Using exponential integrators to solve large stiff problems

UC Merced Team:
John Loffeld,
Paul Tranquilli,
David Hambley

U.C. Berkeley CS267
Applications of Parallel Computers

Why use exponential integrators?

PDEs are usually solved in the form

$$\frac{dy}{dt} = F[y(t)] \ ,$$

$$y(t_0) = y_0$$

by discretization with the method of lines. When the problem is both large and stiff , these problems are most commonly solved today using Newton-Krylov implicit integrators.

Exponential integrators show promise for more efficiently solving these large stiff ODE problems using the Krylov iteration.

Implicit integrators

Implicit schemes, $U_{n+1} = U_n + \sum_{i=n-k}^{n} a_i \Delta t F(U_i) + \Delta t F(U_{n+1})$, solve the nonlinear system

$$G(U_{n+1}) = U_{n+1} - \Delta t F(U_{n+1}) + Z_n = 0$$

using Newton iteration, $W^{m+1} = W^m - \left(G'(W^m)\right)^{-1} G(W^m)$, with Jacobian $G'^{(U)} = I - \Delta t \frac{DF}{DU}(U)$.

For each Newton iteration, a product of the inverse of the Jacobian and a vector has to be approximated, i.e., it needs $f(A)b$ where $f(x) = \frac{1}{1-x}$!

Exponential integrators

The form for a 4^{th} order exponential integrator is:

$$r_1 = y_n + \frac{9}{12}\phi_1\left(\frac{h}{3A_n}\right)hF_n$$

$$r_2 = y_n + \frac{9}{12}\phi_1\left(\frac{2h}{3A_n}\right)hF_n$$

$$y_{n+1} = y_n + \phi_1(hA_n)hF_n + \frac{160}{243}\phi_2(hA_n)R(r_1) + \frac{128}{243}\phi_3(hA_n)\left(-2R(r_1) + R(r_2)\right)$$

Where *Φ-functions are from the series*:

$$\phi_0 = e^A, \quad \phi_1 = \frac{e^A - I}{A}, \quad \phi_2 = \frac{e^A - A - I}{A^2}, \quad \phi_3 = \frac{e^A - \frac{1}{2}A^2 - A - I}{A^3}$$

(these obey the recurrence $\phi_{i+1}(A) = \frac{\phi_i(A) - \frac{1}{i!}}{A}$)

Krylov iteration

We can approximate $f(A)b$ by projecting onto the Krylov basis:

$$\{b, Ab, A^2 b, A^3 b, \dots\}$$

such that $f(A)b \approx \sum_{i=0}^{N} c_i A^{ib}$

Krylov is a projection onto the same basis as the Taylor series. Therefore we expect the convergence to be similar…

Why exponential integrators may be faster

Intuition behind why exponential integrators may be faster:

$f(A) = \frac{I}{I-A}$ has series $f(A) = I + A + A^2 + A^3 + \cdots$, and

$f(A) = e^A$ has series $f(A) = I + A + \frac{A^2}{2!} + \frac{A^3}{3!} + \cdots$

Because of the factorials, the Taylor series for $f(A) = e^A$ converges much more quickly than the series for $f(A) = \frac{I}{I-A}$.

Therefore we also expect the Krylov to converge faster for the exponential than the rational function.

PETSc implementation

We implemented 4th order exponential in PETsc:

- MPI-based.
- Has Krylov iterators built in, but only exposes as linear solvers. We had to build our own Krylov iteration.

We compute $\phi(A)b$ as $\|b\|V_m f(H_m)e_1$, where V_m is the Krylov basis and H_m is a matrix much smaller than A:

- We still have to compute e^H. We did using scaling and squaring of the Taylor series.

1D Brusselator test problem

The equations are:

$$\frac{\partial u}{\partial t} = 1 + uv^2 - 4u + \alpha \frac{\partial^2 u}{\partial x^2}$$

$$\frac{\partial v}{\partial t} = 3u - u^2 v + \alpha \frac{\partial^2 v}{\partial x^2}$$

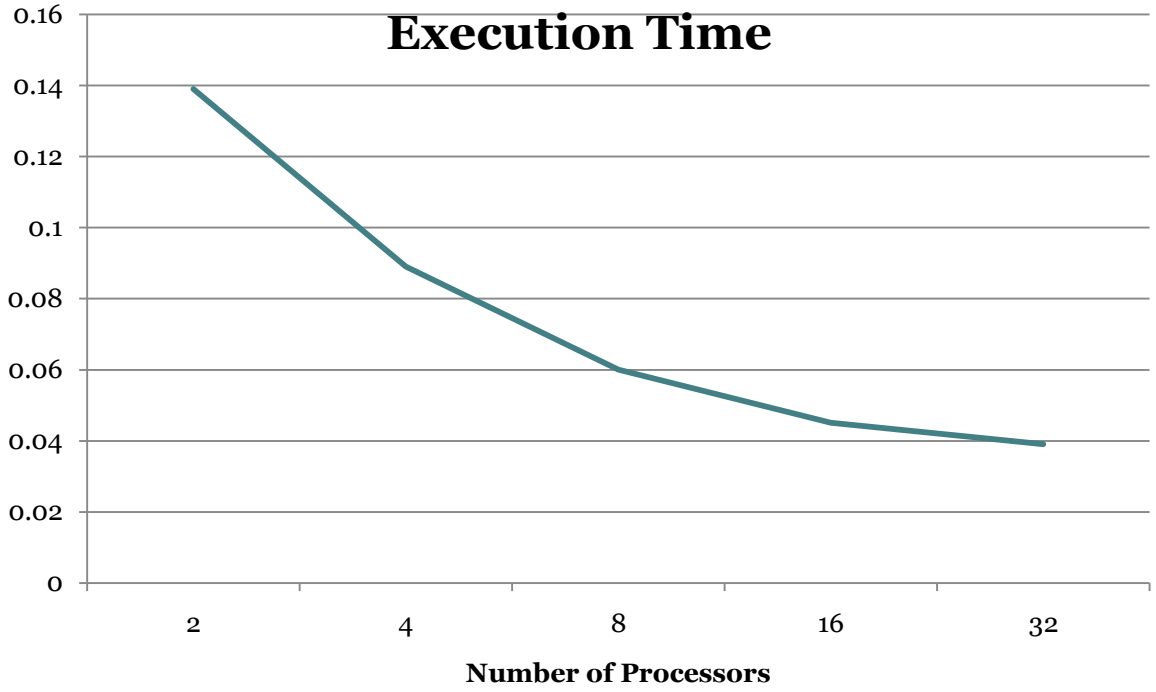
With initial and boundary conditions:

$$u(0,t) = u(1,t) = 1 \qquad v(0,t) = v(1,t) = 3$$

$$u(x,0) = 1 + \sin(2\pi x) \qquad v(x,0) = 3$$

- 2nd order finite differencing
- Jacobian ends up as 4096 x 4096
- $\alpha = .02$

Performance and Conclusions



- Our scalability sucks. We clearly have a lot of work left to do.
- PETSc doesn't expose Krylov iteration through API. We may modify source code to access it.
- Over 75% of time spent in the Krylov iteration.