14 – finite element method – density growth - theory

```
% phd.m
%
% author: Cecilia
% date: 09/08/05

load THESIS_TOPIC

while (funding==true)
    data = run_experiment(THESIS_TOPIC);
    GOOD_ENOUGH = query(advisor);
    if (data > GOOD_ENOUGH)
        graduate();
        break
    else
        THESIS_TOPIC = new();
        years_in_gradschool += 1;
    end
end
```

www.phdcomics.com

14 – density growth

day	date		topic
tue	jan	07	motivation - everything grows!
thu	jan	09	basics maths - notation and tensors
tue	jan	14	project example - growing skin
thu	jan	16	kinematics - growing brains
tue	jan	21	basic kinematics - large deformation and growth
thu	jan	23	kinematics - growing hearts
tue	jan	28	kinematics - growing leaflets
thu	jan	30	basic balance equations - closed and open systems
tue	feb	04	basic constitutive equations - growing muscle
thu	feb	06	basic constitutive equations - growing tumors
tue	feb	11	volume growth - finite elements for growth - theory
thu	feb	13	volume growth - finite elements for growth - matlab
tue	feb	18	basic constitutive equations - growing bones
thu	feb	20	density growth - finite elements for growth
tue	feb	25	density growth - growing bones
thu	feb	27	everything grows! - midterm summary
tue	mar	04	midterm
thu	mar	06	remodeling - remodeling arteries and tendons
tue	mar	11	class project - discussion, presentation, evaluation
thu	mar	13	class project - discussion, presentation, evaluation
thu	mar	14	written part of final projects due

where are we???

homework III - revise your final project

due 02/27/14, 11:00am, 530-127

You can bring a hardcopy of your homework to class or upload it to coursework. You can drop off late homework in a box in front of Durand 217 or upload it. Please mark with date and time @drop off. We will take off 1/10 of points for each 24 hours late, every 12pm after due date. This homework will count 10% towards your final grade.

problem 1 - growth tensors

We have introduced different growth tensors F^g in class. Discuss the following growth tensors.

1.1 $F^g = \vartheta\, I$

1.2 $F^g = I + [\vartheta - 1]\, f_0 \otimes f_0$

1.3 $F^g = I + [\vartheta - 1]\, s_0 \otimes s_0$

1.4 $F^g = \vartheta\, I + [1 - \vartheta]\, n_0 \otimes n_0$

homework #03

problem 2 - growth tensors

Assume the following microstructural vectors, $f_0 = [1,0,0]^t$, $s_0 = [0,1,0]^t$, and $n_0 = [0,0,1]^t$ aligned with the cartesian coordinates, and a growth multiplier of $\vartheta = 2$.

2.1 Calculate the four growth tensors F^g from 1.1 to 1.4.

2.2 Calculate the volume change of a cube of unit length for all four growth tensors F^g from 1.1 to 1.4 using the Jacobian $J^g = \det(F^g)$.

2.3 Draw a cubic block of tissue of unit length in a three-dimensional coordinate system. Add the unit vectors $dX_1 = [1,0,0]^t$, $dX_2 = [0,1,0]^t$, and $dX_3 = [0,0,1]^t$. For each of the growth tensors F^g in 1.1 to 1.4, calculate and illustrate the deformed vectors dx_1, dx_2 and dx_3 using $dx = F^g \cdot dX$. Illustrate the grown block.

homework #03

problem 3 - growing bones in matlab

Last year's class paper "Computational modeling of bone density profiles in response to gait: a subject-specific approach" by Henry Pang, Abhishek Shiwalkar, Chris Madormo, and Rebecca Taylor describes bone growth in the tibia. Read the paper carefully.

3.1 Download the matlab file package `ME337_MATLAB.tar.gz` from the coursework website or from our lab website `http://biomechanics.stanford.edu`.

3.2 To run the example from the class paper, open the main file `nlin_fem.m` in your matlab editor, and make sure that all input file readings are commented out by a % sign in the first column of lines 6 through 22. The only active input line should be line 10 `ex_henry`.

For this part of the homework, it is okay to work in groups, especially if you are not very familiar with matlab. If you create the results in a group, however, the results, interpretations, and discussions must be written individually by each group member. Each group member must understand the matlab algorithm.

homework #03

problem 3 - growing bones in matlab

3.3 In the command window, call the main file by typing `nlin_fem` and wait for the mesh to be generated.

3.4 Run the density evolution algorithm for 5 time steps by typing `step,,5`. Describe what you see in the command window and in the graphics window. How many iterations does a typical load step take to find the equilibrium of the nonlinear problem? Focus on load step 5. Report the residuals, i.e., the errors in solution, to demonstrate quadratic convergence of the Newton Raphson scheme.

3.5 Then, run the algorithm for an additional 35 time steps by typing `step,,35` in the command window. Quit the algorithmic environment by typing `quit`. There are two major fields that describe the geometry of a finite element input file, in this case Henry's tibia. Type `q0` to show one of them and then `size(q0)`. What does the `q0` field contain? Type `edof` to show one of them and then `size (edof)`. What does the `edof` field contain?

3.6 Learn to fake your results! Most people show finite element results in terms of colorful plots. Here, the color figures are produced in the subroutine `plot_int.m`. You can easily manipulate a plot by changing the color axis. Type `caxis([0.5 1.5])` to change the color axis and observe what happens. Type `caxis([0.00 1.25])` to change the color axis back again. You can then plot your final figure, e.g., by `print('-depsc','-r300','figure01.eps')`.

homework #03

problem 4 - revise your final project

5.1 Download the final project sample `me337_project_sample.doc` from the coursework website and paste in your title, outline, opening sentence, introduction, schematic drawing, and references from homework II.

5.2 Expand the reference section to at least three key references and seven additional references. Make sure your citations all have the same style. We will take off points if they don't! This is what some picky reviewers criticize first before even reading your paper.

5.3 Revise your introduction and make sure that all your references are cited. The introduction should: (i) contain your catchy opening sentence with citation, (ii) motivate your work, and (iii) give an overview of the current state of the field. It should be one to two columns long.

