ITER IM Technology Workshop
Chateau de Cadarache
2011/06/08

Data Structures and Code Interfaces of BPSD

A. Fukuyama[1] **and A. Hatayama**[2]

[1]*Graduate School of Engineering, Kyoto University, Kyoto, Japan*

[2]*Graduate School of Science and Technology, Keio University, Tokyo, Japan*

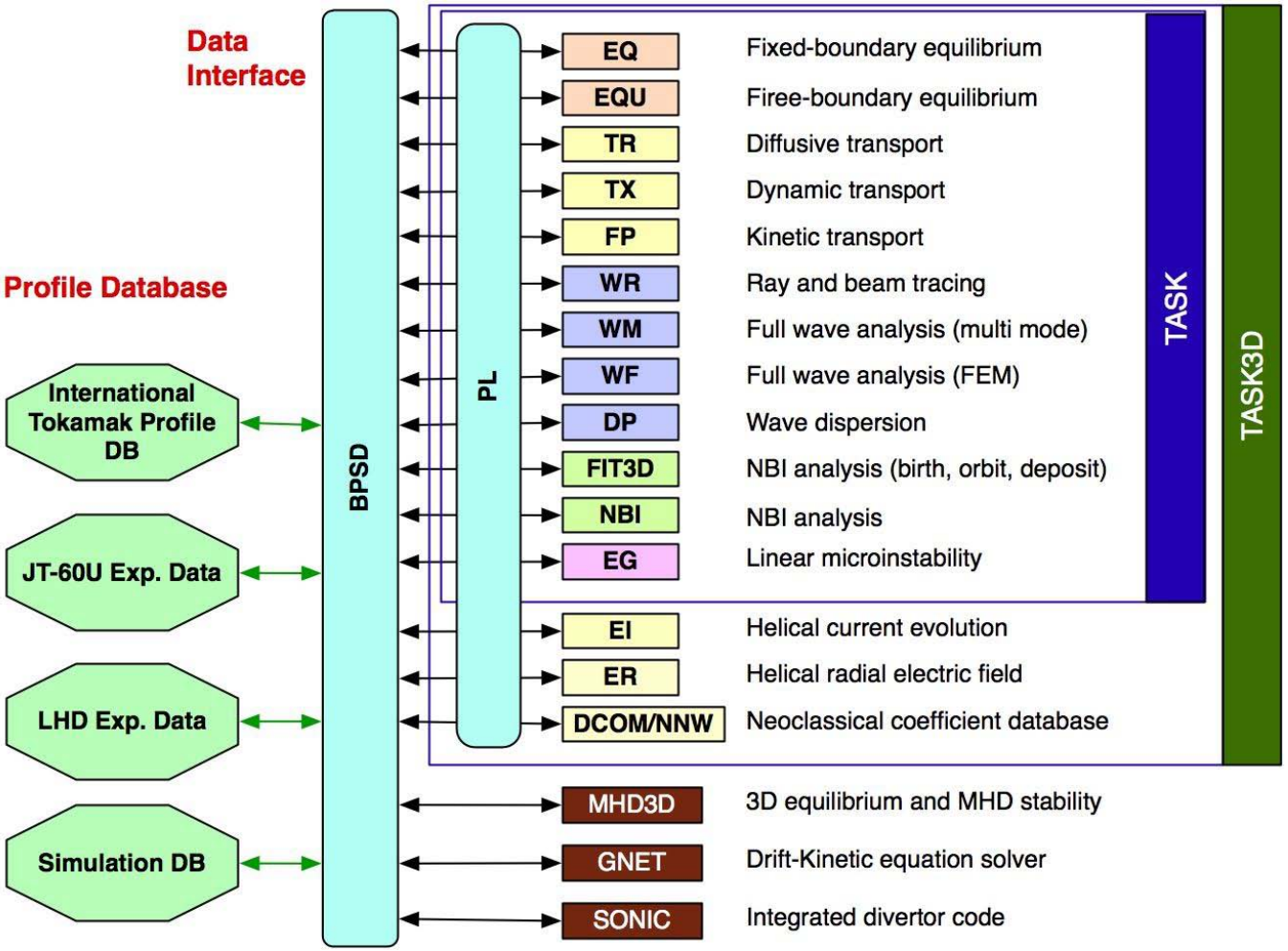
in collaboration with **BPSI working group**

BPSD: Data Exchange Interface
Extension to peripheral plasma
Summary and thoughts

This work is supported by Grant-in-Aid for Scientific Research (S) (20226017) from JSPS, Japan.

