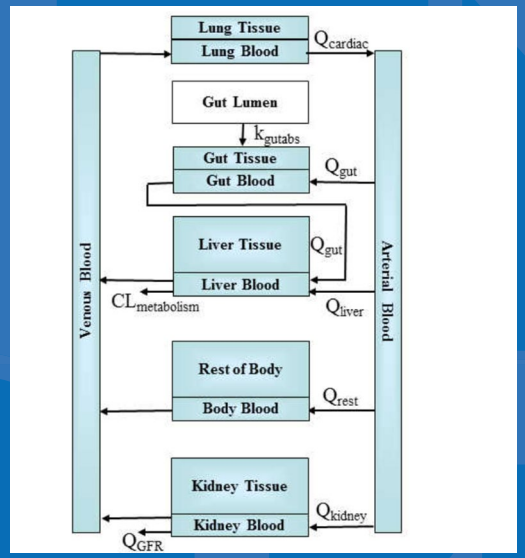
EPA United States Environmental Protection Agency

EPA Center for Computational Toxicology and Exposure (CCTE)

High Throughput Toxicokinetics (httk) Modeling Virtual Training



Marina V Evans and Celia Schacht

EPA NAMs Pilot Training Program

- New Approach Methodologies (NAMs) Training Program is a deliverable in the Agency's Work Plan, first released in 2019 and updated in 2021.
  - Previous trainings include ECOTOX, CompTox Chemicals Dashboard, and GenRA.
- Goal: Develop, implement and maintain an engaging training program.
  - Interactive case studies to encourage active learning
  - Train the trainer
  - Obtain feedback
- The EPA NAMs training website includes existing training resources, including recordings and guidance documents.

EPA NAMs Training: www.epa.gov/chemical-research/new-approach-methods-nams-training

AGENDA

- Welcome and Introductions
- Background of httk
  - Why use generic models?
- Introduction to PBPK
  - Why do different routes matter?
- Introduction to httk package and Rstudio
  - Which constants and parameters are needed?
- httk specific R functions for chemical descriptors
- Summary followed by Q&A session

Conflict of Interest Statement

- No conflicts of interest to declare.

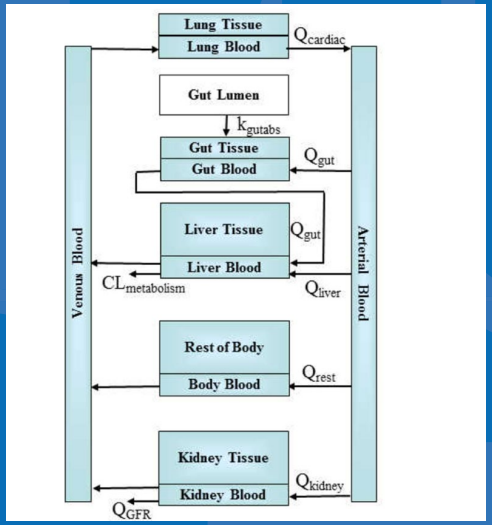
- Disclaimer:

The views expressed herein are those of the presenter and do not necessarily reflect the views or policies of the U.S. EPA

EPA United States Environmental Protection Agency

EPA Center for Computational Toxicology and Exposure (CCTE)

High Throughput Toxicokinetics (httk) Modeling Virtual Training



Marina V Evans and Celia Schacht

Httk Project Team

Facilitators

Dr. Marina V Evans*
Dr. Celia Schacht*
Christopher R Eklund*
Dr. Caroline Ring*

httk Development Team

Dr. John Wambaugh*
Dr. Caroline Ring*
Dr. Barbara Wetmore*
Robert G Pierce
Dr. R Woodrow Setzer
Many others

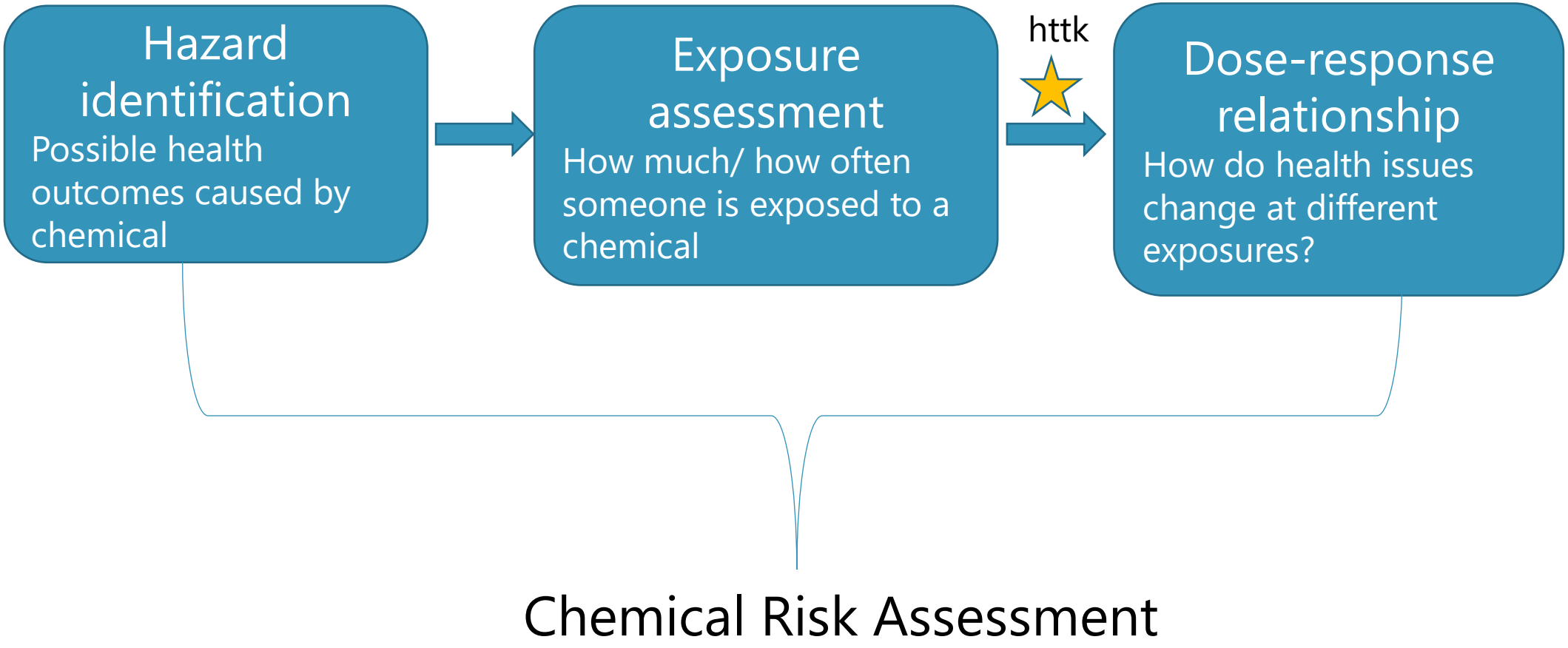
Contributers

Dr. Nisha Sipes*
Dr. Dustin Kapraun*
Dr. Miyuki Breen*
Dr. Mathew Linakis*
Dr. Greg Honda
Dr. Kristin Isaacs*
Dr. Annabel Meade
Risa Sayre*

*current

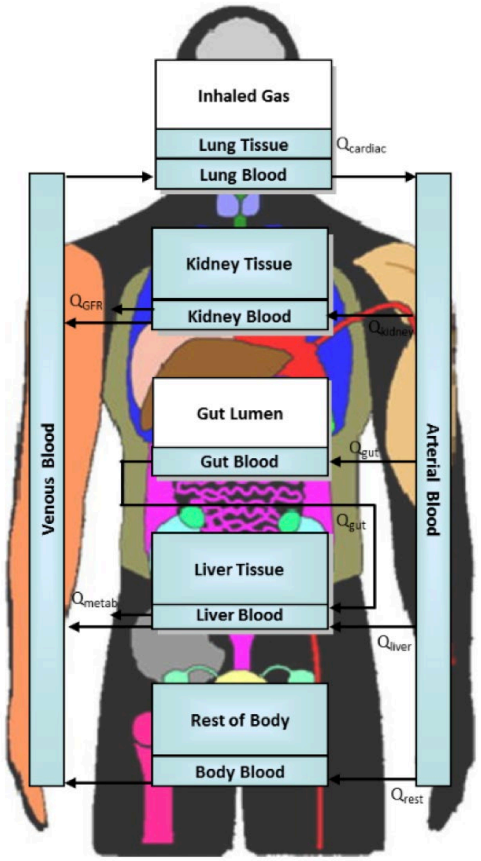
Background: Chemical Risk Assessment

How is toxicity studied and how is risk characterized?



