`bgphy`, a package for Bayesian analysis of mixed Gaussian phylogenetic models.

Bayu Brahmantio (`bayubrahmantio@gmail.com`)

The package at a glance

This package performs Bayesian inference of a class of phylogenetic comparative methods (PCM) which assumes Gaussian processes, specifically Brownian Motion (BM) or Ornstein-Uhlenbeck process (OU), as a model of univariate traits evolution on a phylogenetic tree. Moreover, it performs inferences of mixed Gaussian phylogenetic models (MGPM). In this setting, a phylogenetic tree can be partitioned into multiple different "evolutionary regimes" which consist of different processes along with different parameters.

The package uses PCMBase (Mitov et al. 2020) as the engine for high-speed likelihood calculations. A user-friendly interface for customizing priors and regime configurations is also provided. The Bayesian inference is done using importance sampling algorithm to compute statistics of the posterior distribution, which utilizes `R`'s native `parallel` package to accelerate inference time.

Parameters notation

The BM and OU processes can be written by the following stochastic differential equations (SDEs):

- BM: $dx(t) = \sigma dW(t)$,
- OU: $dx(t) = -\alpha(x(t) - \theta)dt + \sigma dW(t)$,

where $x(t)$ is the trait at time t , $W(t)$ is a Wiener process at time t . The parameter α can be interpreted as the strength of random fluctuation, while α is the strength of selection (or convergence rate) into an optimum value θ . Notice that, BM is a specific case of an OU process when $\alpha = 0$. The ancestral state is also considered as a parameter, which is denoted by X_0.

Given measurements on the tips $\mathbf{x}_{\text{tips}}$ and a model $\mathcal{M}$ with a certain regimes configurations and parameters Θ, the goal is to calculate the posterior distribution

$$P(\Theta|\mathbf{x}_{\text{tips}}) \propto P(\mathbf{x}_{\text{tips}}|\Theta)P(\Theta),$$

and compare different configurations, $\mathcal{M}_1(\Theta_1)$ vs $\mathcal{M}_2(\Theta_2)$ vs $\cdots$ vs $\mathcal{M}_m(\Theta_m)$.

Installation

```
devtools::install_github("bayubeta/bgphy", build_vignettes = TRUE)
```

Getting started

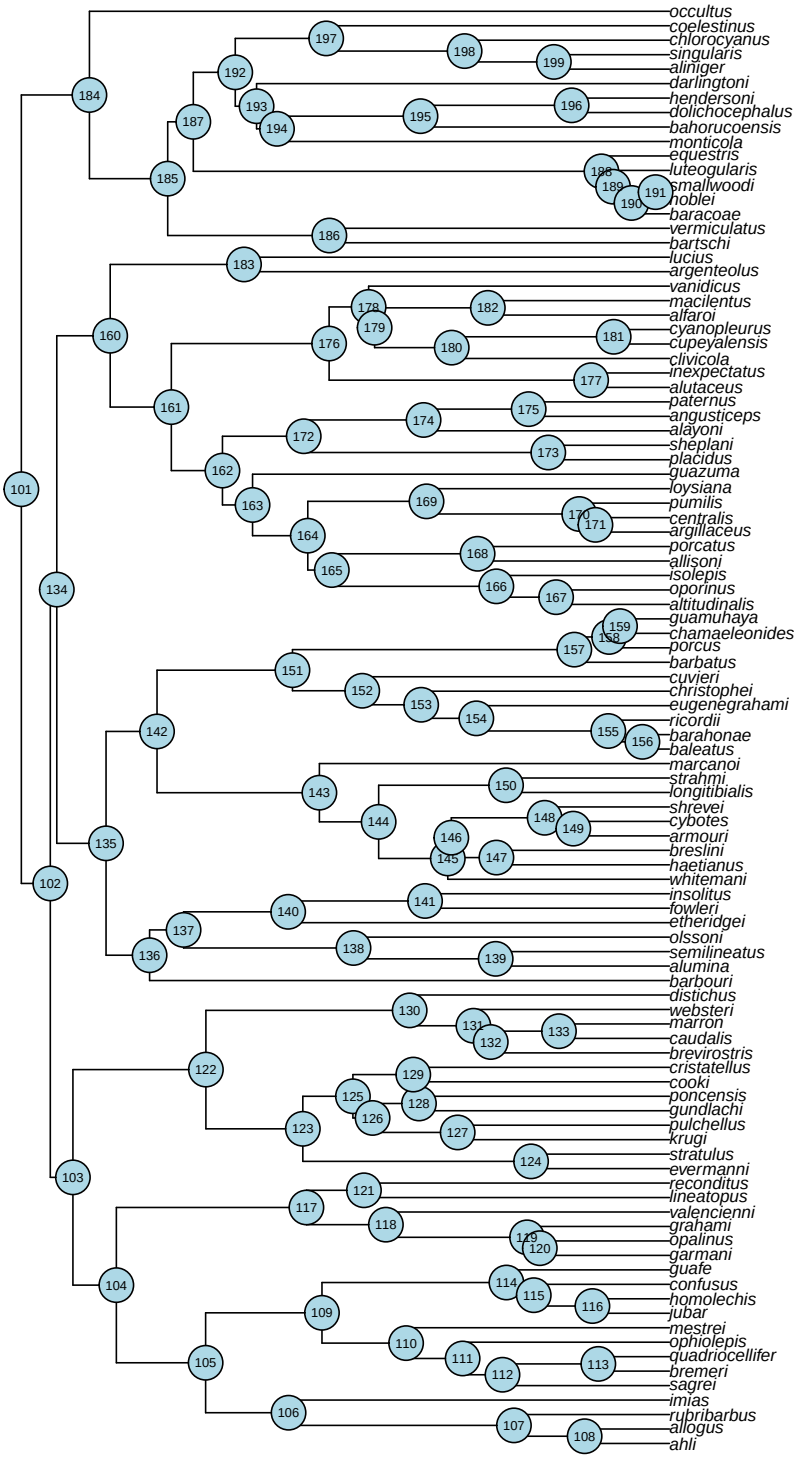
Loading the package:

```
library(bgphy)
library(ape)
```

