

Implementation of Delay Differential Equations in NONMEM

Wojciech Krzyzanski¹, Bob Bauer²

¹University at Buffalo, Buffalo, USA, ²ICON, Gaithersburg, USA.

Introduction

Models involving delay differential equations (DDEs) are increasingly popular in analysis of pharmacokinetic and pharmacodynamics data. A distinct feature of DDEs is a history that needs to be evaluated in order to determine the change of the state at a current time. This requires additional computational time that makes numerical DDE solvers run longer compared to analogous solvers for ordinary differential equations (ODEs). Another distinct feature that is poorly handled by available DDE solvers is presence of bolus dosing events that introduces discontinuous input into DDE models and increases computational time. A technique called method of steps (MOS) has been introduced that transforms any system of DDEs into a system of ODEs [1]. A feature that impacts numerical performance of MOS is high dimensionality of ODE systems. Another practical issue is complexity of implementation of MOS into PKPD software. A program DDEXPAND has been developed for coding DDE based models in NONMEM [2]. DDEXPAND augments the base DDE model equations in a NONMEM control stream with additional MOS differential equations, and the expanded control stream can run using any of the available ODE solvers in NONMEM.

Objectives

The objective of this work was to assess numerical performance of MOS implemented in NONMEM by DDEXPAND compared to well established DDE solver dde23 implemented in MATLAB [3].

Methods

Three previously published DDE models were selected for testing the MOS: delayed logistic growth (LOGISTIC), rheumatoid arthritis (RA), and tumor growth inhibition (TGI) [4]. All models were implemented in NONMEM 7.4 by means of DDEXPAND with ADVAN 13 ODE solver. For each model 100 predictions were simulated for one subject using published typical values without residual variability. The CPU time was recorded for each simulation. Analogously, each model was coded in MATLAB R2017b. The model DDEs were solved by dde23 for an individual subject at 100 observation times identical with NONMEM models. The CPU time was recorded for each simulation. Additionally, each model was implemented in MATLAB using MOS equations based on NONMEM control streams generated by DDEXPAND. The ode45 ODE solver was used to solve the MOS model equations. Similarly, 100 predictions were simulated and CPU times were recorded. The mean of 10 CPU times was used as a measure of performance. The same ATOL=1.0E-06 and RTOL=1.0E-6 values were used for all solvers. Maximum absolute and relative differences between NONMEM and MATLAB solutions were used as metrics of MOS accuracy. Both NONMEM and MATLAB were run on a PC computer with Intel Fortran compiler v 11.1.

Results

Delayed Logistic Growth Model

$$\frac{dy}{dt} = k_g \left(1 - \frac{y(t-T)}{y_{ss}} \right) y(t) \quad \text{for } t > 0 \quad y(t) = \varphi(t) \quad \text{for } t \in (-T, 0]$$

