A Gentle Introduction to Optimal Design for Pharmacometric Models

with PopED and mrgsolve

Tim Waterhouse
Metrum Research Group

8 June, 2020

*We are not affiliated, associated, authorized, endorsed by, or in any way officially connected with PopED, or any of its subsidiaries or its affiliates.

*Only Kyle is affiliated, associated, authorized, endorsed by, or in any way officially connected with mrgsolve, or any of its subsidiaries or its affiliates.

Why are we here?

- You want to design a study
- You'll be fitting a model to the results
- You don't want that model to fail spectacularly
- You don't have the time or the patience to run a bunch of simulations

Outline

- Optimal design background
- Software tools
- Simple model with `PopED`
    - Evaluation
    - Optimization
    - Simulation
- More complex models with `PopED` and `mrgsolve`

Acknowledgements

- Kyle Baron
- Devin Pastoor
- Seth Green

Poll Question #1

What is your experience with optimal design?

1. Never heard of it.
2. I've seen it around, and I'd like to try it.
3. I've seen it around, but it seems kinda dodgy.
4. I've used optimal design for some studies.
5. I invert Fisher information matrices in my head.
6. Huh? Sorry, I was checking email.

Optimal design background

Meet the Fisher information matrix (FIM)

$$M_F(\boldsymbol{\Psi}, \xi) = -\,\mathrm{E}\left[\frac{\partial^2}{\partial \boldsymbol{\Psi} \partial \boldsymbol{\Psi}^T} \log L(\boldsymbol{\Psi}; y) \middle| \boldsymbol{\Psi} \right]$$

where

- $\boldsymbol{\Psi}$ is the vector of populations parameters (e.g. `THETA`s, `OMEGA`s, and `SIGMA`s in NONMEM),
- y is the vector of observations,
- ξ is the vector of design variables (e.g. sampling times), and
- $\log L$ is the log-likelihood.

Fisher in winter coat

Why should I care about that thing?

Cramér-Rao lower bound:

$$\mathrm{cov}(\hat{\boldsymbol{\Psi}}) \geq \left[M_F(\boldsymbol{\Psi}, \xi)\right]^{-1}$$

when $\hat{\boldsymbol{\Psi}}$ is an unbiased estimator of $\boldsymbol{\Psi}$.

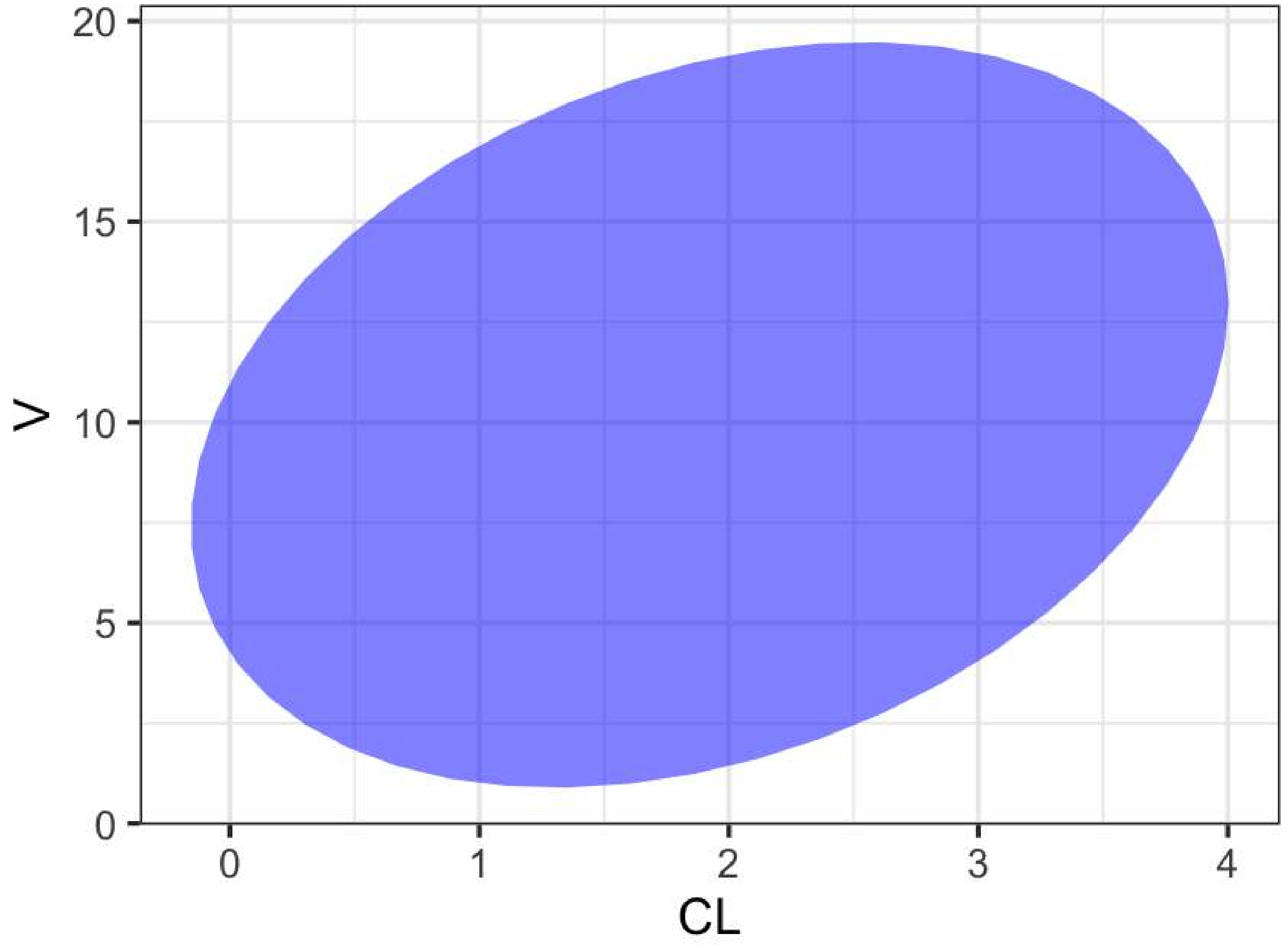
- Lower bounds for relative standard errors (RSEs) can be obtained from the diagonals of the inverse of the FIM
- This means we have a quick way of evaluating (lower bounds on) the precision of our parameter estimates.

OK, but really. Why should I care about that thing?

$$\mathrm{cov}(\hat{\mathbf{\Psi}}) \geq [M_F(\mathbf{\Psi}, \xi)]^{-1}$$

- D-optimality criterion
- D-optimal designs maximise the determinant of the FIM
- Equivalent to minimising the volume of the confidence ellipsoid of the parameter estimates
- Huh?

This is a confidence ellipsoid in 2 dimensions



Catch-22 of optimal design

- For linear models, the dependence of $M_F(\mathbf{\Psi}, \xi)$ on $\mathbf{\Psi}$ disappears
- No such luck for nonlinear models
- In order to design our experiment in a way that will produce the best parameter estimates, we first need to know the values of those parameters

Nonlinear mixed effects models are even more problematic

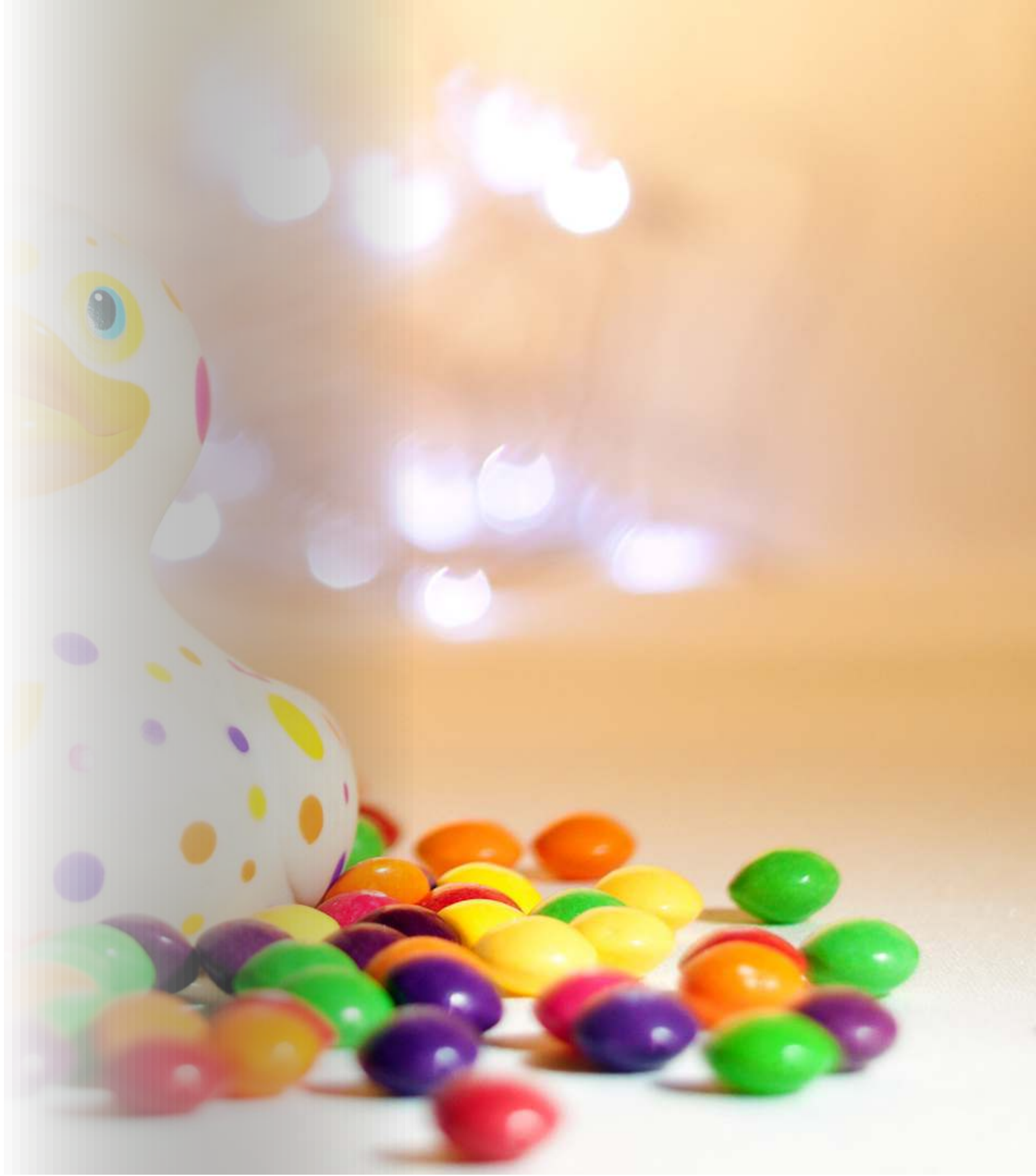
- No analytic expression for the likelihood, so we rely on approximations
- So our FIM is
  - an approximation
  - to a lower bound
  - that depends on the parameter values

- But…
- All is not lost
- Usually we have adequate information on parameter estimates
- Approximate lower bounds are usually not far off from values obtained from simulation
- More to come on simulation…

