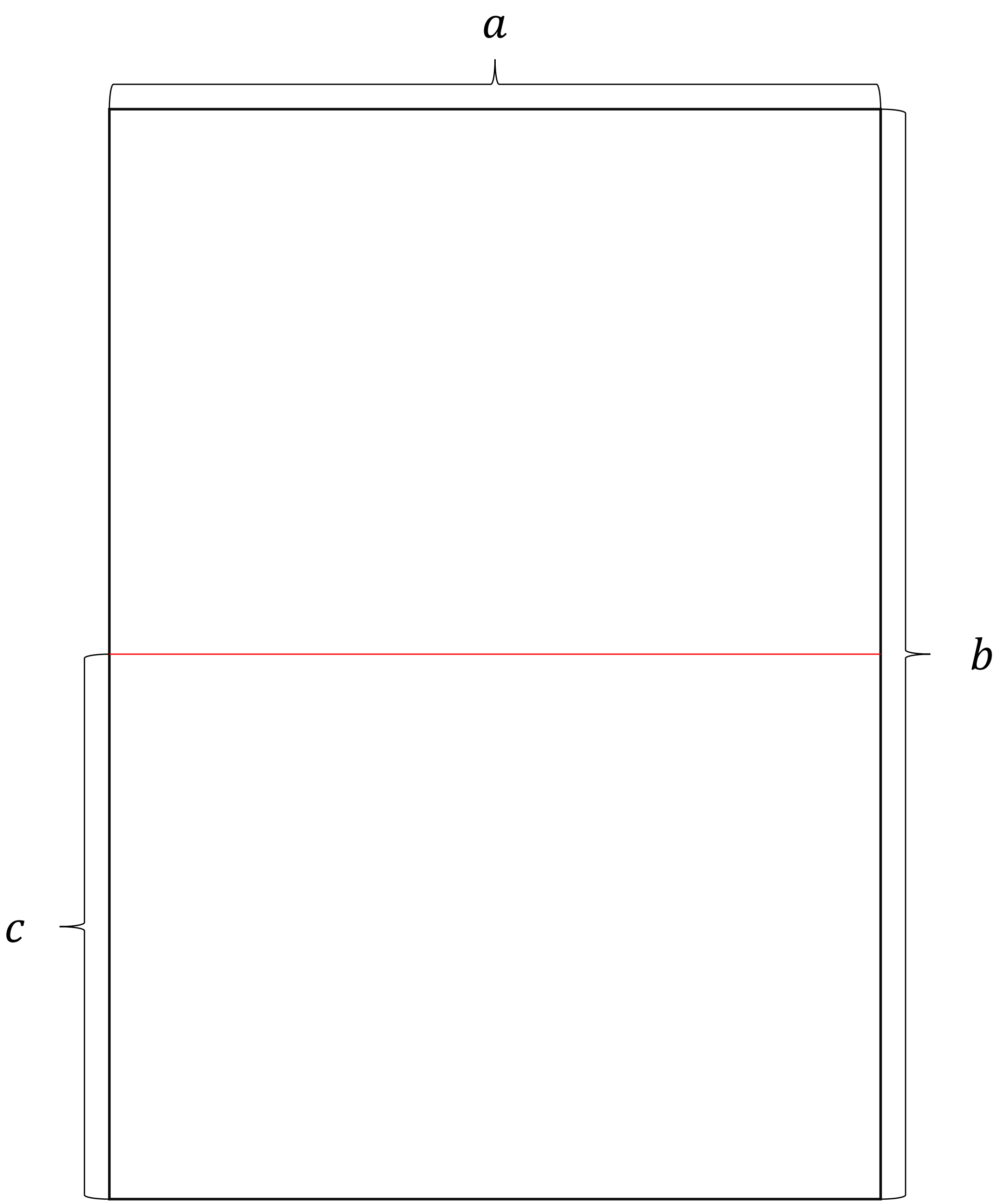
$\sqrt{2}$

Noah Schlenker Leon Baptist Kniffki Christian Krinitsin



Wofür ist $\sqrt{2}$ nötig?

