

## 5. Determinanten

### Rekursion:

Dem rekursiven Ansatz liegt der Laplace'sche Entwicklungssatz zu Grunde. Hierbei wird eine  $n \times n$  Matrix in  $n$  Untermatrizen aufgespalten, wobei diese der Größe  $(n-1) \times (n-1)$  entsprechen. Dies wird so oft wiederholt bis die Matrizen eine Größe von  $2 \times 2$  haben. Ab diesem Punkt werden die Determinanten über die Regel von Sarrus für die jeweiligen Untermatrizen errechnet und über die erhaltenen Determinanten werden die der entsprechend größeren Matrix errechnet. Bis Schlussendlich die Determinante der gesamten Matrix zurück gegeben wird.

Alles weitere Bezüglich der Umsetzung ist in den Kommentaren im Code enthalten.

### Iteration:

Dem iterativen Ansatz liegt der Gauß-Algorithmus zu Grunde. Hierbei wird die Matrix erst einmal durch Zeilenweise Operationen in die 1. Normalform gebracht. Für jede Multiplikation einer Zeile mit  $x$  (ohne das diese direkt auf eine andere addiert oder von solcher subtrahiert wird) wird  $x$  auf das bisherige  $x$  multipliziert und dann als „curMultiplier“ gespeichert. Dieser Wert entspricht dem Wert der Determinanten sobald die Matrix in der ersten Normalform ist und entlang der Hauptdiagonalen nur Einsen stehen.

Auch hier ist alles weitere Bezüglich der Umsetzung in den Kommentaren im Code enthalten.

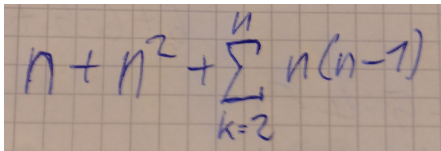
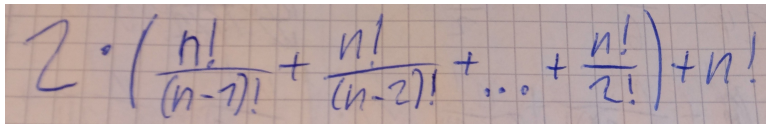
Anzahl der Multiplikation:

Bezüglich der Benötigten Multiplikation bei einer  $n \times n$  Matrix ergaben sich bei uns die Folgenden Werte:

n	Iterativ	Rekursiv
4	40	56
5	70	290
6	112	1752

Daran lässt sich schon mal erkennen das der Iterative Ansatz deutlich besser für Große Matrizen skaliert als der Rekursive.

Aus den Daten der Tests und unter Betrachtung des Codes ergaben sich so Folgende Formeln zur Berechnung der Benötigten Anzahl an Multiplikation für eine  $n \times n$  Matrix.

Iterativ	Rekursiv
	

Tests:

Die Logik wurde für mehrere Matrizen getestet und ergab meist ein Akkurates Ergebnis, die Abweichungen welche auftreten entstehen lediglich durch Rundungen der Doubles und treten nur bei der Iterativen Variante auf solange die Eingebene Matrix nur Ganzzahlen enthält.

Wir haben festgestellt das der Rundungsfehler bei uns immer dann besonders Hoch war wenn die Ausgangs Werte der Matrix weit gestreut waren, also falls man einige sehr große und einige sehr kleine Zahlen verwendet hat.