Development of a plasma turbulence code by C++

N. Miyato
Japan Atomic Energy Agency

19th NEXT Workshop
Katsura campus, Kyoto Univ., Kyoto, Japan
29-30 August 2013

Introduction

- Plasma turbulence in the tokamak edge and peripheral region plays an important role determining heat/particle fluxes towards the divertor plate.
- Reduction of the heat load to the divertor plate is a critical problem for realising the fusion reactor.
- To evaluate the heat load accurately, various physical processes should be considered.
- Use of C++ makes it easier to add physical elements step by step into the code.

Data2d (2D data class)

This class has data for 2D space perpendicular to magnetic field and member functions associated with them.

Data members：

Data in 2D space（pointer to array of pointers）
Coordinate values for X（pointer）
Coordinate values for Y（pointer）
Mesh numbers for X and Y(int)
Components of metric tensor(double)
・・・・・・

Member functions：

Overloading arithmetic operators（＋,－,×）
Overloading ＝ operator
Copy constructer
Computing Poisson brackets with Arakawa scheme
Computing value at any y position from discrete point values
・・・・・・

Implementation of 2D array with new

How to make 2D array with new:

```
double **a;
a = new double* [ nx ];
a[0] = new double [ nx*ny ];
for ( int i=0 ; i<nx ; i++ ) a[i] = a[0] + i*ny;
```

We can refer a value at `(i,j)` by `a[i][j]`.

Two steps are necessary to delete a.

```
delete[] a[0];
delete[] a;
```

Initialisation at constructer may be

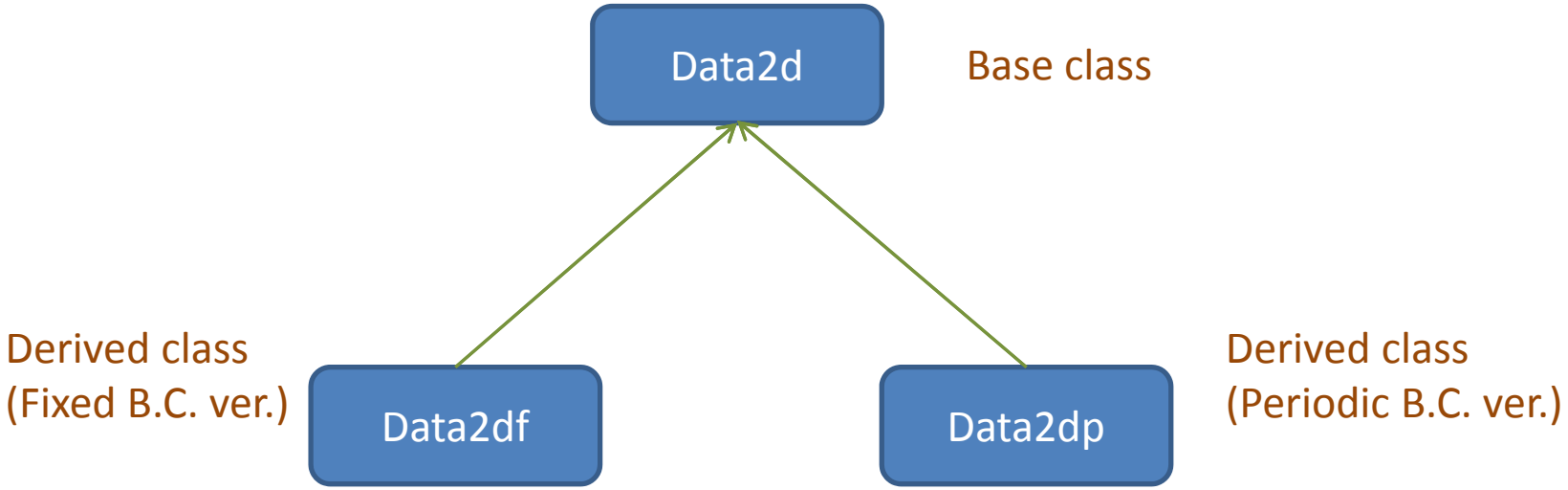
```
a = new double* [1];
a[0] = 0;                // NULL pointer
```

Derived classes of Data2d

We would like to change actions of some functions according to situations (for example, fixed or periodic boundary condition), but to keep the other functions unchanged.

