GKV manual

1 Table of Contents

2 Formulation

* Blue-colored sentences are physical assumptions used in GKV [2-1]. This manual is based on the GKV version gkvp_f0.48.

2.1 Governing equations

One derives gyrokinetic equations based on the following gyrokinetic ordering [2-2],

$$\frac{\tilde{f}}{F} \sim \frac{e\tilde{\phi}}{T} \sim \frac{\tilde{B}}{B} \sim \frac{k_\parallel}{k_\perp} \sim \frac{\omega}{\Omega} \equiv \delta \ll 1. \tag{2.1}$$

GKV follows δf gyrokinetics, where distribution functions are split into equilibrium and perturbed parts $\mathcal{F} = F + \tilde{f}$. Additionally, there are some subsidiary assumptions:

- separation of the equilibrium and perturbed scale lengths $|\nabla F|/F \ll |\nabla \tilde{f}|/f$ || decouples neoclassical physics from turbulent dynamics and treats flute-type perturbations
- low β value || justifies neglect of compressional magnetosonic waves $\tilde{B}_\parallel$ and higher-order correction in β, but retains shear Alfvénic dynamics $\tilde{A}_\parallel$
- low equilibrium flows $v_{\mathrm{eq.}} \ll v_{\mathrm{th}}$ || the present version of GKV cannot treat equilibrium flows
- the equilibrium distribution function is to be a local Maxwellian $F = F_{\mathrm{M}} = n\left(\frac{m}{2\pi T}\right)^{\frac{3}{2}} e^{-\frac{mv_\parallel^2}{2T} - \frac{\mu B}{T}}$
- the equilibrium magnetic field satisfies the MHD equilibrium $\nabla P = \boldsymbol{J} \times \boldsymbol{B}$

Then, the δf gyrokinetic Vlasov-Poisson-Ampère equations are

$$\begin{aligned}
&\frac{\partial \tilde{f}_{\mathrm{s}}}{\partial t} + \left(v_\parallel \frac{\boldsymbol{B} + \tilde{\boldsymbol{B}}_\perp}{B} + \tilde{\boldsymbol{v}}_{\mathrm{E}} + \boldsymbol{v}_{\mathrm{sG}} + \boldsymbol{v}_{\mathrm{sC}} \right) \cdot \nabla \left(\tilde{f}_{\mathrm{s}} + \frac{e_{\mathrm{s}} F_{\mathrm{sM}}}{T_{\mathrm{s}}} J_{0\mathrm{s}} \tilde{\phi} \right) \\
&- \frac{\mu \nabla_\parallel B}{m_{\mathrm{s}}} \frac{\partial}{\partial v_\parallel} \left(\tilde{f}_{\mathrm{s}} + \frac{e_{\mathrm{s}} F_{\mathrm{sM}}}{T_{\mathrm{s}}} J_{0\mathrm{s}} \tilde{\phi} \right) \\
&+ \frac{e_{\mathrm{s}} F_{\mathrm{sM}}}{T_{\mathrm{s}}} \left[v_\parallel \frac{\partial J_{0\mathrm{s}} \tilde{A}_\parallel}{\partial t} - \boldsymbol{v}_{\mathrm{s}*} \cdot \nabla J_{0\mathrm{s}} (\tilde{\phi} - v_\parallel \tilde{A}_\parallel) \right] = C_{\mathrm{s}}, &(2.2) \\
&\left[\nabla_\perp^2 - \frac{1}{\varepsilon_0} \sum_{\mathrm{s}} \frac{e_{\mathrm{s}}^2 n_{\mathrm{s}}}{T_{\mathrm{s}}} (1 - \Gamma_{0\mathrm{s}}) \right] \tilde{\phi} = -\frac{1}{\varepsilon_0} \sum_{\mathrm{s}} e_{\mathrm{s}} \int dv^3 J_{0\mathrm{s}} \tilde{f}_{\mathrm{s}}, &(2.3) \\
&\nabla_\perp^2 \tilde{A}_\parallel = -\mu_0 \sum_{\mathrm{s}} e_{\mathrm{s}} \int dv^3 J_{0\mathrm{s}} v_\parallel \tilde{f}_{\mathrm{s}}, &(2.4)
\end{aligned}$$

where the gyrophase-average operators $J_{0\mathrm{s}} = \oint (d\xi/2\pi) e^{\boldsymbol{\rho}_{\mathrm{s}} \cdot \nabla} = \oint (d\xi/2\pi) e^{-\boldsymbol{\rho}_{\mathrm{s}} \cdot \nabla}$ and $\Gamma_{0\mathrm{s}} = \int dv^3 (F_{\mathrm{sM}}/n_{\mathrm{s}}) J_{0\mathrm{s}}^2$ are used with the gyroradius vector $\boldsymbol{\rho}_{\mathrm{s}} = \boldsymbol{b} \times m_{\mathrm{s}} \boldsymbol{v}/(e_{\mathrm{s}} B)$. The electric and magnetic fields are $\tilde{\boldsymbol{E}} = -\nabla(J_{0\mathrm{s}} \tilde{\phi}) - \boldsymbol{b} \partial \tilde{A}_\parallel / \partial t$ and $\tilde{\boldsymbol{B}}_\perp = \nabla(J_{0\mathrm{s}} \tilde{A}_\parallel) \times \boldsymbol{b}$. The $\boldsymbol{E} \times \boldsymbol{B}$, grad-B, curvature, diamagnetic drift velocities are respectively given by $\tilde{\boldsymbol{v}}_{\mathrm{E}} = \boldsymbol{b} \times \nabla(J_{0\mathrm{s}} \tilde{\phi})/B$, $\boldsymbol{v}_{\mathrm{sG}} = \boldsymbol{b} \times \mu \nabla B/(e_{\mathrm{s}} B)$, $\boldsymbol{v}_{\mathrm{sC}} = \boldsymbol{b} \times m_{\mathrm{s}} v_\parallel^2 \boldsymbol{b} \cdot \nabla \boldsymbol{b}/(e_{\mathrm{s}} B)$ and $\boldsymbol{v}_{\mathrm{s}*} = \boldsymbol{b} \times [T_{\mathrm{s}} \nabla \ln n_{\mathrm{s}} + (m_{\mathrm{s}} v_\parallel^2/2 + \mu B - 3T_{\mathrm{s}}/2) \nabla \ln T_{\mathrm{s}}]/(e_{\mathrm{s}} B)$. C_s is the linearized collision term on the species s and will be explained in Section 2.5. The nonlinear term in the Vlasov eq. (denoted $\mathcal{N}_{\mathrm{s}}$ below), which originates from $\boldsymbol{E} \times \boldsymbol{B}$ and $v_\parallel \tilde{\boldsymbol{B}}_\perp / B$ advections of $\tilde{f}$ and $\tilde{\boldsymbol{E}} \cdot \tilde{\boldsymbol{B}}_\perp$ acceleration of F, can be rewritten as,

$$\begin{aligned}
\mathcal{N}_{\mathrm{s}} &= \left(\tilde{\boldsymbol{v}}_{\mathrm{E}} + v_\parallel \frac{\tilde{\boldsymbol{B}}_\perp}{B} \right) \cdot \nabla \tilde{f}_{\mathrm{s}} + \frac{e_{\mathrm{s}} \tilde{\boldsymbol{E}}}{m_{\mathrm{s}}} \cdot \frac{\tilde{\boldsymbol{B}}_\perp}{B} \left(-\frac{m_{\mathrm{s}} v_\parallel}{T_{\mathrm{s}}} F_{\mathrm{Ms}} \right) \\
&= \frac{\boldsymbol{b}}{B} \cdot \nabla \left(J_{0\mathrm{s}} \tilde{\phi} - v_\parallel J_{0\mathrm{s}} \tilde{A}_\parallel \right) \times \nabla \left(\tilde{f}_{\mathrm{s}} + \frac{e_{\mathrm{s}} F_{\mathrm{Ms}}}{T_{\mathrm{s}}} J_{0\mathrm{s}} \tilde{\phi} \right), \qquad (2.5)
\end{aligned}$$

respectively.

2.2 Geometry and coordinates

When an equilibrium magnetic field is known, one can construct a flux coordinate $(\rho_f, \theta_f, \varphi_f)$ such that,

$$\boldsymbol{B} = \nabla\Psi_p(\rho_f) \times \nabla[q(\rho_f)\theta_f - \varphi_f], \tag{2.6}$$

where we use the safety factor $q(\rho_f) = d\Psi_t/d\Psi_p$ and the toroidal and poloidal flux $\Psi_p(\rho_f)$ and $\Psi_t(\rho_f)$. GKV employs Clebsch-type coordinate as

$$x = c_x(\rho_f - \rho_{f0}), \tag{2.7}$$
$$y = c_y[q(\rho_f)\theta_f - \varphi_f], \tag{2.8}$$
$$z = \theta_f, \tag{2.9}$$

where ρ_{f0}, c_x and c_y are constant. We refer (x, y, z) as the radial, field-line-label, and field-aligned coordinates, respectively. Using this GKV coordinates, the equilibrium magnetic field is represented by

$$\boldsymbol{B} = c_b \nabla x \times \nabla y = \frac{c_b}{\sqrt{g}} \frac{\partial \boldsymbol{r}}{\partial z}, \tag{2.10}$$

where $c_b = (d\Psi_p/d\rho_f)/(c_x c_y)$ and $\sqrt{g} = (\nabla x \cdot \nabla y \times \nabla z)^{-1}$.

Simulation domain of GKV is based on the local flux-tube model [2-3]. Using flute approximation for perturbed quantities $k_\perp \gg k_\parallel$ (consistent with the gyrokinetic ordering Eq. (2.1), vector differential operators in gyrokinetic Eqs. (2.2)-(2.4) become

$$\nabla_\parallel \tilde{f} = \boldsymbol{b} \cdot \nabla \tilde{f} = \frac{c_b}{B\sqrt{g}} \frac{\partial \tilde{f}}{\partial z}, \tag{2.11}$$

$$\begin{aligned} \nabla^2 \tilde{f} &= \frac{1}{\sqrt{g}} \frac{\partial}{\partial r^i} \left[\sqrt{g} \left(\frac{\partial \tilde{f}}{\partial r^j} \nabla r^j \right) \cdot \nabla r^i \right] \\ &\simeq g^{xx} \frac{\partial^2 \tilde{f}}{\partial x^2} + 2g^{xy} \frac{\partial^2 \tilde{f}}{\partial x \partial y} + g^{yy} \frac{\partial^2 \tilde{f}}{\partial y^2}, \end{aligned} \tag{2.12}$$

$$\begin{aligned} \boldsymbol{b} \times \nabla \tilde{h} \cdot \nabla \tilde{f} &= \boldsymbol{b} \cdot \left(\frac{\partial \tilde{h}}{\partial r^i} \nabla r^i \times \frac{\partial \tilde{f}}{\partial r^j} \nabla r^j \right) \\ &\simeq \frac{B}{c_b} \left(\frac{\partial \tilde{h}}{\partial x} \frac{\partial \tilde{f}}{\partial y} - \frac{\partial \tilde{h}}{\partial y} \frac{\partial \tilde{f}}{\partial x} \right), \end{aligned} \tag{2.13}$$

$$\begin{aligned} \boldsymbol{b} \times \nabla H \cdot \nabla \tilde{f} &\simeq \frac{B}{c_b} \left(\frac{\partial H}{\partial x} \frac{\partial \tilde{f}}{\partial y} - \frac{\partial H}{\partial y} \frac{\partial \tilde{f}}{\partial x} \right) \\ &+ \frac{\partial H}{\partial z} \left(\frac{g^{xz} g^{yx} - g^{xx} g^{yz}}{B/c_b} \frac{\partial \tilde{f}}{\partial x} + \frac{g^{xz} g^{yy} - g^{xy} g^{yz}}{B/c_b} \frac{\partial \tilde{f}}{\partial y} \right), \end{aligned} \tag{2.14}$$

where $g^{ij} = \nabla r^i \cdot \nabla r^j$ denotes the metric tensor.

Since the magnetic curvature can be replaced by

$$\boldsymbol{b} \cdot \nabla \boldsymbol{b} = \frac{\nabla_\perp B}{B} + \frac{\nabla P}{B^2/\mu_0}, \tag{2.15}$$

when the equilibrium satisfies the MHD equilibrium, $\nabla P = \boldsymbol{J} \times \boldsymbol{B}$ and $\nabla \times \boldsymbol{B} = \mu_0 \boldsymbol{J}$, the magnetic (i.e., grad-B and curvature) drift velocity is given by

$$\boldsymbol{v}_{\mathrm{sG}} + \boldsymbol{v}_{\mathrm{sC}} = \frac{1}{e_{\mathrm{s}} B} \boldsymbol{b} \times \left(\frac{m_{\mathrm{s}} v_\parallel^2 + \mu B}{B} \nabla B + \frac{m_{\mathrm{s}} v_\parallel^2}{B^2/\mu_0} \nabla P \right), \tag{2.16}$$

and then the magnetic and diamagnetic drift terms are

$$\begin{aligned} (\boldsymbol{v}_{\mathrm{sG}} + \boldsymbol{v}_{\mathrm{sC}}) \cdot \nabla(J_0 \tilde{\phi}) =& \frac{m_{\mathrm{s}} v_\parallel^2 + \mu B}{e_{\mathrm{s}} c_b} \left(K_x \frac{\partial J_0 \tilde{\phi}}{\partial x} + K_y \frac{\partial J_0 \tilde{\phi}}{\partial y} \right) \\ &+ \frac{m_{\mathrm{s}} v_\parallel^2}{e_{\mathrm{s}} c_b} \frac{dP/dx}{B^2/\mu_0} \frac{\partial J_0 \tilde{\phi}}{\partial y}, \end{aligned} \tag{2.17}$$

$$\boldsymbol{v}_{\mathrm{s}*} \cdot \nabla(J_0 \tilde{\phi}) = - \frac{T_{\mathrm{s}}}{e_{\mathrm{s}} c_b} \left[\frac{1}{L_{n\mathrm{s}}} + \left(\frac{m_{\mathrm{s}} v_\parallel^2}{2 T_{\mathrm{s}}} + \frac{\mu B}{T_{\mathrm{s}}} - \frac{3}{2} \right) \frac{1}{L_{T\mathrm{s}}} \right] \frac{\partial J_0 \tilde{\phi}}{\partial y}, \tag{2.18}$$

where

$$K_x = - \frac{\partial \ln B}{\partial y} + \frac{g^{xz} g^{yx} - g^{xx} g^{yz}}{B^2/c_b^2} \frac{\partial \ln B}{\partial z}, \tag{2.19}$$

$$K_y = \frac{\partial \ln B}{\partial x} + \frac{g^{xz} g^{yy} - g^{xy} g^{yz}}{B^2/c_b^2} \frac{\partial \ln B}{\partial z}, \tag{2.20}$$

and the density and temperature scale lengths $L_{n\mathrm{s}} = -(d \ln n_{\mathrm{s}}/dx)^{-1}$, $L_{T\mathrm{s}} = -(d \ln T_{\mathrm{s}}/dx)^{-1}$, and total pressure gradient $dP/dx = d(\sum_{\mathrm{s}} n_{\mathrm{s}} T_{\mathrm{s}})/dx = -\sum_{\mathrm{s}} n_{\mathrm{s}} T_{\mathrm{s}} (L_{n\mathrm{s}}^{-1} + L_{T\mathrm{s}}^{-1})$.

2.3 Local approximation

Simulation box $-L_x \le x < L_x$, $-L_y \le y < L_y$, $-N_\theta \pi < z < N_\theta \pi$ gives flux-tube domain aligned to the equilibrium magnetic field.

By assuming the perpendicular scale separation of equilibrium and perturbed quantities, the equilibrium quantities can be evaluated by the value at the center of flux-tube domain $x = 0$ or equivalently $\rho_f = \rho_{f0}$. When one considers an axisymmetric equilibrium $\partial_y = 0$, the equilibrium quantities are independent to x and y, i.e., $F = F(z, v_\parallel, \mu)$, $B = B(z)$, and so on. In a non-axisymmetric equilibrium case, one may treat a thin flux-tube domain not only in x but also in y direction and evaluate the equilibrium quantities at $x = 0$ and $y = 0$.

2.4 Pseudo-periodic boundary condition along a field line

Since the equilibrium quantities are independent to perpendicular x and y directions, one expand the distribution function and electromagnetic potentials by means of Fourier basis,

$$\tilde{f}_{\mathrm{s}}(\boldsymbol{x}, v_\parallel, \mu, t) = \sum_{k_x}\sum_{k_y} \tilde{f}_{\mathrm{s}\boldsymbol{k}}(z, v_\parallel, \mu, t) e^{i(k_x x + i k_y y)} \tag{2.21}$$

$$\tilde{\phi}(\boldsymbol{x}', t) = \sum_{k_x}\sum_{k_y} \tilde{\phi}_{\boldsymbol{k}}(z, t) e^{i(k_x x' + i k_y y')} \tag{2.22}$$

$$J_{0\mathrm{s}}\tilde{\phi}(\boldsymbol{x}, \mu, t) = \sum_{k_x}\sum_{k_y} J_0(k_\perp \rho_{\mathrm{ts}})\tilde{\phi}_{\boldsymbol{k}}(z, t) e^{i(k_x x + i k_y y)} \tag{2.23}$$

where $\boldsymbol{x}$ is the gyrocenter coordinates and $\boldsymbol{x}' = \boldsymbol{x} + \boldsymbol{\rho}_{\mathrm{s}}$ is the particle-position coordinates.

Additionally, considering the torus periodicity constraint $\tilde{\phi}(\rho_f, \theta_f + 2N_\theta\pi, \varphi_f) = \tilde{\phi}(\rho_f, \theta_f, \varphi_f)$, one finds the pseudo-periodic boundary condition along a field line,

$$\tilde{\phi}_{k_x+\delta k_x, k_y}(z + 2N_\theta \pi) C_{k_y} = \tilde{\phi}_{k_x, k_y}(z), \tag{2.24}$$

where $\delta k_x = -2N_\theta\pi\hat{s}k_y, C_{k_y} = \exp(i2N_\theta\pi k_y c_y q_0)$. This conversion along a field line physically means twisting of the mode by the parallel streaming in the presence of magnetic shear.

2.5 Collision operator

The present version of GKV equips three types of gyrokinetic model collision operators, operating on the non-adiabatic part of the distribution function $\tilde{g}_{\mathrm{s}\boldsymbol{k}} = \tilde{f}_{\mathrm{s}\boldsymbol{k}} + \frac{e_{\mathrm{s}} F_{\mathrm{Ms}}}{T_{\mathrm{s}}} J_{0\mathrm{s}\boldsymbol{k}} \tilde{\phi}_{\boldsymbol{k}}$. NOTE: Although the Lenard-Bernstein model collision gkvp_f0.48 operates on $\tilde{f}_{\mathrm{s}\boldsymbol{k}}$ but not on $\tilde{g}_{\mathrm{s}\boldsymbol{k}}$ due to historical reason, it will be modified near-future update.

