Numerische Methoden

GitHub: noah95/formulasheets

401-0654-00 ETH HS18 Juni 2019 Noah Hütter

I. Quadratur

Ziel: Integral approximieren $I[f]=\int_a^b f(x)dx \approx Q[f]=\sum_j \omega_j f(x_j)$

ω_j: Gewichte x_j: Stützstellen

Polynomiale Interpolation I.1

x_i: $n+1$ Stützstellen y_i: Stützwerte

Finde Polynom n-ten Grades $p_n(x)=a_0+a_1x+a_2x^2+\dots+a_nx^n$

welches IB (Interpolationsbedingung) $p_n(x_j)=y_j$ erfüllt $\rightsquigarrow$ LGS

Matlab $p=\text{polyfit}(x,y,n)$; x,y: Stützstellen, -werte; n: Grad

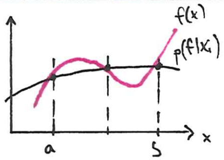
LI (Lagrange Interpol. Formel)

$\boxed{p_n(x)=\sum_{j=0}^{n} y_j \cdot L_j^n(x)}$ $\boxed{L_j^n(x)=\prod_{\substack{i=0\\ i\neq j}}^{n}\frac{x-x_i}{x_j-x_i}}$ ← Lagrange Polynom (LP)

L_j^n sind n-ten Grad

$L_j^n(x_i)=\delta_{ji}=\begin{cases}1, & \text{if } j=i\\ 0, & \text{if } j\neq i\end{cases}$

I.2 Interpolationsfehler



Fehlerfkt. $\boxed{e(x)=f(x)-p[f|x_i](x)=\frac{f^{(n+1)}(\xi)}{(n+1)!}\prod_{j=0}^{n}(x-x_j)}$

Max. Fehler $\boxed{\|e\|_\infty=\max_{x\in I}|e(x)|\leq\frac{\|f^{(n+1)}\|_\infty}{(n+1)!}(b-a)^{n+1}}$

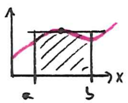
I.3 Numerische Integration (=Quadratur)

Idee: Polynomiale Interpolation von $f(x)$ und integriere $p[f|x_i](x)$

QR: Quadraturregel $\boxed{Q[f]=\sum_{j=0}^{n}\omega_j\cdot f(x_j)}$ $\boxed{\omega_j=\int_a^b L_j^n(x)dx}$

NC: Newton Cotes — QR-Familie mit äquidistanten Stützstellen

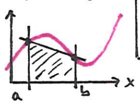
MR: Mittelpkt.-Regel



$\boxed{Q_0[f]=(b-a)f\left(\frac{a+b}{2}\right)}$

$n=0$ Grad $q=1$ Genauigkeitsgrad

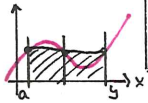
TR: Trapez-Regel



$\boxed{Q_1[f]=\frac{b-a}{2}\left(f(a)+f(b)\right)}$

$n=1$ $q=1$

SR: Simpson-Regel



$\boxed{Q_2[f]=\frac{b-a}{6}\left(f(a)+4f\left(\frac{a+b}{2}\right)+f(b)\right)}$

$n=2$ $q=3$

I.4 Quadraturfehler QF

QF: $E[f]=|Q[f]-I[f]|$ $\boxed{E[f]\leq\frac{\|f^{(q+1)}\|_\infty}{(q+1)!}(b-a)^{q+2}}$

GG: Genauigkeitsgrad $q\in\mathbb{N}$, QR hat GG q falls sie alle Polynome bis und mit Grad q exakt integriert. Bestimmung durch $Q[1]=I[1]$, $Q[x]=I[x]$, .., $Q[x^{q+1}]\neq I[x^{q+1}]$ auf $[-1,1]$

Ordnung: Die Ordnung s einer QR ist def. $\boxed{s=q+1}$

Transformation auf Referenzintervall

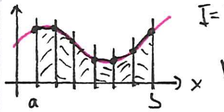
$\boxed{I[f]=\int_a^b f dx=\int_{-1}^{1}f\left(\frac{b-a}{2}t+\frac{a+b}{2}\right)\frac{b-a}{2}dt=\frac{b-a}{2}\int_{-1}^{1}f\left(\frac{b-a}{2}t+\frac{a+b}{2}\right)dt}$

1. $x\in[a,b]$, $t\in[-1,1]$ 2. $\left|\begin{matrix}a=m(-1)+c\\ b=m(1)+c\end{matrix}\right| \to m=\frac{b-a}{2}$, $c=\frac{a+b}{2}$, $x=mt+c$

3. $\frac{dx}{dt}=m \to dx=m\cdot dt$

I.5 Summierte Quadratur

Idee: Unterteile Intervall auf mehrere Teilintervalle und wende QR auf jedes TI an



$I=[a,b]$ $x_j=a+h\cdot j$ $h=\frac{b-a}{N}$ $j=0,..,N$

$I_j=[x_{j-1},x_j]$ h: Intervallbreite, N: Anz. Teilintervalle

SMR: Summierte MR $\boxed{Q_0^N[f]=\sum_{j=1}^{N}h\cdot f\left(\frac{x_{j-1}+x_j}{2}\right)}$

STR: Sum. TR $\boxed{Q_1^N[f]=\frac{h}{2}\left(f(a)+2\sum_{j=1}^{N-1}f(x_j)+f(b)\right)}$

SSR: Sum. SR $\boxed{Q_2^N[f]=\frac{h}{6}\left(f(a)+2\sum_{j=1}^{N-1}f(x_j)+4\sum_{j=1}^{N}f\left(\frac{x_{j-1}+x_j}{2}\right)+f(b)\right)}$

S2PG: Sum. 2Pkt. Gauss $\boxed{G_2^N[f]=\frac{h}{2}\sum_{i=1}^{N}\left(f\left(\frac{(\sqrt{3}+1)x_i+(\sqrt{3}-1)x_{i+1}}{2\sqrt{3}}\right)+f\left(\frac{(\sqrt{3}-1)x_i+(\sqrt{3}+1)x_{i+1}}{2\sqrt{3}}\right)\right)}$

Fehler: $\boxed{E^N[f]\leq\frac{\|f^{(q+1)}\|_\infty}{(q+1)!}(b-a)\cdot h^{q+1}}$ $\boxed{E^N[f]=\mathcal{O}(h^s)}$

Man schreibt $e=\mathcal{O}(h^p)$ falls $|e|\leq C\cdot h^p$ für pos. konst. C & p gilt für h klein genug. $E^N=\mathcal{O}(h^s)$. Ordnung s kann gut aus loglog-Plot des QF ablesen.

2D-Quadratur

$\int_a^b\int_a^b f(x,y)dxdy\approx\sum_{i,j=0}^{n}\hat{\omega}_{ij}\cdot f(x_i,y_j)$ Tabelle $\to$

(i,j)	$(0,0)$	$(0,1)$	–
$\hat{\omega}_{ij}$			
x_i,y_i			

$\int_\Omega f(x,y)d\Omega=\iint f(x(a,b),y(a,b))\,|\det(J)|\,da\,db$ J: Jacobi

$\approx\sum_{i,j}\omega_i\omega_j\,|\det(J)(a,b)|\,f(x(a,b),y(a,b))$

I.6 Gauss Quadratur

Wähle ω_j, x_j s.d. grösstmöglicher GG erreicht wird.

x_j: Nullstellen von $p_{n+1}(x)$ ← Legendre Polynomials

$\boxed{p_0(x)=1}$ $\boxed{p_1(x)=x}$ $\boxed{p_{j+1}(x)=\frac{2j+1}{j+1}x\cdot p_j(x)-\frac{j}{j+1}p_{j-1}(x)}$

GLQ: Gauss-Legendre Quadratur $G_n[f]=\sum_{j=0}^{n}\omega_j\cdot f(x_j)$

x_j: Nullstellen von $p_{n+1}(x)$

$q=2n+1$ (Grösstmögl.!) $\boxed{\omega_j=\frac{2(1-x_j^2)}{((n+1)p_n(x_j))^2}}$ $j=0,..,n$

$n=0$: $x_0=0$ $\omega_0=2$ $n=1$: $x_{0,1}=\mp\sqrt{\frac{1}{3}}$ $\omega_0=\omega_1=1$

$n=2$: $x_{0,2}=\mp\sqrt{\frac{3}{5}}$ $x_1=0$ $\omega_0=\omega_2=\frac{5}{9}$ $\omega_1=\frac{8}{9}$

Konvergenz

Algebraisch: $E(N)=\mathcal{O}(N^{-\alpha})\to\text{loglog}\to\alpha=\frac{\ln(E(N))/\ln(E(N/2))}{\ln(2)}$ Bsp: SSR

Exponentiell: $E(N)=\mathcal{O}(q^N)\to\text{semilog}\to\ln(q)=\frac{\ln(E(M))/\ln(E(N))}{M-N}$ Bsp: GLQ

I.7 Adaptive Quadratur

Interval-Halb. Fehlerschätzer: Vergl. QR mit derselben QR aber mit halbierter Teilintervall-Länge

$E^1[f]=|Q^1-I|\approx k.(b-a)^{s+1}$ $k=\frac{\|f^{(s)}\|_\infty}{s!}$ $E^1=|Q_1-Q_1^2+Q_1^2-I|\leq\frac{|Q_1-Q_1^2|+}{|Q_1^2+I|}$

$E^2[f]=|G^2-I|\approx k\cdot\left(\frac{b-a}{2}\right)^{s+1}+k\left(\frac{b-a}{2}\right)^{s+1}=\frac{k}{2^s}(b-a)^{s+1}=\frac{E^1}{2^s}$ $\approx\frac{E_1}{2^s}$

← nehme diesen i.d.R.