Structure of TASK, TASK3D, and associated codes

- **BPSI**: Burning Plasma Simulation Initiative (Universities, NIFS, JAEA)
  - **TASK**: Integrated code for tokamak
  - **TASK3D**: Integrated code for 3D configuration (helical, tokamak)



Component Collaboration

- **Role of Component Interface**
  - **Data exchange between components**: **BPSD**
    - **Standard dataset**: Specify set of data
    - **Specification of data exchange interface**: initialize, set, get
      Specification of file i/o interface: save, load
  - **Execution control**: **BPSX**
    - **Specification of execution control interface**:
      initialize, setup, exec, visualize, terminate
    - **Uniform user interface**: parameter input, graphic output
- **Role of data exchange interface: TASK/PL**
  - **Keep present status of plasma and device**
  - **Store history of plasma**
  - **Interface to experimental data base**

Policies of BPSD

- **Minimum and Sufficient Dataset**
  - **To minimize the data to be exchanged**
  - **Mainly profile data**
  - **Routines to calculate global integrated quantities**
    - separately provided
- **Minimum Arguments in Interfaces**
  - **To maximize flexibility**
  - **Use derived data type or struct**
  - **Only one dataset in the arguments of an interface**
- **Minimum Kinds of Interfaces**
  - **To make modular programming easier**
  - **Use function overloading**
- **Language**: F90/95

Data Exchange Interface: BPSD

- **Standard dataset**: Specify data to be stored and exchanged
  - **Data structure**: Derived type (Fortran95): structured type

e.g.

time	`plasmaf%time`
number of grid	`plasmaf%nrmax`
number of species	`plasmaf%nsamax`
normalized radius	`plasmaf%rho(nr)`
Species specifier	`plasmaf%ns(nsa)`
plasma density	`plasmaf%data(nr,nsa)%density`
plasma temperature	`plasmaf%data(nr,nsa)%temperature`

- **Specification of API**:
  - **Program interface**

e.g.

Set data	`bpsd_set_data(plasmaf,ierr)`
Get data	`bpsd_get_data(plasmaf,ierr)`
Save data to file	`bpsd_save(ierr)`
Load data from file	`bpsd_load(ierr)`

  - **BPSD data file** (bpsddata): Binary file of all existing bpsd data

BPSD: Standard dataset

- **Internal dataset**: internal use, easier data handling
  - **bpsd_shotx_type**: text data including shotID
  - **bpsd_data0Dx_type**: no profile
  - **bpsd_data1Dx_type**: 1D profile (ρ: normalized radius)
  - **bpsd_data2Dx_type**: 2D profile (ρ, χ: poloidal angle)
  - **bpsd_data3Dx_type**: 3D profile (ρ, χ, ζ: toroidal angle)
- **Standard dataset**: external use, data exchange
  - **User defined dataset**
    - **bpsd_data0D_type, bpsd_data1D_type, bpsd_data2D_type, bpsd_data3D_type**
  - **Predefined dataset**
    - Without dataName: **bpsd_device_type, bpsd_equ1D_type, etc.**
    - With dataName: **bpsd_trmatrix_type**: neoclassical, turbulence, ...
      bpsd_trsource_type: P_{EC}, P_{NB}, S_{NB}, ...

