AToM

Advancing Mission-Critical Tokamak Modeling Workflows on HPC Systems

D. L. Green, J. M. Park, S. J. Diem, J. Candy, O. Meneghini, M. Dorf, C. Holland, D. E. Bernholdt, M. Dorr, D. Shissel, S. P. Smith, P. B. Snyder, W. Elwasif, E. D'Azevedo, D. Batchelor, V. Izzo, D. Orlov, M. Umansky

2015 SciDAC-3 Principal Investigator Meeting

Jul 22nd - 24th, Bethesda, MD

ATOM

GENERAL ATOMICS

OAK RIDGE
National Laboratory

UCSD

Lawrence Livermore
National Laboratory

FASTMATH

SUPER

DEPARTMENT OF ENERGY
UNITED STATES OF AMERICA

SciDAC
Scientific Discovery
through
Advanced Computing

Summary

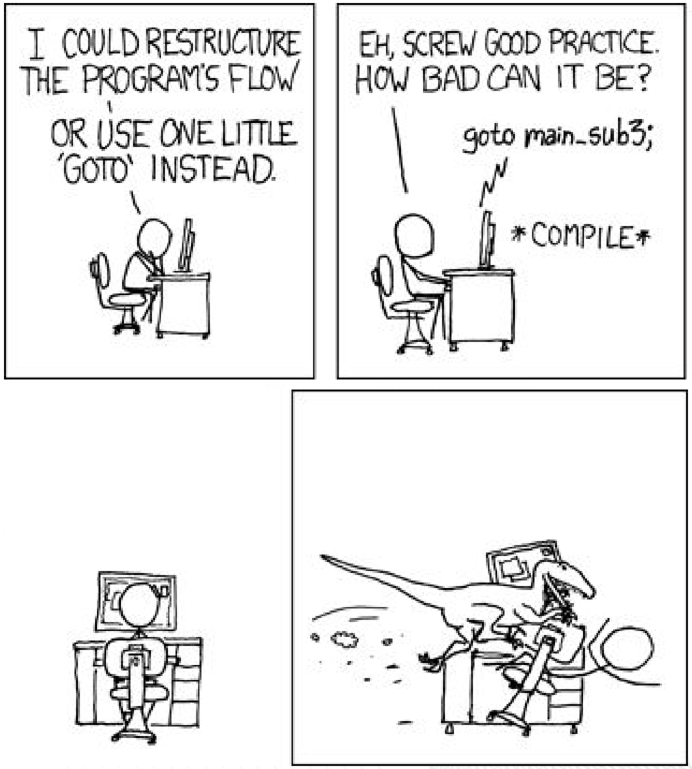
Supporting and enhancing integration use cases

- Physics Motivation : To extend multi-code integrated simulation of the core plasma to include the pedestal and plasma edge regions.
- Approach :
  - Modernize & modularize - update legacy (non-extensible) / mission-critical capability to utilize HPC resources and extend via including HPC capable component codes.
  - Usability - templatizing IPS to enable dynamic construction from a GUI (OMFit) and improve usability and adoption.
- Other example integration workflows :
  - Recalibration of the TGLF turbulent transport model for ITER,
  - Parareal - event driven parallel in time algorithm.

Legacy Integrated Simulation

Risks loss of capability in the longer term

- Much of the existing mission critical use cases of integrated simulation for fusion rely on legacy / non-extensible code bases, e.g.,
  - Device design and
  - Scenario / shot development.
- Such single executable, all-physics-in-one approaches hinder advancing state of the art.
- Limits community contribution, swapping components for different fidelity options, maintainability, etc ...



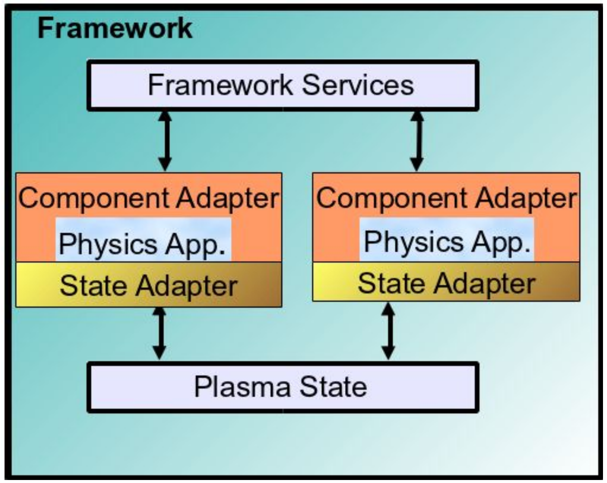
A Modern, Modular Approach

The IPS framework + OMFit interface

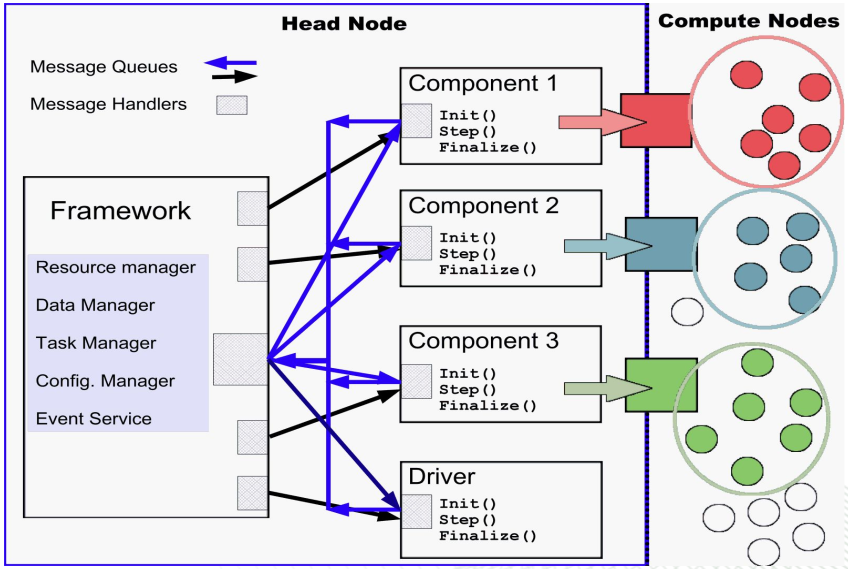
- The basis for **production** scenario development simulation in magnetically confined fusion is an <u>equilibrium solver</u>, plus slow timescale <u>transport solver</u> with various <u>source components</u> …
- A maintainable, extensible, and HPC capable tool that can be contributed to by the community requires that these pieces be modular.
- We provide a framework (**IPS**), communication method (file-based at present), and a top-level transport solver for community components.
- Usability is often overlooked in physics codes, leading to reduced impact of advanced physics models in scenario development simulation. By driving simulations through a standard interface (**OMFit**) we are addressing this.

IPS : Framework Basics

Integrated Plasma Simulator



- Python-based component framework.
- Components are python-wrapped binaries.
- Framework runs in a single batch allocation, manages resources for components.
- Components launch tasks on compute nodes using standard system mechanisms, i.e. mpiexec, aprun …
- “Plasma State” holds primary data for exchange.
  - “Reader-makes-right” model.



IPS : Component Architecture

What the framework provides ...

Framework Services

- Configuration manager.
- Task manager ...
  - Launch underlying applications,
  - Blocking or non-blocking.
- Resource manager ...
  - Nodes allocated to batch job.
- Event service ...
  - Asynchronous pub/sub events.
- Data manager ...
  - File staging (per timestep),
  - Mediate concurrent access to state,
  - Checkpoint/restart (framework level).
- Monitoring via web portal.

