ARTICLE TEMPLATE

An algorithm for nonparametric estimation of a multivariate mixing distribution

ARTICLE HISTORY
Compiled May 22, 2018

ABSTRACT
In this paper we describe a nonparametric maximum likelihood (NPML) algorithm for estimating multivariate mixing distributions. Given N independent observations, convexity theory shows that the NPML estimator is discrete with at most N support points. The original infinite NPML problem then becomes the finite dimensional problem of finding the location and probability of the support points. The probability of the support points is found by a Primal-Dual Interior-Point method; the location of the support points is found by an Adaptive Grid method. Our method is able to handle high-dimensional and complex multivariate mixture models. An important application is discussed for the problem of population pharmacokinetics and a non-trivial example is treated. In addition to population pharmacokinetics, this research also applies to empirical Bayes estimation and many other areas of applied mathematics.

Abbreviations: IPM; NPAG; NPML

KEYWORDS
High dimensional statistics; Nonparametric maximum likelihood; Primal-dual interior-point method; Adaptive grid

WORD COUNT = 5084

1. Introduction

The mixing distribution problem we consider can be stated as follows. Let $\boldsymbol{Y}_1, ..., \boldsymbol{Y}_N$ be a sequence of independent but not necessarily identically distributed random vectors constructed from one or more observations from each of N subjects in the population. Let $\boldsymbol{\theta}_1, ..., \boldsymbol{\theta}_N$ be a sequence of independent and identically distributed random vectors belonging to a compact subset Θ of Euclidean space with common but *unknown* distribution F. The $\{\boldsymbol{\theta}_i\}$ are not observed. It is assumed that the conditional densities $p(\boldsymbol{Y}_i|\boldsymbol{\theta}_i)$ are known, for $i = 1, ..., N$. The mixing distribution of $\boldsymbol{Y}_i$ with respect to F is given by $p(\boldsymbol{Y}_i|F) = \int p(\boldsymbol{Y}_i|\boldsymbol{\theta}_i)dF(\boldsymbol{\theta}_i)$. Because of independence of the $\{\boldsymbol{Y}_i\}$, the mixing distribution of the $\{\boldsymbol{Y}_i\}$ with respect to F is given by

$$L(F) = p(\boldsymbol{Y}_1, ..., \boldsymbol{Y}_N|F) = \prod_{i=1}^{N} \int p\left(\boldsymbol{Y}_i|\boldsymbol{\theta}_i\right) dF\left(\boldsymbol{\theta}_i\right) \tag{1}$$

The mixing distribution problem is to maximize the likelihood function $L(F)$ with respect to all probability distributions F on Θ.

Remark. The distribution F^{ML} that maximizes $L(F)$ is a *consistent* estimator of the true mixing distribution. This was proved originally by Kiefer and Wolfowitz in 1956 [1] . The consistency of F^{ML} is especially important for our application to population pharmacokinetics where F^{ML} is used as a prior distribution for Bayesian dosage regimen design.

The algorithm described in this paper differs from most other published methods in a number of ways. Our algorithm allows for high dimensional Θ. Most published methods require the dimension of Θ to be small and many require the dimension of Θ to be 1, see Section 2. We have treated examples where the dimension of Θ is as high as 29, see Section 5.

Also most published algorithms require the $\{\boldsymbol{Y}_i\}$ to be identically distributed and assume that the conditional densities $\{p(\boldsymbol{Y}_i|\boldsymbol{\theta}_i)\}$ are rather simple, such as $p(\boldsymbol{Y}_i|\boldsymbol{\theta}_i)$ is a multivariate normal density with mean vector and covariance matrix $\boldsymbol{\Sigma}$. Even if $\boldsymbol{\Sigma}$ is unknown and has to be estimated, the structure of this model is straightforward. However, the estimation of $\boldsymbol{\Sigma}$ has to be done carefully to avoid singularities, see Wang and Wang [2]. As will be described in Section 5, we allow $p(\boldsymbol{Y}_i|\boldsymbol{\theta}_i)$ to be calculated from a system of nonlinear ordinary differential-algebraic equations.

We now describe the details of our algorithm. It was proved by Lindsay [3] and Mallet [4] , under simple hypotheses on the conditional densities $\{p(\boldsymbol{Y}_i|\boldsymbol{\theta}_i)\}$, that the global maximizer F^{ML} of $L(F)$ could be represented by a discrete distribution with at most N support points.

This result leads immediately to a finite dimensional optimization problem for F^{ML}, namely to maximize the likelihood function

$$L(\boldsymbol{\lambda},\boldsymbol{\phi}) = \prod_{i=1}^{N}\sum_{k=1}^{K}\lambda_k p\left(\boldsymbol{Y}_i|\boldsymbol{\phi}_k\right) \tag{2}$$

with respect to the support points $\boldsymbol{\phi} = (\boldsymbol{\phi}_1, ..., \boldsymbol{\phi}_K)$ and weights $\boldsymbol{\lambda} = (\lambda_1, ..., \lambda_K)$ such that $\boldsymbol{\phi}_k \in \Theta, \lambda_k \geq 0$ for $k = 1, ..., K$, $K \leq N$ and $\sum_{k=1}^{K}\lambda_k = 1$.

In our algorithm $l(\boldsymbol{\lambda},\boldsymbol{\phi}) = \log L(\boldsymbol{\lambda},\boldsymbol{\phi})$ is maximized, so that

$$l(\boldsymbol{\lambda},\boldsymbol{\phi}) = \sum_{i=1}^{N}\log\sum_{k=1}^{K}\lambda_k p\left(\boldsymbol{Y}_i|\boldsymbol{\phi}_k\right) \tag{3}$$

and the maximization problem becomes

$$\text{maximize } l(\boldsymbol{\lambda},\boldsymbol{\phi}) \tag{4}$$

such that $\boldsymbol{\phi} \in \Theta^K$, $\boldsymbol{\lambda} = (\lambda_1, ..., \lambda_K) \in \mathbb{R}_+^K$, $K \leq N$ and $\sum_{k=1}^{K}\lambda_k = 1$.

Although the maximization problem in Eq. (4) is finite dimensional, it is still high dimensional. The dimension of the maximization problem in Eq. (4) is $N(\dim\Theta) + (N-1)$.

The optimization problem in Eq. (4) is naturally divided into two problems:

Problem 1. Given a set of support points $\{\boldsymbol{\phi}_k\}$, find the optimal weights $\{\lambda_k\}$.

Problem 2. Find the locations of the optimal support points

Problems 1 and 2 are solved cyclically until convergence, i.e. no significant improvement in $l(\boldsymbol{\lambda},\boldsymbol{\phi})$.

Problem 1 is a convex programming problem. In our algorithm, we solve this problem by the Primal-Dual Interior-Point (PDIP) method. This type of method is standard in convex optimization theory, see Boyd and Vandenberghe [5]. However, the exact implementation for a specific problem varies from problem to problem. The exact details of our implementation is described in the Appendix. See also Bell [6], Baek [7] and Yamada et al. [8]. Our PDIP implementation is very fast and can easily handle thousands of variables.

Finding the location of the optimal support points in Problem 2 is a more difficult problem. This location problem is a non-convex global optimization problem with many local extrema and whose dimension is potentialy $N \times \dim \Theta$. The details of our algorithm, called the Adaptive Grid (AG) method, will be described in Section 3 and in Algorithm 1. Roughly speaking, an initial large grid of possible support points is defined in Θ. Problem 1 is solved on this large grid. After PDIP, most of the original grid points are removed due to near-zero weights leaving a smaller high-probability grid. Problem 1 is then solved on this smaller grid. Then the adaptive step takes place. For each remaining grid point, up to $2 \times \dim \Theta$ new (daughter) support points are added. A daughter point outside the search space Θ or too close to a parent point is discarded. The new grid contains the current high-probability points plus the added daughter points. The algorithm is then ready for Problem 1, again. By construction, each iteration increases the value of $l(\boldsymbol{\lambda}, \boldsymbol{\phi})$. This process continues until the function $l(\boldsymbol{\lambda}, \boldsymbol{\phi})$ does not significantly change.

2. Other algorithms

2.1. Comparable Methods

Because of space limitations, in this section we only discuss NPML methods that optimize Eq. 4; methods that treat multivariate distributions; and methods which allow general conditional probabilities $\{P(\boldsymbol{Y}_i, \boldsymbol{\theta}_i)\}$. As explained in this paper, any such NPML algorithm has to address two problems: *locations* of support points and *weights* of support points. NPAG does *locations* by an Adaptive Grid method and *weights* by the Primal-Dual Interior-Point (PDIP) method.

