Journal of Pharmacokinetics and Pharmacodynamics, Vol. 33, No. 2, April 2006 (© 2006)
DOI: 10.1007/s10928-006-9004-6

A Genetic Algorithm-Based, Hybrid Machine Learning Approach to Model Selection

Robert R. Bies,[1,*] Matthew F. Muldoon,[2] Bruce G. Pollock,[3] Steven Manuck,[4] Gwenn Smith,[5] and Mark E. Sale[6]

Received August 4, 2005—Final January 10, 2006—Published Online March 28, 2006

We describe a general and robust method for identification of an optimal non-linear mixed effects model. This includes structural, inter-individual random effects, covariate effects and residual error models using machine learning. This method is based on combinatorial optimization using genetic algorithm.

KEY WORDS: nonlinear mixed effects modeling; covariate selection; automated machine learning; genetic algorithm; population paramacokinetics; model building.

GRANT SUPPORT: NIBIB P41 EB001975, NIMH PO1 HL40962, MH064823.

OBJECTIVE

To describe the application of a genetic algorithm (GA) based machine learning approach to compartmental pharmacokinetic and pharmacodynamic model selection.

[1]Department of Pharmaceutical Sciences and Psychiatry, University of Pittsburgh, Pittsburgh, PA, USA.
[2]Center for Clinical Pharmacology, University of Pittsburgh, Pittsburgh, PA, USA.
[3]Rotman Research Institute, Centre for Addiction and Mental Health, University of Toronto, Toronto, Canada.
[4]Department of Psychology, University of Pittsburgh, Pittsburgh, PA, USA.
[5]Albert Einstein College of Medicine, The Zucker Hillside Hospital, Glen Oaks, NY, USA.
[6]Next Level Solutions, Raleigh, NC, USA.
*To whom correspondence should be addressed. E-mail: rrb47@pitt.edu

INTRODUCTION

The last two decades have seen tremendous progress in numerical and statistical methods for estimating parameters for a given model from a given data set. Less progress has been made in the process by which the model itself is selected. The basis for model selection has changed little from that originally described by Pinheiro *et al.* (1). This model selection process, however it is accomplished, is essentially a combinatorial optimization. That is, the model selection process consists of searching for the best combination of structural features (e.g., number of compartments, lag times, bioavailability, effects, and/or non-linear processes), covariate effects (e.g., linear effect of weight on clearance), and random effects (e.g., additive interindividual error on clearance or proportional residual error). More recent work has provided valuable tools for detecting or eliminating features, notably, the use of *post hoc* estimates for covariate detection (2) and generalized additive models (3). However, the basis of all of these methods remains a step-wise search. What is frequently called "step-wise regression" in the pharmacokinetic/pharmacodynamic (PK/PD) model building literature is a categorical search algorithm known as a local search or a binary search (4,5). This algorithm is generally regarded as much less robust (less likely to find the true optimal solution) although very efficient (minimized with few function evaluations). Among the reasons for the lack of robustness in step-wise regression is the assumption of independence of the effects, i.e., that a two-compartment model will be "better" than a one-compartment model, regardless of other components of the model. This has been shown not to be the case (6).

The Search Space

The "fitting" of model parameters to a data set typically uses a derivative based optimization algorithm. In derivative based non-linear regression, the goal is to find the values of continuous, real valued parameters that optimize some objective function, commonly a goodness of fit statistic. An n-dimensional Euclidian space is defined, where n is the number of parameters. The optimal solution is the point in the space with the minimum (or maximum) value for the goodness of fit statistic. Several algorithms are available to search such a space, most of them based on partial derivatives of the goodness of fit statistic with respect to each dimension (parameter). Commonly, this search space is finite, where most or all of the dimensions have upper and lower boundaries. An analogous search space can be defined for combinatorial optimization. For combinatorial optimization, the dimensions are collections of mutually exclusive features. For example, a PK/PD model search space may include the number of

compartments. That is, one might be interested in searching across models with 1–3 or perhaps more compartments. This constitutes a set of mutually exclusive features, a model must have exactly 1, 2 or 3 compartments. It must have a whole number of compartments. It cannot have 1 and 2, nor can it have 1.5 compartments. Similarly, there may or may not be a lag time in the model. The lag time dimension has two possible values, and constitutes another set of mutually exclusive features. These two-dimensions (number of compartment and presence/absence of lag time) might represent a simple two-dimensional search space, with six possible solutions (all possible combinations of three possible values for number of compartments and two possible values for lag time). The important differences between a continuous (Euclidean) search space and a categorical or discrete search space are:

- A finite number of solutions in the combinatorial space.
- There is no derivative in the combinatorial space.

Further, combinatorial space can be divided into ordered categorical and strictly categorical spaces. Ordered categorical spaces consist of values that can be unambiguously ranked as more or less than the other values. In PK/PD modeling, most of the dimensions are strictly categorical. The exception is the number of compartments, where two compartments is "more", i.e., a greater number than one. However, having a lag time is not clearly more or less than not having one and an additive residual error is not more or less than a proportional residual error.

Many algorithms for combinatorial optimization have been developed including simplex, integer programming, simulated annealing, and GA. One algorithm has been developed that uses a more global approach to selecting covariate models, using the Wald Approximation (7). This approach may overcome the problems of local search, although it is limited to covariate selection.

Of these, simplex and integer programming are largely limited to ordered categorical variables. In the optimization of PK/PD models, the possible values in a set are typically not ordered. It is not clear how selecting a PK/PD model can be easily fit into a simulated annealing approach. For these reasons, a machine learning approach, based on GA (8–11) was developed.

Background

The GA is a very general, very robust optimization algorithm. The GA is based on the mathematics of evolution/natural selection and survival of the fittest. Like evolution, GA is also very inefficient, and makes frequent mistakes. Also like evolution, the power of GA is based on relatively large

numbers and something good happening fairly rarely, then preserving that good feature through selection.