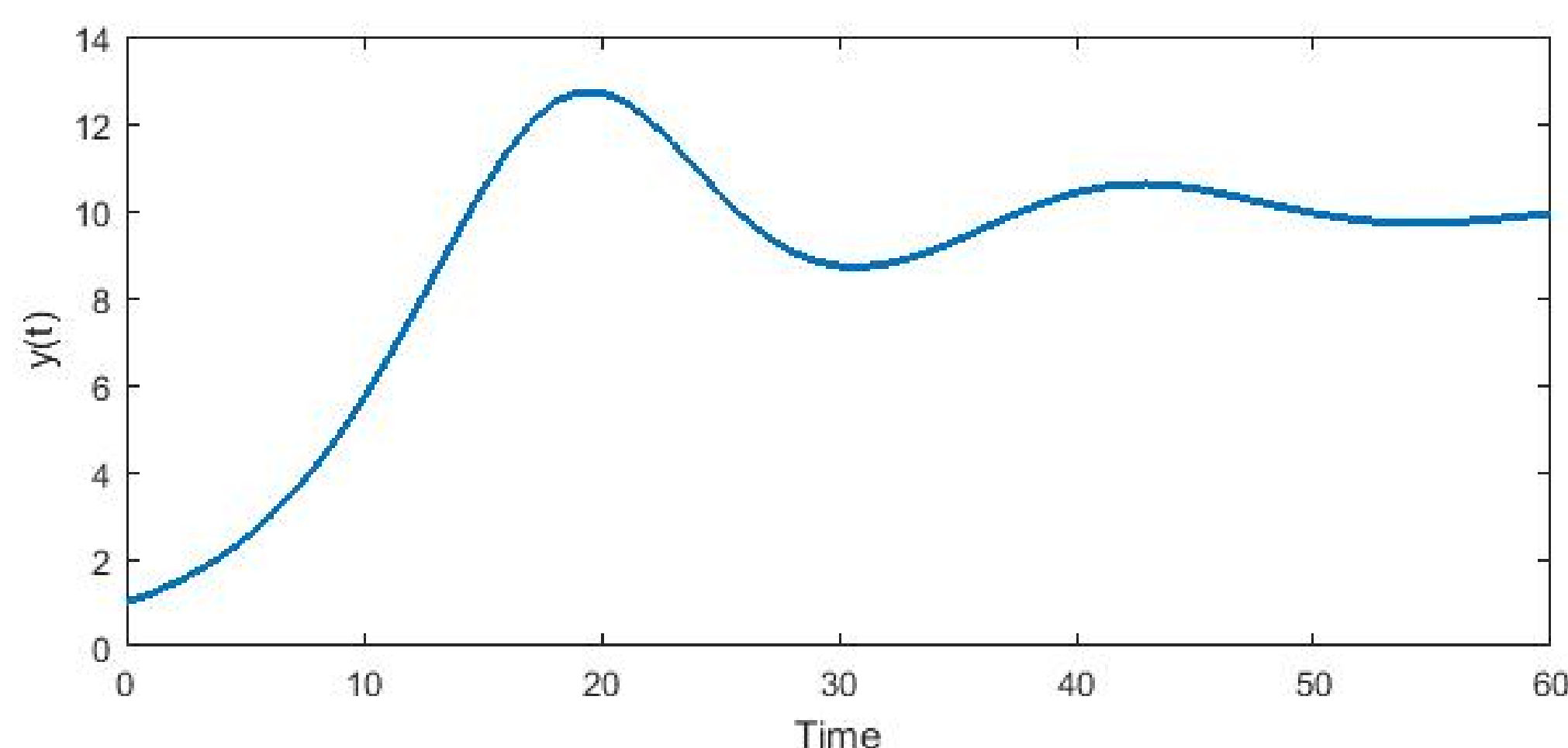


Fig. 1. Simulation of delayed logistic growth curve with $k_g = 0.2$, $y_{ss} = 10$, $T = 5$, and $\varphi(t) = 1$. Number of steps 13.