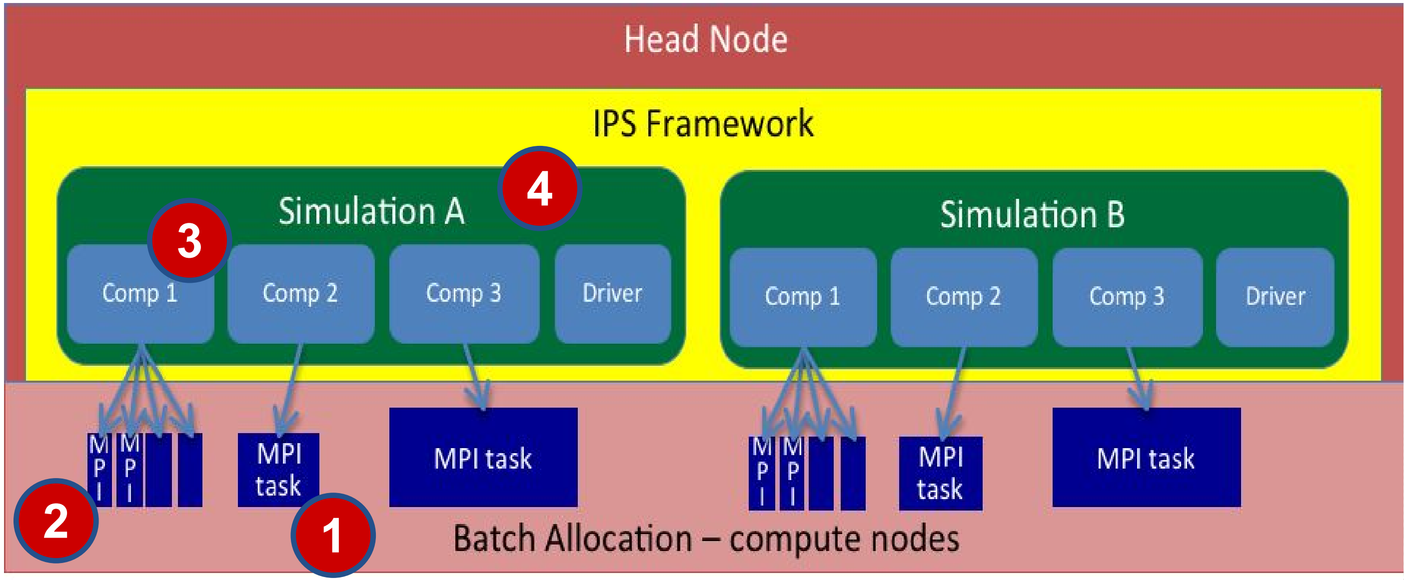
Components

- Components characterized by ports (class) and implementation (instance).
  - All implementations of a port are expected to be fundamentally equivalent in their interactions with other components.
- Primary component interface ...
  - init()
  - step()
  - finalize()
- Distinguished components ...
  - Driver
  - Monitor

IPS : Multi-Level Parallelism

Maximal resource utilization via hierarchical concurrency

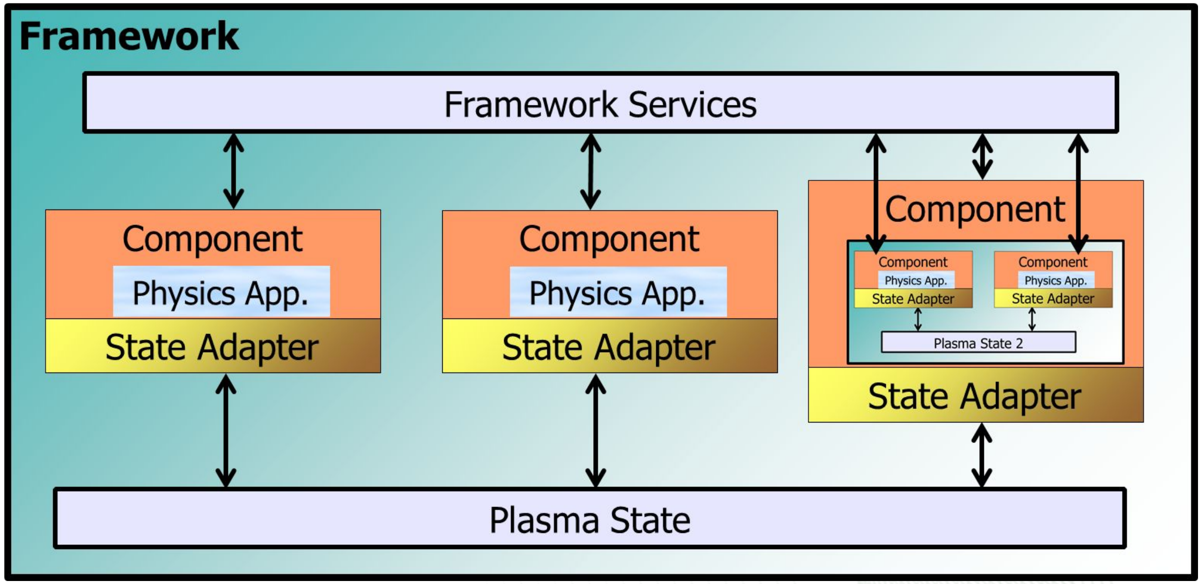
1. Individual "tasks" (physics executables) can be parallel.
2. Components can launch multiple tasks.
3. Multiple components can run concurrently.
4. Multiple independent simulations can run concurrently.



IPS : Nested Workflows

To enable workflow re-use

- Embed (one or more) “sub-workflow” into top-level master workflow.
- Sub-workflows execute in a separate context, bridged to the parent simulation via a bridge component.
- No limit on the number of nesting levels.
- Entire hierarchy executes using a single resource allocation, mediated via the IPS task and resource managers.



Framework

Framework Services

Component
Physics App.
State Adapter

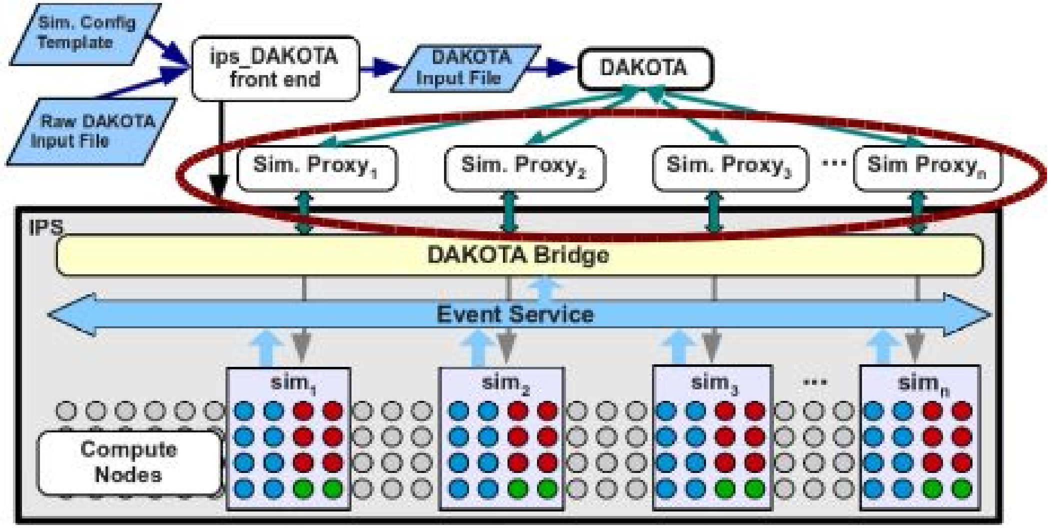
Component
Physics App.
State Adapter

Component
Component
Physics App.
State Adapter
Component
Physics App.
State Adapter
Plasma State 2
State Adapter