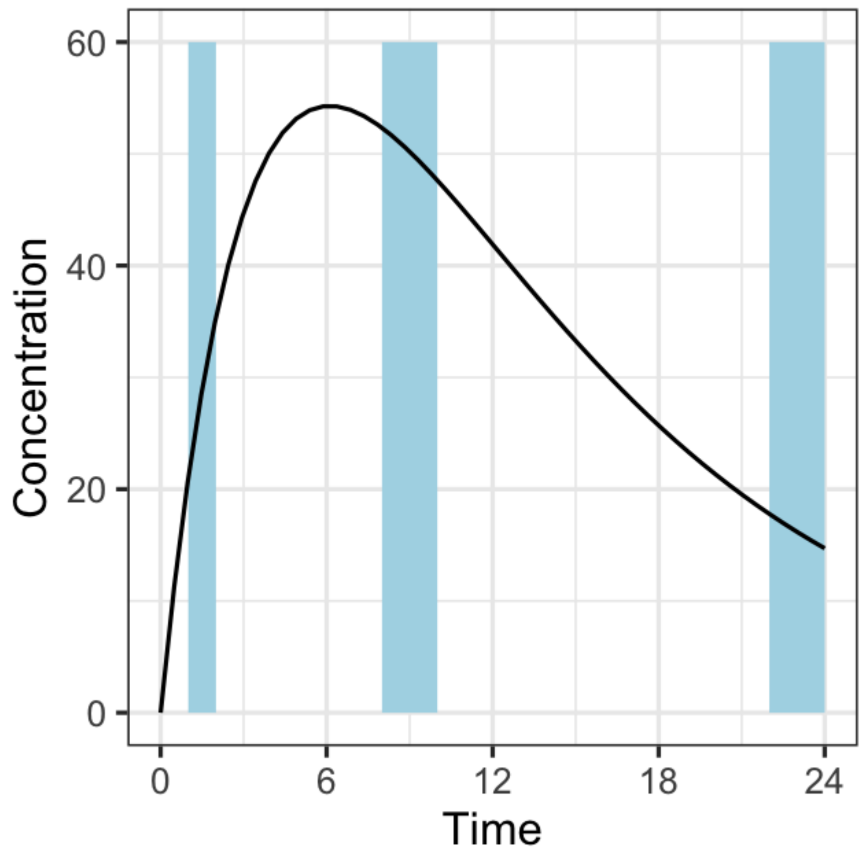
Mentre, Mallet, and Baccar (1997)
Retout, Duffull, and Mentre (2001)
Retout and Mentre (2003)

Evaluation vs Optimisation

- Optimal design can be used to optimise a study (duh)
- We can also just use the FIM to quickly evaluate a design by calculating RSEs
- Optimisation is often a last resort (we can just evaluate a few candidate designs in many situations)
- Sometimes resources are too tightly constrained or our intuition isn't good enough to find feasible designs without optimising using a search algorithm

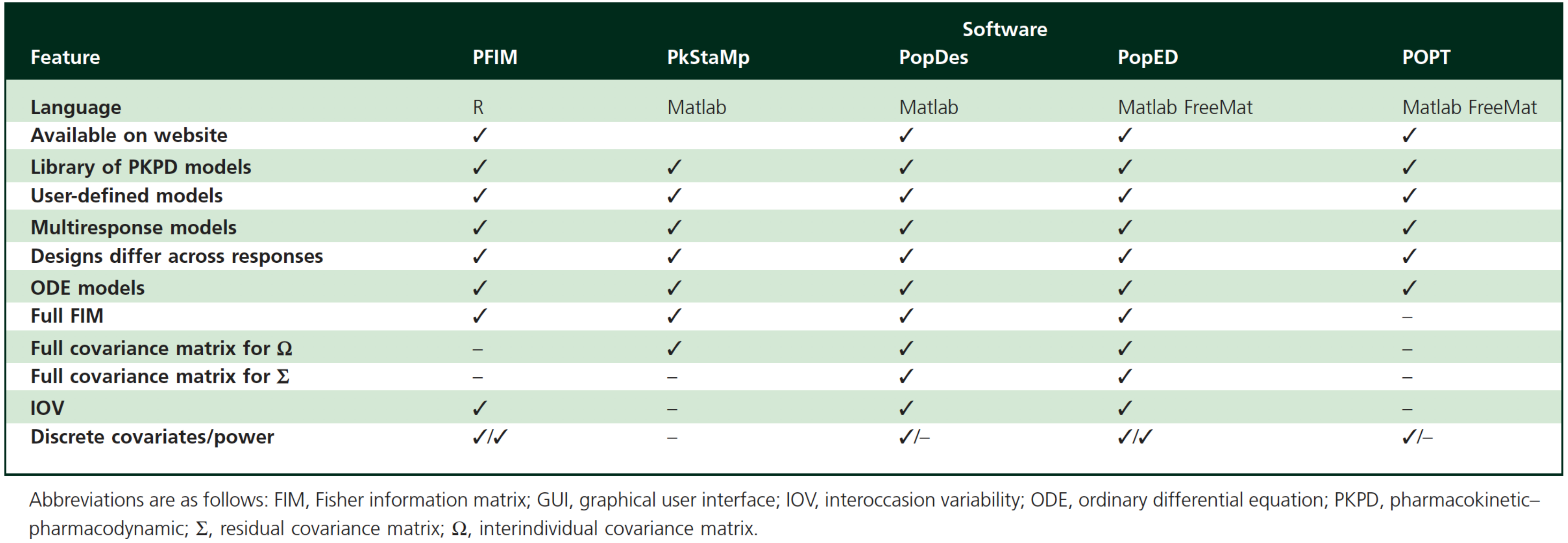


Sampling windows



- "Optimal" sampling times are often not practical
- Even without optimisation, we can't always collect samples at precise times
- Sampling windows can be optimised or determined manually

Tools for optimal design

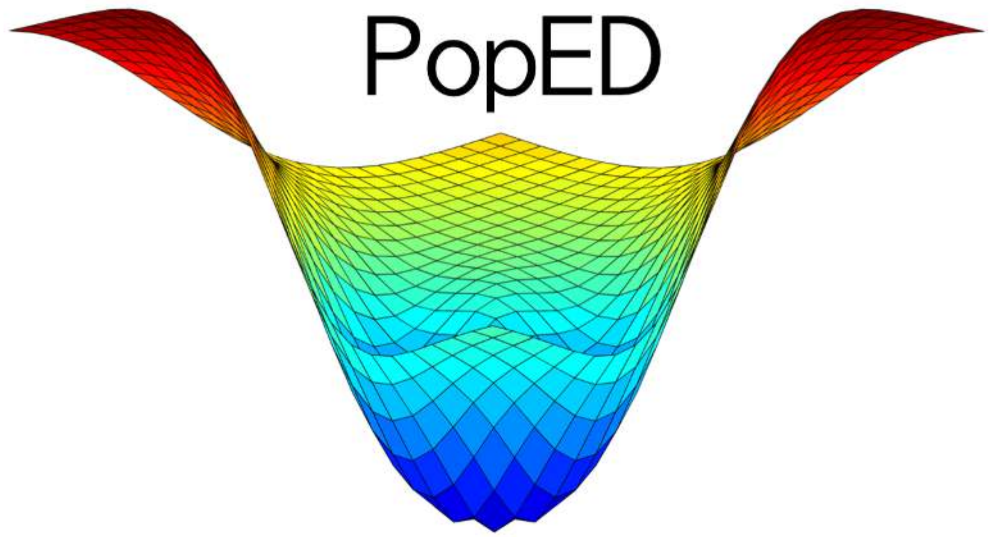


Feature	Software				
	PFIM	PkStaMp	PopDes	PopED	POPT
Language	R	Matlab	Matlab	Matlab FreeMat	Matlab FreeMat
Available on website	✓		✓	✓	✓
Library of PKPD models	✓	✓	✓	✓	✓
User-defined models	✓	✓	✓	✓	✓
Multiresponse models	✓	✓	✓	✓	✓
Designs differ across responses	✓	✓	✓	✓	✓
ODE models	✓	✓	✓	✓	✓
Full FIM	✓	✓	✓	✓	–
Full covariance matrix for Ω	–	✓	✓	✓	–
Full covariance matrix for Σ	–	–	✓	✓	–
IOV	✓	–	✓	✓	–
Discrete covariates/power	✓/✓	–	✓/–	✓/✓	✓/–

Abbreviations are as follows: FIM, Fisher information matrix; GUI, graphical user interface; IOV, interoccasion variability; ODE, ordinary differential equation; PKPD, pharmacokinetic–pharmacodynamic; Σ, residual covariance matrix; Ω, interindividual covariance matrix.

Nyberg, Bazzoli, Ogungbenro, et al. (2014)

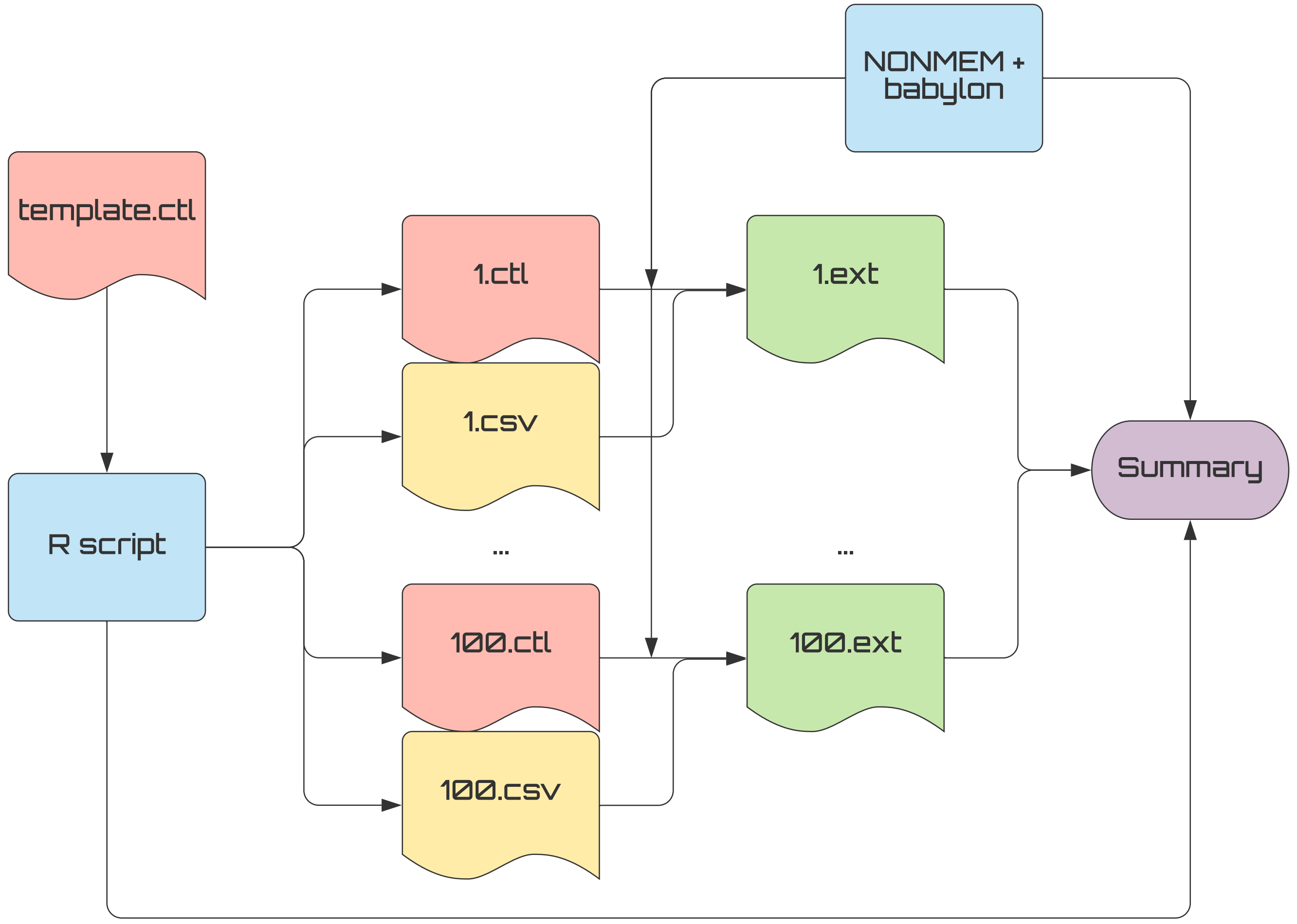
PopED



- https://andrewhooker.github.io/PopED/
- Originally in O-Matrix and Matlab, now an R package

