# Aug 1, 2023 zoom with Arthur

## Checklist for finite strain (email)

throughout the code, look for ep1\_xx xy, etc that’s **F deformation gradient tensor**

and look for dep1\_xx xy, etc that’s **L velocity gradient tensor**

except I set L in a different way, you need to code L = [∂vx/∂x ∂vx/dy; ∂vy/dx - ∂vx/∂x]

### 1-first initialize F (SiStER\_Initialize) [done]

% FINITE PLASTIC STRAIN

% components of the deformation gradient tensor F

ep1\_xxm = ones(size(ep));

ep1\_yym = ones(size(ep));

ep1\_xym = zeros(size(ep));

ep1\_yxm = zeros(size(ep));

initialize F (the 4 components of F) on all the markers

### 2-then add a SiStER\_update\_finite\_strain to the main, after SiStER update stresses for example [done]

SiStER % mv SiStER\_update\_finite\_plastic\_strain.m SiStER\_update\_finite\_strain.m

### 3. dont forget to output your finite strain (save to a mat file) [done]

% OUTPUT VARIABLES OF INTEREST (prior to rotation & advection)

if (mod(t,dt\_out)==0 && dt\_out>0) || t==1 || t==Nt % SAVING SELECTED OUTPUT

disp('SAVING SELECTED VARIABLES TO OUTPUT FILE')

filename=num2str(t);

[etam]=SiStER\_interp\_shear\_nodes\_to\_markers(etas,x,y,xm,ym,icn,jcn); % to visualize viscosity on markers

% saving plastic strain tensor (pst)

pst.ep1\_xxm = ep1\_xxm;

pst.ep1\_xym = ep1\_xym;

pst.ep1\_yym = ep1\_yym;

pst.ep1\_yxm = ep1\_yxm;

pst.ep2\_xxm = ep2\_xxm;

pst.ep2\_xym = ep2\_xym;

pst.ep2\_yym = ep2\_yym;

pst.ep2\_yxm = ep2\_yxm;

save(filename,'X','Y','vx','vy','p','time','xm','ym','etam','rhom','BC','etan','Tm','im','idm','epsIIm','sxxm','sxym','ep','epNH','icn','jcn','qd','topo\_x','topo\_y','Rm','xim','fcm','pst')

end

### 4. INSIDE THE UPDATE PLASTIC STRAIN FUNCTION:

**at every time step you must update F**

##### ***FIRST assemble L***

∂vy/∂y = - ∂vx/∂x

L = [∂vx/∂x ∂vx/dy; ∂vy/dx - ∂vx/∂x]

∂vx/∂x = EXX

EXY = .5 (∂vx/∂y + ∂vy/dx)

rotation rate om = .5 (∂vx/∂y - ∂vy/dx)

∂vx/dy = EXY+om;

∂vy/dx = ( EXY-om);

* get EXY from nodes to markers (interpolate shear nodes to markers)

[rrs]=SiStER\_get\_rotation\_rate(vx,vy,dx,dy,BC); % full rotation rate on shear nodes

[om]=SiStER\_interp\_shear\_nodes\_to\_markers(rrs,x,y,xm,ym,icn,jcn); % rotation rate on markers

now you have rotation rate om on markers

[EXX,EXY]=SiStER\_get\_strain\_rate(vx,vy,dx,dy,BC);

<> was called by SiStER\_flow\_solve

[EXYm]=SiStER\_interp\_shear\_nodes\_to\_markers(EXY,x,y,xm,ym,icn,jcn); % rotation rate on markers

<> [added; was in SiStER\_stress\_update.m but it was not called anywhere]

[EXXm]=SiStER\_interp\_normal\_nodes\_to\_markers(EXX,xc,yc,xm,ym,icn,jcn); % rotation rate on markers

now you have strain rates on markers

* then you can calculate L on the markers

L = [EXXm EXYm+om; EXYm-om -EXXm];

thats the velocity gradient tensor

##### 

##### ***THEN update F [done]***

ep1\_xx, ep1\_xy, yy, yx = deformation gradient tensor F (initialized at identity)

the L tensor (velocity gradient tensor) is dep1\_xx, xy, yy, yx

Euler 1st order forward discretization of dFij/dt = LikFkj [Li et al., 2014 JGR subduction]:

%% ACTUAL FINITE STRAIN UPDATE

ep1\_xxm\_new = ep1\_xxm + dt\_m\*(dep1\_xxm.\*ep1\_xxm + dep1\_xym.\*ep1\_yxm);

ep1\_xym\_new = ep1\_xym + dt\_m\*(dep1\_xxm.\*ep1\_xym + dep1\_xym.\*ep1\_yym);

ep1\_yym\_new = ep1\_yym + dt\_m\*(dep1\_yxm.\*ep1\_xym + dep1\_yym.\*ep1\_yym);

ep1\_yxm\_new = ep1\_yxm + dt\_m\*(dep1\_yxm.\*ep1\_xxm + dep1\_yym.\*ep1\_yxm);

ep1\_xxm = ep1\_xxm\_new;

ep1\_xym = ep1\_xym\_new;

ep1\_yym = ep1\_yym\_new;

ep1\_yxm = ep1\_yxm\_new;

### between time steps you must advect F on the markers

**see marker\_patch (SiStER\_patch\_marker\_holes.m) [done]**

make sure F components are handled in ***SiStER\_move\_remove\_and\_reseed\_markers***; [done]

and then (you can do it outside of the code, once youve output the components of F to a mat file)

### OUTSIDE THE CODE\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**at the end get stretch ellipse**

Definition

V = (FF^T)^(1/2)

BUT numerically its better to calculate eigenvalues of F\*F^T and THEN take the sqrt of the eigenvalues

%% GET STRETCH ELLIPSE

F1 = [ep1\_xxm ep1\_xym; ep1\_yxm ep1\_yym];

V1 = (F1\*F1');

[VV1,D1] = eig(V1);

str\_min1 = sqrt(D1(1,1));

str\_max1 = sqrt(D1(2,2));

str\_max1\_vx = VV1(1,2);

str\_max1\_vy = VV1(2,2);