UQÀM

Time-Parallel Algorithms for Weather Prediction and Climate Simulation

Jean Côté
Adjunct Prof.
ESCER Centre
(Étude et la Simulation du Climat à l'Échelle Régionale)
Sciences de la Terre & de l'Atmosphère
UQAM

EndGAME vs GEM

- Similarities
  - Iterative Crank-Nicolson
  - Non-linear Helmholtz problem
    - Reference profile
  - Coriolis with non-linear terms
  - Semi-Lagrangian continuity equation
  - Placement of variables
  - Variable resolution
  - …
- Differences
  - Deep vs Shallow atmosphere
  - Height vs Pressure terrain following vertical coord.
  - Iterative vs Direct solver kernel
  - …
- Increased collaboration in the future

Table of Content

- Time parallelism & why in NWP?
- Schematic of time parallelism for Initial Value Pb
- “Parareal” Algorithm
  - Description
  - Analysis
  - Applications
- First Steps
  - Advection-Diffusion
  - Burgers
  - …
- Future

Time Parallelism

- Small visionary paper by Nievergelt (1964)
- Need to parallelize seemingly essentially serial algorithms as solving evolution equations to obtain real time solutions on highly parallel computers
- Crucial point: perform redundant operations as long as this allows to reduce real time
- Beneficiary: Numerical Weather Prediction
- “that more general and improved versions of these methods will be of great importance when computers capable of executing many computations in parallel become available”

Why Time-Parallelism in NWP?

- Computer parallelism is expected to increase more rapidly than model resolution
- If there is an optimal number of grid points per processor (core) and the number of horizontal points is fixed then there is a limit on the number of processors that a model can use efficiently
- If the vertical can't be parallelized (because of physics parameterization, etc) then the time direction should be parallelized!
- A research problem – not for tomorrow

