Numerical Computation of ODE Sensitivities

Jonathan Calver

Department of Computer Science
University of Toronto

CAIMS 2017
Halifax, Nova Scotia
July 17-21, 2017

Outline

Motivation

- Mathematical models contain unknown parameters
- Estimating these parameters can take the form of a least squares (LSQ) minimization
- $n_o = \#$ of observations
- $\hat{\mathbf{y}}(t_i) =$ observation at time t_i
- $\mathbf{y}(t_i, \mathbf{p}) =$ model prediction at time t_i
- $\mathbf{p} =$ vector of unknown parameters
- $J(\mathbf{p}) = \sum_{i=1}^{n_o} \frac{\|\tilde{\mathbf{y}}(t_i)\|\mathbf{y}(t_i,\mathbf{p})^2}{2}$
- If the model is non-linear, this optimization requires gradient information (**sensitivities**)

This problem may be computationally expensive if

- the model is complex (i.e system of nonlinear ODEs)
- a good initial guess for $\mathbf{p}$ is unavailable
- observations are only available for a subset of $\mathbf{y}$

Definitions

ODEs

We consider the initial value problem (IVP),

$$\begin{aligned}
\dot{\mathbf{y}}(t) &= \mathbf{f}(t, \mathbf{y}(t), \mathbf{p}) \\
\mathbf{y}(0) &= \mathbf{y}_0 \\
t &\in (0, T)
\end{aligned}$$

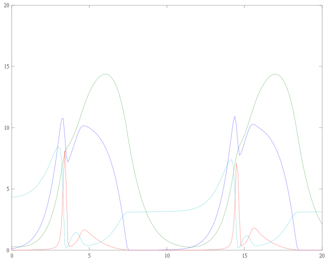
Computing Sensitivities I

In order to solve our LSQ optimization using a gradient based optimizer, we require sensitivity information.

- Finite Differences
- Variational Equations
- Complex Step Method (Automatic Differentiation)
- Adjoint Method (gradient)

All results in this presentation are for the Calcium Ion test problem [**?**, **?**]:

- mildly stiff system of ODEs
- $n_p = 17$
- $n_y = 4$
- $n_o = 11$



Finite Differences I

Forward Differences (FD)

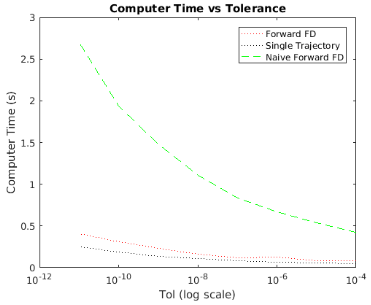
$$\frac{y(p+\epsilon_{FD}) - y(p)}{\epsilon_{FD}} = y'(p) + O(\epsilon_{FD})$$

Centered Differences (CD)

$$\frac{y(p+\epsilon_{CD}) - y(p-\epsilon_{CD})}{2\epsilon_{CD}} = y'(p) + O(\epsilon_{CD}^2)$$

- can suffer from cancellation error and truncation error
- the finite difference perturbations ϵ_{FD} and ϵ_{CD} should be chosen to balance the cancellation and truncation error, keeping in mind the tolereance of the ODE solver (ϵ_{ODE})

Results for Calcium Ion Problem in Matlab I



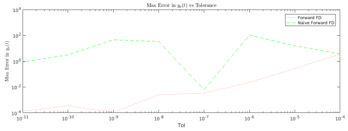
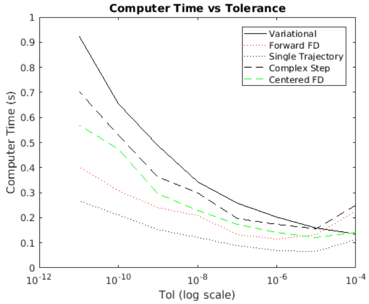


Figure: Experimental results demonstrating the poor performance of FD when a common step size is not used.

