Parametrické modelování

L-gramatiky

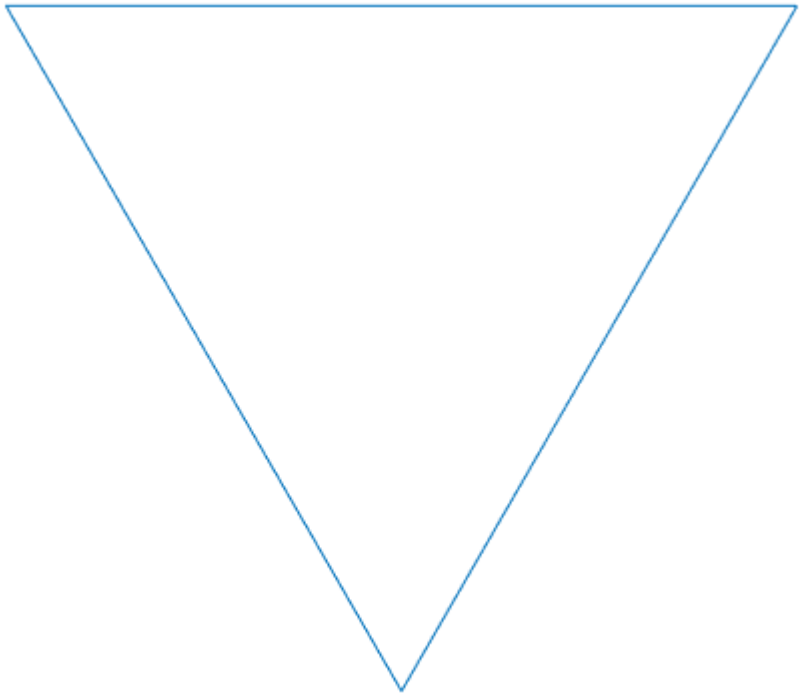
```
pravidla = {{"0", "1[0]1[0]0"}; {"1", "11"}; {"[", "["}; {"]", "]"}};

start = "0";

krok = start;
for i = 1 : 3
    % v tomto pripade mohu aplikovat pravidla od konce
    for j = size(pravidla,1) : -1 : 1
        krok = replace(krok,pravidla{j}{1},pravidla{j}{2});
    end
    krok
end
```

```
krok =
"1[0]1[0]0"
krok =
"11[1[0]1[0]0]11[1[0]1[0]0]1[0]1[0]0"
krok =
"1111[11[1[0]1[0]0]11[1[0]1[0]0]1[0]1[0]0]1111[11[1[0]1[0]0]11[1[0]1[0]0]1[0]1[0]0]11[1[0]1[0]0]11[1[0]1[0]0]1[0]1[
```

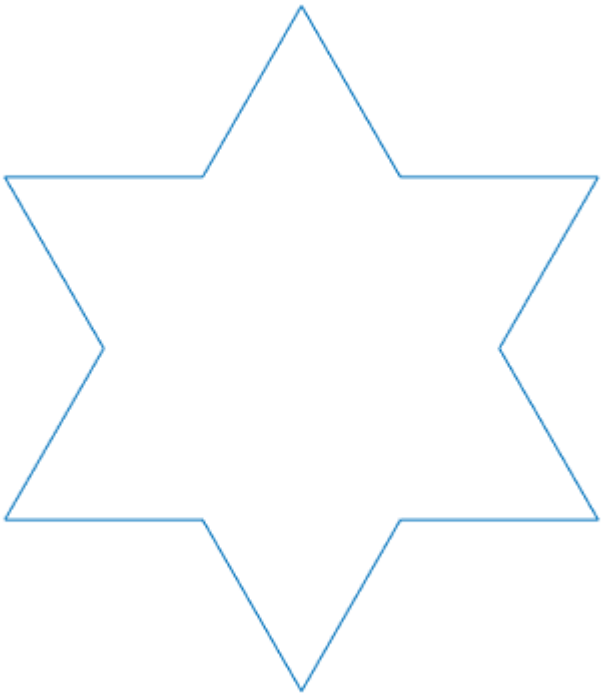
Fraktály

Kochova vločka

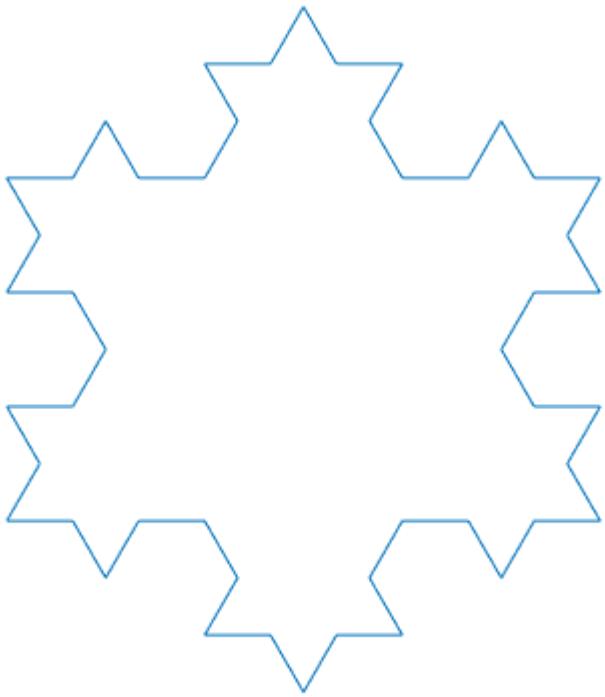
```
figure, koch_vlocka(0)
```



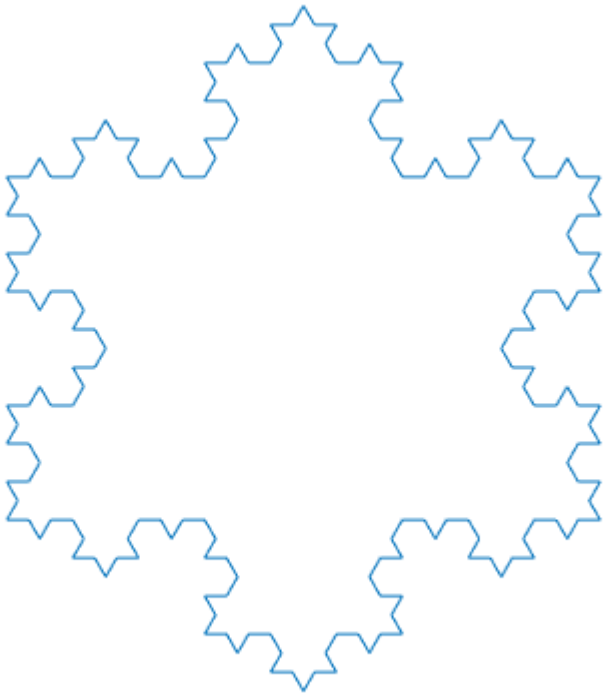
```
figure, koch_vlocka(1)
```



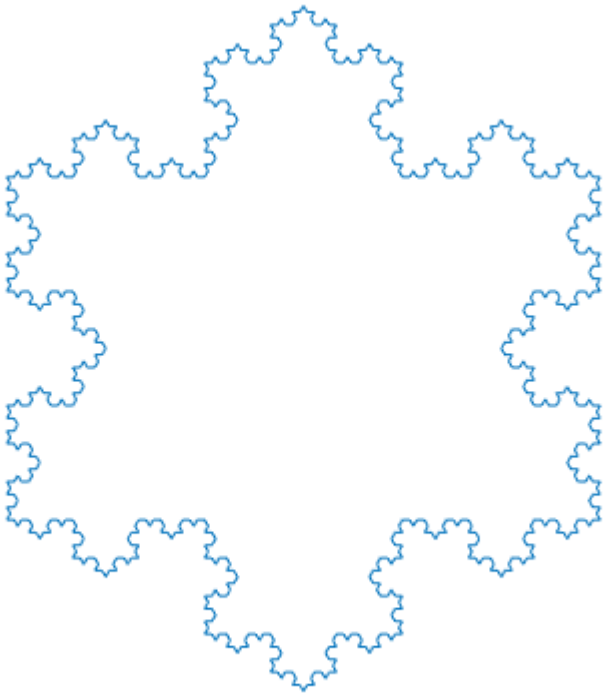
```
figure, koch_vlocka(2)
```



```
figure, koch_vlocka(3)
```



```
figure, koch_vlocka(4)
```



```
pocet = 5;
figure,
for i = 1 : pocet
    subplot(1,pocet,i), koch_vlocka(i-1);
end
```

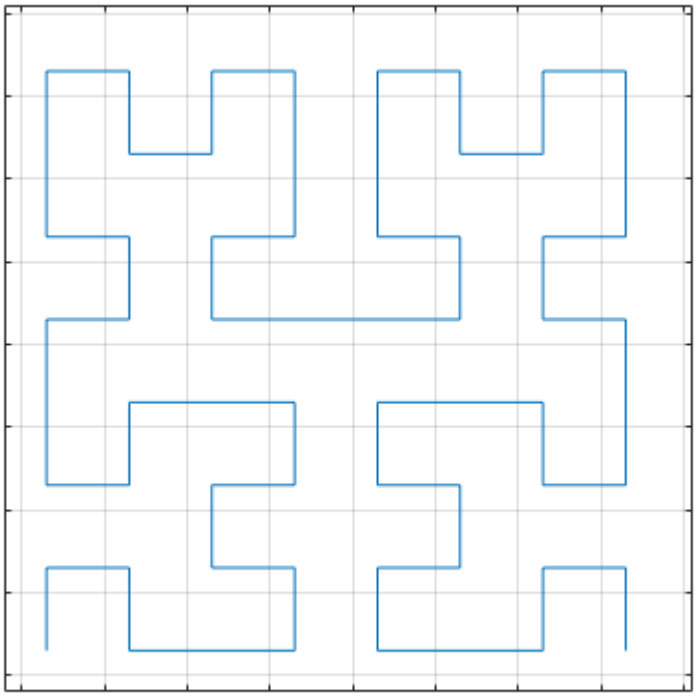
Kochova křivka

```
pocet = 4;
figure,
for i = 1 : pocet
    %koch(i-1);
    %pause(0.2);
    subplot(1,pocet,i), koch(i-1);
end
```

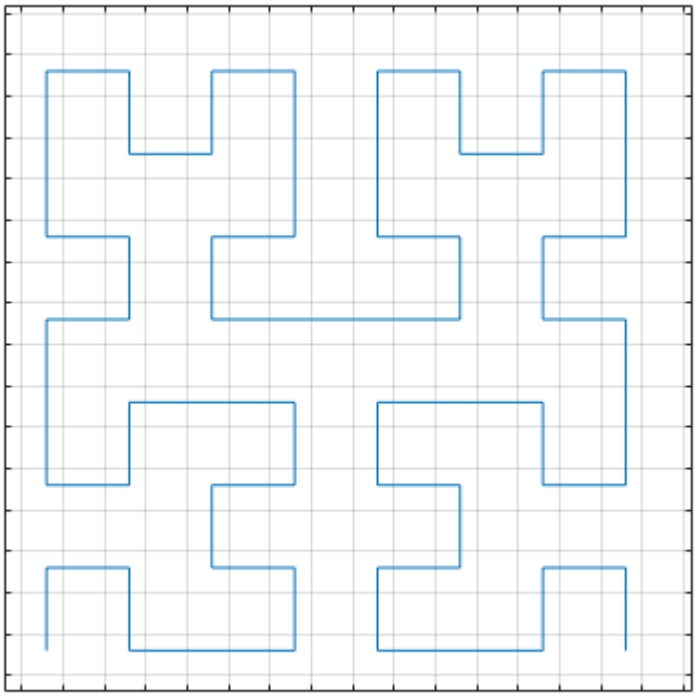
Hilbertova space-filling curve

```
pocet = 5;
figure,
for i = 1 : pocet
    subplot(1,pocet,i), hilbert(i);
end
```

