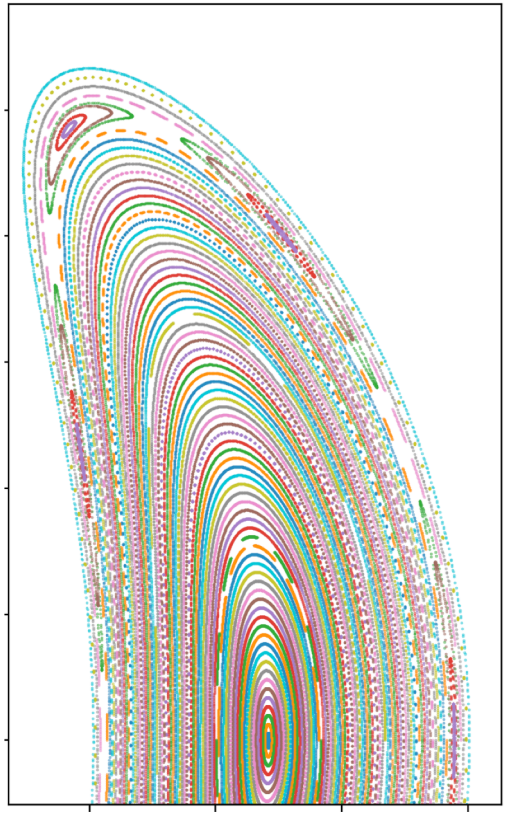
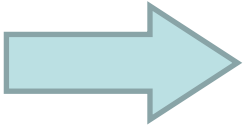
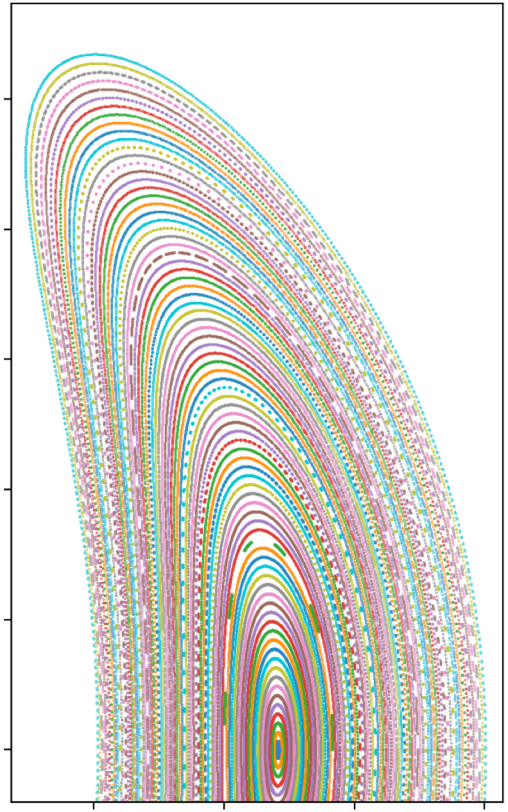
# Optimizing SPEC with SIMSOPT: Some preliminary results







Matt Landreman, University of Maryland, Jan 26, 2021 Spectaculars

# Outline

- Simsopt and interface to spec
- Benchmark: stellopt vs simsopt+vmec vs simsopt+spec.
- Optimization of Greene's residues in WISTELL-A.
- Next steps & how you can get involved.

# Simsopt: A new software infrastructure for stellarator optimization

- https://github.com/hiddenSymmetries/simsopt
- Proposed in Simons Foundation “Hidden symmetries & fusion energy” project.
- Everyone is invited to make suggestions, contribute, and use.
- Modular: Shouldn’t need to touch working code to connect a new code.
- Testing and continuous integration.
- Python drivers and infrastructure, calling to C++/fortran where needed for speed.
- Take advantage of python package system (pip install)
- Components:
  - Interfaces to physics codes
  - Surface & coil objects
  - Tools for defining objective function & parameter space, e.g. fixed vs free d.o.f.s
  - Parallelized finite differences

# The present simsopt-spec interface uses communication via files rather than via memory

- py_spec used to read namelists, run SPEC via subprocess, read .h5 output.
- pyoculus used to compute Greene's residue from solutions.
- Caoxiang has started new python wrapper with direct in-memory communication.

# Outline

- Simsopt and interface to spec
- Benchmark: stellopt vs simsopt+vmec vs simsopt+spec.
- Optimization of Greene's residues in WISTELL-A.
- Next steps & how you can get involved.

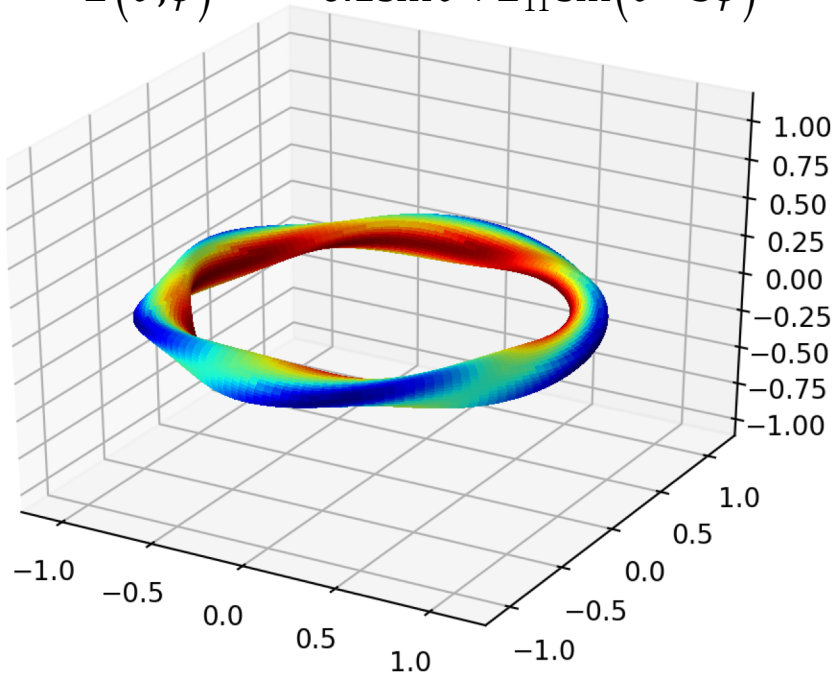
# Simsopt+spec gives same results as stellopt and simsopt+vmec for a few benchmark problems

