ジャイロ運動論的シミュレーションコードGKVを用いた微視的不安定性・乱流輸送解析

前山伸也

名大理

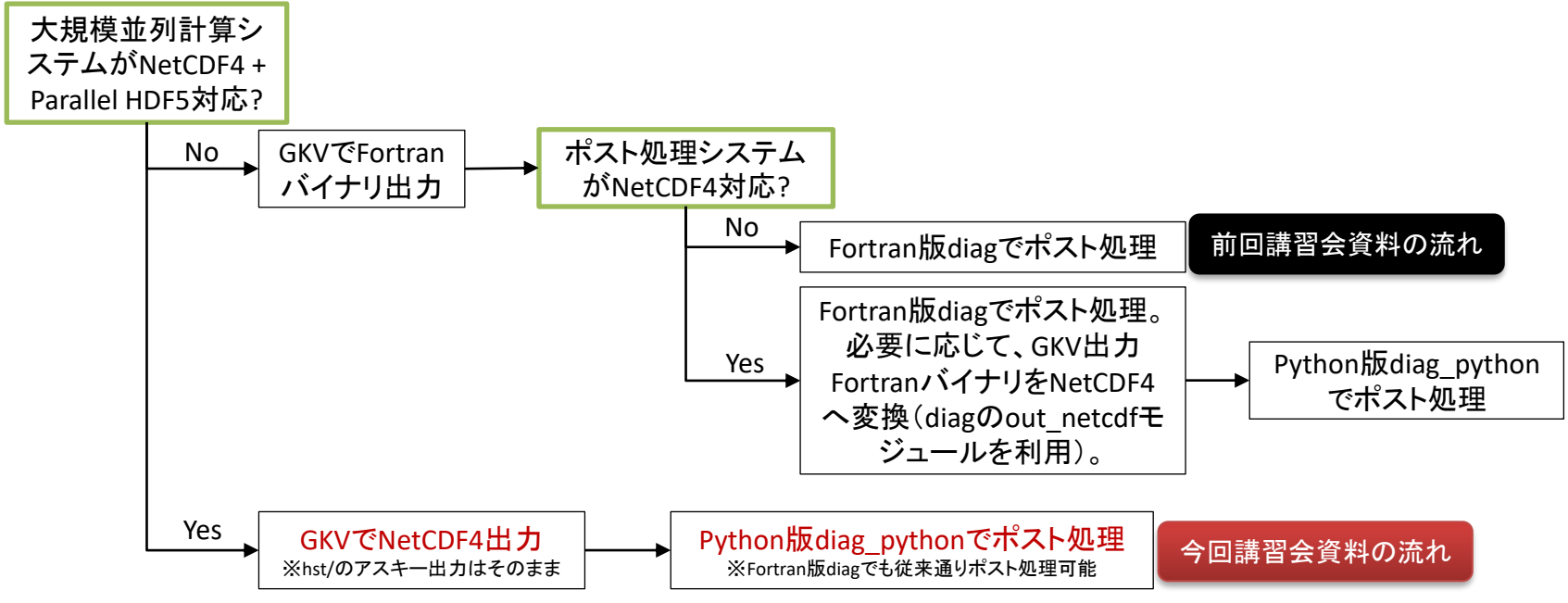
第6回GKV講習会
2022年3月31日

GKVホームページ - https://www.p.phys.nagoya-u.ac.jp/gkv/
GitHub organization - https://github.com/GKV-developers/

今回講習会の目標

GKVの実行から解析まで→基本的には、前回（第5回）講習会資料と同様。

今回は、さらに最近実装されたGKVのNetCDF4出力とPython版ポスト処理プログラムdiag_pythonを使ってみます。



Contents

- GKVのコード構造
- 数値パラメータ、物理パラメータ、計算機環境の設定
- コンパイルおよび実行
- 出力データ構造
- ポスト処理
- まとめ

ターミナルコマンド表記例（薄青色で囲って示します）

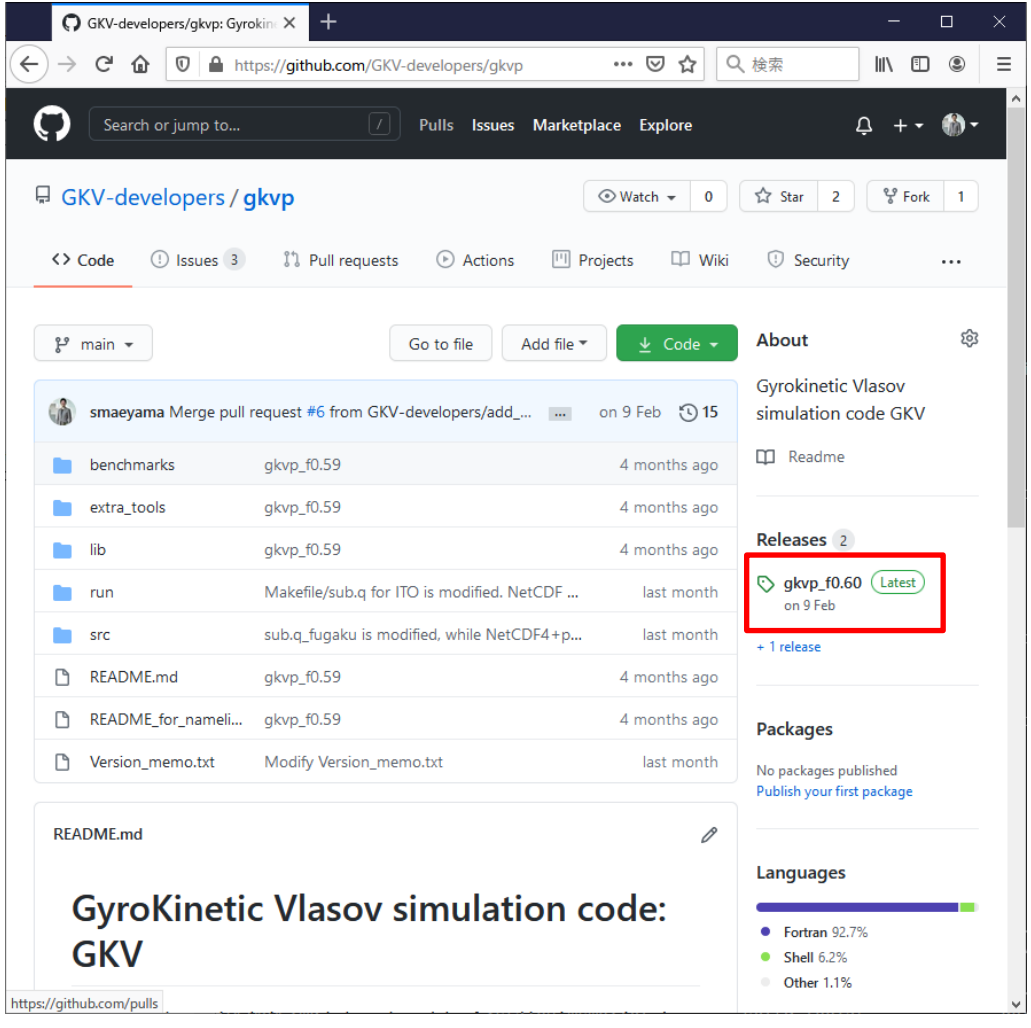
```
$ pstime      # プラズマシミュレータ計算資源量確認
$ lsquota     # プラズマシミュレータディスク利用確認
$ rscinfo     # プラズマシミュレータリソース情報確認
```

ソースコード表記例（黒色で囲って示します）

src/gkvp_main.f90

```
PROGRAM GKV_main
!-----------------------------------------
!
!    GKV+: nonlinear gyrokinetic Vlasov code
```

始めに：GKVコードのダウンロード



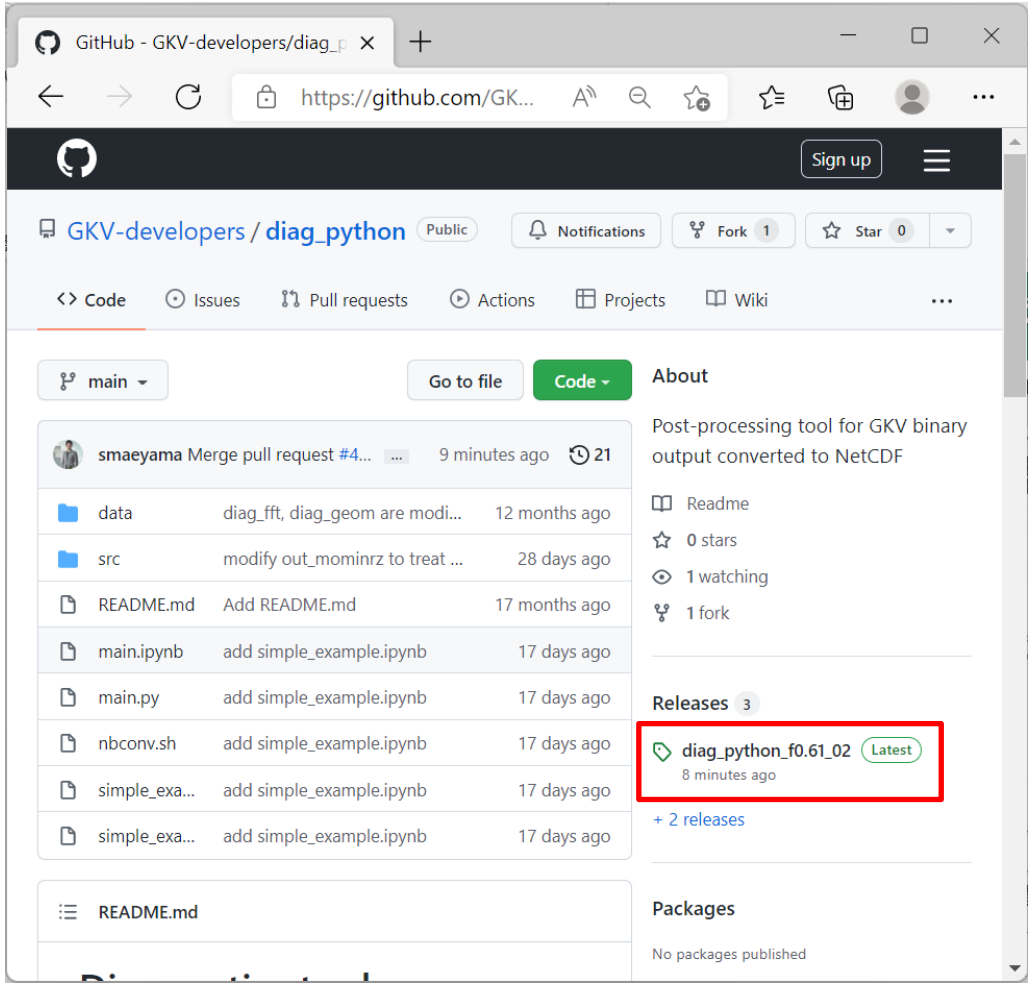
【方法1】Webブラウザから https://github.com/GKV-developers/gkvp/ にアクセスし、Releasesボタンからgkvp_f0.61を選択し、gkvp-f0.61.tar.gzをダウンロード。

【方法2】wget コマンド
プラズマシミュレータログインメニューから[3] Shell promptにログインして、

```
$ wget https://github.com/GKV-developers/gkvp/archive/f0.61.tar.gz -O gkvp-f0.61.tar.gz
```

【方法3】git コマンド
Gitに慣れている方は、[3] Shell promptにログインして、git clone.

