Numerical modelling of atmosphere and oceans
Lecture 6

P.L. Vidale

NCAS Climate, Department of Meteorology
University of Reading, UK
p.l.vidale@reading.ac.uk

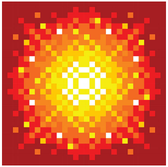
23 February 2023

A graphical description

Let us start with something familiar, e.g. simple advection solved with CTCS. We have our beloved CFL criterion from studying the problem of advection:
$\alpha = U\frac{\Delta t}{\Delta h} \leq \alpha_{max}$, where Δh is our grid spacing.

What is the practical meaning of CFL for Eulearian finite differences?

If we think of it as a method for propagating information in space, the CFL criterion is telling us that we are not allowed to transmit information from grid point i to grid point $i+2$ without first making a stop at grid point $i+1$. If our information "leapfrogs", very bad things will happen.



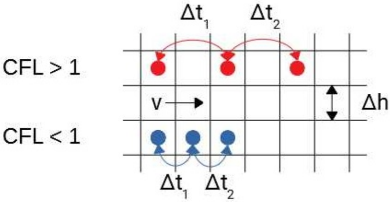


Figure: CFL in the x direction

The example of advection in 1D

Let us start with something familiar, e.g. simple advection solved with CTCS:

$$\frac{A_j^{n+1} - A_j^{n-1}}{2\Delta t} + c\left(\frac{A_{j+1}^n - A_{j-1}^n}{2\Delta x}\right) = 0 \quad (1)$$

We can substitute a (Von Neumann) solution of this type: $A_j^n = B^{n\Delta t}e^{i\mu m\Delta x}$, where μ is the horizontal wavenumber and $m\Delta x$ is the distance along the x axis. B can be any complex number. If we substitute into our finite difference (CTCS) scheme above we obtain:

$$\left(B^{(n+1)\Delta t} - B^{(n-1)\Delta t}\right)e^{i\mu m\Delta x} = -\frac{c\Delta t}{\Delta x}B^{n\Delta t}\left(e^{i\mu(m+1)\Delta x} - e^{i\mu(m-1)\Delta x}\right) \quad (2)$$

If we remember Euler's formula: $e^{i\theta} = \cos\theta + i\sin\theta$ and we multiply the expression above by $B^{\Delta t}$ we obtain a simple equation, after we cancel out the common term A_j^n:

$$B^{2\Delta t} + 2i\sigma B^{\Delta t} - 1 = 0 \quad (3)$$

where $\sigma = \frac{\mathbf{c\Delta t}}{\mathbf{\Delta x}}\sin\mu\mathbf{\Delta x}$, so that (3) has the solution:

$$B^{\Delta t} = -i\sigma \pm \left(1-\sigma^2\right)^{1/2} \quad (4)$$

Two cases may be considered: **stable**, with $|\sigma| \leq 1$ and **unstable**, with $|\sigma| > 1$.

The stable case is very interesting from the point of view of phase, group velocity etc., but does not seem to pose major threats to the stability of the simulation (see Haltiner and Williams, pages 112-119).

The unstable case concerns us from the point of view of the amplitude of the signal. In fact, if $|\sigma| > 1$, then $\left(1-\sigma^2\right)^{1/2}$ is imaginary and both roots will be pure imaginary:

$$B_{+}^{\Delta t} = -i\left(\sigma - S\right), \mathrm{where} |\sigma| > S \equiv \left(\sigma^2 - 1\right)^{1/2}$$
$$B_{-}^{\Delta t} = -i\left(\sigma + S\right)$$

If σ is positive, the magnitude $R = |B_{-}^{\Delta t}| > 1$ and the solution $B_{-}^{\Delta t} = Re^{-i\pi/2}$ will thus grow exponentially when raised to the power of n across the time steps. If σ is positive, the other root has a magnitude exceeding 1.
In either case, the solution: $A_j^n = \left(MB_{+}^{n\Delta t} + EB_{-}^{n\Delta t}\right) e^{i\mu m\Delta x}$ (with two arbitrary constants, M, E, to be determined later), will amplify with increasing time, which is not a desired property, since it does not correspond to the true solution of the differential equation. This phenomenon of *exponential amplification of the solution* is known as *computational instability* and must be avoided at all costs.

The Courant-Friedrichs-Levy (CFL) condition in 1D

So, we are left with having to impose a *condition for a stable solution* $|\sigma| \leq 1$, that is:

$$\left| \frac{c\Delta t}{\Delta x} \sin \mu \Delta x \right| \leq 1 \tag{5}$$

If this condition is to hold for all admissible values of the wavelength μ, then the maximum value of $\sin \mu \Delta x$ will happen for the highest resolved wavenumber (that is a wavelenght $L = 4\Delta x$), which requires that:

$$\left| \frac{c\Delta t}{\Delta x} \right| \leq 1 \tag{6}$$

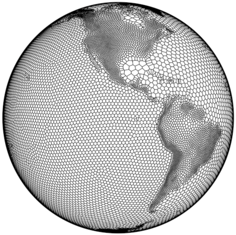
which is commonly referred to as the **Courant-Friedrichs-Levy (CFL)** *condition for computational stability*.

