Learning pharmacometric covariate model structures with symbolic regression networks

Ylva Wahlquist (ylva.wahlquist@control.lth.se)
Lund University
Jesper Sundell
Lund University
Kristian Soltesz
Lund University

Research Article

Keywords: Pharmacometrics, covariate modeling, pharmacokinetics, symbolic regression, neural networks

Posted Date: June 21st, 2023

DOI: https://doi.org/10.21203/rs.3.rs-3062691/v1

Additional Declarations: No competing interests reported.

Learning pharmacometric covariate model structures with symbolic regression networks

Ylva Wahlquist[1*], Jesper Sundell[1] and Kristian Soltesz[1]

[1]Department of Automatic Control, Lund University, P.O. Box 118, Lund, 221 00, Sweden.

*Corresponding author(s). E-mail(s): ylva.wahlquist@control.lth.se;
Contributing authors: jesper.sundell@control.lth.se;
kristian.soltesz@control.lth.se;

Abstract

Efficiently finding covariate model structures that minimize the need for random effects to describe pharmacological data is challenging. The standard approach focuses on identification of relevant covariates, and present methodology lacks tools for automatic identification of covariate model structures. Although neural networks could potentially be used to approximate covariate-parameter relationships, such approximations are not human-readable and come at the risk of poor generalizability due to high model complexity. In the present study, a novel methodology for simultaneous selection of covariate model structure and optimization of its parameters is proposed. It is based on symbolic regression, posed as an optimization problem with smooth loss function. This enables training the model through back-propagation using efficient gradient computations. Feasibility and effectiveness is demonstrated by application to a clinical pharmacokinetic data set for propofol, containing infusion and blood sample time series from 1,031 individuals. The resulting model is compared to a published state-of-the-art model for the same data set. Our methodology finds a covariate model structure and corresponding parameter values with a slightly better fit, while relying on notably fewer covariates than the state-of the-art model. Unlike contemporary practice, finding the covariate model structure is achieved without an iterative procedure involving manual interactions.

Keywords: Pharmacometrics, covariate modeling, pharmacokinetics, symbolic regression, neural networks

Introduction

Pharmacokinetics (PK) are the dynamics governing drug uptake, distribution and elimination from the body. For many drugs, it is common to assert a low-order linear and time-invariant (LTI) compartment model for PK modeling. Such low-order models typically capture the uptake, distribution, and elimination dynamics adequately. Increasing model complexity through, for example, additional compartments results in models where the parameters are not practically identifiable from data collected during clinical trials or clinical practice.

While suitable PK model structures (e.g. number of compartments and topology) for common drugs can be established from data, a notable challenge exists in that the parameter values that explain the PK for one individual, are often not suitable for another. Such inter-individual variability is partly explainable by individual-specific features that are referred to as covariates, and partly attributed to random effects.

The purpose of pharmacometric covariate modeling is to identify covariates (i.e., fixed effects) responsible for inter-individual variability, thus minimizing random effects. The gold standard is to approach this problem in a Bayesian setting using mixed-effect modeling, for which there exists both mature [1] and novel [2] tools. However, to apply mixed-effect modeling, one needs to decide which of possibly many covariates (e.g. age, body mass, gender, or genetic factors) to include. One also needs to decide on a parametric function that maps included covariates to parameters of the PK model. There is a tradition of using functions that are of sufficiently low complexity to be human-readable, and some function classes are more popular than others [3; 4]. However, for data sets including numerous covariates, selection of covariates and functions to consider is a combinatorial problem, where the search space may become limiting. Furthermore, due to step-wise approaches typically used for covariate identification, functions including multiple covariates are typically only considered if supported by prior knowledge.

In the present paper, we provide an automatic method for simultaneous covariate selection and identification of covariate functions using an adaptation of machine learning. Similarly to the current standard approach, we maintain simple human-readable expressions. The method is based on symbolic regression [5], that approximates the underlying combinatorial problem with a smooth continuous one, thus enabling the use of efficient gradient-based optimization methods.

To illustrate utility, we apply our methodology to a large data set for the drug propofol and compare the resulting covariate model with the current state-of-the-art model [4]. Our method produces a model with increased predictive performance, using fewer covariates. Furthermore, the covariate model is optimized autonomously in contrast to the prevalent iterative and manual procedure.

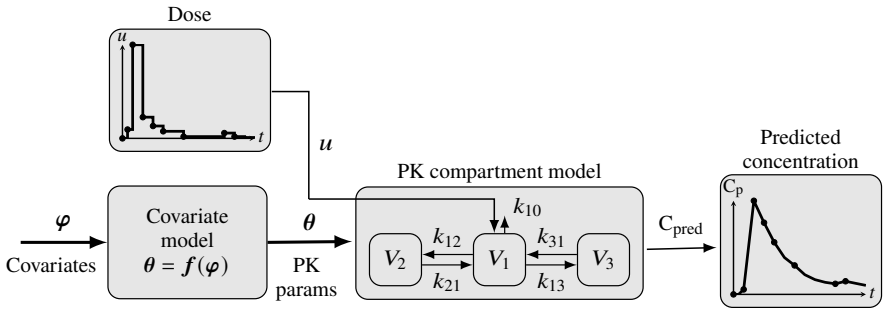


Fig. 1 Mammillary three-compartment model example illustrating our novel method. The objective is to automatically learn the covariate model $\boldsymbol{f}$ that maps a known covariate vector $\boldsymbol{\varphi}$ (comprising e.g., age, gender, or genetic factors) to the parameter vector $\boldsymbol{\theta}$ (e.g. rate constant $k_{\cdot\cdot}$ and volumes $V_{\cdot}$) of a fixed-structure pharmacometric (PK) model. The method is data-driven in that it uses drug administration profiles (time series data) u, and model-based in that it assumes a PK model of known structure. In this example, $\boldsymbol{f}$ is learned to minimize some error measure between observed (i.e. measured from samples) blood plasma concentrations and corresponding predictions C_{pred} by the model. Dots in the graphs show instances of dose changes and blood samples, respectively.

