Verslag Python Challenge 3

Aiko Decaluwe, Fien Dewit, Viktor van Nieuwenhuize (Groep 4)
Opleiding: Tweede Bachelor Fysica en Sterrenkunde
Vak: Statistiek en Gegevensverwerking Datum: 15 December 2020

1 Inleiding

Voor de gegeven functie:

$$f(x) = \pi x \cos\left(\frac{\pi}{2}x^2\right)$$

passen we de hit and miss methode toe en de inverse cumulatieve. We berekenen de integraal via de gewone Monte-Carlo methode en de stratificatie methode. Verder genereren we ook 2D toevalsgetallen volgens een dichtsheidsfunctie waarna we hiervan een 1D-doorsnede nemen.

2 Implementeer en vergelijk twee samples

2.1 Hit and Miss

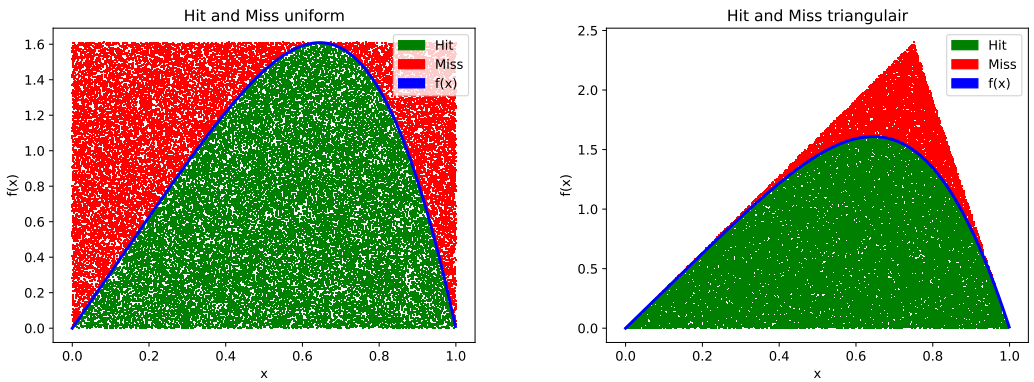
Voor de 'hit and miss' methode werden er 2 verschillende verdelingen gebruikt. Eerst een simpele uniforme verdeling en vervolgens ook een iets complexere triangulaire verdeling.

Om te beginnen implementeren we een 'Hit and Miss' algoritme. Met de 'Hit and Miss' methode worden er eerst samples gegenereerd voor $h(x)$. Hiervoor wordt er een uniforme verdeling gegenereerd. Hierna wordt dit punt vermenigvuldigd met de maximum hoogte die de verdeling heeft op dat punt. Voor de uniforme verdeling is die hoogte constant met $h(x) = 1.61$. Voor de trianguaire verdeling is dit iets complexer en zal $h(x)$ afhangen van x:

$$h(x) = \begin{cases} 3,2x & \text{als } 0 \leq x \leq 0,75 \\ -9,6x + 9.6 & \text{als } 0.75 < x \leq 1 \end{cases}$$

Indien $h(x)$ met 1 wordt vermenigvuldigd krijgen we de maximum hoogte. Indien dit met een ander getal is tussen 0 en 1 zal de y-waarde van dat punt op een bepaald percentage van het maximum liggen.

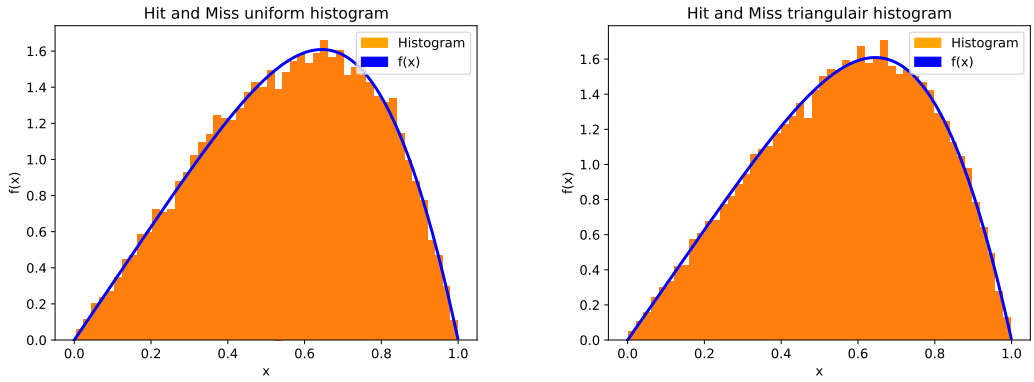
Nadien werd er bepaal dof dit punt onder of boven de functie lag. Een punt dat onder de functie ligt is een 'hit' en werd aangeduid in het groen op figuur 1. Een punt dat boven de functie ligt is een 'miss' en werd aangeduid in het rood.



