BO User Guide

-A powerful fluid and kinetic plasma wave and instability analysis tool

Hua-sheng XIE (谢华生), huashengxie@gmail.com

CCF-ENN

2019-03-07

This guide focuses on the kinetic part of BO, i.e., BO-K (PDRK).

Introduction

- Why kinetic dispersion relation important? – Waves and instabilities are one of the most important feature of plasma.

- Why difficult to solve? –Many branches, difficult to convergent in some parameters.

- What is BO-K?– The first kinetic plasma dispersion relation solver that can give all the important solutions at one time without requiring initial guess for root finding.

BO（‘波’，i.e., ‘wave’ in Chinese) is a state-of-art plasma wave and instability analysis tool. It currently includes two codes, the BO-F (PDRF, A general dispersion relation solver for multi-fluid plasma) and BO-K (PDRK, A general kinetic dispersion relation solver for both magnetized and unmagnetized plasma).

Equations

$$\frac{\partial F_s}{\partial t} + \boldsymbol{v}\cdot\frac{\partial F_s}{\partial \boldsymbol{r}} + \left[\boldsymbol{a}_s + \frac{q_s}{m_s}(\boldsymbol{E}+\boldsymbol{v}\times\boldsymbol{B})\right]\cdot\frac{\partial F_s}{\partial \boldsymbol{v}} = -\nu_s(F_s - F_{s0}), \qquad \begin{aligned}\partial_t\boldsymbol{E} &= c^2\nabla\times\boldsymbol{B} - \boldsymbol{J}/\epsilon_0,\\ \partial_t\boldsymbol{B} &= -\nabla\times\boldsymbol{E},\end{aligned}$$

BO-K (PDRK) v190307 solves uniform plasma dispersion relation with an extended Maxwellian based equilibrium distribution function.

Using Matlab, via matrix transformation method. (Also python version provided by Dr. Xin TAO at USTC)

For D(omega, k)=0, give k, solve series omega(s).

D(omega,k) is too complicated and not shown here.

where $\omega_{cs} = \frac{q_s B_0}{m_s}$ is the cyclone frequency. Without loss of generality, we can assume[2] the wave vector $\boldsymbol{k} = (k_x, 0, k_z) = (k\sin\theta, 0, k\cos\theta)$, which gives $k_\perp = k_x$ and $k_\parallel = k_z$. We study four cases:

1. Electromagnetic or Darwin, magnetized: $B_0 \neq 0$, $\boldsymbol{B}_1 \neq 0$.
2. Electromagnetic or Darwin, unmagnetized: $B_0 = 0$ ($\omega_{cs} = 0$), $\boldsymbol{B}_1 \neq 0$.
3. Electrostatic, magnetized: $B_0 \neq 0$, $\boldsymbol{B}_1 = 0$ ($\boldsymbol{k}\times\boldsymbol{E}_1 = 0$).
4. Electrostatic, unmagnetized: $B_0 = 0$ ($\omega_{cs} = 0$), $\boldsymbol{B}_1 = 0$ ($\boldsymbol{k}\times\boldsymbol{E}_1 = 0$).

For convenient to theoretical study, we let the user to choose whether a species is magnetized or unmagnetized[3], i.e., say for a electromagnetic case, the different species can be either magnetized (labeled as 'm') or unmagnetized (labeled as 'u').

2.1 Equilibrium distribution function

We assume equilibrium distribution function $F_{s0}(v'_\parallel, v'_\perp) = n_{s0} f_{s0}(v'_\parallel, v'_\perp)$, with $v'_\parallel = v_z$, $v'_\perp = \sqrt{(v_x - v_{dsx})^2 + (v_y - v_{dsy})^2}$, and

$$f_{s0}(v'_\parallel, v'_\perp) = f_{s0z}(v'_\parallel) f_{s0\perp}(v'_\perp) \tag{62}$$

$$= \frac{1}{\pi^{3/2} v_{zts} v_{\perp ts}^2}\exp\left[-\frac{(v'_\parallel - v_{dsz})^2}{v_{zts}^2}\right]\left\{\frac{r_{sa}}{A_{sa}}\exp\left[-\frac{(v'_\perp - v_{dsr})^2}{v_{\perp ts}^2}\right] + \frac{r_{sb}}{\alpha_s A_{sb}}\exp\left[-\frac{(v'_\perp - v_{dsr})^2}{\alpha_s v_{\perp ts}^2}\right]\right\},$$

where $r_{sa} = \left(\frac{1-\alpha_s\Delta_s}{1-\alpha_s}\right)$ and $r_{sb} = \left(\frac{-\alpha_s+\alpha_s\Delta_s}{1-\alpha_s}\right)$, and

Solvers compare

	WHAMP [Ronnmark1982]	NHD [Verscharen2018]	KUPDAP [Sugiyama2015]	DSHARK [Astfalk2015]	BO (PDRK) [Xie2016,2019]
Initial guess?	Must	Must	Must	Must	Not required
Fast?	Fast	Middle	Middle	Middle	Middle
Support high harmonic?	Difficult	?	Difficult	Difficult	Easy
Separate modes?	Difficult	?	Difficult	Difficult	Easy
All solutions?	No	No	Multi solutions in given range	No	Yes
With perp drift?	No	No	No	No	Yes
With collision?	No	No	No	No	Yes
Key feature	Fast, widely used	?	Multi solutions	Kappa distribution	All solutions, rich models

What makes BO attractive? It solves the difficulty of root finding, i.e., not requires initial guess and can give all the important solutions at one time. You do not need luck any more.

New version supports: anisotropic temperature / loss cone / drift in arbitrary direction / ring beam / collision, unmagnetized / magnetized species, electrostatic/electromagnetic/Darwin, can kpara<=0, etc.

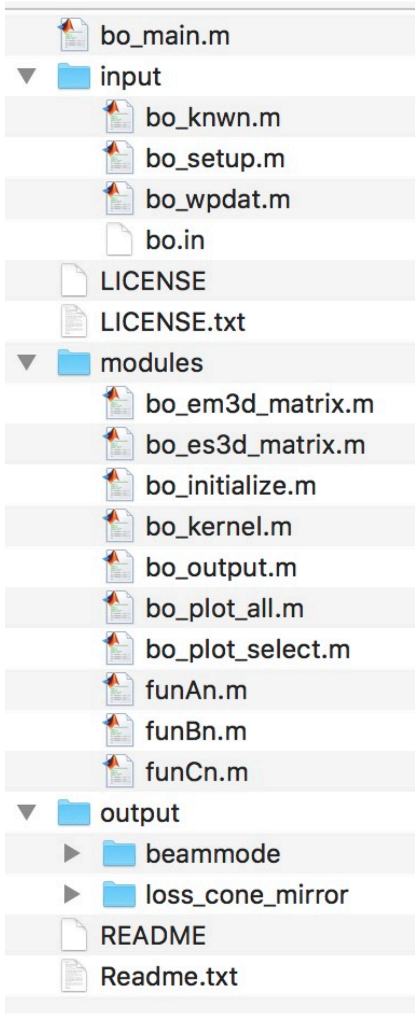
Code structure of BO

- Doc

- Code
  -input file – modify them for your cases
  -main.m – run this file to start
  -modules – do not need change for most cases
  -output – output data/figure and a copy of input files

- License – BSD

- Readme



Steps to run BO

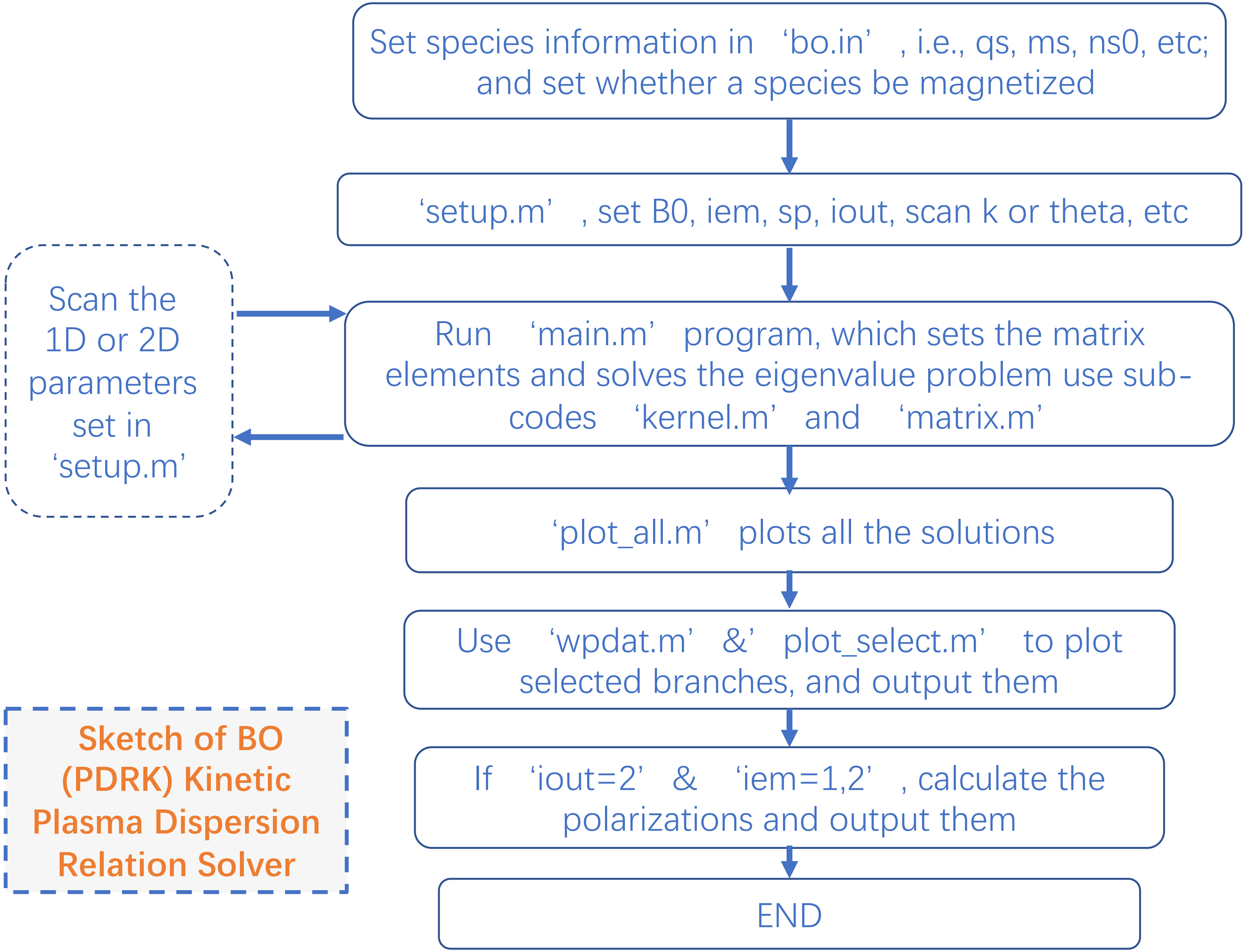
Note, v190307 has provided a unified version.