Plasma State

IPS : Embedded optimizer

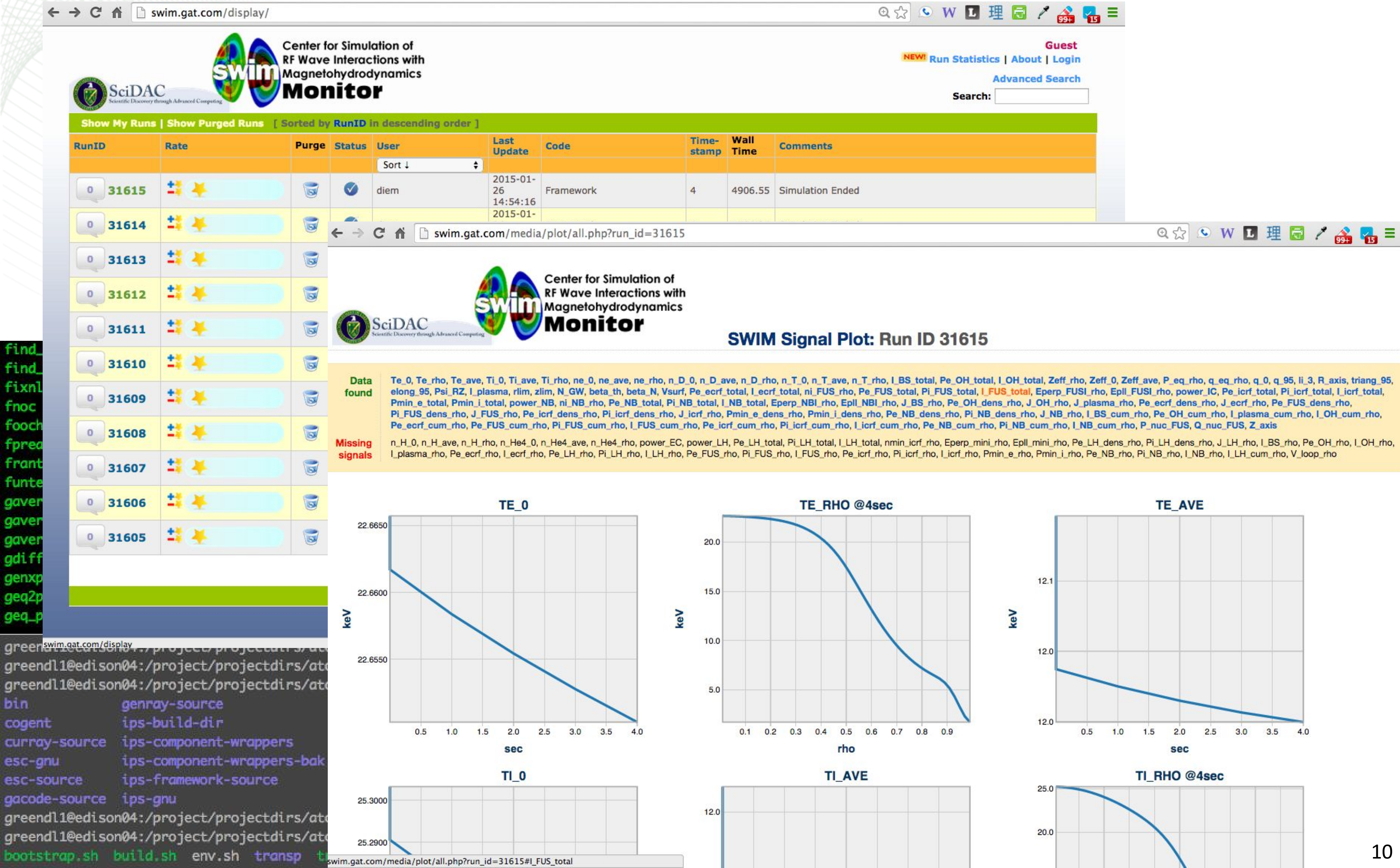
IPS-DAKOTA



- DAKOTA toolkit from SNL
    - Toolkit for design optimization, parameter estimation, UQ, sensitivity analysis, ..
- IPS-DAKOTA integration
    - Single IPS framework instance
    - Manage many, dynamically created DAKOTA (coupled) simulations.
- ATOM use cases, so far these are simple parameter scans ...
    - Core-pedestal coupling (IPS-EPED).
    - TGLF ITER calibration (IPS-GYRO).

IPS : Web Monitor

Track simulation progress online



RunID	Rate	Purge	Status	User	Last Update	Code	Time-stamp	Wall Time	Comments
0 31615			✓	diem	2015-01-26 14:54:16	Framework	4	4906.55	Simulation Ended
0 31614					2015-01-				
0 31613									
0 31612									
0 31611									
0 31610									
0 31609									
0 31608									
0 31607									
0 31606									
0 31605									

SWIM Signal Plot: Run ID 31615

Data found Te_0, Te_rho, Te_ave, Ti_0, Ti_ave, Ti_rho, ne_0, ne_ave, ne_rho, n_D_0, n_D_ave, n_D_rho, n_T_0, n_T_ave, n_T_rho, I_BS_total, Pe_OH_total, I_OH_total, Zeff_rho, Zeff_0, Zeff_ave, P_eq_rho, q_eq_rho, q_0, q_95, li_3, R_axis, triang_95, elong_95, Psi_RZ, I_plasma, rlim, zlim, N_GW, beta_th, beta_N, Vsurf, Pe_ecrf_total, I_ecrf_total, ni_FUS_rho, Pe_FUS_total, Pi_FUS_total, I_FUS_total, Eperp_FUSI_rho, Epll_FUSI_rho, power_IC, Pe_icrf_total, Pi_icrf_total, I_icrf_total, Pmin_e_total, Pmin_i_total, power_NB, ni_NB_rho, Pe_NB_total, Pi_NB_total, I_NB_total, Eperp_NBI_rho, Epll_NBI_rho, J_BS_rho, Pe_OH_dens_rho, J_OH_rho, J_plasma_rho, Pe_ecrf_dens_rho, J_ecrf_rho, Pe_FUS_dens_rho, Pi_FUS_dens_rho, J_FUS_rho, Pe_icrf_dens_rho, Pi_icrf_dens_rho, J_icrf_rho, Pmin_e_dens_rho, Pmin_i_dens_rho, Pe_NB_dens_rho, Pi_NB_dens_rho, J_NB_rho, I_BS_cum_rho, Pe_OH_cum_rho, I_plasma_cum_rho, I_OH_cum_rho, Pe_ecrf_cum_rho, Pe_FUS_cum_rho, Pi_FUS_cum_rho, I_FUS_cum_rho, Pe_icrf_cum_rho, Pi_icrf_cum_rho, I_icrf_cum_rho, Pe_NB_cum_rho, Pi_NB_cum_rho, I_NB_cum_rho, P_nuc_FUS, Q_nuc_FUS, Z_axis

