Simple manual of AFEchidna

Version: V1.54
2021-8-27

1 Description

AFEchidna is a R package developed from the software Echidna V1.54 by Arthour Gilmour. The AFEchidna package is free to the public, please email to yzhlinscau@163.com to get a copy. The AFEchidna package is for academic research only and not for commercial use.

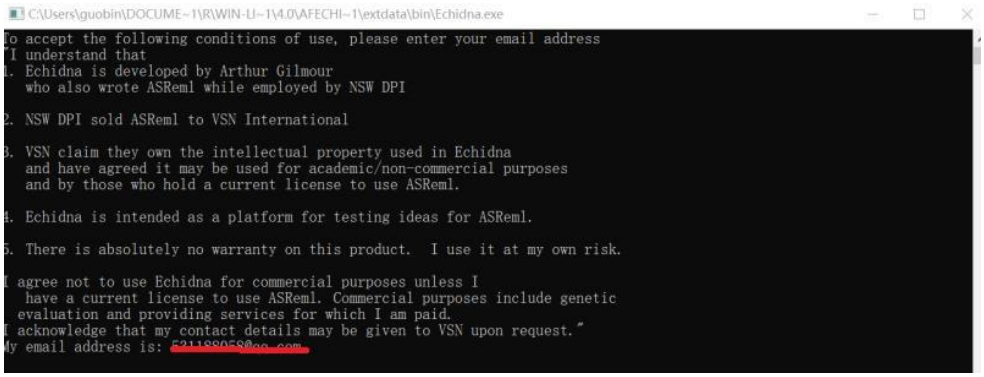
2 Installation

AFEchidna package could be installed from local file AFEchidna_1.54.zip, or run remotes::install_github('yzhlinscau/AFEchidna') to install online from Github, then run `AFEchidna::CheckPack()` in R to check whether the R dependent package is installed or not. If not, it will be installed automatically.

Note: if Echidna software is first time for user, user should register an email address as following:

```
## create a folder 'D:/Rtraining'
dir.create('D:/Rtraining')

## copy data file fm.csv to D:/Rtraining
AFEchidna::read.example(package='AFEchidna',setpath = TRUE)
file.copy("fm.csv",'D:/Rtraining')

setwd('D:/Rtraining')

library(AFEchidna)

## generate .es file
get.es0.file(dat.file='fm.csv',message=T)
```

Then a window would appear, user enters an email address and turns off the window.



Note: When new version of AFEchidna is available, user could run `AFEchidna::loadsoft(update=TRUE)` in R to update Echidna version.

3 workflow

step one– – generate temple .es0 file;
step two–– specify the mixed model;
step three–– check the results.

(1) temple .es0 file

if a data file is 'Provenance.csv', using `get.es0.file()` to get .es0 file:

```
library(AFEchidna)

get.es0.file(dat.file='Provenance.csv')  # .es file
get.es0.file(es.file='Provenance.es')    # .es0 file
```

The example for .es0 file, please see the append files.

Note: If the names of factor variables were started by capital letter, `get.es0.file()` would classify them automatically.

(2) specify the mixed model

```
HT.esr<-echidna(fixed=height~1+Prov,
                random=~Female+Block+Female:Block,
                residual=~units,
                predict='Prov',
                es0.file='Provenance.es0'
                )
```

(3) check the results

```
names(HT.esr)
names(HT.esr$org.par)
HT.esr$org.par$call

plot(HT.esr)

Var(HT.esr)

IC(HT.esr)

## fixed effect solution
coef(HT.esr)$fixed

## random effect solution
raneff<-coef(HT.esr)$random

library(dplyr)
```

```
raneff %>% filter(Term=='Female') %>% head
raneff %>% filter(Term=='Female') %>% arrange(desc(Effect)) %>% head

## calculate accuracy of random effects
gca<-raneff %>% filter(Term=='Female')
head(gca)
names(gca)[3]<-'gca' # rename 'Effect' into 'gca'

gca<-raneff.acc(HT.esr,gca,Var=0.19)
gca$bv<-2*gca$gca    # bv = 2* gca
head(gca)

## calculate genetic parameters
pin(HT.esr)

pin(HT.esr,mulp=c(famr~V3/(V1+V3+V4),
                  plotr~V4/(V1+V3+V4),
                  errorr~V1/(V1+V3+V4),
                  h2~V3*4/(V1+V3+V4)))


## prediction
HT.pred<-predict(HT.esr)

HT.pred$pred
HT.pred$ased
```

