Advanced model options in extended grids version of SOLPS-ITER

W. Dekeyser[1], W. Van Uytven[1], S. Carli[1], N. Horsten[1], S. Van den Kerkhof[1], M. Blommaert[1], and M. Baelmans[1]

[1]KU Leuven, Department of Mechanical Engineering, Leuven, Belgium

March 24, 2022

1 Introduction

Brief documentation of the additional model options in the extended grids version of SOLPS-ITER. Focus of this document is to provide support in setting up the input files to use the new model options. For details on the models themselves, see following references:

- Extended grids version of SOLPS-ITER [1, 2]
- Advanced Fluid Neutral (AFN) models [3, 10, 11]
- Hybrid fluid-kinetic neutral models [12, 13]
- Improved anomalous transport models [16, 18–20]
- Adjoint-based parameter estimation and optimization [14, 15, 17]

At the moment, this is a working document that is expected to evolve. In case you detect errors, problems, inconsistencies: please contact us so we can make amendments (wouter.dekeyser@kuleuven.be; stefano.carli@kuleuven.be; niels.horsten@kuleuven.be; wim.vanuytven@kuleuven.be)!

2 Converting and running the extended grids version of SOLPS-ITER

2.1 Compiling the extended grids code

The extended grids version of SOLPS-ITER can be obtained by executing the script `solps-iter_update_wg_release` or, in alternative, by checking out the branch `feature/wg-release` from the SOLPS-ITER, B2.5 and EIRENE repositories at ITER. This is version 3.2.0. Compilation follows the regular procedure. Note that at the moment, only compilation on the ITER cluster and TOK cluster at IPP has been attempted. Config files for other machines still have to be adapted – initially this will be done upon user request. At the moment, we recommend to compile only the B2.5 or B2.5-Eirene programs (not the remainder of the SOLPS-ITER package), i.e.

```
make b25
make b25eirene
make b25eirene_mpi
```

2.2 Conversion of existing cases

The new code requires a different format of input files such as `b2fgmtry`, `fort.3[3-5]`, `b2fstati`, `b2.boundary.parameters`, `b2.neutrals.parameters`, `b2.feedback_control.parameters`, `input.dat`. These can be obtained by converting an existing case with the program `b2us`. Note that the extended grids version has branched off of SOLPS-ITER version 3.0.6. Conversion of `b2fgmtry` input files more recent than this version is not supported yet, but will be possible in the (near) future. Conversion of `b2fstati` files up to version 3.0.7 has been succesfully done.
The suggested conversion procedure is:

1 Create a directory `convert` parallel to `baserun`.

2 Copy in such directory all the files from the `run` directory to be converted. Alternatively copy in there the original `b2fgmtry`, `fort.3[3-5]`, `b2fstati`, `b2.boundary.parameters`, `b2.neutrals.parameters`, `b2.feedback_control.parameters`, `input.dat` and `b2mn.dat`.

3 Perform the conversion by executing `b2run b2us` in the directory `convert`.

4 Upon successful conversion, new files in the unstructured format will be available. Create a new `run` directory and import these new files with the script `import_files_from_convert_wg`. Alternatively these can be manually copied into `baserun` or the new `run` directory to substitute the original files

- `b2fgmtry_us` → `b2fgmtry`
- `triangle.coordinates` → `fort.33`
- `triangle.elements` → `fort.34`
- `triangle.neighbors` → `fort.35`
- `b2fstati_us` → `b2fstati`
- `b2us.boundary.parameters` → `b2.boundary.parameters`
- `b2us.neutrals.parameters` → `b2.neutrals.parameters`
- `b2us.feedback_control.parameters` → `b2.feedback_control.parameters`
- `input.dat.us` → `input.dat`

An additional file `b2us.regions.parameters` is also created, in which the user can find the relation between the old (structured) x-directed and y-directed face regions (previously stored in `region(ix,iy,1:2)`) and the new (unstructured) face regions, now stored in the mapping array `fcReg`. Note that volume regions are now stored in the mapping array `cvReg` and do not change numbering convention compared to a structured case.

In the new code format there is no use of variables and switches like `jxa, jxi` for defining the outer midplane. Therefore, the user must provide a line segment which intersects the B2.5 grid, where all control volumes intersected by this segment will be part of the outer midplane. This line segment is defined by two pairs of R-Z coordinates in `b2.user.parameters` using the variable `rzomp`, such as:
`rzomp(1,1) = 0.5,` R-coordinate of first segment point (m)
`rzomp(1,2) = 0.8,` R-coordinate of second segment point (m)
`rzomp(2,1) = 0.0,` Z-coordinate of first segment point (m)
`rzomp(2,2) = 0.0,` Z-coordinate of second segment point (m)
A similar argument holds for the inner midplane, which can be defined in the same way with the variable `rzimp`.

2.3 Important switches and parameters

For the development of the extended grids version of SOLPS-ITER, only the default plasma model based on SOLPS5.2 [21,22] has been converted. This implies in particular the following switches:

- `'b2mndt_style' '1'` (obsolete)
- `'b2tfnb_drift_style' '1'`: drifts are now always defined on cell faces

Note that several other switches have become obsolete, because older code options not compatible with the unstructured data format of the solver have been removed. These should now be removed from the input file. In most cases the code will return an error if an obsolete/illegal switch is used, providing information of the implied value of the switch from the original, structured code.
Other suggested switches at this time:

- `'b2sigp_style' '1'`
- `'*_mdf' '1'` in case of drifts
- `'b2trcl_conductive_limit' '1'` for flux limits (`'*_lim_flux' '1'` option not available anymore)

In order to start a case with 9-point stencil (default in the new code version), use

- `'b2mndr_use_9pt_stencil' '1'`

This is essential for simulations with fluid neutrals. 9-point stencil simulations may be very sensitive to grid quality. Avoid rapidly varying cell sizes, excessively small cells, and grid line bunching as much as possible (typically for example around the 'baffles' in strongly shaped divertors). To continue a case with the 9-point stencil that was originally performed with 5-point stencil, it may be necessary to reduce time step and under-relaxation factors, at least for the first iterations, e.g. set `'b2npco_rxg', 'b2npmo_rxg', 'b2npht_rxg' '1.0e-1'`, and reduce time step with e.g. an order of magnitude.

2.4 Backwards compatibility

Running the same case on the structured and extended grid code version can result in small discrepancies, even when using the same grid (and 5-point stencil). These discrepancies have been investigated in detail and were found to be mainly caused by the numerical treatment of various fluxes, boundary conditions and transport coefficients. Also some bugs in the original code have been identified and corrected in the process. For equivalent input files, the simulation results should thus be equivalent up to discretization effects.

3 Recommended boundary conditions for extended grids

General note: not all boundary conditions have been converted to the new, unstructured format. At the moment, the code will return an error message if a non-converted BC is selected. If this is the case, please contact the developers.
Regarding sheath conditions, the following set of consistent BCs is recommended:

Continuity equation The following generalized BC for the particle flux is implemented at a sheath boundary (`BCCON=14`):

$$\mathbf{\Gamma} \cdot \boldsymbol{\nu} = |\mathbf{b} \cdot \boldsymbol{\nu}|\, n_a c_{\mathrm{s}} + \frac{D}{\lambda} n_a \tag{1}$$

with $\boldsymbol{\nu}$ defined as the outward normal at the boundary (i.e. pointing away from the plasma), and $\mathbf{b}$ the unit vector along the magnetic field direction. The first term is the usual 'sound speed flux'. The second term is an additional anomalous piece. The decay length λ is an input parameter `CONPAR(,,2)`. If `CONPAR(,,2)=0`, the second term is not present (recommended for non-extended cases).

Momentum equation The generalized sheath boundary condition states that the velocity component normal to the wall is equal to pitch times sound speed (`BCMOM=13`):

$$\mathbf{V}_a \cdot \boldsymbol{\nu} = |\mathbf{b} \cdot \boldsymbol{\nu}|\, c_\mathrm{s} \tag{2}$$

In the absence of drifts, this becomes

$$u_{||} \mathbf{b} \cdot \boldsymbol{\nu} = |\mathbf{b} \cdot \boldsymbol{\nu}|\, c_\mathrm{s} \tag{3}$$

If drifts are included, this becomes

$$u_{||} \mathbf{b} \cdot \boldsymbol{\nu} + \mathbf{V}_a^{E\times B} \cdot \boldsymbol{\nu} = (u_{||} \mathbf{b} + V_{a,\theta}^{E\times B} \mathbf{e}_\theta + V_{a,r}^{E\times B} \mathbf{e}_r) \cdot \boldsymbol{\nu} = |\mathbf{b} \cdot \boldsymbol{\nu}|\, c_\mathrm{s} \tag{4}$$

(In a possible further generalization, also the grad-B drift could be included here.)
To elaborate this expression further, we need some equalities:

$$\begin{aligned}
\cos\alpha &= \mathbf{e}_\theta \cdot \boldsymbol{\nu}_\mathrm{f} = \mathbf{e}_r \cdot \mathbf{e}_y && (5)\\
\sin\alpha &= \mathbf{e}_r \cdot \boldsymbol{\nu}_\mathrm{f} = -\mathbf{e}_\theta \cdot \mathbf{e}_y && (6)\\
\boldsymbol{\nu} &= \pm \boldsymbol{\nu}_\mathrm{f} && (7)\\
\mathbf{b} \cdot \boldsymbol{\nu} &= (\mathbf{b} \cdot \mathbf{e}_\theta)(\mathbf{e}_\theta \cdot \boldsymbol{\nu}) = |B_\theta/B|\, \mathbf{e}_\theta \cdot \boldsymbol{\nu} && (8)
\end{aligned}$$