同様に、diag_pythonもダウンロードしてください



【方法1】Webブラウザから https://github.com/GKV-developers/diag_python/ にアクセスし、Releasesボタンから、diag_python-f0.61_02.tar.gzをダウンロード。

【方法2】wget コマンド
プラズマシミュレータログインメニューから[3] Shell promptにログインして、

```
$ wget https://github.com/GKV-developers/diag_python/archive/f0.61_02.tar.gz -O diag_python-f0.61_02.tar.gz
```

【方法3】git コマンド
Gitに慣れている方は、[3] Shell promptにログインして、git clone.

実習のための下準備

プラズマシミュレータログインメニューから、[1] Front System にログイン。

ダウンロードしてきたソースコード（例えばgkvp-f0.61.tar.gz）を展開。

```
$ tar xzvf gkvp-f0.61.tar.gz
```

作成された gkvp-f0.61/run/ ディレクトリ内に移動し、プラズマシミュレータ向けMakefile, sub.q, shoot スクリプトをbackup/ディレクトリからコピー・上書き。

```
$ cd gkvp-f0.61/run/
$ cp backup/Makefile_ps_sx ./Makefile    # プラズマシミュレータ向けテンプレート Makefile_ps_sx
$ cp backup/sub.q_ps_sx ./sub.q          # プラズマシミュレータ向けテンプレート sub.q_ps_sx
$ cp backup/shoot_ps_sx ./shoot          # プラズマシミュレータ向けテンプレート shoot_ps_sx
```

※現状、スーパーコンピュータ Fugaku (RIKEN R-CCS), Plasma Simulator (NIFS), JFRS-1 (IFERC-CSC), Flow (Nagoya Univ.), ITO (Kyushu Univ.)向けのテンプレートが用意してあります。また、小規模PCクラスタ（名大P研）や個人ラップトップ(Windows-WSL(Ubuntu),Mac)での利用事例もあります。

NetCDF4+Parallel HDF5利用の設定

gkvp_f0.60以降、バイナリ出力形式として、既存のFortranバイナリ分割出力の他に、NetCDF並列出力が選択可能。
システムにNetCDF4+Parallel HDF5がインストールされていることが要件。

プラズマシミュレータの例:
1. SX用クロスコンパイル済みNetCDF4+Parallel HDF5のモジュールをロードする。

```
$ module load netcdf-parallelIO-fortran-sx
```

2. Makefile内30-31行の、FILEIO=gkvp_fileio_fortranをコメントアウトし、FILEIO=gkvp_fileio_netcdfを有効にする。
これにより、GKVバイナリファイル入出力モジュール GKV_fileio をNetCDF用に切り替える。

run/Makefile (PS SX-Aurora用)

```
#FILEIO=gkvp_fileio_fortran
FILEIO=gkvp_fileio_netcdf
### Usage of NetCDF (module load netcdf-parallelIO-fortran-sx)
ifeq ($(FILEIO),gkvp_fileio_netcdf)
  FC = mpinfort
  LIB += -lnetcdff -lnetcdf -lhdf5_hl -lhdf5
endif
```

3. バッチジョブスクリプト sub.q 内でもモジュールをロードしておく。

run/sub.q

```
module load netcdf-parallelIO-fortran-sx
```

GKVのコード構造(バージョンgkvp_f0.61)

gkvp-f0.61/
- README_for_namelist.txt　簡単な説明書き
- Version_memo.txt　最近の更新履歴
- src/　ソースファイル群
  - gkvp_header.f90　解像度、MPIの設定モジュール
  - gkvp_out.f90　標準データ出力モジュール
- lib/　乱数・ベッセル関数ライブラリ呼び出しモジュール
- extra_tools/　ポスト処理ツール等
  - fig_stdout_20201012.tar.gz　アスキーデータのPDF化
- run/　コンパイルおよび計算実行
  - gkvp_namelist　物理パラメータの設定
  - sub.q　バッチジョブ用スクリプト(計算機依存)
  - shoot　ジョブ投入スクリプト(計算機依存)
  - Makefile　コンパイル情報(計算機依存)
  - backup/　各計算機向けsub.q,shoot,Makefileのバックアップ

GKVで扱える問題

ある平衡の下で、微視的不安定性や乱流揺動、粒子・熱輸送の局所解析。

1. 線形解析

- 線形モードの成長率、実周波数、揺動間のクロスフェーズなどを調べる。
- 線形モードの独立性から、特定の波数のみ解析するため計算は早い。
- 非線形飽和機構が入っていないので、振幅の絶対値は求まらない。

2. 非線形解析

- 揺動スペクトル、粒子・熱輸送、その他諸々の乱流揺動解析を行う。
- 乱流混合を扱うために多数のモード間の非線形結合を解く必要があり、計算に時間がかかる。要求解像度も問題に依るので、数値的健全性確保のためにエントロピーバランスやスペクトルの収束性確認が必要。

今日の実習の問題設定

円形トカマクモデル磁場の下で、微視的不安定性の線形解析をする。

1. 物理パラメータを、run/gkvp_namelistに入力する。
2. src/gkvp_header.f90に計算格子数およびMPI並列数を入力する。
3. バッチジョブスクリプトsub.qを設定する。
4. ジョブ投入スクリプトshootにディレクトリの設定をする。
5. コンパイルし、計算を実行。
6. 出力データを解析する。（ポスト処理ツールの利用）

※非線形解析も計算タイプ”nonlinear”とし高解像度化する位で同様の手順。

1. 実験→GKV換算、namelistへの入力

実験計測より、

- 解析する半径位置を決めて、局所パラメータを算出
- MHD平衡を構築

GKVで解析するため、

- 実験→GKVパラメータ換算（規格化）
- 対応する物理パラメータのrun/gkvp_namelistへの入力
- MHD平衡からGKVで必要となるメトリックデータへの加工

今回はMHD平衡磁場配位データは利用せずに、s-alphaモデルと呼ばれる円形トカマク磁場モデルを用いるので、必要なパラメータは逆アスペクト比、安全係数、磁気シア、密度勾配、温度勾配などのみ。

※実験データの換算やMHD平衡磁場配位データの読み込みについては今回は割愛。過去講習会資料（例えば第3,4回はVMEC平衡データを読み込んだハンズオン実習を実施）など参照。
https://www.p.phys.nagoya-u.ac.jp/gkv/_src/1163/gkv_setting_191213.pdf

1. 実験→GKV換算、namelistへの入力

run/gkvp_namelist

```
&cmemo memo="GKV-plus f0.61 developed for pre-exa-scale computing", &end
&calct calc_type="lin_freq",                  # 計算タイプ lin_freq / nonlinear
       z_bound="outflow",
       z_filt="off",
       z_calc="cf4",
       art_diff=0.1d0,
       init_random=.false.,
       num_triad_diag=0, &end
&triad mxt = 0, myt = 0/
&equib equib_type = "analytic", &end          # 平衡磁場モデル analytic / s-alpha /
…                                                s-alpha-shift / vmec / eqdsk / slab
…
&runlm e_limit = 60.d0, &end                  # 計算実行の実時間[秒]
&times tend = 200.d0,                         # シミュレーション上の上限時間[R_ref/v_ref]
       dtout_fxv = 10.d0,                     # データ出力の時間間隔1
       dtout_ptn = 0.1d0,                     # データ出力の時間間隔2
       dtout_eng = 0.1d0,                     # データ出力の時間間隔3
       dtout_dtc = 0.1d0, &end                # 自動時間刻み幅の調整間隔
…
```

run/gkvp_namelist内の計算実行に関わる部分を適宜編集する。
（例：非線形計算を行う calc_type="nonlinear", 計算実行時間を延ばす e_limit=3600.d0）

1. 実験→GKV換算、namelistへの入力

run/gkvp_namelist

```
&physp R0_Ln = 2.22d0,                  # 規格化密度勾配 R_0/L_{n_s}
       R0_Lt = 6.92d0,                  # 規格化温度勾配 R_0/L_{T_s}
       nu = 1.d0,
       Anum = 1.d0,                     # 質量数 m_s/m_ref
       Znum = 1.d0,                     # 価数の絶対値 |e_s|/e
       fcs = 1.d0,                      # 電荷密度 e_s n_s/(e n_ref)
       sgn = 1.d0,                      # 電荷の符号 e_s/|e_s|
       tau = 1.d0,                      # 温度 T_s/T_ref
       dns1 = 1.d-2,
       tau_ad = 1.d0,
       lambda_i = 0.d0,
       beta = 0.d0,                     # プラズマベータ値 μ_0 n_ref T_ref/B_ref^2
…
&nperi n_tht = 3,                       # 磁力線方向ボックスサイズ(ポロイダル角で±n_tht*pi)
       kymin = 0.05d0,                  # 磁力線ラベル方向ボックスサイズ ly = pi/kymin
       m_j   = 1,                       # 半径径方向ボックスサイズ lx = pi/kxmin
       del_c = 0.d0, &end                                kxmin = |2*pi*s_hat*kymin/m_j|
&confp eps_r    = 0.18d0,               # 逆アスペクト比 r_{at-fluxtube-center}/L_ref
       eps_rnew = 1.d0,
       q_0      = 1.4d0,               # 安全係数 q
       s_hat    = 0.8d0,               # 磁気シア ŝ
…
```

2. src/gkvp_header.f90に計算格子数とMPI並列数を入力する。

src/gkvp_header.f90

```
!-------------------------------------
!  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 = 2, nyw = 20
  integer, parameter :: nx = 0, global_ny = 12 ! 2/3 de-aliasing rule
  integer, parameter :: global_nz = 48, global_nv = 24, global_nm = 15

  integer, parameter :: nzb = 2, &  ! the number of ghost grids in z
                        nvb = 2     ! the number of ghost grids in v and m

!-------------------------------------
!  Data distribution for MPI
!-------------------------------------

  integer, parameter :: nprocw = 2, nprocz = 4, nprocv = 2, nprocm = 2, nprocs = 1
```

2. src/gkvp_header.f90に計算格子数とMPI並列数を入力する。

ここで、

nx | kxモード数 –nx:nx
global_ny | kyモード数 0:global_ny
（さらにnxw>nx*3/2, nyw>global_ny*3/2となるように設定。）
global_nz | 磁力線方向座標 –n_tht*pi<zz<n_tht*pi を –global_nz:global_nz-1で離散化
global_nv | 磁力線方向速度 -vmax<vl<vmaxを1:2*global_nvで離散化
global_nm | 磁気モーメント 0<mu<vmax^2/2を0:global_nmで離散化

nprocw, nprocz, nprocv, nprocm, nprocs はky,zz,vl,mu方向と粒子種sのMPI領域分割数。

ただし、
✓ (global_ny+1)/nprocw, global_nz/nprocz,
global_nv/nprocv, (global_nm+1)/nprocmは整数。
✓ nprocsは扱う粒子種数と一致。

3.バッチジョブスクリプトsub.qを設定する。

run/sub.q

```
…
#PBS -q small                    # queue name
#PBS --group=20234               # resource group
#PBS -T necmpi                   # necessary for MPI job
#PBS -l elapstim_req=00:15:00    # elapsed time limit

#PBS --venode=4                  # total number of VE
#PBS --venum-lhost=8             # number of VE per a logical node
#PBS -v OMP_NUM_THREADS=1        # number of threads per MPI process

MPI_procs=32                     # number of MPI processes (= venode*8 for flat MPI)
…
```

GKV講習会用グループID **21224** に変更してください。

run/sub.qでMPI/OpenMP並列数を指定する。

- ✓ MPIプロセス数はsrc/gkvp_header.f90と整合するように
  MPI_procs = nprocw*nprocz*nprocv*nprocm*nprocs と設定。
- ✓ SX-Aurora TSUBASAでは、2021年3月現在、フラットMPIの方が性能が高いので、
  venode = MPI_procs / 8（整数）と設定。