Foracchia, Hooker, Vicini, et al. (2004)
Nyberg, Ueckert, Strömberg, et al. (2012)

SSE: Stochastic Simulation and Estimation



Example: Closed-form PK model

Introducing our example

Mockdrozaline has been studied in adult subjects, and we now must design a study in pediatric patients.

A study objective is to evaluate the PK in this new population, but PK sampling is necessarily sparse.

Our mission is to ensure that these samples are timed such that we can sufficiently estimate the PK parameters in pediatric patients.

- Population
    - 12 subjects
    - Aged 6 to < 12
    - Expected median weight of 32 kg
- Treatment
    - 10 mg QD mockdrozaline for 24 weeks
- PK samples
    - Proposed samples:
        - 5 hours postdose on Day 1;
        - predose on Weeks 8, 12, 24; and
        - 168 hours after the final dose

The model

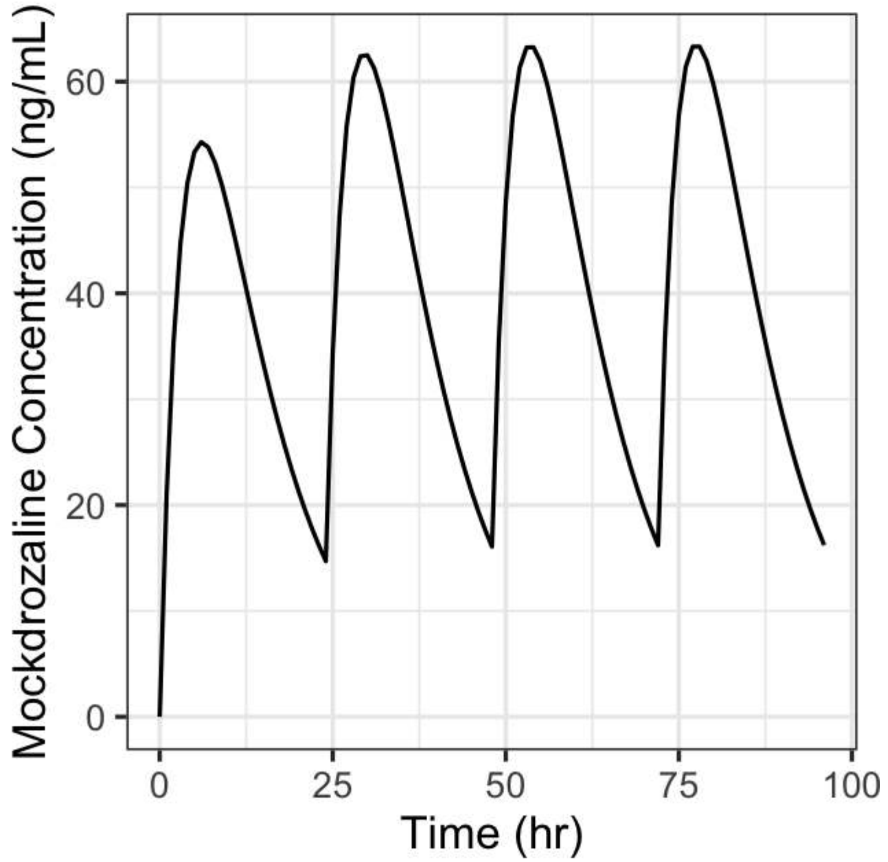
1-compartment model with 1st-order absorption and weight covariates on CL and V:

CL	V	KA	wt_cl	wt_v
10	100	0.25	0.75	1

Log-normal IIV on CL, V, and KA; additive & proportional residual error:

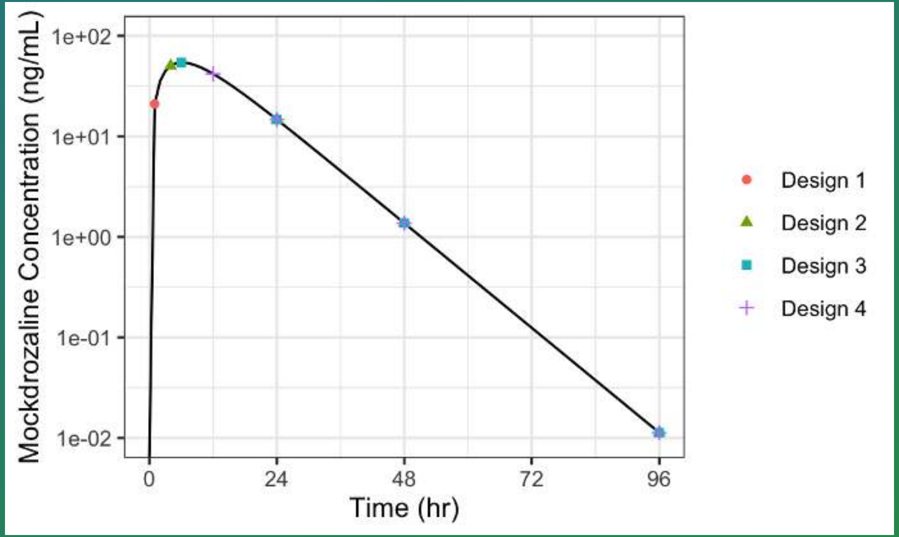
om_CL	om_V	om_KA	sigma_prop	sigma_add
0.08	0.1	0.2	0.05	1

(these are variances)



Poll Question #2

Which of these designs will give us the best* RSE for KA, assuming a single 10 mg dose in 10 adult (70 kg) subjects?



1. 1, 24, 48, 96 hours
2. 4, 24, 48, 96 hours
3. 6, 24, 48, 96 hours
4. 12, 24, 48, 96 hours
5. My kid came looking for snacks and I missed everything you just said.

[*]According to PopED

$t_{\max} \approx 6$ hours

Poll Question #2

Which of these designs will give us the best[*] RSE for KA, assuming a single 10 mg dose in 10 adult (70 kg) subjects?

- Design 1 (first sample at 1 hours): RSE = 23.4%
- Design 2 (first sample at 4 hours): RSE = 27.1%
- Design 3 (first sample at 6 hours): RSE = 30.5%
- Design 4 (first sample at 12 hours): RSE = 48.9%

[*]According to PopED

The PopED setup

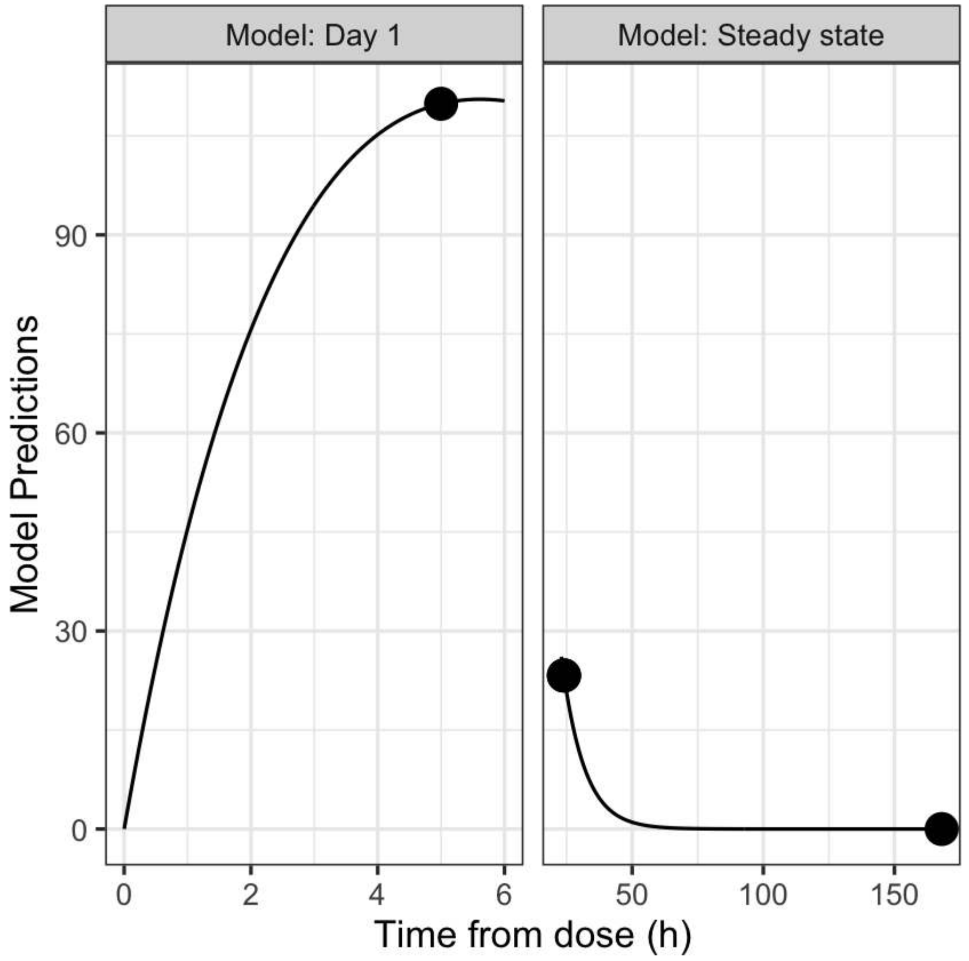
PopED requires 3 functions in order to define a model:

- `ff()`, the structural model;
- `fg()`, the parameter model (including IIV and IOV);
- `feps()`, the residual error model.

`create.poped.database()` collects together these model functions, parameter values, and everything related to study design.

