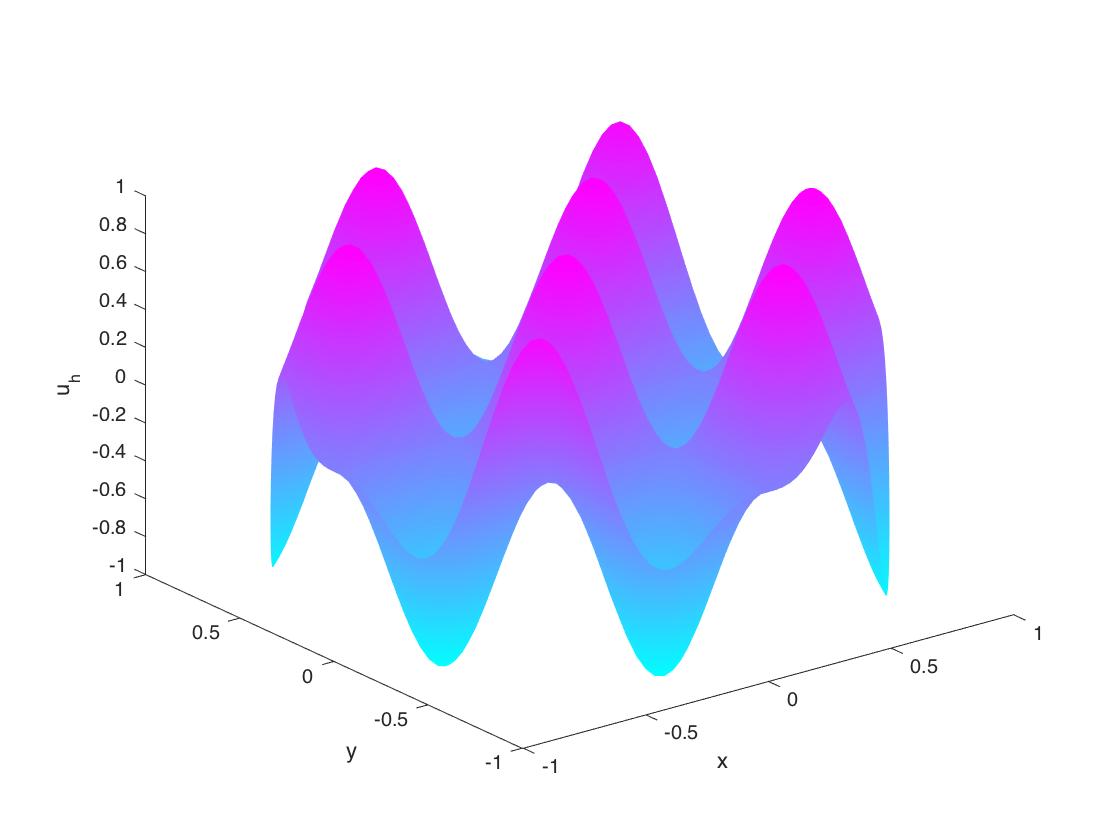
