BATools: An R Package for Whole Genomic Analysis with Bayesian Models

Chunyu Chen, Lei Zhou, Robert J. Tempelman

2015-09-23

The package `BATools` is used to perform genome-wide association using a various Bayesian models. It is a implemented using both Markov Chain Monte Carlo (MCMC) and expectation maximization (EM) algorithm.

The basic functions in `BATools` is `bafit`, which fits a genomic selection model using different prior selection. The main characteristic of this package are:

- Fit model with different prior specification including the Antedependence model
- Flexibility to choose between MCMC and EM algorithm, both of which are able to estimate the hyperparameters
- Accepts `gpData` objects from package `synbreed` as input data. It can also use `numeric` and `matrix` as input
- It is computationally efficient
- GWA using EM BayesA/C (under development)

1. Introduction

Whole genome prediction (WGP) is an evolutionary development in animal breeding. Currently, many models have been developed for WGP, which included rrBLUP, BayesA, BayesB, BayesC, Bayesian Lasso, Antedepedence BayesA/B (Meuwissen et al. 2001, VanRaden 2008, de los Campos et al. 2009, Habier et al. 2011, and Yang and Tempelman 2012). The major difference of these models are different prior assumptions on marker effects. Software packages like `BGLR` and `GenSel` implement BayesA, BayesB and Bayes Lasso model using MCMC algorithm. No public software is available to implement Antedependence models. At the same time, no R package is available for implement BayesA/C using EM algorithm for animal breeding. `BATools` package provides tools to fit Antedependence models in addition to some of the most popular models and provider faster EM algorithms to fit the model. The table below is a comparison between `BATools` and `BGLR`:

Model/Algorithms	MCMC	EM
rrBLUP	BATools/BGLR	BATools
BayesA	BATools/BGLR	BATools
BayesB	BATools/BGLR	under development
BayesC	BATools/BGLR	BATools
Bayesian Lasso	BATools/BGLR	under development
AnteBayesA	BATools	under development
AnteBayesB	BATools	under development

2. Basic Model

The basic model used by `BATools` is:

$$\boldsymbol{y} = \boldsymbol{X} \cdot \boldsymbol{b} + \boldsymbol{Z} \cdot \boldsymbol{g} + \boldsymbol{e},$$

where:

- $\boldsymbol{y}$ is the vector of response variables
- $\boldsymbol{X} \cdot \boldsymbol{b}$ models the fixed effects
- $\boldsymbol{g}$ is the SNP marker effect and $\boldsymbol{Z}$ is corresponding genotype matrix of $n \cdot m$
- $\boldsymbol{e}$ are the vector of effects residual, $\boldsymbol{e} \sim N\left(\boldsymbol{0}, \boldsymbol{I}\sigma_e^2\right)$

Notice that for different models, the priors on $\boldsymbol{g}_i$ are different:

- rrBLUP: $\boldsymbol{g}_j \sim N\left(\boldsymbol{0}, \boldsymbol{I}\sigma_g^2\right)$
- BayesA: $\boldsymbol{g}_j \sim N\left(\boldsymbol{0}, \boldsymbol{D}\sigma_g^2\right)$, where $\boldsymbol{D} = \{\tau_1, \tau_2, ..., \tau_m\}$ and $\tau_j \sim \chi^{-2}\left(\nu_g, \nu_g\right)$
- BayesB: $\boldsymbol{g}_j \sim N\left(\boldsymbol{0}, \boldsymbol{D}\sigma_g^2\right)$, where $\boldsymbol{D} = \{\tau_1, \tau_2, ..., \tau_m\}$ and

$$\tau_j = \begin{cases} 0 & \text{with probability} \quad \pi \\ \sim \chi^{-2}\left(\nu_g, \nu_g\right) & \text{with probability} \quad 1-\pi \end{cases}$$

