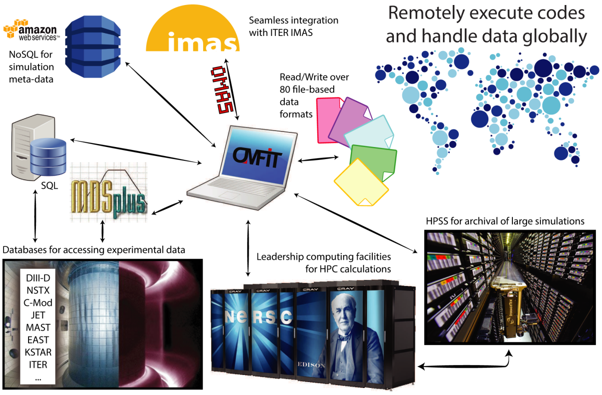
Computational Frameworks in Fusion and Plasma Physics

Sterling Smith[1]

[1] General Atomics

2020 PPPL Introduction to Fusion Energy and Plasma Physics Course
June 24, 2020



GENERAL ATOMICS

Outline

Personal background

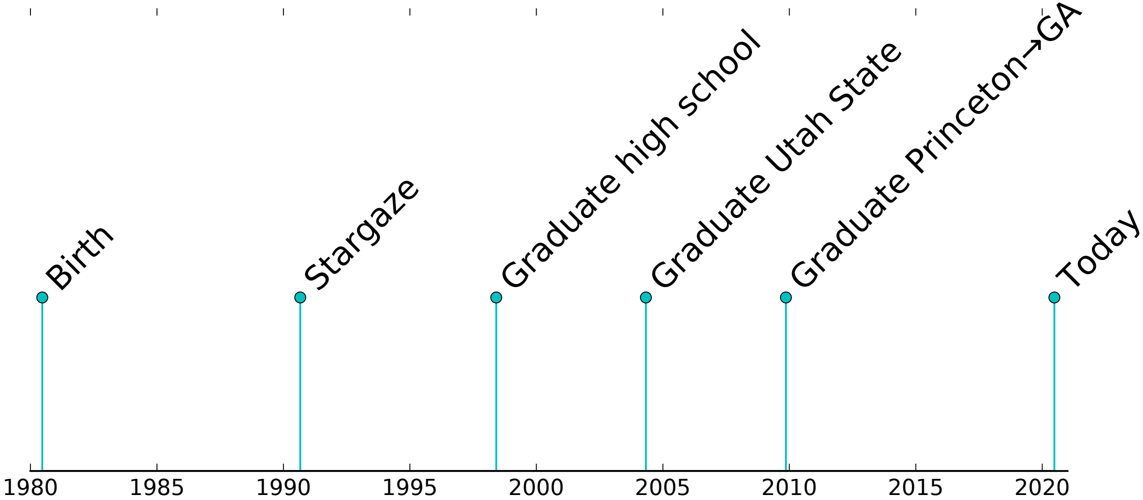
Do not reinvent the wheel

Give back to the community

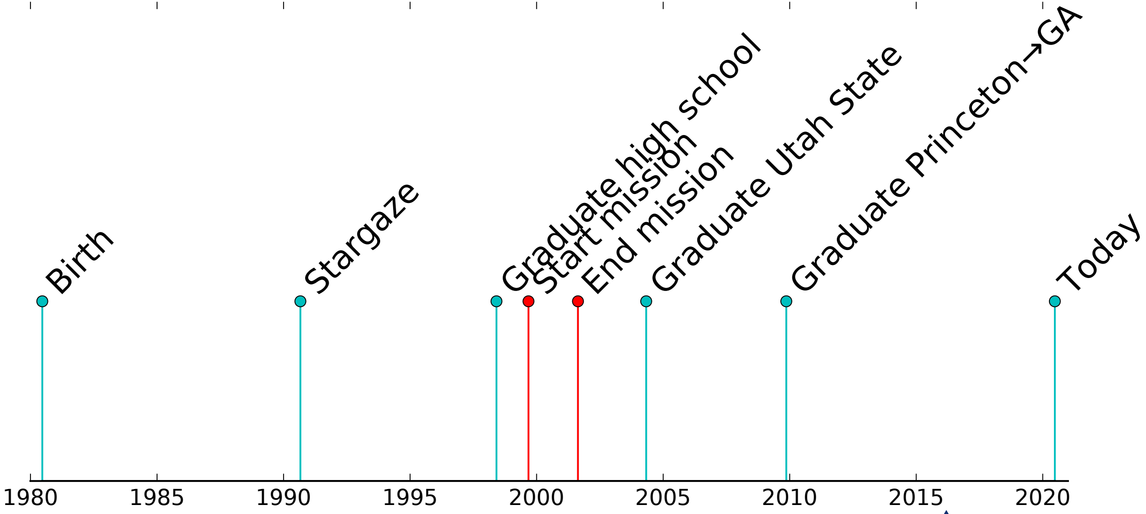
What is OMFIT?

