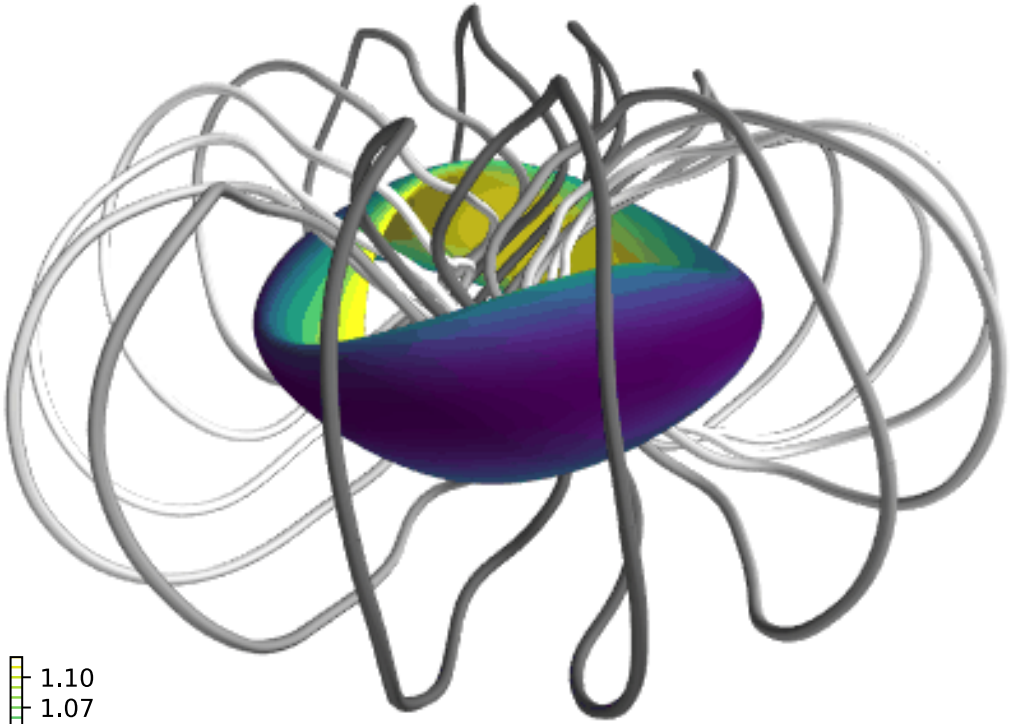
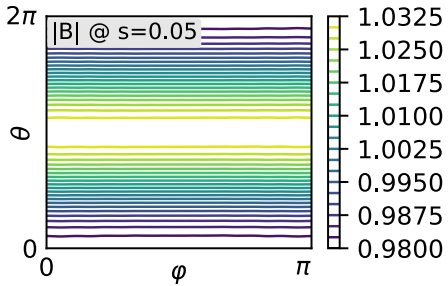
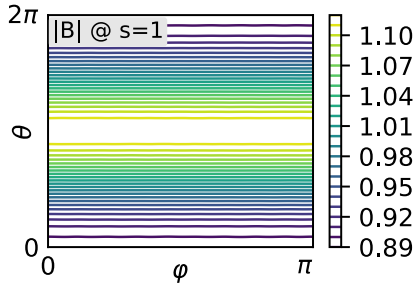
SIMSOPT status

- How to get involved
- Science highlights
- Software aspects
- To do
- Discussion questions







Ways to get involved with simsopt

- 100% open source. Anyone can fork the repository and edit the code. github.com/hiddenSymmetries/simsopt
- Can install via pre-compiled binaries (python ”wheels”), conda package, containers (Docker, Singularity, & Shifter), or source.
- Simsopt slack channel. Let me know if you need an invite.
- Group meeting on zoom every Monday 9am Eastern. Announcements on slack #developers channel.
- Would be nice if more Hidden Symmetries team members contributed code. Please contribute!
- Lots of code has been contributed by people not funded by Simons.

hiddenSymmetries / simsopt

Public

Unpin | Unwatch 3 | Fork 10 | Star 30

Code | Issues 8 | Pull requests 2 | Discussions | Actions | Projects 1 | Wiki | Security | Insights

master | 70 branches | 23 tags | Go to file | Add file | Code

landreman Merge pull request #195 from hiddenSymmetries/ml/profiles ... f577f40 5 days ago 2,410 commits

.github/workflows	spec import updated	15 days ago
ci	fix pmix installation for openmpi	last month
conda.recipe	Tidying up requirements	last month
docs	Merge branch 'master' into ml/profiles	12 days ago
examples	Minor formatting	29 days ago
src	helicity_n now optional in j_dot_B_Redl. In Quasisymmetry, renamed ...	5 days ago
tests	helicity_n now optional in j_dot_B_Redl. In Quasisymmetry, renamed ...	5 days ago
thirdparty	added a regular grid interpolator	10 months ago
.coveragerc	Removing 3.10 to wait for dependencies to release compatible whe...	4 months ago
.gitignore	Ignore the _version.py generated by setuptools_scm	3 months ago
.gitmodules	back to using libeigen repo	last month

About

Simons Stellarator Optimizer Code

simsopt.readthedocs.io

plasma | optimization | fusion | plasma-physics | nuclear-fusion | stellarator | stellarators

Readme

LGPL-3.0 License

30 stars

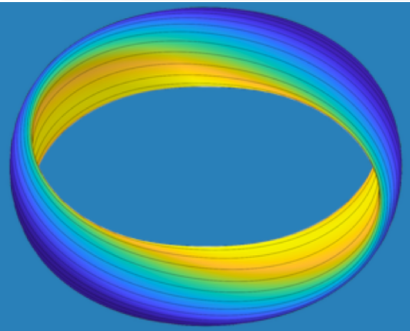
3 watching

10 forks

Releases 13

v0.7.0 Latest

on Jan 18



latest

Search docs

CONTENTS

- Getting started
- Docker container
- Concepts
- Defining optimization problems
- Testing
- Source code on GitHub
- Publications
- Contributing to Simsopt

EXAMPLES

Simsopt documentation

`simsopt` is a framework for optimizing stellarators. The high-level routines are in python, with calls to C++ or fortran where needed for performance. Several types of components are included:

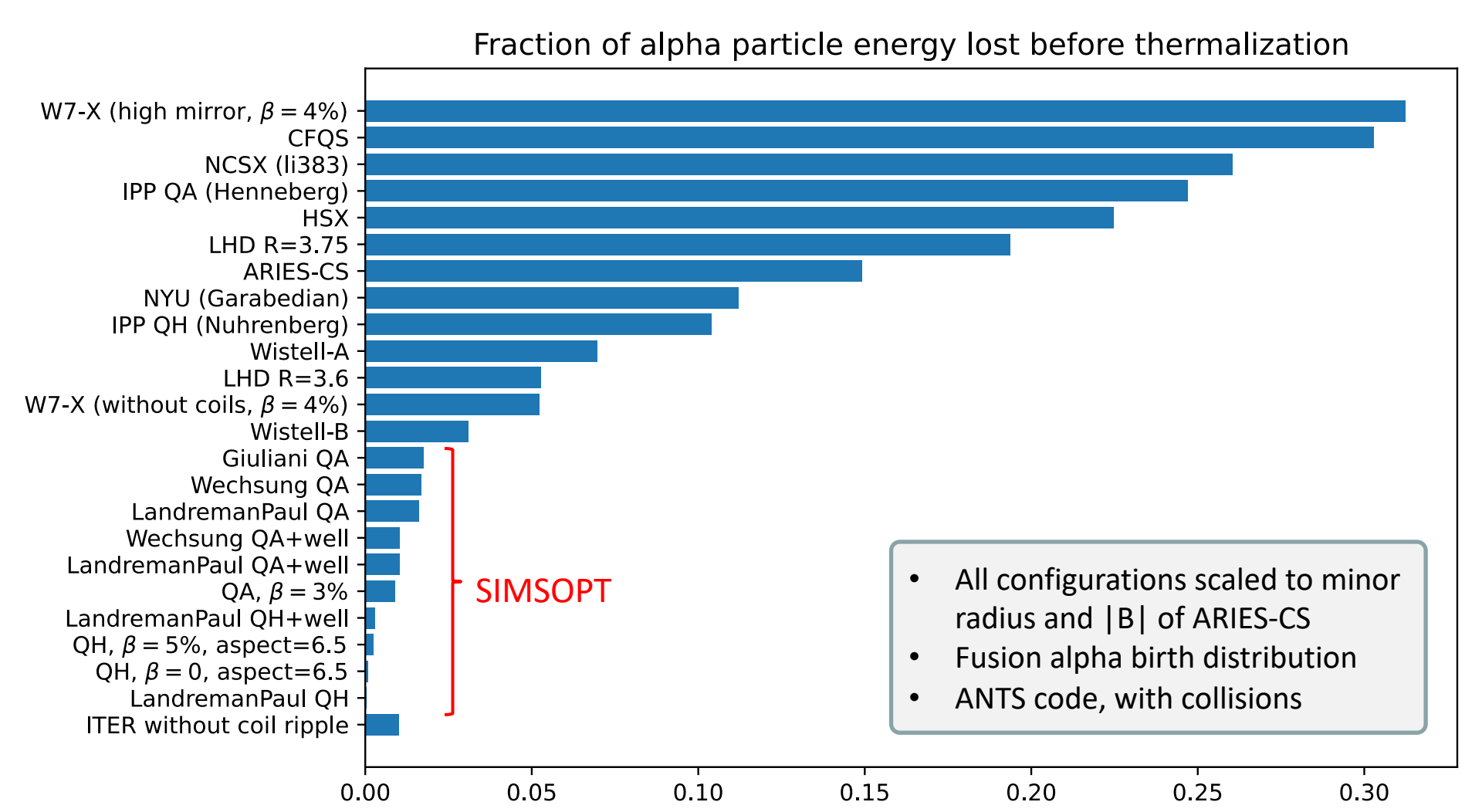
- Interfaces to physics codes, e.g. for MHD equilibrium.
- Tools for defining objective functions and parameter spaces for optimization.
- Geometric objects that are important for stellarators – surfaces and curves – with several available parameterizations.
- Efficient implementations of the Biot-Savart law and other magnetic field representations, including derivatives.
- Tools for parallelized finite-difference gradient calculations.

The design of `simsopt` is guided by several principles:

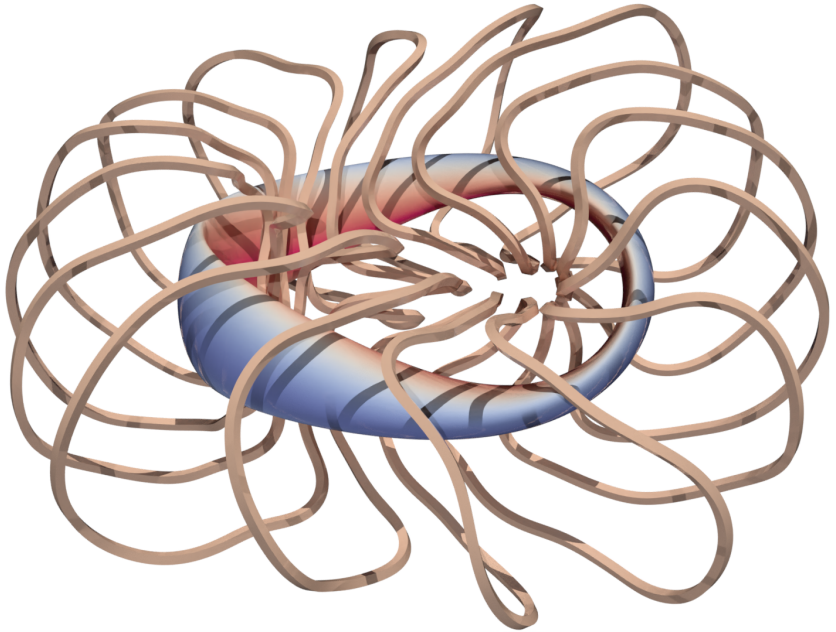
- Thorough unit testing, regression testing, and continuous integration.
- Extensibility: It should be possible to add new codes and terms to the objective function without editing modules that already work, i.e. the open-closed principle . This is because any edits to working code can potentially introduce bugs.
- Modularity: Physics modules that are not needed for your optimization problem do not need to be installed. For instance, to optimize SPEC equilibria, the VMEC module need not be installed.
- Flexibility: The components used to define an objective function can be re-used for applications other than standard optimization. For instance, a `simsopt` objective function is a standard

Includes API documentation scraped from source code

`simsopt` is fully open-source, and anyone is welcome to use it, make suggestions, and contribute.