github.com/landreman/stellopt_scenarios

Boundary:

$$R(\theta,\varphi) = 1 + 0.1\cos\theta + R_{11}\cos(\theta - 5\varphi),$$

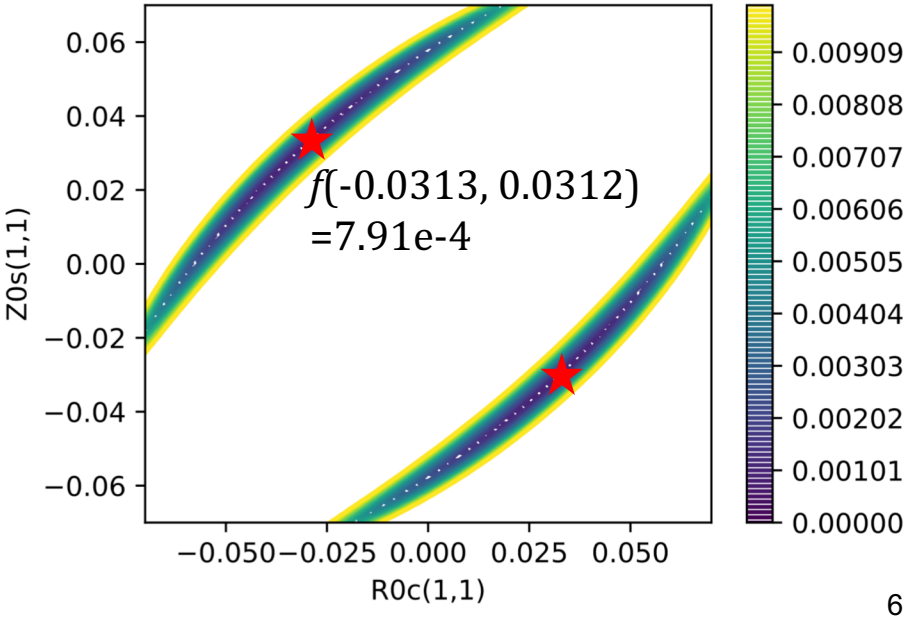
$$Z(\theta,\varphi) = \quad 0.1\sin\theta + Z_{11}\sin(\theta - 5\varphi)$$




Independent variables: $\{R_{11},\ Z_{11}\}$

Objective function:

$$f = (\iota_0 - 0.41)^2 + (\text{volume} - 0.15)^2$$



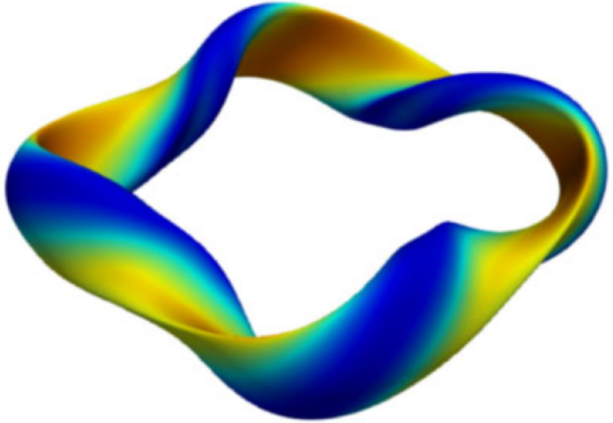
```
from simsopt import Spec, LeastSquaresProblem, least_squares_serial_solve

# Initialize SPEC from an input file:
equil = Spec("Spec2DOF_targetIotaAndVolume.sp")
surf = equil.boundary

# Define the parameter space:
surf.all_fixed()
surf.set_fixed("rc(1,1)", False)
surf.set_fixed("zs(1,1)", False)

# Define the objective function:
desired_volume = 0.15
desired_iota = -0.41
volume_weight = 1
iota_weight = 1
prob = LeastSquaresProblem([(equil.volume, desired_volume, volume_weight),
                            (equil.iota, desired_iota, iota_weight)])

# Solve the minimization problem:
least_squares_serial_solve(prob)
```

