Managing Device Configurations in Corsica

R. H. Bulmer and L. D. Pearlstein

2010-12-23 [Rev: 1.2]

Work performed under ITER Expert Contract IO/2010/ADM-107
Responsible Officer: Thomas Casper, ITER Organization

1 Introduction

Modifying Corsica models of tokamaks may be as simple as starting up the code with a save-file, making a few changes to device parameters in an interactive session, "running" an equilibrium with the changes, then making a new save-file. However, there are many device configuration parameters in Corsica and understanding *what you have* is not necessarily straightforward nor is determining *what you need to change*.

This document describes a procedure for managing changes in device configurations in Corsica by preparing human-readable *device-files* containing configuration specifications and then loading them into a session with one routine. The text files may be placed under a revision control system, along with the associated equilibrium save-files created with them, so that subsequent analyses can be readily associated with a particular device configuration.

In the following sections we describe this configuration management system in detail, using the ITER tokamak—which motivated the development of this procedure—as an example. Section 2 gives a brief overview of the procedure, which is all you need to read if you already have a set of device configuration input files and need to make a few changes to them. Section 3 describes how Corsica represents tokamak configurations—important to read if you are a new user. Section 4 describes how to prepare the device configuration input files. Finally, Section 5 describes some of the details associated with importing device-files into a Corsica model.

2 Overview

Magnetic configuration revisions are frequently issued by a project organization during the design phase and machine configurations are changed during construction and perhaps after operations have begun. The means by which device parameters are disseminated are, or course, project dependent, and may or may not lend themselves to automation. We will assume here that that a complete set of parameters defining the device are available and not concern ourselves with how the information gets translated into the device configuration input files.

Project data describing the poloidal field coils, vacuum vessel, first-wall, etc., must be translated into the set of eight device configuration files listed in Table 1. Page numbers in the table refer to page numbers in this document where the format and preparation of the input files are described in detail, including sample files for ITER.

Table 1: Device configuration input files

File-Name	*Page*	*File contents*
`coils.in`	14	Poloidal field coils and other toroidal current sources
`dgaps.in`	15	Diagnostic gap specifications
`limits.in`	16	Coil current, field and force limits
`params.in`	18	Nominal plasma and device parameters
`passive.in`	20	Passive structure specifications
`shape.in`	21	Target plasma boundary coordinates
`tfcoil.in`	22	Toroidal field coil specifications
`wall.in`	23	Plasma-facing first-wall, divertor and limiter geometry

The configuration input files are read by routines defined in Corsica standard script file[1] `device.bas`. The device-reader routines have names like `read_coils`, `read_wall`, etc. and are discussed in Section 5. In most cases we will load the entire set of files all at once by calling the `read_device` routine as described in the next paragraph.

Start-up with an old free-boundary save-file, read the device script and execute the top-level device-reader routine:

```
caltrans old.sav device.bas
corsica # or 'package eq'
call read_device
run
saveq("new.sav")
```

In this example the `read_device` routine reads configuration specifica-

[1] Standard script files are simply Corsica scripts that are available in all distributions and are located under the directory named by environment variable `CORSICA_SCRIPTS`.

tions and maps them to Corsica code variables. The `run` command executes a free-boundary equilibrium calculation using the new configuration specifications and the updated equilibrium with the new device configuration may be saved in a new save-file.

Note the `corsica` statement to put the code in `eq` mode in case it was in `ceq` mode. This is an important step if any configuration parameters are being used as independent variables in a `ceq` problem, in which case any changes to them by `read_device` will be overwritten when the `run` command is executed.

3 Device Representation in Corsica

This section describes how Corsica represents the various elements that comprise the device configuration for a tokamak equilibrium model. The user-generated specifications contained in the device configuration files listed in Table 1 are translated by the routines in the device-script into the Corsica model described in the subsections that follow.

3.1 Poloidal field coils

This subsection describes how poloidal field coils and other driven toroidal current sources, excluding the plasma, are modeled in Corsica. Elements with toroidal *passive* currents are described in Section 3.5.

3.1.1 Coil parameters

Corsica PF coil specification parameters are listed in Table 2. The coils may have rectangular or parallelogram cross-sections, following the EFIT [1] convention (see Figure 1, page 27). There are three counters used for coils: N_c, N_{PFC} and $N_{c_{plot}}$, where N_c is used to declare the size of the arrays containing the coil specifications. The number of driven coils, N_{PFC}, is normally equal to N_c, but in circuit calculations passive conductors are appended to the coil arrays so N_{PFC} is used to distinguish between driven and passive elements. The $N_{c_{plot}}$ counter is used by some graphics routines to restrict the number of coils shown on plots to a subset of the total.

Corsica represents coils with a uniformly-distributed array of filamentary current loops, where the number of filaments are the user-specified parameters $n_{\Delta R_c}$ and $n_{\Delta Z_c}$. Note the parallelogram coil sections are defined in a way (see Figure 1) that the cross-sectional area of the coil is always $\Delta R_c \times \Delta Z_c$. These current filaments are used in determining coil-plasma Green's functions used in solving the free-boundary Grad-Shafranov problem. In developing new tokamak configurations it is important to perform

Table 2: Poloidal field coil parameters

Parameter	*symbol*	*variable*	*units*
Number of coils	N_c	`nc`	–
No. driven coils	N_{PFC}	`npfc`	–
No. coils to plot	$N_{c_{plot}}$	`ncplot`	–
Coil name	–	`pfid(1:nc)`	–
Mean radius	R_c	`rc(1:nc)`	m
Axial position	Z_c	`zc(1:nc)`	m
Radial build	ΔR_c	`drc(1:nc)`	m
Axial build	ΔZ_c	`dzc(1:nc)`	m
No. radial filaments	$n_{\Delta R_c}$	`nrc(1:nc)`	–
No. axial filaments	$n_{\Delta Z_c}$	`nzc(1:nc)`	–
Type-1 angle	$\alpha_{1,c}$	`ac(1:nc)`	rad.
Type-2 angle	$\alpha_{2,c}$	`ac2(1:nc)`	rad.
No. turns	N	`ntc(1:nc)`	–
Resistivity	ρ_{eff}	`rhc(1:nc)`	Ω m
External inductance	L_{ext}/N^2	`lextc(1:nc)`	μH
External resistance	R_{ext}/N^2	`rextc(1:nc)`	$\mu\Omega$
Circuit index	i_c	`ic(1:nc)`	–
Gridding parameter	n_g	`ngp(1:nc)`	–

a sensitivity analysis in order to determine appropriate values of the number of filaments used to model each coil.

Coil diagnostic quantities, particularly the peak field and radial and axial forces, are evaluated by numerical quadrature assuming uniform current density in the coil, as described in Section 3.1.4, not with the filamentary Green's functions.

3.1.2 Coil circuit parameters

External inductance and resistance values must be reduced by the factor N^2 to be consistent with how Corsica evaluates coil inductance and resistance internally [2]. Corsica calculates the 1-turn inductance of each coil, L_1, using Gaussian quadrature over the rectangular or parallelogram cross-section, again assuming uniform current distribution. Thus, the actual coil inductance is:

$$L_{coil} = N^2 L_1$$

Similarly, the 1-turn resistance is evaluated with:

$$R_1 = \rho_{eff} \left(\frac{2\pi R_c}{\Delta R_c \Delta Z_c} \right)$$

and the actual coil resistance is:

$$R_{coil} = \rho \left(\frac{2\pi R_c N}{A_c} \right)$$

where A_c is the actual conducting area of each turn. Neglecting any inter-turn insulation, cooling channels, etc., $\rho_{eff} = \rho$ and $NA_c = \Delta R_c \times \Delta Z_c$ which leaves us with:

$$R_{coil} = N^2 R_1$$

but in general we must impose an effective resistivity:

$$\rho_{eff} = \rho \left(\frac{\Delta R_c \Delta Z_c}{NA_c} \right)$$

since most coils have internal insulation, coolant channels or may be sparsely wound.

