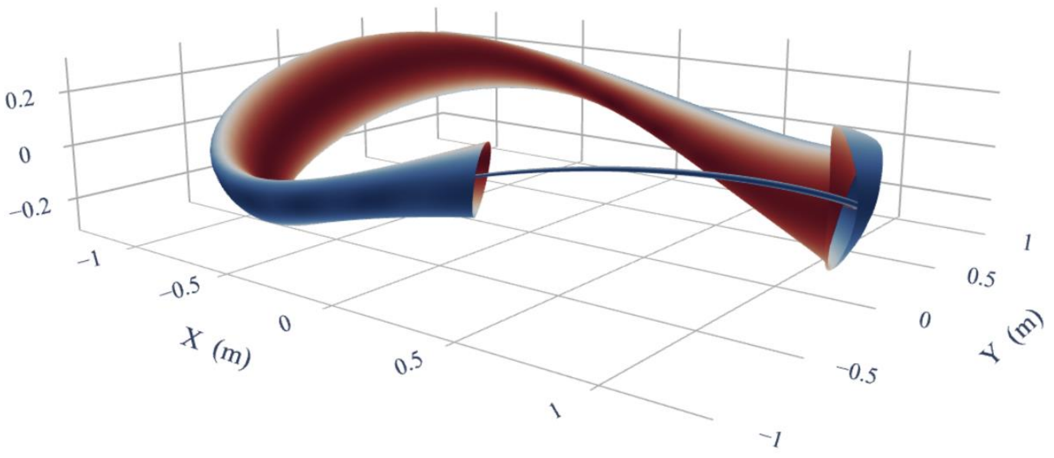
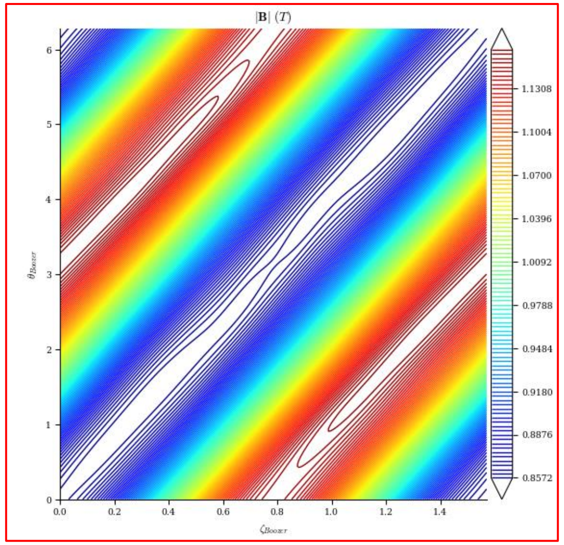
Stellarator Equilibrium and Optimization with DESC

Dario Panici, Daniel Dudt, Rory Conlin, Patrick Kim, Rahul Gaur, Yigit Gunsur Elmacioglu, Kian Orr, Eduardo Rodriguez, Egemen Kolemen

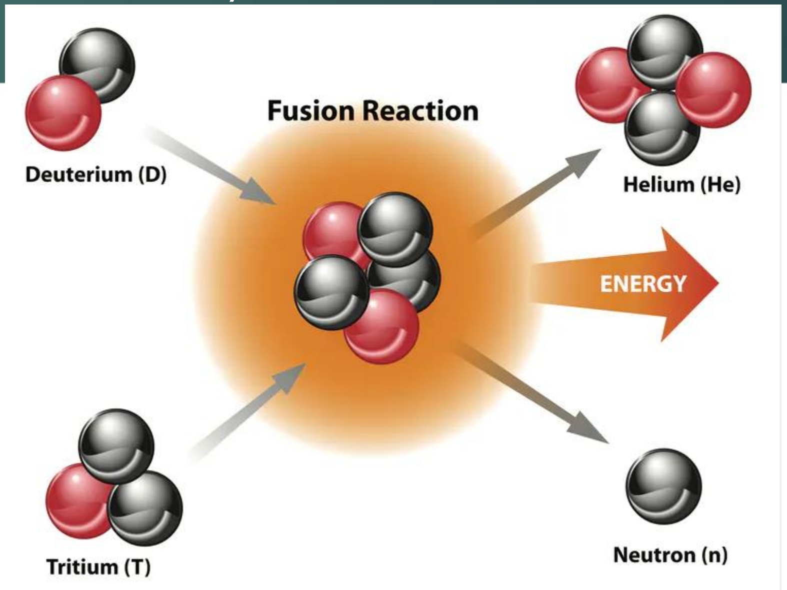




Plasma Control

Nuclear Fusion Energy

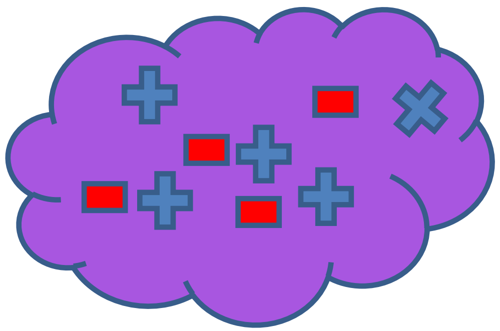
https://theconversation.com/nuclear-fusion-building-a-star-on-earth-is-hard-which-is-why-we-need-better-materials-155917

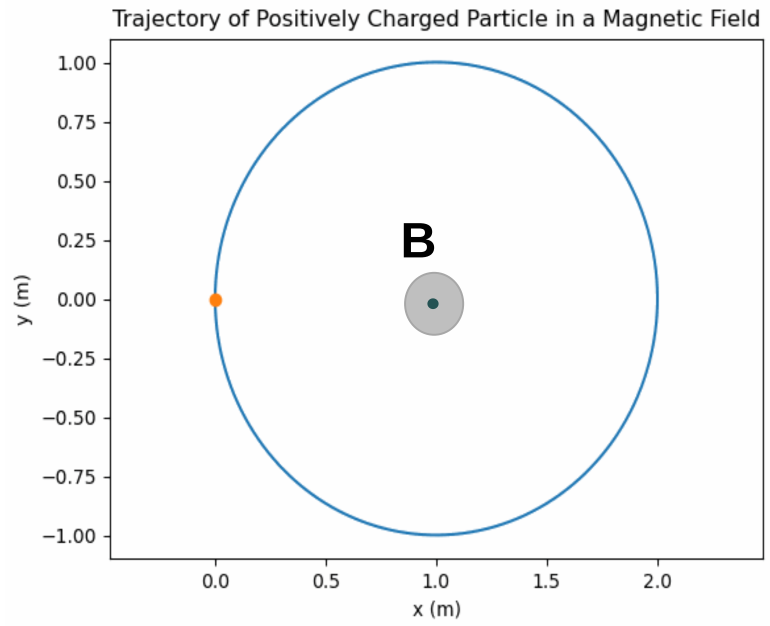


- **Carbon-free energy source**
- **Fusion reaction requires extremely high temperatures ($\mathcal{O}$(10 keV) = $\mathcal{O}$(1E8 °C))**
- **How to hold the hot fusion plasma in place?**

Magnetic Confinement

Plasma



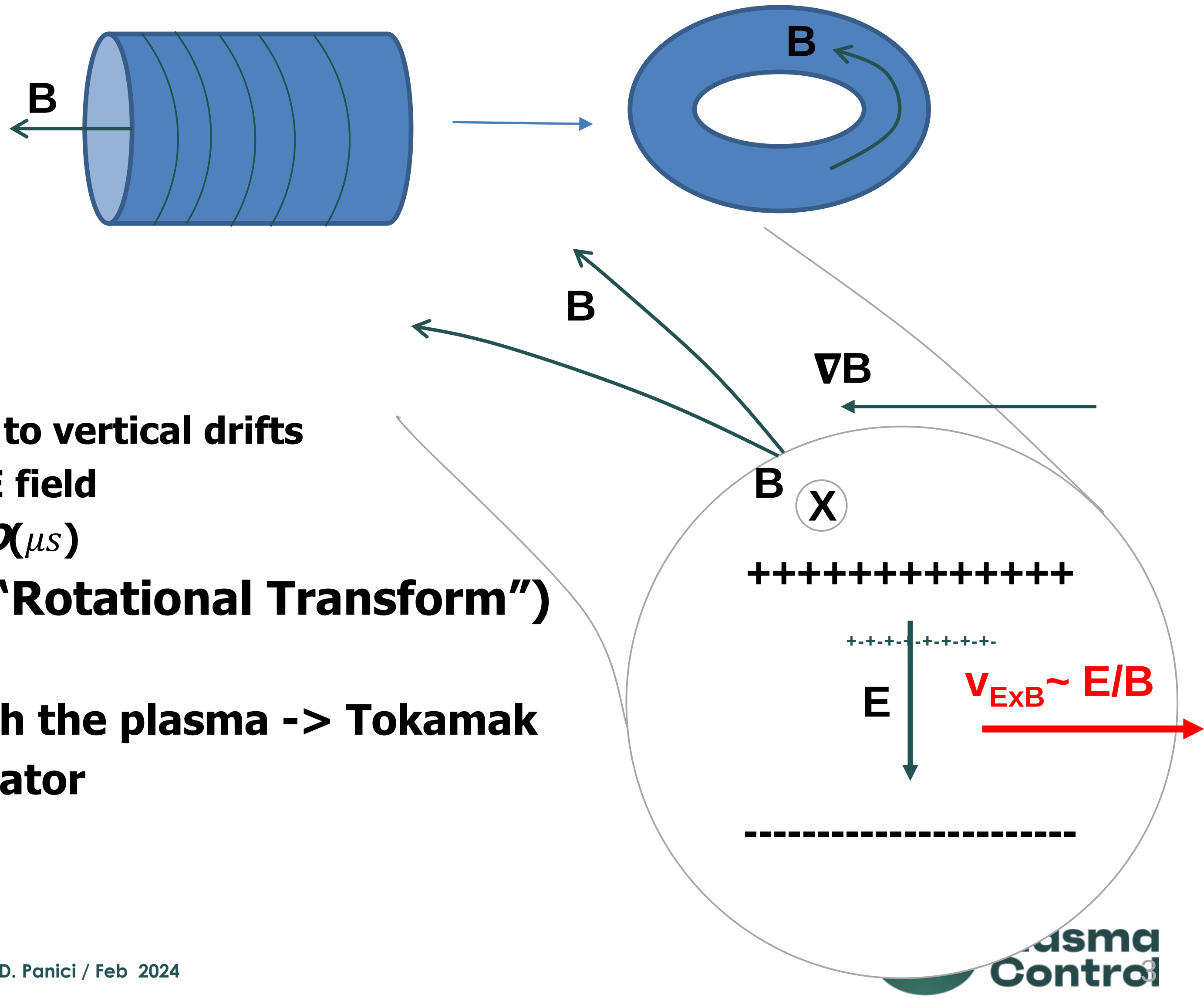


Magnetic Confinement Geometry

$$F = qv \times B$$

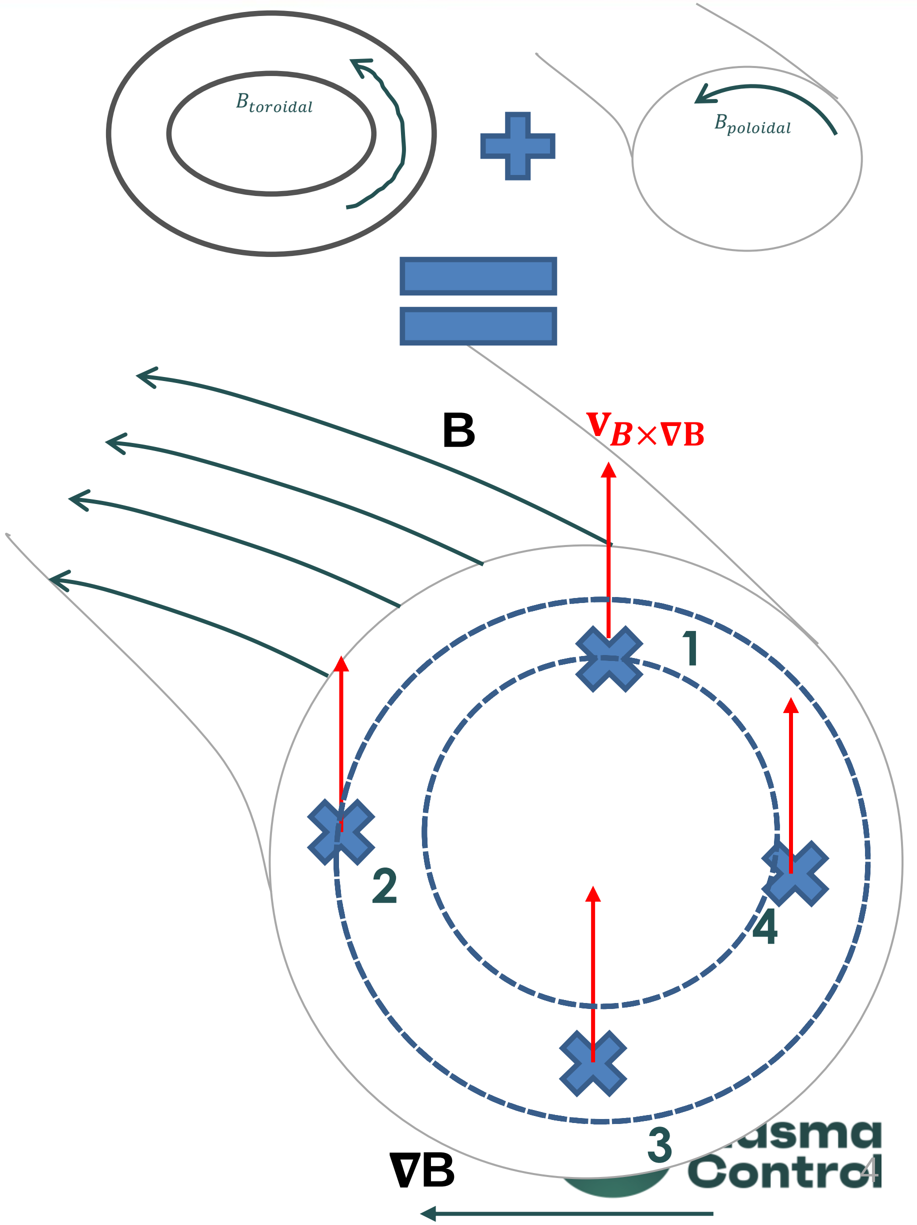
$$v_{drift} = \frac{(F/q) \times B}{B^2}$$

- **Simplest Magnetic field is a solenoid**
  - **Problem: Parallel Confinement**
- **Solution: Plug Ends by 'biting its tail'**
  - **Toroidal confinement**
- **However, confinement still an issue!**
  - **Curvature and gradient introduced in B leads to vertical drifts**
  - **Causes charge separation in $\mathcal{O}(ms)$, leads to E field**
  - **E x B drift causes particles to exit plasma in $\mathcal{O}(\mu s)$**
- **Solution: Make B field twist poloidally ("Rotational Transform")**
  - **How to create this twist?**
    - **By driving a toroidal current through the plasma -> Tokamak**
    - **By twisting external coils -> Stellarator**



Rotational Transform

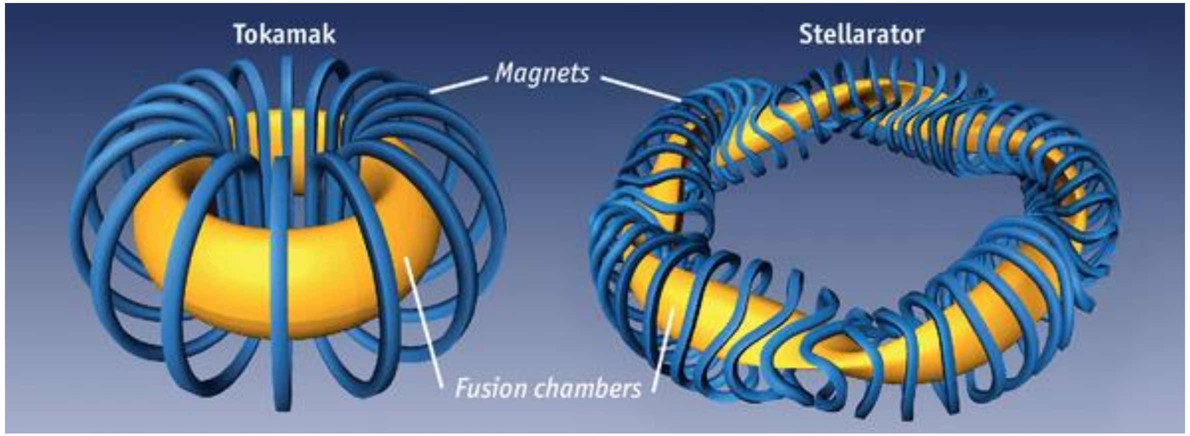
- **Rotational Transform works by moving particle poloidally each time it transits toroidally**
  - **Vertical drift is in same direction, so effectively averages out the vertical drift to zero**
- **Rotational Transform =** $\iota = \frac{\#\ of\ Poloidal\ Turns}{\#\ of\ Toroidal\ Turns}$
- **Rotational Transform means there is a toroidal and a poloidal B field**



Magnetic Confinement Devices: Stellarator vs. Tokamak

Tokamaks	Stellarators

• Axisymmetric
  ○ Simpler geometry
  ○ Guaranteed particle confinement
    ○ Due to Noether's Theorem
• Requires substantial plasma current
  ○ Must be driven
  ○ NOT steady state!
• Source of free energy for instabilities -> disruptions
	• Inherently 3-D
	  ○ Complex geometry and coils
	  ○ Confinement not guaranteed
	    ▪ certain fields exist which recover this (Quasisymmetry)
	  ○ Larger design space
	• Does not need plasma current
	  ○ Steady state
	  ○ No disruptions



https://www.economist.com/science-and-technology/2015/10/24/stellar-work

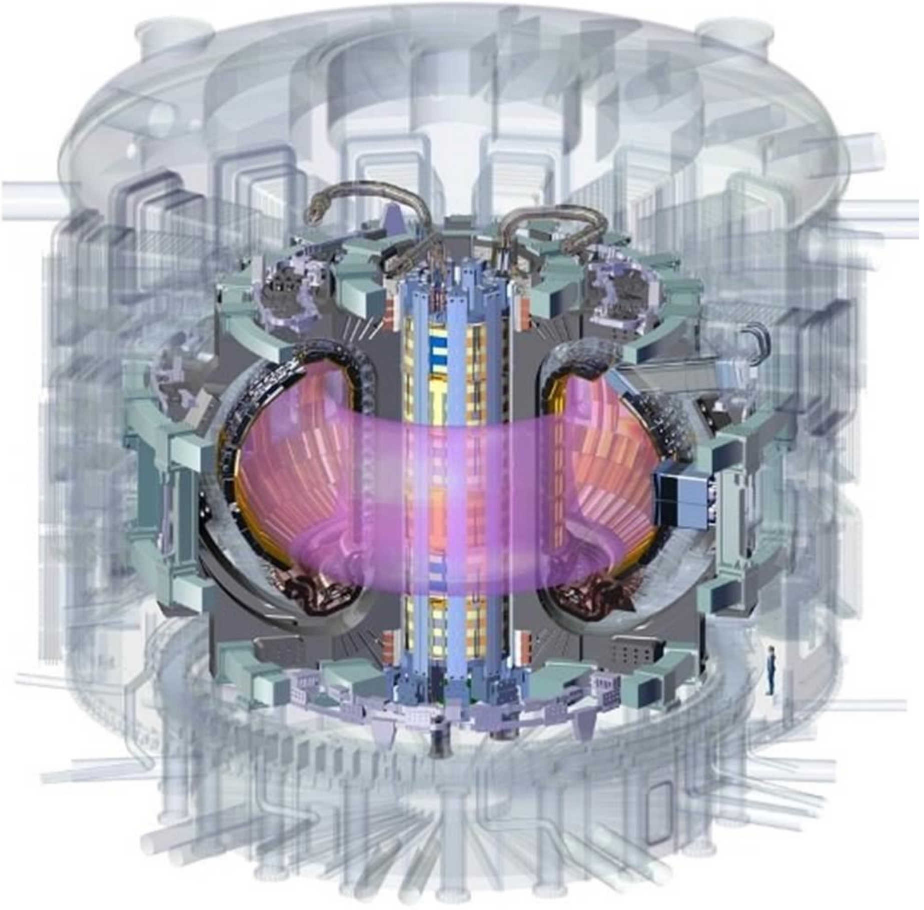
Plasma Control

Plasma Equilibria: What and Why?

Plasma Equilibrium: Configuration of magnetic fields that describes a plasma in steady-state (Ideal MHD)

$$\boldsymbol{F} = \boldsymbol{J} \times \boldsymbol{B} - \nabla p = 0$$

- **Reactor Design and Optimization**
- **Experimental Reconstruction**
- **Necessary for many further plasma physics studies**
  - Particle Transport
  - Stability



https://www.ansys.com/news-center/press-releases/ansys-enables-iter-organization-design-worlds-largest-highly-sustainable-nuclear-fusion-power-plant

Plasma Model – Ideal MHD

Mass:	$\frac{\partial \rho}{\partial \mathrm{t}} + \nabla \cdot (\rho \mathbf{v}) = 0$
Momentum:	$\rho \frac{d\mathbf{v}}{dt} = \mathbf{J} \times \mathbf{B} - \nabla p$
Energy:	$\frac{d}{dt}\left(\frac{p}{\rho^\gamma}\right) = 0$
Ohm's law:	$\mathbf{E} + \mathbf{v} \times \mathbf{B} = 0$
Maxwell:	$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$
	$\nabla \times \mathbf{B} = \mu_0 \mathbf{J}$
	$\nabla \cdot \mathbf{B} = 0$

Freidberg *Ideal MHD* (2014)

ρ	Ion Mass Density
$\boldsymbol{v}$	Ion Flow Velocity
p	Pressure
$\boldsymbol{B}$	Magnetic Field
$\boldsymbol{J}$	Current Density

Plasma Model – Ideal MHD Equilibrium

Mass: $\cancel{\frac{\partial \rho}{\partial t}} + \nabla \cdot (\rho \cancel{\mathbf{v}}) = 0$

Momentum: $\rho \cancel{\frac{d\mathbf{v}}{dt}} = \mathbf{J} \times \mathbf{B} - \nabla p$

Energy: $\cancel{\frac{d}{dt}}\left(\frac{p}{\rho^\gamma}\right) = 0$

Ohm's law: $\mathbf{E} + \cancel{\mathbf{v}} \times \mathbf{B} = 0$

Maxwell: $\nabla \times \mathbf{E} = -\cancel{\frac{\partial \mathbf{B}}{\partial t}}$

$$\nabla \times \mathbf{B} = \mu_0 \mathbf{J}$$
$$\nabla \cdot \mathbf{B} = 0$$

Freidberg *Ideal MHD* (2014)

$\frac{\partial}{\partial t} \rightarrow 0$

$\boldsymbol{v} \rightarrow 0$

$$\boldsymbol{J} \times \boldsymbol{B} = \nabla p$$
$$\nabla \times \boldsymbol{B} = \mu_0 \boldsymbol{J}$$
$$\nabla \cdot \boldsymbol{B} = 0$$

Many Assumptions… But Matches Experiments!

Ideal MHD Equilibrium - What Do the Equations Tell Us?

$$\boldsymbol{J} \times \boldsymbol{B} = \nabla p$$

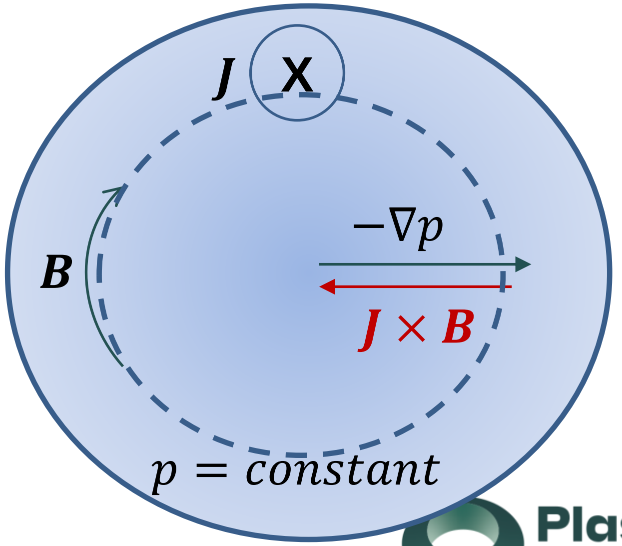
$$\nabla \times \boldsymbol{B} = \mu_0 \boldsymbol{J}$$

$$\nabla \cdot \boldsymbol{B} = 0$$

- **Physically, plasma is in equilibrium with pressure gradient balanced by the J x B force**
- **B, J lie on surfaces of constant pressure**
  - Flux Surfaces $\rightarrow \boldsymbol{B} \cdot \boldsymbol{n} = 0$

$$\boxed{\mathbf{B} \cdot \nabla p = 0} \qquad \boxed{\mathbf{J} \cdot \nabla p = 0}$$

- **Mathematically, is a coupled system of nonlinear PDEs**
- **Goal of Equilibrium Solving: Find the magnetic field B that satisfies these equations, given some BCs and inputs**



1D Example: Z-pinch

Ideal MHD Equilibrium: How it is Solved

- **Axisymmetric cases (i.e. Tokamak) reduce to 2D Grad-Shafranov Equation**

$$\boxed{\Delta^* \psi(R,Z) = R\frac{\partial}{\partial R}\frac{1}{R}\frac{\partial \psi}{\partial R} + \frac{\partial^2 \psi}{\partial Z^2} = -\mu_0 R^2 p'(\psi) - F(\psi)F'(\psi)}$$

- **In fully 3D geometry, no general analytic solution exists -> Equilibria must be found numerically**

$$\boldsymbol{J} \times \boldsymbol{B} = \nabla p$$

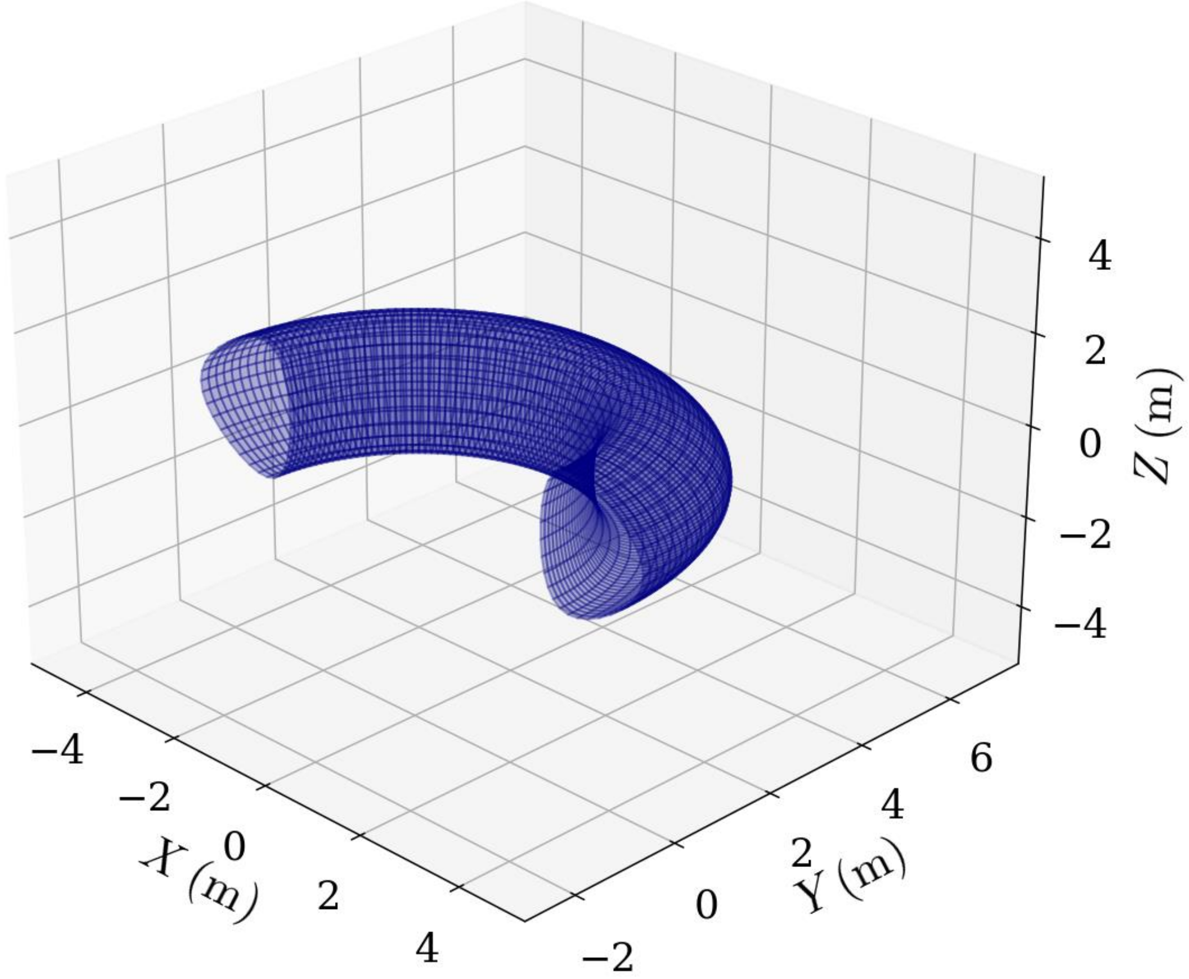
$$\nabla \times \boldsymbol{B} = \mu_0 \boldsymbol{J}$$

$$\nabla \cdot \boldsymbol{B} = 0$$

Flux Surfaces

$$\mathbf{B} \cdot \nabla p = 0$$

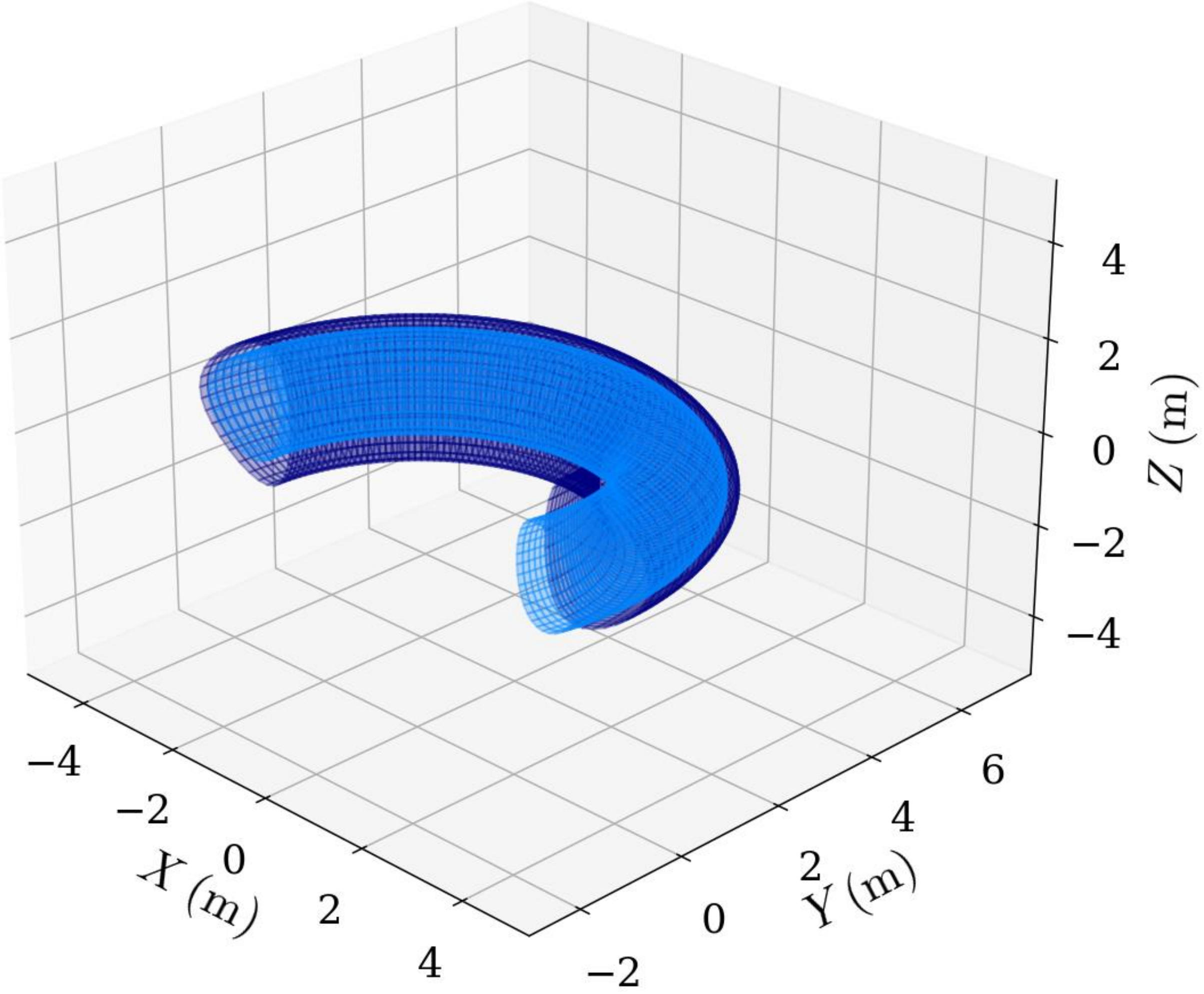
Surfaces in space upon which magnetic field lines lie



Flux Surfaces

$$\mathbf{B} \cdot \nabla p = 0$$

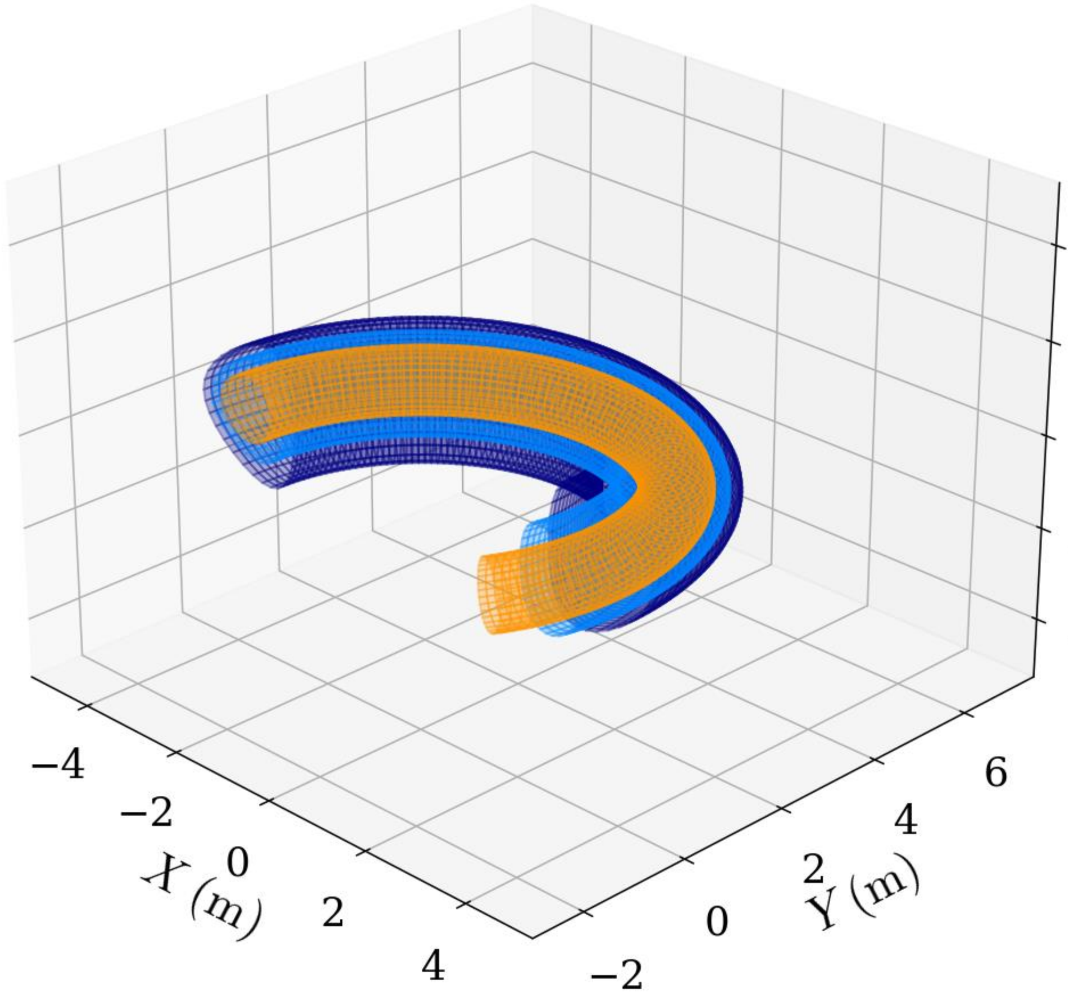
Surfaces in space upon which magnetic field lines lie