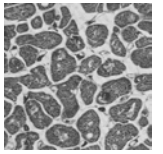
5.4 Draft an outline of all the figures you would like to include in your manuscript. This is the most important step of drafting your paper, since most scientific papers are written around figures. For each figure, create a place holder or the figure itself. Create meaningful figure captions. The figure captions in the sample file are actually not a good example. In the biological literature, captions are usually more detailed and can be several lines long. When you adopt figures from the literature, cite your source. Remember that in a real journal paper, you cannot use other authors' figures without copyright agreement.

homework #03

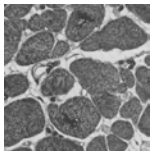
balance equations for open systems - mass

$$D_t \rho_0 = \mathrm{Div}(\boldsymbol{R}) + \mathcal{R}_0$$

mass flux $\boldsymbol{R}$
- cell movement (migration)



mass source $\mathcal{R}_0$
- cell growth (proliferation)
- cell division (hyperplasia)
- cell enlargement (hypertrophy)



biological equilbrium

cowin & hegedus [1976], beaupré, orr & carter [1990], harrigan & hamilton [1992], jacobs,levenston,beaupré,simo & carter [1995], huiskes [2000], carter & beaupré [2001]

growing bone as open system

balance equations for open systems - momentum

- volume specific version

$$\mathrm{D}_t(\rho_0 \boldsymbol{v}) = \mathrm{Div}(\boldsymbol{P} + \boldsymbol{v} \otimes \boldsymbol{R}) + [\boldsymbol{b}_0 + \boldsymbol{v}\mathcal{R}_0 - \nabla_X \boldsymbol{v} \cdot \boldsymbol{R}]$$

- subraction of weighted balance of mass

$$\boldsymbol{v}\,\mathrm{D}_t\rho_0 = \mathrm{Div}(\boldsymbol{v} \otimes \boldsymbol{R}) + \boldsymbol{v}\mathcal{R}_0 - \nabla_X \boldsymbol{v} \cdot \boldsymbol{R}$$

- mass specific version

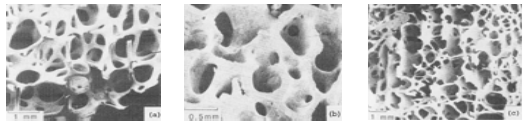
$$\rho_0\,\mathrm{D}_t\boldsymbol{v} = \mathrm{Div}(\boldsymbol{P}) + \boldsymbol{b}_0$$

mechanical equilbrium

growing bone as open system

constitutive equations for open systems

- free energy $\psi_0 = \left[\frac{\rho_0}{\rho_0^*}\right]^n \psi_0^{\mathrm{neo}}(\boldsymbol{F})$
- stress $\boldsymbol{P} = \left[\frac{\rho_0}{\rho_0^*}\right]^n \boldsymbol{P}^{\mathrm{neo}}(\boldsymbol{F})$
- mass flux $\boldsymbol{R} = R_0\,\nabla_X \rho_0$
- mass source $\mathcal{R}_0 = \left[\frac{\rho_0}{\rho_0^*}\right]^{-m} \psi_0(\boldsymbol{F}) - \psi_0^*$

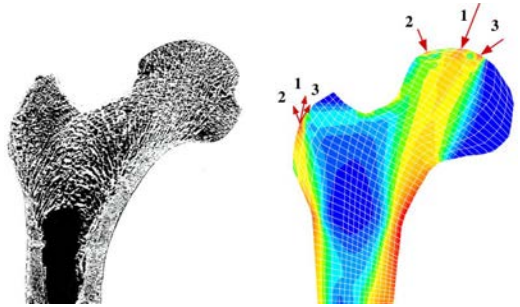


coupling of growth and deformation

gibson & ashby [1999]

growing bone as open system

finite elements for open systems



- dense system of compressive trabaculae carrying stress into calcar region
- secondary arcuate system, medial joint surface to lateral metaphyseal region
- ward ‘s triangle, low density region contrasting dense cortical shaft

carter & beaupré [2001]

growing bone as open system

staggered solution - integration point based

weinans, huiskes & grootenboer [1992], harrigan & hamilton [1992],, [1994], jacobs, levenston, beaupré,simo & carter [1995]

simultaneous solution - node point based

jacobs, levenston, beaupré,simo & carter [1995], fischer, jacobs, levenston & carter [1997], nackenhorst [1997], levenston [1997]]

sequential solution - element based

huiskes, weinans, grootenboer, dalstra, fudala & slooff [1987], carter, orr, fhyrie [1989], beaupré, orr & carter [1990], weinans, huiskes & grootenboer [1992], [1994], jacobs, levenston, beaupré, simo & carter [1995], huiskes [2000], carter & beaupré [2001]

finite elements - integration point based

recipe for finite element modeling

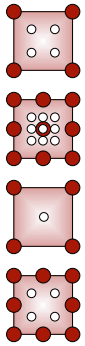
from continuous problem...

$$\begin{aligned} \mathrm{D}_t\rho_0 &= \mathrm{Div}(\boldsymbol{R}) + \mathcal{R}_0 \approx \mathbf{0} \\ \rho_0\,\mathrm{D}_t\boldsymbol{v} &= \mathrm{Div}(\boldsymbol{P}) + \boldsymbol{b}_0 \end{aligned}$$

- temporal discretization — implicit euler backward
- spatial discretization — finite element method
- staggered/simultaneous — newton raphson iteration
- linearization — gateaux derivative