The original methods of Lindsay [3] and Mallett [4] were based on algorithms of optimal design in the style of Fedorov [9]. In Schumitzky [10], an algorithm was proposed which did both *locations* and *weights* by the EM algorithm. It was very stable but also very slow.

In Lesperance and Kalbfleisch [11], a new method was introduced which did *weights* by the dual method described in Section 5 of Lindsay [3] and *locations* by what they called the Intra-Simplex Direction Method (ISDM). Even though, the Lesperance and Kalbfleisch paper was restricted to univariate distributions, the ISDM method has been generalized to the multivariate case. To briefly describe ISDM, let $D(\boldsymbol{\theta}, F)$ be the directional derivative of $\log L(F)$ in the direction of the Dirac distribution $\delta_{\boldsymbol{\theta}}$ supported at $\boldsymbol{\theta} \in \Theta$. (This function is defined in Section 4 below.) ISDM is an iterative algorithm. At stage k, let F^k be the current estimate F^{ML}. Then find all the local maxima of $D(\boldsymbol{\theta}, F^k)$. These local maxima are added to the current set of support points and a new F^{k+1} is calculated. If there are no new local maxima, then the algorithm is done.

In Pilla, Bartolucci, and Lindsay [12], another new method was developed where the *locations* were found by an initial fine grid. But the *weights* were found by a dual version of the PDIP method.

In Savic, Kjellsson, and Karlsson [13], a nonparametric method was added to the popular NONMEM program. NONMEM-NP is a hybrid parametric-nonparametric approach The *locations* of support points were found by a parametric maximum likelihood algorithm. Then the *weights* were found by maximizing Eq. (4) relative to the newly found support points. NONMEM-NP can handle high dimensional and complex multivariate distributions. An extension to NONMEM-NP was developed in Savic and Karlsson [14] where additional support points are added to the original set. A comparison between NONMEM-NP and NPAG is discussed in Leary [15].

In Wang and Wang [2] , a new algorithm was developed for multivariate distributions. The *locations* were found by a combination of EM and a variant of ISDM. The *weights* were found by a family of Quadratic Programs. In [2], examples are done for 8 and 13 dimensional mutivariate mixing distributions.

Note: The Quadratic Programming algorithm (QP) of Wang and Wang [2] has a very attractive feature. For a prescribed set of support points, QP finds the zero probabilities exactly. Thus QP avoids the Grid Condensation step where support points from PDIP with sufficiently low probabilities are deleted. However, QP and PDIP are based on different numerical methods and a comparison of the efficiency of both algorithms has not been determined.

The algorithms which have shown by published examples to handle the highest dimensional multivariate problems are NONMEM NP, Wang and Wang [2], and NPAG.

2.2. Benders Decomposition

For any set of grid points $\boldsymbol{\phi} = (\boldsymbol{\phi}_1, ..., \boldsymbol{\phi}_m)$ in Θ^m, let $\boldsymbol{\lambda} = \hat{\boldsymbol{\lambda}}(\boldsymbol{\phi})$ be the corresponding set of optimal weights given by the PDIP method. Then the function $F(\boldsymbol{\phi}) = l(\hat{\boldsymbol{\lambda}}(\boldsymbol{\phi}), \boldsymbol{\phi})$ depends only on $\boldsymbol{\phi}$ and can be maximized directly. For optimization methods, this technique is called Benders Decomposition. The NPAG algorithm maximizes $F(\boldsymbol{\phi})$ by an adaptive search method. In a method proposed by James Burke, $F(\boldsymbol{\phi})$ is maximized by a Newton type method. Since the function $F(\boldsymbol{\phi})$ is not necessarily differentiable, a relaxed Newton method must be used similar to what is described in the Appendix for the Primal-Dual Algorithm. For details of Benders Decomposition as applied to our problem, see Bell [6], Baek [7] and Jordan-Squire [16].

3. Adaptive Grid Method

3.1. NPAG Implementation (NPAG - Algorithm 1)

NPAG is a Fortran program consisting of a number of subroutines as described below. The main program performs the Adaptive Grid (AG) method (consisting of expansion and compression algorithms) and calls the Primal-Dual Interior-Point (PDIP) subprogram. The PDIP algorithm solves the maximization problem of Eq. (4) for a fixed grid and is described precisely in the Appendix.

For the purpose of this discussion, we can think of PDIP as a function $\hat{\boldsymbol{\lambda}}$ from Θ^m into the set $S^m = \{\boldsymbol{\lambda} \in \mathbb{R}_+^m : \sum_{k=1}^m \lambda_k = 1\}$ defined as follows: If $\boldsymbol{\phi} = (\boldsymbol{\phi}_1, ..., \boldsymbol{\phi}_m)$ then $\hat{\boldsymbol{\lambda}}(\boldsymbol{\phi}) = (\hat{\lambda}_1, ..., \hat{\lambda}_m)$ maximizes Eq. (4) relative to the fixed set of grid points $(\boldsymbol{\phi}_1, ..., \boldsymbol{\phi}_m)$. In this case we write $G = (\boldsymbol{\phi}, \hat{\boldsymbol{\lambda}}(\boldsymbol{\phi}))$ and $l(G) = l(\boldsymbol{\phi}, \hat{\boldsymbol{\lambda}}(\boldsymbol{\phi}))$.

In NPAG there are two types of grids: expanded and condensed. The expanded grids are the initial grid and the grids after Grid Expansion (Algorithm 2). The condensed grids are generated by Grid Condensation (Algorithm 3). Each cycle of NPAG begins with an expanded grid. The likelihood calculation is done on the condensed grids.

Now for the Adaptive Grid method. Assume that Θ is a bounded Q-dimensional hyper-rectangle. Initially we let $\boldsymbol{\phi}^0_{expanded} = (\boldsymbol{\phi}^0_1, ..., \boldsymbol{\phi}^0_M)$ be the set of M Faure grid points in Θ, see [17–19]. Alternatively, we could initially let $\boldsymbol{\phi}^0_{expanded}$ be generated by a uniform distribution on Θ or by a prior run of the program.

Remark. The Faure grid points for a hyper-rectangle Θ are a low-discrepancy set which in some sense optimally and uniformly covers Θ. In our implementation of NPAG, the Faure point sets come in discrete sizes which nest with each other. (Allowable number of points equals 2129, 5003, 10007, 20011, 40009, 80021, and multiples of 80021.) This nesting property is useful for checking the optimality of F^{ML}, see Section 4. We have found that replacing the initial Faure set by a set generated by a uniform distribution on Θ increases the time to convergence but results in the same optimal distribution.

Now set $G^0_{expanded} = (\boldsymbol{\phi}^0, \hat{\boldsymbol{\lambda}}(\boldsymbol{\phi}^0))$. Our approach is to generate a sequence of solutions G^n to Eq. (4) of increasingly greater likelihood, where unless otherwise specified, G^n refers to the condensed grid at the n^{th} cycle of the algorithm. If G^n has log likelihood negligibly different than G^{n-1}, then G^n is considered the optimal solution to Eq. (4) and is relabeled F^{ML}. If not, then the process continues using the $\boldsymbol{\phi}^n$ as the new seed. This loop is repeated until F^{ML} is found.

The stopping conditions for NPAG are defined precisely in Algorithm 1. If the stopping conditions are not met prior to a set maximum number of iterations, the program will exit after writing the last calculated G^n into a file.

3.2. Grid Expansion (EXPAND - Algorithm 2)

The crux of the Adaptive Grid method is how to go from G^0 to G^1 or more generally, from G^n to G^{n+1}. The details of doing this are now explained roughly below and precisely in Algorithm 1.

Let Q be the dimension of Θ. Suppose at stage n we have a grid of high-probability support points $\boldsymbol{\phi}^n$. We then add $2Q$ daughter points for each support point $\boldsymbol{\phi}_k \in \boldsymbol{\phi}^n$. The daughter points are the vertices of a small hyper-rectangle centered at each $\boldsymbol{\phi}_k$ with size proportional to the original size of the hyper-rectangle defining Θ. The size of this small hyper rectangle decreases as the accuracy of the estimates increases. (See Algorithm 2.)

Let $\boldsymbol{\phi}^{n+1}_{expanded} = \boldsymbol{\phi}^n \cup$ Daughter-Points. Then the PDIP subprogram is applied to $\boldsymbol{\phi}^{n+1}_{expanded}$ resulting in the new solution set $G^{n+1}_{expanded} = (\boldsymbol{\phi}^{n+1}_{expanded}, \hat{\boldsymbol{\lambda}}(\boldsymbol{\phi}^{n+1}_{expanded}))$; see Algorithm 1. The solution set $G^{n+1}_{expanded}$ is now ready for grid condensation.

