Text for blogpost

Stuff I missed:

* Changes to dosing events
* Ibuprofen co-administration
* Refer more to what Michiel did in the shiny app

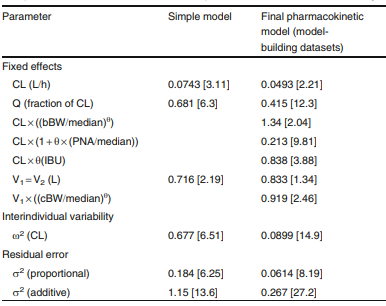
Sometimes it’s useful to build a model from literature and play around with the parameters, especially when you are trying to learn the basics of modeling & simulation. Here I’ll give a short example on how to build a population model using the information from a paper. This example relies on Part 1 & 2 of “Creating a simple pharmacometric Shiny application with mrgsolve in R” and the results of this paper [include hyperlink].

After following this example, you should be able to:

* Add covariates to a model
* Change the administration route to IV infusion

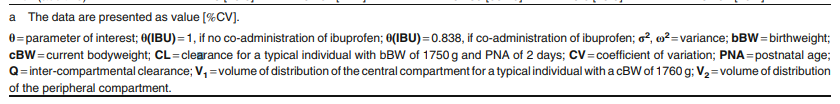
In short, De Cock *et. al.* built a population PK model to describe the disposition of amikacin (antibiotic) in preterm neonates. In the results section they state that a two-compartment model fits the data best. They also relate Q to CL and Vp = Vd. It’s common in pediatric analyses to use covariates on your parameters 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. Current weight (cBW) was implemented as an exponential covariate on Vd (and Vp since Vp = Vd). These relationships are described in Table II. Below you can find a screenshot of the first 3 columns of it showing the parameters description 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. 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 the previous post. We need to:

* Put in the final parameter estimates in the model file (as it is specific for this particular drug)
* Add initial values for the covariates and the parameters characterizing how they are related to the typical population values
* Remove the GUT compartment, as this is specific for oral administration, and this drug is normally given as IV (in this case 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 table. In this case, since this model is drug specific, you may want to restrict the user from making changes to (all of) them.

I always start with building the model just to make sure that the model I just built works the way I expect it to. Only after I have performed a sanity check - the model simulations match the publication - I start working on my Shiny app.

In Part 1 of [..] we learned how to perform simulations with mrgsolve so I’ll start with that to create the R script 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]

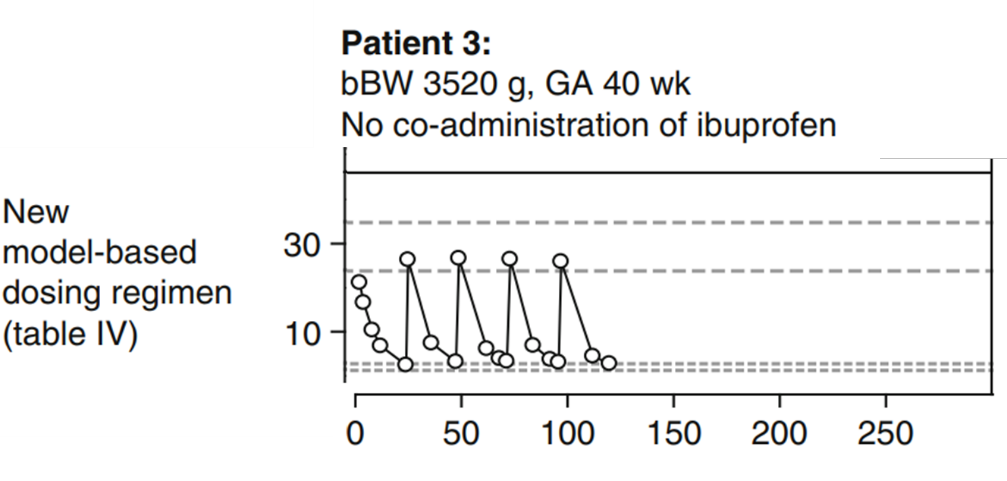
As stated previously we need to add the option to administrate an IV infusion. For this we need the rate of infusion which you can obtain by dividing the dose by the duration of the infusion (RATE = DOSE/DURATION). For amikacin, the duration of the IV infusion is between 20 – 30 min. I selected 20 min 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 to get an idea on how widely this changes to fit the needs of each patient).

[code]

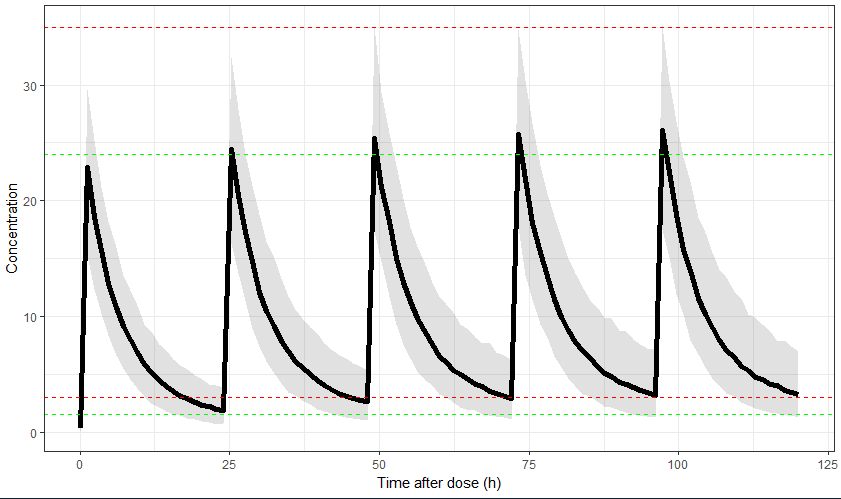
Don’t forget to look at the variability matrix and check that it provides variability only on CL and is fixed to 0 for the other parameters. We do this to match the findings of the paper. In the Shiny app we can give the user the flexibility to change the variability and observe how that impacts the 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. This is how we end up with a dose of 12 mg/kg administrated every 24 hours.



[and add here the simulation results]



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

Now let’s turn this into a shiny application. First let’s establish some goals for our app:

* See how different dosing regimen work in patients with different characteristics
* How variability impacts our simulation results. Are there patients who are overdosed or underdosed?

Let’s start with the user interface. For flexible dosing we want our user to be able to give either an IV bolus or infusion. We need to specify the dose in mg/kg, and the dosing frequency. For the infusion we need to specify the duration of the infusion extra.

[code]

Then we want to be able to change the characteristics of the patient to ensure the best dosing. Plugging in only the covariates that affect the model makes most sense.

[code]

Last but not least, give the user the flexibility to play with variability.

[code]

The server file is similar to what we had in the implementing in R part. For more details about this go the Part 2. Here you have all the functionalities described there available.

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

\*.cpp file = model code

The R code around the model