# Building a two-dimensional pseudo-spectral Code

Marek Stastna

# Introduction

- We have seen that building an FFT-based code is conceptually straight-forward.

- In this slide deck we want to keep the same framework, but consider a more complex set of equations in a more realistic set up.

- We will:

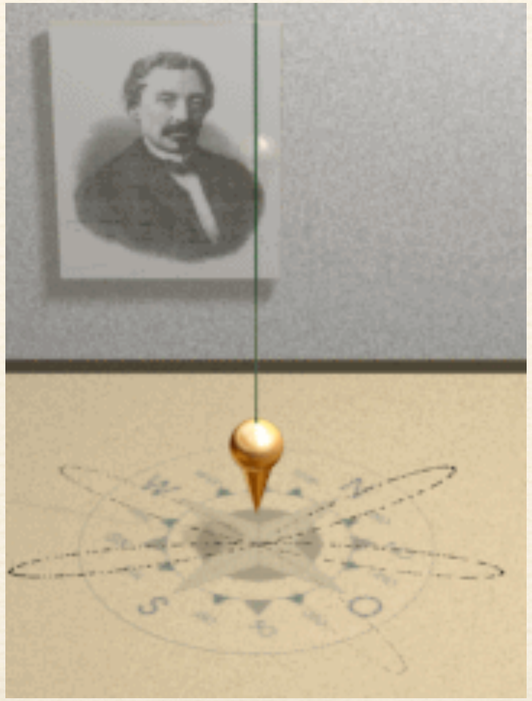
    1. Consider the shallow water equations on the f-plane in two-dimensions

    2. Use this system to identify possible issues, and offer a numerical remedy for these issues

    3. Show an alternative that uses physics to improve the equations

# Rotation

- Any measurement made on Earth is one made in a rotating reference frame due to the rotation of the planet about its axis.
- Rotation matters little for small scales, but even on moderate length scales, long time scales imply that rotation cannot be discounted (Foucault pendulum).
- The size and shape of the Earth also imply that in order to use Cartesian coordinates certain approximations must be made (see diagram on next slide).



## Assorted Demos on Youtube

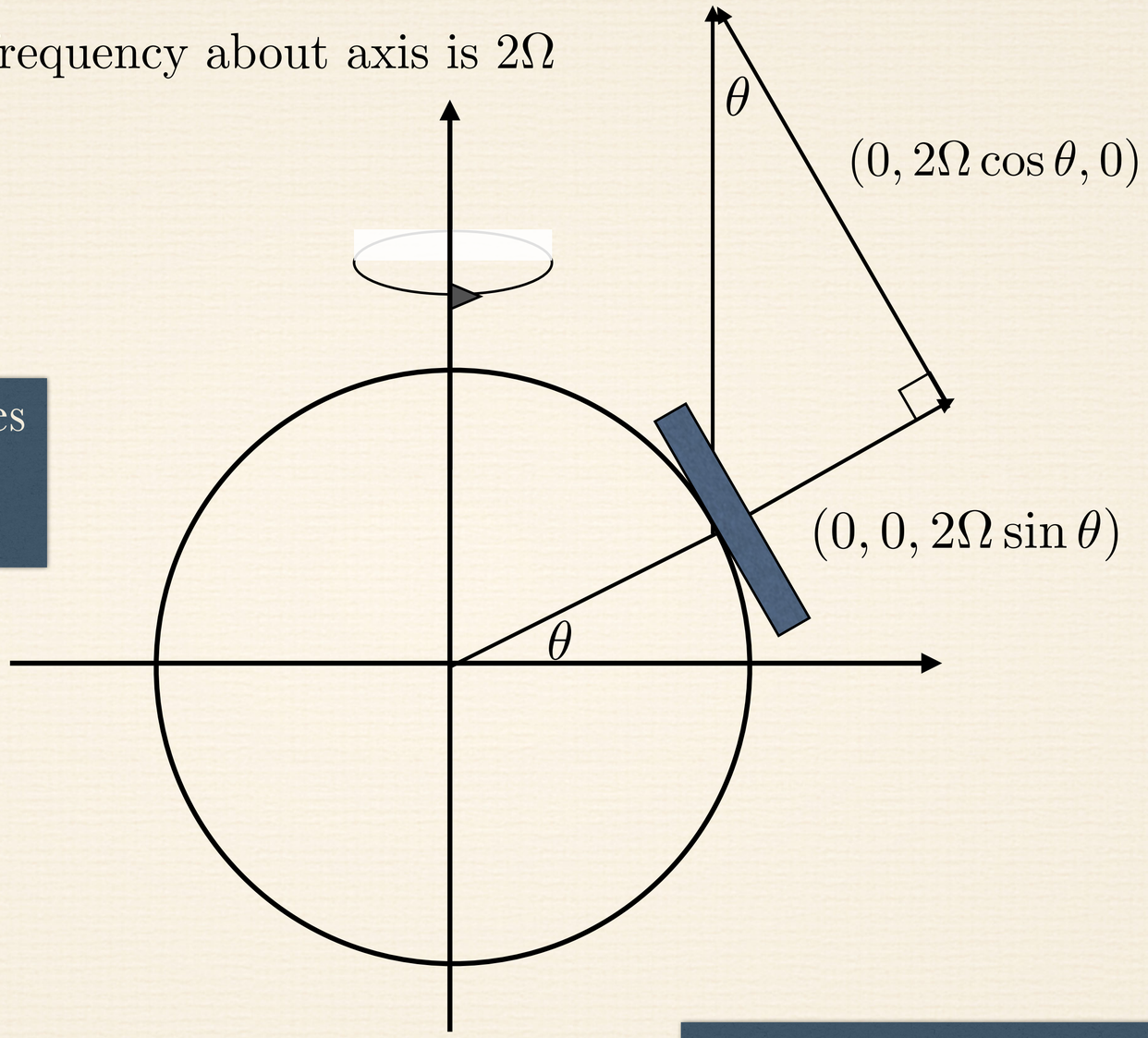
https://www.youtube.com/watch?v=pKGgICawAKc

https://www.youtube.com/watch?v=QAZPJakvabA

https://www.youtube.com/watch?v=7TjOy56-x8Q

**Large scale motions on Earth must account for planetary rotation**

Rotation frequency about axis is $2\Omega$




To get Cartesian coordinates use a tangent plane to the sphere at the local latitude