The CFL condition in 1D: plugging in some numbers

What happens in practice? What typical time step are we contending with, in the horizontal and vertical directions?

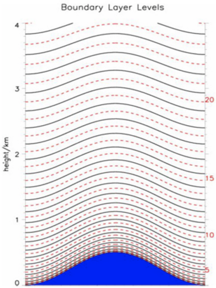
Horizontal example: $\Delta x = 10km$, $c = O(100)ms^{-1}$ what Δt can we afford?

But what happens for a contemporary grid, like this one?



Vertical example: $\Delta z = 500m$, $c = O(1)ms^{-1}$ what Δt can we afford?

But what happens for a typical vertical atmospheric grid, telescoped, like this one?



The Courant-Friedrichs-Levy (CFL) condition in 2D

Same problem as before, albeit advection in 2D:

$$\frac{\partial A}{\partial t} + \mathbf{V}_s \cdot \nabla A = 0, \text{ where } \mathbf{V}_s = U\mathbf{i} + V\mathbf{j} = \text{const.} \tag{7}$$

Let $\Delta x = \Delta y = d$; $x = jd$; $y = kd$; $t = n\Delta t$ and apply CTCS as before:

$$A^{n+1}_{j,k} - A^{n-1}_{j,k} = -\frac{U\Delta t}{d}\left(A^n_{j+1,k} - A^n_{j-1,k}\right) - \frac{V\Delta t}{d}\left(A^n_{j,k+1} - A^n_{j,k-1}\right) \tag{8}$$

Just as before, we apply a solution following the Von Neumann method: $A^n_{j,k} = B^{n\Delta t} e^{ipjd+qkd}$ and we simplify by cancelling common terms, just as we did in the 1D case, leading to:

$$B^{n+1} = B^{n-1} - \frac{2i\Delta t}{d}\left(U \sin pd + V \sin qd\right) B_n \tag{9}$$

We can solve by defining $D = B^{n-1}$, re-writing (9) in matrix form and finding its eigenvalues. This is quite similar to what is done on page 127 of Haltiner and Williams to solve the 1D case we treated before, which results in a quadratic equation ((5-55), page 127) with coefficients identical to what we found previously (equation (4), else see (5-16) in H&W, page 112).

The 2D eigenvalues for stability and the 2D CFL

The eigenvalues are:

$$\lambda = -i\frac{\Delta t}{d}\left(U\sin pd + V\sin qd\right) \pm \sqrt{1-(\frac{\Delta t}{d})^2\left(U\sin pd + V\sin qd\right)^2} \tag{10}$$

which will have (the desired) magnitude of 1 provided that:

$$\frac{\Delta t}{d}\left|U\sin pd + V\sin qd\right| \leq 1 \tag{11}$$

At this time we consider U and V as projections of the vector $\mathbf{V}_s$ onto the x and y directions, making use of the "angle of the wind", or *direction of the wave front* θ: $U = V_s\cos\theta$ and $V = V_s\sin\theta$, which gives us:

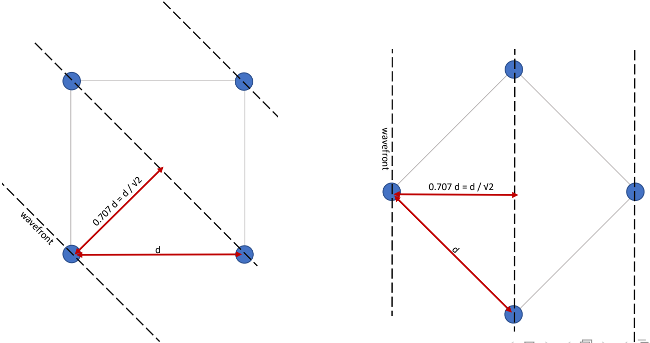
$$\frac{V_s\Delta t}{d}\left|\cos\theta\sin pd + \sin\theta\sin qd\right| \leq 1 \tag{12}$$

Which is the wind direction θ that is most likely to violate CFL? Since the wave numbers p, q are independent, we can choose the maximum value, 1, for each of the two terms: $\sin pd$ and $\sin qd$, and we are left with the task of maximising the remaining sum: $\cos\theta + \sin\theta$, which is $\sqrt{2} = 0.707$, corresponding to $\theta = \pi/4 = 45^o$. So, **for 2D CTCS, the CFL condition** that will allow us to avoid computational instability under all circumstances (even with wind blowing at an angle of 45^o) is:

$$V_s\Delta t\sqrt{2}/d \leq 1 \text{ or } V_s\Delta t \leq 0.707d \tag{13}$$

2D CFL: graphical interpretation

In summary, we have seen that in 2D the maximum value of Δt required for computational stability is almost 30% smaller than for the 1D case, other things being equal. In order to understand this, consider the two figures below: on the left is the grid as seen normally, while on the right is the grid as seen when aligned with the progressing wavefront, at an angle of 45^o. A wave propagating at that angle (e.g from the SW to the NE) encounters an effective distance of $d/\sqrt{2}$ between gridpoints. So, the CFL stability criterion says that the wave cannot move more than the effective distance between gridpoints ($d/\sqrt{2}$ at its shortest) during time Δt, without incurring computatational instability.