Methods

The method described in this paper aims to automatically learn closed-form pharmacometric parameter–covariate relationships from data. We will consider learning expressions for PK model parameters from drug administration and resulting blood plasma concentration data, both of which are time series, but not necessarily equitemporally or synchronously sampled. The overarching setup for this is shown in Figure 1 and we will focus on a concrete example based on a large multi-study data set published in [4] for the anesthetic drug propofol, commonly modeled with a mammillary three-compartment model [6].

Data set

We demonstrate our method on a data set composed of propofol plasma concentration observations from 1,031 individuals[1] from 30 clinical studies aggregated by Eleveld et al. [4], from here on referred to as the *data set*. Ethical approvals of the underlying studies are declared in the original publications, referenced by [4]. The data set contains 15,433 observations, of which 11,530 were arterial and 3,903 venous. Of the 1,031 individuals, there were 670 males and 361 females with ages ranging from 27 weeks to 88 years, and weights ranging from 0.68 to 160 kg.

The main reason for choosing this data set as a demonstrator is that propofol is a drug with well-studied pharmacokinetics. Evidence of this, which also

[1]The original data set in [4] comprises of 1,033 individuals but two of them were excluded due to lack of observation data.

Table 1 Covariate candidates considered in the PK model of [4] as well as in our modeling. Individuals of the data set fall within the reported ranges.

Covariate	Interpretation	Unit	Range
φ_1	age	years	0–88
φ_2	weight	kg	0.68–160
φ_3	BMI	$\mathrm{kg\,m^{-2}}$	6.2–52.8
φ_4	gender	male/female	670 M/ 361 F
φ_5	blood sampling site	arterial/venous	727 A/ 306 V

constitutes a benchmark for our demonstrator, is the model presented in [4]. Furthermore, the data set is that it has been openly disclosed by Eleveld et al., enabling transparent third-party analysis of our work.

In our model development, we consider age, weight, BMI, gender, and blood sampling site (arterial or venous) as potential covariates of our PK model. These are the same candidates as considered in [4], where individual demographics were disclosed as part of the data set.

The data was pre-processed the same way as in [4]: data points corresponding to subsequent infusion changes spaced closer than 1 s apart in the time dimension, or 0.5 $\mu\mathrm{g\,s^{-1}}$ in the dose dimension were merged.

Pharmacokinetic model

We consider a three-compartment mammillary model to describe the pharmacokinetics of propofol. The drug concentration x_i [$\mu\mathrm{g\,L^{-1}}$] in compartment $i \in \{1, 2, 3\}$ is

$$\dot{x}_1 = -(k_{10} + k_{12} + k_{13})x_1 + k_{21}x_2 + k_{31}x_3 + \frac{1}{V_1}u, \tag{1a}$$

$$\dot{x}_2 = k_{12}x_1 - k_{21}x_2, \tag{1b}$$

$$\dot{x}_3 = k_{13}x_1 - k_{31}x_3, \tag{1c}$$

where k_{ij} describes the drug transfer rate [1/s] from compartment j to i. The drug is administered at rate u [$\mu\mathrm{g\,s^{-1}}$] to the central compartment ($i = 1$), which is also where the propofol plasma concentration is measured. The volume of the central compartment is V_1 [L].

In the literature, the equivalent parameterization of volumes (V_1, V_2, V_3) and clearances (CL, Q_2, Q_3) constitutes a common alternative to (1). The conversion between these parameterizations is

$$CL = k_{10}V_1, \qquad [\mathrm{L\,s^{-1}}] \tag{2a}$$

$$Q_2 = k_{12}V_1, \qquad [\mathrm{L\,s^{-1}}] \tag{2b}$$

$$Q_3 = k_{13}V_1, \qquad [\mathrm{L\,s^{-1}}] \tag{2c}$$

$$V_2 = \frac{k_{12}}{k_{21}} V_1, \qquad \text{[L]} \quad \text{(2d)}$$

$$V_3 = \frac{k_{13}}{k_{31}} V_1. \qquad \text{[L]} \quad \text{(2e)}$$

We have chosen to implement our method using the parameterization (1) due to numeric benefits. These are further explained in [7], where we developed a fast and natively differentiable simulator for the three-order mammillary model.

Covariate model

The covariate model, shown in Figure 1, is expressed as a function $\boldsymbol{f}$ that maps the covariate vector $\boldsymbol{\varphi} = [\varphi_1, \ldots, \varphi_{n_\varphi}]^\top$ to the vector $\boldsymbol{\theta} = [\theta_1, \ldots, \theta_{n_\theta}]^\top$ of PK model parameters. Thus $\boldsymbol{f}$ has components $f_1, \ldots, f_{n_\theta}$, each mapping the covariate vector $\boldsymbol{\varphi}$ to one of the n_θ PK model parameters.

In our example with the three-compartment model for propofol, there are $n_\varphi = 5$ covariates and $n_\theta = 6$ PK model parameters according to Table 1.

Predictive performance

To assess the quality of a particular covariate model candidate $\boldsymbol{f}$, we need a performance measure that captures how well $\boldsymbol{f}$ reflects the training data. Most optimization methods, including the ones used in this paper, relies on this measure being scalar.

For comparability, we employ the same scalar performance measures as those used in [4]: We train our model to minimize an ensemble average of absolute logarithmic error (ALE). Subsequently, we evaluate predictive performance in terms of ALE, and three additional error measures: logarithmic error (LE), prediction error (PE), and absolute prediction error (APE).

For each individual in the data set, there is a vector of observation–prediction errors, where each entry corresponds to the difference between a blood sample observation and the value predicted by the model. Observation-prediction errors could thus be computed for each sample, over a time series for one individual, or the entire data set. This prompts a consistent notation and we index by ij the error over a sample j for individual i, whereas a total error over a time series for an individual is indexed by i.

The per-sample (absolute) logarithmic error (A)LE [8] is thus

$$\mathrm{LE}_{ij} = \ln(C_{\mathrm{obs}_{ij}} / C_{\mathrm{pred}_{ij}}), \tag{3a}$$

$$\mathrm{ALE}_{ij} = |\mathrm{LE}_{ij}|, \tag{3b}$$

where $C_{\mathrm{pred}_{ij}}$ are observed (measured) plasma concentrations and $C_{\mathrm{pred}_{ij}}$ are corresponding predictions produced by the model.

