

Introduction to Eigenvalue Solver for ITG and Gyrofluid Simulation using BOUT++ GLF code in Core Region

S. S. Kim, H. Jhang, P. H. Diamond, T. Rhee, G. Y. Park
WCI Center for Fusion Theory, NFRI, Korea

X. Q. Xu, P. W. Xi, A. Dimits, M. Umansky
LLNL

BOUT++ 2013 Workshop

Introduction to Eigenvalue Solver for ITG

Governing equations for ITG

(Ottaviani et. al., PoP '99)

- Vorticity equation

$$\frac{\partial \Omega_i}{\partial t} + \mathbf{V}_E \cdot \nabla \Omega_i = -n \nabla_{\parallel} V_{\parallel} + n (\mathbf{V}_E + \mathbf{V}_{p_i}) \cdot (\boldsymbol{\kappa} + \nabla \ln B) + n \mathbf{V}_{p_i} \cdot \nabla \left(\frac{n_1 - \Omega_i}{n} \right) - \mathbf{V}_E \cdot \nabla n_{eq} + D_c \Omega_i$$

where $\Omega_i = n_1 - n_{pol}$: generalized vorticity, $n_{pol} = \frac{ne}{m_i \omega_{ci}^2} \nabla_{\perp}^2 \varphi$, $n_1 = n_{eq} \frac{e \varphi_1}{T_e}$: adiabatic response

- Ion temperature equation

$$\mathbf{V}_E = \frac{c}{B} \mathbf{b} \times \nabla \varphi, \quad \mathbf{V}_{p_i} = \frac{c}{neB} \mathbf{b} \times \nabla p_i$$

$$\frac{\partial T_i}{\partial t} + \mathbf{V}_E \cdot \nabla T_i = -\frac{2}{3} T_i \nabla_{\parallel} V_{\parallel} + D_c T_i + D_{glf} T_i$$

- Ion parallel velocity equation

$$\frac{\partial V_{\parallel}}{\partial t} + \mathbf{V}_E \cdot \nabla V_{\parallel} = -\frac{e}{m_i} \nabla_{\parallel} \varphi - \frac{1}{m_i n} \nabla_{\parallel} p_i + D_c V_{\parallel}$$

where $D_c F = \mu_1 \nabla_{\perp}^2 F - \mu_2 \nabla_{\perp}^4 F$: viscous damping, $D_{glf} T_i = -\sqrt{\frac{8T_{i,eq}}{\pi m_i}} |\nabla_{\parallel}| T_{i1}$: Landau damping.

Linearized equations

$$\frac{\partial \tilde{\Omega}_i}{\partial \bar{t}} = -\hat{n}_i \hat{\nabla}_{\parallel} \tilde{V}_{i\parallel} + i(\hat{n}_i \omega_{di} \tilde{\varphi} + \omega_{di} \tilde{p}_i) + \left[\frac{\hat{p}_i}{\rho_*}, \tilde{\nabla}_{\perp}^2 \tilde{\varphi} \right] - \left[\tilde{\Psi}, \frac{\hat{n}_i}{\rho_*} \right] + \bar{D}_c \tilde{\Omega}_i$$

$$\frac{\partial \tilde{T}_i}{\partial \bar{t}} + \left[\tilde{\Psi}, \frac{\hat{T}_i}{\rho_*} \right] = -\frac{2}{3} \hat{T}_i \nabla_{\parallel} \tilde{V}_{i\parallel} + \bar{D}_c \tilde{T}_i + \bar{D}_{glf} \tilde{T}_i$$

$$\frac{\partial \tilde{V}_{i\parallel}}{\partial \bar{t}} = -\hat{\nabla}_{\parallel} \tilde{\varphi} - \frac{1}{\hat{n}_i} \hat{\nabla}_{\parallel} \tilde{p}_i + \bar{D}_c \tilde{V}_{i\parallel}$$

where

$$\tilde{\Omega}_i = \tilde{n}_i - \hat{n}_i \tilde{\nabla}_{\perp}^2 \tilde{\varphi}, \quad \tilde{n}_i = \frac{\hat{n}_i}{\hat{T}_e} \tilde{\varphi},$$

$$\tilde{\Psi} = \Gamma_0^{1/2} \tilde{\varphi} = \left(1 - \frac{1}{2} \tilde{\nabla}_{\perp}^2 \right)^{-1} \tilde{\varphi}$$

is gyroaveraged potential.

following normalizations are used,

$$F = F_{eq} + F_1, \quad F = (\Omega, V_{\parallel}, n, T, \varphi), \quad F_0 = \left(n_0, c_{s0}, n_0, T_0, \frac{T_0}{e} \right)$$

$$\hat{F} = \frac{F_{eq}}{F_0}, \quad \tilde{F} = \frac{F_1}{\rho_* F_0}, \quad \tilde{\nabla}_{\perp} = \rho_* a \nabla_{\perp}, \quad \hat{\nabla}_{\parallel} = a \nabla_{\parallel}, \quad \rho = \frac{r}{a}, \quad \bar{t} = \frac{t}{t_0}$$

$$\rho_* = \frac{\rho_{s0}}{a}, \quad \rho_{s0} = \frac{c_{s0}}{\omega_{c0}}, \quad c_{s0} = \sqrt{\frac{T_0}{m_i}}, \quad \omega_{c0} = \frac{e_i B_0}{m_i c}, \quad t_0 = \frac{a}{c_{s0}}$$

$$[f, g] \equiv \rho_{s0}^2 \mathbf{b} \times \nabla f \cdot \nabla g = \frac{\rho_*^2}{\rho} \left(\frac{\partial f}{\partial \rho} \frac{\partial g}{\partial \theta} - \frac{\partial g}{\partial \rho} \frac{\partial f}{\partial \theta} \right)$$

$$i \omega_{di} \tilde{F} \equiv 2 \rho_{s0} a \mathbf{b} \times \nabla \tilde{F} \cdot \nabla \ln B = 2 \rho_* \frac{a}{R} \left(\frac{1}{\rho} \cos \theta \frac{\partial \tilde{F}}{\partial \theta} + \sin \theta \frac{\partial \tilde{F}}{\partial \rho} \right)$$

Note: F_0 is constant (not the value on magnetic axis)

$$\frac{\partial \tilde{\Omega}_i}{\partial \bar{t}} = -\hat{n}_i \hat{\nabla}_{\parallel} \tilde{V}_{i\parallel} + i(\hat{n}_i \omega_{di} \tilde{\varphi} + \omega_{di} \tilde{p}_i) + \left[\frac{\hat{p}_i}{\rho_*}, \tilde{\nabla}_{\perp}^2 \tilde{\varphi} \right] + \left[\frac{\hat{n}_i}{\rho_*}, \tilde{\Psi} \right] + \bar{D}_c \tilde{\Omega}_i$$

$$\Rightarrow \omega M \tilde{\varphi} = \hat{n}_i k_{\parallel} \tilde{V}_{i\parallel} - a_w \tilde{\varphi} - b_w \tilde{T}_i + G_{\hat{p}_i} L \tilde{\varphi} + G_{\hat{n}_i} J_0 \tilde{\varphi} + i \bar{D}_c M \tilde{\varphi}$$

$$\frac{\partial \tilde{T}_i}{\partial \bar{t}} = \left[\frac{\hat{T}_i}{\rho_*}, \tilde{\Psi} \right] - \frac{2}{3} \hat{T}_i \nabla_{\parallel} \tilde{V}_{i\parallel} + \bar{D}_c \tilde{T}_i + \bar{D}_{glf} \tilde{T}_i$$

$$\Rightarrow \omega \tilde{T}_i = G_{\hat{T}_i} J_0 \tilde{\varphi}_i + \frac{2}{3} \hat{T}_i k_{\parallel} \tilde{V}_{i\parallel} + i \bar{D}_c \tilde{T}_i + i \bar{D}_{glf} \tilde{T}_i$$

$$\frac{\partial \tilde{V}_{i\parallel}}{\partial \bar{t}} = -\hat{\nabla}_{\parallel} \tilde{\varphi} - \frac{1}{\hat{n}_i} \hat{\nabla}_{\parallel} \tilde{p}_i + \bar{D}_c \tilde{V}_{i\parallel}$$

$$\Rightarrow \omega \tilde{V}_{i\parallel} = \left(1 + \frac{\hat{T}_i}{\hat{T}_e} \right) k_{\parallel} \tilde{\varphi} + k_{\parallel} \tilde{T}_i + i \bar{D}_c \tilde{V}_{i\parallel}$$

where we introduce

$$\frac{\partial}{\partial \bar{t}} \equiv -i\omega, \quad \tilde{\nabla}_{\perp}^2 \tilde{\varphi} \equiv L \tilde{\varphi}, \quad \hat{\nabla}_{\parallel} \equiv -ik_{\parallel}$$

$$\tilde{\Omega}_i = \left(\frac{\hat{n}_i}{\hat{T}_e} - \hat{n}_i \tilde{\nabla}_{\perp}^2 \right) \tilde{\varphi} \equiv M \tilde{\varphi}$$

$$\tilde{\Psi} = \left(1 - \frac{1}{2} \tilde{\nabla}_{\perp}^2 \right)^{-1} \tilde{\varphi} \equiv J_0 \tilde{\varphi}$$

$$\left[\frac{\hat{F}}{\rho_*}, \right] = \rho_{s0}^2 \mathbf{b} \times \nabla \frac{\hat{F}}{\rho_*} \cdot \nabla \equiv -iG_{\hat{F}}$$

$$i(\hat{n}_i \omega_{di} \tilde{\varphi} + \omega_{di} \tilde{p}_i) \equiv i(a_w \tilde{\varphi} + b_w \tilde{T}_i)$$

and use

$$\tilde{p}_i = \frac{\hat{p}_i}{\hat{T}_e} \tilde{\varphi}_i + \hat{n}_i \tilde{T}_i.$$

$$\frac{\partial \tilde{\Omega}_i}{\partial \bar{t}} = -\hat{n}_i \hat{\nabla}_{\parallel} \tilde{V}_{i\parallel} + i(\hat{n}_i \omega_{di} \tilde{\varphi} + \omega_{di} \tilde{p}_i) + \left[\frac{\hat{p}_i}{\rho_*}, \tilde{\nabla}_{\perp}^2 \tilde{\varphi} \right] + \left[\frac{\hat{n}_i}{\rho_*}, \tilde{\Psi} \right] + \bar{D}_c \tilde{\Omega}_i$$

$$\Rightarrow \omega M \tilde{\varphi} = \hat{n}_i k_{\parallel} \tilde{V}_{i\parallel} - a_w \tilde{\varphi} - b_w \tilde{T}_i + G_{\hat{p}_i} L \tilde{\varphi} + G_{\hat{n}_i} J_0 \tilde{\varphi} + i \bar{D}_c M \tilde{\varphi}$$

$$\Rightarrow \omega \tilde{\varphi} = M^{-1} \left(-a_w + G_{\hat{p}_i} L + G_{\hat{n}_i} J_0 + i \bar{D}_c M \right) \tilde{\varphi} - M^{-1} b_w \tilde{T}_i + M^{-1} \hat{n}_i k_{\parallel} \tilde{V}_{i\parallel}$$

$$\frac{\partial \tilde{T}_i}{\partial \bar{t}} = \left[\frac{\hat{T}_i}{\rho_*}, \tilde{\Psi} \right] - \frac{2}{3} \hat{T}_i \nabla_{\parallel} \tilde{V}_{i\parallel} + \bar{D}_c \tilde{T}_i + \bar{D}_{glf} \tilde{T}_i$$

$$\Rightarrow \omega \tilde{T}_i = G_{\hat{T}_i} J_0 \tilde{\varphi}_i + \frac{2}{3} \hat{T}_i k_{\parallel} \tilde{V}_{i\parallel} + i \bar{D}_c \tilde{T}_i + i \bar{D}_{glf} \tilde{T}_i$$

$$\Rightarrow \omega \tilde{T}_i = G_{\hat{T}_i} J_0 \tilde{\varphi}_i + (i \bar{D}_c + i \bar{D}_{glf}) \tilde{T}_i + \frac{2}{3} \hat{T}_i k_{\parallel} \tilde{V}_{i\parallel}$$

$$\frac{\partial \tilde{V}_{i\parallel}}{\partial \bar{t}} = -\hat{\nabla}_{\parallel} \tilde{\varphi} - \frac{1}{\hat{n}_i} \hat{\nabla}_{\parallel} \tilde{p}_i + \bar{D}_c \tilde{V}_{i\parallel}$$

