Gene Regulatory Network Modeling with Multiple Linear Regressions

Shing-Hwang Doong

Department of Information Management

ShuTe University, YenChau, KaoHsiung 824

tungsh@mail.stu.edu.tw

Abstract

Gene regulatory network modeling is a difficult inverse problem. Given limited amount of experimental data about gene expressions, a dynamic model is sought to fit the data to infer interesting biological processes. In this study, a well-known ecological system, the Lotka-Volterra system of differential equations, is used to model the dynamics of genes regulations. After replacing derivatives by estimated slopes, this system is decoupled into several independent systems of linear equations. Coefficients of the original Lotka-Volterra system are inferred from these linear systems by using multiple linear regressions. Two function approximation techniques, namely the cubic spline and the artificial neural network, are used to help estimate the stated slopes. It is found that the cubic spline interpolation and multiple linear regressions have provided useful solutions to the gene regulatory network problem.

Keywords ： Gene regulatory network, multiple linear regressions, cubic spline interpolation, artificial neural network

1. Introduction

As more high quality time dense profiles of metabolites or proteins are available due to the rapid technological development in mass spectrometry and nuclear magnetic resonance, biochemical pathway analysis is becoming an important tool in system biology. Using microarray technology, the expression levels of thousands of genes may be monitored together. Labs can collect these data repeatedly during an interesting biological process. These time courses become important profiles to understand how genes regulate each other in a biological process.

Gene regulatory network (GRN) is a pathway analysis that studies the activation or repression of genes expressions through the nucleotide or protein products of other genes. In addition to the wet land approach and literature summarization, GRN can be developed by using mathematical modeling. A successful modeling of GRN will give biologists insights to design more effective and efficient experiments in validating a pathway hypothesis.

A mathematical modeling of GRN starts with a model to describe the dynamics involved in a GRN. Then, the model is fit with the observed profile data to find model parameters. Finally, these parameters are interpreted and used to discover interesting biological findings.

Researchers in bioinformatics have proposed many interesting models to describe the dynamics of a gene regulatory network [2]. Several time-based models are based on systems of differential equations.

Kikuchi et al. used cluster computing and genetic algorithm to find the network structure of a GRN [6]; Noman and Iba used differential evolution to solve the GRN problem [9]. These researchers basically solve the differential equations directly to compute the fitness of a set of network coefficients, which are evolved heuristically by using evolutionary computation. On the other extreme, Almeida and Voit [1] and Veflingstad et al. [14] avoided the tedious steps of solving differential equations by converting them into nonlinear regression problems. Dynamic Bayesian network [10] and Petri net [7] have also been proposed to study GRN in previous research.

Though many modeling tools are available for GRN analysis, this study adopts the most popular one by using systems of ordinary differential equation (ODE). Previous studies have focused on the S-system, which is a canonical system that can represent many types of biochemical processes [13]. In this study, we focus on another canonical system that has been used primarily in ecology [8]. The Lotka-Volterra system (LVS) of differential equations is an extension to the well-known predator-prey system that governs the population of predators and preys in an ecological environment. This LVS is easier to solve than an S-system, and it is also easy to interpret outputs from this model in terms of genes regulation (activation or repression). It has also been shown that these two types of canonical systems are mathematically equivalent [15].

Finding the model parameters of an LVS, given the expression profiles of genes participating in a regulatory network, is an inverse problem. We follow the approach in [1,14] to avoid the time consuming process of solving differential equations directly. Each time trace of the expression level of a gene is approximated by using cubic spline interpolation or artificial neural network. These expanded traces are used to estimate derivatives in the LVS. Using a special structural feature of the system, the LVS is decoupled into several independent systems of linear equations. We propose to use multiple linear regressions to find significant factors in these linear systems, and thus solve the GRN problem.