Derivative based search algorithms assume that there is a downhill direction, and if one keeps moving that way, it will continue to go downhill, and further that the second derivative with respect to the objective function informs something about how large a step can be taken before it starts going uphill. The GA makes no such assumption; indeed there is no derivative, or even downhill in a strictly categorical space. The GA assumes only that good features (e.g., two compartments or presence of a lag time) *tend* to be good features regardless of other features. A combination of features may be what provides the "good feature". If a particular feature is less "fit" it tends to disappear. Unlike the step-wise approach, where a limited number of tests (often one) is done to determine whether a feature (e.g., lag time) should be included, GA does not completely abandon any feature. Features that result in less fit models only tend to decrease in frequency in the population. Features can always reappear either by recombination or mutation, at any point in the search. Indeed, even if a feature such as presence of a lag time completely disappears from the population, it can reappear at any time through mutation. A typical mutation rate in is 0.01, so that if the population size is 300, a lag time will appear (and be tested again, in new combinations) on average three times de novo in every generation.

Most of the applications of GA have been in the engineering fields, including finding optimal geographical locations for mobile phone towers, optimal design for computer chips and designing antennae (12). There are also applications in the financial sector, specifically applied to investment portfolio management (13,14). In addition, feature recognition has been approached using a hybrid GA approach (15). Applications in biology have been somewhat more limited. Most application in biology have been in the area of molecular structural resolution (16,17) and bioinformatics (18,19). We have applied machine learning, based on genetic algorithm to PK/PD model selection.

How GA Works

Assume one has a data set for which a compartmental model is sought. The initial obstacle to solving a problem using GA is to describe the solution space to be searched. The solution space is defined as sets of mutually exclusive features. Let us consider the problem of finding the best PK model, considering only models with 1, 2 or 3 compartments, with or without a lag time, and with additive or proportional residual error. There are 12 possible solutions in the solution space:

1 compartment, without lag time, additive residual,
1 compartment without lag time, proportional residual,
1 compartment, with lag time, additive residual,
1 compartment with lag time, proportional residual,
2 compartment, without lag time, additive residual,
2 compartment without lag time, proportional residual,
2 compartment, with lag time, additive residual,
2 compartment with lag time, proportional residual,
3 compartment, without lag time, additive residual,
3 compartment without lag time, proportional residual,
3 compartment, with lag time, additive residual,
3 compartment with lag time, proportional residual.

As this is a small set, an exhaustive search could be performed, creating and fitting each model to a data set, then evaluating each model and deciding which model best describes the data. However, exhaustive search is not practical when the search space becomes large. So, we would like to search this candidate model space using GA. We have three dimensions in the solution space, specifically

- Number of compartments.
- Presence of lag time.
- Residual error model.

In a deviation from true evolution, GA typically uses a binary language. Evolution uses a language with four possible values. These values are the possible "values" for nucleotide in DNA; A, C, T, and G. A string of these A|C|T|G values is assembled into a "gene", and the gene determines a feature (blue eyes, etc). The language of GA is similarly discrete, except there are only two values (binary), 0 and 1. These 0s and 1s are assembled into genes, which determine features of the model. In this example, the first dimension is the "number of compartments" gene. There are three possible phenotypes, 1–3. Two bits (and therefore four genotypes) will be required to describe these three possible phenotypes

- 0,0 will be 1 compartment,
- 0,1 or 1,0 will be 2 compartment,
- 1,1 will be three compartment.

Note that this code is degenerate, i.e., more than one bit string will result in a two-compartment model. This is not a problem, and in fact replicates the natural DNA language.

The second dimension, the "presence of lag time" gene has only two values, and so, only one bit is needed.

- 0 will be without lag time,
- 1 will be with lag time.

The third dimension, the "residual error" gene also has two values, one bit is needed.

- 0 will be additive residual error,
- 1 will be proportional residual error.

These three genes (number of compartments, presence of lag time, and residual error model) are then concatenated to form the genome. There will be a total of three genes, and a total of four bits in this genome.

Model (phenotype)	genome (genotype)
1 compartment, without lag time, additive	0,0,0,0
1 compartment without lag time, proportional	0,0,0,1
1 compartment with lag time, additive	0,0,1,0
1 compartment with lag time, proportional	0,0,1,1
2 compartment without lag time, additive	0,1,0,0 or 1,0,0,0
2 compartment without lag time, proportional	0,1,0,1 or 1,0,0,1
2 compartment, with lag time, additive	0,1,1,0 or 1,0,1,0
2 compartment with lag time, proportional	0,1,1,1 or 1,0,1,1
3 compartment without lag time, additive	1,1,0,0
3 compartment without lag time, proportional	1,1,0,1
3 compartment with lag time, additive	1,1,1,0
3 compartment with lag time, proportional	1,1,1,1

The search is initialized by creating and initial "population". A population in GA is a set of "individuals", each corresponding to a genome, which in turn specifies a model. The initial population (in this example, we'll use a population size of 4) is created randomly. So, we randomly create four genomes, perhaps

0,1,0,0
1,0,1,1
1,1,0,0
and
0,0,1,1

These genomes are then translated into the phenotypes

2 compartment, no lag time, additive residual error,
2 compartment, lag time, proportional residual error,

3 compartment, no lag time, additive residual error,
and
1 compartment, with lag time, proportional residual error.

These four models can then be created and standard methods used to fit the model parameters to the data. Assume that these are the outcomes of the fitting

Individual	Goodness of fit statistics	Number of parameters
1	133	5 (ka, ke, k12, k21, and volume)
2	132	6 (ka, ke, k12, k21, volume, and lag time)
3	129	7 (ka, ke, k12, k21, k13, k31, and volume)
4	137	4 (ka, ke, volume, and lag time)

The next obstacle to implementing a GA search is to define the "fitness" of the model. In evolutionary biology, fitness is the ability of an individual to propagate into the next generation. That is, the expected value of the number of offspring an individual will produce/2. This ratio is used as two individuals are combined to produce a single progeny. A value greater than 1.0 results in the genes in that individual tending to increase in the population, a value of less than one suggest that those genes will become less common. Fitness then, in evolutionary biology is an objective function—the objective of evolution/natural selection being to propagate genes. Populations, through natural selection, in some way, strive for fitness. In GA, fitness is also the overall objective function (to be distinguished from the individual model fitting objective function). To define the fitness, it is necessary to ask, what it the optimal outcome? In optimizing cellular telephone tower location, the optimal solution is some combination of minimal cost and maximal reliability of service, in antennae design it may be some combination of minimal cost, minimal power consumption, maximal reliability and maximal transmission fidelity. For PK/PD models, we typically have a list of optimal outcomes from a model. Typically, we would like a model to:

- Have a small (large) goodness of fit statistic.
- Be parsimonious.
- Perhaps pass some tests of statistical validity (positive definite covariance matrix, absolute values of off-diagonal element of estimation correlation matrix < 0.95, ratio of largest to smallest eigenvalues < 1000, perhaps even passing posterior predictive check).