CFL in the real world

In lectures 4,5 we started to learn about Arakawa-sensei's staggered grids, and I showed you my current C-grid configuration, for the UM's EndGame dynamical core.

In the last row near the poles, my $\Delta x \approx 5$km GCM has a grid spacing of 3.1m. Given a "worst case scenario" for wave propagation, what should my time step be, according to the 1D ("optimistic!") CFL criterion? Please compute that now, and raise your hand as soon you have the answer.

Here is my real time step:

Table 5 Computational Aspects. To give a rough idea of the performance of the different models their throughput is given, and contextualized by some basic information about architectures on which they were run.

Model	Nodes	Cores	SDPD	Δt	Processor (date of launch into market)
ARPEGE-NH	300	7200	2.6	100 s	Intel Xeon "Ivy Bridge" (2013)
FV3	384	13824	19	4.5 s	Intel Xeon "Broadwell" (2016)
GEOS	512	20480	6.2	3.75 s	Intel Xeon "Skylake" (2017)
ICON	540	12960	6.1	4.5 s	Intel Xeon "Haswell" (2014))
IFS	360	12960	124	240 s	Intel Xeon "Broadwell" (2016)
MPAS	256	9216	3.5	10 s	Intel Xeon "Broadwell" (2016)
NICAM	640	2560	2.6	10 s	NEX SX-ACE vector processor (2015)
SAM	128	4608	6.0	7.5 s	Intel Xeon "Broadwell" (2016)
UM	340	12240	6.0	90.0 s	Intel Xeon "Broadwell" (2016)

Figure: DYAMOND GCMs, with typical $\Delta x \approx 5$km, from Stevens et al., 2019

How can such magic be possible?

CFL and coupled equations: where does Δt matter?

Let us not confuse the advection process with the other processes, described in parametrizations, or with the coupling between the equations.
Also, remember that the time step may be set by the 2D CFL, but the vertical grid spacing is very likely far smaller, meaning that we must use a different numerical scheme in the vertical.
As an example, take a look at our SWEs:

$$
\begin{aligned}
\frac{u^{n+1}-u^{n-1}}{2\Delta t} - fv^n + \frac{g}{2}\left(\frac{\partial h}{\partial x}^{n+1} + \frac{\partial h}{\partial x}^{n-1}\right) &= 0 \\
\frac{v^{n+1}-v^{n-1}}{2\Delta t} + fu^n + \frac{g}{2}\left(\frac{\partial h}{\partial y}^{n+1} + \frac{\partial h}{\partial y}^{n-1}\right) &= 0 \\
\frac{h^{n+1}-h^{n-1}}{2\Delta t} + \frac{H}{2}\left(\frac{\partial u}{\partial x}^{n+1} + \frac{\partial u}{\partial x}^{n-1} + \frac{\partial v}{\partial y}^{n+1} + \frac{\partial v}{\partial y}^{n-1}\right) &= 0.
\end{aligned}
$$

Staggered time integration schemes for the SWE

The advection-diffusion equation in one dimension is a useful prototype with which to explore issues of stability, convergence and approximation error. However, the equations ocean modellers use are frequently multi-dimensional. This leads to special problems and results in some cases. With regards to temporal stability, however, schemes which are unstable in one dimension are also unstable in multi-dimensional problems. Schemes which are conditionally stable in one dimension are also conditionally stable in two dimensions or higher, but with more restrictive conditions on Δt.

For equations that support more than one type of process, **the stability criterion will depend on the fastest propagating processes**. However, the fastest propagating phenomena might be of little physical importance and therefore the stability condition might be too constraining. For instance, the linear shallow water equations allow the existence of inertia-gravity and Rossby waves. The propagation speed of the former is about $\sqrt{gH} \approx 100$ ms^{-1}, which is quite large. For Rossby waves, the propagation is of the order of βR_D^2, which is much smaller. The time integration scheme will therefore greatly depend on the physical processes of interest.

Reminder from Lecture 5: a semi-implicit scheme for SWEs

The shallow water equations linearised about a state of rest are below discretised using a leapfrog scheme for Coriolis terms but a trapezoidal scheme (mixed implicit-explicit) for the gravity wave terms:

$$
\begin{aligned}
\frac{u^{n+1}-u^{n-1}}{2\Delta t} - fv^n + \frac{g}{2}\left(\frac{\partial h}{\partial x}^{n+1} + \frac{\partial h}{\partial x}^{n-1}\right) &= 0 \\
\frac{v^{n+1}-v^{n-1}}{2\Delta t} + fu^n + \frac{g}{2}\left(\frac{\partial h}{\partial y}^{n+1} + \frac{\partial h}{\partial y}^{n-1}\right) &= 0 \\
\frac{h^{n+1}-h^{n-1}}{2\Delta t} + \frac{H}{2}\left(\frac{\partial u}{\partial x}^{n+1} + \frac{\partial u}{\partial x}^{n-1} + \frac{\partial v}{\partial y}^{n+1} + \frac{\partial v}{\partial y}^{n-1}\right) &= 0.
\end{aligned}
$$

Re-arranging with future values on the left:

$$
\begin{aligned}
u^{n+1} + \Delta t g \frac{\partial h}{\partial x}^{n+1} &= A \\
v^{n+1} + \Delta t g \frac{\partial h}{\partial y}^{n+1} &= B \\
h^{n+1} + \Delta t H\left(\frac{\partial u}{\partial x}^{n+1} + \frac{\partial v}{\partial y}^{n+1}\right) &= C
\end{aligned}
$$

Can we mix and match at will? Let us start again with the basic ideas of explicit and implicit.