Here, $\mathbf{e}_\theta$ is the unit vector along the poloidal projection of the magnetic field, $\mathbf{e}_r$ is normal to $\mathbf{e}_\theta$ in the poloidal plane forming a right-handed system $\{\theta, r\}$, and $\boldsymbol{\nu}_\mathrm{f}$ is the normal to the local cell face, pointing in the ∇y direction of the local coordinate system. The difference between $\boldsymbol{\nu}$ and $\boldsymbol{\nu}_\mathrm{f}$ is encoded in `bcFcOr`. Typically: $\boldsymbol{\nu} = \boldsymbol{\nu}_\mathrm{f}$ for what used to be 'N' and 'E' boundaries (`bcFcOr = 1.0`), and $\boldsymbol{\nu} = -\boldsymbol{\nu}_\mathrm{f}$ for what used to be 'S' and 'W' boundaries (`bcFcOr = -1.0`).
Elaborating eq. (4) gives

$$\pm u_{||}\, |B_\theta/B| \cos\alpha \pm V_{a,\theta}^{E\times B} \cos\alpha \pm V_{a,r}^{E\times B} \sin\alpha = |B_\theta/B|\, |\cos\alpha|\, c_\mathrm{s} \tag{9}$$

At small incidence angles of the magnetic field (or large $E \times B$ drifts), this condition breaks down. For tangential boundaries ($\mathbf{b} \cdot \boldsymbol{\nu} = 0$), the condition states that the component of the $E \times B$ velocity normal to the boundary (i.e. the radial component) must be zero, which actually implies the condition that the vertex values of the potential must be equal at these boundaries. Further theoretical development is needed to clarify which condition should be applied in these cases. Therefore, the current implementation simply changes the BC to a leakage type BC for momentum if $|\cos\alpha| < 10^{-3}$:

$$\boldsymbol{\Gamma}^\mathrm{m} \cdot \boldsymbol{\nu} = -c^\mathrm{m} \frac{D}{\lambda} m_a n_a u_{||} \tag{10}$$

with c^m an input paramter `MOMPAR(,,2)`, and λ equal to `CONPAR(,,2)` as above. If $c^\mathrm{m} = 1$, this behaves as zero normal gradient condition for the parallel velocity. If needed this correction can be disabled by setting `'b2stbc_Qalfmin' '0.0'` (default: 10^{-3}) to ensure converted structured cases are not affected.

Energy equations Here we need to consider the energy flow in the parallel and perpendicular directions. Following the generalized particle flux to the boundary, the following ion energy condition is implemented for the ion heat flux $\mathbf{Q}_\mathrm{i}$ (`BCENI=15`):

$$\mathbf{Q}_\mathrm{i} \cdot \boldsymbol{\nu} = \delta_{\mathrm{i},1} \left|\mathbf{b} \cdot \boldsymbol{\nu}\right| n_a c_\mathrm{s} T_\mathrm{i} + \delta_{\mathrm{i},2} \frac{D}{\lambda} n_a T_\mathrm{i} \tag{11}$$

with $\delta_{\mathrm{i},1}$ the usual sheath transmission coefficient for ions ($\sim$1.5) and $\delta_{\mathrm{i},2}$ a transmission coefficient for the anomalous piece ($\sim$2.5) as input parameters `ENIPAR(,1)` and `ENIPAR(,2)`.
For the electron energy equation, following condition is implemented (`BCENE=15`)::

$$\mathbf{Q}_\mathrm{e} \cdot \boldsymbol{\nu} = \left(\frac{1+\gamma_\mathrm{e}}{1-\gamma_\mathrm{e}} T_\mathrm{e} + e\phi\right) \max\left(0, \left|\mathbf{b} \cdot \boldsymbol{\nu}\right| n_a c_\mathrm{s} - \frac{j_{||}}{e} \mathbf{b} \cdot \boldsymbol{\nu}\right) + \left(\delta_{\mathrm{e},2} T_\mathrm{e} + e\phi\right) \frac{D}{\lambda} n_a \tag{12}$$

with $\delta_{\mathrm{e},2}$ a sheath transmission factor for the anomalous piece (`ENEPAR(,2)`).

Potential equation The boundary condition for the potential equation is not modified compared to original code (`BCPOT=11`):

$$\mathbf{j} \cdot \boldsymbol{\nu} = e \left(\left|\mathbf{b} \cdot \boldsymbol{\nu}\right| n_a c_\mathrm{s} - (1-\gamma_\mathrm{e}) \left|\mathbf{b} \cdot \boldsymbol{\nu}\right| n_\mathrm{e} \frac{1}{\sqrt{2\pi}} \sqrt{\frac{T_\mathrm{e}}{m_\mathrm{e}}} \exp\left(-\frac{e\phi}{T_\mathrm{e}}\right)\right) \tag{13}$$

3.1 Boundary condition specification in `b2.boundary.parameters`

The newly developed unstructured solver no longer supports the use of 'E', 'W', 'N', or 'S' boundary conditions, as these labels lose their meaning in unified face representation of the new code. Moreover, individual boundaries tend to have complex shapes and cannot be identified any more by individual poloidal or radial cell indices. Instead, the concept of *automatic* boundary condition type 'A' from SOLPS5.0-ITM is carried over. Boundary faces are designated with a label `fcLbl`. A boundary condition can then be assigned to (groups of) boundary faces by specifying the start and end of the `fcLbl` range to which the boundary condition applies, for example

```
NBC = ...
BCCHAR = 'A', 'A', ...
BCSTART = 1, 2, ...
BCEND = 1, 3, ...
```

It is thus up to the user to logically assign face labels to specific (groups of) boundaries during the grid generation step, so that boundaries can later be identified. Moreover, it is recommended (but not strictly necessary) to group all core boundary faces into a single boundary group.
An analogous approach is implemented to specify recycling strata in `b2.neutrals.parameters`.
NOTE: when converting a structured-grid case, face labels will automatically be assigned to the old 'E', 'W', 'N', and 'S' boundaries, and the `b2.boundary.parameters` file converted to correctly account for the new format. Typical numbering for SN case:

- Core: `BCSTART = BCEND = -21`
- West target: `BCSTART = BCEND = -13`
- East target: `BCSTART = BCEND = -34`
- Inner PF: `BCSTART = BCEND = -23`
- Outer PF: `BCSTART = BCEND = -24`
- North wall: `BCSTART = -44, BCEND = -42`

4 Advanced Fluid Neutral models

This section describes the steps needed to activate the Advanced Fluid Neutral (AFN) models in a (fluid neutral) simulation.

4.1 Setup `b2frates`

The AFN models require transport coefficients for the atoms to be computed based on the same reactions used by the kinetic model in EIRENE (charge-exchange / ionization / recombination for atoms). On the B2.5 side, this is achieved by using the correct reactions when creating the `b2frates`-file.
Typical input file `b2ar.dat` to create `b2frates` for use with AFN (D-only case):

```
*tlohi (tlo, thi; real, free format)
1.0E-02 1.0E+04
*nlohi (nlo, nhi; real, free format)
1.0E+12 1.0E+22
*numnuc (nnuc; integer, free format)
1
*nucspec (nz, izlo, izhi; integers, one triplet per line)
1 0 1
*flag ('adpak', 'strahl', 'adas' or 'amds', free format)
'adas'
*tailep (real, free format)
0.08
 'b2ardr_no_smoothing'  '1'        # Eliminate smoothing for consistency with EIRENE
 'b2ardr_rtnt'          '100'      # Rather fine tables to improve convergence may be needed
 'b2ardr_rtnn'          '100'      # Rather fine tables to improve convergence may be needed
 'b2ardr_no_weisheit'   '1'        # No correction of hydrogen neutrals
 'b2ardr_amjhydhel'     '1'        # If 1 ==> Take Amjuel/Hydhel for hydrogen neutrals
 'b2ardr_t_min'         '1.0E-1'   # minimum temp.  for which Amjuel rates are evaluated [eV]
 'b2ardr_t_max'         '1.0E+30'  # maximum temp.  for which Amjuel rates are evaluated [eV]
 'b2ardr_nn_min'        '1.0E+14'  # minimum density for which Amjuel rates are evaluated [m^-3]
 'b2ardr_nn_max'        '1.0E+30'  # maximum density for which Amjuel rates are evaluated [m^-3]
 'b2ardr_fix_recomb'    '1'        # also requires 'b2stel_fix_recomb_energy '1' in b2mn.dat
***Amjuel/Hydhel Hydrogen reaction cards
*Ionization
AMJUEL H.4 2.1.5 EI
AMJUEL H.102.1.5 EI
*Charge-exchange
AMJUEL H.1 3.1.8 CX
*Recombination
AMJUEL H.4 2.1.8 RC
AMJUEL H.102.1.8 RC
```

4.2 Setup main run with b2mn

To use the AFN model, the following switches are required in b2mn.dat:

```
'b2mndr_use_9pt_stencil'     '1'  # correct treatment of grid non-orthogonality
                                  # value 1 also recommended for standard fluid neutral model!
'b2tqna_transport_afn'       '1'  # use transport coefficients of AFN model
'b2stbr_recycle_afn'         '1'  # use AFN recycling model
'b2stel_fix_recomb_energy'   '1'  # needed for consistency with b2frates file of AFN model
'b2mn_tn_style'              '0'  # 0:  shared temperature Ti+Tn; 1:  pure ion temperature Ti (onl
                                  # 2:  separate ion/neutral temperature Ti & Tn
                                  # At the moment, only tn_style 0 supported for multifluid cases
```

4.3 Setup b2.neutrals.parameters

To set the parameters for the AFN recycling model, the following options can be specified in b2.neutrals.parameters

surf_mat(1:NSTRAI)	specify wall material of the stratum available options: 'W', 'Be', 'C'; make consistent with EIRENE side
maxw(1:NSTRAI)	BC based on drifting Maxwellian (1) or diffusion model (0) recommended value 1
accel_ion(1:NSTRAI)	sheath accelaration at this stratum (1) or not (0) typically 1 at E/W (sheath) boundaries and 0 at N/S (void) boundaries
e_fc	Franck Condon dissociation energy (make consistent with EIRENE) usual EIRENE value 3.0

Note that is it recommended to implement 'leakage' conditions for fluid neutrals (e.g. at N/S boundaries) through the recyc-arrays, not through a leakage BC.