Similarly, the per-sample (absolute) prediction error (A)PE [9] is

$$\mathrm{PE}_{ij} = \left(\frac{C_{\mathrm{obs}_{ij}} - C_{\mathrm{pred}_{ij}}}{C_{\mathrm{pred}_{ij}}} \right) \cdot 100\% \tag{4a}$$

$$\mathrm{APE}_{ij} = |\mathrm{PE}_{ij}|. \tag{4b}$$

To avoid divisions by zero if $C_{\mathrm{pred}_{ij}} = 0$, a special casing is needed where the error is set to zero for such samples.

Using (3) and (4), corresponding per-individual errors were computed by taking the median, resulting in the median absolute logarithmic error (MdALE):

$$\mathrm{MdALE}_i = \mathrm{median}\left(\mathrm{ALE}_{ij}\right), \quad j = 1, \ldots, n_i, \tag{5}$$

where n_i is the number of entries of the time series from individual i.

To train the model of [4] and the ones considered here, a single scalar loss function representing model fit is required. This is obtained by averaging across the individuals. Training the covariate model to minimize ALE thus translates into minimizing

$$J_{\mathrm{ALE}} = \frac{1}{n} \sum_{i=1}^{n} \mathrm{MdALE}_i, \tag{6}$$

where n is the number of individuals in the data set.

Similarly to the analysis performed in [4], ALE and APE were used as indicators of model accuracy and LE and PE were used as indicators of bias. Values closer to zero for ALE and APE reflect better accuracy and values closer to zero for LE and PE indicate less bias.

Clinically acceptable ranges for MdPE_i and MdAPE_i are $10 - 20\%$ and $20 - 40\%$, respectively [10; 4; 9]. This translates to acceptable clinical ranges for bias measure $\mathrm{MdLE}_i < 0.18$ and accuracy measure $\mathrm{MdALE}_i < 0.34$.

Symbolic regression networks

At the core of our methodology lies a small artificial neural network (ANN) with a specific structure, that of a *symbolic regression network* [11], representing a simple closed-form expression. In our case, the purpose is to learn human-readable closed-form expressions describing the covariate model $\boldsymbol{f}$. A schematic illustration of such network is shown in Figure 2.

In general, ANNs can be viewed as flexible function approximators that can be trained to represent input-output mappings that fit available data. An ANN consists of n_l layers, where the output vector $\boldsymbol{x}_{l+1}$ of layer l is obtained by applying a vector of nonlinear *activation functions* $\boldsymbol{g}_l$ to an affine transformation $\boldsymbol{z}_l$ to the layer input $\boldsymbol{x}_l$:

$$\boldsymbol{z}_l = W_l \boldsymbol{x}_l + \boldsymbol{b}_l, \tag{7a}$$

$$\boldsymbol{x}_{l+1} = \boldsymbol{g}_l(\boldsymbol{z}_l). \tag{7b}$$

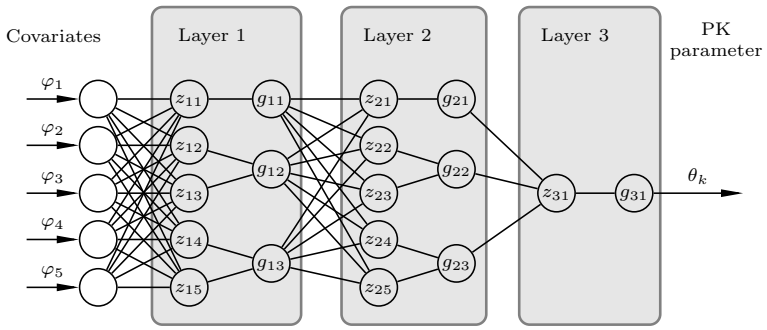


Fig. 2 Symbolic regression network with three layers, each marked by a gray box. The output of node z_{li} at layer l is the i^{th} component of $\boldsymbol{z}_l = W_l \boldsymbol{x}_l + \boldsymbol{b}_l$, where $\boldsymbol{x}_l$, W_l and $\boldsymbol{b}_l$ are the input vector, weight matrix, and bias vector of that layer. The base expressions g_{li} acting on z_{li} take on the role of activation functions used in ordinary ANNs. For example is the output of the first layer (and therefore input to the second layer) $\boldsymbol{x}_2 = \boldsymbol{g}_1(W_1 \boldsymbol{x}_1 + b_1)$. Input and output of the network is the covariate vector $\boldsymbol{\varphi} = \boldsymbol{x}_1$ and the PK parameter $\theta_k = \boldsymbol{x}_4$, respectively.

The linear weight matrices W_l and bias vectors $\boldsymbol{b}_l$ constitute the free parameters used to train the network. In the conventional case, the components of the activation functions $\boldsymbol{g}_l$ are monotonously increasing functions, such as sigmoids [12]. In contrast, a symbolic regression network can be understood as an ANN with the additional constraint that the resulting approximator $\boldsymbol{f}$ should be a human-readable closed-form expression of a mathematical function [13; 11; 14; 15]. Training of symbolic regression networks is therefore often referred to as equation learning [11]. The methodology has gained broad attention for its ability to produce impressive results in discovering (known) physical laws from data [16].

The activation functions $\boldsymbol{g}$ of a symbolic regression network represent mathematical functions that we refer to as *base expressions*. In standard symbolic regression, a sequence of topologies with different base expressions and scalar parameters–of which the weight matrices and bias vectors in (7) would constitute a special case–are evaluated in search of one that fits the available data well [17]. Conventional equation learning is thus a combinatorial problem, with poor (exponential) time complexity, and therefore often approached using genetic algorithms [17].

Instead of relying on random perturbations as in genetic algorithms, we use gradient-based optimization methods common to conventional ANN training. To ensure human-readability, we enforce sparsity of the ANN by alternating training epochs with pruning epochs, in which the least important parameters are removed from the network. We next rely on a concrete example based on the Eleveld data set, to describe and illustrate this approach.

We set out with a nominal (unpruned) network of Figure 2. It constitutes our nominal representation of f_k, where the inputs are the covariates according

to Table 1. Thus, the output of the network represents one of the PK parameters θ_k, such as k_{10} or V_1 of (1). For a model with n_θ PK parameters, we use a parallel interconnection of n_θ symbolic regression networks, each modeling one component f_k of $\boldsymbol{f}$, mapping the covariate vector $\boldsymbol{\varphi}$ to each PK parameter $\theta_k,\ k = 1, \ldots, n_\theta$.