3.1.3 One-turn voltage

With the 1-turn inductance and resistance values we introduce a symbol for 1-turn current, $I_1 = NI$, which is what Corsica evaluates in array `cc` when solving the Grad-Shafranov problem. We now write the Kirchhoff voltage equation for an isolated *R*-*L* circuit:

$$V = (R_{coil} + R_{ext})I + (L_{coil} + L_{ext})\frac{dI}{dt}$$

in terms of 1-turn quantities:

$$V_1 = \left(R_1 + \frac{R_{ext}}{N^2} \right) I_1 + \left(L_1 + \frac{L_{ext}}{N^2} \right) \frac{dI_1}{dt}$$

where $V_1 = V/N$. Corsica 2D array `pfim` contains the 1-turn inductance matrix, M_1, where the diagonal elements contain the external values:

$$M_{1_{i,i}} = L_{1_i} + \frac{L_{ext_i}}{N_i^2}$$

Similarly, the Corsica array `rxc` contains the total 1-turn resistance values, R_t:

$$R_{t_i} = R_{1,i} + \frac{R_{ext_i}}{N_i^2}$$

3.1.4 Coil diagnostics

The magnetic field distribution in selected PF coils is evaluated[2] with a modified version of EFFI [3] algorithms on grid points in each coil, the number of points being determined by the coil gridding parameter, n_g. The peak field is the maximum value of $|B|$ found on the grid for each coil and the average fields, $\langle B_R \rangle$ and $\langle B_Z \rangle$, are used to calculate the axial and radial forces.

[2] In Corsica, the switch `lop0` must be set to 1 to enable the evaluation of coil diagnostics.

The coil gridding parameter n_g determines the grid size Δs for each coil:

$$\Delta s = \frac{\min(\Delta R_c, \Delta Z_c)}{n_g - 1}$$

where we require $n_g \geq 2$ so that coil grid points always exist on the surface of the winding pack where the peak field usually occurs. The number of grid points across the minor dimension of the winding pack is then n_g and across the major dimension the integer value of:

$$n_{maj} = \frac{\max(\Delta R_c, \Delta Z_c)}{\Delta s} + \frac{1}{2}$$

Coil diagnostics are not calculated for coils where n_g is zero.

3.2 Diagnostic gaps

The coordinates of "diagnostic gap" positions and the minimum allowed distance from each gap location to the confined plasma (last closed flux surface) are stored in script-defined variables for use by auxiliary diagnostic routines. These diagnostic gap specifications are not preserved in save-files but in a separate portable database file.

3.2.1 Diagnostic gap parameters

At each gap location, two minimum allowable distances, or clearances, may be defined: one for low-power (LP) operation and one for full-power (FP) operation. Table 3 defines the diagnostic gap variables.

Table 3: Diagnostic gap quantities

Parameter	*variable*	*units*
Number of gaps	`ndg`	–
Radial coordinate	`rdg(1:ndg)`	m
Axial coordinate	`zdg(1:ndg)`	m
Minimum LP clearance	`dlpdgmn(1:ndg)`	mm
Minimum FP clearance	`dfpdgmn(1:ndg)`	mm
Gap identification	`iddg(1:ndg)`	–

3.3 Coil limits

Limits on current, field and forces are specified separately from the coil specifications described in Section 3.1. Coil limits include limits on individual coils and also limits for combinations of coils.

Coil limits are not automatically applied in Corsica. Their presence may trigger the evaluation of diagnostic quantities, but their application must be explicitly programmed by the user (see Section 3.3.5).

3.3.1 Individual coil limits

Limit specifications for individual coils are listed in Table 4. Current limits are generally applicable for non-superconducting coils and limit-line [4] criteria are used for superconducting coils. The `imaxc` array holds maximum conductor currents for each coil, and the `fzminc` and `fzmaxc` arrays hold minimum and maximum axial forces for each coil. The limit-line arrays `blimc` and `ilimc` hold the B and I-axis *intercepts* of the B-I limit-line for superconducting coils.

Table 4: Individual coil limits

Parameter	*symbol*	*variable*	*units*
Maximum current	I_{max}	`imaxc(1:nc)`	A
Minimum axial force	$F_{z_{min}}$	`fzmin(1:nc)`	MN
Maximum axial force	$F_{z_{max}}$	`fzmax(1:nc)`	MN
Limit-line field	B_{lim}	`blimc(1:nc)`	T
Limit-line current	I_{lim}	`ilimc(1:nc)`	A

The coil current magnitudes may be evaluated with Corsica expression `abs(cc/ntc)` and compared to the allowable values in `imaxc`. The radial and axial coil fores are available in the `pffr` and `pffz` arrays. When B_{lim} and I_{lim} are non-zero for a coil, the superconductor coil utilization factor, f_u, will be evaluated in the corresponding element of Corsica array `ufc`.

3.3.2 Imbalance current

Corsica evaluates the imbalance current for coil combinations, which are designated by two lists of coil names, one list for positive contributors and the other for negative contributors. The relevant Corsica parameters are listed in Table 5. The names of all coils whose circuit currents are to be *added* are specified in array `imbccp` and the names of all coils whose circuit currents are to be *deducted* are specified in array `imbccn`. The allowable imbalance current is held in variable `imbalmx`.

Table 5: Imbalance current specifications

Parameter	*variable*	*units*
No. positive contributors	`nccp`	–
No. negative contributors	`nccn`	–
Positive contributor names	`imbccp(1:nccp)`	–
Negative contributor names	`imbccn(1:nccp)`	–
Max. imbalance current	`imbalmx`	A

The presence of coil names in the contributor lists triggers evaluation of the imbalance current, returned in variable `imbal`.

3.3.3 PF force combinations

One set of coil combinations may be designated where their total axial force has upper and lower limits. The relevant parameters are given in Table 6. The presence of coil names in the `combos` list triggers evaluation of their

Table 6: Axial force limits for coil combinations

Parameter	*variable*	*units*
No. coils	`ncombos`	–
Coil names	`combos(1:ncombos)`	–
Minimum axial force	`fzcomn`	MN
Maximum axial force	`fzcomx`	MN

total axial force, returned in variable `fzcombos`.

3.3.4 CS axial forces

Central solenoid (CS) axial force criteria may be specified using the parameters shown in Table 7. Corsica assumes any coil having a name beginning

Table 7: Axial force limits for CS coils

Parameter	*variable*	*units*
Net axial force	`fzcsmxn`	MN
Repulsion force	`fzcsmxr`	MN

with "CS" is part of a central-solenoid assembly and is to be included in these force diagnostics evaluations. The two criteria are (1) net axial force on the CS coils and (2) repulsion force. The repulsion force is evaluated by first finding all $2 \times N_{CS}$ combinations of "upwards" and "downwards" axial forces ($F_{z,CS_{up}}$ and $F_{z,CS_{down}}$) which are returned in Corsica arrays `csfz_up` and `csfz_dn`. The repulsion force is then evaluated by averaging the magnitudes of the maximum and minimum values:

$$F_{CS_{rep}} = \frac{1}{2}\left(\max_{i=1}^{N_{CS}} F_{z,CS_{up}} + \left|\min_{i=1}^{N_{CS}} F_{z,CS_{down}}\right|\right)$$

The net axial force and repulsion force are returned in variables `csfz_net` and `csfz_rep`, and are evaluated whenever there are any coil names beginning with "CS".

