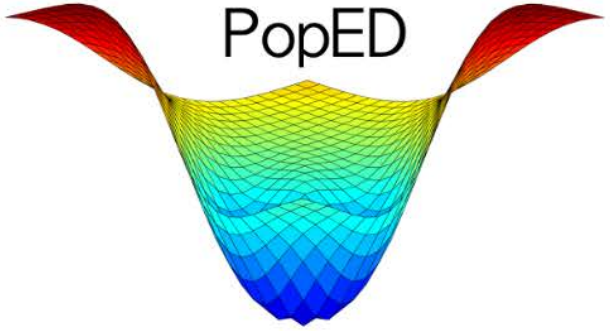
A Gentle Introduction to Optimal Design for Pharmacometric Models



with PopED, PFIM, and mrgsolve

Tim Waterhouse & Kyle Baron
Metrum Research Group

Julie Bertrand & Romain Leroux
Université de Paris, INSERM, IAME

28 March, 2023

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` and `PFIM`
    - Evaluation
    - Optimization
    - Simulation
- More complex models with `PopED` and `mrgsolve`

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 population 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:

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

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

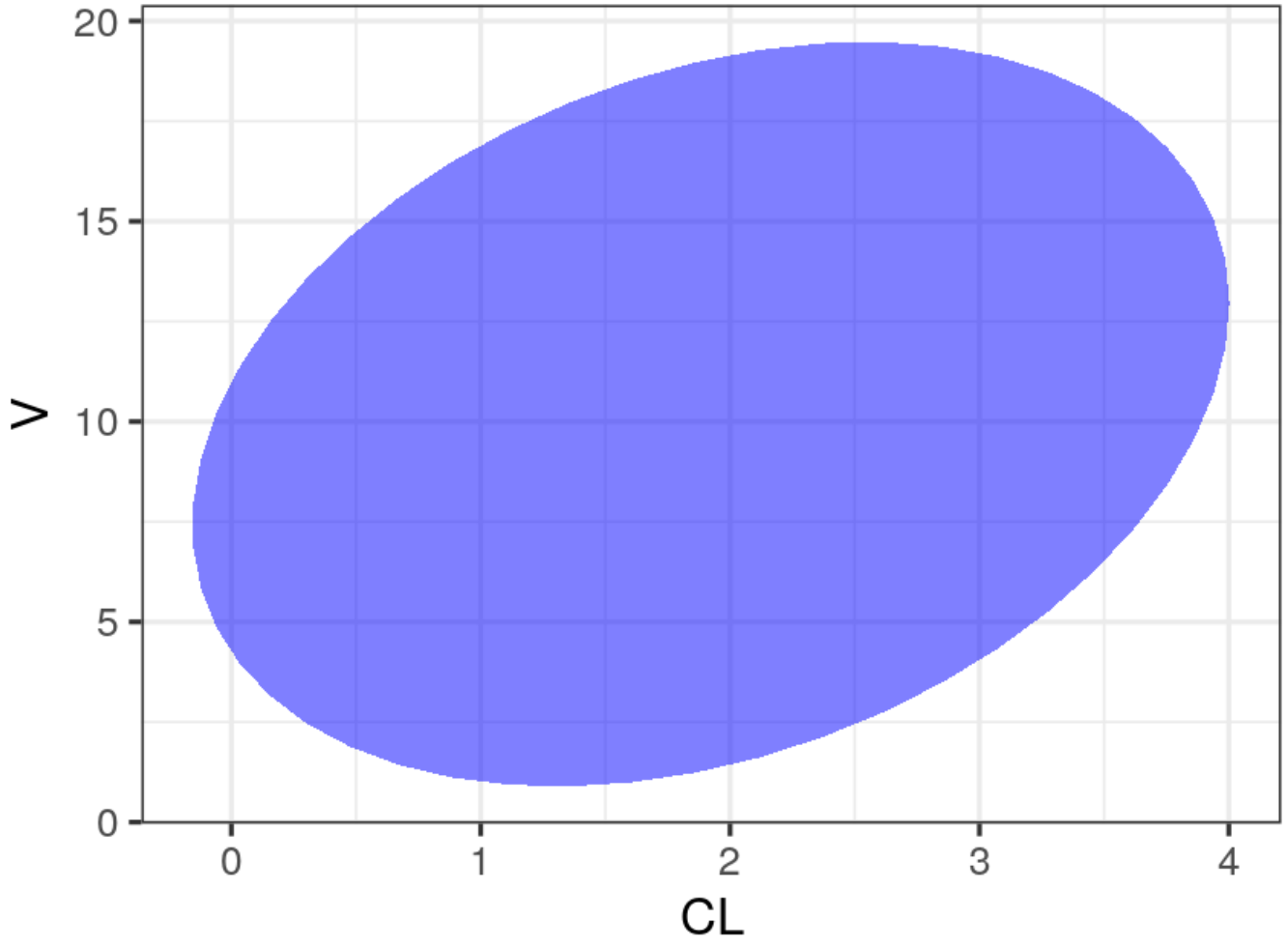
- Lower bounds for relative standard errors (RSEs) can be obtained from the diagonals of the inverse of the FIM
- This means we have a quick way of evaluating (lower bounds on) the precision of our parameter estimates.

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 $M_F(\boldsymbol{\Psi}, \xi)$ on $\boldsymbol{\Psi}$ 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

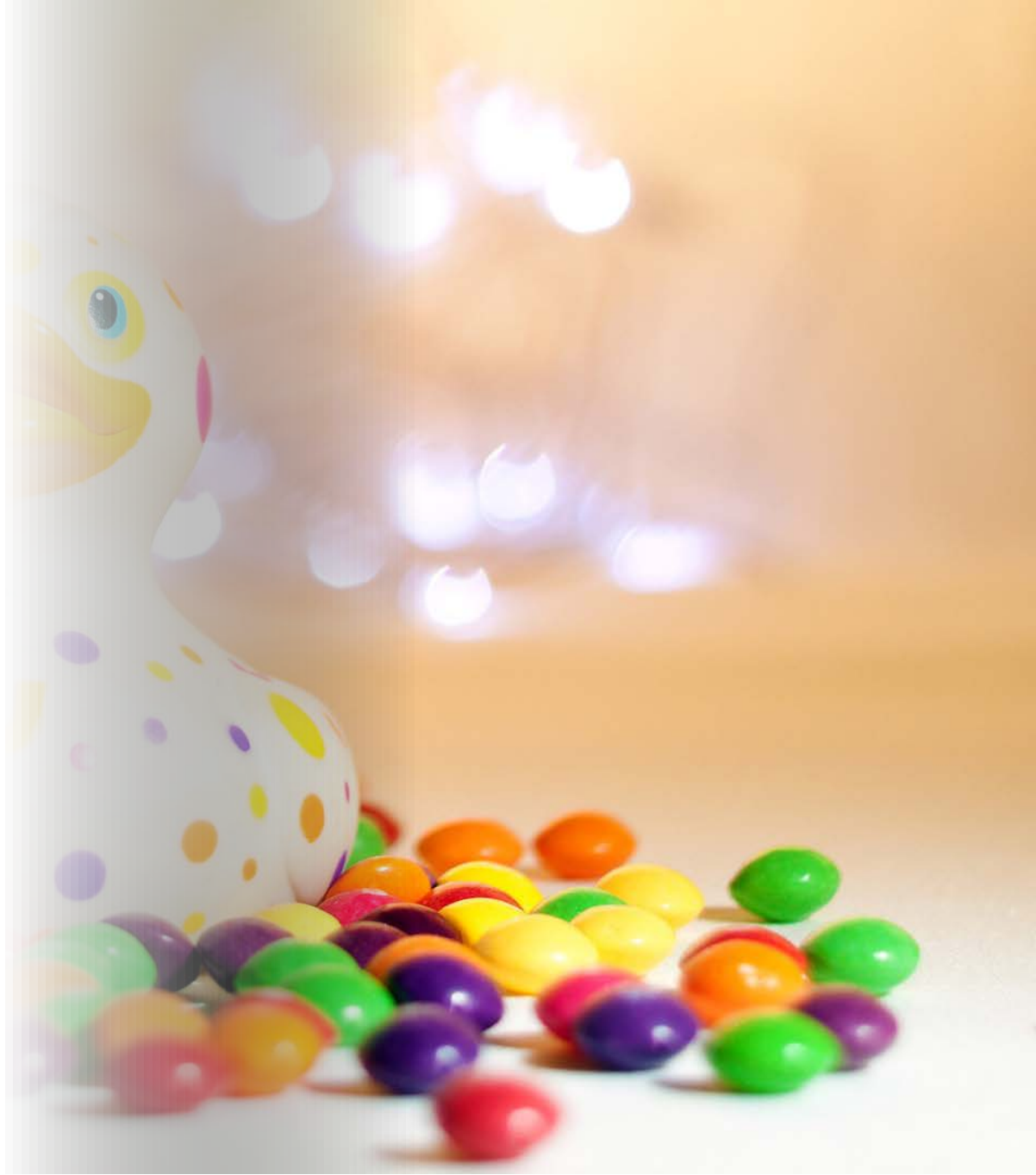
- No analytic expression for the likelihood, so we rely on approximations
- So our FIM is
  - an approximation
  - to a lower bound
  - that depends on the parameter values