- BayesC: $\boldsymbol{g}_i \sim N\left(\boldsymbol{0}, \boldsymbol{D}\sigma_g^2\right)$, where $\boldsymbol{D} = \{\tau_1 + \frac{1-\tau_1}{c}, \tau_2 + \frac{1-\tau_2}{c}, ..., \tau_m + \frac{1-\tau_m}{c}\}$ and $\tau_j \sim Bernoulli(\pi)$, $\tau_j = 0, 1$
- Bayesian Lasso: $\boldsymbol{g}_j \sim N\left(\boldsymbol{0}, \boldsymbol{D}\sigma_g^2\right)$, where $\boldsymbol{D} = \{\tau_1, \tau_2, ..., \tau_m\}$ and $\tau_j \sim Exp\left(\lambda^2\right)$

Furthermore, the Antedepedence models specify correlation structure for $\boldsymbol{g}$ based on the relative physical location of SNP markers along the chromosome :

$$g_j = \begin{cases} \delta_j & \text{if} \quad j = 1 \\ t_{j,j-1}\delta_{j-1} + \delta_j & \text{if} \quad 2 \leq j \leq m \end{cases}$$

where $t_{j,j-1} \sim N\left(\mu_t, \sigma_t^2\right)$

3. BATools example

In `BATools`, we adhered the data structure of the object `gpData` in the `synbreed` package. The input and output objects are named as `baData` and `BAout`, which are R object class `list`. Therefore, users can directly use `synbreed` object as the input for `BATools`, and vice versa. More detailed explanation about `baData` and `BAout` can be found in the package manual file.

Example

We will use a toy dataset from the MSUPRP population to illustrate the use of `BATools`

Load packages and data

```
library(BATools)
data("MSUPRP_sample")
summary(MSUPRP_sample)
```

```
## object of class 'gpData'
## covar
##   No. of individuals 253
##            phenotyped 176
##             genotyped 251
## pheno
##   No. of traits:        3
##
##      ph_24h         temp_24h        driploss
##  Min.   :5.190   Min.   :1.100   Min.   :0.000
##  1st Qu.:5.450   1st Qu.:1.600   1st Qu.:0.560
##  Median :5.540   Median :1.900   Median :0.940
##  Mean   :5.552   Mean   :1.983   Mean   :1.141
##  3rd Qu.:5.640   3rd Qu.:2.300   3rd Qu.:1.545
##  Max.   :6.350   Max.   :3.400   Max.   :4.330
##  NA's   :35      NA's   :18      NA's   :18
##
## geno
##   No. of markers 20597
##   genotypes 0 1 2
##   frequencies 0.287 0.404 0.308
##   NA's 0.000 %
## map
##   No. of mapped markers  20597
##   No. of chromosomes     18
```

```
##
##   markers per chromosome
##
##    1    2    3    4    5    6    7    8    9   10   11   12   13   14   15
## 1968 1419 1261 1559 1076 1143 1289  939 1426  958  936  833 1319 1263 1079
##   16   17   18
##  972  739  418
##
## pedigree
## Number of
##   individuals  253
##   males :  98 , females :  155
##   Par 1 (sire)  10
##   Par 2 (dam)   44
##   generations  3
```

Then we can create the data object used in `BATools` by `create.baData`. In this example we treat the sex as fixed effects

```
pheno<-data.frame(MSUPRP_sample$pheno[,,])
geno<-MSUPRP_sample$geno[,1:500]
ped<-MSUPRP_sample$pedigree
map=MSUPRP_sample$map
sex<-ped$sex
sex<-as.factor(sex)
x<-model.matrix( ~ sex -1,contrasts.arg=list(sex=contrasts(sex, contrasts=F)))
colnames(x)<-c("female","male")
rownames(x)<-ped$ID
pig=create.baData(pheno=pheno,geno=geno,map=map,pedigree=ped,fixed=x,makeAinv=F)
```

Set up initial values for the model

We choose to demonstrate how to fit BayesA using MCMC and EM. We start with MCMC:

```
init=list(df=5,scale=0.01,pi=1)
run_para=list(niter=5000,burnIn=2500,skip=10)
print_mcmc=list(piter=500)
update_para=list(df=FALSE,scale=TRUE,pi=FALSE)
op<-create.options(model="BayesA",method="MCMC",ante=FALSE,priors=NULL,init=init,
  update_para=update_para,run_para=run_para,save.at="BayesA",cv=NULL,print_mcmc=print_mcmc)
```

