Package ‘plethem’

January 2, 2019

Title  Population Life Course Exposure to Health Effects Modeling
Version  0.1.7
Description  Functions, data and user interfaces for performing Physiologically Based Pharmacokinetic (PBPK) Modeling, In-vitro to In-vivo Extrapolation (IVIVE) and exposure estimation. Also contains user interfaces to run models from the 'httk' package. Taken together these provide an easy to use and powerful modeling tool that can be used for all steps along the source-to-outcome continuum. All the analysis tools in the package can only be run as shiny apps. Check vignettes and package help for more information. More information on PBPK modeling can be found in the book 'Physiologically Based Pharmacokinetic Modeling: Science and Applications' by Reddy et al <doi:10.1002/0471478768>.
Depends  R (>= 3.4.0)
License  MIT + file LICENSE
Encoding  UTF-8
LazyData  true
Imports  shiny, ggplot2, shinyBS, shinyWidgets, miniUI, shinydashboard, rstudioapi, shinythemes, data.table, rhandsontable, deSolve, truncdist, reshape2, dplyr, readxl, gdata, devtools, RSQLite, formatR, plotly, magrittr, sqldf, utils, stats, DT, shinyjs, V8, httk
Suggests  testthat, rmarkdown, knitr
RoxygenNote  6.1.1
VignetteBuilder  knitr
NeedsCompilation  yes
Author  Salil Pendse [aut, cre], ScitoVation, LLC [cph]
Maintainer  Salil Pendse <spendse@scitovation.com>
Repository  CRAN
Date/Publication  2019-01-02 15:50:03 UTC
## R topics documented:

- `addChemsToHTTK` .................................................. 3
- `addDataSet` ...................................................... 4
- `addDataSetUI` .................................................... 4
- `calcMPCPPGL` ..................................................... 5
- `calculatePartitionCoefficients` ............................... 5
- `calc_cyp2b6` .................................................... 6
- `clearProjectDb` .................................................. 6
- `createUserDb` ................................................... 6
- `doIVIVE` .......................................................... 7
- `externDbSelect` .................................................. 7
- `getAllCypData` ................................................... 8
- `getAllParamValuesForModel` .................................. 8
- `getAllSetChoices` ............................................... 9
- `getAllVariabilityValuesForModel` ........................... 9
- `getDbConn` ....................................................... 10
- `getLifecourseBodyHeight` ...................................... 10
- `getLifecourseBodyHeightMale` ............................... 11
- `getLifecourseBodyWeight` ...................................... 11
- `getLifecourseBodyWeightFemale` ............................. 12
- `getLifecourseBodyWeightMale` ............................... 12
- `getLifecourseCardiacOutput` .................................. 13
- `getLifecourseGlomerularFiltrationRate` ..................... 13
- `getLifecourseLungDeadSpace` ................................ 14
- `getLifecourseTidalVolume` .................................... 14
- `getLifecourseTissuePerfusion` ............................... 15
- `getLifecourseTissueVolumes` .................................. 15
- `getLifecourseUrineProductionRate` ......................... 16
- `getLifecourseVentilationRate` ............................... 16
- `getMetabData` ................................................... 17
- `getNextID` ....................................................... 17
- `getObservationSetChoices` .................................... 18
- `getParameterSet` ............................................... 18
- `getProjectChemicalList` ...................................... 19
- `getVariabilitySetChoices` ..................................... 19
- `httkCalcOralEqDose` .......................................... 20
- `httkParameterPBTK` ............................................ 20
- `HT_IVIVE` ....................................................... 21
- `HT_IVIVEUI` ..................................................... 21
- `importBatchExposure` ......................................... 22
- `importBatchExposureUI` ...................................... 22
- `importHTTKData` ............................................... 23
- `importHTTKDataUI` ............................................. 23
- `importParameterSet` .......................................... 24
- `importParameterSetUI` ....................................... 24
- `importSEEMData` .............................................. 25
- `importSEEMDataUI` ............................................ 25
addChemsToHTTK

Function that adds chemical to htk package chem list.

Description

internal function that interfaces with htk package
addDataSetUI

Usage
addChemsToHTTK()

Examples

## Not run:
addChemsToHTTK()

## End(Not run)

---

**addDataSet**

**Server side function for the UI used to add external observation datasets to projects**

---

**Description**

The server function for the add dataset module used in PLETHEM. This function interacts with the server function of the PLETHEM model. It saves the dataset imported as a .RDS file in the project folder. It also adds the dataset reference to the project database for further use.

**Usage**

addDataSet(input, output, session, data_type)

**Arguments**

- **input**
  - the input object from the add dataset module UI
- **output**
  - the output object from the add dataset module UI
- **session**
  - the shiny session information where the add dataset in called currently
- **data_type**
  - the data_type returned by the module.

---

**addDataSetUI**

**UI for adding external observation datasets to projects**

---

**Description**

The UI function for the add dataset module used in PLETHEM. This function creates the add dataset dialog box that is triggered by the modeling interface. This is a module function and the user will never need to call it.