- 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…

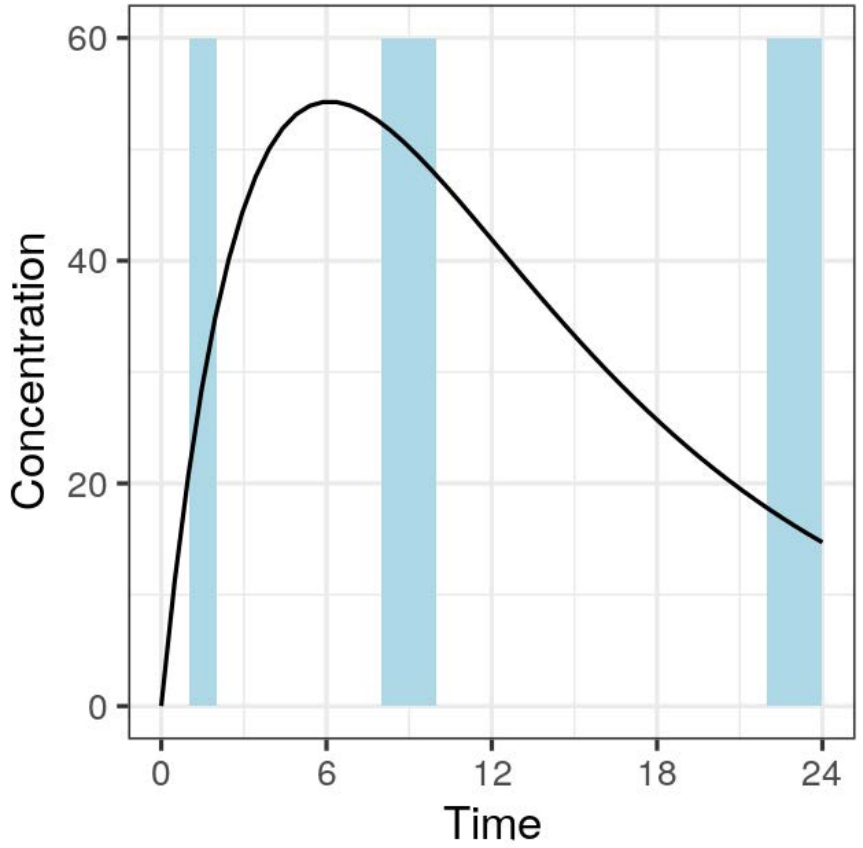
Mentre, Mallet, and Baccar (1997)
Retout, Duffull, and Mentre (2001)
Retout and Mentre (2003)

Evaluation vs Optimisation

- Optimal design can be used to optimise a study design (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

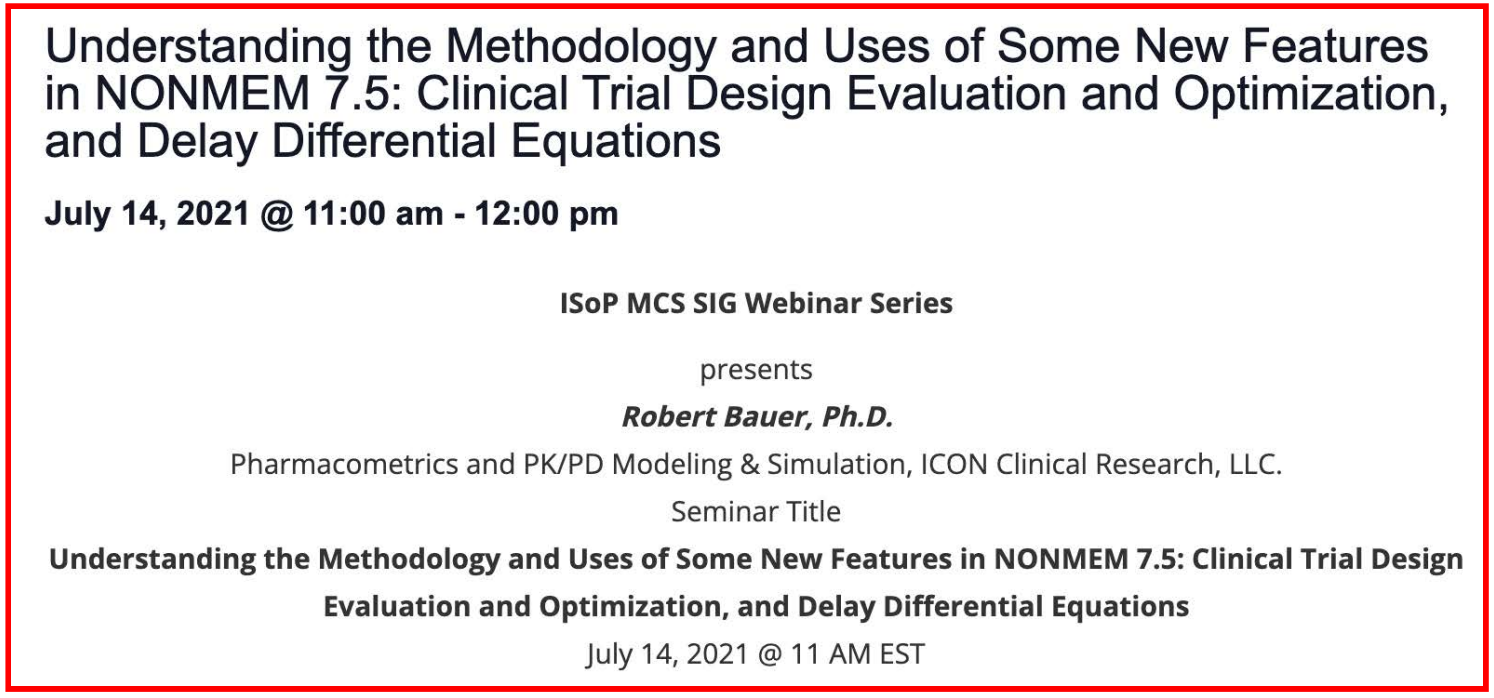
Tools for optimal design

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, Aliev, Leonov, Duffull, Hooker, and Mentré (2014)

Notable exception: NONMEM $DESIGN



Understanding the Methodology and Uses of Some New Features in NONMEM 7.5: Clinical Trial Design Evaluation and Optimization, and Delay Differential Equations

July 14, 2021 @ 11:00 am - 12:00 pm

ISoP MCS SIG Webinar Series

presents

Robert Bauer, Ph.D.

Pharmacometrics and PK/PD Modeling & Simulation, ICON Clinical Research, LLC.

Seminar Title

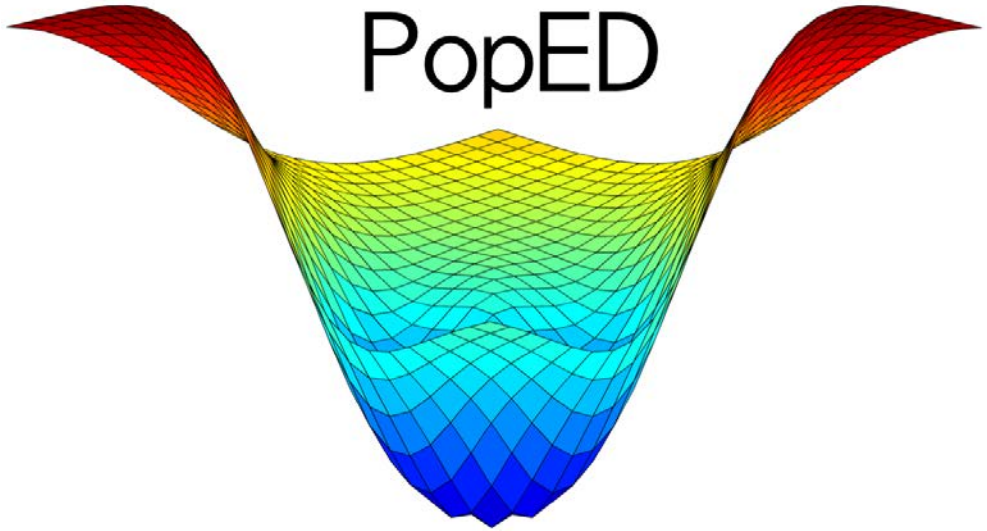
Understanding the Methodology and Uses of Some New Features in NONMEM 7.5: Clinical Trial Design Evaluation and Optimization, and Delay Differential Equations

July 14, 2021 @ 11 AM EST

Recording: https://sites.google.com/view/mcssig/mcs-sig#h.v7bdq6l7d86v

Bauer, Hooker, and Mentre (2021)

PopED



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

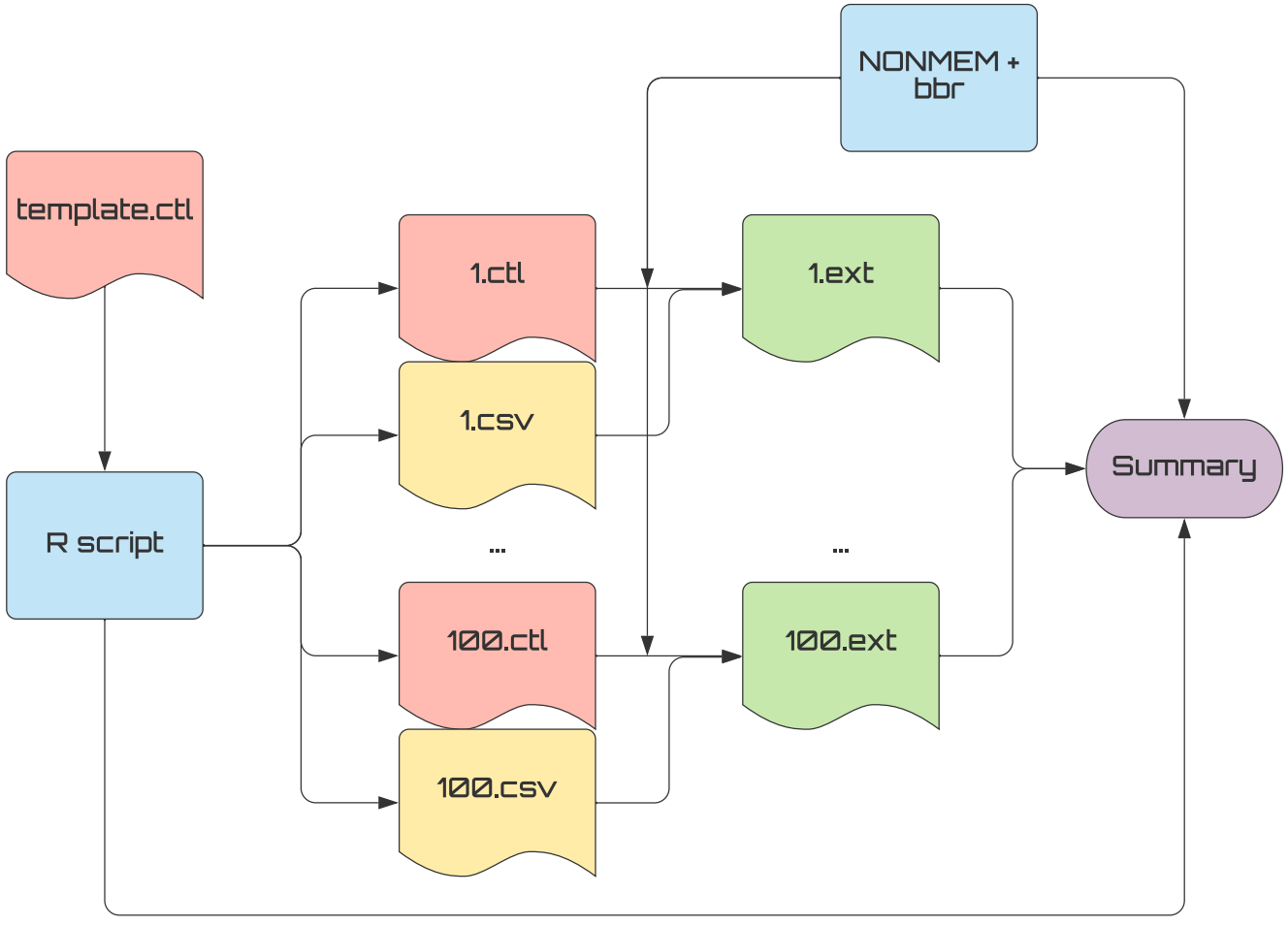
Foracchia, Hooker, Vicini, and Ruggeri (2004)
Nyberg, Ueckert, Strömberg, Hennig, Karlsson, and Hooker (2012)