- Gesucht: Seitenverhältnis x
- $b = xa, a = xc, c = \frac{1}{2}b$
- $b = xxc = x^2c$
- $b = \frac{x^2}{2}b$
- $1 = \frac{x^2}{2}$
- $2 = x^2$
- $x = \sqrt{2}$
- Durch das Seitenverhältnis $1{:}\sqrt{2}$ bleibt das Format beim Halbieren erhalten

Wie bestimmt man $\sqrt{2}$?

$$\begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}^n = \begin{pmatrix} x_{n-1} & x_n \\ x_n & x_{n+1} \end{pmatrix} \Rightarrow \lim_{n\to\infty} 1 + \frac{x_n}{x_{n+1}} = \sqrt{2}$$

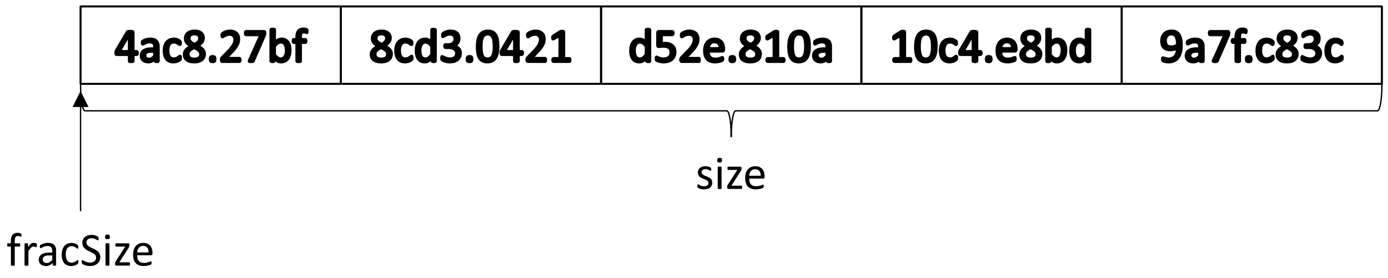
Wie bestimmt man $\sqrt{2}$?

$$\begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}^n = \begin{pmatrix} x_{n-1} & x_n \\ x_n & x_{n+1} \end{pmatrix} \Rightarrow \lim_{n\to\infty} 1 + \frac{x_n}{x_{n+1}} = \sqrt{2}$$

- Probleme:
  - Beliebig große Zahlen
  - Exponentiationen dauern lange
  - Division beliebig genauer Zahlen

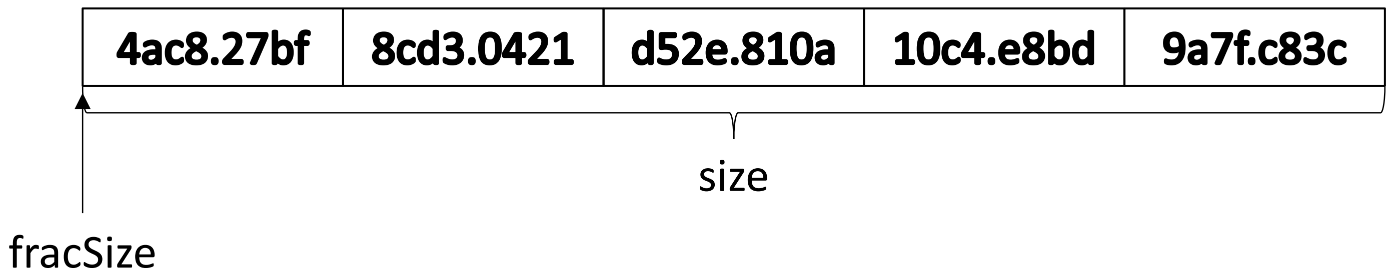
Zahlen mit beliebiger Genauigkeit

- Die Fixkommazahl



Zahlen mit beliebiger Genauigkeit

- Die Fixkommazahl

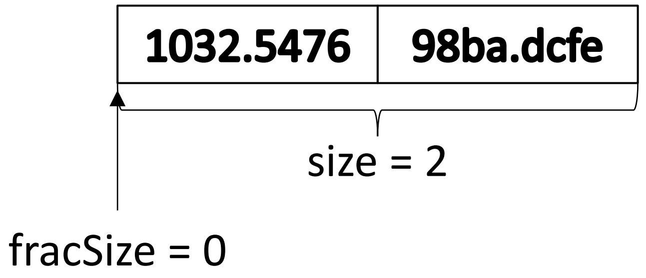


- Implementierung in C:
  - Little-Endian wie in x86-64

```c
struct bignum {
    uint32_t *digits;
    size_t size;
    size_t fracSize;
};
```

