sem1_Roškarič

2022-10-31

1. Uvod

Namen te seminarske naloge je, da s pomočjo integriranja z Monte Carlo metodo in gostote normalne porazdelitve ocenimo vrednost števila pi in s tem dobimo dober približek. Gre torej za numerično metodo, s katero simuliramo točke na nekem območju in izračunamo delež takšnih točk, ki ležijo znotraj ploščine, s katero se ukvarjamo.

1.1 Predpostavka uporabljene metode

Naše ocenjevanje temelji na naslednji predpostavki:

$$\int_{-\infty}^{\infty} e^{-\lambda x^2} dx = \sqrt{\frac{\pi}{\lambda}} \qquad ; \lambda > 0$$

Kot dokaz za zgornjo formulo smo vzeli gostoto normalne porazdelitve in jo dali pod določen integral od minus neskončno do neskončno. Izraz je potemtakem bil sledeč:

$$\int_{-\infty}^{\infty} \frac{1}{\sigma\sqrt{2\pi}} * e^{\frac{-1}{2}(\frac{x-\mu}{\sigma})^2} dx = 1$$

Zgornja enačba nam predstavlja ploščino pod celotno krivuljo gostote normalne porazdelitve. Skupna ploščina je seveda ena.

Da smo imeli znane parametre mi in sigma ter s tem odpravili nekaj dodatnih spremenljivk, smo si izbrali standardno normalno porazdelitev, torej X ~ N(0,1). Podatke smo vstavili v formulo in dobili sledeč izraz:

$$\int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi}} * e^{\frac{-1}{2}x^2} dx = 1$$

Nato smo morali izraziti pi. Izraz smo pomnožili z imenovalcem ulomka (torej koren iz 2 pi).

$$\int_{-\infty}^{\infty} e^{\frac{-1}{2}x^2} dx = \sqrt{2\pi}$$

Na tej točki smo lahko ugotovili, da prvotno omenjena formula drži. Naša lambda v tem primeru je bila namreč 1/2. Pod korenom pa se je zgodilo sledeče:

$$\int_{-\infty}^{\infty} e^{\frac{-1}{2}x^2} dx = \sqrt{\frac{\pi}{\frac{1}{2}}}$$

Ko smo izraz postavili v takšno obliko, smo opazili, da naša prvotna predpostavka velja. V tem konkretnem primeru je lambda bila 1/2, vendar dokaz velja za vsako lambdo, ki je večja od 0.

Na podlagi veljavnih predpostavk smo začeli s pripravami za omenjeno numerično metodo.

2. Določanje pozicije točk

Prvo smo ustvarili funkcijo pod_grafom, s katero smo preverili, ali posamezna točka leži nad oziroma pod grafom sledeče funkcije:

$$f(x) = e^{-\lambda x^2}$$

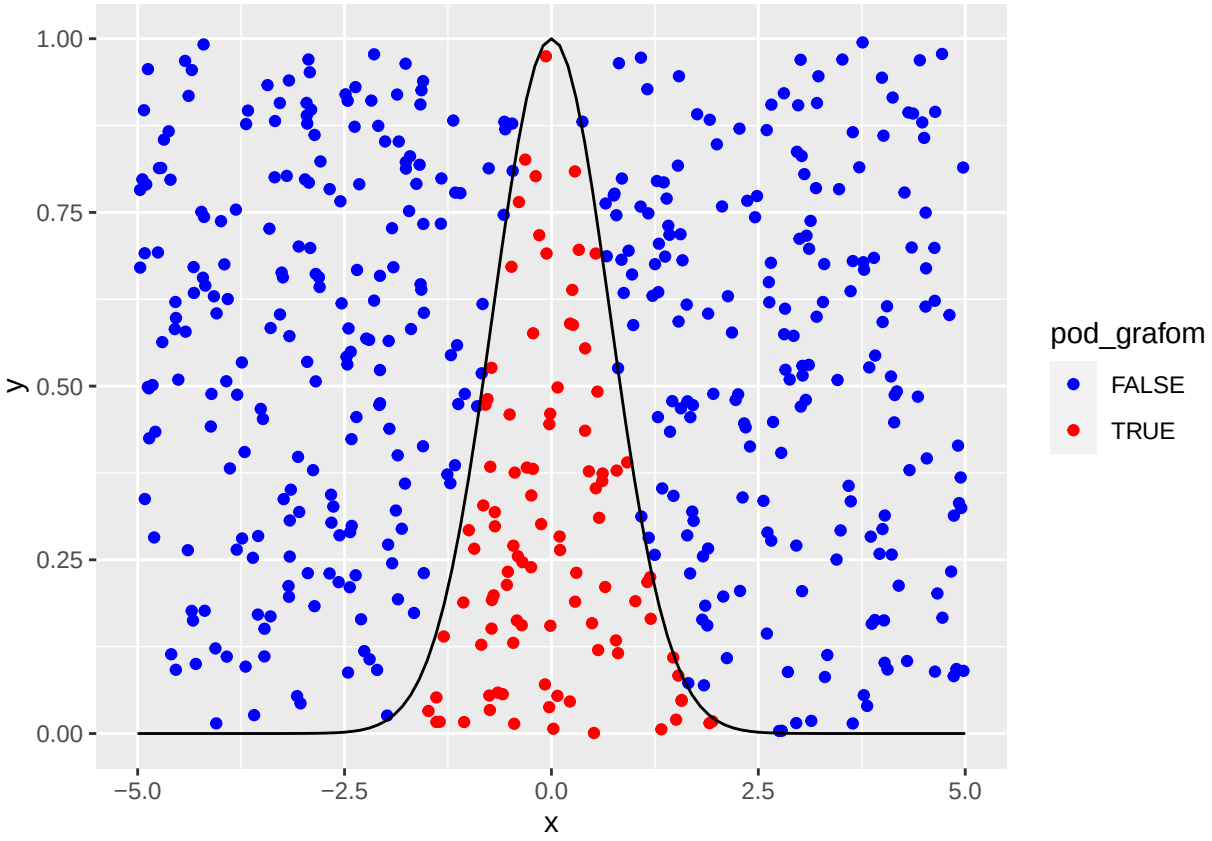
```
set.seed(70)

# Preverimo, ali točka leži pod krivuljo
pod_grafom = function(x, y, lambda){
  funkcija = exp(1)^(-lambda*x^2)
  # Če je vrednost funkcije večja od y koordinate točke, je točka spodaj
  if (funkcija > y){
    return(TRUE)
  } else{
    return(FALSE)
  }
}

# Privzeta vrednost lambde naj bo 1
lambda = 1

# Preizkus funkcije
pod_grafom(-0.5,0.25,lambda)
```

```
## [1] TRUE
```

Da smo testirali zgornjo funkcijo, smo ustvarili 500 točk na intervalu [-5,5] X [0,1], jih vstavili v funkcijo pod_grafom() ter izrisali rezultat.

```
x = runif(500, -5 ,5)
y = runif(500)

# Vektor logičnih vrednosti za pozicijo točk
true_false = c()
for (i in 1:length(x)){
  true_false = append(true_false,pod_grafom(x[i],y[i],lambda))
}

# Podatkovni okvir za risanje
df = data.frame(x,y,true_false)

# Funkcijo graf() potrebujemo kot parameter v stat_function()
graf = function(lambda,x){
  graf = exp(1)^(-lambda*x^2)
  return(graf)
}

ggplot(data=df,aes(x,y)) +

  geom_point(aes(col=true_false)) +
```

```
# Izris krivulje
stat_function(fun=graf, args = list(lambda=lambda)) +

xlim(-5,5) +

scale_colour_manual(values=c("blue","red")) +

labs(col="pod_grafom")
```



3. Ocenjevanje števila pi

Naslednji korak je bil, da smo ustvarili funkcijo oceni(), ki je na podoben način na območju [-5,5] X [0,1] ustvarila n točk in izračunala delež teh točk, ki so se nahajale pod krivuljo. S pomočjo tega podatka smo dobili oceno za število pi.

Pridobljen rezultat je namreč predstavljal oceno deleža ploščine pod našo krivuljo glede na ploščino celotnega delovnega območja. Ta delež smo pomnožili s celotno ploščino postavljenega območja, ki je znašala 10*1, torej 10. Tako izračunana cifra je ponazarjala oceno površine pod krivuljo, torej levi del spodnje enačbe.

$$\int_{-5}^{5} e^{-\lambda x^2} = \sqrt{\frac{\pi}{\lambda}}$$

Za lažjo ponazoritev smo na levo stran pisali S kot “surface”.

$$S = \sqrt{\frac{\pi}{\lambda}}$$

Nato smo izrazili še pi in dobili končno formulo.

$$S^2 * \lambda = \pi$$

3.1 Funkcija za oceno števila pi

```
# Funkcija za oceno števila pi
oceni = function(n,lambda){
  x = runif(n, -5 ,5)
  y = runif(n)
  true_false = c()
  for (i in 1:length(x)){
    true_false = append(true_false,pod_grafom(x[i],y[i],lambda))
  }

  df = data.frame(x,y,true_false)

  # Delež TRUE vrednosti (pod krivuljo) med vsemi točkami
  delez = length(true_false[true_false==TRUE])/length(true_false)

  # Pomnožimo z 10, saj je to ploščina celotnega območja, s katerim operiramo
  povrsina = delez*10

  # Iz formule izrazimo pi
  pi = (povrsina^2)*lambda
  return(pi)
}

# Testiranje funkcije
oceni(1000,1)
```

```
## [1] 3.1684
```

