Supplementary Information for

Simulating Toxicokinetic Variability to Identify Susceptible and Highly Exposed Populations

Miyuki Breen[1], John F Wambaugh[1], Amanda Bernstein[2], Mark Sfeir[3], Caroline L Ring[1*]

[1] Center for Computational Toxicology and Exposure, US Environmental Protection Agency, Research Triangle Park, NC, USA

[2] Oak Ridge Institute for Science and Education (ORISE) fellow at the Center for Public Health and Environmental Assessment, Research Triangle Park, NC, USA

[3] Oak Ridge Institute for Science and Education (ORISE) fellow at the Center for Computational Toxicology and Exposure, Research Triangle Park, NC, USA

[*]Corresponding Author:
Caroline L Ring
US EPA, Center for Computational Toxicology and Exposure
109 T.W. Alexander Dr.
Research Triangle Park, NC, 27711, U.S.A.
Email: Ring.Caroline@epa.gov
Tel: 919-541-2519

SUPPLEMENTAL MATERIAL

When analyzing the uncertainty of *in vitro* TK measurements, there can be key differences. Some chemical data are from measurements performed by the U.S. EPA, its collaborators, and its contractors, while other data are obtained from the peer-reviewed literature. Regardless of the source, for some chemical measurements only a "point" estimate (most likely value) is available while in other cases confidence intervals are available (indicated by the suffix ".dist" referring to distributions). When only a point estimate is available a default coefficient of variation is assumed (1), while if a confidence interval is available a distribution with matching quantiles (median, lower-, and upper-95th percentile) is used.

For f_{up}, the influence of the measured value on uncertainty is related to whether the measured value for free chemical in plasma was above the limit of detection. If no free fraction was detected, a range of values less than the limit of detection are simulated using a uniform distribution between min. f_{up} (typically 10^{-5}) and the limit of detection (typically 0.01). If, for a f_{up} measurement with a reported distribution, only the upper 95th limit is above the limit of detection then the function "rmed0non0u95" is used to simulate a distribution with a median of zero and a non-zero upper 95th limit. For cases above the limit of detection a beta distribution (which returns results between zero and one) is used, where the parameters have been found to be consistent with the median and 95% interval limits. When no distribution is available the 95% interval is calculated using +-1.96 coefficient of variation (typically 0.4) * f_{up} with truncation at 0 and 1. As depicted in Supplemental Figure S1, all values of f_{up} are adjusted for lipid binding by the method described by Pearce *et al.* (2) unless the option "adjusted.funbound.plasma" is set to FALSE. Population variability for f_{up} is simulated using a normal distribution truncated to a minimum of min. f_{up} and a maximum of 1 (Supplemental Figure S2).

As with f_{up}, if only a point estimate is available for Cl_{int} then a normal distribution is assumed to calculate a lower and upper limit for a 95% interval centered on the reported value. Then we check to see if both the median and upper 95th percentiles are above zero; if both are zero then all values of Cl_{int} are set to zero. If only the upper 95th percentile is above zero then the function "rmed0non0u95" is used. Otherwise, draws are made from a log-normal distribution (which is, by definition, constrained to produce positive values) using parameters that match the quantiles. Finally, all values of Cl_{int} are adjusted (if adjusted. Cl_{int} is set to TRUE) according to the Kilford *et al.* (3) correction for free fraction of chemical within the hepatocyte clearance assay.

REFERENCES

References

1. Wambaugh JF, Wetmore BA, Ring CL, Nicolas CI, Pearce RG, Honda GS, et al. Assessing Toxicokinetic Uncertainty and Variability in Risk Prioritization. Toxicol Sci. 2019;172(2):235-51.

2. Pearce RG, Setzer RW, Strope CL, Sipes NS, Wambaugh JF. Httk: R package for high-throughput toxicokinetics. Journal of Statistical Software. 2017;79(1):1-26.

3. Kilford PJ, Gertz M, Houston JB, Galetin A. Hepatocellular binding of drugs: correction for unbound fraction in hepatocyte incubations using microsomal binding or drug lipophilicity data. Drug Metab Dispos. 2008;36(7):1194-7.

FIGURE LEGENDS

Figure S1. Monte Carlo Uncertainty Simulation for Fraction Unbound in Plasma (f_{up}).

Figure S2. Special Considerations for using optim and Beta Distributions when median Fup ~ 1.

Figure S3. Monte Carlo Variability Simulation for Fraction Unbound in Plasma (f_{up}).

Figure S4. Monte Carlo Uncertainty Simulation for Intrinsic Hepatic Clearance (Cl_{int}).

Figure S5. Monte Carlo Variability Simulation for Intrinsic Hepatic Clearance (Cl_{int}).

Figure S6. Upper panel: Histogram of intrinsic hepatic clearance (Cl_{int}) for chemicals whose percent change in equivalent dose (as in Figure 3B) was more negative than -10 % (i.e., the lower peak in Figure 3B). Lower panel: Histogram of intrinsic hepatic clearance (Cl_{int}) for chemicals whose percent change in equivalent dose (as in Figure 3) was less negative than -10 % (i.e., the upper peak in Figure 3B). If Cl_{int} was provided as a distribution in chem.physical_and_invitro.data, its median is plotted here.

Figure S7. The scatter plots of weights versus age of the previous cohort (blue) and updated cohort (red) for subgroups: Total, Male, Female.

Figure S8. The scatter plots of heights versus age of the previous cohort (blue) and updated cohort (red) for subgroups: Total, Male, Female.

Figure S9. The scatter plots of weights versus heights of the previous cohort (blue) and updated cohort (red) for subgroups: Total, Male, Female.

TABLES

Compound	Abbrev.	CAS-RN	DTXSID
2, 4-D	2, 4D	94-75-7	DTXSID0020442
Alachlor	Alac	15972-60-8	DTXSID1022265
Alprazolam	Alpr	28981-97-7	DTXSID4022577
Antipyrine	Anti	60-80-0	DTXSID6021117
Bensulide	Bens	741-58-2	DTXSID9032329
Bisphenol A	BPA	80-05-7	DTXSID7020182
Boscalid	Bosc	188425-85-6	DTXSID6034392
Bosentan	Bose	147536-97-8	DTXSID7046627
Carbaryl	Cbyl	63-25-2	DTXSID9020247
Carbendazim	Cbzm	10605-21-7	DTXSID4024729
Chloridazon	Cdzn	1698-60-8	DTXSID3034872
Chlorpyrifos	Cpfs	2921-88-2	DTXSID4020458
Cyclanilide	Cycl	113136-77-9	DTXSID5032600
Cyclosporin A	CycA	59865-13-3	DTXSID0020365
Diazinon-o-analog	Diaz	962-58-3	DTXSID5037523
Diclofenac	Dicl	15307-86-5	DTXSID6022923
Diltiazem	Dilt	42399-41-7	DTXSID9022940
Dimethenamid	Dime	87674-68-8	DTXSID4032376
Etoxazole	Etox	153233-91-1	DTXSID8034586
Fenarimol	Fena	60168-88-9	DTXSID2032390
Flufenacet	Fluf	142459-58-3	DTXSID2032552
Formetanate hydrochloride	Form	23422-53-9	DTXSID4032405
Hexobarbitone	Hexo	56-29-1	DTXSID9023122

Ibuprofen	Ibup	15687-27-1	DTXSID5020732
Imazalil	Imaz	35554-44-0	DTXSID8024151
Imidacloprid	Imid	138261-41-3	DTXSID5032442
Imipramine	Imip	50-49-7	DTXSID1043881
Metoprolol	Meto	51384-51-1	DTXSID2023309
Midazolam	Mida	59467-70-8	DTXSID5023320
Nilvadipine	Nilv	75530-68-6	DTXSID2046624
Novaluron	Nova	116714-46-6	DTXSID5034773
Ondansetron	Onda	99614-02-5	DTXSID8023393
Perfluorooctanoic acid	PFOA	335-67-1	DTXSID8031865
Permethrin	Perm	52645-53-1	DTXSID8022292
Phenacetin	Pacn	62-44-2	DTXSID1021116
Phenytoin	Pytn	57-41-0	DTXSID8020541
Propamocarb hydrochloride	Prop	25606-41-1	DTXSID6034849
Propyzamide	Prpy	23950-58-5	DTXSID2020420
Pyrithiobac sodium	Pyri	123343-16-8	DTXSID8032673
Resmethrin	Resm	10453-86-8	DTXSID7022253
S-Bioallethrin	S-Bi	28434-00-6	DTXSID2039336
Simazine	Sima	122-34-9	DTXSID4021268
Tolbutamide	Tolb	64-77-7	DTXSID8021359
Triclosan	Tric	3380-34-5	DTXSID5032498
Valproic acid	Valp	99-66-1	DTXSID6023733

Table S1. Pharmaceutical and non-pharmaceutical compounds used for the analysis with chemical names, abbreviations, CAS-RN, and DTXSID

httk function	Descripton	Key Arguments
calc_mc_css	Monte Carlo steady state plasma concentration for 1 mg/kg/day	chemical identity
calc_mc_tk	Monte Carlo PBPTK simulations	chemical identity, dose
create_mc_samples	Overall function for httk uncertainty and variability simulation via Monte Carlo	
parameterize_[MODEL]	Generate chemcial-specific parameters for [MODEL]	chemical identity
monte_carlo	Perform Monte Carlo variation of parameters by fixed coefficients of variation (cv)	which parameters are to be varied
httkpop_mc	Perform Monte Carlo using co-varying population biometrics	demographic descriptors
httkpop_generate	Generate biometrics for population of individuals consistent with requested demographics from NHANES	demographic descriptors

httkpop_biotophys_default	Convert biometrics to general httk model parameters	individual-specific biometrics
invitro_mc	Perform Monte Carlo uncertainty and variability simulation for in vitro measured parameters	lowest measurable values

Table S2. Descriptions and key arguments of the key functions involved in Monte Carlo uncertainty and variability simulation in “httk”.