$\boxed{E^1[f]\approx\frac{2^s}{2^s-1}|Q^1[f]-Q^2[f]|}$ $\boxed{E^2[f]\approx\frac{1}{2^s-1}|Q^1[f]-Q^2[f]|}$

Q^1: zu vergl. QR Q^2: QR auf 2 Intervallen s: Ordn. der Referenz QR

MATLAB

```
function Q=adapt_quad(f,a,b,tol)
  if E<tol
    Q = quad(f,a,b);
  else
    Q1 = adapt_quad(f,a,(a+b)/2, tol/2);
    Q2 = adapt_quad(f,(a+b)/2, b, tol/2);
    Q = Q1+Q2;
end
```

```
%z.B SR auf 2 Teilintervallen
m = (a+b)/2;
ml = (3/4a + 1/4b);
Ql = (m-a)/6.(f(a)+4f(ml) + f(m));
mr = (1/4a + 3/4b);
Qr = (b-m)/6. (f(m)+4f(mr)+f(b));
E2 = 1/(2^4-1).abs (Q-(Ql+Qr));
E = E2;
```

II. Explizite Einschrittverfahren (ESV)

II.2 Grundbegriffe

AWP: Anfangswertproblem
$$\left.\begin{array}{ll}\dot y(t) = f(t, y(t)) & \text{DGL} \\ y(t_0) = y_0 & \text{AW}\end{array}\right\} \text{AWP}$$

Reduktion auf Sys. 1st Order

① $z_0 = y$, $z_1 = \dot y = \dot z_0$, $\vdots$, $z_{n-1} = y^{(n-1)} = \dot z_{n-2}$

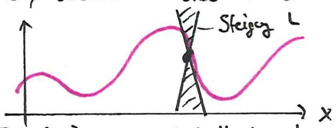
② $\dot{\vec z} = \vec g(t, \vec z(t))$ $\quad \vec z(t) = \begin{pmatrix} z_0 \\ z_1 \\ \vdots \\ z_{n-1}\end{pmatrix}$ $\quad \vec z(t_0) = \vec y_0$

③ $\vec g(t, \vec z) = \begin{pmatrix} z_1 \\ z_2 \\ \vdots \\ z_{n-1} \\ f(t, z_0, z_1, \dots)\end{pmatrix}$

Autonom: DGL ist autonom wenn $\boxed{f(t, y(t)) = f(y(t))}$

DGL autonomisieren:
$$\vec z(t) = \begin{pmatrix} \vec y(t) \\ t \end{pmatrix} = \begin{pmatrix} \vec z_1 \\ z_{n+1}\end{pmatrix}, \quad \vec g(\vec z) = \begin{pmatrix} f(t, \vec y) \\ 1 \end{pmatrix} = \begin{pmatrix} f(z_{n+1}, \vec z_1) \\ 1\end{pmatrix}$$
$$\dot{\vec z} = \vec g(\vec z) \qquad \vec z(t_0) = \begin{pmatrix} \vec y(t_0) \\ t_0 \end{pmatrix}$$

Lipschitz: $\vec f: I \times D \to \mathbb{R}^n$ ist LS (Lipschitz-stetig) in $\vec y$ mit Lipschitz-Konstante $L \geq 0$, wenn für alle $t \in I$ und $\vec y, \vec z \in D$ gilt
$$\|\vec f(t, \vec y) - \vec f(t, \vec z)\| \leq L \cdot \|\vec y - \vec z\|$$



Fkt f immer ausserhalb Kegel

i.) stetig diff.bare Fkt. sind LS mit $L = \max |f'(x)|$

ii.) auch nicht diff.bare können LS sein, z.B $|x| = f(x)$

Picard-Lindelöf: Sei $\vec f$ stetig in $(t, \vec y)$ und LS in $\vec y$ auf $[t_0, t_0 + \delta] \times D$, $\delta > 0$ und D eine Umgebung von AW $\vec y_0$, dann existiert eine eindeutige Lsg. $\vec y(t)$ für das AWP zumindest für eine kurze Zeit $[t_0, t_0 + \varepsilon]$

Satz II.2: $\vec f$ sei LS mit L dann gilt
$$\boxed{\|\vec y(t) - \vec z(t)\| \leq e^{L(t - t_0)} \|\vec y_0 - \vec z_0\|}$$

$\sqrt{x}$ ist stetig

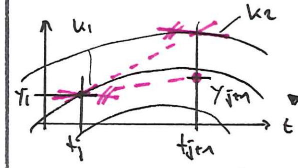
II.3 Runge-Kutta Verfahren RK

AWP integrieren: $y(t) = t_0 + \int_{t_0}^{t} f(\tau, y(\tau))\, d\tau \approx y_0 + h \cdot \sum w_i f(t_0 + hc_i, y(t_0 + hc_i))$

Integral mit QR approximieren und mit Euler schätzen

RK-ESV: RK Einschrittverfahren

$$\boxed{y_{j+1} = y_j + h \cdot \sum_{i=1}^{s} b_i k_i} \qquad \boxed{k_i = f\left(t_j + hc_i, y_j + h \sum_{\ell=1}^{s} a_{i\ell} k_\ell\right)}$$



BT: Butcher Table
$$\begin{array}{c|ccc} c_1 & a_{11} & a_{12} \cdots & a_{1s} \\ c_2 & a_{21} & a_{22} & \vdots \\ \vdots & \vdots & \ddots & \\ c_s & a_{s1} & \cdots & a_{ss} \\ \hline & b_1 & b_2 \cdots & b_s \end{array} \Big\} \text{mit } A$$

Explizit wenn $\vec A$ lower triangular

Euler, $p = 1$:
$$\begin{array}{c|c} 0 & 0 \\ \hline & 1 \end{array}$$

expl. MR / Vers. Euler, $p = 2$:
$$\begin{array}{c|cc} 0 & & \\ 1/2 & 1/2 & \\ \hline & 0 & 1 \end{array}$$

expl. TR / Heun, $p = 2$:
$$\begin{array}{c|cc} 0 & & \\ 1 & 1 & \\ \hline & 1/2 & 1/2 \end{array}$$

RK4, $p = 4$:
$$\begin{array}{c|cccc} 0 & & & & \\ 1/2 & 1/2 & & & \\ 1/2 & 0 & 1/2 & & \\ 1 & 0 & 0 & 1 & \\ \hline & 1/6 & 2/6 & 2/6 & 1/6 \end{array}$$

Picard-Lindelöf Verfahren: $\dot{\vec y} = A\vec y$, $\vec y(0) = \vec y_0$

$\vec y_{k+1}(t) = \vec y_0 + \int_0^t A \vec y_k(s)\, ds \qquad e^{At} = \sum_{k=0}^{\infty} \frac{(At)^k}{k!} \qquad \boxed{\vec y(t) = e^{At} \vec y_0}$

explizite Berechnung von $\vec y(t)$

II.4 Fehlerbetrachtung für ESV

Allgemeines ESV: $y_{j+1} = y_j + h \cdot \phi(t_j, y_j, h)$ $\qquad t_j = t_0 + j \cdot h$, $h = \frac{T - t_0}{N}$, $y(t_j) \approx y_j$

Verfahrensfkt $\phi = \sum b_i k_i$ für RK-ESV

GDF: Globaler Diskretisierungsfehler $\boxed{E_j = y(t_j) - y_j}$

KO: Konvergenzordnung: Ein ESV hat KO p, wenn gilt
$$\boxed{E = \max_j |y(t_j) - y_j| = \mathcal{O}(h^p)}$$

LDF: Lokaler Diskretisierungsfehler $\boxed{e_j = y(t_j) - \left(y(t_{j-1}) + h \cdot \phi(t_{j-1}, y(t_{j-1}), h)\right)}$

Fehler bei t_j nach einem Schritt ausgehend von der exakten Lösung.

KF: Konsistenzfehler $\boxed{\tau_j = \frac{e_j}{h} = \frac{y(t_j) - y(t_{j-1})}{h} - \phi(t_{j-1}, y(t_{j-1}), h)}$

Konsistenzordnung p wenn gilt $\boxed{\tau = \max_j |\tau_j| = \mathcal{O}(h^p)}$

Berechnung der Konsistenzordnung

Durch Taylor-Entwicklung von τ_{j+1} in h
$$\tau_{j+1} = \dot y(t_j) - \phi(t_j, y(t_j), 0) + \frac{h}{2}\left(\ddot y(t_j) - 2\phi'(t_j, y(t_j), 0)\right) + \frac{h^2}{6}\left(\dddot y(t_j) - 3\phi''(t_j, y(t_j), 0)\right) + \dots + \mathcal{O}(h^p)$$

$\dot y(t) = f(t, y(t))$

$\ddot y(t) = \partial_t f(t, y(t)) + (\partial_y f(t, y(t))) \cdot f(t, y(t))$

$\dddot y(t) = \partial_t^2 f + 2 \frac{\partial f}{\partial t \partial y} \cdot f + \frac{\partial f}{\partial y^2} \cdot f^2 + \frac{\partial f}{\partial t} \cdot \frac{\partial f}{\partial y} + \left(\frac{\partial f}{\partial y}\right)^2 \cdot f$

Für Entwicklung von ϕ: 2D-Taylor: - einzeln für $k_1, k_2, \dots$ rechnen - $t_n, y(t)$ festhalten
$$f(t + \Delta t, y + \Delta y) = f(t, y) + \frac{\partial f}{\partial t} \cdot \Delta t + \frac{\partial f}{\partial y} \cdot \Delta y + \frac{1}{2} \frac{\partial^2 f}{\partial t^2} \cdot \Delta t^2 + \frac{\partial^2 f}{\partial t \partial y} \Delta t \Delta y + \frac{1}{2} \frac{\partial^2 f}{\partial y^2} \Delta y^2 + \dots$$

Satz II.3: Falls $\vec f$ und $\vec\phi$ LS in $\vec y$ sind, dann gilt fürs ESV: $E = \max \|y(t_j) - y_j\| \leq \left(\|y(t_0) - y_0\| + \sum \|e_j\|\right) \cdot e^{\tilde L (t_N - t_0)}$

mit $\tilde L$: LK von ϕ. Daraus folgt

$$\boxed{\text{Konsistenzordnung} = \text{Konvergenzordnung}}$$

⚠ Berechne Konsistenzordnung und folgere mit Satz II.3 die Konvergenzordnung ⚠

Autonomisierungsinvarianz: Das Verfahren ist AI wenn es für autonomisierte AWP dasselbe Resultat liefert. ESV ist AI wenn $\boxed{c_i = \sum_{\ell=1}^{s} a_{i\ell}}$

Summiere Zeilen von $A = c$

→ Wenn AI wird p-Berechnung sehr einfach!