In local Cartesian coordinates the rotation vector has two nonzero components: $(0, f^*, f)$

One could also write $2\Omega\sin\theta = 2\Omega_{local}$

**Detailed slides for derivation available on request**

$$\left(\frac{d\vec{r}}{dt}\right)_F = \left(\frac{d\vec{r}}{dt}\right)_R + \vec{\Omega}\times\vec{r} \text{ where } \vec{r} \text{ is the position}$$

$$\vec{u}_F = \vec{u}_R + \vec{\Omega}\times\vec{r} \text{ for the velocity}$$

$$\frac{d\vec{u}_F}{dt} = \frac{d}{dt}(\vec{u}_R + \vec{\Omega}\times\vec{r})_R + \vec{\Omega}\times(\vec{u}_R + \vec{\Omega}\times\vec{r})$$

$$\frac{d\vec{u}_F}{dt} = \left(\frac{d\vec{u}_R}{dt}\right)_R + \vec{\Omega}\times\left(\frac{d\vec{r}}{dt}\right)_R + \vec{\Omega}\times\vec{u}_R + \vec{\Omega}\times(\vec{\Omega}\times\vec{r})$$

$$\vec{a}_F = \vec{a}_R + 2\vec{\Omega}\times\vec{u}_R + \vec{\Omega}\times(\vec{\Omega}\times\vec{r})$$

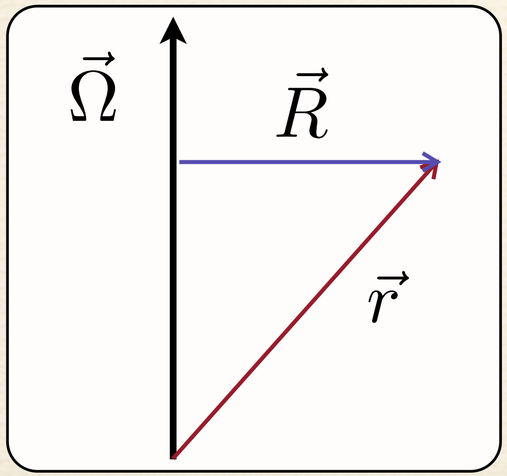
fixed frame acceleration $\vec{a}_F$

rotating frame acceleration $\vec{a}_R$

Coriolis acceleration $2\vec{\Omega}\times\vec{u}_R$

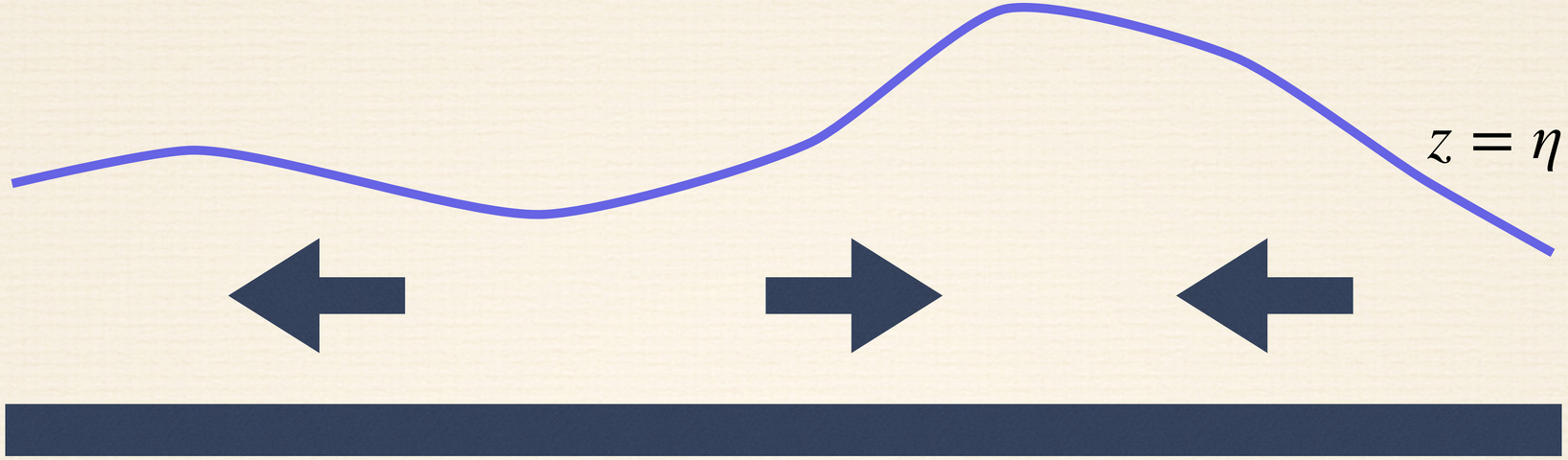
centripetal acceleration $-|\vec{\Omega}|^2\vec{R}$

often neglected




$$\vec{\Omega}\times(\vec{\Omega}\times\vec{r}) = \vec{\Omega}\times(\vec{\Omega}\times\vec{R}) = -|\vec{\Omega}|^2\vec{R}$$

# Background: Shallow Water Equations (with rotation)




- The simplest set of equations that apply to large scale motions.
- Assume the fluid moves up and down in columns and thus consider depth averaged velocity, only.
- Neglect viscosity, assume the bottom is flat (at z=-H) and let eta denote the free surface. Particles at the surface stay there and w(z=-H)=0.
- Assume that the pressure is hydrostatic so that the weight of the overlying fluid, determined by eta, drives motion through its horizontal gradients.
- The resulting conservation of mass equation states that the free surface rises and falls as response to local convergence or divergence of the mass flux.

