Numerik

Eine problemangepaßte Einführung

Michael Hinze

Vorlesungsskript zur Vorlesung **Numerik**, gehalten im SS 2003 an der TU Dresden,
Stand 04.07.2003

Inhaltsverzeichnis

Abbildungsverzeichnis

Danksagung

Herrn Dr. Gerd Pönisch und Herrn Dr. Reiner Vanselow meinen herzlichen Dank für viele Hinweise und Verbesserungsvorschläge bei der Erstellung des Manuskriptes.

1 Modellierung

Nach Pinnau [12] geschieht die Modellierung eines realen Problems in Form einer Kaskade, bei der jede Stufe mehrere Schritte umfasst. Jede Stufe der Kaskade steht dabei für die Komplexität des gegenwärtigen Modells, wobei die Kaskade von einfachen zu komplexen Modellen führen und jede Stufe wiederum zum besseren Verständnis des realen Problems beitragen sollte. In jeder Stufe der Kaskade sind die folgenden Schritte auszuführen.

1. Verstehe das Problem
2. Diskutiere mögliche mathematische Methoden
3. Formuliere ein mathematisches Modell
4. Löse das mathematische Modell entweder analytisch oder numerisch
5. Interpretiere die Lösung
6. Validiere die Lösung durch Vergleich mit dem Ausgangsproblem
7. Verfeinere eventuell das mathematische Modell $\hookrightarrow$ nächste Stufe der Kaskade und gehe nach 2.
8. Präsentiere die Ergebnisse

Im Folgenden werden Facetten dieses Vorgehens beispielhaft anhand der Modellierung von Transportvorgängen in der Physik und von Portfolio Optimierung erläutert. Es stellt sich heraus (und das nicht nur bei diesen Beispielklassen), daß die mathematischen Modelle i.A. nicht analytisch gelöst werden können und deshalb numerische Methoden zur Lösung verwendet werden sollten.

1.1 Transportprozesse in der Physik, makroskopisch Modellierung

Im Folgenden wird die Modellbildung für

- Ausbreitung eines Gases in der Atmosphäre,
- Ausbreitung einer Substanz im Lösungsmittel und/oder
- Ausbreitung von Temperatur

durchgeführt. Grundlage für das mathematische Modell sind hier Bilanzen. Der Ausgangspunkt ist dabei das Prinzip der Massenerhaltung. In einem Volumenelement gilt:
Die zeitliche Änderung der Masse einer Substanz ergibt sich aus

i) der in Quellen erzeugten Masse,

ii) der in Senken vernichteten Masse,

iii) der durch die Oberfläche des Volumenelements eintretenden Masse,

iv) der durch die Oberfläche des Volumenelements austretenden Masse.

Bezeichne

$v(t,x)$ $[\frac{m}{s}]$ die Geschwindigkeit des Stoffes am Ort x zur Zeit t,
$\rho(t,x)$ $[\frac{kg}{m^3}]$ dessen Dichte .

Dann heißt

$$w(t,x) \;:=\; v(t,x)\;\rho(t,x)\left[\frac{kg}{m^2 s}\right], w = \begin{bmatrix} w^1 \\ w^2 \\ w^3 \end{bmatrix}$$

Teilchenstromdichte.
Zur Beschreibung von iii) und iv) betrachte ein Volumenelement um x zur Zeit t. Das Volumen des

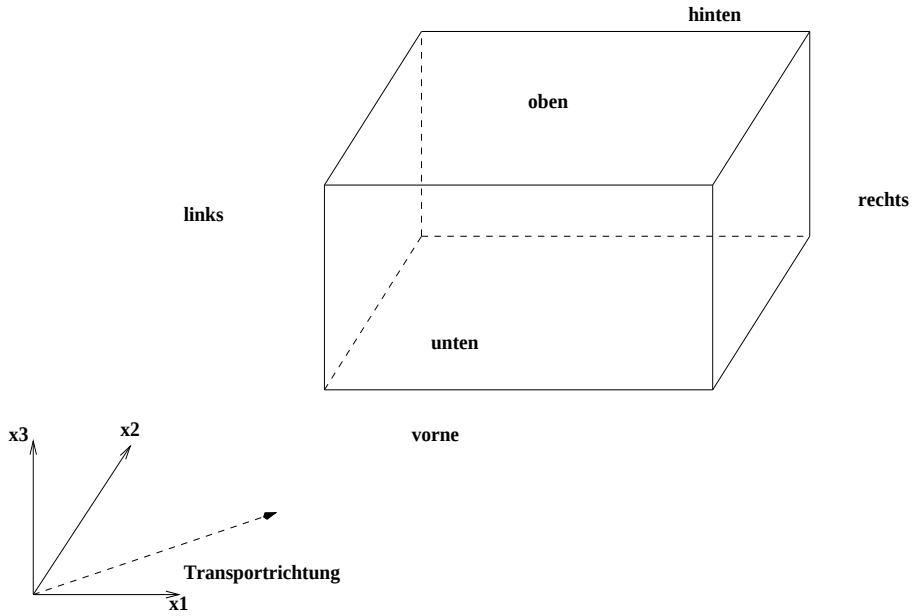


Abbildung 1: Volumenelement, F_1^1 links, F_1^2 rechts, F_2^1 vorne, F_2^2 hinten, F_3^1 unten, F_3^2 oben

Elements ist gegeben durch

$$\Delta V = \Delta x_1 \Delta x_2 \Delta x_3;$$

während des Zeitraums Δt tritt durch

$$\begin{array}{lll} F_1^1 & \text{die Masse} & w_1(t, x_1 - \frac{1}{2}\Delta x_1, x_2, x_3)\Delta t |F_1^1| \\ F_2^1 & \text{die Masse} & w_2(t, x_1, x_2 - \frac{1}{2}\Delta x_2, x_3)\Delta t |F_2^1| \\ F_3^1 & \text{die Masse} & w_3(t, x_1, x_2, x_3 - \frac{1}{2}\Delta x_3)\Delta t |F_3^1| \end{array}$$

in das Volumenelement ein, analog durch die gegenüberliegenden Flächen Masse aus. Für die Massenänderung im Volumenelement gilt demnach

$$\begin{array}{rcl} \Delta m^o & = & \Delta m_1 + \Delta m_2 + \Delta m_3, \\ \Delta m_i & = & [w_i(t, \cdot, x_i - \frac{1}{2}\Delta x_i, \cdot) - w_i(t, \cdot, x_i + \frac{1}{2}\Delta x_i, \cdot)]\Delta t \left\{ |F_i^1| + |F_i^2| \right\} /2, \end{array}$$

in erster Näherung also

$$\begin{array}{rcl} \Delta m_i & \doteq & -\partial_{x_i} w_i(t, x)\Delta t \Delta V, \\ \Delta m^o & \doteq & -\mathrm{div}_x w(t, x)\Delta t \Delta V. \end{array}$$

Zur Modellierung enthalte das Volumenelement jetzt zusätzlich Quellen und/oder Senken mit Quelldichten

$$q_k[\frac{kg}{m^3 s}], \quad k = 1, \ldots, K.$$

Die im Ort gemittelte Quelldichte ist dann

$$q := \frac{1}{K} \sum_{j=1}^{K} q_j.$$

Während des Zeitraums Δt wird demnach am Ort x die Masse

$$\int_t^{t+\Delta t} q(s, x) ds \doteq q(t, x)\Delta t$$

erzeugt, im Volumenelement in erster Näherung also die Masse

$$\Delta m^t = q(t,x)\Delta t \Delta V.$$

Zusammengefaßt ergibt die Massenbilanz

$$[-\mathrm{div}_x w(t,x) + q(t,x)]\Delta t\Delta V = \Delta m^o + \Delta m^q =: \Delta m. \tag{1}$$

Andererseits ist

$$\text{Masse} = \text{Dichte} \times \text{Volumen} \ (\ \text{Energie} = \text{Temperatur} \times \text{Volumen}),$$

oder

$$\begin{aligned} \Delta m &= \tfrac{1}{\Delta t}\,[\rho(t+\Delta t,x) - \rho(t,x)]\Delta t\Delta V \\ &\doteq \rho_t(t,x)\Delta t\Delta V. \end{aligned} \tag{2}$$

Insgesamt ergibt sich aus (1) und (2)

$$\rho_t(t,x) = -\mathrm{div}_x w(t,x) + q(t,x), \tag{3}$$

und damit ein Zusammenhang zwischen zeitlicher Dichteänderung und Teilchenstromdichte. Es fehlt noch ein Zusammenhang zwischen Teilchenstromdichte und Dichte. Dazu zerlege zunächst die Teilchenstromdichte gemäß

$$w(t,x) = w_D(t,x) + w_T(t,x)$$

in einen Diffusionsstromdichteanteil w_D und einen Transportstromdichteanteil w_T. Für Diffusionsströme postuliere

[F1] Diffusionsströme fließen in Richtung des größten Konzentrationsgefälles und sind in ihrer Stärke proportional zur Konzentrationsdifferenz.

Dieses ist das 1. Fick'sche Gesetz (bei Wärmeleitung 1. Fourier'sches Gesetz). In Formeln (großes Konzentrationsgefälle $\sim |\nabla_x \rho(t,x)|$ groß)

$$w_D(t,x) = -K(t,x)\nabla_x \rho(t,x),$$

so daß mit (3)

$$\rho_t(t,x) = \mathrm{div}_x(K(t,x)\nabla_x\rho(t,x)) - \mathrm{div}_x w_T(t,x) + q(t,x)$$

erhalten wird. Die Matrix $K = K(t,x)$ heißt Diffusionskoeffizient und ist abhängig vom Material. Mit

$$w_T(t,x) = v_T(t,x)\rho(t,x) \tag{4}$$

ergibt sich

$$\rho_t(t,x) = \mathrm{div}_x(K(t,x)\nabla_x\rho(t,x)) - v_T(t,x)\nabla_x\rho(t,x) - \rho(t,x)\mathrm{div}_x v_T(t,x) + q(t,x). \tag{5}$$

Zur vollständigen Beschreibung fehlen noch die

- Vorgabe der Dichte zu Beginn des Prozesses,
- Beschreibung des Verhaltens der Dichte auf der Berandung des Behältnisses, in welchem der Prozeß stattfindet.

Bezeichnet Ω das Behältnis, so ist (5) zu ergänzen nach

$$\left\{\begin{array}{lcll} \rho_t(t,x) & = & \mathrm{div}_x(K(t,x)\nabla_x\rho(t,x)) - v_T(t,x)\nabla_x\rho(t,x) - & \\ & & \rho(t,x)\mathrm{div}_x v_T(t,x) + q(t,x) & \text{in } \Omega^T \\ RW(\rho)(t,x) & = & r(t,x) & \text{auf } \partial\Omega^T \\ \rho(0,x) & = & \rho_0(x) & \text{in } \Omega, \end{array}\right. \tag{6}$$

wobei $\Omega^T := (0,T)\times\Omega$, $\partial\Omega^T$ analog. Im Folgenden werden Effekte vernachlässigt, welche durch Eigenbewegung des Stoffes induziert werden.

1.2 Spezialfälle und Modellierung des Randverhaltens

Wärmeleitungs- oder Diffusionsgleichung
Dabei ist

$$K(t,x) \equiv K \equiv 1, v_T \equiv 0, q \equiv 0.$$

Es ergibt sich

$$\text{(7)} \qquad \begin{cases} \rho_t(t,x) & = & \Delta_x \rho(t,x) & \text{in } \Omega^T \\ RW(\rho)(t,x) & = & r(t,x) & \text{auf } \partial\Omega^T \\ \rho(0,x) & = & \rho_0(x) & \text{in } \Omega. \end{cases}$$

Temperatur- oder Konzentrationsvorgabe auf dem Rand vorgeschrieben:

$$\text{(8)} \qquad RW(\rho)(t,x) = \rho(t,x) = r_1(t,x), \quad (t,x) \in \partial\Omega^T.$$

Temperatur- bzw. Konzentrationsverlauf durch die Berandung ist vorgeschrieben:

$$\text{(9)} \qquad RW(\rho)(T,x) = \partial_\eta \rho(t,x) = r_2(t,x), \quad (t,x) \in \partial\Omega^T.$$

Temperatur- bzw. Konzentrationsverlauf durch die Berandung ist proportional zur vorhandenen Temperatur/Konzentration:

$$\text{(10)} \qquad RW(\rho)(t,x) = \partial_\eta \rho(t,x) + \alpha\rho(t,x) = r_3(t,x), \quad (t,x) \in \partial\Omega^T.$$

Behältnis verhält sich wie ein schwarzer Strahler gemäß dem Boltzmann'schen Gesetz:

$$\text{(11)} \qquad RW(\rho)(t,x) = \partial_\eta \rho(t,x) + \alpha\rho(t,x)^4 = r_4(t,x), \quad (t,x) \in \partial\Omega^T.$$

Konvektions - Diffusionsgleichung
Dabei ist der Diffusionskoeffizient gegeben durch

$$K(t,x) \equiv \varepsilon > 0 \; v_T(t,x) = v_T \; (\Rightarrow \operatorname{div}_x v_T = 0) \; , \; q \equiv 0.$$

Es ergibt sich

$$\text{(12)} \qquad \begin{cases} \rho_t(t,x) & = & \epsilon\Delta_x \rho(t,x) - v_T \nabla_x \rho(t,x) & \text{in } (\Omega)^T \\ RW(\rho)(t,x) & = & r(t,x) & \text{auf } \partial\Omega^T \\ \rho(0,x) & = & \rho_0(x) & \text{in } \Omega \end{cases}$$

mit Randwerten wie in (8) bis (11).
Diese Gleichung trägt dem Umstand Rechnung, daß sich etwa bei einem chemischen Prozeß Diffusion und Transport auf verschiedenen Skalen abspielen ($0 < \epsilon \ll |v_T|$), d.h., die Diffusionsgeschwindigkeit ist wesentlich geringer als die Strömungsgeschwindigkeit des Mediums. Diesem Umstand muß bei der numerischen Diskretisierung Rechnung getragen werden.

Konvektion - Diffusion mit Advektion
Hier ist wie oben

$$K(t,x) \equiv \epsilon > 0 \; , \; v_T(t,x) = v_T,$$

aber

$$q(t,x) = q(\rho(t,x)) = a(t,x)\rho(t,x) \; , \; a(t,x) \equiv a$$

d.h., die Quelldichte ist proportional zur vorhandenen Konzentration. Es ergibt sich

$$\text{(13)} \qquad \begin{cases} \rho_t(t,x) & = & \overbrace{\epsilon\Delta_x \rho(t,x)}^{\text{Diffusion}} - \overbrace{v_T \nabla_x \rho(t,x)}^{\text{Konvektion}} + \overbrace{a\rho(t,x)}^{\text{Advektion}} & \text{in } (\Omega)^T \\ RW(\rho)(t,x) & = & r(t,x) & \text{auf } \partial\Omega^T \\ \rho(0,x) & = & \rho_0(x) & \text{in } \Omega. \end{cases}$$

Beim gestörten Gelfand Problem (Zündung in einem Festkörperbrennstoff) ist, siehe auch Kapitel 5.

$$K(t,x) \equiv K \ , \ v_T \equiv 0$$

und

$$q(t,x) = q(\rho(t,x)) = \delta(t,x)e^{\rho(t,x)} \ , \ \delta(t,x) = \delta > 0.$$

d.h., die Quelldichte hängt von der Konzentration ab und die von der Quelle erzeugte Konzentration ist proportional zur Änderung der von der Quelle erzeugten Konzentration (Gesetz von Arrhenius). In Formeln

$$\frac{\partial}{\partial \rho} q(\rho) \approx q(\rho).$$

Damit ergibt sich

$$\left\{ \begin{array}{lcll} \rho_t(t,x) & = & K\Delta_x \rho(t,x) + \delta e^{\rho(t,x)} & \text{in } (\Omega)^T \\ RW(\rho)(t,x) & = & r(t,x) & \text{auf } \partial\Omega^T \\ \rho(0,x) & = & \rho_0(x) & \text{in } \Omega. \end{array} \right. \tag{14}$$

Hierbei handelt es sich um ein stark nichtlineares Problem, das in Kapitel 5 wieder aufgegriffen wird. Z.B. kann mit diesem Modell ansatzweise der chemische Ablauf erklärt werden, der zur Challenger-Katastrophe führte.

Transportgleichung

Hier ist

$$K(t,x) \equiv 0, \quad \mathrm{div}_x v_T(t,x) = 0.$$

Damit ergibt sich

$$\left\{ \begin{array}{lcll} \rho_t(t,x) & = & -v_T \nabla_x \rho(t,x) + q(t,x) & \text{in } (\Omega)^T \\ RW(\rho)(t,x) & = & r(t,x) & \text{auf } \partial\Omega^T \\ \rho(0,x) & = & \rho_0(x) & \text{in } \Omega. \end{array} \right. \tag{15}$$

Im Allgemeinen hat es keinen Sinn, bei dem Transportproblem (15) Randwerte auf den ganzen Rand des Gebietes vorzuschreiben, denn betrachte

$$\rho_t(t,x) + v\rho_x(t,x) = 0, \quad \rho(0,x) = \rho_0(x).$$

Die eindeutige Lösung dieser Gleichung ist gegeben durch

$$\rho(t,x) = \rho_0(x - vt).$$

Sei jetzt $\Omega := (0,1)$ und

$$\rho(t,0) = 0, \quad \rho(t,1) = 1.$$

Dann ist notwendigerweise

$$\begin{array}{lcll} \rho_0(-vt) & = & 0 & \forall\ t \geq 0, \\ \rho_0(1-vt) & = & 1 & \forall\ t \geq 0, \end{array}$$

woraus zur Zeit $t = \frac{1}{v}$ der Widerspruch

$$1 = \rho_0(1-vt)_{t=\frac{1}{v}} = \rho_0(0) = \rho_0(-vt)_{t=0} = 0$$

folgt. Zudem müßte die Anfangsbedingung zu den Randwerten kompatibel sein.

Jetzt exemplarisch zur numerischen Behandlung von (7) mit Randwerten (8). Dazu nehmen wir an, daß $\Omega \subset \mathbb{R}$ ein eindimensionales Gebilde ist (wir untersuchen also den Schnitt eines Schnittes durch ein dreidimensionales Gebiet und nehmen an, daß die Gleichung (7) auch für solche Schnitte richtig ist). Ohne Einschränkung sei $\Omega = (0,1)$. Der Zeithorizont sei T, so daß $\Omega^T = (0,T) \times (0,1)$. Wir wollen Gleichung (7) numerisch zugänglich machen. Dazu sei

$$t^k = k\Delta t, \ (k = 0, \ldots, n_t), \ \Delta t = \frac{T}{n_t} \quad x_i = i\Delta x, \ (i = 0, \ldots, n_x + 1), \ \Delta x = \frac{1}{n_x + 1}$$

wodurch auf $(0,T)\times(0,1)$ ein Gitter mit Zeitschrittweite Δt und Ortsauflösung Δx definiert wird. Ferner bezeichne ρ_i^n eine Approximation der Dichte ρ zur Zeit $t=t^n$ am Ort $x=x_i$, d.m.

$$\rho_i^n \approx \rho(t^n, x_i).$$

Damit ergibt sich aus (7) (Ableitungen werden durch entsprechende Differenzenquotienten ersetzt und die dabei aufretenden Funktionswerte durch Approximationen ρ_i^n) etwa

$$\rho_i^{n+1} - \rho_i^n = \frac{\Delta t}{\Delta x^2}\left(\rho_{i-1}^n - 2\rho_i^n + \rho_{i+1}^n\right), \quad \rho_i^0 := \rho_0(x_i)\ (i=1,\ldots,n_x). \tag{16}$$

Dabei ist noch zu berücksichtigen, daß $\rho_j^n = r(t^n, x_j)$ $(j=0, n_x+1; n\geq 0)$. Damit können die Werte ρ_i^{n+1} sukzessive mit Hilfe der ρ_i^n berechnet werden. Dieses Vorgehen wird *explizit* (bzgl. der Zeit) genannt. Diese Methode ist *stabil* (was auch immer das hier heißen mag) nur für hinreichend kleine Quotienten $\frac{\Delta t}{\Delta x^2}$. Bessere *Stabilitätseigenschaften* besitzen sogenannte *implizite* Verfahren. Eine Schar solcher ist etwa gegeben durch

$$\rho_i^{n+1} - \rho_i^n = \frac{\Delta t}{\Delta x^2}\left\{(1-\Theta)\left(\rho_{i-1}^n - 2\rho_i^n + \rho_{i+1}^n\right) + \Theta\left(\rho_{i-1}^{n+1} - 2\rho_i^{n+1} + \rho_{i+1}^{n+1}\right)\right\}, \quad \rho_i^0 := \rho_0(x_i)\ (i=1,\ldots,n_x) \tag{17}$$

wobei $\Theta\in[0,1]$. Für $\Theta=0$ wird wieder Verfahren (16) erhalten. Für $\Theta\in(0,1]$ ist zu jedem Zeitpunkt t^{n+1} ein lineares Gleichungssystem der Form

$$\left(I + \Theta\frac{\Delta t}{\Delta x^2}A\right)\rho^{n+1} = \left(I - (1-\Theta)\frac{\Delta t}{\Delta x^2}A\right)\rho^n + r^n \tag{18}$$

zu lösen, wobei $\rho^n := [\rho_1^n, \ldots, \rho_{n_x}^n]^t$.

Aufgabe 1.1. Füllen Sie (18) mit Leben, d.m. geben Sie die Matrix A und den Vektor r^n an.

1.3 Modellierung von Verkehrsfluß

Das Verkehrsaufkommen auf einer einspurigen Landstraße kann mit den Überlegungen des Kapitels 1.1 modelliert werden, und zwar wie folgt.
Bezeichnen

$\rho(t,x)$ Fahrzeugdichte (Anzahl Autos/Kilometer Straße)

$q(t,x)$ Fahrzeugstrom (Anzahl Autos/Stunde)

$[a,b]$ Abschnitt Straße (Strecke von Punkt a bis Punkt b)

t_1, t_2 $\Delta\, t = t_2 - t_1$ Zeitraum,

so gilt

$$\#\text{ Fahrzeuge in } [a,b] \text{ zur Zeit } t = \int_a^b \rho(t,x)dx.$$

Sind nun $\int_a^b \rho(t_1,x)dx$ und $\int_a^b \rho(t_2,x)dx$ verschieden, so müssen entweder Autos bei a den Streckenabschnitt erreichen oder bei b verlassen. Weil die # Fahrzeuge, die bei a die Strecke [a,b] im Zeitraum zwischen t_1 und t_2 erreichen $= \int_{t_1}^{t_2} q(t,a)dt$ ist und die # Fahrzeuge, die bei b die Strecke [a,b] im Zeitraum zwischen t_1 und t_2 verlassen $= -\int_{t_1}^{t_2} q(t,b)dt$, kann dieser Sachverhalt in Formeln wie folgt ausgedrückt werden:

$$\underbrace{\int_a^b \rho(t_2,x) - \rho(t_1,x)dx}_{\substack{\text{Differenz \# Autos zu den}\\ \text{verschiedenen Zeitpunkten bei } x}} = \underbrace{\int_{t_1}^{t_2} q(t,a) - q(t,b)dt}_{\substack{\text{Autos rein bei } a \text{ minus}\\ \text{Autos raus bei } b}}$$

Diese Relation liefert

$$\underbrace{\frac{1}{b-a}\int_a^b \frac{\rho(t_2,x)-\rho(t_1,x)}{t_2-t_1}\,dx}_{\longrightarrow\ \rho_t(t_1,a)\ (t_2\to t_1,\,b\to a)} \quad = \quad \underbrace{\frac{1}{t_2-t_1}\int_{t_1}^{t_2} \frac{q(t,a)-q(t,b)}{b-a}\,dt}_{\longrightarrow\ -q_x(t_1,a)\ (t_2\to t_1,\,b\to a)}.$$

Die gewünschte Erhaltungsgleichung ist in diesem Fall

$$\rho_t(t,x) \;=\; -q_x(t,x). \tag{19}$$

An dieser Stelle muß wieder ein phänomenologischer Ansatz bemüht werden, um q mit ρ in Beziehung zu setzen. Als Ansatz dazu diene

$$q(t,x) \;=\; h(\rho(t,x))$$

und die folgende Beobachtung.

> "Der Fahrzeugstrom q erfährt bei zunehmender Fahrzeugdichte ρ bei der Dichte ρ_m eine Sättigung h_0 und nimmt für größere Dichten ab."

Als einfachen Ansatz für h wähle

$$h(\rho) = -\frac{1}{2}(\rho-\rho_m)^2 + h_0.$$

Damit ergibt sich in (19) unter Verwendung von AW'en

$$\begin{cases} \rho_t(t,x) - (\rho(t,x)-\rho_m)\rho_x(t,x) &= 0 & \text{in } (\Omega)^T \\ \rho(0,x) &= \rho_0(x) & \text{in } \Omega^T, \end{cases}$$

so daß sich schließlich mit der Transformation $\rho = -(\rho-\rho_m)$ die Burgers Gleichung

$$\begin{cases} \rho_t(t,x) + \rho(t,x)\rho_x(t,x) &= 0 & \text{in } (\Omega)^T \\ \rho(0,x) &= \rho_m - \rho_0(x) & \text{in } \Omega, \end{cases} \tag{20}$$

ergibt.
Allgemeine Modelle für die mathematische Beschreibung von Verkehrsfluß führen auf kinetische Gleichungen der Form

$$f_t(t,x,v) + v(t,x)f_x(t,x,v) = C(f)(t,x,v), \tag{21}$$

wobei f die Anzahl der Fahrzeuge bei x zur Zeit t mit Geschwindigkeit v bezeichnet. Ist wieder $\rho(t,x)$ die Fahrzeugdichte, so gilt

$$\rho(t,x) = \int_0^{v_{max}} f(t,x,v)dv.$$

C bezeichnet dabei den Interaktionsparameter, von welchem in der Modellierung angenommen wird, daß er sich aus einem Anteil zusammensetzt, welcher Bremsvorgänge beschreibt, und aus einem Anteil, der Beschleunigungsvorgänge modelliert, i. e.

$$C(f) = B(f) + A(f). \tag{22}$$

Näheres hierzu ist zu finden in [18], download unter
http://www.mathematik.tu-darmstadt.de/~klar.
Weitere Beispiele für die Modellierung von Transportproblemen und allgemeinen hyperbolischer Erhaltungsgleichungen werden von Kröner in [9] angegeben.
Jetzt haben wir soviel modelliert und wissen noch nicht, wie die Lösungen der bis hierher entwickelten Modelle letztendlich aussehen. Analytische Lösungsdarstellungen wird es i.d.R. nicht geben (von wenigen, meist bedeutungslosen Ausnahmesituationen einmal abgesehen). Hier kommt jetzt die Numerik ins Spiel. Mit deren Hilfsmittel werden wir diskrete Modelle herleiten, welche die kontinuierlichen Modelle (hoffentlich) näherungsweise beschreiben.

Wir betrachten die skalare Erhaltungsgleichung

$$\left\{\begin{array}{lcll} \rho_t(t,x) + h(\rho(t,x))_x & = & 0 & \text{in} \quad (0,\infty)\times\mathbb{R} \\ \rho(0,x) & = & \rho_0(x) & \text{in } \mathbb{R} \end{array}\right. \tag{23}$$

setzen uns deren numerische Diskretisierung zum Ziel. Dabei sollten wir von vornherein folgendes Paradigma so gut es geht berücksichtigen.

Merkregel 1.2. Ein Verfahren zur numerischen Simulation eines physikalischen Prozesses sollte idealerweise alle physikalischen Eigenschaften des Systems (so gut es geht) reproduzieren. Ein solches Verfahren heißt dann **konservativ**.

