2.29 Numerical Fluid Mechanics
Fall 2011 – Lecture 14

REVIEW Lecture 12-13:

- Classification of PDEs and examples of finite-difference discretization
- Error Types and Discretization Properties
- Finite Differences based on Taylor Series Expansions
  - Higher Order Accuracy Differences, with Examples
    - Incorporate more higher-order terms of the Taylor series expansion than strictly needed and express them as finite differences themselves (making them function of neighboring function values)
    - If these finite-differences are of sufficient accuracy, this pushes the remainder to higher order terms => increased order of accuracy of the FD method
    - General approximation:

$$\boxed{\left(\frac{\partial^m u}{\partial x^m}\right)_j - \sum_{i=-r}^{s} a_i \, u_{j+i} = \tau_{\Delta x}}$$

  - Taylor Tables or Method of Undetermined Coefficients (Polynomial Fitting)
    - More systematic (Tables) way to solve for coefficients a_i of higher-order FD

2.29 Numerical Fluid Mechanics
Fall 2011 – Lecture 14

REVIEW Lecture 13, cont'd:

- **Finite Differences based Polynomial approximations**
  - Obtain polynomial (in general un-equally spaced), then differentiate if needed
    - Newton's interpolating polynomial formulas

Triangular Family of Polynomials (case of Equidistant Sampling, similar if not equidistant)

$$\boxed{\begin{aligned} f(x) &= f_0 + \frac{\Delta f_0}{h}(x-x_0) + \frac{\Delta^2 f_0}{2!h^2}(x-x_0)(x-x_1) + \cdots \\ &+ \frac{\Delta^n f_0}{n!h^n}(x-x_0)(x-x_1)\cdots(x-x_{n-1}) + \frac{f^{(n+1)}(\xi)}{(n+1)!}(x-x_0)\cdots(x-x_n) \end{aligned}}$$

- Lagrange polynomial

(Reformulation of Newton's polynomial)

$$\boxed{f(x) = \sum_{k=0}^{n} L_k(x) f(x_k) \quad \text{with} \quad L_k(x) = \prod_{j=0, j\neq k}^{n} \frac{x - x_j}{x_k - x_j}}$$

- Hermite Polynomials and Compact/Pade's Difference schemes

(Use the values of the function and its derivative(s) at nodes)

$$\boxed{\sum_{i=-r}^{s} b_i \left(\frac{\partial^m u}{\partial x^m}\right)_{j+i} - \sum_{i=-p}^{q} a_i \, u_{j+i} = \tau_{\Delta x}}$$

FINITE DIFFERENCES – Outline for Today

- Polynomial approximations
  - Newton's formulas
  - Lagrange polynomial and un-equally spaced differences
  - Hermite Polynomials and Compact/Pade's Difference schemes
- Finite Difference: Boundary conditions
- Finite-Differences on Non-Uniform Grids and Uniform Errors: 1-D
- Grid Refinement and Error Estimation:
  - Order of convergence, discretization error, Richardson's extrapolation and Iterative improvements using Roomberg's algorithm
- Fourier Analysis and Error Analysis
  - Differentiation, definition and smoothness of solution for ≠ order of spatial operators
- Stability
  - Heuristic Method
  - Energy Method
  - Von Neumann Method (Introduction)
- Hyperbolic PDEs

References and Reading Assignments

- Chapter 3 on "Finite Difference Methods" of "J. H. Ferziger and M. Peric, Computational Methods for Fluid Dynamics. Springer, NY, 3rd edition, 2002"

- Chapter 3 on "Finite Difference Approximations" of "H. Lomax, T. H. Pulliam, D.W. Zingg, *Fundamentals of Computational Fluid Dynamics (Scientific Computation).* Springer, 2003"

- Chapter 23 on "Numerical Differentiation" and Chapter 18 on "Interpolation" of "Chapra and Canale, Numerical Methods for Engineers, 2010/2006."