	Male	Female	Total
Mexican American	2018 (2514)	2270 (2484)	4288 (4998)
Other Hispanic	1213 (1358)	1277 (1450)	2590 (2808)
Non-Hispanic White	3910 (4666)	3823 (4466)	7733 (9132)
Non-Hispanic Black	2594 (2705)	2629 (2744)	5133 (5449)
Other	1951 (1092)	1925 (1067)	3876 (2159)
Total	11,596 (12,335)	12,024 (12,211)	23,620 (24,546)

Table S3. Number of NHANES respondents included in HTTK-Pop dataset, by race/ethnicity and sex for updated NHANES 2013-2018. Previous number of NHANES respondents (NHANES 2007-2012) are shown in parentheses.

Age range	N
0-3 year	2036 (2289)
3-6 year	1236 (1285)
6-11 year	2408 (2503)
11-18 year	2975 (3020)
18-65 year	12293 (12758)
65+ year	2672 (2691)

Table S4. Number of NHANES respondents included in HTTK-Pop dataset by age group for updated NHANES 2013-2018. Previous number of NHANES respondents (NHANES 2007-2012) are shown in parentheses.

		Chemical name
Absolute change (mg/L)		
Minimum	-134.4	Chlorpyrifos
Maximum	19.60	Imazalil
Mean	-5.24	
Relative change (%)		
Minimum	-99.4	Permethrin
Maximum	10.9	Imazalil
Mean	-22.5	

Table S5. Minimum, maximum, and mean values of absolute and relative Monte Carlo Css change for 95th percentile between before and after the revision of chemical-specific uncertainty propagation with the “httk” PBTK model. All 42 chemicals were affected by the revision.

		Chemical name
Absolute change (mg/L)		
Minimum	-98.8	Imazalil
Maximum	3.05	Ondansetron
Mean	2.26	
Relative change (%)		
Minimum	-49.4	Imazalil
Maximum	1250.0	Imipramine
Mean	32.3	

Table S6. Minimum, maximum, and mean values of absolute and relative Monte Carlo Css change for 95th percentile between the previous “httk” pKa data and updated OPERA pKa data with the “httk” PBTK model.

		Chemical name
Absolute change (mg/L)		
Minimum	-3.65	Ibuprofen
Maximum	11.2	Perfluorooctanoic acid
Mean	0.56	

Relative change (%)		
Minimum	-21.3	Diclofenac
Maximum	23.4	Triclosan
Mean	3.57	

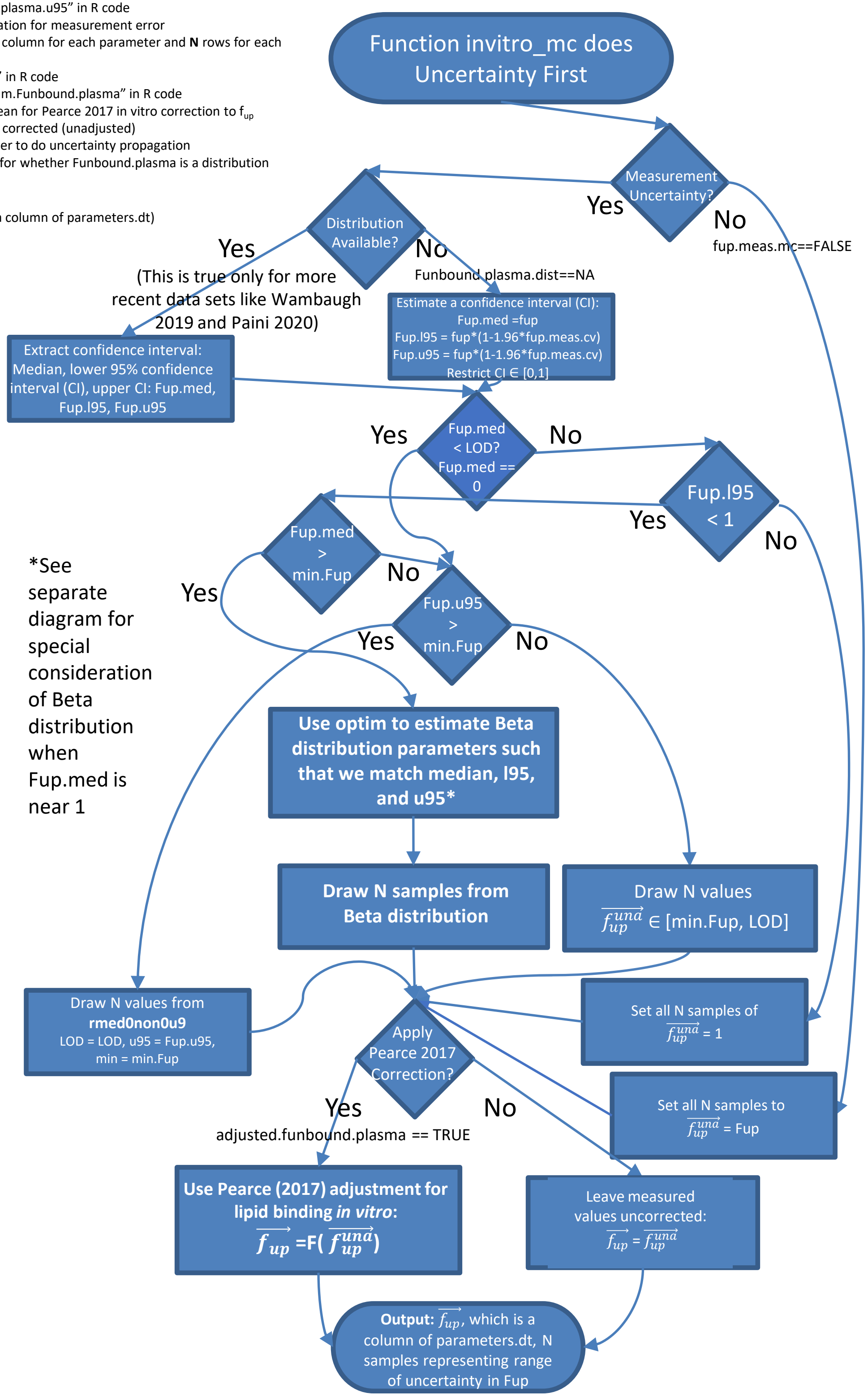
Table S7. Minimum, maximum, and mean values of absolute and relative Monte Carlo Css change for 95th percentile between the previous cohort (2007-12 NHANES cohort) and updated cohort (2013-18 NHANES cohort) with the “httk” PBTK model.

httk/R/invitro_mc.R
Function invitro_mc()
Lines 296-458 (v2.2.0)

Monte Carlo Uncertainty Simulation for Fraction Unbound in Plasma (f_{up})

Inputs (matching format in R code or abbreviated as noted**):**
Fup: short for "Funbound.plasma" in R code, "f_{up}" in paper
(Can be a string of three values separated by commas)
Fup.med: Median f_{up}, "Funbound.plasma" in R code
Fup.l95: Lower 95th credible interval value for f_{up}, "Funbound.plasma.l95" in R code
Fup.u95: Upper 95th, "Funbound.plasma.u95" in R code
fup.meas.cv = Coefficient of variation for measurement error
parameters.dt: data table with a column for each parameter and **N** rows for each sample/draw
LOD: Limit of detection, "fup.lod" in R code
min.Fup: minimum f_{up} , "minimum.Funbound.plasma" in R code
adjusted.funbound.plasma: Boolean for Pearce 2017 in vitro correction to f_{up}
F_{up}^{una}: Values that have not been corrected (unadjusted)
fup.meas.mc: Boolean for whether to do uncertainty propagation
Funbound.plasma.dist: Boolean for whether Funbound.plasma is a distribution

Outputs:
$\overrightarrow{f_{up}}$: A vector of N f_{up} values (as a column of parameters.dt)



Function invitro_mc does Uncertainty First

Measurement Uncertainty?
Yes
No
fup.meas.mc==FALSE

Distribution Available?
Yes
(This is true only for more recent data sets like Wambaugh 2019 and Paini 2020)
No
Funbound.plasma.dist==NA

Extract confidence interval: Median, lower 95% confidence interval (CI), upper CI: Fup.med, Fup.l95, Fup.u95

