Quick and Accurate 3D MHD Equilibria with DESC

Dario Panici, Daniel Dudt, Rory Conlin, Egemen Kolemen

Presented at: CCP 2022 (Virtual)

Princeton Plasma Control

control.princeton.edu

PRINCETON PLASMA PHYSICS LABORATORY

PRINCETON UNIVERSITY

Plasma Equilibria: What and Why?

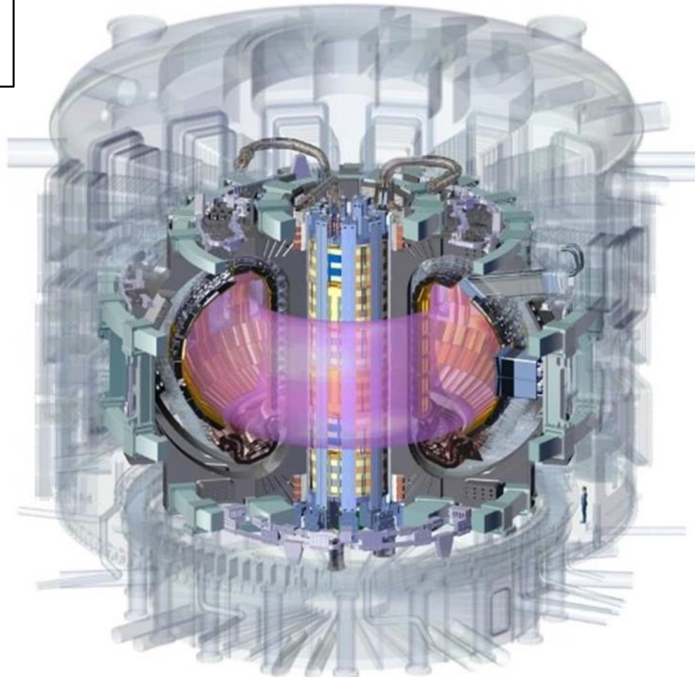
$$\boldsymbol{F} = \boldsymbol{J} \times \boldsymbol{B} - \nabla p = 0$$

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

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

Quick

Accurate



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

PPPL PRINCETON PLASMA PHYSICS LABORATORY

PRINCETON UNIVERSITY

Stellarator Equilibrium - DESC

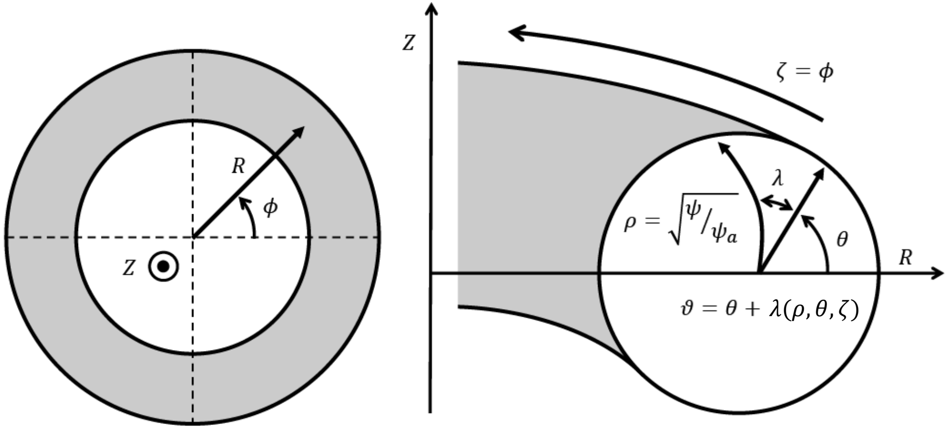
$$\boldsymbol{F} = \boldsymbol{J} \times \boldsymbol{B} - \nabla p = 0$$

DESC

(Dudt and Kolemen 2020)

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

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



Stellarator Equilibrium - VMEC

$$\frac{\partial X_j^{mn}}{\partial t} = F_j^{mn}$$

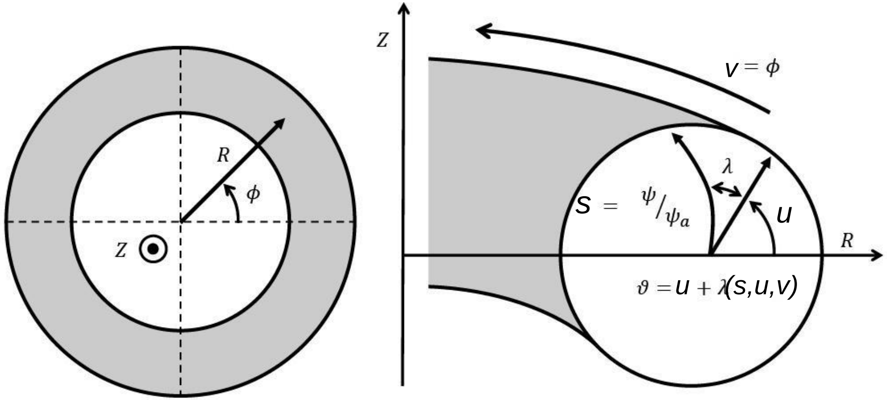
$$W = \int_V \left(\frac{B^2}{2\mu_0} + \frac{p}{\gamma - 1} dV \right)$$

- Spectral inverse equilibrium code (Hirshman and Whitman, 1983)
- Assumes Nested Flux Surfaces, Ideal MHD
- Fourier series on flux surfaces, defined only on discrete radial grid
- Angular derivatives analytic, but radial derivatives are finite difference
- Minimizes energy with steepest-descent method based on variational principle

$$R(s,u,v) = \sum_{m=0,n=-N}^{M,N} R_{mn,c}(s) cos(mu - nvN_{FP}) + R_{mn,s}(s) sin(mu - nvN_{FP})$$

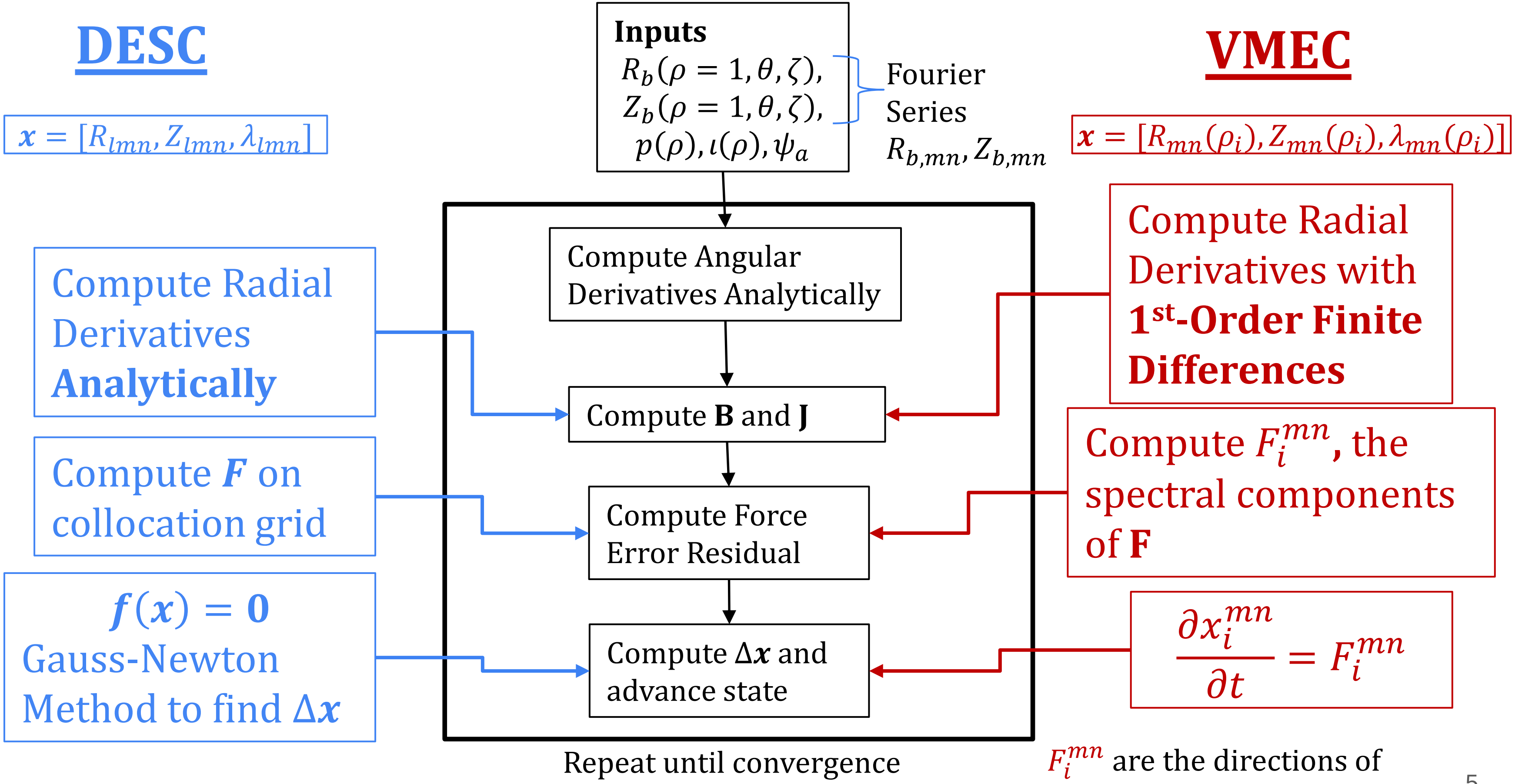
$$\lambda(s,u,v) = \sum_{m=0,n=-N}^{M,N} \lambda_{mn,c}(s) cos(mu - nvN_{FP}) + \lambda_{mn,s}(s) sin(mu - nvN_{FP})$$