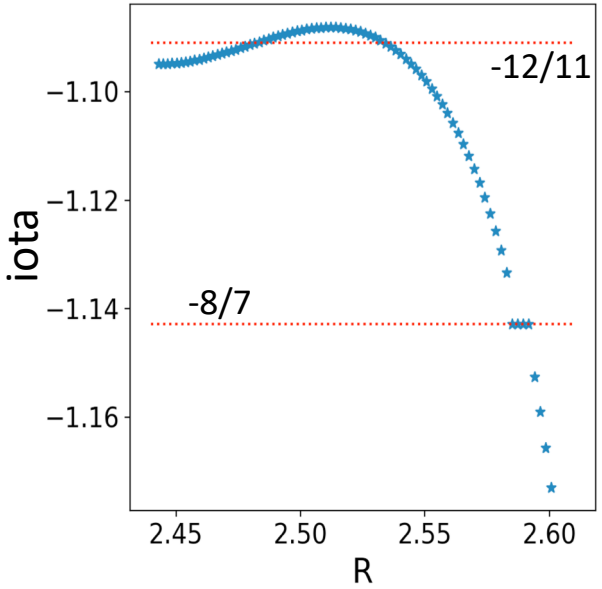
# Outline

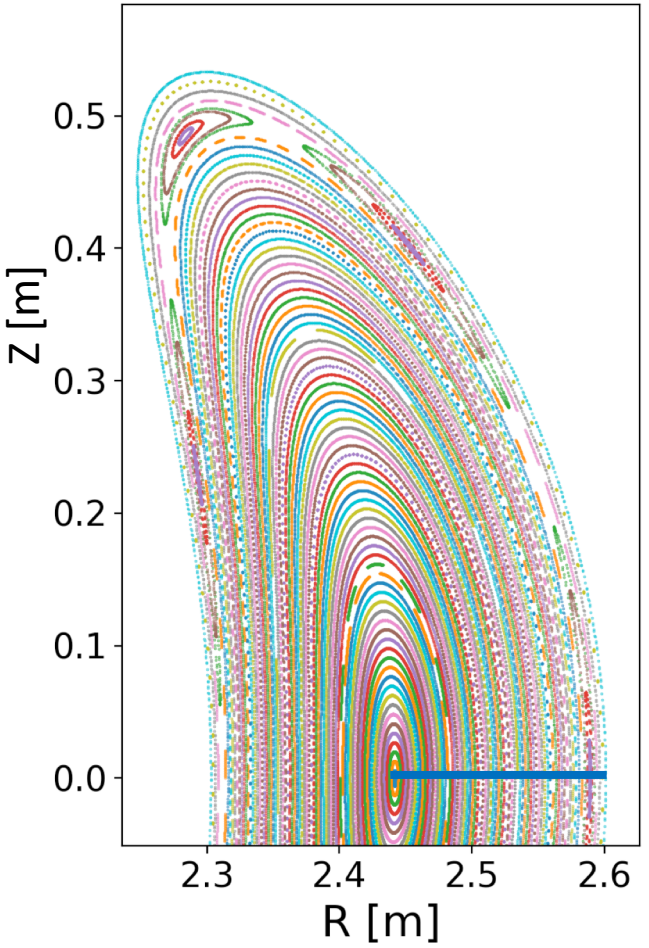
- Simsopt and interface to spec
- Benchmark: stellopt vs simsopt+vmec vs simsopt+spec.
- Optimization of Greene's residues in WISTELL-A.
- Next steps & how you can get involved.

# WISTELL-A (a.k.a. Aten) is an interesting configuration which needs optimization for better surfaces

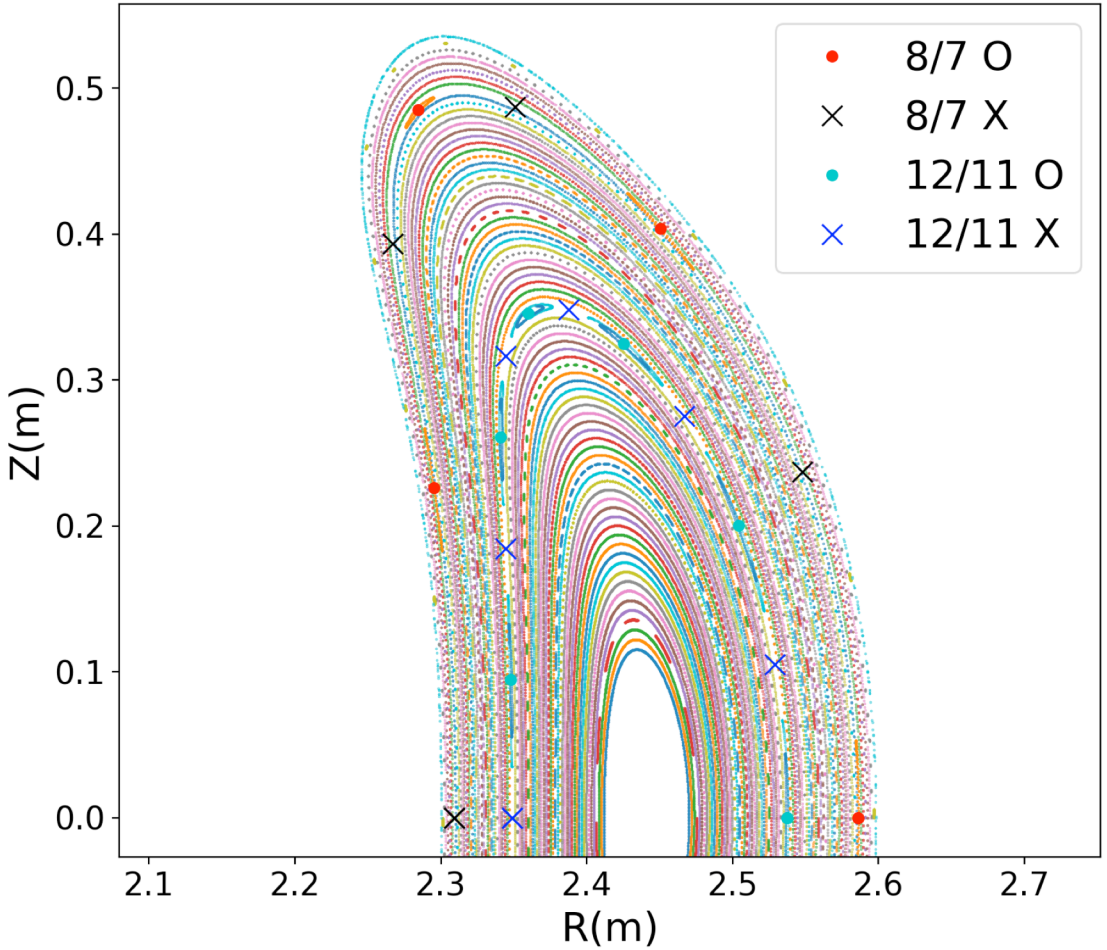
- QH configuration from Wisconsin
- Bader et al, JPP (2020)
- Result of optimization with ROSE + VMEC.
- Vacuum field