PFIM

- http://www.pfim.biostat.fr/
- Previously set of R functions, now an R package (Version 5.0)
- Version 4.0 also available as GUI: PFIM Interface

Leroux, Seurat, Nagard, Mentré, and PFIM group (2022)

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

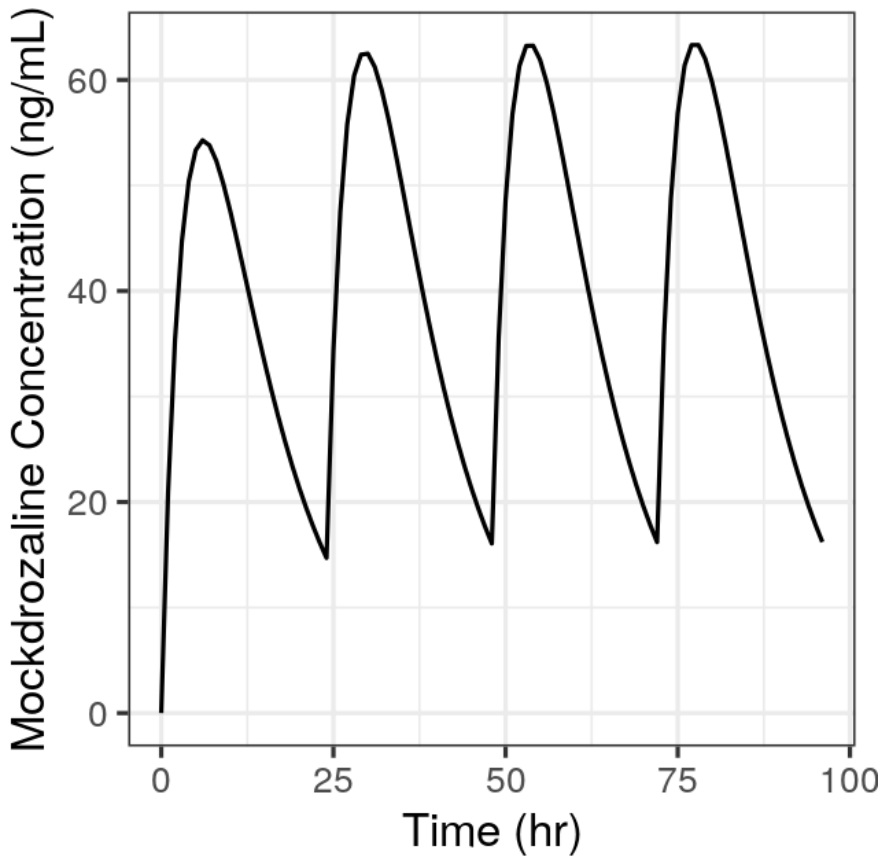
1-compartment model with 1st-order absorption and weight covariates on CL and V:

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:

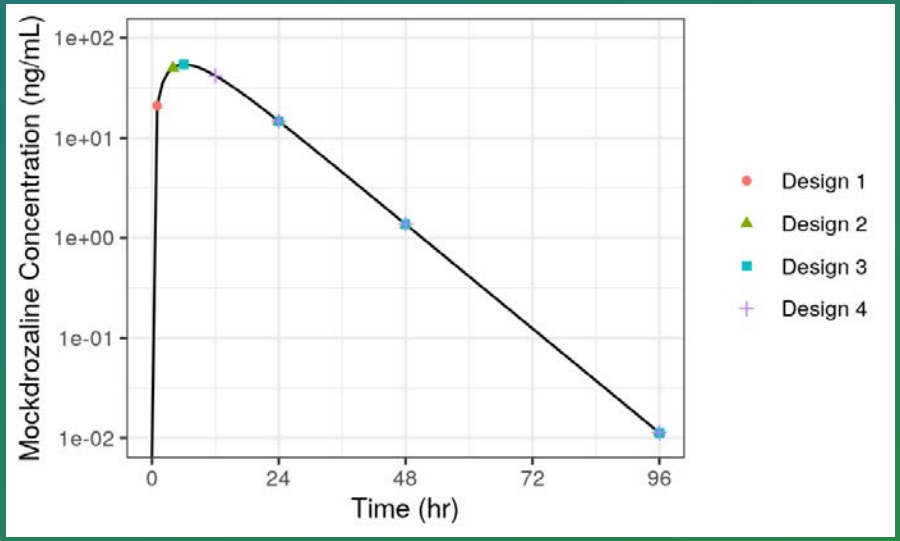
om_CL	om_V	om_KA	sigma_prop	sigma_add
0.08	0.1	0.2	0.05	1

(these are variances)



Pop Quiz

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

[*]According to FIM
$t_{\max} \approx 6$ hours

Pop Quiz

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.

PopED: 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()
```

PopED: Evaluate FIM

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

```
. [1] 0.04804071
```

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

```
.           CL             V            KA          d_CL           d_V          d_KA   SIGMA[1,1]    S
. 2.983332e+05 3.132099e+06 4.936376e+06 5.192188e+01 4.888612e+02 6.485818e+02 2.997297e+01 4.0
```

PopED: Tweak timepoint and evaluate FIM

```
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)
```

```
.          CL           V          KA        d_CL         d_V        d_KA SIGMA[1,1] SIGMA[2,2]
.    15.48778   142.08530   223.35925    50.88485   418.53695   549.75962    29.68444    40.82396
```

PopED: *D*-optimal design: Starting from the original design

```
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: 0.3275
. Group 1: Model 2:     23     24     24     96
.
. OFV = 20.2291
.
. Efficiency:
.   ((exp(ofv_final) / exp(ofv_init))^(1/n_parameters)) = 18
.
. Expected relative standard error
. (%RSE, rounded to nearest integer):
.     Parameter   Values     RSE_0   RSE
.            CL       10    298333    38
.             V      100   3132099   149
.            KA     0.25   4936376   153
.          d_CL     0.08        52    50
.           d_V      0.1       489   204
.          d_KA      0.2       649   127
.    SIGMA[1,1]     0.05        30    30
.    SIGMA[2,2]        1        41    41
.
. Total running time: 18.044 seconds
```

PopED: Add sample after final (SS) dose

```
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))
)
```

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

```
.          CL           V          KA        d_CL         d_V        d_KA SIGMA[1,1] SIGMA[2,2]
.    12.31062    86.75053   136.64068    50.42738   317.03881   419.93344    29.55819    28.95827
```

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

```
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: 0.4455
. Group 1: Model 2:     23     23     23  41.09    168
.
. OFV = 27.717
.
. Efficiency:
.   ((exp(ofv_final) / exp(ofv_init))^(1/n_parameters)) = 2.
.
. Expected relative standard error
. (%RSE, rounded to nearest integer):
.     Parameter   Values   RSE_0   RSE
.            CL       10      12     9
.             V      100      87    14
.            KA     0.25     137    17
.          d_CL     0.08      50    48
.           d_V      0.1     317    74
.          d_KA      0.2     420    63
.    SIGMA[1,1]     0.05      30    29
.    SIGMA[2,2]        1      29    39
.
. Total running time: 21.498 seconds
```

PopED: Near-optimal design

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

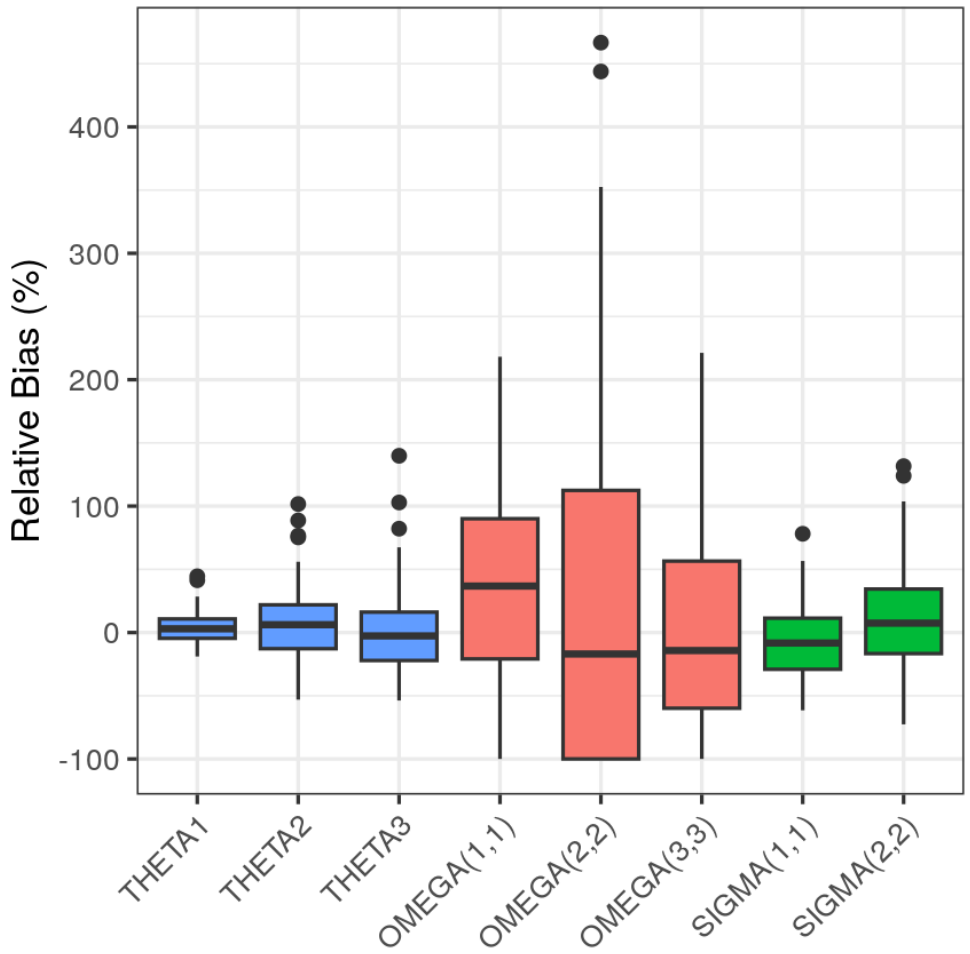
```
FIM_practical <- evaluate.fim(poped_db_practical)
get_rse(FIM_practical, poped_db_practical)
```

