Design evaluation and optimisation in nonlinear mixed effects models with the R package PFIM 6.0

Romain Leroux, Jérémy Seurat, Lucie Fayette, Nguyen Tran Bach,
France Mentré on behalf of the PFIM group

IAME, UMR 1137, INSERM, Université Paris Cité, Paris, France

Background and objectives

- Nonlinear mixed-effects models (NLMEMs) are widely used in model-based drug development to analyze longitudinal data. For optimizing the design of longitudinal studies in pharmacometrics, the use of the Fisher information matrix (FIM) is a good alternative to time-consuming clinical trial simulations
- PFIM 4.0 was released in 2014 [1] and is one of the tools developed for FIM-based evaluation and/or optimization of designs in NLMEMs. The next version PFIM 5.0 was an R package releasead on the CRAN.
- The R package PFIM 6.0 is the new version of PFIM, that provides powerful tools to perform FIM evaluation and design optimization under given design constraints in NLMEMs
- **Save the date: PFIM 6.0 will be submitted on the Comprehensive R Archive the 31/07/2023**

https://cran.r-project.org/web/packages/PFIM/index.html

The R package PFIM 6.0

Methods

- The package PFIM 6.0 was entirely rewritten, keeping the formal object oriented system S4 [2]
- The object-oriented system S4 defines objects having clear object-oriented programming characteristics including class and argument definitions, inheritance, as well as argument checking, instantiation and implementation mThe user can write his entire project with the easiest possible R scripts

Notable features

- Individual, Population and Bayesian Fisher Information Matrix
- User-defined models (analytic and ODE) for one or several responses
- Library of PK, PD and PKPD models: the PK library includes different administration (bolus, infusion and oral - first order absorption), linear and Michaelis-Menten elimination, 1- and 2-compartment models. The PD library contains direct and indirect, linear and nonlinear models. The PK/PD models are obtained by combining the models from the PK and PD libraries. PFIM 5.0 also offers the possibility to extend models from the library
- Parameters with different distributions (normal and lognormal)

New features in PFIM 5.0 and PFIM 6.0

- Eased definition of multiple administrations with different routes (oral, IV bolus, IV infusion) and at different doses
- PKPD and PKPKPD models in analytic and ODE form
- A data summary for design evaluation and optimization. The standard data visualization package ggplot2 is used to display the results in clear graphical form (sensitivity graphs, responses over time and optimal design, SE and RSE obtained with the evaluated or the optimised design) The package rmarkdown is used to turn all the results into high quality reports that can be easily shared
- Possibility to get and use the FIM based results after design evaluation or optimization
- Optimization algorithms: the simplex algorithm (Nelder-Mead) [3], PGBO (Population Genetic Based Optimization) [4], PSO (Particle Swarm Optimization) [5], Fedorov-Wynn [6], Multiplicative algorithm [7,8]
- Possibility to optimize both the doses and the measurement times (e.g. by using the Multiplicative algorithm)
- Developer documentation on all the methods and classes implemented
- User-friendly vignettes to demo of the package's capabilities

Main changes versus PFIM 5.0

- Provide the easiest possible scripts to use for design evaluation and optimization
- Facilitate the modularity with the next features
- Improve the code performances for design evaluation and optimization.

Perspectives

- Covariates and Wald test power predictions [9]
- Alternative methods to evaluate the FIM (e.g. MC/AGQ [10]) for discrete response models and robust design optimization accounting for model and/or parameter uncertainty [11,12]
- Increase the interoperability of PFIM and the library of PKPD model with estimation parameter softwares
- A GUI version of the package PFIM 6.0

