BDISTRIB User Manual

Revised August 25, 2015

Contents

CHAPTER 1

Overview

This program computes efficiency-ordered distributions of magnetic fields as described in [1]. The main method is described on page 12, and some key definitions are given on page 9.

There are some similarities between this program and the `NESCOIL` code used to compute stellarator coils [2]. `NESCOIL` effectively computes the 'inductance' between a plasma surface and a coil surface, meaning the linear relationship by which the current potential on the coil surface gives rise to a normal component of the magnetic field $\mathbf{B}$ on the plasma surface. `NESCOIL` takes the normal component of $\mathbf{B}$ on the plasma surface associated with net poloidal coil current plus plasma current, and effectively multiplies the result by the (pseudo)inverse of the inductance matrix, yielding the current potential required to achieve a magnetic surface (zero normal component of $\mathbf{B}$) on the plasma surface. In `bdistrib`, however, we do not consider net coil current or plasma current, and we compute two inductance matrices instead of one. The two inductance matrices in `bdistrib` refer to the same outer surface (on which the current potential is defined), but to different surfaces on which the normal $\mathbf{B}$ is evaluated. Multiplying one inductance matrix by the (pseudo)inverse of the other gives the so-called 'transfer matrix'. Applying a singular value decomposition (SVD) to this transfer matrix, the singular vectors represent modes of the magnetic field, and the singular values represent the efficiency by which these modes propagate from one toroidal surface to the other.

In `bdistrib` the three toroidal surfaces are named 'plasma', 'middle', and 'outer', in order from the innermost to outermost. The 'plasma' surface can correspond to the outermost surface of a `vmec` equilibrium, or it can be a plain circular toroidal surface. The name of the relevant surface appears as a suffix on most variables.

On each surface we use a poloidal angle u and a toroidal angle v, defined as in `NESCOIL`. The coordinate u lies in the range $[0, 1]$. The surfaces have `nfp` identical toroidal periods, and an increase in v by 1 corresponds to 1 of these periods. Thus, v increases by `nfp` in a complete toroidal revolution. In the output file, there is an array `v` corresponding to one toroidal period, as well as an array `vl` corresponding to all `nfp` toroidal periods. Note that v is proportional to the standard cylindrical angle ϕ: $\phi = 2\pi v/\texttt{nfp}$. Generally, field lines are not straight in the (u, v) coordinates. On the plasma surface, u is identical to the `vmec` poloidal angle divided by 2π, while the `vmec` toroidal angle differs by $2\pi/\texttt{nfp}$compared to v.

In the various variable names in the code and output file, 'r' refers to the position vector, not to a radius. In various arrays with a dimension of length 3, this dimension always corresponds to Cartesian coordinates (x, y, z).

The 'normal' quantities in the code and output file refer to the surface normal vector N = (dr/dv) cross (dr/du) as in the NESCOIL paper. Note that this vector does not have unit magnitude.

1.1 Required libraries

- `LIBSTELL` (for reading `vmec wout` files, and for the `ezcdf` subroutines)
- `NetCDF` (for writing the output file)
- `BLAS/LAPACK` (for matrix multiplication and the SVD subroutine)

If `OpenMP` is available, calculations with the code are parallelized. The plotting and testing functions use `python`, `numpy`, and `scipy`. The plotting routines `bdistribPlot` and `compareSingularValuesPlot` use `matplotlib`.

1.2 Parallelization

The code does not use `MPI`, and so it runs on a single computing node. However, it is possible to use multiple threads on the node to accelerate computations. The multi-threaded parallelization is done in part using `OpenMP` and in part using a multi-threaded `BLAS` routine.

The slowest steps in `bdistrib` occur when assembling the two inductance matrices. For each of these two matrices, there are two slow steps. The first is computation of the magnetic dipole formula between each pair of points on the two toroidal surfaces. The loop for this computation is parallelized using `OpenMP`. The other slow step is the integration of this result against each of the basis functions, which is done using matrix multiplication with the `BLAS` subroutine `DGEMM`. To parallelize this step you can link `bdistrib` with a multi-threaded `BLAS` library, such as the Intel Math Kernel Library (MKL).