Implicit schemes

In the shallow water equations, fast gravity waves are due to the divergence and gradient terms, and slow Rossby waves are due to the Coriolis term. By treating them differently, one could try to circumvent the stability condition due to gravity waves. The following time integration schemes allow one to change the degree of implicity of the divergence, Coriolis and gradient terms:;

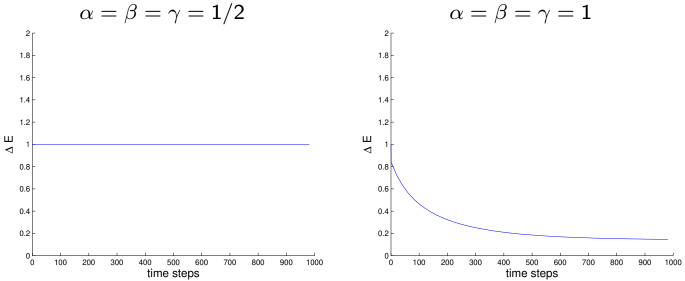
$$\begin{aligned} \eta^{n+1} + \alpha h \Delta t \boldsymbol{\nabla} \cdot \mathbf{u}^{n+1} &= \eta^n - (1-\alpha) h \Delta t \boldsymbol{\nabla} \cdot \mathbf{u}^n, \\ \mathbf{u}^{n+1} + \beta f \Delta t \mathbf{k} \times \mathbf{u}^{n+1} + \gamma g \Delta t \boldsymbol{\nabla} \eta^{n+1} &= \mathbf{u}^n - (1-\beta) f \Delta t \mathbf{k} \times \mathbf{u}^n - (1-\gamma) g \Delta t \boldsymbol{\nabla} \eta^n. \end{aligned}$$

The implicity coefficients α, β and $\gamma \in [0,1]$. One way to have "some information" about the accuracy and stability of a time integration scheme is to compute the evolution of the energy:

$$E^n = \int_\Omega \frac{1}{2} \rho \left(g(\eta^n)^2 + h \|\mathbf{u}^n\|^2 \right) \, \mathrm{d}\Omega.$$

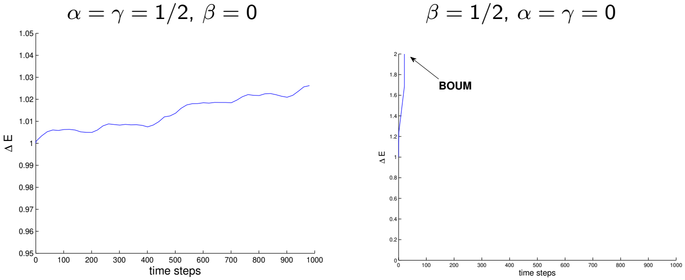
Note that this quantity only gives information about the changes in the amplitude of the solution. It does not indicate how the numerical method is affecting the phase of the solution.

“Mostly implicit” schemes are unconditionally stable



However, they can be quite dissipative if they are “a bit too implicit”. The semi-implicit scheme ($\alpha = \beta = \gamma = 1/2$) is exactly conserving energy while a fully-implicit scheme ($\alpha = \beta = \gamma = 1$) is very dissipative and thus not always accurate.

Both the gravity and Coriolis terms must be at least semi-implicit

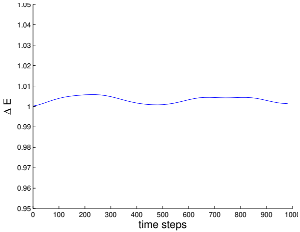


When there is no dissipation, the equations are purely hyperbolic ($\mathcal{R}e(\kappa) = 0$) and the time integration scheme can only be stable if the implicity coefficients are all larger or equal to 1/2. In that case, the solution of the system requires to invert a non-diagonal matrix, which can be computationally expensive. The effect of using a "mostly implicit" time integration scheme with a large time step is to slow down fast propagating gravity waves. These schemes are thus useful for long simulation for which gravity waves are physically insignificant.

Example of a stable explicit scheme: Adams-Bashforth 3

$$\eta^{n+1} = \eta^n - h\Delta t \boldsymbol{\nabla} \cdot \left(\frac{23}{12}\mathbf{u}^n - \frac{16}{12}\mathbf{u}^{n-1} + \frac{5}{12}\mathbf{u}^{n-2} \right),$$

$$\mathbf{u}^{n+1} = \mathbf{u}^n - f\Delta t\mathbf{k} \times \left(\frac{23}{12}\mathbf{u}^n - \frac{16}{12}\mathbf{u}^{n-1} + \frac{5}{12}\mathbf{u}^{n-2} \right) - g\Delta t \boldsymbol{\nabla} \left(\frac{23}{12}\eta^n - \frac{16}{12}\eta^{n-1} + \frac{5}{12}\eta^{n-2} \right)$$



This scheme is conditionally stable with a stability condition prescribed by gravity waves. A leap-frog scheme would have about the same properties but it would become unstable if some dissipation was added to the scheme.