Lenard-Bernstein model collision operator

$$\begin{aligned} C_{\mathrm{a}\boldsymbol{k}}^{\mathrm{LB}} = \nu_{\mathrm{a}} \Bigg[& v_{\mathrm{ta}}^2 \frac{\partial^2 \tilde{g}_{\mathrm{a}\boldsymbol{k}}}{\partial v_\parallel^2} + v_{\mathrm{ta}}^2 \frac{\partial^2 \tilde{g}_{\mathrm{a}\boldsymbol{k}}}{\partial v_\perp^2} + v_\parallel \frac{\partial \tilde{g}_{\mathrm{a}\boldsymbol{k}}}{\partial v_\parallel} \\ & + \left(\frac{v_{\mathrm{ta}}^2}{v_\perp} + v_\perp \right) \frac{\partial \tilde{g}_{\mathrm{a}\boldsymbol{k}}}{\partial v_\perp} + 3\tilde{g}_{\mathrm{a}\boldsymbol{k}} - k_\perp^2 \rho_{\mathrm{ta}}^2 \tilde{g}_{\mathrm{a}\boldsymbol{k}} \Bigg]. \end{aligned} \tag{2.25}$$

Lorentz model collision operator

$$\begin{aligned} C_{\mathrm{a}\boldsymbol{k}}^{\mathrm{Lorentz}} = \nu_{\mathrm{D}}^{\mathrm{ab}} \Bigg[& \frac{v_\perp^2}{2} \frac{\partial^2 \tilde{g}_{\mathrm{a}\boldsymbol{k}}}{\partial v_\parallel^2} + \frac{v_\parallel^2}{2} \frac{\partial^2 \tilde{g}_{\mathrm{a}\boldsymbol{k}}}{\partial v_\perp^2} - v_\parallel v_\perp \frac{\partial^2 \tilde{g}_{\mathrm{a}\boldsymbol{k}}}{\partial v_\parallel \partial v_\perp} \\ & - v_\parallel \frac{\partial \tilde{g}_{\mathrm{a}\boldsymbol{k}}}{\partial v_\parallel} + \frac{v_\perp}{2} \left(\frac{v_\parallel^2}{v_\perp^2} - 1 \right) \frac{\partial \tilde{g}_{\mathrm{a}\boldsymbol{k}}}{\partial v_\perp} \\ & - \frac{k_\perp^2 \rho_{\mathrm{ta}}^2}{4 v_{\mathrm{ta}}^2} (2v_\parallel^2 + v_\perp^2) \tilde{g}_{\mathrm{a}\boldsymbol{k}} \Bigg]. \end{aligned} \tag{2.26}$$

Sugama model collision operator [2-4]

$$C_{\mathrm{a}\boldsymbol{k}}^{\mathrm{Sugama}} = \sum_{\mathrm{b}} \left[C_{\mathrm{ab}}^{\mathrm{V}}(\tilde{g}_{\mathrm{a}\boldsymbol{k}}) + C_{\mathrm{ab}}^{\mathrm{D}}(\tilde{g}_{\mathrm{a}\boldsymbol{k}}) + C_{\mathrm{ab}}^{\mathrm{F}}(\tilde{g}_{\mathrm{b}\boldsymbol{k}}) \right]. \tag{2.27}$$

The test-particle differential term $C_{\mathrm{ab}}^{\mathrm{V}}$, the test-particle non-isothermal term $C_{\mathrm{ab}}^{\mathrm{D}}$, and the field-particle term $C_{\mathrm{ab}}^{\mathrm{F}}$ are given by,

$$\begin{aligned}
C_{\mathrm{ab}}^{\mathrm{V}}(\tilde{g}_{\mathrm{a}\boldsymbol{k}}) = & \frac{\nu_{\|}^{\mathrm{ab}} v_{\|}^2 + \nu_{\mathrm{D}}^{\mathrm{ab}} v_{\perp}^2}{2} \frac{\partial^2 \tilde{g}_{\mathrm{a}\boldsymbol{k}}}{\partial v_{\|}^2} + \frac{\nu_{\mathrm{D}}^{\mathrm{ab}} v_{\|}^2 + \nu_{\|}^{\mathrm{ab}} v_{\perp}^2}{2} \frac{\partial^2 \tilde{g}_{\mathrm{a}\boldsymbol{k}}}{\partial v_{\perp}^2} \\
& + (\nu_{\|}^{\mathrm{ab}} - \nu_{\mathrm{D}}^{\mathrm{ab}}) v_{\|} v_{\perp} \frac{\partial^2 \tilde{g}_{\mathrm{a}\boldsymbol{k}}}{\partial v_{\|} \partial v_{\perp}} \\
& + \nu_{\mathrm{g}}^{\mathrm{ab}} v_{\|} \frac{\partial \tilde{g}_{\mathrm{a}\boldsymbol{k}}}{\partial v_{\|}} + \left[\nu_{\mathrm{g}}^{\mathrm{ab}} + \frac{\nu_{\mathrm{D}}^{\mathrm{ab}}}{2} \left(1 + \frac{v_{\|}^2}{v_{\perp}^2} \right) \right] v_{\perp} \frac{\partial \tilde{g}_{\mathrm{a}\boldsymbol{k}}}{\partial v_{\perp}} \\
& + \left[\frac{\nu_{\mathrm{h}}^{\mathrm{ab}} x_{\mathrm{a}}^2}{2} - \frac{k_{\perp}^2}{4\Omega_{\mathrm{a}}^2} \left\{ \nu_{\mathrm{D}}^{\mathrm{ab}} (2 v_{\|}^2 + v_{\perp}^2) + \nu_{\|}^{\mathrm{ab}} v_{\perp}^2 \right\} \right] \tilde{g}_{\mathrm{a}\boldsymbol{k}}, \qquad (2.28)
\end{aligned}$$

$$C_{\mathrm{ab}}^{\mathrm{D}}(\tilde{g}_{\mathrm{a}\boldsymbol{k}}) = \sum_{j=1}^{6} X_j^{\mathrm{ab}} M_j^{\mathrm{ab}}, \qquad (2.29)$$

$$C_{\mathrm{ab}}^{\mathrm{F}}(\tilde{g}_{\mathrm{b}\boldsymbol{k}}) = \sum_{j=1}^{6} Y_j^{\mathrm{ab}} M_j^{\mathrm{ba}}, \qquad (2.30)$$

where $x_{\mathrm{a}} = v/(\sqrt{2} v_{\mathrm{ta}})$, $\alpha_{\mathrm{ab}} = v_{\mathrm{ta}}/v_{\mathrm{tb}}$, $\nu_{\mathrm{g}}^{\mathrm{ab}} = \nu_{\|}^{\mathrm{ab}} x_{\mathrm{a}}^2 (1 - \alpha_{\mathrm{ab}})$, and $\nu_{\mathrm{h}}^{\mathrm{ab}} = 3\sqrt{\pi} \tau_{\mathrm{ab}}^{-1} \alpha_{\mathrm{ab}} \Phi'(x_{\mathrm{b}})/(4 x_{\mathrm{a}}^2)$. The energy-diffusion and deflection frequencies are respectively given by $\nu_{\|}^{\mathrm{ab}} = 3\sqrt{\pi} \tau_{\mathrm{ab}}^{-1} G(x_{\mathrm{b}})/(2 x_{\mathrm{a}}^3)$ and $\nu_{\mathrm{D}}^{\mathrm{ab}} = 3\sqrt{\pi} \tau_{\mathrm{ab}}^{-1} [\Phi(x_{\mathrm{b}}) - G(x_{\mathrm{b}})]/(4 x_{\mathrm{a}}^3)$ with the error function $\Phi(x) = \mathrm{erf}(x)$ and $G(x) = [\Phi(x) - x\Phi'(x)]/(2x^2)$. Expressions of the other coefficients X_j^{ab} and Y_j^{ab} and of the fluid moments M_j^{ab} are found, e.g., in the literature [2-5].

2.6 Summary of formulation

Finally, one obtains the δf gyrokinetic Vlasov-Poisson-Ampère equations in a local flux-tube model, represented in perpendicular wave-number space,

$$\begin{aligned}
& \frac{\partial \tilde{f}_{\mathrm{s}\boldsymbol{k}}}{\partial t} + \left(v_{\|} \nabla_{\|} + i\boldsymbol{k} \cdot \boldsymbol{v}_{\mathrm{sG}} + i\boldsymbol{k} \cdot \boldsymbol{v}_{\mathrm{sC}} \right) \left(\tilde{f}_{\mathrm{s}\boldsymbol{k}} + \frac{e_{\mathrm{s}} F_{\mathrm{sM}}}{T_{\mathrm{s}}} J_{0\mathrm{s}\boldsymbol{k}} \tilde{\phi}_{\boldsymbol{k}} \right) + N_{\mathrm{s}\boldsymbol{k}} \\
& - \frac{\mu \nabla_{\|} B}{m_{\mathrm{s}}} \frac{\partial}{\partial v_{\|}} \left(\tilde{f}_{\mathrm{s}\boldsymbol{k}} + \frac{e_{\mathrm{s}} F_{\mathrm{sM}}}{T_{\mathrm{s}}} J_{0\mathrm{s}\boldsymbol{k}} \tilde{\phi}_{\boldsymbol{k}} \right) \\
& + \frac{e_{\mathrm{s}} F_{\mathrm{sM}}}{T_{\mathrm{s}}} \left[v_{\|} \frac{\partial J_{0\mathrm{s}\boldsymbol{k}} \tilde{A}_{\|\boldsymbol{k}}}{\partial t} - i\boldsymbol{k} \cdot \boldsymbol{v}_{\mathrm{s}*} J_{0\mathrm{s}\boldsymbol{k}} (\tilde{\phi}_{\boldsymbol{k}} - v_{\|} \tilde{A}_{\|\boldsymbol{k}}) \right] = C_{\mathrm{s}\boldsymbol{k}}, \qquad (2.31)
\end{aligned}$$

$$\left[k_{\perp}^2 + \frac{1}{\varepsilon_0} \sum_{\mathrm{s}} \frac{e_{\mathrm{s}}^2 n_{\mathrm{s}}}{T_{\mathrm{s}}} (1 - \Gamma_{0\mathrm{s}\boldsymbol{k}}) \right] \tilde{\phi}_{\boldsymbol{k}} = \frac{1}{\varepsilon_0} \sum_{\mathrm{s}} e_{\mathrm{s}} \int dv^3 J_{0\mathrm{s}\boldsymbol{k}} \tilde{f}_{\mathrm{s}\boldsymbol{k}}, \qquad (2.32)$$

$$k_{\perp}^2 \tilde{A}_{\|\boldsymbol{k}} = \mu_0 \sum_{\mathrm{s}} e_{\mathrm{s}} \int dv^3 J_{0\mathrm{s}\boldsymbol{k}} v_{\|} \tilde{f}_{\mathrm{s}\boldsymbol{k}}, \qquad (2.33)$$

where $J_{0\mathrm{s}\boldsymbol{k}} = J_0(k_{\perp} \rho_{\mathrm{s}})$ and $\Gamma_{0\mathrm{s}\boldsymbol{k}} = I_0(k_{\perp}^2 \rho_{\mathrm{ts}}^2) e^{-k_{\perp}^2 \rho_{\mathrm{ts}}^2}$ with 0th-order Bessel and modified Bessel functions J_0 and I_0. The included operators are again listed below,

$$\nabla_\parallel = \frac{c_b}{B\sqrt{g}}\frac{\partial}{\partial z}, \tag{2.34}$$

$$k_\perp^2 = g^{xx}k_x^2 + 2g^{xy}k_xk_y + g^{yy}k_y^2, \tag{2.35}$$

$$i\boldsymbol{k}\cdot(\boldsymbol{v}_{\mathrm{sG}}+\boldsymbol{v}_{\mathrm{sC}}) = \frac{m_\mathrm{s}v_\parallel^2+\mu B}{e_\mathrm{s}c_b}(iK_xk_x+iK_yk_y) + i\frac{m_\mathrm{s}v_\parallel^2}{e_\mathrm{s}c_b}\frac{dP/dx}{B^2/\mu_0}k_y, \tag{2.36}$$

$$i\boldsymbol{k}\cdot\boldsymbol{v}_{\mathrm{s}*} = -i\frac{T_\mathrm{s}}{e_\mathrm{s}c_b}\left[\frac{1}{L_{n\mathrm{s}}} + \left(\frac{m_\mathrm{s}v_\parallel^2}{2T_\mathrm{s}} + \frac{\mu B}{T_\mathrm{s}} - \frac{3}{2}\right)\frac{1}{L_{T\mathrm{s}}}\right]k_y, \tag{2.37}$$

$$\begin{aligned}\mathcal{N}_{\mathrm{s}\boldsymbol{k}} = &-\sum_{\boldsymbol{k}'}\sum_{\boldsymbol{k}''}\delta_{\boldsymbol{k}'+\boldsymbol{k}'',\boldsymbol{k}}\frac{\boldsymbol{b}\cdot\boldsymbol{k}'\times\boldsymbol{k}''}{c_b}J_{0\mathrm{s}\boldsymbol{k}'}\\ &\times\left(\tilde{\phi}_{\boldsymbol{k}'} - v_\parallel\tilde{A}_{\parallel\boldsymbol{k}'}\right)\left(\tilde{f}_{\mathrm{s}\boldsymbol{k}''} + \frac{e_\mathrm{s}F_{\mathrm{Ms}}}{T_\mathrm{s}}J_{0\mathrm{s}\boldsymbol{k}''}\tilde{\phi}_{\boldsymbol{k}''}\right).\end{aligned} \tag{2.38}$$

The coefficients for magnetic drift K_x, K_y are given by Eqs. (2.19) and (2.20), and the collision operator $C_{\mathrm{s}\boldsymbol{k}}$ is given by one of Eqs. (2.25) - (2.27).

2.7 References

[2-1] T.-H. Watanabe, and H. Sugama, *Nucl. Fusion* **46**, 24 (2006).
[2-2] X. Garbet, Y. Idomura, L. Villard, and T.-H. Watanabe, *Nucl. Fusion* **50**, 043002 (2010).
[2-3] M. A. Beer, S. C. Cowley, and G. W. Hammett, *Phys. Plasmas* **2**, 2687 (1995).
[2-4] H. Sugama, T.-H. Watanabe, and M. Nunami, *Phys. Plasmas* **16**, 112503 (2009).
[2-5] M. Nakata, M. Nunami, T.-H. Watanabe, and H. Sugama, *Comput. Phys. Commun.* **197**, 61 (2015).

3 Normalization

3.1 Reference units

We denote the reference values of physical quantities as follows.

- Reference magnetic field strength B_ref ($= B_a$ magnetic field strength at the magnetic axis)
- Reference length L_ref ($= R_a$ major radius at the magnetic axis)
- Reference density n_ref ($= n_e(\rho_0)$ electron density at the center of flux-tube domain)
- Reference temperature T_ref ($= T_i(\rho_0)$ main ion temperature at the center of flux-tube domain)
- Reference mass m_ref ($= m_\mathrm{p}$ the proton mass)
- Reference electric charge e_ref ($= e$ elementary charge)

We also define the following notations $v_\mathrm{ref} = \sqrt{T_\mathrm{ref}/m_\mathrm{ref}}$, $\rho_\mathrm{ref} = m_\mathrm{ref}v_\mathrm{ref}/(e_\mathrm{ref}B_\mathrm{ref})$, $\delta_\mathrm{ref} = \rho_\mathrm{ref}/L_\mathrm{ref}$. For single-species simulations with adiabatic electron/ion models, see Appendix A.6.

3.2 Normalized equations

We represent a dimensionless quantity by an overline, $\bar{f}$, in this section.

The coordinates, variables, operators are normalized as

$$
\begin{aligned}
&t = \frac{L_{\rm ref}}{v_{\rm ref}}\bar{t},\ \ x = \rho_{\rm ref}\bar{x},\ \ k_x = \frac{1}{\rho_{\rm ref}}\bar{k}_x,\ \ y = \rho_{\rm ref}\bar{y},\ \ k_y = \frac{1}{\rho_{\rm ref}}\bar{k}_y,\ \ z = \bar{z},\\
&v_\parallel = v_{\rm ts}\bar{v}_\parallel = v_{\rm ref}\sqrt{\frac{\bar{T}_{\rm s}}{\bar{m}_{\rm s}}}\bar{v}_\parallel,\ \ \mu = \frac{T_{\rm s}}{B_{\rm ref}}\bar{\mu} = \frac{T_{\rm ref}}{B_{\rm ref}}T_{\rm s}\bar{\mu},\\
&\tilde{f}_{{\rm s}\boldsymbol{k}} = \delta_{\rm ref}\frac{n_{\rm s}}{v_{\rm ts}^3}\bar{f}_{{\rm s}\boldsymbol{k}},\ \ \tilde{\phi}_{\boldsymbol{k}} = \delta_{\rm ref}\frac{T_{\rm ref}}{e_{\rm ref}}\bar{\phi}_{\boldsymbol{k}},\ \ \tilde{A}_{\parallel\boldsymbol{k}} = \delta_{\rm ref}\rho_{\rm ref}B_{\rm ref}\bar{A}_{\parallel\boldsymbol{k}},\\
&n_{\rm s} = n_{\rm ref}\bar{n}_{\rm s},\ \ T_{\rm s} = T_{\rm ref}\bar{T}_{\rm s},\ \ m_{\rm s} = m_{\rm ref}\bar{m}_{\rm s},\ \ e_{\rm s} = e_{\rm ref}\bar{e}_{\rm s},\\
&\nabla_\parallel = \frac{1}{L_{\rm ref}}\bar{\nabla}_\parallel,\ \ \boldsymbol{v}_{\rm sG} = \delta_{\rm ref}v_{\rm ref}\bar{\boldsymbol{v}}_{\rm sG},\ \ \boldsymbol{v}_{\rm sC} = \delta_{\rm ref}v_{\rm ref}\bar{\boldsymbol{v}}_{\rm sC},\ \ \boldsymbol{v}_{{\rm s}*} = \delta_{\rm ref}v_{\rm ref}\bar{\boldsymbol{v}}_{{\rm s}*},\\
&F_{\rm sM} = \frac{n_{\rm s}}{v_{\rm ts}^3}\bar{F}_{\rm sM},\ \ J_{0{\rm s}\boldsymbol{k}} = \bar{J}_{0{\rm s}\boldsymbol{k}},\ \ \Gamma_{0{\rm s}\boldsymbol{k}} = \bar{\Gamma}_{0{\rm s}\boldsymbol{k}},\\
&K_x = \frac{1}{L_{\rm ref}}\bar{K}_x,\ \ K_y = \frac{1}{L_{\rm ref}}\bar{K}_y, L_{n{\rm s}} = L_{\rm ref}\bar{L}_{n{\rm s}},\ \ L_{T{\rm s}} = L_{\rm ref}\bar{L}_{T{\rm s}},\\
&\frac{dP}{dx} = \frac{n_{\rm ref}T_{\rm ref}}{L_{\rm ref}}\frac{d\bar{P}}{d\bar{x}},\ \ c_b = B_{\rm ref}\bar{c}_b,\ \ B = B_{\rm ref}\bar{B},\\
&\frac{\partial \ln B}{\partial x} = \frac{1}{L_{\rm ref}}\frac{\partial \ln \bar{B}}{\partial \bar{x}},\ \ \frac{\partial \ln B}{\partial y} = \frac{1}{L_{\rm ref}}\frac{\partial \ln \bar{B}}{\partial \bar{y}},\ \ \frac{\partial \ln B}{\partial z} = \frac{\partial \ln \bar{B}}{\partial \bar{z}},\\
&g^{xx} = \bar{g}^{xx},\ \ g^{xy} = \bar{g}^{xy}, g^{xz} = \frac{1}{L_{\rm ref}}\bar{g}^{xz},\ \ g^{yy} = \bar{g}^{yy},\\
&g^{yz} = \frac{1}{L_{\rm ref}}\bar{g}^{yz},\ \ g^{zz} = \frac{1}{L_{\rm ref}^2}\bar{g}^{zz},\ \ \sqrt{g} = L_{\rm ref}\sqrt{\bar{g}},\ \ \nu = \frac{v_{\rm ref}}{L_{\rm ref}}\bar{\nu},\\
&N_{{\rm s}\boldsymbol{k}} = \frac{v_{\rm ref}}{L_{\rm ref}}\delta_{\rm ref}\frac{n_{\rm s}}{v_{\rm ts}^3}\bar{N}_{{\rm s}\boldsymbol{k}},\ \ C_{{\rm s}\boldsymbol{k}} = \frac{v_{\rm ref}}{L_{\rm ref}}\delta_{\rm ref}\frac{n_{\rm s}}{v_{\rm ts}^3}\bar{C}_{{\rm s}\boldsymbol{k}}.
\end{aligned}
\tag{3.1}
$$