$$z = \eta(x, y, t) \text{ defines the free surface}$$

$$\frac{D}{Dt}\left[z - \eta(x, y, t)\right] = 0 \text{ particles on free surface stay there}$$

$$w = \frac{D\eta}{Dt} \text{ at } z = \eta$$

$$\int_{-H}^{\eta} \nabla \cdot \vec{u} dz = 0 \text{ integrate the continuity equation}$$

$$w|_{z=-H}^{z=\eta} = -\nabla_H \cdot (\bar{u}, \bar{v}) \text{ integrate and rearrange}$$

$$\frac{\partial \eta}{\partial t} + \nabla \cdot ([H + \eta]u, [H + \eta]v) = 0 \text{ drop bars and subscripts}$$

Conservation of Mass

$$\frac{Du}{Dt} - fv = -g\frac{\partial \eta}{\partial x}$$

$$\frac{Dv}{Dt} + fu = -g\frac{\partial \eta}{\partial y}$$

Layer averaged momentum equations with pressure assumed to be hydrostatic:
$p = p_{reference} - \rho_0 g\eta$

Subscripts denote partial derivatives

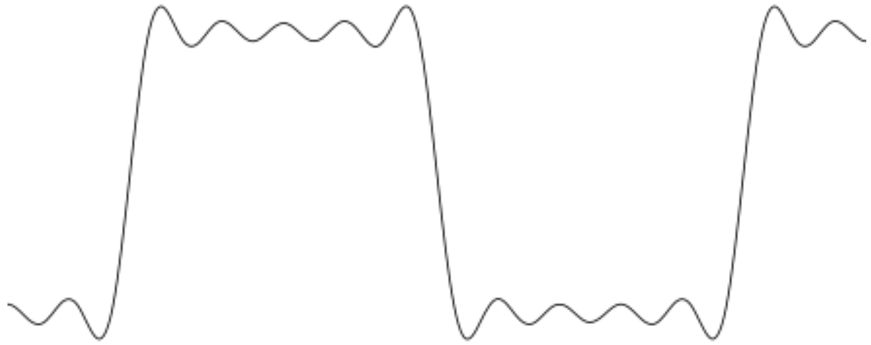
$$u_t + uu_x + vu_y - fv = -g\eta_x$$
$$v_t + uv_x + vv_y + fu = -g\eta_y$$
$$\eta_t + (u\eta)_x + (v\eta)_y = 0$$

- We want to solve the above equations on a doubly periodic domain using an FFT based method.
- Before we get to some software carpentry, consider what the possible problematic aspects of the above equations are?
- It is clearly the nonlinear terms, which appear to have nothing to balance them (because shallow water equations are an inviscid theory).
- So first consider the model of the inviscid Burgers equation $A_t + AA_x = 0$

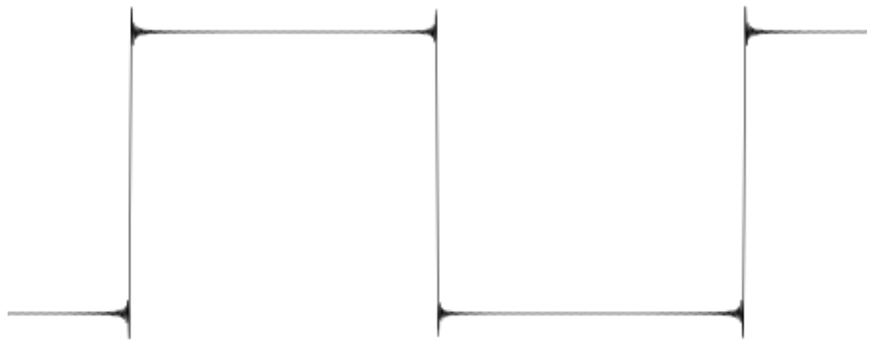
- We used $A_t + AA_x = 0$ as the inviscid limit of the Burgers equation to illustrate the formation of shocks/discontinuities.

- This is problematic for a Fourier Method due to the well known Gibbs phenomenon.

- When approximating functions with a jump, the Fourier series has locally slowed convergence.



5 sinusoids



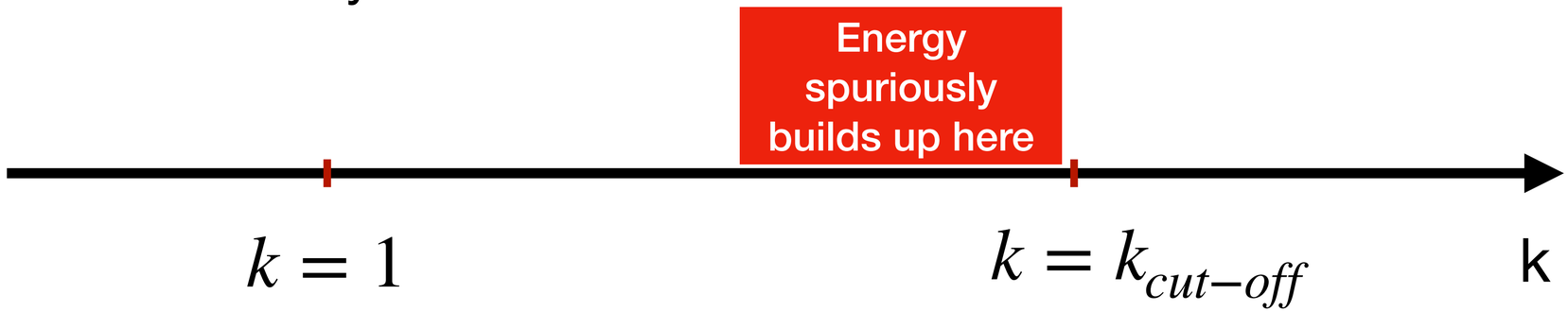
125 sinusoids

- If we used an FFT based, explicit method then $\bar{A}^{(n+1)} = \bar{A}^{(n)} - 0.5\Delta t i k \overline{A^2}^{(n)}$ in Fourier space and $A^{(n+1)} = A^{(n)} - \Delta t (A A_x)^{(n)}$ in physical space

- If we were to start with $A^{(0)} = \cos(x)$ a simple calculation shows that $A^{(1)} = \cos(x) + 0.5\Delta t \sin(2x)$ so that after one step we have activated a wavenumber that is twice the size of the one for the initial condition.