[1]

[1] More details: Durran D.R. (1991) "The Third-order Adams-Bashforth method: An attractive alternative to Leap-frog time differencing". *Monthly Weather Review*, 119, 702-720.

Another example: Forward-Backward in Time scheme

The Forward-Backward in Time (FBT) scheme has been introduced by Sielecki (1968) and later analysed and improved by Beckers and Deleersnijder (1993)[2]. It tries to mimic a semi-implicit scheme by alternatively changing the order in which the two momentum equations are solved for. The future height anomaly term is used in the two momentum equations as soon as it is available; the Coriolis term is discretized by using the most recently computed velocity component, which requires alternation:

$$\begin{cases} \eta^{n+1} = \eta^n - h\Delta t \boldsymbol{\nabla}\cdot\mathbf{u}^n, \\ u^{n+1} = u^n + f\Delta t v^n - g\Delta t \dfrac{\partial\eta}{\partial x}^{n+1}, \\ v^{n+1} = v^n - f\Delta t u^{n+1} - g\Delta t \dfrac{\partial\eta}{\partial y}^{n+1}, \end{cases}$$

and

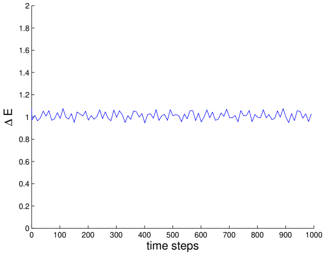
$$\begin{cases} \eta^{n+2} = \eta^{n+1} - h\Delta t \boldsymbol{\nabla}\cdot\mathbf{u}^{n+1}, \\ v^{n+2} = v^{n+1} - f\Delta t u^{n+1} - g\Delta t \dfrac{\partial\eta}{\partial y}^{n+2}, \\ u^{n+2} = u^{n+1} + f\Delta t v^{n+2} - g\Delta t \dfrac{\partial\eta}{\partial x}^{n+2}. \end{cases}$$

[2] Sielecki A. (1968) "An energy conserving difference scheme for the storm surge equations", *Monthly Weather Review*, 96, 150-156.

Beckers J. and Deleersnijder E. (1993) "Stability of a FBTCS scheme applied to the propagation of shallow-water inertia-gravity waves on various grids", *Journal of Computational Physics*, 108, 95-104.

Forward-Backward in Time scheme (cont'd)

It can be seen that for each set of equations, the Coriolis term is first discretized explicitly and then implicitly. The order of the two momentum equations is switched in the second set of equations. As a result, the time discretization of the Coriolis term appears to be semi-implicit "on average". The divergence is always explicit and the gradient is always implicit. It can be shown that the scheme is conditionally stable with about the same stability condition as AB3.



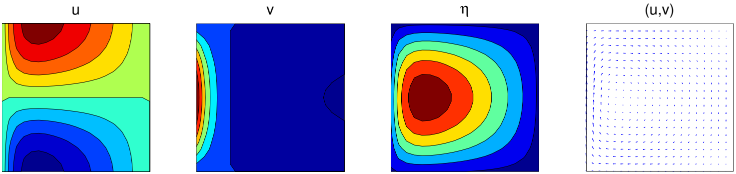
Comments about the second project

The goal of the second project is to solve the Stommel model on the finite difference C-grid with a FBT time integration scheme. The Stommel model is the simplest geophysical flow model able to represent a wind-driven circulation in a closed basin with a western boundary layer. The model equations read:

$$\frac{\partial \mathbf{u}}{\partial t} + (f_0 + \beta y)\mathbf{e}_z \times \mathbf{u} = -g\boldsymbol{\nabla}\eta - \gamma\mathbf{u} + \frac{\boldsymbol{\tau}^\eta}{\rho h}, \tag{14}$$

$$\frac{\partial \eta}{\partial t} + h\boldsymbol{\nabla}\cdot\mathbf{u} = 0, \tag{15}$$

where f_0 is the reference value of the Coriolis parameter, β is the reference value of the Coriolis parameter first derivative in the y-direction, γ is a linear friction coefficient, ρ is the homogeneous density of the fluid and $\boldsymbol{\tau}^\eta$ is the wind stress acting on the surface of the fluid. The model solutions look like these:



How do we decide on grid, grid spacing, time step?

Remember that we started with the CFL criterion in 1D:

$$\mu = U\frac{\Delta t}{\Delta x} \leq 1 \tag{16}$$

There are now several ways in which we can meet the stability criteria for our particular 2D SWE problem. Thinking a bit harder about these decisions seems to lead to some circular reasoning: where do we start? **Are there tradeoffs?**

Questions that we must consider when we choose the numerical method to solve our problem:

1. What is the "worst case scenario" distance relevant to CFL for SWEs in 2D?
2. How many grid points are the minimum required to resolve anything?
3. What is the size of our domain?
4. What is the size of the phenomenon we are trying to simulate?
5. What do all these decisions end up causing in terms of the physical realism of the solutions? For instance, compromising the accuracy of solution for one wave type.

