Integrated Plasma Simulator (IPS)

TRANSP Users Group Meeting

March 23-24, 2015
PPPL
D. B. Batchelor

- **What the IPS is**
- **Example workflows implemented in IPS**
- **Possible applicability to TRANSP**
- **A miscellany of trenchant comments**

IPS? What's that? → Started under SWIM SCIDAC now part of AToM SCIDAC

The IPS is a framework that allows physics modules communicate and to efficiently interoperate

Plus a collection of physics components comprising many well known and highly tested codes

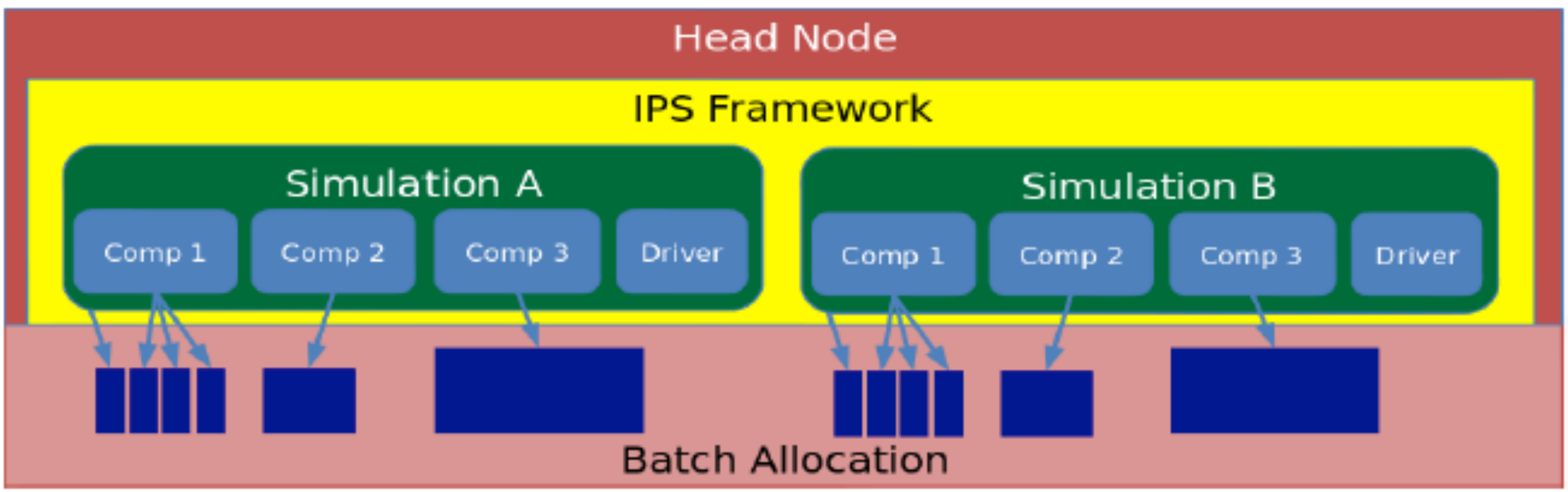
- **1.5 D transport – TSC, FASTRAN, Corsica,** ***PTSOLVER/ISOLVER in progress***
- **Neutral Beam – NUBEAM**
- **Ion cyclotron – TORIC, AORSA, GENRAY**
- **Electron cyclotron – GENRAY**
- **Lower Hybrid – GENRAY**
- **Fokker-Planck – CQL3D**
- **Gyrokinetic – GYRO, GS2,** ***TGLF***
- **Edge – EPED, SOLPS**
- **MHD – NIMROD, PEST-II …**

Design Characteristics

- **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 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 codes – use existing I/O file structures**
  - **Avoid name-space/compiler/library incompatibilities between components**
- **Plasma State:** ***Optional*** **transfer mechanism for time-evolving data that must be transferred between components**

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

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

- **Driver written in programming language (Python) – allows arbitrarily complicated component invocation → *but no python required by user***
- **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
  - 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**

Web Portal Provides Snapshot of Current Status of SWIM Simulations (http://swim.gat.com:5050/)

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 >30,000 SWIM runs stored in the system