2. Methods

2.1 Lotka-Volterra system

The Lokta-Voltera system is a system of first order second degree ODE. In equation (1), function $X_j(t)$ describes the expression level of the jth gene. The rate of change of this expression level is affected by the second degree slope function on the right hand side. The effect of the w_j term is primarily due to the time changes since solving $X_j^{'} = w_j X_j$ yields $X_j = Exp(w_j t)$; thus we focus on the summation term to look for gene regulatory effects. Because gene expression levels must be nonnegative, the effect of the kth gene on the rate of change of the jth gene is determined by the sign and magnitude of b_{jk}. When b_{jk} is positive (negative), we say that the kth gene activates (represses) the jth gene. The kth gene has no effect on the jth gene if $b_{jk} = 0$.

$$\frac{dX_j}{dt} = X_j(w_j + \sum_{k=1}^{n} b_{jk} X_k), j = 1,\ldots,n \quad (1)$$

The LVS is an extension to the predator-prey system in an ecological environment, where limited resources have governed the population of the predator species and the prey species. When $b_{jj.} = 0$ in equation (1), the LVS reduces to the predator-prey system [3].

2.2 Runge-Kutta algorithm

The LVS can be uniquely solved once an initial condition (expression level) is given: $\vec{X}(t_1) = (X_1(t_1),\cdots,X_n(t_1))$. This is based on a numerical integration procedure that consumes moderate CPU cycles when the coefficients (w_j, b_{jk}) in equation (1) are

known. A fourth order Runge-Kutta (RK4) algorithm is usually used to furnish such a procedure [11]. Let $\vec{F}$ denote the velocity vector on the right hand side of (1), with a mesh size of h, the RK4 algorithm advances the expression profile from the time event t to the time event $(t + h)$ as follows:

$$\vec{k}_1 = h \ \vec{F}(\vec{X}(t))$$

$$\vec{k}_2 = h \ \vec{F}(\vec{X}(t) + 0.5 * \vec{k}_1)$$

$$\vec{k}_3 = h \ \vec{F}(\vec{X}(t) + 0.5 * \vec{k}_2) \qquad (2)$$

$$\vec{k}_4 = h \ \vec{F}(\vec{X}(t) + \vec{k}_3)$$

$$\vec{X}(t + h) = \vec{X}(t) + (\vec{k}_1 + 2\vec{k}_2 + 2\vec{k}_3 + \vec{k}_4) / 6$$

The running time of RK4 is moderate. However, when the system coefficients (w_j, b_{jk}), also termed network coefficients in a GRN, are unknown and need to be discovered, the repetitive running of RK4 consumes almost 95% of the total operation time for solving a GRN problem [14].

2.3 Multiple linear regressions

A multiple linear regression (MLR) expresses a dependent variable Y as a linear combination of multiple independent variables x_j and a random noise as follows:

$$Y = \beta_0 + \beta_1 x_1 + \ldots + \beta_k x_k + \varepsilon \qquad (3)$$

Kennedy has a good discussion on MLR from the viewpoint of an econometrician [5]. The regression coefficients $\beta_0, \ldots, \beta_k$ are fixed but unknown, and they can be estimated using several estimators, given a set of observed data $(\vec{x}_i, y_i), i = 1, \ldots, n$. $\vec{x}_i = (x_{1i}, \cdots, x_{ki})$ is the input vector for the ith sample and y_i is the corresponding output value. Many well-known formulae have been developed to estimate the regression coefficients. For example, the ordinary least square (OLS) estimator can be used to compute coefficients so that the sum of squared errors $SSE = \sum_{i=1}^{n} (\widehat{y}_i - y_i)^2$ is minimized. Here $\widehat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_{1i} + \cdots + \hat{\beta}_k x_{ki}$ is computed with coefficients $\hat{\beta}_0, \hat{\beta}_1, \cdots, \hat{\beta}_k$ [5].

In the frequentist approach to MLR, it is assumed that the random noise ε in equation (3) can be realized repeatedly so that we have many realizations of the data. Across all realizations, the input vector $\vec{x}_i$ is unchanged and the output y_i is corrupted by a realized noise. With an estimator like the OLS estimator, each of these realizations yields an estimated value for the unknown coefficients. With all these estimates of the coefficients, interesting results about the coefficients can be inferred by hypothesis testing procedures in statistics [4]. A particularly interesting test for these coefficients is about the significant level of deviation from zero. In social sciences, this is called a significance test: how significant is the estimated coefficient $\hat{\beta}_j$ differently from zero? Most statistical packages (SPSS and Excel data analysis package) give not only an estimated coefficient but also a p-value to indicate the significance level. The p-value is a number between 0 and 1, and the smaller it is, the more significant of deviation from 0 a coefficient is. In general, a cut-off value of .05 is used in social science studies. So, when a coefficient is estimated with a value of 1000 with a p-value of .85, it generally causes less attraction to a researcher than another coefficient estimated as 10 with a p-value of .01.

