**Geostatistika 8**

## Otázky

Pracuji-li v MATLAB s náhodnými čísly, tak pokud nastavím před příklady stejný *seed*, pak bys nad měla být sekvence reprodukovatelná tzn. dostanu stený čísla jako někdo přede mnou. Je to pravda a správný postup?

Zdá se mi, že některé funkce, které dělají graf uvnitř cyklu běží hrozně dlouho pro velká n, např randwalk. Často se mi úplně zasekne MATLAB. Může to být tím?

Trochu jsem bojoval s MATLABEM, zkusil jsem ale náhodnou procházku (symetrickou i nesymetrickou v 1D zatím) v Pythonu. Přikládám nakonec.

## Úkol 1. Rovnoměrné rozdělení

Náhodná proměnná X je rovnoměrné rozdělená, pokud každý hodnota této náhodné proměnné má stejnou pravděpodobnost, že nastane při opakování pokusu. Z histogramů je vidět, že pro zvyšující se počet pokusů, se četnosti v jednotlivých jednotlivých náhodně generovaných čísel přibližují. Příklad: Pokud bychom házeli kostkou, tak pro diskrétní náhodnou proměnnou *padnutí čísla* s možnými hodnotami 1, 2, 3, 4, 5, 6, se bude spolu s rostoucí hodnotou n frekvence padnutí každého z čísel blížit stejné hodnotě.

seed = 3;

rng(seed);

xs = zeros(100);

Definovak jsem tuto funkci v samostatném souboru, kterou používám pro vykreslení.

function [] = randomPoints1DUniform(n, seed)

rng(seed)

xs = rand(1, n);

figure();

plot(xs);

title(sprintf('Random sequence of %d variables', n));

figure();

histogram(xs);

title(sprintf('Random sequence of %d variables', n));

end

Pokus pro .

n = 100;

% 1. náhodný výběr pro n = 100.

randomPoints1DUniform(n, 3);

% 2. náhodný výběr pro n = 100.

randomPoints1DUniform(n, 3);

% 3. náhodný výběr pro n = 100.

randomPoints1DUniform(n, 3);

Pokus pro .

n = 1000;

% 1. náhodný výběr pro n = 1000.

randomPoints1DUniform(n, 3);

% 2. náhodný výběr pro n = 1000.

randomPoints1DUniform(n, 3);

% 3. náhodný výběr pro n = 1000.

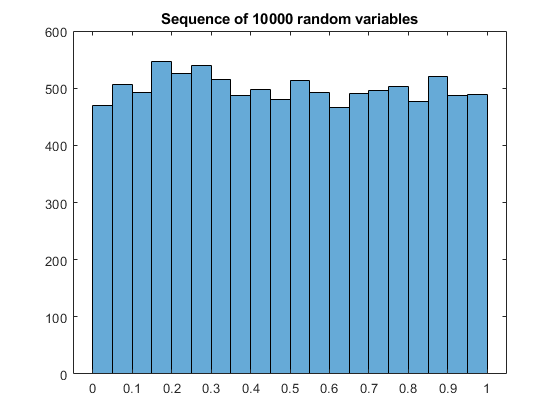
randomPoints1DUniform(n, 3);

Pokus pro .

n = 10000;

% 1. náhodný výběr pro n = 10000.

randomPoints1DUniform(n, 3);



## Úkol 2. Normální rozdělení

Normálni rozdělení je se označuje jako , kde  a  jsou parametry Gaussovy distrtibuční funkce. Pokud je náhodná proměnná ditribuována podle této funkce, pak o ní říkáme že je normálně rozdělená, proto to . Pro různé volby  a  dostaneme různé tvary této funkce.

**A)**

for n = [10, 100, 10000]

x = randn(1, n);

sprintf('n=%d mean=%f', n, mean(x));

sprintf('n=%d var=%f' , n, var(x));

sprintf('n=%d std=%f' , n, std(x));

end

**B)**

m = 175;

s = 7;

for n = [10, 100, 10000]

v = m + s \* randn(1, 100);