OAK RIDGE National Laboratory

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

- 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

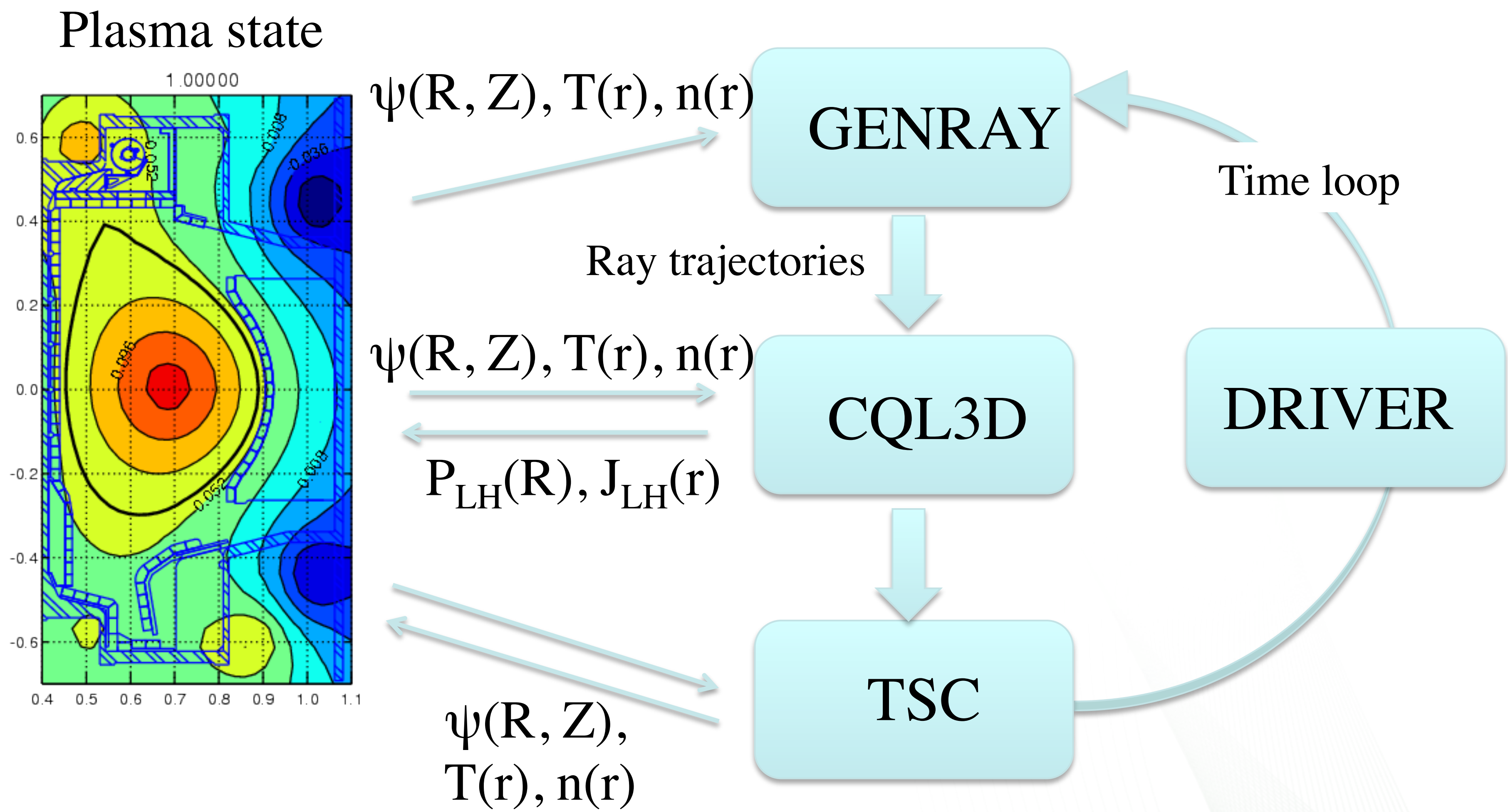
Parareal – hybrid time/iteration loop with logic embedded in components.