- ✓ EM3D: iem=1;
- ✓ Darwin: iem=2;
- ✓ ES3D: iem=0;
- ✓ ES1D: iem=0, theta=0.
- ✓ Magnetized or unmagnetized species: imus=0,1

Thus, it is extremely simple to switch between different models.

- 1. Set species parameters in ‘bo.in’ ;
- 2. Set ‘setup.m’ , B0, k, theta, etc;
- 3. Run ‘main.m’ ;
- 4. After run ‘plot_all.m’ , zoom in and select which branch(es) to further plot;
- 5. Set the ‘wpdat’ in ‘wpdat.m’ , ‘plot_select.m’ will search the solutions in the same branches in ‘wpdat’ , and then store and plot them;
- 6. If you require polarization info, run ‘output.m’

Sketch of BO (PDRK) Kinetic Plasma Dispersion Relation Solver



iem:
=0 electrostatic run;
=1, electromagnetic run;
=2, Darwin run.

sp:
=0, solve all the solutions;
=1, solve nw solutions around initial guess wg;
=2, solve nw=1 solution of each branches with multi initial guess wg.

'plot_select.m' will search the solutions in same branches specialized in 'wpdat' automatically

Pade J-pole:
=8, enough for most cases
=4,3,2, fast, less artificial solutions
=12,16,24, accurate

Typical cases 1: Cold plasma

bo.in

qs(e)	ms(mp)	ns(m^-3)	Tzs(eV)	Tps(eV)	alphas	Deltas	vds/c
1	1	8.7e6	2.857e-3	2.857e-3	1.0	1.0	0.0
-1	5.447e-4	8.7e6	2.857e-3	2.857e-3	1.0	1.0	0.0

B0=100.0E-9; N=1; J=8; iem=1;

(ipa,ipb) = (1,1) scan k, fixed theta=60

pa=0:0.5:100

Default unit: SI
Normalization:
wn=abs(wcs(1));
cwp=sqrt(c2)/wps(1);
kn=1/cwp;

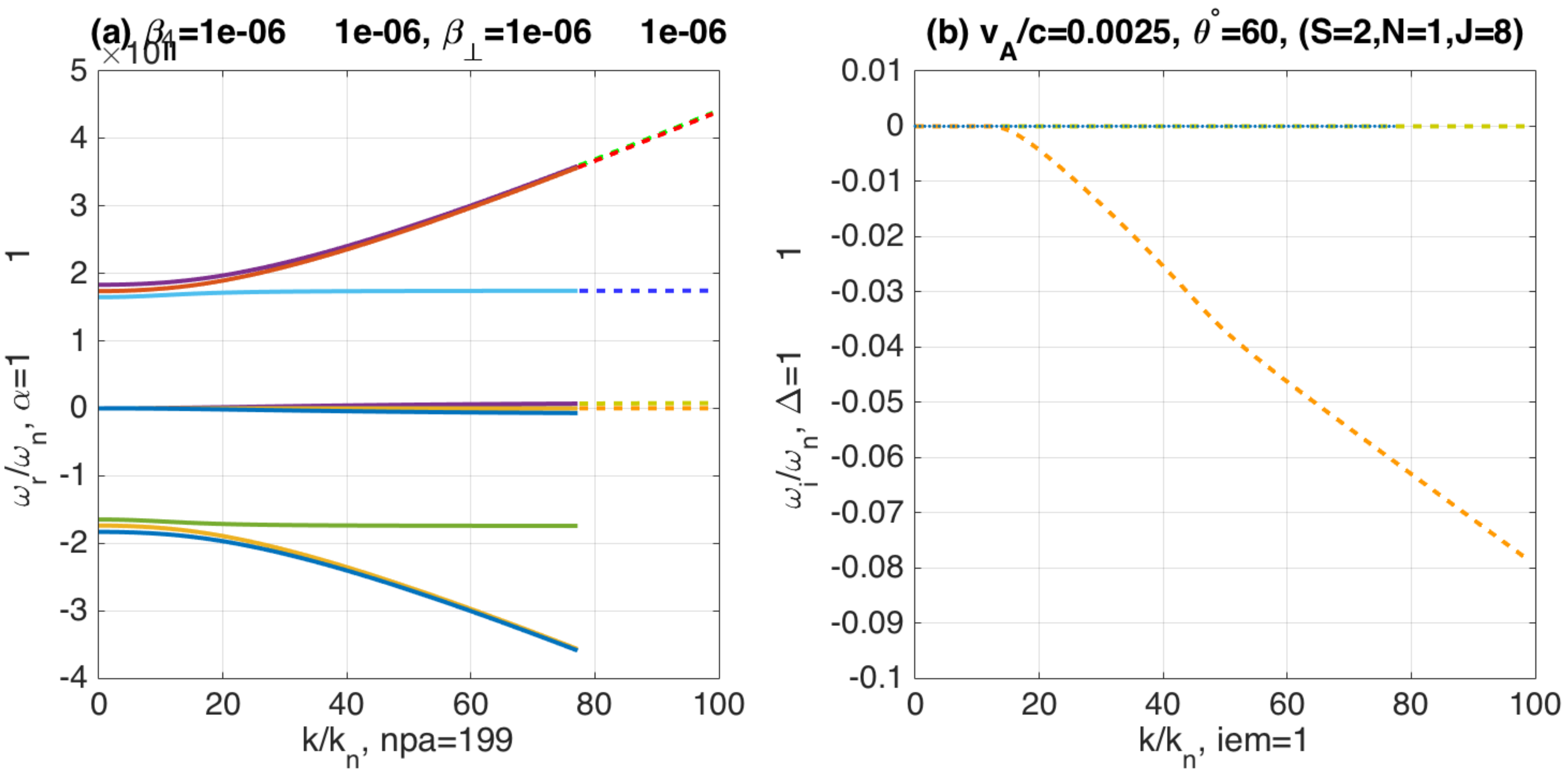
bo_main.m | bo_knwn.m | bo_setup.m | bo_wpdat.m | bo.in | pdrf_SI.m | +

qs(e)	ms(m_unit)	ns(m^-3)	Tzs(eV)	Tps(eV)	alphas	Deltas	vdsz/c	vdsx/c	vdsy/c	vdsr/c	nu_s	m_or_u(1/0)
1	1	8.7e6	2.857e-3	2.857e-3	1.0	1.0	0.0	0.0	0.0	0.0	0.0	1
-1	5.447e-4	8.7e6	2.857e-3	2.857e-3	1.0	1.0	0.0	0.0	0.0	0.0	0.0	1

Solid lines: Fluid solver BO-F (PDRF) results.

Dash lines: BO-K results.

We find good agreement, except a slight difference at large k for the ion cyclotron wave, which is damped due to kinetic effect.



Typical cases 2: Loss cone mirror

v181027

bo.in

qs(e)	ms(mp)	ns(m^-3)	Tzs(eV)	Tps(eV)	alphas	Deltas	vds/c
1	1	1.e6	24840.	49680.	0.5	0.1	0.0
-1	5.447e-4	1.e6	24840.	24840.	0.5	0.1	0.0

B0=100.0E-9; **N=3;** J=8; iem=1;