2.4 The proposed method

2.4.1 Decoupling the system

The LVS is a system of ODE, thus each individual ODE must hold as well:

3

$$\frac{dX_j}{dt} = X_j(w_j + \sum_{k=1}^{n} b_{jk} X_k), t_1 \leq t \leq t_T \quad (4)$$

Particularly, equation (4) must hold for the discrete time events $t_1, t_2, \cdots, t_T$ where the expression levels are taken in a lab. If we only need to evaluate the ODE at these discrete events, there is no problem at all. Unfortunately, using RK4 to solve a single ODE like equation (4) often needs values of X_j at events other than the lab experimental events. This is due to the fact that effective mesh size in RK4 is normally smaller than the gap between two successive experimental events. Thus, if we want to solve equation (4), other methods should be developed to fill in values for X_k, $k \neq j$, at time different from experimental events.

Two methods are considered to fulfill this requirement. The first one is the cubic spline interpolation [11]. For a time trace $X_k(t), t = t_1, \cdots, t_T$ of the kth gene, a cubic spline is a piecewise polynomial function of the third degree that matches the given time trace and is continuous at these discrete time events up to the second derivatives. A cubic spline interpolates the given time trace by keeping the degree of polynomial low in each segment. This has effectively ruled out the rapid jumping phenomenon of the common interpolation formula.

The second method used to achieve the goal of interpolation is artificial neural network (ANN). ANN has been successfully used in many science and engineering problems to approximate a function, given samples of the function outputs [12]. In our case, the sample function outputs consist of the time trace, and a neural network is trained based on these sample outputs. The trained ANN is used to compute the gene expression level at any other time.

2.4.2 Converting the system

We do not intend to solve the ODE in (4) by RK4. Instead, we follow the conversion approach in [1,14] to find the

network coefficients. As explained before, equation (4) must hold at discrete time events $t_1, t_2, \cdots, t_T$, thus we have:

$$X_j'(t_1) = X_j(t_1)(w_j + \sum b_{jk} X_k(t_1))$$
$$\vdots \qquad\qquad (5)$$
$$X_j'(t_T) = X_j(t_T)(w_j + \sum b_{jk} X_k(t_T))$$

Equation (5) is a system of T linear equations in w_j and b_{jk} once we can estimate the derivatives at the left hand side. An obvious way to estimate these values is to use the given time trace as follows:

$$\frac{X_j(t_l + h) - X_j(t_l)}{h X_j(t_l)} = w_j + \sum b_{jk} X_k(t_l) \quad (6)$$

However, the mesh size $h = t_{l+1} - t_l$ given by the time trace may be too coarse, and finer estimate is needed. By considering the fact that

$$\frac{X_j'(t)}{X_j(t)} = \frac{d \ln(X_j(j))}{dt},$$

equation (5) can also be estimated as

$$\frac{\ln(X_j(t_l + h) / X_j(t_l))}{h} = w_j + \sum b_{jk} X_k(t_l)$$
$$(7)$$

We will call equation (6) a naïve conversion of (4) and (7) a log conversion of (4). It can be verified that the log conversion is more precise than the naïve conversion, since setting $u = X_j(t_l + h) / X_j(t_l) - 1$ in the Taylor series expansion

$$\ln(1 + u) = u - u^2 / 2 + u^3 / 3 + \ldots, -1 < u \leq 1,$$

we immediately recognize that the left hand side of equation (6) is simply a first order estimate of the left hand side of (7).

2.4.3 Solving MLR

After converting an ODE of (4) into a system of linear equations in (5) with the help of naïve conversion (6) or log conversion (7), the system can be solved by using MLR. Unlike [14], where network coefficients are extracted by examining the magnitude patterns of all coefficients, we will use patterns of p-values to select significant network coefficients.