... to linearized discrete initial boundary value problem

integration point based solution of balance of mass



loop over all time steps — nlin_fem
  global newton iteration — nlin_fem
    loop over all elements — element1
      loop over all quadrature points — cnst_law
        local newton iteration $\rho_{0\,n+1}$ — upd_dens
        determine element residual & tangent — cnst_law
      determine global residual and tangent — element1
    determine φ_{n+1} — nlin_fem
  determine state of biological equilibrium — nlin_fem

staggered solution of density and displacements

nlin_fem.m

```
%%% loop over all load steps %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for is = (nsteps+1):(nsteps+inpstep);
   iter = 0;  residuum = 1;
%%% global newton-raphson iteration %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
   while residuum > tol
      iter=iter+1;
      R = zeros(ndof,1); K = sparse(ndof,ndof);
      e_spa = extr_dof(edof,dof);
%%%%% loop over all elements %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      for ie = 1:nel
         [Ke,Re,Ie] = element1(e_mat(ie,:),e_spa(ie,:),i_var(ie,:),mat);
         [K, R, I ] = assm_sys(edof(ie,:),K,Ke,R,Re,I,Ie);
      end
%%%%% loop over all elements %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
      u_inc(:,2)=dt*u_pre(:,2);  R = R - time*F_pre;   dofold = dof;
      [dof,F] = solve_nr(K,R,dof,iter,u_inc);
      residuum= res_norm((dof-dofold),u_inc);
    end
%%% global newton-raphson iteration %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
    time = time + dt;   i_var = I;   plot_int(e_spa,i_var,nel,is);
end
%%% loop over all load steps %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```


integration point based

- discrete residual

check in matlab!

$$\mathbf{R}^{\varphi}_{J} = \mathbf{A}_{e=1}^{n_{\mathrm{el}}} \int_{\mathcal{B}_0^e} \nabla N^j_{\varphi} \cdot \boldsymbol{P}_{n+1}\,\mathrm{d}V$$

- residual of mechanical equilibrium/balance of momentum

righthand side vector for global system of equations

from discrete residual …

- linearization / newton raphson scheme

$$\mathbf{R}^{\varphi k+1}_{Jn+1} = \mathbf{R}^{\varphi k}_{Jn+1} + \mathrm{d}\mathbf{R}^{\varphi}_{J} \doteq 0 \quad \forall\, J = 1, ..., n_{\mathrm{np}}$$

- incremental residual

$$\mathrm{d}\mathbf{R}^{\varphi}_{J} = \sum_{L=1}^{n_{\mathrm{en}}} \mathbf{K}^{\varphi\varphi}_{JL} \cdot \mathrm{d}\boldsymbol{\varphi}_L \qquad \mathbf{K}^{\varphi\varphi}_{JL} = \frac{\mathrm{d}\mathbf{R}^{\varphi}_{J}}{\mathrm{d}\boldsymbol{\varphi}_L}$$

- system of equations

$$\mathbf{K}^{\varphi\varphi}_{JL}\, \mathrm{d}\boldsymbol{\varphi}_L = -\mathbf{R}^{\varphi k}_{Jn+1}$$

- incremental iterative update

$$\Delta\boldsymbol{\varphi}_L = \Delta\boldsymbol{\varphi}_L + \mathrm{d}\boldsymbol{\varphi}_L \qquad \forall\, L = 1, ..., n_{\mathrm{np}}$$

… to linearized residual

linearized residual

- stiffness matrix / iteration matrix

$$\mathbf{K}^{\varphi\varphi}_{JL} = \frac{\partial \mathbf{R}^{\varphi}_{J}}{\partial \boldsymbol{\varphi}_L} = \mathbf{A}^{n_{\mathrm{el}}}_{e=1} \int_{\mathcal{B}^e_0} N^j_\varphi \rho \mathrm{D}_\varphi(\mathrm{D}_t \boldsymbol{v}) N^l_\varphi \mathrm{d}V + \int_{\mathcal{B}^e_0} \nabla N^j_\varphi \cdot \mathrm{D}_F \boldsymbol{P} \cdot \nabla N^l_\varphi \mathrm{d}V$$

4th order tensor - derivatives of 2nd order tensors wrt 2nd order tensor

- linearization of residual wrt nodal dofs

iteration matrix for global system of equations

linearized residual

- stiffness matrix / iteration matrix

check in matlab!

$$\mathbf{K}^{\varphi\varphi}_{JL} = \frac{\partial \mathbf{R}^{\varphi}_{J}}{\partial \boldsymbol{\varphi}_L} = \mathbf{A}^{n_{\mathrm{el}}}_{e=1} \int_{\mathcal{B}^e_0} \nabla N^j_\varphi \cdot \mathrm{D}_F \boldsymbol{P} \cdot \nabla N^l_\varphi \mathrm{d}V$$

- linearization of residual wrt nodal dofs

iteration matrix for global system of equations

quads_2d.m

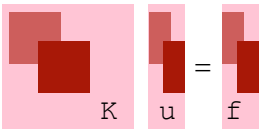
```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [Ke,Re,Ie]=element1(e_mat,e_spa,i_var,mat)
%%% element stiffness matrix Ke, residual Re, internal variables Ie %%%%
Ie = i_var;
Re = zeros(8,1);
Ke = zeros(8,8);
nod=4;              delta = eye(2);
indx=[1;3;5;7];   ex_mat=e_mat(indx);
indy=[2;4;6;8];   ey_mat=e_mat(indy);
%%% integration points %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
g1=0.577350269189626; w1=1;
gp(:,1)=[-g1; g1;-g1; g1];    w(:,1)=[ w1; w1; w1; w1];
gp(:,2)=[-g1;-g1; g1; g1];    w(:,2)=[ w1; w1; w1; w1];
wp=w(:,1).*w(:,2);        xsi=gp(:,1);        eta=gp(:,2);
%%% shape functions and derivatives in isoparametric space %%%%%%%%%%%%%%
N(:,1)=(1-xsi).*(1-eta)/4;    N(:,2)=(1+xsi).*(1-eta)/4;
N(:,3)=(1+xsi).*(1+eta)/4;    N(:,4)=(1-xsi).*(1+eta)/4;
dNr(1:2:8  ,1)=-(1-eta)/4;    dNr(1:2:8  ,2)= (1-eta)/4;
dNr(1:2:8  ,3)= (1+eta)/4;    dNr(1:2:8  ,4)=-(1+eta)/4;
dNr(2:2:8+1,1)=-(1-xsi)/4;    dNr(2:2:8+1,2)=-(1+xsi)/4;
dNr(2:2:8+1,3)= (1+xsi)/4;    dNr(2:2:8+1,4)= (1-xsi)/4;
JT=dNr*[ex_mat;ey_mat]';
%%% element stiffness matrix Ke, residual Re, internal variables Ie %%%%
```

quads_2d.m