III. Adaptive Schrittweitensteuerung

III.1 Fehlerbetrachtung & Toleranz

$$\left.\begin{array}{ll} |e_j| \le atol & \text{abs.} \\ |e_j| \le rtol \cdot |y_{j-1}| & \text{rel.} \\ |e_j| \le atol + rtol \cdot |y_{j-1}| & \text{abs.+rel.} \end{array}\right\} \text{- toleranz Kriterium} \quad e_j: \text{LDF}$$

III.2 Lokale Fehlerschätzer

$$|e_j| = |y(t_j) - y(t_{j-1}) - h_j \phi(t_{j-1}, y_{j-1}, h_j)| = \mathcal{O}(h^{p+1}) = C \cdot h^{p+1}$$

kann nicht exakt bestimmt werden → Schätzen

Schrittweitenhalbierung Geg: Verfahren mit Ko p. Berechne mit h

$$y_{j+1} = y_j + h \cdot \phi(t_j, y_j, h)$$

und 2 Schritte mit $h/2$ (bzw. Kontrollverfahren)

$$\hat{y}_{j+1/2} = y_j + \tfrac{h}{2}\phi(t_j, y_j, h/2) \qquad \hat{y}_{j+1} = \hat{y}_{j+1/2} + \tfrac{h}{2}\phi(t_{j+1/2}, \hat{y}_{j+1/2}, h/2)$$

Der LDF ist dann $|e_{j+1}| = |y(t_{j+1}) - y_{j+1}| = C h^{p+1} + \mathcal{O}(h^{p+2})$

und $|\hat{e}_{j+1}| = |y(t_{j+1}) - \hat{y}_{j+1}| = 2C\left(\frac{h}{2}\right)^{p+1} + \mathcal{O}(h^{p+2}) = \frac{|e_{j+1}|}{2^p}$

$$\boxed{|e_{j+1}| \approx \frac{2^p}{2^p-1}|\hat{y}_{j+1} - y_{j+1}| = \varepsilon_{j+1}} \qquad \boxed{|\hat{e}_{j+1}| \approx \frac{1}{2^p-1}|\hat{y}_{j+1} - y_{j+1}| = \hat{\varepsilon}_{j+1}}$$

sind sog. a posteriori Fehlerschätzer ↑ nehme diesen

Eingebettete RK-Verfahren

Vergleich des Verfahrens mit Verfahren höherer Ordnung mit gleicher A-Matrix und $\vec{c}$-Vektor

$$\begin{array}{c|c} \vec{c} & A \\ \hline & \vec{b} \\ & \hat{\vec{b}} \end{array} \qquad \boxed{|e_{j+1}| \approx |\hat{y}_{j+1} - y_{j+1}| = \varepsilon_{j+1}}$$

← Kontrollverfahren

Bsp.: $\begin{array}{c|cc} 0 & & \\ 1 & 1 & \\ \hline & 1 & 0 \\ & 1/2 & 1/2 \end{array}$ ← Euler, ← Heun

III.3 Adapt. Schrittweitensteuerung

h – klein genug für Toleranz
– gross wie möglich für Performance

$$\boxed{H = h \cdot \min\left(fac_{max}, \max\left(fac_{min}, fac \cdot \left(\frac{tol}{\varepsilon}\right)^{\frac{1}{p+1}}\right)\right)} \quad tol = atol + rtol \cdot \max(|y_j|, |\hat{y}_j|)$$

ε

fac_{max} : begrenzte Vergrösserung $\approx 1.5-5$

fac_{min} : begrenzte Verkleinerung ≈ 0.5

fac : "Sicherheits"-Faktor ≈ 0.9

IV. Newton Verfahren

→ Nichtlineare Gleichung numerisch lösen

IV.1 skalare nichtlin. Gl.

Problem: Löse $f(x) = 0$ für $f: D = [a,b] \in \mathbb{R}$

$$\underset{\substack{\uparrow \\ \text{Startwert}}}{x^{(0)}} \longrightarrow \underset{\substack{\uparrow \\ \text{Algorithmus}}}{x^{(1)} \longrightarrow} \cdots \longrightarrow x^* = \lim_{k\to\infty} x^{(k)} \quad \text{mit } f(x^*) = 0$$

Wir nehmen an, dass f stetig ist.

Fixpkt. Iteration & Grundlagen

Problem $f(x) = 0$ kann umgeschrieben werden in FPP (Fixpunktproblem) $\boxed{x = \phi(x)}$ $\boxed{x^* = \phi(x^*)}$

ϕ: Fixpunktfunktion $\quad x^*$: Fixpunkt (FP)

Konsistenz mit Nullstellenproblem falls gilt

$$\boxed{f(x^*) = 0 \iff \phi(x^*) = x^*}$$

FPI: Fixpunktiteration $\boxed{x^{(k+1)} = \phi(x^{(k)})}$ $k = 0, 1, ..$

i.) FPI ist nicht eindeutig
ii.) FPI können divergieren
iii.) Falls konvergent, können sie verschieden schnell

Konvergenzordnung $\boxed{|x^{(k+1)} - x^*| \le C \cdot |x^{(k)} - x^*|^p}$

p: Konvergenzordnung (1: linear 2: quadratisch)
C: Konvergenzrate

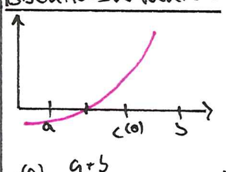
Fehler $\boxed{\varepsilon^{(k)} = |x^{(k)} - x^*|}$ $\varepsilon^{(k)}$: Fehler in k-ter Iteration

$$\boxed{p = \frac{\log(\varepsilon^{(k+1)}) - \log(\varepsilon^{(k)})}{\log(\varepsilon^{(k)}) - \log(\varepsilon^{(k-1)})}} \qquad \boxed{C = \frac{\varepsilon^{(k+1)}}{(\varepsilon^{(k)})^p} = \frac{\varepsilon^{(k)}}{(\varepsilon^{(k-1)})^p}}$$

ABK: Abbruchkriterium $\varepsilon^{(k)} \le atol \quad \varepsilon^{(k)} \le atol + rtol\,|x^{(k)}|$

$\varepsilon^{(k)} \le rtol \cdot |x^{(k)}| \quad \varepsilon \le ftol$

Bisektionsverfahren



$c^{(0)} = \frac{a+b}{2}$

Annahme: a, b mit $f(a) < 0 < f(b)$ bekannt

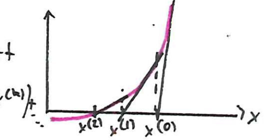
Idee: Halbiere $[a,b]$ und behalte Intervall welches Bed. erfüllt

i.) sehr einfach und robust
ii.) A priori Fehlerschätzer $\varepsilon^{(k)} = \frac{b-a}{2^{k+1}}$
iii.) langsame lin. Konvergenz
iv.) Nicht direkt anwendbar auf System

Newton-Verfahren

Idee: Linearisiere f in jedem Schritt

Taylor von f in $x^{(k)}$: $f(x) = \underbrace{f(x^{(k)}) + f'(x^{(k)})(x - x^{(k)})}_{\tilde{f}(x) = 0 \;\rightsquigarrow\; x^{(k+1)}} + \ldots$



$$\boxed{x^{(k+1)} = x^{(k)} - \frac{f(x^{(k)})}{f'(x^{(k)})}}$$

i.) benötigt f'
ii.) Quadratische Konvergenz
iii.) kann schief gehen
iv.) verallgem. bar für System

IV.2 nichtlin. Gl.-System

Problem: $f: D \subset \mathbb{R}^n \to \mathbb{R}^n$, löse $\vec{f}(\vec{x}) = 0$ $\boxed{\vec{x}^{(k+1)} = \vec{\phi}(\vec{x}^{(k)})}$

Newton-Verfahren

Taylor von $\vec{f}$ in $\vec{x}^{(k)}$: $\vec{f}(\vec{x}) = \vec{f}(\vec{x}^{(k)}) + D\vec{f}(\vec{x}^{(k)})(\vec{x} - \vec{x}^{(k)}) + \ldots$ ← Jacobian Matrix

$$D\vec{f}(\vec{x}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_n} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_m}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_n} \end{bmatrix}(\vec{x}) \quad \rightsquigarrow \boxed{\vec{x}^{(k+1)} = \vec{x}^{(k)} - D\vec{f}(\vec{x}^{(k)})^{-1}\vec{f}(\vec{x}^{(k)})}$$

i.) Inverse der Matrix nie berechnen!
ii.) Benötigt Jacobi-Matrix
iii.) Kann schief gehen

