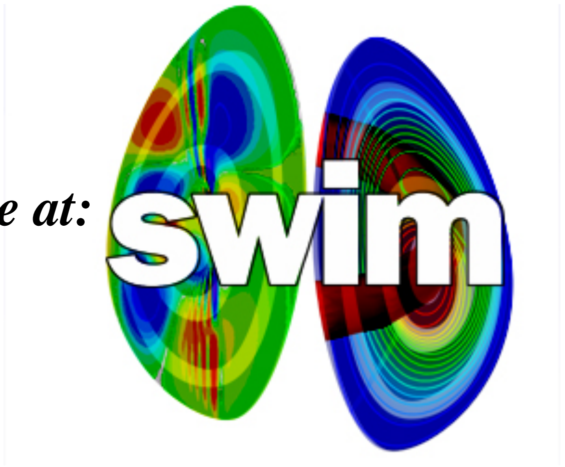
The Integrated Plasma Simulator: A flexible framework for coupled fusion simulations

November 2013
JET, EFDA ITM Code Camp

D. B. Batchelor, L. A. Berry, E. F. Jaeger, D. A. Spong – *ORNL Fusion Energy*
W. Elwasif, D. E. Bernholdt, E. D' Azevedo, S. Foley (NCCS) – *ORNL Computer Science*
S. C. Jardin, E. Feibush, D. McCune, J. Chen, L. P Ku, M. Chance, J. Breslau, F. Poli – *PPPL*
G. Abla, M. Choi, D. P. Schissel – *General Atomics ,* **R. W. Harvey** – *CompX*
R. Bramley– *Indiana University,* **D. Keyes** – *Columbia University,* **D. Schnack** – *U. Wisconsin*
P. T. Bonoli, J. Ramos, J.Wright – *MIT,* **S. Kruger, T. Jenkins** – *TechX*, **G. Bateman** – *Lehigh University*

Unfunded participants:
D. Samaddar – *U. Alaska, ITER IO, JET*
L. Sugiyama – *MIT,* **J. D. Callen, C. C. Hegna, C. Sovinec** – *University of Wisconsin,* **E. H. St. John** – *General Atomics,* **A. Kritz** – *Lehigh Univ.*

See our fun website at:
www.cswim.org



Project – Simulation of Wave Interactions with MHD (SWIM) Goals:

- **Provide a base of experience with framework/component architecture applied to integrated fusion simulation that can be factored into the design of a larger-scale Fusion Simulation Project.**
- **Develop a computational environment that is useful for a broad range of plasma simulation applications (⇒ *Is the tool of choice for those performing tokamak simulations*)**

And adddress physics questions such as:

- **How does RF control sawtooth instability behavior? → Can the ITER ICRF system influence sawteeth?**
- **How does electron cyclotron current drive control Neoclassical Tearing Modes? → How much power will it take on ITER?**

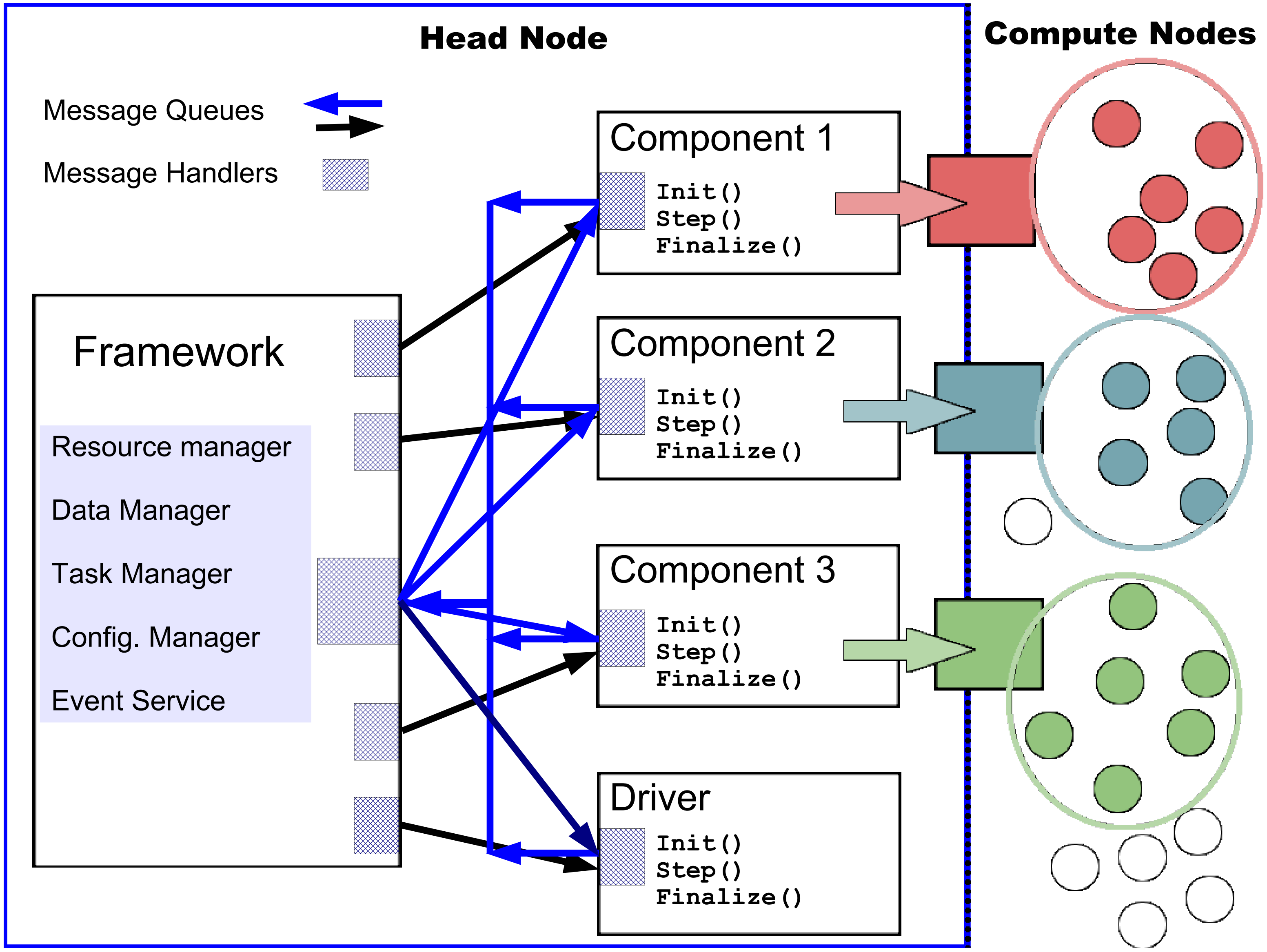
Software infrastructure: Integrated Plasma Simulator (IPS)
A flexible, extensible computational framework capable of coupling state-of-the-art models for energy and particle sources, transport, and stability for tokamak core plasma

IPS Design Approach – borrows from common component architecture (CCA)

Objective – permit massively parallel physics modules to interoperate flexibly and efficiently

- **Framework/component architecture – *written in Python***
  - **Flexibility – multiple codes can implement components interchangeably**
  - **Extensibility – easy to add components to framework**
  - **Components can be tested stand-alone**
- **Components implemented using existing whole codes *(usually in Fortran)* wrapped in standard component interface *(written in Python)***
  - **Rapid deployment – minimize changes to physics codes to adapt**
  - **Avoid bifurcation of physics modules – no SWIM/stand-alone versions**
- **File-based communication**
  - **Zero change to physics code – use existing I/O file structures**
  - **Avoid name-space/compiler/library incompatibilities between components**
- **Plasma State: official transfer mechanism for time-evolving data that must be transferred between components *(optional)***

IPS Architecture

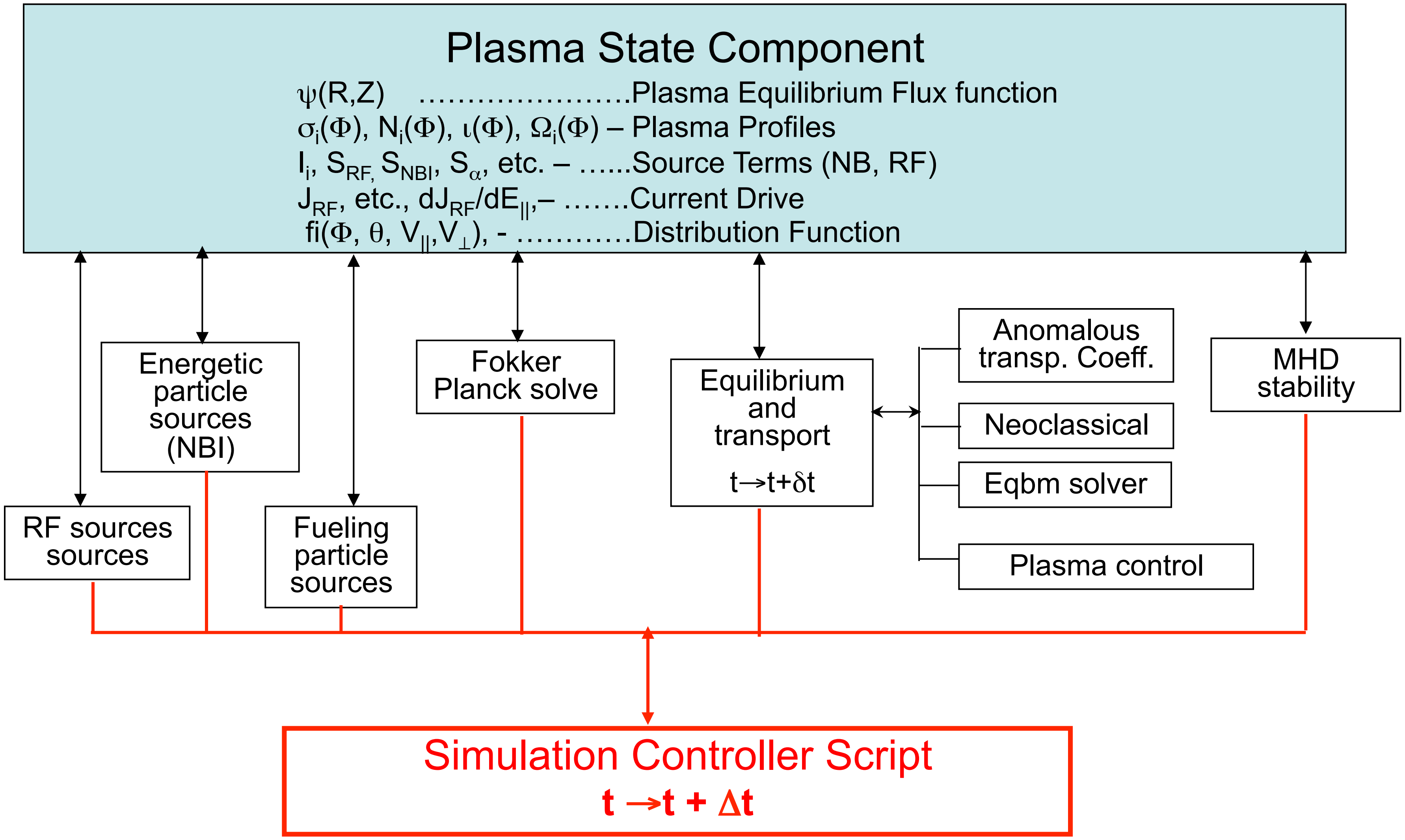


Typically simulation data exchanged between components through Plasma State module

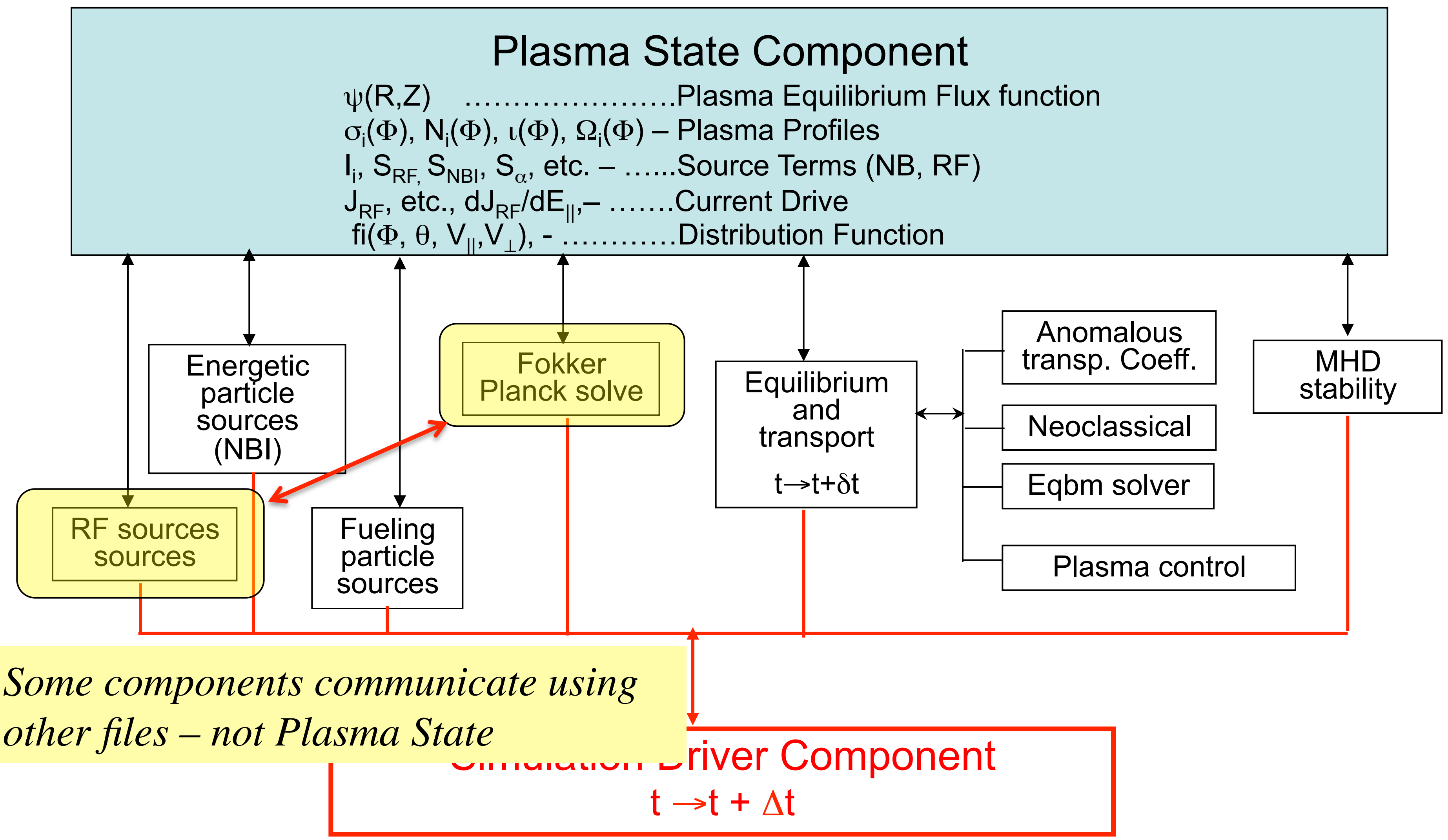
- Fortran 90 Module – supports in-memory or file-based data exchange (netCDF)
- Very simple user interface → functions: get, store, commit, merge partial
- Other powerful functions available, but not required → e.g. grid interpolation
- Supports multiple state instances, partial states
- Code is automatically generated from state specification text file → ease and accuracy of update
- Being shared with other projects
  - Component-to-component data exchange in TRANSP and PTRANSP
  - Coupling of neutral beam and fusion product sources to FACETS C/C++ transport driver

More generally "*plasma state*" consists of a set of files that are managed and transported as group by the framework – eg eqdsk files, distribution functions