```
.          CL           V          KA        d_CL         d_V        d_KA SIGMA[1,1] SIGMA[2,2]
.    10.15307    19.61044    22.79548    48.67831    97.63716    74.56113    26.52843    40.69749
```

PopED: 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

The PFIM setup

PFIM requires types 3 of objects in order to define a StatisticalModel:

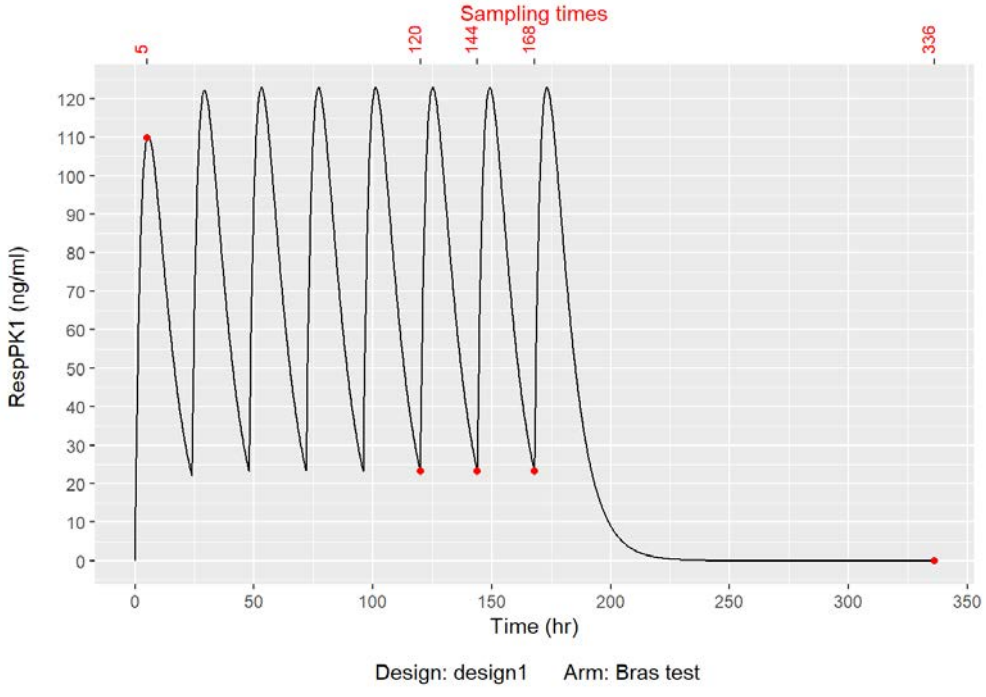
- ModelEquations, the structural model;
- ModelParameter, the parameter models (including IIV and IOV);
- Response, including the residual error model.

A PFIMProject object collects together these model objects with everything related to study design:

- Arm objects, which include SamplingTimes and Administration objects.

PFIM: Plot of initial design

```
evaluationPop <- EvaluatePopulationFIM(MyPro
plotOptions <- list(
  unitTime = c("hr"),
  unitResponses = c("ng/ml")
)

plotResponse <- plotResponse(
  evaluationPop,
  plotOptions
)

print(plotResponse[[1]])
```



PFIM: Evaluate FIM

```
evaluationPop <- EvaluatePopulationFIM(MyProject)
show(evaluationPop)
```

```
**************************  design1  ******************************

   Arm_name Response tau Tinf                          Time_dose
1 Bras test  RespPK1   -    - 0, 24, 48, 72, 96, 120, 144, 168 10000, 10000, 10000, 10000, 10000
```

PFIM: Evaluate FIM

```
evaluationPop <- EvaluatePopulationFIM(MyProject)
show(evaluationPop)
```

```
************************
Fisher information matrix
************************

*** Fixed effect

              μ_Cl         μ_ka           μ_V
μ_Cl  0.594311668  -0.8211656 -0.002425272
μ_ka -0.821165609 564.9226170 -2.218057475
μ_V  -0.002425272  -2.2180575  0.008762577
```

PFIM: Evaluate FIM

```
evaluationPop <- EvaluatePopulationFIM(MyProject)
show(evaluationPop)
```

```
************************
Fisher information matrix
************************

*** Variance components
```

	ω²_Cl	ω²_ka	ω²_V	σ_inter_RespPK1	σ_slope_RespPK1
ω²_Cl	147.1693161	0.1756023	0.2450810	0.1067300	7.591955
ω²_ka	0.1756023	51.9429628	128.1192438	0.2065110	21.923776
ω²_V	0.2450810	128.1192438	319.9281155	0.4876588	51.107443
σ_inter_RespPK1	0.1067300	0.2065110	0.4876588	34.2620161	238.748328
σ_slope_RespPK1	7.5919547	21.9237756	51.1074432	238.7483285	5561.894327

PFIM: Evaluate FIM

```
evaluationPop <- EvaluatePopulationFIM(MyProject)
show(evaluationPop)
```

```
******************
Correlation matrix
******************

*** Fixed effect

     μ_Cl μ_ka μ_V
μ_Cl    1    1   1
μ_ka    1    1   1
μ_V     1    1   1
```

PFIM: Evaluate FIM

```
evaluationPop <- EvaluatePopulationFIM(MyProject)
show(evaluationPop)
```

```
*****************
Correlation matrix
*****************

*** Variance components

                        ω²_Cl        ω²_ka         ω²_V σ_inter_RespPK1 σ_slope_RespPK1
ω²_Cl             1.000000000 -0.007756317  0.007627033     0.003569392    -0.008783515
ω²_ka            -0.007756317  1.000000000 -0.993851261     0.013130503    -0.027650113
ω²_V              0.007627033 -0.993851261  1.000000000    -0.010893393     0.022752548
σ_inter_RespPK1   0.003569392  0.013130503 -0.010893393     1.000000000    -0.547261481
σ_slope_RespPK1  -0.008783515 -0.027650113  0.022752548    -0.547261481     1.000000000
```

PFIM: Evaluate FIM

```
evaluationPop <- EvaluatePopulationFIM(MyProject)
show(evaluationPop)
```

```
*******************************************
Determinant, condition numbers and D-criterion
*******************************************

                                      [,1]
Determinant                   4.394250e+00
Cond number fixed effects     1.718267e+14
cond number variance components 1.017806e+04
D-criterion                   1.203263e+00
```

PFIM: Evaluate FIM

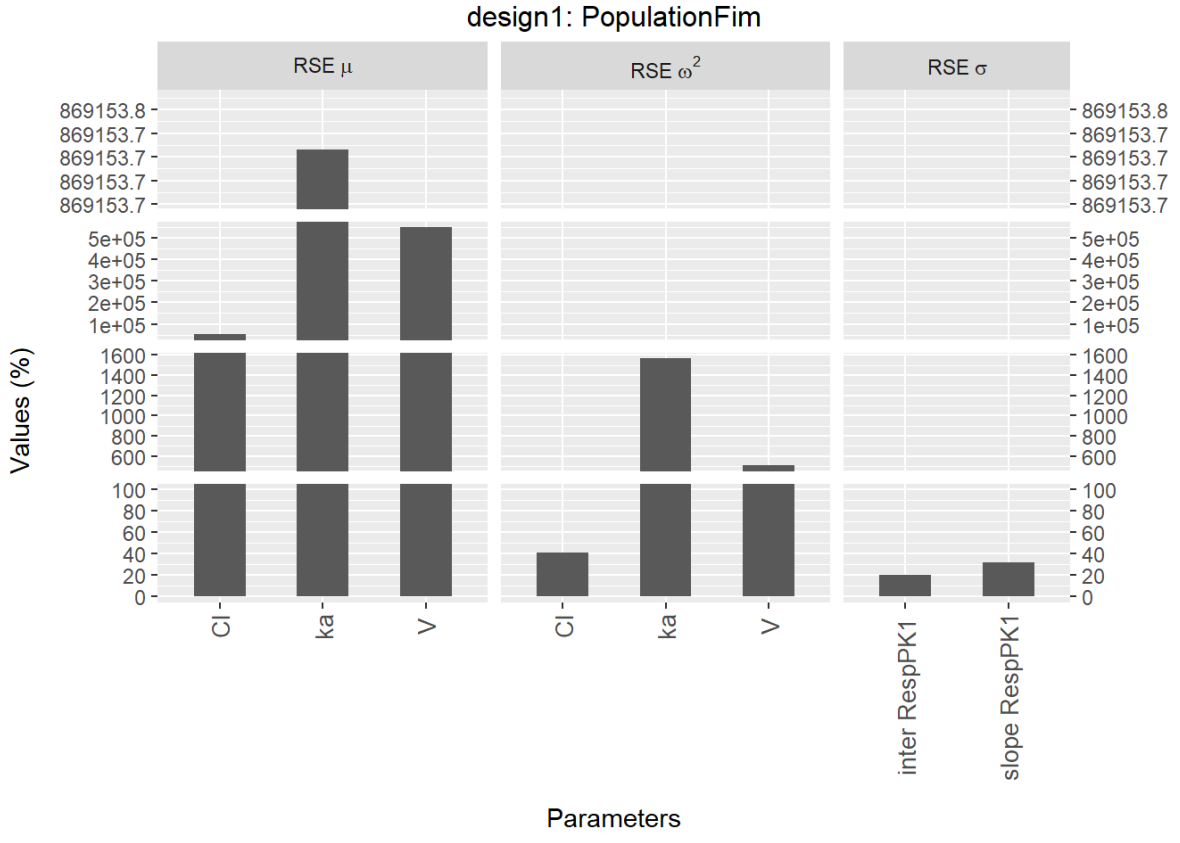
```
evaluationPop <- EvaluatePopulationFIM(MyProject)
show(evaluationPop)
```