Konservativ ist in diesem Zusammenhang alles andere als negativ behaftet. Und sehr wichtig ist bei dieser Sichtweise, daß *Physik* hier die durch das mathematische Modell beschriebene Wirklichkeit meint und nicht etwa die tatsächliche Welt. Verfahren können also nur in den von unserem mathematischen Modell gesteckten Grenzen konservativ sein. Was das dann mit der Wirklichkeit zu tun hat ist Sache der Modellierung bzw. deren Güte.
Zur Herleitung konservativer numerischer Schemata für (23) wird die Erhaltungsgleichung in integraler Form geschrieben. Dazu sei

$$V := [t_1, t_2] \ \times \ [a, b]$$

ein Testvolumen. Die Erhaltungsgleichung in integraler Form ergibt sich durch die Integration der Erhaltungsgleichung in (1.20) über V, i.e.:

$$\int_{t_1}^{t_2}\int_{a}^{b} \rho_t(t,x)dxdt + \int_{t_1}^{t_2}\int_{a}^{b} h(\rho(t,x))_x dxdt = 0,$$

(vergl. Kapitel 1, Modellierung) was widerum äquivalent zu

$$\int_{a}^{b} \rho(t_2,x) - \rho(t_1,x)dx \ + \ \int_{t_1}^{t_2} h(\rho(t,b)) \ - \ h(\rho(t,a))dt = 0 \tag{24}$$

ist. Um diese Identität für die Konstruktion von numerischen Verfahren zu nutzen, werden die auftretenden Integranden numerisch approximiert. Dazu sei

$$t^n = n\Delta t, \quad x_i = i\Delta x, \quad n \in \mathbb{N}_0, \quad i \in \mathbb{Z}$$

wodurch auf $(0,\infty)\times\mathbb{R}$ ein Gitter mit Zeitschrittweite Δt und Ortsauflösung Δx definiert wird. Für

$$V = \left[t^n, t^{n+1}\right] \ \times \ \left[x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}\right],$$

wobei

$$x_{i-\frac{1}{2}} := (i - \frac{1}{2})\Delta x, \quad x_{i+\frac{1}{2}} := (i + \frac{1}{2})\Delta x,$$

ergibt sich in (24)

$$\underbrace{\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \rho(t^{n+1},x) - \rho(t^n,x)dx}_{I_1} \ + \ \underbrace{\int_{t^n}^{t^{n+1}} h(\rho(t,x_{i+\frac{1}{2}})) - h(\rho(t,x_{i-\frac{1}{2}}))dt}_{I_2} \ = 0. \tag{25}$$

Zu Beginn dieses Kapitels haben wir gesehen, daß I_1 die Massenänderung in $[x_{i-\frac{1}{2}}, x_{i+\frac{1}{2}}]$ während des Zeitraums Δt beschreibt, $-I_2$ die Masse, welche während Δt über $x_{i-\frac{1}{2}}$ und $x_{i+\frac{1}{2}}$ ein- bzw. austritt. Mit anderen Worten: (25) garantiert, daß die Masse erhalten bleibt. Um (25) numerisch zugänglich

zu machen, bezeichnen ρ_i^n eine Approximation von $\frac{1}{\Delta x}\int_{x_{i-\frac{1}{2}}}^{x_{i+\frac{1}{2}}} \rho(t^n,x)dx$ und g_i^n eine Approximation von $\frac{1}{\Delta t}\int_{t^n}^{t^{n+1}} h\left(\rho(t,x_i)\right)dt$.
Mit den obigen Setzungen wird unter Berücksichtigung von (25) das Schema

$$\rho_i^{n+1} - \rho_i^n = -\frac{\Delta t}{\Delta x}(g_{i+\frac{1}{2}}^n - g_{i-\frac{1}{2}}^n). \tag{26}$$

zur numerischen Approximation einer Lösung von (23) erhalten. Oder allgemeiner

$$\rho_i^{n+1} - \rho_i^n = -\frac{\Delta t}{\Delta x}\left\{\Theta(g_{i+\frac{1}{2}}^{n+1} - g_{i-\frac{1}{2}}^{n+1}) + (1-\Theta)(g_{i+\frac{1}{2}}^n - g_{i-\frac{1}{2}}^n)\right\}, \tag{27}$$

wobei $0 \leq \Theta \leq 1$. Für $\Theta = 0$ heißt die Vorschrift explizit, für $0 < \Theta < 1$ semi-implizit und für $\Theta = 1$ implizit.
Daß das Schema (27) tatsächlich konservativ ist, folgt aus

$$\sum_{i\in\mathbb{Z}} \left(g_{i+\frac{1}{2}}^n - g_{i-\frac{1}{2}}^n\right) = 0 \qquad \text{(falls Summe existent)},$$

was widerum

$$\sum_{i\epsilon\mathbb{Z}} \rho_i^{n+1} = \sum_{i\epsilon\mathbb{Z}} \rho_i^n \quad \text{für alle } n \geq 0$$

impliziert, Massenerhaltung der diskreten Größe also.
Der Ausdruck

$$\delta h_i^n := \frac{1}{\Delta x}\left(g_{i+\frac{1}{2}}^n - g_{i-\frac{1}{2}}^n\right) \text{ (Numerischer Fluß)} \tag{28}$$

kann als Approximation von

$$h(\rho(t^n,x_i))_x,$$

angesehen werden.
In (27) ist je nach Wahl von Θ ein nichtlineares Gleichungssystem zur Bestimmung von ρ_i^{n+1} $i \in \mathbb{N}$ zu lösen, wobei sukzessive in n fortgeschritten wird. Im Fall $n = 0$ ist der Anfangswert ρ_0 gegeben und es kann für $n = 1$ numerisch gelöst werden. Dann wird n um eins erhöht und so fortgefahren.

Aufgabe 1.3. (Modellierung des Verkehrsflusses auf einer Autobahn)
Es bezeichnen $u = u(t,x)$ die Anzahl der Fahrzeuge pro Kilometer (Fahrzeugdichte) und $q = q(t,x)$ die Anzahl der Fahrzeuge pro Stunde (Fahrzeugfluß) auf einer Autobahn. Dabei ist x die eindimensionale Ortskoordinate und t die Zeit. Ferner seien $[a,b]$ ein endlicher Autobahnabschnitt und $t_1 < t_2$ zwei Zeitpunkte.

1. Wie groß ist die Anzahl der Fahrzeuge in dem Autobahnabschnitt $[a,b]$ zu den Zeitpunkten t_1 und t_2 ?

2. Wie groß ist die Anzahl der Fahrzeuge, die während der Zeitspanne $\delta t = t_2 - t_1$ über a in den Autobahnabschnitt ein- bzw. über b aus dem Autobahnabschnitt ausfahren?

3. Drücken Sie das folgende Erhaltungsgesetz

   Die Anzahl der Fahrzeuge in $[a,b]$ zur Zeit t_2 minus der Anzahl der Fahrzeuge in $[a,b]$ zur Zeit t_1 entspricht der Differenz der in a ein- und in b ausfahrenden Fahrzeuge

   in mathematischer Schreibweise aus (mit Herleitung).

Aufgabe 1.4. (Modellierung und Konsequenzen aus dem Modell)
Mit u und q aus Aufgabe 1.3 sei jetzt q eine Funktion von u, d.h. $q(t,x) = f(u(t,x))$. Die Erhaltungsgleichung aus Aufgabe 1.3 kann damit geschrieben werden als

$$(P_1) \quad \begin{cases} \frac{\partial}{\partial t}u(t,x) &= -\frac{\partial}{\partial x}f(u(t,x)), \qquad x \in [a,b], \quad t \in (t_1,\infty), \\ u(t_1,x) &= v(x), \end{cases}$$

wobei v die Fahrzeugdichte zur Zeit $t = t_1$ bezeichne.

1. Interpretieren Sie qualitativ den Ansatz

   $$f(u) = -(u-u_m)^2 + f_0,$$

   wobei $f_0, u_m > 0$ konstant geeignet gewählt sind.

2. Sei u eine Lösung von (P_1) mit $a = -2, b = 2, f(u) = \frac{1}{2}u^2$ und ∞-oft differenzierbaren Anfangswerten

   $$v(x) = \begin{cases} 1, & x \in [-2,-1) \\ 0, & x \in [1,2] \\ g(x) & x \in [-1,1). \end{cases}$$

   Zeigen Sie, dass es keine Lösung $u \in C^0((t_1,\infty) \times [a,b])$ des Problems (P_1) gibt.
   Hinweis: Zeigen Sie zuerst ganz allgemein, dass u entlang der Kurve $(t,\gamma(t))$ konstant ist, wobei $\gamma(t)$ eine Lösung der Anfangswertaufgabe

   $$\dot{\gamma}(t) = \frac{\partial}{\partial u}f(u(t,\gamma(t))), \qquad \gamma(t_1) = s \in [a,b]$$

   ist.
   Danach sollten Sie zeigen, daß diese Kurven $(t,\gamma(t))$ Geraden in der t-x-Ebene sind. Wenn Sie nun diese beiden Aussagen auf das Beispiel anwenden, erhalten Sie die gewünschte Aussage.

3. Wenden Sie das Ergebnis aus 2. auf das Problem (P_1) mit f aus 1. an und interpretieren Sie das Ergebnis qualitativ.

Aufgabe 1.5. Gegeben sei die Transportgleichung (vergl. (23))

$$(P_1) \quad \begin{cases} \frac{\partial}{\partial t}u(x,t) &= -\frac{\partial}{\partial x}f(u(x,t)), \qquad x \in [0,6], \quad t \in (0,6], \\ u(x,0) &= v(x). \end{cases}$$

In dieser Differentialgleichung soll die Ortsableitung mit dem rückwärtsgenommenen Differenzenquotienten erster Ordnung zur Schrittweite h approximiert werden. Das führt dann in jedem Ortspunkt x_i auf eine gewöhnliche Differentialgleichung in t. Diese können dann mit dem vorwärtsgenommenen Eulerverfahren numerisch gelöst werden. Diese Vorgehensweise heißt **Linienmethode**.
Ihr Programm soll für $f(u) = \frac{1}{2}u^2$ die Transportgleichung mittels der Linienmethode lösen. Die Anfangs- und Randwerte seien dabei:

1. $$v(x) = \begin{cases} -0.2x + 1.0, & x \le 5 \\ 0.0, & x > 5 \end{cases}$$

   mit den Randwerten $u(0,t) = 1.0,\ u(6,t) = 0.0$.

2. $$v(x) = \begin{cases} -0.2x + 2.0, & x \le 5 \\ 1.0, & x > 5 \end{cases}$$

   mit den Randwerten $u(0,t) = 2.0,\ u(6,t) = 1.0$.

Für die Ortsdiskretisierung wählen Sie bitte $h = 0.05$ und als Zeitschrittweite $k = 2.0 \cdot h^2$.
Nun zur Ausgabe, die das Programm liefern soll:
Geben Sie unter Verwendung des *plot* Befehls in MATLAB die Anfangsverteilung $u(x,0)$ und die Endverteilung $u(x,6)$ aus. Danach erzeugen Sie bitte unter MATLAB mit dem *movie*-Befehl einen Film, der die zeitliche Entwicklung der Lösung zeigt. Stellen Sie dabei bitte nur jede **zehnte** Zeitschicht dar.

1.4 Portfolio Modellierung und Optimierung

Jeder Investor sollte sich darüber im Klaren sein, daß Gewinnvergrösserung einhergeht mit steigendem Investitionsrisiko. Stellen wir uns folgenden Situation vor;

> Es liegen n Investionsmöglichkeiten vor mit Renditen r_i $(i = 1, \dots, n)$.
>
> Die Renditen sind a-priori nicht bekannt und werden als normalverteilte Zufallsvariablen ($N(\mu, \sigma^2)$-verteilt) angenommen. Die Erwartungswerte seien $\mu_i = E[r_i]$ und die Varianzen $V(r_i) = \sigma_i^2 = E[(r_i - \mu_i)^2]$.
>
> Die Menge an flüssigen Finanzmitteln sei gleich 1.
>
> Wir konstruieren unser Portfolio, indem wir x_i Anteile unserer flüssigen Mittel in Investionsmöglichkeit i investieren. Dabei gilt $\sum_{i=1}^{n} x_i = 1$.

Wie wird jetzt die Rendite modelliert? Es gilt natürlich

$$R = \sum_{i=1}^{n} x_i r_i.$$

Jetzt sollten wir noch ein Maß dafür aufstellen, wie wir ein Portfolio bewerten. Dazu betrachten wir den Erwartungswert der Rendite

$$E[R] = E\left[\sum_{i=1}^{n} x_i r_i\right] = \sum_{i=1}^{n} x_i E[r_i] = x^t \mu,$$

wobei wir die Anteile x_i und die Erwartungswerte μ_i in Vektoren $x = (x_1, \dots, x_n)^t$ und $\mu = (\mu_1, \dots, \mu_n)^t$ zusammengefaßt haben. Und schließlich noch die Varianz der Rendite,

$$E[(R - E[R])^2] = \sum_{i=1}^{n} \sum_{j=1}^{n} x_i x_j \sigma_i \sigma_j \rho_{ij} = x^t G x,$$

wobei $G = (g_{ij})_{i,j=1,\dots,n}$, $g_{ij} := \sigma_i \sigma_j \rho_{ij}$ und

$$\rho_{ij} := \frac{E[(r_i - \mu_i)(r_j - \mu_j)]}{\sigma_i \sigma_j}, \quad i, j = 1, \dots, n,$$

die Kovarianz zwischen den Anlagen i und j bezeichnet. Sie ist ein Maß dafür, inwieweit sich die Renditen der Anlagen i und j in die selbe Richtung entwickeln.
Wie soll unser Portfolio jetzt aussehen bzw. welches sind die Kriterien, nach denen wir unser Portfolio gestalten wollen? Der gesunde Menschenverstand schlägt uns vor, möglichst viel Gewinn bei kleinen Risiken zu erwirtschaften (denn wir sind ja keine Zocker!), in den oben eingeführten Termen heißt das
Optimierungsziel: Maximale Rendite bei möglichst kleiner Varianz des Portfolios.
Wir landen somit bei der Optimierungsaufgabe

$$\max_{x \in \mathbb{R}^n} F(x) := x^t \mu - \kappa x^t G x \quad \text{bei } \sum_{i=1}^{n} x_i = 1,\ x \geq 0. \tag{29}$$

Dabei variiert der Parameter $\kappa \in [0, \infty)$ und ist offensichtlich ein Maß dafür, wie wichtig uns geringes Risiko bei der Portfoliogestaltung letztendlich ist (Großes κ meint kleines Risiko). Dieses Modell der Portfoliogestaltung geht zurück auf Markowitz [10].
Bis hierhin haben wir verschwiegen, wie wir uns die Daten μ_i und σ_i und ρ_{ij} für das Optimierungsproblem (29) verschaffen. Das ist wiederum ein Optimierungsproblem für sich. Bei klassischen Börsenwerten könnten z.B. die Daten aus der Vergangenheit genommen werden (die vergangenen 5 Jahre etwa). Das geht allerdings nicht bei Startups. Am besten ist wohl eine Mischung aus Erfahrungen bzw. Kenntnis der Materie und Daten aus der Vergangenheit, gepaart mit einem guten Schuß Gottvertrauen oder so.
Nun nehmen wir an, daß der Vektor μ, die Matrix G und der Parameter κ in (29) bekannt sind. Wir haben uns folgende Fragen zu stellen.

Besitzt das Optimierungsproblem (29) eine Lösung?

Sind mehrere Lösungen möglich?

Ganz wichtig: Wie können Lösungen numerisch berechnet werden?

Zur Beantwortung dieser Frage dient folgende

Aufgabe 1.6. Zeigen Sie, daß

a.) die Matrix $G \in \mathrm{MAT}(\mathbb{R}^n)$ positiv semi-definit und symmetrisch ist, d.m. $x^t G x \geq 0$ für alle $x \in \mathbb{R}^n$ und $g_{ij} = g_{ji}$ für alle $i, j = 1, \ldots, n$.

b.) Nehmen Sie nun an, daß G positiv definit ist. Dann hat die Optimierungsaufgabe (29) genau eine Lösung $x^* \in \mathbb{R}^n$. Unter welchen Umständen ist G positiv definit? Beschreiben Sie hinreichende Bedingungen für diese Eigenschaft von G in Termen der obigen Modellierung.

c.) Schreiben Sie (29) um in ein äquivalentes Minimierungsproblem

$$\min_{x \in \mathbb{R}^n} J(x) \quad \text{bei } \sum_{i=1}^n x_i = 1,\ x \geq 0.$$

d.) Zeigen Sie, daß

– $\nabla F(x) = \mu - 2\kappa G x$ richtig ist und

-* die bei positiv definitem G eindeutig bestimmte Lösung x^* von Problem (29) charakterisiert ist durch das Gleichungssystem

$$\begin{aligned} 2\kappa G x^* - \mu + p^* \underline{1} + \xi^* &= 0 \\ \sum_{i=1}^n x_i &= 1 \\ \xi^* &= \min\{0, \xi^* - c x^*\} \text{ komponentenweise }, c > 0 \text{ beliebig.} \end{aligned}$$

Dabei bezeichnen $p^* \in \mathbb{R}$ und $\xi^* \in \mathbb{R}^n$ die sogenannten *Lagrange Multiplikatoren* zu den Nebenbedingungen $\sum_{i=1}^n x_i^* = 1$ und $x^* \geq 0$ komponentenweise.

Wieder wird deutlich, daß Antworten auf unsere Fragen in der Lösungsmenge von Gleichungssystemen verborgen sind. Wollen wir uns also der numerischen Behandlung von (linearen) Gleichungssystemen näher widmen.

2 Numerische Behandlung von Gleichungssystemen

In diesem Kapitel werden wir uns mit der numerischen Lösung von Gleichungssystemen beschäftigen. Beispiele für diese Aufgabenstellung haben wir in Kapitel 1 kennengelernt. Der wesentlich Baustein zur numerischen Integration von (17) besteht in jedem zeitlichen Integrationsschritt aus der Lösung eines linearen Gleichungssystems zur Berechnung von ρ^{n+1}. Ähnlich liegt die Sache bei der numerischen Berechnung von ρ^{n+1} in (27), jedoch ist hier ein nichtlineares Gleichungssystem zu lösen (die Funktion h bei in der Definition der Flüsse g_i^n ist i.d.R. nichtlinear). Schließlich haben wir auch bei der Portfolio Optimierung gesehen, daß die numerische Berechnung eines optimalen Investments auf die Lösung des nichtlinearen Gleichungssystems aus Aufgabenteil d.) von Aufgabe 1.6 führt.

2.1 Lineare Gleichungssysteme

In diesem Kapitel untersuchen wir numerische Lösungsverfahren zur Bestimmung von Lösungen $x \in \mathbb{R}^n$ (der Lösungsmenge) des linearen Gleichungssystem

$$Ax = b. \tag{30}$$

Dabei bezeichnen

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} \in \mathbb{R}^{n,n}, b = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix} \in \mathbb{R}^n \text{ und } x = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} \in \mathbb{R}^n$$

die gegebene Koeffizientenmatrix, den Vektor der rechten Seite und die gesuchte Lösung des Gleichungssystems.
Wir verlangen:

Voraussetzung 2.1. System (30) besitzt genau eine Lösung $x \in \mathbb{R}^n$.

Diese Annahme ist sinnvoll. Liefert unsere Modellierung eine Aufgabenstellung der Form (30) und ist diese nicht eindeutig lösbar, sollten wir uns nochmals die Modellierung vornehmen und dort nach den Ursachen für diesen mißlichen Zustand forschen.
Es gibt zwei prinzipielle Möglichkeiten, (30) numerisch zu lösen;

- **direkt**, d.m. die Lösung x wird exakt im Rahmen der Maschinengenauigkeit durch eine einzige Befragung eines Orakels (Aufruf einer Prozedur) bereitgestellt (besser als maschinengenau wird es leider nicht gehen), und
- **iterativ**, d.m. die Lösung x (bzw. eine Näherung dieser) wird durch sukzessive Befragung von Orakeln (sukzessives Aufrufen von Prozeduren) bereitgestellt.

Wir werden für beide Varianten Orakel (Prozeduren) angeben. An dieser Stelle sei bereits bemerkt, daß die Anzahl der zur exakten Berechnung von x notwendigen Orakelaufrufe im Fall der iterativen Lösung nicht endlich zu sein braucht. Auf dem Rechner werden wir daher mit iterativen Verfahren nur Näherungslösungen der (maschinengenau) exakten Lösung x berechnen können (denn in dem uns bekannten Universum ist nun einmal alles endlich...).

2.1.1 Direkte Verfahren

Gauß hat bereits erkannt, daß lineare Gleichungssystem mit Matrizen der Form

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \dots & a_{nn} \end{bmatrix} = LR = \begin{bmatrix} 1 & 0 & \dots & 0 \\ l_{21} & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ l_{n1} & l_{n2} & \dots & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & \dots & r_{1n} \\ 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \dots & 0 & r_{nn} \end{bmatrix}$$

sehr einfach aufzulösen sind. Der Algorithmus schreibt sich dann wie folgt;

1. Löse $Lh = b$ und erhalte h,
2. Löse $Rx = h$ und erhalte x.

Dabei wird h mittels Vorwärtseinsetzen erhalten (bestimme h_1, dann h_2 usw. und schließlich h_n). Die gesuchte Lösung x ergibt sich dann mittels Rückwärtseinsetzen (bestimme x_n), danach x_{n-1} und schließlich x_1. Wir haben dabei angenommen, daß alle Diagonalelemente von L den Wert 1 haben. Überlegen Sie sich bitte, daß dies' keine Einschränkung der Allgemeinheit bedeutet und Vorwärts- Rückwärtseinsetzen ein wohldefinierter Prozeß ist, sofern A invertierbar.
Leider existiert eine Zerlegung der o.g. Art nicht für jede invertierbare Matrix A, wie das

Beispiel 2.2.

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

zeigt (Nachweis!). Es gilt aber

Satz 2.3. Zu jeder invertierbaren Matrix $A \in \mathbb{R}^{n,n}$ existiert eine Permutationmatrix $P \in \mathbb{R}^{n,n}$ derart, daß

$$PA = LR \tag{31}$$

gültig ist. Dabei ist L untere Dreiecksmatrix mit $l_{ii} = 1$ für $i = 1, \ldots, n$ und R ist obere Dreiecksmatrix.

Beweis. Wir geben einen Algorithmus an, der Gewünschtes leistet. Schon der erste Schritt enthält alles, was beachtet werden muß. Wir überführen

$$A = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{bmatrix} \rightarrow A^1 = \begin{bmatrix} a^1_{11} & a^1_{12} & \ldots & a^1_{1n} \\ 0 & a^1_{22} & \ldots & a^1_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & a^1_{n2} & \ldots & a^1_{nn} \end{bmatrix}$$

Das erreichen wir dadurch, daß wir den folgenden Algorithmus für $k = n$ auf die Matrix A anwenden.

Algorithmus 2.4. Gauß(K, k)
Vorgelegt sei die Matrix $K = [k_{ij}]^k_{i,j=1} \in \mathbb{R}^{kk}$

1. Bestimme ein Element $k_{r,1} \neq 0$, $r \in \{1, \ldots, k\}$ und gehe zu 2. Falls kein solcher Index r existiert setze Gauß$(K, k) = K$, STOP. K ist singulär.
2. Vertausche die Zeilen r und 1 von K und erhalte so die Matrix $\bar{K} = [\bar{k}_{ij}]^k_{i,j=1}$.
3. Subtrahiere für $i = 2, \ldots, k$ das l_{i1}-fache der ersten von der i-ten Zeile der Matrix $\bar{K}$,
$$\bar{k}_{ij} = \bar{k}_{ij} - l_{i1}\bar{k}_{1j} \ (j = 1, \ldots, k), \text{ wobei } l_{i1} := \frac{\bar{k}_{i1}}{\bar{k}_{11}}.$$
4. Das Resultat ist die Matrix Gauß(K, k).

Sei also $A^1 :=$ Gauß(A, n). Eine genaue Inspektion ergibt, daß wir dabei

- $A \rightarrow \bar{A}$ mit $\bar{A} = PA$ gebildet haben, wobei P die Permutationsmatrix bezeichnet, welche die Zeilen 1 und r vertauscht,
- um anschließend $A^1 = G\bar{A} = GPA$ zu bilden.

Dabei ist

$$G = \begin{bmatrix} 1 & 0 & \ldots & 0 \\ -l_{21} & 1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ -l_{n1} & 0 & \ldots & 1 \end{bmatrix}, \text{ mit } l_{i1} = \frac{\bar{a}_{i1}}{\bar{a}_{11}} \ (i = 2, \ldots, k)$$

eine sogenannte **Frobenius Matrix**. Frobenius Matrizen unterscheiden sich nur in einer Spalte unterhalb der Diagonalen von der Einheitsmatrix. Es gilt

$$G^{-1} = \begin{bmatrix} 1 & 0 & \ldots & 0 \\ l_{21} & 1 & \ldots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ l_{n1} & 0 & \ldots & 1 \end{bmatrix}$$

und ferner $P^{-1} = P$ und Setzen wir jetzt $P^1 := P$ und $L^1_1 := G^{-1}$, so gilt $P^1A = L^1_1A^1$. Sei jetzt $Z^1 := [a^1_{ij}]_{i,j=2,\ldots,n} \in \mathbb{R}^{n-1,n-1}$ der erste Hauptminor von A^1. Wir berechnen

1. $Z^2 :=$ Gauß$(Z^1, n-1)$, wobei die Permutationsmatrix $P \in \mathbb{R}^{n-1,n-1}$ und die Matrix $G \in \mathbb{R}^{n-1,n-1}$ erzeugt werden.

2. Setze

$$P^2 := \begin{bmatrix} 1 & 0 \\ 0 & P \end{bmatrix}, \quad L_2^2 := \begin{bmatrix} 1 & 0 \\ 0 & G^{-1} \end{bmatrix} \text{ und } A^2 := \begin{bmatrix} a_{11}^1 & a_{1j}^1 \\ 0 & Z^2 \end{bmatrix}.$$

Damit gilt wie oben $P^2A^1 = L_2^2A^2$. Ferner gilt $P^1A = L_1^1A^1$. Einsetzen liefert

$$P^2(L_1^1)^{-1}P^1A = P^2(L_1^1)^{-1}P^2P^2P^1A = L_2^2A^2 \Longrightarrow P^2P^1A = \underbrace{P^2L_1^1P^2}_{=:L_2^1}L_2^2A^2.$$

Aufgabe 2.5. Weisen Sie nach, daß $L_2^1 := P^2L_1^1P^2$ wieder eine Frobeniusmatrix ist, die sich nur in zwei Einträgen in der ersten Spalte von L_1^1 unterscheidet.

Das induktive Vorgehen ist jetzt klar. Wir erhalten nach $n-1$ Aufrufen von Gauß und den sich anschließenden Modifikationen

$$PA = LR$$

mit

$$P = P^{n-1}P^{n-2}\dots P^1 \text{ und } L = L_{n-1}^1L_{n-1}^2\dots L_{n-1}^{n-1}.$$

Dabei gilt nach dem k-ten Aufruf $(k = 1, \dots, n-1)$

$$L_k^j = P^kL_{k-1}^jP^k, \quad j = 1, \dots k-1. \tag{32}$$

Damit ist die Aussage bewiesen. □

Wir sind jetzt in der Lage, die Dreieckszerlegung einer Matrix A zu berechnen bzw. festzustellen, ob eine Matrix singulär ist.
Die Bereitstellung der Zerlegung $PA = LR$ ist immer dann sinnvoll, wenn mehre Gleichungssysteme mit verschiedenen rechten Seiten b gelöst werden sollen. Besteht die Aufgabenstellung lediglich in der Lösung eines einzigen Gleichungssystems mit rechter Seite b, ergibt sich aus dem Beweis von Satz 2.3 direkt

Algorithmus 2.6. Gauß-Elimination mit Pivotisierung

Eingabe: Eine reguläre Matrix $A \in \mathbb{R}^{n,n}$ und ein Vektor $b \in \mathbb{R}^n$.