$$Z(s,u,v) = \sum_{m=0,n=-N}^{M,N} Z_{mn,c}(s) cos(mu - nvN_{FP}) + Z_{mn,s}(s) sin(mu - nvN_{FP})$$



(Hirshman and Whitson, 1983)

Code Algorithms



F_i^{mn} are the directions of steepest descent for energy

Force Error as an Accuracy Metric

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

- Goal is to satisfy MHD equilibrium force balance in volume
- Looking at residual force error is an **intuitive** metric of how well the governing equations are being solved
- Use force error residual as metric to compare DESC and VMEC codes
- VMEC does not output force error in real space
  - -> Must calculate from outputs (R,Z,λ)

Force Error Metrics

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

- **Volume-Averaged Force Error**
  - Taken from $s = 0.1 \rightarrow s = 0.99$

$$< F >_{vol} = \frac{\int_{\theta=0}^{2\pi} \int_{\phi=0}^{2\pi} \int_{s=0.1}^{0.99} |F||\sqrt{g}|ds d\phi d\theta}{\int_{\theta=0}^{2\pi} \int_{\phi=0}^{2\pi} \int_{s=0.1}^{0.99} |\sqrt{g}|ds d\phi d\theta}$$

- **Flux-Surface-Averaged Force Error**

$$< F >_{fsa}(s) = \frac{\int_{\theta=0}^{2\pi} \int_{\phi=0}^{2\pi} |F(s)||\sqrt{g}(s)|d\phi d\theta}{\int_{\theta=0}^{2\pi} \int_{\phi=0}^{2\pi} |\sqrt{g}(s)|d\phi d\theta}$$

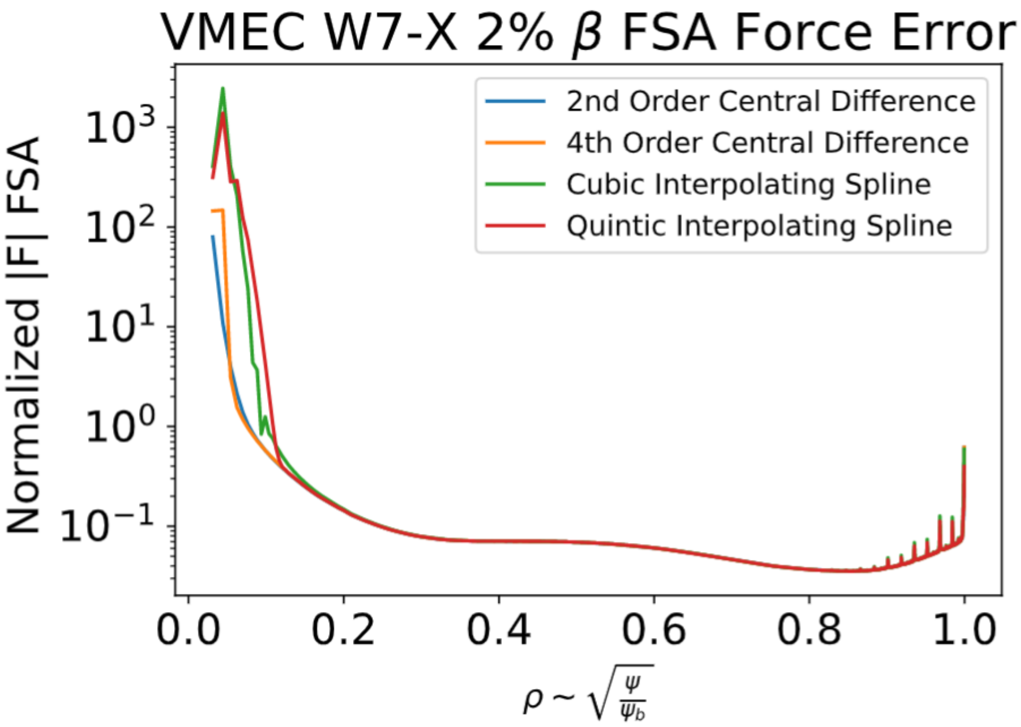
- **Both normalized by pressure gradient volume average**

$$< |\nabla p| >_{vol} = \frac{\int_{\theta=0}^{2\pi} \int_{\phi=0}^{2\pi} \int_{s=0}^{1} |\nabla p||\sqrt{g}|ds d\phi d\theta}{\int_{\theta=0}^{2\pi} \int_{\phi=0}^{2\pi} \int_{s=0}^{1} |\sqrt{g}|ds d\phi d\theta}$$

Calculated Force Error Insensitive to Radial Derivative Method

Only error calculated near-axis changes significantly with method

2nd Order Central Differences used for remainder of results shown in this presentation