4.4 Setup b2.boundary.parameters

For the neutral continuity equation, it is recommended to use BCCON=5 with CONPAR(:,:,1)=CONPAR(:,:,2)=0.0 to impose a zero neutral particle flux at each boundary. The particle flux is determined by the recycling flux as determined by the settings in b2.neutrals.parameters. For the neutral momentum equation, it is recommended to use BCMOM=5 with MOMPAR(:,:,1)=MOMPAR(:,:,2)=0.0 at all boundaries, except at the core boundary. At the core boundary, the momentum flux boundary condition is not valid and it is recommended to impose a zero gradient for the neutral parallel velocity (BCMOM=2 with MOMPAR(:,:,1)=0.0).
Note that is it recommended to implement 'leakage' conditions for fluid neutrals (e.g. at N/S boundaries) through the recyc-arrays, not through a leakage BC.

4.5 Flux limits for neutrals

To avoid unphysically large values of the diffusive neutral fluxes, these are typically flux limited. In the new unstructured code, the following expression is used to flux limit the neutral particle flux:

$$\mathbf{\Gamma}_{\mathrm{n}} = n_{\mathrm{n}} u_{||,\mathrm{n}} \mathbf{b} - \tilde{D}^{p}_{\mathrm{n}} \nabla p_{\mathrm{n}}, \tag{14}$$

with

$$\tilde{D}^{p}_{\mathrm{n}} = \frac{D^{p}_{\mathrm{n}}}{\left(1 + \left(\frac{D^{p}_{\mathrm{n}} ||\nabla p_{\mathrm{n}}||}{\alpha_1 n_{\mathrm{n}} \frac{v_{\mathrm{T}}}{4}}\right)^{\gamma_1}\right)^{1/\gamma_1}}, \qquad v_{\mathrm{T}} = \sqrt{\frac{8 T_{\mathrm{i}}}{\pi m_{\mathrm{n}}}}. \tag{15}$$

Default values for α_1 and γ_1 are 0 and 1. This is the usual expression for flux limits found in the literature. However, in the original, structured version of the code this flux limit is applied to the poloidal and radial components of the flow separately. This means that the poloidal resp. radial

component of the gradient is used in the denominator of expression (15) to compute the flux limit for the poloidal resp. radial flow component, instead of the norm of the gradient vector for both. The result is that the flux limit factor was different for both flow directions, leading to an artificical 'rotation' of the diffusive neutral flux, which would no longer be in the direction of the (negative) neutral pressure gradient. In contrast, the new unstructured solver now applies a single, 'isotropic' flux limit based on the total gradient of the neutral pressure, thereby only modifying the size of the diffusive flow, not its direction. Similar expressions have been implemented to limit the neutral heat conductivity and viscosity:

$$\tilde{\kappa}_\mathrm{n} = \frac{\kappa_\mathrm{n}}{\left(1+\left(\frac{\kappa_\mathrm{n}||\nabla T_\mathrm{i}||}{\alpha_2 n_\mathrm{n} v_\mathrm{h} T_\mathrm{i}}\right)^{\gamma_2}\right)^{1/\gamma_2}}, \qquad v_\mathrm{h} = \sqrt{\frac{2T_\mathrm{i}}{m_\mathrm{n}}}, \tag{16}$$

$$\tilde{\eta}_\mathrm{n} = \frac{\eta_\mathrm{n}}{\left(1+\left(\frac{\eta_\mathrm{n}||\nabla u_{||,\mathrm{n}}||}{\alpha_3 n_\mathrm{n} T_\mathrm{i}}\right)^{\gamma_3}\right)^{1/\gamma_3}}. \tag{17}$$

The flux limits are applied with the following switches:

- `'b2trno_flux_limit_to_dpa' '1'` (default)
- `'b2tlc0_style' '1'` (default)
- `'b2tlc0_alpha'` (default: 0.0 (off); set to 1.0 to turn on)
- `'b2tlc0_gamma'` (1.0 or 2.0 (default))
- `'b2trno_flux_limit_to_vsa' '1'` (default)
- `'b2tlv0_style' '1'` (default)
- `'b2tlv0_alpha'` (default: 0.0 (off); set to 1.0 to turn on)
- `'b2tlv0_gamma'` (1.0 or 2.0 (default))
- `'b2tlh0_style' '1'` (default)
- `'b2tlh0_alpha'` (default: 0.0 (off); set to 1.0 to turn on)
- `'b2tlh0_gamma'` (1.0 or 2.0 (default))

5 Spatially hybrid neutral model

The spatially hybrid fluid-kinetic neutral model consists of a fluid model for the atoms in the plasma grid region coupled to a kinetic model for the atoms sampled at the plasma-void interfaces (originating from ion recycling and fluid atoms entering the void regions) and a kinetic model for the molecules in the entire domain. The plasma-void interfaces are indicated with red dashed lines in Fig. 1. The coupling of the fluid model to a kinetic model allows to accurately model the transport of atoms in regions with zero or low plasma density (light brown regions in Fig. 1), which are regions where the fluid limit is not valid. Also the coupling to a kinetic model for molecules typically strongly improves the accuracy of the results.
Note that at the moment, the spatially hybrid method has only been tested for D-only cases, but should work equally well for multifluid cases. Extensions to treat impurity neutrals fully kinetically at all strata in combination with a spatially hybrid method for only the D atoms are under development. To activate the spatially hybrid approach, some additional adaptations of the input files have to be made on top of the settings for the AFN model, as already described in Section 4.

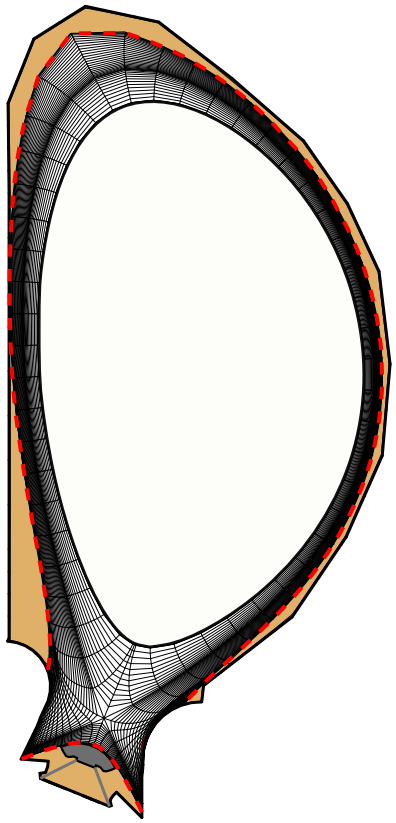

Figure 1: Illustration of a typical mesh for plasma edge transport calculations for the ITER tokamak. The void regions outside the plasma mesh into which neutrals can be transported are indicated in light brown. Figure taken from Ref. [12].

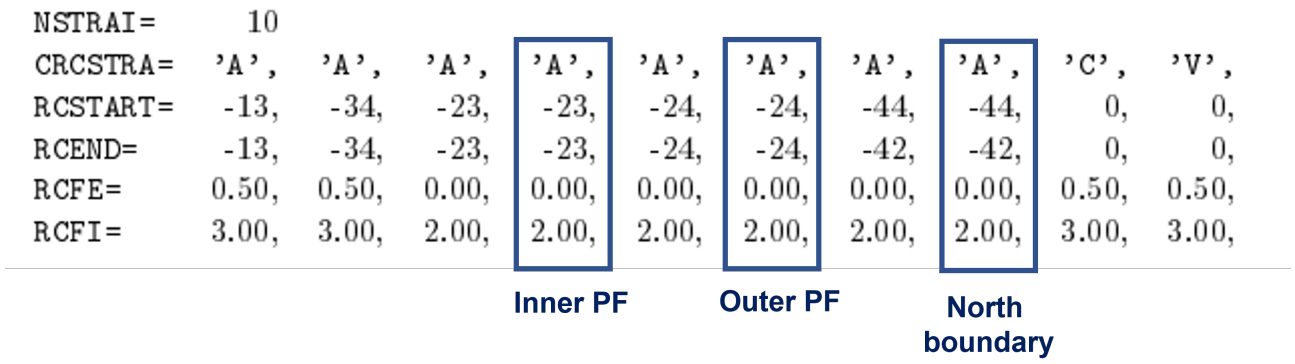

Figure 2: Example of extension of `b2.neutrals.parameters` with additional strata for the spatially hybrid approach.

