



How to keep up with fast- ions: the technical part

A. Snicker et al.

24/11/2023 VTT – beyond the obvious



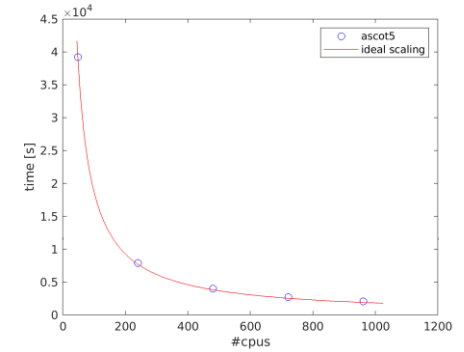
- Numerical models in nutshell
- Comparing typical runs
- Newest new: ASCOT-BMC, ASCOT-GPU
- Code development: git, slack, community

A. Snicker et al.

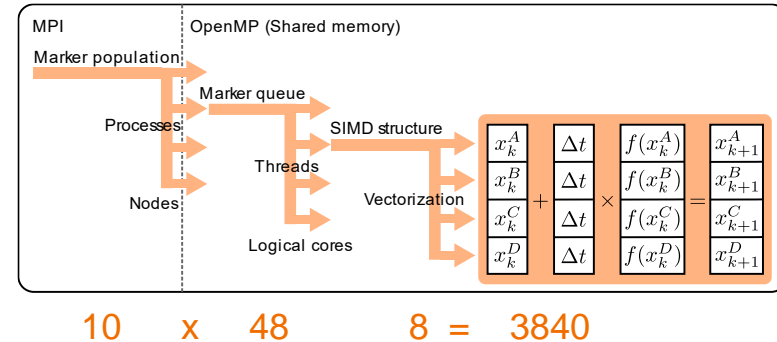
24/11/2023 VTT – beyond the obvious

Numerical models, of HPC relevance 1/2

- ASCOT is a Monte Carlo orbit-following code
 - For minority species (like fast-ions)
 - No self-interactions
 - Coulomb collisions with background plasma
 - Near embarrassingly parallel using MPI/MPI+OpenMP
- Several layers of parallelism available in ASCOT5
 - Marker ensemble -> MPI
 - Marker queue -> OpenMP
 - Vectorization -> SIMD

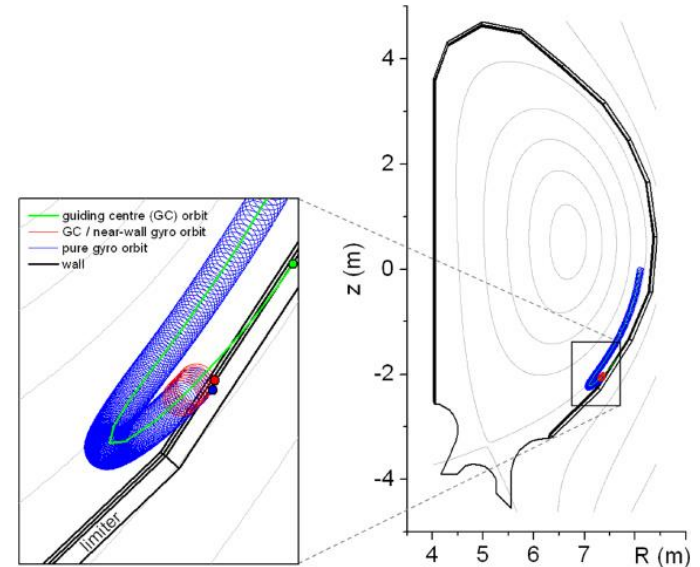


Scalability test of ASCOT5 with the number of CPUs in MARCONI



Numerical models, of HPC relevance 2/2

- Guiding center vs. gyro motion [1]
 - For most applications GC is fine
 - But GC approximation does fail, e.g. [2]
- Running with gyro motion comes with cost
 - Roughly 30-100 times more CPU hours
- Typical tokamak run with 2D axisymmetric field
 - ~30 core hours (100k markers)
- Typical stellarator run with 3D field
 - ~300 core hours (100k markers)
- Additional considerations:
 - Alfvén eigenmodes [3] – 1D spline per mode
 - ICRH via RFOF operator – two-stage simulation process [4]



[1] A. Snicker *et al* 2012 *Nucl. Fusion* **52** 094011

[2] A. Sperduti *et al* 2021 *Nucl. Fusion* **61** 016028, A. Sperduti *et al* 2021 *Plasma Phys. Control. Fusion* **63** 015015, P. Ollus *et al* 2022 *Plasma Phys. Control. Fusion* **64** 035014

[3] A. Snicker *et al* 2013 *Nucl. Fusion* **53** 093028

[4] S. Sipilä *et al* 2021 *Nucl. Fusion* **61** 086026



VTT

- Numerical models in nutshell
- Comparing typical runs
- Newest new: ASCOT-BMC, ASCOT-GPU
- Code development: git, slack, community

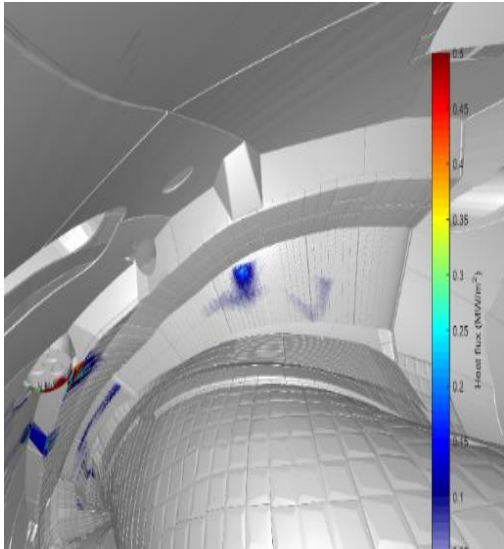
A. Snicker et al.

24/11/2023 VTT – beyond the obvious

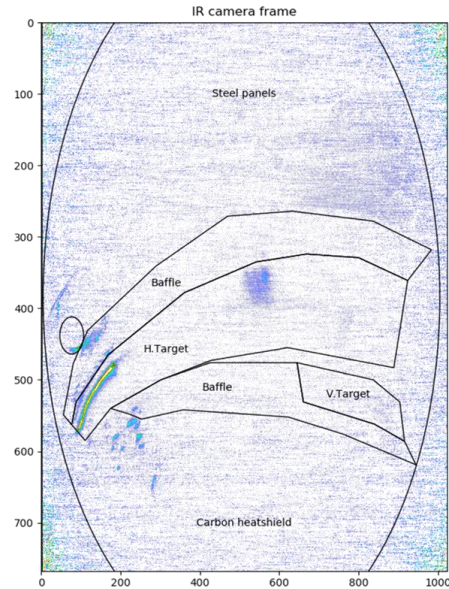
ASCOT simulations showcases

- W7-X wall power load high-definition
 - Example of complex 3D geometry and useful validation
- ASCOT-RFOF for AUG
 - Example of the added CPU demands by IC operator
- ITER FILD simulations
 - Example of the intrinsic Monte Carlo statistics problem

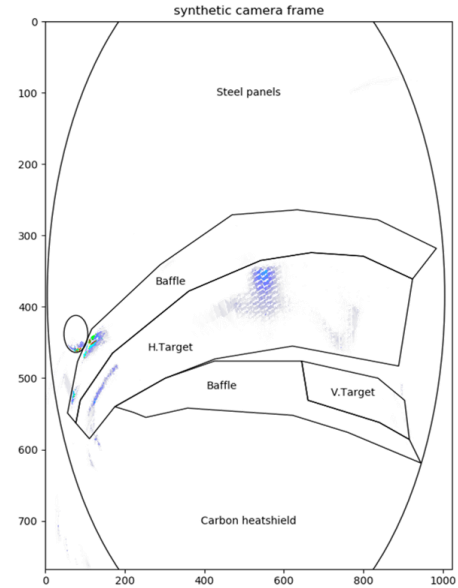
Synthetic IR studies with ASCOT (in W7-X)



ASCOT'S view of W7-X intestines



InfraRed (IR) camera frame of the same place



ASCOT's synthetic IR camera frame

Wall design improved thanks to ASCOT simulations !