- With each subsequent step, higher and higher wavenumber end up being activated. But this cannot go on forever since there is a highest possible wavenumber for any given grid ($k_{cut-off}$ below).

- This process is called aliasing and requires action in any code because if left unchecked it invariably leads to code failure



- To understand how one may treat the aliasing issue consider the discretization of the Burgers equation from our last slide deck:
  $(1 + \nu k^2 \Delta t)\bar{B}^{(n+1)} = \bar{B}^{(n)} - ik\Delta t(\bar{B^2})^{(n)}$

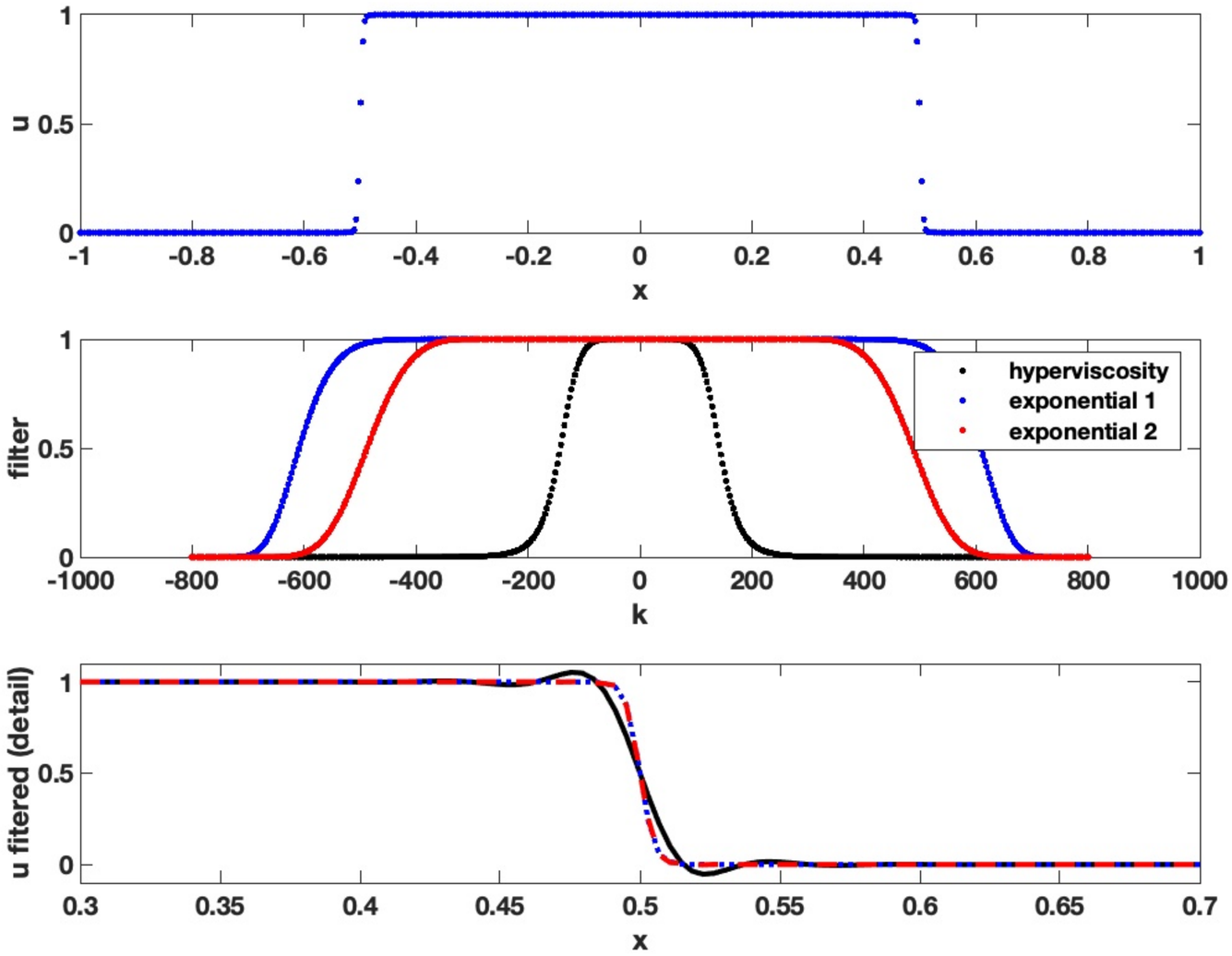
- The factor on the left hand side comes from the diffusion and the time stepping can be rewritten as $\bar{B}^{(n+1)} = M(k)\left(\bar{B}^{(n)} - ik\Delta t(\bar{B^2})^{(n)}\right)$ where $M(k)$ is a function that equals 1 when k=0 and is decreasing to zero as the magnitude of k increases.

- Engineers would call this a spectral filter. The one above comes from diffusion but in practice one could choose filters that are closer to 1 for a larger number of ks and then decrease rapidly as the cutoff wavenumber is approached.

- In practice, the filter represents phenomena neglected in the derivation of the shallow water equations.

