

Viele Wege führen zu π

Von Cleve Moler

Die Berechnung der Zahl π auf mehrere hundert oder sogar Billionen von Stellen war lange ein probates Mittel, Stresstests an Hardware vorzunehmen, Software zu validieren oder ganz einfach auf sich aufmerksam zu machen. Die MATLAB®-Implementierungen der am häufigsten zur Berechnung von π eingesetzten Algorithmen sind exemplarisch für zwei ganz unterschiedliche Typen von Arithmetik, die in der Symbolic Math Toolbox™ zur Verfügung stehen: exakte rationale Arithmetik und Fließkommaarithmetik mit variabler Genauigkeit.

Im vergangenen August verkündeten die Hobbyinformatiker Alexander Yee und Shigeru Kondo einen neuen Weltrekord mit der Bestimmung von 5 Billionen Stellen von π (Abbildung 1). Ihre Berechnung dauerte 90 Tage auf einem Computer-Eigenbau. Yee absolviert derzeit ein Graduiertenstudium an der University of Illinois. Seine Rechensoftware „y-cruncher“ entstand aus einem Schulprojekt an der Palo Alto High School. Kondo ist Systemingenieur und baut sich zu Hause in Japan seine PCs selbst auf. Der Rechner, den er für dieses Projekt zusammengestellt hat (Abbildung 2), verfügt über zwei Intel® Xeon®-Prozessoren mit insgesamt 12 Kernen, 96 Gigabyte RAM und 20 externe Festplatten mit einer Gesamtkapazität von 32 Terabyte.

Die Berechnung von 5 Billionen Stellen ist eine gigantische Datenverarbeitungsaufgabe, bei der die benötigte Zeit für den Datentransfer genauso entscheidend ist wie die eigentliche Rechenzeit.

y-cruncher nutzt mehrere verschiedene Formeln zur Berechnung und Verifikation. Sein Herzstück ist eine eindrucksvolle Formel, die 1987 von David und Gregory Chudnovsky entdeckt wurde:

$$\frac{1}{\pi} = 12 \sum_{k=0}^{\infty} (-1)^k \frac{(6k)! (13591409 + 545140134k)}{(3k)! (k!)^3 (640320)^{3k+3/2}}$$

Die Brüder Chudnovsky, denen zwei interessante Artikel im Magazin *The New Yorker* sowie ein NOVA-Dokumentarfilm gewidmet wurden, haben ebenfalls ihre eigenen Computer in ihrer Wohnung in Manhattan zusammengebaut. Der Massenparallelrechner, den sie in den 90er Jahren aus handelsüblichen Teilen zusammengestellt haben, galt damals als Supercomputer.

Hochexakte Arithmetik in der Symbolic Math Toolbox

Die Symbolic Math Toolbox bietet zwei Arten von hochexakter Arithmetik: `sym` und `vpa`.

Die Funktion `sym` initiiert die *exakte rationale Arithmetik*: Arithmetische Größen werden durch Quotienten sowie durch Wurzeln großer Ganzzahlen dargestellt. Die Integer-Länge wird je nach Bedarf vergrößert und ist ausschließlich durch die Rechenzeit und

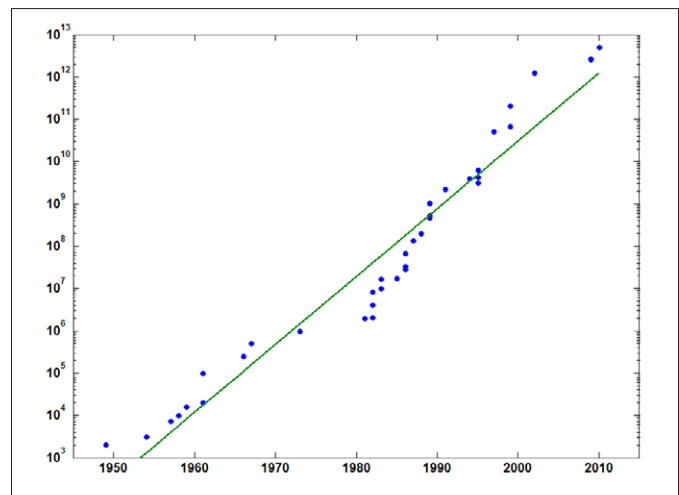


Abb. 1: Logarithmische Darstellung der im Wikipedia-Artikel „Chronology of computation of pi“ aufgelisteten Daten. Sie zeigt, wie sich der Weltrekord der Anzahl berechneter Stellen in den letzten 60 Jahren entwickelt hat. Die lineare Anpassung an den Logarithmus verrät, dass hier eine Variante des Mooreschen Gesetzes erfüllt ist. Die Stellenzahl verdoppelt sich im Schnitt alle 22,5 Monate.

den benötigten Speicher begrenzt. Quotienten und Wurzeln werden nur berechnet, wenn das Ergebnis eine exakte Ganzzahl ist.