These can be combined into a fitness function. For this example, we assume that the goodness of fit statistic is a −2*log likelihood. This is used as the basis for the fitness, and we add a penalty for each outcome that is undesirable. If we want a parsimonious model, a penalty could be added to each estimated parameter. It might be determined that we would prefer a model that has one fewer parameter (more parsimonious) even if the −2*log likelihood was 10 points higher. Then, the fitness would be:

$$\text{fitness} = \text{OBJ} + 10 \cdot \text{Number of parameters}$$

where OBJ is −2*log likelihood and number of parameters is the number of parameters in the model.
So, for this example, the fitness's of the models would be:

Individual	Goodness of fit	Number of parameters	fitness
1	133	5 (ka, ke, k12, k21, and volume)	183
2	132	6 (ka, ke, k12, k21, volume and lag time)	192
3	129	7 (ka, ke, k12, k21, k13, k31, and volume)	199
4	137	4 (ka, ke, volume and lag time)	177

And so, we would prefer model 4, even though the goodness of fit statistic is higher, because it is more parsimonious.

Once the fitness is calculated for each individual in the population, the next generation of individuals can be created. Subsequent generations are created by first selecting sets of parents. Parents are selected randomly, with replacement, with probability of selection proportional to fitness. Typically raw fitness is not used, but instead is scaled to prevent individual large or small values from dominating the probabilities. In addition, for this model, we need to reverse the fitness, we are searching for a lower value of fitness (−2*log likelihood + penalties), that would result in a highscaled fitness, resulting in a higher probability of selection. Let's assume that the individuals in the first generation selected as parents for the next generation are individuals 4, 1, 4, and 2.

Note that the most fit individual (no.4) was selected twice—selection with replacement is done, and the least fit (no.3) wasn't selected at all. These parents are then grouped into pairs. Assume these pairs:

4, 1
4, 2

where each of these numbers corresponds to the models described above.

Next, the genomes of the parents are lined up and "crossover" occurs. As in biology, the genetic material from the parents is recombined by randomly selecting some number of points in the genome. The left part of one parents genome is then combined with the right part of the other parents genome, and vice versa, In the current implementation of GA, this is done in some user-defined fraction of the parent sets (perhaps 70%), at a single, random location in the genome. Assume that the first pair of parents (4 and 2) are to be crossed-over. First, their genomes are lined up:

Individual 4 genome 0 0 1 1
Individual 1 genome 0 1 0 0

A random location is chosen for cross over (assume location = 2). Both genomes are divided after the second bit.

The left part of individual 4 (0,0) is combined with the right part of individual 1 (0,0) giving a progeny (0,0,0,0). Correspondingly, the right part of individual 4 (1,1) is combined with the left part of individual 1 (0,1), giving a progeny (0,1,1,1). Finally, mutation is applied to the two progeny, wherein some small fraction (typically 0.001–0.01) or the bits are reversed. Given the small genome and population size, mutation is unlikely to occur in this example.

The same process is applied to the other set of parents, assume that cross over occurs at bit 3. The original genomes for individuals 4 and 2 are:

Individual 4 genome 0 0 1 1
Individual 2 genome 1 0 1 1

The resulting progeny are:

1,0,1,1
0,0,1,1

Note that the cross over had no effect, since bits 3 and 4 are the same in both genomes.

Mutation is again applied after the crossover, assuming this time, that one bit is randomly chosen to be reverse, the third bit in the first individuals. The final genomes for these two progeny then will be:

1,0,0,1
0,0,1,1

This results in a new generation with the following genomes:

0,0,0,0
0,1,1,1
1,0,0,1
0,0,1,1

corresponding to the models

1 compartment, no lag time, additive residual error,
2 compartment, lag time, proportional residual error,
2 compartment, no lag time, proportional residual error,

and

1 compartment, with lag time, proportional residual error.

The new generation results tend to preserve the good qualities of the previous generation, while recombining them into new combinations of features, and even generating (through mutation) features not present in the previous generation. Deviations from the biological process include ensuring that the best individual from a generation is preserved for the next generation. If this is not done, it is not uncommon for even the best individual to be entirely lost and not to be found again for many generations.

Other features have been found to improve robustness and or convergence of the GA search. These include "nicheing" (19), wherein a penalty is applied to individuals who are similar to a large number of other individuals. Similarity is defined as being different at no more than a user specified number of bits—called the "niche-radius". This technique preserves diversity in the population, preventing too rapid convergence, and a more thorough search. This has been implemented in the current method, additional details are given in Appendix A.

The current implementation of machine learning uses a number of other methods in addition to GA that help to fine tune the selection process (i.e., that this is not strictly a GA but a hybrid GA). The GA has an interesting property of being quite good at finding generally good solutions, but very inefficient at finding the very best in a small region. Downhill methods tend to be efficient at finding local minima in small regions. These two methods then are complementary when used together, giving the robustness of GA and the efficiency of downhill search. In the current method a downhill search is performed after every 10 generations of GA. The downhill search is implemented as follows.

- The best (lowest fitness) individual in the population is identified.
- A new population is created, wherein each individual is the best individual with one bit reversed. The new population will be of size N, where N is the number of bits in the genome.
- Those models are run, and the fitness calculated.
- If the fitness of the best individual in the new population is better (lower) than that of the previous population, the process is

repeated, creating another population, based on the new best individual.

- This process is repeated until no further improvement is seen.

When the downhill step is completed, the GA resumes again.

Convergence of GA

Convergence of GA can be difficult to determine. Determination of convergence is done in part by inspection of a plot of fitness vs generation, and noting when improvement has not been seen for several generations. Occasionally, no improvement will be seen for several generations, after which improvement resumes. This can occur in non-linear regression as well. One method we have found to be useful is re-randomization. In re-randomization, once improvement has not occurred for several generations, the entire population, except the single best individual is discarded and recreated randomly. This allows the niches to be reformed and the process to (largely) begin again. This is in some ways analogous to the application of multiple chains to assess convergence in Markov Chain Monte Carlo analysis (20). Experience suggests that if no improvement is seen after re-randomization is done twice, convergence has occurred.