In our example, the base expressions of each layer $l \in \{1, 2, 3\}$, with input vector $\boldsymbol{z}_l = \left[z_{l1},\ z_{l2},\ \ldots\right]^\top$, were chosen to cover previously published PK models for propofol, such as [4] and [6]:

$$\begin{aligned}
\boldsymbol{g}_1(\boldsymbol{z}_1) &= \begin{bmatrix} z_{11} \\ z_{12} \cdot z_{13} \\ |z_{14}|^{z_{15}}, \end{bmatrix} \\
\boldsymbol{g}_2(\boldsymbol{z}_2) &= \begin{bmatrix} z_{21} \\ z_{22} \cdot z_{23} \\ \frac{z_{24}}{z_{25}+1}, \end{bmatrix} \\
g_3(z_3) &= |z_3|,
\end{aligned}$$

where the g_3 assures positive output of the final layer. The division in $\boldsymbol{g}_2$ has the term one in the denominator to assure that the output does not blow if $z_{25} \geq 0$ approaches zero.

Training

Minimizing the *loss function* (6) across trainable parameters of the covariate model $\boldsymbol{f}$ is referred to as training. In our case, these parameters are stacked into a vector $\boldsymbol{\gamma}$, made up by the elements of all weight matrices and bias vectors. Since the data set is static, we can view training as minimization of the scalar-valued function $J_{\text{ALE}}(\boldsymbol{\gamma})$. In order to do so, we need to evaluate $J_{ALE}(\boldsymbol{\gamma})$. Doing so requires simulation of one PK model for each data set individual, to obtain the predicted plasma concentration at each observation time instance. This can be efficiently done using the method introduced in [7].

In addition to the supporting fast simulation, the method of [7] enables exact and efficient evaluation of derivatives of individual prediction errors with respect to the trainable model parameters. This allows us to train the covariate model $\boldsymbol{f}$ using conventional ANN back-propagation, using the stochastic gradient-based optimization algorithm ADAM [18]. Our implementation, using the neural network package Flux [19], relies on the differential programming capabilities of the Julia language [20]. A full disclosure of our implementation is found in the GitHub repository [21].

Pruning

To obtain simple human-readable expressions from an initially dense symbolic regression network, such as the one in Figure 2, we alternate between parameter training of the fixed network structure and pruning of the network.

The goal of this pruning is to obtain a sparse network structure, which translates into a readable expression of the corresponding covariate model. In the pruning process, we remove covariates and network parameters that have relatively little influence on the network output. This process is visually exemplified in in Figure 3.

It is desirable to only include as many covariates as needed to explain the data [22]. Therefore, we start by identifying and removing the least important covariates from the symbolic regression network to obtain expressions with fewer covariates. Next, we prune network parameters $\boldsymbol{\gamma}$ (linear weights and biases) to obtain simple covariate expressions. Pruning a network parameter is achieved by fixing its value to zero and removing it from the vector $\boldsymbol{\gamma}$ of trainable parameters.

Before each pruning iteration, we train the network until convergence. This translates into finding a local minimum of the loss function in (6). At such minima, the partial derivatives of the loss function with respect to the trainable parameters are zero. Therefore, the second derivatives define the first non-zero terms of the Taylor series expansion that describe local parameter sensitivities, as further explained in appendix A. The second-order derivatives make up the elements of the Hessian matrix. Specifically its diagonal elements represent sensitivities in the corresponding individual parameters.

In the Taylor series expansion, the second-order terms take on the form

$$S(\gamma_k) = \gamma_k^2 H_k, \tag{8}$$

where H_k is the k^{th} Hessian diagonal element of the loss function with respect to the network parameter γ_k. $S(\gamma_k)$ denotes *salience* of a parameter γ_k [23].

For pruning of the network parameters $\boldsymbol{\gamma}$, (8) may be used directly. However, computing the salience for a covariate is not as straightforward since the Hessian elements would differ between individuals. A solution to this is to sum the salience contribution of each individual,

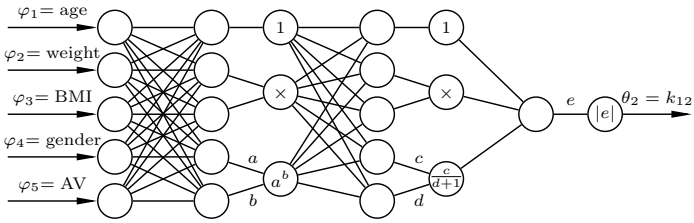
$$S(\varphi_k) = \sum_{i=1}^{n} \varphi_{ik}^2 (H_i)_k, \tag{9}$$

where $(H_i)_k$ is the k^{th} diagonal element of the Hessian, H_i, determining the sensitivity of the covariate φ_{ik} for individual i, and n is the number of individuals.

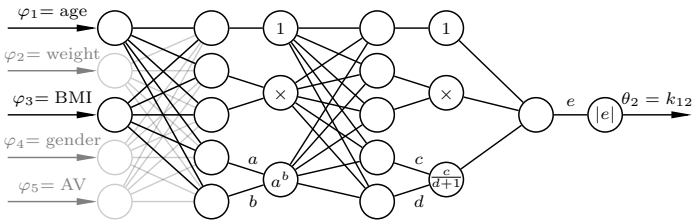
We use the Zygote package [24] in Julia to compute the Hessian diagonal elements. In appendix A, we give a more detailed description, providing mathematical insight into this methodology.