Typical Integrated Plasma Simulation Workflow

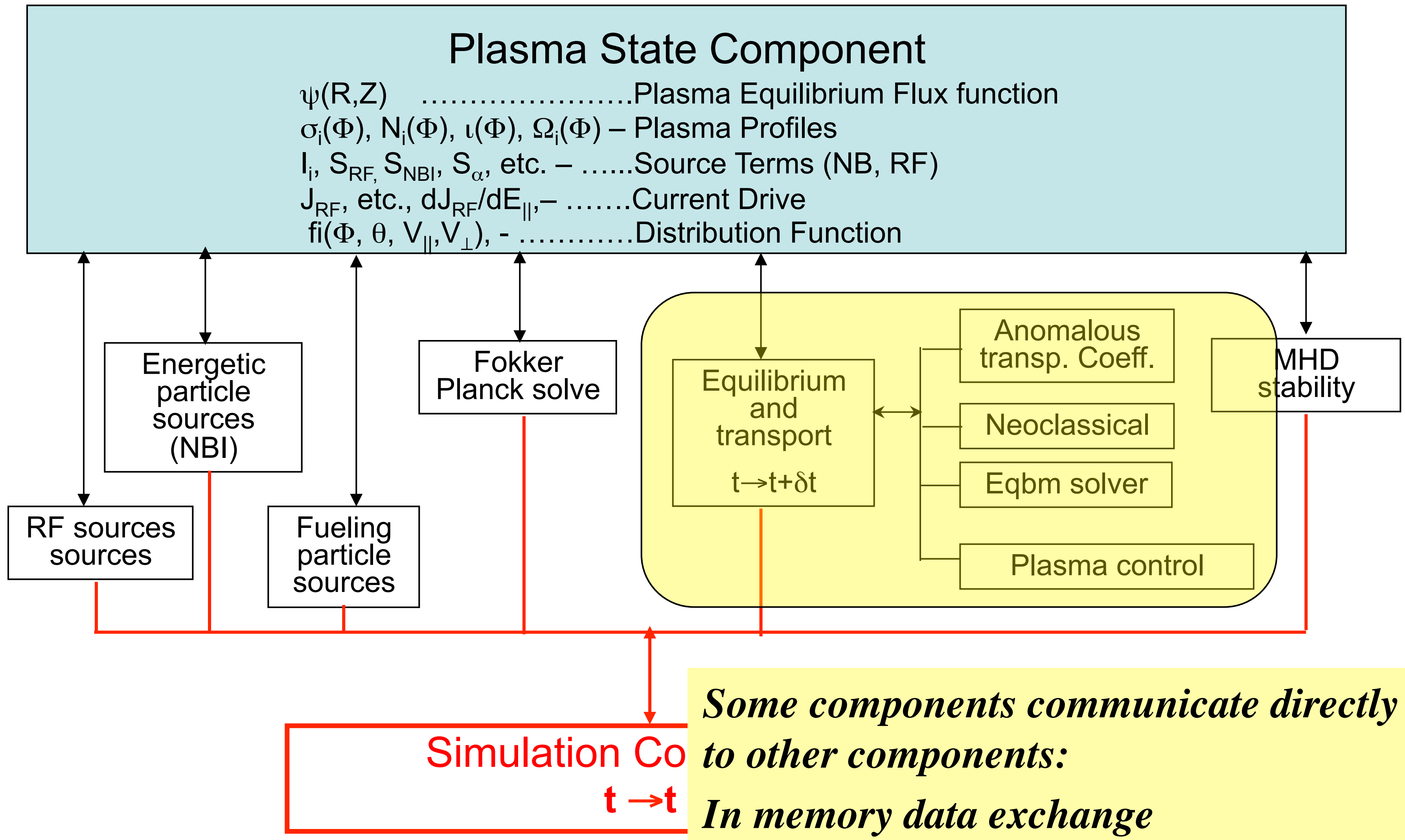

Plasma State Component
$\psi(R,Z)$ ....Plasma Equilibrium Flux function
$\sigma_i(\Phi)$, $N_i(\Phi)$, $\iota(\Phi)$, $\Omega_i(\Phi)$ – Plasma Profiles
I_i, $S_{RF,}$ S_{NBI}, S_α, etc. – ......Source Terms (NB, RF)
J_{RF}, etc., $dJ_{RF}/dE_{||}$,– ....Current Drive
$fi(\Phi, \theta, V_{||}, V_\perp)$, - ....Distribution Function
Energetic particle sources (NBI)
Fokker Planck solve
Equilibrium and transport
$t \rightarrow t+\delta t$
Anomalous transp. Coeff.
Neoclassical
Eqbm solver
Plasma control
MHD stability
RF sources sources
Fueling particle sources
Simulation Controller Script
$t \rightarrow t + \Delta t$

Typical Integrated Plasma Simulation Workflow


Plasma State Component
$\psi(R,Z)$ ....Plasma Equilibrium Flux function
$\sigma_i(\Phi)$, $N_i(\Phi)$, $\iota(\Phi)$, $\Omega_i(\Phi)$ – Plasma Profiles
I_i, $S_{RF,}$ S_{NBI}, S_α, etc. – ......Source Terms (NB, RF)
J_{RF}, etc., $dJ_{RF}/dE_{||}$,– ....Current Drive
$fi(\Phi, \theta, V_{||}, V_\perp)$, - ....Distribution Function
Energetic particle sources (NBI)
Fokker Planck solve
RF sources sources
Fueling particle sources
Equilibrium and transport $t \rightarrow t+\delta t$
Anomalous transp. Coeff.
Neoclassical
Eqbm solver
Plasma control
MHD stability
Simulation Driver Component
$t \rightarrow t + \Delta t$

Some components communicate using other files – not Plasma State

Typical Integrated Plasma Simulation Workflow


Plasma State Component
ψ(R,Z) ……………………Plasma Equilibrium Flux function
$\sigma_i(\Phi)$, $N_i(\Phi)$, $\iota(\Phi)$, $\Omega_i(\Phi)$ – Plasma Profiles
I_i, $S_{RF,}$ S_{NBI}, S_α, etc. – ……Source Terms (NB, RF)
J_{RF}, etc., $dJ_{RF}/dE_{||}$,– …….Current Drive
fi(Φ, θ, $V_{||}$,$V_\perp$), - …………Distribution Function
Energetic particle sources (NBI)
Fokker Planck solve
RF sources sources
Fueling particle sources
Equilibrium and transport
$t \rightarrow t+\delta t$
Anomalous transp. Coeff.
Neoclassical
Eqbm solver
Plasma control
MHD stability
Simulation Co
$t \rightarrow t$

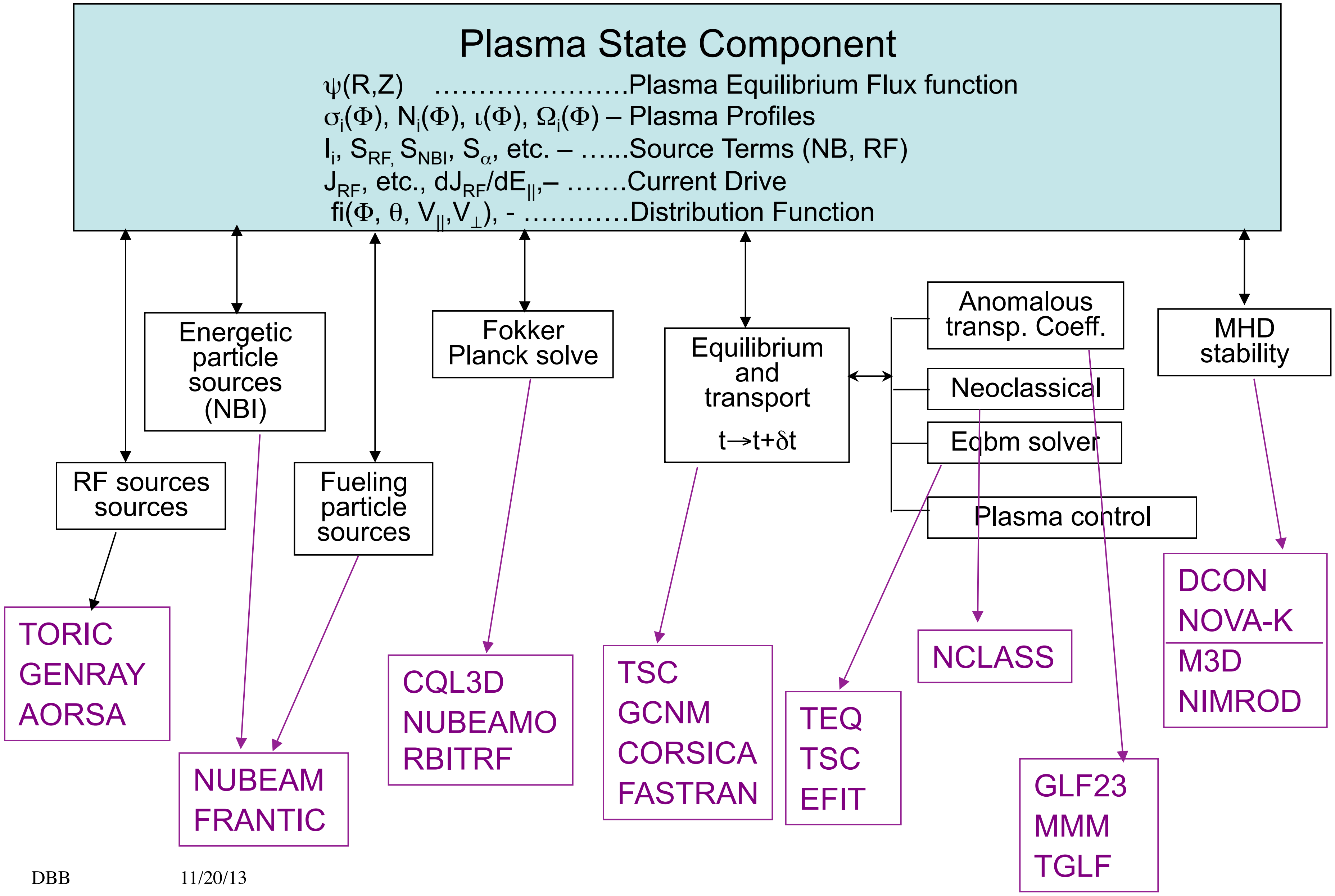
Some components communicate directly to other components:

In memory data exchange

Data not through Plasma State

Control independent of controller script

Integrated Plasma Simulator – Components are implemented by mature, well-validated codes

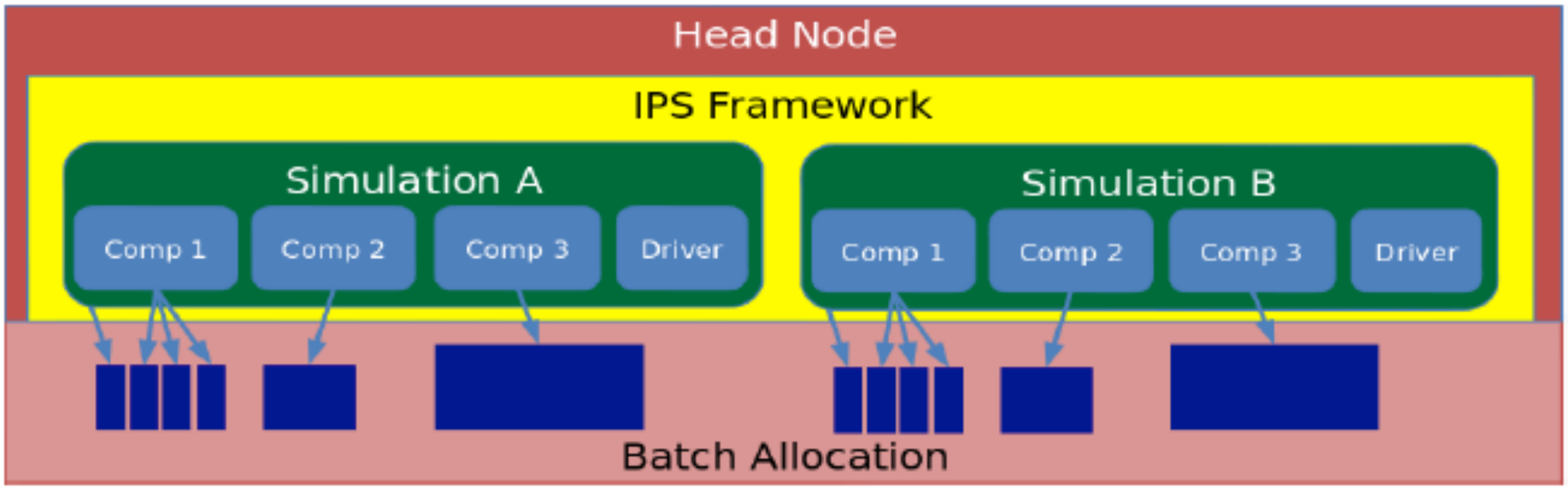

Plasma State Component
$\psi(R,Z)$ ....Plasma Equilibrium Flux function
$\sigma_i(\Phi)$, $N_i(\Phi)$, $\iota(\Phi)$, $\Omega_i(\Phi)$ – Plasma Profiles
I_i, $S_{RF,}$ S_{NBI}, S_α, etc. – ......Source Terms (NB, RF)
J_{RF}, etc., $dJ_{RF}/dE_{||}$,– ....Current Drive
fi(Φ, θ, $V_{||}$,$V_\perp$), - ....Distribution Function
RF sources sources
Energetic particle sources (NBI)
Fueling particle sources
Fokker Planck solve
Equilibrium and transport
$t \rightarrow t+\delta t$
Anomalous transp. Coeff.
Neoclassical
Eqbm solver
Plasma control
MHD stability
TORIC
GENRAY
AORSA
NUBEAM
FRANTIC
CQL3D
NUBEAMO
RBITRF
TSC
GCNM
CORSICA
FASTRAN
TEQ
TSC
EFIT
NCLASS
GLF23
MMM
TGLF
DCON
NOVA-K
M3D
NIMROD

Capabilities of the IPS: Why you should love and use the IPS

- **Driver written in programming language (Python) – allows arbitrarily complicated component invocation**
- **Framework does the dirty work for you (provides services)**
  - Configuration management – assembles and connects needed components (config file)
  - Task management – manages execution of underlying applications
  - Data management – move/archive component input/output and plasma state files
  - Resource management – efficiently manages access to computing resources for concurrently running processes
  - Event management – provides asynchronous publish/subscribe model of data exchange in running simulation
  - Supports component Checkpoint/Restart
  - Task relaunch (prototype) – allows tasks to be relaunched if they have failed with an error code
  - Simulation monitoring – publishes simulation meta-data and events to web-based SWIM portal
- **Monitor component aggregates time-slice data from separate physics components into viewable time series**
- **Web based portal provides real-time monitoring of simulation progress and convenient archive information on previous runs**

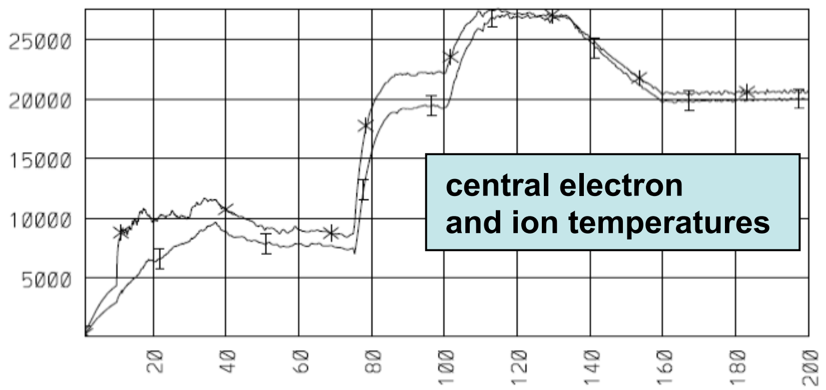
The IPS framework supports four levels of parallelism

