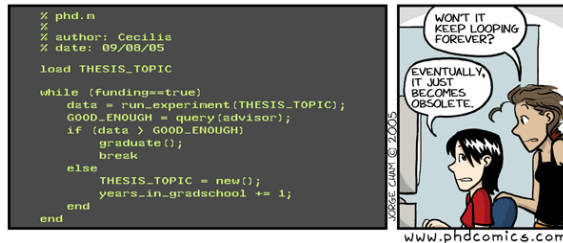


14 – finite element method – density growth - theory



14 – density growth

1

day	date	topic
tue	jan 10	motivation - everything grows!
thu	jan 12	basics maths - notation and tensors
tue	jan 17	basic kinematics - large deformation and growth
thu	jan 19	kinematics - growing hearts
tue	jan 24	guest lecture - growing skin
thu	jan 26	guest lecture - growing leaflets
tue	jan 31	basic balance equations - closed and open systems
thu	feb 02	basic constitutive equations - growing tumors
tue	feb 07	volume growth - finite elements for growth
thu	feb 09	volume growth - growing arteries
tue	feb 14	volume growth - growing skin
thu	feb 16	volume growth - growing hearts
tue	feb 21	basic constitutive equations - growing bones
thu	feb 23	density growth - finite elements for growth
tue	feb 28	density growth - growing bones
thu	mar 01	everything grows! - midterm summary
tue	mar 06	midterm
thu	mar 08	remodeling - remodeling arteries and tendons
tue	mar 13	class project - discussion, presentation, evaluation
thu	mar 15	class project - discussion, presentation, evaluation
thu	mar 15	written part of final projects due

where are we???

homework III - revise your final project

due 03/01/12, 09:30am, 300-020

Late homework can be dropped off in a box in front of Durand 217. Please mark clearly 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 $F^g = \sqrt{\vartheta} I + [1 - \sqrt{\vartheta}] n_0 \otimes n_0$

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 Mador-mo, 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.

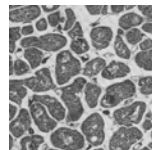
homework #03

balance equations for open systems - mass

$$D_t \rho_0 = \text{Div}(\mathbf{R}) + \mathcal{R}_0$$

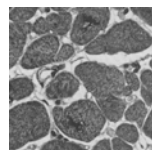
mass flux \mathbf{R}

- cell movement (migration)



mass source \mathcal{R}_0

- cell growth (proliferation)
- cell division (hyperplasia)
- cell enlargement (hypertrophy)



biological equilibrium

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

growing bone as open system

problem 4 - wikipedia websites on growth

Create or edit a wikipedia website on growth. To create a site, you can think of any growth example. To edit a site, e.g., go to http://en.wikipedia.org/wiki/Wolff's_law or http://en.wikipedia.org/wiki/Tissue_expansion. Add at least one paragraph of text with at least two references. Ideally, even add a figure. Print your submission and hand it in with your homework.

problem 5 - revise your final project

homework #03

balance equations for open systems - momentum

- volume specific version

$$D_t(\rho_0 \mathbf{v}) = \text{Div}(\mathbf{P} + \mathbf{v} \otimes \mathbf{R}) + [\mathbf{b}_0 + \mathbf{v} \mathcal{R}_0 - \nabla_X \mathbf{v} \cdot \mathbf{R}]$$

- subtraction of weighted balance of mass

$$\mathbf{v} D_t \rho_0 = \text{Div}(\mathbf{v} \otimes \mathbf{R}) + \mathbf{v} \mathcal{R}_0 - \nabla_X \mathbf{v} \cdot \mathbf{R}$$

- mass specific version

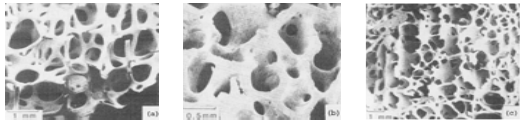
$$\rho_0 D_t \mathbf{v} = \text{Div}(\mathbf{P}) + \mathbf{b}_0$$

mechanical equilibrium

growing bone as open system

constitutive equations for open systems

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



coupling of growth and deformation

gibson & ashby [1999]

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]