```
*********
SE and RSE
*********
```

	Value	SE	RSE (%)
μ_Cl	10.00	5.252748e+03	52527.47704
μ_ka	0.25	2.172884e+03	869153.72640
μ_V	100.00	5.514727e+05	551472.68815
ω²_Cl	0.20	8.243712e-02	41.21856
ω²_ka	0.08	1.254450e+00	1568.06291
ω²_V	0.10	5.054008e-01	505.40077
σ_inter_RespPK1	1.00	2.041242e-01	20.41242
σ_slope_RespPK1	0.05	1.603788e-02	32.07577

PFIM: Plot RSE

```
plotRSE <- plotRSE(evaluationPop)
print(plotRSE[[1]])
```



PFIM: *D*-optimal design: Starting from the original design

```
brasTestDesign1 <- addSampling(brasTestDesign1, SamplingTimes(
  outcome = "RespPK1",
  sample_time = c(5, c(120, 144, 168), 336)
))

samplingBoundsConstraint <- SamplingConstraint(
  response = "RespPK",
  continuousSamplingTimes = list(
    c(0, 6),
    c(119, 120),
    c(143, 144),
    c(167, 168),
    c(264, 336)
  )
)
```

PFIM: *D*-optimal design: Starting from the original design

```
simplexOptimizer <- SimplexAlgorithm(
  pct_initial_simplex_building = 20,
  max_iteration = 5000,
  tolerance = 1e-6
)

optimization_populationFIM <- OptimizeDesign(
  MyProject,
  simplexOptimizer,
  PopulationFim()
)

show(optimization_populationFIM)
```

```
*****************************
 Optimal designs
*****************************

4.98139743936754, 119.999815287607, 143.000000389
167.999998299556, 271.44467773064

*********
SE and RSE
*********
```

	Value	SE	RSE (%)
μ_Cl	10.00	1.38874526	13.88745
μ_ka	0.25	0.22079467	88.31787
μ_V	100.00	56.80758006	56.80758
ω²_Cl	0.20	0.08241346	41.20673
ω²_ka	0.08	0.48045173	600.56467
ω²_V	0.10	0.19874160	198.74160
σ_inter_RespPK1	1.00	0.20408418	20.40842
σ_slope_RespPK1	0.05	0.01571215	31.42429

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

```
brasTestDesign1 <- addSampling(
  brasTestDesign1,
  SamplingTimes(
    outcome = "RespPK1",
    sample_time = c(5, c(120, 144, 168), 240, 336
  )
)
samplingBoundsConstraint <- SamplingConstraint(
  response = "RespPK",
  continuousSamplingTimes = list(
    c(0, 6),
    c(119, 120),
    c(143, 144),
    c(167, 168),
    c(168, 336),
    c(336, 336)
  )
)
```

```
*****************************
 Optimal designs
*****************************

4.52576291331019, 119.000000458829, 143.000791500
169.416690394867, 205.731885951118, 331.266297788

*********
SE and RSE
*********

                    Value          SE  RSE (%)
μ_Cl                10.00  1.29952262 12.99523
μ_ka                 0.25  0.02873640 11.49456
μ_V                100.00 10.96078787 10.96079
ω²_Cl                0.20  0.08235915 41.17958
ω²_ka                0.08  0.05679475 70.99344
ω²_V                 0.10  0.05102656 51.02656
σ_inter_RespPK1      1.00  0.18241440 18.24144
σ_slope_RespPK1      0.05  0.01236994 24.73988
```

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`

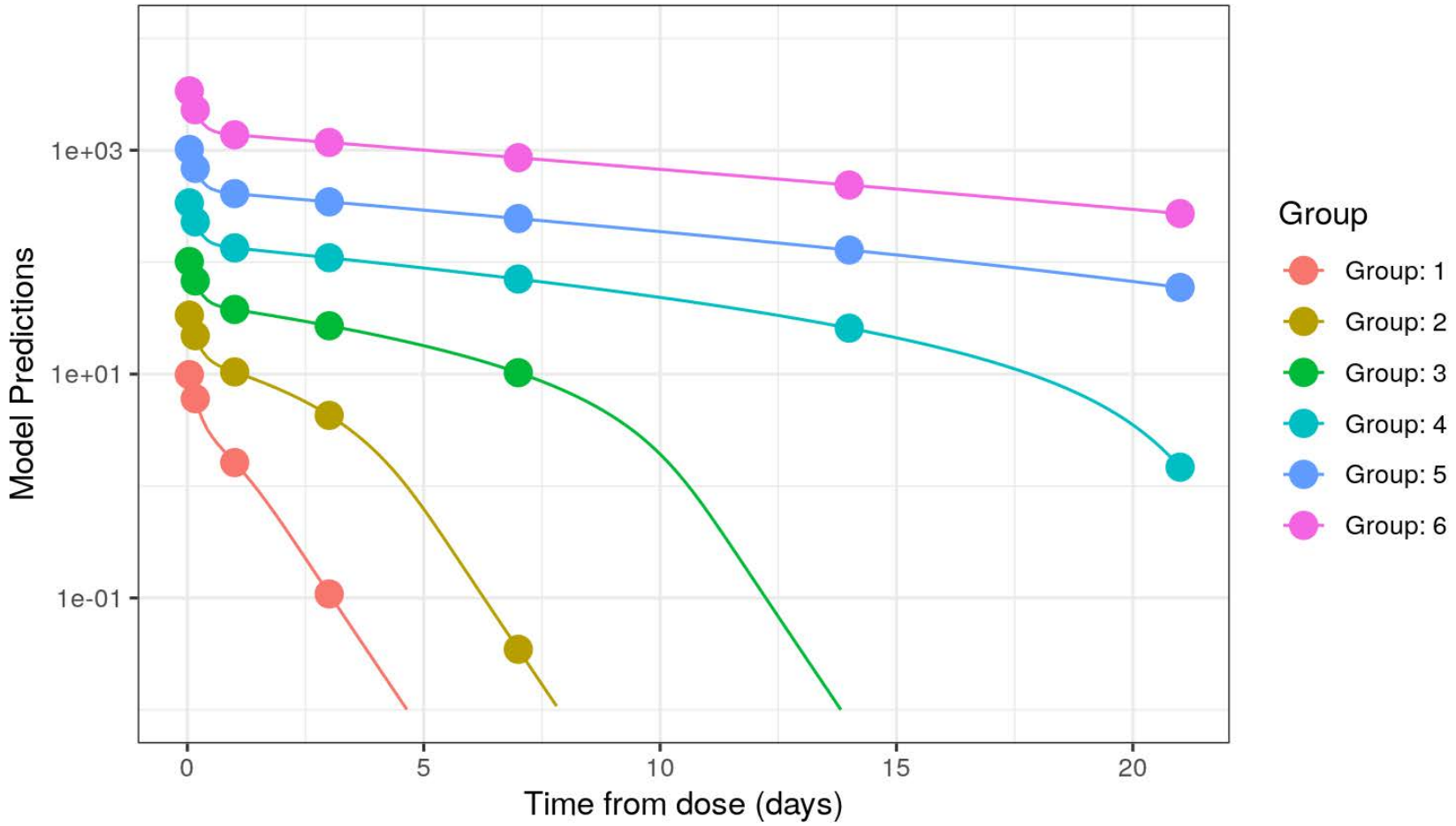
```
.
. Model file:  model_poped.mod
. [ param ]
. CL = 1, VMAX = 10, KM = 10, V1 = 8, Q = 10, V2 = 100
.
. [ cmt ] CENT PERIPH
.
. [ main ]
. double ke  = CL/V1;
. double k12 = Q/V1;
. double k21 = Q/V2;
.
. [ ode ]
. double CP = CENT/V1;
.
. dxdt_CENT = k21*PERIPH - k12*CENT - VMAX*CP/(KM + CP) - ke*CENT;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