5.1 Setup `b2mn.dat`

The following options need to be set in `b2mn.dat`:

```
'b2mndr_eirene'                    '1'          # coupling to EIRENE for the kinetic
                                                # part
'b2mndr_rescale_neutrals'          '1.0e+00'    # do not rescale the fluid density
'b2stbr_rescale_neutrals_sources'  '1.0e+00'    # do not rescale the sources from the
                                                # fluid neutrals
'b2mn_spatial_hybrid'              '1'          # turn on the spatially hybrid
                                                # approach
```

5.2 Setup `b2.neutrals.parameters`

At each plasma-void interface, an additional stratum stemming from fluid atoms entering the void regions needs to be defined. For example, a single-null configuration with originally 7 strata (5 surface recycling strata, a gas puff, and volumetric recombination) now consists of 10 strata. The columns due to the 3 additional strata are indicated in Fig. 2.
All other variables in `b2.neutrals.parameters` which have dimensions depending on `NSTRAI` have to be extended for the additional strata.
Some important settings:

recyc(0,1)=	1.00,	1.00,
recyc(0,2)=	1.00,	1.00,
recyc(0,3)=	1.00,	1.00,
recyc(0,4)=	1.00,	1.00,
recyc(0,5)=	1.00,	1.00,
recyc(0,6)=	1.00,	1.00,
recyc(0,7)=	1.00,	1.00,
recyc(0,8)=	1.00,	1.00,
recyc(0,9)=	1.00,	1.00,
recyc(0,10)=	1.00,	1.00,
b2recyc(0,1)=	1.00,	1.00,
b2recyc(0,2)=	1.00,	1.00,
b2recyc(0,3)=	**0.00,**	**0.00,**
b2recyc(0,4)=	**0.00,**	**0.00,**
b2recyc(0,5)=	**0.00,**	**0.00,**
b2recyc(0,6)=	**0.00,**	**0.00,**
b2recyc(0,7)=	**0.00,**	**0.00,**
b2recyc(0,8)=	**0.00,**	**0.00,**
b2recyc(0,9)=	1.00,	1.00,
b2recyc(0,10)=	1.00,	1.00,

Table 1: Typical values for `recyc` and `b2recyc` for the spatially hybrid approach, where `b2recyc` becomes zero at the plasma-void interfaces.

`recyc(0:nsdmax-1,1:NSTRAI)`	Typically 1.0 for each stratum, unless there are surfaces that absorb a fraction of the incident flux. It should be noted that these recycling coefficients are used on the EIRENE side and not on the B2.5 side.
`b2recyc(0:nsdmax-1,1:NSTRAI)`	`b2recyc` has been introduced to determine the recycling coefficients on the B2.5 side and to make it different from the EIRENE side. `b2recyc` is typically 0.0 at the plasma-void interfaces (see Table 1), as recycled ions are typically treated kinetically at these interfaces and the fluid atoms are converted to kinetic atoms. If smaller than zero, then takes the value from `recyc`. (default: -1.0).
`recycm(0:nsdmax-1,1:NSTRAI)`	Fraction of thermally released particles that is emitted as molecules. `recycm=1.0` for cases with molecules; `recycm=0.0` for cases without molecules (e.g., for a simulation with purely fluid neutral model). The thermally released atoms get an energy `e_fc` (see Section 4.3). (default: 0.0)
`mol(1:NSTRAI)`	Determines if a stratum is a molecular stratum in EIRENE (`NLMOL=.TRUE.`). Should be only 1 for the targets in case of a simulation with molecules. (default: 0)
`maxw(1:NSTRAI)`	It is recommended to use `maxw=1`, as the diffusion assumption for the underlying distribution is not used on the EIRENE side.

5.3 Setup `input.dat`

The fluid neutral solution has to be loaded as a new background species. For a pure D case without impurities, input block 5a is shown in Fig. 3.
The strata can either be specified in input block 7 or input block 14 (when `INDSRC=6` in block 7). For

```
** 5a. Bulk ion species
     2
 1 D+        2  1  1  1  1 -1  0  1  0  0  0  0              0
    13   115   111     0 30000
 1.40000E+01 0.00000E+00 0.00000E+00 0.00000E+00 1.00000E+00
 2 D_0       2  1  1  0  1  1  0  0  0
```

Figure 3: Example of block 5a for spatially hybrid approach for a pure D case

SOLPS-ITER users, we advice to use input block 14, because it automatically sets the recommended options. In this document, the description is limited to input block 14 and we discuss the default assumptions. For settings deviating from these defaults, it is necessary to specify them in input block 7, as described in the EIRENE manual.

5.3.1 Setup of input block 14

Firstly, the additional neutral background species has to be added here as well, i.e., `NFLA` has to be increased and an additional line has to be added for the fluid neutral background. Secondly, the additional strata have to be added. E.g., for a pure D single-null case, `NTARGI` is increased from 5 to 8. For each stratum, the following line has to be specified:

```
 I   NDT   NPTC   NPTCM   NSPZI   NSPZE   NEMOD   ITYPE
```

The only new variable here is `ITYPE`, which specifies the type of the species (numbering according to the EIRENE convention for `ITYP`): `ITYPE=0` (default): bulk ion species (`NLPLS=.TRUE.`); `ITYPE=1`: atoms (`NLATM=.TRUE.`); `ITYPE=2`: molecules; `ITYPE=3`: trace ions; `ITYPE=4`: bulk ions; `ITYPE=5`: photons.
Fig. 4 shows an example of input block 14 for a pure D single-null case, including molecules. The following things are important for the molecular strata (`ITYPE=2`):

1. `NSPZI` should be equal to `NSPZE` and should point to the fluid neutral background species (2 in our case).
2. For `NEMOD=0`, the EIRENE variable `NEMODS` is set to 116. This means that the molecules are sampled from a Maxwellian with the temperature given by `SORENI` and the drift velocity determined by `SORVDX`, `SORVDY` and `SORVDZ`. The latter variables are set in infcop: `SORENI=0.1` and `SORVDX=SORVDY=SORVDZ=0.0`, which means that a non-drifting Maxwellian with a temperature of 0.1 eV is assumed.
3. `NSPEZ` is assumed to be 1, which is valid for a case with one hydrogen species. Adaptations to select the correct molecule are still needed for multi-isotope cases.

Items 1 and 3 are also important for the kinetic atoms launched at the plasma-void interfaces (`ITYPE=1`). The atoms are sampled from a Maxwellian distribution with the fluid neutral background properties.

6 Feedback schemes

Feedback schemes are now all applied in the same way in a dedicated module `b2us_feedback`, with a structure similar to the one used in the old `b2.feedback_control.parameters`. Switches in `b2mn.dat` are still available and will override whatever is specified in `b2.feedback_control.parameters`, but it is recommended to move to the new structure and eventually not use switches anymore. This will hopefully allow more flexibility in applying such schemes. Old module `b2mod_feedback` is now used only for converting from structured to unstructured format and routine `b2stbc_fb` is not used anymore.

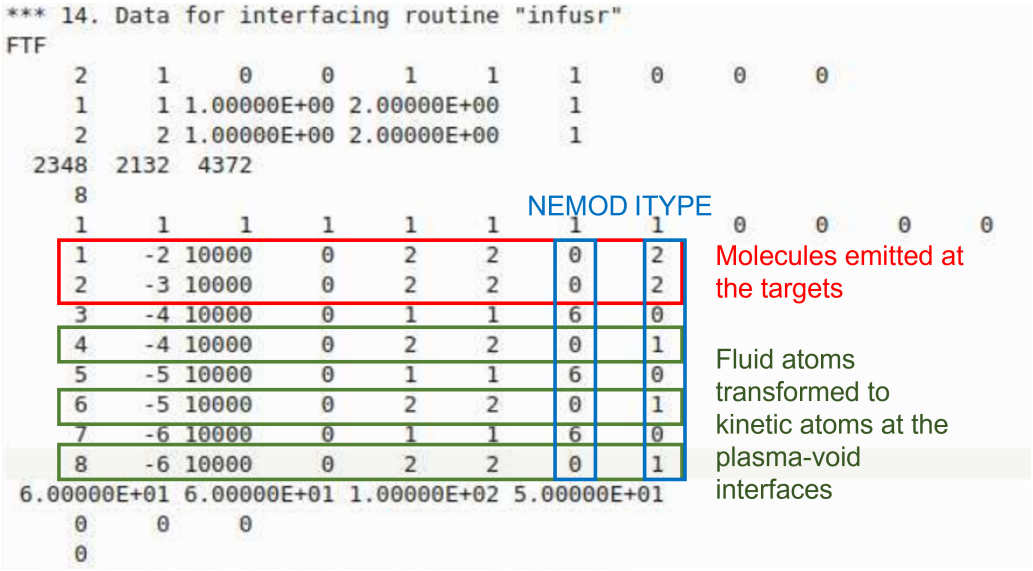

Figure 4: Example of block 14 for spatially hybrid approach for a single-null pure D case

Some of the old feedback switches assumed structured grid formats, for which an exact counterpart in the unstructured code is not available anymore. Therefore, they have been (partially) converted but will not be 100% backward compatible. Additionally, the rescaling of such feedbacks has been arbitrarily converted to `fb_rescaling=1` (the one more similar to the original, see below), to avoid a different rescaling option for each type of feedback. A warning is issued in run.log for these switches.

`CBSNA, CBSHE`, etc. are not used anymore to apply the feedback (i.e. are not the actuator), which is instead directly applied to the boundary condition on `CONPAR, ENEPAR`, etc. This means also that specific feedback boundary conditions like `BCCON=11,12` are not used anymore.

Use of same file `b2.feedback_control.parameters` to specify feedbacks. When converting from structured to unstructured using `b2us` the program will automatically convert the old schemes into new format.

Use of same `b2.feedback_save.parameters` to save the current feedback actuator. Only `NA_FEEDBACK_ACTUATOR` will be saved now, all the other `SAVED_CBSXX` are not used anymore.

Feedbacks types enabled with `#ifdef COMPLICATED` needs further revision. VACUUM communication not yet converted.

