
Übung 1 *Konvergenzordnung*

In vielen Fällen kann die Konvergenzordnung eines Grenzprozesses

$$a(h) \to a(h \to 0), \quad a(h) - a = O(h^\alpha),$$

nur experimentell bestimmt werden. Dazu werden bei bekanntem Limes a für zwei Werte h und $h/2$ die Fehler $a(h) - a$ und $a(h/2) - a$ berechnet und dann die Ordnung α über den formalen Ansatz $a(h) - a = ch^\alpha$ aus der folgenden Formel ermittelt:

$$\alpha = \frac{1}{\log(2)} \log\left(\left| \frac{a(h) - a}{a(h/2) - a)} \right| \right)$$

1. Rekapitulieren Sie die Rechtfertigung dieser Formel.
2. Wie kann man vorgehen, wenn kein exakter Limes a bekannt ist?
3. Bestimmen Sie die Konvergenzordnungen für die folgenden von Funktionen $a(h)$ und $b(h)$ abgegriffenen Werte:

h	$a(h)$	$b(h)$
2^{-1}	7.188270827204928	8.89271737217539
2^{-2}	7.095485351135761	8.971800326329658
2^{-3}	7.047858597600531	8.992881146463981
2^{-4}	7.023726226390662	8.998220339291473
2^{-5}	7.011579000356371	8.999559782988968
2^{-6}	7.005485409034109	8.999895247704067
Limes	$a(0) = 7.0$	$b(0) = 9.0$

(5 Punkte)

Übung 2 *Kollokation konkret*

In dieser Übung sollen Sie ein Runge-Kutta-Verfahren durch Kollokation erzeugen.

1. Bestimmen Sie die Koeffizienten des impliziten Runge-Kutta-Verfahrens, dass durch Kollokation mit den Stützstellen der Simpsonregel

$$Q(f) = \frac{b-a}{6}\left(f(a) + 4f\left(\frac{a+b}{2}\right) + f(b) \right)$$

   erstellt wird.
2. Welche Konsistenzordnung hat dieses Verfahren?

(5 Punkte)

Übung 3 *Kollokation Eigenschaften*

Zeigen Sie, dass die Koeffizienten eines durch Kollokation definierten impliziten Runge-Kutta-Verfahrens folgende Beziehungen erfüllen:

$$\sum_{j=1}^{s} b_j c_j^{k-1} = \frac{1}{k}, \quad k = 1, \dots, s$$
$$\sum_{j=1}^{s} a_{ij} c_j^{k-1} = \frac{c_i^k}{k}, \quad i, k = 1, \dots, s$$

Anmerkung: Für $k = 1$ folgt damit insbesondere die Konsistenz und die Invarianz unter Autonomisierung.

(5 Punkte)

Übung 4 *Newton*

In dieser Aufgabe approximieren Sie die Lösung der 2-dimensionalen AWA

$$u_1'(t) = \sin(u_1(t))\sin(u_2(t)), \quad t \geq 0, \quad u_1(0) = 3,$$
$$u_2'(t) = \sin(u_1(t))\sin(u_2(t)), \quad t \geq 0, \quad u_2(0) = 4,$$

mit Hilfe der impliziten Trapezregel zweiter Ordnung. Die Lösung konvergiert für $t \to \infty$ gegen einen konstanten Vektor, dessen Wert bestimmt werden soll.

Gehen Sie dafür folgendermaßen vor:

1. Schreiben Sie eine Modellklasse für dieses Problem in der Sie Methoden für die Auswertung des Modells

```
//! model evaluation
void f (const T& t, const hdnum::Vector<N>& x, hdnum::Vector<N>& result) const
```

und der Jakobi Matrix

```
//! jacobian evaluation needed for implicit solvers
void f_x (const T& t, const hdnum::Vector<N>& x, hdnum::DenseMatrix<N>& result) const
```

bereitstellen.

2. Implementieren Sie die implizite Trapezregel

$$y_n^h = y_{n-1}^h + \frac{h_n}{2}(f(t_{n-1}, y_{n-1}^h) + f(t_n, y_n^h))$$

für allgemeine Systeme von Differentialgleichungen. Verwenden Sie das Newton Verfahren zur Lösung der dabei auftretenden nichtlinearen Gleichung

$$F(x) = 0.$$

Dabei dürfen Sie entweder die Klasse `Newton` aus der *hdnum* verwenden [†] oder Sie implemen-

[†]Beispiele: `hdnum/examples/num0/wurzel.cc` und `hdnum/src/ode.hh`. Setzen Sie `reduction=1e-8` und `abslimit=1e-14`

tieren das Newton Verfahren selbst nach folgender Vorlage:

```
Data: Startwert x^0
maxit = 25;
linesearchsteps = 10;
reduction = 1e − 8;
abslimit = 1e − 14;
for k = 0, ..., maxit − 1 do
    if ‖F(x^k)‖ ≤ abslimit then
        break;
    end
    Solve F_x(x^k) z^k = F(x^k);
    λ=1.0;
    for i = 1, ..., linesearchsteps do
        if ‖F(x^k − λz^k)‖ ≤ (1 − λ/4)‖F(x^k)‖ then
            x^{k+1} = x^k − λz^k;
            break;
        end
        else
            λ = λ/2;
        end
        if i = linesearchsteps then
            ERROR;
        end
    end
    if ‖F(x^{k+1})‖ / ‖F(x^0)‖ ≤ reduction then
        break;
    end
    else if k = maxit − 1 then
        ERROR;
    end
end
```

Algorithm 1: Newton method solving $F(x) = 0$.

3. Plotten Sie die numerische Lösung (für geeignetes $0 \leq t \leq T$).

(5 Punkte)