Programmes

Plot_coast.m

```
%Graphique LEG et STATIONS:

CTDS1=importdata('CTDS1.txt');
CTDS2=importdata('CTDS2.txt');
CTDS3=importdata('CTDS3.txt');

%Lecture des côtes:
cotes=load('27154.dat');
xloncot=cotes(:,1);
ylatcot=cotes(:,2);

%Tracer des côtes:
lwc=1.5;
h0=plot(xloncot,ylatcot,'k');
set(h0,'linewidth',lwc);

xlim=[6.5 7.5];
ylim=[42.8 43.5];
axis([xlim ylim]);

hold on
plot(CTDS1(:,2),CTDS1(:,1),'+r','markers',10)
plot(CTDS2(:,2),CTDS2(:,1),'+b','markers',10)
plot(CTDS3(:,2),CTDS3(:,1),'+g','markers',10)
xlabel('Longitude');
ylabel('Latitude');
legend('Traits de côte','LEG 3 (29/03/17)', 'LEG 4 (30/03/17)', 'LEG 5 (31/01/17)')
title('LEG et STATIONS')

hold on
str=['2','3','4','5','6'];
for k=1:length(str)
   text(CTDS1(k,2)+0.01,CTDS1(k,1)+0.01,str(1,k),'FontSize',14);
end

str1=['1','2','3'];
for l=1:length(str1)
   text(CTDS2(l,2)+0.01,CTDS2(l,1)+0.01,str1(1,l),'FontSize',14);
   text(CTDS3(l,2)+0.01,CTDS3(l,1)+0.01,str1(1,l),'FontSize',14);
end

% %Pour la délimitation dans rapport :
% figure
% plot(xloncot,ylatcot,'k');
% title('La Baie de Saint-Tropez')
% xlabel('Longitude')
% ylabel('Latitude')
% xlim=[6 8];
% ylim=[42 44];
% axis([xlim ylim]);
```

sw_dens0.m

```
function dens = sw_dens0(S,T)

% SW_DENS0   Denisty of sea water at atmospheric pressure
%=========================================================================
% SW_DENS0  $Revision: 1.3 $  $Date: 1994/10/10 04:54:09 $
%         Copyright (C) CSIRO, Phil Morgan 1992
%
% USAGE:  dens0 = sw_dens0(S,T)
%
% DESCRIPTION:
%    Density of Sea Water at atmospheric pressure using
%    UNESCO 1983 (EOS 1980) polynomial.
%
% INPUT:  (all must have same dimensions)
%   S = salinity    [psu      (PSS-78)]
%   T = temperature [degree C (IPTS-68)]
%
% OUTPUT:
%   dens0 = density  [kg/m^3] of salt water with properties S,T,
%         P=0 (0 db gauge pressure)
%
% AUTHOR:  Phil Morgan 92-11-05  (morgan@ml.csiro.au)
%
% DISCLAIMER:
%   This software is provided "as is" without warranty of any kind.
%   See the file sw_copy.m for conditions of use and licence.
%
% REFERENCES:
%     Unesco 1983. Algorithms for computation of fundamental properties of
%     seawater, 1983. _Unesco Tech. Pap. in Mar. Sci._, No. 44, 53 pp.
%
%     Millero, F.J. and  Poisson, A.
%     International one-atmosphere equation of state of seawater.
%     Deep-Sea Res. 1981. Vol28A(6) pp625-629.
%=========================================================================

% CALLER: general purpose, sw_dens.m
% CALLEE: sw_smow.m

%----------------------
% CHECK INPUT ARGUMENTS
%----------------------
if nargin ~=2
   error('sw_dens0.m: Must pass 2 parameters')
end %if

[mS,nS] = size(S);
[mT,nT] = size(T);

if (mS~=mT) | (nS~=nT)
```

```
   error('sw_dens0.m: S,T inputs must have the same dimensions')
end %if

Transpose = 0;
if mS == 1  % a row vector
  S = S(:);
  T = T(:);
  Transpose = 1;
end %if

%----------------------
% DEFINE CONSTANTS
%----------------------
%     UNESCO 1983 eqn(13) p17.

b0 =  8.24493e-1;
b1 = -4.0899e-3;
b2 =  7.6438e-5;
b3 = -8.2467e-7;
b4 =  5.3875e-9;

c0 = -5.72466e-3;
c1 = +1.0227e-4;
c2 = -1.6546e-6;

d0 = 4.8314e-4;

%$$$ dens = sw_smow(T) + (b0 + b1*T + b2*T.^2 + b3*T.^3 + b4*T.^4).*S  ...
%$$$                   + (c0 + c1*T + c2*T.^2).*S.*sqrt(S) + d0*S.^2;

dens = sw_smow(T) + (b0 + (b1 + (b2 + (b3 + b4*T).*T).*T).*T).*S  ...
                  + (c0 + (c1 + c2*T).*T).*S.*sqrt(S) + d0*S.^2;

if Transpose
  dens = dens';
end %if

return
%-------------------------------------------------------------------
```