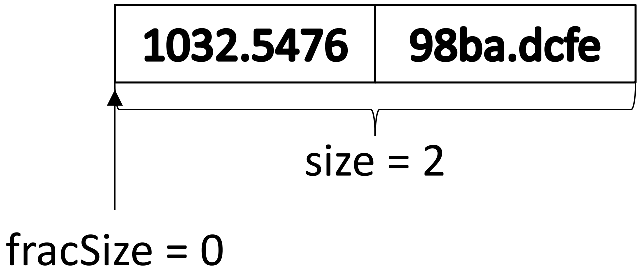
Zahlen mit beliebiger Genauigkeit

- 0xfedc.ba98.7654.3210

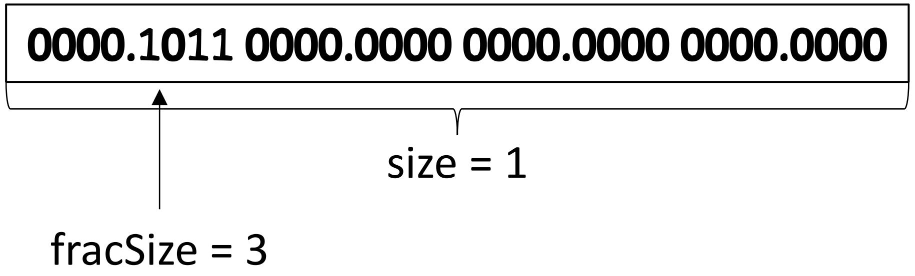


Zahlen mit beliebiger Genauigkeit

- 0xfedc.ba98.7654.3210



- 1,375 = 1,011



Grundlegende Arithmetik

Einfach

Addition $$a + b = \sum_{i=0}^{size-1} 2^{32i}(a_i + b_i)$$

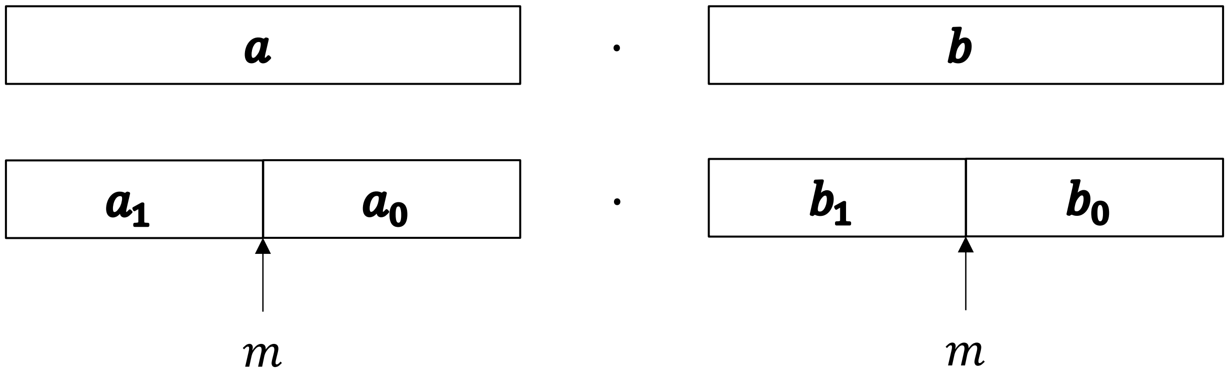
Multiplikation $$a \cdot b = \sum_{i=0}^{size-1} \left(2^{32i} b_i \sum_{j=0}^{size-1} 2^{32j} a_i \right)$$

Grundlegende Arithmetik

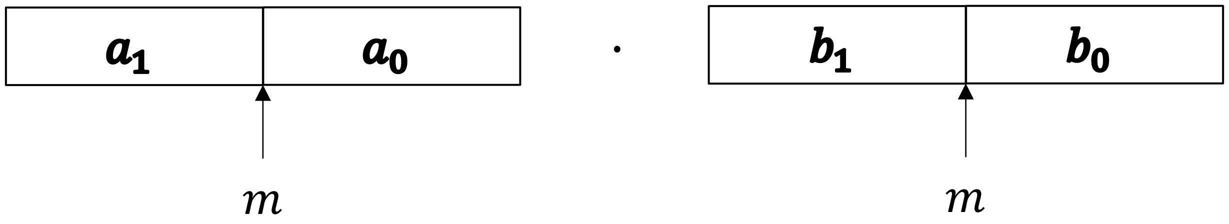
	Einfach	Mit SIMD
Addition	$a + b = \sum_{i=0}^{size-1} 2^{32i}(a_i + b_i)$	$= \sum_{i=0}^{\left\lfloor \frac{size}{4} \right\rfloor - 1} 2^{128i}(a_i + b_i) + \sum_{i=size-size\%4}^{size-1} 2^{32i}(a_i + b_i)$
Multiplikation	$a \cdot b = \sum_{i=0}^{size-1} \left(2^{32i} b_i \sum_{j=0}^{size-1} 2^{32j} a_i \right)$	$= \sum_{i=0}^{\left\lfloor \frac{size}{2} \right\rfloor - 1} \left(2^{64i} b_i \sum_{j=0}^{\left\lfloor \frac{size}{2} \right\rfloor - 1} 2^{64j} a_i + \sum_{j=size-size\%2}^{size-1} \cdots \right) + \cdots$