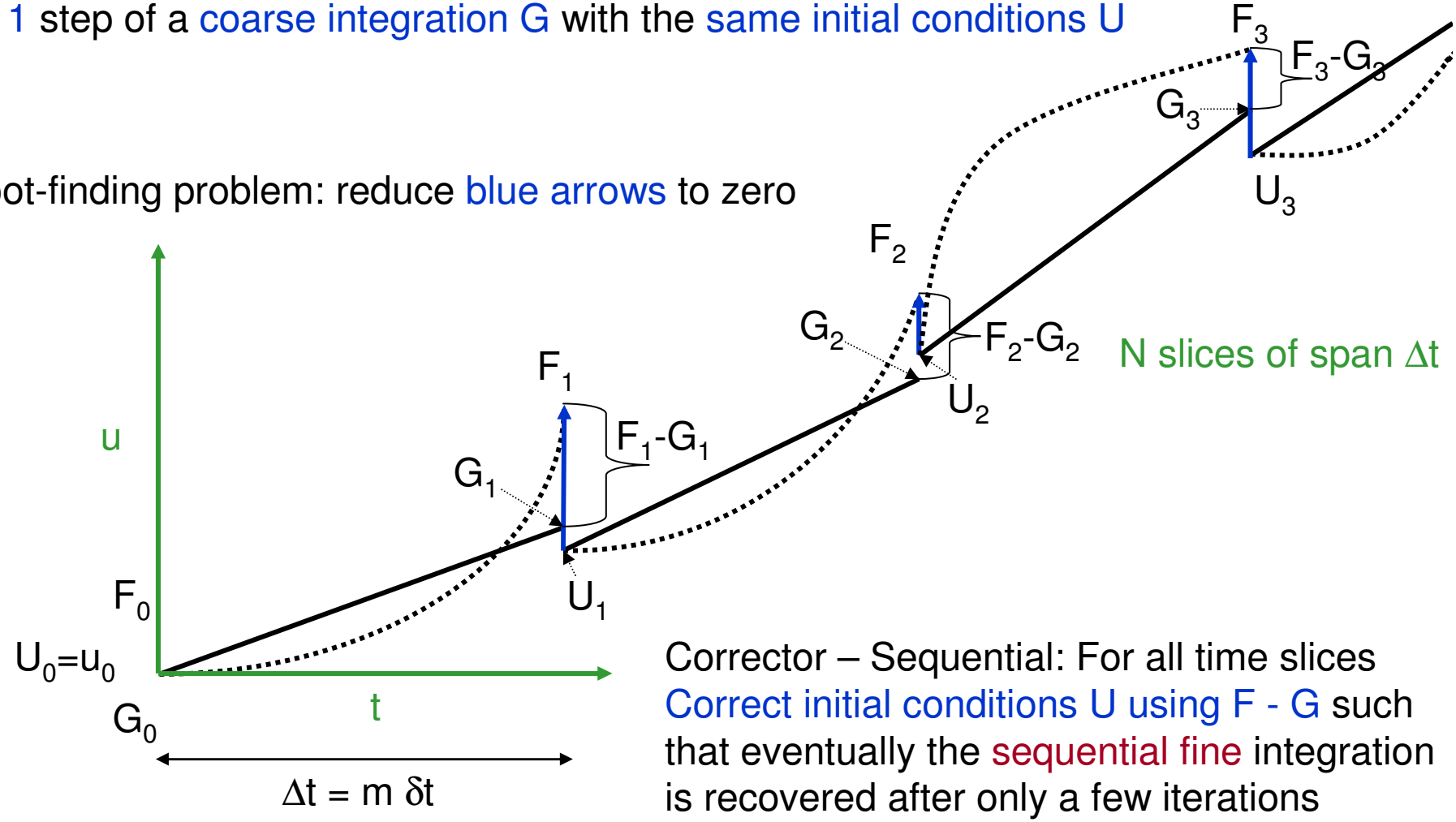
Iterative Scheme

- Suppose we have the discrete sequential solution of an ODE/PDE at every timestep δt for a time span T
- If we were to sample it at every $\Delta t = m\ \delta t$ ($T = N\ \Delta t$)
- Then for every time slice we could obtain in parallel the solution with timestep δt starting from our sampled values
- That solution would be obtained much faster in wall clock time
- An idea for the parallel solution of our ODE/PDE is to start from a set of guessed samples which will give us an approximation of the discrete solution we are interested in
- Somehow correct our guesses and obtain a better solution
- Iterate until convergence to the discrete solution
- If the convergence is fast enough we will need only few iterations and the wall clock time will be reduced (at the expense of spending more CPU time)

Parallel Iterative Scheme for u′ = f(u)

Predictor – Parallel: For all the time slices Δt compare m steps of a fine integration F to 1 step of a coarse integration G with the same initial conditions U

Root-finding problem: reduce blue arrows to zero



N slices of span Δt

Corrector – Sequential: For all time slices Correct initial conditions U using F - G such that eventually the sequential fine integration is recovered after only a few iterations

The "Parareal" Algorithm (bis)