3.3. Grid Condensation (CONDENSE - Algorithm 3)

The above solution set $G^{n+1}_{expanded}$ may have many support points with very low probability. We remove all support points which have corresponding probability less than $(\max \boldsymbol{\lambda})\Delta_{\boldsymbol{\lambda}}$, where $\boldsymbol{\lambda}$ is the vector of current probabilities and the default for $\Delta_{\boldsymbol{\lambda}}$ is 10^{-3}. (Note that at this point the remaining probabilities are not normalized.) The probabilities of the remaining support points are normalized by a second call to the PDIP subprogram. This second call to PDIP is very fast. The likelihood associated with these remaining support points and normalized probabilities is then used to update the program control parameters and check for convergence (Algorithm 1 and

Section 3.5). If convergence is attained, then the output of this second call to PDIP provides the support points and probabilities of the final solution. If convergence is not attained, then the remaining support points are sent to the Grid Expansion subprogram (Algorithm 2), initializing the next cycle.

At the end of the program, the output of this second call to PDIP provides the location and weights of the final solution.

3.4. PDIP Subprogram - See Appendix A

The PDIP subprogram finds the optimal solution to Eq. 4 with respect to $\boldsymbol{\lambda}$ for fixed $\boldsymbol{\phi}$. PDIP employs a primal-dual interior-point method that uses a relaxed Newton method to solve the corresponding Karush-Kuhn-Tucker equations. (See Eqs. 14 -17 of Appendix A.)

For any $\boldsymbol{Y}=(\boldsymbol{Y}_1, ..., \boldsymbol{Y}_N)$ and any $\boldsymbol{\phi}=(\boldsymbol{\phi}_1, ..., \boldsymbol{\phi}_K) \in \Theta^K$, the input to the PDIP subprogram is the $N \times K$ matrix $\{p(\boldsymbol{Y}_i|\boldsymbol{\phi}_k)\}$. The output consists of the optimal weights $\hat{\boldsymbol{\lambda}}(\boldsymbol{\phi})$ and the corresponding log-likelihood $l(\hat{\boldsymbol{\lambda}}(\boldsymbol{\phi}), \boldsymbol{\phi})$. An in-depth description of the PDIP algorithm and its implementation is presented in Appendix A. See also [6–8].

3.5. NPAG Stopping Conditions

Algorithm

As explained above, a *potential* solution to F^{ML} is not accepted as a global optimum until successive sequences of G^n produce final distributions evaluating to sufficiently close log likelihood. The various upper and lower bounds Δ for NPAG control and stopping conditions are defined below and are used in Algorithms 1, 2, and 3.

Δ_L Primary upper bound on the allowable difference between two successive estimated Log-Likelihoods; the default initialization is 10^{-4}.

Δ_F Secondary upper bound on the allowable difference between two successive estimated Log-Likelihoods of *potential* F^{ML}; the default initialization is 10^{-2}.

Δ_e Sets an upper bound on the accuracy variable *eps* of Algorithm 1. The default initialization for Δ_e is 10^{-4}. The default initialization for *eps* is 0.2 and is stepped down until $eps \leq \Delta_e$

Δ_F and Δ_e define the two stopping conditions for Algorithm 1.

Δ_D Sets a lower bound on how close two support points can get; the default initialization is 10^{-4}.

Δ_λ Sets a lower bound factor on the probabilities of the weights λ; the default initialization is 10^{-3}.

3.6. Calculation of $p(\boldsymbol{Y_i}|\boldsymbol{\phi_k})$

Given observations $\boldsymbol{Y}_i$, $i = 1, ..., N$ and grid points $\boldsymbol{\phi}_k$, $k = 1, ..., K$, the PDIP subprogram only depends on the $N \times K$ matrix $\{p(\boldsymbol{Y}_i|\boldsymbol{\phi}_k)\}$. NPAG can be used for any problem once this matrix is defined. However, the default setting of NPAG is for the problem of population pharmacokinetics. For a good background of population pharmacokinetics see Davidian and Giltinan [20, 21].

In population pharmacokinetics, generally $\boldsymbol{Y}_i = (\boldsymbol{y}_{i,1}, ..., \boldsymbol{y}_{i,M})$ is a matrix of vector observations for the i-th subject. Since NPAG allows multiple outputs, each $\boldsymbol{y}_{i,m}$ is itself a q-dimensional vector $\boldsymbol{y}_{i,m}$ $=(y_{i,m,1},\cdots,y_{i,m,q})$. The observations $y_{i,m,j}$, are then typically given by a regression equation of the form:

$$\begin{aligned} &y_{i,m,j} = f_{i,m,j}(\boldsymbol{\theta}_i) + \nu_{i,m,j},\ j = 1,\cdots,q \\ &\nu_{i,m,j} \sim N(0,(\sigma_{i,m,j}(\boldsymbol{\theta}_i))^2) \\ &\quad \boldsymbol{\theta}_i \text{ are unobserved parameters specific for } \boldsymbol{Y}_i \end{aligned} \tag{5}$$

In the above Eq. 5, $f_{i,m,j}$ is a known nonlinear function depending on the model structure, the dosage regimen, the sampling schedule, all covariates and of course the subject-specific parameter vector $\boldsymbol{\theta}_i$. Except for simple models, $f_{i,m,j}$ requires the solution of (possibly nonlinear) ordinary differential equations.

In the current implementation of NPAG, it is assumed that the $(\boldsymbol{y}_{i,1},...,\boldsymbol{y}_{i,M})$ are independent. Then

$$p(\boldsymbol{Y}_i|\boldsymbol{\phi}_k) = \frac{\exp\left(-\frac{1}{2}\sum_{m=1}^{M}(\boldsymbol{y}_{i,m} - \boldsymbol{f}_{i,m}(\boldsymbol{\phi}_k))\boldsymbol{\Sigma}_{i,m}^{-1}(\boldsymbol{\phi}_k)(\boldsymbol{y}_{i,m} - \boldsymbol{f}_{i,m}(\boldsymbol{\phi}_k))^T\right)}{\prod_{m=1}^{M}\sqrt{(2\pi)^q \det \boldsymbol{\Sigma}_{i,m}(\boldsymbol{\phi}_k)}} \tag{6}$$

where $\boldsymbol{f}_{i,m} = (f_{i,m,1},...,f_{i,m,q})$ and $\boldsymbol{\Sigma}_{i,m} = diag(\sigma_{i,m,1}^2,...,\sigma_{i,m,q}^2)$. For the purposes of matrix multiplication in Eq. 6 ,we think of $\boldsymbol{y}_{i,m}$ and $\boldsymbol{f}_{i,m}$ as q-dimensional row vectors.

To complete the description of Eq. 6 we need to model the standard deviation terms $\sigma_{i,m,j}$ of the assay noise. In our implementation of NPAG, four different models are allowed. Let

$$\alpha_{i,m,j}(\boldsymbol{\phi}_k) = c_0 + c_1 f_{i,m,j}(\boldsymbol{\phi}_k) + c_2 f_{i,m,j}^2(\boldsymbol{\phi}_k) + c_3 f_{i,m,j}^3(\boldsymbol{\phi}_k) \tag{7}$$

and set

$$\sigma_{i,m,j} = \begin{cases} \alpha_{i,m,j} & \text{assay error polynomial only} \\ \gamma\alpha_{i,m,j} & \text{multiplicative error} \\ \sqrt{\alpha_{i,m,j}^2 + \gamma^2} & \text{additive error} \\ \gamma & \text{constant level of error} \end{cases} \tag{8}$$

The parameter γ in Eq. 8 is a variance factor. Artificially increasing the variance during the first several cycles of NPAG increases the likelihood for each $\boldsymbol{\phi}$, allowing the algorithm to use these cycles to find a better initial state from which to begin optimization. NPAG also has an option to "optimize" γ. This changes NPAG from a nonparametric method to a "semiparametric" method and will not be discussed here. The interested reader can consult [8].

Next if $c_0 = 0$ in Eq. 7, then $\alpha_{i,m,j}$ can become very small for certain values of $\boldsymbol{\phi}$ that in early iterations can be far from optimal. This in turn causes numerical problems as the likelihood is infinite if $\sigma_{i,m,j} = 0$. One way to avoid this problem is to take $\sigma_{i,m,j} = constant$. Another way would be to assume that $\alpha_{i,m,j}$ is *known* and is given by

$$\alpha_{i,m,j} = c_0 + c_1 y_{i,m,j} + c_2 y_{i,m,j}^2 + c_3 y_{i,m,j}^3 \tag{9}$$

That is, to approximate σ by using a polynomial of the observed values rather than model predicted values. In our experience with NPAG, the approximation of Eq. 9 is useful for ensuring computational stability (especially during the early cycles of the algorithm). However, from a theoretical perspective, this change violates the conditions of maximum likelihood and will not be discussed here. Again the interested reader can consult [8].