Finite Difference Schemes: Implementation of Boundary conditions

- For unique solutions, information is needed at boundaries
- Generally, one is given either:

i) the variable: $u\,(x = x_{\text{bnd}}, t) = u_{\text{bnd}}(t)$ (Dirichlet BCs)

ii) a gradient in a specific direction, e.g.: $\left.\dfrac{\partial u}{\partial x}\right|_{(x_{\text{bnd}}, t)} = \phi_{\text{bnd}}(\text{t})$ (Neumann BCs)

iii) a linear combination of the two quantities (Robin BCs)

- Straightforward cases:
  - If value is known, nothing special needed (one doesn't solve for the BC)
  - If derivatives are specified, for first-order schemes, this is also straightforward to treat

Finite Difference Schemes: Implementation of Boundary conditions, Cont'd

- Harder cases: when higher-order approximations are used
  - At and near the boundary: nodes outside of domain would be needed
- Remedy: use different approximations at and near the boundary
  - Either, approximations of lower order are used
  - Or, approximations go deeper in the interior and are one-sided. For example,
    - 1st order forward-difference: $\left.\frac{\partial u}{\partial x}\right|_{(x_{\text{bnd}},t)} = 0 \quad \Rightarrow \quad \frac{u_2 - u_1}{x_2 - x_1} \approx 0 \quad \Rightarrow u_2 = u_1$
    - Parabolic fit to the bnd point and two inner points:

$$\left.\frac{\partial u}{\partial x}\right|_{(x_{\text{bnd}},t)} \approx \frac{-u_3(x_2-x_1)^2 + u_2(x_3-x_1)^2 - u_1\left[(x_3-x_1)^2 - (x_2-x_1)^2\right]}{(x_2-x_1)(x_3-x_1)(x_3-x_2)} \qquad \left(\approx \frac{-u_3 + 4\,u_2 - 3u_1}{2\Delta x} \text{ for equidistant nodes}\right)$$

    - Cubic fit to 4 nodes (3rd order difference): $\left.\frac{\partial u}{\partial x}\right|_{(x_{\text{bnd}},t)} \approx \frac{2u_4 - 9u_3 + 18\,u_2 - 11u_1}{6\Delta x} + O(\Delta x^3)$ for equidistant nodes
    - Compact schemes, cubic fit to 4 pts: $u_{(x_{bnd},t)} = u_1 \approx \frac{18\,u_2 - 9u_3 + 2u_4}{11} - \frac{6\Delta x}{11}\left(\frac{\partial u}{\partial x}\right)_1$ for equidistant nodes
- In Open-boundary systems, boundary problem is not well posed =>
  - Separate treatment for inflow/outflow points, multi-scale approach and/or generalized inverse problem (using data in the interior)

Finite-Differences on Non-Uniform Grids: 1-D

- Truncation error depends not only on grid spacing but also on the derivatives of variable

$$f(x_{i+1}) = f(x_i) + \Delta x\, f'(x_i) + \frac{\Delta x^2}{2!} f''(x_i) + \frac{\Delta x^3}{3!} f'''(x_i) + \ldots + \frac{\Delta x^n}{n!} f^n(x_i) + R_n$$

$$R_n = \frac{\Delta x^{n+1}}{n+1!} f^{(n+1)}(\xi)$$

- Uniform error distribution can not be achieved on a uniform grid => non-uniform grids
  – Use smaller (larger) Δx in regions where derivatives of the function are large (small) => uniform discretization error
  – However, in some approximation (centered-differences), specific terms cancel only when the spacing is uniform
- Example: let's define $\underline{\Delta x_{i+1} = x_{i+1} - x_i \;,\quad \Delta x_i = x_i - x_{i-1}}$