**Usage**

addDataSetUI(namespace, data_type)
calcMPCPPGL

**Arguments**
- namespace: the namespace for this module
- data_type: the type of data to be uploaded based on where the UI is called from

**Description**
Calculates the Microsomal Protein per Gram liver (MPPGL) and Cytosolic Protein Per Gram Liver (CPPGL) for humans based on the age of the person in years. The equations for MPPGL and CPPGL were developed internally at Scitovation.

**Usage**
calcMPCPPGL(age)

**Arguments**
- age: age of the human in years

**Value**
list containing the "MPPGL" and "CPPGL" values for the

calculatePartitionCoefficients

**Calculate Partition Coefficient**

**Description**
This function calculates the partition coefficients based on the QSAR model selected for the given tissues. Currently only one QSAR model is supported by PLETHEM.

**Usage**
calculatePartitionCoefficients(selected_qsar = "one",
chem_params = NULL, tissue_list, selected_org = "human")

**Arguments**
- selected_qsar: QSAR model to use for estimating partition coefficient
- chem_params: A named list of chemical params. The list should contain the minimal number of parameters needed to run the QSAR model selected
- tissue_list: List of tissues for which the partition coefficients need to be calculated. See vignette on QSAR based parameter estimation of more details
- selected_org: Organism. Either "human" or "rat"
calc_cyp2b6  
*Calculate cyp2b6 fraction*

**Description**
get expression for cyp2b6 as a fraction of adult

**Usage**
calc_cyp2b6(age)

**Arguments**
age  
age of the human in years

--

clearProjectDb  
*Clear Project Db*

**Description**
This function clears the project Db. It is called internally when a new project is created. It is also used by developers to make a clean project db

**Usage**
clearProjectDb()

--

createUserDb  
*Creates a new userDb based on the empty database in the package*

**Description**
The function allows the users to create a new empty user database file that is needed to run plethem.

**Usage**
createUserDb(path = NULL)

**Arguments**
path  
path to where the user database needs to be stored. Make sure you have write permission to this folder. If no path is provided, it launches a folder select dialog.
### doIVIVE

**run IVIVE gadget and save files as needed.**

#### Examples

```r
## Not run:
createUserDb()
createUserDb("C:/Users/Documents/")

## End(Not run)
```

#### Description

Run IVIVE gadget and save CSV files needed to create metabolism sets in the PBPK model

#### Usage

```r
doIVIVE()
```

#### Value

Values returned by the IVIVE gadget

#### Examples

```r
## Not run:
doIVIVE()

## End(Not run)
```

### externDbSelect

**Runs all select queries an arbitrary database**

#### Description

The function runs the select queries issued to the user db and returns the dataframe the path to user database is stored in main plethem database and is selected from there

#### Usage

```r
externDbSelect(query, db_path)
```

#### Arguments

<table>
<thead>
<tr>
<th>Argument</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>query</td>
<td>A valid SQL Query</td>
</tr>
<tr>
<td>db_path</td>
<td>A valid path</td>
</tr>
</tbody>
</table>
getAllCypData

get data for human cyp fraction by age

description

The function returns the expression of cyps at each age as a fraction of the adult (25 years). The number needs to be multiplied by the cyp abundance information in the database to get the nominal expression value for the cyp.

Usage

getAllCypData(age)

Arguments

age the age of the person in years

Value

dataframe containing the fractional expression for all the cyps

getAllParamValuesForModel

Gets all the parameter values for the model. This function should not be used by the model

description

Get all the parameter values that are required for the model to run. The values are obtained from the Project database. Only those values that are used in the model as determined by the master database are returned by the function.

Usage

getAllParamValuesForModel(simid, model)

Arguments

simid Integer The id for simulation selected to run
model Character The string identifying the model to be run

Value

list List that can be passed to the solver as model params
getAllSetChoices

get all set names for a given parameter set

Description
This function returns all the sets of a given set type from the current project database. This is used internally to update drop downs or to get simulation choices.

Usage
getAllSetChoices(set_type = "physio")

Arguments
set_type The type of set can be "physio", "chem", "expo", "metab" or "sim"

Value
named list of set names

ggetAllVariabilityValuesForModel

Gets all the variability values for the model. This data returned by the function is not meant to be understandable by the user.

Description
Get all the variability values required for creating parameter sets for montecarlo analysis. The values are obtained from the Project database.

Usage
ggetAllVariabilityValuesForModel(simid, params, mc_num)

Arguments
simid Integer The id for simulation selected to run
params list of model parameters
mc_num number of montecarlo runs

Value
matrix of parameters that will be used for individual montecarlo runs
getLifecourseBodyHeight

Description
The function is used to calculate the average body height for humans based on age and gender. The function uses life course equations developed by Scitovation.

Usage
getLifecourseBodyHeight(age, gender)

Arguments
age
age in years

gender
Either "M" for male or "F" for female

Value
Body Height
getLifecourseBodyHeightMale

Calculate body height of average human male using the lifecourse equation

Description

This function is not called directly by the user.