4. Convergence

For a given initial grid $\boldsymbol{\phi}^0$, the NPAG algorithm is only guaranteed to find a local maximum of $L(F)$. More precisely, if $\boldsymbol{\phi}^*$ is the final grid of NPAG starting from $\boldsymbol{\phi}^0$, then $\hat{\boldsymbol{\lambda}}(\boldsymbol{\phi}^*)$ is a global maximum on $\boldsymbol{\phi}^*$ but the support points $\boldsymbol{\phi}^*$ may be only a local maximum.

Global convergence of a nonparameteric maximum likelihood method for estimation of a multivariate mixing distribution is very difficult. For one-dimensional distributions the problem is straightforward. The idea of proof goes back to at least Fedorov [9] in 1972, which involves the use of *Directional Derivatives.*

Let F be any distribution on Θ. Then the directional derivative of $\log L(F)$ in the direction of the Dirac distribution $\delta_{\boldsymbol{\theta}}$ supported at $\boldsymbol{\theta}$ is defined by

$D(\boldsymbol{\theta}, F)=[\sum_{i=1}^{N} P(\boldsymbol{Y}_i|\boldsymbol{\theta})/P(\boldsymbol{Y}_i|F)] - N$, $\boldsymbol{\theta} \in \Theta$, where $p(\boldsymbol{Y}_i|F) = \int p(\boldsymbol{Y}_i|\boldsymbol{\theta})dF(\boldsymbol{\theta})$. Let F_k be the current NPML estimate at iteration k. The Fedorov method involves maximizing $D(\boldsymbol{\theta}, F_k)$ for $\boldsymbol{\theta} \in \Theta$, at every iteration. Then the point at which the maximum occurs is added in an optimal way to F_k to give F_{k+1}. Under the assumptions of regularity, Fedorov shows that $L(F_k)$ converges to $L(F^{ML})$, see Fedorov [9], (Theorem 2.5.3). Many improvements to this method have been made. In Lesperance and Kalbfleisch [11] and Wang and Wang [2], instead of just adding the point at which $D(\boldsymbol{\theta}, F_k)$ occurs, all the points where local maxima occur are added in an optimal way. Again under the assumptions of regularity, convergence as above is proved. In one-dimension these methods are very efficient. In higher dimensions, these methods are not computationally practical.

We now suggest a method to check whether the final distribution of NPAG is globally optimal and if not optimal, how close it is to the optimal. It also involves the use of the directional derivative $D(\boldsymbol{\theta}, F)$, but only at the last iteration of NPAG. Now define

$D(F) = \max_{\boldsymbol{\theta} \in \Theta} D(\boldsymbol{\theta}, F)$

Note that the *max* in the above expression is only over Θ and not over Θ^N. It is proved in Lindsay [3] that F^* is a global maximum of $L(F)$, i.e. $F^* = F^{ML}$, if and only if $D(F^*) = 0$

Even if $D(F^*) \neq 0$, it is useful to make this computation as it is also proved in Lindsay [3] that

$L(F^{ML}) - L(F^*) \leq D(F^*)$,

so this last expression gives an estimate of the accuracy of the final NPAG result.

Now even though we said above it is not practical to calculate $D(F)$ at every iteration of an algorithm, we are just suggesting to make this calculation at the end of the algorithm. This calculation can be performed by a deterministic or stochastic optimization algorithm.

5. Examples

First of all, the NPAG program has been used successfully in high-dimensional and very complex pharmacokinetic-pharmacodynamic models. In Ramos-Martin et al. [22], the NPAG program was used for a population model of the pharmacodynamics of vancomycin for CoNS infection in neonates. (Vancomycin is an antibiotic used to treat a number of serious bacterial infections. Coagulase-negative staphylococci (CoNS) are the most commonly isolated pathogens in the neonatal intensive care unit.) This model had 7 nonlinear differential equations and 11 random parameters. The population was a combination of 300 experimental and animal subjects. In Drusano et al. [23], the NPAG program was used for a population model of two drugs for the treatment of tuberculosis. This model had 5 nonlinear differential equations, 3 nonlinear algebraic equations, 1671 observations from 6 outputs and 29 random parameters. In the algebraic equations, the state variables were only defined implicitly and had to be solved for by an iterative method.

The above two examples are too complex to use for simulation purposes. Consequently we present here a simpler model which has an analytic solution and which can be checked by other algorithms. Nevertheless, the estimation of parameters in this model is not trivial. We consider a three-compartment PK model with a continuous IV infusion into the central compartment and a bolus input into the absorption compartment. The individual subject model is described by the following differential equations:

$$\begin{aligned}
\frac{\mathrm{d}x_1}{\mathrm{d}t} &= -K_a x_1, \quad x_1(t) = \begin{cases} 0 & \text{for } 0 \le t < 5 \\ b & \text{if } t = 5 \end{cases} \\
\frac{\mathrm{d}x_2}{\mathrm{d}t} &= K_a x_1 - (K_{el} + K_{cp})\, x_2 + K_{pc} x_3 + r(t), \quad x_2(0) = 0 \\
\frac{\mathrm{d}x_3}{\mathrm{d}t} &= K_{cp} x_2 - K_{pc} x_3, \quad x_3(0) = 0
\end{aligned}$$

and output equation
$y_1(t) = x_2(t)/V_c + w(t), w(t) \sim N(0, \sigma^2),\ \sigma = 5.5$
The inputs are a bolus $b = 2000$ at $t = 5$ and a continuous infusion $r(t) = 500$, for $t \ge 0$. This model has 5 random parameters (V, K_a, K_{el}, K_{cp}, K_{pc}). A diagram of this model is given in Figure A1. It is known that this model is structurally identifiable, see Godfrey [24]. However, we have found that for a continuous IV infusion, the parameters K_{cp} and K_{pc} are very difficult to estimate in a noisy environment.

The details of the simulation are as follows. There were 300 simulated subjects. The random variables (V, K_a, K_{cp}, K_{pc}) were independently simulated from normal distributions with means respectively equal to (1.2, 0.8, 0.2, 2.0) and standard deviations equal to 25% coefficient of variation.

The random variable K_{el} was independently simulated from a bimodal mixture of two normal distributions with means respectively equal to 0.5 and 1.5, with standard deviations equal to 10% coefficient of variation, and with weights equal to 0.2 and 0.8. This distribution would apply to an elimination rate constant with a bimodal distribution where 80% of the subjects have a mean of 1.5, and only 20% have a mean of 0.5. The power of the nonparametric method allows the detection of the 20% group.

Twelve observations were taken at times
$t = 1.1,\ 5.4,\ 6.1,\ 6.5,\ 6.7,\ 7.8,\ 8.4,\ 9.2,\ 13.5,\ 15.3,\ 15.5,\ 15.8.$

These sampling times were chosen in an ad hoc fashion and are not to be considered optimal. In Figure A2 we show the profiles of the 300 noisy model outputs y_1. These profiles are plotted as piecewise linear functions with nodes at the observation times.

The initial Faure set had $80,321$ support points. After the first iteration of the NPAG algorithm, the number of support points was down to 300, where it essentially stayed for the rest of the algorithm. After 100 iterations NPAG was stopped based on the convergence criteria of Section 3.5.

The simulated and estimated marginal distributions are shown in Figures A3 and A4. It is seen that the estimated marginal distributions were quite accurate. when compared to the simulated histograms. In particular the bimodal shape of K_{el} was uncovered.

NPAG is designed to estimate the whole joint distribution of the parameters. As mentioned earlier, the estimate F^{ML} is especially important for our application to population pharmacokinetics where F^{ML} is used as a prior distribution for Bayesian dosage regimen design. However, F^{ML} is a consistent estimator of the true mixing distribution and consequently, the moments of F^{ML} should be consitent estimators of the true moments. Means and variances of parameter estimates for F^{ML} can be easily obtained by integrating the corresponding marginal distributions. So as a check of this fact, in Table 1, the comparisons of estimated versus simulated means and variances are shown. Again, results are quite accurate, see Table A1.

Finally, in Figure A5 we include a graph of Predicted versus Observed values which shows the all around good fit of the data. The predicted values are gotten as follows: For each subject, the Bayesian mean estimate of the parameters are found using the final NPAG distribution as a prior and that subject's observations. Then based on these parameter means, the subject's concentration profile is calculated.