Example of design evaluation of an ODE PKPD model with infusion

R script

```
# Define the model equations

modelEquations = list(

  outcomes = list( "RespPK" = "Cc",
                   "RespPD" = "E" ),

  equations = list(  "Deriv_Cc" = "dose_RespPK/V*ka*exp(-ka*t) - Cl/V*Cc",
                     "Deriv_E" = "Rin*(1-Imax*(Cc**gamma)/(Cc**gamma + IC50**gamma))-kout*E" ) )

# Define the model parameters

modelParameters = list(
  ModelParameter( name = "V",     distribution = LogNormal( mu = 0.74, omega = 0.316 ) ),
  ModelParameter( name = "Cl",    distribution = LogNormal( mu = 0.28, omega = 0.456 ) ),
  ModelParameter( name = "ka",    distribution = LogNormal( mu = 10, omega = 0.0 ), fixedMu = TRUE ),
  ModelParameter( name = "kout",  distribution = LogNormal( mu = 6.14, omega = 0.947 ) ),
  ModelParameter( name = "Rin",   distribution = LogNormal( mu = 614, omega = 0 ), fixedMu = TRUE ),
  ModelParameter( name = "Imax",  distribution = LogNormal( mu = 0.76, omega = 0.439 ) ),
  ModelParameter( name = "IC50",  distribution = LogNormal( mu = 9.22, omega = 0.452 ) ),
  ModelParameter( name = "gamma",distribution = LogNormal( mu = 2.77, omega = 1.761 ) ) )

# Define the error model

errorModelRespPK = Proportional( outcome = "RespPK", sigmaSlope = 0.21 )
errorModelRespPD  = Constant( outcome = "RespPD", sigmaInter = 9.6 )

modelError = list( errorModelRespPK,  errorModelRespPD )

# Define the administration parameters

arm1AdministrationRespPK = Administration( outcome = "RespPK", timeDose = c( 0 ), dose = c( 0.2 ) )
arm2AdministrationRespPK = Administration( outcome = "RespPK", timeDose = c( 0 ), dose = c( 0.64 ) )
arm3AdministrationRespPK = Administration( outcome = "RespPK", timeDose = c( 0 ), dose = c( 2 ) )
arm4AdministrationRespPK = Administration( outcome = "RespPK", timeDose = c( 0 ), dose = c( 6.24 ) )
arm5AdministrationRespPK = Administration( outcome = "RespPK", timeDose = c( 0 ), dose = c( 11.24 ) )
arm6AdministrationRespPK = Administration( outcome = "RespPK", timeDose = c( 0 ), dose = c( 20 ) )

# Define the sampling times

samplingTimesRespPK = SamplingTimes( outcome = "RespPK",
                                     samplings = c( 0.25, 0.5, 0.75, 1, 1.25, 1.5, 2, 3, 4 ) )

samplingTimesRespPD = SamplingTimes( outcome = "RespPD",
                                     samplings = c( 0.25, 0.5, 0.75, 1, 1.25, 1.5, 2, 3, 4 ) )

# Define  the arms

arm1 = Arm( name = "0.2mg Arm", size = 6,
            administrations = list( arm1AdministrationRespPK ) ,
            samplingTimes = list( samplingTimesRespPK, samplingTimesRespPD ),
            initialCondition = list( "Cc" = 0, "E" = 100 ) )

arm2 = Arm( name = "0.64mg Arm", size = 6,
            administrations = list( arm2AdministrationRespPK ) ,
            samplingTimes = list( samplingTimesRespPK, samplingTimesRespPD ),
            initialCondition = list( "Cc" = 0, "E" = 100 ) )

arm3 = Arm( name = "2mg Arm", size = 6,
            administrations = list( arm3AdministrationRespPK ) ,
            samplingTimes = list( samplingTimesRespPK, samplingTimesRespPD ),
            initialCondition = list( "Cc" = 0, "E" = 100 ) )

arm4 = Arm( name = "6.24mg Arm", size = 6,
            administrations = list( arm4AdministrationRespPK ) ,
            samplingTimes = list( samplingTimesRespPK, samplingTimesRespPD ),
            initialCondition = list( "Cc" = 0, "E" = 100 ) )

arm5 = Arm( name = "11.24mg Arm", size = 6,
            administrations = list( arm5AdministrationRespPK ) ,
            samplingTimes = list( samplingTimesRespPK, samplingTimesRespPD ),
            initialCondition = list( "Cc" = 0, "E" = 100 ) )

arm6 = Arm( name = "20mg Arm", size = 6,
            administrations = list( arm6AdministrationRespPK ) ,
            samplingTimes = list( samplingTimesRespPK, samplingTimesRespPD ),
            initialCondition = list( "Cc" = 0, "E" = 100 ) )

# Set the design

design1 = Design( name = "design1", arms = list( arm1, arm2, arm3, arm4, arm5, arm6 ) )

# Evaluate the population Fim and display the results

evaluationFIM = Evaluation( name = "PKPD_ODE_multi_doses_populationFIM",
                            modelEquations = modelEquations,
                            modelParameters = modelParameters,
                            modelError = modelError,
                            outcomes = list( "RespPK" = "Cc", "RespPD" = "E" ),
                            designs = list( design1 ),
                            fim = "population",
                            odeSolverParameters = list( atol = 1e-8, rtol = 1e-8 ) )

evaluationFIM = run( evaluationFIM )

show( evaluationFIM )

Report( evaluationFIM, outputPath = « … », outputFile = « … » )
```