4.ジョブ投入スクリプトshootにディレクトリの設定をする。

run/shoot

```
…
if [ $# -lt 2 ]; then
  echo "HOW TO USE: ./shoot [START_NUMBER] [END_NUMBER] ([JOB-ID])"
  exit
fi

#### Environment setting
DIR=/data/lng/maeyama/gkvp/f0.61/ITGae-lin      # 実行後のデータ出力ディレクトリ
LDM=gkvp.exe
NL=gkvp_namelist
SC=qsub
JS=sub.q
### For VMEC, set VMCDIR including metric_boozer.bin.dat
#VMCDIR=./input_vmec/vmec_sample_nss501ntheta1024nzeta0
### For IGS, set IGSDIR including METRIC_{axi,boz,ham}.OUT
#IGSDIR=../../input_eqdsk_for_eqdskbench/
…
```

各自のユーザーディレクトリ以下の好きなパスに書き換えてください。
例： **DIR=/data/lng/(ユーザーID)/gkv_training/linear_test/**

5. コンパイルし、計算を実行

コンパイルする。

```
$ cd gkvp-f0.61/run/
$ make clean              # 省略可能
$ make
```

計算を実行する。以下の形式でshootスクリプトを利用することでステップジョブ実行される。

./shoot　START_NUM　END_NUM　(JOB_ID)

例）シングルジョブ投入（*.001）　　./shoot 1 1
シングルジョブ投入（*.002）　　./shoot 2 2
ステップジョブ投入（*.003 - *.005)　　./shoot 3 5
継続ステップジョブ投入　　./shoot 6 7 11223
（*.005まで計算するジョブJOB_ID=11223がキュー中にあり、
それに続けて*.006 - *.007のジョブを実行させようとした。）

今回はひとまず、1回分のシングルジョブを投入してみましょう。

```
$ ./shoot 1 1
```

※投入したジョブの処理状況の確認 `$ qstat`

5. コンパイルし、計算を実行

正常に計算が実行されれば、run/shootで設定した出力ディレクトリ
（例: DIR=/data/lng/(ユーザーID)/gkv_training/linear_test/）
に以下のデータが書き出される。

log/ 計算ログ
cnt/ 継続計算用バイナリデータ
fxv/ 分布関数バイナリデータ(いくつかの磁力線方向座標位置で)
phi/ ポテンシャル、流体モーメント、エントロピーバランスに関するバイナリデータ
hst/ アスキー形式の標準出力
その他:実行環境バックアップのためのコピー

Appendix A. GKVの出力データー覧 にまとめた。GKV Manualにも同様の記載あり。
さらに詳細は、ソースコード src/gkvp_out.f90 を参照。

6. 出力データを解析する。

出力データを解析するには、

6-a) 自力で何とかする。
- GKVの出力データは一覧にまとめてあるので、後は適当にポスト処理する。
- アスキー形式の標準出力くらいなら簡単。
- MPI領域分割されたバイナリデータを読み込むのは結構手間。

※アスキー形式の標準出力についていろいろとプロットしてみましょう。

コードのオープン化にあたり、ポスト処理ツールとして以下の3つを提供。

6-b) hst/のアスキー標準出力を一括でPDF化するためのスクリプト fig_stdout
6-c) phi/などのバイナリデータ解析のためのFortran版ポスト処理プログラム diag（割愛）
6-d) バイナリデータ解析のためのPython版ポスト処理プログラム diag_python

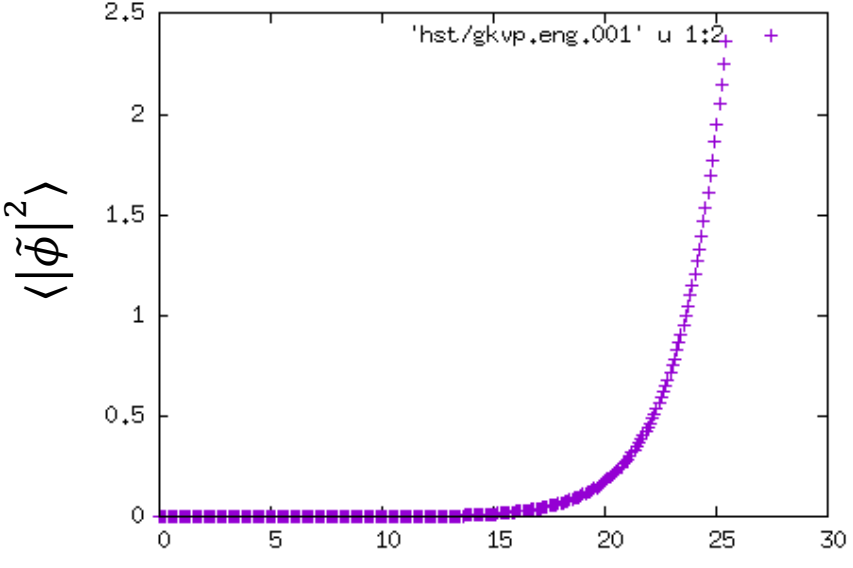
※ fig_stdout を実行してみましょう。
※ diag_pythonを実行してみましょう。

6-a). hst/のアスキー標準出力をプロットする。

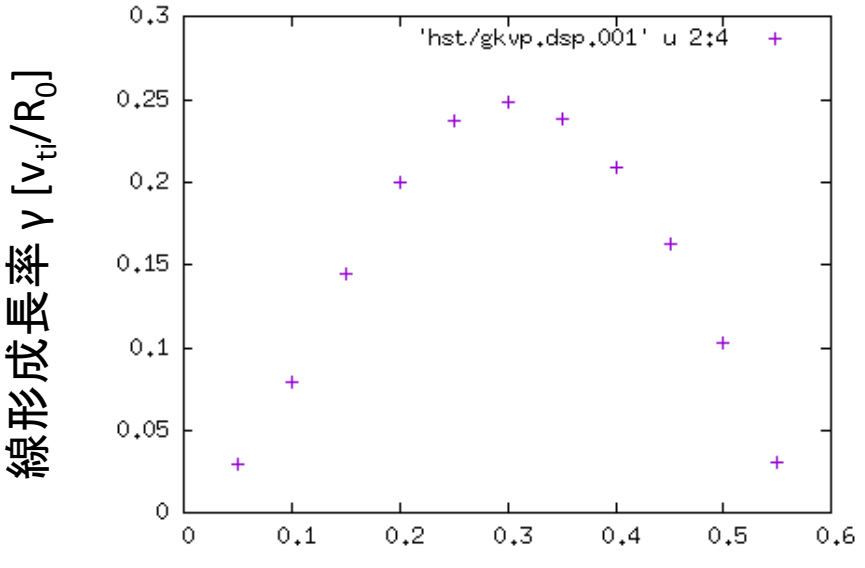
例：

```
$ cd /data/lng/(ユーザーID)/gkv_training/linear_test/
$ gnuplot
gnuplot> plot ‘hst/gkvp.eng.001’ u 1:2
gnuplot> plot ‘hst/gkvp.dsp.001’ u 2:4
```

揺動振幅$\langle|\tilde{\phi}|^2\rangle$の線形成長



線形分散関係



6-b) hst/のアスキー標準出力を一括でPDF化するためのスクリプト fig_stdout

LaTeXを利用するために、[2] Visualization Processing Server にログイン。

```
$ exit
Connection to fes1.ps.nifs.ac.jp closed.
 +-----------------------------------------------+
 |  Server Login Menu                            |
 +-----------------------------------------------+
 | [1] Front System (fes1-4)                     |
 | [2] Visualization Processing Server (vis1-4)  |
 | [3] Shell Prompt                              |
 | [4] I want to specify the node to access      |
 | [q] Logout                                    |
 +-----------------------------------------------+
Select No : 2
```

gkvp-f0.61/extra_tools/fig_stdout_20210316.tar.gz をGKV出力データのあるディレクトリ（例：DIR=/data/lng/(ユーザーID)/gkv_training/linear_test/）に展開する。

```
$ cd gkvp-f0.61/extra_tools/
$ tar xzvf fig_stdout_20210316.tar.gz
$ mv fig_stdout_20210316/  /data/lng/(ユーザーID)/gkv_training/linear_test/
```

6-b) hst/のアスキー標準出力を一括でPDF化するためのスクリプト fig_stdout

展開したディレクトリ内は以下の構成：

```
fig_stdout_20210316/
    make_pdf.sh    PDF作成シェルスクリプト
    pdf/           PDFが格納されるディレクトリ
    eps/           PDF作成に用いられたepsが格納されるディレクトリ
    data/          eps作成に用いられた元データが格納されるディレクトリ
    src/           gnuplot用スクリプト等
```

となっており、make_pdf.shスクリプトを実行

```
$ cd /data/lng/(ユーザーID)/gkv_training/linear_test/fig_stdout_20210316/
$ ./make_pdf.sh clean              # 省略可能
$ ./make_pdf.sh
```

すると、一覧のPDF（fig_stdout_20210316/pdf/fig.pdf）やeps、元データが格納される。

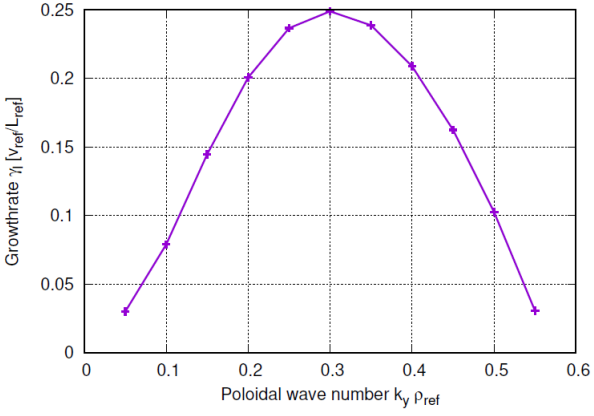
※必ずしも図のスケール等が見やすいとは限らない。
※プラズマシミュレータのgnuplotは通常バージョン4.6.2ですが、下記パスにあるバージョン5.2.7も利用できます。
/system/apps/rhel7/lx/gnuplot/5.2.7/bin/gnuplot

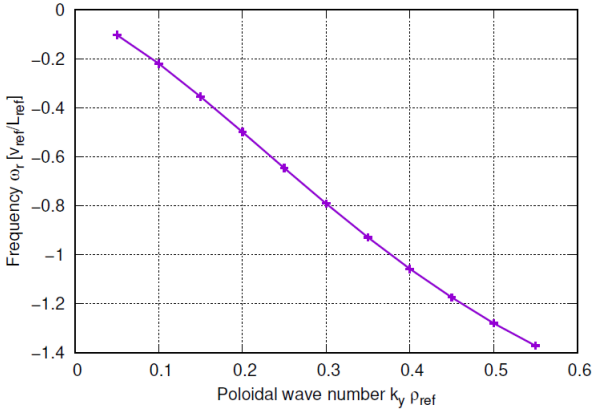
線形計算出力例：fig_stdout/pdf/fig.pdfより抜粋

PDFをevinceで開いてみる。

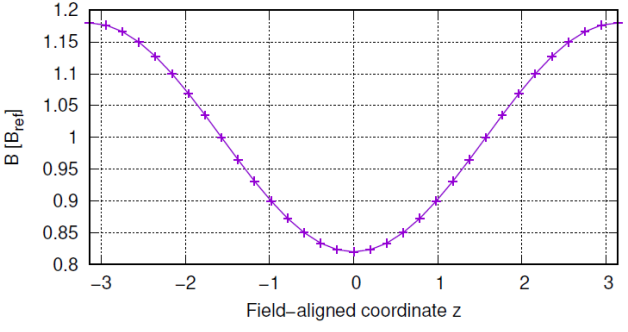
```
$ cd fig_stdout_20210316/
$ evince pdf/fig.pdf
```