$$\Rightarrow \omega \tilde{V}_{i\parallel} = \left(1 + \frac{\hat{T}_i}{\hat{T}_e} \right) k_{\parallel} \tilde{\varphi} + k_{\parallel} \tilde{T}_i + i \bar{D}_c \tilde{V}_{i\parallel}$$

where we introduce

$$\frac{\partial}{\partial \bar{t}} \equiv -i\omega, \quad \tilde{\nabla}_{\perp}^2 \tilde{\varphi} \equiv L \tilde{\varphi}, \quad \hat{\nabla}_{\parallel} \equiv -ik_{\parallel}$$

$$\tilde{\Omega}_i = \left(\frac{\hat{n}_i}{\hat{T}_e} - \hat{n}_i \tilde{\nabla}_{\perp}^2 \right) \tilde{\varphi} \equiv M \tilde{\varphi}$$

$$\tilde{\Psi} = \left(1 - \frac{1}{2} \tilde{\nabla}_{\perp}^2 \right)^{-1} \tilde{\varphi} \equiv J_0 \tilde{\varphi}$$

$$\left[\frac{\hat{F}}{\rho_*}, \right] = \rho_{s0}^2 \mathbf{b} \times \nabla \frac{\hat{F}}{\rho_*} \cdot \nabla \equiv -i G_{\hat{F}}$$

$$i(\hat{n}_i \omega_{di} \tilde{\varphi} + \omega_{di} \tilde{p}_i) \equiv i(a_w \tilde{\varphi} + b_w \tilde{T}_i)$$

and use

$$\tilde{p}_i = \frac{\hat{p}_i}{\hat{T}_e} \tilde{\varphi}_i + \hat{n}_i \tilde{T}_i.$$

$$\frac{\partial \tilde{\Omega}_i}{\partial \tilde{t}} = -\hat{n}_i \hat{\nabla}_{\parallel} \tilde{V}_{i\parallel} + i(\hat{n}_i \omega_{di} \tilde{\varphi} + \omega_{di} \tilde{p}_i) + \left[\frac{\hat{p}_i}{\rho_*}, \tilde{\nabla}_{\perp}^2 \tilde{\varphi} \right] + \left[\frac{\hat{n}_i}{\rho_*}, \tilde{\Psi} \right] + \bar{D}_c \tilde{\Omega}_i$$

$$\Rightarrow \omega M \tilde{\varphi} = \hat{n}_i k_{\parallel} \tilde{V}_{i\parallel} - a_w \tilde{\varphi} - b_w \tilde{T}_i + G_{\hat{p}_i} L \tilde{\varphi} + G_{\hat{n}_i} J_0 \tilde{\varphi} + i \bar{D}_c M \tilde{\varphi}$$

$$\Rightarrow \omega \tilde{\varphi} = M^{-1} \left(-a_w + G_{\hat{p}_i} L + G_{\hat{n}_i} J_0 + i \bar{D}_c M \right) \tilde{\varphi} - M^{-1} b_w \tilde{T}_i + M^{-1} \hat{n}_i k_{\parallel} \tilde{V}_{i\parallel}$$

Those operators can be represented as matrices in a functional space.

$$\frac{\partial \tilde{T}_i}{\partial \tilde{t}} = \left[\frac{\hat{T}_i}{\rho_*}, \tilde{\Psi} \right] - \frac{2}{3} \hat{T}_i \nabla_{\parallel} \tilde{V}_{i\parallel} + \bar{D}_c \tilde{T}_i + \bar{D}_{glf} \tilde{T}_i$$

$$\Rightarrow \omega \tilde{T}_i = G_{\hat{T}_i} J_0 \tilde{\varphi}_i + \frac{2}{3} \hat{T}_i k_{\parallel} \tilde{V}_{i\parallel} + i \bar{D}_c \tilde{T}_i + i \bar{D}_{glf} \tilde{T}_i$$

$$\Rightarrow \omega \tilde{T}_i = G_{\hat{T}_i} J_0 \tilde{\varphi}_i + (i \bar{D}_c + i \bar{D}_{glf}) \tilde{T}_i + \frac{2}{3} \hat{T}_i k_{\parallel} \tilde{V}_{i\parallel}$$

where we introduce

$$\frac{\partial}{\partial \tilde{t}} \equiv -i\omega, \quad \tilde{\nabla}_{\perp}^2 \tilde{\varphi} \equiv L \tilde{\varphi}, \quad \hat{\nabla}_{\parallel} \equiv -k_{\parallel}$$

$$\tilde{\Omega}_i = \left(\frac{\hat{n}_i}{\hat{T}_e} - \hat{n}_i \tilde{\nabla}_{\perp}^2 \right) \tilde{\varphi} \equiv M \tilde{\varphi}$$

$$\tilde{\Psi} = \left(1 - \frac{1}{2} \tilde{\nabla}_{\perp}^2 \right)^{-1} \tilde{\varphi} \equiv J_0 \tilde{\varphi}$$

$$\left[\frac{\hat{F}}{\rho_*}, \right] = \rho_{s0}^2 \mathbf{b} \times \nabla \frac{\hat{F}}{\rho_*} \cdot \nabla \equiv -i G_{\hat{F}}$$

$$i(\hat{n}_i \omega_{di} \tilde{\varphi} + \omega_{di} \tilde{p}_i) \equiv i(a_w \tilde{\varphi} + b_w \tilde{T}_i)$$

and use

$$\tilde{p}_i = \frac{\hat{p}_i}{\hat{T}_e} \tilde{\varphi}_i + \hat{n}_i \tilde{T}_i.$$

Final form of eigenvalue equation

$$\omega \tilde{\varphi} = A_{11} \tilde{\varphi} + A_{12} \tilde{T}_i + A_{13} \tilde{V}_{i\parallel}$$

$$\omega \tilde{T}_i = A_{21} \tilde{\varphi} + A_{22} \tilde{T}_i + A_{23} \tilde{V}_{i\parallel}$$

$$\omega \tilde{V}_{i\parallel} = A_{31} \tilde{\varphi} + A_{32} \tilde{T}_i + A_{33} \tilde{V}_{i\parallel}$$



$$\omega \mathbf{F} = \mathbf{A} \cdot \mathbf{F}$$

$$\mathbf{F} = \begin{pmatrix} \tilde{\varphi} \\ \tilde{T}_i \\ \tilde{V}_{i\parallel} \end{pmatrix}, \quad \mathbf{A} = \begin{pmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} & \mathbf{A}_{13} \\ \mathbf{A}_{21} & \mathbf{A}_{22} & \mathbf{A}_{23} \\ \mathbf{A}_{31} & \mathbf{A}_{32} & \mathbf{A}_{33} \end{pmatrix}$$

$$\tilde{\varphi} = \sum_k \tilde{\varphi}_k |k\rangle = \begin{pmatrix} \tilde{\varphi}_1 \\ \tilde{\varphi}_2 \\ \tilde{\varphi}_3 \\ \vdots \end{pmatrix}, \quad \dots$$

$|k\rangle$ is a basis function

where

$$A_{11} = M^{-1} \left(-a_w + G_{\hat{p}_i} L + G_{\hat{n}_i} J_0 + i \bar{D}_c M \right), \quad A_{12} = -M^{-1} b_w, \quad A_{13} = M^{-1} \hat{n}_i k_{\parallel}$$

$$A_{21} = G_{\hat{T}_i} J_0, \quad A_{22} = i \bar{D}_c + i \bar{D}_{glf}, \quad A_{23} = \frac{2}{3} \hat{T}_i k_{\parallel}$$

$$A_{31} = \left(1 + \frac{\hat{T}_i}{\hat{T}_e} \right) k_{\parallel}, \quad A_{32} = k_{\parallel}, \quad A_{33} = i \bar{D}_c$$

Basis functions

Field F can be represented as $F(\rho, \theta, \zeta, t) = e^{-i\omega t} \sum_{k=mn p} F_k W_k(\rho) e^{i(m\theta - n\zeta)}$ in concentric - circular geometry :

$$\mathbf{F} = |F\rangle = e^{-i\omega t} \sum_k F_k |k\rangle, \quad |k\rangle = |mnp\rangle = W_{mnp}(\rho) e^{i(m\theta - n\zeta)},$$

inner product :

$$\langle k | F \rangle \equiv \frac{1}{4\pi^2} \int_0^1 \rho d\rho \int_0^{2\pi} d\theta \int_0^{2\pi} d\zeta W_{mnp}(\rho) e^{-i(m\theta - n\zeta)} F, \quad \langle k | k' \rangle = \delta_{kk'}$$

- Radial basis function

For Hermite basis ,

$$W_{mnp}(\rho) = \frac{1}{\sqrt{2\rho w_{mn} \nu_p}} H_p(x) e^{-x^2/2}$$

where $x = \frac{\rho - \rho_{mn}}{w_{mn}} \left(\rho_{mn} \text{ is radial position of rational surface satisfying } q = \frac{m}{n} \right), \quad \nu_p = 2^{p/2} \Gamma(p+1)^{1/2} \pi^{1/4}$

For Bessel basis,

$$W_{mnp}(\rho) = \frac{\sqrt{2}}{J_{m+1}(\alpha_{mp})} J_m(\alpha_{mp} \rho) \quad \text{where } J_m(\alpha_{mp}) = 0$$

Matrix (M, J0, Dc) involving Laplacian L

For Bessel basis,

$$L_{kk'} = \langle k | \tilde{\nabla}_{\perp}^2 | k' \rangle = \rho_*^2 \langle k | \left(\frac{\partial^2}{\partial \rho^2} + \frac{1}{\rho} \frac{\partial}{\partial \rho} - \frac{m'^2}{\rho^2} \right) | k' \rangle = -k_{\perp}^2 \delta_{kk'}$$

where $k_{\perp} = \rho_* \alpha_{mp}$

$$M_{kk'} \approx \left. \frac{\hat{n}_i}{\hat{T}_e} \right|_{\rho=\rho_{mn}} \delta_{kk'} - \hat{n}_i \Big|_{\rho=\rho_{mn}} L_{kk'}$$

$$J_{0,kk'} = \left(1 - \frac{1}{2} L_{kk'} \right)^{-1}$$

$$\overline{D}_{c,kk'} = \overline{\mu}_1 L_{kk'} - \overline{\mu}_2 L_{kk'}^2$$

$$L\tilde{\varphi} = \tilde{\nabla}_{\perp}^2 \tilde{\varphi}$$

$$M\tilde{\varphi} = \left(\frac{\hat{n}_i}{\hat{T}_e} - \hat{n}_i \tilde{\nabla}_{\perp}^2 \right) \tilde{\varphi} = \tilde{\Omega}_i$$

$$J_0 \tilde{\varphi} = \left(1 - \frac{1}{2} \tilde{\nabla}_{\perp}^2 \right)^{-1} \tilde{\varphi} = \tilde{\Psi}$$

$$\overline{D}_c F = \overline{\mu}_1 \tilde{\nabla}_{\perp}^2 F - \overline{\mu}_2 \tilde{\nabla}_{\perp}^4 F$$

Matrix (k_{\parallel} , D_{glf}) involving parallel wavenumber

$$\hat{F}k_{\parallel, kk'} = i\langle k | \hat{F}\hat{V}_{\parallel} | k' \rangle = \delta_{mm'}\delta_{nn'} \frac{a}{R} \int_0^1 \hat{F}(\rho) \left(\frac{m}{q(\rho)} - n \right) W_{mnp}(\rho) W_{mnp'}(\rho) \rho d\rho$$

$$\overline{D}_{glf, kk'} = -\langle k | \sqrt{\frac{8\hat{T}_i}{\pi}} | \hat{V}_{\parallel} | k' \rangle = -\delta_{mm'}\delta_{nn'} \sqrt{\frac{8}{\pi}} \frac{a}{R} \int_0^1 \sqrt{\hat{T}_i(\rho)} \left| \frac{m}{q(\rho)} - n \right| W_{mnp}(\rho) W_{mnp'}(\rho) \rho d\rho$$

Matrix (G) for ExB and diamagnetic drifts

$$G_{\hat{F}} \tilde{\varphi} = i \left[\frac{\hat{F}}{\rho_*}, \tilde{\varphi} \right]:$$

$$G_{\hat{F}, kk'} = i\langle k | \left[\frac{\hat{F}}{\rho_*}, \right] | k' \rangle = i\rho_* \langle k | \frac{1}{\rho} \frac{\partial \hat{F}}{\partial \rho} \frac{\partial}{\partial \theta} | k' \rangle = -\delta_{mm'}\delta_{nn'} \rho_* m \int_0^1 \frac{\partial \hat{F}}{\partial \rho} W_{mnp}(\rho) W_{mnp'}(\rho) d\rho$$