Plot of initial design

```
plot_model_prediction(
  poped_db,
  model.names = c("Day 1", "Steady state"),
  facet_scales = "free_x",
  model_num_points = 200
) +
  labs(x = "Time from dose (h)") +
  theme_bw()
```



Evaluate FIM

```
FIM <- evaluate.fim(poped_db)
det(FIM)
```

```
. [1] 0.04804071
```

```
get_rse(FIM, poped_db)
```

```
.      bpop[1]      bpop[2]      bpop[3]       D[1,1]       D[2,2]       D[3,3]
. 2.983306e+05 3.132072e+06 4.936333e+06 5.192188e+01 4.888612e+02 6.485818e+02
.   SIGMA[1,1]   SIGMA[2,2]
. 2.997297e+01 4.082483e+01
```

```
poped_db2 <- create.poped.database( poped_db, xt = c(5, c(23, 24, 24, 168)) )
FIM2 <- evaluate.fim(poped_db2)
get_rse(FIM2, poped_db2)
```

```
.    bpop[1]    bpop[2]    bpop[3]     D[1,1]     D[2,2]     D[3,3] SIGMA[1,1]
.   15.48778  142.08530  223.35925   50.88485  418.53695  549.75962   29.68444
```

D-optimal design: Starting from the original design

```r
poped_db <- create.poped.database(
  ...
  xt = c(5, c(rep(24, 3), 168)),
  minxt = c(0, c(rep(23, 3), 96)),
  maxxt = c(6, c(rep(24, 3), 168)),
  ...
)
output <- poped_optim(
  poped_db,
  opt_xt = TRUE,
  parallel = TRUE,
  parallel_type = "multicore",
  seed = 1
)
summary(output)
```

```
. ===================================================================
. FINAL RESULTS
. Optimized Sampling Schedule
. Group 1: Model 1: 0.3276
. Group 1: Model 2:      23      24      24      96
.
. OFV = 20.2291
.
. Efficiency:
.   ((exp(ofv_final) / exp(ofv_init))^(1/n_parameters)) = 18.3
.
. Expected relative standard error
. (%RSE, rounded to nearest integer):
.     Parameter   Values     RSE_0   RSE
.       bpop[1]       10    298333    38
.       bpop[2]      100   3132099   149
.       bpop[3]     0.25   4936376   153
.        D[1,1]     0.08        52    50
.        D[2,2]      0.1       489   204
.        D[3,3]      0.2       649   127
.    SIGMA[1,1]     0.05        30    30
.    SIGMA[2,2]        1        41    41
.
. Total running time: 9.327 seconds
```

Add another post dose sample at SS

```r
poped_db_extra_ss <- create.poped.database(
  ...
  xt = c(5, c(rep(24, 3), 4, 168)),
  minxt = c(0, c(rep(23, 3), 0, 96)),
  maxxt = c(6, c(rep(24, 3), 6, 168)),
  ...
)
```

```r
FIM_extra_ss <- evaluate.fim(poped_db_extra_ss)
get_rse(FIM_extra_ss, poped_db_extra_ss)
```

```
.     bpop[1]     bpop[2]     bpop[3]      D[1,1]      D[2,2]      D[3,3] SIGMA[1,1]
.    13.38487    80.99149   119.36373    46.82132   253.41994   288.82097   24.18497
.  SIGMA[2,2]
.    40.80823
```

D-optimal design: Add another post dose sample at SS

```r
output_extra_ss <- poped_optim(
  poped_db_extra_ss,
  opt_xt = TRUE,
  parallel = TRUE,
  parallel_type = "multicore",
  seed = 1
)
summary(output_extra_ss)
```

```
. =============================================================
. FINAL RESULTS
. Optimized Sampling Schedule
. Group 1: Model 1:      6
. Group 1: Model 2: 0.9979     23     23     24     96
.
. OFV = 24.6494
.
. Efficiency:
.   ((exp(ofv_final) / exp(ofv_init))^(1/n_parameters)) = 1.66
.
. Expected relative standard error
. (%RSE, rounded to nearest integer):
.     Parameter   Values   RSE_0   RSE
.       bpop[1]       10      13     9
.       bpop[2]      100      81    26
.       bpop[3]     0.25     119    35
.        D[1,1]     0.08      47    47
.        D[2,2]      0.1     253   124
.        D[3,3]      0.2     289   120
.    SIGMA[1,1]     0.05      24    25
.    SIGMA[2,2]        1      41    41
.
. Total running time: 12.69 seconds
```

Add sample after the final (SS) dose

```r
poped_db_final <- create.poped.database(
  poped_db_extra_ss,
  xt = c(5, c(rep(24, 3), 72, 168)),
  minxt = c(0, c(rep(23, 3), 0, 168)),
  maxxt = c(6, c(rep(24, 3), 168, 168))
)
```

```r
FIM_final <- evaluate.fim(poped_db_final)
get_rse(FIM_final, poped_db_final)
```

```
.    bpop[1]    bpop[2]    bpop[3]     D[1,1]     D[2,2]     D[3,3] SIGMA[1,1] 
.   12.31062   86.75053  136.64068   50.42738  317.03881  419.93344   29.55819 
. SIGMA[2,2] 
.   28.95827 
```

D-optimal design: Add sample after the final (SS) dose

```r
output_final <- poped_optim(
  poped_db_final,
  opt_xt = TRUE,
  parallel = TRUE,
  parallel_type = "multicore",
  seed = 1
)
summary(output_final)
```

```
. ===========================================================
. FINAL RESULTS
. Optimized Sampling Schedule
. Group 1: Model 1: 0.4454
. Group 1: Model 2:     23     23     23  41.09    168
.
. OFV = 27.717
.
. Efficiency:
.   ((exp(ofv_final) / exp(ofv_init))^(1/n_parameters)) = 2.78
.
. Expected relative standard error
. (%RSE, rounded to nearest integer):
.     Parameter   Values   RSE_0   RSE
.       bpop[1]       10      12     9
.       bpop[2]      100      87    14
.       bpop[3]     0.25     137    17
.        D[1,1]     0.08      50    48
.        D[2,2]      0.1     317    74
.        D[3,3]      0.2     420    63
.    SIGMA[1,1]     0.05      30    29
.    SIGMA[2,2]        1      29    39
.
. Total running time: 11.964 seconds
```

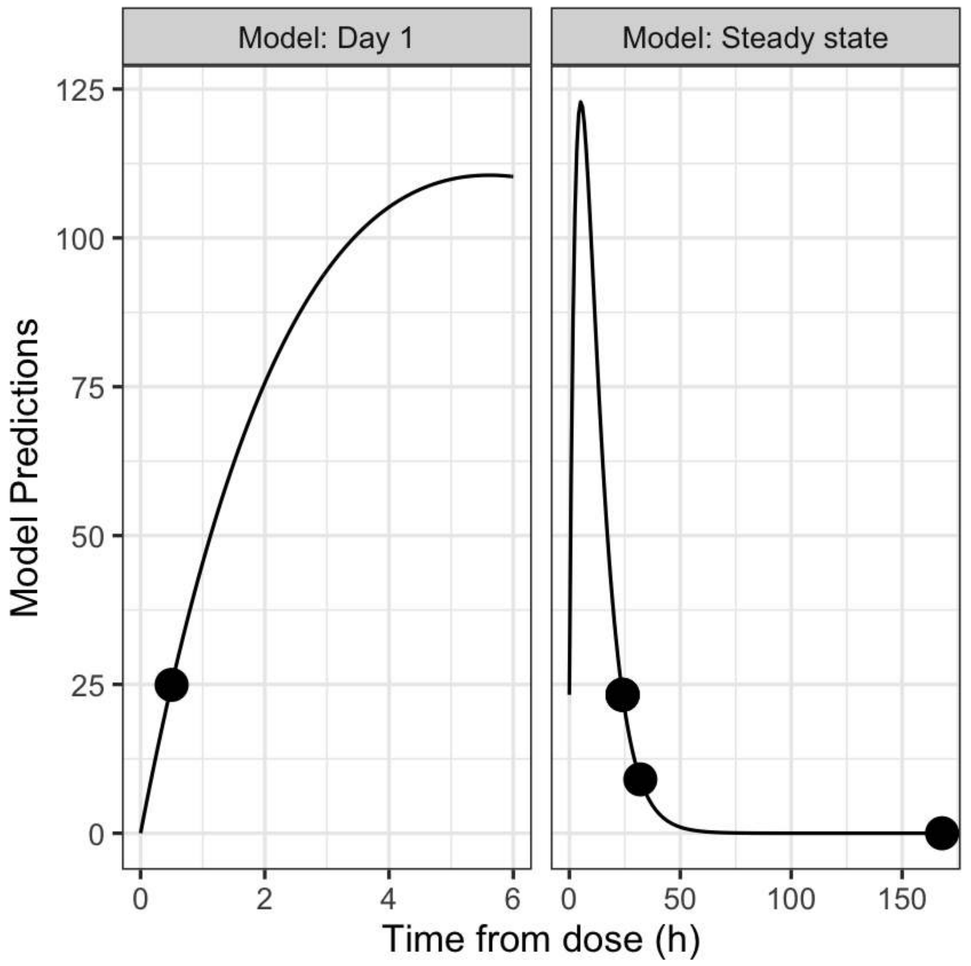
Near-optimal design

```
poped_db_practical <- create.poped.database(
  poped_db_final,
  xt = c(0.5, rep(24, 3), 32, 168)
)
```

```
FIM_practical <- evaluate.fim(poped_db_practical)
get_rse(FIM_practical, poped_db_practical)
```

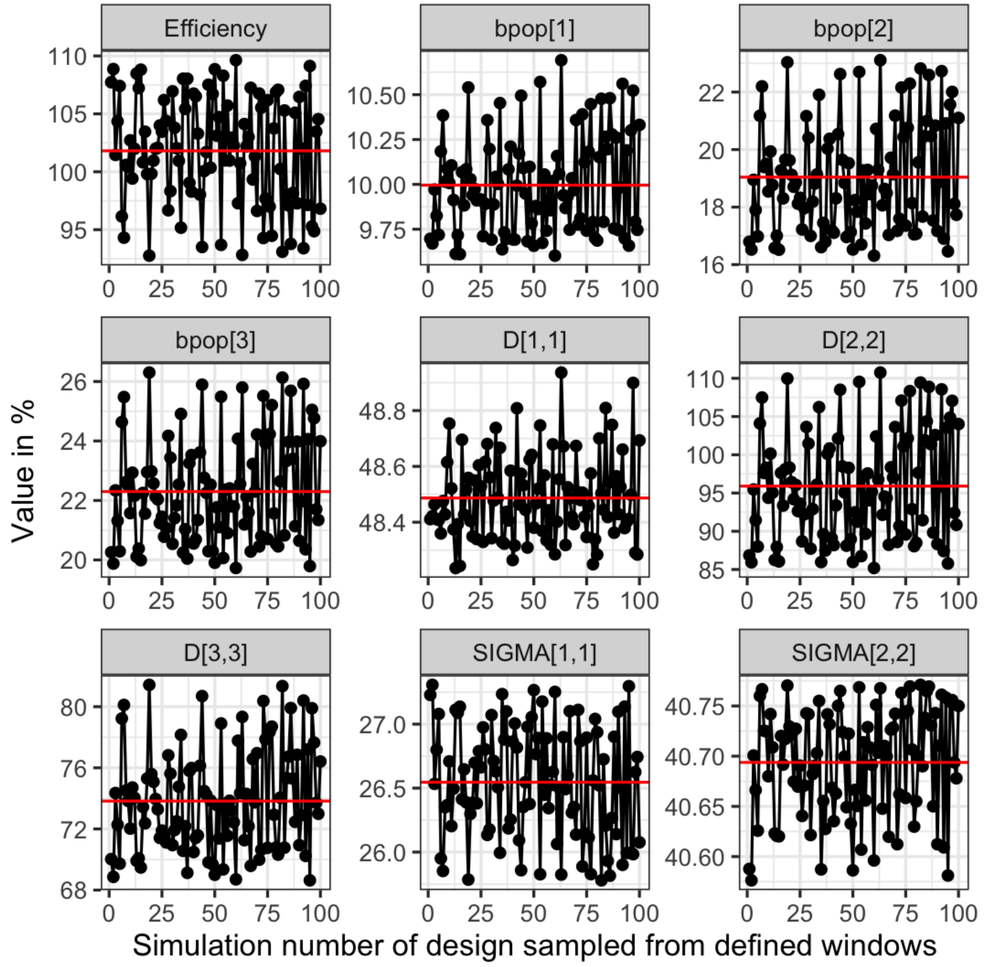
```
.     bpop[1]     bpop[2]     bpop[3]      D[1,1]      D[2,2]      D[3,3] SIGMA[1,1]
.    10.15307    19.61044    22.79548    48.67831    97.63716    74.56113   26.52843
.  SIGMA[2,2]
.    40.69749
```

```r
plot_model_prediction(
  poped_db_practical,
  model.names = c("Day 1", "Steady state"),
  facet_scales = "free_x",
  model_num_points = 200
) +
  labs(x = "Time from dose (h)") +
  theme_bw()
```

