New algorithm for background toroidal flow shear in the local gyrokinetic code GS2

Nicolas Christen\(^1\), Michael Barnes\(^1\), Felix Parra\(^1\)

\(^1\)Rudolf Peierls Centre for Theoretical Physics, University of Oxford, Oxford OX1 3PU, UK

September 21, 2018
1. Motivation
2. Overview of GS2
3. Linear algorithm
4. Nonlinear term
5. Nonlinear simulations: old vs new code
6. Conclusions
Motivation
Motivation

▶ Shear in the background toroidal rotation can substantially affect turbulent transport [1][2] → use Neutral Beam Injection

▶ Want to include experimentally relevant levels of NBI induced toroidal background flows and flow shear in local GK simulations

▶ Current algorithm [3] gives unphysical behaviour with low flow shear:


Does this affect turbulent fluxes ?
⇒ new algorithm treating shear continuously over time
Overview of GS2
Overview of GS2 – Coordinates

\[ \theta - \text{poloidal angle} \]
\[ \psi - \text{poloidal mag. flux} \]
\[ \alpha = \zeta - q(\psi)\theta \]

with \( B = \nabla \alpha \times \nabla \psi \) and \((\zeta, \alpha)\) are straight field line coordinates.

GS2 coordinate system

\( \theta \) as the coordinate along \( B \), \( x \) and \( y \) in the perpendicular plane:

\[ x = \frac{1}{r_r B_r} \frac{q_0}{r_{\psi,0}} (\psi - \psi_0) \]

\[ y = \frac{1}{r_r B_r} \frac{\partial \psi}{\partial r_\psi} \bigg|_{r_{\psi,0}} (\alpha - \alpha_0) \]
Overview of GS2 – Orderings & equations

\[ f = \langle f \rangle + \delta f \]

\[ \frac{\delta f}{\langle f \rangle} \sim \frac{\rho_i}{a} \equiv \rho_* \ll 1 \]

Microscopic:

\[ \frac{\omega}{\Omega_i} \sim \rho_i \hat{b} \cdot \nabla \sim \mathcal{O}(\rho_*) \quad \rho_i \nabla_\perp \sim \mathcal{O}(1) \]

Macroscopic:

\[ \frac{\tau_E^{-1}}{\Omega_i} \sim \mathcal{O}(\rho_*^3) \quad \rho_i \nabla \sim \mathcal{O}(\rho_*) \]

\[ \frac{v_E}{v_{thr}} \sim \mathcal{O}(\rho_*) \quad \rho_* \ll \text{Mach} \ll 1 \]

Overview of GS2 – Orderings & equations

GK & quasineutrality (QN) equations

\( \varphi \) – electrostatic potential fluctuation

\( g_s \equiv \langle \delta f_s \rangle_R \) – avg over gyrophase

\[
\frac{\partial g_s}{\partial t} + \left( u + w_\parallel \hat{b} + V_{Ds} + \langle V_E \rangle_R \right) \cdot \nabla g_s - \langle C_L [h_s] \rangle_R =
\]

\[
- \frac{Z_s e F_{0s}}{T_s} \left[ w_\parallel \hat{b} + V_{Ds} \right] \cdot \nabla \langle \varphi \rangle_R - \left\{ \nabla F_{0s} + \frac{m_s F_{0s}}{T_s} \frac{I w_\parallel}{B} \nabla \Omega_\varphi \right\} \cdot \langle V_E \rangle_R
\]

\[
\sum_s Z_s \int d^3 v \langle g_s \rangle_r = \sum_s \frac{Z_s^2 e}{T_s} \left( n_s \varphi - \int d^3 v \langle \varphi \rangle_R F_{0s} \right)
\]
The background plasma flow can be written as [5]:

\[
\mathbf{u} = \frac{c}{B} \mathbf{\hat{b}} \times \nabla \varphi_{tot} + u_\parallel \mathbf{\hat{b}} = \Omega_\psi(\psi) R^2 \nabla \phi + O(\rho \ast v_{th})
\]

Local approximation:

\[
\Omega \simeq \Omega_{\psi,0} + \gamma_E \left[ \frac{\partial r_{\psi,N}}{\partial \psi_N} \right]_{\psi_0}
\]

where we defined

\[
\gamma_E = \left. \frac{r_{\psi,0}}{q_0} \frac{\partial \Omega_\phi}{\partial r_\psi} \right|_{\psi_0}
\]

Overview of GS2 – Shearing frame

Need to handle flow shear in the GK eq.:

\[
\frac{\partial g_s}{\partial t} + \gamma_E x \frac{\partial g_s}{\partial y} = S[g_s]
\]

To work in Fourier space, need to get rid of the non-periodic term.