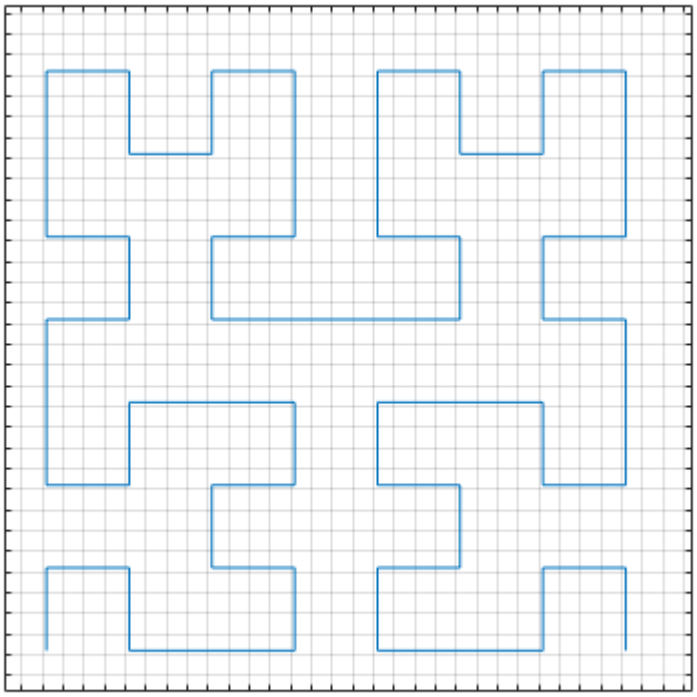
```
figure, hilbert(3);
axis on;
grid on;
ylim([-0.5,7.8]);
xlim([-0.5,7.8]);
set(gca,'xtick',[-0.3:1:10], 'xticklabels', []);
set(gca,'ytick',[-0.3:1:10], 'yticklabels', []);
```



```
figure, hilbert(3);
axis on;
grid on;
ylim([-0.5,7.8]);
xlim([-0.5,7.8]);
set(gca,'xtick',[-0.3:0.5:10], 'xticklabels', []);
set(gca,'ytick',[-0.3:0.5:10], 'yticklabels', []);
```



```
figure, hilbert(3);
axis on;
grid on;
ylim([-0.5,7.8]);
xlim([-0.5,7.8]);
set(gca,'xtick',[-0.3:0.25:10], 'xticklabels', []);
set(gca,'ytick',[-0.3:0.25:10], 'yticklabels', []);
```



Fraktální dimenze - odhad

```
% zluta 20, modra 15, sv. zelena 10, zelena 5
polomery = [20 15 10 5];
pocty = [8 10 14 27];

M = [];

for i = 1 : 3
    for j = i + 1 : 4

        r = polomery(j)/polomery(i);
        dim = log(pocty(j)/pocty(i))/log(1/r);
        M = [M; i j dim];
    end
end

T = table(M(:,1), M(:,2), M(:,3), 'VariableNames',{'i' 'j' 'dimenze'})
```

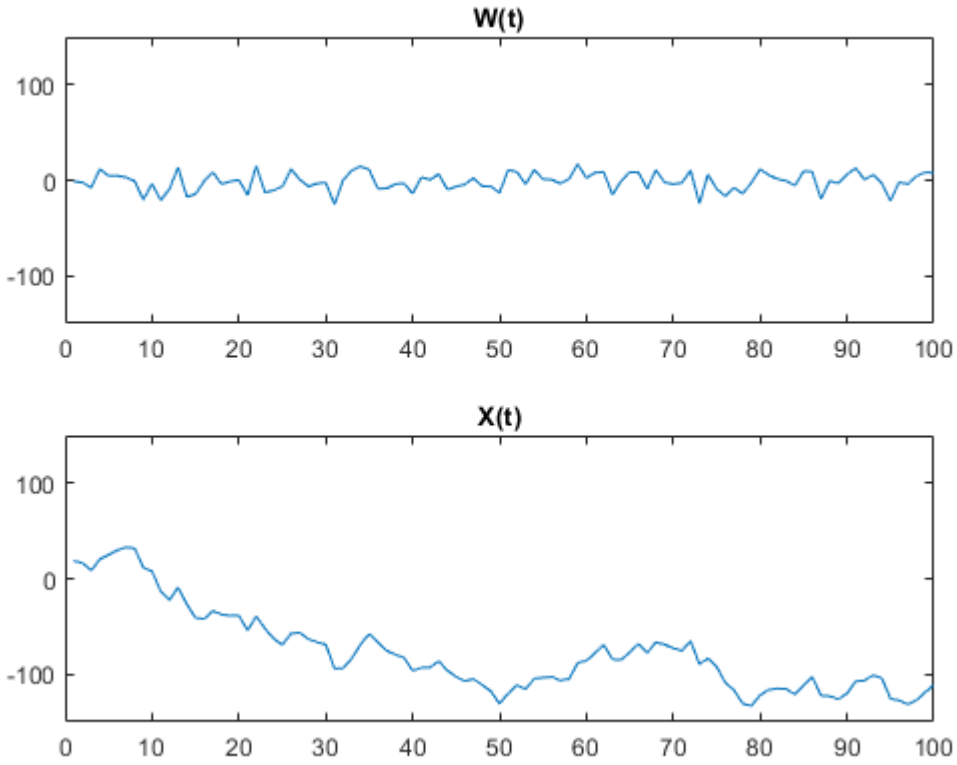
T = 6×3 table

	i	j	dimenze
1	1	2	0.7757
2	1	3	0.8074
3	1	4	0.8774
4	2	3	0.8298

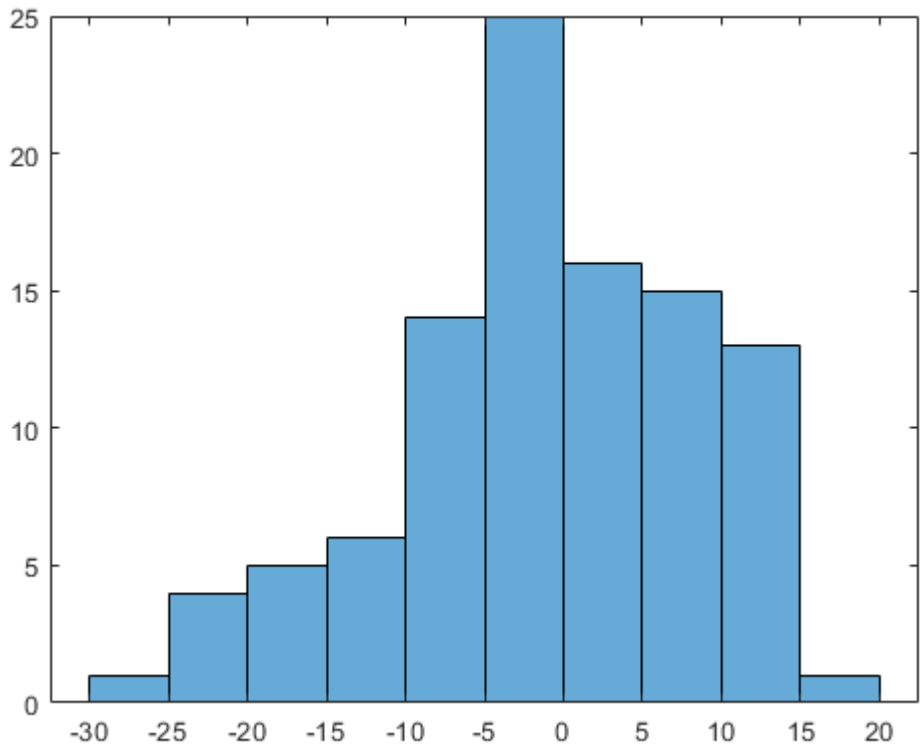
	i	j	dimenze
5	2	4	0.9041
6	3	4	0.9475

Brown motion

```
W = normrnd(0,10,[1,100]);
X_start = 20;

X = X_start + cumsum(W);

figure,
subplot(2,1,1), plot(W);
ylim([-150,150]);
title("W(t)");
subplot(2,1,2), plot(X);
ylim([-150,150]);
title("X(t)");
```




```
figure, histogram(W)
```



Zobecněný BM

```
W = normrnd(0,5,[1,100]);
```

```
X = cumsum(W);
T = 1:100;

dimenze = [1.9 1.5 1.1];
r = 2;

figure,
subplot(size(dimenze,2)+1,1,1), plot(T,X);
title("X(t)");
xlim([0,100]);
ylim([-80,80]);

t1 = 1;
t2 = 50;
t3 = 75;

disp("STD")
```

```
STD
```

```
disp(std(X(t1:t2)));
```

```
   17.5985
```

```
disp(std(X(t2:t3)));
```

```
    5.8599
```

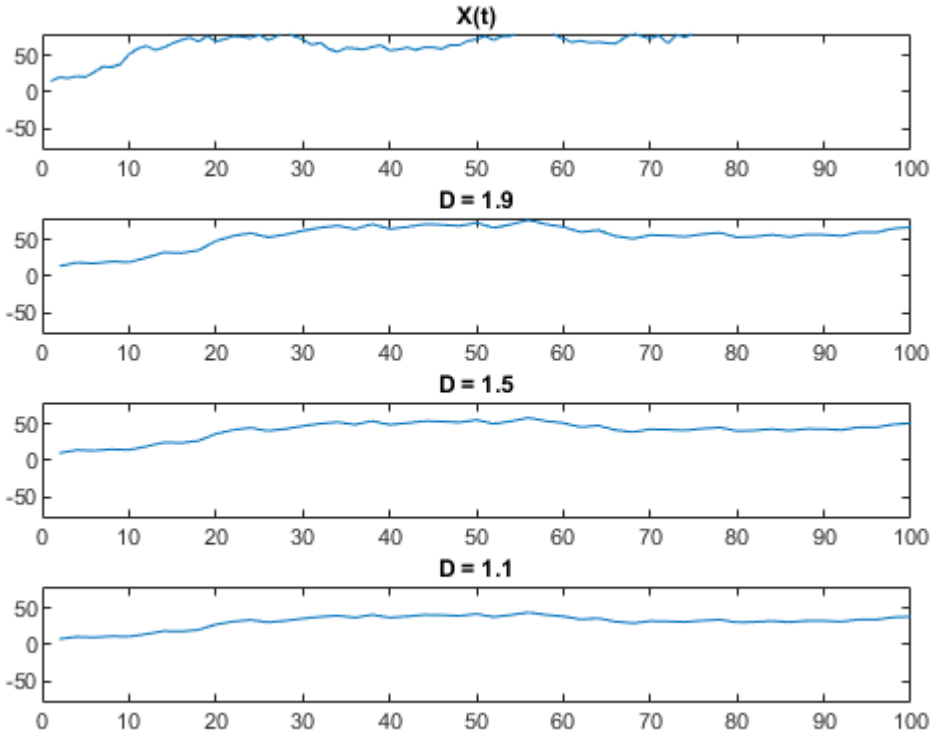
```
disp("Prumer")
```

```
Prumer
```

```
disp(mean(X));
```

```
   71.9926
```

```
for i = 1 : size(dimenze,2)
    H = 2-dimenze(i);
    T2 = r*T;
    X2 = (1/(r^H))*X;
    subplot(size(dimenze,2)+1,1,i+1), plot(T2,X2);
    title("D = " + num2str(dimenze(i)));
    xlim([0,100]);
    ylim([-80,80]);

%     disp("STD " + num2str(dimenze(i)))
%     disp(std(X2(t1:t2)));
%     disp(std(X2(t2:t3)));
   % disp("Prumer " + num2str(dimenze(i)))
   % disp(mean(X2));
end
```




```
X = cumsum(W);
T = 1:100;

dimenze = [1.9 1.5 1.1];
r = 2;

figure,
subplot(size(dimenze,2)+1,1,1), plot(T,X);
title("X(t)");
xlim([0,100]);
ylim([-80,80]);

t1 = 25;
t2 = 50;
t3 = 75;

disp("STD")
```

```
STD
```

```
disp(std(X(t1:t2)));
```

```
    7.5276
```

```
disp(std(X(t2:t3)));
```

```
    5.8599
```