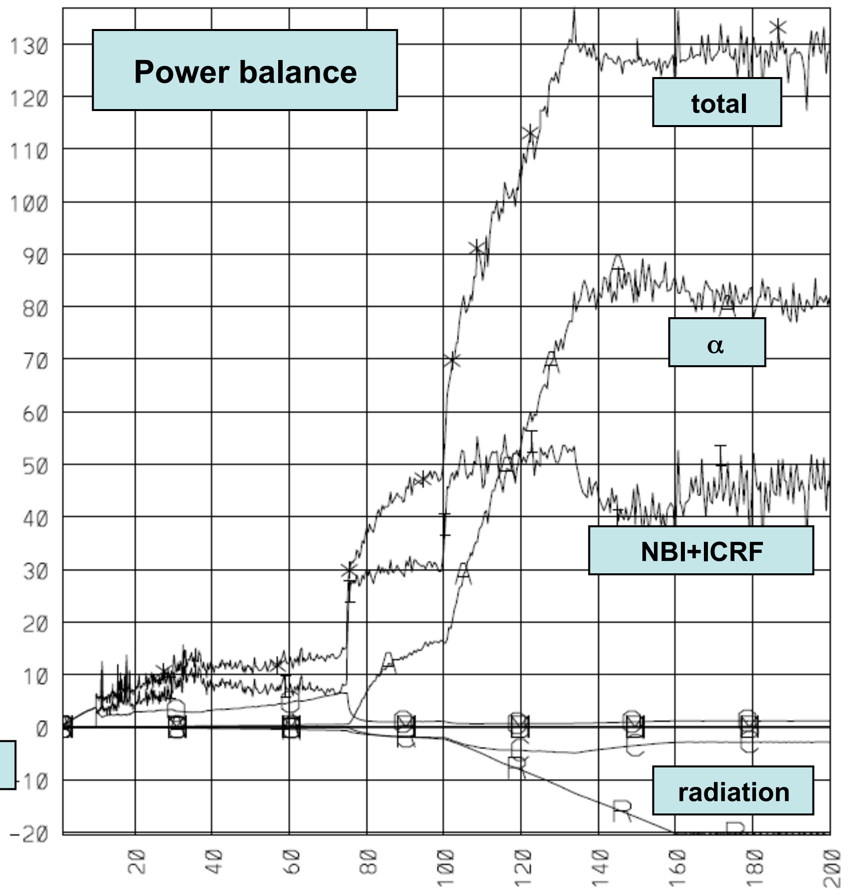
- Components can launch parallel jobs – MPI code execution
- A component can launch multiple tasks concurrently – e.g. multiple flux surfaces or toroidal mode numbers
- Multiple components can run concurrently
- Multiple simulations can be managed by the framework concurrently – share resources allocated to a single batch submission (task pool)

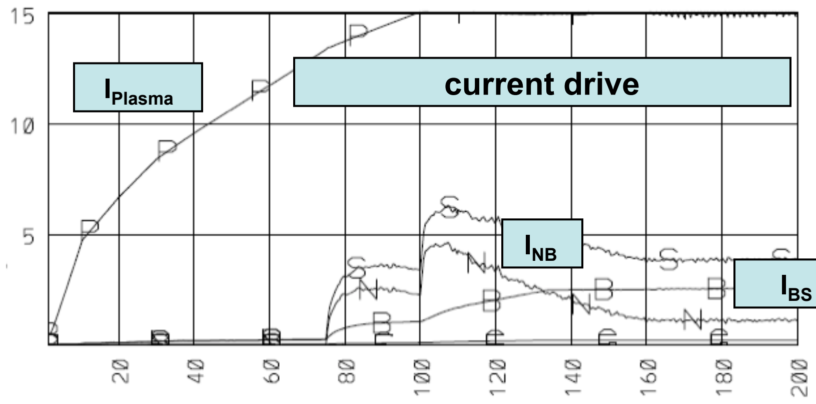


Many code executions –> one qsub, one que wait

When we started, a single long-time-scale ITER simulation, such as below, with minimal level of physics detail took up to 6 weeks with serial processor technology







TSC (Free-Boundary Equilibrium and Profile Advance)

TORIC (RF Ion Cyclotron) – 32 poloidal Fourier modes (poorly converged)

NUBEAM (neutral beam injection) – 1,000 Monte Carlo particles (poor statistics)

Demonstrating levels of parallelism – ITER simulations originally taking ~ 6 weeks with serial codes

Simulations at very high resolutions to show capability of massive parallelism

- TSC +AORSA + NUBEAM (1,000,000 particles/species)
- TSC + TORIC (255 poloidal modes) + NUBEAM (1,000,000 particles/species)
- *running times ~ 30 hr on 1600 cores*

Simulations at resolutions more typical of present practice for comparison

- ITER hybrid scenario
- TSC (1 core), TORIC (31 poloidal modes, 4 cores), NUBEAM (5,000 particles/species, 16 cores)
- Typically ramp-up from 1.5 sec into flattop 550 sec

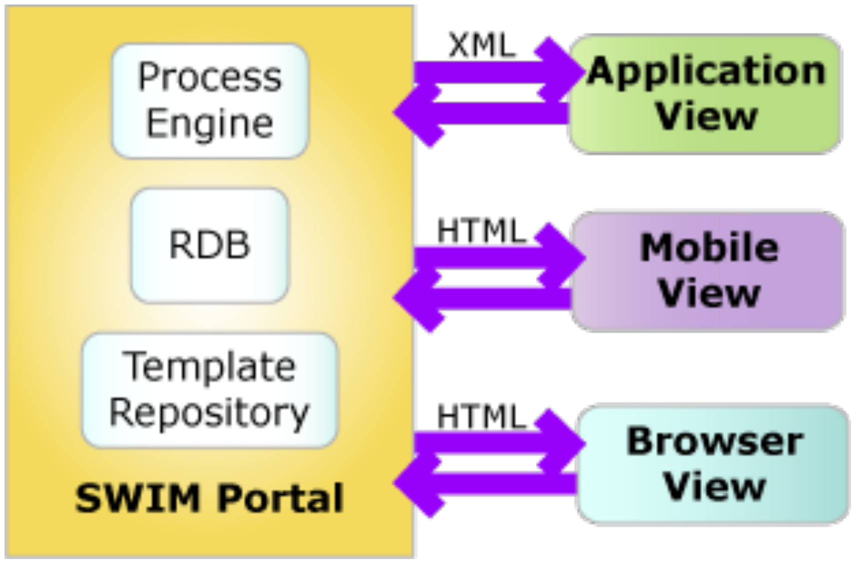
- TSC alone – using TSC internal (analytic) models for NBI and ICRF
  - *No parallelism, 1 core, running time ~ 11 hr*
- TORIC + NUBEAM + TSC – sequential execution of parallel components
  - *One level of parallelism, 16 cores, running time ~ 28 hr*
- TORIC + NUBEAM + TSC – concurrent execution of parallel components
  - *Two levels of parallelism, 24 cores, running time ~ 12 hr*
- Parameter study – pedestal location, pedestal height (chi pedestal)
  - Nine concurrent simulations run simultaneously
  - *Three levels of parallelism, 128 cores, running time ~ 16 hr*

Checkpoint/Restart – at the simulation and component level work together to save and restore the simulation at a given point

- String multiple runs together to continue computation from one run to another
- Use checkpoint/restart as a debugging tool for coupled simulations
- Run less interesting physics to a certain point and then examine the following physics phase in greater detail (with different components, configurations, multiple simulations with different parameters, etc.)
- Restructure or load balance simulation to use different components or numbers of processes

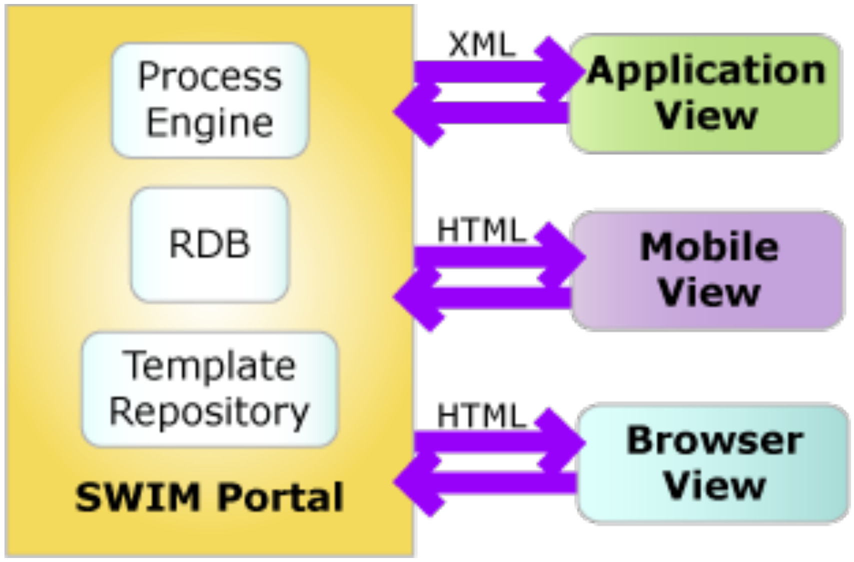
And of course recover from failures

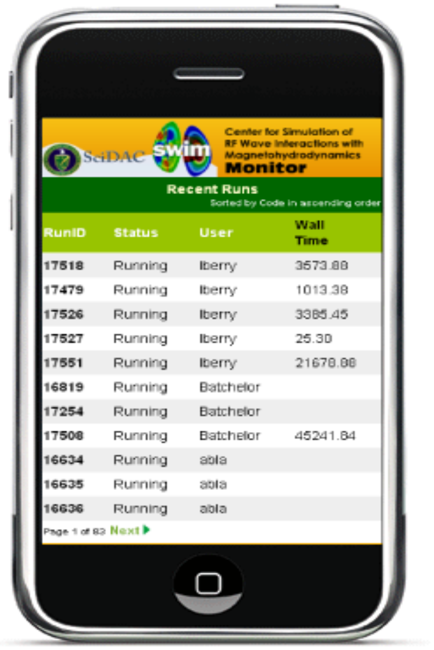
SWIM Portal Collects Information About SWIM Simulations and Serves Multiple Clients



- A variety of information being tracked and stored in Relational DB
  - User name, simulation name current status, code name, last time stamp, wall clock time, simulation comment, tokamak, host computer
  - Supports search and sort
- Each simulation is identified with unique RunID
- Using MDS+ for data storage
- Coming soon – access to experimental data archives, comparison with simulations

SWIM Portal Collects Information About SWIM Simulations and Serves Multiple Clients





Center for Simulation of RF Wave Interactions with Magnetohydrodynamics
Monitor

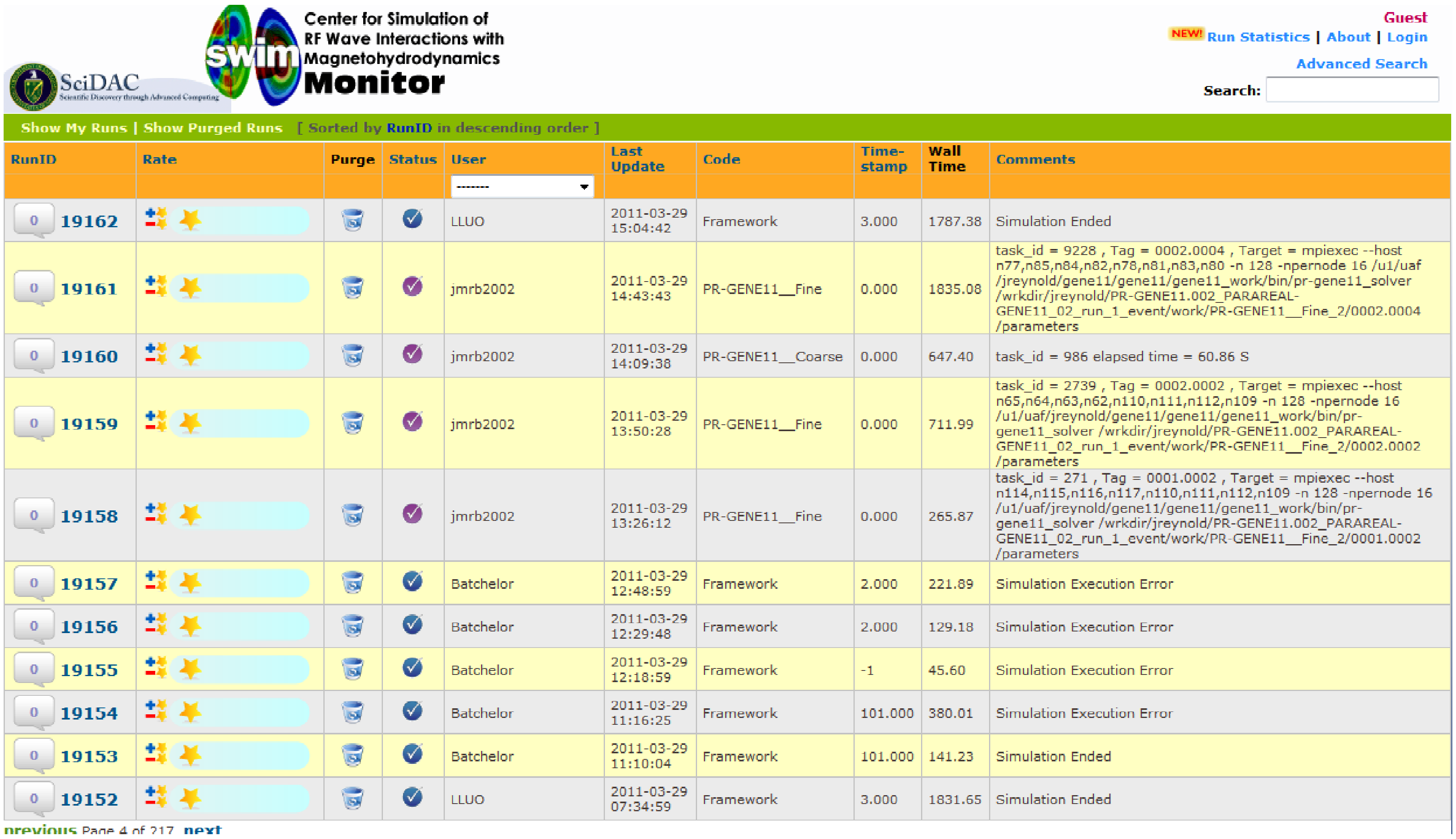
Recent Runs
Sorted by Code in ascending order

RunID	Status	User	Wall Time
17518	Running	lberry	3573.88
17479	Running	lberry	1013.38
17526	Running	lberry	3385.45
17527	Running	lberry	25.30
17551	Running	lberry	21678.88
16819	Running	Batchelor	
17254	Running	Batchelor	
17508	Running	Batchelor	45241.84
16634	Running	abla	
16635	Running	abla	
16636	Running	abla	

Page 1 of 83 Next

- A variety of information being tracked and stored in Relational DB
  - User name, simulation name current status, code name, last time stamp, wall clock time, simulation comment, tokamak, host computer
  - Supports search and sort
- Each simulation is identified with unique RunID
- Using MDS+ for data storage
- Coming soon – access to experimental data archives, comparison with simulations

Summary Page Provides Snapshot of Current Status of SWIM Simulations (http://swim.gat.com:8080/)


SciDAC Scientific Discovery through Advanced Computing

swim Center for Simulation of RF Wave Interactions with Magnetohydrodynamics **Monitor**

Guest
NEW! Run Statistics | About | Login
Advanced Search
Search:

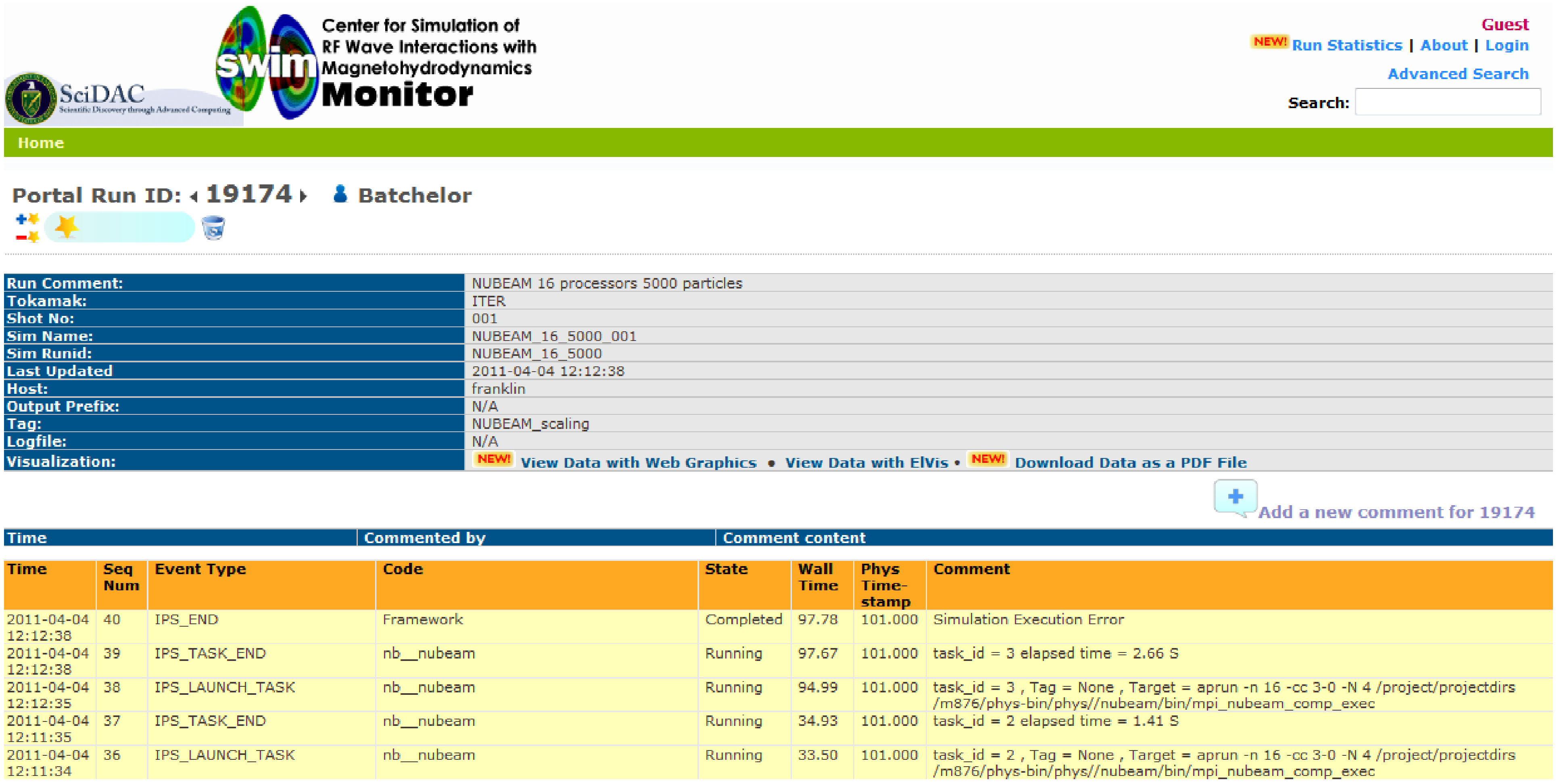
Show My Runs | Show Purged Runs [Sorted by RunID in descending order]

RunID	Rate	Purge	Status	User	Last Update	Code	Time-stamp	Wall Time	Comments
				--------					
0 19162			✓	LLUO	2011-03-29 15:04:42	Framework	3.000	1787.38	Simulation Ended
0 19161			✓	jmrb2002	2011-03-29 14:43:43	PR-GENE11__Fine	0.000	1835.08	task_id = 9228 , Tag = 0002.0004 , Target = mpiexec --host n77,n85,n84,n82,n78,n81,n83,n80 -n 128 -npernode 16 /u1/uaf /jreynold/gene11/gene11/gene11_work/bin/pr-gene11_solver /wrkdir/jreynold/PR-GENE11.002_PARAREAL-GENE11_02_run_1_event/work/PR-GENE11__Fine_2/0002.0004 /parameters
0 19160			✓	jmrb2002	2011-03-29 14:09:38	PR-GENE11__Coarse	0.000	647.40	task_id = 986 elapsed time = 60.86 S
0 19159			✓	jmrb2002	2011-03-29 13:50:28	PR-GENE11__Fine	0.000	711.99	task_id = 2739 , Tag = 0002.0002 , Target = mpiexec --host n65,n64,n63,n62,n110,n111,n112,n109 -n 128 -npernode 16 /u1/uaf/jreynold/gene11/gene11/gene11_work/bin/pr-gene11_solver /wrkdir/jreynold/PR-GENE11.002_PARAREAL-GENE11_02_run_1_event/work/PR-GENE11__Fine_2/0002.0002 /parameters
0 19158			✓	jmrb2002	2011-03-29 13:26:12	PR-GENE11__Fine	0.000	265.87	task_id = 271 , Tag = 0001.0002 , Target = mpiexec --host n114,n115,n116,n117,n110,n111,n112,n109 -n 128 -npernode 16 /u1/uaf/jreynold/gene11/gene11/gene11_work/bin/pr-gene11_solver /wrkdir/jreynold/PR-GENE11.002_PARAREAL-GENE11_02_run_1_event/work/PR-GENE11__Fine_2/0001.0002 /parameters
0 19157			✓	Batchelor	2011-03-29 12:48:59	Framework	2.000	221.89	Simulation Execution Error
0 19156			✓	Batchelor	2011-03-29 12:29:48	Framework	2.000	129.18	Simulation Execution Error
0 19155			✓	Batchelor	2011-03-29 12:18:59	Framework	-1	45.60	Simulation Execution Error
0 19154			✓	Batchelor	2011-03-29 11:16:25	Framework	101.000	380.01	Simulation Execution Error
0 19153			✓	Batchelor	2011-03-29 11:10:04	Framework	101.000	141.23	Simulation Ended
0 19152			✓	LLUO	2011-03-29 07:34:59	Framework	3.000	1831.65	Simulation Ended

previous Page 4 of 217 next

Each job has unique run-ID. Run data archived in MDS+. Search function allows rapid searching through the ~29,000 SWIM runs stored in the system

Simulation Details Page Enables You Examine the Every Step of Your Run – *as it runs*



Center for Simulation of RF Wave Interactions with Magnetohydrodynamics **Monitor**

SciDAC Scientific Discovery through Advanced Computing

Guest

NEW! Run Statistics | About | Login

Advanced Search

Search:

Home

Portal Run ID: 19174 Batchelor

Run Comment:	NUBEAM 16 processors 5000 particles
Tokamak:	ITER
Shot No:	001
Sim Name:	NUBEAM_16_5000_001
Sim Runid:	NUBEAM_16_5000
Last Updated	2011-04-04 12:12:38
Host:	franklin
Output Prefix:	N/A
Tag:	NUBEAM_scaling
Logfile:	N/A
Visualization:	NEW! View Data with Web Graphics • View Data with ElVis • NEW! Download Data as a PDF File

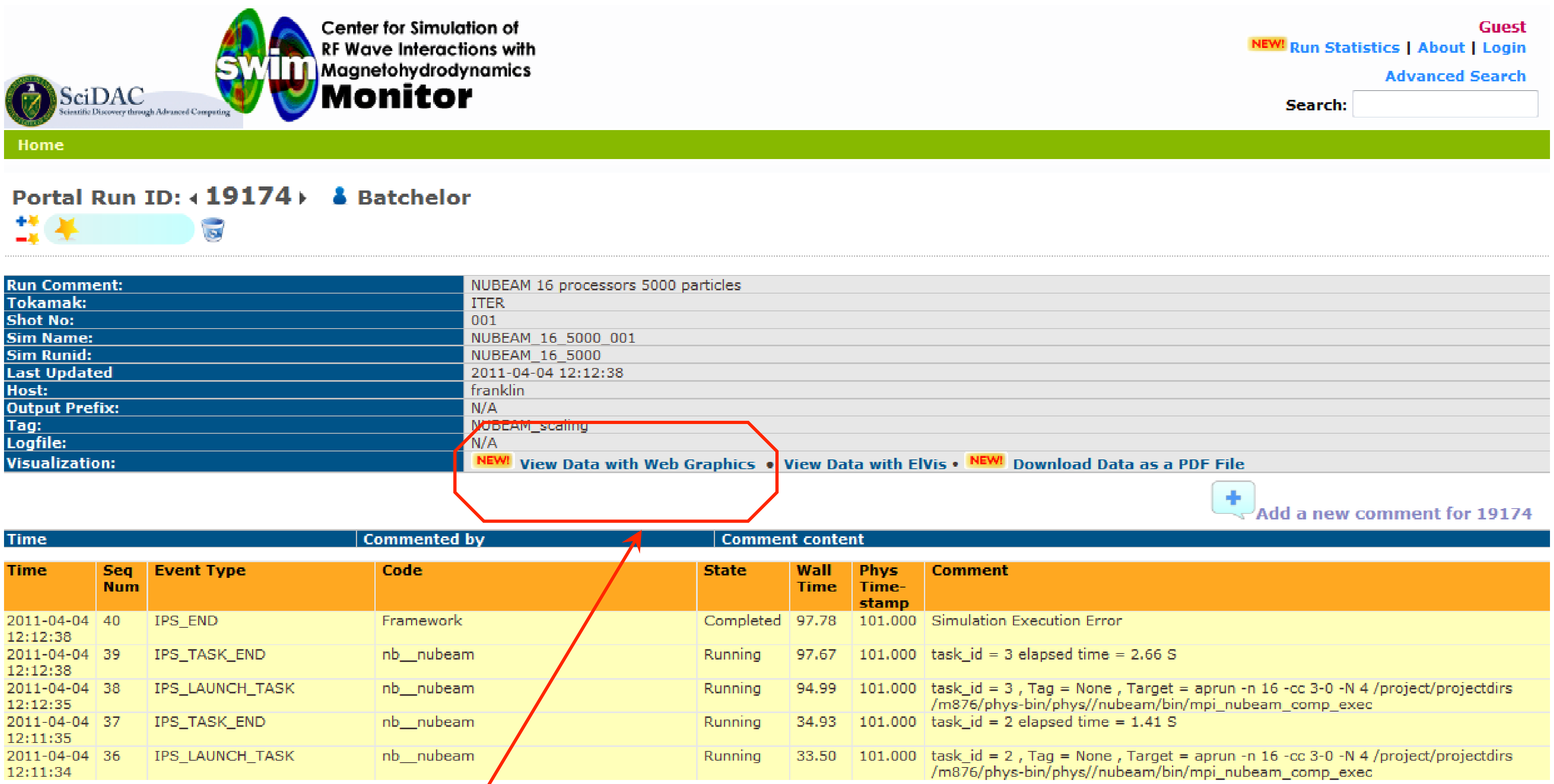
Add a new comment for 19174

Time	Commented by	Comment content

Time	Seq Num	Event Type	Code	State	Wall Time	Phys Time-stamp	Comment
2011-04-04 12:12:38	40	IPS_END	Framework	Completed	97.78	101.000	Simulation Execution Error
2011-04-04 12:12:38	39	IPS_TASK_END	nb__nubeam	Running	97.67	101.000	task_id = 3 elapsed time = 2.66 S
2011-04-04 12:12:35	38	IPS_LAUNCH_TASK	nb__nubeam	Running	94.99	101.000	task_id = 3 , Tag = None , Target = aprun -n 16 -cc 3-0 -N 4 /project/projectdirs /m876/phys-bin/phys//nubeam/bin/mpi_nubeam_comp_exec
2011-04-04 12:11:35	37	IPS_TASK_END	nb__nubeam	Running	34.93	101.000	task_id = 2 elapsed time = 1.41 S
2011-04-04 12:11:34	36	IPS_LAUNCH_TASK	nb__nubeam	Running	33.50	101.000	task_id = 2 , Tag = None , Target = aprun -n 16 -cc 3-0 -N 4 /project/projectdirs /m876/phys-bin/phys//nubeam/bin/mpi_nubeam_comp_exec

Clicking on a run-ID gives a run details page including the ability to view data with web graphics

Simulation Details Page Enables You Examine the Every Step of Your Run – *as it runs*



Center for Simulation of RF Wave Interactions with Magnetohydrodynamics Monitor

Guest

NEW! Run Statistics | About | Login

Advanced Search

Search:

Home

Portal Run ID: 19174 Batchelor

Run Comment:	NUBEAM 16 processors 5000 particles
Tokamak:	ITER
Shot No:	001
Sim Name:	NUBEAM_16_5000_001
Sim Runid:	NUBEAM_16_5000
Last Updated	2011-04-04 12:12:38
Host:	franklin
Output Prefix:	N/A
Tag:	NUBEAM_scaling
Logfile:	N/A
Visualization:	NEW! View Data with Web Graphics • View Data with ElVis • NEW! Download Data as a PDF File

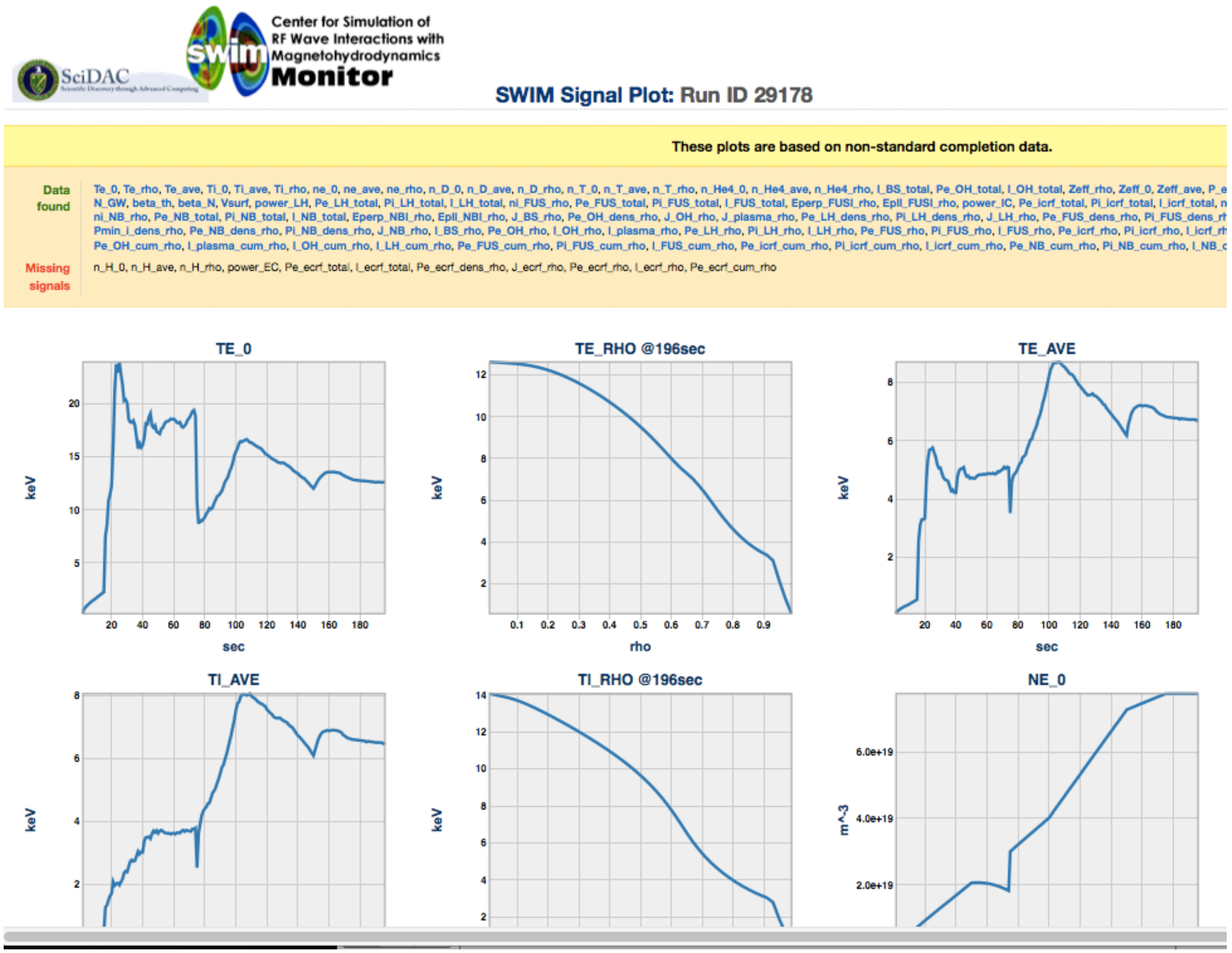
Add a new comment for 19174

Time	Commented by	Comment content

Time	Seq Num	Event Type	Code	State	Wall Time	Phys Time-stamp	Comment
2011-04-04 12:12:38	40	IPS_END	Framework	Completed	97.78	101.000	Simulation Execution Error
2011-04-04 12:12:38	39	IPS_TASK_END	nb__nubeam	Running	97.67	101.000	task_id = 3 elapsed time = 2.66 S
2011-04-04 12:12:35	38	IPS_LAUNCH_TASK	nb__nubeam	Running	94.99	101.000	task_id = 3 , Tag = None , Target = aprun -n 16 -cc 3-0 -N 4 /project/projectdirs /m876/phys-bin/phys//nubeam/bin/mpi_nubeam_comp_exec
2011-04-04 12:11:35	37	IPS_TASK_END	nb__nubeam	Running	34.93	101.000	task_id = 2 elapsed time = 1.41 S
2011-04-04 12:11:34	36	IPS_LAUNCH_TASK	nb__nubeam	Running	33.50	101.000	task_id = 2 , Tag = None , Target = aprun -n 16 -cc 3-0 -N 4 /project/projectdirs /m876/phys-bin/phys//nubeam/bin/mpi_nubeam_comp_exec

Clicking on a run-ID gives a run details page including the ability to view data with web graphics

Can view all of the plots in the latest monitor file with one click. Can download monitor file and get pdf hardcopy of all plots



SciDAC

Center for Simulation of RF Wave Interactions with Magnetohydrodynamics

Monitor

SWIM Signal Plot: Run ID 29178

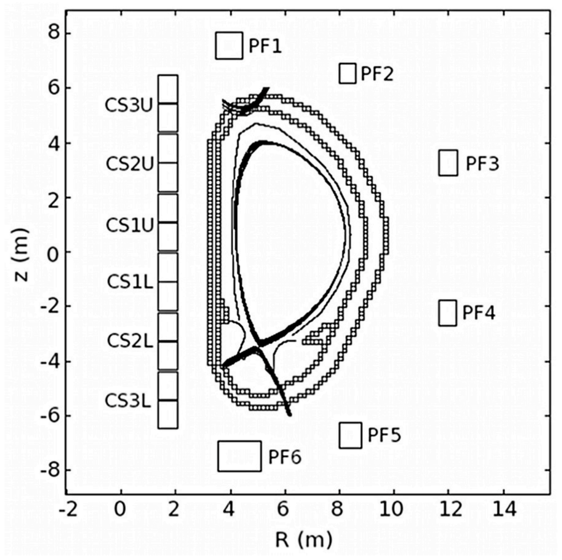
These plots are based on non-standard completion data.