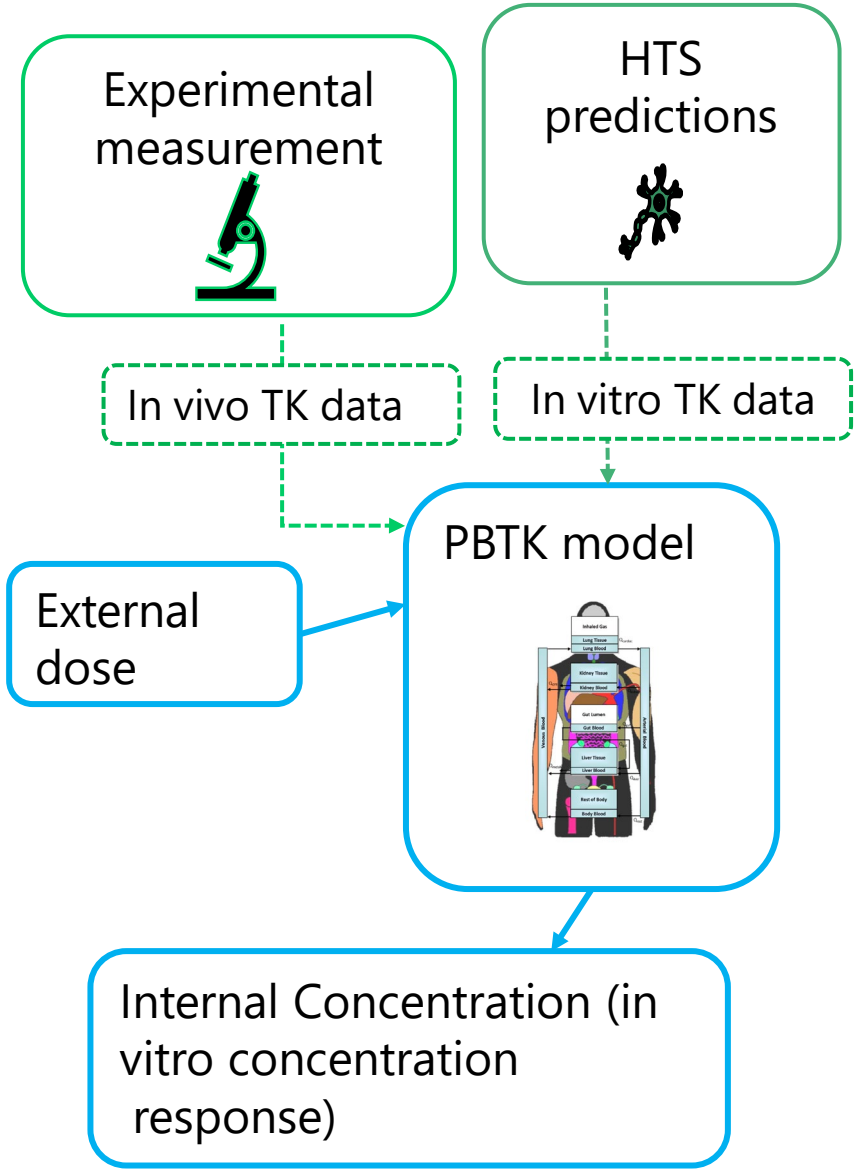
Tools to Estimate Risk: Toxicokinetics

- Toxicokinetics (TK) describes Absorption, Distribution, Metabolism, and Excretion (ADME) of a chemical by the body.
  - TK helps prediction of tissue concentrations resulting from chemical exposures → inform dose-response relationship
- Physiological Based Toxicokinetic (PBTK) Models are Used to simulate kinetics of ADME
  - Constructed with a series of ODEs
  - Parameterized by anatomical and physiological variables
  - Estimate human exposure levels from internal doses
- But TK models require chemical-specific parameters commonly found in vivo.
  - Very little in vivo data!



What is httk?

- R package
  - Created with systems of ODEs developed in MCSim, solved using compiled C code
- Goal: provide human dose information for bioactive in vitro concentrations from HTS (i.e. IVIVE)
- **Generic** models can be rapidly customized for thousands of chemicals chemicals/numerous species.



Why generic PBTK modeling?

- Generic vs. customized PBTK models.
- Model types are available within R package "httk", which is open source.
- PBTK Model parameterization
  - Physiologic parameters – different species are included.
  - Chemical-specific parameters
    - httk functions use information from the Dashboard for multiple chemicals.

Why generic PBTK modeling (cont)?

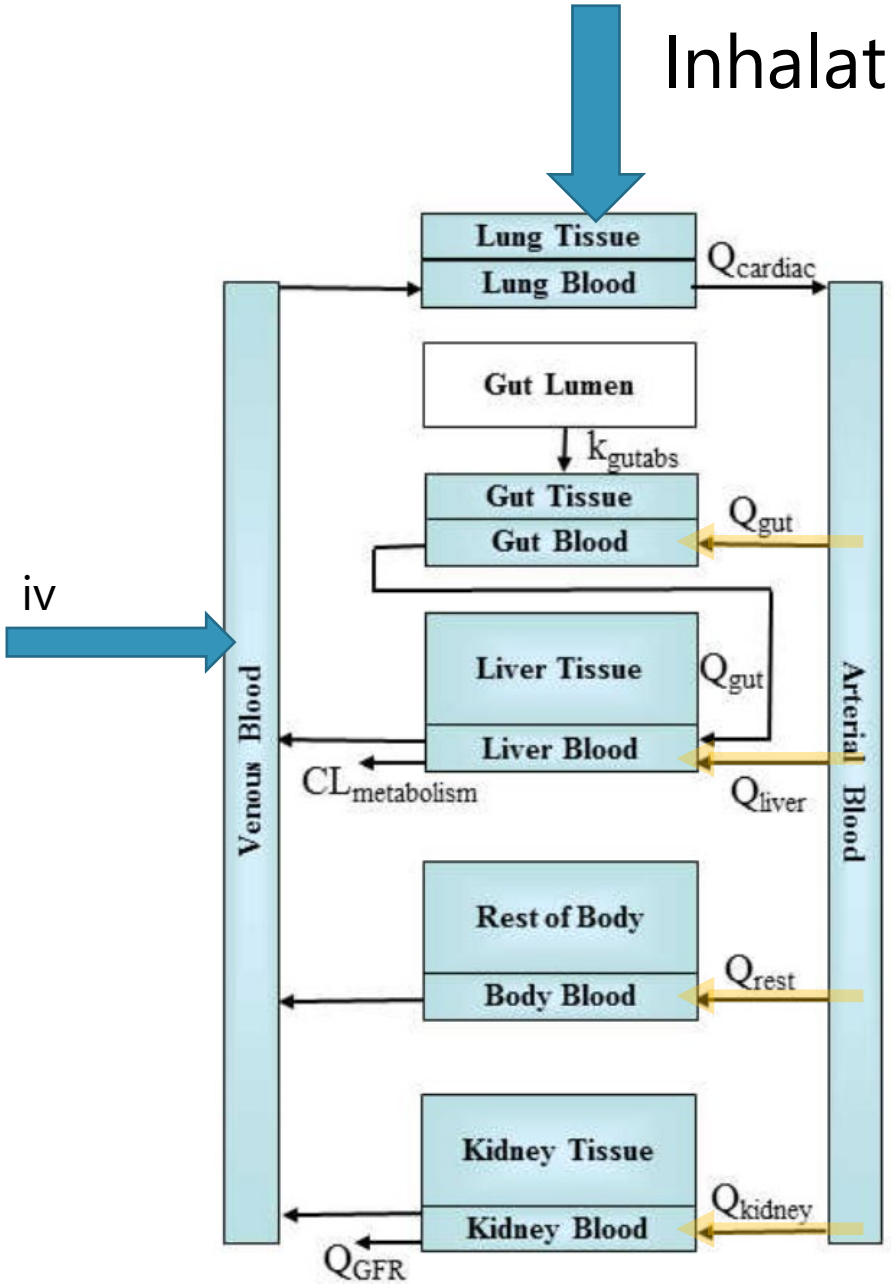
- The core model equations can be checked once and re-used for multiple chemicals.
- Estimates are needed for PBTK parameters –
  - Partition coefficients for blood and tissues
  - Clearance values
- Experimental values for PBTK parameters are not available.
  - Can be predicted using basic physico-chemical descriptors
    - Partition coefficients are based on logKow
    - Fraction unbound – only free chemical moves into tissue
    - Clearance estimation makes use of fraction unbound.

Summary

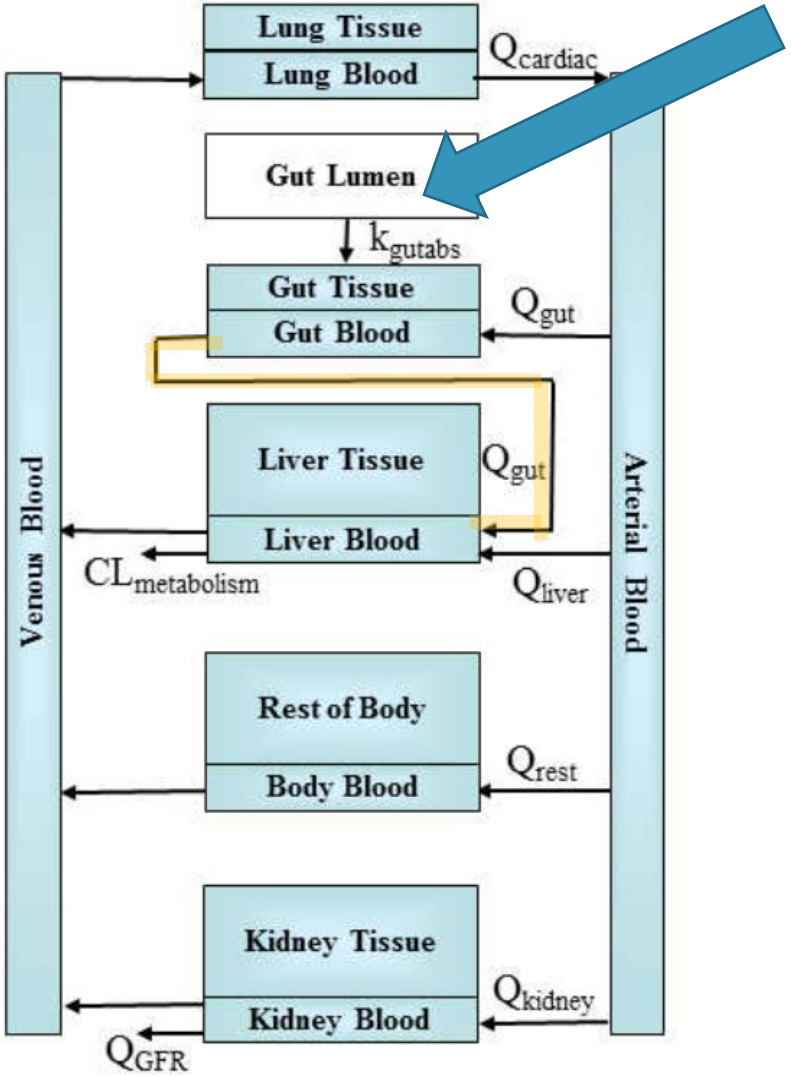
high throughput toxicokinetics =

***In vitro* toxicokinetic data + generic toxicokinetic model**

- Each of the models provided by the R package “httk” is a generic model
  - Each model is designed to used standardized chemical-specific *in vitro* measurements (fraction unbound in plasma, intrinsic hepatic clearance)
- Standardized physiology is assumed, regardless of chemical:
  - The same parameters such as volumes, flows, and rates are used
  - The same processes are included (hepatic metabolism, glomerular filtration) or omitted
- The generic model is a hypothesis
  - If we have evaluation data then we can check if we need to elaborate the model - CvTdb
- We can estimate the accuracy of a generic model for a new chemical using performance across multiple chemicals where data happen to exist



Inhalation and iv are similar in that same amount is distributed by arterial blood in parallel to each tissue by each organ's blood flow.



Oral dosing is different in that the full dose is seen by the liver first, then a fraction is cleared by the liver. The remaining amount returns to the body via venous blood.

This concept is known as **first pass effect** due to clearance by the liver.

Difference between R and RStudio

R is a programming language. https://www.r-project.org/

Rstudio is an Integrated Development Environment (IDE) designed for R.
https://rstudio-education.github.io/hopr/starting.html
Instructions included for MAC, Windows and UNIX

This course will make use of Rstudio for all examples.