3. Experiments and Results

To illustrate the proposed method, a synthetic LVS is generated and listed in equation (8). Five genes participate in this hypothetical genetic network. A time course with the initial data $\vec{X}(0) = (0.7, 0.12, 0.14, 0.16, 0.18)$ is developed by using the RK4 algorithm with a mesh size of 0.01. Every five mesh spaces constitute a hypothetic microarray experiment. Thus, there are 41 experiments occurring at the time events 0, 0.05, 0.10, …, 1.95 and 2.0. This time profile serves as the given expression data.

$$X_1'(t) = X_3(t) - X_5(t)$$
$$X_2'(t) = 2X_1(t)$$
$$X_3'(t) = -X_2(t) \qquad (8)$$
$$X_4'(t) = 2X_3(t) - X_5(t)$$
$$X_5'(t) = 2X_4(t)$$

3.1 Using the basic time profile

The chart of the basic time profile for the five genes can be depicted in Figure 1. First, we use equation (6) to estimate the derivatives at 40 time events:

$$\frac{X_j(.05(l+1)) - X_j(.05l)}{.05}, l = 0,1,\cdots,39$$

Thus, each ODE in (8) is converted into a linear system of 40 equations. Each linear system is solved with the MLR package in Excel with the estimated coefficients and p-values (in parentheses) reported in Table 2. Each row in the table represents the regression results of a linear system. The p-values of X_3 and X_5 on the first row are several orders smaller than the p-values of other coefficients. Therefore, their corresponding coefficients are deemed significantly different from 0, compared to the other coefficients. According to the sign of the coefficients, we conclude that X_3 activates X_1 and X_5 represses X_1. The other genes (X_1, X_2 and X_4) have no regulatory effects on the expression of X_1. On the other hand, if we choose the network coefficients based on the magnitude patterns as in [14], X_4 would have a positive effect on X_1. But, this contradicts the network effects shown in Table 1, which lists the true coefficients corresponding to the LVS in (8). Following the same reasoning to track patterns of p-values for the other variables, we have bold-faced significant factors in the table. One can see that the p-value method has misrecognized the positive effect of X_2 on X_2. However, since this coefficient (.0361) is very small compared to the other coefficients, we may discard this network connection. The same method does not detect any significant pattern of p-values on the fourth row, though the regulating genes (X_3 and X_5) have the smallest p-values on this row. Thus, the naïve conversion for this basic time profile has failed to detect the full network structure.
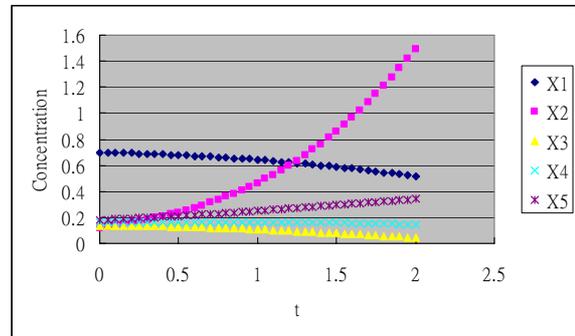
Figure 1. Basic time profile of five genes

When the log conversion in (7) is used

to convert LVS to linear systems, the regression results are reported in Table 3. One can see that, using the pattern detection of *p*-values, the network connections have been recognized correctly with signs according to the true values in Table 1. This shows the merits of the log conversion.

3.2 Using an expanded time profile

Since the default time gap of .05 provided by the basic time profile is too coarse to estimate the derivates, we expand the time profile by adding the RK4 outputs for every .01 change in the time domain. This has created an expanded profile of 201 events at 0, .01, .02, …, 1.99 and 2. The MLR results for the naïve and log conversions are respectively reported in Tables 4 and 5. One can observe two facts: first, the patterns of *p*-values can be more easily recognized; second, again the log conversion has provided more significant level with smaller *p*-value, and its estimated coefficients of significant genes are closer to the true values. This shows the merits of a finer time profile.