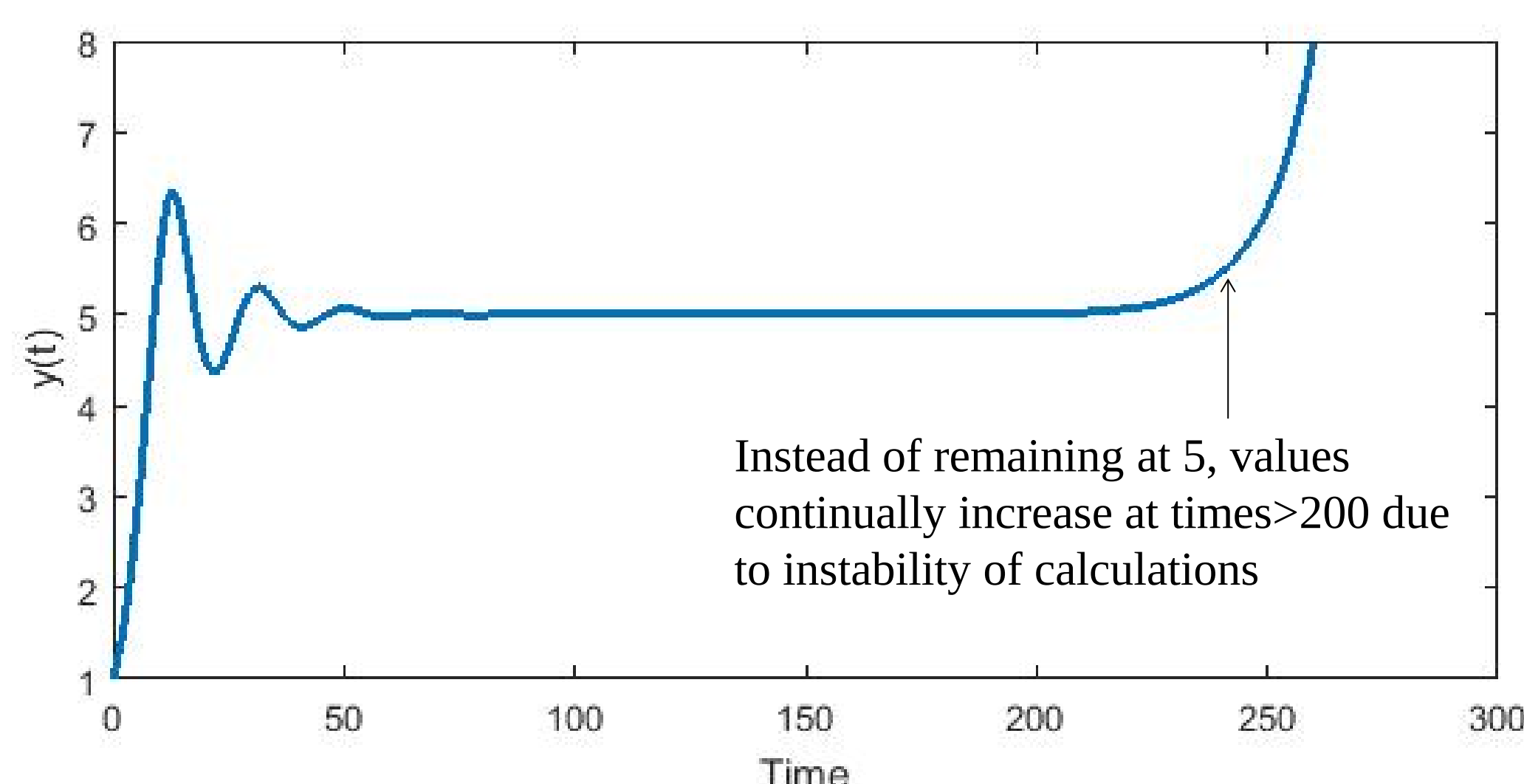


Fig. 2. Instability of MOS. If the number of steps becomes large numerical solutions of MOS become unstable. Simulation of delayed logistic growth curve