# Filtering examples 1

- On the right we show a marginally resolved smooth “boxcar” function.
- In the middle panel we show three filters as functions of k. In black is a so-called hyperviscosity while in blue and red we have two exponential filters.
- In the bottom panel we show the detail of the filtered functions. You can see that all of them lead to little oscillations, though the hyperviscosity has the most obvious oscillations.



# Filtering examples 2

- Consider the inviscid Burgers equation $A_t + AA_x = 0$ and let's use something a bit better than forward Euler.

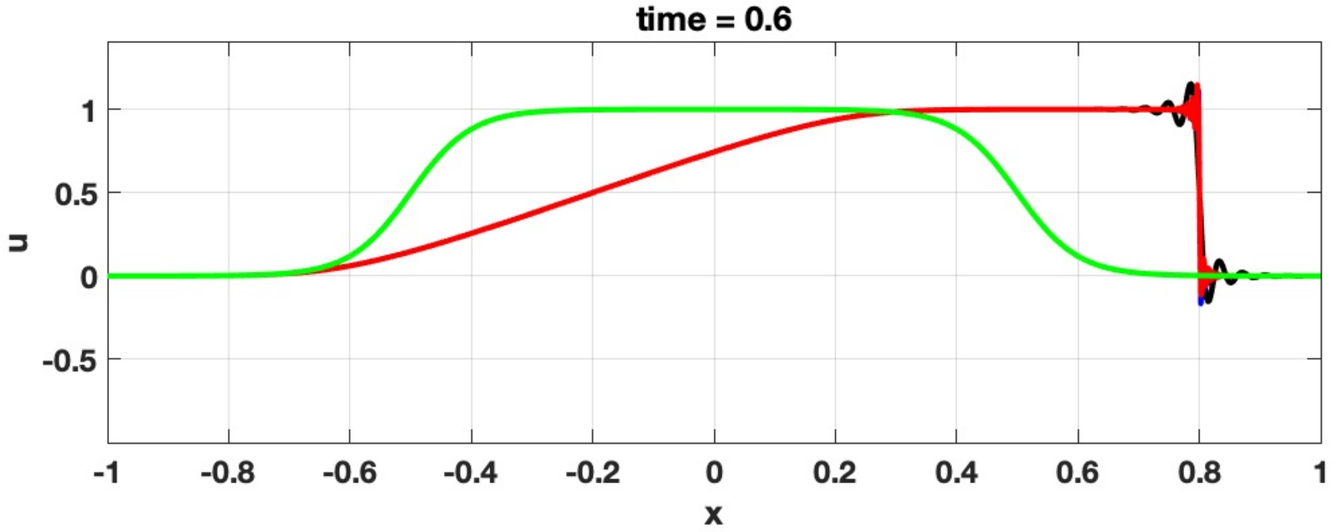
- Recall that entered differencing in space is second order, so let's use second order centre differencing in time: $\dfrac{A^{(n+1)} - A^{(n-1)}}{2\Delta t} = -(AA_x)^{(n)}$

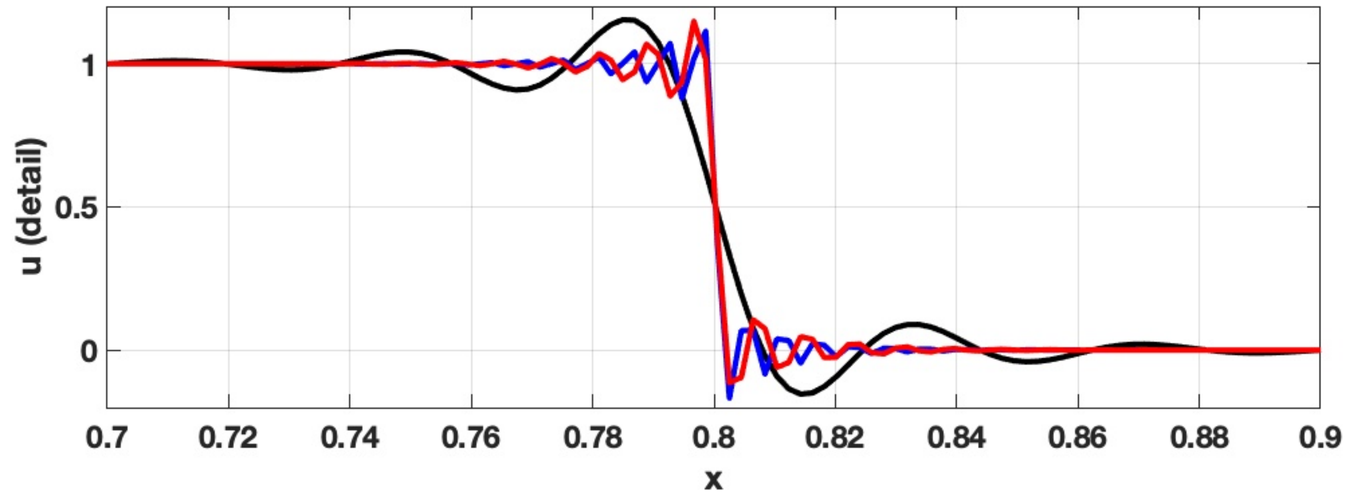
- Rearrange and take the FFT to get: $\bar{A}^{(n+1)} = \bar{A}^{(n-1)} - ik\Delta t(\bar{A^2})^{(n)}$

- And let's follow the forward step with an application of the filters from the last slide: $\bar{A}^{(n+1)} = \left(\bar{A}^{(n-1)} - ik\Delta t(\bar{A^2})^{(n)}\right) G(k)$

# Filtering examples 3

- On the right we show a numerical solution using 1024 points, with the three filters from two slides back.
- You can see that all the filters keep the solution bounded as the shock forms.
- They all lead to oscillations but for the exponential filters these are localized closer to the shock.
- Hyperviscosity leads to the longest oscillations.



