Full flux surface (FFS) δf-gyrokinetic code: `stella`

G. O. Acton[1,2], M. Barnes[1], S. Newton[2], H. Thienpondt[5], F. Parra[3], W. Dorland[4]

[1] Rudolf Peierls Centre For Theoretical Physics,
University of Oxford, Oxford, OX1 3PU, UK

[2] Culham Centre for Fusion Energy,
United Kingdom Atomic Energy Authority,
Abingdon, OX14 3EB, UK

[3] Princeton Plasma Physics Laboratory, Princeton, NJ 08543, USA

[4] Department of Physics, University of Maryland, College Park, MD 20742, United States of America

[5] Laboratorio Nacional de Fusion, CIEMAT, 28040 Madrid, Spain

EFTC 2023
Padova

Table of Contents

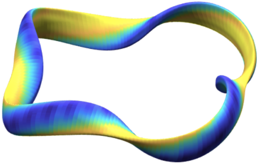
Table of Contents

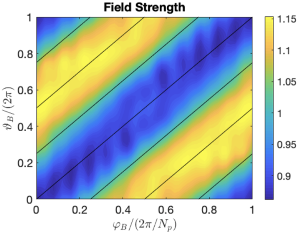
Motivation

- Stellarators can be neoclasically optimised

Motivation

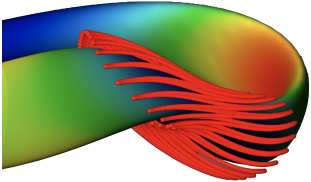
- Stellarators are neoclassically optimised
- Demand omnigeneity
- Require time-averaged radial magnetic drifts away from flux surface to vanish for all particles
- Particle orbits and neoclassical transport are the same in quasisymmetric devices as in truly axisymmetric ones
- "Unwrap" stellarator with certain transformation and magnetic field looks the same to particles

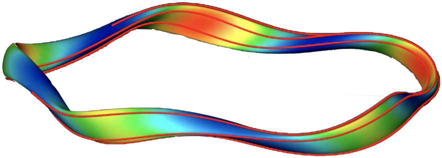




Motivation

- Stellarators are neoclassically optimised
- Demand omnigeneity
- Require time-averaged radial magnetic drifts away from flux surface to vanish for all particles
- Particle orbits and neoclassical transport are the same in quasisymmetric devices as in truly axisymmetric ones



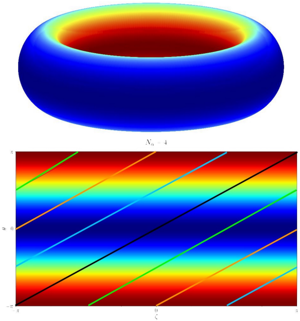


Motivation

- Stellarators are neoclassically optimised
- Hence, turbulent transport dominates → model using δf-gyrokinetics

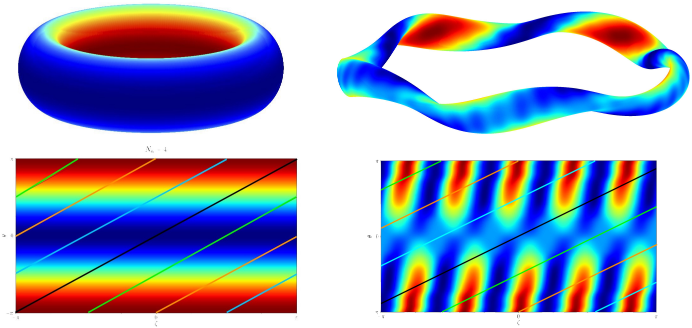
Motivation

- Stellarators are neoclassically optimised
- Hence, turbulent transport dominates → model using δf-gyrokinetics
- Axisymmetry means all field lines are equivalent



Motivation

- Stellarators are neoclassically optimised
- Hence, turbulent transport dominates → model using δf-gyrokinetics
- Axisymmetry means all field lines are equivalent
- Stellarator magnetic geometry varies with field line. Single field line is **not** sufficient!



Motivation

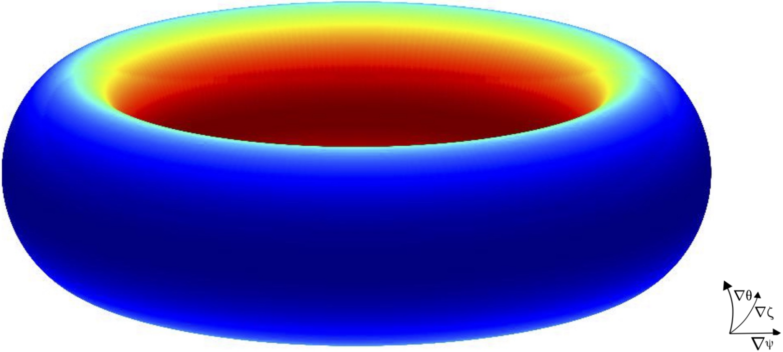
- Stellarators are neoclassically optimised
- Hence, turbulent transport dominates → model using δf-gyrokinetics
- Axisymmetry means all field lines are equivalent
- Stellarator magnetic geometry varies with field line. Single field line is **not** sufficient!
- Modes on different field lines interact → complicates algorithms due to α-inhomogeneity

Motivation

- Stellarators are neoclassically optimised
- Hence, turbulent transport dominates $\rightarrow$ model using δf-gyrokinetics
- Axisymmetry means all field lines are equivalent
- Stellarator magnetic geometry varies with field line. Single field line is **not** sufficient!
- Modes on different field lines interact $\rightarrow$ complicates algorithms due to α-inhomogeneity
- 3 main approaches to model turbulence:
    - real space
    - flux tube
    - full flux annulus

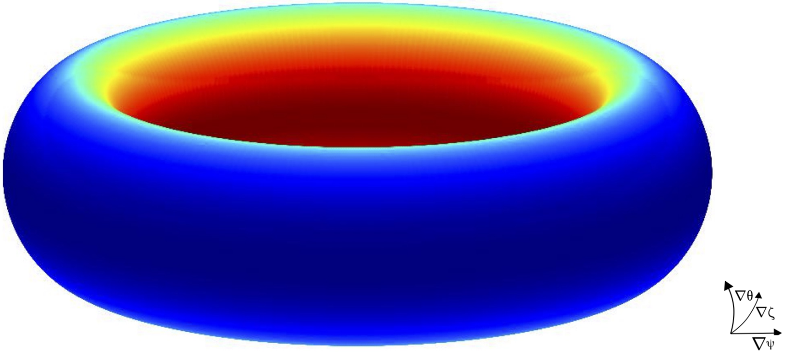
Table of Contents

Real space 3D formalism



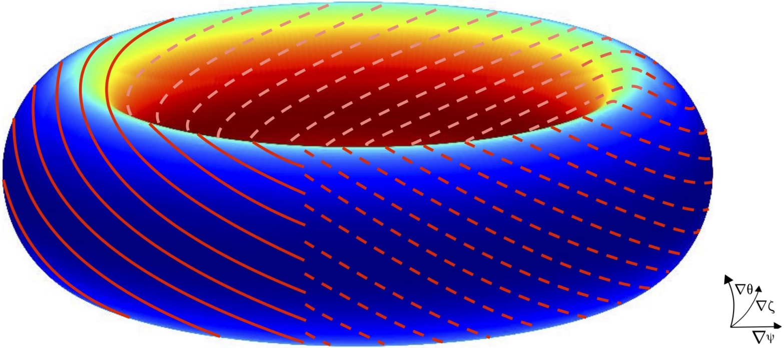
- Full device simulation in real space

Real space 3D formalism



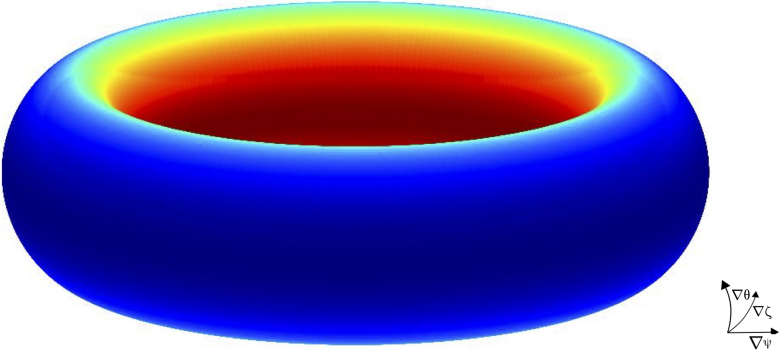
- Full device simulation in real space
- Initialise at $t = 0$ across whole device and evolve globally according to GK equation
- Use finite difference schemes to take derivatives
- Impose periodicity in ζ

Real space 3D formalism: toroidal boundary conditions



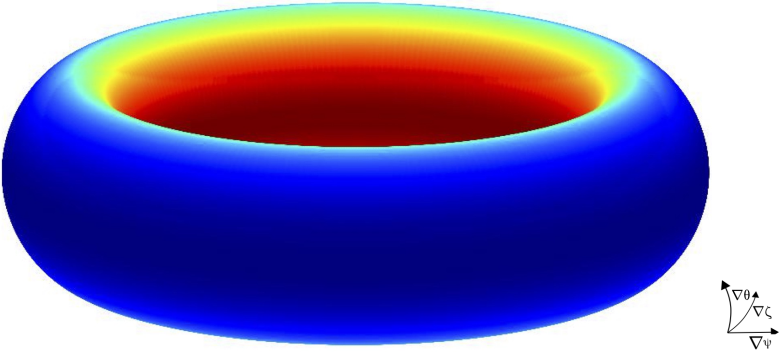
- Want domain to be 2π-periodic in ζ
- Field lines on a non-rational surface will not close on each other
- Need to interpolate field lines back onto ones which lie on our α-grid