PRINCETON PLASMA PHYSICS LABORATORY

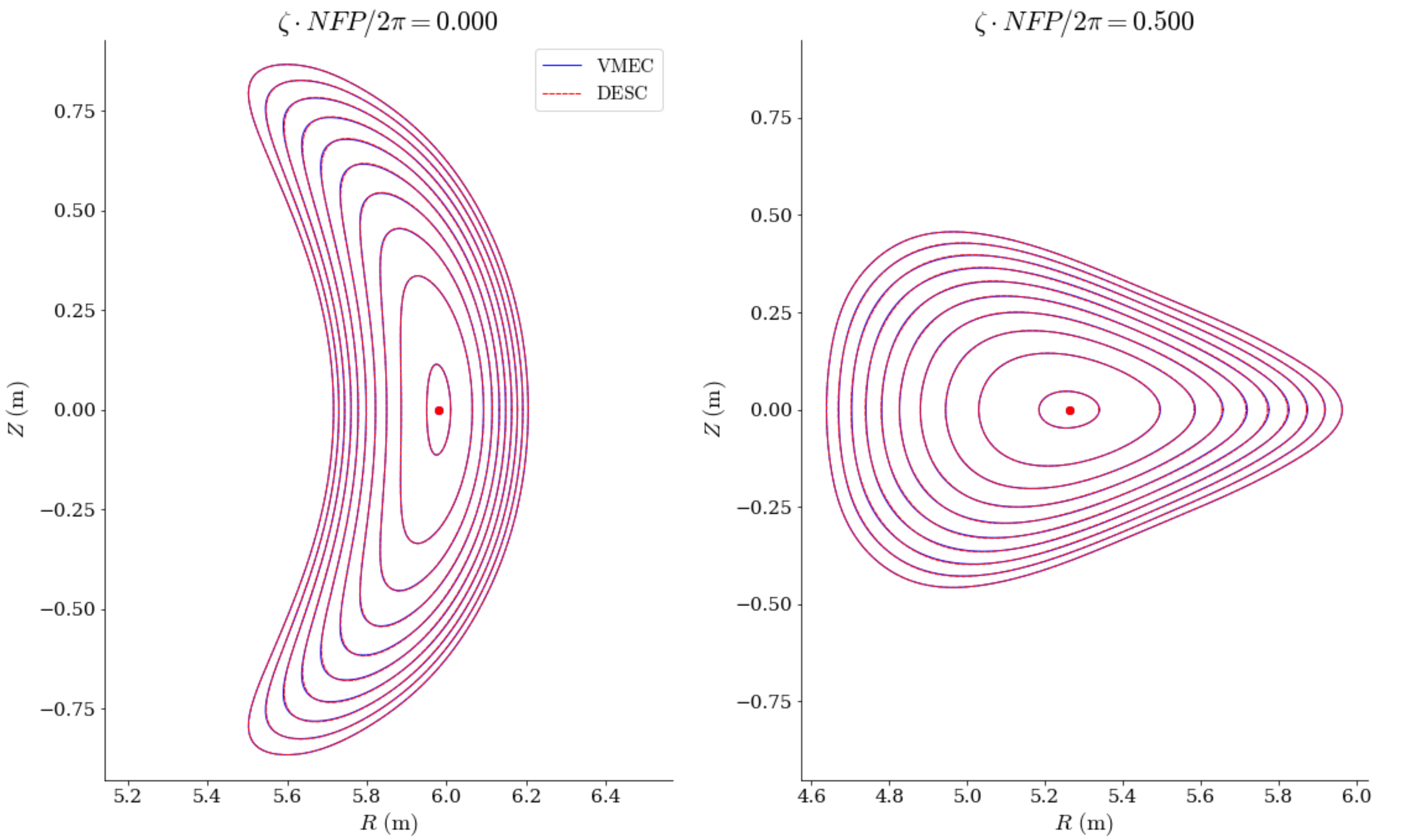
PRINCETON UNIVERSITY

Code Solution Comparison Procedure

- The **same W7X-like input** boundary and profiles (available on DESC github) were used for all comparisons
- **Angular and radial resolutions** for each code were varied and ran to form a set of solutions
- **Normalized force balance error metrics** for each solution was calculated
- All solutions were ran in **fixed boundary** mode
- All solutions were ran on **identical architectures**
  - A single AMD EPYC 7281 CPU core with 32GB of RAM on PPPL's portal computing clusters

	Angular Resolution	**Radial Resolution**	**Other Parameters**
DESC	M=N=[8,10,12,14,16,18,20]	L=M=N	Fringe and ANSI spectral indexing
VMEC	M=N=[8,10,12,14,16,18,20]	NS=[256,512,1024]	FTOL=[1E-4,1E-8,1E-12]

Solution Comparison - Flux Surfaces Indistinguishable by Eye

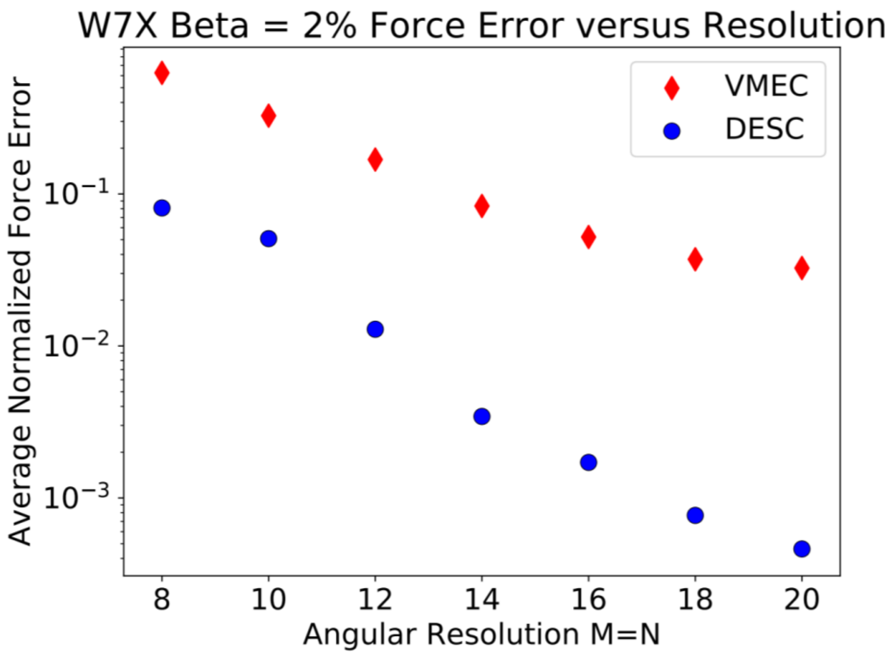


PRINCETON PLASMA PHYSICS LABORATORY

PRINCETON UNIVERSITY

For Given Resolution, DESC has lower Force Error - **Accurate**

W7X M=N=12			
	Energy (J)	|F|/|∇p|	Runtime (1 CPU)
DESC (L=12)	8.4648759e+07	0.013	0.71 hours
VMEC (ns=1024)	8.4648752e+07	0.168	1.19 hours



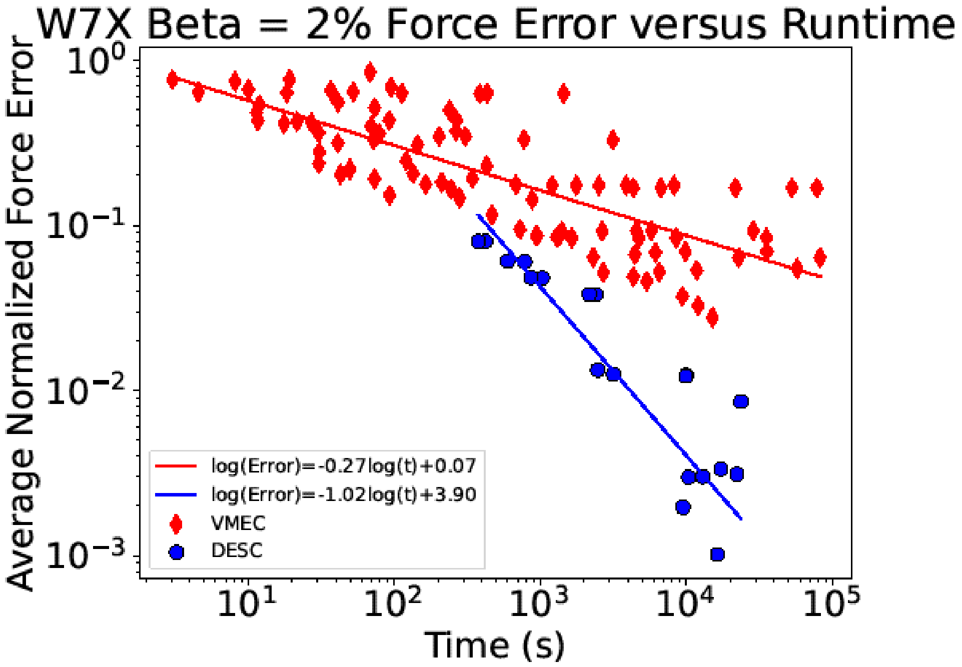
$$< F >_{vol} = \frac{\int_{\theta=0}^{2\pi}\int_{\phi=0}^{2\pi}\int_{\rho=0.1}^{0.99} |F||\sqrt{g}|d\rho d\phi d\theta}{\int_{\theta=0}^{2\pi}\int_{\phi=0}^{2\pi}\int_{\rho=0.1}^{0.99} |\sqrt{g}|d\rho d\phi d\theta}$$