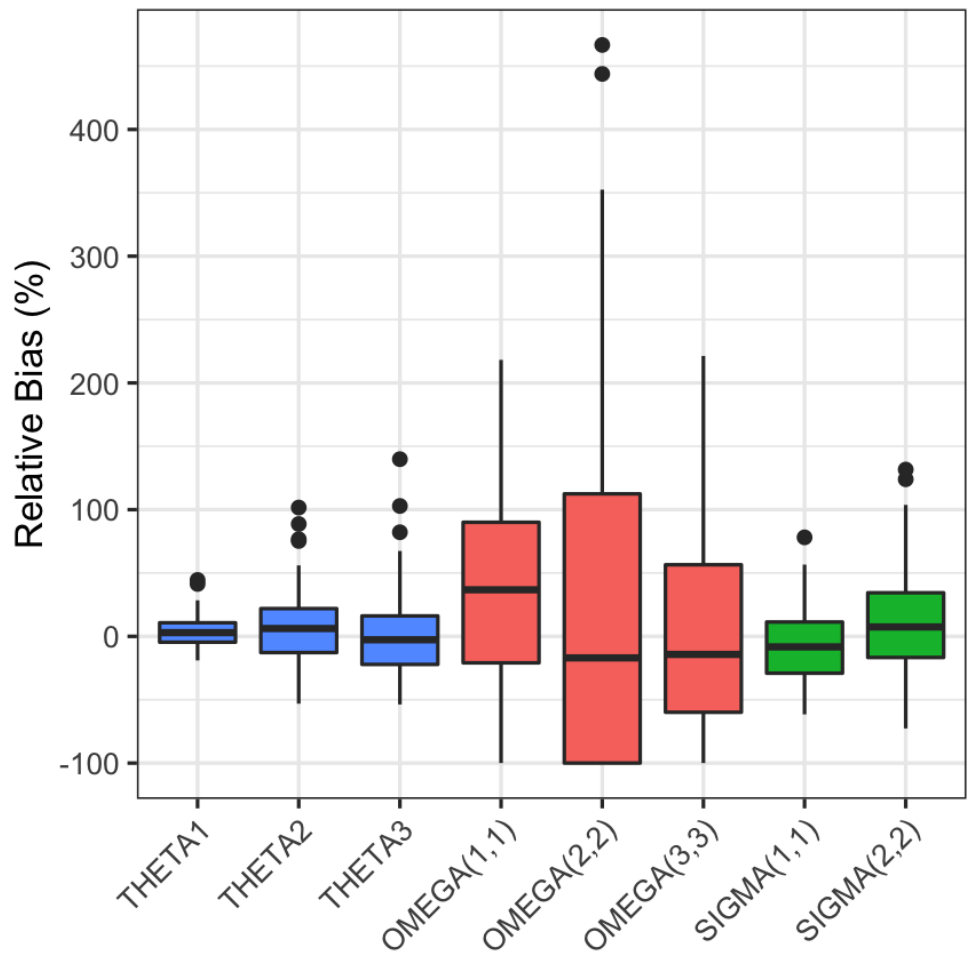


Sampling windows

```
plot_efficiency_of_windows(
  poped_db_practical,
  xt_plus  = c(0.25, rep(0, 3), 2, 0),
  xt_minus = c(0.25, rep(1, 3), 2, 4)
)
```



SSE Results



param	poped_pct_rse	sim_pct_rse	sim_pct_bias
THETA1	10.2	12.2	4.2
THETA2	19.6	29.0	6.7
THETA3	22.8	32.0	0.4
OMEGA(1,1)	48.7	74.2	41.7
OMEGA(2,2)	97.6	132.8	25.9
OMEGA(3,3)	74.6	90.0	7.2
SIGMA(1,1)	26.5	28.6	-6.7
SIGMA(2,2)	40.7	41.3	11.9

Example: ODE PK model

Study design and model

Fakinumab is being studied in humans for the first time.

- Single IV bolus doses of fakinumab: 0.03, 0.1, 0.3, 1, 3, and 10 mg.
- 6 subjects per dose group will be on active drug.
- Proposed samples: 1 and 4 hours post dose, and 1, 3, 7, 14, 21 days post dose.

Based on projections from animal PK data, we predict that a 2-compartment model with linear and nonlinear (Michaelis-Menten) clearance from the central compartment will desribe the data.

CL	VMAX	KM	V1	Q	V2
0.5	20	1.2	2.5	10	4

We include log-normal IIV on `CL`, `VMAX`, and `V1`, and a proportional residual error.

om_CL	om_VMAX	om_V1	sigma_prop
0.2	0.2	0.1	0.15

Model in `mrgsolve`

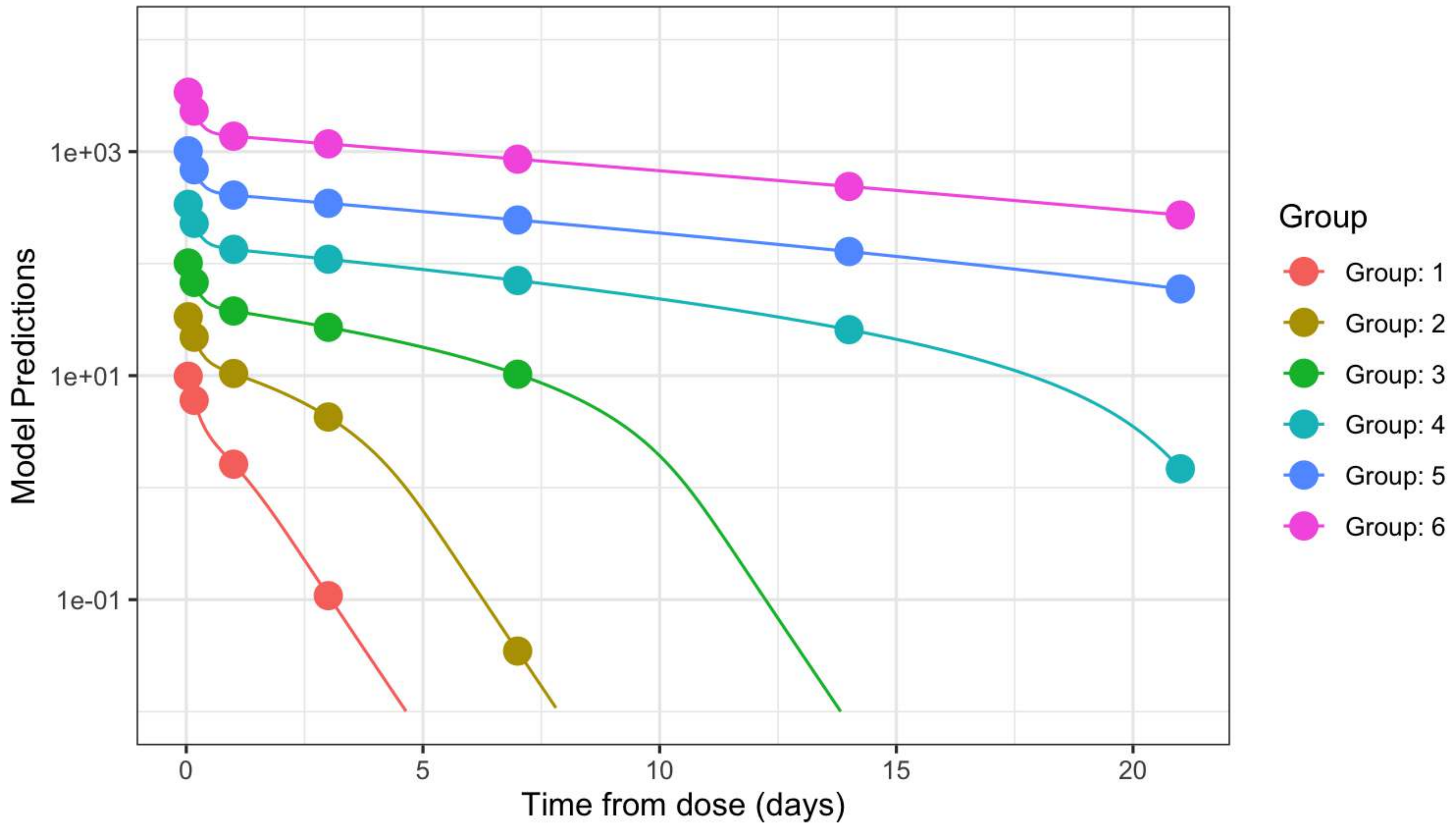
```
.
. Model file:  model_poped.cpp
. [ param ]
. CL = 1, VMAX = 10, KM = 10, V1 = 8, Q = 10, V2 = 100
.
. [ cmt ] CENT PERIPH
.
. [ main ]
. double ke  = CL/V1;
. double k12 = Q/V1;
. double k21 = Q/V2;
.
. [ ode ]
. double CP = CENT/V1;
.
. dxdt_CENT = k21*PERIPH - k12*CENT - VMAX*CP/(KM + CP) - ke*CENT;
. dxdt_PERIPH = k12*CENT - k21*PERIPH;
.
. [ capture ]
. CP
```

ff() for mrgsolve model

```
ff <- function(model_switch, xt, parameters, poped.db) {
  obs_time <- as.numeric(xt)
  dose_time <- 0

  dose <- data.frame(
    ID = 1,
    amt = parameters[["DOSE"]]*1000,
    cmt = 1,
    evid = 1,
    time = dose_time
  )
  obs <- data.frame(
    ID = 1,
    amt = 0,
    cmt = 1,
    evid = 0,
    time = sort(obs_time)
  )
```

```
  data <- arrange(bind_rows(dose,obs),time)

  mod <- param(mod, parameters)

  out <- mrgsim_q(mod,data,output="matrix")

  out <- out[data$evid==0,"CP",drop=FALSE][match(obs_time,obs$t

  return(list(y = out, poped.db = poped.db))
}
```