Matrix (a_w, b_w) involving curvature

$$\hat{\beta}\omega_{di}\hat{\alpha}|k'\rangle = -2i\rho_* \frac{a}{R} \hat{\beta} \left(\frac{1}{\rho} \cos \theta \frac{\partial}{\partial \theta} + \sin \theta \frac{\partial}{\partial \rho} \right) \hat{\alpha}|k'\rangle$$

$$= \rho_* \frac{a}{R} \left[\left(\frac{m'}{\rho} \hat{\beta} \hat{\alpha} W_{k'} + \hat{\beta} \frac{\partial \hat{\alpha} W_{k'}}{\partial \rho} \right) e^{i(m'-1)\theta} + \left(\frac{m'}{\rho} \hat{\beta} \hat{\alpha} W_{k'} - \hat{\beta} \frac{\partial \hat{\alpha} W_{k'}}{\partial \rho} \right) e^{i(m'+1)\theta} \right] e^{-in'\zeta}$$

$$\Rightarrow \langle k | \hat{\beta} \omega_{di} \hat{\alpha} | k' \rangle = \rho_* \frac{a}{R} \left[\left\{ (m+1) \int_0^1 \hat{\beta} \hat{\alpha} W_{mnp} W_{m+1,n,p'} d\rho + \int_0^1 \hat{\beta} W_{mnp} \frac{\partial \hat{\alpha} W_{m+1,n,p'}}{\partial \rho} \rho d\rho \right\} \delta_{m+1,m'} \delta_{nn'} \right. \\ \left. + \left\{ (m-1) \int_0^1 \hat{\beta} \hat{\alpha} W_{mnp} W_{m-1,n,p'} d\rho - \int_0^1 \hat{\beta} W_{mnp} \frac{\partial \hat{\alpha} W_{m-1,n,p'}}{\partial \rho} \rho d\rho \right\} \delta_{m-1,m'} \delta_{nn'} \right]$$



$$a_{W,kk'} = \langle k | \left(\hat{n}_i \omega_{di} + \omega_{di} \frac{\hat{p}_i}{\hat{T}_e} \right) | k' \rangle = a_W^+(k, p') \delta_{m+1,m'} \delta_{nn'} + a_W^-(k, p') \delta_{m-1,m'} \delta_{nn'}$$

$$b_{W,kk'} = \langle k | \omega_{di} \hat{n}_i | k' \rangle = b_W^+(k, p') \delta_{m+1,m'} \delta_{nn'} + b_W^-(k, p') \delta_{m-1,m'} \delta_{nn'}$$

$$a_W \tilde{\varphi} = \hat{n}_i \omega_{di} \tilde{\varphi} + \omega_{di} \frac{\hat{p}_i}{\hat{T}_e} \tilde{\varphi}$$

$$b_W \tilde{T}_i = \omega_{di} \hat{n}_i \tilde{T}_i$$

where

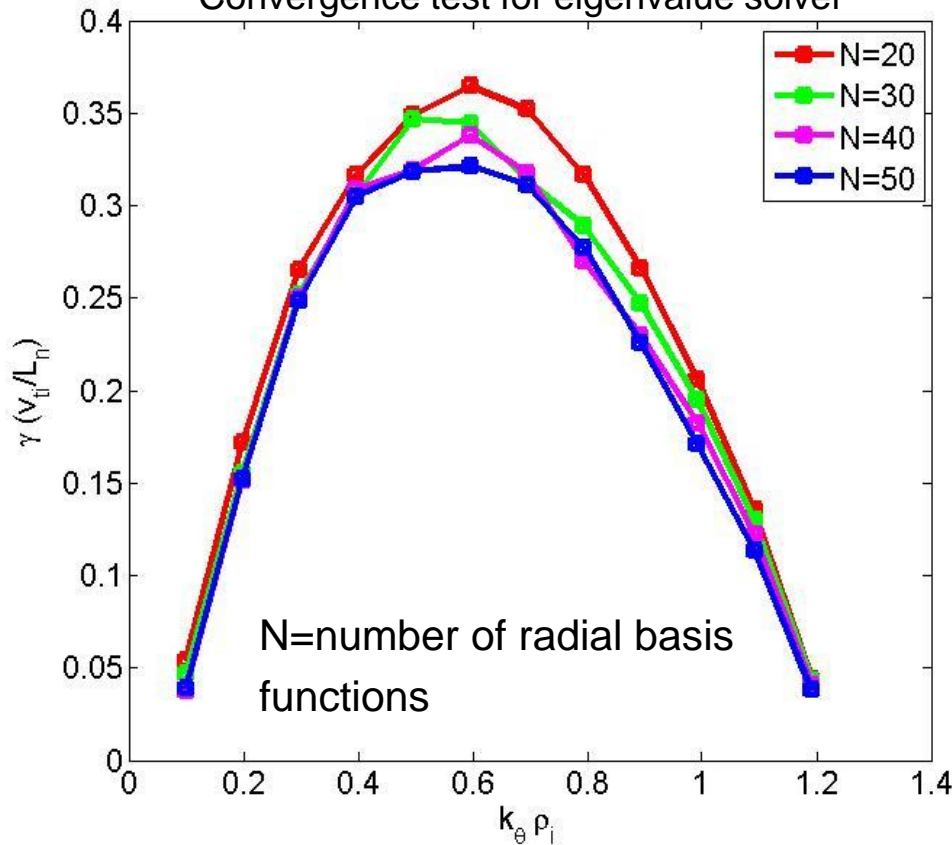
$$a_W^\pm(k, p') = \rho_* \frac{a}{R} \left[\int_0^1 \hat{n}_i \left\{ (m \pm 1)(1 + \tau) \pm \left(\frac{\partial \ln \hat{n}_i}{\partial \rho} + \frac{\partial \ln \tau}{\partial \rho} \right) \tau \right\} W_{mnp} W_{m \pm 1, n, p'} d\rho \pm \int_0^1 \hat{n}_i (1 + \tau) W_{mnp} \frac{\partial W_{m \pm 1, n, p'}}{\partial \rho} \rho d\rho \right]$$

$$b_W^\pm(k, p') = \rho_* \frac{a}{R} \left[\int_0^1 \left(m \pm 1 + \frac{\partial \ln \hat{n}_i}{\partial \rho} \right) \hat{n}_i W_{mnp} W_{m \pm 1, n, p'} d\rho \pm \int_0^1 \hat{n}_i W_{mnp} \frac{\partial W_{m \pm 1, n, p'}}{\partial \rho} \rho d\rho \right], \quad \tau = \frac{\hat{T}_i}{\hat{T}_e}$$

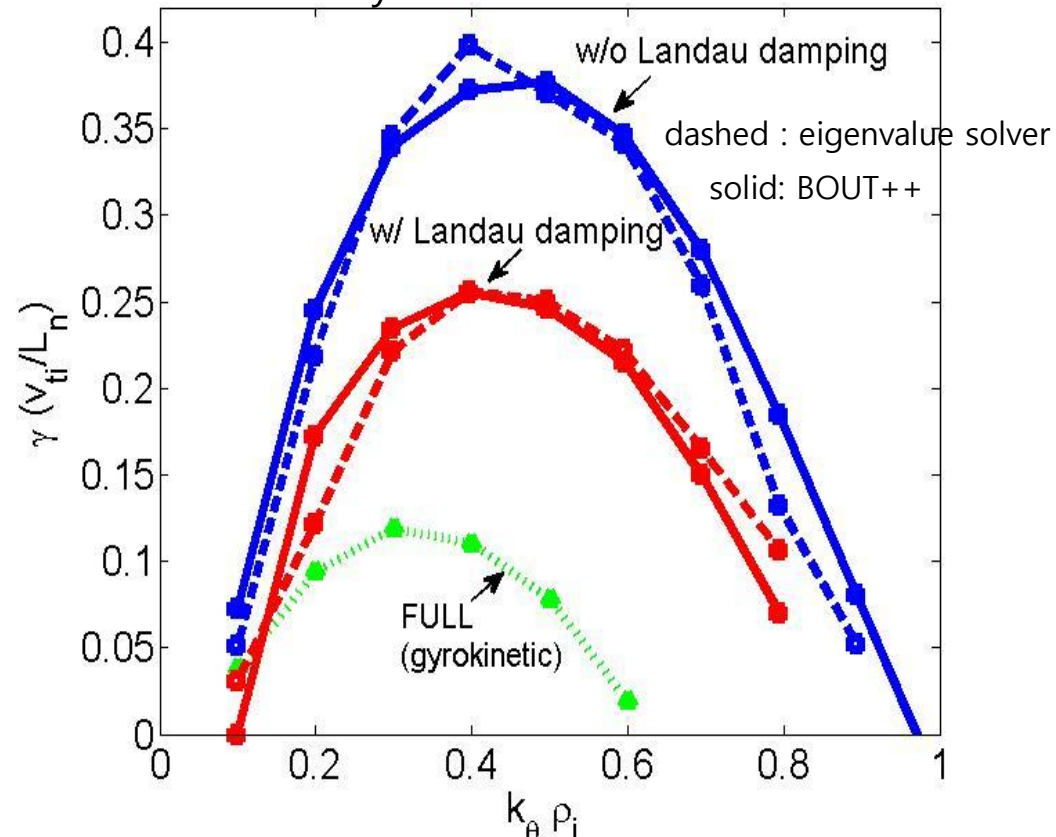
Comparison of BOUT++ with eigenvalue solver

- In eigenvalue solver, $W_{mp} = \frac{\sqrt{2}}{J_{m+1}(\alpha_{mp})} J_m\left(\frac{\alpha_{mp}}{a} r\right) e^{i(m\theta - n\zeta)}$ ($p=1 \sim N$) is used as basis function.
- The fluid equations are then projected on to the set of basis functions.
- Eigenvalues are obtained by using matlab.
- BOUT++ results agree well with eigenvalues, which means BOUT++ correctly solves the given equations.

Convergence test for eigenvalue solver



Cyclone base case



**Gyrofluid Simulation using
BOUT++ GLF Code
in Core Region**

Gyrofluid equation for ITG

(M. A. Beer and G. W. Hammett, PoP '96)

- Guiding center density equation**

$$\frac{\partial \tilde{n}_i}{\partial t} + \mathbf{V}_{\tilde{\Phi}} \cdot \nabla \tilde{n}_i + \frac{n_0}{2T_{i0}} [\hat{\mathbf{V}}_{\perp}^2 \mathbf{V}_{\tilde{\Phi}}] \cdot \nabla \tilde{T}_{i\perp} + B \nabla_{\parallel} \frac{n_0 \tilde{V}_{i\parallel}}{B} - \frac{en_0}{T_{i0}} \left(1 + \frac{1}{2} \eta_i \hat{\mathbf{V}}_{\perp}^2 \right) i \omega_* \tilde{\Phi} + \frac{en_0}{T_{i0}} \left(2 + \frac{1}{2} \hat{\mathbf{V}}_{\perp}^2 \right) i \omega_d \tilde{\Phi} + \frac{1}{T_{i0}} i \omega_d (\tilde{p}_{i\parallel} + \tilde{p}_{i\perp}) = 0$$

- Guiding center parallel velocity equation**

$$\frac{\partial \tilde{V}_{i\parallel}}{\partial t} + \mathbf{V}_{\tilde{\Phi}} \cdot \nabla \tilde{V}_{i\parallel} + \frac{e}{m_i} \nabla_{\parallel} \tilde{\Phi} + \frac{B}{m_i n_0} \nabla_{\parallel} \frac{\tilde{p}_{i\parallel}}{B} + \left(\frac{\tilde{T}_{i\perp}}{m_i} + \frac{e}{2m_i} \hat{\mathbf{V}}_{\perp}^2 \tilde{\Phi} \right) \nabla_{\parallel} \ln B + \frac{1}{p_{i0}} i \omega_d (\tilde{q}_{i\parallel} + \tilde{q}_{i\perp}) + 4i \omega_d \tilde{V}_{i\parallel} = 0$$