PPPL PRINCETON PLASMA PHYSICS LABORATORY

PRINCETON UNIVERSITY

For Given Time to Solution, DESC has lower Force Error - **Quick**

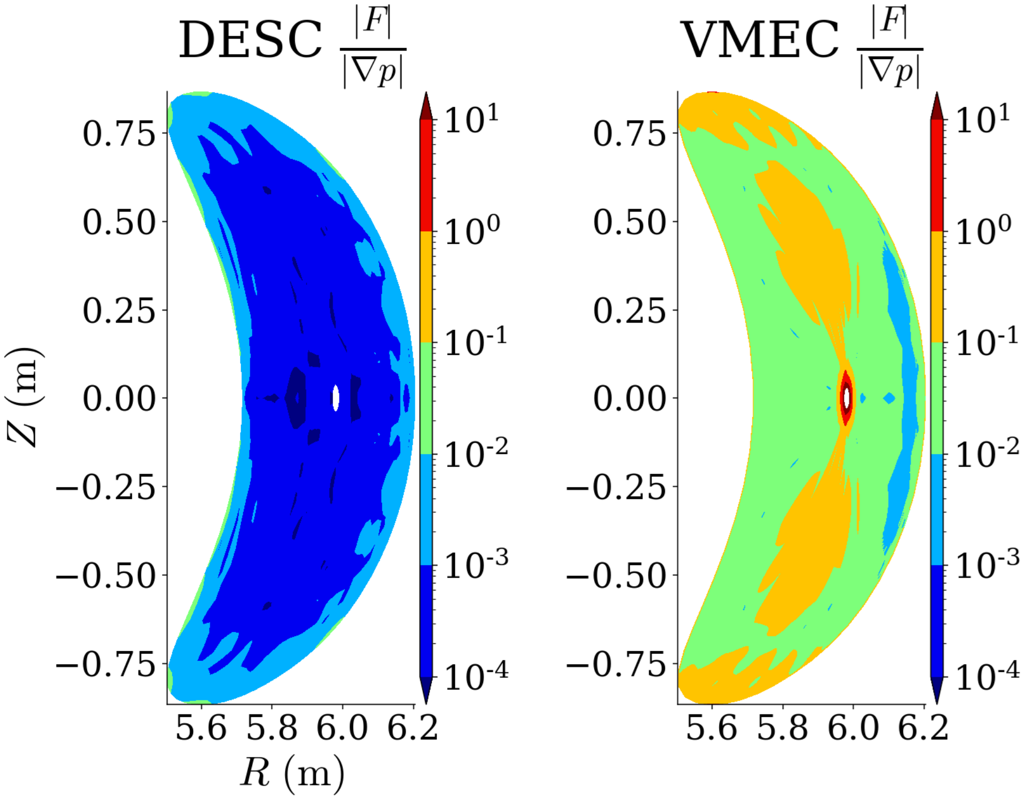
W7X M=N=12			
	Energy (J)	|F|/|∇p|	Runtime (1 CPU)
DESC (L=12)	8.4648759e+07	0.013	0.71 hours
VMEC (ns=1024)	8.4648752e+07	0.168	1.19 hours



$$< F >_{vol} = \frac{\int_{\theta=0}^{2\pi} \int_{\phi=0}^{2\pi} \int_{\rho=0.1}^{0.99} |F||\sqrt{g}| d\rho d\phi d\theta}{\int_{\theta=0}^{2\pi} \int_{\phi=0}^{2\pi} \int_{\rho=0.1}^{0.99} |\sqrt{g}| d\rho d\phi d\theta}$$

VMEC Force Error is Noticeably Higher Near-axis

- This could be due to VMEC's Fourier coefficients not explicitly obeying analyticity constraint near axis



PPPL PRINCETON PLASMA PHYSICS LABORATORY

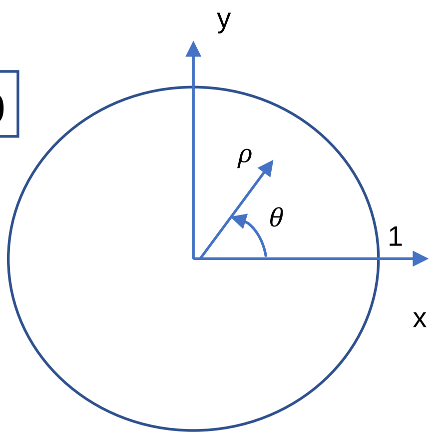
PRINCETON UNIVERSITY

Analyticity Constraints for Functions in Polar Domains

If a function $f(\rho,\theta)$ is analytic everywhere on the unit disk, then

$$\lim_{\rho\to 0}\frac{a_m}{\rho^m} < \infty \quad \lim_{\rho\to 0}\frac{b_m}{\rho^m} < \infty$$

i.e $a_m, b_m \sim \rho^m$ as $\rho \to 0$



where

$$f(\rho,\theta) = \sum_{m=0}^{\infty} a_m(\rho)\cos(m\theta) + \sum_{m=0}^{\infty} b_m(\rho)\, sin(m\theta)$$

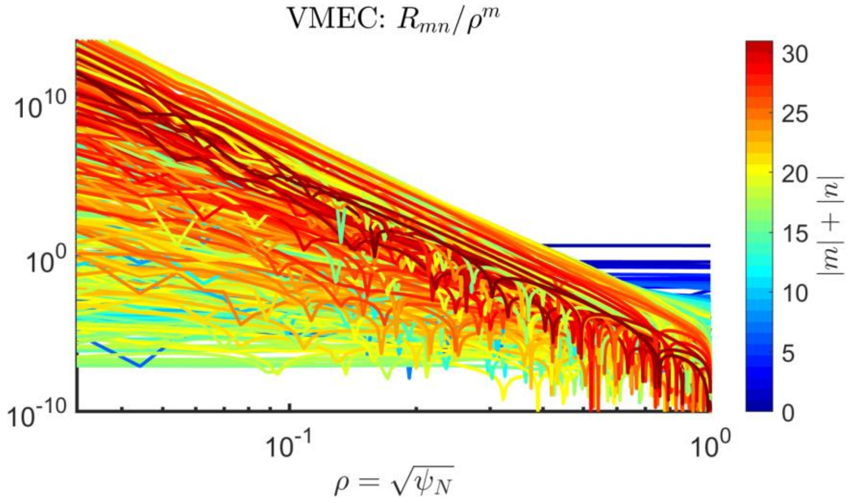
- **Physical Quantities (like B) are analytic**

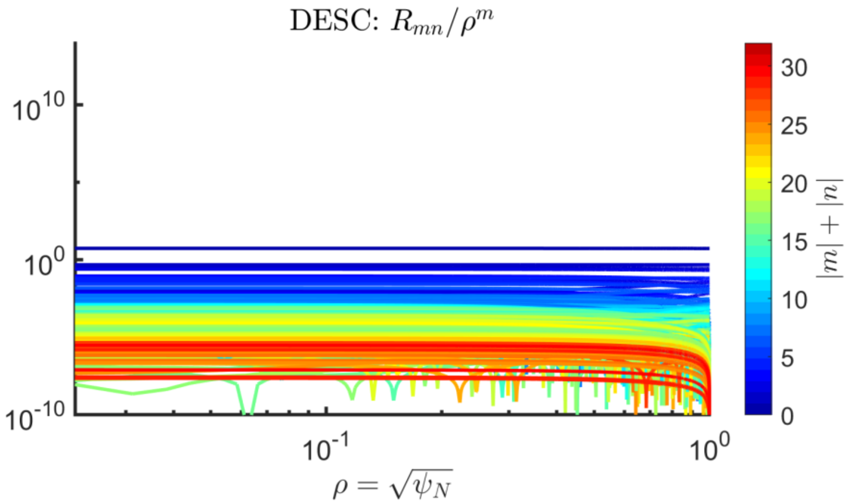
- The Zernike basis radial-poloidal mode coupling **automatically satisfies this constraint**

Analytic Constraint Near Axis

$$\lim_{\rho \to 0} \frac{a_m}{\rho^m} < \infty$$

- Fourier coefficients of an analytic function must scale as ρ^m near axis (Lewis and Bellan 1990)
- DESC coefficients obey this inherently due to Zernike basis, VMEC do not, especially for higher order modes





Plots Courtesy of Daniel Dudt

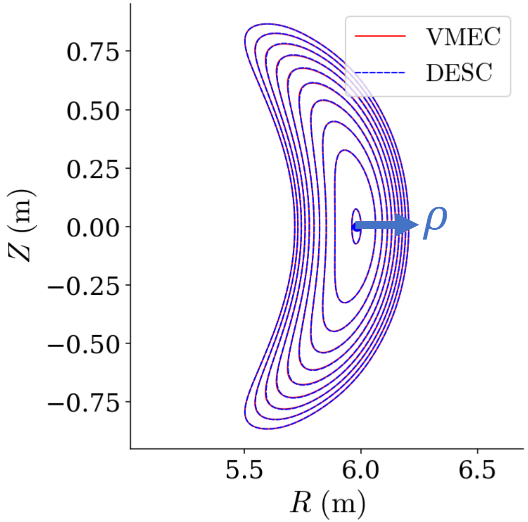
PRINCETON PLASMA PHYSICS LABORATORY

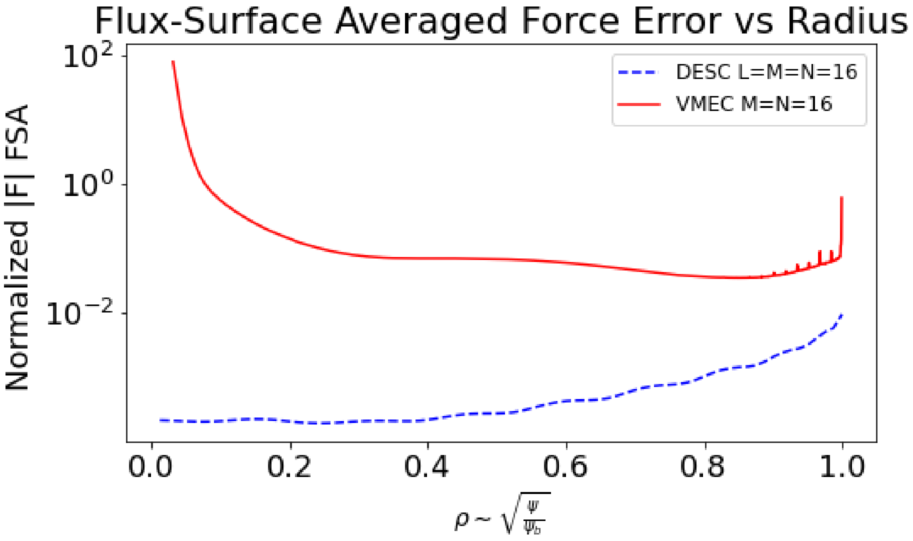
PRINCETON UNIVERSITY

DESC compares well to VMEC – Force Error

- Surface-Averaged Force Balance Error **lower in DESC** than VMEC

- VMEC error spikes near $\rho \to 0$: Issues at axis!