Fit the model

We then fit the model using MCMC for the trait `driploss` with the above setups:

```
ba<-bafit(dataobj=pig,op=op,trait="driploss")
```

```
## iter=  500  vare=  0.617604 scale=  0.0009667 timepercycle=  0 estimated time left= 1.8
## iter=  1000  vare=  0.602824 scale=  0.00070879 timepercycle=  0.001 estimated time left=
## iter=  1500  vare=  0.698644 scale=  0.00028016 timepercycle=  0 estimated time left= 1.6
## iter=  2000  vare=  0.589461 scale=  0.00085074 timepercycle=  0 estimated time left= 1.3
## iter=  2500  vare=  0.574677 scale=  0.00089771 timepercycle=  0 estimated time left= 1.1
## iter=  3000  vare=  0.491537 scale=  0.00032364 timepercycle=  0 estimated time left= 0.8
## iter=  3500  vare=  0.607441 scale=  0.00066083 timepercycle=  0 estimated time left= 0.6
## iter=  4000  vare=  0.596305 scale=  0.00037768 timepercycle=  0 estimated time left= 0.4
## iter=  4500  vare=  0.551192 scale=  0.00092832 timepercycle=  0 estimated time left= 0.2
## iter=  5000  vare=  0.401027 scale=  0.00104794 timepercycle=  0 estimated time left= 0
```

```
ba
```

```
## BATools analysis of trait: driploss
##
## estimated fixed effects:
##    female      male
## 0.9749968 0.8689422
##
## estimated hyperparameters:
##         vare         varg
## 0.5203206886 0.0006049592
##
## effective sample size for hyperparameters:
##      vare      varg
## 500.00000  11.46826
```

Graphics

We can obtain the traceplot for MCMC:

```
par(mar=c(2,2,2,2))
baplot(dataobj=pig,BAout=ba,type="trace",op=op)
```

