Solving systems of ordinary differential equations that arise in PKPD. Part I - linear systems with time-constant parameters.

Gareth Hegarty*† Stephen Duffull*

Abstract

In this the first of three papers exploring the solution of systems of ordinary differential equations that arise in PKPD we look at linear constant coefficient ODEs. We will show that this type of ODEs can be solved explicitly using matrix exponentials and that these solutions can be evaluated simply using eigenvalues and eigenvectors. These solutions allow us to evaluate the PKPD model quickly and accurately.

1 Introduction

Pharmacokinetic-pharmacodynamic (PKPD) models are used to describe the time course of pharmacological response. Models that are developed to describe PKPD processes are often defined as ordinary differential equations (ODEs) [1]. An ordinary differential equation is defined as a differential equation that depends on a single independent variable. In the case of ODEs for PKPD the independent variable is invariably time therefore providing a description (or prediction) in the rate of change in some response(s) of interest

*School of Pharmacy, Division of Health Sciences, University of Otago, Dunedin, New Zealand.

†Corresponding Author. Current address: Department of Mathematics and Statistics, University of Otago, P.O. Box 56, Dunedin 9054, New Zealand. E-mail: ghegarty@maths.otago.ac.nz

over time. Specifying models as (a series of) ODEs allows users enormous flexibility in which to present and explore the mechanistic content of the biological system. In PKPD we are generally interested in first order ODEs of the form:

$$\frac{dy}{dt} = f(t, y; \beta); \quad y(t_0) = y_0 \tag{1}$$

where t is time, $y(t)$ is a vector of response variables (i.e. concentrations and/or effects), β is a vector of parameters inherent in the model and y_0 is a vector of initial conditions at $t = 0$. In a mathematical sense and for the sake of this paper ODEs can be formally grouped into three basic categories depending on the way in which the parameters and response variables y enter the model:

1. linear constant coefficient
2. linear variable coefficient
3. nonlinear

This grouping, although specified from a mathematical perspective, retains useful distinctions which we see in PKPD (see [2] for background material on ODEs). Note in case three we do not consider whether the coefficients are constant or variable since the influence of nonlinearity on the system outweighs any consideration of how the coefficients are expressed. The term linear indicates that the right hand side of equation (1) is linear in the response variable y. So for example the following ODE is linear in y.

$$\frac{dy}{dt} = \beta y; \quad y(t_0) = y_0 \tag{2}$$

The term "constant coefficient" implies that the coefficients (β in this case) are constant over the time range that the ODEs are to be solved. In contrast, a variable coefficient is one where its value may change over the time range of the ODE. In PKPD a linear variable coefficient ODE would be one where β varied over time, commonly termed a time varying covariate. Such an ODE may have the general form:

$$\frac{dy}{dt} = \beta(t) y; \quad y(t_0) = y_0 \tag{3}$$

Which remains linear in y, but is now time varying in β. Such systems are common in PKPD where β would typically be a function of the concentration of some substrate which itself changes over time. The last two groups are those where the right hand side of the ODE is nonlinear in y. A common PKPD example of such an ODE is:

$$\frac{dy}{dt} = \frac{\beta_1 y}{\beta_2 + y}; \quad y(t_0) = y_0 \tag{4}$$

In this case the ODE is nonlinear in y and (in this particular case) time-constant in β. This latter observation makes little difference when considering its solution.

In PKPD we encounter all categories of ODEs in particular categories 2 and 3. For simplicity and in keeping with the PKPD literature we will use the terms linear or nonlinear (to indicate linearity with respect to the response) and time-constant and time-varying (to indicate the dependence of the parameters on time). Division of ODEs into these categories is helpful when we consider their solutions.

Integration of ODEs over time to provide a solution to the function of response versus time is not easy. Five groups of methods are generally available (although others are possible).