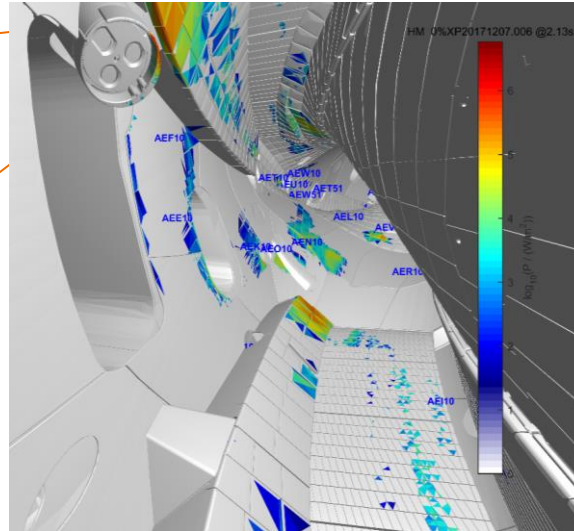
Fragile (sapphire) vacuum windows



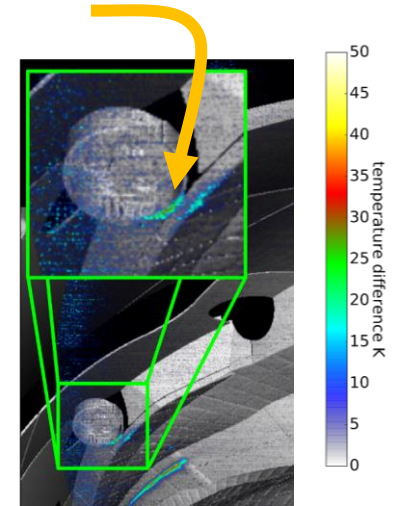
ASCOT predicted
excessive NBI power loads



Protective collar installed
before starting the beams



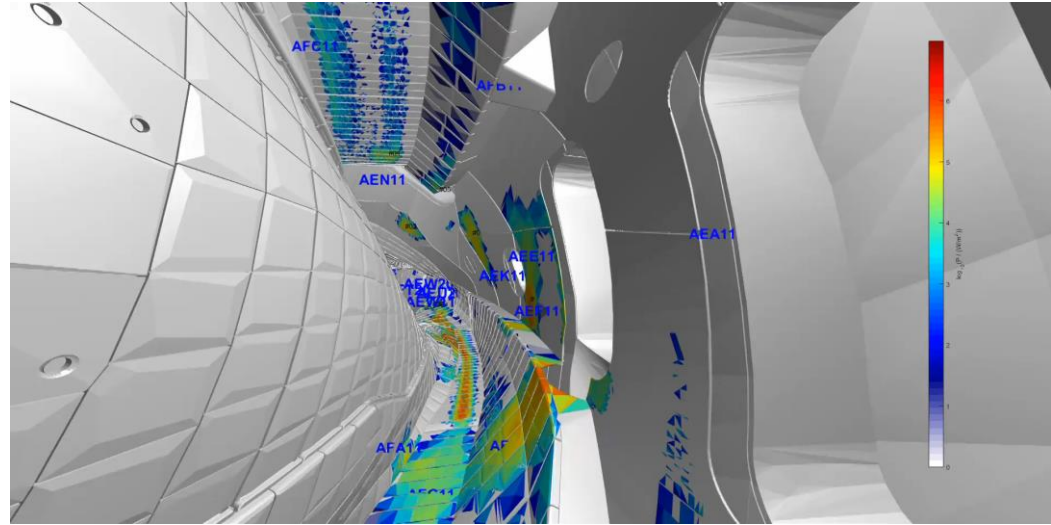
Wendelstein 7-X á l'ASCOT



power loads in excess
of 1.5MW/m^2 measured

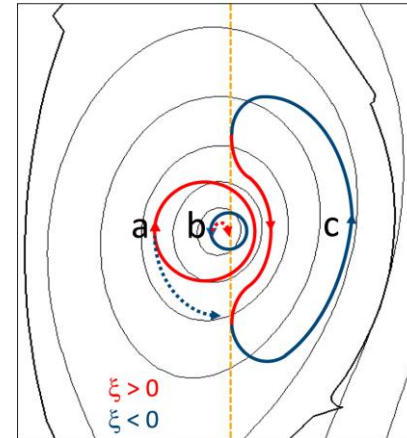
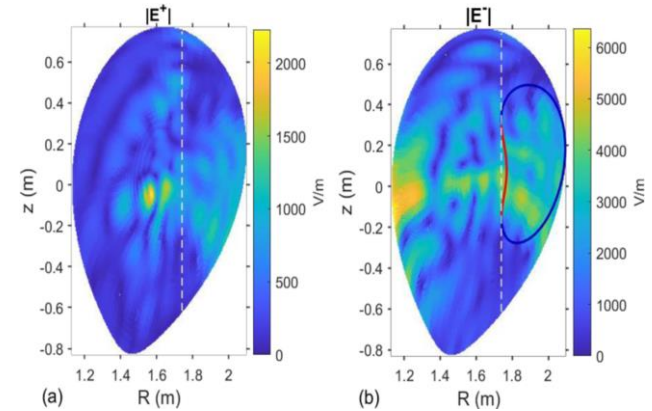
Simulation specifications

- 1M-100M GC markers
 - Converged peak heat load (100M)
 - Hot spots and estimated peaks (1M)
- CAD 3D wall, ~4M triangles
 - Triangle areas 1 mm² to 0.2 m²
- Simulation was MARCONI commissioning
- Estimated CPU hours: 300k



AUG-RFOF synthetic FILD

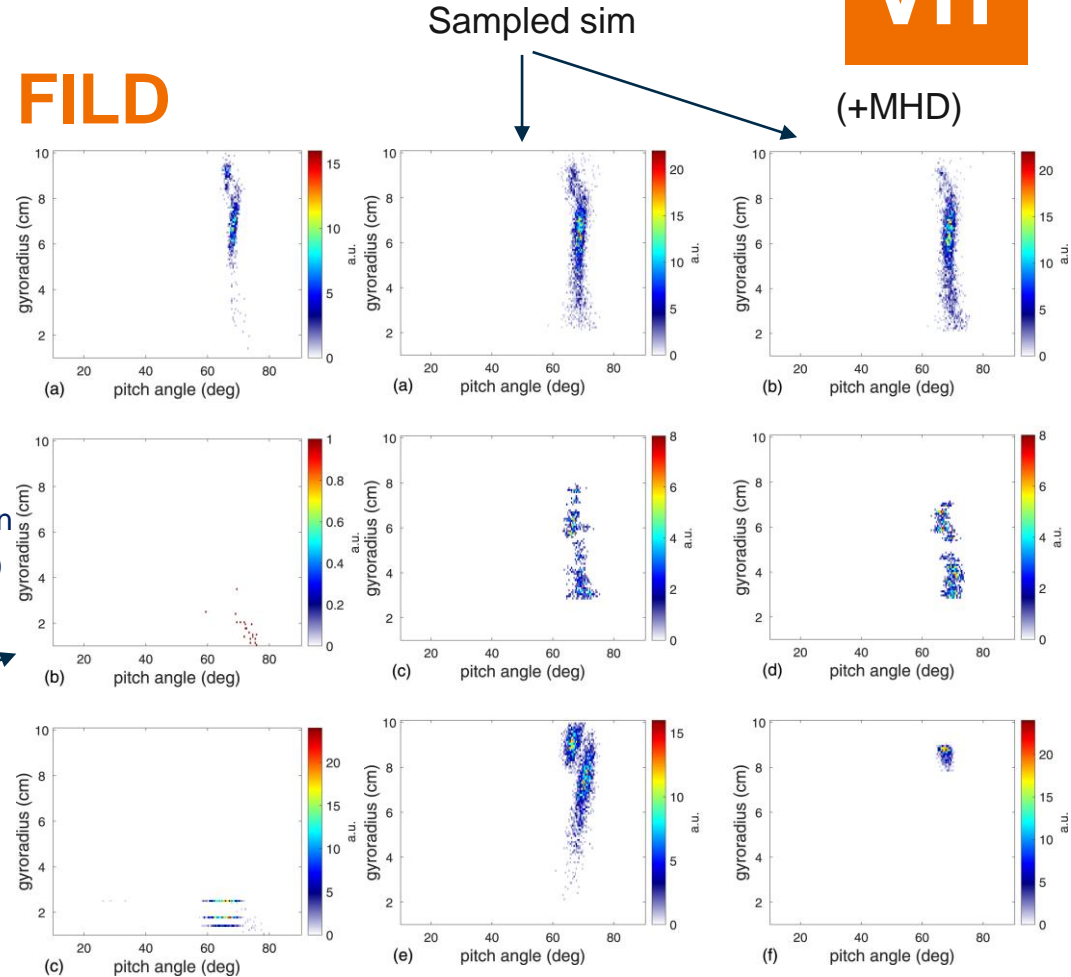
- Simulate ICRH-heated fast-ions
 - Both NBI and minority ions absorb IC power
 - Synthetic FILD vs. measured FILD
- Two-staged simulation process
 - Run with MC RFOF operator
 - Obtain the steady-state distribution function
 - Marker sampling for synthetic FILD simulation
- Improved statistics for the synthetic FILD