If you do not have Rstudio installed, you can use:
https://web.pdx.edu/~gerbing/R/RStudioCloud.pdf

Note that you will need to create a free account before using.

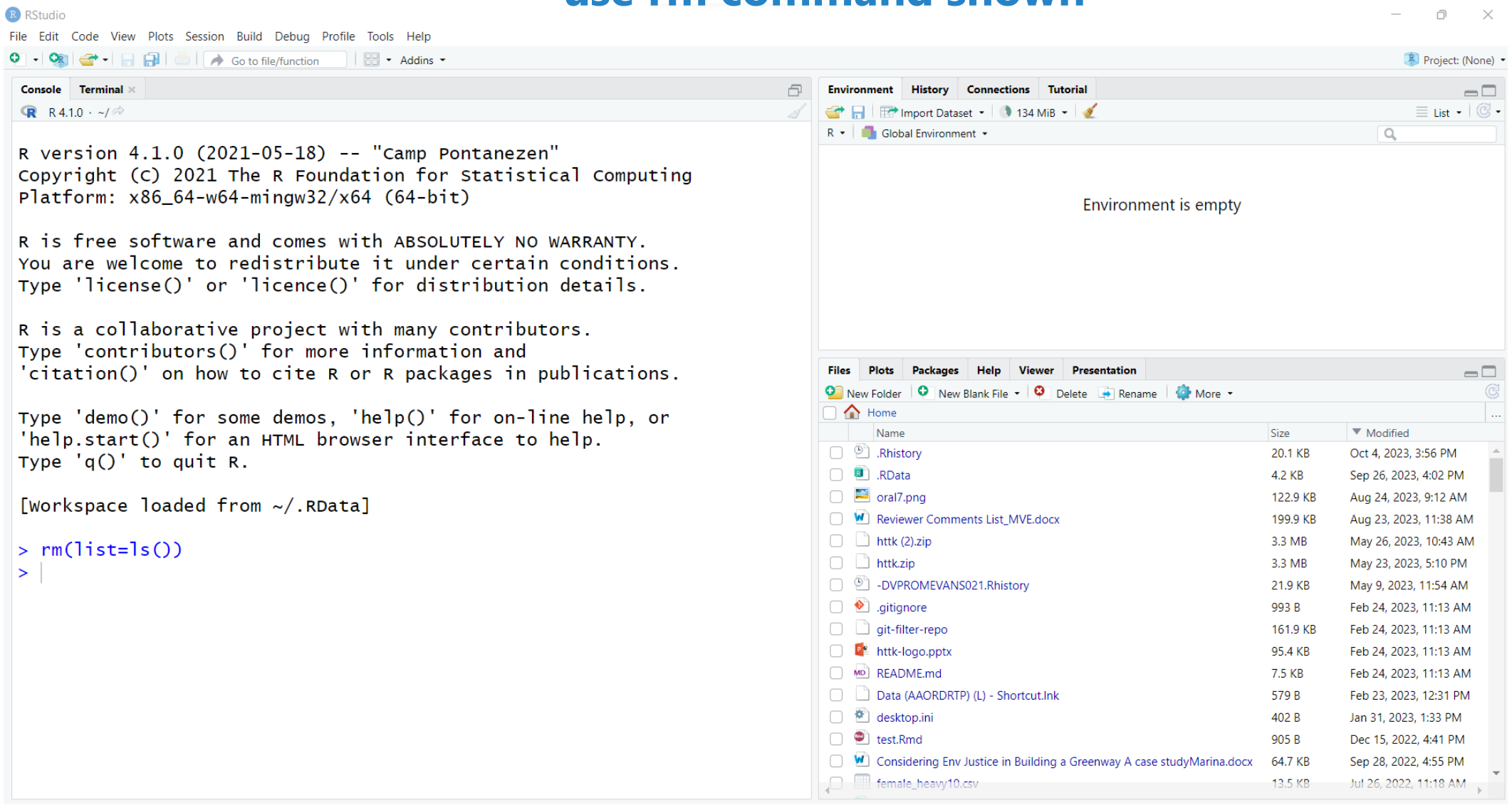
EPA United States Environmental Protection Agency

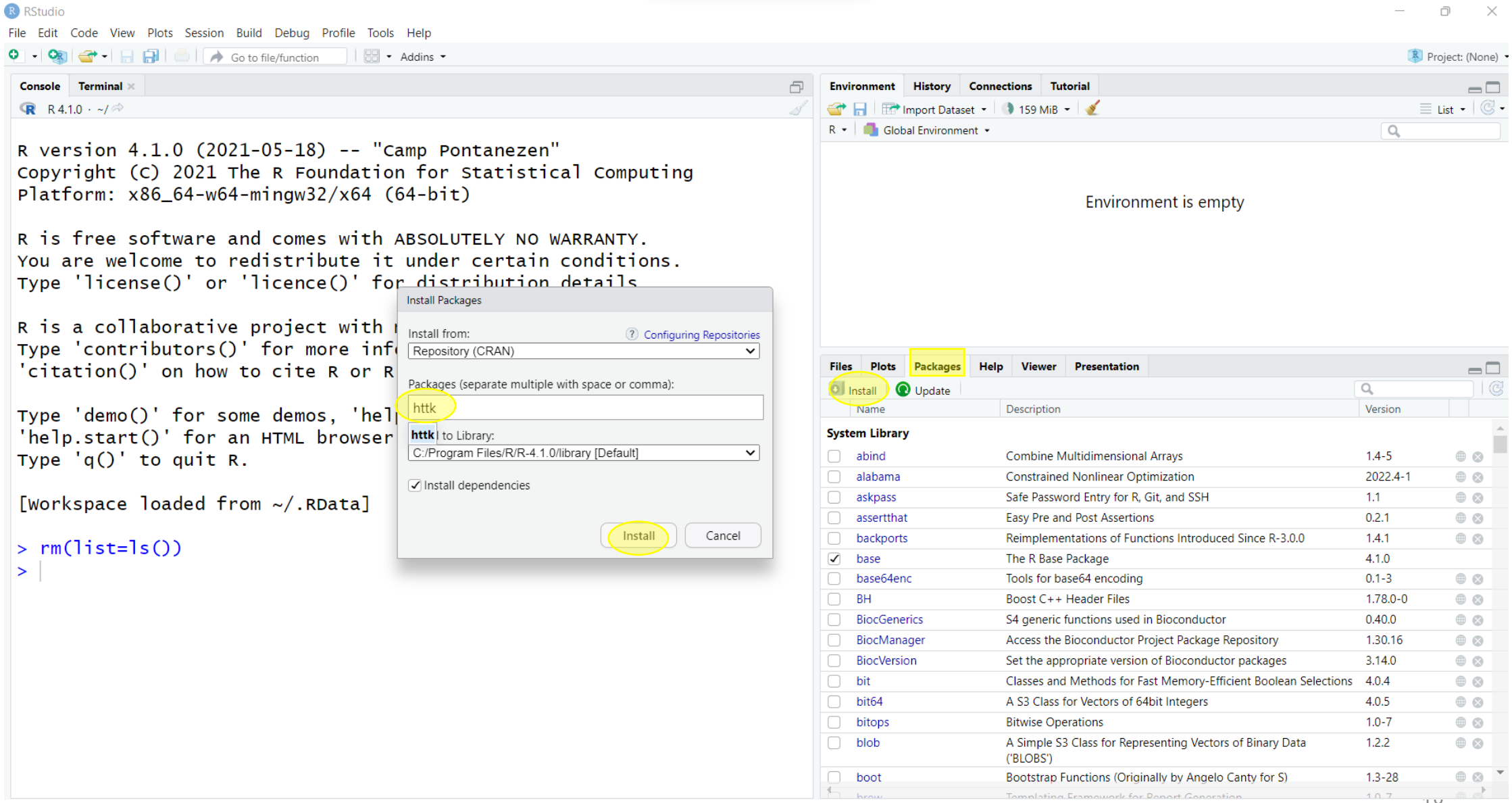
Live Demonstration

R Studio

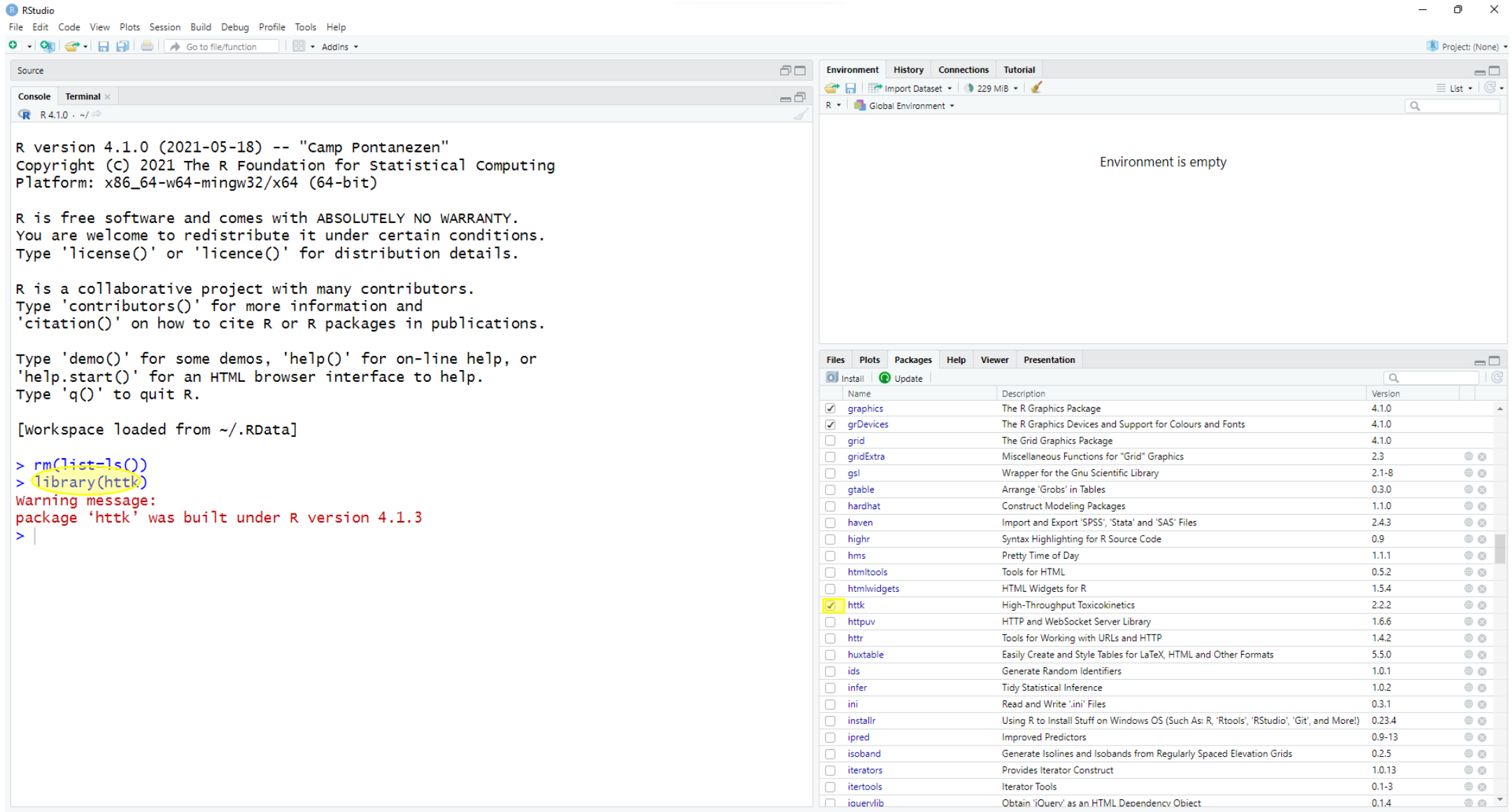
This portion of the training will be presented live. The following slides are meant to be a guide, but may not be the exact content presented during the training.