線形成長率・実周波数のポロイダル方向波数依存性

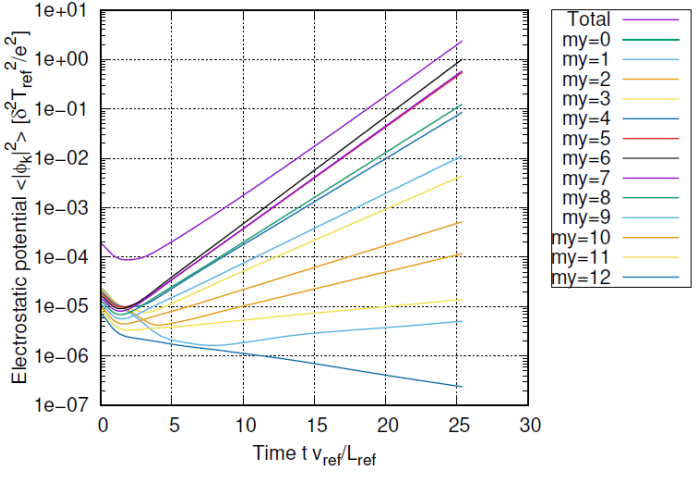




磁場強度の磁力線方向z分布



z=0がトーラス外側（悪い曲率領域）、
z=πがトーラス内側（良い曲率領域）
に相当。



ポロイダルモード数毎の揺動振幅 $\langle|\tilde{\phi}|^2\rangle$の時間発展

6-d) バイナリデータ解析のためのPython版ポスト処理プログラム diag_python

6ページの手順でダウンロードしてきた diag_python-f0.61_02.tar.gz をGKV出力ディレクトリ（例： DIR=/data/lng/(ユーザーID)/gkv_training/linear_test/）に展開する。

```
$ tar xzvf diag_python-f0.61_02.tar.gz
$ mv diag_python-f0.61_02/  /data/lng/(ユーザーID)/gkv_training/linear_test/
```

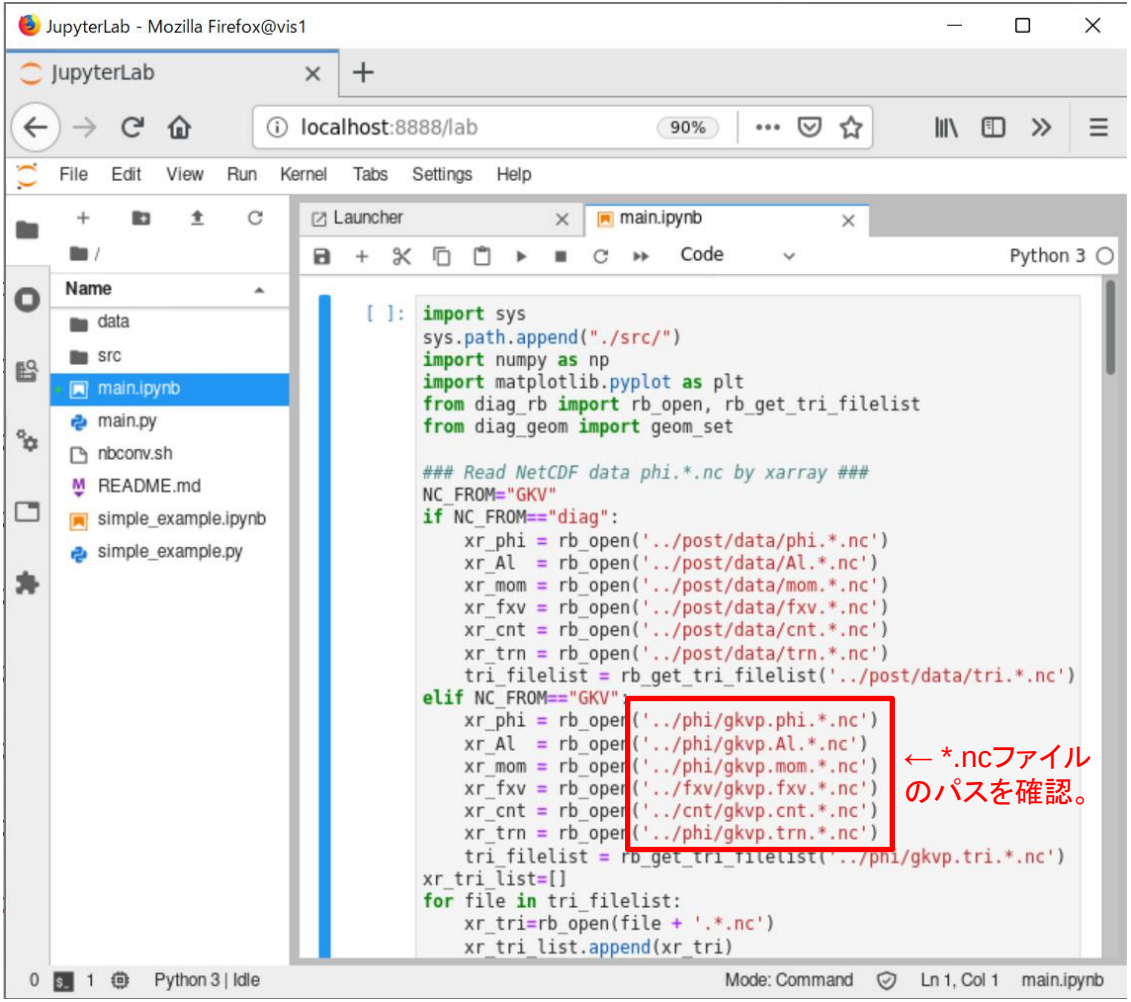
diag_python は、GKV出力NetCDFデータを読み込みポスト処理を行うPythonスクリプト群です。ユーザーがなじみのPython環境から呼び出して利用してもらえばよいですが、ここでは可視化処理サーバにインストールされている Jupyter lab を利用してみましょう。

```
$ cd /data/lng/(ユーザーID)/gkv_training/linear_test/diag_python-f0.61_02/
$ jupyter lab
```

※ 上記コマンドで、可視化処理サーバ上でJupyter labサーバとブラウザ(Firefox)が立ち上がり、X Window経由で各自のPC上に表示されます。X経由に由来する処理遅延が気になる場合、Plasma SimulatorではNoMachineクライアントによるリモートデスクトップ接続が推奨されています。

- インストール方法 [PSホームページ - NoMachineクライアント] https://www.ps.nifs.ac.jp/wordpress/?page_id=72
- 操作方法 [PSホームページ – 利用の手引き – 13.1節] https://www.ps.nifs.ac.jp/documents/2/ps-usersguide-v8.pdf

6-d) バイナリデータ解析のためのPython版ポスト処理プログラム diag_python



基本操作

- main.ipynbを開く。
- セルを選択し、[Shift] + [Enter]でセルの内容を実行。
- 実行途中の変数など、すべて記憶されているので、もう一度初めからやり直したい場合は、上部タブから [Kernel] – [Restart Kernel and Clear All Outputs]を選択。

※一つ目のセルは、NetCDFファイルのパスやheader, namelist, メトリックデータのパスを読み込み、各種形状データの初期設定を行う部分で、実行必須です。
※2つ目以降のセルは行いたい解析に合わせて実行。

6-d) バイナリデータ解析のためのPython版ポスト処理プログラム diag_python

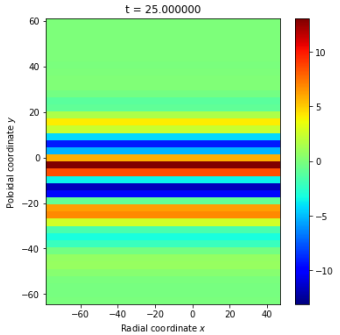
diag_pythonの構成について(simple_example.ipynbを例に)

```
import sys
sys.path.append("./src/")

### Read NetCDF data phi.*.nc by xarray ###
from diag_rb import rb_open, rb_get_tri_filelist
xr_phi = rb_open('../phi/gkvp.phi.*.nc')

### Set geometric constants ###
from diag_geom import geom_set
geom_set(headpath='../src/gkvp_header.f90',
         nmlpath="../gkvp_namelist.001",
         mtrpath='../hst/gkvp.mtr.001')

# Plot phi[y,x] at t[it], zz[iz]
from out_mominxy import phiinxy
it = 250
iz = 48
phiinxy(it, iz, xr_phi, flag="display")
```



← diag_python/src/内のPythonスクリプト群をimportできるようにパスを追加。

← NetCDFデータ gkvp.phi.*.nc を読み込み(xarray)。

← namelist等の情報を読み込み、配位データを作成。
例えば、磁気面平均などの計算に利用します。

========== ここまで初期設定 ==========

← 行いたい解析に応じたモジュールから関数をimportし、実行。

左の例では、./src/out_mominxy.py 内で定義される phiinxy 関数をimportしています。ある時刻 t[it] におけるある磁力線方向座標位置 zz[iz] での静電ポテンシャルの垂直方向2次元分布 phi(y,x) を計算します。
flag="display"や"savefig"は画像を表示したり、保存したりするオプションです。flag=Noneの場合は、x,y,phi(y,x)のデータをreturnしますので、自分で好きに描画できます。
※ 詳しくはヘルプ help(phiinxy) を参照ください。

応用：非線形乱流シミュレーションの実行とデータ解析

線形計算から、非線形乱流シミュレーション用に物理パラメータを変更する。

run/gkvp_namelist

```
&calct calc_type="nonlinear",                    # 計算タイプを非線形に変更
…
&runlm e_limit = 3600.d0, &end                   # 計算実行の実時間[秒]を延ばす
…
&nperi n_tht = 1,                          # 磁力線方向ボックスサイズをポロイダル角で±piに変更
       kymin = 0.05d0,                     # 磁力線ラベル方向ボックスサイズ ly = pi/kymin
       m_j   = 4,                          # 半径径方向ボックスサイズ lx = pi/kxmin
       del_c = 0.d0, &end                                  kxmin = |2*pi*s_hat*kymin/m_j|
…                                                  lxとlyがある程度近い値になるように変更
```

複数モード扱うように解像度・MPI数も変更。

src/gkvp_header.f90

```
 integer, parameter :: nxw = 84, nyw = 42
 integer, parameter :: nx = 55, global_ny = 27 ! 2/3 de-aliasing rule
 integer, parameter :: global_nz = 24, global_nv = 32, global_nm = 15
…
 integer, parameter :: nprocw = 2, nprocz = 4, nprocv = 4, nprocm = 2, nprocs = 1
```

応用：非線形乱流シミュレーションの実行とデータ解析

計算実行時間・MPI数に合わせて、sub.qも変更

run/sub.q

```
#PBS -q small24VH                  # queue name
#PBS --group=21224                 # resource group
#PBS -T necmpi                     # necessary for MPI job
#PBS -l elapstim_req=01:05:00      # elapsed time limit

#PBS --venode=8                    # total number of VE
#PBS --venum-lhost=8               # number of VE per a logical node
#PBS -v OMP_NUM_THREADS=1          # number of threads per MPI process

MPI_procs=64                       # number of MPI processes (= venode*8 for flat MPI)
```

出力先ディレクトリも変更

run/shoot

```
DIR=/data/lng/maeyama/gkv_training/nonlinear_test/ #実行後のデータ出力ディレクトリ
```

※今回の解像度設定例は、Benchmarks/内のITGae-nlケースに基づいています。今実習時間中に計算が終わらなくても、後日、自分で計算した結果と、Benchmarks/内に入っている参考データが合うか確認してみると良いでしょう。なお、非線形計算では計算誤差の蓄積などで時系列データの逐次的振舞いはベンチマークと一致しない可能性があります。