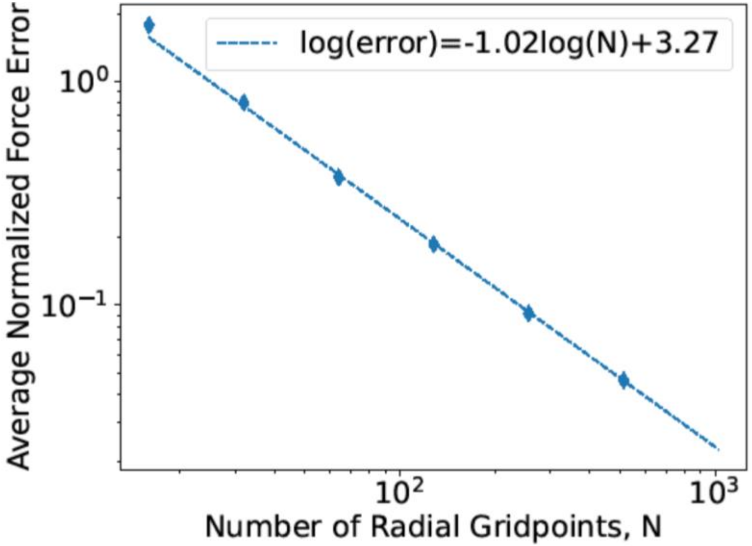




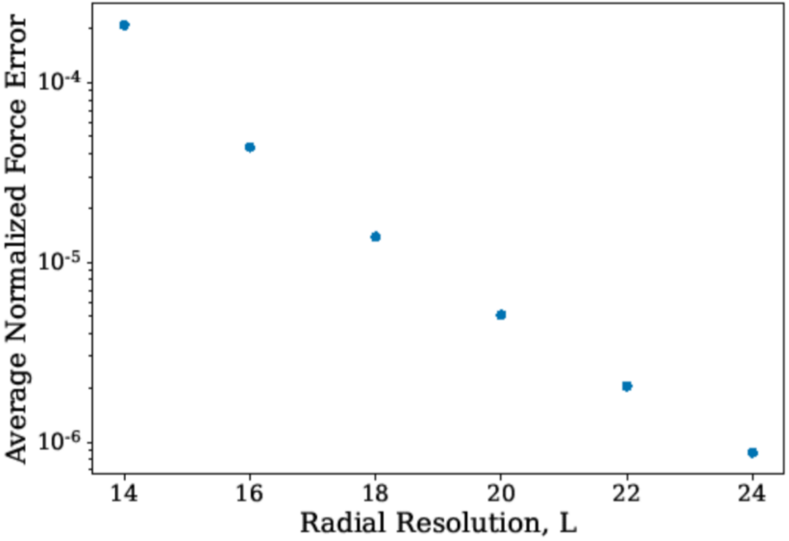
DESC Achieves Superior Radial Convergence over VMEC (DSHAPE)

	Angular Convergence	**Radial Convergence**
DESC	Exponential	**Exponential**
VMEC	Exponential	**Algebraic** $O\left(N_{radial}^{-1}\right)$

VMEC: Algebraic Radial Convergence



DESC: Exponential Radial Convergence



PPPL PRINCETON PLASMA PHYSICS LABORATORY

Left: log-log axis

Right: semi-log axis

PRINCETON UNIVERSITY

DESC is an Improvement over VMEC

VMEC	DESC
Analyticity Issues at Magnetic Axis	Zernike Polynomials resolves axis issues

DESC is an Improvement over VMEC

VMEC	DESC
Analyticity Issues at Magnetic Axis	Zernike Polynomials resolves axis issues
Convergence limited by finite difference accuracy	Pseudospectral method convergence limited only by smoothness of solution

DESC is an Improvement over VMEC

VMEC	DESC
Analyticity Issues at Magnetic Axis	Zernike Polynomials resolves axis issues
Convergence limited by finite difference accuracy	Pseudospectral method convergence limited only by smoothness of solution
Energy Minimization makes solution quality difficult to assess	Force Error Minimization makes quality intuitive (lower **F = better**)

DESC is an Improvement over VMEC

VMEC	DESC
Analyticity Issues at Magnetic Axis	Zernike Polynomials resolves axis issues
Convergence limited by finite difference accuracy	Pseudospectral method convergence limited only by smoothness of solution
Energy Minimization makes solution quality difficult to assess	Force Error Minimization makes quality intuitive (lower **F = better**)
Gradient descent method to find $\Delta \boldsymbol{x}$	Gauss-Newton Method to find $\Delta \boldsymbol{x}$ $\rightarrow$ super-linear convergence

DESC is an Improvement over VMEC

[1] https://github.com/PrincetonUniversity/STELLOPT/blob/3b0f12d31926e4900c15b473fcafb01ed90605c7/VMEC2000/Sources/General/totzsp_mod.f

VMEC	DESC
Analyticity Issues at Magnetic Axis	Zernike Polynomials resolves axis issues
Convergence limited by finite difference accuracy	Pseudospectral method convergence limited only by smoothness of solution
Energy Minimization makes solution quality difficult to assess	Force Error Minimization makes quality intuitive (lower **F = better**)
Gradient descent method to find $\Delta \boldsymbol{x}$	Gauss-Newton Method to find $\Delta \boldsymbol{x}$ $\rightarrow$ super-linear convergence
Poorly documented, aging Fortran `!> @note FIXME Figure out what rcn1 and zcn1 are.` [1]	Recent code, Python

DESC is an Improvement over VMEC

[1]https://github.com/PrincetonUniversity/STELLOPT/blob/3b0f12d31926e4900c15b473fcafb01ed90605c7/VMEC2000/Sources/General/totzsp_mod.f

VMEC	DESC
Analyticity Issues at Magnetic Axis	Zernike Polynomials resolves axis issues
Convergence limited by finite difference accuracy	Pseudospectral method convergence limited only by smoothness of solution
Energy Minimization makes solution quality difficult to assess	Force Error Minimization makes quality intuitive (lower **F = better**)
Gradient descent method to find $\Delta \boldsymbol{x}$	Gauss-Newton Method to find $\Delta \boldsymbol{x}$ → super-linear convergence
Poorly documented, aging Fortran `!> @note FIXME Figure out what rcn1 and zcn1 are.` [1]	Recent code, Python
Parallelized across CPUs	Ability to use GPUs for speedup
	Automatic Differentiation

Conclusions

- DESC **more accurate** than VMEC at given resolution or time-to-solution
- DESC solution accuracy **better than VMEC near axis**
- DESC radial convergence **not limited by finite differences**
- Future work can make DESC faster – pre-compilation of objective, parallelize across CPUs/GPUs

Check out our Code and Publications!