with $k_g = 0.25$, $y_{ss} = 5$, $T = 4$, and $\varphi(t) = 1$. Number of steps 66.

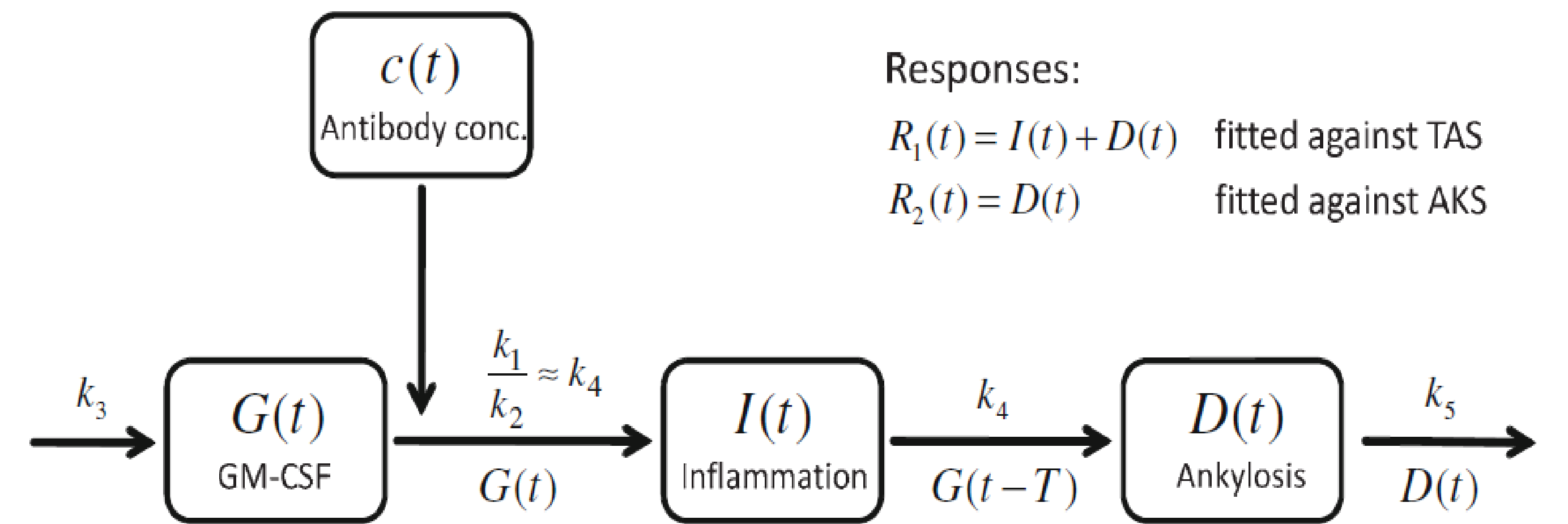
CPU ddexp (mean±SD)	CPU dde23 (mean±SD)
0.0591±0.0097	0.05830±0.0183

