Softwares for Model Simulations
RESOLVE Project - February 2012

N. Bellomo, and L. Fermo
http://calvino.polito.it/fismat/poli

1 General Indications

The database in the Annex contains the scientific programs implemented to simulate the biological problems treated within the Resolve project. For practical purposes it is divided into three folders:

1. **Keloid**,
2. **Surgery**,
3. **Lung**,

which contain, respectively, the routines used to develop the results presented in the following papers:

[1] C. BIANCA, L. FERMO, Bifurcation diagrams for the moments of a kinetic type model of keloid-immune system competition, Computers and Mathematics with Applications, 61(2):277-288, 2011.

[2] L. FERMO, N. BELLOMO AND D.B LUMENTA, Assessment of surgical strategies for addressing keloids, An optimization problem, Computers and Mathematics with Applications, 62: (2011) 2417-2423.

[3] M. CHILOSI, A. CARLONI, V. POLLETTI, N. BELLOMO, L.FERMO, Modeling Lung Fibrosis, paper in progress, (2012).

Each of these case studies present a different type of computing, namely the first one concern progression and mutation at the cellular scale. The second one an optimization problem, while the third one a problem at the level of mechanics of biological tissues. Indeed, these were the case studies selected at the Mid Term meeting. Therefore their ensemble constitute a significant panorama of the computing activity within the consortium.

All folders contain a text file having the instructions to run the programs which are written in Matlab.

2 Program for Progression and Mutation

The folder **Keloid** contains the scientific program used to develop the results presented in paper [1] cited in Section 1. More precisely, it contains nine files as follows:

1. `Bifurcation`
2. `Initial_Conditions`
3. `Model`
4. `Model_Bif`
5. `NodWei`
6. `Par_alpha`
7. `RK4sist`
8. `Shift_Node`
9. `System`

The main routines are:

`Bifurcation` that gives the bifurcation diagrams of the local number density and the local mean activity when a specific parameter varies in a fixed range. To run this program, it is sufficient to write the name of the file on the command window of Matlab. At this point, Matlab asks to write the values of all the parameters appearing in the model (see p. 280 of [1]). Then the required graphs are authomatically generated.

`Par_alpha` that graphically represents the number density of the cells involved in the model (see [1] for details). To run this program it needs to write `Par_alpha` on the command window of Matlab. Then it is necessary to write the values of all parameters involved in the model and the graphs are automatically given.

This two programs use all the other sub-routines:

`Initial_Conditions` containing the initial conditions;

`NodWei` that defines the zeros and the weights;

`RK4sist` having the instruction to apply the Runge Kutta method;

`System` containing the equations of system we solve;

`Model_Bif` that is used by the program `Bifurcations`;

`Model` that is used by the program `Par_alpha`;

`Shift_Node` that is a technical program aiming to avoid numerical problems.

In the following subsection we give all the programs in details.

