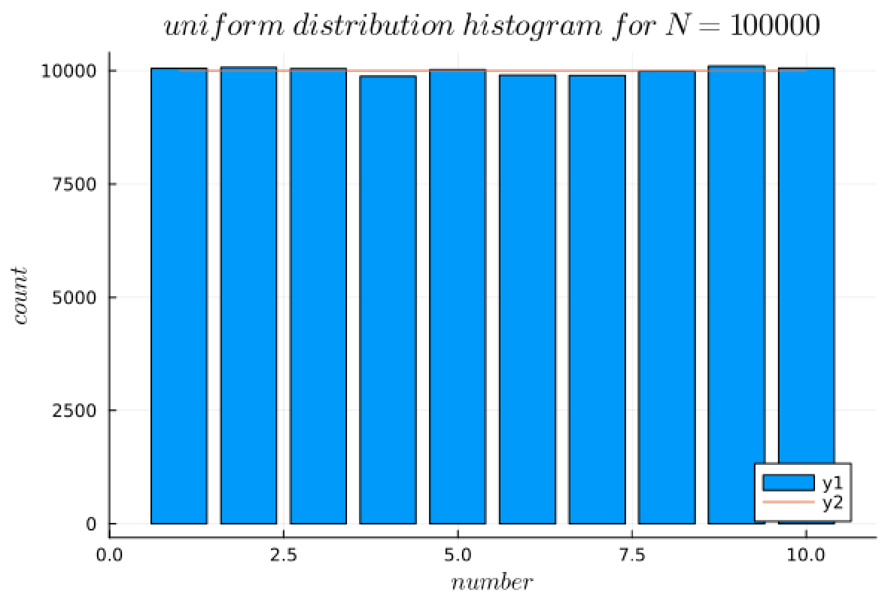
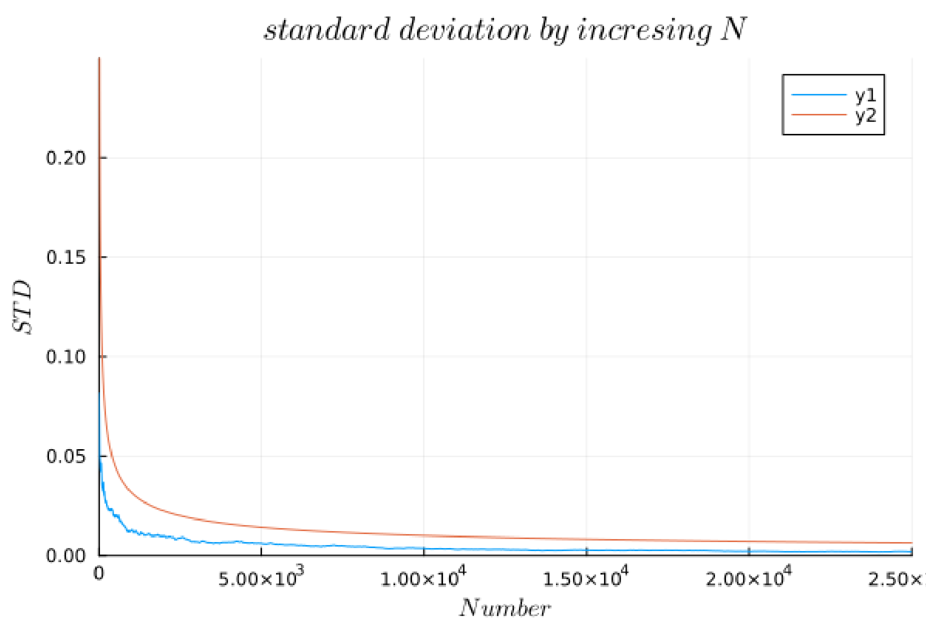
گزارش کار تمرین ششم شبیه سازی زهرا اکبری 98100612

سوال اول فصل شش

در این تمرین فراوانی اعداد رندوم تولید شده توسط جولیا را با شمارش و بدون استفاده از تابع هیستوگرام رسم کرده ایم که با تقریب خوبی یکنواخت است:



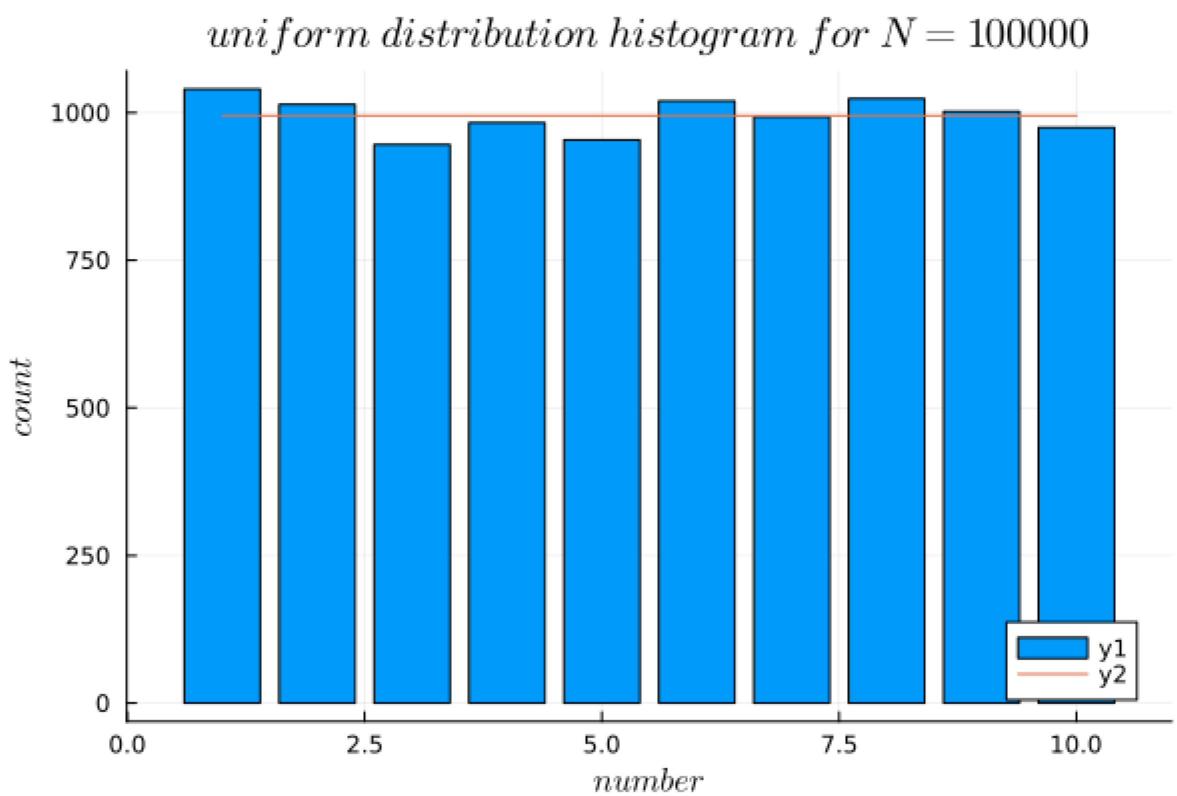


در ادامه انحراف معیار را که در حلقه حساب کرده بودیم برای یک دسته از اعداد تصادفی بر حسب افزایش تعداد رسم میکنیم. دقت میکنیم میتوانستیم برای N دسته از اعداد تصادفی متفاوت این نمودار را رسم کنیم اما از نظرم نشان دادن تحول در یک دسته از اعداد رندوم منطقی تر بود. البته میتوان با لیست های متفاوت تاثیر fluctuation بیشتر قابل مشاهده بود.

انتخاب اعدادی تصادفی بین 1 تا 10 همانند انتخاب یک سلول در ارایه ای برای ول نشست است. مشابه تمرین ول نشست سلولی انتخاب شده و به مقدار آن یکی اضافه میشود.