Real space 3D formalism: pros and cons



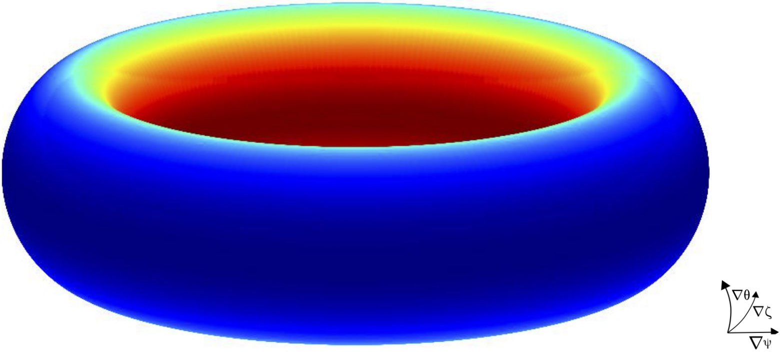
- Capable of modelling full device - good for benchmarking codes

Real space 3D formalism: pros and cons



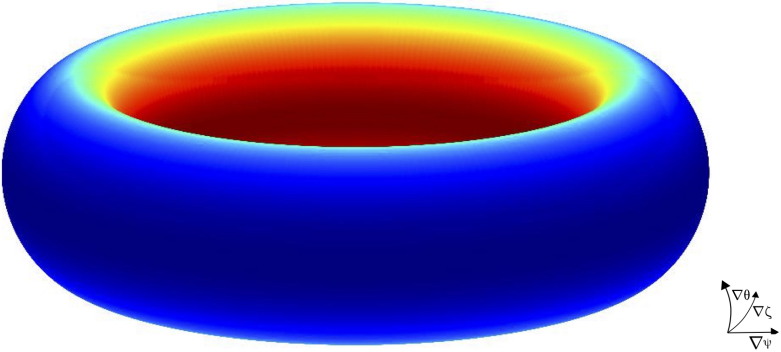
- Capable of modelling full device - good for benchmarking codes
- Can be very computationally expensive
  - Need high resolution to capture gyro-orbit effects
  - Can take months on multiple CPU cores

Real space 3D formalism: pros and cons



- Capable of modelling full device - good for benchmarking codes
- Can be very computationally expensive
  - Need high resolution to capture gyro-orbit effects
  - Can take months on multiple CPU cores
- Loose spectral accuracy in derivatives

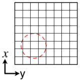
Real space 3D formalism: pros and cons



- Capable of modelling full device - good for benchmarking codes
- Can be very computationally expensive
  - Need high resolution to capture gyro-orbit effects
  - Can take months on multiple CPU cores
- Loose spectral accuracy in derivatives
- Radial boundary conditions are difficult to choose

Table of Contents

Flux tube formalism



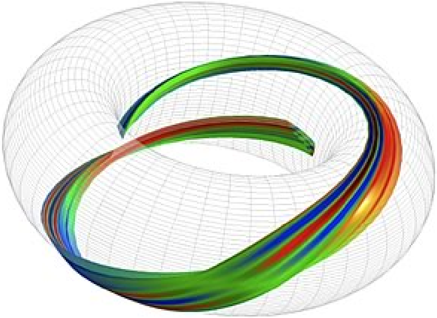
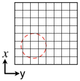


Image: SCD CCP-Plasma

- Simulation coordinates: $(x, y, z) \rightarrow (\psi, \alpha, \zeta)$

Flux tube formalism



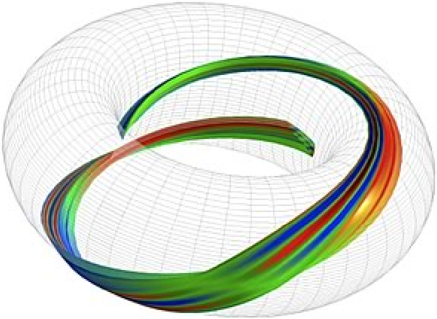


Image: SCD CCP-Plasma

- Simulation coordinates: $(x, y, z) \rightarrow (\psi, \alpha, \zeta)$
- Initialise some δf and ϕ at $t = 0$ on simulation domain

Flux tube formalism

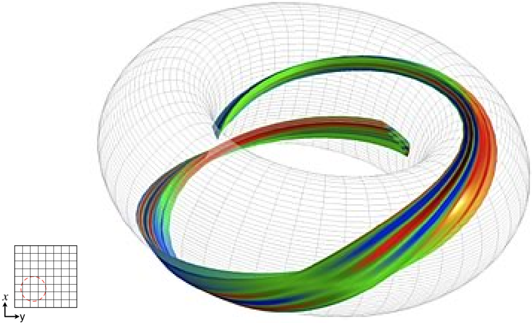


Image: SCD CCP-Plasma

- Simulation coordinates: $(x, y, z) \to (\psi, \alpha, \zeta)$
- Initialise some δf and ϕ at $t = 0$ on simulation domain
- Evolve gyrokinetic equation pseudo-spectrally

Flux tube formalism

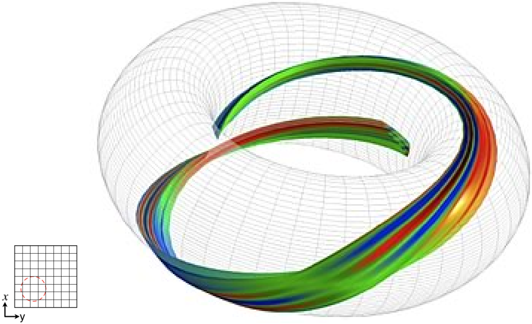


Image: SCD CCP-Plasma

- Simulation coordinates: $(x, y, z) \to (\psi, \alpha, \zeta)$
- Initialise some δf and ϕ at $t = 0$ on simulation domain
- Evolve gyrokinetic equation pseudo-spectrally
  - Decay in $v_\parallel$; $g(t, \boldsymbol{x}, v_\parallel \to \pm\infty, \mu) \to 0$
  - Turbulence is taken as periodic in perpendicular directions, $k_x, k_y \gg 1/L$
  - Use twist-and-shift boundary conditions in z to capture extended modes

Flux tube formalism: twist-and-shift

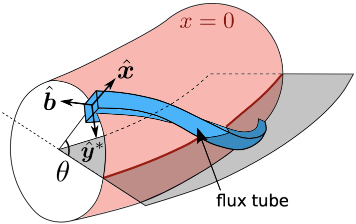


Image: Nicolas Christen, Bistable turbulent transport in fusion plasmas with rotational shear (2021)

Flux tube formalism: twist-and-shift

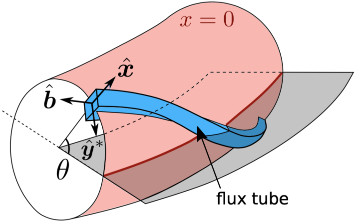
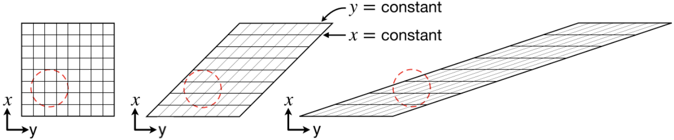


Image: Nicolas Christen, Bistable turbulent transport in fusion plasmas with rotational shear (2021)



- If $\hat{s} \propto \mathrm{d}q/\mathrm{d}\psi \neq 0$ then domain gets sheared as it travels around device

Flux tube formalism: twist-and-shift

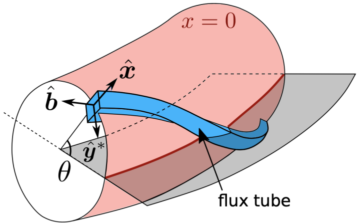
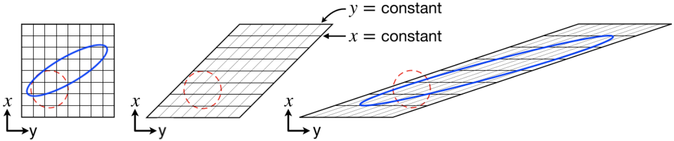
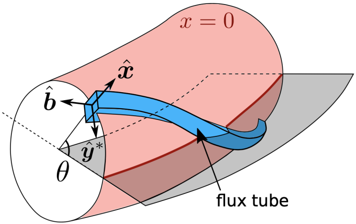


Image: Nicolas Christen, Bistable turbulent transport in fusion plasmas with rotational shear (2021)



- If $\hat{s} \propto \mathrm{d}q/\mathrm{d}\psi \neq 0$ then domain gets sheared as it travels around device
- Eddies get sheared
- Pushed to higher perpendicular wavenumbers

Flux tube formalism: twist-and-shift



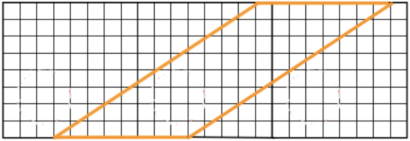
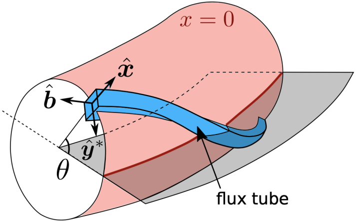


Image: Nicolas Christen, Bistable turbulent transport in fusion plasmas with rotational shear (2021)

- Use "twist-and-shift" boundary conditions to map sheared domain back onto original one

