UNIVERZA V LJUBLJANI
FAKULTETA ZA MATEMATIKO IN FIZIKO

Finančna matematika – 1. stopnja

Sara Pohl

Metoda Euler-Maruyama in njena ekstrapolacija

Delo diplomskega seminarja

Mentor: prof. dr. Dejan Velušček

Ljubljana, 2014

Kazalo

Metoda Euler-Maruyama in njena ekstrapolacija

Povzetek

V svetu financ se pogosto srečujemo s slučajnimi diferencialnimi enačbami, za katere pa velikokrat ne moremo izračunati eksaktne pričakovane vrednosti. Zato uporabimo numerične metode. Ker za natančen izračun potrebujemo zelo veliko ponovitev simulacije slučajnega procesa, in pa zelo majhne korake, to vzame računalniku veliko časa za izračun primerne natančnosti.
Z uporabo ekstrapolacije pa lahko čas izračunavanja bistveno zmanjšamo.
V diplomskem seminarju sem na nekaj primerih obravnavala konvergenco napake proti 0 v odvisnosti od števila ponovitev, števila korakov in z uporabo ekstrapolacije 1. in 2. reda.

Euler-Maruyama method and its extrapolation

Abstract

In finance, we frequently come across stochastic differential equations, for which we often can not calculate the exact expected value. Therefore we use numerical methods. As we need a lot of repetition of the simulation of the random process and very small steps, it takes a lot of computation time to calculate the value of the appropriate precision.
By using extrapolation, we can significantly reduce the computation time. In the seminar, I have considered the convergence of error towards 0 on several cases, depending on the number of repetitions, number of steps and using extrapolation of the 1st and 2nd order.

Math. Subj. Class. (2014):

Ključne besede: opcija, ekstrapolacija, konvergenca, slučajne diferencialne enačbe, Wienerjev proces

Keywords: option, extrapolation, convergence, stochastic diferential equations, Wiener process

1. Predstavitev osnovnih pojmov

1.1. Opcija.

Opcija je pogodba, pri kateri ima nosilec pogodbe pravico odločiti se, ali bo nakup/prodaja osnovnega premoženja izvršena. Poznamo dve osnovni vrsti opcij - nakupne in prodajne. Če ima lastnik opcije pravico do nakupa osnovnega premoženja, ji pravimo nakupna opcija, če pa ima pravico do prodaje, jo imenujemo prodajna opcija.

Glede na možno izvršitev in izplačilno funkcijo ločimo več vrst opcij. Tiste, katerih izvršitev je mogoča le ob času zapadlosti T, imenujemo evropske opcije. Tiste, katerih izvršitev je mogoča kadarkoli do časa zapadlosti, pa imenujemo ameriške opcije. Glede na izplačilno funkcijo pa ločimo klasične, binarne, azijske, eksotične itd. Jaz bom obravnavala klasične evropske opcije in pa binarne opcije.

Za vrednotenje klasične evropske opcije (*vanilla option*) v času T potrebujemo dva podatka in sicer S_T, ki označuje vrednost osnovnega premoženja v času T, in pa izvršilno ceno K, ki je določena v pogodbi. Nosilec nakupne opcije bo opcijo izvršil le, če je $S_T > K$. Izplačilo nakupne opcije je zato enako: $C_T = max\{S_T - K, 0\}$. Pri prodajni opciji pa je ravno obratno, torej jo bo nosilec izvršil, če bo veljalo $S_T < K$, izplačilo pa bo $P_T = max\{K - S_T, 0\}$.

Binarne opcije (*all-or-nothing options*) so tiste, pri katerih lastnik opcije dobi neko v naprej določeno vrednost K, če vrednost osnovnega premoženja v času T zadošča v pogodbi določenim pogojem. V nasprotnem primeru ne dobi ničesar. Najpogostejši primer take opcije je opcija, pri kateri lastnik trdi, da bo vrednost osnovnega premoženja presegla neko vrednost ali pa ne. Če je njegova napoved pravilna, dobi vrednost K, sicer 0.

Ker pa v času 0 ne poznamo vrednosti osnovnega premoženja za čas T, za vrednotenje opcij poznamo več načinov, ki ceno določijo glede na predvideno vrednost osnovnega premoženja v času T.

1.2. Brownovo gibanje.

Brownovo gibanje je fizikalni pojav, ki ga je odkril Robert Brown med opazovanjem cvetnega prahu v vodi. Drobna zrna so se premikala v naključne smeri.

Najpreprostejši matematični model, ki opisuje enorazsežno Brownovo gibanje, je Wienerjev proces, poimenovan po Norbertu Wienerju. To je slučanji proces W_t, za katerega veljajo naslednje lastnosti:

- $W_0 = 0$
- funkcija $t \mapsto W_t$ je skoraj gotovo povsod zvezna.
- $W_t - W_s \sim N(0, t-s)$ za $0 <= s <= t$, kjer je $N(0, t-s)$ normalna porazdelitev z matematičnim upanjem 0 in varianco $t - s$.
- Za poljubne $0 <= t_1 <= t_2 <= \ldots <= t_n$ velja: $W_{t1}, W_{t2} - W_{t1}, \ldots, W_{tn} - W_{tn-1}$ so neodvisne.

Geometrijsko Brownovo (poznano tudi kot eksponentno Brownovo gibanje) pa je slučajni proces, pri katerem logaritem slučajne spremenljivke sledi Wienerjevemu procesu. Za slučajni proces S_t pravimo, da sledi geometrijskemu Brownovemu gibanju, če zadošča naslednji stohastični diferencialni enačbi:

$$dS_t = \mu S_t dt + \sigma S_t dW_t$$

pri čemer W_t označuje Wienerjev proces.

1.3. **Black-Scholesov model.**

Eden najpogosteje uporabljenih modelov za ocenjevanje vrednosti opcije je Black-Scholesov model. Njegova pomembnost izhaja iz dejstva, da je mogoče izračunati točno sedanjo vrednost opcije.

Model je osnovan na podlagi naslednjih predpostavk:

- Na trgu obstaja netvegana, konstantna obrestna mera v času do zapadlosti.
- Na trgu ni arbitraže.
- Finančni trg je brez trenja (ni stroškov transakcij, davkov, dividend itd.)
- Vrednostni papirjii so neskončno deljivi (lahko kupimo/prodamo poljubnen delež vrednostnega papirja)
- Vrednost tveganega vrednostnega papirja v času je geometrijsko Brownovo gibanje. To pomeni, da vrednost S_t sledi enačbi

$$dS_t = \mu S_t dt + \sigma S_t dW_t \tag{1}$$

  z začetno vrednostjo $S_0 > 0$. W_t predstavlja Wienerjev proces, μ je netvegana obrestna mera (označimo tudi z R), $\sigma > 0$ pa je parameter, ki določa velikost naključnega odmika vrednosti.

Več o tem si bralec lahko prebere v [13].

1.3.1. *Kratka izpeljava fomule.*

(1) Vrednost osnovnega premoženja sledi enačbi:

$$dS_t = \mu S_t dt + \sigma S_t dW_t$$

(2) Z uporabo Itove leme dobimo enačbo za ceno nakupne opcije C, ki je odvisna od vrednosti osnovnega premoženja S in časa t.

$$dC = (\frac{\partial C}{\partial S}\mu S + \frac{\partial C}{\partial t} + \frac{1}{2}\frac{\partial^2 C}{\partial S^2}\sigma^2 S^2)dt + \frac{\partial C}{\partial S}\sigma S dW$$

(3) Naslednji korak je, da konstruiramo portfelj, ki ni odvisen od slučaja. Za to moramo predpostaviti, da na trgu obstaja finančni instrument z netvegano obrestno mero R.