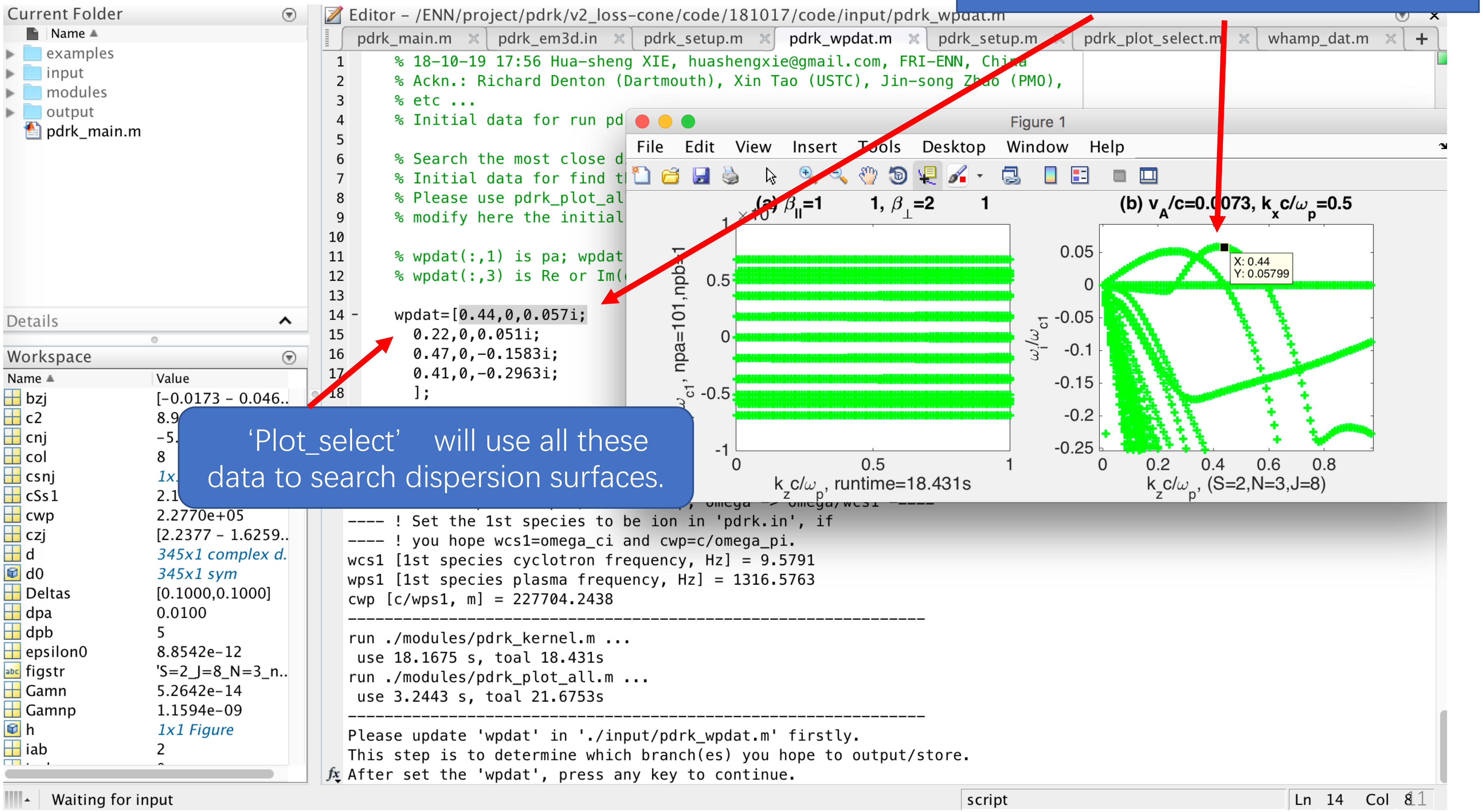
(ipa,ipb) = (3,3) scan kz, fixed kx=0.5

pa=0:0.01:1

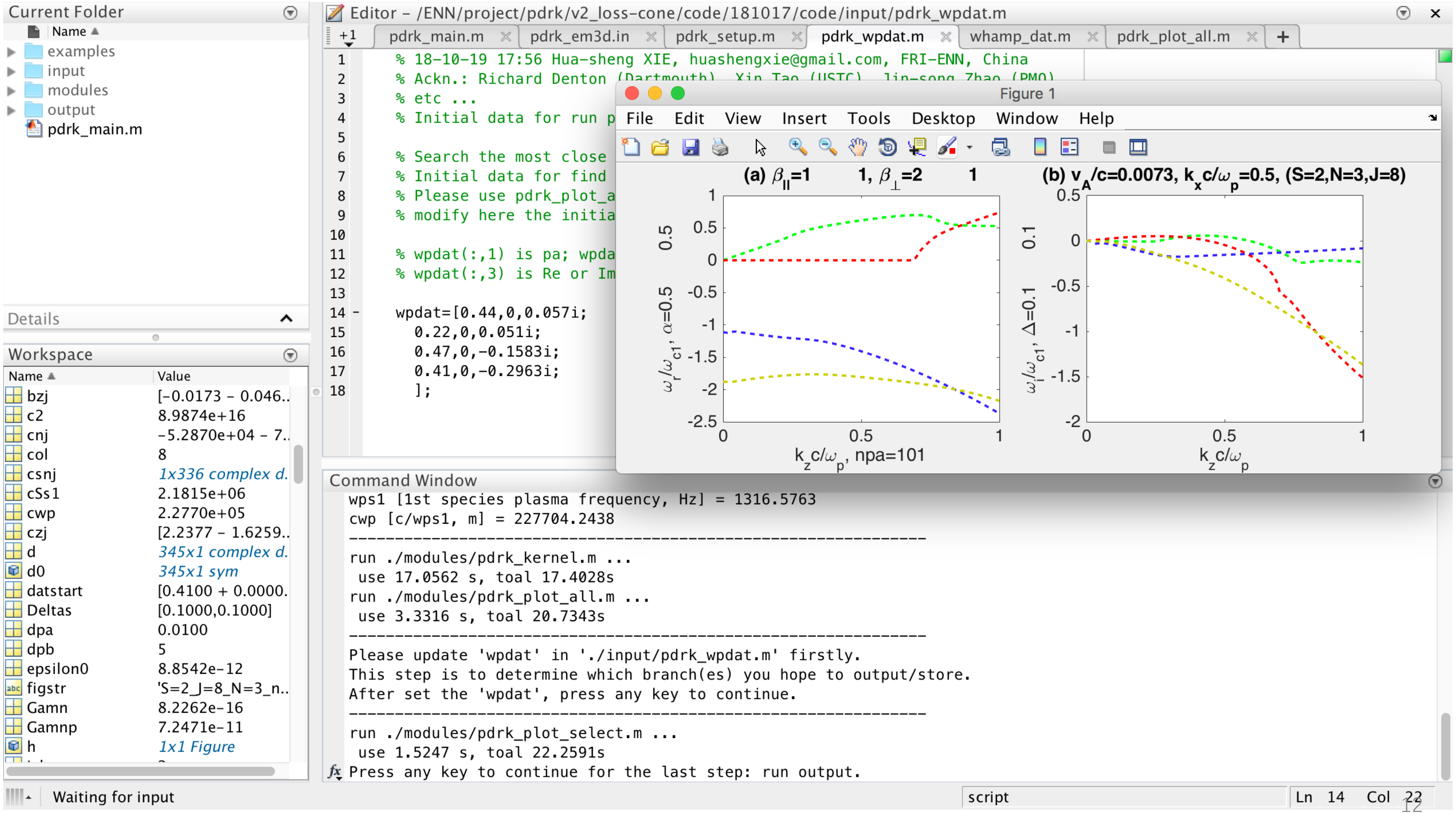
Ensure your results are convergent by trying a larger N.

After run main.m and plot_all.m, zoom in the Fig(b) to select 'wpdat'.

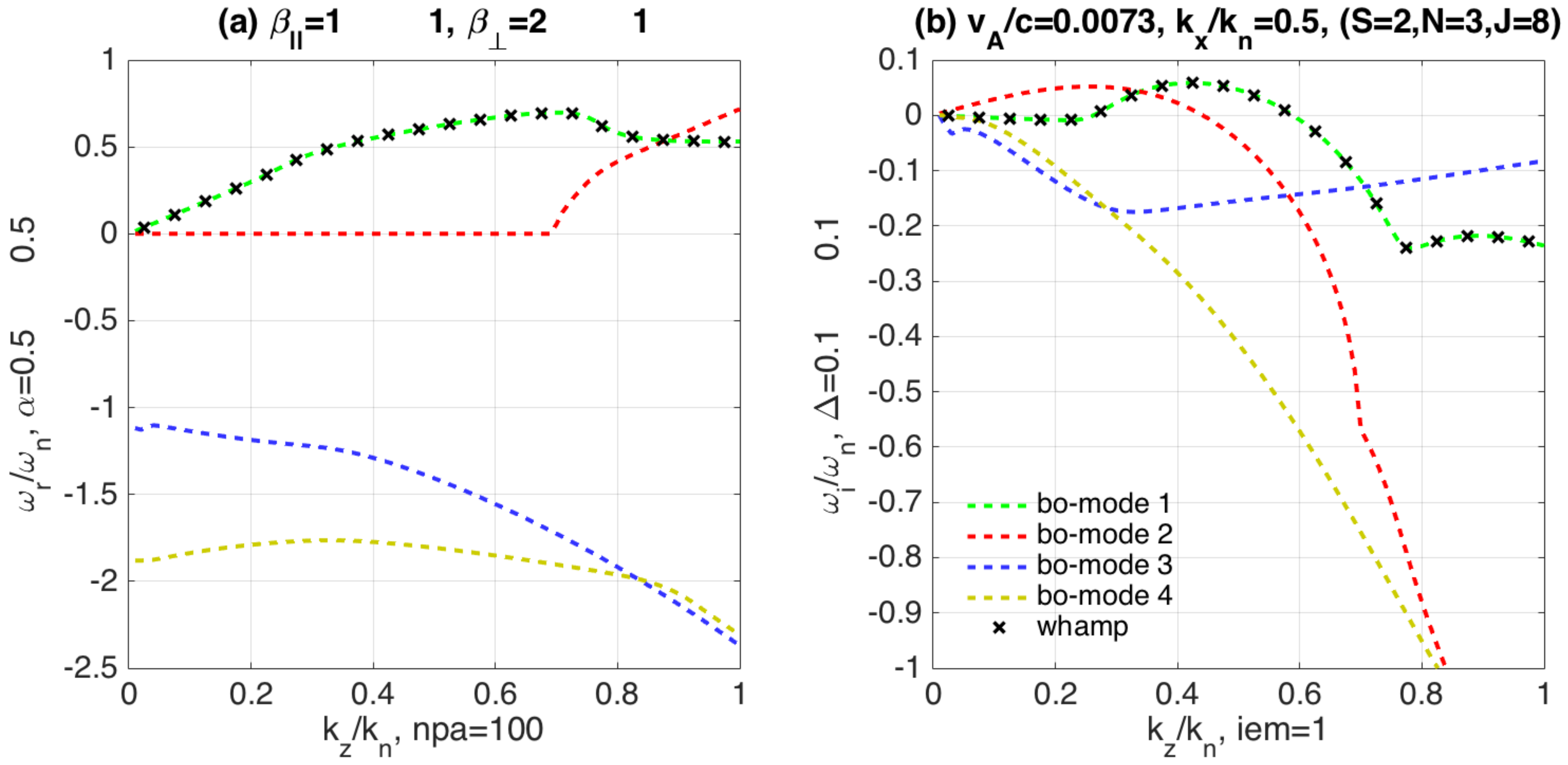
Find a point in 'plot_all' figure and set it into 'pdrk_wpdat.m'



After set 'wpdat' , press enter, we obtain the follow figure.



Compare to WHAMP data, we find good agreement. However, WHAMP can only find one solution at one time and require good initial guess for root finding.



Typical cases 3: Parallel Multi-species Beam mode

bo.in

qs(e)	ms(mp)	ns(m^-3)	Tzs(eV)	Tps(eV)	alphas	Deltas	vdsz/c
1	1	2.528e5	3.5387e4	3.5387e4	1.0	1.0	0.0
-1	5.447e-4	3.16e5	2.831e4	2.831e4	1.0	1.0	3.7013e-3
1	1	3.16e4	28.31e4	28.31e4	1.0	1.0	3.7013e-2
1	2	3.16e4	28.31e4	28.31e4	1.0	1.0	0.0

B0=60.0E-9; N=1; J=8; iem=1;