3.3.5 Applying coil limits

Coil limits are not automatically applied in Corsica—they must be programmed by the user. As an example, the limit-line criteria can be applied by posing a constrained equilibrium problem using the `ceq` package. Say we want to to adjust the current in coil #1 so that it operates at its superconducting limit, $f_{u_1} = 1$, or `ufc(1)` in the code.

We can invoke the ceq package and exceute a one-constraint problem as follows:

```
package ceq
ic(1) = 0 # May need to adjust other ic values
nctot = 1
vo = "ufc(1)"; vo0 = 1
vi = "cc(1)"; x0 = cc(1)
ihy = 20; lop0 = 1; run
```

Here we set the circuit index ic(1) to zero so that its current will be fixed with each Grad-Shafranov solution. This may require adjustment of other ic values so that the smallest non-zero entry is 1. The number of constraints is indicated with nctot and we specify the constraint and its desired value in the vo and vo0 arrays. The independent variable is specified by name in array vi and it is initialized to its present value in x0. We then execute the constrained equilibrium solver, for up to 20 iterations, with coil diagnostics turned on. It will call the G-S solver at each iteration, adjusting the coil current until the constraint is satisfied.

3.4 Parameters

This subsection describes parameters that pertain to the configuration in general or quantities used for initialization purposes. These parameters, in the context of device configuration specifications, refer to a *small* set of informational attributes, nominal plasma parameters and some initial value settings for the coils. Realize that almost all plasma parameters are simply inherited from the contents of the equilibrium save-file currently in memory when a configuration is modified with the device-script.

The informational attributes are character-string variables: device_name, device_date and device_config that contain the name of the device, the date the device configuration was issued, and a long string that contains descriptive information about the configuration.

There is one plasma parameter presently considered a device parameter, the signed plasma current, I_p, stored in Corsica variable plcm [MA]. All other plasma parameters are simply defined by the equilibrium in memory when a new device configuration is loaded.

There are three initializations of coil arrays that are specified in the parameters file: circuit index array, ic, initial value of the coil currents (cc) and values of the coil gridding parameter stored in array ngp (see Section 3.1.4).

3.5 Passive structure

This subsection describes the passive structure model used in Corsica. The model was originally created for use by the variational stability package, VPF, but passive structure elements are often appended to the coil set for time-dependent circuit calculations.

3.5.1 Passive structure parameters

The definition of passive structure in Corsica is contained in so-called "wire" elements that have the same type of geometrical specifications as PF coils. The relevant variables are given in Table 8. The discretization of the rectangular or parallelogram cross-section into filamentary current loops is somewhat different for passive elements compared to PF coils. Since there are usually hundreds of passive elements required to model a passive component like a vacuum vessel, the number of filaments in each element is calculated using variable `nfils` (with a default value of 1) to determine the number of filaments across the minor dimension with the number of filaments across the major dimension made in proportion to the aspect ratio of the coil cross-section, with an upper limit of 10 filaments.

Table 8: Passive structure parameters

Parameter	*variable*	*units*
Number of elements	`nwires`	–
Name of element	`idwires(1:nwires)`	–
Mean radius	`rwiresc(1:nwires)`	m
Axial position	`zwires(1:nwires)`	m
Radial build	`drwires(1:nwires)`	m
Axial build	`dzwires(1:nwires)`	m
Type-1 angle	`awires(1:nwires)`	rad.
Type-2 angle	`awires2(1:nwires)`	rad.
Resistivity	`rhwires(1:nwires)`	$\Omega\,$m
External inductance	`lxwires(1:nwires)`	μH
External resistance	`rxwires(1:nwires)`	$\mu\Omega$

3.5.2 Passive structure initialization

Corsica script function `psm` can be used to initialize a passive structure model. It can group adjacent elements to expedite calculations, although this is usually not necessary with present-day computing capability. The `psm` routine calls the Corsica `vpfwire` subroutine to evaluate wire-wire mutual inductances and to calculate the resistance of the elements. It then lists the resistance of each component, where a component consists of groups of wire elements having the same name in `idwires`.

3.6 Reference plasma shape

This subsection describes how a reference plasma shape is characterized in Corsica. The reference plasma shape, a set of *R*-*Z* coordinates defining a "target" separatrix, can be used as both a shape constraint for the free-boundary Grad-Shafranov solver or be used simply as a basis of comparing plasma shapes during equilibrium calculations.

3.6.1 Shape parameters

In Corsica the plasma shape is defined by so-called "fuzzy-boundary" points wih an associated array of weight factors as shown in Table 9. When these points are used as shape constraints in a free-boundary equilibrium calculation (i.e., where $\alpha_{fbd} > 0$), coil currents are adjusted to match the fuzzy-boundary points in a least-squares sense. There are also "hard-boundary points" available in Corsica, where the coil currents will be adjusted to conform the plasma boundary to the hard points exactly. Generally a large number of fuzzy-boundary points will be used but just a few hard-boundary points.

Table 9: Fuzzy-boundary points

Parameter	*symbol*	*variable*	*units*
Number of points	N_{fbd}	`nfbd`	–
Radial coordinate	R_{fbd}	`rfbd(1:nfbd)`	m
Axial coordinate	Z_{fbd}	`zfbd(1:nfbd)`	m
Weight factor	α_{fbd}	`alfbd(1:nfbd)`	–

To use the fuzzy-boundary points only as a basis for shape comparison, set all elements of α_{fbd} to zero.

3.7 Toroidal field coils

This subsection describes how toroidal field coils are described in Corsica. Toroidal field coil specifications are not included in Corsica, but are recognized by some of the standard script routines to display TF coils in graphical output and to evaluate out-of-plane $\mathbf{J_{TF}} \times \mathbf{B_{pol}}$ forces on the coils.

3.7.1 TF coil parameters

The TF coil parameters are listed in Table 10, where † indicates the variables are defined in Corsica (and preserved in save-files). The other TF coil variables are script-defined and are therefore not preserved in save-files. The

Table 10: TF coil parameters

Parameter	*symbol*	*variable*	*units*
Number of TF coils	N_{TFC}	`ntfc`	–
Toroidal field	B_φ	`btor`†	G
Reference radius	R_{ref}	`ro`†	cm
No. center points	$N_{TFC_\odot}$	`ntfcpts`	–
Center *R*-coordinates	$R_{TFC_\odot}$	`rtfcpts(1:ntfcpts)`	m
Center *Z*-coordinates	$Z_{TFC_\odot}$	`rtfcpts(1:ntfcpts)`	m
No. inner points	N_{TFC_i}	`ntfi`	–
Inner *R*-coordinates	R_{TFC_i}	`rtfi(1:ntfcpts)`	m
Inner *Z*-coordinates	Z_{TFC_i}	`rtfi(1:ntfcpts)`	m
No. outer points	N_{TFC_o}	`ntfo`	–
Outer *R*-coordinates	R_{TFC_o}	`rtfo(1:ntfcpts)`	m
Outer *Z*-coordinates	Z_{TFC_o}	`rtfo(1:ntfcpts)`	m

number of TF coils is used to determine the total current in each winding:

$$NI_{TFC} = \frac{2\pi}{\mu_0}\frac{R_{ref}B_\varphi}{N_{TFC}}$$

in order to evaluate out-of-plane forces. The reference radius, R_{ref}, is the radial coordinate where the vacuum toroidal field has the value B_φ and also, by convention in Corsica, the mid-point of the *R*-*Z* grid. Furthermore, to minimize confusion, Corsica models often have the *nominal* major radius of the plasma, R_0, and the reference radius, R_{ref}, coincide.