3.3 Expanded profile with cubic spline

Unfortunately, it is not always possible to expand the time profile by using RK4. Finding network coefficients is part of the job in a GRN problem.
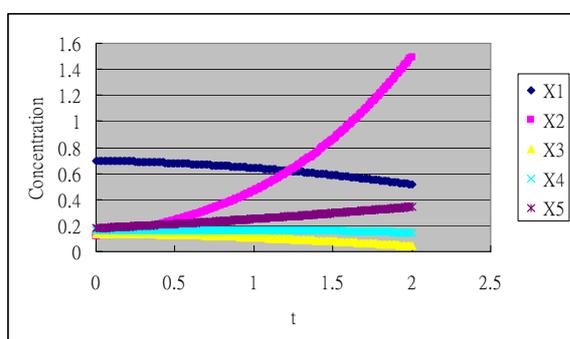
Figure 2. Expanded time profile with spline.

We use a cubic spline method to extend the basic time profile to an expanded profile

with the same fine resolution (.01) as above. This cubic spline matches a time trace at 41 time events: $X_j(.05l)$, $l = 0, …, 40$. Values of the spline at the step of .01 are plotted in Figure 2. The mean square errors between RK4 derived values and spline interpolated values of all five time traces are in the order of 1.0E-15. Using the spline expanded profile, we conduct the same experiments with the results reported in Table 6 (naïve conversion) and Table 7 (log conversion). These results are a little inferior to the results from the RK4 expanded profile. But, the pattern of *p*-values can still be easily recognized. Again, the log conversion provides better results than the naïve conversion.

3.4 Using ANN

The basic profile is expanded with an ANN using the NeuroSolutions package from NeuroDimension, Inc. The network structure is 1 x 4 x 1, where 4 is the number of hidden nodes. A hyperbolic tangent (tanh) is selected as the activation function, and 50000 epochs of network training are conducted. The mean square errors between RK4 derived values and ANN predicted values for all five traces are in the order of 1.0E-6. The expanded time profile is plotted in Figure 3, which is again similar to Figure 1. However, the results for the log conversion, listed in Table 8, show several misidentified network connections.
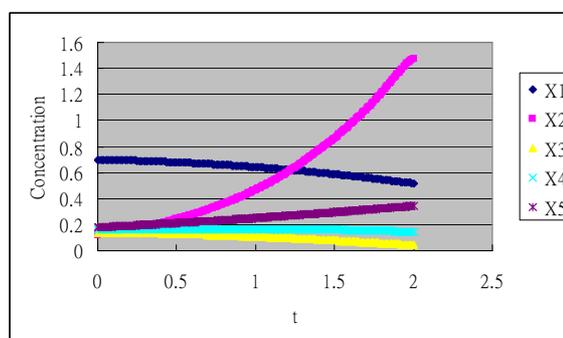
Figure 3. Expanded time profile with ANN.

4. Conclusions

In this study, we model gene regulatory network by using a Lokta-Voltera system. A synthetic time profile of gene expressions is developed by solving the LVS with a RK4 algorithm. This time profile is used in a reverse engineering sense to derive the network coefficients of the LVS. We first convert the LVS into several independent systems of linear equations, which are solved with the MLR technique in statistics. Two conversions are used to estimate the derivatives in LVS: the naïve conversion and the log conversion. We use the pattern of p-values, which indicate the significant level of deviation from 0 for regression coefficients, to infer network coefficients. From the experiments, we conclude the following facts:

   (1)   Pattern detection of p-values can be used to infer network coefficients in LVS;

   (2)   The log conversion provides better results than the naïve conversion;

   (3)   When finer time profile is available, the results also improve; and

   (4)   The cubic spline method is better than the ANN method to expand a time profile in our case.

In the future, it pays to investigate why the ANN does not provide efficient and effective function approximation for the time profile. Are there other methods that can be used to better approximate the time profile? How is the p-value method affected by the noise commonly encountered in real gene expression profiles?

5. Acknowledgements

6. References

[1] J.S. Almeida, E.O. Voit, "Neural-network-based parameter estimation in S-system models of biological networks", Genome Informatics, vol. 14, pp. 114-123, 2003.