Flux tube formalism: twist-and-shift



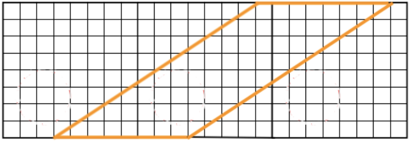
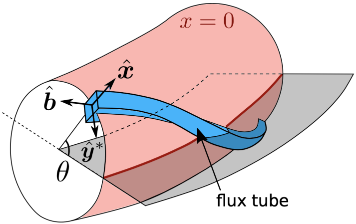


Image: Nicolas Christen, Bistable turbulent transport in fusion plasmas with rotational shear (2021)

- Use "twist-and-shift" boundary conditions to map sheared domain back onto original one
- "Twist-and-shift" is the Fourier equivalent of the real-space boundary condition *

Flux tube formalism: twist-and-shift



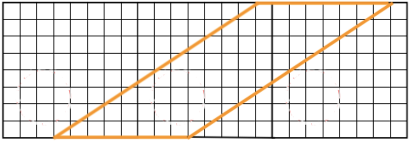


Image: Nicolas Christen, Bistable turbulent transport in fusion plasmas with rotational shear (2021)

- Use "twist-and-shift" boundary conditions to map sheared domain back onto original one
- "Twist-and-shift" is the Fourier equivalent of the real-space boundary condition *
- Enforce periodicity after one geometric turn:

$$\hat{A}_{\boldsymbol{k}}(z) = \hat{A}_{\boldsymbol{k}'}(z + 2p\pi)(\text{phase factor}) \tag{1}$$

Flux tube formalism

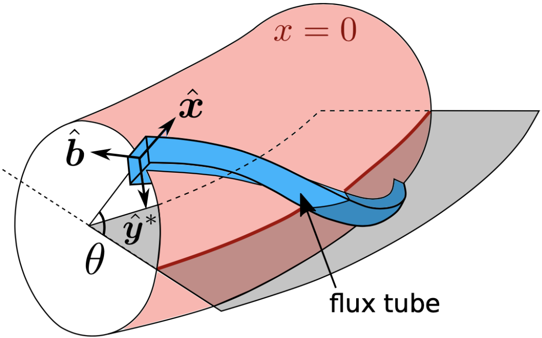


Image: Nicolas Christen, Bistable turbulent transport in fusion plasmas with rotational shear (2021)

- Flux-tube simulations are sufficient for tokamaks because we can stitch our flux tubes together

Flux tube formalism

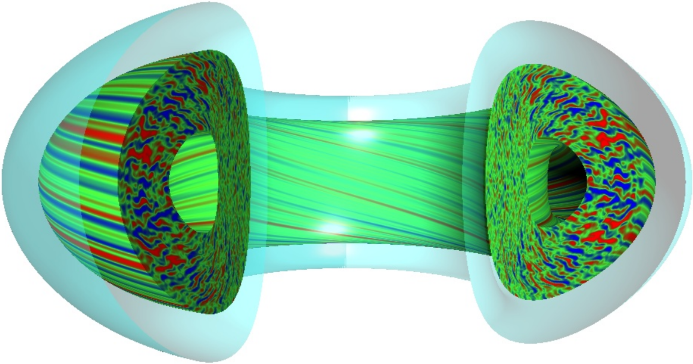


Image: J.Candy, Waltz, GYRO simulation of DIII-D

- Flux-tube simulations are sufficient for tokamaks because we can stitch our flux tubes together

Flux tube formalism: pros and cons

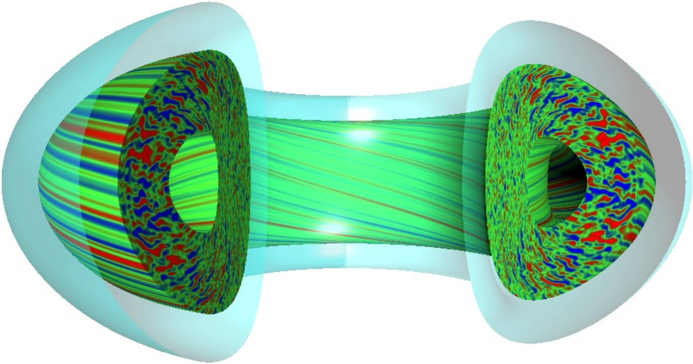


Image: J.Candy, Waltz, GYRO simulation of DIII-D

- Very fast codes which yield quick results - can perform many simulations in quick succession

Flux tube formalism: pros and cons

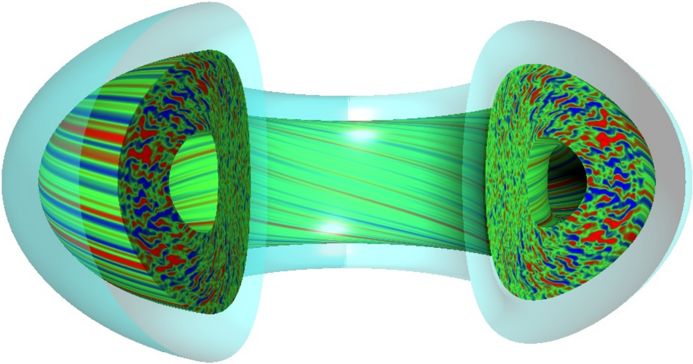


Image: J.Candy, Waltz, GYRO simulation of DIII-D

- Very fast codes which yield quick results - can perform many simulations in quick succession
- Easy to interpret - normal modes are well defined in this system

Flux tube formalism: pros and cons

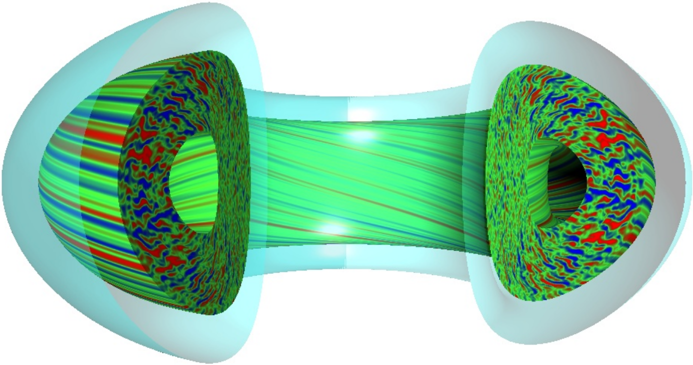


Image: J.Candy, Waltz, GYRO simulation of DIII-D

- Very fast codes which yield quick results - can perform many simulations in quick succession
- Easy to interpret - normal modes are well defined in this system
- Retains spectral accuracy in spacial derivatives

Flux tube formalism: pros and cons

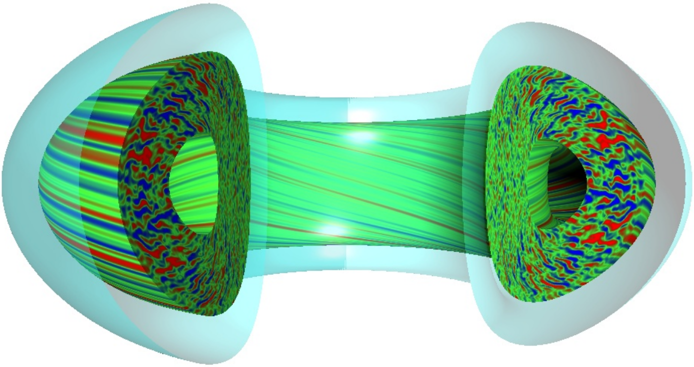


Image: J.Candy, Waltz, GYRO simulation of DIII-D

- Very fast codes which yield quick results - can perform many simulations in quick succession
- Easy to interpret - normal modes are well defined in this system
- Retains spectral accuracy in spacial derivatives
- Does not capture global effects like coupling between different field lines

Table of Contents

Flux annulus formalism: motivation

- Stellarator geometry varies with field line
- Method of stitching together flux tubes no longer holds

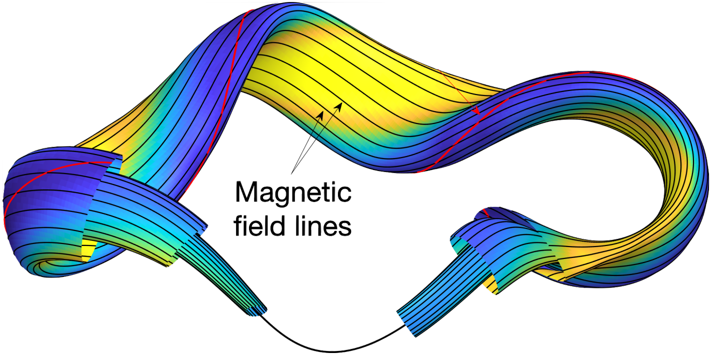
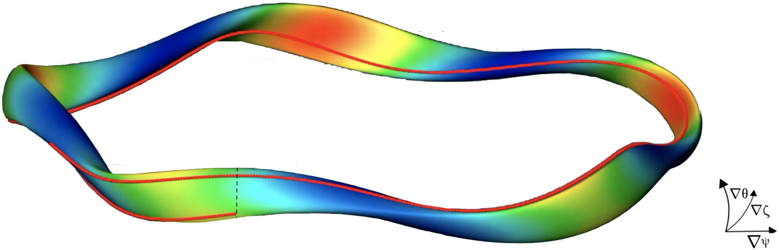