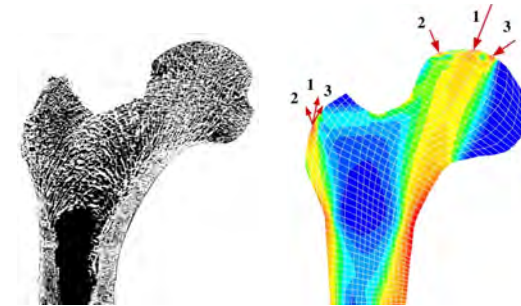
sequential solution - element based



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 11

finite elements for open systems



- **dense system of compressive trabeculae** 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

recipe for finite element modeling



from continuous problem...

$$\begin{aligned} D_t \rho_0 &= \text{Div}(\mathbf{R}) + \mathcal{R}_0 \\ \rho_0 D_t \mathbf{v} &= \text{Div}(\mathbf{P}) + \mathbf{b}_0 \end{aligned}$$

- 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

finite elements - integration point based 12

key transformation - from strong form to weak form (1d)

- strong / differential form

$$\sum f = f^{\text{int}} + f^{\text{ext}} \doteq 0 \quad f^{\text{int}} = P'(\varphi)$$

- strong form / residual format

$$R(\varphi) = P'(\varphi) + f^{\text{ext}} \doteq 0$$

- weak / integral form - nonsymmetric $\forall \delta\varphi$

$$G(\delta\varphi; \varphi) = \int \delta\varphi \cdot [P'(\varphi) + f^{\text{ext}}] dx \doteq 0$$

- integration by parts

$$\int \delta\varphi \cdot P' dx = \int [\delta\varphi \cdot P]' dx - \int \delta\varphi' \cdot P dx$$
- integral theorem & neumann bc's

$$\int [\delta\varphi \cdot P]' dx = \delta\varphi \cdot P|_{x=0}^{x=l}$$
- weak form / integral form - symmetric $\forall \delta\varphi$

$$\int \delta\varphi' \cdot P dx - \delta\varphi \cdot P|_{x=0}^{x=l} - \int \delta\varphi \cdot f^{\text{ext}} dx \doteq 0$$

finite elements – integration point based 13

residual equation...

- strong / differential form

$$\mathbf{R}^\varphi = \rho_0 D_t \mathbf{v} - \text{Div}(\mathbf{P}) - \mathbf{b}_0 = \mathbf{0} \quad \text{in } \mathcal{B}_0$$

- dirichlet / essential boundary conditions (displacements)

$$\varphi - \bar{\varphi} = \mathbf{0} \quad \text{on } \partial\mathcal{B}_0^\varphi \quad \text{with } \partial\mathcal{B}_0^\varphi \cup \partial\mathcal{B}_0^{T^\varphi} = \partial\mathcal{B}_0$$

- neumann / natural boundary conditions (tractions)

$$\mathbf{P} \cdot \mathbf{N} - \bar{\mathbf{T}}^\varphi = \mathbf{0} \quad \text{on } \partial\mathcal{B}_0^{T^\varphi} \quad \text{and } \partial\mathcal{B}_0^\varphi \cap \partial\mathcal{B}_0^{T^\varphi} = \emptyset$$

... and boundary conditions

finite elements – integration point based 15

from equilibrium equation...

- start with nonlinear mechanical equilibrium equation

$$\rho_0 D_t \mathbf{v} \approx \mathbf{0} \text{ quasi-static} \quad \rho_0 D_t \mathbf{v} \approx \mathbf{0} \text{ no gravity}$$

$$\rho_0 D_t \mathbf{v} = \text{Div}(\mathbf{P}) + \mathbf{b}_0$$



- cast it into its residual format

$$\mathbf{R}^\varphi(\varphi) = \mathbf{0} \quad \text{in } \mathcal{B}_0$$

- with residual

$$\mathbf{R}^\varphi = \rho_0 D_t \mathbf{v} - \text{Div}(\mathbf{P}) - \mathbf{b}_0$$

... to residual format

finite elements – integration point based 14

from strong form...