The center points in Table 10 refer to the *R*-*Z* coordinates of the TF coil winding pack center-line, where the out-of-plane forces are evaluated. The inner and outer points refer to coordinates of the inner and outer periphery of the coil structure.

3.8 Wall representation

This subsection describes how the plasma-facing wall is specified in Corsica. Its use is optional, in that the code will operate without any wall elements since an independently specified limiter point may be defined by the user, which is often the case in the early stages of a new tokamak design.

The wall in Corsica may include any elements which can be represented by line segments, but it is often used for only plasma-facing surfaces such as power-absorbing limiter and divertor surfaces and the first-wall surrounding the plasma.

3.8.1 Wall parameters

A wall is specified with a set of "plate" elements, listed in Table 11. A plate

Table 11: Wall representation

Parameter	*symbol*	*variable*	*units*
Number of plates	N_{plates}	`nplates`	–
R coordinates	R_{plate}	`rplate(1:nplates,1:2)`	cm
Z coordinates	Z_{plate}	`zplate(1:nplates,1:2)`	cm

element is a 2D line segment defined by its end-point coordinates.

3.8.2 Using plate elements

The plate elements in Corsica are used to evaluate strike-point coordinates and they may also be used, if properly ordered, as a plasma-boundary limiting-surface in the free-boundary Grad-Shafranov solver.

The plasma-boundary limiting-surface feature is invoked by setting Corsica switch `limw` to 1. When activated, this feature will utililze a set of *smoothed* limiting-surface coordinates constructed from the plate elements and stored in arrays `rlimw(1:lmax)` and `zlimw(1:lmax)`. The number of elements, `lmax`, is derived from other Corsica parameters, which may be changed by the user, but this is seldom necessary. The smoothed limiter points may be viewed with the Corsica `pb` or `pbg` routines.

The free-boundary solver determines the limiting flux surface of the plasma from either a separatrix surface or a point on the limiting-surface by determining which is closer in poloidal flux to the magnetic axis. The actual limiting-point found within the limiting-surface elements is stored in `rlim(0)`,`zlim(0)`.

4 Device Configuration Files

This section describes how to prepare device configuration files to be read with the reader routines in the device-script. The reader routines translate the information from the device configuration files into the Corsica representation described in Section 3.

The following conventions are used in interpreting the contents of device configuration files:

1. blank lines are ignored;
2. comments begin with the # character and end with the newline character;
3. data records consist of one or more data elements and are separated by the newline character;
4. data elements are position-dependent and are separated by tab or space characters (multiple space characters are equivalent to one space);

5. a dash may be used to skip a data element (or they may simply be omitted if they are at the end of the data record);
6. keywords are sometimes required to identify subsequent data elements; and
7. numeric input quantities must be expressed in S. I. units unless otherwise stated.

The following subsections describe the device configuration input files for the (1) PF coils, (2) diagnostic gaps, (3) coil limits, (4) parameters, (5) passive structure, (6) plasma shape, (7) TF coils, and (8) first-wall, mirroring the order of Section 3. The sample files are for an ITER configuration [5].

4.1 Coils-file description

Poloidal field coil specifications and other toroidal current sources are specified in a *coils-file* (default file-name: `coils.in`), which is read by the `read_coils` routine.

4.1.1 Coils-file contents

Each data record consists of up to eleven items (see Table 2:

1. coil name, 8 characters or less, stored in Corsica array `pfid`;
2. mean radius of the winding pack, R_c, stored in array `rc`;
3. axial position of the coil, Z_c, stored in array `zc`;
4. radial build of the winding pack, ΔR_c, stored in array `drc`;
5. axial build of the winding pack, ΔZ_c, stored in array `dzc`;
6. type-1 parallelogram angle, α_1 [degrees], stored in array `ac`;
7. type-2 parallelogram angle, α_2 [degrees], stored in array `ac2`;
8. number of turns, N, stored in array `ntc`;
9. effective resistivity of the winding pack, ρ_{eff} [$\mu\Omega m$], stored in array `rhc`;
10. 1-turn external inductance, L_{ext}/N^2 [μH], stored in array `lextc`; and
11. 1-turn external resistance, R_{ext}/N^2 [$\mu\Omega$], stored in array `rextc`.

Corsica variables `nc`, `npfc` and `ncplot` containing: (a) the number of coils, (b) the number of driven coils and (c) the number of coils appearing in layout plots, will all be initialized to the number of coils specified in the input file.

4.1.2 Sample coils-file for ITER

```
# ITER Coil Specifications

# Date of issue...
#   CS & PF Coils:            2010-07-30
```

```
#   Stabilization (VS) coils: 2010-04-01
#   ELM Control (EC) coils:   2010-04-01
#   TF coil busbars:          2010-01-14

# Name       Rc       Zc      DRc     DZc  a1  a2      N   Resis    Lext   Rext
  PF1    3.9431   7.5741  0.9590  0.9841   0   0  248.6
  PF2    8.2851   6.5398  0.5801  0.7146   0   0  115.2
  PF3   11.9919   3.2752  0.6963  0.9538   0   0  185.9
  PF4   11.9630  -2.2336  0.6382  0.9538   0   0  169.9
  PF5    8.3908  -6.7469  0.8125  0.9538   0   0  216.8
  PF6    4.3340  -7.4665  1.5590  1.1075   0   0  459.4
  CS3L   1.6960  -5.4150  0.7340  2.1200   0   0  553
  CS2L   1.6960  -3.2450  0.7340  2.1200   0   0  553
  CS1L   1.6960  -1.0750  0.7340  2.1200   0   0  553
  CS1U   1.6960   1.0950  0.7340  2.1200   0   0  553
  CS2U   1.6960   3.2650  0.7340  2.1200   0   0  553
  CS3U   1.6960   5.4350  0.7340  2.1200   0   0  553
  VSU    5.8261   4.9249  0.116   0.116    0   0    4   0.1104   6.25  31.25
  VSL    7.5222  -2.4912  0.116   0.116    0   0    4   0.1104   6.25  31.25
  ECUA   7.7454   3.3938  0.138   0.138    0   0    6   0.0868  11.25
  ECUB   8.2736   2.6388  0.138   0.138    0   0    6   0.0868  11.25
  ECMA   8.6297   1.8018  0.138   0.138    0   0    6   0.0877   8.00
  ECMB   8.6728  -0.5415  0.138   0.138    0   0    6   0.0877   8.00
  ECLA   8.2413  -1.5391  0.138   0.138    0   0    6   0.0917  13.32
  ECLB   7.7818  -2.3757  0.138   0.138    0   0    6   0.0917  13.32
  TFCBB  5.297  -10.385   0.03    0.03     0   0    1
```

4.1.3 Comments on the sample coils-file

All "driven" coils and toroidal current sources are specified in a coils-file. In the sample file we include superconducting shaping coils (PF) and induction coils (CS), followed by resistive vertical stabilization coils (VS) and axisymmetric equivalents for resistive ELM control coils (EC) which approximate an $n = 0$ mode in the array of 27 (3 poloidal $\times$ 9 toroidal) saddle coils mounted on the interior of the vacuum vessel. Residual toroidal current from the TF coil interconnecting bus-work is also included. The `read_coils` routine will initialize the coil counters `nc`, `npfc` and `ncplot` to the number of records in the coils-file and calculate the number of filamentary current-loops for each coil (see Section 5.1).