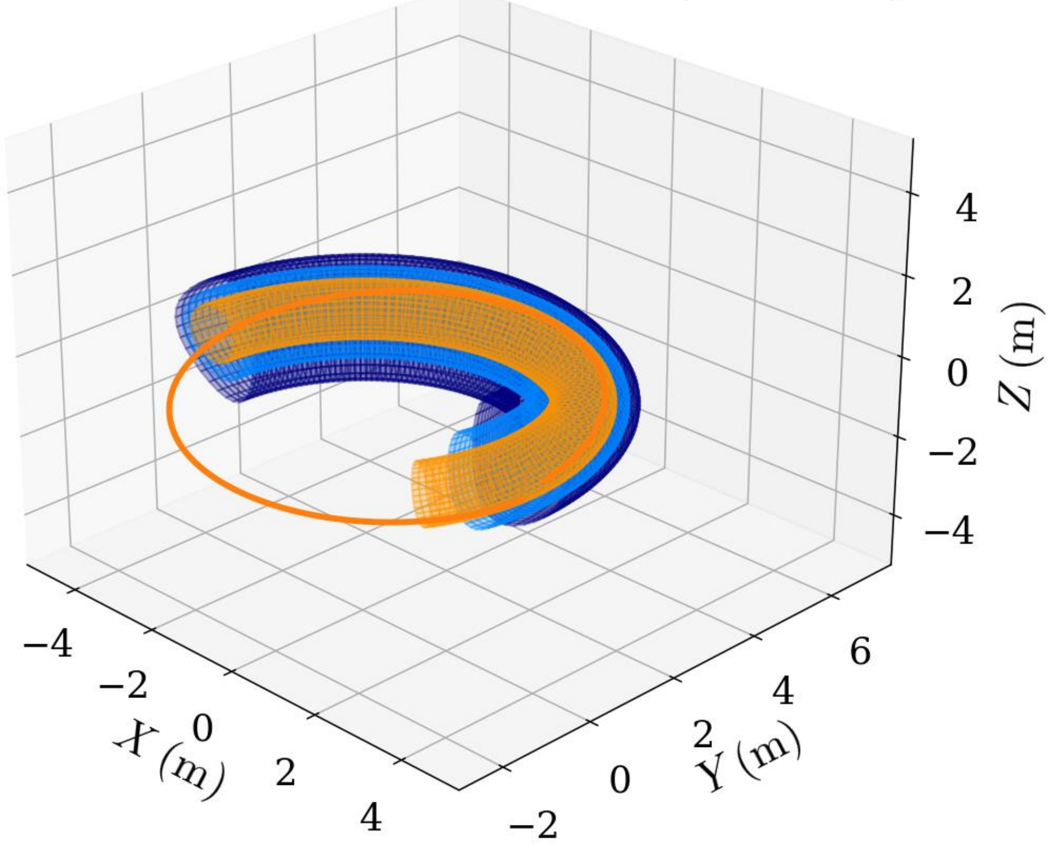
Flux Surfaces

$$\mathbf{B} \cdot \nabla p = 0$$

Surfaces in space upon which magnetic field lines lie



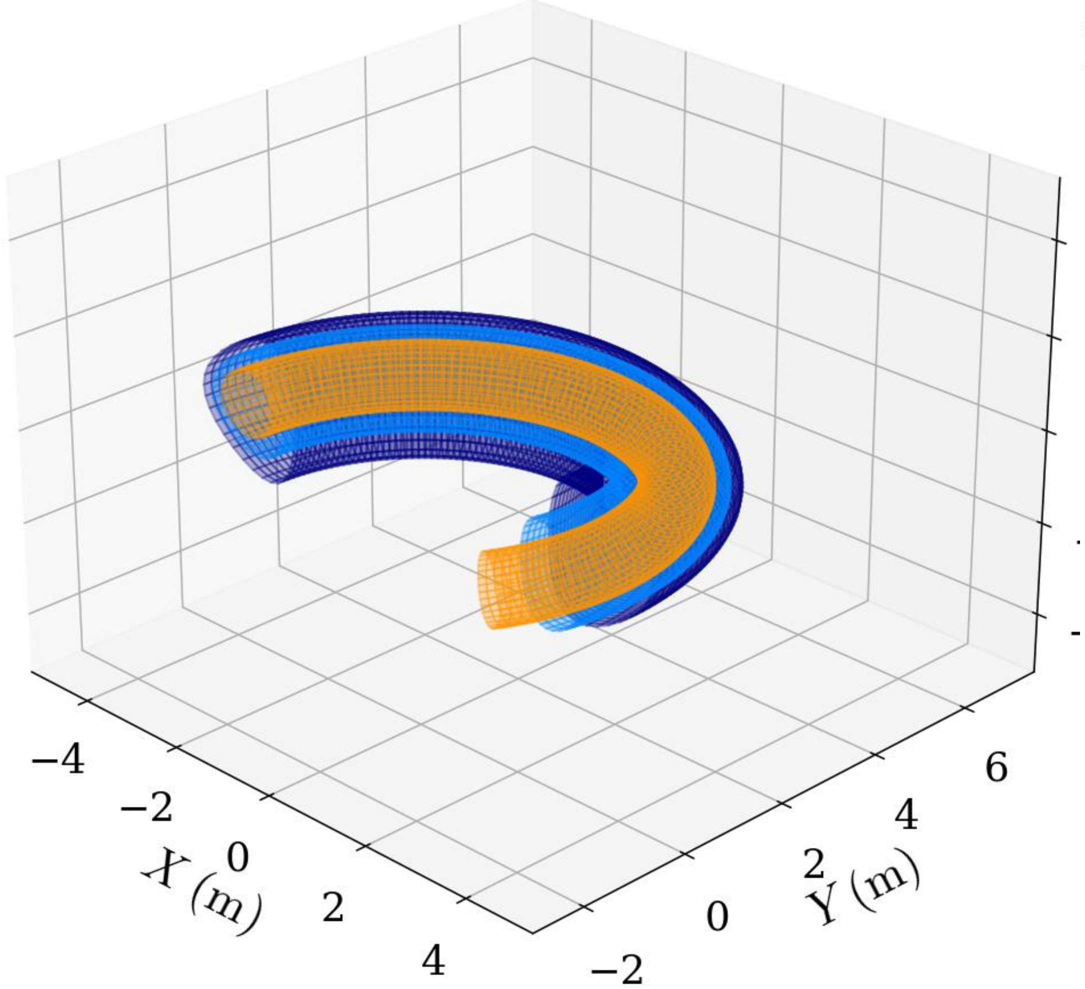
Magnetic field lines trace out flux surfaces



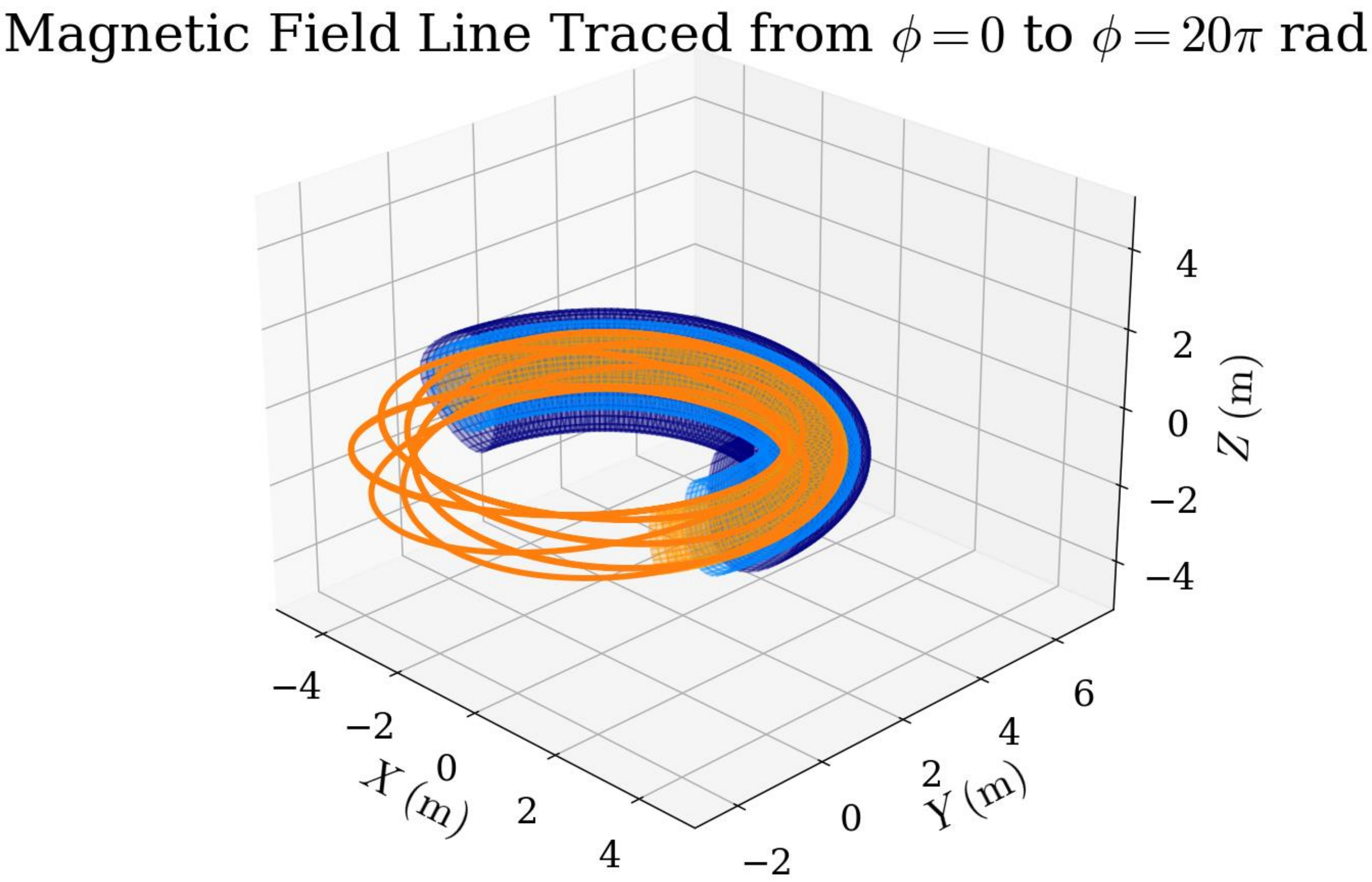
Flux Surfaces

$$\mathbf{B} \cdot \nabla p = 0$$

Surfaces in space upon which magnetic field lines lie



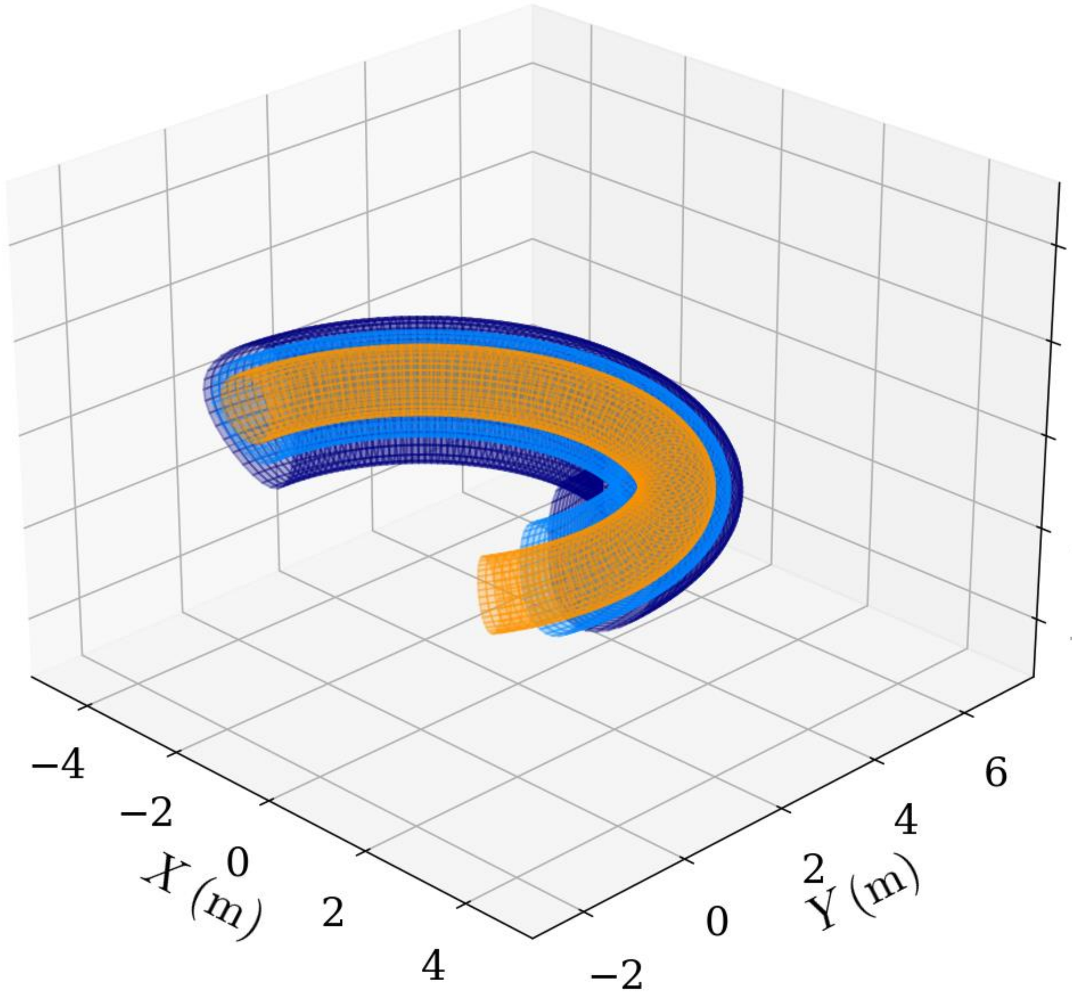
Magnetic field lines trace out flux surfaces



Flux Surfaces

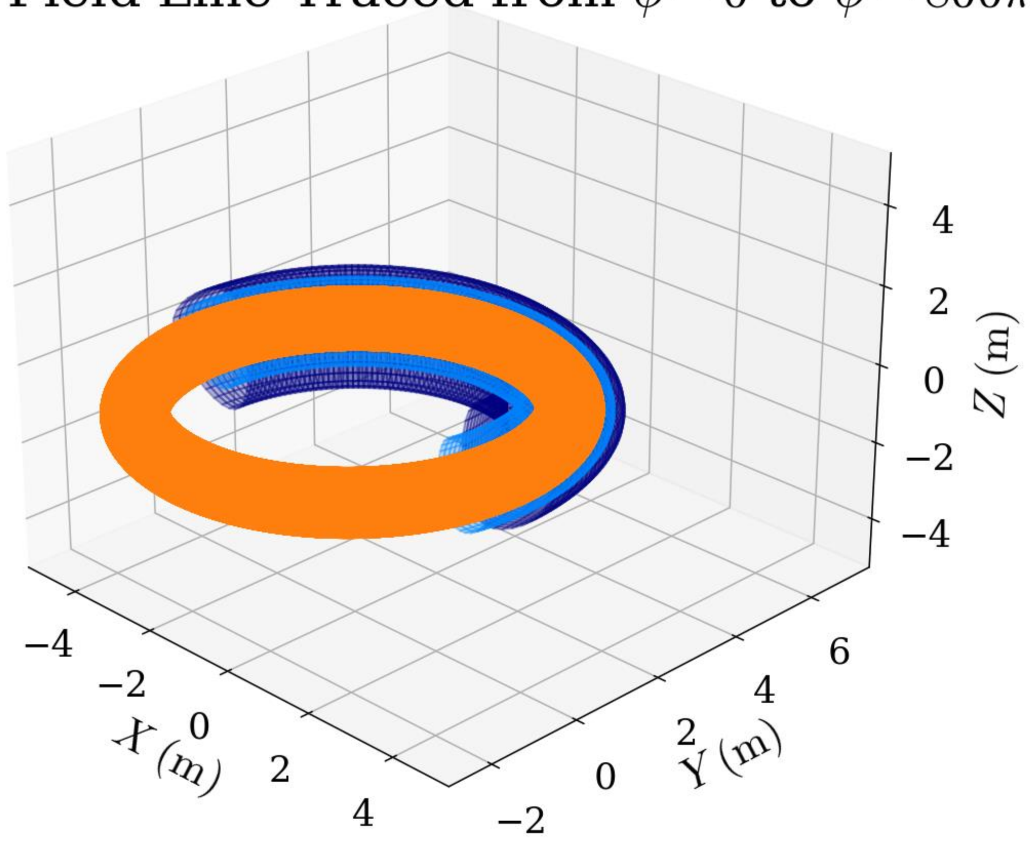
$$\mathbf{B} \cdot \nabla p = 0$$

Surfaces in space upon which magnetic field lines lie



Magnetic field lines trace out flux surfaces

Magnetic Field Line Traced from $\phi = 0$ to $\phi = 800\pi$ rad

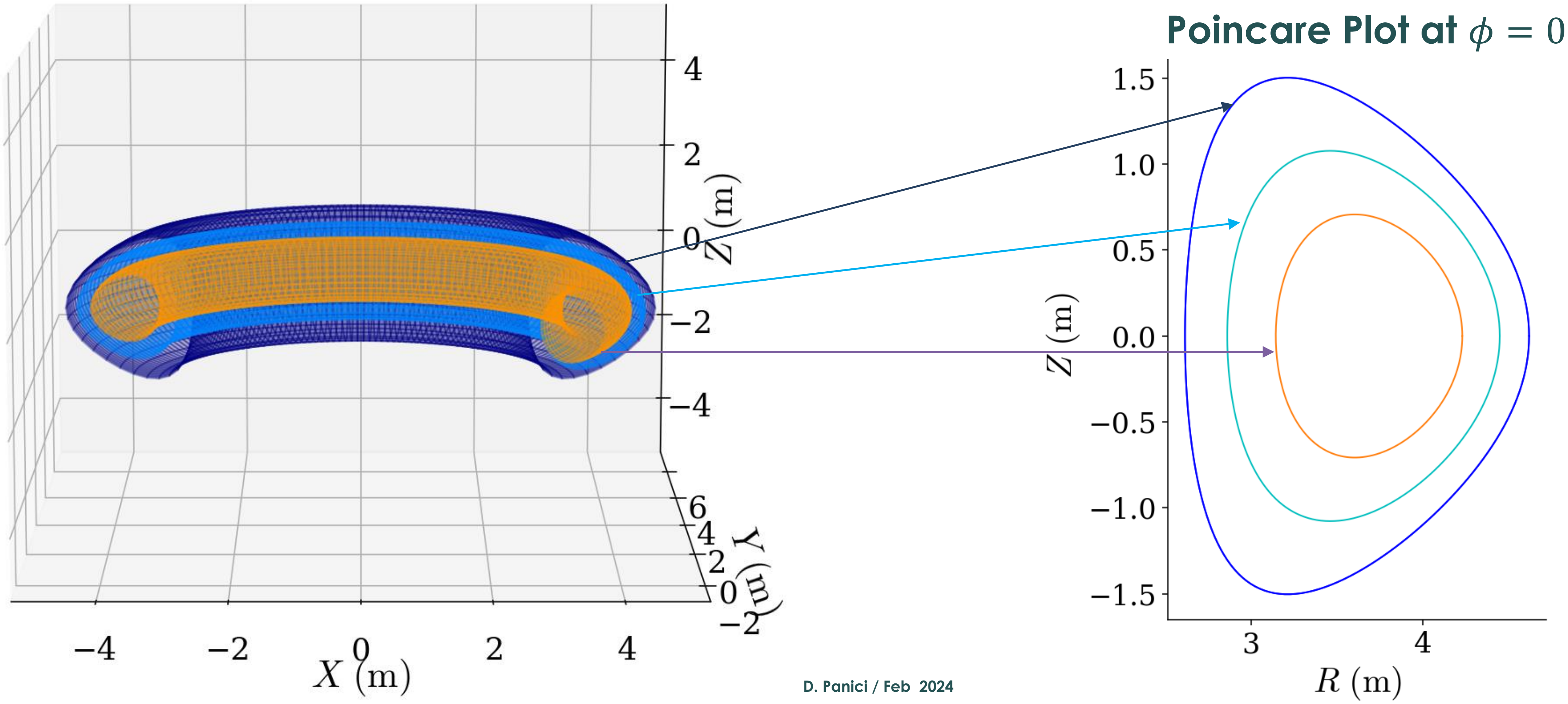


Flux Surfaces

$$\mathbf{B} \cdot \nabla p = 0$$

Surfaces in space upon which magnetic field lines lie

Magnetic field lines trace out flux surfaces



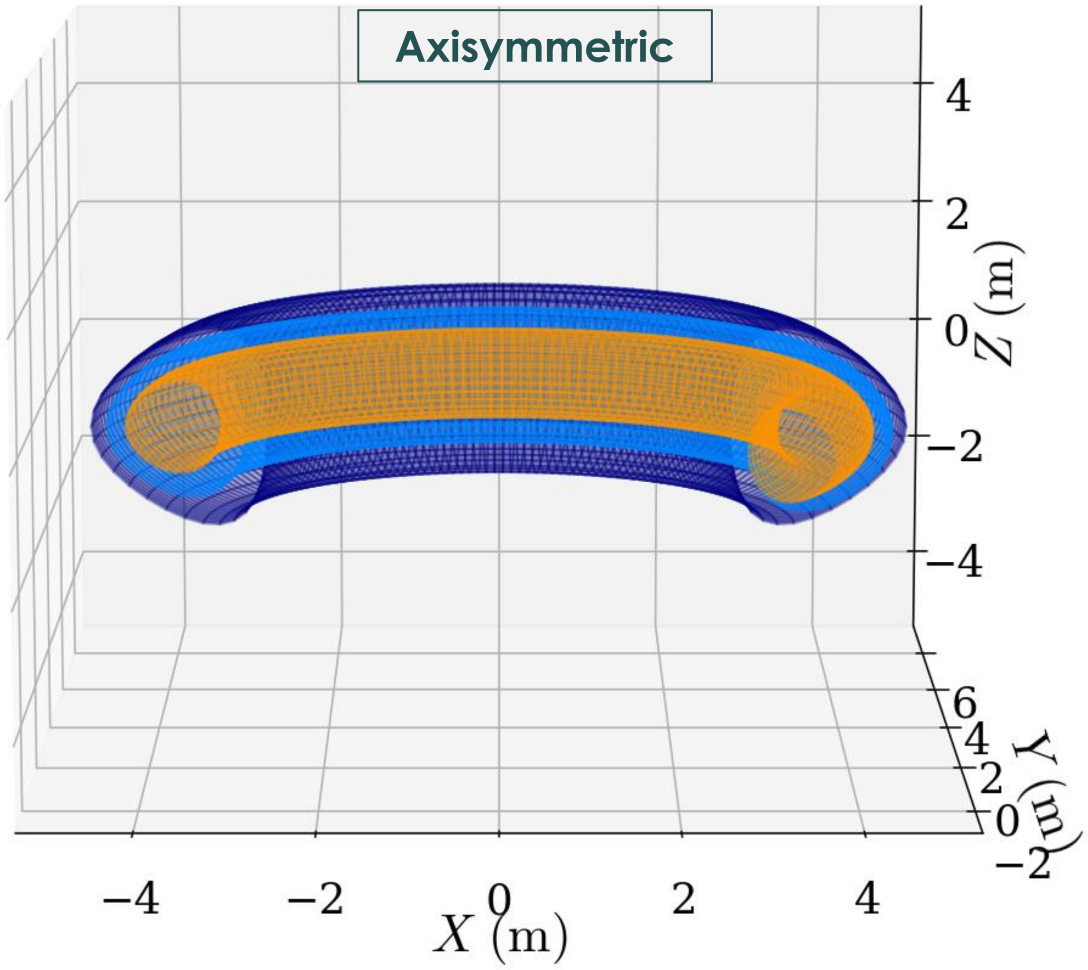
Flux Surfaces

$$\mathbf{B} \cdot \nabla p = 0$$

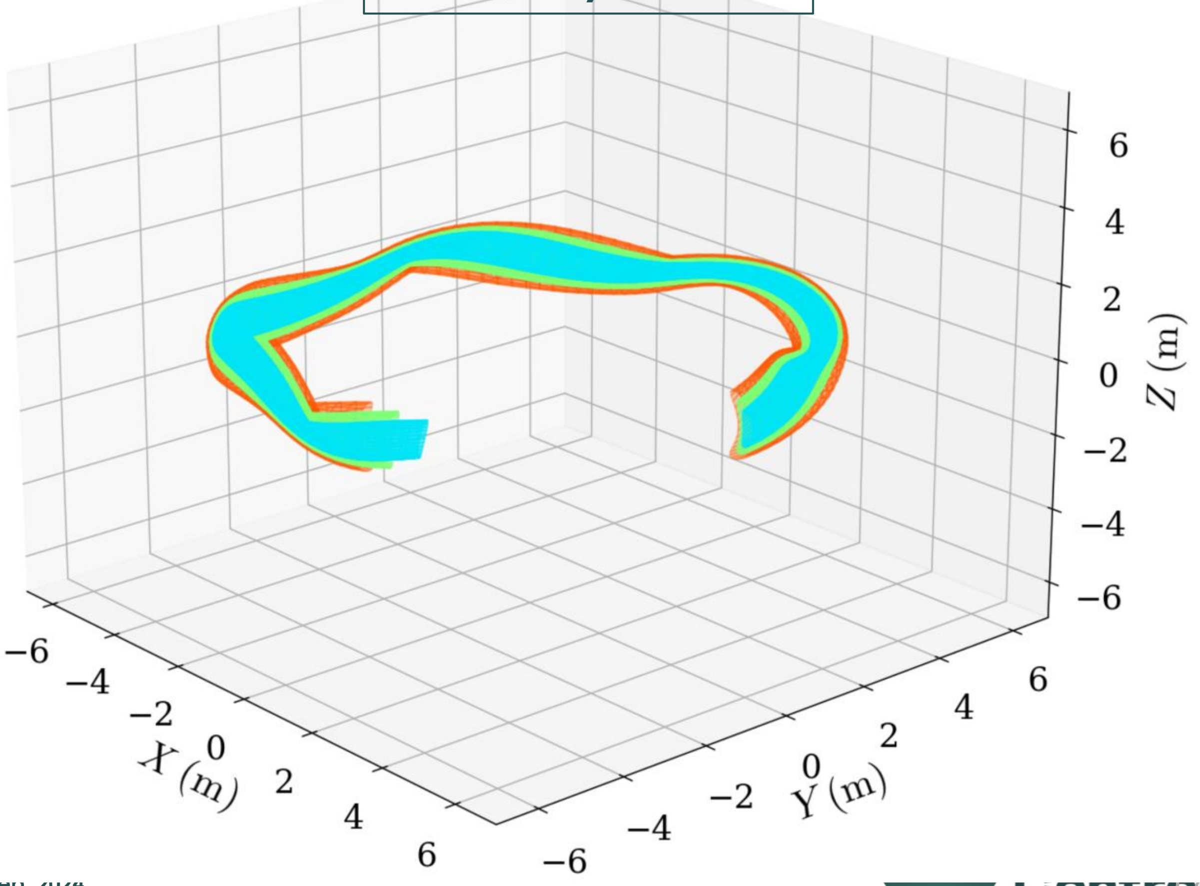
Surfaces in space upon which magnetic field lines lie

In stellarators, the flux surfaces are not axisymmetric (3D spatial dependence)

Axisymmetric



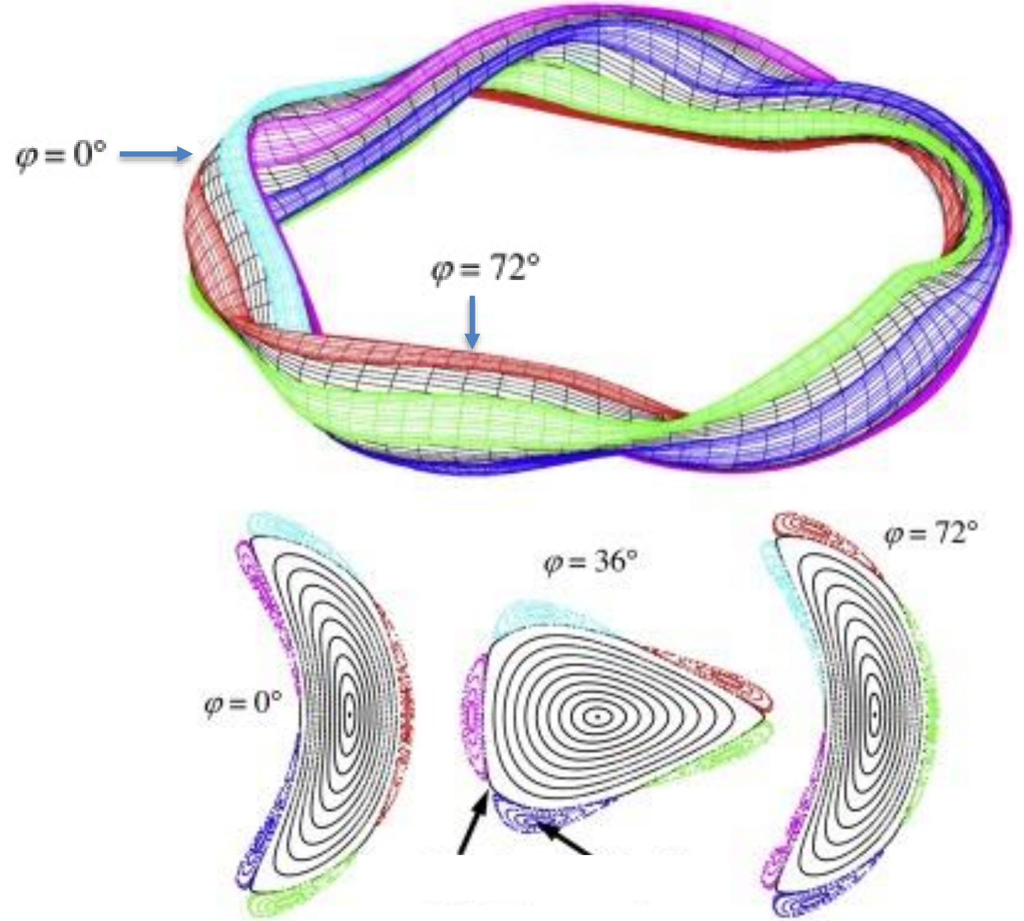
Non-axisymmetric



Discrete Toroidal Symmetry: Field Periods

$\mathbf{B} \cdot \nabla p = 0$

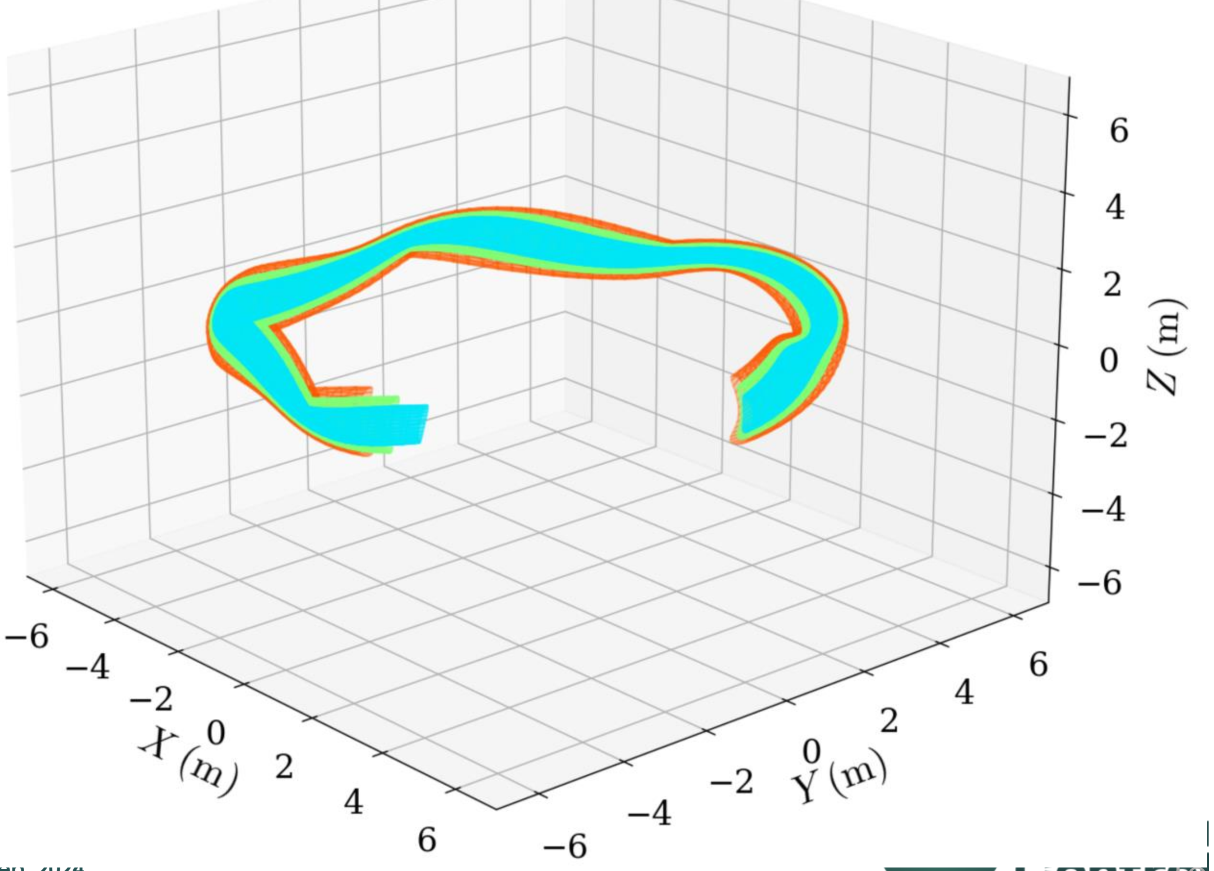
Discrete toroidal symmetry can exist in stellarators : Number of field periods (NFP) is 5 for this stellarator
$X(\phi) \sim sin(nN_{FP}\phi)$



