SBML ODE SOLVER LIBRARY: EXTENSIONS FOR INVERSE ANALYSIS

James Lu[1], *Stefan Müller*[1], *Rainer Machné*[2], *Christoph Flamm*[2]

[1]Johann Radon Institute for Computational and Applied Mathematics (RICAM),
Austrian Academy of Sciences,
Altenbergerstrasse 69, 4040 Linz, Austria
james.lu@oeaw.ac.at, stefan.mueller@oeaw.ac.at
[2]Institute of Theoretical Chemistry, University of Vienna,
Währingerstrasse 17, 1090 Wien, Austria
raim@tbi.univie.ac.at, xtof@tbi.univie.ac.at

ABSTRACT

The SBML ODE Solver Library (SOSlib) [1] is a C/C++ programming library for the symbolic and numerical analysis of ODE systems derived from biochemical reaction networks encoded in the Systems Biology Markup Language (SBML) [2]. It is written in ANSI/ISO C and distributed under the terms of the GNU Lesser General Public License (LGPL).

Recent efforts in the development of SOSlib have been focused on extensions that allow one to perform not only the *forward analysis* but also the *inverse analysis* of biochemical models. In particular, SOSlib has been extended with forward and adjoint capabilities to enable the identification of model parameters and initial conditions from (noisy) experimental data, measured either continuously or at discrete time points. Via on-the-fly compilation of right-hand-side functions and Jacobian routines, a significant speed-up in numerical integration has been achieved.

1. SIMULATION AND SENSITIVITY ANALYSES

We denote the underlying ODE system and initial condition as, respectively,

$$\begin{aligned} \dot{x}(t) &= f(x,\alpha), \\ x(0) &= x_0, \end{aligned} \tag{1}$$

where $x \in \mathbb{R}^n$ is the state variable, $\alpha \in \mathbb{R}^m$ the parameters and $f(x,\alpha) : \mathbb{R}^{n+m} \to \mathbb{R}^n$ the parameter-dependent vector field. To allow for the sensitivity analysis of (1), we assume the differentiability of $f(x,\alpha)$ with respect to both x and α.

In studying many biological models, one would like to not only obtain the solution $x(t)$ for a given set of nominal parameter values but also to examine its parametric dependence. This can be computationally studied by solving the forward sensitivty equations as discussed in Section 1.1. For applications such as parameter identification and optimization of biological systems, one is interested in computing the parametric dependence not for the whole time-series but only for certain *functionals* that map solutions to real numbers. For these applications, the adjoint approach to sensitivity analysis as discussed in Section 1.2 is the preferred method in terms of the computational efficiency.

1.1. Forward Sensitivity Analysis

The simulation of the ODE system (1) can be thought of as applying an operator $\mathcal{F}$ which takes as input the initial condition and parameter values, mapping it to the ODE solution. That is, we have $\mathcal{F} : (x_0,\alpha) \in \mathbb{R}^{n+m} \to x(t) \in C^1([0,T],\mathbb{R}^n)$, where C^1 denotes the space of continuously differentiable functions. One can then consider differentiations of the operator $\mathcal{F}$ either at an algorithmic level, or at a mathematical level. For the former, one would symbolically "differentiate" the steps taken in the chosen numerical algorithm in going from the input data, x_0, α, to the N-point numerical approximation over the requested time interval, $\{x(t_0), \cdots, x(t_N)\}$. Such an approach is known as *automatic differentiation* (AD) [3].

In our work, we take the forward sensitivity equations approach whereby one formally differentiates the operator $\mathcal{F}$ with respect to the initial concentrations, x_0, and the parameter values, α. In this case, the differential equations are first derived and then arbitrary numerical methods can be applied to solve the resulting system of equations. This is the approach that we have implemented in SOSlib and discussed in Sections 1.1.1 and 1.1.2. In Section 1.1.3, we discuss an application of the forward sensitivity solver, namely for computing the Fisher Information Matrix from which a lower-bound for the standard deviation in the estimated parameters can be derived [4].