```
%%% loop over all integration points %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for ip=1:4
indx=[2*ip-1; 2*ip];  detJ=det(JT(indx,:));
if detJ<10*eps;  disp('Jacobi determinant less than zero!');  end;
JTinv=inv(JT(indx,:));  dNx=JTinv*dNr(indx,:);
F=zeros(2,2);
for j=1:4
   jndx=[2*j-1; 2*j];
   F=F+e_spa(jndx)'*dNx(:,j)';
end
var = i_var(ip);
[A,P,var]=cnst_law(F,var,mat);
Ie(ip) = var;
for i=1:nod
   en=(i-1)*2;
   Re(en+ 1) =  Re(en+ 1) +(P(1,1)*dNx(1,i)' ...
                          + P(1,2)*dNx(2,i)') * detJ * wp(ip);
   Re(en+ 2) =  Re(en+ 2) +(P(2,1)*dNx(1,i)' ...
                          + P(2,2)*dNx(2,i)') * detJ * wp(ip);
end
%%% loop over all integration points %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% element stiffness matrix Ke, residual Re, internal variables Ie %%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

assm_sys.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [K,R,I]=assm_sys(edof,K,Ke,R,Re,I,Ie)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% assemble local element contributions to global tangent & residual %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% input:  edof  = [ elem X1 Y1 X2 Y2 ]    ... incidence matrix
%%%         Ke    = [ nedof x nedof ]       ... element tangent Ke
%%%         Re    = [ fx_1 fy_1 fx_2 fy_2 ] ... element residual Re
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%% output: K     = [ ndof x ndof ]         ... global tangent K
%%%         R     = [ ndof x 1 ]            ... global residual R
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[nie,n]=size(edof);
I(edof(:,1),:)=Ie(:);
t=edof(:,2:n);
for i = 1:nie
   K(t(i,:),t(i,:)) = K(t(i,:),t(i,:))+Ke;
   R(t(i,:))        =R(t(i,:))       +Re;
end
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```



@ integration point level

- constitutive equations - given $\boldsymbol{F} = \nabla\boldsymbol{\varphi}$ calculate $\boldsymbol{P}$

- update density for current stress state
  from $\rho_{0\,n}$ and $\mathrm{D}_t\rho_0 = \left[\frac{\rho_0}{\rho_0^*}\right]^{-m}\psi_0(\boldsymbol{F}) - \psi_0^*$ calculate $\rho_{0\,n+1}$
- calculate first piola kirchhoff stress of solid material
  $\boldsymbol{P}^{\mathrm{neo}}(\boldsymbol{F}) = \mu_0\,\boldsymbol{F} + [\,\lambda_0 \ln(\det(\boldsymbol{F})) - \mu_0\,]\boldsymbol{F}^{-\mathrm{t}}$
- calculate first piola kirchhoff stress of porous material
  $\boldsymbol{P}\,(\boldsymbol{F}) = \left[\frac{\rho_0}{\rho_0^*}\right]^n \boldsymbol{P}^{\mathrm{neo}}$

stress for righthand side vector

@ integration point level

- constitutive equations - given $\boldsymbol{F}$ calculate $\boldsymbol{P}$

check in matlab!

$$\boldsymbol{P}\,(\boldsymbol{F}) = \left[\frac{\rho_0}{\rho_0^*}\right]^n \mu_0\,\boldsymbol{F} + [\,\lambda_0 \ln(\det(\boldsymbol{F})) - \mu_0\,]\boldsymbol{F}^{-\mathrm{t}}$$

- stress calculation @ integration point level

stress for righthand side vector

@ integration point level

- constitutive equations - given $\boldsymbol{F}$ calculate $\mathrm{D}_F\boldsymbol{P}$

$$\mathrm{D}_F\boldsymbol{P} = \partial_F\boldsymbol{P} - \partial_{\rho_0}\boldsymbol{P}\,[\partial_{\rho_0}\mathcal{R}_0]^{1-}\,\partial_F\mathcal{R}_0$$

with

$$\partial_F\boldsymbol{P} = \left[\frac{\rho_0}{\rho_0^*}\right]^n \left[\mu\boldsymbol{I}\overline{\otimes}\boldsymbol{I} + \lambda\boldsymbol{F}^{-t}\otimes\boldsymbol{F}^{-t} - [\lambda\ln J - \mu]\,\boldsymbol{F}^{-t}\underline{\otimes}\boldsymbol{F}^{-1}\right]$$

$$\partial_{\rho_0}\boldsymbol{P} = \frac{1}{\rho_0}\,n\,\boldsymbol{P}$$

$$\partial_{\rho_0}\mathcal{R}_0 = \left[\frac{\rho_0}{\rho_0^*}\right]^{-m}\frac{1}{\rho_0}\,[n-m]\,\psi_0$$

depending on time discretization

$$\partial_F\mathcal{R}_0 = \left[\frac{\rho_0}{\rho_0^*}\right]^{-m}\boldsymbol{P}$$

tangent for iteration matrix

@ integration point level

- tangent operator / constitutive moduli

check in matlab!

$$\mathbf{A} = \mathrm{D}_F\boldsymbol{P} = \partial_F\boldsymbol{P} - \partial_{\rho_0}\boldsymbol{P}\,[\partial_{\rho_0}\mathcal{R}_0]^{1-}\,\partial_F\mathcal{R}_0$$

- linearization of stress wrt deformation gradient

tangent for iteration matrix

cnst_den.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [A,P,var]=cnst_den(F,var,mat)
%%% determine tangent, stress and internal variable %%%%%%%%%%%%%%%%%%%
emod = mat(1);    nue  = mat(2);    rho0 = mat(3);    psi0 = mat(4);
expm = mat(5);    expn = mat(6);    dt   = mat(7);
xmu  = emod/2.0/(1.0+nue);  xlm = emod * nue /(1.0+nue)/(1.0-2.0*nue);
F_inv = inv(F);      J = det(F);    delta = [1 0; 0 1];
%%% update internal variable %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[var,facs,fact]=upd_dens(F,var,mat);
%%% first piola kirchhoff stress %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
P =  facs * (xmu * F + (xlm * log(J) - xmu) * F_inv');
%%% tangent %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:2
for j=1:2
for k=1:2
for l=1:2
    A(i,j,k,l) =  xlm                 * F_inv(j,i)*F_inv(l,k) ...
                - (xlm * log(J) - xmu) * F_inv(l,i)*F_inv(j,k) ...
                +                 xmu  * delta(i,k)*delta(j,l);
    A(i,j,k,l) = facs * A(i,j,k,l)   +  fact  * P(i,j)*P(k,l);
end, end, end, end
%%% determine tangent, stress and internal variable %%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```


