Landau Damping Simulation Models

Hua-sheng Xie (谢华生, huashengxie@gmail.com)
Institute for Fusion Theory and Simulation, Department of Physics,
Zhejiang University, Hangzhou, 310027, People's Republic of China

Oct. 4, 2013

Abstract

Several models for Landau damping simulations are provided: the simplest linear model, particle-in-cell (PIC) simulation and Vlasov continuity simulation as well as the discussions of the numerical aspects of the dispersion relation.

1 Introduction

Landau damping is one of the most interesting phenomena found in plasma physics. However, the mathematical derivation and physical understanding of it are usually headache, especially for beginners.

Here, I will tell how to use simple and short codes to study this phenomena. A shortest code to produce Landau damping accurately can be even **less than 10** lines!

2 Linear simulation model

We focus on the electrostatic 1D (ES1D) Vlasov-Poisson system (ion immobile).

The simplest method to study Landau damping is solving the following equations

$$\partial_t \delta f = -ikv\delta f + \delta E \partial_v f_0, \tag{1a}$$

$$ik\delta E = -\int \delta f dv, \tag{1b}$$

An example code can be

```
k=0.4; dt=0.01; nt=8000; dv=0.1; vv=-8:dv:8;
df0dv=-vv.*exp(-vv.^2./2)/sqrt(2*pi);
df=0.*vv+0.1.*exp(-(vv-2.0).^2); tt=linspace(0,nt*dt,nt+1);
dE=zeros(1,nt+1); dE(1)=0.01;
for it=1:nt
    df=df+dt.*(-1i*k.*vv.*df+dE(it).*df0dv);
```

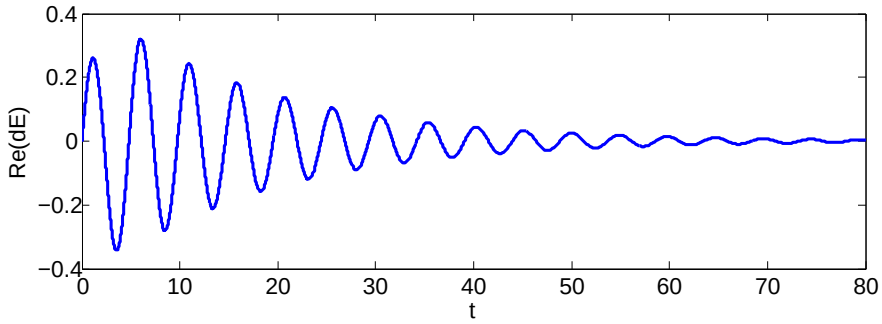

Figure 1: Linear simulation of Landau damping.

```
    dE(it+1)=(1i/k)*sum(df)*dv;
end
plot(tt,real(dE)); xlabel('t'); ylabel('Re(dE)');
```

The simulation result is shown in Fig. 1.

Exercise 1: Solving the following fluid equations

$$\partial_t \delta n = -ik\delta u, \tag{2a}$$
$$\partial_t \delta u = -\delta E - 3ik\delta n, \tag{2b}$$
$$ik\delta E = -\delta n, \tag{2c}$$

using the above method to reproduce the Langmuir wave

$$\omega^2 = 1 + 3k^2. \tag{3}$$

3 Dispersion relation

We have seen the simplest model for Landau damping simulation in Sec. 2. How to understand the simulation results? Comparing with the analytical theory!

Dispersion relation

$$D(k,\omega) = 1 - \frac{1}{k^2}\int_C \frac{\partial f_0/\partial v}{v-\omega/k}dv = 0, \tag{4}$$

where C is the Landau integral contour. For Maxwellian distribution $f_0 = \frac{1}{\sqrt{2\pi}}e^{-\frac{v^2}{2}}$, we will meet the well-known plasma dispersion function (PDF)

$$Z_M(\zeta) = \frac{1}{\sqrt{\pi}}\int_C \frac{e^{-z^2}}{z-\zeta}dz. \tag{5}$$

Hence, (4) is rewritten to

$$D(k,\omega) = 1 - \frac{1}{k^2}\frac{1}{2}Z_M'(\zeta) = 0. \tag{6}$$

Solving it, we obtain Table 1.