OMFIT solves common computer usage issues

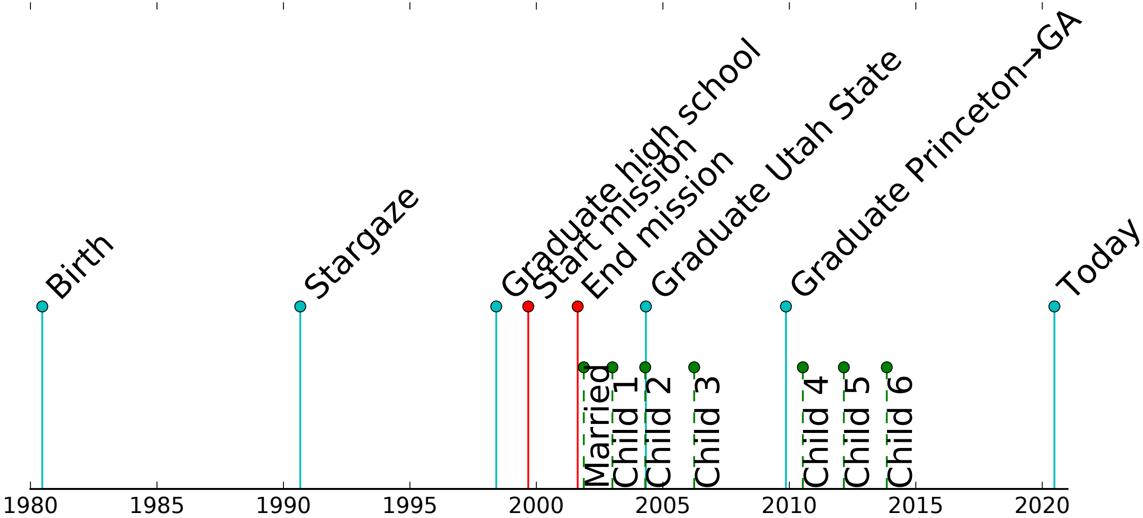
My mother took me stargazing



In Dominican Republic, I was inspired to find a more steady source of electricity



Life happens along the way



GENERAL ATOMICS

The "best" computer language depends on the application

- I started dabbling with `qbasic`[1] in high school (late 1990's)
- I took a `C++`[2] class as an undergraduate
- The Utah State physics program used `MathCAD`[3] and `Maple`[4] for undergraduate instruction
- In graduate school, I first learned about `unix`[5]/`linux`[6]
- My 1st year project required `IDL`[7] for data processing
- My 2nd year project introduced me to `python`[8]
- The graduate laboratory class exposed me to `MatLab`[9]
- I wrote my thesis code in `Fortran90`[10], with `python` pre- and post- processing
- At DIII-D, I took over and improved some analysis routines written in `IDL` (`ZIPFIT`)
- I later took over maintenance of the `ONETWO` code - a half million lines of `Fortran77`
- Along the way, I worked to implement the coordination of code running (integrated modeling) via `python` (`IMFIT`, then `OMFIT`[11])

GENERAL ATOMICS

Outline

Personal background

Do not reinvent the wheel

Give back to the community

What is OMFIT?

OMFIT solves common computer usage issues

Computers have been around for a while now

Initial code recipes have evolved into entire libraries and frameworks

- *Numerical Recipes* books [12]
- BLAS [13] and LAPACK [14] - Linear Algebra libraries
- BOUT [15] - A framework for solving an arbitrary set of equations including parallelization
- hDF5 [16]/netCDF4 [17] - A library for storing self-describing data
- MDS+ [18] - A library for storing data with a client/server interface (most US tokamaks use this)
- OMFIT [11] - One Modeling Framework for Integrated Tasks - More later
- conda [19] - Cross-platform package and dependency management
- IMAS [20] - ITER Integrated Modeling and Analysis Suite

This list is not exhaustive, but rather representative of the types of libraries that are available.

Do your homework and ask around

If you are going to write some code that is more than a few lines long:

- Do a google search
- Do a search on github.com
- Ask your advisor:
  - Have you done something already?
  - Do you know someone that may have done something similar?
  - How would you do it?
- Ask the community the same questions. Here are some communities:
  - NTCC[21] - Started down path of code collection for MFE
  - The AToM project[22] is knowledgeable of existing MFE codes
  - The Plasmapy project[23] represents the most modern and open collection of efforts in plasma research
- Use existing input/output standards
  - netcdf[17], hdf5[16], yaml[24], IMAS DD[20], OpenPMD[25], namelist[26]

Outline

Personal background

Do not reinvent the wheel

Give back to the community

What is OMFIT?

OMFIT solves common computer usage issues

Not reinventing the wheel is not enough; feedback is invaluable

- If the documentation is lacking, report it
- If you need help, ask for it
- If you find a bug, report it
- If you figure out how to fix the bug, report it

Use the `http://www.sscce.org` pattern when asking for help

Short, Self Contained, Correct (Compilable), Example[27]

Short Cut out all of the extra code that is not causing a problem

- Sometimes this step by itself is sufficient to help you see the solution yourself

Self contained Include any non-public libraries, files, data, or metadata that are necessary to produce the problem

- If on a cluster be sure file permissions allow access

Correct The code provided should work/compile

Example The code should actually demonstrate the problem

When you start writing code, use version control

`cvs`[28] was one of the initial version control systems

`svn`[29] came along as a replacement for `cvs` with some improvements for ease of use

`git`[30] is emerging as the program of choice to track code versions

- All clones of a `git` repository contain the whole repository
- A central canonical version may be hosted at GitHub(.com) or a local GitLab server
- Branching is easy and cheap
- GitHub provides code review and continuous integration testing tools

Random thoughts

- Use a package manager - conda[19] is my manager of choice
  - The more portable a code is, the more users it will gain
  - The more users it gains, the more contributors it will garner
  - The more contributors it has, the more stable and feature-rich the code will become
- There are no bugs, just unwritten regression tests (see previous talk by Dr. Murphy)
- Get a coding buddy
- Look over someone's shoulder as they work at a computer - what shortcuts do they have for working?

Homework: Interact with an open source project on GitHub.com

1. Pick a repo
   - `https://github.com/MFEFormulary/MFEFormulary`
   - `https://github.com/PlasmaPy/plasmapy`
   - `https://github.com/gafusion/OMFIT-source/` [a]
   - `https://github.com/smithsp/cpsfr`
2. Open an issue for a feature request or bug
3. Clone the repo
4. Make some changes on a branch and push it back to GitHub
5. Open a Pull Request
6. Read the Cathedral and the Bazaar[31]

[a] Send me your GitHub username for access

GENERAL ATOMICS

Outline

Personal background

Do not reinvent the wheel

Give back to the community

What is OMFIT?