- Guiding center parallel & perpendicular pressure equation**

$$\begin{aligned} \frac{\partial \tilde{p}_{i\parallel}}{\partial t} + \mathbf{V}_{\tilde{\Phi}} \cdot \nabla \tilde{p}_{i\parallel} + \frac{n_0}{2} [\hat{\mathbf{V}}_{\perp}^2 \mathbf{V}_{\tilde{\Phi}}] \cdot \nabla \tilde{T}_{i\perp} + B \nabla_{\parallel} \frac{1}{B} (\tilde{q}_{i\parallel} + p_{i0} \tilde{V}_{i\parallel}) + 2p_{i0} B \nabla_{\parallel} \frac{\tilde{V}_{i\parallel}}{B} + 2\tilde{q}_{i\perp} \nabla_{\parallel} \ln B \\ - en_0 \left(1 + \eta_i + \frac{1}{2} \eta_i \hat{\mathbf{V}}_{\perp}^2 \right) i \omega_* \tilde{\Phi} + en_0 \left(4 + \frac{1}{2} \hat{\mathbf{V}}_{\perp}^2 \right) i \omega_d \tilde{\Phi} + 4i \omega_d \tilde{p}_{i\parallel} + n_0 i \omega_d (3\tilde{T}_{i\parallel} + \tilde{T}_{i\perp}) + 2n_0 |\omega_d| (\nu_1 \tilde{T}_{i\parallel} + \nu_2 \tilde{T}_{i\perp}) = 0 \\ \frac{\partial \tilde{p}_{i\perp}}{\partial t} + \mathbf{V}_{\tilde{\Phi}} \cdot \nabla \tilde{p}_{i\perp} + \frac{1}{2} [\hat{\mathbf{V}}_{\perp}^2 \mathbf{V}_{\tilde{\Phi}}] \cdot \nabla \tilde{p}_{i\perp} + n_0 \left[\hat{\mathbf{V}}_{\perp}^2 \mathbf{V}_{\tilde{\Phi}} \right] \cdot \nabla \tilde{T}_{i\perp} + B^2 \nabla_{\parallel} \frac{1}{B^2} (\tilde{q}_{i\perp} + p_{i0} \tilde{V}_{i\parallel}) \\ - en_0 \left[1 + \frac{1}{2} \hat{\mathbf{V}}_{\perp}^2 + \eta_i \left(1 + \frac{1}{2} \hat{\mathbf{V}}_{\perp}^2 + \hat{\mathbf{V}}_{\perp}^2 \right) \right] i \omega_* \tilde{\Phi} + en_0 \left(3 + \frac{3}{2} \hat{\mathbf{V}}_{\perp}^2 + \hat{\mathbf{V}}_{\perp}^2 \right) i \omega_d \tilde{\Phi} + 3i \omega_d \tilde{p}_{i\perp} + n_0 i \omega_d (\tilde{T}_{i\parallel} + 2\tilde{T}_{i\perp}) + 2n_0 |\omega_d| (\nu_3 \tilde{T}_{i\parallel} + \nu_4 \tilde{T}_{i\perp}) = 0 \end{aligned}$$

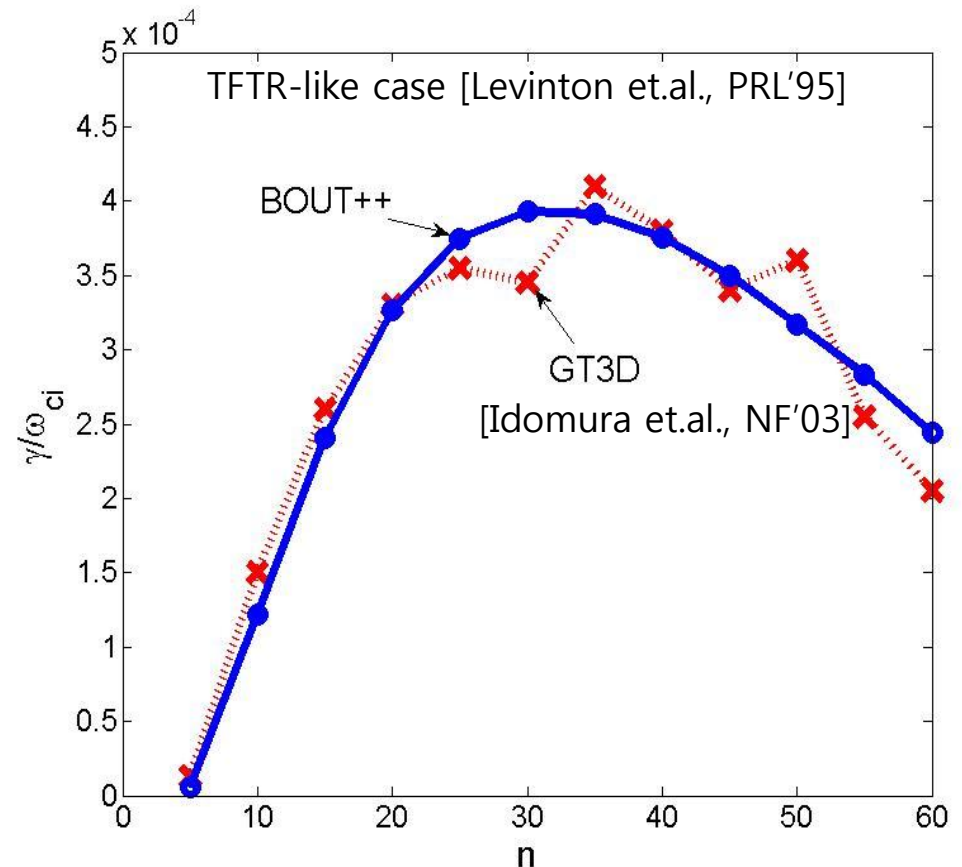
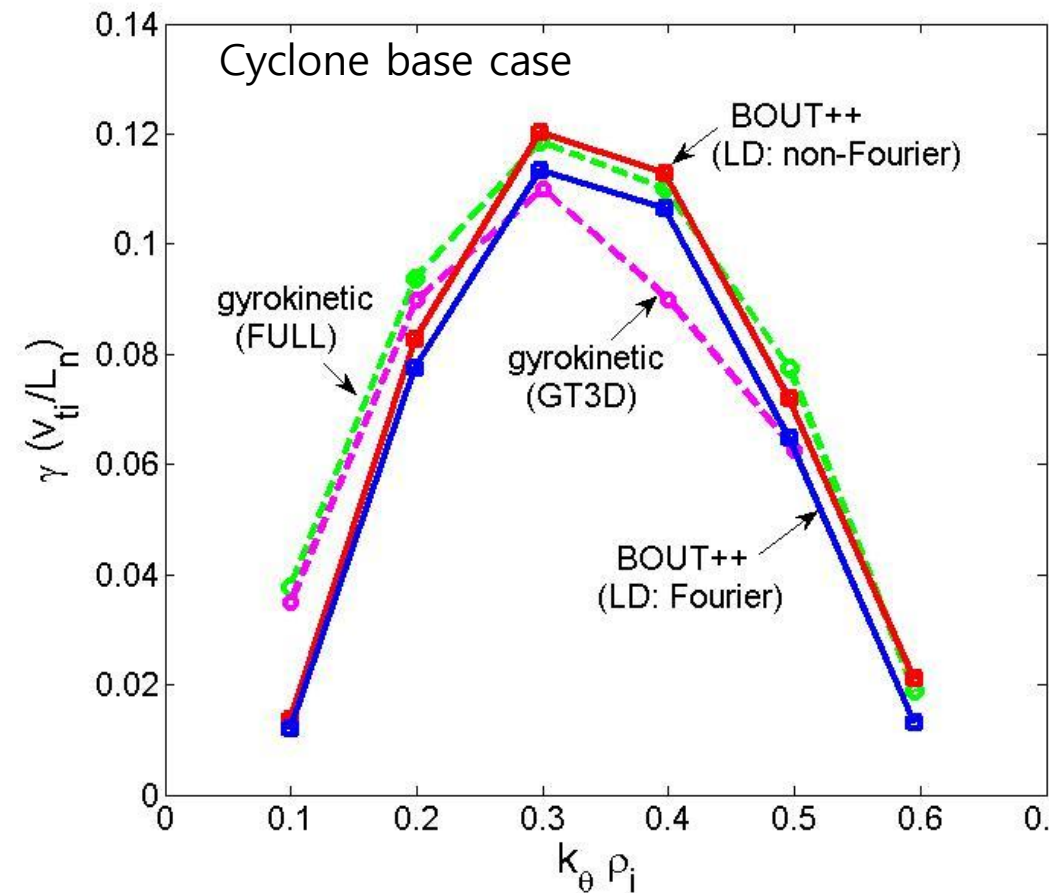
- Quasi-neutrality**

$$\frac{n_0}{T_{e0}} \tilde{\varphi} + \frac{n_0}{T_{i0}} (1 - \Gamma_0) \tilde{\varphi} = \Gamma_0^{1/2} \tilde{n}_i + \frac{n_0}{T_{i0}} b \frac{\partial \Gamma_0^{1/2}}{\partial b} \tilde{T}_i \quad \text{where } \tilde{\Phi} = \Gamma_0^{1/2} \tilde{\varphi}, \quad \frac{1}{2} \hat{\mathbf{V}}_{\perp}^2 \tilde{\Phi} = \left(b \frac{\partial \Gamma_0^{1/2}}{\partial b} \right) \tilde{\varphi}, \quad \hat{\mathbf{V}}_{\perp}^2 \tilde{\Phi} = b \frac{\partial^2}{\partial b^2} (b \Gamma_0^{1/2}) \tilde{\varphi}, \quad \Gamma_0^{1/2} \equiv \frac{1}{1 + b/2}$$

$$i \omega_d = (\nu_t^2 / \omega_c) \mathbf{b} \times \nabla \ln B \cdot \nabla, \quad b = -\rho_i^2 \nabla_{\perp}^2$$