- 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

IPS coupled with DAKOTA – optimization searches and parameter sweeps

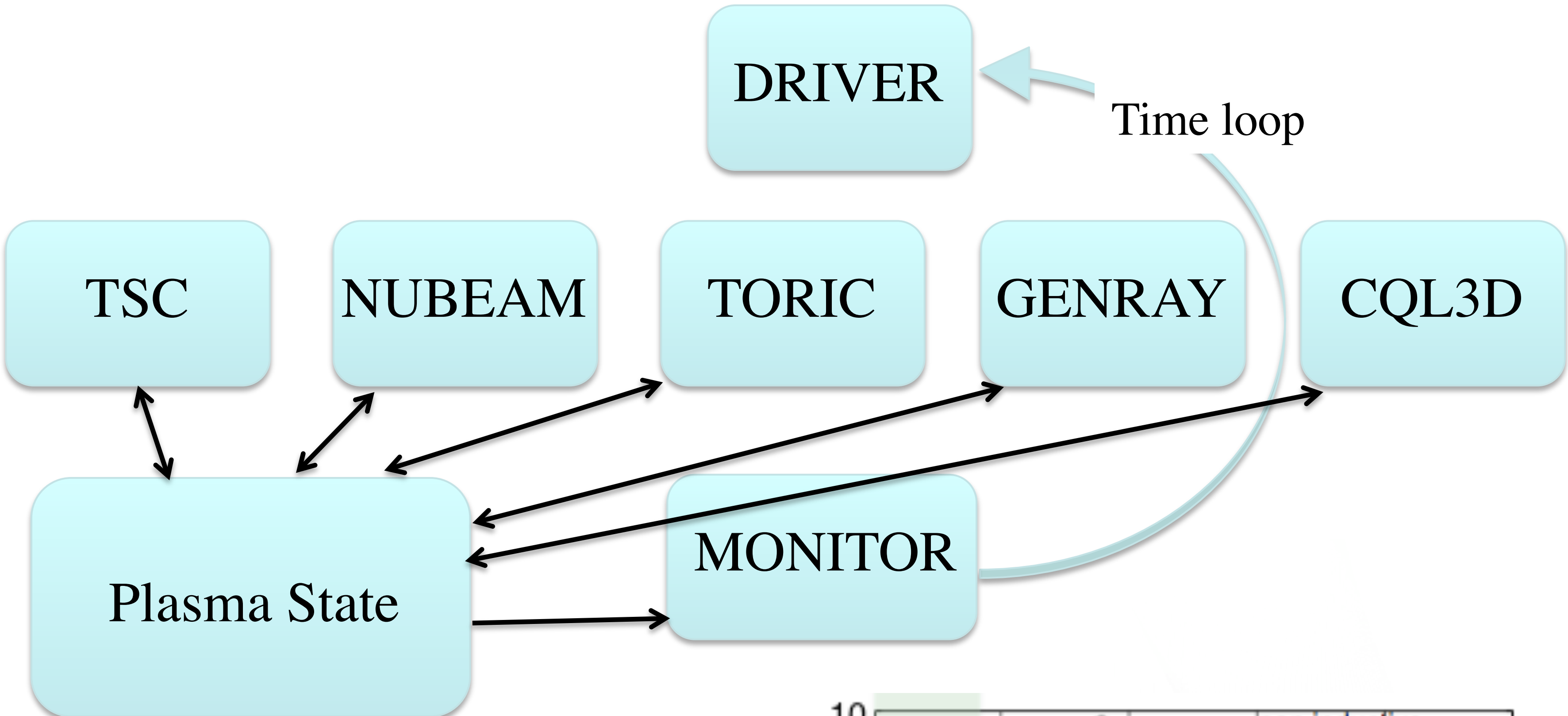
- Manages large numbers of essentially serial jobs

Simple, explicit time loop coupling lower hybrid RF (GENRAY), Fokker-Planck CD (CQL3D), and transport (TSC)

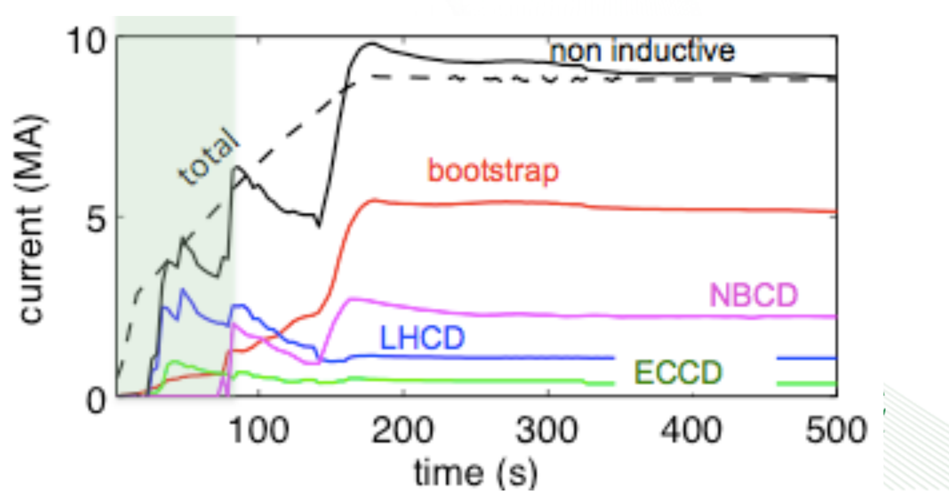


Application: Time-domain simulation of nearly fully non-inductive discharge in C-Mod using LHCD (Bonoli *et al*)