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, n_He4_0, n_He4_ave, n_He4_rho, I_BS_total, Pe_OH_total, I_OH_total, Zeff_rho, Zeff_0, Zeff_ave, P_e N_GW, beta_th, beta_N, Vsurf, power_LH, Pe_LH_total, Pi_LH_total, I_LH_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, n 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_LH_dens_rho, Pi_LH_dens_rho, J_LH_rho, Pe_FUS_dens_rho, Pi_FUS_dens_rh Pmin_i_dens_rho, Pe_NB_dens_rho, Pi_NB_dens_rho, J_NB_rho, I_BS_rho, Pe_OH_rho, I_OH_rho, I_plasma_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_rh Pe_OH_cum_rho, I_plasma_cum_rho, I_OH_cum_rho, I_LH_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_c
Missing signals	n_H_0, n_H_ave, n_H_rho, power_EC, Pe_ecrf_total, I_ecrf_total, Pe_ecrf_dens_rho, J_ecrf_rho, Pe_ecrf_rho, I_ecrf_rho, Pe_ecrf_cum_rho

Physics studies with IPS

- **ITER discharge simulations with massively-parallel RF and neutral beam components**
- **Use of IPS to study ECCD resistive tearing mode stabilization and motion of flux surfaces – coupling to GENRAY ECH ray tracing to NIMROD nonlinear MHD**
- **Use of IPS to study parallelization in time *(parareal algorithm)* of DTEM turbulence, 1.5D transort**
- **Studies of RF driven energetic tail formation on Alcator C-mod**
- **Onset of saturated n = 1, m = 1,2 modes in NSTX – coupling of IPS to M3D**
- **Use of IPS to study control of sawtooth onset time with lower hybrid waves on C-mod**

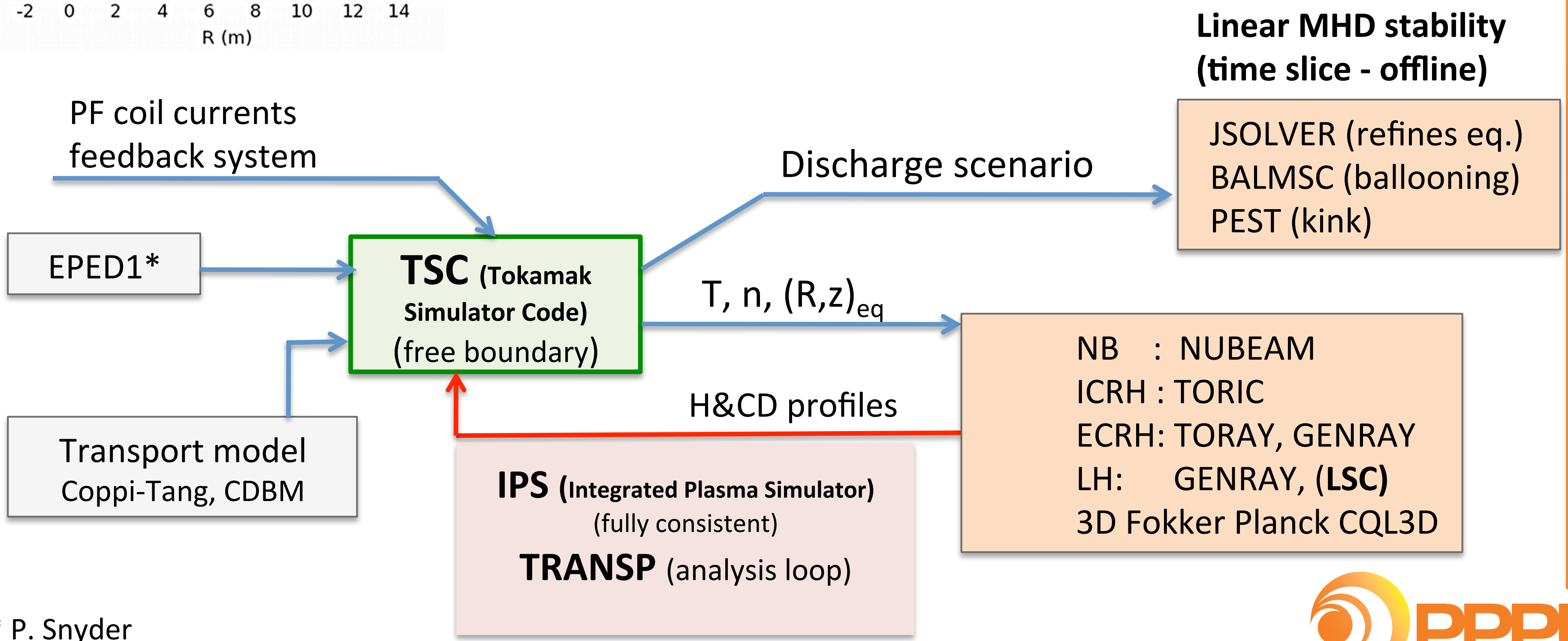
Time-dependent simulations evolve plasma equilibrium and H&CD source profiles consistently



Target plasma

R=6.2, a=2.0, κ=1.8, δ~0.45

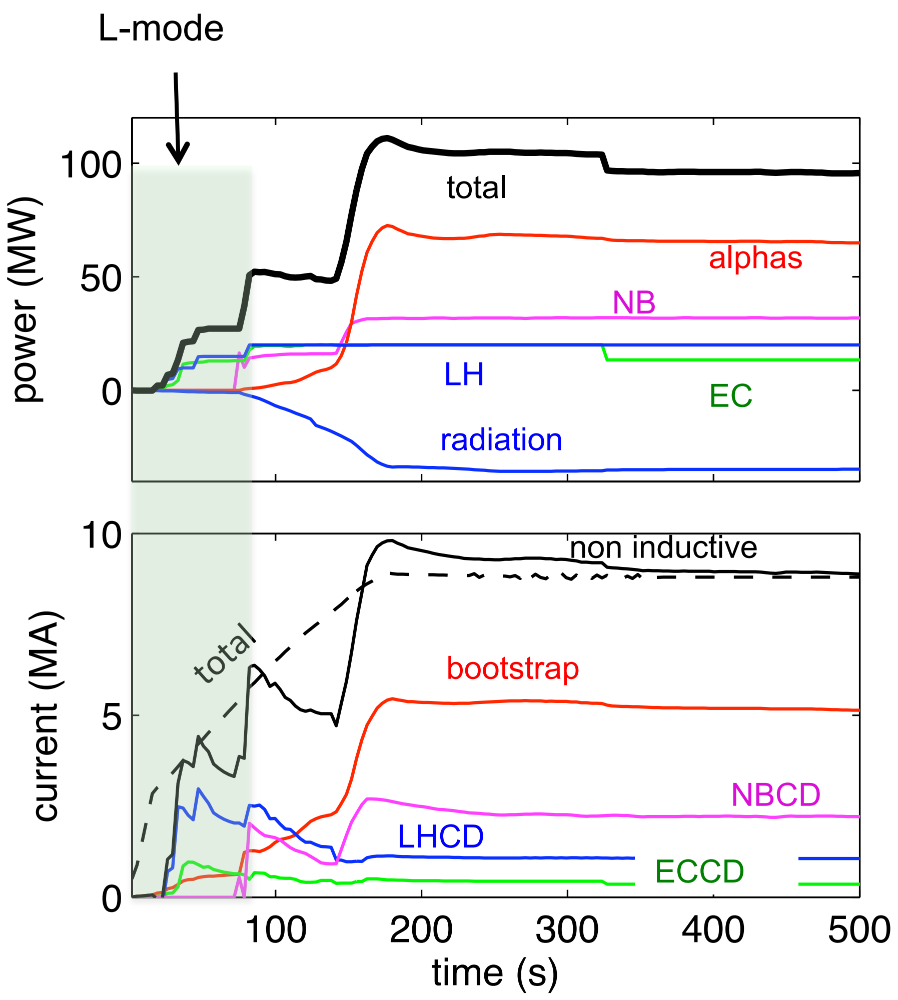
n/n_G>0.75



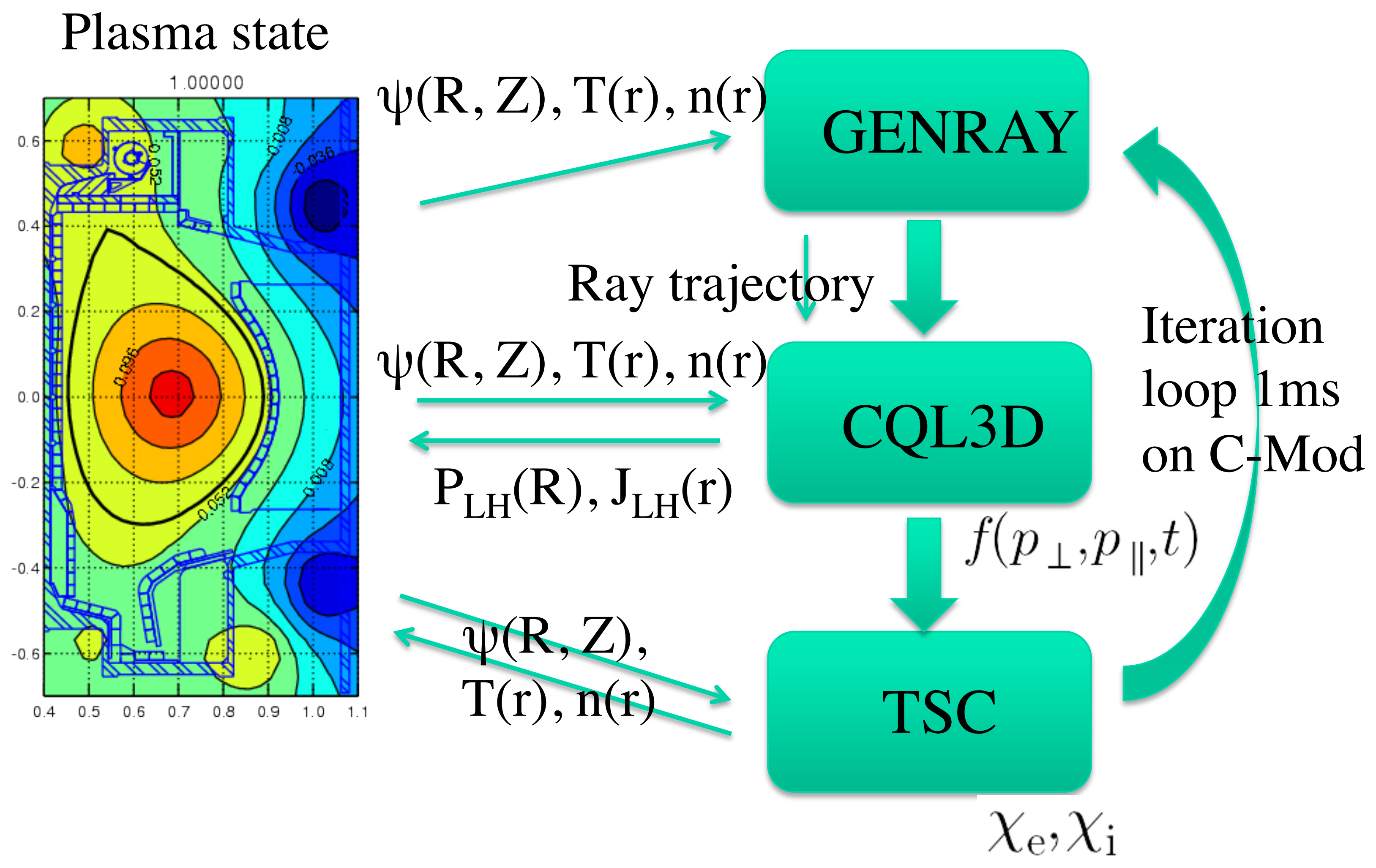
* P. Snyder

Simulate rampup and relaxation in flattop to self-consistently study kinetic profiles and MHD stability evolution

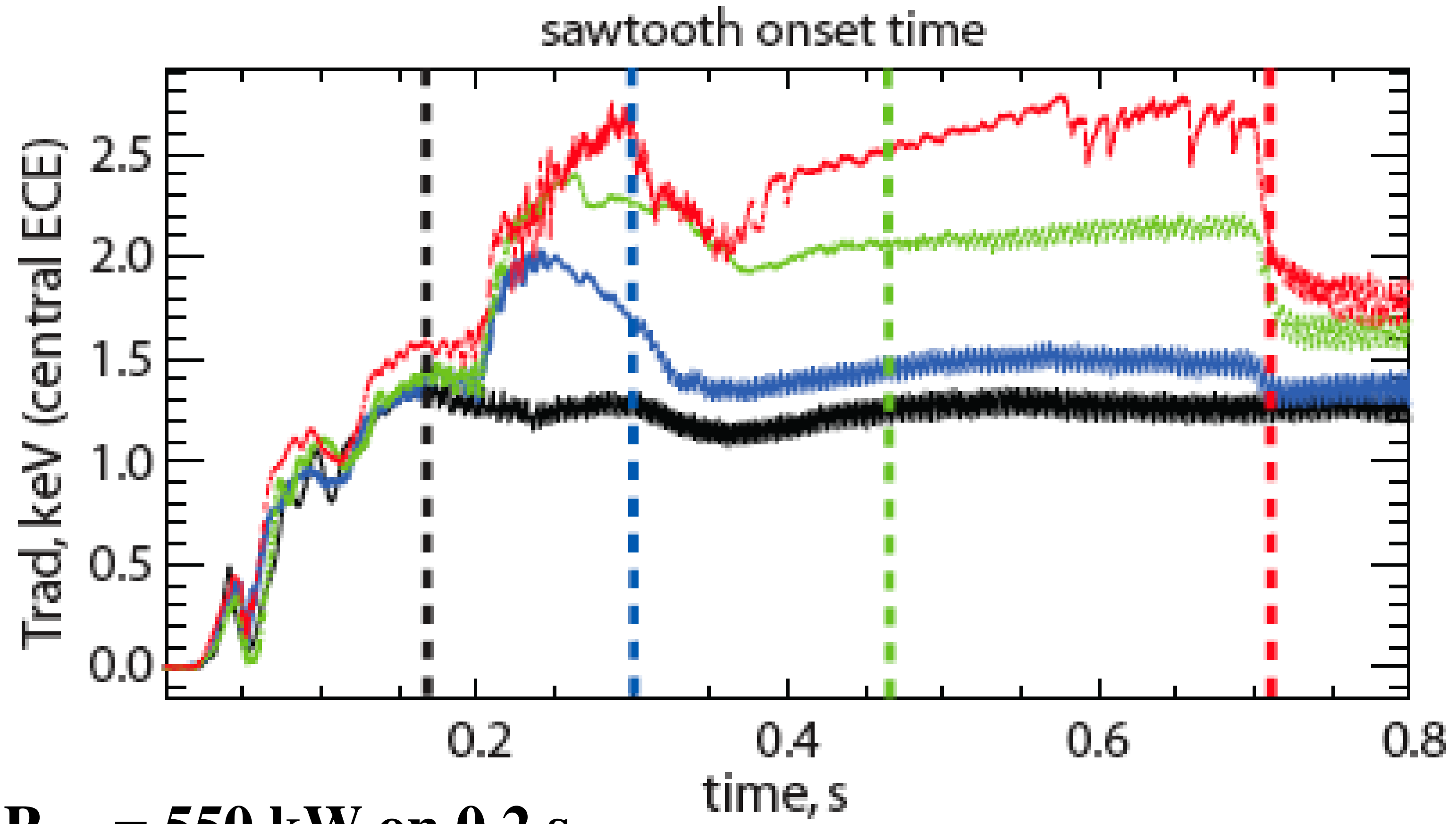
- Ramp-up phase
  - RF to form reverse shear profiles
  - Inductive rampup still important
- Flat-top phase
  - 100% non-inductive current
  - Sustain moderate reverse shear
- Radiated power keeps divertor loads within acceptable levels