Recipe: Symbolic regression

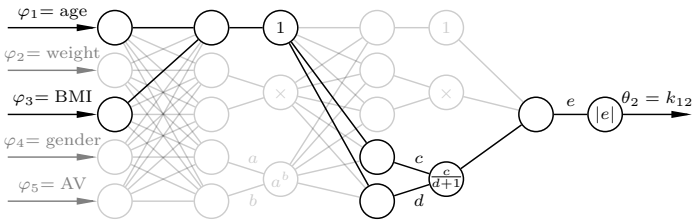
A compact summary of our training and pruning scheme is provided below. Initially, we start with a nominal symbolic regression network, like the one shown in Figure 2. It is sequentially trained and pruned until we obtain a final



(a) Nominal (dense) symbolic regression network



(b) Symbolic regression network after covariate pruning



(c) Final symbolic regression network after covariate and parameter pruning

Fig. 3 Pruning sequence of a symbolic regression network with output pharmacokinetic parameter $\theta_2 = k_{12}$. Input covariates are age, weight, gender, and arterial or venous sampling (AV). The nominal network has three dense layers, each followed by base expressions such as 1 (feedforward), multiplication, power function, division and absolute value. A black line represents a connection between two nodes, and a gray line represents a pruned (removed) connection. The final network represents the covariate expression of (10b)

expression of sufficient complexity and fit. Our training and pruning sequence of a symbolic regression network is as follows:

1. Choose a nominal symbolic regression network architecture, and corresponding base expressions.
2. Train the network until convergence.
3. Compute the salience of each (remaining) covariate, $S(\varphi_k)$ of (9).
4. Sort the covariates by saliency and remove the covariate with the smallest salience.
5. If the desired final number of covariates is reached, continue to step 6, otherwise return to step 2.
6. Train the reduced network until convergence.

7. Compute the salience of each (remaining) trainable network parameter, $S(\gamma_k)$ of (8).
8. Sort the network parameters by saliency and remove N parameters with the smallest salience.
9. If the desired final number of network parameters is reached, continue to step 10, otherwise return to step 6.
10. Train the reduced network until convergence.
11. Convert the resulting network to a readable functional expression.

The number of covariates and network parameters to keep in the final symbolic regression network is a trade-off between fit to data and complexity of the final expression. In our example, illustrated in Figure 3, we remove one covariate at each pruning iteration, until only two covariates are left. Next, we remove the $N = 10$ least sensitive parameters in the first parameter pruning iteration, and then one parameter per subsequent pruning iteration until only twelve network parameters are left. More details of the pruning and training can be found in [21].

Initialization of the parameter values associated with step 1 of the recipe affects the fit of the resulting final model. To mitigate the risk of poor model fit due to an unfortunate initialization, the recipe could be run several times, where the best fitting model is kept. In our example, we have executed the recipe eight times.

Limits of performance

Structural mismatch between the asserted PK model structure (1) and the data, in combination with measurement errors, induce upper and lower limits on prediction errors of the trained covariate model. Here we explain how these limits can be characterized by training two additional models.

Even with a very complex covariate model, one cannot expect perfect fit to data (zero loss). This is because the fixed (three-compartment) PK model structure is only an approximation of the actual pharmacokinetics, combined with (blood sample) measurement errors. Part of the loss remaining after applying our covariate modeling scheme can thus be attributed to this mismatch. To indicate how much, we optimize one set of (three-compartment) PK parameters for each individual in the data set. This results in a completely covariate-free model. While it will fit data better than any covariate model, it does not generalize. This makes it practically useless for purposes other than providing an upper limit for performance.

Another natural question to ask is how much we gain (in terms of loss) by considering covariate dependencies. To do this, we optimize a constant, i.e. covariate-free, model—one where all individuals share the same (three-compartment) PK parameter values. This model does constitute a lower bound of the achievable predictive performance that any covariate-based model should beat.

For these two additional models, PK parameter optimization was done with the optimization package Optim.jl [25], to minimize the same loss as for our covariate model based on symbolic regression. Implementation details can be found in [21].

Results

Applying the proposed methodology to the Eleveld data set [4], resulted in the following covariate model, mapping covariates to rate constants and central compartment volume of the PK model (1):

$$k_{10} = 2.76 \cdot 10^{-5}\mathrm{WGT} + 0.00342 \quad [\mathrm{s}^{-1}] \quad (10a)$$

$$k_{12} = \left| \frac{18\mathrm{AGE} - 0.815\mathrm{BMI} - 1877}{72.8\mathrm{AGE} - 3.3\mathrm{BMI} - 7428} \right| \quad [\mathrm{s}^{-1}] \quad (10b)$$

$$k_{13,\mathrm{male}} = \frac{0.749\mathrm{AGE}^2 + 23.7\mathrm{AGE} + 2602}{354\mathrm{AGE}^2 + 11191\mathrm{AGE} + 601067} \quad [\mathrm{s}^{-1}] \quad (10c)$$

$$k_{13,\mathrm{female}} = \frac{0.75\mathrm{AGE}^2 - 23.7\mathrm{AGE} + 2602}{354\mathrm{AGE}^2 - 11191\mathrm{AGE} + 601067} \quad [\mathrm{s}^{-1}] \quad (10d)$$

$$k_{21} = \left| 1.46 \cdot 10^{-6}\mathrm{BMI}^2 - 1.54 \cdot 10^{-5}\mathrm{BMI} - 6.75 \cdot 10^{-7}\mathrm{BMI} \cdot \mathrm{WGT} + 0.00218 \right| \quad [\mathrm{s}^{-1}] \quad (10e)$$

$$k_{31,\mathrm{male}} = 4.52 \cdot 10^{-5} + 2.18 \cdot 10^{-7}\mathrm{AGE} \quad [\mathrm{s}^{-1}] \quad (10f)$$

$$k_{31,\mathrm{female}} = 4.52 \cdot 10^{-5} - 2.18 \cdot 10^{-7}\mathrm{AGE} \quad [\mathrm{s}^{-1}] \quad (10g)$$

$$V_1 = 6.78 \cdot 10^{-4}\mathrm{AGE} + 0.117\mathrm{WGT} - 5.34 \cdot 10^{-4}\mathrm{WGT}^2 - 2.5 \cdot 10^{-4}\mathrm{AGE} \cdot \mathrm{WGT} - 0.0557. \quad [\mathrm{L}] \quad (10h)$$

AGE, WGT, BMI represent age [years], weight [kg] and body mass index [$\mathrm{kg\,m^{-2}}$]. The subscript male or female indicates different PK parameter expressions depending on gender. The blood sampling site (arterial or venous) was available as a modeling covariate, but was automatically pruned by the symbolic regression algorithm.

The obtained model of (10) is less complex than the Eleveld model in [4], provided in appendix B for reference, and comparable to simpler covariate models for propofol, such as [3; 10].