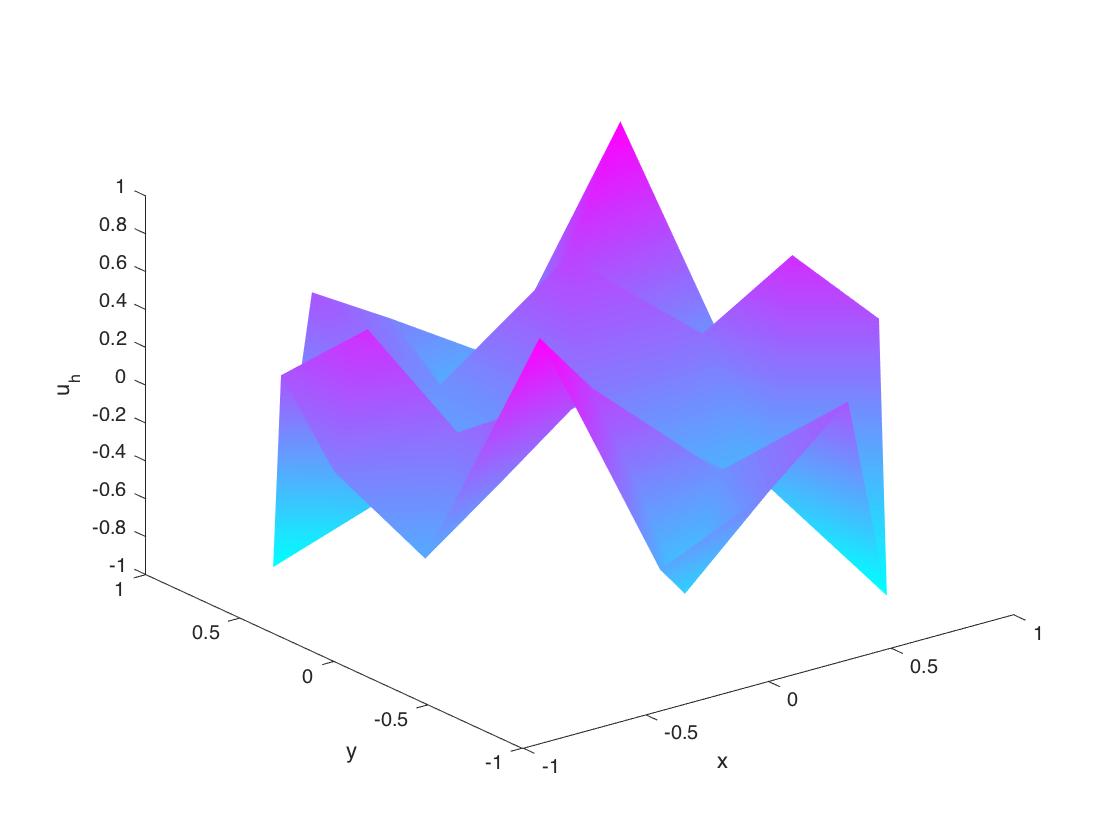
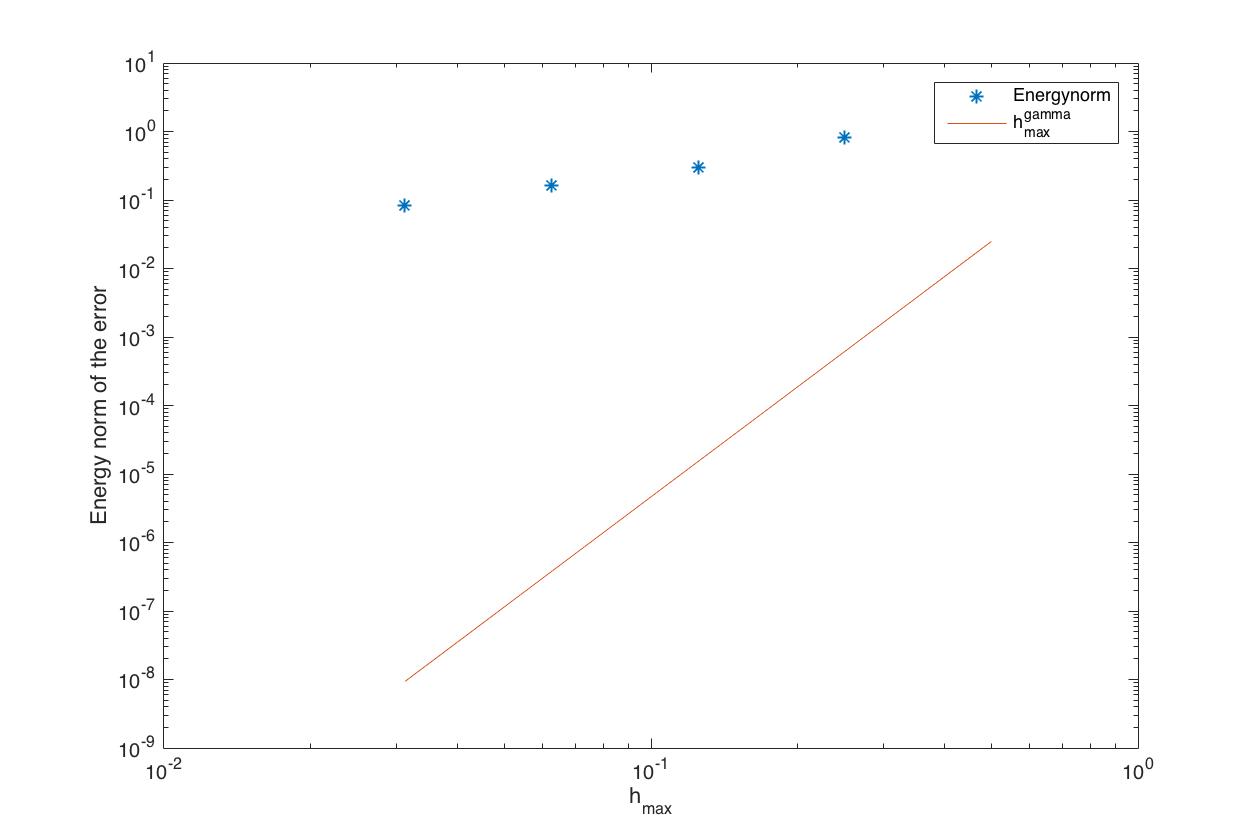