Usage

getcharacteristicsbodyheightmale(age)

Arguments

age age in years

Value

Body height

getcharacteristicsbodyweight

Get average body weight using the life course equation

Description

The function is used to calculate the average body weight in kgs for humans based on age and gender. The function uses life course equations developed by Scitovation.

Usage

getcharacteristicsbodyweight(age, gender)

Arguments

age age in years

gender Either "M" for male or "F" for female

Value

Body Weight in kgs
getLifecourseBodyWeightFemale

*Calculate body weight in kg of average human female using the life-course equation*

---

**Description**

This function is not called directly by the user. See getLifeCourseBodyWeight

**Usage**

getLifecourseBodyWeightFemale(age)

**Arguments**

age  
age in years

**Value**

Body weight in kg

---

getLifecourseBodyWeightMale

*Calculate body weight in kg of average human male using the life-course equation*

---

**Description**

This function is not called directly by the user. See getLifeCourseBodyWeight

**Usage**

getLifecourseBodyWeightMale(age)

**Arguments**

age  
age in years

**Value**

Body weight in kg
getLifecourseCardiacOutput

Get the cardiac output based on the life course equations

Description

Get the cardiac output for a specific age and gender. Uses either Bosgra or ScitoVation equations

Usage

getLifecourseCardiacOutput(age, gender, source = "sciv", qc_var = 0)

Arguments

age Age in years
gender Gender can be either "M" for male or "F" for female
source Source of the equations. Either from Bosgra et al. "bosgra" or from ScitoVation publications as "sciv"
qc_var variability for cardiac output. Used to account for exercise level. Can only be used when source = "sciv"

getLifecourseGlomerularFiltrationRate

Get the Glomerular Filtration Rate for a given age and gender

Description

Uses life course equation to calculate a GFR for a specific age and gender

Usage

getLifecourseGlomerularFiltrationRate(age, gender)

Arguments

age Age in years
gender Either "M" for Male or "F" for Female
getLifecourseLungDeadSpace

*Get the Lung Dead Space for a given age and gender*

**Description**

Uses life course equation to calculate the lung dead space for a specific age and gender

**Usage**

```
getLifecourseLungDeadSpace(age, gender)
```

**Arguments**

- **age**
  - Age in years
- **gender**
  - Either "M" for Male or "F" for Female

getLifecourseTidalVolume

*Get the tidal volume for a given age and gender*

**Description**

Uses life course equation to calculate the tidal volume for a specific age and gender

**Usage**

```
getLifecourseTidalVolume(age, gender, activity = "rest")
```

**Arguments**

- **age**
  - Age in years
- **gender**
  - Either "M" for Male or "F" for Female
- **activity**
  - Activity level. Can be "rest" or "light activity"
getLifecourseTissuePerfusion

Get perfusion for tissues provided on the basis of age and gender

Description
Get perfusion for tissues provided on the basis of age and gender

Usage
getLifecourseTissuePerfusion(age = 25, gender = "M",
    tissues = list(), source = "sciv")

Arguments
- **age**: Age of the organism in years
- **gender**: The gender of the organism "M" for Male or "F" for Female
- **tissues**: List of tissues for which blood perfusion is needed. The tissues can be any of the following liver, fat, bone, brain, gonad, heart, intestine, kidney, lung, pancreas, skin, spleen, stomach, thymus, muscle, remaining.
- **source**: source of the equation. defaults to "sciv" for scitovation

Value
list containing blood perfusion of tissues for the given age and gender.

getLifecourseTissueVolumes

Get volumes for tissues provided on the basis of age and gender

Description
Get volumes for tissues provided on the basis of age and gender

Usage
getLifecourseTissueVolumes(age = 25, gender = "M", perf_frct = 0.85,
    tissues = list())

Arguments
- **age**: Age of the organism in years
- **gender**: The gender of the organism "M" for Male or "F" for Female
- **perf_frct**: fraction of perfused tissue. default to 0.85
- **tissues**: List of tissues for which the volumes are needed. The tissues can be one of the following liver, blood, fat, bone, brain, gonad, heart, intestine, kidney, lung, pancreas, skin, spleen, stomach, thymus, remaining, muscle
getLifecourseVentilationRate

Value

list containing volumes for age, gender and tissues.

getLifecourseUrineProductionRate

Get the urine production rate for a given age and gender

Description

Uses life course equation to calculate a urine production rate for a specific age and gender

Usage

getLifecourseUrineProductionRate(age, gender)

Arguments

<table>
<thead>
<tr>
<th>age</th>
<th>Age in years</th>
</tr>
</thead>
<tbody>
<tr>
<td>gender</td>
<td>Either &quot;M&quot; for Male for &quot;F&quot; for Female</td>
</tr>
</tbody>
</table>

getLifecourseVentilationRate

Get the ventilation rate for a given age and gender

Description

Uses life course equation to calculate a ventilation rate for a specific age and gender

Usage

getLifecourseVentilationRate(age, gender, activity = "rest", source = "sciv")

Arguments

<table>
<thead>
<tr>
<th>age</th>
<th>Age in years</th>
</tr>
</thead>
<tbody>
<tr>
<td>gender</td>
<td>Either &quot;M&quot; for Male for &quot;F&quot; for Female</td>
</tr>
<tr>
<td>activity</td>
<td>Activity level. Can be &quot;rest&quot; or &quot;light exercise&quot;</td>
</tr>
<tr>
<td>source</td>
<td>source of equations either &quot;sciv&quot; or &quot;bosgra&quot;</td>
</tr>
</tbody>
</table>
getMetabData

Description

The function returns the relevant metabolism data if the simulation contains data from the metabolism set.