Constructing a model using `setModel()`

Given a phylogenetic tree of class `phylo` and we can construct a model by using `setModel()` function by specifying the model types (`"BM"` or `"OU"`) and their position in the tree. For example, we use the phylogenetic tree of Caribbean anoles (Mahler et al. 2013; retrieved from Bastide and Didier 2023) . The tree is available within the package as `lizardTree`. Below is the tree plotted with its node numbers:

```
plot(lizardTree, no.margin = TRUE, cex = 0.7)
nodelabels(cex = 0.5, frame = "circle")
```


occultus
coelestinus
chlorocyanus
singularis
aliniger
darlingtoni
hendersoni
dolichocephalus
bahorucoensis
monticola
equestris
luteogularis
smallwoodi
noblei
baracoae
vermiculatus
bartschi
lucius
argenteolus
vanidicus
macilentus
alfaroi
cyanopleurus
cupeyalensis
clivicola
inexpectatus
alutaceus
paternus
angusticeps
alayoni
sheplani
placidus
guazuma
loysiana
pumilis
centralis
argillaceus
porcatus
allisoni
isolepis
oporinus
altitudinalis
guamuhaya
chamaeleonides
porcus
barbatus
cuvieri
christophei
eugenegrahami
ricordii
barahonae
baleatus
marcanoi
strahmi
longitibialis
shrevei
cybotes
armouri
breslini
haetianus
whitemani
insolitus
fowleri
etheridgei
olssoni
semilineatus
alumina
barbouri
distichus
websteri
marron
caudalis
brevirostris
cristatellus
cooki
poncensis
gundlachi
pulchellus
krugi
stratulus
evermanni
reconditus
lineatopus
valencienni
grahami
opalinus
garmani
guafe
confusus
homolechis
jubar
mestrei
ophiolepis
quadriocellifer
bremeri
sagrei
imias
rubribarbus
allogus
ahli

Global model

Assume that the traits are evolving under the same process throughout all parts of the tree (BM or OU). To construct such model we can run:

```
# Brownian Motion (BM)
BM <- setModel(tree = lizardTree, regime_names = "R1", modeltypes = "BM")

# Ornstein-Uhlenbeck (OU)
OU <- setModel(tree = lizardTree, regime_names = "R1", modeltypes = "OU")
```

This will return an object of class `bgphy_model` which contains the information of the model in `PCMBase`, the tree, and the default priors for the parameters of the model. The information of the models can be printed:

```
print(BM)
```

```
Regime    Model
------    -----
R1        BM(sigma)

Priors:
X0 ~ normal(mean = 0, sd = 10)
sigma ~ halft(nu = 1, sigma = 3)
```

```
print(OU)
```

```
Regime    Model
------    -----
R1        OU(alpha, theta, sigma)

Priors:
X0 ~ normal(mean = 0, sd = 10)
alpha ~ halfnormal(sigma = 6.931472)
theta ~ normal(mean = 0, sd = 10)
sigma ~ halft(nu = 1, sigma = 3)
```

Mixed model

Suppose we are interested in splitting the tree into two regimes:

- Ancestral: starts from the ancestral node (101).
- New: starts from node 135.