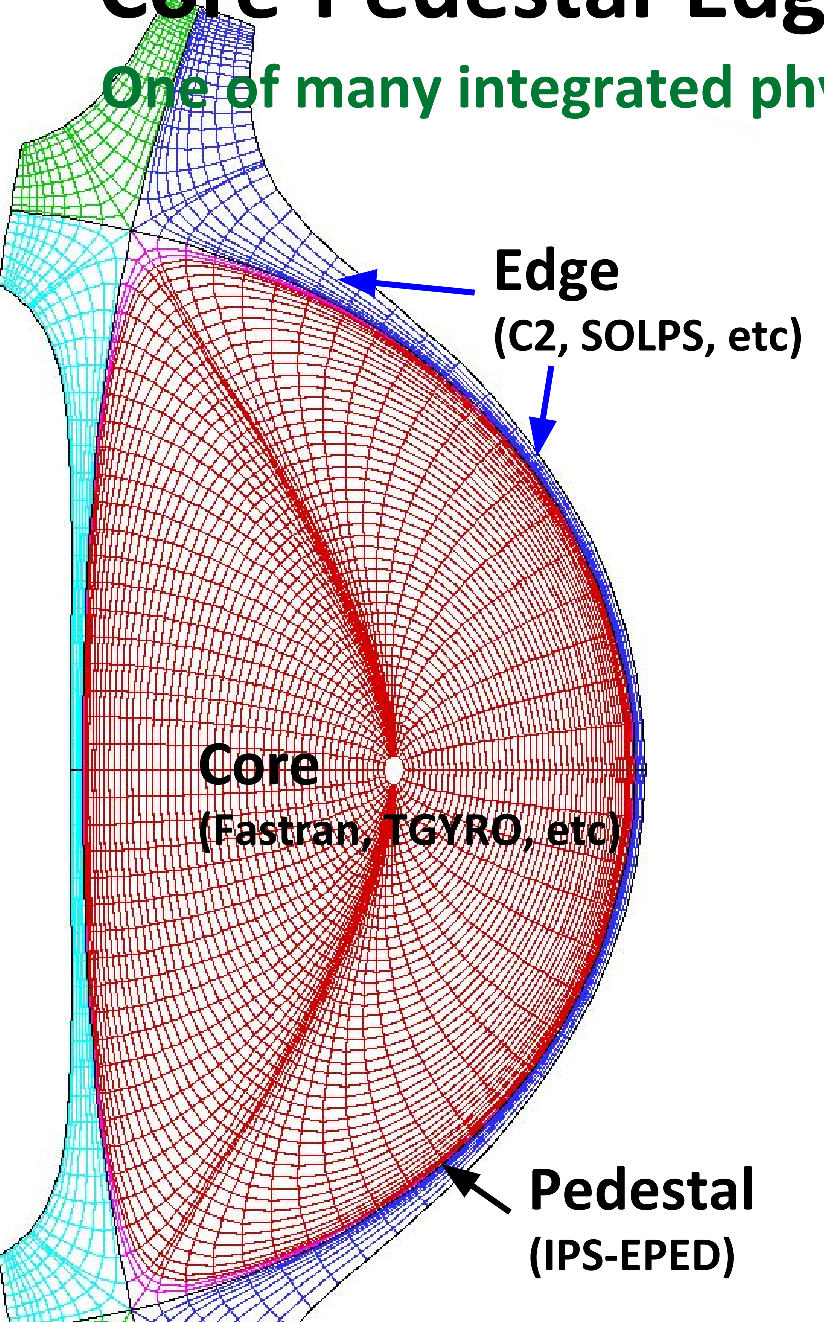
Missing signals n_H_0, n_H_ave, n_H_rho, n_He4_0, n_He4_ave, n_He4_rho, power_EC, power_LH, Pe_LH_total, Pi_LH_total, I_LH_total, nmin_icrf_rho, Eperp_mini_rho, Epll_mini_rho, Pe_LH_dens_rho, Pi_LH_dens_rho, J_LH_rho, I_BS_rho, Pe_OH_rho, I_OH_rho, I_plasma_rho, Pe_ecrf_rho, I_ecrf_rho, Pe_LH_rho, Pi_LH_rho, I_LH_rho, Pe_FUS_rho, Pi_FUS_rho, I_FUS_rho, Pe_icrf_rho, Pi_icrf_rho, I_icrf_rho, Pmin_e_rho, Pmin_i_rho, Pe_NB_rho, Pi_NB_rho, I_NB_rho, I_LH_cum_rho, V_loop_rho

```
find_
find_
fixnl
fnoc
fooch
fprea
frant
funte
gaver
gaver
gaver
gdiff
genxp
geq2p
geq_p
```

```
gree
greendl1@edison04:/project/projectdirs/atc
greendl1@edison04:/project/projectdirs/atc
bin             genray-source
cogent          ips-build-dir
curray-source   ips-component-wrappers
esc-gnu         ips-component-wrappers-bak
esc-source      ips-framework-source
gacode-source   ips-gnu
greendl1@edison04:/project/projectdirs/atc
greendl1@edison04:/project/projectdirs/atc
bootstrap.sh  build.sh  env.sh  transp  t
```

Core-Pedestal-Edge Coupling

One of many integrated physics use cases

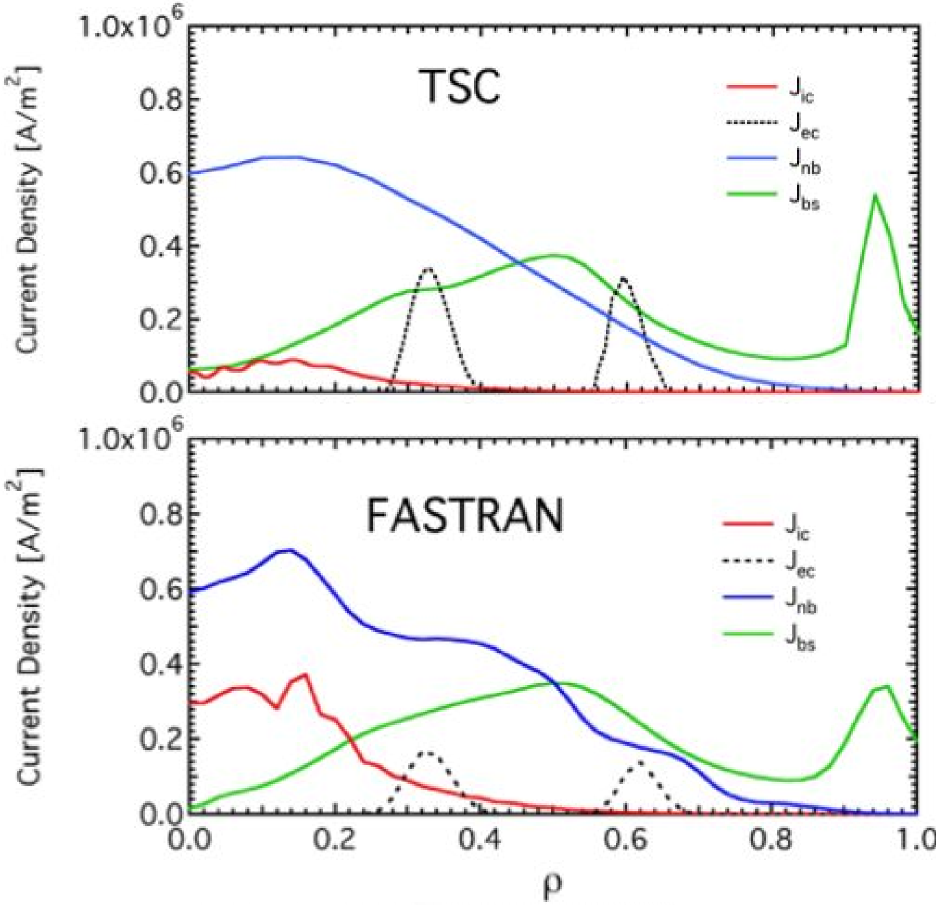


1. Validate (with sources) modular core transport solver **[FASTRAN vs TSC]**.
2. Couple core region to pedestal **[TGYRO + EPED, FASTRAN + EPED]**.
3. Validate (with fluid neutrals to start) edge transport solver **[C2 vs SOLPS-ITER]**.
4. Couple edge solver to pedestal **[C2 + EPED]**.
5. Couple edge + pedestal + core **[FASTRAN + EPED + C2]**.

