ICE IVIVE workflow

Description

The workflow allows the flexibility to select from three different rat and human PK models: a 1 compartment model that incorporates Monte Carlo simulation to simulate the population variance (1C), a 3 compartment model leveraging the EPA's httk package (Solve_3comp), and a pbpk model that is tailored for compounds with additional glucuronidation(3CGlu). The workflow is to predict the daily equivalent administered dose (EAD, mg/kg/day) that would lead to steady state blood concentration equivalent to the bioactive concentration from in vitro assays and compared to the predicted lowest effective levels (LELs) of in vivo assays, which is user provided

Load libraries

```r
# libraries only needed for glucuronidation models ("3CGlu", "Solve_3comp")
library(plyr) # splitting, combining and applying data
library(deSolve) # Solves for initial value problems of differential equations
library(tidyr) # helps tidy data easily
library(ggplot2) # for creating elegant complex plots
library(scales) # scaling functions for visualizing
library(foreach) # for copying functions and libraries to each cluster
library(doParallel) # for parallelization
```

```
## Loading required package: iterators
```

```
## Loading required package: parallel
```

Input variables

There are several input variables needed to run the code. Some variables are model specific, as detailed.

```r
f0 <- "ChemicalData_rnotebook.txt" # chemicals dats from ICE, include CASRN field as identifier
f1 <- "invitroData_xc.txt" # invitro data from ICE, includes CASRN field as identifier then acc/ac50 va
species <- "human" # human or rat
modelType <- "1C" # "1C", "3CGlu", "Solve_3comp"

output_file <- "outtest2.txt"
```

For the 1 compartment model, values are needed to parameterize the Monte Carlo simulation

```
nsamples <- 300 # user-provided value for the mc simulations, any number between 10 - 10,000
```

For the pbpk models some addtional parameters can be modified

```
route <- "oral" # oral or iv, only needed for pbpk models ("3CGlu", "Solve_3comp")
interv <- 24 #dosing interval, hours, only needed for glucuronidation models ("3CGlu", "Solve_3comp")
ndays <- 3 #number of days dosin is done, only needed for glucuronidation models ("3CGlu", "Solve_3comp
ncores <- 4 #this is specific for the "3CGlu" to conduct parallelization
```

Load functions

```
#All required R scripts and input files should be in the working directory
source("steadyState.R") # required for 1C model
source("glu_MaxConc.R") # required for "3CGlu"
source("CalcEAD.R") # required for 1C and 3CGlu
source("EADboxplot.R") # required for plotting
```

Load data

```
chemical <- read.table(f0, sep = "\t", header = TRUE, quote = "", stringsAsFactors = FALSE) #Modify the
chemical[1:2,]
```

```
##      CASRN        DTXSID    ChemicalName Species     LogP      Clint         fu
## 1 104-40-5 DTXSID5033836   4-Nonylphenol   human 5.686413 -0.2171391 0.01150152
## 2 104-43-8 DTXSID1022508 4-Dodecylphenol   human 7.962457  1.0480756 0.37827307
##   pka_Donor pka_Accept        HL       MW    pkidney     pliver      pbody
## 1  10.85302         NA -4.975969 220.1827   28835.35   36223.35   15512.89
## 2  10.85302         NA -5.230207 262.2297 5395359.23 6771888.51 2902037.01
##      Kgut2pu
## 1   19534.34
## 2 3663854.82
```

```
invitro <- read.table(f1, sep = "\t", header = TRUE)
invitro[1:2,]
```

```
##      CASRN      ACC1     ACC2     ACC3     ACC4     ACC5 ACC6      ACC7
## 1 104-40-5        NA 4.149879 1.460656       NA       NA   NA 13.616425
## 2 104-43-8 0.1505835 0.059900 0.023200 52.10369 6.247932   NA  1.938221
##        ACC8      ACC9     ACC10     ACC11     ACC12     ACC13     ACC14
## 1 15.472105 7.7682923 6.3014026 9.2682982 7.5393300        NA        NA
## 2  1.956001 0.7179356 0.9955551 0.4595619 0.5517946 0.2352066 0.2136595
##      ACC15     ACC16
## 1       NA 8.3742888
## 2 2.851906 0.6309691
```