Ausgabe: Ein zu $Ax = b$ äquivalentes lineares Gleichungssystem $Rx = r$ mit oberer Dreiecksmatrix R.

1. $i = n$ und $j = 1$.
2. Ist $a_{ij} = 0$ gehe zu 4.

   Ist $a_{jj} = 0$ vertausche a_{ik} und a_{jk} für $k = 1, \dots, n$ sowie b_i und b_j (Pivotschritt).
3. Setze $l_{ij} = \frac{a_{ij}}{a_{jj}}$ und berechne

   $$a_{ik} := a_{ik} - l_{ij}a_{jk} \text{ für } k = 1, \dots, n \text{ und } b_i := b_i - l_{ij}b_j.$$
4. $i > j + 1$: $i := i - 1$ und gehe zu 2.

   $j < n - 1$: $j := j + 1$, $i := n$ und gehe zu 2.

   STOP

Der Pivotschritt in Algorithmus 2.6 entspricht dem Permutationsschritt 1. in Algorithmus 2.4. Der Vertauschungsschritt kann natürlich auch durchgeführt werden, wenn der Eintrag a_{jj} von Null verschieden ist. Es besteht dann die Hoffnung, durch geschickte Wahl des Pivotelements einen numerisch stabilen Algorithmus zu erhalten. Beliebte Auswahlregeln sind

Bemerkung 2.7. Pivot Auswahlregeln
Wähle als Pivotelement (d.h. wir führen etwa in Algorithmus 2.6 in Schritt 2. zunächst aus, bzw. bewerkstelligen die Zeilenvertauschung in Algorithmus 2.4 gemäß)

R1 a_{rj} mit $r = \text{argmax}_{i \geq j} |a_{ij}|$ (Pivot Element ist das betragsmäßig größte Element der Spalte)

R2 $s_i := 1/\sum_{k=1}^{n} |a_{ik}|$, $i = 1, \ldots, n$. Wähle a_{rj} mit $r = \text{argmax}_{i \geq j} |a_{ij}| s_i$

R3 a_{rs} mit $(r, s) = \text{argmax}_{i \geq j, k \geq j} |a_{ik}|$ (Pivot Element ist das betragsmäßig größte Element der noch zu behandelnden Untermatrix)

Hinter Regel R2 verbirgt sich der Versuch, die Auswahl so zu gestalten, als würde ein diagonal skaliertes Gleichungssystem gelöst werden (siehe Stichwort equilibrierte Matrizen in [2].Dort ist auch eine ausführlichere Diskussion von Pivot Strategien zu finden).
Ein wesentlich Diskussionspunkt bei numerischen Algorithmen ist deren Komplexität. Einfaches Abzählen liefert

Folgerung 2.8. Die Algorithmen 2.6 und 2.4 haben eine Komplexität von $\mathcal{O}(n^3)$. Eine Operation setzt sich dabei aus einer Multiplikation und einer Addition zusammen.

Damit ergibt sich

Bemerkung 2.9.

- Ist $A = PLR$ vorgelegt, so kann die i-te Spalte r_i der Matrix A^{-1} (sollte diese Matrix wirklich einmal gebraucht werden) durch Lösen des Gleichungssystems

$$LRr_i = Pe_i,$$

  erhalten werden, wobei e_i den i-ten Einheitsvektor in $\mathbb{R}^n$ bezeichnet. Der numerische Aufwand zur Berechnung der Inversen einer regulären Matrix beträgt demnach $\mathcal{O}(n^3)$, vergleiche auch Algorithmus von Gauß-Jordan, etwa in [2]. **?**

- Ist $A = PLR$, so gilt $\det A = (-1)^{\#\text{Zeilenvertauschungen}} \det R$, d.h. die Determinante von A kann mit $\mathcal{O}(n^3)$ Operationen berechnet werden.

Hier werden die **Landau'schen** Symbole O (und später auch) o verwendet, vergleiche [13]. Diese sind wie folgt definiert.

Definition 2.10. Seien f, g zwei Funktionen. Wir sagen

$f(x) = \text{O}(g(x))$ $(x \to x_0)$ (sprich f ist groß O von g für x gegen x_0) $\iff \exists C > 0, U(x_0)\ \forall x \in U(x_0)$: $|f(x)| \leq C|g(x)|$.

$f(x) = \text{o}(g(x))$ $(x \to x_0)$ (sprich f ist klein O von g für x gegen x_0) $\iff \forall \epsilon > 0\ \exists U(x_0)\ \forall x \in U(x_0)$: $|f(x)| \leq \epsilon |g(x)|$.

Aufgabe 2.11. $A = [a_{ij}]_{i,j=1,\ldots,n}$ heißt Bandmatrix mit Bandbreite m, falls $a_{ij} = 0$ für alle $|i - j| \geq m$. Zeigen Sie, daß R aus Algorithmus 2.4 höchstens die Bandbreite $2m - 1$ besitzt und in jeder Spalte von L höchstens m Elemente von Null verschieden sind.

Hinsichtlich Speicherplatzbedarf verhalten sich die Algorithmen 2.4,2.6 demnach gut, falls die Ausgangsmatrix Bandstruktur aufweist. Ist die Ausgangsmatrix jedoch nur schwach besetzt ohne Bandstruktur, so können die Faktoren L und R unten bzw. oben voll besetzt sein (sogenanntes **Fill In**).
Eine in der Anwendung wichtige Klasse von Matrizen bilden die positiv definiten Matrizen.

Definition 2.12. Eine (komplexe) Matrix A heißt positiv definit:$\iff$

a.) $A = A^H$, d.m. A ist hermitesche Matrix,

b.) $x^H A x > 0$ für alle $x \in \mathbb{C}^n, x \neq 0$.

Folgende Eigenschaften positiv definiter Matrizen sind aus der Linearen Algebra bekannt.

Hilfsatz 2.13.

- A^{-1} existiert und ist positiv definit.
- Alle Hauptuntermatrizen von A sind positiv definit.
- Alle Hauptminoren von A sind positiv.

Für positiv definite Matrizen gilt

Satz 2.14. Cholesky Zerlegung
Sei $A \in \mathbb{C}^{n,n}$ positiv definit. Dann gibt es genau eine untere Dreiecksmatrix L mit $l_{ik} = 0$ $(k > i)$, $l_{ii} > 0$ $(i = 1, \ldots, n)$ und

$$A = LL^H.$$

Ist A reell, so auch L.

Beweis. Mittels Induktion nach n.

Anfang $n = 1$: $A = a_{11} = l_{11}\bar{l}_{11}$ mit $l_{11} := \sqrt{a_{11}}$. Das ist möglich, da $a_{11} > 0$.

Annahme: $A_n = L_n L_n^H$, $l_{ik} = 0$ $(k > i)$, $l_{ii} > 0$ $(i = 1, \ldots, n)$.

Schritt $n \to n + 1$: Wir partitionieren $A \in \mathbb{C}^{n+1,n+1}$

$$A = \begin{bmatrix} A_n & b \\ b^H & a_{n+1n+1} \end{bmatrix},$$

mit $b \in \mathbb{C}^n$ und A_n positiv definit gemäß Hilfsatz 2.13. Also erfüllt A_n die Induktionsannahme und es gilt $A_n = L_n L_n^H$. Wir machen für die gesuchte Matrix L den Ansatz

$$L = \begin{bmatrix} L_n & 0 \\ c^H & \alpha \end{bmatrix}$$

und bestimmen c und α aus der Beziehung

$$\begin{bmatrix} L_n & 0 \\ c^H & \alpha \end{bmatrix} \begin{bmatrix} L_n^H & c \\ 0 & \alpha \end{bmatrix} = \begin{bmatrix} A_n & b \\ b^H & a_{n+1n+1} \end{bmatrix} \tag{33}$$

Es ergeben sich die Bedingungen

$$L_n c = b$$
$$c^H c + \alpha^2 = a_{n+1n+1}.$$

Die erste Gleichung liefert den Vektor c, da die Matrix L_n regulär ist. Ferner gilt $\alpha^2 = a_{n+1n+1} - c^H c$ und $\det(L_n) > 0$. Aus (33) ergibt sich

$$\det(A) = |\det(L_n)|^2 \alpha^2,$$

also $\alpha^2 > 0$, da $\det(A) > 0$. Daher gibt es genau ein positives α in (33), nämlich $\alpha = \sqrt{a_{n+1n+1} - c^H c}$ und der Beweis fertig.

□

Die Cholesky Zerlegung einer positiv definiten Matrix kann natürlich auch algorithmisch realisiert werden. Direkt aus dem Beweis von Satz 2.14 ergibt sich

Algorithmus 2.15. Cholesky Zerlegung einer Matrix $A \in \mathbb{C}^{n,n}$

1. $i = 1$ und $j = 1$.
2. $i > j$:

$$l_{ij} = \frac{a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk}}{l_{jj}}.$$

$i = j$:

$$l_{ii} = \sqrt{a_{ii} - \sum_{k=1}^{i-1} l_{ik}^2}.$$

Ist $l_{ii} \leq 0$ oder l_{ii} nicht reell: Matrix nicht positiv definit, STOP.

$i < j$:

$$l_{ij} = 0.$$

3. $i \leq n-1$: $i := i+1$ und gehe zu 2.

   $j \leq n-1$: $j := j+1$, $i := 1$ und gehe zu 2.

   STOP.

Die Matrixelemente des Faktors L können nicht beliebig groß werden, den es gilt

Aufgabe 2.16. Für die Matrixelemente des in Algorithmus 2.15 generierten Faktors L gilt

$$|l_{ij}| \leq \sqrt{a_{ii}} \text{ für } j = 1, \ldots, i \text{ und } i = 1, \ldots, n.$$

Der numerische Aufwand von Algorithmus 2.15 besteht in der Berechnung von n Quadratwurzeln sowie etwa $n^3/6$ Operationen.

2.1.2 Normen und Fehlerabsch"atzungen

Um messen zu können benötigen wir Normen.

Definition 2.17. Eine Abbildung $\|\cdot\| : \mathbb{C}^n \to \mathbb{R}$ heißt Norm, falls

a.) $\|x\| > 0$ für alle $x \in \mathbb{C}^n$, $x \neq 0$ (Definitheit)

b.) $\|\alpha y\| = |\alpha|\|x\|$ für alle $\alpha \in \mathbb{C}$, $x \in \mathbb{C}^n$ (Homogenität)

c.) $\|x + y\| \leq \|x\| + \|y\|$ für alle $x, y \in \mathbb{C}^n$ (Dreiecksungleichung)

Eigenschaften von Normen sind zusammengefaßt in

Aufgabe 2.18.

- Normen erfüllen die umgekehrte Dreiecksungleichung

  $$|\|x\| - \|y\|| \leq \|x - y\| \text{ für alle } x, y \in \mathbb{R}^n(\mathbb{C}^n). \tag{34}$$

- Normen sind gleichmäßig stetig bzgl. der Metrik $\rho(x, y) := \max_i |x_i - y_i|$ des $\mathbb{R}^n(\mathbb{C}^n)$ (was ist denn eine Metrik?)

- Alle Normen auf dem $\mathbb{R}^n\mathbb{C}^n$ sind äquivalent, d.m. für jedes Paar $\|\cdot\|_a, \|\cdot\|_b$ von Normen gibt es Konstanten c, C derart, daß

  $$c\|x\|_a \leq \|x\|_b \leq C\|x\|_a \text{ für alle } x, y \in \mathbb{R}^n(\mathbb{C}^n).$$

All' das sollen Sie nachweisen.

Für Matrizen $A \in M(m, n)$ (im Folgenden Matrizen mit m Zeilen, n Spalten und Einträgen aus $\mathbb{R}(\mathbb{C})$) werden Normen analog zu Definition 2.17 eingeführt (lediglich 'für alle $x \in \mathbb{C}^n$' durch 'für alle $A \in M(m, n)$' ersetzen. Gewöhnungsbedürftig ist

Definition 2.19.

- Eine Matrixnorm $\|\cdot\|$ heißt mit den Vektornormen $\|\cdot\|_a$ auf dem $\mathbb{C}^n$ und $\|\cdot\|_b$ auf dem $\mathbb{C}^m$, **verträglich (passend)**, falls

  $$\|Ax\|_b \leq \|A\|\|x\|_a \text{ für alle } x \in \mathbb{C}^n, A \in M(m,n)$$

  gültig ist.

- Eine Matrixnorm $\|\cdot\|$ für quadratische Matrizen heißt **submultiplikativ**, falls

  $$\|AB\| \leq \|A\|\|B\| \text{ für alle } A, B \in M(n,n).$$

- Sei $\|\cdot\|$ eine Vektornorm.

  $$\text{lub}(A) := \max_{x \neq 0} \frac{\|Ax\|}{\|x\|}$$

  heißt **Grenznorm**. Dabei kommt 'lub' von least upper bound, was soviel heißt wie Kleinste obere Schranke.

- Für invertierbare Matrizen $A \in M(n,n)$ heißt $\kappa(A) := \|A\|\|A^{-1}\|$ Konditionszahl von A (bzgl. der Matrixnorm $\|\cdot\|$).

Aus der Definition 2.19 wird klar, daß die Grenznorm von der verwendeten Vektornorm abhängt, ebenso die Konditionszahl. Wir bezeichnen im Folgenden lub_2 bei Verwendung der Euklidischen, lub_∞ bei Verwendung der Maximum Norm in der Definition der Grenznorm, κ_2 und κ_∞ entsprechend.

Beispiel 2.20. *Matrixnormen sind etwa*

- $\|A\|_\infty = \max_i \sum_{k=1}^{n} |a_{ik}|$ *(Zeilensummen Norm)*
- $\|A\|_1 = \max_j \sum_{i=1}^{m} |a_{ij}|$ *(Spaltensummen Norm)*
- $\|A\|_G = \max_{i,j} |a_{ij}|$ *(Gesamt Norm)*
- $\|A\|_E = \left(spur(A^H A)\right)^{\frac{1}{2}}$ *(Euklidische oder Schur Norm)*
- $\|A\|_2 = \max\{\sqrt{\lambda}; \lambda \in \mathbb{R}, A^H Ax = \lambda x \text{ für ein } x \in \mathbb{C}^n\}$ *(Spektral oder Hilbert Norm).*

Sind $\|\cdot\|_a$ auf dem $\mathbb{C}^n$, $\|\cdot\|_b$ auf dem $\mathbb{C}^m$ Vektornormen und ist $A \in M(m,n)$, so ist die Matrixnorm

$$\|A\| := \max_{x \neq 0} \frac{\|Ax\|_b}{\|x\|_a} \text{ Operator Norm}$$

mit $\|\cdot\|_a$ und $\|\cdot\|_b$ verträglich (passend) (Nachweis!, auch der Norm Eigenschaften).

Aufgabe 2.21. Weisen Sie nach, daß

- für jede mit der Vektornorm $\|\cdot\|_v$ verträgliche Matrix Norm $\|\cdot\|$ die Abschätzung $\text{lub}_v(A) \leq \|A\|$ gilt,
- $\text{lub}_\infty(A)$ der Zeilensummen Norm von A entspricht,
- $\text{lub}_2(A)$ der Spektral Norm von A entspricht,
- zu Vektornormen $\|\cdot\|_a$ auf dem $\mathbb{C}^n$, $\|\cdot\|_b$ auf dem $\mathbb{C}^m$ durch

  $$\|A\| := \max_{x \neq 0} \frac{\|Ax\|_b}{\|x\|_a}$$

  eine Matrixnorm auf $M(m,n)$ definiert wird, welche mit $\|\cdot\|_a$ und $\|\cdot\|_b$ verträglich ist.

Jetzt zu Fehlerabschätzungen. Wir wollen untersuchen, wie numerische Fehler bei der Lösung von linearen Gleichungssystemen schlimmstenfalls verstärkt werden. Dazu betrachten wir zu $Ax = b$ mit invertierbarer Koeffizientenmatrix $A \in M(n,n)$ das gestörte Gls

$$(A + \delta A)(x + \delta x) = b + \delta b. \tag{35}$$

Ziel soll nun sein, in einer gegebenen Vektornorm $\|\cdot\|$ die Fehler

$$\|\delta x\| \text{ (absoluter Fehler) und } \frac{\|\delta x\|}{\|x\|} \text{ (relativer Fehler)}$$

in den Störungen δA und δb abzuschätzen. Im Folgenden seien Vektor und Matrix Normen stets so gewählt, daß sie passend und submultiplikativ sind. Es gilt der

Satz 2.22. Es gelte für die Störung δA in (35) die Abschätzung

$$\kappa(A)\frac{\|\delta A\|}{\|A\|} < 1. \tag{36}$$

Dann erfüllt der Fehler δx die Abschätzungen

$$\|\delta x\| \leq \frac{\|A^{-1}\|}{1 - \kappa(A)\frac{\|\delta A\|}{\|A\|}} \left(\|\delta b\| + \|A^{-1}\|\|\delta A\|\|b\|\right) \tag{37}$$

und

$$\frac{\|\delta x\|}{\|x\|} \leq \kappa(A) \left(1 - \kappa(A)\frac{\|\delta A\|}{\|A\|}\right)^{-1} \left(\frac{\|\delta A\|}{\|A\|} + \frac{\|\delta b\|}{\|b\|}\right). \tag{38}$$

Die Konditionszahl (kurz Kondition) der Matrix spielt in diesen Abschätzungen die zentrale Rolle. In diesem Zusammenhang sprechen wir von gut konditionierten Problemen, falls der Verstärkungsfaktor für den relativen Fehler klein (d.m. moderat) ausfällt, andernfalls von schlecht konditionierten Problemen. Wegen (38) wird damit auch die Verwendung des Begriffs Konditionszahl für $\kappa(A)$ klar.

Beweis. (von Satz 2.22)
Aus (36) folgt unmittelbar, daß $A + \delta A$ invertierbar ist mit

$$\|(A + \delta A)^{-1}\| \leq \frac{\|A^{-1}\|}{1 - \kappa(A)\frac{\|\delta A\|}{\|A\|}}.$$

Wegen

$$\delta x = (A + \delta A)^{-1}(\delta b - \delta A x)$$

ergibt sich sofort

$$\|\delta x\| = \|(A + \delta A)^{-1}(\delta b - \delta A x)\| \leq \|(A + \delta A)^{-1}\|\left(\|\delta b\| + \|\delta A\|\|x\|\right)$$

und wegen $x = A^{-1}b$ unmittelbar (37). Division in dieser Ungleichung durch $\|x\|$, Anwendung der Dreiecksungleichung, Erweiterung mit $\|A\|/\|A\|$ und die Tatsache $\|b\| \leq \|A\|\|x\|$ liefern schließlich (38). □

Aufgabe 2.23. Bezeichne $\|\cdot\|$ die Grenznorm. Weisen Sie nach, daß für $F \in M(n,n)$ mit $\|F\| < 1$ die Matrix $(I + F)^{-1}$ existiert und die Abschätzung

$$\|(I + F)^{-1}\| \leq \frac{1}{1 - \|F\|}$$

erfüllt ist. Beweisen Sie mit diesem Hilfsmittel, daß mit den Notationen von Satz 2.22 aus (36) die Existenz von $(A + \delta A)^{-1}$ und die Abschätzung

$$\|(A + \delta A)^{-1}\| \leq \frac{\|A^{-1}\|}{1 - \kappa(A)\frac{\|\delta A\|}{\|A\|}}$$

folgen.

Weitere, nach Prager und Oettli benannte Fehlerabschätzungen, welche auch ohne die Inverse von A auskommen, sind in [2, (4.4.19)Satz] zu finden.

2.1.3 Iterative Verfahren

Wieder wollen wir das Gls

$$Ax = b$$

numerisch lösen, diesmal jedoch iterativ. Wieder sei A regulär, so daß zu jedem $b \in \mathbb{R}^n$ genau eine Lösung $x \in \mathbb{R}^n$ existiert. Wir spalten A auf in

$$A = W - R,$$

mit regulärem W. Dann gilt

$$Ax = b \Longleftrightarrow Wx = Rx + b.$$

Wir definieren

Definition 2.24. Sei $x^0 \in \mathbb{R}^n$ vorgelegt. Die Vorschrift

$$x^{m+1} = W^{-1}Rx^m + W^{-1}b =: Mx^m + c, \quad m \in \mathbb{N}, \quad N = W^{-1},\ M = W^{-1}R,\ c = Nb,$$

heißt **Basis Iterationsverfahren** zur Lösung von $Ax = b$.

Wir bemerken, daß das Basis Iterationsverfahren umgeschrieben werden kann als

$$W\left(x^{m+1} - x^m\right) = b - Ax^m := r^m, \quad m \in \mathbb{N}.$$

Spezielle Iterationsverfahren werden jetzt durch Variation von W und R bzw. M und N erhalten. Wir gehen im Folgenden davon aus, daß

$$A = D + L + R,$$

wobei D den Diagonalanteil, L das strikte untere und R das strikte obere Dreieck von A bezeichnen.

Definition 2.25. Das Iterationsverfahren $x^{m+1} = Mx^m + c$ mit

- $M = M^J := -D^{-1}(L + R)$, $c = c^J := D^{-1}b$ heißt **Jacobi Verfahren**,
- $M = M^J_\omega := (1 - \omega)I - \omega D^{-1}(L + R)$, $c = c^J_\omega := \omega D^{-1}b$ heißt **relaxiertes Jacobi Verfahren**,
- $M = M^{GS} := -(L + D)^{-1}R$, $c = c^{GS} := (L + D)^{-1}b$ heißt **Gauß-Seidel Verfahren**,
- $M = M^{SOR}_\omega := -(\omega L + D)^{-1}\left[(\omega - 1)D + \omega R\right]$, $c = c^{SOR}_\omega := \omega(\omega L + D)^{-1}b$ heißt **SOR- Verfahren** bzw. relaxiertes **Gauß-Seidel Verfahren**,

Motivieren lassen sich die relaxierten Verfahren wie folgt. Das Jacobi Verfahren schreibt sich

$$Dx^{m+1} = b - (L + R)x^m = Dx^m + b - (L + D + R)x^m = Dx^m + b - Ax^m = Dx^m + s^m.$$

Hier wird die alte Iterierte x^m im wesentlichen mit dem diagonalskalierten Residuum aufdatiert. Versucht wird nun, mit Hilfe eines Dämpfungsfaktors $\omega > 0$ das Konvergenzverhalten der Iteration zu verbessern. Wir erhalten

$$Dx^{m+1} = Dx^m + \omega s^m = Dx^m + \omega(b - (L + D + R)x^m)$$

und daraus unmittelbar die Vorschrift

$$x^{m+1} = M^J_\omega x^m + c^J_\omega.$$

Analog wird bei der Herleitung des SOR Verfahrens aus dem Gauß-Seidel Verfahren argumentiert. Wir haben

$$Dx^{m+1} = b - Rx^m - Lx^{m+1} = Dx^m + b - (D + R)x^m - Lx^{m+1} = Dx^m + s^{m+1}.$$

Dämpfen ergibt

$$Dx^{m+1} = Dx^m + \omega s^{m+1} = Dx^m + \omega b - \omega(D + R)x^m - \omega Lx^{m+1},$$

nach Umstellung nach x^{m+1} ergibt sich sofort

$$x^{m+1} = M^{SOR}_\omega x^m + c^{SOR}_\omega.$$

Es ist klar, daß die o.g. Verfahren nur durchführbar sind, falls alle auftretenden Matrizen wohldefiniert sind.
Jetzt zur Konvergenz von iterativen Verfahren. Wir betrachten zunächst die Basis Iteration $x^{m+1} = Mx^m + c$ und beweisen

Satz 2.26. Das Basis Iterationsverfahren aus Definition 2.24 konvergiert genau dann gegen die Lösung x^* von $Ax^* = b$, wenn der Spektralradius

$$\rho(M) := \max\{|\lambda_i|; \lambda_i \text{ Eigenwert von } M\} < 1 \tag{39}$$

erfüllt.

Beweis. $\Longrightarrow$: Nach Konstruktion ist x^* Lösung der Gleichung $x^* = Mx^* + c$. Damit ergibt sich unter Verwendung des Startvektors x^0

$$x^* - x^{m+1} = M(x^* - x^m) = M^m(x^* - x^0) = \lambda_i^m(x^* - x^0),$$

falls $(x^* - x^0)$ zufällig ein Vielfaches eines Eigenvektors von M zum Eigenwert λ_i ist. Da

$$0 \leftarrow \|x^* - x^{m+1}\| = |\lambda_i|^m \|(x^* - x^0)\|,$$

muß natürlich $|\lambda_i|^m$ für $m \to \infty$ gegen Null konvergieren, also notwendig $|\lambda_i| < 1$ erfüllt sein.
$\Longleftarrow$: Wir zeigen, daß $M^m \to 0$ für $m \to \infty$. Damit folgt dann die Behauptung aus

$$x^* - x^{m+1} = M^m(x^* - x^0).$$

Wir bringen M auf Jordan'sche Normalform mit einer nichtsingulären Matrix T (siehe etwa [4, 4.6.7]),

$$M = TJT^{-1}.$$

Damit gilt auch

$$M^m = TJ^mT^{-1} \text{ für alle } m \in \mathbb{N}.$$

Es reicht offenbar aus, $J^m \to 0$ für $m \to \infty$ nachzuweisen. Dazu schreiben wir J hin;

$$J = \begin{bmatrix} J_1 & & 0 \\ & \ddots & \\ 0 & & J_k \end{bmatrix}, \quad J_i = J_i(\lambda_i) = \operatorname{diag}(B_i^l(\lambda_i)),\ B_i^l = \begin{bmatrix} \lambda_i & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ & & & \lambda_i \end{bmatrix} l-\text{ter Jordanblock zu } \lambda_i,$$

wobei nach Voraussetzung $|\lambda_i| < 1$ für $i = 1, \ldots, k$ und k die Anzahl der verschiedenen Eigenwerte von M bezeichnet. Es gilt jetzt

$$J^m = \begin{bmatrix} J_1^m & & 0 \\ & \ddots & \\ 0 & & J_k^m \end{bmatrix},$$

weshalb zu beweisen bleibt, daß $J_i^m \to 0$ für $m \to \infty$, $i = 1, \ldots, k$. Das ist sicherlich erfüllt, falls $B_i^{lm} \to 0$ für $m \to \infty$. Sei jetzt λ Eigenwert von M und $B = B(\lambda)$ Jordan Block zu λ der Länge s ($1 \le s \le$ Vielfachheit von λ). Dann gilt für die Einträge von $B^m(\lambda)$

$$e_i^t B^m(\lambda) e_k = \begin{cases} \lambda^{m-(k-i)} \binom{m}{k-i} & 1 \le i \le k \le s \\ 0 & \text{sonst.} \end{cases}$$

Das heißt aber, daß bei fixen k, i die Einträge von $B^m(\lambda)$ für $m \to \infty$ gegen Null konvergieren, siehe Aufgabe 2.27. Damit ist alles bewiesen. □

Aufgabe 2.27. Weisen Sie nach, daß mit $B(\lambda)$ aus obigem Beweis

$$e_i^t B^m(\lambda) e_k = \begin{cases} \lambda^{m-(k-i)} \binom{m}{k-i} & 1 \le i \le k \le s \\ 0 & \text{sonst.} \end{cases}$$

und daß für $|\lambda| < 1$ bei fixen k, i

$$\lambda^{m-(k-i)} \binom{m}{k-i} \to 0 \text{ für } m \to \infty.$$

Folgerung 2.28. Hinreichend für die Konvergenz der Basisiteration aus Definition 2.24 ist

$$\|M\| < 1.$$

Beweis. Ist x Eigenvektor von M zum Eigenwert λ mit $\|x\| = 1$, so folgt

$$|\lambda| = |\lambda|\|x\| = \|\lambda x\| = \|Mx\| \leq \|M\|\|x\| = \|M\|,$$

d.m. mit $\|M\| < 1$ ist auch $\rho(M) < 1$. Satz 2.26 liefert nun die Behauptung. □

Für die numerische Realisierung iterativer Verfahren werden **Abbruch Bedingungen** benötigt. Abbruch Bedingungen sollten natürlich nur berechenbare Größen enthalten. Ist eine Toleranz $\epsilon > 0$ vorgelegt, so können wir die Iteration etwa stoppen, falls

- $\|r^m\| = \|b - Ax^m\| \leq \epsilon\|r^0\|$, oder
- $\|x^{m+1} - x^m\| \leq (1 - \rho_m)\epsilon\|x^m\|$

erfüllt ist. In der zweiten Bedingung ist

$$\rho_m = \frac{\|x^{m+1} - x^m\|}{\|x^m - x^{m-1}\|} \approx \rho(M) \text{ für große } m \in \mathbb{N},$$

für eine ausführliche Diskussion siehe [15, Kapitel 2.5]. Hier sei noch bemerkt, daß im Fall $\|M\| < 1$ die Mindestanzahl k_ϵ von Iterationen abgeschätzt werden kann, die zur Reduktion des Ausgangsfehlers $\|d^0\| = \|x^0 - x^*\|$ auf $\epsilon\|d^0\|$ benötigt werden. Dazu setzen wir an

$$\|M\|^{k_\epsilon}\|d^0\| \leq \epsilon\|d^0\| \Longrightarrow k_\epsilon \geq \frac{\ln \epsilon}{\ln \|M\|}.$$

Dabei wird natürlich davon ausgegangen, daß $\|M\|$ berechnet werden kann.
Jetzt noch einige Konvergenzresultate für die Verfahren aus Definition 2.25.