Image: UMD stellarator group

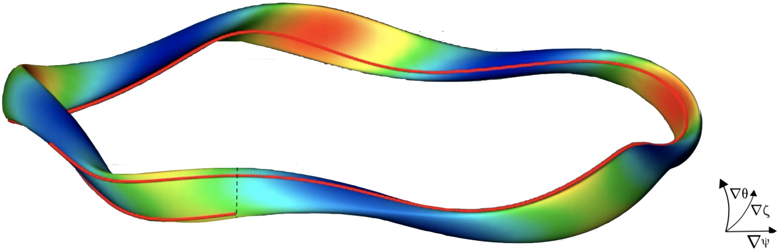
Flux annulus framework

- Domain is now 2π in ζ, not 2π in θ



Flux annulus framework

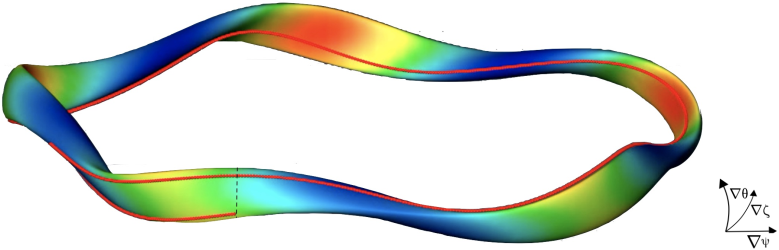
- Domain is now 2π in ζ, not 2π in θ



- Evolve pseudo-spectrally to retain spectral accuracy in derivatives

Flux annulus framework

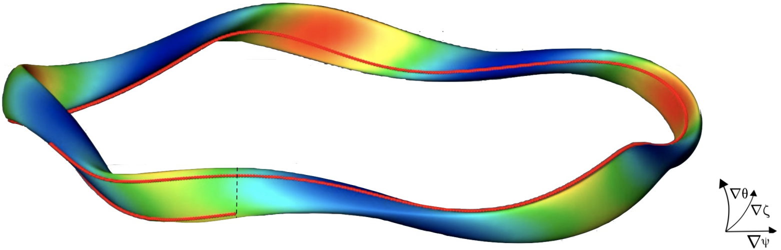
- Domain is now 2π in ζ, not 2π in θ



- Evolve pseudo-spectrally to retain spectral accuracy in derivatives
- Simulate N_y field lines, which cover different geometry and are now coupled together

Flux annulus framework

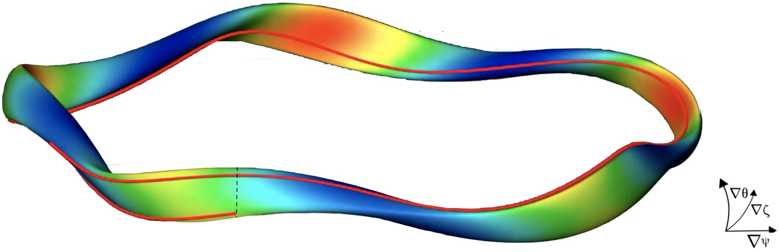
- Domain is now 2π in ζ, not 2π in θ



- Evolve pseudo-spectrally to retain spectral accuracy in derivatives
- Simulate N_y field lines, which cover different geometry and are now coupled together
- Want to match every incoming field line to its connecting field line - apply twist-and-shift to entire poloidal domain

Flux annulus framework

- Domain is now 2π in ζ, not 2π in θ



- Evolve pseudo-spectrally to retain spectral accuracy in derivatives
- Simulate N_y field lines, which cover different geometry and are now coupled together
- Want to match every incoming field line to its connecting field line - apply twist-and-shift to entire poloidal domain
- ρ_* now becomes an important physical parameter in simulations $\rightarrow$ determines Δk_y

How 3D geometry modifies equations

- Geometry is no longer trivial
- But how does geometry enter our code?

How 3D geometry modifies equations

- Geometry is no longer trivial
- But how does geometry enter our code?

Simplified notation GK:

$$\frac{\partial g}{\partial t} = (\text{geometric factors}) \cdot (\boldsymbol{\nabla} g + \boldsymbol{\nabla} \langle \phi \rangle_{\boldsymbol{R}}) \tag{2}$$

How 3D geometry modifies equations

- Geometry is no longer trivial
- But how does geometry enter our code?

Simplified notation GK:

$$\frac{\partial g}{\partial t} = \underbrace{(\text{geometric factors})}_{\text{e.g } \hat{\boldsymbol{b}}\cdot\boldsymbol{\nabla} z} \cdot \left(\boldsymbol{\nabla} g + \boldsymbol{\nabla} \underbrace{\langle \phi \rangle_{\boldsymbol{R}}}_{J_{0,\boldsymbol{k}}\hat{\phi}_{\boldsymbol{k}}}\right) \tag{2}$$

- Bessel functions $J_0(a_{\boldsymbol{k}})$ with $a_{\boldsymbol{k}} = \frac{k_\perp v_\perp}{\Omega_s}$
- Geometric factors are α-dependent

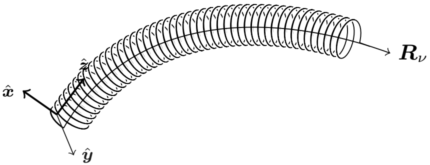
How 3D geometry modifies equations

- Geometry is no longer trivial
- But how does geometry enter our code?

Simplified notation GK:

$$\frac{\partial g}{\partial t} = \underbrace{(\text{geometric factors})}_{\text{e.g } \hat{\boldsymbol{b}} \cdot \nabla z} \cdot (\nabla g + \nabla \underbrace{\langle \phi \rangle_{\boldsymbol{R}}}_{J_{0,\boldsymbol{k}} \hat{\phi}_{\boldsymbol{k}}}) \tag{2}$$

- Bessel functions $J_0(a_{\boldsymbol{k}})$ with $a_{\boldsymbol{k}} = \frac{k_\perp v_\perp}{\Omega_s}$ $\leftarrow$ $k_\perp$ and B in argument
- Geometric factors are α-dependent
- Gyro-averaging introduces coupling between different k_y-modes $\rightarrow$ no longer a local operation



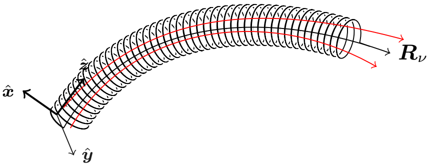
How 3D geometry modifies equations

- Geometry is no longer trivial
- But how does geometry enter our code?

Simplified notation GK:

$$\frac{\partial g}{\partial t} = \underbrace{\text{(geometric factors)}}_{\text{e.g } \hat{\boldsymbol{b}} \cdot \boldsymbol{\nabla} z} \cdot \left(\boldsymbol{\nabla} g + \boldsymbol{\nabla} \underbrace{\langle \phi \rangle_{\boldsymbol{R}}}_{J_{0,\boldsymbol{k}} \hat{\phi}_{\boldsymbol{k}}} \right) \quad (2)$$

- Bessel functions $J_0(a_{\boldsymbol{k}})$ with $a_{\boldsymbol{k}} = \frac{k_\perp v_\perp}{\Omega_s}$ $\leftarrow$ $k_\perp$ and B in argument
- Geometric factors are α-dependent
- Gyro-averaging introduces coupling between different k_y-modes $\rightarrow$ no longer a local operation



How 3D geometry modifies equations

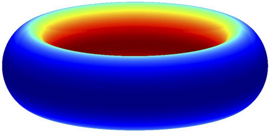
- Geometry is no longer trivial
- But how does geometry enter our code?

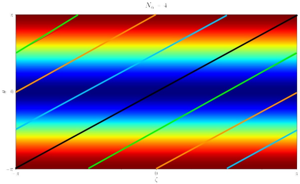
Simplified notation GK:

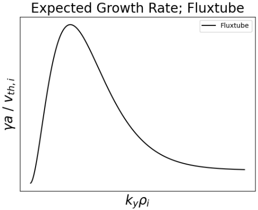
$$\frac{\partial g}{\partial t} = \underbrace{(\text{geometric factors})}_{\text{e.g } \hat{\boldsymbol{b}}\cdot\boldsymbol{\nabla} z} \cdot \left(\boldsymbol{\nabla} g + \boldsymbol{\nabla} \underbrace{\langle\phi\rangle_{\boldsymbol{R}}}_{J_{0,\boldsymbol{k}}\hat{\phi}_{\boldsymbol{k}}}\right) \quad (2)$$

- Bessel functions $J_0(a_{\boldsymbol{k}})$ with $a_{\boldsymbol{k}} = \frac{k_\perp v_\perp}{\Omega_s}$
- Geometric factors are α-dependent
- Gyro-averaging introduces coupling between different k_y-modes $\rightarrow$ no longer a local operation
- α-inhomogeneity leads to convolutions

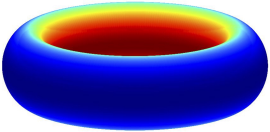
How 3D geometry modifies expectations

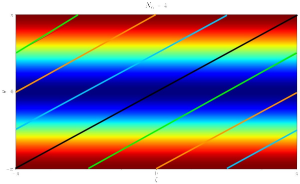


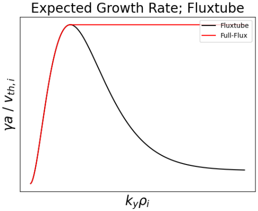




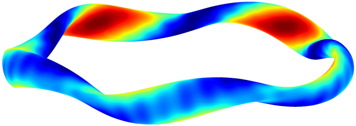
How 3D geometry modifies expectations

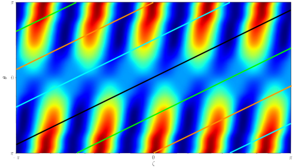






How 3D geometry modifies expectations





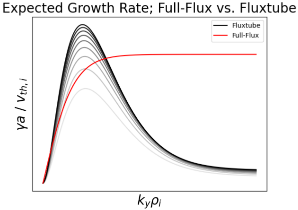


Table of Contents

Algorithm

- Use operator splitting to solve normalised GK equation:

Algorithm

- Use operator splitting to solve normalised GK equation:

$$\frac{\partial g_\nu}{\partial t} + \underbrace{\underbrace{\mathcal{S}_\nu[g_\nu,\varphi_\nu]}_{\text{streaming}} + \underbrace{\mathcal{M}_\nu[g_\nu]}_{\text{mirror}}}_{\text{Implicit}} + \underbrace{\underbrace{\mathcal{D}_\nu[g_\nu,\varphi_\nu] + \mathcal{G}_\nu[\varphi_\nu]}_{\text{drifts}} + \underbrace{\mathcal{N}_\nu[g_\nu,\varphi_\nu]}_{\text{non-linear}} = \underbrace{\mathcal{C}_\nu[\{g_{\nu'}\},\{\varphi_{\nu'}\}]}_{\text{collisions}}}_{\text{Explicit}}, \tag{3}$$

$$\partial_t g_\nu = \sum_{i=1}^{3} (\partial_t g_\nu)_i \quad \left| \quad \begin{aligned} (\partial_t g_\nu)_1 + \mathcal{D}_\nu[g_\nu,\varphi_\nu] + \mathcal{G}_\nu[\varphi_\nu] + \mathcal{N}_\nu[g_\nu,\varphi_\nu] &= 0 \\ (\partial_t g_\nu)_2 + \mathcal{M}_\nu[g_\nu] &= 0 \\ (\partial_t g_\nu)_3 + \mathcal{S}_\nu[g_\nu,\varphi_\nu] &= 0 \end{aligned} \right.$$

Algorithm

- Use operator splitting to solve normalised GK equation:

$$\frac{\partial g_\nu}{\partial t} + \underbrace{\underbrace{\mathcal{S}_\nu[g_\nu,\varphi_\nu]}_{\text{streaming}} + \underbrace{\mathcal{M}_\nu[g_\nu]}_{\text{mirror}}}_{\text{Implicit}} + \underbrace{\underbrace{\mathcal{D}_\nu[g_\nu,\varphi_\nu] + \mathcal{G}_\nu[\varphi_\nu]}_{\text{drifts}} + \underbrace{\mathcal{N}_\nu[g_\nu,\varphi_\nu]}_{\text{non-linear}} = \underbrace{\mathcal{C}_\nu[\{g_{\nu'}\},\{\varphi_{\nu'}\}]}_{\text{collisions}}}_{\text{Explicit}}, \tag{3}$$

$$\partial_t g_\nu = \sum_{i=1}^{3}(\partial_t g_\nu)_i \quad \left| \quad \begin{aligned} (\partial_t g_\nu)_1 + \mathcal{D}_\nu[g_\nu,\varphi_\nu] + \mathcal{G}_\nu[\varphi_\nu] + \mathcal{N}_\nu[g_\nu,\varphi_\nu] &= 0 \\ (\partial_t g_\nu)_2 + \mathcal{M}_\nu[g_\nu] &= 0 \\ (\partial_t g_\nu)_3 + \mathcal{S}_\nu[g_\nu,\varphi_\nu] &= 0 \end{aligned} \right.$$

- Geometric coefficients introduce coupling between different k_α. For example, the gyroaverage $\varphi_\nu = \langle\phi\rangle_{\mathbf{R}_\nu}$

Flux tube:

$$\hat{\varphi}_{\mathbf{k},\nu} = J_{0,(k_\psi,k_\alpha),\nu}\phi_{(k_\psi,k_\alpha)}$$

Full flux annulus: *

$$\hat{\varphi}_{\mathbf{k},\nu} = \sum_{k'_\alpha} \underbrace{\hat{J}_{(k_\psi,k_\alpha-k'_\alpha),k'_\alpha,\nu}}_{matrix} \hat{\phi}_{(k_\psi,k_\alpha-k'_\alpha)},$$

- Compute Fourier coefficients, $\hat{J}_{\mathbf{k}'',k'_\alpha,\nu}$, of $J_{0,(k_\psi,k_\alpha),\nu}$ once at the beginning of simulation for computational efficiency.

Implicit treatment

- `stella` is a fast GK code

Implicit treatment

- `stella` is a fast GK code
- Electron dynamics imposes stringent CFL condition on time step $\rightarrow$ treat parallel streaming and mirror terms implicitly

Implicit treatment

- **stella** is a fast GK code
- Electron dynamics imposes stringent CFL condition on time step $\rightarrow$ treat parallel streaming and mirror terms implicitly
- However, geometric-dependent coefficients add inhomogeneous α-dependence

$$\frac{\partial g_\nu}{\partial t} = - \underbrace{v_\parallel \hat{\boldsymbol{b}} \cdot \boldsymbol{\nabla} z}_{\alpha\text{-dependent}} \underbrace{\left(\frac{\partial g_\nu}{\partial z} + \frac{Z_\nu e}{T_\nu} \frac{\partial \varphi_s}{\partial z} F_{0,\nu} \right)}_{\alpha\text{-dependent}}. \tag{4}$$

Implicit treatment

- `stella` is a fast GK code
- Electron dynamics imposes stringent CFL condition on time step $\rightarrow$ treat parallel streaming and mirror terms implicitly
- Circumvent by splitting into implicit and explicit contributions:

$$\frac{\partial g_\nu}{\partial t} = \underbrace{- v_\parallel \overline{\hat{\boldsymbol{b}} \cdot \boldsymbol{\nabla} z} \left(\frac{\partial g_\nu}{\partial z} + \frac{Z_\nu e}{T_\nu} \frac{\partial \bar{J} \bar{\phi}}{\partial z} \bar{F}_{0,\nu} \right)}_{\text{Implicit}} \underbrace{- v_\parallel \overline{\hat{\boldsymbol{b}} \cdot \boldsymbol{\nabla} z} \frac{Z_\nu e}{T_\nu} \left[\frac{\partial \varphi_\nu}{\partial z} F_{0,\nu} - \frac{\partial \bar{J} \bar{\phi}}{\partial z} \bar{F}_{0,\nu} \right]}_{\text{Explicit -- source to implicit equation}}$$
$$\underbrace{- v_\parallel \left(\hat{\boldsymbol{b}} \cdot \boldsymbol{\nabla} z - \overline{\hat{\boldsymbol{b}} \cdot \boldsymbol{\nabla} z} \right) \left(\frac{\partial g_\nu}{\partial z} + \frac{Z_\nu e}{T_\nu} \frac{\partial \varphi_\nu}{\partial z} F_{0,\nu} \right)}_{\text{Explicit}} \tag{4}$$

Implicit treatment

- `stella` is a fast GK code
- Electron dynamics imposes stringent CFL condition on time step → treat parallel streaming and mirror terms implicitly
- Circumvent by splitting into implicit and explicit contributions:

$$\frac{\partial g_\nu}{\partial t} = \underbrace{-v_\parallel \overline{\hat{\boldsymbol{b}} \cdot \boldsymbol{\nabla} z} \left(\frac{\partial g_\nu}{\partial z} + \frac{Z_\nu e}{T_\nu} \frac{\partial \bar{J} \bar{\phi}}{\partial z} \bar{F}_{0,\nu} \right)}_{\text{Implicit}} \underbrace{- v_\parallel \overline{\hat{\boldsymbol{b}} \cdot \boldsymbol{\nabla} z} \frac{Z_\nu e}{T_\nu} \left[\frac{\partial \varphi_\nu}{\partial z} F_{0,\nu} - \frac{\partial \bar{J} \bar{\phi}}{\partial z} \bar{F}_{0,\nu} \right]}_{\text{Explicit – source to implicit equation}}$$
$$\underbrace{- v_\parallel \left(\hat{\boldsymbol{b}} \cdot \boldsymbol{\nabla} z - \overline{\hat{\boldsymbol{b}} \cdot \boldsymbol{\nabla} z} \right) \left(\frac{\partial g_\nu}{\partial z} + \frac{Z_\nu e}{T_\nu} \frac{\partial \varphi_\nu}{\partial z} F_{0,\nu} \right)}_{\text{Explicit}} \tag{4}$$

- Here $\overline{\hat{\boldsymbol{b}} \cdot \boldsymbol{\nabla} z} \doteq \frac{1}{2\pi} \int_0^{2\pi} d\alpha \left(\frac{1}{2\pi} \int_{\zeta_{\min}}^{\zeta_{\max}} \frac{d\zeta'}{\hat{\boldsymbol{b}} \cdot \nabla \zeta'} \right)^{-1}$ is an average over fieldlines
- $\bar{J} = \hat{J}_{(k_\psi, 0), k_\alpha}$ is the constant-in-alpha component of the Bessel function for a given k_α
- $\bar{\phi}$ is an artificial field that solves a modified quasineutrality condition (i.e. $\bar{\phi}$ is a contribution to ϕ that is treated implicitly) *

Implicit treatment

- `stella` is a fast GK code
- Electron dynamics imposes stringent CFL condition on time step $\rightarrow$ treat parallel streaming and mirror terms implicitly
- Circumvent by splitting into implicit and explicit contributions:

$$\frac{\partial g_\nu}{\partial t} = \underbrace{-v_\parallel \overline{\hat{\boldsymbol{b}} \cdot \boldsymbol{\nabla} z} \left(\frac{\partial g_\nu}{\partial z} + \frac{Z_\nu e}{T_\nu} \frac{\partial \bar{J} \bar{\phi}}{\partial z} \bar{F}_{0,\nu} \right)}_{\text{Implicit}} \underbrace{- v_\parallel \overline{\hat{\boldsymbol{b}} \cdot \boldsymbol{\nabla} z} \frac{Z_\nu e}{T_\nu} \left[\frac{\partial \varphi_\nu}{\partial z} F_{0,\nu} - \frac{\partial \bar{J} \bar{\phi}}{\partial z} \bar{F}_{0,\nu} \right]}_{\text{Explicit -- source to implicit equation}}$$
$$\underbrace{- v_\parallel \left(\hat{\boldsymbol{b}} \cdot \boldsymbol{\nabla} z - \overline{\hat{\boldsymbol{b}} \cdot \boldsymbol{\nabla} z} \right) \left(\frac{\partial g_\nu}{\partial z} + \frac{Z_\nu e}{T_\nu} \frac{\partial \varphi_\nu}{\partial z} F_{0,\nu} \right)}_{\text{Explicit}} \qquad (4)$$