⇒ GS2 works in the shearing frame \((x, y', \theta)\) [5] where:

\[y' \equiv y - \gamma_{Ext}\]

Overview of GS2 – Fourier space

In $(x, y', \theta)$, any fluctuating quantity $\Phi$ can be expressed as a Fourier series in the $\perp$-plane:

$$\Phi(t, x, y', \theta) = \sum_{k_x, k_y} \hat{\Phi}_{k_x, k_y}(\theta) e^{ik_x x + ik_y y'}$$

- Can re-write exponent as: $ik_x x + ik_y y' = i (k_x - \gamma_E t k_y) x + k_y y$  
  \[ \equiv k^*_x(t) \]

- New time dependences in GK-QN eqs when $\gamma_E \neq 0$:

$$\left. \frac{\partial \Phi}{\partial x} \right|_y = \sum_{k_x, k_y} i(k_x - \gamma_E t k_y) \hat{\Phi}_{k_x, k_y} e^{ik_x x + ik_y y'}$$

$$\langle \Phi \rangle_R = \sum_{k_x, k_y} J_0(k^*_\perp \rho) \hat{\Phi}_{k_x, k_y} e^{ik_x x + ik_y y'}$$  
  \[ \equiv J^*_0(t) \]
Overview of GS2 – Discretised equations

Equations to solve in GS2 (electrostatic)

Gyrokinetic equation:

\[ A^{\ast} \hat{g}[it + 1] + B^{\ast} \hat{g}[it] = C^{\ast} \hat{\varphi}[it + 1] + D^{\ast} \hat{\varphi}[it] + \text{NL}(\hat{g}[it], \hat{\varphi}[it]) \]

Quasineutrality:

\[ U^{\ast} \hat{\varphi}[it + 1] = V^{\ast} \hat{g}[it + 1] \]

In an explicit code (ie with all terms evaluated at \( it \), except for \( \partial / \partial t \)):

- evaluate \( k_x^{\ast}(t) \) and \( J_0^{\ast}(t) \) at every time step
- reasonable computational cost
- ... but GS2 is implicit linearly
Linear algorithm
Linear algorithm – Discretised equations

Gyrokinetic equation (electrostatic, linear):

\[ A^* g[it + 1] + B^* g[it] = C^* \phi[it + 1] + D^* \phi[it] + \text{NL}(g[it], \phi[it]) \]

Quasineutrality:

\[ U^* \phi[it + 1] = V^* g[it + 1] \]  \hspace{1cm} (1)

- **Implicit algorithm** [6] (e.g. \( \phi \) terms evaluated at \( it + 1 \) in GK)
  - Better stability than fully explicit schemes
  - **But need to compute** \( \phi[it + 1] \) **to get** \( g[it + 1] \)
  - **Use a Green’s function approach** [7]

---


Linear algorithm – Green’s function method

1. Split \( g[it + 1] \) into two parts:

   \[
   g[it + 1] = g_p + g_c
   \]

   **Predictor:** known from \( t[it] \)

2. Define response matrix by re-writing \( g_c \) as:

   \[
   g_c = \left( \frac{\delta g}{\delta \varphi} \right)^* \cdot (\varphi[it + 1] - \varphi[it])
   \]

   if \( \gamma_E = 0 \) can compute at init.

3. Plug into QN eq. to get:

   \[
   \varphi[it + 1] = \varphi[it] + (M^*)^{-1}[it + 1] \cdot (V*[it + 1]g_p - U*[it + 1]\varphi[it])
   \]

   with \( M^* = U*1 + V*(\frac{\delta g}{\delta \varphi})^* \)
Linear algorithm – Problems to solve

When \( \gamma_E \neq 0 \), two issues have to be addressed:

- In the lab frame, eddies get constantly sheared over time
  \( \Rightarrow \) in the sim. there would be no structures elongated in \( x \).