METHODS

The current method is incorporated into a special use application, written in Microsoft Visual Studio and Compaq Digital Fortran. The machine learning method implements NONMEM version 5.1 (21), by creating NMTRAN control files, calling NMTRAN, compiling the resulting code and extracting the required values (e.g., $-2^*\log$ likelihood, number of parameters, eigenvalues, success of covariance step, estimation correlation matrix) directly from the NONMEM commons. The method for creating the NMTRAN control files will be discussed.

First, a "control file template" is created. The template resembles an NMTRAN control file, except that it contains special purpose labels, that are searched for and replaced by other strings (called tokens) to result in a syntactically correct NMTRAN control file. For example

The line
$SUBS ADVAN

in a control file template contains the user defined label ADVAN. The user then defines a group of mutually exclusive strings (called token groups)

than can be substituted into the control file template in place of this label. For example, one might consider a 1, 2 or 3 compartment model. In these cases, the relevant tokens would be

"ADVAN2 TRANS1"
"ADVAN4 TRANS1"

and

"ADVAN12 TRANS1"

Substituting this text (without the quotes) in place of the label (ADVAN) in the control file template will result in syntactically correct NMTRAN code

$SUBS ADVAN2 TRANS1
$SUBS ADVAN4 TRANS1

or

$SUBS ADVAN12 TRANS1

However, creating a full control file is more complicated, as there are frequently interdependencies in the code. For example, if this model is to be one compartment (ADVAN2), there is no requirement for defining K12 or K21, but there is if the model is to be two compartment. Therefore, tokens must occur as sets, and be substituted into control file templates for label sets. For example, for ADVAN12, there is a requirement to define K23, K32, K24, and K42, Also, requirements for initial estimates for these parameters in the $THETA block. Therefore an example of labels (with the corresponding block records in an NMTRAN control file) will be:

$SUBS ADVAN(1)
.
.
.
$PK
ADVAN(2)

$THETA
ADVAN(3)

The set of labels is (ADVAN(1), ADVAN(2), and ADVAN(3))
and the corresponding token sets, for 1–3 compartment models will be:

1 compartment
token 1 = "ADVAN2"
token 2 = " "

token 3 = " "
2 compartment
token 1 = "ADVAN4"
token 2 = "K23=THETA() /n K32=THETA()"
token 3 = "(0,1) /n (0,1)"

3 compartment
token 1 = "ADVAN12"
token 2 = "K23=THETA() /n K32=THETA()/n K24=THETA() /n K42=THETA()"
token 3 = "(0,1) /n (0,1) /n (0,0.5) /n (0,0.5)"

where

" " is a blank token and */n* is carriage return line feed.

Note that if token set three is specified (by the number of compartments gene for this individual model), the tokens for the three-compartment model will be substituted into the control file template, resulting in the following code:

```
$SUBS ADVAN12
.
.
.
$PK
     K23=THETA()
     K32=THETA()
     K24=THETA()
     K42=THETA()
.
.
.
$THETA
     (0,1)
     (0,1)
     (0,0.5)
     (0,0.5)
```

Inspection will show that other values for the number of compartments gene will result in similar code.

Note that the specific index for each THETA can only be determined after the model has been defined, and then can be inserted into the $PK and $THETA block.

EXAMPLE

Datasets

A pilot study for the effects of IV citalopram on neuroendocrine measures in normal subjects was used for the initial pharmacokinetic analysis. The study was approved by the Institutional Review Boards at all institutions and all patients gave informed consent to participate in the study. This dataset comprised 377 concentration samples from 58 individuals enrolled at North Shore University Hospital (NSUH), New York and at the University of Pittsburgh Medical Center (UPMC), Pittsburgh, PA. The individuals at NSUH were administered 40 mg over 50 min ($n = 23$). At the University of Pittsburgh, 25 individuals received 10 mg over 30 min. and 10 individuals received 20 mg over 30 min. The sampling times at NSUH were 0, 60 (end of infusion) 75, 90, 120, and 150 min. The sampling times at UPMC were 0, 30 (end of infusion), 45, 90, and 150 min.

Stepwise and GA approaches assessed the covariates age, weight and sex on the disposition parameters (clearances and volumes of distribution). In addition to the fixed effect of covariates, presence/absence of random effects (inter individual and residual error models) were addressed. The two approaches were compared based on the lowest objective function value, the number of parameters in the models and the size of the random effects distributions detected across the methods. Both methods (stepwise and GA) can address other components of the model (e.g., number of compartments or lag time). However, in this case, the basic structure was well defined from previous experience, and was not investigated.

Step-wise Approach

A standard step-wise search for a model structure was conducted using NONMEM V using Fortran G77 (22). Model evaluations were conducted using First-Order Conditional Estimation (FOCE) with interaction.

Construction of the Base (covariate free) Model

The base model was established using a two-compartment model structure while testing various inter-individual and residual variability structures in the model building prior to evaluating any covariate effects.

Specifically, all inter-individual parameter variabilities were implemented as exponentiated functions mimicking a log-normal distribution.

The relationship between the population pharmacokinetic parameter and its variance was evaluated using the structure:

$$\mathrm{CL}j = \mathrm{TVCL}^* \exp(\mathrm{eta}(1))$$

where $\mathrm{CL}j$ was the clearance for individual j, TVCL was the typical value for clearance for the population and eta(1) represented the difference of the individual's estimate from the population mean where the distribution of eta across individuals was centered at zero with variance ω^2.

Residual variability was evaluated with several structures including an additive, proportional and combined additive/proportional structure. The specific structures are shown below:

Additive error: $y_{ij} = F_{ij} + \varepsilon ps(1)_{ij}$

Proportional error: $y_{ij} = F_{ij}(1 + \varepsilon ps(1)_{ij})$

Combined additive and proportional error: $y_{ij} = F_{ij}(1 + \varepsilon ps(1)_{ij}) + \varepsilon ps(2)'_{ij}$,

where y_{ij} was the ith observation in the jth individual, F_{ij} was model prediction for that particular individual and observation and $\varepsilon ps(1)_{ij}$ (or $\varepsilon ps(2)'_{ij}$) is a normally distributed random error with a mean of zero and a variance of σ^2.