sw_smow.m

```
function dens = sw_smow(T)

% SW_SMOW    Denisty of standard mean ocean water (pure water)
%=========================================================================
% SW_SMOW  $Revision: 1.3 $  $Date: 1994/10/10 05:51:46 $
%          Copyright (C) CSIRO, Phil Morgan 1992.
%
% USAGE:  dens = sw_smow(T)
%
```

```
% DESCRIPTION:
%    Denisty of Standard Mean Ocean Water (Pure Water) using EOS 1980.
%
% INPUT:
%   T = temperature [degree C (IPTS-68)]
%
% OUTPUT:
%   dens = density  [kg/m^3]
%
% AUTHOR:  Phil Morgan 92-11-05  (morgan@ml.csiro.au)
%
% DISCLAIMER:
%   This software is provided "as is" without warranty of any kind.
%   See the file sw_copy.m for conditions of use and licence.
%
% REFERENCES:
%    Unesco 1983. Algorithms for computation of fundamental properties of
%    seawater, 1983. _Unesco Tech. Pap. in Mar. Sci._, No. 44, 53 pp.
%    UNESCO 1983 p17  Eqn(14)
%
%    Millero, F.J & Poisson, A.
%    INternational one-atmosphere equation of state for seawater.
%    Deep-Sea Research Vol28A No.6. 1981 625-629.    Eqn (6)
%=========================================================================

%----------------------
% CHECK INPUT ARGUMENTS
%----------------------
% TEST INPUTS
if nargin ~= 1
   error('sw_smow.m: Only one input argument allowed')
end %if

Transpose = 0;
[mT,nT] = size(T);
if mT == 1 % a row vector
   T = T(:);
   Tranpose = 1;
end %if

%----------------------
% DEFINE CONSTANTS
%----------------------
a0 = 999.842594;
a1 =   6.793952e-2;
a2 =  -9.095290e-3;
a3 =   1.001685e-4;
a4 =  -1.120083e-6;
a5 =   6.536332e-9;

dens = a0 + (a1 + (a2 + (a3 + (a4 + a5*T).*T).*T).*T).*T;
```

```
if Transpose
  dens = dens';
end %if

return
%-------------------------------------------------------------------
```

tracecotes.m

```
function tracecotes



xlim=[6.6 7];
ylim=[43 43.4];
% lecture des cotes
cotes=load('27154.dat');
xloncot=cotes(:,1);
ylatcot=cotes(:,2);

xlabel('longitude');
ylabel('latitude');

% trace des cotes

lwc=1.5;
h0=plot(xloncot,ylatcot,'k');
set(h0,'linewidth',lwc);

axis([xlim ylim]);

end
```

