Implementation of poloidal density variation in collision operator

September 28, 2017

The aim with this document is to explain the modifications in `SFINCS` that have been done, in order to include poloidal density variation in the Fokker-Planck collision operator.

As is explained in ref. [1] poloidal density variation can be included by modifying the lowest order distribution function

$$f_{Ms}(\psi) \rightarrow f_{Ms}(\psi) e^{-Z_s e \Phi_1(\theta,\zeta)/T_s},$$

where $f_M(\psi)$ is the flux-function Maxwellian, Φ_1 are first order variation of the electrostatic potential, and Z_s, T_s are the charge and temperature of species s respectively. To implement this change one may simply modify the species density accordingly

$$n_s \rightarrow n_s e^{-Ze\Phi_1(\theta,\zeta)/T},$$

in the original equations for the various terms in the collision operator, described in ref. [2].

The linearised collision operator takes the form

$$\begin{aligned} C_{ab}^{L:f0} = C_{ab}\left\{f_{aM}, f_{bM}\right\} e^{-(Z_a/T_a + Z_b/T_b)e\Phi_1(\theta,\zeta)} + C_{ab}\left\{f_{a1}, f_{bM}\right\} e^{-Z_b/T_b e\Phi_1(\theta,\zeta)} + \\ + C_{ab}\left\{f_{aM}, f_{b1}\right\} e^{-Z_a/T_a e\Phi_1(\theta,\zeta)}, \end{aligned} \tag{1}$$

which describes collisions between species a and b. In normalised `SFINCS` units this becomes

$$\begin{aligned} \hat{C}_{ab}^{L:f0} = \hat{C}_{ab}\left\{f_{aM}, f_{bM}\right\} e^{-\left(\hat{Z}_a/\hat{T}_a + \hat{Z}_b/\hat{T}_b\right)\alpha\hat{\Phi}_1(\theta,\zeta)} + \hat{C}_{ab}\left\{f_{a1}, f_{bM}\right\} e^{-\hat{Z}_b/\hat{T}_b\alpha\hat{\Phi}_1(\theta,\zeta)} + \\ + \hat{C}_{ab}\left\{f_{aM}, f_{b1}\right\} e^{-\hat{Z}_a/\hat{T}_a\alpha\hat{\Phi}_1(\theta,\zeta)}. \end{aligned} \tag{2}$$

The first term on the RHS of Eq. (2) is the *temperature equilibration term*. Since this term does not include f_{1s} it has to be treated differently compared to the other two cases.

The contribution to the Jacobian are $\partial\hat{C}_{ab}^{L:f0}/\partial\hat{f}_{1s}$ and $\partial\hat{C}_{ab}^{L:f0}/\partial\hat{\Phi}_1$. The first is the same as in Eq. (2) except that the collision operator does not act on the distribution function. In the second case we get

$$\begin{aligned} \hat{C}_{ab}^{L:f0} = -\left(\hat{Z}_a/\hat{T}_a + \hat{Z}_b/\hat{T}_b\right)\alpha\hat{C}_{ab}\left\{f_{aM}, f_{bM}\right\} e^{-\left(\hat{Z}_a/\hat{T}_a + \hat{Z}_b/\hat{T}_b\right)\alpha\hat{\Phi}_1(\theta,\zeta)} \\ -\hat{Z}_b/\hat{T}_b\alpha\ \hat{C}_{ab}\left\{f_{a1}, f_{bM}\right\} e^{-\hat{Z}_b/\hat{T}_b\alpha\hat{\Phi}_1(\theta,\zeta)} - \hat{Z}_a/\hat{T}_a\alpha\hat{\Phi}_1(\theta,\zeta)\hat{C}_{ab}\left\{f_{aM}, f_{b1}\right\} e^{-\hat{Z}_a/\hat{T}_a\alpha\hat{\Phi}_1(\theta,\zeta)}. \end{aligned} \tag{3}$$

Apart from the different prefactor, the term in Eq. (3) differs from term in $\partial \hat{C}_{ab}^{L:f0}/\partial \hat{f}_{1s}$ by acting on the first order distribution function f_{1s}. For example, in the first case, terms such as $\partial f_{1s}/\partial x_s$ become

$$\frac{1}{\partial f_{1s}} \frac{\partial f_{1s}}{\partial x_s} = \frac{\partial}{\partial x_s},$$

whereas in the second case, when take the derivative with respect to Φ_1, we have to actually calculate the derivative $\partial f_{1s}/\partial x_s$.

Implementation in the Code

To implement Eq. (2) we need to modify the evaluation of the residual and the jacobian. The residual is evaluated in `evaluateResidual.F90` and the jacobian in `evaluateJacobian.F90`. Both these two routines call the functions in `PopulateMatrix.F90`, where the actual assembly of the residual and jacobian matrices are done. Therefore, we only need to modify `PopulateMatrix.F90`.