Core Plasma

FASTRAN vs TSC Benchmarking.

3000s simulation = 135 k CPU hours



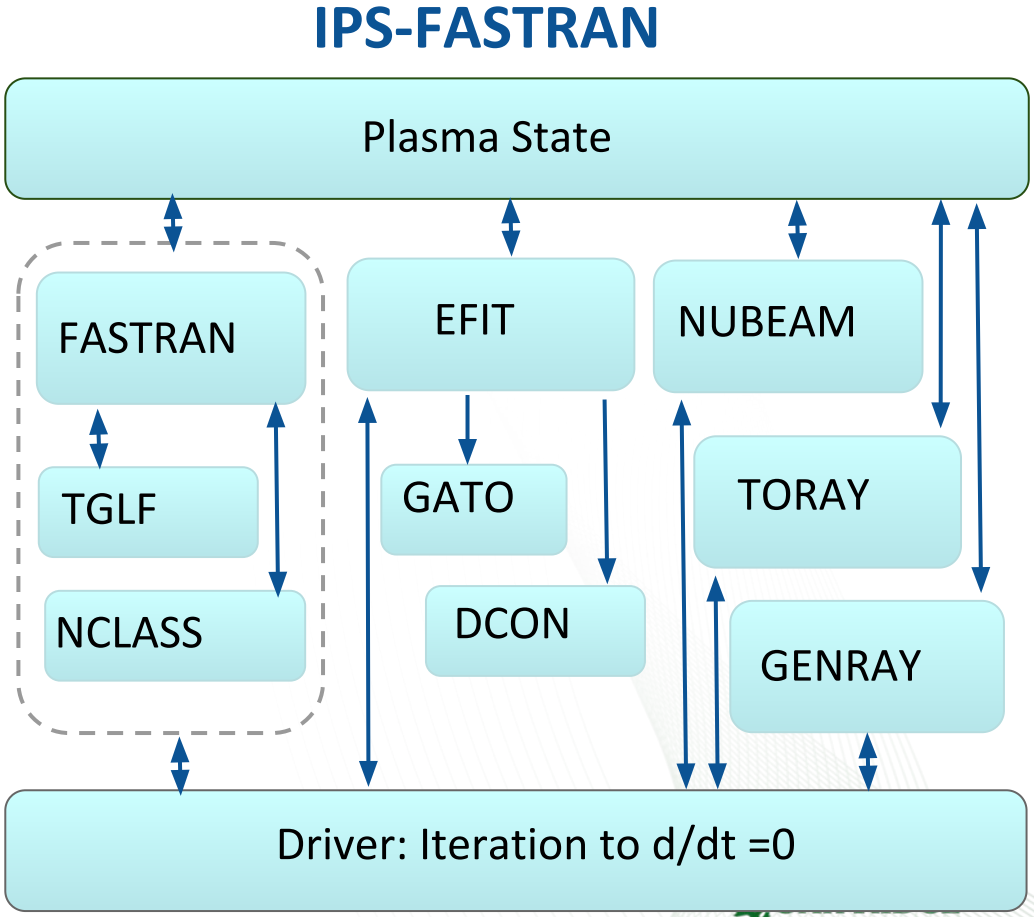
- **Benchmarking monolithic and multi-code component simulations is non-trivial.**
- **Initial benchmarking of TSC with FASTRAN was plagued with difficulties** in matching input settings, input profiles, binary differences, etc.
- An IPS-TSC workflow has been constructed to use exactly the same source components (binaries, inputs, profiles, etc).
- The benchmarking **should be as simple as** running TSC to steady-state, switching a couple of lines in the IPS config file, then letting FASTRAN take over - **work in progress**.

IPS-FASTRAN

An IPS based modular core transport solver

All component codes are IPS components that communicate via the plasma-state file.

- **GENRAY** : EC heating & current drive
- **TORIC** : IC heating & current drive
- **NUBEAM** : NBI heating
- TSC models entire plasma startup & evolution.
- FASTRAN iterates on last TSC solution.

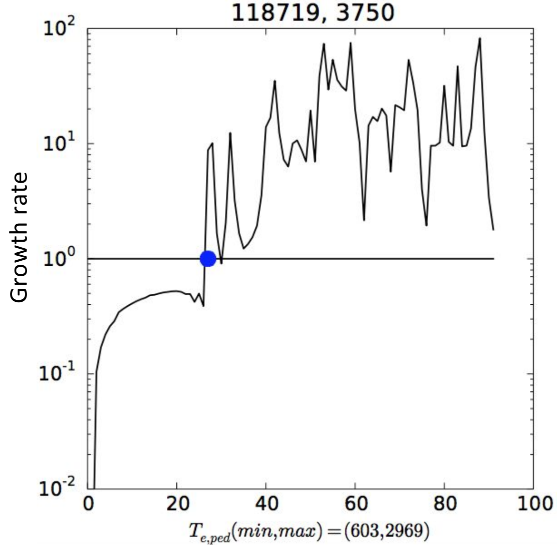


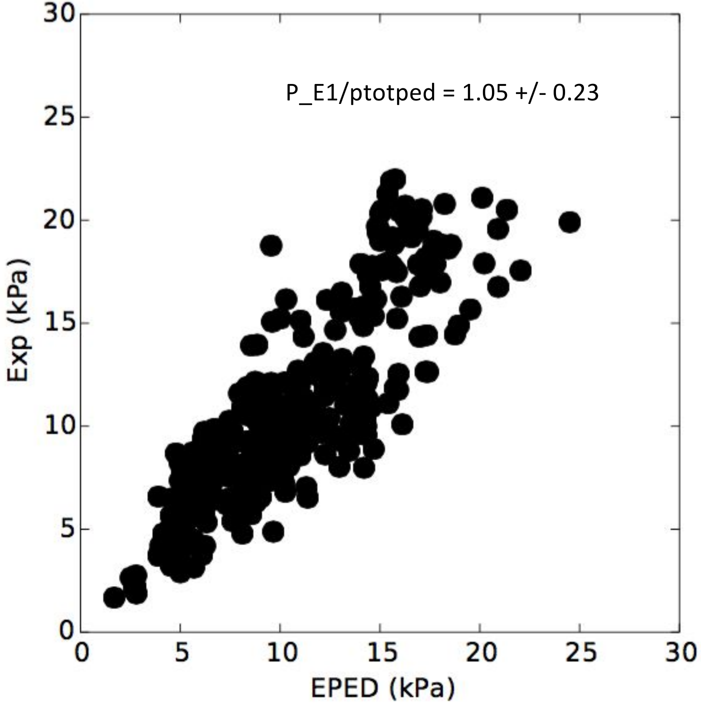
Pedestal

IPS-EPED

- EPED [Snyder et al.*] is a composite component that has been upgraded to a modular IPS-EPED workflow
  - ○ Model equilibrium component: TOQ
  - ○ Kinetic Ballooning component: BALOO
  - ○ Peeling Ballooning component: ELITE
- IPS-EPED runs in parallel ...