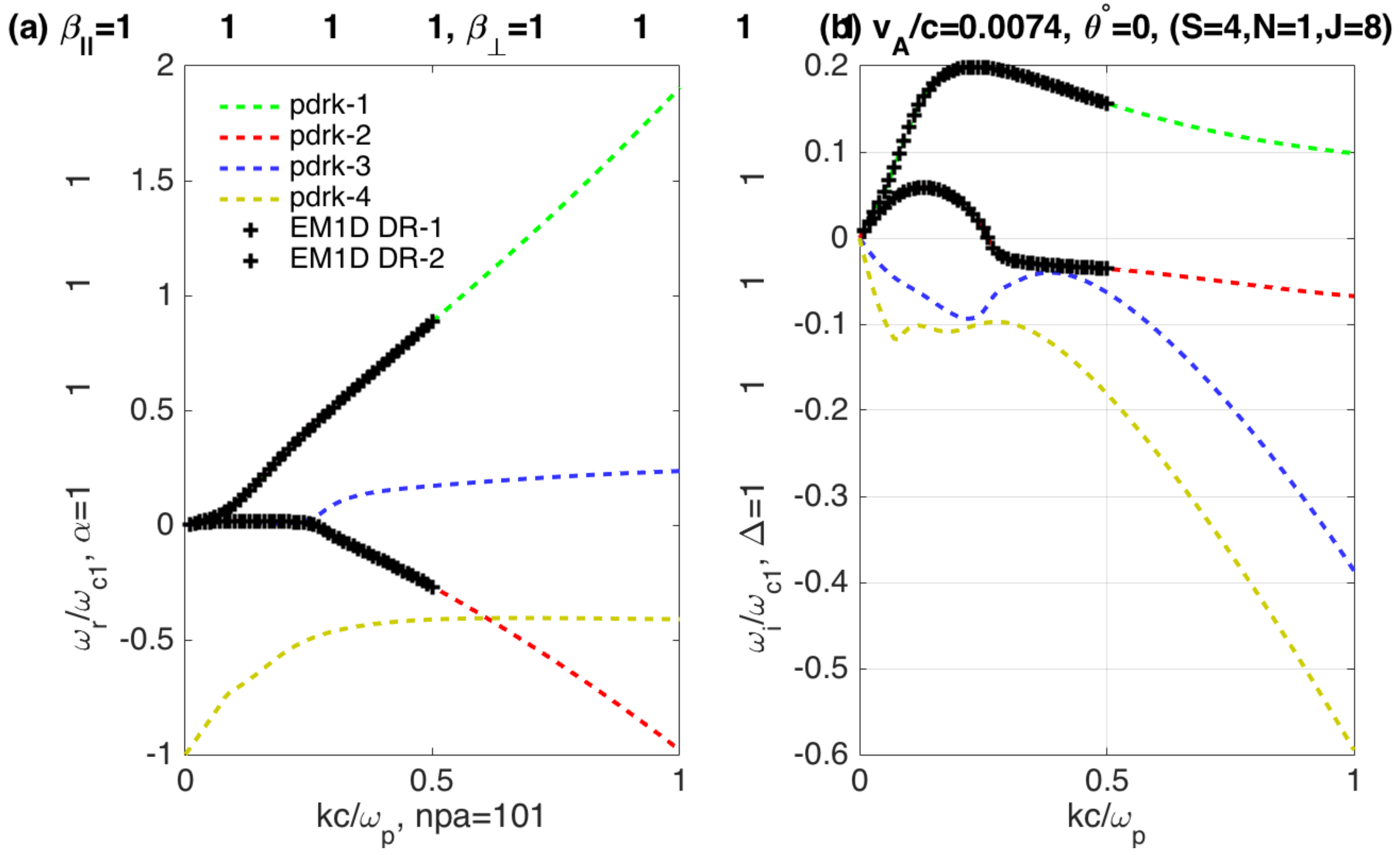
(ipa,ipb) = (1,1) scan k, fixed theta=0

pa=0:0.01:1;

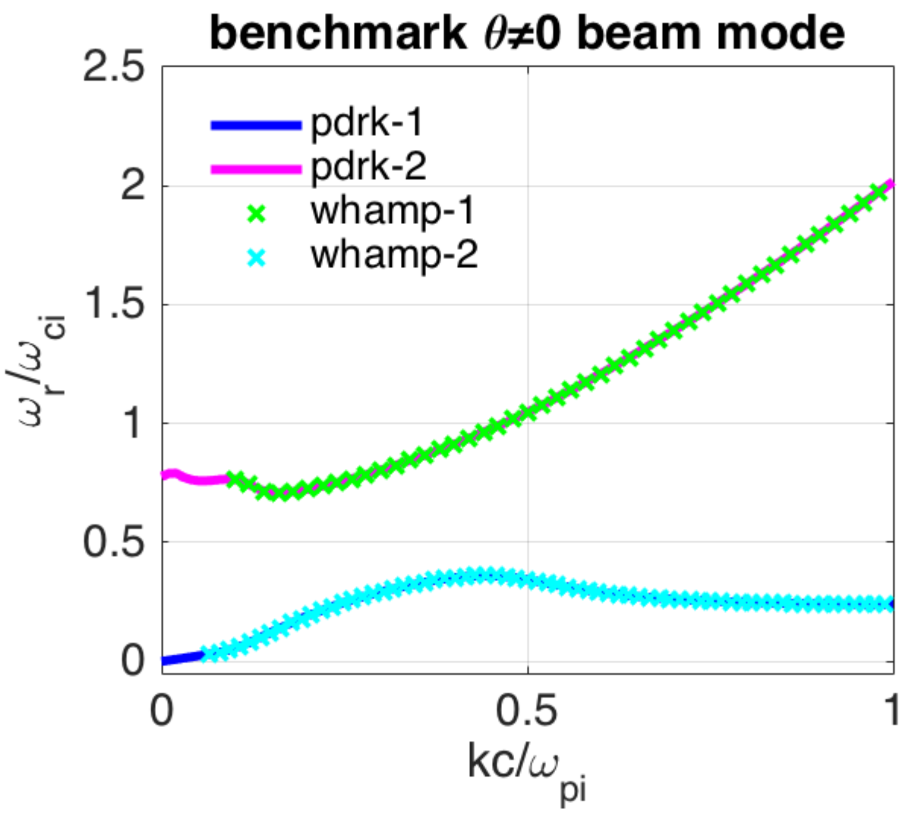
iout=2;

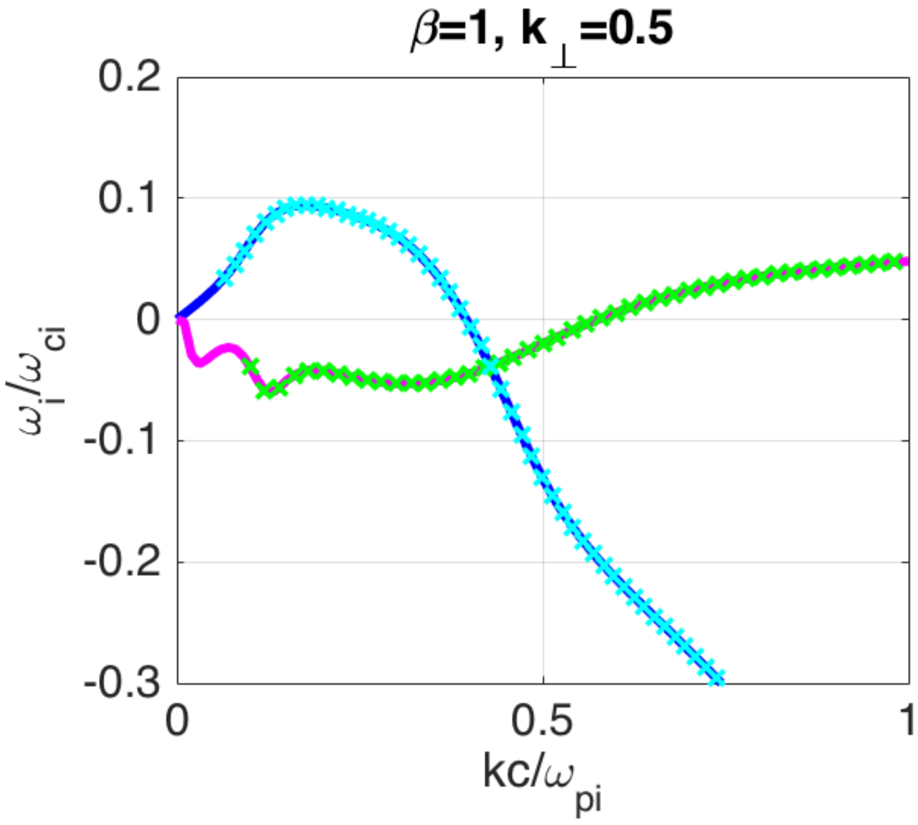
v181027

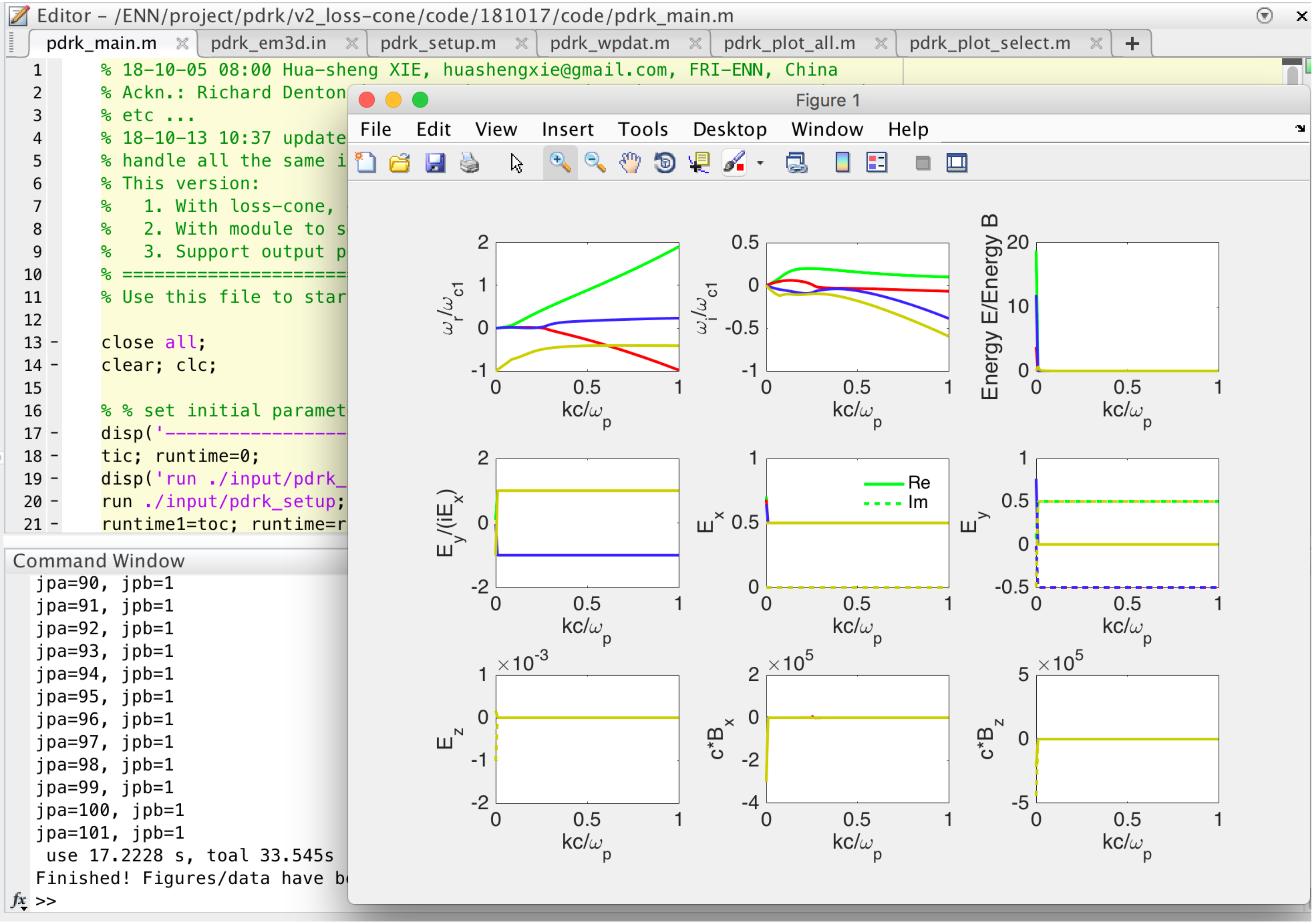
Good agreement to EM1D theta=0 dispersion relation solutions.



