Implicit schemes for wave models

Mathieu Dutour Sikirić

Rudjer Bošković Institute, Croatia

and Universität Rostock

April 17, 2013

I. Wave models

Stochastic wave modelling

- Oceanic models are using grids (structured or unstructured) of size $1km \leq d \leq 10km$ to simulate the ocean
- But oceanic waves have a typical wavelength $2m \leq L \leq 100m$. So, we cannot resolve waves in the ocean.
- But if one uses phase averaged models and uses stochastic assumptions then it is possible to model waves by a spectral wave action density $N(\mathbf{x}, \mathbf{k})$
- This density satisfies a Wave Action Equation (WAE) which represents advection, refraction, frequency shifting and source terms:

$$\frac{\partial N}{\partial t} + \nabla_x((\mathbf{c}_g + \mathbf{u}_A)N) + \nabla_k(\dot{k}N) + \nabla_\theta(\dot{\theta}N) = S_{tot}$$

with

$$S_{tot} = S_{in} + S_{nl3} + S_{nl4} + S_{bot} + S_{ds} + S_{break} + S_{bf}$$

The WWM model

- The Wind Wave Model (WWM) is a unstructured grid spectral wave model.
- It is comparable to WaveWatch III, SWAN, WAM or SWAVE.
- It incorporates most existing source term formulation for wind input and dissipation (Cycle III, Cycle IV, Ardhuin, Makin, ...)
- It has been coupled to SELFE, SHYFEM, TIMOR and ROMS.
- It uses Residual Distribution schemes for the horizontal advection.
- It integrates the WAE by using the Operator Splitting Method in explicit or implicit mode.

Operator Splitting Method

- A standard technique for integrating partial differential equations is the operator splitting method.
- Over the interval $[t_0, t_1]$ we successively solve the equations

$$\left\{ \begin{array}{rll} \frac{\partial N_1}{\partial t} + \nabla_\theta(\dot{\theta} N_1) = 0 & \text{with} & N_1(t_0) = N(t_0) \\ \frac{\partial N_2}{\partial t} + \nabla_k(\dot{k} N_2) = 0 & \text{with} & N_2(t_0) = N_1(t_1) \\ \frac{\partial N_3}{\partial t} + \nabla_x((\mathbf{c}_g + \mathbf{u}_A) N_3) = 0 & \text{with} & N_3(t_0) = N_2(t_1) \\ \frac{\partial N_4}{\partial t} = S(t) & \text{with} & N_4(t_0) = N_3(t_1) \end{array} \right.$$

  and we set $N(t_1) = N_4(t_1)$.
- No matter what the order of the successive integration schemes is the final order will be 1.
- It it is possible to have higher order by more complex integration procedures (Strang splitting, iterative splitting, etc.)

The CFL criterion

- If the discretization has characteristic length l and the physical speed is c then we have the condition

$$\frac{c\Delta t}{l} \leq 1$$

- For the integration of the frequency and directional equations we can subdivide the integration time step if necessary because everything is decoupled.
- This is not possible for the geographical advection:
  - The dependency in direction/frequency is small or negligible
  - The problem is that the group speed is $\sqrt{gh}$ and so the CFL number varies with the depth and the resolution.
- So, we will present an implicit scheme for integrating N_1, i.e. in order to avoid the CFL limitation for advection.
- Remark: the advection scheme used in implicit mode in WWM is the residual distribution scheme PSI.

II. MPI parallelization

MPI parallelization I

- The parallelization of geophysical models is usually done by using the Mesage Passing Interface MPI formalism.
- The set of computational nodes of the model is thus split into a number of different subdomains.
- in MPI the exchanges are explicit. The explicit way of doing it is via:

  ```
  CALL MPI_SEND(ArrSend,len,dest,tag,comm,ierr)
  CALL MPI_RECV(ArrRecv,len,orig,tag,comm,istat,ierr)
  ```

  Those operations are blocking, i.e. the program waits until all sends and recvs have been processed.
- This means that all exchanges are processed by the order in which they are stated.
- It is generally better to decrease the total number of exchanges in order to get better performance.