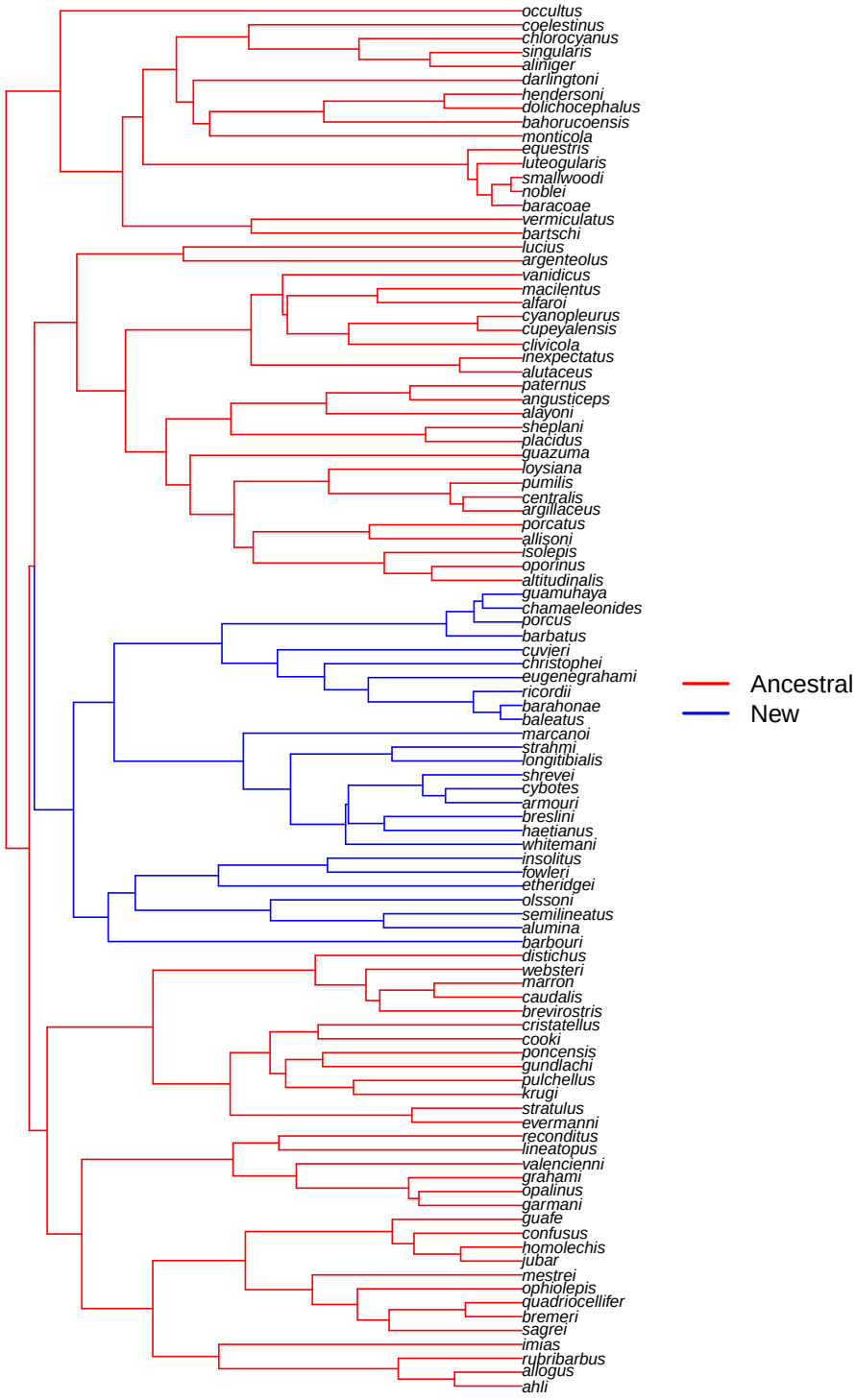
Furthermore, the ancestral regime is assumed to follow a BM process while the New regime follows an OU process. The model is constructed as:

```
BMOU <- setModel(tree = lizardTree, regime_names = c("Ancestral", "New"),
                 modeltypes = c("BM", "OU"),
                 startNodes = list(Ancestral = "101", New = "135"))
print(BMOU)
```

```
Regime      Model
------      -----
Ancestral   BM(sigma_1)
New         OU(alpha_2, theta_2, sigma_2)

Priors:
X0 ~ normal(mean = 0, sd = 10)
sigma_1 ~ halft(nu = 1, sigma = 3)
alpha_2 ~ halfnormal(sigma = 6.931472)
theta_2 ~ normal(mean = 0, sd = 10)
sigma_2 ~ halft(nu = 1, sigma = 3)
```