Estimate a confidence interval (CI):
Fup.med =fup
Fup.l95 = fup*(1-1.96*fup.meas.cv)
Fup.u95 = fup*(1-1.96*fup.meas.cv)
Restrict CI ∈ [0,1]

Fup.med < LOD? Fup.med == 0
Yes
No

Fup.l95 < 1
Yes
No

Fup.med > min.Fup
Yes
No

Fup.u95 > min.Fup
Yes
No

*See separate diagram for special consideration of Beta distribution when Fup.med is near 1

Use optim to estimate Beta distribution parameters such that we match median, l95, and u95*

Draw N samples from Beta distribution

Draw N values $\overrightarrow{f_{up}^{una}}$ ∈ [min.Fup, LOD]

Draw N values from **rmed0non0u9**
LOD = LOD, u95 = Fup.u95, min = min.Fup

Set all N samples of $\overrightarrow{f_{up}^{una}}$ = 1

Set all N samples to $\overrightarrow{f_{up}^{una}}$ = Fup

Apply Pearce 2017 Correction?
Yes
adjusted.funbound.plasma == TRUE
No

Use Pearce (2017) adjustment for lipid binding *in vitro*:
$\overrightarrow{f_{up}}$ **=F(** $\overrightarrow{f_{up}^{una}}$**)**

Leave measured values uncorrected:
$\overrightarrow{f_{up}}$ = $\overrightarrow{f_{up}^{una}}$

Output: $\overrightarrow{f_{up}}$, which is a column of parameters.dt, N samples representing range of uncertainty in Fup

Figure S1

httk/R/invitro_mc.R
Function invitro_mc()
Lines 394-430 (v2.2.0)

Inputs (matching format in R code or abbreviated as noted**):**
Fup: short for "Funbound.plasma" in R code, "f_{up}" in paper
(Can be a string of three values separated by commas)
Fup.med: Median f_{up}, "Funbound.plasma" in R code
Fup.l95: Lower 95th credible interval value for f_{up}, "Funbound.plasma.l95" in R code
Fup.u95: Upper 95th, "Funbound.plasma.u95" in R code

Outputs:
$\overrightarrow{f_{up}}$: A vector of N f_{up} values (as a column of parameters.dt)

Special Considerations for using optim and Beta Distributions when median F_{up} ~ 1

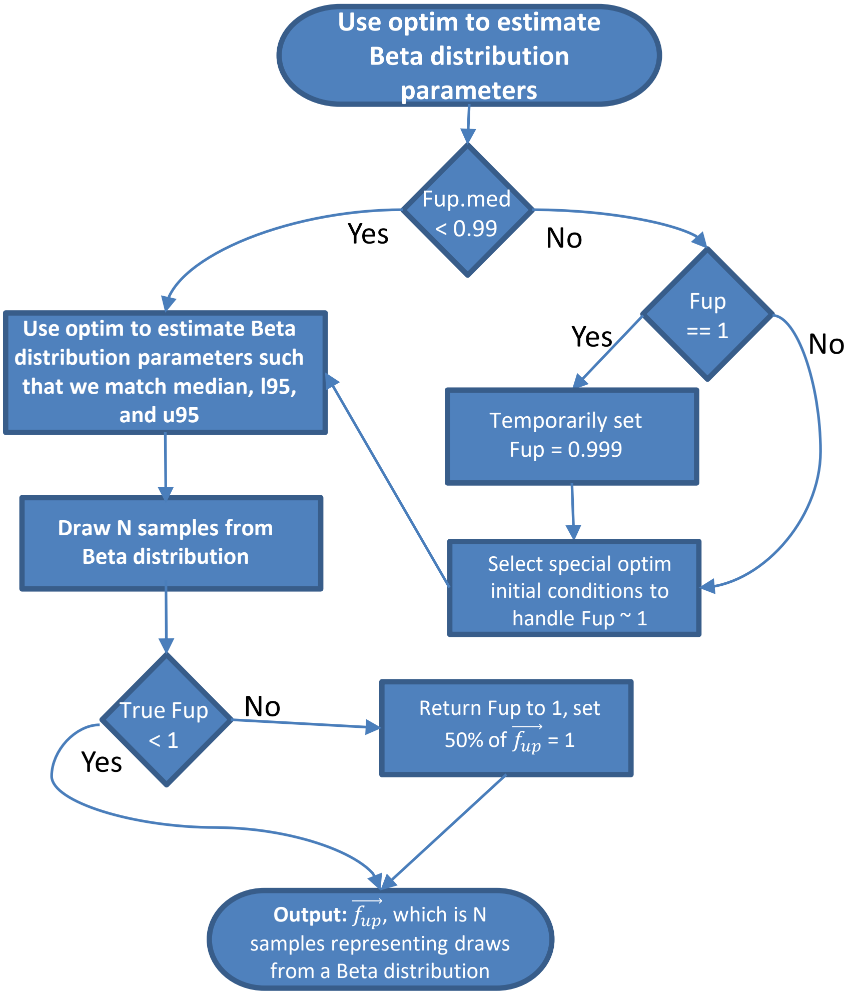

Use optim to estimate Beta distribution parameters
Fup.med < 0.99
Yes
No
Use optim to estimate Beta distribution parameters such that we match median, l95, and u95
Fup == 1
Yes
No
Temporarily set Fup = 0.999
Select special optim initial conditions to handle Fup ~ 1
Draw N samples from Beta distribution
True Fup < 1
No
Yes
Return Fup to 1, set 50% of $\overrightarrow{f_{up}}$ = 1
Output: $\overrightarrow{f_{up}}$, which is N samples representing draws from a Beta distribution

Figure S2

httk/R/invitro_mc.R
Function invitro_mc()
Lines 505-534 (v2.2.0)

Inputs:

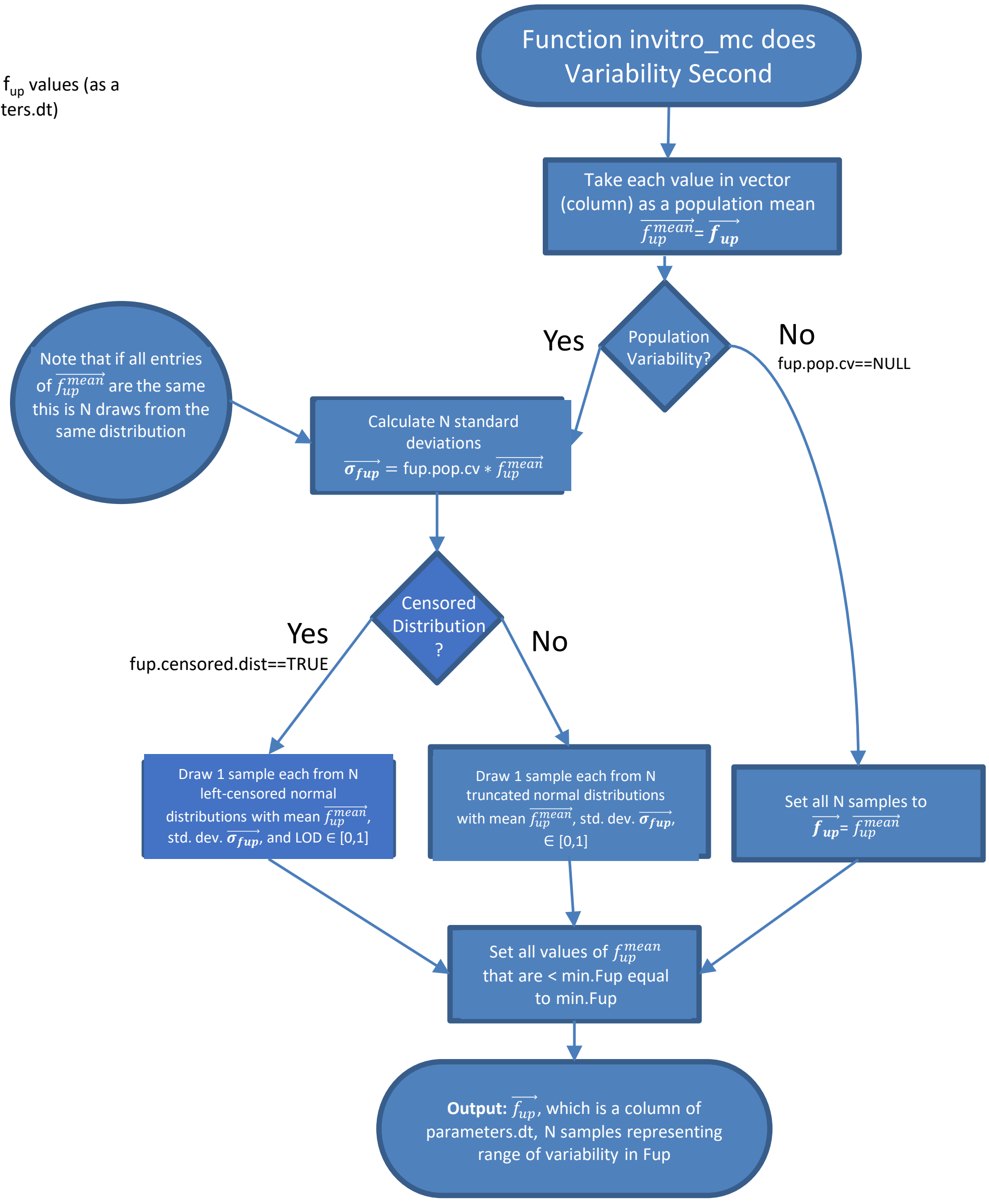
$\overrightarrow{\boldsymbol{f_{up}}}$: a vector of **N** possible values for the true measured value of f_{up}, as a column of data table **parameters.dt**, “Fup” in R code

fup.pop.cv: Coefficient of variation for population variability

Outputs:

$\overrightarrow{\boldsymbol{f_{up}}}$: A vector of N f_{up} values (as a column of parameters.dt)

Monte Carlo Variability Simulation for Fraction Unbound in Plasma (f_{up})



Function invitro_mc does Variability Second

Take each value in vector (column) as a population mean $\overrightarrow{f_{up}^{mean}} = \overrightarrow{\boldsymbol{f_{up}}}$

Population Variability?