Results for Calcium Ion Problem in Matlab II



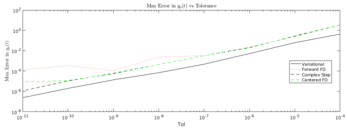


Figure: Experimental results demonstrating the relative performance of several of the methods in Matlab.

Variational Approach I

The variational approach results in the following ODE, which we obtain by taking the time derivative of $\frac{\partial \mathbf{y}}{\partial \mathbf{p}}(t)$,

$$\begin{aligned}
\frac{d}{dt}\frac{\partial \mathbf{y}}{\partial \mathbf{p}}(t) &= \frac{\partial}{\partial \mathbf{p}}\frac{d\mathbf{y}}{dt}(t) \\
&= \frac{\partial}{\partial \mathbf{p}}\mathbf{f}(t, \mathbf{y}(t, \mathbf{p}), \mathbf{p}) \\
&= \frac{\partial \mathbf{f}}{\partial \mathbf{y}}(t)\frac{\partial \mathbf{y}}{\partial \mathbf{p}}(t) + \frac{\partial \mathbf{f}}{\partial \mathbf{p}}(t).
\end{aligned}$$

This matrix valued ODE can be approximated simultaneously with the original system,(4), with the initial conditions, $\frac{\partial \mathbf{y}}{\partial \mathbf{p}}(0)$, whose (i, j) entry is,

$$\frac{\partial \mathbf{y}_i}{\partial \mathbf{p}_j}(0) = \begin{cases} 1, & \text{if } \mathbf{p}_j \text{ is the initial condition for } \mathbf{y}_i \\ 0, & \text{otherwise} \end{cases}.$$

Green's Function Method (GFM) [?] I

$$K'(t,\tau) = f_y(t)K(t,\tau),\ K(\tau,\tau) = \mathbf{1}$$

$$K(t,\tau) = \frac{dy(t)}{dy(\tau)}$$

$$y_p(t) = K(t,0)y_p(0) + \int_0^t K(t,\tau)f_p(\tau)\,d\tau$$

This reduces the variational equations to n_y^2 differential equations and $n_y n_p$ integrals. Kernel Propagation:

$$K(t,\tau) = K(t,s)K(s,\tau)$$

Forward Green's Function Method I

$$y_p(t+\Delta t) = K(t+\Delta t, t) y_p(t) + \int_t^{t+\Delta t} K(t,\tau) f_p(\tau)\, d\tau$$

Step $y_p(t)$ through time, using the Piecewise Magnus Method (PMM) to obtain $K(t+\Delta t, t)$.

$$K(t+\Delta t, t) = \exp \Omega(t+\Delta t, t)$$

- $\Omega(t+\Delta t, t)$ is the Magnus series (truncate and approximate numerically)
- Need to compute the matrix exponential
- Forward propagation of $y_p(t)$ amplifies errors
- At each quadrature point, t_q, in $\int_t^{t+\Delta t} K(t,\tau) f_p(\tau)\, d\tau$, we have to approximate $K(t+\Delta t, t_q)$.

Adjoint Green's Function Method I

$$K(t,\tau) = K^{\dagger}(\tau,t)$$

$$K^{\dagger\prime}(\tau,t) = K^{\dagger}(\tau,t) f_y(\tau),\ K^{\dagger}(t,t) = \mathbf{1}$$

- Can simulate the adjoint Green's function kernel in reverse between observation points.
- Can then propagate $y_p(t)$ forward between observation points.

$$y_p(t_i) = K(t_i,t_{i-1}) y_p(t_{i-1}) + \int_{t_{i-1}}^{t_i} K(t_i,t_{i-1}) f_p(\tau)\, d\tau$$