Table 1: Numerical solutions of the Landau damping dispersion relation

$k\lambda_D$	ω_r/ω_{pe}	γ_r/ω_{pe}
0.1	1.0152	-4.75613E-15
0.2	1.06398	-5.51074E-05
0.3	1.15985	-0.0126204
0.4	1.28506	-0.066128
0.5	1.41566	-0.153359
0.6	1.54571	-0.26411
0.7	1.67387	-0.392401
0.8	1.7999	-0.534552
0.9	1.92387	-0.688109
1.0	2.0459	-0.85133
1.5	2.63233	-1.77571
2	3.18914	-2.8272

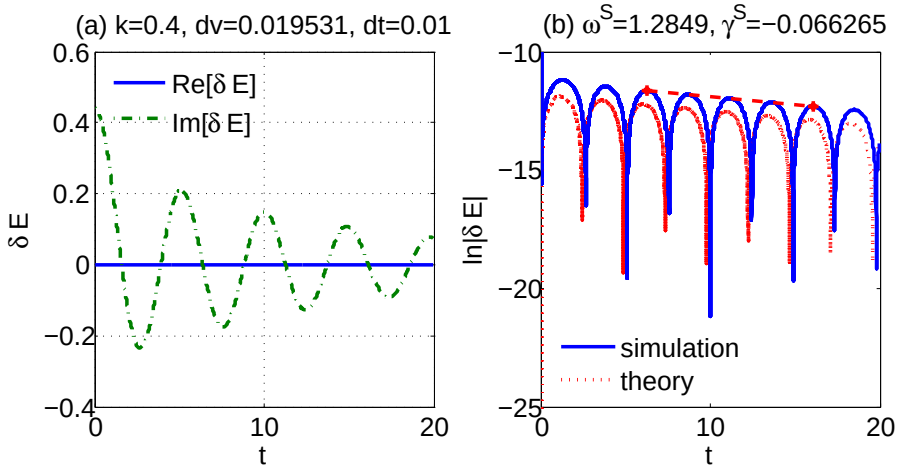

Figure 2: Linear simulation of Landau damping and compared with theory.

Adding some diagnosis lines to the code in Sec. 2, the comparison result is shown in Fig. 2. Perfect agreement is obtained: $\omega^{\text{theory}} = 1.28506 - 0.066128i$ and $\omega^{\text{simulation}} = 1.2849 - 0.06627i$

Several related topics have been omitted here, e.g., Case-van Kampen ballistic modes (eigenmode problem), non-physical recurrence effect (the Poincaré recurrence) at $T_R = 2\pi/(k\Delta v)$[1], solving the dispersion relation with general equilibrium distribution functions (not limited to Maxwellian), advanced schemes (e.g., 4th R-K), and so on. One can refer Ref.[2] and references in for more details.

4 Particle-in-cell simulation

Normalized equations (Lagrangian approach)

$$\begin{aligned} d_t x_i &= v_i, && \text{(7a)} \\ d_t v_i &= -E(x_i), && \text{(7b)} \\ d_x E(x_j) &= 1 - n(x_j), && \text{(7c)} \end{aligned}$$

where $i = 1, 2, \cdots, N_p$ is particle (marker) label and $j = 0, 1, \cdots, N_g - 1$ is grid label. The particles i can be every where, whereas the field is discrete in grids $x_j = j\Delta x$, where $\Delta x = L/N_g$.

We solve the above equations in the domain $0 < x < L$ [note: $\int n(x)dx = L$], and consider periodic boundary conditions $n(0) = n(L)$ and $\langle E(x) \rangle_x = 0$. Any particle crosses the right boundary of the solution domain must reappear at the left boundary with the same velocity, and vice versa.

The initial probability distribution function [e.g., $f_0 = \frac{1}{\sqrt{2\pi}} e^{-\frac{v^2}{2}}$] is generated by N_p random numbers.

Two key steps for PIC are: 1. Field $E(x_j)$ on grids to $E(x_i)$ on particle position; 2. Particle density $n(x_i)$ to grids $n(x_j)$. Suppose that the i-th electron lies between the j-th and $(j+1)$-th grid-points, i.e., $x_j < x_i \le x_{j+1}$. Usually, the below interpolation method is used

$$\begin{aligned} n_j &= n_j + \frac{x_{j+1} - x_i}{x_{j+1} - x_j} \frac{1}{\Delta x}, && \text{(8a)} \\ n_{j+1} &= n_{j+1} + \frac{x_i - x_j}{x_{j+1} - x_j} \frac{1}{\Delta x}. && \text{(8b)} \end{aligned}$$

The above procedure is repeated from the first particle to the last particle. Similar procedure are used to mapping $E(x_j)$ to $E(x_i)$.