Common parts are defined in base class, and functions which act differently are defined in derived classes.

Poisson solver for periodic B. C. is different from that for fixed B. C. Therefore, it is defined in derived classes with same name.



Tips for derived class

Reuse of operator= overloaded in base class

Overloading of arithmetic operators（＋,－,×） are derived from base class, and we can use them for derived classes.
But overloading of operator= is not derived, so we have to redefine it in derived classes. If we want to use operator= overloaded in base class in derived classes, we can use it as follows:

```
derived& operator=(const base &ob2) {
  base::operator=(ob2);
  return *this;
};
```

Constructer of derived class

Even if we want no action for constructer of derived class, we must define default constructer which has no action as

```
derived() {};
```

Note: constructer of base class is called automatically even in above case.

When we want to pass parameters to base class

```
derived(int a, int b) : base(a,b) {};
```

Arakawa scheme

[Arakawa JCP 1966]

Arakawa spatial discretisation preserves conservation properties of Poisson brackets and is used in some plasma turbulence codes [Naulin *et al.* PoP 2003, Scott PoP 2005].

Three representations of Poisson brackets [f,g]

$$J^{[1]} = \frac{\partial f}{\partial x}\frac{\partial g}{\partial y} - \frac{\partial f}{\partial y}\frac{\partial g}{\partial x}$$

$$J^{[2]} = \frac{\partial}{\partial x}\left(f\frac{\partial g}{\partial y}\right) - \frac{\partial}{\partial y}\left(f\frac{\partial g}{\partial x}\right)$$

$$J^{[3]} = \frac{\partial}{\partial y}\left(g\frac{\partial f}{\partial x}\right) - \frac{\partial}{\partial x}\left(g\frac{\partial f}{\partial y}\right)$$

Discretising them and taking average,

$$[f,g] = \frac{1}{3}(J^{[1]} + J^{[2]} + J^{[3]})$$

finally we have

$$\begin{aligned}
[f,g]_{i,j} = -\frac{1}{12h^2}[&(g_{i,j-1} + g_{i+1,j-1} - g_{i,j+1} - g_{i+1,j+1})(f_{i+1,j} + f_{i,j}) \\
&-(g_{i-1,j-1} + g_{i,j-1} - g_{i-1,j+1} - g_{i,j+1})(f_{i,j} + f_{i-1,j}) \\
&+(g_{i+1,j} + g_{i+1,j+1} - g_{i-1,j} - g_{i-1,j+1})(f_{i,j+1} + f_{i,j}) \\
&-(g_{i+1,j-1} + g_{i+1,j} - g_{i-1,j-1} - g_{i-1,j})(f_{i,j} + f_{i,j-1}) \\
&+(g_{i+1,j} - g_{i,j+1})(f_{i+1,j+1} + f_{i,j}) - (g_{i,j-1} - g_{i-1,j})(f_{i,j} + f_{i-1,j-1}) \\
&+(g_{i,j+1} - g_{i-1,j})(f_{i-1,j+1} + f_{i,j}) - (g_{i+1,j} - g_{i,j-1})(f_{i,j} + f_{i+1,j-1})]
\end{aligned}$$

Karniadakis scheme

[Karniadakis *et al.* JCP 1991]

Third-order "stiffly stable" algorithm derived by Karniadakis *et al.* is used for time advance scheme [Scott PoP 2005].

$$\frac{\partial F}{\partial t} = S + D(f)$$

where F is the functional of the dependent variables f, S is the right hand side, D is the artificial dissipation operator.

F at new time step is evaluated from current and past data by

$$F_1 = \frac{6}{11}\left[3F_0 - \frac{3}{2}F_{-1} + \frac{1}{3}F_{-2} + \Delta t(3S_0 - 3S_{-1} + S_{-2})\right]$$

$$F_1 \leftarrow F_1 + \Delta t D(f_0)$$

(apply B. C.)

After that f_1 is computed from F_1.

