Implementation of Common Non-Maxwellians Background Distributions in GENE

Boris Andrews, Dr Salomon Janhunen
Tokamak Energy Ltd

2nd November 2022

UNIVERSITY OF OXFORD
Mathematical Institute

Oxford Mathematics

Contents

BACKGROUND: Salomon has run beam simulations in `GENE`, using 3 (4) species:

Species	Particle	Density	Distribution
Electrons	Electron	High	Maxwellian $\begin{cases} \text{Velocity} = \mathbf{0} \\ \text{Temperature} = \text{Low} \end{cases}$
Ions	Proton	High	Maxwellian $\begin{cases} \text{Velocity} = \mathbf{0} \\ \text{Temperature} = \text{Low} \end{cases}$
Beam	Proton	Low	Maxwellian $\begin{cases} \text{Velocity} = \mathbf{0} \\ \text{Temperature} = \text{High} \end{cases}$

CRITICISM

- Poorly represented beam distribution function!
- Can we improve this?

BACKGROUND: Salomon has run beam simulations in `GENE`, using 3 (4) species:

Species	Particle	Density	Distribution
Electrons	Electron	High	Maxwellian $\begin{cases} \text{Velocity} = \mathbf{0} \\ \text{Temperature} = \text{Low} \end{cases}$
Ions	Proton	High	Maxwellian $\begin{cases} \text{Velocity} = \mathbf{0} \\ \text{Temperature} = \text{Low} \end{cases}$
Beam	Proton	Low	Maxwellian $\begin{cases} \text{Velocity} = \mathbf{0} \\ \text{Temperature} = \text{High} \end{cases}$

CRITICISM

- Poorly represented beam distribution function!
- Can we improve this?

BACKGROUND: Salomon has run beam simulations in `GENE`, using 3 (4) species:

Species	Particle	Density	Distribution	
Electrons	Electron	High	Maxwellian	Velocity $= \mathbf{0}$ Temperature $=$ Low
Ions	Proton	High	Maxwellian	Velocity $= \mathbf{0}$ Temperature $=$ Low
Beam	Proton	Low	Maxwellian	Velocity $= \mathbf{0}$ Temperature $=$ High

CRITICISM

- Poorly represented beam distribution function!
- Can we improve this?

BACKGROUND: Salomon has run beam simulations in `GENE`, using 3 (4) species:

Species	Particle	Density	Distribution
Electrons	Electron	High	Maxwellian $\begin{cases} \text{Velocity} = \mathbf{0} \\ \text{Temperature} = \text{Low} \end{cases}$
Ions	Proton	High	Maxwellian $\begin{cases} \text{Velocity} = \mathbf{0} \\ \text{Temperature} = \text{Low} \end{cases}$
Beam	Proton	Low	Maxwellian $\begin{cases} \text{Velocity} = \text{Non}-\mathbf{0} \\ \text{Temperature} = \text{High} \end{cases}$

CRITICISM

- Poorly represented beam distribution function!
- Can we improve this?

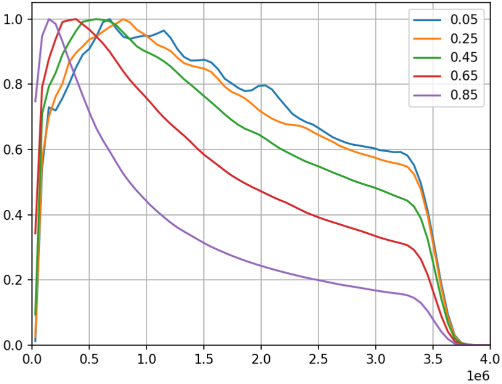


Figure: Energy density profiles (on different flux tubes) showing the damping of (alpha) particles over a certain threshold energy(/velocity)

BACKGROUND: Salomon has run beam simulations in `GENE`, using 3 (4) species:

Species	Particle	Density	Distribution	
Electrons	Electron	High	Maxwellian	Velocity $= \mathbf{0}$ Temperature $=$ Low
Ions	Proton	High	Maxwellian	Velocity $= \mathbf{0}$ Temperature $=$ Low
Beam	Proton	Low	Damped Maxwellian	Velocity $=$ Non$-\mathbf{0}$ Temperature $=$ High Damping speed $= v_A$

PROBLEM

GENE supports:

- ✔ *Maxwellian* backgrounds
- ✱ *Shifted Maxwellian* backgrounds
- ✘ *Fully non-Maxwellian* backgrounds (i.e. here damped distributions)

PROJECT

- *Fix* generic background distributions support
- *Add* damped Maxwellians

PROBLEM

`GENE` supports:

- ✔ *Maxwellian* backgrounds
- ✱ *Shifted Maxwellian* backgrounds
- ✘ *Fully non-Maxwellian* backgrounds (i.e. here damped distributions)

PROJECT

- *Fix* generic background distributions support
- *Add* damped Maxwellians

Contents

Contents

UNIVERSITY OF OXFORD

WORK

- Some work done by me/some by Gabriele Merlo
- (*Lots* of digging in *lots* of lines of code)

TESTING

- In collaboration with team here, team at PPPL have some simulations of 2 ST40 shots (9831 & 9894) in TRANSP
- Should be able to run comparisons against these!