1.1.1. Initial condition sensitivity

First, we consider variations in the solution arising from variations in the initial conditions. One supposes that the parameters α are fixed, and differentiates the original ODE system with respect to the initial conditions, which is then reflected in the initialization for the sensitivity variables. If we denote s^i as the set of sensitivity variables (of dimension n) corresponding to the perturbation of the system (1) with respect to the ith component of x_0, then one obtains the following equations for the $n \times n$ sensitivity system $\{s^1(t), \cdots, s^n(t)\}$:

$$\begin{aligned} \dot{s}^i(t) &= f_x(x(t),\alpha)s^i(t), \\ (s^i)_j(0) &= \delta_{ij}, \end{aligned}$$

where the notation $(s^i)_j$ denotes the j-th component of s^i, f_x is the Jacobian matrix and δ_{ij} is the Kronecker delta defined as $\delta_{ii} = 1$ and $\delta_{ij} = 0$ otherwise (for $1 \leq i, j \leq n$).

1.1.2. Parameter sensitivity

Next, we consider variations in the solution arising from variation in the parameters. Since no perturbation in the initial state is introduced, it is easy to see that the parametric sensitivity variables have the homogeneous initial condition. One thus obtains the following $n \times m$ linear ODE system for $\{s^1(t), \cdots, s^m(t)\}$:

$$\begin{aligned} \dot{s}^i(t) &= f_x(x(t),\alpha)s^i(t) + f_{\alpha_i}(x(t),\alpha), \\ s^i(0) &= 0. \end{aligned}$$

Using the symbolic differentiation capability of SOSlib, the expressions f_x, f_{α_i} are computed and passed to be called from the CVODES solver.

1.1.3. Application: Fisher Information Matrix

Let us consider the problem of estimating m parameters from time-course data. For each data-point t_i, let us denote $S(t_i)$ as the $n \times m$ matrix of sensitivity solutions:

$$S(t_i) = \begin{pmatrix} s_1^1(t_i) & \cdots & s_1^m(t_i) \\ \vdots & \ddots & \vdots \\ s_n^1(t_i) & \cdots & s_n^m(t_i) \end{pmatrix}.$$

If we denote the covariance matrix of (discrete) measurement errors as V, then the Fisher Information Matrix (F) is given by the following formula:

$$F = \sum_{t_i=1}^{N} S(t_i)^T V^{-1} S(t_i).$$

Thus, using the forward sensitivity solver, F can be computed by simply summing matrix products over experimental time points. For deriving parameter confidence intervals from F, refer to [4].

1.2. Adjoint Sensitivity Analysis

Given a functional of interest, $J : C^1([0,T],\mathbb{R}^n) \to \mathbb{R}$, we consider the following Lagrangian,

$$L(x,\psi) = J(x) + \langle \psi, \dot{x} - f(x,\alpha)\rangle_{L_2},$$

where $\psi(t) \in C^*([0,T],\mathbb{R}^n)$ is the associated adjoint variable (of bounded variation) in the dual space of continuous functions, serving as Lagrange multiplier to the ODE constraint $\dot{x} - f(x,\alpha) = 0$. Integration by parts of the above gives

$$\begin{aligned} L(x,\psi) = J(x) + \langle -\dot{\psi}, x\rangle_{L_2} - \langle \psi, f(x,\alpha)\rangle_{L_2} \\ + \psi(T)x(T) - \psi(0)x(0). \end{aligned} \quad (2)$$

The equations for the adjoint variable are obtained by considering the variational equations $\delta L(x,\psi;\delta x) = 0$, for all variations: $\delta x \in C^1([0,T],\mathbb{R}^n)$, $\delta x(0) = 0$. In Sections 1.2.1 and 1.2.2, we show the adjoint ODE systems for cases where the objective corresponds to either continuously or discretely measured experimental data respectively. For a discussion on the adjoint equations and its numerical solution in the general context of differential-algebraic equations (DAEs), refer to [5].