Use the above parameter, scan kz, with kperp=0.5 (theta\neq0) is also show, also good agreement with whamp.







Iout=2, polarization are also calculate.

We find Ey/(iE_x)=+1 or -1, i.e., only left and right-hand polarized modes. Agree with theory.

Typical cases 4: Dispersion surface

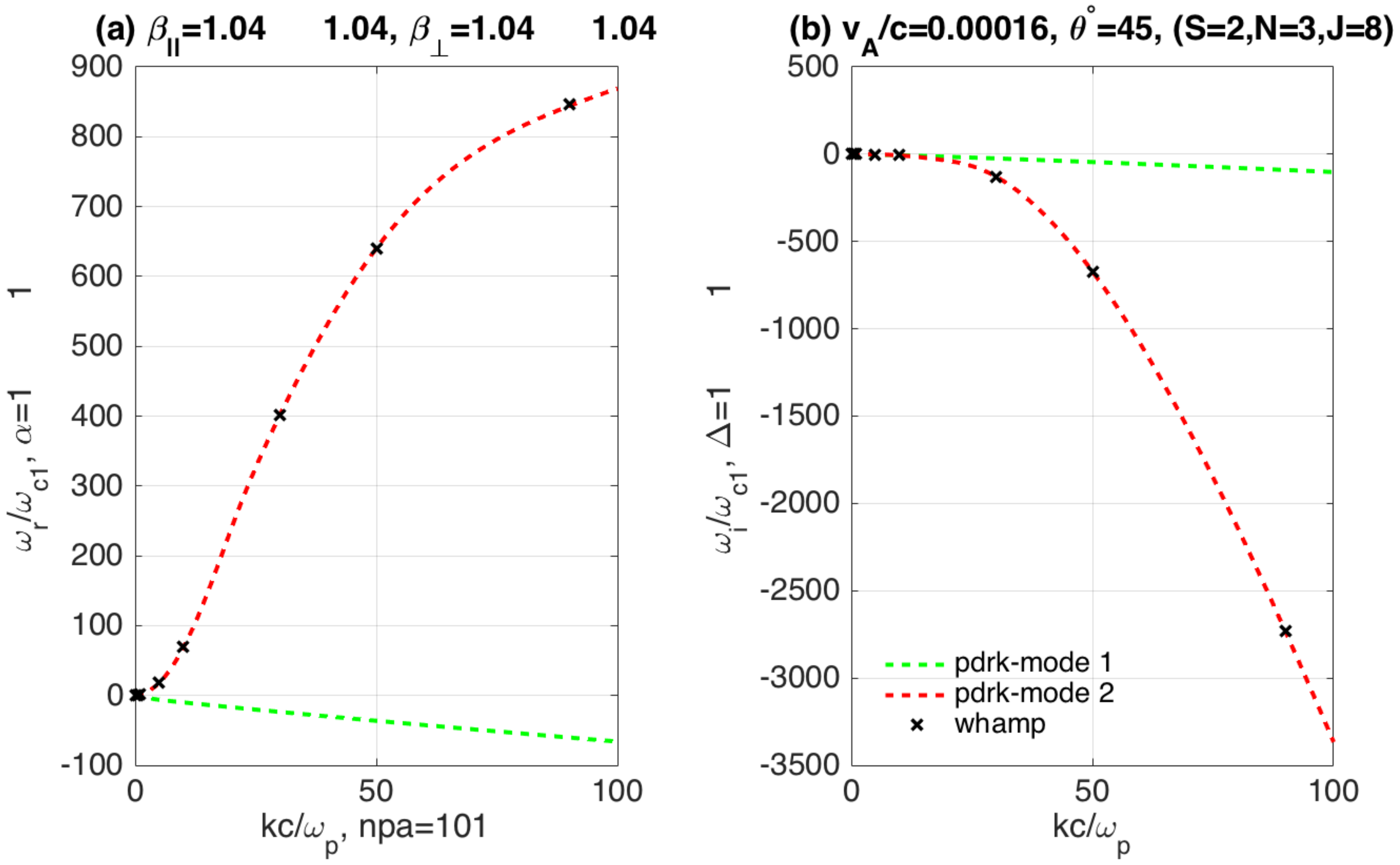
v181027

bo.in

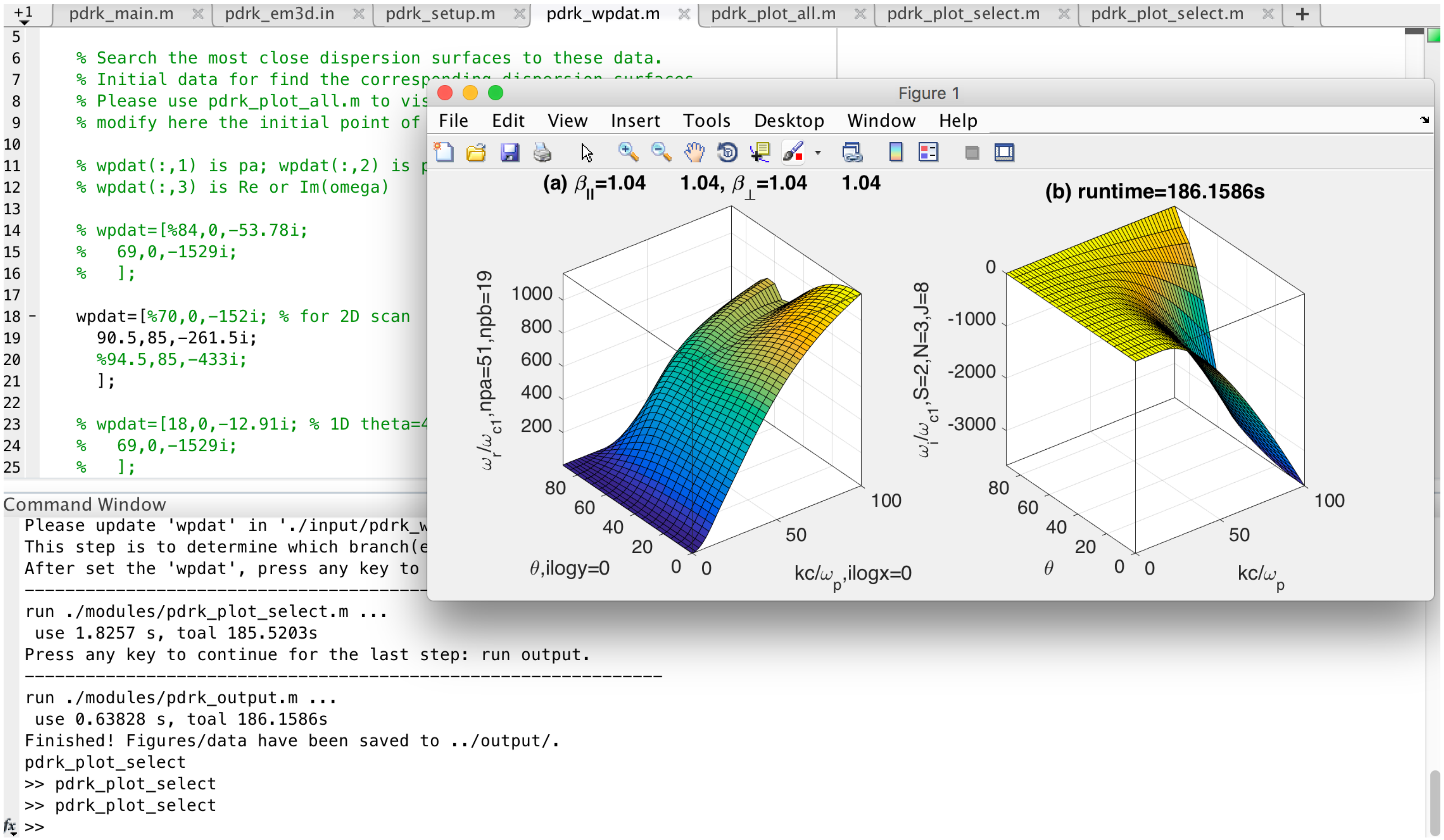
qs(e)	ms(mp)	ns(m^-3)	Tzs(eV)	Tps(eV)	alphas	Deltas	vds/c
1	1	5.e6	12.94	12.94	1.0	1.0	0.0
-1	5.447e-4	5.e6	12.94	12.94	1.0	1.0	0.0

```
B0=5.0E-9; N=3; J=8; iem=1;

(ipa,ipb) = (1,2) scan 2D (k, theta)

pa=0.1:2:100;pb=0:5:90;

iout=0;
```

Firstly, theta=45, 1D scan k agrees with whamp.



The 2D scan with proper ‘wpdat’ gives a nice whistler wave dispersion surface. Give multi ‘wpdat’ can also plot other dispersion surfaces.



