httk and HTTK-Pop: open-source software for simulation of population variability in high-throughput toxicokinetic modeling for in vitro-in vivo extrapolation and rapid chemical prioritization

Caroline L. Ring; Miyuki Breen; John F. Wambaugh

Center for Computational Toxicology and Exposure

Office of Research and Development

United States Environmental Protection Agency

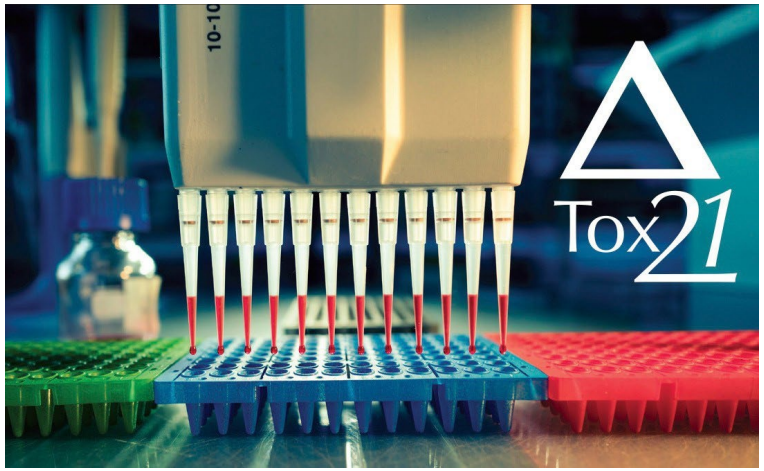
The views expressed in this presentation are those of the author and do not necessarily reflect the views or policies of the U.S. EPA

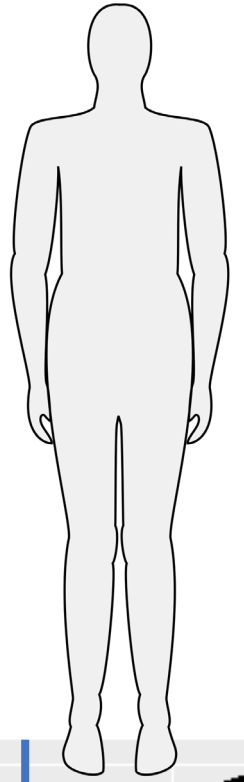
Scenario: Screening a large number of data-poor chemicals for potential human health risk

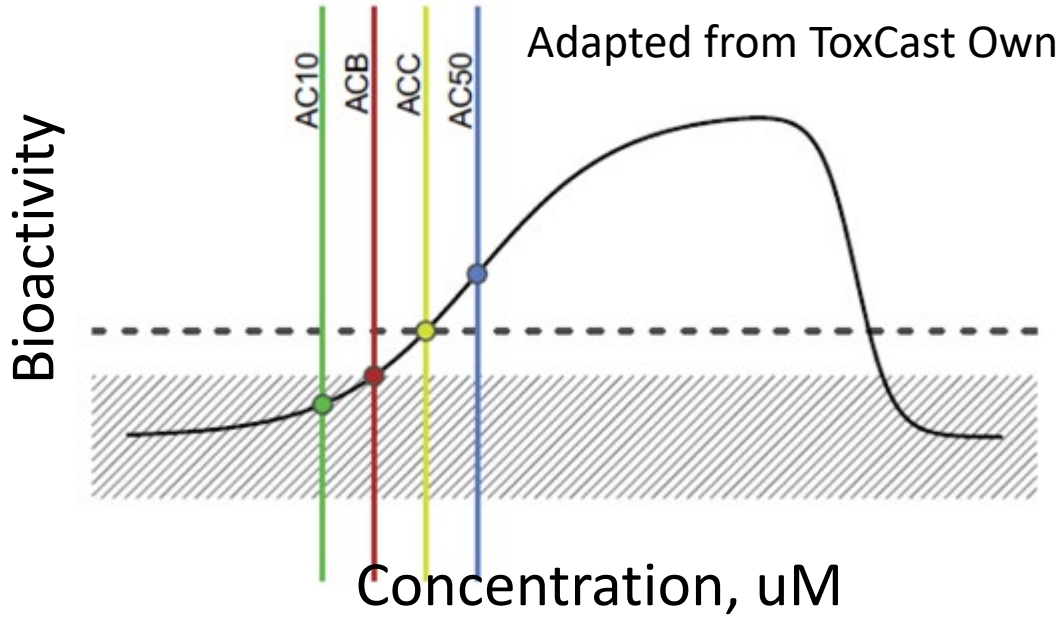
- *In vivo* toxicity data aren't available for thousands of chemicals present in the environment & used in commerce [NRC 2007; Bell et al. 2018; Bessems et al. 2014]
- Alternative: *in vitro* high-throughput screening (HTS) assays (e.g. ToxCast/Tox21) [Schmidt 2009; Dix et al. 2007; Kavlock et al. 2018]
  - Chemicals are examined by a battery of *in vitro* tests for biological activity across a variety of different endpoints
  - *In vitro* HTS data are available on the EPA CompTox Dashboard

In vitro to *in vivo* extrapolation (IVIVE)

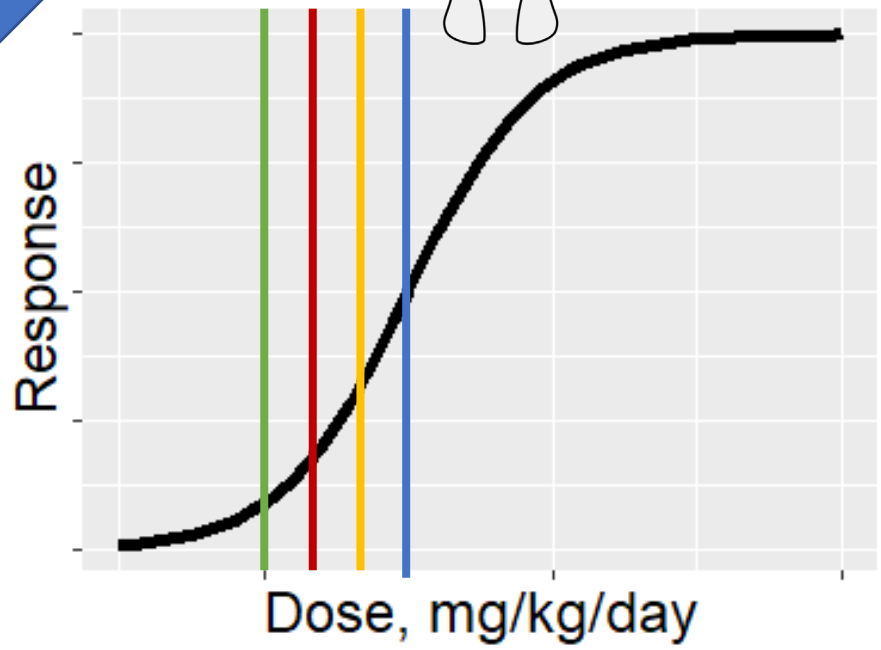
Translate *in vitro* bioactive concentration to an equivalent *in vivo* dose







Adapted from ToxCast Owners Manual



IVIVE is performed using toxicokinetics (TK) modeling: relate external dose to internal body concentration

External dose

Body concentration

- Absorption
- Distribution
- Metabolism
- Excretion

Reverse dosimetry: go from concentration to dose

External dose

Body concentration

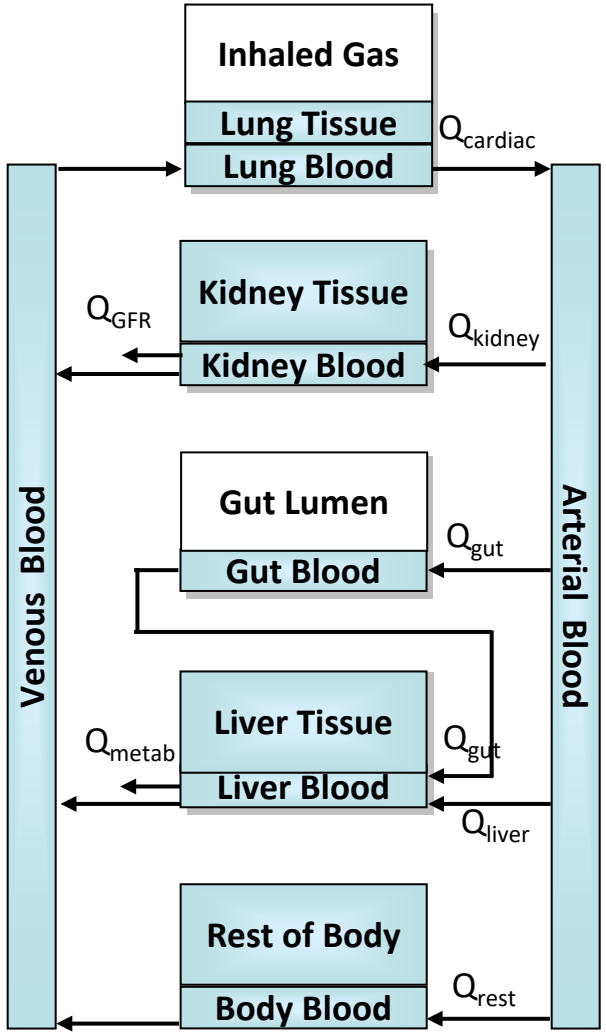
Find the **administered equivalent dose (AED)**: The dose that would produce a body concentration equal to the *in vitro* bioactive concentration.

- Absorption
- Distribution
- Metabolism
- Excretion

Assume: if this is equal to the *in vitro* bioactive concentration, then you might see some effects *in vivo*

Tan et al. 2007; Wetmore et al. 2015

A TK model relates dose and body concentration by describing how a chemical moves through the body



Linakis et al., 2020

Body is represented by "compartments" connected by "flows" — mass balance applies

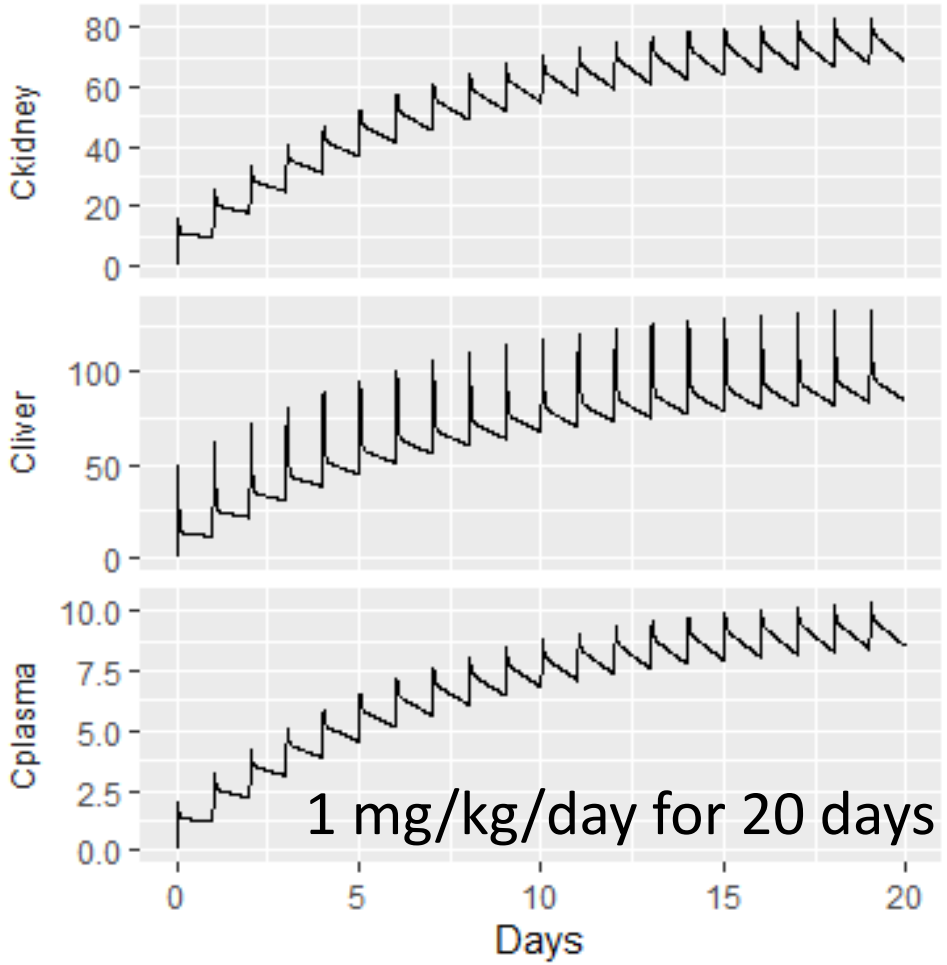
For a *physiologically-based* TK (PBTK) model, compartments represent individual organs/tissues (like liver, kidney, gut, lung, blood), and/or represent "lumped" groups of tissues (like a catch-all "rest of body" compartment)

PBTK model parameters fall into two groups:

- **Physiological parameters**: Describe physiological quantities that stay the same regardless of chemical, like organ masses; blood flows to organs; body weight; kidney function
- **Chemical-specific parameters**: Describe quantities that change for different chemicals, like intrinsic hepatic metabolism; plasma protein binding; blood:tissue partition coefficients (how much of the chemical diffuses into organ tissue vs. staying in the bloodstream)

