Demo

This is a step by step demo for analysis FeNO data using our U-HB method together with other unified or two-stage method for comparison.

1. Generate dateset for simulation

First, we need to build a dataset that mimic the real FeNO data. Here we use Scenario 1 for example (all NO parameters are affected by environment factor X) There are several key parameters here: number of subject in your study, number of manuvours, manucour flow rates, etc. We set them as same as we used in the paper but shrink the number of subjects to 20 for a quick run. You may expand to other sample size you want. We will build a balanced dataset and there is also an unbalanced dataset example attached at section #.

```
demoseed <- 2020
Ndat <- 20
source("/Users/wengjingying/Downloads/FeNO/eno/Demofunctions.R")
```

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

```
##
## Attaching package: 'nlme'
```

```
## The following object is masked from 'package:lme4':
##
##     lmList
```

```
# Total number of subjects in simulated datasets


enodata<-Obs_FB(alpha=alpha,beta=beta,X=X,Flow=flow,SD=sderror,NOcov=CovMat_chs_Caw)
save(enodata, file="data_S1.Rdata")
```

2. Model fitting

2.1 U-HB

When fitting U-HB model, we would like to start by using short iterations for a pilot run to check any existing bugs. So the initial burning and mcmc iterations (N.iterT/2) were set to be 120 and 120. We let it go through a 120 iteration update (addon.ite=120, Max_update=1) and pocess to the 600 iteration posterior sampling stage (n.final). Those interations are so short that we may expecting a large Rhat value indicate failure of converge. So we also set Rhat criteria (rhat) as large as 5. Those iteration numbers should be at least ten times as what we set here for convergence (Rhat<1.1).

After setting the parameters, we will record the starting time, run a burning and initial update, check Rhat first, if Rhat $<$ rhat, we go to posterior sampling stage, if not, we update the mcmc again to reach the Rhat criteria before use all the update numbers allowed and go to posteria sampling stage.

```
# set Gelman Rubin criteria
rhat=5
# set how many iterations needed for update MCMC
addon.iter=120
# maximum updates
Max_update=1
# iterarions needed to generate posterior distribution
n.final=600
```

```
# starting iteration numbers (1/3 will be burning)
N.iterT=240
# measurement error
sigma2=1e-3
```

2.2 TS-NLS

NLS is the

```
logenodata<-melt(data.frame(id=c(1:Ndat),enodata$logeno),id="id")
logenodata<-logenodata[order(logenodata$id,decreasing = FALSE),]
datOut <- data.frame(id=rep(c(1:Ndat),each=length(flow)),eno=NA,logeno=NA,flow=rep(flow,Ndat),X=rep(eno
datOut$logeno <- logenodata$value
datOut$eno    <-exp(datOut$logeno)
datOut$invFlow<-1/datOut$flow
datOut$enoflow<-datOut$flow*datOut$eno

dat <- datOut
X <- enodata$X

starttime=proc.time()[3]
# prelminary function for NLS 2-stage and for NLME functions

# fit for TS-NLS
fitnls <- nlsListStart(dat)
# summary how many subjects' data failed to fit NLS
nls_NA<-sum(is.na(coef(fitnls)[,1]))


# generate NLS result
NLSout <- tryCatch(nls_FINAL(fitnls,X), error = function(x) {
    print("Error with nls");
    warnings();
    data.frame(    est=rep(NA,6),
                   lb=rep(NA,6),
                   ub=rep(NA,6),
                   se=rep(NA,6),
                   p=rep(NA,6)
    )
  }
  )

elapsed=proc.time()[3]-starttime
colnames(NLSout)<-paste("NLS",c("est","lb","ub","se","p"),sep="_")
save(elapsed,NLSout,nls_NA, file="NLS_S1.Rdata")
```

2.3 TS-HMA

```
# HMA function

starttime=proc.time()[3]
```

```
HMAout <- tryCatch(HMA_FINAL(dat,X), error = function(x) {
    print("Error with HMA");
    warnings();
    data.frame(    est=rep(NA,6),
                   lb=rep(NA,6),
                   ub=rep(NA,6),
                   se=rep(NA,6),
                   p=rep(NA,6)
    )
  }
)
elapsed=proc.time()[3]-starttime
save(elapsed,HMAout, file="HMA_S1.Rdata")
```

2.4 TS-NLME and U-NLME

```
starttime=proc.time()[3]



print("nlmeSepStart")
fitD.nlme <- tryCatch(nlme(nlsList(logeno ~
                          log(exp(logCaw) + (Ca-exp(logCaw))*exp(-exp(logDaw)/flow))|id,
                          data=dat, start=list(Ca = 1.5, logCaw =4 , logDaw = 3),
                          control=list(tolerance=0.001)),
                          random= pdDiag(Ca+logCaw+logDaw ~ 1),
                          verbose=FALSE,
                          control=list(tolerance = 0.1,maxIter=100,pnlsTol=1e-4,msVerbose=TRUE)),
                      error = function(x) {
                                print("Error with nlmeSepstart");
                                nlmeSepout<-data.frame(    est=rep(NA,6),
                                                           lb=rep(NA,6),
                                                           ub=rep(NA,6),
                                                           se=rep(NA,6),
                                                           p=rep(NA,6)
                                )
                              }
    )


print("nlmeSepStart_u")
fitU.nlme <- tryCatch(update(fitD.nlme,random=pdLogChol(Ca+logCaw+logDaw ~ 1),
                             verbose=FALSE), error = function(x) {
      print("Error with nlmeSimStart");
      nlmeSimout<-data.frame(    est=rep(NA,6),
                                 lb=rep(NA,6),
                                 ub=rep(NA,6),
                                 se=rep(NA,6),
                                 p=rep(NA,6)
      )
    }
    )
```

```
# out put TS-NLME results
nlmeSepout <- tryCatch(nlme_sep_FINAL(fitU.nlme), error = function(x) {
      print("Error with nlmeSep");
      nlmeSepout<-data.frame(est=rep(NA,6),
                             lb=rep(NA,6),
                             ub=rep(NA,6),
                             se=rep(NA,6),
                             p=rep(NA,6)
      )
    }
    )

# output U-NLME results
nlmeSimout <- tryCatch(nlme_sim_FINAL(fitU.nlme), error = function(x) {
      print("Error with nlmeSim");
      nlmeSimout<-data.frame(    est=rep(NA,6),
                                 lb=rep(NA,6),
                                 ub=rep(NA,6),
                                 se=rep(NA,6),
                                 p=rep(NA,6)
      )
    }
    )


save(nlmeSepout,nlmeSimout, file="NLME_S1.Rdata")
```