(a) *Hit and miss met uniforme verdeling h(x)* (b) *Hit and miss met triangulaire verdeling h(x)*

Figuur 1: *Hit or miss met 'hit' de groene punten, 'miss' de rode punten en f(x) de blauwe lijn*

In de figuur is te zien dat de triangulaire verdeling efficiënter is dan de uniforme verdeling aangezien daar de dichtheid van de punten groter is dan bij de uniforme verdeling. M.a.w. het aantal groene punten ten opzichte van het totaal aantal getrokken punten is voor de triangulaire verdeling groter dan voor de uniforme verdeling. Dit zien we ook als we naar de waarden kijken van de efficiëntie, wat gelijk is aan (het aantal nuttige samples)/(totaal aantal samples). Voor de triangulaire verdeling is de efficiëntie $\varepsilon = 0.83$ en voor de uniforme verdeling is dit $\varepsilon = 0.62$.



(a) *Uniforme distributie* (b) *Triangulaire distributie*

Figuur 2: *Histogrammen van de 'Hit and miss' methode voor beide distributies*

Wanneer we nu de histogrammen bekijken voor de 'hit or miss' methode op basis van beide verdelingen (te zien in figuur 2) zien we maar weinig verschil. Dit is niet zo vreemd aangezien we verwachten dat de methode voor beide verdelingen even goed $f(x)$ zal benaderen maar dat de ene dat efficiënter zal doen dan de andere. Algemeen is in figuur 2 ook te zien dat de 'Hit and Miss' methode een goede benadering oplevert voor $f(x)$.

2.2 Inverse Cumulatieve

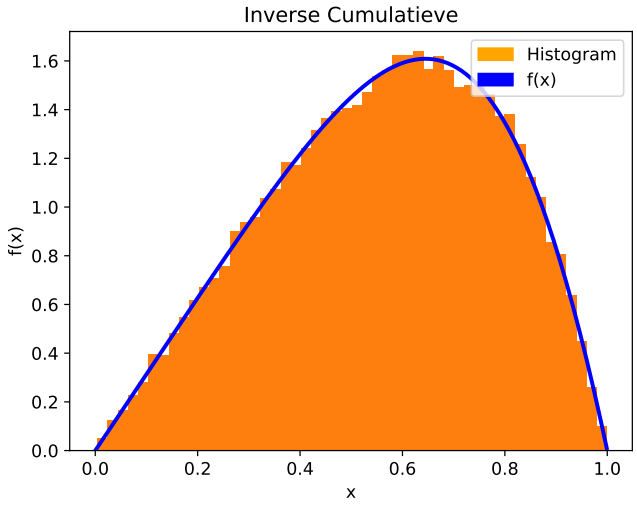
Nu maken we gebruik van de methode van de inverse cumulatieve om samples te genereren. Als eerst rekenen we analytisch de cumulatieve $F(x) = \int_0^x f(x)dx$ en haar inverse uit:

$$\begin{aligned} F(x) &= \int_0^x f(x)dx \\ &= \int_0^x \pi x \cos\left(\frac{\pi}{2}x^2\right) \\ &= \left[\sin\left(\frac{\pi r^2}{2R^2}\right)\right]\Bigg|_0^x \\ &= \sin\left(\frac{\pi}{2}x^2\right) \\ &\implies F(x) = u \\ &\iff R^{-1}(u) = x \end{aligned}$$

En dus is

$$x = \sqrt{\frac{2}{\pi}\arcsin u}$$

Nu worden er opnieuw waarden van u uit een uniforme verdeling gehaald en wordt x steeds berekend aan de hand van bovenstaande vergelijking. Vervolgens wordt er een histogram gemaakt van deze waarden. Ook $f(x)$ wordt opnieuw afgebeeld op het plot.