The predicted concentrations of the final covariate model on the Eleveld data set are shown in Figure 4 together with the corresponding Eleveld model predictions. The distribution of errors between predicted and observed predictions is shown in the boxplot in Figure 5, indicating comparable predictive capability of the models, despite our model being less complex, and involving fewer of the available covariates. For ease of comparison, we present the corresponding average prediction errors in Table 2. We compare our model to the Eleveld model, to a covariate-free PK model where all individuals share the same parameters values, and to a covariate-free individual models with unique parameter sets. As expected, adding covariates explains some, but not all, variability between patients. This can be seen by comparing our model to the constant PK model and the individual models. As seen in Table 2, all of the prediction errors: MdLE, MdALE, MdPE, and MdAPE fall within clinically acceptable ranges, as presented further above.

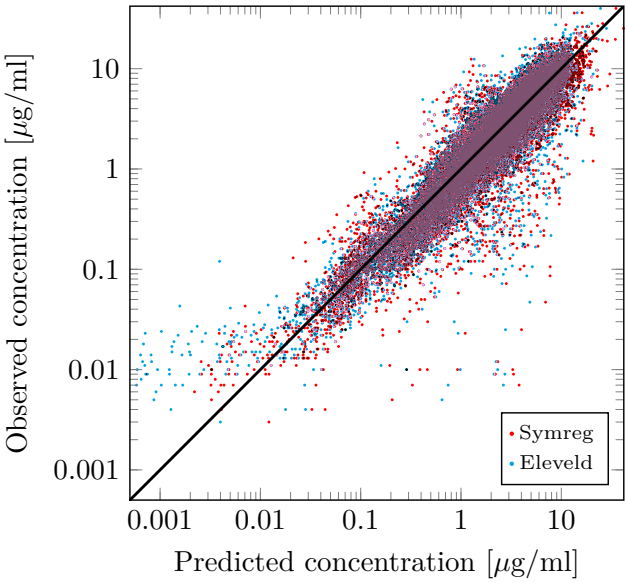


Fig. 4 Predicted versus observed propofol concentrations of our covariate model (Symreg, red) compared to the Eleveld covariate model in [4] (Eleveld, blue) in logarithmic scale. The identity function, representing a perfect model fit, is shown in black.

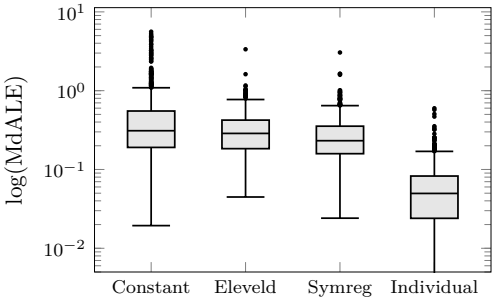


Fig. 5 Comparison of prediction error MdALE (5) between predicted and observed propofol concentrations for pharmacokinetic models. Our covariate model is denoted Symreg and the Eleveld covariate model is described in [4]. The constant model represents one parameter set over the population and Individual represents individual set of model parameters. The lower whisker for the individual models goes to zero.

Discussion

We have introduced a novel symbolic regression methodology for simultaneously automating the search for a suitable covariate model structure and optimization of its parameters. Similar to contemporary methodologies, it relies on a user-specified set of base expressions from which the covariant

Table 2 Comparison of prediction errors for the propofol data set in [4] (1,031 individuals) for several pharmacokinetic models, all trained with MdALE (5) as loss. Individual and constant PK model(s) are shown for comparison, representing best and worst case limits, respectively.

Method	Mean MdALE	Mean MdLE	Mean MdAPE	Mean MdPE
Constant model	0.501	0.140	211	181
Eleveld model	0.325	0.0791	34.8	14.4
Symbolic regression	0.279	-0.0489	27.4	-0.266
Individual models	0.0623	$1.65 \cdot 10^{-4}$	6.21	0.0178

model can be composed. However, and in important contrast to contemporary methods, the need for a combinatorial search across combinations of these expressions is voided.

Throughout the paper, a propofol PK modeling example has been used as a demonstrator. Within this example, the proposed methodology manages to accomplish slightly better data fit than the result of state-of-the-art modeling [4] (see Figures 4 and 5, Table 2), while relying on fewer covariates, see (10). The obtained values of individual volumes and clearances were comparable to those in [4].

The introduced methodology is broadly applicable to PK modeling from time series data, and likewise to pharmacodynamic (PD) modeling, and combined pharmacokinetic and pharmacodynamic (PK/PD) modeling. It's main benefit lies in that it poses the search for a suitable PK model as symbolic regression with a smooth loss function. This, in combination with efficient methods for simulation and gradient computations [7] enables efficient model learning using back-propagation. Another, related, advantage is that the method can find covariate functions that are both simple and explain available data, while having a structure that would generally not be considered in a manual model structure search, unless explicitly supported by prior knowledge.

In this paper, we have focused on models that are of sufficiently low complexity to be human-readable, which is achieved by enforcing sparsity of the neural network that constitutes the expression tree of the covariate model. If human readability is not necessary, an ordinary deep ANN could be employed instead. However, there is also a trade-off between fit to training data and generalization to yet unseen data due to possible over-fitting. Enforcing human-readability naturally limits flexibility of the model, thus decreasing this risk of over-fitting.

We assessed the model's out-of-sample prediction performance using a five-fold cross-validation. The data set was divided into five equal parts, and a covariate model was trained on four of the partitions, excluding one each time. The excluded partition, referred to as the validation set, was used to evaluate the model. This process was repeated for all five partitions to obtain an average predictive performance. The resulting mean (range) MdALE from cross-validation on the test sets was 0.303 (0.152–0.423), similar to the mean

MdALE of 0.279 in Table 2. There was thus only a modest 10 % difference in mean error between the training and validation sets, which suggests no overfitting issue. If such a problem had occurred, reducing the number of covariates and parameters (harder pruning) could have balanced the prediction errors on both sets.

The way we have employed the methodology here differs from how covariate models are usually trained using mixed-effect modeling. Rather than asserting parameter priors and selecting the most likely parameters from the posterior distribution that the data infers, we have chosen to train a scalar loss function. It would in theory be possible to embed our methodology within an inference engine, but at the cost of high computational cost. If a Bayesian interpretation is desired, a likely better alternative is to first run our methodology to arrive at a covariate model structure–as we have done in our example–and then assert parameter priors to the parameters of the resulting model, to finally apply mixed-effect modeling to compute the corresponding posteriors.

