Penalized Nonlinear Least Squares Estimation of Time-Varying Parameters in Ordinary Differential Equations *

Jiguo Cao, Jianhua Z. Huang and Hulin Wu

*Jiguo Cao is Assistant Professor, Department of Statistics and Actuarial Science, Simon Fraser University (Email: jca76@sfu.ca); Jianhua Z. Huang is Professor, Department of Statistics, Texas A&M University, College Station, TX 77843-3143 (Email: jianhua@stat.tamu.edu); Hulin Wu is Professor, Department of Biostatistics and Computational Biology, University of Rochester (Email: hwu@bst.rochester.edu). Cao's work was supported by a discovery grant from the Natural Science and Engineering Research Council of Canada (NSERC). Huang's research was partly supported by NCI (CA57030), NSF (DMS-0907170), and by Award No. KUS-C1-016-04, made by King Abdullah University of Science and Technology (KAUST). Wu's work was partially supported by grants from NIH/NIAID.

Abstract

Ordinary differential equations (ODEs) are widely used in biomedical research and other scientific areas to model complex dynamic systems. It is an important statistical problem to estimate parameters in ODEs from noisy observations. In this article we propose a method for estimating the time-varying coefficients in an ODE. Our method is a variation of the nonlinear least squares where penalized splines are used to model the functional parameters and the ODE solutions are approximated also using splines. We resort to the implicit function theorem to deal with the nonlinear least squares objective function that is only defined implicitly. The proposed penalized nonlinear least squares method is applied to estimate a HIV dynamic model from a real data set. Monte Carlo simulations show that the new method can provide much more accurate estimates of functional parameters than the existing two-step local polynomial method which relies on estimation of the derivatives of the state function. Supplemental materials for the article are available online.

KEYWORDS: Dynamic models; function estimation; penalized splines; time-varying parameters

Running title: Estimating Time-Varying Parameters in ODEs

1. Introduction

Ordinary differential equations (ODEs) describe the dynamics of continuously changing processes by relating a process and its rate of change. They are widely used in many areas of science and technology. In particular, ODEs have been used in HIV/AIDS research to model the viral dynamic system for better understanding of the pathogenesis of HIV infection and to evaluate the antiviral response of antiretroviral regimens (see Wu, 2005, and reference therein). For example, Ho et al. (1995) considered a simple dynamic model $dX(t)/dt = P(t) - cX(t)$, where $X(t)$ is HIV-1 RNA concentrations (viral load) in plasma, $P(t)$ is the virus production rate, and c is the clearance rate of free virions. In order to explain the virus production rate $P(t)$ using the patient's immune status, Chen and Wu (2008) recently studied the following ODE with time-varying coefficients

$$\frac{dX(t)}{dt} = a_1(t) + a_2(t)\mathrm{CD4}(t) - cX(t), \tag{1}$$

where $\mathrm{CD4}(t)$ is the CD4+ T-cell counts. The two functional parameters $a_1(t)$ and $a_2(t)$ are time-varying coefficients that represent the linear relationship between the CD4+ T-cell counts and the virus production rate. In clinical studies, both the viral load $X(t)$ and the CD4 counts are measured at some discrete non-equally spaced points and it is of interest to estimate the time-varying parameters based on these discretely observed measurements.

Motivated by the above HIV dynamic model, we consider a general deterministic dynamic model specified by the ODE

$$\frac{dX(t)}{dt} = \sum_{\ell=1}^{d} a_\ell(t) Z_\ell(t) - g\{X(t)\}, \tag{2}$$

where $X(t)$ is called a state variable, $Z_\ell(t)$, $\ell = 1, \cdots, d$, are the covariates, $a_\ell(t)$, $\ell = 1, \cdots, d$, are the time-varying coefficients, and $g\{X(t)\}$ is some known function of $X(t)$. We assume that the state variable is measured at some discrete time points with certain measurement errors. Specifically, the noisy measurements satisfy

$$Y(t_i) = X(t_i) + \epsilon(t_i), \qquad i = 1, \ldots, n,$$

where t_i, $i = 1, \cdots, n$, are the observation times and $\epsilon(t_i)$ are corresponding measurement errors. It is assumed that the measurement error $\epsilon(t_i)$ is a random variable with mean 0 and variance σ_ϵ^2. Our objective is to estimate the functional parameters $a_\ell(t)$, $\ell = 1, \cdots, d$, using the noisy measurements $Y(t_i)$, $i = 1, \cdots, n$.

Chen and Wu (2008) proposed a two-step local polynomial method for estimating the time-varying parameters in (2). In the first step, the state function $X(t)$ and its first-order derivative $dX(t)/dt$ are estimated from the noisy measurements using local polynomial regression techniques. After plugging estimates of the state function and its derivative into the ODE (2), one obtains a varying-coefficient regression model, which in turn is fitted using existing methods. Since the two-step local polynomial method does not require solving the ODEs, it is a computationally fast method for ODE parameter estimation. However, because the derivative $dX(t)/dt$ usually cannot be accurately estimated from noisy data, the two-step method is not statistically efficient.

In this paper we propose a new method that does not estimate the derivative directly and show using an extensive simulation study that the new method is statistically much more efficient that the two-step method. Our approach blends the method of nonlinear least squares with penalized splines function estimation and solving ODEs with finite elements/splines. The functional parameters are estimated using penalized splines where roughness penalties are used to regularize the functional estimates. Differently from the standard nonlinear least squares problems, our objective function is defined implicitly and does not have a closed-form expression. We apply the implicit function theorem to obtain the numerical gradients required by the Gauss-Newton algorithm. The use of splines also makes the new method computationally fast. Our simulation study shows that the new method provides substantial improvement over the two-step method: The average root mean square error of function estimation by the new method is always smaller than that by the two-step method, and the reduction is greater than 75% in most cases and greater than 90% in more than half of the cases we considered (Section 4).

While this paper focuses on the much less researched area of estimation of functional parameters in ODEs, we would like to point out that there has been an extensive literature on estimation of Euclidean parameters in ODEs; see Section 1.3 of Ramsay et al. (2007) for a review, and Huang et al. (2006), Ramsay et al. (2007), Liang and Wu (2008) for recent development. Our method can be viewed as an extension of Ramsay et al. (2007) to a setting with functional parameters.