Usage

getMetabData(metabid, physioid, chemid, model)

Arguments

- metabid: The metabolism ID to get information from the metabolism tables.
- physioid: Physiological ID to get the age from the physiological table.
- chemid: Internal chemical ID.
- model: Model name.

Value

A list containing the metabolism values needed to run PBPK model or display simulation information.

getNextID

Description

Gets the next valid id for a given table. This function is used internally to decide what ID number should be used to save a parameter set.

Usage

getNextID(tble_name, db_path = "database/project.sqlite")

Arguments

- tble_name: Name of the table for which the ID is required.
- db_path: The location of the project database. This defaults to database/project.sqlite and is not expected to change.

Value

An integer ID of the next row in the table.
getObservationSetChoices

*Get all observation sets*

**Description**

Get all the sets associated with observation in a given project. Observations need to be handled differently from the other set types since they can themselves be of multiple types.

**Usage**

```
getObservationSetChoices(obs_type)
```

**Arguments**

- `obs_type` type of observation to return, can be "cl" or "conc" for clearance and concentration data

**Value**

named list of all sets of the `obs_type`

getParameterSet

*Get the values for parameters in a given set*

**Description**

Get all the parameter values for a given dataset and id.

**Usage**

```
getParameterSet(set_type = "physio", id = 1)
```

**Arguments**

- `set_type` Either "physio","chem"or "expo"
- `id` integer id for the required set
getProjectChemicalList

get the list of chemicals currently in the project database

Description

Gets the chemicals that are currently a part of the project. This list comes from the chemical table in the project database. This function returns the chemical properties needed by all the models within PLETHEM. Chemical information specific to the model currently used is requested through a different function. This function can be used directly by the user to get a list of chemicals in the current project.

Usage

getProjectChemicalList()

Value

Returns a chemical list containing the names(chem_name), CAS numbers(CAS), Molecular Weight(MW), KM and Fraction unbound Unbound in Plasma (FuPls) for all the chemicals in the project.

getVariabilitySetChoices

Get all variability sets

Description

Get all the variability datasets in a given projects. Variabilities need to be handled differently from the other set types since they can themselves be of multiple types.

Usage

getVariabilitySetChoices(var_type = "physio")

Arguments

var_type type of Variability set to return, can be "physio","chem" or "expo" or "conc"

Value

named list of all sets of the var_type
**httCalcOralEqDose**  
*Function that runs the httk oral equivalent dose gadget.*

**Description**

Launches the HTTK oral equivalent dose gadget. It servers as a UI wrapper around HTTK’s functions to calculate oral equivalent dose.

**Usage**

`httCalcOralEqDose()`

**Examples**

```r
## Not run:
httCalcOralEqDose()

## End(Not run)
```

---

**httParameterPBTK**  
*Call httk function for parameterizing PBPK models*

**Description**

Function that calls parameterize_pbpk function within the httk package

**Usage**

`httParameterPBTK(chem_name, species = "Human")`

**Arguments**

- `chem_name` name of the chemical for which to parameterize the model. Has to be in the httk database.
- `species` species for which to parameterize the model. Defaults to human

**Value**

list containing parameters for the PBPK model
**HT_IVIVE**

*server function of high throughput dosimetry*

---

**Description**

This function is needed internally by the package to handle the server functions related to adding compounds in the HT-IVIVE UI. It is never intended to be called by the user.

**Usage**

\[
\text{HT_IVIVE}(\text{input, output, session, vals = "", type = "}\text{"}, \\
\quad \text{chem_list = list(), idx = 0, row_selected = 0})
\]

**Arguments**

- **input**: input object from the data input UI
- **output**: output object from the data input UI
- **session**: session in which this module is called
- **vals**: values for clearance
- **type**: IVIVE type
- **chem_list**: List of imported chemicals in the project
- **idx**: index of the row
- **row_selected**: row selected for editing

---

**HT_IVIVEUI**

*Module for editing high throughput reverse dosimetry functions*

---

**Description**

The UI for defining HT-IVIVE parameters in the HT-IVIVE project. It is called by the HT-IVIVE server script when a new row is added or existing row is edited. It is never called directly by the user.

