Illustrating Integration and Interpretation of the Deep Compartment Model Approach using Keras and R in a Population PK Modeling Analysis

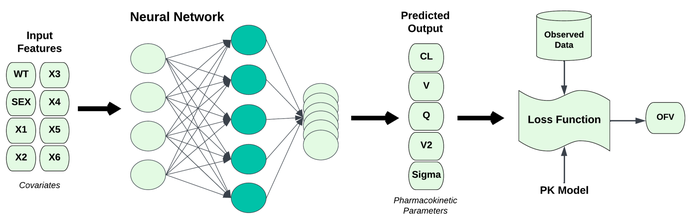
Matthew Wiens, M.A.[1], Hillary Husband, Ph.D.[1], Brian Reilly, Pharm.D., Ph.D.[1,2], Samuel P. Callisto, Ph.D.[1]
[1]*Metrum Research Group, Tariffville, CT, USA*, [2]*Vertex Pharmaceuticals Inc., Boston, MA, USA (current)*

Background and Overview

- Deep compartment models (DCMs) are a proposed alternative to traditional nonlinear mixed effect (NLME) pharmacometrics approaches [1]. DCM uses neural networks to represent estimated pharmacokinetic parameters which can then be used in either closed-form or ordinary differential equation (ODE)-based representations of pharmacokinetic models
- Herein an alternative implementation of DCMs based on the `keras` and `tensorflow` packages in R [2] is investigated. By providing a fully open-source implementation in a language that is already familiar to the pharmacometrics community, we intend to lower the barrier to entry and promote broader use of these techniques
- Shapley Additive Values (SHAP) are commonly applied to neural networks as an aid to interpretation [3]. Given the importance of interpretation of covariate effects in pharmacometric applications, our illustration of DCM includes computation of SHAP values
- Traditional NLME-based approaches quantify residual unexplained variability (RUV) using hierarchical error structures. A new method to estimate RUV in DCMs using evidential neural networks [4] was implemented in this work
- Various implementations of the DCM were compared to each other as well as a traditional NLME-based approach
- Graphics processing units (GPUs) have the potential to run neural networks quicker than central processing units (CPUs). A comparison in model fit times informs the decision to use more expensive GPU architectures

Methods

Figure 1: Workflow Representation Diagram



Simulated Population

- A hypothetical compound following two-compartment kinetics was administered intravenously to 10,000 virtual subjects, stratified into four approximately equal dose levels (5 mg, 20 mg, 50 mg, and 100 mg)
- Post-dose plasma concentrations were simulated at 30 minutes, hourly from 1-12 hours, daily on days 1, 2, 3, 5, and 9
- Covariates for the virtual population included body weight, sex, and six additional independent, theoretical covariates with each having a unique and complex covariate relationship on clearance
- Models were fit to 100 unique data sets generated by randomly sampling 250 subjects from the population of 10,000 virtual subjects without dose stratification

Deep Compartment Models

- DCMs were implemented in R using the Keras interface for TensorFlow [2]. PK parameters were estimated for each individual in this approach using the observed covariates and drug concentrations as inputs to a neural network
- Alternative network structures were compared to a base neural network defined by one input layer, two hidden layers with nonlinear activation functions, and one output layer of size five (corresponding to the estimated PK parameters: CL, V, V2, Q, and Sigma)
- Run times to fit identical DCMs were compared between GPU and 16-core CPU architectures on the flexible and autoscaling Metworx® platform
- A custom loss function representing the compartmental model was implemented using an ODE solver available in Keras, which can be extended to additional ODE-based population PK structural models

Comparison to NLME

- A population PK model was built for the data defined by a two-compartment structural model with fixed allometric scaling on all clearance and volume parameters
- Covariates were constructed using a naive full model approach which was blinded to the true covariate relationships: categorical covariates were parameterized using an exponential model, whereas continuous covariates were parameterized using a normalized power model
- Models were fit to each of the 100 data sets using the SAEM estimation method of NONMEM® version 7.5 (ICON Development Solutions, Hanover, MD)

Covariate Assessment

- Nonlinear covariate effects and covariate interactions in the DCMs were visualized and assessed using Shapley Additive Values (SHAP) [3]
- SHAP provides an explanation of how the prediction for each data point differs from the mean by deriving an effect for each covariate. These values sum to equal the difference between model mean and individual predictions
- Scatterplots and smooths of these values are used to assess the relationship between covariate value and effect in the model

