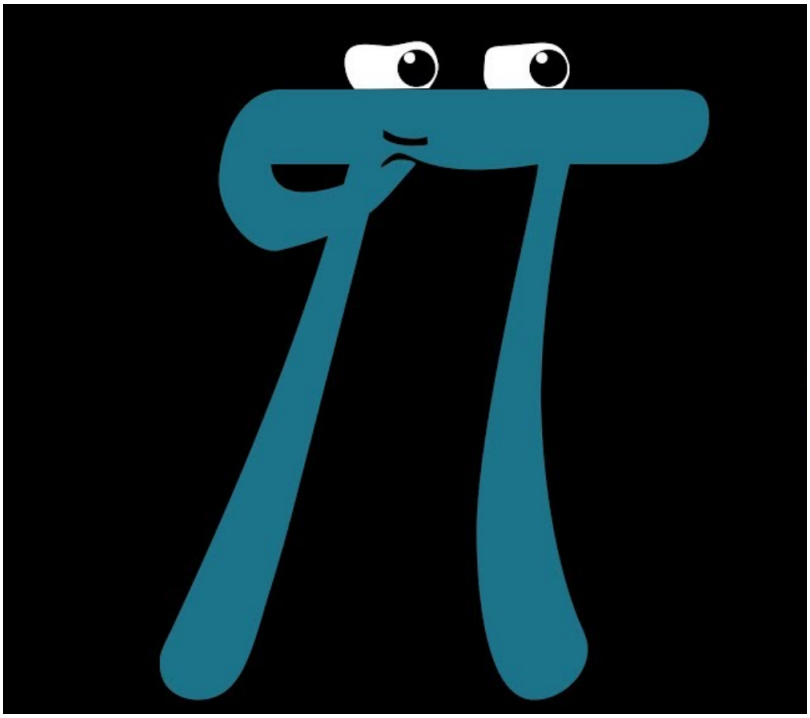
A poor duo's approximation to π

Pim Meulensteen, Dim Diemer

February 2, 2020

Computeralgebra-LaTeX

Begeleiding: Ms. Majken T. Roelfszema MSc, Reijer Boodt



Korteweg-de Vries Instituut voor Wiskunde
Faculteit der Natuurwetenschappen, Wiskunde en Informatica
Universiteit van Amsterdam

Abstract

Naar aanleiding aanleiding van het artikel *"A Poor person's approximation to π"*, hebben we gekeken naar andere (historische) manierer op π te benaderen. Met *Mathematica* hebben we de tijd per decimaal per algortime gemeten. We kunnen concluderen dat de manier uit het hiervoor genoemde artikel een zéér matige methode is.

Titel: A poor duo's approximation to π
Auteurs:
Pim Meulensteen, pim.meulensteen@student.uva.nl, 12751510
Dim Diemer, dim.diemer@student.uva.nl 12679674
Begeleiding: Ms. Majken T. Roelfszema MSc, Reijer Boodt
Tweede beoordelaars: Ms. Majken T. Roelfszema MSc, Reijer Boodt
Einddatum: February 2, 2020

Korteweg-de Vries Instituut voor Wiskunde
Universiteit van Amsterdam
Science Park 904, 1098 XH Amsterdam
`http://www.kdvi.uva.nl`

Contents

1. Inleiding

Het artikel *"A Poor person's approximation to π"* [1] gaat over het bereknenen van π op een bijzonder onhandige manier. De premisse van het artikel is dat er een computer, die enkel kan delen. De vraag die wordt beantwoord is of het met deze computer nog steeds mogelijk is om π te benaderen. Hoe dit gedaan wordt volgt in aanpak.

"A Poor person's approximation to π" begint met een tweetal andere zaken: delen met behulp van modulorekenen en de benadering van e(ulers getal).

Dit eerste is een alternatief voor delen. Dit is met het oog op de snelheid een zeer matig alternatief om te delen zonder delen. Deze manier wordt komt in de rest van het artiekel ook niet van daarom ook niet terug.

Voor e wordt de formule

$$(1 + 1/n)^n < e < (1 + 1/n)^{n+1}$$

omgeschreven naar de boven en ondergrens

$$t_m = n(n + 1)^n; T_m = (n + 1)^{(n+1)};$$

repsectievelijk. Het verschil tussen deze is dan maximaal $n^n e$. Op deze manier kan er zonder te delen tóch een benadering voor e krijgen.