V. Stabilität & implizite Verfahren

V.1 Stabilitätsgebiet & A-Stabilität

SF: Stabilitätsfunktion ESV angewendet auf Dahlquist AWP ($\dot{y} = \lambda y$, $y(0) = y_0$) kann man in folgender Form schreiben:

$$\boxed{y_{j+1} = g(z) \cdot y_j} \quad z = h\lambda \qquad g(z): \text{Stabilitätsfunktion}$$

Exp. Euler $^{A✗}$ $g(z) = 1 + z$, $SI = (-2, 0)$ — Heun $^{A✗}$ $g(z) = 1 + z + \frac{1}{2}z^2$ $SI = (-2, 0)$

Impl. Euler $^{A✓}$ $g(z) = \frac{1}{1-z}$ $SI = (-\infty, 0) \cup (2, \infty)$ — Imp. MR $^{A✓}$ $g(z) = \frac{2+z}{2-z}$ $SI = (-\infty, 0)$

Verb. Euler $^{A✗}$ $g(z) = 1 + z + \frac{1}{2}z^2$ $SI = (-2, 0)$ — RK4 $^{A✗}$ $g(z) = 1 + z + \frac{1}{2}z^2 + \frac{1}{6}z^3 + \frac{1}{24}z^4$ $SI = (-2{,}78, 0)$

Impl. TR A✓

SG: Stabilitätsgebiet $\boxed{SG = \{z = h\lambda \in \mathbb{C} \mid |g(z)| < 1\}}$

SI: Stabilitätsintervall $\boxed{SI = \{x = h\lambda \in \mathbb{R} \mid |g(x)| < 1\}}$

A-Stabilität Ein Verfahren heisst A-Stabil wenn die gesamte linke komplexe Halbebene im SG enthalten ist ← expl. Verfahren sind nie A-Stabil

$$\boxed{\{z = h\lambda \in \mathbb{C} \mid Re(z) < 0\} \subset SG} \qquad \begin{pmatrix}\text{explizit} \\ \text{implizit}\end{pmatrix} \text{Euler ist} \begin{pmatrix}\text{nicht} \\ \cdot\end{pmatrix} \text{A-Stabil}$$

L-Stabil Ein Verfahren heisst L-Stabil wenn es A-Stabil ist und $\boxed{\lim_{z\to-\infty} g(z) = 0}$ erfüllt. Denn die exakte Lsg gilt $\lim_{t\to\infty} y(t) = 0$ soll auch für Verfahren gelten

impl Euler ✓ ist L-St. imp. MR ✗ ist nicht

V.2 Implizite RK-Verfahren

i.A. muss ein nichtlin. GS gelöst werden.

Implizit Euler $\begin{array}{c|c} 1 & 1 \\ \hline & 1 \end{array}$ Imp. MR ($p=2$) $\begin{array}{c|c} 1/2 & 1/2 \\ \hline & 1 \end{array}$ Impl. TR ($p=2$) $\begin{array}{c|cc} 0 & & \\ 1 & 1/2 & 1/2 \\ \hline & 1/2 & 1/2 \end{array}$

RK-Gauss ($p=4$) $\begin{array}{c|cc} 1/2-\sqrt{3}/6 & 1/4 & 1/4-\sqrt{3}/6 \\ 1/2+\sqrt{3}/6 & 1/4+\sqrt{3}/6 & 1/4 \\ \hline & 1/2 & 1/2 \end{array}$

SDIRK (Singly Diagonal Implicit RK, $p=3$) $\begin{array}{c|cc} \gamma & \gamma & \\ 1-\gamma & 1-2\gamma & \gamma \\ \hline & 1/2 & 1/2 \end{array}$ $\gamma = \frac{3 \pm \sqrt{3}}{6}$

V.3 Steife Probleme

Treten auf bei DGL Systemen mit stark unterschiedlichen Abklingzeiten. D.h. Prozesse laufen auf stark unterschiedlichen Zeitskalen ab.

$\dot{\vec{y}}(t) = A\vec{y}(t) + \vec{S}(t)$, $A \in \mathbb{R}^{n \times n}$ bezeichnet man als steif wenn für die Eigenwerte λ_i von A gilt $\boxed{Re(\lambda_i) < 0}$ und

$$\boxed{S = \frac{\max |Re(\lambda_i)|}{\min |Re(\lambda_i)|} \gg 1}$$

Eigenwerte: $\det(A - \lambda I) = 0 \rightsquigarrow \lambda = \dots$

Für nichtlin. DGL $\dot{\vec{y}} = \vec{f}(t, \vec{y}(t))$ ($\vec{f}$: non-lin.) definiert man ein lokales Mass der Steifheit durch Linearisieren an $t_1, \vec{y}_1$:

$$\vec{f}(t, \vec{y}(t)) = \vec{f}(t_1, \vec{y}_1) + \frac{\partial \vec{f}}{\partial t}(t_1, \vec{y}_1)(t - t_1) + \underbrace{J}_{\text{Jacobi-Matrix}}(t_1, \vec{y}_1) \cdot (\vec{y} - \vec{y}_1)$$

$$\rightsquigarrow \dot{\vec{y}}(t) = \underbrace{J(t_1, \vec{y}_1)\vec{y}(t)}_{A \cdot \vec{y}(t)} + \underbrace{\left(\vec{f}(t_1, \vec{y}_1) + \frac{\partial \vec{f}}{\partial t}(t_1, \vec{y}_1)(t - t_1) - J(t_1, \vec{y}_1)\vec{y}_1\right)}_{\vec{S}}$$

Ist diese Linearisierung steif, so nennt man das nicht lin. DGL-sys. lokal steif um $(t_1, \vec{y}_1)$

$\left.\begin{array}{l} \text{explizit} \\ \text{implizit} \end{array}\right\}$ ist $\left\{\begin{array}{l} \text{günstig} \\ \text{teuer} \end{array}\right\}$ pro Schritt, h durch $\left\{\begin{array}{l} \text{schnellste Komp.} \\ \text{Genauigkeit} \end{array}\right\}$ gegeben

$\Rightarrow$ implizite Methoden sind aufwändiger aber für steife Probleme effizienter

$\Rightarrow$ Use BDFk Verfahren

VI. Strukturerhaltende Verfahren

Struktur = essentielle Eigenschaft einer (zeitl./räuml.) Entwicklung

Bsp:
- Mehrkörpersimulation: Erhaltung von Energie, Impuls, Drehimpuls
- Maxwell: $\vec{\nabla} \cdot B = 0$
- Pendel: Totale Energie $E = \frac{m}{2}(l\dot{\varphi})^2 + mgl(1 - \cos\varphi) = \text{const.}$
  $\ddot{\varphi} + \frac{g}{l}\sin\varphi = 0$

Von den bisher erwähnten ESV erhält nur die impl. MR die Energie $\rightarrow$ Familie der geometrischen Integratoren

Verlet Verfahren AWP: $F = m \cdot a$ $\quad r(t=0) = r_0$

$$\boxed{\begin{array}{ll} v_{j+1/2} = v_j + \frac{F_j}{m} \cdot \frac{\Delta t}{2} & r_{j+1} = r_j + v_{j+1/2} \cdot \Delta t \\ v_{j+1} = v_{j+1/2} + \frac{F_{j+1}}{m} \cdot \frac{\Delta t}{2} & \end{array}}$$

V.3 Rückwärtsdifferenzmethoden

lin. Mehrschrittverfahren Form: $\boxed{\sum_{l=0}^{k} \alpha_l y_{j+1-l} = h \cdot \sum_{l=0}^{k} \beta_l \cdot f_{j+1-l}}$

Adams-Bashforth/-Moulton sind Teil dieser Familie.

AB2 ($p=2$): $y_{n+2} = y_{n+1} + h\left(\frac{3}{2}f(t_{n+1}, y_{n+1}) - \frac{1}{2}f(t_n, y_n)\right)$ ABp sind explizit

AB1: $y_{n+1} = y_n + h \cdot f(t_n, y_n)$ Euler Methode

AM2: AMp sind implizit

BDF, Backwards differenzing methods BDFk: k-Schritt BDF

Idee: Rechte Seite f auswerten an $f(t_{i+1}, y_{i+1})$ und gleichsetzen mit Approx. der Ableitung zur Zeit t_{i+1} welche man mittels Interpolation von $y_{i+1}, y_i, \dots, y_{i+1-k}$ bestimmt.