- exact solutions in closed form
- approximate solutions in closed form (e.g. where the solution is approximated by a finite series)
- iterative numerical forms that converge to the exact solution
- time-stepping solutions
- Monte-Carlo solutions

The key requirement for any method is a solution that is highly accurate and computationally fast. It is clear that the first method is highly desirable as the solution is accurate to machine precision and the speed is simply that required to evaluate the function (which is likely to be extremely rapid). In the PKPD literature, however, the majority of reports describe the use of method

four (time-stepping). Time-stepping procedures are numerical techniques for computing solutions to the ODE by conditioning the next time step solution on the immediate previous time point. Euler's method (described in [3]) is perhaps the simplest and is defined by a linear progression:

$$y_{n+1} = y_n + hf(t_n, y_n; \beta) \tag{5}$$

where (t_n, y_n) is the current point, h is the step size and $f(t, y; \beta)$ is given by the right hand side of equation (1). More accurate (and complex) methods are used for actual data analysis such as Runge-Kutta methods [3], for example MATLABs ode23 or ode45, that provide discrete approximations over finite time periods. These methods are implemented in common software packages such as ADAPT [4], NONMEM [5], WinBUGS [6] and Monolix [7]. The benefit of time-stepping methods is their generality for solving ODEs that may arise from any of the three categories. They do have drawbacks however, in that they tend to be slow and prone to propogation of errors.

It is relevant to reflect briefly on the importance of accurately defining the model. PKPD models are generally developed by use of estimation techniques. These techniques rely (at least to some extent) on evaluating the likelihood which is itself a function of the model. In circumstances where the maximum likelihood estimator is desired then it is not uncommon to use search methods based on taking derivatives of the likelihood (see [8]). Similarly in a design setting the model is the basis of the likelihood and the information matrix is formed by the second partial derivative of the likelihood with respect to the parameters. In both of these cases inaccuracies in estimating the model may have considerable implications for dependent processes.

This paper is part one of a three-part series that describe accurate and fast solutions for ODEs that typically arise in PKPD analysis. In part I we describe well known solutions for linear ODEs with time-constant parameters. In part II we propose solutions for linear ODEs with time-varying parameters and finally in part III we describe new methods for solving nonlinear ODEs.

2 Linear constant coefficient ODEs

2.1 1-compartment model

Example 1. Suppose we give a bolus dose D of a drug (at $t = 0$) into a central compartment with volume of distribution V and elimination rate constant k ($= CL/V$, where CL is clearance). The amount in the central compartment at time t, $A_m(t)$, will satisfy the ODE:

$$\frac{dA_m}{dt} = -kA_m; \quad A_m(0) = D \tag{6}$$

Here $y(t) = A_m(t)$, $\beta = (CL, V)$, and $f(t, y; \beta) = -kA_m$. We can obviously *solve* this ODE to get $A_m(t)$ and $C(t)$:

$$A_m(t) = De^{-kt}; \quad C(t) = \frac{A_m(t)}{V} \tag{7}$$

Of course this example is very simple, but it does illustrate that, rather than the ODE itself, we are more interested in the *solution* of the ODE. However, as we will see, the are many examples of ODEs in PKPD where it may be difficult or *impossible* to write down an *exact* formula in *closed form* for the solution (especially when the ODE is *nonlinear*).

2.2 Matrix exponentials

Here we show how the general form for the solution of linear constant coefficient ODEs is given by matrix exponentials. We will see how matrix exponentials may be evaluated formally using Taylor series, but more practically evaluated in terms of the eigenvalues and eigenvectors of the respective matrix. The first example in this section (example 2) is include for illustrative purposes as the series approximation to the exponential can be used exactly. In the following section it is inconvenient to use the series expansion exactly and an eigenvalue decomposition is shown for two PK examples.

2.2.1 A Taylor series expansion

For the linear ODE:

$$\frac{dy}{dt} = Ay, \; y(0) = y_0 \tag{8}$$

the solution is obviously

$$y(t) = e^{tA} y_0 \tag{9}$$

This same solution formula works when y is a vector of length n and A is an $n \times n$ matrix. To define the exponential e^{tA} we may use the Taylor series formula[1]:

$$e^{tA} = I + \frac{tA}{1!} + \frac{t^2 A^2}{2!} + ... \tag{10}$$

- where I is the $n \times n$ identity matrix.