recipe for temporal discretization

explicit euler forward

- evolution of density

$$\mathrm{D}_t\rho_0 = \tfrac{1}{\Delta t}[\rho_{0\,n+1} - \rho_{0\,n}]$$

finite difference approximation

$$\mathrm{D}_t\rho_0 = \left[\frac{\rho_{0\,n}}{\rho_0^*}\right]^{-m}\psi_0(\boldsymbol{F}) - \psi_0^*$$

euler forward

- direct update of growth multiplier

$$\rho_{0\,n+1} = \rho_{0\,n} + \left[\left[\frac{\rho_{0\,n}}{\rho_0^*}\right]^{-m}\psi_0\,(\boldsymbol{F}) - \psi_0^*\right]\Delta t$$

conditionally stable - limited time step size

recipe for temporal discretization

implicit euler backward

- evolution of density

$$\mathrm{D}_t\rho_0 = \tfrac{1}{\Delta t}[\rho_{0\,n+1} - \rho_{0\,n}]$$ finite difference approximation

$$\mathrm{D}_t\rho_0 = \left[\tfrac{\rho_{0\,n+1}}{\rho_0^*}\right]^{-m}\psi_0(\boldsymbol{F}) - \psi_0^*$$ euler backward

- discrete residual

$$\mathrm{R}^{\rho}_{n+1} = \tfrac{1}{\Delta t}[\rho_{0\,n+1} - \rho_{0\,n}] - \left[\tfrac{\rho_{0\,n+1}}{\rho_0^*}\right]^{-m}\psi_0(\boldsymbol{F}) - \psi_0^* \doteq 0$$

- local newton iteration

$$\mathrm{R}^{\rho k+1}_{n+1} = \mathrm{R}^{\rho k}_{n+1} + \mathrm{dR}^{\rho} \doteq 0 \qquad \mathrm{dR}^{\rho} = \frac{\mathrm{dR}^{\rho}}{\mathrm{d}\rho_0}\cdot \mathrm{d}\rho_0$$

$$\rho_{0\,n+1} \leftarrow \rho_{0\,n+1} + \mathrm{d}\rho_0 \quad \mathrm{d}\rho_0 = \left[\frac{\mathrm{dR}^{\rho}}{\mathrm{d}\rho_0}\right]^{-1} \mathrm{R}^{\rho k}_{n+1}$$ iterative update

unconditionally stable - larger time steps

@ integration point level

- discrete residual of density update

check in matlab!

$$\mathrm{R}^{\rho}_{n+1} = \tfrac{1}{\Delta t}[\rho_{0\,n+1} - \rho_{0\,n}] - \left[\tfrac{\rho_{0\,n+1}}{\rho_0^*}\right]^{-m}\psi_0(\boldsymbol{F}) - \psi_0^* \doteq 0$$

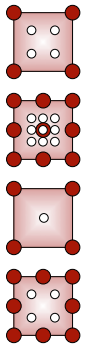
- residual of biological equilibrium / balance of mass

local newton iteration

upd_dens.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [var,facs,fact]=upd_dens(F,var,mat)
%%% update internal variable density %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
tol = 1e-8;      var =  0.0;
xmu = emod / 2.0 / (1.0+nue);     xlm = emod * nue /(1.0+nue)/(1.0-2.0*nue);
J  = det(F);C  = F'*F;            I1  = trace(C);
psi0_neo = xlm/2 * log(J)^2  +  xmu/2 *(I1 - 2 - 2*log(J));
rho_k0 = (1+var)*rho0;     rho_k1 = (1+var)*rho0;     iter = 0;    res  = 1;
%%% local newton-raphson iteration %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
while abs(res) > tol
   iter=iter+1;
   res =((rho_k1/rho0)^(expn-expm)*psi0_neo-psi0)*dt-rho_k1+rho_k0;
   dres= (expn-expm)*(rho_k1/rho0)^(expn-expm)*psi0_neo*dt/rho_k1-1;
   drho=- res/dres;    rho_k1 = rho_k1+drho;
end
%%% local newton-raphson iteration %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
rho  = rho_k1;   var = rho / rho0 - 1;
facs = (rho/rho0)^expn;
facr = 1/dt - (expn-expm) * (rho/rho0)^(expn-expm) / rho *psi0_neo;
fact =         expn / rho * (rho/rho0)^(    -expm) / facr;
%%% update internal variable density %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

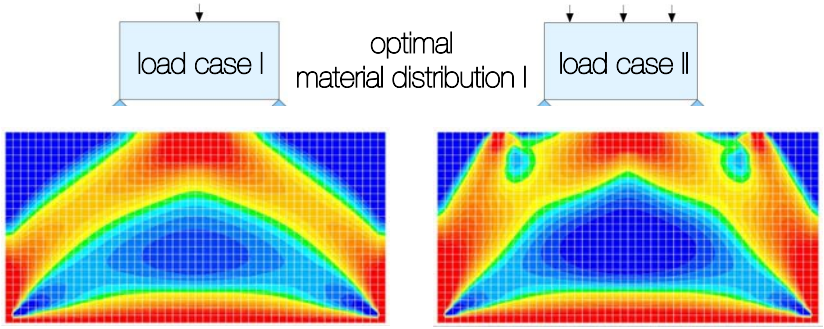

integration point based solution of balance of mass



loop over all time steps	nlin_fem
global newton iteration	nlin_fem
loop over all elements	element1
loop over all quadrature points	cnst_law
local newton iteration $\rho_{0\,n+1}$	upd_dens
determine element residual & tangent	cnst_law
determine global residual and tangent	element1
determine φ_{n+1}	nlin_fem
determine state of biological equilibrium	nlin_fem