Plot of initial design



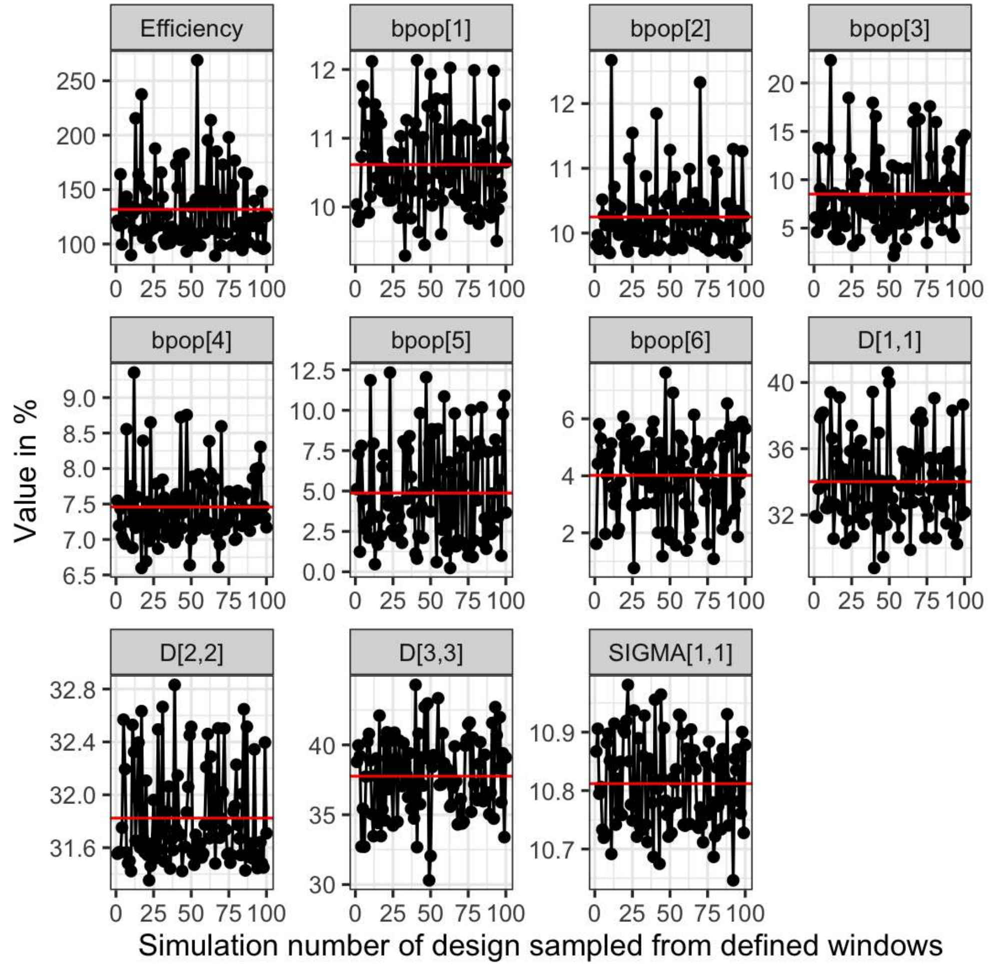
Evaluate FIM

```r
FIM_mrg <- evaluate.fim(poped_db_mrg)
get_rse(FIM_mrg, poped_db_mrg)
```

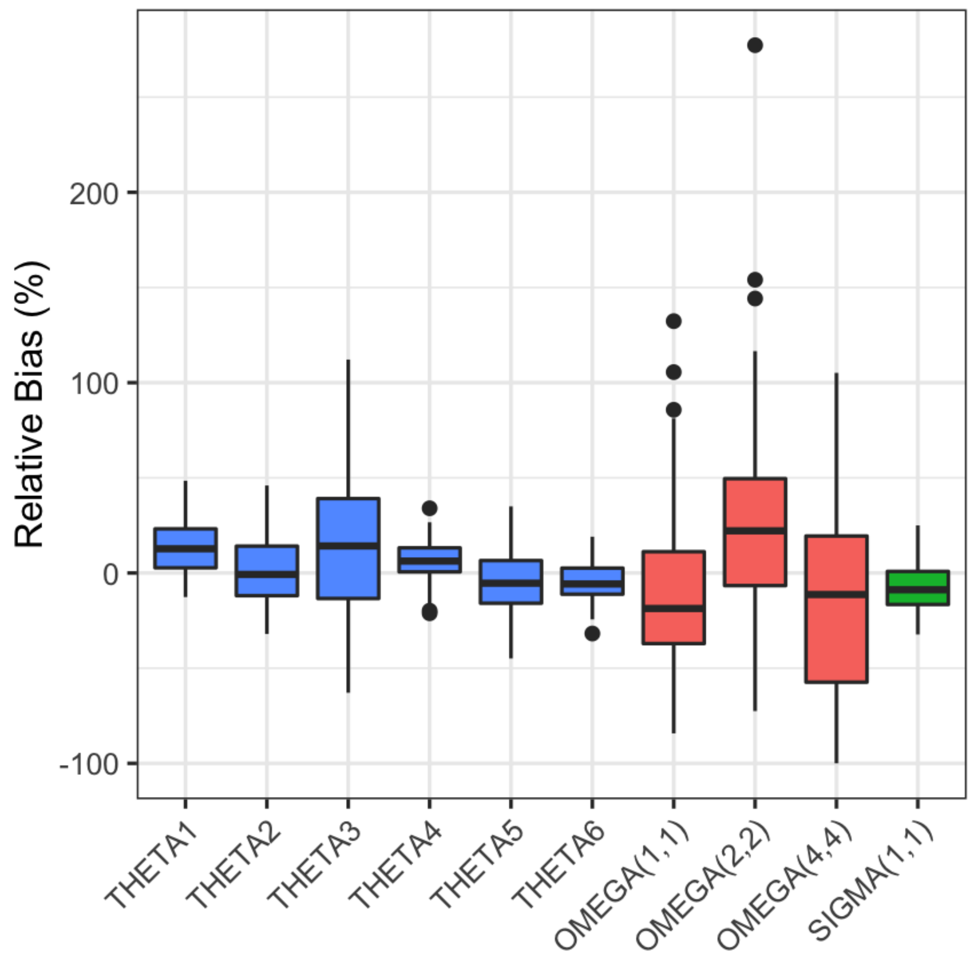
```
.    bpop[1]    bpop[2]    bpop[3]    bpop[4]    bpop[5]    bpop[6]     D[1,1]
.  10.874541  10.660202   7.935122   6.355306   8.894763   3.620234  38.990317
.     D[2,2]     D[3,3] SIGMA[1,1]
.  32.096512  31.668012  10.775738
```

Sampling windows

```r
plot_efficiency_of_windows(
  poped_db_mrg,
  xt_plus  = c(rep(1/24, 2), rep(3/24, 5)),
  xt_minus = c(rep(1/24, 2), rep(3/24, 5))
)
```



SSE results

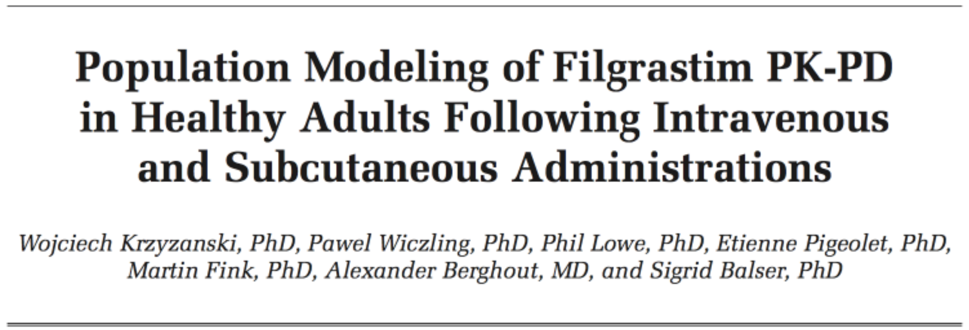


param	poped_pct_rse	sim_pct_rse	sim_pct_bias
THETA1	10.9	13.9	13.6
THETA2	10.7	17.1	1.8
THETA3	7.9	38.4	15.5
THETA4	6.4	10.4	6.6
THETA5	8.9	16.9	-5.8
THETA6	3.6	9.6	-4.4
OMEGA(1,1)	39.0	38.9	-11.9
OMEGA(2,2)	32.1	51.1	24.6
OMEGA(4,4)	31.7	52.3	-12.6
SIGMA(1,1)	10.8	12.3	-7.0

Example: ODE PK/PD model

Study design & model

- QD SC doses of **filgrastim**: 1, 3, and 10 μg/kg.
- 10 subjects per dose group will be on active drug.
- Dense PK and ANC samples on days 1 and 7.