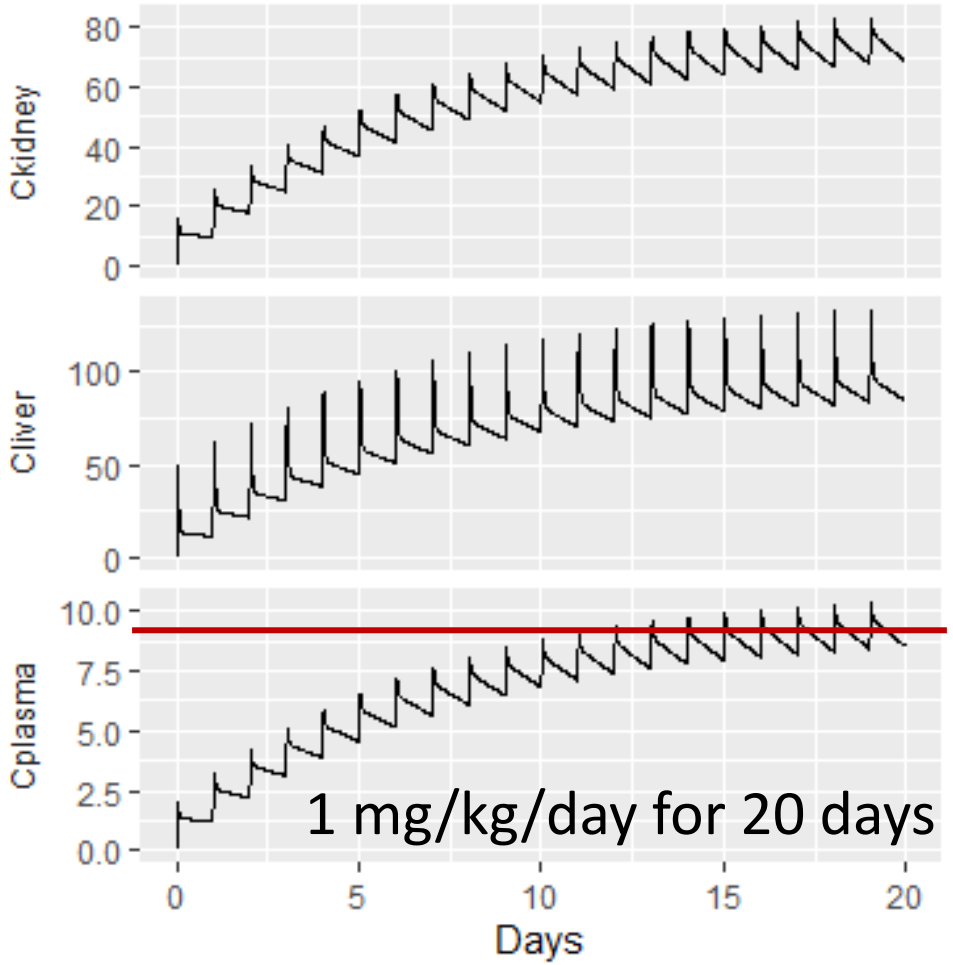
Campbell et al., 2012

TK model tracks the amount or concentration of a chemical in each compartment (vs. time), after single or repeated dosing



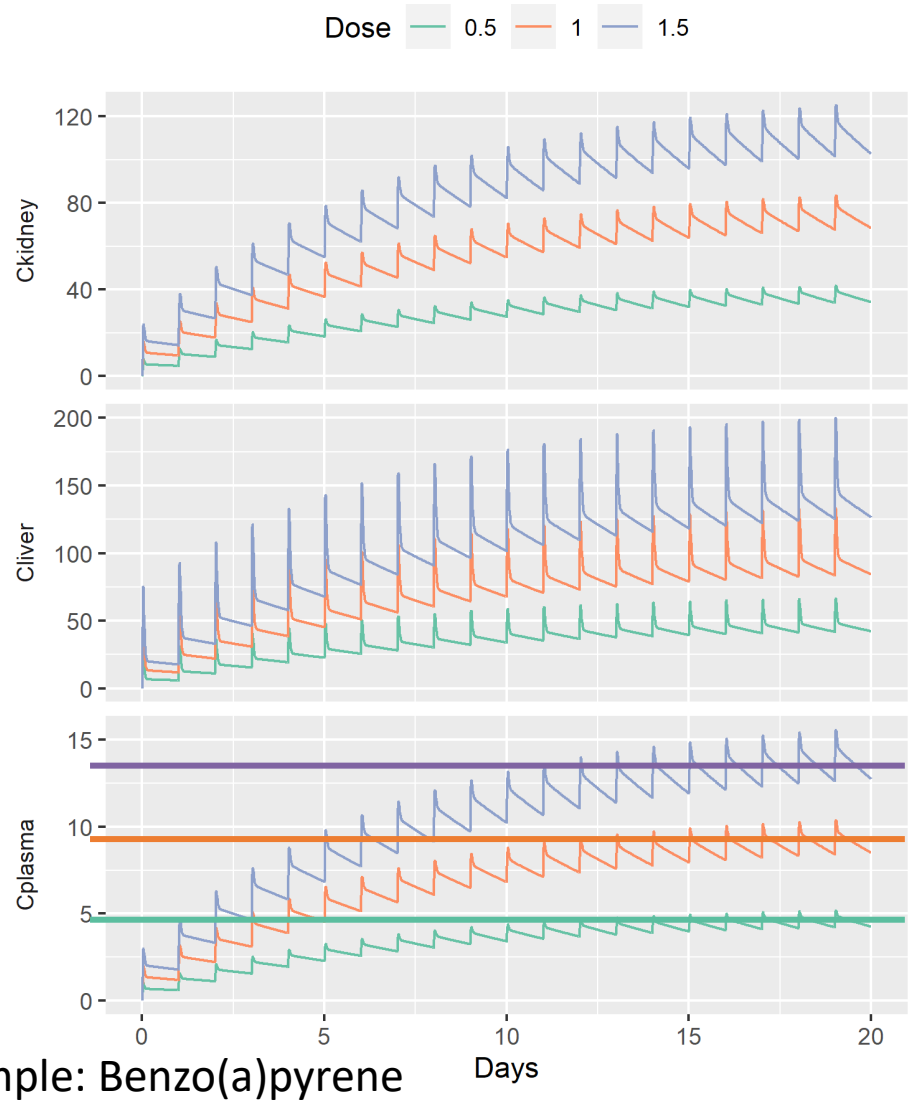
Example: Benzo(a)pyrene

For screening purposes, we are usually interested in long-term, low-level exposures, so we focus on the steady-state plasma concentration (Css) after long-term repeated dosing

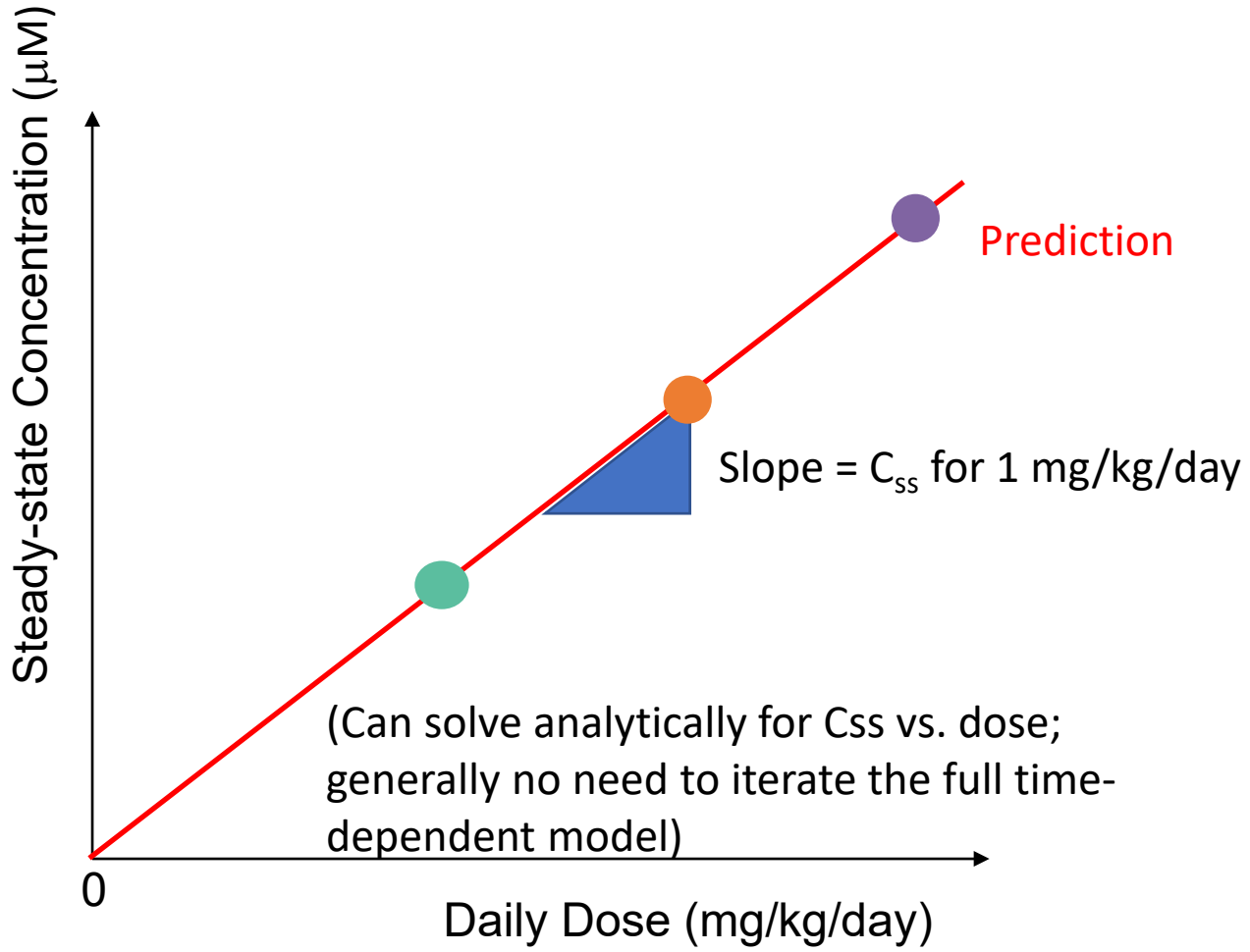


Example: Benzo(a)pyrene

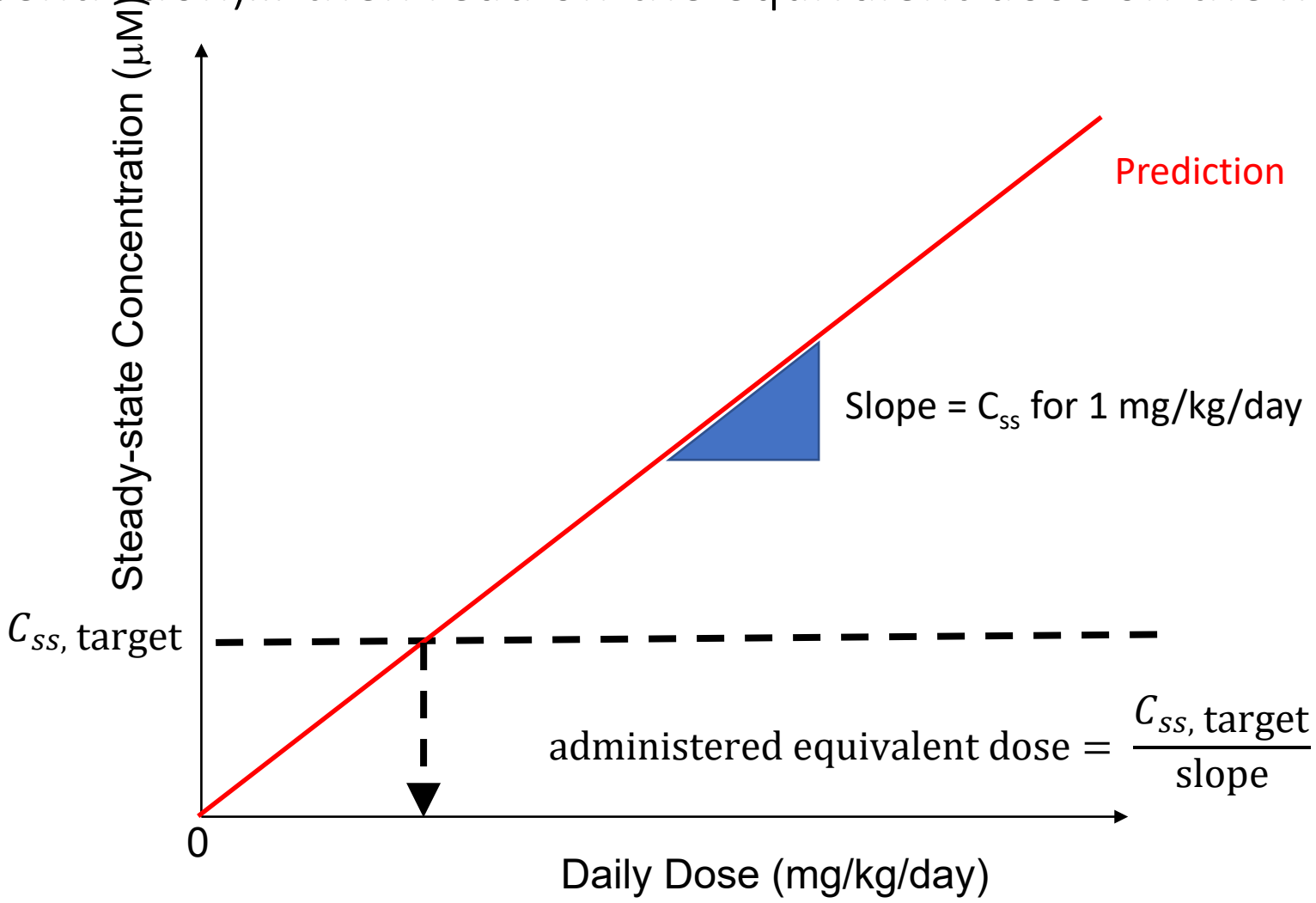
We use relatively simple TK models where Css has a linear relationship with dose