staggered solution of density and displacements

form follows function - bioinspired design



find the lightest structure to support a given set of loads

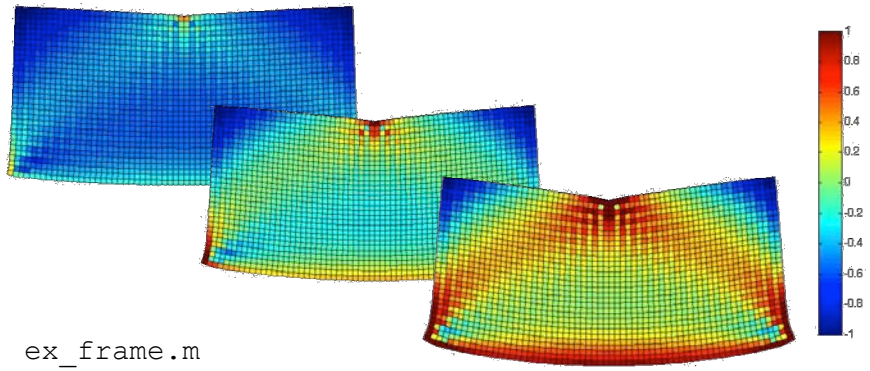
example - topology optimization

ex_frame.m

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function [q0,edof,bc,F_ext,mat,nel,node,ndof,nip] = ex_frame
%%% input data for frame example %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
emod = 1000;  nue  = 0.3;  rho0 = 1.0;  psi0 = 1.0;
expm = 3.0;   expn = 2.0;  dt   = 1.0;
mat = [emod,nue,rho0,psi0,expm,expn,dt];

xbox(1) = 0.0;  xbox(2) = 2.0;   nx = 8;
ybox(1) = 0.0;  ybox(2) = 1.0;   ny = 4;
[q0,edof] = mesh_sqr(xbox,ybox,nx,ny);
[nel, sizen] = size(edof);[ndof,sizen] = size(q0);
node        = ndof/2;     nip = 4;

%%% dirichlet boundary conditions
bc(1,1) = 2*(ny+1)*(0     ) +2;    bc(1,2) = 0;
bc(2,1) = 2*(ny+1)*(nx    ) +2;    bc(2,2) = 0;
bc(3,1) = 2*(ny+1)*(nx/2+0) +1;    bc(3,2) = 0;
bc(4,1) = 2*(ny+1)*(nx/2+1)   ;    bc(4,2) = -ybox(2)/50;