Figuur 3: Histogram voor inverse cumulatieve methode

In figuur 3 is te zien dat ook de methode van de inverse cumulatieve, net zoals de 'Hit or Miss' methode, een goede benadering is voor $f(x)$. Er lijkt ons weinig verschil in nauwkeurigheid tussen de 3 methoden.

Het verschil in methoden zit hem echter in de tijd die elke methode in beslag neemt voor het genereren van de samples. Hier genereerden we steeds 50 000 samples. Voor de 'Hit and Miss' methode duurde het genereren van deze 50 000 samples 0.0056 seconden wanneer we gebruik maakten van de uniforme verdeling ($= 0.112 \cdot 10^{-6}$ sec/sample) en 0.0078 seconden voor de triangulaire verdeling ($= 0.156 \cdot 10^{-6}$ sec/sample). De uniforme verdeling is dus sneller dan de triangulaire verdeling, wat niet zo verrassend is aangezien er bij de triangulaire verdeling een extra functie wordt aangeroepen om de hoogte te berekenen. Voor de inverse cumulatieve methode duurde het 0.0020 seconden voor het genereren van 50 000 samples ($= 0.040 \cdot 10^{-6}$ sec/sample). Deze waarde voor de tijd is nog sneller dan de 'Hit or Miss' functie met h(x) een uniforme verdeling. Dit is opnieuw niet zo verbazend. Er moet voor de inverse cumulatieve methode namelijk niet voor elk punt gecheckt worden of het punt moet verworpen worden of niet. Het nadeel aan de inverse cumulatieve methode is dan wel het bepalen van $F(x)$ zeer complex kan worden. In ons geval was dit geen probleem, maar voor moeilijkere functies kan dit lastig worden.

We kunnen dus besluiten dat zowel de 'Hit or Miss' methode als de inverse cumulatieve methode de functie even goed benaderen maar dat de inverse cumulatieve methoden dit sneller doet. Verder is bij de 'Hit or Miss' methode de uniforme verdeling de snelste, maar de triangulaire verdeling de efficiëntste.

3 Monte Carlo integratie

We maken een Monte Carlo schatting van $F(u) = \int_0^u f(x)dx$ voor u=1 en u=2. Dit doen we met behulp van 2 verschillende methodes. Eerst passen we de ruwe Monte Carlo methode toe en daarna de stratificatie methode.
Bij de ruwe Monte Carlo methode nemen we n x-waarden die uniform verdeeld zijn tussen de grenswaarden 0 en u. Vervolgens rekenen we de integraal uit met behulp van de volgende benadering:

$$I = \int_a^b f(x)dx \approx \frac{b-a}{n} \sum_{i=1}^{n} f(x_i)$$

x_i werd bepaald door waarden te trekken uit een uniforme verdeling. Deze waarden liggen allemaal tussen 0 en b. In dit geval is $a = 0$ en berekenen we de integraal voor $b = 1$ en $b = 2$.

Voor de stratificatie methode doen we hetzelfde als bij de Monte Carlo methode, maar deze keer verdelen we het x-gebied in twee en schatten we de deelintegralen afzonderlijk met een even groot aantal n-waarden

in elk gebied. We trekken dus 2 keer $\frac{n}{2}$ random samples uit een uniforme verdeling. De eerste set waarden ligt dan tussen 0 en $\frac{b}{2}$. De tweede set waarden zal tussen $\frac{b}{2}$ en b liggen.
Voor n = 100 bekomen we met deze 2 methodes de volgende waarden:

$$I_{MC} = 0.968748006860793 \text{ en } I_{MC} = 0.3534689939437366$$

$$I_S = 1.034379450430744 \text{ en } I_S = 0.0014546984434278488$$

De eerste rij waarden komt overeen met de Monte Carlo methode en de tweede rij waarden met de stratificatie methode voor respectievelijk u=1 en u=2. Doen we dit opnieuw, maar nu voor n = 1000000, dan bekomen we de waarden:

$$I_{MC} = 0.9996848155594446 \text{ en } I_{MC} = 0.0004131261579395382$$

$$I_S = 0.9994513454936259 \text{ en } I_S = 0.009902454082952148$$

De analytisch uitgerekende waarden voor de integralen zijn:

$$I = \int_0^1 \pi x \cos\frac{\pi}{2}x^2 dx = \int_0^1 \cos t dt = \left[\sin(\frac{\pi}{2}x^2)\right]_0^1 = 1$$