- Here $\overline{\hat{\boldsymbol{b}} \cdot \boldsymbol{\nabla} z} \doteq \frac{1}{2\pi} \int_0^{2\pi} d\alpha \left(\frac{1}{2\pi} \int_{\zeta_{\min}}^{\zeta_{\max}} \frac{d\zeta'}{\hat{\boldsymbol{b}} \cdot \nabla \zeta'} \right)^{-1}$ is an average over field lines
- $\bar{J} = \hat{J}_{(k_\psi, 0), k_\alpha}$ is the constant-in-alpha component of the Bessel function for a given k_α
- $\bar{\phi}$ is an artificial field that solves a modified quasineutrality condition (i.e. $\bar{\phi}$ is a contribution to ϕ that is treated implicitly) *

Table of Contents

Expectations and results

- Comparing code with axisymmetric geometry

Expectations and results

- Comparing code with axisymmetric geometry
- CBC looks like stellarator in non-tokamak coordinates
- Flux tube and FFS simulations should agree as $N_y \to \infty$ and $\rho_* \to 0$

Expectations and results

- Comparing code with axisymmetric geometry
- CBC looks like stellarator in non-tokamak coordinates
- Flux tube and FFS simulations should agree as $N_y \to \infty$ and $\rho_* \to 0$

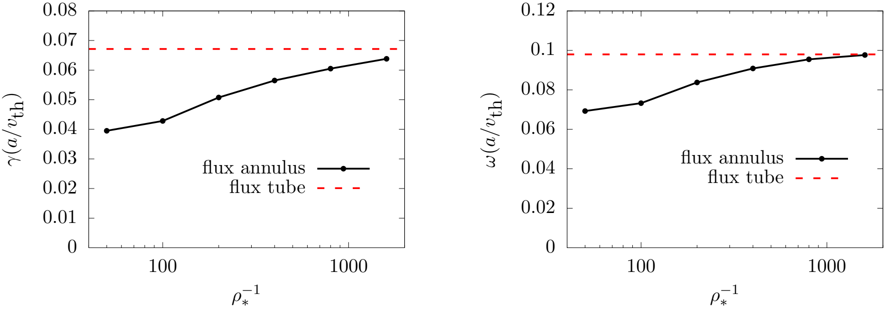
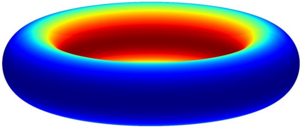


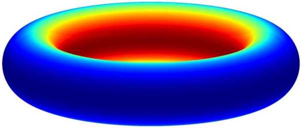
Figure 1: ρ_* scans in CBC with adiabatic electrons

Simulation results: CBC, adiabatic electrons



- Comparing code with axisymmetric geometry
- Use modified Boltzmann response $\delta n_e = \frac{e n_e}{T_e} \left(\phi - \langle \phi \rangle_{\mathrm{FSA}}\right)$, where $\langle \phi \rangle_{\mathrm{FSA}}$ is the flux-surface-averaged ϕ

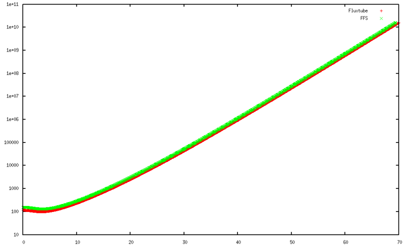
Simulation results: CBC, adiabatic electrons



- Comparing code with axisymmetric geometry
- Use modified Boltzmann response $\delta n_e = \frac{e n_e}{T_e}\left(\phi - \langle\phi\rangle_{\mathrm{FSA}}\right)$, where $\langle\phi\rangle_{\mathrm{FSA}}$ is the flux-surface-averaged ϕ

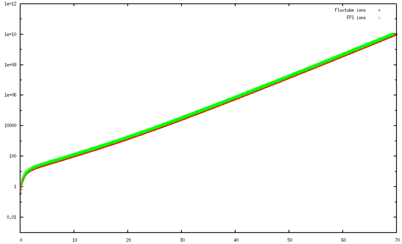
<u>Linear simulations</u>

$\log|\phi|^2$



$tv_{th,i}/a$

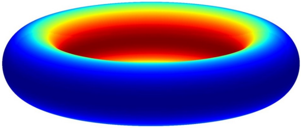
$Q/Q_{\mathbf{GB}}$



$tv_{th,i}/a$

Figure 2: Linear simulations for CBC with modified electron response; $N_y = N_x = 30$, $\rho_* = 0.025$

Simulation results: CBC, adiabatic electrons



- Comparing code with axisymmetric geometry
- Use modified Boltzmann response $\delta n_e = \frac{e n_e}{T_e}\left(\phi - \langle\phi\rangle_{\mathrm{FSA}}\right)$, where $\langle\phi\rangle_{\mathrm{FSA}}$ is the flux-surface-averaged ϕ

<u>Non-linear simulations</u>

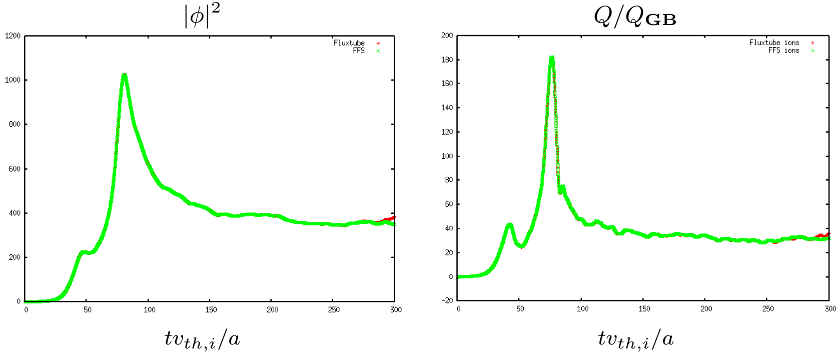
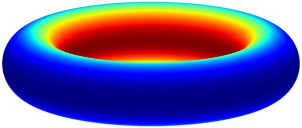


Figure 2: Non-linear simulations for CBC; $N_y = N_x = 30$, $\rho_* = 0.025$

Simulation results: CBC, kinetic electrons

- Add in kinetic electrons

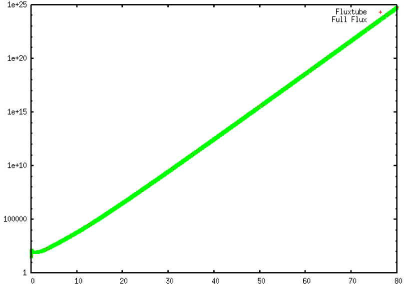
Simulation results: CBC, kinetic electrons



- Add in kinetic electrons

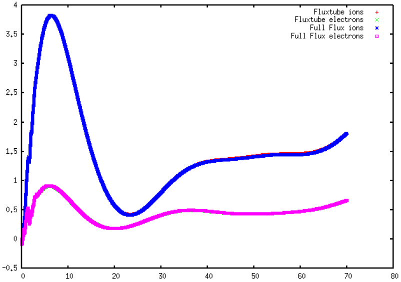
Linear simulations

$\log|\phi|^2$



$tv_{th,i}/a$

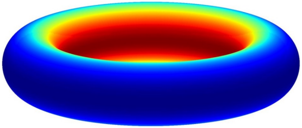
$Q/Q_{\mathbf{GB}}$



$tv_{th,i}/a$

Figure 3: Linear simulations for CBC with kinetic electrons; $N_y = N_x = 30$, $\rho_* = 0.025$

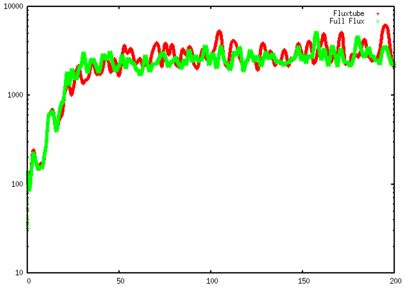
Simulation results: CBC, kinetic electrons



- Add in kinetic electrons

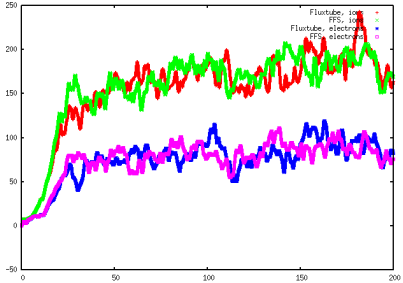
Non-linear simulations

$\log|\phi|^2$



$tv_{th,i}/a$

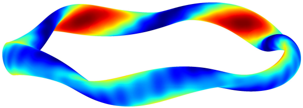
$Q/Q_{\mathbf{GB}}$



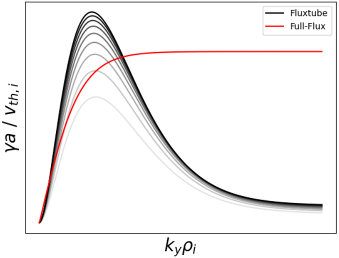
$tv_{th,i}/a$

Figure 3: Non-linear simulations for CBC with kinetic electrons; $N_y = 30, N_x = 150$, $\rho_* = 0.025$