$$f(x) = f(x_i) + (x - x_i)\, f'(x_i) + \frac{(x-x_i)^2}{2!} f''(x_i) + \frac{(x-x_i)^3}{3!} f'''(x_i) + \ldots + \frac{(x-x_i)^n}{n!} f^n(x_i) + R_n$$

and

$$R_n = \frac{(x-x_i)^{n+1}}{n+1!} f^{(n+1)}(\xi)$$

Non-Uniform Grids Example: 1-D Central-difference

- Evaluate $f(x)$ at x_{i+1} and x_{i-1} , subtract results, lead to central-difference

$$f(x_{i+1}) = f(x_i) + \Delta x_{i+1} f'(x_i) + \frac{\Delta x_{i+1}^{\ 2}}{2!} f''(x_i) + \frac{\Delta x_{i+1}^{\ 3}}{3!} f'''(x_i) + \ldots + \frac{\Delta x_{i+1}^{\ n}}{n!} f^n(x_i) + R_n$$

$$- \quad f(x_{i-1}) = f(x_i) - \Delta x_i f'(x_i) + \frac{\Delta x_i^{\ 2}}{2!} f''(x_i) - \frac{\Delta x_i^{\ 3}}{3!} f'''(x_i) + \ldots + \frac{(-\Delta x_i)^n}{n!} f^n(x_i) + R_n$$

$$f'(x_i) = \frac{f(x_{i+1}) - f(x_{i-1})}{x_{i+1} - x_{i-1}} \underbrace{- \frac{\Delta x_{i+1}^{\ 2} - \Delta x_i^{\ 2}}{2!\,(x_{i+1} - x_{i-1})} f''(x_i) - \frac{\Delta x_{i+1}^{\ 3} + \Delta x_i^{\ 3}}{3!\,(x_{i+1} - x_{i-1})} f'''(x_i) + \ldots + R_n}_{= \text{ Truncation error } \tau_{\Delta x}}$$

- For a non-uniform mesh, the leading truncation term is $O(\Delta x)$
  - The more non-uniform the mesh, the larger the 1st term in truncation error
  - If the grid contracts/expands with a constant factor r_e : $\boxed{\Delta x_{i+1} = r_e\, \Delta x_i}$
  - Leading truncation error term is : $\boxed{\tau_{\Delta x}^{r_e} \approx \frac{(1 - r_e)\, \Delta x_i}{2} f''(x_i)}$
  - If r_e is close to one, the first-order truncation error remains small: this is good for handling any types of unknown function $f(x)$

Non-Uniform Grids Example: 1-D Central-difference

- However, what matters is: "rate of error reduction as grid is refined"!
- Consider case where refinement is done by adding more grid points but keeping a constant ratio of spacing (geometric progression), i.e.

$$\Delta x_{i+1}^{2h} = r_{e,2h}\, \Delta x_i^{2h}$$

$$\Delta x_{i+1} = r_{e,h}\, \Delta x_i$$

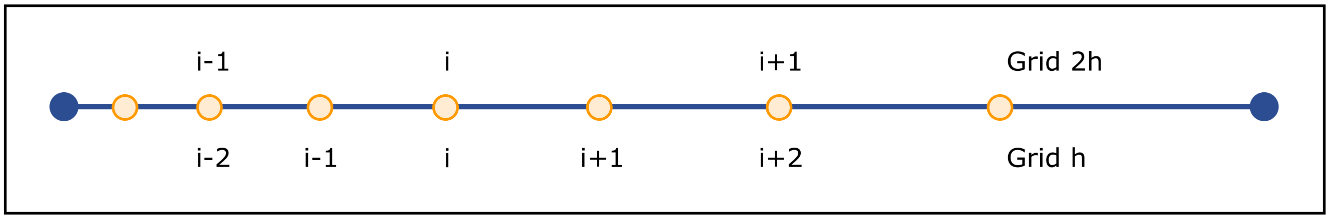


Image by MIT OpenCourseWare.