Explicit time stepping with concurrently running components

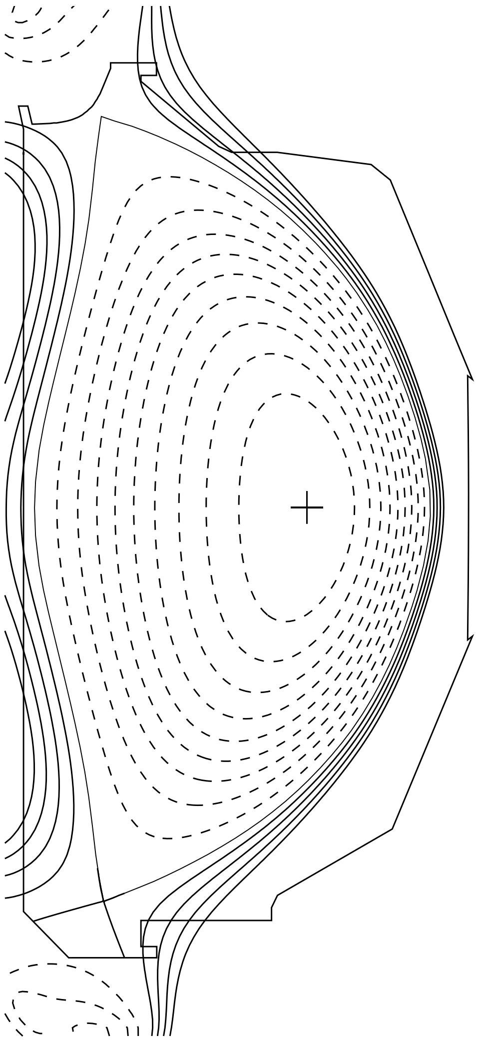


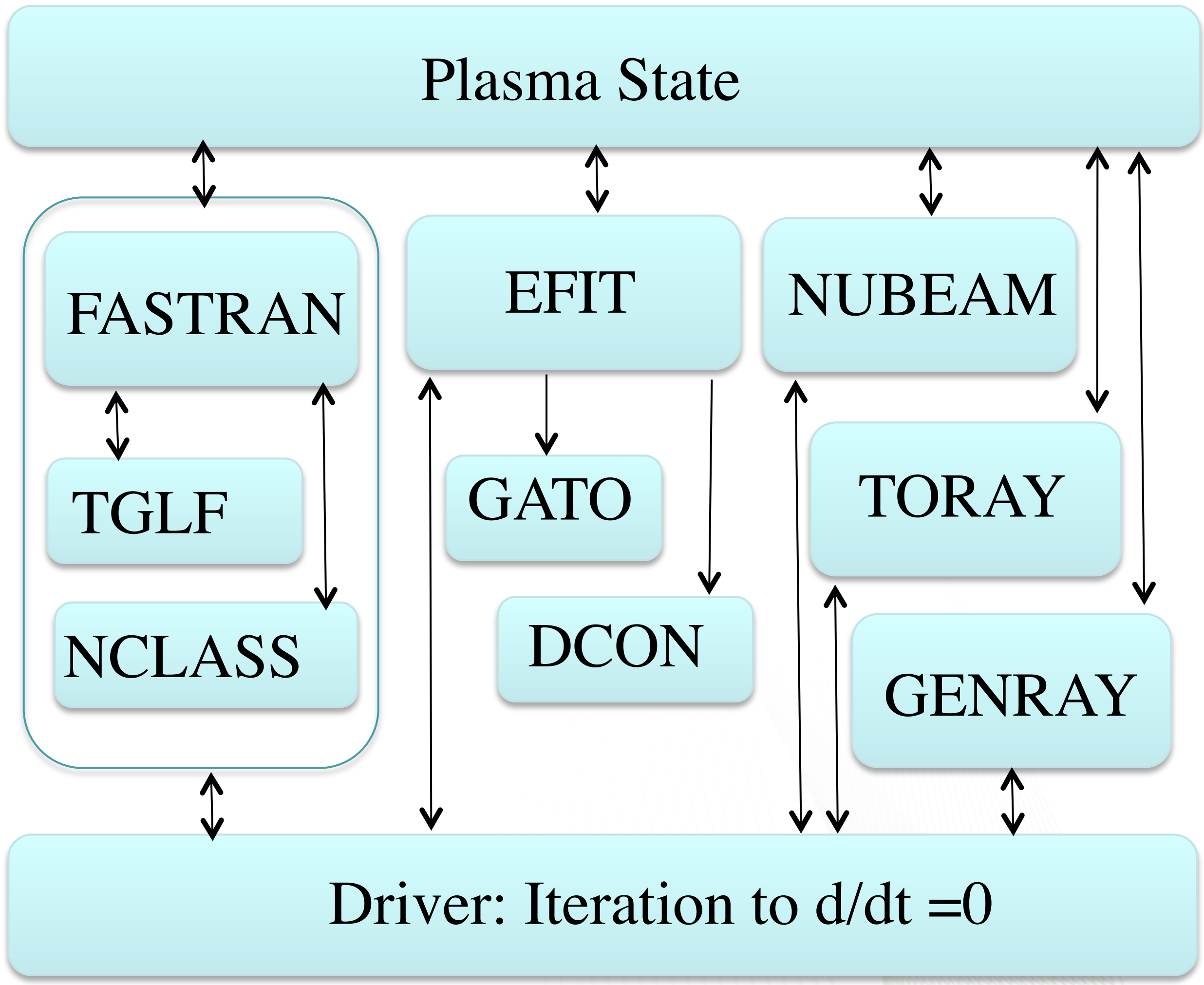
Application: ITER scenario studies (Poli *et al*)



