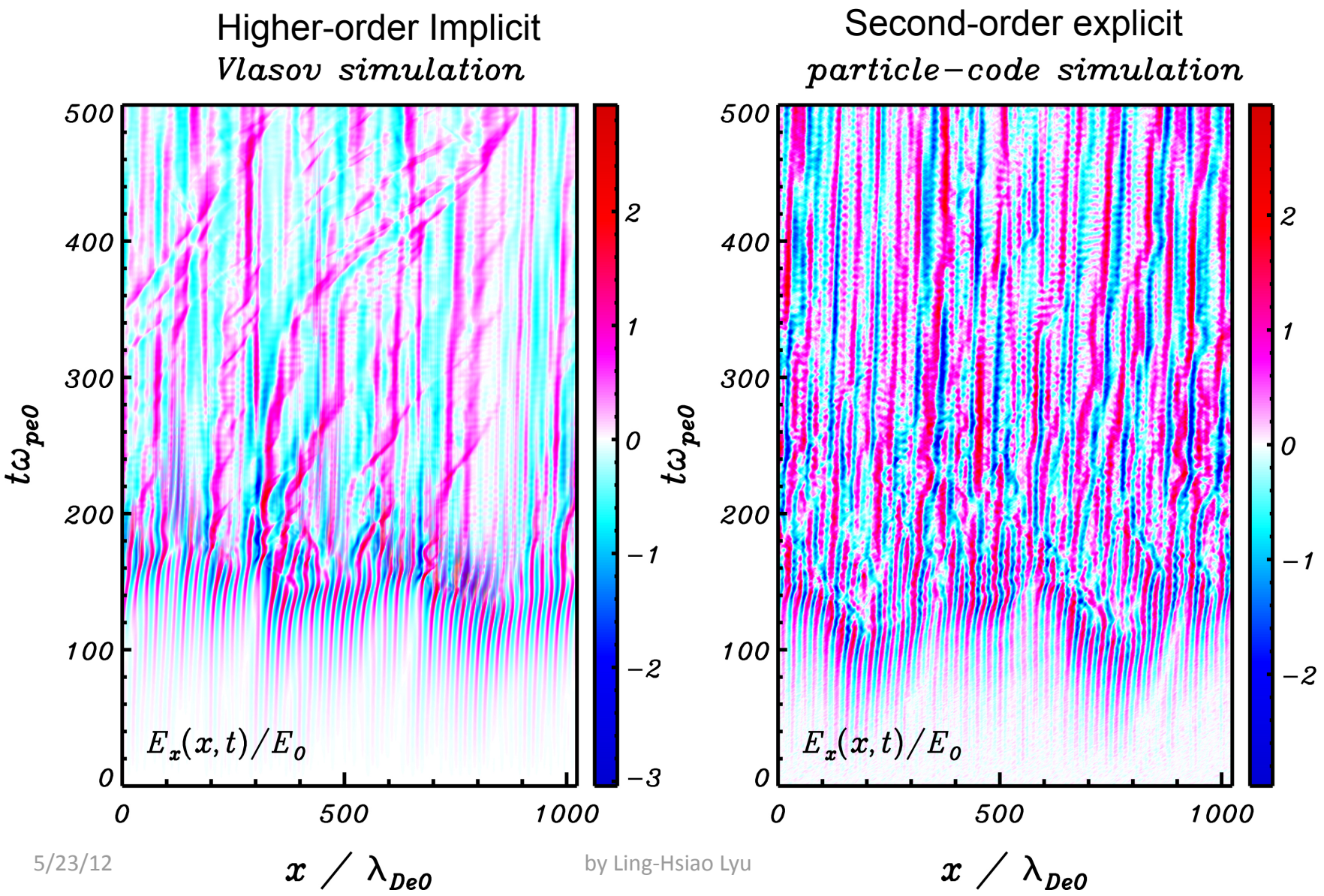
Particle Simulation

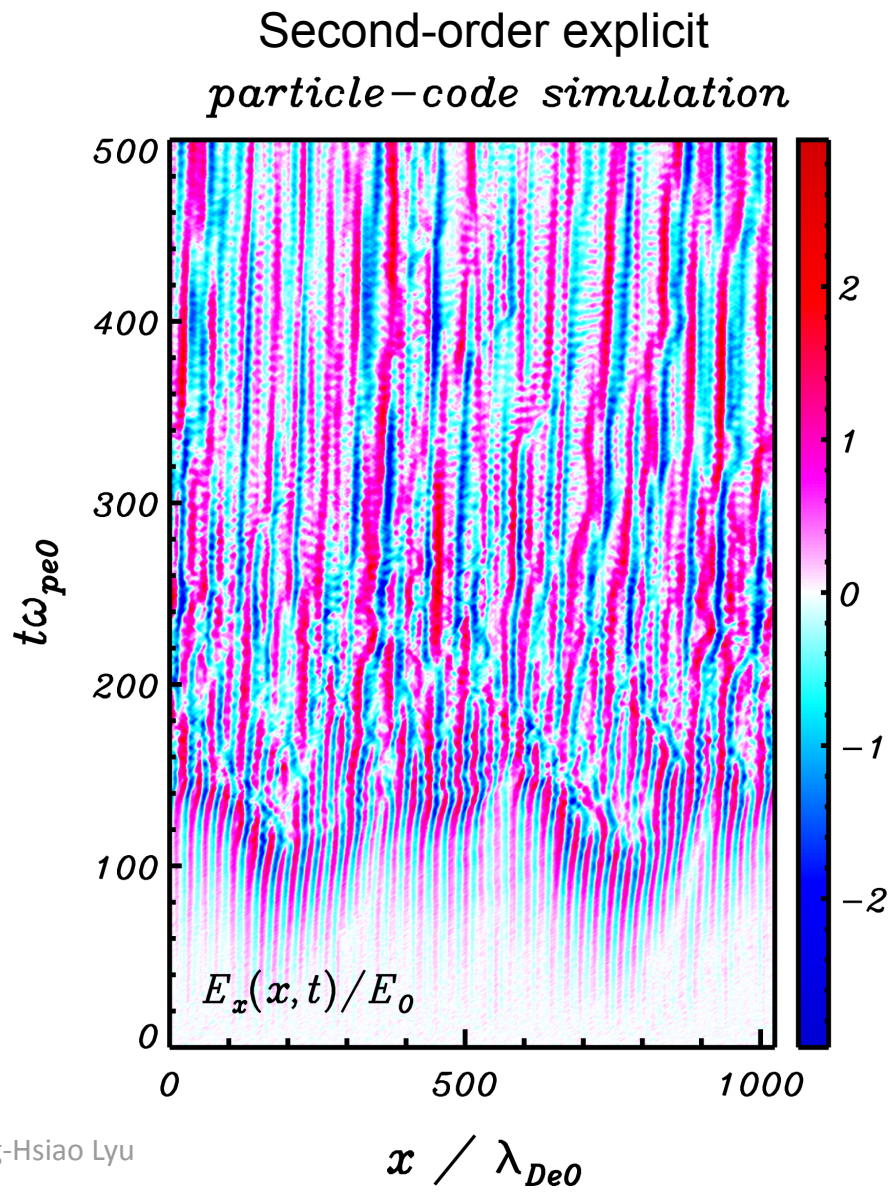
呂凌霄
Ling-Hsiao Lyu
Institute of Space Science, NCU