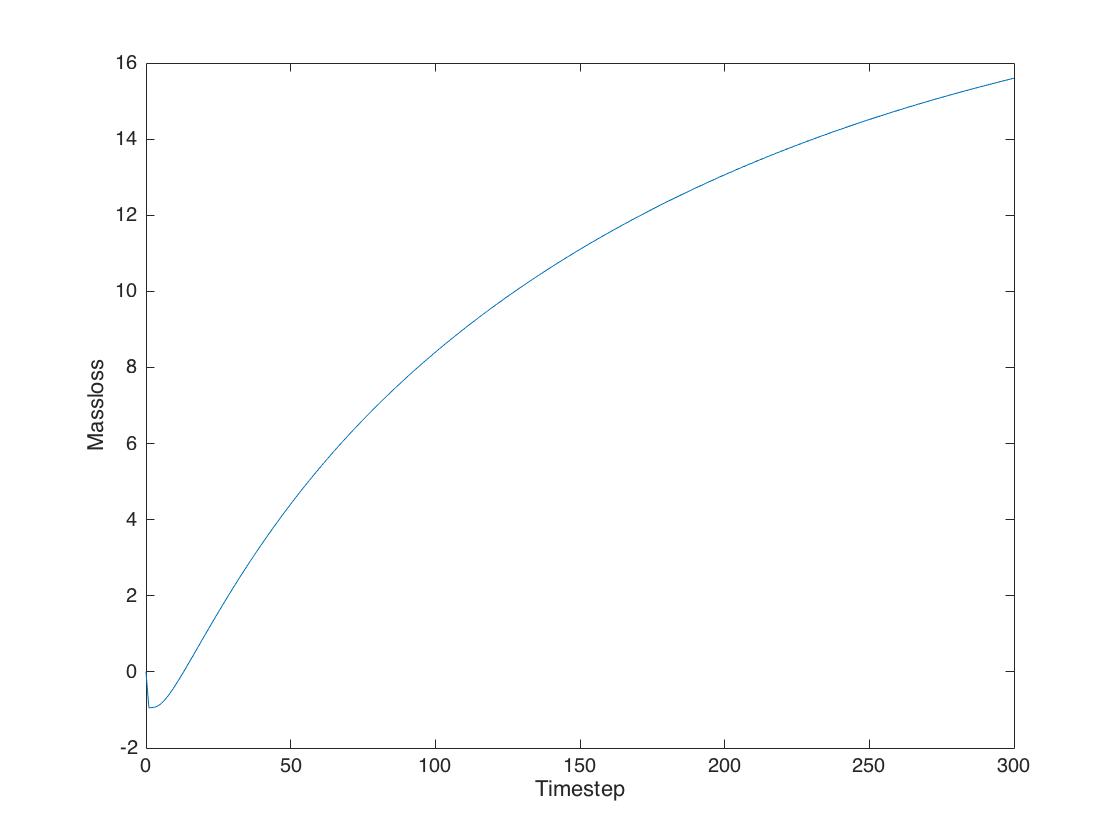
Part A