Preparing data

Minor prep work is done on the data that comes from ICE Partitioning coefficients are only needed for the 3CGlu model as they are not calculated as part of the workflow here. These can be obtained by running the "predict_partitioning_schmitt" function from the httk package.

```
## input data assigned as chemical, column names should be labeled correctly
## Following columns are required for glucuronidation models, 1C does not require partitioning coeffici
## Funbound. should be fu, Kkidney2pu should be pkidney, Kliver2pu should be pliver, Krest2pu should be
#colnames(chemical) <- gsub("Funbound.*", "fu", colnames(chemical))

#Converting units for internsic clearance values
chemical$Clint <- 10^chemical$Clint #this moves from log10 ul/ml/10^6 cells to just ul/ml/10^6 cells

#setting constants:
#dose = 1 #current calcualtions assume 1mg/kg/day

if (is.null(species)) {
  species <- "Human"
}
if (is.null(route)) {
  iv.dose <- TRUE
}
if (is.null(nsamples)) {
  nsamples <- 1000
}
if(!exists("dose") || is.null(dose)){
  dose <- 1
}
if(!exists("ConcentrationUnit") || is.null(ConcentrationUnit)){
  ConcentrationUnit <- "uM"
}
if(!exists("ncores") || is.null(ncores)){
  ncores<-detectCores() - 1
}
```

If modelType given by the user is "1C"

```
if (modelType == "1C") {    ### modelType needs to be selected as 1C in order for this step to work

  chemInput <- chemical[,c("CASRN", "ChemicalName", "Clint", "fu", "MW")] ## subsetting chemical data f

  CSS <- steadyState(inputData = chemInput, nsamples = nsamples, species = species, ConcentrationUnit =

  EAD.out50 <- CalcEAD(Css = CSS[,c("CASRN", "50%", "fu")], inVitro = invitro, adj.fu = "fu") ## CalcEA
  colnames(EAD.out50) <- gsub("EAD","EAD.50", colnames(EAD.out50))
  EAD.out95 <- CalcEAD(Css = CSS[,c("CASRN", "95%", "fu")], inVitro = invitro, adj.fu = "fu")
  colnames(EAD.out95) <- gsub("EAD","EAD.95", colnames(EAD.out95))
  EAD.out <- join_all(list(EAD.out50,EAD.out95))
  EAD.out <- EAD.out[,setdiff(colnames(EAD.out), c("adj.fu", "fu", "adj.arm","arm"))] # remove the colu
```

```
 #creating output file, should have EAD values and the parameters for calculating the circulating conce

  CSS2 <- as.data.frame(CSS, stringsAsFactors = FALSE); colnames(CSS2) <- gsub("50%", "Css, 50%ile", col
  CSS2$Species<-species;CSS2$Model<-"1-compartment"; CSS2$"dose, mg/kg"<-dose; CSS2$"nSimulations" <- ns

   CSS2<-CSS2[,c("CASRN", "ChemicalName", "Css, 50%ile","Css, 95%ile", "Css_Unit", "Species","Model","d
                "Clint", "fu", "MW")]

  ssEAD.out <- EAD.out[,c("CASRN", setdiff(colnames(EAD.out), c(colnames(invitro), "50%","95%")))]
  outputData <- join_all(list(CSS2,ssEAD.out))

  write.table(outputData, file = output_file, sep = '\t', col.names = TRUE, row.names = FALSE, quote = 
  EADplot <- EADboxplot(EAD.out = outputData,label="EAD.50", species = species, route = route, modelTyp
}
```

```
## Joining by: CASRN, fu, adj.fu, adj.arm, ACC1, ACC2, ACC3, ACC4, ACC5, ACC6, ACC7, ACC8, ACC9, ACC10,
```

```
## Joining by: CASRN
```

```
## 
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:plyr':
## 
##     arrange, count, desc, failwith, id, mutate, rename, summarise,
##     summarize
```

```
## The following objects are masked from 'package:stats':
## 
##     filter, lag
```

```
## The following objects are masked from 'package:base':
## 
##     intersect, setdiff, setequal, union
```

```
## Warning: Removed 36 rows containing non-finite values (stat_boxplot).
```