(4) S pomočjo tega portfelja nato pridemo do Black-Scholsove parcialne diferencialne enačbe:

$$\frac{\partial C}{\partial t} + \frac{1}{2}\frac{\partial^2 C}{\partial S^2}\sigma^2 S^2 + R\frac{\partial C}{\partial S}S_t = RC$$

(5) Rešitev te enačbe pa predstavlja Black-Scholesovo formulo za vrednotenje evropskih opcij:

$$\text{nakupna:} \qquad C = S_0 N(d_1) - K N(d_2) e^{-RT}$$

$$\text{prodajna:} \qquad P = K e^{-RT} N(-d_2) - S_0 N(-d_1)$$

kjer sta

$$d_1 = \frac{log(\frac{S_0}{K} e^{RT}) + \frac{\sigma^2}{2} T}{\sigma \sqrt{T}}$$

$$d_2 = d_1 - \sigma \sqrt{T}$$

N pa je porazdelitvena funkcija standardne normalne porazdelitve

$$N(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{x} e^{-\frac{y^2}{2}} dy$$

1.4. **Ornstein-Uhlenbeckov proces.**

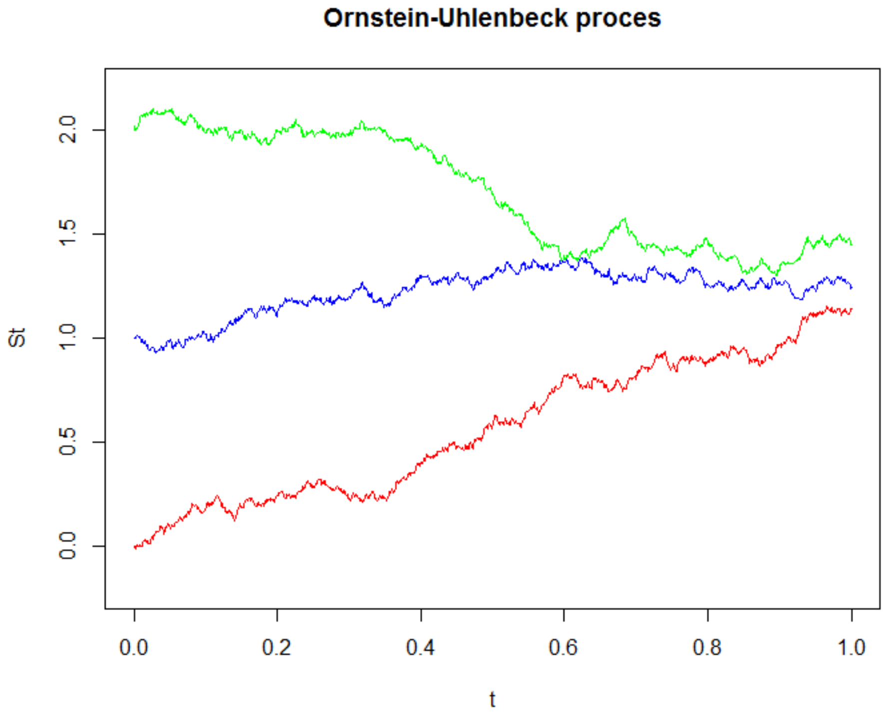
V financah se zelo pogosto pojavi proces, ki se s časom približuje nekemu dolgoročnemu povprečju. Tak proces lahko vidimo že pri gibanju cen blaga. Če je cena previsoka, se bo povpraševanje zmanjšalo, zato bo cena padla. Če pa bo cena prenizka, se bo zgodilo ravno nasprotno. Podobno gibanje opazimo tudi pri gibanju obrestnih mer itd. Najbolj popularen model je Ornstein-Uhlenbeckov proces.
Proces sledi naslednji stohastični diferencialni enačbi:

$$dX_t = \theta(\mu - X_t)dt + \sigma dW_t \tag{2}$$

kjer so $\theta > 0$, μ in $\sigma > 0$ parametri, W_t pa označuje Wienerjev proces.
Kot sem že omenila, je za O-U proces značilno, da se dolgoročno približuje povprečju μ. Hitrost približevanja je večja, ko je proces odmaknjen dlje od centra. Parameter σ predstavlja velikost slučajnih odmikov, θ pa stopnjo, po kateri se ti odmiki izgubljajo in se spremenljivka približuje ravnotežni vrednosti.

Spodaj je grafični prikaz za parametre $\theta = 1$, $\mu = 1.2$ in $\sigma = 0.3$. Prikazani so trije različni procesi, ki se razlikujejo v začetni vrednosti. Začnejo v 0, 1 in v 2.

```
OU <- function(S0, theta, mi, sigma, T, n){
    dt  <- T/n
    dW<-rnorm(n, 0, sqrt(dt))
    S = rep(0, n+1)
    S[1] <- S0
    for (i in 2:(n+1)){
      S[i]  <-  S[i-1] + theta * (mi - S[i-1])*dt + sigma * dW[i-1]
    }
  return(S)
}

S1 <- OU(2, 1, 1.2, 0.3, 1, 1000)
S2 <- OU(1, 1, 1.2, 0.3, 1, 1000)
S3 <- OU(0, 1, 1.2, 0.3, 1, 1000)
```



Ornstein-Uhlenbeckov proces lahko predstavimo s formulo

$$S_t = S_0 e^{-\theta t} + \mu(1 - e^{-\theta t}) + \frac{\sigma}{\sqrt{2\theta}} e^{-\theta t} W_{e^{2\theta t}-1}$$

kjer je $W_{e^{2\theta t}-1}$ porazdeljen $N(0, e^{2\theta t} - 1)$.

S_t je afina transformacija $W_{e^{2\theta t}-1}$ z raztegom $\frac{\sigma}{\sqrt{2\theta}} e^{-\theta t}$ in premikom $S_0 e^{-\theta t} + \mu(1 - e^{-\theta t})$. Iz tega sledi, da je S_t porazdeljen:

$$S_t \sim N(S_0 e^{-\theta t} + \mu(1 - e^{-\theta t}), \frac{\sigma^2}{2\theta}(1 - e^{-2\theta t})).$$

Želimo izračunati vrednost opcije, ki sledi O-U procesu. Izplačilna funkcija je $max\{X_t - K, 0\}$, zato bomo dobili ceno nakupne opcije z izračunom pričakovane vrednosti izplačila. To storimo tako, da izračunamo naslednji integral. Funkcijo gostote pri tem označimo s f.

$$C(K) = \int_{-\infty}^{\infty} (x - K)_+ f(x) dx \tag{3}$$

Ta integral bo imel vrednost 0 za vse x, ki so manjši od K, saj bo v tem primeru vrednost izraza $max\{X_t - K, 0\}$ enaka 0. Integriramo lahko zato od K do ∞. Dobljeni integral še razdelimo na dva in dobimo sledeče:

$$\int_K^{\infty} (x - K) f(x) dx = \int_K^{\infty} x f(x) dx + K \int_K^{\infty} f(x) dx$$

Zadnji integral lahko preuredimo in dobimo ravno vrednost porazdelitvene funkcije F od X v točki K.

$$\int_K^{\infty} f(x) = 1 - \int_{-\infty}^{K} f(x) = F(K).$$

2. Metoda Euler-Maruyama

2.1. **Motivacija.**

Za relativno malo slučajnih diferencialnih enačb poznamo način za izračun točnih vrednosti. Zato potrebujemo neko metodo, s katero lahko izračunamo čim boljše približke za vrednost procesa, ki sledi tej slučajni diferencialni enačbi, v nekem konkretnem času T.

Najenostavnejša metoda za računanje približkov navadnih diferencialnih enačb je Eulerjeva metoda. Podano imamo diferencialno enačbo $y' = f(x, y)$ z začetnim pogojem $y(x_0) = y_0$. Iz točke (x_0, y_0) se želimo premakniti v točko (x_1, y_1), kjer je y_1 približek za vrednost $y(x_1)$ in $x_1 = x_0 + h$. Diferencialna enačba nam da še smer tangente v prvi točki. Ideja je, da se za h premaknemo v smeri tangente.

Približke dobivamo po naslednjih dveh formulah:

$$y_{n+1} = y_n + hf(x_n, y_n)$$
$$x_{n+1} = x_n + h$$

Z manjšanjem koraka h pa seveda dobivamo vedno boljše približke.

Izrek 2.1. *Eulerjeva metoda je numerična metoda, ki vrne globalno napako prvega reda.*

Dokaz. Razvoj funkcije f v točki $x_0 + h$ izgleda takole:

$$f(x_0 + h) = f(x_0) + hf'(x_0) + \frac{h^2}{2} y''(x_0) + O(h^3)$$

Če za približek $f(x_0+h)$ vzamemo $f(x_0)+hf'(x_0)$, nam ostane lokalna napaka reda $O(h^2)$. Če te lokalne napake seštejemo, dobimo globalno napako reda $O(h)$, torej je Eulerjeva metoda prvega reda. □

Podobno idejo lahko uporabimo tudi pri reševanju slučajnih diferencialnih enačb. Tej metodi rečemo metoda Euler-Maruyama.

2.2. **Postopek metode Euler-Maruyama.**

Metodo uporabimo na slučajnih diferencialnih enačbah oblike

$$dX_t = a(X_t)dt + b(X_t)dW_t$$

z začetnim pogojem $X_0 = x_0$. W_t v enačbi ponovno označuje Wienerjev proces. Privzamemo tudi, da želimo rešiti to diferencialno enačbo na nekem intervalu $[0, T]$.

Potem je Euler-Maruyama aproksimacija rešitve X definirana tako:

- Interval razdelimo na manjše podintervale dolžine $h = \triangle t$.
- Vrednost X_0 postavimo na x_0.
- rekurzivno ponavljamo naslednji korak:

$$X_{n+1} = X_n + a(X_n)\triangle t + b(X_n)\triangle W_n \qquad n \in \{0, 1, \dots, (\frac{T}{h} - 1)\}$$

kjer je $\triangle W_n = W_{h(n+1)} - W_{hn}$.
Opazimo, da so $\triangle W_n$ neodvisne slučajne spremenljivke, porazdeljene normalno, z matematičnim upanjem 0 in varianco h.