AbsErr	RelErr
2.27E-03	2.28E-04

$$\text{AbsErr} = \max_{0 \leq t \leq 60} |y_{ddexp}(t) - y_{dde23}(t)|$$

$$\text{RelErr} = \max_{0 \leq t \leq 60} \left| \frac{y_{ddexp}(t)}{y_{dde23}(t)} - 1 \right|$$

Rheumatoid Arthritis Model



CPU ddexp (mean±SD)	CPU dde23 (mean±SD)
0.08890±0.0196	0.7516±0.0537

Dose	AbsErr	RelErr
0	1.33E-05	3.30E-05
1	4.08E-05	3.79E-05
10	1.14E-05	4.99E-05
100	8.36E-06	6.04E-05

$$\text{AbsErr} = \max_{0 \leq t \leq 25} |AKS_{ddexp}(t) - AKS_{dde23}(t)|$$

$$\text{RelErr} = \max_{0 \leq t \leq 60} \left| \frac{AKS_{ddexp}(t)}{AKS_{dde23}(t)} - 1 \right|$$

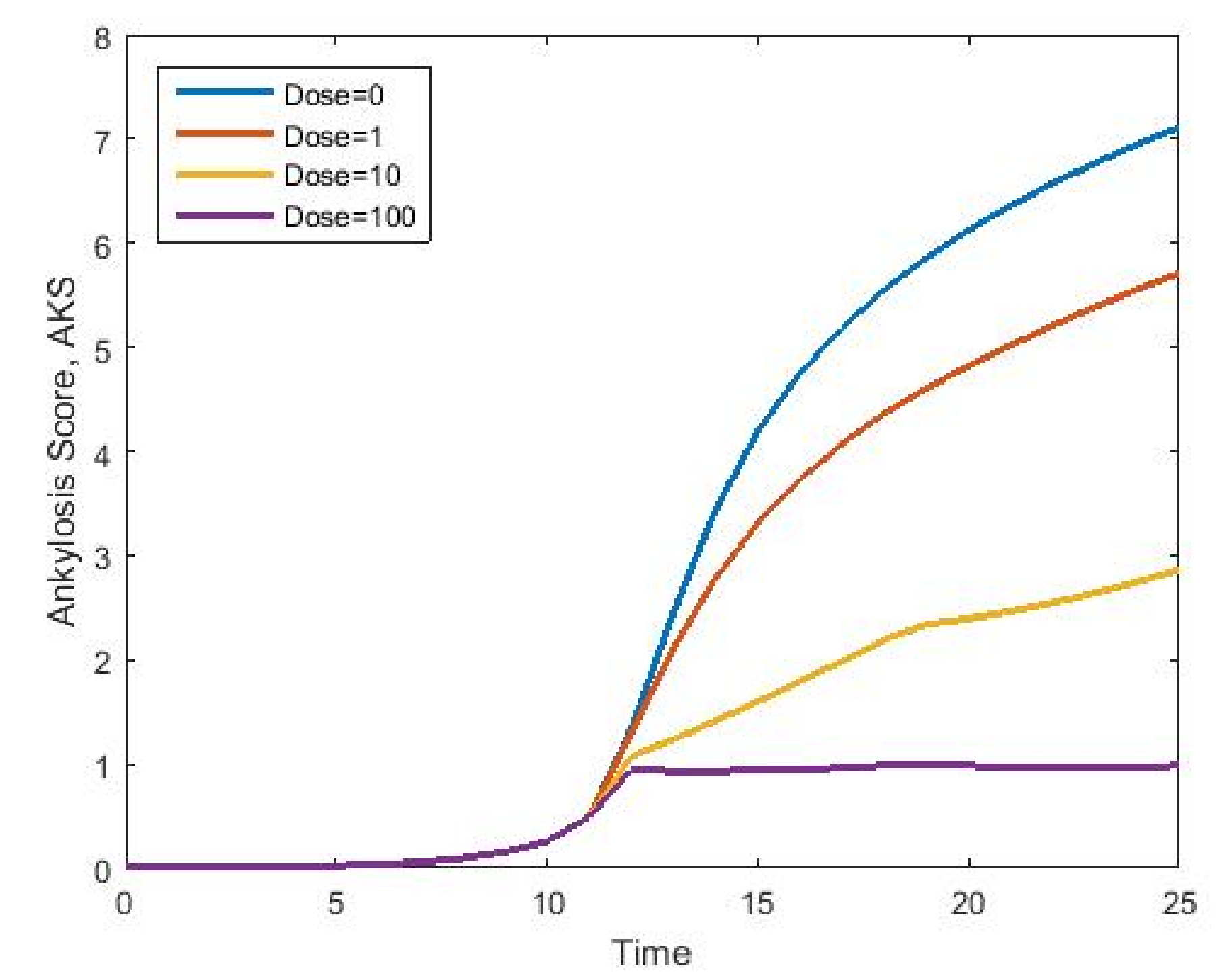
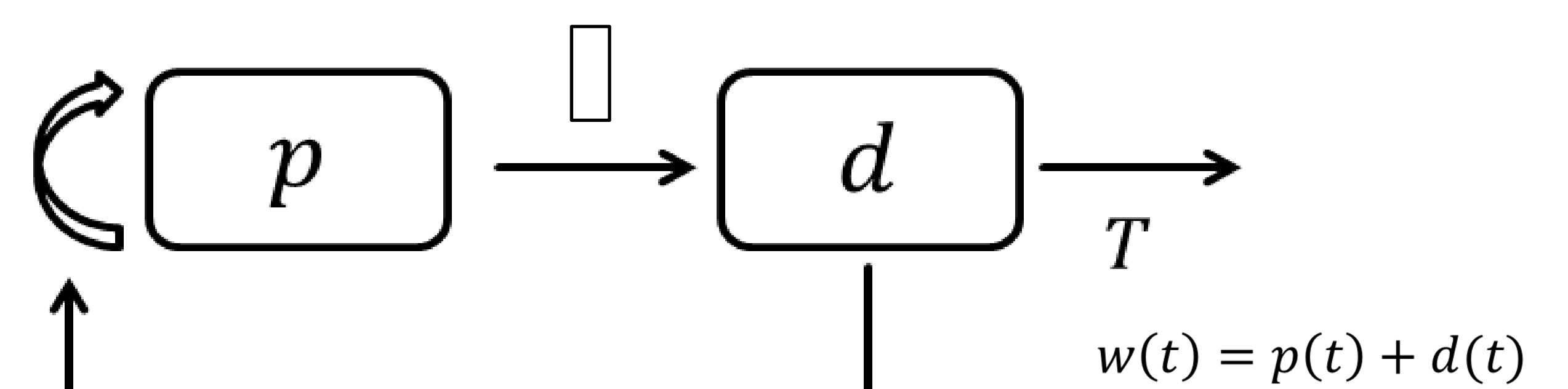