There is more to choosing Δt and Δx

It is not a given that going for a small μ is always the best possible decision. For instance, let us start with the advection equation and let us choose the **upstream scheme**, which we know is **dissipative**. That means, each time step we lose a bit of the signal.
If we increase the resolution of our model, making the time step smaller, we may think that we will obtain a better solution. However, this is not necessarily true for every μ.

Consider a domain of size D, resolved by a number J of Δx points, in which we are solving the advection equation:

$$\frac{A_j^{n+1} - A_j^n}{\Delta t} + c\left(\frac{A_j^n - A_{j-1}^n}{\Delta x}\right) = 0 \tag{17}$$

$D = J\Delta x$ and **every time that we decrease Δx we increase J, so that the domain size D does not change**. For a signal that has wavenumber k in the x direction:

$$k\Delta x = \frac{kD}{J} \tag{18}$$

Stability analysis of the upstream advection equation

If you remember your Von Neuman analysis, **the amplification factor,** λ for upstream advection (equation 17) has this form:

$$|\lambda|^2 = 1 + 2\mu(\mu - 1)(1 - \cos k\Delta x) = 1 + 2\mu(\mu - 1)[1 - \cos(\frac{kD}{J})] \qquad (19)$$

which tells us that λ depends on both wavenumber (k) and our Courant number (μ). In order to maintain computational stability, *we keep μ fixed as Δx decreases*, and that decision limits our time step:

$$\Delta t = \frac{\mu \Delta x}{c} = \frac{\mu D}{cJ} \qquad (20)$$

Since we know the velocity of the fluid, c, we also know how long it takes for it to cross the entire domain: $T = \frac{D}{c}$. If we choose to carry out the time integration in N steps:

$$N = \frac{T}{\Delta t} = \frac{D}{c\Delta t} = \frac{D}{\mu \Delta x} = \frac{J}{\mu} \qquad (21)$$

How damped is the signal after N time steps?

The total amount of damping that "accumulates" throughout the time integration (N time steps) is given by:

$$|\lambda|^N = (|\lambda^2|)^{N/2} = \left\{1 - 2\mu(1-\mu)\left[1 - \cos(\frac{kD}{J})\right]\right\}^{\frac{J}{2\mu}} \tag{22}$$

This relationship says that λ depends on J (the resolution) in two different ways, so that making Δx smaller (increasing the resolution) can be good or bad.

Which is it? Let us pick a wavelength half the domain size, so that $kD = 4\pi$. This causes the cos factor in the equation to approach 1, which **weakens the damping**; on the other hand it also causes the exponent to increase, which **strenghtens the damping**. This is shown in the figure, for two values of μ.

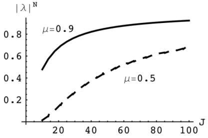


Figure 3.6: "Total" damping experienced by a disturbance crossing the domain, as a function of J, the number of grid points across the domain, for two different fixed values of μ. In these examples we have assumed $D/L = 2$, i.e., the wavelength is half the width of the domain.

Overall, **increasing the resolution, J, is good**: λ tends to 1 on the right hand side, even though we take more time steps, N, to complete the integration. However, if we fix J and decrease μ, by decreasing the time step Δt, the damping increases and **the solution will be less accurate**.

Recommendations on choosing Δt and Δx

- perform numerical analysis, speficically for each scheme
- **for the upstream scheme**, the amplitude error can be minimised by using the largest stable value of μ.
- that is, do not exaggerate in making μ much less than 1: this decision impacts both cost and quality of the solutions
- in other words, it pays off to live dangerously!

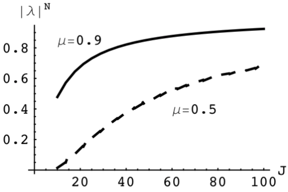
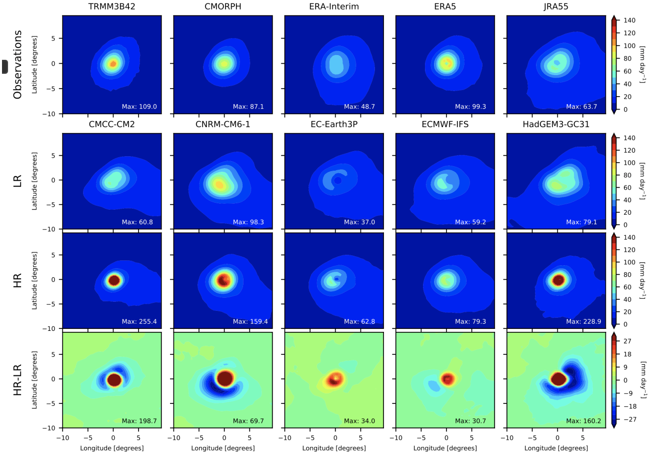
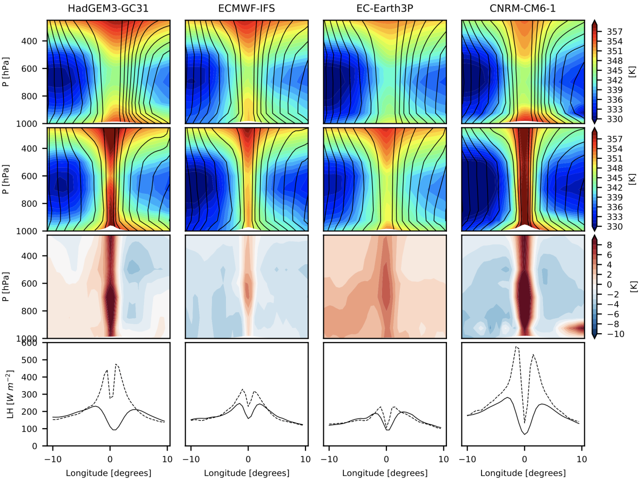