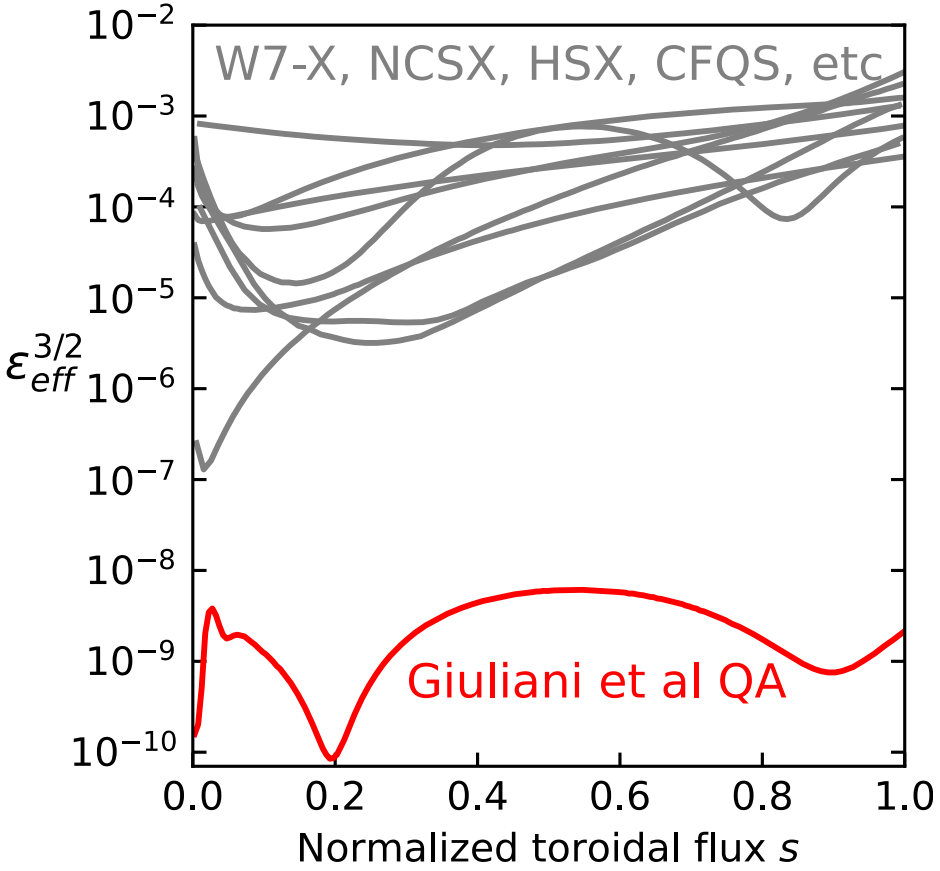

Fraction of alpha particle energy lost before thermalization
W7-X (high mirror, $\beta = 4\%$)
CFQS
NCSX (li383)
IPP QA (Henneberg)
HSX
LHD R=3.75
ARIES-CS
NYU (Garabedian)
IPP QH (Nuhrenberg)
Wistell-A
LHD R=3.6
W7-X (without coils, $\beta = 4\%$)
Wistell-B
Giuliani QA
Wechsung QA
LandremanPaul QA
Wechsung QA+well
LandremanPaul QA+well
QA, $\beta = 3\%$
LandremanPaul QH+well
QH, $\beta = 5\%$, aspect=6.5
QH, $\beta = 0$, aspect=6.5
LandremanPaul QH
ITER without coil ripple
SIMSOPT
0.00
0.05
0.10
0.15
0.20
0.25
0.30
• All configurations scaled to minor radius and |B| of ARIES-CS
• Fusion alpha birth distribution
• ANTS code, with collisions

Single-stage optimization of coil shapes for quasisymmetry (Giuliani et al, arXiv):

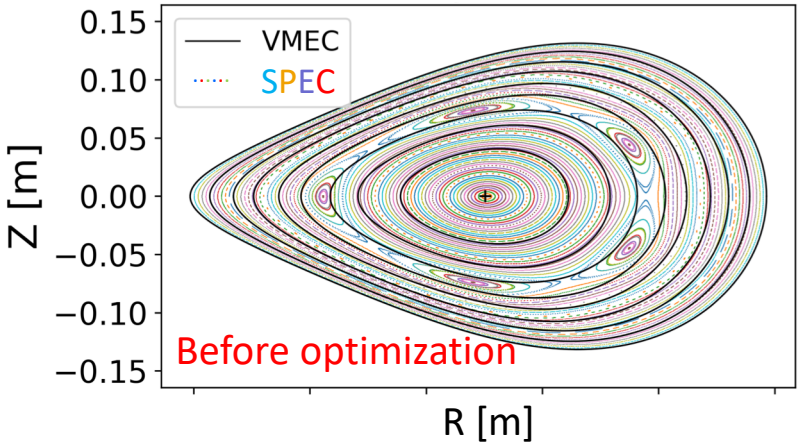


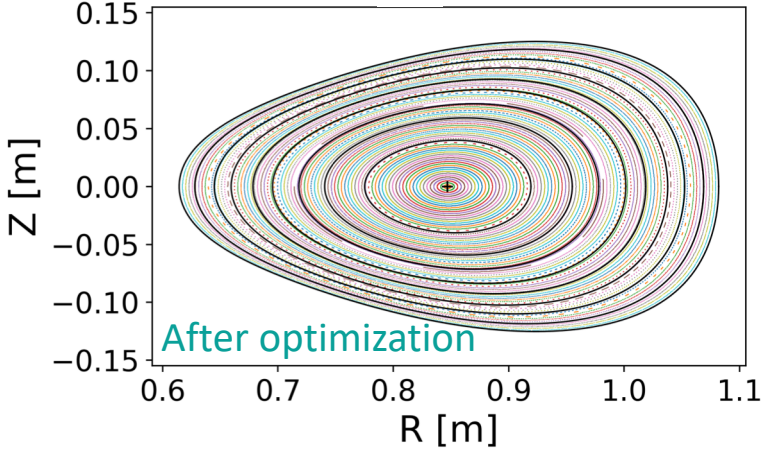
16 coils, $\varepsilon_{eff}^{3/2} < 5 \times 10^{-9}$

Collisional heat flux

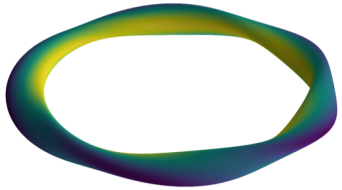


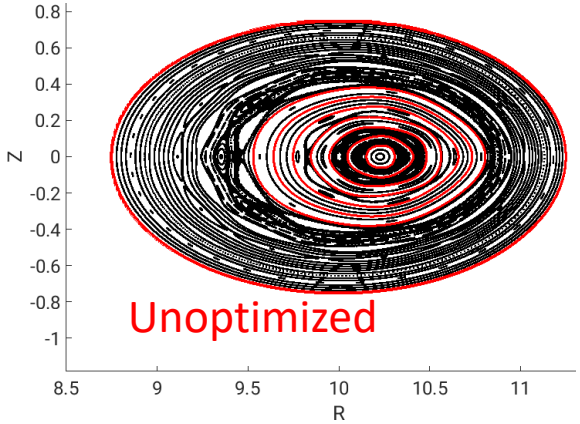
Combined VMEC+SPEC optimization for quasisymmetry + good flux surfaces (*Phys Plasmas* (2021))