Code pices1d.m

```
close all;clear all;clc;
data = [0.1    1.0152    -4.75613E-15
0.2    1.06398    -5.51074E-05
0.3    1.15985    -0.0126204
0.4    1.28506    -0.066128
0.5    1.41566    -0.153359
0.6    1.54571    -0.26411
0.7    1.67387    -0.392401
0.8    1.7999    -0.534552
0.9    1.92387    -0.688109
1.0    2.0459    -0.85133
1.5    2.63233    -1.77571
2    3.18914    -2.8272];
id=7;
k=data(id,1);
wr=data(id,2); wi=data(id,3);

% parameters
```

```
L=2*pi/k; dt=.01; nt=1000; ntout=100; ng=32; np=50000;
vb=1.0; xp1=5.0e-1; vp1=0.0;
vt=1; % note: the normalization sqrt(2) will be found in randn()
wp=1; qm=-1;
q=wp^2/(qm*np/L); rho_back=-q*np/L; dx=L/ng;

% initial loading for the 2 Stream instability
xp=linspace(0,L,np)';
% vp=vt*randn(np,1)+(1-2*mod([1:np]',2)).*vb;
vp=vt*randn(np,1); % randn is {exp[-(x-mu)^2/(2*sgm^2)]}/[sgm*sqrt(2*pi)]

% Perturbation
vp=vp+vp1*cos(k*xp);
xp=xp+xp1*cos(k*xp);
p=1:np;p=[p p];

% Main computational cycle
for it=1:nt
    % apply periodic bc on the particle positions
    xp=xp./L+10.0; xp=L.*(xp-floor(xp));

    % diagnosing
    if(mod(it,ntout)==0)
        plot(xp,vp,'b.','Markersize',2);
        axis([0,L,-3*(abs(vt)+abs(vb)),3*(abs(vt)+abs(vb))]);
        title(['Phase space plotting, vp-x, t=',num2str(it*dt)]);
        xlabel('xp');ylabel('vp'); pause(0.2);
%         print(gcf, '-dpng', ['vp-x,t=',num2str(it*dt),'.png']);
    end

    % update xp
    xp=xp+vp*dt;

    % projection p->g
    g1=floor(xp/dx-.5)+1;g=[g1;g1+1];
    fraz1=1-abs(xp/dx-g1+.5);
    fraz=[fraz1;1-fraz1];

    % apply bc on the projection
    out=(g<1);g(out)=g(out)+ng;
    out=(g>ng);g(out)=g(out)-ng;
    mat=sparse(p,g,fraz,np,ng);
    rho=full((q/dx)*sum(mat))'+rho_back;

    % computing fields, dE/dx
    Eg=zeros(ng,1);
    for j=1:ng-1
        Eg(j+1)=Eg(j)+(rho(j)+rho(j+1))*dx/2;
```

```
    end
    Eg(1)=Eg(ng)+rho(ng)*dx;
    Eg=Eg-mean(Eg);

    % projection q->p and update of vp
    vp=vp+mat*qm*Eg*dt;

    EEk(it)=0.5*abs(q)*sum(vp.^2); % kinetic energy
    EEf(it)=0.5*sum(Eg.^2)*dx; % potential energy
    t(it)=it*dt;
end

%%
h = figure('Unit','Normalized','position',...
            [0.02 0.4 0.6 0.3],'DefaultAxesFontSize',15);
subplot(121);plot(t,EEk,t,EEf,t,EEk+EEf,'r:','LineWidth',2);
title(['(a) k=',num2str(k),', \omega_{theory}=',...
    num2str(wr+1i*wi)],'fontsize',15);
xlabel('t'); ylabel('Energy');legend('E_k','E_e','E_{tot}',4);
legend('boxoff');

subplot(122);

% Find the corresponding indexes of the extreme max values
lndE=log(sqrt(real((EEf(1:nt))))); % EEf=E^2=[exp(gam*t)]^2=exp(2*gam*t)
it0=floor(nt*1/20); it1=floor(nt*17/20);
yy=lndE(it0:it1);
extrMaxIndex = find(diff(sign(diff(yy)))==-2)+1;
t1=t(it0+extrMaxIndex(1));t2=t(it0+extrMaxIndex(end));
y1=yy(extrMaxIndex(1));y2=yy(extrMaxIndex(end));
plot(t,lndE,[t1,t2],[y1,y2],'r*--','LineWidth',2);
omega=pi/((t2-t1)/(length(extrMaxIndex)-1));
gammas=(real(y2)-real(y1))/(t2-t1);
title(['(b) \omega^S=',num2str(omega),', \gamma^S=',num2str(gammas)]);
axis tight;
```