OMFIT solves common computer usage issues

OMFIT is a versatile framework designed to facilitate experimental data analysis and enable integrated simulations

OMFIT was created to fill a gap in the fusion community

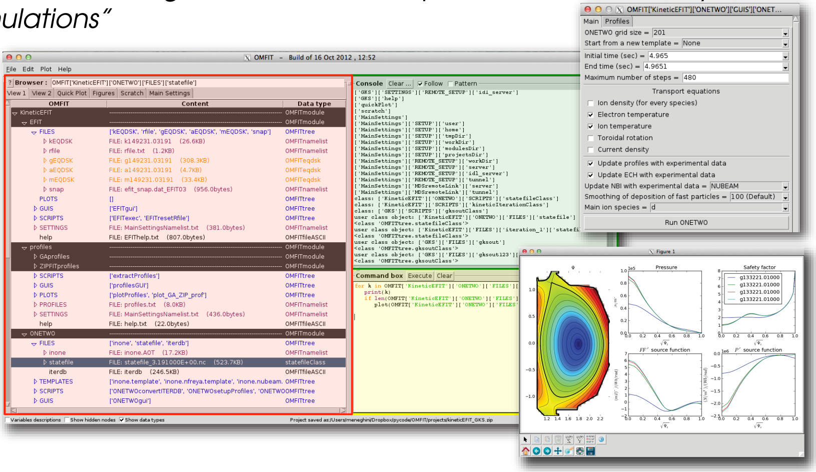
- A unified tool for
  - Accessing experimental data
  - Running existing codes
  - Parsing code inputs and outputs
  - Streaming together outputs of one workflow as the inputs of another workflow
  - Collecting community scripts
  - Creating robust GUIs
  - Visualizing results
- A tool that is
  - Portable
  - Based on a modern language
  - Public
  - Documented
  - Tested

OMFIT by the numbers since July 2015

- Cumulative graphical sessions: **120,926**
- Cumulative commits: **33048**
- Cumulative users: **843**

– One Modeling Framework for Integrated Tasks

"A versatile framework designed to facilitate experimental data analysis and enable integrated simulations"



https://omfit.io

O. Meneghini, S. Smith, et al. Nuclear Fusion, **55** 083008 (2015)

OMFIT – One Modeling Framework for Integrated Tasks

1. **Integrated Modeling Framework**
   - Enables data exchange among different components and coordinates their execution in complex workflows

2. **Lightweight & pure Python**
   - Remote execution of interactive/batch jobs
   - Installs & runs anywhere: public/private, cluster/laptop

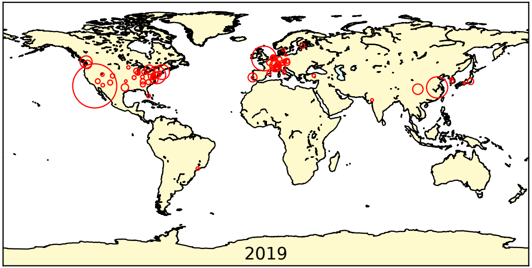
3. **Free-form tree data structure**
   - Support for most fusion-relevant data formats
   - Does not exclude use of standard data structures, such as IMAS (ITER Integrated Modeling and Analysis Suite)

4. **Interactive and graphical or scripted**
   - Accelerate most time consuming tasks: develop, setup, visualize, share

5. **Version control and community**
   - Grow at scale cheaply and remain focused

OMFIT is installed on many fusion public clusters and accesses data from many tokamaks around the world

- ITER
- GA/DIII-D
- NERSC
- MIT
- PPPL/NSTX
- EPFL/TCV
- CCFE/JET/MAST
- ITM-Gateway
- IGI-CNR/RFX
- IPP-CA/COMPASS
- MPI-IPP/ASDEX-U
- WEST
- ASIPP/EAST
- SWIP/HL2A/HL2M
- NFRI/KSTAR
- CFETR



OMFIT is actively used for current research
At least 54 presentations in the last APS DPP Conference

5 invited, 14 orals, 1 miniconference, 34 posters

Kriete	BI2.3	Crocker	BP10.44	Mcclenaghan	PO5.15
Austin	BP10.3	Xing	BP10.102	Montes	PP10.67
Weisberg	BP10.4	Sciortino	CO5.11	Holland	PP10.107
Haskey	BP10.7	Ashourvan	CO7.4	Xie	PP10.110
Zhao	BP10.8	Pankin	CO7.10	Neiser	PP10.125
Morton	BP10.9	Houshmandyar	GP10.61	Orlov	PP10.132
Smith	BP10.10	Knolker	JI2.1	Brookman	TI2.6
Mckee	BP10.11	Grierson	JO5.2	Samuell	TP10.112
Truong	BP10.12	Chen	JO5.5	Yu	TP10.117
G. Wang	BP10.16	Hu	JO5.6	Hong	TP10.118
Victor	BP10.23	Jian	JO5.11	Nelson	TP10.119
Park	BP10.24	H. Wang	JO5.15	Zhang	TP10.128
Ding	BP10.25	Munaretto	JP10.23	Abbate	TP10.136
Park	BP10.30	Smith	PM8.6	Morosohk	TP10.137
Yan	BP10.35	Logan	PO5.3	Laggner	UO8.3
Collins	BP10.37	Ernst	PO5.5	Thome	VI3.4
Marini	BP10.39	Rea	PO5.7	Casali	YI3.1
Tang	BP10.43	Bykov	PO5.11	Meneghini	YP10.38

Outline

Personal background

Do not reinvent the wheel

Give back to the community

What is OMFIT?