Yes

No

fup.pop.cv==NULL

Note that if all entries of $\overrightarrow{f_{up}^{mean}}$ are the same this is N draws from the same distribution

Calculate N standard deviations $\overrightarrow{\boldsymbol{\sigma_{fup}}}$ = fup.pop.cv $*$ $\overrightarrow{f_{up}^{mean}}$

Censored Distribution ?

Yes

fup.censored.dist==TRUE

No

Draw 1 sample each from N left-censored normal distributions with mean $\overrightarrow{f_{up}^{mean}}$, std. dev. $\overrightarrow{\boldsymbol{\sigma_{fup}}}$, and LOD ∈ [0,1]

Draw 1 sample each from N truncated normal distributions with mean $\overrightarrow{f_{up}^{mean}}$, std. dev. $\overrightarrow{\boldsymbol{\sigma_{fup}}}$, ∈ [0,1]

Set all N samples to $\overrightarrow{\boldsymbol{f_{up}}} = \overrightarrow{f_{up}^{mean}}$

Set all values of $\overrightarrow{f_{up}^{mean}}$ that are < min.Fup equal to min.Fup

Output: $\overrightarrow{f_{up}}$, which is a column of parameters.dt, N samples representing range of variability in Fup

Figure S3

httk/R/invitro_mc.R
Function invitro_mc()
Lines 190-285 (v2.2.0)

Monte Carlo Uncertainty Simulation for Intrinsic Hepatic Clearance (Cl_{int})

Inputs (matching format in R code or abbreviated as noted):
Clint: "Cl_{int}" in paper
(Can be a string of four values separated by commas)
Clint.med: Median Cl_{int}
Clint.l95: Lower 95th credible interval value for Cl_{int}
Clint.u95: Upper 95th
Clint.pValue: Probability that there was no clearance observed
clint.meas.cv = Coefficient of variation for measurement error
parameters.dt: data table with a column for each parameter and **N** rows for each sample/draw
clint.meas.mc: Boolean for whether to do uncertainty propagation
Clint.dist: Boolean for whether Clint is a distribution
$f_{u,hep}$: Fraction unbound in hepatocyte assay from Kilford et al. (2008), "Fhep.assay.correction" in R code
adjusted.clint: Boolean for whether to apply Kilford correction

Outputs:
$\overrightarrow{Cl_{int}}$: A vector of N Clint values (as a column of parameters.dt)

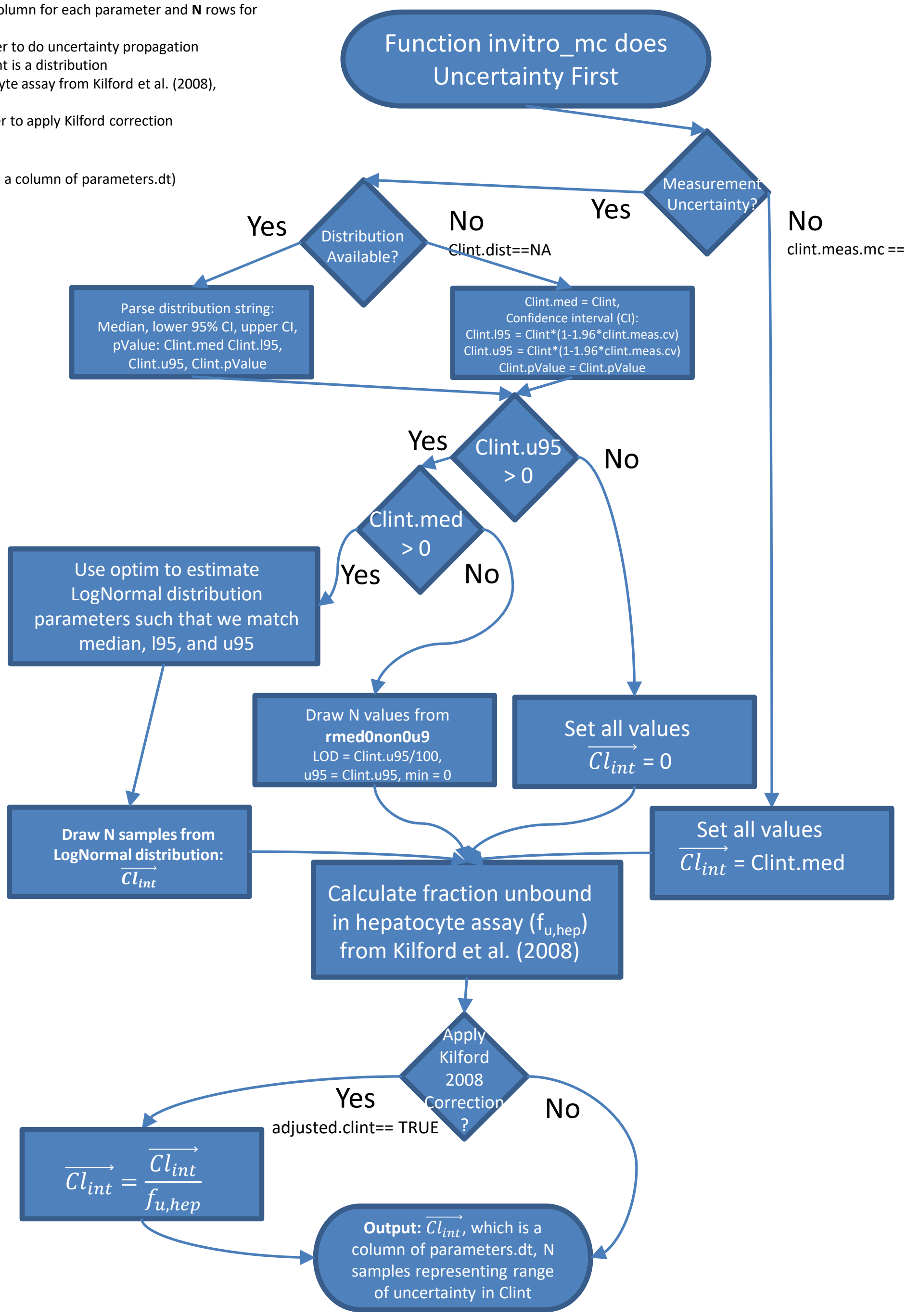

Function invitro_mc does Uncertainty First
Measurement Uncertainty?
Yes
No
clint.meas.mc == FALSE
Distribution Available?
Yes
No
Clint.dist==NA
Parse distribution string: Median, lower 95% CI, upper CI, pValue: Clint.med Clint.l95, Clint.u95, Clint.pValue
Clint.med = Clint, Confidence interval (CI): Clint.l95 = Clint*(1-1.96*clint.meas.cv) Clint.u95 = Clint*(1-1.96*clint.meas.cv) Clint.pValue = Clint.pValue
Clint.u95 > 0
Yes
No
Clint.med > 0
Yes
No
Use optim to estimate LogNormal distribution parameters such that we match median, l95, and u95
Draw N values from rmed0non0u9 LOD = Clint.u95/100, u95 = Clint.u95, min = 0
Set all values $\overrightarrow{Cl_{int}}$ = 0
Draw N samples from LogNormal distribution: $\overrightarrow{Cl_{int}}$
Set all values $\overrightarrow{Cl_{int}}$ = Clint.med
Calculate fraction unbound in hepatocyte assay ($f_{u,hep}$) from Kilford et al. (2008)
Apply Kilford 2008 Correction ?
Yes
adjusted.clint== TRUE
No
$\overrightarrow{Cl_{int}} = \frac{\overrightarrow{Cl_{int}}}{f_{u,hep}}$
Output: $\overrightarrow{Cl_{int}}$, which is a column of parameters.dt, N samples representing range of uncertainty in Clint