Results are shown in Fig. 3. The energy conservation is very well. Real frequency and damping rate agree roughly with theory. A main drawback of PIC is the noise. Usually, very large N_p is required to reduce the influence of the noise.

5 Vlasov continuity simulation

Euler approach.

Vlasov equation

$$\partial_t f(x,v,t) = -v\partial_x f - \partial_x \phi \partial_v f, \tag{9}$$

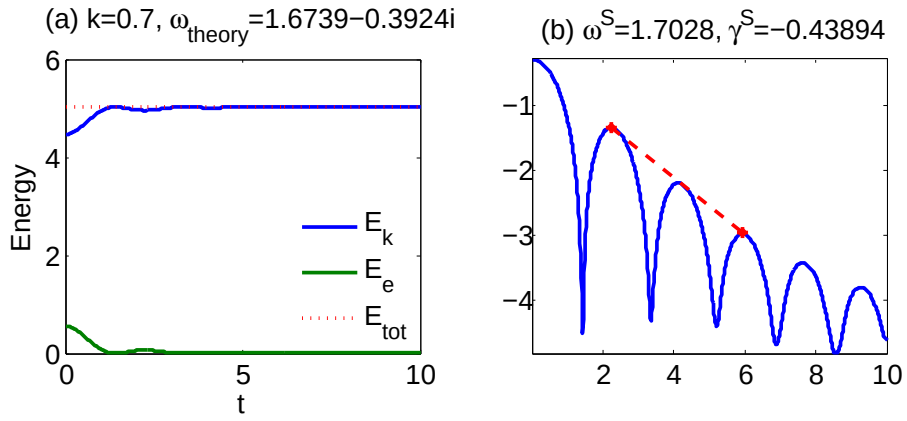

Figure 3: PIC simulation of Landau damping.

Discrete

$$\frac{f_{i,j}^{n+1}-f_{i,j}^{n}}{\Delta t}=-v_j\frac{f_{i+1,j}^{n}-f_{i-1,j}^{n}}{2\Delta x}-\frac{\phi_{i+1,j}^{n}-\phi_{i-1,j}^{n}}{2\Delta x}\frac{f_{i,j+1}^{n}-f_{i,j-1}^{n}}{2\Delta v}, \tag{10}$$

gives

$$f_{i,j}^{n+1}=f_{i,j}^{n}-v_j\frac{f_{i+1,j}^{n}-f_{i-1,j}^{n}}{2}\frac{\Delta t}{\Delta x}-\frac{\phi_{i+1,j}^{n}-\phi_{i-1,j}^{n}}{2\Delta x}\frac{f_{i,j+1}^{n}-f_{i,j-1}^{n}}{2}\frac{\Delta t}{\Delta v}. \tag{11}$$

Poisson equation

$$\partial_x^2\phi=\int f dv-1. \tag{12}$$

Discrete

$$\frac{\phi_{i+1}-2\phi_i+\phi_{i-1}}{\Delta x^2}=\sum_j f_{i,j}\Delta v-1\equiv\rho_i, \tag{13}$$

i.e.,

$$\begin{bmatrix}-2&1&0&\cdot&\cdot&0&1\\1&-2&1&\cdot&\cdot&\cdot&0\\0&1&-2&1&\cdot&\cdot&\cdot\\\cdot&\cdot&\cdot&\cdot&\cdot&\cdot&\cdot\\1&0&\cdot&\cdot&\cdot&1&2\end{bmatrix}\begin{bmatrix}\phi_1\\\phi_2\\\cdot\\\cdot\\\phi_N\end{bmatrix}=\begin{bmatrix}\rho_1\\\rho_2\\\cdot\\\cdot\\\rho_N\end{bmatrix}\Delta x^2, \tag{14}$$

where we have used the periodic boundary condition $\phi(0)=\phi(L)$, i.e., $\phi_1=\phi_{N+1}$ and $\phi_0=\phi_N$.