Then, the normalized equations are

$$
\begin{aligned}
&\frac{\partial \bar{f}_{{\rm s}\boldsymbol{k}}}{\partial \bar{t}} + \left(\sqrt{\frac{\bar{T}_{\rm s}}{\bar{m}_{\rm s}}}\bar{v}_\parallel\bar{\nabla}_\parallel + i\bar{\boldsymbol{k}}\cdot\bar{\boldsymbol{v}}_{\rm sG} + i\bar{\boldsymbol{k}}\cdot\bar{\boldsymbol{v}}_{\rm sC}\right)\left(\bar{f}_{{\rm s}\boldsymbol{k}} + \frac{\bar{e}_{\rm s}\bar{F}_{\rm sM}}{\bar{T}_{\rm s}}\bar{J}_{0{\rm s}\boldsymbol{k}}\bar{\phi}_{\boldsymbol{k}}\right)\\
&+ \bar{N}_{{\rm s}\boldsymbol{k}} - \sqrt{\frac{\bar{T}_{\rm s}}{\bar{m}_{\rm s}}}\bar{\mu}\bar{\nabla}_\parallel\bar{B}\frac{\partial}{\partial \bar{v}_\parallel}\left(\bar{f}_{\rm s} + \frac{\bar{e}_{\rm s}\bar{F}_{\rm sM}}{\bar{T}_{\rm s}}\bar{J}_{0{\rm s}}\bar{\phi}\right)\\
&+ \frac{e_{\rm s}F_{\rm sM}}{T_{\rm s}}\left[\sqrt{\frac{\bar{T}_{\rm s}}{\bar{m}_{\rm s}}}\bar{v}_\parallel\frac{\partial \bar{J}_{0{\rm s}}\bar{A}_\parallel}{\partial \bar{t}} - i\bar{\boldsymbol{k}}\cdot\bar{\boldsymbol{v}}_{{\rm s}*}\bar{J}_{0{\rm s}}(\bar{\phi} - \sqrt{\frac{\bar{T}_{\rm s}}{\bar{m}_{\rm s}}}\bar{v}_\parallel\bar{A}_\parallel)\right] = \bar{C}_{{\rm s}\boldsymbol{k}},
\end{aligned}
\tag{3.2}
$$

$$
\left[\bar{\lambda}_{\rm D}^2\bar{k}_\perp^2 + \sum_{\rm s}\frac{\bar{e}_{\rm s}^2\bar{n}_{\rm s}}{\bar{T}_{\rm s}}\left(1 - \bar{\Gamma}_{0{\rm s}\boldsymbol{k}}\right)\right]\bar{\phi}_{\boldsymbol{k}} = \sum_{\rm s}\bar{e}_{\rm s}\bar{n}_{\rm s}\int d\bar{v}^3\bar{J}_{0{\rm s}\boldsymbol{k}}\bar{f}_{{\rm s}\boldsymbol{k}},
\tag{3.3}
$$

$$
\bar{k}_\perp^2\bar{A}_{\parallel\boldsymbol{k}} = \bar{\beta}\sum_{\rm s}\bar{e}_{\rm s}\bar{n}_{\rm s}\int d\bar{v}^3\bar{J}_{0{\rm s}\boldsymbol{k}}\sqrt{\frac{\bar{T}_{\rm s}}{\bar{m}_{\rm s}}}\bar{v}_\parallel\bar{f}_{{\rm s}\boldsymbol{k}},
\tag{3.4}
$$

where we introduced $\bar{\lambda}_{\rm D}^2 = \frac{\lambda_{\rm D,ref}^2}{\rho_{\rm ref}^2} = \frac{\varepsilon_0 T_{\rm ref}/e_{\rm ref}^2 n_{\rm ref}}{\rho_{\rm ref}^2}$ and $\bar{\beta} = \frac{\mu_0 n_{\rm ref}T_{\rm ref}}{B_{\rm ref}^2}$. The included terms and operators are

$$
\begin{aligned}
&\bar{\nabla}_\parallel = \frac{\bar{c}_b}{\bar{B}\sqrt{\bar{g}}}\frac{\partial}{\partial z}, \quad \bar{k}_\perp^2 = \bar{g}^{xx}\bar{k}_x^2 + 2\bar{g}^{xy}\bar{k}_x\bar{k}_y + \bar{g}^{yy}\bar{k}_y^2, \\
&i\bar{\boldsymbol{k}}\cdot(\bar{\boldsymbol{v}}_{\mathrm{sG}} + \bar{\boldsymbol{v}}_{\mathrm{sC}}) = \frac{\bar{T}_\mathrm{s}(\bar{v}_\parallel^2 + \bar{\mu}\bar{B})}{\bar{e}_\mathrm{s}\bar{c}_b}\left(i\bar{K}_x\bar{k}_x + i\bar{K}_y\bar{k}_y\right) + i\frac{\bar{T}_\mathrm{s}\bar{v}_\parallel^2}{\bar{e}_\mathrm{s}\bar{c}_b}\bar{\beta}\frac{d\bar{P}/d\bar{x}}{\bar{B}^2}\bar{k}_y, \\
&i\bar{\boldsymbol{k}}\cdot\bar{\boldsymbol{v}}_{\mathrm{s}*} = -i\frac{\bar{T}_\mathrm{s}}{\bar{e}_\mathrm{s}\bar{c}_b}\left[\frac{1}{\bar{L}_{n\mathrm{s}}} + \left(\frac{\bar{v}_\parallel^2}{2} + \bar{\mu}\bar{B} - \frac{3}{2}\right)\frac{1}{\bar{L}_{T\mathrm{s}}}\right]\bar{k}_y, \\
&\bar{\mathcal{N}}_{\mathrm{s}\boldsymbol{k}} = -\sum_{\bar{\boldsymbol{k}}'}\sum_{\bar{\boldsymbol{k}}''}\delta_{\bar{\boldsymbol{k}}'+\bar{\boldsymbol{k}}'',\bar{\boldsymbol{k}}}\frac{\bar{\boldsymbol{b}}\cdot\bar{\boldsymbol{k}}'\times\bar{\boldsymbol{k}}''}{\bar{c}_b} \\
&\qquad\times\bar{J}_{0\mathrm{s}\boldsymbol{k}'}\left(\bar{\phi}_{\boldsymbol{k}'} - \sqrt{\frac{\bar{T}_\mathrm{s}}{\bar{m}_\mathrm{s}}}\bar{v}_\parallel\bar{A}_{\parallel\boldsymbol{k}'}\right)\left(\bar{f}_{\mathrm{s}\boldsymbol{k}''} + \frac{\bar{e}_\mathrm{s}\bar{F}_{\mathrm{Ms}}}{\bar{T}_\mathrm{s}}\bar{J}_{0\mathrm{s}\boldsymbol{k}''}\bar{\phi}_{\boldsymbol{k}''}\right), \\
&\bar{K}_x = -\frac{\partial\ln\bar{B}}{\partial\bar{y}} + \frac{\bar{g}^{xz}\bar{g}^{yx} - \bar{g}^{xx}\bar{g}^{yz}}{\bar{B}^2/\bar{c}_b^2}\frac{\partial\ln\bar{B}}{\partial\bar{z}}, \\
&\bar{K}_y = \frac{\partial\ln\bar{B}}{\partial\bar{x}} + \frac{\bar{g}^{xz}\bar{g}^{yy} - \bar{g}^{xy}\bar{g}^{yz}}{\bar{B}^2/\bar{c}_b^2}\frac{\partial\ln\bar{B}}{\partial\bar{z}}, \\
&\frac{d\bar{P}}{d\bar{x}} = -\sum_\mathrm{s}\bar{n}_\mathrm{s}\bar{T}_\mathrm{s}\left(\frac{1}{\bar{L}_{n\mathrm{s}}} + \frac{1}{\bar{L}_{T\mathrm{s}}}\right), \quad \bar{F}_{\mathrm{sM}} = \frac{1}{(2\pi)^{\frac{3}{2}}}e^{-\frac{\bar{v}_\parallel^2}{2}-\bar{\mu}\bar{B}}, \\
&\bar{J}_{0\mathrm{s}\boldsymbol{k}} = J_0\left(\bar{k}_\perp\bar{\rho}_\mathrm{s}\right), \quad \bar{\Gamma}_{0\mathrm{s}\boldsymbol{k}} = I_0\left(\bar{k}_\perp^2\bar{\rho}_{\mathrm{ts}}^2\right)e^{-\bar{k}_\perp^2\bar{\rho}_{\mathrm{ts}}^2}, \\
&\bar{\rho}_\mathrm{s} = \sqrt{\frac{2\bar{m}_\mathrm{s}\bar{T}_\mathrm{s}\bar{\mu}}{\bar{e}_\mathrm{s}^2\bar{B}}}, \quad \bar{\rho}_{\mathrm{ts}} = \frac{\sqrt{\bar{m}_\mathrm{s}\bar{T}_\mathrm{s}}}{\bar{e}_\mathrm{s}\bar{B}}
\end{aligned} \tag{3.5}
$$

and the normalized Lenard-Bernstein, Lorentz, and Sugama collision operators are

$$
\bar{C}_{\mathrm{a}\boldsymbol{k}}^{\mathrm{LB}} = \bar{\nu}_\mathrm{a}\left[\frac{\partial^2\bar{g}_{\mathrm{a}\boldsymbol{k}}}{\partial\bar{v}_\parallel^2} + \frac{\partial^2\bar{g}_{\mathrm{a}\boldsymbol{k}}}{\partial\bar{v}_\perp^2} + \bar{v}_\parallel\frac{\partial\bar{g}_{\mathrm{a}\boldsymbol{k}}}{\partial\bar{v}_\parallel} + \left(\frac{1}{\bar{v}_\perp} + \bar{v}_\perp\right)\frac{\partial\bar{g}_{\mathrm{a}\boldsymbol{k}}}{\partial\bar{v}_\perp} + 3\bar{g}_{\mathrm{a}\boldsymbol{k}} - \bar{k}_\perp^2\bar{\rho}_{\mathrm{ts}}^2\bar{g}_{\mathrm{a}\boldsymbol{k}}\right], \tag{3.6}
$$

$$
\begin{aligned}
\bar{C}_{\mathrm{a}\boldsymbol{k}}^{\mathrm{Lorentz}} = \bar{\nu}_\mathrm{D}^{\mathrm{ab}}\Bigg[&\frac{\bar{v}_\perp^2}{2}\frac{\partial^2\bar{g}_{\mathrm{a}\boldsymbol{k}}}{\partial\bar{v}_\parallel^2} + \frac{\bar{v}_\parallel^2}{2}\frac{\partial^2\bar{g}_{\mathrm{a}\boldsymbol{k}}}{\partial\bar{v}_\perp^2} \\
&- \bar{v}_\parallel\bar{v}_\perp\frac{\partial^2\bar{g}_{\mathrm{a}\boldsymbol{k}}}{\partial\bar{v}_\parallel\partial\bar{v}_\perp} - \bar{v}_\parallel\frac{\partial\bar{g}_{\mathrm{a}\boldsymbol{k}}}{\partial\bar{v}_\parallel} + \frac{\bar{v}_\perp}{2}\left(\frac{\bar{v}_\parallel^2}{\bar{v}_\perp^2} - 1\right)\frac{\partial\bar{g}_{\mathrm{a}\boldsymbol{k}}}{\partial\bar{v}_\perp} \\
&- \frac{\bar{k}_\perp^2\bar{\rho}_{\mathrm{ta}}^2}{4}(2\bar{v}_\parallel^2 + \bar{v}_\perp^2)\bar{g}_{\mathrm{a}\boldsymbol{k}}\Bigg],
\end{aligned} \tag{3.7}
$$

$$\bar{C}^{\text{Sugama}}_{\mathrm{a}\boldsymbol{k}} = \sum_{\mathrm{b}} \left[C^{\mathrm{V}}_{\mathrm{ab}}(\bar{g}_{\mathrm{a}\boldsymbol{k}}) + C^{\mathrm{D}}_{\mathrm{ab}}(\bar{g}_{\mathrm{a}\boldsymbol{k}}) + C^{\mathrm{F}}_{\mathrm{ab}}(\bar{g}_{\mathrm{b}\boldsymbol{k}}) \right], \tag{3.8}$$

$$\begin{aligned} C^{\mathrm{V}}_{\mathrm{ab}}(\bar{g}_{\mathrm{a}\boldsymbol{k}}) = & \frac{\bar{\nu}^{\mathrm{ab}}_{\parallel}\bar{v}^2_{\parallel} + \bar{\nu}^{\mathrm{ab}}_{\mathrm{D}}\bar{v}^2_{\perp}}{2} \frac{\partial^2 \bar{g}_{\mathrm{a}\boldsymbol{k}}}{\partial \bar{v}^2_{\parallel}} \\ & + \frac{\bar{\nu}^{\mathrm{ab}}_{\mathrm{D}}\bar{v}^2_{\parallel} + \bar{\nu}^{\mathrm{ab}}_{\parallel}\bar{v}^2_{\perp}}{2} \frac{\partial^2 \bar{g}_{\mathrm{a}\boldsymbol{k}}}{\partial \bar{v}^2_{\perp}} + (\bar{\nu}^{\mathrm{ab}}_{\parallel} - \bar{\nu}^{\mathrm{ab}}_{\mathrm{D}})\bar{v}_{\parallel}\bar{v}_{\perp} \frac{\partial^2 \bar{g}_{\mathrm{a}\boldsymbol{k}}}{\partial \bar{v}_{\parallel}\partial \bar{v}_{\perp}} \\ & + \bar{\nu}^{\mathrm{ab}}_{\mathrm{g}}\bar{v}_{\parallel}\frac{\partial \bar{g}_{\mathrm{a}\boldsymbol{k}}}{\partial \bar{v}_{\parallel}} + \left[\bar{\nu}^{\mathrm{ab}}_{\mathrm{g}} + \frac{\bar{\nu}^{\mathrm{ab}}_{\mathrm{D}}}{2}\left(1 + \frac{\bar{v}^2_{\parallel}}{\bar{v}^2_{\perp}}\right)\right] \bar{v}_{\perp}\frac{\partial \bar{g}_{\mathrm{a}\boldsymbol{k}}}{\partial \bar{v}_{\perp}} \\ & + \left[\frac{\bar{\nu}^{\mathrm{ab}}_{\mathrm{h}}\bar{v}^2}{4} - \frac{\bar{k}^2_{\perp}\bar{\rho}^2_{\mathrm{ta}}}{4}\left\{ \bar{\nu}^{\mathrm{ab}}_{\mathrm{D}}(2\bar{v}^2_{\parallel} + \bar{v}^2_{\perp}) + \bar{\nu}^{\mathrm{ab}}_{\parallel}\bar{v}^2_{\perp} \right\} \right] \bar{g}_{\mathrm{a}\boldsymbol{k}}, \end{aligned} \tag{3.9}$$

$$C^{\mathrm{D}}_{\mathrm{ab}}(\bar{g}_{\mathrm{a}\boldsymbol{k}}) = \sum_{j=1}^{6} \bar{X}^{\mathrm{ab}}_j \bar{M}^{\mathrm{ab}}_j, \tag{3.10}$$

$$C^{\mathrm{F}}_{\mathrm{ab}}(\bar{g}_{\mathrm{b}\boldsymbol{k}}) = \sum_{j=1}^{6} \bar{Y}^{\mathrm{ab}}_j \bar{M}^{\mathrm{ba}}_j. \tag{3.11}$$

Normalized input parameters for GKV are summarized in Appendix A.1.

4 Discretization

4.1 Spatial discretization

GKV is a Vlasov (continuum) simulation code. The perpendicular directions are already given by a discrete representation in Fourier space (k_x, k_y). The other three directions $(z, v_\parallel, \mu)$ are discretized by an equidistant grid. Defining box sizes $-L_x \le \bar{x} < L_x, -L_y \le \bar{y} < L_y, -L_z \le \bar{z} < L_z, -L_v \le \bar{v}_\parallel \le L_v, 0 \le \bar{\mu} \le \frac{L_v^2}{2}$ and grid numbers $(2\mathtt{nx}+1, \mathtt{global\_ny}+1, 2\mathtt{global\_nz}, 2\mathtt{global\_nv}, \mathtt{global\_nm}+1)$ in $(k_x, k_y, z, v_\parallel, \mu)$, the grid of GKV is given by

$$\begin{aligned} k_x &= \mathtt{mx}\Delta k_x \ \ (-\mathtt{nx} \le \mathtt{mx} \le \mathtt{nx}), \\ k_y &= \mathtt{my}\Delta k_y \ \ (0 \le \mathtt{my} \le \mathtt{global\_ny}) \\ z &= \mathtt{iz}\Delta z \ \ (-\mathtt{global\_nz} \le \mathtt{iz} \le \mathtt{global\_nz} - 1), \\ v_\parallel &= -L_v + (\mathtt{iv} - 1)\Delta v_\parallel \ \ (1 \le \mathtt{iv} \le 2\mathtt{global\_nv}), \\ \mu &= \frac{(\mathtt{im}\Delta w)^2}{2} \ \ (0 \le \mathtt{im} \le \mathtt{global\_nm}), \end{aligned}$$

where $\Delta k_x = \frac{\pi}{L_x}, \Delta k_y = \frac{\pi}{L_y}, \Delta z = \frac{L_z}{\mathtt{global\_nz}}, \Delta v_\parallel = \frac{2L_v}{2\mathtt{global\_nv}-1}, \Delta w = \frac{L_v}{\mathtt{global\_nm}}$.

Derivatives in $(z, v_\parallel, \mu)$ are discretized by finite difference method, and then, GKV solves the δf gyrokinetic equations, Eqs. (3.2)-(3.4), in $(k_x, k_y, z, v_\parallel, \mu)$ space, except for the nonlinear term. Since direct calculation of nonlinear convolution in wavenumber space is computationally expensive, the nonlinear term is evaluated in the real space, employing $(2\mathtt{nxw}, 2\mathtt{nyw}, 2\mathtt{global\_nz}, 2\mathtt{global\_nv}, \mathtt{global\_nm}+1)$ grid points in $(x, y, z, v_\parallel, \mu)$, and is transformed back to the wavenumber space by means of 2D Fast Fourier Transform (FFT) algorithm and the 3/2 de-aliasing rule in (k_x, k_y).