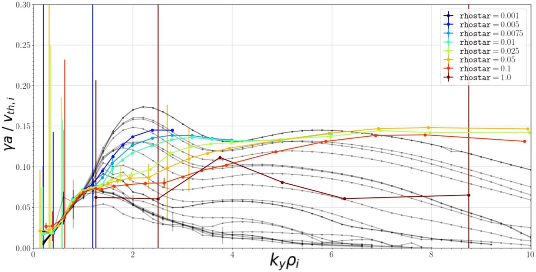
Expectations and results



- Anticipate that including higher k_α leads to a global growth rate
- Growth rate should be some average of the most unstable mode across all field lines

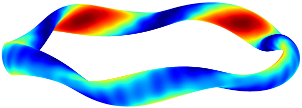


Expectation



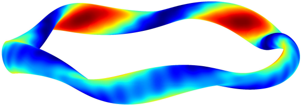
Simulation results

Simulation results: W7-X



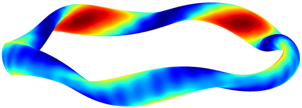
- Impose that each field line in FFS has the same geometry to benchmark algorithm

Simulation results: W7-X



- Impose that each field line in FFS has the same geometry to benchmark algorithm

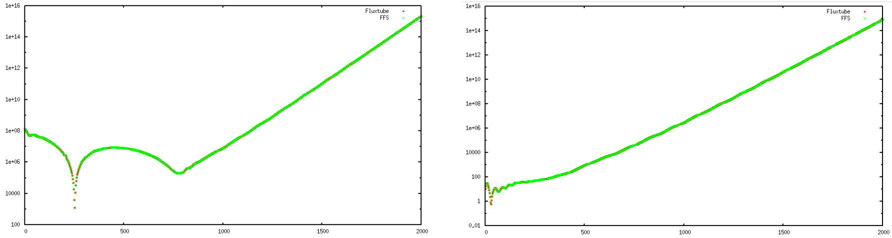
Simulation results: W7-X



- Impose that each field line in FFS has the same geometry to benchmark algorithm
- Modified adiabatic electrons

<u>Linear simulations</u>

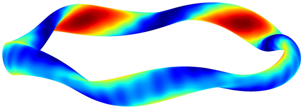
$\log|\phi|^2$ $\qquad\qquad$ $Q/Q_{\mathbf{GB}}$



$tv_{th,i}/a$ $\qquad\qquad$ $tv_{th,i}/a$

Figure 4: Linear simulations for W7-X geometry with modified adiabatic electrons testing algorithm; $N_y = N_x = 72$, $\rho_* = 0.01$

Simulation results: W7-X



- Impose that each field line in FFS has the same geometry to benchmark algorithm
- Modified adiabatic electrons

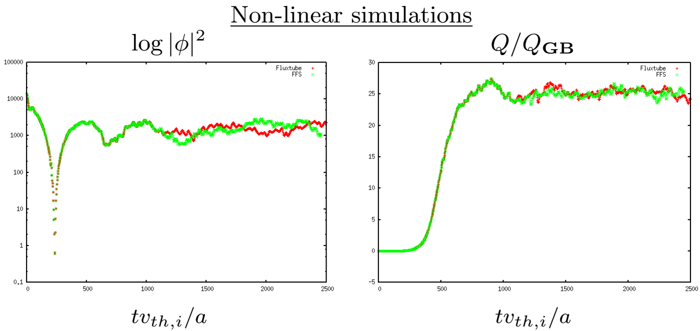
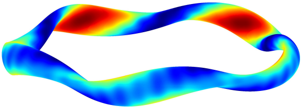


Figure 4: Non-linear simulations for W7-X geometry with modified adiabatic electrons; $N_y = N_x = 30$, $\rho_* = 0.025$

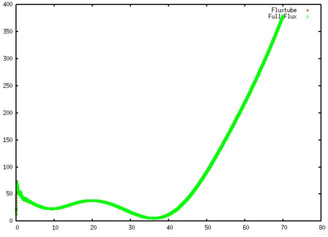
Simulation results: W7-X



- Impose that each field line in FFS has the same geometry to benchmark algorithm
- Add in kinetic electrons

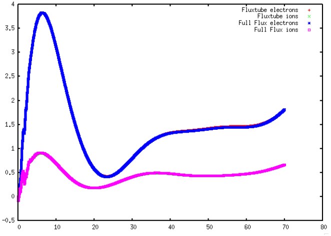
<u>Linear simulations</u>

$\log|\phi|^2$



$tv_{th,i}/a$

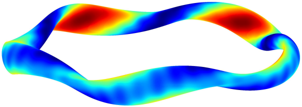
$Q/Q_{\mathbf{GB}}$



$tv_{th,i}/a$

Figure 4: Linear simulations for W7-X geometry with kinetic electrons; $N_y = N_x = 30$, $\rho_* = 0.025$

Simulation results: W7-X



- Impose that each field line in FFS has the same geometry to benchmark algorithm
- Add in kinetic electrons

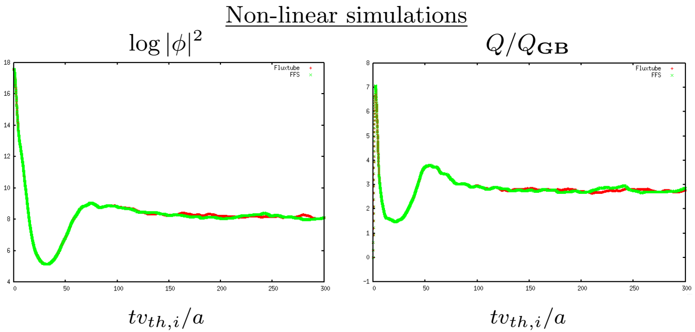
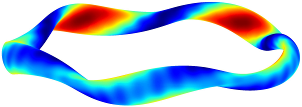


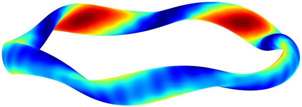
Figure 4: Non-linear simulations for W7-X geometry with kinetic electrons; $N_y = N_x = 64$, $\rho_* = 0.05333$

Simulation results: W7-X geometric variation



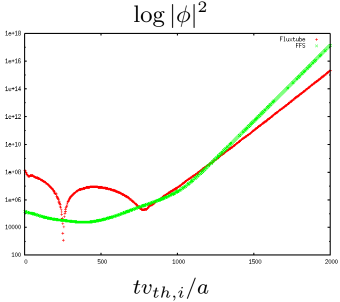
- How does geometric variation modify simulation results?

Simulation results: W7-X geometric variation



- How does geometric variation modify simulation results?

<u>Linear simulations</u>



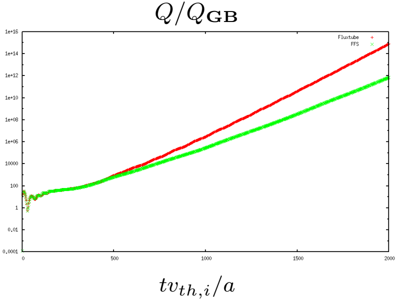
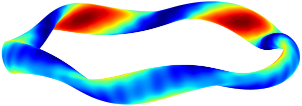


Figure 5: Linear simulations for W7-X geometry with modified adiabatic electrons including full flux effects; $N_y = N_x = 72$, $\rho_* = 0.01$

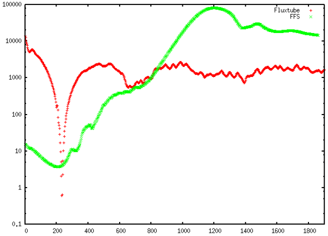
Simulation results: W7-X geometric variation



- How does geometric variation modify simulation results?

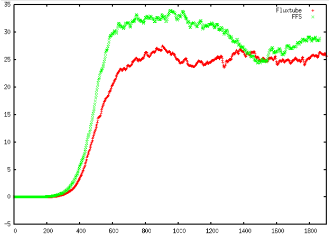
Non-linear simulations

$\log|\phi|^2$



$tv_{th,i}/a$

$Q/Q_{\mathbf{GB}}$



$tv_{th,i}/a$

Figure 5: Non-linear simulations for W7-X geometry with modified adiabatic electrons $N_y = N_x = 72$, $\rho_* = 0.01$

Simulation results: code efficiency

Implicit vs. explicit

- A time-step of `1E-006` is needed to run explicit kinetic electron simulations for W7-X $\rightarrow$ still numerically unstable
- Implicit treatment of parallel streaming allows for time-step of `5E-002`

Simulation results: code efficiency

Implicit vs. explicit

- A time-step of `1E-006` is needed to run explicit kinetic electron simulations for W7-X → still numerically unstable
- Implicit treatment of parallel streaming allows for time-step of `5E-002`

Flux tube vs. full flux annulus

- Currently full-flux code takes $\sim \times 4/5$ longer to run compared with flux tube simulations

- Non-linear simulations with adiabatic electrons:

Flux tube	Full flux algorithm	Full flux with geometric variation
218.03 min	562.15 min	794.66 min
$\times 1$	$\times 2.6$	$\times 3.7$

- Like-for-like resolutions: 12 nodes, 576 cores, 2000 normalised times steps
- Anticipate with optimisation this can be reduced to $\sim \times 3$

Summary and future work

Summary

- There is a need in the community to accurately simulate turbulence on an entire flux surface for non-axisymmetric devices
- We have developed an algorithm to deal with full-flux effects in arbitrary geometry
- We are getting some promising results
- There is still work to be done
- The code is currently being checked, and optimised