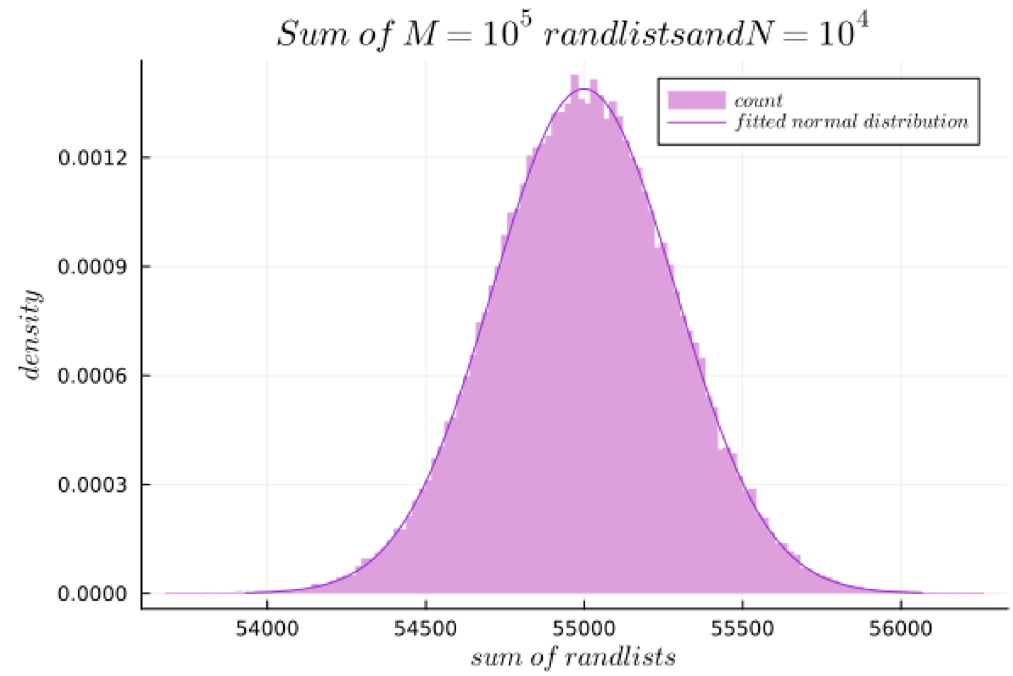
تمرین دوم فصل ششم

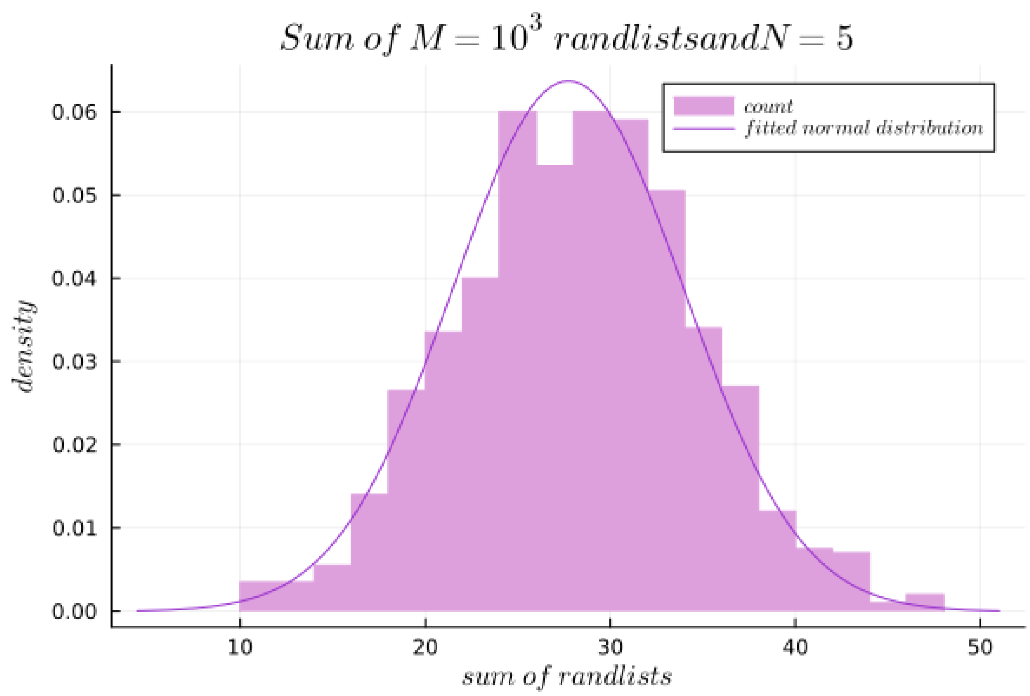
این بار توزیع اعداد بعد یک عدد مشخص مثل 4 را نمایش میدهیم مجددا به سادگی و ایجاد یک حلقه اعداد بعد 4 را در یک آرایه میریزیم و به روش قبل تعدادشان را شمرده و رسم میکنیم که همگن است اما نه به اندازه تمرین قبل.