Die Funktion `vpa` dagegen ruft die *Fließkommaarithmetik mit variabler Genauigkeit* auf: Hier werden arithmetische Größen durch Dezimalbrüche mit einer bestimmten Stellenzahl sowie eine angehängte Zehnerpotenz dargestellt.

Arithmetische Operationen, auch Divisionen und Wurzeln, können Rundungsfehler aufweisen, deren Größe durch die gewählte Genauigkeit bestimmt ist. So erzeugt etwa die MATLAB-Anweisung

```
p = rat(pi)
```

drei Terme der Näherung von π durch einen Kettenbruch

```
p = 3 + 1/(7 + 1/16)
```

Die Anweisung

```
sym(p)
```

liefert danach den rationalen Ausdruck

```
355/113
```

der eine reizvolle Alternative zum sonst üblichen Bruch $22/7$ darstellt. Dagegen ergibt die Anweisung

```
vpa(p, 8)
```

den achtstelligen Fließkommawert

```
3,1415929
```

Der Chudnovsky-Algorithmus

Die vorstehenden MATLAB-Programme implementieren nur drei aus hunderten möglicher Algorithmen zur Berechnung von π . Der Chudnovsky-Algorithmus (Abbildung 3) bedient sich bis auf den letzten Schritt der exakten rationalen Arithmetik. Jeder Ausdruck der Chudnovsky-Reihe liefert etwa 14 Dezimalstellen, d. h. um d Stellen zu erhalten, benötigt man $\lceil d/14 \rceil$ Ausdrücke. Die Anweisung

```
chud_pi(42)
```

erzeugt beispielsweise durch exakte rationale Berechnung

$$\frac{3405449678154416971853498155008000000\sqrt{640320}}{867407410133324147761288805130794983129}$$

Der letzte Schritt erzeugt dann mithilfe von Fließkommaarithmetik mit variabler Genauigkeit das 42-stellige Ergebnis

```
3,14159265358979323846264338327950288419717
```

Eine interessante Besonderheit in den Ziffern von π offenbart sich durch

```
chud_pi(775)
```

oder

```
vpa(pi, 775)
```

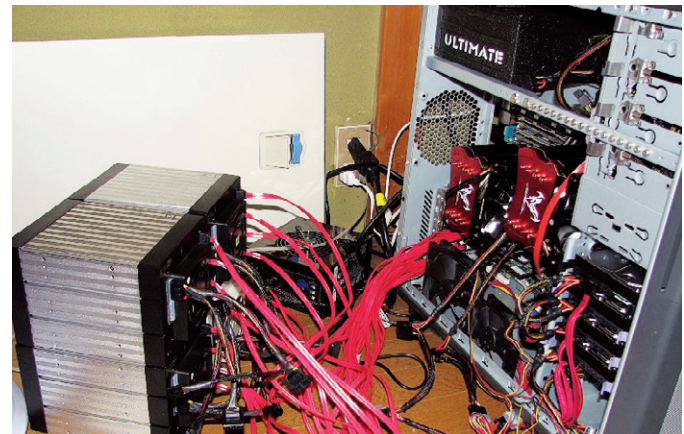


Abb. 2: Shigeru Kondos Eigenbau-PC, der derzeit den Weltrekord der berechneten Stellen von π hält.

```
function P = chud_pi(d)
% CHUD_PI Chudnovsky algorithm for pi.
% chud_pi(d) produces d decimal digits.

k = sym(0);
s = sym(0);
sig = sym(1);
n = ceil(d/14);
for j = 1:n
    s = s + sig * prod(3*k+1:6*k)/prod(1:k)^3 * ...
        (13591409+545140134*k) / 640320^(3*k+3/2);
    k = k+1;
    sig = -sig;
end
S = 1/(12*s);
P = vpa(S,d);
```

Abb. 3: MATLAB-Implementierung des Chudnovsky-Algorithmus. `sym` initiiert darin die exakte rationale Arithmetik.

Beide erzeugen 775 Stellen, die jeweils anfangen und enden mit:

```
3,141592653589793238 ..... 707211349999998372978
```

Wir erkennen hier nicht nur, dass `chud_pi` korrekt arbeitet, sondern auch, dass um die Position 765 in der Dezimalentwicklung von π sechsmal hintereinander die Ziffer 9 erscheint (Abbildung 4).