BDF1 IP $p_1(t) = y_{i+1}\frac{t - t_i}{h} - y_i\frac{t - t_{i+1}}{h}$

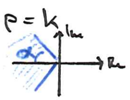
$\left.\frac{d}{dt}p_1(t)\right|_{t=t_{i+1}} = \frac{y_{i+1} - y_i}{h} \approx \dot{y}(t) = f(t_{i+1}, y_{i+1}) \Rightarrow y_{i+1} - y_i = h \cdot f(t_{i+1}, y_{i+1})$ impl. Euler

BDF2 $p_2(t) = y_{i-1}\frac{1}{2h^2}(t - t_i)(t - t_{i+1}) - y_i\frac{1}{h^2}(t - t_{i-1})(t - t_{i+1}) + y_{i+1}\frac{1}{2h^2}(t - t_{i-1})(t - t_i)$

$\left.\frac{d}{dt}p_2(t)\right|_{t=t_{i+1}} = y_{i-1}\frac{1}{2h} - y_i \cdot \frac{2}{h} + y_{i+1}\frac{3}{2h} \approx \dot{y}(t) \Rightarrow y_{i+1} - \frac{4}{3}y_i + \frac{1}{3}y_{i-1} = \frac{2}{3}hf(t_{i+1}, y_{i+1})$

$\alpha_0 = 1 \quad \alpha_1 = -\frac{4}{3} \quad \alpha_2 = \frac{1}{3} \quad \beta_0 = \frac{2}{3}$

- BDF werden für steife Probleme verwendet
- BDFk hat Ordnung $p = k$
- BDF1..2 sind A-Stabil
- BDF3..6 sind $A(\alpha)$-Stabil



Matlab

Built - in solvers

Stiff ODEs: ode15s (best), ode23s, ode23t, ode23tb

Nonstiff ODE: ode45, ode23, ode113 (med, low, low-high accuracy)

Full implicit: ode15i usage: ode45(@(t,y) 2*t+y, [t0 T], y0);

Basics

- a:h:b ← start : inkrement : ende
  a:b ← start : 1 : ende
- linspace(a,b,n) a = start b = end n = anz äquidist. Stellen
- p = polyfit(x,y,2) ⇒ p(1)·x^2 + p(2)·x + p(3)
- fsolve(f, x0) finde Nullstelle von f um x0
- integral(f,a,b) integriere f mit adaptiver Quadratur von a→b
- quad(f,a,b) simpson adaptive Quadratur
- eig(A) Eigenwerte finden
- A\B löse A*x = B → x = A\B

I. Quadratur

Polynomiale Interpolation

```
p = polyfit(x,y,n)     n: order
```

summierte Quadraturen

```
function y = sumtrapez(f,a,b,N)
  x = linspace(a,b,N+1);
  h = (b-a)/N;
  fx = f(x);
  y = 0.5*h*(fx(1)+2*sum(fx(2:end-1))+fx(end));
end

function y = sum2pgauss(f,a,b,N)
  x = linspace(a,b,N);
  h = (b-a)/N;  y=0;
  for i=1:N
    y = y+0.5*h*(f(((sqrt(3)+1)*x(i)...
      +(sqrt(3)-1)*x(i+1))/(2*sqrt(3)))...
      +f(((sqrt(3)-1)*x(i)+(sqrt(3)-1)...
      *x(i+1))/(2sqrt(3))));
  end
end

function sumbestordnung
  a=..; b=..; integrand = @(x)...;
  refsol = integral(integrand, a, b);
  N = 2.^(1:6);
  apsol = zeros(1,length(N));
  for i=1:length(N)
    apsol = quadratur(integrand,a,b,N(i));
  end
  err = norm(apsol-refsol);
  figure;
  loglog(N,err,'-*')
  p = polyfit(log(N), log(err), 1);
  ordnung = -p(1);
end
```

Konvergenz

```
exp.:  p=polyfit(1:nmax, log(E(1:nmax)),1);
       q=exp(p(1));

alg.:  p=polyfit(log(N), log(E), 1);
       q = -p(1);
```

adaptive Quadratur

```
function adapt_trapez_simple(f,a,b,tol)
  s=2;  % Ordnung
  diff = (b-a)/2;
  mean = (a+b)/2;
  Q = diff * (f(a)+f(b));
  Ql = 0.5*(mean-a)*(f(a)+f(mean));
  Qr = 0.5*(b-mean)*(f(mean)+f(b));
  E2 = 1/(2^s-1).*abs(Q-(Ql+Qr));
  if E2<tol
    Q = Ql+Qr;
  else
    Ql = adapt_trapez_simple(f,a,mean,tol/2);
    Qr = ———— (f,mean,b,tol/2);
    Q = Ql+Qr;
  end
end
```

2D Quadratur

```
function Q=trapez2D(f,a,b,Nx,c,d,Ny)
  hx = (b-a)/Nx;   hy = (d-c)/Ny;
  x = linspace(a,b,Nx+1);
  y = linspace(a,b,Ny+1);
  [x,y] = meshgrid(x,y);
  w = 2*ones(size(x));
  w(1,1)=1; w(1,end)=1;
  w(end,1)=1; w(end,end)=1;
  w(2:end-1,2:end-1) = 4;
  Q = 0.25*hx*hy*sum(sum(w.*f(x,y)));
end
```

II. ESV

```
function [t,y]=expEuler(f,t0,T,y0,N)
  h=(T-t0)/N;
  t = zeros(1,N+1);
  y = zeros(length(y0),N+1);
  t(1) = t0;
  y(:,1) = y0;
  for k=1:N
    t(k+1)=t0+k*h;
    y(:,k+1)=y(:,k)+h*f(t(k),y(:,k));
  end
end
```

Konvergenzordnung

```
function konvOrdnungEmpirisch
  f = @(t,y) ..;
  y0 = ..; t0 = ..; T=..;
  exact_f = @(t) ..;
  h=zeros(7,1); err=zeros(7,1);
  for i=0:6
    h(i+1)=2^(-i-3);  N=2^(i+3);
    [t,y] = esv(f,t0,T,y0,N);
    err(i+1)=abs(y(:,end)-exact_f(T));   % norm
  end
  loglog(h,err,'-*')
  p=polyfit(log(h),log(err), 1);
  emp_konv_rate = p(1);
end
```

RK klassisch

```
function [t,y] = klassischesRK(f,t0,T,y0,N)
  h = (T-t0)/N;
  t = zeros(1,N+1);
  y = zeros(length(y0), N+1);
  t(1)=t0;
  y(:,1) = y0;
  for k=1:N
    t(k+1)=t0+k*h;
    k1 = f(t(k), y(:,k));
    k2 = f(t(k)+h/2, y(:,k)+h/2*k1);
    k3 = f(t(k)+h/2, y(:,k)+h/2*k2);
    k4 = f(t(k)+h, y(:,k)+h*k3);
    y(:,k+1)=y(:,k)+h/6*(k1+2*k2+2*k3...
              +k4);
  end
end
```

III. Adaptive Schrittweitensteuerung

Adaptives Heun Verfahren

```
function [t,y]=adaptHeun(f,t0,T,y0,h0,atol,rtol)
  p=2;  % konsistenzord. Heun
  t=t0;  y=y0;
  hj = h0;
  while (t(end)<T)
    tj = t(end);
    yj = y(:,end);
    hj = min(T-tj,hj);
    % Heun auf vollem Intervall
    k1= f(tj,yj);
    k2= f(tj+hj,yj+hj*k1);
    yip= yj + hj/2*(k1+k2);
    % Heun auf 2 Teilintervallen
    k2=f(tj+hj/2,yj+hj/2*k1);
    yj12 = yj + hj/4*(k1+k2);
    k1= f(tj+hj/2, yj12);
    k2 = f(tj+hj,yj12+hj/2*k1);
    yi2 = yj12 + hj/4*(k1+k2);
    epsilon = norm(yi2-yip)/(2^p-1);
    if (epsilon < atol & norm(yj)*rtol)
      t=[t,tj+hj];
      y=[y,yi2];      } akzeptieren
    else
      hj=0.5*hj;   % halbieren
    end
  end
end
```

Adaptives RK Verfahren

```
function [ts,ys]=RKF45(f,t0,T,y0,h0,atol,rtol)
  y=y0; t=t0; h=h0;
  p=4;  % Ordnung
  hmin, facmin, facmax, fac = ...;
  epsilon = atol + norm(y0)*rtol;
  ts = t0;   ys = y0;
  while (t<T)
    tol = atol + norm(y)*rtol;
    h = h* min(facmax,max(facmin, ...
        fac*(tol/epsilon)^(1/(p+1))));
    if h<hmin
      break;
    end
    h=min(h,T-t);
    % Rechne RK4 und RK5
    epsilon=norm(y5-y4);
    if epsilon < tol
      t=t+h;
      y = y5;
      ts = [ts,t]; ys=[ys,y];
    end
  end
end
```

IV. Newton Verfahren

Fixpunkt

```
function x = fixpunkt(phi, x0, niter)
  x = x0;
  for i=1:niter
    xold = x;
    x = phi(xold);
  end
end

function fixpunkt(phi,x0,atol,rtol)
  xold = x0;
  xnew = phi(x0);
  x = [x0];
  err = abs(xnew-xold);
  while (err>atol + rtol*abs
  while err > (abstol + rtol*abs(xnew))
    xnew = phi(xold);
    err = abs(xnew-xold);
    x = [x, xnew];
    xold = xnew;
  end
end

function fixpunkt_konv(phi, x0)
  atol = 1e-7;  rtol = 1e-5;
  x = fixpunkt(phi,x0,atol,rtol);
  err = abs(x(end)-x(1:end-1));

  for i=2:length(x)-2
    p = (log(err(i+1))-log(err(i)))/...
        (log(err(i))-log(err(i-1)));

    c = err(i)/(err(i-1)^p);
  end
  % p : Konvergenzordnung
  % c : Konvergenzrate
end
```

n Dimensionen Newton

Lösung des Nullstellenproblems $F(x)=0$

DF: Jacobi Matrix von F

```
function x = newton(x0,F,DF,nMax,atol,rtol)
  x = x0;
  for i=1:nMax
    x_old = x;
    dx = -DF(x)\F(x);
    x = x+dx;
    if norm(DF(x_old)\F(x)) < tol
      break
    end
  end
end
```

`\` löst LGS, nicht $F(x)\cdot inv(DF(x))$ mlsolve

Implizit Euler + Newton

```
function impEuler(F, DF, T, N, y0)
  h = T/N;
  y = zeros(length(y0), N+1);
  y(:,1) = y0;
  for i=1:N
    F = @(x) y(:,i) + h*f(h*(i+1),y) - x;
    DF = @(x) h*Df(h*(i+1),x) - 1;
    y(:,i+1) = newton(F,DF,y(:,i));
  end
end
```