Satz 2.29. (Starkes Zeilen- und Spaltensummenkriterium)
Die Matrix A erfülle das starke Zeilensummenkriterium, d.m.

$$|a_{ii}| > \sum_{k \neq i} |a_{ik}| \text{ für alle } 1 \leq i \leq n, \tag{40}$$

oder das starke Spaltensummenkriterium, d.m.

$$|a_{kk}| > \sum_{i \neq k} |a_{ik}| \text{ für alle } 1 \leq k \leq n, \tag{41}$$

Dann sind Gesamt- und Einzelschrittverfahren für jeden Startwert $x^0 \in \mathbb{C}^n$ konvergent. Ferner gilt

$$\text{lub}_\infty(M^{GS}) \leq \text{lub}_\infty(M^J) < 1.$$

Beweis. (Nur für Gesamtschrittverfahren, vollständig siehe [3]). Wegen $M^J = -D^{-1}(L + R)$ gilt

$$\text{lub}_\infty(M^j) = \max_i \frac{1}{|a_{ii}|} \sum_{k \neq i} |a_{ik}| < 1.$$

Folgerung 2.28 liefert daher die Behauptung für das Zeilensummenkriterium. Das Spaltensummenkriterium entspricht dem Zeilensummenkriterium für A^t. Daher konvergiert das Gesamtschrittverfahren für A^t und es gilt nach Satz 2.26 $1 > \rho(M^J(A^t)) = \rho(M^{J^t}) = \rho(M^J)$, weil die Transponierte einer Matrix dieselben Eigenwerte wie die Matrix hat. Also konvergiert nach Satz 2.26 das Gesamtschrittverfahren auch für A. Der Nachweis für das Einzelschrittverfahren inc. der Abschätzung der Grenznormen findet sich etwa in [3]. □

Satz 2.30. (Schwaches Zeilensummenkriterium]
Die Matrix A sei unzerlegbar, d.m. der ihr zugeordnete gerichtete Graph D mit Knoten $\{P_1, \ldots, P_n\}$ und Kanten $e_{ij} = P_i \to P_j$, falls $a_{ij} \neq 0$ sei wegzusammenhängend, d.m. von jedem Knoten P_i führt zu jedem anderen Knoten ein gerichteter Weg. Ferner sei das schwache Zeilen- oder Spaltensummenkriterium für A erfüllt, d.m.

$$|a_{ii}| \geq \sum_{k \neq i} |a_{ik}| \text{ für alle } 1 \leq i \leq n \text{ und } |a_{i_0 i_0}| > \sum_{k \neq i_0} |a_{i_0 k}| \text{ für ein } i_0. \tag{42}$$

Dann konvergieren Gesamt- und Einzelschrittverfahren.

Ein Beweis findet sich wiederum in [3].
Konvergenzaussagen für das relaxierte Gesamtschrittverahren finden sich in [16]. Jetzt noch einige Konvergenzaussagen für das relaxierte Einzelschrittverfahren.

Satz 2.31. Es gilt

$$\rho(M_\omega^{SOR}) \geq |\omega - 1|, \tag{43}$$

d.h. daß das SOR Verfahren nur für $0 < \omega < 2$ konvergent sein kann.

Beweis. Untersuche die Eigenwerte von $M_\omega^{SOR} = -(\omega L + D)^{-1}[(1-\omega)D + \omega R]$. Zunächst bemerken wir, daß $\det(\omega L + D) = \det D$, also

$$\phi(\lambda) := \det(\lambda I - M_\omega^{SOR}) = \frac{1}{\det D} \det[(\omega L + D)(\lambda I - M_\omega^{SOR})] = \frac{1}{\det D} \det[(\lambda + 1 - \omega)D + \omega R + \lambda \omega L].$$

Ferner ist aus der linearen Algebra bekannt, daß $\phi(0) = \det M_\omega^{SOR} = \prod \lambda_i(M_\omega^{SOR})$. Demnach

$$\prod_i \lambda_i(M_\omega^{SOR}) = \phi(0) = \frac{1}{\det D} \det[(1-\omega)D + \omega R] = (1-\omega)^n.$$

□

Diese Bedingung schlägt sich nieder im

Satz 2.32. (Ostrowski/Reich)
Ist A positiv definit, so gilt

$$\rho(M_\omega^{SOR}) < 1 \text{ für alle } 0 < \omega < 2. \tag{44}$$

Diesen Satz zu beweisen ist Gegenstand von

Aufgabe 2.33. (Literaturaufgabe)
Beweisen Sie Satz 2.32.

Beispiel 2.34. *Wir betrachten die Finite Differenzen Diskretisierung der* **Poisson Gleichung**

$$-\Delta u(x) = f(x) \text{ in } \Omega := (0,1) \times (0,1), \quad u(x) = 0 \text{ auf } \partial\Omega, \tag{45}$$

wobei $\Delta := \sum_i \frac{\partial^2}{\partial x_i^2}$ *den Laplace Operator bezeichnet. Zur Diskretisierung mit finiten Differenzen benötigen wir ein Gitter. Dazu sei zu* $n \in \mathbb{N}$ *die Gitterweite* $h := \frac{1}{n+1}$, $x_i := ih$, $y_j := jh$, $i, j = 0, \ldots n+1$ *und*

$$\Omega_h := \{(x_i, y_j); i, j = 1, \ldots, n\} \text{ Gitterpunkt Menge.}$$

Ziel ist die Berechnung von Näherungen $u_{ij} \approx u(x_i, y_j)$, *wobei* u *die Lösung des Poisson Problems bezeichnet. Dazu ersetzen wir in der Differentialgleichung den Laplace Operator durch geeignete dividierte Differenzen und werten* f *an den Stellen* (x_i, y_j) *aus (Diskretisierung mittels 5-Punkt Stern):*

$$\Delta u(x_i, y_j) \approx \frac{1}{h^2}\left(u_{i+1j} - 2u_{ij} + u_{i-1j} + u_{ij+1} - 2u_{ij} + u_{ij-1}\right) = f(x_i, y_j) \text{ für } i, j = 1, \ldots, n. \tag{46}$$

Wir erhalten n^2 Gleichungen für $(n+2)^2$ Unbekannte. Die restlichen $4n+4$ Unbekannten ergeben sich aus der Forderung $u = 0$ auf $\partial\Omega$, hier $u_{i0} = u_{in+1} = u_{0,j} = u_{n+1j} = 0$ für $i, j = 0, \ldots, n+1$. Arbeiten wir diese Bedingungen in (46) ein, erhalten wir ein lineares Gleichungssystem mit block-tridiagonaler Koeffizienten Matrix;

$$AU = F \Longleftrightarrow \begin{bmatrix} T & -I & & \\ -I & T & \ddots & \\ & \ddots & \ddots & -I \\ & & -I & T \end{bmatrix} \begin{bmatrix} U_1 \\ U_2 \\ \vdots \\ U_n \end{bmatrix} = h^2 \begin{bmatrix} F_1 \\ F_2 \\ \vdots \\ F_n \end{bmatrix}, \tag{47}$$

wobei

$$T := \begin{bmatrix} 4 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 4 \end{bmatrix} \in M(n,n),\ I \in M(n,n)\ \textit{Einheitsmatrix}\ , U_i := \begin{bmatrix} u_{1i} \\ u_{2i} \\ \vdots \\ u_{ni} \end{bmatrix} \in \mathbb{R}^n,\ F_i\ \textit{analog.}$$

Aufgabe 2.35. Weisen Sie nach, daß die Matrix A aus (47) positiv definit ist. Tip (nur gültig mit Nachweis): Die Eigenvektoren z^{kl} und Eigenwerte λ^{kl} von A sind

$$z^{(kl)} \in \mathbb{R}^{n^2}, z_{ij}^{(kl)} = \sin k\pi ih \sin l\pi jh,\ \lambda^{(kl)} = 4 - 2\left(\cos k\pi h + \cos l\pi h\right).$$

Mit den Aussagen dieser Aufgabe folgern wir sofort aus dem Satz 2.32, daß das SOR Verfahren für unser Beispiel immer konvergiert, falls $0 < \omega < 2$ gilt.

Die Matrix A aus (47) ist ein Beispiel aus der Klasse der sogenannten konsistent geordneten Matrizen. Für Matrizen aus dieser Klasse kann der Relaxationsparameter ω^* angegeben werden, für welchen das SOR Verfahren am besten konvergiert.

Definition 2.36. Eine Matrix $A = (L + D + R) \in M(n,n)$ heißt **konsistent geordnet** $:\Longleftrightarrow$

$$J(\alpha) := \alpha D^{-1}L + \frac{1}{\alpha}D^{-1}R,\ \alpha \neq 0,$$

hat Eigenwerte, die unabhängig sind von α.

Beispiel 2.37. *Die Matrix A aus (47) ist konsistent geordnet, den es gilt in diesem Fall*

$$J(\alpha) = S_\alpha J(1) S_\alpha^{-1}\ \textit{mit}\ S_\alpha := \textit{diag}(K, \alpha K, \ldots, \alpha^{n-1}K) \in M(n^2, n^2),\ \textit{wobei}\ K := \textit{diag}(1, \alpha, \ldots, \alpha^{n-1}).$$

Für konsistent geordnete Matrizen fassen wir zusammen (siehe [3, Kapitel 8]

Satz 2.38. (Varga/Young)
Sei A konsistent geordnet. Dann gilt mit der Notation aus Definition 2.25

i. $\rho(M^{GS}) = \rho^2(M^J)$.

ii. Seien alle Eigenwerte von M^J reell und $\rho(M^J) < 1$. Dann ist

$$\omega^* := \text{argmin}_{0<\omega<2}\rho(M_\omega^{SOR})$$

explizit bestimmbar und es gilt

$$\omega^* = \frac{2}{1 + \sqrt{1 - \rho(M^J)^2}},\ \text{sowie}\ \rho(M_{\omega^*}^{SOR}) = \omega^* - 1. \tag{48}$$

Für den Spektralradius von M_ω^{SOR} gilt

$$\rho(M_\omega^{SOR}) = \begin{cases} \omega - 1 & \text{für } \omega^* \leq \omega \leq 2 \\ 1 - \omega + \frac{1}{2}\omega^2\rho(M^J)^2 + \omega\rho(M^J)\sqrt{1 - \omega + \frac{1}{4}\omega^2\rho(M^J)^2} & \text{für } 0 \leq \omega \leq \omega^*. \end{cases} \tag{49}$$

Sie sind jetzt sicherlich in Lage, wieder mal eine Aufgabe zu lösen.

Aufgabe 2.39. Es bezeichne wieder A die Matrix aus der Diskretisierung des Poisson Problems (45). Dann gilt

i. $\rho(M^J) = \cos \pi h$.

ii. $\rho(M^{GS}) = \cos^2 \pi h$.

iii.
$$\omega^* = \frac{2}{1 + \sin \pi h} \text{ und } \rho(M^{SOR}_{\omega^*}) = \frac{\cos^2 \pi h}{(1 + \sin \pi h)^2}.$$

iv. Die Zahl $I(n)$ mit $\rho(M^J)^{I(n)} = \rho(M^{SOR}_{\omega^*})$ gibt an, wie viele Schritte des Jacobi Verfahrens dieselbe Fehlerreduktion wie ein Schritt des optimalen Relaxationsverfahrens liefern. Weisen Sie nach, daß asymptotisch gilt
$$I(n) = \frac{4(n+1)}{\pi}.$$

Hinweis für i.: Die Matrix M^J besitzt dieselben Eigenvektoren wie A.

Eine Folgerung aus der vorangegangenen Aufgabe ist, daß die Konvergenz der in Definition 2.25 definierten iterativen Verfahren bei der numerischen Lösung des Poisson Problems mit zunehmender Diskretisierungsfeinheit h abnimmt.

2.1.4 Iterative Verfahren mit endlich vielen Iterationen

Neben direkten und iterativen Verfahren zur numerischen Lösung von linearen Gleichungssystemen gibt es noch iterative Verfahren, die bei exakter Rechnung die Lösung des Gleichungssystems nach endlich vielen Schritten berechnen. Zunächst beschreiben wir das Verfahren der konjugierten Gradienten, kurz **CG-Verfahren** (Conjugate Gradient Method), das zur numerischen Lösung von Gleichungssystemen mit positiv definiter Koeffizienten Matrix angewendet wird. Wir gehen wieder aus von dem Gleichungssystem

$$Ax = b \tag{50}$$

mit positiv definiter Matrix A.

Algorithmus 2.40. (Basis CG-Verfahren)
Gegeben seien $b, x^0 \in \mathbb{R}^n$ und eine positiv definite Matrix $A \in M(n,n)$. Berechnet wird die Lösung x von (50).

Initialisierung

$r^0 = b - Ax^0$
$p^0 = r^0$
$i = 0$

Do While $i \leq n$ und $r^i \neq 0$

$\alpha_i = (r^i, p^i)/(p^i, Ap^i)$
$x^{i+1} = x^i + \alpha_i p^i$
$r^{i+1} = r^i - \alpha_i Ap^i$
$\beta_i = (r^{i+1}, Ap^i)/(p^i, Ap^i)$
$p^{i+1} = r^{i+1} - \beta_i p^i$
$i = i + 1$

Endwhile

Der numerische Aufwand in jedem Iterationsschritt besteht in einer Matrix-Vektor-Multiplikation und der Berechnung von 3 Skalarprodukten. Ein Skalarprodukt wird eingespart, falls

$$\left\{\begin{array}{lcl} \alpha_i & = & |r^i|^2/(p^i, Ap^i) \\ \beta_i & = & |r^{i+1}|^2/|r^i|^2 \\ p^{i+1} & = & r^{i+1} + \beta_i p^i \end{array}\right. \tag{51}$$

im obigen Algorithmus gesetzt wird. Aufgrund von Orthogonalitätseigenschaften der Iterierten ergibt sich mit (51) ein zum Algorithmus 2.40 äquivalenter Algorithmus. Die Erfahrung zeigt allerdings, daß der numerische Mehraufwand stabilisierend wirken kann, d.h., daß die sparsame Variante häufiger versagt als die spendablere. Zunächst einige Aufgaben.

Aufgabe 2.41. cg-Verfahren
Berechnen Sie mit Hilfe des cg-Verfahrens Algorithmus 2.40 die **exakte** Lösung des Gleichungssystems

$$\begin{bmatrix} 2 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 2 \end{bmatrix} x = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}.$$

Aufgabe 2.42. Eigenschaften des cg-Verfahrens
Betrachten Sie das cg-Verfahren aus Algorithmus 2.40.

1. Zeigen Sie, dass sowohl im Basis cg-Verfahren (Algorithmus 2.40) als auch in der Formulierung (51) stets r^{i+1} orthogonal zu p^i ist.
2. Zeigen Sie, dass beide Formulierungen des cg-Verfahrens äquivalent sind.

Der Algorithmus 2.40 hat folgende Eigenschaften.

Satz 2.43. (CG-Verfahren ist ein direktes Verfahren)
Sei $A \in M(n,n)$ symmetrisch und positiv definit. Dann konvergiert das CG-Verfahren in Algorithmus 3.35 nach höchstens n Schritten gegen die exakte Lösung x^* des Gleichungssystems (50), wobei die Wahl des Startwertes x^0 beliebig ist.

Beweis. [5, Satz 5.2] □

Das CG-Verfahren ist demnach ein direktes Verfahren oder kann als solches aufgefaßt werden. Seine rekursive Formulierung läßt jedoch zu, die Lösung von (50) bis zu einer vorgelegten Genauigkeit zu berechnen. Die Anzahl der dazu benötigten Iterationen kann abgeschätzt werden.

Satz 2.44. (Konvergenzgeschwindigkeit des CG-Verfahrens)
Sei $A \in M(n,n)$ positiv definit und $x^0 \in \mathbb{R}^n$ beliebiger Startwert. Die Iterierten x^k des CG-Verfahrens erfüllen

$$\sqrt{(A(x^k - x^*), x^k - x^*)} \;\leq\; 2\left\{\frac{\sqrt{\lambda_{max}(A)} - \sqrt{\lambda_{min}(A)}}{\sqrt{\lambda_{max}(A)} + \sqrt{\lambda_{min}(A)}}\right\}^k \sqrt{(A(x^0 - x^*), x^0 - x^*)} \tag{52}$$

Beweis. [5, Satz 5.3] □

Es bezeichne

$$|x|_A := \sqrt{x^t A x}$$

die zur Matrix A assoziierte Vektornorm und

$$\kappa(A) = \frac{\lambda_{max}(A)}{\lambda_{min}(A)} (= \kappa_2(A)) \tag{53}$$

die Kondition der symmetrischen, positiv definiten Matrix A. Aus (51) wird eine Abschätzung erhalten dafür, wieviele Iterationen des CG-Verfahrens höchstens durchzuführen sind, um den Ausgangsfehler auf die Größenordnung ϵ zu reduzieren. Hinreichend dafür ist, daß k

$$\left\{\frac{\sqrt{\lambda_{max}(A)} + \sqrt{\lambda_{min}(A)}}{\sqrt{\lambda_{max}(A)} - \sqrt{\lambda_{min}(A)}}\right\}^k \geq \frac{2}{\epsilon}$$

erfüllt. Mit Hilfe von (53) kann das auch umgeschrieben werden zu

$$\left(\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1}\right)^k \;\geq\; \frac{2}{\epsilon},$$

also

$$k \;\geq\; \ln\frac{2}{\epsilon} \;/\; \ln\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1}.$$

Wird noch

$$\ln\frac{\sqrt{\kappa}+1}{\sqrt{\kappa}-1} \geq \frac{2}{\sqrt{\kappa}}, \quad \kappa > 1,\ \kappa = \kappa(A)$$

berücksichtigt, so ist für

$$k \;\geq\; \frac{1}{2}\sqrt{\kappa(A)}\ln\left(\frac{2}{\epsilon}\right)+1 \tag{54}$$

sichergestellt, daß

$$|x^k - x^*|_A \;\leq\; \epsilon|x^1 - x^*|_A$$

gilt.

Die maßgebliche Größe für die Konvergenzgeschwindigkeit ist hier die Kondition der Matrix A. Ihre Güte hängt maßgeblich von der Breite des Spektrums von A ab, d.h., je weiter $\lambda_{max}(A)$ und $\lambda_{min}(A)$ auseinanderliegen, desto pessimistischer sind die Erwartungen hinsichtlich der Fehlerreduktion. Die Konvergenzgeschwindigkeit kann mittels Vorkonditionierung verbessert werden. Dazu bemerke, daß die eindeutige Lösung x^* von (50) auch das Minimum der Funktion

$$f(x) := \frac{1}{2}x^tAx - b^tx$$

darstellt. Sei jetzt C eine nichtsinguläre Matrix. Dann ist mit

$$\tilde{A} := C^{-1}AC^{-t}, \quad \tilde{b} := C^{-1}b$$

das eindeutige Minimum y^* der Funktion

$$\tilde{f}(y) := \frac{1}{2}y^t\tilde{A}y - \tilde{b}^ty$$

durch

$$y^* = \tilde{A}^{-1}\tilde{b} = C^tx^*$$

gegeben, wobei x^* die Lösung von (50) darstellt. Ferner gilt wegen der positiven Definitheit von $\tilde{A}$ mit (52)

$$|y^k - y^*|_{\tilde{A}} \leq 2\left\{\frac{\sqrt{\kappa(\tilde{A})}-1}{\sqrt{\kappa(\tilde{A})}+1}\right\}^k |y^0 - y^*|_{\tilde{A}} \tag{55}$$

und zu gegebenen $\epsilon > 0$ gilt für die in (53) hergeleitete Zahl $k = k(\epsilon)$

$$k \geq \frac{1}{2}\sqrt{\kappa(\tilde{A})}ln\left(\frac{2}{\epsilon}\right)+1.$$

Ist $\kappa(\tilde{A}) < \kappa(A)$, so wird wohl y^* schneller gut approximiert als x^*. Weil mit

$$x^* = C^{-t}y^*, \quad x^k = C^{-t}y^k$$

$$|y^k - y^*|_{\tilde{A}} = |x^k - x^*|_A$$

folgt, motiviert sich aus diesen Überlegungen unter Beachtung von

$$C^{-t}\tilde{A}C^t = C^{-t}C^{-1}A =: W^{-1}A$$

(d.h., $W^{-1}A$ und $\tilde{A}$ haben dieselben Eigenwerte) der

Algorithmus 2.45. (Vorkonditioniertes CG-Verfahren)
Gegeben seien $b, x^0 \in \mathbb{R}^n$, eine positiv definite Matrix $A \in M(n,n)$ und ein geeigneter **Vorkonditionierer** $W \in M(n,n)$, i.e. W positiv definit. Berechnet wird die Lösung x^* von (50).

Initialisierung

$$r^0 = b - Ax^0$$
$$p^0 = W^{-1}r^0$$
$$\beta_0 = (p^0, r^0)$$
$$i = 0$$

Do While $i \leq n$ und $r^i \neq 0$

$$\alpha_i = \beta_i/(p^i, Ap^i)$$
$$x^{i+1} = x^i + \alpha_i p^i$$
$$r^{i+1} = r^i - \alpha_i Ap^i$$
$$q^{i+1} = W^{-1}r^{i+1}$$
$$\beta_{i+1} = (q^{i+1}, r^{i+1})$$
$$p^{i+1} = q^{i+1} + \frac{\beta_{i+1}}{\beta_i}p^i$$
$$i = i + 1$$

end while

Die Kunst besteht jetzt darin, die Matrix W so zu wählen, daß

i) $\kappa(\tilde{A})$ nahe bei 1 bzw. $K(\tilde{A}) << K(A)$ und/oder
ii) $q = W^{-1}r$ leicht auflösbar

gewährleistet werden kann. Diese beiden Eigenschaften widersprechen sich natürlich im Allgemeinen und gesucht ist hier der Königsweg.
Abschließend werden noch einige Vorkonditionierer angegeben und ihre Eigenschaften vorgestellt.

Beispiel 2.46. 1. *Diagonale Skalierung. Dabei wird*

$$W := \ diag(A) \tag{56}$$

gewählt.
2. *SOR-Vorkonditionierung. Dabei wird ausgehend von der Zerlegung* $A = L + D + R$

$$W := \frac{1}{2-\omega}\left(\frac{1}{\omega}D + L\right)\left(\frac{1}{\omega}D\right)^{-1}\left(\frac{1}{\omega}D + R\right) \tag{57}$$

gewählt. Im symmetrischen Fall heißt das SSOR Vorkonditionierung.
3. *Vorkonditionierung mittels unvollständiger LR-Zerlegung. Dabei wird* $A = M + R$ *mit* $M = \tilde{L}\tilde{R}$ *und* $R = A - \tilde{L}\tilde{R}$ *angesetzt und*

$$W := \tilde{L}\tilde{R} \quad (= \tilde{L}D\tilde{L}^t \text{ für symmetrische Matrizen}) \tag{58}$$

gewählt, vergleiche [11]. Zur Berechnung der Faktoren $\tilde{L}$ *und* $\tilde{R}$ *wird die L-R-Zerlegung etwa nur auf den Nicht-Nullelementen von A durchgeführt.*

Es gilt

Satz 2.47. (Konditionsverbesserung)
Für die SSOR-Vorkonditionierung gilt

$$\min_{0<w<2} \kappa(\tilde{A})(\omega) \leq \frac{1}{2} + \sqrt{\frac{1}{2}\kappa(A)}, \tag{59}$$

für die Vorkonditionierung mittels unvollständiger LR-Zerlegung

$$\kappa(\tilde{A}) \leq C\sqrt{\kappa(A)}, \tag{60}$$

falls die LR-Zerlegung nur auf den Nicht-Nullelementen der Ausgangsmatrix A durchgeführt wird.

Beweis. [11, (1.73 c)](1.73 c) für (59), (60) in [1]. □

Zur Vorbereitung auf das nächste Beispiel dient

Satz 2.48. (Eigenwerte einer Tridiagonalmatrix)
Sei

$$B := \begin{bmatrix} \beta & \gamma & & \\ \alpha & \ddots & \ddots & \\ & \ddots & \ddots & \gamma \\ & & \alpha & \beta \end{bmatrix} \in \mathbb{R}^{nxn}$$

mit $\alpha \cdot \gamma > 0$. Dann besitzt B die Eigenwerte

$$\lambda_i = \beta + 2\sqrt{\alpha\gamma}\text{sign}(\alpha)\cos\frac{i\pi}{n+1},\ 1 \leq i \leq n \tag{61}$$

und die Eigenvektoren v^i mit den Komponenten

$$v^i_j = (\frac{\alpha}{\gamma})^{\frac{j-1}{2}} \sin\frac{ij\pi}{n+1},\ 1 \leq i \leq n, 1 \leq j \leq n. \tag{62}$$

Der Beweis ist Gegenstand von

Aufgabe 2.49. Beweisen Sie Satz 2.48.

Beispiel 2.50. Wählen Sie

$$A = \frac{1}{h^2}\begin{bmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{bmatrix}, \quad h = \frac{1}{n+1}.$$

Dann ist A positiv definit und Satz 2.48 liefert für die Eigenwerte

$$\begin{aligned} h^2\lambda_{max}(A) &= 2 - 2\cos\tfrac{\pi n}{n+1} && \leq 4 \\ h^2\lambda_{min}(A) &= 2 - 2\cos\tfrac{\pi}{n+1} &= \tfrac{\pi^2}{(n+1)^2} + \mathcal{O}\left(\tfrac{1}{n^4}\right), \end{aligned}$$

also mit (53)

$$\kappa(A) \approx \frac{4}{\pi^2}h^{-2}.$$

Benötigt demnach das CG-Verfahren k Iterationen zur Reduktion des Ausgangsfehlers auf ϵ Teile seiner Ausgangsgröße, so kommen die mit SSOR und unvollständiger LR-Zerlegung vorkonditionierten Varianten mit $\mathcal{O}(\sqrt{\kappa})$ Iterationen aus.

Bemerkung 2.51. (Mehrgittermethoden)
Mit der Hilfe von Mehrgittermethoden lassen sich Vorkonditionierer entwickeln, mit deren Hilfe die Anzahl der zur Lösung notwendigen Iterationen im CG-Verfahren unabhängig von der Dimension des Gleichungssystems gehalten werden kann. Eine Ausführliche Diskussion zu Mehrgitterverfahren gibt Hackbusch in [7].