Karazuba – Eine bessere Multiplikation

- Russische Bauernmultiplikation liegt in $O(n^2)$
- Eine bessere Methode:
  - Divide and conquer
  - Wähle m, sodass $a = a_0 + 2^m a_1$ und $b = b_0 + 2^m b_1$



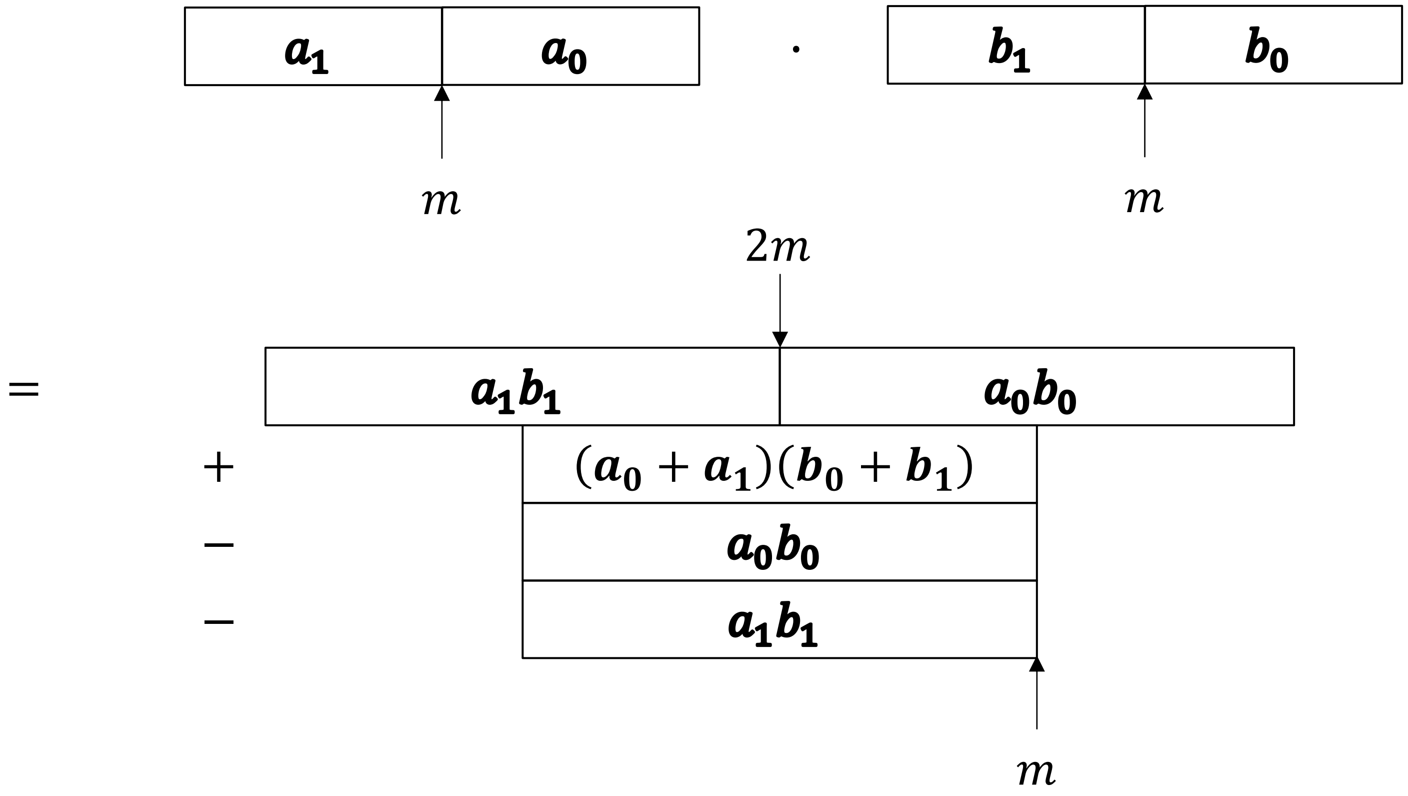
Karazuba – Eine bessere Multiplikation



$$(a_0 + 2^m a_1)(b_0 + 2^m b_1) = a_0 b_0 + 2^m (a_0 b_1 + a_1 b_0) + 2^{2m} a_1 b_1$$

$$= a_0 \cdot b_0 + 2^m \big((a_0 + a_1) \cdot (b_0 + b_1) - a_0 b_0 - a_1 b_1\big) + 2^{2m} a_1 \cdot b_1$$