Description of the study

Model and data

Taken from a PKPD population study on a non-steroidal molecule [13]

Administration

Single doses of 0.2, 0.64, 2, 6.24, 11.24, or 20mg were given respectively to 6 groups, each of which contained 6 rats, following a parallel design

Evaluation

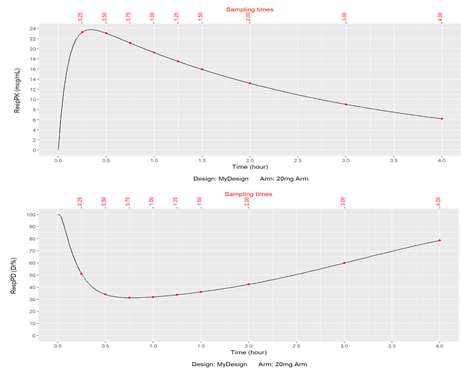
Blood sampling and drug response evaluation were conducted at 15, 30, and 45min and at 1, 1.25, 1.5, 2, 3 and 4 hours after administration

Results for design evaluation

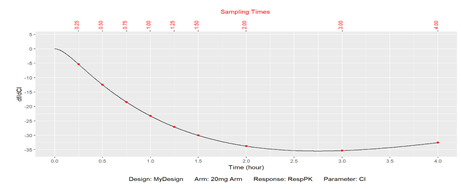
Determinant, condition numbers and D-criterion of the FIM

Determinant	Condition numbers		D-criterion
2.868691e+22	3826.572	17005.66	40.19021

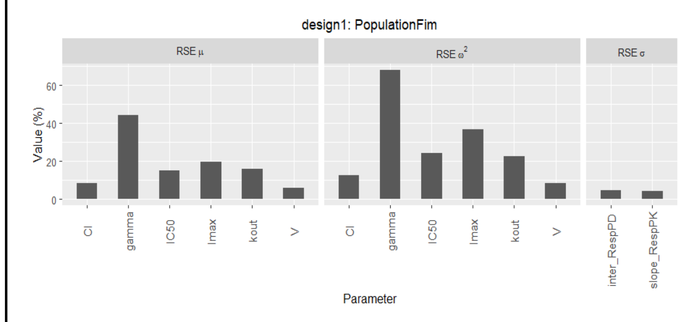
Plots of the evolution of the PK response and PD response for the arm « 20mg Arm »



Plot of the evolution of the sensitivity function with respect to the parameter Cl for the PK response for the arm « 20mg Arm »