AUG-RFOF synthetic FILD

- Simulate ICRH-heated fast-ions
 - Both NBI and minority ions absorb IC power
 - Synthetic FILD vs. measured FILD
- Two-staged simulation process
 - Run with MC RFOF operator
 - Obtain the steady-state distribution function
 - Marker sampling for synthetic FILD simulation
- Improved statistics for the synthetic FILD

Direct sim

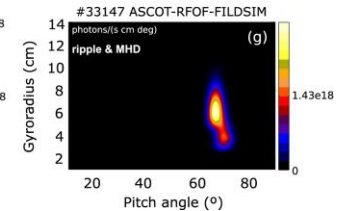
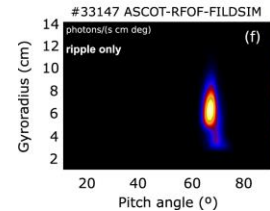
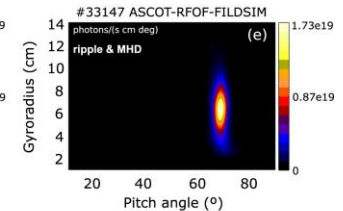
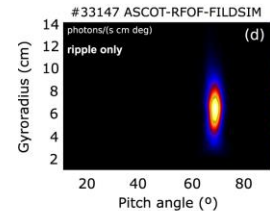
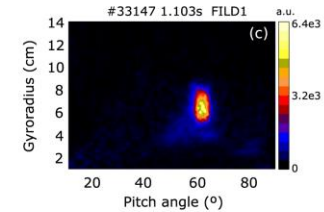
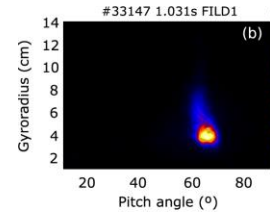
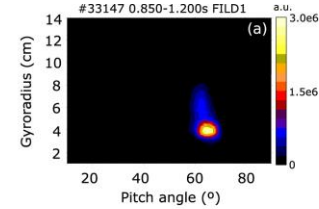


AUG-RFOF synthetic FILD

- Simulate ICRH-heated fast-ions
 - Both NBI and minority ions absorb IC power
 - Synthetic FILD vs. measured FILD
- Two-staged simulation process
 - Run with MC RFOF operator
 - Obtain the steady-state distribution function
 - Marker sampling for synthetic FILD simulation
- Improved statistics for the synthetic FILD
 - Questions remain:
 - What is the actual H population?
 - Why the H tail is simulated higher?

H

D

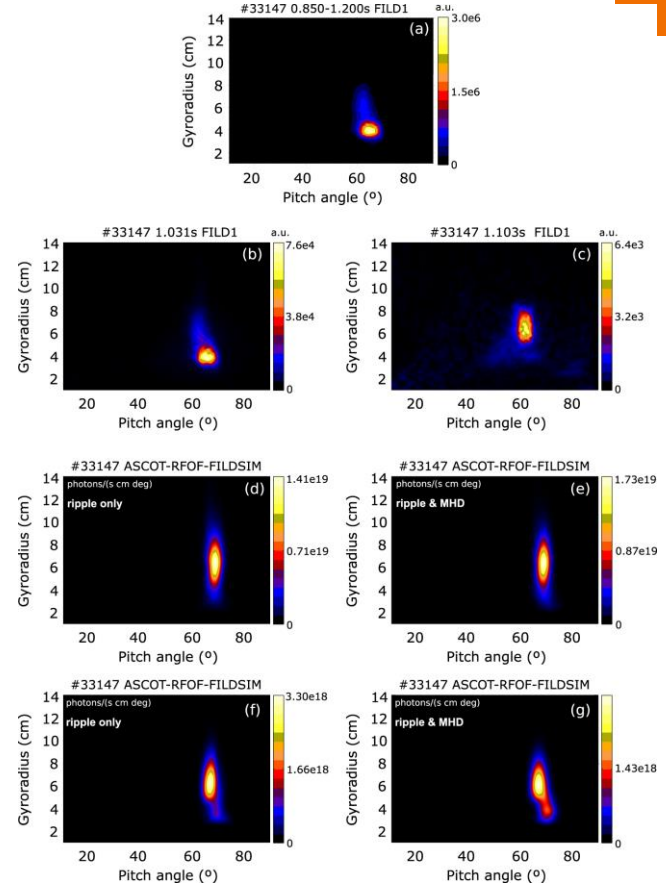


Simulation specifications

- 200k-500k GC markers
 - With MHD 200k, without 500k
 - Sampling: 3k FILD hits= \sim 38M markers
- Simulations in MARCONI fusion
- CPU costs:
 - Heating simulation for H, 55k CPUh
 - FILD simulation for H, 3k CPUh
 - From above: MHD cost=factor of 2.5

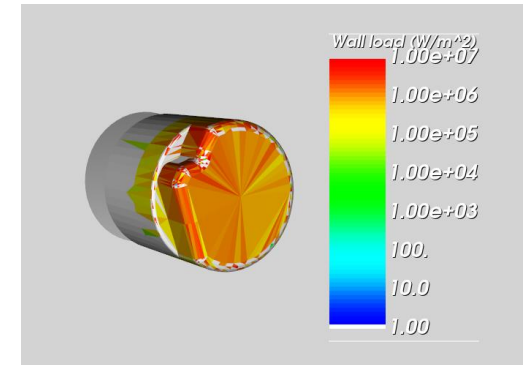
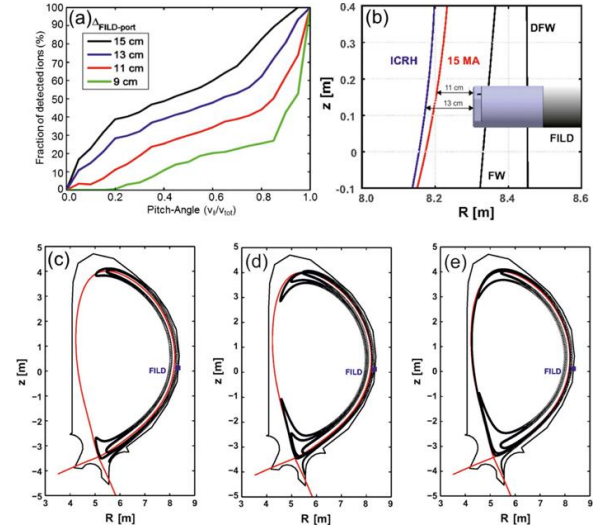
H

D



ITER FILD simulations

- Goal: simulate synthetic signal using ASCOT+FILDSIM
- Intrinsic issue: poor statistics of narrow-escape
 - First wall 600? m², FILD head 413 cm², the pinhole 1 cm²
 - Likelihood to hit pinhole (dummy math) 1/10M!!!
- Solution for production runs:
 - Brute-force
 - Running maximum number of markers
- Practical implication:
 - Scans for various scenarios, pinhole geometries etc.
 - Only use high statistics when needed
- Can we do better?
 - What if we could start our simulation from the pinhole...