Motivation (Tsai et al., 2010 JGR)



Motivation (Tsai et al., 2010 JGR)

- The S/N ratio is too low in the second-order explicit particle simulation (with first-order distribution and interpolation).
- Will a higher-order implicit particle simulation recover the signal shown in the higher-order implicit Vlasov simulation ?
- By removing the run-off errors, we have successfully suppress the numerical instability in the higher-order finite difference scheme. It is time to build a higher-order implicit particle code for the cross-scale simulations.



Outlines

- Basic Concept of Particle Code Simulation
- Time Integrations
- Basic Equations of Particle Code Simulation
- Review of Classical Particle Simulation Models
  - The UCLA simulation model
  - The PIC (Particle-in-Cell) simulation model
- The New Implicit Higher-Order Particle Simulation Model

Basic Concepts
of Particle Code Simulation

Purpose:

To reduce the calculation steps from N^2 to NM, where N is the number of particles, M is the number of grids, and $N>>M$.

1. Distributing (an inverse interpolating) the charge density and current density to the grids
2. Solving the Maxwell's equation to advance the electric field and magnetic field on each grid
3. Interpolating the field from the grid points to the location of the particle
4. Advancing the velocity and position of a particle by solving the equations of motion

Time Integrations

Explicit Scheme:

- The future information are determined based on the present and the past information
- Easy to program, easy to blowout!
- To avoid blowout, one have to choose shorter time step.
- A short time step means more CPU time

Implicit Scheme:

- The future information are determined based on the future, the present, and the past information
- Difficult to program and/or require more memory
- Stable in large time step.
- Implicit scheme can save more CPU time and provide reliable results

Implicit Time Integration

Procedure of the 4th order Predictor-Corrector Method

Initial	Solving $dy/dt = f$ or $\partial y/\partial t = f$ with $h = \Delta t$ to obtain y^1, y^2, and y^3 from y^0 by the 4th order Runge-Kutta method
Predicting Step	Predicting y^{n+1} from y^n, y^{n-1}, y^{n-2}, and y^{n-3} by the 4th order Adams Open Formula: $y^{n+1} = y^n + h[\frac{55}{24} f^n - \frac{59}{24} f^{n-1} + \frac{37}{24} f^{n-2} - \frac{9}{24} f^{n-3}] + O(h^5 f^{(4)})$
Correcting Steps	Correcting y^{n+1} from y^{n-2}, y^{n-1}, y^n, and the last y^{n+1} by the 4th order Adams Close Formula: $y^{n+1} = y^n + h[\frac{9}{24} f^{n+1} + \frac{19}{24} f^n - \frac{5}{24} f^{n-1} + \frac{1}{24} f^{n-2}] + O(h^5 f^{(4)})$
	Repeat the *Correcting* step until *the iteration converges*.
...	Repeat the *Predicting* and *Correcting Steps* to advance y from y^n to y^{n+1}.

Explicit Time Integrations

The 4th order Runge-Kutta method (an explicit scheme)

Solving $dy/dt = f$ or $\partial y/\partial t = f$ with $h = \Delta t$

$$(y^*)^{n+\frac{1}{2}} = y^n + \frac{h}{2} f(t^n, y^n)$$

$$(y^{**})^{n+\frac{1}{2}} = y^n + \frac{h}{2} f(t^{n+\frac{1}{2}}, (y^*)^{n+\frac{1}{2}})$$

$$(y^{***})^{n+1} = y^n + h f(t^{n+\frac{1}{2}}, (y^{**})^{n+\frac{1}{2}})$$

$$y^{n+1} = y^n + h[\frac{1}{6} f(t^n, y^n) + \frac{2}{6} f(t^{n+\frac{1}{2}}, (y^*)^{n+\frac{1}{2}}) + \frac{2}{6} f(t^{n+\frac{1}{2}}, (y^{**})^{n+\frac{1}{2}}) + \frac{1}{6} f(t^{n+1}, (y^{***})^{n+1})] + O(h^5 f^{(4)})$$

Basic Equations

of Particle Code Simulation

Derivation of the basic equations (PDF for download)

http://www.ss.ncu.edu.tw/~lyu/2012_05_Particle/0_PDFsForDownload/a_BasicEquations_200.pdf

Basic Equations of Particle Code Simulation With Relativistic Plasma

$$\frac{d\mathbf{x}_{\alpha,k}(t)}{dt} = \mathbf{u}_{\alpha,k}(t) \Big/ \sqrt{1+\frac{u_{\alpha,k}^2}{c^2}}$$

$$\frac{d\mathbf{u}_{\alpha,k}(t)}{dt} = \frac{e_\alpha}{m_\alpha}[\mathbf{E}^e(\mathbf{x},t) + \frac{\mathbf{u}_{\alpha,k}(t)}{\sqrt{1+\frac{u_{\alpha,k}^2}{c^2}}} \times \mathbf{B}^e(\mathbf{x},t)]\delta[\mathbf{x}-\mathbf{x}_k(t)]$$

$$+\frac{e_\alpha}{m_\alpha}\{\int [\mathbf{E}^i(\mathbf{x}',t) + \frac{\mathbf{u}_{\alpha,k}(t)}{\sqrt{1+\frac{u_{\alpha,k}^2}{c^2}}} \times \mathbf{B}^i(\mathbf{x}',t)]S(\mathbf{x}-\mathbf{x}')d\mathbf{x}'\}\delta[\mathbf{x}-\mathbf{x}_k(t)]$$

where the external fields $\mathbf{E}^e(\mathbf{x},t)$, $\mathbf{B}^e(\mathbf{x},t)$; and the internal fields $\mathbf{E}^i(\mathbf{x},t)$, $\mathbf{B}^i(\mathbf{x},t)$ satisfy the following Maxwell's equations:

$$\frac{\partial \mathbf{B}^e(\mathbf{x},t)}{\partial t} = -\nabla \times \mathbf{E}^e(\mathbf{x},t)$$