suplabel.m

```
function [ax,h]=suplabel(text,whichLabel,supAxes)
% PLaces text as a title, xlabel, or ylabel on a group of subplots.
% Returns a handle to the label and a handle to the axis.
%  [ax,h]=suplabel(text,whichLabel,supAxes)
% returns handles to both the axis and the label.
%  ax=suplabel(text,whichLabel,supAxes)
% returns a handle to the axis only.
%  suplabel(text) with one input argument assumes whichLabel='x'
%
% whichLabel is any of 'x', 'y', 'yy', or 't', specifying whether the
% text is to be the xlable, ylabel, right side y-label,
% or title respectively.
%
% supAxes is an optional argument specifying the Position of the
%  "super" axes surrounding the subplots.
%  supAxes defaults to [.08 .08 .84 .84]
%  specify supAxes if labels get chopped or overlay subplots
```

```
%
% EXAMPLE:
%  subplot(2,2,1);ylabel('ylabel1');title('title1')
%  subplot(2,2,2);ylabel('ylabel2');title('title2')
%  subplot(2,2,3);ylabel('ylabel3');xlabel('xlabel3')
%  subplot(2,2,4);ylabel('ylabel4');xlabel('xlabel4')
%  [ax1,h1]=suplabel('super X label');
%  [ax2,h2]=suplabel('super Y label','y');
%  [ax3,h2]=suplabel('super Y label (right)','yy');
%  [ax4,h3]=suplabel('super Title'  ,'t');
%  set(h3,'FontSize',30)
%
% SEE ALSO: text, title, xlabel, ylabel, zlabel, subplot,
%           suptitle (Matlab Central)

% Author: Ben Barrowes <barrowes@alum.mit.edu>

%modified 3/16/2010 by IJW to make axis behavior re "zoom" on exit same as
%at beginning. Requires adding tag to the invisible axes


currax=findobj(gcf,'type','axes','-not','tag','suplabel');

if nargin < 3
 supAxes=[.08 .08 .84 .84];
 ah=findall(gcf,'type','axes');
 if ~isempty(ah)
  supAxes=[inf,inf,0,0];
  leftMin=inf;  bottomMin=inf;  leftMax=0;  bottomMax=0;
  axBuf=.04;
  set(ah,'units','normalized')
  ah=findall(gcf,'type','axes');
  for ii=1:length(ah)
   if strcmp(get(ah(ii),'Visible'),'on')
    thisPos=get(ah(ii),'Position');
    leftMin=min(leftMin,thisPos(1));
    bottomMin=min(bottomMin,thisPos(2));
    leftMax=max(leftMax,thisPos(1)+thisPos(3));
    bottomMax=max(bottomMax,thisPos(2)+thisPos(4));
   end
  end
  supAxes=[leftMin-axBuf,bottomMin-axBuf,leftMax-leftMin+axBuf*2,bottomMax-
bottomMin+axBuf*2];
 end
end
if nargin < 2, whichLabel = 'x';  end
if nargin < 1, help(mfilename); return; end

if ~isstr(text) | ~isstr(whichLabel)
  error('text and whichLabel must be strings')
end
whichLabel=lower(whichLabel);
```

```
ax=axes('Units','Normal','Position',supAxes,'Visible','off','tag','suplabel');
if strcmp('t',whichLabel)
  set(get(ax,'Title'),'Visible','on')
  title(text);
elseif strcmp('x',whichLabel)
  set(get(ax,'XLabel'),'Visible','on')
  xlabel(text);
elseif strcmp('y',whichLabel)
  set(get(ax,'YLabel'),'Visible','on')
  ylabel(text);
elseif strcmp('yy',whichLabel)
  set(get(ax,'YLabel'),'Visible','on')
  ylabel(text);
  set(ax,'YAxisLocation','right')
end

for k=1:length(currax), axes(currax(k));end % restore all other axes

if (nargout < 2)
  return
end
if strcmp('t',whichLabel)
  h=get(ax,'Title');
  set(h,'VerticalAlignment','middle')
elseif strcmp('x',whichLabel)
  h=get(ax,'XLabel');
elseif strcmp('y',whichLabel) | strcmp('yy',whichLabel)
  h=get(ax,'YLabel');
end

%%%ah=findall(gcf,'type','axes');
%%%'sssssssss',kb
```