To clear previous calculations shown in the environment panel use rm command shown

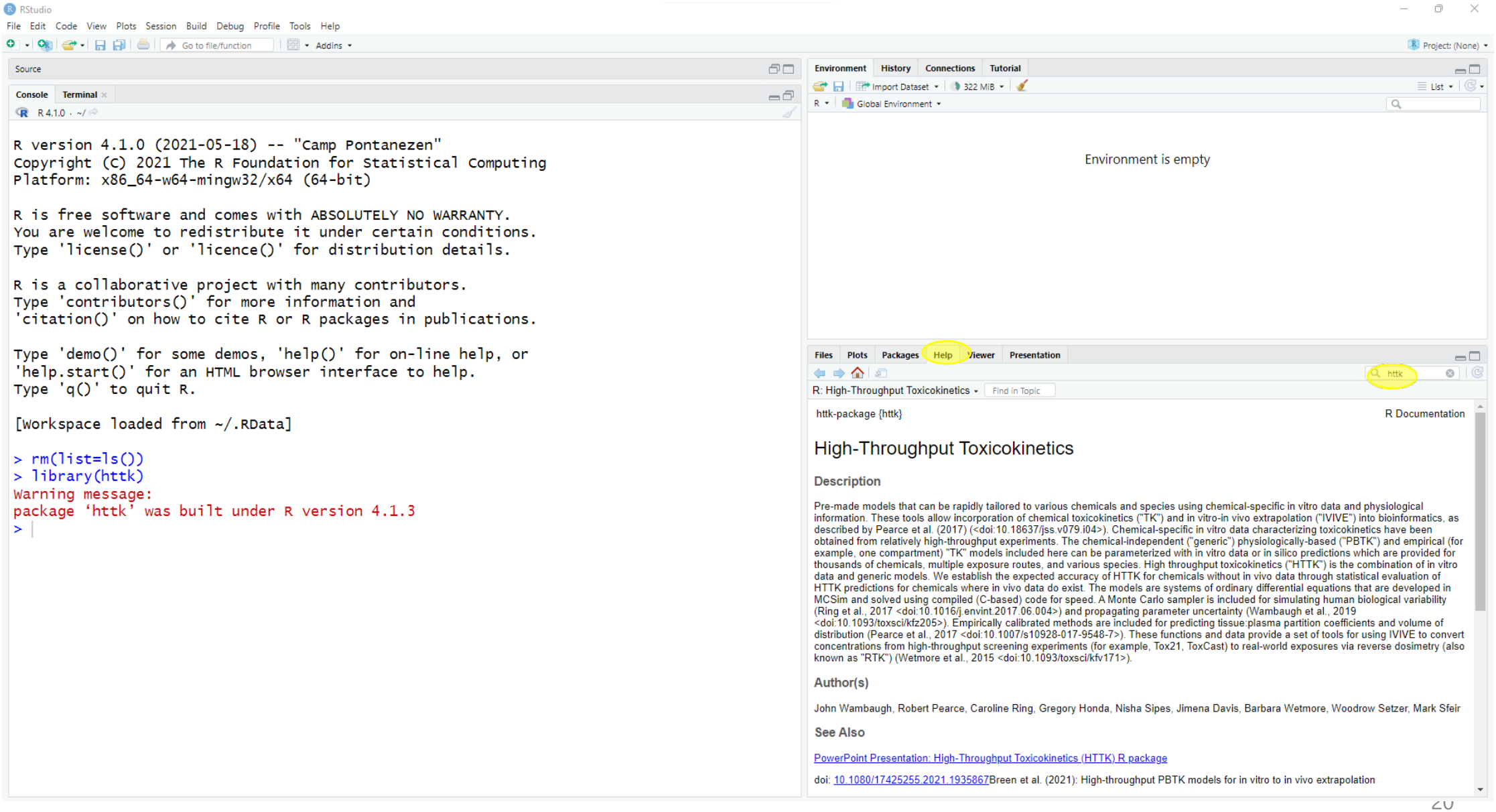



RStudio
File Edit Code View Plots Session Build Debug Profile Tools Help
Go to file/function
Addins
Project: (None)
Console
Terminal
R 4.1.0 · ~/
R version 4.1.0 (2021-05-18) -- "Camp Pontanezen"
Copyright (C) 2021 The R Foundation for Statistical Computing
Platform: x86_64-w64-mingw32/x64 (64-bit)
R is free software and comes with ABSOLUTELY NO WARRANTY.
You are welcome to redistribute it under certain conditions.
Type 'license()' or 'licence()' for distribution details.
R is a collaborative project with
Type 'contributors()' for more inf
'citation()' on how to cite R or R
Type 'demo()' for some demos, 'hel
'help.start()' for an HTML browser
Type 'q()' to quit R.
[Workspace loaded from ~/.RData]
> rm(list=ls())
>
Install Packages
Install from:
Configuring Repositories
Repository (CRAN)
Packages (separate multiple with space or comma):
httk
httk to Library:
C:/Program Files/R/R-4.1.0/library [Default]
Install dependencies
Install
Cancel
Environment
History
Connections
Tutorial
Import Dataset
159 MiB
List
R
Global Environment
Environment is empty
Files
Plots
Packages
Help
Viewer
Presentation
Install
Update
Name
Description
Version
System Library
abind Combine Multidimensional Arrays 1.4-5
alabama Constrained Nonlinear Optimization 2022.4-1
askpass Safe Password Entry for R, Git, and SSH 1.1
assertthat Easy Pre and Post Assertions 0.2.1
backports Reimplementations of Functions Introduced Since R-3.0.0 1.4.1
base The R Base Package 4.1.0
base64enc Tools for base64 encoding 0.1-3
BH Boost C++ Header Files 1.78.0-0
BiocGenerics S4 generic functions used in Bioconductor 0.40.0
BiocManager Access the Bioconductor Project Package Repository 1.30.16
BiocVersion Set the appropriate version of Bioconductor packages 3.14.0
bit Classes and Methods for Fast Memory-Efficient Boolean Selections 4.0.4
bit64 A S3 Class for Vectors of 64bit Integers 4.0.5
bitops Bitwise Operations 1.0-7
blob A Simple S3 Class for Representing Vectors of Binary Data ('BLOBS') 1.2.2
boot Bootstrap Functions (Originally by Angelo Canty for S) 1.3-28

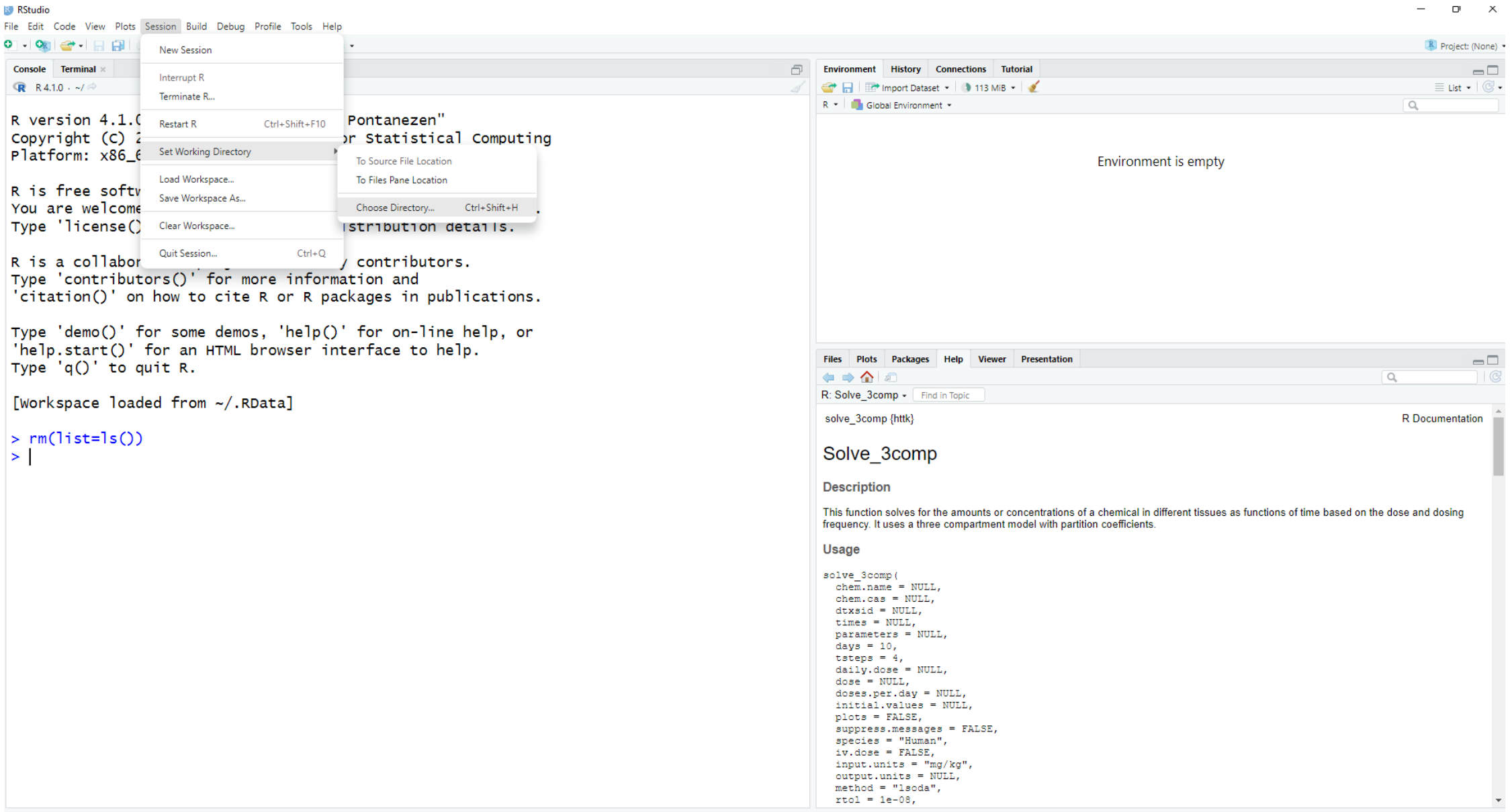
Packages are listed alphabetically inside package tab. Look for httk and check the box