  480 TOQ + 400 BALOO + 5*80 ELITE runs
  - ○ **Now fast enough for time dependent simulation.**
  - ○ IPS-EPED runs in 2 mins using 700 CPUs.
- **Verified successfully against original idl-EPED.**





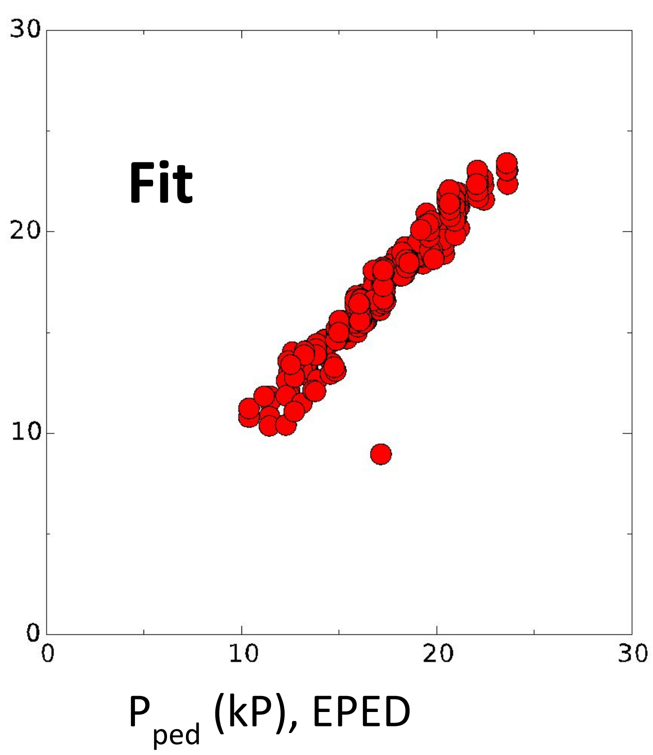
30 concurrent runs on 120 cores each for 1.5 wall clock hours.

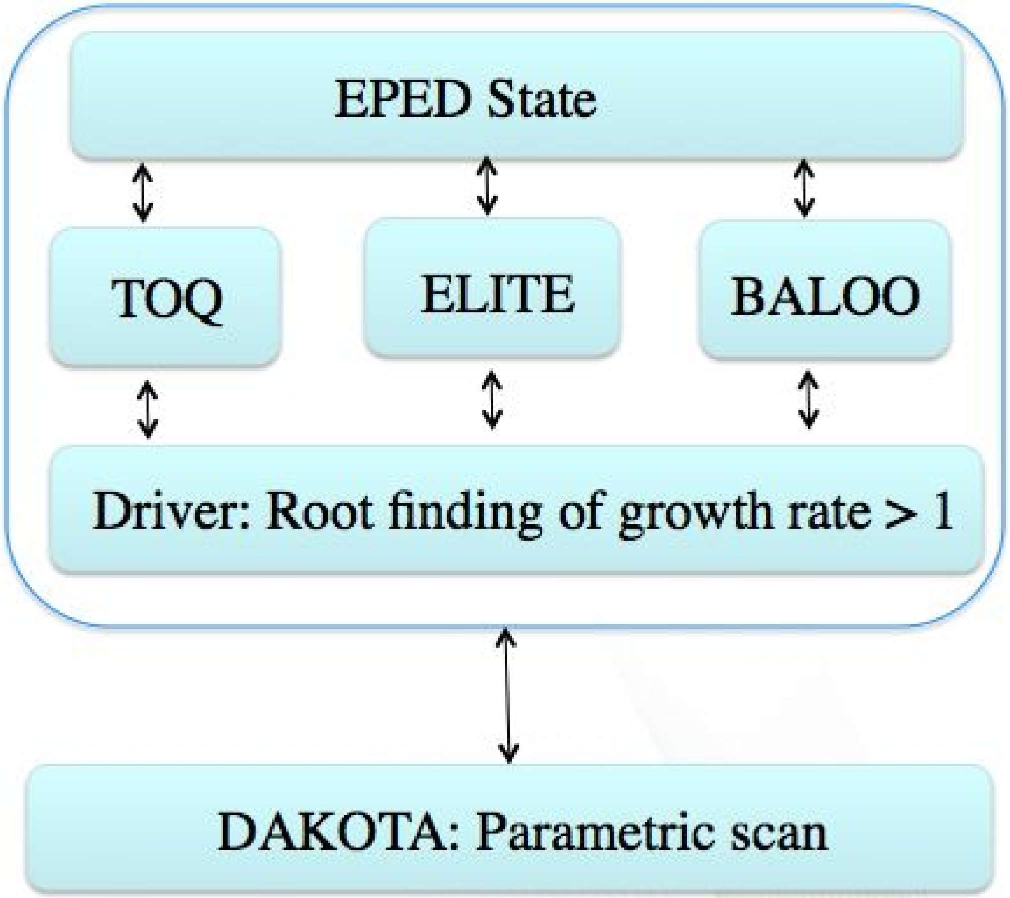
---

[*] P.B. Snyder *et al* 2011 *Nucl. Fusion* **51** 103016 doi:10.1088/0029-5515/51/10/103016

IPS-EPED

A nested IPS workflow





EPED State

TOQ

ELITE

BALOO

Driver: Root finding of growth rate > 1

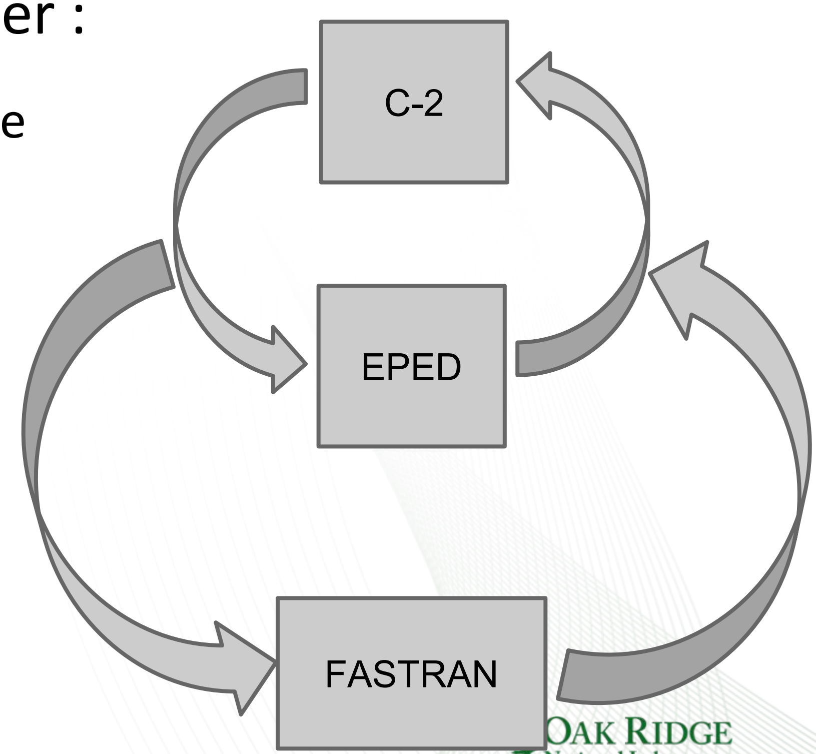
DAKOTA: Parametric scan

- Parametric fit to 500 EPED results - **required 1000 CPUs for 24 wallclock hours**.
- Ultimately a many parameter scan / fit will be performed to create a reduced model for non-HPC application.