Example of internal dataset structure: data1Dx

```
type bpsd_data1Dx_type
   integer :: status = 0! 0:unalloc 1:undef 2:assigned
                        ! 3:sp-alloc 4:splined
   integer :: nrmax     ! Number of radial points
   integer :: ndmax     ! Number of data
   integer :: idum      ! Dummy
   real(rkind) :: time
   real(rkind), dimension(:), pointer :: rho
   real(rkind), dimension(:,:), pointer :: data
   real(rkind), dimension(:,:,:), pointer :: spline
   character(len=32) :: dataName
   character(len=8)  :: created_date
   character(len=10) :: created_time
   character(len=5)  :: created_timezone
   character(len=9)  :: dummy
   character(len=32), dimension(:), pointer :: kid   ! item name
   character(len=32), dimension(:), pointer :: kunit ! unit of item
end type bpsd_data1Dx_type
```

BPSD Standard Dataset

Category	Name	EQ	TR	TX	FP	WR	WM	DP
Shot data	**bpsd_shot_type**	–	–	–	–	–	–	–
Device data	**bpsd_device_type**	in	in	in	in			
1D equilibrium data	**bpsd_equ1D_type**	out	in	in	in			
2D equilibrium data	**bpsd_equ2D_type**	out			in	in	in	in
1D metric data	**bpsd_metric1D_type**	out	in	in	in			
2D metric data	**bpsd_metric2D_type**	out			in	in	in	in
Plasma species data	**bpsd_species_type**	in	in	in	in			in
Fluid plasma data	**bpsd_plasmaf_type**	in	out	out	i/o			in
Kinetic plasma data	**bpsd_plasmak_type**				out			in
Transport matrix data	**bpsd_trmatrix_type**		i/o					
Transport source data	**bpsd_trsource_type**		i/o	i/o	i/o	out	out	
Dielectric tensor data	**bpsd_dielectric_type**					in	in	out
Full wave field data	**bpsd_wavef_type**				in	out		
Ray tracing field data	**bpsd_waver_type**				in		out	
Beam tracing field data	**bpsd_waveb_type**				in		out	
User defined data	**bpsd_0/1/2ddata_type**	–	–	–	–	–	–	–

Defined Dataset (1)

bpsd_shot_type

DeviceID			text
ShotID			number
ModelID			number

bpsd_device_type

rr	R	m	Geometrical major radius
zz	Z	m	Geometrical vertical position
ra	a	m	Geometrical minor radius
rb	b	m	Wall radius
bb	B	T	Vacuum toroidal mag. field
ip	I_{p}	A	Typical plasma current
elip	κ		Elongation at boundary
trig	δ		Triangularity at boundary

bpsd_species_type n_s: Particle species ID

pa	$A(n_s)$	Mass number
pz	$Z(n_s)$	Charge number
pz0	$Z_0(n_s)$	Atomic number

Defined Dataset (2)

bpsd_equ1D_data (ρ: Normalized radius: $\sqrt{\psi_t/\psi_t(a)}$)

psit	$\psi_t(\rho)$	Tm^2	Toroidal magnetic flux
psip	$\psi_p(\rho)$	Tm^2	Poloidal magnetic flux
ppp	$p(\rho)$	MPa	Plasma pressure
piq	$q(\rho)$		Inverse of safety factor
pip	$I_t(\rho)$	Tm	Poloidal current: $2\pi B_\phi R/\mu_0$
pit	$I_p(\rho)$	Tm	Toroidal current: $2\pi B_\chi r/\mu_0$

bpsd_equ2D_data

psi	$\psi_p(R,Z)$	Tm^2	2D poloidal magnetic flux

bpsd_equ3D_data

psi	$\psi_p(R,Z,\phi)$	Tm^2	3D poloidal magnetic flux

bpsd_metric1D_data: $V'(\rho), \langle\nabla V\rangle(\rho), \cdots$

bpsd_metric2D_data: $g_{ij}, \cdots$

bpsd_metric3D_data: $g_{ij}, \cdots$

Defined Dataset (3)