6. Final Remarks and Conclusions

6.1. Final Remarks

The NPAG program was developed at the USC Laboratory of Applied Pharmacokinetics. James Burke (University of Washington) developed the Primal-Dual Interior-Point method discussed in the Appendix. Robert Leary (Pharsight Corporation) developed the Adaptive Grid method and wrote the original Fortran program for NPAG. Michael Neely, MD (USC Children's Hospital of Los Angeles) developed the program package Pmetrics which contains NPAG as a subprogram. Pmetrics is an R package for nonparametric and parametric population modeling and simulation and is available at `www.lapk.org`, see Neely et al. [25].

6.2. Conclusions

We have desribed a nonparametric maximum likelihood method called NPAG for estimating multivariate mixing distributions. NPAG is based on an iterative algorithm employing the Primal-Dual Interior-Point method and an Adaptive Grid method. Our method is able to handle high-dimensional and complex mixture models. Other methods are discussed. A detailed description of NPAG is given. The important application to population pharmacokinetics is described and a non-trivial example is given.

In addition to population pharmacokinetics, this research also applies to empirical Bayes estimation, see Koenker and Mizera [26] and to many other areas of applied mathematics, see Banks et al. [27].

Appendix A. A Primal-Dual Interior-Point Algorithm (PDIP)

To make this paper self-contained, we outline here the PDIP algorithm which was written by James Burke. This algorithm is a FORTRAN subroutine of NPAG. The description below is based on the Matlab and C++ codes found in Bradley Bell's website, see [6]. Definition of general terms and theorems can be found in Boyd and Vandenberghe [5].

A.1. Duality Theory and the Basic Problem

Given a set of support points $\{\phi_k\}$, the problem of finding the optimal weights $\{\lambda_k\}$ in Eq. 4 can be posed as the following optimization problem

$$\mathcal{P} \quad \min \Phi\left(\boldsymbol{\Psi}\boldsymbol{\lambda}\right) \text{ s.t. } 0 \leq \boldsymbol{\lambda},\ \boldsymbol{e}^{\mathsf{T}}\boldsymbol{\lambda} = 1,$$

where $\boldsymbol{\Psi} \in \mathbb{R}^{n\times m}$ is the matrix whose (i,j) entry is $p(y_i \mid \phi_j)$ and where in general, the function $\Phi : \mathbb{R}^k \mapsto \mathbb{R} \cup \{+\infty\}$ is given by

$$\Phi(z) = \begin{cases} -\sum_{i=1}^{k} \log z_i & ,0 < z, \text{ and} \\ +\infty & ,\text{otherwise.} \end{cases} \tag{A1}$$

The symbol $\boldsymbol{e}$ is always to be interpreted as the vector of all ones of the appropriate dimension.

The problem $\mathcal{P}$ is a convex programming problem since the objective function Φ is convex and the constraining region is a convex set. The Fenchel-Rockafellar dual of the convex program $\mathcal{P}$ is the problem

$$\mathcal{D} \quad \min \Phi(\boldsymbol{\omega}) \quad \text{s.t. } \boldsymbol{L}^{\mathsf{T}}\boldsymbol{\omega} \leq m\boldsymbol{e}.$$

From Boyd we obtain the following Karush-Kuhn-Tucker (KKT) equations relating the solutions to the problem $\mathcal{P}$ and $\mathcal{D}$.

$$m\boldsymbol{e} = \boldsymbol{\Psi}^{\mathsf{T}}\boldsymbol{w} + \boldsymbol{y} \tag{A2}$$

$$\boldsymbol{e} = \boldsymbol{W}\boldsymbol{\Psi}\boldsymbol{\lambda} \tag{A3}$$

$$0 = \boldsymbol{\Lambda}\boldsymbol{Y}\boldsymbol{e} \tag{A4}$$

where for any vector $\boldsymbol{x}$, we define $\boldsymbol{X}$ to be the diagonal matrix having $\boldsymbol{x}$ along the diagonal.

A.2. An Interior-Point Path-Following Algorithm

The relaxed KKT is given by

$$m\boldsymbol{e} = \boldsymbol{\Psi}^{\mathsf{T}}\boldsymbol{w} + \boldsymbol{y} \tag{A5}$$

$$\boldsymbol{e} = \boldsymbol{W}\boldsymbol{\Psi}\boldsymbol{\lambda} \tag{A6}$$

$$\mu\boldsymbol{e} = \boldsymbol{\Lambda}\boldsymbol{Y}\boldsymbol{e} \tag{A7}$$

$$0 \le \boldsymbol{\lambda},\ 0 \le \boldsymbol{w},\ 0 \le \boldsymbol{y}, \tag{A8}$$

for $\mu > 0$. (μ is the relaxation parameter.) A damped Newton's method is used to solve the above system.

Consider the function $F:\ \mathbb{R}^{2m+n} \mapsto \mathbb{R}^{2m+n}$ given by

$$F(\boldsymbol{\lambda}, \boldsymbol{w}, \boldsymbol{y}) = \begin{bmatrix} \boldsymbol{\Psi}^{\mathsf{T}}\boldsymbol{w} + \boldsymbol{y} \\ \boldsymbol{W}\boldsymbol{\Psi}\boldsymbol{\lambda} \\ \boldsymbol{\Lambda}\boldsymbol{Y}\boldsymbol{e} \end{bmatrix}.$$

A triple $(\boldsymbol{\lambda}, \boldsymbol{w}, \boldsymbol{y})$ solves Eqs. A.A5 to A.A8 if and only if

$$F\left(\boldsymbol{\lambda}, \boldsymbol{w}, \boldsymbol{y}\right) = \begin{pmatrix} m\boldsymbol{e} \\ \boldsymbol{e} \\ \mu\boldsymbol{e} \end{pmatrix} \tag{A9}$$

and $0 \le \boldsymbol{\lambda}$, $0 \le \boldsymbol{\omega}$, and $0 \le \boldsymbol{y}$. Path-following algorithms attempt to solve A9 by applying Newton's method for progressively smaller values of the relaxation parameter μ. We first need the derivative of F. It follows

$$F'(\boldsymbol{\lambda}, \boldsymbol{\omega}, \boldsymbol{y}) = \begin{bmatrix} 0 & \boldsymbol{\Psi}^{\mathsf{T}} & I \\ \boldsymbol{W}\boldsymbol{\Psi} & \boldsymbol{Z} & 0 \\ \boldsymbol{Y} & 0 & \boldsymbol{\Lambda} \end{bmatrix}$$

where $\boldsymbol{z} = \boldsymbol{\Psi}\boldsymbol{\lambda}$.

At the kth iteration of the algorithm, the Newton step is given by the solution to the nonsingular linear system

$$F\left(\boldsymbol{\lambda}^k, \boldsymbol{w}^k, \boldsymbol{y}^k\right) + F'\left(\boldsymbol{\lambda}^k, \boldsymbol{w}^k, \boldsymbol{y}^k\right) * \left[\boldsymbol{\Lambda}^k, \boldsymbol{W}^k, \boldsymbol{Y}^k\right]^{\mathsf{T}} = \left[\boldsymbol{e}_m, \boldsymbol{e}_n, \mu^k\boldsymbol{e}_m\right]^{\mathsf{T}} \tag{A10}$$

where $\boldsymbol{y}$ is constrained to satisfy the first KKT condition $\boldsymbol{y}^k = \boldsymbol{e}_m - \boldsymbol{\Psi}^{\mathsf{T}}\boldsymbol{w}^k$.

The above set of equations can be reduced by standard techniques. It follows:

$$\Delta\boldsymbol{w} = \boldsymbol{H}^{-1}\boldsymbol{r}_2 \tag{A11}$$

$$\Delta\boldsymbol{y} = -\boldsymbol{\Psi}\Delta\boldsymbol{\omega} \tag{A12}$$

$$\Delta\boldsymbol{\lambda} = \boldsymbol{r}_1 - \boldsymbol{\lambda} - \boldsymbol{D}_1\Delta\boldsymbol{y} \tag{A13}$$

where $\boldsymbol{H} = \boldsymbol{D}_2 - \boldsymbol{\Psi}\boldsymbol{D}_1\boldsymbol{\Psi}^{\mathsf{T}}$, $\boldsymbol{D}_2 = \boldsymbol{Z}\boldsymbol{W}^{-1}$, $\boldsymbol{D}_1 = \boldsymbol{\Lambda}\boldsymbol{Y}^{-1}$, $\boldsymbol{r}_1 = \mu\boldsymbol{Y}^{-1}\boldsymbol{e}$, $\boldsymbol{r}_2 = \boldsymbol{W}^{-1}\boldsymbol{e} - \boldsymbol{\Psi}\boldsymbol{r}_1$ where the superscript k is suppressed for simplicity.