- D.W. Dudt and E. Kolemen (2020). DESC: A stellarator equilibrium solver. *Phys. Plasmas*, 27 (10)
- The DESC Stellarator Code Suite Part I https://arxiv.org/abs/2203.17173
- The DESC Stellarator Code Suite Part II https://arxiv.org/abs/2203.15927
- The DESC Stellarator Code Suite Part III https://arxiv.org/abs/2204.00078

<u>Repository:</u> https://github.com/PlasmaControl/DESC

<u>Python Package:</u> `pip install desc-opt`

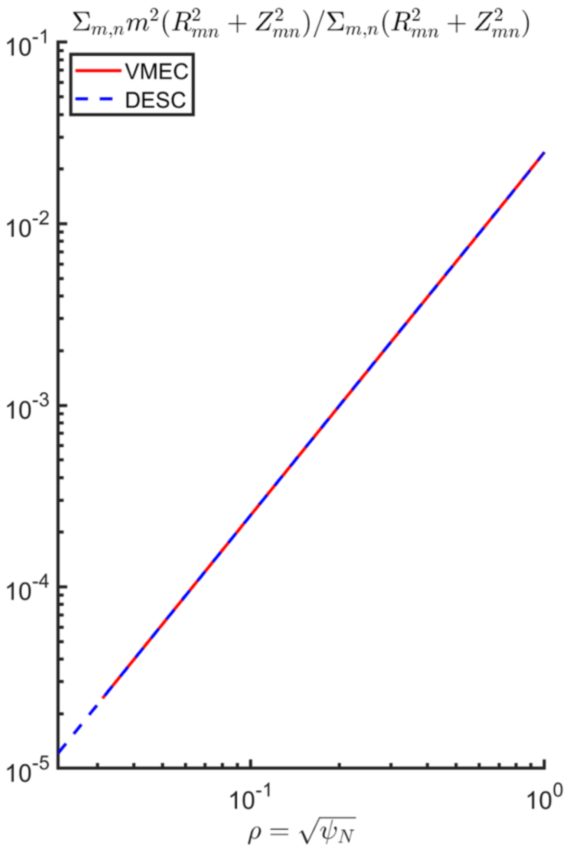
Backup

Both DESC and VMEC Poloidal Angle are Optimal

- Spectral condensation as defined by Hirshman and Meier (1985)

$$M(p,q) \equiv \frac{\Sigma_{m=1}\, m^q S_p(m)}{\Sigma_{m=1} S_p(m)}$$

- Minimization of M wrt poloidal angle corresponds to an optimally condensed Fourier spectrum -> explicit constraint in VMEC
- DESC poloidal angle found through optimization is as optimal as VMEC's

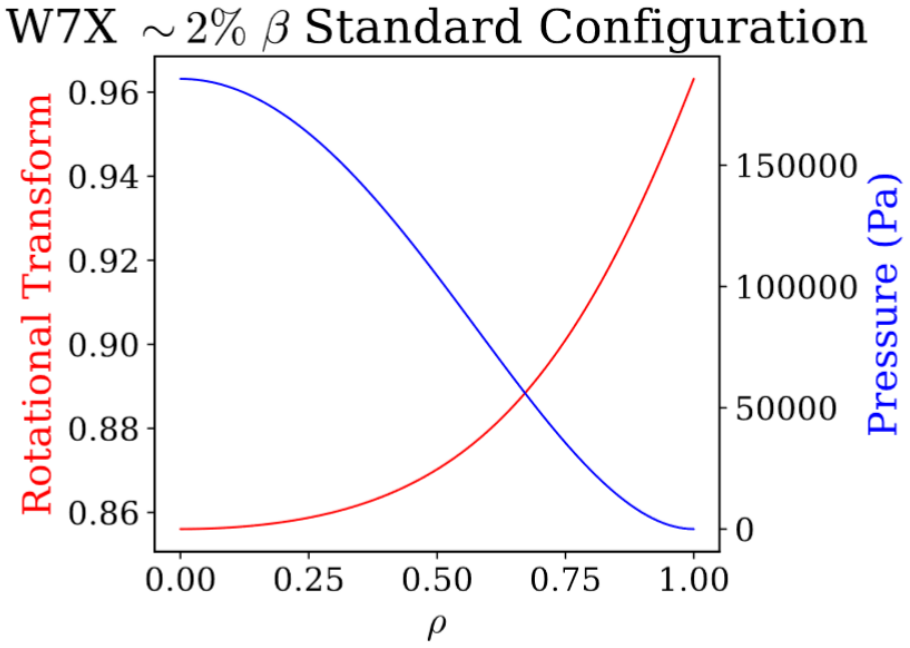


Plot Courtesy of Daniel Dudt

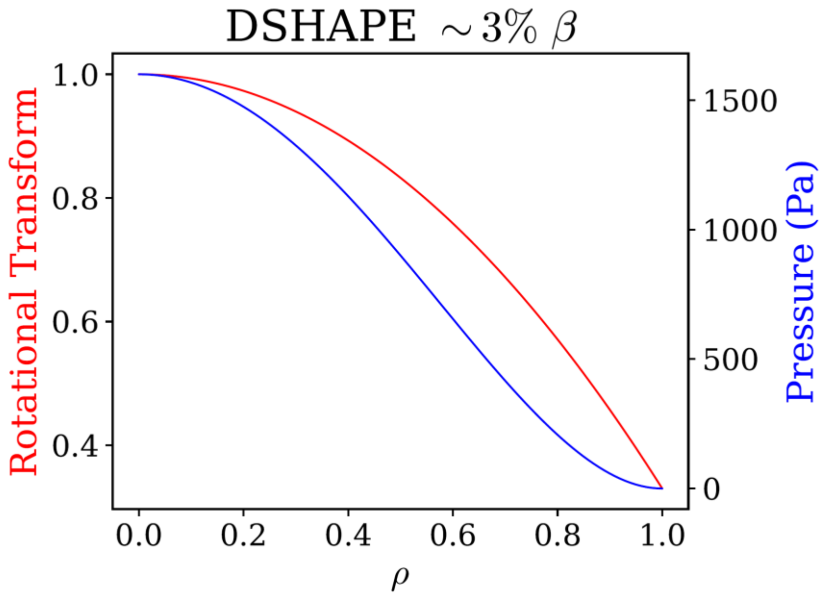
PPPL PRINCETON PLASMA PHYSICS LABORATORY

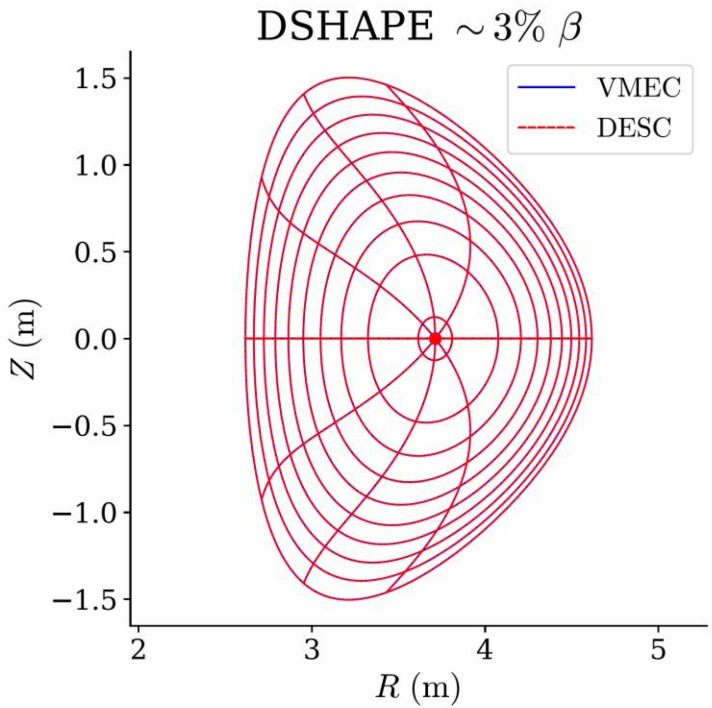
PRINCETON UNIVERSITY

W7-X Equilibrium Input Profiles



DSHAPE Equilibrium and Profiles





PPPL PRINCETON PLASMA PHYSICS LABORATORY

PRINCETON UNIVERSITY

Force Error is Calculated from VMEC Starting with R,Z,λ

- Read in Fourier coefficients from VMEC wout file
  - convert λ from half -> full mesh
- Find necessary angular derivatives **analytically**
- Find necessary radial derivatives **numerically**
  - finite difference, splines, etc.
- Multiply out in real space to find force error **F**
- Use **F** to define accuracy metrics

$$\boldsymbol{R(s,u,v), Z(s,u,v), \lambda(s,u,v)}$$

$$\mathbf{e}_s = \begin{bmatrix} \partial_s R \\ 0 \\ \partial_s Z \end{bmatrix} \quad \mathbf{e}_u = \begin{bmatrix} \partial_u R \\ 0 \\ \partial_u Z \end{bmatrix} \quad \mathbf{e}_v = \begin{bmatrix} \partial_v R \\ R \\ \partial_v Z \end{bmatrix}$$

$$\sqrt{g} = \mathbf{e}_s \cdot \mathbf{e}_u \times \mathbf{e}_v$$

$$B^u = \frac{1}{\sqrt{g}}\left(\chi' - \psi' \frac{\partial \lambda}{\partial v}\right) \qquad B^v = \frac{1}{\sqrt{g}}\psi'\left(1 + \frac{\partial \lambda}{\partial u}\right)$$