2.3. **Konvergenca Euler-Maruyame.**

2.3.1. *Krepka konvergenca.*

Krepka konvergenca se ukvarja s pričakovano vrednostjo napake.

Definicija 2.2. Metoda krepko konvergira z redom γ, če obstaja konstanta C, tako da velja:

$$E[|X_n - X(nh)|] \leq C(h)^\gamma$$

za katerikoli fiksen $nh \in [0, T]$ in dovolj majhen h.

Ker ta definicija velja za vse t iz $[0, T]$, se bomo osredotočili na točke na koncu intervala. To pomeni, da za napako velja:

$$e_{dt}^{krepka} = E[|X_T - X(T)|] \leq C(dt)^\gamma$$

Če to neenačbo logaritmiramo (to smemo storiti, saj je logaritemska funkcija strogo naraščujoča), pa dobimo naslednje:

$$\log e_{dt}^{krepka} \approx \log C + \gamma \log dt$$

Iz tega sledi, da lahko vrednost γ sklepamo iz grafa, kjer imamo na x-osi vrednosti logaritma dolžine koraka dt, na y-osi pa logaritem napake. Vrednost γ bo odvod premice, ki se najbolje prilega točkam. Napake v času T dobimo kot absolutno vrednost razlike med točno rešitvijo in približkom, ki ga dobimo z uporabo Euler-Maruyame. Če postopek ponovimo dovolj velikokrat in izračunamo povprečje, potem je pričakovana vrednost napake enaka $E[|X_T - X(T)|]$ za dano velikost koraka.

Definicija 2.3. Funkcija f zadošča Lipschitzovemu pogoju na $[a, b]$, če obstaja konstanta N, da za vsak $x, y \in [a, b]$ velja

$$|f(x) - f(y)| \leq N|x - y|.$$

Izrek 2.4. *Metoda Euler-Maruyama krepko konvergira reda 0.5, za primerno slučajno diferencialno enačbo, ki zadošča Lipschitzovemu pogoju na [0, T].*

Dokaz najdemo v [1].

Iz krepke konvergence sledi šibka, vendar nam ne prikaže najvišjega reda šibke konvergence.

2.3.2. *Šibka konvergenca.*

Šibka konvergenca pa se ukvarja z napako pričakovane vrednosti. Za metodo pravimo, da konvergira šibko, reda γ, če obstaja konstanta C, tako da za vse gladke funkcije p, ki imajo polinomsko rast, velja:

$$|Ep(X_n) - Ep(X(nh))| \leq C(h)^\gamma$$

za poljuben $nh \in [0, T]$ in dovolj majhen h.

Izrek 2.5. *Metoda Euler-Maruyama šibko konvergira reda 1, za gladke funkcije s polinomsko rastjo.*

Dokaz najdemo v [1].

Kot vemo, pri numeričnih izračunih pride do napak. Ker hočemo čim manjšo napako, pa lahko uporabimo ekstrapolacijo.
Najprej bom razložila idejo Richardsonove ekstrapolacije na Eulerjevi metodi. Če izračunamo približek rezultata dane diferencialne enačbe s korakom h, v resnici dobimo točno vrednost, spremenjeno za $h \cdot A$ in še napako reda $O(h^2)$, ki je zanemarljiva. Ko uporabimo polovičen korak $\frac{h}{2}$, pa dobimo podobno. To prikazujeta spodnji enačbi:

$$Euler(T, h) = Y + hA + O(h^2)$$
$$Euler(T, \frac{h}{2}) = Y + \frac{h}{2}A + O(h^2)$$

Želimo se znebiti napake prvega reda z linearno kombinacijo zgornjih dveh približnih rešitev.

$$\alpha Euler(T, h) + \beta Euler(T, \frac{h}{2}) = Y + O(h^2)$$

Za zgornji primer dobimo enačbi $\alpha + \beta = 1$ pa $\alpha + \frac{\beta}{2} = 0$. Iz tega sledi $\alpha = -1$ in $\beta = 2$.
Z uporabo ekstrapolacije torej dobimo napako reda 2 z metodami prvega reda in zato lahko pridemo do boljšega približka z manj izračuni.
Podobno idejo lahko uporabimo pri optimiziranju metode Euler-Maruyama.

3.1. Ekstrapolacija napake višjega reda.

Talay in Tubaro sta v članku [2] pokazala, da se je z ekstrapolacijo šibke metode Euler-Maruyame mogoče znebiti napake m-tega reda. Pri tem mora za funkciji a in b iz spodnje slučajne diferencialne enačbe veljati, da sta iz C^∞, torej neskončno mnogokrat zvezno odvedljivi. Poleg tega pa mora za odvod poljubnega reda veljati, da je omejen.

$$dX_t = a(X_t)dt + b(X_t)dW_t$$

Pri Euler-Maruyami, ima napaka razvoj:

$$Err(T) = e_r(T)h^r + e_{r+1}(T)h^{r+1} + ... + e_m(T)h^m + O(h^{m+1}),$$

kjer $e_r, \ldots, e_m$ niso odvisni od h. Iz tega sledi naslednja ideja, da se lahko z m različnimi dolžinami korakov znebimo napake m-tega reda.
Vzemimo m različnih korakov

$$h_0 > h_1 > h_2 > ... > h_m$$

Označimo še z $Z_{i,0}$ dobljeni rezultat, pri uporabi koraka h_i in naj bo f gladka funkcija.

$$Z_{i,0} = Ef(T, h_i)$$

S temi oznakami lahko sedaj prikažemo naslednjo shemo, v kateri je $Z_{m,m}$ izračunan po naslednji formuli:

$$Z_{i,k} = Z_{i,k-1} + \frac{Z_{i,k-1} - Z_{i-1,k-1}}{\frac{h_{i-k}}{h_i} - 1}$$

Shema izgleda tako:

$$\begin{array}{ccccc}
Z_{0,0} & & & & \\
 & Z_{1,1} & & & \\
Z_{1,0} & & Z_{2,2} & & \\
 & Z_{2,1} & \vdots & \cdots & Z_{m,m} \\
Z_{2,0} & \vdots & Z_{m,2} & & \\
\vdots & Z_{m,1} & & & \\
Z_{m,0} & & & &
\end{array}$$

Velja:

$$Ef(X_T) - Z_{m,m}(h) = O(h^{m+1})$$

Opazimo lahko, da s to shemo pridemo do istega rezultata za α in β, pri ekstrapolaciji s korakoma h in $\frac{h}{2}$. Naj bo $h_0 = h$ in $h_1 = \frac{h}{2}$. Za ta primer shema izgleda tako:

$$\begin{array}{cc}
Z_{0,0} & \\
 & Z_{1,1} \\
Z_{1,0} &
\end{array}$$

Po formuli za izračun $Z_{1,1}$ dobimo:

$$Z_{1,1} = Z_{1,0} + \frac{Z_{1,0} - Z_{0,0}}{\frac{h_{1-1}}{h_1} - 1} = Z_{1,0} + \frac{Z_{1,0} - Z_{0,0}}{2-1} = 2 * Z_{1,0} - Z_{0,0}$$

Torej sta koeficienta enaka:

$$\alpha = -1 \qquad \beta = 2$$

3.2. **Monte Carlo.**

Do druge izboljšave pa pridemo s pomočjo metode Monte Carlo. To je integracijska metoda za izračun

$$Ef(S_T, K)$$

Postopek ponovimo N-krat, nato pa izračunamo povprečje dobljenih rezultatov. Zakon velikih števil nam pove, da ima metoda Monte Carlo napako

$$Err = \frac{1}{\sqrt{N}}$$

Hitro lahko opazimo, da napaka konvergira proti 0 z večanjem števila ponovitev N.

Za analizo napake, bom primerjala rezultate, ki jih vrne metoda Euler-Maruyama, s točnimi rezultati. To bom naredila za dva primera. Najprej na Black-Scholesovem modelu, nato pa še na procesu Ornstein-Uhlenbeck, saj za oba modela lahko izračunamo tudi točno rešitev.