Figure 3.6: "Total" damping experienced by a disturbance crossing the domain, as a functio[n of] the number of grid points across the domain, for two different fixed values of μ. In these exa[mples] we have assumed $D/L = 2$, i.e., the wavelength is half the width of the domain.

Hurricane and Typhoon simulation in climate GCMs: precipitation composite



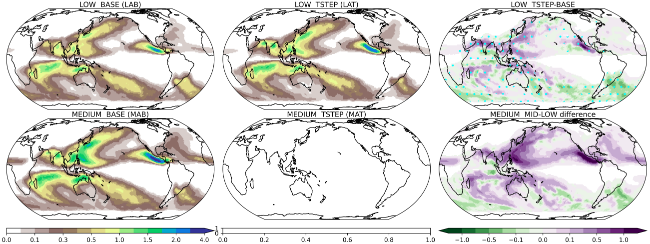
Hurricane and Typhoon simulation in climate GCMs: vertical structure composite



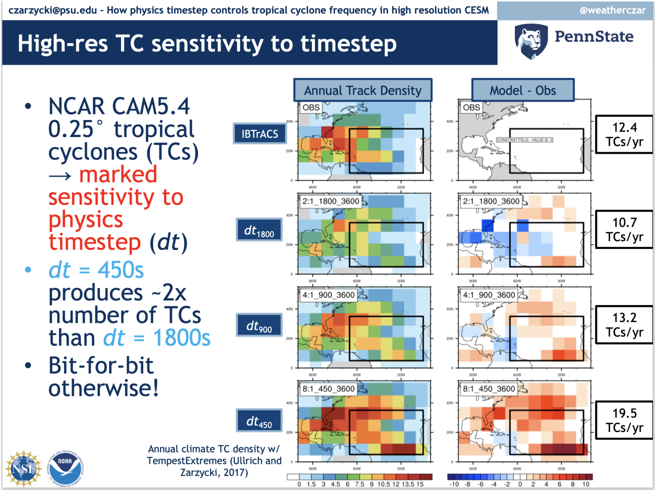
Why the differences

These GCMs are very different in terms of dynamical core, parameterizations etc., but they are also quite different in terms of the use of the time step. I have been experimenting with our UK model (HadGEM3), and Colin Zarzycki has been experimenting with the NCAR model, managing to double the number of hurricanes with a time step of 1/4, but I have also tested the same ideas in the ECMWF model, since it is the one that uses very long time steps.

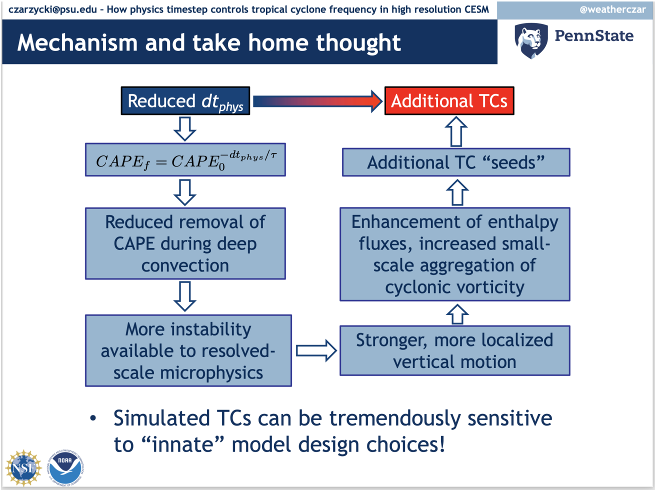
PRESENT : annual climatologies



Hurricane and Typhoon simulation in climate GCMs: frequency



Hurricane and Typhoon simulation in climate GCMs: dynamics and physics



Different types of BC: what they are and what they do

The **Dirichlet (or first-type) boundary condition** is a type of boundary condition, named after Peter Gustav Lejeune Dirichlet (1805–1859). When imposed on an ordinary or a partial differential equation, it specifies the values that a solution needs to take on along the boundary of the domain. For example, for an ODE:

$$y'' + y = 0 \tag{23}$$

we could specify $y(a) = \alpha; y(b) = \beta$ on the interval $[a, b]$, where α, β are given numbers.

The **Neumann (or second-type) boundary condition** is a type of boundary condition, named after Carl Neumann. When imposed on an ordinary or a partial differential equation, the condition specifies the values in which the derivative of a solution is applied within the boundary of the domain. For example, for the same ODE above: we could specify $y'(a) = \alpha; y'(b) = \beta$ on the interval $[a, b]$, where α, β are given numbers.