[2] P. D'haeseleer, S. Liang, R. Somogyi, "Genetic network inference: from co-expression clustering to reverse engineering", Bioinformatics, vol. 16, pp. 707-726, 2000.

[3] M. Hirsch, S. Smale, differential equations, dynamical systems, and linear algebra, Academic Press, 1974.

[4] V.H. Hogg, E.A. Tannis, "Probability and Statistical Inference", fifth edition, Prentice-Hall, 1997.

[5] P. Kennedy, "A Guide to Econometrics, fifth edition", MIT Press, 2003.

[6] S. Kikuchi, D. Tominaga, M. Arita, K. Takahashi, M. Tomita, "Dynamic modeling of genetic networks using genetic algorithm and S-system", Bioinformatics, vol. 19, pp. 643-650, 2003.

[7] H. Matsuno, A. Doi, M. Nagasaki, S. Miyano, "Hybrid Petri net representation of gene regulatory network", Proceedings of pacific symposium on biocomputing, pp. 341-352, 2000.

[8] R. Mauricio, "Can ecology help genomics: the genome as ecosystem?", Genetica, vol. 123, pp. 205-209, 2005.

[9] N. Noman, H. Iba, "Revere engineering genetic networks using evolutionary computation", Genome informatics, vol. 16, pp. 205-214, 2005.

[10] B. Perrin, L. Ralaivola, A. Mazurie, S.

Bottani, J. Maller, F. d'Alche-Buc, "Gene networks inference using dynamic Bayesian networks", Bioinformatics, vol. 19, pp. ii138-ii148, 2003.

[11] W. Press, B. Flannery, S. Teukolsky, W. Vetterling, "Numerical recipes in Fortran: The art of scientific computing", 2nd ed. Cambridge, England, Cambridge University Press, 1992.

[12] J. Principe, N. Euliano, W. Lefebvre, "Neural and adaptive systems: fundamentals through simulations", John Wiley & Sons, 1999.

[13] M.A. Savageau, E.O. Voit, "Recasting nonlinear differential equations as S-systems: a canonical nonlinear form", Mathematical Biosciences, vol. 87, pp. 83-115, 1987.

[14] S. Veflingstad, J. Almeida, E. Voit, "Priming nonlinear searches for pathway identification", Theoretical biology and medical modeling, vol. 1, no. 8, pp. 1-14, 2004.

[15] E.O. Voit, M.A. Savageau, "Equivalence between S-systems and Volterra systems", Mathematical Biosciences, vol. 78, pp. 47-55, 1986.

Table 1. True network coefficients.

	Constant	X_1	X_2	X_3	X_4	X_5
X_1	0	0	0	1	0	-1
X_2	0	2	0	0	0	0
X_3	0	0	-1	0	0	0
X_4	0	0	0	2	0	-1
X_5	0	0	0	0	2	0

Table 2. MLR estimated coefficients and p-values (in parentheses) with the basic time profile and naïve conversion.

	Constant	X_1	X_2	X_3	X_4	X_5
X_1	.6342 (9.13E-6)	-1.0647 (8.68E-6)	.0434 (1.43E-7)	**.5938 (1.3E-9)**	2.7399 (4.41E-6)	**-2.5416 (1.21E-10)**
X_2	.3608 (3.44E-5)	**1.4533 (3.06E-13)**	**.0361 (2.51E-10)**	-.2463 (3.92E-6)	1.8344 (1.26E-6)	-1.0710 (5.17E-7)
X_3	-3.1158 (5.34E-4)	4.8433 (1.14E-3)	**-1.0470 (8.9E-23)**	3.2912 (7.59E-8)	-12.1861 (9.39E-4)	6.7569 (9.57E-4)
X_4	1.3910 (8.16E-6)	-2.3366 (7.67E-6)	.0935 (1.75E-7)	1.1183 (3.11E-8)	6.0143 (3.87E-6)	-4.3751 (2.5E-8)
X_5	.1443 (6.96E-6)	-.2519 (3.5E-6)	.0104 (3.62E-8)	-.0676 (1.86E-4)	**2.6835 (7.86E-23)**	-.3652 (1.34E-6)

Table 3. MLR estimated coefficients and *p*-values (in parentheses) with the basic time profile and log conversion.