Endler 2021

In stellarators, the flux surfaces are not axisymmetric (3D spatial dependence)

Non-axisymmetric



DESC Stellarator Equilibrium and Optimization Code

D. Panici / Feb 2024

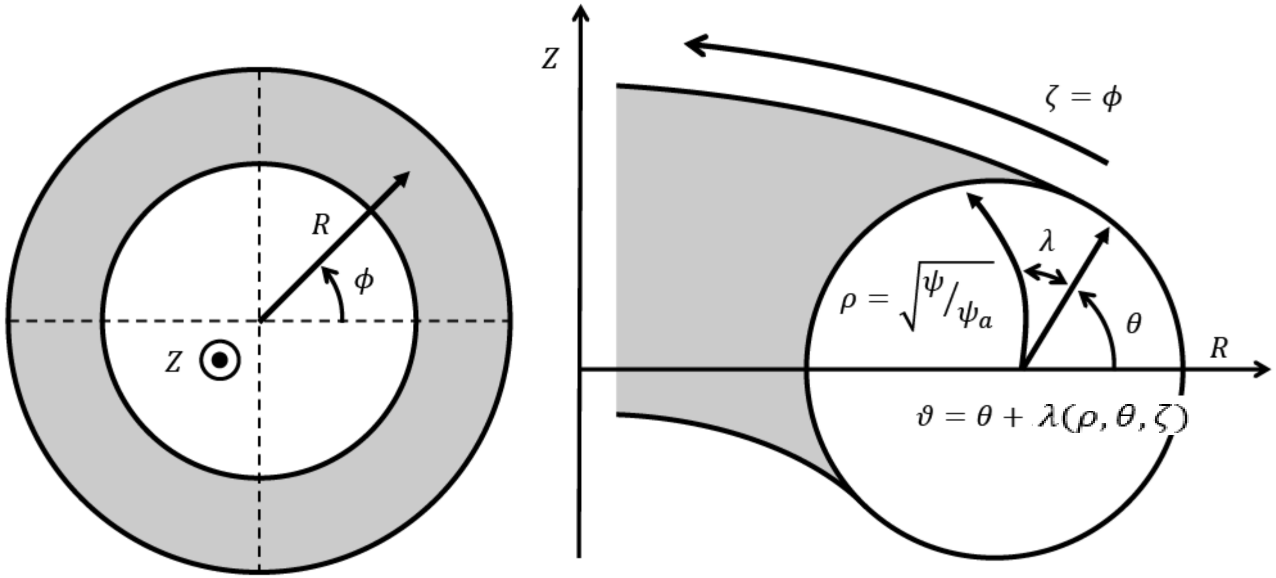
Stellarator Equilibrium and Optimization - DESC

- 3D Ideal MHD Equilibrium Code
- Assumes Nested Flux Surfaces
- Inverse Equilibrium Problem
- Minimizes Force Error Directly

$$\boxed{\boldsymbol{F} = \boldsymbol{J} \times \boldsymbol{B} - \nabla p = 0}$$

- Pseudospectral Code
- Python, AD, GPU-capable

3D Spectral Representation of $\mathbf{x} = (R, \lambda, Z)$ using Fourier-Zernike Basis



(Dudt and Kolemen 2020)

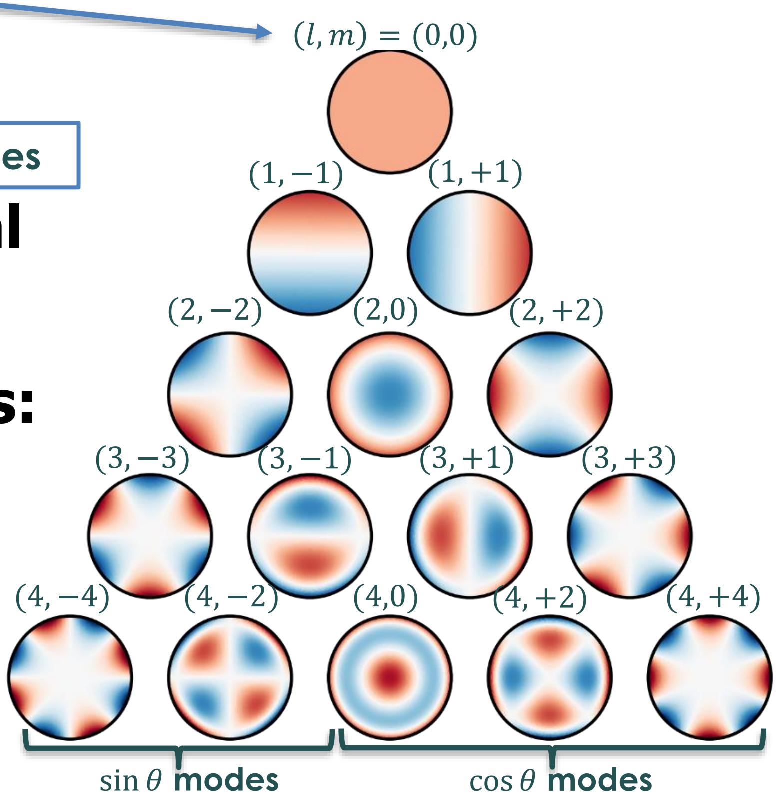
Preliminary: DESC represents solution in Fourier-Zernike Basis

spectral coefficients

Zernike polynomials

$$X(\rho,\theta,\zeta) = \sum_{lmn} X_{lmn} \mathcal{Z}_l^m(\rho,\theta)\mathcal{F}^n(\zeta)$$

Fourier series



- **Periodic boundary conditions for poloidal & toroidal angles**
- **Satisfies analyticity conditions at the magnetic axis:**

$$f(\rho,\theta) = \sum_m \rho^m\left(a_{m,0} + a_{m,2}\rho^2 + \cdots\right)\cos(m\theta) + \sum_m \rho^m\left(b_{m,0} + b_{m,2}\rho^2 + \cdots\right)\sin(m\theta)$$

- **Exponential convergence (if solution exists and is smooth)**

Boyd et al., *J. Comput. Phys.* (2011)
Lewis et al., *J. Math. Phys.* (1990)

DESC was developed from scratch with healthy coding practices in mind

- **Open source Python3 code repository**

  `https://github.com/PlasmaControl/DESC`

- **Well documented, both in the code and external documentation**
- **Continuous Integration to test new code** docs passing | Unit tests passing | Regression tests passing | codecov 95%
- **Modular structure enables custom applications and facilitates adding new capabilities**
- **Easy to install and start using** `pip install desc-opt`
- **Growing user + developer base around the world**

Uses JAX for automatic differentiation and JIT compilation

- **JAX is developed by Google, using the same backend as TensorFlow**
- **Automatic differentiation provides exact derivatives of arbitrary order**

Jacobian matrix required for Newton method: $x_{n+1} = x_n - \left(\frac{\partial f}{\partial x}\right)^{-1} f(x)$

  - **yields derivative with SINGLE function call (no need for finite differences or manually writing out analytic derivatives)**
- **Just-in-time (JIT) compilation improves speed and memory usage using Accelerated Linear Algebra (XLA)**
- **Runs on both CPU & GPU**
- **Easy to implement**

```
import jax.numpy as jnp
```

https://github.com/google/jax

The Stellarator Equilibrium Problem

Equilibrium

Optimization Algorithm

DESC

Gradient Information

Constraints

- Fixed-boundary surface
- Pressure profile
- Current/rotational transform
- Total toroidal magnetic flux

Objectives

- Ideal MHD force balance
- Energy

DESC Equilibrium Solving Algorithm

Initialization

Inputs
$R_b(\rho = 1, \theta, \zeta)$,
$Z_b(\rho = 1, \theta, \zeta)$,
$p(\rho), \iota(\rho), \psi_a$

Fourier Series
$R_{b,mn}, Z_{b,mn}$

Scale Boundary as Initial Guess for Surface Geometry
$R_{mn}(\rho) \sim \rho R_{b,mn}$
$Z_{mn}(\rho) \sim \rho Z_{b,mn}$

$$R(\rho, \theta, \zeta) = \sum_{m=-M,n=-N,l=0}^{M,N,L} R_{lmn} \mathcal{Z}_l^m(\rho, \theta) \mathcal{F}^n(\zeta)$$

$$\lambda(\rho, \theta, \zeta) = \sum_{m=-M,n=-N,l=0}^{M,N,L} \lambda_{lmn} \mathcal{Z}_l^m(\rho, \theta) \mathcal{F}^n(\zeta)$$

$$Z(\rho, \theta, \zeta) = \sum_{m=-M,n=-N,l=0}^{M,N,L} Z_{lmn} \mathcal{Z}_l^m(\rho, \theta) \mathcal{F}^n(\zeta)$$

Main Algorithm

Repeat until Desired Resolution

Compute B and J from
$R(\rho, \theta, \zeta), Z(\rho, \theta, \zeta), \lambda(\rho, \theta, \zeta)$

Compute F_ρ, F_β on collocation grid

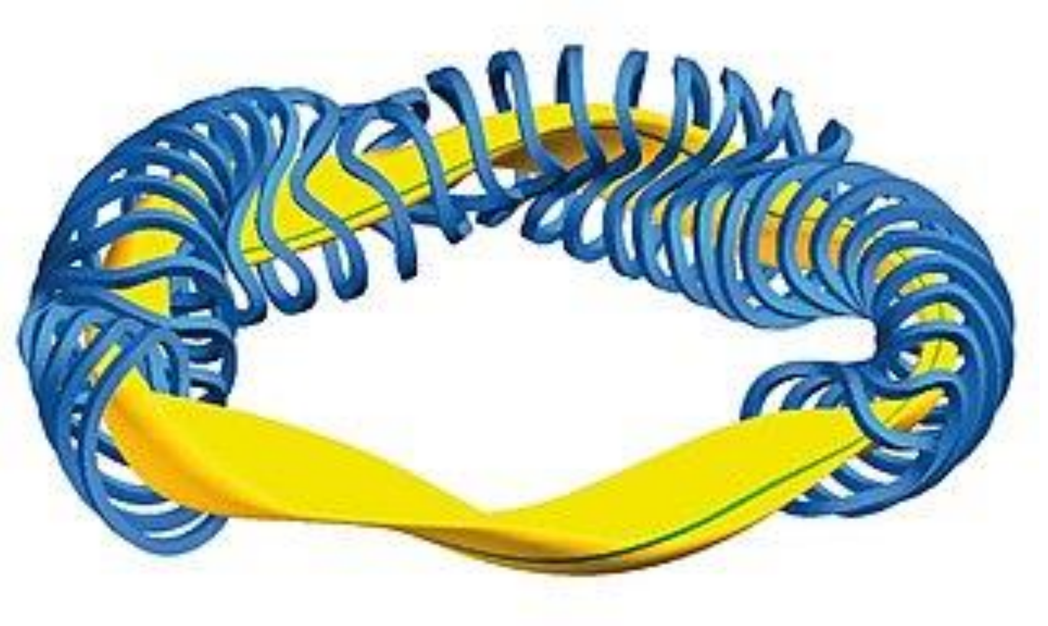
$\boldsymbol{f}(\boldsymbol{x}) = 0$, use Newton (or least-squares) Method to find $\Delta \boldsymbol{x}$

Repeat until converged

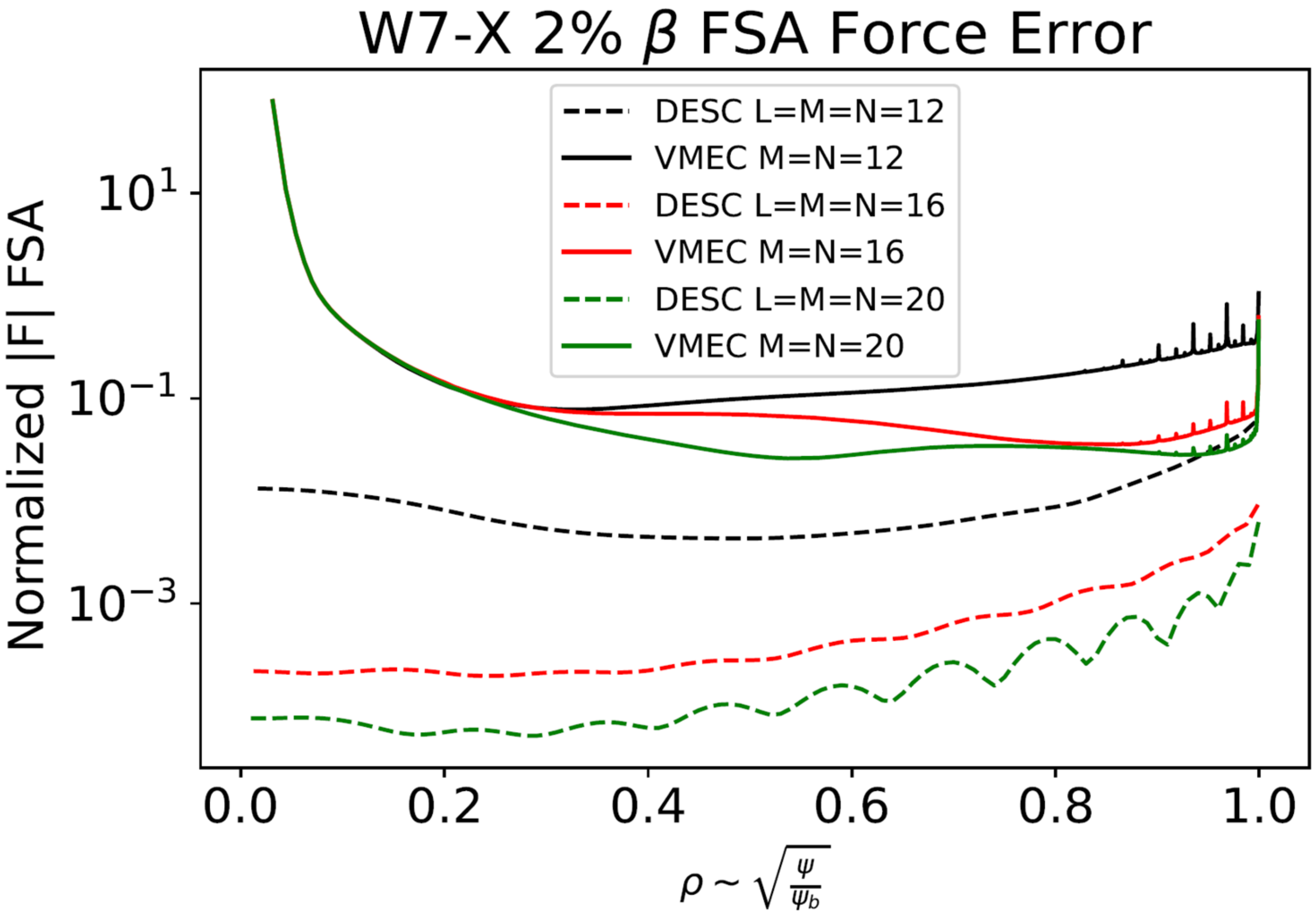
Increase Collocation Grid and/or Spectral Resolution

$\boldsymbol{x} = [R_{lmn}, Z_{lmn}, \lambda_{lmn}]$

DESC spectral methods yield more accurate equilibrium solutions

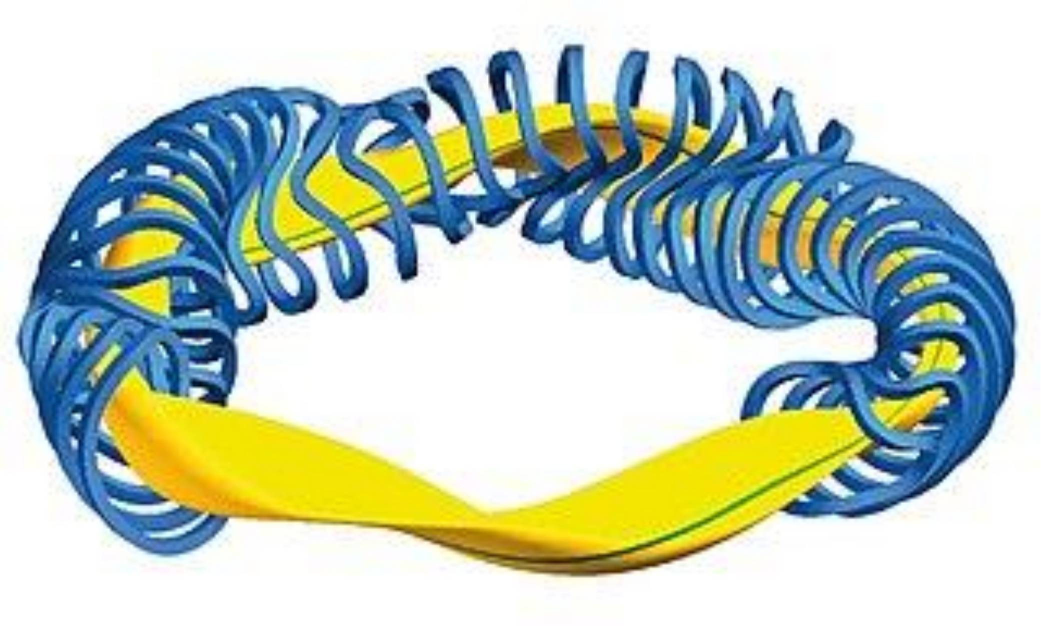


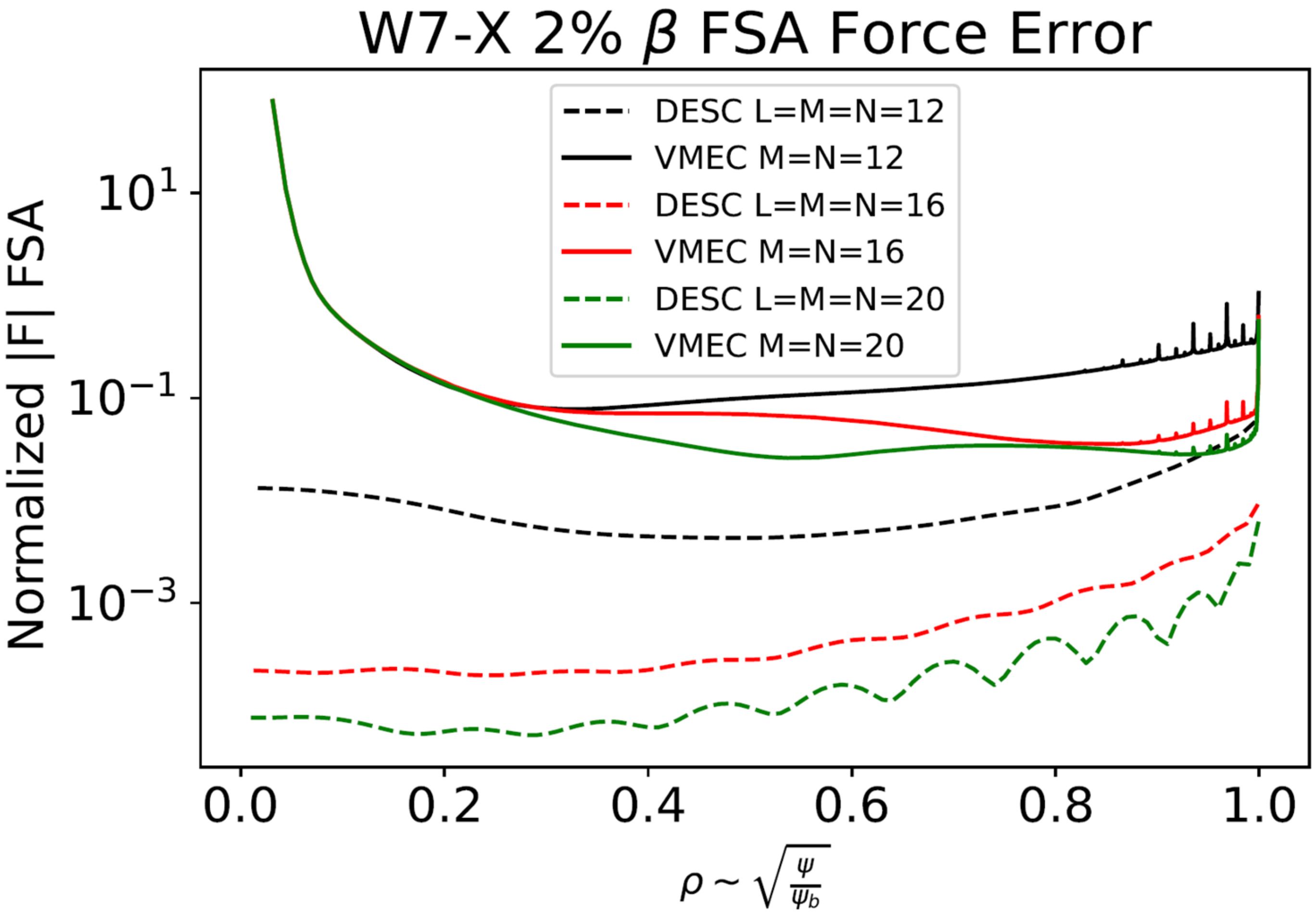
W7-X



Plasma Control

DESC spectral methods yield more accurate equilibrium solutions





W7-X

The Stellarator Optimization Problem

Optimized Stellarator

DESC

Optimization Algorithm

Gradient Information

Constraints

- Ideal MHD force balance
- Equilibrium profiles
- Some boundary modes

Objectives

- Quasi-symmetry
- Mercier stability
- Aspect ratio
- etc.

Stellarator Optimization with DESC Automatic Differentiation (AD)

- Once an equilibrium solution is found, it may not be "good" in the sense of some physics objective g(x,c) (stability, particle confinement, etc)
- So, want to <u>optimize</u> the inputs to the problem to find solutions with improved objective values

Equilibrium Solution	$x^* = [R_{lmn}, Z_{lmn}, \lambda_{lmn}]$	Inputs	$c = [p, i, R_b, Z_b]$	Satisfy Force Balance	$f(x^*, c) = 0$

- Want to optimize some objective g(x,c) wrt the inputs c
- Need derivative information!!

• Conventionally, must use finite differences and change c one element at a time, and resolve • Takes len(c) equilibrium solves -> Expensive • Finite differences are inaccurate	$\Rightarrow \boxed{\dfrac{\partial g}{\partial c}} \Leftarrow$	• DESC AD with JAX gives fast, accurate derivative information • Obtain necessary derivatives with one equilibrium solve!

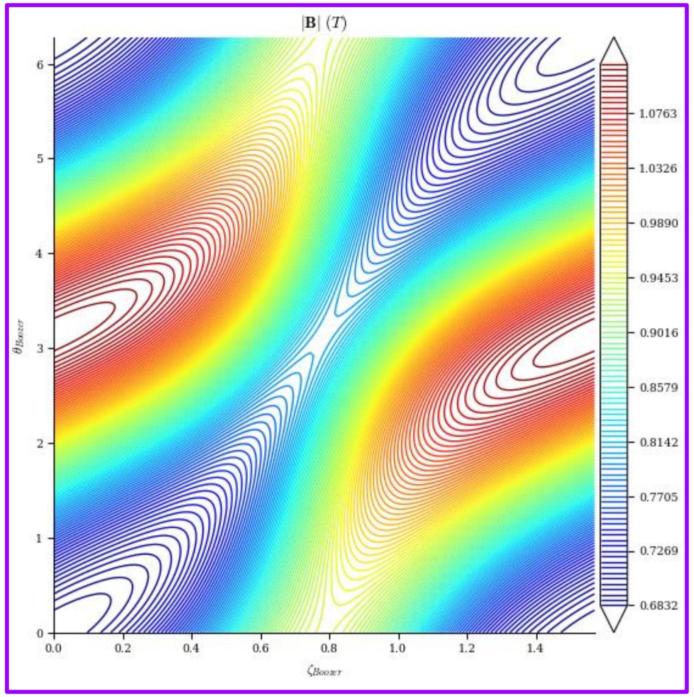
Stellarator Optimization in DESC - Example

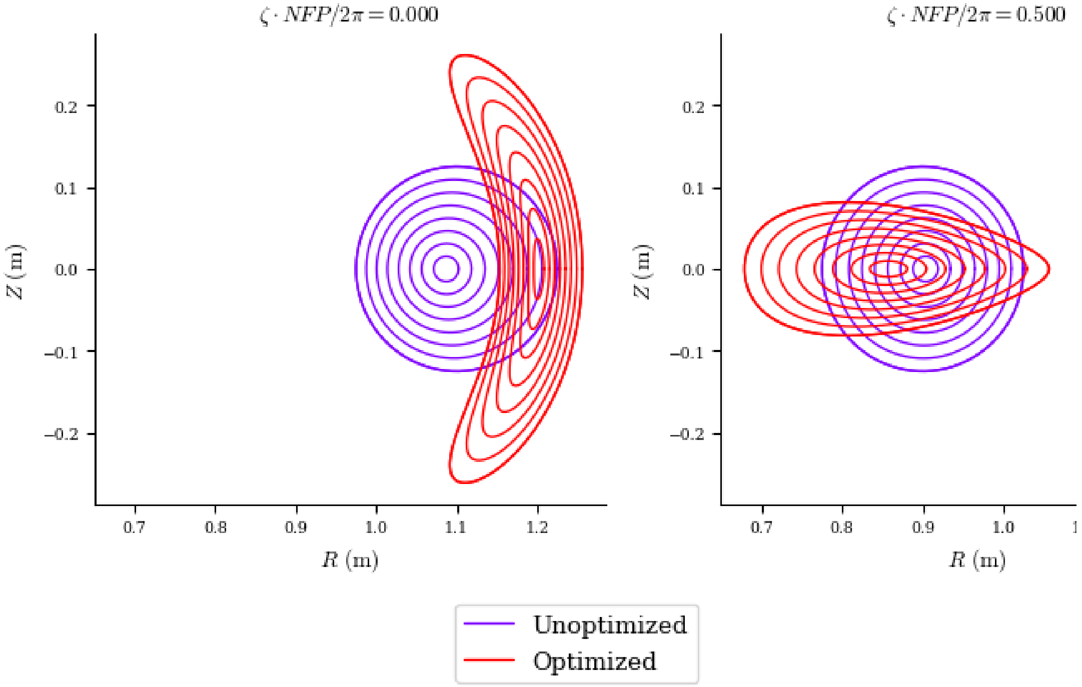
Optimizing for “Quasisymmetry” - Proxy metric for particle confinement

- **In Quasisymmetry, |B| is 2D fxn on a given flux surface**

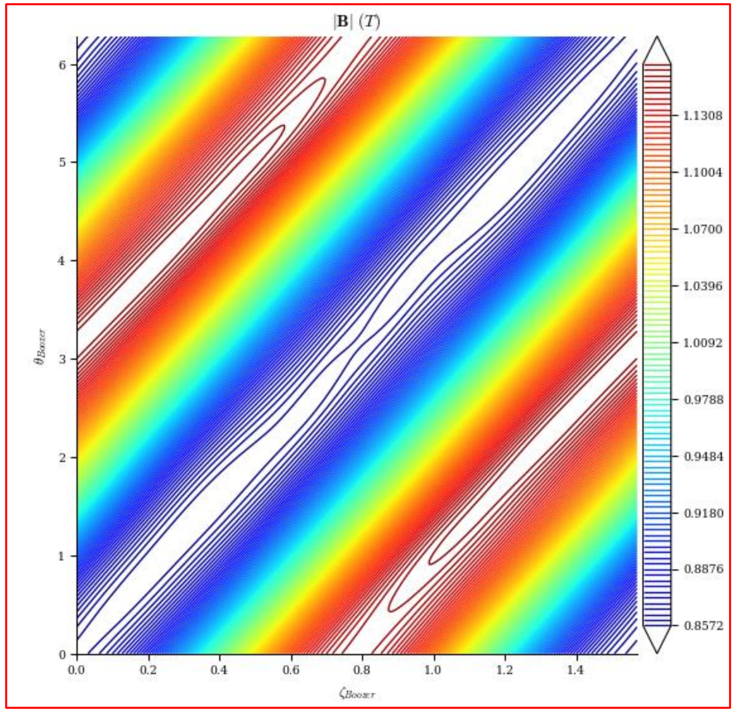
$$|B| = |B|(\rho, M\theta_B - N\zeta_B)$$

Unoptimized |B|

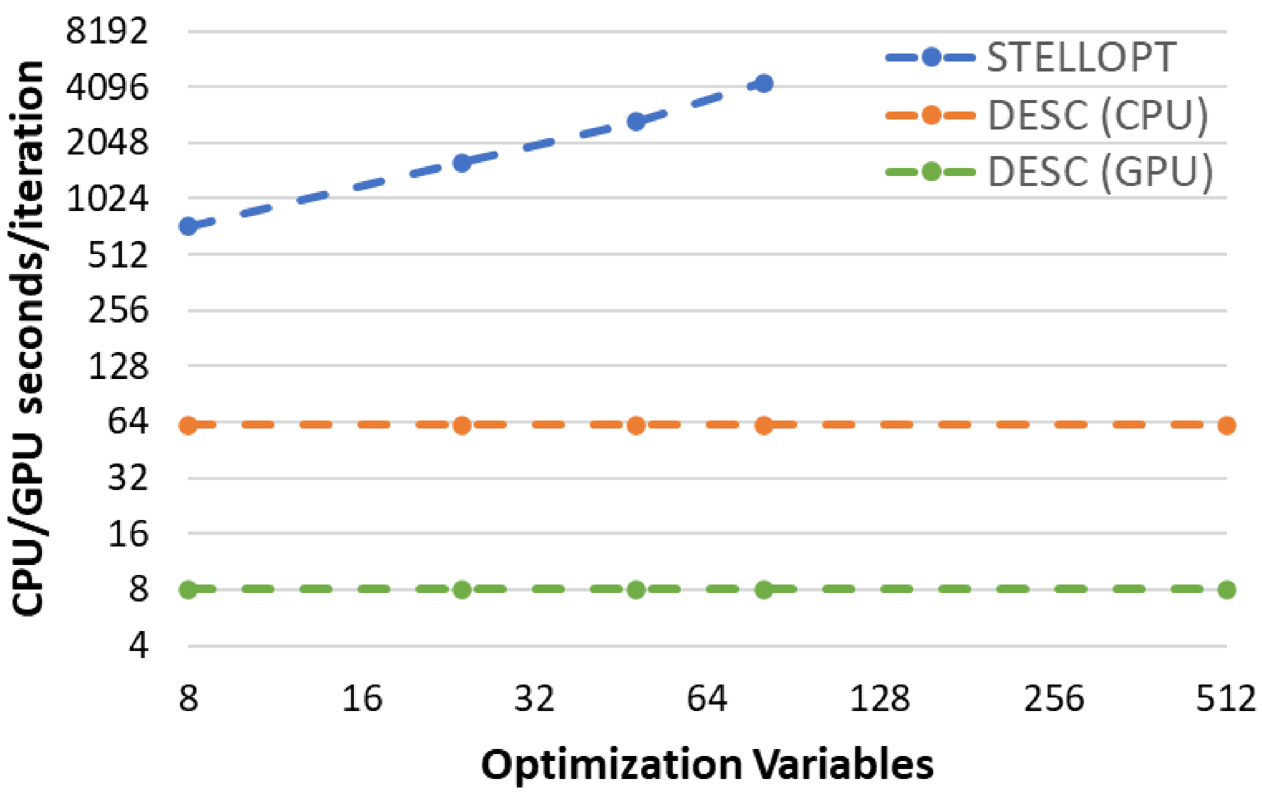




Optimized |B|



DESC Allow Much Faster Stellarator Optimization



- Only require one equilibrium solve per optimization iteration

Optimization Code	Computation Time
STELLOPT (8 CPUs)	~ 2 hours
STELLOPT (16 CPUs)	~ 1.5 hours
STELLOPT (32 CPUs)	~ 1 hour
DESC (1 CPU)	< 30 minutes
DESC (1 GPU)	< 10 minutes

Near-Axis Constrained Equilibria In DESC

D. Panici / Feb 2024

Plasma Control

Near-Axis Expansion

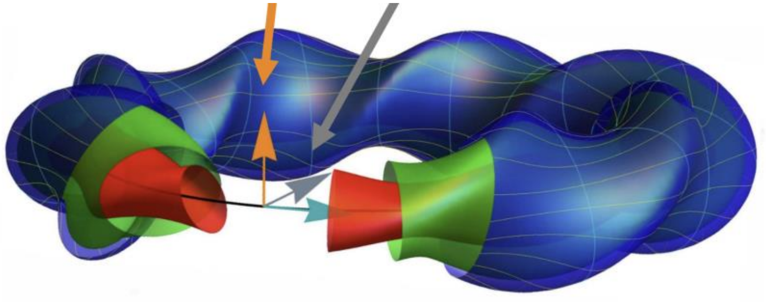
- **Quantities are expanded in form**

$$B_1(\vartheta,\varphi) = B_{1s}(\varphi)\sin(\vartheta) + B_{1c}(\varphi)\cos(\vartheta),$$
$$B_2(\vartheta,\varphi) = B_{20}(\varphi) + B_{2s}(\varphi)\sin(2\vartheta) + B_{2c}(\varphi)\cos(2\vartheta)$$

$$B(r,\vartheta,\varphi) = B_0(\varphi) + rB_1(\vartheta,\varphi) + r^2B_2(\vartheta,\varphi) + r^3B_3(\vartheta,\varphi) + \dots .$$

- **Inputs for $O(r^2)$ solutions**
  - **Axis Shape (R(phi), Z(phi))**
  - $\bar{\eta} = \frac{B_{1c}}{B_0}$ **Measure of magnetic field variation** $B = B_0\left[1 + r\bar{\eta}\cos\vartheta + O(r^2)\right]$
  - σ_0 **Deviation from stellarator symmetry at $\phi = 0$**
    - **Taken as 0 for most cases**
  - I_2 **Current Density on axis**
  - p_2 **Pressure near axis**
  - B_{2c} **magnetic field $O(r^2)$ poloidal variation**

Outputs:
- **Flux surface shapes in neighborhood of axis**
- ι_0 **rotational transform on-axis**
- B_{20} **magnetic field variation on-axis**



(Jorge 2022)

$$\boldsymbol{r}(r,\vartheta,\varphi) = \boldsymbol{r}_0(\varphi) + X(r,\vartheta,\varphi)\boxed{\boldsymbol{n}(\varphi)} + Y(r,\vartheta,\varphi)\boxed{\boldsymbol{b}(\varphi)} + Z(r,\vartheta,\varphi)\boxed{\boldsymbol{t}(\varphi)}$$