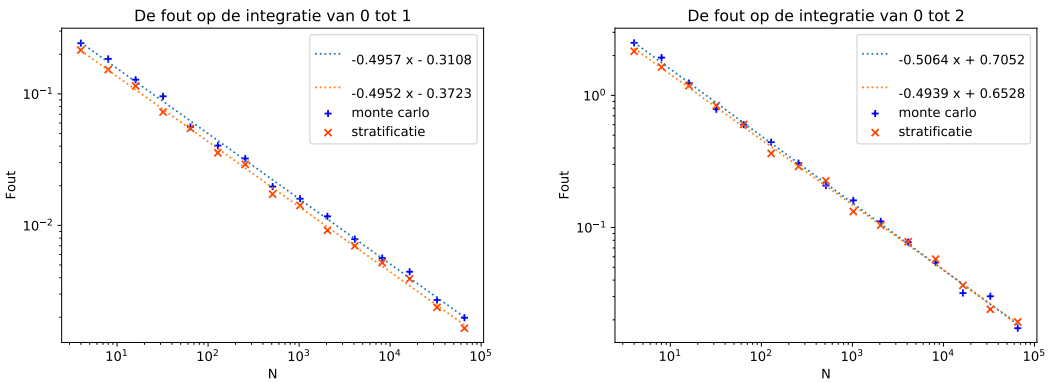
$$I = \int_0^2 \pi x \cos\frac{\pi}{2}x^2 dx = \int_0^2 \cos t dt = \left[\sin(\frac{\pi}{2}x^2)\right]_0^2 = 0$$

Hierin stelden we $t = \frac{\pi}{2}x^2$ zodat $dt = \pi x dx$.
Voor n = 100 zien we dat de geschatte waarden in de beurt van de analytische waarden liggen, maar dat ze enkele honderdsten of tienden kunnen afwijken. Voor n = 1000000 liggen de geschatte waarden zeer dicht bij de analytische waarden en dus is hun verschil zeer miniem. Deze methode is dus pas echt bruikbaar voor een grote n.

We berekenen vervolgens de fout op deze integralen. Dit doen we met behulp van de formule:

$$\hat{\sigma} = \frac{1}{N-1}\sum_{N}^{i=1}(x_i - \overline{x})^2$$

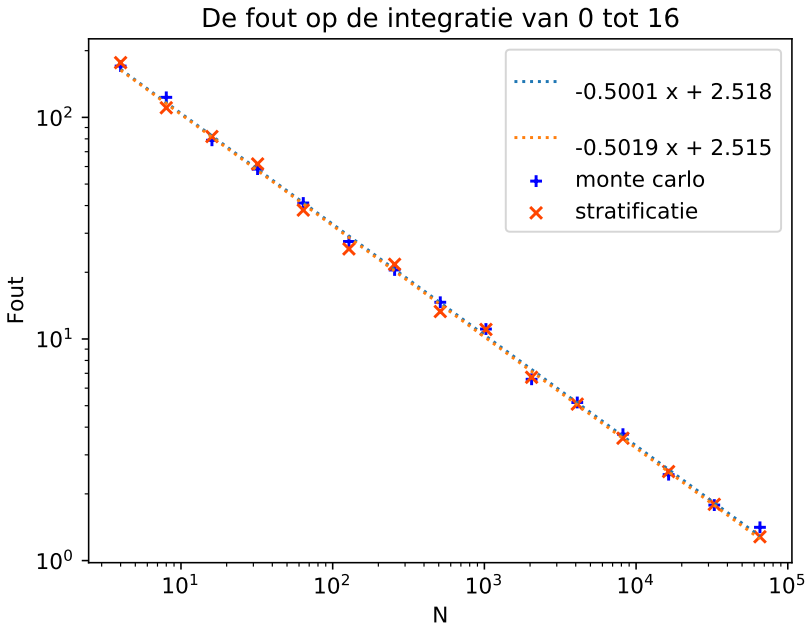
Hierna plotten we deze fouten op een log-log schaal zodat we een lineair verband verkrijgen tussen de fout en het aantal samples N.