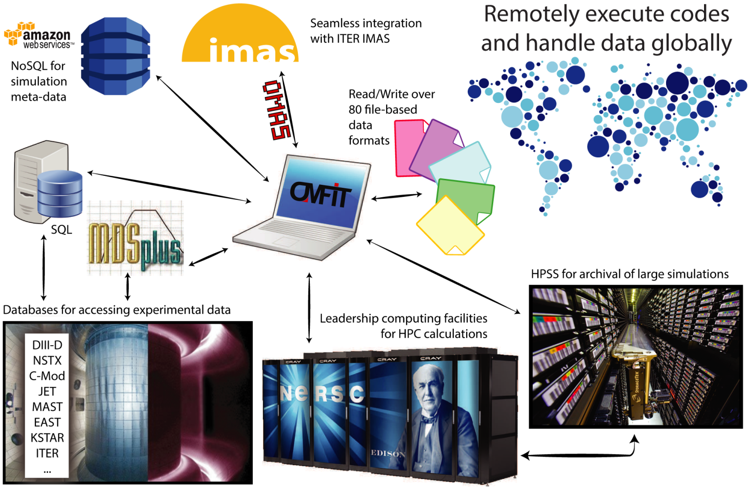
OMFIT solves common computer usage issues

OMFIT solves common computing issues

Issue	OMFIT solution
Get experimental setup	Access experimental data and facilitate interpretive analysis

GENERAL ATOMICS

OMFIT provides access to experimental data and facilitates interpretive analysis

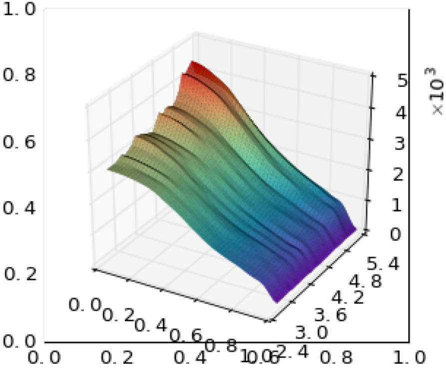


OMFIT can interface with

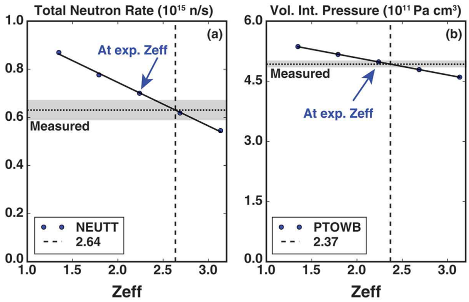
- MDS+
- SQL
- UDA
- s3
- IMAS (ITER)

OMFIT provides access to experimental data and facilitates interpretive analysis

TRANSP is a free-boundary equilibrium and transport solver, used for analysis of tokamak plasma discharges and for experimental planning



Experimental electron temperature profile vs time



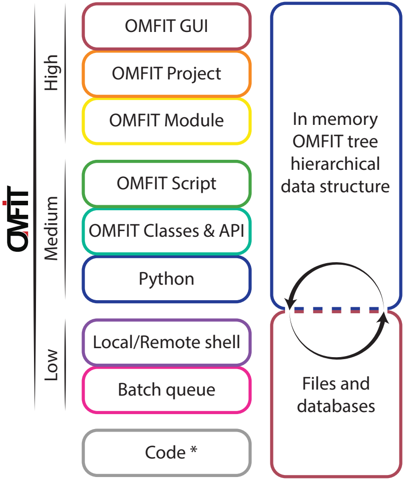
Comparison of predicted neutron rate and total pressure provides constraint for Z_{eff}

B. A. Grierson, et al., Fus. Sci. and Tech., **74**:101; https://transp.pppl.gov

OMFIT solves common computing issues

Issue	**OMFIT solution**
Get experimental setup	Access experimental data and facilitate interpretive analysis
Arcane input format files	**Python classes for parsing and writing broad range of files**

OMFIT parses and writes a variety of input and output files



- Fortran namelist
- NetCDF 3/4
- HDF 5
- g,k,a,m eqdsk
- SOLPS
- IPS config
- TGLF
- (c)GYRO
- TGYRO
- pfile
- ONETWO
- TRANSP
- BOUT++
- M3D-C1
- NIMROD
- GENRAY
- TORAY
- FASTRAN
- IDL savefile
- IONORB
- GATO
- GPEC
- JSON
- MARS

OMFIT parses and writes a variety of input and output files

SOLPS is a fluid code that attempts to solve for the densities and temperatures in the scrape off layer (SOL) given boundary conditions

Example input file:

```
*    7 : Recycling - PFR side      3
FFFF
   2000 10001       6       1
 1.00000E+00 0.00000E+00       0       0       0       0       0       0       0       0
FFFT
     -1
FFTFF
      1
      7       1       1       1       1      53      64       0       0
 1.00000E+00 1.40000E+02 7.00000E+00 0.00000E+00 2.11100E+03
     -1       0       0       0       0       0       0
 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00 0.00000E+00
 1.00000E+00 9.00000E+01 0.00000E+00 0.00000E+00 0.00000E+00
*    8 : Recycling - PFR side      4
FFFF
   2000 10001       6       1
 1.00000E+00 0.00000E+00       0       0       0       0       0       0       0       0
FFFT
     -1
FFTFF
      1
      8       1       1       1       1      90     102       0       0
```

OMFIT	Content
▷ SOLPS	----------
▽ input.dat	FILE: input.dat (34.9kB)
EIRENE input file generated with uinp v.	-{ (1) }-
▷ 1. Data for operating mode	-{ 2 (1) }-
▷ 2. Data for standard mesh	-{ 17 (1) }-
▷ 3a. Data for non default standard surfaces	-{ 29 (1) }-
▷ 3b. Data for additional surfaces	-{ 46 (1) }-
▷ 4. Data for species and atomic physics m	-{ 4 (1) }-
▷ 5. Data for plasma background	-{ 12 (1) }-
▷ 6a. General data for reflection model	-{ 23 (1) }-
▽ 7. Data for primary sources, nstrai strata	-{ 14 (1) }-
NSTRAI	11
▷ INDSRC(ISTRA)	[0, 0, 0, 0, 0, 0, 0, 0, 0, ...]
ALLOC	0.5
▷ 1 Recycling-Target 1	-{ 13 (1) }-
▷ 2 Recycling-Target 2	-{ 13 (1) }-
▷ 3 Recycling-Target 3	-{ 13 (1) }-
▷ 4 Recycling-Target 4	-{ 13 (1) }-
▷ 5 Recycling-PFR side 1	-{ 13 (1) }-
▷ 6 Recycling-PFR side 2	-{ 13 (1) }-
▽ 7 Recycling-PFR side 3	-{ 13 (1) }-
▽ 1	-{ 4 (1) }-
NLAVRP	False
NLAVRT	False
NLSYMP	False
NLSYMT	False
▽ 2	-{ 4 (1) }-
NPTS	2000
NINITL	10001
NEMODS	6
NAMODS	1
▽ 3	-{ 10 (1) }-
FLUX	1
SCALV	0
IVLSF	0
ISCLS	0
ISCLT	0
ISCL1	0
ISCL2	0