$$J^s = \frac{1}{\mu_0 \sqrt{g}}\left(\frac{\partial B_v}{\partial u} - \frac{\partial B_u}{\partial v}\right)$$

$$J^u = \frac{1}{\mu_0 \sqrt{g}}\left(\frac{\partial B_s}{\partial v} - \frac{\partial B_v}{\partial s}\right)$$

$$J^v = \frac{1}{\mu_0 \sqrt{g}}\left(\frac{\partial B_u}{\partial s} - \frac{\partial B_s}{\partial u}\right)$$

$$\boldsymbol{B}(s,u,v), \boldsymbol{J}(s,u,v)$$

$$F_s = \sqrt{g}(J^v B^u - J^u B^v) + p'$$

$$F_\beta = J^s$$

$$\boldsymbol{F}(s,u,v)$$

VMEC – Theory

$$X_j = \{R, \lambda, Z\}, j = 1,2,3$$

$$X_j = \sum_{m,n} X_j^{mn} e^{(i(mu - nv))}$$

$$W = \int_V \left(\frac{B^2}{2\mu_0} + \frac{p}{\gamma - 1} dV \right)$$

First variation, with t as variational parameter

$$\frac{dW}{dt} = \int_V (F_j^{mn})^* \frac{\partial X_j^{mn}}{\partial t} dV$$

Steepest Descent direction: change X_j^{mn} until $\frac{dW}{dt} = 0$ i.e. stationary point is reached

$$\frac{\partial X_j^{mn}}{\partial t} = F_j^{mn}$$

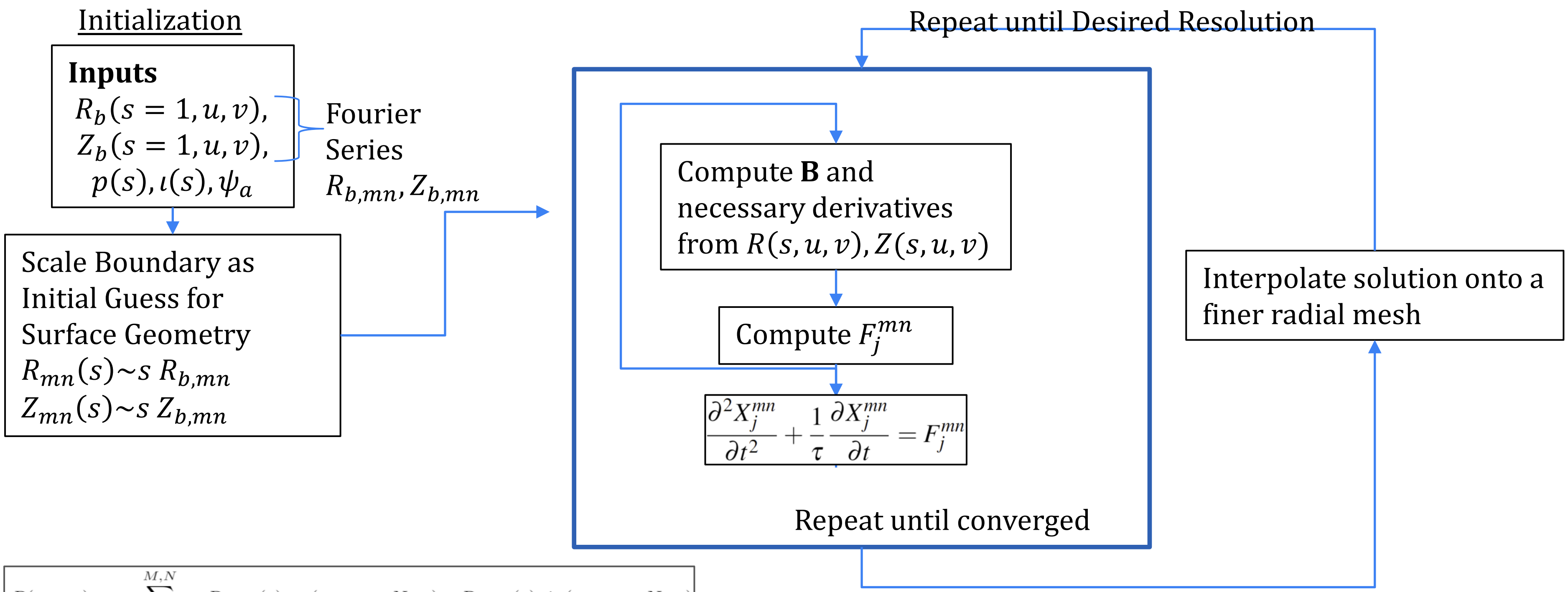
$$\frac{\partial^2 X_j^{mn}}{\partial t^2} + \frac{1}{\tau} \frac{\partial X_j^{mn}}{\partial t} = F_j^{mn}$$

Change to 2nd order for better convergence

(Hirshman and Whitson, 1983)

VMEC Algorithm

Main Algorithm



$$R(s,u,v) = \sum_{m=0,n=-N}^{M,N} R_{mn,c}(s)cos(mu - nvN_{FP}) + R_{mn,s}(s)sin(mu - nvN_{FP})$$

$$\lambda(s,u,v) = \sum_{m=0,n=-N}^{M,N} \lambda_{mn,c}(s)cos(mu - nvN_{FP}) + \lambda_{mn,s}(s)sin(mu - nvN_{FP})$$

$$Z(s,u,v) = \sum_{m=0,n=-N}^{M,N} Z_{mn,c}(s)cos(mu - nvN_{FP}) + Z_{mn,s}(s)sin(mu - nvN_{FP})$$

What DESC Solves

Inputs: $R_b(\theta,\zeta), Z_b(\theta,\zeta), p(\rho), \iota(\rho)$

$$\mathbf{e}_\rho = \begin{bmatrix}\partial_\rho R\\ 0\\ \partial_\rho Z\end{bmatrix} \quad \mathbf{e}_\theta = \begin{bmatrix}\partial_\theta R\\ 0\\ \partial_\theta Z\end{bmatrix} \quad \mathbf{e}_\zeta = \begin{bmatrix}\partial_\zeta R\\ R\\ \partial_\zeta Z\end{bmatrix}$$

$$\sqrt{g} = \mathbf{e}_\rho \cdot \mathbf{e}_\theta \times \mathbf{e}_\zeta$$

$$B^\theta = \frac{\psi'}{\sqrt{g}}\left(\iota - \frac{\partial \lambda}{\partial \zeta}\right)$$

$$B^\zeta = \frac{1}{\sqrt{g}}\psi'\left(1 + \frac{\partial \lambda}{\partial \theta}\right)$$

$$J^\rho = \frac{1}{\mu_0\sqrt{g}}\left(\frac{\partial B_\zeta}{\partial \theta} - \frac{\partial B_\theta}{\partial \zeta}\right)$$

$$J^\theta = \frac{1}{\mu_0\sqrt{g}}\left(\frac{\partial B_\rho}{\partial \zeta} - \frac{\partial B_\zeta}{\partial \rho}\right)$$

$$J^\zeta = \frac{1}{\mu_0\sqrt{g}}\left(\frac{\partial B_\theta}{\partial \rho} - \frac{\partial B_\rho}{\partial \theta}\right)$$

$\mathbf{B}(\rho,\theta,\zeta), \mathbf{J}(\rho,\theta,\zeta)$

$$F_\rho = \sqrt{g}(J^\zeta B^\theta - J^\theta B^\zeta) + p'$$

$$F_\beta = \sqrt{g} J^\rho$$

$\mathbf{F}(\rho,\theta,\zeta)$

R, Z, λ and their derivatives are evaluated on a collocation grid in (ρ,θ,ζ) , then multiplied to calculate **F** on this grid