Geïnspireerd door deze manier van berekenen zijn wij op zoek gegaan naar andere benaderingen van π. We ondezoeken hoe verschillenden manierne voor het benaderen van π zich met elkaar vergelijken. Dit doen we om een prespectief te krijgen hoe "goed" het algoritme uit het artikel is. Dit is gedaan met behulp van *Mathematica*. We duiken dieper in het onderzoek naar het berekenen van de decimalen van π in groot aantal.

We hebben met enigzins willekeurig de volgende methodes gekozen om nader te onderzoeken:

- Ramanujan series (Ramanujan)
- Madhava series (Madhava)
- Alternating series (Leibniz)
- Deelloos (Shiu)

2. Aanpak

We gebruiken de RepeatedTiming-functie van *Mathematica* om te bepalen hoe efficiënt een functie is in het bepalen van π. RepeatedTiming neemt het gemiddelde van een aantal runs van de functie tussen de haakjes en geeft een getal in seconden. Dit kunnen we in een plot zetten en vergelijken met de snelheid van andere methodes. Deze methodes worden hieronder beschreven.

Hierbij hebben we een eigen functie precice gedefinieerd, die aantal correcte decimalen van π return. Deze code, evenals de code voor de andere methodes, is opgenomen in de bijlage.

2.1. Shiu

De manier uit het artikel (van Shiu) baseerd zich op de verhouding tussen de straal en het oppervlakte. Door roosterpunten die op de cirkel liggen uit rekenen, en vervolgens boven en ongrensen voor de bijbehoorende Riemansommen uit te rekenen. Al dit kan in principe zonder delen.

Het tweede deel van het artikel gaat over de analyze en eventuele optmimalizatie, maar is ook veel moeilijker. Wij bezaten niet het wiskundige begrip of tijd om ons hier in te verdiepen, maar dit achten wij ook niet geheel nodig.

2.2. Ramanujan

De Ramanujan-serie, ookwel Ramanujan-Sato benadering, is een benadering van π op basis van een oneindige som, gegeven als

$$\frac{1}{\pi} = \frac{2\sqrt{2}}{99^2} \sum_{k=0}^{\infty} \frac{(4k)!}{k!^4} \frac{26390k + 1103}{396^{4k}}.$$

Deze som blijkt ongeveer exact acht decimalen preciezer p. Dit bizarre gerdag heeft te maken met de term 396^{4k} in de noemer. Dit is te schrijven als $4^{4k} \cdot 99^{4k}$. Daarnaast is $99^{4k} \approx 100^{4k} = 10^{8k}$ zodat $1/100^{4k}$ nagenoeg altijd acht decimalen precies geeft.[1]

Deze benadering is gebruikt om met super- en clustercomputers miljarden decimalen van π te berekenen. Het opduiken van deze formule kwam onverwachts en het is nog onbekend hoe Ramanujan bij de formule kwam. Ramanujan zelf claimde dat een Indiase godin tot hem kwam in een droom.

2.3. Madhava

De Madhavareeks voor π komt uit de versen 2.206-2.209 van de "Tantrasamgraha-vyakhy". Dit is een commentaar op een eerder werk geschreven door Sankara Variar rond het jaar

[1] Bron: https://math.stackexchange.com/questions/908535/ramanujans-approximation-for-pi

1500. Hij baseerde zijn uitdrukking op de taylorreeks voor de arctangens. Hier is de (naar het Engels) vertaalde versie:

"By means of the same argument, the circumference can be computed in another way too. That is as (follows): The first result should by the square root of the square of the diameter multiplied by twelve. From then on, the result should be divided by three (in) each successive (case). When these are divided in order by the odd numbers, beginning with 1, and when one has subtracted the (even) results from the sum of the odd, (that) should be the circumference."