Example: Benzo(a)pyrene



Linear relationship makes reverse dosimetry quick & easy: calculate slope, start with the "target" concentration on the y-axis *(in vitro* bioactive concentration)… then read off the equivalent dose on the x-axis



Wetmore *et al*. (2012)

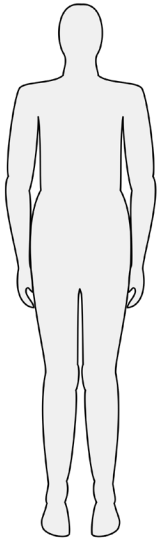
For rapid reverse dosimetry for large numbers of chemicals: we need *high-throughput* TK (HTTK) methods [Wetmore 2015]

- Choose generic PBTK models with minimal chemical-specific parameters
- How to get chemical-specific parameters rapidly (without having to measure them *in vivo*):
  - *in vitro* methods adapted from pharma [Wetmore et al. 2012, 2015]
  - *in silico* prediction methods based on chemical structure [Pradeep et al. 2020; Mansouri et al., 2018; Ingle et al., 2018; Sipes et al., 2017; Vilar et al. 2008; Yin et al. 2014]

Generic PBTK models + *in vitro* TK data to enable HTTK & IVIVE: R package "httk" [Pearce et al., 2017]

- R is an open-source programming language & environment for statistical computing (freely available at https://cloud.r-project.org/)
- R has a strong culture of user-created packages – and our group at EPA decided to create one for HTTK, creatively titled "httk". It is open-source and freely available at https://cran.r-project.org/package=httk
- The "httk" package contains (among other things):
  - Generic PBTK models
  - Tables of chemical-specific TK parameters measured *in vitro* (for about 1000 chemicals) and predicted *in silico* (for 8758 chemicals)
  - Tables of physiological TK parameters (for multiple species)
  - Pre-built functions to let users easily solve TK models & perform reverse dosimetry for large numbers of chemicals

Assessing potential risk for a population: need to model *population variability* in TK



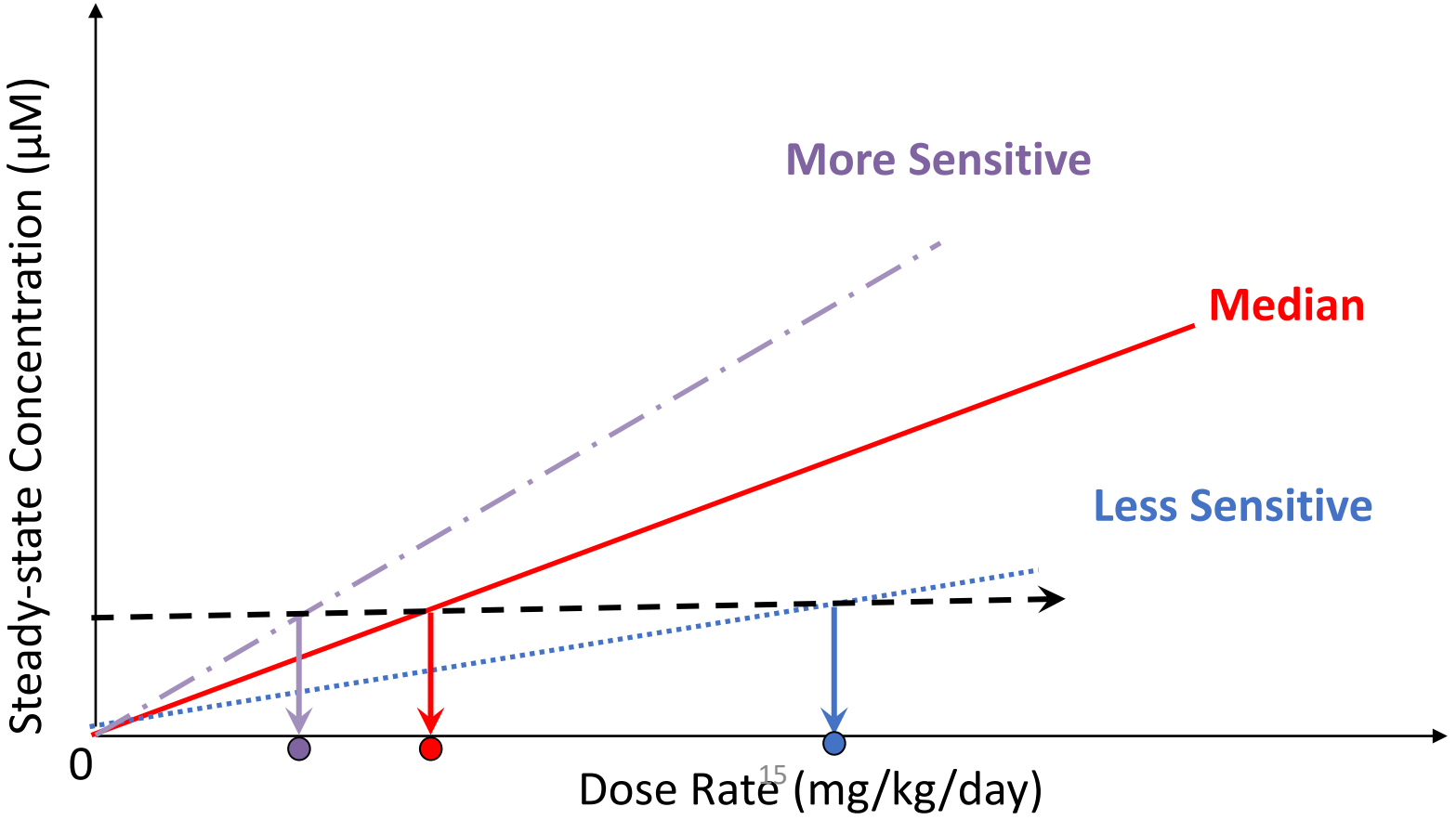
So far, we've shown IVIVE for an "average human".

But people aren't average: we all have different body weights, blood flow, kidney function, hepatic metabolism, etc.

Jamei et al. (2009); Wetmore et al. (2014); Ring et al. (2017)

Population variability in TK = population variability in equivalent dose

In other words – some people are more sensitive and may see effects at a lower dose compared to the “average person”. We don’t want to underestimate their potential risk!



Monte Carlo approach to simulating population variability in physiology: HTTK-Pop module within "httk" R package

Correlated Monte Carlo sampling of physiological model parameters

Sample NHANES measured quantities for actual individuals:

- Sex
- Race/ethnicity
- Age
- Height
- Weight
- Serum creatinine

Regression equations from literature (McNally *et al.*, 2014) (+ residual marginal variability)

(Similar approach used in SimCYP [Jamei et al. 2009], GastroPlus, PopGen [McNally et al. 2014], P3M [Price et al. 2003], physB [Bosgra et al. 2012], etc.)

Predict physiological TK quantities (as used by generic TK model) for each individual:

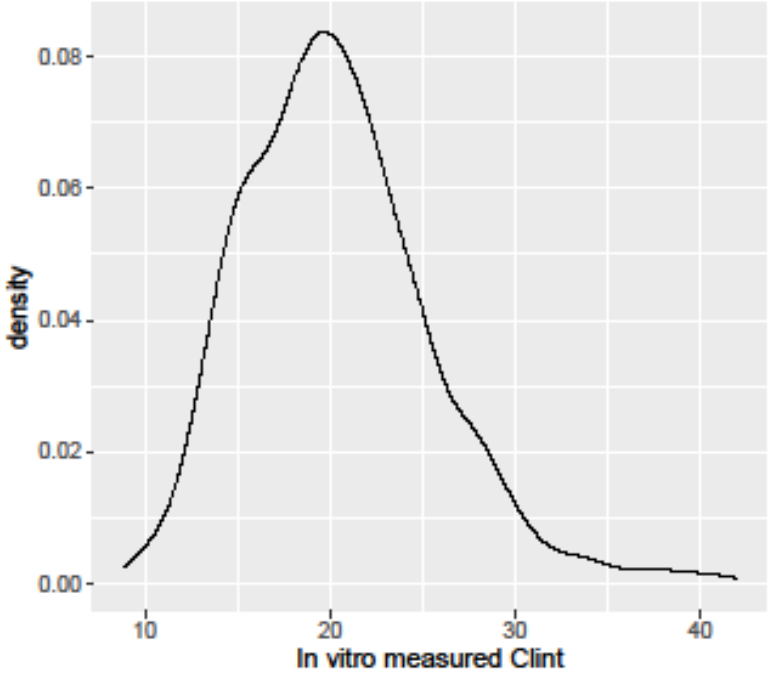
- Tissue masses
- Tissue blood flows
- GFR (kidney function)
- Hepatocellularity

NHANES: US CDC National Health and Nutrition Examination Survey

Ring *et al.* (2017)

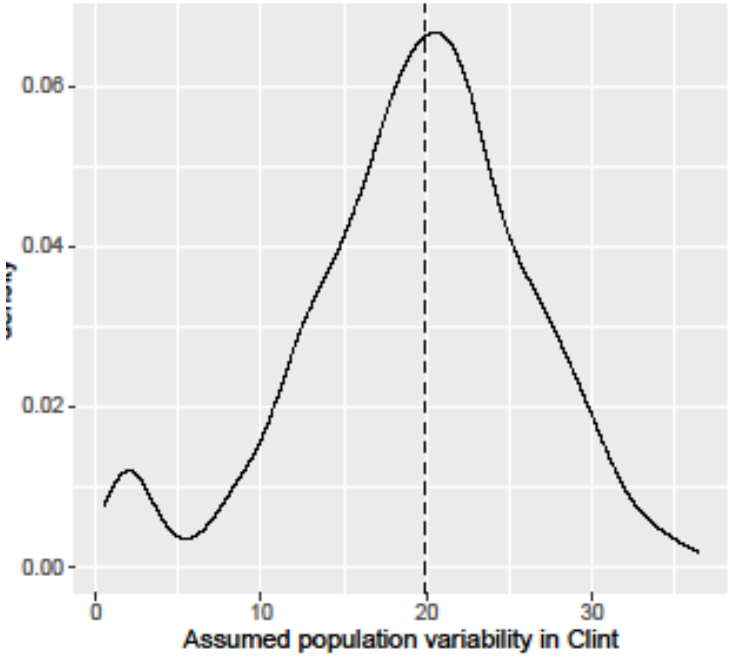
Monte Carlo approach to propagating both *uncertainty* and *variability* in chemical-specific TK parameters

Quantify uncertainty for *in vitro* measured value
Describe as distribution for each chemical



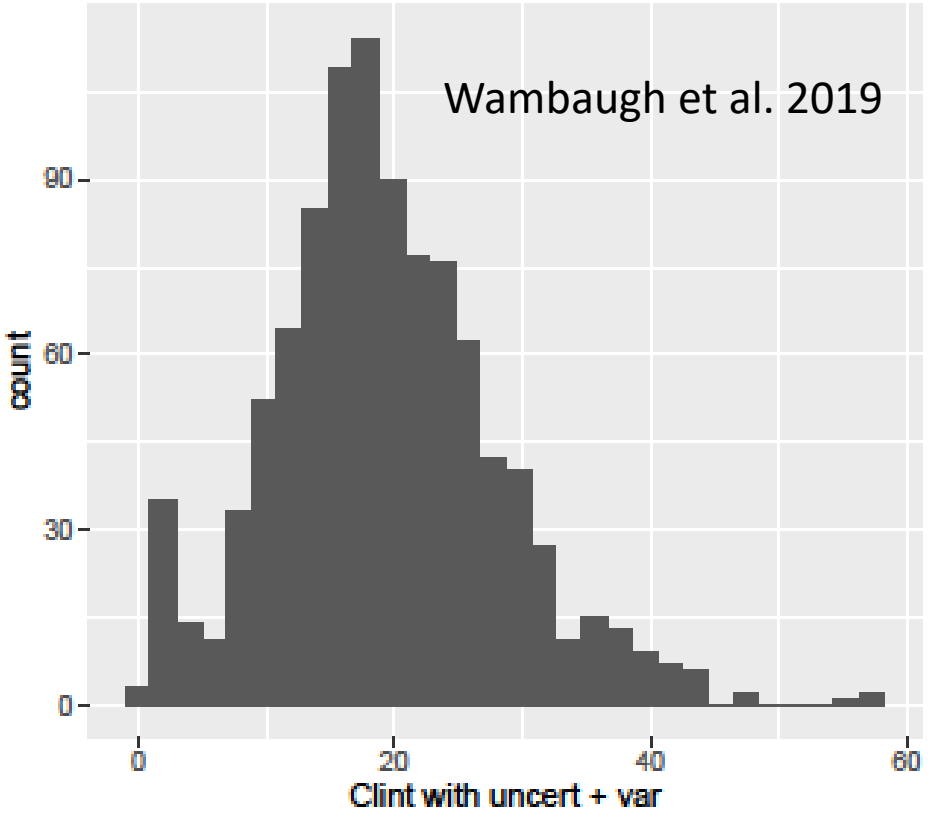
Wambaugh et al. 2019

Assume population variability around *in vitro* measured value



Ring et al. 2017

Two-stage Monte Carlo to get sampled values for each simulated individual that include both uncertainty & variability