Sei jetzt die Matrix A in (50) regulär, aber nicht mehr notwendig symmetrisch. Das ist etwa der Fall bei Gleichungssystemen, die bei der Diskretisierung von Konvektions-Diffusions Gleichungen entstehen, vergleiche (12). Das iterative Verfahren der Wahl ist hier **GMRES** (Generalized Minimal Residual). Die Idee des Verfahrens ist die Folgende.

1. Berechne eine Orthonormalbasis des **Krylov-Raumes**

$$K(A, r^0, i) := \text{span } \left\{r^0, Ar^0, \ldots, A^{i-1}r^0\right\}. \tag{63}$$

2. Berechne eine das Residuum bzgl. der Euklidischen Norm minimierende Iterierte $x^i \in x^0 + K(A, r^0, i)$.

Algorithmus 2.52. (Vorkonditioniertes GMRES)
Gegeben seien $b, x^0 \in \mathbb{R}^{n\times n}$, eine reguläre Matrix $A \in \mathbb{R}^{n\times n}$ und ein Vorkonditionierer $W \in M(n,n)$. Berechnet wird die Lösung x^* von $Ax^* = b$. Die natürliche Zahl $m \geq 1$ sei ein Restart Index.

$r^0 = b - Ax^0$

restart $v^1 = r^0/\|r^0\|_2$

$i = 1$

Do while $1 \leq i \leq m$ and $r^{i-1} \neq 0$

for $j = 1, i$

$z^j = W^{-1}v^j$

$w = Az^j$

for $l = 1, j$

$h_{lj} = w^t v^l$

$w = w - h_{l_j} v^l$

endfor $_l$

$h_{j+1,j} = \|w\|_2$

$v^{j+1} = w/h_{j+1,j}$

endfor$_j$

$H^i = (h_{\alpha\beta})_{\alpha=1,i+1\ \beta=1,i} \in \mathbb{R}^{i+1,i}$

$V^i = [v^1, \ldots, v^i] \in \mathbb{R}^{n,i}$

$e_1^{(i+1)} = (\|r^0\|_2, 0, \ldots, 0)^t \in \mathbb{R}^{i+1}$

$\beta^i = \arg\min_{\beta\in\mathbb{R}^i} \|e_1^{(i+1)} - H^i\beta\|_2$

$x^i = x^0 + W^{-1}V^i\beta^i$

$r^i = b - Ax^i$

$i = i + 1$

end while

If $r^m \neq 0$

$x^0 = x^m$

$r^0 = r^m$

goto restart

endif

Bemerkung 2.53. Der restart Index m sollte groß genug gewählt werden, damit das GMRES-Verfahren seine guten Konvergenzeigenschaften entwickeln kann. Diese kommen zum Tragen, wenn das Residuum in eine *gute* Richtung zeigt.

Herleitung des Algorithmus: Die i-te Iterierte soll die Gestalt

$$x^i = x^0 + \sum_{j=0}^{i-1} \alpha_j A^j r^0 \tag{64}$$

haben, d.h. im Raum $x^0 + K(A, r^0, i)$ liegen und soll

$$\|r^i\|_2 = \|b - Ax^i\|_2 = \min_{\alpha \in \mathbb{R}^i} \|r^0 - \sum_{j=0}^{i-1} \alpha_j A^{j+1} r^0\|_2 \tag{65}$$

erfüllen. Um dieses Minimierungsproblem zu lösen, wird die Struktur von r^i ausgenutzt. Dazu wird zunächst mit Hilfe des **Gram-Schmidt Orthonormalisierungsverfahrens** eine Orthonormalbasis von $K(A, r^0, i)$ berechnet. Algorithmisch

$$\begin{aligned} & v^1 = r^0/\|r^0\|_2 \\ & \text{for } j = 1, i-1 \\ & \quad w^j = Av^j \\ & \quad \hat{v}^{j+1} = w^j - \sum_{l=1}^{j} v^{l^t} w^j v^l \\ & \quad v^{j+1} = \hat{v}^{j+1}/\|\hat{v}^{j+1}\|_2 \\ & \text{endfor } {}_j \end{aligned} \tag{66}$$

Hilfsatz 2.54. (GMRES iteriert nur endlich oft)
Das Gram-Schmidt-Verfahren (66) bricht ab, falls für ein $1 \le k \le i-1$ der Vektor $A^k r^0 \in K(A, r^0, k)$ erfüllt. Dann ist x^k aber die gesuchte Lösung, d.h. $x^k = x^*$.

Beweis. Sei $1 \le k \le i-1$ mit $A^k r^0 \in K(A, r^0, k)$ und k der kleinste Index dieser Art. Dann gilt

$$A^k r^0 = \sum_{j=0}^{k-1} \gamma_j A^j r^0, \quad \gamma_j \text{ geeignet,}$$

wobei $\gamma_0 \neq 0$, denn sonst

$$A^k r^0 = \sum_{j=1}^{k-1} \gamma_j A^j r^0 = A \sum_{j=1}^{k-1} \gamma_j A^{j-1} r^0,$$

woraus wegen der Regularität von A

$$A^{k-1} r^0 = \sum_{j=1}^{k-1} \gamma_j A^{j-1} r^0 = \sum_{j=0}^{k-2} \gamma_{j+1} A^j r^0$$

folgt, $A^{k-1} r^0$ also ein Element aus $K(A, r^0, k-1)$ sein müßte, ein Widerspruch zur Minimalität von k. Für r^k ergibt sich

$$\begin{aligned} r^k &= b - A(x^0 + \sum_{j=0}^{k-1} \alpha_j A^j r^0) \\ &= r^0 - \sum_{j=0}^{k-1} \alpha_j A^{j+1} r^0 \\ &= r^0 - \sum_{j=0}^{k-2} \alpha_j A^{j+1} r^0 - \alpha_{k-1} A^k r^0 \\ &= r^0 - \sum_{j=0}^{k-2} \alpha_j A^{j+1} r^0 - \alpha_{k-1} \sum_{j=0}^{k-1} \gamma_j A^j r^0 \\ &= (1 - \alpha_{k-1}\gamma_0) r^0 - \sum_{j=1}^{k-1} (\alpha_{j-1} + \alpha_{k-1}\gamma_j) A^j r^0. \end{aligned}$$

Wähle jetzt

$$\alpha_{k-1} = 1/\gamma_0, \ \alpha_{j-1} = -\alpha_{k-1}\gamma_j = -\gamma_j/\gamma_0, \quad j = 1, \ldots, k-2$$

so daß $r^k = 0$ folgt. Offensichtlich ist der so definierte Vektor α der in (64) gesuchte, weil dieser $\|r^k\|_2 = 0$ erfüllt. □

Folgerung 2.55. (GMRES ist nach n Iterationen fertig)
Ist $A \in M(n,n)$ regulär, so berechnet GMRES nach höchstens n Schritten die exakte Lösung x^* von (50).

Beweis. Es existieren maximal n linear unabhängige Vektoren im $\mathbb{R}^n$, also bricht Gram-Schmidt nach höchstens n Schritten ab. □

Bemerkung 2.56. Da in jedem GMRES-Schritt bzgl. aller bisherigen Suchrichtungen minimiert wird, ist die Folge $\{\|r^i\|_2\}_{i\in\mathbb{N}}$ monoton fallend.

Jetzt weiter mit der Herleitung. Angenommen, der Prozeß (66) breche nicht innerhalb der ersten i-Schritte ab und erzeuge ein ONB $\{v^1,\dots,v^i\}$ von $K(A,r^0,i)$. Dann gilt mit geeignetem β_j

$$x^i = x^0 + \sum_{j=0}^{i-1} \alpha_j A^j r^0 = x^0 + \sum_{j=1}^{i} \beta_j v^j$$

und

$$r^i = r^0 - \sum_{j=1}^{i} \beta_j A v^j = r^0 - \sum_{j=1}^{i} \beta_j w^j,\ w^j = A v^j.$$

Aus (66) folgt

$$w^j = \|\hat{v}^{j+1}\|_2 v^{j+1} + \sum_{l=1}^{j} v^{l^t} w^j v^l.$$

Definiere jetzt die Matrix $H^i \in \mathbb{R}^{i+1,i}$ gemäß

$$\begin{array}{lcll} h_{lj} & = & v^{l^t} w^j & , \ l \le j \le i, \\ h_{j+1,j} & = & \|\hat{v}^{j+1}\|_2 & , \ 1 \le j \le i, \\ h_{lj} & = & 0 & , \ 1 \le j+1 < l \le i+1 \end{array}$$

und setze $V^{i+1} = [v^1,\dots,v^{i+1}] \in \mathbb{R}^{n\times(i+1)}$ (d.m. daß Prozeß (66) für $j = 1,\dots,i$ ausgeführt wurde). Dann läßt sich w^j mit $\beta = (\beta_1,\dots,\beta_i)^t$ schreiben als

$$w^j = h_{j+1,j}\ v^{j+1} + \sum_{l=1}^{j} h_{lj} v^l = \sum_{l=1}^{j+1} h_{lj} v^l,$$

ergo

$$\begin{array}{rcl} r^i & = & r^0 - \sum\limits_{j=1}^{i} \beta_j \sum\limits_{l=1}^{j+1} h_{lj} v^l \\ & = & r^0 - \sum\limits_{j=1}^{i} \sum\limits_{l=1}^{j+1} v^l h_{lj} \beta_j \\ & = & r^0 - V^{i+1} H^i \beta. \end{array}$$

Mit $v^1 = r^0/\|r^0\|_2$ und $e_1^{(i+1)} = (\|r_0\|_2, 0,\dots,0)^t \in \mathbb{R}^{i+1}$ gilt

$$r^0 = V^{i+1} e_1^{(i+1)},$$

und da die Spalten von V^{i+1} orthonormal sind, folgt

$$\|r^i\|_2 = \|r^0 - V^{i+1} H^i \beta\|_2 = \|V^{i+1}(e_1^{(i+1)} - H^i\beta)\|_2 = \|e_1^{(i+1)} - H^i\beta\|_2. \tag{67}$$

Damit ist das Minimierungsproblem (65) durch das Minimierungsproblem (67) mit oberer Hessenberg-Matrix H^i ersetzt worden. Die Herleitung des Algorithmus 2.52 ist damit abgeschlossen.

Verbleibt die Lösung des Least-Squares-Problems (67). Wir machen betrachten das Problem allgemeiner. Seien dazu $H \in \mathbb{R}^{n+k,n}$ mit $\text{Rang}(A) = n$ und $a \in \mathbb{R}^{n+k}$ gegeben. Betrachtet wird jetzt das Least-Squares (lineare Ausggleichs) Problem

$$\beta^* = \arg\min_{\beta \in \mathbb{R}^n} \|a - H\beta\|_2,$$

dessen Lösung mit Hilfe einer QR-Zerlegung realisiert werden soll. Sei dazu Q eine orthogonale Matrix, d.h. Q erfülle

$$Q^t Q = Id_{n+k}.$$

Dann gilt wegen der Orthogonalität von Q und der Definition der Euklidischen Norm

$$\|Q(a - H\beta)\|_2 = \|a - H\beta\|_2.$$

Wähle jetzt die Matrix Q so, daß

$$QH = \begin{bmatrix} R \\ 0 \end{bmatrix}, \quad R \in M(n,n),\ 0 \in \mathbb{R}^{k \times n}$$

mit oberer Dreiecksmatrix R. Schreibe noch

$$Qa = \begin{bmatrix} a_1 \\ a_2 \end{bmatrix} \in \mathbb{R}^{n+k},\ a_1 \in \mathbb{R}^n,\ a_2 \in \mathbb{R}^k.$$

Dann gilt mit der Lösung β^* von

$$R\beta^* = a_1 \tag{68}$$

auf jeden Fall

$$\beta^* = \arg\min_{\beta \in \mathbb{R}^n} \|a - H\beta\|_2$$

und

$$\min_{\beta \in \mathbb{R}^n} \|a - H\beta\|_2 = \|a_2\|_2.$$

Darüber hinaus ist das Gleichungssystem (68) leicht auflösbar.
Es verbleibt somit die Konstruktion einer Matrix Q, welche H auf die gewünschte Gestalt transformiert. Dazu wird Q aus Drehungen komponiert. Es bezeichne

$$D(p,q,\phi) := \begin{bmatrix} 1 & & & & & & & & & & \\ & \ddots & & & & & & & & & \\ & & 1 & & & & & & & & \\ & & & \cos\phi & & \dots & & \sin\phi & & & \leftarrow p \\ & & & & 1 & & & & & & \\ & & & & & \ddots & & & & & \\ & & & & & & 1 & & & & \\ & & & -\sin\phi & & \dots & & \cos\phi & & & \leftarrow q \\ & & & & & & & & 1 & & \\ & & & & & & & & & \ddots & \\ & & & & & & & & & & 1 \end{bmatrix} \tag{69}$$

eine Drehung um den Winkel ϕ in der Ebene, welche durch die p-te und q-te Koordinatenrichtung aufgespannt wird. Durch Multiplikation von Links mit $D(p,q,\phi)$ soll H so modifiziert werden, daß für

$$H' = D(p,q,\phi)H$$

mit

$$H = (h_{ij})_{i=1,n+k;j=1,n}$$

die Beziehungen

$$0 = h'_{qp} = -h_{pp} \sin\phi + h_{qp} \cos\phi$$

gelten. Mit $c := \cos\phi$ und $s := \sin\phi$ ergibt sich

$$c = \frac{s h_{pp}}{h_{qp}} = \sqrt{1 - c^2} \frac{h_{pp}}{h_{qp}}.$$

Wähle

$$c = \frac{|\alpha|}{\sqrt{1+\alpha^2}}, \quad \text{dann } s = \frac{|\alpha|}{\alpha\sqrt{1+\alpha^2}}, \text{ wobei } \alpha := \frac{h_{pp}}{h_{qp}}.$$

Damit ist aber der Winkel ϕ in $D(p, q, \phi)$ festgelegt.
Diese Methode soll jetzt in (67) angewendet werden, um $H' \in \mathbb{R}^{i+1,i}$ auf obere Dreicksform zu transformieren. Der nötige numerische Aufwand ergibt sich aus

Hilfsatz 2.57. (QR-Zerlegung einer Hessenbergmatrix)
Sei $H \in M(n, n)$ obere Hessenbergmatrix. Eine Zerlegung

$$H = QR$$

mit orthogonalem Q und oberer Dreiecksmatrix R ist mit den $n-1$ Rotationsmatrizen $D(1, 2; \phi_1)$, $D(2, 3; \phi_2)$, ..., $D(n-1, n; \phi_{n-1})$ durchführbar.

Beweis. Übung. □

Als direkte Konsequenz ergibt sich für den numerischen Aufwand in (67):

i.) H^i auf obere Dreiecksformen R^i : i-mal Wurzel ziehen und $4i$ Multiplikationen

ii.) Rückwärts einsetzen mit R^i.

Bemerkung 2.58. Der numerische Aufwand von GMRES steigt mit wachsender Anzahl von Iterationen. Schnelle Konvergenz wird allerdings erst bei relativ großen Werten von i beobachtet, ein Restart sollte demnach in Algorithmus 2.52 nicht mit zu kleinem m erfolgen.

Eine ganz andere Beobachtung hat Jan Brandts 1998 gemacht. Bezeichnen $CL(n)$ n Schritte eines klassischen Verfahrens (etwa des Gauß-Seidel oder des SOR-Verfahrens) und GMRES (k) k Schritte GMRES, so stellt sich häufig folgendes Phänomen heraus.

Beobachtung 2.59. (Pre-Processing für GMRES)
Bei der numerischen Lösung von Gleichungssystemen mit nichtsymmetrischen Matrizen unter Verwendung von GMRES beobachtete Jan Brandts 1998 das Verhalten

$$\text{GMRES}(k-n) \circ CL(n) r^0 \approx \text{GMRES}(k) r^0.$$

Vom Standpunkt des numerischen Aufwands aus ist $\text{GMRES}(k-n) \circ CL(n) r^0$ wesentlich besser als $\text{GMRES}(k) r^0$. Gerade bei Matrizen, welche aus der Diskretisierung von Konvektions-Diffusionsproblemen resultieren, scheint diese Beobachtung zuzutreffen.

Bemerkung 2.60. Für reguläre indefinite Matrizen, wie sie häufig bei Sattelpunktproblemen auftauchen, gibt es Varianten des CG-Verfahrens, welche nicht soviel Speicherplatz wie GMRES benötigen. Genannt seien hier

- **BiCG (Biconjugate Gradients)**
- **BiCG$_3$(Biconjugate Gradients mit 3 Matrixmultiplikationen)**
- **BiCGSTAB(Biconjugate Gradients Stabilisiert)**
- **CGS(Conjugate Gradient Squared)**
- **CSBCG(Composite Step Biconjugate Gradients)**

- **LAL(Look ahead Lanzcos)**
- **QMR(Quasi Minimal Residual)**
- **TFQMR(Transpose Free QMR)**
- **SymmLQ(Symmetrische LQ-Zerlegung)**

Eine vergleichende Besprechung dieser Verfahren findet etwa in [19, 6] statt.

Die nächste Aufgabe dient dazu, das relaxierte Einzelschrittverfahren aus Definition 2.25 mit dem Verfahren der konjugierten Gradienten aus Algorithmus 2.40 numerisch zu vergleichen.

Aufgabe 2.61. Betrachtet wird das Dirichletproblem

$$\begin{aligned} -\Delta u(x,y) &= f(x,y), \quad (x,y) \in \Omega := (0,1) \times (0,1), \\ u(x,y) &= 0, \qquad\quad (x,y) \in \Gamma := \partial\Omega. \end{aligned}$$

Die Diskretisierung erfolge mittels des **5-Punkt Differenzensterns** auf einem quadratischen Gitter der Schrittweite h nach den folgenden beiden Schemata (vgl. Übung)

$$\frac{1}{h^2}\begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix} u^h = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} f \qquad \text{bzw} \qquad = \frac{1}{12}\begin{bmatrix} 0 & 1 & 0 \\ 1 & 8 & 1 \\ 0 & 1 & 0 \end{bmatrix} f$$

1. Lösen Sie die diskreten Probleme für $n = 10$ und $h = \frac{1}{n+1}$ numerisch mit Hilfe des $SOR\text{-}Verfahrens$ jeweils für die Relaxationsparameter $\omega = 1.0, \omega = 1.8$, $\omega = 2 - \frac{c}{\sqrt{c^2+n^2-1}}$ mit $c = 5.6$ und $\omega = \omega^*$ aus Aufgabe 2.39 zu den rechten Seiten

   (a) $f(x,y) = \sin(\pi x)\sin(\pi y)$ und

   (b) $f(x,y) = 2(x + y - x^2 - y^2)$.

   Wie sieht dabei jeweils die exakte Lösung aus? Brechen Sie die Iteration ab, falls der relative Fehler zweier benachbarter Iterierter, gemessen in der Maximumnorm, kleiner als 10^{-6} ausfällt. Dokumentieren Sie die Anzahl der benötigten Iterationen und stellen Sie die numerischen Lösungen graphisch dar.

2. Wie unter 1., jedoch mit dem Basis CG-Verfahren aus Algorithmus 2.40.

3. Ermitteln Sie experimentell die Konvergenzordnungen der beiden Diskretisierungen für das Fehlerfunktional $E(h) := \max_{i,j}|u(x_i, y_j) - u^h_{ij}|$, wobei u^h die numerische Lösung zur Gitterweite h bezweichnet. Dabei ist füür zwei Gitterweiten $h_1 \neq h_2$ die **Experimentelle Konvergenzordnung** eines Fehlerfunktionals $E(h)$ definiert durch

   $$\text{EOC} = \frac{\ln E(h_1) - \ln E(h_2)}{\ln h_1 - \ln h_2} \text{ (Experimental Order of Convergence).} \tag{70}$$

   Was sagt das numerische Ergebnis hinsichtlich der Approximationsgüte der Finite-Differenzen Approximation aus?

4. Die Diskretisierung erfolge jetzt mittels des **Kompakten 9 Punkte Differenzensterns** auf einem quadratischen Gitter der Schrittweite h nach den folgenden beiden Schemata

   $$\frac{1}{6h^2}\begin{bmatrix} -1 & -4 & -1 \\ -4 & 20 & -4 \\ -1 & -4 & -1 \end{bmatrix} u^h = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} f \qquad \text{bzw} \qquad = \frac{1}{12}\begin{bmatrix} 0 & 1 & 0 \\ 1 & 8 & 1 \\ 0 & 1 & 0 \end{bmatrix} f$$

   Wiederholen Sie die Berechnungen aus den vorangegangenen Aufgabenteilen.

5. Implementieren Sie das CG-Verfahren mit SSOR Vorkonditionierung (57) und testen Sie Ihren Algorithmus an den o.g. Beispielen. Wird die Aussage aus (60) verifiziert?

3 Nichtlineare Gleichungen

Jetzt sollen numerische Methoden zur Lösung von nichtlinearen Gleichungen der Form

$$F(x) = 0, \quad x \in \mathbb{R}^n,\ F : \mathbb{R}^n \to \mathbb{R}^n \tag{71}$$

entwickelt und untersucht werden. Aufgabenstellungen dieser Art treten sehr häufig bei der Modellierung physikalischer Vorgänge auf. So ergibt sich etwa nach geeigneter Diskretisierung der nichtlinearen Randwertaufgabe

$$u''(x) = e^{u(x)} \text{ für } x \in (0,1),\ \text{und } u(0) = u(1) = 1$$

zu $n \in \mathbb{N}$ "uber dem Gitter $x_i := ih (i = 0, \ldots, n+1)$ mit Gitterweite $h := \frac{1}{n+1}$ ein nichtlineares Gleichungssystem der Form

$$Au + Z(u) = 0,$$

wobei $u = [u_1, \ldots, u_n]^t$. Dabei gilt unter Verwendung zentraler Differenzen für die Approximation von $u''(x_i)$

$$A = \frac{1}{h^2}\begin{bmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{bmatrix},\ \text{und } Z(u_1, \ldots, u_n) = \begin{bmatrix} e^{u_1} \\ \vdots \\ \vdots \\ e^{u_n} \end{bmatrix}.$$

Die Gleichung (71) soll numerisch mit Hilfe einer **Fixpunkt Iteration**

$$x^0 \in \mathbb{R}^n \text{ gegeben },\, x^{i+1} = G(x^i),\ i = i+1, \tag{72}$$

gelöst werden. Die Funktion $G : \mathbb{R}^n \to \mathbb{R}^n$ heißt dabei **Iterations Funktion**. Iterationsfunktionen ergeben sich häufig direkt aus der Aufgabenstellung, etwa falls F die spezielle Gestalt

$$F(x) = x - H(x)$$

besitzt. In unserem Beispiel wäre etwa $H(x) = -A^{-1}Z(x)$. Dann ist die natürliche Wahl der Iterations Funktion $G := H$.
Die Grundlegende Idee zur numerischen Lösung von (71) und zur Konstruktion von Iterationsfunktionen besteht in der Formulierung einer Sequenz von *einfachen* Ersatzaufgaben. Dazu wird die Funktion F lokal durch geeignete einfach Modelle ersetzt, deren Nullstellen einfach zu berechnen sind. Das führt auf die Betrachtung von Linearisierungen der Funkion F in der Nähe einer Nullstelle ξ. Es gilt dort für gegebenes $x \in \mathbb{R}^n$ (F als stetig differenzierbar vorausgesetzt) nach der Taylor Formel

$$0 = F(\xi) = F(x) + DF(x)(\xi - x) + \mathrm{o}(\|\xi - x\|) \text{ für } (x \to \xi),$$

wobei $DF(x) \in M(n,n)$ die Jacobi Matrix von F bei x bezeichnet. Dabei Es liegt jetzt auf der Hand, die Nullstelle ζ der affin–linearen Funktion

$$T_1[F,x](z) := F(x) + DF(x)(z - x)$$

als Approximation der Nullstelle ξ von F aufzufassen, denn $T_1[F,z]$ ist sicherlich eine gute Approximation an (brauchbares Modell für) die Funktion F, falls x nahe bei ξ liegt. Wir erhalten

$$\eta = x - DF(x)^{-1}F(x).$$

Die Berechnung von η ist hier natürlich nur möglich, falls $DF(x)^{-1}$ existiert. Ausgehend von x^0 in der Nähe von ξ iterieren wir dieses Vorgehen. Das sich daraus ergebende Iterationsverfahren heißt **Newton Verfahren** und hat die Form

Algorithmus 3.1. (Newton Verfahren)

1. $x^0 \in \mathbb{R}^n$ gegeben, $i = 0$,

2. löse nach Δx^i,
$$DF(x^i)\Delta x^i = -F(x^i),$$
3. datiere auf
$$x^{i+1} = x^i + \Delta x^i,$$
4. $i = i + 1$, gehe zu 2.

Die Iterations Funktion des Newton Verfahrens ist gegeben durch

$$G(x) = x - DF(x)^{-1}F(x). \tag{73}$$

Die Wahl des Startwertes x^0 für das Newton Verfahren ist kritisch. In praktischen Aufgabenstellungen resultiert das nichtlineare Gleichungssystem (71) selber aus einer mathematischen Modellierung, so dass wir vielleicht bereits eine Idee haben, welche Vektoren (Werte) für ξ in Frage kommen.

Aufgabe 3.2. Sei $f : \mathbb{R} \to \mathbb{R}$ 2 mal stetig differenzierbar. Konstruieren Sie das **Newton-Raphson Verfahren 2ten Grades** zur numerischen Berechnung von Nullstellen einer Funktion f, welches durch die Iterations Funktion

$$G(x) := x - \frac{2f(x)}{f'(x) + \text{sign}(f'(x))\sqrt{f'(x)^2 - 2f(x)f''(x)}} \tag{74}$$

gegegen ist. Tip: Ersetzen Sie die Funktion f durch ein Modell 2ter Ordnung, vergl. Motivation von Algorithmus 3.1.

3.0.5 Konvergenzsätze

Jetzt zu Konvergenzaussagen für Fixpunkt Iterationen. Dazu zunächst einige Begriffe.

Definition 3.3.

- ξ heißt **Fixpunkt** von $G :\Longleftrightarrow G(\xi) = \xi$.
- Sei ξ Fixpunkt von G und es gelte für alle Startvektoren x^0 für die mittels der Fixpunkt Iteration (72) erzeugten Iterierten $\{x^i\}_{i\in\mathbb{N}}$ für ein festes $p \geq 1$
$$\|x^{i+1} - \xi\| \leq C\|x^i - \xi\|^p \text{ für alle } i \geq 0 \text{ und } C < 1 \text{ falls } p = 1.$$
Dann wird das durch G erzeugte Iterations Verfahren als Verfahren von mindestens p-ter Ordnung bezeichnet.

Aus dieser Definition folgt sofort

Satz 3.4. Jedes Verfahren p-ter Ordnung zur Bestimmung eines Fixpunktes ξ ist lokal konvergent, d.m. es gibt eine Umgebung $U(\xi)$ derart, daß für alle Startwerte $x^0 \in U(\xi)$ die durch die Fixpunktiteration (72) erzeugten Iterierten $\{x^i\}_{i\in\mathbb{N}}$ gegen ξ konvergieren.