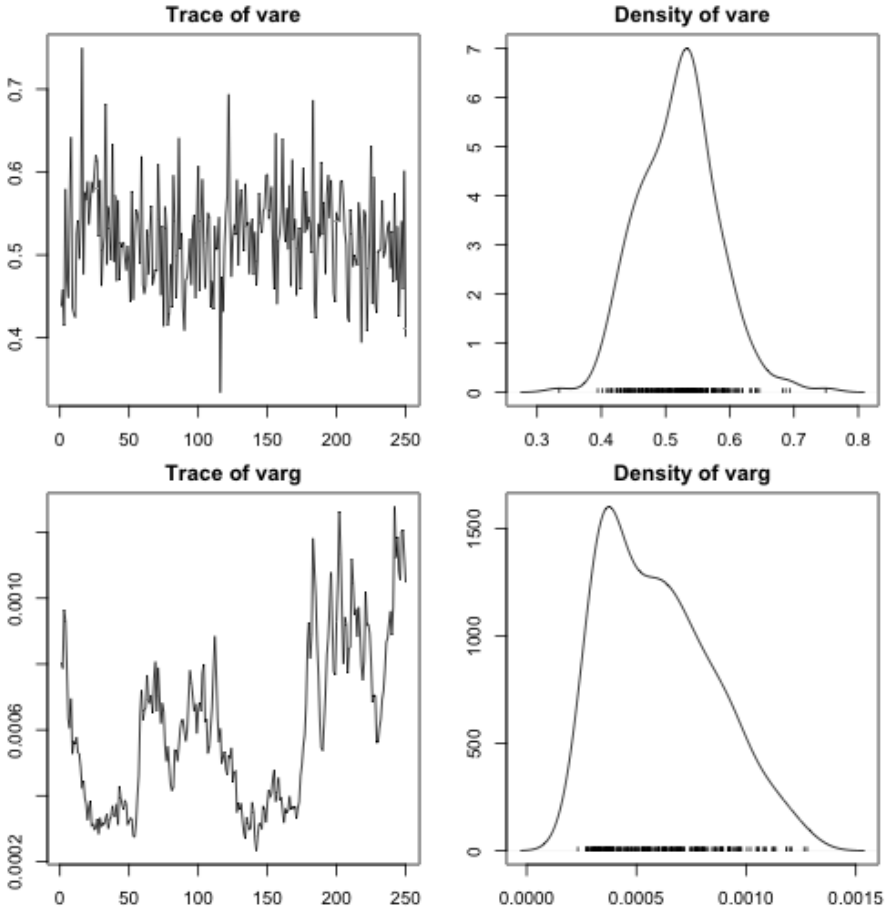

Figure 1: Traceplot

EM algorithm

To use the EM algorithm in `BATools`, we first run an analysis using rrBLUP:

```
##################run rrBLUP REML#####################
init=list(df=NULL,scale=NULL,vare=NULL)
run_para=list(maxiter=100)
update_para=list(df=FALSE,scale=TRUE)
op<-create.options(model="rrBLUP",method="EM",ante=FALSE,priors=NULL,init=init,
    update_para=update_para,run_para=run_para,save.at="rrBLUP",cv=NULL,print_mcmc=NULL,conv

rr<-bafit(dataobj=pig,op=op,trait="dripploss")

## rrBLUP iter= 1
## Residual Variance is 0.5133895 Genetic Variance is 0.0006355719
## rrBLUP iter= 2
## Residual Variance is 0.5138686 Genetic Variance is 0.0006409364
## rrBLUP iter= 3
## Residual Variance is 0.5138934 Genetic Variance is 0.0006406405
## rrBLUP iter= 4
## Residual Variance is 0.5138917 Genetic Variance is 0.000640663
## rrBLUP iter= 5
## Residual Variance is 0.5138919 Genetic Variance is 0.0006406614
##
## rrBLUP converged after 5 iterations and the convergence critira is 2.489739e-07

rr

## BATools analysis of trait: dripploss
##
## estimated fixed effects:
##   female     male
## 1.211052 1.103184
##
## estimated hyperparameters:
##         vare         varg
## 0.5138918681 0.0006406614
```

Then we use rrBLUP results as starting values for EM BayesA:

```
df_i=5
scale_i=(df_i-2)/df_i*rr$hyper_est[2]
init=list(df=df_i,scale=scale_i,vare=rr$hyper_est[1],g=rr$ghat,b=rr$betahat,pi=1)
run_para=list(maxiter=100)
```

```
update_para=list(df=FALSE,scale=TRUE,pi=FALSE)
op<-create.options(model="BayesA",method="EM",ante=FALSE,priors=NULL,init=init,D="V",
    update_para=update_para,run_para=run_para,save.at="BayesA",cv=NULL,print_mcmc=NULL)
ba_em<-bafit(dataobj=pig,op=op,trait="dripploss")
```

```
## BayesA EM iter= 1
## Residual Variance is 0.517764 Genetic Variance is 0.0005838004
## BayesA EM iter= 2
## Residual Variance is 0.5171621 Genetic Variance is 0.0006592904
## BayesA EM iter= 3
## Residual Variance is 0.5168152 Genetic Variance is 0.0006660427
## BayesA EM iter= 4
## Residual Variance is 0.5166557 Genetic Variance is 0.0006666532
## BayesA EM iter= 5
## Residual Variance is 0.5166276 Genetic Variance is 0.0006667861
## BayesA EM iter= 6
## Residual Variance is 0.5166233 Genetic Variance is 0.0006668084
## BayesA EM iter= 7
## Residual Variance is 0.5166225 Genetic Variance is 0.0006668124
## BayesA EM iter= 8
## Residual Variance is 0.5166224 Genetic Variance is 0.0006668131
##
## BayesA converged after 8 iterations and the convergence critira is 2.729907e-07
```

```
ba_em
```

```
## BATools analysis of trait: driploss
##
## estimated fixed effects:
##   female     male
## 1.211268 1.102378
##
## estimated hyperparameters:
##         vare         varg
## 0.5166223676 0.0006668131
```