- For coarse grid pts to be collocated with fine-grid pts: $(r_{e,h})^2 = r_{e,2h}$
- The ratio of the two truncation errors at a common point is then:

$$R \approx \frac{\dfrac{(1-r_{e,2h})\,\Delta x_i^{2h}}{2} f''(x_i)}{\dfrac{(1-r_{e,h})\,\Delta x_i^{h}}{2} f''(x_i)} \quad \text{which is} \quad \boxed{R \approx \frac{(1+r_{e,h})^2}{r_{e,h}}} \quad \text{since} \quad \Delta x_i^{2h} = \Delta x_i + \Delta x_{i-1} = (r_{e,h}+1)\,\Delta x_{i-1}$$

  - The factor $R = 4$ if $r_e = 1$ (uniform grid)
  - When $r_e > 1$ (expending grid) or $r_e < 1$ (contracting grid), the factor $R > 4$

Non-Uniform Grids Example: 1-D Central-difference Conclusions

- When a non-uniform grid is refined, error due to the 1st order term decreases faster than that of 2nd order term !
- Since $(r_{e,h})^2 = r_{e,2h}$, we have $r_{e,h} \rightarrow 1$ as the grid is refined. Hence, convergence becomes asymptotically 2nd order (1st order term cancels)
- Non-uniform grids are thus useful, if one can reduce Δx in regions where derivatives of the unknown solution are large
  - Automated means of adapting the grid to the solution (as it evolves)
  - However, automated grid adaptation schemes are more challenging in higher dimensions and for multivariate (e.g. physics-biology-acoustics) or multiscale problems
- (Adaptive) Grid generation still a very challenging problem in CFD
- Conclusions also valid for higher dimensions and for other methods (finite elements, etc)

Grid-Refinement and Error estimation

- We found that for a convergent scheme, the discretization error ε is of the form: $\varepsilon = \alpha\, O(\Delta x^p) + R$ (recall: $\phi = \hat{\phi} + \varepsilon, \quad L(\phi) = 0, \quad \hat{L}_{\Delta x}(\hat{\phi}) = 0$)

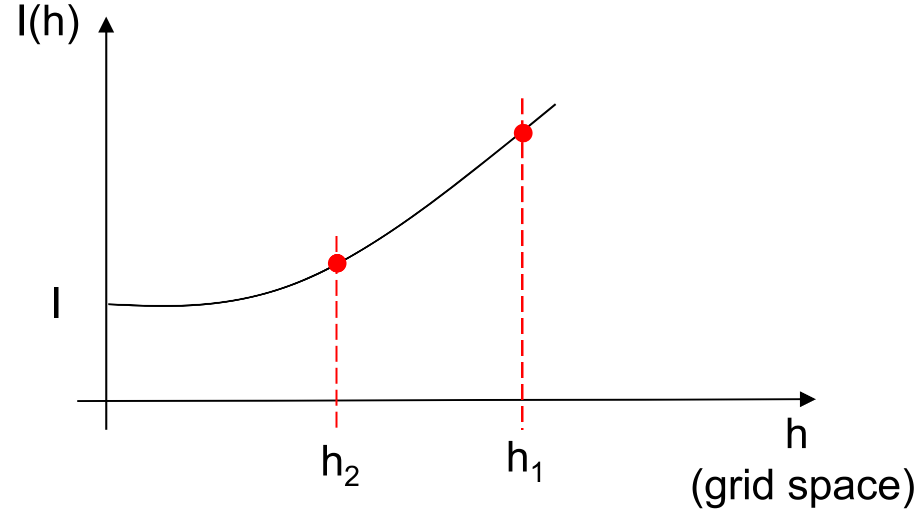
  where R is the remainder

- This discretization error can be estimated between solutions obtained on systematically refined/coarsened grids
  - True solution u can be expressed either as: $\begin{cases} u = u_{\Delta x} + \beta\, \Delta x^p + R \\ u = u_{2\Delta x} + \beta'\,(2\Delta x)^p + R' \end{cases}$
  - Thus, the exponent p can be estimated: $\boxed{p \approx \log\left(\dfrac{u_{2\Delta x} - u_{4\Delta x}}{u_{\Delta x} - u_{2\Delta x}}\right) \Big/ \log 2}$