$$\frac{\partial \mathbf{B}^i(\mathbf{x},t)}{\partial t} = -\nabla \times \mathbf{E}^i(\mathbf{x},t)$$

$$\frac{\partial \mathbf{E}^e(\mathbf{x},t)}{\partial t} = c^2 \nabla \times \mathbf{B}^e(\mathbf{x},t)$$

$$\frac{\partial \mathbf{E}^i(\mathbf{x},t)}{\partial t} = c^2 \nabla \times \mathbf{B}^i(\mathbf{x},t)$$

$$-\frac{1}{\varepsilon_0}\sum_{\alpha} e_\alpha \{\iiint [\int \sum_{k=1}^{N_0} \delta[\mathbf{x}' - \mathbf{x}_{\alpha,k}(t)]\delta[\mathbf{u} - \mathbf{u}_{\alpha,k}(t)]\, S(\mathbf{x}-\mathbf{x}')d\mathbf{x}'] \frac{\mathbf{u}}{\sqrt{1+\frac{u^2}{c^2}}} d^3u\}$$

and the following initial conditions:

$\nabla \cdot \mathbf{B}^e(\mathbf{x}, t=0) = 0$, $\nabla \cdot \mathbf{B}^i(\mathbf{x}, t=0) = 0$, $\nabla \cdot \mathbf{E}^e(\mathbf{x}, t=0) = 0$, and

$\nabla \cdot \mathbf{E}^i(\mathbf{x}, t=0)$

$$= \frac{1}{\varepsilon_0} \sum_\alpha e_\alpha \{ \iiint [\int \sum_{k=1}^{N_0} \delta[\mathbf{x}' - \mathbf{x}_{\alpha,k}(t=0)] \delta[\mathbf{u} - \mathbf{u}_{\alpha,k}(t=0)] \, S(\mathbf{x} - \mathbf{x}') d\mathbf{x}'] d^3u \}$$

where $S(\mathbf{x})$ is a finite-size shape function which satisfies $\int S(\mathbf{x}) d\mathbf{x} = 1$ and $S(\mathbf{x}) \geq 0$ for all $\mathbf{x}$.

The $\delta[\mathbf{x} - \mathbf{x}_{\alpha,k}(t)]$ is replaced by a Taylor expansion of the delta function with respect to the nearest grid point of $\mathbf{x}_{\alpha,k}(t)$ in the UCLA simulation scheme, but the nearest half grid point of $\mathbf{x}_{\alpha,k}(t)$ in the PIC simulation scheme and in the higher-order interpolation scheme proposed in this study.

The Essential Elements
to be Considered in a Particle Simulation Code

- Schemes for Time Integration
  - Explicit Scheme (e.g., Leap-Frog Scheme)
  - Implicit Scheme (e.g., Predictor-Corrector Method)
- Scheme for Spatial Derivatives
  - Fast Fourier Transform
  - Finite Difference
  - Fast Cubic Spline (e.g., Tsai et al., 2010)
- Scheme for "Distribution and Interpolation"
  - Taylor Expansion of the Delta Function (uniform grids)
  - Polynomial Interpolation (Distribution = inverse of Interpolation)
- Shape Function of the Finite-Size Particle
  - To reduce the artificial collisions due to insufficient number of particles used in the simulations
  - The force between two particles is greatly reduced when the spatial extents of the finite-size particles overlap each other

Review of Classical Particle Simulation Models

The UCLA Particle Simulation Model

- Schemes for Time Integration
  - Second-order Explicit Scheme : Leap-Frog Scheme
  - Implicit Scheme : Solving particle's momentum equation implicitly
- Scheme for Spatial Derivatives and Integrations
  - Fast Fourier Transform:
    - High precisions and fast in calculation
    - Good for periodic boundary conditions
- Scheme for "Distribution and Interpolation"
  - First-order Taylor expansion with respect to the nearest grid point (the so-called dipole approximation)
  - Choice of higher-order Taylor expansions (so-called quadrupole and octupole approximations)
- Shape Function of the Finite-Size Particle
  - Gaussian-shaped finite-size particle with a given particle size (i.e., the standard deviation of the Gaussian distribution)

The PIC (Particle-in-Cell) Simulation Model

- Schemes for Time Integration
  - Second-order Explicit Scheme : Leap-Frog Scheme
  - Implicit Scheme : Solving particle's momentum equation implicitly
- Schemes for Spatial Derivatives
  - Second-order Finite Difference
- Scheme for "Distribution and Interpolation"
  - First-order polynomial expansion of the delta function (or Taylor expansion with respect to the nearest half grid)
- Shape Function of the Finite-Size Particle
  - A uniform distribution function with particle shape similar to the shape of the grid distribution
  - The particle size is usually the same as the grid size. So the shape function is invisible in the PIC code. But indeed the particle size can be greater than the grid size if a uniform shape function with size greater than grid size is included in the simulation.

The New Simulation Model

- Schemes for Time Integration
  - The 4th order Implicit Predictor-Corrector Method
- Scheme for Spatial Derivatives
  - Fast Fourier Transform (for periodic boundary condition)
  - The 5th-order Finite Difference Scheme (for small-grid-size simulation, or for simulation to be parallelized with MPI)
  - Fast Cubic Spline (for large grid size)
- Scheme for "Distribution and Interpolation"
  - The 5th-order Taylor Expansion of the Delta Function with respect to the nearest half grid
  - The 5th-order Polynomial Expansion
- Shape Function of the Finite-Size Particle
  - A distribution function between the uniform distribution function and the Gauss-shaped distribution function.

Finite Difference Schemes for Spatial Derivatives

Derivation of these equations (PDF for download)
http://www.ss.ncu.edu.tw/~lyu/2012_05_Particle/0_PDFsForDownload/2_SpatialDerivatives_a_200.pdf

- The 1st order finite difference scheme

$$f_0^{(1)} = \frac{1}{2h}(f_1 - f_{-1}) + O(h^2 f_0^{(3)})$$

- The 3rd order finite difference scheme

$$f_0^{(1)} = \frac{1}{h}(-\frac{1}{12} f_2 + \frac{2}{3} f_1 - \frac{2}{3} f_{-1} + \frac{1}{12} f_{-2}) + O(h^4 f_0^{(5)})$$

- The 5th order finite difference scheme

$$f_0^{(1)} = \frac{1}{h}(+\frac{1}{60} f_3 - \frac{3}{20} f_2 + \frac{3}{4} f_1 - \frac{3}{4} f_{-1} + \frac{3}{20} f_{-2} - \frac{1}{60} f_{-3}) + O(h^6 f_0^{(7)})$$