OMFIT in-memory parsing provides access to variables in a systematic way

In-memory variables can be edited, which propagates back to the file on disk

. S. Kukushkin, et al. J. Nucl. Mater., **86**:2865

OMFIT solves common computing issues

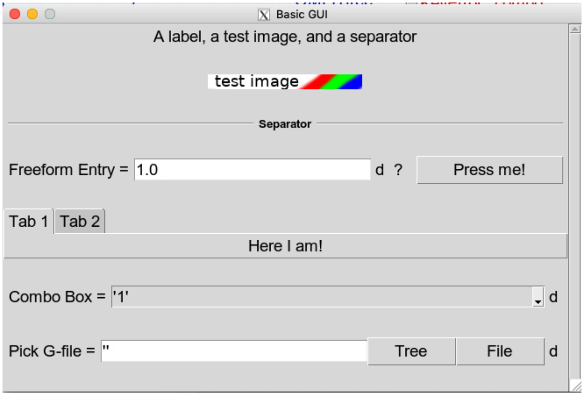
Issue	OMFIT solution
Get experimental setup	Access experimental data and facilitate interpretive analysis
Arcane input format files	Python classes for parsing and writing broad range of files
Technical guide of code is too deep for average user	**Provide customizable GUIs and tutorials**

OMFIT provides easily customizable graphical user interfaces (GUIs) to expose most common settings and workflows in plain English

basic.py - /tmp/smithsp/OMFIT/OMFIT_2019-10-14_14_02_02_588792/project/objects/f

File Edit Format Run Options Window Help

```
OMFITx.TitleGUI('Basic GUI')

OMFITx.Label('A label, a test image, and a separator')

OMFITx.Label('',image=OMFITsrc+'/../samples/test_image.png')

OMFITx.Separator('Separator')

with OMFITx.same_row():
    OMFITx.Entry("OMFIT['entry']", lbl='Freeform Entry',
                 default=1.0, help='A live text entry')
    OMFITx.Button("Press me!", lambda: None)

with OMFITx.same_tab('Tab 1'):
    OMFITx.Label('Here I am!')
with OMFITx.same_tab('Tab 2'):
    OMFITx.Label('Not here')

OMFITx.ComboBox("OMFIT['combo']", {'1': 'This is option 1'},
                lbl='Combo Box', default='1')

OMFITx.ObjectPicker("root['geqdsk']", lbl="G-file",
                    objectType=OMFITgeqdsk)
```



GENERAL ATOMICS

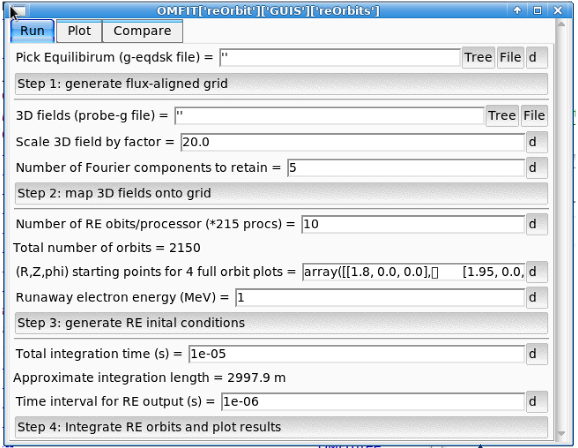
OMFIT provides easily customizable graphical user interfaces (GUIs) to expose most common settings and workflows in plain English

NIMROD extended MHD code used to follow acceleration and deconfinement of runaway electrons

OMFIT NIMROD module:

- Generates initial conditions
- Generates restart files in hdf5 format for non-standard NIMROD particle orbit routines
- Visualizes the results

72 hours on 12 nodes on cori: ~ 30k CPU-hours



V.A. Izzo and P.B. Parks, Phys. Plasmas, **24**:060705
C.R. Sovinec, et al., J. Comp. Phys. **195**:355

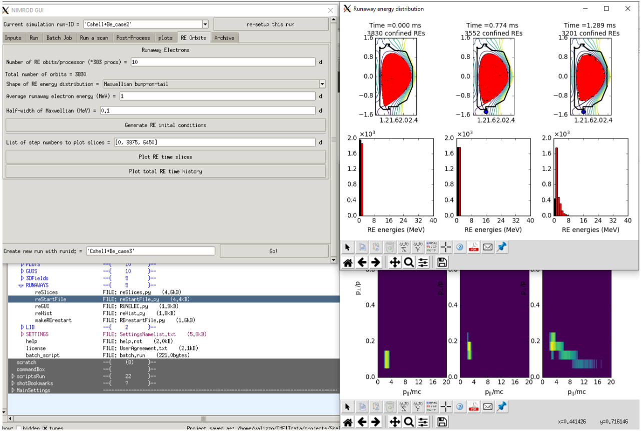
OMFIT provides easily customizable graphical user interfaces (GUIs) to expose most common settings and workflows in plain English

NIMROD extended MHD code used to follow acceleration and deconfinement of runaway electrons

OMFIT NIMROD module:

- Generates initial conditions
- Generates restart files in hdf5 format for non-standard NIMROD particle orbit routines
- Visualizes the results