注記:

※2022年3月現在、解像度を上げた場合のParallel NetCDF4出力が遅く、上記設定では計算が完了しません。GKVの実装上の問題なのか、プラズマシミュレータ上でのParallel NetCDF4の問題なのか、今後調査を進めていきます。

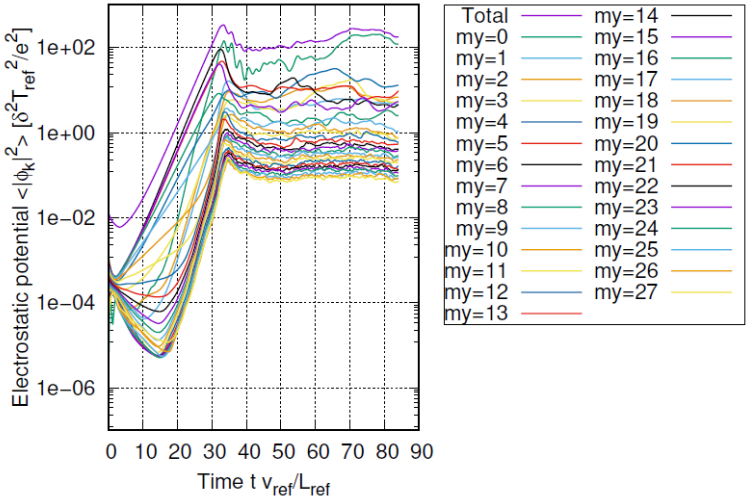
ひとまず、非線形計算を実行するには、7ページのNetCDF4出力の設定を、Fortranバイナリ出力に戻して実行してください。

- この場合でも、hst/内のアスキーファイル出力は不変ですので、本資料24ページまでの手順は変わりません。
- cnt/, fxv/, phi/内の出力がFortranバイナリファイルになります。Fortran版diagを用いて読み込み、解析が可能です。(参照:第2回講習会資料 http://www.p.phys.nagoya-u.ac.jp/gkv/document.html > gkv_training_diag_171215.pdf)
- また、diag内の out_netcdf モジュール内、phiinnetcdf, Alinnetcdf, mominnetcdf, fxvinnetcdf, cntinnetcdf, trninnetcdf, triinnetcdf サブルーチンをcallすることで、Fortranバイナリファイルを NetCDF4ファイルに変換できます。
- 変換したNetCDF4ファイルのパスを指定することで、diag_pythonの利用も可能です。

非線形計算出力例：fig_stdout/pdf/fig.pdfより抜粋

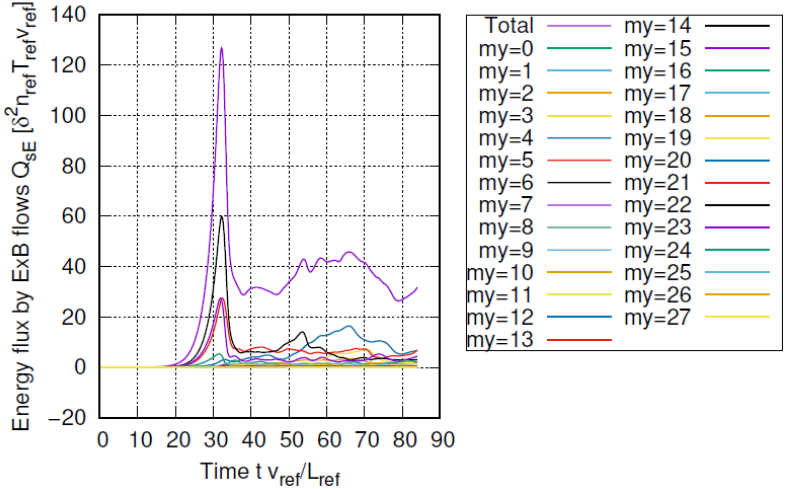
ポロイダルモード数毎の揺動振幅$\langle|\tilde{\phi}|^2\rangle$の時間発展

ky=0(my=0)の帯状流が卓越。

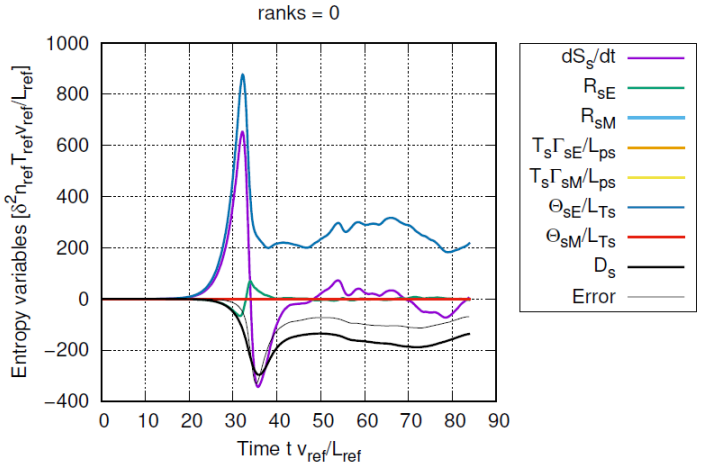


ポロイダルモード数毎のイオン熱輸送フラックスQ_Eの時間発展

ky=0.2(my=4)のモードが主に輸送を作る。



エントロピーバランスの評価



駆動(Θ/L_T)と散逸(D)がつり合い定常状態($dS/dt \sim 0$)に至る。低解像度のため若干の誤差(Error)あり。

まとめ

GKVの利用方法をハンズオン形式で説明した。

要約

GKVの物理モデル・数値モデル
を踏まえた上で、
実験→GKVパラメータ換算
MHD平衡データの加工
の準備をしてから、

src/gkvp_header.f90	解像度、MPIの設定
run/gkvp_namelist	物理・数値パラメータの設定
run/sub.q	MPI・OpenMPの設定
run/shoot	平衡データ・出力ディレクトリの設定

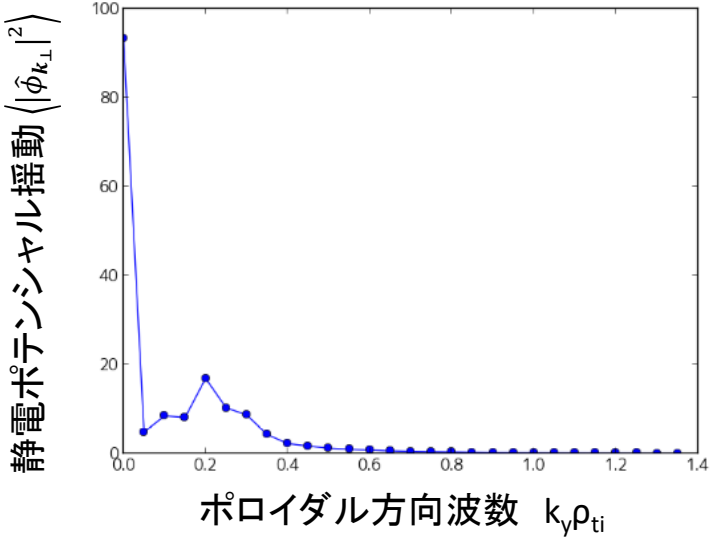
の後にコンパイル(make)、実行(./shoot START_NUM END_NUM)。
hst/のアスキー標準出力を一括でPDF化するためのスクリプト fig_stdout
phi/などのバイナリデータ解析用ポスト処理プログラム
Fortran版 diag または Python版 diag_python
などを利用して、結果を解析する。

発展課題

余力がある人は、

- どんな出力データがあるか、Appendix A を確認しておきましょう。
- イオンの熱輸送フラックスの時間発展もプロットしてみましょう。
- イオン温度勾配を変えて、線形成長率の変化を調べてみましょう。
- 磁力線方向ボックスサイズを変えてみましょう。解像度も変わることに注意。
- MPIやOpenMPの並列数を変えて、処理時間のスケーラビリティを確認してみましょう。
- 非線形シミュレーションにおいて、準定常状態で時間平均した静電ポテンシャル揺動のky波数スペクトルを作ってみましょう。

静電ポテンシャル揺動のポロイダル方向波数スペクトル
(t=40-80で平均)



Pythonスクリプトの例

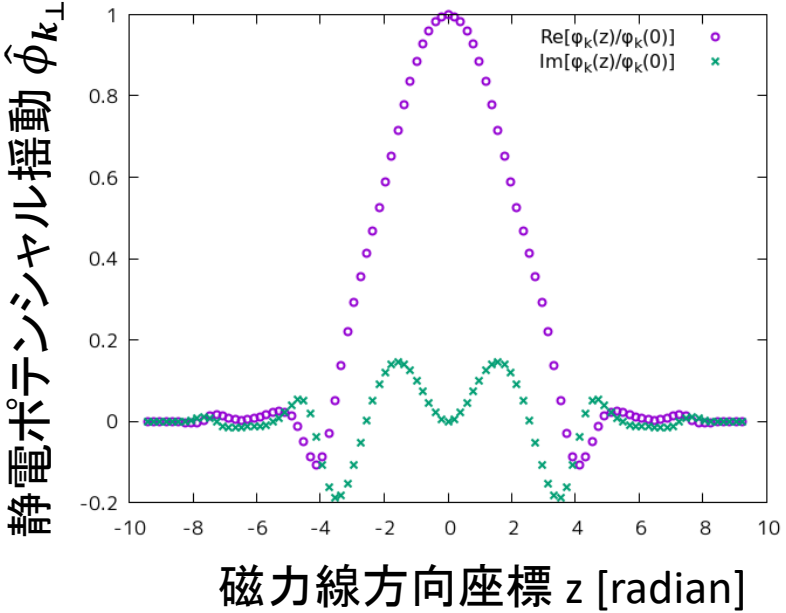
```
#!/usr/bin/env python3
import numpy as np
import matplotlib.pyplot as plt
kymin=0.05 # run/gkvp_namelistで設定したkyの最小値
tsta=40    # ここから、
tend=80    # ここまで時間平均
data = np.loadtxt("gkvp.eng.001")
t=data[:,0]
phi_total=data[:,1]
phi_tky=data[:,2:]
ky=kymin*np.arange(data.shape[1]-2)
ista=np.abs(t[:]-tsta).argmin()
iend=np.abs(t[:]-tend).argmin()
ave_phi = np.average(phi_tky[ista:iend,:],axis=0)
plt.plot(ky,ave_phi,marker="o")
plt.show()
```

発展課題

- もしシステムがNetCDF4対応していない場合は、Fortran版ポスト処理プログラム diag を利用することになります。（参照：第2回講習会資料 http://www.p.phys.nagoya-u.ac.jp/gkv/document.html > gkv_training_diag_171215.pdf）

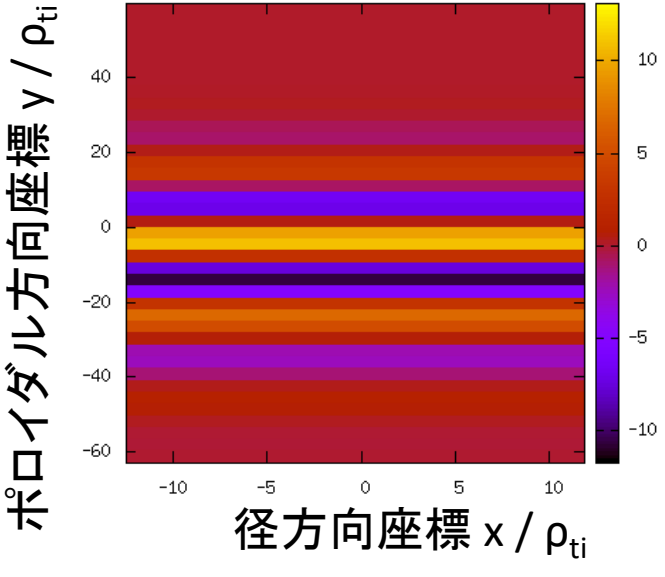
静電ポテンシャル揺動のz方向分布 $\tilde{\phi}_k(z)$

(kx=0,ky=0.3の最大成長モード)