2.1 Scientific Program

```
%----------------------PROGRAM BIFURCATION---------------------------------
clear all
close all
clc

%%%%%%%%%%%%%%%%%%%%PARAMETERS%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
global epsilon beta gamma delta betai deltai lambda m j alfa la y x w
beta=input ('Give beta:');
gamma=input ('Give gamma:');
delta=input ('Give delta:');
betai=input ('Give betai:');
deltai=input ('Give deltai:');
lambda=input ('Give lambda:');
epsilon=input ('Give epsilon:');
m=input('Give the number of nodes:');
j=input('Give the number of distribution functions:');
alfa=input('Give array alfa:');

le=length(alfa);

for la=1:le
    [t,f]=Model_bif(alfa(la));
    A=zeros(y,j);
    D=zeros(y,j);
    for r=1:y
        for i=1:j
            for k=1:m
                A(r,i)=A(r,i)+w(k)*x(k)*f(r,(i-1)*m+k);
                D(r,i)=D(r,i)+w(k)*f(r,(i-1)*m+k);
            end
                if D(r,i)~=0
                   DeAc(r,i)=A(r,i)./D(r,i);
                end
        end
    end
%alfa(la)
%pause
%         for i=1:j
%            % DeAc(:,i)
%            % pause
%             for r=1:y-1
%                 if DeAc(1,i)==DeAc(r+1,i)
%                     p(1,i)=DeAc(r+1,i);
%                     p(2:200,i)=0;
%                 else
%                     p(1:200,i)=linspace(min(DeAc(:,i)),max(DeAc(:,i)),200);
%                 end
%             end
%            % p(1:200,i)
%            % pause
%             figure(i)
%             grid
%             plot(alfa(la),p(1:200,i),'r')
%             hold on
%        end
%----------------------DENSITY ANALYSIS-----------------------------------
        for i=1:j
            for r=1:y-1
                if D(1,i)==D(r+1,i)
                    p(1,i)=D(r+1,i);
                    p(2:200,i)=0;
                else
                    p(1:200,i)=linspace(min(D(:,i)),max(D(:,i)),200);
                end
            end
            figure(i)
            grid
```

```
            plot(alfa(la),p(1:200,i),'r')
            hold on
        end
end
%-------------------------------------------------------------------------
```

```
%--------------PROGRAM PAR_ALPHA-------------------------------------------
clear all
close all
clc

%%%%%%%%%%%%%%%%%%%%PARAMETERS%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
global epsilon beta gamma delta betai deltai lambda m j alfa la
beta=input ('Give beta:');
gamma=input ('Give gamma:');
delta=input ('Give delta:');
betai=input ('Give betai:');
deltai=input ('Give deltai:');
lambda=input ('Give lambda:');      %
epsilon=input ('Give epsilon:');
m=input('Give the number of nodes:');
j=input('Give the number of distribution functions:');
alfa=input('Give array alfa:');

le=length(alfa);
for la=1:le
    [t,f]=model(alfa(la));
    hold on
end
%-------------------------------------------------------------------------
```

```
%-----------------PROGRAM INITIAL_CONDITIONS--------------------------------------
function initx=Initial_Conditions
global m j x
initx=zeros(j*m,1);
initx(1:m,1)=10*exp(-100*(x'-1/2).^2);
initx(m+1:2*m,1)=10*exp(-100*(x'-1/2).^2);
initx(4*m+1:5*m,1)=exp(-100*(x'-1/2).^2);
%-------------------------------------------------------------------------
```

```
%-------------------PROGRAM NODWEI-------------------------------------------
function [x,w]=Nodwei(a,b)
global m
x=cos((2*[1:m]-1)/(2*m)*pi);
x=(b-a)/2.*x+(a+b)/2;
x=x(m:-1:1);
s=floor(m/2);
w=zeros(m,1);
teta=zeros(m,1);
for k=1:m
    teta(k)=(2*k-1)*pi/(2*m);
    for p=1:s
        w(k)=w(k)+cos(2*p*teta(k))/(4*p^2-1);
    end
    w(k)=2/m*(1-2*w(k));
end
w=(b-a)*w/2;
%-------------------------------------------------------------------------
```

```
%--------------------PROGRAM RK4sist-----------------------------------------
function [t,f]=RK4sist(fun,t0,tf,initx,n)
dt=(tf-t0)/n;
t(1)=t0;
f(1,:)=initx;
for k=1:n
    t(k+1)=t0+k*dt;
    k1 = feval(fun,t(k),f(k,:));
    k2 = feval(fun,t(k)+dt/2,f(k,:)+dt/2*k1');
    k3 = feval(fun,t(k)+dt/2,f(k,:)+dt/2*k2');