Beweis: Im Fall $p > 1$ wähle $U(\xi) := \{x; \|x - \xi\| < \delta\}$ so, daß $\|x - \xi\|C^{\frac{1}{p-1}} < \kappa < 1$ für alle $x \in U(\xi)$ gültig ist, also $\delta = C^{-\frac{1}{p-1}}$, wobei C die Konstante in der Definition der Ordnung bezeichnet. Das kann wie folgt eingesehen werden; Für die Iterierten gilt

$$\|x^{i+1} - \xi\| \leq C\|x^i - \xi\|^p \leq \cdots \leq C^{1+p+p^2+\cdots+p^i}\|x^0 - \xi\|^{p^{i+1}} =$$
$$= C^{-\frac{1}{p-1}}\left(C^{\frac{1}{p-1}}\|x^0 - \xi\|\right)^{p^{i+1}} \to 0 \ (i \to \infty),$$

falls $C^{\frac{1}{p-1}}\|x^0 - \xi\| < 1$ und $p > 1$. Im Fall $p = 1$ gilt $C < 1$, also

$$\|x^{i+1} - \xi\| \leq C\|x^i - \xi\| \leq \cdots \leq C^{i+1}\|x^0 - \xi\| \to 0 \ (i \to \infty),$$

so dass Konvergenz bei jeder Wahl des Startwertes vorliegt. □

Wir sprechen im Fall $p = 1$ von **linearer Konvergenz**, und von **superlinearer Konvergenz**, falls

$$\|x^{i+1} - \xi\| \leq C_i\|x^i - \xi\| \text{ für alle } i \geq 0 \text{ mit } \lim_{i\to\infty} C_i = 0.$$

Für skalare Iterationen gilt

Satz 3.5. In (72) gelte $n = 1$. Ferner besitze G den Fixpunkt ξ, sei in einer Umgebung von ξ p-mal stetig differenzierbar und es gelte $G^{(k)}(\xi) = 0$ für $k = 1, \ldots, p-1$, aber $G^{(p)}(\xi) \neq 0$. Dann liegt für $p > 1$ ein Verfahren p-ter Ordnung vor. Ein Verfahren erster Ordnung liegt vor, falls zusätzlich zu $p = 1$ noch $|G'(\xi)| < 1$ gilt.

Der Beweis ist Gegenstand von

Aufgabe 3.6.

1. Beweisen Sie Satz 3.5.
2. Weisen Sie nach, daß das Newton Verfahren für skalare Funktionen $f : \mathbb{R} \to \mathbb{R}$ lokal mindestens von 2ter Ordnung konvergiert, sofern für die entprechende Nullstelle ξ (der Fixpunkt der Verfahrens Funktion) $f'(\xi) \neq 0$ gilt.

Hinreichend für die lokale Konvergenz von Fixpunkt Iterationen ist die Kontraktionseigenschaft der Iterations Funktion G.

Definition 3.7. $G : \mathbb{R}^n \to \mathbb{R}^n$ heißt **stark kontrahierend** in einer Umgebung $U(\zeta) :\Longleftrightarrow$

$$\|G(x) - G(y)\| \leq K\|x - y\| \text{ für alle } x, y \in U(\zeta) \tag{75}$$

mit einem $K < 1$, bzw. **schwach kontrahierend** in einer Umgebung $U(\zeta) :\Longleftrightarrow$

$$\|G(x) - G(y)\| < \|x - y\| \text{ für alle } x, y \in U(\zeta). \tag{76}$$

Kontrahierend meint im Folgenden immer stark kontrahierend.

Die Fixpunkt Iteration einer kontrahierenden Abbildung ist in der Umgebung eines Fixpunktes konvergent mit mindestens Ordnung 1, wie der folgende Satz zeigt.

Satz 3.8. Die Funktion $G : \mathbb{R}^n \to \mathbb{R}^n$ besitzte einen Fixpunkt ξ und sei in einer Umgebung $B_r(\xi) := \{x \in \mathbb{R}^n; \|x - \xi\| < r\}$ kontrahierend. Dann besitzt die Iterierten Folge $\{x^i\}_{i\in\mathbb{N}}$ der Fixpunkt Iteration (72) für jeden Startwert $x^0 \in B_r(\xi)$ die Eigenschaften

- $x^i \in B_r(\xi)$ für alle $i \in \mathbb{N}$,
- $\|x^i - \xi\| \leq K^i\|x^0 - \xi\|$ (was bedeutet, daß $\{x^i\}_{i\in\mathbb{N}}$ mindestens linear gegen ξ konvergiert).

Der Beweis dieses Satzes ist Gegenstand von

Aufgabe 3.9. Beweisen Sie Satz 3.8.

Bei lokal kontrahierenden Abbildungen kann aber auch auf die Existenz eines Fixpunktes geschlossen werden. Neben anderen Dingen besagt das der

Satz 3.10. Sei $x^0 \in \mathbb{R}^n$. Die Funktion $G : \mathbb{R}^n \to \mathbb{R}^n$ sei

i. in einer Umgebung $B_r(x^0) := \{x \in \mathbb{R}^n; \|x - x^0\| < r\}$ kontrahierend mit der Kontraktionskonstanten K,

ii. $\|x^1 - x^0\| = \|G(x^0) - x^0\| \leq (1 - K)r < r$.

Dann gilt für die Iterierten Folge $\{x^i\}_{i\in\mathbb{N}}$ der Fixpunkt Iteration (72) mit Startwert x^0

1. $x^i \in B_r(x^0)$ für alle $i \in \mathbb{N}$,
2. G besitzt in $\bar{B}_r(x^0)$ genau einen Fixpunkt ξ mit $\lim_{x\to\infty} x^i = \xi$ und
3. es gelten die Fehlerabschätzungen

$$\|x^{i+1} - \xi\| \leq K\|x^i - \xi\| \quad \text{und} \quad \|x^i - \xi\| \leq \frac{K^i}{1 - K}\|x^1 - x^0\|. \tag{77}$$

Beweis. 1. Wir zeigen $x^i \in B_r(x^0)$ für alle $i \in \mathbb{N}$ mit Hilfe von Induktion. Aus der Voraussetzung ii. folgt $x^1 \in B_r(x^0)$. Gelte also $x^j \in B_r(x^0)$ für $1 \leq j \leq i$. Aus i. erhalten wir sofort

$$\|x^{i+1} - x^i\| = \|G(x^i) - G(x^{i-1})\| \leq K\|x^i - x^{i-1}\| \leq K^i\|x^1 - x^0\|.$$

Mit Hilfe der Dreiecksungleichung und ii. folgt dann

$$\|x^{i+1} - x^0\| \leq \|x^1 - x^0\| \sum_{j=0}^{i} K^j \leq r(1-K)\sum_{j=0}^{i} K^j = (1-K^{i+1})r < r,$$

also $x^{i+1} \in B_r(x^0)$.
2. Wir zeigen, daß $\{x^i\}_{i\in\mathbb{N}}$ Cauchy Folge ist. Dazu verwende die Dreiecksungleichung, ii., die geometrische Reihe und schließe für $m > l$

$$\|x^m - x^l\| \leq \|x^1 - x^0\|K^l \sum_{j=0}^{m-l-1} K^j < \frac{K^l}{1-K}\|x^1 - x^0\| < K^l r \to 0 \text{ für } m, l \to \infty.$$

Demnach ist $\{x^i\}_{i\in\mathbb{N}}$ Cauchy Folge und konvergiert wegen $x^i \in B_r(x^0)$ gegen ein $\xi \in \bar{B}_r(x^0)$. Dieses ξ ist Fixpunkt von G, denn

$$\|G(\xi) - \xi\| \leq \|G(\xi) - G(x^i)\| + \|G(x^i) - \xi\| \leq K\|x^i - \xi\| + \|x^{i+1} - \xi\| \to 0 \text{ für } i \to \infty.$$

Der Fixpunkt ist eindeutig, denn ist $\bar{\xi}$ ein weiterer Fixpunkt in $\bar{B}_r(x^0)$. Dann folgt $\bar{\xi} = \xi$ aus

$$\|\bar{\xi} - \xi\| = \|G(\bar{\xi}) - G(\xi)\| \leq K\|\bar{\xi} - \xi\|.$$

Schließlich noch die Fehlerabschätzungen;

$$\|\xi - x^l\| = \lim_{m\to\infty} \|x^m - x^l\| \leq \frac{K^l}{1-K}\|x^1 - x^0\|$$

und

$$\|\xi - x^{i+1}\| = \|G(\xi) - G(x^i)\| \leq K\|\xi - x^i\|.$$

Damit ist alles bewiesen. □

Wir wollen jetzt das Newton Verfahren in seiner Basis Variante aus Algorithmus 3.1 untersuchen und beweisen zunächst

Satz 3.11. Sei $F : S \to \mathbb{R}^n$ stetig und auf der offenen Menge C mit $\bar{C} \subseteq S$ differenzierbar. Ferner gebe es zu $x^0 \in C$ positive Konstanten r, a, b, c, h mit den Eigenschaften

i. $B_r(x^0) = \{x \in \mathbb{R}^n; \|x - x^0\| < r\} \subseteq C$,

ii. $h := \frac{abc}{2} < 1$ und

iii. $r := \frac{a}{1-h}$.

Die Funktion F besitzte die Eigenschaften

1) $\|DF(x) - DF(y)\| \leq c\|x - y\|$ für alle $x, y \in C$ (Lipschitz Stetigkeit der Ableitung),

2) $DF(x)^{-1}$ existiert für alle $x \in C$ und es gilt $\|DF(x)^{-1}\| \leq b$ und

3) $\|DF(x^0)^{-1}F(x^0)\| \leq a$.

Dann gilt

1. Ausgehend von x^0 ist die Folge der Newton Iterierten $\{x^i\}_{i\in\mathbb{N}}$ aus Algorithmus 3.1 wohldefiniert und es gilt $x^i \in B_r(x^0)$ für alle $i \in \mathbb{N}$.

2. $\lim_{i\to\infty} x^i = \xi \in \bar{B}_r(x^0)$ und es gilt $F(\xi) = 0$.

3. Es gilt die Fehlerabschätzung

$$\|x^i - \xi\| \leq a\frac{h^{2^i-1}}{1-h^{2^i}} \text{ für alle } i \in \mathbb{N}. \tag{78}$$

4. Das Verfahren ist lokal quadratisch konvergent, d.m. für die Iterierten gilt

$$\|x^{i+1} - \xi\| \leq K\|x^i - \xi\|^2$$

mit einer positiven Konstanten K.

Unter anderem ist das Newton Verfahren wegen $0 < h < 1$ dann mindestens quadratisch konvergent.

Beweis. 1. Wir schreiben Algorithmus 3.1, 2. um in die Form

$$x^{i+1} = x^i - DF(x^i)^{-1}F(x^i).$$

Sind die Iterierten x^j $(j = 0, \ldots, i)$ in $B_r(x^0)$, so ist x^{i+1} wegen Voraussetzung 2) wohldefiniert. Mit Voraussetzung 3) ist diese Aussage richtig für x^0, x^1. Seien also x^j $(j = 0, \ldots, i)$ in $B_r(x^0)$ für ein $i \geq 1$. Dann folgt aus 2) und der Iterationsvorschrift des Newton Verfahrens

$$\|x^{i+1} - x^i\| = \|DF(x^i)^{-1}F(x^i)\| \leq b\|F(x^i)\| = b\|F(x^i) \underbrace{-F(x^{i-1}) - DF(x^{i-1})(x^i - x^{i-1})}_{=0}\|.$$

Der Mittelwertsatz in Integralform liefert $(x := x^i, y := x^{i-1})$

$$F(x) - F(y) - DF(y)(x-y) = \int_0^1 DF(y + s(x-y))ds(x-y) - DF(y)(x-y)$$

$$= \int_0^1 (DF(y + s(x-y)) - DF(y))ds(x-y),$$

also

$$\|F(x) - F(y) - DF(y)(x-y)\| \leq \int_0^1 \|DF(y + s(x-y)) - DF(y)\|ds\|x-y\| \leq c\int_0^1 sds\|x-y\|^2$$

wegen 1) und der Konvexität von C. Damit ergibt sich dann induktiv

$$\|x^{i+1} - x^i\| \leq \frac{cb}{2}\|x^i - x^{i-1}\|^2 \leq ah^{2^i-1}$$

wegen der Definition von h. Dies' wiederum liefert mit Hilfe der Dreiecksungleichung und er geometrischen Reihe

$$\|x^{i+1} - x^0\| \leq \sum_{j=0}^{i} \|x^{j+1} - x^j\| \leq a\sum_{j=0}^{i} h^{2^j-1} < \frac{a}{(1-h)} = r,$$

also $x^{i+1} \in B_r(x^0)$.
2. und 3. Wie in 2. des Beweises von Satz 3.10 wird jetzt bewiesen, daß $\{x^i\}_{i\in\mathbb{N}}$ eine Cauchy Folge ist und daher konvergiert. Es gilt nämlich für $m \geq n$

$$\|x^{m+1} - x^n\| \leq \sum_{j=n}^{m} \|x^{j+1} - x^j\| \leq ah^{2^n-1}\sum_{j=0}^{\infty}\left(h^{2^n}\right)^j < \frac{ah^{2^n-1}}{(1-h^{2^n})} < \epsilon \text{ für } n \geq n_0(\epsilon),$$

weil $0 < h < 1$. Damit gilt

$$\lim_{i\to\infty} x^i = \xi \in \bar{B}_r(x^0)$$

und auch als Folgerung aus der vorangegangenen Abschätzung

$$\lim_{m\to\infty} \|x^{m+1} - x^n\| = \|\xi - x^n\| \leq \frac{ah^{2^n-1}}{(1-h^{2^n})}.$$

Das ist die gewünschte Fehlerabschätzung. Verbleibt der Nachweis, daß ξ Nullstelle von F ist. Dazu schließe aus 1) und $x^1 \in B_r(x^0)$ für alle $i \in \mathbb{N}$

$$\|DF(x^i)\| \leq \|DF(x^0)\| + \|DF(x^i) - DF(x^0)\| \leq \|DF(x^0)\| + c\|x^i - x^0\| < cr + \|DF(x^0)\| =: K.$$

Damit

$$\|F(x^i)\| = \| - DF(x^i)(x^{i+1} - x^i)\| \leq K\|x^{i+1} - x^i\| \leq Kah^{2^i-1} \to 0 \text{ für } i \to \infty.$$

Da F auf S stetig ist gilt auch

$$\lim_{i\to\infty} F(x^i) = F(\xi),$$

also $F(\xi) = 0$.
Wir zeigen noch die quadratische Konvergenz. Dazu schreibe

$$F(x^i) = F(x^i) - F(\xi) = \int_0^1 (DF(\xi + s(x^i - \xi)) - DF(x^i))ds(x^i - \xi) + DF(x^i)(x^i - \xi).$$

Damit ergibt sich

$$x^i - x^{i+1} = DF(x^i)^{-1}F(x^i) = x^i - \xi + DF(x^i)^{-1} \int_0^1 (DF(\xi + s(x^i - \xi)) - DF(x^i))ds(x^i - \xi),$$

also mit der Lipschitz Stetigkeit der Ableitungen

$$\|\xi - x^{i+1}\| \leq \|DF(x^i)^{-1}\| \frac{c}{2}\|\xi - x^i\|^2 \leq \frac{bc}{2}\|\xi - x^i\|^2,$$

das ist die Behauptung mit $K := \frac{bc}{2}$. □

Eine wichtige Eigenschaft der Gleichung (71) ist deren Invarianz gegenüber affinen Transformationen. Denn es gilt ja

$$F(x) = 0 \quad \Longleftrightarrow \quad AF(x) = 0 \text{ für alle Matrizen } A \in GL(n), \tag{79}$$

d.m. Transformationen mit invertierbaren Matrizen ändern die Lösungsmenge nicht. Eine numerische Lösungsmethode sollte diese Invarianz dann auch berücksichtigen, also **konservativ** hinsichtlich der **affinen Invarianz** der Gleichung (71) sein. Das Newton Verfahren erfüllt diese Eigenschaft, denn es gilt

$$H(x) := AF(x) \;\Rightarrow DH(x)^{-1}H(x) = DF(x)^{-1}A^{-1}AF(x) = DF(x)^{-1}F(x).$$

Ein Konvergenz Beweis des Newton Verfahrens sollte daher nur mit Bedingungen formuliert werden, die auch affin invariant sind. In dieser Hinsicht waren wir bei der Formulierung des Satzes 3.11 nicht konsequent, wie das folgende Beispiel zeigt.

Beispiel 3.12. *Betrachte*

$$F(x) := \begin{bmatrix} x_1 - x_2 \\ (x_1 - 8)x_2 \end{bmatrix}$$

mit den beiden Nullstellen

$$x^* = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \text{ und } y^* = \begin{bmatrix} 8 \\ 8 \end{bmatrix}.$$

Wir formulieren unsere Aufgabenstellung (finde x mit $F(x) = 0$) in

$$S := \{x = (x_1, x_2); -1 < x_i < 3, i = 1, 2\} \ni x^*,$$

und schließen damit y^* von der Suche aus. Als Startwert für das Newton Verfahren wählen wir

$$x^0 = \begin{bmatrix} \frac{1}{8} \\ \frac{1}{8} \end{bmatrix}$$

und messen in der Maximum Norm. Dann gilt

- $a_F = \|DF(x^0)^{-1}F(x^0)\| = 0.127$,
- $b_F = \sup_S \|DF(x)^{-1}\| = 3$ *und*
- $c_F = \sup_S \frac{\|DF(x)-DF(y)\|}{\|x-y\|} = 2$.

Das Konvergenzkriterium $h_F = a_F b_F c_F/2 < 1$ ist hier wegen $h = 0.381$ erfüllt. Betrachte jetzt

$$H(x) := \begin{bmatrix} 1 & 1 \\ 0 & \frac{1}{2} \end{bmatrix} F(x).$$

Dann gilt

$$a_H = a_F,\ b_H = 8.5 \text{ und } c_H = 2 \ \Rightarrow\ h_H = 1.0795 > 1.$$

D.m. daß obwohl das Newton Verfahren invariant unter affinen Transformationen ist, gilt das nicht für die theoretische Charakterisierung in Satz 3.11.

Wir formulieren jetzt noch einen lokalen Konvergenzsatz für das Newton Verfahren, der nur affin invariante Voraussetzungen benötigt und zusätzlich auch die lokale Einzigkeit der Nullstelle ξ gewährleistet, siehe [17].

Satz 3.13. (Newton-Kantorovich)
Sei $F : \mathbb{R}^n \supseteq S \to \mathbb{R}^n$ sei stetig differenzierbar, S offen und $x^0 \in S$ mit $DF(x^0)^{-1}$ existent. Ferner nehme an, daß Konstanten $a > 0$, $\omega_0 > 0$ existieren mit den Eigenschaften

i. $\|DF(x^0)^{-1}F(x^0)\| \leq a$,

ii. $\|DF(x^0)^{-1}(DF(x) - DF(y))\| \leq \omega_0\|x - y\|$ für alle $x, y \in S$

iii. $h_0 := a\omega_0 \leq \frac{1}{2}$ und

iv. $\bar{B}_r(x^0) \subset S$, $r := \frac{1-\sqrt{1-2h_0}}{\omega_0}$.

Dann gilt

1. Ausgehend von x^0 ist die Folge der Newton Iterierten $\{x^i\}_{i\in\mathbb{N}}$ aus Algorithmus 3.1 wohldefiniert und es gilt $x^i \in B_r(x^0)$ für alle $i \in \mathbb{N}$.
2. $\lim_{i\to\infty} x^i = \xi \in \bar{B}_r(x^0)$ und es gilt $F(\xi) = 0$.
3. Die Lösung ξ ist eindeutig in
$$\bar{B}_r(x^0).$$

Die Konvergenz der Iteriertenfolge $\{x^i\}_{i\in\mathbb{N}}$ ist mindestens quadratisch.

Beweis. Siehe [17]. □

Aufgabe 3.14. Prüfen Sie für Beispiel 3.12, ob die Voraussetzungen von Satz 3.13 affin invariant sind.

Eine vereinfachte Variante des Newton Verfahrens aus Algorithmus 3.1 verzichtet in Schritt 2. auf die Verwendung jeweils der exakten Jacobi Matrix $DF(x^i)$. Statt dessen wird die Jacobi Matrix aus einem der vorherigen Schritte eingefroren. Es lohnt sich dann, diese Matrix zu zerlegen, da in Schritt 2. (unten Schritt 3.) immer Gleichungssysteme mit derselben Koeffizientenmatrix zu lösen sind.

Algorithmus 3.15. (Vereinfachtes Newton Verfahren)

1. $x^0 \in \mathbb{R}^n$ gegeben, $i = 0$, $iaufdat \geq 1$.
2. Falls $\frac{i}{iaufdat} \in \mathbb{N} \cup \{0\}$ $A := DF(x^i)$
3. löse nach Δx^i,
$$A\Delta x^i = -F(x^i),$$
4. datiere auf
$$x^{i+1} = x^i + \Delta x^i,$$
5. $i = i + 1$, gehe zu 2.

Für diesen Algorithmus gelten lokale Konvergenzaussagen ähnlich denen aus den Sätzen 3.11 und 3.13, die Konvergenzrate ist jedoch i.d.R. nur linear, siehe dazu [17].
Weitere Varianten des Newton Verfahrens sind die sogenannten **Inexakten Newton Verfahren** und **Quasi Newton Verfahren**. Die Idee des inexakten Newton Verfahrens besteht darin, Schritt 2. des Newton Verfahrens 3.1 selbst iterativ zu lösen und die Genauigkeit dieser **inneren Iteration** an das Residuum der äusseren Iteration anzupassen.

Algorithmus 3.16. (Inxaktes Newton Verfahren)

1. $x^0 \in \mathbb{R}^n$ gegeben, $i = 0$, $tol > 0$.
2. $A := DF(x^i)$, $r := \frac{\|F(x^i)\|}{\|F(x^0)\|}$
3. löse iterativ nach Δx^i,
$$A\Delta x^i = -F(x^i),$$
   (a) Δx^i_0 gegeben, $j = 0$
   (b) Solange $\frac{\|A\Delta x^i_j + F(x^i)\|}{\|F(x^0)\|} \geq STOP(r)$ führe aus: $\Delta x^i_{j+1} = Iterate(A, \Delta x^i_j)$
   (c) Resultat ist $\Delta x^i_{j^*}$.
4. $\Delta x^i := \Delta x^i_{j^*}$.
5. datiere auf
$$x^{i+1} = x^i + \Delta x^i,$$
6. $i = i + 1$, gehe zu 2.

Die Konvergenzeigenschaften des inexakten Newton Verfahrens hängen von der Funktion $STOP(r)$ ab. So kann gezeigt werden [8], daß Algorithmus 3.16 lokal

- linear konvergiert, falls $STOP(r) = cr$ mit einem positiven $c < 1$,
- superlinear konvergiert, falls $STOP(r) = r^\alpha$ mit einem $\alpha \in (1, 2)$,
- quadratisch konvergiert, falls $STOP(r) = r^2$.

Die Wahl des Orakels $Iterate(A, z)$ hängt von den Eigenschaften der Matrix A und damit von den Eigenschaften der Jacobi Matrix $DF(x)$ ab. Bei beliebigen regulären Matrizen A kann immer das GMRES Verfahren aus Algorithmus 2.52 verwendet werden. Sind die im Verlauf des Algorithmus auftretenden Jacobi Matrizen positiv definit (was etwa dann wahrscheinlich ist, wenn F das Potential einer skalaren Funktion h darstellt), so ist das vorkonditionierte CG Verfahren aus Algorithmus 2.45 ein Verfahren der Wahl.
Newton Verfahren spielen eine wichtige Rolle bei der numerischen Lösung von Minimierungsproblemen der Form

$$\min_{x \in \mathbb{R}^n} f(x), \tag{80}$$

denn jede Lösung dieses Problems ist bekanntlich ein stationärer Punkt, also eine Nullstelle von $F(x) := \nabla f(x)$.
Andererseits kann ein Nullstellen Problem (71) als Minimierungsproblem formuliert werden. Dazu setze

$$h(x) := \frac{1}{2}\|F(x)\|_2^2, \tag{81}$$

so daß jede Nullstelle von F globales Minimum der Funktion h ist. Wir stellen fest, daß

$$\nabla h(x) = DF(x)^t F(x)$$

erfüllt ist, somit also die Newton Richtung

$$d_N := -DF(x)^{-1}F(x)$$

wegen

$$d_N^t \nabla h(x) = -\|F(x)\|_2^2$$

eine Abstiegsrichung für h darstellt. Das Newton Verfahren aus Algorithmus 3.1 ist dann ein Spezialfall des nachfolgend dargestellten allgemeinen Abstiegsverfahrens für eine skalare Funktion h.

Algorithmus 3.17. (Allgemeines Abstiegsverfahren)

1. $x^0 \in \mathbb{R}^n$ gegeben, $i = 0$.
2. d^i sei Abstiegsrichung,
3. bestimme Schrittweite,
$$\rho^* = \text{argmin}_{\rho \geq 0} h(x^i + \rho d^i),$$
4. datiere auf
$$x^{i+1} = x^i + \rho^* d^i,$$
5. $i = i + 1$, gehe zu 2.

Dieser Algorithmus stimmt mit dem Newton Verfahren für $F(x) = 0$ überein, falls $h(x) = \frac{1}{2}\|F(x)\|_2^2$, d als Newton Richtung und $\rho^* = 1$ in Schritt 3. gewählt wird.
In vielen Anwendungen ist schwierig, die Jacobi Matrix einer Funktion explizit anzugeben. So sind häufig nur Funktionsauswertungen kodiert und Programme zur Berechnung von Ableitungen stehen nicht zur Verfügung. In solchen Fällen kann im Newton Verfahren eine Approximation der Jacobi Matrix $DF(x^i)$ mit Hilfe finiter Differenzen bereitgestellt werden. Das kann allerdings numerisch sehr aufwendig sein (bei *teuren* Funktionsauswertungen) und ist numerisch aufgrund von Auslöschung (siehe etwa [2]) häufig kritisch. Hier stellen **Quasi Newton Verfahren** eine Alternative dar. Wir geben hier ein Verfahren an, das auf Broyden zurück geht.

Algorithmus 3.18. (Quasi Newton Verfahren)

1. $x^0 \in \mathbb{R}^n$ gegeben, $B_0 \in Mat(n,n)$ invertierbar (Approximation an $DF(x^0)$), $i = 0$.
2. $d^i := -B_i^{-1}F(x^i)$,
3. bestimme Schrittweite,
$$\rho_i = \text{argmin}_{\rho \geq 0}\|F(x^i + \rho d^i)\|^2,$$
4. datiere auf
$$x^{i+1} = x^i + \rho_i d^i,$$
5. Modifiziere Quasi Newton Matrix
$$p^i = x^{i+1} - x^i,\ q^i := F(x^{i+1}) - F(x^i), \quad B_{i+1} := B_i + \frac{1}{(p^i)^t p^i}(q^i - B_i p^i)(p^i)^t,$$

6. $i = i + 1$, gehe zu 2.

Die Bestimmung der Schrittweite ρ_i in Schritt 3. dieses Algorithmus' kann mit Biskektion erfolgen;

$$\rho_i := 2^{-k}, \quad k := \min\{j \geq 0; \|F(x^i + 2^{-j}d^i)\| < \|F(x^i)\|\}. \tag{82}$$

Algorithmus 3.18 motiviert sich aus

Satz 3.19. Seien $A, B \in Mat(n,n)$, $b \in \mathbb{R}^n$ und $F(x) := Ax + b$. Es gelte mit $y, y' \in \mathbb{R}^n$

$$p := y' - y, \; q := F(y') - F(y) = Ap \text{ und } B' := B + \frac{1}{p^t p}(q - Bp)p^t.$$

Dann gilt mit der Spektralnorm $\|\cdot\|_2$ (Grenznorm für die Euklidische Norm)

$$\|B' - A\|_2 \leq \|B - A\|_2$$

sowie

$$B'p = Ap = q.$$

Der Beweis ist Gegenstand von

Aufgabe 3.20. Beweisen Sie Satz 3.19.

Bemerkung 3.21. Bei der Modifizierung der Quasi Newton Matrix B zu B' handelt es sich um eine Rang 1 Modifikation, denn $\text{rang}(B - B') \leq 1$. Ist die Inverse von B bekannt, kann die Inverse von B' aus jener von B einfach mit Hilfe der Sherman-Morrison-Woodbury Formel berechnet werden. Diese besagt, daß für reguläres $A \in Mat(n,n)$

$$A + uv^t \text{ regulär ist gdw } \sigma := 1 + v^t A^{-1} u \neq 0.$$

Es gilt dann die Aufdatierungsformel

$$(A + uv^t)^{-1} = A^{-1} - \frac{1}{\sigma} A^{-1} u v^t A^{-1}. \tag{83}$$

Der Beweis ist Übungsaufgabe.