Models of resistive coils require special attention [2] in Corsica. The ITER VS and EC coils are designed with a sparsely-packed array of annular conductors. The winding pack dimensions for the VS and EC coils are thereforfe based on the "bounding-box" dimensions of the actual windings in order to to produce valid inductance values in Corsica. The 1-turn external inductances and resistances for the VS coils represent bus-work values from independent sources. The external inductances for the EC coils make up the difference between the axisymmetric Corsica inductances and the inductances obtained from a 3D model of the ELM control coils.

4.2 Dgaps-file description

Diagnostic gap specifications are entered in a *dgaps-file* (with default name: `dgaps.in`) which is read by the `read_dgaps` routine. The specifications are not preserved in Corsica save-files, but in a separate portable database file named *`device`*`_dgaps.pfb`, where *`device`* is the lowercase device name from the params-file (see Section 4.4). This PFB file is written by the read_dgaps routine.

4.2.1 Dgaps-file contents

An input record consists of five items (see Table 3):

1. radial coordinate of gap location [m], stored in array rdg;
2. axial coordinate of gap location [m], stored in array zdg;
3. minimum allowable gap distance [mm] for low-power operation, stored in array dlpdgmn;
4. minimum allowable gap distance [mm] for full-power operation, stored in array dfpdgmn; and
5. gap identification string (up to 8 characters), stored in array iddg.

The gap description field, shown in the sample file below, is not retained.

4.2.2 Sample dgaps-file for ITER

```
# ITER Diagnostic Gap Specifications

# Date of issue: 2010-12-14

# Diagnostic gap positions (R,Z) and minimum allowable values [mm] for
# Low-Power (LP) and Full-Power (FP) operation.
#
#     R       Z    LP   FP  Name    Description
  4.178  -2.506  120  250  1bot    Bottom of FW1
  4.138  -2.003  120  250  1mid    Middle of FW1
  4.097  -1.500  120  250  n01/02  Intermediate node between FW1 and FW2
  4.082  -0.992   60  150  2mid    Middle of FW2
  4.067  -0.484   60  150  n02/03  Intermediate node between FW2 and FW3
  4.056   0.025    0   70  3mid    Middle of FW3
  4.046   0.533    0   70  n03/04  Intermediate node between FW3 and FW4
  4.046   1.041    0   70  4mid    Middle of FW4
  4.046   1.549    0   70  n04/05  Intermediate node between FW4 and FW5
  4.061   2.057    0   70  5mid    Middle of FW5
  4.076   2.566   60  150  n05/06  Intermediate node between FW5 and FW6
  4.101   3.074   60  150  6mid    Middle of FW6
  6.170   4.213   30  110  10mid   Middle of FW10
  6.994   3.536   20   50  11mid   Middle of FW11
  7.652   2.821   30  110  12mid   Middle of FW12
  8.087   2.073   30  110  13mid   Middle of FW13
  8.270   1.681   30  110  n13/14  Intermediate node between FW13 and FW14
  8.332   1.157    0   50  14mid   Middle of FW14
  8.350   0.106    0   50  15mid   Middle of FW15
  8.103  -0.881    0   50  16mid   Middle of FW16
  7.591  -1.799    0   50  17mid   Middle of FW17
  7.283  -2.257   20   90  n17/19  Intermediate node between FW17 and FW18
  6.775  -2.652   20   90  18mid   Middle of FW18
  6.267  -3.046   20   90  18bot   Bottom of 18
```

4.3 Limits-file description

Coil limits are specified in a *limits-file* (default name: limits.in) which is read by the read_limits routine. Field, force and current limits may be specified for individual coils or for coil combinations.

4.3.1 Limits-file contents

An input record consists of up to nine items listed below.

1. name field (see Section 4.3.2), one of:
   (a) individual coil name as defined in the coils-file,
   (b) coil combination name composed of individual coil names, or
   (c) pre-defined combination name,
2. maximum conductor current, $|I|_{max}$ [kA], stored in array `imaxc`;
3. superconductor field limit point #1, $B_{lim,1}$ [T] (see Section 4.3.3);
4. superconductor current limit point #1, $I_{lim,1}$ [kA] (see Section 4.3.3);
5. superconductor field limit point #2, $B_{lim,2}$ [T] (see Section 4.3.3);
6. superconductor current limit point #2, $I_{lim,2}$ [kA] (see Section 4.3.3);
7. minimum allowable axial force, $F_{z,min}$ [MN], stored in array `fzminc`;
8. maximum allowable axial force, $F_{z,max}$ [MN], stored in array `fzmaxc`; and
9. maximum imbalance current, I_{imbal} [kA], stored in scalar `imbalmx`.

4.3.2 Name field

The first element in a data record is the name field, which may be:

1. an individual coil name, such as "`PF1`";
2. a combination coil name composed of individual coil names concatenated with either "+" or "-" signs, where the sign indicates whether the values (either axial forces or conductor currents) are to be added or subtracted, e.g., "`PF2+PF3-PF4-PF5`"; or
3. a predefined name, either "`fzcsmxn`" or "`fzcsmxr`" to indicate the *net* axial force or *repulsive* force on the CS[3] coils. The definitions of these quantities are given in Section 3.3.4.

4.3.3 Field and current limits

Superconducting coils have *B*-*I* "limit-line" constraints on their operating point. The limit-line is defined as the line passing through two points, $(B_{lim,1}, I_{lim,1})$ and $(B_{lim,2}, I_{lim,2})$. The `read_limits` routine calculates the *B* and *I*-axis intercepts of the limit-line and stores them in the `blimc` and `ilimc` arrays. Additional information about *B*-*I* limit lines is given in [4].

4.3.4 Sample limits-file for ITER

```
# ITER Coil Limits

# Name     Imax  Blim1  Ilim1  Blim2  Ilim2  Fz_min  Fz_max  Imbal
#            kA      T     kA      T     kA      MN      MN     kA
  PF1         -    6.4     48    6.5     41    -150     110
  PF2         -    4.8     55    5.0     50     -75      15
  PF3         -    4.8     55    5.0     50     -90      40
  PF4         -    4.8     55    5.0     50     -40      90
  PF5         -    5.7     52    6.0     33     -10     160
```

[3] Corsica inspects the coil name in array `pfid` and assumes any names beginning the characters "`CS`" are part of a central-solenoid and CS force diagnostics are to be evaluated.

```
  PF6        -    6.8     52    7.0     41    -190     170        # Subcooled to 0.4 K
  CS3L       -   12.6     45   13.0     40
  CS2L       -   12.6     45   13.0     40
  CS1L       -   12.6     45   13.0     40
  CS1U       -   12.6     45   13.0     40
  CS2U       -   12.6     45   13.0     40
  CS3U       -   12.6     45   13.0     40
  VSU       60
  VSL       60
  ECUA      16
  ECUB      16
  ECMA      16
  ECMB      16
  ECLA      16
  ECLB      16

# Vertical force combinations
  PF3+PF4    -      -      -      -      -     -60      10

# Imbalance current
  PF2+PF3-PF4-PF5 - -      -      -      -       -       -   22.5

# CS allowable vertical forces (net and repulsion)
  fzcsmxn    -      -      -      -      -       -      60
  fzcsmxr    -      -      -      -      -       -     120
```

4.3.5 Comments on the sample limits-file

The first entries in the file show limits for all of the individual PF, CS, VS and EC coils. The superconducting coils (PF & CS) have *B-I* limits and the PF coils have individual axial force limits (axial force limits for the CS coils are defined only for coil combinations and are entered at the end of the limits file). The resistive (VS & EC) coils only have current limits on maximum current. The remainder of the file specifies limits on combinations of coils.