The rest of the paper is organized as follows. Section 2 presents the details of the proposed method. Comparison of the new method with the two-step method of Chen and Wu (2008) using real data and simulated data are given in Sections 3 and 4 respectively. The appendix collects some mathematical derivations.

2. Method

2.1 Penalized nonlinear least squares

Denote the solution of the ODE (2) with the initial condition $X(0) = x_0$ by $X^*(t; a_1, \ldots, a_d, x_0)$, which is a nonlinear function of the unknown varying coefficients and the initial condition. It is natural to estimate the unknowns by minimizing the nonlinear least squares criterion

$$\sum_{i=1}^{n} \{Y_i - X^*(t_i; a_1, \ldots, a_d, x_0)\}^2. \tag{3}$$

The time-varying coefficients $a_\ell(t)$, $\ell = 1, \cdots, d$, are functional parameters and are estimated using penalized splines (Eilers and Marx, 1996; Ruppert et al., 2003). Write each $a_\ell(t)$ as a linear combination of B-spline basis functions

$$a_\ell(t) = \sum_{k=1}^{K_\ell} \psi_{\ell k}(t) b_{\ell k} = \psi_\ell(t)^{\mathrm{T}} b_\ell, \tag{4}$$

where $\psi_\ell(t) = (\psi_{\ell 1}(t), \cdots, \psi_{\ell K_\ell}(t))^{\mathrm{T}}$ is a vector of B-splines, and $b_\ell = (b_{\ell 1}, \cdots, b_{\ell K_\ell})^{\mathrm{T}}$ is a vector of basis coefficients. We then replace the functional parameters in (3) by the spline expansion (4) and introduce a roughness penalty to regularize the spline estimates. We adopt the following roughness penalty $\int \{d^2 a_\ell(t)/dt^2\}^2\, dt = b_\ell^{\mathrm{T}} \Omega_\ell b_\ell$, where Ω_ℓ is a penalty matrix with the (k, j)th entry $\int \ddot{\psi}_k(t)\ddot{\psi}_j(t)\, dt$. The penalized nonlinear least squares criterion is

$$\sum_{i=1}^{n} \{Y_i - X^*(t_i; b_1, \ldots, b_d, x_0)\}^2 + \sum_{\ell=1}^{d} \exp(\theta_\ell) b_\ell^{\mathrm{T}} \Omega_\ell b_\ell, \tag{5}$$

where θ_ℓ are penalty parameters whose choice will be discussed later.

One of the challenges of using the nonlinear least squares method is that the ODE solution $X^*(t_i; a_1, \ldots, a_d, x_0)$ usually does not have a closed form. We propose to represent the ODE solution using a spline whose coefficients in a basis expansion are determined by solving another optimization problem. To motivate our method of constructing the ODE solution $X^*(t; b_1, \ldots, b_d, x_0)$ appearing in (5), consider solving (2) with the side conditions $X(t_i) = x_i$, $i = 1, \ldots, n$, for a given set of values x_i. Represent a trial solution $X(t)$ by a basis expansion

$$X(t) = \sum_{j=1}^{J} \alpha_j \phi_j(t) = \phi(t)^{\mathrm{T}} \alpha, \tag{6}$$

where $\phi(t) = (\phi_1(t), \ldots, \phi_J(t))^{\mathrm{T}}$ is a vector of B-splines, and $\alpha = (\alpha_1, \ldots, \alpha_J)^{\mathrm{T}}$ is a vector of coefficients. The method of least squares minimizes

$$\sum_{i=1}^{n} \{x_i - X(t_i)\}^2 + \lambda_x \int \left[\frac{dX(t)}{dt} - \sum_{\ell=1}^{d} a_\ell(t) Z_\ell(t) + g\{X(t)\} \right]^2 dt$$

(Eason, 1976). The factor λ_x is the relative weight of the side-condition residuals with respect to the differential-equation (DE) residuals. In our case, we do not have exact side conditions but the noisy observations Y_i can play a similar role. Thus we propose to obtain the ODE solution by minimizing with respect to α the criterion

$$J \equiv \sum_{i=1}^{n} \{Y_i - X(t_i)\}^2 + \lambda_x \int \left[\frac{dX(t)}{dt} - \sum_{\ell=1}^{d} a_\ell(t) Z_\ell(t) + g\{X(t)\} \right]^2 dt, \tag{7}$$

where $X(t)$ depends on α as specified in (6), and $a_\ell(t)$'s depends on b_ℓ as in (4). Since the DE residuals can be made arbitrarily small by increasing the number of basis functions while the magnitude of the side-condition residuals stays no lower than the noise level, we put more emphasis on the DE residuals by using a very large value of λ_x. Our experience shows that the solution of (7) is not sensitive to the value of λ_x as long as it is large enough; setting λ_x in the range of $10^5 \sim 10^8$ works well in all of our numerical examples. On the other hand, since the ODE solution is approximated by a finite dimensional space of splines, an extremely large value of λ_x will make the error due to spline approximation dominate and will cause the method to collapse. Such a situation can be easily spotted and avoided by checking the sensitivity of the solution to λ_x.

The integration in (7) usually does not have a closed form expression and needs to be evaluated using numerical quadrature. We use the composite Simpson's rule, which provides a good approximation to the exact integral (Burden and Douglas, 2000). Let Q be an even integer. For a function $f(t)$, the composite Simpson's rule is

$$\int_{t_1}^{t_Q} f(t)dt \approx \frac{h}{3}\Big\{ f(s_0) + 2\sum_{q=1}^{Q/2-1} f(s_{2q}) + 4\sum_{q=1}^{Q/2} f(s_{2q-1}) + f(s_Q)\Big\},$$

with the quadrature points $s_q = t_1 + qh$, $q = 0, \cdots, Q$, and $h = (t_Q - t_1)/Q$. To make the approximation accurate, Q needs to be reasonably large, i.e., $Q = 10J$, where J is the number of basis functions used in (6) for representing the ODE solution.