Conclusion

We have presented a novel methodology for automatic and simultaneous covariate model structure discovery and parameter optimization. This model was demonstrated using an example on which it outperforms state-of-the-art modeling, in that it finds expressions that match data slightly better, while relying on notably fewer covariates. We conclude that the potential of automated model structure discovery is substantial; it could greatly optimize the process of pharmacometric covariate modeling. Additionally, it's likely to provide an improved balance between model complexity and data fit. This improved balance is something that could be challenging to achieve when simply assessing a series of pre-set model structure candidates in sequence.

Acknowledgments. We would like to thank Fredrik Bagge Carlson with JuliaHub for support in setting up the computational framework that we have used in the implementation of our proposed methodology.

Author contributions. All authors designed the study, Wahlquist performed the coding, Wahlquist and Soltesz analyzed the results. All authors authored, critically reviewed and approved the manuscript for submission.

Funding. This work was partially supported by the Wallenberg AI, Autonomous Systems and Software Program (WASP) funded by the Knut and Alice Wallenberg Foundation. All authors are members of the ELLIIT Strategic Research Area at Lund University.

Data availability. Code and data for reproducing the results is available in [21].

Declarations

The authors declare no conflict of interest.

Appendix A Hessian-based pruning

In this appendix we provide a mathematical interpretation of the roles played by the Hessian and parameter saliences in our pruning approach.

The effect of perturbing the parameter vector can be analyzed by approximating the loss function $J(\boldsymbol{\gamma})$ by a Taylor series. A perturbation $\partial\boldsymbol{\gamma}$ of the parameter vector $\boldsymbol{\gamma}$ will change the loss function by

$$\partial J(\boldsymbol{\gamma}) = \nabla J(\boldsymbol{\gamma})\partial\boldsymbol{\gamma} + \frac{1}{2}\sum_i H_{ii}\partial{\gamma_i}^2 + \frac{1}{2}\sum_{i\neq j} H_{ij}\partial\gamma_i\partial\gamma_j + \mathcal{O}(||\partial\boldsymbol{\gamma}||^3),$$

where $\partial\gamma_i$ are the components of $\partial\boldsymbol{\gamma}$, $\nabla J(\boldsymbol{\gamma})$ is the gradient of the loss function and H_{ij} are the elements of the Hessian matrix H of J with respect to $\boldsymbol{\gamma}$ so that

$$\nabla J(\boldsymbol{\gamma}) = \frac{\partial J(\boldsymbol{\gamma})}{\partial\boldsymbol{\gamma}} \qquad H_{ij} = \frac{\partial^2 J(\boldsymbol{\gamma})}{\partial\gamma_i\partial\gamma_j}.$$

The goal is to prune the parameters that are *least* sensitive, i.e. those that affect the loss J the least when perturbed. Even for our size of network, repeatedly computing the full Hessian H would notably slow down training of the symbolic regression model. Instead, we make a diagonal approximation of the Hessian, neglecting cross terms H_{ij} with $i \neq j$.

Allowing training to converge before each pruning iteration, ensures that $\nabla J(\boldsymbol{\gamma}) = 0$ (or in practice negligible), thus enabling approximation of $\partial J(\boldsymbol{\gamma})$ by

$$\partial J(\boldsymbol{\gamma}) \approx \frac{1}{2}\sum_i H_{ii}\partial\gamma_i^2.$$

This gives us an importance measure (salience) of our network parameters, so that the salience of parameter γ_i becomes

$$S(\gamma_i) = H_{ii}\gamma_i^2.$$

Appendix B The Eleveld model

The Eleveld PK covariate model in [4] is defined by

$$
\begin{aligned}
&f_{\text{ageing}}(x) = \exp\left(x(\text{AGE} - \text{AGE}_{\text{ref}})\right),\\
&f_{\text{sigmoid}}(x, E50, \lambda) = \frac{x^{\lambda}}{x^{\lambda} + E50^{\lambda}},\\
&f_{\text{central}}(x) = f_{\text{sigmoid}}(x, \theta_{12}, 1),\\
&f_{\text{CLmaturation}} = f_{\text{sigmoid}}(\text{PMA}, \theta_{8}, \theta_{9}),\\
&f_{\text{Q3maturation}} = f_{\text{sigmoid}}(\text{AGE} + 40\text{weeks}, \theta_{14}, 1),\\
&f_{\text{opiates}}(x) = \begin{cases} 1, & \text{absence of opiates}\\ \exp\left(x \cdot \text{AGE}\right), & \text{presence of opiates,}\end{cases}\\
&f_{\text{Al-Sallami}} = \begin{cases} \left(0.88 + \frac{0.12}{1+(\text{AGE}/13.4)^{-12.7}}\right)\left(\frac{9270\text{WGT}}{6680+216\text{BMI}}\right), & \text{males}\\ \left(1.11 + \frac{-0.89}{1+(\text{AGE}/7.1)^{-1.1}}\right)\left(\frac{9270\text{WGT}}{8780+244\text{BMI}}\right), & \text{females,}\end{cases}\\
&V_{1,\text{arterial}}(\text{L}) = \theta_1 \frac{f_{\text{central}}(\text{WGT})}{f_{\text{central}}(\text{WGT}_{\text{ref}})},\\
&V_{1,\text{venous}}(\text{L}) = V1_{\text{arterial}}\left(1 + \theta_{17}(1 - f_{\text{central}}(\text{WGT}))\right),\\
&V_2(\text{L}) = \theta_2 \frac{\text{WGT}}{\text{WGT}_{\text{ref}}} f_{\text{ageing}}(\theta_{10}),\\
&V_3(\text{L}) = \theta_3 \frac{f_{\text{Al-Sallami}}}{f_{\text{Al-Sallami, ref}}} f_{\text{opiates}}(\theta_{13}),\\
&CL_{\text{male}}(\text{L/min}) = \left(\frac{\text{WGT}}{\text{WGT}_{\text{ref}}}\right)^{0.75} \frac{f_{\text{CLmaturation}}}{f_{\text{CLmaturation, ref}}} f_{\text{opiates}}(\theta_{11}),\\
&Q_{2,\text{arterial}}(\text{L/min}) = \theta_5 \left(V_2/V_{2,\text{ref}}\right)^{0.75}\left(1 + \theta_{16}(1 - f_{\text{Q3maturation}})\right),\\
&Q_{2,\text{venous}}(\text{L/min}) = Q_{2,\text{arterial}} \cdot \theta_{18},\\
&Q_3(\text{L/min}) = \theta_6 \left(V_3/V_{3,\text{ref}}\right)^{0.75} \frac{f_{\text{Q3maturation}}}{f_{\text{Q3maturation, ref}}},
\end{aligned}
$$

where the reference patient is male, 35 years old, 70 kg and 1.7 metres tall. All parameters values θ can be found in the original publication of [4].