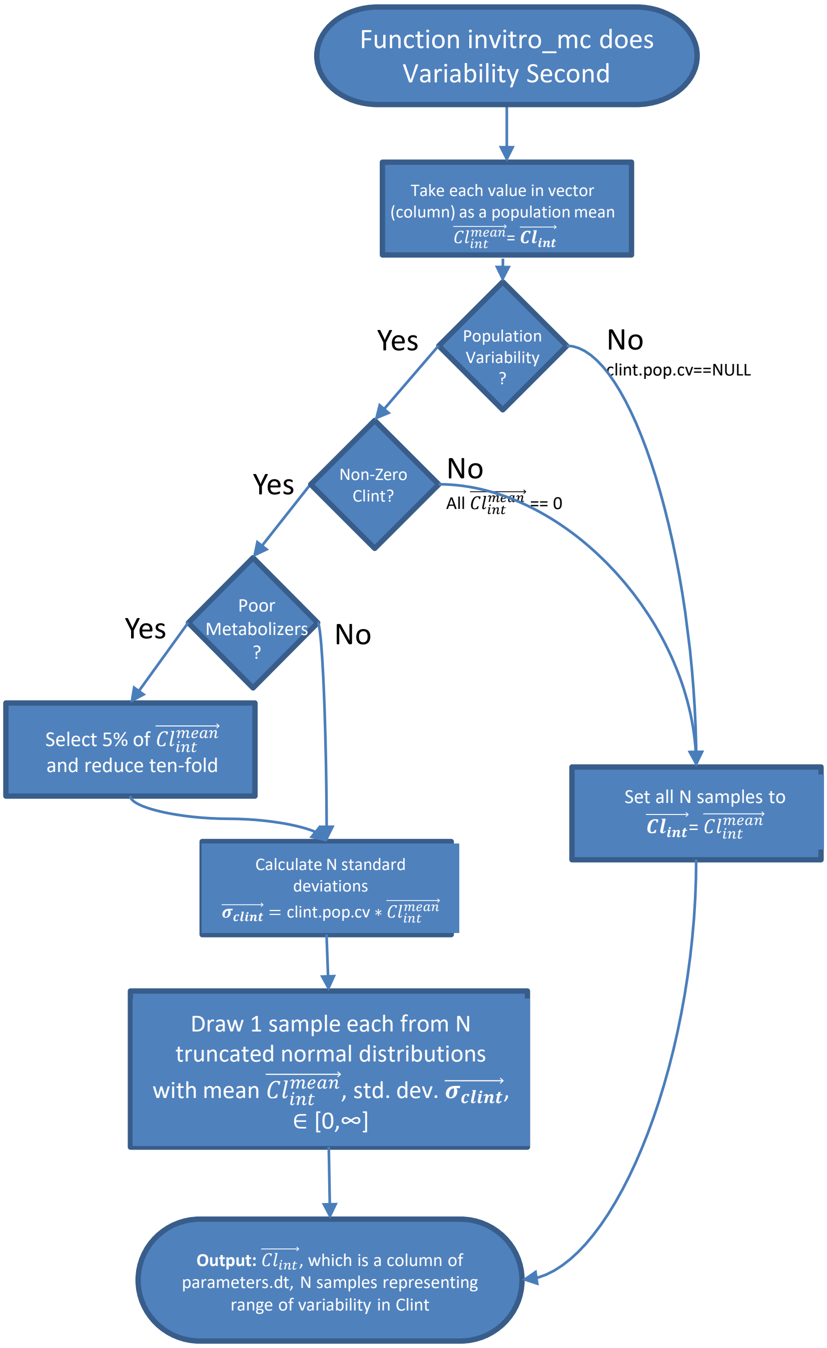
Figure S4

httk/R/invitro_mc.R
Function invitro_mc()
Lines 473-494 (v2.2.0)

Inputs:
$\overrightarrow{\boldsymbol{Cl_{int}}}$: a vector of **N** possible values for the true measured value of Cl_{int}, as a column of data table, “Clint” in R code
clint.pop.cv: Coefficient of variation for population variability

parameters.dt: **Outputs:**
$\overrightarrow{\boldsymbol{Cl_{int}}}$: A vector of N Cl_{int} values (as a column of parameters.dt)

Monte Carlo Variability Simulation for Intrinsic Hepatic Clearance (Cl_{int})



Function invitro_mc does Variability Second

Take each value in vector (column) as a population mean $\overrightarrow{Cl_{int}^{mean}}$= $\overrightarrow{\boldsymbol{Cl_{int}}}$

Population Variability ?

Yes

No

clint.pop.cv==NULL

Non-Zero Clint?

Yes

No

All $\overrightarrow{Cl_{int}^{mean}}$ == 0

Poor Metabolizers ?

Yes

No

Select 5% of $\overrightarrow{Cl_{int}^{mean}}$ and reduce ten-fold

Set all N samples to $\overrightarrow{\boldsymbol{Cl_{int}}}$= $\overrightarrow{Cl_{int}^{mean}}$

Calculate N standard deviations $\overrightarrow{\boldsymbol{\sigma_{clint}}}$ = clint.pop.cv $*$ $\overrightarrow{Cl_{int}^{mean}}$

Draw 1 sample each from N truncated normal distributions with mean $\overrightarrow{Cl_{int}^{mean}}$, std. dev. $\overrightarrow{\boldsymbol{\sigma_{clint}}}$, ∈ [0,∞]

Output: $\overrightarrow{Cl_{int}}$, which is a column of parameters.dt, N samples representing range of variability in Clint

Figure S5

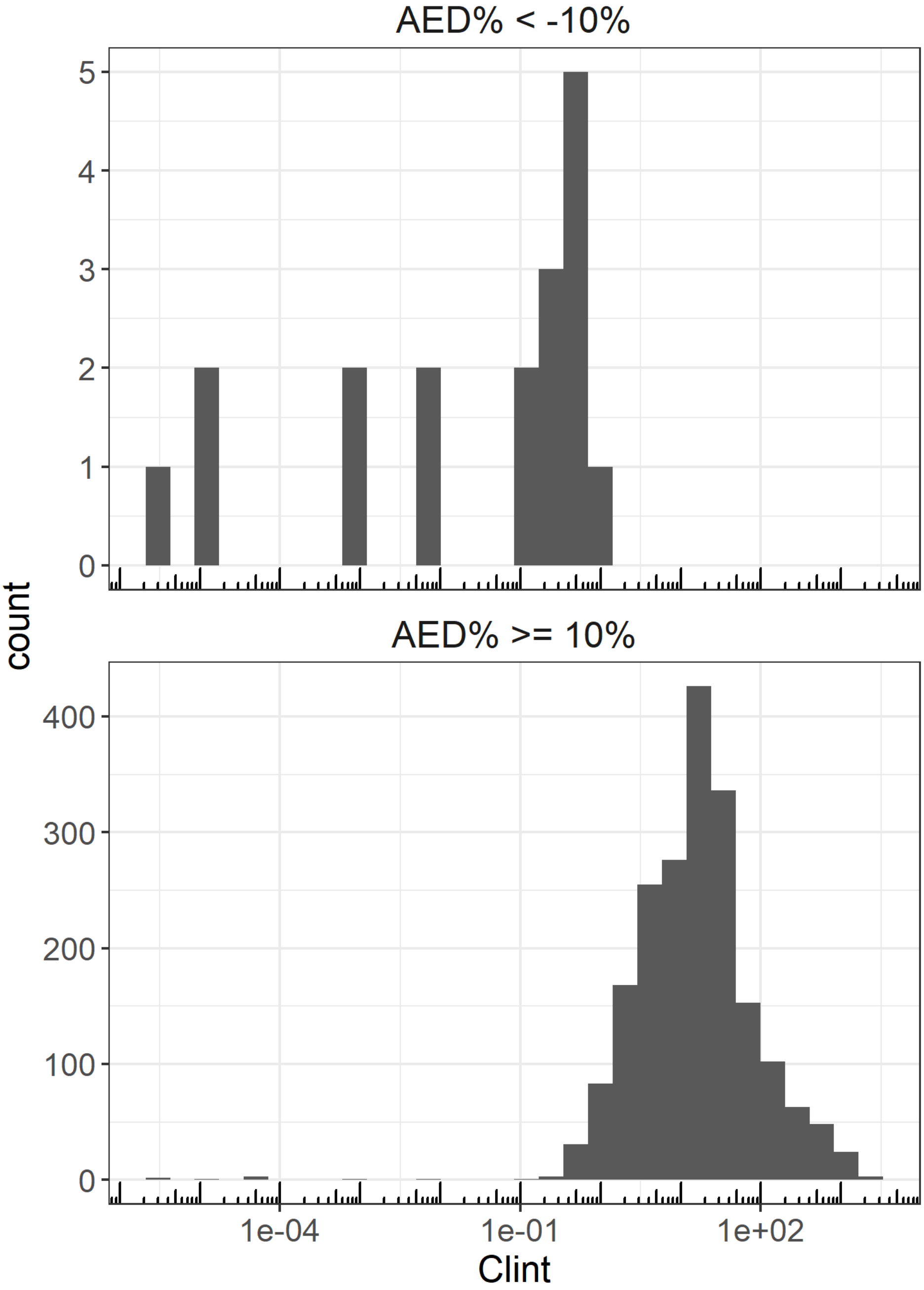

AED% < -10%
5
4
3
2
1
0
count
AED% >= 10%
400
300
200
100
0
1e-04
1e-01
1e+02
Clint