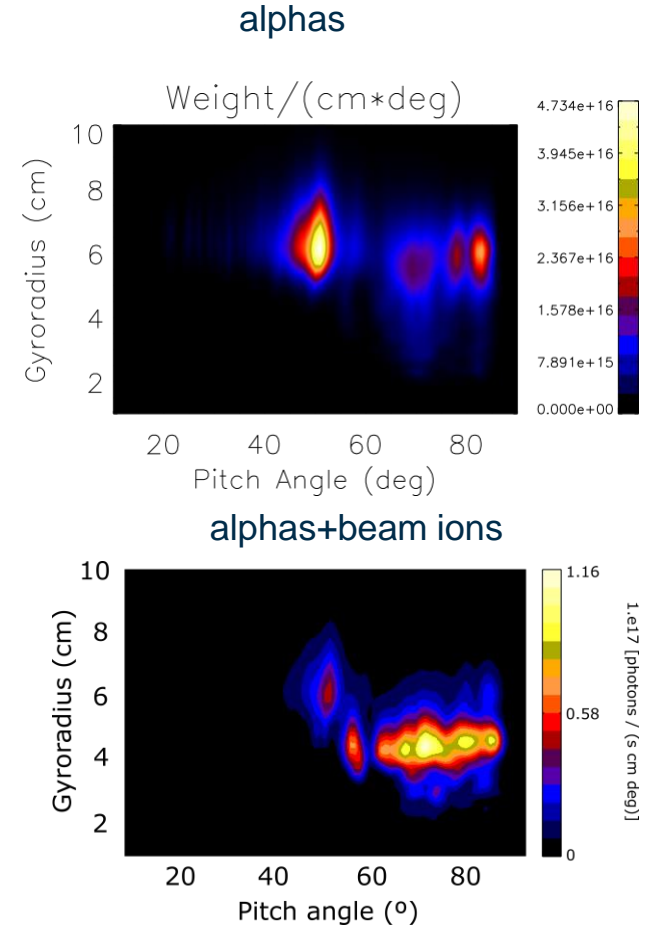
[8] M. Garcia-Munoz et al. *Rev. Sci. Instrum.* 87, 11D829 (2016)

[9] X. Litaudon, submitted to Nuclear Fusion 2023

ITER FILD simulations

- Goal: simulate synthetic signal using ASCOT+FILDSIM
- Intrinsic issue: poor statistics of narrow-escape
 - First wall 600? m², FILD head 413 cm², the pinhole 1 cm²
 - Likelihood to hit pinhole (dummy math) 1/10M!!!
- Solution for production runs:
 - Brute-force
 - Running maximum number of markers
- Practical implication:
 - Scans for various scenarios, pinhole geometries etc.
 - Only use high statistics when needed
- Can we do better?
 - What if we could start our simulation from the pinhole...

[9] X. Litaudon, submitted to Nuclear Fusion 2023





ACADEMY OF FINLAND

VTT

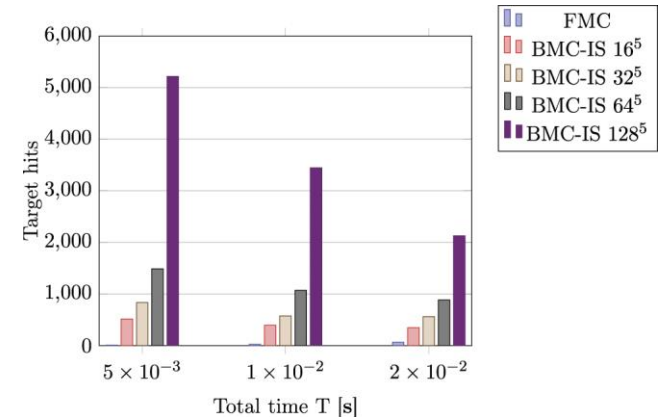
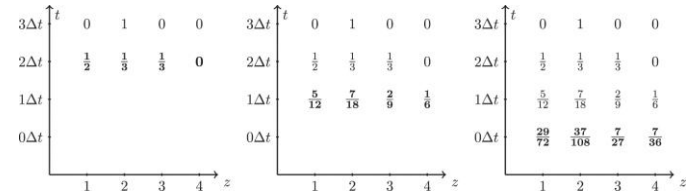
- Numerical models in nutshell
- Comparing typical runs
- Newest new: ASCOT-BMC, ASCOT-GPU
- Code development: git, slack, community

A. Snicker et al.

24/11/2023 VTT – beyond the obvious

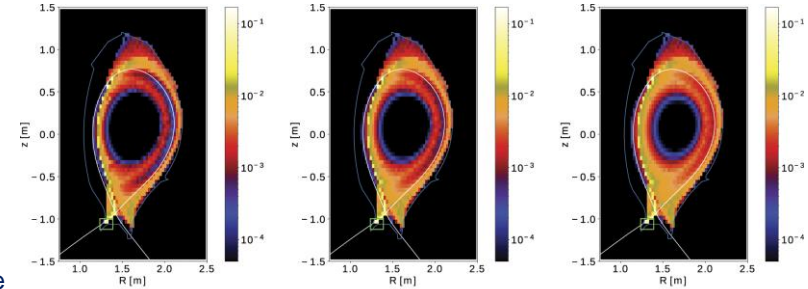
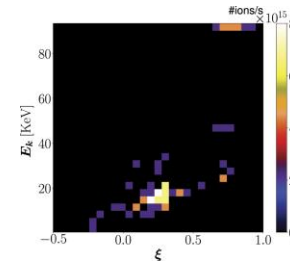
Backwards Monte Carlo

- Instead of brute force, use math and computers
- Iterative equations for probability distribution in phase-space
 - Allows to calculate the likelihood for marker from phase-space to the pinhole
 - Use known birth distribution, convolution of the two will give you a signal
 - Plan B adopted in this publication: use the likelihood to importance sample
- The efficiency increase by a factor of 10-100

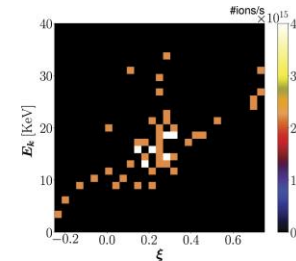


Backwards Monte Carlo

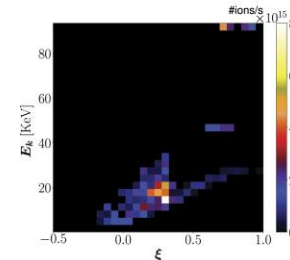
- Instead of brute force, use math and computers
- Iterative equations for probability distribution in phase-space
 - Allows to calculate the likelihood for marker in all phase-space to the pinhole
 - Use known birth distribution, convolution of the two will give you a signal
 - Plan B adopted in this publication: use the likelihood to importance sample
- The efficiency increase by a factor of 10-100
- Shown to reproduce forward model results
- Caveats:
 - Using 2D wall, 3D wall turns to be a nightmare...

a) BMC, $T = 5 \times 10^{-3}$ sb) BMC, $T = 1 \times 10^{-2}$ sc) BMC, $T = 2 \times 10^{-2}$ s

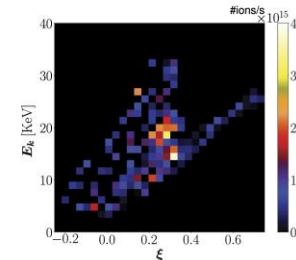
a) FMC



b) Zoomed-in view on FMC



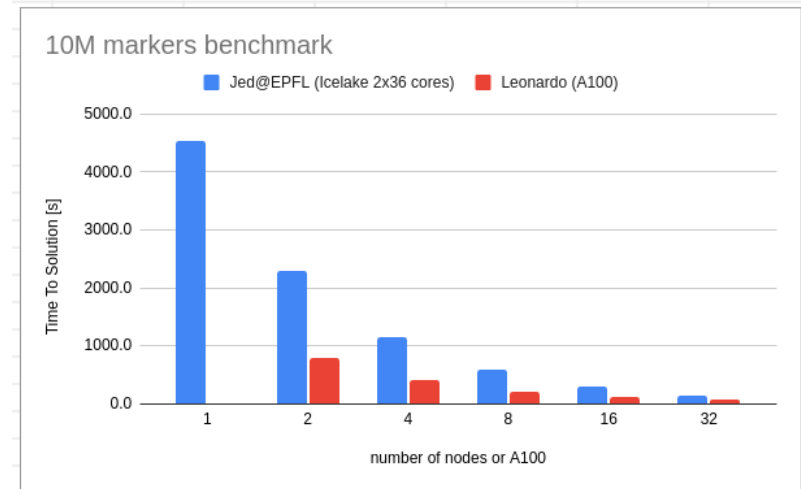
c) BMC-IS



d) Zoomed-in view on BMC-IS

ASCOT GPU

- Originally, ASCOT developed for Xeon Phi
- Can we directly use the same parallelism for GPU?
- How efficient the code will be?
- More details were given earlier today by G. Fourestey
- Next steps:
 - Check that the 3D wall and collisions work
 - Merge to main development branch
 - Try production runs?