Subscripts denote partial derivatives

$$u_t + uu_x + vu_y - fv = -g\eta_x$$
$$v_t + uv_x + vv_y + fu = -g\eta_y$$
$$\eta_t + (u\eta)_x + (v\eta)_y = 0$$

- If we consider the overbar to denote the **double FFT** then we can write out a time stepper for the above equations.
- However, we cannot get away with the trickery we did for the Burgers equations for all the nonlinear terms.
- This means some terms. e.g. $uv_x$, will require multiplication in physical space. There is some cost to this, and we make no claims that the algorithm presented is optimal in terms of complexity.

# Filtering the SW equations: code

```
for jj=1:numstps
   t=t+dt;
   % the main loop uses leapfrog
   unf=fft2(un);unf2=fft2(un2);
   unx=ifft2(i*kk.*unf);unx2=ifft2(i*kk.*unf2);
   uny=ifft2(i*ll.*unf);uny2=ifft2(i*ll.*unf2);
   vnf=fft2(vn);vnf2=fft2(vn2);
   vnx=ifft2(i*kk.*vnf);vnx2=ifft2(i*kk.*vnf2);
   vny=ifft2(i*ll.*vnf);vny2=ifft2(i*ll.*vnf2);
   enf=fft2(en);enf2=fft2(en2);
   enx=ifft2(i*kk.*enf);enx2=ifft2(i*kk.*enf2);
   eny=ifft2(i*ll.*enf);eny2=ifft2(i*ll.*enf2);
   enrhs=ifft2(i*kk.*fft2((H+en).*un)+i*ll.*fft2((H+en).*vn));
   enrhs2=ifft2(i*kk.*fft2((H+en2).*un2)+i*ll.*fft2((H+en2).*vn2));

   uf=up+twodt*(-un.*unx-vn.*uny-g*enx);
   uf2=up2+twodt*(-un2.*unx2-vn2.*uny2-g*enx2);
   vf=vp+twodt*(-un.*vnx-vn.*vny-g*eny);
   vf2=vp2+twodt*(-un2.*vnx2-vn2.*vny2-g*eny2);
   ef=ep+twodt*(-enrhs);
   ef2=ep2+twodt*(-enrhs2);

   % now rotate and filter as you go
   up=un; up2=un2; ep=en; ep2=en2; vp=vn; vp2=vn2;
   un=real(ifft2(myfilta.*fft2(uf)));
   un2=real(ifft2(myfiltb.*fft2(uf2)));
   vn=real(ifft2(myfilta.*fft2(vf)));
   vn2=real(ifft2(myfiltb.*fft2(vf2)));
   en=real(ifft2(myfilta.*fft2(ef)));
   en2=real(ifft2(myfiltb.*fft2(ef2)));
  end
```

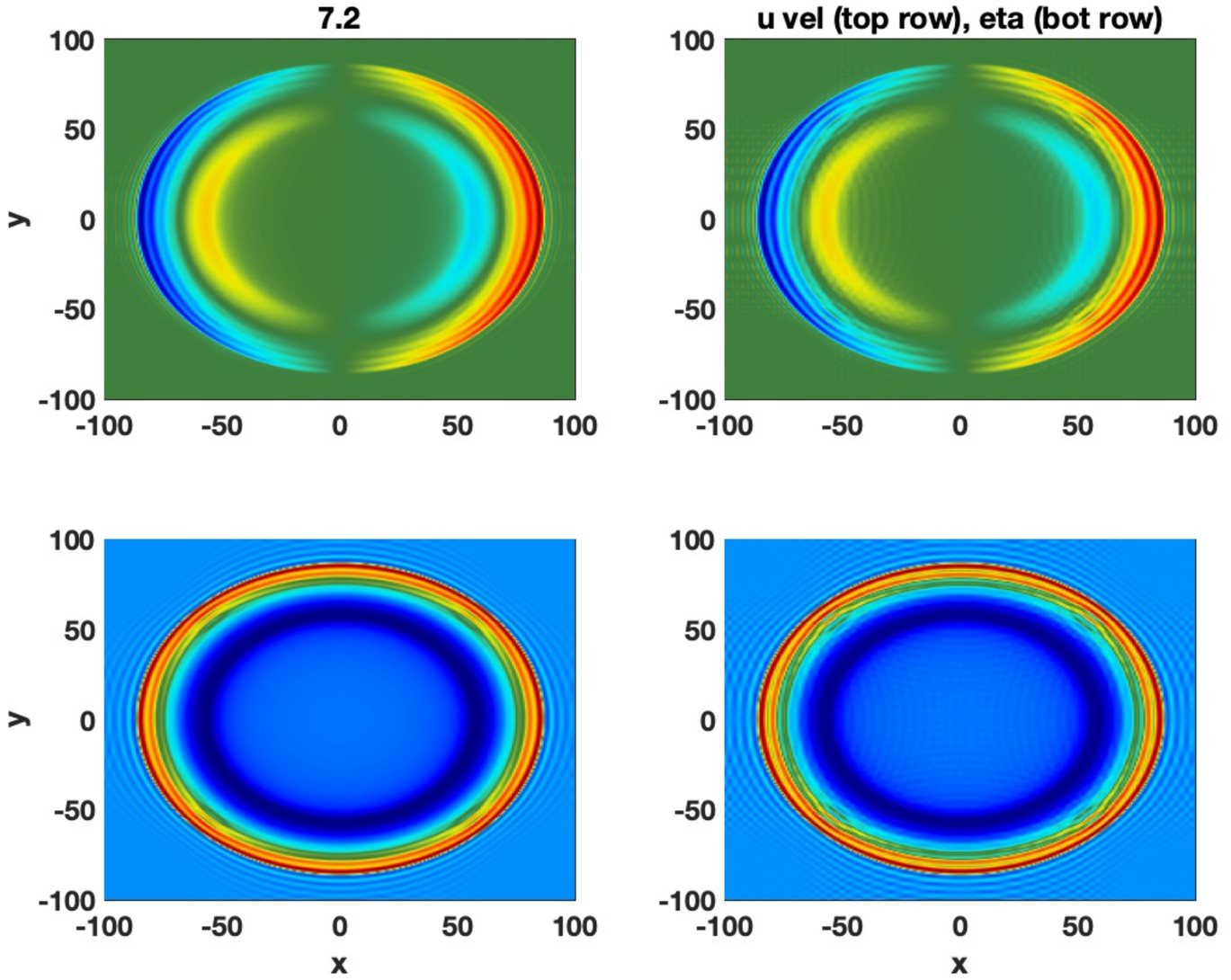
All the derivative pieces, including the nonlinear terms