Die Berechnung von 5000 Stellen mit `chud_pi` erfordert 358 Ausdrücke und dauert auf meinem Laptop ungefähr 20 Sekunden. Der dabei erzeugte symbolische Ausdruck enthält 12.425 Zeichen.

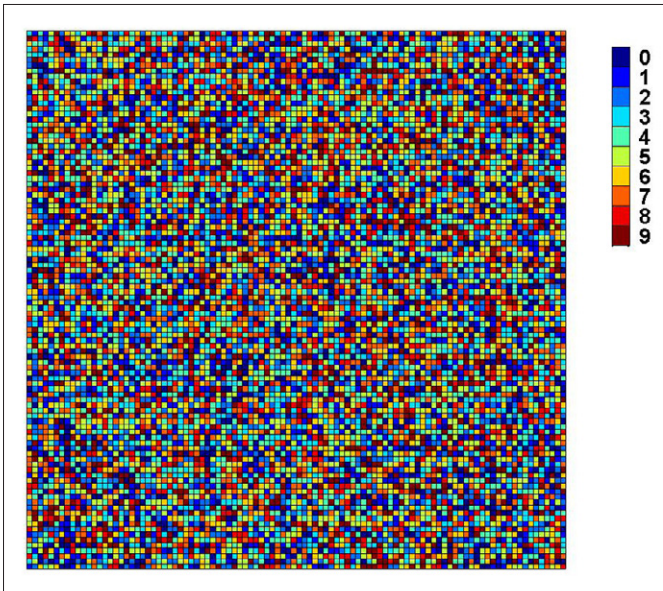


Abb. 4: 10.000 Stellen von π in einer MATLAB-Grafik. Erkennen Sie die sechs aufeinanderfolgenden 9er in der achten Reihe?

Der Algorithmus des algebraisch-geometrischen Mittels

Die Chudnovsky-Formel ist eine *Potenzreihe*: Jeder neue Ausdruck der Partialsumme vergrößert die Stellenzahl um einen bestimmten Betrag. Der Algorithmus für das algebraisch-geometrische Mittel funktioniert völlig anders; er ist *quadratisch konvergent*: Jede neue Iteration verdoppelt die Zahl der korrekten Stellen. Dieser Algorithmus hat eine lange Geschichte, die bis auf Gauss und Legendre zurückgeht. Dass er sich auch zur Berechnung von π eignet, haben 1975 Richard Brent und Eugene Salamin unabhängig voneinander entdeckt.

Das arithmetische Mittel zweier Zahlen, $(a+b)/2$, ist immer größer als ihr geometrisches Mittel, \sqrt{ab} . Ihr arithmetisch-geometrisches Mittel, $agm(a,b)$, wird durch wiederholte Ermittlung arithmetischer und geometrischer Mittelwerte berechnet. Ausgehend von $a_0 = a$ und $b_0 = b$, iteriert man

$$a_{n+1} = \frac{a_n + b_n}{2}$$

$$b_{n+1} = \sqrt{a_n b_n}$$

bis a_n und b_n mit gewünschter Genauigkeit übereinstimmen. Das arithmetische Mittel von 1 und 9 ist beispielsweise 5, ihr geometrisches Mittel 3 und das agm ist 3,9362.

Die mächtigen Eigenschaften des agm liegen nicht nur im Endergebnis, sondern auch in den errechneten Zwischenergebnissen.

Die Berechnung von π ist nur ein Beispiel dafür. Man berechnet dazu

$$M = agm(1, 1/\sqrt{2})$$

Während der Iteration überwacht man die Änderungen

$$c_n = a_{n+1} - a_n$$

Schließlich sei

$$S = \frac{1}{4} - \sum_{n=0}^{\infty} 2^n c_n^2$$

Hieraus folgt

$$\pi = \frac{M^2}{S}$$

Unsere MATLAB-Funktion für den agm-Algorithmus (Abbildung 5) arbeitet von Anfang an mit Fließkommaarithmetik mit variabler Genauigkeit. Würden wir symbolische rationale Arithmetik verwenden, dann erhielten wir eine verschachtelte Reihe von Quadratwurzeln. Nach nur zwei Iterationen erhielten wir

$$\frac{\left(\frac{\sqrt{\sqrt{2}}}{2} + \frac{\sqrt{2}}{8} + \frac{1}{4}\right)^2}{\frac{1}{4} - \left(\frac{\sqrt{2}}{4} - \frac{1}{2}\right)^2 - 2\left(\frac{\sqrt{2}}{8} - \frac{\sqrt{\sqrt{2}}}{2} + \frac{1}{4}\right)^2}$$

Der Dezimalwert dieses Ungetüms ist 3,14168 und noch keine besonders gute Annäherung an π . Die Komplexität verdoppelt sich mit jeder weiteren Iteration.