Impl. Euler: $y_{n+1} = y_n + h\cdot f(t_{n+1}, y_{n+1})$

$\hookrightarrow 0 = y_n + h\cdot f(t_{n+1}, y_{n+1}) - y_{n+1} = F$

$\frac{d}{dy_{n+1}} F = DF = h\cdot f'(t_{n+1}, y_{n+1}) - 1$

Finite Differenzen Approximation

DF_x wird approximiert durch

$DF_x = (F(x+h) - F(x))/h;$

V.3 Rückwärtsdiff. Method

Adams Bashforth AB2

```
function [t,y] = AB2(f,t0,T,y0,N)
  t = zeros(1,N+1);
  h = T/N;
  t = linspace(t0,T,N+1);
  y = zeros(1,N+1);
  y(1) = y0;
  % erster Schritt mit heun
  y(2) = heun_schritt(f,y(1),t(1),h);
  % AB2
  for n=3:N+1
    y(n) = y(n-1) + h/2*(3*f(t(n-1),...
      y(n-1)) - f(t(n-2),y(n-2)));
  end
end

function y = heun_schritt(f,yi,ti,hi)
  k1 = f(ti,yi);
  k2 = f(ti+hi, yi+hi*k1);
  y = yi + hi/2*(k1+k2);
end
```

Adams Moulton AM2

```
function [t,y] = AM2(f,t0,T,y0,N,M)
  h = T/N;  t = linspace(t0,T,N+1);
  y = zeros(1,N+1);  y(1) = y0;
  y(2) = heun_schritt(f,y(1),t(1),h);

  for n=3:N+1
    y_itr = zeros(1,M+1);
    % Predictor: AB2
    y_itr(1) = AB2(f,t(n-1),y(n-1),t(n-2),y(n-2),
                   h);
    % corrector: AM2
    for k=1:M
      y_itr(k+1) = y(n-1) + h/12* ...
        (5*f(t(n),y_itr(k)) + 8*f(t(n-1),y(n-1))
        - f(t(n-2),y(n-2)));
    end
    y(n) = y_itr(M+1);
  end
end
function y = AB2(f,ti,yi,tim,yim,hi)
  y = yi + hi/2*(3*f(ti,yi) - f(tim,yim));
end
```

Ableitungen

$\frac{d}{dx}\ln x = \frac{1}{x}, x>0$ $\quad \frac{d}{dx}a^x = a^x \ln(a)$ $\quad \frac{d}{dx}\sin x = \cos x$ $\quad \frac{d}{dx}\cos x = -\sin x$ $\quad \frac{d}{dx}\tan x = \frac{1}{\cos^2 x}$

$\frac{d}{dx}\cot x = \frac{-1}{\sin^2 x}$ $\quad \frac{d}{dx}\sin^{-1}x = \frac{1}{\sqrt{1-x^2}}$ $\quad \frac{d}{dx}\cos^{-1}x = \frac{-1}{\sqrt{1-x^2}}$ $\quad \frac{d}{dx}\tan^{-1}x = \frac{1}{1+x^2}$

$\frac{d}{dx}\sinh x = \cosh x$ $\quad \frac{d}{dx}\cosh x = \sinh x$ $\quad \frac{d}{dx}\tanh x = \frac{1}{\cosh^2 x}$ $\quad \frac{d}{dx}\coth x = \frac{-1}{\sinh^2 x}$

$(f\cdot g)' = f'\cdot g + f\cdot g'$ $\quad f(g(x))' = f'(g(x))\cdot g'(x)$ $\quad \left(\frac{f}{g}\right)' = \frac{f'g - fg'}{g^2}$

Integrale

$\int x^n = \frac{1}{n+1}x^{n+1}$ $\quad \int \frac{1}{x} = \ln|x|$ $\quad \int a^x = \frac{1}{\ln a}a^x$ $\quad \int \cos x = \sin x$ $\quad \int \sin x = -\cos x$

$\int \frac{1}{\cos^2 x} = \tan x$ $\quad \int \tan^2 x = \tan x - x$ $\quad \int \frac{1}{\sin^2 x} = -\cot x$ $\quad \int \cot^2 x = -\cot x - x$

$\int \frac{1}{\sqrt{1-x^2}} = \sin^{-1}x = -\cos^{-1}x$ $\quad \int \frac{1}{1+x^2} = \tan^{-1}x$ $\quad \int \cosh x = \sinh x$ $\quad \int \sinh x = \cosh x$

$\int \frac{1}{\cosh^2 x} = \tanh x$ $\quad \int \tanh^2 x = x - \tanh x$ $\quad \int \frac{1}{\sinh^2 x} = -\coth x$ $\quad \int \coth^2 x = x - \coth x$

$\int \sqrt{x} = \frac{2}{3}\sqrt{x^3}$ $\quad \int \frac{1}{\sqrt{x}} = 2\sqrt{x}$ $\quad \int \sin^2 x = -\frac{1}{4}\sin(2x) + \frac{1}{2}x$ $\quad \int \cos^2 x = \frac{1}{4}\sin(2x) + \frac{1}{2}x$

$\int_a^b f(\varphi(x))\,\varphi'(x)\,dx = \int_{\varphi(a)}^{\varphi(b)} f(u)\,du$ $\quad \int_a^b u(x)v'(x)\,dx = u(x)v(x)\Big|_a^b - \int_a^b u'(x)v(x)\,dx$ $\quad \int \quad \frac{d}{dx}$

$\int \frac{x}{1+x} = x - \log(x+1)$

Trigo

$\sin^2 x + \cos^2 x = 1$ $\quad \cos(2x) = \cos^2(x) - \sin^2(x) = 1 - 2\sin^2(x) = 2\cos^2(x) - 1$

$c^2 = a^2 + b^2 + 2ab\cos\gamma$ $\quad \frac{a}{\sin\alpha} = \frac{b}{\sin\beta} = \frac{c}{\sin\gamma}$ $\quad \cos^2(x) = \frac{1}{2} + \frac{1}{2}\cos(2x)$

$\sin(a+b) = \sin(a)\cos(b) + \sin(b)\cos(a)$ $\quad \sin(a-b) = \sin(a)\cos(b) - \sin(b)\cos(a)$

$\cos(a+b) = \cos(a)\cos(b) - \sin(a)\sin(b)$ $\quad \cos(a-b) = \cos(a)\cos(b) + \sin(a)\sin(b)$

$\sin(2x) = 2\cdot\sin x\cdot\cos x$ $\quad \sin x + \sin y = 2\cos\left(\frac{x-y}{2}\right)\sin\left(\frac{x+y}{2}\right)$

$\cos(a)\cos(b) = \frac{1}{2}(\cos(a-b) + \cos(a+b))$ $\quad \sin(a)\sin(b) = \frac{1}{2}(\cos(a-b) - \cos(a+b))$

$\cos(a)\sin(b) = \frac{1}{2}(\sin(a+b) - \sin(a-b))$

Summen

$\sum_{k=1}^n k = \frac{n(n+1)}{2}$ $\quad \sum_{k=1}^n k^2 = \frac{n(n+1)(2n+1)}{6}$ $\quad \sum_{k=1}^n k^3 = \frac{n^2(n+1)^2}{4}$ $\quad \sum_{k=0}^{n-1} e^{\frac{k}{n}} = \frac{e-1}{e^{\frac{1}{n}}-1}$

$\sum_{k=0}^n \binom{n}{k} a^{n-k} b^k = (a+b)^n$ $\quad \binom{n}{k} = \frac{n!}{k!(n-k)!}$ $\quad \sum_{k=0}^\infty \frac{x^k}{k!} = e^x$ $\quad \sum_{k=0}^\infty (-1)^k \frac{x^{2k}}{(2k)!} = \cos(x)$

$\sum_{k=0}^\infty (-1)^k \frac{x^{2k+1}}{(2k+1)!} = \sin(x)$

Fourier

$\hat{f}(\omega) = \frac{1}{2\pi}\int_{-\infty}^{\infty} f(t)e^{-i\omega t}\,dt$ $\quad f(t) = \int_{-\infty}^{\infty} \hat{f}(\omega)e^{i\omega t}\,d\omega$

$f(at) \Leftrightarrow \frac{1}{|a|}\hat{f}\left(\frac{\omega}{a}\right)$ $\quad \hat{f}(b\omega) \Leftrightarrow \frac{1}{|b|}f\left(\frac{t}{b}\right)$ $\quad f(t-\tau) \Leftrightarrow \hat{f}(\omega)\cdot e^{-i\omega\tau}$ $\quad \hat{f}(\omega-\Omega) \Leftrightarrow f(t)e^{i\Omega t}$

$f'(t) \Leftrightarrow i\omega\cdot\hat{f}(\omega)$ $\quad f(t)*g(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)\,d\tau \overset{2\pi}{\Leftrightarrow} \hat{f}(\omega)\cdot\hat{g}(\omega)$ $\quad t^n f(t) \Leftrightarrow (-i)^n \hat{f}^{(n)}(\omega)$

$\sin(at) \multimap \frac{1}{2i}(\delta(\omega+a) - \delta(\omega-a))$ $\quad \cos(at) \multimap \frac{1}{2}(\delta(\omega+a) + \delta(\omega-a))$ $\quad e^{-iat} \multimap \delta(\omega+a)$

$\begin{cases} 1 & -L\le t\le L \\ 0 & \text{else} \end{cases} \multimap \frac{\sin(L\omega)}{\omega\pi} = \frac{L}{\pi}\operatorname{sinc}(L\omega)$ $\quad e^{-a|t|} \multimap \frac{2a}{\omega^2+a^2}$ $\quad \operatorname{sinc}(\omega_0 t) \multimap \frac{\pi}{\omega_0}\operatorname{rect}\left(\frac{\omega}{2\omega_0}\right)$