The minimizer α^* of (7) is plugged into (6) to obtain an ODE solution

$$X^*(t; b_1, \dots, b_d) = \phi(t)^{\mathrm{T}}\alpha^*(b_1, \dots, b_d). \tag{8}$$

Note that the dependence of this solution on x_0 is dropped since $X^*(0) = \phi(0)^{\mathrm{T}}\alpha^*$ is determined by α^*, which in turn is determined by the data and $b_1, \dots, b_d$. Given this ODE solution, the penalized nonlinear least squares criterion (5) can be rewritten as

$$H \equiv \sum_{i=1}^{n}\{Y_i - \phi(t_i)^{\mathrm{T}}\alpha^*(b_1, \dots, b_d)\}^2 + \sum_{\ell=1}^{d} \exp(\theta_\ell) b_\ell^{\mathrm{T}}\Omega_\ell b_\ell. \tag{9}$$

The main difference between minimization of (9) and the usual nonlinear least squares problem is that a quantity in the objective function, $\alpha^*(b_1, \ldots, b_d)$, is not defined explicitly but only implicitly as a solution to another optimization problem. In the following subsection we discuss how to overcome the difficulty associated with an implicitly defined objective function and develop an algorithm for solving the minimization problem. Denote the minimizer of (9) as $\hat{b} = (\hat{b}_1^{\mathrm{T}}, \ldots, \hat{b}_d^{\mathrm{T}})^{\mathrm{T}}$, then the estimates of the functional parameters are given by $\hat{a}_\ell(t) = \psi_\ell(t)^{\mathrm{T}}\hat{b}_\ell$, $\ell = 1, \ldots, d$.

2.2 The modified Guass-Newton algorithm

Except in the special case that $g\{X(t)\}$ is a linear function of $X(t)$ where a closed-form solution exists as shown at the end of this subsection, we propose to use the iterative Gauss–Newton algorithm to minimize (9).

The application of the Gauss-Newton algorithm is non-standard here because the vector-valued function $\alpha^*(b_1, \ldots, b_d)$ does not have an analytical expression and its evaluation relies on numerical methods. Given $b_1, \ldots, b_d$, let α_0 be an initial guess value of α, the Newton–Raphson algorithm for computing $\alpha^*(b_1, \ldots, b_d)$ iterates

$$\alpha_{v+1} = \alpha_v - \left(\left.\frac{\partial^2 J}{\partial \alpha \partial \alpha^{\mathrm{T}}}\right|_{\alpha_v} \right)^{-1} \left(\left.\frac{\partial J}{\partial \alpha}\right|_{\alpha_v} \right) \tag{10}$$

until convergence. The analytical expressions of the derivatives involved in the Newton-Raphson iteration (10) are given in the Appendix.

To obtain the gradients needed for the Gauss–Newton algorithm to minimize (9), we resort to the implicit function theorem. Denote $b = (b_1^{\mathrm{T}}, \ldots, b_d^{\mathrm{T}})^{\mathrm{T}}$. Taking the b-derivative on both sides of the identity $\partial J/\partial\alpha|_{\alpha^*} = 0$, we obtain

$$\frac{d}{db^{\mathrm{T}}} \left(\left.\frac{\partial J}{\partial \alpha}\right|_{\alpha^*} \right) = \left.\frac{\partial^2 J}{\partial \alpha \partial b^{\mathrm{T}}}\right|_{\alpha^*} + \left.\frac{\partial^2 J}{\partial \alpha \partial \alpha^{\mathrm{T}}}\right|_{\alpha^*} \frac{\partial \alpha^*(b)}{\partial b^{\mathrm{T}}} = 0,$$

which yields

$$\frac{\partial \alpha^*(b)}{\partial b^{\mathrm{T}}} \quad = \quad -\left(\left.\frac{\partial^2 J}{\partial \alpha \partial \alpha^{\mathrm{T}}}\right|_{\alpha^*}\right)^{-1}\left(\left.\frac{\partial^2 J}{\partial \alpha \partial b^{\mathrm{T}}}\right|_{\alpha^*}\right), \tag{11}$$

provided that $\partial^2 J/(\partial\alpha\partial\alpha^{\mathrm{T}})|_{\alpha^*}$ is non-singular. The analytical expressions of the derivatives appearing on the right side of (11) are given in the Appendix. Given the gradients, we can linearize $\alpha^*(b)$ at a trial value b^0 and then replace the criterion (9) by

$$\sum_{i=1}^{n}\left[\left\{Y_i - \phi(t_i)^{\mathrm{T}}\alpha^*(b^0) + \left.\frac{\partial \alpha^*(b)}{\partial b^{\mathrm{T}}}\right|_{b=b^0} b^0\right\} - \left.\frac{\partial \alpha^*(b)}{\partial b^{\mathrm{T}}}\right|_{b=b^0} b\right]^2 + \sum_{\ell=1}^{d}\exp(\theta_\ell) b_\ell^{\mathrm{T}}\Omega_\ell b_\ell. \tag{12}$$

The Gauss–Newton algorithm minimizes iteratively this ridge-regression or penalized least squares criterion until convergence is reached.

The computation is much simplified in the special case that $g\{X(t)\}$ is a linear function, i.e. $g\{X(t)\} = cX(t)$, where the minimizer of (7) has a closed-from expression

$$\alpha^*(b_1,\ldots,b_d) = (\Phi^{\mathrm{T}}\Phi + \lambda_x A)^{-1}(\Phi^{\mathrm{T}}Y + \lambda_x u),$$

where Φ is an $n\times J$ matrix whose i-th row is $\phi(t_i)^{\mathrm{T}}$, $A = \int\{L_0\phi(t)\}\{L_0\phi(t)\}^{\mathrm{T}}\,dt$ is an $J\times J$ matrix with $L_0\phi(t) = d\phi(t)/dt + c\phi(t)$, $Y = (Y_1,\ldots,Y_n)^{\mathrm{T}}$, and

$$u = \sum_{\ell=1}^{d}\int\{a_\ell(t)Z_\ell(t)\}\{L_0\phi(t)\}dt = \sum_{\ell=1}^{d} b_\ell^{\mathrm{T}}\int\{\psi_\ell(t)Z_\ell(t)\}\{L_0\phi(t)\}dt.$$

Since $\alpha^*(b_1,\ldots,b_d)$ depends on $b = (b_1^{\mathrm{T}},\ldots,b_d^{\mathrm{T}})^{\mathrm{T}}$ linearly, minimization of (9) is a ridge regression type of problem and has a closed-from solution.