Code fkvl1d.m

```
% Hua-sheng XIE, huashengxie@gmail.com, IFTS-ZJU, 2012-03-03 20:34
% fkvl1d.m, 1D full-kinetic Vlasov code for unmagnetized plasma
% ion immobile, periodical boundary condition, phi(1)=phi(Nx+1)=0
close all;clear;clc;

% Landau damping data
data = [0.1    1.0152    -4.75613E-15
0.2    1.06398    -5.51074E-05
0.3    1.15985    -0.0126204
0.4    1.28506    -0.066128
0.5    1.41566    -0.153359
```

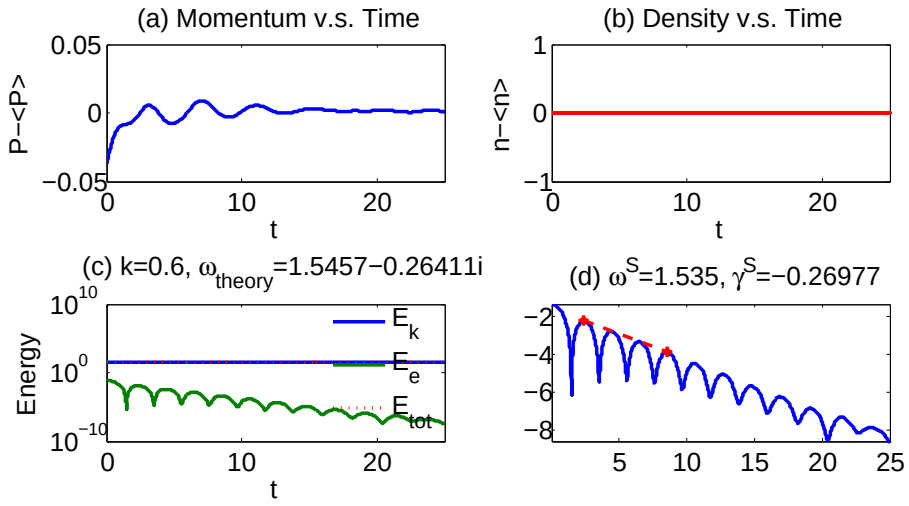

Figure 4: Vlasov continuity simulation, history plotting.