3.2 Postavitev parametrov funkcije in ocenjevanje

S pomočjo več različno velikih vzorcev, ki smo jim spreminjali parameter lambda, smo poskušali oceniti pi. Imamo torej dve pomembni spremenljivki: velikost vzorca in lambdo. Za vsako kombinacijo teh parametrov smo izračunali 95% interval zaupanja za povprečje njihovih ocen.

```
# Lambda prvotno nastavimo na 0 (ob začetku prve izeracije ocen se bo to že spremenilo)
lambda = 0

# Vektorja vse_lambde in vsi_n bosta shranjevala vrednosti za lambdo oziroma n
vse_lambde = c()
vsi_n = c()
```

```
df_ocene = data.frame()
st_ocen = 1000
z = 0
for (i in 1:3){
  # Lambdo 3-krat eksponentno povečamo (+4, +16 in +64)
  lambda = lambda + 4^i
  n = 137
  for (i in 1:5){
    # Za vsako lambo n spremenimo 5 krat
    n = n + 150
    vsi_n = append(vsi_n,n)
    vse_lambde = append(vse_lambde,lambda)

    # spremenljivka z določa vrstico, v katero se naj pišejo ocene
    # (gre od 1 do 15)
    z = z + 1
    for (i in 1:st_ocen){
      df_ocene[z,i] = oceni(n,lambda)
    }
  }
}

povprecja = c()
spodnja_meja = c()
zgornja_meja = c()

# Vsaki vrstici dodamo podatek o povprečju ter spodnji in zgornji meji intervala zaupanja

for (i in 1:z){
  povprecja = append(povprecja,mean(as.numeric(df_ocene[i,])))

  spodnja_meja = append(spodnja_meja,povprecja[i] - 1.96*(sd(df_ocene[i,])/sqrt(st_ocen)))

  zgornja_meja = append(zgornja_meja,povprecja[i] + 1.96*(sd(df_ocene[i,])/sqrt(st_ocen)))
}

df_ocene = cbind(df_ocene, vse_lambde, vsi_n, povprecja, spodnja_meja, zgornja_meja)
```

3.3 Povzetek rezultatov ocenjevanja

S pomočjo knjižnice kableExtra smo povzeli rezultate našega ocenjevanja v obliki tabele.

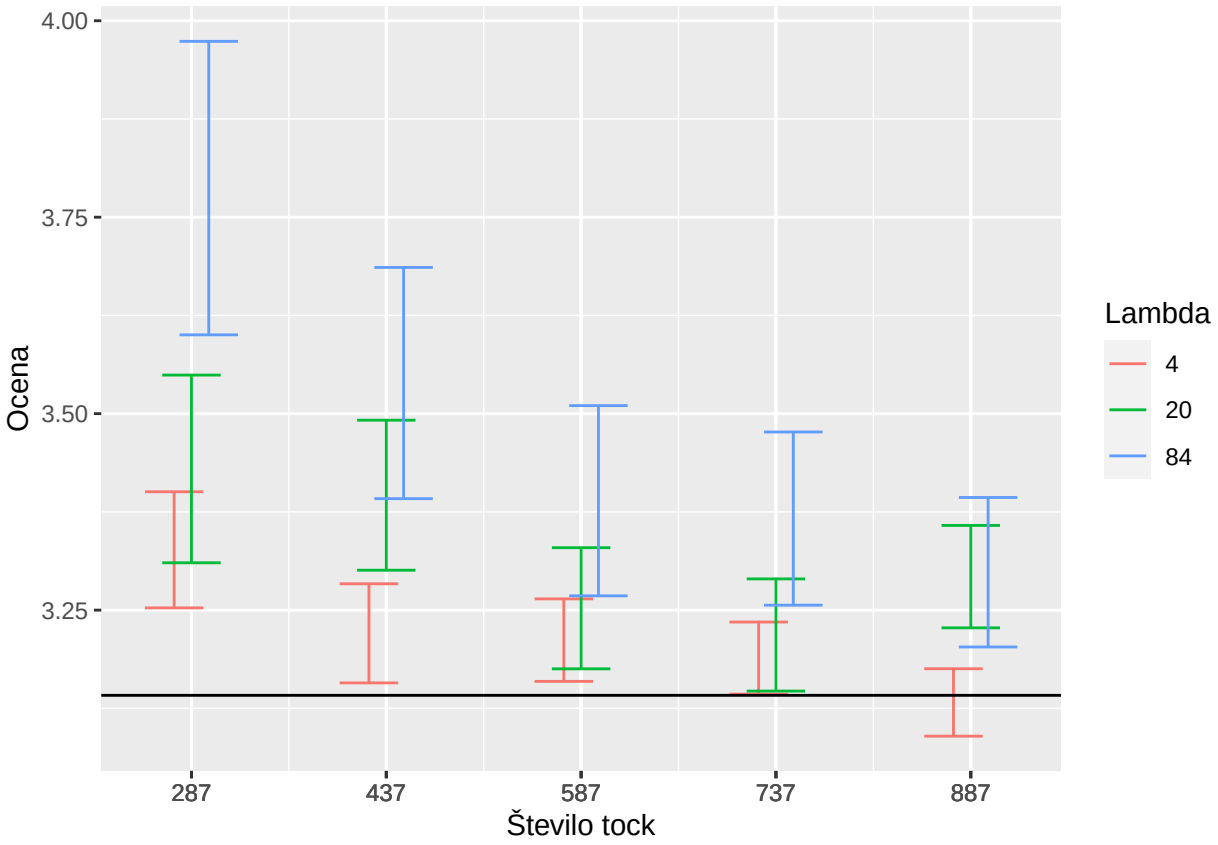
```
# Stolpci st_ocen+1 oz. +5 (1001:1005) vsebujejo zanimive podatke
tabela = df_ocene[,(st_ocen+1):(st_ocen+5)]
names(tabela) = c("Lambda", "Stevilo tock","Povprecje", "Spodnja meja", "Zgornja meja")
kable(tabela,digits=3) %>%
kable_styling()
```