2.3 Variance of the parameter estimates

It follows from (4) that the variances of estimates of the functional parameters are given by

$$\mathrm{var}\{\hat{a}_\ell(t)\} = \psi_\ell(t)^{\mathrm{T}}\mathrm{var}(\hat{b}_\ell)\psi_\ell(t), \tag{13}$$

which depends on the variance matrix of $\hat{b}_\ell$, which in turn is obtained using the Gauss–Newton algorithm. The Gauss–Newton algorithm is essentially an iterative penalized least squares algorithm. At each step the optimizing criterion (12) can be rewritten as

$$\|Y^{\dagger 0} - U^0 b\|^2 + b^{\mathrm{T}} \Omega^{\dagger} b, \tag{14}$$

where the working response vector $Y^{\dagger 0}$ has components

$$Y_i^{\dagger 0} = Y_i - \phi(t_i)^{\mathrm{T}} \alpha^*(b^0) + \left. \frac{\partial \alpha^*(b)}{\partial b^{\mathrm{T}}} \right|_{b=b^0} b^0,$$

the working "design matrix" $U^0 = \{\partial \alpha^*(b)/\partial b^{\mathrm{T}}\}|_{b=b^0}$, and the penalty matrix

$$\Omega^{\dagger} = \mathrm{diag}\{\exp(\theta_1)\Omega_1, \ldots, \exp(\theta_d)\Omega_d\}.$$

The minimizer of (14) has a closed form

$$b^1 = (U^{0\mathrm{T}} U^0 + \Omega^{\dagger})^{-1} U^{0\mathrm{T}} Y^{\dagger 0}.$$

When $b^0 = b^1 = \hat{b}$ is taken to be the value at convergence of the algorithm, its variance has the typical sandwich form

$$\mathrm{var}(\hat{b}) = (U^{0\mathrm{T}} U^0 + \Omega^{\dagger})^{-1} U^{0\mathrm{T}} \mathrm{var}(Y^{\dagger 0}) U^0 (U^{0\mathrm{T}} U^0 + \Omega^{\dagger})^{-1}. \tag{15}$$

To obtain an estimated variance matrix of $\hat{b}$, we simply replace $\mathrm{var}(Y^{\dagger 0})$ by $\hat{\sigma}_\epsilon^2 I$, where $\hat{\sigma}_\epsilon^2$ is the sample variance of $Y_i^{\dagger 0}$, $i = 1, \ldots, n$.

2.4 Choosing the penalty parameters

The penalty parameters can be chosen using generalized cross validation (GCV) as is commonly used in the spline smoothing literature (Wood, 2006, Section 4.5). For the penalized nonlinear least squares criterion (9), the GCV score is defined as

$$\mathrm{GCV}(\theta_1, \ldots, \theta_d) = n \frac{\sum_{i=1}^n \{Y(t_i) - \phi(t_i)^{\mathrm{T}} \alpha^*(\hat{b}_1, \ldots, \hat{b}_d)\}^2}{(n - \mathrm{df})^2}, \tag{16}$$

where df is the effective degrees of freedom obtained using the usual linearization argument. In light of (14), the smoothing matrix after linearization of the problem is given by

$$S = U^0 (U^{0\mathrm{T}} U^0 + \Omega^\dagger)^{-1} U^{0\mathrm{T}}$$

and $\mathrm{df} = \mathrm{trace}(S)$. We minimize the GCV score with respect to $\theta = (\theta_1, \ldots, \theta_d)^\mathrm{T}$ to obtain suitable penalty parameters. When there are only one or two penalty parameters, the grid search method works well for finding good penalty parameters. When there are more than two penalty parameters, to speed up the search, we can resort to a multidimensional optimization algorithm such as the downhill simplex method (Nelder and Mead, 1965) or a Newton type method (Wood, 2006, Section 4.6).

2.5 Parameter cascading

Our estimation procedure consists of three nested levels of optimisation and thus the computational algorithm has three nested loops. In the inner loop, it minimizes the LS criterion (7) with respect to α for fixed b and θ; in the middle loop, it minimizes the PNLS criterion (5) with respect to b for fixed θ; in the outer loop, it minimizes the GCV criterion (16) with respect to θ. The three optimization criteria are interconnected in the way that the parameters in the lower level optimization problems are used to define the criteria in the upper level optimization problems. Such a nested structure leads to the notation of *parameter cascading* (Cao and Ramsay, 2007; Ramsay et al., 2007). The important feature of our parameter cascade is that there are no closed-from expressions to describe the interdependence among the parameters in the three levels of optimization, and the implicit function theorem provides the necessary analytical derivatives for efficient and stable computation.

3. Application

The proposed penalized nonlinear least squares (PNLS) method is applied to a real data set from an AIDS clinical study reported by Chen and Wu (2008). In this study, highly actively antiretroviral therapy (HAART) and immune-based therapy were used to treat HIV-1-infected patients. For each patient, both viral load and CD4+ T-cell counts were measured at baseline and around the scheduled times at weeks 2, 4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 44, 48, 56, 64, 72, 80, 88, and 96 after initiating the treatment. The actual measurement times vary because the patients did not strictly follow the schedule. Separately for each of the five patients, Chen and Wu (2008) estimated the two time-varying coefficients $a_1(t)$ and $a_2(t)$ in model (1) using their two-step local polynomial regression. The constant viral dynamic parameter c appearing in (1) is estimated separately using a different short-term HIV dynamic model (Perelson et al., 1996; Han and Chaloner, 2004) from more frequent viral load data within the first three days. The second column of Table 1 gives the estimate of c for each of five patients.

When applying our method, the trial solution of the ODE was represented using a cubic spline with 72 equally spaced knots (i.e., four times the number of observations), the time-varying coefficients of the ODE were represented using cubic splines with 10 equally spaced knots. Ten knots should be enough for modeling the time-varying coefficients since they are assumed to be smoother than the CD4 profiles for their identifiability. B-splines are used as the basis functions. The weighting parameter λ_x in (7) is set to be 10^8 and the penalty parameters θ_1 and θ_2 in (14) are chosen by minimizing the GCV score as defined in (16). The third and fourth columns of Table 1 display the chosen penalty parameters for each patient.

