Lecture Series on Integrated Simulation of Tokamak Plasmas
SouthWestern Institute of Physics, Chengdu, China
2019/08/12-15

2. How to Install and Run the TASK code

A. Fukuyama

Professor Emeritus, Kyoto University

Acknowledgments

M. Honda, H. Nuga,
and many graduate students
in Okayama University and Kyoto University

Contents of Lecture 2

- How to download the TASK code
- Basics of version control and code storage git
- Install of graphic library GSAF
- Install of data model library BPSD
- Install of integrated code TASK
- Structure of a TASK module
- Run the equilibrium module task/eq
- Parallel processing with MPI
- Run integrated code

Installing TASK and related libraries

- **Home page**:
  - **https://bpsi.nucleng.kyoto-u.ac.jp/task/**
- **Download from git server**:
  - **git: an open-source distributed version control system**
  - **Download only**
    - Download using https protocol
    - No registration required
    - No upload permitted
  - **Download and upload in future** (Bug fix and new module)
    - Download using ssh protocol
    - Registration of an account name and SSH public keys required
    - Upload of the code permitted

Install TASK (1)

- **Check git available**: just command input “git”
  - if not, “sudo apt-get install git” for Ubuntu.
  - sudo apt-get install gcc for gfortran and gcc
  - sudo apt-get install xorg-dev for X-window
- **Set your identification**: Who changed the code?
  - git config - -global user.name “[your-full-name]”
  - git config - -global user.email [your-mail-address]
  - For example,
    - git config - -global user.name “Atsushi Fukuyama”
    - git config - -global user.email fukuyama@nucleng.kyoto-u.ac.jp
  - Data is saved in $HOME/.gitconfig
- **Create a working directory**: any directory name is OK
  - mkdir git
  - cd git

Install TASK (2)

- **Download TASK and necessary libraries** for download only
  - git clone https://git@bpsi.nucleng.kyoto-u.ac.jp/pub/git/gsaf.git
  - git clone https://git@bpsi.nucleng.kyoto-u.ac.jp/pub/git/bpsd.git
  - git clone https://git@bpsi.nucleng.kyoto-u.ac.jp/pub/git/task.git
- **Download TASK and necessary libraries** for download and upload
  - git clone ssh://username@bpsi.nucleng.kyoto-u.ac.jp/pub/git/gsaf.git
  - git clone ssh://username@bpsi.nucleng.kyoto-u.ac.jp/pub/git/bpsd.git
  - git clone ssh://username@bpsi.nucleng.kyoto-u.ac.jp/pub/git/task.git
- **Three directories are created**
  - **gsaf**: graphic library
  - **bpsd**: data interface library
  - **task**: main TASK directory

How to use git (1)

- **Repositories**
  - **local**: in your machine
  - **remotes**: in remote servers
  - **remotes/origin**: in default server: bpsi.nucleng.kyoto-u.ac.jp
- **Branches**
  - **There are several branches for code development**
    - **master**: default, stable version, often rather old
    - **develop**: latest version, where I am working
    - **others**: branches for working specific modules
  - cd task
  - git branch : list branch names, local only
  - git branch -a : list branch names, local and remote

How to use git (2)

- **To use develop branch**
  - **Create local branch develop and associate it with remote develop**
  - git checkout -t -b develop origin/develop
  - git branch
- **Change working branch**
  - git checkout master
  - git checkout develop
- **Update working branch**
  - git pull
    - Your modification is kept, if committed.
    - If uncommitted modification remains, no overwrite.
    - use “git stash” to keep away your modification.
    - If there is a conflict with your committed modification, conflict are indicated in the file. Corrects them and “git pull” again.

