IMAS-ification of TRANSP: generalization of interfaces to data and physics modules

A.Y. Pankin, J. Breslau, M. Goliyad, M. Gorelenkova,
F. M. Poli, G. Perumpilly, J. Sachdev

Princeton Plasma Physics Laboratory, Princeton, NJ, 08543

65th Annual Meeting of the APS Division of Plasma Physics

October 29-November 3, 2023 • Denver, CO

TRANSP: Path from past to future – New Challenges

TRANSP is a code that is used worldwide for interpretive and predictive analysis of tokamak discharges

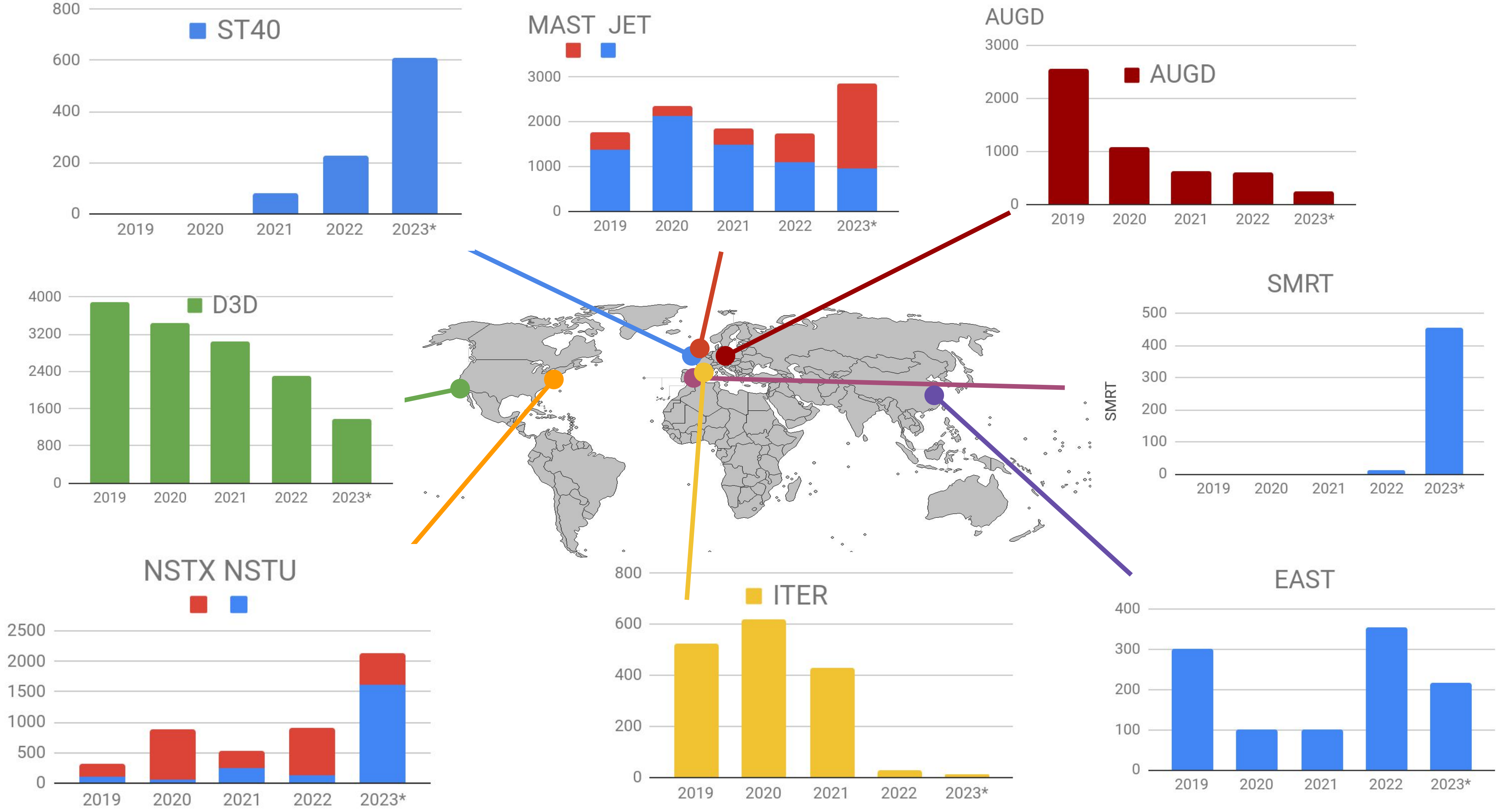
There is ongoing transition to IMAS interface

Challenges of the IMAS transition

- Ensure the code stability
- Changes to the code interface/initialization procedure need to be reported to the users well in advance and documented
- Back compatible as much as we can

TRANSP is a Fusion Interpretive and Predictive Integrated Modeling Code Used Worldwide

TRANSP usage 2019-2023



There is an increasing number of runs that are performed using Containers. These runs are not accounted in this statistics.

TRANSP is a Fusion Interpretive and Predictive Integrated Modeling Code Used Worldwide

Almost 300 (299) users at the PPPL cluster and approximately 50 users of container in China

Researchers from over 30 institutions submitted TRANSP runs

We expect increase of TRANSP runs in 2023

For the ten months of 2023, the total number of TRANSP runs increased by ~50% compared to the same period of 2022:

- Total number of runs increased from 6036 (4575 parallel) to 8348 (5134 parallel)
- For MAST # runs increased from 442 to 1904
- For ST40: from 225 to 608

Efforts on the Modernization of the TRANSP Code

TRANSP is maintained at github with version control

https://github.com/PrincetonUniversity/transp

- **Regression tests using CI**
- **Two versions available for user**
  - pshare: stable version of TRANSP
  - tshare: developer version of TRANSP for testing by users

Modules not developed at PPPL are moved from TRANSP as external modules

- **For external modules with established interfaced, it is easy to updated**
- **Regression tests can be handled separately**
- **These modules include:**
  - TLGF, GLF23, MMM, NEO, Genray, CQL3D, Toray, Torbeam