Fixed-point iteration: FASTRAN with TGLF transport and source components iterating to self-consistency

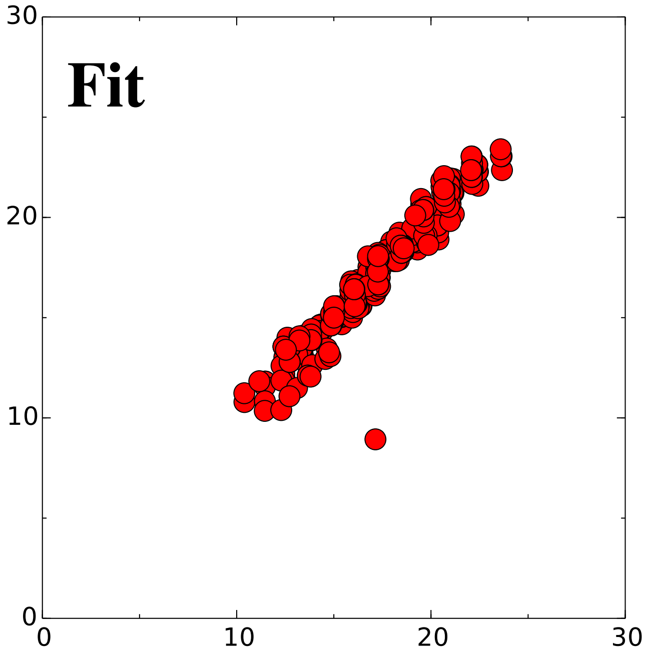
Self-consistent DIII-D equilibrium at $\beta_N = 5$