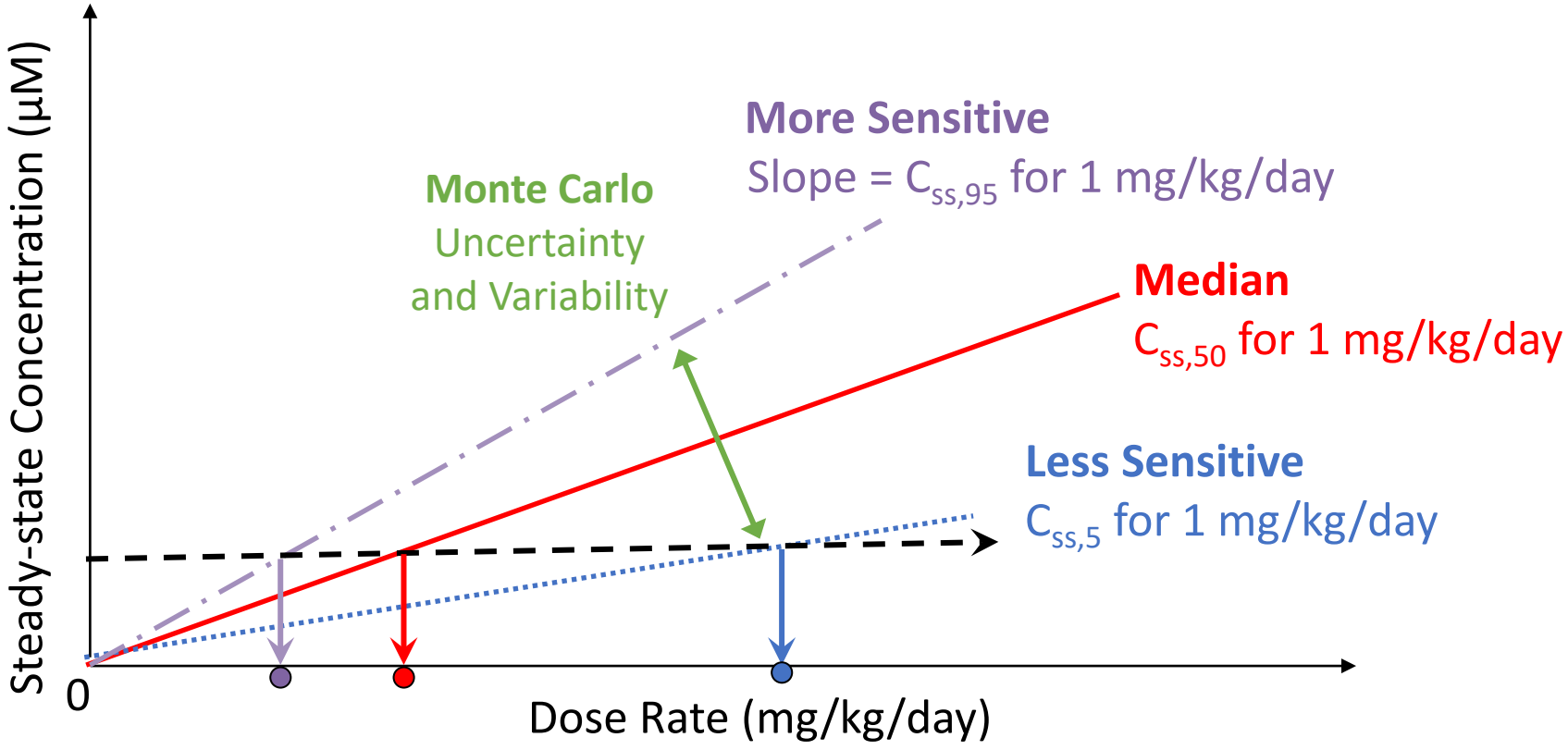


Note: This example is just a hypothetical illustration, not any particular chemical

HTTK-Pop lets us estimate equivalent dose for the more-sensitive portion of the population

The most-sensitive 5% of the population (the steeper, 95th percentile slope) has the lowest equivalent dose (see purple lines in this graphic) —

in other words, this is the level of exposure where we predict that a sensitive portion of the population would potentially see some effects.

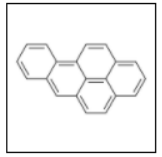


Example: Using httk to find an equivalent dose for a low-end ToxCast AC50 for benzo(a)pyrene

CompTox Chemicals Dashboard

comptox.epa.gov/dashboard/d

EPA United States Environmental Protection Agency | Home | Advanced Search | Batch Search | Lists | Predictions | Downloads | Copy | Share | Submit Comment | Search all data



Benzo(a)pyrene

50-32-8 | DTXSID2020139

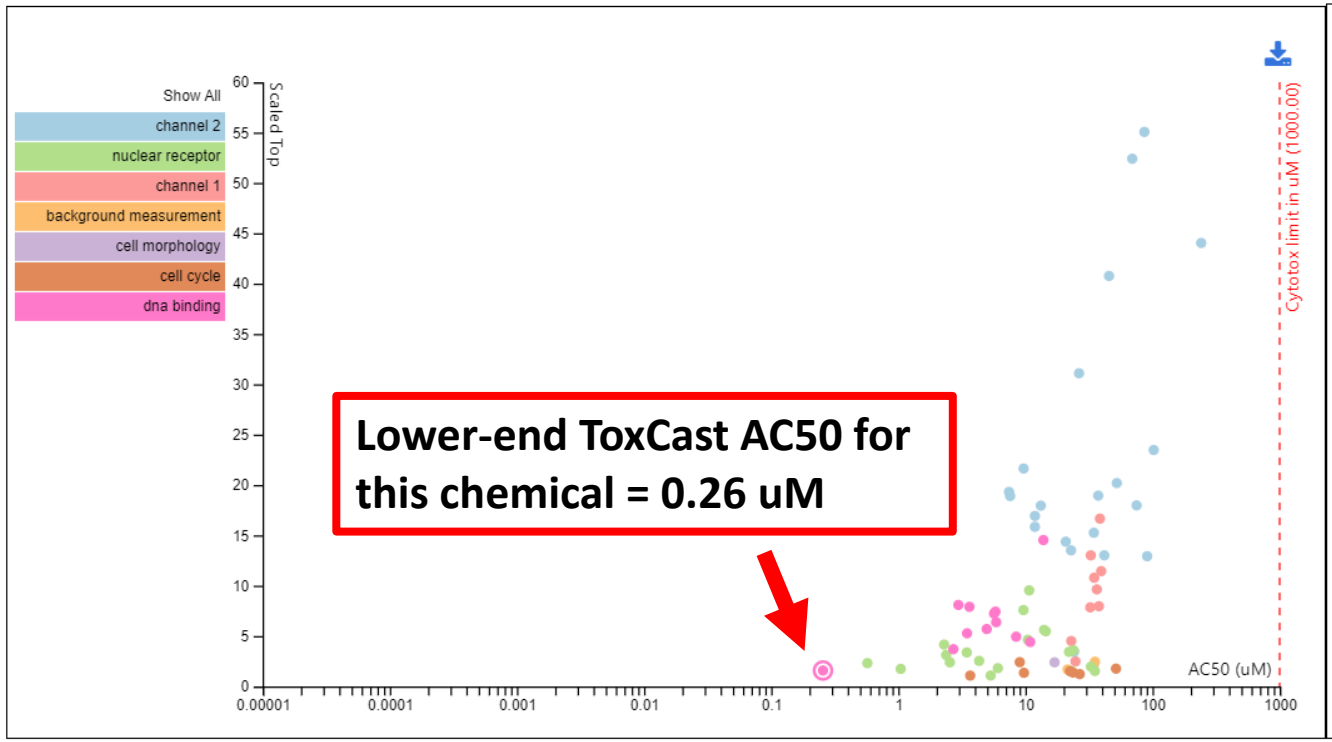
Searched by DSSTox Substance Id.

First: Get AC50 value. ToxCast AC50s can be found on the CompTox Chemicals Dashboard.

- DETAILS
- EXECUTIVE SUMMARY
- PROPERTIES
- ENV. FATE/TRANSPORT
- HAZARD
- SAFETY
- ADME
- EXPOSURE
- **BIOACTIVITY**
  - **TOXCAST: SUMMARY**
  - EDSP21
  - TOXCAST/TOX21
  - PUBCHEM
  - TOXCAST: MODELS
- SIMILAR COMPOUNDS

Chemical Activity Summary

TOXCAST DATA



Lower-end ToxCast AC50 for this chemical = 0.26 uM

ASSAY DETAILS

AC50 (uM): 0.26
Scaled top: 1.58
Assay Endpoint Name: TOX21_SSH_3T3_GLI3_Antagonist
Gene Symbol:
Organism: mouse
Tissue: embryo
Assay Format Type: cell-based
Biological Process Target: regulation of transcription factor activity
Detection Technology: Luciferase-coupled ATP quantitation
Analysis Direction: positive
Intended Target Family: dna binding
Description: Data from the assay component TOX21_SSH_3T3_GLI3_Antagonist was analyzed into 1 assay endpoint. This assay endpoint, TOX21_SSH_3T3_GLI3_Antagonist, was analyzed in the positive fitting direction relative to DMSO as the negative control and baseline of activity. Using a type of inducible reporter, loss-of-signal activity can be used to understand changes in the reporter gene as they relate to the gene GLI3. Furthermore, this assay endpoint can be referred to as a primary readout, because this assay has produced multiple assay endpoints where this one serves a reporter gene function.

To calculate population equivalent dose, use httk function `calc_mc_oral_equiv()`

```
> library(httk)
> set.seed(42)
> #Steady-state equivalent dose (mg/kg BW/day) to produce 0.26 uM in plasma:
 calc_mc_oral_equiv(conc=0.26,
                    chem.name="benzo(a)pyrene",
                    which.quantile = c(0.95, 0.5, 0.05),
                    input.units = "uM",
                    output.units = "mgpkgpday“)
uM concentration converted to mgpkgpday dose for 0.95 0.5 0.05 quantile.
     95%      50%       5%
0.003821 0.019090 0.067080
```

Let's break down this function call a little bit

```
> library(httk)
> set.seed(42)
> #Steady-state equivalent dose (mg/kg BW/day) to produce 0.26 uM in plasma:
 calc_mc_oral_equiv(conc=0.26,
                    chem.name="benzo(a)pyrene",
                    which.quantile = c(0.95, 0.5, 0.05),
                    input.units = "uM",
                    output.units = "mgpkgpday")
uM concentration converted to mgpkgpday dose for 0.95 0.5 0.05 quantile.
     95%      50%       5%
0.003821 0.019090 0.067080
```

First, load the httk package. (You only need to do this once per R session.)

Let's break down this function call a little bit

```
> library(httk)
> set.seed(42)
> #Steady-state equivalent dose (mg/kg BW/day) to produce 0.26 uM in plasma:
 calc_mc_oral_equiv(conc=0.26,
                    chem.name="benzo(a)pyrene",
                    which.quantile = c(0.95, 0.5, 0.05),
                    input.units = "uM",
                    output.units = "mgpkgpday")
uM concentration converted to mgpkgpday dose for 0.95 0.5 0.05 quantile.
      95%       50%        5%
0.003821 0.019090 0.067080
```

Set a seed for R's random number generator. This makes the Monte Carlo sampling **reproducible** (otherwise, we'd get a slightly different answer every time).

Let's break down this function call a little bit

```
> library(httk)
> set.seed(42)
> #Steady-state equivalent dose (mg/kg BW/day) to produce 0.26 uM in plasma:
 calc_mc_oral_equiv(conc=0.26,
                    chem.name="benzo(a)pyrene",
                    which.quantile = c(0.95, 0.5, 0.05),
                    input.units = "uM",
                    output.units = "mgpkgpday")
uM concentration converted to mgpkgpday dose for 0.95 0.5 0.05 quantile.
      95%       50%        5%
0.003821 0.019090 0.067080
```

Any line of R code starting with "#" is a comment (ignored & not executed by R)

Call the function calc_mc_oral_equiv()

Let’s break down this function call a little bit

```
> library(httk)
> set.seed(42)
> #Steady-state equivalent dose (mg/kg BW/day) to produce 0.26 uM in plasma:
calc_mc_oral_equiv(conc=0.26,
                   chem.name="benzo(a)pyrene",
                   which.quantile = c(0.95, 0.5, 0.05),
                   input.units = "uM",
                   output.units = "mgpkgpday“)
uM concentration converted to mgpkgpday dose for 0.95 0.5 0.05 quantile.
      95%       50%        5%
0.003821 0.019090 0.067080
```

Call the function calc_mc_oral_equiv() to actually do the Monte Carlo analysis

Let's break down this function call a little bit

```
> library(httk)
> set.seed(42)
> #Steady-state equivalent dose (mg/kg BW/day) to produce 0.26 uM in plasma:
 calc_mc_oral_equiv(conc=0.26,
                    chem.name="benzo(a)pyrene",
                    which.quantile = c(0.95, 0.5, 0.05),
                    input.units = "uM",
                    output.units = "mgpkgpday")
uM concentration converted to mgpkgpday dose for 0.95 0.5 0.05 quantile.
      95%       50%        5%
0.003821 0.019090 0.067080
```

Supply target Css (here, low-end ToxCast AC50 for benzo(a)pyrene)

Let's break down this function call a little bit

```
> library(httk)
> set.seed(42)
> #Steady-state equivalent dose (mg/kg BW/day) to produce 0.26 uM in plasma:
calc_mc_oral_equiv(conc=0.26,
                   chem.name="benzo(a)pyrene",
                   which.quantile = c(0.95, 0.5,
                   input.units = "uM",
                   output.units = "mgpkgpday")
uM concentration converted to mgpkgpday dose for 0.95 0.5 0.05 quantile.
     95%      50%       5%
0.003821 0.019090 0.067080
```

Supply chemical name (or use `chem.cas = …` to supply CASRN instead). This allows httk to look up its built-in *in vitro* TK data for this chemical.