trajectoire_MVP.m

```
x=importdata('MVP_27mars17.log');
latx=(x.data(:,1)-floor(x.data(:,1)/100)*100)*10/600+floor(x.data(:,1)/100);
lonx=(x.data(:,2)-floor(x.data(:,2)/100)*100)*10/600+floor(x.data(:,2)/100);
cote=load('27154.dat');
cotlon=cote(:,1);
cotlat=cote(:,2);
hold on
plot(cotlon,cotlat,'k')
axis([6.1,7.2,42.7,43.4])
plot(lonx,latx,'m')

% plot(cotlon,cotlat)
axis([6.1,7.2,42.7,43.4])
y=importdata('MVP_28mars17.log');
laty=(y.data(:,1)-floor(y.data(:,1)/100)*100)*10/600+floor(y.data(:,1)/100);
```

```
lony=(y.data(:,2)-floor(y.data(:,2)/100)*100)*10/600+floor(y.data(:,2)/100);
plot(lony,laty,'k')

% plot(cotlon,cotlat)
axis([6.1,7.2,42.7,43.4])
z=importdata('MVP_30mars17.log');
latz=(z.data(:,1)-floor(z.data(:,1)/100)*100)*10/600+floor(z.data(:,1)/100);
lonz=(z.data(:,2)-floor(z.data(:,2)/100)*100)*10/600+floor(z.data(:,2)/100);
plot(lonz,latz,'b')

% plot(cotlon,cotlat)
axis([6.1,7.2,42.7,43.4])
a=importdata('MVP_31mars17.log');
lata=(a.data(:,1)-floor(a.data(:,1)/100)*100)*10/600+floor(a.data(:,1)/100);
lona=(a.data(:,2)-floor(a.data(:,2)/100)*100)*10/600+floor(a.data(:,2)/100);
plot(lona,lata,'g')
```

visualisation.m

```
bleuclair=[0.301 0.745 0.933];
violet=[0.494 0.184 0.556];
jaune=[0.929 0.694 0.125];
bleufonce=[0.0 0.447 0.741];
rouge=[0.85 0.325 0.098];
vert=[0.466 0.674 0.188];

AL727=importdata('matrice_AL7_27.mat');
VAL727=calc_vitesse_accel(AL727);
AL827=importdata('matrice_AL8_27.mat');
VAL827=calc_vitesse_accel(AL827);
AL228=importdata('matrice_AL2_28.mat');
VAL228=calc_vitesse_accel(AL228);
S628=importdata('matrice_#3_28.mat');
VS628=calc_vitesse_accel(S628);
AL730=importdata('matrice_AL7_30.mat');
VAL730=calc_vitesse_accel(AL730);
AL830=importdata('matrice_AL8_30.mat');
VAL830=calc_vitesse_accel(AL830);

hold on
tracecotes
hold on

[al727,Ual727,Val727]=graphe_position(AL727,bleuclair);
[al827,Ual827,Val827]=graphe_position(AL827,violet);
[al228,Ual228,Val228]=graphe_position(AL228,jaune);
[s628,Us628,Vs628]=graphe_position(S628,bleufonce);
[al730,Ual730,Val730]=graphe_position(AL730,rouge);
[al830,Ual830,Val830]=graphe_position(AL830,vert);

title('Trajectoires');
```

```
legend([al727 al827 al228 s628 al730 al830],{...
  strcat('AL7 S7 le 27'),...
    strcat('AL8 S18 le 27')...
    strcat('AL2 S4 le 28'),...
    strcat('#3 S6 le 28'),...
    strcat('AL7 S7 le 30'),...
    strcat('AL8 S18 le 30')...
    });
```