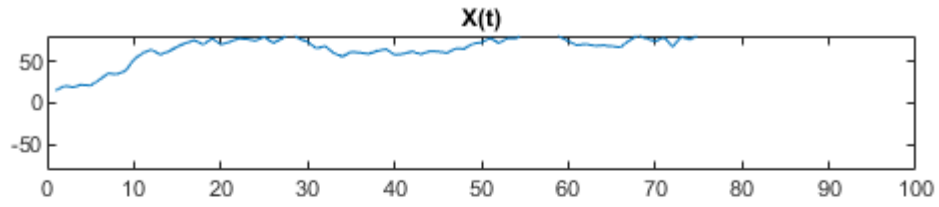
```
disp("Prumer")
```

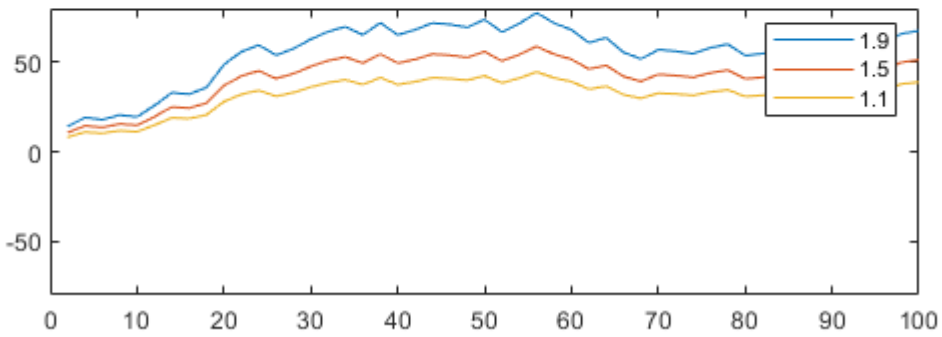
```
Prumer
```

```
disp(mean(X));
```

```
   71.9926
```

```
for i = 1 : size(dimenze,2)
    H = 2-dimenze(i);
    T2 = r*T;
    X2 = (1/(r^H))*X;
    subplot(2,1,2), plot(T2,X2);
    hold on,
    xlim([0,100]);
    ylim([-80,80]);


end
legend(num2str(dimenze(1)), num2str(dimenze(2)), num2str(dimenze(3)));
hold off;
```

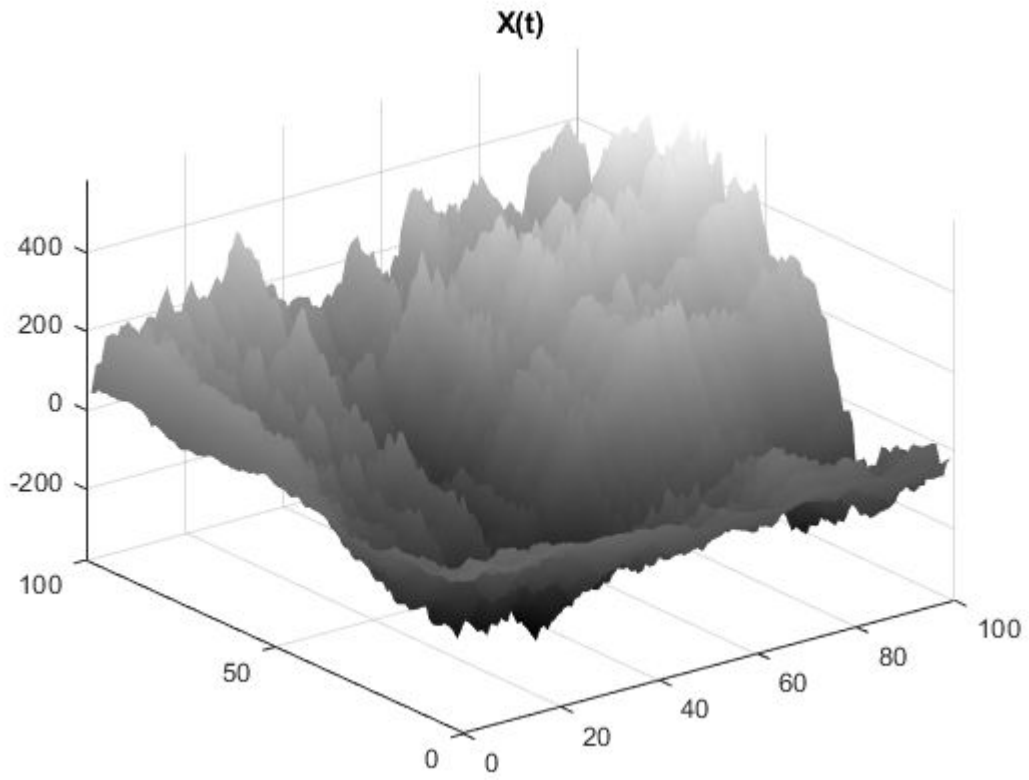





```
W = normrnd(0,5,[100,100]);
```

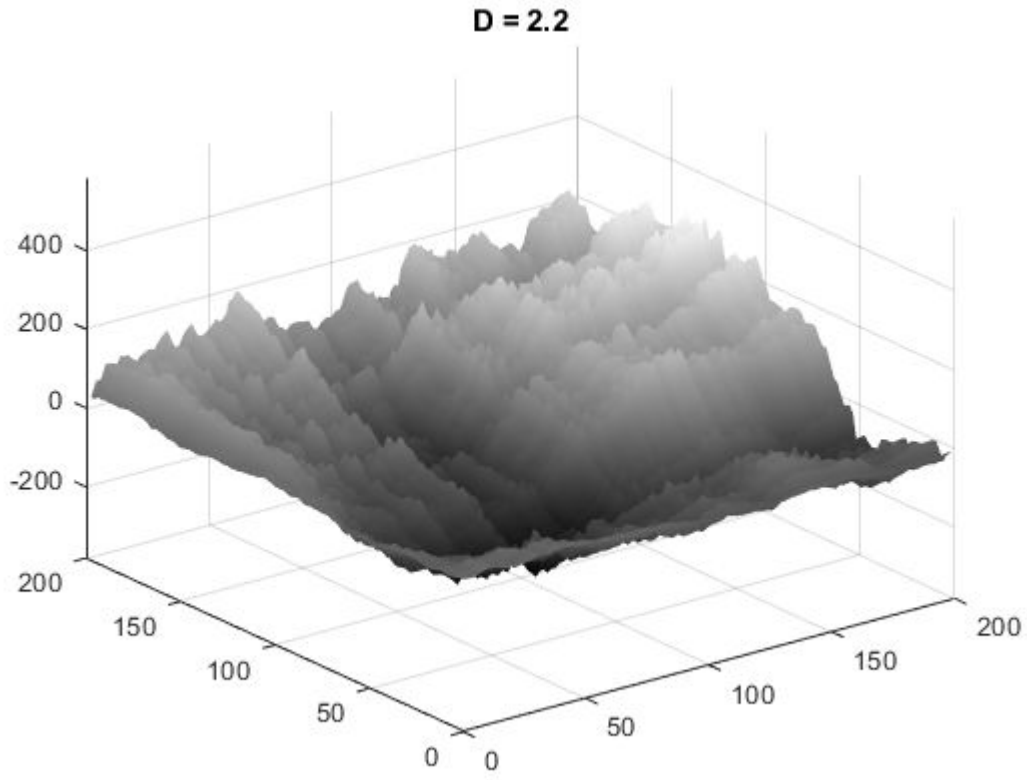
```
X = cumsum(W,1);
X = cumsum(X,2);
```

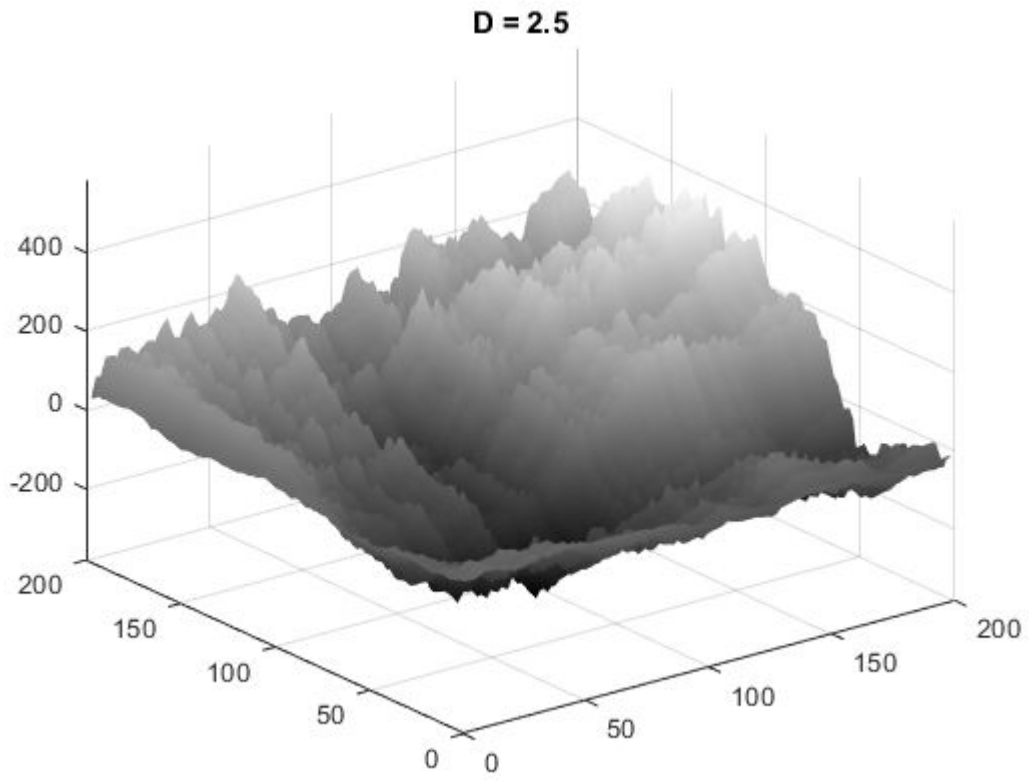
```
[Tx, Ty] = meshgrid(1:100,1:100);

zlimmin = min(min(X));
zlimmax = max(max(X));

dimenze = [2.2 2.5 2.8];
r = 2;

figure,
figure, surf(Tx,Ty,X, 'EdgeColor',"none");
title("X(t)");
zlim([zlimmin-10 zlimmax+10]);
colormap gray;
shading interp
```



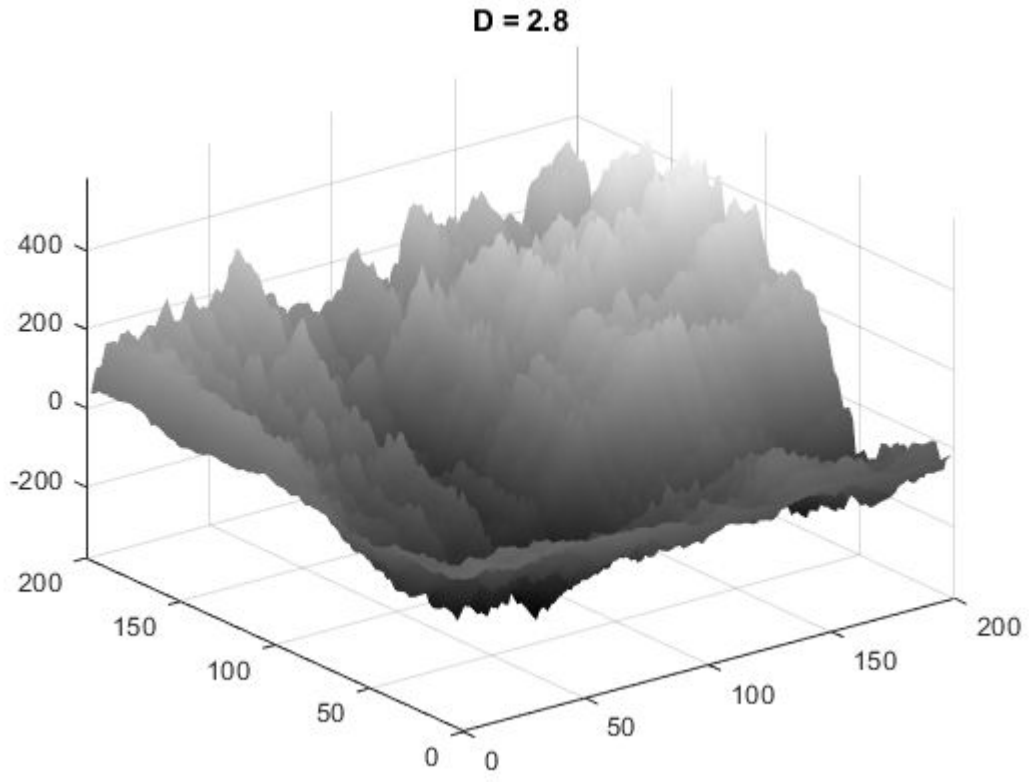

```
for i = 1 : size(dimenze,2)
    H = 3-dimenze(i);
    Tx2 = r*Tx;
    Ty2 = r*Ty;
    X2 = (1/(r^H))*X;
    figure,  surf(Tx2,Ty2,X2, 'EdgeColor',"none");
```

```
    title("D = " + num2str(dimenze(i)));
    zlim([zlimmin-10 zlimmax+10]);
    colormap gray;
shading interp
end
```