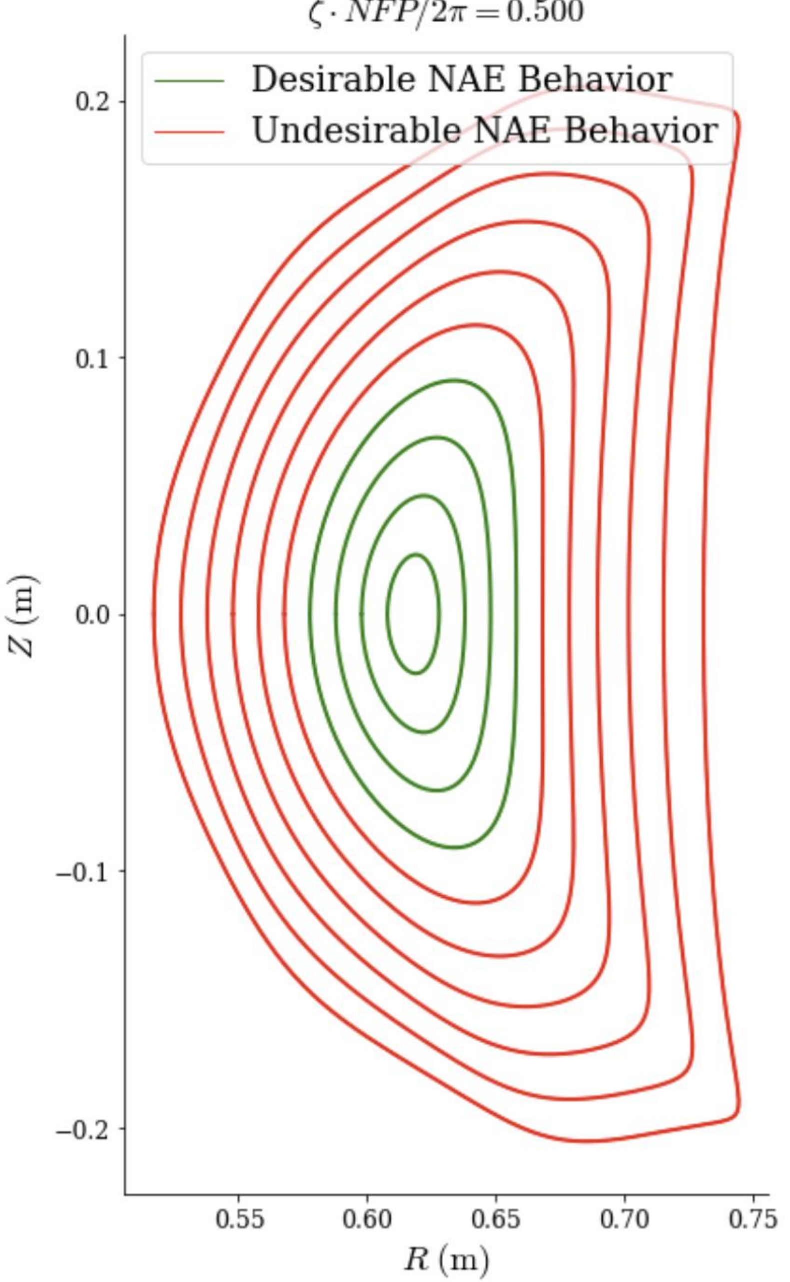
$$X(r,\vartheta,\varphi) = rX_1(\vartheta,\varphi) + r^2X_2(\vartheta,\varphi) + r^3X_3(\vartheta,\varphi) + \dots .$$

(Landreman 2022)

Plasma Control

Near-Axis Expansion (NAE) Constraints in DESC (with E. Rodriguez)

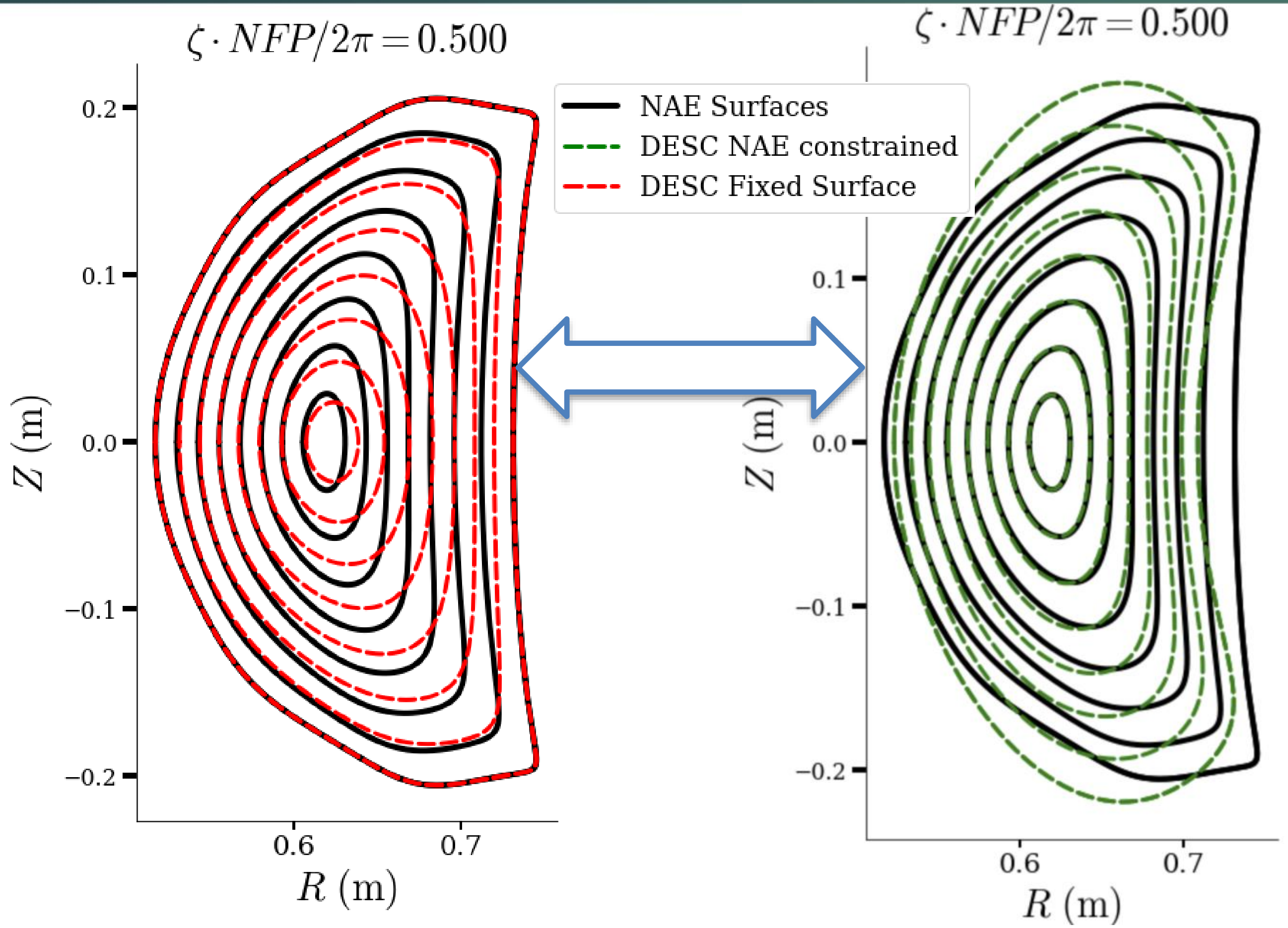
- **Idea is to constrain the global equilibrium to have NAE behavior as $\rho \longrightarrow 0$**
  - **only use information from NAE where it is most valid**
  - **Avoid singular behavior present when evaluating at large r**
- **Map NAE coefficients to Fourier-Zernike modes of DESC to fix $O(\rho^0)$ (axis), $O(\rho^1)$, $O(\rho^2)$ behavior**

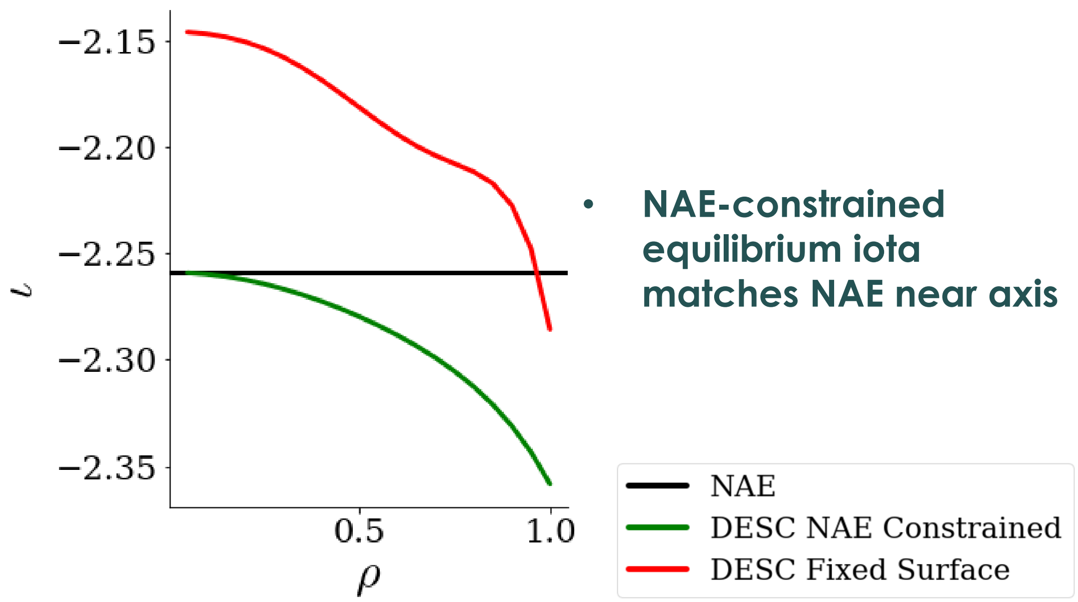


pyQSC equilibrium evaluated at r =0.1

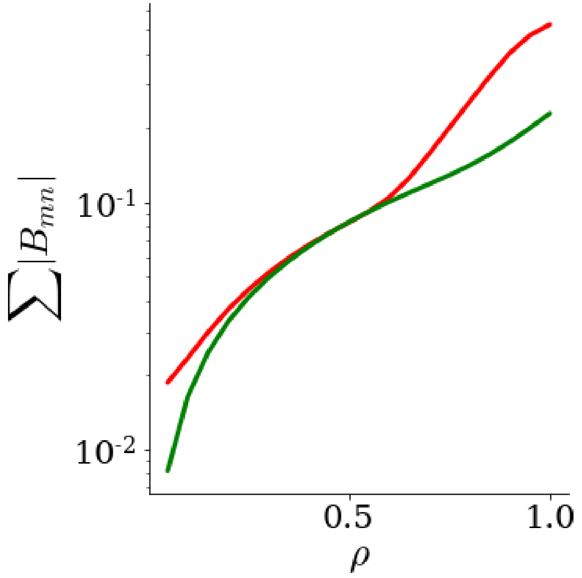
Plasma Control

Near-Axis-Expansion Constrained Equilibria in DESC





- NAE-constrained equilibrium iota matches NAE near axis



- NAE-constrained equilibrium maintains better QS near axis

- Global equilibria solutions with near-axis behavior constrained to match the NAE to O(ρ)
- Enables the connection between global MHD equilibria solutions and the existing insight on optimized stellarators

Coil Optimization In DESC

D. Panici / Feb 2024

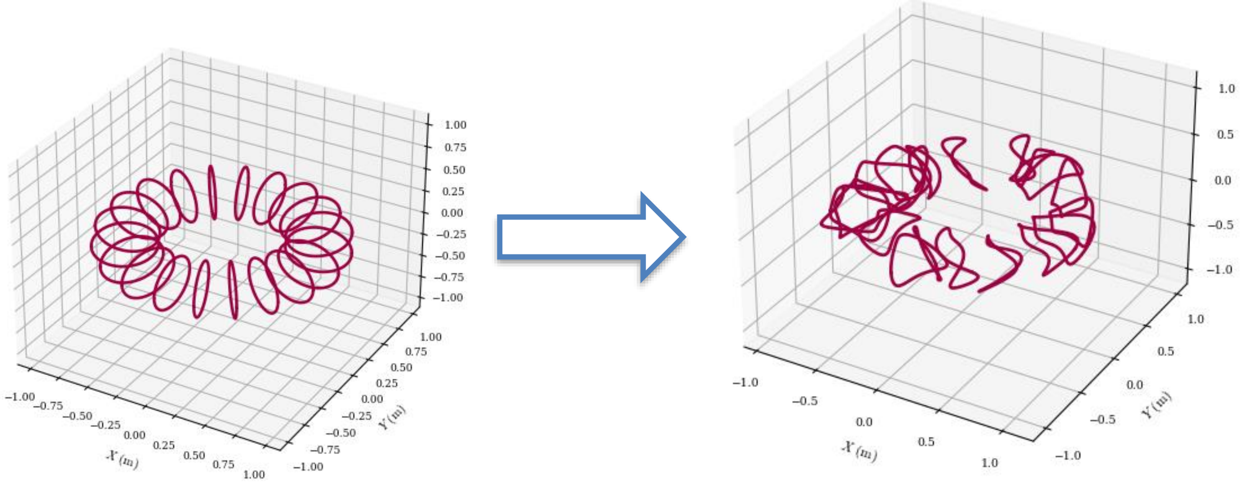
Plasma Control

Stage 2- Coil Optimization

- **During the fixed-boundary equilibrium solve (and optimization) (in DESC), the boundary surface of the equilibrium is assumed to be a flux surface (so B·n=0)**
  - **however, DESC has no knowledge of the coils external to this equilibrium, so <u>we must find coils that make this true</u>**
- **Problem then: Find coils to minimize normal field on surface**

$$\boxed{\chi_B^2 = \int \mathrm{d}^2 a \; B_{\text{normal}}^2}$$

<u>Example Coil Optimization</u>



REGCOIL Algorithm

- **Using surface current distributions on a specified winding is an efficient approach to the coil-finding problem[4,5]**

$$\boldsymbol{K} = \boldsymbol{n} \times \nabla\Phi \qquad \Phi(\theta', \zeta') = \Phi_{sv}(\theta', \zeta') + \frac{G\zeta'}{2\pi} + \frac{I\theta'}{2\pi}$$

Surface Current Density

Unit Normal to Winding Surface

Current Potential

G: Surface Net Poloidal Current

I: Surface Net Toroidal Current

- **Then minimization of quadratic flux becomes a linear (in Φ_{sv}) least-squares problem, after expanding in Fourier Series (I,G, and other terms are known)**

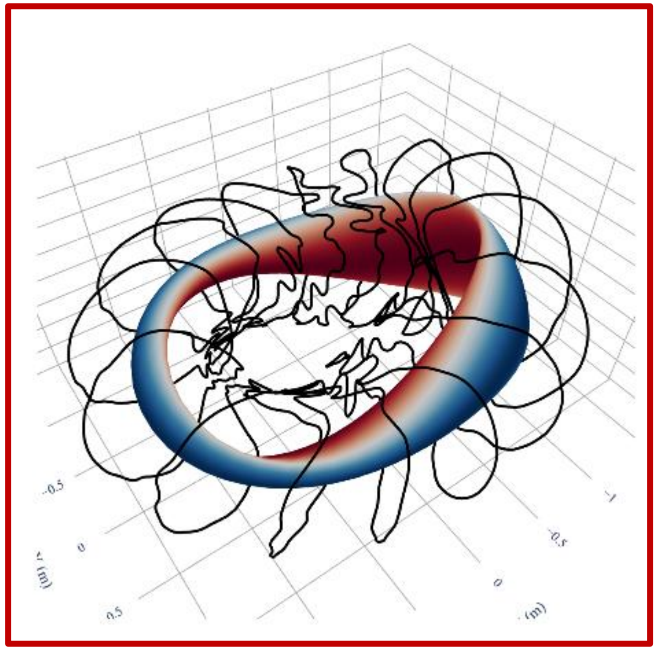
$$\chi_B^2 = \int \mathrm{d}^2a \; B_{\text{normal}}^2 \qquad \Phi_{sv} = \sum_{m,n} \Phi_{sv}^{mn} sin(m\theta' - n\zeta') \qquad B_n = B_n^{ext} + B_n^{pl} + B_n^{GI} + A\Phi_{sv}^{mn}$$

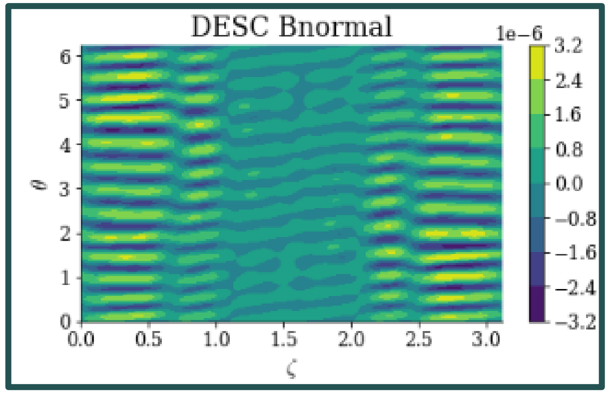
- **However, can lead to poor solutions without regularization -> REGCOIL adds regularization to the problem**

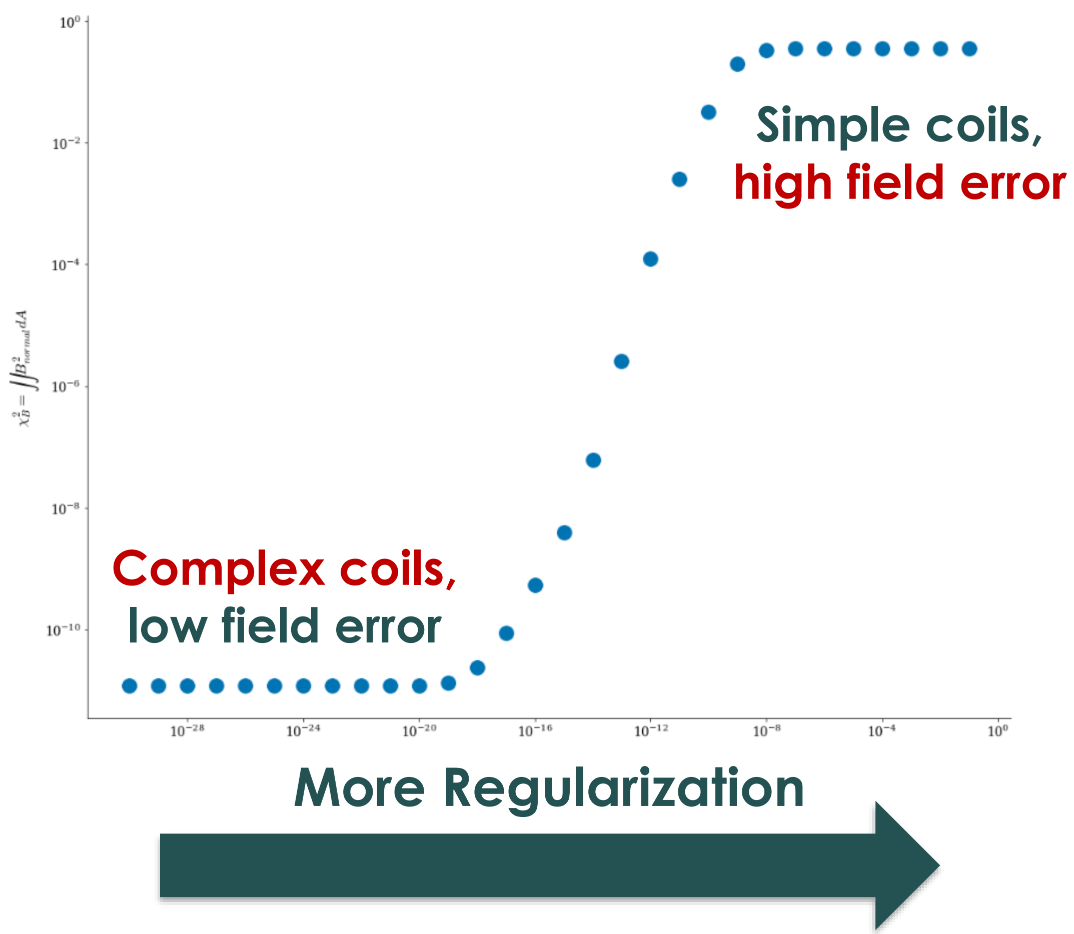
$$\chi_K^2 = \int \mathrm{d}^2a' \; K(\theta', \zeta')^2.$$

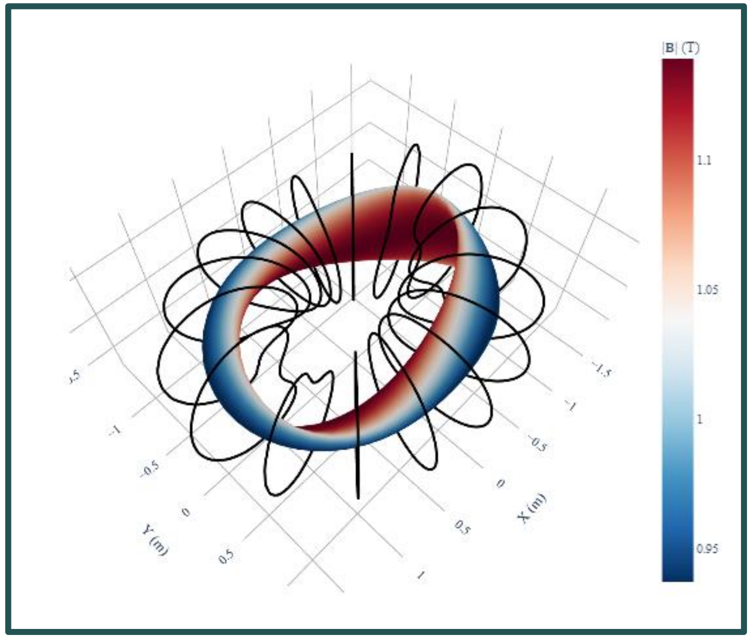
Plasma Control

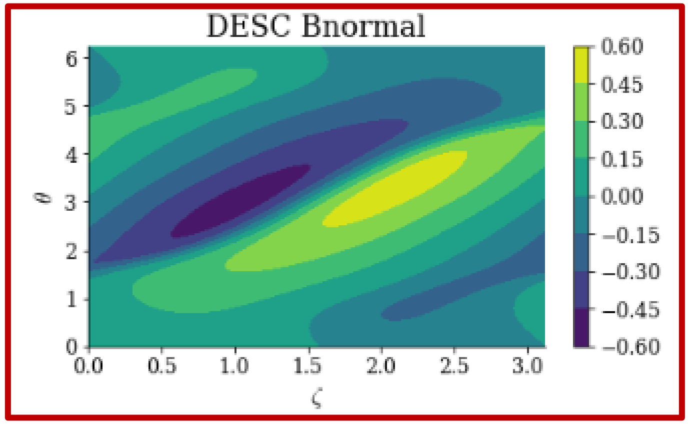
More Regularization Creates Simpler Coils at Expense of Field Error



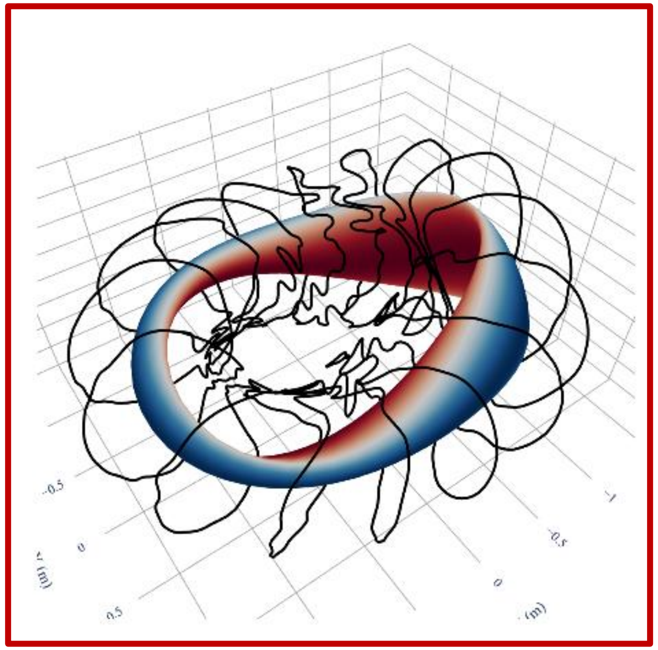


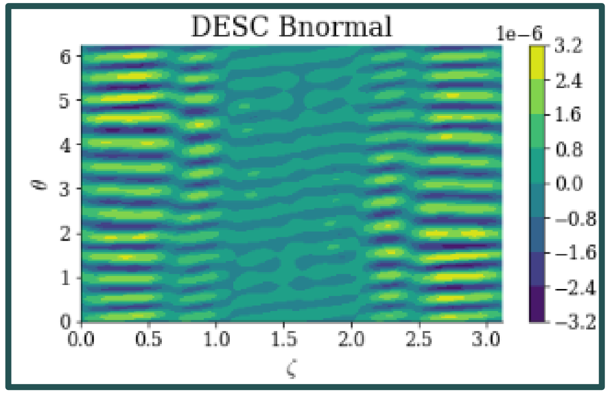




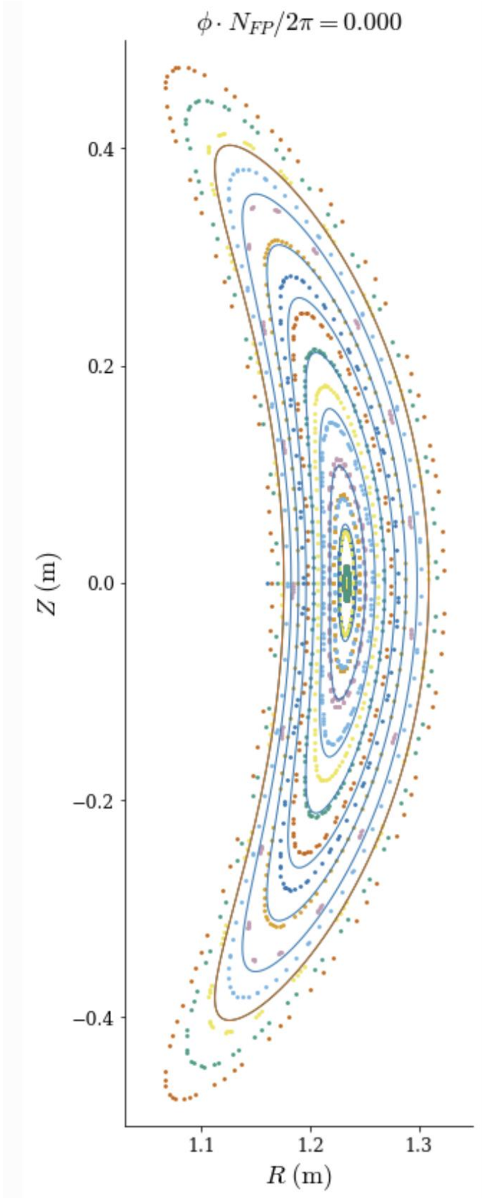


Low Normal Field Error -> Coil field has Flux Surfaces





Biot-Savart Law Field Tracing



Free Boundary Equilibria In DESC

D. Panici / Feb 2024

Free Boundary Problem

Fixed boundary

R^b, Z^b

$$\boldsymbol{J} \times \boldsymbol{B} = \nabla p$$

$$\nabla \times \boldsymbol{B} = \mu_0 \boldsymbol{J}$$

$$\nabla \cdot \boldsymbol{B} = 0$$

Plasma

Free boundary

$$\boldsymbol{J} \times \boldsymbol{B} = \nabla p$$

$$\nabla \times \boldsymbol{B} = \mu_0 \boldsymbol{J}$$

$$\nabla \cdot \boldsymbol{B} = 0$$

$$\nabla \cdot \boldsymbol{B_v} = 0$$

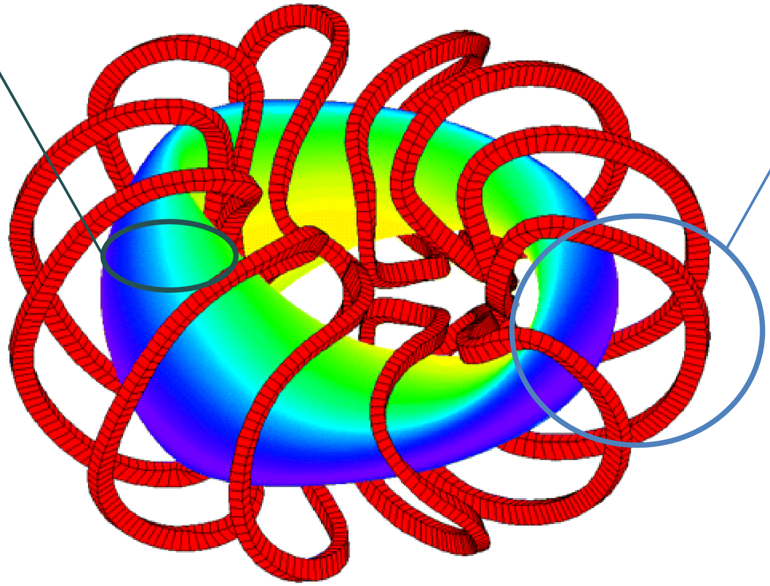
$$\nabla \times \boldsymbol{B_v} = 0$$

Plasma

Vacuum

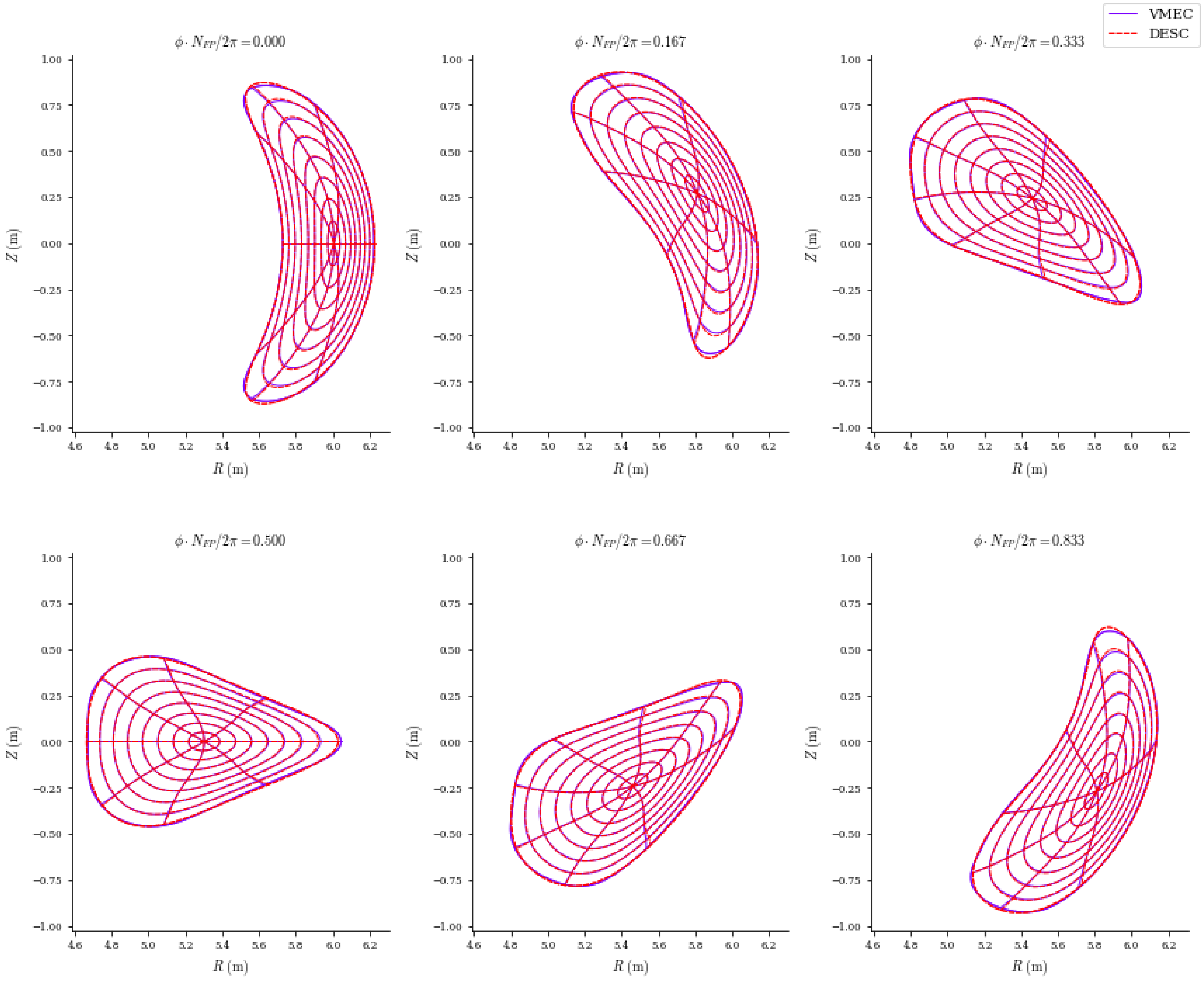
$$\frac{B^2}{2\mu_0} + p = \frac{B_v^2}{2\mu_0}$$

$$\boldsymbol{B} \cdot \boldsymbol{n} = \boldsymbol{0}$$



Alexander and Garabedian (2007)

Free Boundary Solve DESC vs VMEC – Finite Beta W7-X



Plasma Control

Omnigenity Optimization In DESC

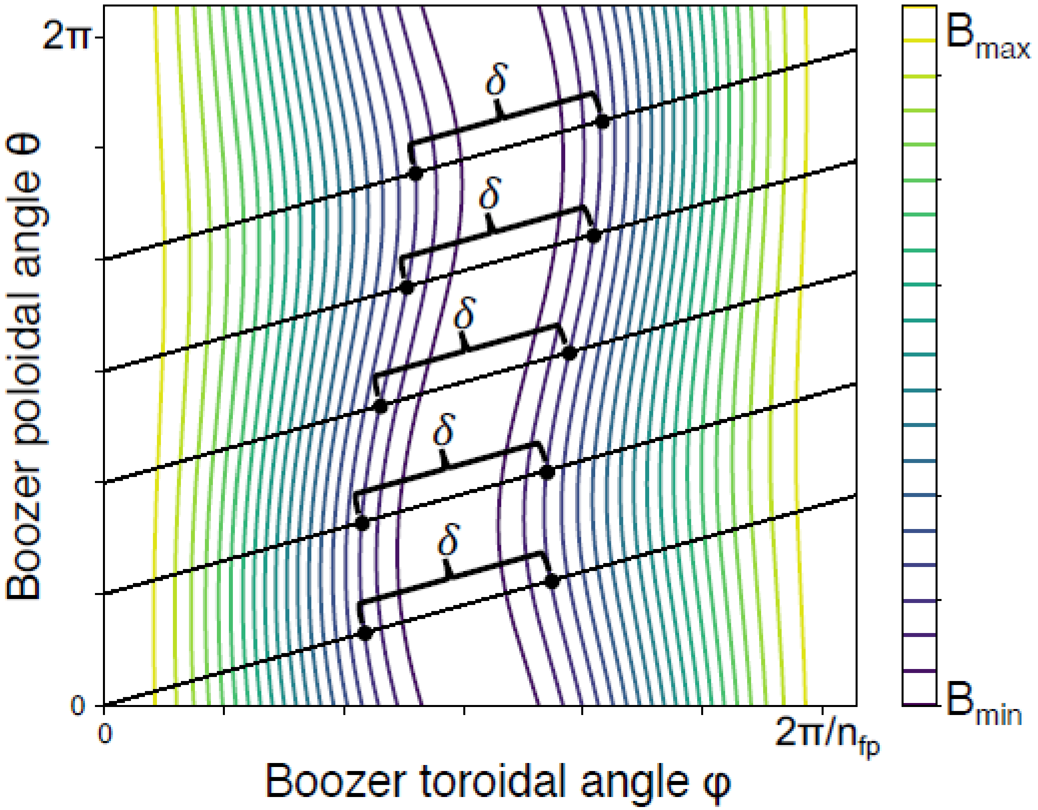
D. Panici / Feb 2024

Plasma Control

Omnigenous magnetic fields

Quasi-Isodynamic (QI) magnetic fields = omnigenous magnetic fields with constant |B| contours that close poloidally