**Usage**

\[
\text{HT_IVIVEUI}(\text{namespace = "}\text{"})
\]

**Arguments**

- **namespace**: namespace for the module. This is unique and decided by the project server function
importBatchExposureUI

**importBatchExposure**  
*Server function for batch data module*

**Description**

Server function for import batch data module. This function should not be called by the user

**Usage**

importBatchExposure(input, output, session, expo_name_df)

**Arguments**

- **input**: input object for the UI
- **output**: input object to the UI
- **session**: session object for the module
- **expo_name_df**: dataframe containing variable names for exposure values

---

importBatchExposureUI  
*UI function for importing generic exposure data*

**Description**

UI function for importing generic exposure data into PLETHEM. The function should not be called by the user

**Usage**

importBatchExposureUI(namespace)

**Arguments**

- **namespace**: namespace for the module
importHTTKData  

*Server function for importing HTTK data*

**Description**

This is the server function for the importHTTKData module. This is not to be called by the user.

**Usage**

```r
importHTTKData(input, output, session)
```

**Arguments**

- **input**: input object from the UI
- **output**: output object with results
- **session**: session from which this module is called

importHTTKDataUI  

*UI for importing data from HTTK*

**Description**

UI function for the shiny module called by the rapidPBPK Exposure UI. This will never be called directly by the user.

**Usage**

```r
importHTTKDataUI(namespace)
```

**Arguments**

- **namespace**: namespace for the module
importParameterSetUI

importParameterSet  

Server for import parameter module

Description

server function for importing parameter sets from user databases. This function should never be called by the user

Usage

importParameterSet(input, output, session, set_type)

Arguments

input  input object from the UI
output  output object for the UI
session  session object for the server
set_type  type of data to be imported

importParameterSetUI  

UI for importing parameter sets

Description

UI for importing parameters from user or main databases

Usage

importParameterSetUI(namespace, set_type)

Arguments

namespace  namespace for this module
set_type  type of data to be imported; physiological, chemical or exposure
**importSEEMData**  

*Server function for seem data module*

---

**Description**

Server function for import seem data module. This function should not be called by the user

**Usage**

```python
importSEEMData(input, output, session, fpath, expo_name_df)
```

**Arguments**

- **input**: input object for the UI  
- **output**: input object to the UI  
- **session**: session object for the module  
- **fpath**: path to the SEEM database  
- **expo_name_df**: dataframe containing variable names for exposure values

---

**importSEEMDataUI**  

*UI for importing SEEM data.*

---

**Description**

This function is called by the pbpk model to import SEEM exposure estimates. Never called by the user

**Usage**

```python
importSEEMDataUI(namespace)
```

**Arguments**

- **namespace**: namespace for the module UI.
**importShedsData**  
*Server function for importing SHEDS data*

**Description**

Server function for importing SHEDS exposures estimates into PLETHEM. The function should not be called by the user.

**Usage**

```
importShedsData(input, output, session, path, expo_name_df)
```

**Arguments**

- `input`: input object for the UI
- `output`: input object to the UI
- `session`: session object for the module
- `path`: path where SHEDS results are stored
- `expo_name_df`: dataframe containing variable names for exposure values

---

**importShedsDataUI**  
*UI function for importing SHEDS data*

**Description**

UI function for importing SHEDS exposures estimates into PLETHEM. The function should not be called by the user.

**Usage**

```
importShedsDataUI(namespace)
```

**Arguments**

- `namespace`: namespace for the module
**interactiveHT**

*Launch HT-IVIVE interface*

**Description**

Used internally to launch the HT-IVIVE UI. HT-IVIVE does not use the project management system that PBPK models uses.

**Usage**

```r
interactiveHT(name = "")
```

**Arguments**

- `name` name of the model. Has to be "HT-IVIVE"

**Examples**

```r
## Not run:
interactiveHT("HT-IVIVE")

## End(Not run)
```

**iviveGadget**

*Command line function for running the IVIVE gadget*

**Description**

Launches the IVIVE gadget for estimating age dependent metabolism.

**Usage**

```r
iviveGadget(save_flag = F, base_path = NULL)
```

**Arguments**

- `save_flag` Boolean. Should the data generated be saved to a csv file. The name of the file can then be defined in the UI
- `base_path` if `save_flag` is True, must specify a base path where the result file will be
loadProject

Load the project from the project file located at the given path

Description

Loads the project data from the project file and then launches the shiny UI that corresponds to the analysis type that the project belongs to.

Usage

loadProject(file_path = "")

Arguments

file_path: path to the project file. If no path is provided, launches a select file dialog box for the user to select the path.

Examples

## Not run:
loadProject(file_path = "C:/Project/TestPBPK.Rdata")
loadProject()

## End(Not run)

loadTRA

run TRA gadget and save files as needed.

Description

Run TRA gadget and save CSV files needed to export TRA exposures for the PBPK model

Usage

loadTRA()

Examples

## Not run:
loadTRA()

## End(Not run)
**mainDbSelect**  
*Runs all select queries to the main database.*

**Description**

The function runs the select queries issued to the main db and returns the dataframe.

**Usage**

```python
cdef mainDbSelect(query, db_path = "database/plethemdb.sqlite")
```

**Arguments**

- **query**: A valid SQL Query
- **db_path**: The location of the project database. This defaults to `database/plethemdb.sqlite` and is not expected to change. This function will not be called by the user directly.