# Pyoculus finds the X/O points and Greene's residues




$$\text{residue} = \frac{1}{4}\left(2 - \mathrm{Tr}\,\vec{\vec{M}}\right)$$

$\vec{\vec{M}}$ = "full orbit tangent map":

$$\begin{pmatrix} \delta R \\ \delta Z \end{pmatrix}_{\text{final}} = \vec{\vec{M}} \begin{pmatrix} \delta R \\ \delta Z \end{pmatrix}_{\text{initial}}$$

O-point if residue $\in (0, 1)$

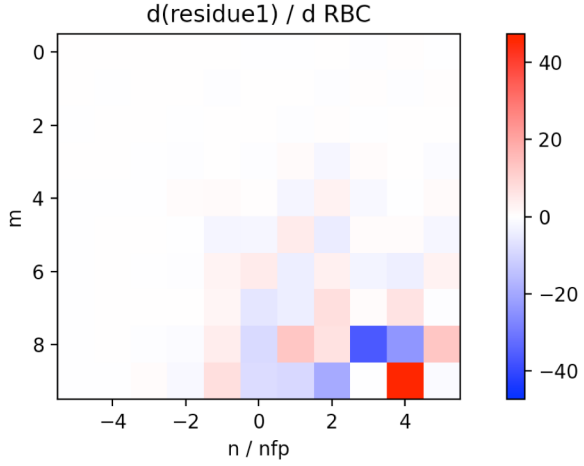
X-point if residue $< 0$ or $> 1$.

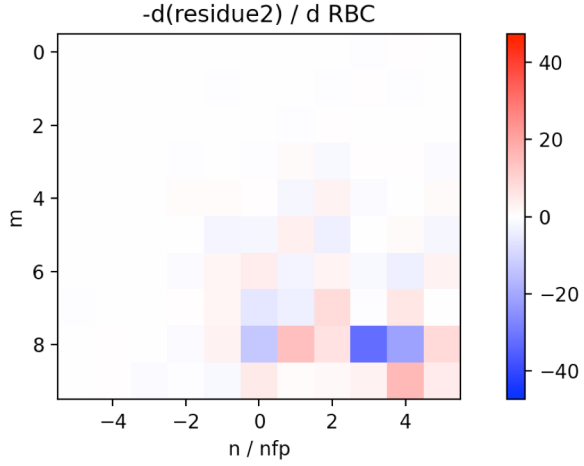
Good surface if residue = 0.

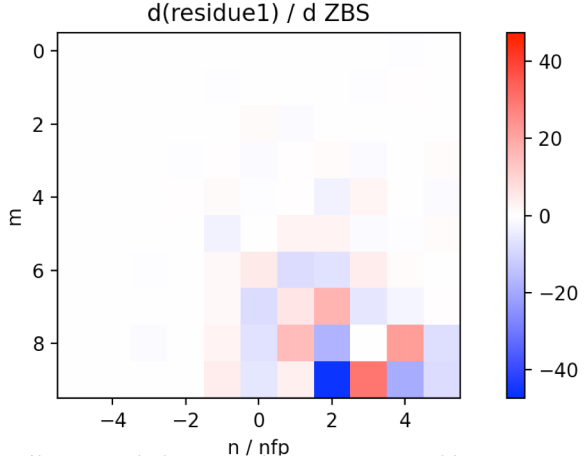
Greene, *J Math Phys* (1979)

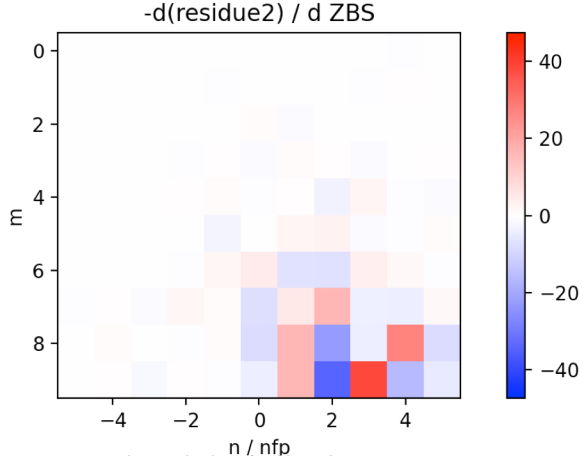
# The residues are most sensitive to high-m modes in the boundary shape

iota = -8/7
residues:



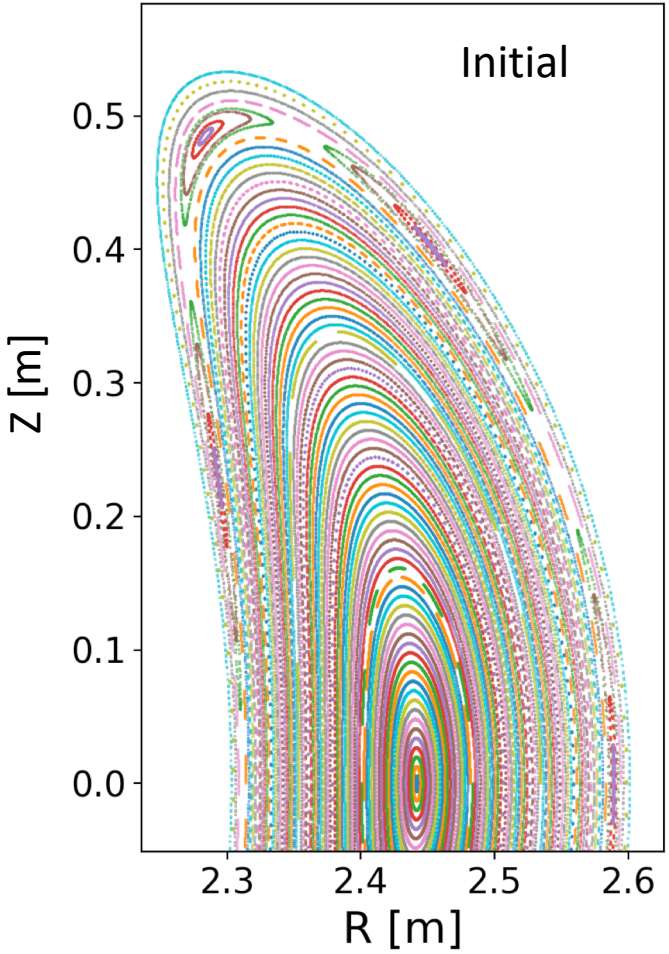



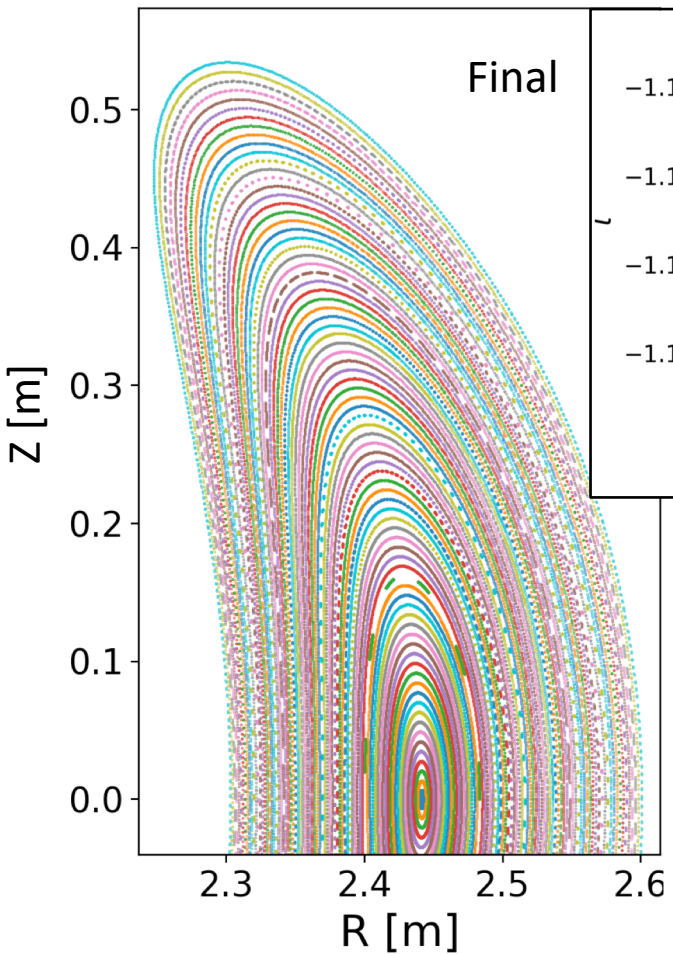



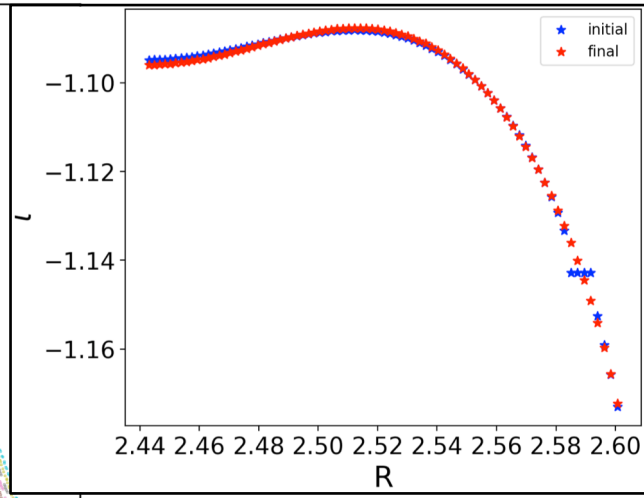



```
from simsopt import Spec, Residue, \
    LeastSquaresProblem, least_squares_serial_solve

spec = Spec('aten.sp')
spec.boundary.change_resolution(mpol=12, ntor=12)

# Define the parameter space:
spec.boundary.all_fixed()
spec.boundary.set_fixed('zs(6,1)', False)

# Resonant surface is iota = p / q:
p = -8
q = 7
residue1 = Residue(spec, p, q)
residue2 = Residue(spec, p, q, theta=np.pi)

# Objective function is \sum_j residue_j ** 2
prob = LeastSquaresProblem([(residue1, 0, 1),
                            (residue2, 0, 1)])

least_squares_serial_solve(prob)
```

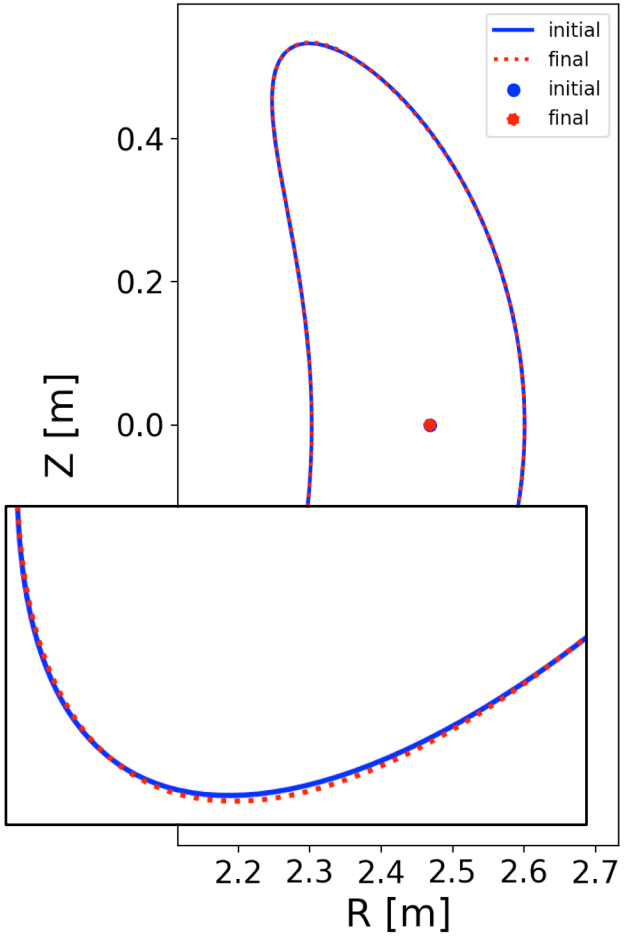
# Now the optimization successfully results in good surfaces