After the adjoint system is solved, the objective gradients with respect to the parameters and initial conditions are simply obtained as (refer to eqn. (2), noting the implicit dependency of $x(t)$ on x_0 and α):

$$\begin{aligned} \frac{dJ(x(\alpha,x_0))}{d\alpha_j} &= \frac{\partial L}{\partial \alpha_j} = -\langle \psi, f_{\alpha_j}(x,\alpha)\rangle_{L_2} \\ \frac{dJ(x(\alpha,x_0))}{d(x_0)_j} &= \frac{\partial L}{\partial (x_0)_j} = -\psi(0)_j. \end{aligned} \quad (3)$$

We remark that the adjoint approach to computing the objective gradient is especially attractive for biological systems of high parameter dimensions. In particular, the dimension of the adjoint variable is the same as that of the state, independent of the number of parameters. After this adjoint system has been numerically integrated, equation (3) shows that gradients of the given objective can then be computed by simply taking inner products over the time domain, or evaluating the adjoint variable at time $t = 0$. Thus, gradient calculations can be done essentially at constant time, independent of the number of parameters present in the model.

1.2.1. Continuous data

Without the loss of generality but for the simplicity of presentation, in what follows we assume a specific form of the objective function. Namely, we consider parameter identification applications where one tries to minimize objectives measuring the data mis-match. That is, if no regularization is used, such an objective may take the form:

$$J_{cont}(x) = \frac{1}{2}\int_0^T (x(t) - x_{data}(t))^2 dt. \quad (4)$$

where $x_{data}(t) \in C^1([0,T],\mathbb{R}^n)$ is some given experimental time-series.

From the Lagrangian expression in (2), setting $\delta L(x,\psi;\delta x) = 0$ gives rise to the following terminal-value problem for the adjoint variable, $\psi(t)$:

$$\begin{aligned} \psi(T) &= 0, \\ \dot{\psi}(t) &= -f_x(x(t),\alpha)^T\psi(t) \\ &\quad +(x(t) - x_{data}(t)). \end{aligned} \quad (5)$$

Once the expression for the objective has been provided to SOSlib and the data $x_{data}(t)$ is read in, the system (5) can again be numerically integrated (backwards in time) using the adjoint solver provided by CVODES.

1.2.2. *Discrete data*

Here, we consider objectives of the following form:

$$J_{disc}(x) = \frac{1}{2}\sum_{k=1}^{N}(x(t_k) - x_{data}(t_k))^2, \qquad (6)$$

consisting of the sum of the data mis-match over the (discrete) time points, $\{t_1, \cdots, t_N\}$. The objective (6) may be rewritten as:

$$J_{disc}(x) = \frac{1}{2}\sum_{k=1}^{N}\int_0^T \delta(t-t_k)(x(t) - x_{data}(t))^2 dt, \quad (7)$$

where $\delta(t - t_k)$ is the delta distribution with the sifting property that for all continuous functions $g(t)$,

$$\int_{-\infty}^{\infty} g(t)\delta(t-t_k)dt = g(t_k).$$

Now that the objective (1.2.2) is of the integral form, one might attempt to write down the adjoint system analogous to (5):

$$\begin{aligned} \psi(T) &= 0, \\ \dot{\psi}(t) &= -f_x(x(t), \alpha)^T \psi(t) \\ &\quad + \sum_{i=k}^{N} \delta(t-t_k)(x(t) - x_{data}(t_k)). \end{aligned}$$

The above ODE system only has meaning in the sense of distributions and no ODE solver can be applied directly without taking special care at the data time points, $\{t_i\}$. Instead, one can solve it by treating it as a concatenation of piecewise continuous trajectories. More specifically, with the terminal condition being $\psi(T) = 0$, we integrate over time intervals in between the set of data time points and introduce jumps at the times when data is given:

$$\begin{aligned} \text{FOR} \quad : \quad & k = N, N-1, \cdots, 1 \\ & \dot{\psi}(t) = -f_x(x(t), \alpha)^T \psi(t), t \in [t_k, t_{k+1}) \\ & \psi(t_k^-) = \psi(t_k^+) - (x(t) - x_{data}(t_k)). \end{aligned}$$

Thus, the adjoint profile can be computed by providing start- and stop-time points $\{t_i\}$ to the CVODES adjoint solver to integrate it piecewise and adding to the adjoint variable in between the integration calls.