    k4 = feval(fun,t(k)+dt,f(k,:)+dt*k3');
    f(k+1,:)=f(k,:)+dt/6*(k1'+2*k2'+2*k3'+k4');
end
```

```
%-----------------PROGRAM Shift_node---------------------------------------
function [ind2,ind3,ind4]=Shift_node
global m epsilon alfa x w la

x_shift2=x-epsilon*alfa(la);
x_shift3=x-alfa(la);
x_shift4=x-epsilon^2*alfa(la);
ind2=ones(1,m);
ind3=ones(1,m);
ind4=ones(1,m);

for i=1:m
    diff2=abs(x_shift2(i)-x);
    diff3=abs(x_shift3(i)-x);
    diff4=abs(x_shift4(i)-x);
    mini2=diff2(1);
    mini3=diff3(1);
    mini4=diff4(1);
    for j=2:m
        if mini2>diff2(j)
            mini2=diff2(j);
            ind2(i)=j;
        end
        if mini3>diff3(j)
            mini3=diff3(j);
            ind3(i)=j;
        end
        if mini4>diff4(j)
            mini4=diff4(j);
            ind4(i)=j;
        end
    end
end
ind2=ind2+m;
ind3=ind3+2*m;
ind4=ind4+3*m;
%------------------------------------------------------------------------------
```

```
%----------------------PROGRAM SYSTEM----------------------------------------
function fv=System(t,f)

global beta gamma delta betai deltai lambda epsilon m j w x alfa

fv=zeros(m*j,1);
[ind2,ind3,ind4]=Shift_node;
```

```
%--------------DENSITY AND ACTIVITY----------------------------
for i=1:j
    n(i)=0;
     a(i)=0;
     for k=1:m
         n(i)=n(i)+w(k)*f((i-1)*m+k);
         a(i)=a(i)+w(k)*x(k)*f((i-1)*m+k);
     end
end

%------------SYSTEM-------------------------------------------
fv(1:m)=1/3*(epsilon*(epsilon*beta*n(1)-delta*n(2))*f(1:m));
fv(m+1:2*m)=1/3*((epsilon*beta*(n(1)+n(5))-delta*n(5)-n(1))*f(m+1:2*m)+n(1)*f(ind2(1:m)));
fv(2*m+1:3*m)=1/3*((beta*n(1)-(1-beta+epsilon^2*delta)*n(2))*f(2*m+1:3*m)----
epsilon^2*delta*f(2*m+1:3*m)*n(5)+f(ind3(1:m))*n(2)+gamma*f(1:m)*(epsilon*n(1)+n(2)));
fv(3*m+1:4*m)=1/3*(((epsilon*beta-1)*n(2)-delta*n(5))*f(3*m+1:4*m)+lambda*f(2*m+1:3*m)*n(2)+f(ind4(1:m))*n(2));
fv(4*m+1:5*m)=1/3*(betai*(n(2)+n(4)+epsilon^2*n(3))*f(4*m+1:5*m)-1/3*deltai*(a(2)+a(4)+epsilon^2*a(3))*f(4*m+1:5*m));
%-------------------------------------------------------------------------------------------
```

```
%------------------------PROGRAM MODEL-------------------------------------------
function [t,f]=Model(alfa)
global m j y D A x w la

[x,w]=Nodwei(0,3);

initx=Initial_Conditions;

[t,f]=RK4sist('System',0,20,initx,200);
f=3*f;
y=length(t);

A=zeros(y,j);
D=zeros(y,j);


for r=1:y
     for i=1:j
         for k=1:m
             A(r,i)=A(r,i)+w(k)*x(k)*f(r,(i-1)*m+k);
             D(r,i)=D(r,i)+w(k)*f(r,(i-1)*m+k);
         end
     end
end
for r=1:y
    for i=1:j
        if D(r,i)~=0
           DeAc(r,i)=A(r,i)./D(r,i);
        end
    end
end
DeAc(:,2)
pause

cont=1;
Lab=[{'a1/n1'},'a2/n2','a3/n3','a4/n4','a5/n5'];
Lab2=[{'n1'},'n2','n3','n4','n5'];
Lab3=[{'a1'},'a2','a3','a4','a5'];
Titles=[{'NFc'};'AV';'KFc';'Mc';'ISc'];
Color=['b','g','r','y','v','p'];

 for i=1:j
     figure(cont)
     cont=cont+1;
     grid
     plot(D(:,i),DeAc(:,i), Color(la))
     hold on
     grid
     ylabel(Lab(i))
     xlabel(Lab2(i))
```

```
    Title(Titles(i))
    legend('alfa=0.09','alfa=0.5','alfa=0.84');
end


for i=1:j
    figure(cont)
    cont=cont+1;
    grid
    plot(D(:,i),A(:,i), Color(la))
    hold on
    grid
    ylabel(Lab3(i))
    xlabel(Lab2(i))
    Title(Titles(i))
    legend('alfa=0.09','alfa=0.5','alfa=0.84');
end


 for i=1:j
   figure(cont)
   cont=cont+1;
   grid
   plot3(D(:,i),t, A(:,i), Color(la));
   hold on
   grid
   xlabel(Lab2(i));
   ylabel('t');
   zlabel((Lab3(i)));
   legend('alfa=0.09','alfa=0.5','alfa=0.84');
end
%-------------------------------------------------------------------------------
```

```
%-------------------------PROGRAM MODEL_BIF-------------------------------------
function [t,f,x,w]=Model_bif(alfa)
global m j y yi x w t f

[x,w]=Nodwei(0,3);

initx=Initial_Conditions;

[t,f]=RK4sist('System',0,20,initx,200);
f=3*f;
y=length(t);
%-------------------------------------------------------------------------------
```