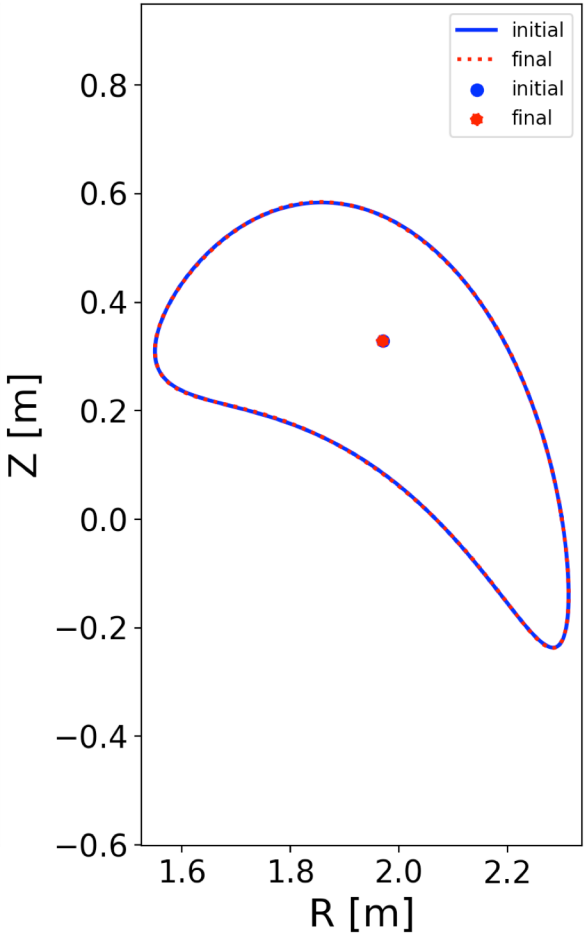


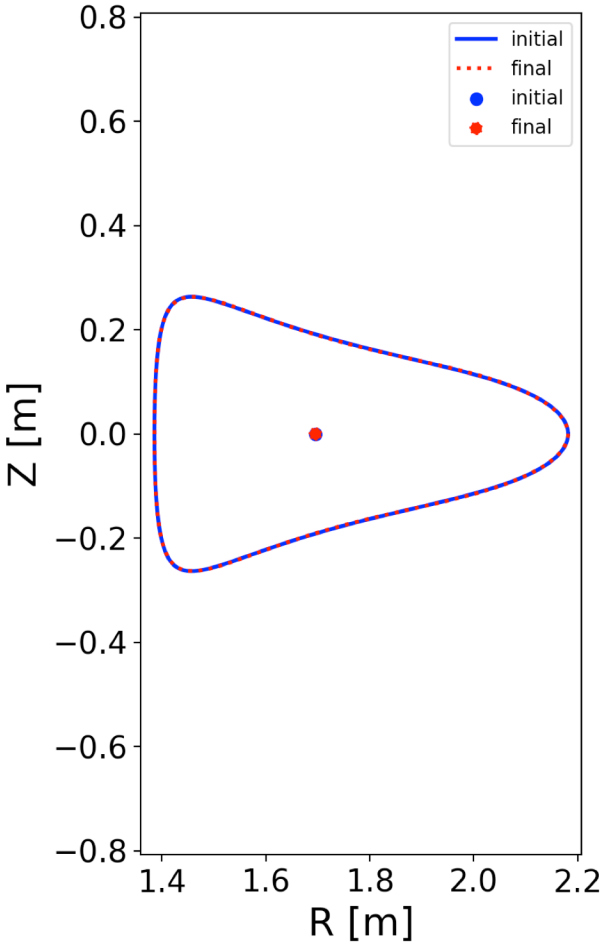



# The change to the surface shapes is extremely small



# Outline

- Simsopt and interface to spec
- Benchmark: stellopt vs simsopt+vmec vs simsopt+spec.
- Optimization of Greene's residues in WISTELL-A.
- Next steps & how you can get involved.

# Next steps

- Direct SPEC-python wrapper started by Caoxiang: help needed!
  - Move allocations out of xspech.f90
  - Move getarg() out of readin()
  - MPI_COMM_WORLD -> MPI_COMM_SPEC
  - Etc. (github issue #119)
- MPI
- Optimize in space of $B_{normal}$ on control surface (Henneberg et al) or coil shapes.
- >1 SPEC volume
- Combined SPEC+VMEC optimization
- Boozer coordinate transformation directly from SPEC

```
from simsopt import Vmec, Spec, LeastSquaresProblem, least_squares_serial_solve

# Initialize VMEC and SPEC:
vmec = Vmec("input.2DOF_vmecOnly_targetIotaAndVolume")
spec = Spec("Spec2DOF_targetIotaAndVolume.sp")

# Set the boundaries to be the same object:
spec.boundary = vmec.boundary

# Define the parameter space:
vmec.boundary.all_fixed()
vmec.boundary.set_fixed("rc(1,1)", False)
vmec.boundary.set_fixed("zs(1,1)", False)

# Define the objective function:
desired_volume = 0.15
desired_iota = 0.41
volume_weight = 1
iota_weight = 1
prob = LeastSquaresProblem([(spec.volume, desired_volume, volume_weight),
                            (vmec.iota, desired_iota, iota_weight)])

# Solve the minimization problem:
least_squares_serial_solve(prob)
```