- strong / differential form

$$\mathbf{R}^\varphi = \rho_0 D_t \mathbf{v} - \text{Div}(\mathbf{P}) - \mathbf{b}_0 = \mathbf{0} \quad \text{in } \mathcal{B}_0$$

- multiplication with test function & integration

$$G^\varphi(\delta\varphi; \varphi) = \int_{\mathcal{B}_0} \delta\varphi \cdot \mathbf{R}^\varphi dV = 0 \quad \forall \delta\varphi \text{ in } \mathcal{H}_1^0(\mathcal{B}_0)$$

- weak form / nonsymmetric $\begin{matrix} \text{no} \\ \text{derivative} \end{matrix}$ $\begin{matrix} \text{second} \\ \text{derivative} \end{matrix}$

$$G^\varphi = \int_{\mathcal{B}_0} \delta\varphi \cdot \rho_0 D_t \mathbf{v} dV - \int_{\mathcal{B}_0} \delta\varphi \cdot \text{Div}(\mathbf{P}) dV - \int_{\mathcal{B}_0} \delta\varphi \cdot \mathbf{b}_0 dV$$

... to nonsymmetric weak form

finite elements – integration point based 16



from non-symmetric weak form...

- integration by parts

$$\int_{B_0} \delta \varphi \cdot \text{Div}(\mathbf{P}) dV = \int_{B_0} \text{Div}(\delta \varphi \cdot \mathbf{P}) dV - \int_{B_0} \nabla \delta \varphi : \mathbf{P} dV$$

- gauss theorem & boundary conditions

$$\int_{B_0} \text{Div}(\delta \varphi \cdot \mathbf{P}) dV = \int_{\partial B_0^{T\varphi}} \delta \varphi \cdot \mathbf{P} \cdot \mathbf{N} dA = \int_{\partial B_0^{T\varphi}} \delta \varphi \cdot \bar{\mathbf{T}}^{\varphi} dA$$

- weak form / symmetric ^{first derivative} ^{first derivative}

$$\mathbf{G}^{\varphi} = \int_{B_0} \delta \varphi \cdot \rho_0 \mathbf{D}_t \mathbf{v} dV + \int_{B_0} \nabla \delta \varphi : \mathbf{P} dV - \int_{\partial B_0^{T\varphi}} \delta \varphi \cdot \bar{\mathbf{T}}^{\varphi} dA - \int_{B_0} \delta \varphi \cdot \mathbf{b}_0 dV$$

... to symmetric weak form

finite elements – integration point based 17

from discrete weak form...

- discrete weak form

$$\mathbf{G}^{\varphi} = \delta \varphi_J \cdot \mathbf{R}_J^{\varphi}(\varphi_{n+1}^h) = 0 \quad \forall \delta \varphi_J$$

- discrete residual format

$$\mathbf{R}_J^{\varphi}(\varphi_{n+1}^h) = \mathbf{0} \quad \forall J = 1, \dots, n_{np}$$

- discrete residual

$$\mathbf{R}_J^{\varphi} = \mathbf{A}_{e=1}^{n_{el}} \int_{B_0^e} N_{\varphi}^J \mathbf{D}_t \mathbf{v}_{n+1} dV + \int_{B_0^e} \nabla N_{\varphi}^J \cdot \mathbf{P}_{n+1} dV - \int_{\partial B_0^e} N_{\varphi}^J \bar{\mathbf{T}}_{n+1}^{\varphi} dA - \int_{B_0^e} N_{\varphi}^J \mathbf{b}_{0n+1} dV$$

... to discrete residual

finite elements - integration point based 19

spatial discretization

- discretization

$$B_0 = \bigcup_{e=1}^{n_{el}} B_0^e$$

- interpolation of test functions

$$\delta \varphi^h|_{B_0^e} = \sum_{j=1}^{n_{en}} N_{\varphi}^j \delta \varphi_j \in \mathcal{H}_1^0(B_0) \quad \nabla \delta \varphi^h|_{B_0^e} = \sum_{j=1}^{n_{en}} \delta \varphi_j \otimes \nabla N_{\varphi}^j$$

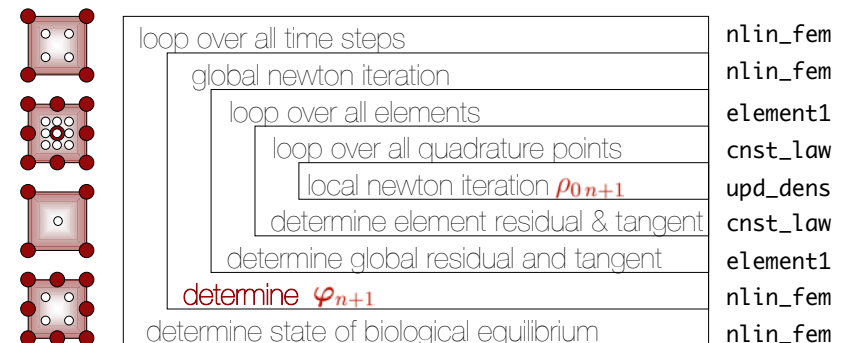
- interpolation of trial functions

$$\varphi^h|_{B_0^e} = \sum_{l=1}^{n_{en}} N_{\varphi}^l \varphi_l \in \mathcal{H}_1(B_0) \quad \nabla \varphi^h|_{B_0^e} = \sum_{l=1}^{n_{en}} \varphi_l \otimes \nabla N_{\varphi}^l$$