To implement the pseudo-periodic boundary condition along a field line, Eq. (2.24), the shift of the radial wave number $\delta k_x(k_y) = -2N_\theta \pi \hat{s} k_y$ as a function of k_y should be equal to the integral multiple of the minimum radial wave number Δk_x. This gives a constraint between radial and field-line-label box sizes. In GKV, the minimum field-line-label wave number Δk_y and the ratio $m = \left| \frac{\delta k_x(\Delta k_y)}{N_\theta \Delta k_x} \right|$ are given in the namelist (`kymin` and `m_j`, respectively), and then, $\Delta k_x = |2\pi \hat{s} \Delta k_y / m|$, $L_x = \pi / \Delta k_x$, $L_y = \pi / \Delta k_y$.

4.2 Temporal discretization

4.2.1 Explicit implementation

GKV usually uses 4th-order explicit Runge-Kutta-Gill method. [4-1]

4.2.2 Explicit collisionless physics and implicit collision implementation

An alternative option is implicit collision solver which is useful for Lorentz or Sugama collision operators having velocity-dependent collision frequencies [4-2]. Splitting collisionless physics and collision operator by means of 2nd-order (Strang) operator split, the collisionless physics is solved by using 4th-order explicit Runge-Kutta-Gill method, while the collision operator is solved by using 2nd-order semi-implicit Crank-Nicolson method. Bi-CGSTAB method is used as an iterative matrix solver for implicit collision.

4.3 Inter-node decomposition by using MPI

The computations are parallelized by using the OpenMP/MPI hybrid parallelization which suites for hierarchical hardware of the nodes having a number of cores with a shared memory and connected by an interconnect network.

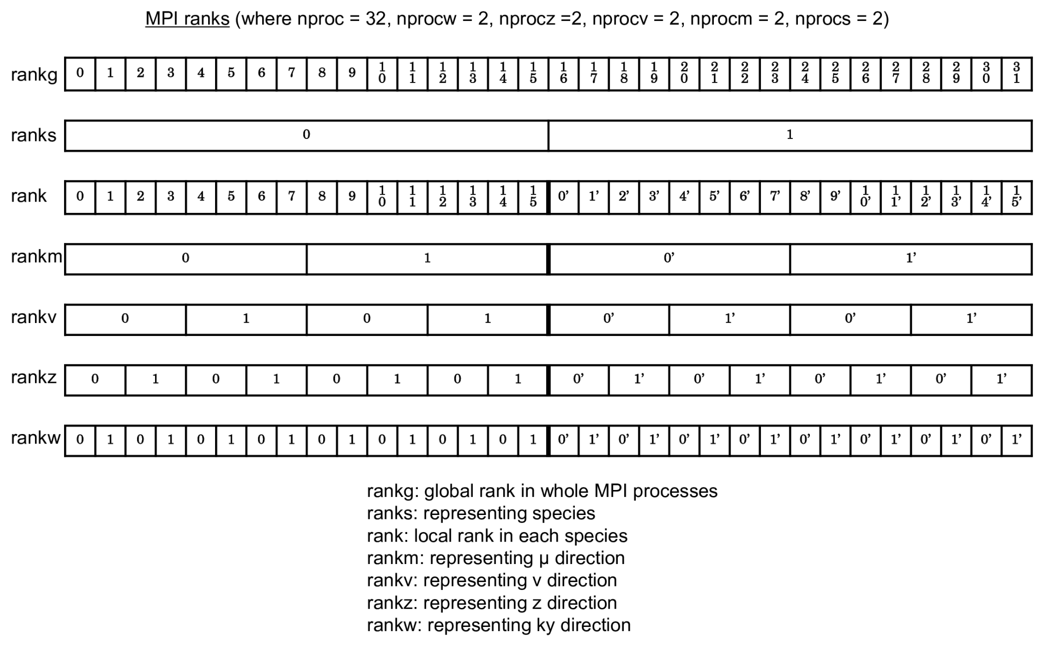
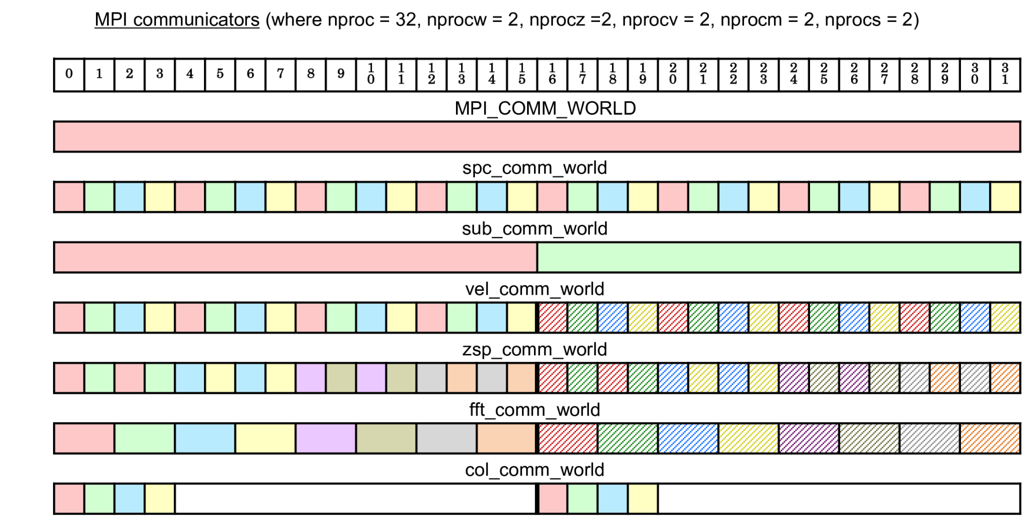

MPI ranks (where nproc = 32, nprocw = 2, nprocz =2, nprocv = 2, nprocm = 2, nprocs = 2)
rankg
ranks
rank
rankm
rankv
rankz
rankw
rankg: global rank in whole MPI processes
ranks: representing species
rank: local rank in each species
rankm: representing μ direction
rankv: representing v direction
rankz: representing z direction
rankw: representing ky direction

Figure 4.1: An example of MPI ranks in GKV.



MPI_COMM_WORLD: Communicate among whole MPI processes
spc_comm_world: Communicate among (rankv,rankm,ranks) with fixed (rankw,rankz).
[for velocity-space integration and summation over species]
sub_comm_world: Communicate in ranks.
vel_comm_world: Communicate among (rankv,rankm) with fixed (rankw,rankz), independent to ranks.
[for velocity-space integration in each species]
zsp_comm_world: Communicate among rankz with fixed (rankw,rankv,rankm), independent to ranks.
[for field-line-aligned integration in each species]
fft_comm_world: Communicate among rankw with fixed (rankz,rankv,rankm), independent to ranks.
[for data transpose of parallel 2D FFT]
col_comm_world: Communicate among ranks with vel_rank=0. [for field-particle operator in Sugama collision operator]

Figure 4.2: An example of MPI communicators in GKV.

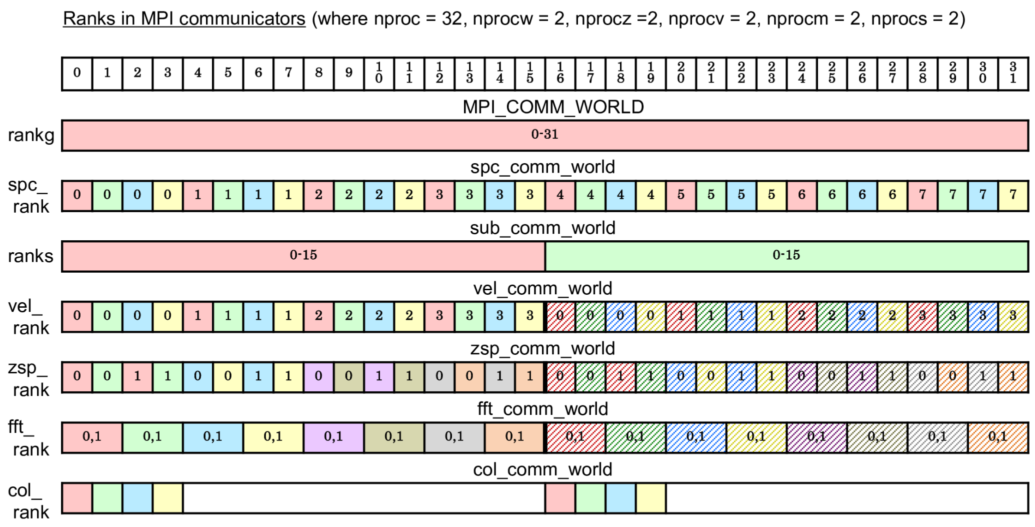


Figure 4.3: An example of MPI ranks in communicators in GKV.

Taking advantage of the multi-dimensional problem, multi-dimensional domain decomposition is applied for y, z, $v_\parallel$, μ and s, where 2D FFTs in x and y are parallelized by means of the transpose split method. Then, the required MPI communications are data transpose for the parallel 2D FFTs in x and y, point-to-point communications in z, $v_\parallel$ and μ for finite difference methods, and reduction communications over $v_\parallel$, μ and s for velocity-space and species integration. Figures 4.1-4.3 show schematic pictures of the multi-layer structure of the multi-dimensional domain decomposition, illustrating the case that y, z, $v_\parallel$, μ and s are respectively split by two MPI processes (and thus $2^5 = 32$ processes in total). Plasma species s are decomposed as $\texttt{ranks} = 0, 1$, and each species are hierarchically decomposed by the magnetic moment μ (`rankm`), the parallel velocity $v_\parallel$ (`rankv`), the parallel direction z (`rankz`), and the perpendicular direction x, y (`rankw`). Thus, data transpose in x and y is performed for different subranks of `rankw` by using `fft_comm_world` communicator, point-to-point communications in z (v or m) are performed between `rankz` (`rankv` or `rankm`), while reduction communications over v, μ and s are performed for fixed `rankz` and `rankw` by using `spc_comm_world` communicator.

4.4 Intra-node decomposition by using OpenMP

Intra-node decomposition is basically implemented by loop-level parallelization with OpenMP. Time-consuming MPI communications are masked by computation-communication overlap technique using MASTER thread. For more details, see Ref. [4-3].

4.5 References

[4-1] S. Gill, *Proc. Cambridge Philosophical Soc.* **47**, 96 (1951).
[4-2] S. Maeyama, T.-H. Watanabe, Y. Idomura, M. Nakata, and M. Nunami, *Comput. Phys. Commun.*, submitted.
[4-3] S. Maeyama, T.-H. Watanabe, Y. Idomura, M. Nakata, M. Nunami, and A. Ishizawa, *Parallel Comput.* **49**, 1 (2015).

5 Simulation

5.1 Structure of GKV

NOTE: This explanation is based on the GKV version gkvp_f0.48. When one expands the GKV package, there are

- `gkvp_f0.48/`
    - `README_for_namelist.txt`
    - `Version_memo.txt`
    - `src/`
        - `gkvp_f0.48_header.f90` (Module for setting grid resolutions and MPI processes)
        - `gkvp_f0.48_out.f90` (Module for data output)
        - ...
    - `lib/`
        - ... (contains libraries for random number and Bessel functions)
    - `extra_tools/`
        - `fig_stdout.tar.gz` (creates a PDF file for visualizing standard ASCII output)
        - `v29diag.tar.gz` (Post processing program for analyzing standard BINARY output)
        - ...
    - `run/`
        - `gkvp_f0.48_namelist` (Namelist for setting physical plasma parameters)
        - `sub.q` (Batch script, depending on machines)
        - `shoot` (Script for submitting jobs, depending on machines)
        - `Makefile` (depending on machines)
        - `backup/`
        - ...

5.2 Setting parameters

Code ***src/gkvp_f0.48_header.f90***

fortran

```fortran
...
!--------------------------------------
! Dimension size (grid numbers)
!--------------------------------------
! Global simulation domain
! in x, y,z,v,m (0:2*nxw-1, 0:2*nyw-1,-global_nz:global_nz-1,1:2*global_nv,0:global_nm)
! in kx,ky,z,v,m ( -nx:nx,0:global_ny,-global_nz:global_nz-1,1:2*global_nv,0:global_nm)
integer, parameter :: nxw = 128, nyw = 64
integer, parameter :: nx = 84, global_ny = 41 ! 2/3 de-aliasing rule
integer, parameter :: global_nz = 24, global_nv = 48, global_nm = 15
integer, parameter :: nzb = 3, &  ! the number of ghost grids in z
                       nvb = 3    ! the number of ghost grids in v and m
!--------------------------------------
! Data distribution for MPI
!--------------------------------------
integer, parameter :: nprocw = 2, nprocz = 2, nprocv = 4, nprocm = 2, nprocs = 2
...
```

Figure 5.1

Grid resolutions and MPI processes are set in src/gkvp_f0.48_header.f90.

- `nxw` | Grid number in x
- `nyw` | Grid number in y
- `nx` | Mode number in k_x (-nx:nx)
- `global_ny` | Mode number in k_y (0:global_ny)
- `global_nz` | Grid number in z (-global_nz:global_nz-1.)
- `global_nv` | Grid number in $v_\parallel$ (1:2global_nv.)
- `global_nm` | Grid number in μ (0:global_nm.)
- `nprocw` | MPI processes for k_y decomposition
- `nprocz` | MPI processes for z decomposition
- `nprocv` | MPI processes for $v_\parallel$ decomposition
- `nprocm` | MPI processes for μ decomposition
- `nprocs` | MPI processes for s decomposition

Note that (i) $\mathtt{nxw} > 3\mathtt{nx}/2$ and $\mathtt{nyw} > 3\mathtt{global\_ny}/2$ in nonlinear simulations; (ii) $\mathtt{global\_nz}/\mathtt{nprocz}$, $\mathtt{global\_nv}/\mathtt{nprocv}$, $(\mathtt{global\_nm}+1)/\mathtt{nprocm}$ should be integer; (iii) `nprocs` is same as the number of kinetic plasma species; (iv) $(\mathtt{global\_nm}+1)/\mathtt{nprocm} \geq 4$. `nzb` and `nvb` are the number of ghost grid in z and $v_\parallel/\mu$, whose required numbers depend on the employed finite difference methods. `nzb` = `nvb` =3 is enough by default. Plasma parameters are set in `run/gkvp_f0.48_namelist`. See Appendix A.1 in detail. GKV has MHD equilibrium interfaces, IGS for Tokamaks and BZX for Stellarators. See Appendix A.2 for the use of IGS and BZX.

5.3 Building

Prepare a proper `run/Makefile`. Some samples are found in `run/backup/`. Then,

csh

```
cd run
make
```

will create the load module `run/gkvp_mpifft.exe`.

5.4 Running

Prepare proper `run/sub.q` and `run/shoot`. Some samples are found in `run/backup/`.

Code ***run/sub.q for Plasma Simulator at NIFS***

csh

```
#!/bin/csh
#PJM -L "rscunit=fx"
#PJM -L "rscgrp=medium"
#PJM -L "node=32"
#PJM -L "elapse=24:00:00"
#PJM -j
#PJM -s
#PJM --mpi "proc=64"
#PJM -g 17000
setenv PARALLEL 16          # Thread number for automatic parallelization
setenv OMP_NUM_THREADS 16   # Thread number for Open MP
set DIR=%%DIR%%
set LDM=gkvp_mpifft.exe
set NL=gkvp_f0.48_namelist.%%%
### Run
cd ${DIR}
setenv fu05 ${DIR}/${NL}
module load fftw-fx/3.3.4
mpiexec ${DIR}/${LDM}
```

Figure 5.2

In the batch script `sub.q`, users specify the numbers of available computation nodes, MPI processes, and OpenMP threads. Required MPI process number of GKV is
$\mathtt{nproc} = \mathtt{nprocw} * \mathtt{nprocz} * \mathtt{nprocv} * \mathtt{nprocm} * \mathtt{nprocs}$. Usually,
$\mathtt{nproc} * \mathtt{OMP\_NUM\_THREADS} = \mathtt{nodes} * (\mathtt{cores\ per\ node})$ is a reasonable choice.

Code ***run/shoot***

csh

```
#!/bin/csh
#### Environment setting
set DIR=/data/lng/maeyama/gkv_training/test01
set LDM=gkvp_mpifft.exe
set NL=gkvp_f0.48_namelist
set SC=pjsub
set JS=sub.q
## For VMEC, set VMCDIR including metric_boozer.bin.dat
set VMCDIR=./input_vmec
## For IGS, set IGSDIR including METRIC_{axi,boz,ham}.OUT
set IGSDIR=./input_eqdsk
...
```

Figure 5.3

In the Script for submitting jobs, `shoot`, users set the output directory `DIR` where all output of GKV will be dumped. After finishing the all settings, type as follow to submit step jobs,

csh

```csh
cd run/
./shoot START_NUM END_NUM (JOB_ID)
```

For example, `./shoot 1 1` gives a single job (*.001). Similarly, `./shoot 2 2` gives a single job (*.002), which continues from the first run (*.001). Step job submission allows some sets of successive continuing jobs, e.g., `./shoot 3 5` gives step jobs (*.003 - *.005). Above three examples assume that the previous job has already finished. If a previous (*.005) job having `JOB_ID = 11223` is still in queue, `./shoot 6 7 11223` adds step jobs (*.006 - *.007) which sequentially follow after the end of previous job (*.005). Before running expensive nonlinear simulations, it is strongly recommended to test computational performance and its scalability: (i) Run a short-time run at the target problem size, (ii) Try some combination of $(\mathtt{nprocw}, \mathtt{nprocz}, \mathtt{nprocv}, \mathtt{nprocm}, \mathtt{nprocs}, \mathtt{OMP\_NUM\_THREADS})$ while keeping the number of $\mathtt{node} * (\mathtt{cores\ per\ node})$, (iii) Check scalablity of the computational performance against the number of computation nodes. Optimal setting may strongly depend on the target problem size.