4 simple case:

4.1 case one– half-sib dataset– family model

single trait:

```
HT.esr<-echidna(fixed=height~1+Prov,
                random=~Female+Block+Female:Block,
                residual=~units,
                predict='Prov',
                es0.file='Provenance.es0'
                )
```

batch analyse of single trait:

```
bst<-echidna(trait=~height+diameter+volume,
                fixed=~1+Prov,
                random=~Female+Block+Female:Block,
                residual=~units,
                predict='Prov',
                batch=T,
                es0.file='Provenance.es0'
```

```
                )

Var(bst)
```

two trait:

```
HD.esr<-echidna(fixed=cbind(height,diameter)~Trait+Trait:Prov,
                random=~us(Trait):Female,
                residual=~units:us(Trait),
                predict='Prov',
                mulT=T,
                es0.file='Provenance.es0'
                )
```

batch analyse of multiple trait:

```
bsm<-echidna(trait=~height+diameter+volume,
                fixed=~Trait+Trait:Prov,
                random=~us(Trait):Female,
                residual=~units:us(Trait),
                batch=T,mulT=T,mulN=2,
                predict='Prov',
                es0.file='Provenance.es0'
                )
Var(bsm)
```

4.2 case two– half-sib dataset– animal model

single trait:

```
tm.esr<-echidna(fixed=height~1+Prov,
                random=~nrm(Treeid)+Block,
                residual=~units,
                predict='Prov',
                es0.file='Provenance.es0')
```

batch analyse of single trait:

```
bst2<-echidna(trait=~height+diameter+volume,
                fixed=~1+Prov,
                random=~nrm(Treeid)+Block,
                residual=~units,
                predict='Prov',
                batch=T,
                es0.file='Provenance.es0'
                )

Var(bst2)
```

two traits:

```
HDt.esr<-echidna(fixed=cbind(height,diameter)~Trait+Trait:Prov,
                 random=~us(Trait):nrm(Treeid),
                 residual=~units:us(Trait),
                 predict='Prov',
                 mulT=T,
                 es0.file='Provenance.es0'
                 )
```

Model comparison:

```
HDt1=update(HDt.esr,random=~diag(Trait):nrm(Treeid))
HDt2=update(HDt1,residual=~units:diag(Trait))

model.comp(HDt.esr,HDt1,HDt2,LRT=TRUE)
```

batch analyse of multiple trait:

```
bsm2<-echidna(trait=~height+diameter+volume,
                 fixed=~Trait+Trait:Prov,
                 random=~us(Trait):nrm(Treeid),
                 residual=~units:us(Trait),
                 batch=T,mulT=T,mulN=2,
                 predict='Prov',
                 es0.file='Provenance.es0'
                 )
Var(bsm2)
```

4.3 case three– spatial analyse

get .es0 file:

```
get.es0.file(dat.file='barley.csv')
get.es0.file(es.file='barley.es')
```

run different model:

```
msp.esr<-echidna(fixed=yield~1+Variety,
                 residual=c(R1~units,
                            R2~ar1(Row):id(Column),
                            R3~ar1(Row):ar1(Column)),
                 predict='Variety',
                 batch.R=T,
                 es0.file='barley.es0')
```

check results:

```
msp1<-b2s(msp.esr)

R1<-msp1$R1
R2<-msp1$R2
R3<-msp1$R3
```

```
model.comp(R1,R2,R3)
```

4.4 case four– genomic blup

generate genomic relation matrix:

```
G.marker<-read.csv(file="G.marker.csv",header=TRUE)
GOF<-GenomicRel( G.marker,1, Gres=TRUE)

write.csv(GOF,file='GOF.grm',row.names=F,quote=F)
```

get .es0 file:

```
get.es0.file(dat.file='G.data.csv')
get.es0.file(es.file='G.data.es',pedS=1)
```

user should make sure the order of pedigree file, .grm file and data file in the .es0 file.