The **Robin boundary condition, or third type boundary condition**, is a type of boundary condition, named after Victor Gustave Robin (1855–1897). When imposed on an ordinary or a partial differential equation, it is a specification of a linear combination of the values of a function and the values of its derivative on the boundary of the domain. For instance:

$$au + b\frac{\partial u}{\partial n}y = g \text{ on } \partial\Omega \tag{24}$$

In 1-D, on the interval $\Omega = [0, 1]$ we could for instance have:

$$au(0) - Bu'(0) = g(0) \tag{25}$$

$$au(1) + Bu'(1) = g(1) \tag{26}$$

This contrasts with **mixed boundary conditions**, which are boundary conditions of different types specified on different subsets of the boundary.

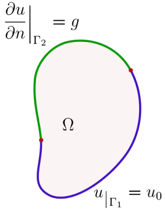


Figure: Mixed boundary conditions

Appendix: why truncated advection is imperfect.

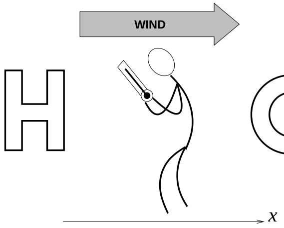


Figure: Temperature at a point increases when wind blows from the warm side.

In Figure 4, the wind is bringing air from the $-x$ direction. If the air 'upstream' is warmer than the air 'downstream', the observer will see the temperature increasing. The rate of change of temperature observed must depend on both the magnitude of this gradient and the speed at which the air is moving, *i.e.,*

$$\frac{\partial T}{\partial t} = -u\frac{\partial T}{\partial x}. \tag{27}$$

where the left hand side means the rate of change at a given position x and the right hand side is called the *advection term*. u is the wind component and $\partial T/\partial x$ means the gradient in the x-direction at a given time.
We can solve this analytically, but the question is what happens when we solve it numerically. As an example, let us look at CTCS.

Let us try using the centred difference in time and the centred difference in space (CTCS for short):

$$\frac{T_j^{n+1} - T_j^{n-1}}{2\Delta t} + u\frac{T_{j+1}^n - T_{j-1}^n}{2\Delta x} = 0 \tag{28}$$

We can rewrite this equation to get:

$$T_j^{n+1} = T_j^{n-1} - \frac{u\Delta t}{\Delta x}\left[T_{j+1}^n - T_{j-1}^n\right] \tag{29}$$

The quantity $u\Delta t/\Delta x$ is dimensionless. It appears so often in numerical schemes for the advection equation that it has its own name — the *Courant number*:

$$\alpha = \frac{u\Delta t}{\Delta x}. \tag{30}$$

The von Neumann stability analysis for the CTCS scheme is most easily done using complex notation for the trial solution,

$$T_j^n = e^{ikx_j} \quad ; \quad T_j^{n+1} = ST_j^n \tag{31}$$

where $S = Ae^{i\phi}$ expresses the change in amplitude and phase over one time-step.

When we substitute the trial solution $T_j^n = e^{ikx_j}$ into a CTCS scheme, we shall end up with three terms that express what temperature is at points:

$$j-1, \quad j \quad \text{and} \quad j+1 \tag{32}$$

These three terms are:

$$e^{ik(x_j-\Delta x)}, \quad e^{ikx_j} \quad \text{and} \quad e^{ik(x_j+\Delta x)} \tag{33}$$

The common term e^{ikx_j} cancels out, so that plugging the trial solution into the model (29) yields:

$$S = S^{-1} - \alpha\left[e^{ik\Delta x} - e^{-ik\Delta x}\right] \tag{34}$$

which can be re-arranged as a quadratic equation for the complex factor, S, by remembering Euler's formula: $\frac{e^{i\theta}-e^{-i\theta}}{2i} = sin(\theta)$:

$$S^2 + S\alpha 2i \sin k\Delta x - 1 = 0 \tag{35}$$

which has two roots:

$$S_{\pm} = -i\alpha \sin k\Delta x \pm \sqrt{1-\alpha^2 \sin^2 k\Delta x} \tag{36}$$

If the Courant number $|\alpha| \leq 1$, the number under the root must be positive since $\sin^2 k\Delta x \leq 1$.

Exercise: Show that when $|\alpha| \leq 1$ the magnitude of the amplification factor $|S|^2 = S^*S = 1$, meaning that the scheme is stable. However, when $|\alpha| > 1$ the S_- solution is unstable.

This is an improvement on the FTCS scheme, which was unconditionally unstable. This is the single most important result in the numerical modelling of fluid flows. It is called the Courant-Friedrichs-Lewy (CFL) condition. What it boils down to is that, for a given grid-spacing Δx, the time step must satisfy

$$\Delta t < \frac{\Delta x}{u}. \tag{37}$$

The physical interpretation of this condition is that *the scheme is unstable if fluid parcels move more than one gridbox in one timestep*.

The solution with $+$ve real part in (36) does not change sign every timestep and is called the *physical mode*. However, the solution with the -ve square root alternates sign every step which is unphysical behaviour and is called the *computational mode*. The relative amplitudes of the two modes is determined by projection from the initial conditions. The implication is that although the CTCS scheme is stable for $|\alpha| \leq 1$ we can expect unphysical oscillations that are worst at the grid-scale (where $\sin k\Delta x = 1$).