$f(t)\cdot g(t) \Leftrightarrow (\hat{f}*\hat{g})(\omega)$ $\quad e^{-at^2} \multimap \frac{1}{\sqrt{4\pi a}}e^{\frac{-\omega^2}{4a}}$ $\quad$ Parseval: $\boxed{\int_{-\infty}^{\infty}|x(t)|^2 dt = \frac{1}{2\pi}\int_{-\infty}^{\infty}|X(\omega)|^2 d\omega}$

Taylor Reihen

$f(x;x_0) = \sum_{n=0}^{\infty}\frac{f^{(n)}(x_0)}{n!}(x-x_0)^n \approx f(x_0) + f'(x_0)(x-x_0) + \frac{f''(x_0)}{2}(x-x_0)^2 + \dots$

2D $\quad g(x,y,x_0,y_0) \approx g(x_0,y_0) + \partial_x g(x_0,y_0)(x-x_0) + \partial_y g(x_0,y_0)(y-y_0) + \frac{1}{2}\partial_{xx} g()(x-x_0)^2 + \dots$

$f(t+\Delta t, y+\Delta y) \approx f(t,y) + \partial_t f(t,y)\Delta t + \partial_y f(t,y)\Delta y$

$+ \frac{1}{2}\partial_{t^2} f(t,y)\Delta t^2 + \partial_{ty} f(t,y)\Delta t\Delta y + \frac{1}{2}\partial_{y^2} f(t,y)\Delta y^2 + \dots$

$\sqrt{1+x} \approx 1 + \frac{x}{2} - \frac{x^2}{8} + \frac{x^3}{16} - \frac{5x^4}{128}$ $\quad \exp(ax) = \sum_0^\infty \frac{(ax)^k}{k!}$

$\sin(ax) = \sum_{k=0}^\infty (-1)^k \frac{(ax)^{2k+1}}{(2k+1)!}$ $\quad \cos(ax) = \sum_{k=0}^\infty (-1)^k \frac{(ax)^{2k}}{(2k)!}$

Komplexe Zahlen

$z = a + ib$ $\quad z = r\cdot e^{i\varphi}$ $\quad |z| = \sqrt{a^2+b^2} = r$ $\quad \arg(z) = \varphi = \arctan\left(\frac{b}{a}\right)$

$z^* = a - ib = r\cdot e^{-i\varphi}$ $\quad e^{ix} = \cos(x) + i\sin(x)$ $\quad \operatorname{Re}\{e^{ix}\} = \cos x$ $\quad \operatorname{Im}\{e^{ix}\} = \sin x$

$\operatorname{Im}\{\sqrt{i}\} = \frac{1}{\sqrt{2}}$ $\quad \circ\ |z|^2 = z\cdot z^*$ $\quad \circ\ i = e^{i\frac{\pi}{2}}$ $\quad \circ\ \operatorname{Re}\{z\} = \frac{1}{2}(z+z^*)$

Bestimmte Integrale

$\circ \int_{-\infty}^{\infty} e^{-ax^2}dx = \sqrt{\frac{\pi}{a}}$ $\quad \circ \int_{-\infty}^{\infty} xe^{-ax^2} = 0, a>0$ $\quad \circ \int_{-\infty}^{\infty} x^2 e^{-ax^2}dx = \frac{1}{2}\sqrt{\frac{\pi}{a^3}}$ $\quad \circ \int_{-\infty}^{\infty} e^{-ax^2+ibx}dx = \sqrt{\frac{\pi}{a}}e^{-b^2/4a}$

$\circ \int_0^\pi \sin^3 x\,dx = \frac{4}{3}$ $\quad \circ \int_0^{2\pi} \sin^2 x\,dx = \int_0^{2\pi}\cos^2 x\,dx = \pi$ $\quad \circ \int_0^\pi \cos^2 x\sin x\,dx = \frac{2}{3}$ $\quad \circ \int_0^\pi \sin^2 x = \int_0^\pi \cos^2 = \frac{\pi}{2}$

$\int_0^\infty x^3 e^{-ax^2}dx = \frac{1}{2a^2}$

Geometrie

Kreis: $A = \pi r^2$ $\quad U = 2\pi r$ $\quad (x-x_0)^2 + (y-y_0)^2 = r^2$

Kugel: $V = \frac{4\pi}{3}r^3$ $\quad O = 4\pi r^2$ $\quad x^2 + y^2 + z^2 = r^2$

Vektoranalysis

S.v. Gauss: $\int_{\partial V}\vec{F}\cdot\vec{n}\,dA = \int_V \nabla\cdot\vec{F}\,dV$ $\quad$ S.v. Stokes: $\int_{\partial A}\vec{F}\cdot d\vec{s} = \int_A [\nabla\times\vec{F}]\cdot\vec{n}\,dA$

<u>Linienintegral</u> $\int_{\partial A}\vec{F}\,d\vec{s}$

1) Linie parametrisieren: $t\in[a,b] \to \begin{pmatrix}x\\y\\z\end{pmatrix} = f(t)$
2) $d\vec{s}$ ist tangential $\to \begin{pmatrix}x\\y\\z\end{pmatrix}$ nach t ableiten
3) $\int_{\partial A}\vec{F}\,d\vec{s} = \int_a^b \vec{F}\cdot\frac{d}{dt}\begin{pmatrix}x\\y\\z\end{pmatrix}dt$

<u>Oberfl.-Integral</u> $\quad \int_{\partial V}\vec{F}\cdot\vec{n}\,dA = \int_0^{2\pi}d\varphi\int_0^\pi d\theta\ \underbrace{R^2\sin\theta}_{dA}\ \underbrace{\begin{pmatrix}\sin\theta\cos\varphi\\ \sin\theta\sin\varphi\\ \cos\theta\end{pmatrix}}_{\vec{n}}\cdot\vec{F}$

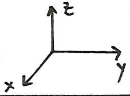
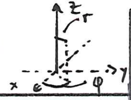
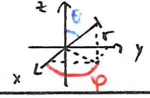
Kugel mit Radius R

<u>Satz von Schwarz</u> Partielle Abl. dürfen vertauscht werden $\boxed{\partial_x\partial_y f = \partial_y\partial_x f}$

<u>Betrag</u> $|\vec{r}-\vec{r}'| = \sqrt{r^2 + r'^2 - 2\vec{r}\cdot\vec{r}'}$

<u>Jacobi Matrix</u> $\vec{f}(\vec{x}) = (f_1(\vec{x}), \dots, f_m(\vec{x})) \to J_f(a) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}(a) & \frac{\partial f_1}{\partial x_2}(a) & \dots & \frac{\partial f_1}{\partial x_n}(a) \\ \vdots & & \ddots & \\ \frac{\partial f_m}{\partial x_1}(a) & \dots & \dots & \frac{\partial f_m}{\partial x_n}(a)\end{bmatrix}$