- \( (\frac{\delta g}{\delta \varphi})^* \) is time dependent and would be computationally very expensive to recompute at every time step.
**Linear algorithm – Old implementation**

\[ E \times B \text{ remapping (aka “wavenumber shift”) } \]

\[ \forall k_y \neq 0, \exists T \text{ such that: } \]

\[ k_x^*(T, k_y) = k_x - \gamma_E T k_y = k_x \pm \Delta k_x \]

After \( T \), update the set of modes with this \( k_y \) (e.g. \( k_y, \gamma_E > 0 \)):

\[ \{ \hat{\Phi} - K_x, \hat{\Phi} - K_x + \Delta k_x, \ldots, \hat{\Phi} K_x \} \rightarrow \{ \hat{\Phi} - K_x + \Delta k_x, \ldots, \hat{\Phi} K_x, \hat{\Phi} K_x + \Delta k_x \} \]

**Nearest neighbour approximation**

\[ k_x^*(t) \rightarrow \text{nearest neighbour on fixed } k_x \text{ grid } (\bar{k}_x). \]

e.g. at \( t = T \):

\[ \frac{\partial \hat{\Phi}}{\partial x} \bigg|_y = \sum_{-K_x}^{K_x} i k_x^*(T) \hat{\Phi}_k x e^{i k_x x + i k_y y'} \rightarrow \sum_{-K_x + \Delta k_x}^{K_x + \Delta k_x} i \bar{k}_x \hat{\Phi}_k x e^{i k_x x + i k_y y'} \quad (2) \]
**Linear algorithm – Old implementation**

\(E \times B\) remap + nearest neighbour approx.:

- ✓ always includes structures elongated in \(x\) (ie \(\bar{k}_x\) stays the same)
- ✓ no time dependences from flow shear in the code
- ✓ only modification in the code when \(\gamma_E \neq 0\):

\[
\begin{align*}
\text{at } t = T: & \\
\phi[i_k x, i_k y] &= \phi[i_k x \pm 1, i_k y] \\
g[i_k x, i_k y] &= g[i_k x \pm 1, i_k y]
\end{align*}
\]

For small \(\gamma_E\) or \(k_y\), flow shear has limited effect for a long time, followed by a discrete jump when the remapping occurs.
Linear algorithm – Old implementation

Test: reproduce linear Floquet mode behaviour

![Graph showing linear Floquet mode behaviour with different times]

| $|\phi|^2$ | $\theta - \theta_0$ |
|---|---|
| 1.5E-03 | −40 |
| 1.0E-03 | −20 |
| 5.0E-04 | 0 |
| 0.0E+00 | 20 |

- $t = 27.6$
- $t = 52.8$
- $t = 77.9$
Sum along a single ballooning mode

N. Christen
Simulating flow shear in GS2
September 21, 2018 20 / 33
Linear algorithm – New implementation

- Keep $E \times B$ remapping

- In GK, treat new time dependences explicitly in time:

$$L[k_x^*, J_0^*] = L[\bar{k}_x, \bar{J}_0] + (L[k_x^*, J_0^*] - L[\bar{k}_x, \bar{J}_0])$$

small $\Rightarrow$ treat explicitly

- Response matrix $\frac{\delta g}{\delta \varphi}$ becomes time independent
- $M^*$-matrix still contains time dependence from QN

- Pre-compute 3 $M$-matrices, then interpolate every time step:

$$\begin{align*}
M^{-1}_L &\equiv M^{-1}(\bar{k}_x - \Delta k_x) \\
M^{-1} &\equiv M^{-1}(\bar{k}_x) \\
M^{-1}_R &\equiv M^{-1}(\bar{k}_x + \Delta k_x)
\end{align*}$$

$$M^{-1}(k_x^*(t)) \approx C_L \bar{M}_L^{-1} + C \bar{M}^{-1} + C_R \bar{M}_R^{-1}$$
Linear algorithm – New implementation

Old

Sum along a single ballooning mode

New

Sum along a single ballooning mode
Nonlinear term
Nonlinear term – In GS2

Advection of fluctuations by turbulent $E \times B$ drift:

$$\langle v_{E \times B} \rangle \mathbf{R} \cdot \nabla g \sim \left[ \frac{\partial \langle \phi \rangle \mathbf{R}}{\partial x} \bigg| \frac{\partial g}{\partial y} - \frac{\partial \langle \phi \rangle \mathbf{R}}{\partial y} \frac{\partial g}{\partial x} \bigg|_y \right]$$

To compute it efficiently in Fourier space:

1. Easy to compute $\mathcal{F} \left[ \frac{\partial \langle \phi \rangle \mathbf{R}}{\partial x} \right], \mathcal{F} \left[ \frac{\partial g}{\partial y} \right], ...$
2. Inverse-transform each of them to direct space
3. Compute NL in direct space
4. Transform result back to Fourier space
Nonlinear term – Old implementation

Poisson bracket is invariant under \((x, y) \rightarrow (x, y')\):

\[
\left. \frac{\partial \langle \varphi \rangle_R}{\partial x} \frac{\partial g}{\partial y} - \frac{\partial \langle \varphi \rangle_R}{\partial y} \frac{\partial g}{\partial x} \right|_y = \left. \frac{\partial \langle \varphi \rangle_R}{\partial x} \frac{\partial g}{\partial y'} - \frac{\partial \langle \varphi \rangle_R}{\partial y'} \frac{\partial g}{\partial x} \right|_{y'}
\]

(1)

▶ Use (1) with \(\Phi = \sum_k \hat{\Phi}_k e^{ik_xx+ik_yy'}\)

▶ Apply nearest neighbour approx:

\[
\Rightarrow (1) \sim \sum_k ik_x \bar{J}_0 \hat{\varphi}_k e^{ik_xx+ik_yy'} \cdot \sum_k ik_y \hat{g}_k e^{ik_xx+ik_yy'} - (\hat{g} \leftrightarrow \bar{J}_0 \hat{\varphi})
\]

▶ No flow shear time dependence ...

▶ ... but does not take into account \(E \times B\) remapping
Nonlinear term – New implementation

- Take remapping into account, e.g. for $\frac{\partial \Phi}{\partial x} |_{y'}$:

$$\left. \frac{\partial \Phi}{\partial x} \right|_{y'} = \sum_{-K_y}^{K_y} \sum_{-K_x \pm N \Delta k_x}^{K_x \pm N \Delta k_x} i k_x \hat{\Phi}_{k_x, k_y} e^{i k_x x + i k_y y'}$$

- $N = N(t, k_y)$: # of $E \times B$ remaps for modes with $k_y$

- Change sum index:

$$\left. \frac{\partial \Phi}{\partial x} \right|_{y'} = \sum_{-K_y}^{K_y} \sum_{-K_x}^{K_x} i (k_x \pm N \Delta k_x) \hat{\Phi}_{k_x \pm N \Delta k_x, k_y} e^{i k_x x + i k_y y'} e^{\pm i N \Delta k_x x}$$

- Also replace $\bar{J}_0 \rightarrow J_0^*$

Alternative method developped for GENE by B. McMillan, J. Ball et al:
McMillan B F '17 arXiv 1711.03830v1
Nonlinear term

Test: start with 2D Gaussian in \((x, y)\) and solve

\[
\frac{\partial \phi}{\partial t} + \gamma E x \frac{\partial \phi}{\partial y} = 0 \iff \frac{\partial \phi}{\partial t}\bigg|_{y'} = 0
\]

- In Fourier space: constant 2D Gaussian in \((k_x, k_y)\)
- Applying \(E \times B\) remapping
- At each timestep, inverse-transform to direct space:

\[\phi\text{ at } t = 0, 1.65, 3.35, 5\]
Nonlinear simulations: old vs new code
**Nonlinear simulations: old vs new code**

**JET ITB:** $|\gamma_E| = 0.15$, $\frac{a}{L_{T_i}} = 8.9$, $\frac{a}{L_{T_e}} = 2.1$,

$N_{k_x} = 128$, $\Delta k_x = 0.05$, $N_{k_y} = 22$, $\Delta k_y = 0.06$, $C^{6+}$, no coll.

$(k_x = -0.88, k_y = 0.06)$ from $t = 0.00$

$\sum_{k_x} \langle |\hat{\phi}_k|^2 \rangle_\theta$ for $k_y = 0.06$

$\langle |\hat{\phi}_k|^2 \rangle_\theta$ for $k_y = 0.06$

### Graphs

- **Left Graph:**
  - $t(a/v_t)$ range: 0 to 100
  - $\langle |\hat{\phi}_k|^2 \rangle_\theta$ vs $t(a/v_t)$
  - Colors: red (old), blue (new)
  - Markers: solid

- **Right Graph:**
  - $t(a/v_t)$ range: 0 to 100
  - $\sum_{k_x} \langle |\hat{\phi}_k|^2 \rangle_\theta$ for $k_y = 0.06$
  - Colors: red (old), blue (new)
  - Markers: solid
Nonlinear simulations: **old vs new code**

**Old**

\[ \langle |\hat{\phi}_k|^2 \rangle_{t,\theta} \quad \forall \ k_y > 0 \]

**New**

\[ \langle |\hat{\phi}_k|^2 \rangle_{t,\theta} \quad \forall \ k_y > 0 \]
Old algorithm does surprisingly well in most cases ...