静電ポテンシャル揺動のx,y断面分布 $\tilde{\phi}(x,y)$

(z=0⇔$\theta = 0$の悪い曲率領域での断面)



Appendix A. GKVの出力データー覧

cnt/*cnt*
fxv/*fxv*
phi/*phi*, *Al*, *mom*, *trn*, (非線形の場合のみ *tri*)
hst/*bln*, *ges*, *gem*, *qes*, *qem*, *wes*, *wem*,
eng, *men*, *dtc*, *mtr*, (線形の場合のみ *frq*, *dsp*)
log/*log*

cnt/gkvp.(MPIランク6桁).cnt.(ラン数3桁)

- ファイル形式： バイナリ
- 出力間隔： ランの終了時
- 出力を行うMPIランク： すべて
- 総ファイル数： nprocw*nprocz*nprocv*nprocm*nprocs * (総ラン数)
- GKVコード中の出力ユニット： ocnt
- 格納データ：
  time, ff(-nx:nx,0:ny,-nz:nz-1,1:2*nv,0:nm)

ここで、

time: 時刻（倍精度実数）

ff: 揺動ジャイロ中心分布関数（倍精度複素数）

【説明】

揺動量は、磁気面座標x, 磁力線ラベル座標y, 磁力線方向座標zにおいて、(x,y)方向にフーリエ級数展開され、

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

*.cnt.*には、

$$\tilde{f}_{s\boldsymbol{k}} = \frac{\rho_{ref}}{L_{ref}} \frac{n_s}{v_{ts}^3} \bar{f}_{s\boldsymbol{k}}$$

として規格化された分布関数$\bar{f}_{s\boldsymbol{k}}(z,v_\parallel,\mu)$が格納されている。

fxv/gkvp.(MPIランク6桁).(粒子種1桁).fxv.(ラン数3桁)

- ファイル形式： バイナリ
- 出力間隔： dtout_fxv
- 出力を行うMPIランク： すべて
- 総ファイル数： nprocw*nprocz*nprocv*nprocm*nprocs * (総ラン数)
- GKVコード中の出力ユニット： ofxv
- 格納データ：
  time, ff(-nx:nx,0:ny,1:2*nv,0:nm)

ここで、

time: 時刻（倍精度実数）

ff: 揺動ジャイロ中心分布関数（倍精度複素数）at iz=-nz （z方向MPIランクrankzに依存して、書き出す磁力線方向位置は異なる。）

【説明】

*.cnt.*の項を参照。

phi/gkvp.(MPIランク6桁).0.phi.(ラン数3桁)

- ファイル形式： バイナリ
- 出力間隔： dtout_ptn
- 出力を行うMPIランク： ranks == 0 .and. vel_rank == 0
- 総ファイル数： nprocw*nprocz * (総ラン数)
- GKVコード中の出力ユニット： ophi
- 格納データ：
  time, phi(-nx:nx,0:ny,-nz:nz-1)

ここで、

time: 時刻（倍精度実数）
phi: 揺動静電ポテンシャル（倍精度複素数）

【説明】

揺動量は、磁気面座標x, 磁力線ラベル座標y, 磁力線方向座標zにおいて、(x,y)方向にフーリエ級数展開され、

$$\tilde{\phi}(x,y,z) = \sum_{k_x}\sum_{k_y} \tilde{\phi}_{\boldsymbol{k}}(z) e^{i(k_x x + k_y y)}$$

*.phi.*には、

$$\tilde{\phi}_{\boldsymbol{k}} = \frac{\rho_{ref}}{L_{ref}} \frac{T_{ref}}{e_{ref}} \bar{\phi}_{\boldsymbol{k}}$$

として規格化された静電ポテンシャル$\bar{\phi}_{\boldsymbol{k}}(z)$が格納されている。

phi/gkvp.(MPIランク6桁).0.Al.(ラン数3桁)

- ファイル形式： バイナリ
- 出力間隔： dtout_ptn
- 出力を行うMPIランク： ranks == 0 .and. vel_rank == 0
- 総ファイル数： nprocw*nprocz * (総ラン数)
- GKVコード中の出力ユニット： oAl
- 格納データ：
  time, Al(-nx:nx,0:ny,-nz:nz-1)

ここで、
time: 時刻（倍精度実数）
Al: 揺動ベクトルポテンシャル（倍精度複素数）

【説明】
揺動量は、磁気面座標x, 磁力線ラベル座標y, 磁力線方向座標zにおいて、(x,y)方向にフーリエ級数展開され、

$$\tilde{A}_{\parallel}(x,y,z) = \sum_{k_x}\sum_{k_y} \tilde{A}_{\parallel \boldsymbol{k}}(z) e^{i(k_x x + k_y y)}$$

*.Al.*には、

$$\tilde{A}_{\parallel \boldsymbol{k}} = \frac{\rho_{ref}}{L_{ref}} \rho_{ref} B_{ref} \bar{A}_{\parallel \boldsymbol{k}}$$

として規格化された静電ポテンシャル$\bar{A}_{\parallel \boldsymbol{k}}(z)$が格納されている。

phi/gkvp.(MPIランク6桁).(粒子種1桁).mom.(ラン数3桁)

- ファイル形式： バイナリ
- 出力間隔： dtout_ptn
- 出力を行うMPIランク： vel_rank == 0
- 総ファイル数： nprocw*nprocz*nprocs * (総ラン数)
- GKVコード中の出力ユニット： omom
- 格納データ：
  time, mom(-nx:nx,0:ny,-nz:nz-1,0:nmom-1)

ここで、

time: 時刻（倍精度実数）

mom: 揺動流体モーメント（倍精度複素数）。現状、nmom=6として以下の6つの流体量を順に出力。

$$\tilde{n}_{s\boldsymbol{k}} = \int dv^3\, J_{0s\boldsymbol{k}} \tilde{f}_{s\boldsymbol{k}}, \qquad \tilde{u}_{\parallel s\boldsymbol{k}} = \int dv^3\, v_\parallel J_{0s\boldsymbol{k}} \tilde{f}_{s\boldsymbol{k}}, \qquad \tilde{p}_{\parallel s\boldsymbol{k}} = \int dv^3\, \frac{v_\parallel^2}{2} J_{0s\boldsymbol{k}} \tilde{f}_{s\boldsymbol{k}},$$

$$\tilde{p}_{\perp s\boldsymbol{k}} = \int dv^3\, \mu B J_{0s\boldsymbol{k}} \tilde{f}_{s\boldsymbol{k}}, \tilde{q}_{\parallel\parallel s\boldsymbol{k}} = \int dv^3\, v_\parallel \frac{v_\parallel^2}{2} J_{0s\boldsymbol{k}} \tilde{f}_{s\boldsymbol{k}}, \tilde{q}_{\parallel\perp s\boldsymbol{k}} = \int dv^3\, v_\parallel \mu B J_{0s\boldsymbol{k}} \tilde{f}_{s\boldsymbol{k}}$$

【説明】

規格化は

$$\tilde{n}_{s\boldsymbol{k}} = \frac{\rho_{ref}}{L_{ref}} n_{ref} \bar{n}_{s\boldsymbol{k}}, \qquad \tilde{u}_{\parallel s\boldsymbol{k}} = \frac{\rho_{ref}}{L_{ref}} n_{ref} v_{ref} \bar{u}_{\parallel s\boldsymbol{k}}, \qquad \tilde{p}_{\parallel s\boldsymbol{k}} = \frac{\rho_{ref}}{L_{ref}} n_{ref} T_{ref} \bar{p}_{\parallel s\boldsymbol{k}},$$

$$\tilde{p}_{\perp s\boldsymbol{k}} = \frac{\rho_{ref}}{L_{ref}} n_{ref} T_{ref} \bar{p}_{\perp s\boldsymbol{k}}, \qquad \tilde{q}_{\parallel\parallel s\boldsymbol{k}} = \frac{\rho_{ref}}{L_{ref}} n_{ref} T_{ref} v_{ref} \bar{q}_{\parallel\parallel s\boldsymbol{k}}, \qquad \tilde{q}_{\parallel\perp s\boldsymbol{k}} = \frac{\rho_{ref}}{L_{ref}} n_{ref} T_{ref} v_{ref} \bar{q}_{\parallel\perp s\boldsymbol{k}}$$

phi/gkvp.(MPIランク6桁).(粒子種1桁).trn.(ラン数3桁)

- ファイル形式： バイナリ
- 出力間隔： dtout_eng
- 出力を行うMPIランク： zsp_rank == 0 .and. vel_rank == 0
- 総ファイル数： nprocw*nprocs * (総ラン数)
- GKVコード中の出力ユニット： otrn
- 格納データ：

  time, $S_{sk}, W_{Ek}, W_{Mk}, R_{sEk}, R_{sMk}, I_{sEk}, I_{sMk}, D_{sk}, \Gamma_{sEk}, \Gamma_{sMk}, Q_{sEk}, Q_{sMk}$

ここで、

time: 時刻（倍精度実数）

他はすべてサイズ(-nx:nx,0:ny)の倍精度実数配列で、左から順に、ジャイロ中心揺動エントロピー、静電揺動エネルギー（イオン分極項含む）、磁場揺動エネルギー、波粒子相互作用（$W_E \rightarrow S_s$）、波粒子相互作用（$W_M \rightarrow S_s$）、ExB流による非線形エントロピー伝達、磁場揺動による非線形エントロピー伝達、衝突散逸、ExB流による粒子輸送フラックス、磁場揺動による粒子輸送フラックス、ExB流によるエネルギー輸送フラックス、磁場揺動によるエネルギー輸送フラックス

【説明】

補足1. エントロピーバランス方程式を参照。また、規格化は以下とする。

$$S_{sk} = \delta_{ref}^2 n_{ref} T_{ref} \bar{S}_{sk}, \qquad W_{Ek} = \delta_{ref}^2 n_{ref} T_{ref} \bar{W}_{Ek}, \qquad W_{Mk} = \delta_{ref}^2 n_{ref} T_{ref} \bar{W}_{Mk},$$

$$R_{sk} = \delta_{ref}^2 \frac{v_{ref}}{L_{ref}} n_{ref} T_{ref} \bar{R}_{sk}, \qquad I_{sk} = \delta_{ref}^2 \frac{v_{ref}}{L_{ref}} n_{ref} T_{ref} \bar{I}_{sk}, \qquad D_{sk} = \delta_{ref}^2 \frac{v_{ref}}{L_{ref}} n_{ref} T_{ref} \bar{D}_{sk},$$

$$\Gamma_{sk} = \delta_{ref}^2 n_{ref} v_{ref} \bar{\Gamma}_{sk}, \qquad Q_{sk} = \delta_{ref}^2 n_{ref} T_{ref} v_{ref} \bar{Q}_{sk}$$

（シア磁場中の磁力線平行方向移流項E_{sk}については現状評価していない。Note:$\sum_{k_x} \sum_{k_y} I_{sk} = 0,\ \sum_{k_x} E_{sk} = 0$）

phi/gkvp.s(粒子種1桁)mx(mxt4桁)my(myt4桁).tri.(ラン数3桁)