Typical cases 5: ES3D loss cone instability

bo.in

qs(e)	ms(mp)	ns(m^-3)	Tzs(eV)	Tps(eV)	alphas	Deltas	vds/c
-1	5.447e-4	1.e6	1.	1.	1.0	1.0	0.0
-1	5.447e-4	1.e6	5.e2	5.e2	0.005	0.0	0.0

B0= 143.5E-9; N=1; J=8; iem=0;

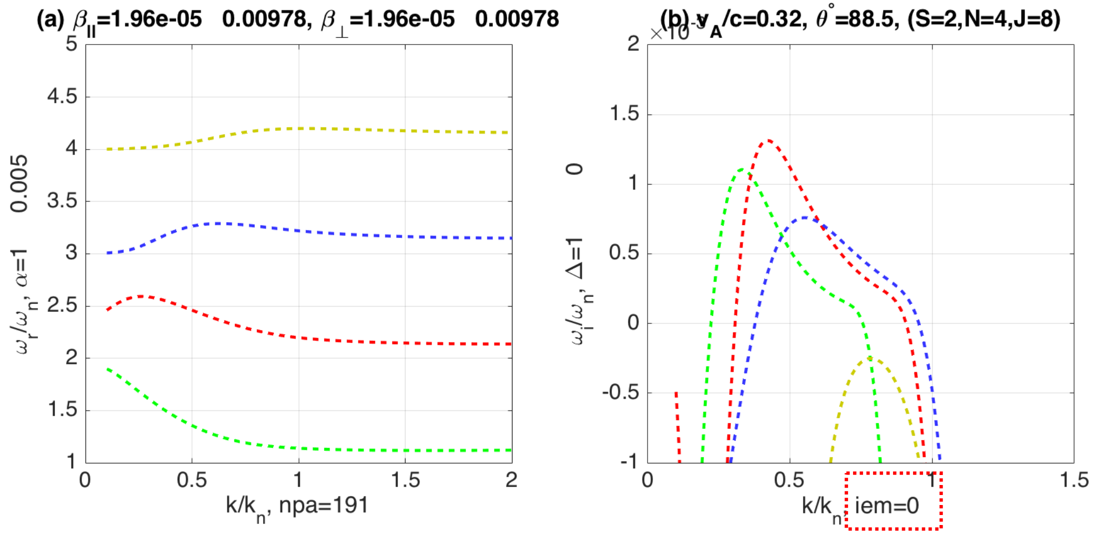
(ipa,ipb) = (1,1) scan k, fixed theta=88.5

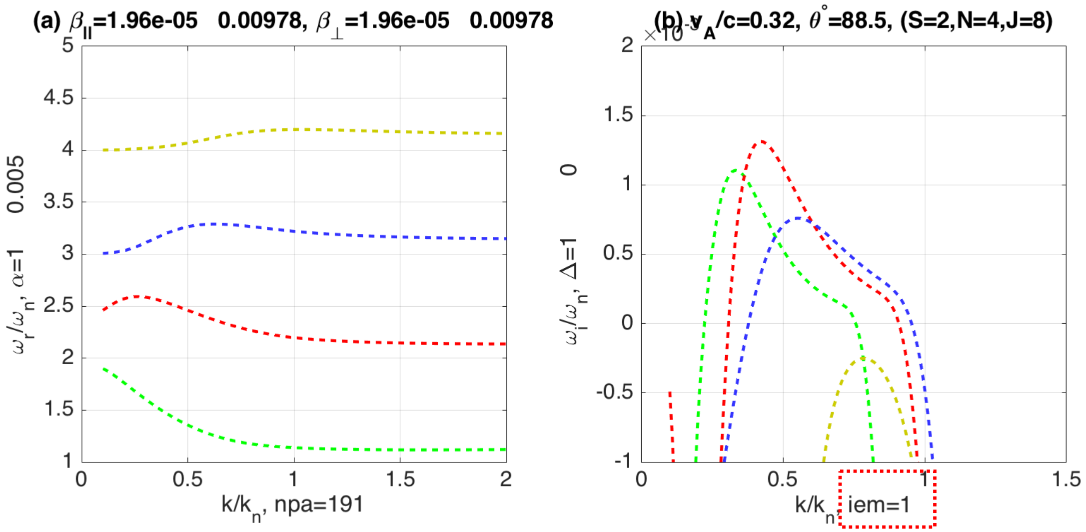
kn=1/lambdaDs(1);

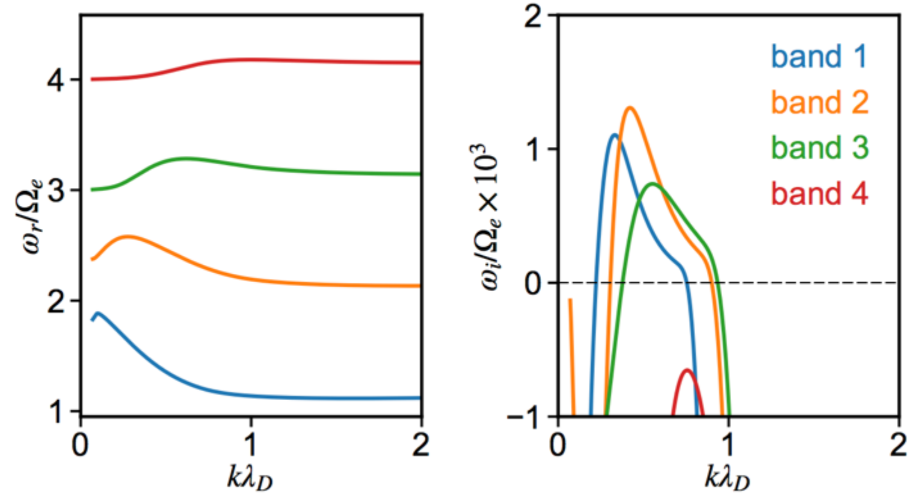
pa=0.1:0.01:2.0

iout=0;

Left: BO electrostatic run, iem=0.
Right: BO electromagnetic run, iem=1.
Down: HOTRAY electrostatic result from X. Tao et al, 2018 (submitted).







HOTRAY agrees with BO very well. And the agreement between iem=0 and iem=1 imply that the mode is indeed electrostatic mode.

Typical cases 6: ES1D beam

'm_or_u' can be either 0 or 1

bo.in

qs(e)	ms(mp)	ns(m^-3)	Tzs(eV)	Tps(eV)	alphas	Deltas	vds/c
-1	5.447e-4	0.9e6	1.	1.	1.0	1.0	0.0
-1	5.447e-4	0.1e6	1.	1.	1.0	1.0	9.8913e-3

B0= 143.5E-9; N=1; J=8; iem=0;

(ipa,ipb) = (1,1) scan k, fixed theta=0

pa=0.1:2:400

iout=0;

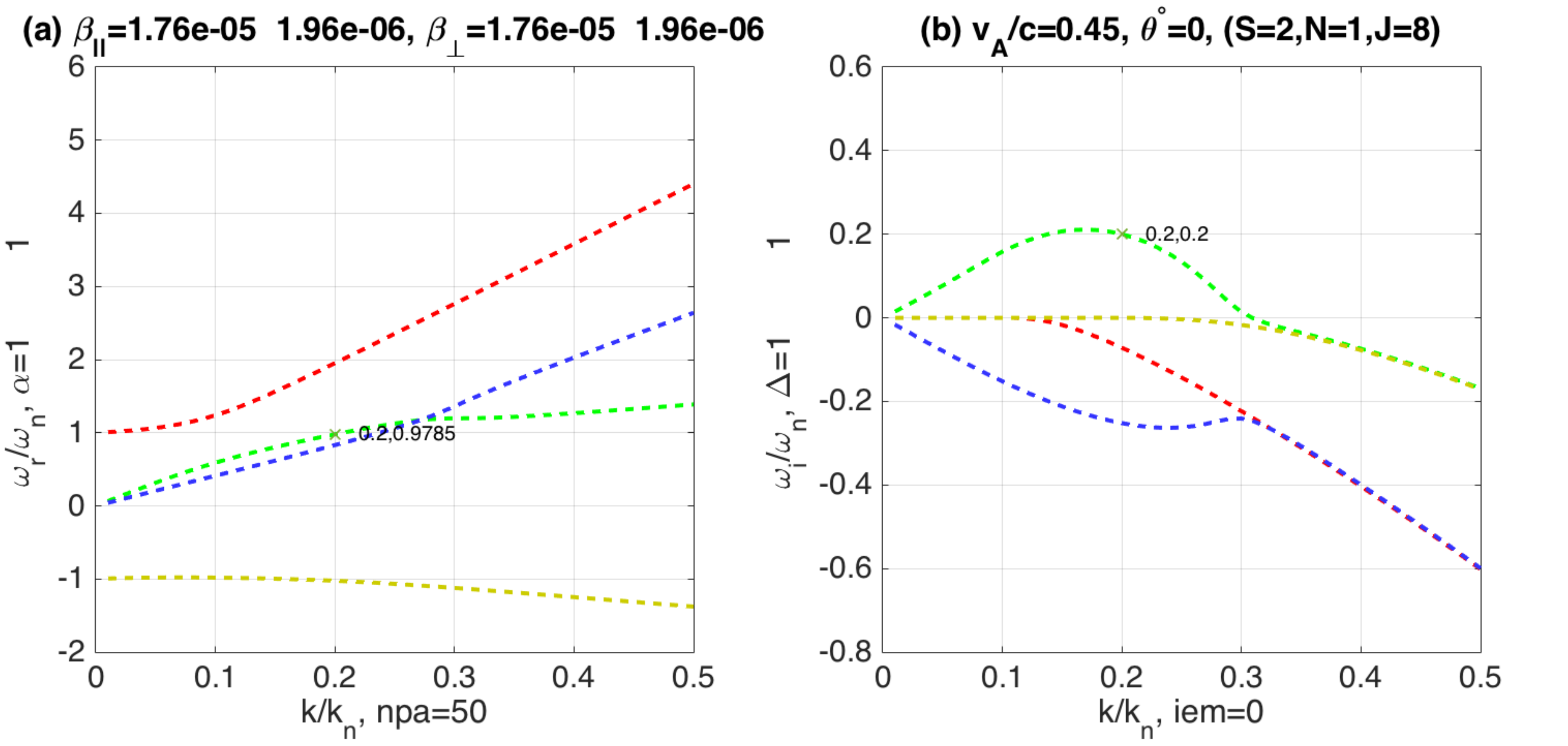
This is default beam test case in Xie2016PST PDRK paper for ES1D version, i.e., kperp=0, nb=0.1n0, ne=n0-nb and vds(2)/vtzs(2)=5.0.

Normalized to omega_pe and lambda_De
wn=abs(sqrt(sum(wps2)));
kn=abs(sqrt(1/sum(1./lambdaDs.^2)));

bo_main.m | bo_knwn.m | bo_setup.m | bo_wpdat.m | bo.in

qs(e)	ms(m_unit)	ns(m^-3)	Tzs(eV)	Tps(eV)	alphas	Deltas	vdsz/c	vdsx/c	vdsy/c	vdsr/c	nu_s	m_or_u(1/0)
-1	5.447e-4	0.9e6	1.	1.	1.0	1.0	0.0	0.0	0.0	0.0	0.0	1
-1	5.447e-4	0.1e6	1.	1.	1.0	1.0	9.8913e-3	0.0	0.0	0.0	0.0	1

Agree well with the original paper, i.e., k*lambda_De=0.2, the most unstable mode omega=0.9785+0.2000i