In moderne taal: voor een cirkel met omtrek c en diameter d geldt

$$c = \sqrt{12d^2} - \frac{\sqrt{12d^2}}{3 \cdot 3} + \frac{\sqrt{12d^2}}{3^2 \cdot 5} - \frac{\sqrt{12d^2}}{3^3 \cdot 7} + \dots.$$

Gezien $c = \pi d$, is dit te herschrijven tot een formule voor π, namelijk

$$\pi = \sqrt{12} \sum_{k=0}^{\infty} \frac{(-3)^{-k}}{2k+1} = \sqrt{12} \left(1 - \frac{1}{3 \cdot 3} + \frac{1}{5 \cdot 3^2} = \frac{1}{7 \cdot 3^3} + \dots\right).$$

2.4. Leibniz

Leibniz' versie van π komt uit de taylorreeks van de arctangens, namelijk:

$$\arctan(x) = x - \frac{x^3}{3} + \frac{x^5}{5} - \frac{x^7}{7} + \frac{x^9}{9} - \dots$$

Voor $x = 1$ geldt $\arctan(x) = \pi/4$ in. Uitgeschreven in de Taylorreeks geeft dit de volgende vegelijking:

$$\frac{\pi}{4} = \sum_{k \geq 0} \frac{(-1)^k}{2k+1} = 1 - \frac{1}{3} + \frac{1}{5} - \frac{1}{7} + \frac{1}{9} - \dots$$

Deze manier van π benaderen is verschrikkelijk inefficiënt en het kost honderden termen om pi te benaderen. Na 100 termen is $\pi \approx 3.15149$ volgens de som, een afwijking van zo'n 1%.

3. Resultaten

We hebben de resultaten verwerkt in twee grafieken, om een goed beeld te geven van schaal.

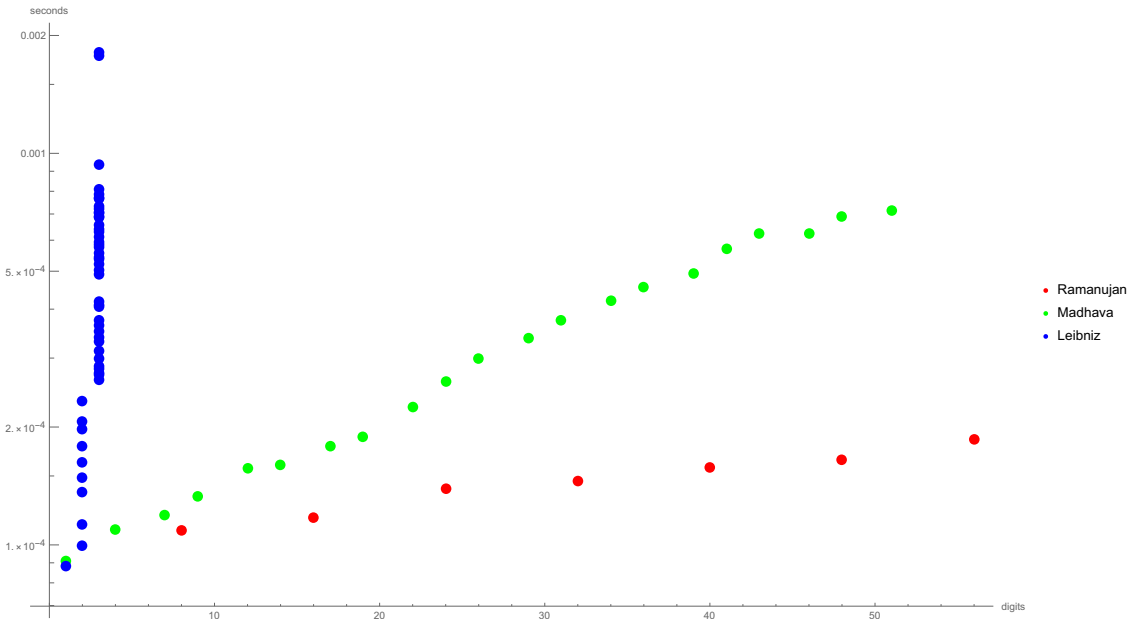


Figure 3.1.: Logaritmische grafiek van tijd uitgezet tegen aantal decimalen berekend door verschillende algoritmes π te benaderen.