MPI parallelization II

- In order to avoid strictly ordained exchanges, the strategy is to do asynchrone exchanges.
- The procedure is done in the following way

  ```
  DO iorig=1,nproc
    CALL MPI_IRECV(U,1,type(iorig), iorig-1, tag,
         comm, rqst(iorig), ierr)
  END DO
  CALL MPI_WAITALL(nproc,rqst, stat, ierr)
  ```

  and similarly for send operations.
- The idea is the following: the array `type(iorig)` contains the list of positions at which the received data needs to be put. Commands for creating such types are for example `mpi_create_indexed_block`.
- By using the above the order of the exchanges is no longer determined by the MPI program which makes it faster but harder to debug.

II. Iterative solution methods

Iterative solution methods

- In order to resolve linear system $Ax = b$ for typical geophysical situation we have matrices of size $N \times N$ with N about 100000
- We cannot use direct methods like Gauss elimination or LU and so we need to use iterative methods.
- For a matrix A and a vector b the Krylov space $K_n(A, b)$ is

$$K_n(A, b) = Vect \left\{ b, Ab, \ldots, A^{n-1} b \right\}$$

- The Generalized Minimal Residual Method (GMRES) takes the best solution in $K_n(A, b)$ of $Ax = b$.
- It is stable but it requires the storing of n vectors, which is memory intensive.
- So, in order to have a good solution strategy we need a method with minimal storage requirement.

The conjugate gradient method

- If the matrix A is positive definite, then the conjugate gradient method can be used:
  - ➠ J.W. Shewchuk, *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain* Edition $1\frac{1}{4}$
  - If the system is N dimensional then N iteration suffices.
  - After i iterations, the residual error e_i satisfies

$$\|e_i\| \leq 2\left(\frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right)^i \|e\|_0 \text{ with } \kappa = \frac{\lambda_{max}}{\lambda_{min}}$$

  - Operations depends on computing Ax for some vectors x.
- For non-symmetric problems, the technique is to use the biconjugate gradient stabilized (BCGS) which works similarly.

Preconditioners

- The convergence of the conjugate gradient depends on κ that is on how far A is from the identity matrix.
- If κ is large, i.e. A is ill conditioned then the number of iterations will be very large.
- We may accept that but then the whole solution strategy becomes very similar to an explicit scheme.
- The idea is to find a matrix K for which we can compute the inverse easily.
- K must similar to A, i.e. share the same property as A.
- In order to apply the BCGS we need to compute Ax and $K^{-1}x$ for some vectors x.
- Example: Jacobi preconditioning is to take the diagonal entries of A.

Preconditioners for advection

- The essential aspect of advection is that it moves things so Jacobi preconditioner will not work.
- Instead partial factorization techniques have to be used
- We write $A = D + E + F$ with D diagonal E lower triangular and F upper triangular.
- The Successive Over Relaxation (SOR) preconditioner is to say

  $$A = (I + ED^{-1})(D + F) + R \text{ with } R = -ED^{-1}F$$

- The incomplete LU factorization (ILU0) is to say

  $$A_{ij} = (LU)_{ij} \text{ for } A_{ij} \neq 0$$

  with L and U having the same sparsity as A.
- Both methods are efficient because they both are of the form $K = LU$.
- So, when solving $Kx = b$ we do

  $$x' = L^{-1}b \text{ and } x = U^{-1}x',$$

  i.e. the solution propagates.

II. Parallelizing solvers

Parallelizing solvers

- Suppose we have to solve $Lx = b$ and let us assume the diagonal is 1.

$$\left\{\begin{array}{lcl} x_1 & = & b_1 \\ l_{2,1}x_1 + x_2 & = & b_2 \\ \vdots & \vdots & \vdots \\ l_{N,1}x_1 + \cdots + l_{N,N-1}x_{N-1} + x_N & = & b_N \end{array}\right.$$

  So, we first determine x_1 then x_2 and finally x_N.