    (need two equations to eliminate both Δx and p, hence $u_{4\Delta x}$)
  - The discretization error on the grid Δx can be estimated by: $\boxed{\varepsilon_{\Delta x} \approx \dfrac{u_{\Delta x} - u_{2\Delta x}}{2^p - 1}}$
  - Good idea: estimate *p* to check code. Is it equal to what it is supposed to be?
  - When solutions on several grids are available, an approximation of higher accuracy can be obtained from the remainder: Richardson Extrapolation!

Richardson Extrapolation and Romberg Integration

Richardson Extrapolation: method to obtain a third improved estimate of an integral based on two other estimates



Consider:

$$I = I(h) + E(h)$$

For two different grid space h1 and h2:

$$I(h_1) + E(h_1) = I(h_2) + E(h_2)$$

Trapezoidal Rule:

$$E(h) = -\frac{b-a}{12} h^2 \bar{f}''$$

$$\Rightarrow \; E(h_1) \approx E(h_2)\left(\frac{h_1}{h_2}\right)^2$$

$$I(h_1) + E(h_2)\left(\frac{h_1}{h_2}\right)^2 \simeq I(h_2) + E(h_2)$$

$$E(h_2) \simeq \frac{I(h_1) - I(h_2)}{1 - (h_1/h_2)^2}$$

Richardson Extrapolation:

$$I = I(h_2) + \frac{I(h_2) - I(h_1)}{(h_1/h_2)^2 - 1} + O(h^4)$$

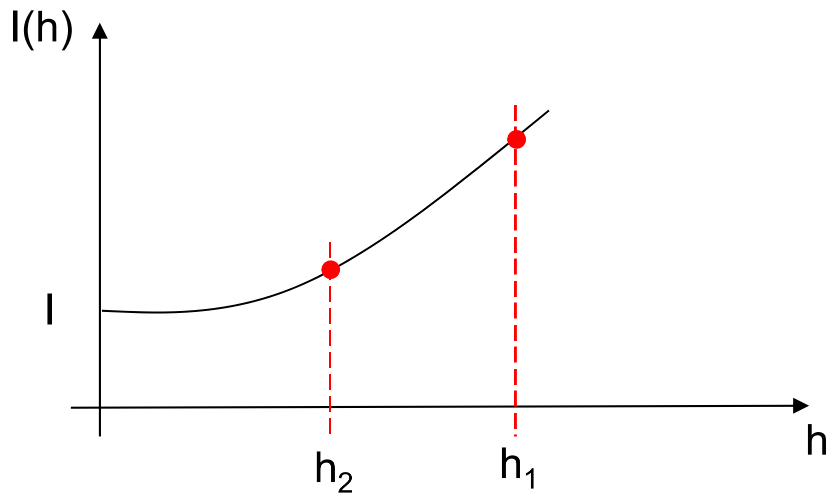
Example

Assume: $h_2 = h_1/2$

$$\begin{aligned} I &= I(h_2) + \frac{I(h_2) - I(h_1)}{(2^2 - 1)} + O(h^4) \\ &\boxed{= \frac{4}{3} I(h_2) - \frac{1}{3} I(h_1) + O(h^4)} \end{aligned}$$

From two $O(h^2)$, we get an $O(h^4)$!