Using the function `plot_regimes()`, the regimes can be visualized on top of the tree:

```
plot_regimes(BMOU$tree, no.margin = TRUE, x.lim = c(0,1.6), cex = 0.7)
```


occultus
coelestinus
chlorocyanus
singularis
aliniger
darlingtoni
hendersoni
dolichocephalus
bahorucoensis
monticola
equestris
luteogularis
smallwoodi
noblei
baracoae
vermiculatus
bartschi
lucius
argenteolus
vanidicus
macilentus
alfaroi
cyanopleurus
cupeyalensis
clivicola
inexpectatus
alutaceus
paternus
angusticeps
alayoni
sheplani
placidus
guazuma
loysiana
pumilis
centralis
argillaceus
porcatus
allisoni
isolepis
oporinus
altitudinalis
guamuhaya
chamaeleonides
porcus
barbatus
cuvieri
christophei
eugenegrahami
ricordii
barahonae
baleatus
marcanoi
strahmi
longitibialis
shrevei
cybotes
armouri
breslini
haetianus
whitemani
insolitus
fowleri
etheridgei
olssoni
semilineatus
alumina
barbouri
distichus
websteri
marron
caudalis
brevirostris
cristatellus
cooki
poncensis
gundlachi
pulchellus
krugi
stratulus
evermanni
reconditus
lineatopus
valencienni
grahami
opalinus
garmani
guafe
confusus
homolechis
jubar
mestrei
ophiolepis
quadriocellifer
bremeri
sagrei
imias
rubribarbus
allogus
ahli
Ancestral
New

A more complicated configuration

Let's say that there are three regimes:

- R1: starts from the ancestral node (101), but includes the taxa *ahli*, *lucius,* and *darlingtoni*. Follows a BM process.
- R2: starts from node 104 and 160, follows an OU process.
- R3: starts from node 184, follows an OU process.