Figure 1 displays our estimated time-varying parameters $a_1(t)$, $a_2(t)$ for five patients, along with the 95% confidence intervals. The estimates by the two-step local polynomial regression method are also displayed. Our PNLS method yields smoother functional estimates. To compare the estimates

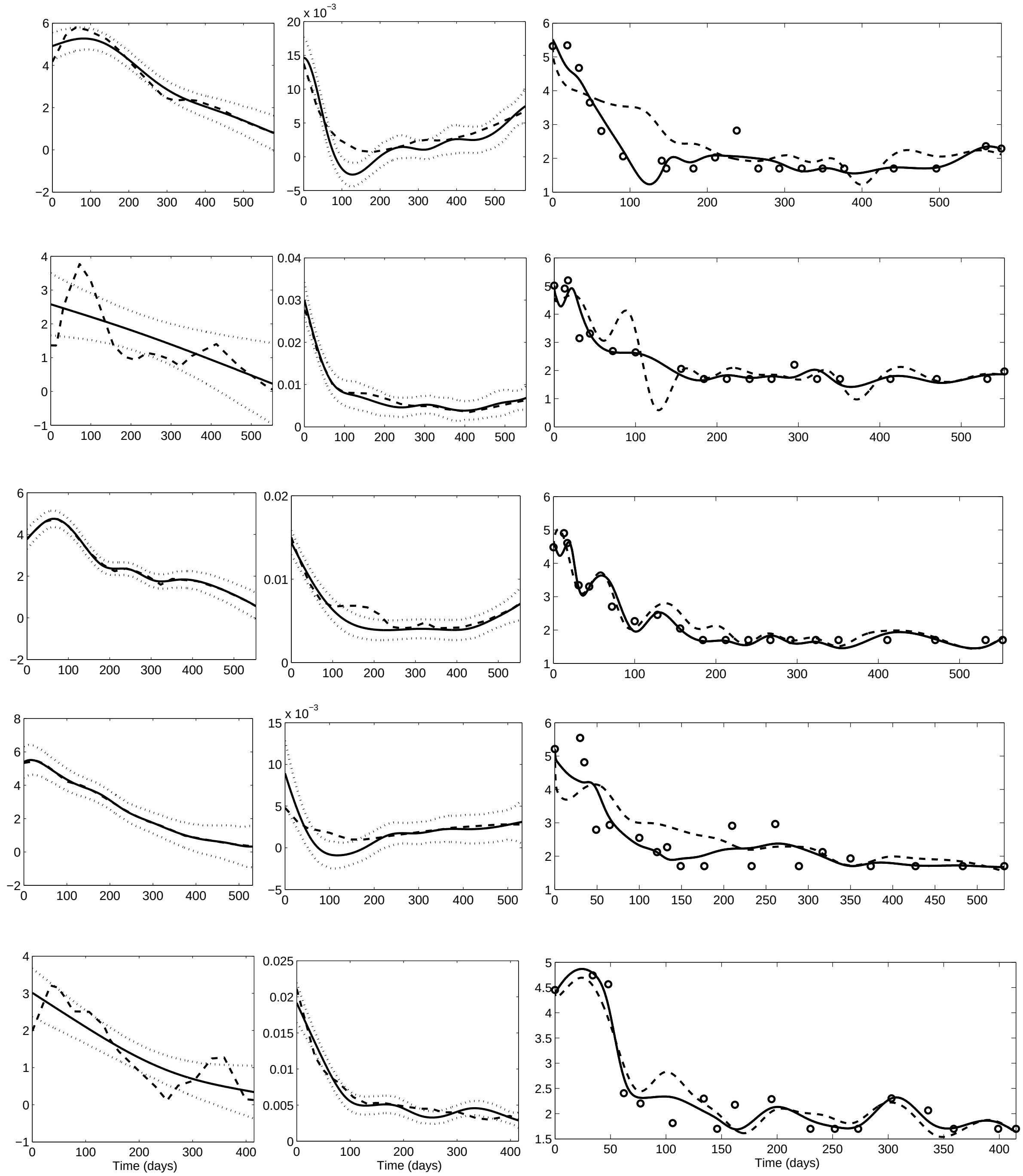

Figure 1: The analysis results for the HIV data with five patients. Each row displays the result for one patient. The solid lines in the left and middle panels show respectively the estimates of $a_1(t)$ and $a_2(t)$ using the PNLS method with their 95% pointwise confidence intervals given in dotted lines. The dashed lines in the left and middle panels are the estimates of $a_1(t)$ and $a_2(t)$ using the two-step local polynomial method. The right panels depict the numerical solutions of the ODE using the estimated $\hat{a}_1(t)$ and $\hat{a}_2(t)$ from the PNLS method (solid lines) and the two-step local polynomial method (dashed lines). The circles represent the actual measurements of the viral load.

from the two methods, we numerically solved the ODE (1) with the estimated parameters and checked the fit of the ODE solutions to observations. Figure 1 shows that the ODE solutions with our PNLS estimated parameters are closer to the observations than those with the two-step local polynomial regression estimates. The last two columns of Table 1 present the mean squared distances (MSD) between the ODE solutions and the observations. Compared with the two-step local polynomial regression estimates, our PNLS method reduces MSDs by 81.8%, 54.2%, 20.0%, 49.1%, and 64.3% for the five patients in the study, respectively.

The PNLS method is currently programmed in Matlab 7.8.0, and the two-step local polynomial regression method is programmed in C, so it is not completely fair to compare their computing time. However, it would still be helpful to report the computing time of these two methods. The PNLS method used 62 seconds to obtain the estimates for Patient 1 in a personal laptop in Windows XP with an Intel Pentium Dual 2.16GHz CPU and a 3.46 GB of RAM, while the two-step local polynomial regression method used 2 seconds to obtain the estimates for Patient 1 on the same laptop. The PNLS method would take less time to run if programmed in the low level C programming language. In general, we expect that the PNLS method would be slower to run than the two-step method even when programmed in the same language, but not too slow to affect its practical application.

4. Simulation

4.1 A simple ODE

Consider a simple ODE

$$\frac{dX(t)}{dt} = -cX(t) + a(t), \tag{17}$$

which has an analytic solution

$$X(t) = e^{-ct}\left\{X(t_0) + \int_{t_0}^{t} e^{cs}a(s)ds\right\}. \tag{18}$$

Table 1: Summary of the analysis results for the HIV data with five patients. $\hat{\sigma}_\epsilon$ represents estimated error standard deviation using the PNLS residuals. MSD_{pnls} represents MSD for the PNLS method; $\text{MSD}_{\text{2-step}}$ represents MSD for the two-step local polynomial method.