A.3. The Algorithm

To describe the algorithm we need to define the variables:
$q = \frac{1}{m}\sum_{i=1}^{m} \lambda_i y_i$
$\rho = \|\boldsymbol{e} - \boldsymbol{W}\boldsymbol{Z}\boldsymbol{e}\|_\infty$
and the scaled duality gap
$\gamma = \frac{|\Phi(\boldsymbol{\omega})+\Phi(\boldsymbol{\Psi}\boldsymbol{\lambda})|}{1+|\Phi(\boldsymbol{\Psi}\boldsymbol{\lambda})|}$.

(*Initialization*)

Initially choose $\boldsymbol{\lambda}^0 = \boldsymbol{e}_m/m$, $\boldsymbol{w}^0 = \boldsymbol{e}_n/\boldsymbol{\Psi}\boldsymbol{\lambda}^0$, and $\boldsymbol{y}^0 = \boldsymbol{e}_m - \boldsymbol{\Psi}^\intercal\boldsymbol{w}^0$. (Division of two vectors is performed component-wise.) Set $\varepsilon = 10^{-8}$.

(*Iteration*)

At iteration $k+1$, set
$\mu^{k+1} = \sigma^k q^k$
where the reduction factor σ is defined by

$$\sigma = \begin{cases} 1 & \text{, if } \mu \le \varepsilon \text{ and } \rho > \varepsilon, \\ \min(0.3, (1-\delta_1)^2), (1-\delta_2)^2, \frac{|\rho-\mu|}{\rho+100\tau} & \text{, otherwise.} \end{cases}$$

The next iterates are given by $\boldsymbol{\lambda}^{k+1} = \boldsymbol{\lambda}^k + \delta_1[\Delta\boldsymbol{\lambda}^k]$, $\boldsymbol{\omega}^{k+1} = \boldsymbol{\omega}^k + \delta_2[\Delta\boldsymbol{\omega}^k]$ and $\boldsymbol{y}^{k+1} = \boldsymbol{y}^k + \delta_2[\Delta\boldsymbol{y}^k]$, where the “damping” factors δ_1 and δ_2 are defined by

$$\begin{aligned} \delta_{1,0} &= -\left[\min(\min(\boldsymbol{\Lambda}^{-1}\Delta\boldsymbol{\lambda}), -\frac{1}{2})\right]^{-1} \\ \delta_{2,0} &= -\left[\min(\min(\boldsymbol{Y}^{-1}\Delta\boldsymbol{y}), \min(\boldsymbol{W}^{-1}\Delta\boldsymbol{w}), -\frac{1}{2})\right]^{-1} \\ \delta_1 &= \min(1, 0.99995\delta_{1,0}) \\ \delta_2 &= \min(1, 0.99995\delta_{2,0}) \end{aligned}$$

(*Exit Conditions*)

Iterate Eqs. A.11-A-13 until
$\mu \le \varepsilon$ and $\rho \le \varepsilon$ and $\gamma \le \varepsilon$.
If these conditions are not satisfied after a set number of iterations, then write “PDIP did not converge in the given number of iterations.”

References

[1] J. Kiefer and J. Wofowitz. Consistency of the maximum likelihood estimator in the presence of infinitely many incidental parameters. *Ann. Math. Statist.*, 27:887–906, 1956.

[2] X. Wang and Y. Wang. Nonparametric multivariate density estimation using mixtures. *Stat Comput*, 25(1):33–43, Jan 2015.

[3] B. G. Lindsay. The geometry of mixture likelihoods: A general theory. *Ann. Statist.*, 11: 86–94, 1983.

[4] A. Mallet. A maximum likelihood estimation method for random coefficient regression models. *Biometrika*, 73:645–656, 1986.

[5] S. Boyd and L. Vandenberghe. *Convex Optimization.* Cambridge UniversityPress, 2004.

[6] Bradley Bell. Non-parametric population analysis. `http://moby.ihme.washington.edu/bradbell/non_par/non_par.xml`, 2012.

[7] Yeongcheon Baek. *An Interior Point Approach to Constrained Nonparametric Mixture Models*. PhD dissertation, University of Washington, Department of Mathematics, 2006.

[8] WM Yamada, J Bartroff, D Bayard, J Burke, M van Guilder, R Jelliffe, R Leary, M Neely, A Kryshchenko, and A Schumitzky. The nonparametric adaptive grid algorithm for population pharmacokinetic modeling. Technical Report TR-2014-1, Children's Hospital Los Angeles, Los Angeles, CA, 2014. URL `http://www.lapk.org/techReports.php`.

[9] V. V. Fedorov. *Theory of Optimal Experiments*. Academic Press, 1972. edited and translated by W.J. Studden and E.M. Klimko.

[10] A. Schumitzky. Nonparametric EM algorithms for estimating prior distributions. *Applied Mathematics and Computation*, 45:143–157, 1991.

[11] M. L . Lesperance and J. D. Kalbfleisch. An algorithm for computing the nonparametric MLE of a mixing distribution. *J. Am. Stat. Assoc.*, 87:120–126, 1992.

[12] R. S. Pilla, F. Bartolucci, and B. G. Lindsay. Model building for semiparametric mixtures. *arXiv*, 2006.

[13] R. M. Savic, M. C. Kjellsson, and M. O. Karlsson. Evaluation of the nonparametric estimation method in NONMEM VI. *European Journal of Pharmaceutical Sciences*, 37: 27–35, 2009.

[14] R. M. Savic and M. O. Karlsson. Evaluation of an extended grid method for estimation using nonparametric distributions. *AAPS J*, 11(3):615–627, 2009.

[15] Robert Leary. An overview of nonparametric estimation methods used in population analysis. In *Abstracts of the Annual Meeting of the Population Approach Group in Europe*, number Abstract 7383, page 26, Budapest, Hungary, June 2017. PAGE: Population Analysis Group Europe. URL `www.page-meeting.org/?abstract=7383`.

[16] Christopher Jordan-Squire. *Convex Optimization over Probability Measures*. PhD dissertation, University of Washington, Department of Mathematics, 2015.

[17] Henri Faure. Discrépance de suites associées á un système de numération (en dimension s). *Acta Arithmetica*, 41:337–351, 1982.

[18] P. Bratley and B. L. Fox. Algorithm 659: Implementing Sobol's quasirandom sequence generator. *ACM Transactions on Mathematical Software*, 14(1):88–100, 1988. ISSN 0098-3500. URL `http://doi.acm.org/10.1145/42288.214372`. http://www.netlib.org/toms/659.

[19] B. L. Fox. Algorithm 647: Implementation and relative efficiency of quasirandom sequence generators. *ACM Transactions on Mathematical Software*, 12(4):362–376, 1986. ISSN 0098-3500. URL `http://doi.acm.org/10.1145/22721.356187`. http://www.netlib.org/toms/647.

[20] M. Davidian and D. M. Giltinan. *Nonlinear Models for Repeated Measurement Data*. Chapman and Hall/CRC Press, 1995.

[21] M. Davidian and D. M. Giltinan. Nonlinear models for repeated measurement data: An overview and update. *Journal of Agricultural, Biological, and Environmental Statistics*, 8(4):387–419, 2003.

[22] V. Ramos-Martin, A. Johnson, J. Livermore, L. McEntee, J. Goodwin, F //. Whalley, F. Docobo-Perez, T. W. Felton, W. Zhao, E. Jacqz-Aigrain, M. Sharland, M.A. Turner, and W. W. Hope. Pharmacodynamics of vancomycin for cons infection: experimental basis for optimal use of vancomycin in neonates. *J Antimicrob Chemother*, 71:992–1002, 2016.

[23] G.L. Drusano, M.N. Neely, M. van Guilder, A. Schumitzky, D. Brown, S. Fikes, C. Peloquin, and A. Louie. Analysis of combination drug therapy to develop regimens with shortened duration treatment for tuberculosis. *PLoS ONE*, 9(7):e101311, 2014.

[24] K. R. Godfrey. The identifiability of parametric models used in biomedicine. *Math Model*, 7:1195–1214, 1986.

[25] M. Neely, M. van Guilder, W.M. Yamada, A. Schumitzky, and R. Jelliffe. Accurate detection of outliers and subpopulations with pmetrics: a non-parametric and parametric

pharmacometric package for R. *Therapeutic Drug Monitoring*, 34(4):467–476, 2012.

[26] R. Koenker and I. Mizera. Convex optimization, shape constraints, compound decisions, and empirical bayes rules. *J. Am. Stat. Assoc.*, 109:674–85, 2014. `http://www.econ.uiuc.edu/~roger/research/ebayes/brown.pdf`.