The base model with the inter-individual and residual error structures was created first using the objective function of NONMEM as a guide with improvement of fitness determined by a change of at least 3.84 between models with one degree of freedom (corresponding to a p value of 0.05). In addition, the distribution of residuals and various diagnostic graphics (i.e., Pred vs. Observed, IPRED vs. Observed) were evaluated in this step of the stepwise model building process. Random and fixed (non-covariate) effects were evaluated and established prior to the covariate search using this criterion.

Stepwise Assessment of Covariate Effects

Covariates were evaluated initially with individual predicted values (eta) from the model built using the method described above under FOCE with interaction using Xpose (23) and other graphical techniques. Specifically, the eta deviations were evaluated for each of the parameters graphically vs. the value of the covariate being assessed. Covariates that appeared to show a relationship across these eta values for the PK

model parameters were then selected and incorporated stepwise in a forward manner testing additive, proportional and exponentiated mathematical forms for the relationship of covariate to parameter. The mathematical forms for the continuous covariate are shown below:

$$\mathrm{TVCL} = \theta_{\mathrm{CL}} \times (\mathrm{Cov}/\mathrm{Med}_{\mathrm{Cov}})^{\theta_{\mathrm{Cov}}}$$
$$\mathrm{TVCL} = \theta_{\mathrm{CL}} + (\mathrm{Cov} - \mathrm{Med}_{\mathrm{Cov}}) \times \theta_{\mathrm{CLCov}}$$
$$\mathrm{TVCL} = \theta_{\mathrm{CL}} \times \exp((\mathrm{Cov}/\mathrm{Med}_{\mathrm{Cov}}) \times \theta_{\mathrm{CLCov}})$$

where Cov is the continuous covariate being tested, the θ values represent population operators modulating the covariate to establish the population parameter for an individual with a specific set of characteristics and the $\mathrm{Med}_{\mathrm{Cov}}$ is the median value of the continuous covariate assessed
Discrete covariates were incorporated as follows:

$$\mathrm{TVCL} = \theta_{\mathrm{CL}} + (1 - \mathrm{Sex}) \times \theta_{\mathrm{Sex}}$$

where the TVCL is the population value for clearance for an individual with specific covariate characteristics and θ_{sex} is the operator adjusting the clearance value for sex (where the binary discrete covariate—in this case sex—is coded as a 1 or 0).

Following addition to the model, the covariates were removed from the full model in a stepwise manner and a worsening of objective function was used as a criterion for continued inclusion in the model (objective function change > 3.84, $\mathrm{df} = 1$) For comparison of covariate models that were not hierarchical (e.g., linear and exponential), selection was based on inspection of plots and the value of -2^* log likelihood. Sixty models were evaluated using the step-wise approach

Machine Learning Approach (GA)

The search space for the machine learning approach included:

Covariate Relationship

Age, Weight, and Gender effect on clearance
Age, Weight, and Gender effect on central volume
Age, Weight, and Gender effect on intercomparmental clearance
Age, Weight, and Gender effect on peripheral volume
The search for relationships between covariates and parameters included the following relationships:
No relationship
Additive relationship (parameters = base value + theta*covariate)

Proportional (parameters = base value*(1+ theta*covariate)
and
Exponential (parameters = base value*(exp(theta*covariate))
All covariates were centered and scaled for use in these expressions.

Random Interindividual Effects

Interindividual random effect on clearance
Interindividual random effect on central volume
Interindividual random effect on intercomparmental clearance
Interindividual random effect on peripheral volume and
Presence of off-diagonal elements in the interindividual covariance matrix. (OMEGA)

Random Residual Effects

Additive, proportional and combined additive and proportional residual error

The penalties used for calculating the fitness are given in Table I. The options for the search are given in Table II. The GA template code is given in Appendix B. The penalties used were based on promoting the survival of the most "parsimonious" model with a penalty of 10 points per additional parameter in both the random and fixed effects domains. In addition, major criterion concerning informativeness (i.e., covariance step), correlation and convergence were associated with significant penalties. This penalties dramatically reduce the fitness of a model that did not successfully converge and/or have a covariance step and/or have highly correlated parameters.

RESULTS

Stepwise Analysis

The first row of Table III shows the results from the stepwise analysis of the dataset. Specifically, weight was found to be a significant covariate on clearance (CL) and peripheral volume of distribution (V2) (obj = 1267 with covariates, 1375 without). The inclusion of this covariate decreased the objective function significantly (108 points total $p < 0.001$, FOCE Interaction).

GA Analysis

Rows 2 and 3 of Table III show the results for the GA analysis in this dataset. The two "best" models based on objective function and GA fitness function, respectively, had objective functions that were at least 150 points lower than the best model found stepwise. Two different results

Table I. Penalty Function for GA Combined with NONMEM Objective Function

Model element	Penalty added to NONMEM objective function
Theta criteria	10
Omega criteria	10
Sigma criteria	10
Covariance criteria	400
Correlation > 0.95	300
Unsuccessful convergence	400
Eigenvalue penalty	100

Table II. Genetic Algorithm Settings

Algorithm element	Setting
Cross over/genome	0.7
Mutation rate	0.01
Upper limit of scaled fitness	4
Lower limit of scaled fitness	0.2
Population size	300
Generation limit	50
Lower limit for non-crash	−9999999
NONMEM timeout (minutes)	600
Generation timeout (minutes)	6000
Number of niches	4
Niche radius (Hamming distance)	4

based on whether or not the objective function of NONMEM or the fitness function of the GA were used to select the "best' model. The model with the lowest objective function (obj = 1109) included the covariate age on clearance and weight and clearance on $V2$. This model had inter-individual variance terms (eta) on inter-compartmental clearance (Q), $V1$, and $V2$. The model selected by the genetic algorithm with the best "fitness" function value had a NONMEM objective function value of 1113 and had a successful covariance step. It is important to note that the decision on the "final" model from either approach (traditional model selection of GA) should not be based on objective function alone, therefore we do not conclude that GA found a "better" model, only that it found several with lower objective functions. The control stream for this model is shown in Appendix C. Like the other model selected, inter-individual variance was present on the same three fixed effects and the covariate effects