※mxt,mytはnamelistで指定したもの。

- ファイル形式： バイナリ
- 出力間隔： dtout_ptn（calc_type==“nonlinear” .and. num_triad_diag>0）
- 出力を行うMPIランク： rank == 0
- 総ファイル数： nprocs * num_triad_diag * (総ラン数)
- GKVコード中の出力ユニット： otri
- 格納データ：

  time, $J_{sE\boldsymbol{k}}^{\boldsymbol{p},\boldsymbol{q}}$,$J_{sE\boldsymbol{p}}^{\boldsymbol{q},\boldsymbol{k}}$,$J_{sE\boldsymbol{q}}^{\boldsymbol{k},\boldsymbol{p}}$,$J_{sM\boldsymbol{k}}^{\boldsymbol{p},\boldsymbol{q}}$,$J_{sM\boldsymbol{p}}^{\boldsymbol{q},\boldsymbol{k}}$,$J_{sM\boldsymbol{q}}^{\boldsymbol{k},\boldsymbol{p}}$

ここで、

time: 時刻（倍精度実数）

他はすべてサイズ(-nx:nx,-global_ny:global_ny)の倍精度実数配列で、モードk=(mxt,myt)に固定して、p=(px,py)の関数として表したもの（qは-k-pで求まる）。

先の3つは、ExB流の非線形性によるp,qからkへのエントロピー伝達とそのcyclicな入れ替え、後の3つは、磁場揺動の非線形性によるp,qからkへのエントロピー伝達とそのcyclicな入れ替え。

【説明】

補足2．三波結合伝達関数を参照。

規格化は、*.trn.*の非線形エントロピー伝達と同様に、$J_{s\boldsymbol{k}}^{\boldsymbol{p},\boldsymbol{q}} = \delta_{ref}^2 \frac{v_{ref}}{L_{ref}} n_{ref} T_{ref} \bar{J}_{s\boldsymbol{k}}^{\boldsymbol{p},\boldsymbol{q}}$,

hst/gkvp.bln.(粒子種1桁).(ラン数3桁)

- ファイル形式： アスキー
- 出力間隔： dtout_eng
- 出力を行うMPIランク： rank == 0
- 総ファイル数： nprocs * (総ラン数)
- GKVコード中の出力ユニット： obln
- 格納データ：

$$\mathrm{time}, S_s, W_E, W_M, R_{sE}, R_{sM}, I_{sE}, I_{sM}, D_s, \frac{T_s \Gamma_{sE}}{L_{ps}}, \frac{T_s \Gamma_{sM}}{L_{ps}}, \frac{\Theta_{sE}}{L_{Ts}}, \frac{\Theta_{sM}}{L_{Ts}}$$

ここで、

time: 時刻（実数）

S_sからD_sまではサイズ(2)の実数配列（配列要素1,2はそれぞれky/=0成分とky==0成分）で、左から順に、ジャイロ中心揺動エントロピー、静電揺動エネルギー（イオン分極項含む）、磁場揺動エネルギー、波粒子相互作用（$W_E \rightarrow S_s$）、波粒子相互作用（$W_M \rightarrow S_s$）、ExB流による非線形エントロピー伝達、磁場揺動による非線形エントロピー伝達、衝突散逸。残り4つは実数で、エントロピーバランス方程式における、粒子輸送項（ExB流、磁場揺動）、熱輸送項（ExB流、磁場揺動）

【説明】

*.trn.*の項を参照。

hst/gkvp.ges.(粒子種1桁).(ラン数3桁)

- ファイル形式： アスキー
- 出力間隔： dtout_eng
- 出力を行うMPIランク： rank == 0
- 総ファイル数： nprocs * (総ラン数)
- GKVコード中の出力ユニット： oges
- 格納データ：
  time, Γ_{sE}, Γ_{sEk_y}(0:global_ny)

ここで、

time: 時刻（実数）

Γ_{sE}: ExB流による粒子輸送フラックス（実数）

Γ_{sEk_y}: ExB流による粒子輸送フラックスのy方向波数スペクトル（実数配列）

【説明】

ExB流による粒子輸送フラックスは以下で与えられる。

$$\Gamma_{sEk_y} = \sum_{k_x} \Gamma_{sE\boldsymbol{k}}, \qquad \Gamma_{sE\boldsymbol{k}} = \mathrm{Re}\left[\left\langle -\frac{ik_y\phi_{\boldsymbol{k}}}{c_b} n^*_{s\boldsymbol{k}} \right\rangle\right]$$

規格化は

$$\Gamma_{sEk_y} = \delta^2_{ref} n_{ref} v_{ref} \bar{\Gamma}_{sEk_y}$$

hst/gkvp.gem.(粒子種1桁).(ラン数3桁)

- ファイル形式： アスキー
- 出力間隔： dtout_eng
- 出力を行うMPIランク： rank == 0
- 総ファイル数： nprocs * (総ラン数)
- GKVコード中の出力ユニット： ogem
- 格納データ：
  time, Γ_{sM}, Γ_{sMk_y}(0:global_ny)

ここで、

time: 時刻（実数）

Γ_{sM}: 磁場揺動による粒子輸送フラックス（実数）

Γ_{sMk_y}: 磁場揺動による粒子輸送フラックスのy方向波数スペクトル（実数配列）

【説明】

磁場揺動による粒子輸送フラックスは以下で与えられる。

$$\Gamma_{sMk_y} = \sum_{k_x} \Gamma_{sM\boldsymbol{k}}, \qquad \Gamma_{sM\boldsymbol{k}} = \mathrm{Re}\left[\left\langle \frac{ik_y A_{\parallel \boldsymbol{k}}}{c_b} u^*_{\parallel s\boldsymbol{k}} \right\rangle\right]$$

規格化は

$$\Gamma_{sMk_y} = \delta_{ref}^2 n_{ref} v_{ref} \bar{\Gamma}_{sMk_y}$$

hst/gkvp.qes.(粒子種1桁).(ラン数3桁)

- ファイル形式： アスキー
- 出力間隔： dtout_eng
- 出力を行うMPIランク： rank == 0
- 総ファイル数： nprocs * (総ラン数)
- GKVコード中の出力ユニット： oqes
- 格納データ：
  time, Q_{sE}, Q_{sEk_y}(0:global_ny)

ここで、

time: 時刻（実数）

Q_{sE}: ExB流によるエネルギー輸送フラックス（実数）

Q_{sEk_y}: ExB流によるエネルギー輸送フラックスのy方向波数スペクトル（実数配列）

【説明】

ExB流によるエネルギー輸送フラックスは以下で与えられる。

$$Q_{sEk_y} = \sum_{k_x} Q_{sE\boldsymbol{k}}, \qquad Q_{sE\boldsymbol{k}} = \mathrm{Re}\left[\left\langle -\frac{ik_y\phi_{\boldsymbol{k}}}{c_b} p^*_{s\boldsymbol{k}} \right\rangle\right]$$

規格化は

$$Q_{sEk_y} = \delta^2_{ref} n_{ref} v_{ref} \bar{Q}_{sEk_y}$$

hst/gkvp.qem.(粒子種1桁).(ラン数3桁)

- ファイル形式： アスキー
- 出力間隔： dtout_eng
- 出力を行うMPIランク： rank == 0
- 総ファイル数： nprocs * (総ラン数)
- GKVコード中の出力ユニット： oqem
- 格納データ：
  time, Q_{sM}, Q_{sMk_y}(0:global_ny)

ここで、

time: 時刻（実数）

Q_{sM}: 磁場揺動によるエネルギー輸送フラックス（実数）

Q_{sMk_y}: 磁場揺動によるエネルギー輸送フラックスのy方向波数スペクトル（実数配列）

【説明】

磁場揺動によるエネルギー輸送フラックスは以下で与えられる。

$$Q_{sMk_y} = \sum_{k_x} Q_{sM\boldsymbol{k}}, \qquad Q_{sM\boldsymbol{k}} = \mathrm{Re}\left[\left\langle \frac{ik_y A_{\parallel \boldsymbol{k}}}{c_b} q^*_{\parallel s\boldsymbol{k}} \right\rangle\right]$$

規格化は

$$Q_{sMk_y} = \delta^2_{ref} n_{ref} v_{ref} \bar{Q}_{sMk_y}$$

hst/gkvp.wes.(ラン数3桁)

- ファイル形式： アスキー
- 出力間隔： dtout_eng
- 出力を行うMPIランク： rankg == 0
- 総ファイル数： (総ラン数)
- GKVコード中の出力ユニット： owes
- 格納データ：
  time, W_E, W_{Ek_y}(0:global_ny)

ここで、

time: 時刻（実数）
W_E： 静電揺動エネルギー（実数）
W_{Ek_y}： 静電揺動エネルギーのy方向波数スペクトル（実数配列）

【説明】
静電揺動エネルギー（分極項含む）は以下で与えられる

$$W_{Ek_y} = \sum_{k_x} W_{E\boldsymbol{k}}, \qquad W_{E\boldsymbol{k}} = \left\langle \left[\varepsilon_0 k_\perp^2 + \sum_s \frac{e_s^2 n_s}{T_s} (1 - \Gamma_{0sk}) \right] \frac{|\phi_{\boldsymbol{k}}|^2}{2} \right\rangle$$

規格化は

$$W_{Ek_y} = \delta_{ref}^2 n_{ref} T_{ref} \bar{W}_{Ek_y}$$

hst/gkvp.wem.(ラン数3桁)

- ファイル形式： アスキー
- 出力間隔： dtout_eng
- 出力を行うMPIランク： rankg == 0
- 総ファイル数： (総ラン数)
- GKVコード中の出力ユニット： owem
- 格納データ：
  time,W_M, W_{Mk_y}(0:global_ny)

ここで、

time: 時刻（実数）
W_M: 磁場揺動エネルギー（実数）
W_{Mk_y}: 磁場揺動エネルギーのy方向波数スペクトル（実数配列）

【説明】
磁場揺動エネルギーは以下で与えられる。

$$W_{Mk_y} = \sum_{k_x} W_{M\boldsymbol{k}}, \qquad W_{M\boldsymbol{k}} = \left\langle \frac{k_\perp^2}{\mu_0} \frac{\left|A_{||\boldsymbol{k}}\right|^2}{2} \right\rangle$$

規格化は

$$W_{Mk_y} = \delta_{ref}^2 n_{ref} T_{ref} \bar{W}_{Mk_y}$$

hst/gkvp.eng.(ラン数3桁)

- ファイル形式： アスキー
- 出力間隔： dtout_eng
- 出力を行うMPIランク： rankg == 0
- 総ファイル数： (総ラン数)
- GKVコード中の出力ユニット： oeng
- 格納データ：

  time,$\sum_{k_x,k_y}\langle|\tilde{\phi}_{\boldsymbol{k}}|^2\rangle$, $\sum_{k_x}\langle|\tilde{\phi}_{\boldsymbol{k}}|^2\rangle$(0:global_ny)

ここで、

time: 時刻（実数）

$\sum_{k_x,k_y}\langle|\tilde{\phi}_{\boldsymbol{k}}|^2\rangle$: 揺動静電ポテンシャル二乗振幅（実数）

$\sum_{k_x}\langle|\tilde{\phi}_{\boldsymbol{k}}|^2\rangle$: 揺動静電ポテンシャル二乗振幅のy方向波数スペクトル（実数配列）

【説明】

規格化は、$\tilde{\phi}_{\boldsymbol{k}} = \frac{\rho_{ref}}{L_{ref}}\frac{T_{ref}}{e_{ref}}\bar{\phi}_{\boldsymbol{k}}$

hst/gkvp.men.(ラン数3桁)

- ファイル形式： アスキー
- 出力間隔： dtout_eng
- 出力を行うMPIランク： rankg == 0
- 総ファイル数： (総ラン数)
- GKVコード中の出力ユニット： omen
- 格納データ：

  time,$\sum_{k_x,k_y}\langle|\tilde{A}_{\parallel \boldsymbol{k}}|^2\rangle$, $\sum_{k_x}\langle|\tilde{A}_{\parallel \boldsymbol{k}}|^2\rangle$(0:global_ny)