Karazuba – Eine bessere Multiplikation



Die Matrixmultiplikation

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11} \cdot b_{11} + a_{12} \cdot b_{21} & a_{11} \cdot b_{12} + a_{12} \cdot b_{22} \\ a_{21} \cdot b_{11} + a_{22} \cdot b_{21} & a_{21} \cdot b_{12} + a_{22} \cdot b_{22} \end{pmatrix}$$

Acht teure Multiplikationen

Die Matrixmultiplikation

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \begin{pmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{pmatrix} = \begin{pmatrix} a_{11} \cdot b_{11} + a_{12} \cdot b_{21} & a_{11} \cdot b_{12} + a_{12} \cdot b_{22} \\ a_{21} \cdot b_{11} + a_{22} \cdot b_{21} & a_{21} \cdot b_{12} + a_{22} \cdot b_{22} \end{pmatrix}$$

Acht teure Multiplikationen

$$\begin{pmatrix} a_1 & a_2 \\ a_2 & a_3 \end{pmatrix} \begin{pmatrix} a_1 & a_2 \\ a_2 & a_3 \end{pmatrix} = \begin{pmatrix} a_1 \cdot a_1 + a_2 \cdot a_2 & a_1 \cdot a_2 + a_2 \cdot a_3 \\ \boldsymbol{a_2} \cdot \boldsymbol{a_1} + \boldsymbol{a_3} \cdot \boldsymbol{a_2} & \boldsymbol{a_2} \cdot \boldsymbol{a_2} + a_3 \cdot a_3 \end{pmatrix}$$

Nur noch fünf Multiplikationen

Mit $\begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}$

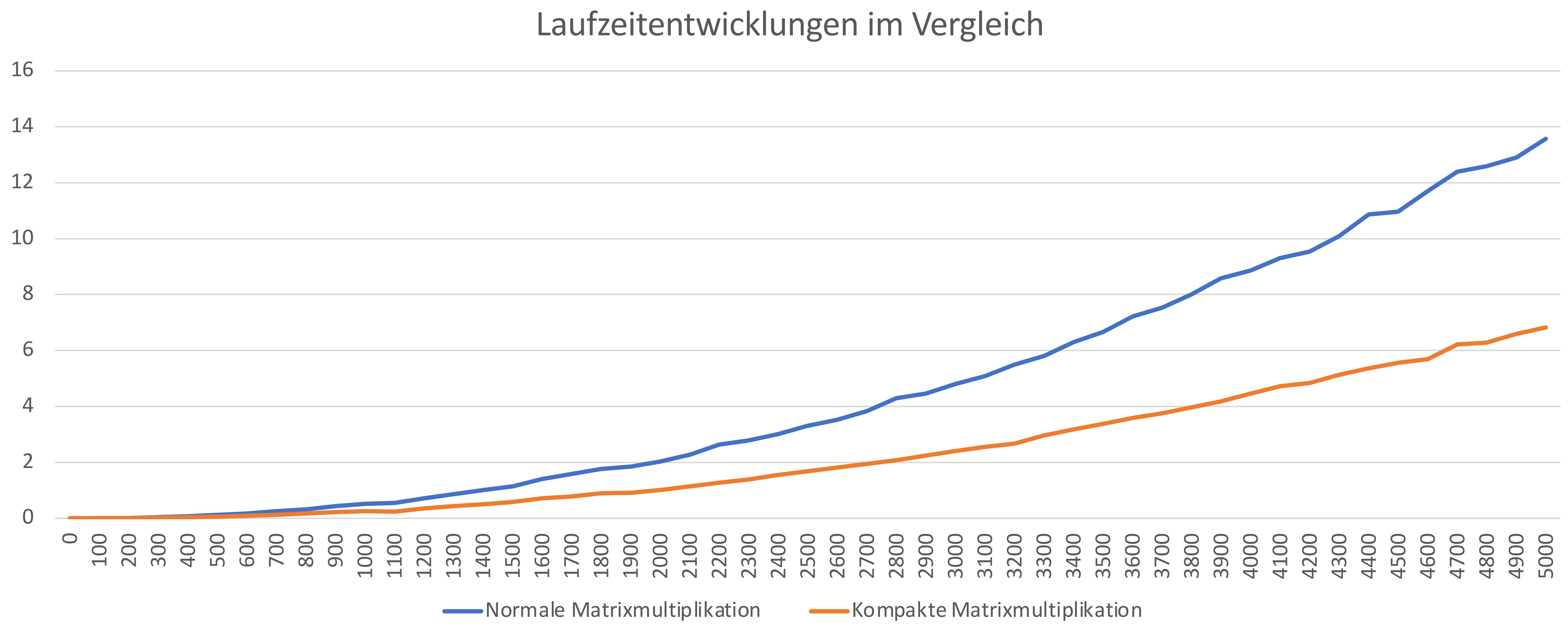
$$\begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}\begin{pmatrix} a_1 & a_2 \\ a_2 & a_3 \end{pmatrix} = \begin{pmatrix} a_2 & a_3 \\ a_1 + 2a_2 & a_2 + 2a_3 \end{pmatrix}$$