PopED: 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

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

PopED: Plot of initial design



PopED: Evaluate FIM

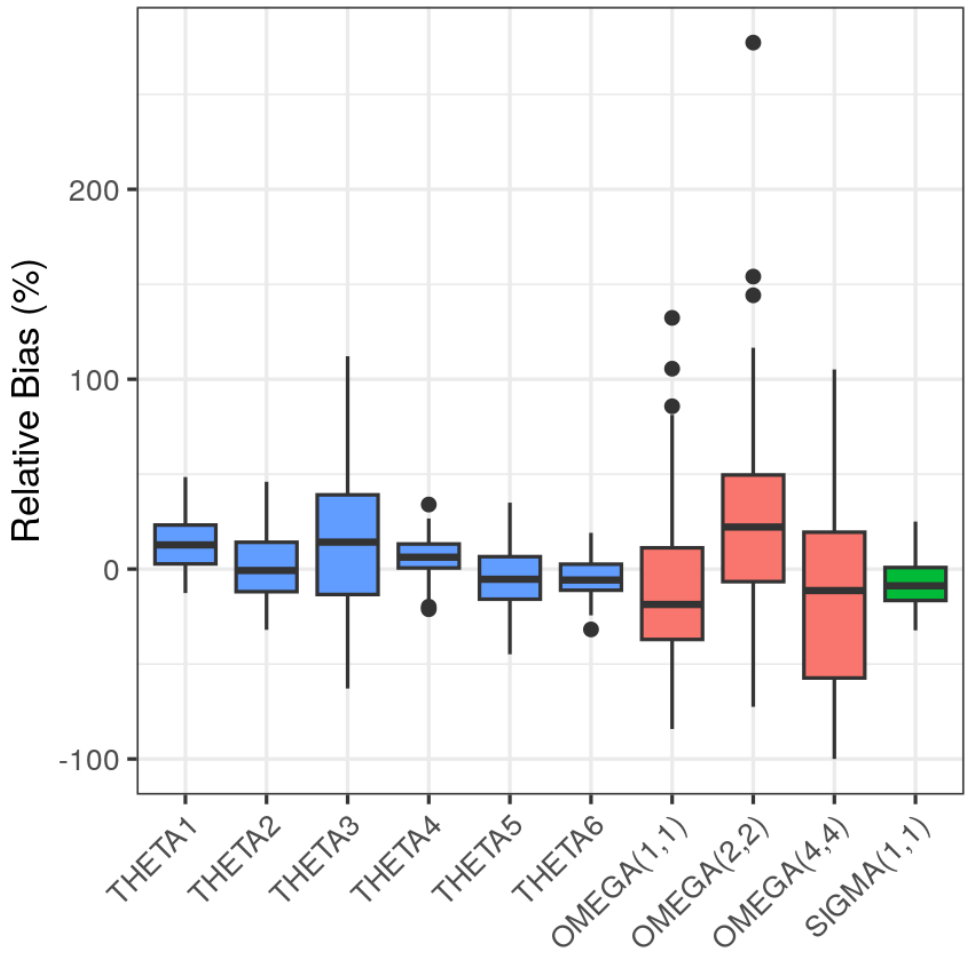
```
FIM_mrg <- evaluate.fim(poped_db_mrg)
get_rse(FIM_mrg, poped_db_mrg)
```

```
.          CL       VMAX          KM          V1           Q          V2       d_CL     d_VMAX          
.  10.874298  10.660146    7.935583    6.355286    8.894680    3.620177  38.989274  32.096484  31.66
```

PopED: Sampling windows

```
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

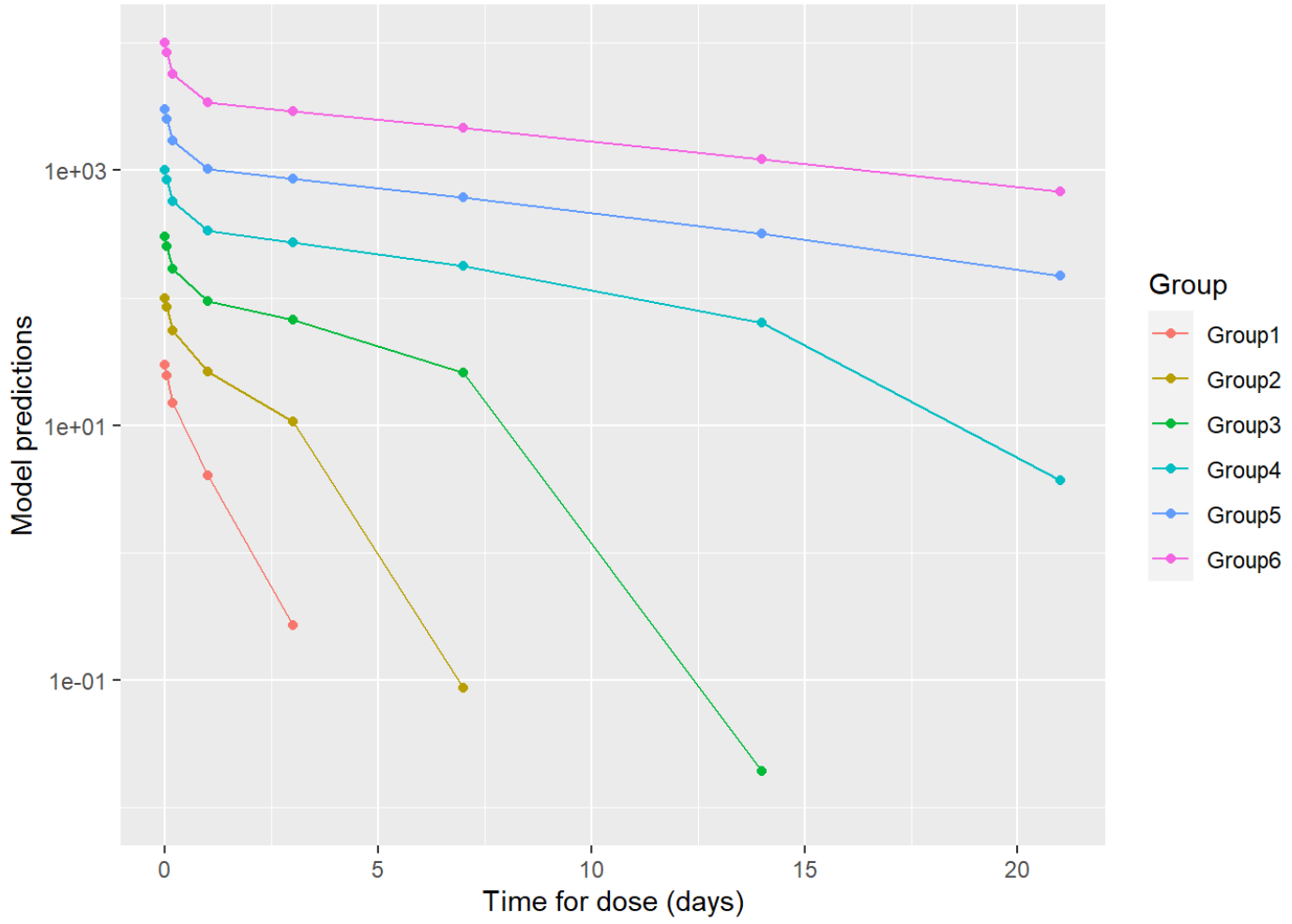


param	poped_pct_rse	sim_pct_rse	sim_pct_bias
THETA1	10.9	13.9	13.6
THETA2	10.7	17.1	1.8
THETA3	7.9	38.4	15.5
THETA4	6.4	10.4	6.6
THETA5	8.9	16.9	-5.8
THETA6	3.6	9.6	-4.4
OMEGA(1,1)	39.0	38.9	-11.9
OMEGA(2,2)	32.1	51.1	24.6
OMEGA(4,4)	31.7	52.3	-12.6
SIGMA(1,1)	10.8	12.3	-7.0

PFIM: ODE model

```
MyStatisticalModel <- StatisticalModel()
MyStatisticalModel <- setParametersOdeSolver(MyStatisticalModel, list(atol = 1e-16, rtol = 1e-6

MyModelEquations <- ModelODEquations(
  list(
    "RespPK1" = expression(C1),
    "RespPK2" = expression(C2)
  ),
  list(
    "Deriv_C1" = expression((Q / V2) * C2 - (Q / V1) * C1 - VMAX * (C1 / V1) /
                              (KM + (C1 / V1)) - (CL / V1) * C1),
    "Deriv_C2" = expression((Q / V1) * C1 - (Q / V2) * C2)
  )
)

MyStatisticalModel <- defineModelEquations(MyStatisticalModel, MyModelEquations)

vC1 <- ModelVariable("C1/V1")
vC2 <- ModelVariable("C2")

MyStatisticalModel <- defineVariable(MyStatisticalModel, vC1)
MyStatisticalModel <- defineVariable(MyStatisticalModel, vC2)
```

PFIM: Plot of initial design



PFIM: Evaluate FIM

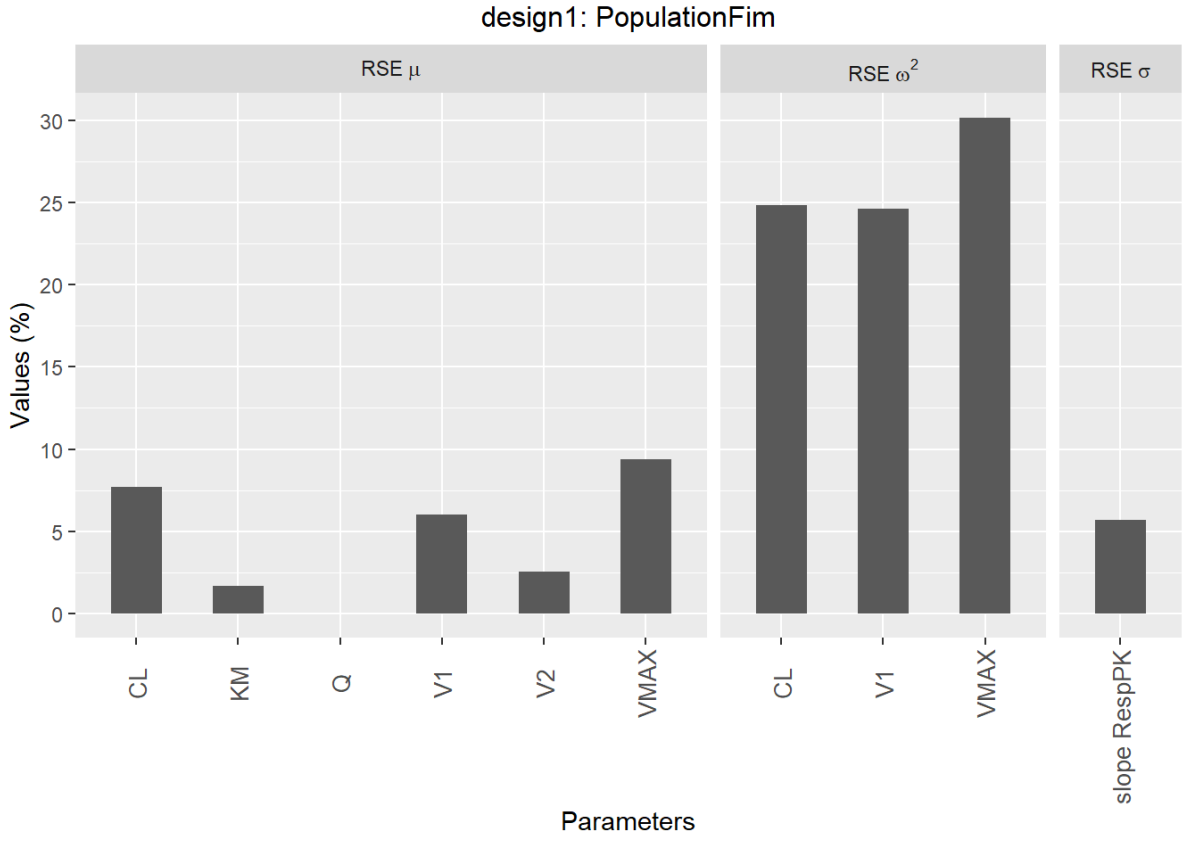
```
evaluationPop <- EvaluatePopulationFIM(MyProject)
show(evaluationPop)
```