Stable well established PPPL modules are being moved as external modules

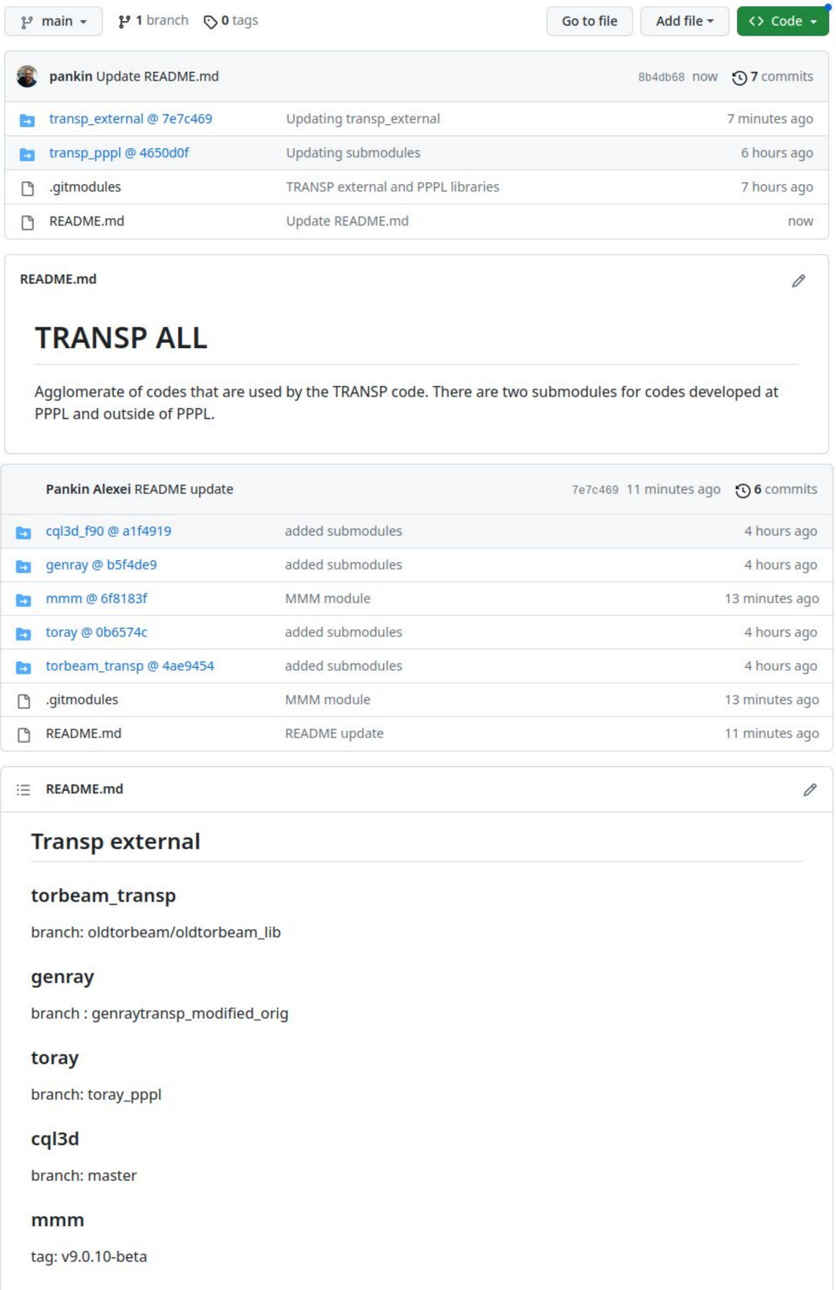
- **The modules include:**
  - PSPLINE, UFILES, PLASMA_STATE, EZCDF, NUBEAM, PT_SOLVER, SIGSUB

Components of the TRANSP code are being tracked from one place

New transp_all repository is being tested

Has two submodules

- **transp_external**
  - TORAY
  - TORBEAM
  - GENRAY
  - CQL3D
  - MMM
- **transp_pppl**
  - TRANSP
  - TRANSP_TEST_HARNESS
  - EZCDF
  - PLASMA_STATE
  - UFILES
  - PSPLINE
  - FRANTIC
  - NUBEAM
  - SIGSUB



pankin Update README.md		8b4db68 now 7 commits
transp_external @ 7e7c469	Updating transp_external	7 minutes ago
transp_pppl @ 4650d0f	Updating submodules	6 hours ago
.gitmodules	TRANSP external and PPPL libraries	7 hours ago
README.md	Update README.md	now

README.md

TRANSP ALL

Agglomerate of codes that are used by the TRANSP code. There are two submodules for codes developed at PPPL and outside of PPPL.

Pankin Alexei README update		7e7c469 11 minutes ago 6 commits
cql3d_f90 @ a1f4919	added submodules	4 hours ago
genray @ b5f4de9	added submodules	4 hours ago
mmm @ 6f8183f	MMM module	13 minutes ago
toray @ 0b6574c	added submodules	4 hours ago
torbeam_transp @ 4ae9454	added submodules	4 hours ago
.gitmodules	MMM module	13 minutes ago
README.md	README update	11 minutes ago

README.md

Transp external

torbeam_transp

branch: oldtorbeam/oldtorbeam_lib

genray

branch : genraytransp_modified_orig

toray

branch: toray_pppl

cql3d

branch: master

mmm

tag: v9.0.10-beta

Components of the TRANSP code are being tracked from one place

New transp_all repository is being tested

Has two submodules

- transp_external
  - TORAY
  - TORBEAM
  - GENRAY
  - CQL3D
  - MMM
- transp_pppl
  - TRANSP
  - TRANSP_TEST_HARNESS
  - EZCDF
  - PLASMA_STATE
  - UFILES
  - PSPLINE
  - FRANTIC
  - NUBEAM
  - SIGSUB

What if external module is not available or agreement form has’t been signed?

– We can disable them at the compilation time:

- export NO_TORAY=1
- export NO_GENRAY=1
- export NO_TORBEAM=1
- export NO_CQL3D=1
-
- export NO_LSCSQ=1
- export NO_PTSOLVER=1
-
- unset IMAS_ON

Uniform approach to setting EC/IC modules

Electron Cyclotron calculations are activated by setting NLECH=.T.. The physics model for EC calculations is set by the flag EC_MODEL.

