Text for blogpost

Stuff I missed:

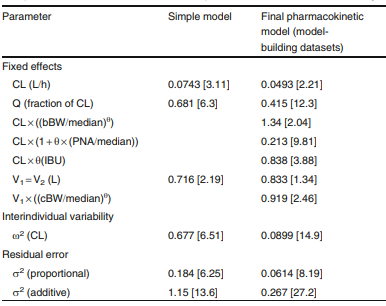
* Headings
* Refer more to what Michiel did in the shiny app

When you are starting to learn about the basics of modeling & simulation, it could be useful to reproduce a model you found in literature and play around with the parameters to get a better understanding about what everything does. Here I’ll give a short example on how to build a population model with the information from a paper. For this, I’ll rely on the previous posts on Part 1 & 2 of “Creating a simple pharmacometric Shiny application with mrgsolve in R” and the results of this paper [include hyperlink].

In addition to what you learned from the previous posts, after following this example, you should be able to *add covariates to a model* and add IV infusion as an administration route.

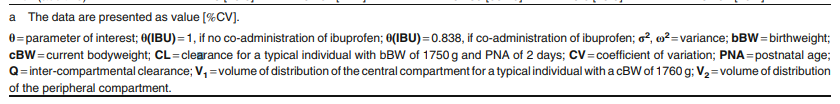
To summarize, I chose a publication of De Cock *et. al.* where they built a population PK model to describe the disposition of amikacin (antibiotic) in preterm neonates following IV infusions. From the results section we learn that a two-compartment model fits the data best. They choose to relate Q to CL and Vp = Vd to increase model stability. PK models developed in a pediatric population often use covariates to explain more of the inter-individual variability. In this publication birthweight (bBW) and postnatal age (PNA) were implemented as covariates on CL following an exponential and a linear relationship, respectively. Some of the pediatric patients receive ibuprofen together with amikacin. Ibuprofen co-administration was found to impact CL so it is included as a covariate on this parameter. Current weight (cBW) was implemented as an exponential covariate on Vd (and Vp since Vp = Vd). All covariate relationships are described in Table II. Below you can find a screenshot of the first 3 columns of this table showing the parameters descriptions and estimates with the simple and final PK models.

[Here is the screen shot]



You probably observed that the individual covariates are related to the median covariate values of the preterm neonates population. These can be found (in this case) in the footnotes of the table.

[Here screenshot of the footnotes]



Now, let’s build the model file, using the example provided in Part 1 of “Creating a simple pharmacometric Shiny application with mrgsolve in R”. First, let’s think of how to adjust the model file code to reproduce the published amikacin model:

* Put in the final parameter estimates in the model file (optional, but I find it nicer in this case because these are drug specific and you don’t want to let the user change them in the app)
* Add initial values for the covariates and the parameters characterizing how they are related to the typical population values (these you should be able to change from the app to enable the user to dose patients with different characteristics)
* Remove the GUT compartment, as this is specific for oral administration, and this drug is given as an IV infusion

With this in mind, the model code changes to this:

amik\_popPK.cpp [maybe highlight changes?]

The typical parameter values should match the ones in the Table II. [….]

I always start with building the model just to make sure that my model works the way I expect it to and matches the results in the publication. Only after I have performed this sanity check I start working on my Shiny app.

In Part 1 of “Creating a simple pharmacometric Shiny application with mrgsolve in R “ we learned how to perform simulations using the mrgsolve package, so I’ll start with that R script and make the necessary changes to integrate the model for amikacin.

We start by providing the covariate values first, as the dose is given in mg/kg, based on current weight.

[code]

