



---

## Serie 1, Teil 1

---

### Projektplan

Im Rahmen dieser Serie wird der erste Teil eines übergreifenden Projektes bearbeitet. Abgesehen von Serie 4 werden alle Serien ein Teil dieses Projektes sein. Nachfolgend finden Sie eine grobe Vorschau der Inhalte der projektbezogenen Serien.

Serie 1: Approximation von Ableitungen mithilfe von finiten Differenzen

Serie 2: Darstellung von Differentialgleichungen als lineare Gleichungssysteme

Serie 3: Lösen linearer Gleichungssysteme mit direkten Verfahren

Serie 4: —

Serie 5: Lösen linearer Gleichungssysteme mit iterativen Verfahren

### Zielstellung dieser Serie

- Erlangen eines Grundverständnis von finiten Differenzen zur Approximation von Ableitungen von Funktionen
- Wiederholung wesentlicher Elemente von Python (Methoden, Klassen, Module) und ggf. Erarbeitung neuer Konzepte
- Analyse des Konvergenzverhalten eines Approximationsverfahrens und die Auswirkung von Rundungsfehlern
- Wiederholung wesentlicher Elemente von  $\text{\LaTeX}$  zur Erstellung eines wissenschaftlichen Berichtes

### Theoretischer Hintergrund — Finite Differenzen

Gegeben seien ein Intervall  $[a, b] \subset \mathbb{R}$ , eine reelle Funktion  $f \in C^\infty([a, b])$  und  $0 < h \in \mathbb{R}$ . Für  $x \in (a, b)$ , sodass  $x_+ := x + h \in [a, b]$ , lässt sich der Funktionswert von  $f$  an der Stelle  $x_+$  mit Hilfe der Taylorentwicklung

$$f(x_+) = \sum_{n=0}^{\infty} \frac{f^{(n)}(x)}{n!} h^n = f(x) + f'(x)h + \frac{f''(x)}{2}h^2 + \dots \quad (1)$$

approximieren. Dann gilt

$$(D_h^{(1)}f)(x) := \frac{f(x_+) - f(x)}{h} = f'(x) + \sum_{n=2}^{\infty} \frac{f^{(n)}(x)}{n!} h^{n-1}.$$

Gilt zusätzlich  $x_- := x - h \in [a, b]$ , so liefert die Taylorentwicklung von  $f$  in  $x_-$

$$f(x_-) = \sum_{n=0}^{\infty} \frac{f^{(n)}(x)}{n!} (-h)^n = f(x) - f'(x)h + \frac{f''(x)}{2}h^2 - \dots \quad (2)$$

Durch Summation von (1) und (2) erhält man dann

$$(D_h^{(2)}f)(x) := \frac{f(x_+) - 2f(x) + f(x_-)}{h^2} = f''(x) + \frac{f^{(4)}(x)h^2}{3 \cdot 4} + \dots$$

Die Funktionen  $D_h^{(1)}f$  und  $D_h^{(2)}f$  werden als erste und zweite finite Differenz von  $f$  bezeichnet. Die Restgliedabschätzungen der verwendeten Taylorentwicklungen liefern

$$(D_h^{(1)}f)(x) = f'(x) + \mathcal{O}(h) \quad \text{und} \quad (D_h^{(2)}f)(x) = f''(x) + \mathcal{O}(h^2). \quad (3)$$

Die Formeln in (3) sind die Grundlage für das numerische Approximieren von Ableitungen einer Funktion. Für  $p \in \mathbb{N}$  und  $i = 0, \dots, p$  seien  $x_i := a + i|b - a|/p$ . Eine Näherung des Approximationsfehlers in der Maximumsnorm erhält man durch

$$e_f^{(k)}(h) := \max_{i=0, \dots, p} |f^{(k)}(x_i) - (D_h^{(k)}f)(x_i)|.$$

## Aufgaben Teil 1 — Finite Differenzen mit Python

---

### Aufgabe 1.1: Implementation

---

Implementieren Sie in einem Modul `derivative_approximation.py`:

- eine Klasse `FiniteDifference(h, f, d_f=None, dd_f=None)`, welche Methoden zur Berechnung der Werte von  $(D_h^{(1)}f)(x)$  und  $(D_h^{(2)}f)(x)$  bereitstellt.
- eine weitere Methode `FiniteDifference.compute_errors(a, b, p)`, welche das Tupel der Fehler  $(e_f^{(1)}(h), e_f^{(2)}(h))$  berechnet.
- eine Methode, oder Funktion, welche die Abbildungen  $f, D_h^{(1)}f, D_h^{(2)}f$  und gegebenenfalls  $f', f''$  auf dem Intervall  $[a, b]$  zeichnet. Die Funktion soll dabei nur an den Punkten  $x_i$  ( $i = 0, \dots, p$ ) ausgewertet werden.
- eine Methode, oder Funktion, welche die Abbildungen  $e_f^{(1)}, e_f^{(2)}, h \mapsto h^j, j = 1, 2, 3$ , für eine gegebene Kollektion von Schrittweiten zeichnet.
- eine Funktion `main()`, welche alle genannten Funktionalitäten anhand der Funktion  $g_1(x) = \sin(x)/x$  auf dem Intervall  $[\pi, 3\pi]$  für  $p = 1000$  demonstriert.

Die genauen Schnittstellen für die vorgegebenen Programmelemente finden Sie in der Datei `derivative_approximation.py` vg. Moodle bzw. nächste Seite.

```

class FiniteDifference:
    """ Represents the first and second order finite difference approximation
    of a function and allows for a computation of error to the exact
    derivatives.

    Parameters
    -----
    h : float
        Stepzise of the approximation.
    f : callable
        Function to approximate the derivatives of.
    d_f : callable, optional
        The analytic first derivative of `f`.
    dd_f : callable, optional
        The analytic second derivative of `f`.

    Attributes
    -----
    h : float
        Stepzise of the approximation.
    """

    def __init__(self, h, f, d_f=None, dd_f=None):
        pass

    def compute_errors(self, a, b, p): # pylint: disable=invalid-name
        """ Calculates an approximation to the errors between an approximation
        and the exact derivative for first and second order derivatives in the
        maximum norm.

        Parameters
        -----
        a, b : float
            Start and end point of the interval.
        p : int
            Number of points used in the approximation of the maximum norm.

        Returns
        -----
        float
            Errors of the approximation of the first derivative.
        float
            Errors of the approximation of the second derivative.

        Raises
        -----
        ValueError
            If no analytic derivative was provided by the user.
        """

```