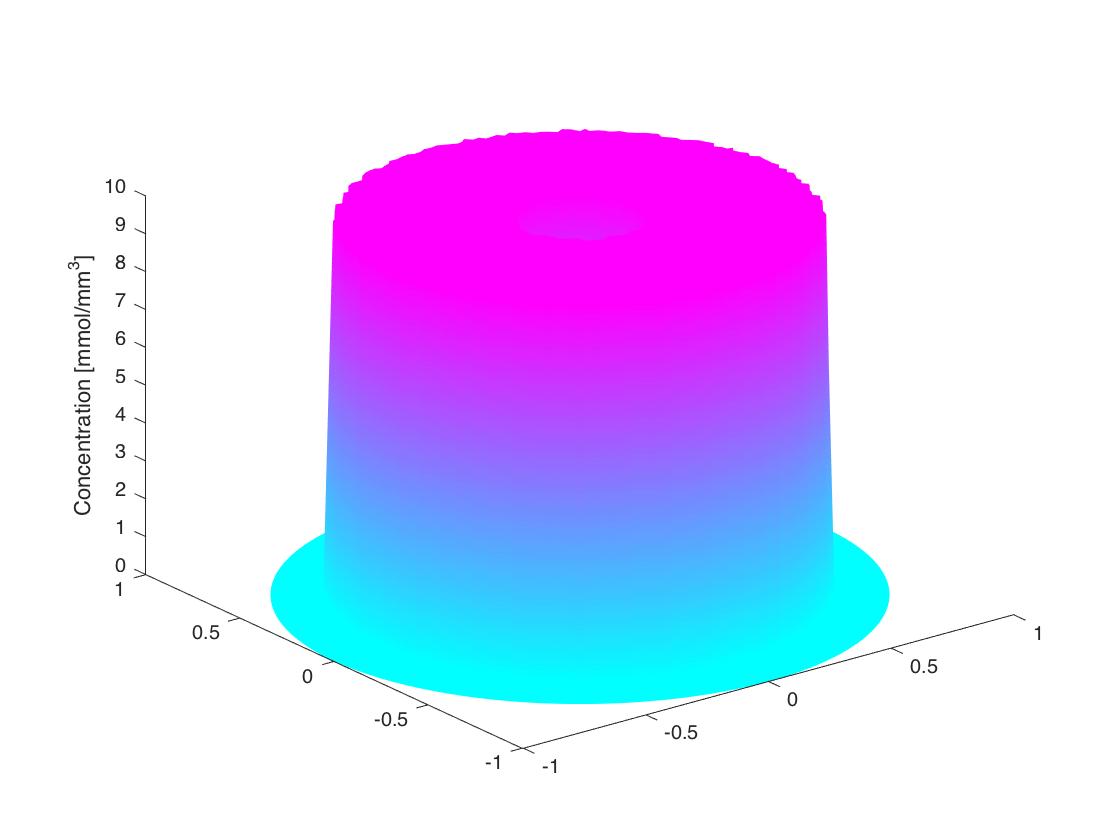
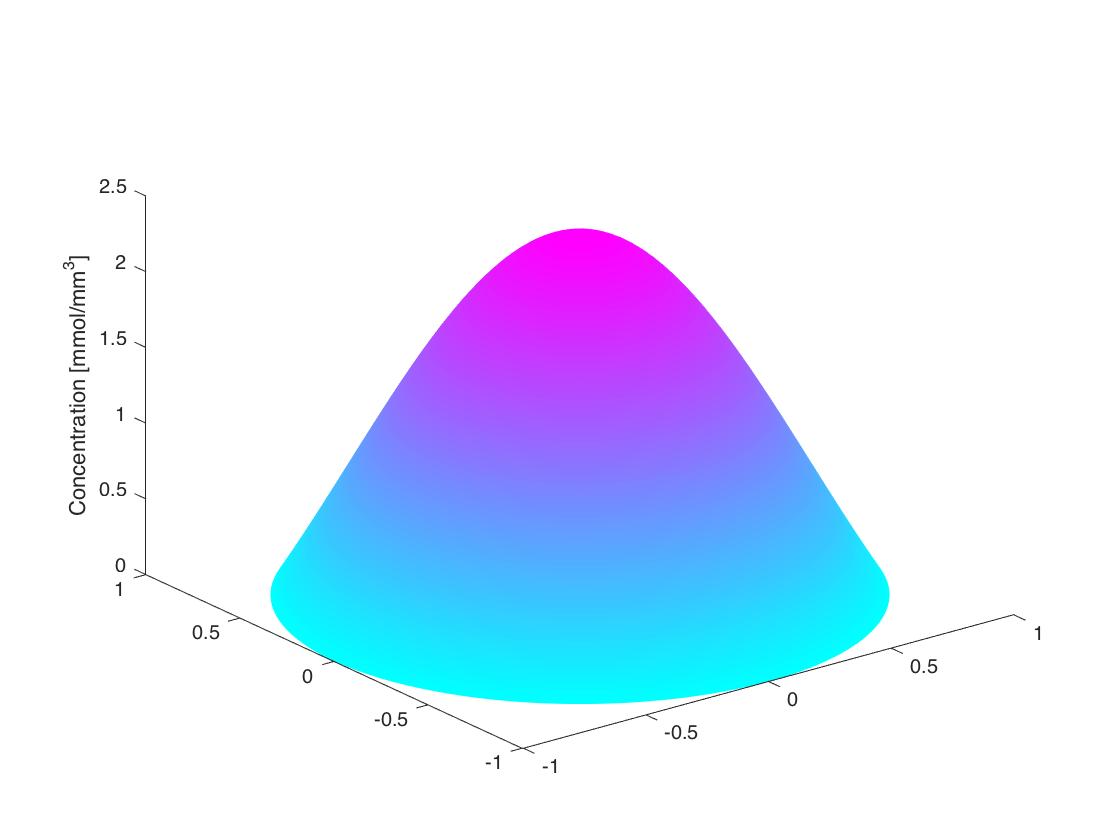