The PF3 and PF4 coils have limits on sum of their axial forces, so we enter the combination name "`PF3+PF4`" followed by a few dash characters to skip over irrelevant limit categories and enter values only in the $F_{z,min}$ and $F_{z,max}$ fields.

An imbalance current limit on the sum of the PF2 and PF3 conductor currents minus the sum of the PF4 and PF5 conductor currents is specified by entering the combination name "`PF2+PF3-PF4-PF5`" followed by seven dashes to skip to the I_{imbal} field.

Finally, the two predefined combinations of net and repulsion axial forces on the CS coils are entered in the 8th ($F_{z_{max}}$) field.

4.4 Params-file description

Parameters and settings for the configuration are defined in a *params-file*, which is read by the `read_params` routine with default name `params.in`.

4.4.1 Params-file contents

The format of this file consists entirely of case-insensitive keyword entries as defined in Table 12. The `device` keyword must be followed by a device

Table 12: Keyword records for a params-file

Keyword	*Data elements*
`device`	Name of the device and issue date of the configuration
`config`	Long string containing descriptive information about the configuration
`current`	Nominal plasma current [MA]
`ic`	Typical circuit index values for the coils
`kA`	Initial values, in kA, of the conductor currents for all coils
`ngp`	Gridding parameter used by coil diagnostics routines

name and the date-of-issue for the configuration. These quantities get assigned to Corsica variables `device_name` and `device_date`. Neither the device name or issue date may contain space characters. The `config` keyword is followed by a long string containing descriptive information about the configuration; assigned to variable `device_config`. The three device identification variables are not saved in equilibrium save-files, but in a separate portable database file named *`device`*`_id.pfb` where *`device`* is the lowercase device name.

The `current` keyword defines the *signed* nominal plasma current. The remaining three keywords (`ic`, `kA` and `ngp`) are used to initialize the coil circuit index array, initialize conductor currents and the coil gridding parameter array, respectively, and are therefore each followed by N_c data elements.

The circuit indexes get assigned to Corsica array `ic`. The conductor currents get translated to total coil currents (NI) in array cc, where values of zero current are installed as 10^{-10} MAt in Corsica. The coil griding parameters get assigned to Corsica array `ngp` and are used to determine the number of grid points used in the evaluation of superconducting coil diagnostics [4]. To bypass diagnostics for a coil, set its `ngp` entry to zero.

4.4.2 Sample params-file for ITER

```
# ITER Configuration Parameters

device  ITER  2010-07-30
config  See workbook ’ITER_data_2010-v3.3.xls’ issued by Yuri Gribov
current 15

# Coil initialization...
#    PF1  PF2  PF3  PF4  PF5  PF6 CS3L CS2L CS1L CS1U CS2U CS3U VSU VSL ECU ECU ECM ECM ECL ECL TFC
ic     1    2    3    4    5    6    7    8    9    9   10   11   0   0   0   0   0   0   0   0   0
kA    10   10   10   10   10   10   10   10   10   10   10   10   0   0   0   0   0   0   0   0 -49
ngp    6    6    6    6    6    8    8    8    8    8    8    8   0   0   0   0   0   0   0   0   0
```

4.4.3 Comments on the sample params-file

In the sample file we see the PF and CS coils are initialized with conductor currents of 10 kA (the sign is unimportant here), the in-vessel VS and ELM control coils are initialized with zero current, and the TF coil busbar current (with its corresponding `ic` value set to zero) is initialized to a constant -49 kA which is consistent with ITER toroidal field of -5.3 T (see Section 4.7). You can initialize a current to zero in the params-file, but a value of 1×10^{-10} MAt will be imposed on any zero entries in array `cc` since Corsica requires all coil currents be non-zero.

4.5 Passive-file description

The specifications for passive structure are contained in a *passive-file* (default file-name: `passive.in`), which is read by the `read_passive` routine.

4.5.1 Passive-file contents

Passive structure is modeled with the same type of rectangular or parallelogram elements used for driven coils in Corsica in order to represent a conducting structure with finite thickness. By convention, the specifications are derived from a set of *center-line end-point* coordinates and the passive conductor element is centered at the midpoint between the two given end-points.

Each data record consists of up to five elements, where the first two are mandatory and the last three take the value from the previous record if they are omitted. The points must be ordered in a *clockwise* sense.

1. R-coordinate of the end-point of a segment;
2. Z-coordinate of the end-point of a segment;
3. Segment thickness, t, (optional)
4. Segment resistivity [$\mu\Omega$m] (optional), stored in array `rhwires`; and
5. Component name (optional), stored in array `idwires`.

The `read_passive` routine initializes the passive structure model by calling the Corsica `psm` routine which will also display the resistance values for each component (elements with the same component name).

Adjacent end-point coordinates (R_i, Z_i), (R_{i+1}, Z_{i+1}) and thickness, t, are used to generate parallelogram geometric specifications which are stored in arrays `rwires`, `zwires`, `drwires`, `dzwires`, `awires` and `awires2`.

4.5.2 Sample passive-file for ITER

The sample passive structure device configuration file for ITER is shown below, where several records have been omitted for the sake of brevity.

```
# ITER Passive Structure Specifications

# Issued: 2010-04-01

#  R        Z        thk    resis   Id
  3.5398   0.0282   0.060   0.800   VVinner
  3.5398   0.2254
  3.5398   0.4225
  ...
  3.5398  -0.3661
  3.5398  -0.1689
  3.5398   0.0282   0.060   0.800   VVinner
  3.2624   0.0297   0.060   0.800   VVouter
  3.2624   0.2271
  3.2624   0.4244
  ...
  3.2624  -0.1676
  3.2624   0.0297   0.060   0.800   VVouter
  7.5765  -2.6080   0.060   0.800   Ring-SS
  6.8328  -3.1841   0.060   0.800   Ring-SS
  6.7877  -3.1906   0.003   0.027   Ring-Cu
  7.5572  -2.5831   0.003   0.027   Ring-Cu
  3.6895  -3.3207   0.080   0.900   DIR
  3.6895  -2.6698   0.080   0.900   DIR
```

4.5.3 Comments on sample passive-file

The ITER passive structure model consists of inner and outer vacuum vessel walls, an outboard copper/stainless-steel "triangular support" which acts as a vertical stabilizer, and an inboard divertor rail. Several passive elements (cryostat, thermal shield, etc.) are present in ITER and are important when analyzing conditions at plasma formation, but they are not included in the Corsica model.

4.6 Shape-file description

The reference plasma shape, or target separatrix, is specified with a set of R, Z coordinates in a *shape-file* with default name `shape.in` and is read by the `read_shape` routine.

4.6.1 Shape-file contents

The data records are a set of ordered R-Z coordinates which are stored in the Corsica "fuzzy boundary" arrays (`rfbd` and `zfbd`). The points must be ordered in a CCW sense, starting from the inboard strike-point, crossing the outboard strike-line at the x-point, and continuing in a CCW sense around to the outboard strike-point. The points should conform to a smooth contour and be uniformly spaced for proper weighting.

The array of weight factors, α_{fbd} (array `alfbd`), applied to the fuzzy boundary points which are currently in memory will not be altered, but if the shape-file contains more points than the present equilibrium in memory, any elements of α_{fbd} that are zero will be set to one.

