Animal Model Comparison

Description

References

Libraries

```
library(AlphaSimR)
```

```
## Loading required package: R6
```

```
library(AGHmatrix)
library(MCMCglmm)
```

```
## Loading required package: Matrix
```

```
## Loading required package: coda
```

```
## Loading required package: ape
```

```
library(brms)
```

```
## Loading required package: Rcpp
```

```
## Loading 'brms' package (version 2.14.4). Useful instructions
## can be found by typing help('brms'). A more detailed introduction
## to the package is available through vignette('brms_overview').
```

```
##
## Attaching package: 'brms'
```

```
## The following object is masked from 'package:MCMCglmm':
##
##     me
```

```
## The following object is masked from 'package:stats':
##
##     ar
```

```
library(rstan)
```

```
## Loading required package: StanHeaders
```

```
## Loading required package: ggplot2
```

```
## rstan (Version 2.21.2, GitRev: 2e1f913d3ca3)
```

```
## For execution on a local, multicore CPU with excess RAM we recommend calling
## options(mc.cores = parallel::detectCores()).
## To avoid recompilation of unchanged Stan programs, we recommend calling
## rstan_options(auto_write = TRUE)
```

```
##
## Attaching package: 'rstan'
```

```
## The following object is masked from 'package:coda':
##
##     traceplot
```

```
options(mc.cores = parallel::detectCores())
rstan_options(auto_write = TRUE)
```

Simulated Population

```
## Founder population
FOUNDERPOP <- runMacs(nInd = 100,
                      nChr = 20,
                      inbred = FALSE,
                      species = "GENERIC")

## Simulation parameters
SIMPARAM <- SimParam$new(FOUNDERPOP)
SIMPARAM$addTraitA(nQtlPerChr = 100,
                   mean = 100,
                   var = 10)
SIMPARAM$setGender("yes_sys")
SIMPARAM$setVarE(h2 = 0.3)

## Random mating for 9 more generations
generations <- vector(mode = "list", length = 10)
generations[[1]] <- newPop(FOUNDERPOP,
                           simParam = SIMPARAM)

for (gen in 2:10) {

  generations[[gen]] <- randCross(generations[[gen - 1]],
                                  nCrosses = 10,
                                  nProgeny = 10,
                                  simParam = SIMPARAM)

}

## Put them all together
combined <- Reduce(c, generations)

## Extract phentoypes
pheno <- data.frame(animal = combined@id,
                    pheno = combined@pheno[,1])

# Important to scale phenotype otherwise priors could be incorrect
pheno$scaled_pheno <- as.vector(scale(pheno$pheno))

## Extract pedigree
ped <- data.frame(id = combined@id,
                  dam = combined@mother,
                  sire =combined@father)
ped2 <- ped
ped2$dam[ped$dam == 0] <- NA
```

```r
ped2$sire[ped$sire == 0] <- NA
```

MCMCglmm

```r
## Gamma priors for variances
prior_gamma <- list(R = list(V = 1, nu = 1),
                    G = list(G1 = list(V = 1, nu = 1)))

## Fit the model
model_mcmc  <- MCMCglmm(scaled_pheno ~ 1,
                        random = ~ animal,
                        family = "gaussian",
                        prior = prior_gamma,
                        pedigree = ped2,
                        data = pheno,
                        nitt = 100000,
                        burnin = 10000,
                        thin = 10)

## Calculate heritability for heritability from variance components
h2_mcmc_object  <- model_mcmc$VCV[, "animal"] /
  (model_mcmc$VCV[, "animal"] + model_mcmc$VCV[, "units"])

## Summarise results from that posterior
h2_mcmc  <- data.frame(mean = mean(h2_mcmc_object),
                       lower = quantile(h2_mcmc_object, 0.025),
                       upper = quantile(h2_mcmc_object, 0.975),
                       method = "MCMC",
                       stringsAsFactors = FALSE)
```

BRMS

```r
# ped matrix needs to have 0s for missing values NOT NAs
  A  <- Amatrix(ped)

  # Run model
  model_brms <- brm(scaled_pheno ~ 1 + (1|animal),
                    data = pheno,
                    family = gaussian(),
                    cov_ranef = list(animal = A),
                    chains = 4,
                    cores = 1,
                    iter = 2000)

# Examine posterior distribution
posterior_brms <- posterior_samples(model_brms,
                                    pars = c("sd_animal", "sigma"))

# Calculate heritability for each sample
h2_brms  <- posterior_brms[,1]^2 /
  (posterior_brms[,1]^2 + posterior_brms[,2]^2)
```

```r
# Mean heritability for all samples
h2_brms  <- data.frame(mean = mean(h2_brms),
                       lower = quantile(h2_brms, 0.025),
                       upper = quantile(h2_brms, 0.975),
                       method = "BRMS",
                       stringsAsFactors = FALSE,
                       row.names=NULL)
```

Heritability comparison

```r
h2 <- rbind(h2_mcmc,h2_brms)
```