Benchmarks

Given

a periodic sinusoidal wave $y(x) = \sin(2\pi x / \lambda)$

or

a non-periodic step function $y(x) = \tanh(3x / w)$

Find the errors in evaluating y' ,

where

$$\text{Error} = \left| \frac{y'_{FiniteDiff} - y'_{analytic}}{\left| y'_{analytic} \right|_{\max}} \right|_{\max}$$

Table 1. Errors in Evaluating y' with $y(x) = \sin(2\pi x / \lambda)$

Grid Size Δ	0.2 λ	0.1 λ	0.01 λ	0.001 λ
5th-order FD	0.021	0.00041	4.4E-10	1.2E-12
3rd-order FD	0.069	0.0050	5.2E-07	5.3E-11
Cubic Spline	0.017	0.00091	8.7E-08	9.6E-12
1st-order FD	0.24	0.065	0.00066	6.6E-06

Table 2. Errors in Evaluating y' with $y(x) = \tanh(3x / w)$

Grid Size Δ	0.2 w	0.1 w	0.01 w	0.001 w
5th-order FD	0.022	0.00090	1.4E-09	6.1E-13
3rd-order FD	0.038	0.0036	4.3E-07	4.3E-11
Cubic Spline	0.017	0.00086	7.2E-08	7.2E-12
1st-order FD	0.10	0.029	0.00030	3.0E-06

Summary

of the numerical differential schemes

- For finite difference scheme Error($\Delta = 0.1\,\lambda$)/Error($\Delta = 0.01\,\lambda$) $\approx 10^{m+1}$ where *m* is the order of the finite difference scheme.
- The 5th-order finite difference scheme with $\Delta = 0.01\,\lambda$ or $\Delta = 0.01$w is the best choice.
- The cubic spline provides better results at large grid size ($\Delta > 0.1\,\lambda$).

Numerical Schemes for "Distribution and Interpolation"

Derivation of these equations (PDF for download)
http://www.ss.ncu.edu.tw/~lyu/2012_05_Particle/0_PDFsForDownload/3.Deposition_interpolation_200.pdf

一度空間之
一階與三階內差／分散法

- 一階：
- 三階：

$$f(x) = \left(\frac{x_1 - x}{x_1 - x_0}\right) f_0 + \left(\frac{x - x_0}{x_1 - x_0}\right) f_1 + O(h^2 f^{(2)}) = b\, f_0 + a\, f_1 + O(h^2 f^{(2)})$$

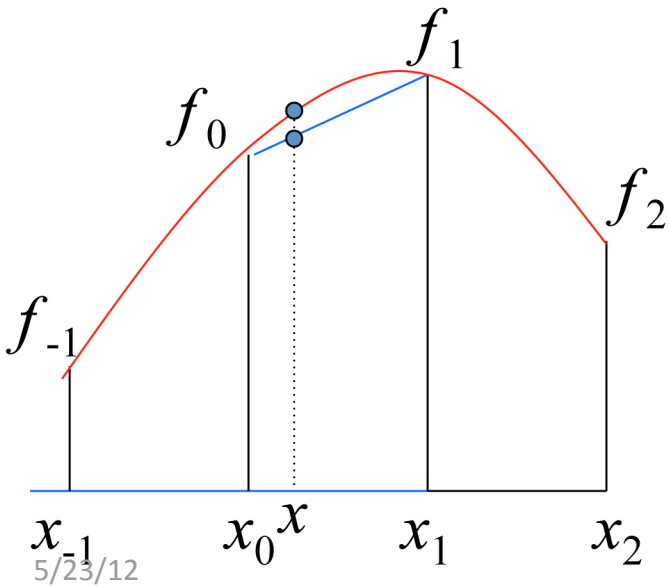
$$f(x) = a_{-1} f_{-1} + a_0 f_0 + a_1 f_1 + a_2 f_2 + O(h^4 f^{(4)})$$

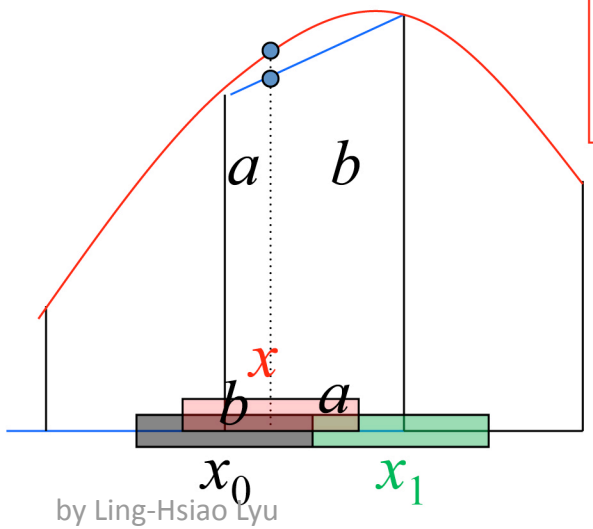
$$a_{-1} = \frac{(x - x_0)(x - x_1)(x - x_2)}{(x_{-1} - x_0)(x_{-1} - x_1)(x_{-1} - x_2)}$$

$$a_0 = \frac{(x - x_{-1})(x - x_1)(x - x_2)}{(x_0 - x_{-1})(x_0 - x_1)(x_0 - x_2)}$$

$$a_1 = \frac{(x - x_{-1})(x - x_0)(x - x_2)}{(x_1 - x_{-1})(x_1 - x_0)(x_1 - x_2)}$$

$$a_2 = \frac{(x - x_{-1})(x - x_0)(x - x_1)}{(x_2 - x_{-1})(x_2 - x_0)(x_2 - x_1)}$$





Different Point of Views:
Deposition
vs.
Interpolation / Distribution
?