4.6.2 Sample shape-file for ITER

The sample shape-file for ITER is given below, with several records omitted.

```
# ITER Target Separatrix

# Issued: 2010-01-14, reodered CCW

#  R [m]     Z [m]
  4.2266   -3.7900
  4.2488   -3.7801
  4.3463   -3.7372
 ...
  8.1802    0.1124
  8.1672    0.0149
  8.1512   -0.0729
  8.1493   -0.0827
 ...
  5.5083   -4.2778
  5.5171   -4.2976
  5.5513   -4.3754
```

4.7 TFCoil-file description

Toroidal field coil specifications are defined in a *tfcoil-file* with default name `tfcoil.in`; read by the `read_tfcoil` routine. The toroidal field coil model in Corsica only requires the specification of the vacuum toroidal field $B_{\varphi,vac}$ at the center of the computational grid (Corsica variables `btor` and `ro`), which exist in all save-files. We need to include toroidal field specifications in input files since this information is needed to create initial magnetization states in Corsica where we start-up the code *without* a save-file.

4.7.1 TFCoil-file contents

The first data record in a tfcoil-file consists of three elements:

1. number of TF coils, N_{TFC},
2. vacuum toroidal field at R_{ref}, $B_{\varphi,vac}$ [T], and
3. reference radius, R_{ref} [m].

These parameters are optionally followed by three sets of R-Z coordinates, each set followed by an integer type-code (0, 1 or 2) which signify that the coordinates define (0) the center-line of the TF conductor, (1) the inner-periphery or 2 the outer-periphery. These quantities are used by the `layout` graphics routine to show the the coil geometry and by an auxiliary script routine to evaluate out-of-plane forces on the TF coils.

The TF coil geometric specifications are not saved in equilibrium save-files, but in a separate portable database file named `device_tfc.pfb` where `device` is the lowercase device name as defined in the params-file (see Section 4.4). The PFB file is written by `read_tfcoil`. Load the TF coil information in each session where it is needed, by executing

```
restore device_name_tfc.pfb
```

4.7.2 Sample tfcoil-file for ITER

The sample tfcoil-file is listed below, with several lines omitted.

```
# ITER TF Coil Configuration

# Issued: 2001-01-17 (ITER/FEAT)

  18  -5.3  6.2  # No. coils, Btor, R0

#   R [m]    Z [m]
 2.6543   0.0359   0
 2.6543   0.2059   0
 2.6543   0.3758   0
 ...
 2.6543  -0.3758   0
 2.6543  -0.2059   0
 2.6543  -0.0359   0
 2.991    0.000    1
 2.991    0.513    1
 2.991    1.009    1
 ...
 2.991   -0.975    1
 2.991   -0.479    1
 2.991    0.000    1
 2.387    0.000    2
 2.387    0.513    2
 2.387    1.009    2
 ...
 2.387   -0.975    2
 2.387   -0.479    2
 2.387    0.000    2
```

4.8 Wall-file description

Coordinates of the plasma-facing elements of the first-wall, divertor and limiter are contained in a *wall-file*, with default name `wall.in`, read by the `read_wall` routine. Data records consist of just three items:

1. radial coordinate of a point on the wall, stored in array `rplate`;
2. axial coordinate of a point on the wall, stored in array `zplate`;
3. wall-type code: 0 for first-wall, 1 for limiter and 2 for divertor

The `read_wall` routine also sets the number of plate elements `nplates`.

Points may be ordered in either in a CW or CCW sense, but the ordering must be consistent within the file. Adjacent points define a *wall segment* and are installed as "plate" segment end-points in the two-dimensional `rplate` and `zplate` arrays. The region between the end of one wall-type and the beginning of the next wall-type is interpreted as a gap; if a gap is not desired then the beginning and ending coordinates of adjacent components must be identical.

4.8.1 Sample wall-file for ITER

The sample wall-file for ITER is listed below, with several lines omitted.

```
# ITER Wall Geometry

# Date-of-issue: 2010-12-14

# Type 0:first-wall, 1:limiter, 2:divertor
#       R          Z  Type
  6.267    -3.046     0
  7.283    -2.257     0
  7.899    -1.342     0
  ...
  4.067    -0.484     0
  4.097    -1.500     0
  4.178    -2.506     0
  3.9579   -2.5384    2
  4.0034   -2.5384    2
  4.1742   -2.5674    2
  ...
  5.9821   -3.2822    2
  6.1710   -3.2350    2
  6.3655   -3.2446    2
  3.9579   -2.5384    2
```

5 Reading device-files

As discussed in Section 2, simply execute the `read_device` routine to read all of the device configuration files with their with default file-names:

```
call read_device
```

Each device configuration file has its own reader and each accepts an optional file-name argument, so they may be invoked independently[4] as listed below:

```
call read_coils(file-name, f_R)
call read_params(file-name)
call read_dgaps(file-name)
call read_limits(file-name)
call read_passive(file-name)
call read_shape(file-name)
call read_tfcoil(file-name)
call read_wall(file-name)
```

The *`file-name`* argument defaults to the file-type followed by suffix "`.in`". The coil reader is unique in that it accepts an optional 2nd argument as described below in Section 5.1. Each reader routine will display a brief help message if the first argument is the string "`help`".

[4] The coils reader must be executed first, followed by the params reader; the remaining device readers may be invoked in any order.

Not all device-files need be present when `read_device` is executed, except for the params-file, which is mandatory.

Figure 2 (page 28) shows the ITER configuration created from the sample device-files of Section 4. The figure was produced with the Corsica `layout` routine.

5.1 Reading the coils-file

The coil reader has an optional 2nd argument:

```
call read_coils(file-name, f_R)
```

where f_R is a fraction used to discretize the coil section into filamentary currents used for coil-plasma Green's function evaluation. The default value of f_R is 2.5×10^{-2}. The number of coil filaments, $n_{\Delta R_c} \times n_{\Delta Z_c}$, for each coil are determined as follows. The smaller of the cross-sectional dimensions of the winding pack for each coil, ΔS_i, is determined:

$$\Delta S_i = \min(\Delta R_{c,i}, \Delta Z_{c,i})$$

The number of filaments, n_i, across the smaller dimension of the coil cross-section is determined for each coil by the integer value of

$$n_i = \min\left(1, \frac{\Delta S_i}{f_R R_{c,max}} + \frac{1}{2}\right)$$

where $R_{c,max}$ is the largest coil radius in the entire coil set. The number of filaments across each dimension of the winding pack for each coil are then the integer values of:

$$n_{\Delta R_c,i} = n_i \left(\frac{\Delta R_{c,i}}{\Delta S_i}\right) + \frac{1}{2}$$

$$n_{\Delta Z_c,i} = n_i \left(\frac{\Delta Z_{c,i}}{\Delta S_i}\right) + \frac{1}{2}$$

Sensitivity studies of the degree of coil discretization should always be conducted for a new configuration, adjusting the values of $n_{\Delta R_c}$ and $n_{\Delta Z_c}$ (code variables `nrc` and `nzc`) as necessary to achieve the desired accuracy. Adjustments to `nrc` and `nzc` can be made directly or for all coils systematically by invoking the coils-reader and changing the value of the f_R argument.

References

[1] Lang Lao, EFIT Equilibrium Fitting Code, General Atomics, San Diego, CA

[2] R. H. Bulmer, Modeling ITER Resistive Coils in Corsica, *Informal report*, 2010-09-11.

[3] S. J. Sackett, "EFFI – A Code for Calculating the Electromagnetic Field, Force and Inductance in Coil Systems of Arbitrary Geometry", Report URCL-52402, Lawrence Livermore National Laboratory, 1978.