(a) *fout op integratie van 0 tot 1*

(b) *fout op integratie van 0 tot 2*

Figuur 4: *fouten op de integratiemethodes*



Figuur 5: fout op integratie van 0 tot 16

In figuur 4 is de fout op de integratie in functie van N afgebeeld voor beide methodes. Er vallen ons al direct enkele dingen op. Eerst en vooral is er een lineair verband op een log-log schaal. Dit betekent dus een machtsverband op een lineaire schaal. Verder zien we dat de fout op de stratificatie een beetje lager ligt dan die op de monte carlo methode. Dit is uiteraard te verwachten. Door het nemen van 2 gebieden zal de fout uiteraard verminderen. Ten slotte zien we dat de fout ongeveer 1 grote-orde groter is bij de integratie van 0 tot 2 dan die is bij de integratie van 0 tot 1. Dit is ook vrij logisch, hoe groter het integratie gebied, hoe meer fouten er zullen optreden. Wanneer we het integratiegebied uitbreiden en integreren van 0 tot 16 zien we dat deze trend zich voortzet (figuur 5). De fout is enkele grote-ordes groter dan die in figuur 4. Ook zien we dat er nog weinig verschil is tussen beide trendlijnen. We kunnen dus ook stellen dat als het gebied vergroot, het verschil in precisie tussen de methodes verkleint.

Ten slotte kunnen we nog naar de trendlijn kijken. Deze heeft een rico van $-\frac{1}{2}$. Aangezien dit een log-log schaal is betekent dat dus dat de fout zal afnemen via $\frac{1}{\sqrt{N}}$. Dit is ook wat we verwachten uit stelling 10.5.1 van de syllabus.

4 2D Toevalsgetallen

Er werden N gebeurtenissen getrokken uit een cirkel met straal R, volgens de dichtheidsfunctie:

$$\rho(r, R) = \frac{\pi r}{R^2} \cos\left(\frac{\pi r^2}{2R^2}\right)$$

Er werd een set poolcoördinaten gegenereerd (r, θ). Voor θ werd een uniforme distributie gebruikt tussen 0 en 2π. De waarden van r werden uit $\rho(r, \theta)$ gegenereerd. Hiervoor werd de inverse cumulatieve methode gebruikt.

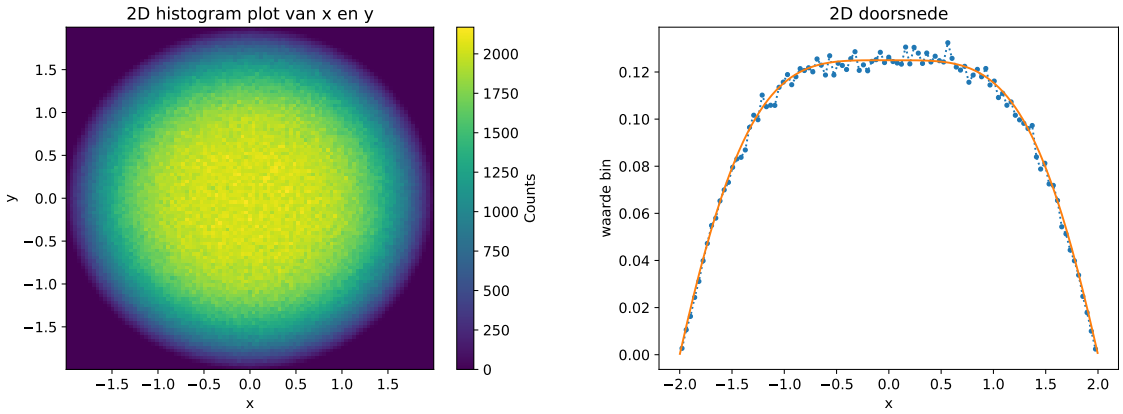
$$
\begin{aligned}
R(r) &= \int_0^r \frac{\pi r}{R^2} \cos\left(\frac{\pi r^2}{2R^2}\right) dr \\
&= \left[\sin\left(\frac{\pi r^2}{2R^2}\right)\right]\Bigg|_0^r \\
&= \sin\left(\frac{\pi r^2}{2R^2}\right) \\
&\implies R(r) = u \\
&\iff R^{-1}(u) = r \\
&\iff r = \sqrt{\frac{2R^2}{\pi}\arcsin(u)} \\
&\iff r = R \cdot F^{-1}(u)
\end{aligned}
$$