If modelType given by the user is “3CGlu”

Note that this model requires some addtional physioloigical properties

```
if (modelType == "3CGlu") { ## modelType needs to be selected as 3CGlu in order for this step to work
  #Developing table for physical data
  physParam <- data.frame(species = c("human", "rat"), bw = c(70, 0.25), QC.mLmin = c(5600, 74), Qliver
  #values for organ volumes from Davies et al 1993, table 2
  #these would ideally be supplied from user, defaults exist for the glucoronidated compound
  t1 <- data.frame(CASRN = chemical$CASRN, vmaxliver = NA, kmliver = NA,kuptakeC = NA, kGIingC = NA, km
 #this would be data coming to ICE from OPERA; partitioning coeff could come from within
  chemParam <- as.data.frame(merge(chemical, t1), stringsAsFactors = FALSE)
  cl<-ncores; registerDoParallel(cl)
  Cmax.glu<- foreach(i = 1:nrow(chemParam),.combine = rbind) %dopar% {glu_MaxConc(physParam, chemParam[

  #Calculating the EADs
  EAD.out_max <- CalcEAD(Css = Cmax.glu[,c("CASRN", "cmax")], inVitro = invitro)
   Cmax.glu$Species<-species;Cmax.glu$Model<-"3C Gluc"; Cmax.glu$Dose<-dose;

  ssEAD.out <- EAD.out_max[,c("CASRN", setdiff(colnames(EAD.out_max), c(colnames(invitro), "cmax")))]
  Cmax.glu$days <- ndays
  colnames(Cmax.glu)<-gsub("cmax", "Cmax", colnames(Cmax.glu)); colnames(Cmax.glu)<-gsub("Cmax_unit", "
  Cmax.glu <- Cmax.glu[,c("CASRN", "ChemicalName", "Cmax", "Cmax_Unit", "Species", "Model", "dose, mg/k

  outputData <- join_all(list(Cmax.glu,ssEAD.out))
  write.table(outputData, file = output_file, sep = '\t', col.names = TRUE, row.names = FALSE, quote = 
  EADplot <- EADboxplot(EAD.out = outputData,label="EAD", species = species, route = route, modelType =
}
```