Example 2. For our first example we consider the simple first order linear system:

$$\begin{cases} \dfrac{dy_1}{dt} = y_2, \\ \dfrac{dy_2}{dt} = -y_1 \end{cases}$$

which is essentially the equations for the simple harmonic oscillator (mass-spring system) from physics. We choose this rather simple example because we can easily evaluate the expontial e^{tA} using the Taylor series formula in this case which highlights the utility of this method. Obviously this can be written in the form (8) by letting:

$$y(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix}, \quad A = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}$$

Now consider powers of the matrix A:

$$A^2 = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} = -I,$$

$$A^3 = -I \times A = -A, \; A^4 = -A \times A = I$$

Generalizing this we get:

$$\left.\begin{array}{l} A^{2n} = (-1)^n I \\ A^{2n+1} = (-1)^n A \end{array}\right\} \; n = 0, 1, 2, ...$$

If we separate the odd and even powers in (10) and use the formulae above for the powers of A, we get:

$$e^{tA} \;\; = \;\; \left(I + \frac{t^2 A^2}{2!} + \frac{t^4 A^4}{4!} + ... \right) + \left(\frac{tA}{1!} + \frac{t^3 A^3}{3!} + ... \right)$$

[1] Assuming that this sum converges.

$$
\begin{aligned}
&= \left(I + \frac{t^2(-1)^1 I}{2!} + \frac{t^4(-1)^2 I}{4!} + ...\right) + \left(\frac{tA}{1!} + \frac{t^3(-1)^1 A}{3!} + ...\right) \\
&= \left(1 - \frac{t^2}{2!} + \frac{t^4}{4!} - ...\right) I + \left(t - \frac{t^3}{3!} + \frac{t^5}{5!} - ...\right) A \\
&= \cos(t) I + \sin(t) A \\
&= \begin{bmatrix} \cos(t) & \sin(t) \\ -\sin(t) & \cos(t) \end{bmatrix}
\end{aligned}
$$

Thus, according to (9), we get the solution:

$$
y(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} = e^{tA} \begin{bmatrix} y_1(0) \\ y_2(0) \end{bmatrix} = \begin{bmatrix} \cos(t) & \sin(t) \\ -\sin(t) & \cos(t) \end{bmatrix} \begin{bmatrix} y_1(0) \\ y_2(0) \end{bmatrix}
$$

■

2.2.2 Eigenvalue decomposition

In practice, however, it is often difficult to evaluate e^{tA} from the series directly. Instead it is more convenient to use the eigenvalue/eigenvector decomposition of A:

$$
A = P\Lambda P^{-1}; \quad \Lambda = \begin{bmatrix} \lambda_1 & 0 & ... & 0 \\ 0 & \lambda_2 & ... & 0 \\ 0 & 0 & \ddots & \vdots \\ 0 & 0 & ... & \lambda_n \end{bmatrix}, \quad P = [\mathbf{v_1}, ..., \mathbf{v_n}]
$$

where $\lambda_1, ..., \lambda_n$ are the eigenvalues[2] of A and $\mathbf{v_1}, ..., \mathbf{v_n}$ are the corresponding eigenvectors (i.e. $n \times 1$ column-vectors). Using this decomposition we can write e^{tA} as:

$$
e^{tA} = P e^{t\Lambda} P^{-1}; \; e^{t\Lambda} = \begin{bmatrix} e^{t\lambda_1} & 0 & ... & 0 \\ 0 & e^{t\lambda_2} & ... & 0 \\ 0 & 0 & \ddots & \vdots \\ 0 & 0 & ... & e^{t\lambda_n} \end{bmatrix}
$$

[2]Here we assume that the eigenvalues are distinct and real, though this is not necessary in general.