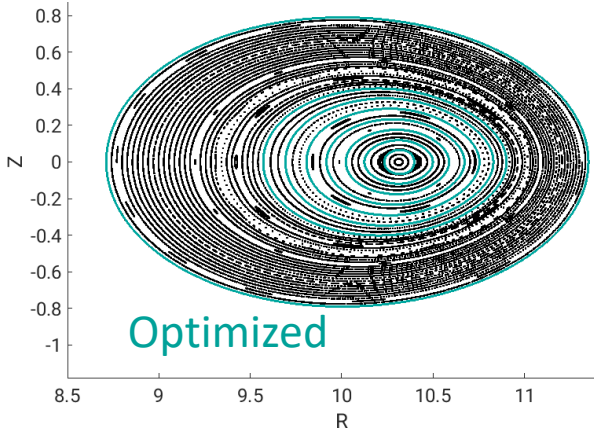




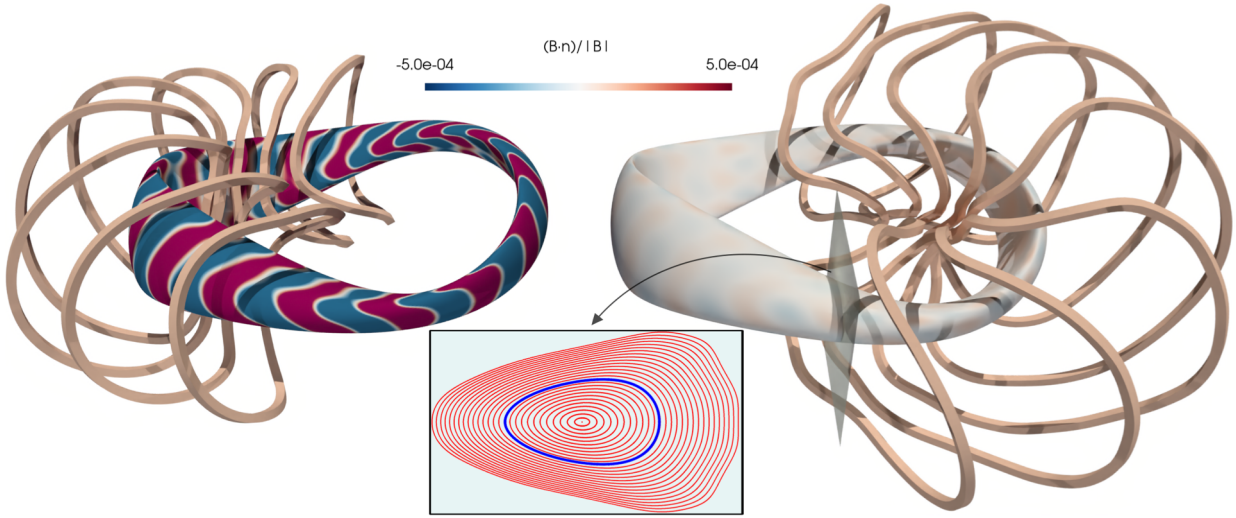
Optimization of SPEC configurations with pressure & current for good surfaces (A Baillod et al):







Coils for precise quasisymmetry & excellent confinement (F Wechsung et al)

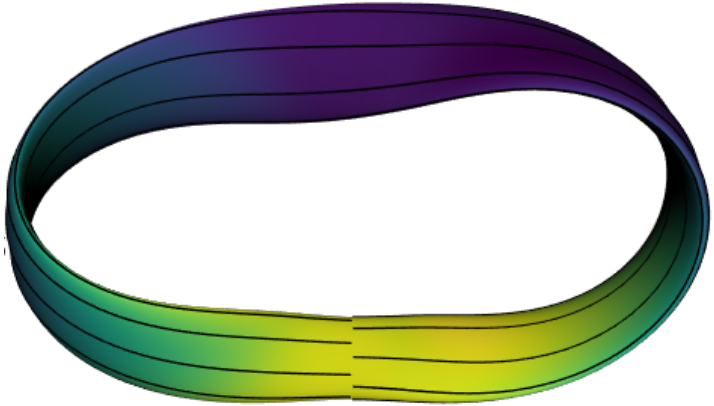


Fast derivative-based stochastic optimization available:

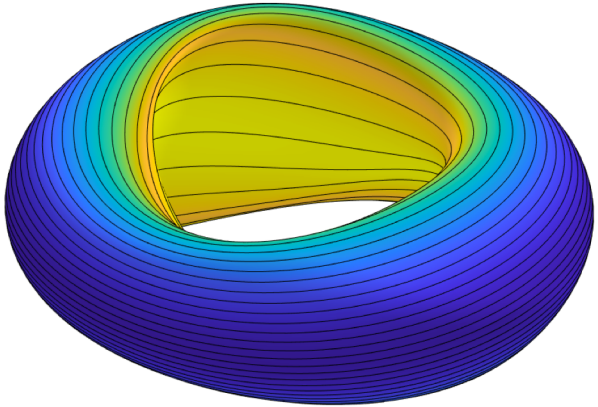
stage_two_optimization_stochastic.py

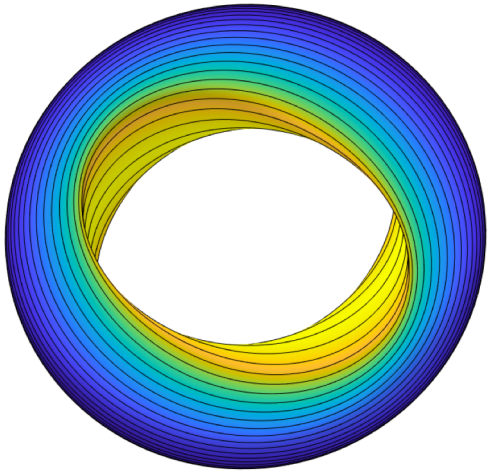
in simsopt/examples/2_Intermediate/

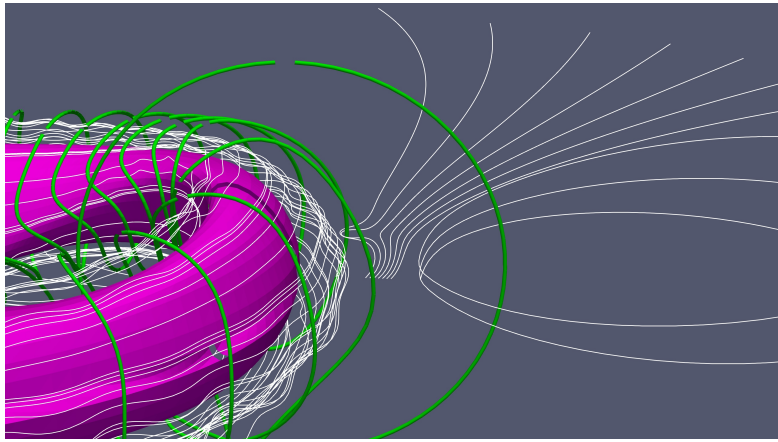
Optimization of near-axis configurations for quasi-isodynamic (R Jorge et al)



