SFINCS User Manual

Version 3

Revised May 21, 2015

Contents

CHAPTER 1 Overview

SFINCS is a nifty code [1].

1.1 `sfincs` vs. `sfincsScan`

The fortran part of `sfincs` solves the kinetic equation for each species at a single flux surface, a single value of E_r, and a single set of resolution parameters. However, often the goal is to determine the ambipolar E_r at one or more surfaces. For this task, the `sfincsScan` family of python scripts

Figure 1-1: Trajectories of trapped particles. (a) In tokamaks and omnigenous stellarators, all collisionless trajectories are confined. (b) In contrast, trapped particles in other stellarators may have a radial drift that does not average to zero.

CHAPTER 2

Installation

2.1 Requirements

To compile sfincs you need the PETSc library (real version, as opposed to complex version) and the serial HDF5 library. PETSc is used for iterative solution of large linear and nonlinear systems of equations, and HDF5 is used for saving output. We have developed and tested sfincs with PETSc versions 3.2 through 3.5. The commands in PETSc often change from version to version, so future versions of PETSc may require modifications to the sfincs source code.

Although sfincs can be run on a single processor, usually you want to run it in parallel. In this case, you need MPI, and you need either mumps or superlu_dist. (Note that superlu_distis a parallel library which is different from the serial library superlu). Both mumps and superlu_dist are parallelized libraries for direct solution of large sparse linear systems. PETSc has a built-in serial sparse direct linear solver, but it sometimes gives an error that there is a “zero pivot” when mumps and superlu_dist have no problem solving the system; therefore you may want to use mumps or superlu_dist even for serial runs. In our experience, mumps requires less memory and time than superlu_dist for solving a given linear system.

If you want to load VMEC wout files in netCDF format, then you need the netCDF library. This library is not required for loading ASCII-format VMEC wout files. If you want to compile sfincs without netCDF , then edit makefile so that no value is assigned to USE_NETCDF.

The plotting routines sfincsPlot and sfincsScanPlot require python 2.X, numpy, scipy, and matplotlib. These python libraries are not required by the core fortran part of sfincs.

Although older MATLAB versions of sfincs are included in the sfincs repository, MATLAB is not required for running the fortran version of sfincs.

2.2 Cloning the repository

The source code for `sfincs` is hosted in a `git` repository at `https://github.com/landreman/sfincs`. You obtain the `sfincs` source code by cloning the repository. This requires several steps.

1. Create an account on `github.com`, and sign in to `github`.
2. Go to your account settings page, by clicking the wrench icon on the top right.
3. Click on "SSH keys" on the left, and add an SSH key for the computer you wish to use. To do this, you may wish to read see the "generating SSH keys" guide which is linked to from that page.
4. From a terminal command line in the computer you wish to use, enter `git clone git@github.com:landreman/sfincs.git` to download the repository.

Any time after you have cloned the repository in this way, you can download future updates to the code by entering `git pull` from any subdirectory within your local copy.

2.3 Makefiles and environment variables

To use `sfincs` you must set the environment variable `SFINCS_SYSTEM`. (For example, using the `bash` shell on the `edison` computer, you would type

```
export SFINCS_SYSTEM=edison
```

at the command line or in your `.bashrc` startup script.) This variable is used in two ways. First, `make` uses this variable to looks for the appropriate makefile in the `makefiles` directory. Second, the `SFINCS_SYSTEM` environment variable affects the behavior of `sfincsScan` in several ways, such as determining the command used to submit jobs to the system's queue.

2.4 Setting up `sfincs` on a new system

If you are setting up `sfincs` on a new system, one for which there is no file `makefiles/makefile.XXX`, there are several things you need to do.

First, copy one of the existing makefiles, and edit it as appropriate.

Second, you will need to edit `utils/sfincsScan`. Look for the `if` block near the top with sections for `sfincsSystem = edison`,`hydra`, and `laptop`. Add an analogous block for your system to set the command used to submit jobs, and a `nameJobFile` function.

2.5 Make test

CHAPTER 3

Input Parameters

In this chapter we first describe all the parameters which can be included in the `input.namelist` file. Then we list some of the command-line flags associated with `PETSc` which can be useful.