Table III. Modeling Results for Stepwise and GA

Analysis	CL L/min (%se)	Q L/min (%se)	$V1$ L (%se)	$V2$ L (%se)	%CV ηCL (%se)	%CV ηQ	%CV η$V1$	%CV η$V2$	Exp WTCL	Exp WTV2	Exp AGE $V2$
Stepwise FOCE_INTER Obj= 1267	0.96	17.7	143	659	54.7	81.2	86.7	54.8	0.6	1.4	
GA best objective function Obj= 1109 WT AGE EXP ON CL AGE ON V2	1.19 (72)	15.2 (10)	191 (6)	761 (7)	0	14.5 (32)	82.2 (32)	1.8 (25)	3.95 (33)	0.75 (40)	−1.03 (20.9)
GA best fitness function Obj= 1113 WT ON CL EXP AGE ON V2 EXP	0.5 (58)	15.2 (7)	191 (13)	741 (8)	0	68.8 (32)	32.7 (27)	9.0 (35)	3.47 (54)	NA	0.6 (47)

of weight on clearance and age on $V2$ were found to be significant. In this assessment, 11820 unique models were evaluated using the machine learning algorithm.

DISCUSSION

In the case of IV citalopram datasets, the machine learning approach consistently found a model with a significantly better objective function (NONMEM objective function). In addition, the between-subject variances were typically lower and the robustness of the parameter estimates was improved. The improved robustness of parameter estimates is suggested by the observation that the covariance step was not successful for the stepwise assessment of the dataset.

A central question in this method is whether the quantitative measures of model "goodness" are sufficient for model building, or whether subjective evaluation of models is essential. Clearly, inspection of plots is essential in the model selection process, at a minimum for understanding sources of bias. A relationship between posthoc etas and a demographics variable may suggest a hypothesis that should be incorporated into the model. Automated methods do not preclude this approach. When such a relationship

is suspected, one can simply include every reasonable relationship into the search space. The concern is not that inspection of plots cannot be done, but that the user will elect not to do so, since the application itself does not require it. To facilitate the use of informative graphics in model selection, plotting tools are incorporated into the present application. Work is underway to permit the search space to be expanded with such hypotheses adding dimensions to the search space while the search is ongoing, rather than incorporating addition features into subsequent GA analyses.

Selecting the final/best model will typically require some subjective evaluation, including informative graphics. An automated approach does not preclude this; one is not obligated to accept the model with the lowest fitness. All models that are created are available for evaluation and graphical inspection. The final choice of the best model is typically left to the user, the algorithm only creates them, runs them and presents then in a table, with the quantitative measures. However, it is possible that fully pre-specified analyses could be done. If a drug was already well characterized (e.g., linear), a search space could be defined prior to data availability. In addition the various options and criteria could be specified in a data analysis plan. In a traditional model building approach, fully pre-specifying an analysis is not practical. The number of possible combinations quickly becomes impractical. However, using a combinatorial optimization approach, enormous search spaces can be searched with reasonable efficiency, and good robustness. In the present application, the number of dimensions to be searched is not limited and we have experience with search spaces of up to 10^{22} models. Formally the search space it is limited to the size of the array that holds the genome, which is $>32,000$. We have used genomes with a length of >80, and >60 dimensions. In this way, a population pharmacokinetic analysis could be as objective as a traditional statistical analysis. It is unlikely that this approach—precluding exploratory analysis—would very frequently lead to optimal models, but, if objectivity were essential, it could be done.

An important limitation of the automated approach, in the present implementation, is that hierarchical models have identical parsimony penalties as non-hierarchical models(24). That is, the theoretical distribution of the log-likelihood function is known for hierarchical models, and the method for testing statistical significance of an additional parameter is in common use. The distribution of the log-likelihood function for non-hierarchical models is not known. However, the Akaike information criteria (AIC)(25) does provide some guidance. Method to apply different criteria to non-hierarchical model (AIC) comparison and hierarchical model comparisons (log likelihood ratio test) are under consideration.

Much discussion has occurred over issues of model selection. This approach applies established optimization methods to model selection.

This is in some opposition to the view that model selection is essentially a statistical process. Practice however suggests that traditional model selection is not a statistical process. Once cannot reduce the inspection of graphics, or the value of biological plausibility to statistical criteria. Similarly, in the present application, strict statistical criteria are not generally applied.

The choice of the model, even constrained to random effects selection and covariate identification, poses a significant challenge and suggests that model building be undertaken with great care. Despite the computational intensity required for the GA search, global search algorithms may be able to improve model selection and help to reveal effects in smaller datasets than those necessary for identification under stepwise conditions. Demonstrating any improvement in robustness for this method over traditional methods will require additional experience, on a sample size larger than that addressed in the work described here. Since these methods are automated, they require dramatically less personnel time, (although dramatically more computer time). Finally, again, as this method is automated, there may be somewhat fewer questions of subjectivity in the model selection process, which may make scientific and regulatory review more straightforward.

It is important to point out that experience to date does not justify a conclusion that a machine learning approach to model selection is "better" than a traditional approach. Most importantly, we have presented only a single example, essentially a study with a sample size of 1, and even for that single sample, we do not conclude that the machine learning solution is "better". Automated search algorithms are, without question, more robust than "trial and error methods". However, it is also very likely that all of the qualities desired in a model (biological plausibility, clinical relevance, value in decision making) cannot be reduced to quantitative measures. It is therefore difficult to incorporate these aspects of model selection into an automated algorithm. It should also be pointed out that, to a significant degree, inspection of graphics can be reduced to quantitative measures, and current work on this algorithm focuses on this (26–28).

It is clear that a great deal more experience will be required before the behavior of machine learning approaches to model selection is well understood. Issues such as appropriate selection of parsimony penalties (for hierarchical and non-hierarchical models), problems with specifying multiple initial estimates, and penalties for non-statistical features (success of covariance step and surrogates for graphical inspection [e.g., posterior predictive check and prediction discrepancy]) remain to be addressed.

Readers interested in obtaining the software for non-commercial use are encouraged to contact the authors.