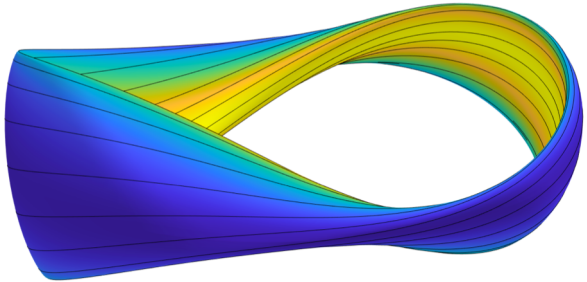
Candidate configurations for EPOS: (Coils by J Smoniewski)

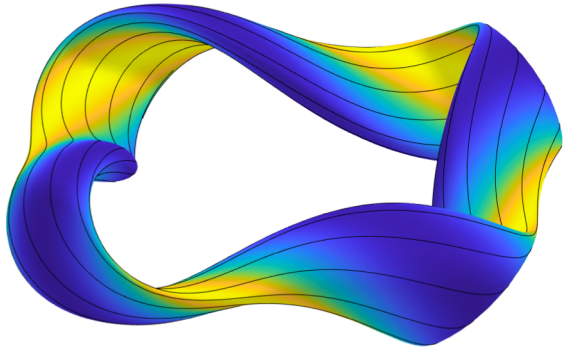


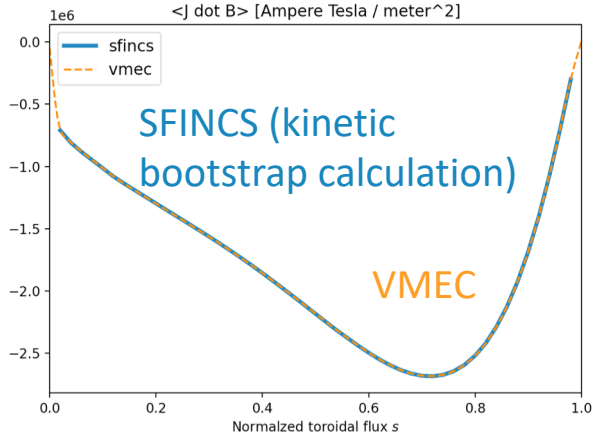




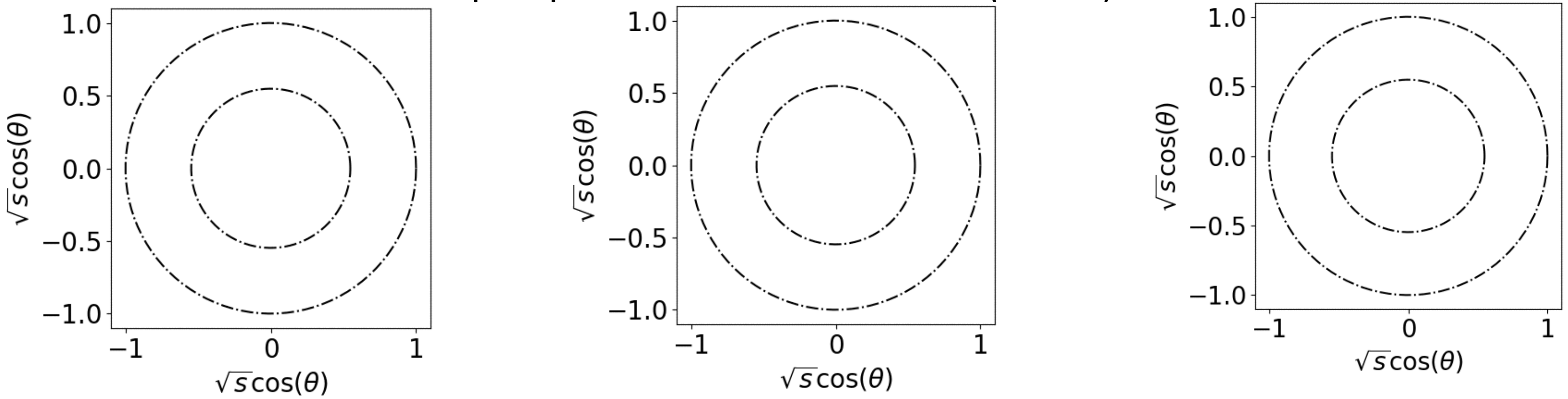
Finite-beta optimization with self-consistent bootstrap current:





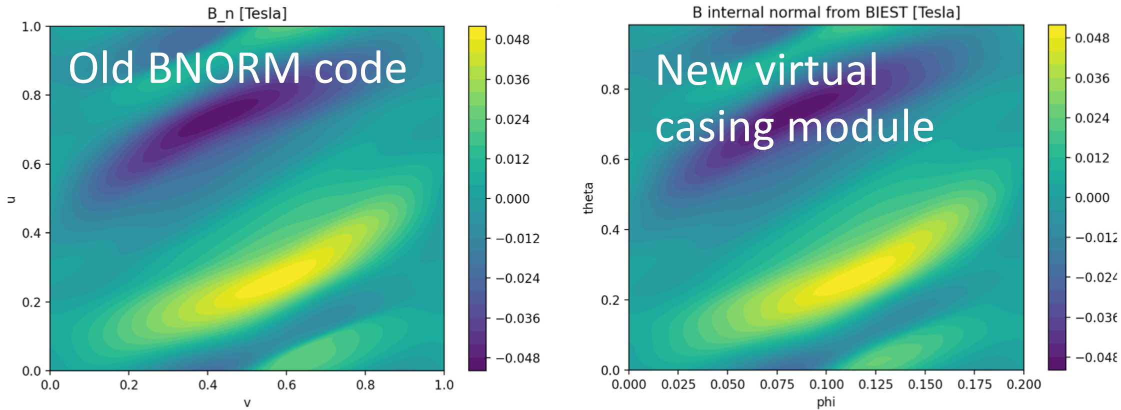


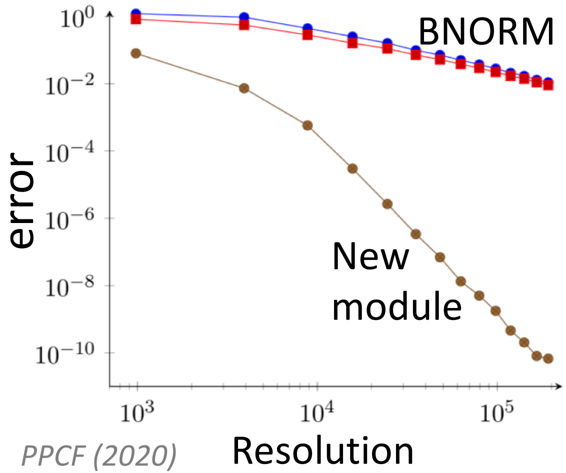
Alpha-particle loss mechanisms (E Paul)