- For $u' = f(u)$, $t = (0,T]$, $u(0) = u_0$
  - Coarse timestep (Δt, $T = N\Delta t$)
  - Fine timestep (δt, $\Delta t = m\,\delta t$)
- For K iterations (k = 1, K)
  - Perform in parallel n = 1, N integrations (predictor):
    - Fine (F_n = final value) of m timesteps
    - Coarse (G_n = final value) of 1 timestep
    - Initial Condition U_{n-1} (k-1) from previous iteration
  - Perform sequentially a coarse integration (corrector) for new initial conditions where a source term is added according to the difference between the fine and the coarse solutions at the common times
    - $U_n(k) = G(U_{n-1}(k)) + F_n - G_n$, $n = 1, N$, $U_0(k) = u_0$
- To start we use the coarse solution

MATLAB Code (adapted from MJ Gander)

```
% initial coarse integration
U = mdl( u_0, T, N, 'coarse' );

% iteration loop
for k = 1 : K

   % parallel loop - predictor
   for n = 1 : N
      Go = mdl( U( :, n ), coarse_step, 1, 'coarse');
      Fo = mdl( U( :, n ), coarse_step, m, 'fine'  );
      G( :, n ) = Go( :, end );
      F( :, n ) = Fo( :, end );
   end % end parallel loop

   % sequential loop - corrector
   if k < K
      for n = 1 : N
         Gn = mdl( U( :, n ), coarse_step, 1, 'coarse' );
         U( :, n + 1 ) = Gn( :, end ) - G( :, n ) + F( :, n );
      end % sequential loop
   end

end % iteration loop
```