二度空間之
一階內差／分散法

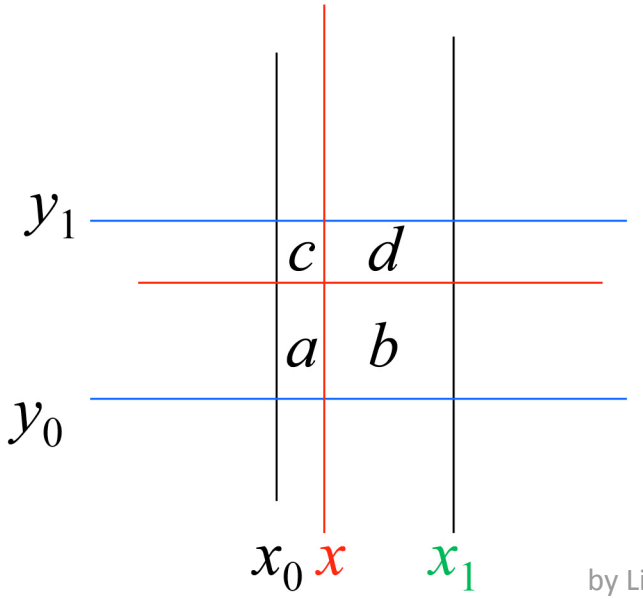
- 一階：

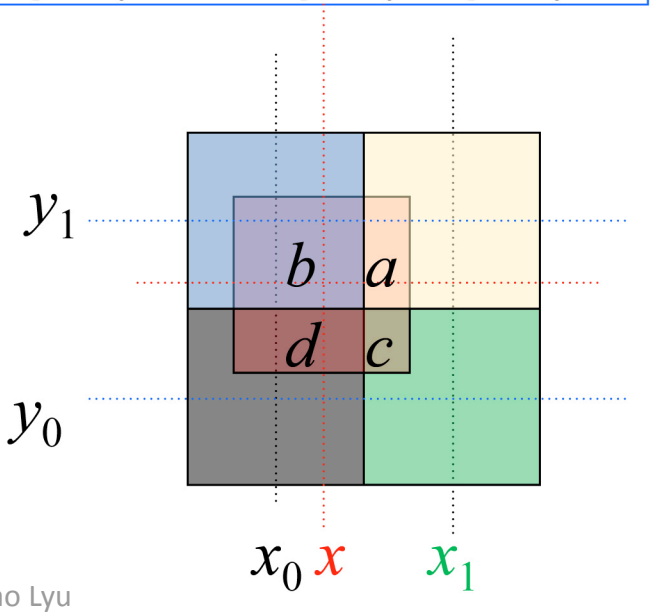
以下Particle-in-Cell
的看法，看似好懂，
其實不易推廣到高階

$$f(x) = d\,f_{00} + c\,f_{10} + b\,f_{01} + a\,f_{11}$$

$$= \left(\frac{x_1 - x}{x_1 - x_0}\right)\left(\frac{y_1 - y}{y_1 - y_0}\right) f_{00} + \left(\frac{x - x_0}{x_1 - x_0}\right)\left(\frac{y_1 - y}{y_1 - y_0}\right) f_{10}$$

$$+ \left(\frac{x_1 - x}{x_1 - x_0}\right)\left(\frac{y - y_0}{y_1 - y_0}\right) f_{01} + \left(\frac{x - x_0}{x_1 - x_0}\right)\left(\frac{y - y_0}{y_1 - y_0}\right) f_{11}$$





二度空間之
一階與三階內差／分散法

- 一階：

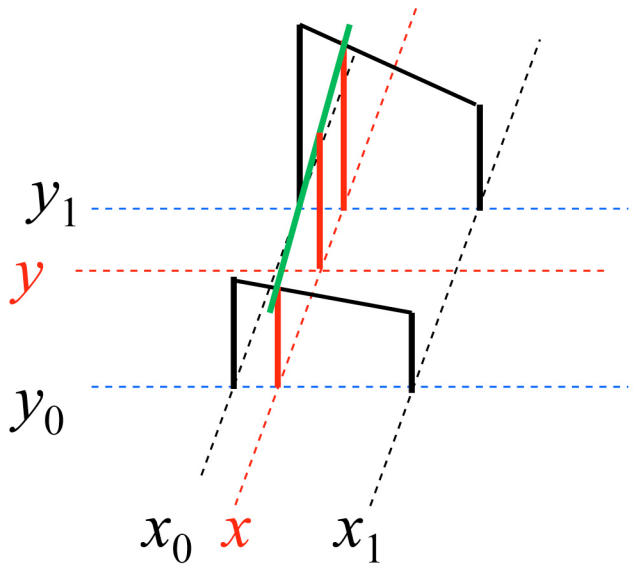
$$f(x)=\left(\frac{x_1-x}{x_1-x_0}\right)\left(\frac{y_1-y}{y_1-y_0}\right)f_{00}+\left(\frac{x-x_0}{x_1-x_0}\right)\left(\frac{y_1-y}{y_1-y_0}\right)f_{10}$$
$$+\left(\frac{x_1-x}{x_1-x_0}\right)\left(\frac{y-y_0}{y_1-y_0}\right)f_{01}+\left(\frac{x-x_0}{x_1-x_0}\right)\left(\frac{y-y_0}{y_1-y_0}\right)f_{11}+O(h^2f^{(2)})$$

同理可得三階的內差／分散公式

- 三階：

$$f(x)=\sum_{i=-1}^{2}\sum_{j=-1}^{2}a_{ij}f_{ij}$$

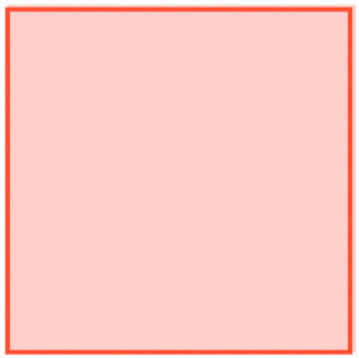
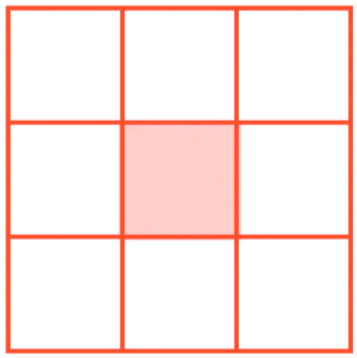
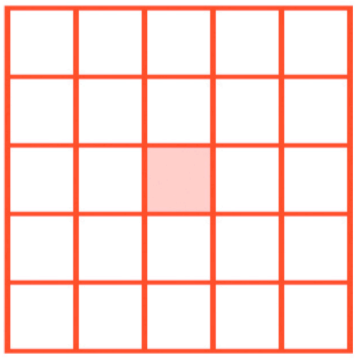
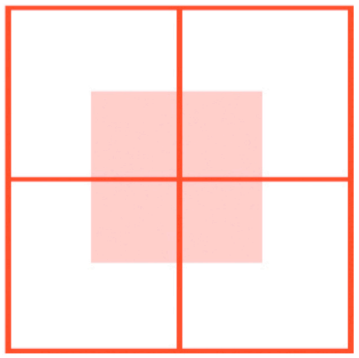
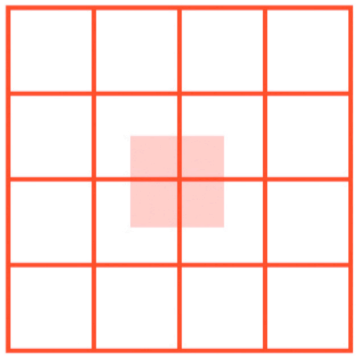
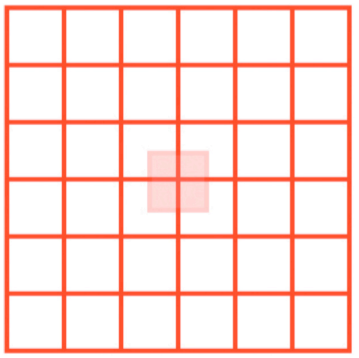
另外，三度空間的差分／分散法，
也可以如法炮製，求得。