Particles in omnigenous magnetic fields have no net radial drifts



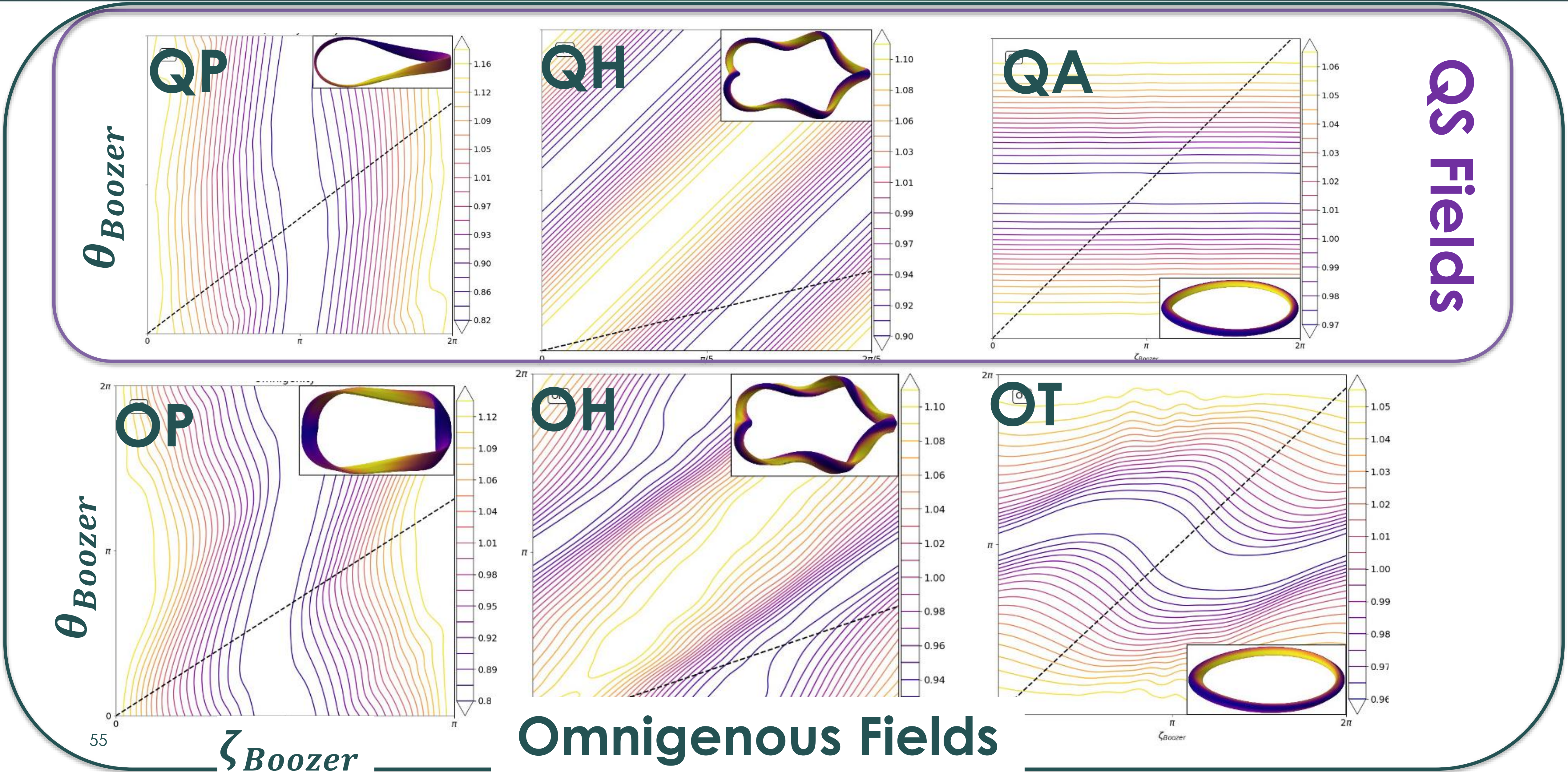
Conditions for Omnigenity:

- B_{max} is a straight contour in Boozer coordinates
- Constant "bounce distance" δ between consecutive points of equal B on each field line α

$$\delta = \sqrt{\Delta\theta_B^2 + \Delta\zeta_B^2} \propto \Delta\zeta_B \qquad \frac{\partial\delta}{\partial\alpha} = 0$$

(Goodman 2022)

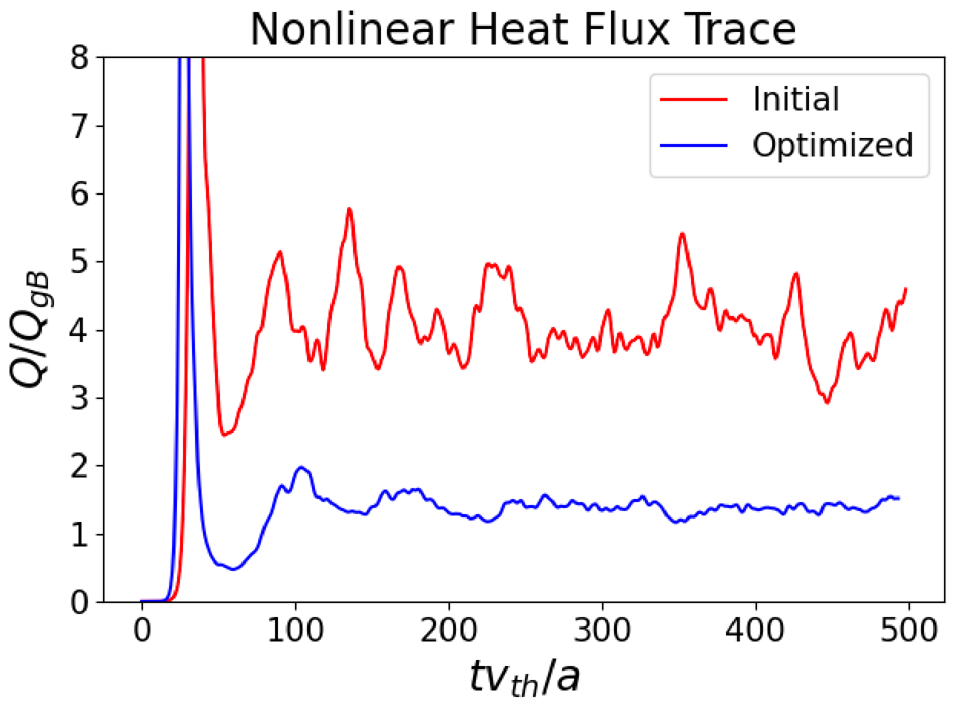
DESC can find equilibria with any omnigenity type

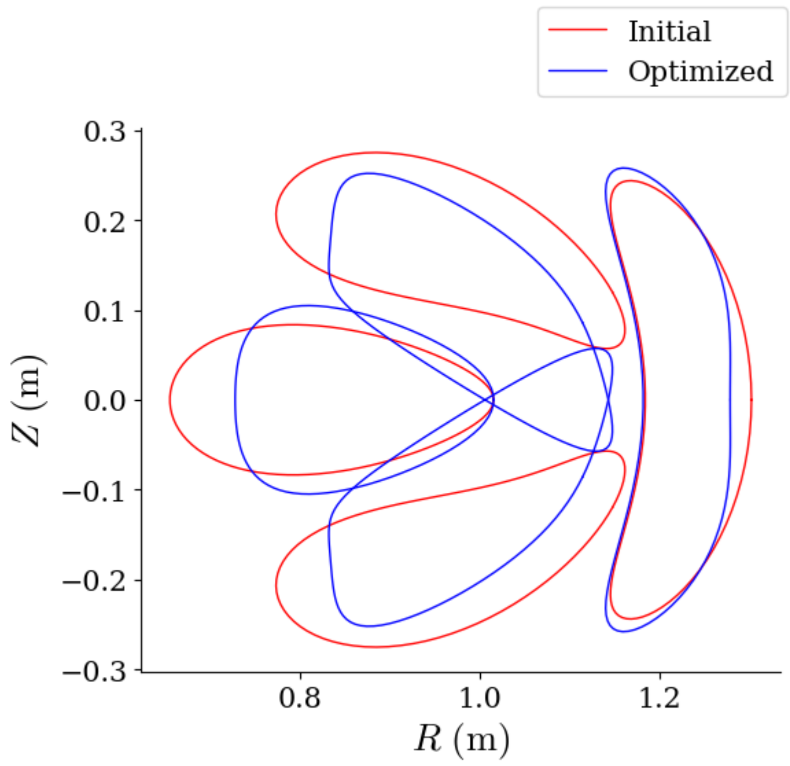


Other Work and Opportunities with DESC

Turbulence + QS Optimization Using GX+ DESC (P. Kim)

- **GX + DESC coupling enables direct optimization of nonlinear heat fluxes with good quasi-symmetry.**
- **SPSA algorithm allows for cheaper gradient approximations for noisy objectives.**
- **Optimizer reduces nonlinear heat flux by a factor of 3, while maintaining good quasi-symmetry.**



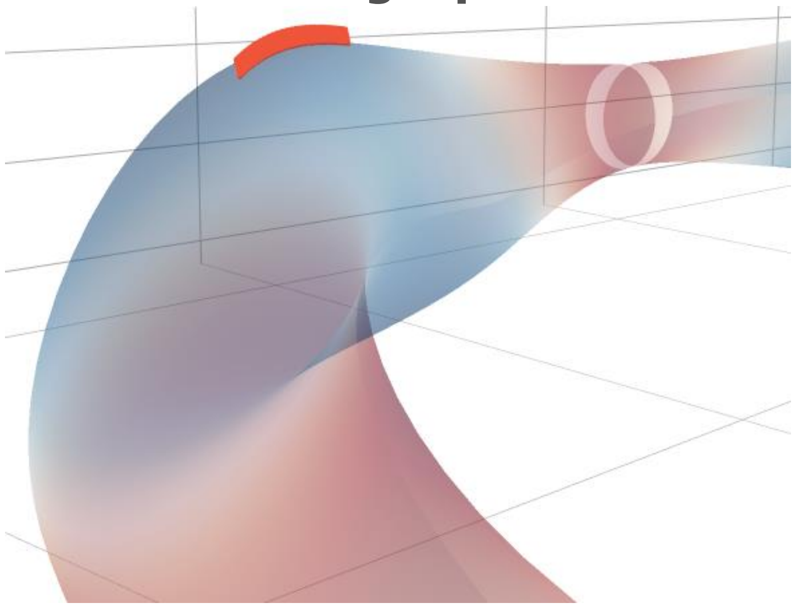


Plasma Control

Direct Optimization of Particle Trajectories (J. Biu, TU. Lisbon)

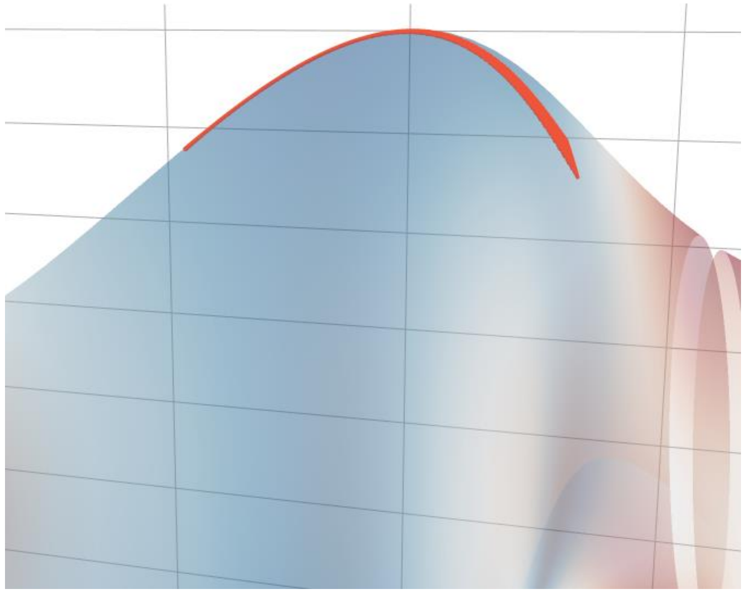
- **Particle Tracing:**
  - **Integrate Guiding Center EoM directly**
  - **Optimize the equilibrium from particle’s trajectories using JAX autodiff**

Starting Equilibrium

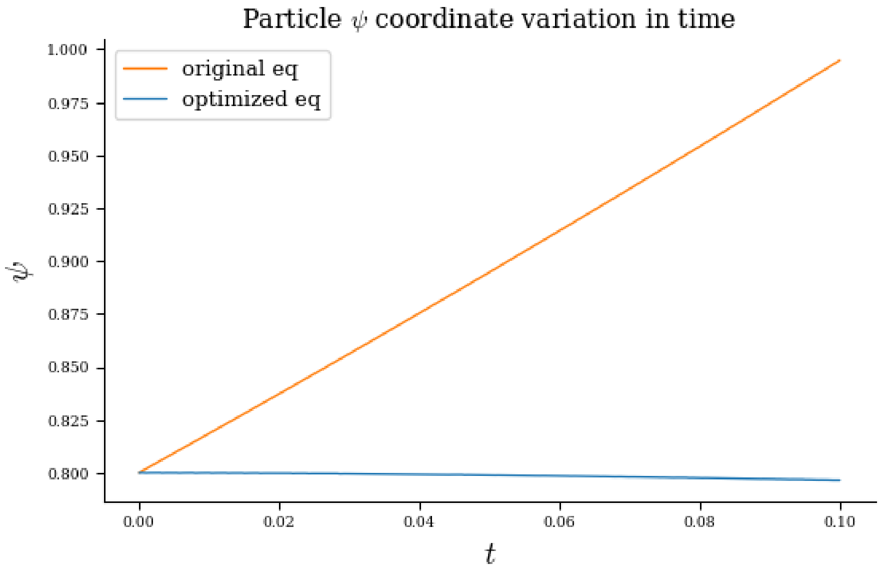


Particle drifting from flux surface ψ

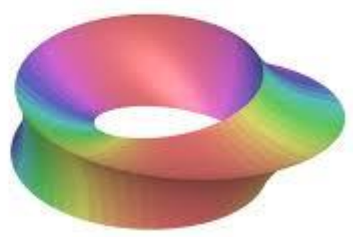
Optimized Equilibrium



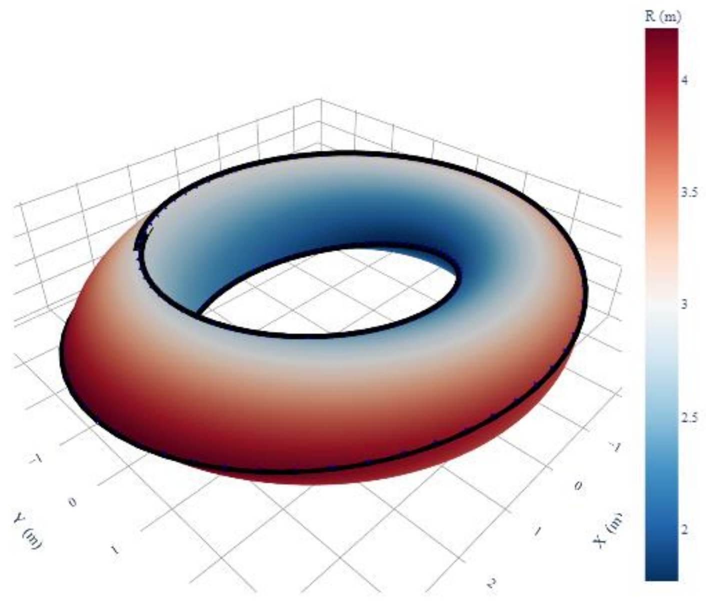
Particle confined in flux surface ψ

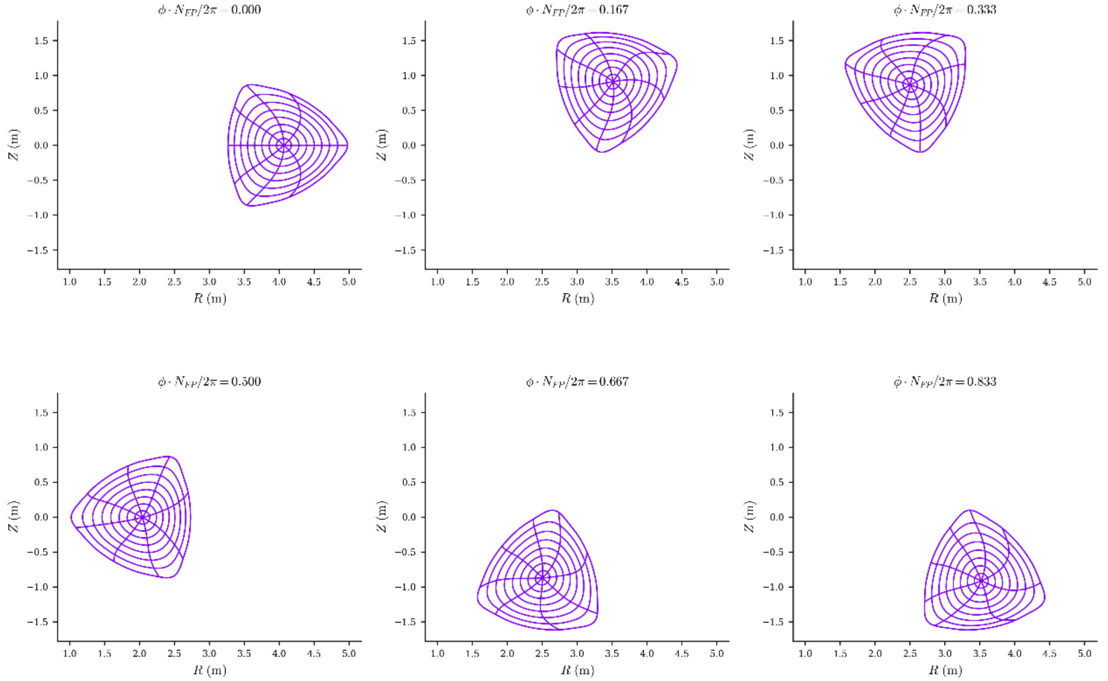


Umbilic Stellarator Design and Analysis (R. Gaur, PU)



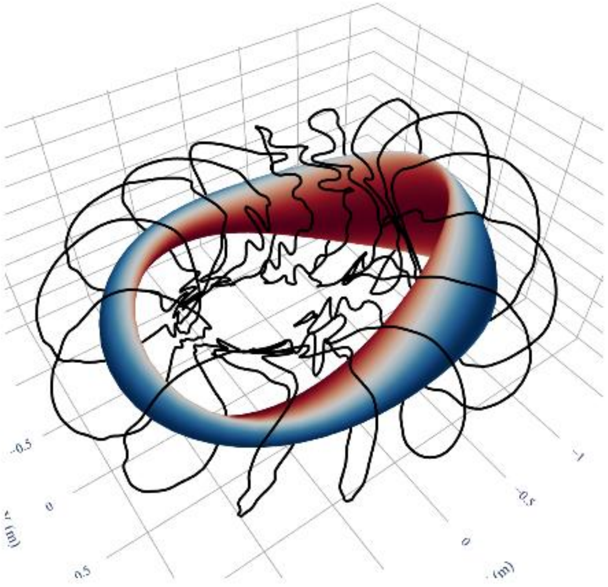
- **“Umbilic” shapes are shapes with a single side (think Mobius strips)**
  - **In our framework, they would be stellarators with fractional NFP**
- **Possible implication for stellarators**
  - **long connection length -> reduce heat flux to divertor**
  - **Natural locations for X-point or island divertor**
- **Capability implemented in DESC to investigate umbilic topologies**

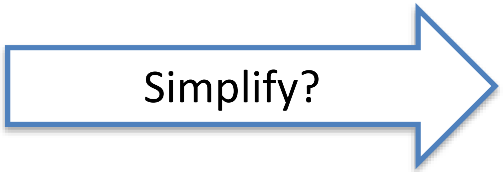


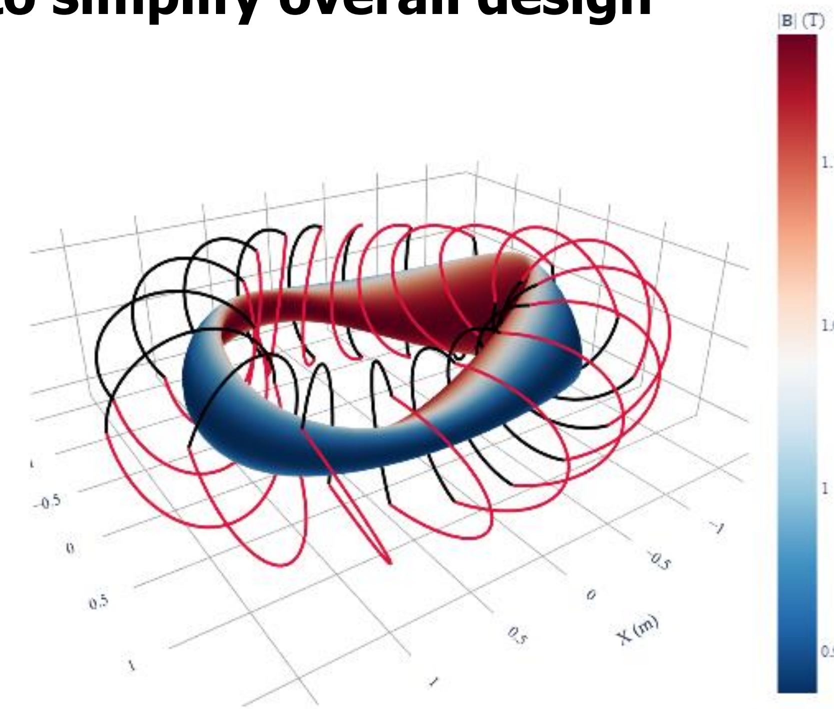


Improvements to Coil Optimization - C^0 Coils?

- **Coil optimization represents coil geometries with Fourier series**
  - **Smooth curves, but can have tight curvature**
  - **Typically penalize things like length, curvature, torsion to ease engineering**
- **However, piecewise continuous (C^0) coils cannot be represented with Fourier**
  - **May be an overlooked design space for coils that are simpler to manufacture than arbitrarily shaped modular coils**
  - **Could use in conjunction with other types of coils to simplify overall design**







DESC Offers Unique Flexibility in Constraints that Open New Possibilities

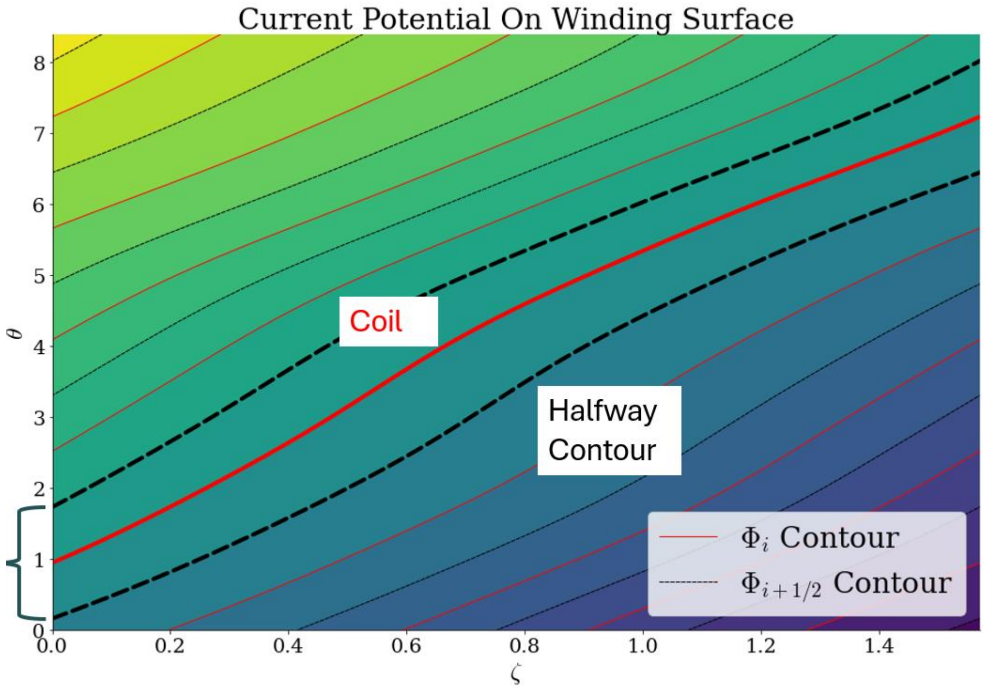
- DESC enables efficient and accurate equilibrium solving and optimization
- Free boundary and coil capabilities now implemented

- Opportunities to get Involved!
  – Umbilic stellarator equilibrium, coil, and divertor design and analysis
  – C^0 coil design and optimization
  – Flexible stellarator optimization using 2D coil surface
  – And more!

- Email Egemen to get involved: ekolemen@princeton.edu
  – I am also happy to meet and talk! dpanici@pppl.gov

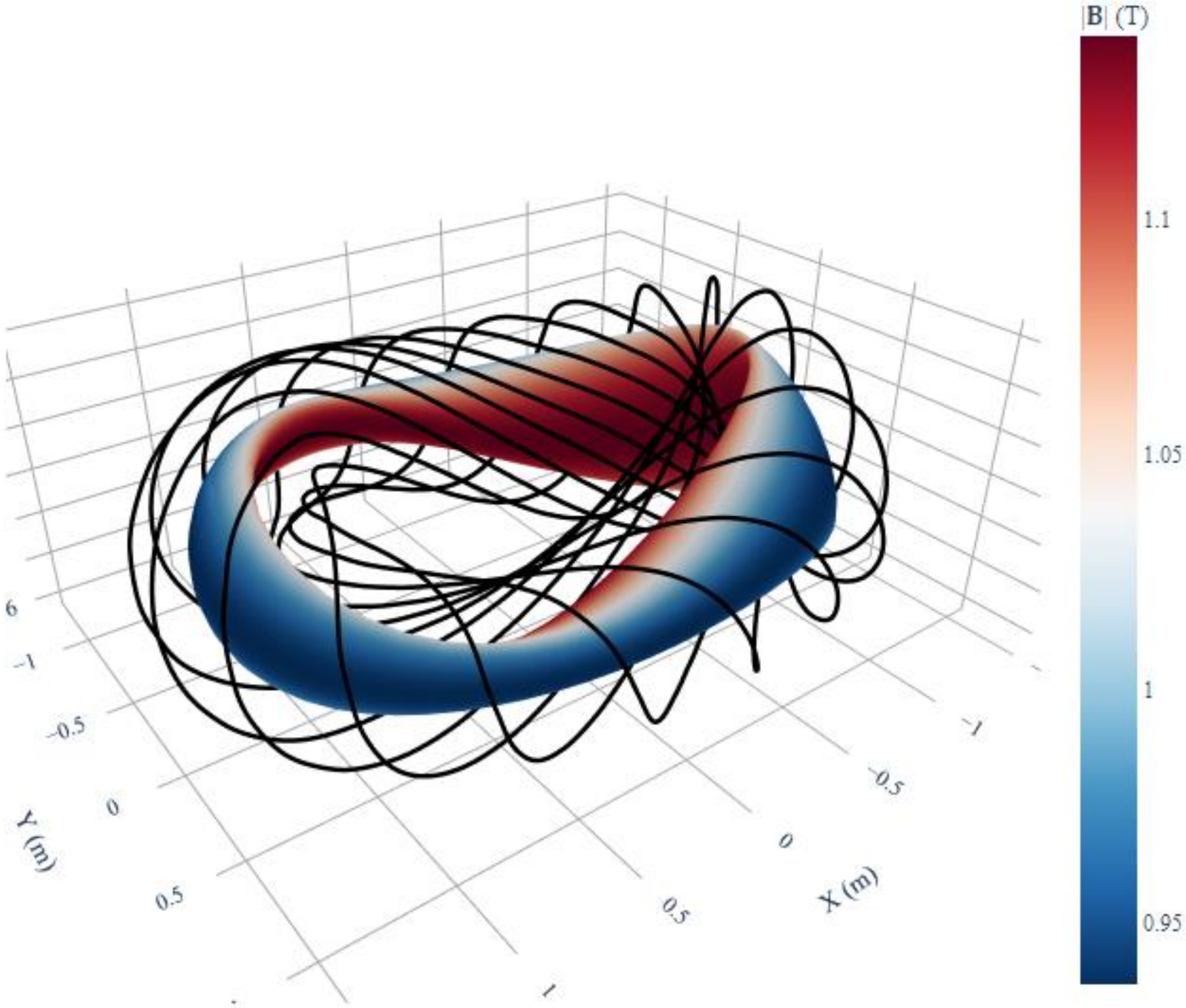
Backup

Helical Coilset in DESC



REGCOIL Algorithm implemented in DESC to find surface currents

Algorithms to discretize into coils also implemented



Stellarator Optimization - Can optimize parameters while maintaining the equilibrium constraint

- Let $\boldsymbol{g}(\boldsymbol{x}, \boldsymbol{c})$ be a physics/engineering objective that we wish to optimize

Taylor expansion:

$$\boldsymbol{g}(\boldsymbol{x} + \Delta\boldsymbol{x}, \boldsymbol{c} + \Delta\boldsymbol{c}) = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{c}) + \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}} \Delta\boldsymbol{x} + \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{c}} \Delta\boldsymbol{c}$$

$\boxed{0}$

equilibrium perturbation

least-squares problem:

$$\left[\frac{\partial \boldsymbol{g}}{\partial \boldsymbol{x}} \left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{x}}\right)^{-1} \frac{\partial \boldsymbol{f}}{\partial \boldsymbol{c}} - \frac{\partial \boldsymbol{g}}{\partial \boldsymbol{c}}\right] \Delta\boldsymbol{c} = \boldsymbol{g}(\boldsymbol{x}, \boldsymbol{c})$$

- Yields the optimal perturbation to improve the objective $\boldsymbol{c}^* = \boldsymbol{c} + \Delta\boldsymbol{c}$
- Extended to higher-order approximations with little additional cost
- Used in an adaptive Gauss-Newton trust-region optimization method

Preliminary: DESC represents solution in Fourier-Zernike Basis

spectral coefficients

Zernike polynomials

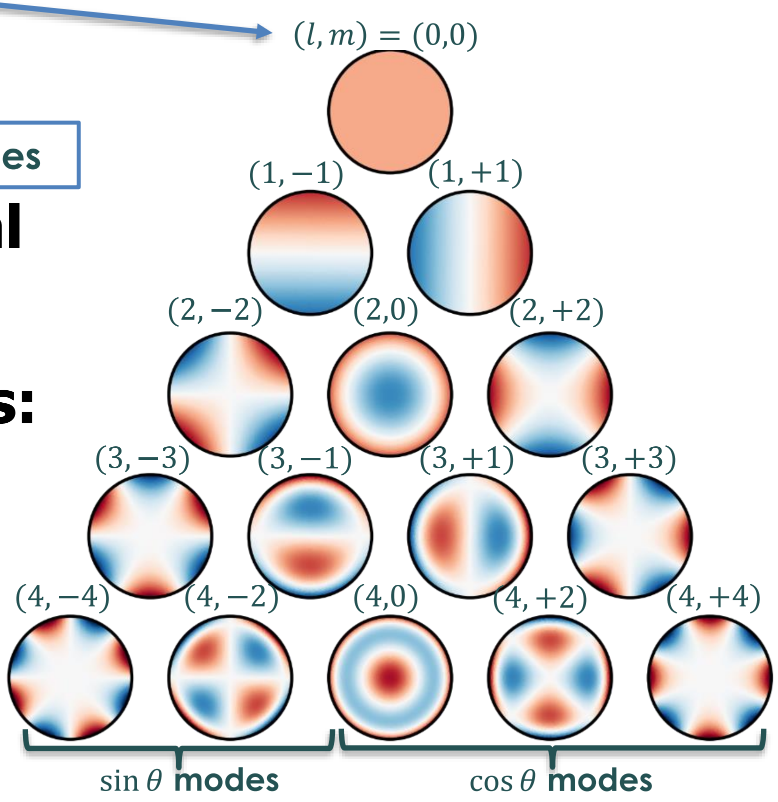
$$X(\rho,\theta,\zeta) = \sum_{lmn} X_{lmn} \mathcal{Z}_l^m(\rho,\theta)\mathcal{F}^n(\zeta)$$

Fourier series

- **Periodic boundary conditions for poloidal & toroidal angles**
- **Satisfies analyticity conditions at the magnetic axis:**

$$f(\rho,\theta) = \sum_m \rho^m\left(a_{m,0} + a_{m,2}\rho^2 + \cdots\right)\cos(m\theta) + \sum_m \rho^m\left(b_{m,0} + b_{m,2}\rho^2 + \cdots\right)\sin(m\theta)$$

- **Exponential convergence (if solution exists and is smooth)**



Boyd et al., *J. Comput. Phys.* (2011)
Lewis et al., *J. Math. Phys.* (1990)

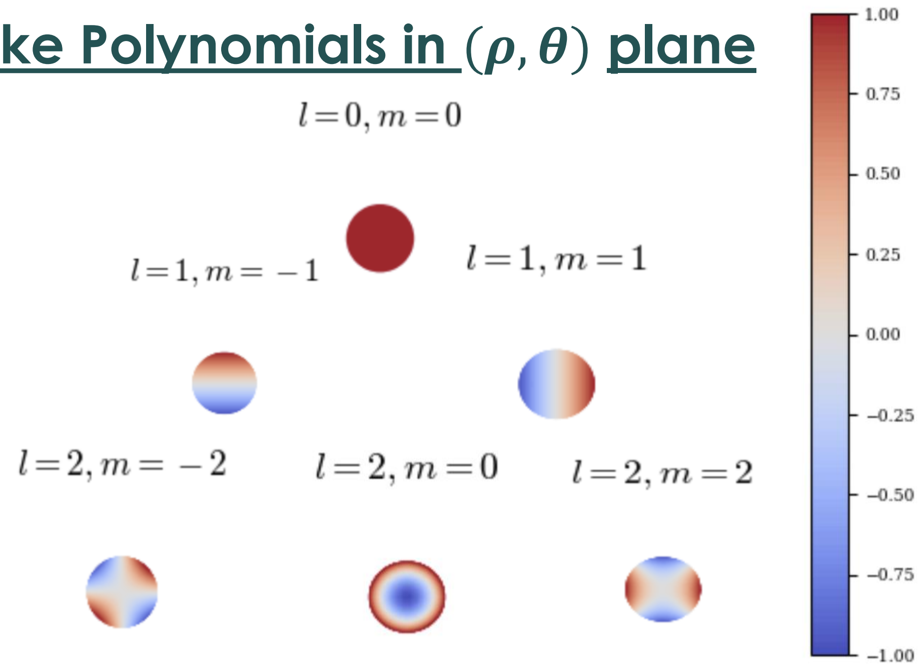
DESC - Fourier-Zernike Spectral Basis

$$R(\rho,\theta,\zeta) = \sum_{m=-M,n=-N,l=0}^{M,N,L} R_{lmn}\mathcal{Z}_l^m(\rho,\theta)\mathcal{F}^n(\zeta)$$

$$\lambda(\rho,\theta,\zeta) = \sum_{m=-M,n=-N,l=0}^{M,N,L} \lambda_{lmn}\mathcal{Z}_l^m(\rho,\theta)\mathcal{F}^n(\zeta)$$

$$Z(\rho,\theta,\zeta) = \sum_{m=-M,n=-N,l=0}^{M,N,L} Z_{lmn}\mathcal{Z}_l^m(\rho,\theta)\mathcal{F}^n(\zeta)$$