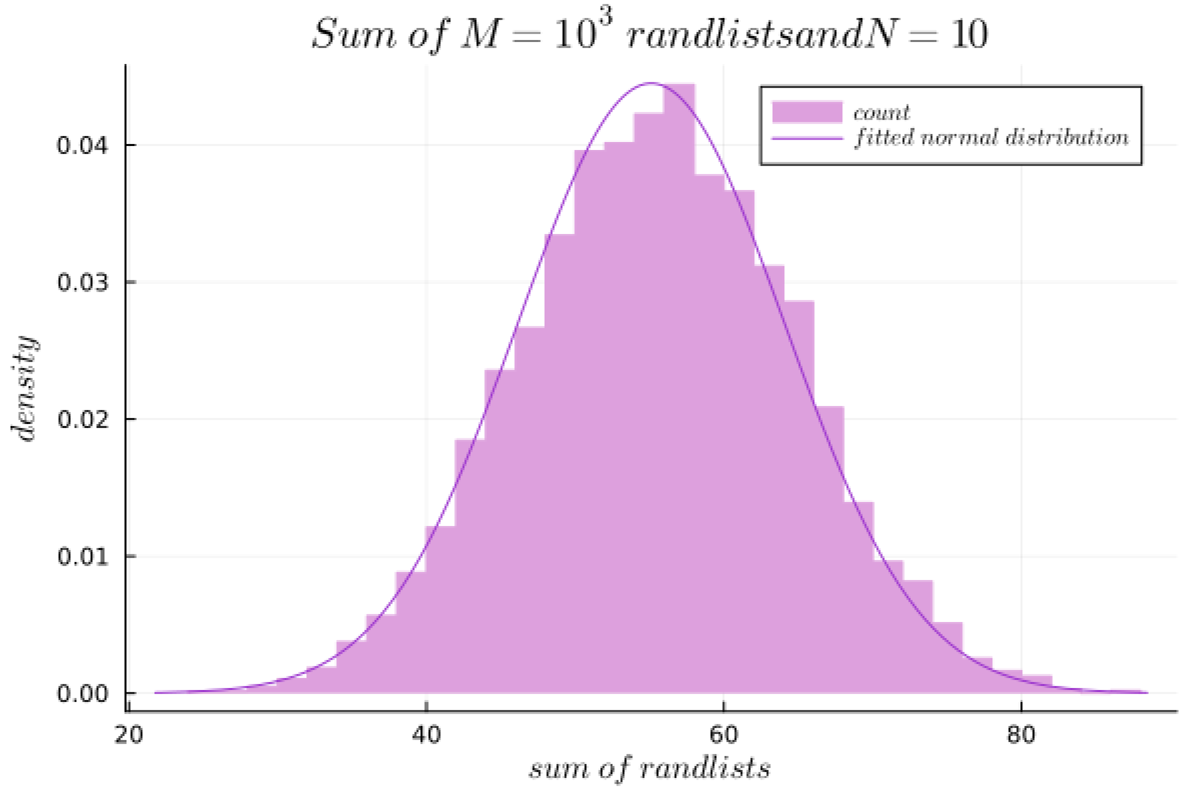


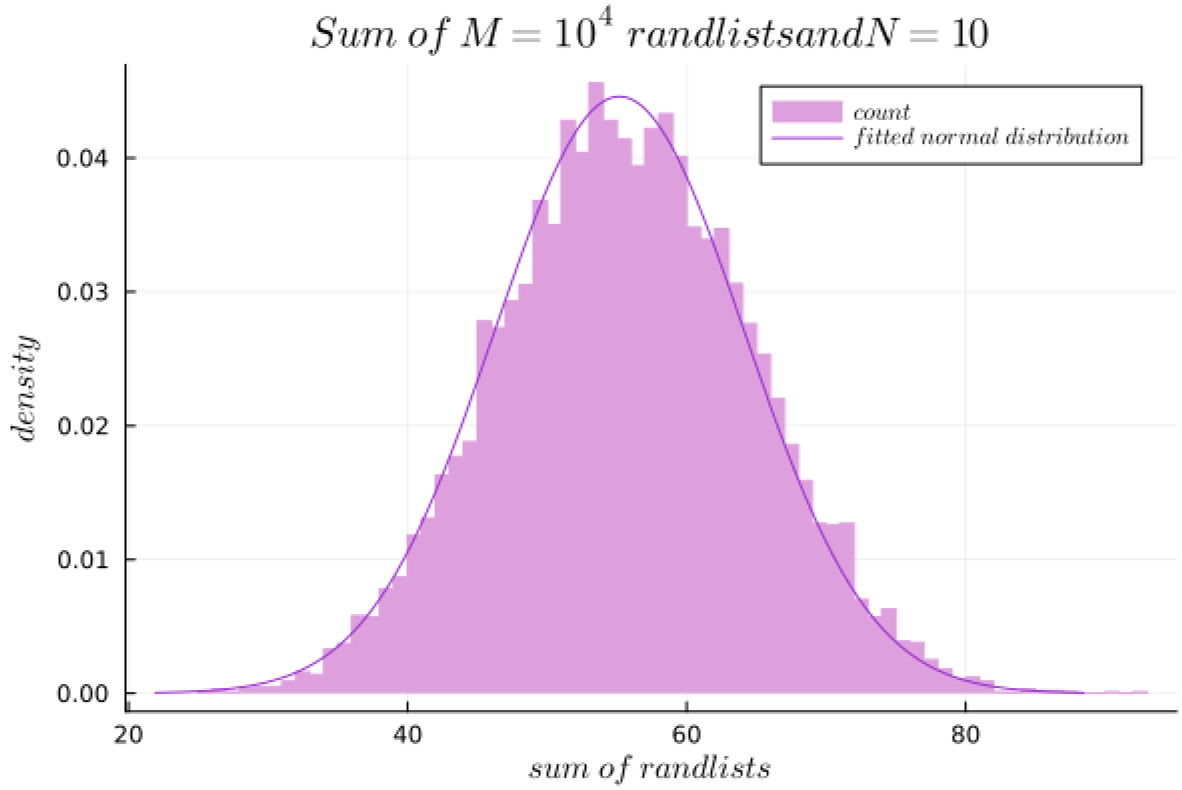
تمرین سوم فصل ششم

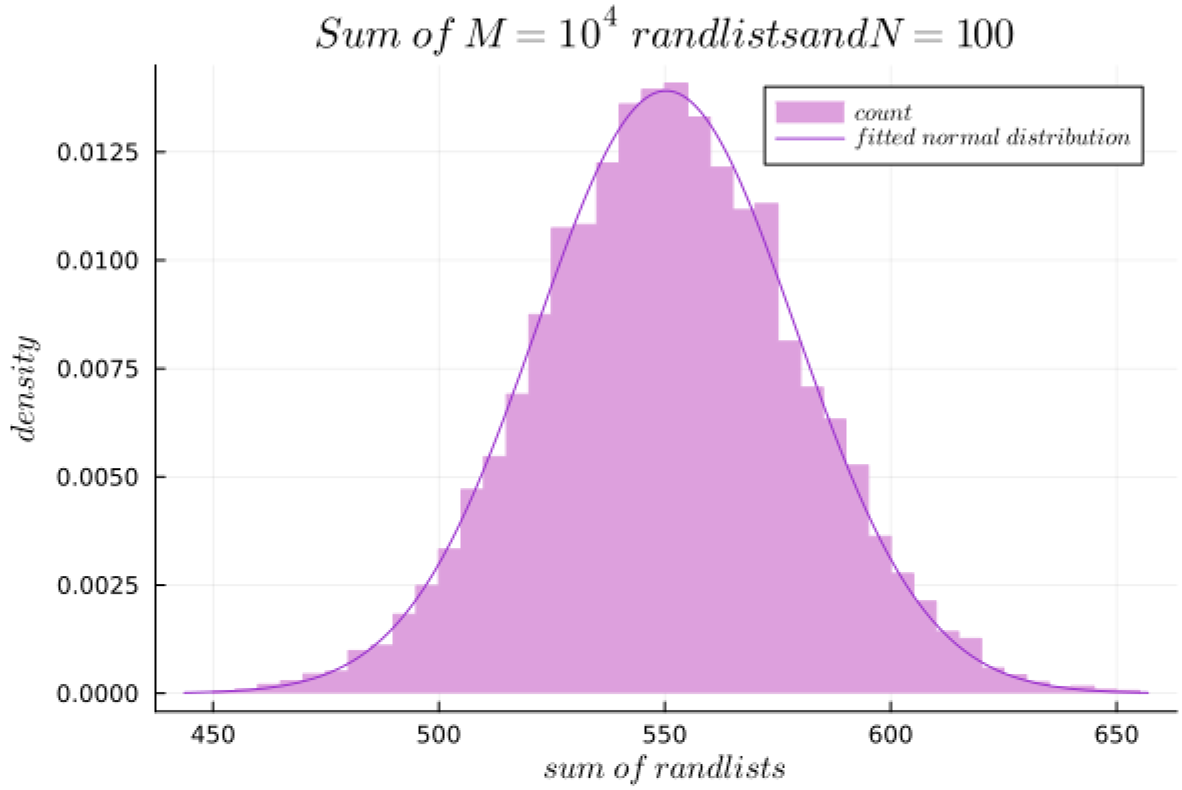
با استفاده از تابع randgenerator N عدد تصادفی تولید میکنیم در حلقه ای جمع M عدد و انحراف از معیارشان را محاسبه میکنیم. در شکل های زیر توزیع نرمال به دست آمده برای مقادیر متفاوت آورده شده اند:

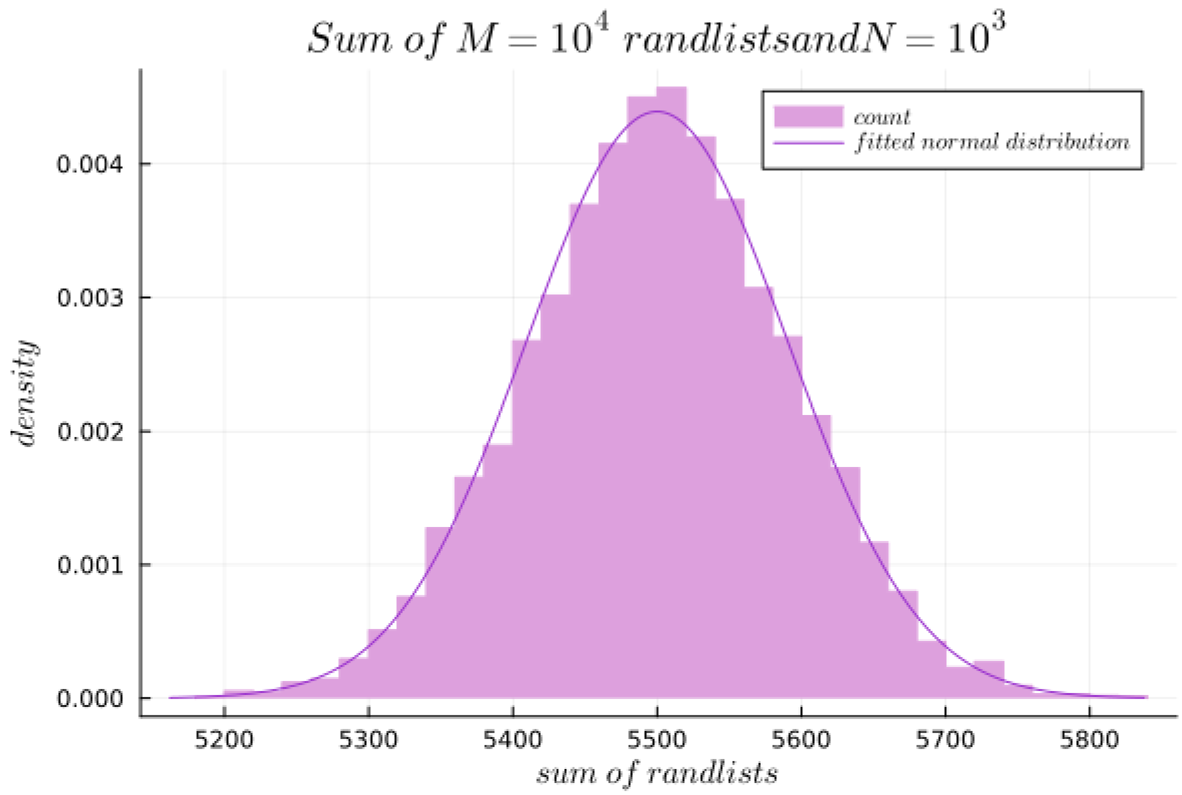





Sum of M = 10^3 randlistsandN = 10
count
fitted normal distribution
density
0.00
0.01
0.02
0.03
0.04
20
40
60
80
sum of randlists


Sum of M = 10^4 randlistsandN = 10
count
fitted normal distribution
density
0.00
0.01
0.02
0.03
0.04
20
40
60
80
sum of randlists





شباهت این تمرین با ولگشت در این است که مکان نهایی قدم زن تصادفی هم عملا جمع چند عدد تصادفی است.