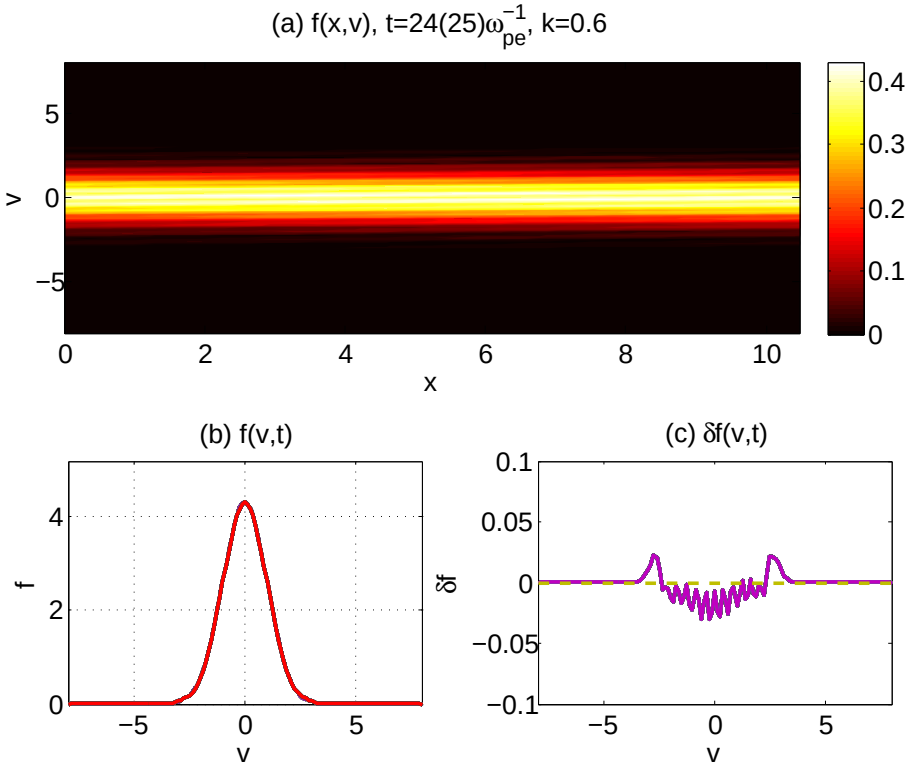

Figure 5: Vlasov continuity simulation, distribution function.

```
0.6    1.54571    -0.26411
0.7    1.67387    -0.392401
0.8    1.7999     -0.534552
0.9    1.92387    -0.688109
1.0    2.0459     -0.85133
1.5    2.63233    -1.77571
2    3.18914    -2.8272];
id=6;

k=data(id,1); wr=data(id,2); gamma=data(id,3);

nx=1*32; nv=2*64; dt=0.01; nt=5000/2; % parameters
L=2*pi/k; vmax=8.0; vmin=-vmax;
dx=L/nx; dv=(vmax-vmin)/nv;
fperp=0.1;

na=1.0; nb=1.0-na; vta=1.0; vtb=1.0; vda=-0.0; vdb=2.0;
% na=0.5; nb=1.0-na; vta=0.5; vtb=0.5; vda=-2.0; vdb=2.0;
% na=0.9; nb=1.0-na; vta=1.0; vtb=0.5; vda=-0.0; vdb=4.0;

vta=sqrt(2);

if(na==1.0)  % for plot Landau damping theoretical results
    tt=20;
else
    tt=0;
end

X=0:dx:L; V=vmin:dv:vmax; [v,x]=meshgrid(V,X); % x and v grids
f=na*exp(-(v-vda).^2/(vta^2))/(sqrt(pi)*vta)+...
    nb*exp(-(v-vdb).^2/(vtb^2))/(sqrt(pi)*vtb); % initial distribution
f=(1.0+fperp.*cos(k.*x)).*f; % initial perturbation

f(:,1)=0; f(:,nv+1)=0; % boundary in V space F must be null
f(nx+1,:)=f(2,:); % boundary in X space are periodic
f(1,:)=f(nx,:);

% f=f./(sum(sum(f))*dx*dv)/L; % re-normalized to avoid discete errors
fv0=sum(f,1)*dx; maxfv0=max(fv0);

rho_back=1.0; % /L

dtodx=dt/dx; dtodv=dt/dv; dx2=dx*dx;

un=ones(nx,1);
poisson=spdiags([un -2*un un],[-1 0 1],nx,nx);
poisson(1,nx)=1;poisson(nx,1)=1; % phi(nx+1)=phi(1), phi(0)=phi(nx)