Code *run/gkvp_f0.48_namelist*

fortran

```fortran
&cmemo memo="GKV-plus f0.48 developed for peta-scale computing", &end
&calct calc_type="nonlinear",
       z_bound="outflow",
       z_filt="off",
       z_calc="cf4",
       art_diff=1.d0,
       num_triad_diag=0, &end
&triad mxt = 0, myt = 0/
&equib equib_type = "analytic", &end
&run_n inum=%%%,
       ch_res = .false., &end
&files f_log="%%DIR%%/log/gkvp_f0.48.",
       f_hst="%%DIR%%/hst/gkvp_f0.48.",
       f_phi="%%DIR%%/phi/gkvp_f0.48.",
       f_fxv="%%DIR%%/fxv/gkvp_f0.48.",
       f_cnt="%%DIR%%/cnt/gkvp_f0.48.", &end
&runlm e_limit = 84600.d0, &end
&times tend = 200.d0,
       dtout_fxv = 1.d0,
       dtout_ptn = 0.1d0,
       dtout_eng = 0.01d0,
       dtout_dtc = 0.001d0, &end
&deltt dt_max = 0.001d0,
       adapt_dt = .true.,
       courant_num = 0.5d0,
       time_advnc = "auto_init", &end
&physp R0_Ln = 2.2d0, 2.2d0,
       R0_Lt = 6.9d0, 6.9d0,
       nu = 1.d0, 1.d0,
       Anum = 5.446d-4, 1.d0,
       Znum = 1.d0, 1.d0,
       fcs = 1.d0, 1.d0,
       sgn = -1.d0, 1.d0,
       tau = 1.d0, 1.d0,
       dns1 = 1.d-9, 1.d-9,
       tau_ad = 1.d0,
       lambda_i = 4.3d-4,
       beta = 0.4d-2,
       ibprime = 0,
       vmax = 4.d0,
       nx0 = 10000, &end
&nperi n_tht = 1,
       kymin = 0.05d0,
       m_j   = 4,
       del_c = 0.d0, &end
&confp eps_r    = 0.18d0,
       eps_rnew = 1.d0,
       q_0      = 1.4d0,
       s_hat    = 0.78d0,
...
&nu_ref Nref = 4.5d19,
        Lref = 1.7d0,
        Tref = 2.0d0,
        col_type = LB,
        iFLR = 1, icheck = 0, &end
```

Figure 5.4

6 Diagnostics

6.1 Output files of GKV

When finishing a run of GKV, all simulation output will be dumped in the output directory `DIR` set in the `run/shoot` script (for example in Figure 5.3, `DIR=/data/lng/maeyama/gkv_training/test01/`), as classified into the following directories,

- `DIR/`
  - `log/` (Log files on simulation runs)
  - `cnt/` (Binary data for restart)
  - `fxv/` (Binary data of distribution functions $\tilde{f}_{s\boldsymbol{k}}(k_x, k_y, v_\parallel, \mu)$ at several positions of z)
  - `phi/` (Binary data of potentials, fluid moments, and variables in the entropy balance equation)
  - `hst/` (Ascii data of the GKV standard output)
  - ... (Others are back up of the source code and environmental settings)

List of GKV output is summarized in Appendix A.1.

Users may diagnose these output data by themselves. The present version of GKV provides two post-processing tools, `fig_stdout` and `diag`, which are contained in `gkvp_f0.48/extra_tools/`.

6.2 PDF generating script for ASCII output: fig_stdout

Noting that `fig_stdout` requires `gnuplot` later than version 5.0. Expanding `gkvp_f0.48/extra_tools/fig_stdout.tar.gz` under the output directory `DIR`,

- `DIR/`
  - `fig_stdout/`
    - `make_pdf.csh` (Script for making a PDF sheet)
    - `src/` (Script for gnuplot)
    - `pdf/` (Directory to be store the created PDF sheet)
    - `eps/` (Directory to be store the plotted EPS files)
    - `data/` (Directory to be store raw data of GKV standard output)

and typing the following commands,

csh

```csh
cd fig_stdout/
./make_pdf.csh clean
./make_pdf.csh
```

users obtain a PDF sheet of the GKV standard output in `fig_stdout/pdf/`.

6.3 Post-processing program for BINARY output: diag

6.3.1 What is diag?

One difficulty of users may read GKV binary output which are decomposed by MPI. Post-processing program `diag` helps to read GKV binary output of a desired quantity at a desired time step. Since the read quantity is constructed as a global variable, e.g., $\tilde{\phi}_{\boldsymbol{k}}(\texttt{-nx:nx,0:global\_ny,-global\_nz:global\_nz-1})$ (not a local variable decomposed by MPI $\tilde{\phi}_{\boldsymbol{k}}(\texttt{-nx:nx,0:ny,-nz:nz-1})$), users do not need to be conscious of MPI parallelization of GKV. The main program `diag_main.f90` calls each diagnostics module `out_*****.f90`, which should be encapsulated so as to avoid interference and misuse. Reading GKV binary file is done by calling `diag_rb` module in each diagnostics module. Although there are some diagnostics modules implemented, users can design a new diagnostics module by themselves.

6.3.2 How to use diag

Expanding `gkvp_f0.48/extra_tools/v29diag.tar.gz`, one finds source codes of `diag`.

- `v29diag/`
  - `Makefile`
  - `go.diag` (Batch script)
  - `backup/`

- `plotfile/` (Sample file for gnuplot)
- `src`
  - `diag_header.f90` (Module for setting grid resolutions and MPI processes in GKV)
  - `diag_main.f90` (Main program calling each diagnostics module)
  - `diag_rb.f90` (Module for reading GKV binary output)
  - `diag_*****.f90` (Module for other settings)
  - ...
  - `out_*****.f90` (Module for each diagnostics)
  - ...

How to use `diag` is in the following steps:

1. Setting parameters in `v29diag/src/diag_header.f90`

Code ***v29diag/src/diag_header.f90***

fortran

```fortran
  ...
!%%% DIAG parameters %%%
  integer, parameter :: snum = 1      ! begining of simulation runs
  integer, parameter :: enum = 1      ! end of simulation runs
!%%%%%%%%%%%%%%%%        ! Set run numbers covering diagnosed time range.

!%%% GKV parameters %%%
  integer, parameter :: nxw = 2, nyw = 8
  integer, parameter :: nx = 0, global_ny = 5 ! 2/3 de-aliasing rule
  integer, parameter :: global_nz = 64, global_nv = 24, global_nm = 15
  integer, parameter :: nzb = 3, &  ! the number of ghost grids in z
                        nvb = 3    ! the number of ghost grids in v and m
  integer, parameter :: nprocw = 1, nprocz = 4, nprocv = 2, nprocm = 2, nprocs = 2
!%%%%%%%%%%%%%%%%%%%%%%%%        ! These should be same as gkvp_f0.48_header.f90.
  ...
```

Figure 6.1

2. Calling diagnostics modules in `v29diag/src/diag_main.f90`

3. Setting the output directory of GKV, `DIR`, in `go.diag`

4. Compile & Execution

5. Output data is dumped in `$DIR/post/`.

6.3.3 Examples of diag

Code ***Example from v29diag/src/diag_main.f90: outputs 2D electrostatic potential in the x–y plane at a given z, phi_tilde(x, y).***

fortran

```fortran
PROGRAM diag
  ...
  use out_mominxy, only : phiinxy    ! Use corresponding diagnostics module
  implicit none
  integer :: giz, loop
  ...
  giz  = 0       ! Set diagnosed grid in z (-global_nz <= giz <= global_nz-1)
  loop = 100     ! Set diagnosed time step (time = dtout_ptn * loop)
  call phiinxy(giz, loop)  ! Output phi_tilde(x,y) at giz=0, loop=100
  ...
END PROGRAM diag
```

Figure 6.2

Code ***Example from v29diag/src/diag_main.f90: outputs 1D electrostatic potential in the field-aligned z coordinate for a given mode (k_x, k_y), i.e., phi_tilde_k(z).***

fortran

```fortran
PROGRAM diag
  ...
  use out_mominz, only : phiinz    ! Use corresponding diagnostics module
  implicit none
  integer :: mx, gmy, loop
  ...
  mx   = 0       ! Set diagnosed radial mode number k_x (-nx <= mx <= nx-1)
  gmy  = 6       ! Set diagnosed bi-normal mode number k_y (0 <= gmy <= global_ny)
  loop = 100     ! Set diagnosed time step (time = dtout_ptn * loop)
  call phiinz(mx, gmy, loop)  ! Output phi_tilde_k(z) at mx=0, gmy=6, loop=100
  ...
END PROGRAM diag
```

Figure 6.3

For more details, Appendix A.4 explains how `diag_rb` module read GKV binary data, and Appendix A.5 shows some examples of diagnostics modules.

A Appendix

A.1 List of GKV namelist

List of run/gkvp_f0.48_namelist

Group	Name	Parameter

Group	Name	Parameter
&cmemo	memo	Memo
&calct	calc_type	"linear" — for linear runs
		"lin_freq" — for linear runs with frequency check $k_x = 0$
		"nonlinear" — for nonlinear runs
	z_bound	"zerofixed" — Fixed boundary in z
		"outflow" — Outflow boundary in z
		"mixed" — Outflow boundary in z only for $\tilde{f}_{\mathbf{sk}}$
	z_filt	"on" — Enable 4th-order filtering in z on $d\tilde{f}_{\mathbf{sk}}/dt$
		"off" — Disable filtering
	z_calc	"cf4" — 4th-order central finite difference for $d\tilde{f}_{\mathbf{sk}}/dz$ (nzb=2)
		"up5" — 5th-order upwind finite difference for $d\tilde{f}_{\mathbf{sk}}/dz$ (nzb=3)
	art_diff	Coefficient of artificial diffusion for z_calc="cf4"
	num_triad_diag	Number of triad transfer diagnostics, consistent with the number of lines "&triad mxt=*, myt=*".
&triad	mxt=*, myt=*	Diagnosed mode number of triad transfer analysis. Add lines of "&triad mxt=*, myt=*" as desired.
&equib	equib_type	"analytic" — Analytic helical field with the metrics in cylinder
		"s-alpha" — s-alpha model with alpha = 0 (cylindrical metrics)

Group	Name	Parameter
		"circ-MHD" — Concentric circular field with consistent metrics
		"vmec" — Tokamak/stellarator field from the VMEC code
		"eqdsk" — Tokamak field (MEUDAS/TOPICS or G-EQDSK) via IGS code
&run_n	inum	Current run number
	ch_res	.true. — Change perpendicular resolutions (editing gkvp_f0.48_set.f90 required)
		.false. — Disable changing resolution
&files	f_log	Data directory for log data
	f_hst	Data directory for time-series data
	f_phi	Data directory for field quantity data
	f_fxv	Data directory for distribution function data
	f_cnt	Data directory for continue data
&runlm	e_limit	Elapsed time limit [sec]
&times	tend	End of simulation time $[L_{\mathrm{ref}}/v_{\mathrm{ref}}]$
	dtout_fxv	Time spacing for data output $[L_{\mathrm{ref}}/v_{\mathrm{ref}}]$
	dtout_ptn	Time spacing for data output $[L_{\mathrm{ref}}/v_{\mathrm{ref}}]$
	dtout_eng	Time spacing for data output $[L_{\mathrm{ref}}/v_{\mathrm{ref}}]$

Group	Name	Parameter
	dtout_dtc	Time spacing for time-step-size adaption $[L_{\mathrm{ref}}/v_{\mathrm{ref}}]$
&deltt	dt_max	Maximum time step size $[L_{\mathrm{ref}}/v_{\mathrm{ref}}]$
	adapt_dt	.true. — Enable time-step-size adaption
		.false. — Time step size fixed to dt = dt_max
	courant_num	Courant number for time-step-size adaption
	time_advnc	"rkg4" — Explicit time integration by 4th-order Runge–Kutta–Gill
		"imp_colli" — 2nd-order operator split + 2nd-order implicit collision solver + 4th-order RKG (collisionless)
		"auto_init" — If collision restricts linear time step size, use "imp_colli"; otherwise "rkg4"
&physp	R0_Ln	Normalized density gradient, $L_{\mathrm{ref}}/L_{\mathrm{ne}}$, $L_{\mathrm{ref}}/L_{\mathrm{ni}}$, ...
	R0_Lt	Normalized temperature gradient, $L_{\mathrm{ref}}/L_{\mathrm{te}}$, $L_{\mathrm{ref}}/L_{\mathrm{ti}}$, ...
	nu	Bias factor for LB collision model (e.g., 1.d0, 0.5d0, 2.d0). *After gkvp_f0.40, collision frequencies are set by (Nref, Tref, Lref) in &nu_ref; nu is just a bias for LB and unused in multi-species (full).*
	Anum	Mass number, $m_{\mathrm{e}}/m_{\mathrm{ref}}$, $m_{\mathrm{i}}/m_{\mathrm{ref}}$, ...
	Znum	Atomic number, $|e_{\mathrm{e}}/e_{\mathrm{ref}}|$, $|e_{\mathrm{i}}/e_{\mathrm{ref}}|$, ...

Group	Name	Parameter
	fcs	Charge fraction, $\lvert e_\mathrm{e} n_\mathrm{e}/(e_\mathrm{ref} n_\mathrm{ref})\rvert$, $\lvert e_\mathrm{i} n_\mathrm{i}/(e_\mathrm{ref} n_\mathrm{ref})\rvert$, ... *Recommended: fcs = 1.0 for electrons (with n_ref = n_e).*
	sgn	Sign of charge, $e_\mathrm{e}/\lvert e_\mathrm{e}\rvert$, $e_\mathrm{i}/\lvert e_\mathrm{i}\rvert$, ...
	tau	Normalized temperature, $T_\mathrm{e}/T_\mathrm{ref}$, $T_\mathrm{i}/T_\mathrm{ref}$, ... *Recommended: $T_\mathrm{i}/T_\mathrm{ref} = 1.0$ for the first ion species.*
	dns1	Initial perturbation amplitude, $(L_\mathrm{ref}/\rho_\mathrm{ref})\,\tilde{n}_\mathrm{e}/n_\mathrm{ref}$, $(L_\mathrm{ref}/\rho_\mathrm{ref})\,\tilde{n}_\mathrm{i}/n_\mathrm{ref}$, ...
	tau_ad	$T_\mathrm{i}/T_\mathrm{e}$ for single-species ITG-ae (sgn=+1), $T_\mathrm{e}/T_\mathrm{i}$ for single-species ETG-ai (sgn=-1)
	lambda_i	Ratio $(\mathrm{Debye\ length}/\rho_\mathrm{ref})^2$ $= \varepsilon_0 B_\mathrm{ref}^2/(m_\mathrm{ref} n_\mathrm{ref})$
	beta	Local beta: $\mu_0 n_\mathrm{ref} T_\mathrm{ref}/B_\mathrm{ref}^2$
	ibprime	"1" — Enable grad-p (finite beta-prime) contribution on magnetic drift *kvd* for equib_type="eqdsk" and "vmec"
		"0" — Ignore it
	vmax	Velocity domain size in units of each thermal speed $[v_\mathrm{ts}]$
	nx0	Radial mode number for initial perturbation. *If nx0 > nx, it is reset to nx.*
&nperi	n_tht	Length of fluxtube, z-domain = $\pm N_\mathrm{tht}\pi$
	kymin	Minimum field-line-label (poloidal) wave number $[1/\rho_\mathrm{ref}]$

Group	Name	Parameter
	m_j	Mode connection number for pseudo-periodic boundary; $k_{x,\mathrm{min}} = \lvert 2\pi s_{\hat{}} kymin/m_j \rvert$
	del_c	Mode connection phase in fluxtube model (standard: del_c = 0.d0)
&confp	eps_r	Inverse aspect ratio at center of fluxtube, $a\,\rho_0/L_{\mathrm{ref}}$
	eps_rnew	Model factor for equib_type="analytic"
	q_0	Safety factor at center of fluxtube, $q(\rho_0)$
	s_hat	Magnetic shear at center of fluxtube, $s(\rho_0)$
	lprd	
	⋮	Model factor for equib_type="analytic"
	malpha	
&vmecp	s_input	Reference radial flux surface ρ_0 in Stellarator (VMEC) equilibrium?
	nss	Number of radial grids on METRIC data?
	ntheta	ntheta = (number of poloidal grids on METRIC) + 1 = 2*global_nz + 1?
	nzeta	Number of toroidal grids on METRIC data?
&bozxf	f_bozx	File location of METRIC data produced by BZX code
&igsp	s_input	Reference radial flux surface ρ_0 in Tokamak (MEUDAS/TOPICS or G-EQDSK) equilibrium

Group	Name	Parameter
	mc_type	"0" — Axisymmetric coordinates
		"1" — Boozer coordinates
		"2" — Hamada coordinates
	q_type	"1" — Use consistent q-value on g-eqdsk equilibrium (**Recommended**)
		"0" — Use inconsistent but given q_0 value in &confp
	nss	Number of radial grids on METRIC data
	ntheta	ntheta = (number of poloidal grids on METRIC) + 1 = 2*global_nz + 1
&igsf	f_igs	File location of METRIC data produced by IGS code
&nu_ref	Nref	Local electron density at center of fluxtube, $n_\mathrm{e}(\rho_0)$ $[\mathrm{m}^{-3}]$
	Lref	Major radius at magnetic axis, R_a $[\mathrm{m}]$
	Tref	Main ion temperature at center of fluxtube, $T_\mathrm{i}(\rho_0)$ $[\mathrm{keV}]$
	col_type	"LB" — Lenard–Bernstein model collision operator
		"lorentz" — Lorentz model collision operator
		"full" — Sugama model collision operator for multiple plasma species
	iFLR	"1" — Enable FLR (gyrophase-averaging & classical diffusion) terms (LB & full)

Group	Name	Parameter
		"0" — Disable it (DK-limit)
	icheck	"0" — For production runs
		"1" — Debug test with Maxwellian Annihilation (use with iFLR = 0)

Note that `inum=%%%` and `f_**="%%DIR%%/..."` will be automatically set by the `shoot` script. In the `&physp` group, species-dependent names `R0_Ln – dns1` are the array of length `nprocs`. The `&vmecp` and `&bozxf` groups are active only when `equib_type = "vmec"`. Similarly, the `&igsp` and `&igsf` groups are active only when `equib_type = "eqdsk"`.

A.2 Use of MHD equilibrium interfaces

In preparation.

A.2.1 Use of IGS (EQDSK for Tokamaks)

In preparation.

A.2.2 Use of BZX (VMEC for Stellarators)

In preparation.

A.3 List of GKV output

GKV output files are:

- The output directory `DIR/`
  - `cnt/*cnt*`
  - `fxv/*fxv*`
  - `phi/*phi*, *Al*, *mom*, *trn*, (*tri* for nonlinear runs)`
  - `hst/*bln*, *geq*, *gem*, *qes*, *qem*, *wes*, *wem*, *eng*, *men*, *dtc*, *mtr*, (*frq*, *dsp* for linear runs)`
  - `log/*log*`

Their explanations are summarized below.