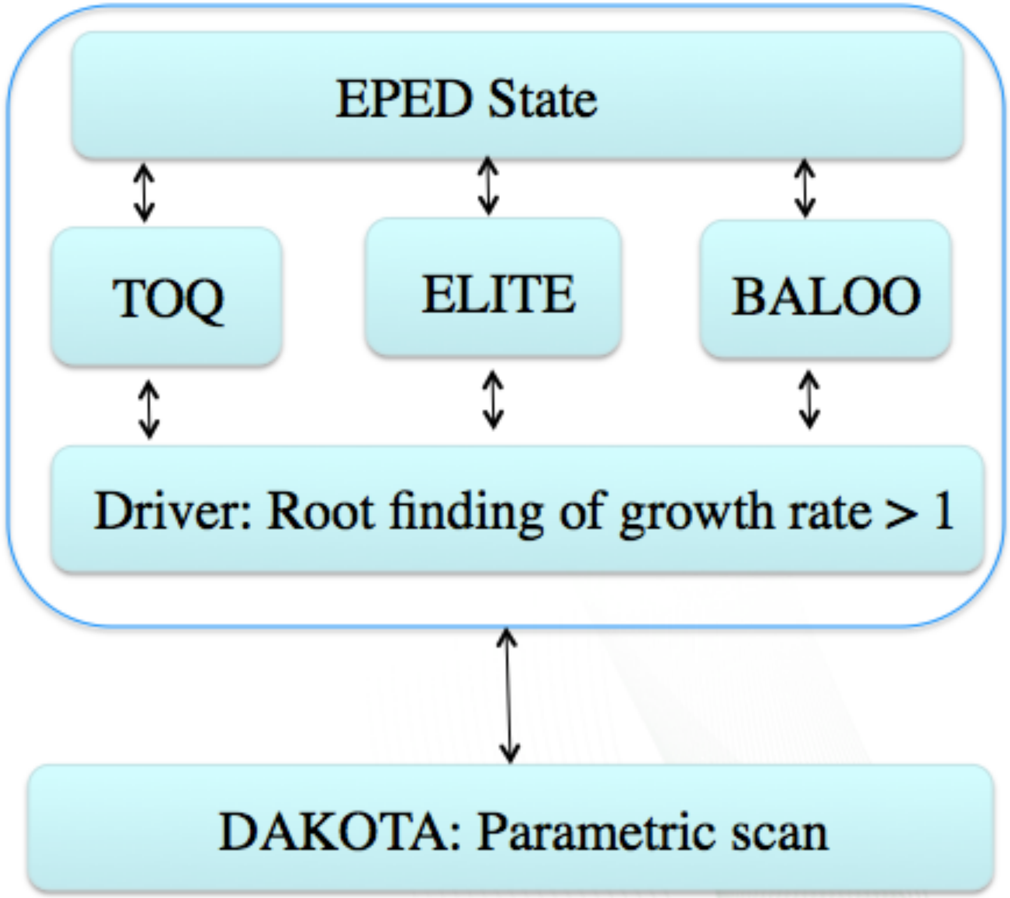


DAKOTA-Enabled IPS – EPED scans and GYRO scans

Regression for 500 EPED runs for DIII-D SS scenario

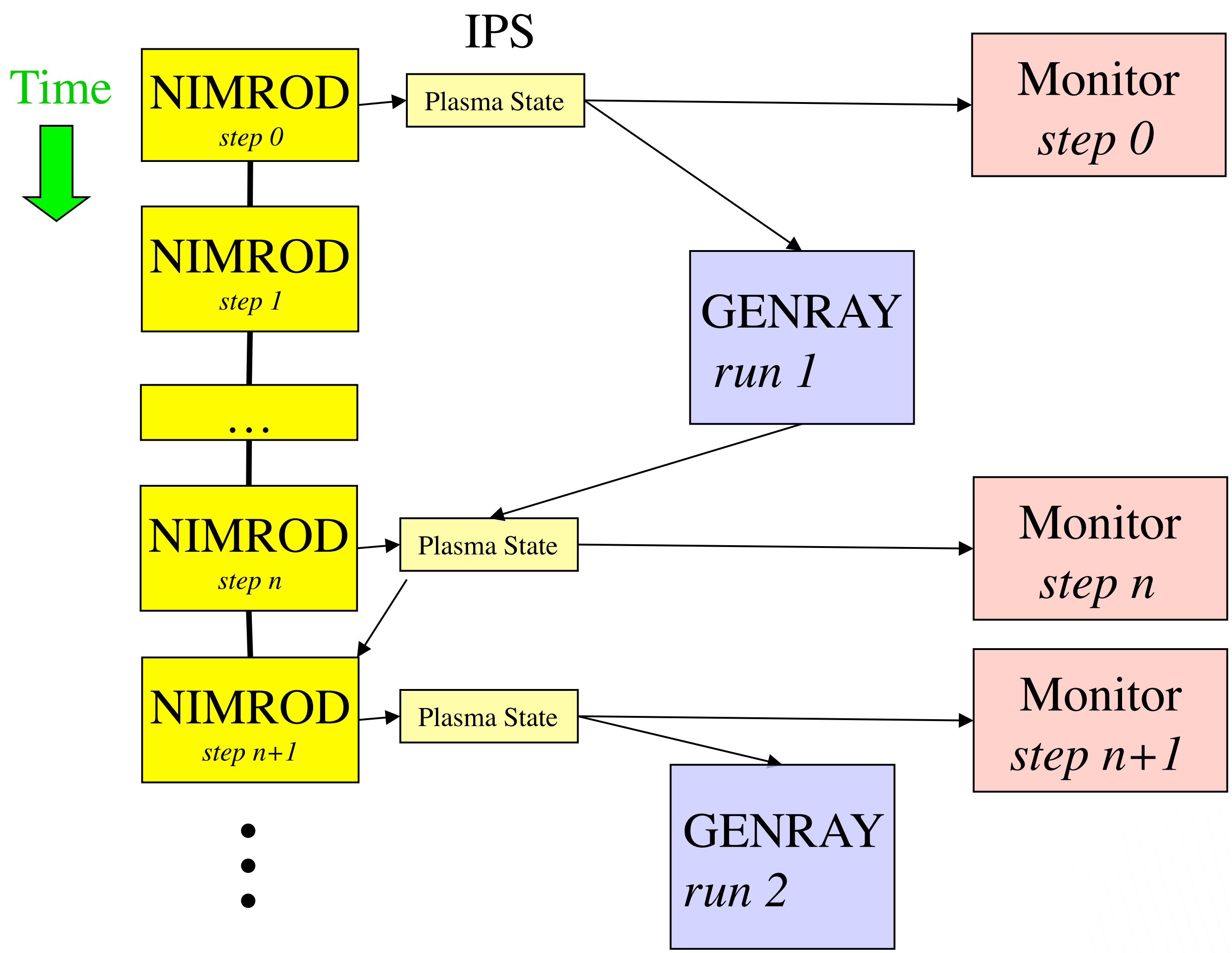


P_{ped} (kP), EPED



Application:
Parametric fit to EPED database (Park *et al*)
Recalibration of TGLF – GYRO (Candy *et al*)