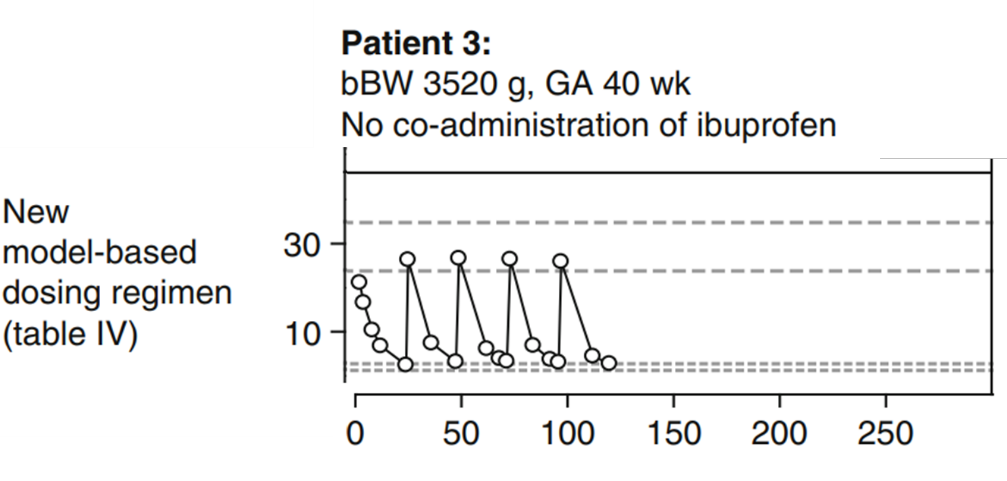
As stated previously we need to add the option to administrate an IV infusion. For this we need the rate of infusion which you can obtain by dividing the dose by the duration of the infusion (RATE = DOSE/DURATION). For amikacin, the duration of the IV infusion is between 20 – 30 min. I selected 20 min and converted that to hours (i.e., 20/60). We should also be able to change the dosing interval from the app, as this should be quite flexible in individualized medicine (see Tables III and IV from the paper to get an idea on how widely this changes to fit the needs of each patient).

[code]

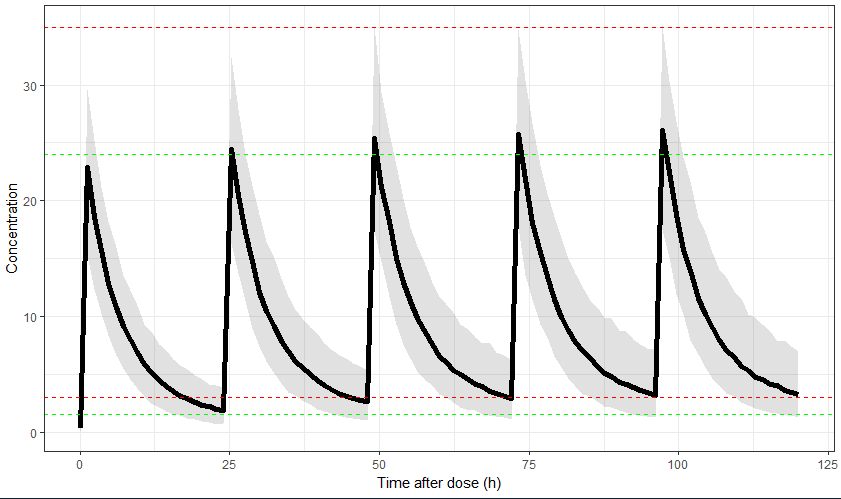
Don’t forget to check the variability matrix. In the paper only one eta was added on CL and for the others they were fixed to 0. In the Shiny app we can give the user the flexibility to change the variability on all the parameters and to get a feeling on how they impacts the simulations.

[code]

We now have a working model and we will use it to perform simulations for our *sanity check*. Let’s reproduce the model-informed dosing recommendation from Table IV for patient 2 displayed in Figure 3 – bottom row. This patient should have a bBW = 3520 g. Since other details on the covariates needed are missing, we’ll assume cBW = 3400 g and a PNA = 5 days. For these characteristics we need to administrate a dose of 12 mg/kg every 24 hours as an IV infusion with the duration of 20-30 min.



[and add here the simulation results]



Since we don’t know the exact patient covariates used for this simulation or the exact infusion duration, we just need to accept that we are close enough.

Now, let’s turn this model into a Shiny application! First we need to think about how would you like to use the app. I’d like to see:

* How different dosing regimen work in patients with different characteristics (covariates)
* Does the patient receive ibuprofen co-medication?
* How variability impacts our simulation results. Could there be patients who are over- or under-dosed?

Let’s start designing the user interface (i.e., ui.R). Our dosing module should be flexible. We want to be able to give patients IV infusions with specified durations at specified frequencies. In pediatric patients, doses are calculated in mg/kg. The IV bolus is still available (for comparison purposes).

[code]

Then we want to be able to input the characteristics of the patient we would like to dose. Providing only the covariates that affect the model makes the most sense.

[code]

Last but not least, give the user the flexibility to play with variability on all parameters (but keep the publication findings as the default).

[code]

The server file (server.R) is similar to what we implemented in the beginning for the simulations in R. For more details about this go the Part 2 of ““Creating a simple pharmacometric Shiny application with mrgsolve in R”. This post has more explanations about this part.

# keywords used by Michiel to use by me in my post too.

\*.cpp file = model code