2. COMPILATION

For parameter identification and optimal control applications, the ODE and sensitivity solvers typically need to be called many times. In order to study systems with high dimensional parameter space within reasonable compute time, it is important to be able to evaluate the right-hand-side functions and Jacobians of the ODE systems efficiently.

Motivated by a need to speed up the solvers, we have implemented two different versions of on-the-fly compilation of these functions. First, we take use of libSBMLs abstract-syntax-tree representation of model equations to directly construct machine code for all equations of the model. As this approach is highly platform-specific (currently, we provide 32 and 64 bit architecture machine code for both Windows and Unix systems), SOSlib also allows for the conversion from the vector-field and associated functions to C source code and makes use of a preinstalled C/C++ compiler (e.g. `gcc`). Both approaches result in around an order of magnitude decrease in the compute time for some test cases; see Section 3.2.

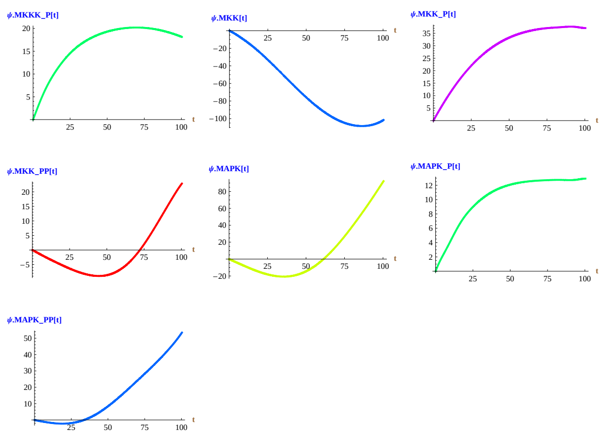

Figure 1. Adjoint solution profile for parameter estimation: using continuous data

3. NUMERICAL DEMONSTRATIONS

Here, we consider parameter idenfication examples formulated as finding the minimizer of the data mis-match. In Section 3.1, we illustrate the difference in the adjoint solution profiles using continuous and discrete data. In Section 3.2, we show the objective convergence using an interior-point optimization solver and demonstrate the speed-up gained by model compilation.

3.1. Adjoint profiles

Here we examine adjoint solutions at the first step of the parameter identification procedure. Figures 1 and 2 illustrate the adjoint solution profiles corresponding to using continuous and discrete data, for a simple oscillatory model of a signaling cascade taken from the BioModels database [1]. In both cases, we use artificial data obtained by simulating the model at its nominal parameters. In Figure 2, one can easily spot the jumps in the adjoint profiles at the 10 data points. Despite this, one can observe some similarity in the general shapes of the profiles given in Figures 1 and 2. In fact, as the number of (discretely) sampled data points increases, one would expect the adjoint solutions to converge in the L_1 norm.

3.2. Convergence and speed

Here we consider the identification of 36 parameters in the 3-gene model as used in [6]. In particular, we use noiseless, artificial data corresponding to the original parameters and start the parameter identification procedure from

[1]`http://www.ebi.ac.uk/biomodels/`

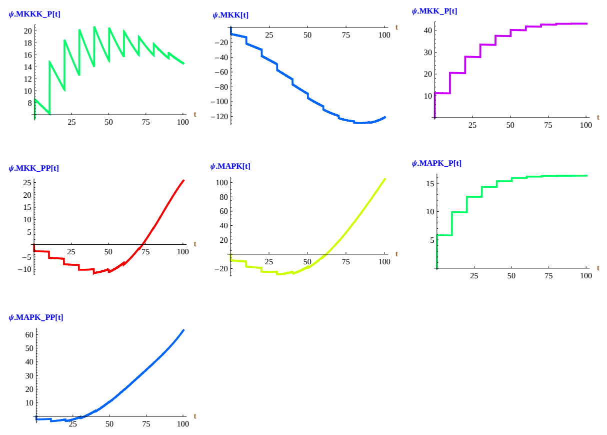

Figure 2. Adjoint solution profile for parameter estimation: using discrete data

parameter values being an order of magnitude smaller than the true ones. To carry out the minimization of the above objective we employ IpOpt [7], an interior-point (local) optimization algorithm. The values for the objective and gradient are provided by the forward and adjoint solvers of SOSlib, respectively.