72 hours on 12 nodes on cori: ~ 30k CPU-hours

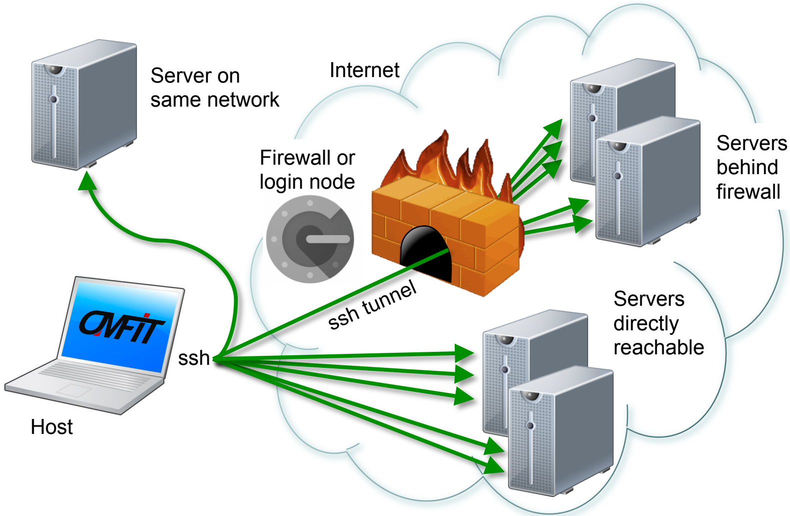


V.A. Izzo and P.B. Parks, Phys. Plasmas, **24**:060705
C.R. Sovinec, et al., J. Comp. Phys. **195**:355

OMFIT solves common computing issues

Issue	OMFIT solution
Get experimental setup	Access experimental data and facilitate interpretive analysis
Arcane input format files	Python classes for parsing and writing broad range of files
Technical guide of code is too deep for average user	Provide customizable GUIs and tutorials
Code only runs at a few locations	**Package OMFIT to be portable, and allow remote execution of arbitrary scripts, even with tunnels or MFA**

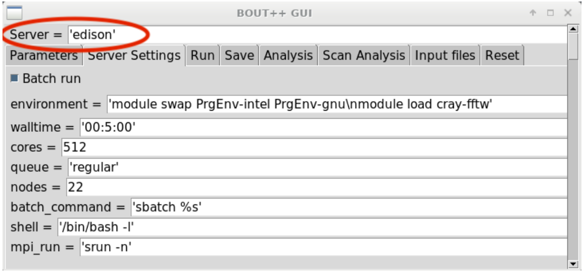
OMFIT can reach remote servers despite tunnels or multi-factor authentication

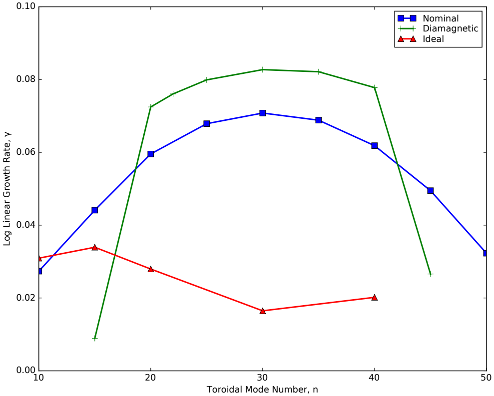


If appropriate, OMFIT runs a python library with the same functionality as One Time Password applications such as Google Authenticator

OMFIT can reach remote servers despite tunnels or multi-factor authentication

BOUT++ is a framework for solving general differential equations. Our specific use case is Edge Localized Mode stability.





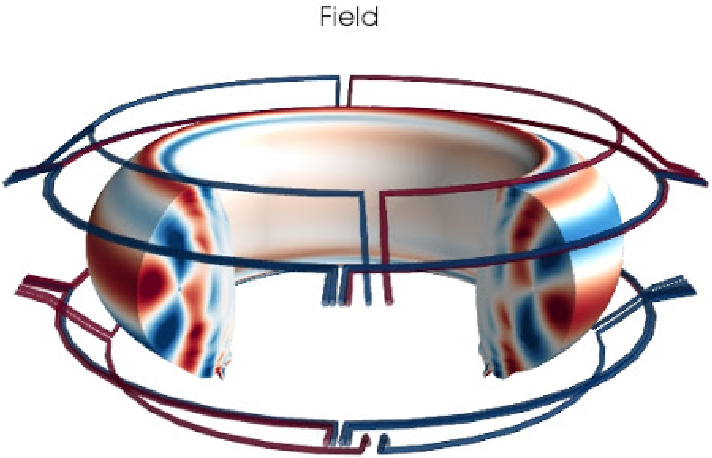
B.D. Dudson, et al., Comp. Phys. Comm. **180**:1467

22 nodes for 30 minutes on edison: 240 CPU-hours

OMFIT solves common computing issues

Issue	OMFIT solution
Get experimental setup	Access experimental data and facilitate interpretive analysis
Arcane input format files	Python classes for parsing and writing broad range of files
Technical guide of code is too deep for average user	Provide customizable GUIs and tutorials
Code only runs at a few locations	Package OMFIT to be portable, and allow remote execution of arbitrary scripts, even with tunnels or MFA
Remote visualization of results is difficult	**In-situ processing with local rendering**

OMFIT offers great visualization (via Mayavi and matplotlib)



GPEC is a suite of codes for calculating magnetohydrodynamic (MHD) instabilities

Shown here is the 3D plasma response to the coils, such as would be applied to suppress edge localized modes (ELMs)

Each user does not have to relearn plotting

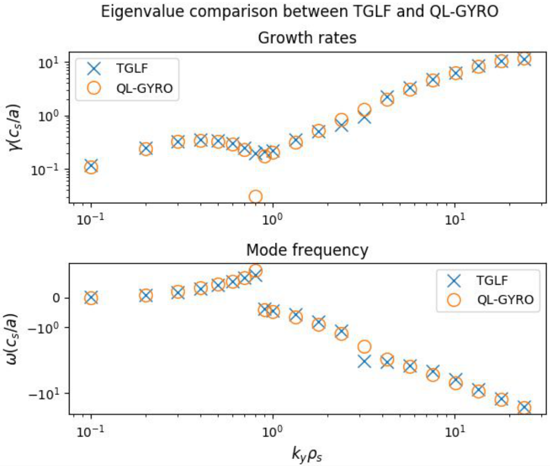
N.C. Logan, et al. Phys. Plasmas **20**, 122507