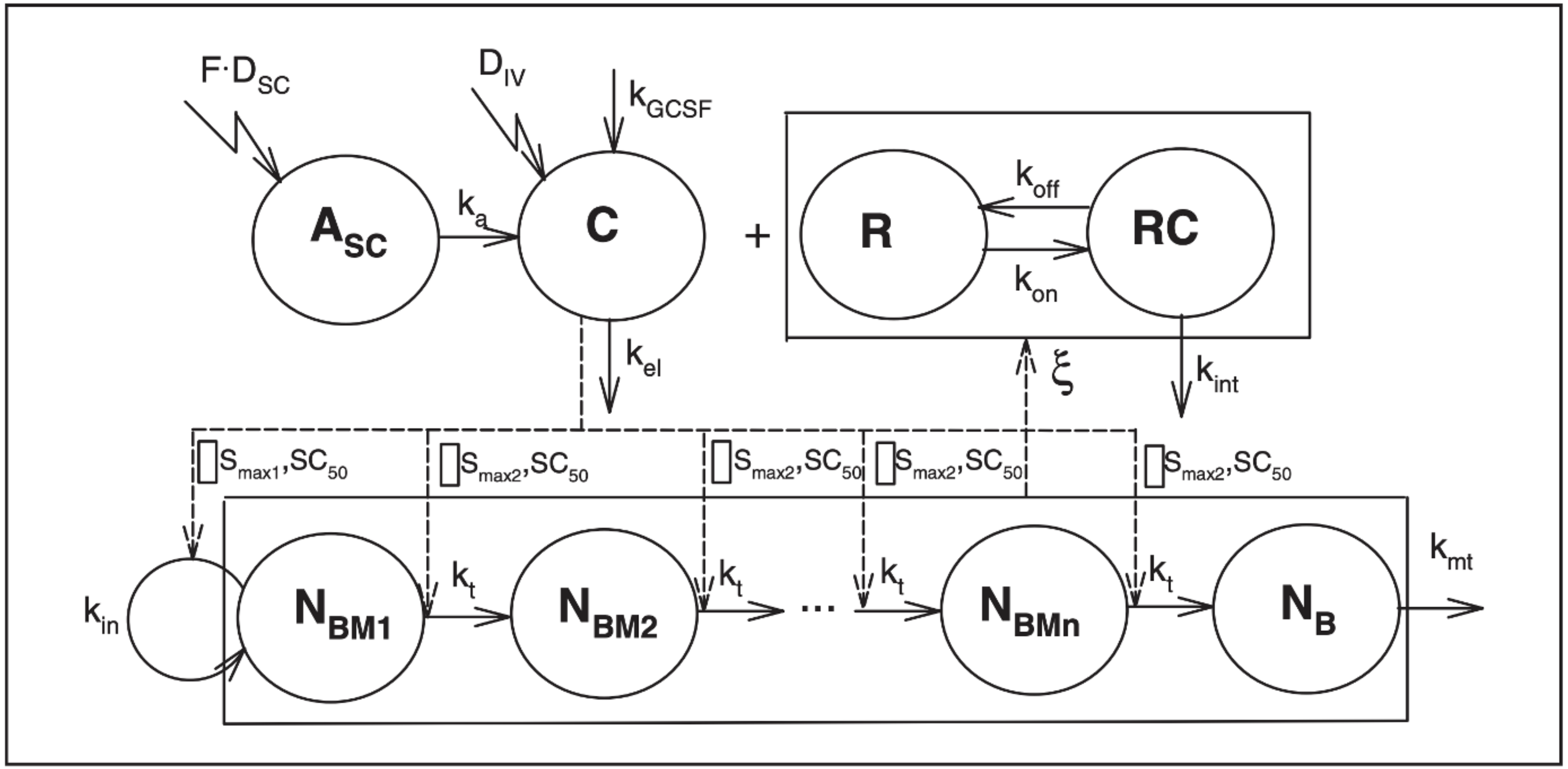
Population Modeling of Filgrastim PK-PD in Healthy Adults Following Intravenous and Subcutaneous Administrations

Wojciech Krzyzanski, PhD, Pawel Wiczling, PhD, Phil Lowe, PhD, Etienne Pigeolet, PhD, Martin Fink, PhD, Alexander Berghout, MD, and Sigrid Balser, PhD

- PK/PD model from DDMORE
  http://repository.ddmore.eu/model/DDMODEL00000077.6
- NONMEM model translated to `mrgsolve`
  https://github.com/mrgsolve/depot/blob/master/vignette/gcsf.md

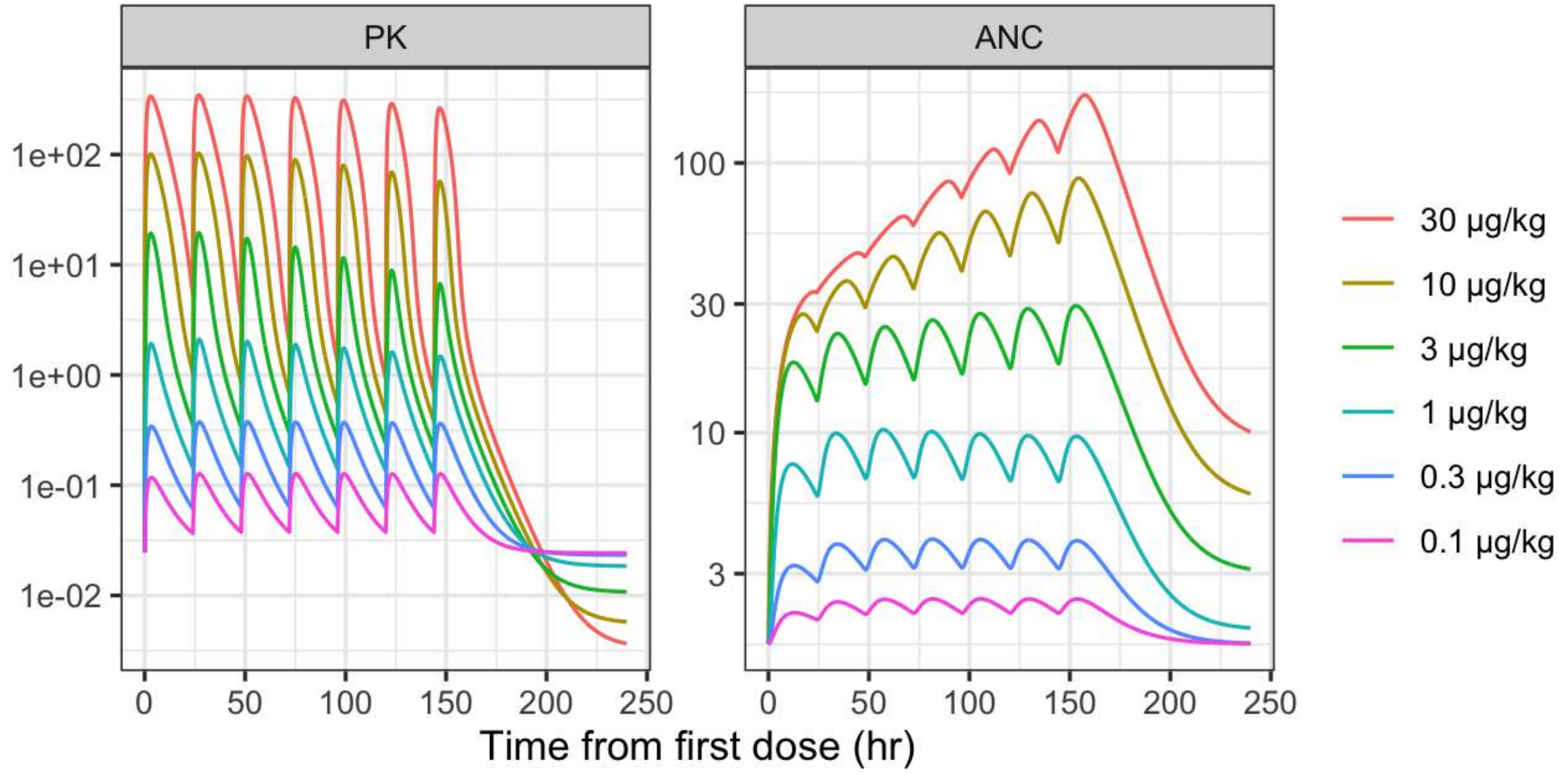
Krzyzanski, Wiczling, Lowe, et al. (2010)

Filgrastim ANC model



Krzyzanski, Wiczling, Lowe, et al. (2010)

PK and ANC simulations



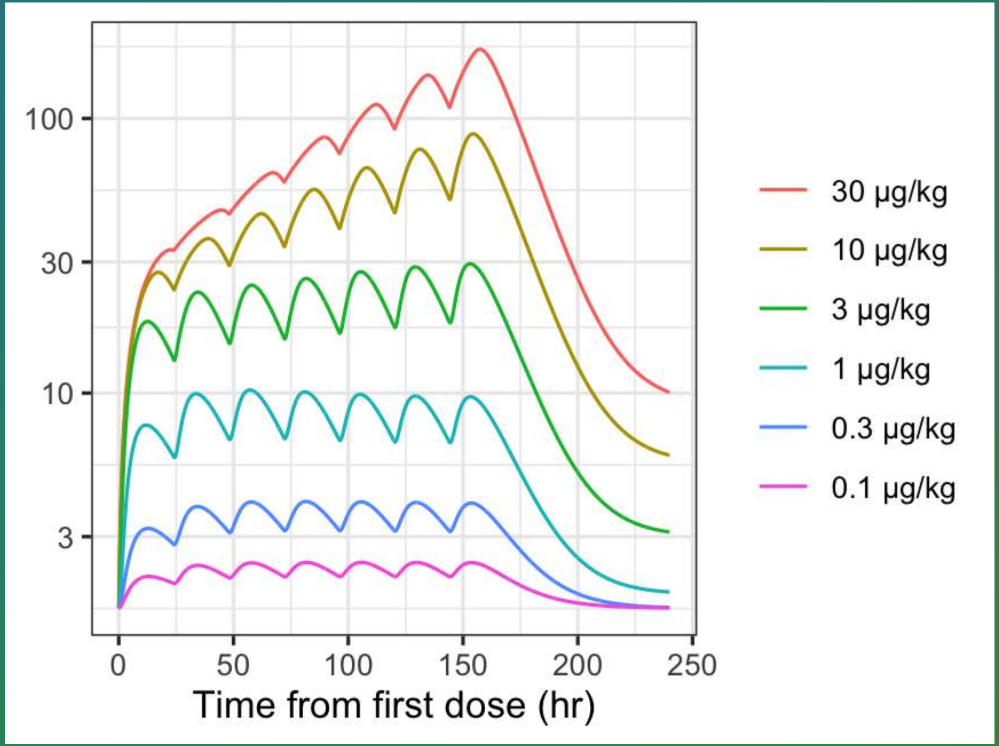
Evaluate FIM for initial design

```r
FIM_gcsf <- evaluate.fim(poped_db_gcsf)
get_rse(FIM_gcsf, poped_db_gcsf)
```

```
.     bpop[2]     bpop[5]     bpop[6]     bpop[7]     bpop[8]     bpop[9]    bpop[11] 
.    3.289140   13.518742   12.604107    8.265412    3.822461   19.528269    4.066511 
.    bpop[13]    bpop[14]    bpop[15]    bpop[16]    bpop[17]      D[1,1]      D[2,2] 
.   17.017908   10.771361   20.851250    8.289988    7.664966   26.961113   32.644613 
.      D[3,3]      D[5,5]      D[6,6]      D[7,7] SIGMA[1,1] SIGMA[3,3] SIGMA[4,4] 
.   31.369081   41.941250   29.156946   26.406506    3.900224    6.541108    7.713365 
```

Poll Question #3

Any better doses for estimating SC50 or Smax?



1. 1, 3, 10 µg/kg
2. 0.3, 1, 3 µg/kg
3. 0.3, 1, 10 µg/kg
4. 0.1, 1, 3 µg/kg
5. Are you still talking? I think you're on mute.

Poll Question #3

Any better doses for estimating SC_{50} or $S_{\max}$?

Parameter	Design 1	Design 2	Design 3	Design 4
bpop[15]	20.9	24.8	21.8	26.4
bpop[16]	8.3	13.6	9.5	13.9
bpop[17]	7.7	13.1	8.7	13.4

Wrap up

What did we learn today?

- Optimal design can be a useful, if imperfect, tool to explore and optimize study design options
- Always run confirmatory simulations
- Optimal design: it's not just for PK sampling any more!

What did we miss?

- Dealing with parameter uncertainty (e.g., ED-, $HC\ln D$-optimality)
- Dealing with model uncertainty (e.g. T-optimality)
- Ignoring unimportant parameters (e.g. D_S-, G-optimality)
- Basically an alphabet of other criteria: A-, C-, G-, V-optimality, etc.
- Almost all of `PopED`'s options

Software resources

- PopED: https://andrewhooker.github.io/PopED/
- mrgsolve: https://mrgsolve.github.io/
- babylon: https://github.com/metrumresearchgroup/babylon
    - rbabylon: https://metrumresearchgroup.github.io/rbabylon/

References

Foracchia, M, A. Hooker, P. Vicini, et al. (2004). "POPED, a software for optimal experiment design in population kinetics". In: *Comput. Methods Programs Biomed.* 74.1, pp. 29-46.