6.1 Overview of new feedback schemes

The file `b2.feedback_control.parameters` must now be present to enable feedback schemes. It contains the following variables:

`nFb`	number of feedback schemes applied
`fb_type(nFb)`	selects the controlled quantity (similar to old `na_feedback_choice`). Types 1-8 are the same as the old `na_feedback_choice`, while types 10-16 basically implement the ones defined with switches (see next section for conversion)
`fb_rescale(nFb)`	selects the type of rescaling as old `na_feedback_option`, numbering unchanged
`fb_actuator(nFb)`	selects the type of actuator as old `na_feedback_actuator`. Types 1-3 same as old scheme, type 4-7 implement the ones specific to switches
`fb_target(nFb)`	same as old `na_feedback_target`
`fb_species(nFb)`	specifies the species used to calculate the controlled variable and to which the actuator is applied.
`fb_ib(nFb)`	same as old `na_feedback_ib`
`fb_istra(nFb)`	Eirene strata where the gas puff feedback is applied
`fb_puff_min(nFb)`	same as old `na_feedback_puff_min`
`fb_puff_max(nFb)`	same as old `na_feedback_puff_max`
`fb_overshoot(nFb)`	same as old `na_feedback_overshoot`
`fb_const(nFb)`	same as old `na_feedback_const`
`fb_time(nFb)`	same as old `na_feedback_time`
`fb_alpha(nFb)`	same as old `na_feedback_alpha`
`fb_beta(nFb)`	same as old `na_feedback_beta`

The following arrays are used in place of the old style `na_feedback_i[xy][12]` to generalize domain specification for feedback types 1-2-4-5-8-10-11-12-13 (possibly 6 when converted for fluid neutrals). These arrays are automatically converted from the structured format into the unstructured one using `b2us` or can be directly provided by the user.

`fb_regP(nFb,2)`	pointing for each feedback to the first index in `fb_reg`, and its number of CVs or faces
`fb_reg(nMxFbReg)`	listing for each feedback the cells or faces where controlled quantity is calculated. If different feedbacks use the same domain then this can be defined only ones and simply point to it twice with `fb_regP`
`fb_reg_par(nFb,2)`	If the two arrays above are not defined, then the code will look for this array. For `fb_type` 6-10-11-12-13, `fb_reg_par(ifb,:)` indicates the starting (iFb,1) and ending (iFb,2) face region label over which the controlled quantity is calculated. For `fb_type` 1-2-4-5-8, `fb_reg_par(ifb,:)` indicates the starting (ifB,1) and ending (iFb,2) volume region label over which the controlled quantity is calculated.

Old switches/variables that are still used:

`'b2stbc_feedback'`	same as before
`'b2stbc_isfeedback'`	same as before (overrides `fb_species` when using switches-defined feedbacks)
`'eirene_nesepm_istra'`	same as before (overrides `fb_istra` when using switches-defined feedbacks)
`LFEEDBACK`	same as before

The possible values for variables `fb_type` and `fb_actuator` are listed here:

fb_type	quantity		fb_type	quantity
1	Average species density on specified domain		10	Boundary fna
2	Average plasma density on specified domain		11	Boundary fhe
3	OMP separatrix density		12	Boundary fhi
4	Total particle content on specified domain		13	Boundary fch
5	Total ion content on specified domain		14	Total particle content on full plasma domain
6	Neutral particle flux through core boundary (only kinetic)		15	Volumetric recombination (only kinetic)
7	Average concentration on separatrix			
8	Average concentration on specified domain		16	Pedestal electron density
9	IMP separatrix density			

fb_actuator	quantity		fb_actuator	quantity
1	Particle flux via gas puff BC		5	Ion heat flux BC
2	Rescale density		6	Current BC
3	Charged species particle flux BC		7	Density BC
4	Electron heat flux BC			

NOTE: now for actuator type 1 (gas puff) for fluid neutral cases (or fluid neutral strata) the feedback is applied on CONPAR directly, while for kinetic cases (or kinetic strata) this is applied to USERFLUXPARM, in which case fb_ib is not needed and not checked for consistency.

NOTE: fb_species(nFb) goes with plasma fluid species (0:ns-1) and not atoms, differently from old treatment. In general, controlled quantities are calculated on the whole isonuclear sequence to which fb_species(ifb) belongs to. Then if fb_actuator = 1, it is applied to neutral species through gas puff, if fb_actuator=2 it is applied to the ionized species only, if actuator=3 it is applied only to fb_species. For fb_type=6 and actuator=3 then it is also applied to all ionized species (as old treatment).

6.2 Old feedback switches conversion

The following switches were used to apply feedbacks in a more automatic way than with b2.feedback_control.parame
For these the actuator was coded to be CBSXX arrays and automatically applied to structured regions defined by coreregno, solregno, etc. These are not explicitly backward compatible, therefore we tried to convert these switches in the best way possible. Additionally, each of these used different rescaling, that now are limited to the ones above.

The new code will first read b2.feedback_control.parameters and then ADD or OVERRIDE any additional feedback defined through the switches. If a switch applies the same feedback as in b2.feedback_control.para then the switch has precedence. In this case, all the default values for switches will be used instead of what may be defined in b2.feedback_control.parameters (e.g. if FB_PUFF_MIN= 1.00E+18, and 'b2stbc_nesepm_minpuff' is not present, then the default 0.0 is used).

NOTE: feedbacks of the kind 'b2stbc_[fna-fhe-fhi-fch]ycore' seemed to change the boundary [fna-fhe-fhi-fch] based on the value of those fluxes at the same boundary. It is argued that those switches are irrelevant and should be removed from the code, being simply substituted by an imposed flux boundary condition.

'b2stbc_isfeedback'	overrides fb_species
'eirene_nesepm_istra'	overrides fb_istra
'b2stbc_nesepm_overshoot'	overrides fb_overshoot
'b2stbc_nesepm_minpuff'	overrides fb_puff_min
'b2stbc_nesepm_maxpuff'	overrides fb_puff_max
'b2stbc_nesepm_alpha'	overrides fb_alpha
'b2stbc_coreregno'	These switches need to be manually adjusted as now they indicate the
'b2stbc_coreregn2'	boundary where the feedback is applied. For switches like 'b2stbc_xxxycore' or
'b2stbc_solregno'	'private_gas_puff' the user has to specify in b2.feedback_control.parameter
'b2stbc_pfrregno[12]'	where the feedback is calculated (see b2us.regions.parameters produced at conversion), while coreregno still indicates where it is applied. pfrregno and solreg lose meaning in kinetic cases if the actuator is the gas puff. Coreregn2 and pfrregn are not yet converted/used
'b2stbc_fnaycore'	will apply fb_type=10, fb_rescale=1, fb_actuator=3 on region defined by coreregno.
'b2stbc_fnaycore_alpha'	overrides fb_alpha
'b2stbc_fheycore'	will apply fb_type=11, fb_rescale=1, fb_actuator=4 on region defined by coreregno.
'b2stbc_fheycore_alpha'	overrides fb_alpha
'b2stbc_fhiycore'	will apply fb_type=12, fb_rescale=1, fb_actuator=5 on region defined by coreregno.
'b2stbc_fhiycore_alpha'	overrides fb_alpha
'b2stbc_fchycore'	will apply fb_type=13, fb_rescale=1, fb_actuator=4 on region defined by coreregno.
'b2stbc_fchycore_alpha'	overrides fb_alpha
'b2stbc_ndes'	will apply fb_type=14, fb_rescale=1, fb_actuator=1 on region defined by pfrregno1. WARNING: in the structured code this switch was asking for BCCON=12, so density BC for fluid neutrals feedback but then it switches to gas puff for kinetic neutrals. Now it is ONLY gas puff.
'b2stbc_ndes_sol'	will apply fb_type=14, fb_rescale=1, fb_actuator=1 on region defined by solregno. WARNING: in the structured code this switch was asking for BCCON=12, so density BC for fluid neutrals feedback but then it switches to gas puff for kinetic neutrals. Now it is ONLY gaspuff.
'b2stbc_volrec_sol'	will apply fb_type=15, fb_rescale=1, fb_actuator=1 on region defined by pfrregno1. WARNING: not available yet for fluid neutrals.
'b2stbc_volrec_overshoot'	overrides fb_overshoot
'b2stbc_volrec_alpha'	overrides fb_alpha
'b2stbc_volrec_beta'	overrides fb_beta
'b2stbc_nepedm_sol'	will apply fb_type=16, fb_rescale=1, fb_actuator=1 on region defined by solregno
'b2stbc_iyped'	index in the OMP CV list (defined with rzomp in b2.user.parameters) of the pedestal cell (defaults to icsepomp/2)
'b2stbc_nesepm'	will apply fb_type=3, fb_rescale=1, fb_actuator=3 on region defined by coreregno. WARNING, for fluid neutrals it supposed to change the density BC only for plasma species isfeedback. Now isfeedback means only atoms, therefore it is applied to is_end, the highest charge state
'b2stbc_nesepm_pfr'	will apply fb_type=3, fb_rescale=1, fb_actuator=1 on region defined by pfrregno1
'b2stbc_nesepm_sol'	will apply fb_type=3, fb_rescale=1, fb_actuator=1 on region defined by solregno

Switches not yet converted/not used:
'b2stbc_ncall_feedback'
'b2stbc_cbsnafac'
'b2stbc_nesepm_gamma'
'b2stbc_private_flux_puff'

7 Pre- and postprocessing programs

7.1 b2ag and b2ai

For now, still work with strutured format. After running these programs, convert to unstructured format with b2us.

7.2 b2ar

Reading of b2fgmtry-file by this program has been commented out (not needed by the program). This enables direct creation of b2frates with the unstructured code, independent of the format of b2fgmtry (structured or unstructured).

7.3 b2uf

As usual, b2uf will convert b2fplasma into b2fplasmf file. In addition, for structured grids it will also create b2fstate_st and b2fplasmf_st: state and plasma files in structured format. Note: the b2fplasmf_st file does not contain geometry information (would need to be reconstructed first from the unstructured data)

7.4 b2co

Now works based on unstructured file formats.

7.5 b2plot

At the moment, b2plot has not been modified yet to take into account the unstructured format.

7.6 ${SOLPSTOP}/scripts

Most scripts have not been adapted to the new output formats yet.

7.7 ${SOLPSTOP}/scripts/MatlabPostProcessing

Scripts to post process data in unstructured format with Matlab are available.