---

**mainDbUpdate**  
*Runs all update queries to the main database.*

**Description**

The function runs the update queries issued to the main db.

**Usage**

```python
cdef mainDbUpdate(query, db_path = "database/plethemdb.sqlite")
```

**Arguments**

- **query**: A valid SQL Query
- **db_path**: The location of the main database. This defaults to `database/plethemdb.sqlite` and is not expected to change. This function will not be called by the user directly.
newEditVariability  

**Server function for defining variability and uncertainty datasets**

**Description**

Server function for defining variability and uncertainty datasets in the rapidPBPK model. This should not be called by the user.

**Usage**

```python
newEditVariability(input, output, session, set_type, ops_type, var_params_list, set_id = 0)
```

**Arguments**

- **input**  
  input object from UI
- **output**  
  output object to the UI
- **session**  
  session object for this module
- **set_type**  
  type of set for which variability is defined
- **ops_type**  
  Operation requested. new variability or edit existing
- **var_params_list**  
  List of parameters for variability
- **set_id**  
  id for the variability set in the database

newEditVariabilityUI  

**UI function for defining variability and uncertainty datasets**

**Description**

UI function for defining variability and uncertainty datasets in the rapidPBPK model. This should not be called by the user.

**Usage**

```python
newEditVariabilityUI(namespace)
```

**Arguments**

- **namespace**  
  namespace for the module when it is called form the PBPK UI
newProject

Start a new PLETHEM project.

Description

A project consists of chemicals, organisms and datasets. The function asks the user for a location to save the project files on exit. It then launches the shiny user interface used to parameterize and run the model.

Usage

newProject(name = "new_project", type = "PBPK", model = "rapidPBPK", mode = "MC")

Arguments

- **name**: The name for the project
- **type**: The type of the model that the project is tied to
- **model**: The model to be used for the project
- **mode**: Either Forward Dosimetry (FD) or Monte Carlo (MC) mode. Only valid for PBPK type models

Examples

```r
## Not run:
newProject(name = "testPBPK", type = "PBPK", model = "rapidPBPK", mode = "MC")
newProject(name = "testPBPK", type = "PBPK", model = "httk_pbpk", mode = "MC")

## End(Not run)
```

newProjectGadget

Command line function launching the new project gadget

Description

This launches the new project gadget for launching a new project in PLETHEM. With RStudio, this function is available as an addin

Usage

newProjectGadget()

Examples

```r
## Not run:
newProjectGadget()

## End(Not run)
```
parseTRAFile \hspace{1cm} Parse uploaded file for Consumer TRA

**Description**

Parses the uploaded Consumer TRA exposure Spreadsheet to extract all the exposure names and values. This function will not be called directly by the user.

**Usage**

```
parseTRAFile(path)
```

**Arguments**

- **path**: Path to Excel File

---

performIVIVE \hspace{1cm} server side function for performing IVIVE for a chemical

**Description**

The server side function for running the IVIVE module from within the PBPK UI.

**Usage**

```
performIVIVE(input, output, session, km)
```

**Arguments**

- **input**: input object for the UI
- **output**: input object to the UI
- **session**: session object for the module
- **km**: km for the chemical
performIVIVEUI

performIVIVEUI  

Shiny module that is called when perform IVIVE button is clicked on the chemical tab of a PBPK model

Description

UI for performing IVIVE from within the PBPK UI. This function should not be called by the user.

Usage

performIVIVEUI(namespace)

Arguments

namespace  namespace for the UI

preprocessUIData

preprocessUIData

preprocess value list from UI

Description

The function converts the data from the UI to standard units of liters/h for clearances and mg/L for concentrations. This function is not available to the end user as it relied on UI names.

Usage

preprocessUIData(val)

Arguments

val  list containing data for each row from the UI

Value

list of clearance values for point of departure values and type of reverse dosimetry
**projectDbSelect**  
*Runs all select queries to the project database.*

**Description**

The function runs the select queries issued to the project db and returns the dataframe.

**Usage**

```python
projectDbSelect(query, db_path = "database/project.sqlite")
```

**Arguments**

- `query`: A valid SQL Query
- `db_path`: The location of the project database. This defaults to `database/project.sqlite` and is not expected to change. This function will not be called by the user directly.

---

**projectDbUpdate**  
*Runs all update queries to the project database.*

**Description**

The function runs the update queries issued to the project db.

**Usage**

```python
projectDbUpdate(query, db_path = "database/project.sqlite")
```

**Arguments**

- `query`: A valid SQL Query
- `db_path`: The location of the project database. This defaults to `database/project.sqlite` and is not expected to change. This function will not be called by the user directly.
**projectReadTable**  
*Read all the contents of a table*

**Description**

The function reads all the contents of the given table. It is used internally to save project data.

**Usage**

```
projectReadTable(tble_name, db_path = "database/project.sqlite")
```

**Arguments**

- **tble_name**: Name of the table to save. NOTE SQLITE table names are not case sensitive.
- **db_path**: The location of the project database. This defaults to database/project.sqlite and is not expected to change.