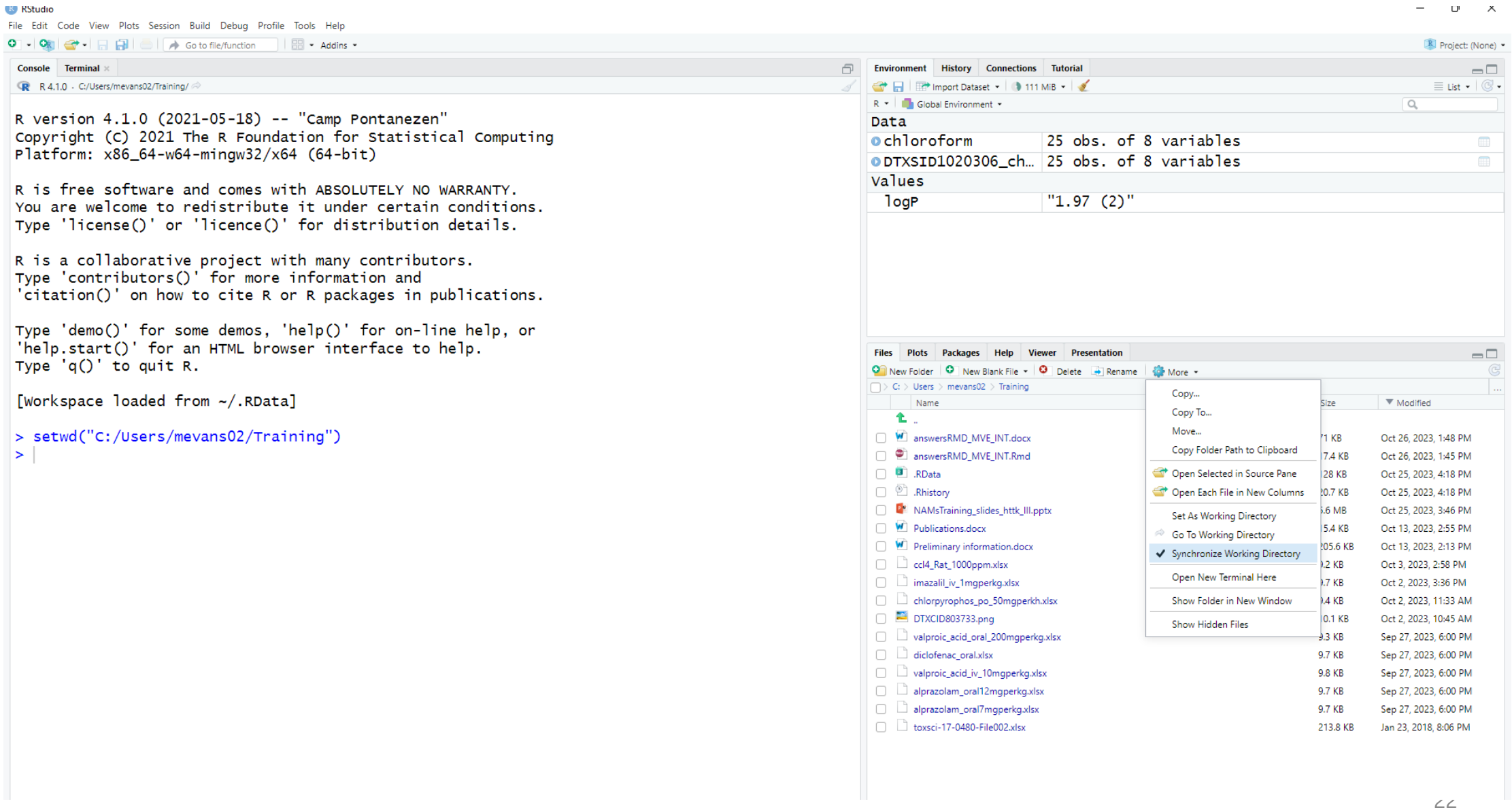
Help feature inside httk package



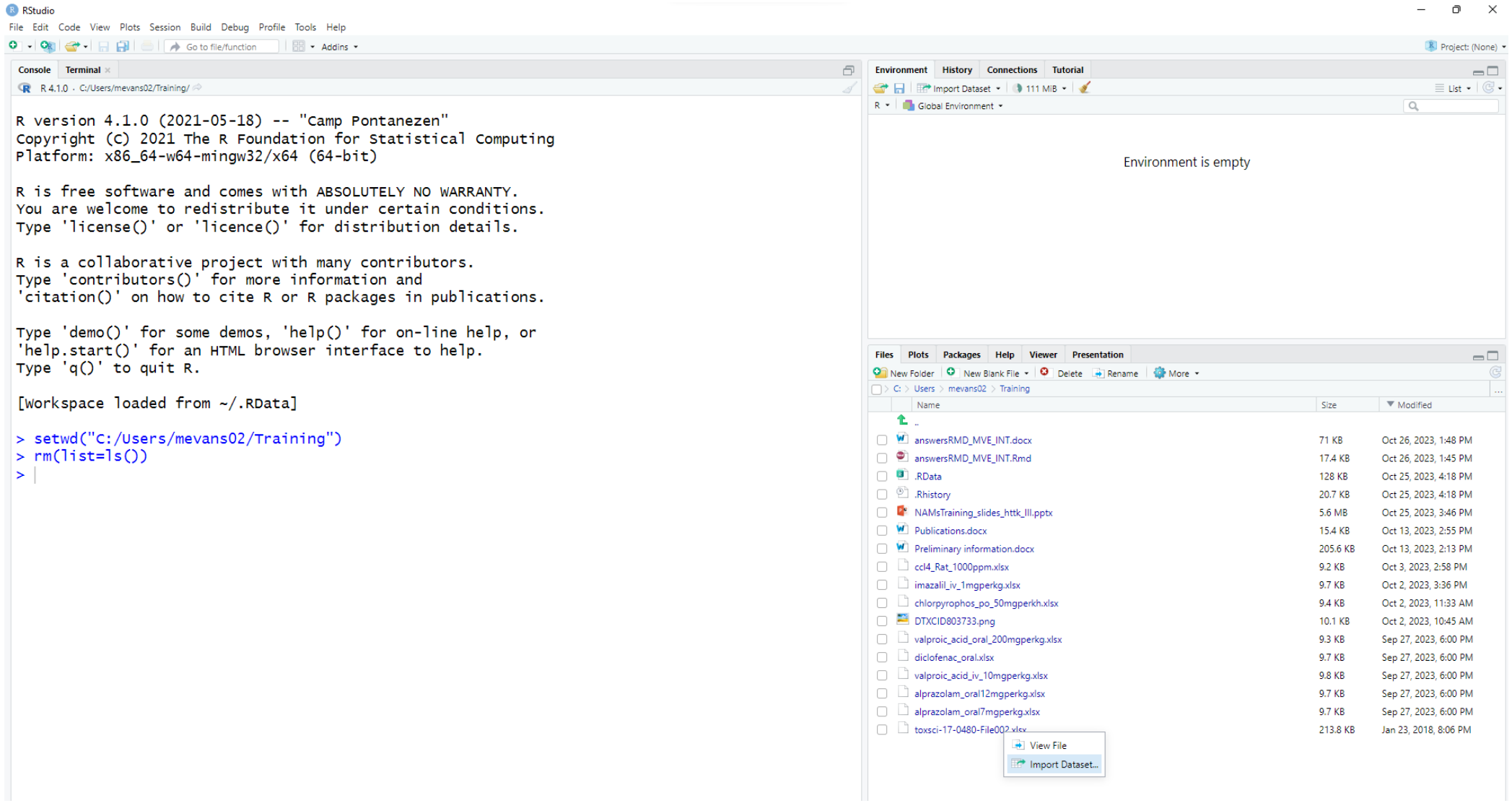
Starting a session and choosing a folder