Core Plasma + Pedestal + Edge

Future plans

- Presently benchmarking C-2 with SOLPS-ITER (for fluid neutrals only).
- Ultimately iterate edge and core fluid transport solvers to convergence with EPED. **Iteration scheme unclear as yet.**
- C-2 used as 2-D SOL transport solver :
  - A 2-D multi-fluid model extending the previous formulations of 1-D core and 2-D edge / SOL transports: valid not only in the collisional edge / SOL regions but also in the high temperature core region.
  - Solve plasma continuity, parallel momentum balance, electron / ion energy and add current continuity.

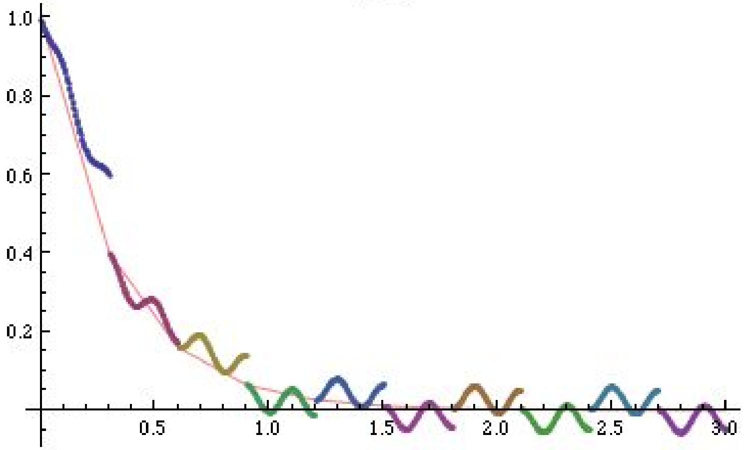


Parallelization over time

IPS-PARAREAL

- Requires ...
  - Accurate (fine) solver F(x, t)
  - Fast (coarse) solver G(x, t)
  - Convergence criterion for solution x
- Break time domain into a large number of chunks, NΔT.
- Run fine solver in parallel using coarse solution as starting point for each time chunk.
- Parareal algorithm - iteration scheme connecting coarse and fine solutions, guaranteed convergence in N iterations, but maybe much faster.

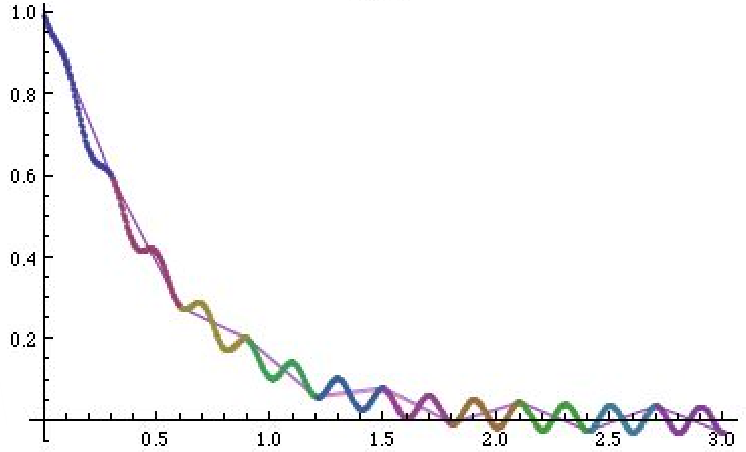
k = 1



k = 1

$$\frac{du}{dt} - \lambda u = \sin(5\pi t) \equiv F_{n,\Delta T} \qquad \frac{du}{dt} - \lambda u = 0 \equiv G_{n,\Delta T}$$

k = 3

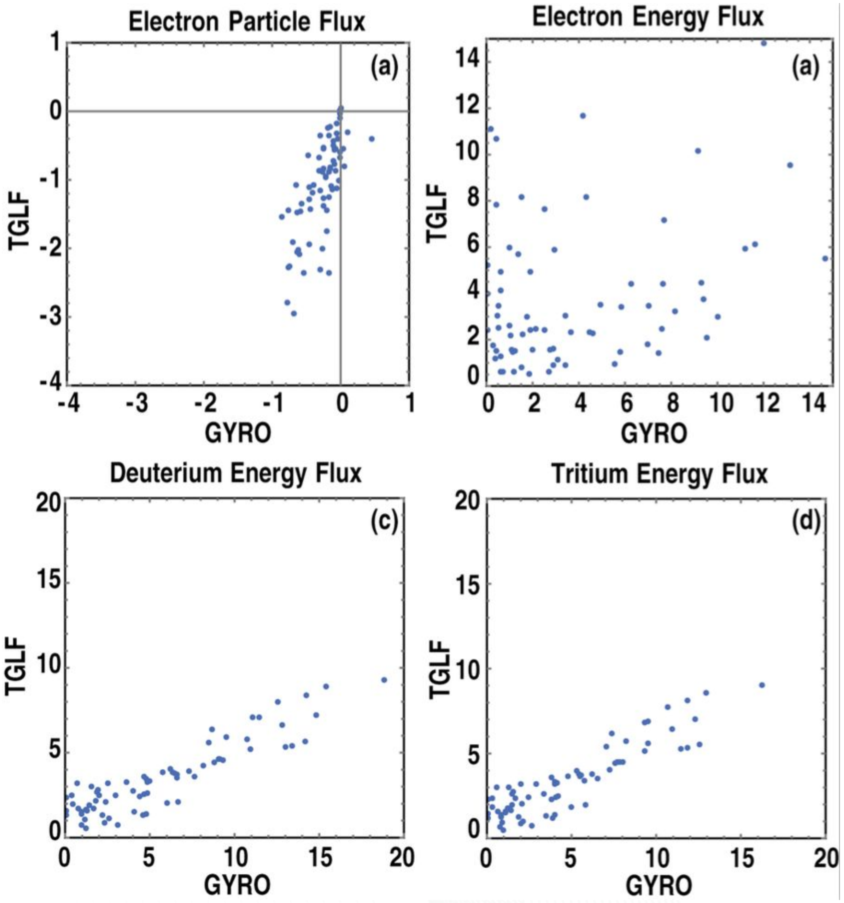


k = 3

Recalibration of TGLF for ITER

IPS-GYRO & TGYRO

- TGLF is a reduced model of turbulent transport used within transport solvers like FASTRAN, TGYRO, etc. It is based on many HPC runs of the GYRO code (a gyro-kinetic continuum code).
- For use in ITER scenario development, a new set of GYRO runs was completed to expand TGLF to the ITER regime.
- Here we show 72 GYRO runs launched via IPS-DAKOTA **requiring 185 k CPU hours**.