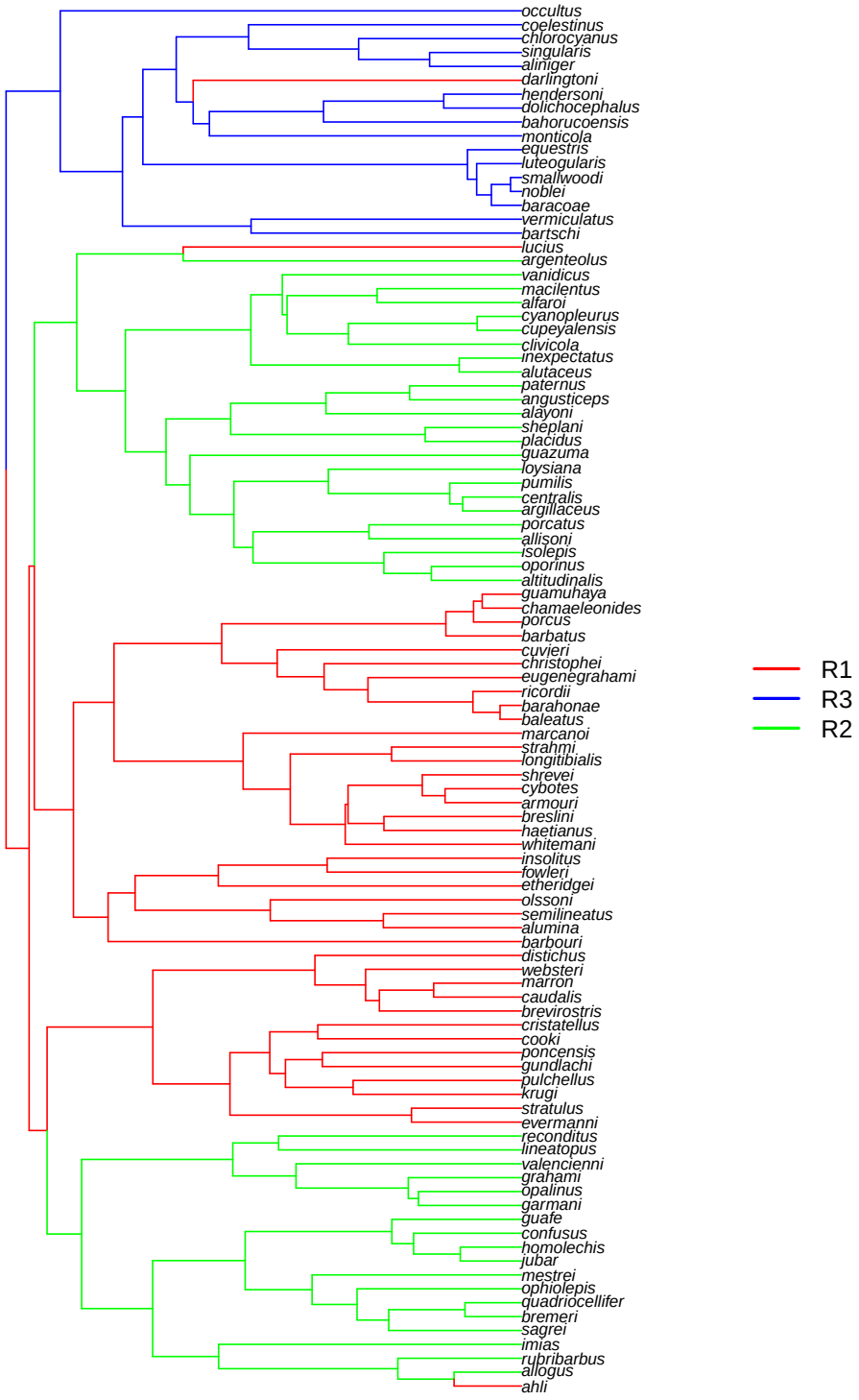
The model can be constructed as follows:

```
model3 <- setModel(tree = lizardTree, regime_names = c("R1", "R2", "R3"),
                   modeltypes = c("BM", "OU", "OU"),
                   startNodes = list(R1 = c("101", "ahli", "lucius", "darlingtoni"),
                                     R2 = c("104", "160"),
                                     R3 = c("184")))

print(model3)
```

```
Regime    Model
------    -----
R1        BM(sigma_1)
R2        OU(alpha_2, theta_2, sigma_2)
R3        OU(alpha_3, theta_3, sigma_3)

Priors:
X0 ~ normal(mean = 0, sd = 10)
sigma_1 ~ halft(nu = 1, sigma = 3)
alpha_2 ~ halfnormal(sigma = 6.931472)
theta_2 ~ normal(mean = 0, sd = 10)
sigma_2 ~ halft(nu = 1, sigma = 3)
alpha_3 ~ halfnormal(sigma = 6.931472)
theta_3 ~ normal(mean = 0, sd = 10)
sigma_3 ~ halft(nu = 1, sigma = 3)
```

And the regimes configuration can be shown by:

```
plot_regimes(model3$tree, no.margin = TRUE, x.lim = c(0,1.6), cex = 0.7)
```


occultus
coelestinus
chlorocyanus
singularis
aliniger
darlingtoni
hendersoni
dolichocephalus
bahorucoensis
monticola
equestris
luteogularis
smallwoodi
noblei
baracoae
vermiculatus
bartschi
lucius
argenteolus
vanidicus
macilentus
alfaroi
cyanopleurus
cupeyalensis
clivicola
inexpectatus
alutaceus
paternus
angusticeps
alayoni
sheplani
placidus
guazuma
loysiana
pumilis
centralis
argillaceus
porcatus
allisoni
isolepis
oporinus
altitudinalis
guamuhaya
chamaeleonides
porcus
barbatus
cuvieri
christophei
eugenegrahami
ricordii
barahonae
baleatus
marcanoi
strahmi
longitibialis
shrevei
cybotes
armouri
breslini
haetianus
whitemani
insolitus
fowleri
etheridgei
olssoni
semilineatus
alumina
barbouri
distichus
websteri
marron
caudalis
brevirostris
cristatellus
cooki
poncensis
gundlachi
pulchellus
krugi
stratulus
evermanni
reconditus
lineatopus
valencienni
grahami
opalinus
garmani
guafe
confusus
homolechis
jubar
mestrei
ophiolepis
quadriocellifer
bremeri
sagrei
imias
rubribarbus
allogus
ahli
R1
R3
R2