3 Surgery Action for Keloids

The folder **Surgery** only contains the file "Surgery". In order to run it, it is sufficient to write on the command window of Matlab the name of the file. At this point, the routine generates all the graphs presented in paper [2].

In the following subsection we report the scientific program in detail.

3.1 Scientific Program

```
%---------------PROGRAM SURGERY-----------------------------------------------
%-----------------------------------------------------------------------------
%---------------FIG. 2,  P.2419 OF [1]----------------------------------------
figure(1)
fplot('0.5*(1-exp(-0.4/0.5*t))', [0 7])
hold on
fplot('0.3*(1-exp(-0.2/0.3*t))', [0 7])
hold on
fplot('0.4*(1-exp(-0.3/0.4*t))', [0 7])
```

```
hold on
fplot('0.2*(1-exp(-0.1/0.2*t))', [0 7])
xlabel('Time')
ylabel('Abnormality')
set(gca,'XTicklabel',[])
set(gca,'YTicklabel',[])
%-----------------FIG. 3, P. 2420 OF [1]-------------------------------------
figure(2)
rectangle('Position', [2,2,4,2],'LineWidth',2,'Linestyle','--')
rectangle('Position', [1,1,6,4],'LineWidth',3,'Linestyle','-')
gtext('Keloid')
gtext('Removal area')
gtext('a')
gtext('b')
gtext('a_r')
gtext('b_r')
axis([0,8,0,8])
%---------------FIG. 4, P. 2421 OF [1]-----------------------------------
figure(3)
fplot('0.2/(1+1.5)*(1-exp(-(0.1/0.2)*(1+1.5)*t))', [0 8],'b')
hold on
fplot('0.2/(1+1.1)*(1-exp(-(0.1/0.2)*(1+1.1)*t))', [0 8],'b')
hold on
fplot('0.2/(1+0.8)*(1-exp(-(0.1/0.2)*(1+0.8)*t))', [0 8],'b')
hold on
fplot('0.2/(1+0.5)*(1-exp(-(0.1/0.2)*(1+0.5)*t))', [0 8],'b')
hold on
fplot('0.2/(1+0.2)*(1-exp(-(0.1/0.2)*(1+0.2)*t))', [0 8],'b')
hold on
fplot('0.2*(1-exp(-0.1/0.2*t))', [0 7],'--')
hold on
xlabel('Time')
ylabel('Abnormality')
gtext('z=1.5')
gtext('z=1.1')
gtext('z=0.8')
gtext('z=0.5')
gtext('z=0.2')
set(gca,'XTicklabel',[])
set(gca,'YTicklabel',[])
%--------------FIG. 5 AND FIG. 6, P. 2421 OF [1]---------------------------
a=0.5;
b=1;
z=linspace(0,2,50);
gamma=linspace(0.2,10,50);
k=1;
T=1;
alpha0=1;
for i=1:50
    J(i,1:50)=4*z.^2+2*z*(1+a/b)+gamma(i)./(1+z)-gamma(i)./(T*k*((1+z).^2)).*(exp(-k*T.*(1+z))-ones(1,50));
    [M(i),r(i)]=min(J(i,1:50));
    figure(4)
    stem(gamma(i),z(r(i)))
    hold on
    xlabel('\gamma')
    ylabel('z_m')
end
figure(5)
plot(z,J(40,1:50))
axis([0,1,10,14])
xlabel('z')
ylabel('J')
%--------------------------------------------------------------------------
%--------------------------------------------------------------------------
```

4 Simulation of Lung Dynamics

The folder **Lung** contains the three following files:

1. `Model`;

2. `Equations`;

3. `Initial_conditions`;

useful to perform the simulations presented in paper [3].

To obtain these simulations, firstly open the file `Model`. Then to run this program it is necessary to write on the command window of Matlab the name of the file.

At this point, the routine gives a graphic representation of the system for different times marking the areas susceptible to more stress.

Remark: Do not change the files `Equations` and `Initial_conditions`. They contain the commands for generating the system described in the paper [1] and the initial configuration of the lung, respectively.

In the following subsection we give all the programs.