- Numerical models in nutshell
- Comparing typical run times
- Newest new: ASCOT-BMC, ASCOT-GPU
- Code development: git, slack, community

A. Snicker et al.

24/11/2023 VTT – beyond the obvious

Ideology of constant development

- Adopting from fission, we need to constantly adapt
 - New supercomputers (the algorithms might not be efficient), e.g. GPU
 - New physics introduced (the code skeleton based on known physics), e.g. ICRH
 - Constant need to adapt to these!
- ASCOT has been built from the scratch twice, during last 10 years
 - ASCOT4 in FORTRAN and MPI around 2013
 - ASCOT5 in C and MPI/OpenMP around 2017
- A large userbase:
 - A set of tutorials to ease the onboarding process
 - Manuals and videos helping
 - Slack and weekly meetings

A new home for the ASCOT - github

- Since October 2023, ASCOT moved to open-source licensing
 - Code is operated under LGPL 3.0 license
 - Currently building the user community there
- Structure of the github:
 - Main branch always stable – only pull requests accepted, tags for new releases
 - Hotfixes done under main
 - Develop open for, well, developers
 - Developers can push, others still need pull request
 - New features under develop in a separate branch
 - Automated testing processes
 - Each push -> testing compilation+fast unit tests (~5min)
 - Pushing to develop -> above + physics tests+tutorials+documentation (~1h)
 - Pull request to main -> above + regression tests (WIP) (~a few days)
 - Each test can be run on will
 - Tutorials and documentation within github

Tutorials within the github - notebooks

Browser address bar: <https://ascot4fusion.github.io/ascot5/tutorials.html>

ASCOT

Search docs

RUNNING SIMULATIONS

- Installing
- Simulations

PROCESSING DATA


- Data
- Input generation
- Post-processing

SIMULATING PHYSICS

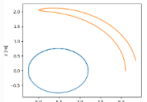
☰ Tutorials

- Introduction
- Visualizing orbits
- Dealing with distributions
- Creating fusion source with AFSI
- Modelling neutral beam injection with BBNBI
- Generating markers
- Simulating fast ion slowing-down process
- Estimating wall loads
- Simulating test particle response to

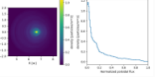
Basics



Introduction



Visualizing orbits



Dealing with distributions

Marker generation

NO THUMB NAIL

Creating fusion source with AFSI

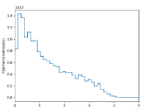
NO THUMB NAIL

Modelling neutral beam injection with BBNBI

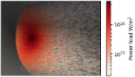
NO THUMB NAIL

Generating markers

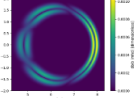
Physics



Simulating fast ion slowing-down process



Estimating wall loads



Simulating test particle response to MHD

NO THUMB NAIL

Modelling CX and tracing neutrals

Tutorials within the github - notebooks



Post-processing

- Live simulations
- Need help?
- Visualizing orbits
- Dealing with distributions
- Creating fusion source with AFSI
- Modelling neutral beam injection with BBNBI
- Generating markers
- Simulating fast ion slowing-down process
- Estimating wall loads
- Simulating test particle response to MHD
- Modelling CX and tracing neutrals
- Tracing markers backwards in time
- Generating Poincaré plots
- Creating 3D field from coil geometry with BioSaw

Physics in ASCOT5

PUBLICATIONS

- Citing ASCOT5
- Gallery

CODE DEVELOPMENT

- Parallelization

A

This example gives a general overview on how to pre- and postprocess ASCOT5 simulations.

1. First simulation: step-by-step
2. Contents of the HDF5 file
3. Python interface to libascot.so
4. Input generation
5. Post processing
6. Live simulations

First simulation: step-by-step

Go to `ascot5/doc/tutorials` folder and type `ipython3` to begin this tutorial. Then repeat these steps:

1. All pre- and post-processing is done via `Ascot` object. To create a new ASCOT5 data file, use `create=True`.

```
[1]: import numpy as np
from a5py import Ascot

a5 = Ascot("ascot.h5", create=True)
print("File created")

File created
```


Documentation within github

The screenshot shows a web browser window displaying the ASCOT5 documentation page. The browser's address bar shows the URL `https://ascot4fusion.github.io/ascot5/index.html`. The page layout includes a dark sidebar on the left with a search bar and a list of navigation links. The main content area features a title 'ASCOT5', a link to the GitHub repository, a paragraph describing the code's purpose, a 'Getting started' section with a numbered list of steps, and a 'Features' section with a paragraph of details. The Windows taskbar is visible at the bottom of the screen.

Search docs

[RUNNING SIMULATIONS](#)

- Installation
- Compiling on different platforms
- Settings when compiling
- Simulations

[PROCESSING DATA](#)

- Data
- Input generation
- Post-processing

[SIMULATING PHYSICS](#)

- Tutorials
- Physics in ASCOT5

[PUBLICATIONS](#)

- Citing ASCOT5
- Gallery

[CODE DEVELOPMENT](#)

- Parallelization
- Testing
- For Developers
- C API

<https://ascot4fusion.github.io/ascot5/physics.html>

Home / ASCOT5 [View page source](#)

ASCOT5

<https://github.com/ascot4fusion/ascot5>

ASCOT5 is a test-particle orbit-following code for solving minority species' distribution functions, transport, and losses in tokamaks and stellarators. For questions related to the code or physics, please join our [Slack channel](#).

Getting started

1. Follow the [installation](#) instructions and [compile](#) the code.
2. Have some quality time by going through [the introductory simulation](#).
3. Familiarize yourself on how to generate [inputs](#) that you need, execute [simulations](#), and post-process [the results](#). Here [the examples](#) and [the physics](#) documentation as well as [the Python API](#) are good sources of help.
4. At some point you might also want to [publish](#) your work or [contribute](#) to the code.

Features

ASCOT5 is a test-particle orbit-following code for computing particle orbits in 3D geometry. The output includes particle orbits, phase-space distributions, transport coefficients, and wall loads. ASCOT5 is frequently applied to study fast ions, impurities, neutrals, and runaway electrons in tokamaks and stellarators. Particle orbits are either solved fully, i.e. including the gyro-motion, or in a reduced picture where only the guiding-center trajectory is traced. The code is extensively parallelized and optimized to support simulations with more than ten million markers.

Building the user community

- EUROfusion funded training camps
 - 1st was organized in 2019 (~12 participants)
 - 2nd last week (~25 participants)
- ASCOT is a global project
 - West (Europe, US) and EAST (China)
 - North (Finland)
 - Need some users from Australia or from South-America to cover global South, anyone?
- Casual discussions using slack
- Weekly meetings via zoom

Building the user community



bey⁰nd

the obvious

A. Snicker

How to keep up with fast ions in the increasingly complex fusion devices?

Taina Kurki-Suonio & Antti Snicker
Aalto University & VTT

Contents

- ★ What are the ***fast ions*** in fusion world?
- ★ Going from pen&paper to simulations requiring supercomputing
 - Axisymmetric, circular plasmas (from pen&paper to analytical models)
 - Real-life tokamaks: introduce a variety of mechanism breaking the axisymmetry
 - The ultimate case: *stellarator*
 - Make contact with the outside world: introduce SOL and the 3D wall structures
 - Accurate power distributions on the first wall: from GC following to resolving gyro orbits
 - Realistic (non-quiescent) plasmas: introduce NTMs, TAEs, turbulence ...

