LES of Turbulent Flows: Lecture 14
(ME EN 7960-003)

Prof. Rob Stoll

Department of Mechanical Engineering

University of Utah

Fall 2014

Stochastic Burgers Equation

- Project #1 is based on a useful model of the Navier-Stokes equations: the 1D stochastic Burgers Equation:

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \nu\frac{\partial^2 u}{\partial x^2} + \eta(x,t)$$

- This analog has been found to be a useful one for the study of turbulent like nonlinear systems (e.g., Basu, 2009 and references contained within).

- Although 1D, this equation has some of the most important characteristics of a turbulent flow making it a good model case.

- in the above equation, the "new" term is η ➔ the stochastic term.

$$\eta(x,t) \Rightarrow \text{ should be white noise in time but spatially correllated}$$

- here we use

$$\eta(x,t) = \sqrt{2D_0 / \Delta t}\, \Im^{-1}\left\{ |k|^{\beta/2} \hat{f}(k) \right\}$$

$D_0 \equiv$ noise amplitude

$\Delta t \equiv$ time step

$\Im^{-1} \equiv$ inverse Fourier transform

$f \equiv$ Gaussian random noise with mean = 0

and standard deviation = $\sqrt{N}$ (where N is the # of pts)

$\beta \equiv$ spectral slose of the noise (taken as $-3/4$ here)

Stochastic Burgers Equation

- Many SBE solutions exist. Here we we follow Basu, 2009 (Fourier collocation). Basically, use Fourier methods but advance time in real space (compare to Galerkin)
- To do this, the main numerical methods we need to know are how to calculate derivatives and how to advance time.
- **Derivatives**:
  - Mathematically the discrete Fourier transform pair is (see also Lecture 2 suppliment):

Fourier transform pair:

(*) $$f(x_j) = \sum_{m=-N/2}^{N/2-1} \hat{f}(k_m) e^{ik_m x_j}$$ ← Backward transform

(**) $$\hat{f}(k_m) = \frac{1}{N}\sum_{j=1}^{N} f(x_j) e^{-ik_m x_j}$$ ← Forward transform

Fourier coefficients (wave amplitudes) → $\hat{f}(k_m)$

wave number (wave period) $\rightarrow k_m = \frac{2\pi m}{\Delta x N}$

recall: $e^{-ik_m x_j} = \cos(k_m x_j) + i\sin(k_m x_j)$

Fourier Derivatives

- How is this used numerically to calculate a derivative?
  -A Fourier series can be used to interpolate $f(x_j)$ at any point x in the flow and at any time t
  -If we differentiate the Fourier representation of $f(x_j)$ (equation ⊛) with respect to x

$$\frac{\partial f}{\partial x} = \frac{\partial}{\partial x}\left[\sum_{m=-N/2}^{N/2-1} \hat{f}(k_m) e^{ik_m x_j}\right] = \sum_{m=-N/2}^{N/2-1} \hat{f}(k_m) \frac{\partial e^{ik_m x_j}}{\partial x} = \boxed{\sum_{m=-N/2}^{N/2-1} ik_m \hat{f}(k_m) e^{ik_m x_j}}$$

- if we compare this to equation ⊛ we notice that we have:

$$\frac{\partial f}{\partial x} = g = \sum_{m=-N/2}^{N/2-1} \underbrace{ik_m \hat{f}(k_m)}_{\hat{g}(k_m)} e^{ik_m x_j} = \boxed{\sum_{m=-N/2}^{N/2-1} \hat{g}(k_m) e^{ik_m x_j}}$$

Fourier Derivatives

- procedurally we can use this to find $\left.\frac{\partial f}{\partial x}\right|_j$ given $f\left(x_j\right)$ as follows:

  -calculate $\hat{f}\left(k_m\right)$ from by a forward transform (equation **)

  -multiply by ik_m to get $\hat{g}\left(k_m\right)$ and then

  -perform a backward (inverse) transform using equation * to get $\left.\frac{\partial f}{\partial x}\right|_j$.

- the method easily generalizes to any order derivative

- although Fourier methods are quite attractive do to the high accuracy and near exact representation of the derivatives, they have a few important limitations.

  - $f\left(x_j\right)$ must be continuously differentiable

  - $f\left(x_j\right)$ must be periodic

  - grid spacing must be uniform

Time advancement

- Time advancement in this code is accomplished using a 2nd order Adams-Bashforth scheme.

- This is a basic extension of the Euler method (see Ferziger and Peric chapter 6)

    - This is a multipoint (in time) method
    - idea: fit a polynomial of desired order (e.g., 2nd) through 3 points in time to get:

$$\phi^{n+1} = \phi^{n} + \frac{\Delta t}{2}\left[3f\left(t^{n},\phi^{n}\right) - f\left(t^{n-1},\phi^{n-1}\right)\right]$$

- For specifics on how these things are implemented see the Matlab code on…