Scientific Programs

```
%-----------------PROGRAM MODEL-----------------------------------------------
clear all
close all
clc
%%%%%%%%%%%%%% DATES OF INPUT %%%%%%%%
global n cost m last lo lv init l V t
n=input('Gice the number of the complete hexagons n:');
cost=input('Give the elastic constant K:');
V=input('Give the viscosity constant c:');
m=input('Give the mass of the nodes m:');
%---------VARIABLES-----------------------------------------
a=18;
b=21;
l=a/n;
last=4*(n-3)+2;
lv=l/2;
lo=sqrt(2)*l/2;
%---- COMPUTATION OF THE SOLUTION OF SYSTEM-------------------------
[init]=initesag(n,l,last);
[t,f]=ode45('Equations',[0:0.1:20],init);
u=length(t);
%---- COMPUTATION OF STRESS-------------------------------------
x=zeros(4*(n-3)+2,n+1);
y=zeros(4*(n-3)+2,n+1);
for s=1:u
    bx=f(s,1:n+1);
    by=f(s,(1+last)*(n+1)+1:(n+1)*(2+last));
    for i=1:last
        for j=1:n+1
            x(i,j)=f(s,(n+1)*i+j);
            y(i,j)=f(s,(n+1)*(last+i+1)+j);
        end
    end
    %-------------------------------------------------------------------
    for i=2:n+1
         L(i-1)=abs(sqrt((bx(i)-x(1,i))^2+(by(i)-y(1,i))^2)-lo);
         L1(i-1)=abs(sqrt((bx(i)-x(1,i-1))^2+(by(i)-y(1,i-1))^2)-lo);
         L2(i-1)=abs(sqrt((x(1,i)-x(2,i))^2+(y(1,i)-y(2,i))^2)-lv);
     end
     for i=1:n-3
         for j=i+1:n+1
            L3(i,j)=abs(sqrt((x(4*i-1,j)-x(4*i,j))^2+(y(4*i-1,j)-y(4*i,j))^2)-lv);
            L4(i,j)=abs(sqrt((x(4*i+1,j)-x(4*i+2,j))^2+(y(4*i+1,j)-y(4*i+2,j))^2)-lv);
            L5(i,j)=abs(sqrt((x(4*i-1,j)-x(4*i-2,j-1))^2+(y(4*i-1,j)-y(4*i-2,j-1))^2)-lo);
            L6(i,j)=abs(sqrt((x(4*i+1,j)-x(4*i,j))^2+(y(4*i+1,j)-y(4*i,j))^2)-lo);
            L7(i,j)=abs(sqrt((x(4*i-2,j)-x(4*i-1,j))^2+(y(4*i-2,j)-y(4*i-1,j))^2)-lo);
         end
         for j=i+1:n
             L8(i,j)=abs(sqrt((x(4*i,j+1)-x(4*i+1,j))^2+(y(4*i,j+1)-y(4*i+1,j))^2)-lo);
         end
```

```
    end
    M=max([max(L),max(L1),max(L2),max(max(L3')),max(max(L4')),max(max(L5')),max(max(L6')),max(max(L7')),max(max(L8))]);
   %------FIGURES-------------------------------------------------------
   figure(s)
    for i=2:n+1
        if M==L(i-1) && s>1
           plot([bx(i),x(1,i)],[by(i),y(1,i)],'LineWidth',2,'Color',[0 0 0])
        else
           plot([bx(i),x(1,i)],[by(i),y(1,i)],'k')
        end
        hold on
         if M==L1(i-1) && s>1
           plot([bx(i),x(1,i-1)],[by(i),y(1,i-1)],'LineWidth',2,'Color',[0 0 0])
         else
           plot([bx(i),x(1,i-1)],[by(i),y(1,i-1)],'k')
         end
    end
    hold on
   for j=1:n+1
       if M==L2(i-1) && s>1
           plot([x(1,j),x(2,j)],[y(1,j), y(2,j)],'LineWidth',2,'Color',[0 0 0])
       else
          plot([x(1,j),x(2,j)],[y(1,j), y(2,j)],'k')
       end
   end
    hold on
    for i=1:n-3
        for j=i+1:n+1
            if M==L3(i,j)&& s>1
             plot([x(4*i-1,j),x(4*i,j)],[y(4*i-1,j), y(4*i,j)],'LineWidth',2,'Color',[0 0 0])
            else
               plot([x(4*i-1,j),x(4*i,j)],[y(4*i-1,j), y(4*i,j)],'k')
           end
            if M==L4(i,j) && s>1
               plot([x(4*i+1,j),x(4*i+2,j)],[y(4*i+1,j), y(4*i+2,j)],'LineWidth',2,'Color',[0 0 0])
            else
               plot([x(4*i+1,j),x(4*i+2,j)],[y(4*i+1,j), y(4*i+2,j)],'k')
            end
            if M==L5(i,j)
               plot([x(4*i-1,j),x(4*i-2,j-1)],[y(4*i-1,j), y(4*i-2,j-1)],'LineWidth',2,'Color',[0 0 0])
            else
               plot([x(4*i-1,j),x(4*i-2,j-1)],[y(4*i-1,j), y(4*i-2,j-1)],'k')
            end
            if M==L6(i,j) && s>1
               plot([x(4*i+1,j),x(4*i,j)],[y(4*i+1,j), y(4*i,j)],'LineWidth',2,'Color',[0 0 0])
            else
               plot([x(4*i+1,j),x(4*i,j)],[y(4*i+1,j), y(4*i,j)],'k')
            end
           if M==L7(i,j) && s>1
               plot([x(4*i-2,j),x(4*i-1,j)],[y(4*i-2,j), y(4*i-1,j)],'LineWidth',2,'Color',[0 0 0])
           else
           plot([x(4*i-2,j),x(4*i-1,j)],[y(4*i-2,j), y(4*i-1,j)],'k')
           end
        end
    end
    hold on
    for i=1:n-3
        for j=i+1:n
            if M==L8(i,j) && s>1
               plot([x(4*i,j+1),x(4*i+1,j)],[y(4*i,j+1), y(4*i+1,j)],'LineWidth',2,'Color',[0 0 0])
            else
               plot([x(4*i,j+1),x(4*i+1,j)],[y(4*i,j+1), y(4*i+1,j)],'k')
            end
        end