We kunnen zien dat deze uitkomst R keer de inverse cumulatieve van de oorspronkelijke functie is. Vervolgens werden deze punten omgezet naar cartesische coördinaten. Dit gebeurt via:

$$
\begin{cases}
x &= r\cos(\theta) \\
y &= r\sin(\theta)
\end{cases}
$$

Vervolgens kan er een 2D histogram aangemaakt worden, gebruik makend van de set waarden voor x en voor y. Hiervoor werden 99^2 bins gebruikt. Hierna werd er een 1D doorsnede door het midden van deze bins gemaakt. Ten slotte werd ook de theoretische curve samen met de 1D doorsnede geplot. De theoretische curve heeft een iets andere vorm dan wat we gebruikten voor de samples te genereren. De functie ziet er als volgt uit:

$$
\rho(r, \theta, R) = \frac{1}{2R^2} \cos\left(\frac{\pi r^2}{2R^2}\right)
$$

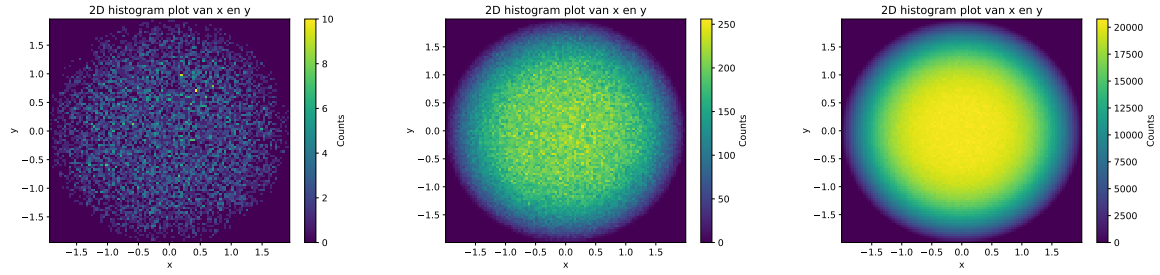


(a) *2D histogram* (b) *1D doorsnede*

Figuur 6: Voorstellingen van de 2D toevalsgetallen voor $N = 10^7$

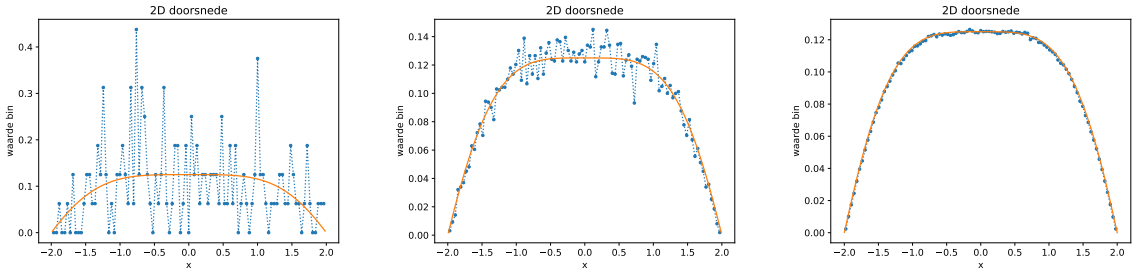
In de bovenstaande figuur is zowel de gemaakte 2D histogram als de 1D doorsnede zichtbaar. In het 2D histogram zien we een vrij goede benadering van wat we zouden krijgen wanneer we de theoretische functie voor de 1D doorsnede over 180° zouden roteren. Ook is het duidelijk dat de random samples een relatief goede benadering is van de theoretische functie.

Indien we het nu nog een stapje verder nemen en deze figuren genereren voor verschillende N waarden waarbij N het aantal random getrokken samples zijn, krijgen we de volgende figuren:



(a) $N = 10^4$ (b) $N = 10^6$ (c) $N = 10^8$

Figuur 7: *Het 2D histogram voor verschillende N*



(a) $N = 10^4$ (b) $N = 10^6$ (c) $N = 10^8$

Figuur 8: *De 1D doorsnede voor verschillende N*

Uit bovenstaande figuren zien we een zeer duidelijke trend. De benadering wordt veel beter naarmate N omhoog gaat. In de benadering voor $N = 10^4$ kan de theoretische functie eigenlijk niet herkend worden. Voor $N = 10^8$ is de benadering al zeer goed en zijn er nog weinig fluctuaties te zien. Dit uitvoeren voor een nog grotere N was echter niet mogelijk door een tekort aan RAM-geheugen.