```
*********
SE and RSE
*********
```

	Value	SE	RSE (%)
μ_CL	0.50	0.0384775981	7.695519616
μ_KM	1.20	0.0196656516	1.638804302
μ_Q	10.00	0.0005779689	0.005779689
μ_V1	2.50	0.1498846589	5.995386356
μ_V2	4.00	0.1009500500	2.523751249
μ_VMAX	20.00	1.8703324957	9.351662478
ω²_CL	0.20	0.0497454303	24.872715150
ω²_V1	0.10	0.0246396407	24.639640672
ω²_VMAX	0.20	0.0603291975	30.164598760
σ_slope_RespPK	0.15	0.0085116372	5.674424799

PFIM: Plot RSE

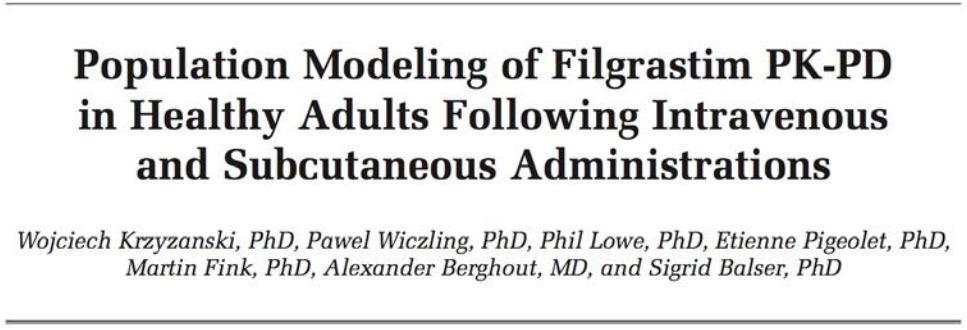
```
plotRSE <- plotRSE(evaluationPop)
print(plotRSE[[1]])
```



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.



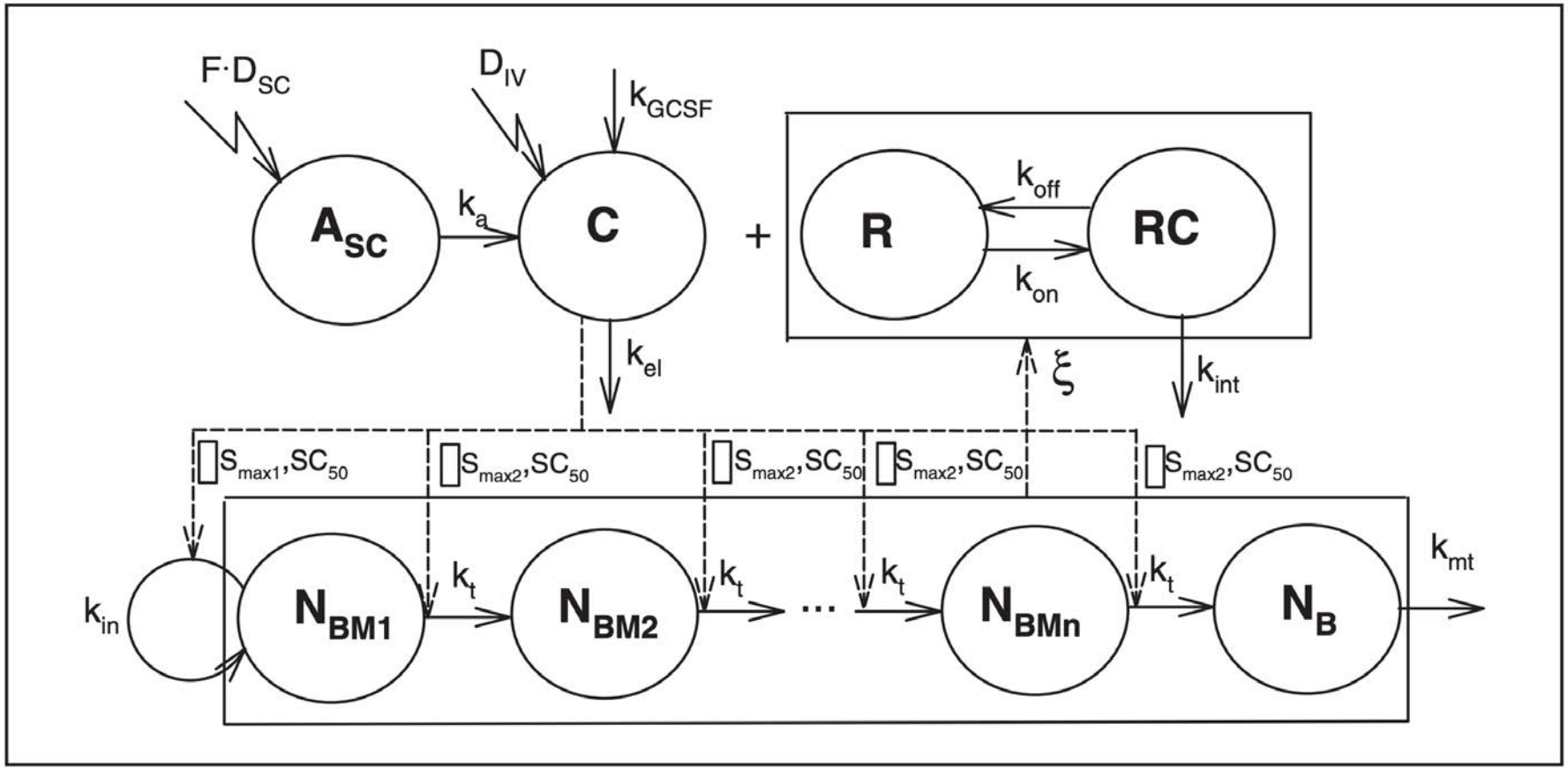
Population Modeling of Filgrastim PK-PD in Healthy Adults Following Intravenous and Subcutaneous Administrations

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

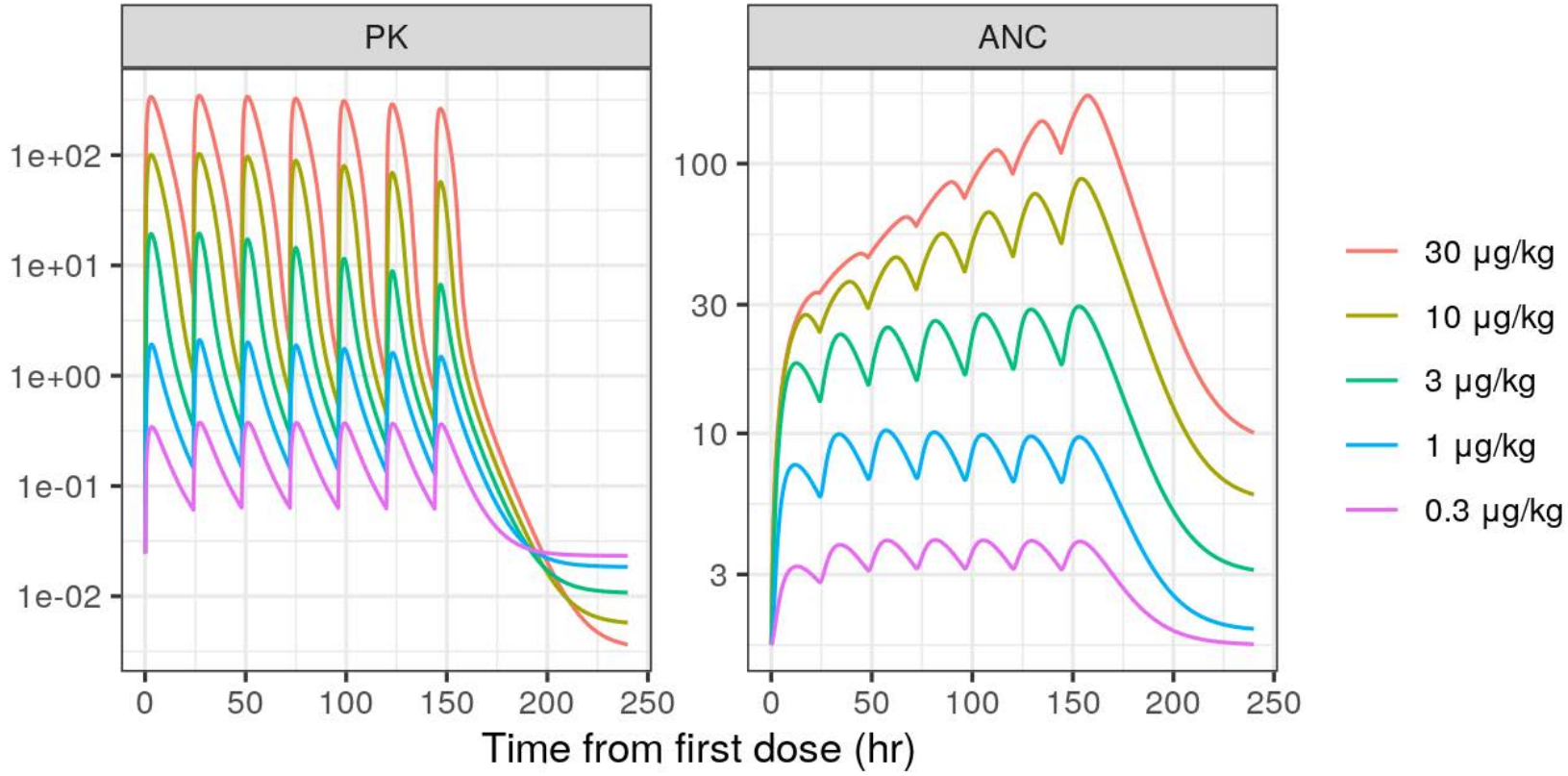
Krzyzanski, Wiczling, Lowe, Pigeolet, Fink, Berghout, and Balser (2010)

Filgrastim ANC model



Krzyzanski, Wiczling, Lowe et al. (2010)

PK and ANC simulations



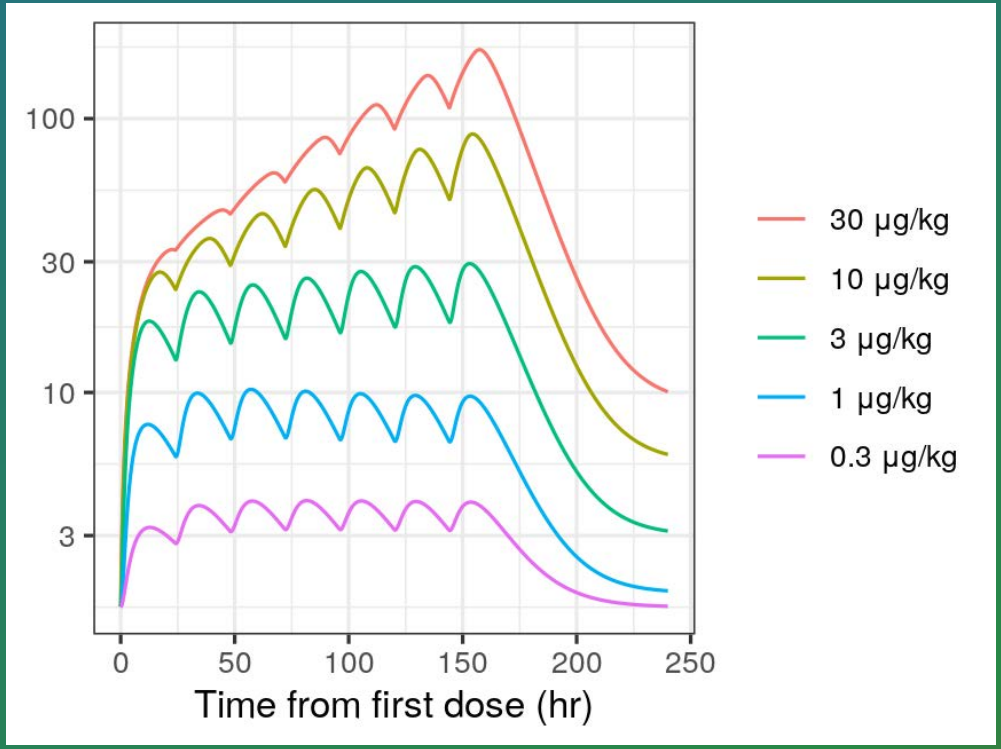
Evaluate FIM for initial design

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