```

```
figure('position',[100 100 800 600]);
set(gcf,'DefaultAxesFontSize',15);
set(gca,'nextplot','replacechildren');
ifig=1;

for it=1:nt

    rho=sum(f,2)*dv-rho_back; % computing density

    phi=poisson\(rho(1:nx)*dx2); % computing fields
    phi=[phi;phi(1)]; % ./rho_back
    Ex=([phi(nx+1); phi(1:nx)]-[phi(2:nx+1);phi(1)])/(2*dx);
    Ex(nx+1)=Ex(1);

    for j=2:nv % computing PDF
        f(1,j)=f(1,j)-V(j)*(f(2,j)-f(nx,j))/2*dtodx+...
                Ex(1)*(f(1,j+1)-f(1,j-1))/2*dtodv;
        for i=2:nx
            f(i,j)=f(i,j)-V(j)*(f(i+1,j)-f(i-1,j))/2*dtodx+...
                Ex(i)*(f(i,j+1)-f(i,j-1))/2*dtodv;
        end
    end

    f(:,1)=0; f(:,nv+1)=0; % boundary in V space F must be null
    f(nx+1,:)=f(1,:); % boundary in X space are periodic

    den=sum(f,2)*dv; % density
    P=sum(v.*f,2)*dv; % momentum
    Ek=sum((v.^2).*f,2)*dv; % kinetic energy
    t(it)=it*dt;
    Den=sum(den); % density history
    PP(it)=sum(P); % momentum history
    EEk(it)=sum(Ek); % kinetic energy history
    EEf(it)=0.5*sum(Ex.*Ex)*dx; % field energy history

    if(mod(it,floor(nt/200)*10)==0) % diagnose
        pause(0.001);

        subplot(2,2,1:2);
        colormap(hot);
        [C,h] =contourf(x,v,f,50); colorbar;
        set(h,'Color','none');
        title(['(a) f(x,v), t=',num2str(it*dt),'(',num2str(nt*dt),...
            ')\omega_{pe}^{-1}, k=',num2str(k)],'fontsize',15);
        xlabel('x');ylabel('v');

        subplot(2,2,3);
        fv=sum(f,1)*dx;
```

```
108             plot(v,fv0,'--',v,fv,'-','LineWidth',2); grid on;
109             title('(b) f(v,t)');xlabel('v');ylabel('f');
110             ylim([0,1.2*maxfv0]);xlim([vmin,vmax]);
111
112             subplot(2,2,4);
113             plot(v,fv-fv0,'-',[vmin,vmax],[0,0],'--','LineWidth',2);
114             title('(c) \delta{}f(v,t)');xlabel('v');ylabel('\delta{}f');
115             xlim([vmin,vmax]);
116             ylim([-1*fperp*maxfv0,1*fperp*maxfv0]);
117             ylim([-1*fperp,1*fperp]);
118
119             Fig(ifig)=getframe(gcf,[0,0,800,600]);
120             ifig=ifig+1;
121         end
122 end
123 print(gcf,'-dpng','plot_fxv.png');
124 movie(Fig);
125 writegif('movie_fxv.gif',Fig,0.1);
126 % close all;
127
128 %%
129 h = figure;
130 set(gcf,'DefaultAxesFontSize',15);
131 subplot(221);plot(t,PP-mean(PP),'LineWidth',2); xlim([0,max(t)]);
132 title('(a) Momentum v.s. Time','fontsize',15);xlabel('t');ylabel('P-<P>');
133 subplot(222);plot(t,Den-mean(Den),'r','LineWidth',2); xlim([0,max(t)]);
134 title('(b) Density v.s. Time','fontsize',15);xlabel('t');ylabel('n-<n>');
135 subplot(223);
136 semilogy(t,EEk,t,EEf,t,EEk+EEf,':','LineWidth',2);
137 title(['(c) k=',num2str(k),', \omega_{theory}=',...
138     num2str(wr+1i*gamma)],'fontsize',15);
139 xlabel('t');ylabel('Energy');legend('E_k','E_e','E_{tot}',4);
140 legend('boxoff'); xlim([0,max(t)]);
141
142 subplot(224);
143 % Find the corresponding indexes of the extreme max values
144 lndE=log(sqrt(real((EEf(1:nt))))); % EEf=E^2=[exp(gam*t)]^2=exp(2*gam*t)
145 it0=floor(nt*1/20); it1=floor(nt*7/20);
146 yy=lndE(it0:it1);
147 extrMaxIndex = find(diff(sign(diff(yy)))==-2)+1;
148 t1=t(it0+extrMaxIndex(1));t2=t(it0+extrMaxIndex(end));
149 y1=yy(extrMaxIndex(1));y2=yy(extrMaxIndex(end));
150 plot(t,lndE,[t1,t2],[y1,y2],'r*--','LineWidth',2);
151 omega=pi/((t2-t1)/(length(extrMaxIndex)-1));
152 gammas=(real(y2)-real(y1))/(t2-t1);
153 title(['(d) \omega^S=',num2str(omega),', \gamma^S=',num2str(gammas)]);
154 axis tight;
155
```