4.1. **Black-Scholesov model.**

Za primer vzemimo nek vrednostni papir, za katerega velja naslednje:

- S0 - vrednost osnovnega premozenja v casu 0 = 100
- K - izvrsilna cena = 110
- R - netvegana obrestna mera = 0.02
- T - cas zapadlosti = 1
- sigma - odklon = 0.5

Te podatke vstavimo v program, ki nam po Black-Scholesovi formuli izračuna eksaktno ceno nakupne opcije.

```
BS <- function(S0, K, R, T, sigma) {
 d1 <- (log(S0 / K)  + (sigma^2/2 + R) * T ) / (sigma * sqrt(T))
 d2 <- d1 - sigma * sqrt(T)

 C <- S0 * pnorm(d1) - K * pnorm(d2) * exp (-R * T)
 return(C)
}
```

Dobimo točno ceno naše nakupne opcije; to je 16.82972.
Nato ceno izračunamo še s pomočjo metode Euler-Maruyama.

```
EM_za_BS <- function(S0, K, R, T, sigma, ponovitve, n){
E <- rep(0, 10)
  for(j in 1:10){
    dt  <- T/n
    rez <- rep(0, ponovitve)
    dW = rep(0, n)
    SlucVektor<-rnorm(n * ponovitve, 0, sqrt(dt))
    S = rep(0, n+1)
    for(p in 1:ponovitve){
      dW<-SlucVektor[((p-1) * n + 1) : (p*n)]
      S[1] <- S0
      for (i in 2:(n+1)){
        S[i]  <- S[i-1] + R * S[i-1] * dt + sigma * S[i-1] * dW[i-1]
      }
      rez[p] <- max(S[n+1] - K, 0)
    }
```

```
    E[j] <- exp(-R * T) * sum(rez) / ponovitve
  }
  tocna_vr <- BS(S0, K, R, T, sigma)
  napake <- rep(0, 10)
  for(j in 1:10){
    napake[j] <- abs(tocna_vr - E[j])
  }
  glob_napaka <- sum(napake) / 10
  return(glob_napaka)
}
```

Zgornji program poleg osnovnih spremenljivk vzame za argument tudi število ponovitev, torej kolikokrat želimo simulirati pot vrednosti osnovnega premoženja, in pa število delilnih točk n, na katere razdelimo osnovni interval T.
Program najprej izračuna dolžino intervala dt, nato pa simulira poti. Na koncu vsake poti določi vrednost dobička in ga diskontira. Ko to poračuna za vse ponovitve, izračuna še povprečje. To je cena opcije, ki bi jo definiral v tem primeru. Za boljši prikaz rezultatov ponovi vse to 10-krat. Nato za vsako od teh ponovitev izračuna napako; to je absolutna vrednost razlike točne vrednosti (16.82972) in pa cene, ki jo je določil za ta primer.
Vrne povprečje teh desetih napak.

Poglejmo si najprej, kako se napaka spreminja v odvisnosti od števila ponovitev.

```
for(i in 0:4){
  ponovitve = 10^i
  n = 2000
  print(EM_za_BS(100, 110, 0.02, 1, 0.5, ponovitve, n))
}
```

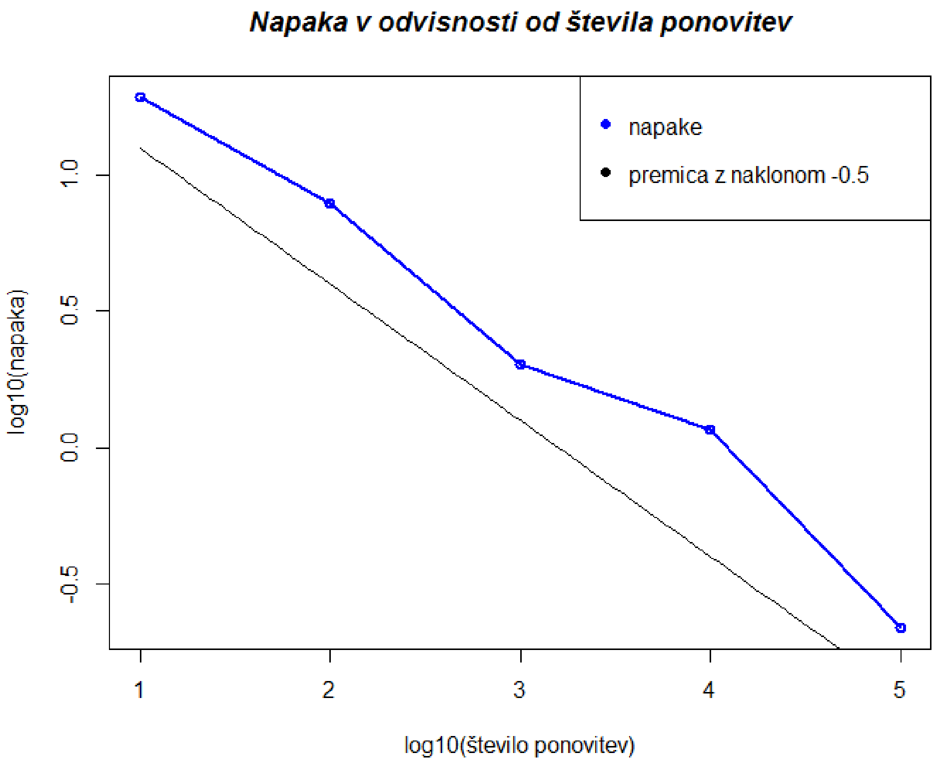
Število delilnih točk postavimo dovolj visoko, da je ta napaka zanemarljiva v primerjavi z napako, ki jo dobimo v odvisnosti od števila ponovitev. Privzela sem $n = 2000$. Izračun pa sem ponovila za $1, 10, \ldots, 10^5$.
Dobljeni rezultati so prikazani na grafu.

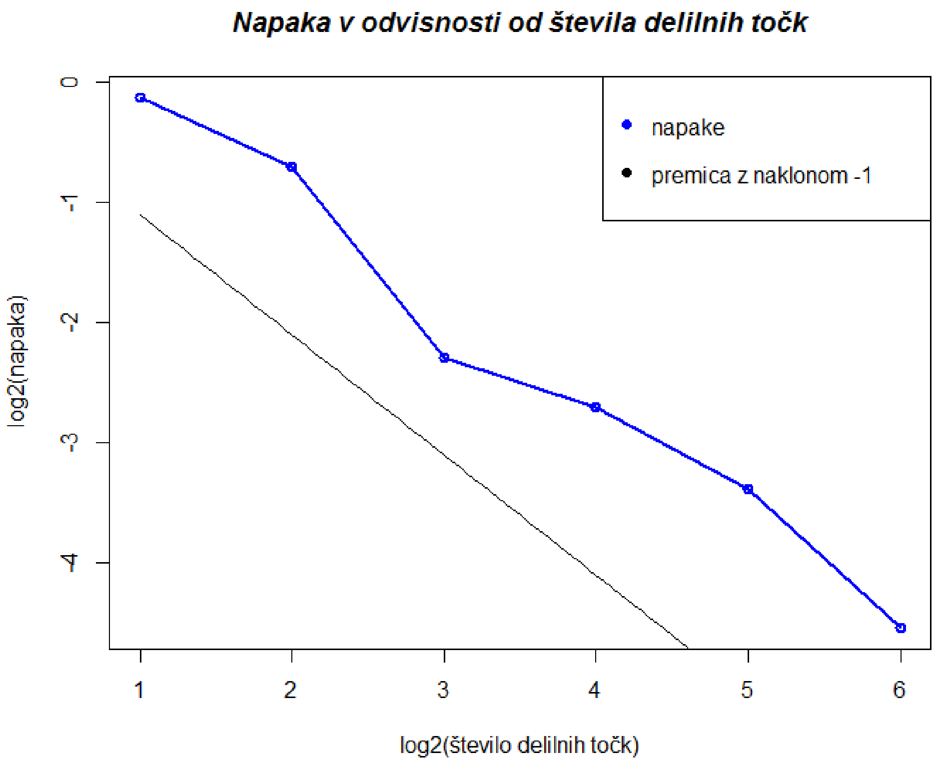
Na log-log grafu opazimo, da se napaka v odvisnosti od števila ponovitev manjša, stopnja konvergence je približno 0.5

Poglejmo si še, kako se obnaša napaka v odvisnosti od števila delilnih točk, pri čemer postavimo število ponovitev dovolj visoko, da ta napaka ne povzroči prevelikega šuma. Vzemimo 100000 ponovitev in si oglejmo rezultate.

```
for(i in 0:5){
   ponovitve = 100000
   n = 2^i
   print(EM_za_BS(100,110,0.02,1,0.5, ponovitve, n))
}
```

Opazimo lahko, da se napaka v odvisnosti od števila ponovitev manjša s konvergenco reda 1.