Graphics

Let's look at the estimated phenotypes v.s. true phenotypes for EM:

We can also compare the difference bewteen MCMC and EM:

```
plot(ba$ghat,ba_em$ghat,xlab="MCMC",ylab="EM",main="BayesA MCMC v.s. EM")
abline(a=0,b=1)
```

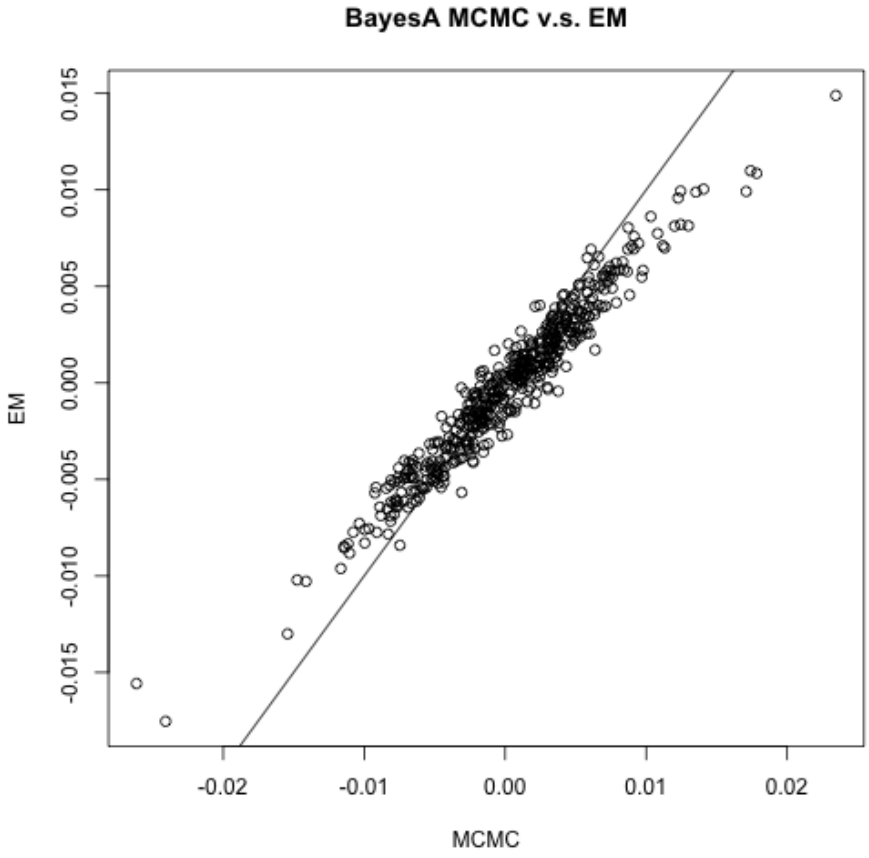
BayesA MCMC v.s. EM
EM
MCMC
0.015
0.010
0.005
0.000
-0.005
-0.010
-0.015
-0.02
-0.01
0.00
0.01
0.02

Figure 2:

BayesC

Running BayesC is similar to running BayesA:

```
init=list(df=5,scale=0.2,pi=0.1,c=1000)
run_para=list(niter=2000,burnIn=1000,skip=1)
print_mcmc=list(piter=200)
update_para=list(df=F,scale=T,pi=T)
priors=list(shape_scale=5,rate_scale=0.1,alphapi=1,betapi=9)
op<-create.options(model="BayesC",method="MCMC",
ante=FALSE,priors=NULL,init=init,update_para=update_para,
run_para=run_para,save.at="BayesCC",cv=NULL,print_mcmc=print_mcmc)
bc<-bafit(dataobj=pig,op=op,trait="driploss")
```

```
## iter=  200  vare=  0.6248 scale=  0.02515235 timepercycle=  0.001 estimated time left= 0.
## iter=  400  vare=  0.429829 scale=  0.07978027 timepercycle=  0.001 estimated time left=
## iter=  600  vare=  0.508852 scale=  0.12238383 timepercycle=  0.001 estimated time left=
## iter=  800  vare=  0.498089 scale=  0.06532467 timepercycle=  0.001 estimated time left=
## iter=  1000  vare=  0.497922 scale=  0.01748966 timepercycle=  0.001 estimated time left=
## iter=  1200  vare=  0.568778 scale=  0.01598056 timepercycle=  0.001 estimated time left=
## iter=  1400  vare=  0.485576 scale=  0.0117588 timepercycle=  0.001 estimated time left=
## iter=  1600  vare=  0.493141 scale=  0.01304677 timepercycle=  0.001 estimated time left=
## iter=  1800  vare=  0.53983 scale=  0.02696047 timepercycle=  0.001 estimated time left=
## iter=  2000  vare=  0.788232 scale=  0.01792949 timepercycle=  0.001 estimated time left=
```

```
bc
```

```
## BATools analysis of trait: driploss
##
## estimated fixed effects:
##    female      male
## 0.7660304 0.6738021
##
## estimated hyperparameters:
##       vare       varg         pi
## 0.52041266 0.01695496 0.05496325
##
## effective sample size for hyperparameters:
##       vare       varg         pi
## 137.962665  14.755601   6.819892
```

```
scale_i=rr$hyper_est[2]/(1/1000*rr$hyper_est[2]*(1-bc$hyper_est[3])+bc$hyper_est[3]*rr$hyper
init=list(df=5,scale=scale_i,vare=rr$hyper_est[1],g=rr$ghat,b=rr$betahat,pi=bc$hyper_est[3],
run_para=list(maxiter=100)
```

```
update_para=list(df=FALSE,scale=TRUE,pi=T)
op<-create.options(model="BayesC",method="EM",ante=FALSE,priors=NULL,init=init,
    update_para=update_para,run_para=run_para,save.at="BayesC",cv=NULL,print_mcmc=NULL,conv
bc_em<-bafit(dataobj=pig,op=op,trait="dripploss")

## BayesC EM iter= 1
## Residual Variance is 0.7144196 Genetic Variance is 8.943218 pi is 0.00180786
## BayesC EM iter= 2
## Residual Variance is 0.6409612 Genetic Variance is 4.471609 pi is 5.899198e-05
## BayesC EM iter= 3
## Residual Variance is 0.5779692 Genetic Variance is 2.235805 pi is 1.889715e-06
## BayesC EM iter= 4
## Residual Variance is 0.5405971 Genetic Variance is 1.117902 pi is 6.057554e-08
## BayesC EM iter= 5
## Residual Variance is 0.5222646 Genetic Variance is 0.1946135 pi is 1.943242e-09
## BayesC EM iter= 6
## Residual Variance is 0.5244976 Genetic Variance is 0.3849745 pi is 6.677803e-11
## BayesC EM iter= 7
## Residual Variance is 0.5185562 Genetic Variance is 0.5210157 pi is 2.091794e-12
## BayesC EM iter= 8
## Residual Variance is 0.5176154 Genetic Variance is 0.5535387 pi is 6.579907e-14
## BayesC EM iter= 9
## Residual Variance is 0.5176008 Genetic Variance is 0.5546304 pi is 2.077079e-15
## BayesC EM iter= 10
## Residual Variance is 0.5176014 Genetic Variance is 0.5546241 pi is 6.556435e-17
## BayesC EM iter= 11
## Residual Variance is 0.5176014 Genetic Variance is 0.5546242 pi is 2.185478e-18
##
## BayesC converged after 11 iterations and the convergence critira is 6.311663e-08

bc_em

## BATools analysis of trait: dripploss
##
## estimated fixed effects:
##   female     male
## 1.209711 1.100544
##
## estimated hyperparameters:
##         vare         varg           pi
## 5.176014e-01 5.546242e-01 2.185478e-18
```

We can also compare the difference bewteen MCMC and EM for BayesC:

```
plot(bc$ghat,bc_em$ghat,xlab="MCMC",ylab="EM",main="BayesC MCMC v.s. EM")
abline(a=0,b=1)
```