1.3 `make test`

To test that your `bdistrib` executable is working, you can run `make test`. Doing so will run `bdistrib` for some or all of the examples in the `examples/` directories. After each example completes, several of the output quantities (typically the singular values of the transfer matrix) will be checked, using the `tests.py` script in the example's directory. The `make test` feature is very useful when making changes to the code, since it allows you to check that your code modifications have not broken anything and that previous results can be recovered.

If you run `make retest`, no new runs of `bdistrib` will be performed, but the `tests.py` script will be run on any existing output files in the `/examples/` directories.

1.4 Questions, Bugs, and Feedback

We welcome any contributions to the code or documentation. For write permission to the repository, or to report any bugs, provide feedback, or ask questions, contact Matt Landreman at `matt.`

landreman@gmail.com

CHAPTER 2

Theory

2.1 Requirements

CHAPTER 3

Input Parameters

In this section we describe all the parameters which can be included in the input namelist.

3.1 Resolution parameters

For any new set of surface geometries you consider, you should vary the resolution parameters in this section to make sure they are large enough. These parameters should be large enough that the code results you care about are unchanged under further resolution increases.

---

nu_plasma
Type: integer
Default: 64
When it matters: Always
Meaning: Number of grid points in poloidal angle used to evaluate the integral To do: XXX in the outer-to-plasma surface inductance matrix. Often 64 is a good value. It is resonable and common but not mandatory to use the same value for nu_plasma, nu_middle, and nu_outer.

---

nu_middle
Type: integer
Default: 64
When it matters: Always
Meaning: Number of grid points in poloidal angle used to evaluate the integral To do: XXX in the outer-to-middle surface inductance matrix. Often 64 is a good value. It is resonable and common but not mandatory to use the same value for nu_plasma, nu_middle, and nu_outer.

---

nu_outer
Type: integer
Default: 64
When it matters: Always

Meaning: Number of grid points in poloidal angle used to evaluate the integral To do: XXX in the outer-to-plasma surface and outer-to-middle surface inductance matrices. Often 64 is a good value. It is resonable and common but not mandatory to use the same value for nu_plasma, nu_middle, and nu_outer.

---

nv_plasma
Type: integer
Default: 64
When it matters: Always
Meaning: Number of grid points in poloidal angle used to evaluate the integral To do: XXX in the outer-to-plasma surface inductance matrix. A value of 64 is often good when the number of field periods is more than 1. A higher value like 256 may be needed when the number of field periods is 1. It is resonable and common but not mandatory to use the same value for nv_plasma, nv_middle, and nv_outer.

---

nv_middle
Type: integer
Default: 64
When it matters: Always
Meaning: Number of grid points in poloidal angle used to evaluate the integral To do: XXX in the outer-to-middle surface inductance matrix. A value of 64 is often good when the number of field periods is more than 1. A higher value like 256 may be needed when the number of field periods is 1. It is resonable and common but not mandatory to use the same value for nv_plasma, nv_middle, and nv_outer.

---

nv_outer
Type: integer
Default: 64
When it matters: Always
Meaning: Number of grid points in poloidal angle used to evaluate the integral To do: XXX in the outer-to-plasma surface and outer-to-middle surface inductance matrices. A value of 64 is often good when the number of field periods is more than 1. A higher value like 256 may be needed when the number of field periods is 1. It is resonable and common but not mandatory to use the same value for nv_plasma, nv_middle, and nv_outer.

---

mpol_plasma
Type: integer
Default: 8
When it matters: Always
Meaning: Maximum poloidal mode number to include for the Fourier basis functions on the plasma surface. The value can be positive or zero. It is resonable and common but not mandatory to use the same value for mpol_plasma, mpol_middle, and mpol_outer.

---

mpol_middle
Type: integer

Default: 8
When it matters: Always
Meaning: Maximum poloidal mode number to include for the Fourier basis functions on the middle surface. The value can be positive or zero. It is resonable and common but not mandatory to use the same value for mpol_plasma, mpol_middle, and mpol_outer.