**Value**

Table as a dataframe

---

**projectWriteTable**  
*Write the dataframe to the table*

**Description**

The function writes the datafarme to the sqlite table. It will overwrite any data. It is used internally to load project data.

**Usage**

```
projectWriteTable(data, tble_name, db_path = "database/project.sqlite")
```

**Arguments**

- **data**: Data frame containing the data to save to the table. This will overwrite any existing data.
- **tble_name**: Name of the table to write the data to. NOTE SQLITE table names are not case sensitive.
- **db_path**: The location of the project database. This defaults to database/project.sqlite and is not expected to change.

**Value**

None
qsarModelDefault  \hspace{1em} \textit{Calculate Partition Coefficient using the default QSAR model}

**Description**

Calculates the partition coefficient using the default QSAR model. This QSAR model is based on the one described by Jongneelan et al as a part of the IndusChemFate model.

**Usage**

\[
\text{qsarModelDefault(chem_params, selected_org, tissue_list)}
\]

**Arguments**

- `chem_params`: A named list of parameters needed to run the model
- `selected_org`: Either "human" or "rat"
- `tissue_list`: List of tissues for which partition coefficients have to be calculated

**Value**

Named list of partition coefficients, one for each tissue in the tissue list.

---

reshapePlotData  \hspace{1em} \textit{reshape plotted data to create wide form}

**Description**

Reshapes plot data in long form to wide form. The plot data has time as the id.

**Usage**

\[
\text{reshapePlotData(plotData)}
\]

**Arguments**

- `plotData`: Plot Data in long form
runFDPBPK

**Run the PBPK models in forward dosimetry mode**

**Description**

Run the pbpk model in forward dosimetry mode. This function is common across all PBPK models. This function can be used from the console if all the inputs are provided.

**Usage**

`runFDPBPK(initial_values, model = “rapidPBPK”)`

**Arguments**

- `initial_values` A list containing initial values needed to run the model
- `model` The name of the PBPK model to simulate

runHTIVIVE

**Run HT-IVIVE**

**Description**

Launches the HT-IVIVE UI.

**Usage**

`runHTIVIVE()`

**Examples**

```r
## Not run:
runHTIVIVE()

## End(Not run)
```
**runPlthemHTIVIVE**

*High Throughput reverse dosimetry calculation using UI*

**Description**

Main function called from PLETHEM UI to run HT- Reverse Dosimetry. This should not be called by the user.

**Usage**

`runPlthemHTIVIVE(vals)`

**Arguments**

vals values passed from the PLETHEM UI

**Value**

List of oral equivalent dose, steady state plasma concentration and steady state for each name in vals

---

**saveAsParameterSet**

*server side function for saving a new physiological, chemical or exposure set to the project database*

**Description**

Server side function for running the save parameter module. This function should not be called by the user.

**Usage**

`saveAsParameterSet(input, output, session, set_type, main_input, name_df)`

**Arguments**

input input object for the UI
output input object to the UI
session session object for the module
set_type type of parameter set to save
main_input input from the pbpk UI
name_df variable names for parameters
saveAsParameterSetUI  
UI for saving a new physiological, chemical or exposure set to the project database

Description
UI for saving parameter sets. This function should not be called by the user

Usage
saveAsParameterSetUI(namespace, set_type)

Arguments
namespace namespace for the UI
set_type type of parameter set to save

saveProject  
Save the current project to a location

Description
Save the current PBPK or HTIVIVE project the user is working on. This cannot be used to save exposure or IVIVE gadget data. This function should not be called directly from the console. It will be called by the app on exit

Usage
saveProject() 

saveRestoreParameterSet
Server for the restore/save dialog

Description
Server for the save restore dialog box. This function should not be called by the user.

Usage
saveRestoreParameterSet(input, output, session, UI_values, set_values, param_names, type)
Arguments
input  input object for the UI
output input object to the UI
session session object for the module
UI_values values for the parameters in the UI
set_values values for the parameters in the database
param_names names of parameters to save or restore
type type of parameter set to save

saveRestoreParameterSetUI

UI for the restore/save dialog

Description
UI for the save restore dialog box. This function should not be called by the user.

Usage
saveRestoreParameterSetUI(namespace)

Arguments
namespace namespace for the UI

scale_cellular_enzymatic

Scale clearance by age when both cellular and enzymatic clearance are known at reference age.

Description
This function is used internally to calculate age specific metabolism using the IVIVE gadget. It needs both cellular and enzymatic clearance at at least one age, the reference age, to extrapolate to values at other ages.

Usage
scale_cellular_enzymatic(out_ages, tot_scaled_hepcl, tot_scaled_recomcl, cypDb, cypCl, gender)
Arguments

out_ages  Ages for which the clearance needs to be calculated including reference age

tot_scaled_hepcl  Total cellular clearance at reference age in L/h/kg Liver

tot_scaled_recomcl  Total Recombinant enzyme clearance at reference age in L/h/kg Liver

cypDb  Dataframe containing cyp datasets to scale measured clearance values from in-vitro to in-vivo

cypCl  Dataframe containing measured invitro enzymatic clearance.

gender  Gender either "M" for male or "F" for female

Value

List with individual enzyme and total clearance at all ages.

scale_enzymatic  Scale clearance by age for when enzymatic clearance is known at referance age.