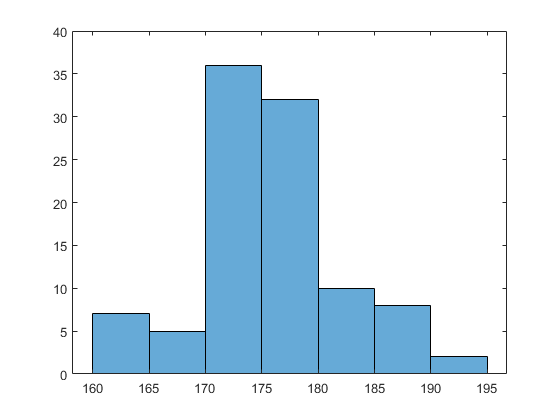
sprintf('n=%d mean=%f', n, mean(v));

sprintf('n=%d var=%f' , n, var(v));

sprintf('n=%d std=%f' , n, std(v));

histogram(v);

end



## Úkol 3. Kovariance nezávislých veličin

Pro rostoucí N se bude kovariance nezávislých proměnných blížit více a více požadované nule.

x = randn(1, 10);

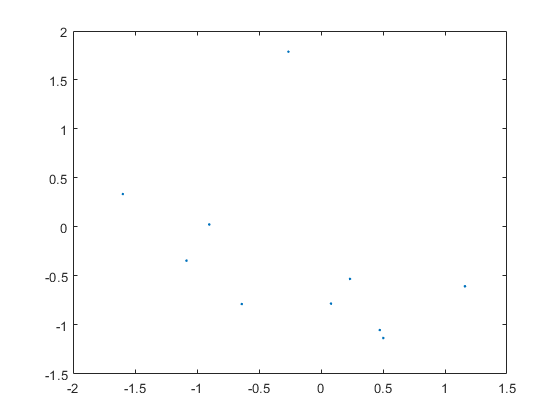
y = randn(1, 10);

covMatrix = cov(x, y);

sprintf('covarince %f', covMatrix(1, 2))

ans = 'covarince -0.304376'

plot(x, y, '.');



x=randn(1, 1000);

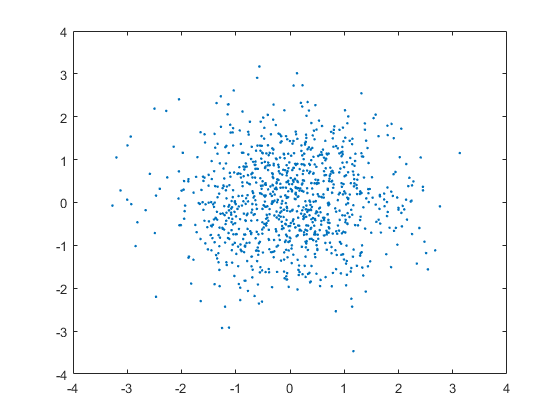
y=randn(1, 1000);

covMatrix = cov(x, y);

sprintf('covarince %f', covMatrix(1, 2))

ans = 'covarince 0.018263'

plot(x, y, '.');



## Úkol 4. Kovariance závislých veličin

x = randn(1, 1000); % [m]

y = randn(1, 1000); % [kg]

z = x + y

z = 1×1000

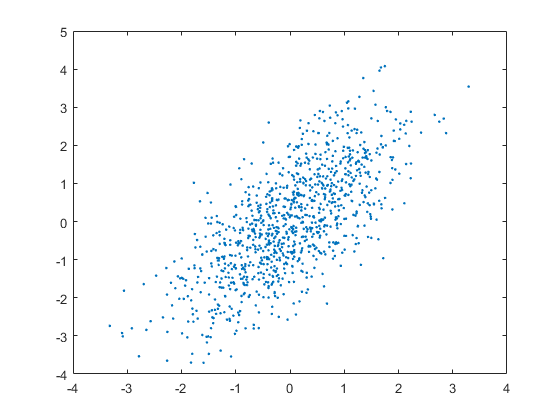
2.1656 -0.5471 -1.5403 0.4392 0.5773 -0.7948 -0.3147 ⋯

covMatrix = cov(x, z);

sprintf('covarince %f', covMatrix(1, 2))

ans = 'covarince 1.007183'

plot(x, z, '.');



Jednotky kovariance jsou  tedy např. . Při zmenšené/zvětšení (např. m na cm) jedné z jednotek se výsledná jednotka změní poměrně např. stokrát (1m = 100cm). Jak uvidíme dochází k deformaci kruhu/elipsy.

x = randn(1, 1000) \* 100; % [cm]

y = randn(1, 1000); % [kg]

z = x + y

z = 1×1000

44.5932 28.4645 106.0858 114.5821 -107.6622 26.7281 62.6540 ⋯

covMatrix = cov(x, z);

sprintf('covarince %f', covMatrix(1, 2))

ans = 'covarince 10471.624341'

plot(x, z, '.');

