

Inhaltsverzeichnis

1	Abschnitte	1
2	Formeln	1
3	Graphen	1
4	Programmcodes	2

1 Abschnitte

Das ist eine *section*. Sie wird im Inhaltsverzeichnis aufgeführt. In einer pdf-Datei ist sie so verlinkt, dass man mit einem Klick auf die Zeile direkt zum entsprechenden Abschnitt gelangt.

Auch *subsections* werden im Inhaltsverzeichnis aufgelistet und entsprechend verlinkt.

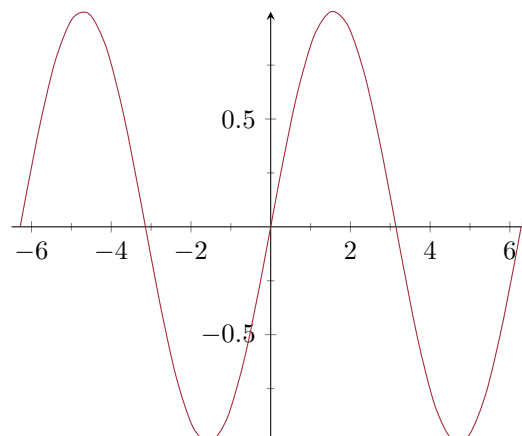
2 Formeln

Mit \LaTeX können schöne Formeln einfach dargestellt werden:

$$p_{ph} = \frac{h}{\lambda} = \frac{h * f}{c} = m * c \quad (1)$$

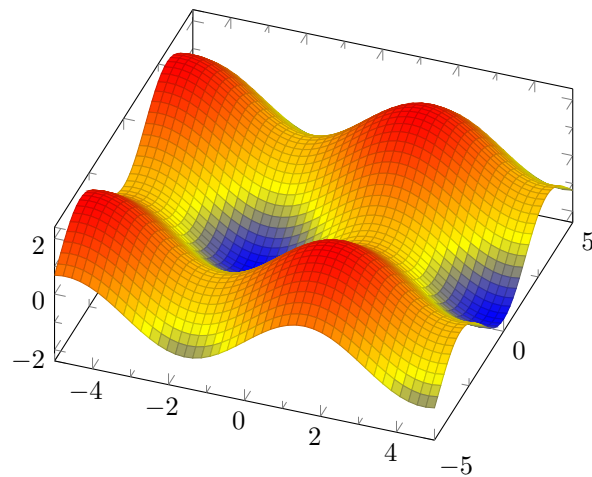
Mit dieser Formel kann man zum Beispiel den Impuls eines Photons berechnen!

3 Graphen



Auch Graphen lassen sich mit \LaTeX schön abbilden. Dafür wird das Paket *pgfplots* genutzt, wie oben mit der Sinus-Funktion gezeigt. Wird die Funktion mit der Farbe *facultyColor* gezeichnet, erscheint sie automatisch mit der Farbe der ausgewählten Fakultät.

pgfplots erlaubt es sogar, dreidimensionale Graphen zu zeichnen, hier am Beispiel $\sin(x) - \cos(y)$ veranschaulicht:



4 Programmcodes

Mit dieser \LaTeX -DocumentClass können ebenfalls Programmcodes mit Code Highlighting dargestellt werden:

```

1 public class Main {
2     public static void main(String args[]) {
3         int[] array = new int[10];
4         for(int i = 0; i < array.length; i++)
5             array[i] = 1 + (int)(Math.random() * 10);
6
7         int[] sorted = sort(array);
8
9         String initialOutput = "Starting Array: ";
10        String sortedOutput = "Sorted Array: ";
11
12        for(int i = 0; i < sorted.length; i++) {
13            initialOutput += array[i] + " ";
14            sortedOutput += sorted[i] + " ";
15        }
16
17        System.out.println(initialOutput + "\n" + sortedOutput);
18    }
19
20    public static int[] sort(int[] array) {
21        int[] sorted = new int[array.length];
22
23        for(int i = 0; i < sorted.length; i++)
24            sorted[i] = array[i];
25
26        for(int i = 0; i < sorted.length; i++) {
27            for(int j = i; j < sorted.length; j++) {
28                if(sorted[i] > sorted[j]) {
29                    int cache = sorted[j];
30                    sorted[j] = sorted[i];
31                    sorted[i] = cache;
32                }
33            }
34        }
35
36        return sorted;
37    }
38 }

```

Hinweis: Falls der Programmcode besonders kurz ist, kann das Listing auch mit dem Attribut `xleftmargin` ausgestattet werden, um die links angezeigten Zahlen korrekt am linken Seitenrand zu positionieren. Ein Wert von 10pt richtet die Zeilen für bis zu 9 Zeilen lange Programmcodes aus:

```
\begin{lstlisting}[xleftmargin=10pt]
% ...
\end{lstlisting}
```

Analog dazu lassen sich auch die Zeilennummern von besonders langen Programmcodes positionieren.

Der spezifische Wert für `xleftmargin` kann durch

$(\text{Anzahl Stellen der größten Zeilennummer} + 1) * 5\text{pt}$

berechnet werden.