---

mpol_outer
Type: integer
Default: 8
When it matters: Always
Meaning: Maximum poloidal mode number to include for the Fourier basis functions on the outer surface. The value can be positive or zero. It is resonable and common but not mandatory to use the same value for mpol_plasma, mpol_middle, and mpol_outer.

---

ntor_plasma
Type: integer
Default: 8
When it matters: Always
Meaning: Maximum toroidal mode number to include for the Fourier basis functions on the plasma surface. The value can be positive or zero. It is resonable and common but not mandatory to use the same value for ntor_plasma, ntor_middle, and ntor_outer. You can set the three ntor parameters to zero to examine only axisymmetric modes.

---

ntor_middle
Type: integer
Default: 8
When it matters: Always
Meaning: Maximum toroidal mode number to include for the Fourier basis functions on the middle surface. The value can be positive or zero. It is resonable and common but not mandatory to use the same value for ntor_plasma, ntor_middle, and ntor_outer. You can set the three ntor parameters to zero to examine only axisymmetric modes.

---

ntor_outer
Type: integer
Default: 8
When it matters: Always
Meaning: Maximum toroidal mode number to include for the Fourier basis functions on the outer surface. The value can be positive or zero. It is resonable and common but not mandatory to use the same value for ntor_plasma, ntor_middle, and ntor_outer. You can set the three ntor parameters to zero to examine only axisymmetric modes.

3.2 Geometry parameters for the plasma surface

geometry_option_plasma
Type: integer

Default: 0
When it matters: Always
Meaning: This option controls which type of geometry is used for the plasma surface.

`geometry_option_plasma` = 0: The plasma surface will be a plain circular torus. The major radius will be `R0_plasma`. The minor radius will be `a_plasma`.

`geometry_option_plasma` = 1: Identical to option 0. (This option exists just for consistency with `geometry_option_middle` and `geometry_option_outer`.)

`geometry_option_plasma` = 2: The plasma surface will be the last surface in the `vmec` file specified by `woutFilename`.

`R0_plasma`
Type: real
Default: 10.0
When it matters: Only when `geometry_option_plasma` is 0 or 1.
Meaning: Major radius of the plasma surface, when this surface is a plain circular torus.

`a_plasma`
Type: real
Default: 0.5
When it matters: Only when `geometry_option_plasma` is 0 or 1.
Meaning: Minor radius of the plasma surface, when this surface is a plain circular torus.

`nfp_imposed`
Type: integer
Default: 1
When it matters: Only when `geometry_option_plasma` is 0 or 1.
Meaning: When the plasma surface is a plain circular torus, only toroidal mode numbers that are a multiple of this parameter will be considered. This parameter thus plays a role like `vmec`'s `nfp` (number of field periods), and is used when `nfp` is not already loaded from a `vmec` file.

`woutFilename`
Type: string
Default: ''
When it matters: Only when `geometry_option_plasma` is 2.
Meaning: Name of the `vmec wout` output file which will be used for the plasma surface. You can use either a `netCDF` or `ASCII` format file.

3.3 Geometry parameters for the middle surface

`geometry_option_middle`
Type: integer
Default: 0

When it matters: Always
Meaning: This option controls which type of geometry is used for the middle surface.

`geometry_option_middle` = 0: The middle surface will be a plain circular torus. The major radius will be the same as the plasma surface: either `R0_plasma` if `geometry_option_plasma` is 0 or 1, or `Rmajor_p` from the `vmec wout` file if `geometry_option_plasma` is 2. The minor radius will be `a_middle`.

`geometry_option_middle` = 1: Identical to option 0, except the major radius of the middle surface will be set by `R0_middle`.

`geometry_option_middle` = 2: The middle surface will computing by expanding the plasma surface uniformly by a distance `separation_middle`.

`geometry_option_middle` = 3: The middle surface will be the 'coil' surface in the `NESCOIL` 'nescin' input file specified by `nescin_filename_middle`.