GENRAY/CQL3D are advanced in a “tight” time loop on the transport time scale using the IPS



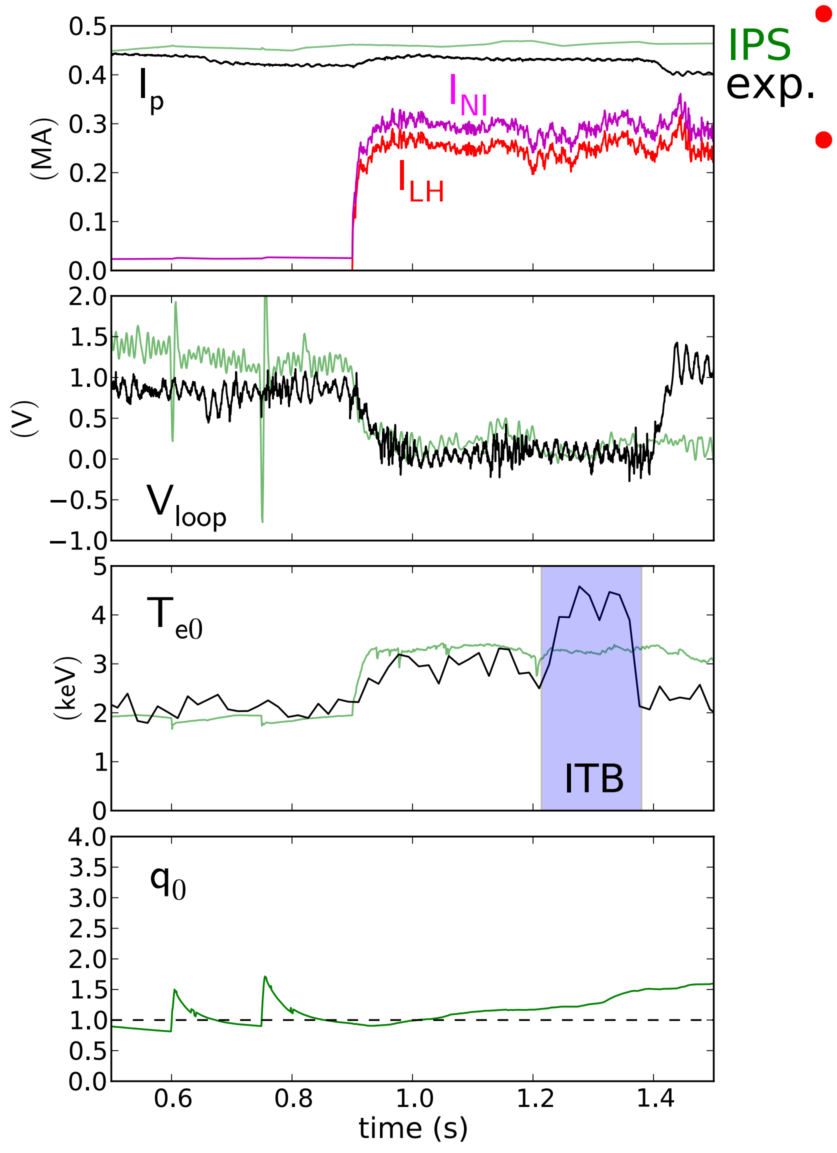
Off-axis LHCD was used to delay the onset of sawteeth during current ramp up in C-Mod [7]



P_{LH} = 550 kW on 0.2 s

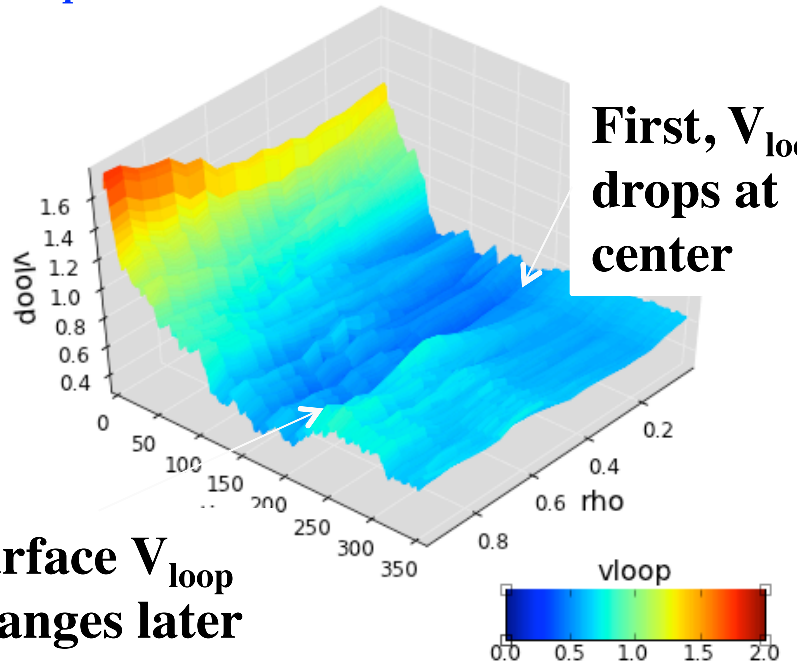
n_e=4 × 10^{19}m^{-3} (red)
n_e=9 × 10^{19}m^{-3} (blue)
n_e=7 × 10^{19}m^{-3} (green)
Ohmic (black) - 9 × 10^{19}m-3

Time-domain discharge simulation (TSC) coupled with RF (GENRAY) and FP (CQL3D) compared with C-Mod nearly full NI LHCD discharge using IPS



- **Good reproduction of experimental plasma parameters, such as T_e, and V_{loop}**
- **$q_0 > 1$ after 1s, while sawtooth was suppressed at ~0.98s in the experiment, suggesting the current profile evolution is modeled consistently**

V_{loop} evolution after LH turn-on



First, V_{loop} drops at center

Surface V_{loop} changes later

(S. Shiraiwa, P. T. Bonoli)

Many different workflows have been implemented in IPS

Conventional time loop controlled by driver component

- **Transport simulations with multiple source components – ITER, Cmod, NSTX**

Time loop controlled by one of the components

- **Coupling of NIMROD with GENRAY to study ECCD stabilization of tearimg modes**

Iteration loop, rather than time loop

- **FASTRAN with TGLF transport and source components iterating to self consistency**

Concurrent execution within component

- **Full ICRF antennal toroidal mode spectrum concurrent with TORIC**

Parareal – hybrid time/iteration loop with logic embedded in components.

- **An iterative algorithm for parallelization over time**
- **Time domain divided into chunks, iterative correction of time chunks run concurrently**
- **Multiple time chunks at different iteration levels running concurrently**
- ***Debasmita Samaddar is the expert***

Many different workflows have been implemented in IPS

Conventional time loop controlled by driver component

- **Transport simulations with multiple source components – ITER, Cmod, NSTX**

Time loop controlled by one of the components

- **Coupling of NIMROD with GENRAY to study ECCD stabilization of tearing**

Optimization and paramater scans with IPS/Dakota

I

- **consistency**

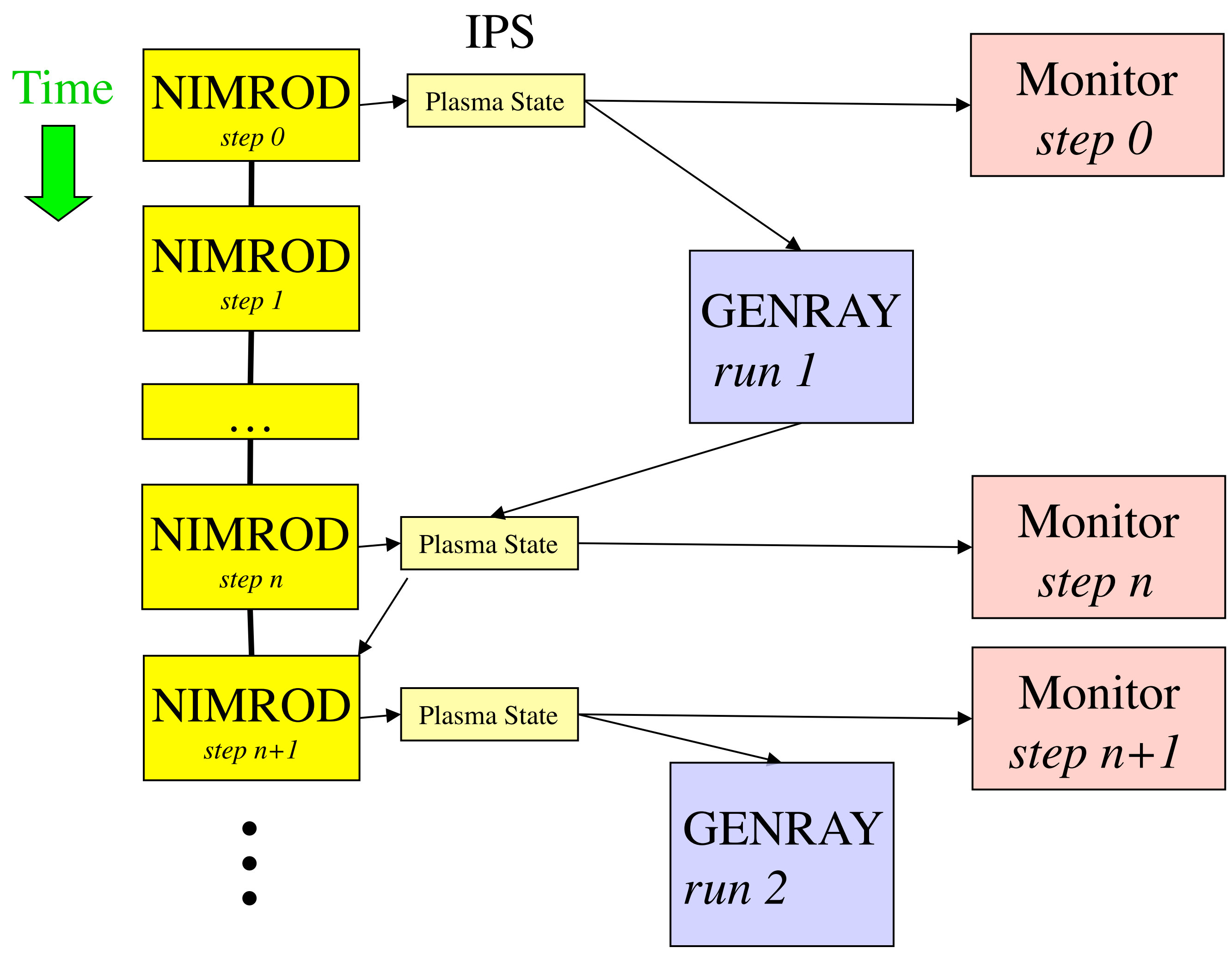
Concurrent execution within component

- **Full ICRF antennal toroidal mode spectrum concurrent with TORIC**

Parareal – hybrid time/iteration loop with logic embedded in components.

- **An iterative algorithm for parallelization over time**
- **Time domain divided into chunks, iterative correction of time chunks run concurrently**
- **Multiple time chunks at different iteration levels running concurrently**
- ***Debasmita Samaddar is the expert***

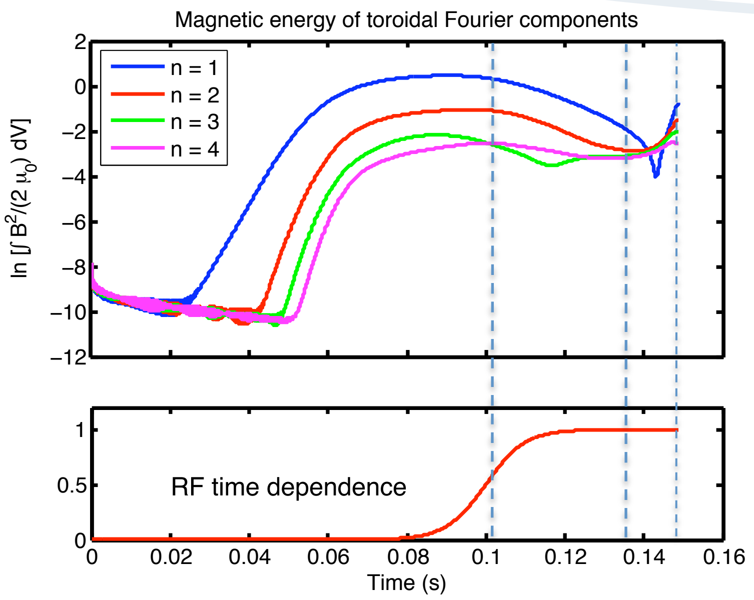
NIMROD/GENRAY coupling in IPS – NIMROD is run as a service, but controls time loop via simulation *event handling*



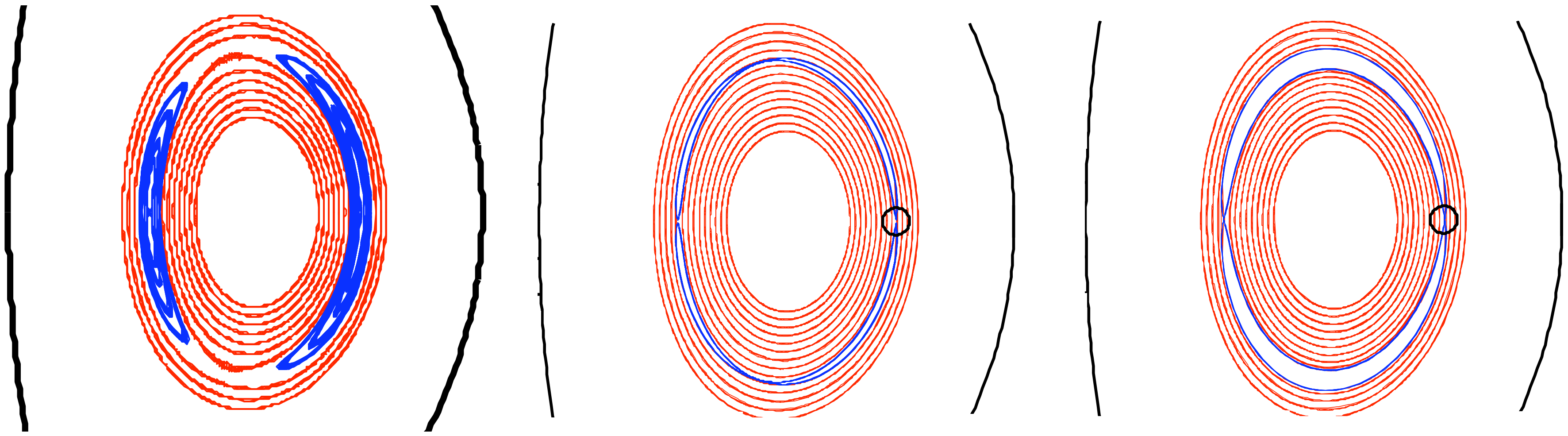
- **NIMROD exports magnetic geometry and n,T profiles to Plasma State**
- **GENRAY then calculates RF propagation and power deposition; exporting these quantities to the Plasma State**
- **NIMROD converts GENRAY data into momentum and energy source terms.**
- **Ultimately will include kinetic closure model**