2D Data class test

Equation system

$$\frac{\partial \rho}{\partial t} + [\varphi, \rho] = 0$$ Vorticity equation

$$\nabla^2 \varphi = -\rho$$ Poisson equation

where Poisson brackets are defined by

$$[\varphi, \rho] = \frac{\partial \varphi}{\partial x}\frac{\partial \rho}{\partial y} - \frac{\partial \varphi}{\partial y}\frac{\partial \rho}{\partial x}$$

Conservation laws

Enstrophy

$$\frac{d}{dt}\int \rho^2 dxdy = 0$$

Electric energy

$$\frac{d}{dt}\int |\nabla \varphi|^2 dxdy = 0$$

It is possible to test Poisson solver, Arakawa scheme, Karniadakis scheme.

2D Kelvin-Helmholtz instability

[Shoucri-Knorr Physica Scripta 1976]

We consider the following simulation domain:

$0 < x < 2\pi$, fixed or periodic boundary condition

$0 < y < L_y$, periodic boundary condition

Equilibrium: $\rho_0(x) = \sin x, \varphi_0(x) = \sin x$

ExB flow in y direction $U_0 = \partial_x \varphi_0 = \cos x$ if $\mathbf{B} = B\hat{z}$.

With fixed B. C. for x, consider the following initial perturbation

$$\rho(x, y, t = 0) = \sin x + \epsilon \sin(x/2) \cos(ky)$$

where $k = 2\pi/L_y$ is wavenumber in y direction.

Kelvin-Helmholtz instability occurs if $k < k_s = \sqrt{3}/2$.

This problem is useful for checking numerical codes.
[Ghizzo JCP 1993, Sonnendrücker JCP 1999, Watanabe NIFS-792 2004, etc.]

When we consider the following initial perturbation with periodic B. C. for x,

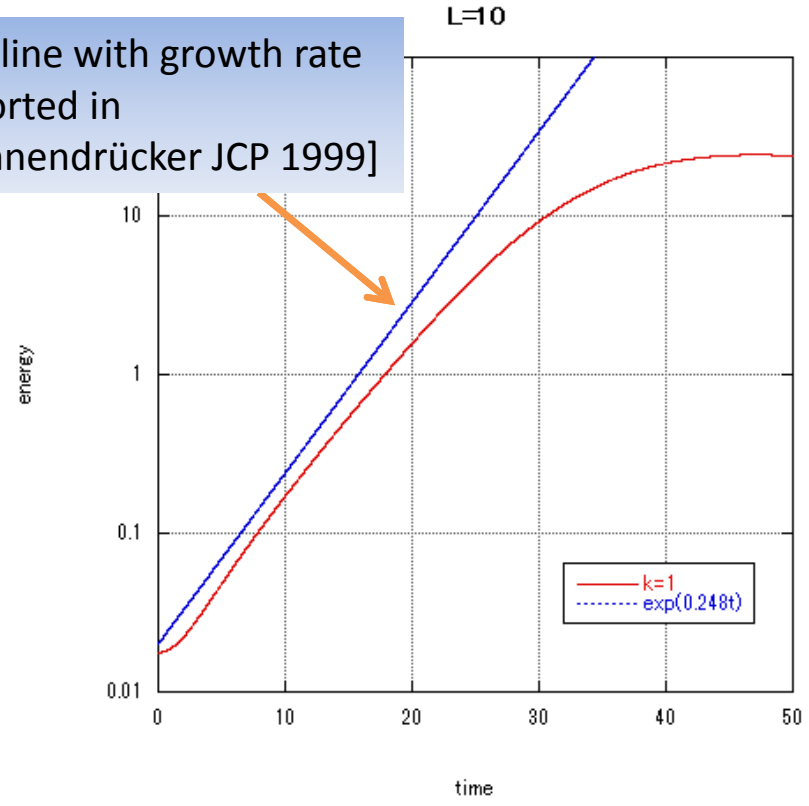
$$\rho(x, y, t = 0) = \sin x + \epsilon \cos(ky)$$