Explanations on GKV output files

cnt/gkvp_f0.48.(rankg in 6 digits).cnt.(inum in 3 digits)

Field	Description
File type	`Binary`
Output timing	`End of the run`
MPI ranks	`All`
Total files	`nprocw*nprocz*nprocv*nprocm*nprocs*(Total run numbers)`
GKV unit	`ocnt`
Stored data	`time, ff` **time** (real*8): Simulation time t $[L_{\mathrm{ref}}/v_{\mathrm{ref}}]$ ***ff(-nx:nx,0:ny,-nz:nz-1,1:2nv,0:nm)*** (complex*8): Perturbed distribution function $\tilde{f}_{\mathrm{s}\mathbf{k}}$.

fxv/gkvp_f0.48.(rankg in 6 digits).(ranks in 1 digit).fxv.(inum in 3 digits)

Field	Description
File type	`Binary`
Output timing	`dtout_fxv`
MPI ranks	`All`
Total files	`nprocw*nprocz*nprocv*nprocm*nprocs*(Total run numbers)`
GKV unit	`ofxv`
Stored data	`time, ff` **time** (real*8): Simulation time t $[L_{\mathrm{ref}}/v_{\mathrm{ref}}]$ ***ff(-nx:nx,0:ny,1:2nv,0:nm)*** (complex*8): Perturbed distribution function $\tilde{f}_{\mathrm{s}\mathbf{k}}$ at `iz = -nz` in each `rankz`.

phi/gkvp_f0.48.(rankg in 6 digits).0.phi.(inum in 3 digits)

Field	Description
File type	`Binary`

Field	Description
Output timing	`dtout_ptn`
MPI ranks	`ranks == 0 .and. vel_rank == 0`
Total files	`nprocw*nprocz*(Total run numbers)`
GKV unit	`ophi`
Stored data	`time, phi` **time** (real*8*)*: Simulation time* t $[L_{\mathrm{ref}}/v_{\mathrm{ref}}]$ ***phi(-nx:nx,0:ny,-nz:nz-1)*** (*complex*8): Perturbed electrostatic potential $\tilde{\phi}_{\mathbf{k}}$.

phi/gkvp_f0.48.(rankg in 6 digits).0.Al.(inum in 3 digits)

Field	Description
File type	`Binary`
Output timing	`dtout_ptn`
MPI ranks	`ranks == 0 .and. vel_rank == 0`
Total files	`nprocw*nprocz*(Total run numbers)`
GKV unit	`oAl`
Stored data	`time, Al` **time** (real*8*)*: Simulation time* t $[L_{\mathrm{ref}}/v_{\mathrm{ref}}]$ ***Al(-nx:nx,0:ny,-nz:nz-1)*** (*complex*8): Perturbed vector potential $\tilde{A}_{\parallel\mathbf{k}}$.

phi/gkvp_f0.48.(rankg in 6 digits).(ranks in 1 digit).mom.(inum in 3 digits)

Field	Description
File type	`Binary`
Output timing	`dtout_ptn`
MPI ranks	`vel_rank == 0`

Field	Description
Total files	`nprocw*nprocz*nprocs*(Total run numbers)`
GKV unit	`omom`
Stored data	`time, mom` **time** (real8): *Simulation time* t $[L_{\mathrm{ref}}/v_{\mathrm{ref}}]$ ***mom(-nx:nx,0:ny,-nz:nz-1,0:nmom-1)*** (*complex*8): Perturbed fluid moments. In the present version `nmom = 6`: • $\tilde{n}_{\mathbf{sk}} = \int dv^3 \, J_{0\mathbf{sk}} \, \tilde{f}_{\mathbf{sk}}$ • $\tilde{u}_{\parallel\mathbf{sk}} = \int dv^3 \, v_\parallel \, J_{0\mathbf{sk}} \, \tilde{f}_{\mathbf{sk}}$ • $\tilde{p}_{\parallel\mathbf{sk}} = \int dv^3 \, \frac{m_s v_\parallel^2}{2} \, J_{0\mathbf{sk}} \, \tilde{f}_{\mathbf{sk}}$ • $\tilde{p}_{\perp\mathbf{sk}} = \int dv^3 \, \mu B \, J_{0\mathbf{sk}} \, \tilde{f}_{\mathbf{sk}}$ • $\tilde{q}_{\parallel\parallel\mathbf{sk}} = \int dv^3 \, v_\parallel \frac{m_s v_\parallel^2}{2} \, J_{0\mathbf{sk}} \, \tilde{f}_{\mathbf{sk}}$ • $\tilde{q}_{\parallel\perp\mathbf{sk}} = \int dv^3 \, v_\parallel \mu B \, J_{0\mathbf{sk}} \, \tilde{f}_{\mathbf{sk}}$ Normalized by $\delta_{\mathrm{ref}} n_{\mathrm{ref}}$, $\delta_{\mathrm{ref}} n_{\mathrm{ref}} v_{\mathrm{ref}}$, $\delta_{\mathrm{ref}} n_{\mathrm{ref}} T_{\mathrm{ref}}$, $\delta_{\mathrm{ref}} n_{\mathrm{ref}} T_{\mathrm{ref}}$, $\delta_{\mathrm{ref}} n_{\mathrm{ref}} T_{\mathrm{ref}} v_{\mathrm{ref}}$, $\delta_{\mathrm{ref}} n_{\mathrm{ref}} T_{\mathrm{ref}} v_{\mathrm{ref}}$, respectively.

phi/gkvp_f0.48.(rankg in 6 digits).(ranks in 1 digit).trn.(inum in 3 digits)

Field	Description
File type	`Binary`
Output timing	`dtout_eng`
MPI ranks	`zsp_rank == 0 .and. vel_rank == 0`
Total files	`nprocw*nprocs*(Total run numbers)`
GKV unit	`otrn`
Stored data	`time, S_{s k}, W_{E k}, W_{M k}, R_{sE k}, R_{sM k}, I_{sE k}, I_{sM k}, D_{s k}, Γ_{sE k}, Γ_{sM k}, Q_{sE k}, Q_{sM k}` **time** (real8): *Simulation time* t $[L_{\mathrm{ref}}/v_{\mathrm{ref}}]$ • $S_{\mathbf{sk}}(-nx : nx, 0 : ny)$: *Perturbed gyrocenter entropy* $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}}]$ (*real*8)

Field	Description
	<ul><li>$W_{\mathrm{E}\mathbf{k}}(-nx:nx,0:ny)$: Electrostatic field energy including polarization $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}}]$ (real*8)</li><li>$W_{\mathrm{M}\mathbf{k}}(-nx:nx,0:ny)$: *Magnetic field energy* $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}}]$ *(real*8)</li><li>$R_{\mathrm{sE}\mathbf{k}}(-nx:nx,0:ny)$: Wave−particle interaction $(W_{\mathrm{E}\mathbf{k}} \to S_{\mathrm{s}\mathbf{k}})$ $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}} v_{\mathrm{ref}}/L_{\mathrm{ref}}]$ (real*8*)</li><li>$R_{\mathrm{sM}\mathbf{k}}(-nx:nx,0:ny)$: *Wave−particle interaction* $(W_{\mathrm{M}\mathbf{k}} \to S_{\mathrm{s}\mathbf{k}})$ $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}} v_{\mathrm{ref}}/L_{\mathrm{ref}}]$ *(real*8)</li><li>$I_{\mathrm{sE}\mathbf{k}}(-nx:nx,0:ny)$: Nonlinear entropy transfer by $\mathbf{E}\times\mathbf{B}$ flow $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}} v_{\mathrm{ref}}/L_{\mathrm{ref}}]$ (real*8*)</li><li>$I_{\mathrm{sM}\mathbf{k}}(-nx:nx,0:ny)$: *Nonlinear entropy transfer by magnetic flutter* $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}} v_{\mathrm{ref}}/L_{\mathrm{ref}}]$ *(real*8)</li><li>$D_{\mathrm{s}\mathbf{k}}(-nx:nx,0:ny)$: Collisional dissipation $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}} v_{\mathrm{ref}}/L_{\mathrm{ref}}]$ (real*8*)</li><li>$\Gamma_{\mathrm{sE}\mathbf{k}}(-nx:nx,0:ny)$: *Particle flux by* $\mathbf{E}\times\mathbf{B}$ *flow* $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} v_{\mathrm{ref}}]$ *(real*8)</li><li>$\Gamma_{\mathrm{sM}\mathbf{k}}(-nx:nx,0:ny)$: Particle flux by magnetic flutter $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} v_{\mathrm{ref}}]$ (real*8*)</li><li>$Q_{\mathrm{sE}\mathbf{k}}(-nx:nx,0:ny)$: *Energy flux by* $\mathbf{E}\times\mathbf{B}$ *flow* $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}} v_{\mathrm{ref}}]$ *(real*8)</li><li>$Q_{\mathrm{sM}\mathbf{k}}(-nx:nx,0:ny)$: Energy flux by magnetic flutter $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}} v_{\mathrm{ref}}]$ (real*8)</li></ul>*See also Appendix B.1.*

phi/gkvp_f0.48.s(ranks in 1 digits)mx(mxt in 4 digits)my(myt in 4 digits).tri.(inum in 3 digits)

Field	Description
File type	`Binary`
Output timing	`dtout_ptn` (when `calc_type == "nonlinear"` and `num_triad_diag > 0`)
MPI ranks	`rank == 0`
Total files	`nprocs*num_triad_diag*(Total run numbers)`
GKV unit	`otri`

Field	Description
Stored data	`time, J_{sE k}^{p,q}, J_{sE p}^{q,k}, J_{sE q}^{k,p}, J_{sM k}^{p,q}, J_{sM p}^{q,k}, J_{sM q}^{k,p}` **time** (real8): *Simulation time* t $[L_\mathrm{ref}/v_\mathrm{ref}]$ • $J_{\mathrm{sE}\mathbf{k}}^{\mathbf{p},\mathbf{q}}(-nx:nx,-global_y:global_y)$: *Triad transfer function from modes* $\mathbf{p},\mathbf{q}$ *to mode* $\mathbf{k}$ *via* $\mathbf{E}\times\mathbf{B}$ *nonlinearity* $[\delta_\mathrm{ref}^2 n_\mathrm{ref} T_\mathrm{ref} v_\mathrm{ref}/L_\mathrm{ref}]$ (real8) • $J_{\mathrm{sE}\mathbf{p}}^{\mathbf{q},\mathbf{k}}(-nx:nx,-global_y:global_y)$: Cyclic change $(\mathbf{k},\mathbf{p},\mathbf{q})\to(\mathbf{p},\mathbf{q},\mathbf{k})$ (real8) • $J_{\mathrm{sE}\mathbf{q}}^{\mathbf{k},\mathbf{p}}(-nx:nx,-global_y:global_y)$: *Cyclic change* $(\mathbf{p},\mathbf{q},\mathbf{k})\to(\mathbf{q},\mathbf{k},\mathbf{p})$ (real8) • $J_{\mathrm{sM}\mathbf{k}}^{\mathbf{p},\mathbf{q}}(-nx:nx,-global_y:global_y)$: Triad transfer function from modes $\mathbf{p},\mathbf{q}$ to mode $\mathbf{k}$ via magnetic-flutter nonlinearity $[\delta_\mathrm{ref}^2 n_\mathrm{ref} T_\mathrm{ref} v_\mathrm{ref}/L_\mathrm{ref}]$ (real8) • $J_{\mathrm{sM}\mathbf{p}}^{\mathbf{q},\mathbf{k}}(-nx:nx,-global_y:global_y)$: *Cyclic change* $(\mathbf{k},\mathbf{p},\mathbf{q})\to(\mathbf{p},\mathbf{q},\mathbf{k})$ (real8) • $J_{\mathrm{sM}\mathbf{q}}^{\mathbf{k},\mathbf{p}}(-nx:nx,-global_y:global_y)$: Cyclic change $(\mathbf{p},\mathbf{q},\mathbf{k})\to(\mathbf{q},\mathbf{k},\mathbf{p})$ (real*8) Diagnosed for a fixed mode $\mathbf{k}=(\mathtt{mxt},\mathtt{myt})$ and plotted as a 2D function of $\mathbf{p}=(p_x,p_y)$, where the triad condition determines $\mathbf{q}=-\mathbf{k}-\mathbf{p}$. *See also Appendix B.2*.

hst/gkvp_f0.48.bln.(ranks in 1 digits).(inum in 3 digits)

Field	Description
File type	`Ascii`
Output timing	`dtout_eng`
MPI ranks	`rank == 0`
Total files	`nprocs*(Total run numbers)`
GKV unit	`obln`
Stored data	`time, S_s, W_E, W_M, R_sE, R_sM, I_sE, I_sM, D_s, (T_s Γ_sE)/L_ps, (T_s Γ_sM)/L_ps, Θ_sE/L_Ts, Θ_sM/L_Ts`

Field	Description
	time (real8): Simulation time t $[L_{\mathrm{ref}}/v_{\mathrm{ref}}]$ • $S_{\mathrm{s}}(0:1)$: Perturbed gyrocenter entropy $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}}]$ (real8) • $W_{\mathrm{E}}(0:1)$: Electrostatic field energy including polarization $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}}]$ (real8) • $W_{\mathrm{M}}(0:1)$: Magnetic field energy $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}}]$ (real8) • $R_{\mathrm{sE}}(0:1)$: Wave–particle interaction from $W_{\mathrm{E}\mathbf{k}}$ to $S_{\mathrm{s}\mathbf{k}}$ $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}} v_{\mathrm{ref}}/L_{\mathrm{ref}}]$ (real8) • $R_{\mathrm{sM}}(0:1)$: Wave–particle interaction from $W_{\mathrm{M}\mathbf{k}}$ to $S_{\mathrm{s}\mathbf{k}}$ $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}} v_{\mathrm{ref}}/L_{\mathrm{ref}}]$ (real8) • $I_{\mathrm{sE}}(0:1)$: Nonlinear entropy transfer by $\mathbf{E}\times\mathbf{B}$ flow $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}} v_{\mathrm{ref}}/L_{\mathrm{ref}}]$ (real8) • $I_{\mathrm{sM}}(0:1)$: Nonlinear entropy transfer by magnetic flutter $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}} v_{\mathrm{ref}}/L_{\mathrm{ref}}]$ (real8) • $D_{\mathrm{s}}(0:1)$: Collisional dissipation $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}} v_{\mathrm{ref}}/L_{\mathrm{ref}}]$ (real8) • $\frac{T_{\mathrm{s}}\Gamma_{\mathrm{sE}}}{L_{ps}}$: Particle flux term by $\mathbf{E}\times\mathbf{B}$ flow $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}} v_{\mathrm{ref}}/L_{\mathrm{ref}}]$ (real8) • $\frac{T_{\mathrm{s}}\Gamma_{\mathrm{sM}}}{L_{ps}}$: Particle flux term by magnetic flutter $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}} v_{\mathrm{ref}}/L_{\mathrm{ref}}]$ (real8) • $\frac{\Theta_{\mathrm{sE}}}{L_{T\mathrm{s}}}$: Heat flux term by $\mathbf{E}\times\mathbf{B}$ flow $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}} v_{\mathrm{ref}}/L_{\mathrm{ref}}]$ (real8) • $\frac{\Theta_{\mathrm{sM}}}{L_{T\mathrm{s}}}$: Heat flux term by magnetic flutter $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}} v_{\mathrm{ref}}/L_{\mathrm{ref}}]$ (real*8) The 0th and 1st components of S_{s}–D_{s} correspond to non-zonal $(k_y \neq 0)$ and zonal $(k_y = 0)$ fluctuations, respectively.

hst/gkvp_f0.48.ges.(ranks in 1 digits).(inum in 3 digits)

Field	Description
File type	`Ascii`
Output timing	`dtout_eng`
MPI ranks	`rank == 0`
Total files	`nprocs*(Total run numbers)`

Field	Description
GKV unit	`oges`
Stored data	`time, Γ_sE, Γ_{sE k_y}` **time** (real): Simulation time t $[L_{\rm ref}/v_{\rm ref}]$ $\Gamma_{\rm sE}$: Total particle flux by $\mathbf{E}\times\mathbf{B}$ flow $[\delta_{\rm ref}^2 n_{\rm ref} v_{\rm ref}]$ (real) $\Gamma_{{\rm sE}k_y}(0 : global\_ny)$: k_y spectrum of the particle flux by $\mathbf{E}\times\mathbf{B}$ flow (same units) (real).

hst/gkvp_f0.48.gem.(ranks in 1 digits).(inum in 3 digits)

Field	Description
File type	`Ascii`
Output timing	`dtout_eng`
MPI ranks	`rank == 0`
Total files	`nprocs*(Total run numbers)`
GKV unit	`ogem`
Stored data	`time, Γ_sM, Γ_{sM k_y}` **time** (real): Simulation time t $[L_{\rm ref}/v_{\rm ref}]$ $\Gamma_{\rm sM}$: Total particle flux by magnetic flutter $[\delta_{\rm ref}^2 n_{\rm ref} v_{\rm ref}]$ (real) $\Gamma_{{\rm sM}k_y}(0 : global\_ny)$: k_y spectrum of the particle flux by magnetic flutter (same units) (real).

hst/gkvp_f0.48.qes.(ranks in 1 digits).(inum in 3 digits)

Field	Description
File type	`Ascii`
Output timing	`dtout_eng`
MPI ranks	`rank == 0`
Total files	`nprocs*(Total run numbers)`

Field	Description
GKV unit	`oqes`
Stored data	`time, Q_sE, Q_{sE k_y}` **time** (real): Simulation time t $[L_{\mathrm{ref}}/v_{\mathrm{ref}}]$ Q_{sE}: Total energy flux by $\mathbf{E}\times\mathbf{B}$ flow $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}} v_{\mathrm{ref}}]$ (real) $Q_{\mathrm{sE}k_y}(0:global\_ny)$: k_y spectrum of the energy flux by $\mathbf{E}\times\mathbf{B}$ flow (same units) (real).

hst/gkvp_f0.48.qem.(ranks in 1 digits).(inum in 3 digits)

Field	Description
File type	`Ascii`
Output timing	`dtout_eng`
MPI ranks	`rank == 0`
Total files	`nprocs*(Total run numbers)`
GKV unit	`oqem`
Stored data	`time, Q_sM, Q_{sM k_y}` **time** (real): Simulation time t $[L_{\mathrm{ref}}/v_{\mathrm{ref}}]$ Q_{sM}: Total energy flux by magnetic flutter $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}} v_{\mathrm{ref}}]$ (real) $Q_{\mathrm{sM}k_y}(0:global\_ny)$: k_y spectrum of the energy flux by magnetic flutter (same units) (real).

hst/gkvp_f0.48.wes.(inum in 3 digits)

Field	Description
File type	`Ascii`
Output timing	`dtout_eng`
MPI ranks	`rankg == 0`
Total files	`(Total run numbers)`

Field	Description
GKV unit	`owes`
Stored data	`time, W_E, W_{E k_y}` **time** (real): Simulation time t $[L_{\mathrm{ref}}/v_{\mathrm{ref}}]$ W_{E}: Total electrostatic field energy including polarization $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}}]$ (real) $W_{\mathrm{E}k_y}(0 : global\_ny)$: k_y spectrum of the electrostatic field energy (same units) (real).

hst/gkvp_f0.48.wem.(inum in 3 digits)

Field	Description
File type	`Ascii`
Output timing	`dtout_eng`
MPI ranks	`rankg == 0`
Total files	`(Total run numbers)`
GKV unit	`owem`
Stored data	`time, W_M, W_{M k_y}` **time** (real): Simulation time t $[L_{\mathrm{ref}}/v_{\mathrm{ref}}]$ W_{M}: Total magnetic field energy $[\delta_{\mathrm{ref}}^2 n_{\mathrm{ref}} T_{\mathrm{ref}}]$ (real) $W_{\mathrm{M}k_y}(0 : global\_ny)$: k_y spectrum of the magnetic field energy (same units) (real).

hst/gkvp_f0.48.eng.(inum in 3 digits)

Field	Description
File type	`Ascii`
Output timing	`dtout_eng`
MPI ranks	`rankg == 0`
Total files	`(Total run numbers)`

Field	Description
GKV unit	`oeng`
Stored data	`time, sum_{kx,ky} <`

hst/gkvp_f0.48.men.(inum in 3 digits)