Fig. 3. Simulated time courses of AKS following bolus injections specified doses of 22E9 at times $t = 1, 8, 15$. Parameter values are as in [4]. Number of steps 2.

Tumor Growth Inhibition Model



CPU ddexp (mean±SD)	CPU dde23 (mean±SD)
0.09360±0.0222	6.644±0.0511

Dose	AbsErr	RelErr
0	2.34E-04	3.28E-04
100	1.46E-04	3.1E-04

$$\text{AbsErr} = \max_{0 \leq t \leq 25} |w_{ddexp}(t) - w_{dde23}(t)|$$

$$\text{RelErr} = \max_{0 \leq t \leq 25} \left| \frac{w_{ddexp}(t)}{w_{dde23}(t)} - 1 \right|$$

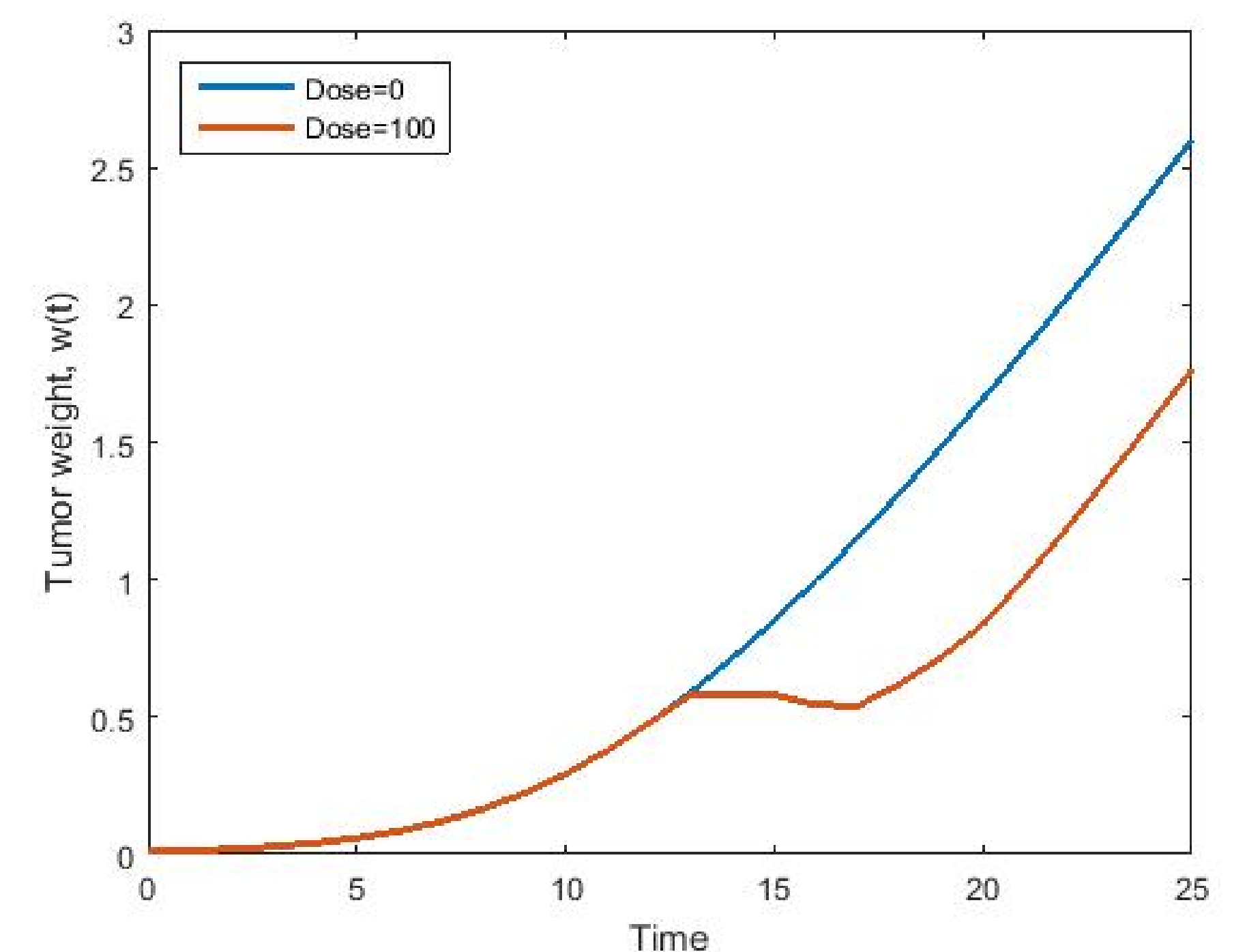


Fig. 4. Simulated time courses of tumor weight with $\lambda_0 = 0.197$, $\lambda_1 = 0.244$, $T = 3.3$, $k_{pot} = 0.007$, and $w_0 = 0.009$. The dose = 100 was orally administered at times $t = 12, 13, 14, 15$ and 16. Number of steps 2.

Conclusions

- For LOGISTIC model NONMEM is equally fast as MATLAB, but it is one order less accurate than MATLAB (accuracy was improved with smaller ATOL and RTOL without significantly affecting computation time).
- If the number of steps becomes large, MOS may become unstable.
- For RA model with multiple bolus injections NONMEM is 8.5 times faster than MATLAB and equally accurate.
- For TGI model with multiple drug administrations NONMEM is 71 times faster than MATLAB and equally accurate.

References

- Smith H (2011) An introduction to delay differential equations with applications to the life sciences. Springer, New York
- Bauer B, Krzyzanski W, DDEXPAND interface for coding delay differential equations based models in NONMEM PAGE 26 (2017) Abstr 7312 [www.page-meeting.org/?abstract=7312]. Source files can be found at <https://nonmem.iconplc.com/nonmem/dde>
- Shampine LF, Thompson S (2001) Solving DDEs in MATLAB. Appl Numer Math 37:441–458
- Koch G, Krzyzanski W, Perez-Ruixo JJ, Schropp J (2014) Modeling of delays in PKPD: classical approaches and a tutorial for delay differential equations. J. Pharmacokin. Pharmacodyn. 41:291–318