h=1/2 h=1/32



Gamma is around 5





T=30 T=0

close all

clear all

%comment

geometry =

B2A

@circleg;

error2 = zeros(5,1);

step = zeros(5,1);

for h = 1:5

    step(h) = 1/2^h;

    hmax = 1/2^h;

    [p,e,t] = initmesh(geometry ,'hmax',hmax);

    np = size(p,2);

    A = sparse(np, np);

    bk = zeros(np,1);

    r = zeros(np,1);

    I = eye(length(p));

    for K = 1:size(t,2);                    % loop over the triangles

        nodes = t(1:3,K);                     % find triangle K?s nodes

        % compute the (3 x 3) stiffness matrix AK

        nodeCoordinates = p(:,nodes);

        [area, b ,c] = Gradients(nodeCoordinates(1,:), nodeCoordinates(2,:));

        CoG = sum(nodeCoordinates, 2)/3;

        AK = area\*[b\*b' + c\*c'];

        A(nodes,nodes) = A(nodes,nodes)+AK;   % add AK(i,j), i,j=1,2,3,

        % to A(nodes(i),nodes(j))

        %Compute bik

        bk(nodes,1) = bk(nodes,1)+ [1; 1; 1].\*myFun(CoG(1), CoG(2))\*area/3;

    end

    %Compute rik by iterating through the edges

%     for E = 1:size(e,2)

%         nodes = e(1:2, E);

%         x = p(1, nodes);

%         y = p(2, nodes);

%         length\_E = sqrt((x(1)-x(2))^2+(y(1)-y(2))^2);

%         r(nodes,1) = r(nodes,1) + length\_E\*myExacFun(x,y)'/2;

%     end

    A(e(1,:),:) = I(e(1,:),:);

    bk(e(1,:)) = 0;

    fixed = unique([e(1,:) e(2,:)]);            % boundary nodes

    g = myExacFun(p(1, fixed),p(2, fixed));

    free = setdiff([1:np],fixed);               % interior nodes

    bk = bk(free)-A(free,fixed)\*g';             % modify load vector

    A = A(free,free);                           % modify stiffness matrix

    Z = zeros(np,1);                            % allocate solution vector

    Z(fixed) = g';                              % insert fixed node values

    Z(free) = A\bk;                             % solve for free node values

    %Error

    exactSol = myExacFun(p(1, free),p(2, free));

    error1 = exactSol'-Z(free);

    error2(h) = sqrt(error1'\*A\*error1);

    if (h == 1 | h == 5)

         pdesurf(p,t,Z)

         xlabel('x','fontsize',16)

         ylabel('y','fontsize',16)

         zlabel('u\_h','fontsize',16)

         figure

    end

    %pdemesh(p,e,t)