Aus diesem Satz ergibt sich, daß, falls B eine Approximation der Matrix A (in der Spektralnorm) darstellt, die Matrix B' mindestens eine ebenso gute Approximation an A darstellt. Ferner transformieren A und B' den Vektor P in denselben Bildvektor q. Für nichtlineare Funktionen F braucht die Aussage von Satz 3.19 natürlich nicht gelten. Es besteht allerdings die Hoffnung, daß sich die dort formulierten Eigenschaften lokal über die Linearisierung von F auf die Funktion F übertragen. Das ist die Grundidee von Algorithmus 3.18. Einen Beweis für das folgende lokale Konvergenzresultat von Broyden, Dennis und Moré findet sich in [8].

Satz 3.22. (Lokale Konvergenz des Quasi Newton Verfahrens)
Sei ξ Nullstelle von F. Ferner gebe es eine Umgebung $U(\xi)$, so daß

1. F in $U(\xi)$ stetig differenzierbar ist,
2. $\|DF(x) - DF(y)\| \leq L\|x - y\|$ für alle $x, y \in U(\xi)$ und
3. $DF(\xi)^{-1}$ existiert.

In Algorithmus 3.18 werde in Schritt 3. immer $\rho_i = 1$ gesetzt. Dann gibt es Umgebungen $B_r(\xi)$ und $B_s(DF(\xi))$, so daß für jede Wahl von $x^0 \in B_r(\xi) \cap U(\xi)$ und $B_0 \in B_s(DF(\xi))$ die Iterierten Folge $\{x^i\}_{i \in \mathbb{N}}$ von Algorithmus 3.18 superlinear gegen ξ konvergiert.

4 Interpolation

Interpolation von Funktionen und allgemeiner Daten ist ein häufig auftretendes Problem sowohl in der Mathematik als auch in vielen Anwendungen.
Dateninterpolation: Gegen seien Daten (x_i, f_i) $(i = 0, \ldots, n)$. Gesucht ist eine Funktion $p(x)$ mit

$$p(x_i) = f_i \quad i = 0, \ldots, n. \tag{84}$$

Ein wichtiger Spezialfall liegt vor, falls die f_i selber als Funktionswerte einer Funktion $f : \mathbb{R} \to \mathbb{R}$ auf **Stützstellen** $x_0, \ldots, x_n$ gegeben sind, d.m. $f_i = f(x_i)$ $(i = 0, \ldots, n)$. Der Sinn dieser Aufgabenstellung liegt darin, daß die Funktion f selber schwer, die Funktion p hingegen leicht auszuwerten ist.
Interpolieren können wir auf mannigfaltige Weise. Wir fangen an mit

4.1 Polynominterpolation

Wir wollen das Problem der Dateninterpolation lösen mit Polynomen, d.m. Funktionen der Form

$$p(x) := \sum_{i=0}^{m} a_i x^i,$$

wobei die Koeffizienten $a_0, \ldots, a_m$ zu bestimmen sind und m den Grad des Polynoms bezeichnet. Es gilt

Satz 4.1. Seien Daten (x_i, f_i) $(i = 0, \ldots, n)$ gegeben mit $x_i \neq x_j$ für $i \neq j$. Dann gibt es genau ein **Interpolationspolynom** p vom Grade n mit

$$p(x_i) = f_i, \quad i = 0, \ldots, n.$$

Beweis. Definiere zu x_i das **Lagrange Polynom** n-ten Grades gemäß

$$L_i(x) := \prod_{\substack{j=0 \\ x_j \neq x_i}}^{n} \frac{x - x_j}{x_i - x_j}. \tag{85}$$

Definiere

$$p(x) := \sum_{i=0}^{n} f_i L_i(x).$$

Dann ist p ein Polynom n-ten Grades und es gilt $p(x_i) = f_i$ für $i = 0, \ldots, n$. Das sichert die Existenz des Polynoms. Gibt es Polynome $p_1 \neq p_2$ welche beide die Interpolationsaufgabe lösen, so ist $q := p_1 - p_2$ ein Polynom höchstens vom Grade n mit $n + 1$ Nullstellen $x_0, \ldots, x_n$, muß also identisch verschwinden. Das sichert auch die Eindeutigkeit von p. □

Die Auswertung des Interpolationspolynoms p in Lagrange Darstellung an einer Stelle x benötigt $O(n^2)$ mathematische Operationen. Ferner sollten die Nenner der Lagrange Polynome vorab berechnet werden, damit diese nicht bei jeder Auswertung neu berechnet werden müssen. Die Lagrange Darstellung von p ist nützlich immer dann, wenn nachträglich Meßwerte f_i geändert werden, weil sich die Darstellung von p dann nicht ändert. Sollen allerdings Datenpunkte hinzu gefügt werden, ist diese Darstellung unpraktisch, weil dann alle Lagrange Polynome neu berechnet werden müssen. Günstiger ist dann die sogenannte **Newton Darstellung** des Interpolationspolynoms p. Dazu benötigen wir

Definition 4.2. (Devidierte Differenzen)
Zu x_i, f_i $(i = 0, \ldots, n)$ mit $x_i \neq x_j$ definieren wir die **Dividierten Differenzen** gemäß

$$f_{[x_i]} := f_i, \text{ und rekursiv } f_{[x_l, \ldots, x_{l+k}]} := \frac{f_{[x_{l+1}, \ldots, x_{l+k}]} - f_{[x_l, \ldots, x_{l+k-1}]}}{x_{l+k} - x_l} \text{ für } 0 \leq l < l + k \leq n.$$

Damit gilt

Satz 4.3. Seien Daten (x_i, f_i) $(i = 0, \ldots, n)$ gegeben mit $x_i \neq x_j$ für $i \neq j$. Dann besitzt das **Interpolationspolynom** p vom Grade n mit

$$p(x_i) = f_i, \quad i = 0, \ldots, n.$$

die Darstellung

$$\begin{aligned}(86)\quad p(x) &= f_{[x_0]} + f_{[x_0,x_1]}(x - x_0) + \cdots + f_{[x_0,x_1,\ldots,x_n]}(x - x_0)(x - x_1)\ldots(x - x_{n-1}) \\ &=: F_0 + F_1(x - x_0) + \cdots + F_n(x - x_0)(x - x_1)\ldots(x - x_{n-1}).\end{aligned}$$

Der Beweis dieser Aussage ist Gegenstand von

Aufgabe 4.4. Beweisen Sie Satz 4.3.

Die Rekursion aus Definition 4.2 liefert den folgenden Algorithmus zur Berechnung der dividierten Differenzen.

Algorithmus 4.5. (Berechnung der dividierten Differenzen)
Gegeben seien die Daten $(x_i, f_i), i = 0, \ldots, n$. Berechnet werden die dividierten Differenzen F_i in der Darstellung (86) des Newton Interpolationspolynoms.

Für $i = 0, 1, \ldots, n$ setze $h_i := f_i$

Für $j = i - 1, \ldots, 0$ (rückwärts zählen) berechne $h_j := \frac{h_{j+1} - h_j}{x_i - x_j}$

Setze $F_i := h_0$

Der numerische Aufwand in diesem Algorithmus beträgt $O(n^2)$. Das Interpolationspolynom (86) kann jetzt mit Hilfe des **Horner Schemas** an jeder beliebigen Stelle sehr effizient ausgewertet werden.

Algorithmus 4.6. (Horner Schema zur Auswertung von $p(x)$)

$p := F_n$,

Für $i = n - 1, \ldots, 0$ führe aus: $p := p(x - x_i) + F_i$.

Das Interpolationspolynom vom Grade n zu den Daten (x_i, f_i) ist eindeutig bestimmt, weswegen jede beliebige Anordnung der Daten in Algorithmus 4.5 zu demselben Ergebnis führen muß. Weitere Eigenschaften dividierter Differenzen sind zusammengefaßt im nachfolgenden Satz, dessen Beweis etwa in [2] nachzulesen ist.

Satz 4.7. (Eigenschaften dividierter Differenzen)

- $f_{[x_0,\ldots,x_l]} = f_{[x_{i_0},\ldots,x_{i_l}]}$ für jede Permutation $(i_0, \ldots, i_l)$ von $(0, \ldots, l)$.
- Sei f n-mal stetig differenzierbar und $f_i := f(x_i)$. Dann gilt

  $$f_{[x_0,\ldots,x_n]} = \frac{f^{(n)}(\xi)}{n!} \text{ für ein } \xi \in I(x_0, \ldots, x_n),$$

  wobei $I(x_0, \ldots, x_n)$ das kleinste Intervall bezeichnet, welches $x_0, \ldots, x_n$ enthält.
- $f_{[x_l,\ldots,x_{l+k}]} = \frac{f^{(l)}(x_l)}{l!}$ falls $x_l = \cdots = x_{l+k}$. Siehe auch Hermite Interpolation!

Wir wollen uns jetzt um den Interpolationsfehler kümmern. Die Aufgabe bestehe darin, eine gegebene Funktion $f : [a, b] \to \mathbb{R}$ an Stützstellen $x_0, \ldots, x_n \in [a, b]$ mit einem Interpolationspolynom $p \in \Pi_n$ (Raum der Polynome mit Grad kleiner oder gleich n) zu interpolieren. Es soll also gelten:

$$p(x_i) = f(x_i) \text{ für } i = 0, \ldots, n.$$

Was können wir über den Fehler

$$\max_{x \in [a,b]} |p(x) - f(x)|$$

aussagen? Zunächst ergibt sich

Satz 4.8. Die Funktion $f : [a,b] \to \mathbb{R}$ sei $(n+1)$-mal stetig differenzierbar und $p \in \Pi_n$ bezeichne das eindeutig bestimmte Interpolationspolynom zu den Daten $(x_0, f(x_0)), \ldots, (x_n, f(x_n))$. Dann gilt mit

$$\omega(x) := \prod_{i=0}^{n} (x - x_i)$$

für jedes $\bar{x} \in [a,b]$ mit einem $\xi = \xi(\bar{x}) \in I(x_0, \ldots, x_n, \bar{x})$ die Fehlerdarstellung

$$f(\bar{x}) - p(\bar{x}) = \omega(\bar{x}) \frac{f^{(n+1)}(\xi)}{(n+1)!}. \tag{87}$$

Beweis. Sei $\bar{x} \in [a,b]$, $\bar{x} \neq x_i$ für alle $i = 0, \ldots, n$ (denn sonst ist nichts zu zeigen). Setze

$$H(x) := f(x) - p(x) - K\omega(x)$$

und bestimme die Konstante K so, daß $H(\bar{x}) = 0$. Ist $H(\bar{x}) = 0$, so besitzt H in $I(x_0, \ldots, x_n, \bar{x})$ die $n+2$ Nullstellen $x_0, \ldots, x_n, \bar{x}$, H' dort nach dem Satz von Rolle mindestens $n+1$ Nullstellen und schließlich $H^{(n+1)}$ mindestens eine Nullstelle $\xi \in I(x_0, \ldots, x_n, \bar{x})$. Wegen $p^{(n+1)} \equiv 0$ ergibt sich

$$H^{(n+1)}(\xi) = f^{(n+1)}(\xi) - K(n+1)!,$$

wobei wir benutzt haben, daß der Koeffizient vor der höchstens Potenz von $\omega(x)$ die Eins ist. Damit ergibt sich

$$K = \frac{f^{(n+1)}(\xi)}{(n+1)!}$$

und somit die Behauptung. □

Wir erhalten für den Fehler in der Maximum Norm

Folgerung 4.9. Die Funktion $f : [a,b] \to \mathbb{R}$ sei $(n+1)$-mal stetig differenzierbar und $p \in \Pi_n$ bezeichne das eindeutig bestimmte Interpolationspolynom zu den Daten $(x_0, f(x_0)), \ldots, (x_n, f(x_n))$, wobei Δ: $a \leq x_0 < x_1 < \cdots < x_n \leq b$ eine **Zerlegung** von $[a,b]$ bezeichne. Dann gilt für den **Interpolationsfehler**

$$\|f - p\|_\infty := \max_{x \in [a,b]} |f(x) - p(x)| \leq \frac{\|f^{(n+1)}\|_\infty (b-a)^{n+1}}{(n+1)!}. \tag{88}$$

Leider kann **nicht** geschlossen werden, daß für feiner werdende Unterteilungen des Intervalls $[a,b]$ die zugehörigen Interpolationspolynome einer stetigen Funktion f gleichmäßig gegen die Funktion konvergieren, wie ein Resultat von Faber [13, Theorem 1.19] zeigt.
Die Wahl der Stützstellen ist bei der Interpolationsaufgabe noch frei. Wir können diese so wählen, daß der **Interpolationsfehler** möglichst gering ausfällt. Genaue Inspektion von (87) führt uns auf das Problem, eine Zerlegung Δ^* zu finden derart, daß Δ^* die MinMax Aufgabe

$$\min_{\Delta: a \leq x_0 < \cdots < x_n \leq b} \max_{x \in [a,b]} |\omega(x)| \quad \left(= \frac{(b-a)^{n+1}}{2 \cdot 4^n} \text{ (Nachweis!)} \right) \tag{89}$$

löst. Für das Intervall $[-1,1]$ ist die optimale Zerlegung Δ^* durch die Nullstellen des $(n+1)$-ten **Tschebyscheff Polynoms** gegeben [13, Theorem 1.24]. Diese Polynome sind wie folgt definiert;

Definition 4.10. Tschebyscheff Polynome sind rekursiv definiert durch die Vorschrift

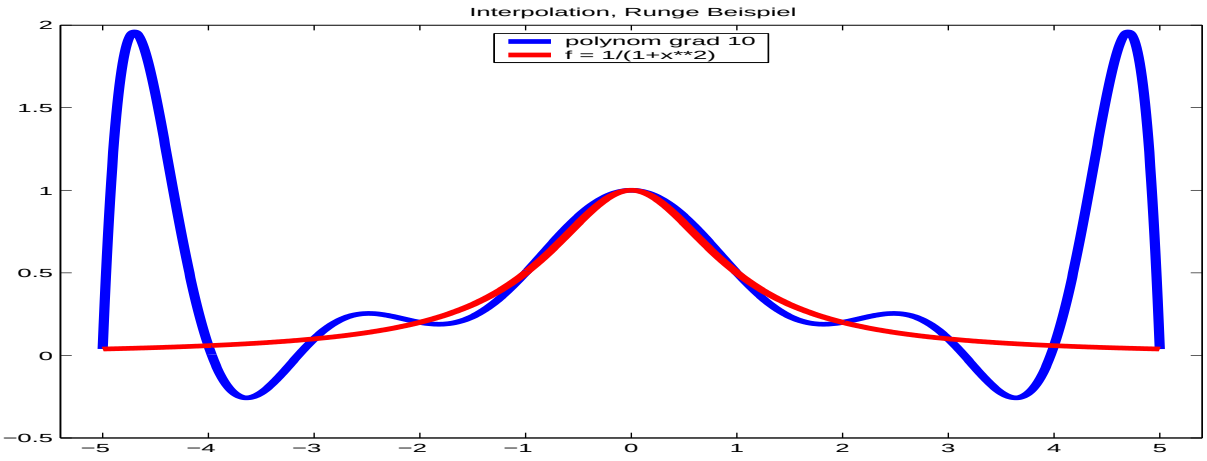
$$T_0(x) := 1, \; T_1(x) := x \text{ und } T_{n+1}(x) := 2xT_n(x) - T_{n-1}(x) \text{ für } n \geq 1.$$

Für sie gilt die geschlossene Darstellung

$$T_n(x) = \cos(n \arccos x) \text{ für } x \in [-1,1], \; n = 0, 1, \ldots, \tag{90}$$

wovon Sie sich leicht selbst mittels Induktion überzeugen können. Ferner besitzt T_n die Nullstellen

$$t_j^{(n)} = \cos \frac{(2j-1)\pi}{2n} \; (j = 1, \ldots, n)$$



und bei

$$s_j^{(n)} = \cos\frac{j\pi}{n} \ (j = 0, \dots, n)$$

Extremstellen.
Die optimale Zerlegung für beliebige Intervalle $[a, b]$ wird durch die Anwendung der linearen Transformation $\chi : [-1, 1] \to [a, b]$, $t \mapsto \frac{1}{2}((b-a)t + a + b)$ auf die Nullstellen von T_{n+1} erhalten [13, Theorem 1.25].
Wie das Beispiel von Runge in Fig. 4.1 eindrucksvoll belegt, ist ein Interpolationspolynom i.d.R. nicht als Modell einer Funktion geeignet. Polynom Interpolation funktioniert allerdings gut auf kleinen Intervallen, wie die Fehlerabschätzung (88) belegt. Daher hat sich in praktischen Aufgabenstellungen ein stückweises Interpolationskonzept durchgesetzt, die sogannte **Spline Interpolation**.

4.2 Spline Interpolation

Die Interpolationsaufgabe bestehe wieder darin, vorgelegte Daten $(x_i, f_i := f(x_i))$ $(i = 0, \dots, n+1)$ zu interpolieren und damit ein geeignetes Modell für die Funktion $f : [a, b] \to \mathbb{R}$ zu erhalten. Wir geben uns ein Gitter

$$\Delta : a = x_0 < x_1 < \cdots < x_{n+1} = b$$

vor, definieren $\|\Delta\| := \max_i h_i$, $h_i := x_i - x_{i-1}$ $(i = 1, \dots, n+1)$ und verlangen jetzt von unserer interpolierenden Funktion S_Δ, daß sie ein **interpolierender Spline k-ter Ordnung** ist.

Definition 4.11. Sei $\Delta : a = x_0 < x_1 < \cdots < x_{n+1} = b$ eine Zerlegung des Intervalls $[a, b]$. Eine Funktion $S_{\Delta,k} : [a, b] \to \mathbb{R}$ heißt **interpolierender Spline k-ter Ordnung** (zur Zerlegung Δ und Daten $(x_i, f_i := f(x_i))$ $(i = 0, \dots, n+1)$), falls

$S_{\Delta,k}(x_i) = f_i$ für $i = 0, \dots, n+1$ und

$S_{\Delta,k}|_{[x_i, x_{i+1}]} \in \Pi_k$, $S_\Delta \in C^{k-1}([a, b])$(für $k \geq 1$).

Wir wollen uns zwei Beispiele anschauen.

Beispiel 4.12. *Sei $(x_i, f_i := f(x_i))$ $(i = 0, \dots, n+1)$ ein Datensatz. Der interpolierende Spline 0-ter Ordnung ist gegeben durch*

$$S_{\Delta,0}(x) := f_i, \quad x \in [x_i, x_{i+1}), \ (i = 0, \dots, n) \ \text{ und } S_{\Delta,0}(x_{n+1}) := f_{n+1}.$$

Der interpolierende **lineare Spline** *(= interpolierender Spline erster Ordnung) ist gegben durch*

$$S_{\Delta,1}(x) := \sum_{i=0}^{n+1} f_i b_i(x), \tag{91}$$

wobei die stückweise linearen, auf $[a,b]$ *global stetigen Funktionen* $b_i : [a,b] \to \mathbb{R}$ *definiert sind durch*

$$(92)\quad b_i(x) := \begin{cases} \frac{x-x_{i-1}}{x_i-x_{i-1}} & x \in [x_{i-1}, x_i) \\ \frac{x_{i+1}-x}{x_{i+1}-x_i} & x \in [x_i, x_{i+1}) \\ 0 & \textit{sonst} \end{cases} \quad \textit{für } i = 1, \ldots, n,$$

$$b_0(x) := \begin{cases} \frac{x_1-x}{x_1-x_0} & x \in [x_0, x_1) \\ 0 & \textit{sonst} \end{cases}, \quad b_{n+1}(x) := \begin{cases} \frac{x-x_n}{x_{n+1}-x_n} & x \in [x_n, x_{n+1}) \\ 0 & \textit{sonst} \end{cases}$$

Interpolierende Splines dritter Ordnung heißen

4.2.1 Kubische Splines

Wir wollen uns überlegen, wie wir **Kubische Splines** S_Δ numerisch berechnen können (wir lassen die 3 als Index der Einfachheit halber fort). Dazu stellen wir fest, daß mit $h_i := x_i - x_{i-1}$ und $S''_\Delta(x_i) =: M_i$ (den sogenannten **Momenten**)

$$S''_{\Delta_{|_{[x_{i-1},x_i]}}}(x) = M_i \frac{x - x_{i-1}}{h_i} + M_{i-1} \frac{x_i - x}{h_i}$$

gültig ist. Integration liefert sofort

$$S'_{\Delta_{|_{[x_{i-1},x_i]}}}(x) = M_i \frac{(x - x_{i-1})^2}{2h_i} - M_{i-1} \frac{(x_i - x)^2}{2h_i} + A_i$$

und

$$S_{\Delta_{|_{[x_{i-1},x_i]}}}(x) = M_i \frac{(x - x_{i-1})^3}{6h_i} + M_{i-1} \frac{(x_i - x)^3}{6h_i} + A_i(x - x_{i-1}) + B_i$$

mit geeigneten Konstanten A_i, B_i. Diese Konstanten können jetzt durch die Momenten M_i, M_{i-1} und die Daten f_i, f_{i-1} ausgedrückt werden. Nach leichter Rechnung ergibt sich

$$B_i = f_{i-1} - M_{i-1}\frac{h_i^2}{6} \text{ und } A_i = \frac{f_i - f_{i-1}}{h_i} - \frac{h_i}{6}(M_i - M_{i-1})$$

Damit ist $S_{\Delta_{|_{[x_{i-1},x_i]}}}$ durch M_{i-1}, M_i bestimmt, S_Δ demnach durch $M_0, \ldots, M_{n+1}$. Wie können wir jetzt die Momenten bestimmen? Dazu erinnern wir uns, daß $S_\Delta \in C^2$ gilt und daher auf den inneren Gitterpunkten $x_1, \ldots, x_n$ sicher

$$\lim_{x \nearrow x_i} S'_\Delta(x) = M_i \frac{h_i}{2} + A_i = -M_i \frac{h_{i+1}}{2} + A_{i+1} = \lim_{x \searrow x_i} S'_\Delta(x)$$

richtig ist. Es ergibt sich sofort mit

$$A_i = \frac{f_i - f_{i-1}}{h_i} - \frac{h_i}{6}(M_i - M_{i-1}) \text{ und } A_{i+1} = \frac{f_{i+1} - f_i}{h_{i+1}} - \frac{h_{i+1}}{6}(M_{i+1} - M_i)$$

das System von Gleichungen

$$(93)\qquad M_{i-1}\frac{h_i}{6} + M_i \frac{h_i + h_{i+1}}{3} + M_{i+1}\frac{h_{i+1}}{6} = \frac{f_{i+1} - f_i}{h_{i+1}} - \frac{f_i - f_{i-1}}{h_i} \text{ für } i = 1, \ldots, n.$$

Das sind n Gleichungen für $n+2$ Unbekannte $M_0, \ldots, M_{n+1}$. Die Restlichen zwei Bedingungen werden i.d.R. wie folgt gestellt:

i. Natürlicher Spline: $S''_\Delta(a) = 0 = S''_\Delta(b)$, d.h. $M_0 = 0 = M_{n+1}$.

ii. Periodischer Spline: $S''_\Delta(a) = S''_\Delta(b)$ $(\Rightarrow M_0 = M_{n+1})$ und $S'_\Delta(a) = S'_\Delta(b)$.

iii. $S'_\Delta(a) = f'_0$ und $S'_\Delta(b) = f'_{n+1}$.

Damit ist das Interpolationsproblem für kubische Splines gelöst. Wir wollen abschließend oben formulierte Bedingungen in Form von linearen Gleichungssystemen formulieren. Dazu schreiben wir (93) um in

$$\mu_i M_{i-1} + 2M_i + \lambda_i M_{i+1} = d_i \text{ für } i = 1, \dots, n, \tag{94}$$

wobei

$$\lambda_i := \frac{h_{i+1}}{h_{i+1}+h_i},\ \mu_i := 1 - \lambda_i \text{ und } d_i := \frac{6}{h_{i+1}+h_i}\left(\frac{f_{i+1}-f_i}{h_{i+1}} - \frac{f_i - f_{i-1}}{h_i}\right).$$

Wir setzen noch in den Fällen

i. Natürlicher Spline: $\lambda_0 := 0$, $d_0 := 0$, $\mu_{n+1} := 0$ und $d_{n+1} := 0$,

ii. Periodischer Spline: $\lambda_{n+1} := \frac{h_1}{h_{n+1}+h_1}$, $\mu_{n+1} := 1-\lambda_{n+1} = \frac{h_{n+1}}{h_{n+1}+h_1}$ und $d_{n+1} := \frac{6}{h_{n+1}+h_1}\left(\frac{f_1-f_{n+1}}{h_1} - \frac{f_{n+1}-f_n}{h_{n+1}}\right)$ (beachte, daß $f_0 = f_{n+1}$ wegen Periodizität), und

iii. $\lambda_0 := 1$, $d_0 := \frac{6}{h_1}\left(\frac{f_1-f_0}{h_1} - f_0'\right)$, $\mu_{n+1} := 1$ und $d_{n+1} := \frac{6}{h_{n+1}}\left(f_{n+1}' - \frac{f_{n+1}-f_n}{h_{n+1}}\right)$

und erhalten für die Fälle i. und iii. das lineare Gleichungssystem

$$\begin{bmatrix} 2 & \lambda_0 & & & 0 \\ \mu_1 & 2 & \lambda_1 & & \\ & \mu_2 & \ddots & \ddots & \\ & & \ddots & 2 & \lambda_n \\ 0 & & & \mu_{n+1} & 2 \end{bmatrix} \begin{bmatrix} M_0 \\ M_1 \\ \vdots \\ M_n \\ M_{n+1} \end{bmatrix} = \begin{bmatrix} d_0 \\ d_1 \\ \vdots \\ d_n \\ d_{n+1} \end{bmatrix}, \tag{95}$$

bzw. für den Fall ii.

$$\begin{bmatrix} 2 & \lambda_1 & & & \mu_1 \\ \mu_2 & 2 & \lambda_2 & & \\ & \mu_3 & \ddots & \ddots & \\ & & \ddots & 2 & \lambda_n \\ \lambda_{n+1} & & & \mu_{n+1} & 2 \end{bmatrix} \begin{bmatrix} M_1 \\ M_2 \\ \vdots \\ M_n \\ M_{n+1} \end{bmatrix} = \begin{bmatrix} d_1 \\ d_2 \\ \vdots \\ d_n \\ d_{n+1} \end{bmatrix}. \tag{96}$$

Damit ist das Problem der kubischen Spline Interpolation gelöst, denn es gilt der Satz

Satz 4.13. Die Gleichungssysteme (95) und (96) sind für jede Zerlegung $\Delta : a = x_0 < x_1 < \cdots < x_{n+1} = b$ des Intervalls $[a, b]$ eindeutig lösbar.

Beweis. Wir betrachten i. und iii., der Fall ii. wird analog behandelt. Sei

$$A := \begin{bmatrix} 2 & \lambda_0 & & & 0 \\ \mu_1 & 2 & \lambda_1 & & \\ & \mu_2 & \ddots & \ddots & \\ & & \ddots & 2 & \lambda_n \\ 0 & & & \mu_{n+1} & 2 \end{bmatrix} \in \mathrm{MAT}(n+2, n+2).$$

Wir zeigen

$$Az = w \Longrightarrow \|z\|_\infty \le \|w\|_\infty,$$

denn mit dieser Eigenschaft würde aus $Az = 0$ sofort $z = 0$, also die Injektivität des der Matrix A zugeordneten Endomorphismus' folgen. Sei also $\|z\|_\infty = |z_r|$. Dann gilt sicher wegen $\lambda_r + \mu_r = 1$ ($\mu_0 = 0 = \lambda_{n+1}$)

$$\|w\|_\infty \ge |w_r| \ge 2|z_r| - \mu_r|z_{r-1}| - \lambda_r|z_{r+1}| \ge 2|z_r| - \mu_r|z_r| - \lambda_r|z_r| \ge (2 - \mu_r - \lambda_r)|z_r| = \|z\|_\infty.$$

□

Bemerkung 4.14.