Typical cases 7: Drift across field [Muschietti17]

bo.in

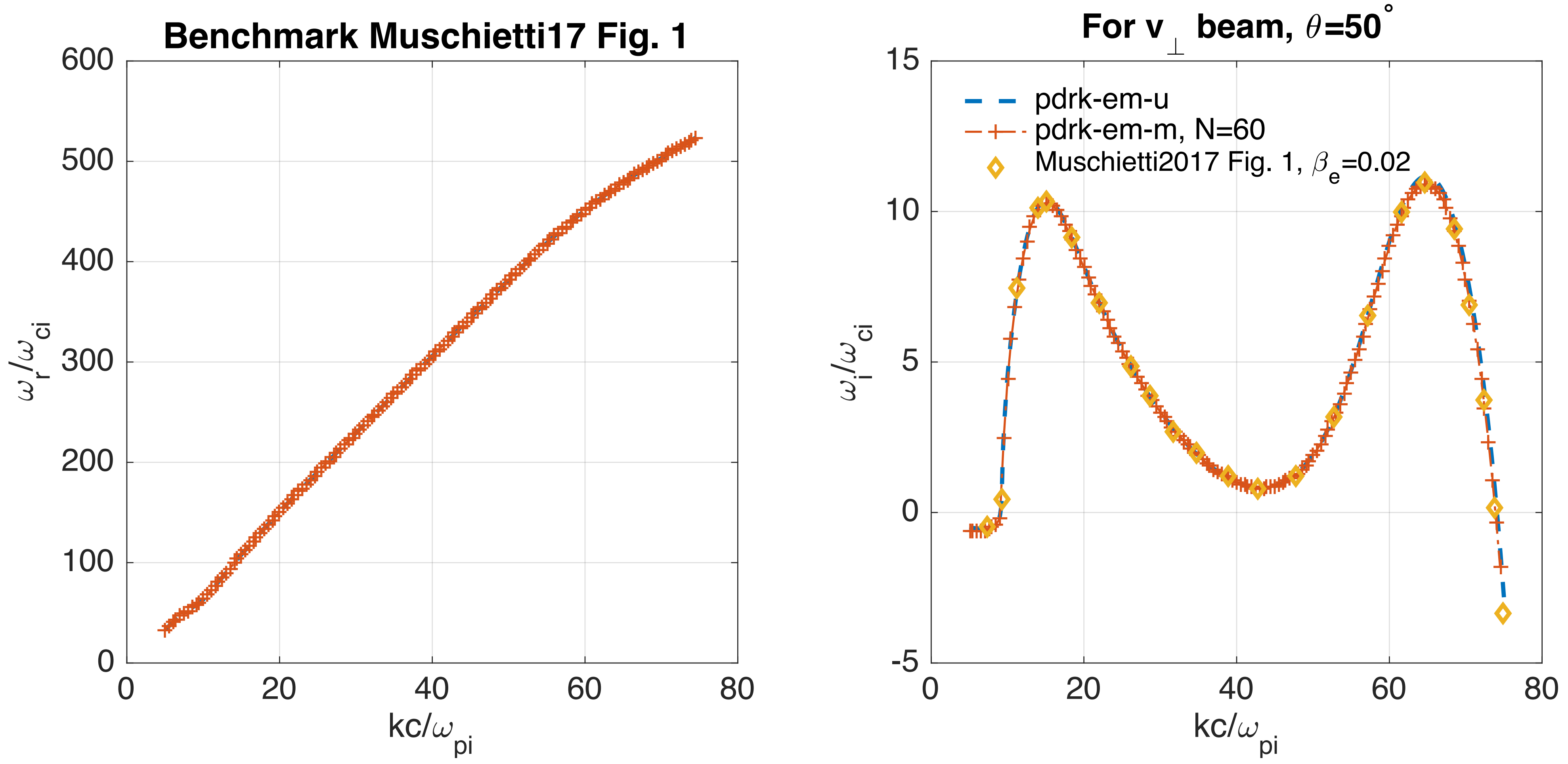
qs(e)	ms(m_unit)	ns(m^-3)	Tzs(eV)	Tps(eV)	vdsx/c	m_or_u(1/0)
1	1.0	0.8e0	0.111	0.111	-8.333e-3	0
1	1.0	0.2e0	0.111	0.111	3.333e-2	0
-1	1.1111e-3	1.0e0	0.010	0.010	0.0	1

```
B0= 1.0; N=2; J=8; iem=1;

(ipa,ipb) = (1,1) scan k, fixed theta=50

pa=10:1:75

iout=0;
```

```
c2=300^2
mu0=1.0;
epsilon0=1/(c2*mu0);
kB=1.0;
qe=1;
mp=1;
```

```
wpi=sqrt(wps2(1)+wps2(2));
kn=wpi/sqrt(c2);
```

In this example, ions drift at x-direction and are treated as unmagnetized. Unit is also not SI.
Set ions `m_or_u=1`, and use N=60 (sp=2 to speed up), the magnetized ion model agree with the unmagnetized ion model.



Typical cases 8: Ring Beam [Umeda12]

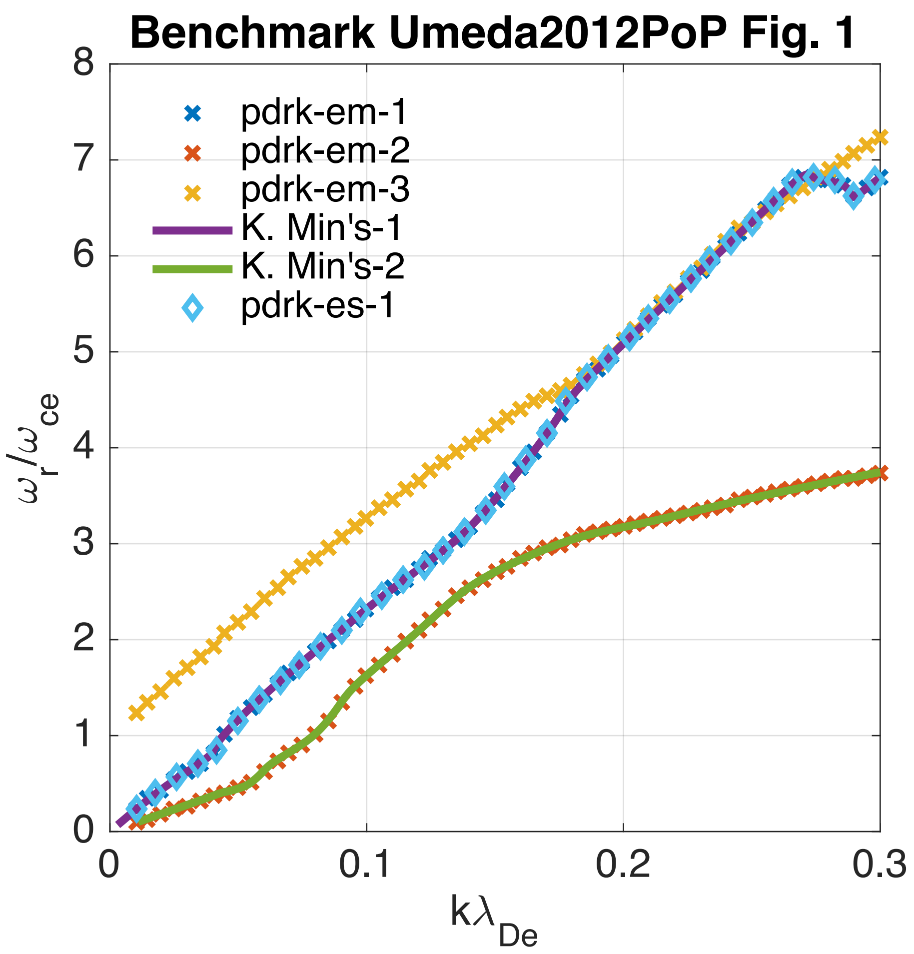
bo.in

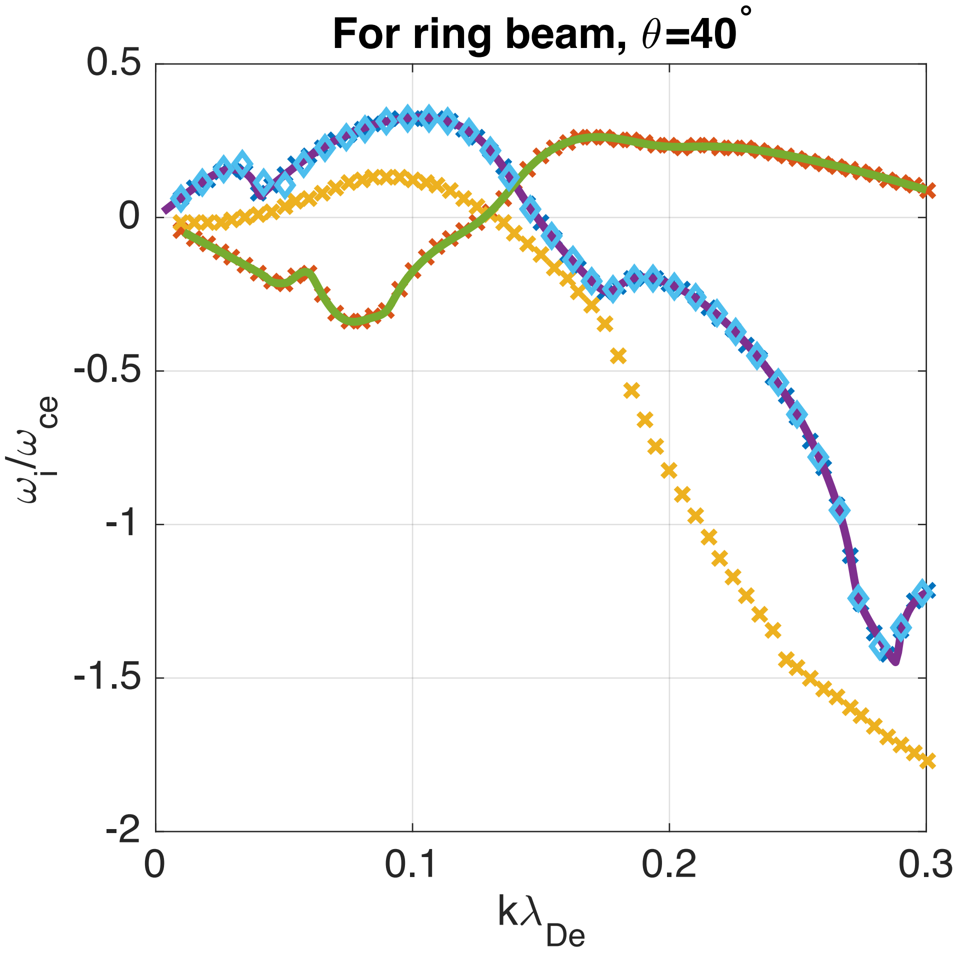
qs(e)	ms(m_unit)	ns(m^-3)	Tzs(eV)	Tps(eV)	alphas	Deltas	vdsz/c	vdsx/c	vdsy/c	vdsr/c	nu_s	m_or_u(1/0)
-1	5.447e-4	1.e5	0.51e2	0.51e2	1.0	1.0	0.1	0.0	0.0	0.05	0.0	1
-1	5.447e-4	9.e5	0.51e2	0.51e2	1.0	1.0	0.0	0.0	0.0	0.0	0.0	1

```
B0=96.24E-9; N=8; J=8; iem=1;

(ipa,ipb) = (1,1) scan k, fixed theta=40

pa=0.01:0.0025:0.3

iout=0; munit=mp;
```