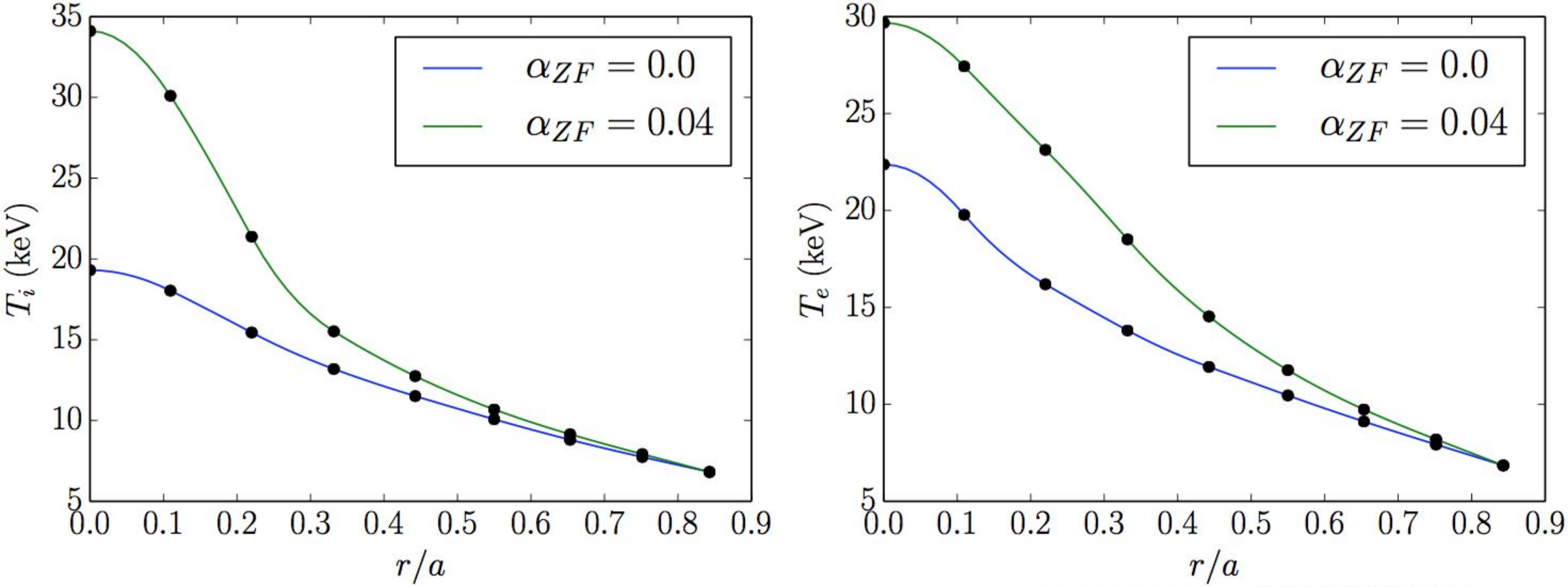


Scatter plot of 72 IPS-GYRO runs comparing TGLF and GYRO fluxes. The spread in electron energy flux was addressed by adding a small amount of collisions to subsequent cases.

Recalibration of TGLF for ITER

IPS-GYRO & TGYRO

Preliminary result : The inclusion of zonal-flow physics required in the ITER regime to TGLF may significantly affect the fusion power.

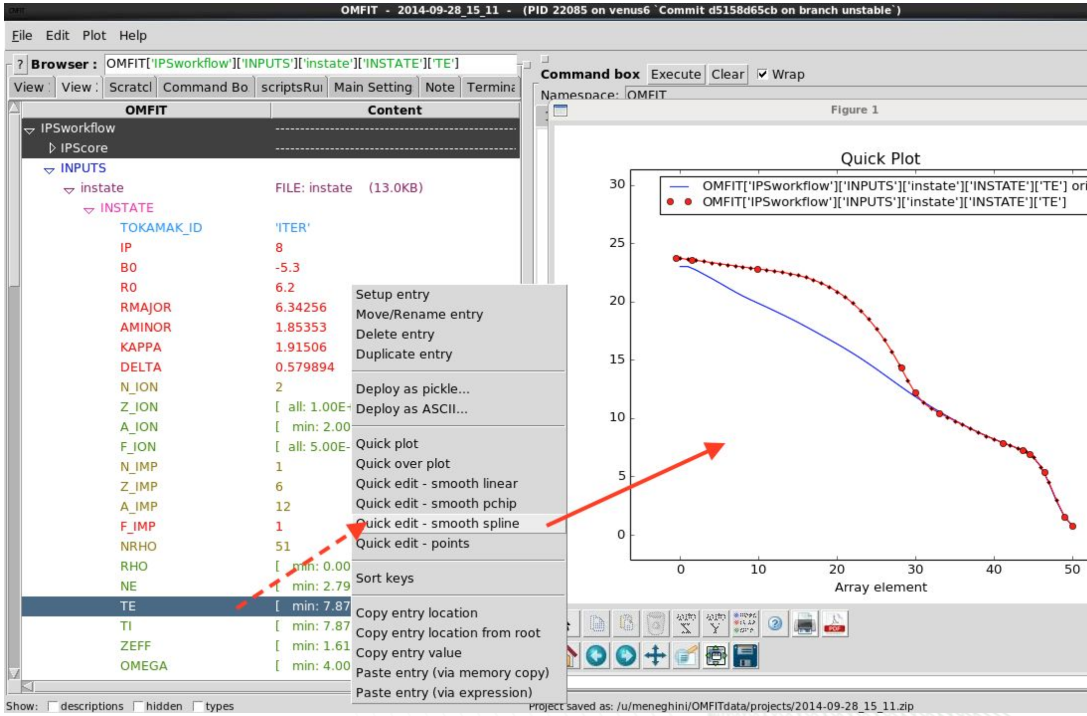


Ultimately the TGLF calibration runs of GYRO are likely to become an OMFit workflow to further automate this process.

Usability is Paramount

OMFit to drive the IPS

- Setting up an integrated simulation for fusion scenario development is at present a non-trivial task prone to human error in the myriad of parameters, files, inputs, outputs, etc.
- Work towards templatization of IPS simulation configurations is continuing to ultimately enable HPC based workflows to be constructed and run from the OMFit GUI.



OMFIT - 2014-09-28_15_11 - (PID 22085 on venus6 `Commit d5158d65cb on branch unstable`)

Browser : OMFIT['IPSworkflow']['INPUTS']['instate']['INSTATE']['TE']

Namespace: OMFIT

OMFIT	Content
IPSworkflow	--
IPScore	--
INPUTS	
instate	FILE: instate (13.0KB)
INSTATE	
TOKAMAK_ID	'ITER'
IP	8
B0	-5.3
R0	6.2
RMAJOR	6.34256
AMINOR	1.85353
KAPPA	1.91506
DELTA	0.579894
N_ION	2
Z_ION	[all: 1.00E-
A_ION	[min: 2.00
F_ION	[all: 5.00E-
N_IMP	1
Z_IMP	6
A_IMP	12
F_IMP	1
NRHO	51
RHO	[min: 0.00
NE	[min: 2.79
TE	[min: 7.87
TI	[min: 7.87
ZEFF	[min: 1.61
OMEGA	[min: 4.00

Show: ☐ descriptions ☐ hidden ☐ types

Project saved as: /u/meneghini/OMFITdata/projects/2014-09-28_15_11.zip