... to discrete weak form

finite elements - integration point based 18

integration point based solution of balance of mass



staggered solution of density and displacements

finite elements - integration point based

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 %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

finite elements - integration point based 21

from discrete residual ...

- linearization / newton raphson scheme

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

- incremental residual

$$d\mathbf{R}_J^{\varphi} = \sum_{L=1}^{n_{el}} \mathbf{K}_{JL}^{\varphi\varphi} \cdot d\varphi_L \quad \mathbf{K}_{JL}^{\varphi\varphi} = \frac{d\mathbf{R}_J^{\varphi}}{d\varphi_L}$$

- system of equations

$$\mathbf{K}_{JL}^{\varphi\varphi} d\varphi_L = -\mathbf{R}_{Jn+1}^{\varphi k}$$

- incremental iterative update

$$\Delta\varphi_L = \Delta\varphi_L + d\varphi_L \quad \forall L = 1, \dots, n_{np}$$

... to linearized residual

finite elements - integration point based 23

integration point based



- discrete residual

check in matlab!

$$\mathbf{R}_J^{\varphi} = \mathbf{A}_{e=1}^{n_{el}} \int_{B_0^e} \nabla N_{\varphi}^j \cdot \mathbf{P}_{n+1} dV$$

- residual of mechanical equilibrium/balance of momentum

righthand side vector for global system of equations

finite elements - integration point based 22

linearized residual



- stiffness matrix / iteration matrix

$$\mathbf{K}_{JL}^{\varphi\varphi} = \frac{\partial \mathbf{R}_J^{\varphi}}{\partial \varphi_L} = \mathbf{A}_{e=1}^{n_{el}} \int_{B_0^e} N_{\varphi}^j \rho D_{\varphi} (D_t \mathbf{v}) N_{\varphi}^l dV \\ + \int_{B_0^e} \nabla N_{\varphi}^j \cdot \mathbf{D}_F \mathbf{P} \cdot \nabla N_{\varphi}^l dV$$

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

finite elements - integration point based 24

linearized residual

- stiffness matrix / iteration matrix



check in matlab!

$$\mathbf{K}_{JL}^{\varphi\varphi} = \frac{\partial \mathbf{R}_J^{\varphi}}{\partial \varphi_L} = \mathbf{A}_{e=1}^{n_{el}} \int_{B_0^e} \nabla N_{\varphi}^j \cdot \mathbf{D}_F \mathbf{P} \cdot \nabla N_{\varphi}^l dV$$

- linearization of residual wrt nodal dofs

iteration matrix for global system of equations

finite elements - integration point based 25

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 %%%
    %%%%%%%%%%%

```



finite elements - integration point based 27

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 %%%

```



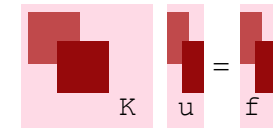
finite elements - integration point based 26

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   = [ ndof x ndof ] ... 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
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```



finite elements - integration point based 28


@ integration point level

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

stress for righthand side vector

finite elements - integration point based 29

@ integration point level

- constitutive equations - given \mathbf{F} calculate $\mathbf{D}_F \mathbf{P}$ 
- $\mathbf{D}_F \mathbf{P} = \partial_F \mathbf{P} - \partial_{\rho_0} \mathbf{P} [\partial_{\rho_0} \mathcal{R}_0]^{1-} \partial_F \mathcal{R}_0$
with
- $\partial_F \mathbf{P} = \left[\frac{\rho_0}{\rho_0^*} \right]^n [\mu \mathbf{I} \otimes \mathbf{I} + \lambda \mathbf{F}^{-t} \otimes \mathbf{F}^{-t} - [\lambda \ln J - \mu] \mathbf{F}^{-t} \underline{\otimes} \mathbf{F}^{-1}]$
- $\partial_{\rho_0} \mathbf{P} = \frac{1}{\rho_0} n \mathbf{P}$
- $\partial_{\rho_0} \mathcal{R}_0 = \left[\frac{\rho_0}{\rho_0^*} \right]^{-m} \frac{1}{\rho_0} [n - m] \psi_0$ depending on time discretization
- $\partial_F \mathcal{R}_0 = \left[\frac{\rho_0}{\rho_0^*} \right]^{-m} \mathbf{P}$

tangent for iteration matrix

finite elements - integration point based 31

@ integration point level

- constitutive equations - given \mathbf{F} calculate \mathbf{P} 

check in matlab!