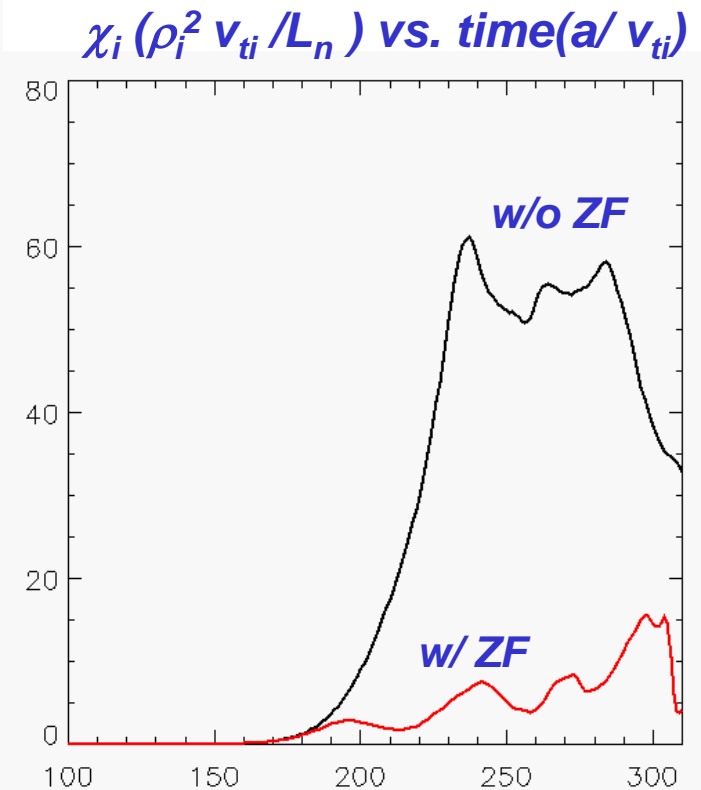
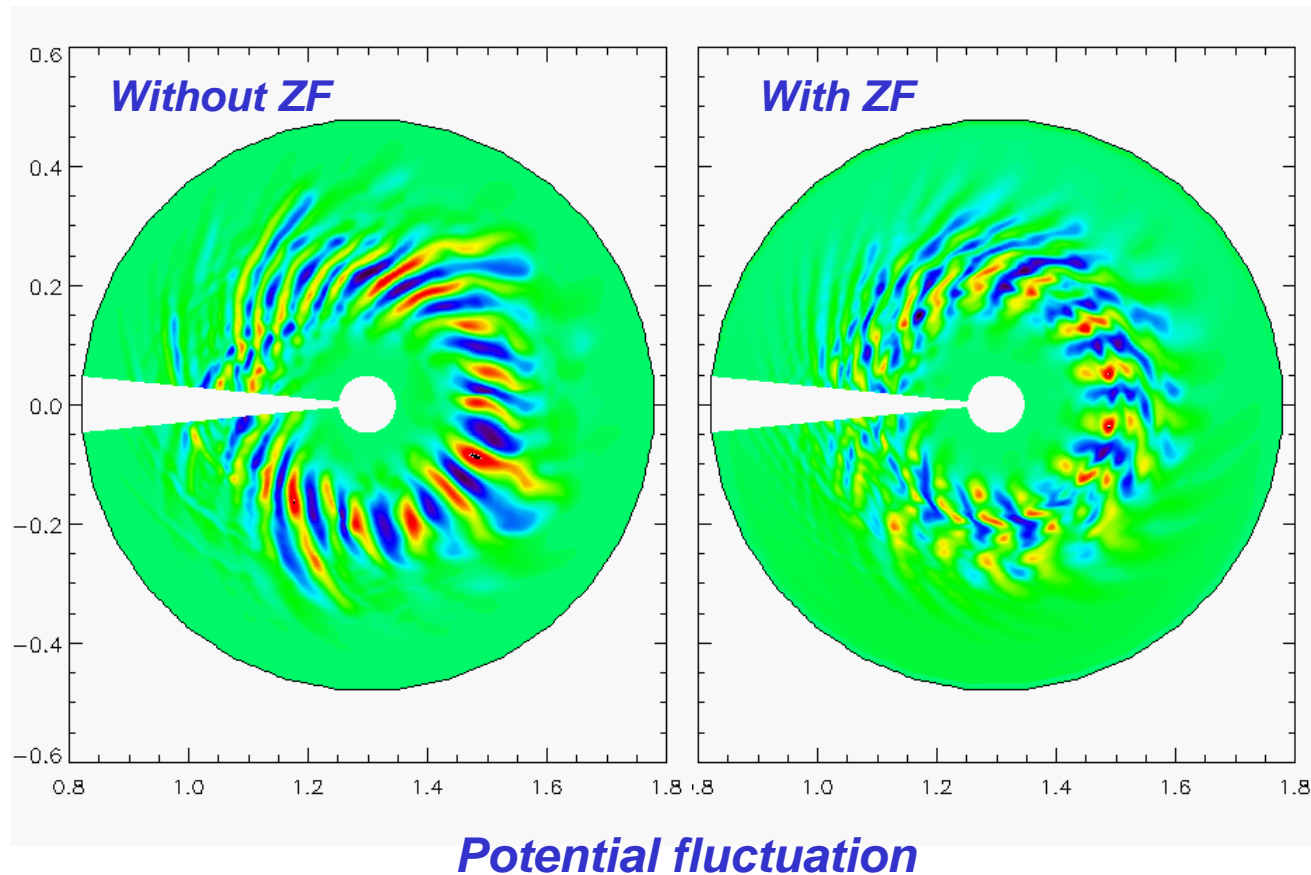
Linear benchmarks

- BOUT++ using Beer's "3+1" field model agrees well with gyrokinetic codes.
- Non-Fourier method for Landau damping shows good agreement with Fourier method.



Nonlinear simulations of ITG

- Global nonlinear simulations using Beer model performed at fixed profile
- Turbulence suppression by zonal flow observed



Hands-on Exercise for Eigenvalue Solver using MATLAB

0) Run matlab :

> module load matlab-nofonts

> matlab

1) Run eigensolver_init.m and eigensolver_ITG.m in matlab:

>> eigensolver_init : assign equilibrium profiles, calculate radial basis functions, arrange mode index, evaluate matrix elements involving integrals

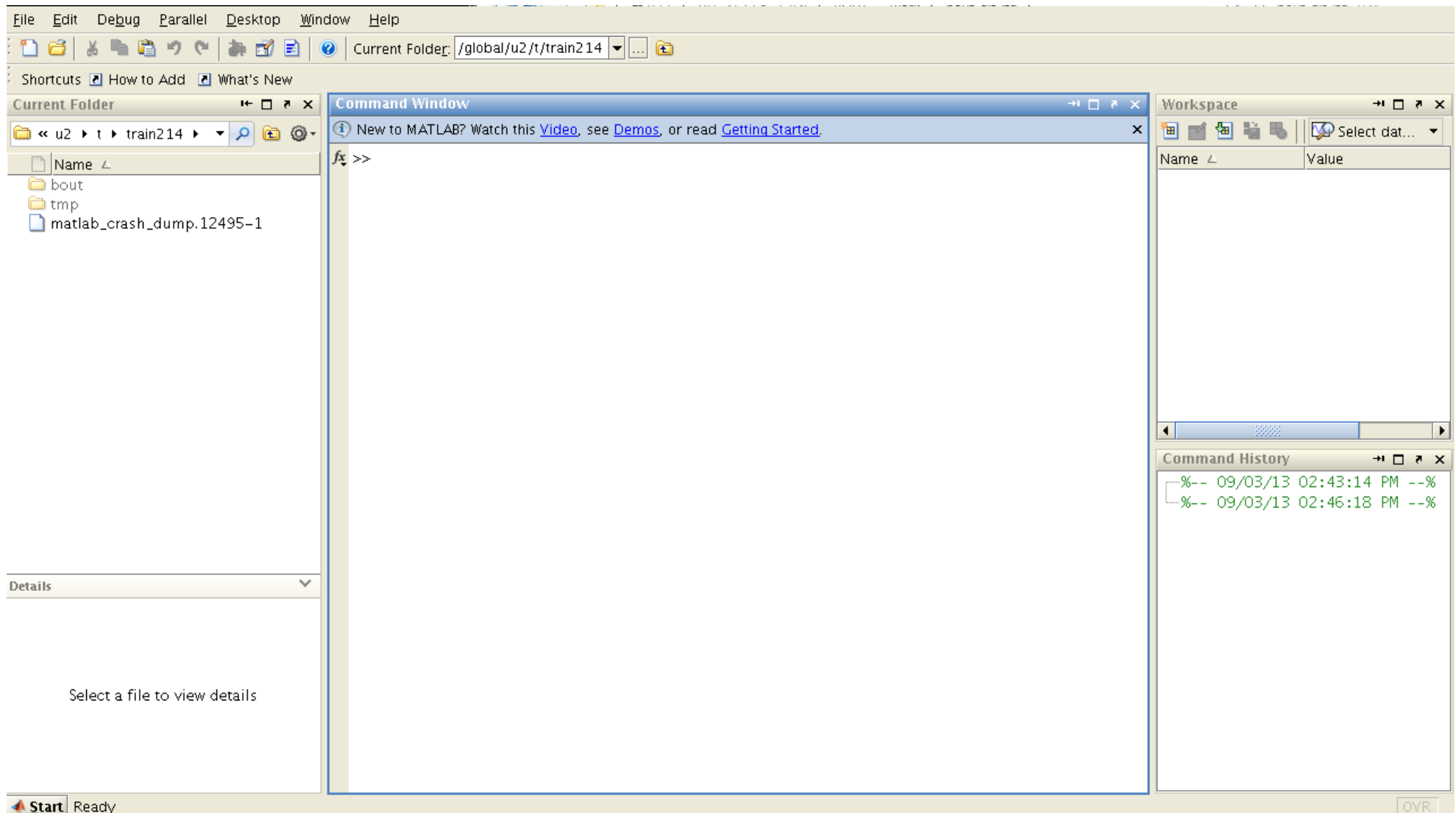
>> eigensolver_ITG : set up matrix and solve the eigenvalue equation for ITG

2) Run plot_eigenvalue.m and plot_eigenmode.m in matlab:

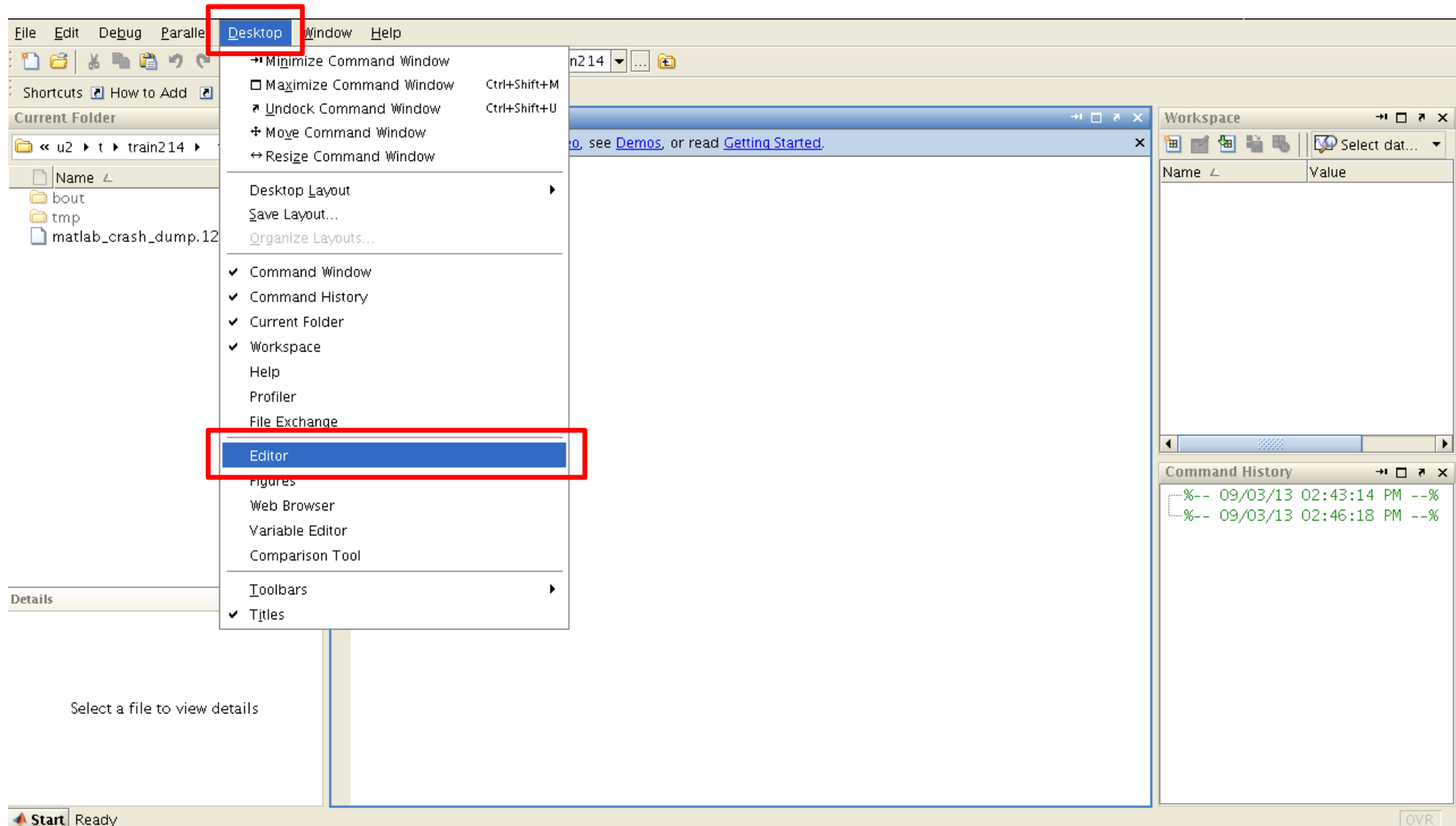
>> plot_eigenvalue : plot linear ITG growth rate and real frequency vs. $k_{\theta} \rho_i$