Three modules for the calculation of propagation of waves in the range of frequencies of the Electron Cyclotron resonance:

- TORAY-GA (ray-tracing)
- GENRAY (ray tracing)
- TORBEAM (beam tracing)

The ray tracing code GENRAY has capabilities for calculating also EBW, Lower Hybrid, Helicon waves and High Harmonics Fast Waves

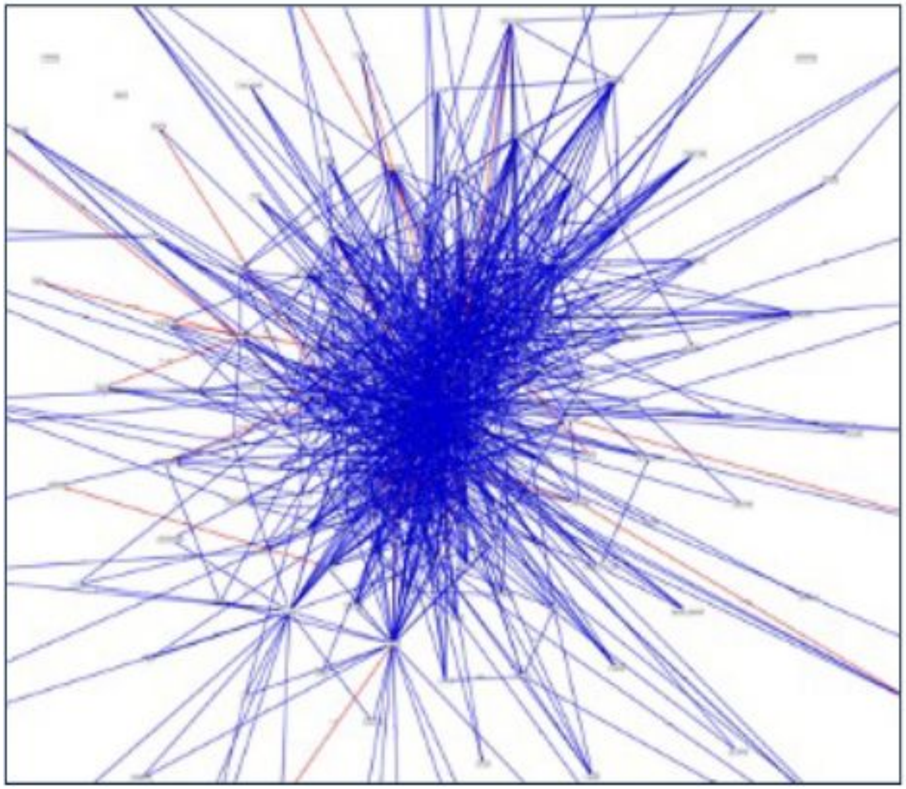
- GENRAY. To activate this model, set EC_MODEL='GENRAY'
- TORAY-GA. To activate this model, set EC_MODEL='TORAY'
- TORBEAM. To activate this model, set EC_MODEL='TORBEAM'

When EC/IMAS module is full activated, we will have another option

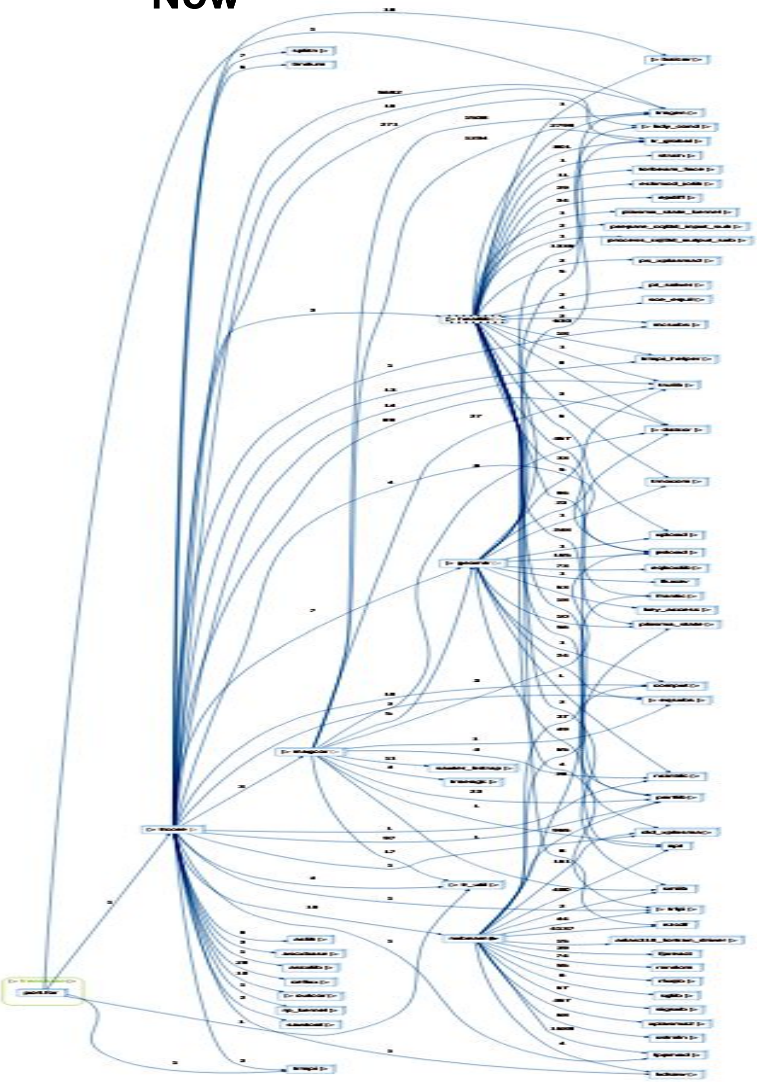
- EC_MODEL='IMAS' and specific name of the model is set from corresponding IDS

Analysis of dependencies in TRANSP performed with the Understand software by SciTool