How to synchronize your files with your R path

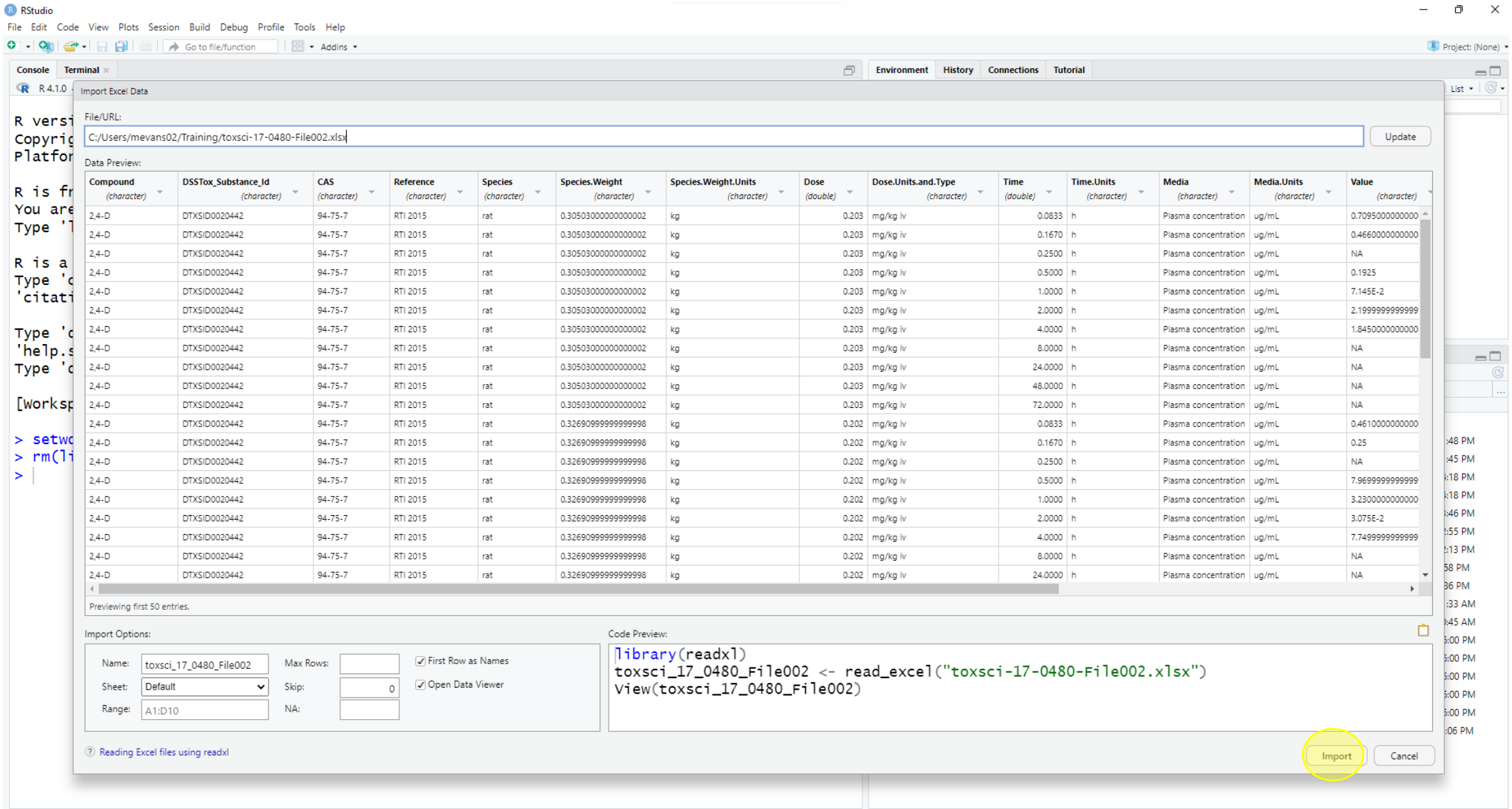


Reading an .xlsx file using readxl package

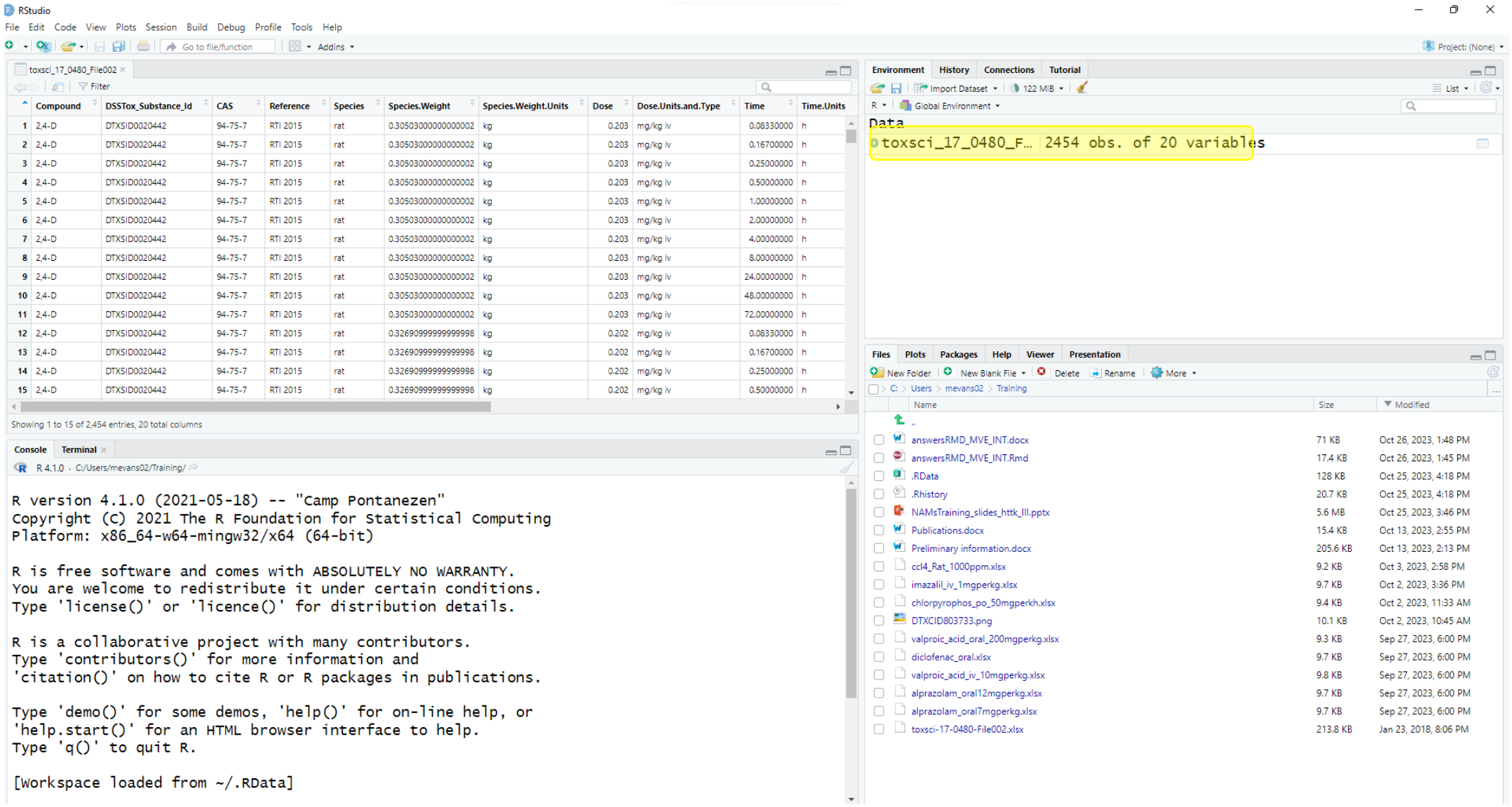


Import Dataset> From Excel

The readxl package must be installed before importing or you will get an error



Data frame will be shown in the upper left corner – script window



- We are going to use provided time course data published as supplementary information using excel file.

- All data is specific to rat, route is either iv or oral.

- There are three chemical identifiers used in httk:
  - Chemical name
  - CAS number
  - DTXSID number (US EPA generated these identifiers)

  In httk,
  Chemical name chem.name=""
  CAS number chem.cas=""
  DTXSID number dtxsid=""

  Please note: R is case sensitive.

EPA United States Environmental Protection Agency

Live Demonstration

Introduction to R Commands

Basic R commands

- Variables names are called symbols in R and are stored in the environment window.
- Names are case sensitive, must not contain reserved words and can have unlimited length.
- Variable names cannot start with an _ (underscore).
- You can assign a value to a variable using <- operator.
- Example:
  logP<- 2.2 and LogP <- 2.2 are different variables

https://shanghai.hosting.nyu.edu/data/r/concepts.html

EPA United States Environmental Protection Agency

How to index a column/matrix of numbers in R

- First, let's create a column of numbers:

  even<-c(2,4,6,8,10)

- The column is indexed from 1 thru 5 in sequential order. R always starts with a number one index.
- To obtain the number 6, we need to refer to the third index. In the Rstudio console type:

  >even[3]
  [1] 6

- Matrices have both rows and columns. Inside the brackets, always start with [row,column]

```
> numbers<-matrix(1:10, nrow=2,ncol=5)
```

```
> numbers
     [,1] [,2] [,3] [,4] [,5]
[1,]    1    3    5    7    9
[2,]    2    4    6    8   10
```

```
> numbers[2,4]
[1] 8
```

https://sudo-labs.github.io/r-data-science/vectors/

United States Environmental Protection Agency

We can set y as a vector of numbers and find its range and its length

```
> y = c(0, 1.1, 2.4, 3, 3.6, 3.4, 3, 2.2)
> y
[1] 0.0 1.1 2.4 3.0 3.6 3.4 3.0 2.2
> min(y); max(y)
[1] 0
[1] 3.6
> range(y)
[1] 0.0 3.6
> tail(y,1)
[1] 2.2
> length(y)
[1] 8
> y[length(y)]
[1] 2.2
```

A vector can also contain characters

```
> variables = c("time", "concentration")
> variables
[1] "time"          "concentration"
```

Set a sequence of numbers: `seq(from, to, step)`

```
> x = seq(1,8,1)
> x
[1] 1 2 3 4 5 6 7 8
```

Combine columns with cbind (or rbind to join by row)

```
> mat = cbind(x,y); colnames(mat) = variables
> mat
     time concentration
[1,]    1           0.0
[2,]    2           1.1
[3,]    3           2.4
[4,]    4           3.0
[5,]    5           3.6
[6,]    6           3.4
[7,]    7           3.0
[8,]    8           2.2
> mat[,"time"]
[1] 1 2 3 4 5 6 7 8
```

Or create a data frame (use the $ operator to call columns)

```
> df = data.frame(x = x,y = y); colnames(df) = variables
> df$concentration
[1] 0.0 1.1 2.4 3.0 3.6 3.4 3.0 2.2
```

Load and analyze data from an excel file

```
# Install "readxl" if not already installed and load it with "library“
install.packages("readxl")
library(readxl)
```

Remember to have the .xlsx file loaded in your same working directory!

- **Read in ToxSci data and convert it to a data frame**

```
> toxsci.data <- data.frame(read_excel("toxsci-17-0480-File002.xlsx"))
```

- **Find what information is contained in toxsci.data through the column names.**
- **Find the dimensions of the data frame.**

```
> colnames(toxsci.data)
 [1] "Compound"             "DSSTox_Substance_Id"  "CAS"                  "Reference"
 [5] "Species"              "Species.Weight"       "Species.Weight.Units" "Dose"
 [9] "Dose.Units.and.Type"  "Time"                 "Time.Units"           "Media"
[13] "Media.Units"          "Value"                "Units"                "Route"
[17] "Source"               "LOQ"                  "Subject"              "info"
> nrow(toxsci.data); ncol(toxsci.data)
[1] 2454
[1] 20
> dim(toxsci.data)
[1] 2454   20
```

This data file has 2454 rows of data and 20 columns to describe each row.

You may notice that there are repeated values in many columns. Find the unique values of elements from a vector with repeats using `unique()`.