3.1 The `general` namelist

`RHSMode`
Type: integer
Default: 1
When it matters: Always
Meaning: Option related to the number of right-hand sides (i.e. inhomogeneous drive terms) for which the kinetic equation is solved.

`RHSMode=1`: Solve for a single right-hand side.

`RHSMode=2`: Solve for 3 right-hand sides to get the 3x3 transport matrix. Presently implemented only for 1 species.

`RHSMode=3`: Solve for the 2x2 monoenergetic transport coefficients. When this option is chosen, `Nx` is set to 1 and only 1 species is used.

`outputFileName`
Type: string
Default: "sfincsOutput.h5"
When it matters: Always
Meaning: Name which will be used for the HDF5 output file. If this parameter is changed from the default value, `sfincsScan` will not work.

`saveMatlabOutput`
Type: Boolean
Default: false
When it matters: Always
Meaning: If this switch is set to true, Matlab m-files are created which store the system matrix, right-hand side, and solution vector. If an iterative solver is used, the preconditioner matrix is also saved. PETSc usually generates an error message if you ask to save Matlab output when the size of the linear system is more then 1400×1400, so usually this setting should be false except for very small test problems.

`MatlabOutputFilename`
Type: string
Default: “sfincsMatrices”
When it matters: Only when `saveMatlabOutput == .true.`
Meaning: Start of the filenames which will be used for Matlab output.

`saveMatricesAndVectorsInBinary`
Type: Boolean
Default: false
When it matters: Always
Meaning: If this switch is set to true, the matrix, right-hand-side, and solution of the linear system will be saved in PETSc’s binary format. The preconditioner matrix will also be saved if `tryIterativeSolver == .true.`

`binaryOutputFilename`
Type: string
Default: “sfincsBinary”
When it matters: Only when `saveMatricesAndVectorsInBinary == .true.`
Meaning: Start of the filenames which will be used for binary output.

`solveSystem`
Type: Boolean
Default: true
When it matters: Always
Meaning: If this parameter is false, the system of equations will not actually be solved. Sometimes it can be useful to set this parameter to `.false.` when debugging.

3.2 Directives for `sfincsScan`

The parameters for `sfincsScan` begin with the code `!ss` and so are not read by the fortran part of `sfincs`. These parameters matter only when `sfincsScan` is called and are all ignored when `sfincs` is executed directly. These parameters can appear anywhere in the `input.namelist` file, in any namelist or outside of any namelist. Note that `sfincsScan` parameters do not have defaults, unlike fortran namelist parameters.

scanType
Type: integer
When it matters: Any time `sfincsScan` is called.
Meaning: Which type of scan will be run when `sfincsScan` is called.

`scanType=1`: Convergence scan. (Scan the parameters in the resolutionParameters namelist.)

`scanType=2`: Scan of E_r.

`scanType=3`: Scan any one input parameter that takes a numeric value.

`scanType=4`: Scan radius, taking the density and temperature profiles from the `profiles` file. In this type of scan, the same radial electric field is used at every radius. See `utils/profiles.XXX` for examples.

`scanType=5`: Scan radius, and at each radius, scan E_r. Density and temperature profiles are again taken from the `profiles` file; see `utils/profiles.XXX` for examples.

`scanType=21`: Read in a list of requested runs from a file `runspec.dat`. See `utils/sfincsScan_21` for an example file.

3.3 PETSc commands

Command-line flags can be used to modify the behavior of any `PETSc` application, including `sfincs`. There are hundreds of `PETSc` options, and a list can be obtained by running with the command-line flag `-help`. Here we list some of the more useful options.

-help
Meaning: Dumps a list of available command-line options to stdout.

-ksp_view
Meaning: Dumps detailed information to stdout related to the linear solver.

CHAPTER 4

Numerical resolution parameters

4.1 Convergence testing

4.2 Typical resolution requirements

CHAPTER 5

Parallelization

Choosing the number of nodes and procs.

5.1 Issues with running on 1 processor

References

[1] M. Landreman, H. M. Smith, A. Mollén, and P. Helander. *Phys. Plasmas*, **21**, 042503 (2014).