Patient	$\hat{c}$	$\hat{\theta}_1$	$\hat{\theta}_2$	$\hat{\sigma}_e$	MSD_{pnls}	$\text{MSD}_{\text{2-step}}$
1	2.09	−12.17	−10.85	0.29	0.08	0.44
2	2.37	−11.53	−9.26	0.33	0.11	0.24
3	2.40	−16.72	−10.21	0.22	0.04	0.05
4	1.69	−15.34	−9.86	0.68	0.27	0.53
5	2.34	−10.59	−8.51	0.22	0.05	0.14

Set $c = 0.1$ as a known constant, the initial value $X(t_0) = 1$, and the functional parameter $a(t) = \cos(2\pi t)$ is to be estimated. The simulated data are obtained by evaluating the ODE solution (18) at 101 equally spaced "time" points in [0,1] and then adding independent zero-mean Gaussian noise with standard deviation 0.02.

When applying our PNLS method, the state variable $X(t)$ is represented using cubic B-splines with knots positioned at the observation times, the functional parameter $a(t)$ is estimated as cubic B-splines with one interior knot at 0.5. Since the ODE (17) has an analytic solution, one can directly minimize the PNLS criterion (5) to obtain an estimate of the functional parameter. Although this direct PNLS method is not applicable in general because of typical lack of analytical ODE solutions, it does provide a good benchmark in this simple example to evaluate the approximation property of our proposed PNLS method and examine the effects of varying the control factor λ_x in our approximated ODE solving criterion (7).

Figure 2 displays the mean squared differences (over a grid of 101 points) of the functional

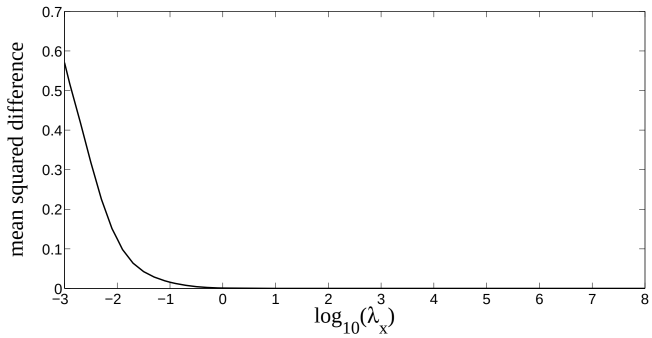

Figure 2: The mean squared differences of estimates of the functional parameter by the proposed PNLS method with different control factors and by the direct PNLS method, for the simple ODE $dX(t)/dt = -cX(t) + a(t)$.

parameter estimates obtained by our PNLS method with various values of λ_x and the estimate by the direct PNLS method. The differences become smaller as λ_x gets larger and can be negligible when $\lambda_x \geq 1$. When λ_x is large enough, the estimates by our method is almost identical to that by the direct PNLS method.

We also compared our PNLS method with the two-step local polynomial estimate. Figure 3 shows that the PNLS method produces estimates with smaller biases, standard deviations and root mean squared errors than the two-step method.

4.2 Simulation models generated from the real data

We have conducted an extensive simulation study which clearly shows that our PNLS method consistently provides more accurate parameter estimates than the two-step local polynomial method. We present first the results from two setups where the simulated data were generated from a model obtained by fitting the real data from patient 1 of the HIV study using respectively the PNLS and the two-step local polynomial method. The time-varying coefficient functions are depicted in the

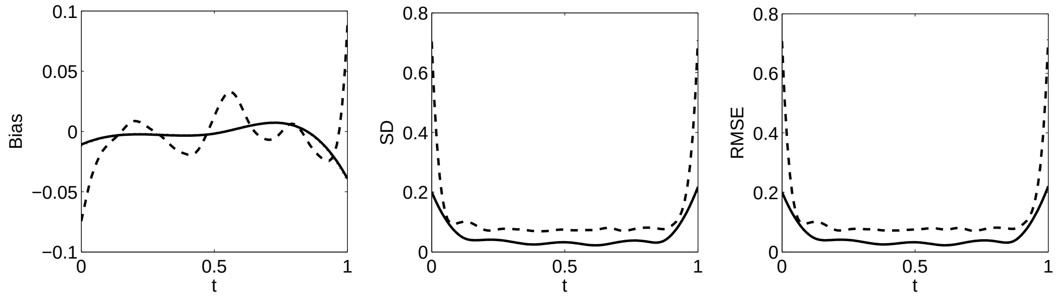

Figure 3: Simulation results for the simple ODE $dX(t)/dt = -cX(t) + a(t)$. The biases, standard deviations (SD), and the root mean squared errors (RMSE) for estimates of the functional parameter, from 100 simulations, are shown. The solid and dashed lines correspond respectively to the PNLS and the two-step local polynomial methods.

first row of Figure 1. For each setup, the ODE with given coefficient functions was solved and then zero-mean normal errors with the standard deviation 0.29 were added to the ODE solution. The observation times were set to be the same as the real data. For each simulated data set, both the PNLS method and the two-step local polynomial method were applied to estimate the time-varying coefficients. When applying the PNLS method, the basis functions were specified as in the real data analysis and the GCV criterion (16) was used for choosing the penalty parameters.

To compare different methods, define the integrated squared error (ISE) and the integrated absolute error (IAE) when using $\hat{f}$ to estimate the function f over the interval $[a, b]$ as

$$\text{ISE} = \int_a^b \{\hat{f}(t) - f(t)\}^2 \, dt \quad \text{and} \quad \text{IAE} = \int_a^b |\hat{f}(t) - f(t)| \, dt.$$

In our simulation study, the boundary of the interval is taken to be the data range and the integral is evaluated as a Riemann sum with 1001 equally spaced points.

