```
// Linear small amplitude wave theory - deep water
// Canal fermé des 2 côtés - agitation
clf()
option=2 //1 pour propagation vers la droire, -1 pour propagation vers la gauche
// 2 pour agitation

LONG=10; // longueur du canal
a=0.1;// amplitide de l'onde

b=get("current_axes");
b.data_bounds=[0,-1;10,0.6];
f=get("current_figure") //get the handle of the current figure :
//if none exists, create a figure and return the corresponding handle
f.figure_position
f.figure_size=[1500,400]
c=b.children
// fixer la couleur du champ de vecteur à "blanc"
f=gcf()
f.color_map=wintercolormap(32)
titre='Agitation d''une houle de faible amplitude - théorie linéaire : '
title(titre,'position',[0.5 0.5],'fontsize',3)

function r=dispersion(k, sigma, g, h)
    r=sigma^2/(g*tanh(k*h))-k
endfunction
g=9.81 // gravite
T=1; // période de l'onde
h=1; //profondeur d'eau
sigma=2*%pi/T
// résolution de l'équation de dispersion
k1=(sigma^2)/2*g;
k2=sqrt(sigma^2/(g*h))
delta=dispersion(k1,sigma,g,h)
if(delta<0) then kmin=k1;kmax=k2;
else kmin=k2;kmax=k1;
end
while abs(delta)>0.0001
    k=(kmin+kmax)/2;
    delta=dispersion(k,sigma,g,h)
    if(delta<0) then kmin=k;
    elseif(delta>0) then kmax=k;
    end,
end

if option<>2 then k=option*k;  end
x=0:0.1:LONG //discrétisation suivant x
sigma=2*%pi/T // fréquence
agksursigma=a*g*k/sigma
//couleurs des aires des courbes
id1=color('white')
id2=color(0,191,255)
xvect=[1 2 3 4 5 6 7 8 9 ];yvect=[-0.9:0.2:-0.2]
fx=zeros(9,4);fx1=fx;fx2=fx;;fy=fx;fy1=fx;fy2=fx
//=========================================================================
function [fx, fy]=vitesse(k, d, agksursigma, t, xvect, yvect, sigma)
    fvarx=agksursigma*(cos(k*xvect-sigma*t))
    fvary=agksursigma*(sin(k*xvect-sigma*t))
    fprofx=cosh(k*(yvect+d))/cosh(k*d)
    fprofy=sinh(k*(yvect+d))/cosh(k*d)
    fx=fvarx'*fprofx
    fy=fvary'*fprofy
endfunction
//=========================================================================
i=0
//boucle en temps
for t=0:0.05:1
    i=i+1
    if i<>1 then yprec=y; end
    y=a*cos(k*x-sigma*t)
    if option==2 then y=y+a*cos(-k*x-sigma*t); end // agitation
    if i==1 then yprec=y; end
    xfpolys([x';LONG;0],[yprec';-1;-1],[id1])
    plot(x',yprec',"w")
```

```
0074      xfpolys([x';LONG;0],[y';-1;-1],[id2])
0075      deltay=max(y,yprec)
0076      num=string(t)
0077      xpause(1000);
0078
0079      title(titre+num+' sec', 'position',[0.5 0.5],'fontsize',3)
0080
0081      plot2d(x',y')
0082  // dessin des vecteurs vitesse
0083
0084      select option
0085      case -1 then
0086          [fx2, fy2]=vitesse(k,h,agksursigma,t,xvect,yvect,sigma)
0087      case 1 then
0088          [fx1, fy1]=vitesse(k,h,agksursigma,t,xvect,yvect,sigma)
0089
0090      else
0091          [fx1, fy1]=vitesse(k,h,agksursigma,t,xvect,yvect,sigma)
0092          [fx2, fy2]=vitesse(-k,h,-agksursigma,t,xvect,yvect,sigma)
0093      end
0094      fx=fx1+fx2;fy=fy1+fy2
0095      flechmax=0.3
0096      xvect(10)=20;fx(10,:)=flechmax;fy(10,:)=flechmax;
0097
0098      b=get("current_axes");
0099      b.data_bounds=[0,-1;10,0.6];
0100      b.auto_scale="off"
0101
0102      champ(xvect',yvect',fx,fy,arfact=1)
0103
0104      //GIF export
0105      xs2gif(0,'houle3_'+string(i)+'.gif');
0106
0107      // longueur des vecteurs vitesse
0108      lv=sqrt(fx.*fx+fy.*fy)
0109      //delete()
0110
0111  end
```