```
.          KA         KEL          VD          KD        KINT         KSI         KMT         KTT
.    9.332963   13.760898   12.690538    8.603883    3.911012   20.404806    3.418712   17.263477    4.00
.        d_SC1       d_SM1 SIGMA[1,1] SIGMA[3,3] SIGMA[4,4]
.   27.653544   26.362398    3.926952    6.505779    7.638103
```

Pop Quiz

Any better doses for estimating SC50 or Smax?



1. 1, 3, 10 µg/kg
2. 0.3, 1, 3 µg/kg
3. 3, 10, 30 µg/kg

Pop Quiz

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

Parameter	Design 1	Design 2	Design 3
SC1	17.8	19.7	18.1
SM1	7.2	10.2	6.2
SM2	6.4	9.3	5.4

Wrap up

What did we learn today?

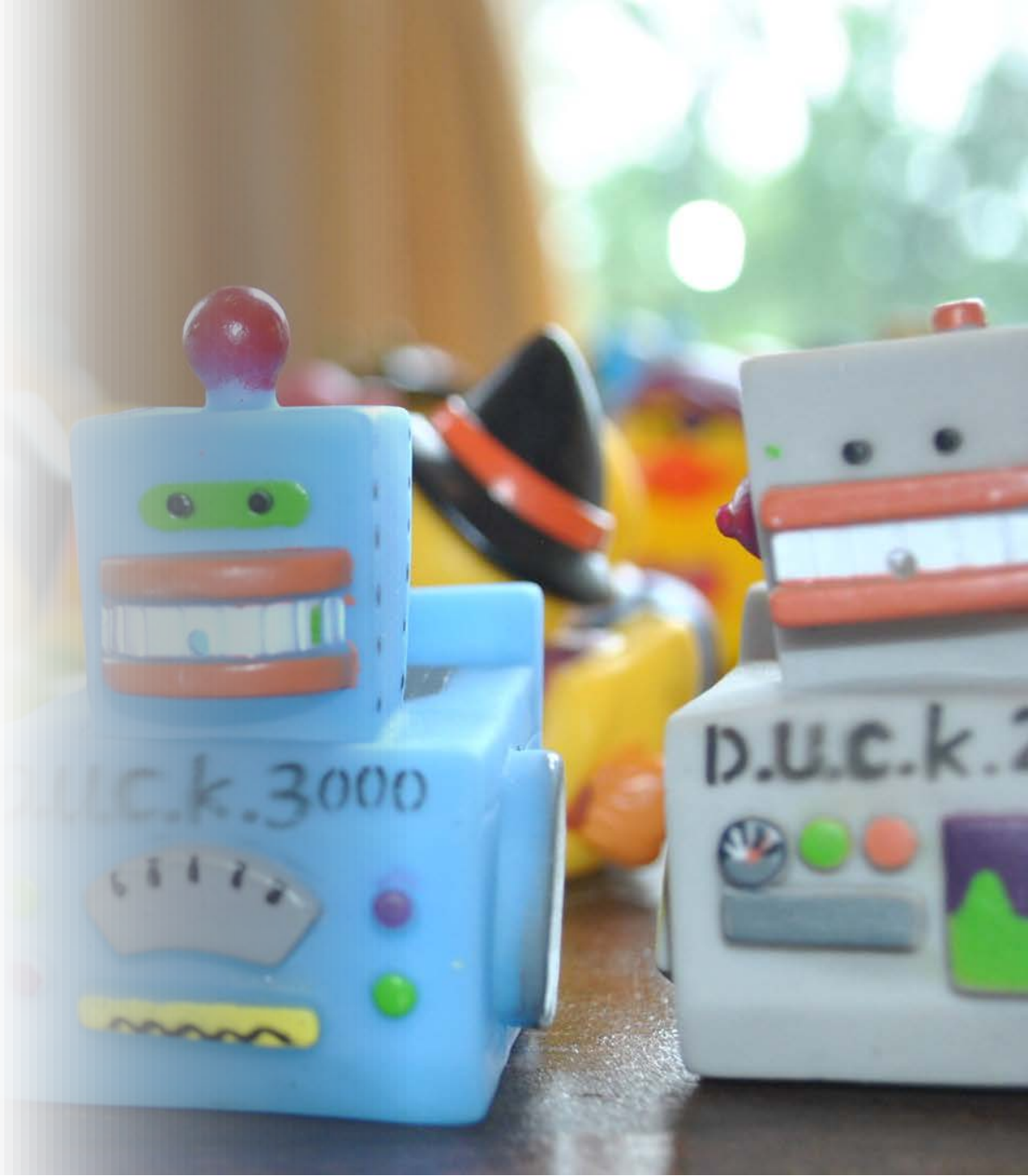
- Optimal design can be a useful, if imperfect, tool to explore and optimize study design options
- Always run confirmatory simulations
- Optimal design: it's not just for PK sampling any more!

What did we miss?

- Dealing with parameter uncertainty (e.g., ED-, $HC\ln D$-optimality)
- Dealing with model structure 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 options available in `PopED` and `PFIM`

Software resources

- `PopED`: https://andrewhooker.github.io/PopED/
- `PFIM`: http://www.pfim.biostat.fr/
- `mrgsolve`: https://mrgsolve.github.io/
- `bbi`: https://github.com/metrumresearchgroup/bbi
    - `bbr`: https://metrumresearchgroup.github.io/bbr/



References

Bauer, R. J., A. C. Hooker, and F. Mentre (2021). "Tutorial for \$ DESIGN in NONMEM: Clinical trial evaluation and optimization". In: *CPT: Pharmacometrics & Systems Pharmacology* 10.12, pp. 1452-1465. DOI: https://doi.org/10.1002/psp4.12713. eprint: https://ascpt.onlinelibrary.wiley.com/doi/pdf/10.1002/psp4.12713. URL: https://ascpt.onlinelibrary.wiley.com/doi/abs/10.1002/psp4.12713.

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.

Leroux, R., J. Seurat, H. Nagard, et al. (2022). *Design evaluation and optimisation in nonlinear mixed effects models with the R package PFIM*. Poster. Abstr 10183.

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

timw@metrumrg.com

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 bbr

```
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 `bbr`

```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

```
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

```
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, p
  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()

```
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()

```
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,
  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()

```
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,
  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 (`THETA`s)
- `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()

```
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,
  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()

```
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,
  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,
  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