8 Code switches

Some obsolete switches

- b2mndt_style: model in unstructured solver is based on b2mndt_style = 1 (SOLPS5.2)
- b2news_guard_flows: faces between guard cells no longer exist in unstructured solver (new treatment corresponds, more or less, to value 2)

Note that some of these switches have become prohibited to force users to remove them from their input files. This is done to avoid possible confusion in the interpretation of future code output, assuming that e.g. a certain switch is still available/active.
List of recommended numerics switches, in particular for truly extended cases:

- `b2tral_mode`: recommended value: 2 for extended cases (default: 1). Can have significant impact on numerical stability.
- `b2tfnb_style_int_vel`: recommended value: 0
- `b2sihs_style_int_vel`: recommended value: 0 if `b2tfnb_style_int_vel = 0`, 1 if `b2tfnb_style_int_vel = 1`. Can have significant impact on code stability. Original code had both set to 0.
- `b2sihs_style_visc`: recommended value: 1 (default, backwards compatible: 0). Consistent treatment viscous heating term with hybrid numerical scheme in momentum equation. Largely removes unphysical ion temperature spikes in low density regions.

9 Discretization of the governing equations on unstructured grids

The equations governing plasma transport in the unstructured B2.5 code follow the model desribed in ref. [22], which has thus far also been the default model in the structured B2.5 code in SOLPS-ITER. The model includes continuity and parallel momentum equations for ions and (atomic) neutral species, and ion and electron internal energy equations. The condition that the divergence of the currents must be zero leads to an equation for the plasma potential. Parallel transport is classical, following Braginskii, while a typical ad-hoc, diffusive approximation is used for the anomalous radial transport. The equations are solved in the 2D poloidal plane, assuming symmetry in the toroidal direction.
To describe the discretization of the governing equations, we base the discussion on a general nonlinear, anisotropic, convection–diffusion equation, following the notation introduced in Ref. [2]:

$$\frac{\partial u}{\partial t} + \nabla \cdot \mathbf{\Gamma} = S. \tag{18}$$

In this equation, u is a transported quantity, $\mathbf{\Gamma}$ the corresponding flux, and S a source term. The flux of a particular quantity follows a convection-diffusion prescription,

$$\mathbf{\Gamma} = \boldsymbol{C}u - D\nabla u. \tag{19}$$

The anisotropy in the equations is most naturally expressed w.r.t. the poloidal and radial directions, which by definition are locally orthogonal to each other. We denote these directions with $\{\theta, r\}$, with respective unit vectors $\mathbf{e}_\theta$ and $\mathbf{e}_r$. In our notation, the poloidal direction points along the poloidal projection of the magnetic field, and the radial direction is orthogonal to it in the poloidal plane, forming a right-handed system with the third unit vector $\mathbf{e}_\phi$ along the toroidal direction pointing out of the page. These unit vectors therefore flip orientation with poloidal field reversal. The coefficient $\boldsymbol{C} = C_\theta \mathbf{e}_\theta + C_r \mathbf{e}_r$ in Eq. (19) is a vector describing the convective flux of u, with components defined in the poloidal and radial directions, and $D = [D_{\theta\theta} D_{\theta r}, D_{r\theta} D_{rr}]$ a diffusive tensor. The cross-diffusivities $D_{\theta r}$ and $D_{r\theta}$ are included because they are representative for the treatment of drift flows. Indeed, because in general drift flows have a form $\sim \nabla v \times \mathbf{B}$, with v some scalar field (e.g. pressure, electric potential, magnetic field strength,...), they have the same structure as a cross-diffusion term with $D_{\theta\theta} = D_{rr} = 0$ and $D_{\theta r} = -D_{r\theta} = D_\mathrm{d}$ [2].
In B2.5, the balance equations (18) are discretized using a finite volume technique, which requires the evaluation of the fluxes (19) across cell faces:

$$\begin{aligned} \mathbf{\Gamma} \cdot \boldsymbol{S} &= \Gamma_\theta S_\theta + \Gamma_r S_r, \\ &= \left(C_\theta u - D_{\theta\theta}\nabla_\theta u - D_{\theta r}\nabla_r u\right) S_\theta + \left(C_r u - D_{r\theta}\nabla_\theta u - D_{rr}\nabla_r u\right) S_r, \end{aligned} \tag{20}$$

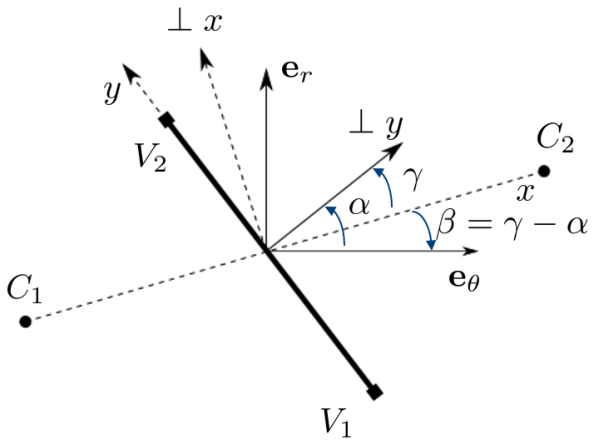

Figure 5: Local coordinate systems defining cell face orientation w.r.t. magnetic field, cell centers, and vertices.

where, $\boldsymbol{S} = S_\theta \mathbf{e}_\theta + S_r \mathbf{e}_r$ is the surface vector of a face, expanded in its poloidal and radial components. In an extended-grids context, a cell face will in general not be aligned with either the poloidal or the radial directions. This is certainly true for the faces in direct contact with the vessel (targets, main chamber, ...), but also already in existing structured grids, because radial faces are usually distorted to match divertor targets. Therefore we consider first a face arbitrarily oriented w.r.t. the magnetic field, see fig. 5. The face under consideration is the thick solid line connecting vertices V_1 and V_2. C_1 and C_2 are the cell centers on either side of the face. The x-direction is defined as the line connecting C_1 and C_2, going through the center of the cell face, and positive from C_1 to C_2. The y-direction is tangent to the face, positive from V_1 to V_2. The local coordinate system defined by x and y is in general not orthogonal. γ is the complement of the angle between the x and y directions. The normal to the face, $\perp y$, is defined such that $\cos\gamma > 0$ (i.e. $-\pi/2 < \gamma < \pi/2$). For a face perpendicular to the connector line we have a locally orthogonal $\{x, y\}$ system, with $\gamma = 0$. Also at the face, we draw the local (orthogonal) poloidal and radial unit vectors. The orientation of the face w.r.t. the local magnetic field is given by the angle α between the poloidal magnetic field and the normal to the face. The angle $\beta = \gamma - \alpha$ between the x direction and the poloidal field direction is introduced for ease of notation below.

With these definitions, the typical faces that appear in structured grids are special instances of this general face description. For example, for a poloidal field clockwise from inner to outer target,1) an orthogonal 'poloidal' face between magnetically rectangular cells has $\alpha = \beta = \gamma = 0$, $\mathbf{e}_x = \mathbf{e}_\theta$, $\mathbf{e}_y = \mathbf{e}_r$, 2) an aligned 'radial' face between magnetically rectangular cells has $\alpha = \pi/2, \beta = -\pi/2, \gamma = 0$, $\mathbf{e}_x = \mathbf{e}_r$, $\mathbf{e}_y = -\mathbf{e}_\theta$, while 3) a 'misaligned' poloidal face has $\alpha = \gamma, \beta = 0$, $\mathbf{e}_x = \mathbf{e}_\theta$ and 4) an aligned radial face between cells with 'misaligned' poloidal faces has $\alpha = \pi/2, \gamma \neq 0$, $\mathbf{e}_y = -\mathbf{e}_\theta$. Hence, our description here is a generalization of the one we presented in ref. [2]. The definition of these face angles for a few typical cell configurations is illustrated in figure 6.

To compute the poloidal and radial components of the gradient required in eq. (20), we relate them to the gradients in the x and y directions, which in turn can be computed directly from differences between cell center (for $\nabla_x u$) and vertex (for $\nabla_y u$) values of u. The desired expression for the components of the gradient can be obtained by expanding the gradient in the orthogonal poloidal–radial system, $\nabla u = \nabla_\theta u\, \mathbf{e}_\theta + \nabla_r u\, \mathbf{e}_r$, and applying the definition of the gradient, $\nabla_\mathrm{d} u = \nabla u \cdot \mathbf{d}$ for an arbitrary direction $\mathbf{d}$, to the unit vectors $\mathbf{e}_x$ and $\mathbf{e}_y$:

$$\nabla_x u = \nabla u \cdot \mathbf{e}_x = \nabla_\theta u\, \mathbf{e}_\theta \cdot \mathbf{e}_x + \nabla_r u\, \mathbf{e}_r \cdot \mathbf{e}_x, \tag{21}$$

$$\nabla_y u = \nabla u \cdot \mathbf{e}_y = \nabla_\theta u\, \mathbf{e}_\theta \cdot \mathbf{e}_y + \nabla_r u\, \mathbf{e}_r \cdot \mathbf{e}_y. \tag{22}$$

From fig. 5 we see that $\mathbf{e}_\theta \cdot \mathbf{e}_x = \cos\beta$, $\mathbf{e}_r \cdot \mathbf{e}_x = -\sin\beta$, $\mathbf{e}_\theta \cdot \mathbf{e}_y = -\sin\alpha$ and $\mathbf{e}_r \cdot \mathbf{e}_y = \cos\alpha$. Inverting