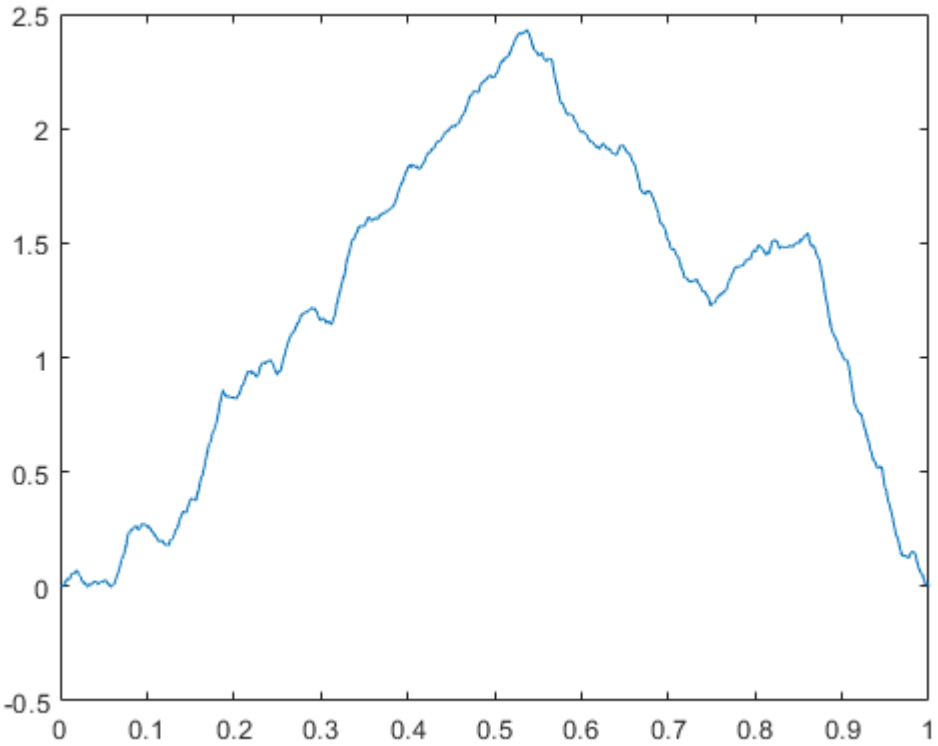

D = 2.5
400
200
0
-200
200
150
100
50
0
0
50
100
150
200

```
hold off;
```


D = 2.8
400
200
0
-200
200
150
100
50
0
0
50
100
150
200

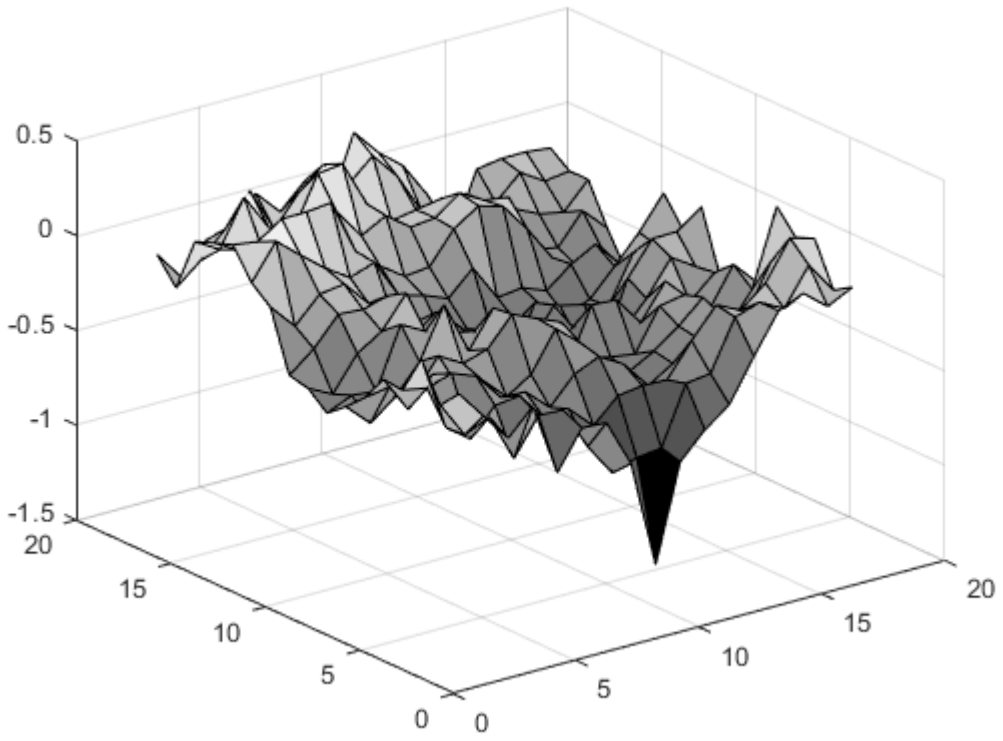
Náhodné přesouvání středu

```
[T,X] = nahodnePresouvani(0,0,1,0,1,10,1.5);
figure, plot(T,X);
```



Diamond square algoritmus

```
M = nahodne_presouvani3D([0 0;0 0], 1.5, 5);
figure, surf(M);
colormap gray;
```



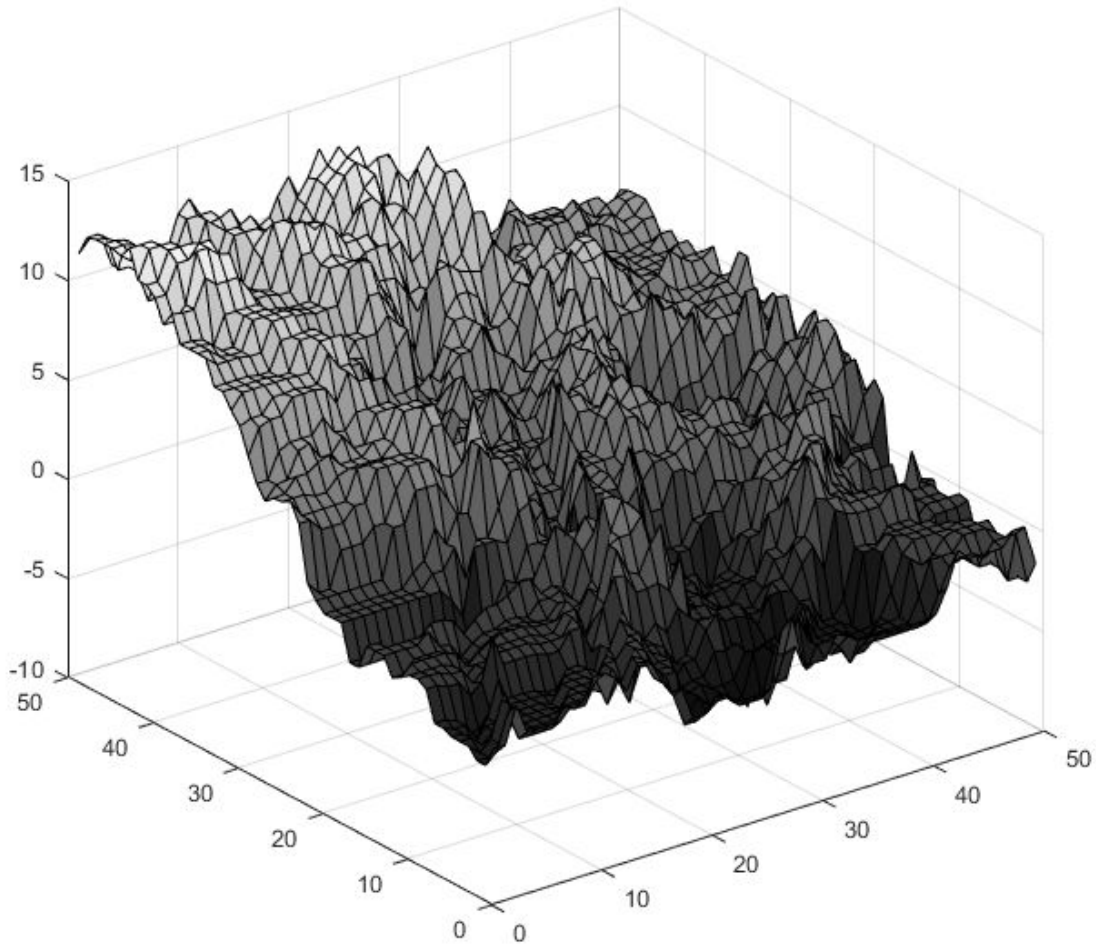

```
M = nahodne_presouvani3D([0 0;0 0], 1.5, 10);
imshow(M);
```

```
%imwrite(M,'mrak.png');
```

Náhodné poruchy

```
M = ones(50);

H = 3-2.5;

idx = 1;

for iter = 1 : 100

    x1 = randi([-50,100]);

    if (x1 > 50)
        x2 = randi([-50,50]);
    elseif (x1 < 0)
        x2 = randi([0,100]);
    else
        x2 = randi([-50,100]);
```