References

[1] Sheiner LB, Beal SL (1980) Evaluation of methods for estimating population pharmacokinetics parameters. i. Michaelis-Menten model: Routine clinical pharmacokinetic data. J Pharmacokinet Biopharm 8(6):553–571. https://doi.org/10.1007/BF01060053

[2] Rackauckas C, Ma Y, Noack A, et al (2020) Accelerated predictive healthcare analytics with pumas, a high performance pharmaceutical modeling and simulation platform. bioRxiv https://doi.org/10.1101/2020.11.28.402297

[3] Marsh B, White M, Morton N, et al (1991) Pharmacokinetic model driven infusion of propofol in children. Brit J Anaesth 67(1):41–48. https://doi.org/10.1093/bja/67.1.41

[4] Eleveld DJ, Colin P, Absalom AR, et al (2018) Pharmacokinetic–pharmacodynamic model for propofol for broad application in anaesthesia and sedation. Brit J Anaesth 120(5):942–959. https://doi.org/10.1016/j.bja.2018.01.018

[5] Davidson JW, Savic DA, Walters GA (2003) Symbolic and numerical regression: Experiments and applications. Inf Sci 150(1):95–117. https://doi.org/10.1016/S0020-0255(02)00371-7

[6] Schnider TW, Minto CF, Gambus PL, et al (1998) The influence of method of administration and covariates on the pharmacokinetics of propofol in adult volunteers. Anesthesiology 88(5):1170–1182. https://doi.org/10.1097/00000542-199805000-00006

[7] Wahlquist Y, Bagge Carlson F, Soltesz K (2023) Fast simulation of pharmacokinetics. IFAC PapersOnline

[8] Masui K, Upton RN, Doufas AG, et al (2010) The performance of compartmental and physiologically based recirculatory pharmacokinetic models for propofol: A comparison using bolus, continuous, and target-controlled infusion data. Anesth Analg 111(2):368–379. https://doi.org/10.1213/ANE.0b013e3181bdcf5b

[9] Varvel JR, Donoho DL, Shafer SL (1992) Measuring the predictive performance of computer-controlled infusion pumps. J Pharmacokinet Biopharm 20(1):63–94. https://doi.org/10.1007/BF01143186

[10] Schüttler J, Kloos S, Schwilden H, et al (1988) Total intravenous anaesthesia with propofol and alfentanil by computer-assisted infusion. Anaesthesia 43(s1):2–7. https://doi.org/10.1111/j.1365-2044.1988.tb09059.x

[11] Martius G, Lampert CH (2017) Extrapolation and learning equations. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, https://doi.org/10.48550/arXiv.1610.02995

[12] Dubey SR, Singh SK, Chaudhuri BB (2022) Activation functions in deep learning: A comprehensive survey and benchmark. Neurocomputing 503:92–108. https://doi.org/10.1016/j.neucom.2022.06.111

[13] Orzechowski P, La Cava W, Moore JH (2018) Where are we now? a large benchmark study of recent symbolic regression methods. In: Proceedings of the Genetic and Evolutionary Computation Conference, Kyoto, Japan, pp 1183—1190, https://doi.org/10.1145/3205455.3205539

[14] Sahoo SS, Lampert CH, Martius G (2018) Learning equations for extrapolation and control. In: Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholm, Sweden, Proceedings of Machine Learning Research, vol 80. PMLR, pp 4439–4447, https://doi.org/10.48550/arXiv.1806.07259

[15] Kim S, Lu PY, Mukherjee S, et al (2021) Integration of neural network-based symbolic regression in deep learning for scientific discovery. IEEE Trans Neural Networks Learn Syst 32(9):4166–4177. https://doi.org/10.1109/TNNLS.2020.3017010

[16] Udrescu SM, Tan A, Feng J, et al (2020) AI feynman 2.0: Pareto-optimal symbolic regression exploiting graph modularity. In: Proceedings of the 34th International Conference on Neural Information Processing Systems, NIPS'20, pp 4860–4871, https://doi.org/10.48550/arXiv.2006.10782

[17] Koza JR (1992) Genetic Programming: On the Programming of Computers by Means of Natural Selection. MIT Press, Cambridge, MA, USA, https://doi.org/10.1007/BF00175355

[18] Kingma DP, Ba J (2017) Adam: A method for stochastic optimization. In: Proceedings of the 3rd International Conference on Learning Representations (ICLR), https://doi.org/10.48550/arXiv.1412.6980

[19] Innes M (2018) Flux: Elegant machine learning with Julia. J Open Source Softw https://doi.org/10.21105/joss.00602

[20] Bezanson J, Edelman A, Karpinski S, et al (2017) Julia: A fresh approach to numerical computing. SIAM Rev 59(1):65–98. https://doi.org/10.1137/141000671

[21] Wahlquist Y (2023) Learning pharmacometric structures. URL https://github.com/wahlquisty/learning-pharmacometric-covariate-structures, commit: dfafb1f

[22] Jonsson EN, Karlsson MO (1998) Automated covariate model building within NONMEM. Pharm Res 15(9):1463–1468. https://doi.org/10.1023/a:1011970125687

[23] LeCun Y, Denker J, Solla S (1989) Optimal brain damage. In: Advances in neural information processing systems, vol 2. Morgan-Kaufmann

[24] Innes M (2018) Don't unroll adjoint: Differentiating SSA-form programs. CoRR URL arXiv.1810.07951

[25] Mogensen PK, Riseth AN (2018) Optim: A mathematical optimization package for Julia. J Open Source Softw 3(24):615. https://doi.org/10.21105/joss.00615