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

- stress calculation @ integration point level

stress for righthand side vector

finite elements - integration point based 30

@ integration point level

- tangent operator / constitutive moduli 

check in matlab!

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

- linearization of stress wrt deformation gradient

tangent for iteration matrix

finite elements - integration point based 32

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 /(1.0-2.0*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 %%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

finite elements - integration point based



recipe for temporal discretization

explicit euler forward

- evolution of density

$$D_t \rho_0 = \frac{1}{\Delta t} [\rho_{0n+1} - \rho_{0n}] \quad \text{finite difference approximation}$$

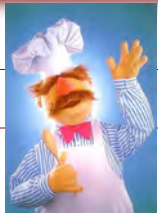
$$D_t \rho_0 = \left[\frac{\rho_{0n}}{\rho_0^*} \right]^{-m} \psi_0(\mathbf{F}) - \psi_0^* \quad \text{euler forward}$$

- direct update of growth multiplier

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

conditionally stable - limited time step size

finite elements - integration point based 34



recipe for temporal discretization

implicit euler backward

- evolution of density

$$D_t \rho_0 = \frac{1}{\Delta t} [\rho_{0n+1} - \rho_{0n}] \quad \text{finite difference approximation}$$

$$D_t \rho_0 = \left[\frac{\rho_{0n+1}}{\rho_0^*} \right]^{-m} \psi_0(\mathbf{F}) - \psi_0^* \quad \text{euler backward}$$

- discrete residual

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

- local newton iteration

$$R_{n+1}^{\rho k+1} = R_{n+1}^{\rho k} + dR^\rho \doteq 0$$

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

unconditionally stable - larger time steps

finite elements - integration point based 35

@ integration point level



- discrete residual of density update

check in matlab!

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

- residual of biological equilibrium / balance of mass

local newton iteration

finite elements - integration point based 36

upd_dens.m

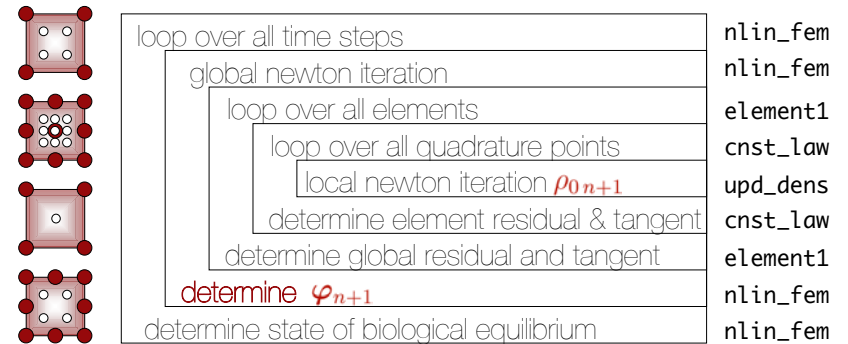
```

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
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)^(expn-expm) / facr;
%% update internal variable density %%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

```

finite elements - integration point based

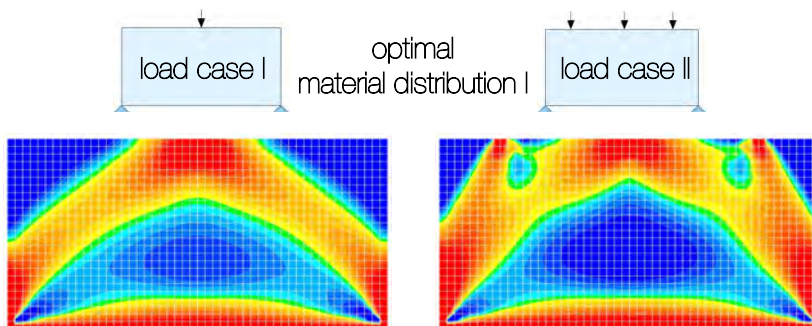
integration point based solution of balance of mass



staggered solution of density and displacements

finite elements - integration point based

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, size] = size(edof); [ndof,size] = size(q0);
node = ndof/2; nip = 4;