%%% neumann boundary conditions
F_ext =  zeros(ndof,1);
%%% input data for frame example %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
```

example - topology optimization

form follows function - bioinspired design



find the lightest structure to support a given set of loads

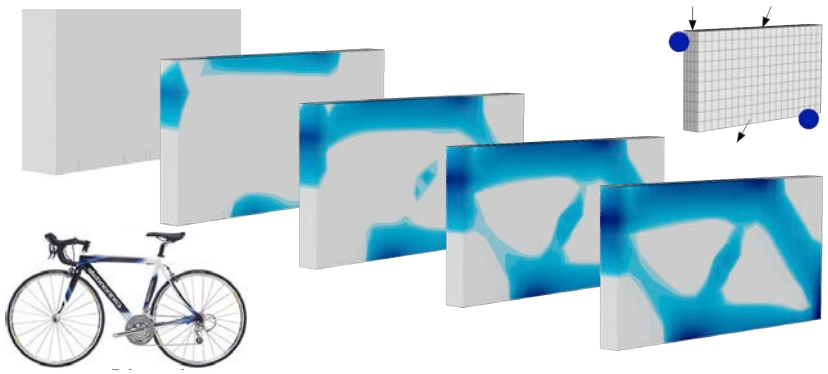
example - topology optimization

form follows function

bicycle frames 1817-2005

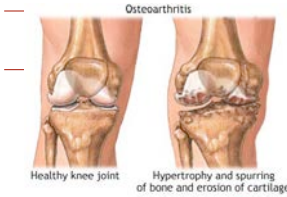
example - topology optimization

form follows function



design of bicycle frame

example - topology optimization



osteoarthritis

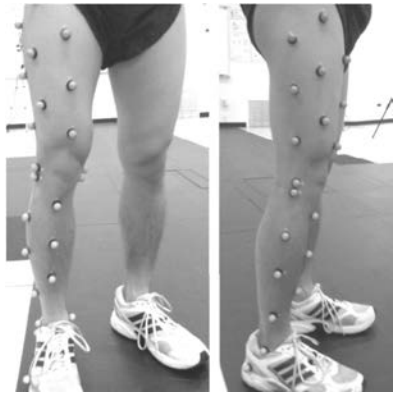
osteoarthritis, also known as degenerative joint disease or osteoarthrosis, is a group of mechanical abnormalities involving degradation of joints, including articular cartilage and subchondral bone. it affects about 27 million people in the united states alone. symptoms may include joint pain, tenderness, stiffness, locking, and sometimes an effusion. a variety of causes, hereditary, developmental, metabolic, and mechanical, may initiate processes leading to loss of cartilage. when bone surfaces become less well protected by cartilage, bone may be exposed and damaged. treatment generally involves a combination of exercise, lifestyle modification, or, in severe cases,surgical joint replacement.

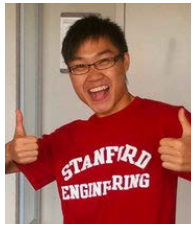
WIKIPEDIA
The Free Encyclopedia

example - osteoarthritis

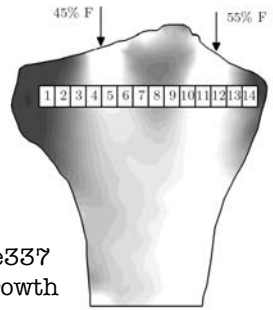
Computational modeling of bone density profiles in response to gait: A subject-specific approach

Henry Pang[1], Abhishek P. Shiwalkar[1], Chris M. Madormo[1], Rebecca E. Taylor[1], Thomas P. Andriacchi[1,2], Ellen Kuhl[1,3,4]





class project me337
mechanics of growth

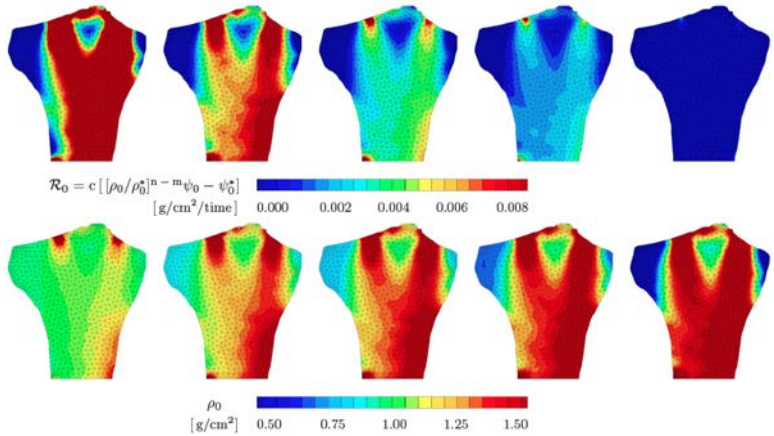


trial	I	II	III	mean
max knee force [N]	552	549	579	560
max knee force [% BW]	94.9	94.4	99.4	96.2

pang, shiwalkar, madormo, taylor, andriacchi, kuhl [2012]

example – henry's knee

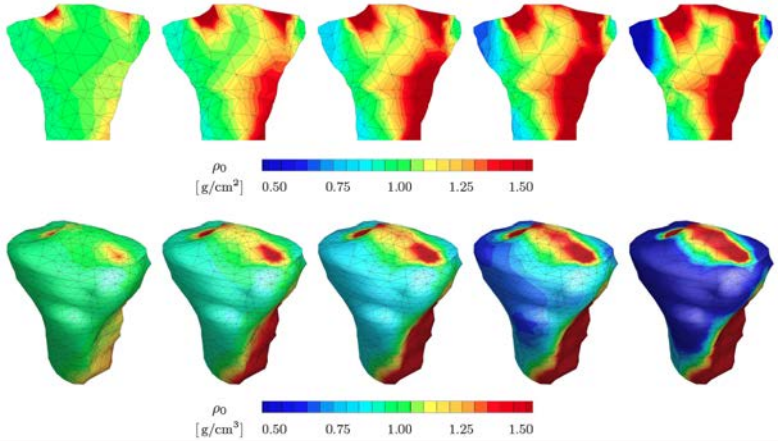
how does henry's bone grow?



pang, shiwalkar, madormo, taylor, andriacchi, kuhl [2012]

example – henry's knee

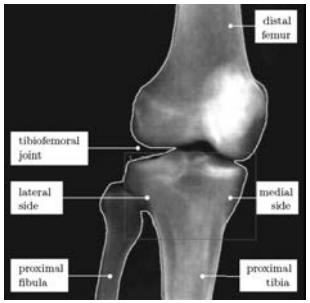
how does henry's bone grow?