Implementation — Fixing Generic Backgrounds

UNIVERSITY OF OXFORD

WORK

- Some work done by me/some by Gabriele Merlo
- (*Lots* of digging in *lots* of lines of code)

TESTING

- In collaboration with team here, team at PPPL have some simulations of 2 ST40 shots (9831 & 9894) in TRANSP
- Should be able to run comparisons against these!

Contents

.WHAT DISTRIBUTIONS TO CONSIDER...

Tried to be as general here as possible:

- Consider arbitrary "damping" as a map on particle speeds

$$(v_{\|}, \mu)^{\text{pre}} \mapsto (v_{\|}, \mu)^{\text{post}} \tag{1}$$

- Equivalently maps distribution *before* damping, f_{s0}^{pre}, to distribution *after* damping, f_{s0}^{post}

$$f_{s0}^{\text{post}}\left(v_{\|}^{\text{post}}, \mu^{\text{post}}\right) = J\left(v_{\|}, \mu\right) f_{s0}^{\text{pre}}\left(v_{\|}^{pre}, \mu^{\text{pre}}\right) \tag{2}$$

$$J = \partial_{v_{\|}^{\text{post}}} v_{\|}^{\text{pre}} \partial_{\mu^{\text{post}}} \mu^{\text{pre}} - \partial_{\mu^{\text{post}}} v_{\|}^{\text{pre}} \partial_{v_{\|}^{\text{post}}} \mu^{\text{pre}} \tag{3}$$

Implementation — Adding Damped Distribution

Example

$$v^{\text{pre}} = \left[1 + \left(\frac{v^{\text{post}}}{v^{\text{damp}}}\right)^k\right] v^{\text{post}} \tag{4}$$

As $v \to \infty$:

$$f_{s0}^{\text{pre}} = \exp\left(-\text{const.}v^2\right) \tag{5}$$

$$f_{s0}^{\text{post}} = \exp\left(-\text{const.}v^{2+k}\right) \tag{6}$$

- Example such damping function

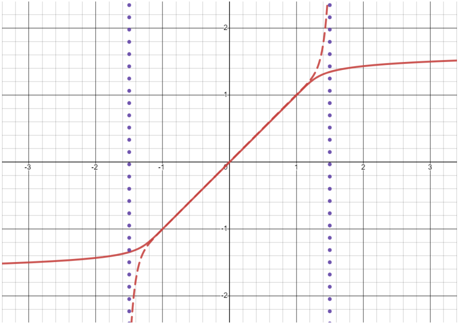


Figure: $v^{\text{damp}} = 1.5,\ k = 20$

- Damped velocity distribution

Figure: $v^{\mathrm{damp}} = 1.5,\ k = 20,\ u = 0,\ T = 0.48$

- Damped energy distribution

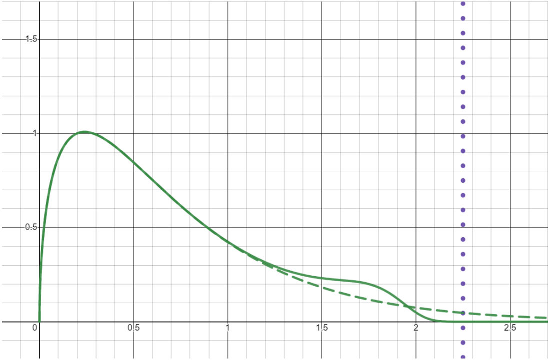


Figure: $v^{\text{damp}} = 1.5,\ k = 20,\ u = 0,\ T = 0.48$

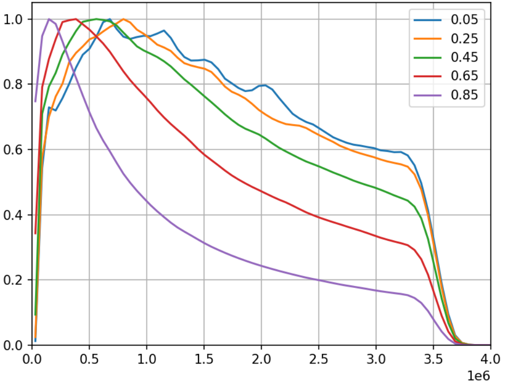


Figure: Energy density profiles (on different flux tubes) showing the damping of (alpha) particles over a certain threshold energy(/velocity)