>> plot_eigenmode : choose toroidal mode number and plot eigenmode structure

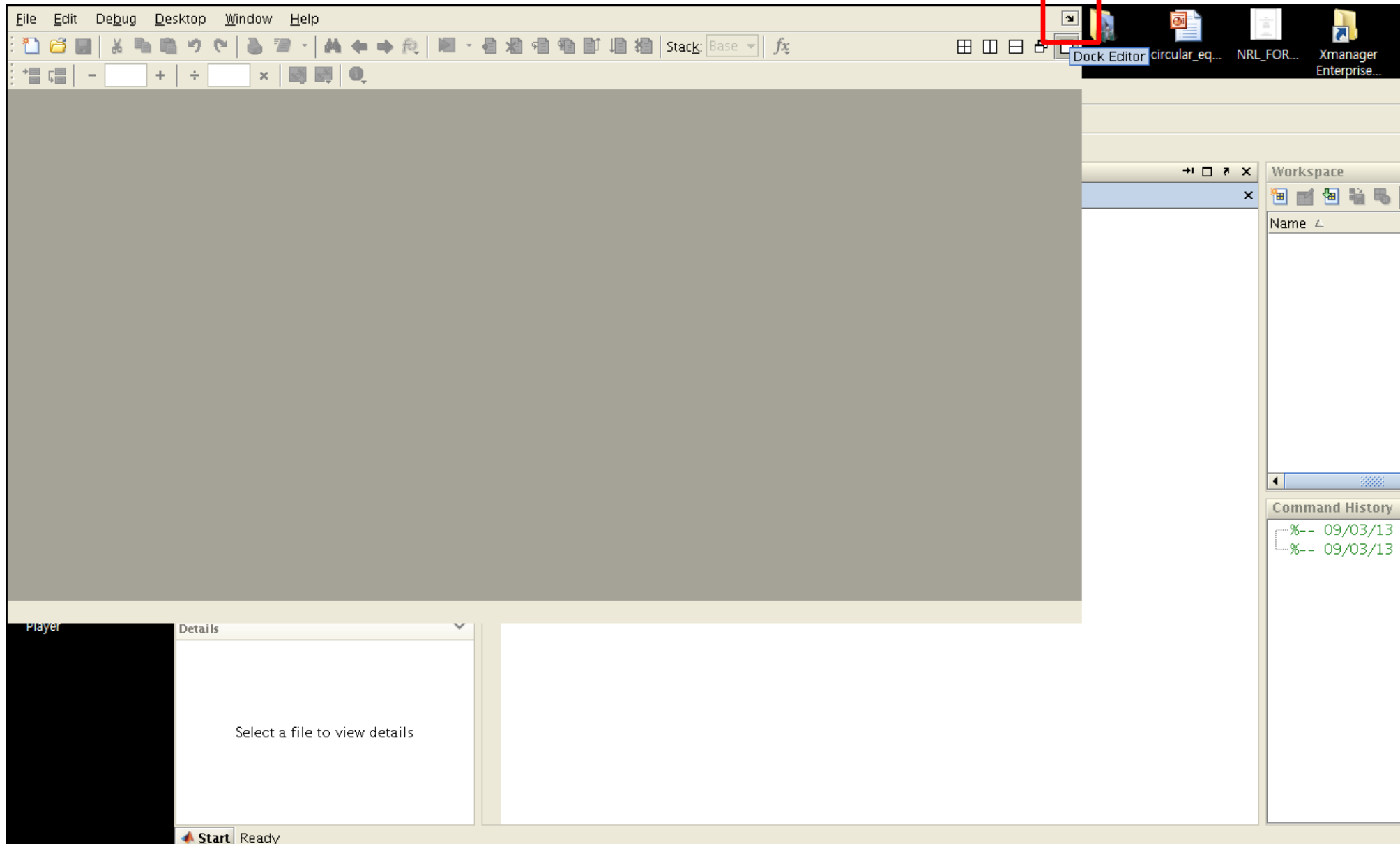
Run matlab : > module load matlab-nofonts
 > matlab

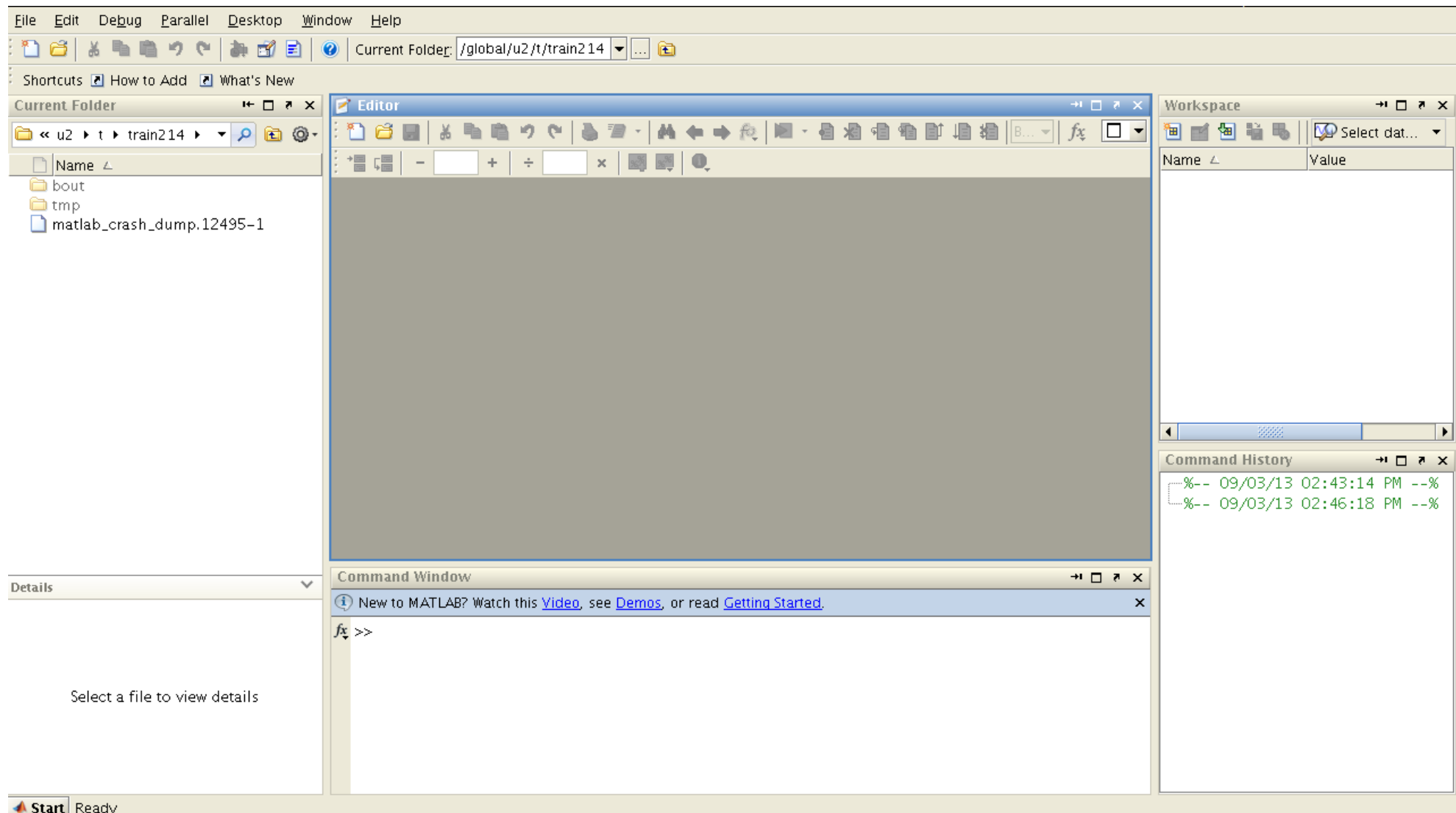


Run Editor :

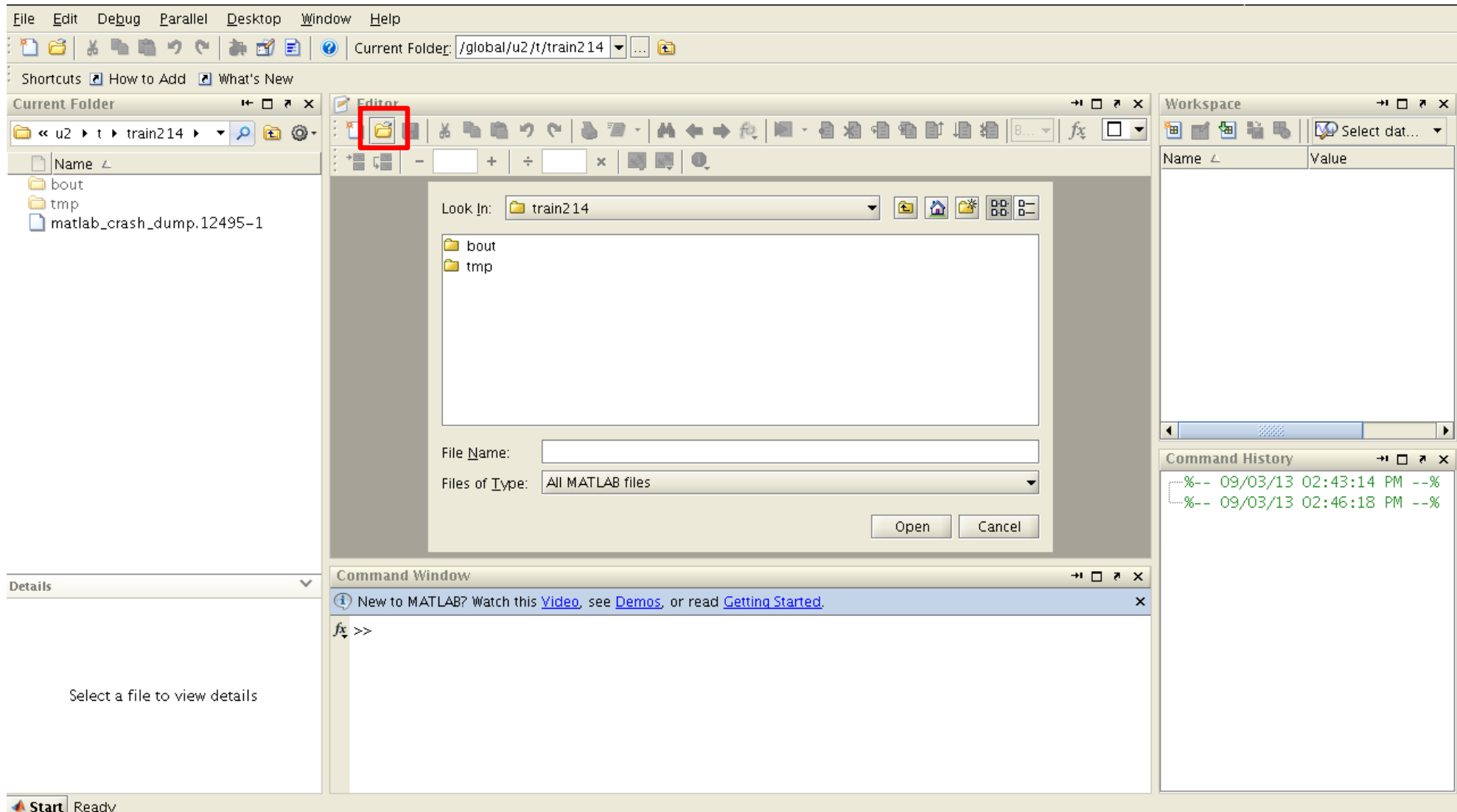


Dock Editor: Push this button





Open eigensolver_init.m, eigensolver_ITG.m, plot_eigenvalue.m, plot_eigenmode.m in tools/eigenvalue_solver.



Run eigensolver_init.m;

The image shows the MATLAB R2012a environment. The main window is the Editor, displaying the script `eigensolver_init.m` located at `/global/u1/s/sskim/bout/tools/eigenvalue_solver/eigensolver_init.m`. The script contains comments and code for preprocessing an eigenvalue equation solver. A red box highlights the Run button (a green play icon) in the Editor's toolbar. Below the toolbar, a button labeled "Run eigensolver_init.m (F5)" is visible. The Command Window at the bottom shows the prompt `>>`. The Workspace panel on the right shows the current workspace contents, and the Command History panel shows the sequence of commands executed.

Current Folder: /global/u1/s/sskim

Editor - /global/u1/s/sskim/bout/tools/eigenvalue_solver/eigensolver_init.m

```
1 %=====
2 % Preprocess for eigenvalue equation solver
3 % Calculate radial basis functions, arrange mode index,
4 % and evaluate matrix elements involving integrals.
5 % This part was developed based on
6 % TRB code [Ref. X. Garbet et.al., Phys. Plasmas 8, 2793 (2001)].
7 %=====
8
9 jmax=256; % number of radial grids
10 nmax=15; % maximum of toroidal mode number
11 nmin=5; % minimum of toroidal mode number
12 ndel=5; % toroidal mode number spacing
13 mmax=150; % maximum of poloidal mode number
14 mmin=1; % minimum of poloidal mode number
15 mdel=1; % poloidal mode number spacing
16 kyrhoscute=1.3; % maximal ktheta*rhoi allowed in the calculation
17 xmu1=0.1; % viscous damping coefficient
18 xmu2=0.0; % hyper-viscous damping coefficient
19 r0=0.2; % inner boundary
20 r1=0.9; % outer boundary
21
22 hermite=1;
23 basis=bessel;
24 % choose radial basis function
```

Command Window

New to MATLAB? Watch this [Video](#), see [Demos](#), or read [Getting Started](#).