Virtual casing, BNORM replacement (D Malhotra, et al, B Medasani)

```
VirtualCasing.from_vmec('wout_li383.nc', nphi=300)
```



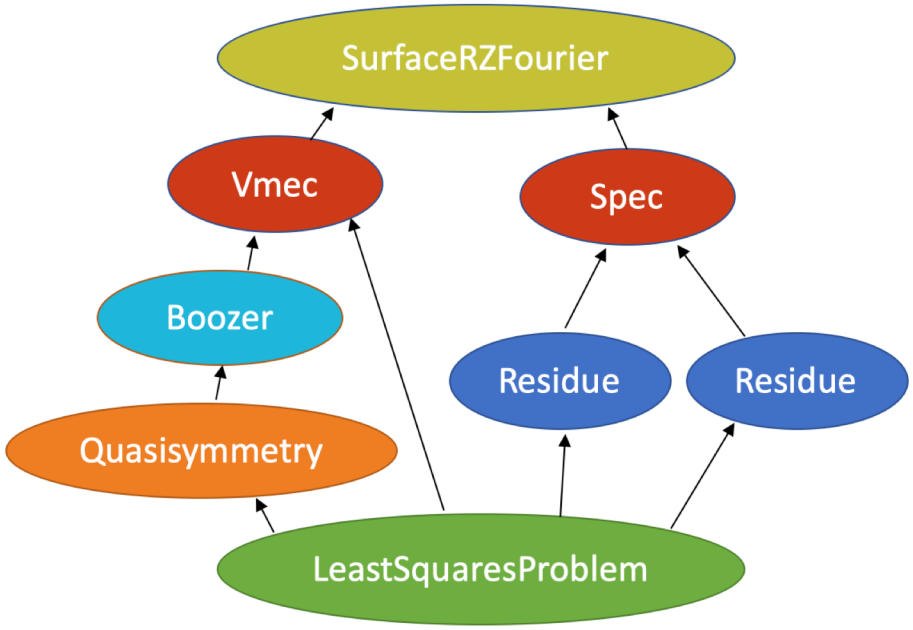


PPCF (2020)

Papers that include simsopt results

1. M Landreman, B Medasani, F Wechsung, A Giuliani, R Jorge, and C Zhu, “SIMSOPT: A flexible framework for stellarator optimization”, Journal of Open Source Software 6, 3525 (2021).
2. F Wechsung, A Giuliani, M Landreman, A Cerfon, and G Stadler, “Single-stage gradient-based stellarator coil design: stochastic optimization”, In press, Nuclear Fusion.
3. M Landreman, B Medasani, and C Zhu, “Stellarator optimization for good magnetic surfaces at the same time as quasisymmetry”, Physics of Plasmas 28, 092505 (2021).
4. A Bader, D T Anderson, M Drevlak, B J Faber, C C Hegna, S Henneberg, M Landreman, J C Schmitt, Y Suzuki, and A Ware, “Energetic particle transport in optimized stellarators”, Nuclear Fusion 61, 116060 (2021).
5. M Landreman and E Paul, “Magnetic fields with precise quasisymmetry for plasma confinement”, Physical Review Letters 128, 035001 (2022).
6. F Wechsung, M Landreman, A Giuliani, A Cerfon, and G Stadler, “Precise stellarator quasisymmetry can be achieved with electromagnetic coils”, Proceedings of the National Academy of Sciences (2022).
7. A Baillod, J Loizu, J P Graves, and M Landreman, “Stellarator optimization for good magnetic surfaces at finite β and toroidal current”, Submitted.
8. A Giuliani, F Wechsung, M Landreman, G Stadler, and A Cerfon, “Direct computation of magnetic surfaces in Boozer coordinates and coil optimization for quasi-symmetry”, Submitted.
9. F Wechsung, A Giuliani, M Landreman, A Cerfon, and G Stadler, “A-posteriori optimization for increasing manufacturing tolerances in stellarator coil design”, Submitted.
10. R Jorge, G G Plunk, M Drevlak, M Landreman, J-F Lobsien, K Camacho, and P Helander, “A single-field-period quasi-isodynamic stellarator”, Ready to submit.

Optimization problems are represented as a directed acyclic graph



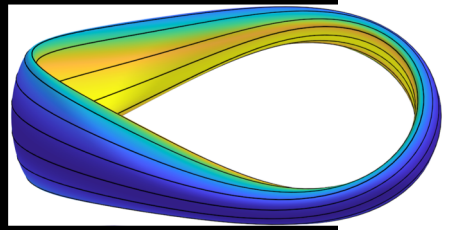
Graph framework handles:

- Dependencies & caching
- Assembly of global state vector from free dofs
- Naming of dofs
- Chain rule for derivatives