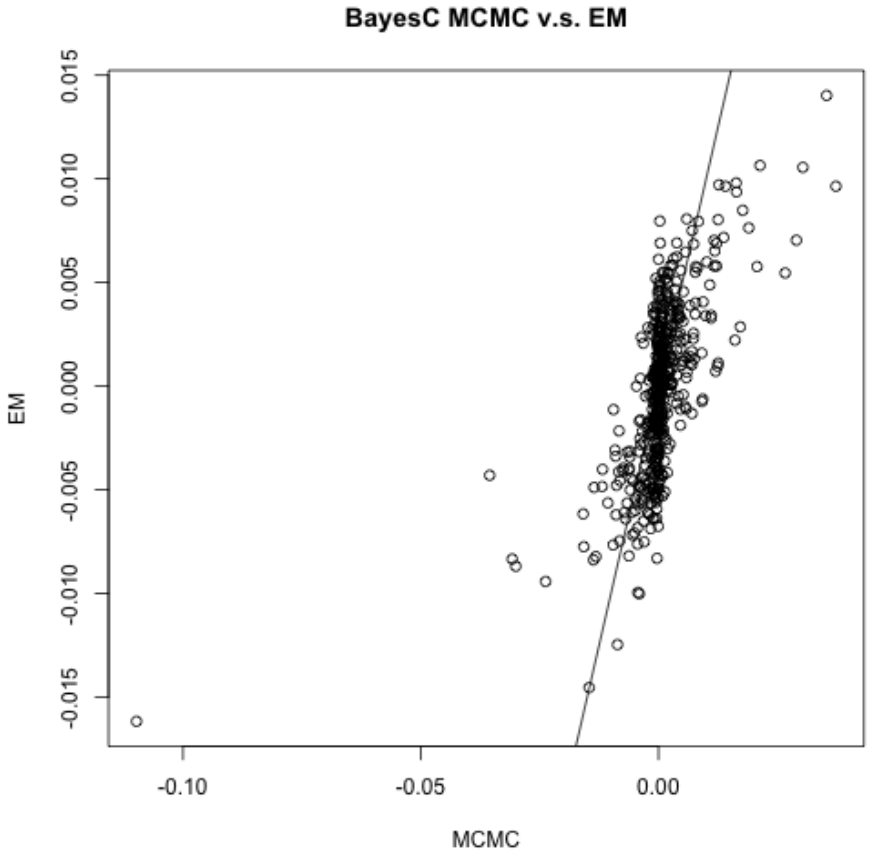

Figure 3: BayesC

We can also compare the difference bewteen BayesA and BayesC for MCMC:

```
plot(ba$ghat,bc$ghat,xlab="BayesA",ylab="BayesC",main="BayesA v.s. BayesC in MCMC")
abline(a=0,b=1)
```