$\vec{x} = \begin{pmatrix}x_1\\x_2\\ \vdots\\ x_n\end{pmatrix}$

Operation	Cartesian (x,y,z) 	Cylindrical (r,φ,z) 	Spherical (r,θ,φ) 
Vector field A	$A_x\vec{u}_x + A_y\vec{u}_y + A_z\vec{u}_z$	$A_r\vec{u}_r + A_\varphi\vec{u}_\varphi + A_z\vec{u}_z$	$A_r\vec{u}_r + A_\theta\vec{u}_\theta + A_\varphi\vec{u}_\varphi$
Gradient ∇f	$\frac{\partial f}{\partial x}\vec{u}_x + \frac{\partial f}{\partial y}\vec{u}_y + \frac{\partial f}{\partial z}\vec{u}_z$	$\frac{\partial f}{\partial r}\vec{u}_r + \frac{1}{r}\frac{\partial f}{\partial \varphi}\vec{u}_\varphi + \frac{\partial f}{\partial z}\vec{u}_z$	$\frac{\partial f}{\partial r}\vec{u}_r + \frac{1}{r}\frac{\partial f}{\partial \theta}\vec{u}_\theta + \frac{1}{r\sin\theta}\frac{\partial f}{\partial \varphi}\vec{u}_\varphi$
Divergence $\nabla\cdot A$	$\frac{\partial A_x}{\partial x} + \frac{\partial A_y}{\partial y} + \frac{\partial A_z}{\partial z}$	$\frac{1}{r}\frac{\partial(rA_r)}{\partial r} + \frac{1}{r}\frac{\partial A_\varphi}{\partial \varphi} + \frac{\partial A_z}{\partial z}$	$\frac{1}{r^2}\frac{\partial(r^2A_r)}{\partial r} + \frac{1}{r\cdot\sin\theta}\frac{\partial}{\partial\theta}(A_\theta\sin\theta) + \frac{1}{r\cdot\sin\theta}\cdot\frac{\partial A_\varphi}{\partial\varphi}$
Curl $\nabla\times A$	$(\partial_y A_z - \partial_z A_y)\vec{u}_x + (\partial_z A_x - \partial_x A_z)\vec{u}_y + (\partial_x A_y - \partial_y A_x)\vec{u}_z$	$(\frac{1}{r}\partial_\varphi A_z - \partial_z A_\varphi)\vec{u}_r + (\partial_z A_r - \partial_r A_z)\vec{u}_\varphi \cdots + \frac{1}{r}(\partial_r(r\cdot A_\varphi) - \partial_\varphi A_r)\vec{u}_z$	$\frac{1}{r\cdot\sin\theta}\left(\frac{\partial}{\partial\theta}(A_\varphi\sin\theta) - \frac{\partial A_\theta}{\partial\varphi}\right)\vec{u}_r + \frac{1}{r}\left(\frac{1}{\sin\theta}\frac{\partial A_r}{\partial\varphi} - \frac{\partial}{\partial r}(r\cdot A_\varphi)\right)\vec{u}_\theta + \frac{1}{r}\left(\frac{\partial}{\partial r}(r\cdot A_\theta) - \frac{\partial A_r}{\partial\theta}\right)\vec{u}_\varphi$
Laplace op. $\nabla^2 f = \Delta f$	$\frac{\partial^2 f}{\partial x^2} + \frac{\partial^2 f}{\partial y^2} + \frac{\partial^2 f}{\partial z^2}$	$\frac{1}{r}\frac{\partial}{\partial r}\left(r\frac{\partial f}{\partial r}\right) + \frac{1}{r^2}\left(\frac{\partial^2 f}{\partial\varphi^2}\right) + \frac{\partial^2 f}{\partial z^2}$	$\frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial f}{\partial r}\right) + \frac{1}{r^2\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial f}{\partial\theta}\right) + \frac{1}{r^2\sin^2\theta}\cdot\frac{\partial^2 f}{\partial\varphi^2}$
Vector Laplacian $\nabla^2 A = \Delta A$ $= \nabla(\nabla\cdot A) - \nabla\times(\nabla\times A)$	$\nabla^2 A_x\vec{u}_x + \nabla^2 A_y\vec{u}_y + \nabla^2 A_z\vec{u}_z$	$\left(\nabla^2 A_r - \frac{A_r}{r^2} - \frac{2}{r^2}\frac{\partial A_\varphi}{\partial\varphi}\right)\vec{u}_r + \left(\nabla^2 A_\varphi - \frac{A_\varphi}{r^2} + \frac{2}{r^2}\frac{\partial A_r}{\partial\varphi}\right)\vec{u}_\varphi + \nabla^2 A_z\vec{u}_z$	$\left(\nabla^2 A_r - \frac{2A_r}{r^2} - \frac{2}{r^2\sin\theta}\frac{\partial(A_\theta\sin\theta)}{\partial\theta} - \frac{2}{r^2\sin\theta}\frac{\partial A_\varphi}{\partial\varphi}\right)\vec{u}_r + \left(\nabla^2 A_\theta - \frac{A_\theta}{r^2\sin^2\theta} + \frac{2}{r^2}\frac{\partial A_r}{\partial\theta} - \frac{2\cos\theta}{r^2\sin^2\theta}\frac{\partial A_\varphi}{\partial\varphi}\right)\vec{u}_\theta + \left(\nabla^2 A_\varphi - \frac{A_\varphi}{r^2\sin^2\theta} + \frac{2}{r^2\sin\theta}\frac{\partial A_r}{\partial\varphi} + \frac{2\cos\theta}{r^2\sin^2\theta}\frac{\partial A_\theta}{\partial\varphi}\right)\vec{u}_\varphi$
differential Volume dV	$dx\,dy\,dz$	$r\cdot dr\,d\varphi\,dz$	$r^2\sin\theta\cdot dr\,d\theta\,d\varphi$

Unit vector conversion

in terms of dest./src.

	Cartesian	Cylindrical	Spherical
Cartesian	N/A	$\vec{u}_x = \cos\varphi\vec{u}_r - \sin\varphi\vec{u}_\varphi$ $\vec{u}_y = \sin\varphi\vec{u}_r + \cos\varphi\vec{u}_\varphi$ $\vec{u}_z = \vec{u}_z$ $\vec{u}_x = \frac{x\vec{u}_r - y\vec{u}_\varphi}{\sqrt{x^2+y^2}}$ $\vec{u}_y = \frac{y\vec{u}_r + x\vec{u}_\varphi}{\sqrt{x^2+y^2}}$ $\vec{u}_z = \vec{u}_z$	$\vec{u}_x = \sin\theta\cos\varphi\vec{u}_r + \cos\theta\cos\varphi\vec{u}_\theta - \sin\varphi\vec{u}_\varphi$ $\vec{u}_y = \sin\theta\sin\varphi\vec{u}_r + \cos\theta\sin\varphi\vec{u}_\theta + \cos\varphi\vec{u}_\varphi$ $\vec{u}_z = \cos\theta\vec{u}_r - \sin\theta\vec{u}_\theta$ $\vec{u}_x = \frac{x(\sqrt{x^2+y^2}\vec{u}_r + z\vec{u}_\theta) - y\sqrt{x^2+y^2+z^2}\vec{u}_\varphi}{\sqrt{x^2+y^2}\sqrt{x^2+y^2+z^2}}$ $\vec{u}_y = \frac{y(\sqrt{x^2+y^2}\vec{u}_r + z\vec{u}_\theta) + x\sqrt{x^2+y^2+z^2}\vec{u}_\varphi}{\sqrt{x^2+y^2}\sqrt{x^2+y^2+z^2}}$ $\vec{u}_z = \frac{z\vec{u}_r - \sqrt{x^2+y^2}\vec{u}_\theta}{\sqrt{x^2+y^2+z^2}}$
Cylindrical	$\vec{u}_r = \frac{x\vec{u}_x + y\vec{u}_y}{\sqrt{x^2+y^2}}$ $\vec{u}_\varphi = \frac{-y\vec{u}_x + x\vec{u}_y}{\sqrt{x^2+y^2}}$ $\vec{u}_z = \vec{u}_z$ $\vec{u}_r = \cos\varphi\vec{u}_x + \sin\varphi\vec{u}_y$ $\vec{u}_\varphi = -\sin\varphi\vec{u}_x + \cos\varphi\vec{u}_y$ $\vec{u}_z = \vec{u}_z$	N/A	$\vec{u}_\rho = \sin\theta\vec{u}_r + \cos\theta\vec{u}_\theta$ $\vec{u}_\varphi = \vec{u}_\varphi$ $\vec{u}_z = \cos\theta\vec{u}_r - \sin\theta\vec{u}_\theta$ $\vec{u}_\rho = \frac{\rho\vec{u}_r + z\vec{u}_\theta}{\sqrt{\rho^2+z^2}}$ $\vec{u}_z = \frac{z\vec{u}_r - \rho\vec{u}_\theta}{\sqrt{\rho^2+z^2}}$ $\vec{u}_\varphi = \vec{u}_\varphi$
Spherical	$\vec{u}_r = \frac{x\vec{u}_x + y\vec{u}_y + z\vec{u}_z}{\sqrt{x^2+y^2+z^2}}$ $\vec{u}_\theta = \frac{(x\vec{u}_x + y\vec{u}_y)z - (x^2+y^2)\vec{u}_z}{\sqrt{x^2+y^2}\sqrt{x^2+y^2+z^2}}$ $\vec{u}_\varphi = \frac{-y\vec{u}_x + x\vec{u}_y}{\sqrt{x^2+y^2}}$ $\vec{u}_r = \sin\theta(\cos\varphi\vec{u}_x + \sin\varphi\vec{u}_y) + \cos\theta\vec{u}_z$ $\vec{u}_\theta = \cos\theta(\cos\varphi\vec{u}_x + \sin\varphi\vec{u}_y) - \sin\theta\vec{u}_z$ $\vec{u}_\varphi = -\sin\varphi\vec{u}_x + \cos\varphi\vec{u}_y$	$\vec{u}_r = \frac{\rho\vec{u}_\rho + z\vec{u}_z}{\sqrt{\rho^2+z^2}}$ $\vec{u}_\theta = \frac{z\vec{u}_\rho - \rho\vec{u}_z}{\sqrt{\rho^2+z^2}}$ $\vec{u}_\varphi = \vec{u}_\varphi$ $\vec{u}_r = \sin\theta\vec{u}_\rho + \cos\theta\vec{u}_z$ $\vec{u}_\theta = \cos\theta\vec{u}_\rho - \sin\theta\vec{u}_z$ $\vec{u}_\varphi = \vec{u}_\varphi$	N/A

Identities

- $\nabla\times(\nabla f) = 0$
- $\nabla\cdot(\nabla\times A) = 0$
- $\nabla\times(\nabla\times A) = \nabla(\nabla\cdot A) - \nabla^2 A$
- $\nabla\times(fA) = f(\nabla\times A) - A\times(\nabla f)$
- $\nabla\cdot(A\times B) = (\nabla\times A)\cdot B - A\cdot(\nabla\times B)$
- $A\cdot(B\times C) = B\cdot(C\times A) = C\cdot(A\times B)$
- $A\times(B\times C) = B\times(C\times A) = C\times(A\times B) = 0$
- $A\times B = -B\times A$
- $A\times(B\times C) = (A\cdot C)B - (A\cdot B)C = B(A\cdot C) - C(A\cdot B)$

Coordinate conversion

To ↓ From →	Cartesian	Cylindrical	Spherical
Cartesian	...	$x = \rho\cos\varphi$ $y = \rho\sin\varphi$ $z = z$	$x = r\sin\theta\cos\varphi$ $y = r\sin\theta\sin\varphi$ $z = r\cos\theta$
Cylindrical	$\rho = \sqrt{x^2+y^2}$ $\varphi = \arctan\left(\frac{y}{x}\right)$ $z = z$	...	$\rho = r\sin\theta$ $\varphi = \varphi$ $z = r\cos\theta$
Spherical	$r = \sqrt{x^2+y^2+z^2}$ $\theta = \arctan\left(\frac{\sqrt{x^2+y^2}}{z}\right)$ $\varphi = \arctan\left(\frac{y}{x}\right)$	$r = \sqrt{\rho^2+z^2}$ $\theta = \arctan\left(\frac{\rho}{z}\right)$ $\varphi = \varphi$	...