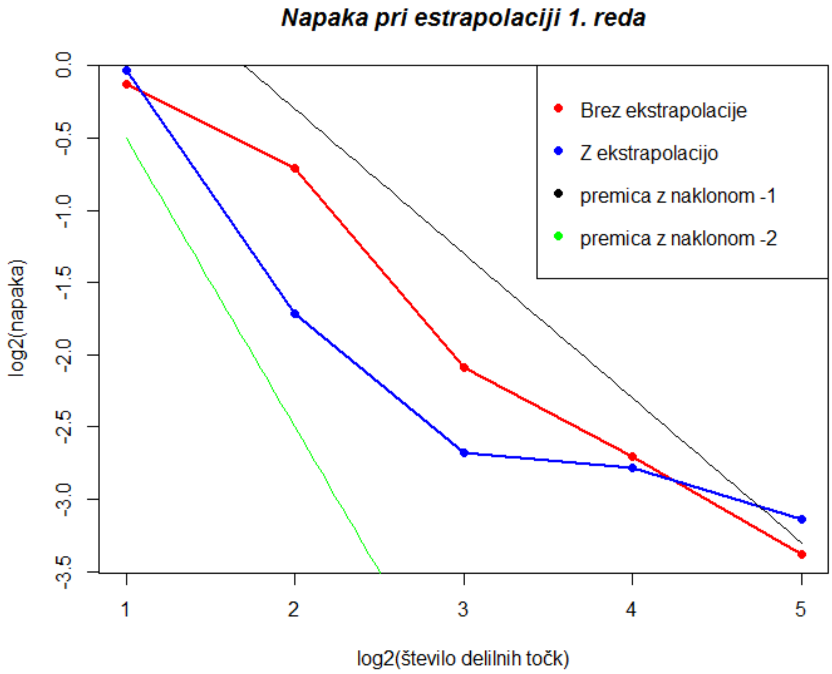




Poglejmo si še rezultate, ki jih dobimo z uporabo ekstrapolacije 1. reda.

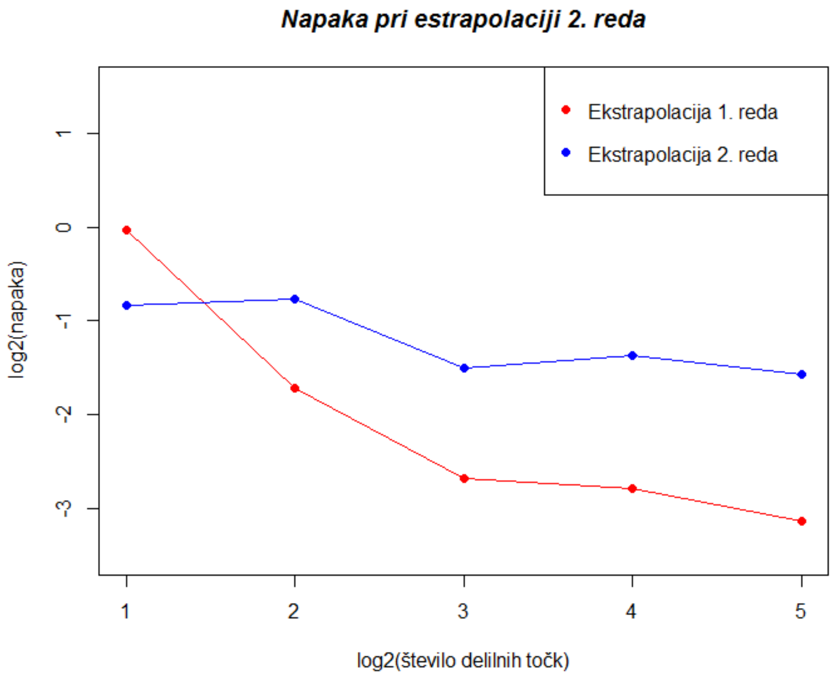
```
Ekstrapolacija1_BS <- function(S0, K, R, T, sigma, ponovitve, n){
  E <- rep(0, 10)
  for(k in 1:10){
    dt1  <- T/n
    dt2 <- dt1/2
    rez1 <- rep(0, ponovitve)
    rez2 <- rep(0, ponovitve)
    dW1 <- rep(0, n)
    dW2 <- rep(0, 2*n)
    SlucVektor1<-rnorm(n * ponovitve, 0, sqrt(dt1))
    SlucVektor2<-rnorm(2 * n * ponovitve, 0, sqrt(dt2))
    S1 = rep(0, n+1)
    S2 = rep(0, 2*n + 1)
    for(p in 1:ponovitve){
      dW1<-SlucVektor1[((p-1) * n + 1) : (p*n)]
      dW2<-SlucVektor2[((p-1) * 2 * n + 1) : (2 * p * n)]
      S1[1] <- S0
      S2[1] <- S0
      for (i in 2:(n+1)){
        S1[i]  <- S1[i-1] + R * S1[i-1]*dt1 + sigma*S1[i-1]*dW1[i-1]
      }
      for (j in 2:(2*n + 1)){
        S2[j]  <-  S2[j-1] + R * S2[j-1]*dt2 + sigma*S2[j-1]*dW2[j-1]
      }
      rez1[p] <- max(S1[n+1] - K, 0)
      rez2[p] <- max(S2[2*n+1] - K, 0)
    }
    E1 = exp(-R * T) * sum(rez1) / ponovitve
    E2 = exp(-R * T) * sum(rez2) / ponovitve
    E[k] <- 2 * E2 - E1
  }
  tocna_vr <- BS(S0, K, R, T, sigma)
  napake <- rep(0, 10)
  for(k in 1:10){
    napake[k] <- abs(tocna_vr - E[k])
  }
  glob_napaka <- sum(napake) / 10
  return(glob_napaka)
}

for(i in 0:4){
  ponovitve = 100000
  n = 2^i
  print(c("n:  ", 2^i))
  print(Ekstrapolacija1_EM_za_BS(100,110,0.02,1,0.5, ponovitve, n))
}
```

Iz naslednjega grafa opazimo, da je stopnja konvergence na začetku 1. Ponovno, kot sem omenila že prej, se nato konvergenca ustavi zaradi napake, ki jo povzroči število ponovitev.



Pogledjmo še napako pri ekstrapolaciji 2. reda, ki je prikazana na naslednjem grafu, skupaj z napako ekstrapolacije 1. reda. Opazimo, da je sicer na začetku napaka ekstrapolacije 2. reda manjša, vendar pa se z večanjem števila delilnih točk neha manjšati. Še več, stabilizira se višje kot napaka pri ekstrapolaciji 1. reda.

To se zgodi, ker se pri ekstrapolaciji integracijske napake lahko seštejejo (pomnožene z absolutnimi vrednostmi faktorjev).



4.2. **Ornstein-Uhlenbeck proces.**

V poglavju 1.4 sem opisala, kako izračunamo porazdelitev vrednosti osnovnega premoženja v času t. Za primer privzemimo naslednje vrednosti:

- S0 - vrednost osnovnega premozenja v casu $0 = 1$
- K - izvrsilna cena $= 1.05$
- μ - dolgoročna pričakovana vrednost $= 1.2$
- T - cas zapadlosti $= 1$
- θ - hitrost približevanja $\mu = 1$
- sigma - odklon $= 0.5$

Porazdelitev vrednosti osnovnega premoženja v času $T = 1$ je:

$$S_1 \sim N(1 * e^{-1} + 1.2 * (1 - e^{-1}), \frac{0.5^2}{2}(1 - e^{-2})) \sim N(1.126424, 0.1080831)$$

Poračunamo oba integrala:

$$\int_{1.05}^{\infty} x \frac{1}{\sqrt{2\pi}\sqrt{0.1080831}} e^{-\frac{(x-1.126424)^2}{2*0.1080831}} = 0.794402$$

$$\int_{1.05}^{\infty} \frac{1}{\sqrt{2\pi}\sqrt{0.1080831}} e^{-\frac{(x-1.126424)^2}{2*0.1080831}} = 0.59191$$

Rešitvi združimo in izračunamo pričakovano vrednost izplačila v času 1:

$$0.794402 - 1.05 * 0.59191 = 0.1728965$$

Eksaktna pričakovana vrednost izplačila je torej 0.1728965.
Zanima nas, kako dobro se z metodo Euler-Maruyama približamo tej rešitvi. Pri tem si pomagamo s spodnjim programom.