Specifying the priors

By default, the priors for each parameters are given as follows:

- $X_0 \sim \text{Normal}(\mu = 0, \sigma^2 = 100)$,
- $\alpha \sim \text{Half-Normal}(\sigma = \log(2)/(2 \cdot 0.05 \cdot \ell_T))$,
- $\theta \sim \text{Normal}(\mu = 0, \sigma^2 = 100)$,
- $\sigma \sim \text{Half-t}(\nu = 1, \sigma = 3\ell_T)$,

in which ℓ_T is the height of the tree.

We can customize the priors by changing the elements of the `model$priors` by `priorpdf` objects provided by `bgphy`:

```
?bgphy::priorpdf
```

For example, we want to modify priors of the regime "New" in the `BMOU` model:

```
BMOU$priors$alpha_2 <- prior_halfnormal(sigma = 2)
BMOU$priors$theta_2 <- prior_normal(mean = 2, sd = 5)
BMOU$priors$sigma_2 <- prior_halft(nu = 1, sigma = 2)
print(BMOU$priors)
```

```
X0 ~ normal(mean = 0, sd = 10)
sigma_1 ~ halft(nu = 1, sigma = 3)
alpha_2 ~ halfnormal(sigma = 2)
theta_2 ~ normal(mean = 2, sd = 5)
sigma_2 ~ halft(nu = 1, sigma = 2)
```

Running the posterior inference using `bgphy()`

The main function of the package is the `bgphy()` function, which takes arguments:

- `model`: an object of class `bgphy_model`, constructed by using `setModel()`.
- `X`: a data matrix of dimension $1 \times N$, where N is the number of tips of the tree. The column names of `X` must match the tip labels, that can be accessed by `model$tree$tip.label`.
- `nsample`: number of particles for the importance sampling method (the default is 10000).

- `scale`: scaling factor used for the estimated covariance matrix obtained from the Laplace approximation, defaults to 1.
- `parallel`: a Boolean variable determining the use of paralleled computing for the algorithm. By default is set to `TRUE`.

For this example, let us use simulated dataset `XOU` that is available in the package. This dataset contains 100 realizations of simulation under a certain OU process on the same tree that we currently use (Caribbean anoles). Since the data matrix needs to have column names that match the names on the tips of the tree, and since `XOU` consists of 100 realizations of the simulation, we can pick only one row for our example, and turn it into a format that the `bgphy()` accepts:

```
X <- matrix(XOU[1,], nrow = 1, dimnames = list(NULL, colnames(XOU)))
```

Let us run a Bayesian inference of the model specified by `BMOU` and using the data `X`, and using 100000 particles. The processing time of the algorithm is also recorded.

```
set.seed(1234, kind = "L'Ecuyer-CMRG")
start <- Sys.time()
post_BMOU <- bgphy(model = BMOU, X = X, nsample = 100000)
end <- Sys.time()
```

The `bgphy()` function outputs an object of class `bgphy_posterior`, and the posterior information can be seen by:

```
print(post_BMOU)
```

```
             mean   std_error        std      2.5%       50%     97.5%
X0      1.6969890 0.037240725 0.20342597 1.2602733 1.7274833 2.0338315
sigma_1 0.6607642 0.000592078 0.04779001 0.5701736 0.6593562 0.7710253
alpha_2 1.6539189 0.245208596 1.03210545 0.5792817 1.4614579 4.1067720
theta_2 1.4679745 0.085452457 0.30333132 0.8551299 1.4494112 1.8781583
sigma_2 1.2135716 0.020399047 0.23876043 0.8437996 1.1251695 1.8217511

WAIC: 135.55799157259
```

Where `mean` denotes the posterior mean, as seen from each parameter, with `std_error` determines the standard error of the `mean`. The column `std` shows the standard deviation of the posterior distribution, while the last three columns describe the 2.5-th, 50-th (median), and 97.5-th percentile of the posterior distribution. `WAIC` is the widely applicable information criterion score (a.k.a. Watanabe-Akaike information score), where a lower value indicates a better fit to the data.

Furthermore, it contains a matrix `Q` and a vector `W` which are samples from the proposal distribution (multivariate Normal) and the (normalized) importance weights.