[4] R. H. Bulmer, Evaluation of the Superconductor Limit-Line Criterion in Corsica, *Informal report*, 2010-05-07

[5] Yuri Gribov, ITER Organization, "Data_for_study_of_ITER_plasma_magnetic_c_33NHXN_v3.4.xls", *Workbook*, issued 2010-12-14.

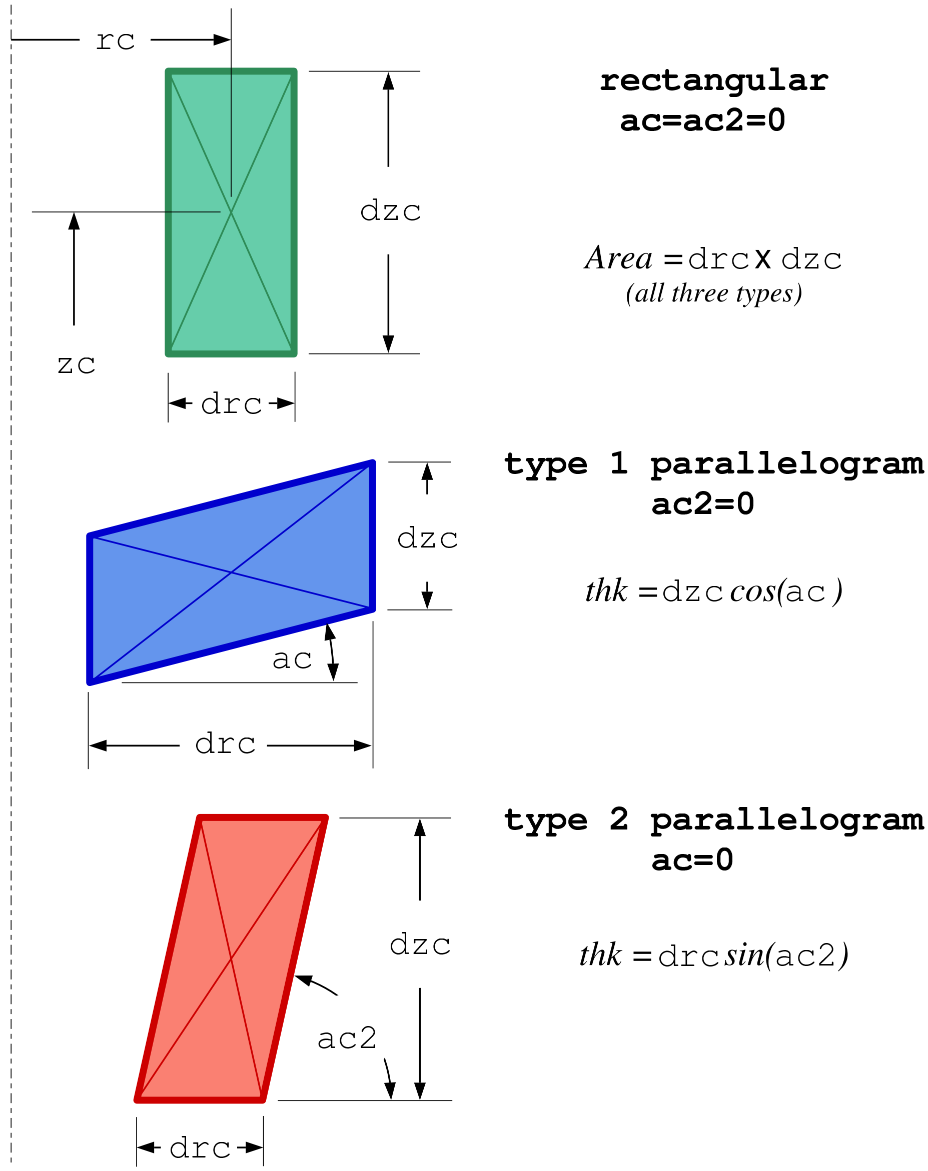

Figure 1: Corsica PF coil parameters; coils may be rectangular (or square), or have one of two types of parallelogram cross-sections. Passive structure elements in Corsica use the same geometric model.

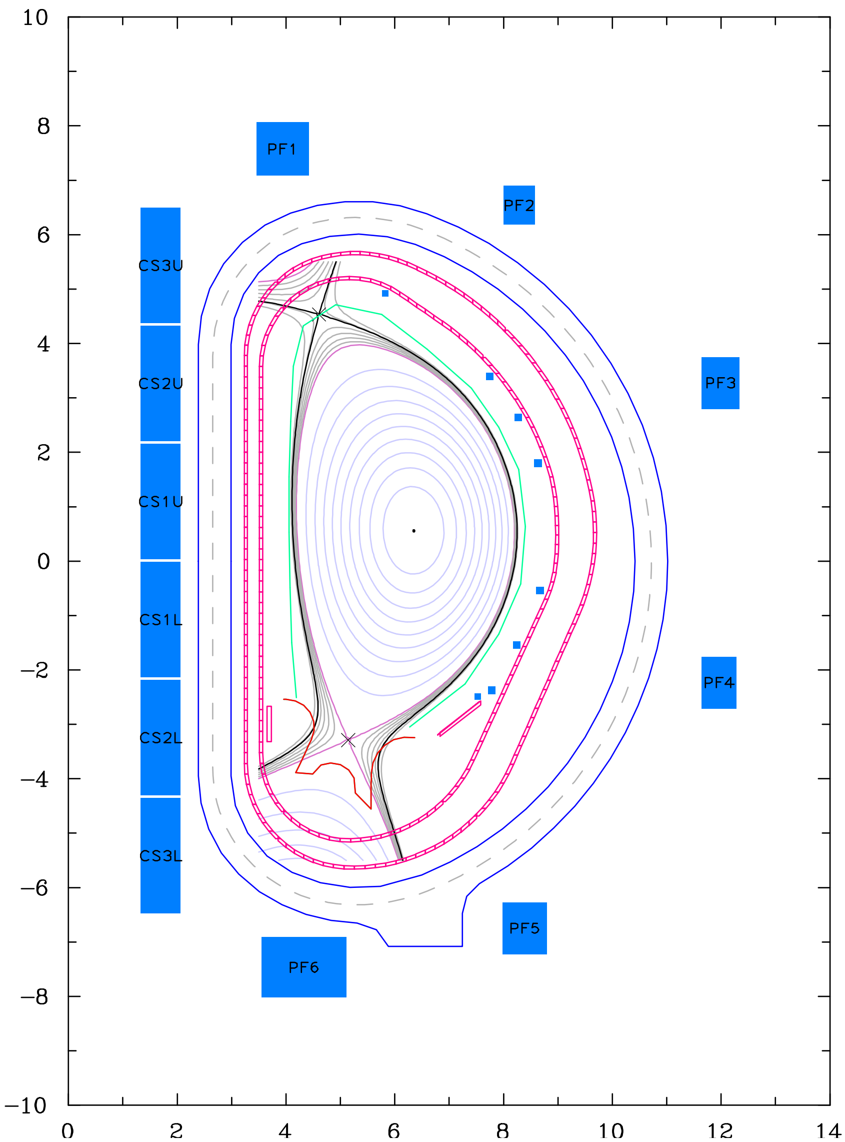

Figure 2: Corsica model for ITER showing the PF and TF coils, passive structure (consisting of the double-walled vacuum vessel, SS/Cu triangular support and divertor inboard rail, in red), first-wall (green), divertor surface (orange) and the reference plasma shape. VS and EC coil names have been omitted from the image so as not to obscure the small cross-section coils.