

1. domača naloga pri predmetu napredna računalniška orodja

Aljaž Luznar

Univerza v Ljubljani Fakulteta za strojništvo

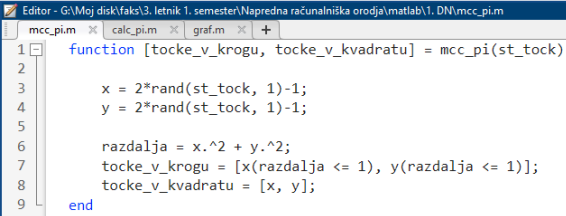
October 22, 2023

Kazalo

1. Predstavitev Monte Carlo metode za izračun vrednosti π
2. Rezultati izračuna
3. Git
4. Beamer

Predstavitev Monte Carlo metode za izračun vrednosti π

- ▶ Ustvaril sem funkcijsko datoteko `mcc_pi.m`, ki ima kot vhodni parameter število točk, vrne pa nam pa koordinate točk znotraj kroga in koordinate točk znotraj kvadrata.



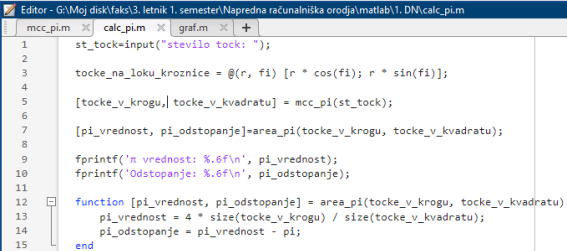
```
Editor - G:\Moj disk\faks\3. letnik 1. semester\Napredna računalniška orodja\matlab\1. DN\mcc_pi.m
mcc_pi.m x calc_pi.m x graf.m x +
1 function [tocke_v_krogu, tocke_v_kvadratu] = mcc_pi(st_tock)
2
3     x = 2*rand(st_tock, 1)-1;
4     y = 2*rand(st_tock, 1)-1;
5
6     razdalja = x.^2 + y.^2;
7     tocke_v_krogu = [x(razdalja <= 1), y(razdalja <= 1)];
8     tocke_v_kvadratu = [x, y];
9 end
```

Figure: koda za funkcijo `mcc_pi.m`



Predstavitev Monte Carlo metode za izračun vrednosti π

- Nato sem ustvaril programsko datoteko calc_pi.m. Datoteka vsebuje funkcijo area_pi.m. Funkcija izračuna število π na podlagi Monte Carlo metode in vrne napako izračuna.



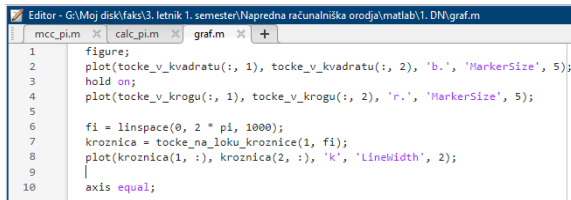
```
1 st_tock=input("stevalo tock: ");
2
3 tocke_na_loku_kroznice = @(r, fi) [r * cos(fi); r * sin(fi)];
4
5 [tocke_v_krogu, tocke_v_kvadratu] = mcc_pi(st_tock);
6
7 [pi_vrednost, pi_odstopanje]=area_pi(tocke_v_krogu, tocke_v_kvadratu);
8
9 fprintf('pi vrednost: %.6f\n', pi_vrednost);
10 fprintf('Odstopanje: %.6f\n', pi_odstopanje);
11
12 function [pi_vrednost, pi_odstopanje] = area_pi(tocke_v_krogu, tocke_v_kvadratu)
13     pi_vrednost = 4 * size(tocke_v_krogu) / size(tocke_v_kvadratu);
14     pi_odstopanje = pi_vrednost - pi;
15 end
```

Figure: koda za funkcijo calc_pi.m



Rezultati izračuna

- Rezultate prikažemo s pomočjo programa na spodnjo sliki.



```
Editor - G:\Moj disk\fax\3. letnik 1. semester\Napredna računalniška orodja\matlab\1. DN\graf.m
mcc_pi.m x calc_pi.m x graf.m x +
1 figure;
2 plot(tocke_v_kvadratu(:, 1), tocke_v_kvadratu(:, 2), 'b.', 'MarkerSize', 5);
3 hold on;
4 plot(tocke_v_krogu(:, 1), tocke_v_krogu(:, 2), 'r.', 'MarkerSize', 5);
5
6 fi = linspace(0, 2 * pi, 1000);
7 kroznica = tocke_na_loku_kroznice(1, fi);
8 plot(kroznica(1, :), kroznica(2, :), 'k', 'LineWidth', 2);
9 |
10 axis equal;
```

Figure: koda za prikaz rezultatov



Rezultati izračuna

- Pri vrednosti 1000 točk dobimo naslednji graf (verzija kode grafa z označenimi osmi je naložena v mojem GitHubu)

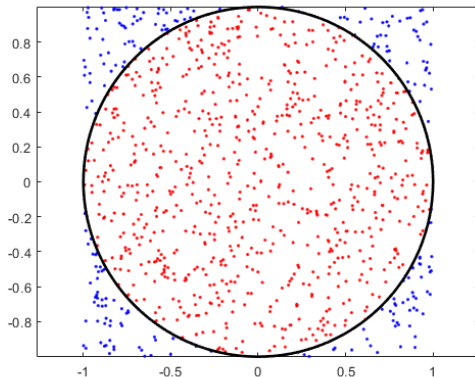


Figure: koda za prikaz rezultatov



Rezultati izračuna

- ▶ S spreminjanjem števila točk dobimo naslenje rezultate in odstopke. Ugotovimo da se z povečevanjem števila točk večja natančnost izračuna.



Rezultati izračuna

- ▶ S spreminjanjem števila točk dobimo naslenje rezultate in odstopke. Ugotovimo da se z povečevanjem števila točk večja natančnost izračuna.
- ▶ 10 točk: 3.615385 napaka: 0.473792
100 točk: 2.880448 napaka: -0.261145
1000 točk: 3.080004 napaka: -0.061589
10000 točk: 3.162000 napaka: 0.020407



- ▶ Vse datoteke uporabljene v tej nalogi sem naložil na svoj Git profil.



- ▶ Za del naloge ki zahteva uporabo orodja Beamer sem izdelal to predstavitev.