The processing time takes around

```
end - start
```

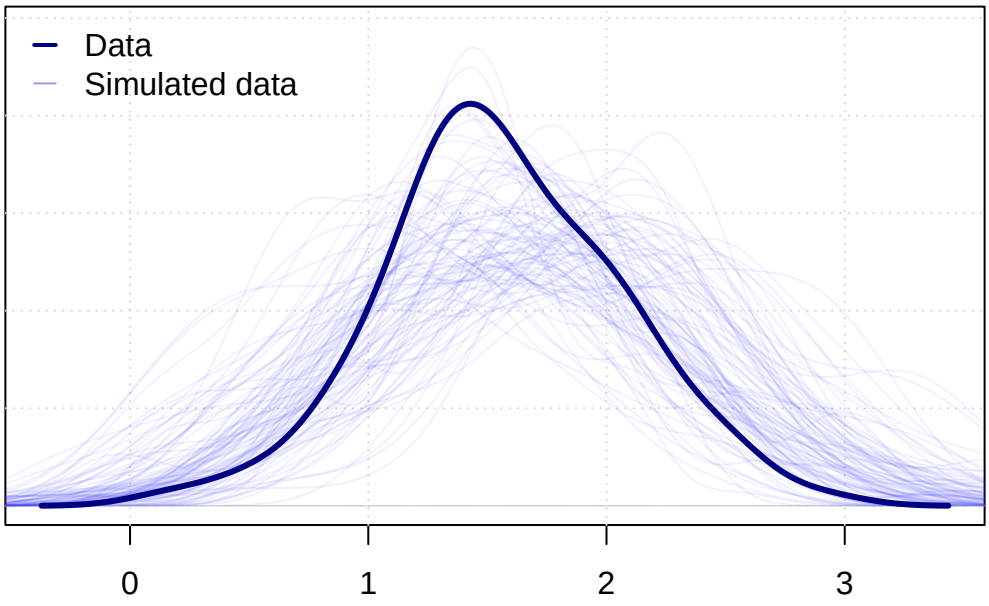
```
Time difference of 2.492461 mins
```

on a personal computer with 6-cores AMD Ryzen 5 PRO 4650U with Radeon Graphics 2.10 GHz processor, and 16 GB of RAM.

Posterior predictive distribution check using `post_pred_check()`

We can also see how the posterior distribution of our model fits the original data by simulating data from the posterior predictive distribution using `post_pred_check()`:

```
post_pred_check(post_BMOU)
```



As we can see, the simulated data do not look very similar to the original data, since we assumed a mixed BM to OU model while the original data was simulated from a global OU model.

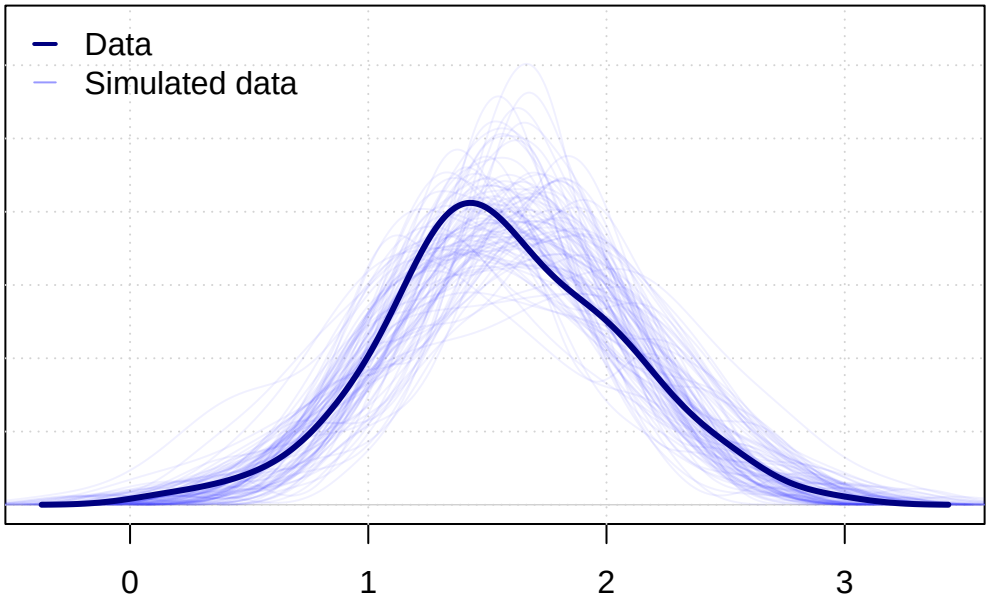
Let us now do the same procedure, but with an OU model defined in `OU`:

```
set.seed(1234, kind = "L'Ecuyer-CMRG")
post_OU <- bgphy(model = OU, X = X, nsample = 100000)
print(post_OU)
```

```
             mean    std_error       std        2.5%      50%     97.5%
X0    -0.04072598 0.250516609 6.7173087 -13.8042433 0.118926 14.338151
alpha  2.68626668 0.033208425 0.6548028   1.4919360 2.597807  4.209463
theta  1.72393757 0.022916946 0.5517664   0.5638046 1.688757  2.771920
sigma  1.11785342 0.004589569 0.1323854   0.8948859 1.106145  1.418570

WAIC: 123.546521423371
```