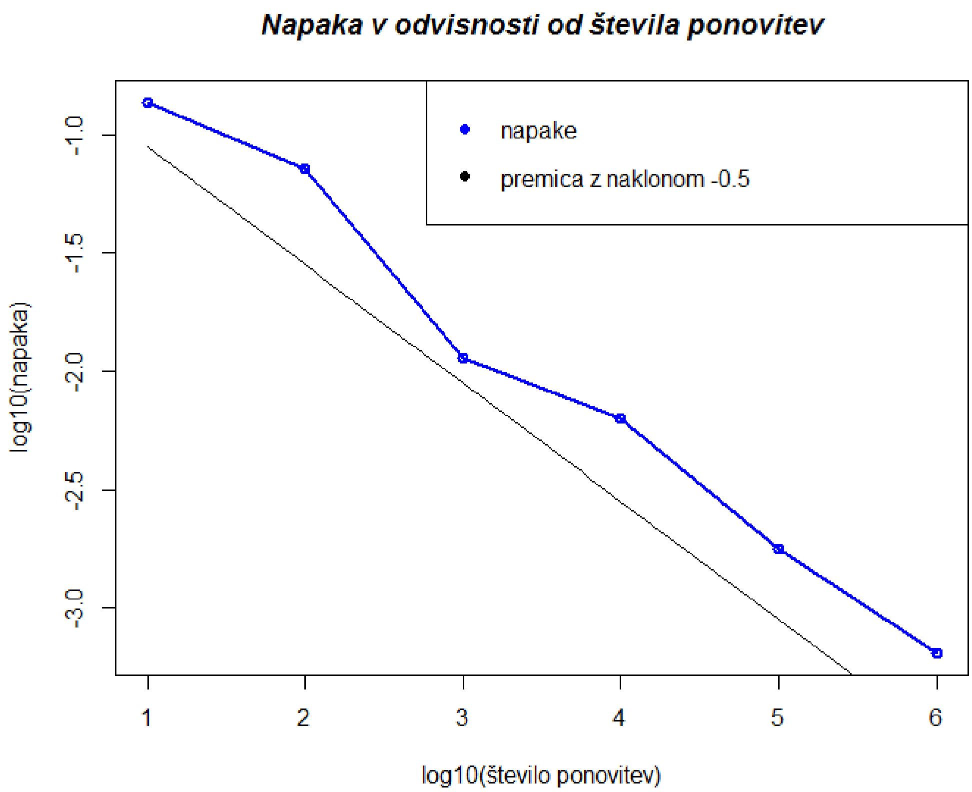
```
EM_za_OU <- function(S0, K, mi, theta, sigma, T, ponovitve, n){
E <- rep(0, 10)
  for(j in 1:10){
    dt  <- T/n
    rez <- rep(0, ponovitve)
    dW = rep(0, n)
    SlucVektor<-rnorm(n * ponovitve, 0, sqrt(dt))
    S = rep(0, n+1)
    for(p in 1:ponovitve){
      dW<-SlucVektor[((p-1) * n + 1) : (p*n)]
      S[1] <- S0
      for (i in 2:(n+1)){
        S[i]  <-  S[i-1] + theta * (mi - S[i-1])*dt + sigma * dW[i-1]
      }
      rez[p] <- max(S[n+1] - K, 0)
    }
    E[j] <- sum(rez) / ponovitve
  }
  tocna_vr <- 0.1728965
  napake <- rep(0, 10)
```

```
  for(j in 1:10){
    napake[j] <- abs(tocna_vr - E[j])
  }
  glob_napaka <- sum(napake) / 10
  return(glob_napaka)
}
```

Enako kot pri računanju za model Black-Scholesa, si izberemo število ponovitev in pa število delilnih točk intervala, za katera želimo dobiti izračun. Program izračuna dolžino intervala dt, nato pa simulira poti. Na koncu vsake poti določi vrednost dobička. Ko to poračuna za vse ponovitve, izračuna še povprečje. To je vrednost izplačila opcije v času T. Za boljši prikaz rezultatov, postopek ponovi 10-krat. Nato za vsako od teh ponovitev izračuna napako; absolutno vrednost razlike točne vrednosti (0.1728965) in približka z metodo Euler-Maruyama.
Vrne povprečje desetih napak.

Poglejmo si najprej, kako se napaka spreminja v odvisnosti od števila ponovitev.

```
for(i in 0:5){
  ponovitve = 10^i
  n = 2000
  EM_za_OU(1, 1.05, 1.2, 1, 0.5, 1, ponovitve, n)
}
```

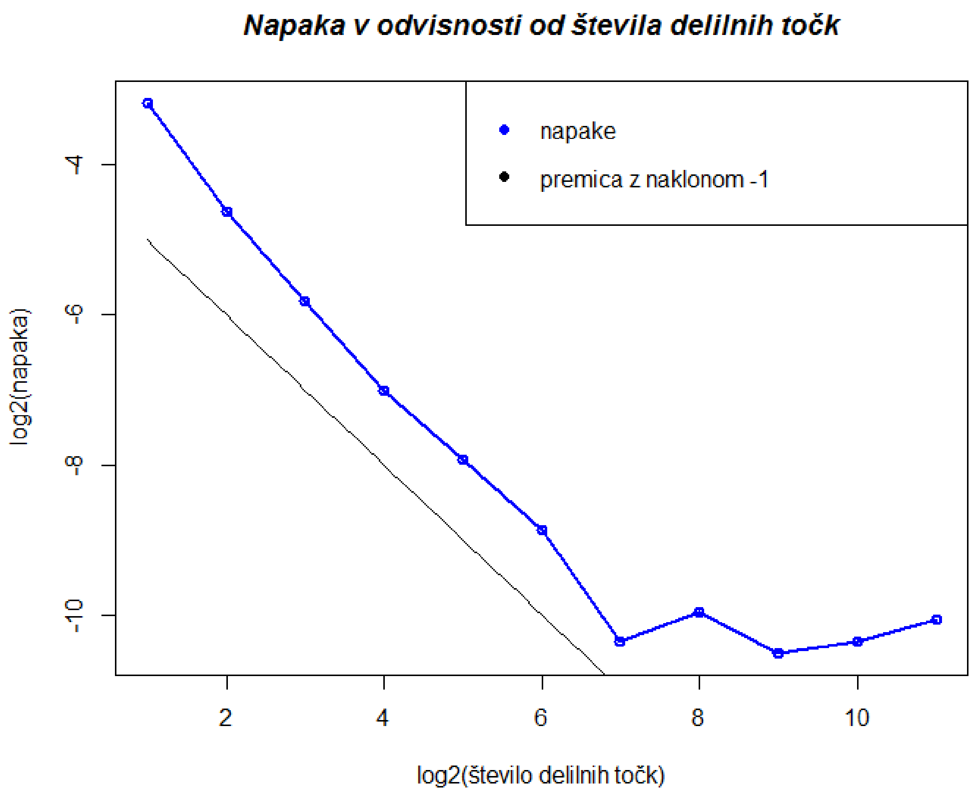


Enako kot pri Black-Scholesu določimo število delilnih točk dovolj veliko, da je ta napaka zanemarljiva glede na napako, ki jo dobimo v odvisnosti od števila ponovitev. Privzela sem $n = 2000$. Izračun pa sem ponovila za $1, 10, \ldots, 10^5$.
Dobljeni rezultati so prikazani na grafu !!!.

Naslednji program pa nam izpiše vrednosti napak, ko povečujemo število delilnih točk. Določenih je 50000 ponovitev. Rezultati so prikazani spodaj.

```
for(i in 0:10){
  ponovitve = 50000
  n = 2^i+
  print(EM_za_OU(1, 1.05, 1, 1.2, 0.5, 1, ponovitve, n))
}
```

Opazimo, da se pri 2^8 točk napaka neha manjšati. Na tej točki je napaka ≈ 0.001478491. Približno tako velika pa je bila napaka tudi, ko smo pri prejšnjem grafu primerjali napake v odvisnosti od ponovitev, pri zelo velikem številu delilnih točk. To pomeni, da tukaj že prevlaga napaka zaradi premajhnega števila ponovitev.



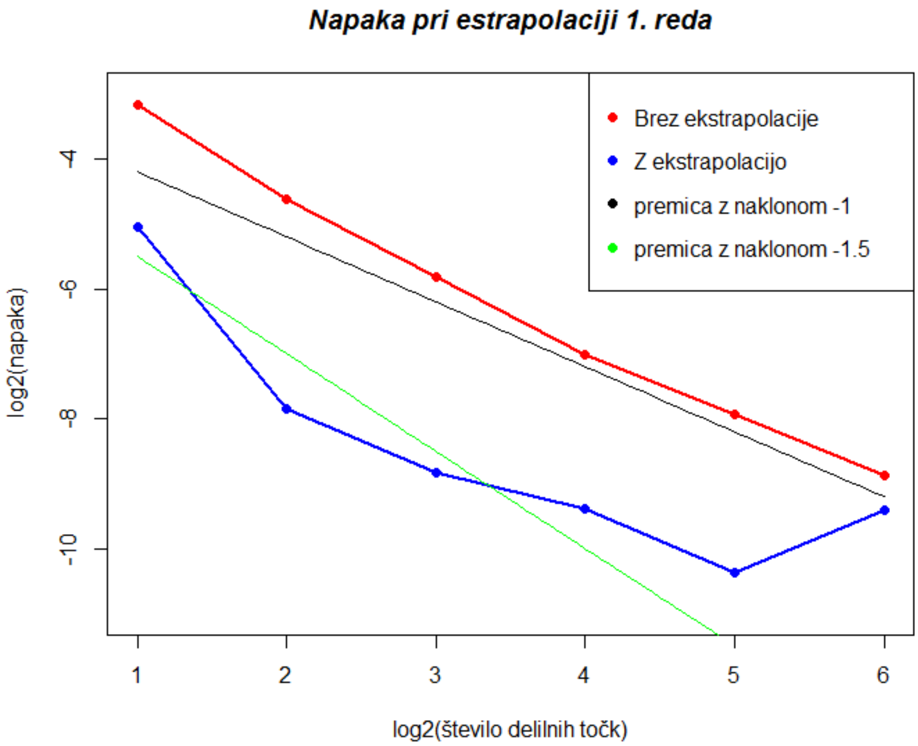
Poglejmo še, kakšno napako dobimo, če uporabimo ekstrapolacijo.

```
Ekstrapol_OU <- function(S0, K, theta, mi, sigma, T, ponovitve, n){
  E <- rep(0, 10)
  for(k in 1:10){
```