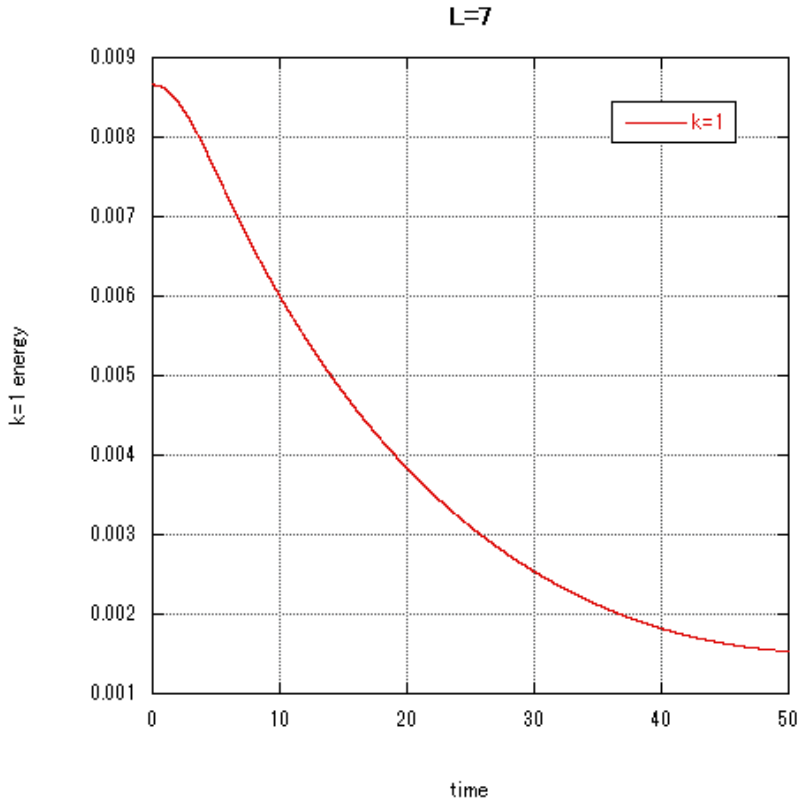
$k_s = 1$ is stability boundary.

Fixed boundary condition

$k_s = 2\pi/L_s = \sqrt{3}/2$ is stability boundary,
then $L > L_s \approx 7.255$.

The line with growth rate reported in [Sonnendrücker JCP 1999]