[27] H. T. Banks, Z. R. Kenz, and W. C. Thompson. A review of selected techniques in inverse problem nonparametric probability distribution estimation. *Journal of Inverse and Ill-posed Problems*, 20:429–460, 2012.

Algorithm 1 NPAG Algorithm. Input: $(\boldsymbol{Y}, \boldsymbol{\phi}^0, \boldsymbol{a}, \boldsymbol{b}, \Delta_D, \Delta_L, \Delta_F, \Delta_e, \Delta_\lambda)$, $\boldsymbol{a}$ and $\boldsymbol{b}$ are the lists of lower and upper bounds, respectively, of Θ; Δ_D is the minimum distance allowable between points in the estimated F^{ML}. Δ_x see §3.5. Output: $(\boldsymbol{\phi}, \boldsymbol{\lambda}, l(\boldsymbol{\lambda}, \boldsymbol{\phi}))$.

```
procedure NPAG($\boldsymbol{Y}$, $\boldsymbol{\phi}^0, \boldsymbol{a}, \boldsymbol{b}, \Delta_D$)    ▷ Estimate $F^{ML}$ given $\boldsymbol{Y}$
    Initialization: $\boldsymbol{\phi} = \boldsymbol{\phi}^0$, $LogLike = -10^{30}$, $F_0 = 10^{30}$, $F_1 = 2 * F_0$, $eps = 0.2$,
    $\Delta_e = 10^{-4}$, $\Delta_F = 10^{-2}$, $\Delta_L = 10^{-4}$, $\Delta_\lambda = 10^{-3}$, $n = 0$
    while $eps \geq \Delta_e$ or $|F_1 - F_0| \geq \Delta_F$ do
        Calculate $\boldsymbol{\Psi}(\boldsymbol{\phi})$    ▷ $N \times K$ matrix $\{p(Y_i|\phi_k)\}$
        $[\hat{\boldsymbol{\lambda}}(\boldsymbol{\phi}), l(\hat{\boldsymbol{\lambda}}(\boldsymbol{\phi}), \boldsymbol{\phi})] \longleftarrow$ PDIP($\boldsymbol{\Psi}(\boldsymbol{\phi})$)    ▷ Appendix A
        if (MAXCYCLES == 0) then
            $F^{ML}_{est} \longleftarrow l(\hat{\boldsymbol{\lambda}}(\boldsymbol{\phi}), \boldsymbol{\phi})$
            $\boldsymbol{\lambda} \longleftarrow \hat{\boldsymbol{\lambda}}(\boldsymbol{\phi})$
            return $[\boldsymbol{\phi}, \boldsymbol{\lambda}, F^{ML}_{est}]$
        end if
        $n \longleftarrow n + 1$
        $\boldsymbol{\phi}^c \longleftarrow$ CONDENSE($\boldsymbol{\phi}, \hat{\boldsymbol{\lambda}}(\boldsymbol{\phi}), \Delta_\lambda$)    ▷ Alg. 3
        $[\hat{\boldsymbol{\lambda}}(\boldsymbol{\phi}^c), l(\hat{\boldsymbol{\lambda}}(\boldsymbol{\phi}^c), \boldsymbol{\phi}^c)] \longleftarrow$ PDIP($\boldsymbol{\Psi}(\boldsymbol{\phi}^c)$)    ▷ PDIP returns $G^n$
        $NewLogLike = l(\hat{\boldsymbol{\lambda}}(\boldsymbol{\phi}^c), \boldsymbol{\phi}^c)$
        if ($n >$ MAXCYCLES) then
            $F^{ML}_{est} \longleftarrow l(\hat{\boldsymbol{\lambda}}(\boldsymbol{\phi}^c), \boldsymbol{\phi}^c)$
            $\boldsymbol{\lambda} \longleftarrow \hat{\boldsymbol{\lambda}}(\boldsymbol{\phi}^c)$
            return $[\boldsymbol{\phi}, \boldsymbol{\lambda}, F^{ML}_{est}]$
        end if
        if $|NewLogLike - LogLike| \leq \Delta_L$ and $eps > \Delta_e$ then
            $eps = eps/2$    ▷ Adjust precision
        end if
        if $eps \leq \Delta_e$ then    ▷ check EXIT conditions
            $F_1 = NewLogLike$
            if $|F_1 - F_0| \leq \Delta_F$ then
                $F^{ML}_{est} \longleftarrow F_1$
                $\boldsymbol{\phi} \longleftarrow \boldsymbol{\phi}^c; \boldsymbol{\lambda} \longleftarrow \hat{\boldsymbol{\lambda}}(\boldsymbol{\phi}^c)$
                return $[\boldsymbol{\phi}, \boldsymbol{\lambda}, F^{ML}_{est}]$
            else
                $F_0 = F_1$; $eps = 0.2$    ▷ Reset Algorithm
            end if
        end if
        $\boldsymbol{\phi} \longleftarrow \boldsymbol{\phi}^e \longleftarrow$ EXPAND($\boldsymbol{\phi}^c, eps, \boldsymbol{a}, \boldsymbol{b}, \Delta_D$)    ▷ Alg. 2
        $LogLike \leftarrow NewLogLike$
    end while
end procedure
```

Algorithm 2 EXPAND. Input: $\boldsymbol{\phi}$=$(\phi_1,\cdots,\phi_K)$, Δ_G, $\Theta = [a_1,b_1]\times[a_2,b_2]\times ... \times [a_Q,b_Q]$, $\boldsymbol{a} = [a_1,\cdots,a_Q]$, $\boldsymbol{b} = [b_1,\cdots,b_Q]$, Δ_D. Output: $\boldsymbol{\phi}'$=$(\phi'_1,\cdots,\phi'_M)$, where $M \leq K(1+2Q)$. Note: In this algorithm, $\boldsymbol{\phi}$=$(\phi_1,\cdots,\phi_K)$ is a $Q\times K$ matrix, with $Q = \dim\Theta$.

```
function EXPAND($\boldsymbol{\phi}$, $\Delta_G$, $\boldsymbol{a}$, $\boldsymbol{b}$, $\Delta_D$)
    Initialize: $[Q,K] = size(\boldsymbol{\phi})$, $\mathbf{I} = \mathbf{Q}\times\mathbf{Q}$ Identity matrix, new$\boldsymbol{\phi} \longleftarrow \boldsymbol{\phi}$
    for $k = 1,...,K$ do                                  ▷ $K =$ number of input support points
        for $d = 1,...,Q$ do                              ▷ $Q = \dim\Theta$
            $T(d) = \Delta_G(b(d) - a(d))$
            if $\phi(d,k) + T(d) \leq b(d)$ then          ▷ Check upper boundary
                $\boldsymbol{\phi}^+ = \boldsymbol{\phi}(:,k) + T(d)\mathbf{I}(:,d)$
                $dist = 10^{30}$
            end if
            for $k_{in} = 1$ : length(new$\boldsymbol{\phi}$) do
                newdist = $\sum$abs($\boldsymbol{\phi}^+ -$ new$\boldsymbol{\phi}(:,k_{in})$)./($\boldsymbol{b} - \boldsymbol{a}$)     ▷ x ./y done component-wise
                $dist = \min(dist, \text{newdist})$
            end for
            if $dist \geq \Delta_D$ then                  ▷ Check distance to new support point
                new$\boldsymbol{\phi} \longleftarrow$ [new$\boldsymbol{\phi}, \boldsymbol{\phi}^+$]
            end if
            if $\phi(d,k) - T(d) \geq a(d)$ then          ▷ Check lower boundary
                $\boldsymbol{\phi}^- = \boldsymbol{\phi}(:,k) - T(d)\mathbf{I}(:,d)$
                $dist = 10^{30}$
            end if
            for $k_{in} = 1$ : length(new$\boldsymbol{\phi}(1,:)$) do
                newdist = $\sum$(abs($\boldsymbol{\phi}^- -$ new$\boldsymbol{\phi}(:,k_{in})$) ./($b - a$))     ▷ x./y done component-wise
                $dist = \min(dist, \text{newdist})$
            end for
            if $dist \geq \Delta_D$ then                  ▷ Check distance to new support point
                new$\boldsymbol{\phi} \longleftarrow$ [new$\boldsymbol{\phi}, \boldsymbol{\phi}^-$]
            end if
        end for
    end for
    $\boldsymbol{\phi} \longleftarrow$ new$\boldsymbol{\phi}$
end function
```

Algorithm 3 Condense Algorithm. Input: $(\boldsymbol{\phi}, \boldsymbol{\lambda}, \Delta_\lambda)$, Output: $\boldsymbol{\phi}^c$ Note: $\boldsymbol{\phi}^c$ is considered a subset of $\boldsymbol{\phi}$

```
function CONDENSE($\boldsymbol{\phi}, \boldsymbol{\lambda}, \Delta_\lambda$)
    ind = find ( $\boldsymbol{\lambda} > (\max\boldsymbol{\lambda})\Delta_\lambda$ )     ▷ Inequality and max are performed component-wise
    $\boldsymbol{\phi}^c = \boldsymbol{\phi}(:,\mathbf{ind})$
    return $\boldsymbol{\phi}^c$
end function
```

Table A1. Simulation versus optimization. Row 1: True simulated means for each parameter. Row 2: NPAG estimates of corresponding means. Row 3: True simulated variances for each parameter. Row 4: NPAG estimated variances for each parameter.