Figure S6

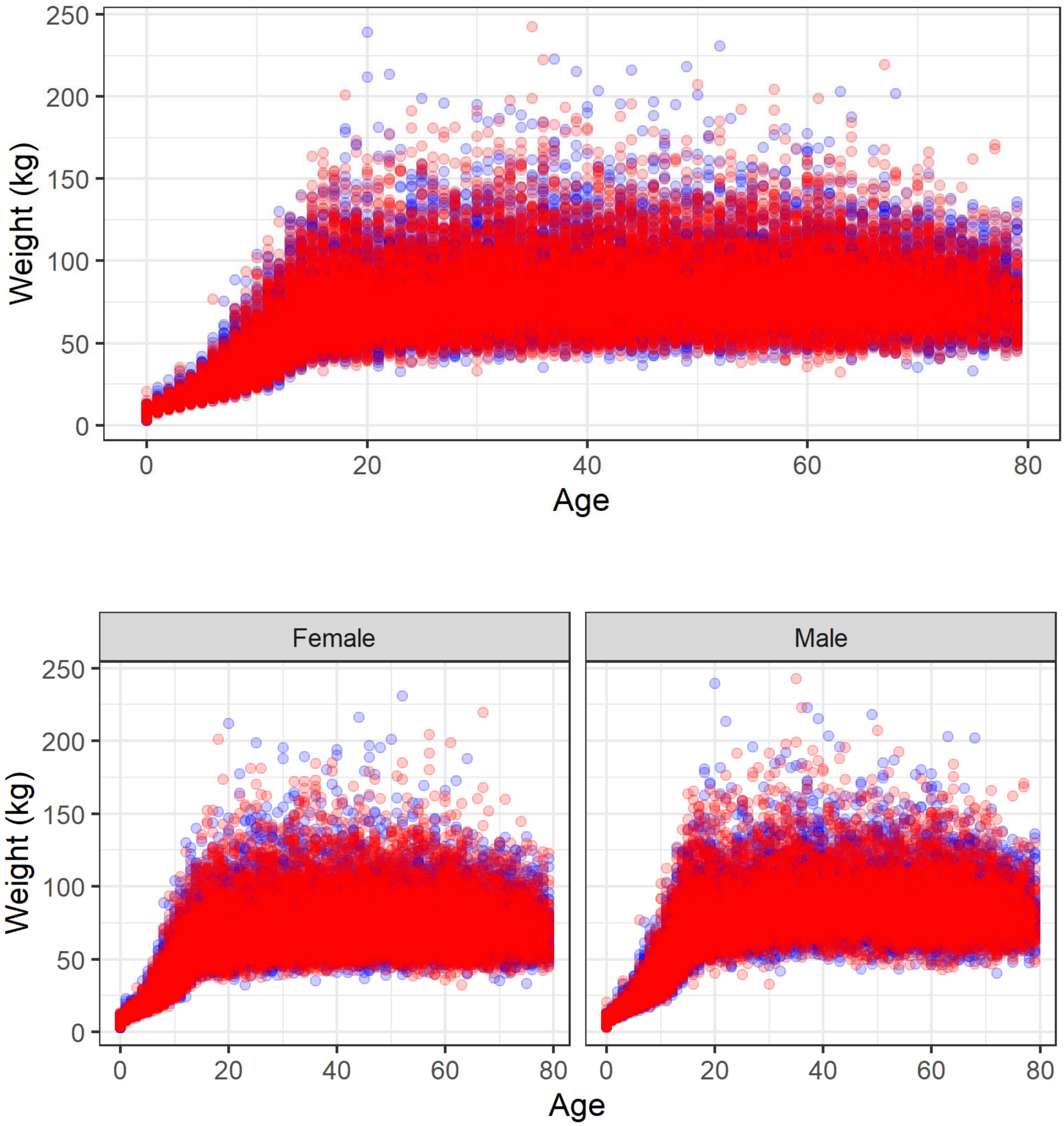

250
200
150
100
50
0
Weight (kg)
0
20
40
60
80
Age
Female
Male
250
200
150
100
50
0
Weight (kg)
0
20
40
60
80
0
20
40
60
80
Age

Figure S7

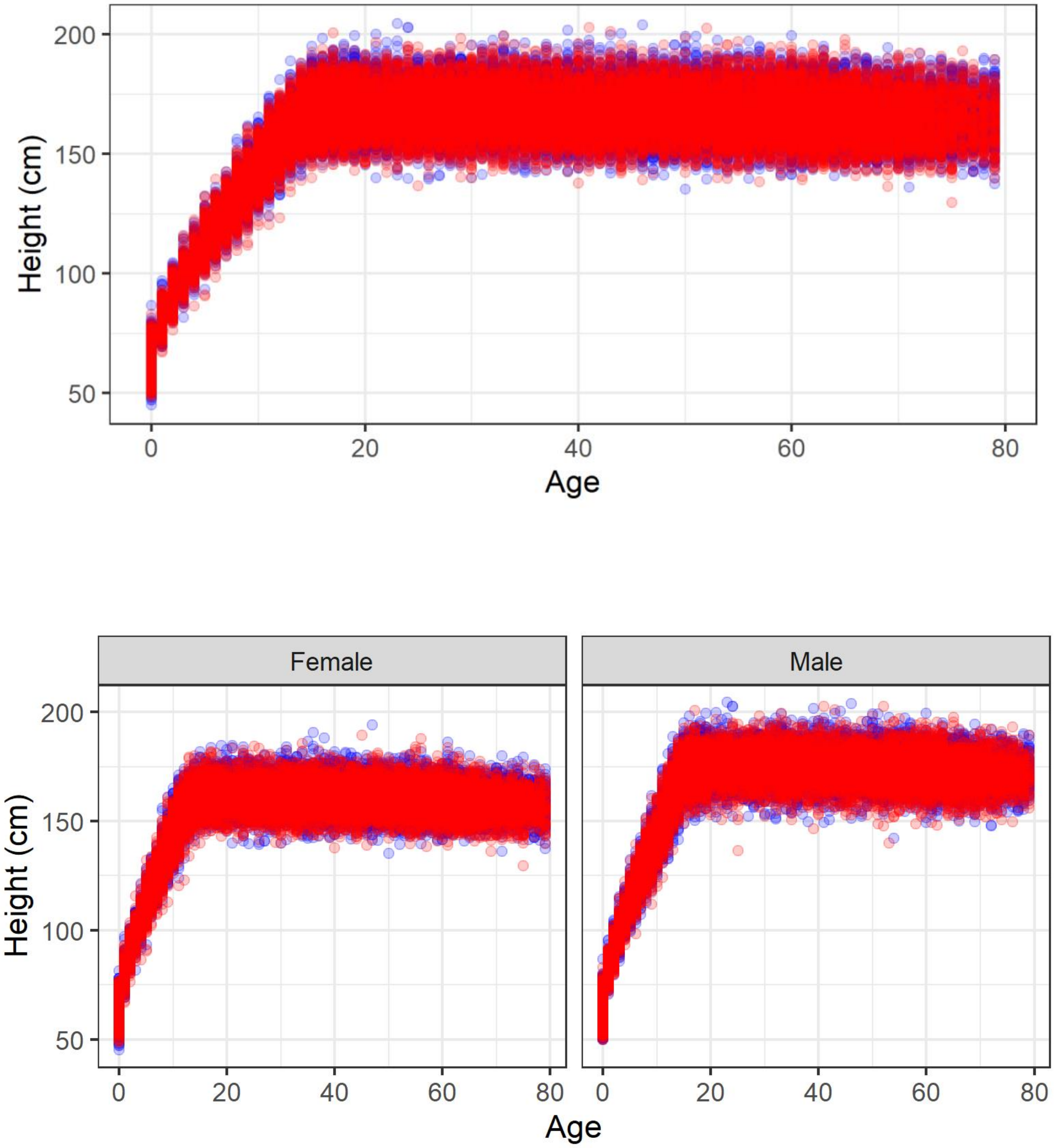

Height (cm)
200
150
100
50
0
20
40
60
80
Age
Female
Male
Height (cm)
200
150
100
50
0
20
40
60
80
0
20
40
60
80
Age