Let's break down this function call a little bit

```
> library(httk)
> set.seed(42)
> #Steady-state equivalent dose (mg/kg BW/day) to produce 0.26 uM in plasma:
calc_mc_oral_equiv(conc=0.26,
                   chem.name="benzo(a)pyrene",
                   which.quantile = c(0.95, 0.5, 0.05),
                   input.units = "uM",
                   output.units = "mgpkgpday")
uM concentration converted to mgpkgpday dose for 0.95 0.5 0.05 quantile.
      95%       50%        5%
0.003821 0.019090 0.067080
```

Specify which quantiles of Css *slope* to calculate equivalent doses for (95th percentile *slope* = lower-end equivalent dose)

Let's break down this function call a little bit

```
> library(httk)
> set.seed(42)
> #Steady-state equivalent dose (mg/kg BW/day) to produce 0.26 uM in plasma:
calc_mc_oral_equiv(conc=0.26,
                   chem.name="benzo(a)pyrene",
                   which.quantile = c(0.95, 0.5, 0.05),
                   input.units = "uM",
                   output.units = "mgpkgpday")
uM concentration converted to mgpkgpday dose for 0.95 0.5 0.05 quantile.
      95%       50%        5%
0.003821 0.019090 0.067080
```

Optional: explicitly specify input units (for conc) and output units (for equivalent dose). If not specified, these are the defaults.

Let’s break down this function call a little bit

```
> library(httk)
> set.seed(42)
> #Steady-state equivalent dose (mg/kg BW/day) to produce 0.26 uM in plasma:
 calc_mc_oral_equiv(conc=0.26,
                    chem.name="benzo(a)pyrene",
                    which.quantile = c(0.95, 0.5, 0.05),
                    input.units = "uM",
                    output.units = "mgpkgpday“)
uM concentration converted to mgpkgpday dose for 0.95 0.5 0.05 quantile.
      95%       50%        5%
0.003821 0.019090 0.067080
```

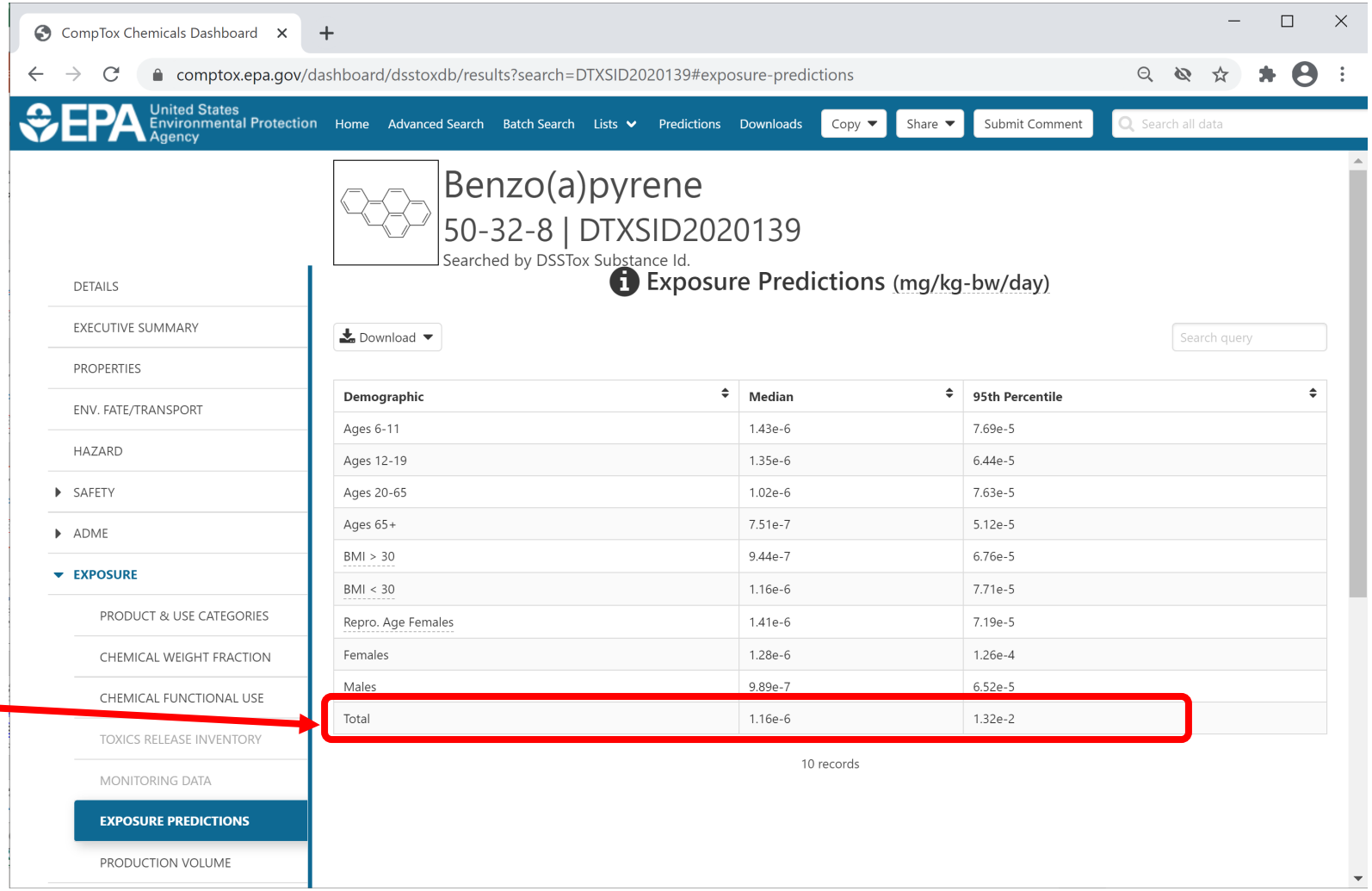
The function returns the results (plus some messages & warnings, which I’ve trimmed out to save space here).

Compare these results to HT exposure predictions available on EPA CompTox Chemicals Dashboard

Monte Carlo equivalent dose from `httk::calc_mc_oral_equiv()`:

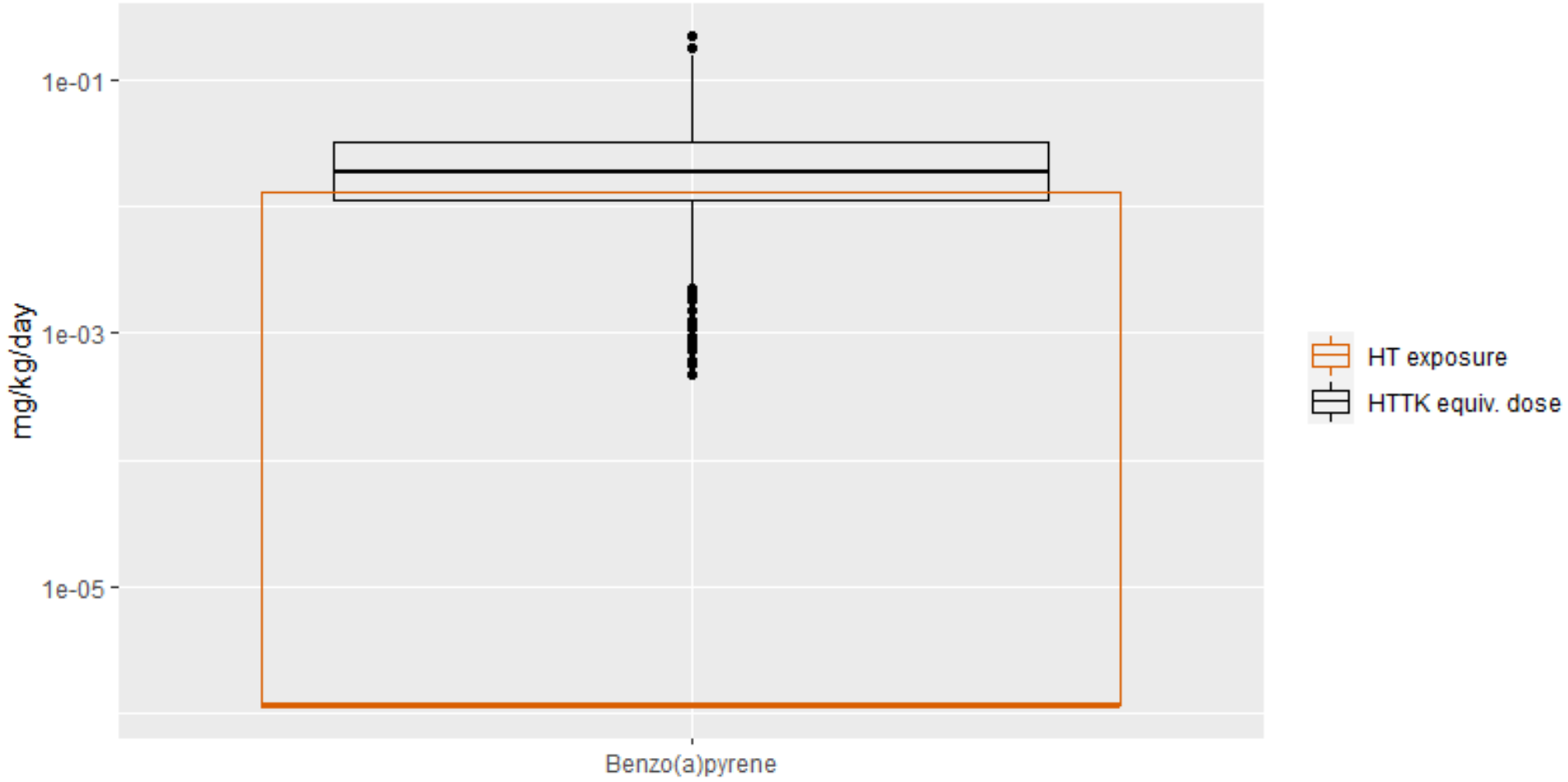
```
uM concentration converted to
mgpkgpday dose for 0.95 0.5
0.05 quantile.
      95%       50%        5%
0.003821 0.019090 0.067080
```

HT exposure predictions from Dashboard:
median = 1.16e-6;
upper bound on median = 1.32e-2
mg/kg/day



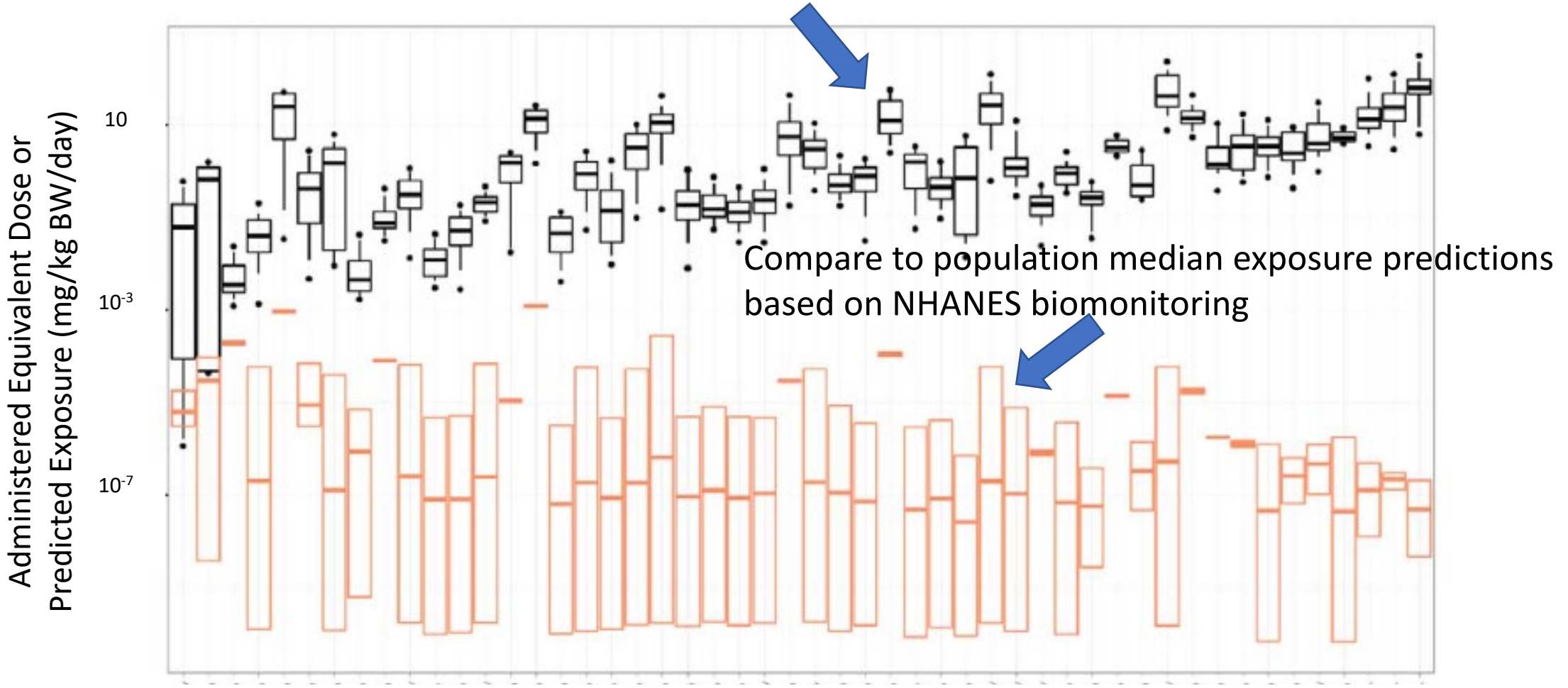
Ring et al. 2019, Wambaugh et al. 2014

Graphical comparison of HTTK-predicted equivalent dose for ToxCast AC50, vs. HT exposure prediction