Romberg's Integration:
Iterative application of Richardson's extrapolation



Romberg Integration Algorithm, for any order *k*

$$\boxed{I_{j,k} \simeq \frac{4^{k-1} I_{j+1,k-1} - I_{j,k-1}}{4^{k-1} - 1}}$$

For Order 2 (case of previous slide):

$$k = 2, j = 1$$

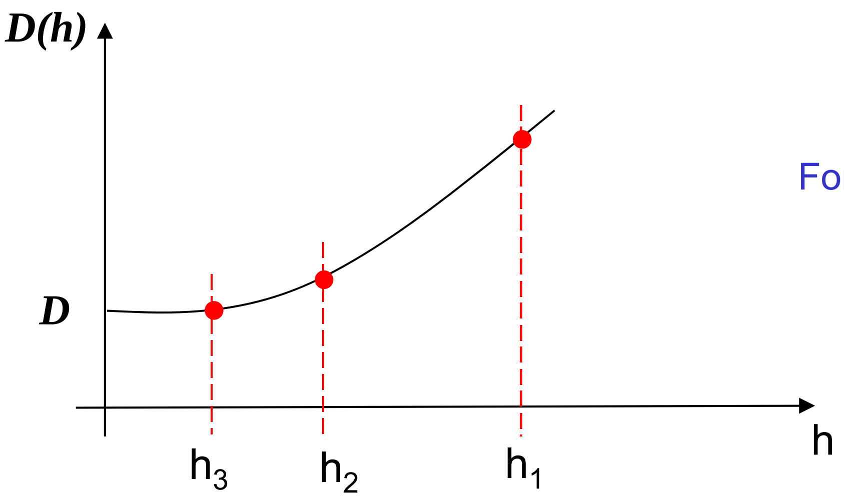
$$\boxed{I_{1,2} \simeq \frac{4 I_{2,1} - I_{1,1}}{3}}$$

$k \longrightarrow$

$j \downarrow$

	1: O(h^2)	2: O(h^4)	3: O(h^6)	4: O(h^8)
a.	0.172800	1.367467		
	1.068800			
b.	0.172800	1.367467	1.640533	
	1.068800	1.623467		
	1.484800			
c.	0.172800	1.367467	1.640533	1.640533
	1.068800	1.623467	1.640533	
	1.484800	1.639467		
	1.600800			

Romberg's Differentiation:
Iterative application of Richardson's extrapolation



'Romberg' Differentiation Algorithm, for any order *k*

$$\boxed{\boldsymbol{D}_{j,k} \simeq \frac{4^{k-1}\boldsymbol{D}_{j+1,k-1} - \boldsymbol{D}_{j,k-1}}{4^{k-1}-1}}$$

For Order 2 (as previous slide, but for differentiation):

$$k = 2, j = 1$$

$$\boxed{\boldsymbol{D}_{1,2} \simeq \frac{4\boldsymbol{D}_{2,1} - \boldsymbol{D}_{1,1}}{3}}$$

$k \longrightarrow$; $j \downarrow$

	1: O(h^2)	2: O(h^4)	3: O(h^6)	4: O(h^8)
a.	0.172800	1.367467		
	1.068800			
b,.	0.172800	1.367467	1.640533	
	1.068800	1.623467		
	1.484800			
c.	0.172800	1.367467	1.640533	1.640533
	1.068800	1.623467	1.640533	
	1.484800	1.639467		
	1.600800			

Fourier (Error) Analysis: Definitions

- Leading error terms and discretization error estimates can be complemented by a Fourier error analysis
- Fourier decomposition:
  - Any arbitrary periodic function can be decomposed into its Fourier components:

$$f(x) = \sum_{k=-\infty}^{\infty} f_k \, e^{ikx} \quad (k \text{ integer, wavenumber})$$

$$\int_0^{2\pi} e^{ikx} e^{-imx} = 2\pi \delta_{km} \quad \text{(orthogonality property)}$$

Using the orthog. property, taking the integral/FT of $f(x)$:

$$f_k = \frac{1}{2\pi} \int_0^{2\pi} f(x) \, e^{-ikx} dx$$

  - Note: rate at which $|f_k|$ with $|k|$ decays determine smoothness of $f(x)$
    - Examples drawn in lecture: $\sin(x)$, Gaussian $exp(-\pi x^2)$, multi-frequency functions, etc