Results

Deep Compartment Models: Interpretion and Analysis of Sample Sizes

- The architecture of the neural network for estimating the PK parameters and sample size in a two-compartment IV bolus dose model had a substantial effect on the ability of the model to describe covariate relationships
- The fully connected model (Figure 2) did not perform as well as a more complicated architecture estimating each parameter with a submodel (Figure 3)
- The nonlinear covariate effect was accurately identified by SHAP with around 100-250 subjects with the submodel approach requiring fewer subjects to accurately capture the effect (Figures 2 & 3)
- The interaction with sex required more data to be accurately described than the covariate effects without interactions (Figures 2 & 3)

Figure 2: Fully Connected Network for all 4 Parameters

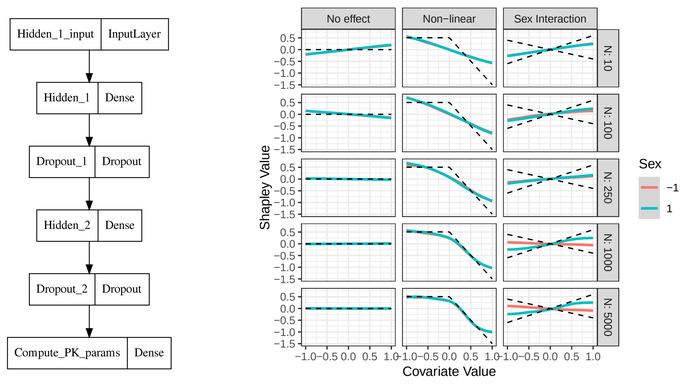
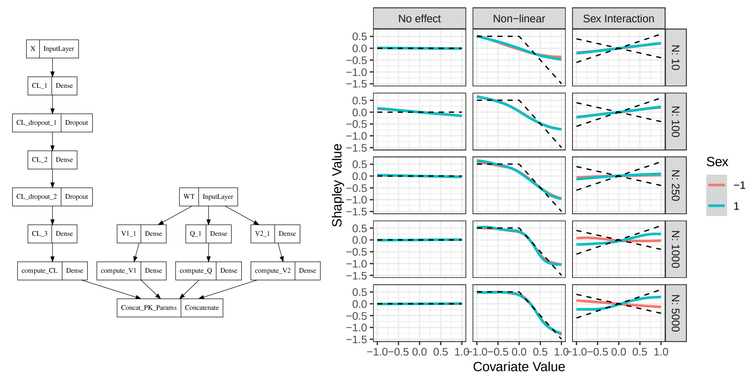


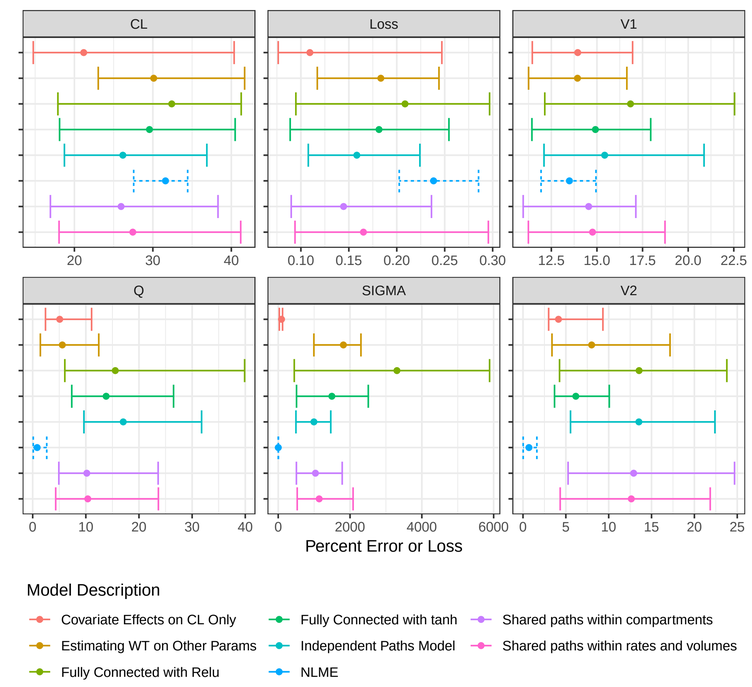
Figure 3: More complex model structure: Network architecture emphasizing estimation of covariate effects on clearance, using smaller networks with only weight as an input for the remaining PK parameters