`R0_middle`
Type: real
Default: 10.0
When it matters: Only when `geometry_option_middle` is 1.
Meaning: Major radius of the middle surface, when this surface is a plain circular torus.

`a_middle`
Type: real
Default: 1.0
When it matters: Only when `geometry_option_middle` is 0 or 1.
Meaning: Minor radius of the middle surface, when this surface is a plain circular torus.

`separation_middle`
Type: real
Default: 0.2
When it matters: Only when `geometry_option_middle` is 2.
Meaning: Amount by which the middle surface is offset from the plasma surface.

`nescin_filename_middle`
Type: string
Default: ''
When it matters: Only when `geometry_option_middle` is 3.
Meaning: Name of a `NESCOIL nescin` input file. The coil surface from this file will be used as the middle surface for `bdistrib`.

3.4 Geometry parameters for the outer surface

geometry_option_outer
Type: integer
Default: 0
When it matters: Always
Meaning: This option controls which type of geometry is used for the outer surface.

geometry_option_outer = 0: The outer surface will be a plain circular torus. The major radius will be R0_outer. The minor radius will be a_outer.

geometry_option_outer = 1: Identical to option 0.

geometry_option_outer = 2: The outer surface will computing by expanding the plasma surface uniformly by a distance separation_outer.

geometry_option_outer = 3: The outer surface will be the 'coil' surface from the NESCOIL 'nescin' input file specified by nescin_filename_outer.

R0_outer
Type: real
Default: 10.0
When it matters: Only when geometry_option_outer is 1.
Meaning: Major radius of the outer surface, when this surface is a plain circular torus.

---

a_outer
Type: real
Default: 1.5
When it matters: Only when geometry_option_outer is 0 or 1.
Meaning: Minor radius of the outer surface, when this surface is a plain circular torus.

---

separation_outer
Type: real
Default: 0.4
When it matters: Only when geometry_option_outer is 2.
Meaning: Amount by which the outer surface is offset from the plasma surface. (Not the offset between the outer and *middle* surfaces!).

---

nescin_filename_outer
Type: string
Default: ''
When it matters: Only when geometry_option_outer is 3.
Meaning: Name of a NESCOIL nescin input file. The coil surface from this file will be used as the outer surface for bdistrib.

3.5 Other parameters

---

basis_set_option
Type: integer
Default: 1
When it matters: Always
Meaning: Determines which set of basis functions is used.

`basis_set_option` = 1: Use $\sin(2\pi[mu + nv])$ basis functions.

`basis_set_option` = 2: Use $\cos(2\pi[mu + nv])$ basis functions.

`basis_set_option` = 3: Use both $\sin(2\pi[mu + nv])$ and $\cos(2\pi[mu + nv])$ basis functions.

---

pseudoinverse_thresholds
Type: real array
Default: 1e-12
When it matters: Always
Meaning: To form the pseudoinverse of the outer-to-middle inductance matrix, singular values larger than this threshold will be replaced by their inverse, whereas singular values smaller than this threshold will be replaced by zero. You can supply more than one threshold value in order to compare results for different choices. Good threshold values are in the range 1e-6 to 1e-13, i.e. small compared to 1 but larger than machine precision.

---

n_singular_vectors_to_save
Type: integer
Default: 12
When it matters: Always
Meaning: Number of columns of U and V to save in the output file, where U and V are the matrices of left and right singular vectors of the transfer matrix, $T = U\Sigma V^T$. Regardless of this parameter, all singular *values* will be saved in the output file.

---

save_level
Type: integer
Default: 2
When it matters: Always
Meaning: Option related determining how many variables are saved in the `netCDF` output file. The larger the value, the smaller the output file.

`save_level` = 0: Save everything.

`save_level` = 1: Save everything except the inductance matrices.

`save_level` = 2: Save everything except the inductance matrices and the `drdu`, `drdv`, and `normal` arrays.

References

[1] A. Boozer. *Nucl. Fusion*, **55**, 025001 (2015).

[2] P. Merkel. *Nucl. Fusion*, **27**, 867 (1987).