Plot of the RSE



Example of design optimization of an ODE PK model with sampling constraints

R script

```
# Define the model equations

modelEquations = list(

  outcomes = list( "RespPK" ),

  equations = list( duringInfusion = list( "RespPK" = "dose_RespPK/Tinf_RespPK/Cl * (1 - exp(-Cl/V * t))" ),
                    afterInfusion  = list( "RespPK" = "dose_RespPK/Tinf_RespPK/Cl * (1 - exp(-Cl/V *
                                                      Tinf_RespPK)) * (exp(-Cl/V * (t - Tinf_RespPK)))") ) )

# Define the model parameters

modelParameters = list(
  ModelParameter( name = "V",  distribution = LogNormal( mu = 50, omega = sqrt( 0.26 ) ) ) ,
  ModelParameter( name = "Cl", distribution = LogNormal( mu = 5,  omega = sqrt( 0.34 ) ) ) )

# Define the error model

errorModelRespPK = Combined1( outcome = "RespPK", sigmaInter = 0.5, sigmaSlope = sqrt( 0.15 ) )
modelError = list( errorModelRespPK )

# Define the administration

administrationRespPK = Administration( outcome = "RespPK", Tinf = c(1, 1, 1, 1, 1),
                                                           timeDose = c( 0, 24, 48, 72, 96 ),
                                                           dose = c( 400, 200, 200, 200, 200 ) )
# Define the sampling times

samplingTimesRespPK = SamplingTimes( outcome = "RespPK", samplings = c( 1, 20, 44, 120 ) )

# Define the administration and sampling constraints

administrationConstraintsRespPK = AdministrationConstraints( outcome = "RespPK", doses = c( 100 ) )

samplingConstraintsRespPK  = SamplingTimeConstraints( outcome = "RespPK",
                                                      samplings = c(1, 20, 44, 120 ),
                                                      samplingsWindows = list( c( 1, 48),c( 72,120 ) ),
                                                      numberOfTimesByWindows = c(2,2),
                                                      minSampling = 5 )
# Define the arm and the design

arm1 = Arm( name = "BrasTest1",
            size = 150,
            administrations = list( administrationRespPK ),
            samplingTimes   = list( samplingTimesRespPK ),
            administrationsConstraints = list( administrationConstraintsRespPK ),
            samplingTimesConstraints = list( samplingConstraintsRespPK ) )

design1 = Design( name = "design1", arms = list( arm1 ), numberOfArms = 150 )

# Define and run the design optimization and display the results

optimization = Optimization( name = "PK_ODE_multi_doses_populationFIM",
                             modelEquations = modelEquations,
                             modelParameters = modelParameters,
                             modelError = modelError,

                             optimizer = "PSOAlgorithm",
                             optimizerParameters = list( maxIteration = 100, populationSize = 20,
                                                         personalLearningCoefficient = 2.05,
                                                         globalLearningCoefficient = 2.05,
                                                         showProcess = TRUE  ),

                             designs = list( design1 ),
                             fim = "population",
                             outcomes = list( "RespPK" ) )

optimizationPSO = run( optimization )

show( optimizationPSO )

Report(optimizationPSO, outputPath = « … », outputFile = « … » )
```

Description of the study

Model and data

Taken from a PK population study of the active metabolite of Remdesivir [14]

Administration

One hundred and fifty (150) subjects receive a 400mg loading dose on the first day, followed by 4 daily doses of 200mg. Blood samples are taken at the end of the 1st infusion (H1), H20, H44 and H120

Optimization

4 sampling times on intervals [1,48] and [72,120] using PSO (Particle Swarm Optimization) algorithm with a minimal sampling interval of 5h.