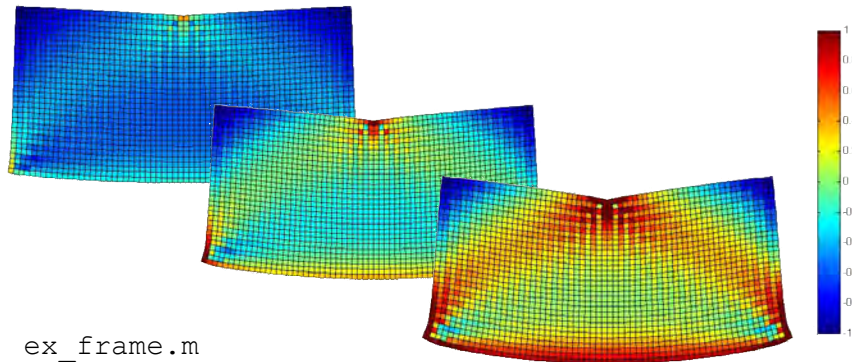
%% dirichlet boundary conditions
bc(1,1) = 2*(ny+1)*(0) +2; bc(1,2) = 0;
bc(2,1) = 2*(ny+1)*(nx) +2; bc(2,2) = 0;
bc(3,1) = 2*(ny+1)*(nx/2+0) +1; bc(3,2) = 0;
bc(4,1) = 2*(ny+1)*(nx/2+1) ; bc(4,2) = -ybox(2)/50;