	Constant	X_1	X_2	X_3	X_4	X_5
X_1	.4813	-.8126	.0328	**.7078**	2.0911	**-2.1773**
	(1.37E-5)	(1.19E-5)	(2.55E-7)	**(2.28E-14)**	(6.16E-6)	**(1.12E-11)**
X_2	.1869	**1.6889**	.0149	-.0981	.8145	-.4943
	(1.02E-5)	**(4.87E-25)**	(7.13E-9)	(5.92E-5)	(4.29E-6)	(9.46E-7)
X_3	.1397	-.2872	**-.9956**	.0749	.7977	-.4536
	(1.37E-20)	(1.94E-23)	**(4.09E-92)**	(3.07E-19)	(4.1E-25)	(1.91E-25)
X_4	.9355	-1.5813	.0631	**1.4354**	4.0656	**-3.2772**
	(1.44E-5)	(1.23E-5)	(3.3E-7)	**(7.55E-15)**	(6.47E-6)	**(5.45E-9)**
X_5	.1184	-.2028	.0088	-.0593	**2.5379**	-.3027
	(1.46E-5)	(1.03E-5)	(4.57E-8)	(1.48E-4)	**(3.78E-24)**	(2.58E-6)

Table 4. MLR estimated coefficients and *p*-values (in parentheses) with the expanded time profile and naïve conversion. Extra events come from RK4.

	Constant	X_1	X_2	X_3	X_4	X_5
X_1	.1210	-.2032	.0084	**.9224**	.5239	**-1.2948**
	(9.03E-18)	(7.14E-18)	(1.19E-25)	**(9.8E-186)**	(3.89E-19)	**(2.0E-103)**
X_2	.0630	**1.9058**	.0065	-.0431	.3258	-.1906
	(2.25E-15)	**(7.1E-206)**	(5.36E-39)	(2.79E-19)	(1.16E-21)	(2.26E-23)
X_3	-.7462	1.1725	**-1.0170**	.7525	-2.9705	1.6482
	(4.34E-11)	(4.7E-10)	**(2.5E-216)**	(5.19E-25)	(1.69E-10)	(1.78E-10)
X_4	.2642	-.4440	.0179	**1.8324**	1.1451	**-1.6427**
	(5.67E-18)	(4.3E-18)	(2.94E-25)	**(1.7E-178)**	(2.24E-19)	**(5.9E-65)**
X_5	.0259	-.0453	.0019	-.0117	**2.1238**	-.0659
	(1.65E-18)	(7.63E-20)	(8.39E-29)	(2.26E-12)	**(4.1E-224)**	(1.11E-21)

Table 5. MLR estimated coefficients and *p*-values (in parentheses) with the expanded time profile and log conversion. Extra events come from RK4.

	Constant	X_1	X_2	X_3	X_4	X_5
X_1	.0906	-.1530	.0062	**.9450**	.3946	**-1.2223**
	(4.64E-17)	(2.58E-17)	(1.42E-24)	**(1.1E-209)**	(1.5E-18)	**(2.1E-119)**
X_2	.0345	**1.9425**	.0028	-.0179	.1510	-.0921
	(8.79E-18)	**(8.0E-266)**	(4.31E-32)	(1.64E-14)	(2.06E-19)	(2.55E-22)
X_3	.0221	-.0474	**-.9996**	.0185	.1329	-.0758
	(1.05E-59)	(2.39E-76)	**(0)**	(1.57E-83)	(1.36E-85)	(1.73E-87)
X_4	.1757	-.2972	.0120	**1.8940**	.7657	**-1.4290**
	(5.47E-17)	(2.84E-17)	(4.18E-24)	**(3.6E-212)**	(1.76E-18)	**(7.73E-81)**
X_5	.0216	-.0371	.0016	-.0106	**2.0989**	-.0557
	(4.14E-17)	(8.86E-18)	(2.65E-28)	(6.99E-13)	**(1.0E-233)**	(2.14E-20)

Table 6. MLR estimated coefficients and *p*-values (in parentheses) with the expanded time profile and naïve conversion. Extra events come from spline interpolation.