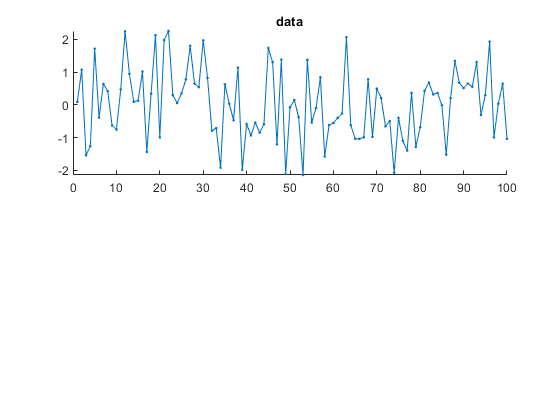
## Úkol 5 + 6. Doplním, nějak nerozumím asi zadání.

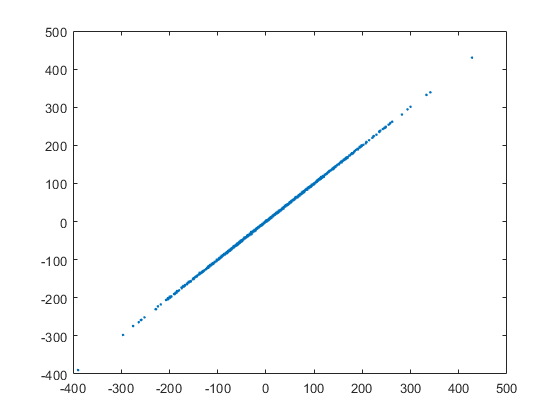
* Kontrolní otázka 1: Data zs jsme vyrobili jako vážený průměr z náhodné sekvence. Jak souvisí délka klesání kovariance u počátku s počtem průměrovaných datových hodnot (s počtem vah)?
* Kontrolní otázka 2: Proč je na 9. řádku wma příkaz zs=mean(z)\*ones(1,n);? **Vytvoříme si vektor o**  **prvcích obsahující** .
* Kontrolní otázka 3: Jak lze pomocí vhodné volby vah napodobit obr. 6.7 dole?
* Kontrolní otázka 4: Data zs jsme vyrobili jako vážený průměr z náhodné sekvence. Jak souvisí délka stoupání variogramu u počátku s počtem průměrovaných datových hodnot (s počtem vah)

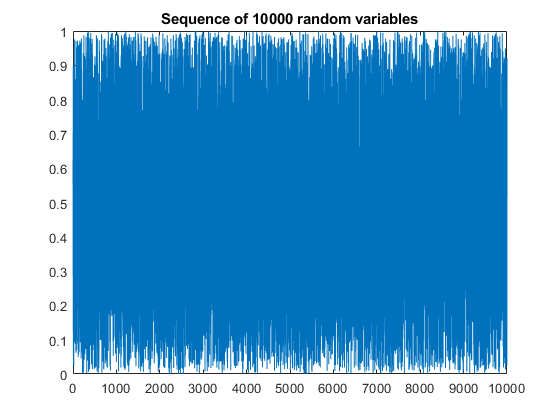
z = randn(1, 100);

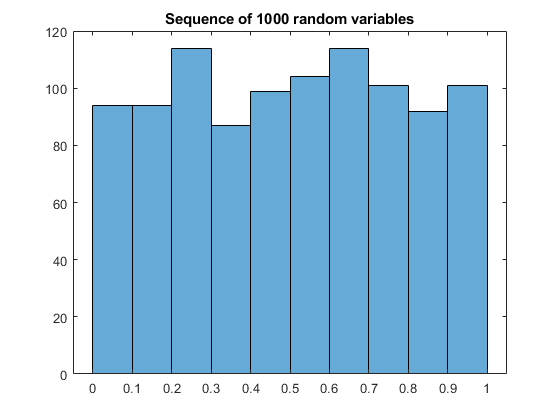
c = kovariance(z, 30);