	Kel	Vc	Ka	Kcp	Kpc
μ_{SIM}	1.305	1.194	0.800	0.205	0.408
μ_{NPAG}	1.308	1.189	0.798	0.209	0.410
σ^2_{SIM}	0.170	0.093	0.042	0.002	0.010
σ^2_{NPAG}	0.173	0.086	0.040	0.003	0.011

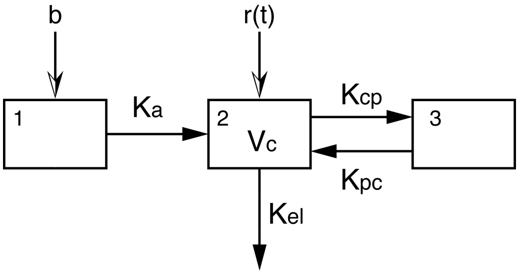


Figure A1. Model.

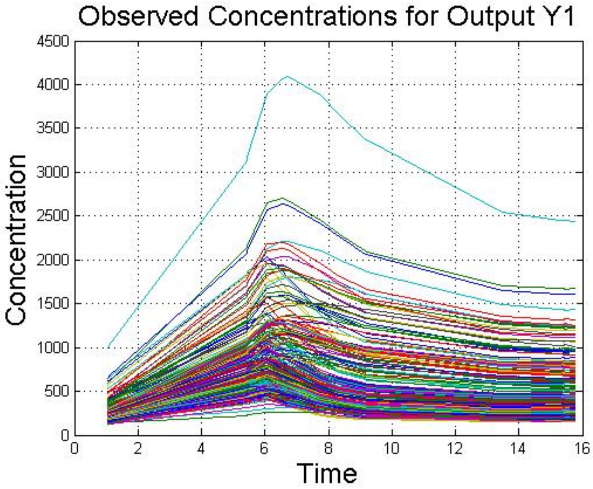
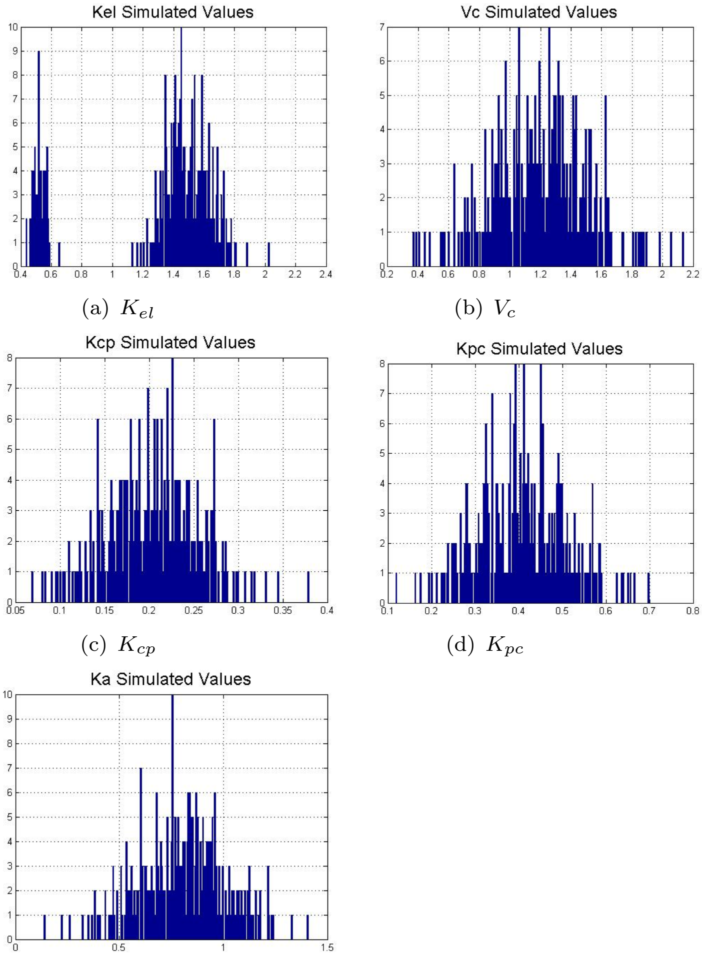


Figure A2. True simulated model profiles.



(a) K_{el}

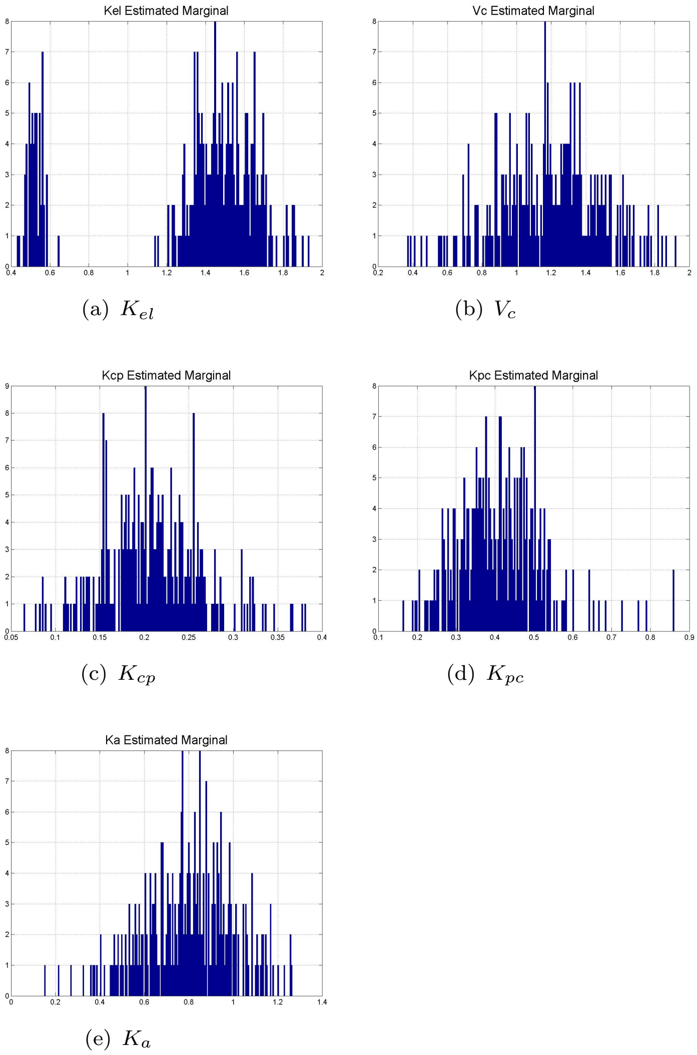
(b) V_c

(c) K_{cp}

(d) K_{pc}

(e) K_a

Figure A3. Histogram of simulated PK parameters.



(a) K_{el}

(b) V_c

(c) K_{cp}

(d) K_{pc}

(e) K_a

Figure A4. Estimated Marginals of PK parameters.

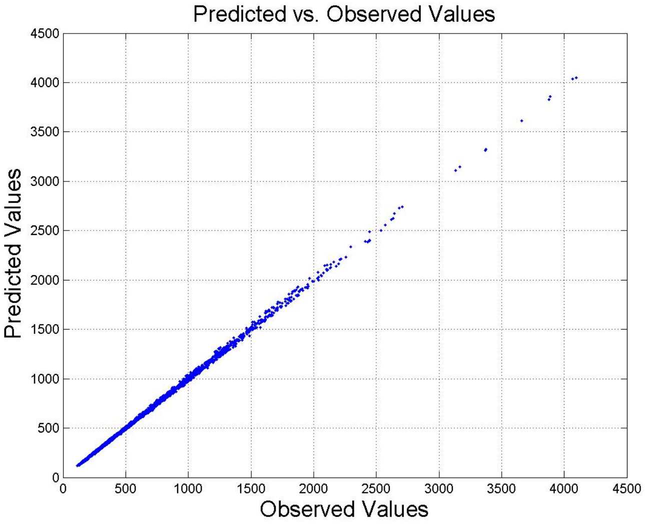


Figure A5. Predicted vs. Observed.