Example 3. Our first linear PK example is a 1-compartment model with first order input, governed by the linear system of ODEs:

$$\begin{cases} \dfrac{dy_1}{dt} = -k_a y_1, & y_1(0) = D \\ \dfrac{dy_2}{dt} = k_a y_1 - k y_2, & y_2(0) = 0 \end{cases} \tag{11}$$

This example is useful to see the process as it is not so trivial that an eigenvalue solution is ungainly but sufficiently well known that the exact solution is well recognised:

$$C(t) = \frac{D}{V}\frac{k_a}{(k_a - k)}(e^{-kt} - e^{-k_a t}) \tag{12}$$

In this example we derive the full process in order to illustrate how it is achieved. It is not necessary in practice to go to this level since the solution to $Pe^{t\Lambda}P^{-1}$ is able to be evaluated in most software packages (e.g. MATLAB) without recourse to the workings shown below. MATLAB code to evaluate matrix exponentials which could be easily adapted to this example is given in appendix A.

The matrix A is given by:

$$A = \begin{bmatrix} -k_a & 0 \\ k_a & -k \end{bmatrix}$$

The product of A and y given by:

$$\begin{bmatrix} -k_a & 0 \\ k_a & -k \end{bmatrix}\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} -k_a y_1 + 0 y_2 \\ k_a y_1 - k y_2 \end{bmatrix}$$

It is clear to see that this matrix represents the system of ODEs given in equation (11). Evaluating the eigenvalues and eigenvectors for A we find:

$$P = \begin{bmatrix} k - k_a & 0 \\ k_a & 1 \end{bmatrix}, \quad \Lambda = \begin{bmatrix} -k_a & 0 \\ 0 & -k \end{bmatrix}$$

Taking the Taylor series expansion for e^{tA} we get

$$\begin{aligned} e^{tA} &= I + \frac{tP\Lambda P^{-1}}{1!} + \frac{t^2(P\Lambda P^{-1})^2}{2!} + ... && (13) \\ &= P(I + \frac{t\Lambda}{1!} + \frac{t^2\Lambda^2}{2!} + ...)P^{-1} && (14) \\ &= Pe^{t\Lambda}P^{-1} && (15) \end{aligned}$$

and hence substituting P and Λ into the following expression for the system

$$\begin{aligned}\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} &= Pe^{t\Lambda}P^{-1}\begin{bmatrix} D \\ 0 \end{bmatrix} \\ &= \frac{1}{k-k_a}\begin{bmatrix} k-k_a & 0 \\ k_a & 1 \end{bmatrix}\begin{bmatrix} e^{-k_a t} & 0 \\ 0 & e^{-kt} \end{bmatrix}\begin{bmatrix} 1 & 0 \\ -k_a & k-k_a \end{bmatrix}\begin{bmatrix} D \\ 0 \end{bmatrix}\end{aligned}$$

we get:

$$\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} = \frac{1}{k-k_a}\begin{bmatrix} (k-k_a)e^{-k_a t} & 0 \\ k_a(e^{-k_a t}-e^{-kt}) & (k-k_a)e^{-kt} \end{bmatrix}\begin{bmatrix} D \\ 0 \end{bmatrix}$$

This simplifies to:

$$\begin{bmatrix} y_1(t) \\ y_2(t) \end{bmatrix} = \frac{D}{k-k_a}\begin{bmatrix} (k-k_a)e^{-kt} \\ k_a(e^{-k_a t}-e^{-kt}) \end{bmatrix} \tag{16}$$

Dividing the second row by V will provide the solution for the concentration in the central compartment. It is clear to see in this simple example that the solution is now available for each row of the derivative (i.e. for y_1 & y_2). This allows the solution to be evaluated for each compartment (e.g. gut and central) and if the values in these compartments are recorded it is simple to naturally handle recursive dosing structures. ■

Example 4. The 2nd linear PK example is a standard multiple response parent-metabolite model. Where a drug is given orally and then the parent is irreversibly metabolised to a metabolite. Both parent and metabolite are measured.