Example: using HTTK for chemical prioritization

Equivalent doses for most-sensitive 5% of population for ToxCast AC50s



 Ring *et al*. (2017)

Other things you can do with “httk”: get population equivalent doses for a specific demographic (e.g. adults ages 65+)

```
> library(httk)
> set.seed(42)
> calc_mc_oral_equiv(conc=0.26, #lowest ToxCast AC50 in uM
                     chem.name="benzo(a)pyrene",
                     which.quantile = 0.95,
                     input.units = "uM",
                     output.units = "mgpkgpday",
                      httkpop.generate.arg.list = list(method = "direct
resampling",
                     agelim_years = c(65,80)
                     )
       )
uM concentration converted to mgpkgpday dose for 0.95 quantile.
     95%
0.001781
```

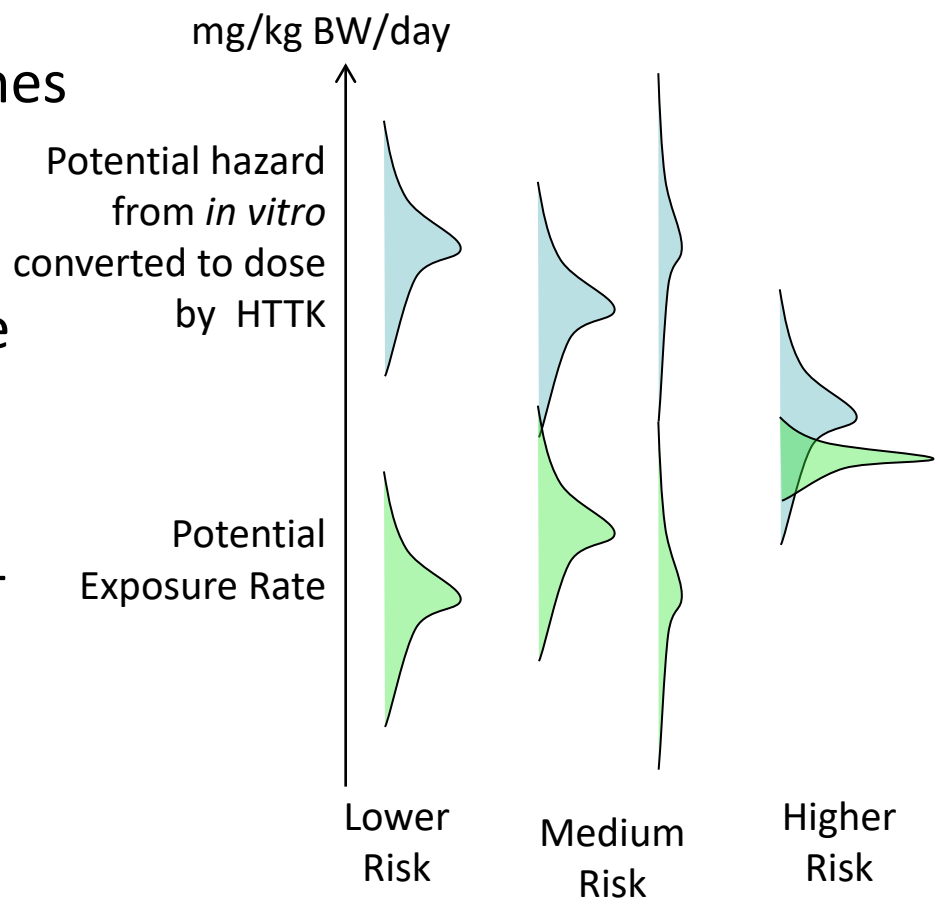
a named list of arguments that control the underlying population-simulation function, `httkpop_generate()`

Even more things you can do with httk

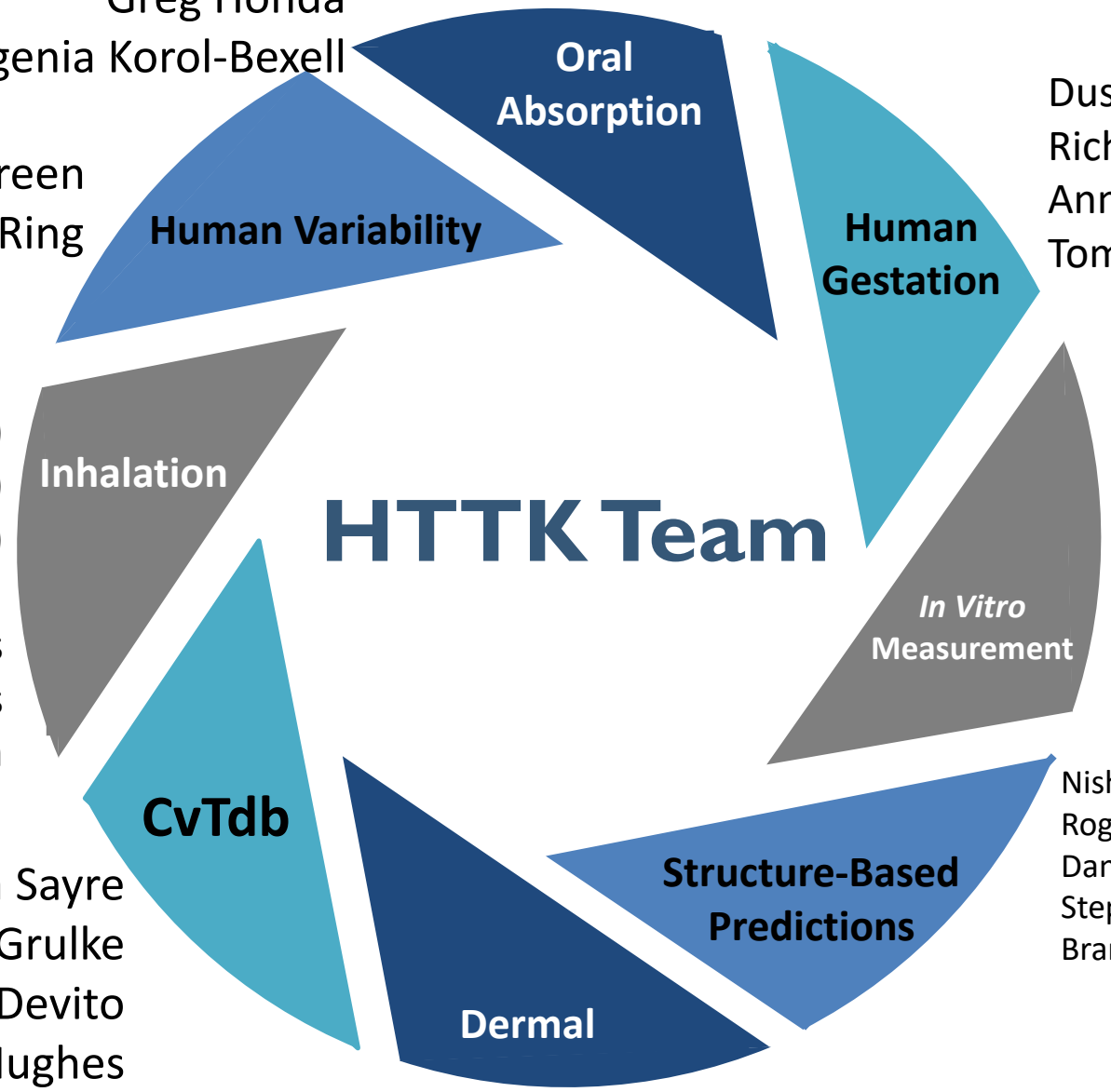
- Forward, time-dependent TK modeling with function `solve_model()`
  - Summary TK statistics (e.g. mean concentration, peak concentration, AUC) using function `calc_stats()`
- Add your own TK data for new chemicals, with function `add_chemtable()`
- Inter-species extrapolation of *in vivo* tox data from animal studies, using built-in TK data for various species (e.g. rat, mouse, dog, monkey, human) + combination of forward and reverse dosimetry
- Use HTTK-Pop module separately to generate a sample of population physiology, body measures, demographics for use in other modeling applications (e.g. population exposure models [East et al., 2020])

Summary

- We would like to know more about the risk posed by thousands of chemicals in the environment – which ones should we start with?
- We can use *in vitro* high-throughput screening (HTS) assays to fill data gaps when *in vivo* toxicology data are not available
- To extrapolate *in vitro* HTS data to equivalent *in vivo* doses, we use high-throughput toxicokinetics (HTTK) -- generic model that can be parameterized with *in vitro* data
- HTTK methods are available through the free, open source R package “httk”
- Simulating population variability and measurement uncertainty for TK parameters allows us to examine potential risk for potentially sensitive sub-populations



The views expressed in this presentation are those of the author and do not necessarily reflect the views or policies of the U.S. EPA



John Wambaugh
Principal Investigator

Mark Sfeir
Software Engineer

Greg Honda
Evgenia Korol-Bexell

Miyuki Breen
Caroline Ring

Dustin Kapraun
Richard Judson
Annie Lumen (FDA)
Tom Knudsen

David Ayres
Roger Dinallo
Janice Lau
Chris Strock

Anna Kreutz
Marci Smeltz
Barbara Wetmore
Lucas Albrecht

Matt Linakis (AFRL)
Heather Pangburn (AFRL)
Jeffery Gearhart (AFRL)
Nisha Sipes
Kristin Isaacs
Marina Evans
Miyuki Breen

Nisha Sipes
Rogelio Tornero-Velez
Daniel Dawson
Stephen Ferguson (NTP)
Brandall Ingle
Richard Judson
Jon Arnot (ARC)
Kamel Mansouri (ILS)
Michael Lawless (Simulations Plus)
Prachi Pradeep

Russell Thomas
Mike Devito
David Murphy
Katherine Coutros
Ann Richard
Tony Williams

Risa Sayre
Chris Grulke
Mike Devito
Mike Hughes

Marina Evans
Longjian Chen (Unilever)
Tom Moxon (Unilever)
Beate Nicol (Unilever)

Alumni

Robert Pearce
Woody Setzer
Greg Honda
Chantel Nicolas
Cory Strope
Jimena Davis

Funded by EPA's Office of Research and Development and Office of Science Coordination and Policy

Appendix:
Additional information

Modules within R Package "httk"

Feature	Description	Reference
Chemical Specific *In Vitro* Measurements	Metabolism and protein binding for ~1000 chemicals in human and ~200 in rat	Wetmore et al. (2012, 2013, 2015), plus others
Chemical-Specific *In Silico* Predictions	Metabolism and protein binding for ~8000 Tox21 chemicals	Sipes et al. (2017)
Generic toxicokinetic models	One compartment, three compartment, physiologically-based oral, intravenous, and inhalation (PBTK)	Pearce et al. (2017a), Linakis et al. (2020)
Tissue partition coefficient predictors	Modified Schmitt (2008) method	Pearce et al. (2017b)
Variability Simulator	Based on NHANES biometrics	Ring et al. (2017)
In Vitro Disposition	Armitage et al. (2014) model	Honda et al. (2019)
Uncertainty Propagation	Model parameters can be described by distributions reflecting uncertainty	Wambaugh et al. (2019)