Summary of the Higher-Order Interpolations or Distributions

Cubic interpolations For $x_{-1} < x_0 < x < x_1 < x_2$	*Fifth-order interpolations* For $x_{-2} < x_{-1} < x_0 < x < x_1 < x_2 < x_3$
$f(x) = \sum_{i=-1}^{2} a_i f(x_i)$ $f(x,y) = \sum_{j=-1}^{2} b_j (\sum_{i=-1}^{2} a_i f(x_i, y_j))$ $f(x,y,z)$ $= \sum_{k=-1}^{2} c_k [\sum_{j=-1}^{2} b_j (\sum_{i=-1}^{2} a_i f(x_i, y_j, z_k))]$	$f(x) = \sum_{i=-2}^{3} a_i f(x_i)$ $f(x,y) = \sum_{j=-2}^{3} b_j (\sum_{i=-2}^{3} a_i f(x_i, y_j))$ $f(x,y,z) = \sum_{k=-2}^{3} c_k [\sum_{j=-2}^{3} b_j (\sum_{i=-2}^{3} a_i f(x_i, y_j, z_k))]$
where	where
$a_{-1} = \frac{(x-x_0)(x-x_1)(x-x_2)}{(x_{-1}-x_0)(x_{-1}-x_1)(x_{-1}-x_2)}$ $a_0 = \frac{(x-x_{-1})(x-x_1)(x-x_2)}{(x_0-x_{-1})(x_0-x_1)(x_0-x_2)}$ $a_1 = \frac{(x-x_{-1})(x-x_0)(x-x_2)}{(x_1-x_{-1})(x_1-x_0)(x_1-x_2)}$ $a_2 = \frac{(x-x_{-1})(x-x_0)(x-x_1)}{(x_2-x_{-1})(x_2-x_0)(x_2-x_1)}$	$a_{-2} = \frac{(x-x_{-1})(x-x_0)(x-x_1)(x-x_2)(x-x_3)}{(x_{-2}-x_{-1})(x_{-2}-x_0)(x_{-2}-x_1)(x_{-2}-x_2)(x_{-2}-x_3)}$ $a_{-1} = \frac{(x-x_{-2})(x-x_0)(x-x_1)(x-x_2)(x-x_3)}{(x_{-1}-x_{-2})(x_{-1}-x_0)(x_{-1}-x_1)(x_{-1}-x_2)(x_{-1}-x_3)}$ $a_0 = \frac{(x-x_{-2})(x-x_{-1})(x-x_1)(x-x_2)(x-x_3)}{(x_0-x_{-2})(x_0-x_{-1})(x_0-x_1)(x_0-x_2)(x_0-x_3)}$ $a_1 = \frac{(x-x_{-2})(x-x_{-1})(x-x_0)(x-x_2)(x-x_3)}{(x_1-x_{-2})(x_1-x_{-1})(x_1-x_0)(x_1-x_2)(x_1-x_3)}$ $a_2 = \frac{(x-x_{-2})(x-x_{-1})(x-x_0)(x-x_1)(x-x_3)}{(x_2-x_{-2})(x_2-x_{-1})(x_2-x_0)(x_2-x_1)(x_2-x_3)}$ $a_3 = \frac{(x-x_{-2})(x-x_{-1})(x-x_0)(x-x_1)(x-x_2)}{(x_3-x_{-2})(x_3-x_{-1})(x_3-x_0)(x_3-x_1)(x_3-x_2)}$
so are the b_j and c_k.	so are the b_j and c_k.

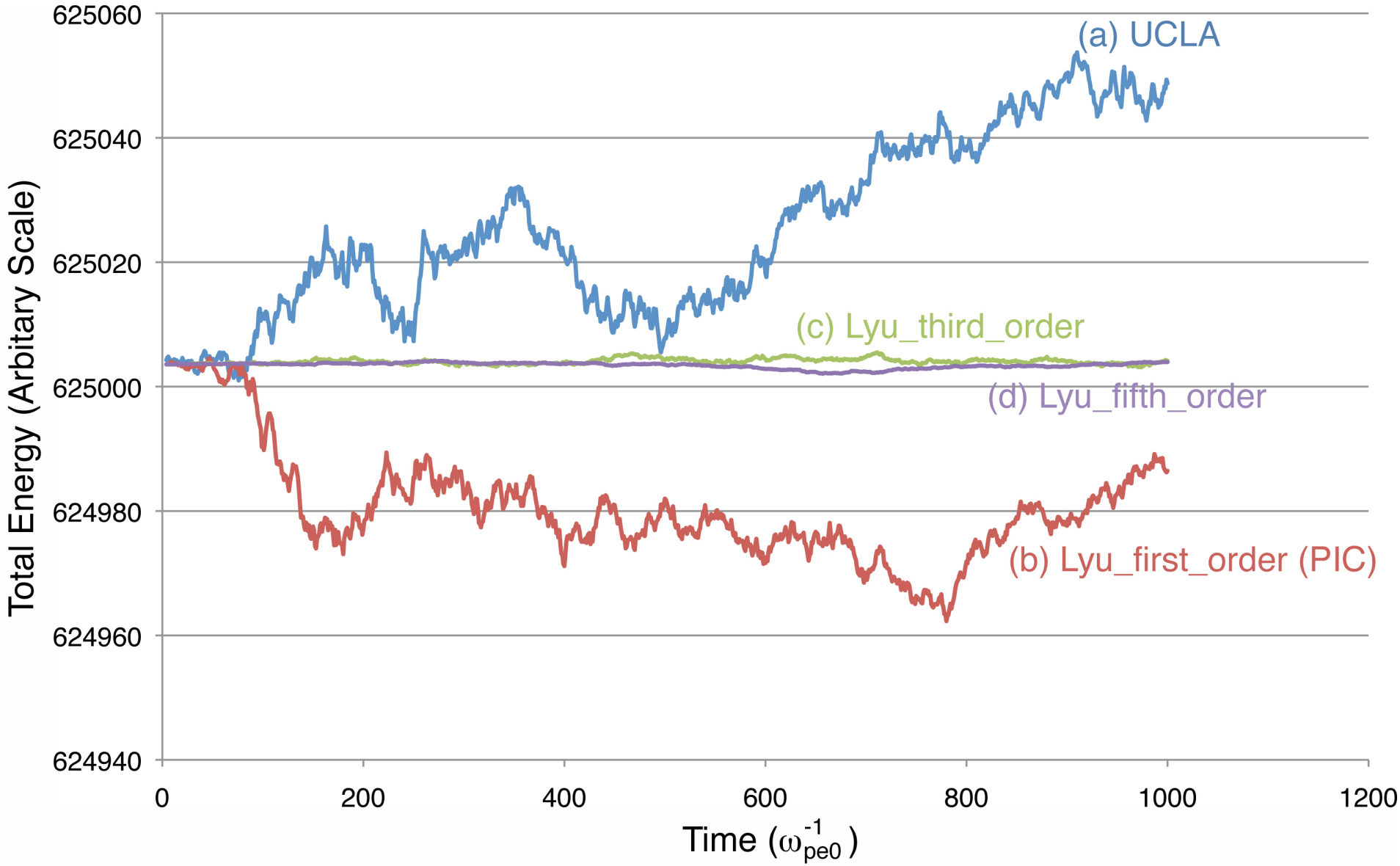
Grids in the expansion of $\delta[\mathbf{x}-\mathbf{x}_{\alpha,k}(t)]$ with $\mathbf{x}_{\alpha,k}$ located in the shaded region