Normalized:
lambdaD=sqrt(1./sum(1./lambdaDs.^2));
wn=abs(wcs(1));
kn=1/lambdaD;

Use Umeda2012 Fig.1 parameters, agree with K. Min’ s code [Min2015JGR]





Typical cases 9: Darwin Model

bo.in

qs(e)	ms(m_unit)	ns(m^-3)	Tzs(eV)	Tps(eV)	alphas	Deltas	vdsz/c	vdsx/c	vdsy/c	vdsr/c	nu_s	m_or_u(1/0)
-1	0.25	1.e6	9.38e6	9.38e6	1.0	1.0	0.0	0.0	0.0	0.0	0.0	1
1	1.0	1.e6	9.38e6	9.38e6	1.0	1.0	0.0	0.0	0.0	0.0	0.0	1

B0= 82.47E-7; N=1; J=8; iem=1 or 2;

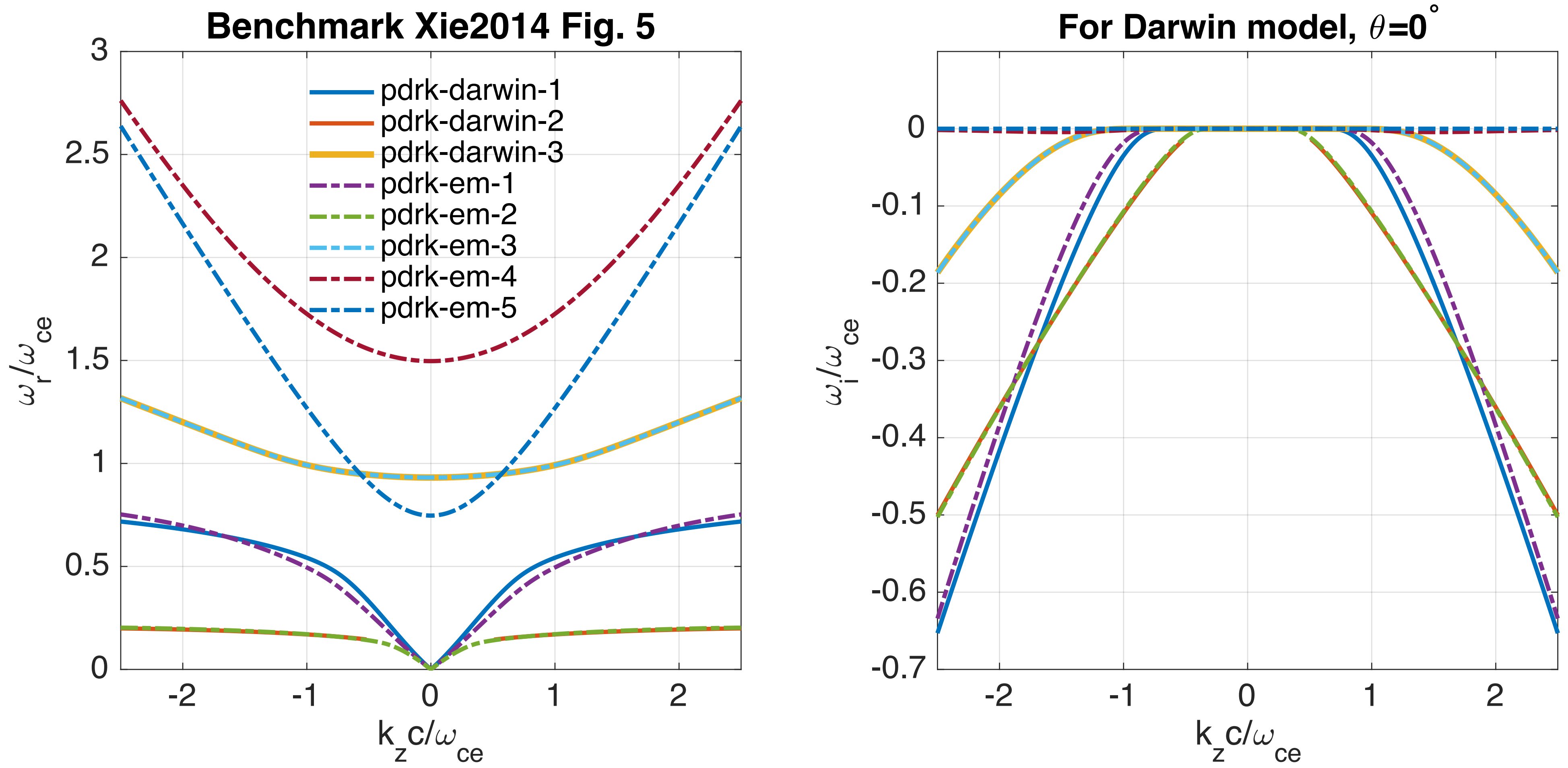
(ipa,ipb) = (3,3) scan kz, fixed kx=0

pa=0.01:0.01:2.5

iout=0; munit=mp;

Normalized:
wn=abs(wcs(1));
cwp=sqrt(c2)/abs(wcs(1));
kn=1/cwp;

Compare EM3D and Darwin model, with Xie2014 parameters, agree with expectation



Enjoy!

If you meet any problems or find BO does not agree some benchmarks, please not hesitate to email me (huashengxie@gmail.com), I will long term support this code. Thanks! The suggestions to improve this code are appreciated.

Download: http://code.ennresearch.com/bo/ (with latest version), https://github.com/hsxie/pdrk (v181027 and older), http://hsxie.me/codes/pdrk/ (v2016) or http://hsxie.me/codes/pdrf/(BO-F)

You are welcome to rewrite BO to other versions or other languages, also welcome the collaboration for further developments.

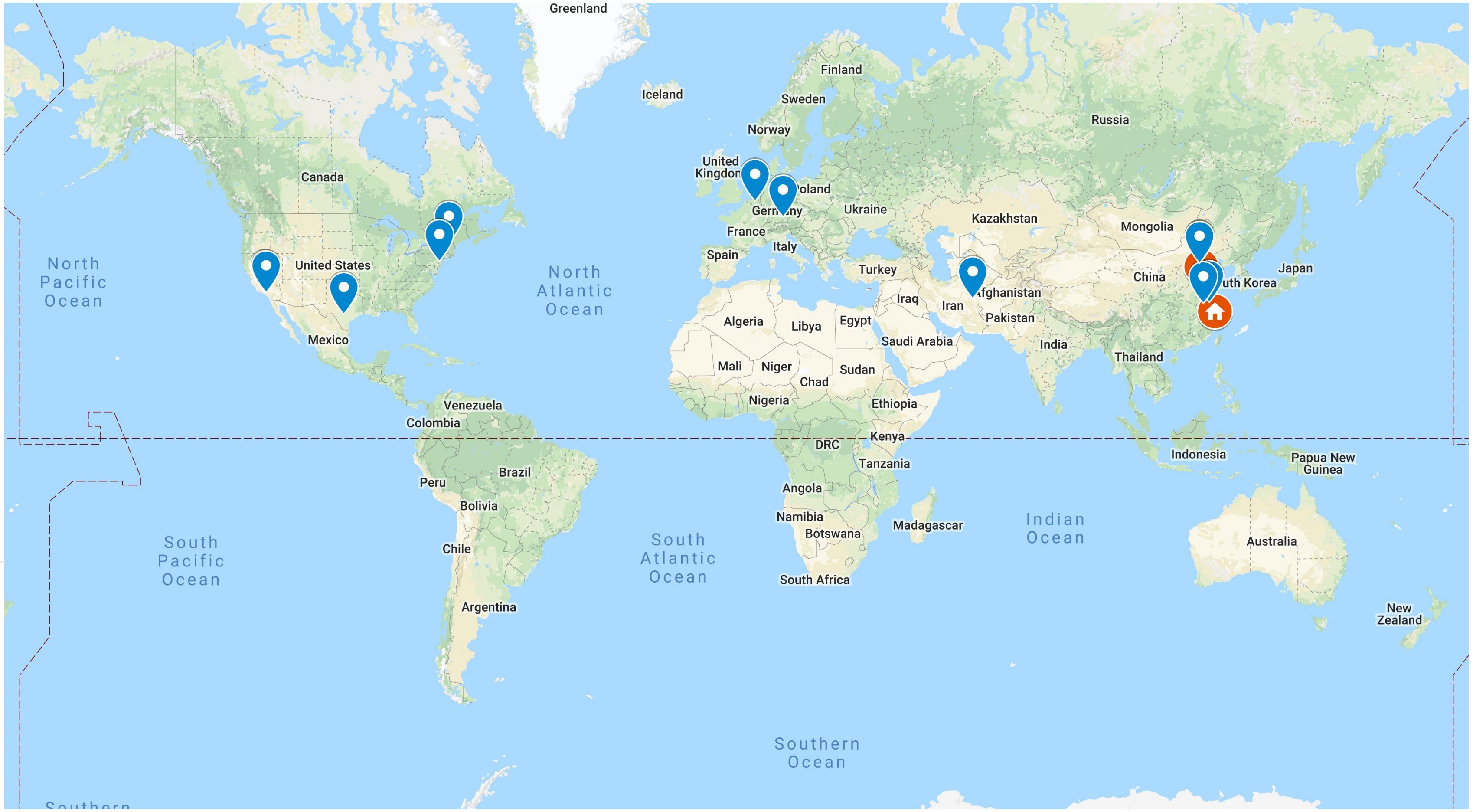
If you use BO, please cite at least one of:

- [Xie2016] Huasheng Xie and Yong Xiao, PDRK: A General Kinetic Dispersion Relation Solver for Magnetized Plasma, Plasma Science and Technology, 18, 2, 97 (2016). DOI: 10.1088/1009-0630/18/2/01. Update/Bugs fixed at http://hsxie.me/codes/pdrk/ or https://github.com/hsxie/pdrk.
- [Xie2019] Huasheng Xie, A Unified Numerically Solvable Framework for Complicated Kinetic Plasma Dispersion Relations, arXiv, 2019, https://arxiv.org/abs/1901.06902.
- [Xie2014] Huasheng Xie, PDRF: A general dispersion relation solver for magnetized multi-fluid plasma, Computer Physics Communications, 185, 670 (2014). DOI: 10.1016/j.cpc.2013.10.012. Update/Bugs fixed at http://hsxie.me/codes/pdrf/.

BO Contributors/Users Map (2018)

BO Contributors/Users

- ENN
- IFTS-ZJU
- PMO
- NSSC
- USTC
- Dartmouth
- Princeton
- SWRI
- PKU
- UCLA
- NASA
- IPP
- KULeuven
- Birjand



BO (aka, PDRK/PDRF) is a powerful fluid and kinetic plasma wave and instability analysis tool.

Welcome to join us!

http://code.ennresearch.com/bo/