# Possibility: Transformation to Boozer coordinates on SPEC interfaces

Re-write of BOOZ_XFORM in C++/python, with testing, continuous integration, able to accept equilibria other than VMEC.

https://github.com/hiddenSymmetries/booz_xform

```
pip install booz_xform
```

Quantities needed from SPEC to get $B\left(\theta_{Booz},\ \zeta_{Booz}\right)$ on a surface:

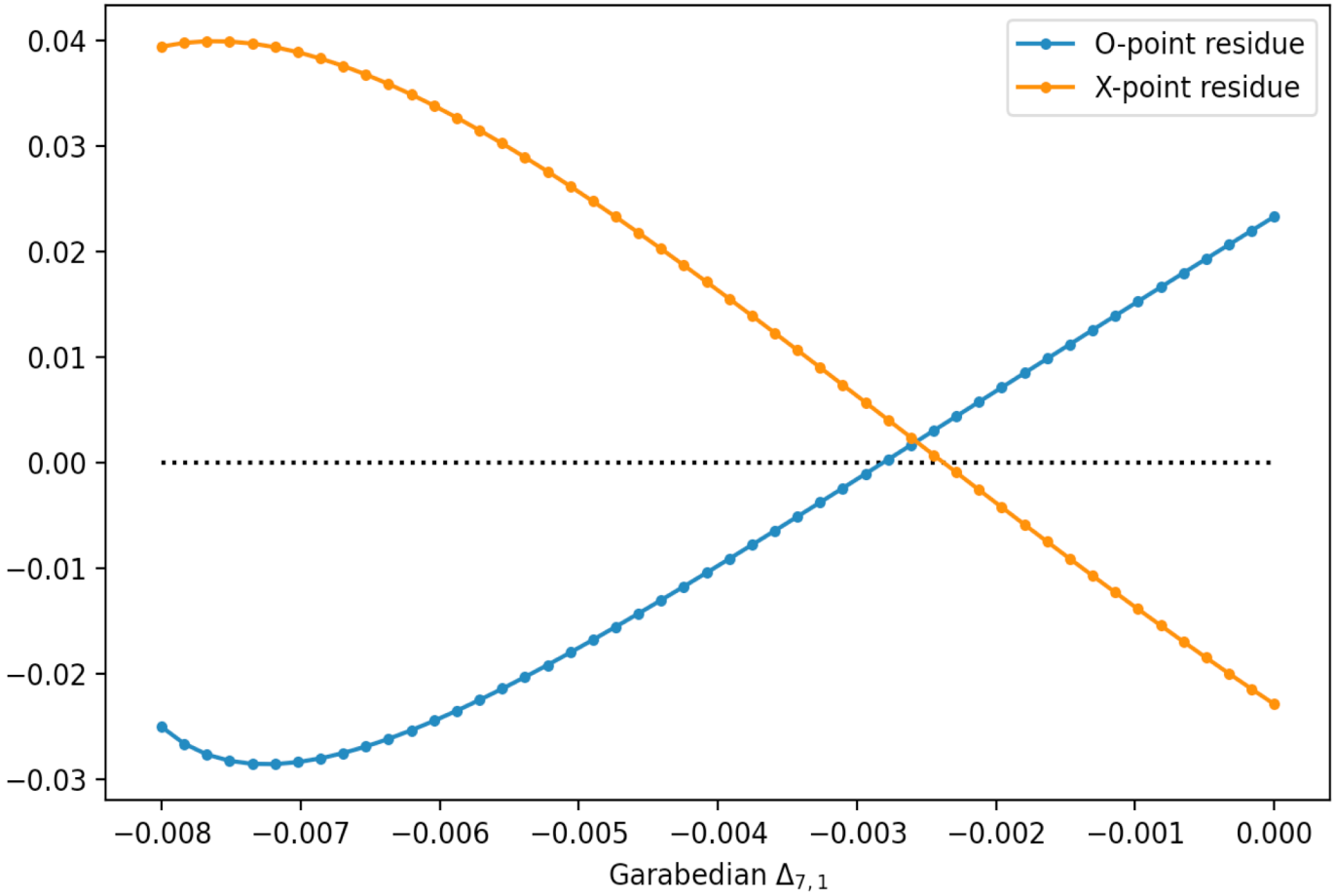
$$B_{\theta_{SPEC}} = \mathbf{B}\cdot\frac{\partial \mathbf{r}}{\partial \theta_{SPEC}},\quad B_{\zeta_{SPEC}} = \mathbf{B}\cdot\frac{\partial \mathbf{r}}{\partial \zeta_{SPEC}},\quad \lambda\left(\theta_{SPEC},\ \zeta_{SPEC}\right),\quad \iota$$