Durch Bitshifts und Addition realisierbar

$$x_0 = 0,\ x_1 = 1$$
$$x_n = 2x_{n-1} + x_{n-2}$$

Vier Multiplikationen

Die Matrixmultiplikation



Die schnelle Exponentiation

- Iteratives Aufmultiplizieren benötigt $O(n)$ Matrixmultiplikationen
- Schneller: Wiederholtes quadrieren der Basis

$$\begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}^6 = \begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}$$

$$= \left(\begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}\right)\left(\begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}\begin{pmatrix} 0 & 1 \\ 1 & 2 \end{pmatrix}\right)$$

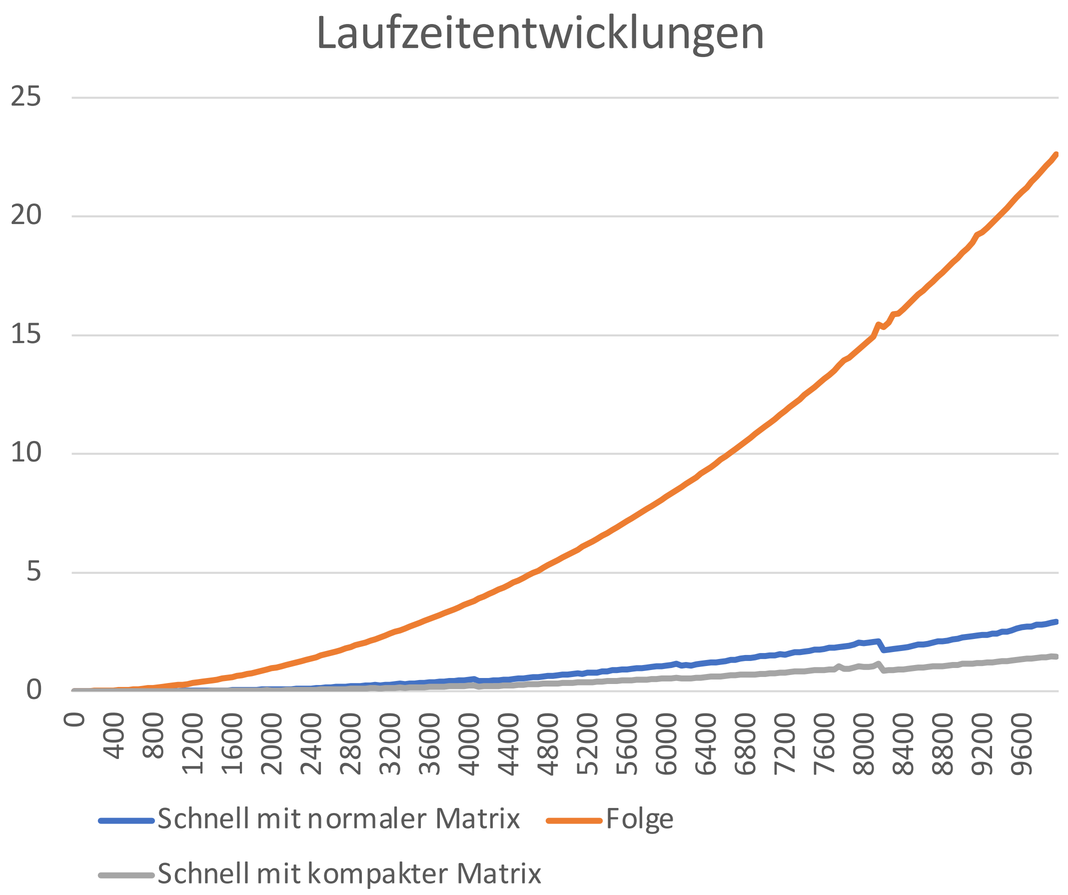
Die schnelle Exponentiation

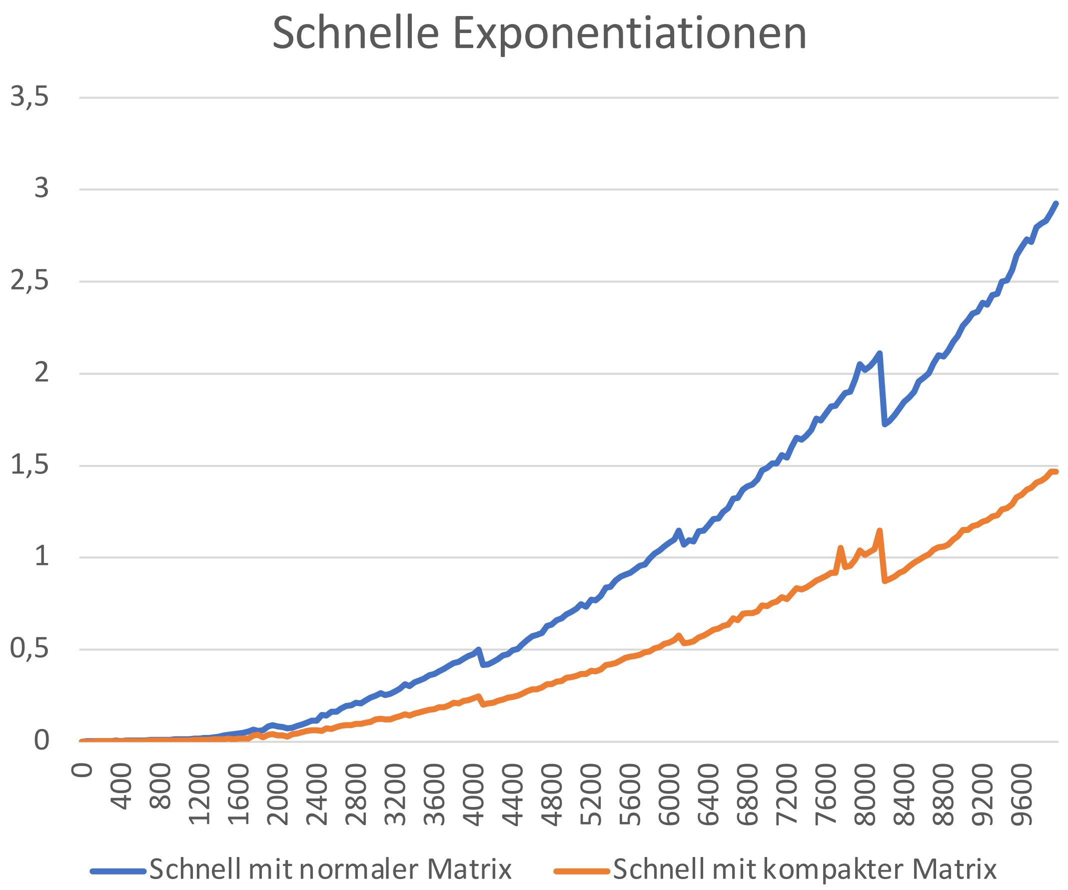