load_SST.m

```
clear;clc;close all

ncfile='SST_MED_SST_L3S_NRT_OBSERVATIONS_010_012_b_1493995667241.nc';
lat_SST=double(ncread(ncfile,'lat'));
lon_SST=double(ncread(ncfile,'lon'));
SST=double(ncread(ncfile,'sea_surface_temperature'));
SST=permute(SST,[3 1 2]);
GL=mean(SST);
GL=permute(GL,[2 3 1]);

figure(1)
clf
pcolor(lon_SST, lat_SST, (GL-273.15)'); colorbar; shading flat
axis([6.5 7.5 42.8 43.5])
caxis([14.1 14.6])
hold on
run Plot_coast.m
title('LEG et STATIONS + Température')
xlabel('Longitude')
ylabel('Latitude')
legend('Température en °C','Traits de côte','LEG 3 (29/03/17)', 'LEG 4 (30/03/17)', 'LEG 5
(31/01/17)')
hold off

figure(2)
clf
pcolor(lon_SST, lat_SST, (GL-273.15)'); colorbar; shading flat
axis([6.1 7.2 42.7 43.4])
caxis([14.1 14.6])
hold on
run trajectoire_VMP.m
title('Trajectoires du MVP + Température')
xlabel('Longitude')
ylabel('Latitude')
legend('Température en °C','Traits de côte','27/03/17', '28/03/17', '30/03/17','31/03/17')
hold off

figure(3)
clf
GH=pcolor(lon_SST, lat_SST, (GL-273.15)'); colorbar; shading flat;
```

```
axis([6.6 7 43 43.4])
caxis([14.1 14.6])
hold on
run visualisation.m
title('Trajectoires des flotteurs + Température')
xlabel('Longitude')
ylabel('Latitude')
legend([GH al727 al827 al228 s628 al730 al830],{...
   strcat('Température en °C'),...
   strcat('AL7 S7 le 27'),...
   strcat('AL8 S18 le 27')...
   strcat('AL2 S4 le 28'),...
   strcat('#3 S6 le 28'),...
   strcat('AL7 S7 le 30'),...
   strcat('AL8 S18 le 30')...
   });
hold off
```

load_CHL.m

```
clear;clc;close all

ncfile='dataset-oc-med-chl-multi-l3-chl_1km_daily-rt-v02_1493968328945.nc';
lat_CHL=double(ncread(ncfile,'lat'));
lon_CHL=double(ncread(ncfile,'lon'));
CHL=double(ncread(ncfile,'CHL'));
CHL=permute(CHL,[3 1 2]);
GL=mean(CHL);
GL=permute(GL,[2 3 1]);

figure(1)
clf
pcolor(lon_CHL, lat_CHL, log10(GL)'); colorbar; shading flat
axis([6.5 7.5 42.8 43.5])
caxis([-1 -0.6])
hold on
run Plot_coast.m
title('LEG et STATIONS + Concentration en chlorophylle a')
xlabel('Longitude')
ylabel('Latitude')
legend('Concentration en chlorophylle a','Traits de côte','LEG 3 (29/03/17)', 'LEG 4 (30/03/17)',
'LEG 5 (31/01/17)')
hold off

figure(2)
clf
pcolor(lon_CHL, lat_CHL, log10(GL)'); colorbar; shading flat
axis([6.1 7.2 42.7 43.4])
caxis([-1 -0.6])
hold on
run trajectoire_VMP.m
```

```
title('Trajectoires du MVP + Concentration en chlorophylle a')
xlabel('Longitude')
ylabel('Latitude')
legend('Concentration en chlorophylle a','Traits de côte','27/03/17', '28/03/17',
'30/03/17','31/03/17')
hold off

figure(3)
clf
GH=pcolor(lon_CHL, lat_CHL, log10(GL)'); colorbar; shading flat;
axis([6.6 7 43 43.4])
caxis([-1 -0.6])
hold on
run visualisation.m
title('Trajectoires des flotteurs + Concentration en chlorophylle a')
xlabel('Longitude')
ylabel('Latitude')
legend([GH al727 al827 al228 s628 al730 al830],{...
   strcat('Concentration en chlorophylle a'),...
   strcat('AL7 S7 le 27'),...
   strcat('AL8 S18 le 27')...
   strcat('AL2 S4 le 28'),...
   strcat('#3 S6 le 28'),...
   strcat('AL7 S7 le 30'),...
   strcat('AL8 S18 le 30')...
   });
hold off
```