Figure 4 displays the biases, standard deviations, and root mean squared errors for the estimates

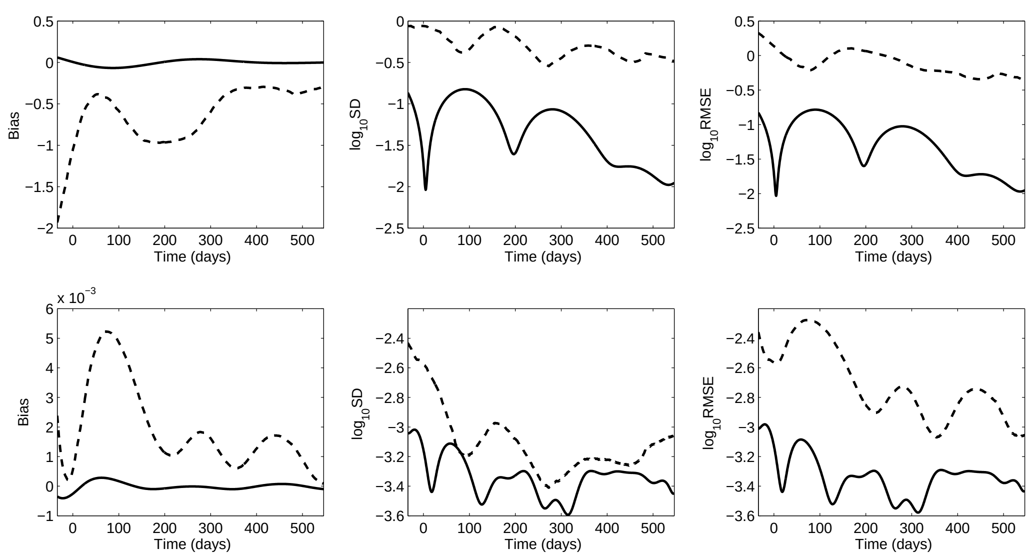

Figure 4: Simulation results when the simulation data are generated from the model obtained by fitting the patient 1 data using the PNLS method. The biases, standard deviations (SD), and the root mean squared errors (RMSE) for the estimates $\hat{a}_1(t)$ and $\hat{a}_2(t)$ from 100 simulations are shown. The solid and dashed lines correspond respectively to the PNLS and the two-step local polynomial methods.

of $\hat{a}_1(t)$ and $\hat{a}_2(t)$ from 100 simulations where the true coefficient functions are obtained by fitting the real data using the PNLS method. Note that the standard deviation and root mean squared errors are plotted in the log scale with base 10, thus one unit smaller means 90% reduction. It is clear from Figure 4 that the PNLS method produces much more accurate functional parameter estimates than the two-step local polynomial method. Indeed, comparing with the two-step local polynomial method, the PNLS method reduces the average squared biases, the average standard deviations, and the average root mean squared errors by 99%, 98%, 99% for $a_1(t)$ and 99%, 81%, 96% for $a_2(t)$, respectively.

Figure 5 shows the same simulation as in Figure 4 but with the coefficient functions in the simulation model obtained by fitting the real data using the two-step local polynomial method. Again we observe that the PNLS method outperforms the two-step method by having substantially smaller biases, standard deviations, and root mean squared errors.

In addition to the two setups presented above, we also considered eight other simulation setups where the data from patients 2–5 were used to generate the simulation models. The summary statistics of the ISE and IAE for the two methods, based on 100 simulation runs for each setup, for all ten setups are given in Tables 2 and Tables 3. The PNLS method consistently yields smaller mean ISEs and IAEs than the two-step method; the reduction is substantial and is more than 75% in many cases.

Appendix

The Analytical expressions of the derivatives in (10) **and** (11)

$$\frac{\partial J}{\partial \alpha} = -2\sum_{i=1}^{n}[\{Y_i - X(t_i)\}\phi(t_i)] + 2\lambda_x \int \left[\frac{dX(t)}{dt} - \sum_{\ell=1}^{d} a_\ell(t) Z_\ell(t) + g\{X(t)\}\right]\left[\frac{d\phi(t)}{dt} + \left.\frac{dg(x)}{dx}\right|_{x=X(t)}\phi(t)\right]dt,$$

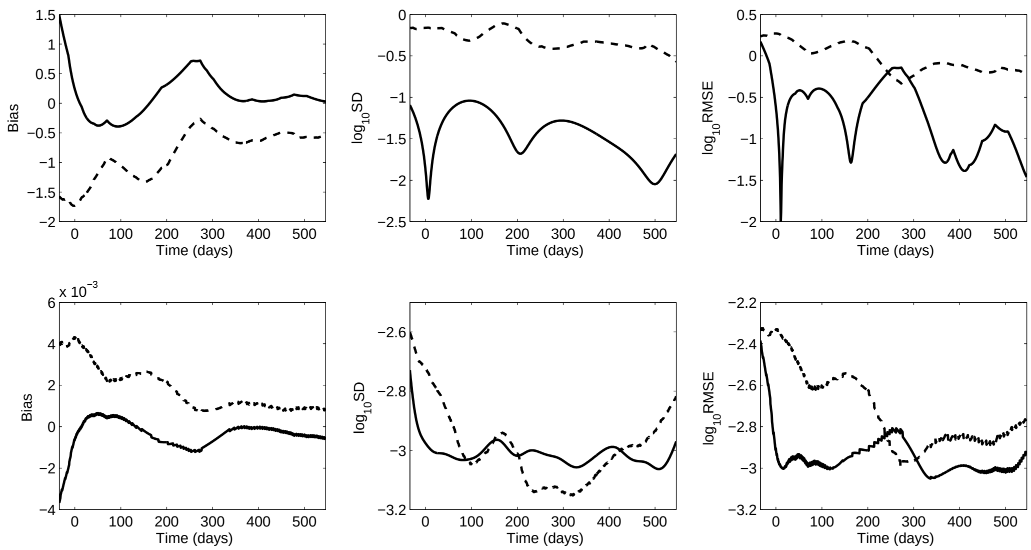

Figure 5: Simulation results when the simulation data are generated from the model obtained by fitting the patient 1 data using the two-step local polynomial method. The biases, standard deviations (SD), and the root mean squared errors (RMSE) for the estimates $\hat{a}_1(t)$ and $\hat{a}_2(t)$ from 100 simulations are shown. The solid and dashed lines correspond respectively to the PNLS and the two-step local polynomial methods.

Table 2: The means (SEs) of the ISEs and IAEs for estimating the functional parameters when the data are generated from a model by fitting the real data from each of the five patients using the PNLS method. PNLS represents the penalized nonlinear least squares method; 2-Step represents the 2-step local polynomial method.