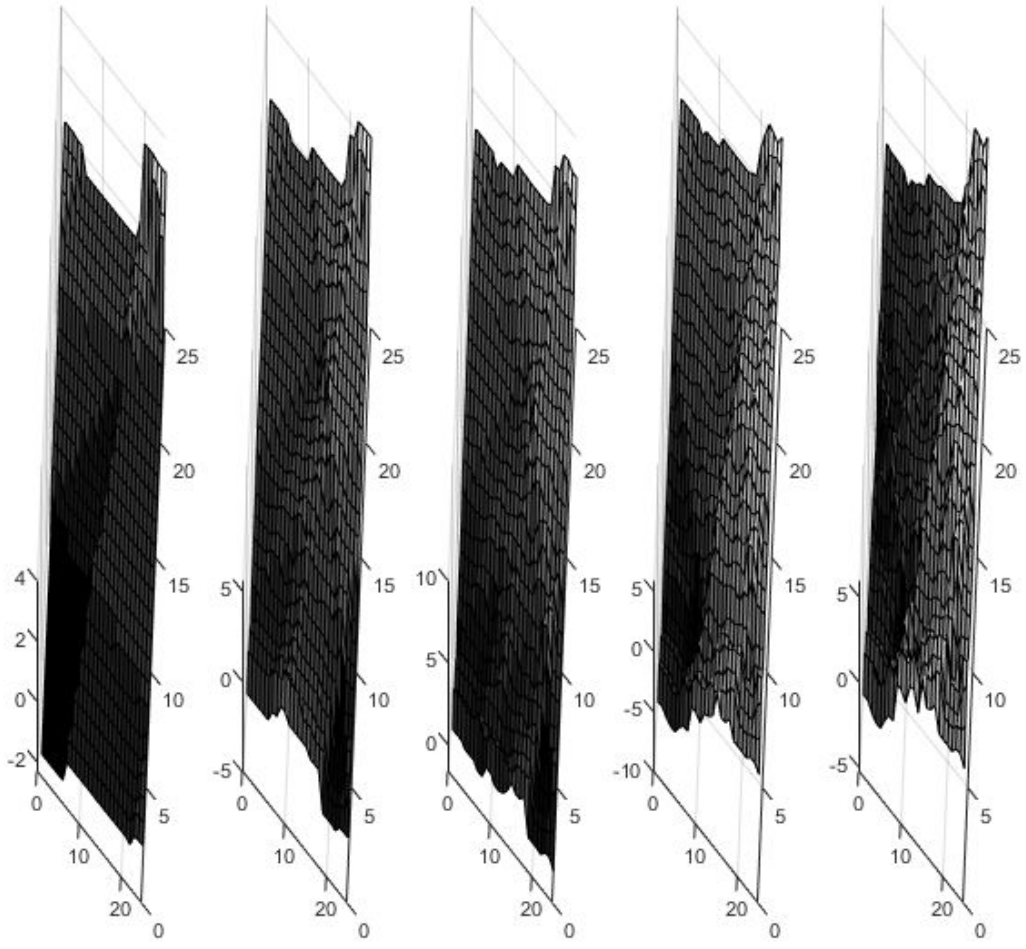
```
    end

    N =  DDA(x1+50+1,1,x2+50+1,50,152);
    N = logical(N(51:100,1:50));

    d = H * normrnd(0,1);
    M(N) = M(N) + d;
    M(~N) = M(~N) - d;

end


surf(M);
colormap gray;
```



obrazek do slidu

```
M = ones(25);
```

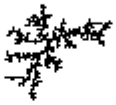
```
H = 3-2.5;

idx = 1;

for iter = 1 : 100

    x1 = randi([-50,100]);

    if (x1 > 25)
        x2 = randi([-50,25]);
    elseif (x1 < 0)
        x2 = randi([0,100]);
    else
        x2 = randi([-50,100]);
    end

    N =  DDA(x1+25+1,1,x2+25+1,50,152);
    N = logical(N(26:50,1:25));

    d = H * normrnd(0,1);
    M(N) = M(N) + d;
    M(~N) = M(~N) - d;

     if mod(iter+8,20)==0
         subplot(1,5,idx), surf(M);
         colormap gray;
         view([12.90 72.00])
         idx = idx+1;
     end
end


colormap gray;
```

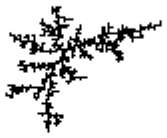


DLA

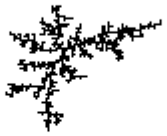
```
velikost = 200;
M = zeros(velikost);

M(ceil(velikost/2),ceil(velikost/2)) = 1;

%M(ceil(velikost/2),:) = 1;



for iter = 1 : 200000
    pozice = randi([0, 3]); % pro bod
    %pozice = randi([0, 1]); % pro caru
    switch(pozice)
        case 0
            x = 1;
            y = randi([1,velikost]);
```

```
        case 1
            x = velikost;
            y = randi([1,velikost]);
        case 2
            x = randi([1,velikost]);
            y = 1;
        case 3
            x = randi([1,velikost]);
            y = velikost;

    end
    % random walk

    while true
        % v ktere dim se pohneme
        dim = randi([0, 1]);
        if dim == 0
            x1 = x + randi([-1,1]);
            if (x1 < 1 || x1 > velikost)
                break;
            end
            y1=y;
        else
            y1 = y + randi([-1,1]);
            if (y1 < 1 || y1 > velikost)
                break;
            end
            x1 = x;
        end
        % dotyk
        if(M(x1,y1)==1)
            M(x,y) = 1;
            break;
        end
        x = x1;
        y = y1;
    end

    if(mod(iter,50000)==0)
         figure, imshow(~M);
         %imwrite(~M,"dla" + num2str(iter) + ".png");
    end
end
```





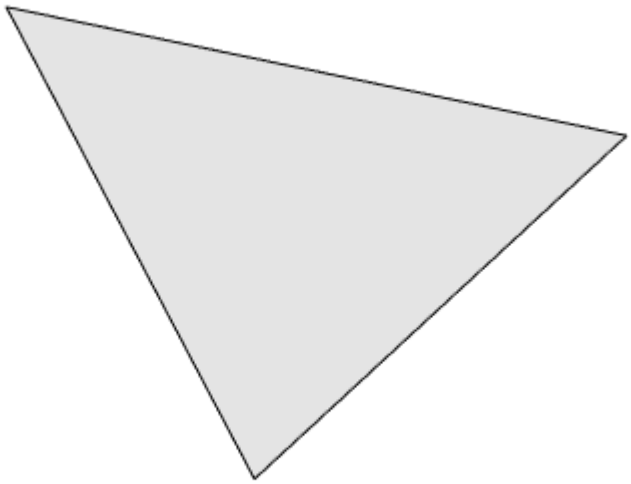
```
figure, imshow(~M);
```



Pobřeží

Náhodné přesouvání středů

```
% tvar ostrova
souradnice = [0 0.01; 0.005 0.007; 0.002 -0.001];
ostrov1 = polyshape(souradnice(:,1),souradnice(:,2));
figure, plot(ostrov1, 'FaceColor',[0.7 0.7 0.7]);
axis off;
```



```
souradnice2 = [];
for i = 1 : size(souradnice, 1) - 1
    [T,X] = nahodnePresouvaniOstrov(souradnice(i,1),souradnice(i,2),souradnice(i+1,1),souradnic
    souradnice2 = [souradnice2; [T(1:end-1)' X(1:end-1)']];
end
[T,X] = nahodnePresouvaniOstrov(souradnice(end,1),souradnice(end,2),souradnice(1,1),souradnice(
souradnice2 = [souradnice2; [T(1:end-1)' X(1:end-1)']];

ostrov2 = polyshape(souradnice2(:,1),souradnice2(:,2));
```

```
Warning: Polyshape has duplicate vertices, intersections, or other inconsistencies that may produce
inaccurate or unexpected results. Input data has been modified to create a well-defined polyshape.
```

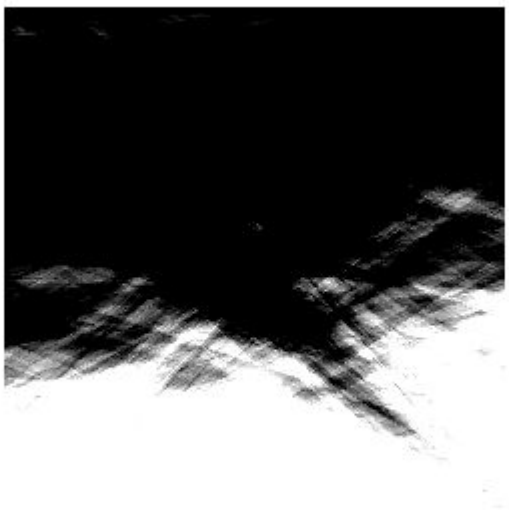
```
figure, plot(ostrov2, 'FaceColor',[0.7 0.7 0.7]);
axis off;
```

Mraky

zobecněný fBm

```
M = ones(250);

H = 3-2.1;

idx = 1;

for iter = 1 : 1500

    x1 = randi([-50,300]);

    if (x1 > 250)
        x2 = randi([-50,250]);
    elseif (x1 < 0)
        x2 = randi([0,300]);
    else
        x2 = randi([-50,300]);
    end

    N =  DDA(x1+50+1,1,x2+50+1,250,400);
    N = logical(N(51:300,1:250));

    d = H * normrnd(0,1);
```

```
    M(N) = M(N) + d;
    M(~N) = M(~N) - d;

end

mrak = M/25;
mrak(mrak<0) = 0;
mrak(mrak>1) = 1;

imshow(imcomplement(mrak));
```

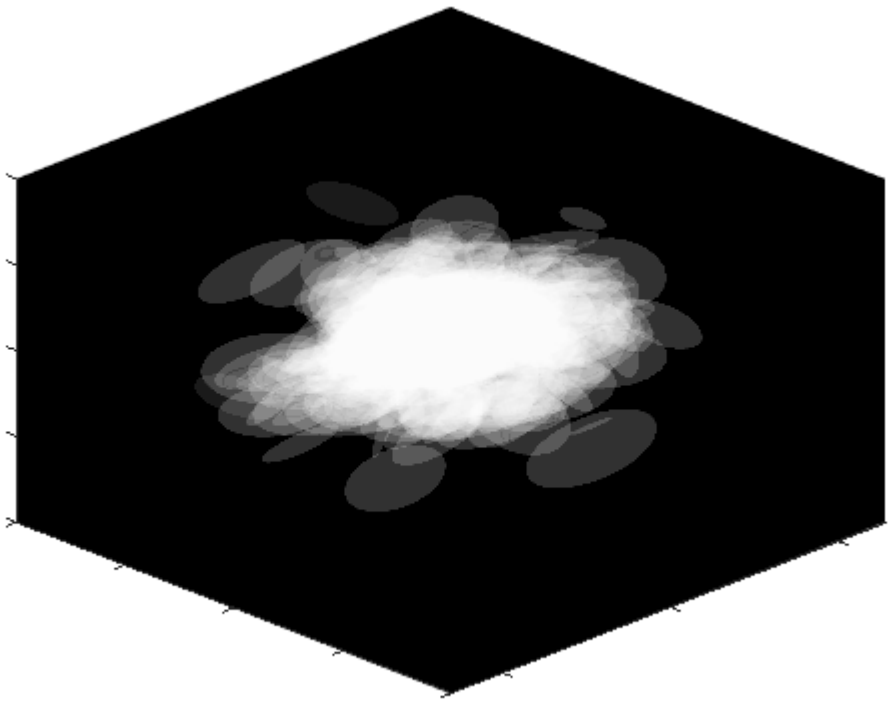


```
imwrite(imcomplement(mrak),'mrak.png');
```

Mrak pomocí elipsoidů

```
figure,
for i = 1 : 300
    r = 10*rand(1);
    [x,y,z] = sphere();

    %zmena meritka
    x = x*(2*rand());
    y = y*(2*rand());
    z = z*(2*rand());



    hold on;
    s = surf(x,y,z,'FaceColor',[1 1 1],'FaceAlpha',0.1,'EdgeColor','none');
    rotate(s,randi([0,1],[1,3]),360*rand());
```

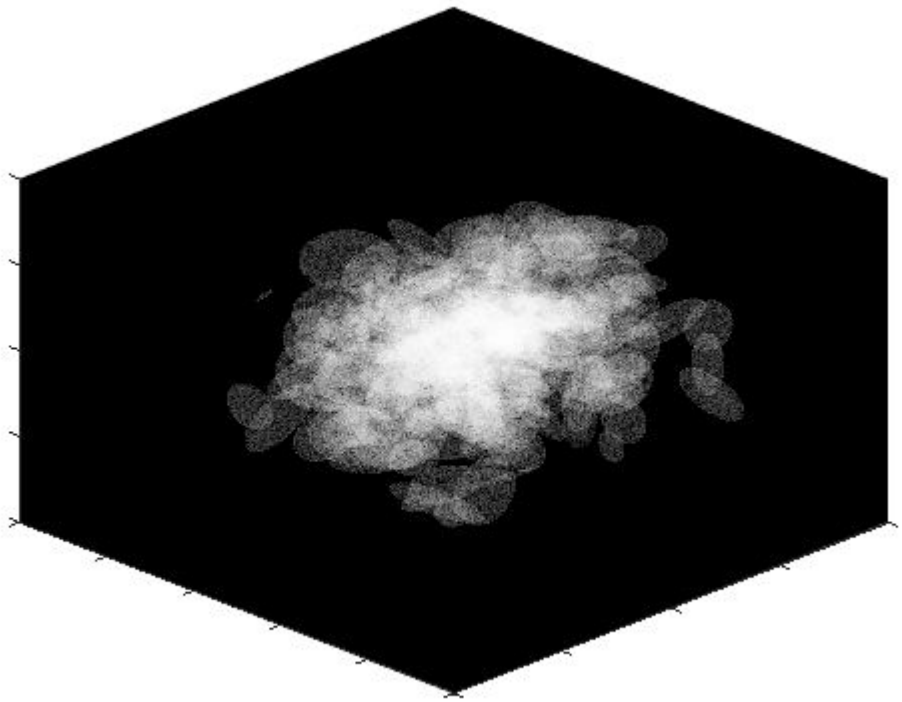
```
    % posun
    XD = get(s, 'XData');
    YD = get(s, 'YData');
    ZD = get(s, 'ZData');
    surf(XD+2*normrnd(0,1), YD+2*normrnd(0,1), ZD+2*normrnd(0,1),'FaceColor',[1 1 1],'FaceAlpha


end
set(gca,'Color','k');
set(gca,'YTickLabel',[]);
set(gca,'XTickLabel',[]);
set(gca,'ZTickLabel',[]);
view([10,10,10])
hold off;
```



náhodná průhlednost stěn elipsidu

```
figure,
for i = 1 : 300

    r = 10*rand(1);
    [x,y,z] = sphere(100);

    %zmena meritka
```