Fast ion in fusion devices

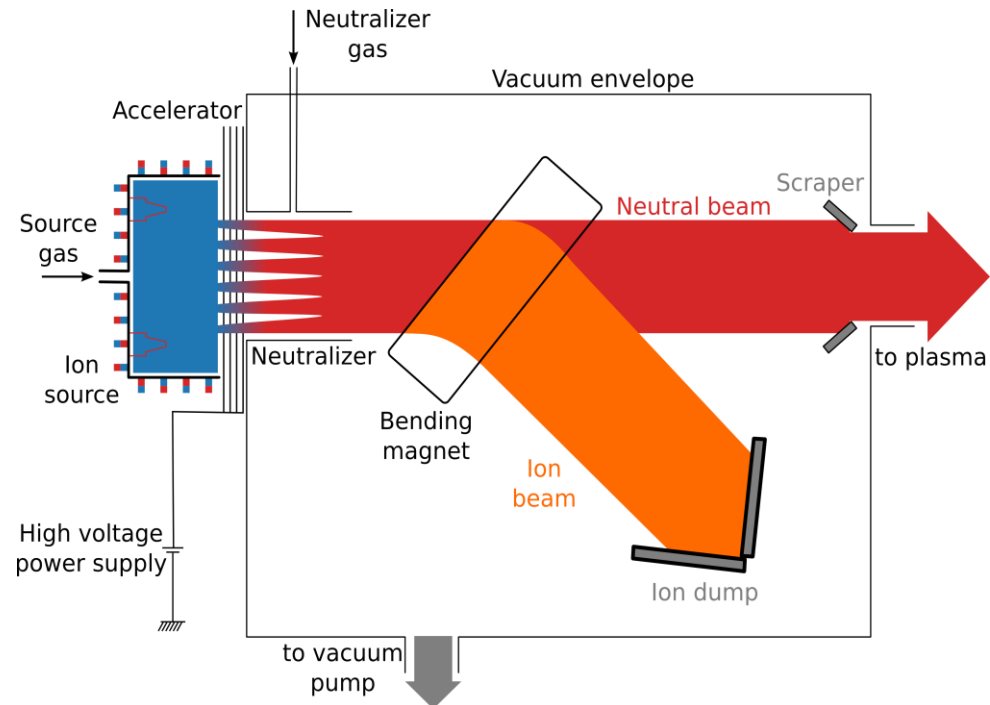
- ★ The fusion plasma as a whole is hot, >10 keV
- ★ To keep it hot, we need to have particles with even higher energies that collisionally heat the fuel plasma to compensate for the inevitable losses'

These particles are the *fast ions*

- ★ In today's devices, fast ions are generated externally
- ★ In a fusion reactor, the fast ions are generated by the fusion reactions themselves, and the fusion conditions are self-sustained ...

Generating fast ions externally

- ★ ICRH → minority ions in MeV range
- ★ Neutral beams:
 - PNBI → D (~100 keV)
 - NNBI → D (~1 MeV)

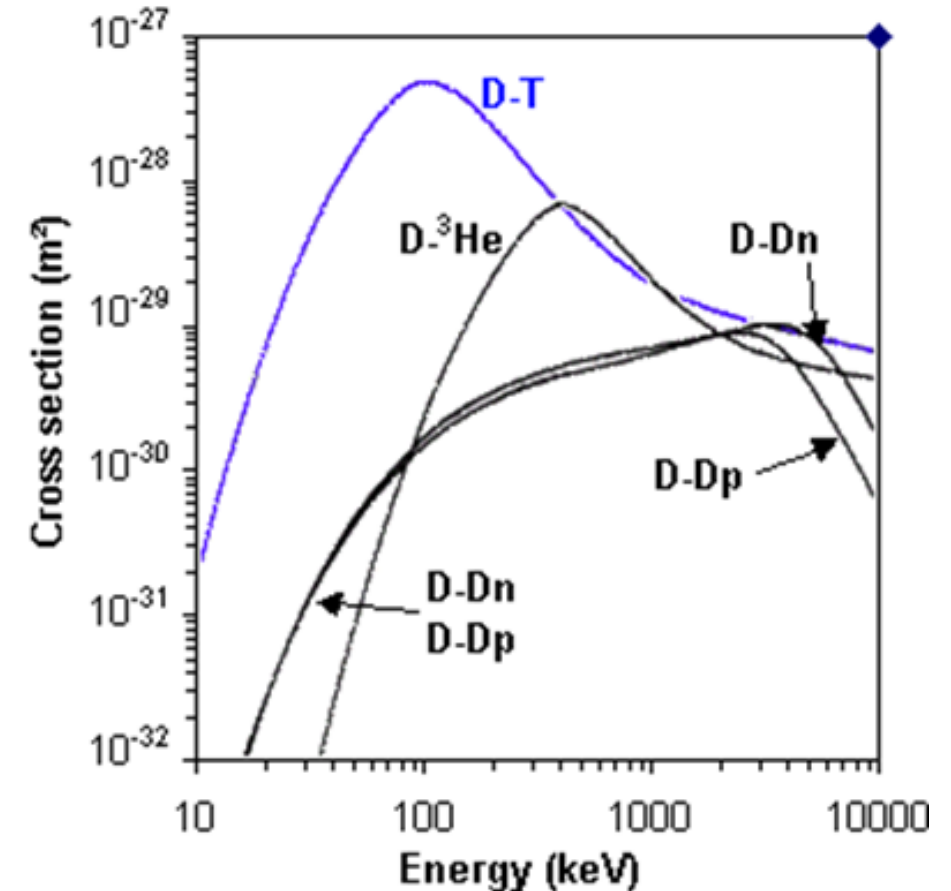


JET ITER-like antenna

Fast ions from fusion reactions

★ α 's, p's, T's & ^3He 's in MeV range from fusion reactions – now and in the future:

- $\text{D} + \text{D} \rightarrow ^3\text{He} (0.8 \text{ MeV}) + \text{n} (2.45 \text{ MeV})$
- $\text{D} + \text{D} \rightarrow \text{T} (1 \text{ MeV}) + \text{p} (3 \text{ MeV})$
- $\text{D} + \text{T} \rightarrow \alpha (3.5 \text{ MeV}) + \text{n} (14.1 \text{ MeV})$
- $^3\text{He} + \text{D} \rightarrow \alpha (3.6 \text{ MeV}) + \text{n} (14.7 \text{ MeV})$



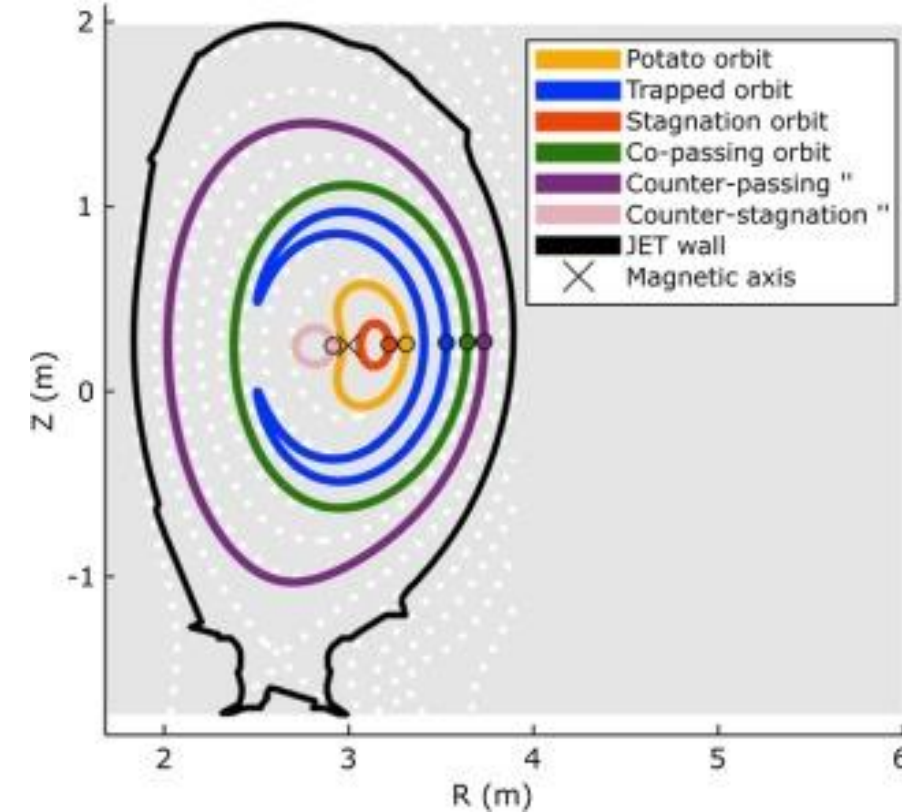
Confining fast ions

Axisymmetric tokamak

★ 'perfect' confinement

- Mathematically guaranteed: according to *Noether's theorem* any symmetry is associated with a constant of motion
- In axisymmetric tokamak this is the *toroidal canonical momentum*
- Axisymmetry ensures that the particle *drift orbits* close upon themselves and do not wander radially