The “Parareal” Algorithm …

- If convergence is reached with few iterations (<<N): real time gain at the expense of CPU time
- At convergence U = F_{serial} at the common times
  - No need to iterate further than the accuracy of F
- Acceleration = sequential time/parallel time

$$\frac{N}{K(1+\frac{N}{m})}$$

- Efficiency = Acceleration/number of processors

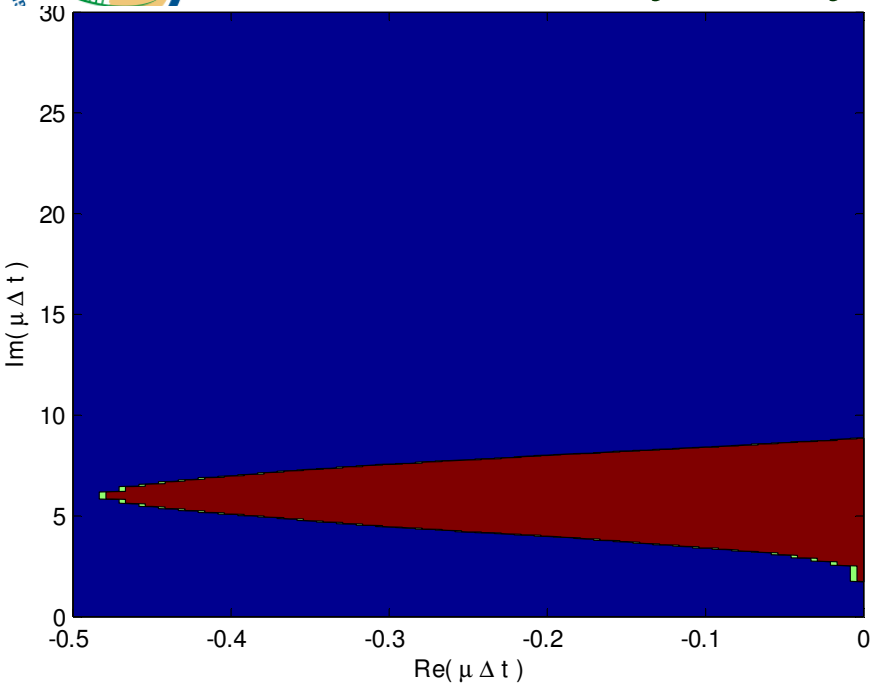
$$\frac{1}{K(1+\frac{N}{m})}$$

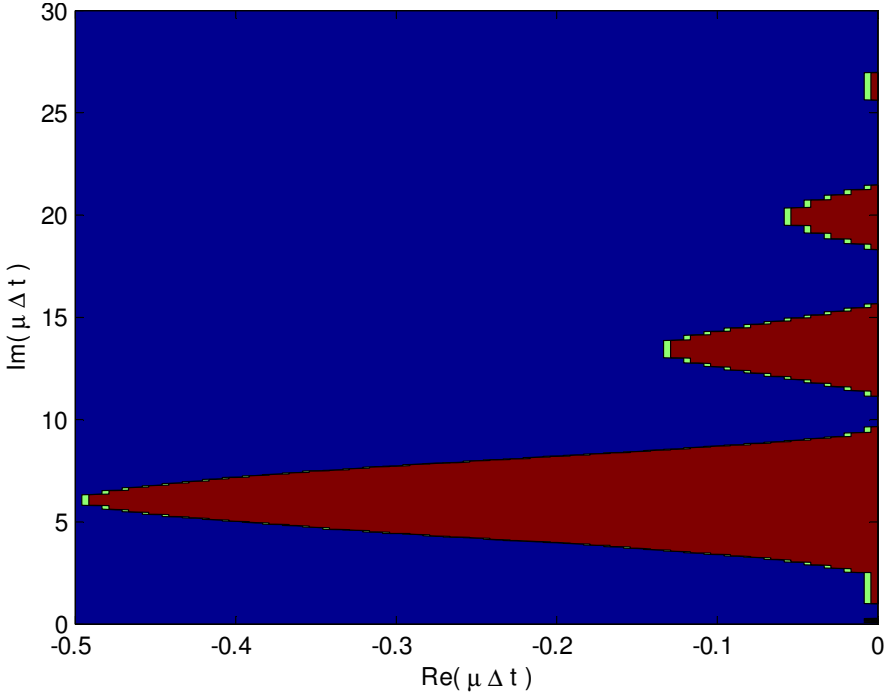
# of processors = # of coarse steps = N

Analysis

- Stability (Staff & Rønquist, Bal)
  - Considering the model equation $\phi' = \mu \phi$
  - Can be unstable if the imaginary part of μ is much larger in modulus than the modulus of the real part of μ (linear stability)
  - There exists a work around for these important cases (Krylov Space Enhanced Parareal, modified Parallel Implicit Time Algorithm (PITA))
- Convergence (Maday, Gander & Vandewalle, Gander & Hairer &c)
  - Supralinear
  - Not a problem, if it converges, it converges fast