Here, we find the unique names of the compounds in the data file.

```
> unique(toxsci.data$Compound)
 [1] "2,4-D"                        "Alachlor"                "Alprazolam"
 [4] "Antipyrine"                   "Bensulide"               "Bisphenol A"
 [7] "Boscalid"                     "Bosentan"                "Carbaryl"
[10] "Carbendazim"                  "Chloridazon"             "Chlorpyrifos"
[13] "Cyclanilide"                  "Cyclosporin A"           "Diazinon-o-analog"
[16] "Diclofenac"                   "Diltiazem"               "Dimethenamid"
[19] "Etoxazole"                    "Fenarimol"               "Flufenacet"
[22] "Formetanate hydrochloride"    "Hexobarbitone"           "Ibuprofen"
[25] "Imazalil"                     "Imidacloprid"            "Imipramine"
[28] "Metoprolol"                   "Midazolam"               "Nilvadipine"
[31] "Novaluron"                    "Ondansetron"             "Pentadecafluorooctanoic acid"
[34] "Permethrin"                   "Phenacetin"              "Phenytoin"
[37] "Propamocarb hydrochloride"    "Propyzamide"             "Pyrithiobac sodium"
[40] "Resmethrin"                   "S-Bioallethrin"          "Simazine"
[43] "Tolbutamide"                  "Triclosan"               "Valproic acid"
```

Subset the data to contain information only for the compound “2,4-D”

```
> chemical_24d = subset(toxsci.data, Compound == "2,4-D")
> head(chemical_24d)
  Compound DSSTox_Substance_Id     CAS Reference Species      Species.Weight Species.Weight.Units  Dose Dose.Units.and.Type   Time Time.Units
1    2,4-D       DTXSID0020442 94-75-7  RTI 2015     rat 0.30503000000000002                   kg 0.203            mg/kg iv 0.0833          h
2    2,4-D       DTXSID0020442 94-75-7  RTI 2015     rat 0.30503000000000002                   kg 0.203            mg/kg iv 0.1670          h
3    2,4-D       DTXSID0020442 94-75-7  RTI 2015     rat 0.30503000000000002                   kg 0.203            mg/kg iv 0.2500          h
4    2,4-D       DTXSID0020442 94-75-7  RTI 2015     rat 0.30503000000000002                   kg 0.203            mg/kg iv 0.5000          h
5    2,4-D       DTXSID0020442 94-75-7  RTI 2015     rat 0.30503000000000002                   kg 0.203            mg/kg iv 1.0000          h
6    2,4-D       DTXSID0020442 94-75-7  RTI 2015     rat 0.30503000000000002                   kg 0.203            mg/kg iv 2.0000          h
                 Media Media.Units                  Value Units Route   Source   LOQ Subject info
1 Plasma concentration       ug/mL   0.70950000000000002 mg/kg    iv RTI 2015 0.001   42736   NA
2 Plasma concentration       ug/mL   0.46600000000000003 mg/kg    iv RTI 2015 0.001   42736   NA
3 Plasma concentration       ug/mL                    NA mg/kg    iv RTI 2015 0.001   42736   NA
4 Plasma concentration       ug/mL                0.1925 mg/kg    iv RTI 2015 0.001   42736   NA
5 Plasma concentration       ug/mL             7.145E-2 mg/kg    iv RTI 2015 0.001   42736   NA
6 Plasma concentration       ug/mL 2.1999999999999999E-2 mg/kg    iv RTI 2015 0.001   42736   NA
```

Find the rows that contains a certain element or has a condition (iv route and body weight conditions) and view certain columns

```
> these.rows = which(chemical_24d$Route== "iv" & chemical_24d$Species.Weight < .32)
> these.rows
 [1]  1  2  3  4  5  6  7  8  9 10 11
> chemical_24d[these.rows,c("Time","Value")]
      Time                  Value
1   0.0833   0.70950000000000002
2   0.1670   0.46600000000000003
3   0.2500                    NA
4   0.5000                0.1925
5   1.0000             7.145E-2
6   2.0000 2.1999999999999999E-2
7   4.0000 1.8450000000000001E-3
8   8.0000                    NA
9  24.0000                    NA
10 48.0000                    NA
11 72.0000                    NA
```

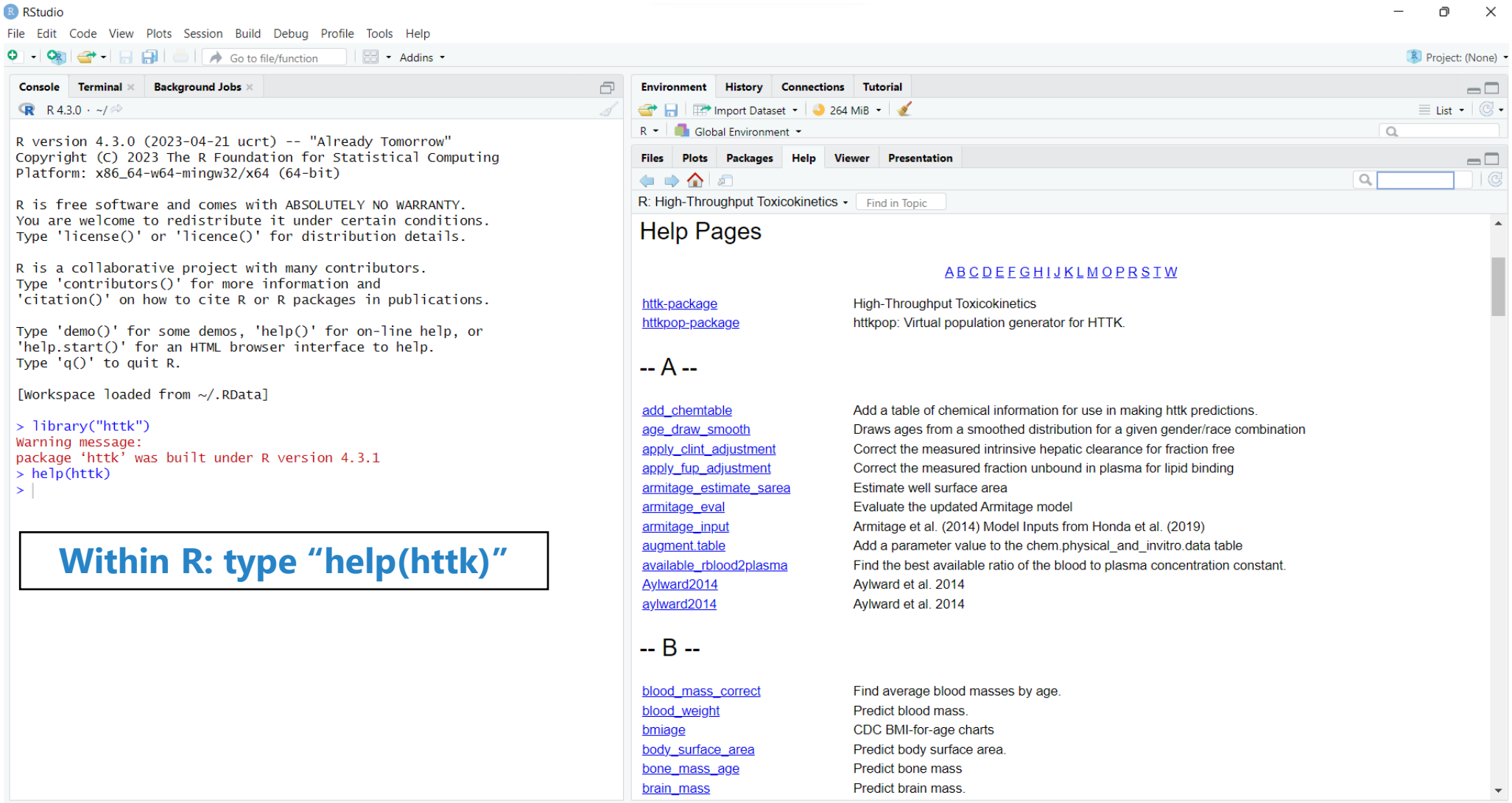
EPA United States Environmental Protection Agency

Live Demonstration

httk-Specific Functions

United States Environmental Protection Agency

Visit httk: High-Throughput Toxicokinetics (r-project.org) for a complete guide to httk



get_physchem_param {httk} R Documentation

Get physico-chemical parameters from chem.physical_and_invitro.data table

Description

This function retrieves physico-chemical properties ("param") for the chemical specified by chem.name or chem.cas from the vLiver tables.

Usage

```
get_physchem_param(param, chem.name = NULL, chem.cas = NULL, dtxsid = NULL)
```

Arguments

`param`	The desired parameters, a vector or single value.
`chem.name`	The chemical names that you want parameters for, a vector or single value
`chem.cas`	The chemical CAS numbers that you want parameters for, a vector or single value
`dtxsid`	EPA's 'DSSTox Structure ID (https://comptox.epa.gov/dashboard) the chemical must be identified by either CAS, name, or DTXSIDs

```
> get_physchem_param(param="logP", chem.cas = "94-75-7")
[1] 2.81
```

United States Environmental Protection Agency

chem.physical_and_invitro.data {httk} R Documentation

Physico-chemical properties and in vitro measurements for toxicokinetics

Description

This data set contains the necessary information to make basic, high-throughput toxicokinetic (HTTK) predictions for compounds, including Funbound.plasma, molecular weight (g/mol), logP, logMA (membrane affinity), intrinsic clearance(uL/min/10^6 cells), and pKa. These data have been compiled from multiple sources, and can be used to parameterize a variety of toxicokinetic models. See variable EPA.ref for information on the reference EPA.