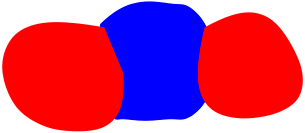
- Parallelization is impossible if all L_{ij} are non-zeros because data from one processor
- What save us is sparsity because the matrices are of the following type:

$$f(x)_v = \sum_{v' \sim v} C_{v,v'} x_{v'}$$

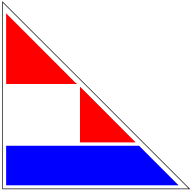
  with $v \sim v'$ mean that v and v' are adjacent nodes.

Ordering nodes

- All incomplete factorizations depend on the ordering of the nodes.
- We are free to choose the ordering that suits us best and by doing so we change the preconditioner LU.
- Since the iterative solution algorithms return approximate solutions this means that the approximate solutions depend on the partitioning and also on the number of processors.
- The situation is the following:



Colored domains



The L matrix

Coloring theory

- A graph G is formed by a set V of vertices and a set E of pairs of vertices named edges.
- A coloring with N colors is a function $f : V \rightarrow \{1, \dots, N\}$ such that for any edge $e = (a, b)$ we have $f(a) \neq f(b)$.
- The chromatic number $\chi(G)$ is the minimum number of colors needed to color.
- It is known that $\chi(G) \leq 4$ for G a planar graph (Appel, Haken, 1976).
- Unfortunately, the subdomains given by parmetis are not necessarily connected and so the graph is not necessarily planar.
- But in practice we can expect that the chromatic number is rarely above 5.

Using colorings to solve $Kx = b$

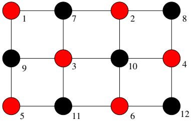
Suppose that we managed to color with c colors

1. The indexing is done
   - 1.1 First index the nodes in domains of color 1 by 1, 2, . . . , n_1
   - 1.2 Then the nodes of color 2 by $n_1 + 1$, $n_1 + 2$, . . . , $n_1 + n_2$.
   - 1.3 .... until color c.
2. The solution of $Lx = b$ is then done by
   - 2.1 Solving $Lx = b$ on the nodes of color 1.
   - 2.2 Nodes of color 1 send data to nodes of higher color.
   - 2.3 Solve $Lx = b$ on the nodes of color 2.
   - 2.4 Continue ...
3. The solution of $Uy = x$ is then done in reverse.

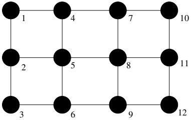
Efficiency of preconditioners

There is no general theory on the efficiency of preconditioners.

- The bad news is that the ordering of the nodes has an effect on the performance of the preconditioner.
- The worst ordering for κ is the red-back ordering in finite difference schemes. The best is the linear ordering.



Red Black



Linear Ordering

- So, the best ordering for the quality of the preconditioner is the one that is hardest to parallelize.
- The ordering that we used is somewhat intermediate. It is like red-black globally, but over individual subdomains it is linear.

II. Solution for wave models

Organizing the computation

1. The penalty of parallelizing come in two ways:
    1.1 The preconditioner quality that decreases.
    1.2 The cost of waiting for data is c.
2. If we have N_{freq} frequencies and N_{dir} directions then this makes $N_{tot} = N_{freq} \times N_{dir}$ independent linear systems to solve.
3. The strategy is then to split N_{tot} into b blocks $B_1, \ldots, B_b$
    3.1 After domains of color 1 have finished block B_1 data is sent and block B_2 is solved.
    3.2 So domains of color 2 can start working before the ones of color 1 are finished.
4. So, by using say $b = 5$ we can essentially remove the second cost.

Further work

1. The work done so far is for the SOR preconditioner.
2. We need to test the ILU0 preconditioner, it is harder to compute but the same strategy can be applied.
3. Another possibility is to integrate implicitly the advection in geographical, frequency and direction.
   This requires an ordering of the $N_{node} \times N_{freq} \times N_{dir}$ matrix entries but by doing so we can diminish the splitting error.
4. And overall improve the speed.

THANK YOU