B Medasani, F Wechsung

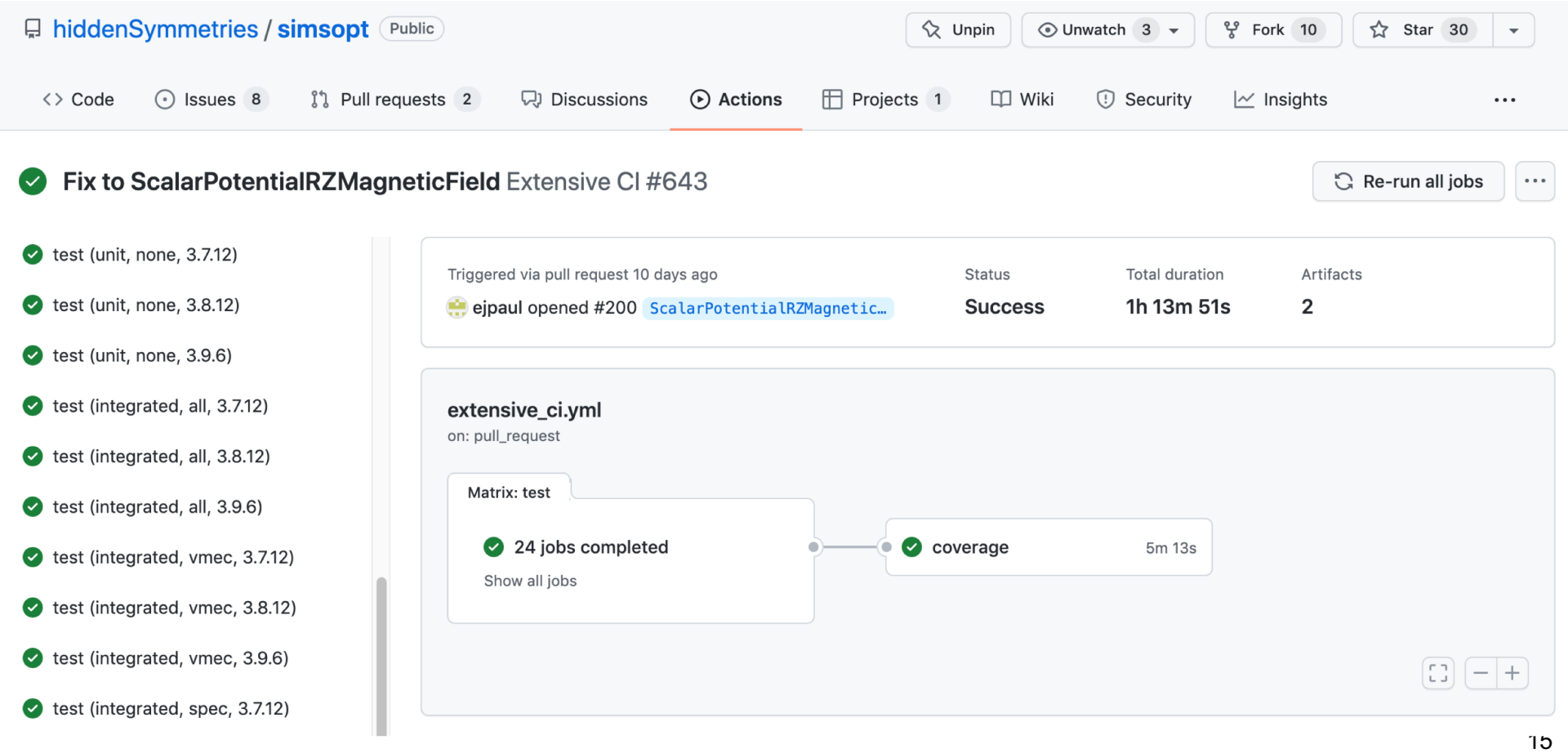
- There is no single simsopt executable. Problem specification with a python script (not a static input file) allows flexible modes of operation: notebooks, dynamic resolution, etc.

```
mpi = MpiPartition()
vmec = Vmec("input.QH_nfp2", mpi=mpi)

# Define objective function:
qs = QuasisymmetryRatioResidual(vmec, np.arange(0, 1.01, 0.1))

prob = LeastSquaresProblem.from_tuples([(vmec.aspect, 6, 1),
                                        (vmec.mean_iota, 0.42, 1),
                                        (qs.residuals, 0, 1)])
for step in range(4):  # Optimize in 4 steps.
    # Add Fourier modes to the parameter space:
    max_mode = step + 1
    vmec.boundary.fixed_range(mmin=0, mmax=max_mode,
                              nmin=-max_mode, nmax=max_mode, fixed=False)
    vmec.boundary.fix("rc(0,0)")

    # Increase code resolution:
    vmec.indata.mpol = 3 + step
    vmec.indata.ntor = vmec.indata.mpol

    # Run this optimization step:
    least_squares_mpi_solve(prob, mpi, grad=True)
```



- There is no single simsopt executable. Problem specification with a python script (not a static input file) allows flexible modes of operation: notebooks, dynamic resolution, etc.
- Any python function can go into the objective.
- A simsopt objective function is a standard python function. Can be used by any other python function or library.
- You can optimize with any new surface discretization or coil discretization without touching any code in simsopt itself. Minimizes bugs.
- Includes both derivative-based problems with analytic/automatic-differentiation/adjoint derivatives, and derivative-free problems.
- Mix of fast analytic derivatives (e.g. Biot-Savart) with automatic-differentiation for non-time-critical parts (e.g. finite coil thickness).

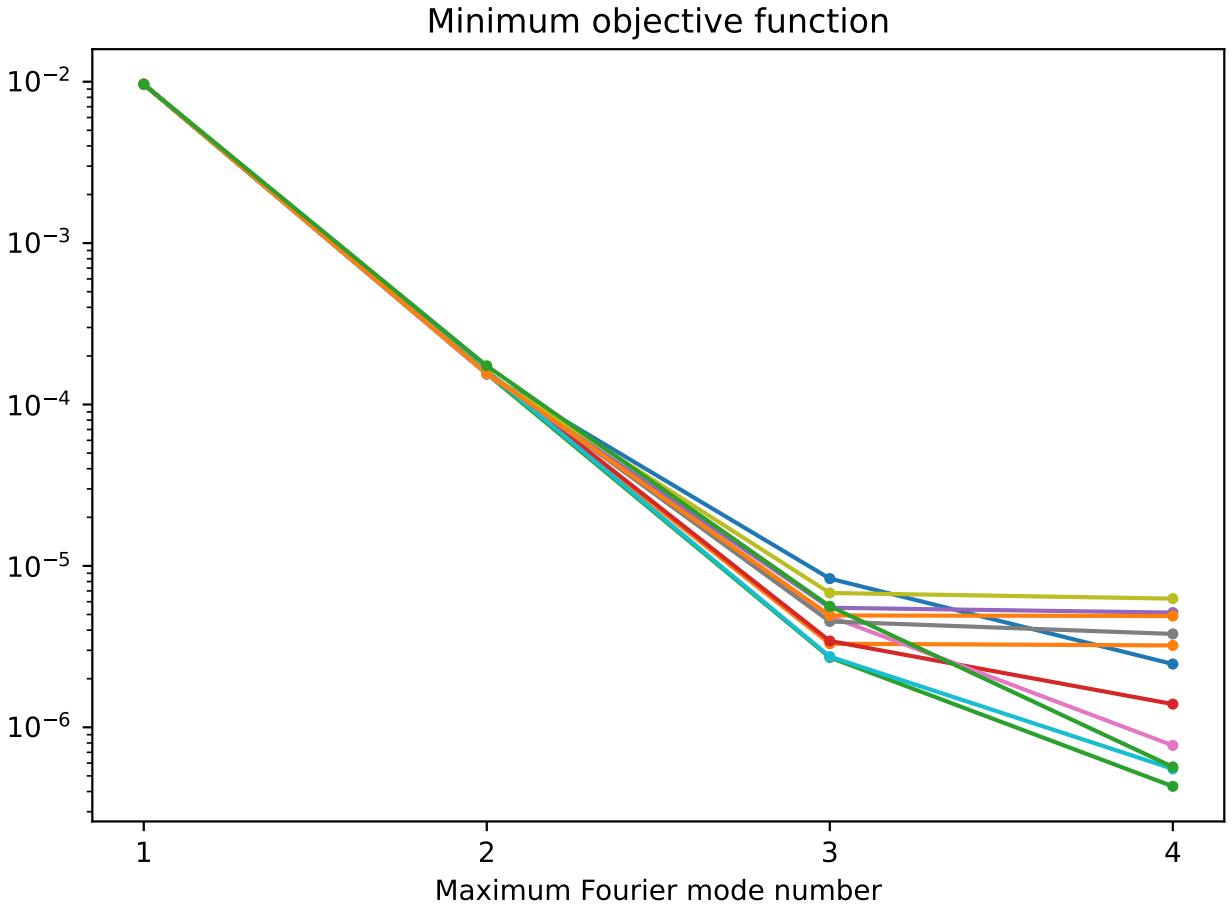
91% test coverage, with Github Actions continuous integration.
> 400 test cases run automatically every commit.