Zernike Polynomials in (ρ,θ) plane



$$\mathcal{Z}_m^l(\rho,\theta) = \begin{cases} \mathcal{R}_l^{|m|}(\rho)cos(|m|\theta) & \text{for } m \geq 0 \\ \mathcal{R}_l^{|m|}(\rho)sin(|m|\theta) & \text{for } m < 0 \end{cases}$$

Zernike Polynomials

$$\mathcal{R}_l^{|m|}(\rho) = \sum_{s=0}^{(l-|m|)/2} \frac{(-1)^s(l-s)!}{s!\,[(l+|m|)/2-s]!\,[(l-|m|)/2+s]!}\rho^{l-2s}$$

Radial Polynomial is of degree *l-2s*, and *l=m,m+2,…, L*

$$\mathcal{F}^n(\zeta) = \begin{cases} cos(|n|N_{FP}\zeta)\text{for } n \geq 0 \\ sin(|n|N_{FP}\zeta)\text{for } n < 0 \end{cases}$$

Toroidal Fourier Series

Plasma Control

DESC Allows Flexible Constraints when Defining Equilibrium Problem - Fixed ρ=1 Boundary

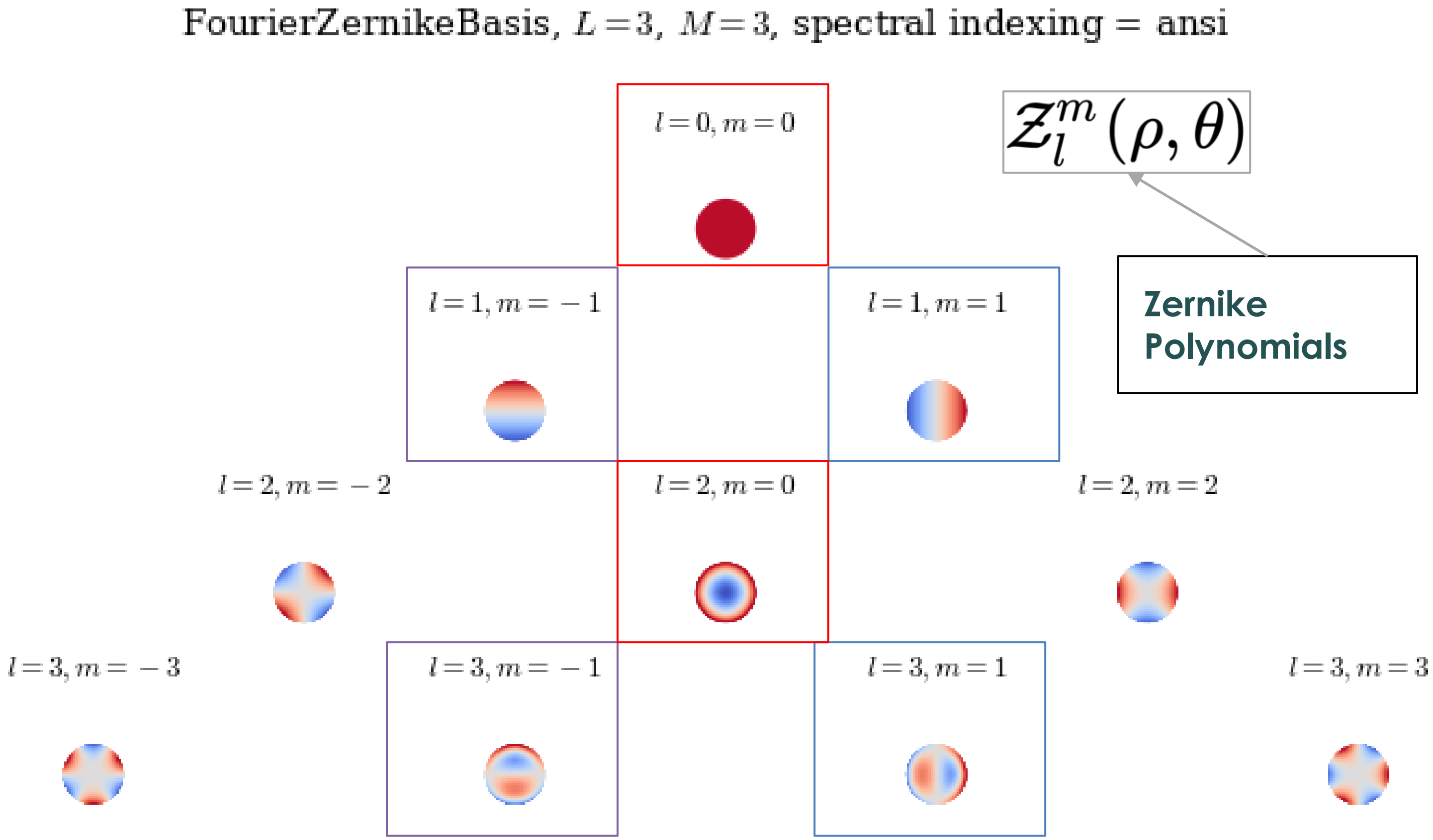
$$R^b(\theta,\zeta) = \sum_{m=0}^{M}\sum_{n=-N}^{N} R^b_{m,n} cos(m\theta - n\zeta)$$

$$Z^b(\theta,\zeta) = \sum_{m=0}^{M}\sum_{n=-N}^{N} Z^b_{m,n} sin(m\theta - n\zeta)$$

Fixed-Boundary ρ=1 Constraint

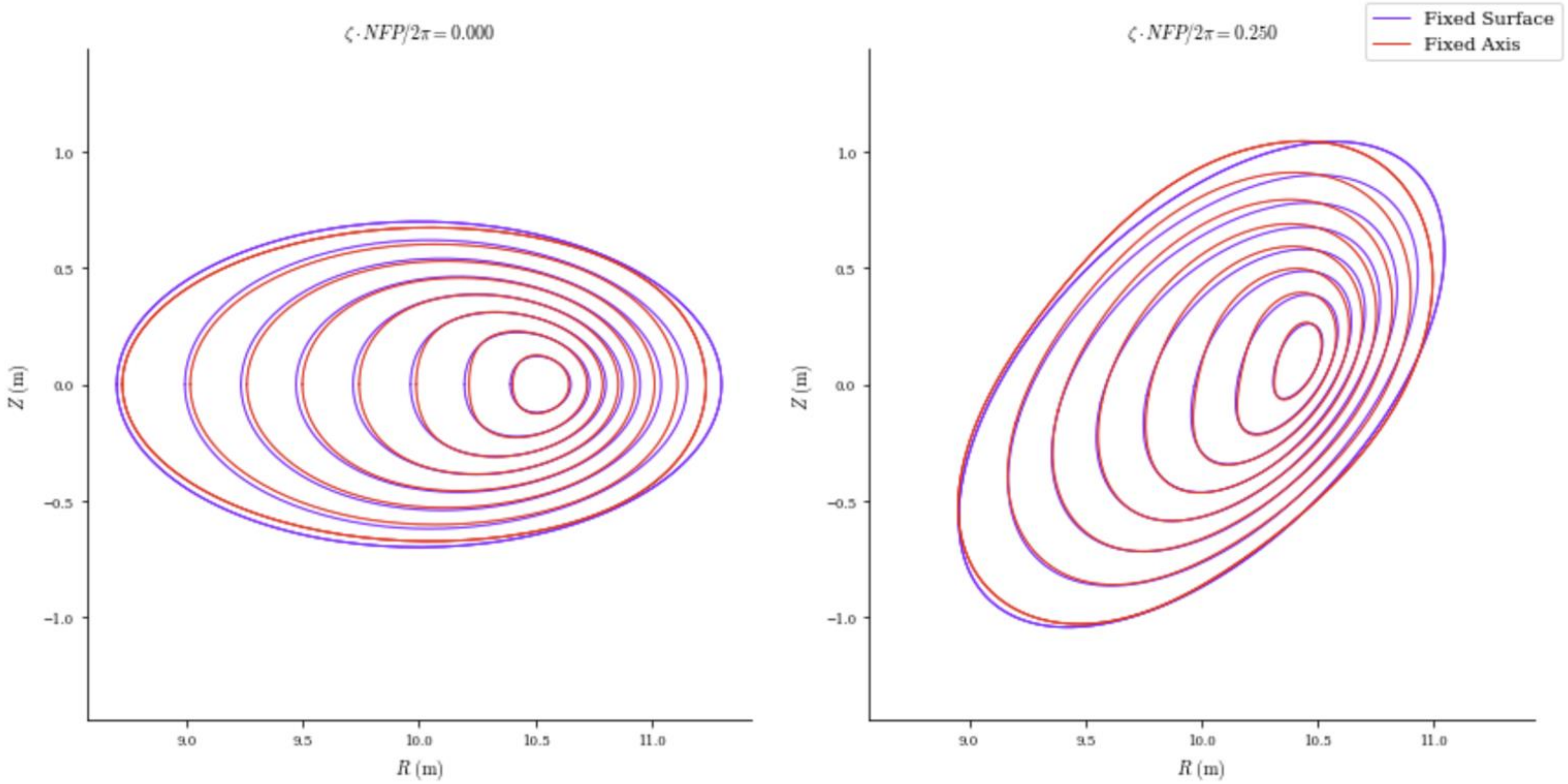
$$\sum_{l=0}^{L} R_{lmn}\mathcal{Z}_l^m(\rho=1,\theta) = R^b_{mn} \quad \forall m,n$$

$$\sum_{l=0}^{L} Z_{lmn}\mathcal{Z}_l^m(\rho=1,\theta) = Z^b_{mn} \quad \forall m,n$$

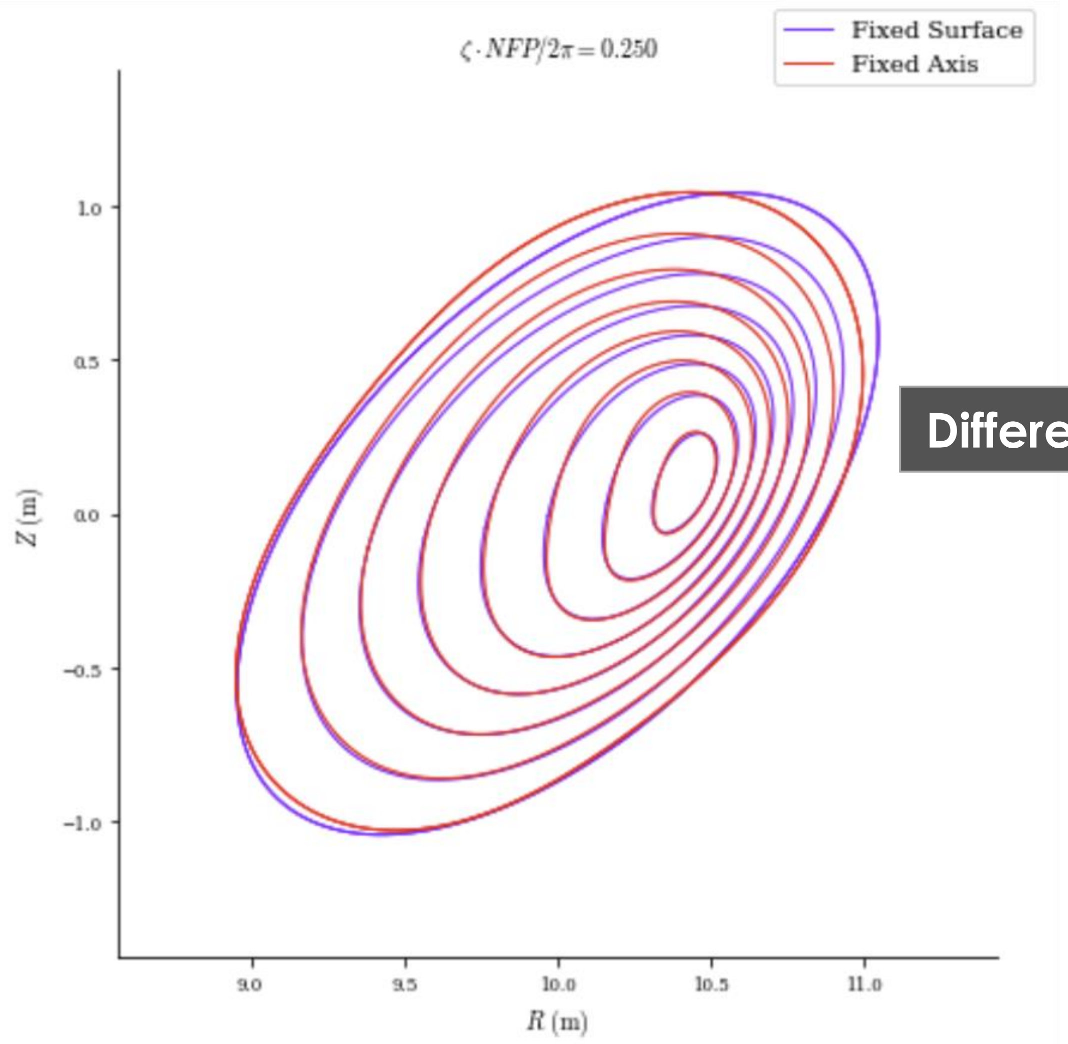


Plasma Control

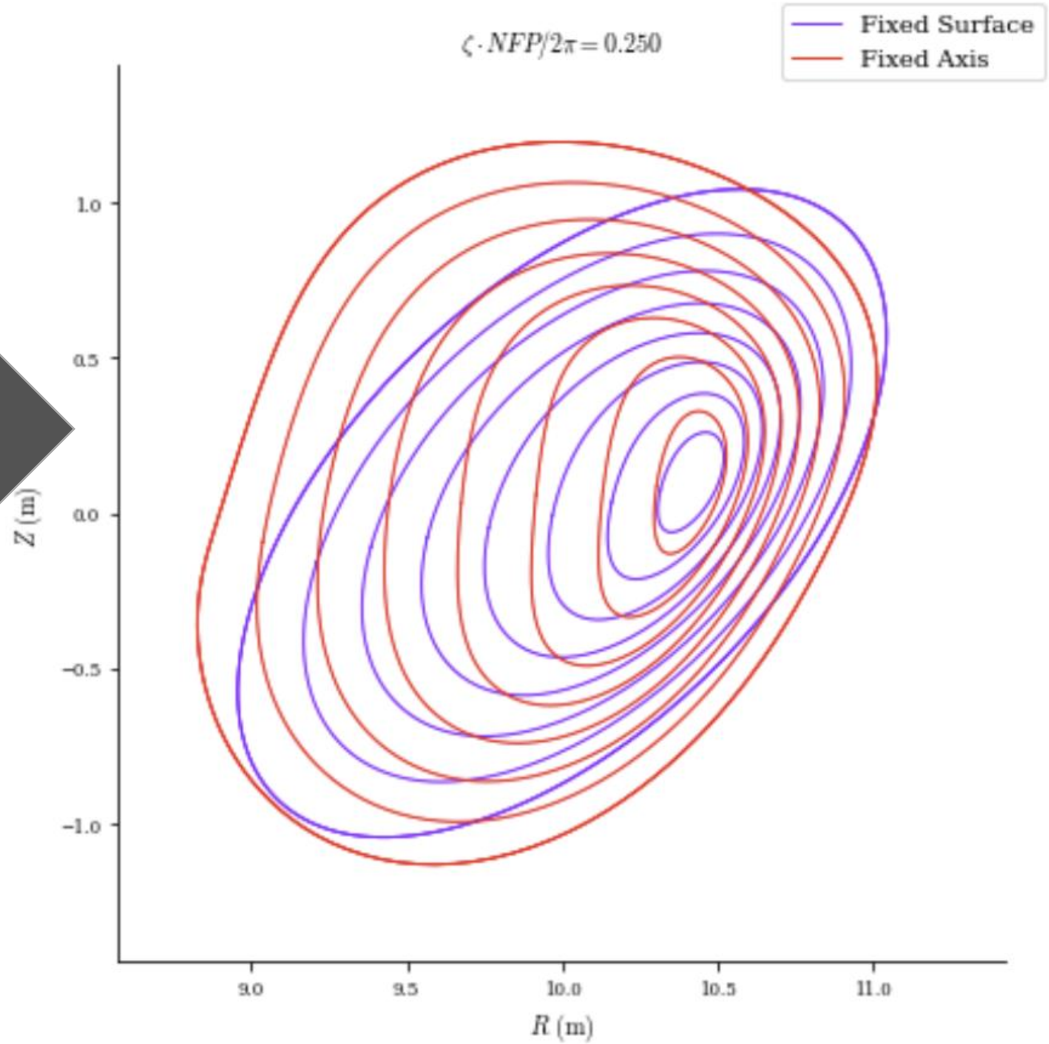
O(ρ^0) (axis) Constraint in DESC - Example Solve



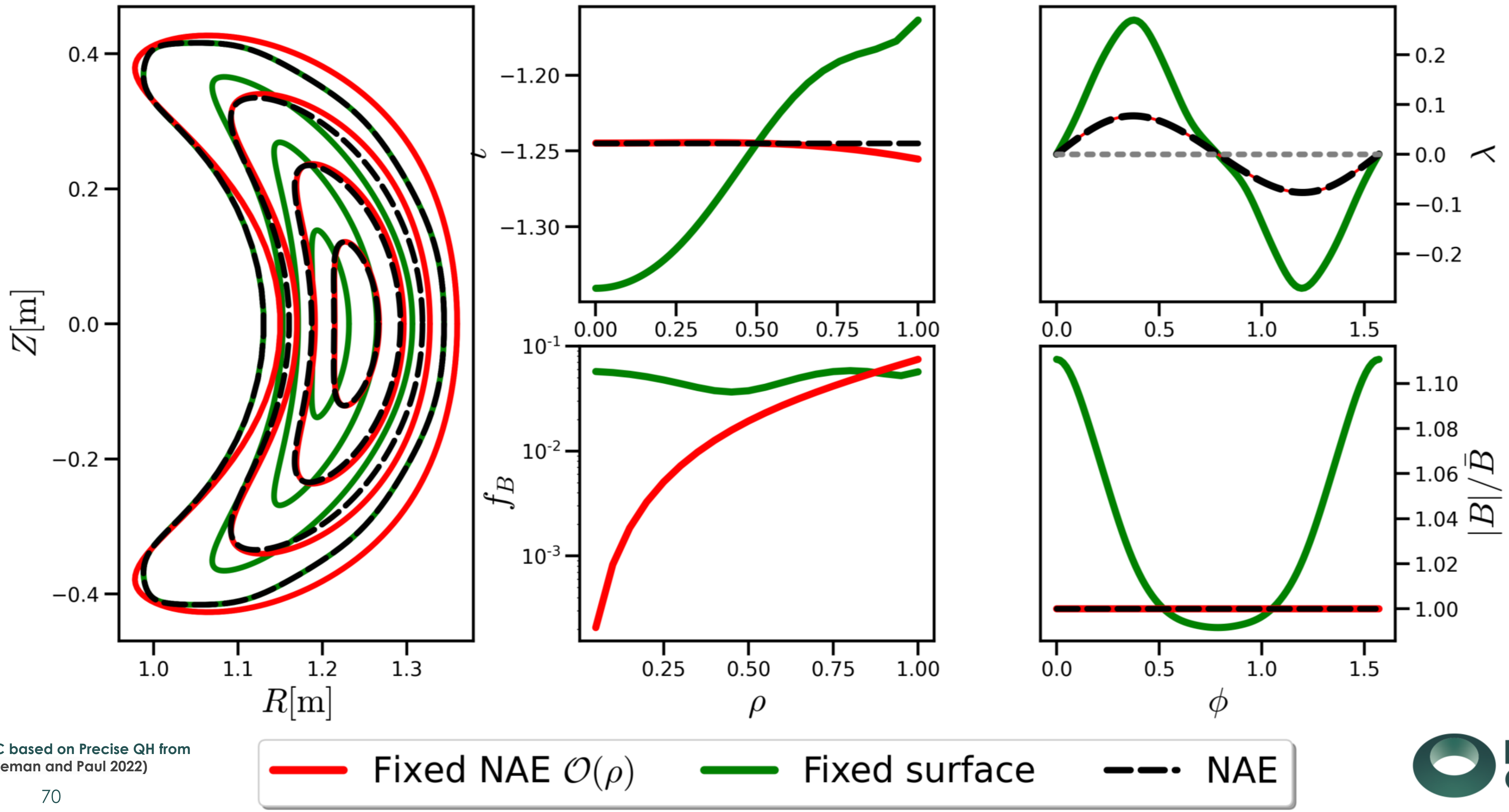
O(ρ^0) (axis) Constraint in DESC - Under-constrained Problem, Finds Closest Equilibrium



Different Initial Guess



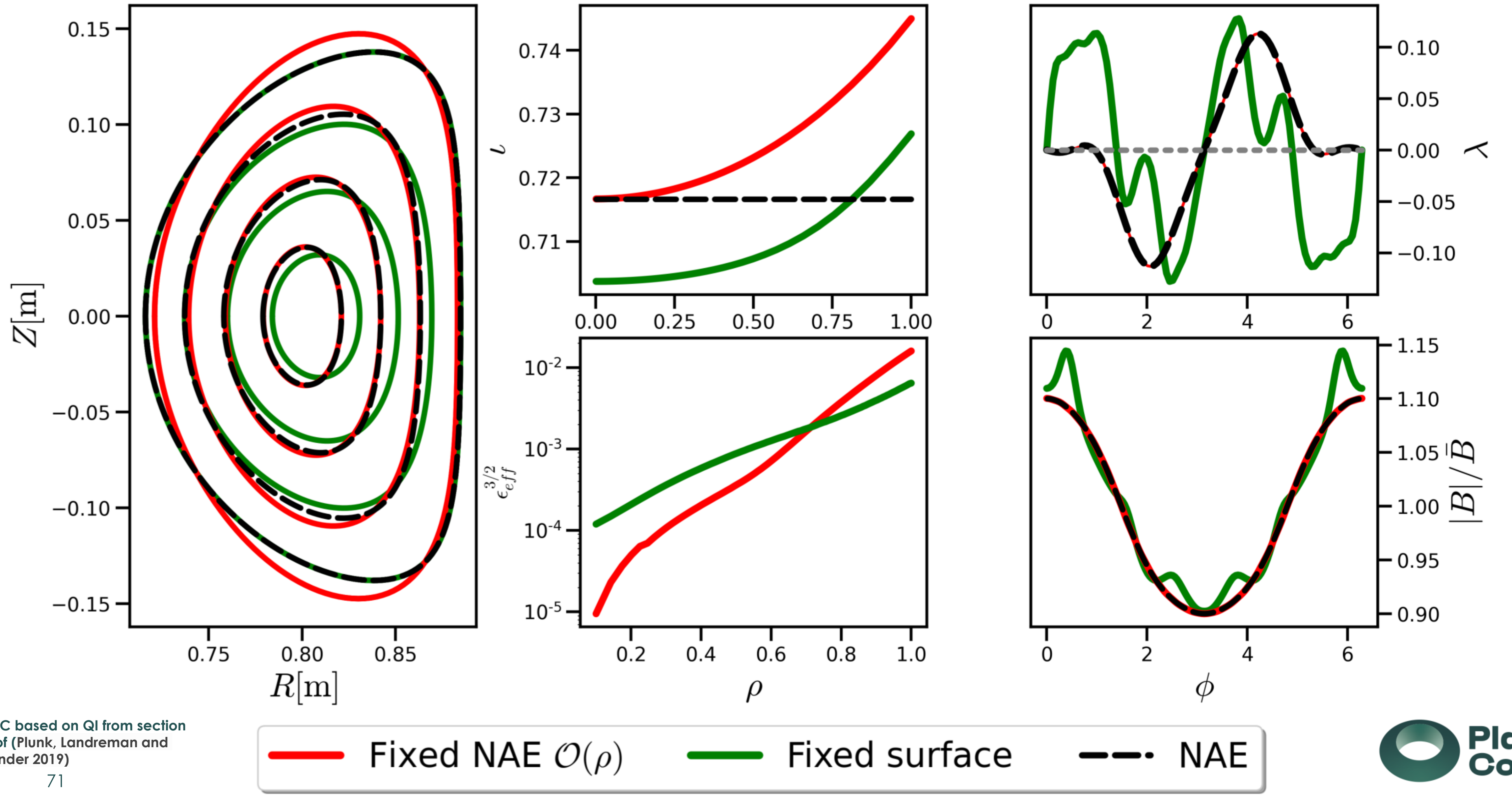
QH O(ρ^1) NAE-constrained Equilibrium



pyQSC based on Precise QH from (Landreman and Paul 2022)

Plasma Control

QI O(ρ^1) NAE-constrained Equilibrium

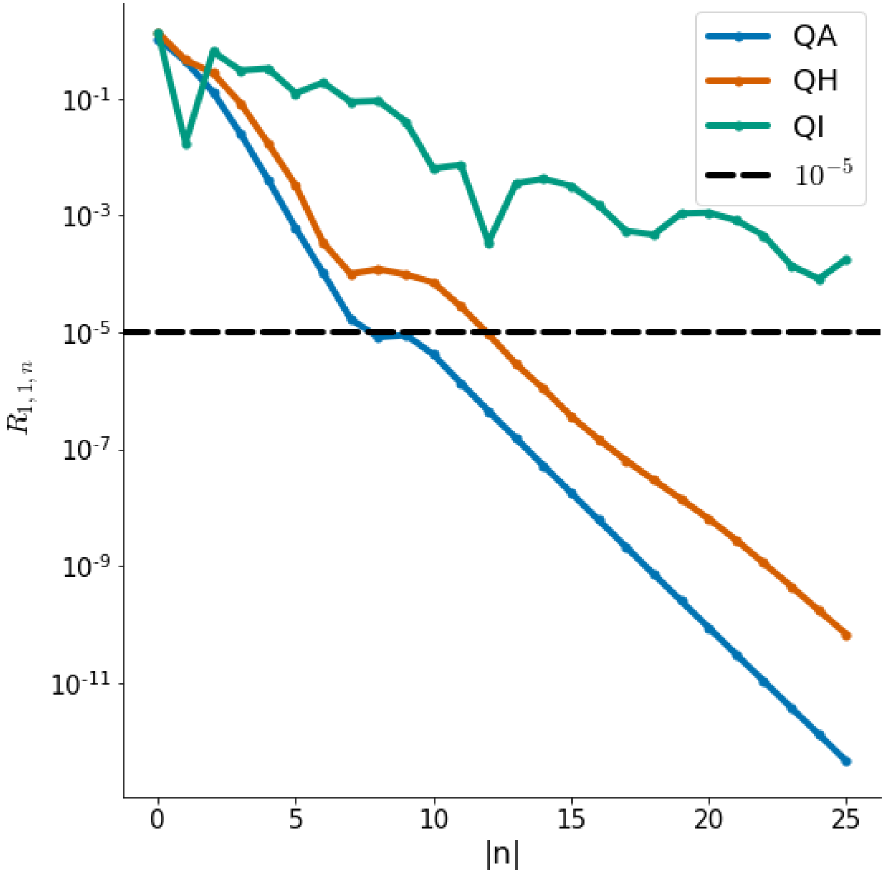


pyQIC based on QI from section 8.2 of (Plunk, Landreman and Helander 2019)

Fitting NAE Behavior with Toroidal Fourier Series – O(ρ^1)

O(ρ) Coefficient $R_{1,1,n}$

$$R_1 = \mathcal{R}_{1,1}(\phi)\cos\theta + \mathcal{R}_{1,-1}(\phi)\sin\theta$$

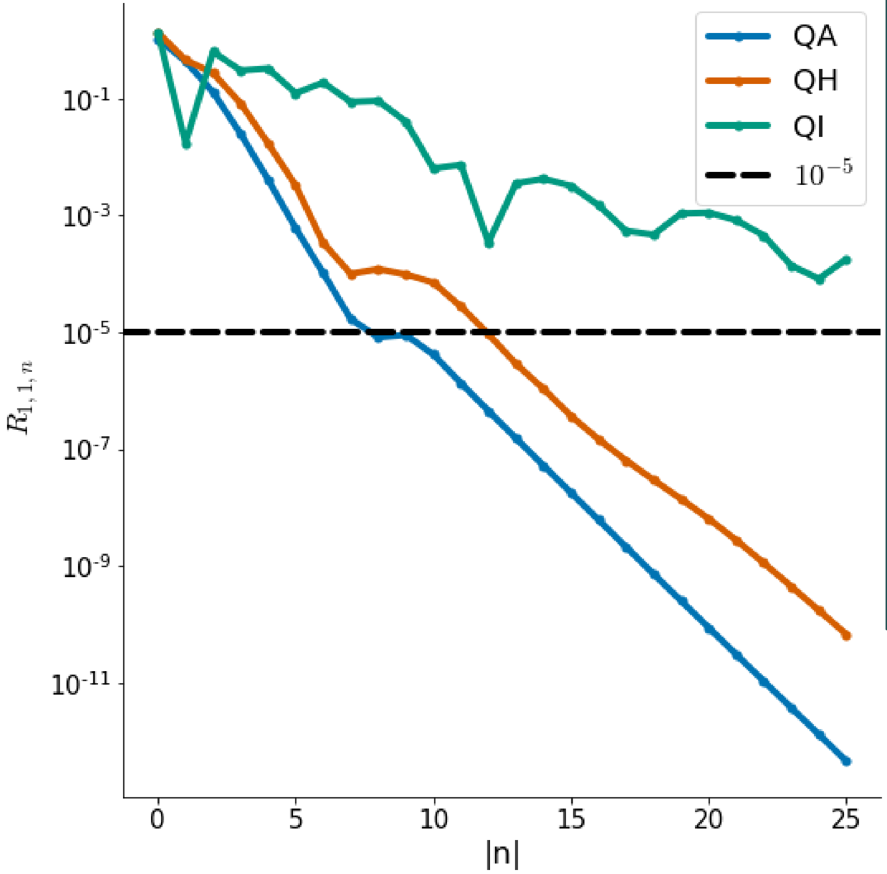


QA NAE behavior simplest to describe

QI NAE behavior very difficult to describe with cylindrical angle!