★ Only Coulomb collisions slowly kick the ions outward

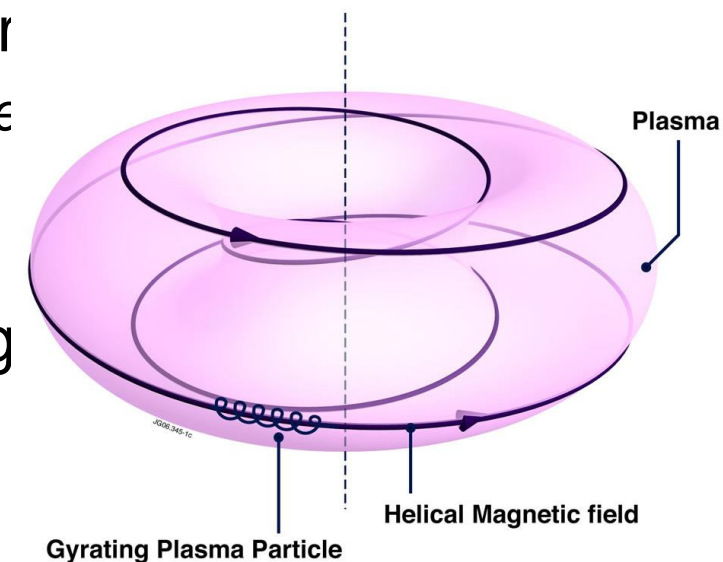


Benjamin et al., Computer Physics Communications 292 (2023) 108893

Simulating fast ions in axisymmetric tokamak...

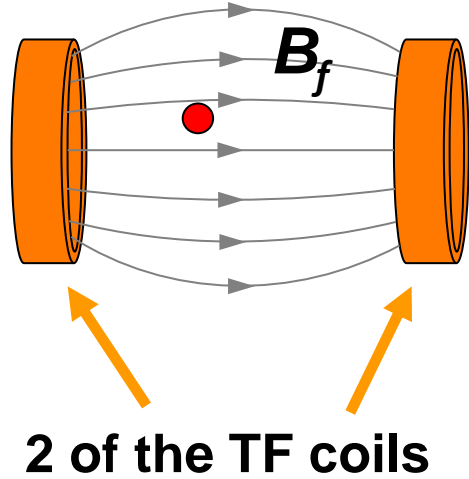
... is very easy and fast!

- ★ The toroidal magnetic geometry can be expressed *analytically*
 - No numerical divergence
 - No need for interpolations
- ★ One can use *field-aligned flux coordinates* in following par
 - Long time steps allowed → Integrating equations of motion for the guiding-centers is very fast
- ★ This approach (ASCOT 1.0 and 2.0) was ok for assessing the zeroth-order effects due to Coulomb collisions:
 - particle "confinement" (staying inside separatrix)
 - power deposition



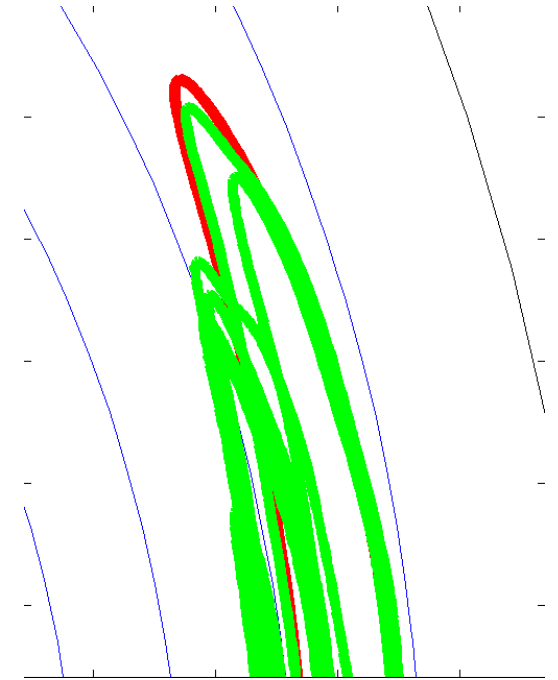
Real tokamaks and real needs, culminating to stellarators...

The axisymmetry is broken: Finite number of coils with finite size ...



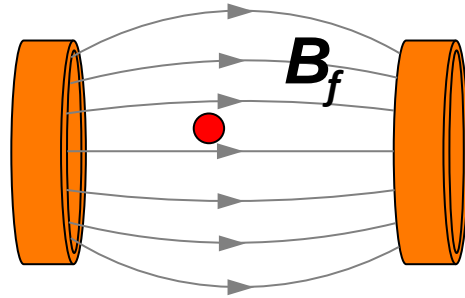
A finite number of TF coils
→ non-axisymmetric field, **toroidal magnetic ripple**

The local magnetic “bottle” between two TF coils can trap charged particles, which quickly drift out of the plasma due to vertical grad-B drift.

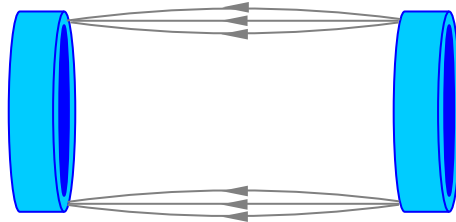


Banana orbits are no longer guaranteed to close in the poloidal plane and can start wandering even without collisions ...

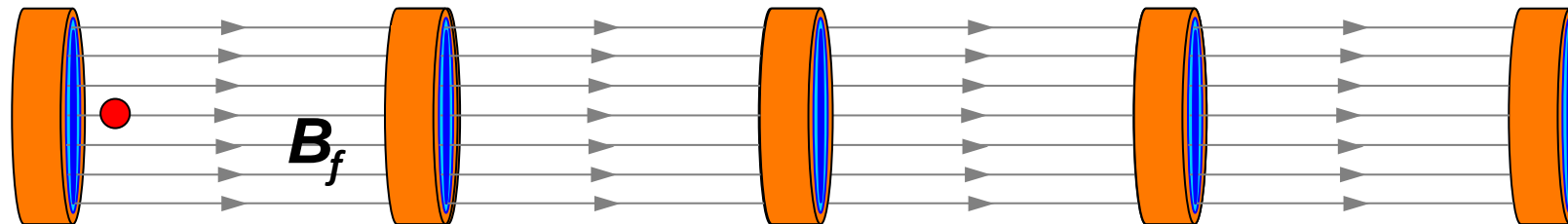
Trying to fix the broken symmetry: *ferritic inserts*



+

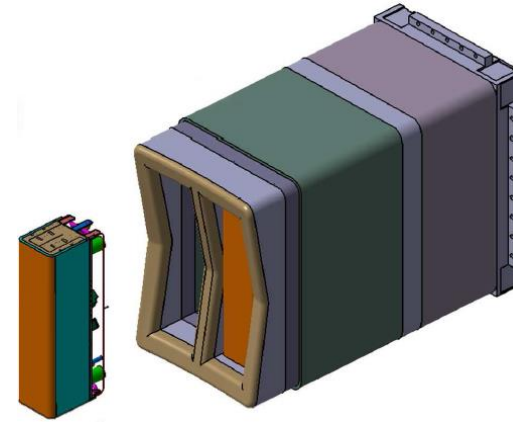
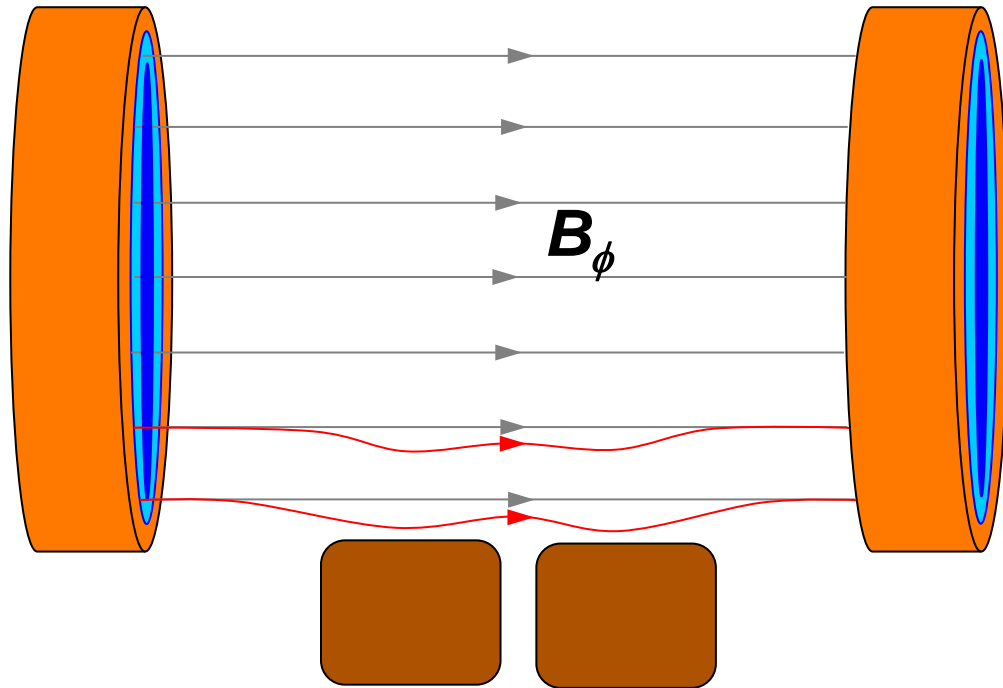


=



With ferromagnetic steel inserts placed at the coils, the ripple can be minimized.