Results for design optimization

Initial design

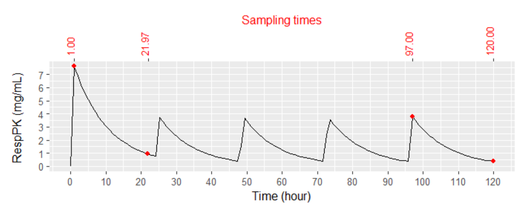
Determinant, condition numbers and D-criterion of the FIM

Determinant	Condition numbers		D-criterion
38108963686	101.5327	21.28127	58.01022

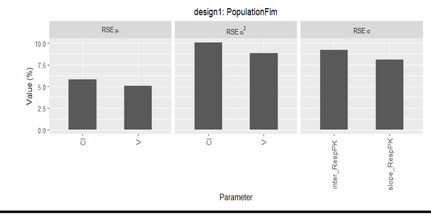
Optimal design

Determinant, condition numbers and D-criterion of the FIM

Determinant	Condition numbers		D-criterion
214400764558	77.13688	7.193726	77.36379



Plot of the RSE



References

[1] Dumont C, Lestini G, Le Nagard H, Mentré F, Comets E, Nguyen TT, et al. PFIM 4.0, an extended R program for design evaluation and optimization in nonlinear mixed-effect models. Comput Methods Programs Biomed. 2018;156:217–29.
[2] Chambers JM. Object-Oriented Programming, Functional Programming and R. Stat Sci. 2014;29:167–80.
[3] Nelder JA, Mead R. "A simplex method for function minimization." Comput J. 1965;7:308–13.
[4] Le Nagard H, Chao L, Tenaillon O. The emergence of complexity and restricted pleiotropy in adapting networks. BMC Evol Biol 2011;11:326.
[5] Eberhart RC, Kennedy J. A new optimizer using particle swarm theory. Proc. of the Sixth International Symposium on Micro Machine and Human Science, Nagoya, 4-6 October 1995;39-43.
[6] Fedorov, VV. Theory of Optimal Experiments. Academic Press, New York, 1972.
[7] Yu Y. Monotonic convergence of a general algorithm for computing optimal designs. Ann Stat. 2010;38:1593–606.
[8] Seurat J, Tang Y, Mentré F, Nguyen, TT. Finding optimal design in nonlinear mixed effect models using multiplicative algorithms. Computer Methods and Programs in Biomedicine. 2021;207.106126.
[9] Retout S, Comets E, Samson A, Mentré F. Design in nonlinear mixed effects models: optimization using the Fedorov-Wynn algorithm and power of the Wald test for binary covariates. Stat Med. 2007;26:5162–79.
[10] Ueckert S, Mentré F. A new method for evaluation of the Fisher information matrix for discrete mixed effect models using Monte Carlo sampling and adaptive Gaussian quadrature. Comput Stat Data Anal. 2017;111:203–19.
[11] Loingeville F, Nguyen TT, Riviere MK, Mentré F. Robust designs in longitudinal studies accounting for parameter and model uncertainties - application to count data. Journal of Biopharmaceutical Statistics. 2020;30(1):31-45.
[12] Seurat J, Nguyen TT, Mentré F. Robust designs accounting for model uncertainty in longitudinal studies with binary outcomes. Statistical Methods in Medical Research. 2020;29(3):934-952.
[13] Flores-Murrieta FJ, Ko HC, Flores-Acevedo DM, López-Muñoz FJ, Jusko WJ, Sale ME, Castañeda-Hernández G. Pharmacokinetic-pharmacodynamic modeling of tolmetin antinociceptive effect in the rat using an indirect response model: a population approach. J Pharmacokinet Biopharm. 1998
[14] Sukeishi, Asami, Kotaro Itohara, Atsushi Yonezawa, Yuki Sato, Katsuyuki Matsumura, Yoshiki Katada, Takayuki Nakagawa, et al. Population Pharmacokinetic Modeling of GS-441524, the Active Metabolite of Remdesivir, in Japanese COVID-19 Patients with Renal Dysfunction. *CPT: Pharmacometrics & Systems Pharmacology* 2022;11(1):94-103.