Linear Stability Analysis: Implicit Runge-Kutta (Radau 3)





$$z = \mu\Delta t = \mathrm{Re}(z) + \mathrm{Im}(z)i$$

Amplification factor for 1 coarse timestep (λ) and for m fine timesteps (Λ)

$$\Lambda \equiv R(z) = \frac{1 + z/3}{1 - 2z/3 + z^2/6}$$

$$\lambda \equiv R^m(z/m)$$

Amplification factor of "Parareal" at coarse timestep n of iteration k

$$H(n,k,\lambda,\Lambda) = \sum_{j=0}^{k} \binom{n}{j} (\lambda - \Lambda)^j \Lambda^{n-j}$$

Stability domain (in blue) for $\mathrm{Re}(z) < 0$

$$\left|H(n,k,\lambda,\Lambda)\right| \le 1, \text{for all } n,k \le N$$

Linear Stability Analysis: Semi-Lagrangian

- System of ODEs dependent on scale (wave number) of perturbation
- For constant advection and diffusion in a periodic domain we can obtain the amplification factor
  - Various types of interpolation (linear, cubic)
  - Various discretizations for the diffusion operator
  - Various temporal weighting of advection and diffusion
- Preliminary experiments show that the criteria for convergence is that λ and Λ be close enough
- Is the long timestep advantage of the Semi-Lagrangian scheme maintained?
- It works for well resolved waves

Semi-Lagrangian advection with cubic spline

$$\theta = \pi k \Delta x$$

Residual CFL number

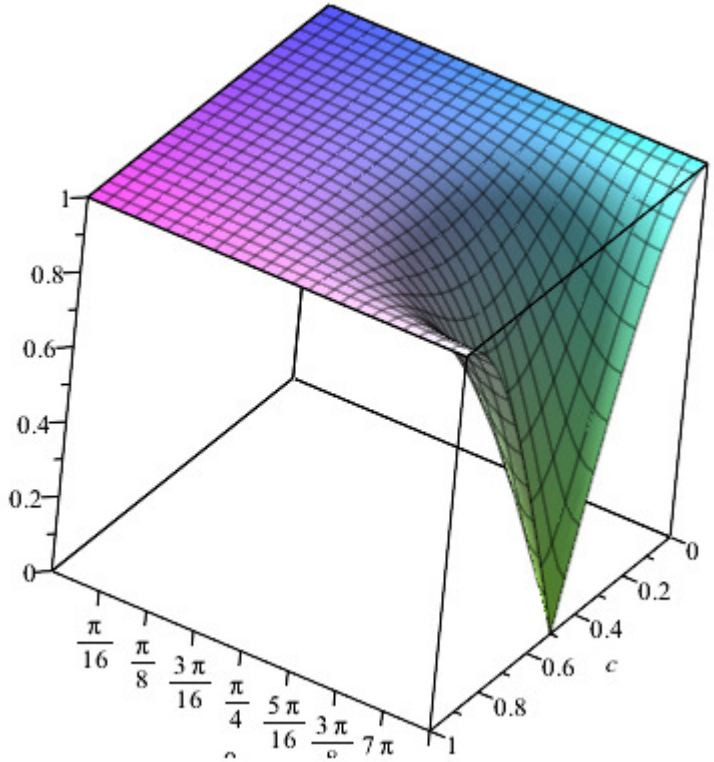
$$c = CFL - n_{CFL} \quad CFL = \frac{u_0 \Delta t}{\Delta x} \quad n_{CFL} = trunc(\ CFL\ )$$

Response of second derivative operator

$$\rho_{xx} = \frac{\sin(\theta)^2}{1 - \frac{2}{3}\sin(\theta)^2}$$

Response of cubic spline interpolation

$$\Lambda = e^{-(1+n_{CFL})2\theta I}\left[\begin{array}{l} c + (1-c)e^{2\theta I} + \ldots \\ \frac{2c(1-c)\rho_{xx}}{3}(1 + c + (2-c)e^{2\theta I}) \end{array}\right]$$