$$\begin{cases} \dfrac{dy_1}{dt} = -k_{12}y_1, & y_1(0) = D \\ \dfrac{dy_2}{dt} = k_{12}y_1 - [k_{23} + f_m k_{24} + (1-f_m)k_2]y_2 + k_{32}y_3, & y_2(0) = 0 \\ \dfrac{dy_3}{dt} = k_{23}y_2 - k_{32}y_3, & y_3(0) = 0 \\ \dfrac{dy_4}{dt} = f_m k_{24}y_2 - k_4 y_4, & y_4(0) = 0 \end{cases} \tag{17}$$

where:

$$y_1(t) \;=\; \text{Amount of drug in gut at time } t$$

$$
\begin{aligned}
y_2(t) &= \text{Amount in central compartment for parent} \\
y_3(t) &= \text{Amount in peripheral compartment for parent} \\
y_4(t) &= \text{Amount in central compartment for metabolite} \\
k_{mn} &= \text{Rate constant from compartment m to n.} \\
f_m &= \text{fraction of parent converted to metabolite}
\end{aligned}
$$

In this model samples would be taken from compartments 2 and 4, as in figure 1. Thus writing the system (17) in the form (8):

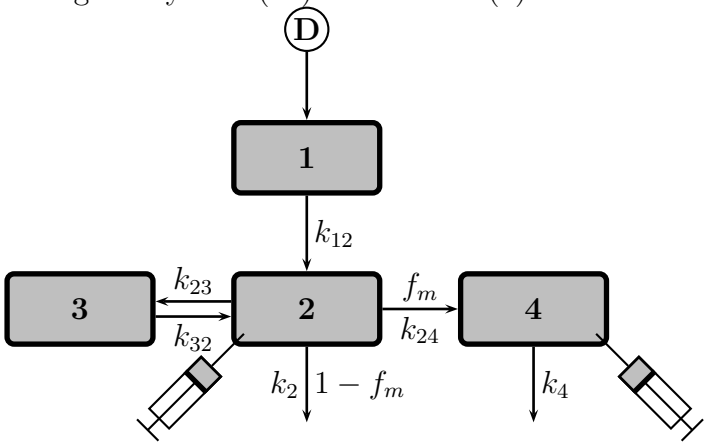

Figure 1: Compartments in PK model

$$\frac{dy}{dt} = Ay; \quad y(0) = y_0$$

where :

$$A = \begin{bmatrix} -k_a & 0 & 0 & 0 \\ k_a & -[k_{23} + f_m k_{24} + (1 - f_m)k_2] & k_{32} & 0 \\ 0 & k_{23} & -k_{32} & 0 \\ 0 & f_m k_{24} & 0 & -k_4 \end{bmatrix},$$

and

$$y = y(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \\ y_3(t) \\ y_4(t) \end{bmatrix}, \; y_0 = \begin{bmatrix} D \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

We can use the eigenvalue/eigenvalue decomposition as before to write $A = P\Lambda P^{-1}$, where:

$$\Lambda = \begin{bmatrix} (-\alpha-\beta)/2 & 0 & 0 & 0 \\ 0 & (-\alpha+\beta)/2 & 0 & 0 \\ 0 & 0 & -k_4 & 0 \\ 0 & 0 & 0 & -k_a \end{bmatrix},$$

and

$$P = \begin{bmatrix} 0 & 0 & 0 & \frac{(k_a-k_4)}{f_m k_{24} k_a}\left\{\alpha - \frac{(k_{32}^2+k_{32}k_{23}-k_a^2)}{k_{32}-k_a}\right\} \\ \frac{-\alpha+\beta-2k_4}{2f_m k_{24}} & \frac{-\alpha+\beta-2k_4}{2f_m k_{24}} & 0 & \frac{k_4-k_a}{f_m k_{24}} \\ \frac{k_{23}(\alpha+\beta-2k_4)}{f_m k_{24}(\alpha-2k_{32}+\beta)} & \frac{k_{23}(-\alpha+\beta-2k_4)}{f_m k_{24}(\alpha-2k_{32}+\beta)} & 0 & \frac{k_{23}k_4-k_{23}k_a}{f_m k_{24}k_{32}-f_m k_{24}k_a} \\ 1 & 1 & 1 & 1 \end{bmatrix}$$

and

$$\alpha = k_2(1-f_m) + k_{23} + f_m k_{24} + k_{32}$$

$$\beta = \sqrt{(k_2(1-f_m) + k_{23} + f_m k_{24})^2 - 4k_{32}(k_2(1-f_m) + f_m k_{24})}$$