Krzyzanski, W, P. Wiczling, P. Lowe, et al. (2010). "Population modeling of filgrastim PK-PD in healthy adults following intravenous and subcutaneous administrations". In: *J. Clin. Pharmacol.* 50.9 Suppl, pp. 101S-112S.

Mentre, F, A. Mallet, and D. Baccar (1997). "Optimal design in random-effects regression models". In: *Biometrika* 84.2, pp. 429-442.

Nyberg, J, C. Bazzoli, K. Ogungbenro, et al. (2014). "Methods and software tools for design evaluation for population pharmacokinetics-pharmacodynamics studies". In: *Br. J. Clin. Pharmacol.*.

Nyberg, J, S. Ueckert, E. A. Strömberg, et al. (2012). "PopED: an extended, parallelized, nonlinear mixed effects models optimal design tool". In: *Comput. Methods Programs Biomed.* 108.2, pp. 789-805.

Retout, S, S. Duffull, and F. Mentre (2001). "Development and implementation of the population Fisher information matrix for the evaluation of population pharmacokinetic designs". In: *Comput. Methods Programs Biomed.* 65.2, pp. 141-151.

Retout, S. and F. Mentre (2003). "Further developments of the Fisher information matrix in nonlinear mixed effects models with evaluation in population pharmacokinetics". In: *J. Biopharm. Stat.* 13.2, pp. 209-227.

Thank you

Backup slides

Simulation

SSE: template.csv

```
$PROB RUN# run_num
$INPUT ID TIME EVID MDV CMT AMT SS II DV WT
$DATA data_fname IGNORE=@
...
$SIMULATION (run_num)
$ESTIMATION METHOD=1 INTER PRINT=1 MSFO=./run_num.msf
```

SSE: Run the models with `rbabylon`

```r
run_model <- function(.design_dir, .run_num) {
  # Modify and write the control stream
  ctl_template %>%
    str_replace("run_num", as.character(.run_num)) %>%
    str_replace("data_fname", paste0("../", .run_num, ".csv")) %>%
    writeLines(file.path(.design_dir, paste0(.run_num, ".ctl")))

  # Run the model
  new_model(
    .yaml_path = paste0(.run_num, ".yaml"),
    .description = .run_num,
    .directory = design_dir
  ) %>%
    submit_model()
}
```

SSE: Collect the results with `rbabylon`

```r
get_est <- function(.design_dir) {
  est <- map_dfr(seq_len(n_rep), function(.run_num) {
    mod <- read_model(paste0(.run_num, ".yaml"), .directory = .design_dir)
    mod_sum <- try(model_summary(mod), silent = TRUE)
    if (inherits(mod_sum, "try-error")) return(NULL)
    mod_sum %>%
      param_estimates() %>%
      filter(fixed == 0) %>%
      select(param = names, estimate) %>%
      mutate(rep = .run_num)
  }) %>%
    left_join(true_vals) %>%
    mutate(
      param = factor(param, levels = unique(.[["param"]])),
      pct_bias = (estimate - value) / abs(value) * 100
    ) %>%
    filter(abs(pct_bias) < 5000)
  return(est)
}
```

PopED setup

ff(): the structural model

```r
ff <- function(model_switch, xt, parameters, poped.db){
  with(as.list(parameters),{

    CL <- CL*(WT/70)^(WT_CL)
    V <- V*(WT/70)^(WT_V)

    y_sd <- (DOSE * KA/(V * (KA - CL/V))) *
      (exp(-CL/V * xt) - exp(-KA * xt))

    y_ss <- (DOSE * KA/(V * (KA - CL/V))) *
      (exp(-CL/V * xt) / (1 - exp(-CL/V * TAU)) -
         exp(-KA * xt) / (1 - exp(-KA * TAU)))

    y <- xt
    y[model_switch == 1] <- y_sd[model_switch == 1]
    y[model_switch == 2] <- y_ss[model_switch == 2]

    return(list(y = y, poped.db = poped.db))
  })
}
```

PopED expects a function with the following arguments:

- `model_switch`: A vector of values identifying which model response should be computed for the corresponding `xt` value
- `xt`: A vector of independent variable values (often time).
- `parameters`: A named list of parameter values.
- `poped.db`: A PopED database.

fg(): the parameter model

```r
fg <- function(x, a, bpop, b, bocc){
  parameters = c(
    CL    = bpop[1] * exp(b[1]),
    V     = bpop[2] * exp(b[2]),
    KA    = bpop[3] * exp(b[3]),
    WT_CL = bpop[4],
    WT_V  = bpop[5],
    DOSE  = a[1] * 1000,
    TAU   = a[2],
    WT    = a[3]
  )
  return(parameters)
}
```

- `x`: A vector of discrete design variables (not used here).
- `a`: A vector of covariates.
- `bpop`: A vector of fixed effect parameters (i.e., `THETA`s).
- `b`: A vector of individual IIV random effects (i.e., `ETA`s).
- `bocc`: A vector of individual IOV random effects (i.e., `ETA`s) (not used here).

In this example, we include IIV on `CL`, `V`, and `KA`, and pass through dose, tau, and body weight as covariates.

`feps()`: the residual error model

```
feps <- function(model_switch, xt, parameters, epsi, pc
  returnArgs <- do.call(
    poped.db$model$ff_pointer,
    list(model_switch, xt, parameters, poped.db)
  )
  y <- returnArgs[[1]]
  poped.db <- returnArgs[[2]]
  y = y * exp(epsi[, 1])
  return(list(y = y, poped.db = poped.db))
}
```

- `epsi`: A matrix of residual random effects (i.e. `EPSs` or `ERRs`).

create.poped.database()

```r
poped_db <- create.poped.database(
  ff_fun = ff,
  fg_fun = fg,
  fError_fun = feps.add.prop,
  bpop = c(CL = 10, V = 100, KA = 0.25, WT_CL = 0.75, WT_V = 1),
  notfixed_bpop = c(1, 1, 1, 0, 0),
  d = c(CL = 0.08, V = 0.1, KA = 0.2),
  sigma = c(0.05, 1),
  m = 1,
  groupsize = 12,
  xt = c(5, c(rep(24, 3), 168)),
  minxt = c(0, c(rep(23, 3), 96)),
  maxxt = c(6, c(rep(24, 3), 168)),
  model_switch = c(1, rep(2, 4)),
  a = cbind(DOSE = 10, TAU = 24, WT = 32)
)
```

create.poped.database()

```r
poped_db <- create.poped.database(
  ff_fun = ff,
  fg_fun = fg,
  fError_fun = feps.add.prop,
  bpop = c(CL = 10, V = 100, KA = 0.25, WT_CL = 0.75, W
  notfixed_bpop = c(1, 1, 1, 0, 0),
  d = c(CL = 0.08, V = 0.1, KA = 0.2),
  sigma = c(0.05, 1),
  m = 1,
  groupsize = 12,
  xt = c(5, c(rep(24, 3), 168)),
  minxt = c(0, c(rep(23, 3), 96)),
  maxxt = c(6, c(rep(24, 3), 168)),
  model_switch = c(1, rep(2, 4)),
  a = cbind(DOSE = 10, TAU = 24, WT = 32)
)
```

- `ff_fun`, `fg_fun`, `fError_fun`: Model functions

create.poped.database()

```r
poped_db <- create.poped.database(
  ff_fun = ff,
  fg_fun = fg,
  fError_fun = feps.add.prop,
  bpop = c(CL = 10, V = 100, KA = 0.25, WT_CL = 0.75, W
  notfixed_bpop = c(1, 1, 1, 0, 0),
  d = c(CL = 0.08, V = 0.1, KA = 0.2),
  sigma = c(0.05, 1),
  m = 1,
  groupsize = 12,
  xt = c(5, c(rep(24, 3), 168)),
  minxt = c(0, c(rep(23, 3), 96)),
  maxxt = c(6, c(rep(24, 3), 168)),
  model_switch = c(1, rep(2, 4)),
  a = cbind(DOSE = 10, TAU = 24, WT = 32)
)
```

- `bpop`: our current best estimates of the fixed effect parameters (`THETAs`)
- `notfixed_bpop`: whether or not they're being estimated
- `d`: diagonal elements of the IIV covariance matrix (`OMEGA`)
- `sigma`: diagonal elements of the residual covariance matrix (`SIGMA`)

create.poped.database()

```r
poped_db <- create.poped.database(
  ff_fun = ff,
  fg_fun = fg,
  fError_fun = feps.add.prop,
  bpop = c(CL = 10, V = 100, KA = 0.25, WT_CL = 0.75, W
  notfixed_bpop = c(1, 1, 1, 0, 0),
  d = c(CL = 0.08, V = 0.1, KA = 0.2),
  sigma = c(0.05, 1),
  m = 1,
  groupsize = 12,
  xt = c(5, c(rep(24, 3), 168)),
  minxt = c(0, c(rep(23, 3), 96)),
  maxxt = c(6, c(rep(24, 3), 168)),
  model_switch = c(1, rep(2, 4)),
  a = cbind(DOSE = 10, TAU = 24, WT = 32)
)
```

- `m`: number of groups
- `groupsize`: number of subjects in each group

create.poped.database()

```r
poped_db <- create.poped.database(
  ff_fun = ff,
  fg_fun = fg,
  fError_fun = feps.add.prop,
  bpop = c(CL = 10, V = 100, KA = 0.25, WT_CL = 0.75, W
  notfixed_bpop = c(1, 1, 1, 0, 0),
  d = c(CL = 0.08, V = 0.1, KA = 0.2),
  sigma = c(0.05, 1),
  m = 1,
  groupsize = 12,
  xt = c(5, c(rep(24, 3), 168)),
  minxt = c(0, c(rep(23, 3), 96)),
  maxxt = c(6, c(rep(24, 3), 168)),
  model_switch = c(1, rep(2, 4)),
  a = cbind(DOSE = 10, TAU = 24, WT = 32)
)
```

- `xt`: initial sampling design
- `minxt`: lower bound
- `maxxt`: upper bound
- `model_switch`: associate sampling times with model

create.poped.database()

```
poped_db <- create.poped.database(
  ff_fun = ff,
  fg_fun = fg,
  fError_fun = feps.add.prop,
  bpop = c(CL = 10, V = 100, KA = 0.25, WT_CL = 0.75, W
  notfixed_bpop = c(1, 1, 1, 0, 0),
  d = c(CL = 0.08, V = 0.1, KA = 0.2),
  sigma = c(0.05, 1),
  m = 1,
  groupsize = 12,
  xt = c(5, c(rep(24, 3), 168)),
  minxt = c(0, c(rep(23, 3), 96)),
  maxxt = c(6, c(rep(24, 3), 168)),
  model_switch = c(1, rep(2, 4)),
  a = cbind(DOSE = 10, TAU = 24, WT = 32)
)
```

- a: covariates