Abs(Λ), $\theta = 0, \pi/2$, c = 0, 1

Applications (partial)

- Finance (Bal, 2002)
- Molecular Dynamics (Baffico & al., 2002)
- Navier-Stokes Equations for diffusion dominated flows (Fischer & al., 2005)
- Reservoir simulation (Garrido & al., 2005)
- Inverse parabolic problems (Daoud, 2007)
- Non-linear Ordinary Differential Equations + Burgers Equation (Gander & Hairer, 2008)
- Korteveg-deVries-Burgers Equations (Kaber & Maday, 2007)
- Chemical Kinetics (Blouza & al. 2010)
- Chemical Reaction-Diffusion (Duarte, & al., 2011)
- Method of Characteristics (Gander, 2008)
- Krylov Space Enhanced Parareal (Gander & Petcu, 2008)

… and in Numerical Weather Prediction?

- Lorenz Equations (Gander & Hairer, 2008)
- An effort began at MeteoSwiss in collaboration with the German Weather Service (DWD)
  - Ruprecht & Krause (June 2011)
  - Schematic problems
  - They are looking at waves that are potentially problematic for “Parareal” but by applying the work around (Krylov Space Enhanced Parareal) and adding damping (divergence damping)

Project

- Stability & convergence
- Performance gain (predicted & measured)
- Choice of schemes on fine (less implicit?) & coarse grid
- Stabilization mechanism
- Ratio between fine & coarse grid timesteps
- Spatial resolution of the coarse grid model
- Relative performance of direct vs. iterative solvers
- Impact on the balance of the flow
- …

Roadmap

- 1D (advection, rotation, Shallow-Water) & 2D (advection, Shallow-Water, vertical slice)
  - cover most problems met in NWP
  - Advection + (diffusion/reaction, sources)
  - Rotation
  - Mix of slow waves (Rossby) & fast waves (gravity, sound)
- Stopping criteria for iterations
- Williamson & al. (1992) cases, for horizontal
- Rising Bubble case (Robert, 1993) & mountain flow mountain (Pinty & al. 1995), for vertical
- If time & work force permit
  - 3D Primitive Equations
  - 3D Non-Hydrostatic Equations
  - 3D model Full Model (including physics parameterization)

First Steps

- Reproduce Gander & Hairer 2008
  - Always 5 iterations (K = 5)
- Ordinary Differential Equations (ODEs)
  - Brusselator
  - Arenstorf Orbits
  - Lorenz Equations
  - 4th Order Runge-Kutta (“classic”)
- Partial Differential Equations (PDEs)
  - Burgers Equation
  - Finite-Difference model similar to paper
  - Semi-Lagrangian model similar to GEM model

Constant Advection-Diffusion SL

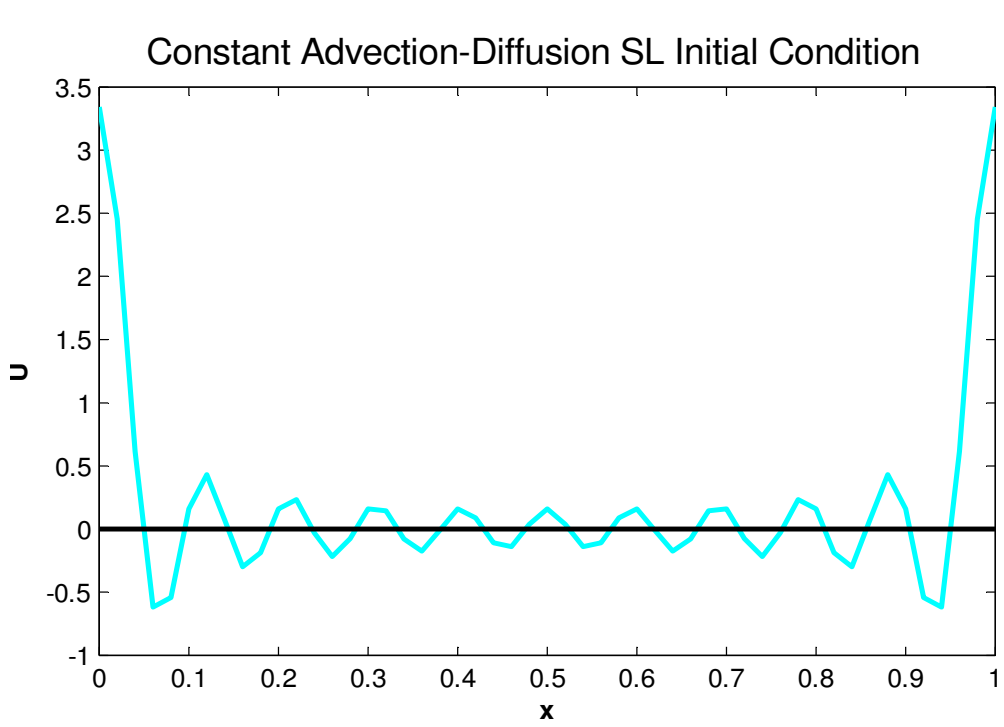
c = 0.432 ν = 5 10^{-6} n_x = 51 T = 30 N = 5 m = 20 K = 5