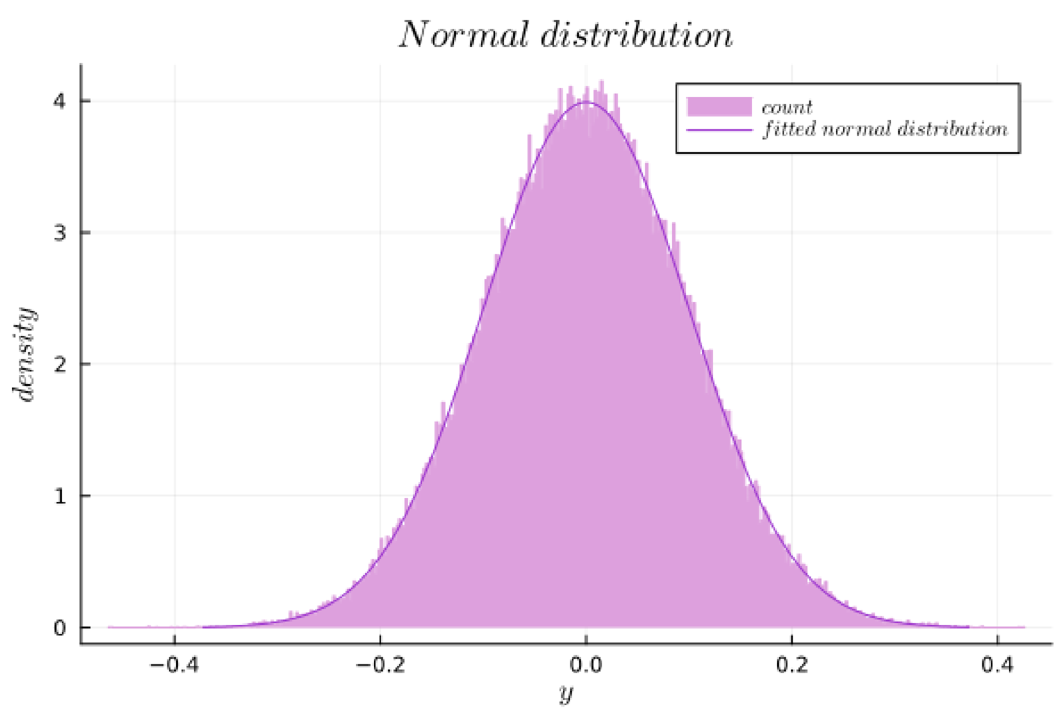
تمرین چهارم فصل ششم

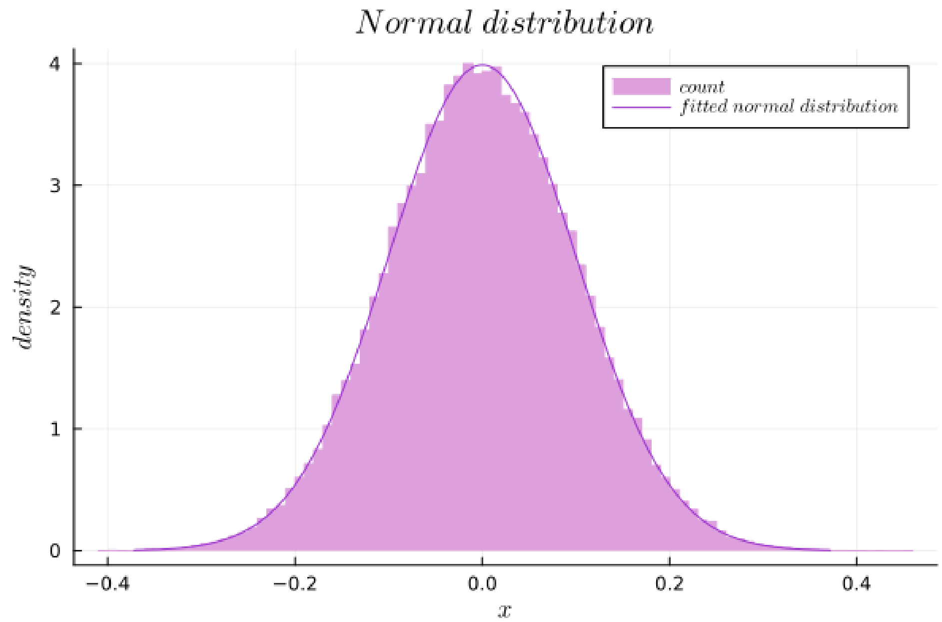
همانند روش کتاب از اعداد تصادفی با توزیع یکنواخت بین 0 تا 1 توزیع گوسی میسازیم. با انتگرال گیری توابع به کار رفته به دست می آیند.

در دو تابع زیر در مختصات قطبی اعداد تصادفی را ایجاد کرده و نهایتا با تبدیل معکوس شان توزیع ها را رسم میکنیم

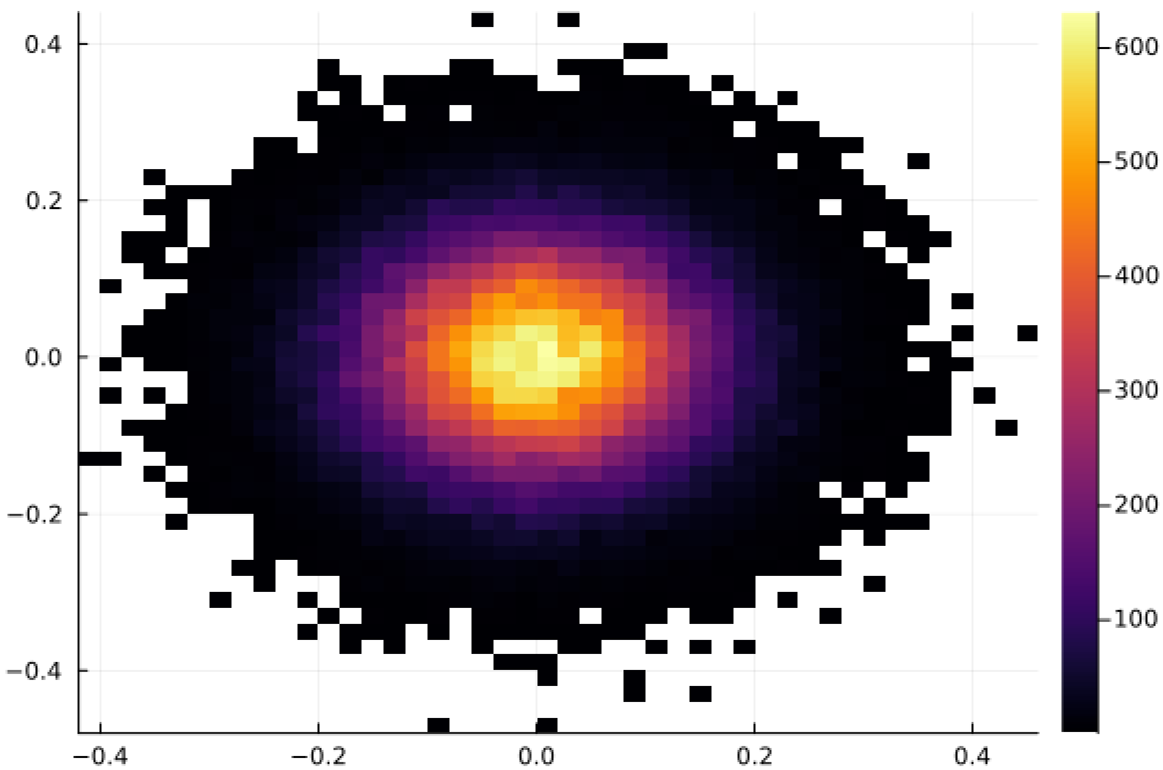
```
function ρInverse(x)          # Generating the cumilative inverse function mentioned in text book for both parameters
    ρ = sqrt(-2*(σ^2)*log(x))
    return(ρ)
end
function θInverse(y)
    θ = 2*π*y
    return(θ)
end
X = NormalRandListR .* cos.(NormalRandListθ) #Findin X and Y from polar coordinates for plotting
Y = NormalRandListR .* sin.(NormalRandListθ)
```