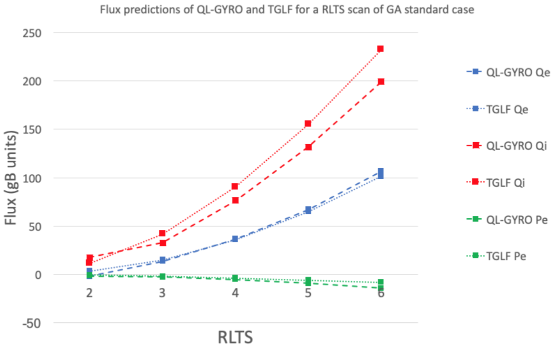
OMFIT solves common computing issues

Issue	OMFIT solution
Get experimental setup	Access experimental data and facilitate interpretive analysis
Arcane input format files	Python classes for parsing and writing broad range of files
Technical guide of code is too deep for average user	Provide customizable GUIs and tutorials
Code only runs at a few locations	Package OMFIT to be portable, and allow remote execution of arbitrary scripts, even with tunnels or MFA
Remote visualization of results is difficult	In-situ processing with local rendering
Tedious to run scans	**Automate parsing inputs, creating directories, plotting**

OMFIT automates the parsing of inputs, creating directories, and plotting across scans

The **GYRO** code solves the gyrokinetic equation for the turbulent linear growth rates or nonlinear fluxes given input local parameters





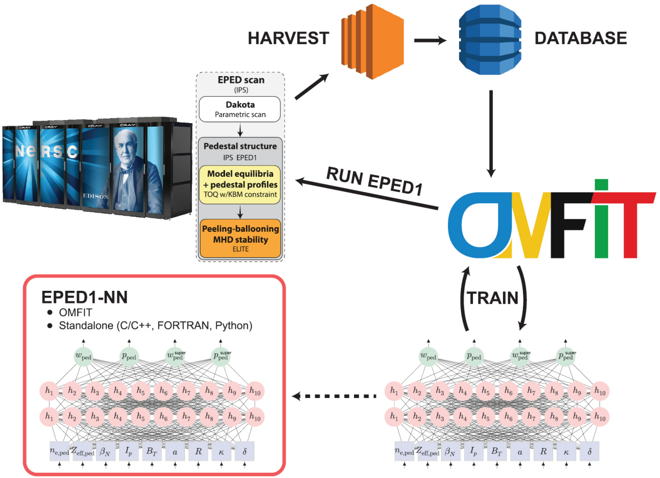
J. Candy and R.E. Waltz. J. Comput. Phys., **186**:545

J. Candy, E.A. Belli, and R.V. Bravenec. J. Comput. Phys., **324**:73

OMFIT solves common computing issues

Issue	**OMFIT solution**
Get experimental setup	Access experimental data and facilitate interpretive analysis
Arcane input format files	Python classes for parsing and writing broad range of files
Technical guide of code is too deep for average user	Provide customizable GUIs and tutorials
Code only runs at a few locations	Package OMFIT to be portable, and allow remote execution of arbitrary scripts, even with tunnels or MFA
Remote visualization of results is difficult	In-situ processing with local rendering
Tedious to run scans	Automate parsing inputs, creating directories, plotting
HPC results cannot be used directly by community	**Harvest results and build NN**

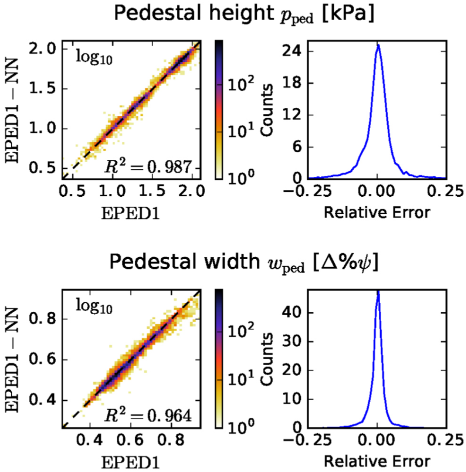
OMFIT builds databases, then trains machine learning models as method for whole community to benefit from HPC usage



OMFIT has compiled a variety of databases from scans over experimental conditions or simulation parameters and built corresponding neural networks

- Experimental heat flux
- Pedestal height/width
- Bootstrap current
- Turbulence heat flux

OMFIT builds databases, then trains machine learning models as method for whole community to benefit from HPC usage



EPED-1 predicts pedestal height and width given 8 inputs

- Stability evaluated for scans of model equilibria
- The IPS (Integrated Plasma Simulator) framework parallelized the workflow to run on ~ 760 CPUs, taking ~ 3 minutes
- EPED-1 was run for a database of $\sim 20k$ existing and future experimental conditions
- EPED results harvested then used for training neural network

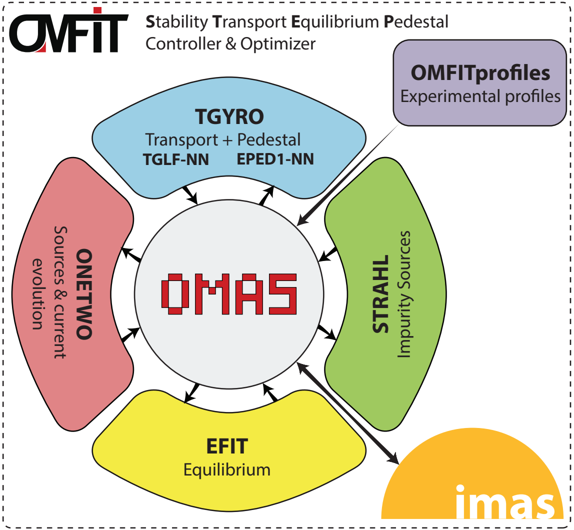
P.B. Snyder et al. Phys. Plasmas **16**:056118