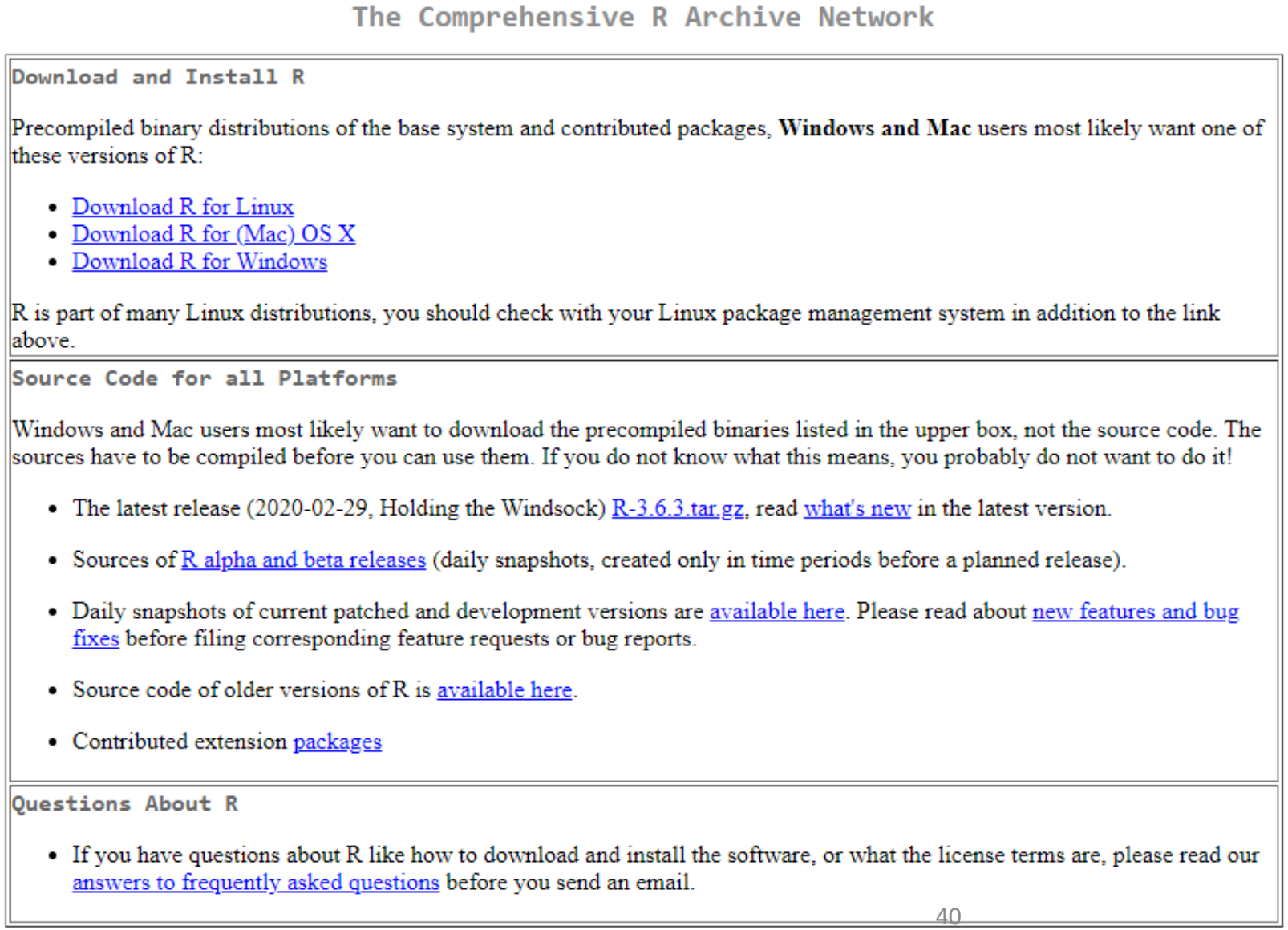
Setup: Getting R

- R is freely available from the Comprehensive R Archive Network (CRAN):
  https://cloud.r-project.org/

Available for Windows, Mac, Linux

- I like to use the RStudio integrated development environment (IDE) , which is also freely available:
  https://rstudio.com/

(but use of RStudio is optional – R comes with a basic GUI, or it can be used completely from the system command line)



The Comprehensive R Archive Network

Download and Install R

Precompiled binary distributions of the base system and contributed packages, **Windows and Mac** users most likely want one of these versions of R:

- Download R for Linux
- Download R for (Mac) OS X
- Download R for Windows

R is part of many Linux distributions, you should check with your Linux package management system in addition to the link above.

Source Code for all Platforms

Windows and Mac users most likely want to download the precompiled binaries listed in the upper box, not the source code. The sources have to be compiled before you can use them. If you do not know what this means, you probably do not want to do it!

- The latest release (2020-02-29, Holding the Windsock) R-3.6.3.tar.gz, read what's new in the latest version.
- Sources of R alpha and beta releases (daily snapshots, created only in time periods before a planned release).
- Daily snapshots of current patched and development versions are available here. Please read about new features and bug fixes before filing corresponding feature requests or bug reports.
- Source code of older versions of R is available here.
- Contributed extension packages

Questions About R

- If you have questions about R like how to download and install the software, or what the license terms are, please read our answers to frequently asked questions before you send an email.

Setup: Installing and loading “httk” package at the R command line

```
> install.packages("httk")
Installing package into ‘c:/Users/jwambaug/Rpackages’
(as ‘lib’ is unspecified)
--- Please select a CRAN mirror for use in this session ---
trying URL 'https://cloud.r-
project.org/bin/windows/contrib/3.6/httk_2.0.1.zip'
Content type 'application/zip' length 10127063 bytes (9.7 MB)
downloaded 9.7 MB

package ‘httk’ successfully unpacked and MD5 sums checked

The downloaded binary packages are in

C:\Users\jwambaug\AppData\Local\Temp\Rtmp4STebz\downloaded_packages
```

Install HTTK from the command line
(GUIs like RStudio also provide menus for this)

```
> library(httk)
```

Load the HTTK package: data, models, and functions

```
> packageVersion("httk")
[1] ‘2.0.1’
```

Check what version you are using

Q: How do I know which arguments to use for `httkpop.generate.arg.list` to specify my population demographics?

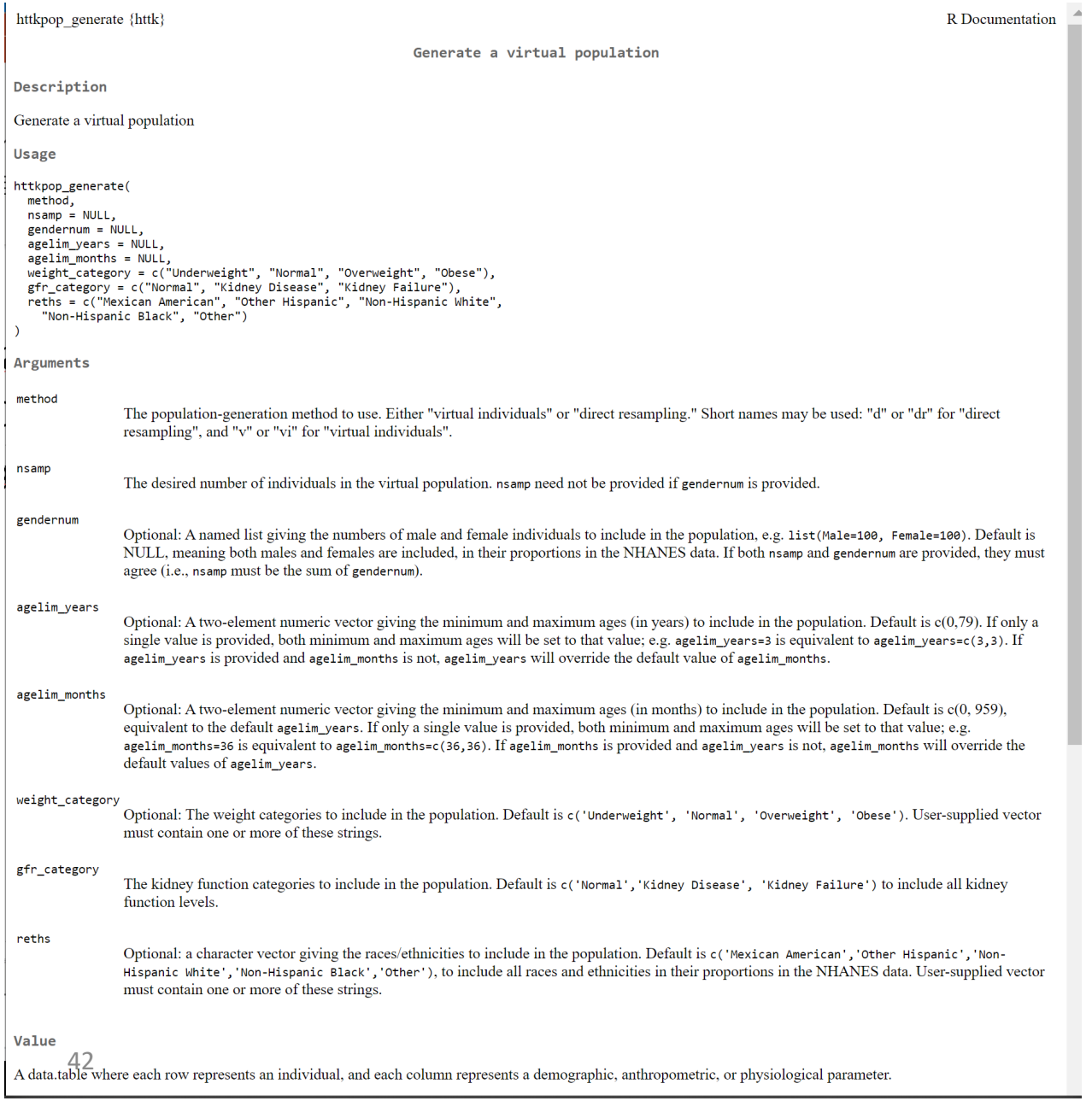
A: Look at the help for `httkpop_generate()`

At the R command line, type

```
> help(httkpop_generate)
```

You will see a detailed help page pop up, with explanations for each function argument, and (usually) some examples of how to use the function.

You can get help on any function this way.



httkpop_generate {httk} R Documentation

Generate a virtual population

Description

Generate a virtual population

Usage

```
httkpop_generate(
  method,
  nsamp = NULL,
  gendernum = NULL,
  agelim_years = NULL,
  agelim_months = NULL,
  weight_category = c("Underweight", "Normal", "Overweight", "Obese"),
  gfr_category = c("Normal", "Kidney Disease", "Kidney Failure"),
  reths = c("Mexican American", "Other Hispanic", "Non-Hispanic White",
    "Non-Hispanic Black", "Other")
)
```

Arguments

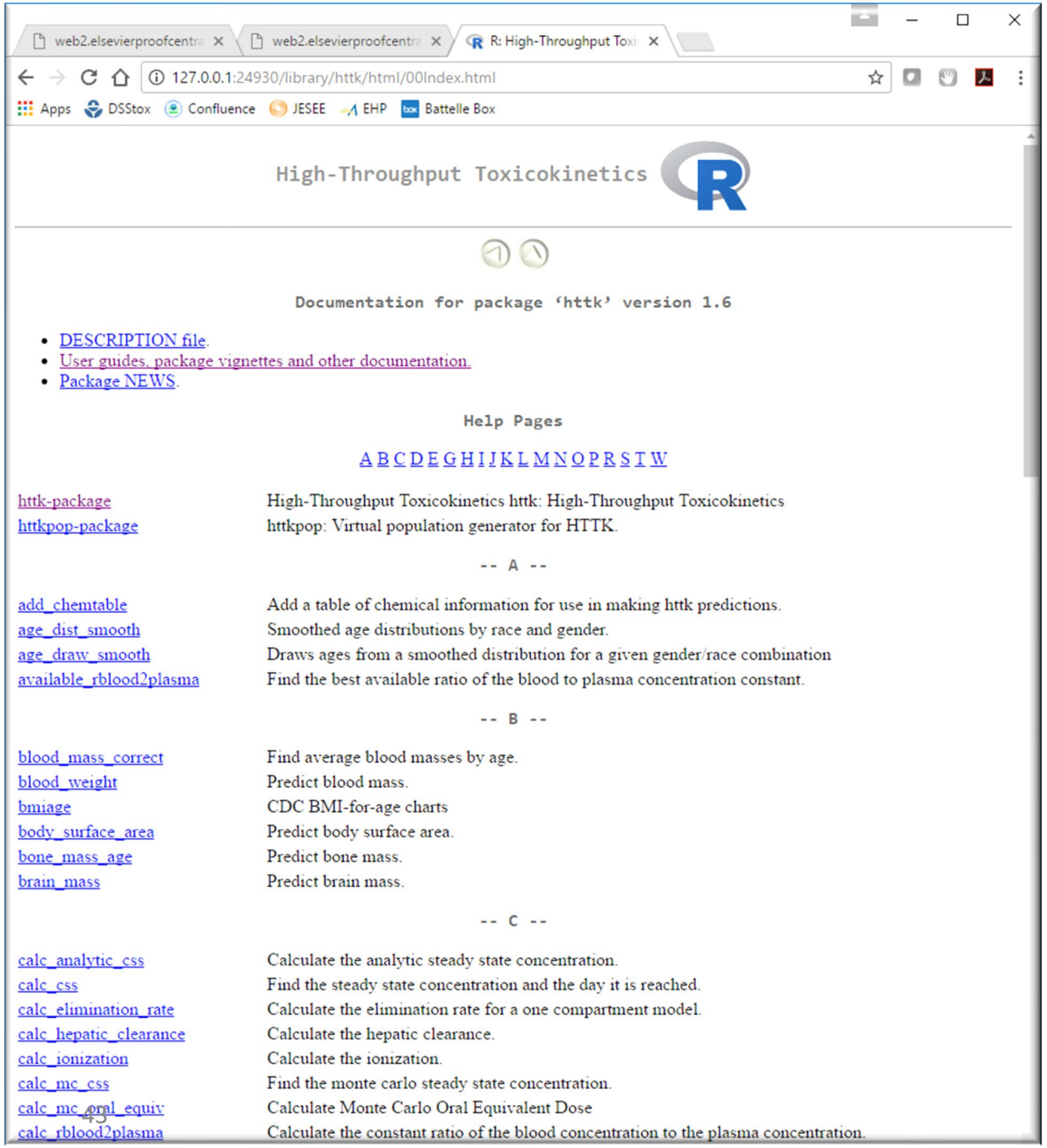
`method`	The population-generation method to use. Either "virtual individuals" or "direct resampling." Short names may be used: "d" or "dr" for "direct resampling", and "v" or "vi" for "virtual individuals".
`nsamp`	The desired number of individuals in the virtual population. `nsamp` need not be provided if `gendernum` is provided.
`gendernum`	Optional: A named list giving the numbers of male and female individuals to include in the population, e.g. `list(Male=100, Female=100)`. Default is NULL, meaning both males and females are included, in their proportions in the NHANES data. If both `nsamp` and `gendernum` are provided, they must agree (i.e., `nsamp` must be the sum of `gendernum`).
`agelim_years`	Optional: A two-element numeric vector giving the minimum and maximum ages (in years) to include in the population. Default is c(0,79). If only a single value is provided, both minimum and maximum ages will be set to that value; e.g. `agelim_years=3` is equivalent to `agelim_years=c(3,3)`. If `agelim_years` is provided and `agelim_months` is not, `agelim_years` will override the default value of `agelim_months`.
`agelim_months`	Optional: A two-element numeric vector giving the minimum and maximum ages (in months) to include in the population. Default is c(0, 959), equivalent to the default `agelim_years`. If only a single value is provided, both minimum and maximum ages will be set to that value; e.g. `agelim_months=36` is equivalent to `agelim_months=c(36,36)`. If `agelim_months` is provided and `agelim_years` is not, `agelim_months` will override the default values of `agelim_years`.
`weight_category`	Optional: The weight categories to include in the population. Default is `c('Underweight', 'Normal', 'Overweight', 'Obese')`. User-supplied vector must contain one or more of these strings.
`gfr_category`	The kidney function categories to include in the population. Default is `c('Normal','Kidney Disease', 'Kidney Failure')` to include all kidney function levels.
`reths`	Optional: a character vector giving the races/ethnicities to include in the population. Default is `c('Mexican American','Other Hispanic','Non-Hispanic White','Non-Hispanic Black','Other')`, to include all races and ethnicities in their proportions in the NHANES data. User-supplied vector must contain one or more of these strings.