bpsd_plasmaf_data

pn	$n_s(\rho)$	m^3	Number density
pt	$T_s(\rho)$	eV	Temperature
ptpr	$T_{s\|\|}(\rho)$	eV	Temperature
ptpp	$T_{s\perp}(\rho)$	eV	Temperature
pu	$u_{s\phi}(\rho)$	m/s	Toroidal rotation velocity
qinv	$1/q(\rho)$		Inverse of safety factor

bpsd_plasmak_data

fp	$f(p, \theta_p, \rho)$	momentum dist. fn at $\chi = 0$

bpsd_trmatrix_data

trd	$D(i, j, \rho)$	Diffusion matrix
tru	$u(i, \rho)$	Flow vector

bpsd_trsource_data

trs	$S(i, \rho)$	Source vector

Defined Dataset (4)

bpsd_dielectric_data

ceps	$\overleftrightarrow{\epsilon}(\rho,\chi,\zeta)$		Local dielectric tensor

bpsd_wavef_data

ce	$\boldsymbol{E}(\rho,\chi,\zeta)$	$\mathrm{V/m}$	Complex wave electric field

bpsd_waver_data

rray	$R(\ell)$	m	R of ray at length ℓ
zray	$Z(\ell)$	m	Z of ray at length ℓ
phiray	$\phi(\ell)$	rad	ϕ of ray at length ℓ
ceray	$\boldsymbol{E}(\ell)$	$\mathrm{V/m}$	Wave electric field at length ℓ
pwray	$\boldsymbol{P}(\ell)$	W	Wave power at length ℓ

bpsd_waveb_data

dray	$\boldsymbol{d}(\ell)$	m	Beam radius at length ℓ
vrau	$\boldsymbol{v}(\ell)$	$1/\mathrm{m}$	Beam curvature at length ℓ

Example of data structure: plasmaf

```
type bpsd_plasmaf_data
   real(kind=rkind) :: pn     ! Number density [m^-3]
   real(kind=rkind) :: pt     ! Temperature [eV]
   real(kind=rkind) :: ptpr   ! Parallel temperature [eV]
   real(kind=rkind) :: ptpp   ! Perpendicular temperature [eV]
   real(kind=rkind) :: pu     ! Parallel flow velocity [m/s]
end type bpsd_plasmaf_data

type bpsd_plasmaf_type
   real(kind=rkind) :: time
   integer :: nrmax      ! Number of radial points
   integer :: nsamax     ! Number of particle species
   integer, dimension(:) :: ns_nsa  ! Species specifier
   real(kind=rkind), dimension(:), allocatable :: rho  ! norm. radius
   real(kind=rkind), dimension(:), allocatable :: qinv ! 1/q
   type(bpsd_plasmaf_data), dimension(:,:), allocatable :: data
end type bpsd_plasmaf_type
```

BPSD Code Interface

- **bpsd_set_data(data,ierr)**:
  - **Copy data into internal dataset**
- **bpsd_get_data(data,ierr)**:
  - **Copy of interpolate data fram internal dataset**
  - If `nrmax=0`, copy data; otherwise interpolate for given mesh.
- **bpsd_save(ierr)**:
  - **Save all BPSD data into a file**
  - Name of the file is optional.
- **bpsd_load(ierr)**:
  - **Load all BPSD data from a file**
  - Name of the file is optional.
- **Interfaces for history archivng are under consideration.**

Examples of sequence in a component

- **TR_EXEC(dt)**: Transport time evolution

```
call bpsd_get_data(plasmaf,ierr)
call bpsd_get_data(metric1D,ierr)
local data <- plasmaf,metric1D
advance time step dt
plasmaf <- local data
call bpsd_set_data(plasmaf,ierr)
```

- **EQ_CALC**: Equilibrium calculation

```
call bpsd_get_data(plasmaf,ierr)
local data <- plasmaf
calculate equilibrium
update plasmaf
call bpsd_set_data(plasmaf,ierr)
equ1D,metric1D <- local data
call bpsd_set_data(equ1D,ierr)
call bpsd_set_data(metric1D,ierr)
```