load_SeaBird_cnv_file.m

```
clear;clc;close all

delimiterIn = ' ';
headerlinesIn = 229;
filename = 'PHYOCE2017_L3_S2.cnv';
%filename = 'PHYOCE2017_L3_S3.cnv';
%filename = 'PHYOCE2017_L3_S4.cnv';
%filename = 'PHYOCE2017_L3_S5.cnv';
%filename = 'PHYOCE2017_L3_S6.cnv';
%filename = 'PHYOCE2017_L4_S1.cnv';
%filename = 'PHYOCE2017_L4_S2.cnv';
%filename = 'PHYOCE2017_L4_S3.cnv';
%filename = 'PHYOCE2017_L5_S1.cnv';
%filename = 'PHYOCE2017_L5_S2.cnv';
%filename = 'PHYOCE2017_L5_S3.cnv';

A = importdata(filename,delimiterIn,headerlinesIn);
Depth =- A.data(:,7);
TemPot = A.data(:,8);
Dens = A.data(:,12);
Sal = A.data(:,10);
```

```
Oxy = A.data(:,14);
Fluo = A.data(:,13);
Irra = A.data(:,16);
Trans = A.data(:,18);

%%%Première figure :
figure(1)
orient tall

%Tracé des isodensités sur le diagramme TS :
%Création des matrices température et salinité.
Temp=[13.2:0.05:15];
Salt=[37.6:0.05:39];

[a iTemp]=size(Temp);
[a jSalt]=size(Salt);

%Calcul densité.
for i = 1:iTemp
   for j = 1:jSalt
      Pdens(i,j)=sw_dens0(Salt(j),Temp(i))-1000;
   end
end

%Graphique salinité en fonction de la tenpérature :
subplot(2,2,1)
contourf(Salt,Temp,Pdens);
hold on
plot(Sal,TemPot,'k+')
xlabel('Salinité (PSU)')
ylabel('Température (en °C)')

%Graphique température en fonction de la profondeur :
subplot(2,2,2)
plot(TemPot,Depth,'r')
xlabel('Température (en °C)')
ylabel('Profondeur (en m)')
axis([13.6 14.4 -600 0])

%Graphique salinité en fonction de la profondeur :
subplot(2,2,3)
plot(Sal,Depth,'m')
xlabel('Salinité (PSU)')
ylabel('Profondeur (en m)')
axis([37.6 39 -600 0])

%Graphique densité en fonction de la profondeur :
subplot(2,2,4)
plot(Dens,Depth,'b')
xlabel('Densité (en kg/m^3)')
ylabel('Profondeur (en m)')
axis([28.1 29.3 -600 0])
```

```
newStr=strrep(filename,'_',' ');
newStrr=strrep(newStr,'.cnv',' ');
[ax,h3]=suplabel(newStrr ,'t');
set(h3,'FontSize',15)

%%%Deuxième figure :
figure(2)
orient tall

%Graphique de log10(Irradiance) en fonction de la profondeur :
subplot(2,2,1)
plot(log10(Irra),Depth,'color', [1 0.594117647058824 0.592156862745098])
xlabel('log10(Eclairement énergétique)')
ylabel('Profondeur (en m)')
axis([-2 3 -200 0])
grid on

%Graphique oxygène en fonction de la profondeur :
subplot(2,2,2)
plot(Oxy,Depth,'c')
xlabel('Oxygène (en ml/L)')
ylabel('Profondeur (en m)')
axis([3.5 5.5 -600 0])

%Graphique concentration en chlorophylle a en fonction de la profondeur :
subplot(2,2,3)
plot(Fluo,Depth,'g')
xlabel('Concentration en chlorophylle a (en µg/L)')
ylabel('Profondeur (en m)')
axis([-0.1 0.5 -600 0])

%Graphique transmittance en fonction de la profondeur :
subplot(2,2,4)
plot(Trans,Depth)
xlabel('Transmittance (en %)')
ylabel('Profondeur (en m)')
axis([95 101 -600 0])

[ax,h3]=suplabel(newStrr,'t');
set(h3,'FontSize',15)

for k=1:2
   figure(k);
   temp=[newStrr,num2str(k),'.jpg'];
   set(figure(k), 'Units', 'Normalized', 'Position', [0 0 1 1]);
   saveas(gca,temp);
end
```