Before

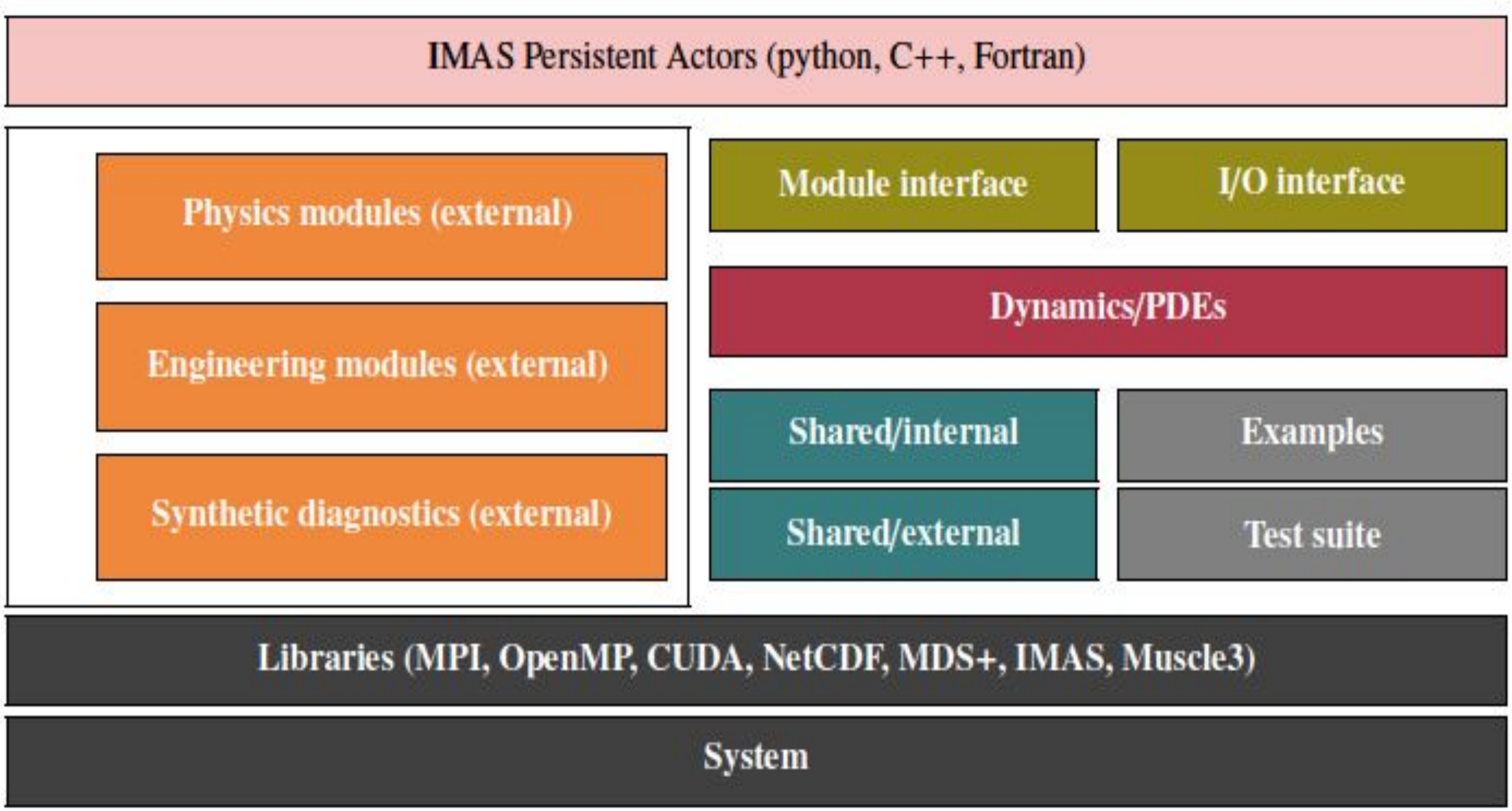


Now



Improvements are made by separating sections of the code in modules, reorganization of variables defined in a global module into separate/smaller modules, removing outdated and unused code

Architecture redesign to enable community contributions


IMAS Persistent Actors (python, C++, Fortran)
Physics modules (external)
Engineering modules (external)
Synthetic diagnostics (external)
Module interface
I/O interface
Dynamics/PDEs
Shared/internal
Examples
Shared/external
Test suite
Libraries (MPI, OpenMP, CUDA, NetCDF, MDS+, IMAS, Muscle3)
System

Design inspired by the climate modeling community

IMAS (ITER Integrated Modeling and Analysis Suite) options in TRANSP

IMAS [F. Imbeaux *et al.* Nucl.Fusion 55 (2015) 123006] establishes standards for fusion data to facilitate coupling of codes and physics