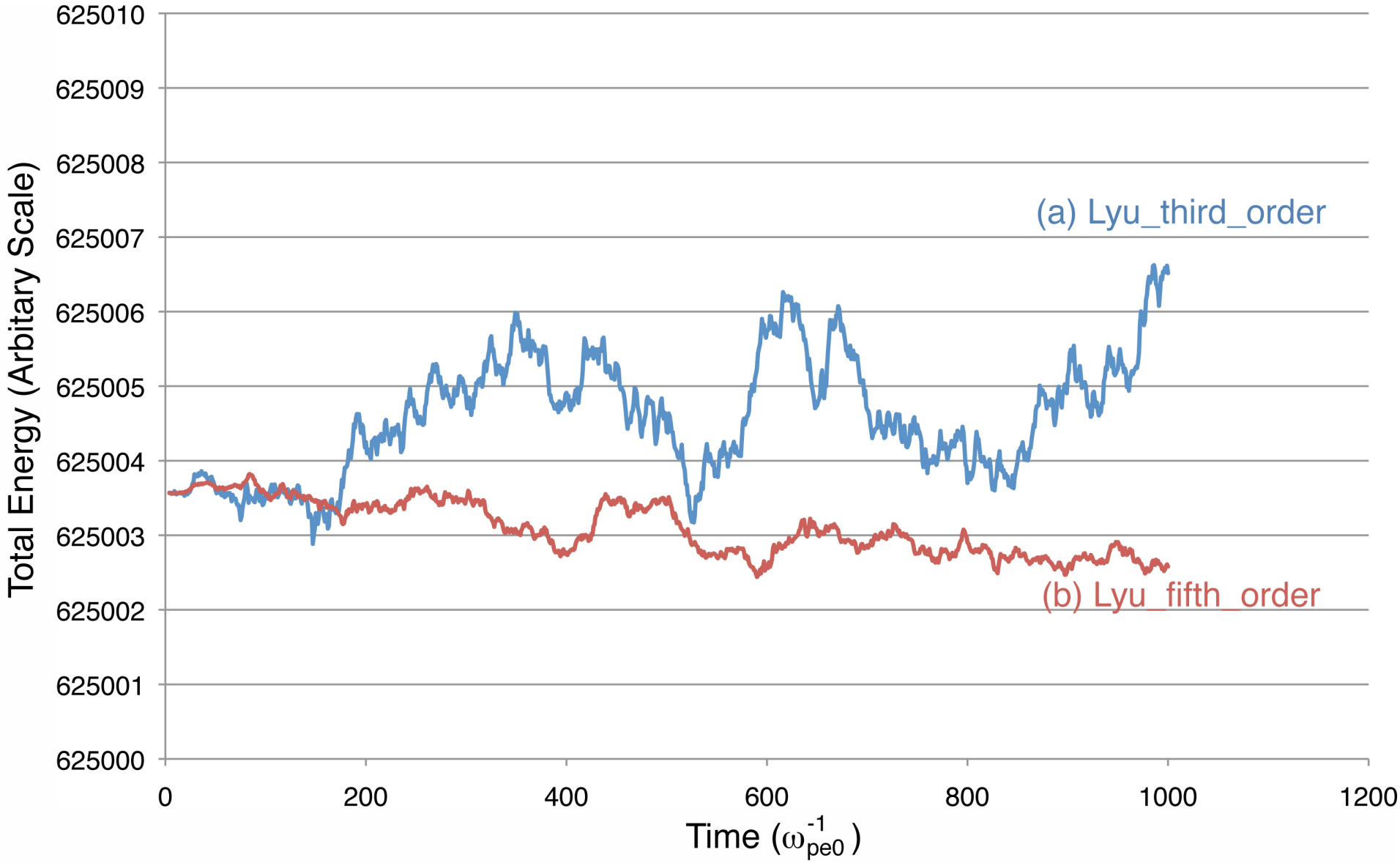
PIC Code & This Study	1st-order 	3rd-order 	5th-order 
UCLA Finite-Size Particle Code	1st- and 2nd-order 	3rd- and 4th-order 	5th- and 6th-order 

Benchmarks

Summary of the Simulation Results With Different Order of Distribution and Interpolation Schemes

Conclusion:
The numerical errors have been greatly reduced in the simulations with higher order deposition-interpolation schemes.


625060
625040
625020
625000
624980
624960
624940
Total Energy (Arbitary Scale)
0
200
400
600
800
1000
1200
Time (ω_{pe0}^{-1})
(a) UCLA
(c) Lyu_third_order
(d) Lyu_fifth_order
(b) Lyu_first_order (PIC)