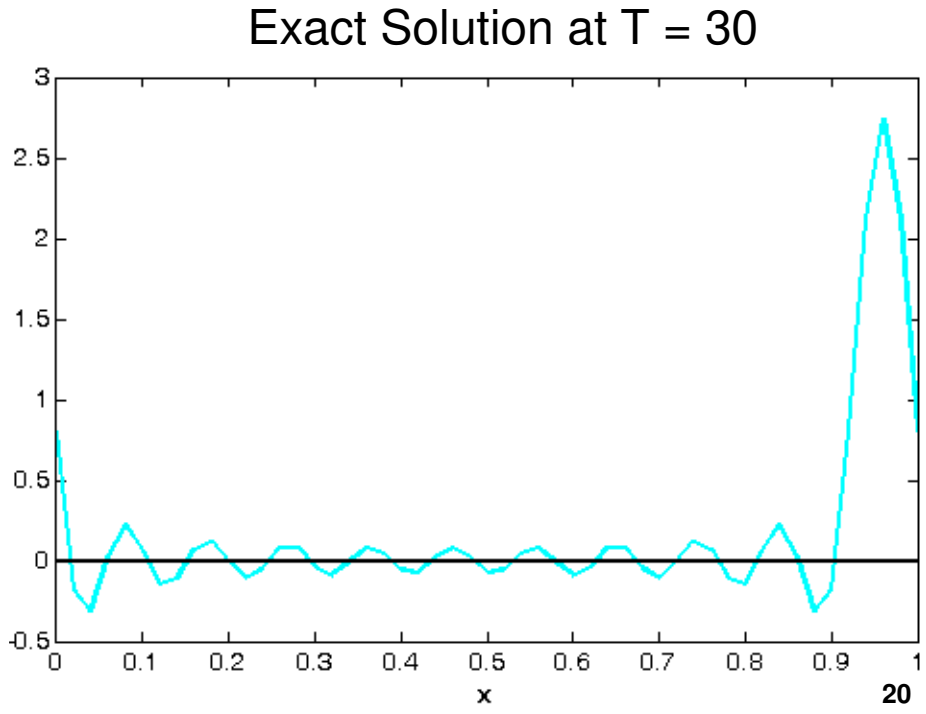
interpolation: cubic spline

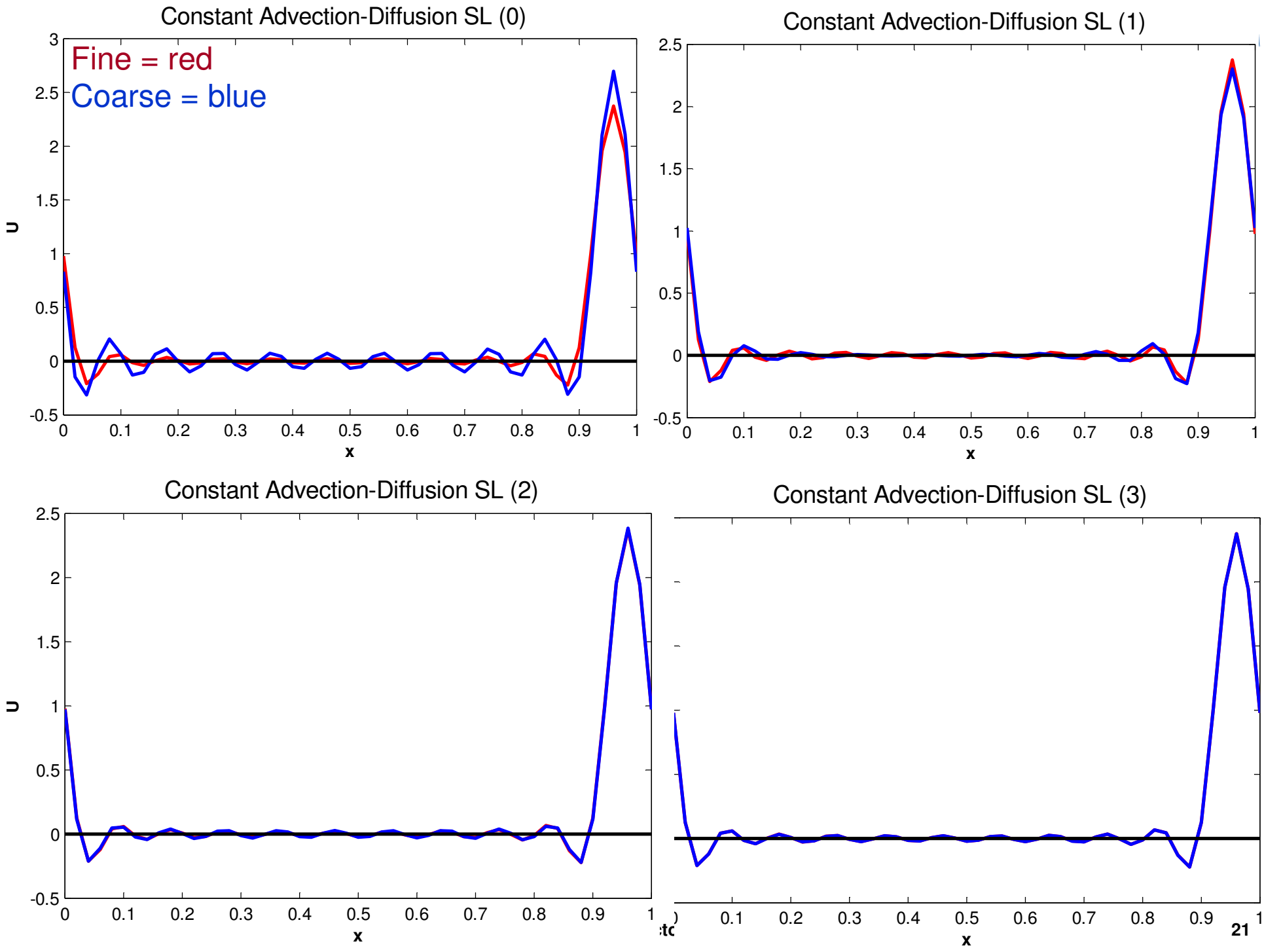
diffusion: finite-element & implicit (γ = 1)