Usage

```
chem.physical_and_invitro.data
```

Format

A data.frame containing 9411 rows and 54 columns.

```
> subset(chem.physical_and_invitro.data, CAS == "94-75-7")
        Compound     CAS CAS.Checksum        DTXSID   Formula                SMILES.desalt
94-75-7    2,4-d 94-75-7         TRUE DTXSID0020442 C8H6Cl2O3 OC(=O)COC1=C(Cl)C=C(Cl)C=C1
                                    All.Compound.Names logHenry logHenry.Reference logMA logMA.Reference logP logP.Reference
94-75-7 2,4-d|Dichlorophenoxy|2,4-dichlorophenoxyacetic acid|94-75-7    -8.53          OPERAv2.7    NA            <NA> 2.81      OPERAv2.7
        logPwa logPwa.Reference logWSol logWSol.Reference  MP MP.Reference  MW MW.Reference pKa_Accept pKa_Accept.Reference pKa_Donor
94-75-7   5.84        OPERAv2.7   -2.16         OPERAv2.7 141    OPERAv2.7 221          EPA       None           Sipes 2017      2.42
        pKa_Donor.Reference All.Species DTXSID.Reference Formula.Reference Human.Clint Human.Clint.pValue Human.Clint.pValue.Reference
94-75-7          OPERAv2.7   Human|Rat              EPA               EPA           0             0.1488                 Wetmore 2012
        Human.Clint.Reference Human.Fgutabs Human.Fgutabs.Reference Human.Funbound.plasma Human.Funbound.plasma.Reference Human.Rblood2plasma
94-75-7         Wetmore 2012           NA                    <NA>               0.04001                 Wetmore 2012                2.11
        Human.Rblood2plasma.Reference Mouse.Funbound.plasma Mouse.Funbound.plasma.Reference Rabbit.Funbound.plasma
94-75-7                           TNO                  <NA>                           <NA>                  <NA>
        Rabbit.Funbound.plasma.Reference Rat.Clint Rat.Clint.pValue Rat.Clint.pValue.Reference Rat.Clint.Reference Rat.Fgutabs
94-75-7                             <NA>         0           0.1365               Wetmore 2013        Wetmore 2013          NA
        Rat.Fgutabs.Reference Rat.Funbound.plasma Rat.Funbound.plasma.Reference Rat.Rblood2plasma Rat.Rblood2plasma.Reference
94-75-7                  <NA>             0.02976                  Wetmore 2013               NA                         <NA>
        SMILES.desalt.Reference Chemical.Class
94-75-7                     EPA
```

Simulating concentrations using solve_[model_name] (Example case is solve_pbtk)

Usage

```
solve_pbtk(
  chem.name = NULL,
  chem.cas = NULL,
  dtxsid = NULL,
  times = NULL,
  parameters = NULL,
  days = 10,
  tsteps = 4,
  daily.dose = NULL,
  dose = NULL,
  doses.per.day = NULL,
  initial.values = NULL,
  plots = FALSE,
  suppress.messages = FALSE,
  species = "Human",
  iv.dose = FALSE,
  input.units = "mg/kg",
  output.units = NULL,
  method = "lsoda",
  rtol = 1e-08,
  atol = 1e-12,
  default.to.human = FALSE,
  recalc.blood2plasma = FALSE,
  recalc.clearance = FALSE,
  dosing.matrix = NULL,
  adjusted.Funbound.plasma = TRUE,
  regression = TRUE,
  restrictive.clearance = TRUE,
  minimum.Funbound.plasma = 1e-04,
  monitor.vars = NULL,
  ...
)
```

- Make sure to set either chemical name, CAS, or DTXSID
- All other function inputs will default unless otherwise specified
- To set the time sequence:
  - Days: number of days
  - Tsteps: number of steps per hour
  - Times: specified sequence of
- Dosing:
  - Dose = single dose (default mg/kg)
  - Daily.dose = total daily dose
  - Doses.per.day
  - Iv.dose = TRUE or FALSE to simulate iv or oral dosing
- Units
  - Set desired units (default output.units umol or uM)

Simulating concentrations using solve_[model_name] (Example case is solve_pbtk)

Use solve_[model_name] with your chosen inputs. Here, we are looking at a 0.203 mg/kg iv dose for a rat over the course of 4 hours

```
> out = solve_pbtk(chem.cas = "94-75-7",
+             species = "Rat",
+             dose = 0.203,
+             input.units = "mg/kg",
+             output.units = "mg/L",
+             days = 4/24,
+             iv.dose = TRUE,
+             suppress.messages = TRUE)
> head(out)
        time     Aven   Cgut Cliver  Cven  Clung  Cart  Crest Ckidney Cplasma  Atubules Ametabolized       AUC
[1,] 0.00000 0.050740 0.0000 0.0000 7.028 0.0000 0.000 0.0000  0.0000  3.3300 0.000e+00            0 0.0000000
[2,] 0.00100 0.008813 0.1793 0.6676 1.220 0.1934 1.220 0.1363  0.5580  0.5784 3.768e-05            0 0.0007114
[3,] 0.01042 0.008760 0.1783 0.6650 1.213 0.1923 1.213 0.1354  0.5547  0.5748 3.426e-04            0 0.0061390
[4,] 0.02083 0.008701 0.1771 0.6606 1.205 0.1910 1.205 0.1345  0.5510  0.5711 6.774e-04            0 0.0121100
[5,] 0.03125 0.008643 0.1759 0.6564 1.197 0.1897 1.197 0.1336  0.5474  0.5671 1.010e-03            0 0.0180400
[6,] 0.04167 0.008586 0.1747 0.6520 1.189 0.1885 1.189 0.1327  0.5437  0.5633 1.341e-03            0 0.0239300
```

Parameterize Function

parameterize_1comp	Parameters for a one compartment (empirical) toxicokinetic model
parameterize_3comp	Parameters for a three-compartment toxicokinetic model (dynamic)
parameterize_fetal_pbtk	Parameterize_fetal_PBTK
parameterize_gas_pbtk	Parameters for a generic gas inhalation physiologically-based toxicokinetic model
parameterize_pbtk	Parameters for a generic physiologically-based toxicokinetic model
parameterize_schmitt	Parameters for Schmitt's (2008) Tissue Partition Coefficient Method
parameterize_steadystate	Parameters for a three-compartment toxicokinetic model at steady-state

```
parameterize_pbtk(
  chem.cas = NULL,
  chem.name = NULL,
  dtxsid = NULL,
  species = "Human",
  default.to.human = FALSE,
  tissuelist = list(liver = c("liver"), kidney = c("kidney"), lung = c("lung"), gut =
    c("gut")),
  force.human.clint.fup = FALSE,
  clint.pvalue.threshold = 0.05,
  adjusted.Funbound.plasma = TRUE,
  adjusted.Clint = TRUE,
  regression = TRUE,
  suppress.messages = FALSE,
  restrictive.clearance = TRUE,
  minimum.Funbound.plasma = 1e-04,
  million.cells.per.gliver = 110,
  liver.density = 1.05,
  kgutabs = 2.18
)
```

- Make sure to set either chemical name, CAS, or DTXSID
- All other function inputs will default unless otherwise specified
- Ie: default species is human but can be set to: Rat, Human, Mouse, Rabbit
- Set default.to.human = TRUE to substitute in human values when species data is unavailable
- Parameters include tissue:plasma partition coefficients, organ volumes, and flows for the tissue lumping scheme specified by argument tissue list.

First few parameters produced from parameterize function

```
> parameterize_pbtk(chem.cas = "94-75-7", species = "Rat")
$BW
[1] 0.25

$Clint
[1] 0

$Clint.dist
[1] NA

$Clmetabolismc
[1] 0

$Fgutabs
[1] 1

$Fhep.assay.correction
[1] 0.9563

$Funbound.plasma
[1] 0.02976
```

Parameter values can be changed by the user. Notice the default value for body weight (BW) for a rat is 0.25. This parameter can be switched by

```
> parms = parameterize_pbtk(chem.cas = "94-75-7",
+         species = "Rat", suppress.messages =  TRUE)
> parms$BW = 0.31
> head(parms)
$BW
[1] 0.31
```

The solve_model function can then be run with only “parameters = parms”

```
> out = solve_pbtk(parameters = parms,
+                  dose = 0.203,
+                  input.units = "mg/kg",
+                  output.units = "mg/L",
+                  days = 4/24,
+                  iv.dose = TRUE,
+                  suppress.messages = TRUE)
```

Plotting in base R

Let's plot the httk solution (for the 0.203 mg/kg iv dose) against the data

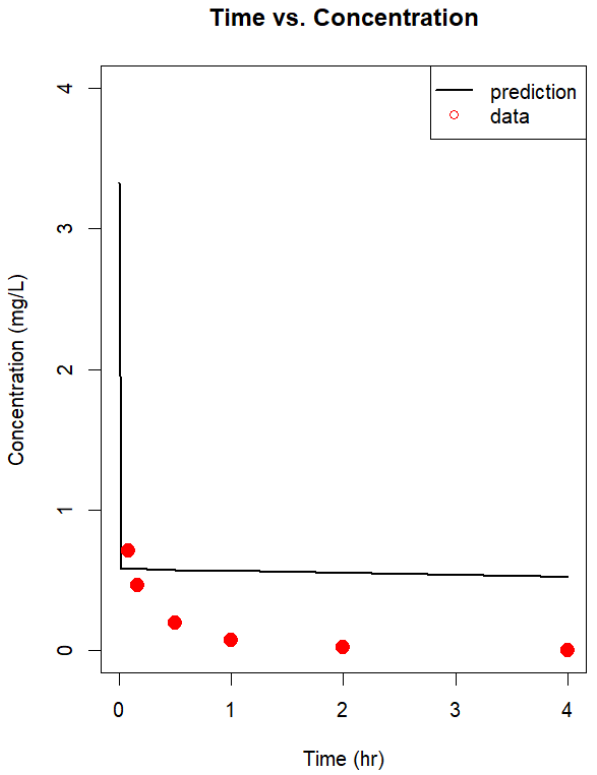
Find your experimental data points

```
experimental.points = subset(chemical_24d, Subject == 42736)[,c("Time","Value")]
experimental.points = data.frame(na.omit(sapply(experimental.points,as.numeric)))
```

```
plot(out[,"time"]*24, out[,"Cplasma"],        # plot x and y values
     type="l",                                # if you are plotting a line, type = "l"
     xlab = "Time (hr)",                      # set x and y axis labels
     ylab = "Concentration (mg/L)",
     main = "Time vs. Concentration",         # set main title
     lwd = 2,                                 # set line width
     ylim = c(0,4))                           # specify y (or x) limits

points(experimental.points$Time, experimental.points$Value,    # use "points" to overlay points
       col="red",                                              # set color
       pch = 19,                                               # set point shape
       cex=1.5)                                                # set point size

legend("topright",                           # set legend position
       legend = c("prediction","data"),      # set text for legend elements
       lwd = c(2,NA),                        # set lwd for legend (NA if corresponds to points)
       pch = c(NA, 1),                       # set pch for legend (NA if corresponds to lines)
       col=c("black","red"))                 # set color for each legend element
```



*To overlay another line, use `lines(x,y)`

Summary

- Introduction to PBTK modeling
- Rstudio basics
- What is httk?
- Basic R commands
- httk specific commands

Tomorrow we will work with concrete PK examples

- Different compartment models available
- How to parameterize a model
- Different routes
- Comparing data and simulations

Please join us for tomorrow's session!

EPA
United States
Environmental Protection
Agency

Q&A

EPA
United States
Environmental Protection
Agency

We'll see you tomorrow!

Begins at 10:00 AM EST.
Join using the link sent via email.