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

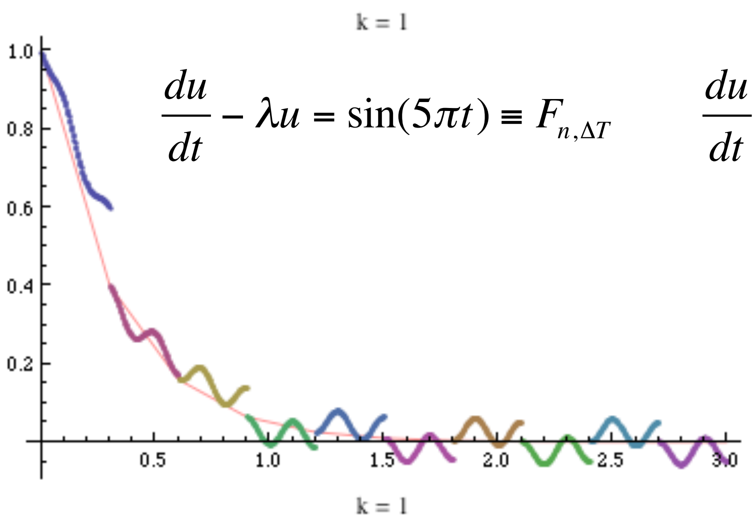


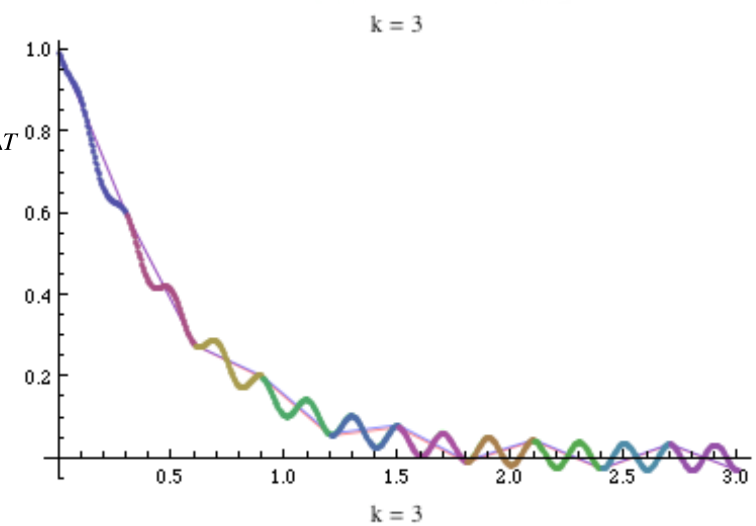
- **NIMROD exports magnetic geometry and n,T profiles to Plasma State**
- **GENRAY then calculates RF propagation and power deposition; exporting these quantities to the Plasma State**
- **NIMROD converts GENRAY data into momentum and energy source terms.**
- **Ultimately will include kinetic closure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

Completely event driven workflow – Parallelization over time domain using parareal algorithm

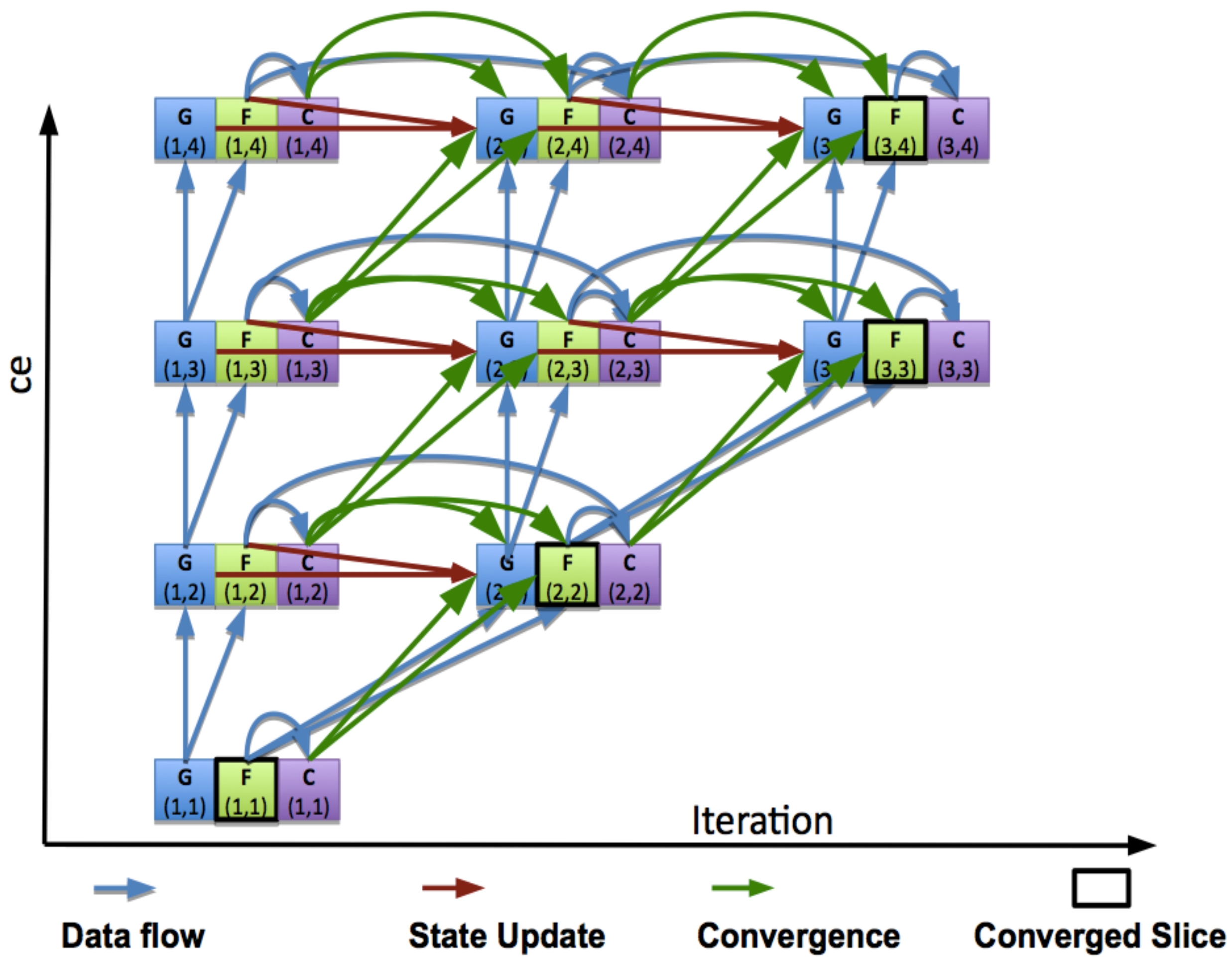
- **Requires :**
  - **Accurate solver, *F(x, t)*, you want to solve from t_{init} to t_{final} → *fine solver***
  - **Fast, less accurate solver, *G(x, t)* → *coarse solver***
  - **Convergence criterion for solution x**
- **Break time domain into a large number of chunks, NΔT**
- **Run coarse solver get approximate solution**
- **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 interations, but maybe much faster**