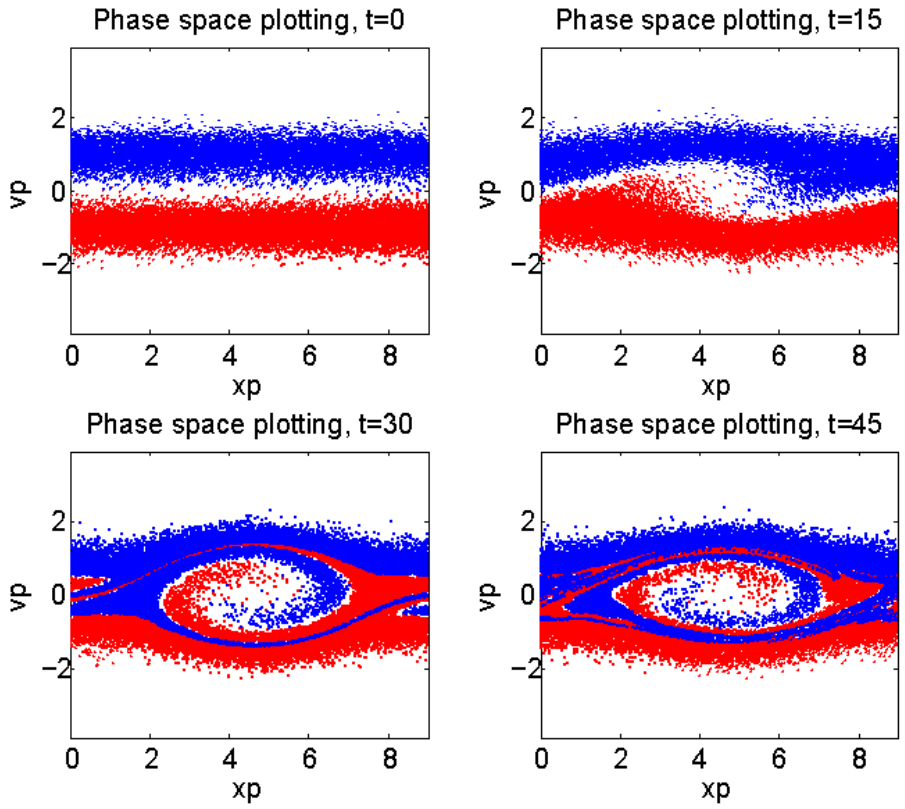

Figure 6: PIC simulation of the two-stream instability, phase space plotting.

```
156 print(h,'-dpng','plot_diag.png');
157 % close all;
```

writegif.m can be found using Google.

The results are shown in Fig. 4 and Fig. 5. Real frequency and damping rate are consistent with theory.

6 Nonlinear simulations

The PIC and Vlasov codes provided in the above sections can be easily modified to study the linear and nonlinear physics of the beam-plasma or two-stream instabilities.

A PIC simulation of two-stream instability is shown in Fig. 6 and Fig. 7. The linear growth and nonlinear saturation are very clear.

Exercise 2: Solving the kinetic or fluid dispersion relations for beam-plasma or two-stream plasma and comparing the results with linear and nonlinear simulations using the above models.

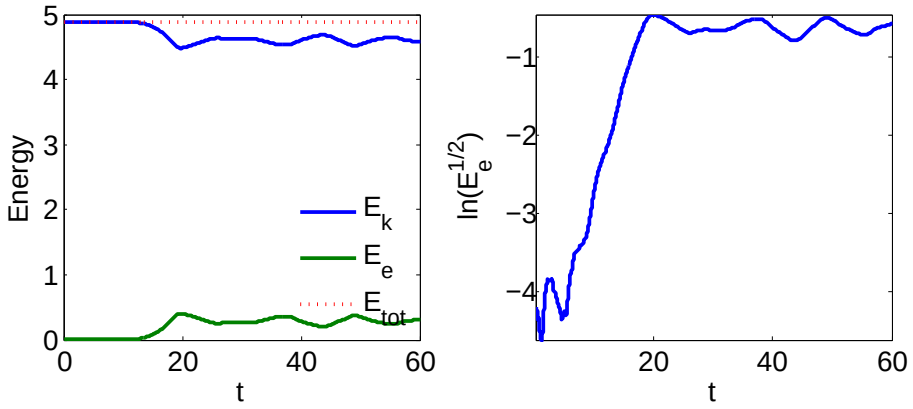

Figure 7: PIC simulation of the two-stream instability, history plotting.

References

[1] C. Z. Cheng and G. Knorr, The integration of the vlasov equation in configuration space, Journal of Computational Physics, 22, 330 - 351, 1976.

[2] H. S. Xie, Generalized Plasma Dispersion Function: One-Solve-All Treatment, Visualizations, and Application to Landau Damping, Phys. Plasmas, 20, 092125, 2013. Also, `http://arxiv.org/abs/1305.6476`. With codes.