Fitting NAE Behavior with Toroidal Fourier Series – O(ρ^2)

O(ρ) Coefficient $R_{1,1,n}$

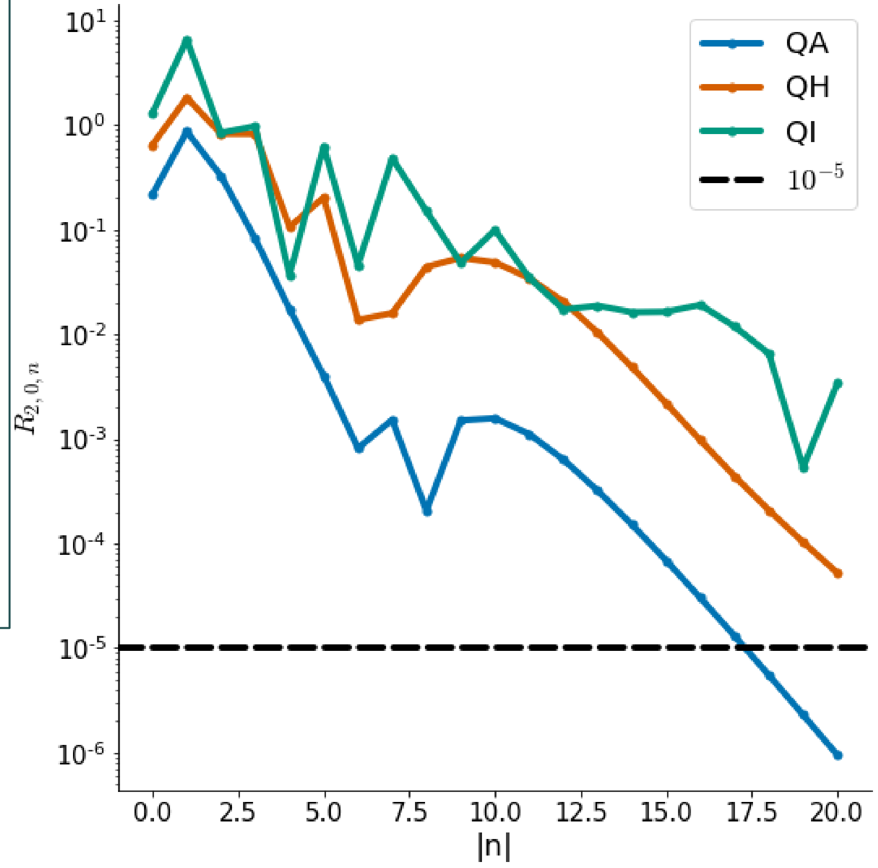


O(ρ) NAE behavior much easier to describe with cylindrical angle than O(ρ^2)

O(ρ^2) – constrained equilibria may require very high N_{tor} to capture behavior accurately

Generalized toroidal angle may help condense spectrum

O(ρ^2) Coefficient $R_{2,0,n}$

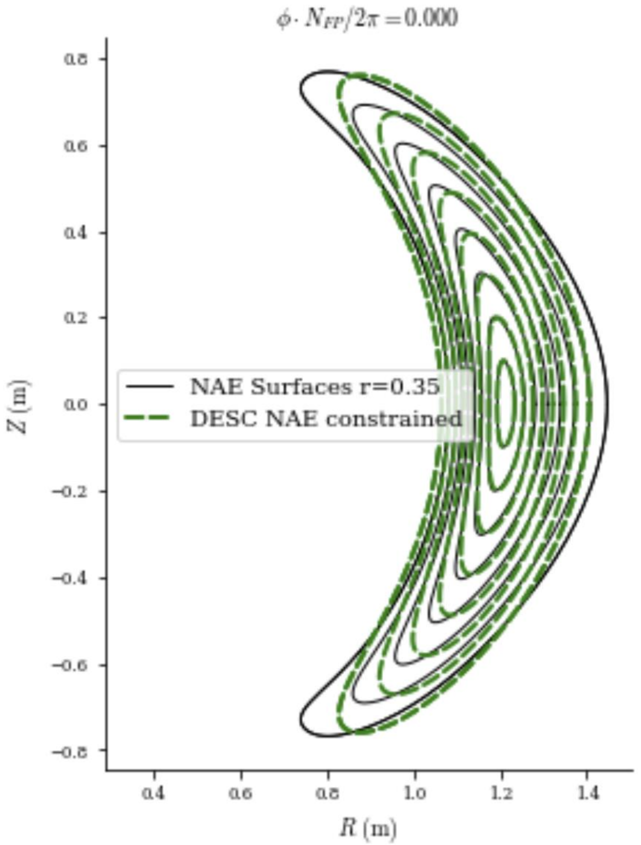


$$R_2 = \mathcal{R}_{2,0}(\phi) + \mathcal{R}_{2,2}(\phi)\cos 2\theta + \mathcal{R}_{2,-2}(\phi)\sin 2\theta$$

Plasma Control

Example Python Code for NAE-Constrained Equilibria in DESC – Simple!

```python
# imports
from qsc import Qsc
from desc.objectives import get_NAE_constraints
from desc.equilibrium import Equilibrium

# fit initial DESC equilibrium from desired qsc solution
qsc = Qsc.from_paper('precise QA',nphi=99)
desc_eq = Equilibrium.from_near_axis(qsc,r=0.35,L=9,M=9,N=10)

# get constraints on axis and O(r) coefficients
# to pass to eq.solve using utility function
cs = get_NAE_constraints(desc_eq, qsc, order=1)

# solve NAE-constrained equilibrium
desc_eq.solve(objective="force",constraints=cs);
```



Tutorial on DESC documentation website:

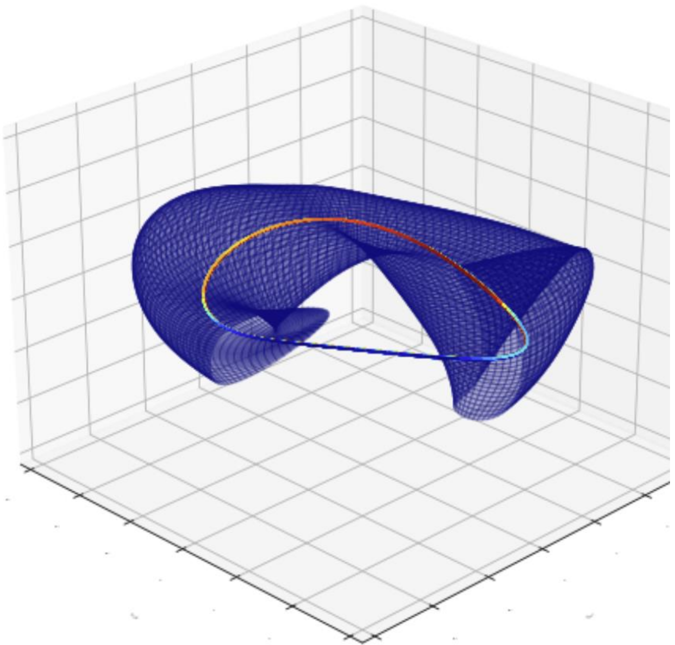
desc-docs.readthedocs.io

Plasma Control

Physical Insights Yield Constraints on XS or near Axis

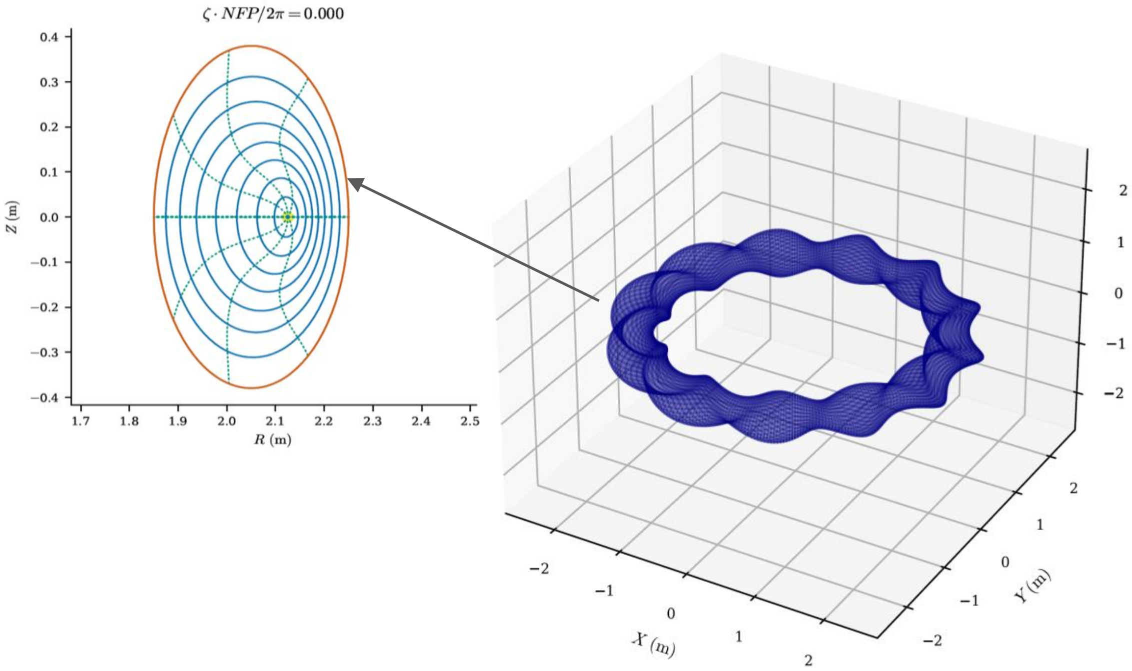
Axis + Near-Axis Behavior

Near-Axis Expansion (NAE) yields what asymptotic behavior of equilibrium should be near the axis, and what the axis shape should be



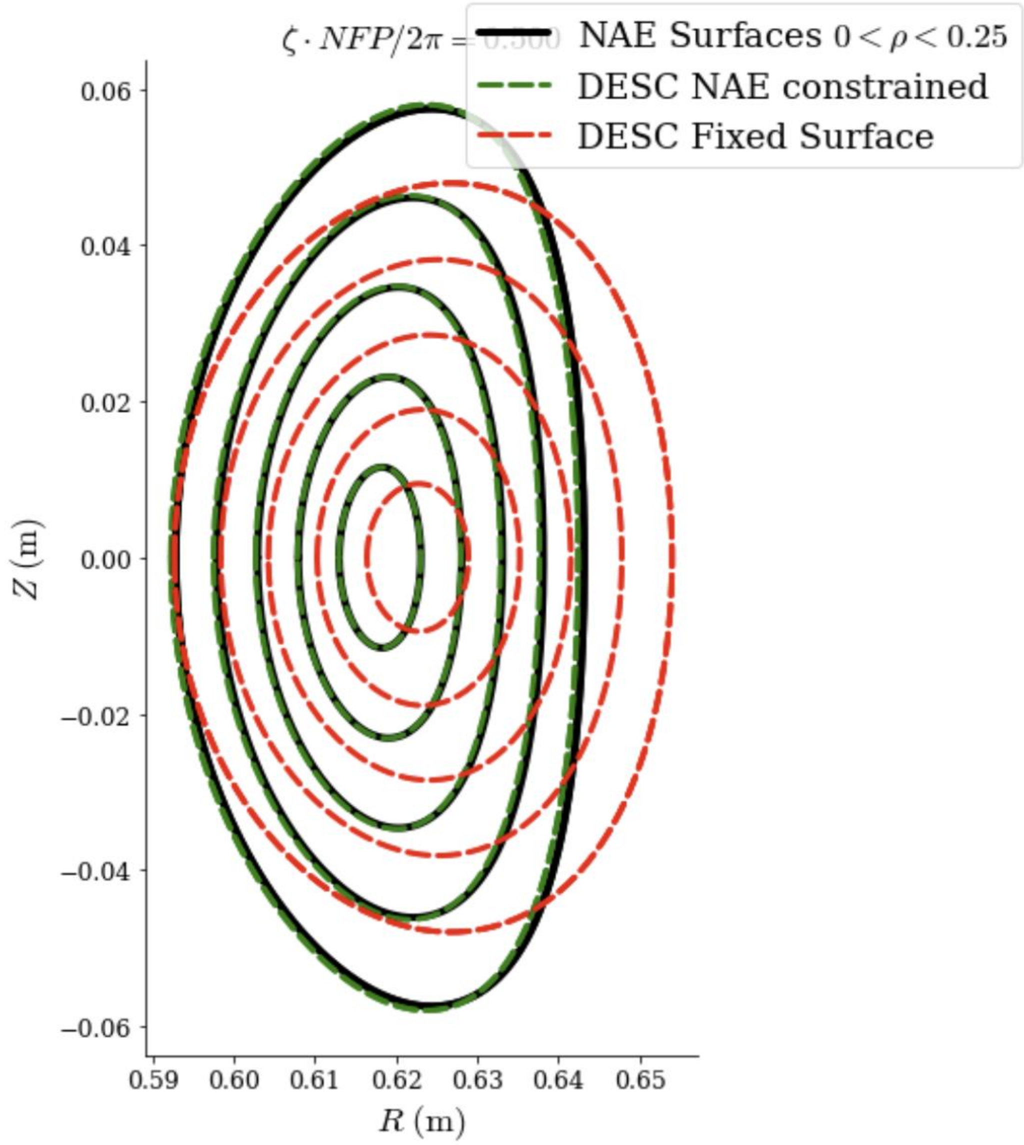
Poincare Section

Desire to avoid magnetic islands, and decoupling poloidal and toroidal resolution



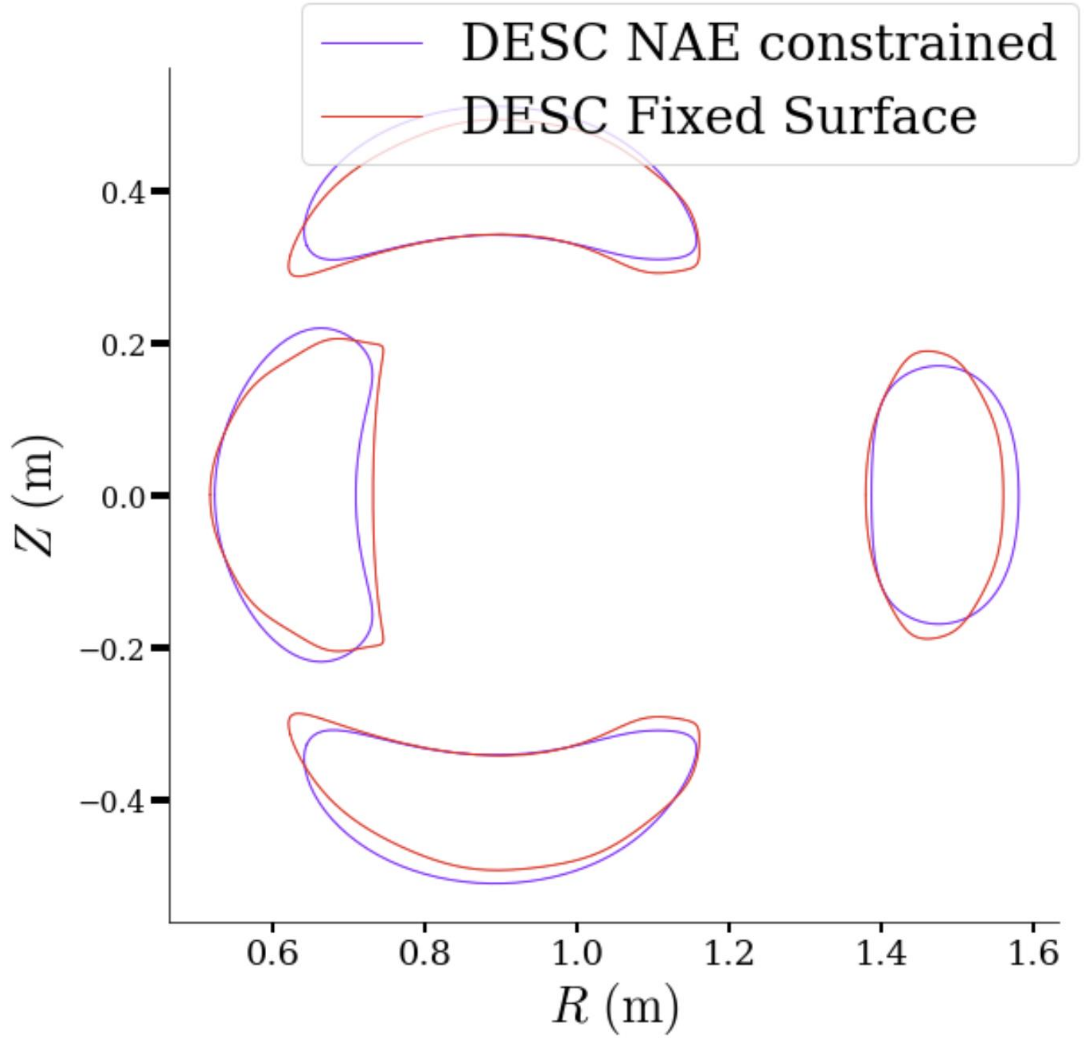
Closer look at flux surfaces near axis for difficult NAE

```
rc = [1, 0.426, 0.044, -6.3646383583351e-11,
2.851584586653665e-05, 3.892992983405039e-08]

zs = [0.0, 0.4110168175146285, 0.04335427796015756,
6.530936323433338e-05, 1.3623898672936873e-05,
1.1620514629503932e-05]

etabar=1.64209358
B2c = 0.11293987662545873
B0=1
nfp = 4

qsc = Qsc(rc=rc, zs=zs, B0=B0, nfp=nfp, I2=0, B2c = B2c,
etabar=etabar, order = "r1", nphi = 201)

desc_eq= Equilibrium.from_near_axis(qsc,r=
r,L=9,M=9,N=N,ntheta=ntheta)
```



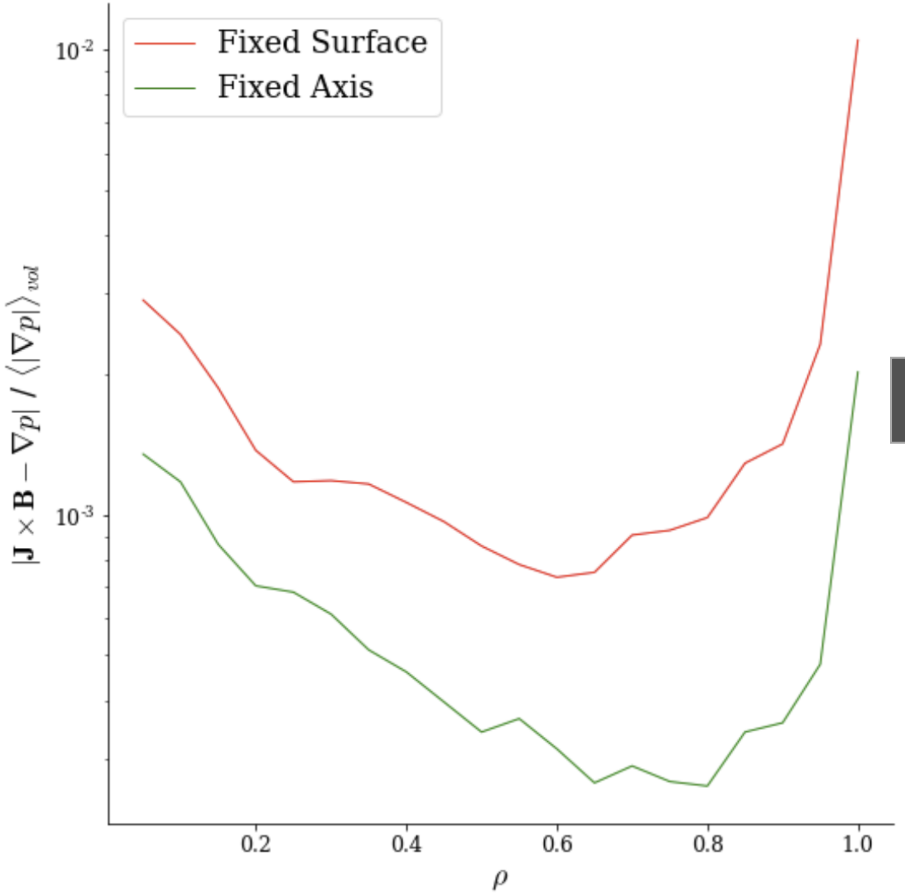
Closer look at LCFS for difficult NAE

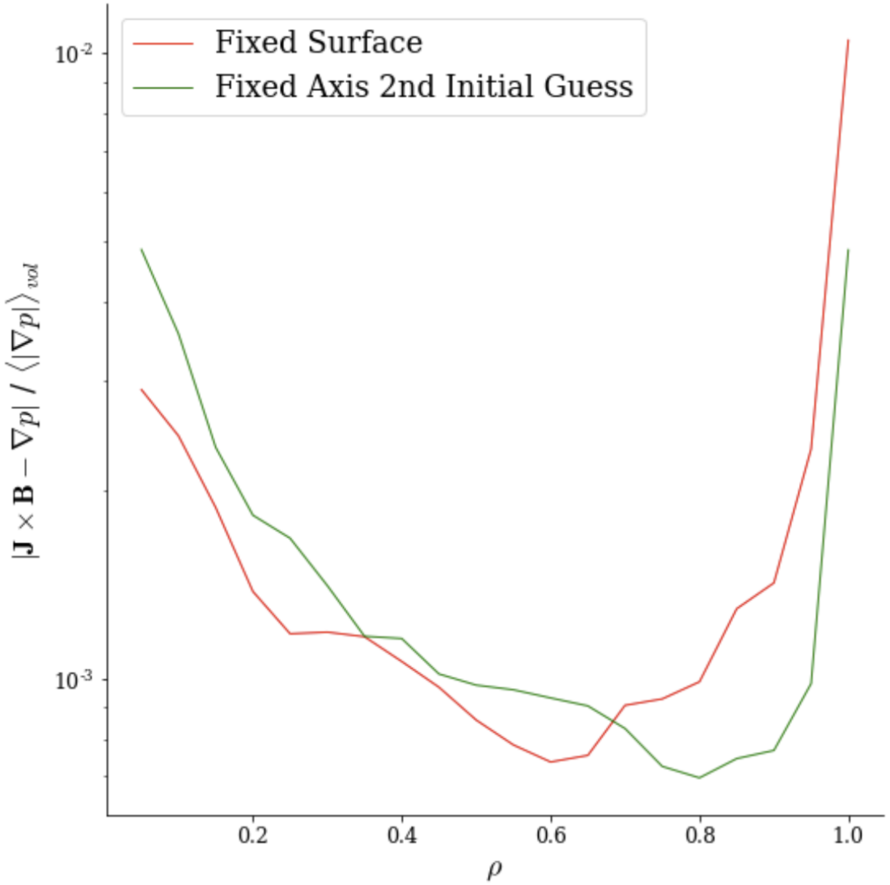
```
rc = [1, 0.426, 0.044, -6.3646383583351e-11,
2.851584586653665e-05, 3.892992983405039e-08]

zs = [0.0, 0.4110168175146285, 0.04335427796015756,
6.530936323433338e-05, 1.3623898672936873e-05,
1.1620514629503932e-05]

etabar=1.64209358
B2c = 0.11293987662545873
B0=1
nfp = 4

qsc = Qsc(rc=rc, zs=zs, B0=B0, nfp=nfp, I2=0, B2c = B2c,
etabar=etabar, order = "r1", nphi = 201)

desc_eq= Equilibrium.from_near_axis(qsc,r=
r,L=9,M=9,N=N,ntheta=ntheta)
```



Plasma Control

O(ρ^0) (axis) Constraint in DESC - Under-constrained Problem





Plasma Control

DESC Allows Flexible Constraints when Defining Equilibrium Problem - Fixed ρ=1 Boundary

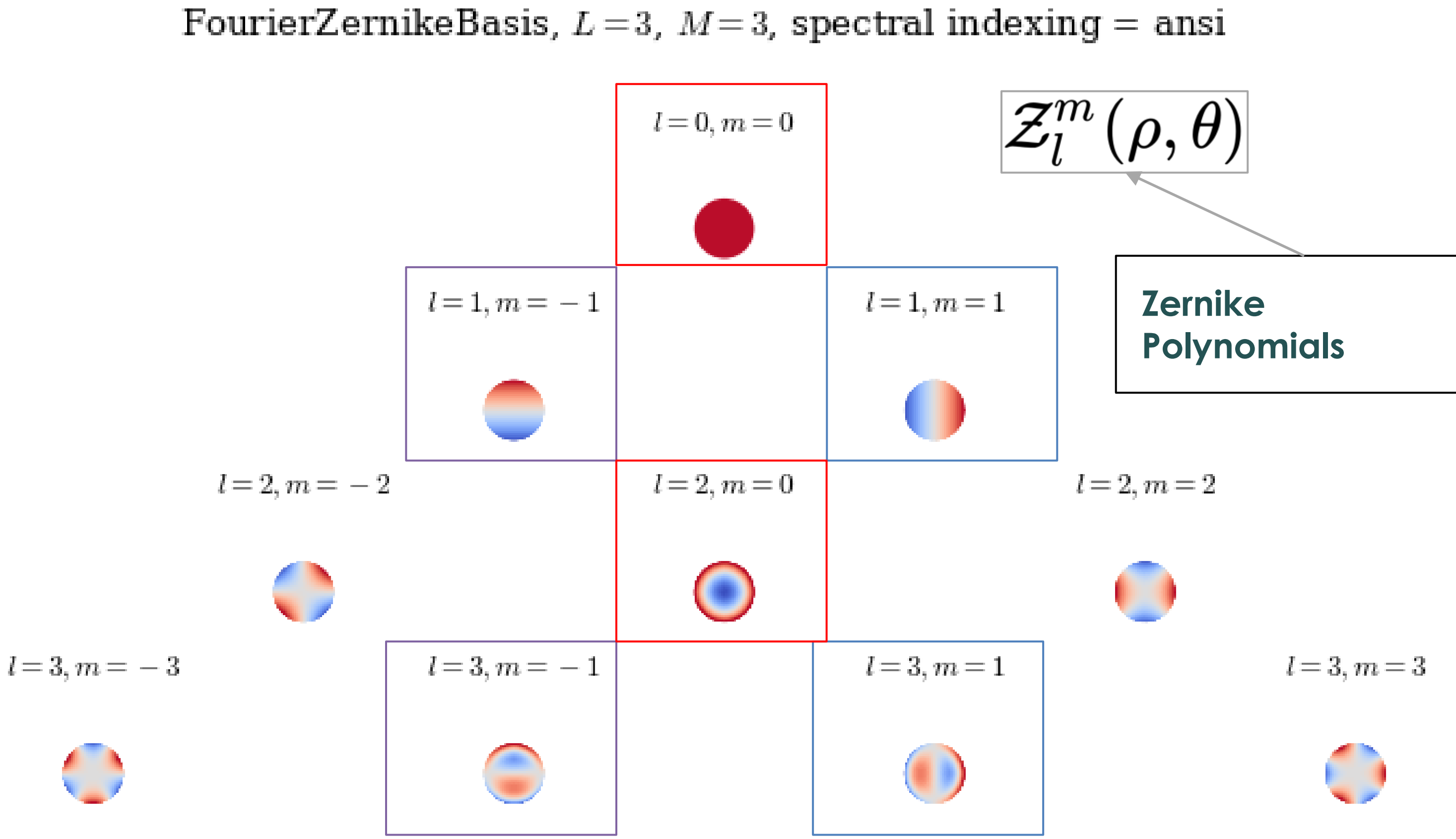
$$R^b(\theta,\zeta) = \sum_{m=0}^{M} \sum_{n=-N}^{N} R^b_{m,n} cos(m\theta - n\zeta)$$

$$Z^b(\theta,\zeta) = \sum_{m=0}^{M} \sum_{n=-N}^{N} Z^b_{m,n} sin(m\theta - n\zeta)$$

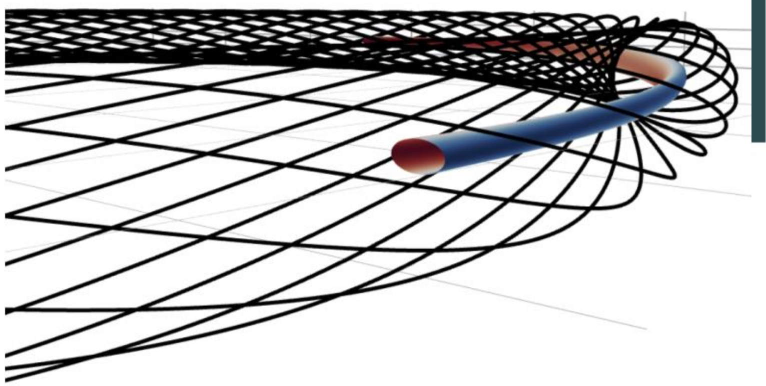
Fixed-Boundary ρ=1 Constraint

$$\sum_{l=0}^{L} R_{lmn} \mathcal{Z}_l^m(\rho = 1, \theta) = R^b_{mn} \quad \forall m, n$$

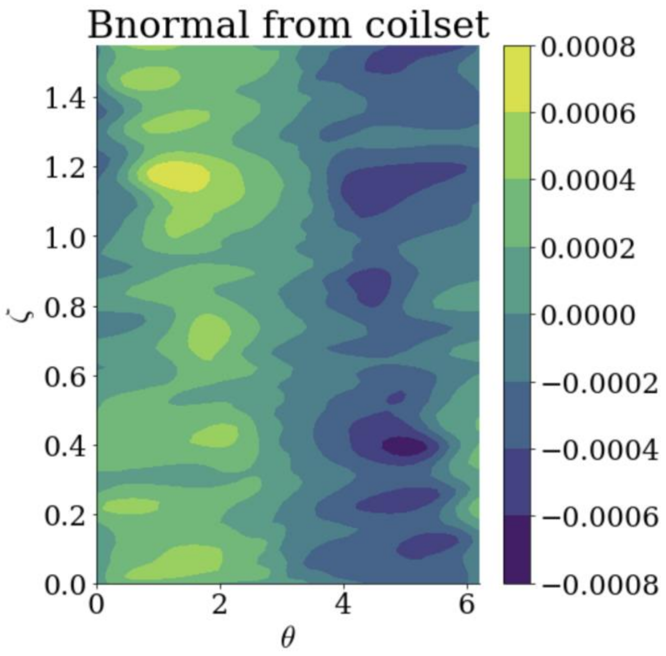
$$\sum_{l=0}^{L} Z_{lmn} \mathcal{Z}_l^m(\rho = 1, \theta) = Z^b_{mn} \quad \forall m, n$$



Example Python Code to create Helical Coilset



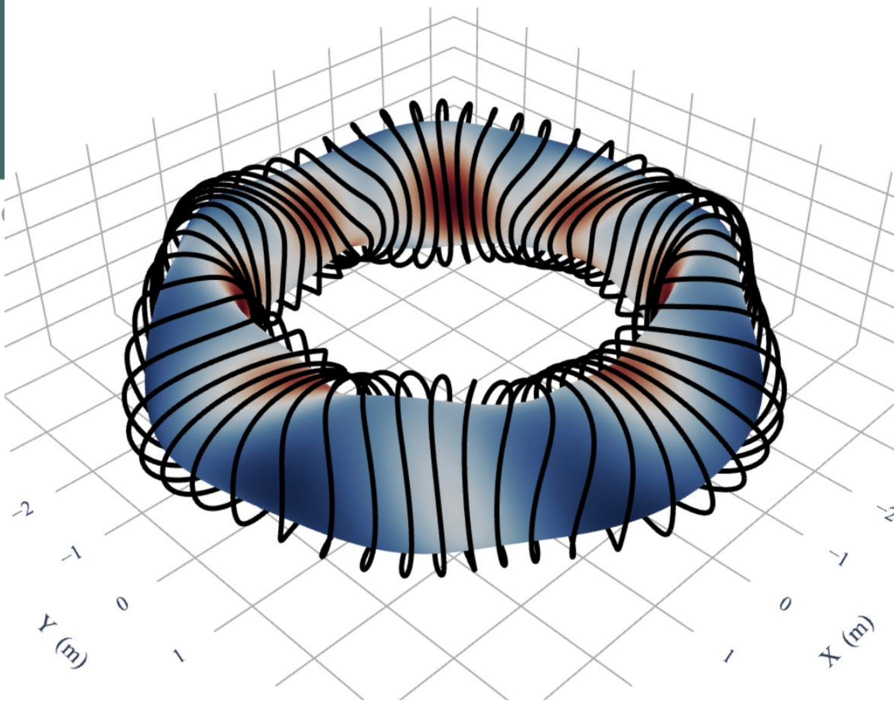
```
# load equilibrium, this case is a simple vacuum rotating ellipse
eqname = "./tests/inputs/ellNFP4_init_smallish.h5"
eq = load(eqname)

# get the surface current which minimizes Bn with REGCOIL algorithm
(surface_current_field, _, _, _, _,) = run_regcoil(
    eqname=eq,
    # resolutions of plasma surface grid upon which Bn is minimized
    eval_grid_M=20,eval_grid_N=20,
    # resolutions of source grid for calculating Bn
    source_grid_M=40, source_grid_N=80,
    alpha=1e-15, # regularization parameter
    # ratio of I/G, 0 for modular, integer for helical coils
    helicity_ratio=-1)

# discretize into helical coils using utility function
numCoils = 15 # we want 15 helical coils
coilset2 = find_helical_coils(surface_current_field, eqname, desirednumcoils=numCoils)
```

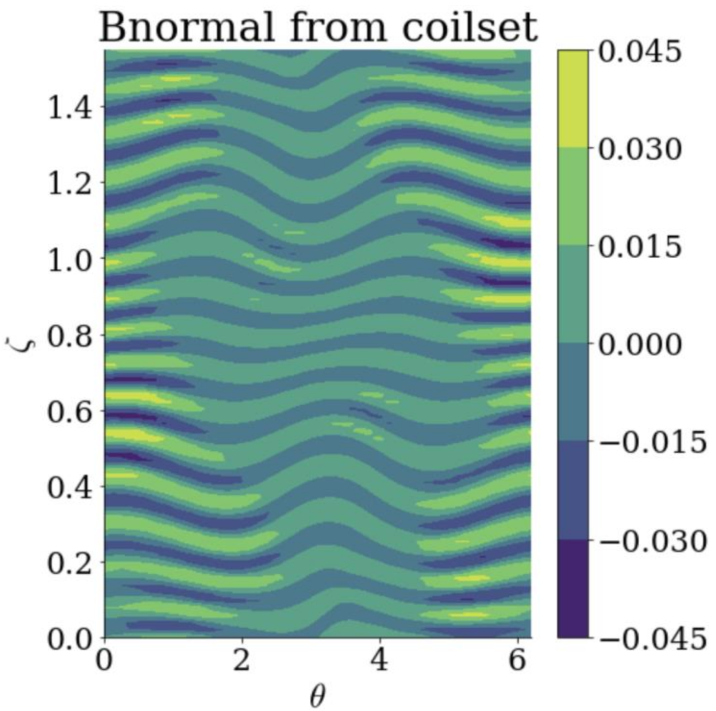


Plasma Control

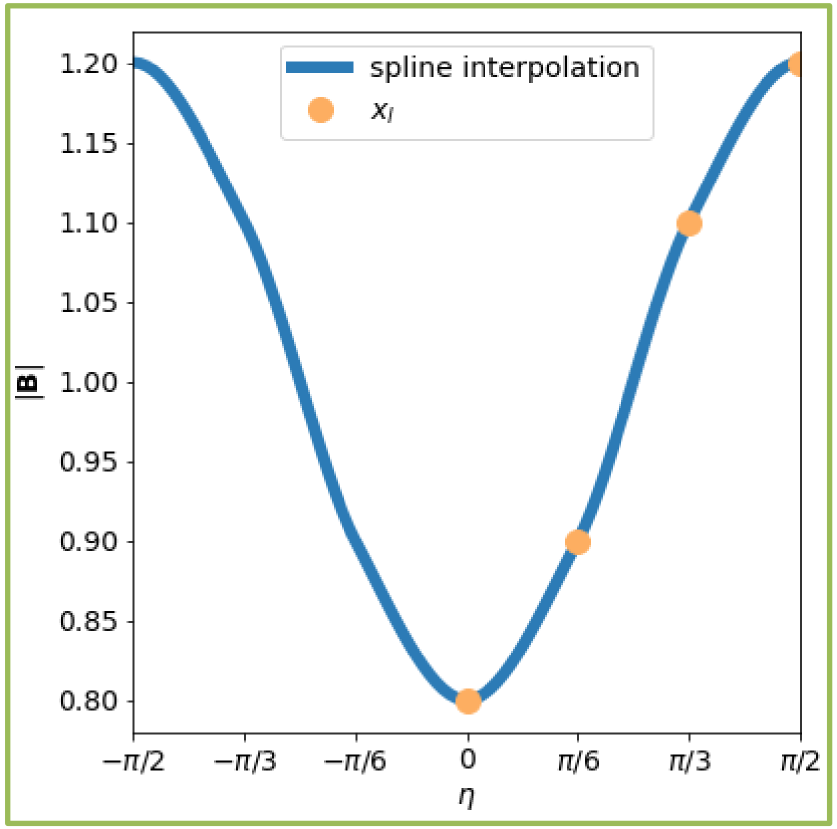
Example Python Code to create Modular Coilset

```
# load equilibrium, this case is a simple vacuum rotating ellipse
eqname = "rotating_ellipse_5_aspect_ratio.h5"
eq = load(eqname)
winding_surf= load("rotating_ellipse_wind_surf.h5")
# get the surface current which minimizes Bn with REGCOIL alg
(surface_current_field, _, _, _, _,) = run_regcoil(
    eqname=eq, basis_M=16, basis_N=16,
# resolutions of plasma surface grid upon which Bn is minimized
    eval_grid_M=60,eval_grid_N=60,
    # resolutions of source grid for calculating Bn
    source_grid_M=100, source_grid_N=100,
    alpha=1e-16, # regularization parameter
    # ratio of I/G, 0 for modular, integer for helical coils
    helicity_ratio=0, winding_surf = winding_surf)

# discretize into modular coils using utility function
numCoils = 60 # we want 60 modular coils, (15 per field period)
coilset2 = find_modular_coils(surface_current_field,
                              eqname, desirednumcoils=numCoils)
```