APPENDIX A—DETAILS OF PENALTIES AND OPTIONS

Penalties

The "fitness" in Genetic algorithm describes the optimal conditions for the results. Standard practice suggests that the objective function (frequently -2^* log likelihood of the data |model) should form the basis for this fitness. In addition, there are other desirable attributes of a model. Table I gives a list of the penalties (criteria) used for the present analysis. First among the desired attributes for most modeling is parsimony. The Akaike information Criteria(24) suggest that a penalty be applied to (-2*log likelihood) for each parameter estimated. This approach is implemented. Akaike suggests a penalty of 2. If models are hierarchical, a value of 3.84 (corresponding to a Chi-square value of $p = 0.05$ with one df) can be justified. This is viewed as anti-conservative by some in the field of PK/PD modeling, and a larger value is frequently used. In the present analysis, a value of 10 was used for each estimated parameter, intersubject variance term (including off diagonal elements) and residual variance term.

A third desired attribute of a model is that the minimization converges successfully. Experience has suggested that a very large penalty/criterion for this improves convergence speed. A large penalty forces the algorithm to first find a model that converges successfully, (largely regardless of objective function, number of parameters etc.), then refine this model to optimize the objective function and parsimony. Similarly, a large penalty for a successful covariance step and having the absolute value of any off-diagonal elements of the estimation correlation matrix > 0.95 is used. In the present application, a penalty also can be applied if the search is to include the first-order method vs. first-order conditional (not used in this analysis), and if it is desired that the ratio of the largest to the smallest eigenvalues is > 1000.

Options

The options used for this analysis are given in Table II. The number of cross-over events/genome is the expected value of the number of cross-over events that will occur between each set of parents. In this analysis, fewer than one cross over (specifically 0.7) cross over events is expected per set of parents. Typically, about 30% of parent genome sets will not undergo crossover. The mutation rate is the probability of changing any bit in the bit string from a 0 to 1, or 1 to 0. It has been found to be useful to scale fitness in genetic algorithm. In the present application, fitness is scaled to be between 4 and 0.2. This is done by performing linear regression of all fitness values within 2 SD of the, mean, then scaling

the highest and lowest values in this range to be 4 and 0.2, respectively. All values less than the mean −2 SD are assigned a scaled fitness value of 0.2, all values greater than the mean +2 SD are assigned the scaled fitness value of 4. The population size is the number of candidate models to be created for each generation. It has been seen that occasionally NONMEM will generate an objective function value that is unrealistically low, e.g., -1×10^{14}. It is not clear why this occurs, although it tends to occur with fairly large complex models. In spite of the very low value for the objective function, inspection of plots suggests that the model does not fit well at all. The only consistently successful solution found to this is to set a lower limit for acceptable values of the objective function, with any values less than this being considered a crash (with a scaled fitness equal to the lower limit of scaled fitness). Experience also suggests that NONMEM can, at times, become "hung", running for long periods of time without ever converging. To avoid this, a timeout is used, with that model process being terminated, and the scaled fitness value equal to the lower limit of scaled fitness being assigned. Similarly, a timeout is available for a generation, should it be needed. The application of "nicheing" to GA has been discussed. The number of niches can be specified. In addition, the degree of similarity required to be considered within a niche (called the Hamming distance) can be specified. A Hamming distance of 4 means that any models that is differ at 4 or fewer bits will be considered in the same niche.

APPENDIX B—CODE FILE TEMPLATE FOR GENETIC ALGORITHM RUN WITH LABELS

```
$PROB GA COVARIATE SEARCH IVCITAL
$DATA C:\GA\IVCIT\IVCC1G.CSV
$INPUT ID TIME AMT RATE DV WT SEX AGE MDV
$SUBS ADVAN3 TRANS4
$PK
    AAGE = AGE/37.5
    AWT = WT/75.5
    TVCL1=THETA(1) COVSCL(1) COVWCL(1)
    TVCL = TVCL1 COVACL(1)
    CL=TVCL CLERR(1)
    TVV1A = THETA(2) COVSV1(1) COVWV1(1)
    TVV1 = TVV1A COVAV1(1)
    V1 = TVV1 V1ERR(1)
    TVQ1 = THETA(3) COVSQ(1) COVWQ(1)
    TVQ = TVQ1 COVAQ(1)
```

```
    Q = TVQ QERR(1)
    TVV2A = THETA(4) COVSV2(1) COVWV2(1)
    TVV2 = TVV2A COVAV2(1)
    V2 = TVV2 V2ERR(1)
    S1 = V1
$ERROR
    IPRD = F
    IRES = DV-IPRD
    Y = IPRD RESERR(1)
$THETA
    (.5, 1.2) ;CL COEFF
    (0,200) ;V1 COEFF
    (0,30) ;Q COEFF
    (0,550); V2 COEFF
    COVSCL(2)
    COVWCL(2)
    COVACL(2)
    COVSV1(2)
    COVWV1(2)
    COVAV1(2)
    COVSQ(2)
    COVWQ(2)
    COVAQ(2)
    COVSV2(2)
    COVWV2(2)
    COVAV2(2)
$OMEGA
    CLERR(2)
    V1ERR(2)
    QERR(2)
    V2ERR(2)
$SIGMA
    RESERR(2)
$EST MAX= 9999 METH= 1 INTER NOABORT MSFO=IVCTNMG.
MSF $COV
$TABLE ID TIME AMT WT AGE SEX IPRD DV CL Q V1 V2 MDV
    NOPRINT FILE = OUT.DAT ONEHEADER
```

APPENDIX C—CONTROL STREAM OF "BEST" MODEL

```
$PROB GA COVARIATE SEARCH IVCITAL
$DATA c:\rob\data.prn
```

```
$INPUT ID TIME AMT RATE DV WT SEX AGE MDV
$SUBS ADVAN3 TRANS4
$PK
AAGE = AGE/37.5
AWT = WT/75.5
TVCL1=THETA(1) *(AWT)**THETA(5)
     TVCL = TVCL1
     CL=TVCL
     TVV1A = THETA(2)
     TVV1 = TVV1A
     V1 = TVV1 *EXP(ETA(1))
     TVQ1 = THETA(3)
     TVQ = TVQ1
     Q = TVQ *EXP(ETA(2))
     TVV2A = THETA(4) *(AGE)**THETA(7)
     TVV2 = TVV2A
     V2 = TVV2 *EXP(ETA(3))
     S1 = V1
$ERROR
     IPRD = F
     IRES = DV-IPRD
     Y = IPRD *EXP(EPS(1))
$THETA
     (.5, 1.2) ;CL COEFF
     (0,200) ;V1 COEFF
     (0,30) ;Q COEFF
     (0,550); V2 COEFF
     (−4,0.01,4);;THETA(5)=
     (−4,0.01,4);;THETA(6)=
     (0,1,5);;THETA(7)=

$OMEGA BLOCK(3)
0.01
.0001 0.01
.0001 .0001 0.01
$SIGMA
(0.01);;EPS(1)=

$EST MAX=9999 METH=1 INTER NOABORT MSFO=IVCTNMG.
MSF $COV
$TABLE ID TIME AMT WT AGE SEX IPRD DV CL Q V1 V2 MDV
     NOPRINT FILE = OUT.DAT ONEHEADER
```

```
;Genome
;2; 1; 1; 1; 2; 1; 1; 3; 1; 1; 1; 3; 1; 3; 1; 3; 2; 2; 2; 2; 1; 1; 2; 1;
;directory = C:\rob\10.2\4
;timeout = 60
;save result = True
;End of control
```