- Damped *shifted* velocity distribution

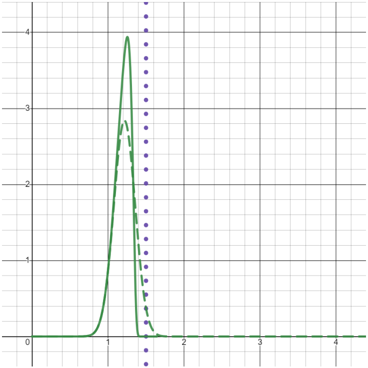


Figure: $v^{\text{damp}} = 1.5,\ k = 20,\ u = 1.2,\ T = 0.04$

- Damped *shifted* energy distribution

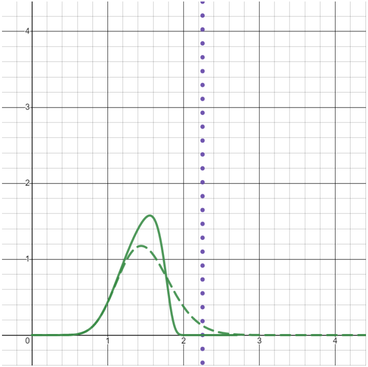
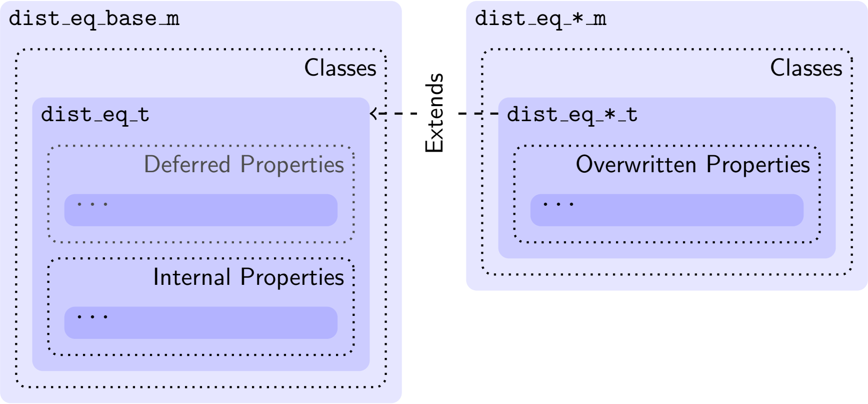


Figure: $v^{\text{damp}} = 1.5,\ k = 20,\ u = 1.2,\ T = 0.04$

HOW DO WE IMPLEMENT THIS IN FULL GENERALITY?

Implementation — Adding Damped Distribution

.HOW DO WE IMPLEMENT THIS IN FULL GENERALITY?



- What kind of functionals does the `dist_eq_*_damped_t` class instance need to evaluate?

Procedure	Functional
`get_*D_dv`	$\partial_{v_\parallel} f_{s0}$
`get_*D_dm`	$\partial_\mu f_{s0}$
`get_*D_dvdm`	$\frac{B_0}{2}\partial_{v_\parallel} f_{s0} - v_\parallel \partial_\mu f_{s0}$
$\vdots$	$\vdots$

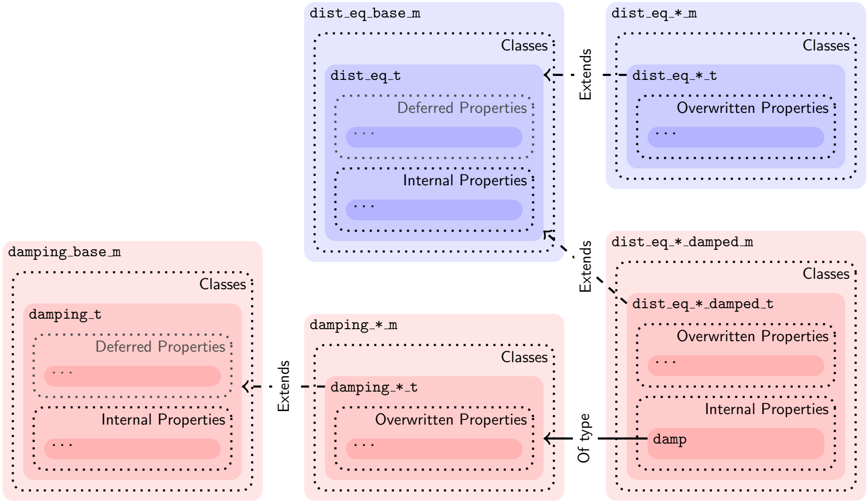
- Needs access to:
  - $v_\parallel^{\text{pre}}$, μ^{pre} (and their derivatives up to 2nd order)
  - J (and its derivative up to 1st order)

- What kind of functionals does the `dist_eq_*_damped_t` class instance need to evaluate?

Procedure	Functional
`get_*D_dv`	$\partial_{v_{\|\|}} f_{s0}$
`get_*D_dm`	$\partial_{\mu} f_{s0}$
`get_*D_dvdm`	$\frac{B_0}{2}\partial_{v_{\|\|}} f_{s0} - v_{\|\|}\partial_{\mu} f_{s0}$
⋮	⋮

- Needs access to:
  - $v_{||}^{\mathrm{pre}}$, μ^{pre} (and their derivatives up to 2nd order)
  - J (and its derivative up to 1st order)

Implementation — Adding Damped Distribution



Implementation — Adding Damped Distribution

UNIVERSITY OF OXFORD

.**CODE NOW AVAILABLE!** In my Gitlab repository:

- Just ask me for access! :)
- https://gitlab.mpcdf.mpg.de/g-borisandrews/gene-gpu
- Also have a far more detailed report to which Salomon should have access