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





Logic contained in components (not driver) – event driven based on availibility of data



Applications:

- **DTEM turbulence – BETA**
- **1.5D transport – Corsica**
- **Edge – SOLPS**
- **Gyrokinetics – GS2**

Samaddar *et al*

Good when time is expensive and processors are cheap

What is IPS good for?

- **Implementing complex workflows – on any scale computer**
- **Coupling massively parallel codes, or codes with varying degrees of parallelism**
  - **Can greatly increase throughput using multi-level parallelism**
  - **Particularly useful if we already have the physics components you need**
- **Managing large numbers of related serial or parallel jobs**
  - **Parameter sweeps**
  - **Optimizations**
- **We have in hand a number of useful workflows you can take *gratis***

SWIM aimed to make it easy (at least possible) to add new components, interchange components, generate workflows, use compute resources efficiently

OAK RIDGE National Laboratory

How do you use IPS? What do you get?

- **User provides a configuration file to specify simulation elements + input files for the component codes**
- **Submit from the command line (e.g. qsub) – *or now through OMFIT***
- **Output is a "results" directory tree consisting of selected physics code output files + Plasma State files (if using)**
- **Also get monitor component – aggregation of selected state data into time-series**
- **Web-based monitoring of job progress. Ability to view monitored data as job runs**

AToM project aims to make it easier for users to set up and analyze IPS runs through coupling to OMFIT

Initial thoughts on application to TRANSP

- **We think that it should not be difficult for TRANSP and IPS to interact – they both speak Plasma State**
- **TRANSP could actually be incorporated as an IPS component**
  - **Could make massively parallel computing more accessible to TRANSP users**
  - **User-efficient/resource-efficient way to do multiple related TRANSP runs or very long runs**
- **TRANSP could spin off IPS sub-workflows**
  - **e.g. have IPS run all RF/Neutral Beam components concurrently**
  - **Implement complicated workflows not already contained in TRANSP**
- **Interpretive mode**
  - **TRANSP generates time sequence of experimental plasma profiles → Plasma States**
  - **IPS runs all source modules concurrently on all time slices, returns plasma states with heating/CD profiles**
- **If it is decided to refactor TRANSP into a more modular structure, IPS might be considered for the framework**

Other thoughts and questions

- **Plasma State**
  - We consider it to be a valuable resource
  - What is PPPL plan on future development/support?
- **Interacton with ITER**
  - We need experience with IMAS and ITER data structures
  - PPPL do Plasma State to IMAS translation
  - Plan to install IMAS at PPPL – schedule and details?
  - We will want to keep US-based simulation capability for our own purposes
- **AToM now working on nested IPS simulations**
- **There are significant advantages to discrete code architecture vs single executable**