Two levels of parallelism – parallel NIMROD run concurrently with GENRAY

In the absence of toroidal rotation, fixed RF stabilizes the resistive TM only temporarily

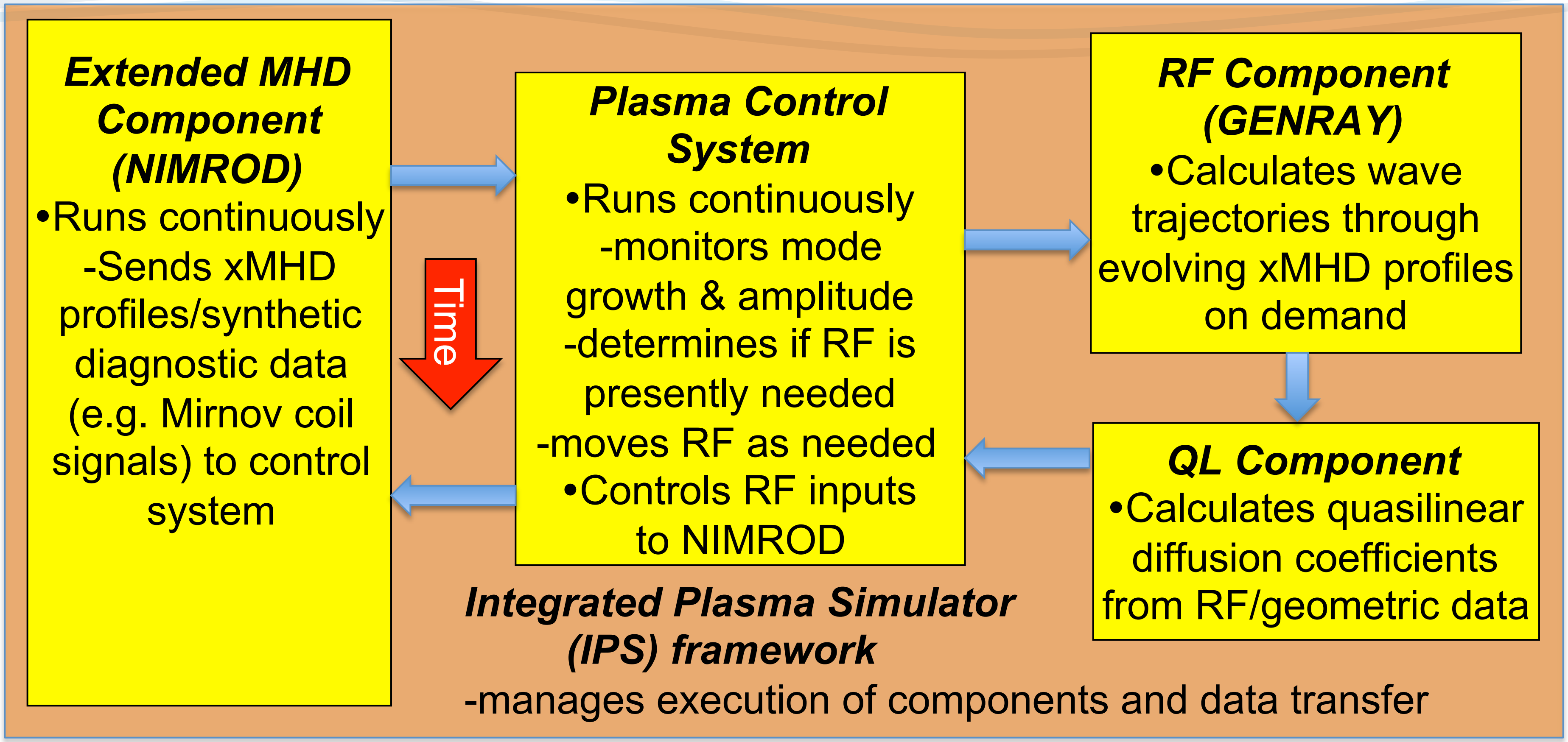


- Inject RF at O-point of saturated (2,1) island
- (4,2) island forms, mode energy decreases (stabilization?)
- (2,1) island with different O-point grows up again



- Here, island size and RF hotspot size are initially comparable.

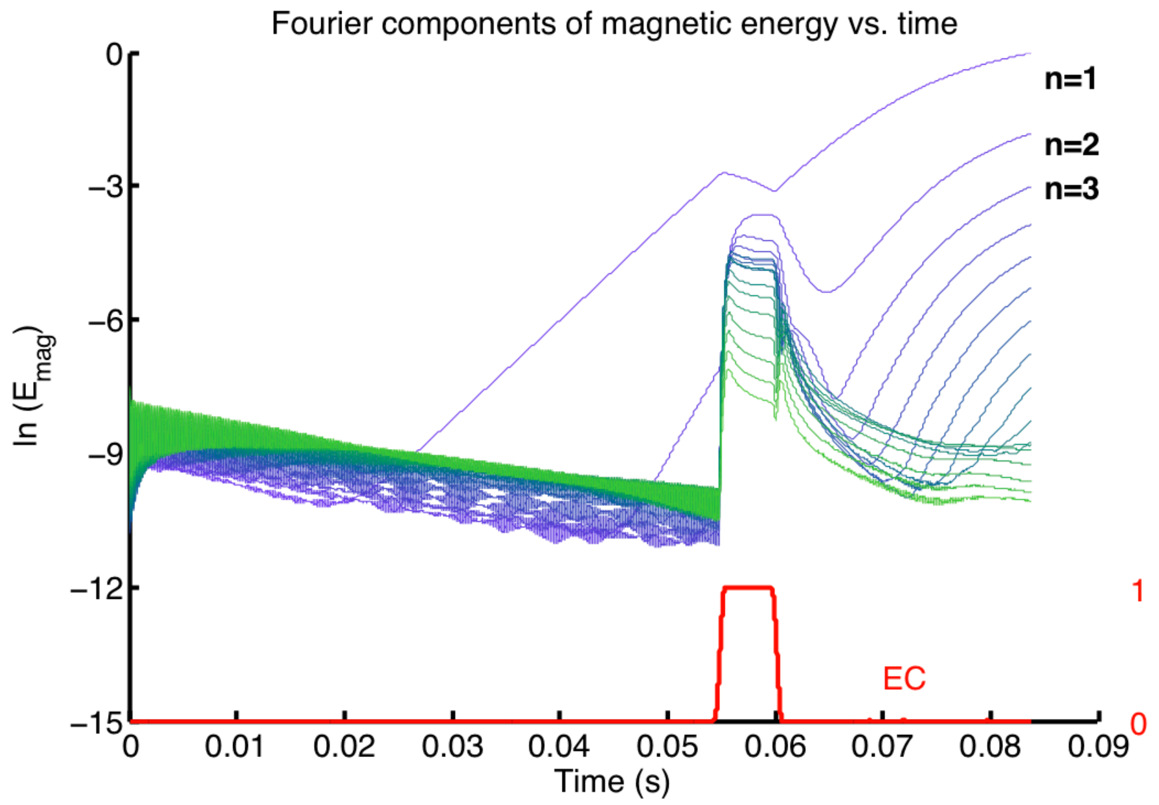
The control system is an additional physics component in the coupled simulation


Extended MHD Component (NIMROD)
•Runs continuously
-Sends xMHD profiles/synthetic diagnostic data (e.g. Mirnov coil signals) to control system
Time
Plasma Control System
•Runs continuously
-monitors mode growth & amplitude
-determines if RF is presently needed
-moves RF as needed
•Controls RF inputs to NIMROD
RF Component (GENRAY)
•Calculates wave trajectories through evolving xMHD profiles on demand
QL Component
•Calculates quasilinear diffusion coefficients from RF/geometric data
Integrated Plasma Simulator (IPS) framework
-manages execution of components and data transfer

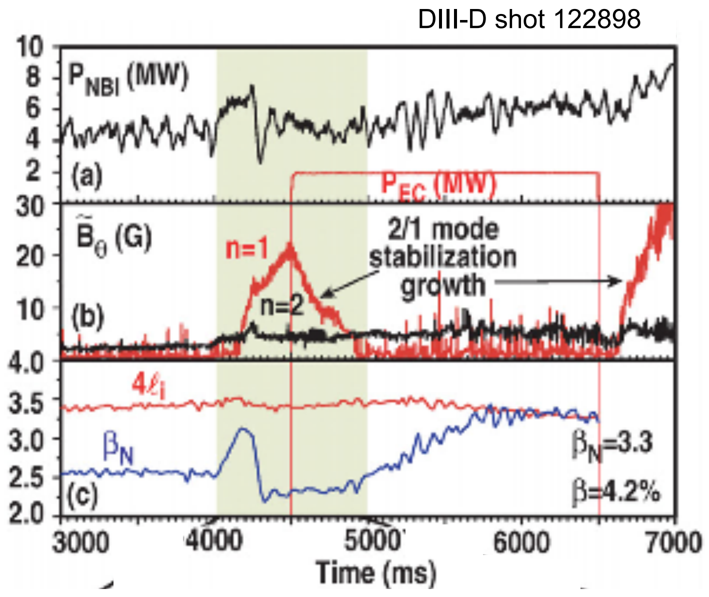
•All physics components run in a larger simulation framework (IPS)

•Explicit coupling exploits the timescale separation between RF and xMHD

Initial results of coupled simulations are promising



- •Control system aligns RF, halts mode growth, shrinks island.
- •Growth resumes when RF is shut off.



What do you need to run an IPS simulation?

Access to IPS	Casual user: Use common build Power User: Make your own. Check out from svn repository. (Available open sourceo n Source Forge) Do top level make
A set of physics components	Physics executables + wrapper codes to connect to IPS and access Plasma State Casual user: Use existing physics components Power User: Add new, or modify existing component Get binary of physics executable from code developer (maybe you) Adapt existing component wrappers
Standard input files for physics codes	These are just the input files you would use to run the code stand-alone. Casual user/power user: modify an existing file
A simulation configuration file	A text file that specifies: meta-data about the simulations (run name, run directory, files constituting plasma state…), and configuration of individual components (specific implementation of component, input data path, top-level operational parameters …) Casual user/power user: modify an existing file
A batch submission file	A simple (~10 lines) script specifying number of processors, and config file for this simulation, … Casual user/power user: modify an existing file

What is an IPS run?

- Submit batch run – qsub
- Sit back and watch your run go on web portal (or get coffee)

What do you get? A simulation run directory containing everything This includes:

- Simulation configuration file used for run – provenance
- All of the input files used to create the run – provenance
- All of the Python component scripts used in the run – provenance
- All of the output files generated by all of the components at each time-step, iteration, or whatever. Under user control.
- All of the files declared to be plasma state for each time-step
- If using SWIM Plasma State and monitor component you get selected plasma state data aggregated into time history in a single file.

IPS is the inverse of FSP(R.I.P.) – bottom up rather than top down

- **We have a small amount of OFES funding to keep IPS functional, maintain the GA web portal, support users, make minor improvements as required by users**
- **We have a number of users from outside the SWIM project – PPPL, MIT, JET (Samaddar), GA**
- **We have users outside fusion – lithium battery simulation**
- **There is tremendous flexibility available within the IPS, but using it is optional**
- **IPS is simple to use if you want to run a previously set up workflow**
- **IPS is useful for constructing workflows even for small, serial codes, but it's real strength is in effectively coupling large, highly parallel modules**

Summary of IPS framework

- **SWIM has emphasized, coupling of large-scale, coarse-granularity components → *I believe that integrated simulations will always have some element of such coarse granularity, especially useful for runtime debugging***
- **IPS written in Python, no third party libraries → *runs anywhere (although complicated to build Plasma State, requires NTCC libraries)***
- **Have demonstrated a wide variety of simulation work-flows**

What we haven't done

- **No formal/automated regression testing**
- **Sketchy data management and provenance tracking – but improving**
- **Tight coupling → *it hasn't come up yet at inter-component level, support tightly coupled, composite components***
- **No GUI → *although working with SECAD SBIR to look at automated config***

Lessons learned about integrated simulation

- The system works – we' ve been able to get multiple, MPP codes to play together nice
- You can get quite far with loose coupling and file-based communication. Even the Slow MHD campaign, *non-linear MHD ↔ ECH RF*, is not requiring close, in-memory coupling.
- But some things should be coupled in memory – we learned early on that iteration of 1.5D transport, transport coefficient calculation, and MHD equilibrium must be coupled in memory. *→ composite, implicit, tightly coupled component*
- Because components vary widely in their parallelizability (RF solvers → 1,000s - 10,000s of processors, 1.5D transport ~ 1 processor), load balancing is a major issue. *→ IPS framework manages processor resource pool, allows multiple concurrent executions*
- New ways of using component codes, even widely used and thoroughly tested ones, uncovers new bugs and failure modes, *→ It takes a lot longer to get things going than you would have thought, but then "no pain, no gain"*

Extra slides

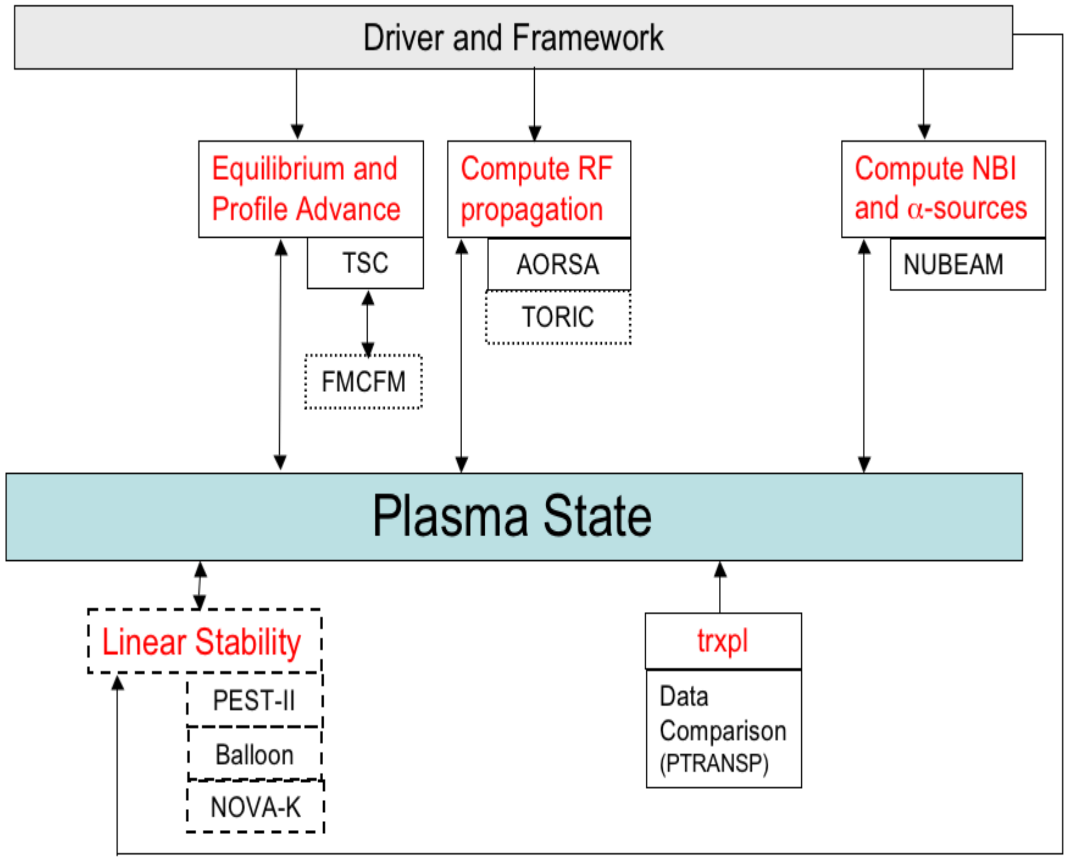
Goal: demonstrate the use of massively-parallel computers to accelerate ITER simulations, while improving the level of physics fidelity of the simulations.