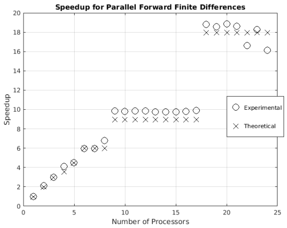
Method Comparison I

method	TOL on $y(t)$	remarks
FD	highest	$n_p + 1$ trajectories, most limited accuracy
CD	high	$2n_p$ trajectories, limited accuracy
CS	above normal	n_p trajectories, complex arithmetic
Vari	normal	requires f_y and f_p, direct error control
GFM	above normal	requires f_y and f_p, indirect error control

Parallel Finite Differences I

Assuming that we have N threads available, the maximum number of simulations a single thread must perform is:

- FD - $\left\lceil \frac{n_p+1}{N_p} \right\rceil$
- CD - $\left\lceil \frac{2n_p}{N_p} \right\rceil$



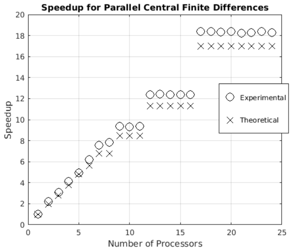


Figure: Speedups for FD and CD for $TOL = 10^{-4}$

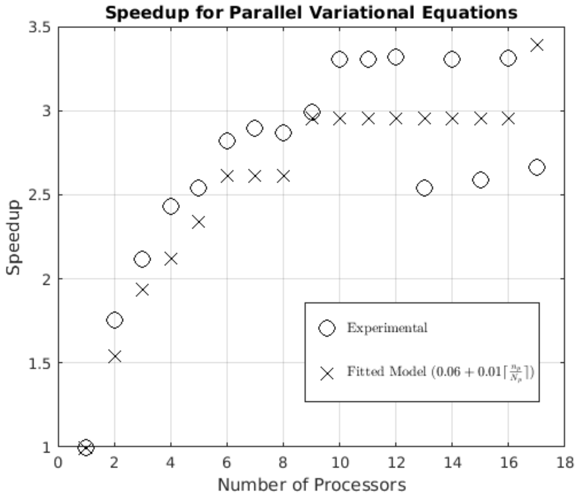
Parallel Variational Approach I

Also best suited for parallelism across parameters

operation	cost (flops)
f	50
f_y	100
f_{p_k}	4
$f_y y_{p_k}$	16
$f_y y_{p_k} + f_{p_k}$	4

- The right hand side requires at least $150 + 24$ flops, no matter how many parameters are being considered.
- The cost associated with a single parameter is roughly 24 flops.
- Max speedup $= \frac{150+(17)24}{150+24} \approx 3.2$.

Parallel Variational Approach II



Parallel Variational Approach III

Figure: Experimental results demonstrating how our parallel version of the Variational approach scales with the number of processors.

Parallel Forward GFM I

- Majority of the work is in approximating the Green's function kernel (independent of the parameters).
- $K(t+\Delta t, t)$ only depends on $y(s)$, $s \in [t, t+\Delta t]$.
- We can approximate $K(t+\Delta t, t)$ before we actually have $y_p(t)$
- Parallelism across the time domain.
- Lower bound on cost will be the cost of simulating $y(t)$.
- Experimentally, cost of simulating $y(t)$ goes up as the number of threads increases.

The parallel version of the adjoint GFM is the same.

Parallel Forward GFM II

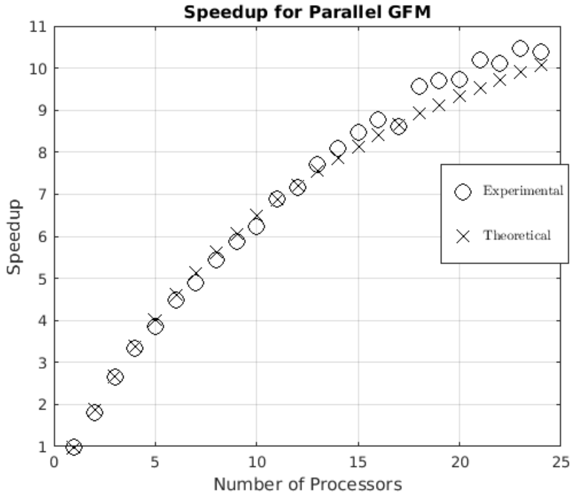


Figure: Speedups for the parallel forward GFM. This is done for $TOL = 10^{-4}$.