$$\begin{pmatrix}0 & 1\\1 & 2\end{pmatrix}^6 = \left(\begin{pmatrix}0 & 1\\1 & 2\end{pmatrix}\begin{pmatrix}0 & 1\\1 & 2\end{pmatrix}\right)\left(\begin{pmatrix}0 & 1\\1 & 2\end{pmatrix}\begin{pmatrix}0 & 1\\1 & 2\end{pmatrix}\begin{pmatrix}0 & 1\\1 & 2\end{pmatrix}\begin{pmatrix}0 & 1\\1 & 2\end{pmatrix}\right)$$

$$= \begin{pmatrix}0 & 1\\1 & 2\end{pmatrix}^2 \begin{pmatrix}0 & 1\\1 & 2\end{pmatrix}^4$$

$$= \begin{pmatrix}0 & 1\\1 & 2\end{pmatrix}^2 \left(\begin{pmatrix}0 & 1\\1 & 2\end{pmatrix}^2\right)^2$$

Die schnelle Exponentiation





Division von Bignums

- Teure Operation
- Muss nur auf gewünschte Nachkommastellen genau sein
- Einfaches Verfahren:
  - Dividend > Divisor → Aktuelles bit = 1, $Dividend = 2(Dividend - Divisor)$
  - Dividend = Divisor → Aktuelles bit = 1, Ergebnis exakt bestimmt
  - Dividend < Divisor → Aktuelles bit = 0, $Dividend = 2Dividend$