    end
    hold on
    plot([init((n+1)*last+n-2),init((n+1)*(last+1))],[init((n+1)*(2*last+1)+n-2),init(2*(n+1)*(last+1))],'k')
    bc=y(4*(n-3)+2,1);
    axis([1,24,0,bc+7])
end

%-------------------------------------------------------------------------
```

```
%-----------------PROGRAM EQUATIONS-----------------------------------------------
function df=Equations(t,f)
global n last cost lo lv V Fxs
df=zeros(4*(n+1)*(last+1),1);
df(1:2*(n+1)*(last+1))=f(2*(n+1)*(last+1)+1:4*(n+1)*(last+1));

if t<=1
for i=1:last
   Fxs(i)=-0.8/21*f((n+1)*(last+i+1)+n+1)+1;
end
Fybasso=Fxs(1);
else
    Fybasso=0;
    Fxs(1:last)=zeros(1,last);
end
for j=2:n+1
    ax=cost*lo*(f(j)-f(n+j))/sqrt((f(j)-f(n+j))^2+(f(n+1+last*(n+1)+j)-f((n+1)+last*(n+1)+n+j))^2)-cost*(f(j)-f(n+j));
    bx=cost*lo*(f(j)-f(n+1+j))/sqrt((f(j)-f(n+1+j))^2+(f(n+1+last*(n+1)+j)-f(2*(n+1)+last*(n+1)+j))^2)-cost*(f(j)-f(n+1+j));
    df(2*(n+1)*(last+1)+j)=ax+bx;
end
for j=1:n
    a=cost*lv*(f(n+1+j)-f(2*(n+1)+j))/sqrt((f(n+1+j)-f(2*(n+1)+j))^2+(f((last+2)*(n+1)+j)-f((last+3)*(n+1)+j))^2)-cost*(f(n+1+j
    b=cost*lo*(f(n+1+j)-f(j))/sqrt((f(n+1+j)-f(j))^2+(f((last+2)*(n+1)+j)-f((last+1)*(n+1)+j))^2)-cost*(f(n+1+j)-f(j));
    c=cost*lo*(f(n+1+j)-f(j+1))/sqrt((f(n+1+j)-f(j+1))^2+(f((last+2)*(n+1)+j)-f((last+1)*(n+1)+j+1))^2)-cost*(f(n+1+j)-f(j+1));
    if j==1
        df((n+1)*(3+2*last)+j)=a+c-Fxs(2);
    else
       df((n+1)*(3+2*last)+j)=a+b+c;
    end
end
for j=1:n
    a=cost*lv*(f(2*(n+1)+j)-f(n+1+j))/sqrt((f(2*(n+1)+j)-f(n+1+j))^2+(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+2*(n+1)+j))^2)-
    cost*(f(2*(n+1)+j)-f(n+1+j));
    c=cost*lo*(f(2*(n+1)+j)-f(3*(n+1)+j+1))/sqrt((f(2*(n+1)+j)-f(3*(n+1)+j+1))^2+(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+4*(n+1)+
    cost*(f(2*(n+1)+j)-f(3*(n+1)+j+1));
    b=cost*lo*(f(2*(n+1)+j)-f(3*(n+1)+j))/sqrt((f(2*(n+1)+j)-f(3*(n+1)+j))^2+(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+4*(n+1)+j))^
    -cost*(f(2*(n+1)+j)-f(3*(n+1)+j));
    if j==1
       df((n+1)*(4+2*last)+j)=a+c-Fxs(3);
    else
       df((n+1)*(4+2*last)+j)=a+b+c;
    end
end
 for i=1:n-3
     k=4*i-1;
      for j=i+1:n
          a=cost*lv*(f((n+1)*k+j)-f((n+1)*(k+1)+j))/sqrt((f((n+1)*k+j)-f((n+1)*(k+1)+j))^2
          +(f((last+k+1)*(n+1)+j)-f((n+1)*(last+2+k)+j))^2)
          -cost*(f((n+1)*k+j)-f((n+1)*(k+1)+j));
          b=cost*lo*(f((n+1)*k+j)-f((n+1)*(k-1)+j-1))/sqrt((f((n+1)*k+j)-f((n+1)*(k-1)+j-1))^2
          +(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j-1))^2)
          -cost*(f((n+1)*k+j)-f((n+1)*(k-1)+j-1));
          c=cost*lo*(f((n+1)*k+j)-f((n+1)*(k-1)+j))/sqrt((f((n+1)*k+j)-f((n+1)*(k-1)+j))^2
          +(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j))^2)
          -cost*(f((n+1)*k+j)-f((n+1)*(k-1)+j));
          if j==i+1
            df((n+1)*(2*last+k+2)+j)=a+b+c-Fxs(k);
           else
            df((n+1)*(2*last+k+2)+j)=a+b+c;
          end
       end
    k=4*i;
       for j=i+1:n
           a=cost*lv*(f((n+1)*k+j)-f((n+1)*(k-1)+j))/sqrt((f((n+1)*k+j)-f((n+1)*(k-1)+j))^2
           +(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j))^2)
           -cost*(f((n+1)*k+j)-f((n+1)*(k-1)+j));
           b=cost*lo*(f((n+1)*k+j)-f((n+1)*(k+1)+j-1))/sqrt((f((n+1)*k+j)-f((n+1)*(k+1)+j-1))^2
           +(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j-1))^2)
           -cost*(f((n+1)*k+j)-f((n+1)*(k+1)+j-1));
```

```
            c=cost*lo*(f((n+1)*k+j)-f((n+1)*(k+1)+j))/sqrt((f((n+1)*k+j)-f((n+1)*(k+1)+j))^2
            +(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j))^2)
            -cost*(f((n+1)*k+j)-f((n+1)*(k+1)+j));
            if j==i+1
               df((n+1)*(2*last+k+2)+j)=a+c-Fxs(k);
            else
               df((n+1)*(2*last+k+2)+j)=a+b+c;
            end
         end
       k=4*i+1;
          for j=i+1:n
            a=cost*lv*(f((n+1)*k+j)-f((n+1)*(k+1)+j))/sqrt((f((n+1)*k+j)-f((n+1)*(k+1)+j))^2
            +(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j))^2)
            -cost*(f((n+1)*k+j)-f((n+1)*(k+1)+j));
            b=cost*lo*(f((n+1)*k+j)-f((n+1)*(k-1)+j))/sqrt((f((n+1)*k+j)-f((n+1)*(k-1)+j))^2
            +(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j))^2)
            -cost*(f((n+1)*k+j)-f((n+1)*(k-1)+j));