How to use git (3)

- **To check your modification**
  - git status
- **To commit your modification with message**
  - git commit -a -m'*message*'
- **To list all modification**
  - git log
- **For more detail, visit**
  - https://git-scm.com/documentation

Install TASK (3)

- **Install graphic library GSAF** (start from directory git)
  - cd gsaf/src
  - cp ../arch/ubuntu-gfortran64-static/Makefile.arch .
  - **Edit Makefile.arch** to adjust BINPATH and LIBPATH
  - make
  - make install : if necessary use “sudo make install”
  - cd test
  - make
  - ./bsctest
  - 5
  - c
  - m: CR to change focus to original window
  - e
  - cd ../../..

Install TASK (4)

- **Setup make.header file**
  - cd task
  - cp make.header.rog make.header
  - **Edit make.header** to remove comments for target OS and compiler
- **Compile data exchange library BPSD**
  - cd ../bpsd
  - make
  - cd ../task
- **Compile TASK**: eq for example
  - cd eq
  - make libs
  - make
  - ./eq

How to use GSAF

- **At the beginning of the program**
  - **Set graphic resolution** (0: metafile output only, no graphics)
  - **commands**
    - **c**: continue
    - **f**: set metafile name and start saving
- **At the end of one page drawing**
  - **commands**
    - **c** or **CR**: change focus to original window and continue
    - **f**: set metafile name and start saving
    - **s**: start saving and save this page
    - **y**: save this page and continue
    - **n**: continue without saving
    - **d**: dump this page as a bitmap file “gsdump*n*”
    - **b**: switch on/off bell sound
    - **q**: quit program after confirmation

Graphic Utilities

- **Utility program**
  - **gsview**: View metafile
  - **gsprint**: Print metafile on a postscript printer
  - **gstoeps**: Convert metafile to eps files of each page
  - **gstops**: Convert metafile to a postscript file of all pages
  - **gstotgif**: Convert metafile to a tgif file for graphic editor tgif
  - **gstotsvg**: Convert metafile to a svg file for web browser
- **Options**
  - **-a**: output all pages, otherwise interactive mode
  - **-s ps**: output from page ps
  - **-e pe**: output until page pe
  - **-p *np***: output contiguous *np* pages on a sheet
  - **-b**: output without title
  - **-r**: rotate page
  - **-z**: gray output

Typical File Name of TASK

- `XXcomm.f90`: Definition of global variables, allocation of arrays
- `XXmain.f90`: Main program for standalone use, read XXparm file
- `XXmenu.f90`: Command input
- `XXinit.f90`: Default values (may still include XXparm.f90)
- `XXparm.f90`: Handling of input parameters
- `XXprep.f90`: Initialization of run, initial profile
- `XXexec.f90`: Execution of run
- `XXgout.f90`: Graphic output
- `XXfout.f90`: Text file output
- `XXsave.f90`: Binary file output
- `XXload.f90`: Binary file input

Typical input command

- When input line includes `=`, interpreted as a namelist input (e.g., `rr=6.5`)
- When the first character is not an alphabet, interpreted as line input
- `r`: Initialize profiles and execute
- `c`: Continue run
- `p`: Namelist input of input parameters
- `v`: Display of input parameters
- `s`: Save results into a file
- `l`: Load results from a file
- `q`: End of the program
- **Order of input parameter setting**
  - Setting at the subroutine `XXinit` in `XXinit.f90`
  - Read a namelist file `XXparm` at the beginning of the program
  - Setting by the input line

Parallel processing by MPI

- MPI interface
  - mtxp library
    - SUBROUTINE mtx_initialize

      dummy without MPI

      CALL MPI_initialize with MPI
    - **SUBROUTINE mtx_setup**: Initialize variables
    - **SUBROUTINE mtx_set_matrix**: Set matric coefficients
    - **SUBROUTINE mtx_set_vector**: Set initial solution vector
    - **SUBROUTINE mtx_set_source**: Set source vector
    - **SUBROUTINE mtx_solve**: Solve matric equation
    - **SUBROUTINE mtx_get_vector**: Get solution vector
    - **SUBROUTINE mtx_cleanup**: Release variables
    - **SUBROUTINE mtx_finalize**: MPI_finalize