`>>`

Workspace

Name	Size
g_g1f1_4	[0.0000 0.0000]
xn	[10;15;20]
g_g1f	[g_g1f1_4;g_g1f2_4;g_g1f3_4]
R_Ln	2.22
q	1.4
r	0.5
rhos0	7.0860
rhoi	rhos0
x	xn*q/r*
gamfactor	R_Ln
g_g1f	gamfactor
plot(x,g_g1f)	

Command History

- plot(x,g_g1f)
- %-- 05/02/2012
- %-- 05/02/2012
- g_g1f1_4=[0.0000 0.0000]
- xn=[10;15;20]
- g_g1f=g_g1f1_4;g_g1f2_4;g_g1f3_4
- R_Ln=2.22; R_Ln=2.22
- q=1.4; r=0.5
- rhos0=7.0860
- rhoi=rhos0
- x=xn*q/r*rhoi
- gamfactor=R_Ln
- g_g1f=gamfactor
- plot(x,g_g1f)
- %-- 09/03/13 0
- eigensolver_
- %-- 09/03/13 0
- %-- 09/03/13 0
- %-- 09/03/13 0
- %-- 09/03/13 0

```
1 %=====
2 % Preprocess for eigenvalue equation solver
3 % Calculate radial basis functions, arrange mode index,
4 % and evaluate matrix elements involving integrals.
5 % This part was developed based on
6 % TRB code [Ref. X. Garbet et.al., Phys. Plasmas 8, 2793 (2001)].
7 %=====
8
9 jmax=256; % number of radial grids
10 nmax=15; % maximum of toroidal mode number
11 nmin=5; % minimum of toroidal mode number
12 ndel=5
13 mmax=1
14 mmin=1
15 mdel=1
16 kyrhos
17 xmu1=0
18 xmu2=0
19 r0=0.2
20 r1=0.9
21
22 hermit
23 besse1=2;
24 basis=besse1; % choose radial basis function
```

MATLAB Editor

File /global/...genvalue_solver/eigensolver_init.m is not found
in the current folder or on the MATLAB path.

To run this file, you can either change the MATLAB current folder or add its
folder to the MATLAB path.

Change Folder

Add to Path

Cancel

Help

Select a file to view details

```
plot(x,g_g1f1_4)
%-- 05/02/2012
%-- 05/02/2012
g_g1f1_4=[0;
xn=[10;15;20;
g_g1f=g_g1f1_4;
R_Ln=2.22; F
q=1.4; r=0.5
rhos0=7.0860
rhoi=rhos0;
x=xn*q/r*rho
gam factor=R
g_g1f=gam fac
plot(x,g_g1f1_4)
%-- 09/03/13 0
eigensolver.
%-- 09/03/13 0
%-- 09/03/13 0
%-- 09/03/13 0
```

Running eigensolver_init.m ... : it takes about 5 min.

The image shows the MATLAB R2012a interface with the following components:

- File Explorer:** Shows the current folder structure: `/global/u1/s/sskim/bout/tools/eigenvalue_solver`. The files listed are `eigensolver_init.m`, `eigensolver_ITG.m`, `plot_eigenmode.m`, `plot_eigenvalue.m`, and `README`.
- Editor:** Displays the script `eigensolver_init.m`. The code includes comments and parameter assignments:

```
1 %=====
2 % Preprocess for eigenvalue equation solver
3 % Calculate radial basis functions, arrange mode index,
4 % and evaluate matrix elements involving integrals.
5 % This part was developed based on
6 % TRB code [Ref. X. Garbet et.al., Phys. Plasmas 8, 2793 (2001)].
7 %=====
8
9 jmax=256; % number of radial grids
10 nmax=15; % maximum of toroidal mode number
11 nmin=5; % minimum of toroidal mode number
12 ndel=5; % toroidal mode number spacing
13 mmax=150; % maximum of poloidal mode number
14 mmin=1; % minimum of poloidal mode number
15 mdel=1; % poloidal mode number spacing
16 kyrhoscute=1.3; % maximal ktheta*rhoi allowed in the calculation
17 xmu1=0.1; % viscous damping coefficient
18 xmu2=0.0; % hyper-viscous damping coefficient
19 r0=0.2; % inner boundary
20 r1=0.9; % outer boundary
21
22 hermite=1;
23 besse1=2;
24 basis=bessel; % choose radial basis function
```
- Command Window:** Shows the execution progress:

```
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Arranging mode index
status =
Calculating radial basis functions:
status =
... finding zeros of Bessel functions
```
- Workspace:** Lists the variables created during the execution, including `g_glf1_4`, `xn`, `R_Ln`, `q`, `rhos0`, `rhoi`, `gam factor`, `g_glf`, and `plot(x, g_glf)`.

File Edit Text Go Cell Tools Debug Parallel Desktop Window Help

Current Folder: /global/u1/s/sskim/hout/tools/eigenvalue_solver

File Edit View Insert Tools Desktop Window Help

Shortcuts How to Add What's New

Current Folder

<< tools >> eigenvalue_solver

Name

eigensolver_init.m
eigensolver_ITG.m
plot_eigenmode.m
plot_eigenvalue.m
README

Editor - /global/u1/s/sskim/hout/tools/eigenvalue_solver

1.0

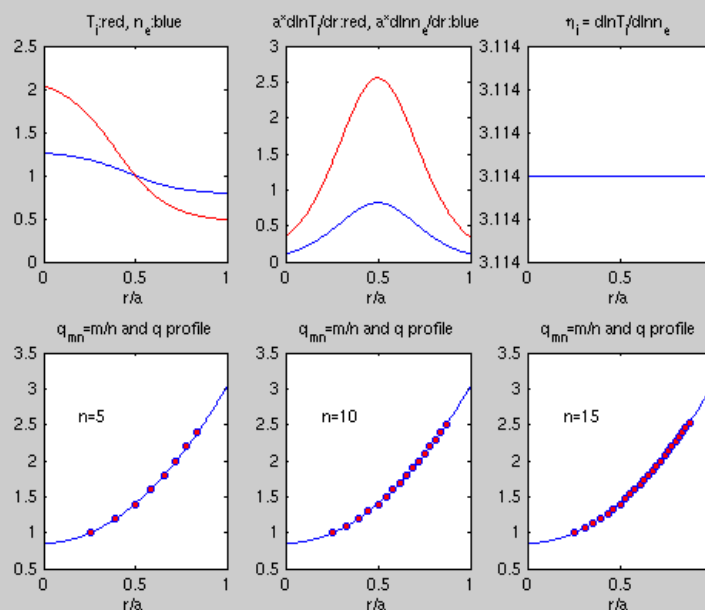
```
1 %=====  
2 % Prepro  
3 % Calcul  
4 % and ev  
5 % This p  
6 % TRB co  
7 %=====  
8  
9 j m  
10 n m  
11 n m  
12 n d  
13 m m  
14 m m  
15 m d  
16 k y  
17 x m  
18 x m  
19 r O  
20 r l  
21  
22 h e  
23 b e  
24 b a  
basis=bessel; % choose radial basis function
```

eigensolver_init.m

eigensolver_ITG.m

plot_eigenmode.m

plot_eigenvalue.m



Workspace

Name

Name	Value
R_Lne	2.22
R_Lte	6.0
R_Lti	6.0
Te_T0	1.0
akpar	<
akpar_enhat	<
akpar_tau	<
akpar_that	<
alphaj	2.0
alphamp	<
amw	<
amw0	0.0
anorm	0.0
apw	<
apw0	0.0
ar	<

Command History

```
%-- 05/02/2012 (
g_glf1_4=[0.1
xn=[10;15;20;
g_glf=g_glf1_
R_Ln=2.22; R=
q=1.4; r=0.5;
rhoi=rhos0;
rhoi=rhos0;
x=xn*q/r*rhoi
gamfactor=R_L
g_glf=gamfact
plot(x,g_glf)
%-- 09/03/13 01
eigensolver_i
%-- 09/03/13 02
%-- 09/03/13 02
%-- 09/03/13 02
eigensolver_i
%-- 09/03/13 03
eigensolver_i
```

Command Window

New to MATLAB? Watch this [Video](#), see [Demos](#), or read [Getting Started](#).

Calculating viscous damping operator

status =

Calculating magnetic curvature terms

status =

***** Initialization done *****

>>

Details

Select a file to view details

Start

Run eigensolver_ITG.m;

The screenshot shows the MATLAB IDE interface with the following components:

- File Explorer (Left):** Displays the current folder structure: `tools > eigenvalue_solver`. Files listed include `eigensolver_init.m`, `eigensolver_ITG.m`, `plot_eigenmode.m`, `plot_eigenvalue.m`, and `README`.
- Editor (Center):** Shows the script `eigensolver_ITG.m` with the following code:

```
1 %=====
2 % Solve eigenvalue equation for ITG :
3 % Ottaviani model used [M. Ottaviani et.al., Phys. Plasmas 6, 3267 (1999)]
4 %=====
5
6 gyroaverage=1; % turn-on(1) or off(0) gyroaverage
7 damping_glf=1; % turn-on(1) or off(0) Landau damping term
8 damping_c=0; % turn-on(1) or off(0) viscous damping term
9
10 l_k=0*ones(1,kmax);
11 l_max=0*ones(1,nmax/ndel);
12
13 l=0;
14 for k=1:kmax
15     if(nfb(k)==nmax)
16         l=l+1;
17     end
18 end
19 l_max=l;
20 Fphi=0*ones(nmax/ndel,l_max);
21
22 status='***** Start eigenvalue solver *****'
23 %=====
24 for n=nmin:ndel:nmax % toroidal mode number scan
```

The **Run** button (a green play icon) in the Editor toolbar is highlighted with a red rectangle. Below the toolbar, the text `Run eigensolver_ITG.m (F5)` is visible.

- Command Window (Bottom):** Displays the execution progress:

```
Calculating viscous damping operator

status =

Calculating magnetic curvature terms

status =

***** Initialization done *****

fx >>
```
- Workspace (Right):** Lists the variables in the workspace, including `R_Lne`, `R_Lte`, `R_Lti`, `Te_T0`, `akpar`, `akpar_enhat`, `akpar_tau`, `akpar_that`, `alpha_j`, `alphamp`, `amw`, `amw0`, `anorm`, `apw`, `apw0`, and `ar`.
- Command History (Bottom Right):** Shows the sequence of commands executed, including the `Run eigensolver_ITG.m` command.

Running eigensolver_ITG.m ... : it takes about 10 min.

The screenshot displays the MATLAB IDE interface. The main editor window shows the script `eigensolver_ITG.m` with the following code:

```
1 %=====
2 % Solve eigenvalue equation for ITG :
3 % Ottaviani model used [M. Ottaviani et.al., Phys. Plasmas 6, 3267 (1999)]
4 %=====
5
6 gyroaverage=1; % turn-on(1) or off(0) gyroaverage
7 damping_glf=1; % turn-on(1) or off(0) Landau damping term
8 damping_c=0; % turn-on(1) or off(0) viscous damping term
9
10 l_k=0*ones(1,kmax);
11 l_maxn=0*ones(1,nmax/ndel);
12
13 l=0;
14 for k=1:kmax
15     if(nfb(k)==nmax)
16         l=l+1;
17     end
18 end
19 l_max=1;
20 Fphi=0*ones(nmax/ndel,l_max);
21
22 status='***** Start eigenvalue solver *****'
23 %-----
24 for n=nmin:ndel:nmax % toroidal mode number scan
```

The Command Window shows the execution progress:

```
Calculating matrix for toroidal coupling via magnetic curvature
status =
Setting up matrix for eigenvalue equation
status =
Calculating eigenvalues and eigenvectors
```

The Workspace window on the right lists variables: `R_Lne`, `R_Lte`, `R_Lti`, `Te_T0`, `akpar`, `akpar_enhat`, `akpar_tau`, `akpar_that`, `alphaj`, `alphamp`, `amw`, `amw0`, `anorm`, `apw`, `apw0`, and `ar`.