... but for some cases, fluxes change significantly

Computational cost $\sim +25\%$ compared to old code

Still to do: adapt collisions to the new algorithm
Conclusions
Conclusions

- Implemented a new algorithm treating flow shear continuously in $t$
- Ran tests for linear and NL terms
- No discrete jumps in the new code
- Agreement between old and new codes in most cases
- Level of turbulent fluxes changes significantly in some cases

Acknowledgements: I would like to thank Ollie Beeke for providing the input parameters of the JET ITB shot, and Michael Hardman for helpful discussions.
Nonlinear term – Old implementation (1)

Poisson bracket is invariant under \((x, y) \rightarrow (x, y')\):

\[
\frac{\partial \langle \varphi \rangle_R}{\partial x} \bigg|_{y} \frac{\partial g}{\partial y} - \frac{\partial \langle \varphi \rangle_R}{\partial y} \frac{\partial g}{\partial x} \bigg|_{y} = \frac{\partial \langle \varphi \rangle_R}{\partial x} \bigg|_{y'} \frac{\partial g}{\partial y'} - \frac{\partial \langle \varphi \rangle_R}{\partial y'} \frac{\partial g}{\partial x} \bigg|_{y'}
\]

(1)

- Use (1) with \(\Phi = \sum \hat{\Phi}_k e^{ik_x x + ik_y y'}\)
- Apply nearest neighbour approx:

\[
\Rightarrow (1) \sim \sum_{k} ik_x \bar{J}_0 \hat{\varphi}_k e^{ik_x x + ik_y y'} \cdot \sum_{k} ik_y \hat{g}_k e^{ik_x x + ik_y y'} - (\hat{g} \leftrightarrow \bar{J}_0 \hat{\varphi})
\]

- No flow shear time dependence ...
- ... but does not take into account \(E \times B\) remapping
Nonlinear term – Old implementation (2)

Poisson bracket is invariant under \((x, y) \rightarrow (x, y')\):

\[
\left. \frac{\partial \langle \varphi \rangle_R}{\partial x} \right|_y \left. \frac{\partial g}{\partial y} \right|_y - \left. \frac{\partial \langle \varphi \rangle_R}{\partial y} \right|_y \left. \frac{\partial g}{\partial x} \right|_y = \left. \frac{\partial \langle \varphi \rangle_R}{\partial x} \right|_{y'} \left. \frac{\partial g}{\partial y'} \right|_{y'} - \left. \frac{\partial \langle \varphi \rangle_R}{\partial y'} \right|_{y'} \left. \frac{\partial g}{\partial x} \right|_{y'}
\]

(2)

- Use (2) with \(\Phi = \sum_k \hat{\Phi}_k e^{ik_x x + ik_y y}\)

- Apply nearest neighbour approx:

\[
\Rightarrow (2) \sim \sum_k i \bar{k}_x \tilde{J}_0 \hat{\varphi}_k e^{i \bar{k}_x x + ik_y y} \cdot \sum_k i k_y \hat{g}_k e^{i \bar{k}_x x + ik_y y} - (\hat{g} \leftrightarrow \tilde{J}_0 \hat{\varphi})
\]

- Need to replace with exact \(k_x^*(t)\). Correction vanishes with better resolution:

\[
|k_x^* - \bar{k}_x| \leq \Delta k_x / 2
\]
Nonlinear term

Test #2: start with two 2D Gaussians in \((x, y)\) and solve in Fourier space

\[
\begin{aligned}
\frac{\partial \phi_1}{\partial t} + \gamma E x \frac{\partial \phi_1}{\partial y} + \frac{\partial \phi_1}{\partial x} \frac{\partial \phi_2}{\partial y} - \frac{\partial \phi_1}{\partial y} \frac{\partial \phi_2}{\partial x} &= 0 \\
\frac{\partial \phi_2}{\partial t} + \gamma E x \frac{\partial \phi_2}{\partial y} &= 0
\end{aligned}
\]