            c=cost*lo*(f((n+1)*k+j)-f((n+1)*(k-1)+j+1))/sqrt((f((n+1)*k+j)-f((n+1)*(k-1)+j+1))^2
            +(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j+1))^2)
            -cost*(f((n+1)*k+j)-f((n+1)*(k-1)+j+1));
            if j==i+1
               df((n+1)*(2*last+k+2)+j)=a+b+c-Fxs(k);
            else
               df((n+1)*(2*last+k+2)+j)=a+b+c;
            end
          end
 end
for i=1:n-4
    k=4*i+2;
   for j=i+1:n
       a=cost*lv*(f((n+1)*k+j)-f((n+1)*(k-1)+j))/sqrt((f((n+1)*k+j)-f((n+1)*(k-1)+j))^2
       +(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j))^2)
       -cost*(f((n+1)*k+j)-f((n+1)*(k-1)+j));
       c=cost*lo*(f((n+1)*k+j)-f((n+1)*(k+1)+j+1))/sqrt((f((n+1)*k+j)-f((n+1)*(k+1)+j+1))^2
       +(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j+1))^2)
       -cost*(f((n+1)*k+j)-f((n+1)*(k+1)+j+1));
       b=cost*lo*(f((n+1)*k+j)-f((n+1)*(k+1)+j))/sqrt((f((n+1)*k+j)-f((n+1)*(k+1)+j))^2
       +(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j))^2)
       -cost*(f((n+1)*k+j)-f((n+1)*(k+1)+j));
               if j==i+1
                      df((n+1)*(2*last+k+2)+j)=a+c-Fxs(k);
               else
                      df((n+1)*(2*last+k+2)+j)=a+b+c;
               end
   end
end
for j=2:n+1
    ay=cost*lo*(f(n+1+last*(n+1)+j)-f((n+1)+last*(n+1)+n+j))/sqrt((f(j)-f(n+j))^2
    +(f(n+1+last*(n+1)+j)-f(n+1+last*(n+1)+n+j))^2)
    -cost*(f(n+1+last*(n+1)+j)-f((n+1)+last*(n+1)+n+j));
    by=cost*lo*(f(n+1+last*(n+1)+j)-f(2*(n+1)+last*(n+1)+j))/sqrt((f(j)-f(n+1+j))^2
    +(f(n+1+last*(n+1)+j)-f(2*(n+1)+last*(n+1)+j))^2)
    -cost*(f(n+1+last*(n+1)+j)-f(2*(n+1)+last*(n+1)+j));
    df(3*(n+1)*(last+1)+j)=ay+by-Fybasso;
end
for j=1:n
    a=cost*lv*(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+2*(n+1)+j))/sqrt((f(2*(n+1)+j)-f(n+1+j))^2
    +(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+2*(n+1)+j))^2)
    -cost*(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+2*(n+1)+j));
    c=cost*lo*(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+4*(n+1)+j+1))/sqrt((f(2*(n+1)+j)-f(3*(n+1)+j+1))^2
    +(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+4*(n+1)+j+1))^2)
    -cost*(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+4*(n+1)+j+1));
    b=cost*lo*(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+4*(n+1)+j))/sqrt((f(2*(n+1)+j)-f(3*(n+1)+j))^2
    +(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+4*(n+1)+j))^2)
    -cost*(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+4*(n+1)+j));
    if j==1
       df((n+1)*(4+3*last)+j)=a+c;
    else
       df((n+1)*(4+3*last)+j)=a+b+c;
    end
end
for j=1:n
    a=cost*lv*(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+2*(n+1)+j))/sqrt((f(2*(n+1)+j)-f(n+1+j))^2
    +(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+2*(n+1)+j))^2)
```

```
    -cost*(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+2*(n+1)+j));
    c=cost*lo*(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+4*(n+1)+j+1))/sqrt((f(2*(n+1)+j)-f(3*(n+1)+j+1))^2
    +(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+4*(n+1)+j+1))^2)
    -cost*(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+4*(n+1)+j+1));
    b=cost*lo*(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+4*(n+1)+j))/sqrt((f(2*(n+1)+j)-f(3*(n+1)+j))^2
    +(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+4*(n+1)+j))^2)
    -cost*(f(last*(n+1)+3*(n+1)+j)-f(last*(n+1)+4*(n+1)+j));
    if j==1
       df((n+1)*(5+3*last)+j)=a+c;
    else
       df((n+1)*(5+3*last)+j)=a+b+c;
    end
end
  for i=1:n-3
     k=4*i-1;
        for j=i+1:n
            a=cost*lv*(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j))/sqrt((f((n+1)*k+j)-f((n+1)*(k+1)+j))^2
            +(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j))^2)
            -cost*(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j));
            b=cost*lo*(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j-1))/sqrt((f((n+1)*k+j)-f((n+1)*(k-1)+j-1))^2
            +(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j-1))^2)
            -cost*(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j-1));
            c=cost*lo*(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j))/sqrt((f((n+1)*k+j)-f((n+1)*(k-1)+j))^2
            +(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j))^2)