ここで、

time: 時刻（実数）

$\sum_{k_x,k_y}\langle|\tilde{A}_{\parallel \boldsymbol{k}}|^2\rangle$: 揺動ベクトルポテンシャル二乗振幅（実数）

$\sum_{k_x}\langle|\tilde{A}_{\parallel \boldsymbol{k}}|^2\rangle$: 揺動ベクトルポテンシャル二乗振幅のy方向波数スペクトル（実数配列）

【説明】

規格化は、$\tilde{A}_{\parallel \boldsymbol{k}} = \frac{\rho_{ref}}{L_{ref}}\rho_{ref}B_{ref}\bar{A}_{\parallel \boldsymbol{k}}$

hst/gkvp.dtc.(ラン数3桁)

- ファイル形式： アスキー
- 出力間隔： dtout_eng
- 出力を行うMPIランク： rankg == 0
- 総ファイル数： (総ラン数)
- GKVコード中の出力ユニット： odtc
- 格納データ：
  time, dt, dt_limit, dt_nl

ここで、

time: 時刻（実数）
dt: 時間刻み幅（実数）
dt_limit: 時間刻み幅の見積もり（実数）
dt_nl: 非線形移流速度から算出した数値安定な時間刻み幅の見積もり（実数）

【説明】
省略。

hst/gkvp.mtr.(ラン数3桁)

- ファイル形式： アスキー
- 出力間隔： ランの開始時
- 出力を行うMPIランク： rankg == 0
- 総ファイル数： (総ラン数)
- GKVコード中の出力ユニット： omtr
- 格納データ：

$$z, \theta(\text{または}\varphi), B, \frac{\partial B}{\partial x}, \frac{\partial B}{\partial y}, \frac{\partial B}{\partial z}, g^{xx}, g^{xy}, g^{xz}, g^{yy}, g^{yz}, g^{zz}, \sqrt{g}$$

ここで、データはすべて実数で、左から順に
磁力線方向座標、ポロイダル角（ただしequib_type==vmecの時はトロイダル角）、磁場強度、磁場強度の微分3つ、メトリックテンソルの要素6つ、Jacobian。

【説明】
省略。

hst/gkvp.frq.(ラン数3桁)

- ファイル形式： アスキー
- 出力間隔： dtout_eng (calc_type == linear .or. calc_type == lin_freq)
- 出力を行うMPIランク： rankg == 0
- 総ファイル数： (総ラン数)
- GKVコード中の出力ユニット： ofrq
- 格納データ：
  time, omega(1:global_ny,-nxfrq:nxfrq)

ここで、

time: 時刻（実数）

omega: 線形複素周波数（複素数）[＝（実周波数，成長率）]のkx,ky方向波数スペクトル

【説明】

$\tilde{\phi}_{\boldsymbol{k}}(t) = \phi_0 e^{-i\omega t} = \phi_0 e^{-i\omega_r t} e^{\gamma t}$の依存性を仮定して、

$$\omega = \omega_r + i\gamma = \frac{\ln\left[\dfrac{\tilde{\phi}_{\boldsymbol{k}}(t+\Delta t)}{\tilde{\phi}_{\boldsymbol{k}}(t)}\right]}{-i\Delta t}$$

により、線形複素周波数の時々刻々の見積もりを得る。

複数のkx,kyモードについてω_rとγの時間発展を出力しているので、どの列にどのモードの値が出力されているかはファイル先頭のコメント行を参照。

hst/gkvp.dsp.(ラン数3桁)

- ファイル形式： アスキー
- 出力間隔： ランの終了時 (calc_type == linear .or. calc_type == lin_freq)
- 出力を行うMPIランク： rankg == 0
- 総ファイル数： (総ラン数)
- GKVコード中の出力ユニット： odsp
- 格納データ：
  kx, ky, omega, diff, 1-ineq

ここで、

kx: x方向波数（実数）
ky: y方向波数（実数）
omega: 線形複素周波数（複素数）[＝（実周波数，成長率）]
diff: 単位時間当たり相対誤差による収束判定([omega(t)-omega(t-dt)]/dt)/omega(t) （複素数）
1-ineq: Schwartzの不等式で評価した収束誤差（実数）

【説明】

$\tilde{\phi}_{\boldsymbol{k}}(t) = \phi_0 e^{-i\omega t} = \phi_0 e^{-i\omega_r t} e^{\gamma t}$の依存性を仮定して、

$$\omega = \omega_r + i\gamma = \frac{\ln\left[\dfrac{\tilde{\phi}_{\boldsymbol{k}}(t+\Delta t)}{\tilde{\phi}_{\boldsymbol{k}}(t)}\right]}{-i\Delta t}$$

により、線形複素周波数の時々刻々の見積もりを得る。
ランの終了時に、周波数・成長率のkx,ky依存性を出力する。まだ周波数・成長率が十分収束していなさそうな場合はコメントとして書き出す。

log/gkvp.(MPIランク6桁).(粒子種1桁).log.(ラン数3桁)

- ファイル形式： アスキー
- 出力間隔： 随時
- 出力を行うMPIランク： すべて
- 総ファイル数： nprocw*nprocz*nprocv*nprocm*nprocs * (総ラン数)
- GKVコード中の出力ユニット： olog
- 格納データ：
  シミュレーションに関するログ

【説明】
省略。

Appendix B. その他補足

補足1. エントロピーバランス方程式

各粒子種、各モードについてのエントロピーバランスは以下で与えられる[Maeyama'14PoP]。

$$\frac{dS_{sk}}{dt} = \frac{T_s\Gamma_{sk}}{L_{ps}} + \frac{\Theta_{sk}}{L_{Ts}} + I_{sk} + R_{sk} + E_{sk} + D_{sk}, \qquad \frac{dW_{Ek}}{dt} = -R_{sEk}, \qquad \frac{dW_{Mk}}{dt} = -R_{sMk}$$

ここで、

$$S_{sk} = \left\langle \int dv^3 \frac{T_s|f_{sk}|^2}{2F_{sM}} \right\rangle, \qquad W_{Ek} = \left\langle \left[\varepsilon_0 k_\perp^2 + \sum_s \frac{e_s^2 n_s}{T_s}(1-\Gamma_{0sk}) \right] \frac{|\phi_k|^2}{2} \right\rangle, \qquad W_{Mk} = \left\langle \frac{k_\perp^2}{\mu_0} \frac{|A_{||k}|^2}{2} \right\rangle,$$

$$\Gamma_{sk} = \Gamma_{sEk} + \Gamma_{sMk} = \mathrm{Re}\left[\left\langle -\frac{ik_y\phi_k}{c_b} n_{sk}^* + \frac{ik_y A_{||k}}{c_b} u_{||sk}^* \right\rangle \right], \qquad \Theta_{sk} = Q_{sk} - \frac{5}{2} T_s \Gamma_{sk}$$

$$Q_{sk} = \mathrm{Q}_{sEk} + \mathrm{Q}_{sMk} = \mathrm{Re}\left[\left\langle -\frac{ik_y\phi_k}{c_b} p_{sk}^* + \frac{ik_y A_{||k}}{c_b} q_{||sk}^* \right\rangle \right],$$

$$I_{sk} = I_{sEk} + I_{sMk} = \mathrm{Re}\left[-\sum_{k'} \sum_{k''} \delta_{k'+k'',k} \left\langle \int dv^3 \frac{T_s g_{sk}^*}{F_{sM}} \left\{ \frac{J_{0sk'}(\phi_{k'} - v_{||}A_{||k'})}{c_b}, g_{sk''} \right\}_\perp \right\rangle \right],$$

$$R_{sk} = R_{sEk} + R_{sMk} = \mathrm{Re}\left[\left\langle -\phi_k^* \frac{\partial e_s n_{sk}}{\partial t} - e_s u_{||sk}^* \frac{\partial A_{||k}}{\partial t} \right\rangle \right], \qquad E_{sk} = \mathrm{Re}\left[-\left\langle \int dv^3 \, v_\parallel \nabla_\parallel \frac{T_s|g_{sk}|^2}{2F_{sM}} \right\rangle \right],$$

$$D_{sk} = \mathrm{Re}\left[\left\langle \int dv^3 \frac{T_s g_{sk}^*}{F_{sM}} C_{sk} \right\rangle \right], \qquad g_{sk} = f_{sk} + \frac{e_s J_{0sk}\phi_k}{T_s} F_{sM}, \qquad p_{sk} = p_{\parallel sk} + p_{\perp sk}, \qquad q_{\parallel sk} = q_{\parallel\parallel sk} + q_{\parallel\perp sk}$$

補足2. 三波結合伝達関数

補足1. エントロピーバランス方程式で説明した非線形エントロピー伝達I_{sk}は三波結合伝達関数$J_{\boldsymbol{sk}}^{\boldsymbol{p,q}}$を用いて、

$$I_{\boldsymbol{sk}} = \sum_{\boldsymbol{p}} \sum_{\boldsymbol{q}} J_{\boldsymbol{sk}}^{\boldsymbol{p,q}}$$

ここで、$\chi_{\boldsymbol{sk}} = J_{0\boldsymbol{sk}}\left(\tilde{\phi}_{\boldsymbol{k}} - v_{\parallel}\tilde{A}_{\parallel\boldsymbol{k}}\right)$, $g_{\boldsymbol{sk}} = f_{\boldsymbol{sk}} + \frac{e_s F_{sM}}{T_s} J_{0\boldsymbol{sk}}\tilde{\phi}_{\boldsymbol{k}}$を用いて、

$$J_{\boldsymbol{sk}}^{\boldsymbol{p,q}} = J_{sE\boldsymbol{k}}^{\boldsymbol{p,q}} + J_{sM\boldsymbol{k}}^{\boldsymbol{p,q}} = \delta_{\boldsymbol{k+p+q,0}} \frac{\boldsymbol{b}\cdot\boldsymbol{p}\times\boldsymbol{q}}{2B} \mathrm{Re}\left[\left\langle \int dv^3 \left(\chi_{s\boldsymbol{p}} g_{s\boldsymbol{q}} - \chi_{s\boldsymbol{q}} g_{s\boldsymbol{p}}\right) \frac{T_s g_{\boldsymbol{sk}}}{F_{sM}} \right\rangle\right]$$

補足3. GKVの積分

磁気面平均

$$\langle \tilde{\phi}(x,y,z) \rangle = \sum_{k_x} \left\langle \tilde{\phi}_{k_x,k_y=0}(z) \right\rangle e^{ik_x x}, \qquad \left\langle \tilde{\phi}_{k_x,k_y=0}(z) \right\rangle = \frac{\int_{-\pi}^{\pi} dz \sqrt{g} \tilde{\phi}_{k_x,k_y=0}(z)}{\int_{-\pi}^{\pi} dz \sqrt{g}}$$

体積平均

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

速度空間積分

$$\int dv^3 \tilde{f}_{\boldsymbol{k}}(z, v_\parallel, \mu) = \int_{-v_{max}}^{v_{max}} dv_\parallel \int_0^{v_{max}} dv_\perp \, 2\pi v_\perp \tilde{f}_{\boldsymbol{k}}(z, v_\parallel, \mu)$$

補足4.バッチジョブスクリプトsub.qを設定する。

シェル環境によってはsub.q内moduleコマンドがうまく働かない場合があるようです。(tcsh等)

【対処方法】sub.q 内に以下のコマンドを加えることで陽にmoduleコマンドを有効化できます。

※ Module load NECNLC-sx の直前に追加。

run/sub.q

```
…
source /etc/profile.d/modules.sh
…
```