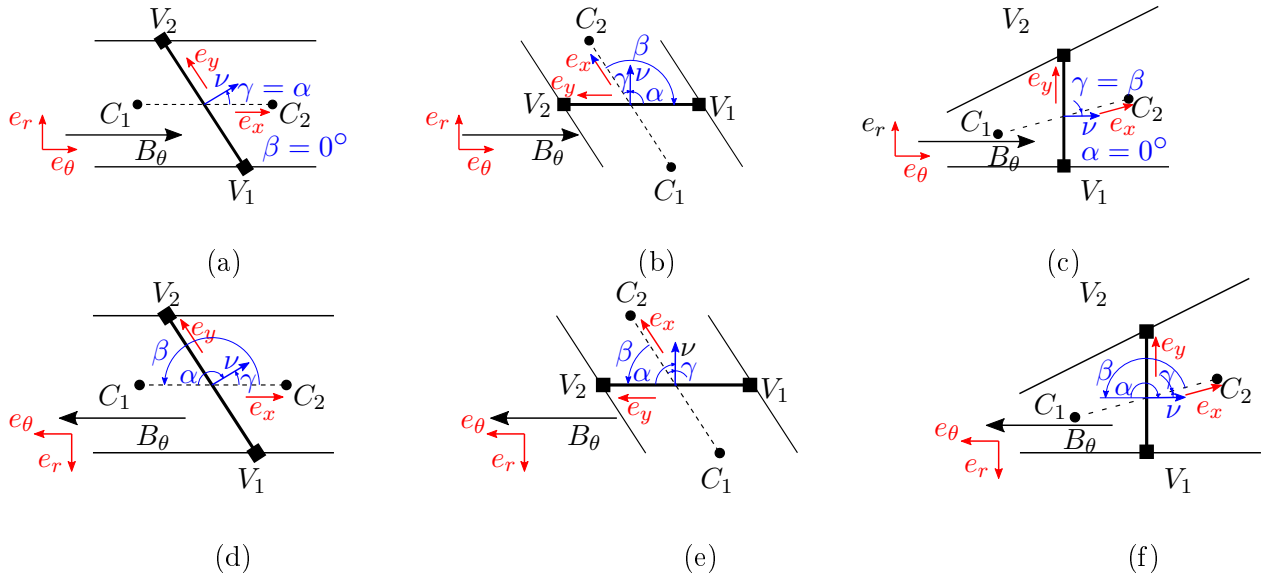

Figure 6: Specification of face angles for a few representative configurations: (a) slanted poloidal face between cells with aligned radial faces; (b) aligned radial face between cells with slanted poloidal faces; (c) orthogonal poloidal face between cells with one radial face aligned with the poloidal field, and the other radial face aligned with the vessel boundary. (d)-(f): same casees, with reversed orientation of the poloidal field.

the system, we obtain

$$\nabla_\theta u = \frac{\cos\alpha}{\cos\gamma}\nabla_x u + \frac{\sin\beta}{\cos\gamma}\nabla_y u, \tag{23}$$

$$\nabla_r u = \frac{\sin\alpha}{\cos\gamma}\nabla_x u + \frac{\cos\beta}{\cos\gamma}\nabla_y u. \tag{24}$$

For completeness we give also the component of the gradient normal the face, which is needed for some boundary conditions,

$$\nabla_\mathrm{n} u = \frac{1}{\cos\gamma}\nabla_x u + \frac{\sin\gamma}{\cos\gamma}\nabla_y u,$$

and note, trivially, that the component tangential to the face is the component in the y-direction: $\nabla_\mathrm{t} u = \nabla_y u$. Finally, the surface vector can be expressed as $\boldsymbol{S} = (\cos\alpha\,\mathbf{e}_\theta + \sin\alpha\,\mathbf{e}_r)S$.

Computing the poloidal and radial components of the gradient in general thus requires a combination of gradients in x and y directions. The need for vertex values, which in turn are computed from cell center values using an approriate interpolation scheme, leads to stencils that can become quite arbitrary and large, especially for grids including triangles. For example, structured grid areas with orthogonal, quadrangular cells lead to a standard 5-point stencils; structured grid areas with only misaligned poloidal faces give rise to a 9-point stencil [2]; the presence of an X-point typically leads to a 13-point stencil for the surrouding (quadrangular) cells, while the presence of triangles in the grid might lead to even broader stencils, since the number of cells connected to a single vertex is then no longer limited to 4. The unstructured solver fully accounts for this arbitrary number of points in the stencil, which leads to improved stability of the solver. However, complex stencils also tend to slow down the matrix inversion process and require smaller time steps, which is why it is recommended to use as much as possible locally structured, orthogonal grid topologies.

Also drift terms can be discretized using the expressions above. Only the component of the drift perpendicular to a face leads to transport across a face. Recalling that drifts have the form of a cross-diffusion terms in eq. (20), the net drift flow $\boldsymbol{\Gamma}_\mathrm{d}$ across a face is of the form

$$\boldsymbol{\Gamma}_\mathrm{d}\cdot\boldsymbol{S} = -D_\mathrm{d}\left(\nabla_r u S_\theta - \nabla_\theta u S_r\right) = -D_\mathrm{d}\nabla_y u S. \tag{25}$$

We have used that $\boldsymbol{S} = (\cos\alpha\,\mathbf{e}_\theta + \sin\alpha\,\mathbf{e}_r)S$. Thus, net drift flow across the face only depends on gradients along the face, while the decomposition of the drift in its poloidal and radial components (as needed for example for the collisions with neutrals) requires again the correct computation of both (poloidal and) radial gradients on each (possibly non-aligned) face.

Note on toroidal field In the current implementation, the poloidal field component is always positive (because $\mathbf{e}_\theta$ is along the poloidal projection of the magnetic field), while the toroidal field has a sign (because $\mathbf{e}_\phi$ points out of the page). Hence there is an inconsistency in the way both field components are treated. This issue can easily be removed as follows:
The alternative is to also take the toroidal unit vector along the toroidal projection of the field. In that case, the r coordinate flips direction compared to Fig. 5 if the toroidal field points into the page. However, the definitions of all the angles remain unchanged, since they relate only to the poloidal field. As a result, some signs change in some of the expressions above. Indeed, for toroidal field into the page we have $\mathbf{e}_\theta \cdot \mathbf{e}_x = \cos\beta$, $\mathbf{e}_r \cdot \mathbf{e}_x = \sin\beta$, $\mathbf{e}_\theta \cdot \mathbf{e}_y = -\sin\alpha$ and $\mathbf{e}_r \cdot \mathbf{e}_y = -\cos\alpha$. Note that eqs. (21) and (22) are still valid. Inverting again to find the poloidal and radial components, we now find

$$\nabla_\theta u = \frac{\cos\alpha}{\cos\gamma}\nabla_x u + \frac{\sin\beta}{\cos\gamma}\nabla_y u, \tag{26}$$

$$\nabla_r u = -\frac{\sin\alpha}{\cos\gamma}\nabla_x u - \frac{\cos\beta}{\cos\gamma}\nabla_y u. \tag{27}$$

As expected, only the signs in the equation for the radial component of the gradient change; the poloidal direction has remained unchanged. Expressions for normal and tangential components of the gradient do not change upon toroidal field reversal, because also the definitions of x and y are not affected by the toroidal field direction. However, the surface vector should now be expressed as $\boldsymbol{S} = (\cos\alpha\,\mathbf{e}_\theta - \sin\alpha\,\mathbf{e}_r)S$. The drift flow across a face now becomes

$$\boldsymbol{\Gamma}_\mathrm{d} \cdot \boldsymbol{S} = -D_\mathrm{d}\left(\nabla_r u S_\theta - \nabla_\theta u S_r\right) = D_\mathrm{d}\nabla_y u S. \tag{28}$$

Regarding the practical implementation, only the following needs to be done:

- Change sign of `fcQalf(:,1)` and `fcQbet(:,0)` for negative toroidal field
- Take absolute value of *Bb(:,3)
- Make sure to store sign of toroidal field somewhere, for correct interpretation/computation of unit vectors! Or: is effectively stored as sign of $\cos\alpha\cos\beta - \sin\alpha\sin\beta$ (> 0 for toroidal field out of page, < 0 for toroidal field into page). This sign change will carry through correctly for all flux/conductange/... computations.

Using the expressions for the fluxes and derivatives on faces given above, the governing equations can be discretized. Gradients in cell centers are obtained through interpolation of the cell face gradients. The discretization schemes of the structured version of the code have been generalized to arbitrary unstructured grids. Hybrid discretization schemes are used for the convection-diffusion flows, which are second order in space for diffusion-dominated flows, but tend to a first-order upwind scheme in convection-dominated flows. The implementation guarantees exact backwards compatibility for original, structured, orthogonal grids. However, the discretization schemes of the unstructured solver no longer neglect effects of misalignment w.r.t. the magnetic field, and hence the solver is more accurate in realistic plasma edge geometries, even for non-extended grid cases. As demonstrated in ref. [2], this is particularly important when using fluid neutral models. During the development of the new solver, various additional improvements to the numerical schemes were implemented. To avoid odd-even decoupling in the parallel direction due to the collocated velocity and density (pressure) fields, a Rhie-Chow scheme for compressible flow as described in ref. [8] is implemented for the parallel

direction. Flux limits for fluid neutrals now act on the total gradient-driven (particle or heat) fluxes, instead of only on the poloidal (for 'poloidal' faces) or radial (for 'radial' faces) component of the gradient as in the original code. This latter improvement removes an unwanted, artificial rotation of the flow direction due to the neutral flux limits in the original code. **??**. While the results are not reported here, the solver has been (and is being) verified extensively using the Method of Manufactured Solutions.