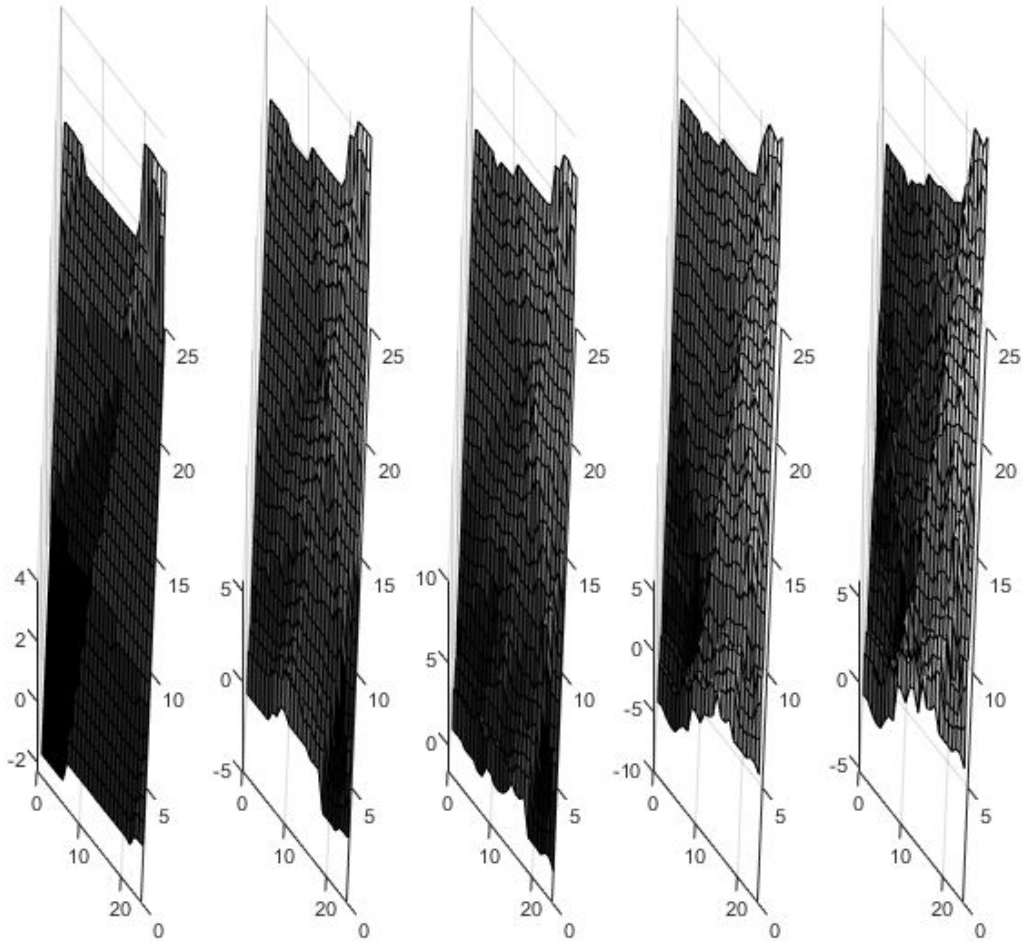
```
    x = x*(2*rand());
    y = y*(2*rand());
    z = z*(2*rand());


    hold on;
    s = surf(x,y,z,'FaceColor',[1 1 1],'FaceAlpha',0,'EdgeColor','none');
    rotate(s,randi([0,1],[1,3]),360*rand());


    % posun
    XD = get(s, 'XData');
    YD = get(s, 'YData');
    ZD = get(s, 'ZData');
    s = surf(XD+2*normrnd(0,1), YD+2*normrnd(0,1), ZD+2*normrnd(0,1),'FaceColor',[1 1 1],'FaceA
    alpha(s,randi([0,200],size(s.XData))/1000);

end
set(gca,'Color','k');
set(gca,'YTickLabel',[]);
set(gca,'XTickLabel',[]);
set(gca,'ZTickLabel',[]);
view([10,10,10])
hold off;
```



Mraky - cellular automata

```
velikost = 100;
% [phs, hum, cld]
data = randi([0,100],[velikost,velikost,velikost,3])>99;
data(1:35,:,:,1) = 0;
data(:,1:35,:,1) = 0;
data(:,:,1:35,1) = 0;
data(65:end,:,:,1) = 0;
data(:,65:end,:,1) = 0;
data(:,:,65:end,1) = 0;

cas = 50;

mraky = zeros([velikost, velikost, velikost]);

pcld = 0.3;
phum = 0.1;
pphs = 0.3;
```

```
figure,
for t = 1 : cas
    for i = 3 : velikost - 2
        for j = 3 : velikost - 1
            for k = 3 : velikost - 2
                mraky(i,j,k) = data(i+2,j,k,1) || data(i+1,j,k,1) || data(i-1,j,k,1) || data(i-
                               data(i,j+1,k,1) || data(i,j-1,k,1) || data(i,j-2,k,1) || ...
                               data(i,j,k+2,1) || data(i,j,k+1,1) || data(i,j,k-1,1) || data(i,
                data(i,j,k,2) = (rand() < phum) || (data(i,j,k,2) && ~data(i,j,k,1));
                data(i,j,k,3) = (rand() < pcld) || (data(i,j,k,3) || data(i,j,k,1));
                data(i,j,k,1) = (rand() < pphs) && (~data(i,j,k,1) && data(i,j,k,2) && mraky(i,
            end
        end
    end

    V = smooth3(mraky,'gaussian',5);
    vol = volshow(V,'BackgroundColor',[0,0,0], 'Renderer', "VolumeRendering", 'IsosurfaceColor'
    alphamap;
    pause(0.01);
end
```



```
velikost = 100;
% [phs, hum, cld]
data = randi([0,100],[velikost,velikost,velikost,3])>99;
data(1:35,:,:,1) = 0;
data(:,1:35,:,1) = 0;
data(:,:,1:35,1) = 0;
data(65:end,:,:,1) = 0;
data(:,65:end,:,1) = 0;
data(:,:,65:end,1) = 0;

cas = 30;

mraky = zeros([velikost, velikost, velikost]);

pcld = 0.3;
phum = 0.1;
pphs = 0.3;
```

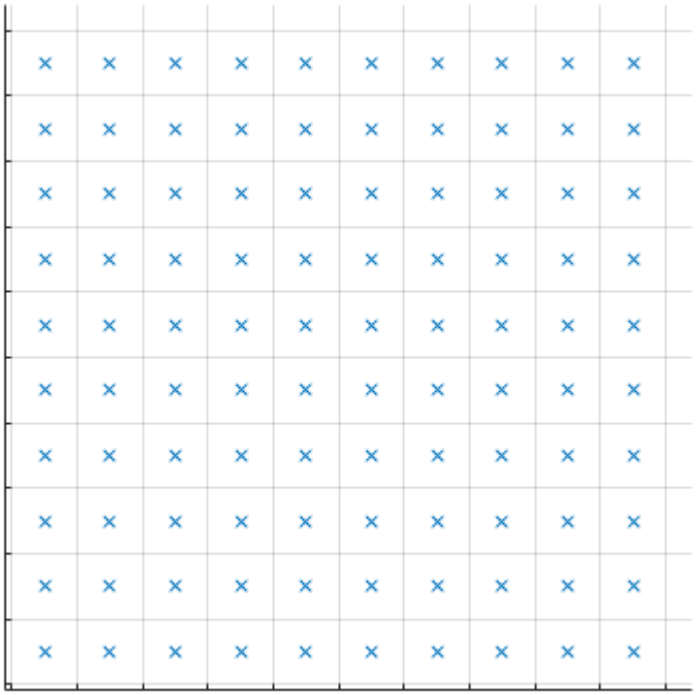
```
figure,
for t = 1 : cas
    for i = 3 : velikost - 2
        for j = 3 : velikost - 1
            for k = 3 : velikost - 2
                mraky(i,j,k) = data(i+2,j,k,1) || data(i+1,j,k,1) || data(i-1,j,k,1) || data(i-
                               data(i,j+1,k,1) || data(i,j-1,k,1) || data(i,j-2,k,1) || ...
                               data(i,j,k+2,1) || data(i,j,k+1,1) || data(i,j,k-1,1) || data(i,
                data(i,j,k,2) = (rand() < phum) || (data(i,j,k,2) && ~data(i,j,k,1));
                data(i,j,k,3) = (rand() < pcld) || (data(i,j,k,3) || data(i,j,k,1));
                data(i,j,k,1) = (rand() < pphs) && (~data(i,j,k,1) && data(i,j,k,2) && mraky(i,
            end
        end
    end
end

V = smooth3(mraky,'gaussian',5);
vol = volshow(V,'BackgroundColor',[0,0,0], 'Renderer', "VolumeRendering", 'IsosurfaceColor',[1
```

```
Warning: Error while executing listener callback for ButtonUp event. Source has been deleted:
Invalid or deleted object.

Error in images.internal.app.volview.CameraController/rotateButtonUp

Error in images.internal.app.volview.CameraController>@(hObj,evt)self.rotateButtonUp(evt)

Error in images.internal.app.volview.VolumeRenderer/updateVolumeWithNewDataLimits

Error in images.internal.volshow.Controller/updateVolumeViewDataLimits

Error in images.internal.volshow.Controller/updateVolumeView

Error in images.internal.volshow.Controller>@(hObj,data)self.updateVolumeView(data)

Error in images.internal.app.volview.Model/set.VolumeData

Error in images.internal.app.volview.Model/triggerVolumeDataChange

Error in volshow/updateProperties

Error in volshow/set.Parent

Error in volshow/parseInputs

Error in volshow

)
vol = volshow(V,'BackgroundColor',[0,0,0], 'Renderer', "VolumeRendering", 'IsosurfaceColor',[1 1 1],
'Alphamap', 0.1*ones([256,1]), 'CameraPosition',[10,0,0]);
```

