Simple manual of AFEchidna

Version: V1.68
2023-01-05

1 Description

This package AFEchidna adds some R functions for Echidna v-1.68. AFEchidna builds on the Echidna software. AFEchidna is for non-commercial academic use. Echidna is targeted for use in animal and plant breeding contexts by Arthur Gilmour. The primary software of Echidna could be downloaded from website (https://www.echidnamms.org/). Echidna is free and a powerful tool for mixed models. AFEchidna is developed to run Echidna in R and similar to asreml at some extent.

The latest version of Echidna is V-1.70 (https://www.echidnamms.org/downloads/). Updated: 2022-Nov-9th.

2 Installation

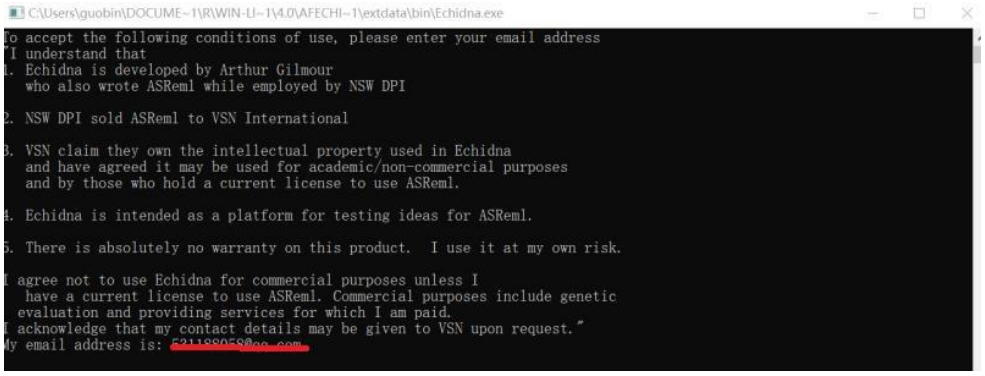
AFEchidna package has been uploaded online, thus user can install it online with the following codes, then run `AFEchidna::CheckPack()` in R to check whether the R dependent package is installed or not. If not, it will be installed automatically.

```
# from github
remotes::install_github('yzhlinscau/AFEchidna')

# from gitee
# install.packages('git2r')
remotes::install_git("https://gitee.com/yzhlinscau/AFEchidna.git")

AFEchidna::checkPack()  # check depended R packages
```

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

```
library(AFEchidna)

AFEchidna::read.example(package='AFEchidna',setpath = TRUE)
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'
                )
bsm %>% b2s %>% lapply(., Var)
```

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'
                )

bst2 %>% b2s %>% lapply(., Var)
```

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'
                 )


bsm2 %>% b2s %>% lapply(., Var)
```

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,option=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:

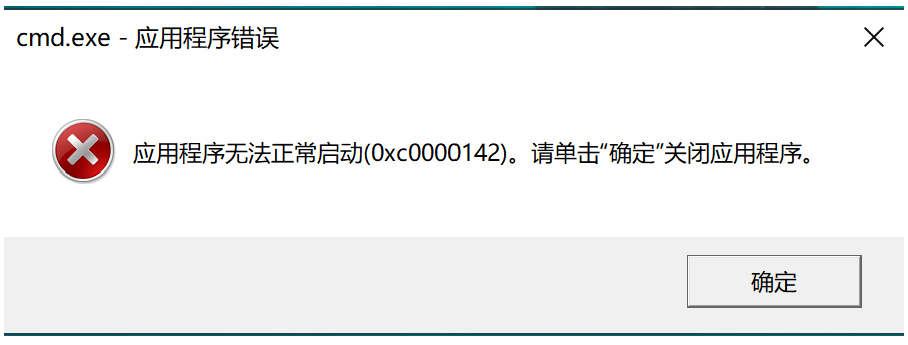
```
Ablup<-echidna(fixed=t1~1+Site,
                   random=~ nrm(ID),
                   es0.file='G.data.es0')

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

check results:

```
Var(Ablup)

Var(Gblup.GOF)
```

5 Error message

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



Solving method: user need restart the computer.

Another error often occurs, a simple case as following:

```
Running Echidna for analysis:  …
Error in if (aa == "LogL Converged") tt$Converge <- TRUE
else tt$Converge <- FALSE :
  missing value where TRUE/FALSE needed
```

Solving method: This error mainly results from the error of .es0 file, not the missing value. User should check the two parts of .es0 file: data fields and data files, whether they are correctly classified or ordered. By the way, when using GBLUP models, the G matrix might result in the same case because of singularity.

6 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.

7 appending .es0 files

Rules for .es0 files

The .es0 file is changed from .es 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.

For example, 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
```

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

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

Some es0 file cases are shown following:

(1) 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 as factors (!A !I !P *) or variates
barley.csv !SKIP 1
```

(2) 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 as factors (!A !I !P *) or variates
G.ped.csv !SKIP 1   # pedigree file, nrm(ID)
GOF.grm # G-matrix 1, grm1(ID)
GD.grm  # G-matrix 2, grm2(ID)
G05.grm # G-matrix 3, grm3(ID)
G.data.csv !SKIP 1 # data file
```

(3) 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 as factors (!A !I !P *) or variates
barley.csv !SKIP 1
```

(4) fm.es0 file:

```
#!WORK 2  !REN  !ARG
TITLE: fm  #!DOPART $1
 # TreeID,Spacing,Rep,Fam,Plot,dj,dm,wd,h1,h2,h3,h4,h5 ...
 # 80001,3,1,70048,1,0.334,0.405,0.358,29,130,239,420,630 ...
TreeID  !I      # 80001
Spacing !I      # 3
Rep !A          # 1
Fam !I          # 70048
Plot !I         # 1
```

```
dj        # 0.334
dm        # 0.405
wd        # 0.358
h1        # 29
h2        # 130
h3        # 239
h4        # 420
h5        # 630
# Verify data fields are as factors (!A !I !P *) or variates
fm.csv !SKIP 1 !filter Spacing !select  1
```

(5) MET1.es0 file:

```
#!WORK 2  !REN  !ARG
TITLE: MET1  #!DOPART $1
 # "Genotype","Loc","Loc2","Row","Col","Rep","Block","Plot","yield" ...
 # "175","3","CSuarez","1","1","1","1","1",17.52 ...
Genotype !P # 175
Loc      !I # 3
Loc2     !A # CSuarez
Row      !I # 1
Col      !I # 1
Rep      !I # 1
Block    !I # 1
Plot     !I # 1
yield       # 17.52
# Verify data fields are as factors (!A !I !P *) or variates
gped.csv !SKIP 1
G.giv  # giv1
H.giv  # giv2
MET1.csv !SKIP 1
```

(6) pig_data.es0 file:

```
#!WORK 2  !REN  !ARG
TITLE: pig_data  #!DOPART $1
 # pig sire   dam    year   sex    pen    weanage weanwt adg    weight loinarea ...
 # 133 2      1      2004   1      52     21      13.25  2      264    5.34 ...
pig   !P      # 133
sire  !I      # 2
dam   !P      # 1
year  !I      # 2004
sex   !I      # 1
pen   !I      # 52
weanage       # 21
weanwt        # 13.25
adg           # 2
weight        # 264
```

```
loinarea        # 5.34
# Verify data fields are as factors (!A !I !P *) or variates
pig_pedigree.txt !SKIP 1
pig_data.txt !SKIP 1
```