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

```

example - topology optimization

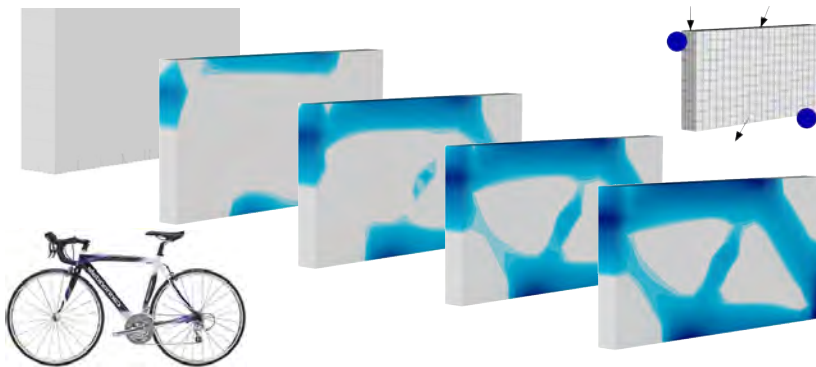
form follows function - bioinspired design



find the lightest structure to support a given set of loads

example - topology optimization

form follows function



design of bicycle frame

example - topology optimization

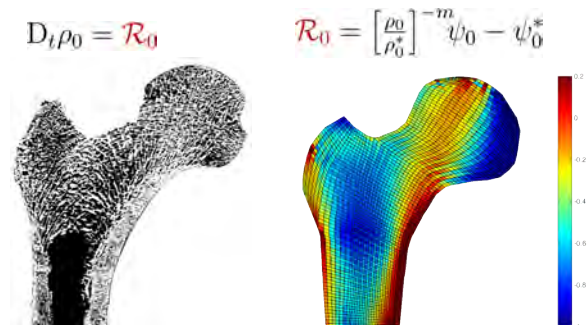
form follows function



bicycle frames 1817-2005

example - topology optimization

functional adaptation of proxima femur



the density develops such that the tissue can just support the given mechanical load

example - growing bone

femoral neck deformity

the femoral neck normally forms an angle of 120-135 degrees with the shaft of the bone. this acts as the lever in easing the action of the muscles around the hip joint. an increase or decrease in this angle beyond the normal limits causes improper action of muscles, and interferes with walking. an increase in the angle beyond 135 degrees is called **coxa valga** or outward curvature of the hip joint. a decrease in the angle below 120 degrees is called **coxa vara** or inward curvature of the hip joint.

simulation vs. x-ray scans

coxa vara

coxa norma

coxa valga

excellent agreement of simulation and x-ray pattern

pauwels [1973], balle [2004], kuhl & balle [2005]

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

Henry Pang¹, Abhishek P. Shiwalkar¹, Chris M. Madormo¹, Rebecca E. Taylor¹, 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]

how does henry's bone grow?

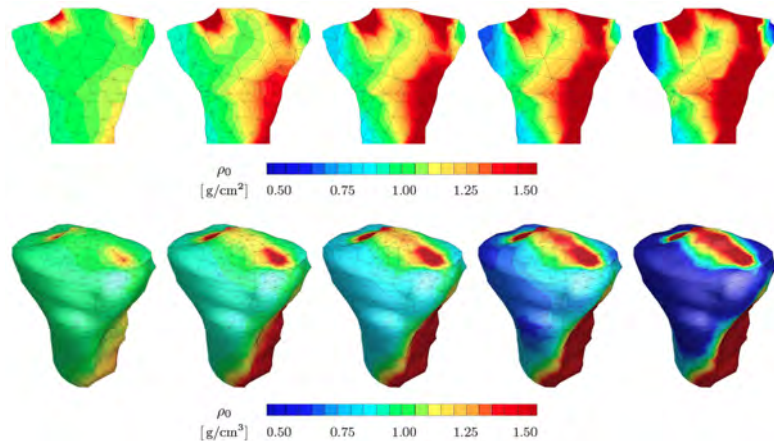
$$\mathcal{R}_0 = c \left[\frac{\rho_0}{\rho_0^*} \right]^n - m \psi_0 - \psi_0^*$$

[g/cm³/time]

ρ_0 [g/cm³]

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

how does henry's bone grow?



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

example - henry's knee

how predictive is the simulation?

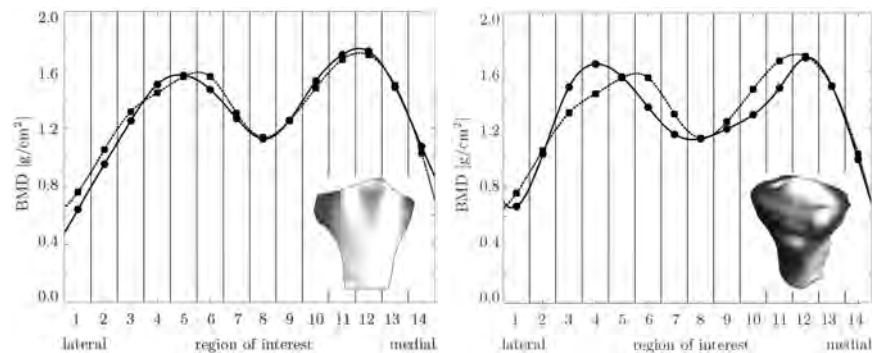


	DEXA [g/cm ²]	2d FEM [g/cm ²]	2d error [%]	3d FEM [g/cm ³]	3d error [%]
BMD _{lat}	1.268	1.233	2.73	1.384	9.33
BMD _{med}	1.621	1.637	0.99	1.556	3.99
	DEXA [-]	2d FEM [-]	2d error [%]	3d FEM [-]	3d error [%]
M:L-ratio	1.279	1.327	3.83	1.122	12.18

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

example - henry's knee

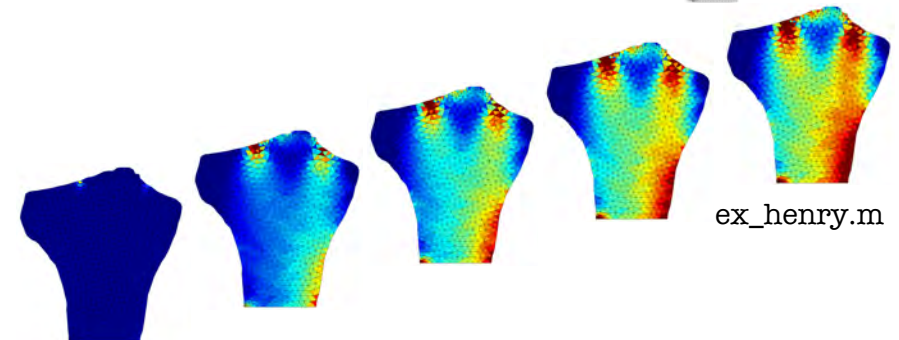
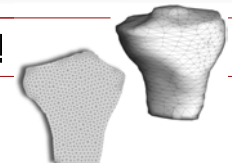
how predictive is the simulation?



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