y <- links$fun(t(mu$Posterior))
```

3. projection

```
beta <- lmcpp(B, y)  # Cpp file from brmsmargins
                     # beta <- solve( t(B) %*% B ) %*% t(B) %*% y
prep <- prepare_predictions(object, newdata = preddat,
                            check_response = FALSE, re_formula = NA)
## Xs: basis function for smooth term, without penalty (ncol = 1)
pred_Xs <- prep$dpars$mu$sm$fe$Xs
## Zs: basis function for smooth term (ncol = k-2)
pred_Zs <- sapply(prep$dpars$mu$sm$re, function(re.)re.$Zs)
pred_Zs <- do.call(cbind, pred_Zs)
## X: linear term
pred_X <- prep$dpars$mu$fe$X
pred_B <- cbind(pred_X, pred_Xs, pred_Zs)
if(centered) pred_B <- sweep(pred_B,2,colMeans(pred_B),'-')
Predicted <- pred_B %*% beta
```

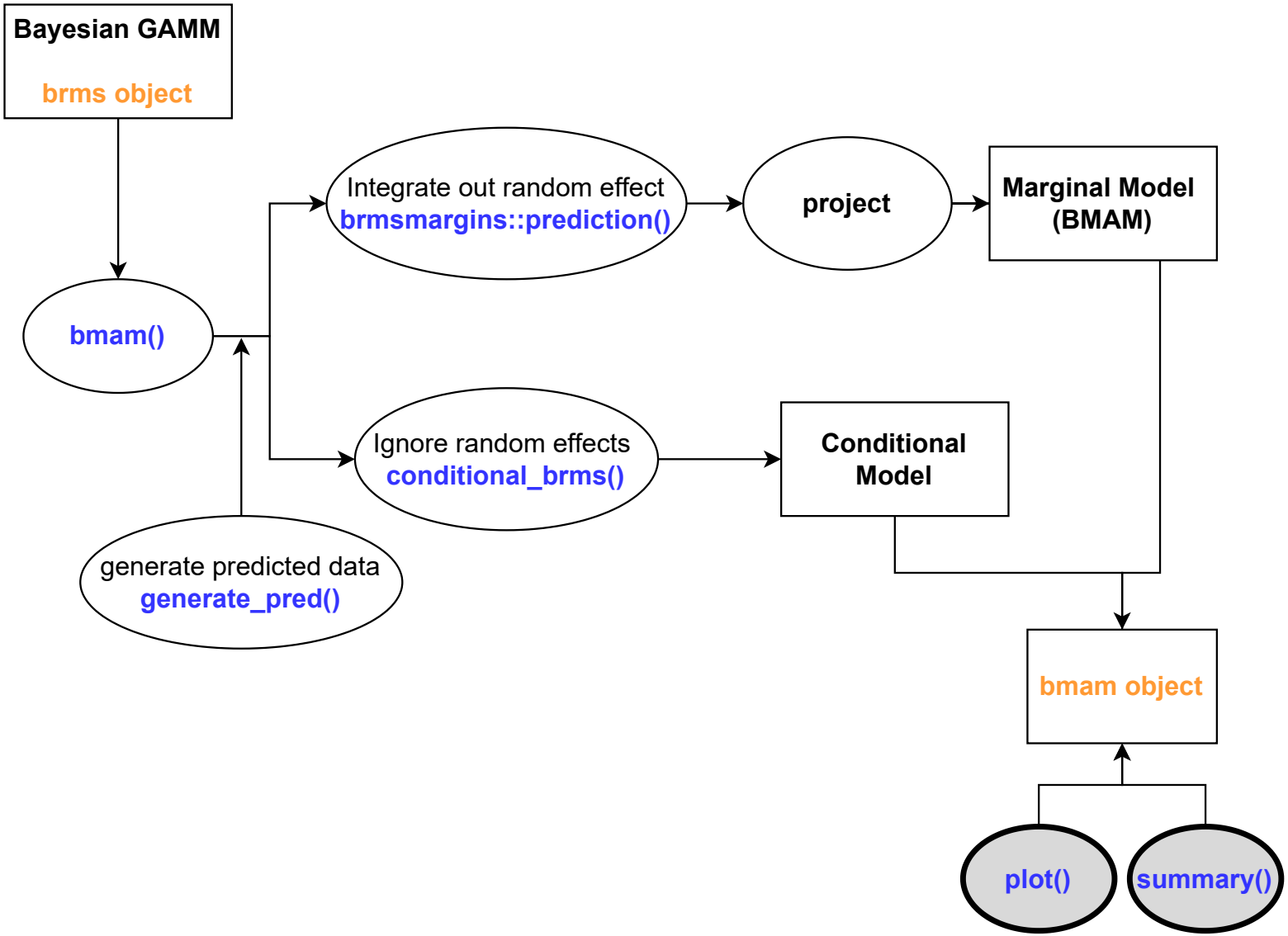

Bayesian GAMM
brms object
bmam()
Integrate out random effect
brmsmargins::prediction()
project
Marginal Model
(BMAM)
Ignore random effects
conditional_brms()
Conditional
Model
generate predicted data
generate_pred()
bmam object
plot()
summary()

bmam() Conditional Model

```
predict_conditional <- conditional_brms(object, preddat,
                                        centered = centered, ...)
```

```
conditional_brms <- function(object, data, centered = FALSE, ...){
  yhat <- fitted(
    object = object, newdata = data,
    re_formula = NA, scale = "linear",
    summary=FALSE)
  if(centered){
    ones <- matrix(rep(1,ncol(yhat)))
    H_matrix <- ones %*% solve(t(ones) %*% ones) %*% t(ones)
    M_matrix <- diag(1,ncol(yhat)) - H_matrix
    predicted <- M_matrix %*% t(yhat)
  }else{
    predicted <- t(yhat)
  }

  as.data.table(do.call(rbind, apply(predicted, 1, bsummary, ...)))
}
```


Bayesian GAMM
brms object
bmam()
Integrate out random effect
brmsmargins::prediction()
project
Marginal Model
(BMAM)
Ignore random effects
conditional_brms()
Conditional
Model
generate predicted data
generate_pred()
bmam object
plot()
summary()

plot()

```
plot.bmam <- function(object, compared.model,
                      conditional = TRUE, display = TRUE,
                      smooth.function){
  ...
}
```

- **object** Objects of Class 'bmam'
- **compared.model** Other model (fitted before calling the function) compared with BMAM. Supported models: 1. mam 2. gam 3. brms gam
- **display** Whether or not to display the plot. Default: TRUE
- **conditional** Whether or not to plot the conditional model. Default: TRUE
- **smooth.function** True values of the smooth functions.

summary()

```
print.bmam <- function(object,...){
  summary(object,...)
  invisible()
}
```

```
summary.bmam <- function(object, plot.smooth = FALSE, ...){
  ### 1. Marginal Model ###################
  ## smooth term; linear term
  ...
  ### 2. Conditional Model #############
  ## smooth term; linear term
  ...
}
```

- **object** Objects of Class 'bmam'
- **plot.smooth** Whether or not to plot the smooth function