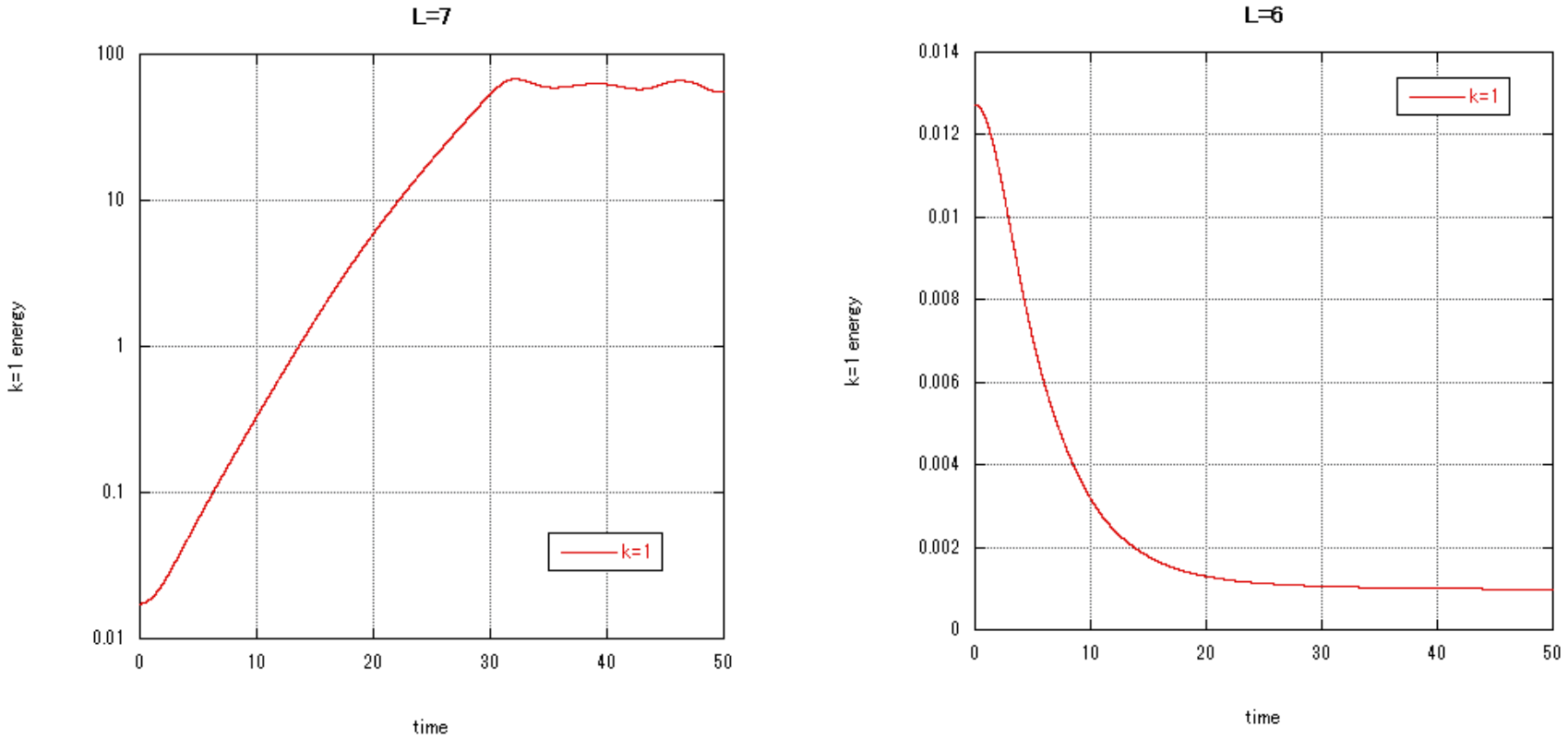




513×512, $\Delta t = 5 \times 10^{-3}$

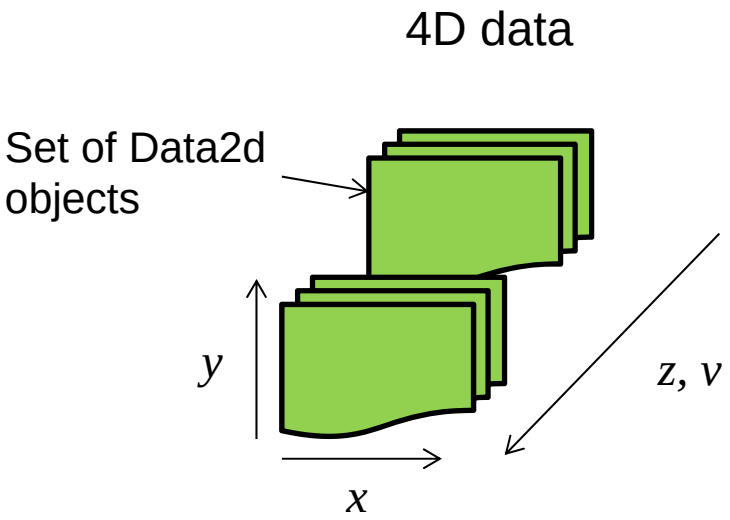
Periodic boundary condition

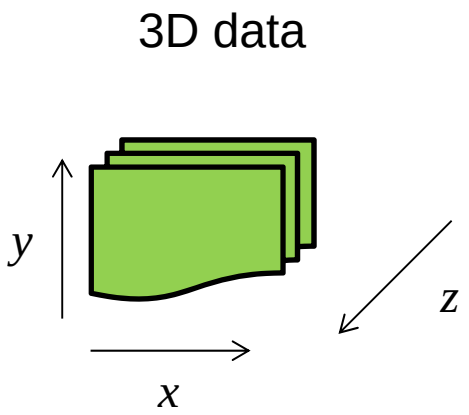
$k_s = 2\pi/L_s = 1$ is stability boundary,
then instability occurs for $L > L_s \approx 6.28$.



512×512, $\Delta t = 5 \times 10^{-3}$

Vector of Data2d





Overloading of operator- is defined in Data2d and operator[ ] is defined in std::vector. Therefore, the following representation is possible for vector（phi_）, vector of Data2d object:

phi_[i+1]-phi_[i-1]

where i denotes the index for z.