IMAS is implemented in TRANSP

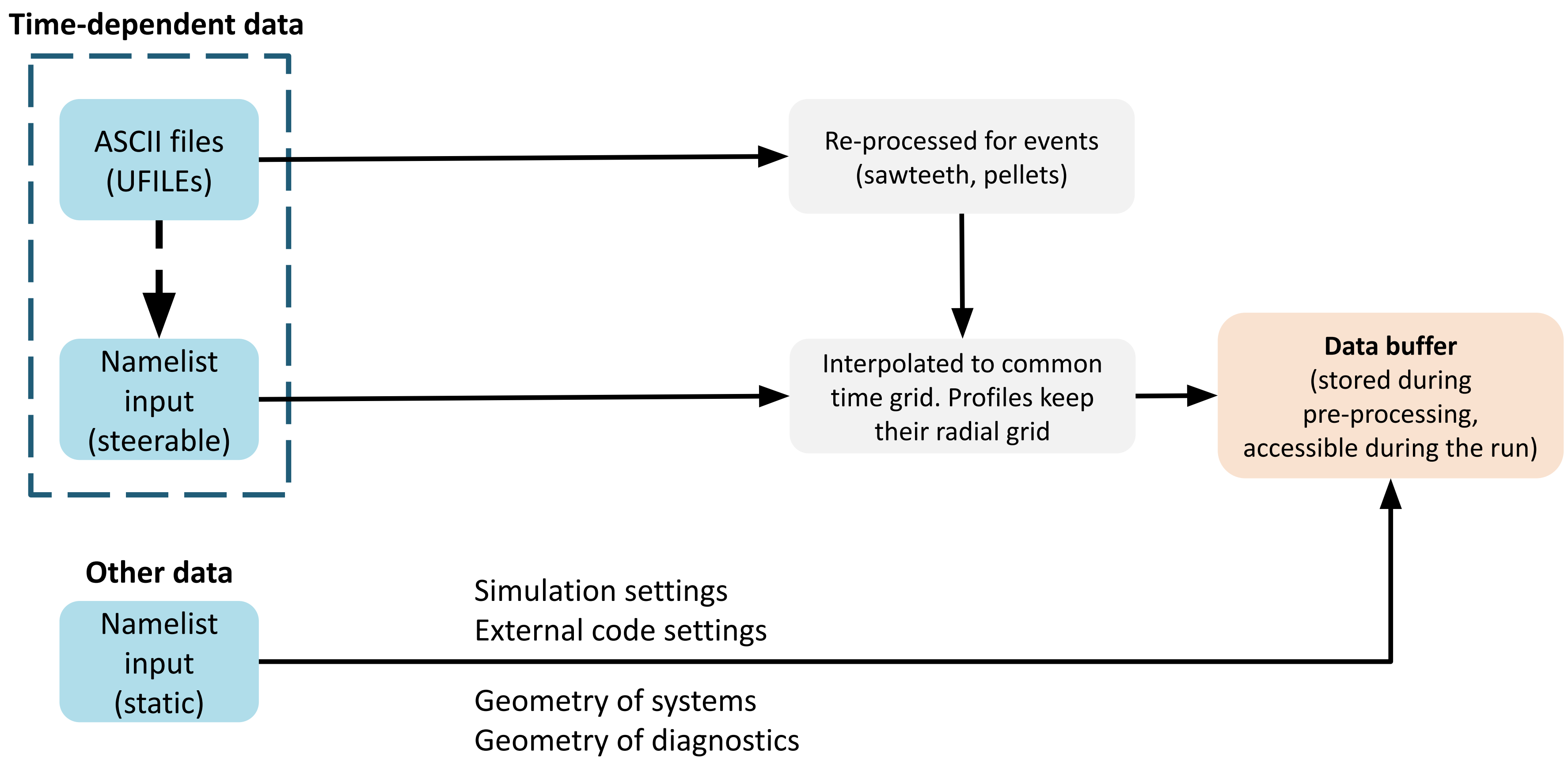
- **Python tool *transp2imas* that converts CDF TRANSP output to IMAS**
  - The translator does not write geometry of hardware from TRANSP namelist
    - Information comes from IDS/machine_description database
      - Done to avoid propagation of incorrect settings in TRANSP runs)
- **TRANSP can directly save output to IMAS during runtime**

TRANSP can be initialized with IMAS data

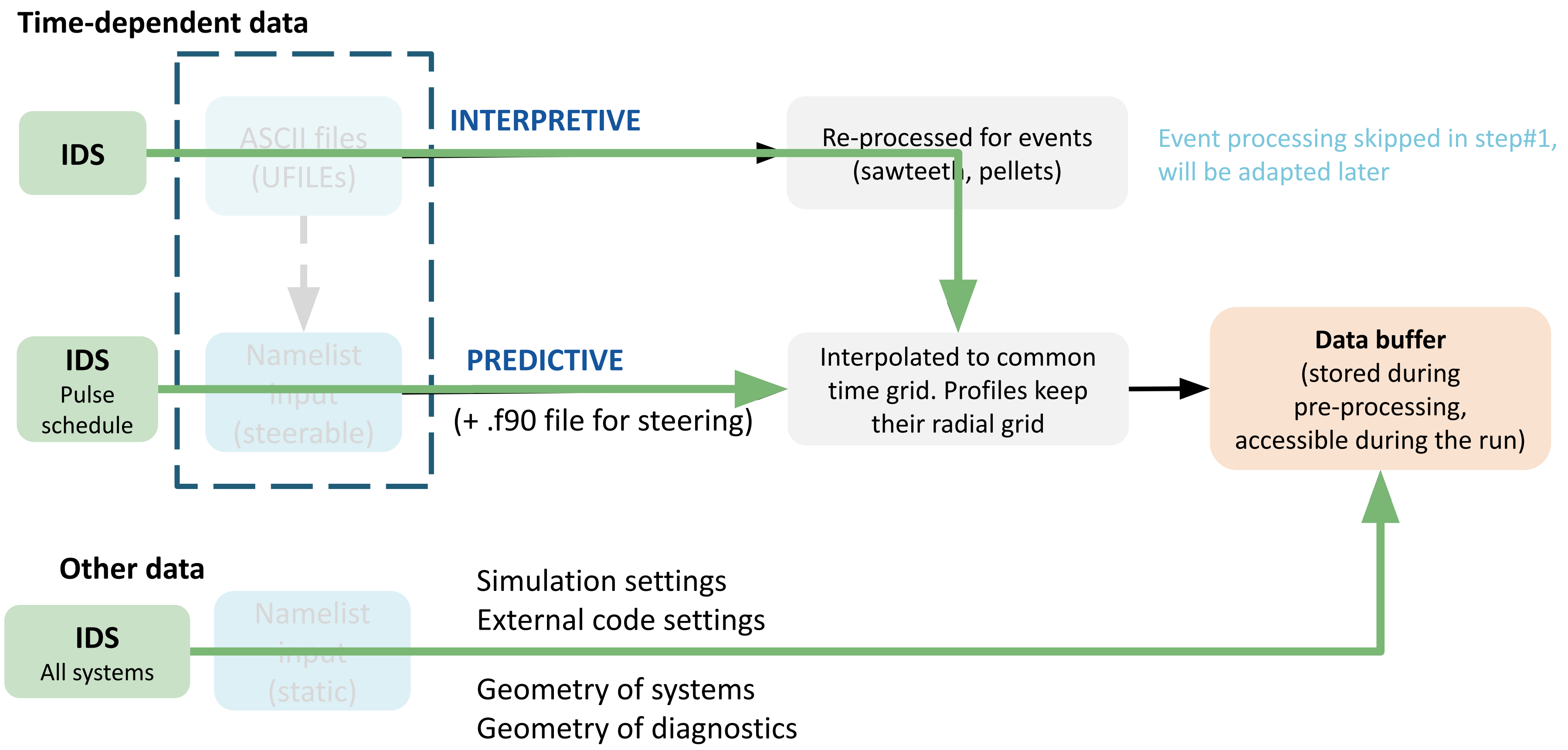
- **By reading external IDSs**
  - Subset of IMAS dataset needed for a particular run
    - Done with trdat pre-processing tool
    - Python converter ***mds2ids_nstu*** for subset of MDS NSTX/NSTX-U database to imas is developed
- **By pre-processing of IMAS data to**
    - Standard TRANSP input files
      - Done with ***imas2transp*** pre-processing tool