To do

- Although pre-compiled binaries (“wheels” & conda packages) & containers (Docker, shifter, & singularity) help, installation can still be complicated, especially vmec/spec.
- Long wish list of objectives: Quasi-isodynamic, MHD stability, other measures of confinement, turbulent transport, etc.
- Key piece we haven’t gotten to: system for calling external standalone MPI physics codes.
- Would be nice if more Hidden Symmetries team members contributed code. Please contribute!

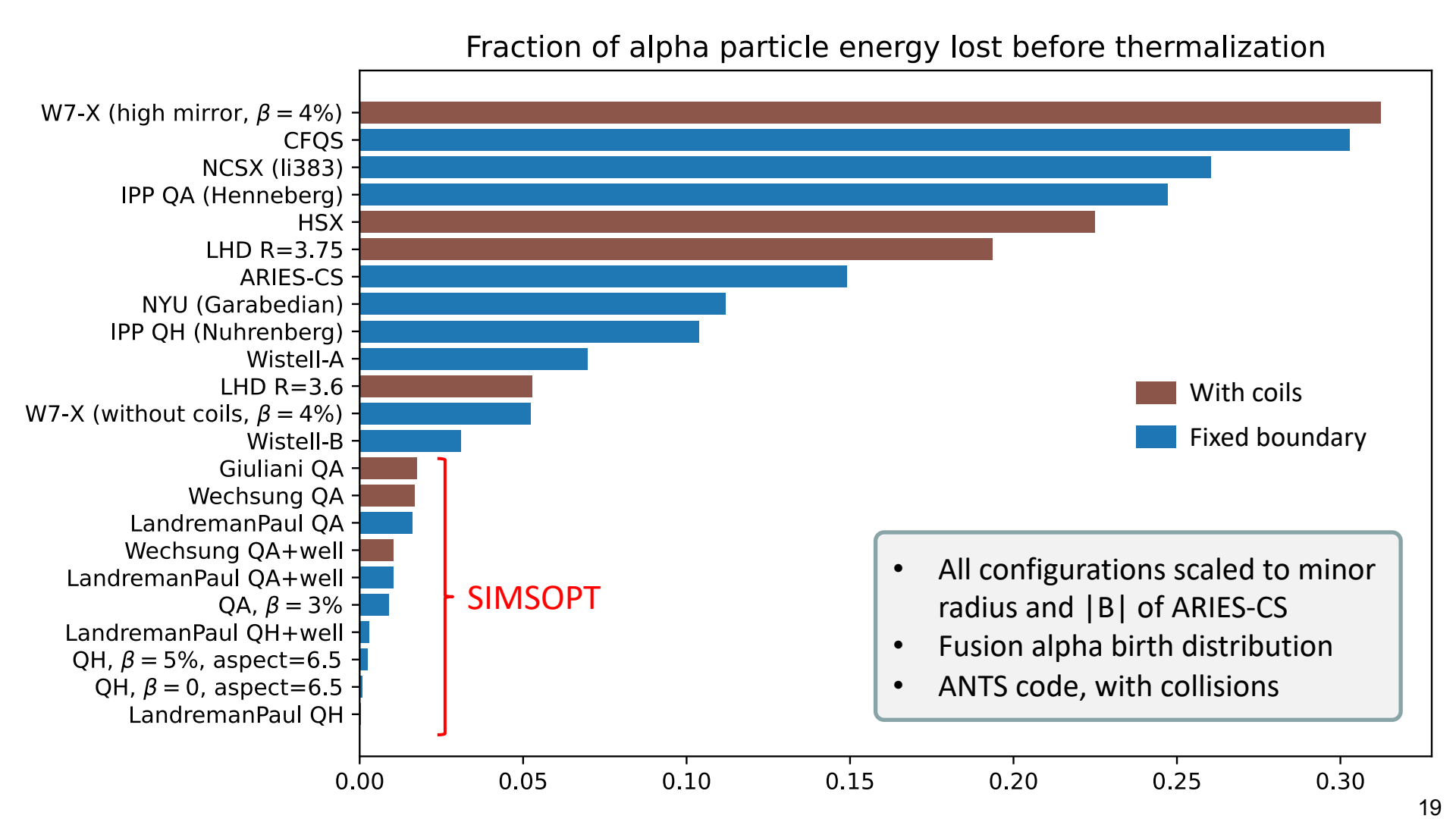
In stage 1, final objective is very sensitive. Can we understand & resolve this?



- Optimization runs differ only in random perturbation to initial condition
- Persists for all optimization algorithms & surface discretizations (RZFourier, Garabedian, Henneberg, real-space) tried so far

Discussion questions

- We have limited person-power, so can't do everything. What should be prioritized?
- What is everybody working on with simsopt now?
- What are you planning on contributing to simsopt in the coming months?
- Are there projects/tasks someone should be hired to do?
- Possible collaborations?
- How is the Monday meeting format going?


Fraction of alpha particle energy lost before thermalization
W7-X (high mirror, $\beta = 4\%$)
CFQS
NCSX (li383)
IPP QA (Henneberg)
HSX
LHD R=3.75
ARIES-CS
NYU (Garabedian)
IPP QH (Nuhrenberg)
Wistell-A
LHD R=3.6
W7-X (without coils, $\beta = 4\%$)
Wistell-B
Giuliani QA
Wechsung QA
LandremanPaul QA
Wechsung QA+well
LandremanPaul QA+well
QA, $\beta = 3\%$
LandremanPaul QH+well
QH, $\beta = 5\%$, aspect=6.5
QH, $\beta = 0$, aspect=6.5
LandremanPaul QH
SIMSOPT
With coils
Fixed boundary
• All configurations scaled to minor radius and |B| of ARIES-CS
• Fusion alpha birth distribution
• ANTS code, with collisions
0.00
0.05
0.10
0.15
0.20
0.25
0.30