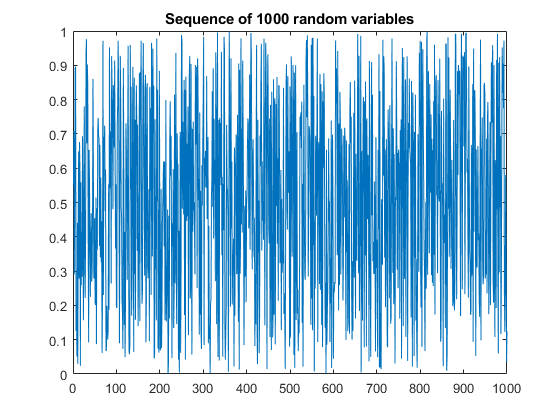
n = 100

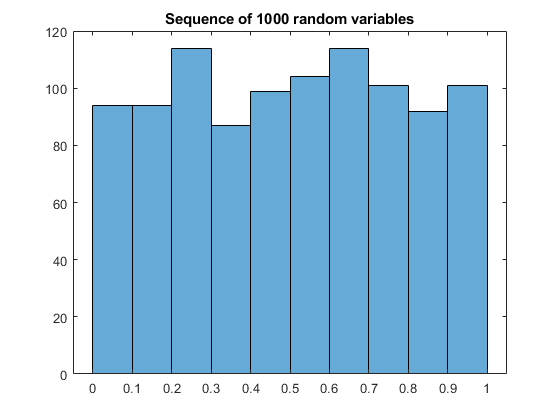


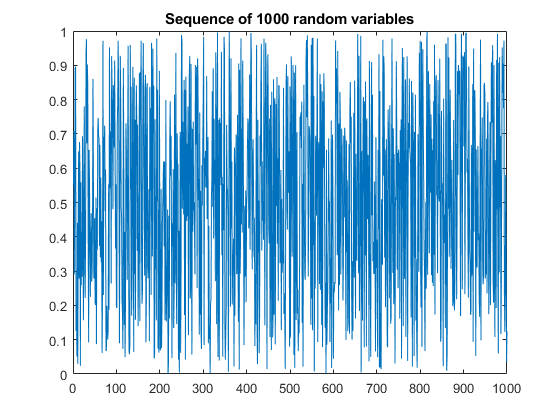


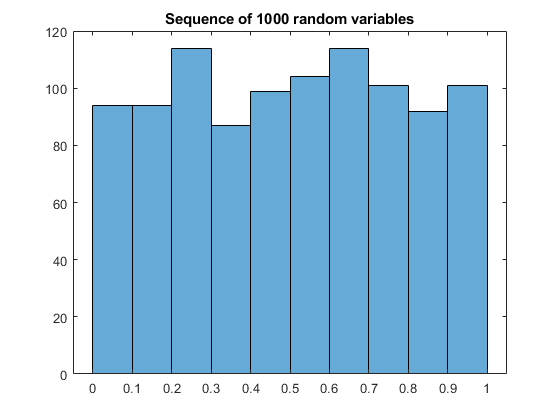


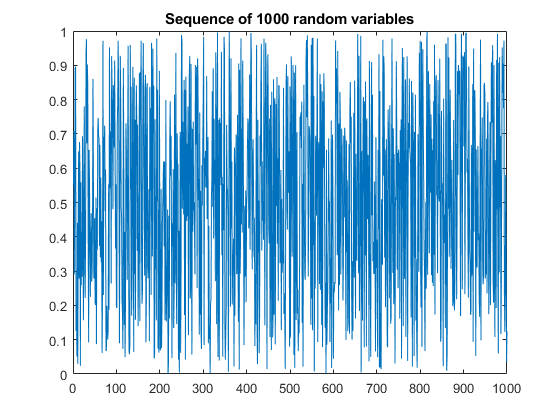


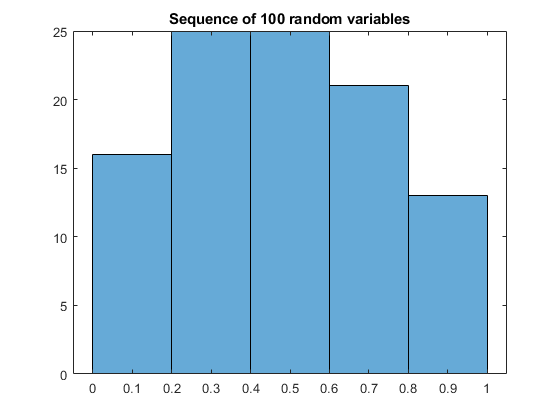


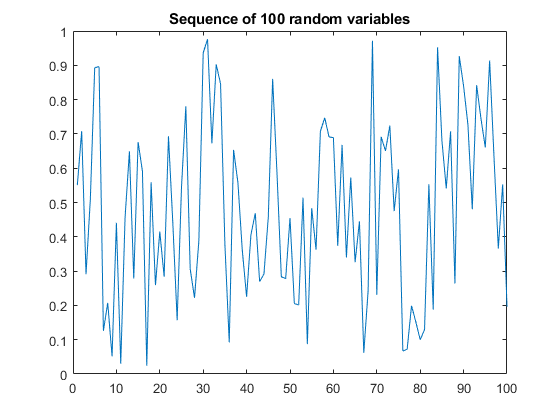


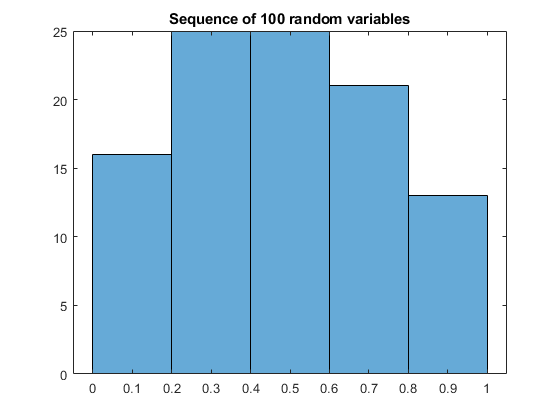


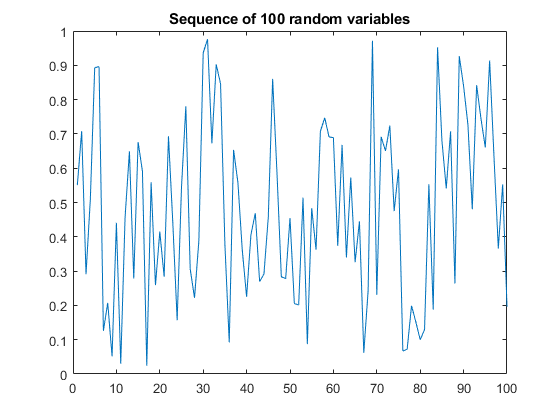


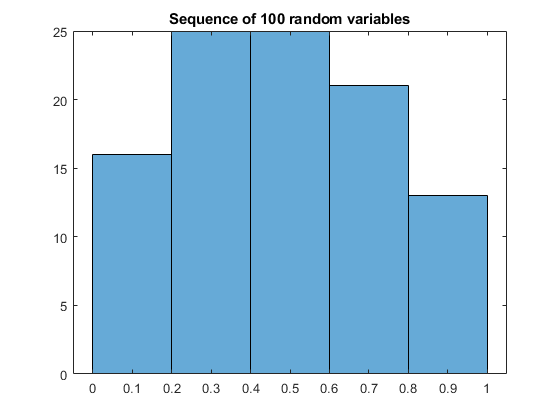