Lambda	Stevilo tock	Povprecje	Spodnja meja	Zgornja meja
4	287	3.327	3.253	3.401
4	437	3.220	3.157	3.284
4	587	3.212	3.159	3.264
4	737	3.189	3.143	3.235
4	887	3.133	3.090	3.175
20	287	3.430	3.310	3.549
20	437	3.396	3.301	3.492
20	587	3.252	3.175	3.329
20	737	3.218	3.147	3.290
20	887	3.293	3.227	3.358
84	287	3.787	3.600	3.974
84	437	3.539	3.392	3.686
84	587	3.389	3.268	3.510
84	737	3.366	3.256	3.477
84	887	3.298	3.203	3.393

3.4 Grafičen prikaz rezultatov

Za konec smo rezultate predstavili tudi na grafičen način. Intervale zaupanja smo narisali za vsak parameter n in lambda. Vodoravna črna črta na vizualizaciji je približek dejanske vrednosti števila pi kot referenčna vrednost. Prikazi intervalov zaupanja so namensko za malenkost zamaknjeni glede na x os v namene preglednosti. Skupina treh intervalov, ki so drug ob drugem, se sklicuje na konkretno vrednost x pod njo.

```
ggplot(data=df_ocene,aes(x=vsi_n,col=factor(vse_lambde)))+

  # Intervali zaupanja
  geom_errorbar(data=df_ocene,

  aes(ymax=zgornja_meja,ymin=spodnja_meja),position = position_dodge(40)) +

  # Vodoravna črta, ki predstavlja dejansko vrednost pi
  geom_hline(yintercept = 3.14159) +

  labs(col="Lambda") +

  ylab("Ocena") +

  scale_x_continuous(breaks=vsi_n) +

  xlab("Število točk")
```



4. Sklep

Na koncu lahko povzamemo glavne ugotovitve. Ocena za pi je natančnejša, ko uporabimo večji n, torej če generiramo večje število točk v namen ocenjevanja, saj je ocena deležev točk pod krivuljo bolj zanesljiva. Z večjim številom opazovanj potemtakem interval zaupanja postaja vse ožji. Glede lambde pa opazimo obraten trend, saj nižja vrednost pomeni ožji interval zaupanja. Večja vrednost lambde namreč povzroči, da je krivulja ožja, zato je manj točk pod njo in je delež seveda tudi manjši. Za izračun ploščine moramo ta delež kvadrirati, kar ga še dodatno zmanjša. Nato pa ploščino pomnožimo z razmeroma velikim številom. S tem dobimo večjo variabilnost v ocenah in širok interval.

Za boljšo oceno števila pi so torej primernejši večji n-ji (večje število točk v vzorcu) in manjše lambde, vendar je potrebno poudariti, da lahko število točk (n) vpliva na časovno zahtevnost algoritma.