The convergence in the objective is shown in Figure 3. We observe a 6 orders of magnitude decrease in the objective over 500 optimization iterations using IpOpt [7]. Table 1 gives the number of function evaluation calls to SOSlib as well as the CPU time taken in the numerical integrations. We see that around 1200 forward ODE and 500 adjoint integrations were carried out in the optimization process. The observed difference in the number of objective calls between the compiled and non-compiled results is due to small numerical discrepancies. If the right-hand side and Jacobian functions are not compiled, the time taken for these calculations take 21.54 seconds; when these functions are compiled with `gcc` only 2.64 seconds are needed, thereby achieving nearly an order of magnitude decrease in the computing time.

Table 1: IpOpt calls to SOSlib

	No compilation	`gcc` compilation
# obj. eval.	1222	1251
# grad. eval	500	500
CPU: SOSlib	**21.54 sec.**	**2.64 sec.**

4. CONCLUSIONS

We have demonstrated extensions to SOSlib that allow one to perform inverse analyses of biological models efficiently. In combination with regularization methods [8], these tools enable one to tackle *ill-posed* parameter identification problems that arise in systems biology.

5. ACKNOWLEDGMENTS

We gratefully acknowledge financial support by the Vienna Science and Technology Fund (WWTF), Project Number MA05.

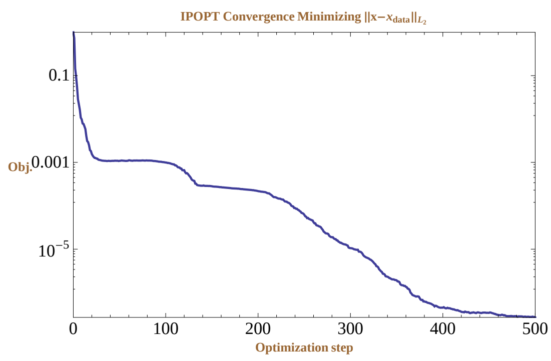

Figure 3. Convergence of objective using the interior point solver, IpOpt

6. REFERENCES

[1] R. Machné, A. Finney, S. Müller, J. Lu, S. Widder, and C. Flamm, "The SBML ODE Solver Library: a native API for symbolic and fast numerical analysis of reaction networks.," *Bioinformatics*, Mar 9 2006.

[2] A. Finney and M. Hucka, "Systems biology markup language: Level 2 and beyond," *Biochem Soc Trans*, vol. 31, no. Pt 6, pp. 1472–1473, Dec 2003.

[3] A. Griewank, "A mathematical view of automatic differentiation," in *Acta Numerica*, vol. 12, pp. 321–398. Cambridge University Press, 2003.

[4] A. Kremling, S. Fischer, K. Gadkar, F. J. Doyle, T. Sauter, E. Bullinger, F. Allgower, and E. D. Gilles, "A benchmark for methods in reverse engineering and model discrimination: problem formulation and solutions.," *Genome Res*, vol. 14, no. 9, pp. 1773–85, 2004.

[5] Y. Cao, S. Li, L. Petzold, and R. Serban, "Adjoint sensitivity analysis for differential-algebraic equations: the adjoint DAE system and its numerical solution," *SIAM J. Sci. Comput.*, vol. 24, no. 3, pp. 1076–1089 (electronic), 2002.

[6] C. G. Moles, P. Mendes, and J. R. Banga, "Parameter estimation in biochemical pathways: a comparison of global optimization methods.," *Genome Res*, vol. 13, no. 11, pp. 2467–74, 2003.

[7] A. Waechter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Math. Program.*, vol. 106, no. 1, pp. 25–57, 2006.

[8] H. W. Engl, M. Hanke, and A. Neubauer, *Regularization of inverse problems*, vol. 375 of *Mathematics and its Applications*, Kluwer Academic Publishers Group, Dordrecht, 1996.