As we can see, the WAIC score is lower (better) than the previous mixed-model. The simulated data curves from the posterior predictive distribution also resemble the original data curve better, as shown on the following plot:

```
post_pred_check(post_OU)
```



Simulating posterior samples using `post_simulate()`

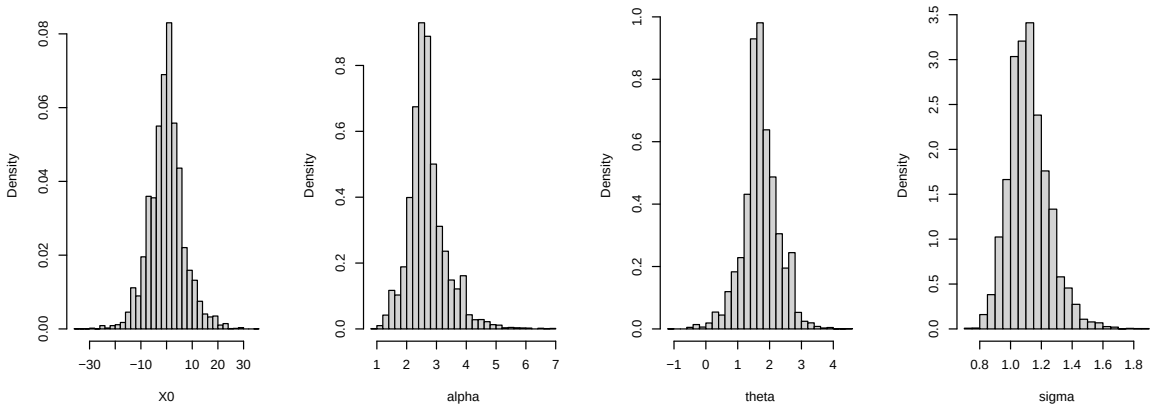
To simulate samples from the posterior distribution, we can use sampling-importance resampling by sampling samples (with replacement) from proposal distribution with importance weights as the probabilities. This can be done by using `post_simulate()` function on the `bgphy_posterior` object:

```
set.seed(1234, kind = "L'Ecuyer-CMRG")
post_OU_samples <- post_simulate(post_OU)
head(post_OU_samples)
```

```
             X0    alpha     theta    sigma
[1,]   1.5727945 2.973173 1.5554914 1.108855
[2,]  13.0507437 2.776830 0.7852093 1.031283
[3,]   2.9599580 2.330741 1.4968732 1.127269
[4,]  -1.5908889 2.251417 1.9581028 1.185786
[5,] -13.5684484 2.709752 2.6254801 1.075217
[6,]  -0.9957337 3.116677 1.7118992 1.166816
```

The marginal posterior distributions can then be visualized:

```
parnames <- names(post_OU$mean)
par(mfrow = c(1,4))

for (i in 1:4){
  hist(post_OU_samples[,i], freq = FALSE, breaks = 25, xlab = parnames[i], main = "")
}
```



References

Bastide, Paul, and Gilles Didier. 2023. "The Cauchy Process on Phylogenies: A Tractable Model for Pulsed Evolution." *Systematic Biology*, August, syad053. https://doi.org/10.1093/sysbio/syad053.

Mahler, D Luke, Travis Ingram, Liam J Revell, and Jonathan B Losos. 2013. "Exceptional Convergence on the Macroevolutionary Landscape in Island Lizard Radiations." *Science* 341 (6143): 292–95.

Mitov, Venelin, Krzysztof Bartoszek, Georgios Asimomitis, and Tanja Stadler. 2020. "Fast Likelihood Calculation for Multivariate Gaussian Phylogenetic Models with Shifts." *Theoretical Population Biology* 131: 66–78.