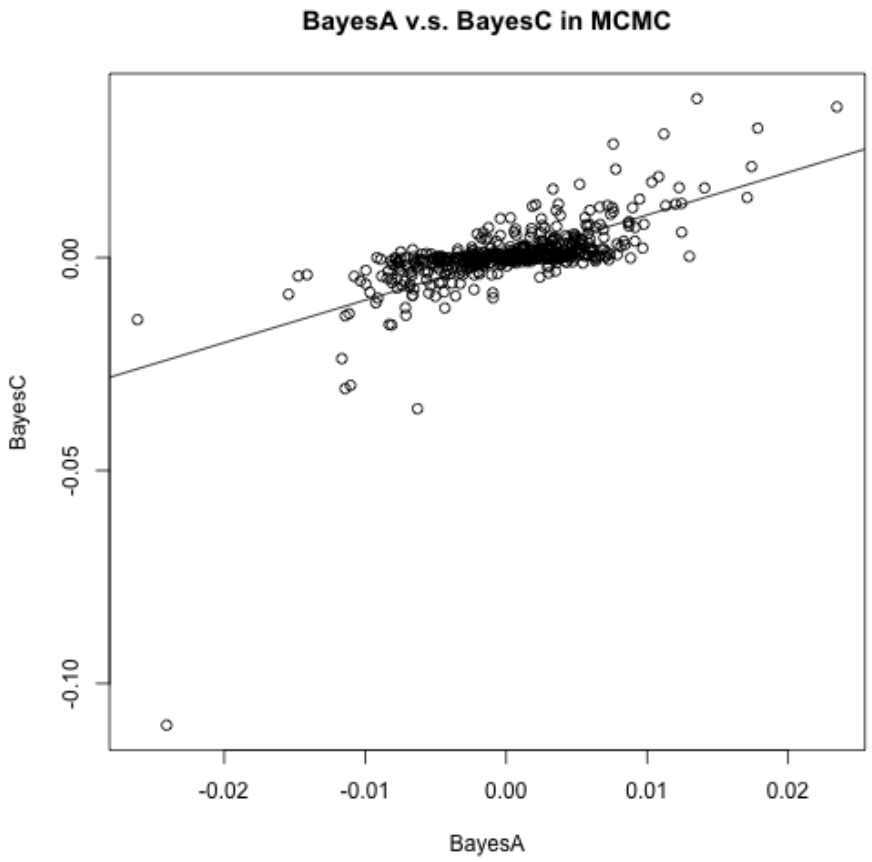

Figure 4: BayesA_C_MCMC

We can also compare the difference bewteen BayesA and BayesC for EM:

```
plot(ba$ghat,bc$ghat,xlab="BayesA",ylab="BayesC",main="BayesA v.s. BayesC in EM")
abline(a=0,b=1)
```

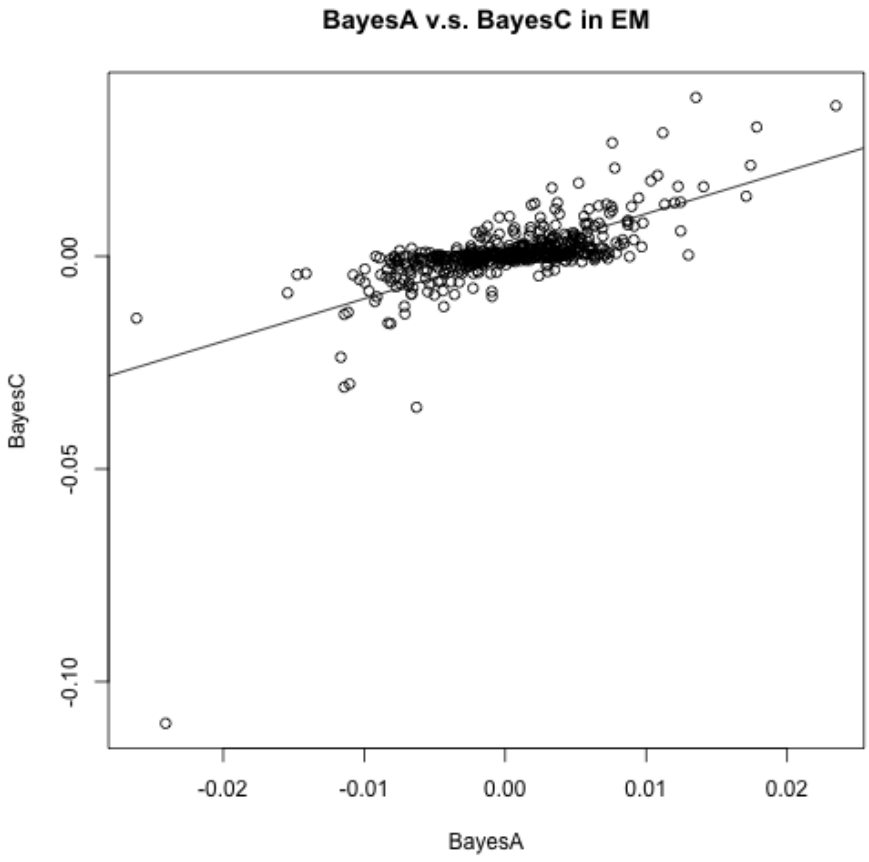

Figure 5: BayesA_C_EM