The Command History window shows the following commands:

```
k1=1;
clear rss1 qss1
for k=1:kmax
    if(nfb(k)==n1)
        rss1(k1)=rss1(k1);
        qss1(k1)=qss1(k1);
        k1=k1+1;
    end
end
plot(rss1,qss1)
% for
% i
% e
% enc
hold off
end
%-- 09/03/2013
eigensolver_i
eigensolver_I
```

Run plot_eigenvalue.m;

File Edit Text Go Cell Tools Debug Parallel Desktop Window Help

Current Folder: /global/u1/s/sskim/bout/tools/eigenvalue_solver

Shortcuts How to Add What's New

Current Folder

<< tools >> eigenvalue_solver

Name
eigensolver_init.m
eigensolver_ITG.m
plot_eigenmode.m
plot_eigenvalue.m
README

Editor - /global/u1/s/sskim/bout/tools/eigenvalue_solver/plot_eigenvalue.m

Run plot_eigenvalue.m (F5)

```
1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2 % plot linear growth rate and real frequency vs. ktheta*rhoi
3 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
4 clear ktheta_rhoi
5 qval=1.4;
6 rval=0.5;
7 for n1=nmin:ndel:nmax
8     ktheta_rhoi(n1/ndel)=n1*qval/rval*rhos0;
9 end
10 R=1.3; a=0.48; % for cyclone base case
11 gamfactor=R_Lne*a/R; % change time unit from a/v_{ti} to L_{ne}/v_{ti}
12 omgfactor=gamfactor*4;
13
14 figure(2)
15 plot(ktheta_rhoi,gamlin/gamfactor,'-or');
16 xlabel('k_{\theta} \rho_{i}')
17 ylabel('\gamma and \omega/4 (v_{ti}/L_{ne})')
18 title('Linear growth rate and real frequency')
19 hold on
20 plot(ktheta_rhoi,omglin/omgfactor,'-ob');
21
```

eigensolver_init.m x eigensolver_ITG.m x plot_eigenmode.m x plot_eigenvalue.m x

Command Window

New to MATLAB? Watch this [Video](#), see [Demos](#), or read [Getting Started](#).

Setting up matrix for eigenvalue equation

status =

Calculating eigenvalues and eigenvectors

status =

***** Eigenvalue solver finished *****

>>

Workspace

Name	Value
Fphi	<
R_Lne	2.0
R_Lte	6.0
R_Lti	6.0
Te_T0	1.0
akpar	<
akpar_enhat	<
akpar_tau	<
akpar_that	<
alpha_j	2.0
alphamp	<
amw	<
amw0	0.0
anorm	0.0
apw	<
apw0	0.0

Command History

```
k1=1;
clear rssl qss
for k=1:kmax
    if(nfb(k)==n1)
        rssl(k1)=rssl(k);
        qss(k1)=qss(k);
        k1=k1+1;
    end
    plot(rssl,qss)
    % for
    % i
    % E
    % end
    hold off
end
%-- 09/03/2013
eigensolver_i
eigensolver_I
```

Details

Select a file to view details

Start

File Edit Text Go Cell Tools Debug Parallel Desktop Window Help

Current Folder: /global/u1/s/sskim/hnut/tools/eigenvalue_solver

File Edit View Insert Tools Desktop Window Help

Shortcuts How to Add What's New

Current Folder

<< tools >> eigenvalue_solver

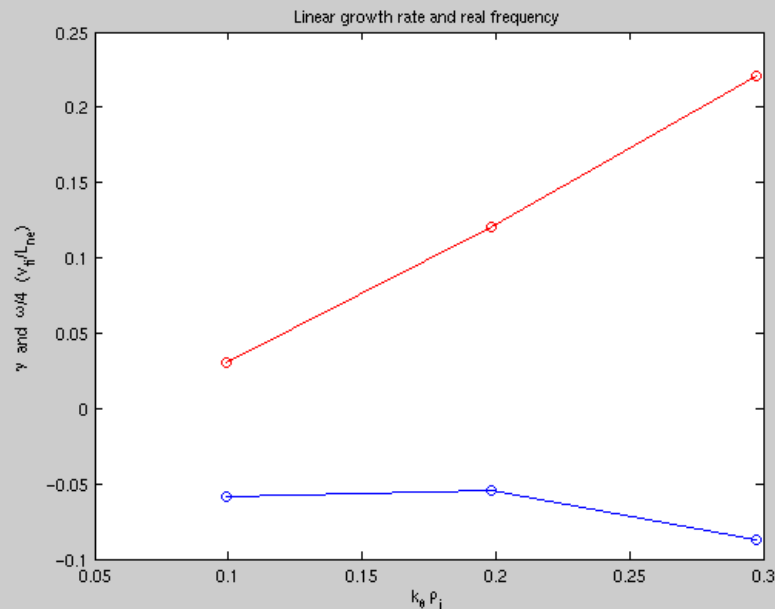
Name

eigensolver_init.m
eigensolver_ITG.m
plot_eigenmode.m
plot_eigenvalue.m
README

Editor - /global/u1/s/sskim/hnut/tools/eigenvalue_solver

1.0

```
1 %%%%%%%%%%
2 % plot linear growth rate and real frequency
3 %%%%%%%%%%
4 clear kth
5 qval=1.4;
6 rval=0.5;
7 for n1=nm
8     kthet
9 end
10 R=1.3; a=
11 gamfactor
12 omgfactor
13
14 figure(2)
15 plot(kthe
16 xlabel('k
17 ylabel('\
18 title('Li
19 hold on
20 plot(kthe
21
```



eigensolver_init.m x eigensolver_ITG.m x plot_eigenmode.m x plot_eigenvalue.m x

Command Window

New to MATLAB? Watch this [Video](#), see [Demos](#), or read [Getting Started](#).

Setting up matrix for eigenvalue equation

```
status =
    Calculating eigenvalues and eigenvectors

status =
    ***** Eigenvalue solver finished *****
```

```
>> plot_eigenvalue
```

```
>>
```

Workspace

Name

Fphi	<
R	1.3
R_Lne	2.0
R_Lte	6.0
R_Lti	6.0
Te_T0	1.0
a	0.0
akpar	<
akpar_enhat	<
akpar_tau	<
akpar_that	<
alphaj	2.0
alphamp	<
amw	<
amw0	0.0
anorm	0.0

Command History

```
clear rssl qss
for k=1:kmax
    if(nfb(k)==n1)
        rssl(k1)=rssl(k);
        qss(k1)=qss(k);
        k1=k1+1;
    end
end
plot(rssl,qss)
% for
% i
% e
% end
hold off
end
-- 09/03/2013
eigensolver_i
eigensolver_I
plot_eigenval
```

Run plot_eigenmode.m:

The screenshot shows the MATLAB IDE interface. The **Editor** window is open to the file `plot_eigenmode.m` located at `/global/u1/s/sskim/bout/tools/eigenvalue_solver/plot_eigenmode.m`. The script contains the following code:

```
1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2 % Contour plot of eigenmode structure for potential
3 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
4 theta=-pi:0.01:pi;
5 mft=mf';
6 matexp=exp(i*mft(1:kmax)*theta);
7 wavemap=wave(:,1:kmax);
8 rcore=r';
9 x=rcore*cos(theta);
10 y=rcore*sin(theta);
11
12 %for n1=nmin:nde1:nmax
13
14 n1=10; % choose toroidal mode number
15
16 nn=n1/nde1;
17 figure(2+nn)
18 ph=0*ones(kmax,1);
19 l=0;
20 for k=1:kmax
21     if(nfb(k)==n1)
22         l=l+1;
23         ph(k)=Fphi(nn,l);
24     end
```

A red box highlights the line `n1=10; % choose toroidal mode number`, with an arrow pointing to it and the text: "You can choose here toroidal mode number."

The **Command Window** shows the following output:

```
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Setting up matrix for eigenvalue equation

status =

    Calculating eigenvalues and eigenvectors

status =

***** Eigenvalue solver finished *****

>> plot_eigenvalue
>>
```

The **Workspace** window on the right lists variables: `Fphi`, `R`, `R_Lne`, `R_Lte`, `R_Lti`, `Te_T0`, `a`, `akpar`, `akpar_enhat`, `akpar_tau`, `akpar_that`, `alphaj`, `alphamp`, `amw`, `amw0`, and `anorm`.

The **Command History** window shows the following commands:

```
clear rssl q
for k=1:kmax
    if(nfb(k)==n1)
        rssl(k1)=rssl
        qssl(k1)=qssl
        k1=k1+1;
    end
end
plot(rssl,qssl)
hold off
end
```

File Edit Text Go Cell Tools Debug Parallel Desktop Window Help

Current Folder: /global/u1/s/sskim/hout/tools/eigenvalue_solver

File Edit View Insert Tools Desktop Window Help

Shortcuts How to Add What's New

Current Folder

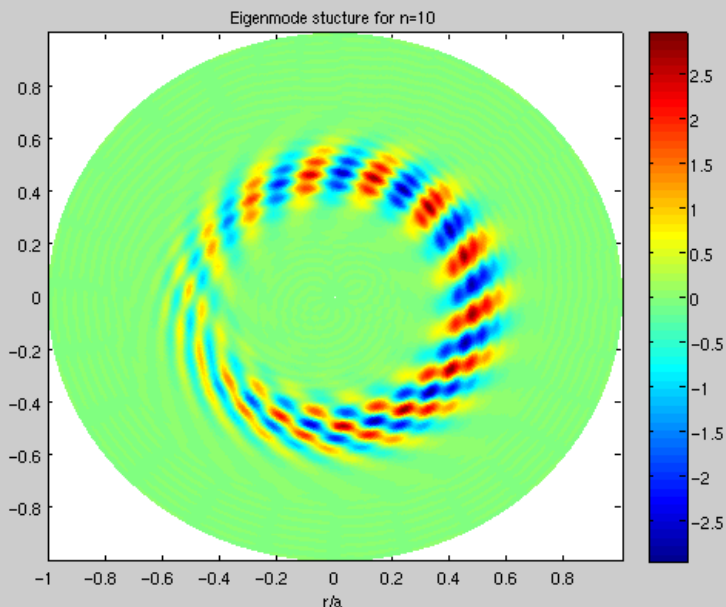
<< tools >> eigenvalue_solver

Name

eigensolver_init.m
eigensolver_ITG.m
plot_eigenmode.m
plot_eigenvalue.m
README

Editor - /global/u1/s/sskim/hout/tools/eigenvalue_solver

```
1 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
2 % Contour
3 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
4 theta=-pi;
5 mft=m'f';
6 matexp=ex;
7 wavemap=w;
8 rcore=r';
9 x=rcore*c;
10 y=rcore*s;
11
12 %for n1=n
13
14 n1=10; %
15
16 nn=n1/n;
17 figure(
18 ph=0*on
19 l=0;
20 for k=1
21 if(nf
22 l=
23 ph
24 end
```



eigensolver_init.m x eigensolver_ITG.m x plot_eigenmode.m x plot_eigenvalue.m x

Command Window

New to MATLAB? Watch this [Video](#), see [Demos](#), or read [Getting Started](#).

```
status =  
  
Calculating eigenvalues and eigenvectors  
  
status =  
  
***** Eigenvalue solver finished *****  
  
>> plot_eigenvalue  
>> plot_eigenmode  
Warning: single buffer visual not available  
fx >>
```

Workspace

Name

Fphi	<
R	1
R_Lne	2
R_Lte	6
R_Lti	6
Te_T0	1
a	0
akpar	<
akpar_enhat	<
akpar_tau	<
akpar_that	<
alphaj	2
alphamp	<
amw	<
amw0	0
anorm	0

Command History

```
for k=1:kmax  
if(nfb(k)==n1  
rss1(k1)=rss(  
qss1(k1)=qss(  
k1=k1+1;  
end  
end  
plot(rss1,qss  
% for  
% i  
%  
% e  
% end  
hold off  
end  
%- 09/03/2013  
eigensolver_i  
eigensolver_I  
plot_eigenval  
plot_eigenmoc
```

Start

Exit matlab:

The screenshot displays the MATLAB environment. The **File** menu is open, with **Exit MATLAB** highlighted at the bottom. The main window shows a plot titled "Eigenmode structure for n=10", which is a circular heatmap with a color bar ranging from -2.5 to 2.5. The plot shows a complex, swirling pattern of colors. The Command History window on the right shows the following code:

```
for k=1:kmax
    if(nfb(k)==n1)
        rss1(k1)=rss(k);
        qss1(k1)=qss(k);
        k1=k1+1;
    end
end
plot(rss1,qss1)
hold off
end
```

The Command Window at the bottom shows the following output:

```
Calculating eigenvalues and eigenvectors

status =

***** Eigenvalue solver finished *****

>> plot_eigenvalue
>> plot_eigenmode
Warning: single buffer visual not available
fx >>
```