Description

Scale clearance by age for when enzymatic clearance is known at reference age.

Usage

scale_enzymatic(out_ages, tot_scaled_recomcl, cypDb, cypCl, gender)

Arguments

out_ages  out_ages Ages for which the clearance needs to be calculated including reference age

tot_scaled_recomcl  Total Recombinant enzyme clearance at reference age in L/h/kg Liver

cypDb  Dataframe containing cyp datasets to scale measured clearance values from in-vitro to in-vivo

cypCl  Dataframe containing measured invitro enzymatic clearance.

gender  Gender, either "M" for male or "F" for female. Needed to get Liver weight
**setUserDb** 
*Sets the path to the existing Db*

**Description**

The function allows the users to set an existing database as the user database to use for plethem.

**Usage**

```r
setUserDb(path = NULL)
```

**Arguments**

- **path**: path to the user database file. If no path is provided, it launches a file select dialog.

**Examples**

```r
## Not run:
setUserDb()
setUserDb("C:/Users/Documents/PLETHEMUserDb.sqlite")

## End(Not run)
```

**updateUIInputs** 
*Update Inputs for PLETHEM UI*

**Description**

This a common function used to update the inputs for any PLETHEM User Interface. It is used by the apps to provide a common pathways for all UI updates to happen.

**Usage**

```r
updateUIInputs(session, param_df)
```

**Arguments**

- **session**: The Shiny session in which all the inputs have to be updated
- **param_df**: The parameter dataframe. The parameter dataframe has the following columns:
  - **Name**: The name of the parameter
  - **Var**: The variable in the UI representing this parameter
  - **ParamType**: The type of input, either Numeric, Radio, Checkbox, Select, Tabset. Used to identify which update function to call
  - **Val**: The value to be update with. If the type is numeric, the value is coerced to be a number.
userDbSelect

**Description**

The function runs the select queries issued to the user db and returns the dataframe the path to user database is stored in main plethem database and is selected from there.

**Usage**

```python
userDbSelect(query)
```

**Arguments**

- **query**
  - A valid SQL Query

userDbUpdate

**Description**

The function runs the update queries issued to the user db.

**Usage**

```python
userDbUpdate(query)
```

**Arguments**

- **query**
  - A valid SQL Query
Index

addChemsToHTTK, 3
addDataSet, 4
addDataSetUI, 4
calc_cyp2b6, 6
calcMPCPPGL, 5
calculatePartitionCoefficients, 5
clearProjectDb, 6
createUserDb, 6
doIVIVE, 7
externDbSelect, 7
getAllCypData, 8
getAllParamValuesForModel, 8
getAllSetChoices, 9
getAllVariabilityValuesForModel, 9
getDbConn, 10
getLifeCourseBodyHeight, 10
getLifeCourseBodyHeightMale, 11
getLifeCourseBodyHeight, 11
getLifeCourseBodyWeightFemale, 12
getLifeCourseBodyWeightMale, 12
getLifeCourseCardiacOutput, 13
getLifeCourseGlomerularFiltrationRate, 13
getLifeCourseLungDeadSpace, 14
getLifeCourseTidalVolume, 14
getLifeCourseTissuePerfusion, 15
getLifeCourseTissueVolumes, 15
getLifeCourseUrineProductionRate, 16
getLifeCourseVentilationRate, 16
getMetabData, 17
getNextID, 17
getObservationSetChoices, 18
getParameterSet, 18
getProjectChemicalList, 19
getVariabilitySetChoices, 19
HT_IVIVE, 21
HT_IVIVEUI, 21
httkCalcOralEqDose, 20
httkParameterPBTK, 20
importBatchExposure, 22
importBatchExposureUI, 22
importHTTKData, 23
importHTTKDataUI, 23
importParameterSet, 24
importParameterSetUI, 24
importSEEMData, 25
importSEEMDataUI, 25
importShedsData, 26
importShedsDataUI, 26
interactiveHT, 27
iviveGadget, 27
loadProject, 28
loadTRA, 28
mainDbSelect, 29
mainDbUpdate, 29
newEditVariability, 30
newEditVariabilityUI, 30
newProject, 31
newProjectGadget, 31
parseTRAFile, 32
performIVIVE, 32
performIVIVEUI, 33
preprocessUIData, 33
projectDbSelect, 34
projectDbUpdate, 34
projectReadTable, 35
projectWriteTable, 35
qsarModelDefault, 36
reshapePlotData, 36
runFDPBPK, 37

44
INDEX

runHTIVIVE, 37
runPlthenHTIVIVE, 38
saveAsParameterSet, 38
saveAsParameterSetUI, 39
saveProject, 39
saveRestoreParameterSet, 39
saveRestoreParameterSetUI, 40
scale_cellular_enzymatic, 40
scale_enzymatic, 41
setUserDb, 42
updateUIInputs, 42
userDbSelect, 43
userDbUpdate, 43