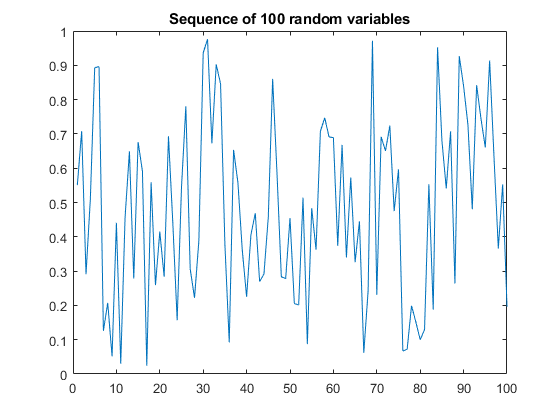












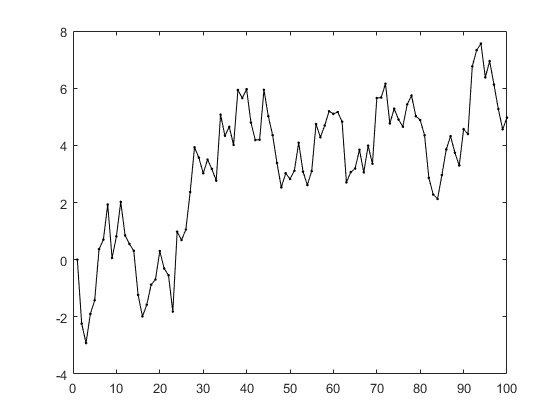
zs = wma(z,[1 1 1] / 3);

c = kovariance(zs, 30);

## Úkol 7. Náhodná procházka

**1D Random Walk**

randomWalk1D(100);



**2D Randowm Walk**

function z = randomWalk2D(steps)

figure;

hold on;

z(1)=0;

y(1)=0;

for i = 2:steps

z(i) = z(i-1) + randn;

y(i) = y(i-1) + randn;

plot([z(i-1) z(i)], [y(i-1) y(i)], 'k.-');

hold on;

end

axis square;

end

randomWalk2D(100);