Les

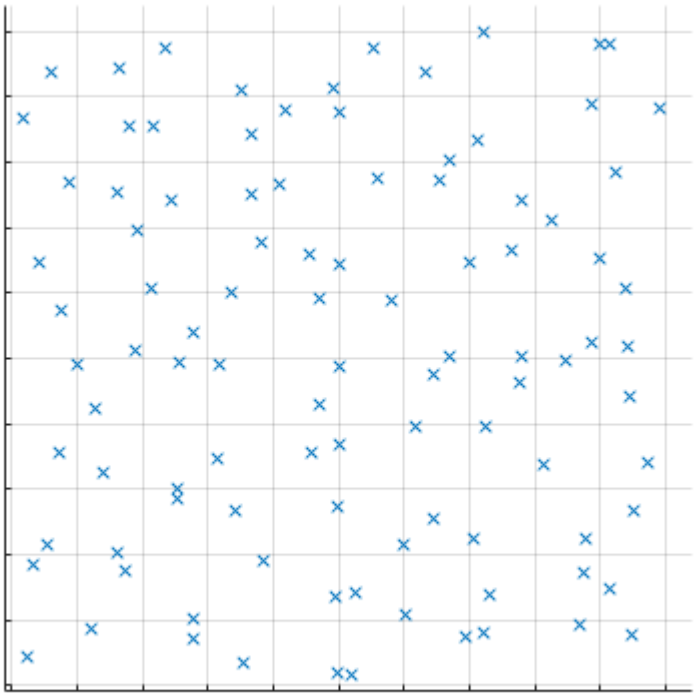
```
% v mrizce 10 x 10
x = 1 : 10:100;
y = 1 : 10:100;

[X,Y] = meshgrid(x,y);
```

```
% kazdy radek reprezentuje souradnice jednoho stromu
stromy = [X(:), Y(:)];

figure,
scatter(stromy(:,1), stromy(:,2), 'x');
axis equal
grid on
xticks(-4 : 10 : 105)
xticklabels([]);
xlim([-5,100])
yticks(-4 : 10 : 105)
yticklabels([]);
ylim([-5,100])
```

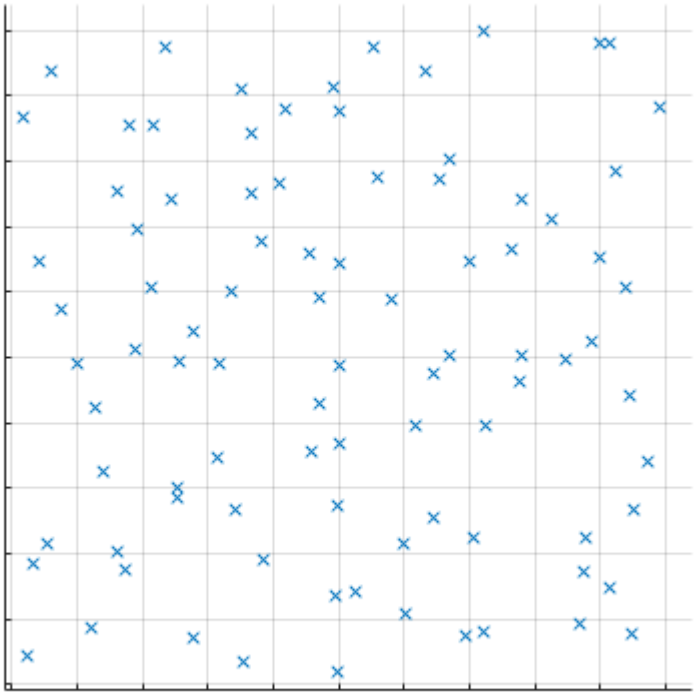


```
% jitter
stromy2 = stromy + randi([-499,499],size(stromy))/100;
figure,
scatter(stromy2(:,1), stromy2(:,2),'x');
axis equal
grid on
xticks(-4 : 10 : 105)
xticklabels([]);
xlim([-5,100])
yticks(-4 : 10 : 105)
yticklabels([]);
```

```
ylim([-5,100])
```



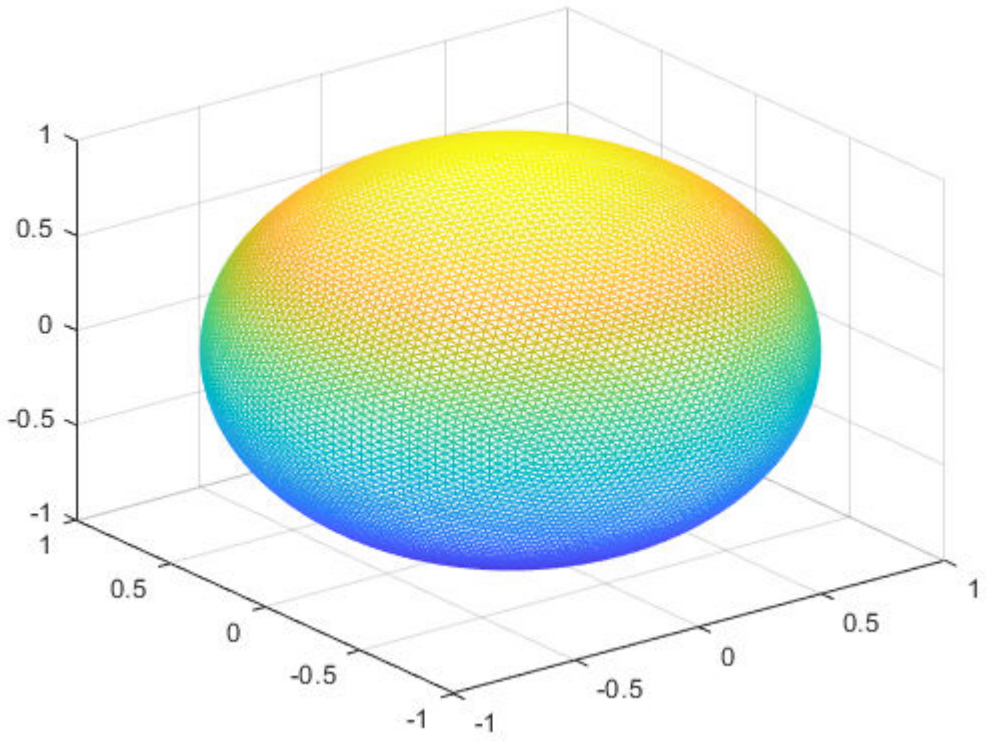
```
% vynechani
procento = 10;
k = randperm(size(stromy,1));
stromy3 = stromy2(k(1:floor(size(stromy,1)*(1-procento/100))),:);
figure,
scatter(stromy3(:,1), stromy3(:,2),'x');
axis equal
grid on
xticks(-4 : 10 : 105)
xticklabels([]);
xlim([-5,100])
yticks(-4 : 10 : 105)
yticklabels([]);
ylim([-5,100])
```



Planety

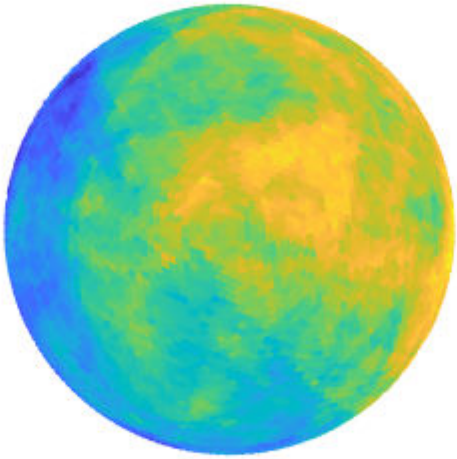
icoSphereMesh -- https://www.mathworks.com/matlabcentral/fileexchange/54434-icospheremesh-n

```
koule = icoSphereMesh(8);

 trimesh(koule.face, koule.x, koule.y, koule.z);
```




```
koule.barva = 128*ones(size(koule.x));
iteraci = 600;
for iter = 1 : iteraci
    pripad = randi([1 3]);
    theta = rand()*2*pi;
    switch (pripad)
        case 1
            AM = [1 0 0 0;
                  0 cos(theta) sin(theta) 0;
                  0 -sin(theta) cos(theta) 0;
                  0 0 0 1];
        case 2
            AM = [cos(theta) 0 -sin(theta) 0;
                  0  1 0 0;
                  sin(theta) 0   cos(theta) 0;
                  0 0 0 1];
        case 3
            AM = [cos(theta) -sin(theta) 0  0;
                  sin(theta)  cos(theta) 0 0;
                  0 0  1 0;
                  0 0 0 1];
    end
    tform = affine3d(AM);
    [koule.x,koule.y,koule.z] = transformPointsForward(tform,koule.x, koule.y, koule.z);
    d = randn();
```

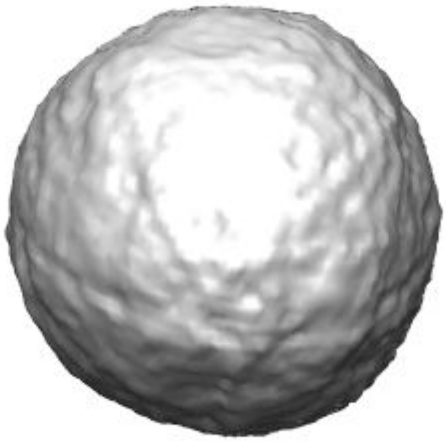
```
    koule.barva(koule.z < 0) = koule.barva(koule.z < 0) + d;
    koule.barva(koule.z >= 0) = koule.barva(koule.z >= 0) - d;
end



 scatter3(koule.x, koule.y, koule.z, 25, koule.barva,'filled');

axis equal;
axis off;
```



Posun bodu na kouli

```
koule.barva = zeros(size(koule.x));
iteraci = 600;
for iter = 1 : iteraci
    pripad = randi([1 3]);
    theta = rand()*2*pi;
    switch (pripad)
        case 1
            AM = [1 0 0 0;
                  0 cos(theta) sin(theta) 0;
                  0 -sin(theta) cos(theta) 0;
                  0 0 0 1];
        case 2
```

```
            AM = [cos(theta) 0 -sin(theta) 0;
                  0  1 0 0;
                  sin(theta) 0   cos(theta) 0;
                  0 0 0 1];
        case 3
            AM = [cos(theta) -sin(theta) 0  0;
                  sin(theta)  cos(theta) 0 0;
                  0 0  1 0;
                  0 0 0 1];
    end
    tform = affine3d(AM);
    [koule.x,koule.y,koule.z] = transformPointsForward(tform,koule.x, koule.y, koule.z);
    d = randn();
    koule.barva(koule.z < 0) = koule.barva(koule.z < 0) + d;
    koule.barva(koule.z >= 0) = koule.barva(koule.z >= 0) - d;
end


V =[koule.x, koule.y,koule.z] + (koule.barva/10).*([koule.x, koule.y,koule.z]./norm([koule.x, k

koule2 = koule;
koule2.x = V(:,1);
koule2.y = V(:,2);
koule2.z = V(:,3);

trimesh(koule2.face, koule2.x, koule2.y, koule2.z,'FaceColor', [0.5 0.5 0.5], 'EdgeColor', 'nor
set(gca, 'color', 'k');
lighting gouraud;
a1 = camlight('right');
a2 = camlight('left');
a3 = camlight('headlight');

axis equal;
axis off;
```



```
TR = triangulation(koule2.face,[koule2.x, koule2.y, koule2.z]);
stlwrite(TR,'planeta.stl');
```