	Constant	X_1	X_2	X_3	X_4	X_5
X_1	.1177	-.1976	.0082	**.9243**	.5103	**-1.2873**
	(1.8E-13)	(1.49E-13)	(6.84E-20)	**(2.9E-173)**	(1.44E-14)	**(1.8E-91)**
X_2	.2253	**1.6341**	.0152	-.1373	.9927	-.5613
	(7.59E-2)	**(5.48E-13)**	(2.67E-2)	(6.68E-2)	(5.80E-2)	(5.37E-2)
X_3	-.7303	1.1459	**-1.0161**	.7434	-2.9054	1.6120
	(6.09E-9)	(4.27E-8)	**(1.7E-206)**	(1.04E-20)	(1.85E-8)	(1.93E-8)
X_4	.2547	-.4281	.0174	**1.8378**	1.1061	**-1.6211**
	(7.03E-13)	(5.69E-13)	(1.43E-18)	**(7.7E-164)**	(5.85E-14)	**(6.91E-53)**
X_5	.0241	-.0423	.0018	-.0107	**2.1164**	-.0618
	(5.67E-11)	(7.43E-12)	(4.38E-18)	(4.31E-7)	**(1.7E-201)**	(4.29E-13)

Table 7. MLR estimated coefficients and *p*-values (in parentheses) with the expanded time profile and log conversion. Extra events come from spline interpolation.

	Constant	X_1	X_2	X_3	X_4	X_5
X_1	.0872	-.1475	.0061	**.9469**	.3810	**-1.2147**
	(1.94E-11)	(1.28E-11)	(6.06E-17)	**(2.2E-191)**	(1.65E-12)	**(1.0E-101)**
X_2	.1947	**1.6745**	.0114	-.1108	.8090	-.4579
	(1.20E-1)	**(8.9E-14)**	(9.22E-2)	(1.33E-1)	(1.17E-1)	(1.1E-1)
X_3	.0378	-.0737	**-.9987**	.0096	.1971	-.1115
	(4.03E-1)	(3.29E-1)	**(2.3E-287)**	(7.19E-1)	(2.90E-1)	(2.81E-1)
X_4	.1662	-.2813	.0115	**1.8993**	.7269	**-1.4074**
	(8.0E-10)	(5.32E-10)	(1.97E-14)	**(1.6E-187)**	(9.12E-11)	**(1.04E-59)**
X_5	.0199	-.0341	.0015	-.0096	**2.0916**	-.0516
	(4.52E-9)	(1.82E-9)	(4.3E-16)	(1.13E-6)	**(2.3E-206)**	(4.79E-11)

Table 8. MLR estimated coefficients and *p*-values (in parentheses) with the expanded time profile and log conversion. Extra events come from ANN approximation.

	Constant	X_1	X_2	X_3	X_4	X_5
X_1	19.7198	-36.5105	-.6664	**38.5948**	17.0428	-12.4503
	(1.22E-6)	(1.63E-7)	(2.37E-4)	**(8.05E-19)**	(1.26E-1)	(4.99E-2)
X_2	-54.6666	102.2523	7.2519	**-196.2190**	127.0465	-52.0429
	(6.48E-4)	(1.94E-4)	(9.32E-20)	**(1.09E-26)**	(4.69E-3)	(4.08E-2)
X_3	104.4663	-193.0210	-4.2156	**191.701**	101.1365	-66.2539
	(4.38E-9)	(2.81E-10)	(1.26E-7)	**(3.55E-23)**	(3.59E-2)	(1.59E-2)
X_4	31.8514	-56.7898	.5310	22.4063	85.1122	-48.9753
	(1.55E-9)	(3.32E-10)	(2.04E-2)	(1.26E-5)	(8.7E-9)	(6.13E-9)
X_5	2.2575	-1.5745	**3.9801**	**-82.1367**	142.4317	-70.3187
	(5.35E-1)	(8.0E-1)	**(1.17E-60)**	**(1.36E-56)**	(5.05E-31)	(1.98E-25)