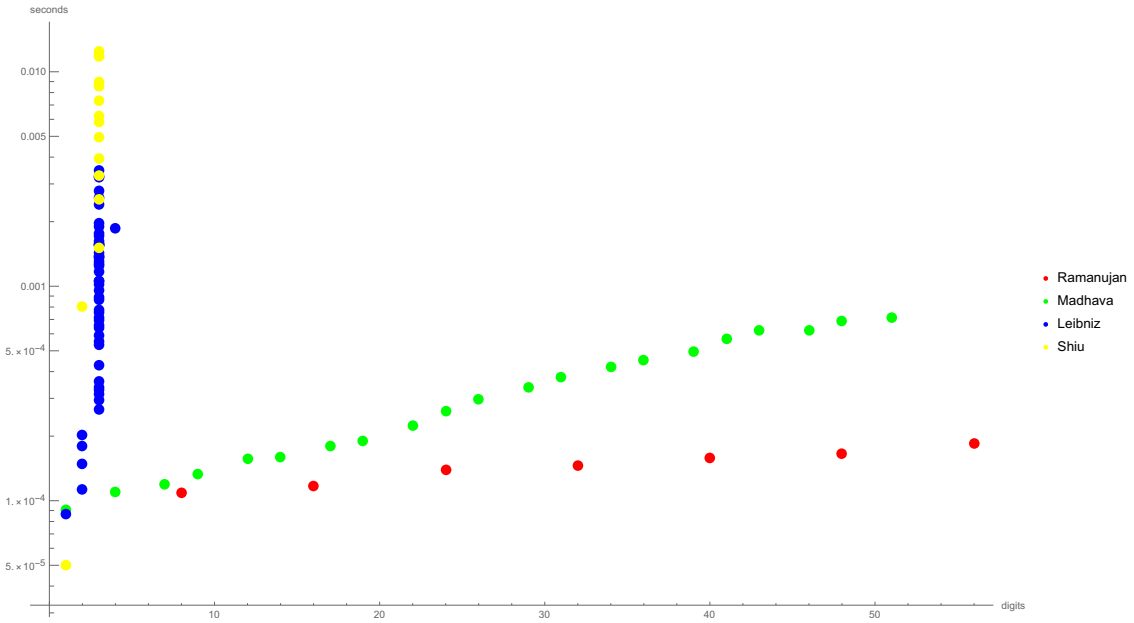


Figure 3.2.: Logaritmische grafiek van tijd uitgezet tegen aantal decimalen berekend door verschillende algoritmes om π mee te benaderen, inclusief het algoritme van Shiu in geel.

4. Conclusie

Van de gekozen methodes, blijk de methode van Ramanujan het snelst. Dit is eigenlijk als verwacht. Door de combinaite van grote getallen (die voor een computer geen probleem zijn) en weinig operaties (wat wel een probleem kan zijn) is deze sneller dan de andere. Vooral de formule van Leibniz is inefficient, door het vele delen en optellen.

Zoals verwacht is de methode beschreven in het artikel het traags. Het is uiteindelijk gewoonweg geen snel algoritme. De formule van Leibniz is niet veel sneller, door het vele delen: iets wat een computer niet altijd snel kan. Het idee van de auteurs is dus ook niet geheel onlogisch: als het delen veel "kost", waarom proberen we dit niet te vermijden? Helaas werkt dit in de praktijk niet zo.

Bibliography

[1] Shiu, Daniel, Shiu, Peter. *"A poor person's approximation to pi."* The Mathematical Gazette Vol. 96, No. 537 (November 2012): pp. 408-414.

A. Mathematicacode

```
precise[n_] := -1* Floor[Log[10, Abs[N[Pi, 2000] - N[n, 2000]]]]
ramanujan[m_] := 1/N[((2 Sqrt[2])/(9801)) *Sum[((4 k)!  (1103 + 26390 k))/(((k!)^4)
(396^(4 k))),{k, 0, m} ], 2000];
madhava[l_] := N[Sqrt[12]*Sum[((-3)^(-k))/(2 k + 1), {k, 0, l } ], 2000]
leibniz[z_] := N[4*Sum[((-1)^(k))/(2 k + 1), {k, 0, z} ], 2000]
shiu[r_] := N[(Sum[Sqrt[r^2 - i^2], {i, 1, r}]*4)/r^2]
```

Appendix A.1 *Mathematica*-functies voor het testen van de methodes.