Value

A data.table where each row represents an individual, and each column represents a demographic, anthropometric, or physiological parameter.

You can see the index of help files for *all* the functions in the httk package by typing at the R command line

```
> help(package="httk")
```



How do I find out which chemicals have sufficient built-in chemical-specific HTTK data to run the model?

```
> library(httk)
```

List all CAS numbers for all chemicals with sufficient data

```
> get_cheminfo()
  [1] "2971-36-0"   "94-75-7"     "94-82-6"     "90-43-7"     "1007-28-9"
  [6] "71751-41-2"  "30560-19-1"  "135410-20-7" "34256-82-1"  "50594-66-6"
 [11] "15972-60-8"  "116-06-3"    "834-12-8"    "33089-61-1"  "101-05-3"
 [16] "1912-24-9"   "86-50-0"     "131860-33-8" "22781-23-3"  "1861-40-1" …
```

Is a chemical available?

```
> "80-05-7" %in% get_cheminfo()
[1] TRUE
```

All data on chemicals A, B, C

```
subset(get_cheminfo(info="all"),Compound %in% c("A","B","C"))
```

```
> get_cheminfo(info="all")
```

List all information

Compound	CAS	logP	pKa Accept	pKa Donor	MW	Human Clint	Human Clint pValue	Human Funbound plasma	DSSTox Substance Id	Formula	Substance Type
2,4-d	94-75-7	2.81	<NA>	2.81	221.03	0	0.149	0.04	DTXSID0020442	C8H6Cl2O3	Single Compound
2,4-db	94-82-6	3.53	<NA>	4.5	249.09	0	0.104	0.01	DTXSID7024035	C10H10Cl2O3	Single Compound
2-phenylphenol	90-43-7	3.09	<NA>	10.6	170.211	2.08	0.164	0.04	DTXSID2021151	C12H10O	Single Compound
6-desisopropylatrazine	1007-28-9	1.15	1.59	<NA>	173.6	0	0.539	0.46	DTXSID0037495	C5H8ClN5	Single Compound

If my chemical doesn't have built-in *in vitro* information, how can I check whether it has built-in *in silico* information?

httk package includes a table of human Clint and Fup values for 8758 chemicals, predicted *in silico* using Simulations Plus ADMET Predictor software (Sipes et al. 2017).

```
> library(httk)
> origlist <- get_cheminfo()
> length(origlist) #number of chems with in vitro TK data
[1] 987
> load_sipes2017() #adds in silico data to built-in TK data set
Loading predictions from Sipes et al. (2017) for 8758 chemicals.
Existing data are not being overwritten. Please wait...
> newlist <- get_cheminfo()
> length(newlist) #number of chems with in vitro OR in silico TK data
[1] 8797
```

Now you can query the `get_cheminfo()` function the same way as on the previous slide

It will now include the *in silico* data as well as the *in vitro* data

If I need to “bring my own” chemical-specific data for a chemical that doesn’t have data built into the httk package, how do I do that?

1. Create a data frame identifying chemicals by (at least) CASRN, containing data on (at least) log P (octanol-water partitioning coefficient), molecular weight, fraction unbound in plasma, and intrinsic hepatic clearance rate. (For example, you may have made your own *in vitro* measurements in-house, used *in silico*/QSAR models to predict these quantities, etc.)
2. Use the `httk` function `add_chemtable()` to add your data frame to `httk`’s built-in table of chemical-specific information. (This only affects your current, local R session – it will need to be re-done every time you restart R.) Type `help(add_chemtable)` to see details on how to use this function.
3. Call TK modeling functions like `calc_mc_oral_equiv()` as usual – `httk` will use the new chemical-specific info you provided.

Example R code for HT reverse dosimetry for multiple chemicals & *in vitro* HTS assays

```
#Assume ac50 = data frame storing in vitro AC50s listed by chemical CASRN &
assay, with columns “CASRN”, “assay”, and “AC50”
equiv_doses <- sapply(1:nrows(ac50), #loop over rows of ac50 data frame
       function(n){ #apply the following function to row n:
            if(chem %in% get_cheminfo()){ #if chemical has TK info
sufficient to run model
                  return(calc_mc_oral_equiv(conc = ac50[n]$AC50,
                                            chem.cas = ac50[n]$CASRN,
                                            which.quantile = 0.95)
                         )
            }else{ #if no TK info, can’t run model, so return NA
                  return(NA_real_)
            } #end if/else block
      } #end function to apply to row n of ac50 data frame
) #end loop over rows of ac50 data frame
```

References

1. National Research Council 2007. Toxicity Testing in the 21st Century: A Vision and a Strategy. Washington, DC: The National Academies Press. https://doi.org/10.17226/11970
2. Bell SM, Chang X, Wambaugh JF, et al. In vitro to in vivo extrapolation for high throughput prioritization and decision making. Toxicology in Vitro. 2018 2018/03/01/;47:213-227.
3. Bessems JG, Loizou G, Krishnan K, et al. PBTK modelling platforms and parameter estimation tools to enable animal-free risk assessment: recommendations from a joint EPAA–EURL ECVAM ADME workshop. Regulatory Toxicology and Pharmacology. 2014;68(1):119-139.
4. Schmidt CW. TOX 21: new dimensions of toxicity testing. National Institute of Environmental Health Sciences; 2009.
5. Dix DJ, Houck KA, Martin MT, et al. The ToxCast program for prioritizing toxicity testing of environmental chemicals. Toxicological Sciences. 2007;95(1):5-12.
6. Kavlock RJ, Bahadori T, Barton-Maclaren TS, et al. Accelerating the Pace of Chemical Risk Assessment. Chemical Research in Toxicology. 2018 2018/05/21;31(5):287-290.
7. Rotroff DM, Wetmore BA, Dix DJ, et al. Incorporating human dosimetry and exposure into high-throughput in vitro toxicity screening. Toxicological Sciences. 2010;117(2):348-358
8. Wetmore BA, Wambaugh JF, Allen B, et al. Incorporating High-Throughput Exposure Predictions With Dosimetry-Adjusted In Vitro Bioactivity to Inform Chemical Toxicity Testing. Toxicological Sciences. 2015 Nov;148(1):121-36
9. Tan Y-M, Liao KH, Clewell HJ. Reverse dosimetry: interpreting trihalomethanes biomonitoring data using physiologically based pharmacokinetic modeling. Journal of Exposure Science and Environmental Epidemiology. 2007;17(7):591-603.
10. Campbell JL, Clewell RA, Gentry PR, et al. Physiologically based pharmacokinetic/toxicokinetic modeling. Computational Toxicology: Volume I. 2012:439-499.

11. Wetmore BA, Wambaugh JF, Ferguson SS, et al. Integration of dosimetry, exposure, and high-throughput screening data in chemical toxicity assessment. Toxicological Sciences. 2012 Jan;125(1):157-74.
12. Wetmore BA. Quantitative in vitro-to-in vivo extrapolation in a high-throughput environment. Toxicology. 2015;332:94-101.
13. Pradeep P, Patlewicz G, Pearce R, et al. Using chemical structure information to develop predictive models for in vitro toxicokinetic parameters to inform high-throughput risk-assessment. Computational Toxicology. 2020;16:100136.
14. Mansouri K, Grulke CM, Judson RS, et al. OPERA models for predicting physicochemical properties and environmental fate endpoints. Journal of Cheminformatics. 2018;10(1):10.
15. Ingle BL, Veber BC, Nichols JW, et al. Informing the Human Plasma Protein Binding of Environmental Chemicals by Machine Learning in the Pharmaceutical Space: Applicability Domain and Limits of Predictability. Journal of Chemical Information and Modeling. 2016;56(11):2243-2252.
16. Sipes NS, Wambaugh JF, Pearce R, et al. An Intuitive Approach for Predicting Potential Human Health Risk with the Tox21 10k Library. Environmental Science & Technology. 2017 2017/09/19;51(18):10786-10796.
17. Vilar S, Cozza G, Moro S. Medicinal chemistry and the molecular operating environment (MOE): application of QSAR and molecular docking to drug discovery. Current topics in medicinal chemistry. 2008;8(18):1555-1572
18. Yin Y, Chang D, Grulke C, et al. Essential Set of Molecular Descriptors for ADME Prediction in Drug and Environmental Chemical Space. Research. 2014 08/22;1.
19. Pearce RG, Setzer RW, Strope CL, et al. Httk: R package for high-throughput toxicokinetics. Journal of Statistical Software. 2017;79(1):1-26.
20. Jamei M, Marciniak S, Feng K, et al. The Simcyp® population-based ADME simulator. Expert Opinion on Drug Metabolism & Toxicology. 2009;5(2):211-223.

21. Wetmore BA, Allen B, Clewell HJ, 3rd, et al. Incorporating population variability and susceptible subpopulations into dosimetry for high-throughput toxicity testing. Toxicological Sciences. 2014 Nov;142(1):210-24.
22. Ring CL, Pearce RG, Setzer RW, et al. Identifying populations sensitive to environmental chemicals by simulating toxicokinetic variability. Environment International. 2017 2017/09/01/;106:105-118.
23. McNally, K., et al. (2014). "PopGen: A virtual human population generator." Toxicology **315**: 70-85.
24. Price, P. S., et al. (2003). "Modeling Interindividual Variation in Physiological Factors Used in PBPK Models of Humans." Critical Reviews in Toxicology **33**(5): 469-503.
25. Bosgra, S., et al. (2012). "An improved model to predict physiologically based model parameters and their inter-individual variability from anthropometry." Crit Rev Toxicol **42**(9): 751-767.
26. Wambaugh, J. F., et al. (2019). "Assessing Toxicokinetic Uncertainty and Variability in Risk Prioritization." Toxicol Sci **172**(2): 235-251.
27. Ring, C. L., et al. (2019). "Consensus Modeling of Median Chemical Intake for the U.S. Population Based on Predictions of Exposure Pathways." Environ Sci Technol **53**(2): 719-732.
28. Wambaugh, J. F., et al. (2014). "High throughput heuristics for prioritizing human exposure to environmental chemicals." Environ Sci Technol **48**(21): 12760-12767.
29. East et al. 2020 (in prep). The Residential Population Generator(RPGen): A tool to parameterize residential, demographic, and physiological data to model intraindividual exposure, dose, and risk. https://github.com/HumanExposure/RPGen

30. Linakis, M. W., et al. (2020). "Development and Evaluation of a High Throughput Inhalation Model for Organic Chemicals " Journal of Exposure Science & Environmental Epidemiology.

31. Pearce, R. G., et al. (2017). "Evaluation and calibration of high-throughput predictions of chemical distribution to tissues." Journal of Pharmacokinetics and Pharmacodynamics **44**(6): 549-565.

32. Honda, G. S., et al. (2019). "Using the concordance of in vitro and in vivo data to evaluate extrapolation assumptions." PLOS ONE **14**(5): e0217564