Parallel Forward GFM III

Theoretical speedups are based on,

$$S_{\text{theo}} = \frac{1}{f + \frac{1-f}{N_p}}.$$

- f = fraction of computation that is not parallelizable (6%)

Theoretical Results I

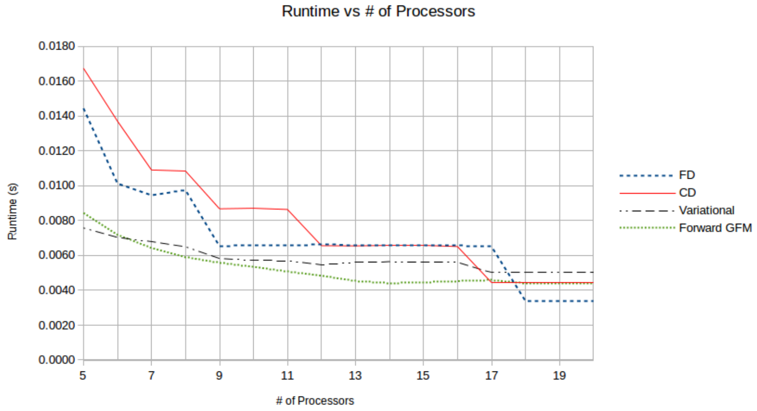


Figure: Experimental results demonstrating how the parallel algorithms compare. This is done for $TOL = 10^{-4}$.

Experimental Results I

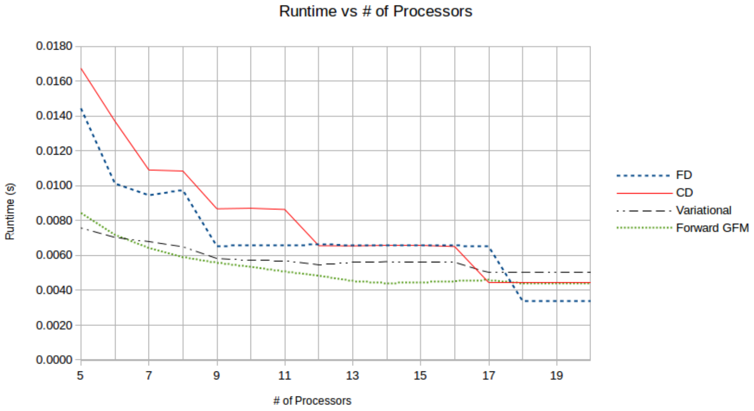


Figure: Experimental results demonstrating how the parallel algorithms compare. This is done for $TOL = 10^{-4}$.

Complex Step (CS) Method [?] I

$$y(p + i\epsilon_{CS}) = y(p) + i\epsilon_{CS} y'(p) - O(\epsilon_{CS}^2) - iO(\epsilon_{CS}^3).$$

Taking the imaginary part and isolating $y'(p)$, we obtain,

$$y'(p) = \frac{1}{\epsilon_{CS}} \Im[y(p + i\epsilon_{CS})] + O(\epsilon_{CS}^2).$$

Furthermore, if we instead take the real part and isolate $y(p)$, we obtain,

$$y(p) = \Re[y(p + i\epsilon_{CS})] + O(\epsilon_{CS}^2).$$

- Similar to forward version of automatic differentiation
- Unlike FD and CD, does not suffer from cancellation
- Requires complex arithmetic

Summary

- Reviewed several methods for computing sensitivities of ODEs
- Studied how each method can exploit parallelism and presented numerical results
- Most methods lend themselves to parallelism across parameters
- The Green's Function Method is best parallised across time

Thanks for listening.