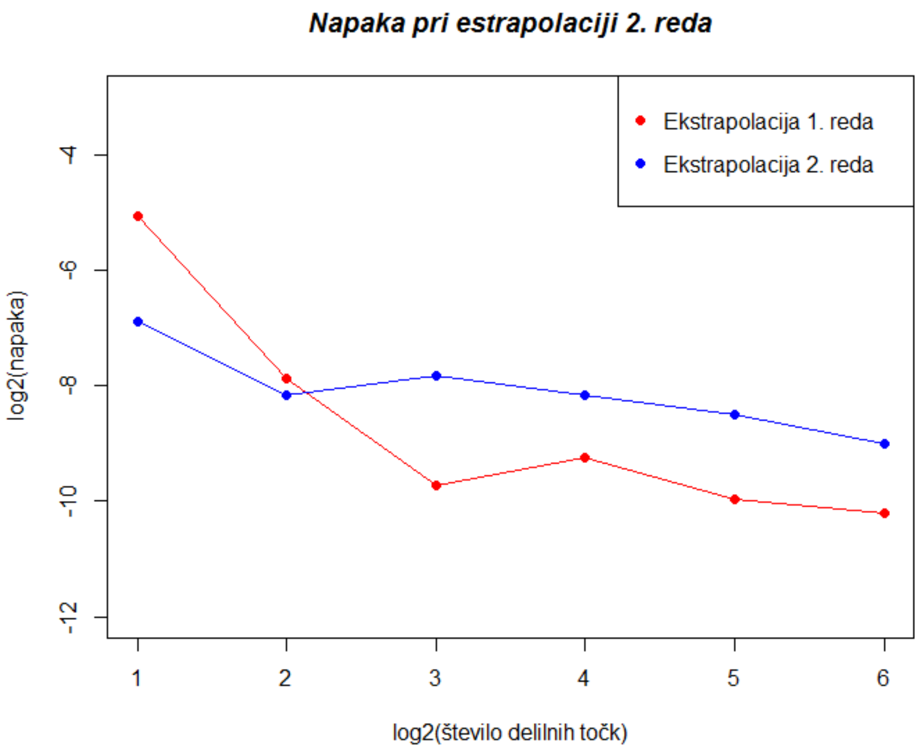
```
    dt1  <- T/n
    dt2 <- dt1/2
    rez1 <- rep(0, ponovitve)
    rez2 <- rep(0, ponovitve)
    dW1 <- rep(0, n)
    dW2 <- rep(0, 2*n)
    SlucVektor1<-rnorm(n * ponovitve, 0, sqrt(dt1))
    SlucVektor2<-rnorm(2 * n * ponovitve, 0, sqrt(dt2))
    S1 = rep(0, n+1)
    S2 = rep(0, 2*n + 1)
    for(p in 1:ponovitve){
      dW1<-SlucVektor1[((p-1) * n + 1) : (p*n)]
      dW2<-SlucVektor2[((p-1) * 2 * n + 1) : (2 * p * n)]
      S1[1] <- S0
      S2[1] <- S0
      for (i in 2:(n+1)){
       S1[i]  <-  S1[i-1] + theta*(mi - S1[i-1])*dt1 + sigma*dW1[i-1]
      }
      for (j in 2:(2*n + 1)){
       S2[j]  <-  S2[j-1] + theta*(mi - S2[j-1])*dt2 + sigma*dW2[j-1]
      }
      rez1[p] <- max(S1[n+1] - K, 0)
      rez2[p] <- max(S2[2*n+1] - K, 0)
    }
    E1 = sum(rez1) / ponovitve
    E2 = sum(rez2) / ponovitve
    E[k] <- 2 * E2 - E1
  }
  tocna_vr <- 0.1728965
  napake <- rep(0, 10)
  for(k in 1:10){
    napake[k] <- abs(tocna_vr - E[k])
  }
  glob_napaka <- sum(napake) / 10
  return(glob_napaka)
}
```

Napake nam izpiše naslednji program. Na grafu so prikazane napake, ki jih dobimo brez uporabe ekstrapolacije, in pa z njeno uporabo. Opazimo, da je napaka, ki jo dobimo z ekstrapolacijo bistveno manjša. Pri uporabi ekstrapolacije, dobimo z 8 delilnimi točkami napako okoli 0.0028, med tem ko pri izračunavanju brez ekstrapolacije enako napako dobimo šele pri 64 delilnih točkah. Na grafu se črti kasneje približata, saj ponovno do izraza pride napaka zaradi števila ponovitev.

```
for(i in 0:5){
  ponovitve = 10000
  n = 2^i
  print(Ekstrapol_OU(1, 1.05, 1, 1.2, 0.5, 1, ponovitve, n))
}
```



Na spodnjem grafu je prikazana še napaka, ki jo dobimo z uporabo ekstrapolacije 2. reda. Opazimo enak pojav kot pri Black-Scholesu, namreč pri večjem številu delilnih točk do izraza pridejo integracijske napake, ki se lahko pri ekstrapolaciji seštejejo.



4.3. **Ornstein-Uhlenbeck proces 2.**

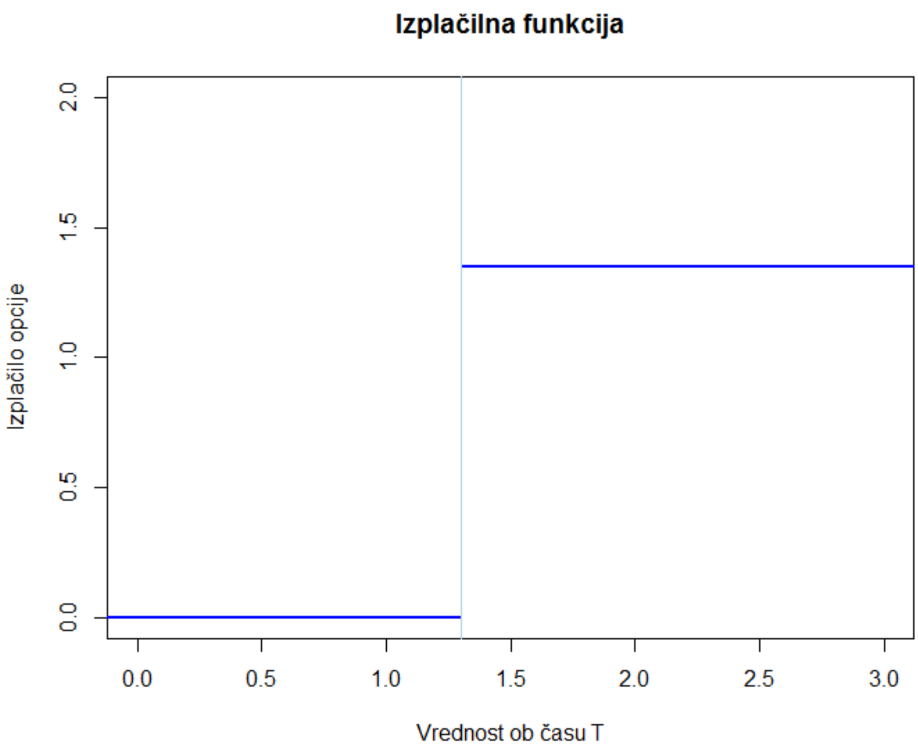
Do zdaj sem obravnavala le zvezne, gladke izplačilne funkcije. Poglejmo si še en primer, ko izplačilna funkcija ni zvezna. Za primer vzemimo binarno opcijo, katere vrednost sledi Ornstein-Uhlenbeckovemu procesu z naslednjimi vrednostmi:

- S0 - vrednost osnovnega premozenja v casu 0 = 1
- μ - dolgoročna pričakovana vrednost = 1.2
- T - cas zapadlosti = 1
- θ - hitrost približevanja μ = 1
- sigma - odklon = 0.5

Recimo, da imamo naslednjo izplačilno funkcijo:

$$f(n) = \begin{cases} 1.35 & \text{če } S_T > 1.3 \\ 0 & \text{sicer} \end{cases}$$

Iz grafa, na katerem je prikazana le-ta funkcija, je razvidno, da ni zvezna.