Thus, the solution to (17) is given by:

$$\begin{bmatrix} y_1(t) \\ y_2(t) \\ y_3(t) \\ y_4(t) \end{bmatrix} = Pe^{t\Lambda}P^{-1}\begin{bmatrix} D \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

where P and Λ are given above. This solution, with parameter values taken from [9], is shown in figure 2. For simplicity and without loss of generality we have omitted the 2nd compartment for the metabolite. In 3 we show the error in the approximate solution generated using the MATLAB solver ode45, i.e. the difference between the ode45 solution (approximate) and the matrix exponential solution (exact). As mentioned prevoiusly, the matrix exponential solution can be computed numerically and MATLAB code for the parent-metabolite example is provided in appendix A.■

Remark. One can also easily generalise the matrix exponential solution method to apply to equations of the form:

$$\frac{dy}{dt} = Ay + f,\ y(t_0) = y_0 \tag{18}$$

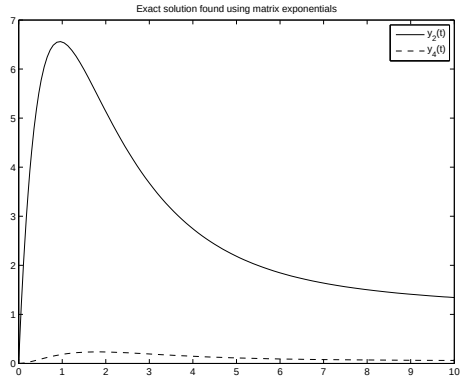

Figure 2: Profiles of $y_2(t)$ and $y_4(t)$

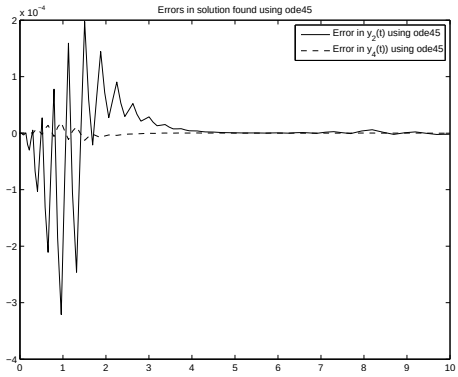

Figure 3: Errors in ODE solutions for $y_2(t)$ and $y_4(t)$

where $f = (f_1, ..., f_n)$ is a vector of constants of the same length as y and A is an invertible $n \times n$ matrix of constants (which is the case in our previous examples). Then we can redefine the solution for $y(t)$ to include f and express this as $z(t)$ to give:

$$z(t) = y(t + t_0) + A^{-1}f$$

Hence we get the linear ODE for $z(t)$:

$$\frac{dz}{dt} = Az, \;\; z(0) = y_0 + A^{-1}f$$

and from the solution to this we can find the solution to (18), i.e.

$$y(t) = z(t - t_0) - A^{-1}f = e^{(t-t_0)A}(y_0 + A^{-1}f) - A^{-1}f$$

This solution is then easily ammenable to the standard matrix exponential method.■

2.3 Discussion

Defining PKPD models as ODEs and evaluating their solution is common practice in pharmacometrics. Evaluation of the ODEs may be required in order to perform simulations from the model or it may be required to provide inference regarding some estimation or optimization procedure. For inferential purposes speed and accuracy are key factors when evaluating ODEs. The

most common solution for ODEs involves numerical integration using time-stepping algorithms such as the Runge-Kutta or similar methods. While these algorithms are very powerful and generally robust they are not without their difficulties. In a general sense time-stepping ODE solvers provide a *black box* solution and unless a reference solution is available it is not easy to determine whether the *a priori* choice of tolerance and ODE solver will best meet the needs of the intended use of the model. Setting tolerance too small (e.g. 1×10^{-8}) may result in lengthy delays and in some circumstances exceed the maximum number of iterations allowable for the ODE solver. Whereas setting the tolerance too large may result in unacceptable errors that are not uniformly distributed over the time domain (e.g. see Figure 3)). In addition, the choice of non-stiff or stiff ODE solvers also poses a problem.