This is the time stepping

This is the filtering

# Filtering the SW equations: results

- On the right we show a numerical solution using 256x256 points, with the two different filters.
- You can see that both of the filters keep the solution bounded as the waves propagate out.
- They also lead to oscillations.
- The oscillations are a reminder that in the real world we observe waves on many different scales and so we can ask whether mimicking dissipation is the only thing we can do as far as improving on the SW equations.



# Improving the SW equations: dispersion

$$\frac{Du}{Dt} - fv = -g\frac{\partial \eta}{\partial x}$$
$$\frac{Dv}{Dt} + fu = -g\frac{\partial \eta}{\partial y}$$

Layer averaged momentum equations with **pressure assumed to be hydrostatic**:

$p = p_{reference} - \rho_0 g \eta$

- The basic idea is to start with the general statement:
  $\frac{D\vec{u}}{Dt} + (-fv, fu) = -\nabla p$ and then write the pressure as $p = p_H + \epsilon p_{NH}$
  where the correction represents small, but not vanishing, non hydrostatic variations.

- A long perturbation calculation relates the correction to $\eta$ to finally yield:
  $$\frac{D\vec{u}}{Dt} + (-fv, fu) = -g\nabla\eta + \frac{H^2}{6}\nabla\nabla\cdot\vec{u}_t$$

**Notice the mixed space and time derivatives**

# Dispersive SW equations in 1D

Some people question the factor 1/6 so let's make it general and call the factor $\beta$

- Start with $\frac{D\vec{u}}{Dt} + (-fv, fu) = -g\nabla\eta + \frac{H^2}{6}\nabla\nabla\cdot\vec{u}_t$ and to get the essence drop rotation and consider 1D only. This gives:

- $u_t + uu_x = -g\eta_x + \beta u_{xxt}$ and if we take the Fourier transform and rearrange we get $(1+\beta k^2)\bar{u}_t = -0.5ik\overline{u^2} - gik\bar{\eta}$ and the dispersive term ends up as a multiplicative factor for the time derivative, and hence can just divide the RHS.

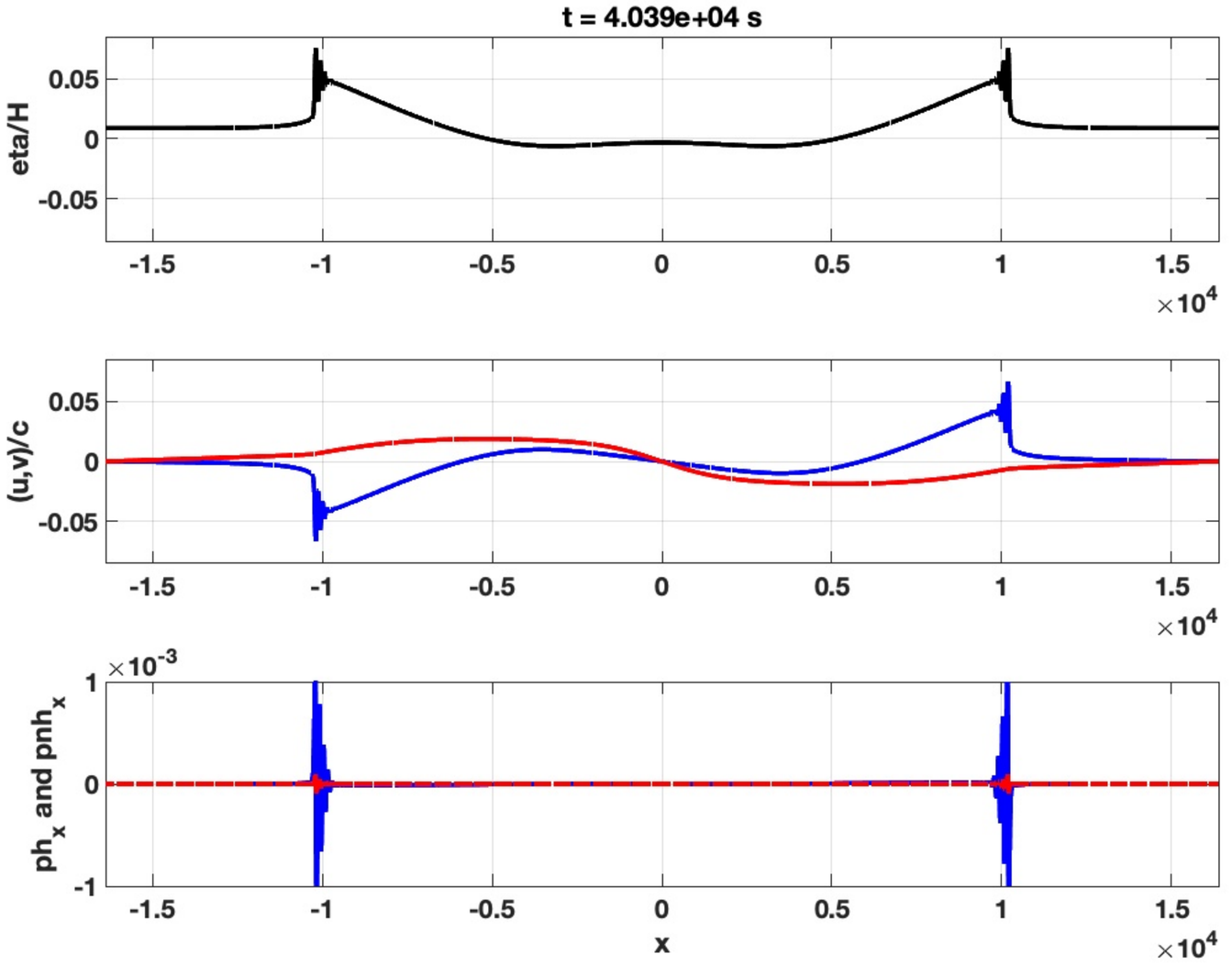
- With a leapfrog in time scheme we would get:

$$\bar{u}^{(n+1)} = \frac{1}{1+\beta k^2}\left(u^{(n-1)} - \Delta t ik\overline{u^2}^{(n)} - 2\Delta t g ik\bar{\eta}^{(n)}\right)$$

Notice how the dispersive factor matters more when $k^2$ gets larger. This is consistent with the physics.

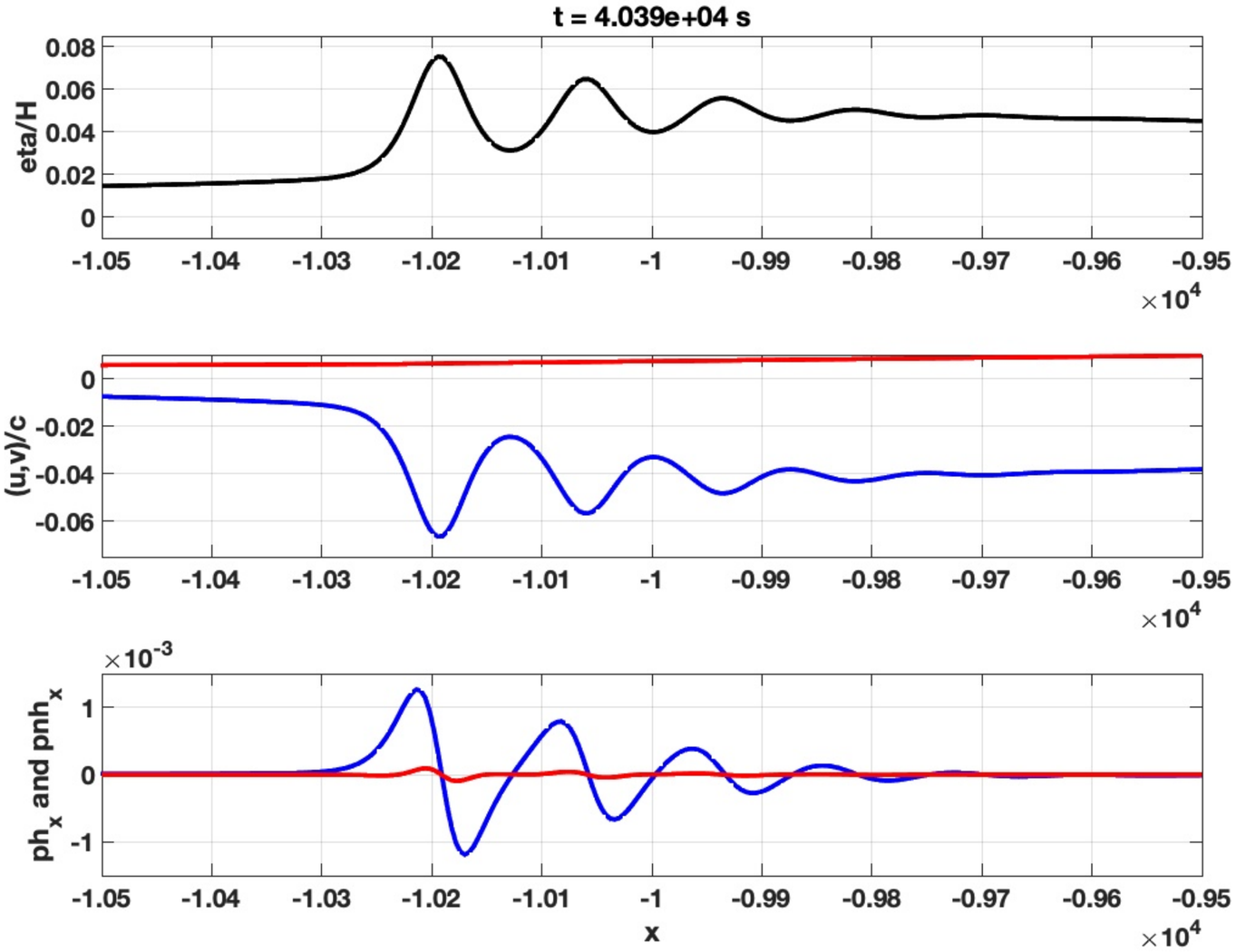
# NH SW example: full field

- On the right we show a numerical solution of the 1D SW equations with the non hydrostatic correction (hence NHSW) on the field scale.
- We have let the run go a long time (over an hour).
- You can see that all the action is located in isolated locations that look choppy on this plot.
- But in fact this simulation has a 4m grid spacing so we should have a closer look.



# NH SW example: detail

- A detailed look shows that adding dispersion has led to the evolutior of wave trains; sometimes called undular bores.
- There is no sign of shocks at all.
- The dispersive correction has allowed us to accurately simulate with no sign of filtering artefacts.
- Dispersion is thus a strong counterpoint to shocks.



- Spectral methods, together with an understanding of wave physics, allow us to accurately simulate the essential aspects of this image.
- They do so without an overwhelming amount of code overhead.
- They are not perfect (e.g. doing boundary conditions correctly) but they do a far better job than classical methods when it comes to the balance between applications and mathematics.