But that's not all, folks: *Test Blanket Modules (TBM)*



TBM's containing ferritic steel very close to the plasma are placed at three toroidal locations between TF coils

Let us count our symmetry breakers...

- ★ Finite # of TF coils → TF ripple
- ★ Ferromagnetic components → localized magnetic perturbations
- ★ TBM blocks (or any other material sucking in magnetic field)
- ★ External coils, such as ELM control coils → stochastization of edge magnetic field

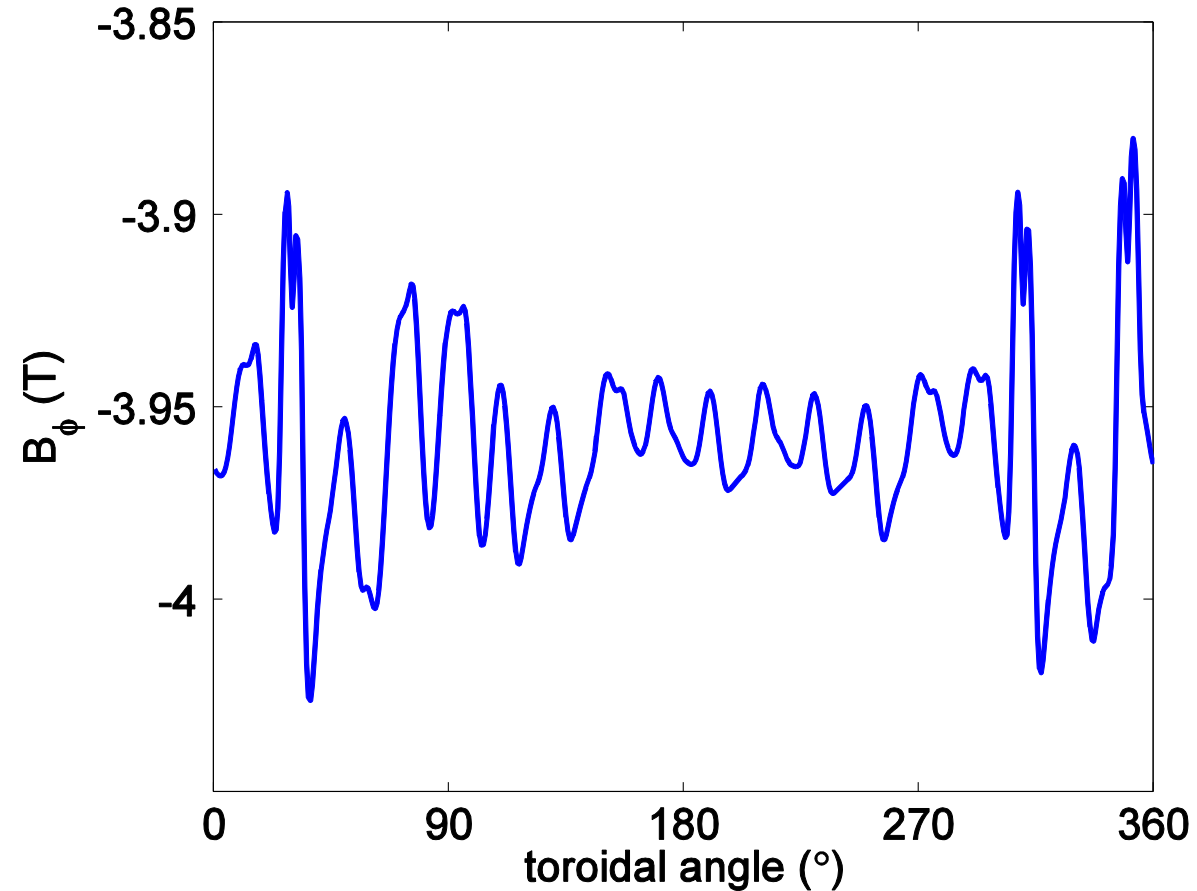
- ★ How does the total magnetic field look like?

$B_T(\varphi)$ at the OMP separatrix in ITER 9MA Scenario

Toroidal ripple
1.1%,

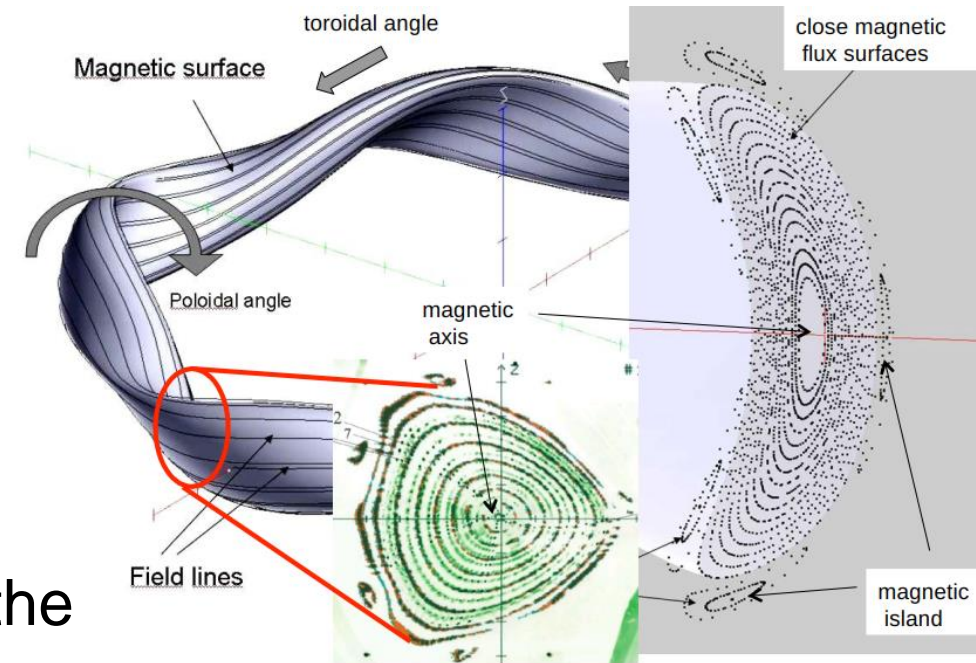
Field bump due
to NBI ports
0.57%

Field bump due
to TBMs
1,1%



What are the implications to our simulations?

- ★ Need to tabulate the field & interpolate
 - enormous increase in memory needs
 - good-bye kiss to field-aligned flux coordinates
 - we have to keep a keen eye not to have
 - Numerical divergence in the field
 - Numerical drifts
- ★ Fine structures have to be seen and obeyed by the fast ions
 - time step has to be shortened



Prime example of ultimate 3D features: *W7-X stellarator*

And as if that was not enough...

- ★ With a large number of fast ions (in ITER, fast ions account for about 1/3 of the total pressure), one has to worry about power loads to the first wall
 - Both the peak power load evaluations and synthetic lost-ion diagnostics require a **high-fidelity first wall**
 - For well-confined (= relevant plasmas), a very large particle ensemble is needed to yield **reasonable statistics** at the wall
 - parallelizations, vectorizations, GPUs.... (this is why ASCOT5 was born)
- ★ ASCOT only includes neoclassical physics, while real plasmas have much more character: *turbulence, NTMs, TAEs, ...*

Including additional physics always has a computational cost – either in CPU/GPU time or memory consumption. Or both...

And now to Antti ...