Solche exakten symbolischen Ausdrücke sind rechnerisch unhandlich und ineffizient. Bei `vpa` dagegen erkennt man sofort die quadratische Konvergenz an den aufeinanderfolgenden Iterationen:

```
3,2
3,142
3,1415927
3,141592653589793
3,1415926535897932384626433832795
3,1415926535897932...79502884197169399375105820974944592
```

Der sechste Eintrag in der Liste hat bereits 64 korrekte Stellen, der

```

function P = agm_pi(d)
% AGM_PI Arithmetic-geometric mean for pi.
% agm_pi(d) produces d decimal digits.

digits(d)
a = vpa(1,d);
b = 1/sqrt(vpa(2,d));
s = 1/vpa(4,d);
p = 1;
n = ceil(log2(d));
for k = 1:n
    c = (a+b)/2;
    b = sqrt(a*b);
    s = s - p*(c-a)^2;
    p = 2*p;
    a = c;
end
P = a^2/s;

```

Abb. 5: MATLAB-Implementierung des Algorithmus für das arithmetisch-geometrische Mittel. `digits` und `vpa` initiieren die Fließkommaarithmetik mit variabler Genauigkeit.

zehnte 1024.

Durch seine quadratische Konvergenz ist `agm` sehr schnell. Um auf 100.000 Stellen zu kommen, benötigt mein Laptop nur 17 Schritte und ca. 2,5 Sekunden.

Die BBP-Formel

Yee hat seine Berechnungen stichprobenartig mit der BBP-Formel überprüft. Diese nach David Bailey, Jonathan Borwein und Simon Plouffe benannte und von Plouffe im Jahr 1995 entdeckte Formel ist eine Potenzreihe, die auf den inversen Potenzen von 16 fußt:

$$\pi = \sum_{k=0}^{\infty} \left(\frac{4}{8k+1} - \frac{2}{8k+4} - \frac{1}{8k+5} - \frac{1}{8k+6} \right) \left(\frac{1}{16} \right)^k$$

Bemerkenswert an der BBP-Formel ist, dass man mit ihr mehrere aufeinanderfolgende Hexadezimalstellen in der Hexadezimalerweiterung von π berechnen kann, ohne die vorherigen Stellen zu kennen und ohne die Genauigkeit mithilfe zusätzlicher arithmetischer Methoden festzulegen. Eine Formel für die Hexadezimalstellen ab der Position d erhält man ganz einfach, indem man die BBP-Formel mit 16^d multipliziert und nur die Nachkommastellen des Ergebnisses betrachtet.

Ich kann nicht widerstehen, dieses Corner mit der Bemerkung zu schließen, dass uns BBP immer nur ein Stückchen des Gesamtbildes von π zeigt, sozusagen „just a piece of π “ (pie: Kuchenstück).

Unser BBP-Programm ist zu lang, um hier aufgeführt zu werden, aber die drei Programme `chudovsky_pi`, `agm_pi` und `bbp_pi` können auf MATLAB Central heruntergeladen werden. ■

Literatur

Alexander J. Yee und Shigeru Kondo, “5 Trillion Digits of Pi,” 2. August 2010.
www.numberworld.org/misc_runs/pi-5t/details.html

Richard Preston, “The Mountains of Pi,” *The New Yorker*, 2. März 1992.
www.newyorker.com/archive/1992/03/02/1992_03_02_036_TNY_CARDS_000362534

Richard Preston, “Capturing the Unicorn,” *The New Yorker*, 11. April 2005.
www.newyorker.com/archive/2005/04/11/050411fa_fact

“Brothers Chudnovsky,” PBS, NOVA, 26. Juli 2005.
www.pbs.org/wgbh/nova/physics/chudnovsky-math.html

David Bailey, Jonathan Borwein, Peter Borwein und Simon Plouffe, “The Quest for Pi,” *Mathematical Intelligencer* 19, S. 50-57, 1997.
<http://crd.lbl.gov/~dhbailey/dhbpapers/pi-quest.pdf>

Jonathan M. Borwein und David H. Bailey, *Mathematics by Experiment: Plausible Reasoning in the 21st Century*, AK Peters, Natick, MA, 2008.

“Chronology of computation of pi,” *Wikipedia*, 9. März 2011.
http://en.wikipedia.org/wiki/Chronology_of_computation_of_pi

Weitere Informationen

Download: MATLAB-Implementierungen der Chudnovsky-, agm- und BBP-Algorithmen
www.mathworks.de/computing-pi

Cleve's Corner Collection
www.mathworks.de/clevescorner