Several Approaches on Workflow

- **Monolithic code approach**: original approach
  - **Memory-based data exchange**
  - **Template**:
    ```
    call bpsd_get_data
    calculation
    call bpsd_set_data
    ```
- **Command approach**: for script and workflow
  - **File-based data exchange**
  - **Template**:
    ```
    call bpsd_load ⟵ bpsddata
    call bpsd_get_data
    calculation
    call bpsd_set_data
    call bpsd_save ⟶ bpsddata
    ```
- **Pre- and post- process approach**: no modification of the code
  - **pre-process**: bpsddata $\Longrightarrow$ input file
  - **post-process**: output file $\Longrightarrow$ bpsddata

SOLPS-IMPGYRO-EDDY integrated code

SOLPS :
Plasma fluid code + Neutral Monte-Carlo Kinetic code

electron /
hydrogen isotope
- density
  temperature
  flow velocity
  potential

MPI_BCAST
to all IMPGYRO
processes

MPI_RECV

electron /
hydrogen isotope
- particle source
  momentum source
  heat source

impurity
- density
  flow velocity
  temperature

MPI_SEND
(after MPI_REDUCE
within IG processes)

IMPGYRO: Impurity Monte-Carlo kinetic code with the GYRO-motion

impurity particles
- incident energy
  incident angle

call as a subroutine
(each IMPGYRO process
call EDDY separately)

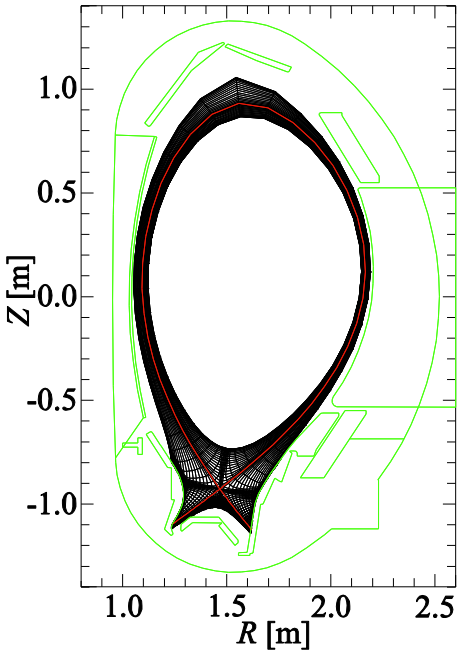
impurity particles
- emission energy
  emission angle