    %     figure

end

loglog(step,error2,'\*')

gamma = abs((error2(end)-error2(1))/(1/2-1/32));

hold on

loglog(step,step.^gamma);

xlabel('h\_{max}','fontsize',16)

ylabel('Energy norm of the error','fontsize',16)

legend('Energynorm', 'h\_{max}^{gamma}')

B2C

close all

clear all

tic

L = 300;                % number of time steps

T = 30;                  % final time

t = linspace(0,T,L+1);   % time grid

h = 5;

alph = 0.01;

geometry = @circleg;

hmax = 1/2^h;

[p,e,t2] = initmesh(geometry ,'hmax',hmax);

U = labfun(p(1,:), p(2,:))';          % inital condition

A = StiffMat2D(p,t2,1);

M = MassMat2D(p, t2);

b0 = LoadVec2D(p, t2, @labfun);

b1 = LoadVec2D(p, t2, @labfun2);

np = size(p,2);

fixed = unique([e(1,:) e(2,:)]);            % boundary nodes

free = setdiff([1:np],fixed);               % interior nodes

g = zeros(size(fixed,2),1);

b0 = b0(free);             % modify load vector

b1 = b1(free);

A = A(free,free);

M = M(free,free);

U(fixed) = 0;

Mass0 = 0;

UR = U;

for K = 1:size(t2, 2);

    nodes = t2(1:3,K);

    area = polyarea(p(1,nodes), p(2,nodes));

    Mass0 = Mass0 + 1/3\*sum(U(nodes))\*area;

end

pdesurf(p,t2,U)

MassT = zeros(1,L);

zlabel('Concentration [mmol/mm^3]', 'fontsize', 16);

for l = 1:L

    k = t(l+1) - t(l);

    U(free) = (M+k/2\*A\*alph)\((M- k\*alph/2\*A)\*U(free)+k/2\*(b1+b0));    %nota that b should be zero...

    b0 = 0;

    %           pdesurf(p,t2,U)

    %           drawnow

    %           pause(T/100);

    for K = 1:size(t2, 2);

        nodes = t2(1:3,K);

        area = polyarea(p(1,nodes), p(2,nodes));

        MassT(l) = MassT(l) + 1/3\*sum(U(nodes))\*abs(area);

    end

end

figure

pdesurf(p,t2,U);

zlabel('Concentration [mmol/mm^3]', 'fontsize', 16)

figure

MassLoss = Mass0 - MassT;

plot(0:300, [0 MassLoss]);

ylabel('Massloss', 'fontsize', 16);

xlabel('Timestep', 'fontsize', 16);

toc

LOAD2VEC

function b = LoadVec2D(p,t,f)

np = size(p,2);

nt = size(t,2);

b = zeros(np,1);

for K = 1:nt

    loc2glb = t(1:3,K);

    x = p(1,loc2glb);

    y = p(2,loc2glb);

    area = polyarea(x,y);

    bK = [f(x(1),y(1));

        f(x(2),y(2));

        f(x(3),y(3))]/3\*area; % element load vector

    b(loc2glb) = b(loc2glb) ...

        + bK; % add element loads to b

end

MyFun

function z = myFun(x,y)

z =8\*pi^2\*sin(2\*pi\*x).\*sin(2\*pi\*y);

STIFF2MAT

function A = StiffMat2D(p,t,a)

np = size(p,2);

nt = size(t,2);

A = sparse(np,np);

for K = 1:nt

    loc2glb = t(1:3,K); % local-to-global map

    x = p(1,loc2glb); % node x-coordinates

    y = p(2,loc2glb); % node y-

    [area,b,c] = Gradients(x,y);

    xc = mean(x); yc = mean(y); % element centroid

    abar = 1; % value of a(x,y) at centroid

    AK =    (b\*b'...

        +c\*c')\*area; % element stiffness matrix

    A(loc2glb,loc2glb) = A(loc2glb,loc2glb) ...

        + AK; % add element stiffnesses to A

end

size(A)

MyExacFun

function z = myExacFun(x,y)

z = sin(2\*pi\*x).\*sin(2\*pi\*y);

Gradients

function [area,b,c] = Gradients(x,y)

area=polyarea(x,y);

b=[y(2)-y(3); y(3)-y(1); y(1)-y(2)]/2/area;

c=[x(3)-x(2); x(1)-x(3); x(2)-x(1)]/2/area;

LabFun2

function z = labfun2(x,y)

z = 0;