GENERAL ATOMICS

OMFIT solves common computing issues

Issue	OMFIT solution
Get experimental setup	Access experimental data and facilitate interpretive analysis
Arcane input format files	Python classes for parsing and writing broad range of files
Technical guide of code is too deep for average user	Provide customizable GUIs and tutorials
Code only runs at a few locations	Package OMFIT to be portable, and allow remote execution of arbitrary scripts, even with tunnels or MFA
Remote visualization of results is difficult	In-situ processing with local rendering
Tedious to run scans	Automate parsing inputs, creating directories, plotting
HPC results cannot be used directly by community	Harvest results and build NN
Code run is part of larger integrated problem	**String together tasks regardless of code run type (local serial or queued remote HPC)**

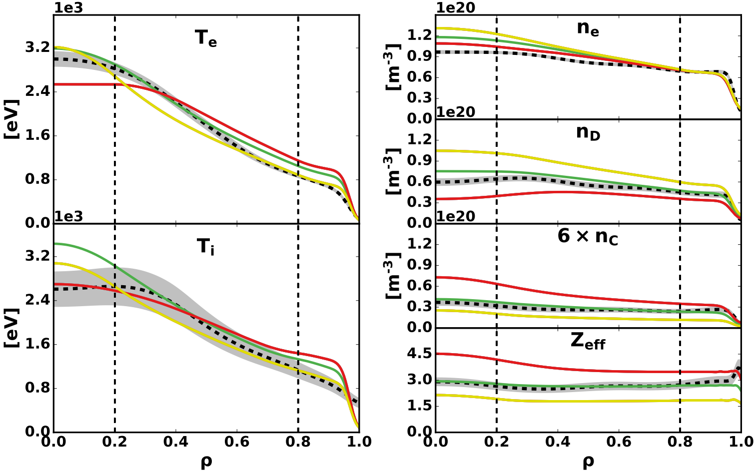
OMFIT can string together tasks regardless of code run type (local/remote or serial/batch) to perform integrated modeling with experimental validation



The STEP module combines many different physics codes to obtain a self-consistent prediction of core kinetic profiles

- Serial local
  - TGLF — G.M. Staebler, et al. Phys. Plasmas, **14**:055909
  - NEO — E.A. Belli, et al. Plasma Phys. Control. Fus., **51**:075018
  - TGYRO — J. Candy, et al. Phys. Plasmas, **16**:060704
- Serial remote
  - ONETWO — W.W. Pfeiffer GA-A16178
  - EFIT — L. Lao, et al. Nuc. Fus.**25**:1611
  - STRAHL — K. Behringer JET-R(87)08
- Batch remote
  - EPED — P.B. Snyder et al. Phys. Plasmas **16**:056118

OMFIT can string together tasks regardless of code run type (local/remote or serial/batch) to perform integrated modeling with experimental validation



- Self-consistent core, edge, pedestal, impurity simulations of **DIII-D**
- Colors are (0.5 , 1.0 , 1.5) $\times$ experimental $\int n_C \, d\rho$
- The experimental profiles are shown in gray for comparison

OMFIT enables computing usage across fusion topics

- OMFIT streamlines the tedious parts of running existing codes
- OMFIT has a convenient API for accessing remote (even HPC) resources
- OMFIT has a convenient API for creating application specific Graphical User Interaces
- Code sharing is encouraged

Homework - Part 2

1. Consult with your advisor to see if OMFIT or plasmapy would be useful for your project. If OMFIT would be useful, do the following:
2. Watch the first OMFIT video tutorial
3. Determine whether you will be using a public installation or installing a personal copy
4. Sign up for GitHub, if not already
5. Fill out the OMFIT User Agreement

If you feel that I missed out on some fusion relevant software efforts, please send me more information, and we can be sure to expand the horizons of this topic in the future.

Weekly community meetings - come join us

OMFIT Mondays at 11 PT/14 ET `https://discord.gg/wJAAFWh`

PlasmaPy Tuesdays at 11 PT/14 ET `https://www.plasmapy.org/meetings/weekly/`

References I

(1) https://en.wikipedia.org/wiki/QBasic.

(2) https://en.wikipedia.org/wiki/C%2B%2B.

(3) https://en.wikipedia.org/wiki/Mathcad.

(4) https://www.maplesoft.com/products/Maple/.

(5) https://en.wikipedia.org/wiki/Unix.

(6) https://en.wikipedia.org/wiki/Linux.

(7) https://www.harrisgeospatial.com/Software-Technology/IDL.

(8) https://python.org.

(9) https://www.mathworks.com/products/matlab.html.

(10) https://en.wikipedia.org/wiki/Fortran.

(11) https://omfit.io.

(12) http://numerical.recipes/.

(13) http://www.netlib.org/blas/.

References II

(14) http://www.netlib.org/lapack/.

(15) https://boutproject.github.io/.

(16) https://www.hdfgroup.org/solutions/hdf5/.

(17) https://www.unidata.ucar.edu/software/netcdf/.

(18) https://mdsplus.org/.

(19) https://docs.conda.io/.

(20) https://imas.iter.org.

(21) https://w3.pppl.gov/ntcc/.

(22) https://atom.scidac.io/.

(23) https://www.plasmapy.org/.

(24) https://yaml.org/.

(25) https://github.com/openPMD.

(26) Michael Metcalf, John Reid, and Malcolm Cohen. *fortran 95/2003 explained*. Oxford Univ. Press, 2004.

References III

(27) http://www.sscce.org.

(28) https://en.wikipedia.org/wiki/Concurrent_Versions_System.

(29) https://subversion.apache.org/.

(30) https://git-scm.com/.

(31) http://www.catb.org/~esr/writings/cathedral-bazaar/.