Coupling of TSC (Free-Boundary Equilibrium and Profile Advance)

AORSA (massively parallel RF Ion Cyclotron solver) - 256×256 poloidal Fourier modes

TORIC (semi-spectral ICRF solver) – 147 poloidal modes, 409 radial nodes

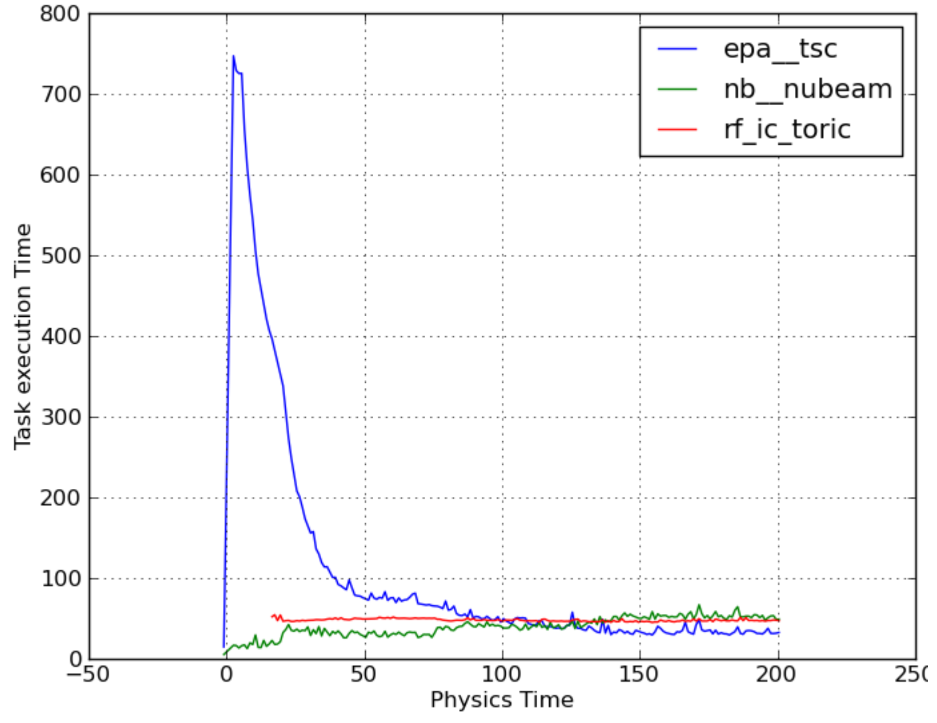
NUBEAM (parallel neutral beam injection) – 1,000,000 Monte Carlo particles



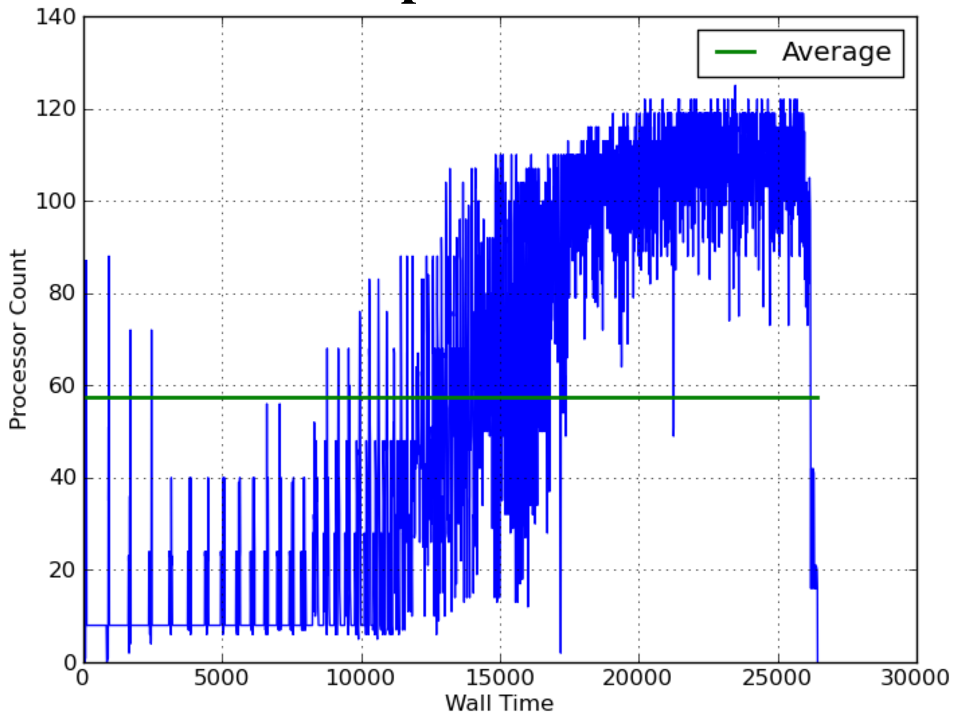
These simulations benefit from component level concurrency to minimize time in (near) serial operations

Component execution times and task processor usage for 9 interleaved simulations on NERSC Franklin

Execution time for IPS tasks



Processor utilization for 9-simulation parameter scan



- **Average processor usage for first 200 sec of simulation is about 58%. Is this good?**
- **Can I know how many simultaneous simulations to run and how many cores to use?**
- **A resource usage simulator (RUS) was created to simulate resource utilization of SWIM workloads – gives guidance for choosing processor count/component, number of simulations, etc**

Tales from the *parareal* – simple algorithm that allows parallelization in time *sometimes* (J. Lyons, Y. Mayday, G. Turinici, CR Acad. Sci. I – Math 332, (2001), 661-668)

Consider time evolution problem: $\dfrac{du}{dt} = F(u), \quad u(0) = u_0$

Define: $T_n = n\Delta T, \quad u_n = u(T_n)$

Assume have two time advance operators:

$F_{n,\Delta T}$ **fine – accurate but takes a long time to run** $u_{n+1} = F_{n,\Delta T}(u_n)$

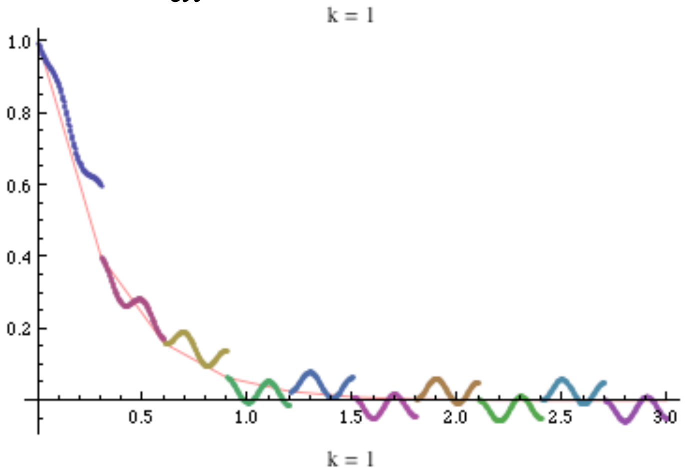
$G_{n,\Delta T}$ **coarse – inaccurate but runs very quickly** $u_{n+1} \sim G_{n,\Delta T}(u_n)$

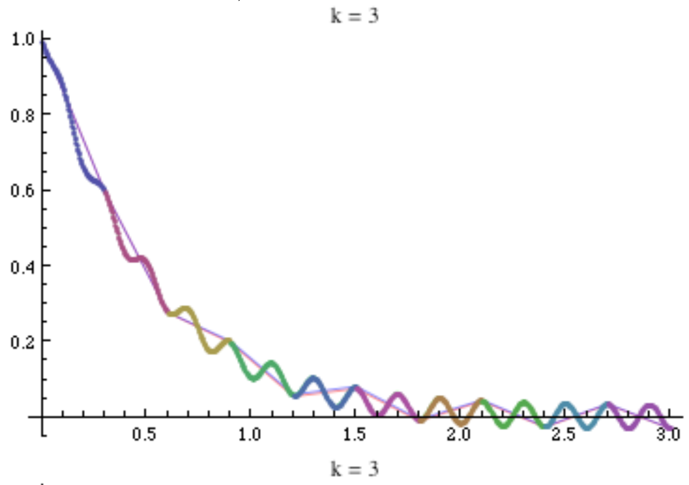
The method is based on the iteration scheme:

$$u_{n+1}^0 = G_{n,\Delta T}(u_n^0)$$

$$u_{n+1}^{k+1} = G_{n,\Delta T}(u_n^{k+1}) + F_{n,\Delta T}(u_n^k) - G_{n,\Delta T}(u_n^k)$$

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





Can parareal be used to accelerate real physics calculations (e.g evolution of fully developed turbulence)? → BETA a pseudo-spectral solver for model DTEM physics

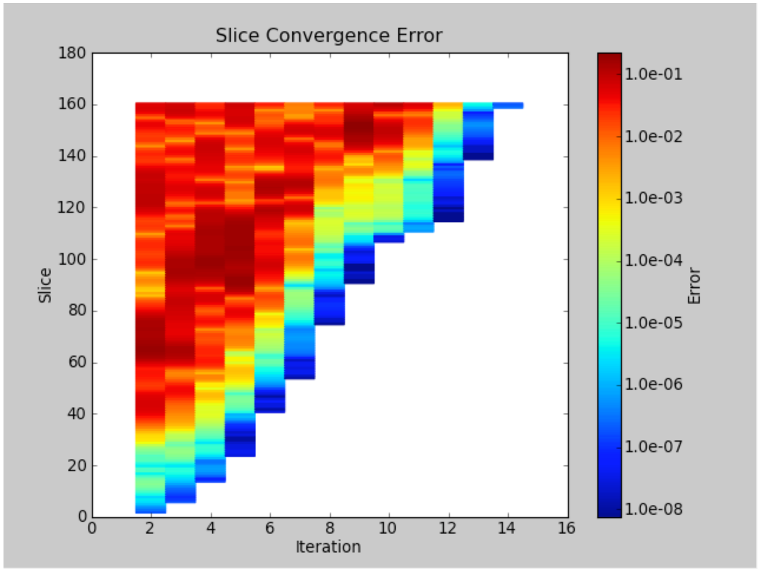
- **Fine solver based on Hasagawa-Mima:**

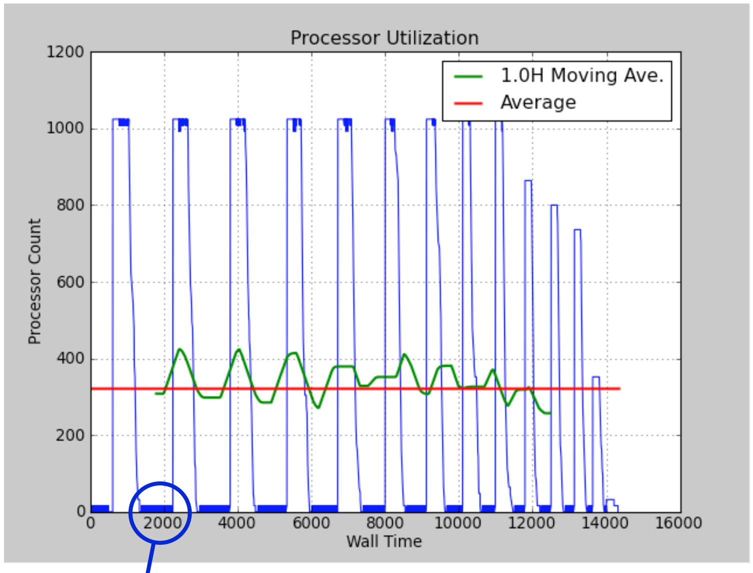
$$\frac{\partial}{\partial t}\left(1-\rho_s^2\nabla_\perp^2\right)\phi + D\frac{\partial^2\phi}{\partial y^2} + \frac{V_D}{2}\frac{\partial\phi}{\partial y} - \frac{4L_n}{\varepsilon^{1/2}}\left[\nabla_\perp\left(\frac{\partial\phi}{\partial y}\right)\times\hat{z}\right]\bullet\nabla_\perp\phi = Sources - Sinks$$

- **For the coarse solver use same equation as fine solver, but:**
  - **Reduce spatial resolution: ~half**
  - **Faster, less precise time integrator: 4th order RK instead of VODPK**
  - **Change dissipation scale**
- **For projection from fine to coarse solution → truncation**
- **For lifting from coarse to fine solution → match spectral slope, use random phase; other wise, keep high order coefficients from previous iteration**
- **For convergence → total mode energy was shown to be a good proxy for convergence of low k modes. Thus only one convergence measure was needed.**
- **Initially implemented entirely in MPI (very complicated) – *Samaddar, Newman, Sanchez, J. Comp Phys 229 (2010) 6558-6573***

The parareal algorithm was re-implemented in the IPS without modification to the IPS → much more straightforward implementation

- **IPS implementation:**
  - Three IPS components (no plasma state) – fine solver, coarse solver, convergence test
  - Task pool manager – efficiently handles parallel executions of fine solver
  - *Traditional* loop control – iteration loop, not time loop
  - *Two levels of parallelism – MPI coarse and fine solver codes, multiple instances of fine solver component*
- **Dividing the simulation time interval into 160 slices, convergence was obtained in 14 iterations for a reduction of simulation time of about ×6**





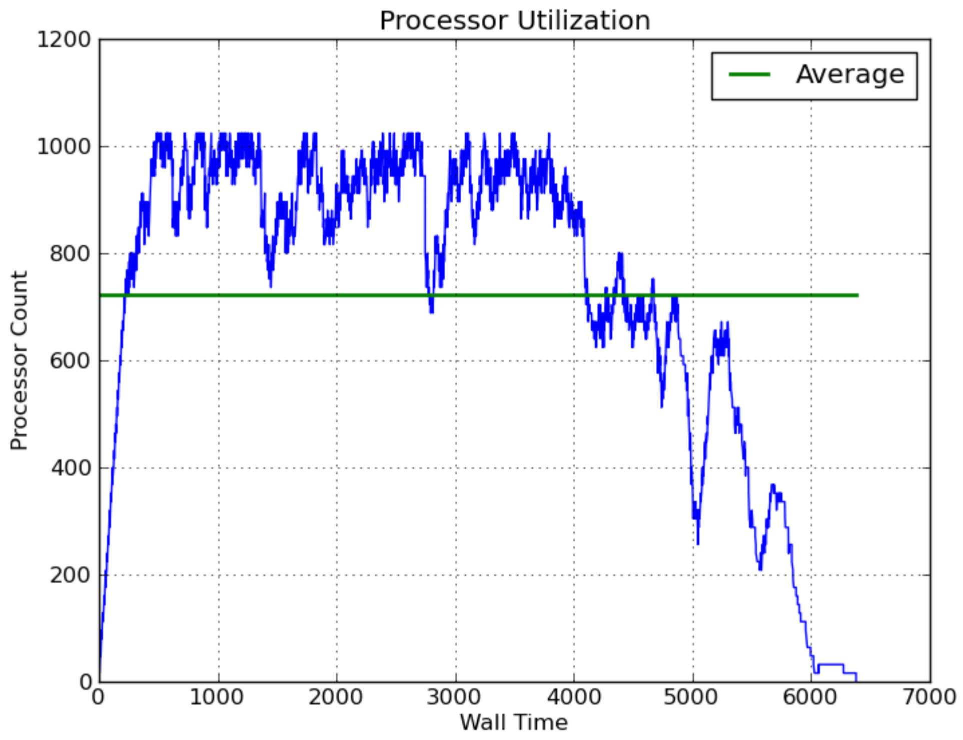
Suffers from inefficiency during long run of coarse solver

Innovative modification of the parareal workflow using IPS results in added factor of 2 improvement of efficiency and run time

- **Obvious observation (but for years nobody observed it) – You don't have to wait for all coarse solves to complete before starting the iteration and the next round of fine solves.** → *You can interleave them*
- *Three levels of parallelism – MPI coarse and fine solver codes, multiple instances of coarse and fine solver components, concurrent execution of coarse solver, fine solver and convergence components*
- **Completely event driven** → *No traditional loop*

Is this the route to turbulence modeling on the transport time scale or extended MHD studies at ITER relevant Lundquist numbers? Might be.

Presently being used with gyrokinetics code GENE


Processor Utilization
Average
Processor Count
Wall Time