            -cost*(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j));
            df((n+1)*(3*last+k+3)+j)=a+b+c;
        end
    k=4*i;
       for j=i+1:n
           a=cost*lv*(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j))/sqrt((f((n+1)*k+j)-f((n+1)*(k-1)+j))^2
           +(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j))^2)
           -cost*(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j));
           b=cost*lo*(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j-1))/sqrt((f((n+1)*k+j)-f((n+1)*(k+1)+j-1))^2
           +(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j-1))^2)
           -cost*(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j-1));
           c=cost*lo*(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j))/sqrt((f((n+1)*k+j)-f((n+1)*(k+1)+j))^2
           +(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j))^2)
           -cost*(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j));
           if j==i+1
              df((n+1)*(3*last+k+3)+j)=a+c;
           else
              df((n+1)*(3*last+k+3)+j)=a+b+c;
           end
       end
    k=4*i+1;
        for j=i+1:n
           a=cost*lv*(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j))/sqrt((f((n+1)*k+j)-f((n+1)*(k+1)+j))^2
           +(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j))^2)
           -cost*(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j));
           b=cost*lo*(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j))/sqrt((f((n+1)*k+j)-f((n+1)*(k-1)+j))^2
           +(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j))^2)
           -cost*(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j));
           c=cost*lo*(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j+1))/sqrt((f((n+1)*k+j)-f((n+1)*(k-1)+j+1))^2
           +(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j+1))^2)
           -cost*(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j+1));
           df((n+1)*(3*last+k+3)+j)=a+b+c;
        end
 end
for i=1:n-4
     k=4*i+2;
     for j=i+2:n
        a=cost*lv*(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j))/sqrt((f((n+1)*k+j)-f((n+1)*(k-1)+j))^2
        +(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j))^2)
        -cost*(f((last+k+1)*(n+1)+j)-f((last+k)*(n+1)+j));
       b=cost*lo*(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j))/sqrt((f((n+1)*k+j)-f((n+1)*(k+1)+j))^2
       +(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j))^2)
       -cost*(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j));
       c=cost*lo*(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j+1))/sqrt((f((n+1)*k+j)-f((n+1)*(k+1)+j+1))^2
       +(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j+1))^2)
       E-cost*(f((last+k+1)*(n+1)+j)-f((last+k+2)*(n+1)+j+1));
        if j==i+2
           df((n+1)*(3*last+k+3)+j)=a+c;
        else
           df((n+1)*(3*last+k+3)+j)=a+b+c;
```

```
        end
    end
end
df(2*(n+1)*(last+1)+1:4*(n+1)*(last+1))=df(2*(n+1)*(last+1)+1:4*(n+1)*(last+1))-V*df(1:2*(n+1)*(last+1));
%-------------------------------------------------------------------------------
```

```
%---------PROGRAM  INITIAL_CONDITIONS-------------------------------------------
function [init]=initesag(n,l,last)
init=zeros(1,4*(n+1)*(last+1));
%-----------------------------------------------------------------
for j=1:n+1
    x(1:2,j)=4+l*(j-1);
end
for j=2:n+1
    x(3:4,j)=4+l/2+l*(j-2);
end
%-----------------------------------------------------------------
for i=5:4:4*(n-2)
    x(i:i+3,:)=x(1:4,:);
end
for k=1:n-3
    for i=4*(n-2):-4:8+4*(k-1) % 8+k
        c=[x(i-7:i-6,k);zeros(2,1)];
        x(i-3:i,:)=x(i-7:i-4,:)-[zeros(4,k-1),c,zeros(4,n+1-k)];
    end
end
 for k=2:n-2
     for i=4*(n-2):-4:1+4*(k-1)
         c=[zeros(2,1);x(i-5:i-4,k)];
         x(i-3:i,:)=x(i-3:i,:)-[zeros(4,k-1),c,zeros(4,n+1-k)];
     end
 end
x(1+4*(n-2):2+4*(n-2),n-1:n+1)=x(1:2,n-1:n+1);
for i=1:last
    y(i,:)=4+l/2*i*ones(1,n+1);
end
init(1)=0;
init(2:n+1)=x(3,2:n+1);
init((1+last)*(n+1)+1:(n+1)*(2+last))=4*ones(1,n+1);
for i=1:last
     for j=1:n+1
         init((n+1)*i+j)=x(i,j);
         init((n+1)*(last+i+1)+j)=y(i,j);
     end
end
%-------------------------------------------------------------------------------
```