If modelType given by the user is “Solve_3comp”

```
if (modelType == "Solve_3comp") { ## modelType needs to be selected as Solve_3comp in order for this st
  library(httk)
  #preprocessing variables:
  if (tolower(species) == "rat") {
    species_1 <- "Rat"
  }
  if (tolower(species) == "human") {
    species_1 <- "Human"
  }
  # Processing the chem information
  # ##will need to verify the labels on the opera data
  # chems <- as.data.frame(read.table(f0, header = TRUE, sep = "\t", stringsAsFactors = FALSE, quote ="
  # inVitro <- read.table(f1, sep = "\t", header = TRUE)
  options(stringsAsFactors = FALSE)
  chem3cInput <- chemical[,setdiff(colnames(chemical), c("pkidney","pliver","pbody","Kgut2pu"))]#removi
  chem3cInput$logPwa <- 1*chem3cInput$HL #this gets the water octanal coeff
  #chemical$ClintNorm<-10^chemical$CLint #to get into correct units   #this is addressed in preprocessi

  #add chemical info to the table. Using variable coming from ICE

chem.physical_and_invitro.data <- add_chemtable(chem3cInput, current.table = chem.physical_and_invitro.

  if (route == "oral") {
    iv.dose = FALSE
  } else {iv.dose = TRUE}

  EDcmax3day.3comp <- NULL
  cmax3dayivall <- NULL

  for (this.cas in chem3cInput[,'CASRN']) {
    outiv <- solve_3comp(chem.cas = this.cas, parameters = NULL, doses.per.day = 24/interv, days = nday
    concMax <- max(outiv[,'Cplasma'])
```

```r
    cmax3dayiv <- as.data.frame(cbind(this.cas,concMax, ConcentrationUnit))
    cmax3dayivall <- rbind(cmax3dayivall,cmax3dayiv)
    cmax3dayivall$concMax <- as.numeric(cmax3dayivall$concMax)
  }
  Cmax <- merge(chem3cInput, cmax3dayivall, by.x = "CASRN", by.y = "this.cas")
  #outcmax3dayiv <- ThreeC_httk(chemical, species = species_1, iv.dose)

  #Calculating the EADs
  EAD.out_max <- CalcEAD(Css = Cmax[,c("CASRN", "concMax")], inVitro = invitro)
  ssEAD.out <- EAD.out_max[,c("CASRN", setdiff(colnames(EAD.out_max), c(colnames(invitro), "Cmax")))]
  names(ssEAD.out) <- gsub("concMax", "Cmax", names(ssEAD.out) )

  names(Cmax) <- gsub("concMax", "Cmax", gsub("ConcentrationUnit", "Cmax_Unit", names(Cmax) ) )
  Cmax$route <- route
  Cmax$interv <- interv
  Cmax$days <- ndays
  Cmax$Species<-species; Cmax$Model<-"Solve_3comp"; Cmax$"dose, mg/kg"<-dose;
  Cmax <- Cmax[, c("CASRN", "ChemicalName", "Cmax", "Cmax_Unit", "Species", "Model", "dose, mg/kg","inte
  outputData <- join_all(list(Cmax,ssEAD.out))

  write.table(outputData, file = output_file, sep = '\t', col.names = TRUE, row.names = FALSE, quote =
  EADplot <- EADboxplot(EAD.out = outputData, label="EAD", species = species, route = route, modelType =
}
```

```r
sessionInfo()
```

```
## R version 3.6.3 (2020-02-29)
## Platform: x86_64-w64-mingw32/x64 (64-bit)
## Running under: Windows 10 x64 (build 17763)
##
## Matrix products: default
##
## locale:
## [1] LC_COLLATE=English_United States.1252
## [2] LC_CTYPE=English_United States.1252
## [3] LC_MONETARY=English_United States.1252
## [4] LC_NUMERIC=C
## [5] LC_TIME=English_United States.1252
##
## attached base packages:
## [1] parallel  stats     graphics  grDevices utils     datasets  methods
## [8] base
##
## other attached packages:
## [1] dplyr_1.0.0       doParallel_1.0.15 iterators_1.0.12  foreach_1.5.0
## [5] scales_1.1.1      ggplot2_3.3.2     tidyr_1.1.0       deSolve_1.28
## [9] plyr_1.8.6
##
## loaded via a namespace (and not attached):
##  [1] Rcpp_1.0.4.6     pillar_1.4.4     compiler_3.6.3   tools_3.6.3
##  [5] digest_0.6.25    evaluate_0.14    lifecycle_0.2.0  tibble_3.0.1
##  [9] gtable_0.3.0     pkgconfig_2.0.3  rlang_0.4.6      yaml_2.2.1
## [13] xfun_0.15        withr_2.2.0      stringr_1.4.0    knitr_1.29
## [17] generics_0.0.2   vctrs_0.3.1      grid_3.6.3       tidyselect_1.1.0
## [21] glue_1.4.1       R6_2.4.1         rmarkdown_2.3    purrr_0.3.4
## [25] farver_2.0.3     magrittr_1.5     codetools_0.2-16 ellipsis_0.3.1
## [29] htmltools_0.5.0  colorspace_1.4-1 stringi_1.4.6    munsell_0.5.0
## [33] crayon_1.3.4
```