Izračunati moramo točno pričakovano vrednost izplačila. Vemo, da je porazdelitev vrednosti osnovnega premoženja v času $T = 1$:

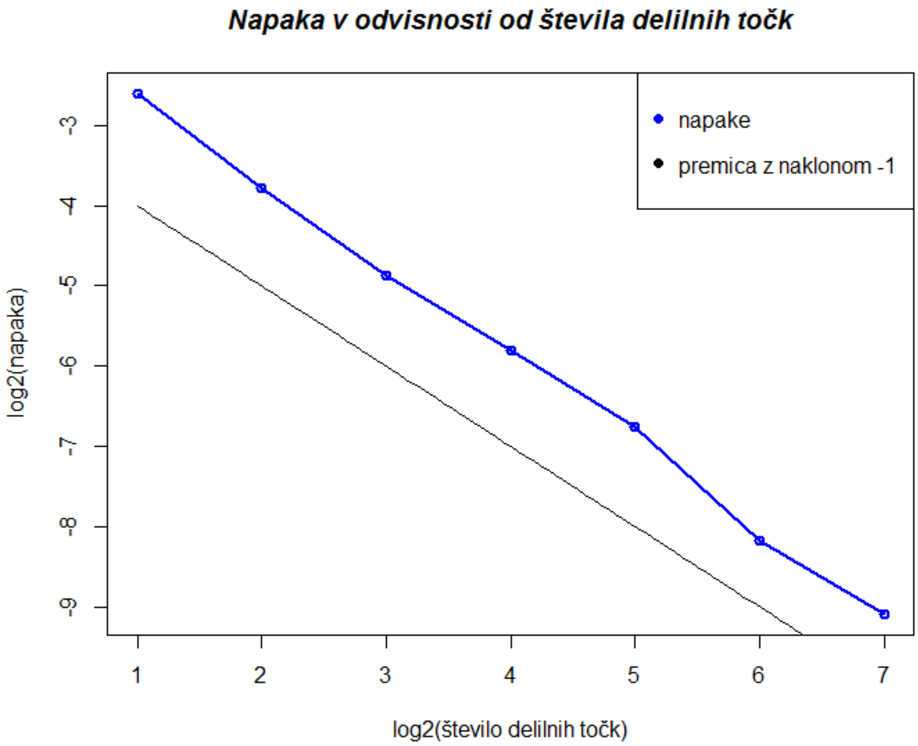
$$S_1 \sim N(1 * e^{-1} + 1.2 * (1 - e^{-1}), \frac{0.5^2}{2}(1 - e^{-2})) \sim N(1.126424, 0.1080831)$$

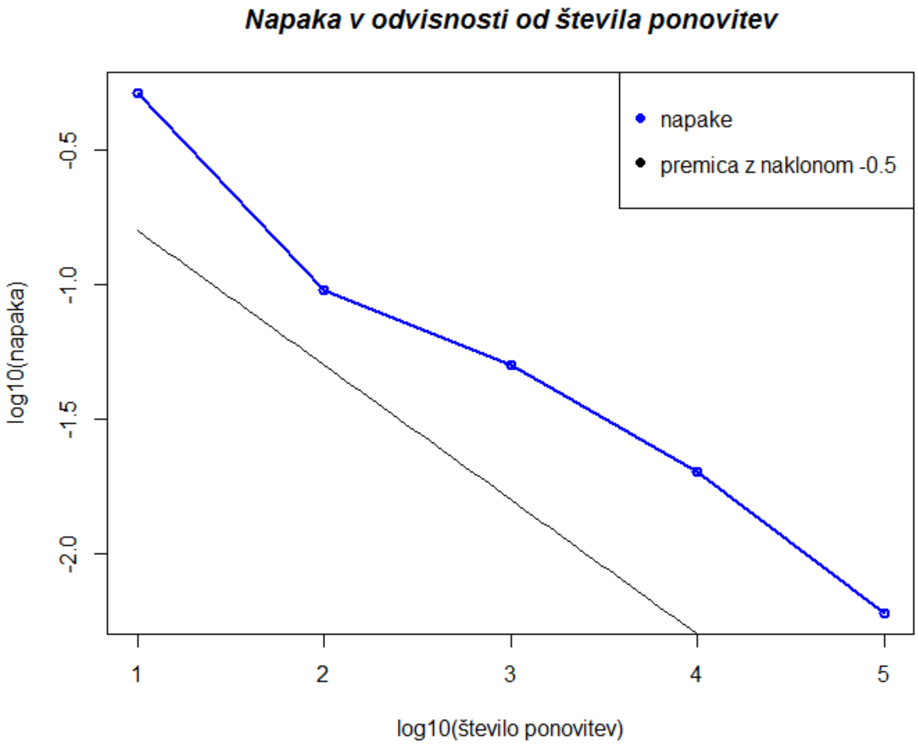
Poračunamo integral:

$$\int_{-\infty}^{1.3} 0 + \int_{1.3}^{\infty} 1.35 \frac{1}{\sqrt{2\pi}\sqrt{0.1080831}} e^{-\frac{(x-1.126424)^2}{2*0.1080831}} = 0.403325$$

Zdaj se pojavi vprašanje, ali tudi za takšno izplačilno funkcijo metoda Euler-Maruyama konvergira. Da bi to preverila, sem uporabila naslednji program.

```
EM_za_binarno <- function(S0, theta, mi, sigma, T, ponovitve, n){
  E <- rep(0, 10)
  for(j in 1:10){
    dt  <- T/n
    rez <- rep(0, ponovitve)
    dW = rep(0, n)
    SlucVektor<-rnorm(n * ponovitve, 0, sqrt(dt))
    S = rep(0, n+1)
    for(p in 1:ponovitve){
      dW<-SlucVektor[((p-1) * n + 1) : (p*n)]
      S[1] <- S0
      for (i in 2:(n+1)){
        S[i]  <-  S[i-1] + theta * (mi - S[i-1])*dt + sigma * dW[i-1]
      }
      if(S[n+1] >= 1.3) rez[p] = 1.35
      else rez[p] = 0
    }
    E[j] <- sum(rez) / ponovitve
  }
  tocna_vr <- 0.403325
  napake <- rep(0, 10)
  for(j in 1:10){
    napake[j] <- abs(tocna_vr - E[j])
  }
  glob_napaka <- sum(napake) / 10
  return(glob_napaka)
}



for(i in 0:4){
  ponovitve = 10^i
  n = 1000
  print(c("st ponovitev ", 10^i))
  print(EM_za_binarno(1, 1, 1.2, 0.5, 1, ponovitve, n))
}



for(i in 0:6){
  ponovitve = 50000
  n = 2^i
  print(c("n:  ", 2^i))
  print(EM_za_binarno(1, 1, 1.2, 0.5, 1, ponovitve, n))
}
```





Iz zgornjih dveh grafov je razvidno, da metoda Euler-Maruyama konvergira tudi za nezvezne izplačilne funkcije.

Literatura

[1] Peter E. Kloeden in Eckhard Platen, *Numerical Solution of Stochastic Differential Equations*, Springer-Verlag, Berlin, 1992

[2] Denis Talay in Luciano Tubaro, *Expansion of the global error for numerical schemes solving stochastic differential equations*, Rapports de Recherche, France, 1989

[3] Mark Richardson, *Stochastic Differential Equations Case Study*, University of Oxford, 2009

[4] Mark Richardson, *Numerical Methods for Option Pricing*, University of Oxford, 2009

[5] Chenggui Yuan in Xuerong Mao, *Convergence of the Euler-Maruyama method for stochastic differential equations with Markovian switching*, University of Strathclyde, Glasgow, 2003

[6] Timothy Sauer, *Numerical Solution of Stochastic Differential Equations in Finance*, George Mason University, Fairfax, 2012

[7] William Smith, *On the Simulation and Estimation of the Mean-Reverting Ornstein-Uhlenbeck Process*, 2010

[8] *Brownian motion*, [ogled 10. 5. 2014], dostopno na `http://en.wikipedia.org/wiki/Brownian_motion`.

[9] *Wiener process*, [ogled 10. 5. 2014], dostopno na `http://en.wikipedia.org/wiki/Wiener_process`.

[10] *Ornstein–Uhlenbeck process*, [ogled 17. 5. 2014], dostopno na `http://en.wikipedia.org/wiki/Ornstein%E2%80%93Uhlenbeck_process`.

[11] *Black-Scholes model*, [ogled 17. 5. 2014], dostopno na `http://en.wikipedia.org/wiki/Black%E2%80%93Scholes_model`.

[12] *Black–Scholes equation*, [ogled 17. 5. 2014], dostopno na `http://en.wikipedia.org/wiki/Black%E2%80%93Scholes_equation`.

[13] Neil A. Chriss, Ira Kawaller *Black-Scholes and Beyond: Option Pricing Models*, McGraw-Hill, 1996