# Your help is needed!

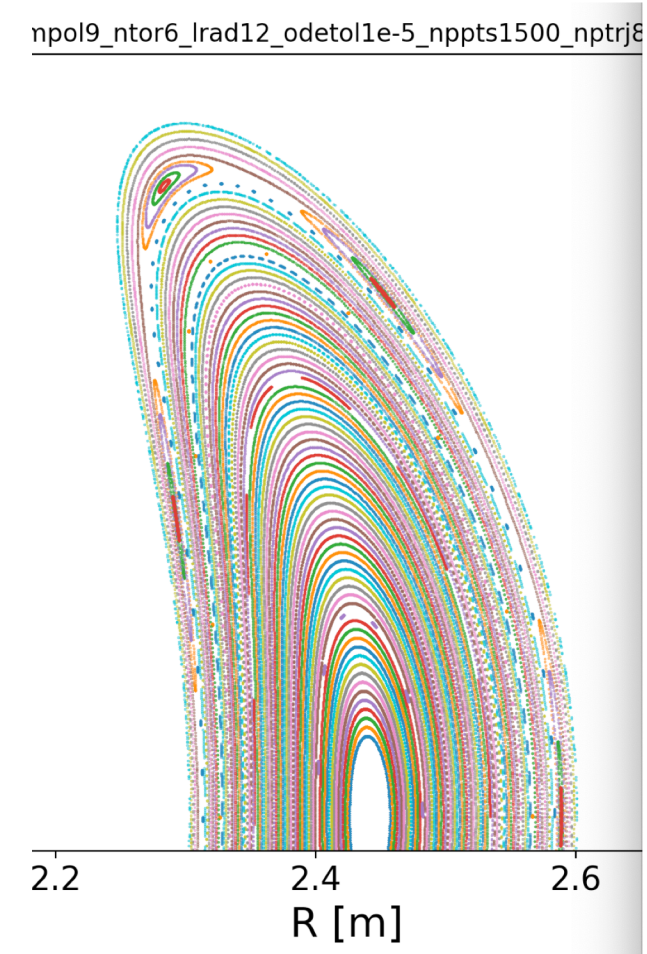
- Tweaks to SPEC needed to complete SPEC-python interface
- Think up more optimization problems & scenarios:
  - What parameter space? What objective function? What geometry?
- Help with SPEC - BOOZ_XFORM interface
- Save iota & oita on surfaces and magnetic axis
- Play around with simsopt
- Everyone is welcome at development meetings: Mondays @ 9am NY, 3pm Europe
- simsopt slack workspace, #developers channel

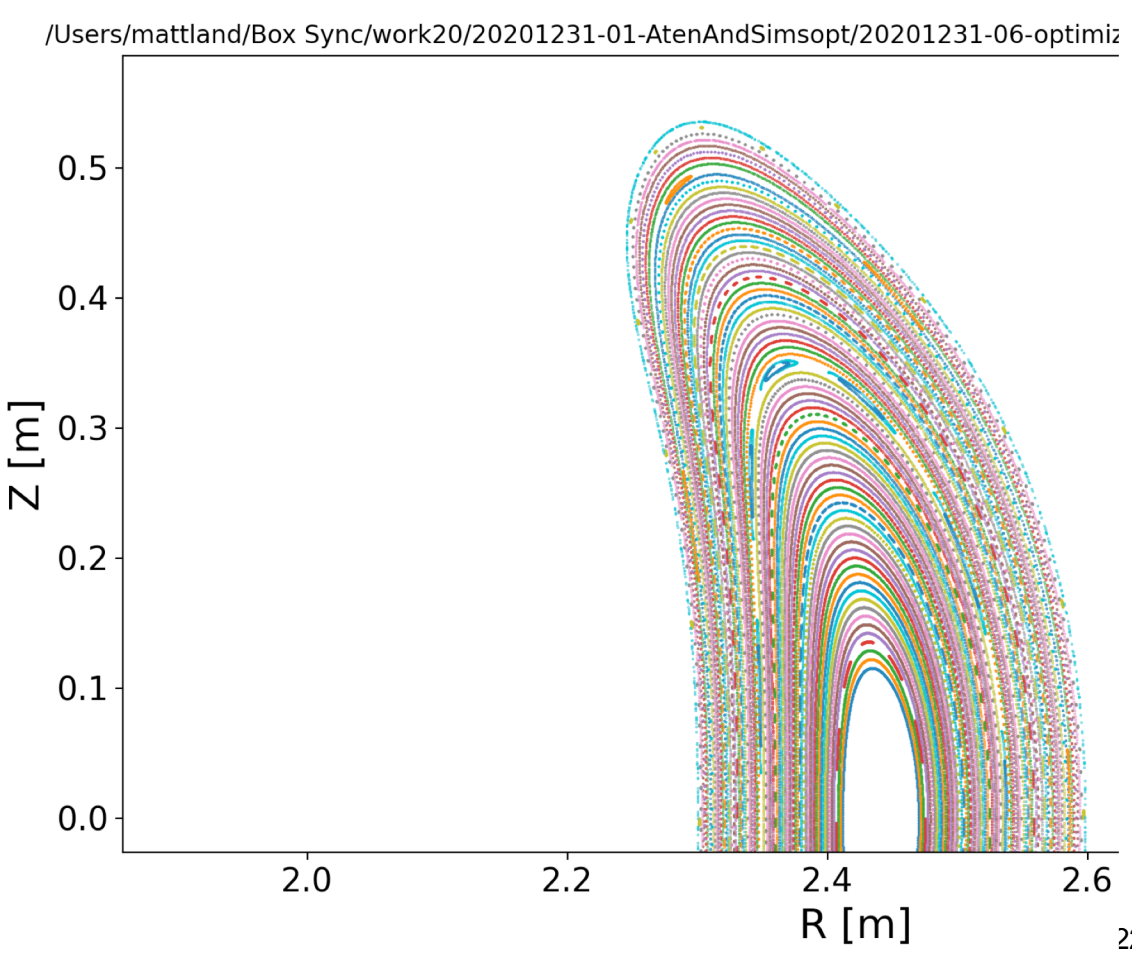
# Extra slides

# Residues are smooth functions of the boundary shape



# Targeting only iota=8/7 residue



```
Initial residues: 0.0233153446531 -0.02287635525375 -8.65246263890e-05 -9.1902208954e-08

2 DOFs:
Final residues:   2.855453140e-06  2.6668174004e-06  3.71536407903e-05 -4.3768913271e-08

64 DOFs:
final residues:  -1.786060961e-07  2.2653632614e-07  3.51474072068e-10 -9.5618735151e-10
```

# Targeting just the O-point residue also works fine

Targeting both X- and O-point residues on the left. Targeting only the O-point residue on the right. For both cases, only the iota=-8/7 resonance is considered, not the iota=-12/11 resonance.