Figure S8

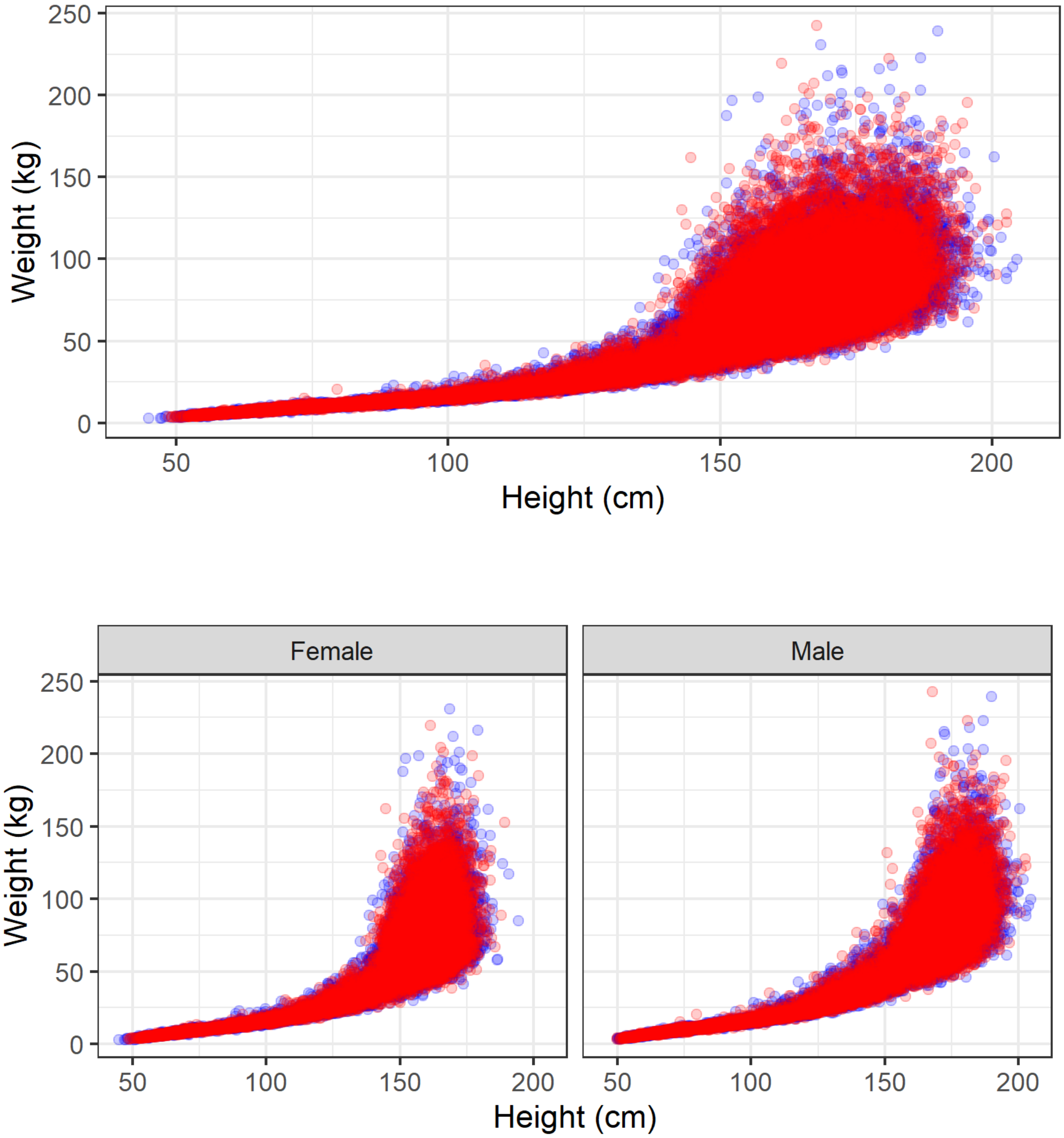

Weight (kg)
250
200
150
100
50
0
50
100
150
200
Height (cm)
Female
Male
Weight (kg)
250
200
150
100
50
0
50
100
150
200
50
100
150
200
Height (cm)

Figure S9

CKD-EPI residual variability

Caroline Ring

11/5/2020

```
devtools::load_all()
library(data.table)
library(ggplot2)
```

For reproducibility of the random sampling, set a seed for the random number generator.

```
set.seed(42)
```

We're going to try to back-calculate the log-scale residual variability for the CKD-EPI regression (Levey et al. 2009) using the info on the natural scale residual variability and the info on the eGFR distribution given in Levey et al. (2009).

Here's the math.

On the log scale, the log measured GFR (mGFR) is equal to the log of the CKD-EPI predicted estimated GFR (eGFR) plus some residual error ϵ.

$$(1) \quad \log \text{mGFR} = \log \text{eGFR} + \epsilon$$

We can assume that ϵ obeys a zero-mean normal distribution with constant variance and some unknown standard deviation σ (this is a standard assumption for ordinary least-squares regression).

$$\epsilon \sim \text{Normal}(\mu = 0, \sigma = \sigma)$$

On the natural scale, the measured GFR (mGFR) is equal to the CKD-EPI predicted estimated GFR (eGFR) plus some residual error δ.

$$(2) \quad \text{mGFR} = \text{eGFR} + \delta$$

What is the distribution of δ? This is the residual variability.

It's clear from Figure 1 of Levey et al. (2009) that δ does *not* obey a zero-mean normal distribution with constant variance. In particular, it seems that variance increases with eGFR.

We can get the distribution of δ from the distribution of ϵ by deriving an equation for δ in terms of ϵ.

Convert the log scale to the natural scale by exponentiating both sides of Equation 1:

$$(3) \quad \text{mGFR} = \exp \epsilon \times \text{eGFR}$$

Combine (2) and (3):

$$(4)\quad \text{eGFR} + \delta = \exp\epsilon \times \text{eGFR}$$

And solve for δ

$$(5)\quad \delta = \exp\epsilon \times \text{eGFR} - \text{eGFR} = \text{eGFR} \times (\exp\epsilon - 1)$$

That is,

$$(6)\quad \log(\delta + \text{eGFR}) = \epsilon + \log(\text{eGFR})$$

Now, if $\epsilon \sim \text{Normal}(\mu = 0, \sigma = \sigma)$, then

$$\epsilon + \log(\text{eGFR} \sim \text{Normal}(\mu = \log(\text{eGFR}), \sigma = \sigma)$$

Which means that

$$(7)\quad \log(\delta + \text{eGFR})\ \sim \text{Normal}(\mu = \log(\text{eGFR}), \sigma = \sigma)$$

This implies a three-parameter log-normal distribution for δ: log-scale mean equal to log(eGFR); log-scale standard deviation equal to σ, and a shift parameter equal to $-$eGFR. The shift parameter just means that δ can't be less than $-$eGFR – which makes sense, because according to Equation 2, if δ were less than $-$eGFR, then mGFR would be negative, which is not physically possible.

So, that means we just need to find the value of σ that does the best job of reproducing the residual summary statistics provided in Levey et al. (2009), Appendix Table 6.

However, since the distribution of δ depends on eGFR, that means we have to get the right distribution of eGFR in order to get the right marginal distribution of δ.

Our only information about the distribution of eGFR comes from Table 4 of Levey et al. (2009), in which they report percentiles of eGFR predicted by the CKD-EPI equation in the "external validation" dataset.

eGFR distribution from Levey et al. 2009, Table 4 (external validation dataset):

eGFR	Percent	n
<15	3.7%	144
15-29	12.1%	473
30-59	33.2%	1295
60-89	25.5%	992
>90	25.4%	989

From Figure 1 of Levey et al. (2009), the rough upper bound for eGFR looks to be around 150.

This gives us a rough eCDF for eGFR if we compute the cumulative percent at or below each upper bound.

```r
egfr_pct <- data.table(eGFR = c(0, 15, 29, 59, 89, 150),
                       pct_bin = c(0.0, 3.7, 12.1, 33.2, 25.5, 25.4),
                       n = c(0, 144,473, 1295, 992, 989))
egfr_pct[, pctile:=cumsum(pct_bin)/100]
egfr_pct[, cumul_n:=cumsum(n)]
egfr_pct[, bin:=c(
  "0",
  "0-15",
                  "15-29",
                  "30-59",
                  "60-89",
                  "89-150")]
knitr::kable(egfr_pct[, .(bin,
                          n,
                          cumul_n,
                           pct_bin,
                          pctile*100)],
             col.names = c("eGFR",
                           "n in bin",
                           "Cumulative n",
                           "% in bin",
                           "Percentile"))
```

eGFR	n in bin	Cumulative n	% in bin	Percentile
0	0	0	0.0	0.0
0-15	144	144	3.7	3.7
15-29	473	617	12.1	15.8
30-59	1295	1912	33.2	49.0
60-89	992	2904	25.5	74.5
89-150	989	3893	25.4	99.9

Now, this distribution is only for the external evaluation dataset, whereas the reported residual statistics are for the development dataset. However, I think it's reasonable to assume that the eGFR distribution is similar for the external evaluation dataset as for the development dataset.

To sample from this distribution: draw from Uniform[0,1] and apply inverse CDF.

Inverse CDF: numerically solve CDF for a given cumulative probability.

CDF will be estimated by linear interpolation of the percentiles table above. Inverse CDF can likewise be estimated by linear interpolation, just swapping the independent and dependent variables.

```
egfr_inv_cdf <- function(x, y_spec, x_in, y_in){
  q <- approx(x = y_in,
                         y = x_in ,
                     xout = y_spec,
                         method = "linear",
                     rule = 2)$y
  return(q)
}
```

Let's try it. Randomly draw a set of samples from Uniform(0,1). (Actually, we use Uniform(0,0.999), because the percentages from Appendix Table 6 actually only add up to 0.999.) Then apply the inverse CDF function to them. This will give a sample from the estimated eGFR distribution.