Selected output IMAS options are directly added to TRANSP

Making TRANSP more modular important step on the IMAS implementation in TRANSP


Time-dependent data
ASCII files (UFILEs)
Namelist input (steerable)
Re-processed for events (sawteeth, pellets)
Interpolated to common time grid. Profiles keep their radial grid
Data buffer
(stored during pre-processing, accessible during the run)
Other data
Namelist input (static)
Simulation settings
External code settings
Geometry of systems
Geometry of diagnostics

Making TRANSP more modular important step on the IMAS implementation in TRANSP



Time-dependent data

IDS

ASCII files (UFILEs)

INTERPRETIVE

Re-processed for events (sawteeth, pellets)

Event processing skipped in step#1, will be adapted later

IDS Pulse schedule

Namelist input (steerable)

PREDICTIVE

(+ .f90 file for steering)

Interpolated to common time grid. Profiles keep their radial grid

Data buffer (stored during pre-processing, accessible during the run)

Other data

IDS All systems

Namelist input (static)

Simulation settings
External code settings

Geometry of systems
Geometry of diagnostics

IMAS (ITER Integrated Modeling and Analysis Suite) options in TRANSP

IMAS establishes standards in coupling of fusion codes → IMAS-ready module can be easily added to the TRANSP code

Coupling option using IMAS has been tested for FAR3D/TRANSP coupling

- TRANSP profiles needed by FAR3D are stored in equilibrium and core_profiles IDSs
- FAR3D reads these IDSs and analysis the profile stability
- FAR3D/TRANSP coupling using IMAS has been verified by comparison of the stability results obtained using conventional interface

Interfaces for the coupling of EC/IC/NBI modules that support IMAS are being developed

NUBEAM code optimization on GPUs

[M. Gorelenkova *et al*. NP11.00070]

DIII-D Super-H mode discharge 174783 during high ion mode with dynamically changing parameters is selected

- Dynamic is followed in interpretive analysis with TRANSP-NUBEAM

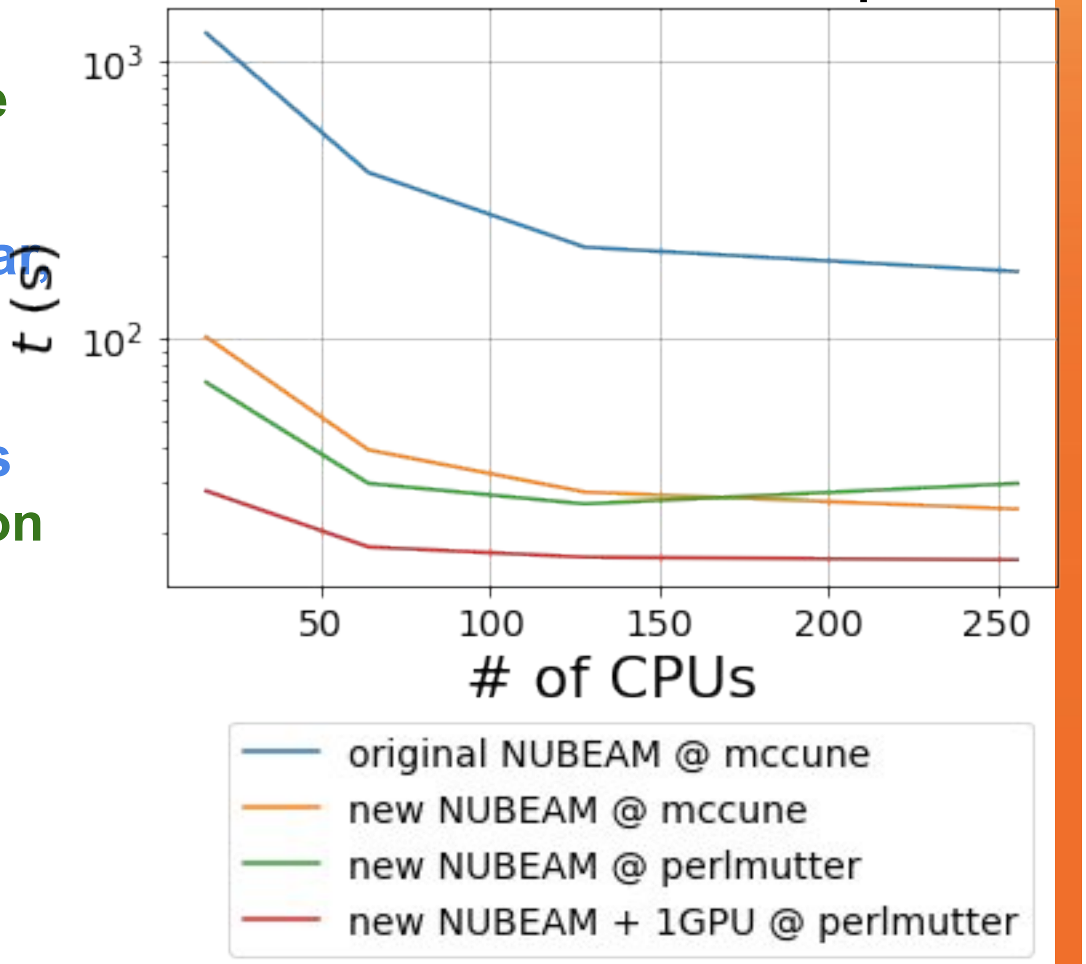
With all optimization work done so far, NUBEAM simulation with 160k MC particles remains within 16 sec and is suitable for between-shot-analysis

- Code refactoring and modernization yields speedup of more than x 12
- Use of more modern computer architecture results in additional speedup of x 1.4
- Use of one GPU yields additional factor of x 2