Deep Compartment Modeling Performance

- DCMs implemented in Keras and TensorFlow successfully fit simulated PK data and generally outperformed the NLME models (Figure 4)
- The `keras` and `tensorflow` R packages allowed construction of nonlinear network architectures which incorporated prior knowledge and assumptions of the PK model structure
- In addition to PK parameters, DCMs can estimate Sigma (Σ), the RUV in a PK model
- Error estimates were higher in the fully connected networks than for the other architectures, suggesting that non-fully connected networks may estimate PK parameters more accurately in some circumstances (Figure 4)
- The NLME model failed to converge when including all possible covariates, so only the subset of covariates with visible relationships in Eta plots were included in the model using a normalized power model relationship
- Individual PK parameter estimates, including covariate effects but excluding interindividual variability (i.e. Etas), were compared to estimates from the DCM (Figure 4)
- As expected, DCMs had higher variance than NLME in estimation, but also the potential for lower bias, representing the bias-variance tradeoff in nonparametric methods
- The DCM outperformed the NLME for estimating the nonlinear covariate effects on CL and had smaller losses (Figure 4)
- The NLME outperformed DCMs for parameters without complex relationships with covariates (Figure 4)

Figure 4: Comparison of individual parameter accuracy for a variety of DCM architectures and an NLME model. 90% intervals of the median error per subject within 100 simulation replicates is plotted.



Comparisons of Run Times on Computer Architecture

- Run time per epoch was comparable for up to 2,000 simulated subjects (Table 1)
- Overhead associated with fitting models using GPU architecture resulted in slower run times for the smallest sample size (Table 1)
- At 5000 subjects, there was a >20 second improvement in run time per epoch for the GPU compared to the CPU architecture (Table 1)
- A true comparison can only be made when both platforms have been optimized for their respective processing unit; therefore, performance evaluation of CPU versus GPU should be interpreted carefully

Table 1: Comparing epoch run times between CPU and GPU architectures

N	CPU	GPU	Difference	% Difference
N=100	2.12 sec	3.82 sec	1.7 sec	44.5%
N=500	11.57 sec	10.03 sec	-1.54 sec	-15.35%
N=750	16.75 sec	14.32 sec	-2.43 sec	-16.97%
N=1000	17.98 sec	19.43 sec	1.45 sec	7.46%
N=2000	36.39 sec	34.86 sec	-1.53 sec	-4.39%
N=5000	108.52 sec	88.22 sec	-20.3 sec	-23.01%
N=7500	152.95 sec	128.2 sec	-24.75 sec	-19.31%

Conclusions

- ODE-based population PK models can be implemented as DCMs using Keras and TensorFlow, commonly-used open source tools that easily integrate with additional deep learning models
- Network architectures should consider alternative topologies and be developed for each specific case because selection of an appropriate network architecture can decrease error in parameter estimates
- Computing platforms with GPU integration can provide substantial reduction in time to train DCMs, especially in data sets with larger sample sizes
- Shapley values are a straightforward way to explore covariate relationships in these models and recapitulate the simulated linear, nonlinear, and interaction covariate effects
- Application of this approach will potentially allow pharmacometricians to integrate models with results from data science and statistics groups using these tools, leverage pre-trained models, and participate in a larger machine learning ecosystem

References

[1] Janssen, A, et al. Deep compartment models: A deep learning approach for the reliable prediction of time-series data in pharmacokinetic modeling. *CPT Pharmacometrics Syst. Pharmacol.* **11** (2022):934–945.

[2] Abadi, M, et al. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems (2015). Software available from tensorflow.org.

[3] Lundberg, S and Lee, S. A Unified Approach to Interpreting Model Predictions. In I. Guyon, U.V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan and R. Garnett, (eds.) *Advances in Neural Information Processing Systems*, volume 30 (Curran Associates, Inc., 2017).
URL `https://proceedings.neurips.cc/paper_files/paper/2017/file/8a20a8621978632d76c43dfd28b67767-Paper.pdf`

[4] Sensoy, M, et al. Evidential Deep Learning to Quantify Classification Uncertainty (2018).
URL `https://doi.org/10.48550/arXiv.1806.01768`

MetrumRG ACoP14 Posters

Repository for Sample Code:
https://github.com/metrumresearchgroup/wiens-acop-2023