We draw N = 5504 values to match the number of values in the development dataset.

```
rvals <- runif(5504, min = 0, max=0.999)
#back-convert using inverse CDF into draws from the approximated eGFR distribution
eGFR_samp <-egfr_inv_cdf(y_spec = rvals,
                          x_in = egfr_pct$eGFR,
                          y_in = egfr_pct$pctile)
```

Here is what the example distribution of eGFR values looks like:

```
ggplot(data.frame(eGFR = eGFR_samp),
       aes(x=eGFR)) +
  geom_histogram()
```

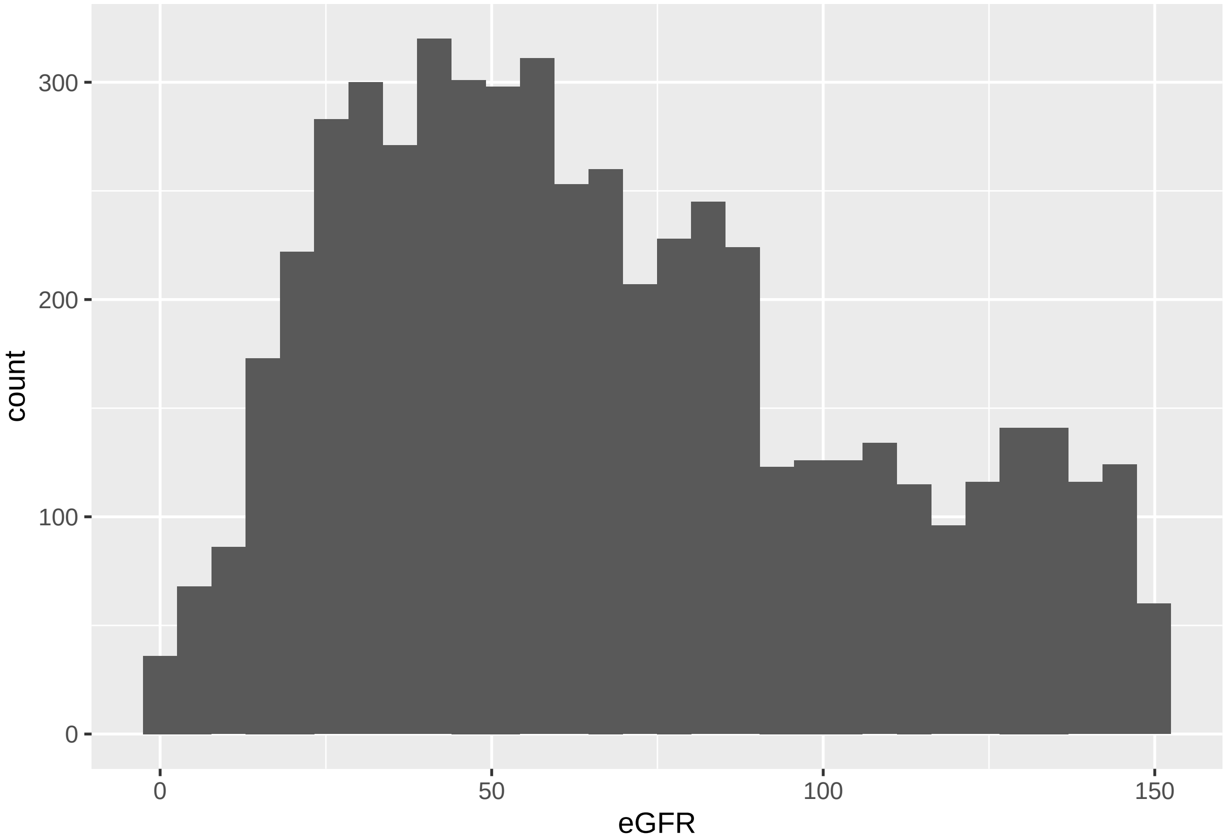


Figure 1: Histogram of sampled eGFR values from empirical CDF

Next: set up a function to draw from the distribution of residuals.

We need to find the log-scale SD that reproduces the residual statistics reported in Levey et al. (2009) Appendix Table 6 for the development dataset: median and IQR of natural-scale residuals; percentage of eGFR values within 30% of measured GFR; and RMSE on the log-scale.

Although RMSE is not explicitly stated to be on the log scale, I believe that it is actually on the log scale because of the follow-up paper (Levey et al. 2020), which report RMSE explicitly stated to be on the log scale that is of the same magnitude. It hardly makes sense for RMSE to be 0.2 if it is on the natural scale and residuals look like Levey et al. 2009 Figure 1 – it would be a lot higher in that case.

Appendix Table 6 reports the following:

Statistic	Value
Median	0.4
IQR	14.7
P30	85.6%

Statistic	Value
RMSE	0.231

```
resid_draw <- function(sigma, eGFR) {
  #residuals = measured GFR - estimated GFR
  epsilon <- rnorm(n=length(eGFR),
                       mean = 0,
                       sd = sigma)
  delta <- eGFR * (exp(epsilon) - 1)
  mGFR <- delta + eGFR
  #compute: median, IQR, P30, RMSE
  return(c("median" = median(delta),
              "IQR" = IQR(delta),
               "P30" = 100*sum(abs(delta)/mGFR <= 0.3)/length(delta),
              "RMSE" = sqrt(mean(epsilon^2))))

}
```

Now, find a value of `sigma` that simultaneously optimizes for median, IQR, P30, and RMSE.

Here is the function to be minimized.

It first calls function `resid_draw()` to draw a set of residuals for a sample of eGFR values (trying to reproduce the development dataset), and compute the median, IQR, P30, and log-scale RMSE for that set of residuals. Then, it computes and returns the square root of the sum of squared errors in median, IQR, P30, and RMSE, compared to the values reported in Levey et al. (2009) Appendix Table 6 for the development dataset. (In effect, this is the Euclidean distance from the reported values.)

This function draws 1000 different samples of eGFR values and draws a corresponding set of residuals for each one, so it computes 1000 different sums of squared errors. It then returns the average of these sums of squared errors. The idea is to average out variability that comes from randomly drawing eGFR values and residuals.

```
#Here is the function to be minimized:
optim_fun <- function(sigma, nrep = 1000, N = 5504) {
  #do 1000 replicates and average,
  #to "average out" variability in the randomly sampled residuals
  foo <- t(replicate(nrep,
                 {
                   #randomly draw values on unif(0,1)
# N = 5504 in development dataset
rvals <- runif(N, min = 0, max=0.999)
#back-convert using inverse CDF
#into draws from the approximated eGFR distribution
eGFR_samp <- egfr_inv_cdf(y_spec = rvals,
                          x_in = egfr_pct$eGFR,
                          y_in = egfr_pct$pctile)

    resid_stats <- resid_draw(sigma,
                eGFR_samp)
                 }
))

sqrt(sum((colMeans(foo) - c("median" = 0.4,
                          "IQR" = 14.7,
                          "P30" = 85.6,
                          "RMSE" = 0.231))^2))
}
```

Now, let's do the optimization.

```
(optim_results <- optim(par = 0.2, #initial value for sigma
      fn = optim_fun,
      method = "L-BFGS-B",
      lower = 1e-6,
      upper = Inf))
```

```
## $par
## [1] 0.2061534
##
## $value
## [1] 0.4202734
##
## $counts
## function gradient
##       11       11
##
## $convergence
## [1] 0
##
## $message
## [1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```

The best-fit sigma value is:

```
optim_results$par
```

```
## [1] 0.2061534
```

What are the calculated residual stats for the selected sigma value, and how do they compare to the reported values in Appendix Table 6?

```
resid_samp <- t(replicate(n = 1000,
                          expr = {
                            eGFR_samp <- egfr_inv_cdf(y_spec = rvals,
                                                      x_in = egfr_pct$eGFR,
                                                      y_in = egfr_pct$pctile)
                            resid_draw(sigma = optim_results$par,
                                       eGFR = eGFR_samp)
                          }
)
)
resid_stats_avg <- apply(resid_samp, 2, mean)

reported_stats <- c("median" = 0.4,
```

```r
                          "IQR" = 14.7,
                          "P30" = 85.6,
                          "RMSE" = 0.231)

df <- data.frame("Original" = reported_stats,
                 "Optimized" = resid_stats_avg,
                 check.names = FALSE)

knitr::kable(t(df), format.args = list(digits = 3))
```

	median	IQR	P30	RMSE
Original	0.40000	14.7	85.6	0.231
Optimized	0.00656	14.9	85.7	0.206

References

- Levey AS, Stevens LA, Schmid CH, Zhang YL, Castro AF, 3rd, Feldman HI, et al. A new equation to estimate glomerular filtration rate. Ann Intern Med. 2009;150(9):604-12.
- Levey AS, Titan SM, Powe NR, Coresh J, Inker LA. Kidney Disease, Race, and GFR Estimation. Clin J Am Soc Nephrol. 2020;15(8):1203-12.