Residual

In the residual we have to include the extra pre-factor appearing in Eq. (2). This is done by modifying the last block in the calculation of the collision operator, just before the terms are saved into the main matrix. We introduce the pre-factor `preFactor` which is normally equal to one, but $e^{-\hat{Z}_a/\hat{T}_a \alpha \hat{\Phi}_1(\theta,\zeta)}$ when poloidal density variation is included (by setting the flag `poloidalVariationInCollisionOperator = .true.`).

```
preFactor = 1.0 ! Initiate the preFactor used to multiply CHat before saving
    into the Main matrix
if (poloidalVariationInCollisionOperator) then
   preFactor =
      exp(-Zs(iSpeciesA)*alpha*Phi1Hat(itheta,izeta)/Thats(iSpeciesA))
```

This factor then multiplies the collision operator, just before saving into the main matrix

```
call MatSetValueSparse(matrix, rowIndex, colIndex, &
     -nu_n*preFactor*CHat(ix_row,ix_col), ADD_VALUES, ierr) ! Multiply CHat
        with preFactor before saving
```

Temperature equilibration term

In the original version of the code, the temperature equilibration term is calculated in `evaluateResidual.F90` by using `whichMatrix = 2` when calling `PopulateMatrix.F90`. The output is then multiplied with the first order distribution function f_{0s}. Since the original version of `SFINCS` already includes Φ_1 in the definition of f_{0s} (in the subroutine `init_f0()` inside `PopulateMatrix.F90`), the same routines as were presented above can be used. In the end, when calling `PopulateMatrix` we include the pre-factor $e^{-\hat{Z}_a/\hat{T}_a \alpha \hat{\Phi}_1(\theta,\zeta)}$.

Then by multiplying with f_{0b} in `evaluateResidual.F90` we get the desired total pre-factor $e^{-\left(\hat{Z}_a/\hat{T}_a+\hat{Z}_b/\hat{T}_b\right)\alpha\hat{\Phi}_1(\theta,\zeta)}$.

Jacobian

The $\partial C/\partial f_{1s}$ contribution in the jacobian is at this point already taking the density variations into account if we implement the modifications explained in Section . However, we need to add the calculation of $\partial C/\partial\Phi_1$.

The $\partial C/\partial\Phi_1$ term

To implement the derivative with respect to Φ_1 we add another block right after the calculation of the residual terms. This block should be evaluated whenever the jacobian (`whichmatrix =1`) or the preconditioner (`whichmatrix =0`) is calculated, with `poloidalVariationInCollisionOperator = .true.`

```
if (poloidalVariationInCollisionOperator .and. (whichMatrix == 1 .or.
    whichMatrix == 0)) then
```

We begin with treating the other two terms appearing in Eq. (3) and will then look at the temperature equilibration term. For the second two terms in Eq. (3) we first have to implement the correct pre-factor.

```
preFactor = -Zs(iSpeciesA)*alpha/Thats(iSpeciesA)*exp(-Zs(iSpeciesA)&
*alpha*Phi1Hat(itheta,izeta)/Thats(iSpeciesA))
```

The second change is including the first order distribution function (and calculating derivatives thereof). For this purpose we first have to initiate this distribution function, by reading the appropriate terms in the current state vector

```
index = getIndex(iSpeciesB,ix,L+1,itheta,izeta,BLOCK_F) ! f1b from statevector
f1b(ix) = stateArray(index + 1)
```

What is left is to multiply this vector with the collision operator matrix `CHat` containing terms such as $\partial/\partial x_s$.

```
CHatTimesf = matmul(CHat,f1b)
```

The output is then saved into the main matrix in the $\partial\hat{C}/\partial\hat{\Phi}_1$ block,

```
call MatSetValue(matrix, rowIndex, colIndex, &
-nu_n*preFactor*CHatTimesf(ix_col), ADD_VALUES, ierr)
```

using the approptiate row and column indices.

```
rowIndex=getIndex(iSpeciesA,ix_row,L+1,itheta,izeta,BLOCK_F)
do ix_col = max(ixMinCol,min_x_for_L(L)),Nx
   ! Get column index for the d/dPhi1 terms
   colIndex=getIndex(1,1,1,itheta,izeta,BLOCK_QN)
```

Temperature equilibration

If the temperature equilibration terms should be included, we have to add additional terms into the main matrix, when calculating $\partial\hat{C}/\partial\hat{\Phi}_1$. To begin with we need to include the different pre-factor (since in this case f_0 from `init_f0()` is not used).

```
if (includeTemperatureEquilibrationTerm) then
   ! Get the correct preFactor
   preFactor = (-Zs(iSpeciesA)*alpha/Thats(iSpeciesA) -Zs(iSpeciesB)*alpha/ &
    Thats(iSpeciesB))
   *exp(-Zs(iSpeciesA)*alpha*Phi1Hat(itheta,izeta)/Thats(iSpeciesA)-Zs(iSpeciesB)
       &
   *alpha*Phi1Hat(itheta,izeta)/Thats(iSpeciesB))
```

The second change originates from the fact that the collision operator in this temperature equilibration case, should multiply the Maxwellian for species b and not f_{1b} (as was previously the case in Section). First, we have to initiate this distribution function

```
 if (includeTemperatureEquilibrationTerm) then
    fM(ix) =
       expxb2(ix)*nhats(iSpeciesB)*mhats(iSpeciesB)*sqrt(mhats(iSpeciesB)) &
     /(pi*sqrtpi*Thats(iSpeciesB)*sqrt(Thats(iSpeciesB)))
     end if
```

and then use it to multiply to the total collision operator.

```
  CHatTimesf = matmul(CHat,fM)
```

Then we add this term into the main matrix, using the the same procedure as was explained in Section

References

[1] S. Buller, *Poloidal variation in collision operator* (2017).

[2] M. Landreman, *Technical Documentation for version 3 of SFINCS* (2014).