Ohňostroj

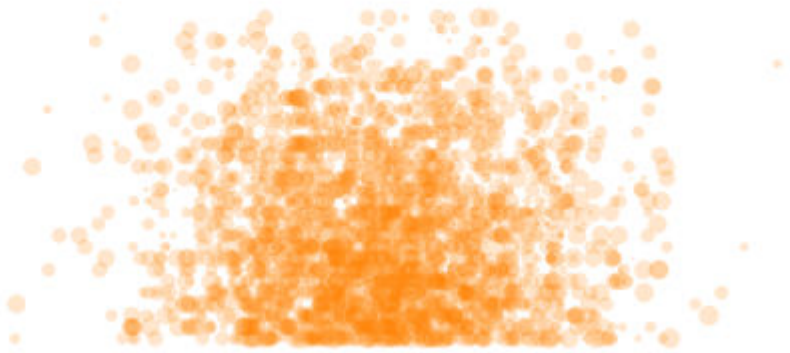
```
pocet = 200 ;
pocatek_x = 0;
pocatek_y = 0;

castice.zivotnost = randi([10,30],[pocet,1]);
castice.deltax = (rand([pocet,1])-0.5);
castice.deltay = (rand([pocet,1])-0.5);
castice.x = pocatek_x*ones([pocet,1]);
castice.y = pocatek_y*ones([pocet,1]);
castice.barva = 0.5*ones([pocet,3]);
%castice.barva = 0.5*ones([pocet,3]) + repmat((rand([pocet,1])-0.5)/5,1,3);
%castice.barva = rand([pocet,3]);
castice.velikost = 30*ones([pocet,1]);
%castice.velikost = randi([10,50],[pocet,1]);

tvar = "*";
%tvar = ".";

figure1 = figure;
```

```
for i = 1 : 35
    castice.x = castice.x + castice.deltax;
    castice.y = castice.y + castice.deltay;
    castice.zivotnost = castice.zivotnost - 1;

    % odstraneni
    castice.x(castice.zivotnost == 0) = [];
    castice.deltax(castice.zivotnost == 0) = [];
    castice.y(castice.zivotnost == 0) = [];
    castice.deltay(castice.zivotnost == 0) = [];
    castice.barva(castice.zivotnost == 0,:) = [];
    castice.velikost(castice.zivotnost == 0,:) = [];
    castice.zivotnost(castice.zivotnost == 0) = [];


    scatter(castice.x, castice.y, castice.velikost, castice.barva, tvar);
    xlim([-30,30]);
    ylim([-30,30]);
    axis off
    %saveas(figure1,"firework1_" + num2str(i) + ".png");
    pause(0.1);
end
```

```
clear castice
```

```
I = perlin2D(20,20)*pi - pi/2;
% na zaklade tohoto vyrobime vektory s uhlem
deltax = cos(I);
deltay = sin(I);

imshow(I,[]);
```

```
pocet = 150 ;
castice.zivotnost = randi([5,30],[pocet,1]);
castice.y = randn([pocet,1])*10+20;
castice.x = zeros([pocet,1]);
castice.velikost = floor(60*rand([pocet,1]))+1;

figure1 = figure;
for i = 1 : 100
    castice.zivotnost = [castice.zivotnost; randi([5,30],[pocet,1])];
    castice.y = [castice.y; randn([pocet,1])*10+20];
    castice.x = [castice.x; zeros([pocet,1])];
    castice.velikost = [castice.velikost;floor(60*rand([pocet,1]))+1];

    % zmena dle perlin noise
    castice.x = castice.x + deltax(floor(castice.x)+35)/100;
    castice.y = castice.y + deltay(floor(castice.y)+35)/50;
    castice.zivotnost = castice.zivotnost - 1;

    % odstraneni
    castice.x(castice.zivotnost == 0) = [];
    castice.y(castice.zivotnost == 0) = [];
    castice.velikost(castice.zivotnost == 0) = [];
    castice.zivotnost(castice.zivotnost == 0) = [];


    scatter(castice.y, castice.x, castice.velikost, [1 0.5 0],"filled",'MarkerFaceAlpha',.2,'Ma
    ylim([0,0.6]);
    xlim([-20,60]);
    axis off

    %saveas(figure1,"fire1_" + num2str(i) + ".png");
    pause(0.1);
end
```

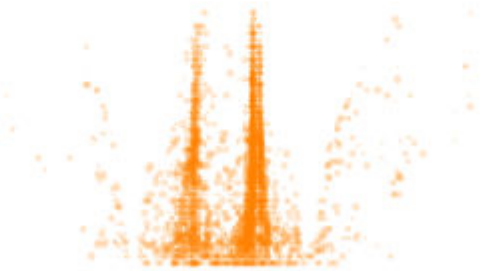


Hustší síť šumu

```
clear castice
I = perlin2D(100,100)*pi - pi/2;
% na zaklade tohoto vyrobime vektory s uhlem
deltax = cos(I);
deltay = sin(I);
```

```
pocet = 150 ;
castice.zivotnost = randi([5,50],[pocet,1]);
castice.y = (randn([pocet,1])*5+50);
castice.x = zeros([pocet,1]);
castice.velikost = floor(10*rand([pocet,1]))+1;

figure1 = figure;
for i = 1 : 100
    castice.zivotnost = [castice.zivotnost; randi([5,50],[pocet,1])];
    castice.y = [castice.y; randn([pocet,1])*5 + 50];
    castice.x = [castice.x; zeros([pocet,1])];
    castice.velikost = [castice.velikost;floor(10*rand([pocet,1]))+1];

    % zmena dle perlin noise
    castice.x = castice.x + deltax(floor(castice.x)+1);
```

```
    castice.y = castice.y + deltay(floor(castice.y));
    castice.zivotnost = castice.zivotnost - 1;

    % odstraneni
    castice.x(castice.zivotnost == 0) = [];
    castice.y(castice.zivotnost == 0) = [];
    castice.velikost(castice.zivotnost == 0) = [];
    castice.zivotnost(castice.zivotnost == 0) = [];

    castice.x(castice.y < 1 | castice.y > 100) = [];
    castice.velikost(castice.y < 1 | castice.y > 100) = [];
    castice.zivotnost(castice.y < 1 | castice.y > 100) = [];
    castice.y(castice.y < 1 | castice.y > 100) = [];


    scatter(castice.y, castice.x, castice.velikost, [1 0.5 0],"filled",'MarkerFaceAlpha',.2,'Ma
    ylim([0,100]);
    xlim([0,100]);
   axis off

    %saveas(figure1,"fire1_" + num2str(i) + ".png");
    pause(0.1);
end
```



Pomocné funkce

```
function [T, X] = nahodnePresouvani(T0in,X0in,T1in,X1in,iterace,maxIterace, H)
    if (iterace <= maxIterace)
        Tpul = (T1in + T0in)/2;
        d = H*normrnd(0,1)*(abs(T0in-T1in));
        Xpul = (X1in + X0in)/2 + d;
        [T0,X0] = nahodnePresouvani(T0in, X0in, Tpul, Xpul, iterace + 1, maxIterace, H);
        [T1,X1] = nahodnePresouvani(Tpul, Xpul, T1in, X1in,  iterace + 1, maxIterace, H);
        T = [T0 T1];
        X = [X0 X1];
    else
        T = [T0in T1in];
        X = [X0in X1in];
    end
end

function [T, X] = nahodnePresouvaniOstrov(T0in,X0in,T1in,X1in,iterace,maxIterace, H)
    if (iterace <= maxIterace)
        Tpul = (T1in + T0in)/2;
        d = H*normrnd(0,0.1)*(abs(T0in-T1in));  %generovani mensiho cisla
        Xpul = (X1in + X0in)/2 + d;
        [T0,X0] = nahodnePresouvani(T0in, X0in, Tpul, Xpul, iterace + 1, maxIterace, H);
        [T1,X1] = nahodnePresouvani(Tpul, Xpul, T1in, X1in,  iterace + 1, maxIterace, H);
        T = [T0 T1];
        X = [X0 X1];
    else
        T = [T0in T1in];
        X = [X0in X1in];
    end
end

function []=koch(n)
    if (n>=0)
       level = 10^n;
       limit=ceil(level/(3^n));
       usecka(0,0,level,0,limit);
       axis equal
       axis off
    end
end

function []=koch_vlocka(n)
    if (n>=0)
       level = 10^n;
       limit=ceil(level/(3^n));
       usecka(0,0,level,0,limit);
       usecka(level,0,(level)/2,-((level)/2)*sqrt(3),limit);
       usecka((level)/2,-((level)/2)*sqrt(3),0,0,limit);
       axis equal
       axis off
    end
```

```
end



function usecka(x1,y1,x5,y5,limit)
    delka=sqrt((x5-x1)^2+(y5-y1)^2);
    if(delka>limit)
       x2=(2*x1+x5)/3;
       y2=(2*y1+y5)/3;
       x3=(x1+x5)/2-(y5-y1)/(2.0*sqrt(3.0));
       y3=(y1+y5)/2+(x5-x1)/(2.0*sqrt(3.0));
       x4=(2*x5+x1)/3;
       y4=(2*y5+y1)/3;

       usecka(x1,y1,x2,y2,limit);
       usecka(x2,y2,x3,y3,limit);
       usecka(x3,y3,x4,y4,limit);
       usecka(x4,y4,x5,y5,limit);
    else
       line([x1;x5],[y1;y5]);
    end
end


function []=hilbert(n)
    A = zeros(0,2);
    B = zeros(0,2);
    C = zeros(0,2);
    D = zeros(0,2);

    a = [ 0  1];
    b  = [ 1  0];
    c = [ 0 -1];
    d  = [-1  0];

    for i = 1:n
      AA = [B ; a ; A ; b  ; A ; c ; C];
      BB = [A ; b  ; B ; a ; B ; d  ; D];
      CC = [D ; d  ; C ; c ; C ; b  ; A];
      DD = [C ; c ; D ; d  ; D ; a ; B];

      A = AA;
      B = BB;
      C = CC;
      D = DD;
    end

    A = [0 0; cumsum(A)];

    plot(A(:,1), A(:,2), 'clipping', 'off')
    axis equal, axis off
end

function M = nahodne_presouvani3D(vstup, H, maxIter)
```

```
    M = vstup;
    range=1;

%    subplot(1,maxIter,1), surf(M);
%    colormap gray;

    for iter=2:maxIter
        M = diamond_square_algorithm(M,range);
        range = range*H*2^(-H);
%        subplot(1,maxIter,iter), surf(M);
%        colormap gray;
    end
end

function y = diamond_square_algorithm(x,range)
    x = okrajx(x);
    y = zvetsi(x);
    y = stredctverce(y,range);
    y = wrap(y);
    y = streddiamond(y,range);
    y=y(2:end-1,2:end-1);
end

function y = okrajx(x)
    y = zeros(size(x)+2);
    y(2:end-1,2:end-1)=x;
end

function y = zvetsi(x)
    n = size(x,1);
    y = zeros(2*n-3,2*n-3);
    for i=2:n-1
        for j=2:n-1
            y(2*i-2,2*j-2)=x(i,j);
        end
    end
end

function y = stredctverce(x,range)
n=size(x,1);
y = x;
for i=3:2:n-2
    for j=3:2:n-2
        y(i,j) = (y(i-1,j-1)+y(i+1,j+1)+y(i+1,j-1)+y(i-1,j+1))*0.25 + 2*range*rand-range;
    end
end
end

function y=wrap(x)
    y = x;
    y(:,1)   = x(:,end-2);
    y(:,end) = x(:,3)    ;
    y(1,:)   = x(end-2,:);
    y(end,:) = x(:,3)    ;
```

```
end

function y = streddiamond(x,range)
    n=size(x,1);
    y = x;
    for i=2:1:n-1
        for j=3-mod(i,2):2:n-1
            y(i,j) = (y(i-1,j)+y(i+1,j)+y(i,j-1)+y(i,j+1))*0.25+2*range*rand-range;
        end
    end
end

function M =  DDA(x0,y0,x1,y1,velikost)
    M = randi([0 1]) * ones(velikost);
    dx = abs(x1-x0);
    dy = abs(y1-y0);
    sx = sign(x1-x0);
    sy = sign(y1-y0);
    dley = 0;
    if (dy > dx)
        step = dy;
        dley = 1;
    else
        step = dx;
    end
    x(1) = x0; y(1) = y0; j = 1;
    for i= 0:1:step
        if (x1 == x)&(y1 == y)
            break;
        end
        j = j+1;
        x(j) = x(j-1) + (dx/step)*sx;
        y(j) = y(j-1) + (dy/step)*sy;
    end
    for k = 1 : size(x,2)
        if dley
            for sx = 1 : round(x(k))
                if(round(x(k)) > 0 && round(y(k)))
                    M(sx, round(y(k))) = not(M(sx, round(y(k))));
                end

            end
        else
            for sy = 1 : round(y(k))
                if(round(x(k)) > 0 && round(y(k)))
                    M(round(x(k)),sy) = not(M(round(x(k)),sy));
                end

            end
        end


    end
end
```

```
function s = perlin2D(m,n)
  s = zeros([m,n]);
  w = m;
  i = 0;
  while w > 3
    i = i + 1;
    d = interp2(randn([m,n]), i-1, 'spline');
    s = s + i * d(1:m, 1:n);
    w = w - ceil(w/2 - 1);
  end
  s = (s - min(min(s(:,:)))) ./ (max(max(s(:,:))) - min(min(s(:,:))));
end
```