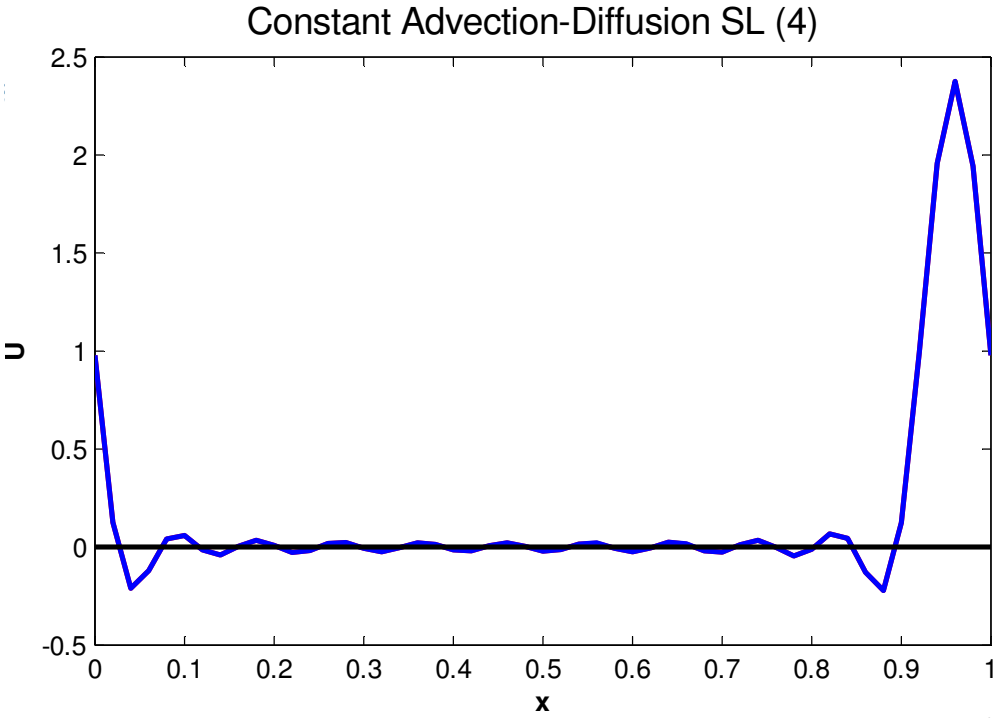
Exact solution

$$u_{exact}(x,t) = \left(1 + \sum_{l=1}^{10} 2\exp(-\nu t(2\pi l)^2)\cos(2\pi l(x-ct))\right)/\pi$$

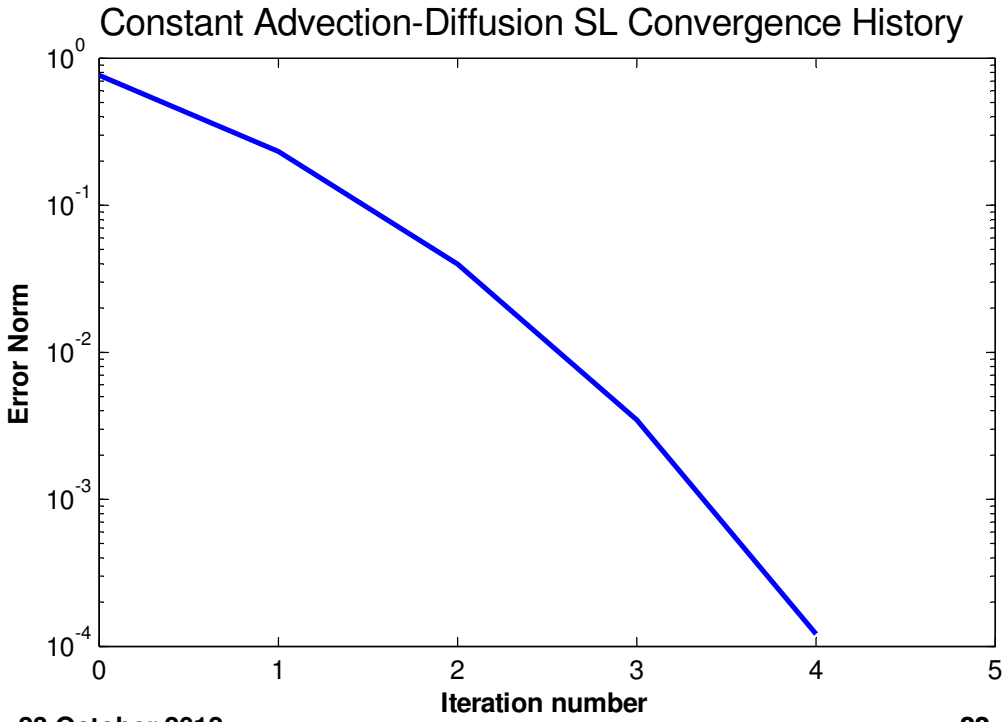





Constant Advection-Diffusion SL (0)
Fine = red
Coarse = blue
u
x
Constant Advection-Diffusion SL (1)
x
Constant Advection-Diffusion SL (2)
u
x
Constant Advection-Diffusion SL (3)
x
21



The coarse solution has converged towards the fine solution



Error Norm at the end of iteration 5 is zero because there are only 5 coarse steps

(K = N = 5)

Future

- Study other formulations
  - Modified PITA (Parallel In Time Algorithm)
  - Deferred corrections
  - “Paraexp”
- Begin Shallow-Water Equations in 1-D
  - Linear & non-linear
  - 2-D horizontal
- Begin Vertical Structure Equation in 1-D
  - Hydrostatic
  - Non-hydrostatic
  - 2-D vertical slice

UQÀM

Thank You!

Questions?

Merci!