This scaling does not take into account secondary neutrals due to CX process

- Still work in progress

Simulation for DIII-D discharge 174783 between 1.8 and 2s with 160k MC particles



Saturation in scaling is related to the optimal # of MC per CPU and GPU

NUBEAM code optimization on GPUs [M. Gorelenkova *et al*. NP11.00070]

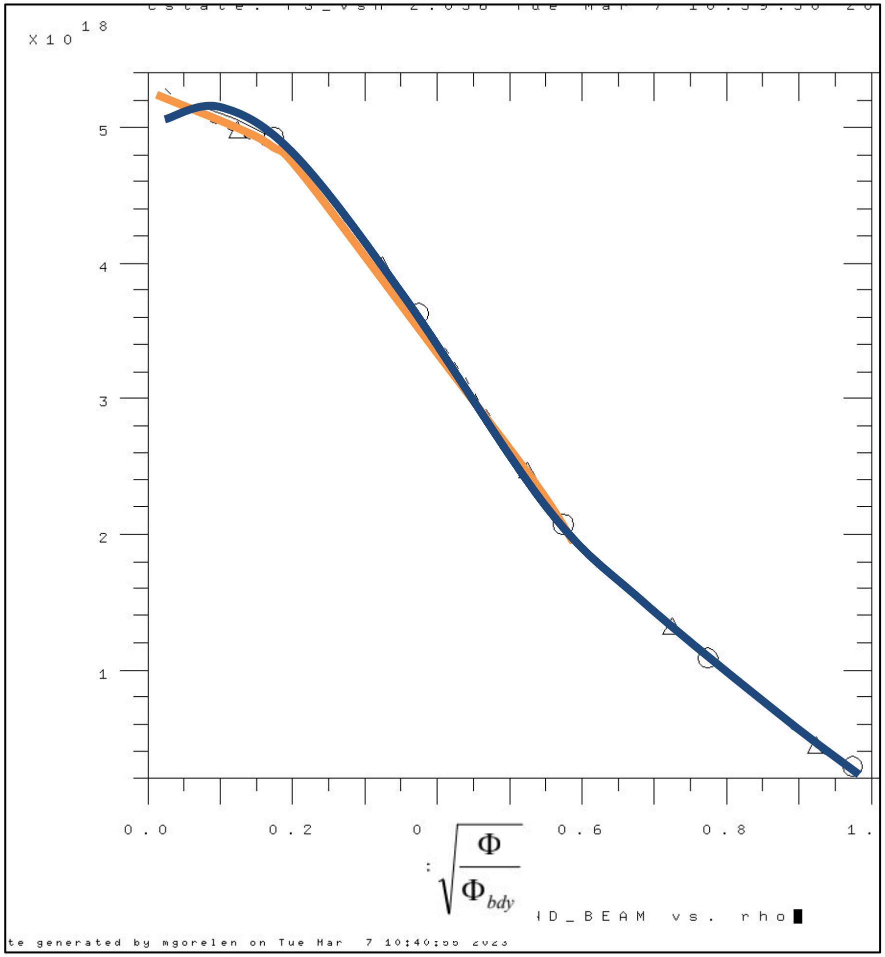
🟧 OpenMPI version

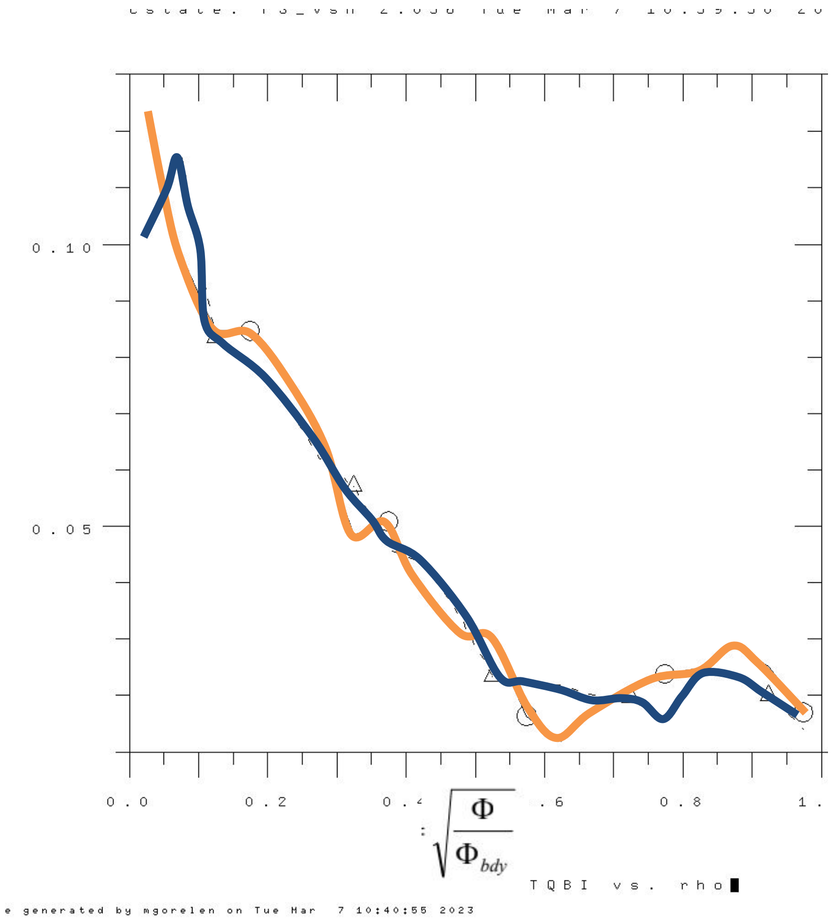
🟦 OpenACC version

Beam ion density

OpenMPI version

Torque TQBI





Predictive TRANSP: V&V TRANSP and ASTRA for DIII-D discharges

Predictive simulations of 218 randomly selected DIII-D discharges performed using TGLF+NUBEAM in TRANSP and TGLF+RABBIT in ASTRA.