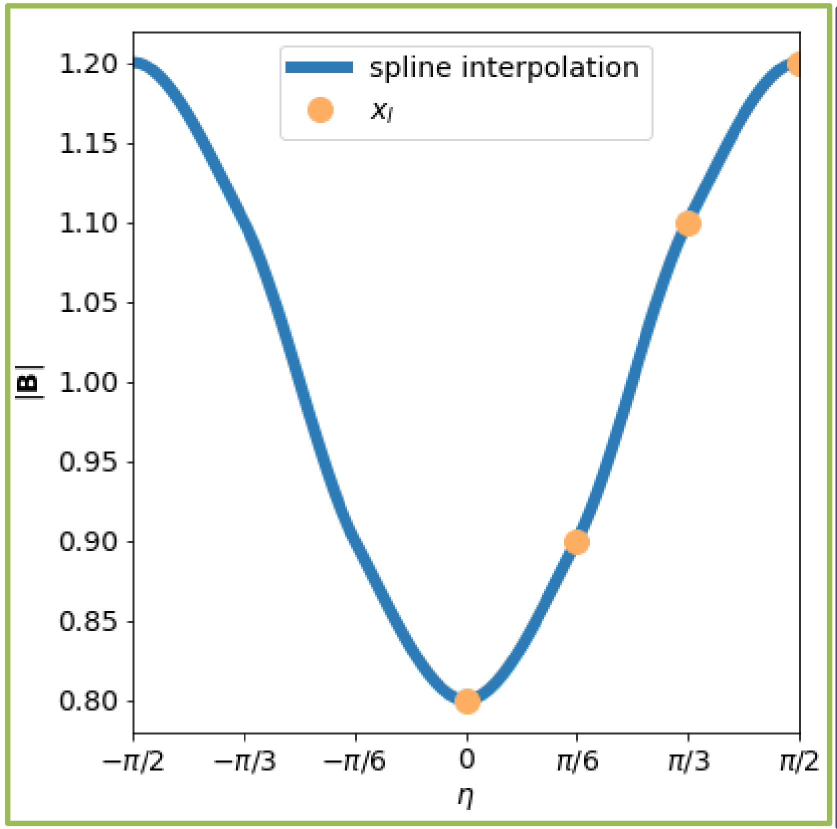


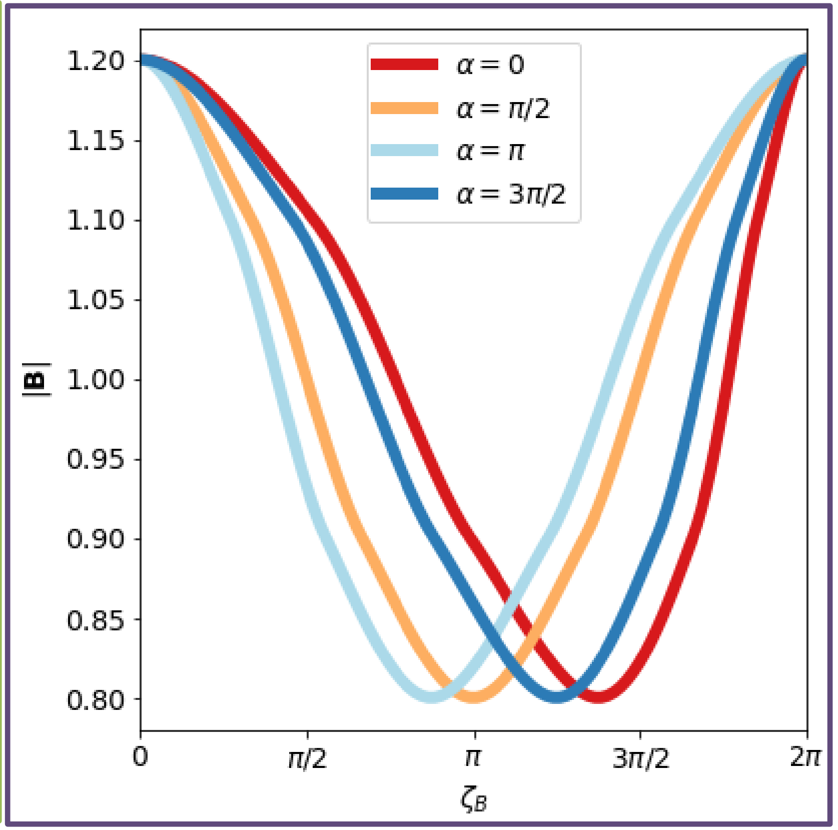
Example: Full QI Phase Space Definition in DESC



- **Specify the magnetic well “shape” in computational coordinate η**

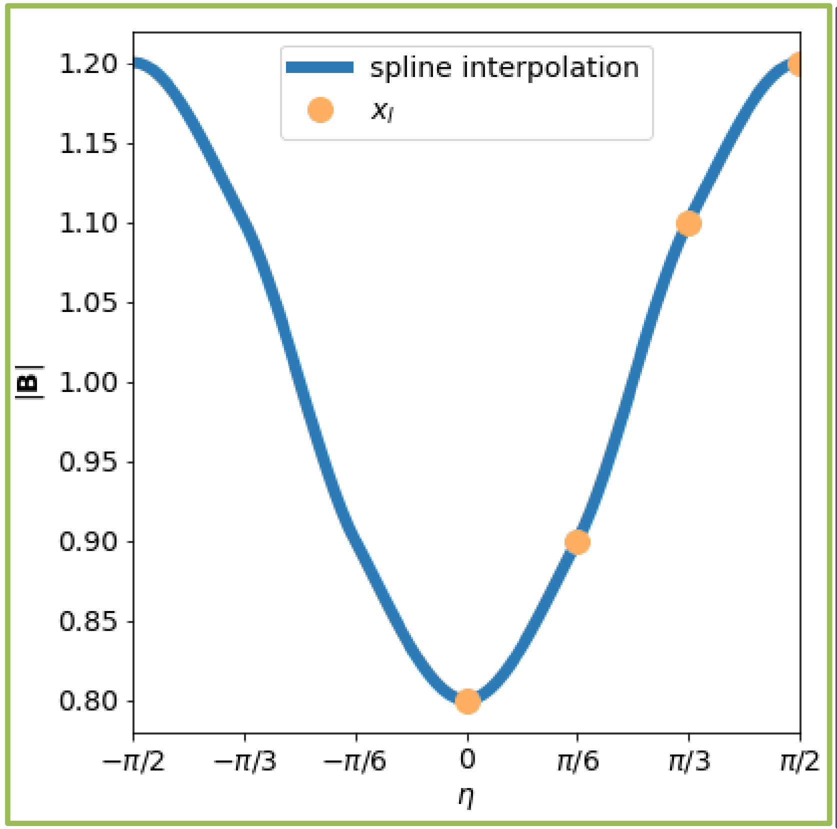
Example: Full QI Phase Space Definition in DESC

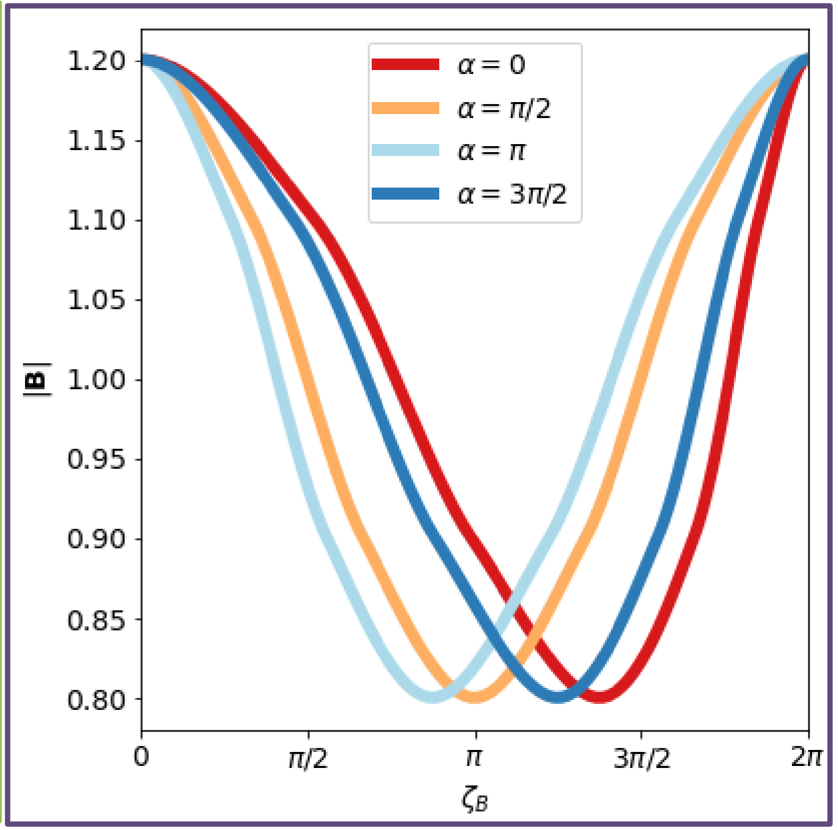


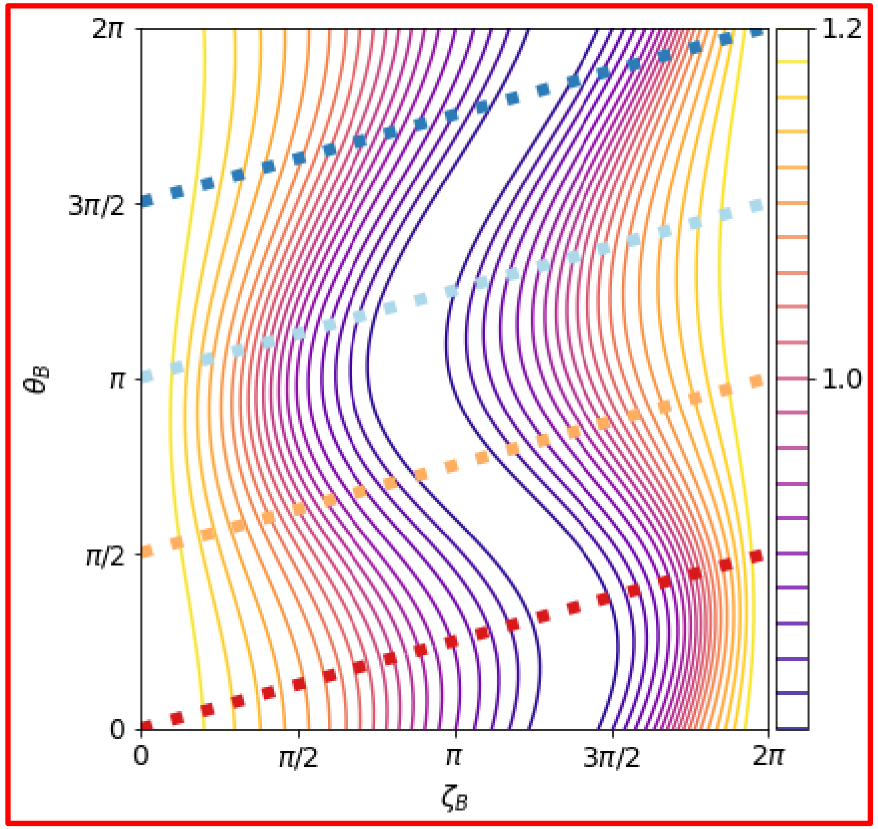


- **Specify the magnetic well "shape" in computational coordinate η**
- **Specify how the well "shifts" on different field lines with a Fourier series x_{mn} in (η, α)**

Example: Full QI Phase Space Definition in DESC

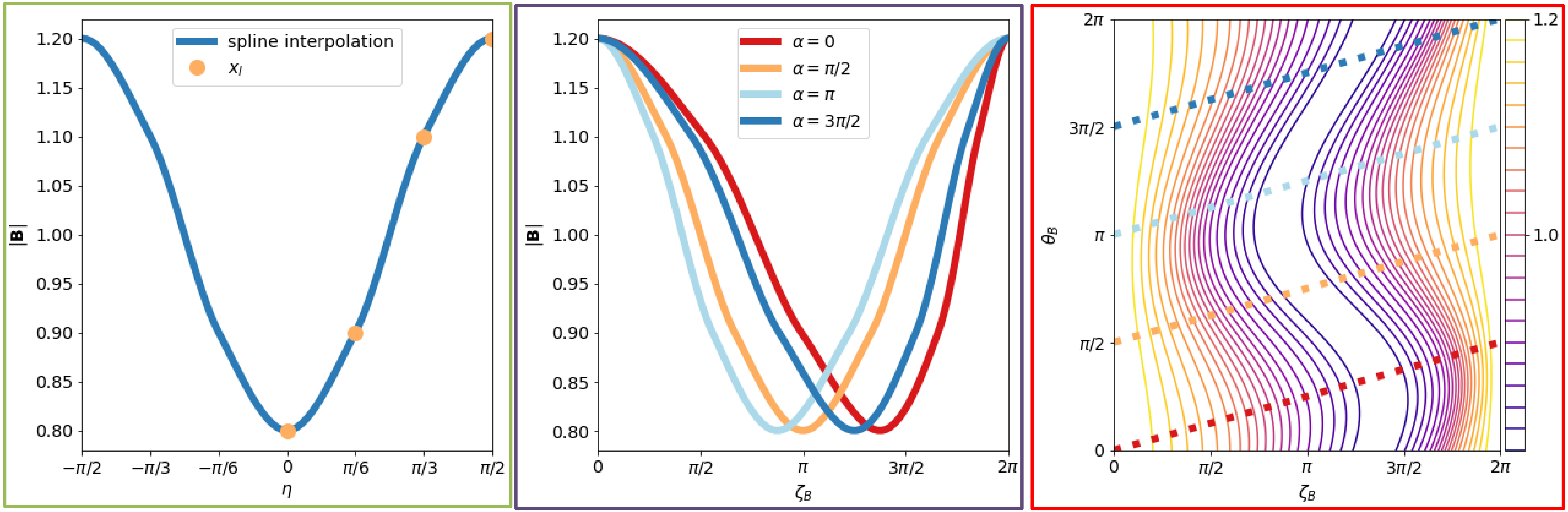






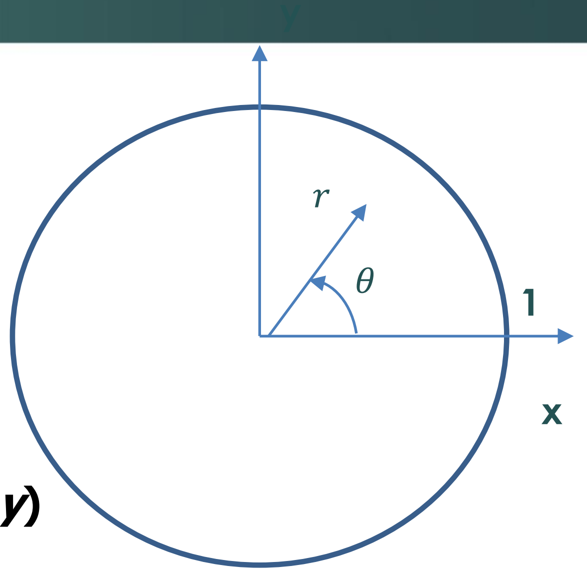
- Specify the magnetic well "shape" in computational coordinate η
- Specify how the well "shifts" on different field lines with a Fourier series x_{mn} in (η, α)
- Generate arbitrary QI magnetic field targets without prior initialization

Example: Full QI Phase Space Definition in DESC



- Specify the magnetic well "shape" in computational coordinate η
- Specify how the well "shifts" on different field lines with a Fourier series x_{mn} in (η, α)
- Generate arbitrary QI magnetic field targets without prior initialization
- Model enables scans of the QI optimization landscape

Analyticity Constraint at Polar Axis Proof



- **Assume $f(r,\theta)$ is a physical scalar, regular at r=0**
- **Expand in a Fourier Series: $\sum_{m=-\infty}^{\infty} a_{m(r)} e^{im\theta} = \sum_{-\infty}^{\infty} f_m(r,\theta)$**
  - **Where the Fourier coefficients are a function of polar radius *r***
- **Assume each $f_m(r,\theta)$ is a regular function of *(x,y)* at r=0**
- **Notice that $e^{im\theta}$ is NOT regular at r=0 (it is multi-valued)**
- **But, $\left[re^{\pm im\theta}\right]^{|m|} = [x \pm iy]^{|m|}$ is a regular function of (*x,y*) b/c it is a <u>polynomial</u> in (*x,y*)**
- **We can rewrite $f(r,\theta)$ as**

$$\begin{aligned} f_m(r,\theta) &= a_m(r) e^{im\theta} \\ &= \frac{a_m(r)}{r^{|m|}} r^{|m|} e^{i\theta m} \\ &= \frac{a_m(r)}{r^{|m|}} \left[re^{\pm i\theta}\right]^{|m|} \begin{cases} + & m > 0 \\ - & m < 0 \end{cases} \end{aligned}$$

[1](Lewis and Bellan, 1990)

Regular at r=0

$$\boxed{f_m(r,\theta)} = \boxed{\frac{a_m(r)}{r^{|m|}}} \boxed{[x \pm iy]^{|m|}}$$

Regular at r=0

Must be regular at r=0!

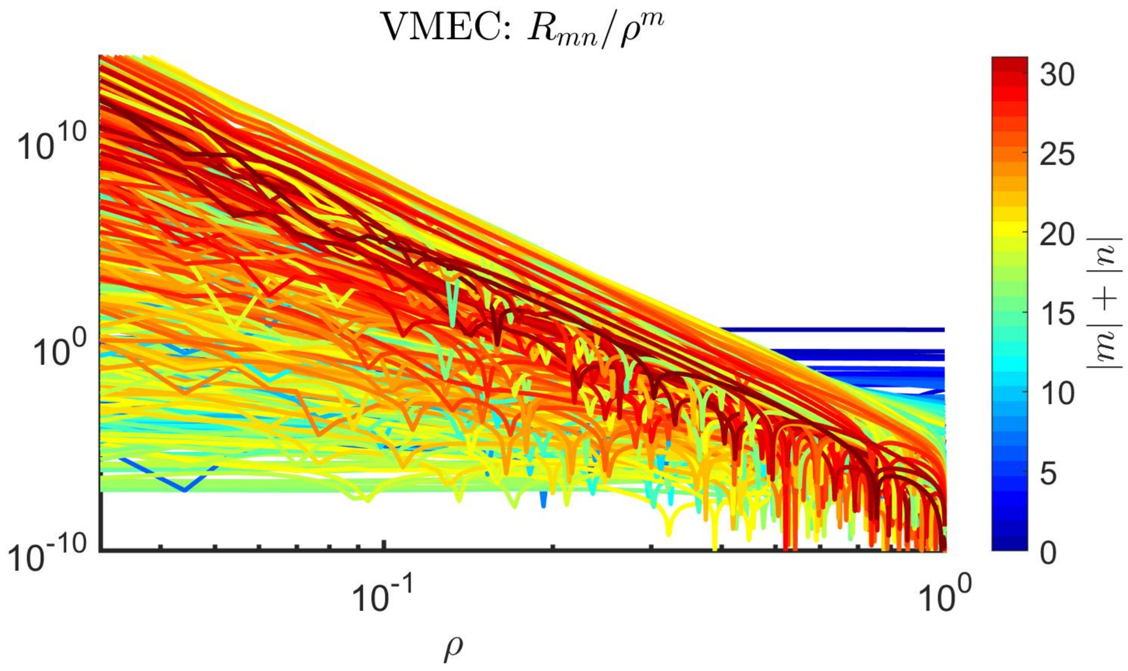
$$\boxed{\lim_{r\to 0} \frac{a_m(r)}{r^{|m|}} < \infty}$$

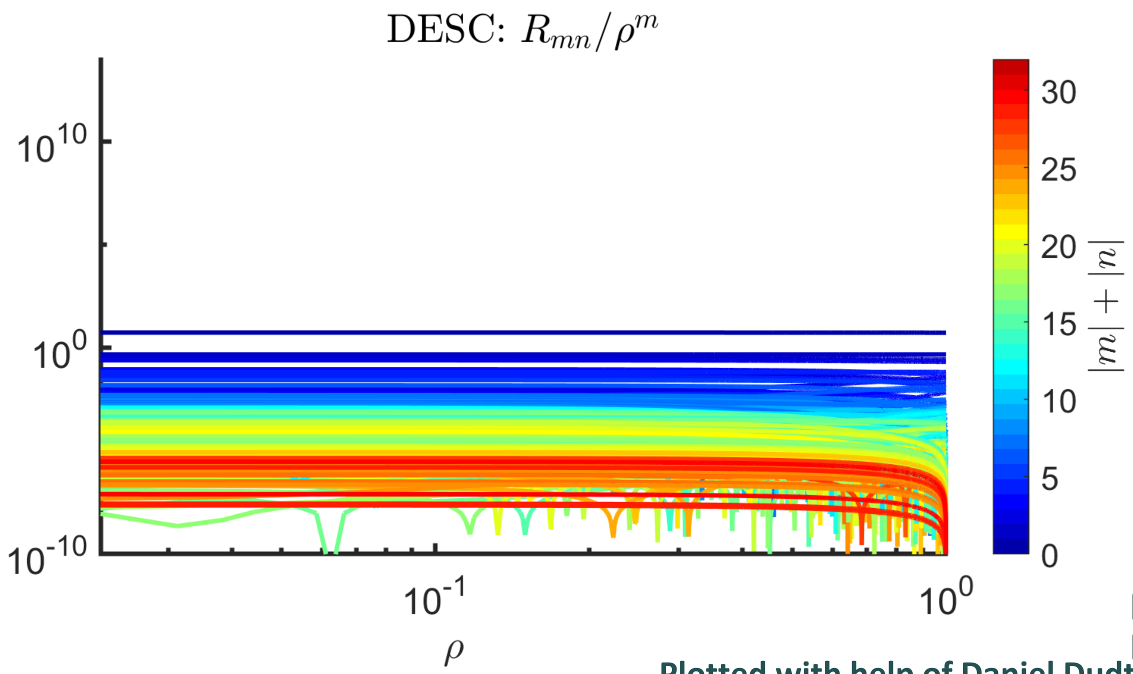
$a_m(r)$ must scale at least as $r^{|m|}$

$a_m(r) \sim r^{|m|} + r^{|m|+2} \ldots$

VMEC and DESC Analytic Constraint Near Axis

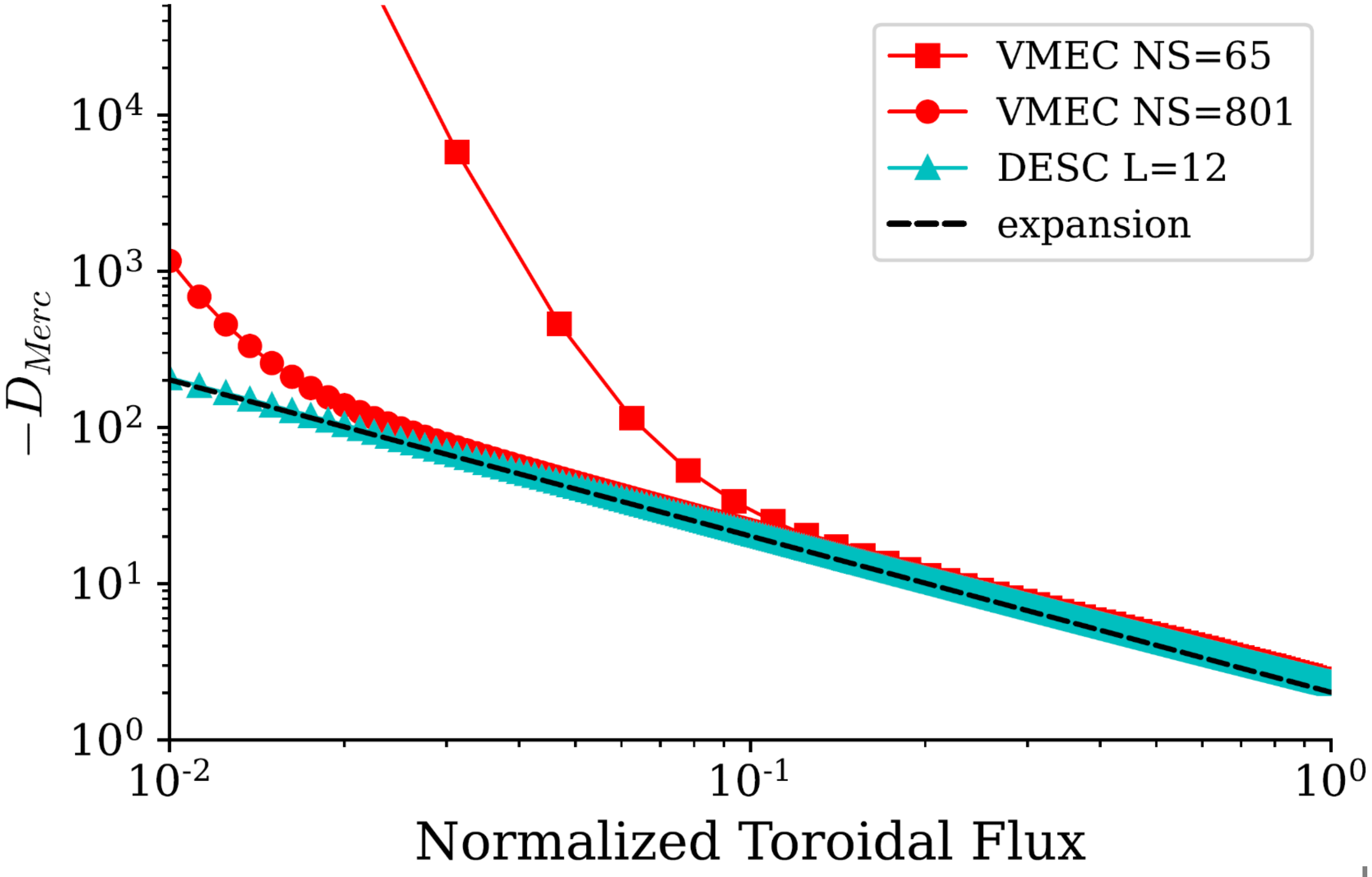
- **DESC coefficients obey constraint inherently due to Zernike basis**
- **VMEC Fourier coefficients do not**
  - **Unphysical modes in VMEC solution Fourier spectrum**





Plotted with help of Daniel Dudt

Accurately resolving the magnetic axis is important for stability calculations



VMEC requires high radial resolution to resolve axis

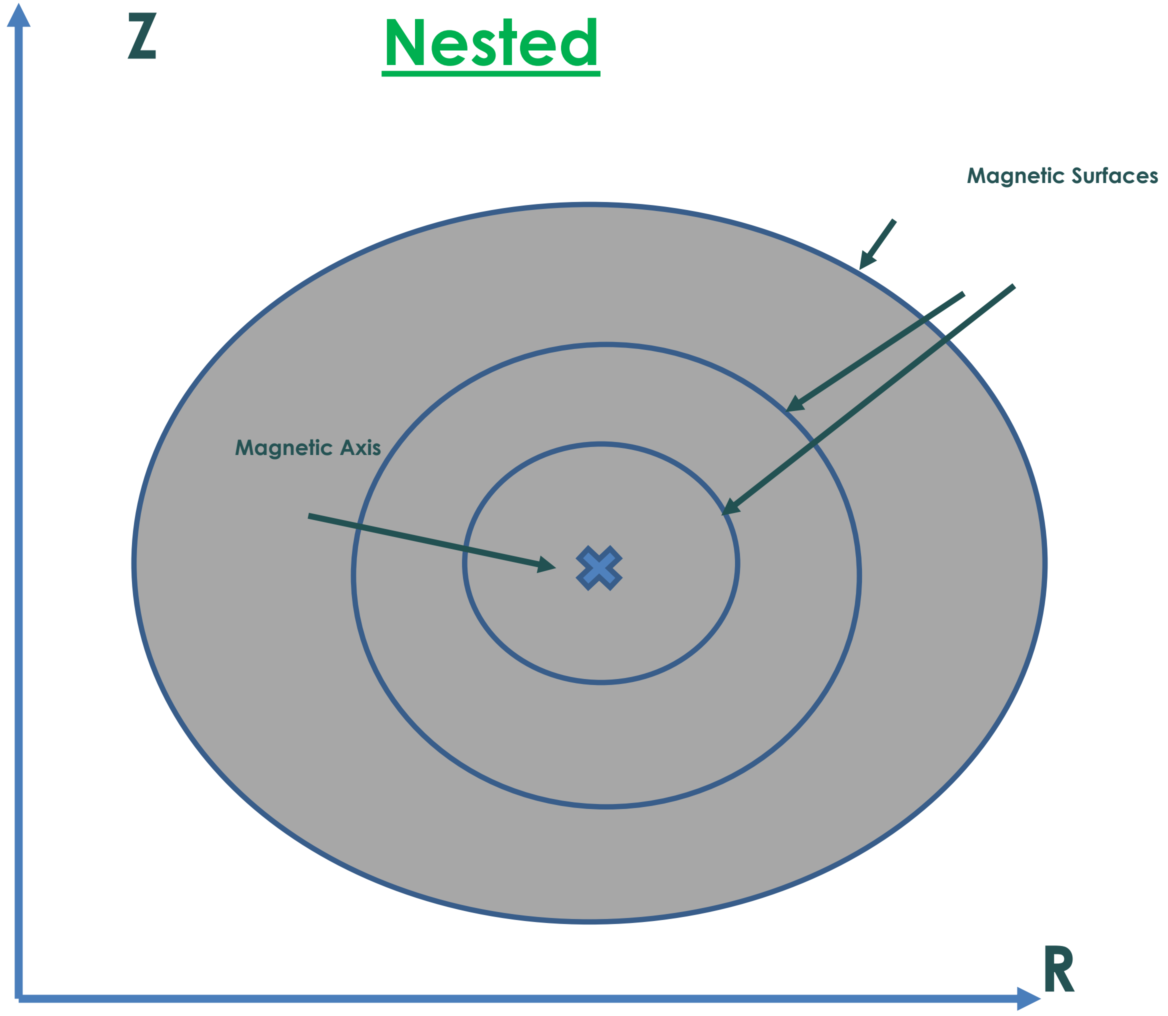
<u>Run times:</u>

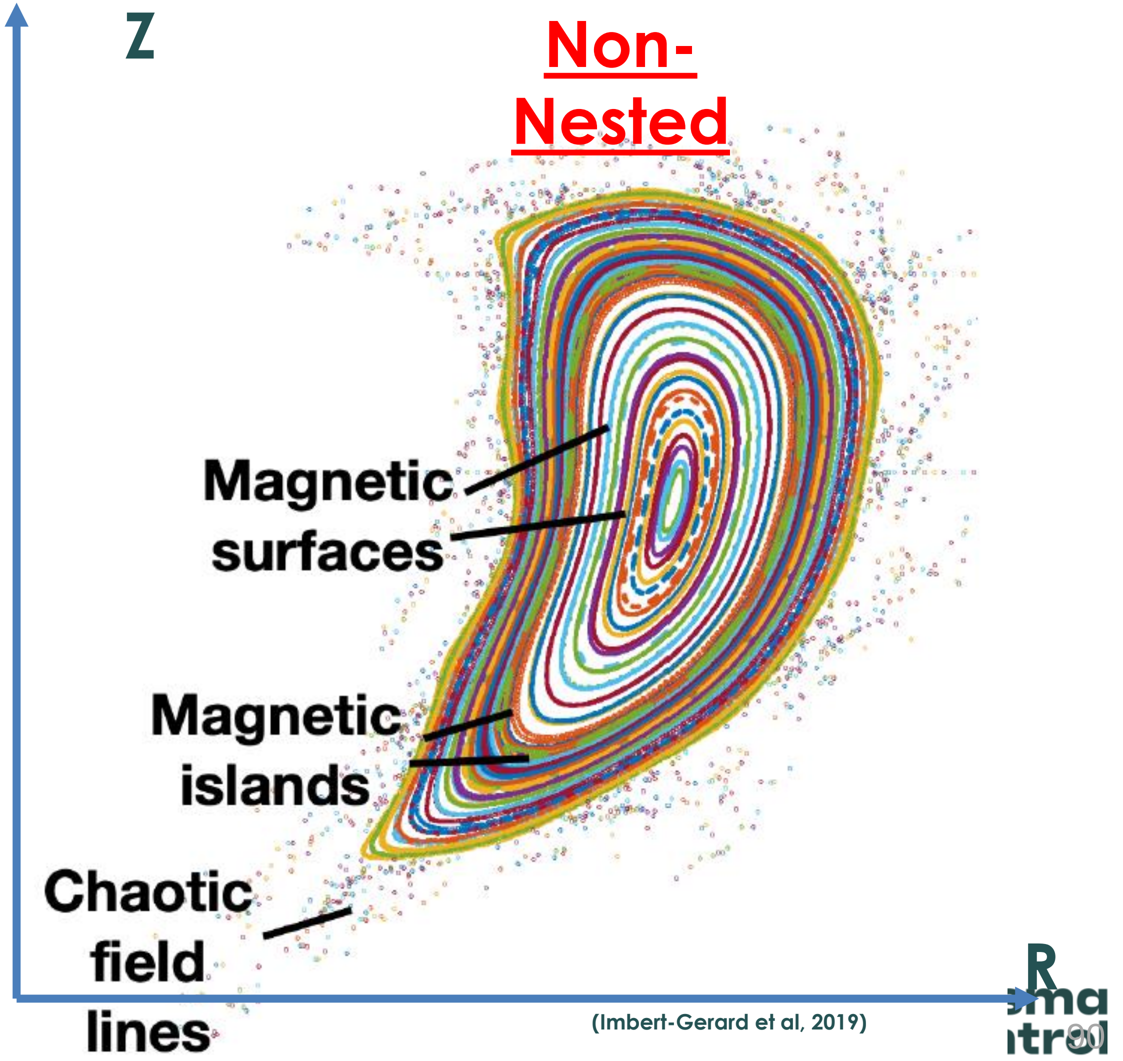
- **DESC = 0.2 GPU-hours (NVIDIA A100)**
- **VMEC = 5.2 CPU-hours (AMD Opteron 6276)**

Landreman & Sengupta, *J. Plasma Phys.* (2019)

Nested vs Non-Nested Flux Surfaces

Getty Images/iStockphoto





(Imbert-Gerard et al, 2019)

O(ρ^0) (axis) Constraint in DESC

NAE axis in pyQSC given as Fourier series in cylindrical toroidal angle ϕ:

$$R = R_0 + \sum_{n=1}^{N} (R_n^C \cos m\phi + R_n^S \sin m\phi) \qquad Z = \sum_{n=1}^{N} (Z_n^C \cos m\phi + Z_n^S \sin m\phi)$$

Constraint in DESC representation is simple: Evaluate DESC R(ρ,θ,ϕ), Z(ρ,θ,ϕ) at ρ=0 and match terms:

NAE Axis Coefficients

$$\boxed{\begin{aligned} R_n^{C/S} &= \sum_{k=0}^{\infty} (-1)^k R_{2k,0,\pm|n|} \\ Z_n^{C/S} &= \sum_{k=0}^{\infty} (-1)^k Z_{2k,0,\pm|n|} \end{aligned}}$$

DESC Fourier-Zernike Coefficients

$O(\rho^1)$ NAE Constraint in DESC

- After a short geometric derivation, one can derive (up to $O(\rho)$) the R,Z position of a point on a flux surface from the NAE in terms of the cylindrical angle

$$\mathbf{r} \approx \mathbf{r}_0(\phi) + \rho R_1 \hat{\mathbf{R}} + \rho Z_1 \hat{\mathbf{Z}}$$

where

$$R_1 = \mathcal{R}_{1,1}(\phi)\cos\theta + \mathcal{R}_{1,-1}(\phi)\sin\theta \qquad Z_1 = Z_{1,1}(\phi)\cos\theta + Z_{1,-1}(\phi)\sin\theta$$

- And the coefficients are functions of the NAE X,Y coefficients and the Frenet-Serret basis vectors

- Then, equating the $O(\rho)$ coefficients in the DESC Fourier-Zernike basis with the above expressions yields:

(Identical expressions for Z as well)

NAE Coefficients

$$\boxed{\begin{aligned}\mathcal{R}_{1,1,n} &= -\sum_{k=1}^{M}(-1)^k k R_{2k-1,1,n},\\ \mathcal{R}_{1,-1,n} &= -\sum_{k=1}^{M}(-1)^k k R_{2k-1,-1,n},\end{aligned}}$$

DESC Fourier-Zernike Coefficients

Plasma Control