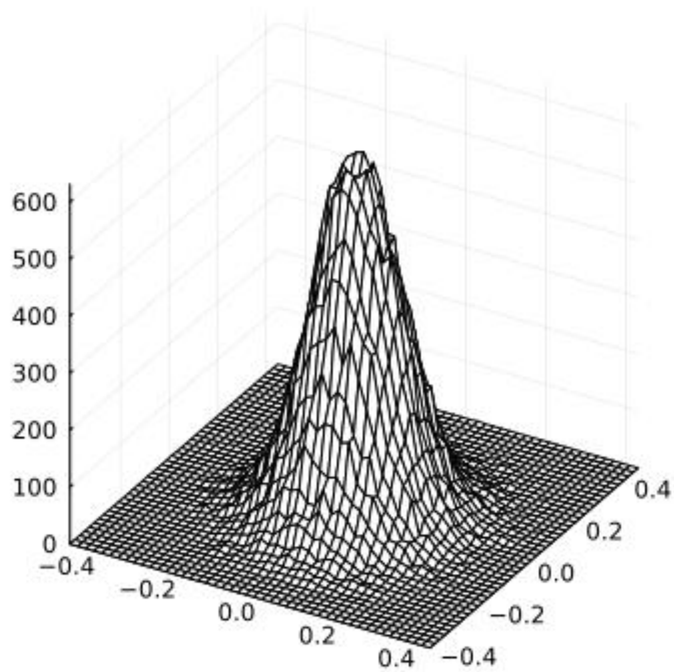
در ادامه تصویر گری های دیگری برای مشاهده شکل توزیع نرمال ایجاد شده تولید شده اند.



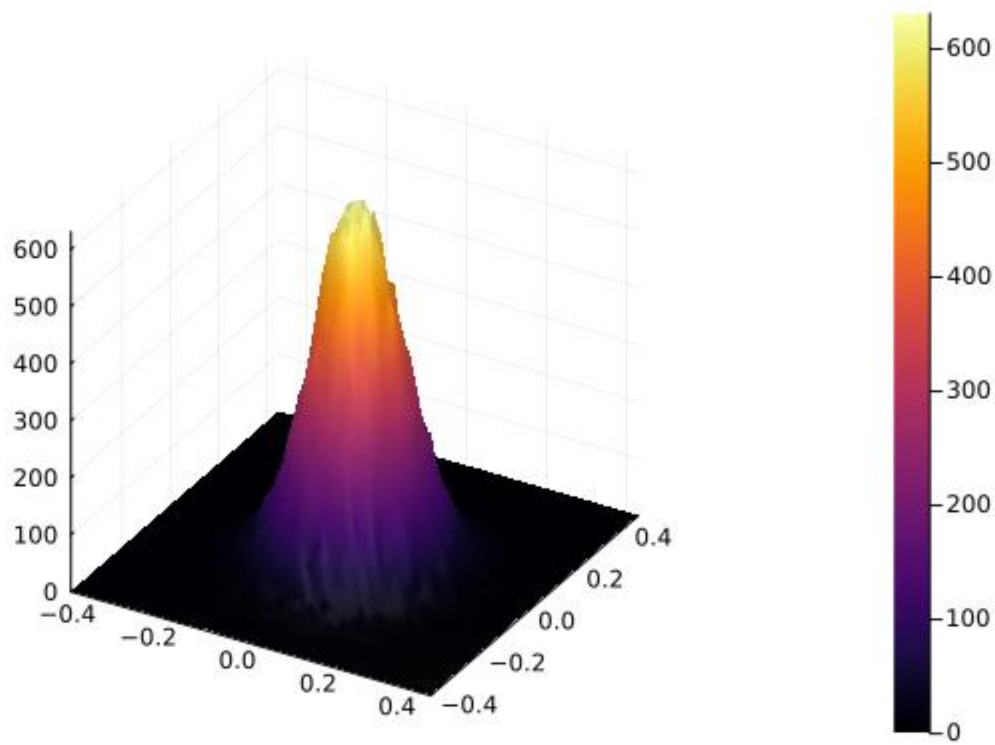


Normal distribution
count
fitted normal distribution
density
x





0
100
200
300
400
500
600
−0.4
−0.2
0.0
0.2
0.4
−0.4
−0.2
0.0
0.2
0.4



0
100
200
300
400
500
600
−0.4
−0.2
0.0
0.2
0.4
−0.4
−0.2
0.0
0.2
0.4
0
100
200
300
400
500
600

تمرین اول فصل هفتم

در این تمرین برای محاسبه انتگرال به روش ساده از توابع زیر استفاده شده که توضیح شان در کامنت ها هم آورده شده است. تابع اول صرفا اعداد تصادفی ایجاد کرده و دومی میانگین شان را در بازه ضرب میکند.