graphe_position.m

```
function [a,u,v]=graphe_position(X,color)



lat=X(:,4);
lon=X(:,5);

plot(lon,lat);

hold on

vit=calc_vitesse_accel(X);
n=length(X);

for i=1:n-1
   theta(i)=abs(atan((X(i+1,4)-X(i,4))/(X(i+1,5)-X(i,5))));
   u(i)=vit(i,2)*cos(theta(i));
   v(i)=vit(i,2)*sin(theta(i));
   if X(i+1,5)>X(i,5) & X(i+1,4)>X(i,4)
      u(i)=u(i);
      v(i)=v(i);
   end
   if X(i+1,5)>X(i,5) & X(i+1,4)<X(i,4)
      u(i)=u(i);
      v(i)=-v(i);
   end
   if X(i+1,5)<X(i,5) & X(i+1,4)<X(i,4)
      u(i)=-u(i);
      v(i)=-v(i);
   end
   if X(i+1,5)<X(i,5) & X(i+1,4)>X(i,4)
      u(i)=-u(i);
      v(i)=v(i);
   end

end



x=X(1:n-1,5);
y=X(1:n-1,4);



a=quiver(x,y,u',v','color',color);

end
```

graphes_vit.m

```
function [U,graphe,u]=graphes_vit_acc(Y,color)
```

```

n=length(Y);

u=mean(Y(2:n-1,2));
U=num2str(u(1,1));

graphe=plot(Y(:,1),Y(:,2),'color',color);

end
```

graphes_vit_acc.m

```
function graphes_vit_acc(Y,nom)



% figure;
%
% subplot(2,1,1);
% plot(Y(:,1),Y(:,2));
% xlabel('temps(heure)');
% ylabel('vitesse');
% title(nom);
% subplot(2,1,2);
% plot(Y(:,1),Y(:,3));
% xlabel('temps(heure)');
% ylabel('acceleration');

figure;
n=length(Y);
u(1:n,1)=mean(Y(2:n-1,2));
v(1:n,1)=mean(Y(3:n-1,3));
U=num2str(u(1,1));
V=num2str(v(1,1));
subplot(2,1,1);
plot(Y(:,1),u);
hold on
plot(Y(:,1),Y(:,2));
xlabel('temps(heure)');
ylabel('vitesse (m.s^-1)');
title(nom);
legend(U);
subplot(2,1,2);
plot(Y(:,1),v);
hold on
plot(Y(:,1),Y(:,3));
xlabel('temps(heure)');
ylabel('acceleration (m.s^-2)');
legend(V);



end
```

graphes_vit_moy.m

```
function graphes_vit_moy(Y,nom,color)



figure('position',[0 100 1000 500]);
n=length(Y);
u(1:n,1)=mean(Y(2:n-1,2));
U=num2str(u(1,1));
plot(Y(:,1),u);
hold on
plot(Y(:,1),Y(:,2),'color',color);
xlabel('temps(heure)');
ylabel('vitesse (m.s^-1)');
title(nom);
legend(strcat(nom,' mean= ',U,'m.s^-1'));



end
```

calc_vitesse_accel.m

```
function Y=calc_vitesse_accel(X)
%les valeurs des vitesses et des accelerations sont calculees à des temps
%moyen entre deux emissions

n=length(X)-1;
Y=zeros(n,3);

for i=1:n
   Y(i,1)=X(i,3)+(X(i+1,3)-X(i,3))/2;
   Y(i,2)=sqrt(((X(i+1,4)-X(i,4))*2*pi/260*6378*10^3)^2+((X(i+1,5)-
X(i,5))*2*pi/260*6378*10^3)^2)/(X(i+1,3)-X(i,3))/3600;
end

for j=2:n
   Y(j,3)=(Y(j,2)-Y(j-1,2))/(Y(j,1)-Y(j-1,1))/3600;
end

end
```