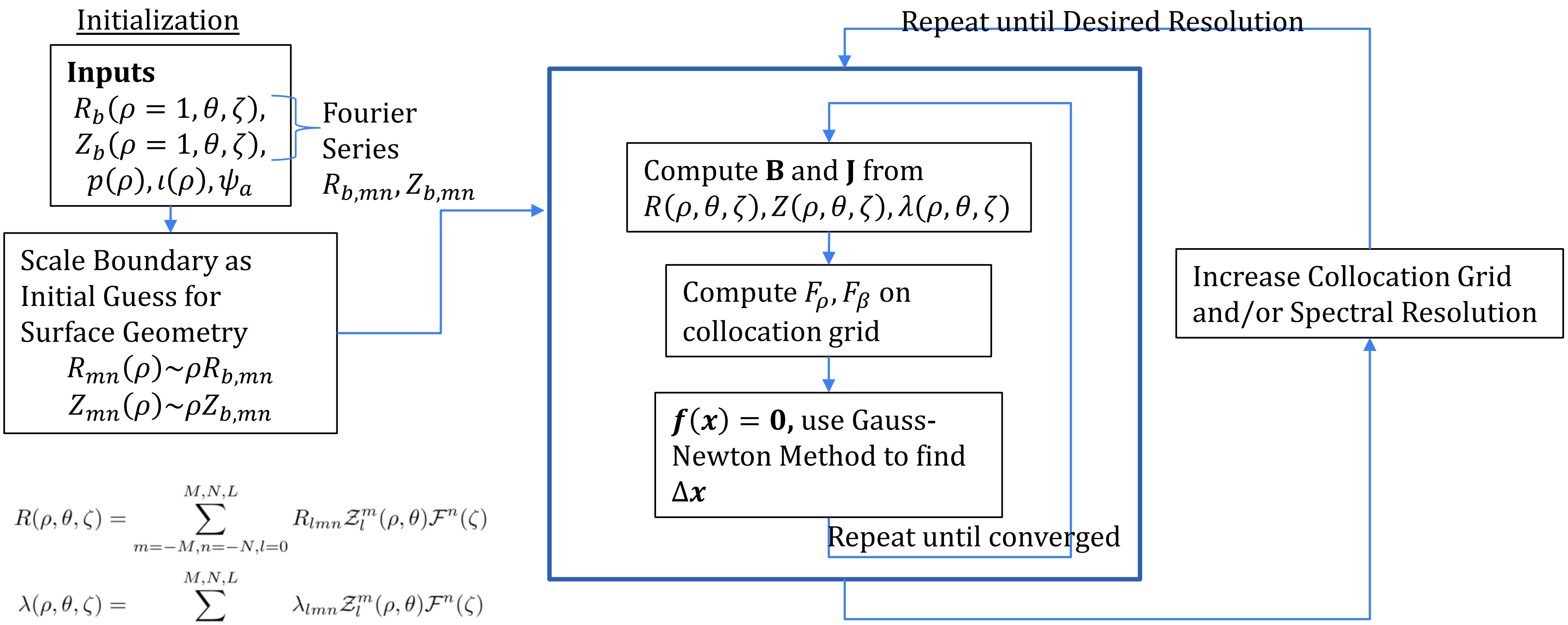
This leads to a system of equations comprised of the force balance error evaluated at the collocation nodes, which we want to make equal to zero -> Can use root-finding or least-squares to solve

$$\boldsymbol{f}(\boldsymbol{x}) = \mathbf{0}$$

x is the spectral coefficients of *R,Z,λ*, which is what we are changing to minimize **f**

DESC Algorithm

Main Algorithm



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

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

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

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

Plasma Model – Ideal MHD

Mass:	$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{v}) = 0$
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:	$\nabla \times \mathbf{E} = -\frac{\partial \mathbf{B}}{\partial t}$
	$\nabla \times \mathbf{B} = \mu_0 \mathbf{J}$
	$\nabla \cdot \mathbf{B} = 0$

Simplest macroscopic plasma fluid model, assumes low-frequency, long wavelength, neglects e^- inertia

$$\rho = m_i n_i$$
$$\mathbf{v} = \mathbf{u}_i$$

$$L \gg \lambda_D$$
$$n_e \approx n_i$$

(Freidberg 2014)

(Hirshman and Whitson, 1983)

VMEC - Coordinate System

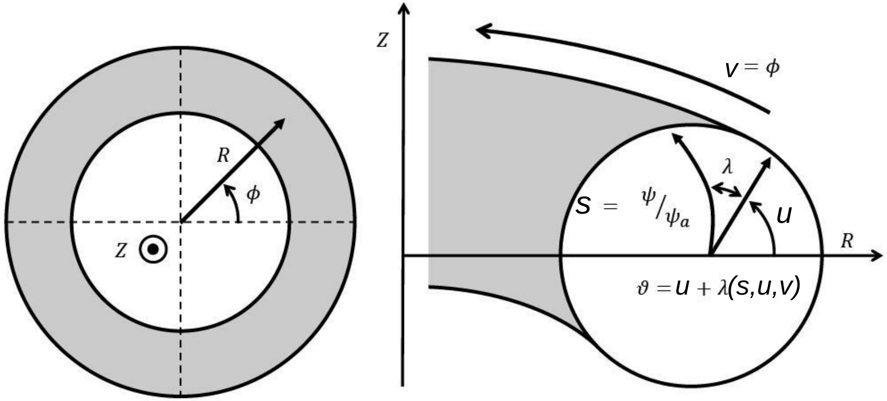
$\rho \rightarrow s$	Flux Surface Label
$\theta \rightarrow u$	Poloidal Angle
ϑ	SFL Poloidal Angle
$\phi \rightarrow v$	Geometric Toroidal Angle

Geometry represented as **Fourier Series** in (u, v) **on each discrete surface**

$$R(s,u,v) = \sum_{m=0,n=-N}^{M,N} R_{mn,c}(s)cos(mu - nvN_{FP}) + R_{mn,s}(s)sin(mu - nvN_{FP})$$

$$\lambda(s,u,v) = \sum_{m=0,n=-N}^{M,N} \lambda_{mn,c}(s)cos(mu - nvN_{FP}) + \lambda_{mn,s}(s)sin(mu - nvN_{FP})$$

$$Z(s,u,v) = \sum_{m=0,n=-N}^{M,N} Z_{mn,c}(s)cos(mu - nvN_{FP}) + Z_{mn,s}(s)sin(mu - nvN_{FP})$$



(Dudt and Kolemen 2020)

DESC - Coordinate System

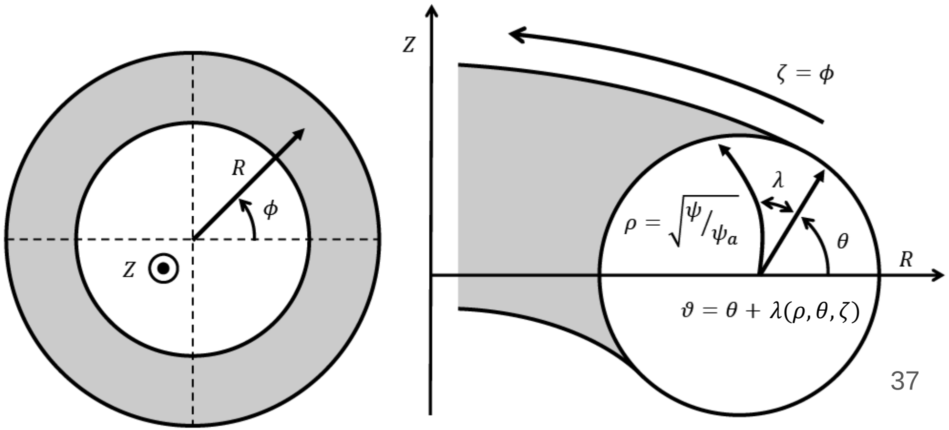
ρ	Flux Surface Label
θ	Poloidal Angle
ϑ	SFL Poloidal Angle
ϕ	Geometric Toroidal Angle

Geometry represented **continuously** with global basis functions

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

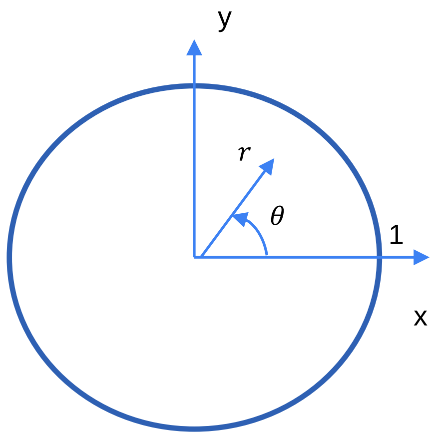
$$\lambda(\rho,\theta,\zeta) = \sum_{m=-M,n=-N,l=0}^{M,N,L} \lambda_{lmn}\mathcal{Z}_l^m(\rho,\theta)\mathcal{F}^n(\zeta)$$

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



[1](Lewis and Bellan, 1990)

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 polynomial in (*x*,*y*)
- We can rewrite $f(r,\theta)$ as

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

Regular at r=0

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

Must be regular at r=0!

Regular at r=0

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

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

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