Field	Description
File type	`Ascii`
Output timing	`dtout_eng`
MPI ranks	`rankg == 0`
Total files	`(Total run numbers)`
GKV unit	`omen`
Stored data	`time, sum_{kx,ky} <`

hst/gkvp_f0.48.dtc.(inum in 3 digits)

Field	Description
File type	`Ascii`
Output timing	`dtout_dtc`
MPI ranks	`rankg == 0`
Total files	`(Total run numbers)`
GKV unit	`odtc`
Stored data	`time, dt, dt_limit, dt_nl` **time** (real): Simulation time t $[L_\mathrm{ref}/v_\mathrm{ref}]$ **dt** (real): Time step size $[L_\mathrm{ref}/v_\mathrm{ref}]$ **dt_limit** (real): Estimate of timestep limit (same units) **dt_nl** (real): Estimate of timestep limit from nonlinear advection (same units).

hst/gkvp_f0.48.mtr.(inum in 3 digits)

Field	Description
File type	`Ascii`
Output timing	`Beginning of the run`
MPI ranks	`rankg == 0`
Total files	`(Total run numbers)`
GKV unit	`omtr`
Stored data	`time, θ (or φ), B, dB/dx, dB/dy, dB/dz, g^{xx}, g^{xy}, g^{xz}, g^{yy}, g^{yz}, g^{zz}, sqrt(g)` **time** (real): Simulation time t $[L_{\mathrm{ref}}/v_{\mathrm{ref}}]$ θ: Poloidal angle (or toroidal angle φ when `equib_type = "vmec"`) (real) B: Magnetic field strength $[B_{\mathrm{ref}}]$ (real) $\partial B/\partial x, \partial B/\partial y, \partial B/\partial z$: Derivatives of B $[B_{\mathrm{ref}}/L_{\mathrm{ref}}]$ (real) Metric tensor components: $g^{xx}, g^{xy}, g^{xz}, g^{yy}, g^{yz}, g^{zz}$ with units as in the text (real) $\sqrt{g}$: Jacobian $[L_{\mathrm{ref}}]$ (real).

hst/gkvp_f0.48.frq.(inum in 3 digits)

Field	Description
File type	`Ascii`
Output timing	`dtout_eng` (when `calc_type == "linear"` or `"lin_freq"`)
MPI ranks	`rankg == 0`
Total files	`(Total run numbers)`
GKV unit	`ofrq`
Stored data	`time, omega` **time** (real): Simulation time t $[L_{\mathrm{ref}}/v_{\mathrm{ref}}]$ **omega(1:global_ny)**: k_y spectrum of complex linear

Field	Description
	frequency $\omega = \omega_\mathrm{r} + i\gamma\,[v_\mathrm{ref}/L_\mathrm{ref}]$ (real, real). Estimated for $k_x = 0$ from $\omega = \dfrac{\ln\tilde{\phi}_\mathbf{k}(t+\Delta t) - \ln\tilde{\phi}_\mathbf{k}(t)}{-\,i\,\Delta t}$ assuming $\tilde{\phi}_\mathbf{k}(t) \propto e^{-i\omega t}$.

hst/gkvp_f0.48.dsp.(inum in 3 digits)

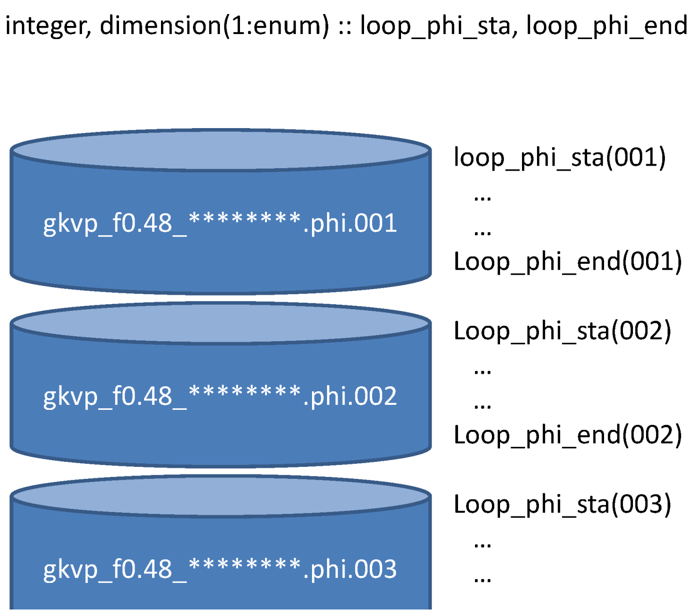
Field	Description
File type	`Ascii`
Output timing	`End of the run` (when `calc_type == "linear"` or `"lin_freq"`)
MPI ranks	`rankg == 0`
Total files	`(Total run numbers)`
GKV unit	`odsp`
Stored data	`ky, omega, diff, 1-ineq` **ky** (real): Field-line-label (poloidal) wavenumber k_y $[\rho_\mathrm{ref}^{-1}]$ **omega** (real, real): Complex linear frequency ω $[v_\mathrm{ref}/L_\mathrm{ref}]$ **diff** (real): Relative residual $\dfrac{\omega(t)-\omega(t-\Delta t)}{\omega(t)}$ **1-ineq** (real): Convergence check based on Schwarz inequality. At the end of the run, estimated complex frequencies for $k_x = 0$ are dumped; unconverged modes may be commented out.

log/gkvp_f0.48.(rankg in 6 digits).(ranks in 1 digit).log.(inum in 3 digits)

Field	Description
File type	`Ascii`
Output timing	`As needed`
MPI ranks	`All`

Field	Description
Total files	`nprocw*nprocz*nprocv*nprocm*nprocs*(Total run numbers)`
GKV unit	`olog`
Stored data	`Simulation log`

A.4 Data-reading module diag_rb in the post-processing program diag


integer, dimension(1:enum) :: loop_phi_sta, loop_phi_end
gkvp_f0.48_********.phi.001
loop_phi_sta(001)
…
…
Loop_phi_end(001)
gkvp_f0.48_********.phi.002
Loop_phi_sta(002)
…
…
Loop_phi_end(002)
gkvp_f0.48_********.phi.003
Loop_phi_sta(003)
…
…
Figure A.1: Output record number in the post-processing program diag

To read GKV binary output in the post-processing program `diag`, use the data-reading module `diag_rb`.

Code ***An example to use diag_rb***

fortran

```
use diag_rb, only : rb_phi_loop
complex(kind=DP) :: phi(-nx:nx, 0:global_ny, -global_nz:global_nz-1)
integer :: loop = 100
call rb_phi_loop(loop, phi) ! Read potential phi at output record loop=100
(time=dtout_ptn*loop)
```

Figure A.2

The output record number `loop` is counted up from the first run (`inum`=1) by evaluating file size of GKV binary output. As shown in Fig. 1.1{reference-type="ref" reference="fig:output_record_number"}, output record number for the binary output `$DIR/phi/*phi*` is from `loop_phi_sta`(001) = 0 to `loop_phi_end(enum)` = `nloop_phi`.

Therefore, even if you analyze only run numbers from `snum` > 1 to `enum`, all GKV binary output data from `inum` =1 should be left in the diagnosed directory.

Taking a look at the source code of `diag_rb`, one finds various types of subroutines which read electrostatic potential $\tilde{\phi}_{\boldsymbol{k}}$ in (k_x, k_y, z) or in (k_x, k_y) at a given z or in (z) for a given mode k_x, k_y, etc., and similarly read magnetic vector potential $\tilde{A}_{\parallel \boldsymbol{k}}$, fluid moments, and so on. Some typical subroutines are listed below. One may find more efficient subroutine in the source code of `diag_rb`.

List of subroutines in the data-reading module `diag_rb`

rb_phi_gettime(loop, time)

Field	Description
Arguments	• `integer, intent(in) :: loop` • `real(kind=DP), intent(out) :: time`
GKV binary output	`phi/gkvp_f0.48_(rankg in 6 digits).0.phi.(inum in 3 digits)`
Description	Read simulation time `time` corresponding to the output record `loop`. $(\mathrm{time} \simeq \mathrm{dtout\_ptn} \times \mathrm{loop})$

rb_Al_gettime(loop, time)

Field	Description
Arguments	• `integer, intent(in) :: loop` • `real(kind=DP), intent(out) :: time`
GKV binary output	`phi/gkvp_f0.48_(rankg in 6 digits).0.Al.(inum in 3 digits)`
Description	Read simulation time `time` corresponding to the output record `loop`. $(\mathrm{time} \simeq \mathrm{dtout\_ptn} \times \mathrm{loop})$

rb_mom_gettime(loop, time)

Field	Description
Arguments	• `integer, intent(in) :: loop` • `real(kind=DP), intent(out) :: time`

Field	Description
GKV binary output	`phi/gkvp_f0.48_(rankg in 6 digits).(ranks in 1 digit).mom.(inum in 3 digits)`
Description	Read simulation time `time` corresponding to the output record `loop`. $(\mathrm{time} \simeq \mathrm{dtout\_ptn} \times \mathrm{loop})$

rb_trn_gettime(loop, time)

Field	Description
Arguments	• `integer, intent(in) :: loop` • `real(kind=DP), intent(out) :: time`
GKV binary output	`phi/gkvp_f0.48_(rankg in 6 digits).(ranks in 1 digit).trn.(inum in 3 digits)`
Description	Read simulation time `time` corresponding to the output record `loop`. $(\mathrm{time} \simeq \mathrm{dtout\_eng} \times \mathrm{loop})$

rb_phi_loop(loop, phi)

Field	Description
Arguments	• `integer, intent(in) :: loop` • `complex(kind=DP), intent(out) :: phi(-nx:nx,0:global_ny,-global_nz:global_nz-1)`
GKV binary output	`phi/gkvp_f0.48_(rankg in 6 digits).0.phi.(inum in 3 digits)`
Description	Read electrostatic potential `phi` corresponding to the output record `loop`. $(\mathrm{time} \simeq \mathrm{dtout\_ptn} \times \mathrm{loop})$

rb_Al_loop(loop, Al)

Field	Description
Arguments	• `integer, intent(in) :: loop` • `complex(kind=DP), intent(out) :: Al(-nx:nx,0:global_ny,-global_nz:global_nz-1)`
GKV binary output	`phi/gkvp_f0.48_(rankg in 6 digits).0.Al.(inum in 3 digits)`
Description	Read vector potential `Al` corresponding to the output record `loop`. $(\mathrm{time} \simeq \mathrm{dtout\_ptn} \times \mathrm{loop})$

rb_mom_imomisloop(imom, is, loop, mom)

Field	Description
Arguments	• `integer, intent(in) :: imom, is, loop` • `complex(kind=DP), intent(out) :: mom(-nx:nx,0:global_ny,-global_nz:global_nz-1)`
GKV binary output	`phi/gkvp_f0.48_(rankg in 6 digits).(ranks in 1 digit).mom.(inum in 3 digits)`
Description	Read a fluid moment `mom` corresponding to the output record `loop` $(\mathrm{time} \simeq \mathrm{dtout\_ptn} \times \mathrm{loop})$, where `is` specifies the plasma species, and `imom = 0-5` correspond to $\tilde{n}_{\mathbf{sk}},\ \tilde{u}_{\parallel\mathbf{sk}},\ \tilde{p}_{\parallel\mathbf{sk}},\ \tilde{p}_{\perp\mathbf{sk}},\ \tilde{q}_{\parallel\parallel\mathbf{sk}},\ \tilde{q}_{\parallel\perp\mathbf{sk}}$.

rb_trn_itrnisloop(itrn, is, loop, trn)

Field	Description
Arguments	• `integer, intent(in) :: itrn, is, loop` • `real(kind=DP), intent(out) :: trn(-nx:nx,0:global_ny)`
GKV binary output	`phi/gkvp_f0.48_(rankg in 6 digits).(ranks in 1 digit).trn.(inum in 3 digits)`
Description	Read a variable corresponding to the entropy balance `trn` at the output record `loop` ($\mathrm{time} \simeq \mathrm{dtout\_eng} \times \mathrm{loop}$), where `is` specifies the plasma species, and `itrn = 0-11` correspond to perturbed gyrocenter entropy, electrostatic field energy

Field	Description
	including polarization, magnetic field energy, wave-particle interaction via electrostatic fluctuations, wave-particle interaction via magnetic fluctuations, nonlinear entropy transfers via $\mathbf{E} \times \mathbf{B}$ flows, nonlinear entropy transfer via magnetic flutters, collisional dissipation, particle flux by $\mathbf{E} \times \mathbf{B}$ flows, particle flux by magnetic flutters, energy flux by $\mathbf{E} \times \mathbf{B}$ flows, energy flux by magnetic flutters.

A.5 Diagnostics modules in the post-processing program diag

Some diagnostics modules are explained below.

** List of subroutines in diagnostics modules **

phiinxy(giz, loop)

Field	Description
Contained in	`out_mominxy` module
Arguments	• `integer, intent(in) :: giz, loop`
Output	`post/data/phiinxy_z(giz in 4 digits)_t(loop in 8 digits).dat`
Description	Write 2D electrostatic potential $\tilde{\phi}(x,y)$ for $z = z(\mathtt{giz})$ at output record `loop` $(\mathrm{time} \simeq \mathrm{dtout\_ptn} \times \mathrm{loop})$.

Alinxy(giz, loop)

Field	Description
Contained in	`out_mominxy` module
Arguments	• `integer, intent(in) :: giz, loop`
Output	`post/data/Alinxy_z(giz in 4 digits)_t(loop in 8 digits).dat`

Field	Description
Description	Write 2D vector potential $\tilde{A}_{\parallel}(x,y)$ for $z = z(\mathtt{giz})$ at output record `loop` $(\mathrm{time} \simeq \mathrm{dtout\_ptn} \times \mathrm{loop})$.

mominxy(giz, is, loop)

Field	Description
Contained in	`out_mominxy` module
Arguments	• `integer, intent(in) :: giz, is, loop`
Output	`post/data/mominxy_z(giz in 4 digits)s(is in 1 digit)_t(loop in 8 digits).dat`
Description	Write 2D fluid moments $\tilde{n}_{\mathrm{s}}(x,y),\ \tilde{u}_{\parallel\mathrm{s}}(x,y),\ \tilde{p}_{\parallel\mathrm{s}}(x,y),\ \tilde{p}_{\perp\mathrm{s}}(x,y), \tilde{q}_{\parallel\parallel\mathrm{s}}(x,y),\ \tilde{q}_{\parallel\perp\mathrm{s}}(x,y)$ of the plasma species `is` for $z = z(\mathtt{giz})$ at output record `loop` $(\mathrm{time} \simeq \mathrm{dtout\_ptn} \times \mathrm{loop})$.

phiinz(mx, gmy, loop)

Field	Description
Contained in	`out_mominz` module
Arguments	• `integer, intent(in) :: mx, gmy, loop`
Output	`post/data/phiinz_mx(mx in 4 digits)my(gmy in 4 digits)_t(loop in 8 digits).dat`
Description	Write electrostatic potential along a field line $\tilde{\phi}_{\mathbf{k}}(z)$ for mode $(k_x(\mathtt{mx}),\ k_y(\mathtt{gmy}))$ at output record `loop` $(\mathrm{time} \simeq \mathrm{dtout\_ptn} \times \mathrm{loop})$.

Alinz(mx, gmy, loop)

Field	Description
Contained in	`out_mominz` module

Field	Description
Arguments	• `integer, intent(in) :: mx, gmy, loop`
Output	`post/data/Alinz_mx(mx in 4 digits)my(gmy in 4 digits)_t(loop in 8 digits).dat`
Description	Write vector potential along a field line $\tilde{A}_{\parallel\mathbf{k}}(z)$ for mode $(k_x(\mathtt{mx}),\ k_y(\mathtt{gmy}))$ at output record `loop` $(\mathrm{time} \simeq \mathrm{dtout\_ptn} \times \mathrm{loop})$.

mominz(mx, gmy, is, loop)

Field	Description
Contained in	`out_mominz` module
Arguments	• `integer, intent(in) :: mx, gmy, is, loop`
Output	`post/data/mominz_mx(mx in 4 digits)my(gmy in 4 digits)s(is in 1 digit)_t(loop in 8 digits).dat`
Description	Write fluid moments along a field line for species `is` at output record `loop` : $\tilde{n}_{s\mathbf{k}}(z),\ \tilde{u}_{\parallel s\mathbf{k}}(z),\ \tilde{p}_{\parallel s\mathbf{k}}(z),\ \tilde{p}_{\perp s\mathbf{k}}(z),\ \tilde{q}_{\parallel\parallel s\mathbf{k}}(z),\ \tilde{q}_{\parallel\perp s\mathbf{k}}(z)$ for mode $(k_x(\mathtt{mx}),\ k_y(\mathtt{gmy}))$ $(\mathrm{time} \simeq \mathrm{dtout\_ptn} \times \mathrm{loop})$.

phiinz_connect(mx, gmy, loop)

Field	Description
Contained in	`out_mominz` module
Arguments	• `integer, intent(in) :: mx, gmy, loop`
Output	`post/data/phiinz_connect_mx(mx in 4 digits)my(gmy in 4 digits)_t(loop in 8 digits).dat`
Description	Write electrostatic potential along a field line $\tilde{\phi}_{\mathbf{k}}(z)$ for mode $(k_x(\mathtt{mx}),\ k_y(\mathtt{gmy}))$ at output record `loop` . Considering the pseudo-periodic boundary in the fluxtube model, extend the mode structure by

Field	Description
	connecting $k_x \pm \delta k_x$ modes in the field-aligned coordinate $(\mathrm{time} \simeq \mathrm{dtout\_ptn} \times \mathrm{loop})$.

phiinkxky(loop)

Field	Description
Contained in	`out_mominkxky` module
Arguments	• `integer, intent(in) :: loop`
Output	`post/data/phiinkxky_t(loop in 8 digits).dat`
Description	Write (k_x, k_y) spectrum of electrostatic potential $\langle \lvert\tilde{\phi}_{\mathbf{k}}\rvert^2 \rangle/2$ at output record `loop` $(\mathrm{time} \simeq \mathrm{dtout\_ptn} \times \mathrm{loop})$.