The ODE examples shown in this paper are defined as linear ODEs. Their solution is relatively simple and provides a good foundation from which to explore the more complex time varying and nonlinear ODE systems. For the purpose of this paper and papers II and III that follow we define models as *closed form* and *exact* based on the following nomenclature. Let our model be defined as

$$y(t) = f(t;\beta). \tag{19}$$

If the model can be defined as a function of time (i.e. $y(t) = ...$) then we determine that the expression is in *closed form*; and if the model can be defined by $f(t;\beta)$ then we denote this as *exact*. The solutions provided for the linear ODEs in this paper are therefore all *closed form* & *exact*. When we address the case of time-varying systems we will introduce the idea of *closed form* & *approximate*, where in this case the function $f(t;\beta)$ is expressed to the level of an analytic series or an integral that can be solved using Gaussian quadrature. Finally, when addressing nonlinear ODEs we will also introduce the solution as an *approximate closed form* such that $y(t) \approx ....$ In all of these cases the approximations used can be expressed to the level of accuracy required by the user and hence can be indistinguishable from a *closed form* & *exact* solution. This differs from differential equations which are neither *closed form* or *exact*. In addition, the solutions provided are much faster than that provided time-stepping ODE.

As a rule it will be seen, other than for the most simple models, that closed form exact solutions generally appear more complex and less general.

Whereas time-stepping numerical solutions for ODEs appear general and due to their *black box* nature simpler. Although this may appear a caveate in the approaches described it will also be seen that standard numerical techniques can be used to solve closed form exact solutions quickly and with a level of accuracy similar to the analytic solution.

In conclusion, closed form and exact solutions, or close approximations to both components, should be used whenever possible. Their utility not only lies in the vastly superior speed of computation but also a greater level of accuracy. These solutions also avoid the *black box* methods which can provide erroneous results without warning.

A MATLAB code for matrix exponentials

```
A = [-Ka,0,0,0;Ka,-(K23+fm*K24+(1-fm)*K2),K32,0;0,K23,-K32, ...
  0;0,fm*K24,0,-K4];
[P,Lam]=eig(A);
t=[.01:.01:8];
q=inv(P)*[D;0;0;0];
R = zeros(length(q),length(t));
for j=1:length(q)
    R(j,:)=exp(Lam(j,j)*t)*q(j);
end
y = P*R;
```

References

[1] Sharma A, Jusko WJ, 'Characteristics of indirect pharmacodynamic models and applications to clinical drug responses', *Br J Clin Pharmacol*, (1998) **45**, 229–239.

[2] Ayres F, 'Theory and Problems of Differential Equations', Schaums Outline Series, McGraw-Hill (1952).

[3] Zwillinger D, 'Handbook of Differential Equations', Third Edition, Academic Press (1998).

[4] D'Argenio DZ, Schumitzky A, ADAPT II User Guide (1997), Biomedical Simulations Resource. Los Angeles: University of Southern California.

[5] Sheiner LB, Beal SL, 'NONMEM Users Guide' (1979), San Francisco: Division of Pharmacy, San Francisco: University of California San Francisco.

[6] Spiegelhalter D, Thomas A, Best N, WinBUGS User Manual, Version 1.1.1.

[7] Laveille M, Mentre F, *Monolix Ver 2.3 User Guide*, INSERM U738, Paris (2007).

[8] Bates DM, Watts DG, 'Nonlinear regression analysis and its applications', Wiley Series in Probability and Statistics', Wiley (1988).

[9] Duffull SB, Chabaud S, Nony P, Laveille C, Girard P, Aarons L, 'A pharmacokinetic simulation model for ivabradine in healthy volunteers', *Eur J Pharm Sci* (2000); **10**, 285–94.