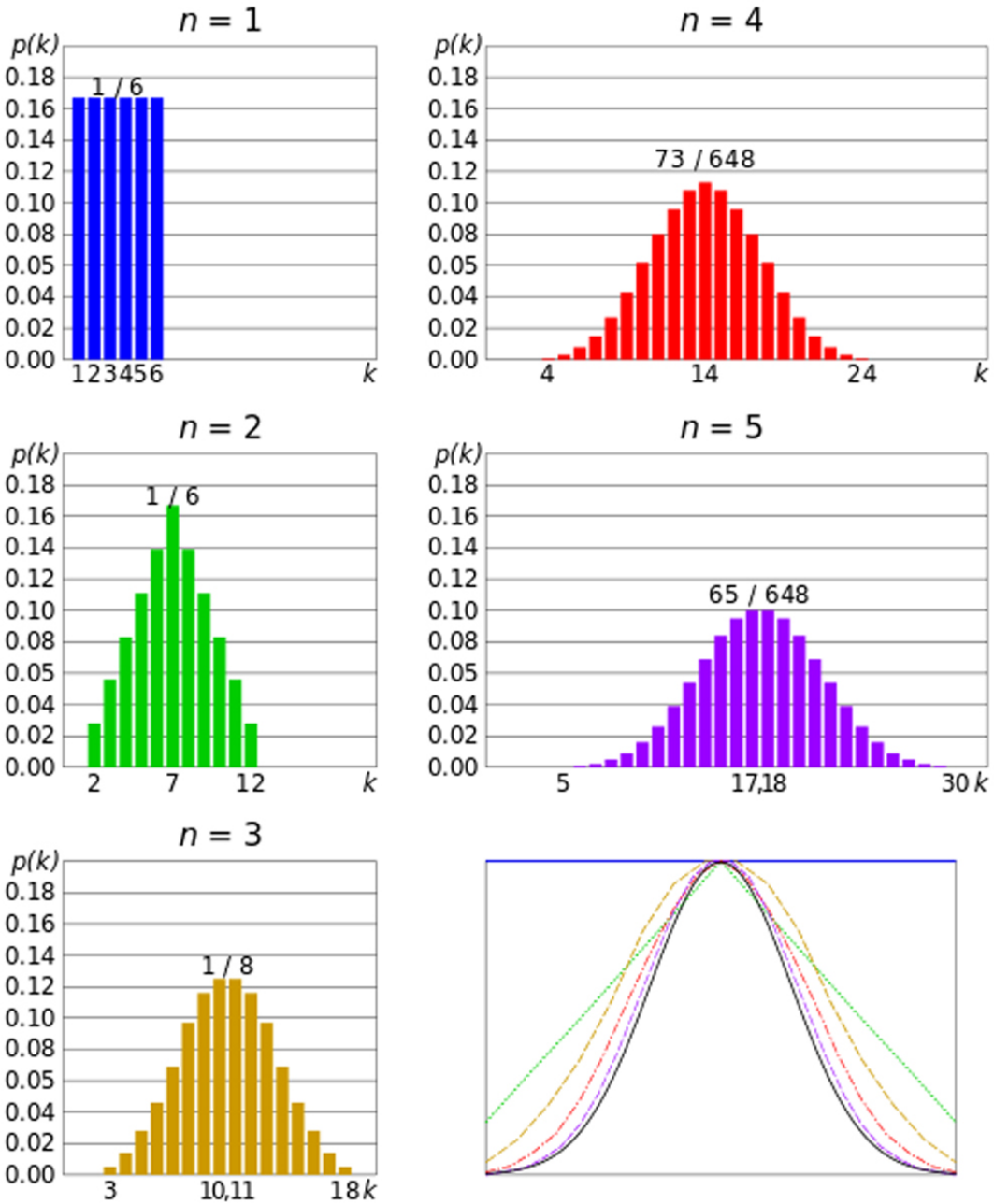

625010
625009
625008
625007
625006
625005
625004
625003
625002
625001
625000
Total Energy (Arbitary Scale)
0
200
400
600
800
1000
1200
Time (ω_{pe0}^{-1})
(a) Lyu_third_order
(b) Lyu_fifth_order

Shape Function of the Finite-Size Particle

Q: How to construct a distribution function with characteristics between the uniform distribution function and the Gaussian-shaped distribution function?

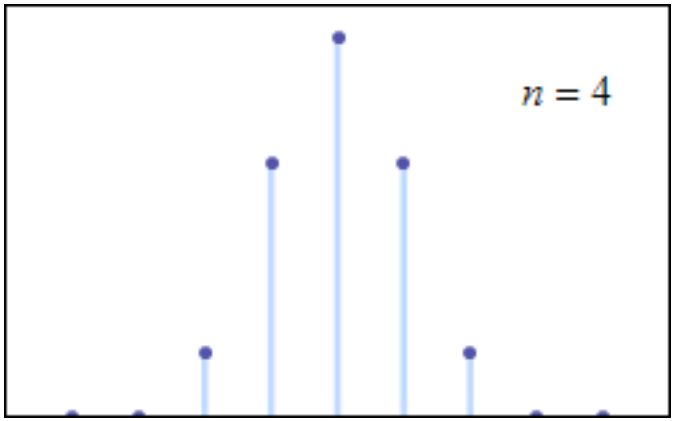
A: Let us consider the probability density functions for the sum of *n* Bernoulli trial (yes/no trail).

- For $n = 1$ the shape of the binomial distribution is a uniform distribution.
- For large *n*, the shape of the binomial distribution will resemble the Gaussian distribution.



These plots used to be a homework of my thermodynamics course. Now, students can find answer from the Wikipedia.
I don't know if it is good or bad to them!

de Moivre-Laplace Theorem
-- A special case of the central-limit theorem



As *n* increases, the shape of the binomial distribution begins to resemble the smooth Gaussian curve.

Movie download
http://www.ss.ncu.edu.tw/~lyu/2012_05_Particle/0_PDFsForDownload/deML.gif

Comparison of probability density functions, *p(k)* for the sum of *n fair 6-sided dice to show their* convergence to a normal distribution with increasing *n, in accordance to the central limit* theorem.

Shape Function of the Finite-Size Particle

Q: Binomial distribution is a discrete probability distribution. How to construct a continuous shape function from the binomial distribution?

A: Step 1:

Determine the cumulative distribution from the discrete binomial distribution.

Step 2:

Use two polynomials to fit the cumulative distribution.

Step 3:

Find the continuous shape function from the derivative of these polynomials.

Step 4:

For a given particle size, adjust the polynomials, so that the standard deviation of the shape function is equal to the particle size.

Shape Function of the Finite-Size Particle

Derivation of the Shape Functions -- Draft (PDF for download)
http://www.ss.ncu.edu.tw/~lyu/2012_05_Particle/0_PDFsForDownload/4.ShapeFunction_Draft_200.pdf

Final Remarks
- important messages to the students

- The format of the finite difference scheme of a given order is not unique.
- You have to know how to derive the given formula or equation before you use it.
  - So, you know the limitations of the given formula or equation.
  - If there is a typo, you will correct it before you put it into your program.
- If you want to invent a new formula or a new simulation scheme, you have to verify it before you use it.
- "To understand is to know how to calculate." --Dirac
- "To understand is to know how to derive."

Thanks for Your Attention!

Have Fun with the Particle Simulation!