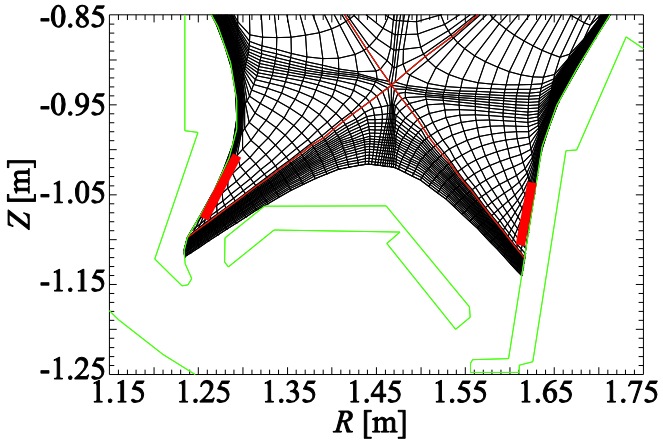
return

EDDY:
Erosion and Deposition code based on the DYnamic method

Example of SOLPS-IMPGYRO-EDDY (S-I-E) integrated code

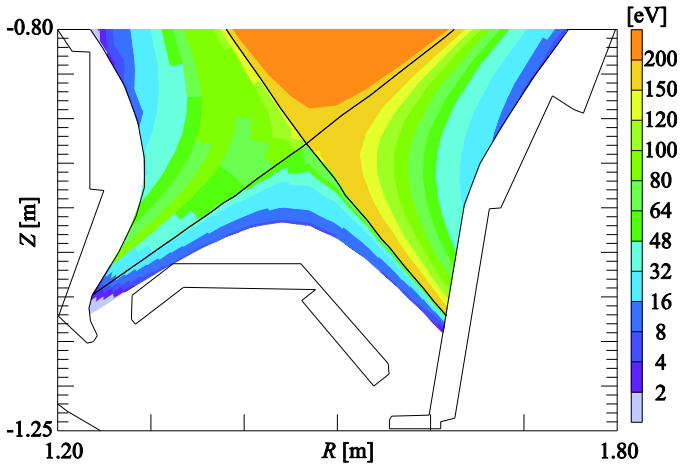
Geometry and Numerical Mesh



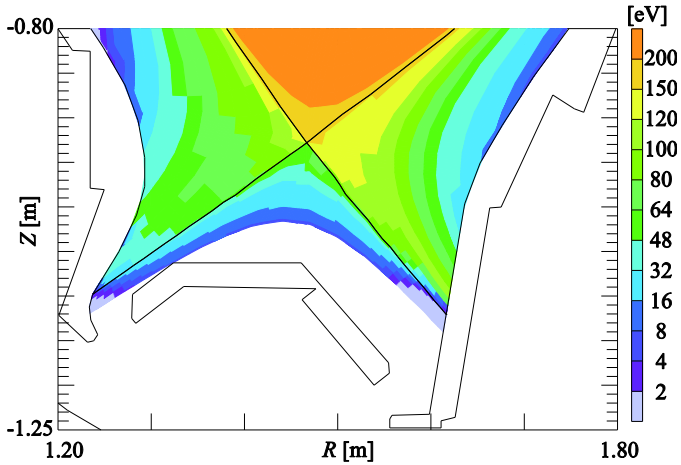


Numerical grids for the S-I-E coupling in ASDEX Upgrade geometry

Zoom-in view of the diverter region :
Test tungsten particles of the **1st generation** are launched **uniformly** along the **red lines**



2D elec. Temperature profile without tungsten impurities



2D elec. temperature profile with tungsten impurities

Required Data & Interface

Main Data required for analyses of SOL/divertor plasma

- MHD equilibrium data
- Wall structure/material data
- Plasma data at the Core / SOL boundary (e.g., density, temperature, …)

Atomic & Molecular data

- Ionization/recombination/excitation rate coefficient/cross section
- Radiation loss function

PWI data

- Particle and Energy Reflection Coefficient at the wall,
- Sputtering yield, Emission energy, Emission angle

Integrated code option : Eddy code (EDDY) has been directly coupled.

data is transferred by arguments of the subroutine

Data base option : TRIM data also can be used.

* At present, A&M data/PWI data are saved in the data files and these data files are accessed before starting the main calculation.

Summary

- For integrated modeling of toroidal plasmas, **platform for data exchange and execution control** is necessary.
- We have been developing **data exchange interface, BPSD**, and implemented in the TASK code.
- Our policy is simple dataset, simple interface, and sufficient flexibility.
- There are several approaches for **workflow**. We are developing
  - **monolithic approach** (memory-based data exchange)
  - **command approach** (file-based data exchange)
  - **pre- and post-process approach** and
  - **MPMD approach** (master-slave using MPI2).
- We have started the discussion on **the data interface for peripheral plasma modeling and atomic data**.

Thoughts on Future Development

- Dataset specification for **3D configuration** will be completed soon.
- **BPSD data viewer** for monitoring is now under development.
- **Data conversion** between integer mesh and half-integer mesh is troublesome.
- **Data interface between BPSD and IMAS** will be developed, if not incompatible.
- We are watching the IMAS decision on **workflow**
- **Web page**
  - TASK: http://bpsi.nucleng.kyoto-u.ac.jp/task/
  - BPSD: http://bpsi.nucleng.kyoto-u.ac.jp/bpsd/
- **English specification of BPSD** will be uploaded by tomorrow morning.