Default settings are used – No additional tuning

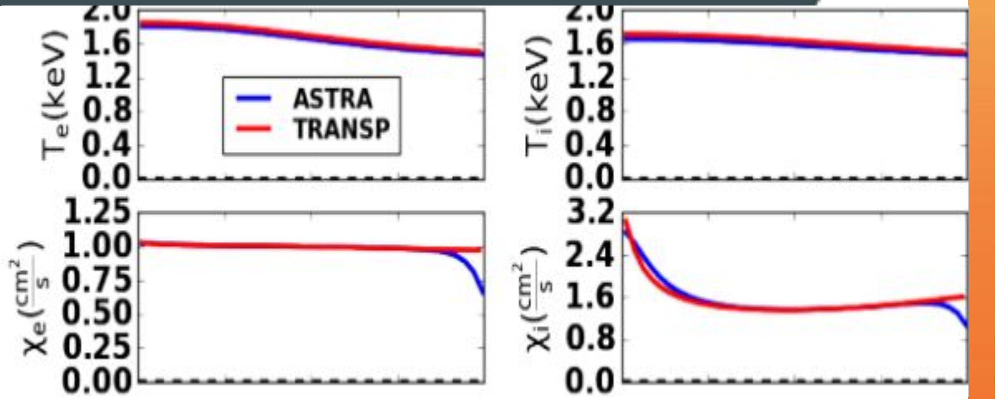
- 38 failed in TRANSP
- 79 failed in ASTRA

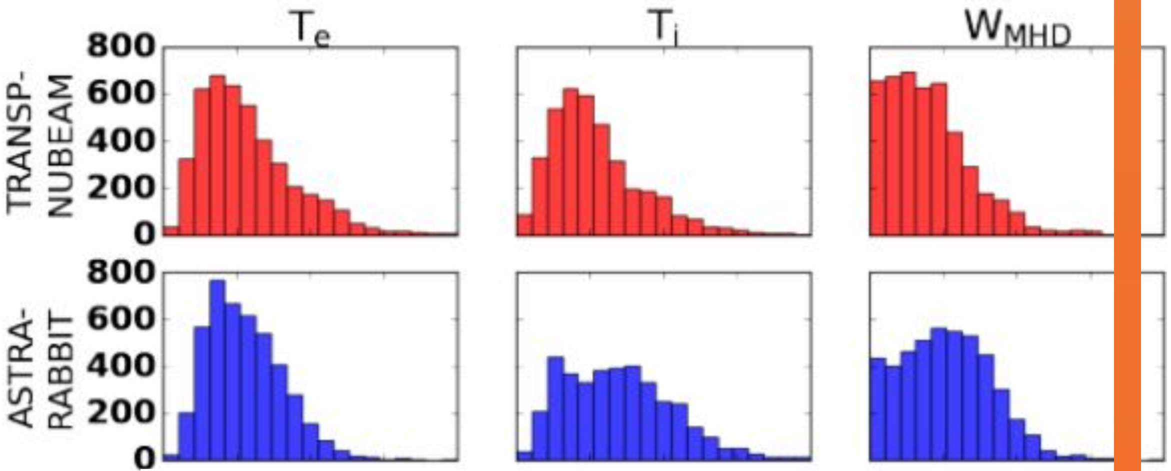
Solvers with predefined diffusivity profiles and source converge to the same solution

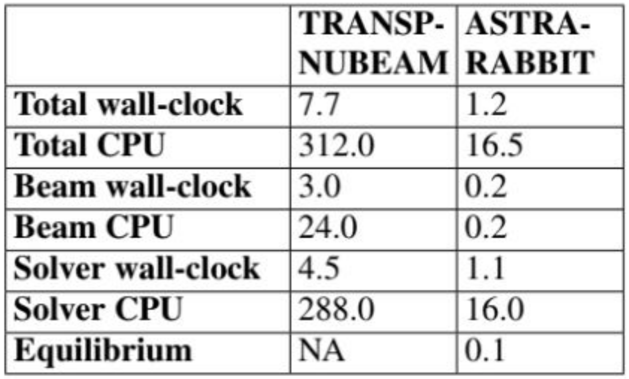
ASTRA performed slightly better for Te

TRANSP performed better for Ti

TRANSP simulations are slower







	TRANSP-NUBEAM	ASTRA-RABBIT
Total wall-clock	7.7	1.2
Total CPU	312.0	16.5
Beam wall-clock	3.0	0.2
Beam CPU	24.0	0.2
Solver wall-clock	4.5	1.1
Solver CPU	288.0	16.0
Equilibrium	NA	0.1

J. Abbate US TTF meeting 2023

User support

User support is provided at

- **TRANSP Q&A session at this meeting:**
  - Governor's Square Rm 17 at Sheraton Denver – Wednesday, November 1 at 12:45pm
- **TRANSP workshops**
  - TRANSP user workshop in January 2022
  - TRANSP planning workshop in September 2023
- **Monthly meeting with representatives from ROs**
  - Every second Wednesday at 10am ET
- **Weekly TRANSP open hour**
  - Each Monday at 12pm ET
- **Users can ask questions and request support at** https://github.com/PrincetonUniversity/TRANSPhub
- **Documentation and tutorials at https://transp.pppl.gov**

Summary

TRANSP remains a community fusion code for interpretive and predictive analysis of tokamak plasmas

Progress in the IMAS-ification of TRANSP

- New tool to save TRANSP output in IMAS format is developed
- TRANSP can start using the IMAS data and save output directly to IMAS
- IMAS interfaces for EC/IC/NBI modules are being developed

Other TRANSP projects

- Modularization of TRANSP components with independent input controls
- CI extension to new cases
- Optimization of NUBEAM using GPU and MPI+OpenMP [M. Gorelenkova et al. NP11.00070]
- Predictive ASTRA and TRANSP runs verified for DIII-D discharges [Joe Abbate]

Future work

Future plans are related to current work

- **Implementation of the IMAS interface in TRANSP creates opportunities for community contributions to TRANSP**
- **We are interested in extending our collaboration with other institutions and making the PPPL external TRANSP modules such as PLASMA_STATE, UFILES, EZCDF, and PT_SOLVER more open for contributions from non-PPPL participants**
  - This modularization effort helps improve the support of individual components with regression (CI) units tests developed for these individual components
- **TRANSP participation in OMFIT was beneficial for TRANSP and we are looking how we can support TRANSP modules in OMFIT**
  - Elements of OMAS project can be beneficial to resolve the IMAS back-compatibility issues [subject to license agreements]