run different models:

```
Ablup<-echidna(fixed=t1~1+Site,
                   random=~ nrm(ID),
                   es0.file='G.data.es0')

Gblup.GOF<-update(Ablup, random=~ grm1(ID))
```

check results:

```
Var(Ablup)

Var(Gblup.GOF)
```

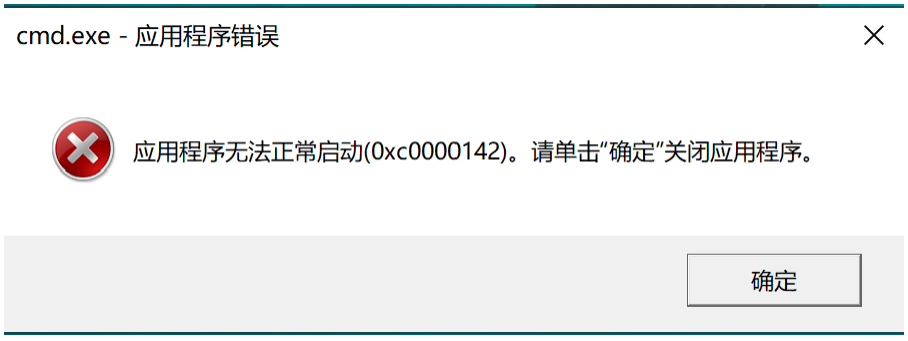
5 References:

1. Gilmour, A.R. (2021) Echidna Mixed Model Software www.EchidnaMMS.org

2. Zhang WH, Wei RY, Liu Y, Lin YZ. AFEchidna is a R package for genetic evaluation of plant and animal breeding datasets. BioRxiv. DOI: 10.1101/2021.06.24.449740.

6 Error message

Sometimes, there would show a window of error message as following:



Solving method: user need restart the computer.

7 appending .es0 files

Rules for .es0 files

.es0 file is changed from .es0 file without linear model parts, and some qualifiers are removed with #. User should only make sure for two places: 1) data fields and 2) data files. In the data fields, user should classify all factor variables. If the names of factor variables are started by capital letter, get.es0.file() could classify them automatically. In the data files, user should make sure whether the pedigree file or other genomic matrix files should be included.

Provenance.es0 file:

```
#!WORK 2  !REN  !ARG
TITLE: Provenance  #!DOPART $1
 # Treeid,Male,Female,Prov,Block,Plot,height,diameter,volume ...
 # 1001,0,170,10,1,3,12.5,19.6,0.1441 ...
Treeid     !P # 1001
Male       !I # 0
Female     !I # 170
Prov       !I # 10
Block      !I # 1
Plot       !I # 3
height         # 12.5
diameter       # 19.6
volume         # 0.1441
# Verify data fields are correctly classified as factors (!A !I !P *) or variates
Provenance.csv !SKIP 1
Provenance.csv !SKIP 1
```

(2) data fields: user should make sure whether the variables are correctly classified.

(1) data files: user should check the order of data files.

barley.es0 file:

```
#!WORK 2  !REN  !ARG
TITLE: barley  #!DOPART $1
 # Rep,RowBlk,ColBlk,Row,Column,Variety,yield ...
 # 1,1,1,1,1,1,1003 ...
Rep      !I # 1
RowBlk   !I # 1
ColBlk   !I # 1
Row      !I # 1
Column   !I # 1
Variety  !I # 1
yield       # 1003
# Verify data fields are correctly classified as factors (!A !I !P *) or variates
barley.csv !SKIP 1
```

G.data.es0 file:

```
#!WORK 2  !REN  !ARG
TITLE: G.data  #!DOPART $1
 # "ID","Female","Male","Year","Site","t1","t2" ...
 # "26","1","12","2001","6",NA,NA ...
ID        !P # 26
Female    !I # 1
Male      !I # 12
Year      !I # 2001
Site      !I # 6
t1        # *
t2        # *
# Verify data fields are correctly classified as factors (!A !I !P *) or variates

G.ped.csv !SKIP 1
GOF.grm
GD.grm
GOF.giv
G.data.csv !SKIP 1
```