Summary and future work

Summary

- There is a need in the community to accurately simulate turbulence on an entire flux surface for non-axisymmetric devices
- We have developed an algorithm to deal with full-flux effects in arbitrary geometry
- We are getting some promising results
- There is still work to be done
- The code is currently being checked, and optimised

Future Work

- Finish off FFS code, and benchmark with other GK codes
- Investigate if/how zonal flows are supported in stellarators

Backup slides: Derivation of twist-and-shift boundary conditions

*

$$A(t,x,y,z) = \sum_k \hat{A}_{k_x,k_y}(t,z) e^{ik_y(y-y_0)+ik_x(x-x_0)} \tag{5}$$

Set $y_0 = 0$, $x_0 = 0$ $A(t,x,y(x,\theta,z),z) = A(t,x,y'(\theta,z+2p\pi),z+2p\pi)$
But $y = y(\theta,z)$

$$\sum_k \hat{A}_{k_x,k_y}(t,z) e^{ik_y y+ik_x x} = \sum_k \hat{A}_{k_x,k_y}(t,z') e^{ik_y(y'(\theta,z'))+ik_x x} \tag{6}$$

So $y'(\theta,z') = y + \frac{\partial y}{\partial z} 2\pi p = y + 2\pi p \frac{\partial y}{\partial \alpha}\frac{\partial \alpha}{\partial z} = y - 2\pi p \iota(\psi)\frac{\partial y}{\partial \alpha}$ Remembering $\iota(\psi) = \iota(\psi_0) + \iota' \frac{\partial \psi}{\partial x}$ so $y' = y - 2\pi p \iota(\psi_0)\frac{\partial y}{\partial \alpha} - 2(x-x_0)\pi p \iota' \frac{\partial y}{\partial \alpha}\frac{\partial \psi}{\partial x}$

$$\sum_k \hat{A}_{k_x,k_y}(t,z) e^{ik_y y+ik_x x} = \sum_k \hat{A}_{k_x,k_y}(t,z+2\pi p) e^{ik_y y+i(k_x-2\pi p\iota' \frac{\partial y}{\partial \alpha}\frac{\partial \psi}{\partial x}k_y)x' - i2\pi p k_y \iota p \frac{\partial y}{\partial \alpha}} \tag{7}$$

Let $\delta k_x = 2\pi p \iota' \frac{\partial y}{\partial \alpha}\frac{\partial \psi}{\partial x} k_y$ and $\Delta = -2\pi p k_y \iota \frac{\partial y}{\partial \alpha}$

$$\sum_k \hat{A}_{k_x,k_y}(t,z) e^{ik_y y+ik_x x} = \sum_k \hat{A}_{k_x,k_y}(t,z+2\pi p) e^{ik_y y+i(k_x-\delta k_x)x'} e^{i\Delta} \tag{8}$$

So relate $k_x = k_x' - 2\pi p \iota' \frac{\partial y}{\partial \alpha}\frac{\partial \psi}{\partial x} k_y$

Backup slides: Response matrix

*

$$\phi^{n+1} = \left[\sum_\nu \frac{Z_\nu^2 n_\nu}{T_\nu}(1-\Gamma_{0,\nu})\right]^{-1} \sum_\nu Z_\nu n_\nu \frac{2B}{\pi^{1/2}} \int_{-\infty}^{\infty} dv_\parallel \int_0^\infty d_\mu J_{0,\nu} g_\nu^{n+1} \tag{9}$$

Let $g^{n+1} = g^{n+1}_{\text{hom}} + g^{n+1}_{\text{inhom}}$

$$\frac{g^{n+1}_{\text{inhom}} - g^n}{\Delta t} = -v_\parallel \hat{\boldsymbol{b}} \cdot \boldsymbol{\nabla} z \left(\frac{\partial g^{n+1}_{\text{inhom}}}{\partial z} + \frac{Z_\nu}{T_\nu} \frac{\partial J_0 \phi^n}{\partial z} F_{0,\nu} \right) \tag{10}$$

$$\frac{g^{n+1}_{\text{hom}} - g^n}{\Delta t} = -v_\parallel \hat{\boldsymbol{b}} \cdot \boldsymbol{\nabla} z \left(\frac{\partial g^{n+1}_{\text{hom}}}{\partial z} + \frac{Z_\nu}{T_\nu} \frac{\partial J_0 \phi^{n+1}}{\partial z} F_{0,\nu} \right) \tag{11}$$

Then

$$g^{n+1} = \sum \frac{\delta g_{\text{hom}}}{\delta \phi} \phi^{n+1} + g^{n+1}_{\text{inhom}} \tag{12}$$

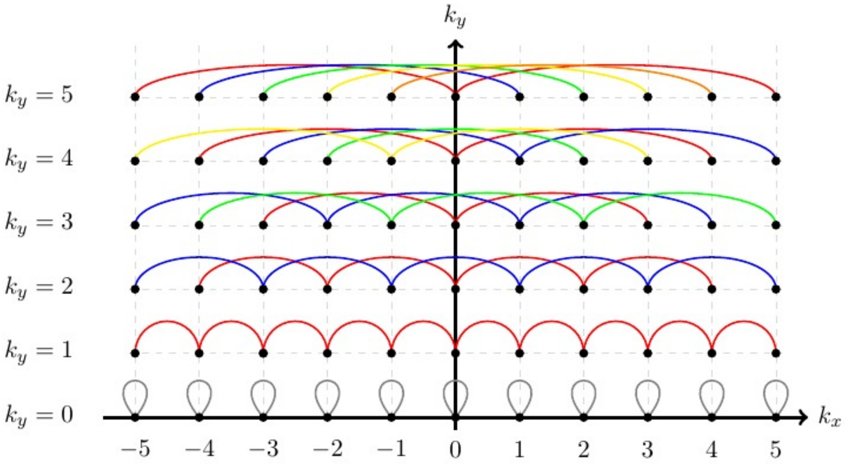
Substitute into quasineutrality equation and solve for ϕ^{n+1}

$$\left[I - Q \sum \frac{\delta g_{\text{hom}}}{\delta \phi} \right] \phi^{n+1} = \phi^{n+1}_{\text{inhom}} \tag{13}$$

With I the identity, $Q = \sum_\nu Z_\nu n_\nu \frac{2B}{\pi^{1/2}} \int_{-\infty}^{\infty} dv_\parallel \int_0^\infty d_\mu J_{0,\nu}$, and $\phi^{n+1}_{\text{inhom}} = Q g^{n+1}_{\text{inhom}}$

Backup slides: Eigenmode chains

*



Backup slides: Bessel functions

*

Explicitly expanding the gyroaveraged electrostatic potential in Fourier harmonics:

$$\varphi_\nu = \sum_{\mathbf{k}''} e^{i\mathbf{k}''\cdot\mathbf{R}} J_0(a_{\mathbf{k}'',\nu})\hat{\phi}_{\mathbf{k}''}, \tag{14}$$

with

$$a_{\mathbf{k}'',\nu} = \frac{ck''_\perp(\alpha,z)}{Z_\nu e}\sqrt{\frac{2m_\nu\mu}{B(\alpha,z)}}. \tag{15}$$

Both k_α and α appear. For axisymmetric systems, the α dependence is absent and so gyro-averaging is a local operation in k_α-space. Now there is coupling between modes with different k_α. Expanding the Bessel function:

$$\hat{\varphi}_{\mathbf{k},\nu} = \int d^2\mathbf{R} \sum_{\mathbf{k}'',k'_\alpha} e^{i\left(k''_\psi - k_\psi\right)\psi} e^{i\left(k'_\alpha + k''_\alpha - k_\alpha\right)\alpha} \hat{J}_{\mathbf{k}'',k'_\alpha,\nu}\hat{\phi}_{\mathbf{k}''}, \tag{16}$$

where we have used

$$J_0(a_{\mathbf{k}'',\nu}) = \sum_{k'_\alpha} \hat{J}_{\mathbf{k}'',k'_\alpha,\nu}(z,\mu) e^{ik'_\alpha\alpha}. \tag{17}$$

Making use of the orthogonality of the Fourier harmonics:

$$\hat{\varphi}_{\mathbf{k},\nu} = \sum_{k'_\alpha} \hat{J}_{(k_\psi,k_\alpha-k'_\alpha),k'_\alpha,\nu}\hat{\phi}_{(k_\psi,k_\alpha-k'_\alpha)}. \tag{18}$$

Backup slides: $\bar{\phi}$ equation

*

- If we let $Q = J_0(k_\perp)B(\alpha)$ and Fourier decompose we get:

$$Q = \sum_{k'_\alpha} \hat{Q}_{k_\alpha, k'_\alpha} e^{ik'_\alpha y} \tag{19}$$

- Define $\bar{Q}$ to be the $k'_\alpha = 0$ component of this
- Take a similar approach for $\Delta(k_\perp) \doteq \sum_\nu \frac{Z_\nu^2 n_\nu}{T_\nu}(1 - \Gamma_{0,\mathbf{k}})$

$$\Delta = \sum_{k'_\alpha} \hat{\Delta}_{k_\alpha, k'_\alpha} e^{ik'_\alpha y} \tag{20}$$

- $\bar{\Delta}$ being the $k'_\alpha = 0$ component
- Putting everything together we get the equation for $\bar{\phi}$

$$\bar{\Delta}_{\mathbf{k}} \bar{\phi}_{\mathbf{k}} = \sum_\nu Z_\nu n_\nu \int \mathrm{d}v_\parallel \int \mathrm{d}\mu \bar{Q}_{\mathbf{k}} g_{\mathbf{k}} \tag{21}$$