Fourier (Error) Analysis: Differentiations

- Consider the decompositions:

$$f(x) = \sum_{k=-\infty}^{\infty} f_k \, e^{ikx} \qquad \text{or} \qquad f(x,t) = \sum_{k=-\infty}^{\infty} f_k(t) \, e^{ikx}$$

- Taking spatial derivatives gives: $$\frac{\partial^n f}{\partial x^n} = \sum_{k=-\infty}^{\infty} f_k(t) \, \left(ik\right)^n \, e^{ikx}$$

- Taking temporal derivatives gives: $$\frac{\partial^r f}{\partial t^r} = \sum_{k=-\infty}^{\infty} \frac{d^r f_k(t)}{dt^r} \, e^{ikx}$$

- Hence, in particular, for even or odd spatial derivatives:

$$n = 2q \qquad \Rightarrow \quad \left(ik\right)^n = (-1)^q \, k^{2q} \qquad \text{(real)}$$

$$n = 2q-1 \qquad \Rightarrow \quad \left(ik\right)^n = -i\,(-1)^q \, k^{2q-1} \qquad \text{(imaginary)}$$

Fourier (Error) Analysis: Generic equation

- Consider the generic PDE: $$\boxed{\frac{\partial f}{\partial t} = \frac{\partial^n f}{\partial x^n}}$$

- Fourier Analysis: $$f(x,t) = \sum_{k=-\infty}^{\infty} f_k(t)\, e^{ikx}$$

- Hence: $$\sum_{k=-\infty}^{\infty} \frac{d\, f_k(t)}{d\, t}\; e^{ikx} = \sum_{k=-\infty}^{\infty} f_k(t)\; \left(ik\right)^n\; e^{ikx}$$

- Thus: $$\boxed{\frac{d\, f_k(t)}{d\, t} = \left(ik\right)^n\; f_k(t) = \sigma f_k(t) \qquad \text{for} \;\; \sigma = \left(ik\right)^n}$$

- And: $$f_k(t) = f_k(0)\;\; e^{\sigma t}, \qquad f(x,t) = \sum_{k=-\infty}^{\infty} f_k(0)\, e^{ikx+\sigma t}$$

  – "Phase speed": $c = -\sigma / ik$

Fourier (Error) Analysis: Generic equation

- Generic PDE, FT:

$$\boxed{\frac{d f_k(t)}{d t} = (ik)^n \; f_k(t) = \sigma f_k(t) \qquad \text{for } \; \sigma = (ik)^n}$$

$$n = 2q \qquad \Rightarrow \quad (ik)^n = (-1)^q \; k^{2q} \qquad \text{(real)}$$

$$n = 2q-1 \qquad \Rightarrow \quad (ik)^n = -i\,(-1)^q \; k^{2q-1} \qquad \text{(imaginary)}$$

- Hence:

$n=1$	$\dfrac{\partial f}{\partial t} = \dfrac{\partial f}{\partial x}$	$\sigma = ik$	Propagation: $c = -\sigma / ik = -1$, No dispersion
$n=2$	$\dfrac{\partial f}{\partial t} = \dfrac{\partial^2 f}{\partial x^2}$	$\sigma = -k^2$	Decay
$n=3$	$\dfrac{\partial f}{\partial t} = \dfrac{\partial^3 f}{\partial x^3}$	$\sigma = -ik^3$	Propagation: $c = -\sigma / ik = +k^2$, With dispersion
$n=4$	$\dfrac{\partial f}{\partial t} = \pm \dfrac{\partial^4 f}{\partial x^4}$	$\sigma = \pm k^4$	$+$: (Fast) Growth, $-$: (Fast) Decay

- Etc

2.29 Numerical Fluid Mechanics
Fall 2011