REFERENCES

1. C. José, Pinheiro, D. M. Bates, and M. J. Lindstrom. *Model building for nonlinear mixed effects models, Proc. of the Biopharmaceutical Section*, Joint Statistical Meetings, 1994.
2. P. O. Maitre, M. Buhrer, D. Thomson, and D. R. Stanski. A three-step approach combining Bayesian regression and NONMEM population analysis: application to midazolam. *J. Pharmacokinet. Biopharm*. **19**(4):377–384 (1991).
3. U. Wahlby, E. N. Jonsson, and M. O. Karlsson. Comparison of stepwise covariate model building strategies in population pharmacokinetic–pharmacodynamic analysis. *Aaps Pharmsci*. **4**(4):E27 (2002).
4. Z. Michalewicz and D. Fogel. *How to Solve it: Modern Heuristics* SpringerVerlag, New York, 1988.
5. C. H. Papadimitriou and K. Steiglitz. *Combinatorial optimization. Algorithms and Complexity*. Dover Publications, Mineola, New York, 1998.
6. J. R. Wade, S. L. Beal, and N. C. Sambol. Interaction between structural, statistical, and covariate models in population pharmacokinetic analysis. *J. Pharmacokinet. Biopharm.* **22**(2):165–177 (1994).
7. K. Kowalski and M. Hutmacher. Efficient screening of covariates in population models using Wald's approximation to the likelihood ratio test. *J. Pharmacokinet. Pharmacodyn.* **28**(3):253–275 (2001).
8. G. Givens and J. Hoeting. *Computational Statistics*. Wiley, New York 2005.
9. D. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Boston, 1989.
10. L. Chambers. *Practical Handbook of Genetic Algorithms—Applications* Vol 1. CRC Press, Boca Raton, 1995.
11. R. Bies, M. E. Sale, and B. E. Pollock. Comparing binary tree and automated machine learning covariate search strategies. PAGE meeting, (2003).
12. J. D. Lohn, G. S. Hornby, and D. S. Linden. An Evolved Antenna for Deployment on NASA's Space Technology 5 Mission, 2004 Genetic Programming Theory Practice Workshop (GPTP-2004).
13. T. K. L. Tong and A. P. C. Chan. Genetic algorithm optimization in building portfolio management. *Construction Manag. Econ.* **19**(6):601–609 (2001).
14. J. Shapcott. Index Tracking: Genetic algorithms for investment portfolio selection. Report EPCC-SS92–24, Edinburgh parallel computing centre, The University of Edinburgh, JCMB 1992.
15. I. S. Oh, J. S. Lee, and B. R. Moon. Hybrid genetic algorithms for feature selection. *IEEE Trans. Pattern Anal. Mach. Intell*. **24**(11):1424–1436 (2004).
16. T. A. Cofolga: A Genetic Algorithm for finding the common folding of RNA. *Comput. Biol. Chem*. (2005).
17. H. N. Lin, K. P. Wu, J. M. Chang, T. Y. Sung, and W. L. Hsu. GANA–a genetic algorithm for NMR backbone resonance assignment. *Nucl. Acids Res*. **33**(14):4593–4601 (2005).
18. P. S. Heckerling, B. S. Gerber, T. G. Tape, and R. S. Wigton. Selection of predictor variables for pneumonia using neural networks and genetic algorithms. *Methods Inf. Med*. **44**(1):89–97 (2005).
19. I. S. Oh, J. S. Lee, and B. R. Moon. Hybrid genetic algorithms for feature selection. *IEEE Trans. Pattern Anal. Mach. Intell*. **26**(11):1424–1437 (2004).

20. N. G. Best, M. K. Cowles, and S. K. Vines. CODA: *Convergence diagnosis and output analysis software for Gibbs sampling output, Version 0.4. MRC Biostatistics Unit*, Cambridge University Press, Cambridge; 1997. http://www.mrc-bsu.cam.ac.uk/bugs/classic/coda04/readme.shtml
21. M. D. Hanover, Globomax. Copyright of the University of California, San Francisco.
22. http://www.gnu.org/software/fortran/fortran.html
23. E. N. Jonsson and M. O. Karlsson. Xpose–an S-PLUS based population pharmacokinetic/pharmacodynamic model building aid for NONMEM. *Comput. Methods Prog. Biomed*. **58**(1):51–64 (1999).
24. J. W. Mandema, D. Verotta, and L. B. Sheiner. Building population pharmacokinetic--pharmacodynamic models. I. Models for covariate effects. *J. Pharmacokinet. Biopharm*. **20**(5):511–528 (1992).
25. H. Akaike. *Information theory and an extension of the maximum likelihood principle*. In *Second International Symposium on Information Theory*, B. Petrox and F. Caski (eds.)
26. Y. Yano, S. L. Beal, and L. B. Sheiner. Evaluating pharmacokinetic/pharmacodynamic models using the posterior predictive check. *J. Pharmacokint. Pharmacodyn*. **28**(2):171–192 (2001).
27. L. B. Sheiner, and S. L. Beal. Some suggestions for measuring predictive performance. *J. Pharmacokinet. Biopharm*. **9**(4):503–512 (1981).
28. F. Mentre and S. Escolano. Prediction discrepancies for the evaluation of nonlinear mixed-effects models. *J. Pharmacokint. Pharmacodyn*. (In press) (2005).