Model	Method	ISE$\{\hat{a}_1(t)\}$ ($\times 10^3$)	IAE$\{\hat{a}_1(t)\}$ ($\times 10^3$)	ISE$\{\hat{a}_2(t)\}$ ($\times 10^8$)	IAE$\{\hat{a}_2(t)\}$ ($\times 10^5$)
1	PNLS	7(2)	26(6)	27(5)	20(4)
	2-Step	806(92)	665(46)	697(38)	207(7)
2	PNLS	21(2)	32(2)	19(1)	34(1)
	2-Step	671(41)	185(44)	537(25)	170(4)
3	PNLS	7(2)	19(6)	36(4)	34(3)
	2-Step	209(10)	377(12)	157(4)	102(2)
4	PNLS	73(2)	144(12)	85(3)	69(2)
	2-Step	552(33)	585(22)	264(8)	122(3)
5	PNLS	57(4)	169(6)	108(4)	79(1)
	2-Step	210(36)	286(25)	551(8)	187(3)

Table 3: The means (SEs) of the ISEs and IAEs for estimating the functional parameters when the data are generated from a model by fitting the real data from each of the five patients using the two-step local polynomial method. PNLS represents the penalized nonlinear least squares method; 2-Step represents the 2-step local polynomial method.

Model	Method	ISE$\{\hat{a}_1(t)\}$ ($\times 10^2$)	IAE$\{\hat{a}_1(t)\}$ ($\times 10^2$)	ISE$\{\hat{a}_2(t)\}$ ($\times 10^7$)	IAE$\{\hat{a}_2(t)\}$ ($\times 10^5$)
1	PNLS	14(0)	28(0)	16(1)	94(3)
	2-Step	117(7)	90(4)	56(3)	190(7)
2	PNLS	22(0)	31(2)	14(1)	75(3)
	2-Step	43(0)	53(3)	21(1)	101(4)
3	PNLS	18(0)	34(0)	9(0)	71(1)
	2-Step	33(2)	46(1)	11(1)	84(2)
4	PNLS	0(0)	33(2)	28(1)	115(2)
	2-Step	45(3)	54(2)	39(1)	145(2)
5	PNLS	287(8)	142(1)	462(4)	596(3)
	2-Step	428(14)	169(4)	520(8)	667(6)

$$
\begin{aligned}
\frac{\partial^2 J}{\partial\alpha\partial\alpha^T} &= 2\sum_{i=1}^{n}\{\phi(t_i)\phi^T(t_i)\} \\
&\quad + 2\lambda_x \int \left[\frac{d\phi(t)}{dt} + \left.\frac{dg(x)}{dx}\right|_{x=X(t)}\phi(t)\right]\left[\frac{d\phi(t)}{dt} + \left.\frac{dg(x)}{dx}\right|_{x=X(t)}\phi(t)\right]^T dt \\
&\quad + 2\lambda_x \int \left[\frac{dX(t)}{dt} - \sum_{\ell=1}^{d} a_\ell(t)Z_\ell(t) + g\{X(t)\}\right]\left[\left.\frac{d^2 g(x)}{dx^2}\right|_{x=X(t)}\phi(t)\phi^T(t)\right]dt, \\
\frac{\partial^2 J}{\partial\alpha\partial b_\ell^T} &= -2\lambda_x \int \left[\frac{d\phi(t)}{dt} + \left.\frac{dg(x)}{dx}\right|_{x=X(t)}\phi(t)\right] Z_\ell\, \psi_\ell^T(t)dt.
\end{aligned}
$$

SUPPLEMENTAL MATERIALS

Matlab Code: The supplemental files for this article include Matlab programs which can be used to replicate the real data analysis in Section 3 of the article. Please read the file README contained in the zip file for more details. (CaoHuangWu.zip, zip archive)

References

Burden, R. L. and Douglas, F. J. (2000), *Numerical Analysis*, Pacific Grove, California: Brooks/Cole, seventh ed.

Cao, J. and Ramsay, J. O. (2007), "Parameter Cascades and Profiling in Functional data analysis," *Computational Statistics*, 22, 335–351.

Chen, J. and Wu, H. (2008), "Efficient Local Estimation for Time-Varying Coefficients in Deterministic Dynamic Models With Applications to HIV-1 Dynamics," *Journal of the American Statistical Association*, 103, 369–383.

Eason, E. D. (1976), "A review of least-squares methods for solving partial differential equations," *International Journal for Numnerical Methods in Engineering*, 10, 1021–1046.

Eilers, P. and Marx, B. (1996), "Flexible smoothing with B-splines and penalties (with Discussion)," *Statistical Science*, 89, 89–121.

Han, C. and Chaloner, K. (2004), "Bayesian Experimental Design for Nonlinear Mixed-Effects Models With Application to HIV Dynamics," *Biometrics*, 60, 25C33.

Ho, D. D., Neumann, A. U., Perelson, A. S., Chen, W., Leonard, J. M., and Markowitz, M. (1995), "Rapid Turnover of Plasma Virions and CD4 Lymphocytes in HIV-1 Infection," *Nature*, 374, 123–126.

Huang, Y., Liu, D., and Wu, H. (2006), "Hierachical Bayesian Methods for Estimation of Parameters in a Longitudinal HIV Dynamic System," *Biometrics*, 62, 413–423.

Liang, H. and Wu, H. (2008), "Parameter Estimation for Differential Equation Models Using a Framework of Measurement Error in Regression Models," *Journal of the American Statistical Association*, 103, 1570–1583.

Nelder, J. and Mead, R. (1965), "A simplex method for function minimization," *Computer Journal*, 7, 308–313.

Perelson, A. S., Neumann, A. U., Markowitz, M., Leonard, J. M., and Ho, D. D. (1996), "HIV-1 Dynamics in vivo: Virion Clearance Rate, Infected Cell Life-Span, and Viral Generation Time," *Science*, 271, 1582–1586.

Ramsay, J. O., Hooker, G., Campbell, D., and Cao, J. (2007), "Parameter estimation for differential equations: a generalized smoothing approach (with discussion)," *Journal of the Royal Statistical Society, Series B*, 69, 741–796.

Ruppert, D., Wand, M. P., and Carroll, R. J. (2003), *Semiparametric Regression*, Cambridge: Cambridge University Press.

Wood, S. N. (2006), *Generalized Additive Models: An Introduction with R*, New York: Chapman and Hall/CRC.

Wu, H. (2005), "Statistical Methods for HIV Dynamic Studies in AIDS Clinical Trials," *Statistical Methods in Medical Research*, 14, 171–192.