Alinkxky(loop)

Field	Description
Contained in	`out_mominkxky` module
Arguments	• `integer, intent(in) :: loop`
Output	`post/data/Alinkxky_t(loop in 8 digits).dat`
Description	Write (k_x, k_y) spectrum of vector potential $\langle \lvert\tilde{A}_{\parallel\mathbf{k}}\rvert^2 \rangle/2$ at output record `loop` $(\mathrm{time} \simeq \mathrm{dtout\_ptn} \times \mathrm{loop})$.

mominkxky(is, loop)

Field	Description
Contained in	`out_mominkxky` module
Arguments	• `integer, intent(in) :: is, loop`

Field	Description
Output	`post/data/mominkxky_s(is in 1 digit)_t(loop in 8 digits).dat`
Description	Write (k_x, k_y) spectra of fluid moments of species `is` at output record `loop`: $\langle\lvert\tilde{n}_{\mathrm{s}\mathbf{k}}\rvert^2\rangle/2,\ \langle\lvert\tilde{u}_{\parallel\mathrm{s}\mathbf{k}}\rvert^2\rangle/2,\ \langle\lvert\tilde{p}_{\parallel\mathrm{s}\mathbf{k}}\rvert^2\rangle/2,\ \langle\lvert\tilde{p}_{\perp\mathrm{s}\mathbf{k}}\rvert^2\rangle/2,\ \langle\lvert\tilde{q}_{\parallel\parallel\mathrm{s}\mathbf{k}}\rvert^2\rangle/2,\ \langle\lvert\tilde{q}_{\parallel\perp\mathrm{s}\mathbf{k}}\rvert^2\rangle/2$ $(\mathrm{time} \simeq \mathrm{dtout\_ptn} \times \mathrm{loop})$.

trninkxky(is, loop)

Field	Description
Contained in	`out_trninkxky` module
Arguments	• `integer, intent(in) :: is, loop`
Output	`post/data/trninkxky_s(is in 1 digit)_t(loop in 8 digits).dat`
Description	Write (k_x, k_y) spectra of variables in the entropy-balance relation of species `is` at output record `loop` $(\mathrm{time} \simeq \mathrm{dtout\_eng} \times \mathrm{loop})$.

triinkxky(mxt, myt, is, loop)

Field	Description
Contained in	`out_triinkxky` module
Arguments	• `integer, intent(in) :: mxt, myt, is, loop`
Output	`post/data/trninkxky_s(is in 1 digit)_t(loop in 8 digits).dat`
Description	Write (p_x, p_y) spectra of triad transfer functions $J_{\mathrm{sE}\mathbf{k}}^{\mathbf{p},\mathbf{q}},\ J_{\mathrm{sE}\mathbf{p}}^{\mathbf{q},\mathbf{k}},\ J_{\mathrm{sE}\mathbf{q}}^{\mathbf{k},\mathbf{p}},\ J_{\mathrm{sM}\mathbf{k}}^{\mathbf{p},\mathbf{q}},\ J_{\mathrm{sM}\mathbf{p}}^{\mathbf{q},\mathbf{k}},\ J_{\mathrm{sM}\mathbf{q}}^{\mathbf{k},\mathbf{p}}$ of species `is` for mode $(k_x(\mathtt{mxt}), k_y(\mathtt{myt}))$ at output record `loop` $(\mathrm{time} \simeq \mathrm{dtout\_ptn} \times \mathrm{loop})$.

A.6 Adiabatic electron/ion model for nprocs=1

When one runs a single-species simulation with setting `nprocs`=1, GKV employs adiabatic model for electrons or ions. In both case, electrostatic limit is assumed ($\tilde{A}_{\|} = 0$), and `lambda_i` and `beta` in `gkvp_f0.48_namelist` are neglected. Setting of kinetic electrons with adiabatic ion model is `nprocs`=1 in `src/gkvp_f0.48_header.f90`, and `Anum`=1.d0, `Znum`=1.d0, `fcs`=1.d0, `sgn`=-1.d0 in `run/gkvp_f0.48_namelist`. Then the Poisson eq. with adiabatic ion model is

$$\left[\frac{e^2 n_0}{T_\mathrm{i}} + \frac{e^2 n_0}{T_\mathrm{e}}(1-\Gamma_{0\mathrm{e}\boldsymbol{k}})\right]\tilde{\phi}_{\boldsymbol{k}} = -e\int dv^3 J_{0\mathrm{e}\boldsymbol{k}}\tilde{f}_{\mathrm{e}\boldsymbol{k}}. \tag{A.1}$$

Density, temperature and mass are normalized electrons' value. Then the normalized Poisson eq. is

$$\left[\frac{T_\mathrm{e}}{T_\mathrm{i}} + 1 - \bar{\Gamma}_{0\mathrm{e}\boldsymbol{k}}\right]\bar{\phi}_{\boldsymbol{k}} = -\int d\bar{v}^3 \bar{J}_{0\mathrm{e}\boldsymbol{k}}\bar{f}_{\mathrm{e}\boldsymbol{k}}. \tag{A.2}$$

The temperature ratio $T_\mathrm{e}/T_\mathrm{i}$ is given by `tau_ad` in `run/gkvp_f0.48_namelist`. Setting of kinetic ions with adiabatic electron model is `nprocs`=1 in `src/gkvp_f0.48_header.f90`, and `Anum`=1.d0, `Znum`=1.d0, `fcs`=1.d0, `sgn`=1.d0 in `run/gkvp_f0.48_namelist`. Then the Poisson eq. with adiabatic electron model is

$$\frac{e^2 n_0}{T_\mathrm{i}}(1-\Gamma_{0\mathrm{i}\boldsymbol{k}})\tilde{\phi}_{\boldsymbol{k}} = -\frac{e^2 n_0}{T_\mathrm{e}}\left(\tilde{\phi}_{\boldsymbol{k}} - \langle\tilde{\phi}_{\boldsymbol{k}}\rangle\delta_{k_y,0}\right) + e\int dv^3 J_{0\mathrm{i}\boldsymbol{k}}\tilde{f}_{\mathrm{i}\boldsymbol{k}}, \tag{A.3}$$

where $\langle\cdots\rangle$ denotes the flux-surface average, and $\delta_{i,j}$ is the Kronecker's delta. Density, temperature and mass are normalized ions' value. Then the normalized Poisson eq. is

$$\left(1-\bar{\Gamma}_{0\mathrm{i}\boldsymbol{k}}\right)\bar{\phi}_{\boldsymbol{k}} + \frac{T_\mathrm{i}}{T_\mathrm{e}}\left(\bar{\phi}_{\boldsymbol{k}} - \langle\bar{\phi}_{\boldsymbol{k}}\rangle\delta_{k_y,0}\right) = \int dv^3 \bar{J}_{0\mathrm{i}\boldsymbol{k}}\bar{f}_{\mathrm{i}\boldsymbol{k}}, \tag{A.4}$$

The temperature ratio $T_\mathrm{i}/T_\mathrm{e}$ is given by `tau_ad` in `run/gkvp_f0.48_namelist`.

B Supplemental

B.1 Entropy balance equation for each wavenumber and plasma species

$$\begin{aligned}
\frac{dS_{\mathrm{s}\boldsymbol{k}}}{dt} &= \frac{T_\mathrm{s}\Gamma_{\mathrm{s}\boldsymbol{k}}}{L_{p\mathrm{s}}} + \frac{\Theta_{\mathrm{s}\boldsymbol{k}}}{L_{T\mathrm{s}}} + I_{\mathrm{s}\boldsymbol{k}} + R_{\mathrm{s}\boldsymbol{k}} + D_{\mathrm{s}\boldsymbol{k}} + E_{\mathrm{s}\boldsymbol{k}}, && \text{(B.1)}\\
\frac{dW_{\mathrm{E}\boldsymbol{k}}}{dt} &= -\sum_\mathrm{s} R_{\mathrm{sE}\boldsymbol{k}}, && \text{(B.2)}\\
\frac{dW_{\mathrm{M}\boldsymbol{k}}}{dt} &= -\sum_\mathrm{s} R_{\mathrm{sM}\boldsymbol{k}}, && \text{(B.3)}
\end{aligned}$$

where

$$S_{\mathrm{s}\boldsymbol{k}} = \left\langle \int dv^3 \frac{T_\mathrm{s}|\tilde{f}_{\mathrm{s}\boldsymbol{k}}|^2}{2F_{\mathrm{sM}}} \right\rangle, \tag{B.4}$$

$$W_{\mathrm{E}\boldsymbol{k}} = \left\langle \left[\varepsilon_0 k_\perp^2 + \sum_\mathrm{s} \frac{e_\mathrm{s}^2 n_\mathrm{s}}{T_\mathrm{s}} (1 - \Gamma_{0\mathrm{s}\boldsymbol{k}}) \right] \frac{|\tilde{\phi}_{\boldsymbol{k}}|^2}{2} \right\rangle, \tag{B.5}$$

$$W_{\mathrm{M}\boldsymbol{k}} = \left\langle \frac{k_\perp^2}{\mu_0} \frac{|\tilde{A}_{\parallel\boldsymbol{k}}|^2}{2} \right\rangle, \tag{B.6}$$

$$\Gamma_{\mathrm{s}\boldsymbol{k}} = \Gamma_{\mathrm{sE}\boldsymbol{k}} + \Gamma_{\mathrm{sM}\boldsymbol{k}} = \mathrm{Re}\left[\left\langle -\frac{ik_y \tilde{\phi}_{\boldsymbol{k}}}{c_b} \tilde{n}_{\mathrm{s}\boldsymbol{k}}^* + \frac{ik_y \tilde{A}_{\parallel\boldsymbol{k}}}{c_b} \tilde{u}_{\parallel\mathrm{s}\boldsymbol{k}}^* \right\rangle \right], \tag{B.7}$$

$$Q_{\mathrm{s}\boldsymbol{k}} = Q_{\mathrm{sE}\boldsymbol{k}} + Q_{\mathrm{sM}\boldsymbol{k}} = \mathrm{Re}\left[\left\langle -\frac{ik_y \tilde{\phi}_{\boldsymbol{k}}}{c_b} \tilde{p}_{\mathrm{s}\boldsymbol{k}}^* + \frac{ik_y \tilde{A}_{\parallel\boldsymbol{k}}}{c_b} \tilde{q}_{\parallel\mathrm{s}\boldsymbol{k}}^* \right\rangle \right], \tag{B.8}$$

$$\Theta_{\mathrm{s}\boldsymbol{k}} = Q_{\mathrm{s}\boldsymbol{k}} - \frac{5}{2} T_\mathrm{s} \Gamma_{\mathrm{s}\boldsymbol{k}}, \tag{B.9}$$

$$I_{\mathrm{s}\boldsymbol{k}} = \sum_{\boldsymbol{p}} \sum_{\boldsymbol{q}} J_{\mathrm{s}\boldsymbol{k}}^{\boldsymbol{p},\boldsymbol{q}}, \tag{B.10}$$

$$R_{\mathrm{s}\boldsymbol{k}} = R_{\mathrm{sE}\boldsymbol{k}} + R_{\mathrm{sM}\boldsymbol{k}} = \mathrm{Re}\left[\left\langle -\tilde{\phi}_{\boldsymbol{k}}^* \frac{\partial e_\mathrm{s} \tilde{n}_{\mathrm{s}\boldsymbol{k}}}{\partial t} - e_\mathrm{s} \tilde{u}_{\mathrm{s}\boldsymbol{k}}^* \frac{\partial \tilde{A}_{\parallel\boldsymbol{k}}}{\partial t} \right\rangle \right], \tag{B.11}$$

$$D_{\mathrm{s}\boldsymbol{k}} = \mathrm{Re}\left[\left\langle \int dv^2 \frac{T_\mathrm{s} \tilde{g}_{\mathrm{s}\boldsymbol{k}}^*}{F_{\mathrm{sM}}} C_{\mathrm{s}\boldsymbol{k}} \right\rangle \right], \tag{B.12}$$

$$E_{\mathrm{s}\boldsymbol{k}} = \mathrm{Re}\left[-\left\langle \int dv^3 v_\parallel \nabla_\parallel \frac{T_\mathrm{s}|\tilde{g}_{\mathrm{s}\boldsymbol{k}}|^2}{2F_{\mathrm{sM}}} \right\rangle \right], \tag{B.13}$$

with

$$\tilde{g}_{\mathrm{s}\boldsymbol{k}} = \tilde{f}_{\mathrm{s}\boldsymbol{k}} + \frac{e_\mathrm{s} J_{0\mathrm{s}\boldsymbol{k}} \tilde{\phi}_{\boldsymbol{k}}}{T_\mathrm{s}} F_{\mathrm{sM}}, \tag{B.14}$$

$$\tilde{n}_{\mathrm{s}\boldsymbol{k}} = \int dv^3 J_{0\mathrm{s}\boldsymbol{k}} \tilde{f}_{\mathrm{s}\boldsymbol{k}}, \tag{B.15}$$

$$\tilde{u}_{\parallel\mathrm{s}\boldsymbol{k}} = \int dv^3 v_\parallel J_{0\mathrm{s}\boldsymbol{k}} \tilde{f}_{\mathrm{s}\boldsymbol{k}}, \tag{B.16}$$

$$\tilde{p}_{\parallel\mathrm{s}\boldsymbol{k}} = \int dv^3 \frac{m_\mathrm{s} v_\parallel^2}{2} J_{0\mathrm{s}\boldsymbol{k}} \tilde{f}_{\mathrm{s}\boldsymbol{k}}, \tag{B.17}$$

$$\tilde{p}_{\perp\mathrm{s}\boldsymbol{k}} = \int dv^3 \mu B J_{0\mathrm{s}\boldsymbol{k}} \tilde{f}_{\mathrm{s}\boldsymbol{k}}, \tag{B.18}$$

$$\tilde{q}_{\parallel\parallel\mathrm{s}\boldsymbol{k}} = \int dv^3 v_\parallel \frac{m_\mathrm{s} v_\parallel^2}{2} J_{0\mathrm{s}\boldsymbol{k}} \tilde{f}_{\mathrm{s}\boldsymbol{k}}, \tag{B.19}$$

$$\tilde{q}_{\parallel\perp\mathrm{s}\boldsymbol{k}} = \int dv^3 v_\parallel \mu B J_{0\mathrm{s}\boldsymbol{k}} \tilde{f}_{\mathrm{s}\boldsymbol{k}}, \tag{B.20}$$

$$\tilde{p}_{\mathrm{s}\boldsymbol{k}} = \tilde{p}_{\parallel\mathrm{s}\boldsymbol{k}} + \tilde{p}_{\perp\mathrm{s}\boldsymbol{k}}, \tag{B.21}$$

$$\tilde{q}_{\parallel\mathrm{s}\boldsymbol{k}} = \tilde{q}_{\parallel\parallel\mathrm{s}\boldsymbol{k}} + \tilde{q}_{\parallel\perp\mathrm{s}\boldsymbol{k}}. \tag{B.22}$$

See also Refs. [B-1] and [B-2].

B.2 Triad transfer function

$$J_{\mathrm{s}\boldsymbol{k}}^{\boldsymbol{p},\boldsymbol{q}} = \delta_{\boldsymbol{k}+\boldsymbol{p}+\boldsymbol{q},\boldsymbol{0}} \frac{\boldsymbol{b}\cdot\boldsymbol{p}\times\boldsymbol{q}}{2c_b} \mathrm{Re}\left[\left\langle \int dv^3 (\chi_{\mathrm{s}\boldsymbol{p}}\tilde{g}_{\mathrm{s}\boldsymbol{q}} - \chi_{\mathrm{s}\boldsymbol{q}}\tilde{g}_{\mathrm{s}\boldsymbol{p}}) \frac{T_{\mathrm{s}}\tilde{g}_{\mathrm{s}\boldsymbol{k}}}{F_{\mathrm{sM}}} \right\rangle\right], \tag{B.23}$$

where $\tilde{g}_{\mathrm{s}\boldsymbol{k}} = \tilde{f}_{\mathrm{s}\boldsymbol{k}} + e_{\mathrm{s}} J_{0\mathrm{s}\boldsymbol{k}} \tilde{\phi}_{\boldsymbol{k}} F_{\mathrm{sM}}/T_{\mathrm{s}}$ and $\chi_{\mathrm{s}\boldsymbol{k}} = J_{0\mathrm{s}\boldsymbol{k}}(\tilde{\phi}_{\boldsymbol{k}} - v_{\parallel}\tilde{A}_{\parallel\boldsymbol{k}})$. The triad transfer function satisfy the following properties [B-3]:

$$J_{\mathrm{s}\boldsymbol{k}}^{\boldsymbol{p},\boldsymbol{q}} = J_{\mathrm{s}\boldsymbol{k}}^{\boldsymbol{q},\boldsymbol{p}}, \tag{B.24}$$

$$J_{\mathrm{s}\boldsymbol{k}}^{\boldsymbol{p},\boldsymbol{q}} + J_{\mathrm{s}\boldsymbol{p}}^{\boldsymbol{q},\boldsymbol{k}} + J_{\mathrm{s}\boldsymbol{q}}^{\boldsymbol{k},\boldsymbol{p}} = 0. \tag{B.25}$$

Note that $J_{\mathrm{s}\boldsymbol{k}}^{\boldsymbol{p},\boldsymbol{q}}$ is symmetrized so as to eliminate asymmetric components, which cancel out in the net entropy transfer $I_{\mathrm{s}\boldsymbol{k}}$ and thus do not contribute to physics. Since the terms of $\tilde{\phi}_{\boldsymbol{k}}$ and of $\tilde{A}_{\parallel\boldsymbol{k}}$ respectively correspond to $\boldsymbol{E}\times\boldsymbol{B}$ and magnetic flutter nonlinearities, these contributions can be evaluated separately,

$$I_{\mathrm{s}\boldsymbol{k}} = I_{\mathrm{sE}\boldsymbol{k}} + I_{\mathrm{sM}\boldsymbol{k}} = \sum_{\boldsymbol{p}}\sum_{\boldsymbol{q}} J_{\mathrm{sE}\boldsymbol{k}}^{\boldsymbol{p},\boldsymbol{q}} + \sum_{\boldsymbol{p}}\sum_{\boldsymbol{q}} J_{\mathrm{sM}\boldsymbol{k}}^{\boldsymbol{p},\boldsymbol{q}}, \tag{B.26}$$

$$J_{\mathrm{s}\boldsymbol{k}}^{\boldsymbol{p},\boldsymbol{q}} = J_{\mathrm{sE}\boldsymbol{k}}^{\boldsymbol{p},\boldsymbol{q}} + J_{\mathrm{sM}\boldsymbol{k}}^{\boldsymbol{p},\boldsymbol{q}}. \tag{B.27}$$

B.3 Integrals in GKV

Flux-surface average:

$$\left\langle \tilde{\phi}(x,y,z) \right\rangle = \sum_{k_x} \left\langle \tilde{\phi}_{k_x,k_y=0}(z) \right\rangle e^{ik_x x}, \tag{B.28}$$

$$\left\langle \tilde{\phi}_{k_x,k_y=0}(z) \right\rangle = \frac{\int_{\pi}^{\pi} dz \sqrt{g}\tilde{\phi}_{k_x,k_y=0}(z)}{\int_{\pi}^{\pi} dz \sqrt{g}}. \tag{B.29}$$

Volume average:

$$\int dx^3 \left|\tilde{\phi}(x,y,z)\right|^2 = \sum_{k_x}\sum_{k_y} \left\langle \left|\tilde{\phi}_{\boldsymbol{k}}(z)\right|^2 \right\rangle. \tag{B.30}$$

Velocity-space integral:

$$\int dv^3 \tilde{f}_{\mathrm{s}\boldsymbol{k}}(z, v_{\parallel}, mu) = \int_{-L_v}^{L_v} dv_{\parallel} \int_0^{L_v} dv_{\perp} 2\pi v_{\perp} \tilde{f}_{\mathrm{s}\boldsymbol{k}}(z, v_{\parallel}, \mu). \tag{B.31}$$

B.4 References

[B-1] H. Sugama, T.-H. Watanabe, and M. Nunami, *Phys. Plasmas* **16**, 112503 (2009).
[B-2] S. Maeyama, A. Ishizawa, T.-H. Watanabe, M. Nakata, N. Miyato, M. Yagi, and Y. Idomura, *Phys. Plasmas* **21**, 052301 (2014).
[B-3] M. Nakata, T.-H. Watanabe, and H. Sugama, *Phys. Plasmas* **19**, 022303 (2012).