pang, shiwalkar, madormo, taylor, andriacchi, kuhl [2012]

example – henry's knee

how predictive is the simulation?

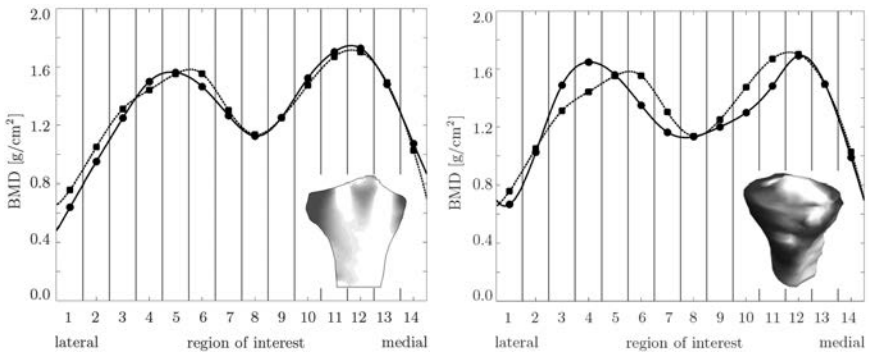


	DEXA [g/cm²]	2d FEM [g/cm²]	2d error [%]	3d FEM [g/cm³]	3d error [%]
BMD_{lat}	1.268	1.233	2.73	1.384	9.33
BMD_{med}	1.621	1.637	0.99	1.556	3.99
	DEXA [–]	2d FEM [–]	2d error [%]	3d FEM [–]	3d error [%]
M:L-ratio	1.279	1.327	3.83	1.122	12.18

pang, shiwalkar, madormo, taylor, andriacchi, kuhl [2012]

example – henry's knee

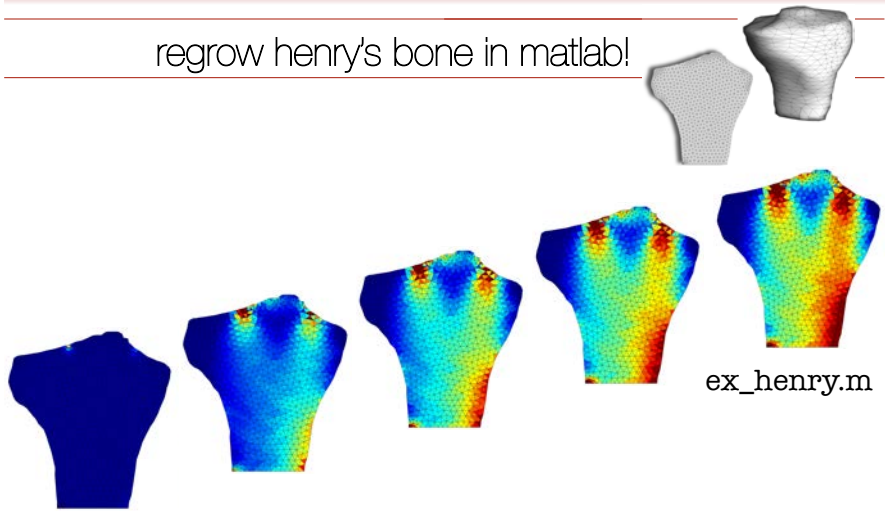
how predictive is the simulation?



pang, shiwalkar, madormo, taylor, andriacchi, kuhl [2012]

example – henry's knee

regrow henry's bone in matlab!



pang, shiwalkar, madormo, taylor, andriacchi, kuhl [2012]

example - henry 's knee

anisotropy of bone mineral density

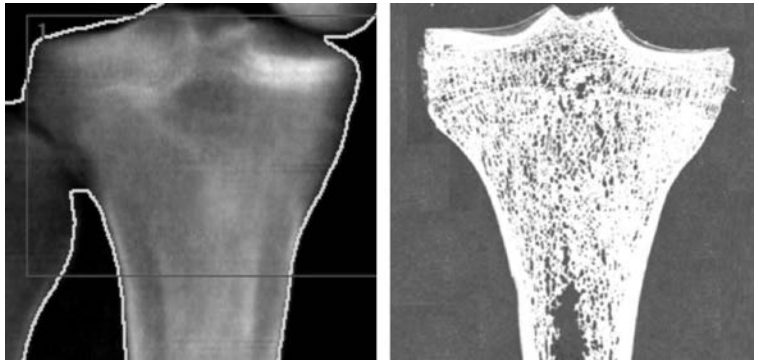


figure 1. healthy proxima tibia. isometric desity distribution, left, visualized through DEXA scan, which displays the characteristic heterogeneous density distriubtion with a dense region in the medial plateau, a lower density region in the lateral plateau, and the lowest density in the central region, pang et al. [2012]. anisotropic density distriubtion, right, visualized through a thin section. Trabeculae are aligned with axis of maximum principal loading, wolff [1870].

anisotropy of bone mineral density

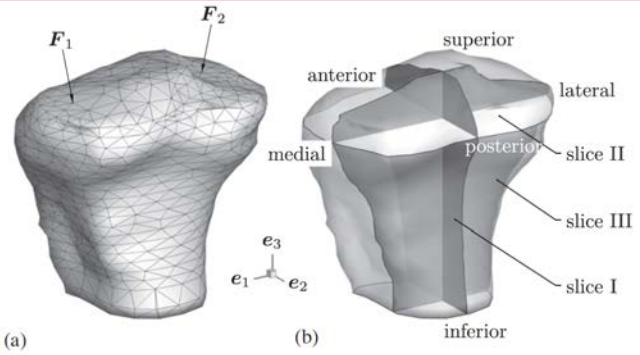
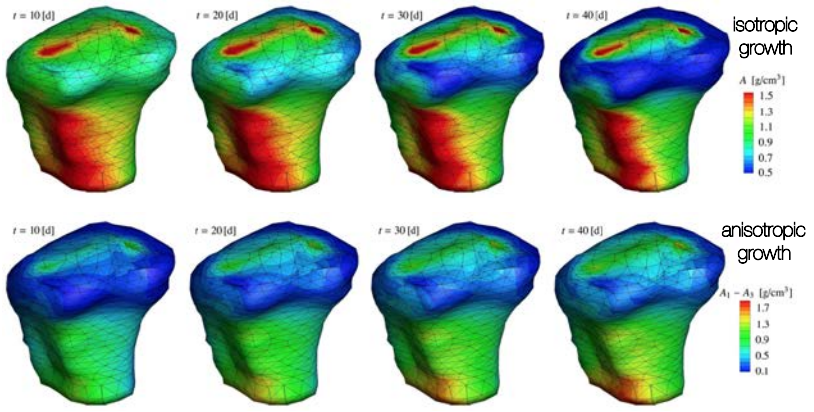


Fig. 6. Geometry of the upper part of the proximal tibia. (a) Finite element mesh with applied concentrated forces F_1, F_2 and (b) location of slices I (sagittal slice), II (axial slice) and III (coronal slice) used in Figs. 9–13.

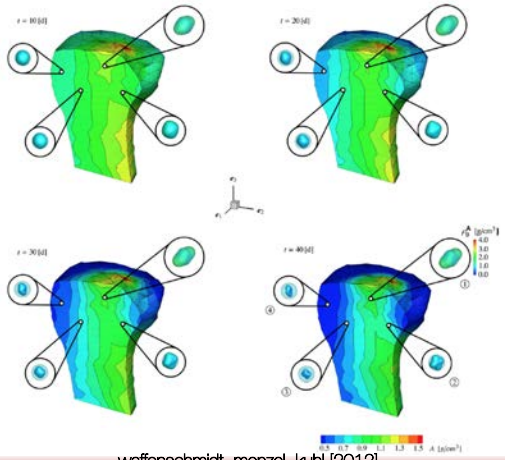
waffenschmidt, menzel, kuhl [2012]

anisotropy of bone mineral density



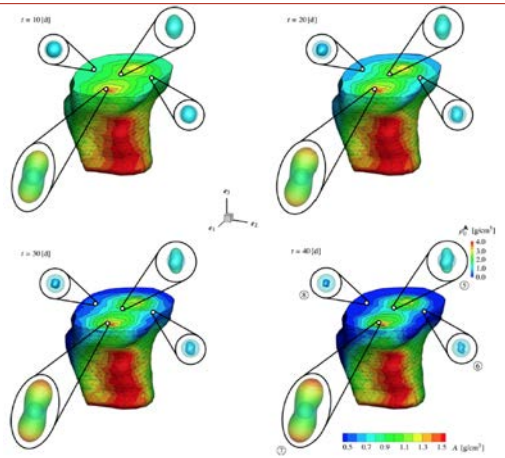
waffenschmidt, menzel, kuhl [2012]

anisotropy of bone mineral density



waffenschmidt, menzel, kuhl [2012]

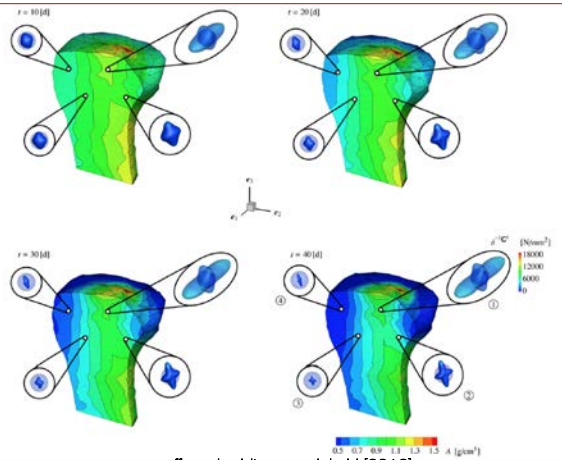
anisotropy of bone mineral density



waffenschmidt, menzel, kuhl [2012]

example - osteoarthritis

anisotropy of bone mineral density



waffenschmidt, menzel, kuhl [2012]

example - osteoarthritis