برای نمونه گیری هوشمند تابع مولد اعداد تصادفی به شکل زیر به دست می آید

$$g(y) = e^{-y} \rightarrow G(y) = \int_0^y e^{-t} dt = 1 - e^{-y} = x \rightarrow y = -\ln(1-x)$$

```
function SimpleRandGenerator(n)                              # Generating uniform
rand list between 0 to 2
    UniformRandlist = rand(n)
    return(UniformRandlist.*2)
end


function SimpleIntegralCalculator(n)
    Xlist = SimpleRandGenerator(n)
    SimpleRandListMean = mean(f.(Xlist))           # Estimating the Integral value by
simple random points and avergaing the function value pf these points which will be
multiplied by b-a
    sigma = var(f.(Xlist))
    return(SimpleRandListMean*2,sigma)

end
```

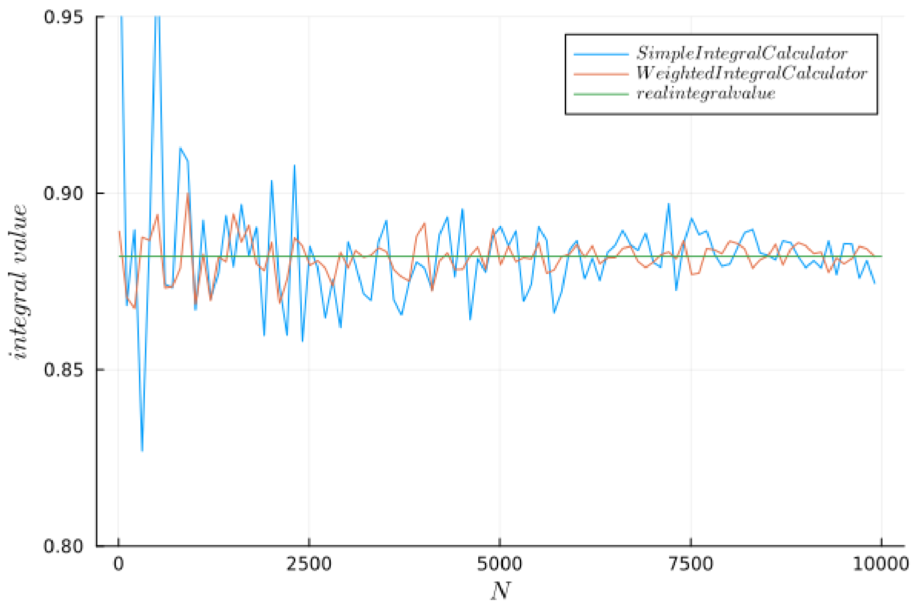
توابع برای محاسبه به روش نمونه گیری هوشمند هم به شکل زیرند، ضرایبی برای نمونه گیری هوشمند در تابع ایجاد اعداد تصادفی و محاسبه انتگرال به کار گرفته شدند. خطا هم در همین توابع محاسبه شده است.

```
function WeightedRandGenerator(n)                            # Generating random
numbers between 0  and 2 with weight e^(-x)
    UniformRandlist = rand(n)
    return(-log.(UniformRandlist.*(1-1/e^2) .+ 1/e^2))
end

function WeightedIntegralCalculator(n)
    Xlist = WeightedRandGenerator(n)
    WeightedRandListMean = mean(f.(Xlist)./(exp.(-Xlist)))       # Calculating Integral
using the weighted random points
    WeightedIntegralValue = WeightedRandListMean* (1-e^(-2))
    #sigma = sqrt(mean((f.(Xlist)./(e.^(-Xlist.^2))^2)-mean(f.(Xlist)./(e.^(-
Xlist.^2))^2)))
    sigma = var(f.(Xlist))
```

```
    return(WeightedIntegralValue,sigma)
end
```

شکل زیر نوسان مقدار تخمین زده شده انتگرال برای تابع مورد نظر به دو روش برای N های متفاوت رسم شده که به وضوح نوسانات کمتر روش هوشمند و نزدیک تر بودنش به مقدار واقعی قابل مشاهده است.



1 مقدار انتگرال در N های مختلف

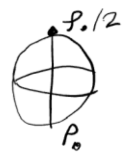
smart	Simp	smart	Simp	smart	Simp	smart	Simple	smart	Simp	
100000	100000	10000	10000	1000	1000	100	100	10	10	
0.882	0.883	0.881	0.883	0.882	0.895	0.884	0.842	0.872	0.940	مقدار انتگرال
0.109	0.119	0.109	0.119	0.110	0.119	0.110	0.118	0.109	0.120	خطای آماری
0	0.001	0.001	0.001	0	0.003	0.002	0.040	0.010	0.058	خطای واقعی
0	0.1	0	0	0.1	0	0	0	0	0	زمان اجرا

در روش هوشمند خطای آماری و واقعی کمتر است و با افزایش تعداد نمونه به مقدار واقعی نزدیک میشویم.

تمرین دوم فصل هفتم

برای به دست آوردن نقاطی تصادفی درون کره، ابتدا در تابع SphereRandGenerator اعدادی تصادفی در مربعی به طول 1 ایجاد کردیم. بعد در حلقه ای نقاطی که فاصله شان از مبدا بزرگتر از 1 است را حذف میکنیم و مابقی را در آرایه ای ذخیره میکنیم. حال به روش نمونه گیری ساده مشابه تمرین قبل با میانگین گیری از مقدارانتگرالی که در ادامه آورده شده(در نقاط تصادفی به دست آمده) و ضرب آن در حجم کره مرکز جرم کره را به دست می آوریم.

$$\int \rho \, x \, dr = x_{cm}$$



$$\frac{\rho_0}{2} = 2Ra$$

$$\frac{\rho_0}{4R} = a$$

$$x = -R \rightarrow \rho = \rho_0 \quad : \quad -aR + b = \rho_0$$

$$b = \rho + aR = \frac{5}{4}\rho_0$$

$$\boxed{\rho = \frac{\rho_0}{4R} x + \frac{5}{4}\rho_0}$$

که در اینجا مرکز جرم با تعداد نمونه ی 10^7، 0.20861 به دست می اید.