Division von Bignums

$$\frac{5_{10}}{12_{10}} = \frac{101_2}{1100_2} =$$

1010	$< 1100 \rightarrow$	0,0
10100	$> 1100 \rightarrow$	0,01
- 1100		
10000	$> 1100 \rightarrow$	0,011
- 1100		
1000	$< 1100 \rightarrow$	0,0110
10000	$> 1100 \rightarrow$	**0,01101**

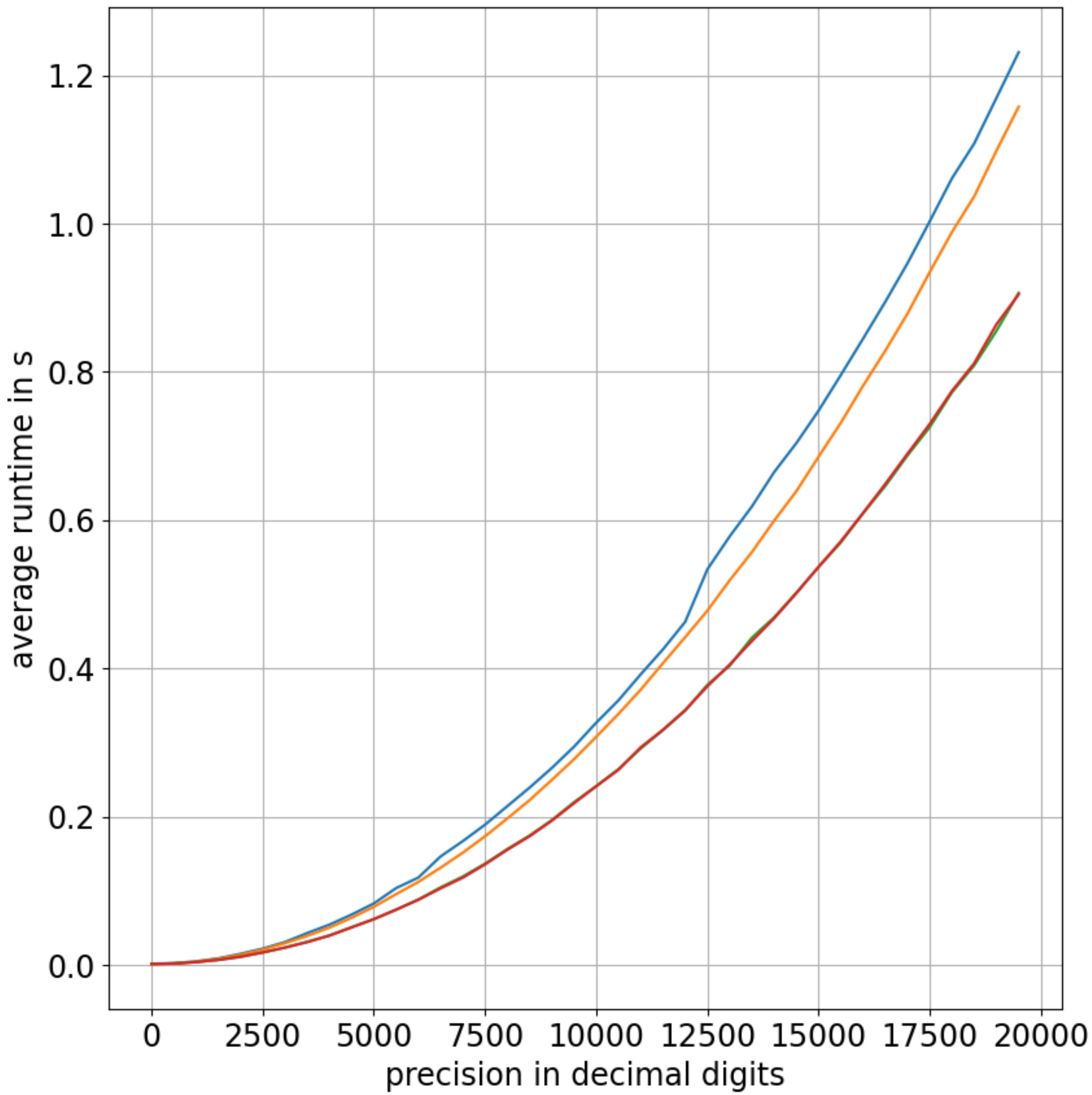
Was hat es gebracht?

Einfache Arithmetik, Normale Matrizen

Einfache Arithmetik, Kompakte Matrizen

SIMD Addition, Karazuba-Multiplikation, Kompakte Matrizen

SIMD Addition, SIMD Multiplikation, Kompakte Matrizen



Berechnung von $\sqrt{2}$