Pfirsch-Schlüter flows The Pfirsch-Schlüter flows introduced in the SOLPS5.2 drift model are used to reduce the strength of convective flows, and thereby stabilize the numerical scheme. These flows are computed following the general drift requirements outlined above. In addition to that, these flow components are also linearized and enter the matrix stencil. To do so, note first that drift terms can be included in convection-diffusion type formulations as cross-diffusion terms with $D_{\theta r} = -D_{r\theta} = D_\mathrm{d}$, so the total drift flow through a face with area S is

$$\boldsymbol{\Gamma} \cdot \boldsymbol{S} = -D_\mathrm{d} \frac{1}{h_\theta} \frac{\partial \phi}{\partial \theta} S \cos\alpha + D_\mathrm{d} \frac{1}{h_r} \frac{\partial \phi}{\partial r} S \sin\alpha, \tag{29}$$

with $S = \frac{\sqrt{g}}{h_x} \frac{1}{\cos\gamma}$. The poloidal and radial components of the flow are, respectively,

$$\text{poloidal drift:} - D_\mathrm{d} \left(\frac{\sin\alpha}{\cos\gamma} \frac{1}{h_x} \frac{\partial \phi}{\partial x} + \frac{\cos\beta}{\cos\gamma} \frac{1}{h_y} \frac{\partial \phi}{\partial y} \right) \frac{\sqrt{g}}{h_x} \frac{\cos\alpha}{\cos\gamma} = -D_\mathrm{d} \left(\sin\alpha \frac{\partial \phi}{\partial x} + \cos\beta \frac{h_x}{h_y} \frac{\partial \phi}{\partial y} \right) \frac{\sqrt{g}}{h_x^2} \frac{\cos\alpha}{\cos^2\gamma} \tag{30}$$

$$\text{radial drift:} \quad D_\mathrm{d} \left(\frac{\cos\alpha}{\cos\gamma} \frac{1}{h_x} \frac{\partial \phi}{\partial x} + \frac{\sin\beta}{\cos\gamma} \frac{1}{h_y} \frac{\partial \phi}{\partial y} \right) \frac{\sqrt{g}}{h_x} \frac{\sin\alpha}{\cos\gamma} = \quad D_\mathrm{d} \left(\cos\alpha \frac{\partial \phi}{\partial x} + \sin\beta \frac{h_x}{h_y} \frac{\partial \phi}{\partial y} \right) \frac{\sqrt{g}}{h_x^2} \frac{\sin\alpha}{\cos^2\gamma} \tag{31}$$

which is nearly identical to a 'regular' diffusive flux, buth with the 'poloidal' conductance multiplied with a 'radial' gradient and vice versa. To compute the relevant conductivities, we can hence simply use the existing `b2txcx` and `b2txcy` functions, and multiply them with *radial* and *poloidal* differences to obtain the drift flow. Also to compute the coefficients for the stencil equations, we can re-use the `calccoef` functionality written for the unstructured solver. To do this, we interpret $-D_{r\theta}$ as an effective *poloidal* conductance, and $D_{\theta r}$ as an effective *radial* conductance, and use the same logic as before. These Pfirsch-Schlüter contributions are now directly included in `flob0,conb0`, as part of the standard linearization of the convection-diffusion equation for a quantity `phi`:

```
flow(iFc) = 0.5*flo0(iFc,0)*(phi_c1 + phi_c2) - con0(iFc,0)*(phi_c2 - phi_c1)
          + 0.5*flo0(iFc,1)*(phi_v1 + phi_v2) - con0(iFc,1)*(phi_v2 - phi_v1)
```

We remark that for some equations, the linearized equation for `phi` is solved in terms of another variable; for example in the continuity equation, we solve for density corrections in terms of pressure

corrections, so `phi = pb*kmpr`. If we insert this in the stencil equation above, we find

$$
\begin{aligned}
\texttt{flow(iFc)} = \;& \texttt{0.5*flo0(iFc,0)*((pb*kmpr)\_c1 + (pb*kmpr)\_c2)} \\
& - \texttt{con0(iFc,0)*((pb*kmpr)\_c2 - (pb*kmpr)\_c1)} \\
& + \texttt{0.5*flo0(iFc,1)*((pb*kmpr)\_v1 + (pb*kmpr)\_v2)} \\
& - \texttt{con0(iFc,1)*((pb*kmpr)\_v2 - (pb*kmpr)\_v1)} \\
= \;& \texttt{0.5*flo0(iFc,0)*(kmpr\_c1 + kmpr\_c2)/2*(pb\_c1 + pb\_c2)} \\
& + \texttt{0.25*flo0(iFc,0)*(kmpr\_c2 - kmpr\_c1)*(pb\_c2 - pb\_c1)} \\
& - \texttt{con0(iFc,0)*(kmpr\_c1 + kmpr\_c2)/2*(pb\_c2 - pb\_c1)} \\
& - \texttt{con0(iFc,0)*(pb\_c1 + pb\_c2)/2*(kmpr\_c2 - kmpr\_c1)} \\
& + \texttt{0.5*flo0(iFc,1)*(kmpr\_v1 + kmpr\_v2)/2*(pb\_v1 + pb\_v2)} \\
& + \texttt{0.25*flo0(iFc,1)*(kmpr\_v2 - kmpr\_v1)*(pb\_v2 - pb\_v1)} \\
& - \texttt{con0(iFc,1)*(kmpr\_v1 + kmpr\_v2)/2*(pb\_v2 - pb\_v1)} \\
& - \texttt{con0(iFc,1)*(pb\_v1 + pb\_v2)/2*(kmpr\_v2 - kmpr\_v1)}
\end{aligned}
\tag{32}
$$

This whole expression can be interpreted as an effectieve convection-diffusion equation for `pb` if we define

$$
\begin{aligned}
\texttt{flo0\_eff(iFc,0)} &= 0.5 * \texttt{flo0(iFc,0)*(kmpr\_c1 + kmpr\_c2) - con0(iFc,0)*(kmpr\_c2 - kmpr\_c1)} \\
\texttt{flo0\_eff(iFc,1)} &= 0.5 * \texttt{flo0(iFc,1)*(kmpr\_v1 + kmpr\_v2) - con0(iFc,1)*(kmpr\_v2 - kmpr\_v1)} \\
\texttt{con0\_eff(iFc,0)} &= 0.5 * \texttt{con0(iFc,0)*(kmpr\_c1 + kmpr\_c2) - 0.25*flo0(iFc,0)*(kmpr\_c2 - kmpr\_c1)} \\
\texttt{con0\_eff(iFc,1)} &= 0.5 * \texttt{con0(iFc,1)*(kmpr\_v1 + kmpr\_v2) - 0.25*flo0(iFc,1)*(kmpr\_v2 - kmpr\_v1)}
\end{aligned}
\tag{33}
$$

The same kind of reasoning has now been applied for the Pfirsch-Schlüter contributions in the stencil. They are now directly included in `flob0,conb0` and put in the matrix in `b2uxus`, not through a separate call to `b2uxm9`, because their format is now identical to the other convection-diffusion type terms.

References

[1] Dekeyser, W., Boerner, P., Voskoboynikov, S., Rozhansky, V., Senichenkov, I., Kaveeva, L., Veselova, I., Vekshina, E., Bonnin, X., Pitts, R., Baelmans, M. (2021). Plasma edge simulations including realistic wall geometry with SOLPS-ITER. Nuclear Materials And Energy, 27, Art.No. 100999. doi: 10.1016/j.nme.2021.100999

[2] Dekeyser, W., Bonnin, X., Lisgo, S.W., Pitts, R.A., LaBombard, B. (2019). Implementation of a 9-point stencil in SOLPS-ITER and implications for Alcator C-Mod divertor plasma simulations. NUCLEAR MATERIALS AND ENERGY, 18, 125-130. doi: 10.1016/j.nme.2018.12.016

[3] Horsten, N., Samaey, G., Baelmans, M. (2017). Development and assessment of 2D fluid neutral models that include atomic databases and a microscopic reflection model. NUCLEAR FUSION, 57, 116043.

[4] D. Bohm, in *The Characteristics of Electrical Discharges in Magnetic Fields*, eds. A. Guthrie and R.K. Wakerling, New York, McGraw-Hill, Ch 3 (1949).

[5] R. Chodura, *Physics of Plasma-Wall Interactions in Controlled Fusion*, eds. D.E. Post and R. Behrisch, New York, Plenum Press (1986).

[6] P.C. Stangeby, *The Plasma Boundary of Magnetic Fusion Devices*, IOP Publishing, Bristol and Philadelphia (2000).

[7] D. Reiter, The EIRENE Code User Manual, available from http://www.eirene.de.

[8] F. Moukalled, L. Mangani, M. Darwish. *The Finite Volume Method in Computational Fluid Dynamics*, Springer, Switzerland, 2016.

[9] R. Marchand and M. Dumberry, Computer Physics Communiations **96**, 232–246 (1996).

[10] M. Blommaert, W. Dekeyser, N. Horsten et al., Contrib. Plasma Phys. **58** 718 (2018).

[11] W. Van Uytven, M. Blommaert, W. Dekeyser et al., Contrib. Plasma Phys. **60**, e201900147 (2020).

[12] M. Blommaert, N. Horsten, P. Boerner, W. Dekeyser, Nuclear Materials and Energy **19** 28–33 (2019).

[13] N. Horsten, M. Groth, et al., Nuclear Materials and Energy **27** 100969 (2021).

[14] M. Baelmans, M. Blommaert, W. Dekeyser, T. Van Oevelen, Nucl. Fusion **57**, 036022 (2017).

[15] S. Carli, M. Blommaert, W. Dekeyser, M. Baelmans, Nucl. Mater. Energy **18**, 6 (2019).

[16] S. Carli,W. Dekeyser, R. Coosemans et al., Contrib. Plasma Phys. **60**, e201900155 (2020).

[17] S. Carli, W. Dekeyser, M. Blommaert et al., Contrib. Plasma Phys., e202100184 (2021).

[18] R. Coosemans, W. Dekeyser, M. Baelmans, Contrib. Plasma Phys. **60**, e201900156 (2020).

[19] R. Coosemans, W. Dekeyser, M. Baelmans, Phys. Plasmas **28**, 012302 (2021).

[20] W. Dekeyser, R. Coosemans, S. Carli, M. Baelmans, Contrib. Plasma Phys., 2022, accepted.

[21] Rozhansky VA et al 2001 *Nuclear Fusion* **41** 387.

[22] Rozhansky VA et al 2009 *Nuclear Fusion* **49** 025007.