Analoges Vorgehen beweist, daß **strikt diagonaldominante** *Matrizen regulär sind. Dabei heißt* $A \in Mat(n,n)$ **strikt diagonaldominant**, *falls* A *das starke Zeilensummenkriterium* (40) *erfüllt. Die System Matrizen aus* (95) *und* (96) *sind offensichtlich strikt diagonaldominant.*

Die Gleichungssysteme in (95) *und* (96) *sind mit* $O(n)$ *arithmetische Operationen auflösbar. Im periodischen Fall ii. wird die Lösung des Gleichungssystems* (96) *mit Hilfe der* **Sherman-Morrison-Woodbury Formel** (83) *leicht auf die Lösung von zwei tridiagonalen Gleichungssystemen zurück geführt.*

Kubische Spline Interpolierende verdienen ihren Namen, denn es gilt mit

$$h_{\max} := \max_{i=1,\dots,n+1} h_i (= \|\Delta\|) \text{ und } h_{\min} := \min_{i=1,\dots,n+1} h_i$$

Satz 4.15. Sei $f \in C^4([a,b])$. Die kubischen Spline Interpolierenden S_Δ zu den Daten $(x_i, f_i := f(x_i))$ $(i = 0,\dots,n+1)$ und den Forderungen iii. (Hermite Splines) erfüllen die Fehlerabschätzungen

$$\|f^{(l)} - S_\Delta^{(l)}\|_\infty \le C \frac{h_{\max}}{h_{\min}} h_{\max}^{4-l} \|f^{(4)}\|_\infty \text{ für } l = 0,1,2,3, \tag{97}$$

wobei C eine positive Konstante bezeichnet, die nicht von $h_{\max}$ und $h_{\min}$ abhängt.

Einen Beweis dieser Aussage finden Sie etwa in [2, (2.4.3.3) Satz].

Bemerkung 4.16. *Spline Funktionen mit Eigenschaft i., ii. und* f *periodisch oder iii. mit* $f_0' = f'(a)$ *und* $f_{n+1}' = f'(b)$ *haben noch die schöne Eigenschaft, daß sie unter allen 2 mal stetig differenzierbaren Funktionen* g *mit* $g(x_i) = f_i$ $(i = 0,\dots,n+1)$ *diejenigen mit kleinster Gesamtkrümmung* $\int_a^b |g''(x)|^2 dx$ *sind, d.m.*

$$\int_a^b |S_\Delta''(x)|^2 dx \le \int_a^b |g''(x)|^2 dx \text{ für alle Funktionen } g \in C^2([a,b]) \text{ mit } g(x_i) = f_i (i = 0,\dots,n+1).$$

vergleiche Aufgabenblatt 8. Dazu sei bemerkt, daß $g''(x)$ *die exakte Krümmung* $\frac{g''(x)}{\sqrt{1+g'(x)^2}^3}$ *für kleine Ableitungen* $g'(x)$ *gut approximiert.*

Jetzt noch kurz Interpolation mit Splines k-ter Ordnung.

4.2.2 Splines k-ter Ordnung

Wir ergänzen zunächst unsere Zerlegung Δ wie folgt um Hilfsgitterpunkte;

$$x_{-k} < \dots x_{-1} < a = x_0 < x_1 < \dots < x_{n+1} = b < x_{n+2} < \dots < x_{n+k+1}$$

und defineren für $i = -k,\dots,n$ zu x_i insgesamt $n+k+1$ Funktionen $B_{i,k}$ wie folgt;

$$B_{i,0}(x) := \begin{cases} 1, & x \in [x_i, x_{i+1}) \\ 0, & \text{sonst} \end{cases} \quad \text{für } i = -k,\dots,n+k,$$

und rekursiv

$$B_{i,l}(x) := \frac{x - x_i}{x_{i+l} - x_i} B_{i,l-1}(x) + \frac{x_{i+l+1} - x}{x_{i+l+1} - x_{i+1}} B_{i+1,l-1}(x) \text{ für } l = 1,\dots k \text{ und } i = -k,\dots,n. \tag{98}$$

Die Funktionen $B_{i,k}$ heißen **B-Splines k-ter Ordnung** zur Zerlegung Δ. Es sei bereits hier bemerkt, daß die obige Konstruktion der B-Splines genau auf unsere Interpolationsaufgabe zugeschnitten ist. **B-Splines k-ter Ordnung** können auch ohne den Interpolationskontext definiert werden, siehe etwa [2, Kapitel 2.4.4].
Einige Eigenschaften der Funktionen $B_{i,k}$ sind zusammengefaßt in

Satz 4.17. Die in (98) definierten Funktionen $B_{i,k}$

sind nicht negativ,

sind linear unabhängig,

bilden eine Partition der 1, d.m. $\sum_i B_{i,k}(x) = 1$,

haben Träger (=Bereich, wo Funktion $\neq 0$ ist) $[x_i, x_{i+k+1}]$,

sind eingeschränkt auf $[x_i, x_{i+1}]$ Polynome vom Grad k und

sind $k-1$ mal stetig differenzierbar,

dessen Beweis Ihnen als Übungsaufgabe überlassen wird. Siehe dazu auch [2, (2.4.4.5) Satz]. Die letzten beiden Eigenschaften besagen, daß es sich bei den Funktionen $B_{i,k}$ um Splines k-ter Ordnung handelt, vergleiche Definition 4.11.
Wir sind nach diesen Vorbereitungen in der Lage, das Interpolationsproblem zu den Daten (x_i, f_i) $(i = 0, \ldots, n+1)$ mit Splines k-ter Ordnung zu lösen. Dazu machen wir für den interpolierenden Spline k-ter Ordnung den Ansatz

$$S_{\Delta,k}(x) := \sum_{i=0}^{n+1} \alpha_i B_{i-1,k}(x)$$

und fordern

$$S_{\Delta,k}(x_i) = f_i \text{ für } i = 0, \ldots, n+1.$$

In Matrix Schreibweise erhalten wir das $(n+2) \times (n+2)$ Gleichungssystem

$$\begin{bmatrix} B_{-1,k}(x_0) & \ldots & B_{n,k}(x_0) \\ \vdots & & \vdots \\ B_{-1,k}(x_{n+1}) & \ldots & B_{n,k}(x_{n+1}) \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \vdots \\ \alpha_{n+1} \end{bmatrix} = \begin{bmatrix} f_0 \\ \vdots \\ f_{n+1} \end{bmatrix} \iff A\alpha = f. \tag{99}$$

Wir bemerken sofort, daß die Koeffizientenmatrix A Bandstruktur mit Bandbreite $k-1$ aufweist $(k \geq 1)$. Darüber hinaus ist die Matrix A regulär, das Interpolationsproblem in unserer Formulierung demnach eindeutig lösbar. Schließlich ergibt einfaches Vergleichen, daß mit den **Hütchen Funktionen** aus (92) die Beziehung

$$b_i = B_{i-1,1}$$

erfüllt wird, die Interpolationsaufgabe für lineare Splines also tatsächlich (91) liefert.
Abschließend sei bemerkt, daß die Interpolationsaufgabe auch für Stützstellen $a \leq \xi_0 < \cdots < \xi_{n+1} \leq b$ zu Stützwerten $f_0, \ldots, f_{n+1}$ formuliert werden kann. Das entsprechende Gleichungssystem verändert sich dann nur marginal;

$$\begin{bmatrix} B_{-1,k}(\xi_0) & \ldots & B_{n,k}(\xi_0) \\ \vdots & & \vdots \\ B_{-1,k}(\xi_{n+1}) & \ldots & B_{n,k}(\xi_{n+1}) \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \vdots \\ \alpha_{n+1} \end{bmatrix} = \begin{bmatrix} f_0 \\ \vdots \\ f_{n+1} \end{bmatrix}. \tag{100}$$

Nach [2, (2.4.5.7) Satz] besitzt (100) genau eine Lösung, wenn für die Diagonalelemente $B_{i-1,k}(\xi_i) \neq 0$ für $i = 0, \ldots, n+1$ erfüllt ist.

5 Numerische Integration

Die exakte Berechnung bestimmter Integrale

$$\int_a^b f(x)dx$$

ist nur in Ausnahmesituationen möglich, etwa dann, wenn eine Stammfunktion der zu integrierenden Funktion bekannt ist. Das allerdings ist nicht der Regelfall. Bestimmte Integrale können aber auch numerisch näherungsweise berechnet werden. Elementare Integrationsformeln werden erhalten, wenn der Integrand f durch ein Interpolationspolynom p ersetzt, dessen Integration dann exakt möglich ist. Diese Vorgehensweise ist nach *Newton-Cotes* benannt.

5.1 Newton-Cotes Formeln der numerischen Integration

Wir wollen Interpolationspolynome p benutzen, um Integrale näherungsweise auszurechnen;

$$\int_a^b f(x)dx \approx \int_a^b p(x)dx.$$

Dazu sei

$$a =: x_0 < \cdots < x_n := b \text{ mit } x_i = a + ih, \text{ mit } h := \frac{b-a}{n}$$

eine äquidistante Unterteilung des Intervalls $[a, b]$. Zu f sei $p \in \Pi_n$ das eindeutig bestimmte Interpolationspolynom mit $p(x_i) = f(x_i) =: f_i$. Dann gilt mit (85)

$$p(x) = \sum_{i=0}^{n} f_i L_i(x),$$

wobei L_i das i-te Lagrange Interpolationspolynom bezeichnet. Integration von p ergibt

$$\int_a^b p(x)dx = \sum_{i=0}^{n} f_i \int_a^b L_i(x)dx =: h \sum_{i=0}^{n} \alpha_i f_i,$$

wobei die Koeffizienten α_i gegeben sind durch

$$\alpha_i = \frac{1}{h} \int_a^b L_i(x)dx.$$

Wir erhalten für $f(x) \equiv 1$ das Interpolationspolynom $p(x) \equiv 1$, also

$$nh = b - a = \int_a^b p(x)dx = h \sum_{i=0}^{n} \alpha_i 1,$$

also

$$\sum_{i=0}^{n} \alpha_i = n.$$

Sei jetzt s so gewählt, daß

$$\sigma_i := s\alpha_i, \quad i = 0, \ldots, n$$

ganze Zahlen sind. Dann gilt

$$\int_a^b p(x)dx = h \sum_{i=0}^{n} \alpha_i f_i = \frac{b-a}{ns} \sum_{i=0}^{n} \sigma_i f_i$$

und die Gewichte σ_i sind tabelliert [2]. Auch der Integrationsfehler kann abgeschätzt werden. Es gilt

Satz 5.1. Sei $f \in C^q([a, b])$ und $h := \frac{b-a}{n}$. Dann gilt für den Integrationsfehler

$$\left| \int_a^b p(x)dx - \int_a^b f(x)dx \right| \leq Ch^{q+1} \|f^{(q)}\|_\infty \tag{101}$$

mit einer positiven Konstanten C, die von q und n abhängt.

Fehlerabschätzungen dieser Art folgen aus der **Peano'schen Darstellungsformel für lineare Funktionale auf $C^k([a,b])$.**, die wir ohne Beweis angeben.

Satz 5.2. Sei $R : C^{k-1}([a,b]) \to \mathbb{R}$ ein lineares Funktional der Form

$$R(v) = \sum_{j=0}^{n} \sum_{i=0}^{m_j} c_{ji} v^{(i)}(x_j) + \sum_{i=0}^{k-1} \int_{I_i} v^{(i)}(t)\omega_i(t)dt$$

mit $x_j \in [a,b]$, $0 \le m_j < k$, $\omega_i \in C^k(I_i)$ ($I_i \subseteq [a,b]$). Ist zusätzlich $v \in C^k([a,b])$, so besitzt es die Darstellung

$$R(v) = \sum_{j=0}^{k-1} a_j v^{(j)}(a) + \int_a^b R_x(G_k(x,t))v^{(k)}(t)dt \tag{102}$$

mit

$$a_j = R\left(\frac{(x-a)^j}{j!}\right),\ 0 \le j \le k-1,\ \text{und } G_k(x,t) = \frac{(x-t)_+^{k-1}}{(k-1)!},$$

$$\text{wobei } x_+ := \max\{0,x\}, x_+^0 := \begin{cases} 1, x \ge 0 \\ 0, x < 0, \end{cases}, x_+^l := (x_+)^l,\ l \ge 1.$$

Der Subindex x an R meint, daß R bezgl.des Arguments x auszuwerten ist.

Ein Beweis findet sich etwa in [16, Kapitel 11].
Die Fehlerabschätzung (101) wird mit Satz 5.2 wie folgt erhalten. Sei $f \in C^{n+1}([a,b])$ und

$$R(f) = \sum_{i=0}^{n} \tilde{\alpha}_i f_i - \int_a^b f(x)dx \text{ mit } \tilde{\alpha}_i := h\alpha_i.$$

Dann gilt obige Darstellung von R mit $k = n+1$, $m_j = 0$, $I_i = [a,b]$, $\omega_0 = -1$, $\omega_i = 0$ für $i \ge 1$. Wir können demnach R in der Form (102) darstellen, wobei dort wegen $R(p) = 0$ für alle $p \in \Pi_n$ alle Koeffizienten $a_0, \ldots, a_n$ verschwinden. Es verbleibt

$$R(f) = \int_a^b R_x\left(\frac{(x-t)_+^n}{n!}\right) f^{(n+1)}(t)dt$$

abzuschätzen. Dabei kann $R_x\left(\frac{(x-t)_+^n}{n!}\right)$ in vielen Fällen exakt angegeben werden. Für die Newton-Cotes Formeln gilt sogar, daß $R_x\left(\frac{(x-t)_+^n}{n!}\right)$ auf $[a,b]$ ein konstantes Vorzeichen besitzt [2, Kapitel 3.2], so daß sich mit dem Mittelwertsatz direkt die Darstellung

$$R(f) = \int_a^b R_x\left(\frac{(x-t)_+^n}{n!}\right) f^{(n+1)}(t)dt = f^{(n+1)}(\xi) \int_a^b R_x\left(\frac{(x-t)_+^n}{n!}\right) dt \text{ mit } \xi \in [a,b] \tag{103}$$

ergibt. Wir beobachten, daß

$$\int_a^b R_x\left(\frac{(x-t)_+^n}{n!}\right) dt$$

nicht mehr von f abhängt, demnach mit der Wahl von $f(x) := x^{n+1}$ in (103) berechenbar ist. Es ergibt sich

$$R(x^{n+1}) = (n+1)! \int_a^b R_x\left(\frac{(x-t)_+^n}{n!}\right) dt,$$

also

$$\int_a^b R_x\left(\frac{(x-t)_+^n}{n!}\right)dt = \frac{R(x^{n+1})}{(n+1)!}. \tag{104}$$

Die Konstante C aus Abschätzung (101) kann jetzt als

$$C = \left|\frac{R(x^{n+1})}{(n+1)!}\right|$$

gewählt werden. Einige der Fehlerdarstellungen für die nachfolgend tabellierten Newton-Cotes Formeln werden Sie im Rahmen von Übungsaufgaben beweisen. In der Darstellung werden die σ_i und der Hauptnenner s aller α_i verwendet. Für größere Werte von n treten auch negative Gewichte σ_i auf und die

n	σ_1	σ_2	σ_3	σ_4	σ_5	σ_6	σ_7	ns	Fehler	Name
1	1	1						2	$h^3\frac{1}{12}f^{(2)}(\xi)$	Trapez Regel
2	1	4	1					6	$h^5\frac{1}{90}f^{(4)}(\xi)$	Simpson Regel
3	1	3	3	1				8	$h^5\frac{3}{80}f^{(4)}(\xi)$	$\frac{3}{8}$ Regel
4	7	32	12	32	7			90	$h^7\frac{8}{945}f^{(6)}(\xi)$	Milne Regel
5	19	75	50	50	75	91		288	$h^7\frac{275}{12096}f^{(6)}(\xi)$	ohne Namen
6	41	216	27	272	27	216	41	840	$h^9\frac{9}{1400}f^{(8)}(\xi)$	Weddle Regel

Tabelle 1: Abgeschlossene Newton-Cotes Formeln für $n = 1, \ldots, 6$

Formeln werden numerisch unbrauchbar, siehe etwa Diskussion [2, Kapitel3.1]. Die Newton-Cotes Formeln aus Tabelle 1 heißen abgeschlossen, weil für die gewählte Unterteilung $\Delta : a = x_0 < \cdots < x_n = b$ gilt, Intervallanfang und -ende demnach Stützstellen der Quadratur sind. Als Vertreter der **Offenen Newton-Cotes Formeln** sei hier nur die **Mittelpunkt Regel** erwähnt. Sie basiert auf der Wahl des Intervallmittelpunktes als Stützstelle für die Quadratur:

$$\int_a^b f(x)dx = (b-a)f\left(\frac{a+b}{2}\right) + \frac{(b-a)^3}{24}f''(\xi) \text{ mit einem } \xi \in [a,b]. \tag{105}$$

Die Fehlerdarstellung wird dabei wieder mittels der Darstellung (102) gewonnen.

5.2 Zusammengesetzte Formeln

Ist das Intervall $[a,b]$ zu groß, ist das Interpolationspolynom $p_n \in \Pi_n$ nicht immer ein gutes Modell für die zu integrierende Funktion f. In solchen Fällen ist es wieder sinnvoller, auf den Teilintervallen $[x_{i-1}, x_i]$ einer Unterteilung $\Delta : a := x_0 < x_1 < \cdots < x_n := b$ stückweise zu interpolieren und als Quadraturformel für f die Summe der Newton-Cotes Quadraturen von f auf den einzelnen Teilintervallen zu verwenden. Sei dazu jetzt die Unterteilung Δ wieder äquidistant, d.m. $x_i = a + ih$, $h := \frac{b-a}{n}$.

5.2.1 Summierte Trapez Regel

Auf $[x_{i-1}, x_i]$ wird linear interpoliert, dort also die Trapez Regel zur Quadratur verwendet. D.m.

$$\int_{x_{i-1}}^{x_i} f(x)dx \approx \frac{h}{2}\left(f(x_{i-1}) + f(x_i)\right).$$

Die **Summierte Trapez Regel** ergibt sich dann zu

$$T(h) := \frac{h}{2}\sum_{i=1}^{n}\left\{\left(f(x_{i-1}) + f(x_i)\right)\right\} = h\left[\frac{f(a)}{2} + f(a+h) + f(a+2h) + \cdots + f(a+(n-1)h) + \frac{f(b)}{2}\right]. \tag{106}$$

Der Fehler

$$T(h) - \int_a^b f(x)dx$$

ergibt sich für $f \in C^2$ als Summe der Fehler der Trapezregel über die Teilintervalle $[x_{i-1}, x_i]$ nach Tabelle 1 mit $h = \frac{b-a}{n}$ zu

$$T(h) - \int_a^b f(x)dx = \frac{h^2}{12}(b-a)\sum_{i=1}^{n}\frac{1}{n}f''(\xi_i) \text{ mit } \xi_i \in [x_{i-1}, x_i].$$

Anwendung des Zwischenwertsatzes auf f'' liefert die Existenz eines $\xi \in (\min \xi_i, \max \xi_i) \subseteq (a, b)$ mit

$$f''(\xi) = \frac{1}{n}\sum_{i=1}^{n} f''(\xi_i),$$

also für den Gesamtfehler

$$T(h) - \int_a^b f(x)dx = \frac{h^2}{12}(b-a)f''(\xi). \tag{107}$$

5.2.2 Summierte Simpson Regel

Ist n gerade, so kann die Simpson Regel zur Quadratur der Funktion f auf den Teilintervallen $[x_{2i-2}, x_{2i}]$ für $i = 1, \ldots, n$ angewendet werden. Deren Länge beträgt $2h$, so daß

$$\int_{x_{2i-2}}^{x_{2i}} f(x)dx \approx \frac{h}{3}\left(f(x_{2i-2}) + 4f(x_{2i-1}) + f(x_{2i})\right)$$

richtig ist. Die **Summierte Simpson Regel** ergibt sich damit zu

$$\begin{aligned} S(h) &:= \frac{h}{3}\sum_{i=1}^{n}\left(f(x_{2i-2}) + 4f(x_{2i-1}) + f(x_{2i})\right) = \\ &\frac{h}{3}\left[f(a) + 4f(a+h) + 2f(a+2h) + \cdots + 2f(a+(n-2)h) + 4f(a+(n-1)h) + f(b)\right], \end{aligned} \tag{108}$$

für deren Gesamtfehler gilt

$$S(h) - \int_a^b f(x)dx = \frac{h^4}{180}(b-a)f^{(4)}(\xi) \text{ mit einem } \xi \in (a, b). \tag{109}$$

In analoger Weise können wir mit Hilfe der anderen Newton-Cotes Formeln auf Teilintervallen weitere zusammengesetzte Quadraturformeln erhalten.

5.3 Extrapolation und Romberg Integration

Die Idee der **Extrapolation** ist sehr einfach; Gegeben sei ein **Funktional** $F(h)$ zur näherungsweisen Auswertung der Funktion R. Wir wollen davon ausgehen, daß mit $h \to 0$ auch $F(h) \to R$ erfüllt ist. Ferner seien $0 < h_n < \cdots < h_0$ Stützstellen mit bekannten Stützwerten $F(h_i)$. Ein naheliegender Gedanke besteht nun darin, den nach **0 extrapolierten Wert** $p_n(0)$ des Interpolationspolynoms p_n zu den Daten $(h_i, F(h_i))$ als verbesserte Approximation von R zu betrachten.
Mit Hilfe der **Euler-Maclaurinschen Summenformel** läßt sich zeigen [2, Kapitel 3.4], daß für die Trapezsumme (106) eine Entwicklung in Potenzen von h^2 angegeben werden kann; präziser sei $f \in C^{2m+2}([a, b])$. Dann gilt

$$T(h) = \tau_0 + \tau_1 h^2 + \cdots + \tau_m h^{2m} + \alpha_{m+1}(h)h^{2m+2} \tag{110}$$

mit

$$\tau_0 = \int_a^b f(x)dx \text{ und } \alpha_{m+1}(h) = \frac{B_{2m+2}}{(2m+2)!}(b-a)f^{(2m+2)}(\xi(h)), \xi(h) \in (a,b),$$

wobei $B_i := (-1)^{i-1}\left[\frac{2i-1}{2(2i+1)} + (2i)! \sum_{k=1}^{i-1} \frac{B_k}{(2i-2k+1)!(2k)!}\right]$ die **Bernoulli Zahlen** bezeichnen.
Funktioniert Extrapolation nach 0 (und ob sie funktioniert), wird sich mit dem Interpolationspolynom

$$p_{2m}(h) = b_0 + b_1 h^2 + \cdots + b_m h^{2m}$$

zu den Daten $(h_i, T(h_i))$ mittels $p_{2m}(0) = b_0$ eine verbesserte Approximation des Integrals $\int_a^b f(x)dx$ ergeben. Ihr Fehler hängt natürlich von der Wahl der Folge $\{h_i\}_{i\in\mathbb{N}}$ ab. In der Praxis werden die Folgen

Definition 5.3.

- $h_0 = b-a$, $h_i = \frac{h_{i-1}}{2}$, $i \geq 1$ (**Romberg Folge**), bzw.
- $h_0 = b-a$, $h_1 = \frac{h_0}{2}$, $h_2 = \frac{h_0}{3}, \ldots, h_i = \frac{h_{i-2}}{2}$, $i \geq 3$ (**Bulirsch Folge**)

verwendet. Verbleibt noch bei gegebener Folge $\{h_i\}_{i\in\mathbb{N}}$ mit Stützwerten $T(h_i)$ die Berechnung von $p_{2m}(h)$ und dessen Auswertung an der Stelle 0. Dazu verwenden wir Algorithmus 4.5 zur Berechnung der dividierten Differenzen und schließlich Algorithmus 4.6 zur Auswertung des Interpolationspolynoms an der Stelle 0. Hierbei ist zu beachten, daß bei der Interpolation

$$h_0^2, h_1^2, \ldots h_m^2 \text{ als Stützstellen zu verwenden sind.}$$

In diesem Zusammenhang siehe auch **Neville Schema** zur effizienten Auswertung von Interpolationspolynomen etwa in [2].
Abschließend ein

Beispiel 5.4. *1. Sei $h_0 = b-a$, $h_1 = \frac{b-a}{2}$. Wir interpolieren zu den Daten*

$$(h_0^2, T(h_0)) \text{ und } (h_1^2, T(h_1))$$

und erhalten

$$p_2(x) = T(h_0) + \frac{T(h_1) - T(h_0)}{h_1^2 - h_0^2}(x - h_0^2),$$

also nach kurzer Rechnung

$$p_2(0) = \frac{4}{3}T(h_1) - \frac{1}{3}T(h_0).$$

Wegen

$$T(h_0) = \frac{b-a}{2}(f(a) + f(b)) \text{ und } T(h_1) = \frac{b-a}{4}\left(f(a) + 2f(\frac{a+b}{2}) + f(b)\right)$$

ergibt sich

$$p_2(0) = \frac{b-a}{6}\left(f(a) + 4f(\frac{a+b}{2}) + f(b)\right),$$

also gerade die Simpson Regel, deren Fehler von der Ordnung h^4 ist. Der Fehler der Trapez Regel hingegen war nur von der Ordnung h^2.
2. Analog wird mit $h_2 = \frac{b-a}{4}$ über $p_4(0)$ die Milne Regel erhalten. Nachweis als Übungsaufgabe.

Bemerkung 5.5. Für gewisse Folgen können bei der Berechnung des Stützwertes $T(h_i)$ Funktionswerte verwendet werden, die schon bei der Berechnung von $T(h_{i-1})$ Verwendet wurden. So gilt etwa für die Romberg Folge

$$T(h_{i+1}) = T\left(\frac{h_i}{2}\right) = \frac{1}{2}T(h_i) + h_{i+1}\left[f(a+h_{i+1}) + f(a+3h_{i+1}) + \ldots, +f(b-h_{i+1})\right].$$

Abschließend noch eine Fehlerabschätzung für die **Romberg Integration** (=Extrapolation der summierten Trapez Regel mit der Romberg Folge) nach [2, Kapitel 3.4].

Satz 5.6. Für den Fehler in der Romberg Integration gilt im Fall $f \in C^{2m+2}([a,b])$ die Fehler Darstellung

$$p_{2m}(0) - \int_a^b f(x)dx = (b-a)h_0^2 h_1^2 \dots h_m^2 \frac{(-1)^m B_{2m+2}}{(2m+2)!} f^{(2m+2)}(\xi) \text{ mit einem } \xi \in (a,b).$$

5.4 Gauß Quadratur

Bisher haben wir bei der Quadratur noch nicht weiter über die Wahl der Stützstellen $x_i \in [a,b]$ nachgedacht. Bei der Polynominterpolation in Kapitel 3 hat sich das ausgezahlt. Es wird sich herausstellen, daß mit Hilfe der Nullstellen x_i von gewissen **Orthogonalpolynomen** Gewichte w_i $(i = 1, \dots, n)$ bestimmt werden können, so daß die Quadraturformeln der Form

$$\sum_{i=1}^n w_i f(x_i) \approx \int_a^b f(x)dx$$

bereitgestellt werden können, die Polynome bis zum Grad $2n-1$ exakt integrieren. Zur Einführung von Orthogonalpolynomen definieren wir zunächst für quadrat-integrierbare Funktionen f und g

$$(f,g) := \int_a^b f(x)g(x)dx.$$

Diese Form definiert ein Skalarprodukt auf dem Raum

$$L^2(a,b) := \{v : (a,b) \to \mathbb{R}; v \text{ meßbar und } \int_a^b |v(x)|^2 dx < \infty\}$$

der über (a,b) **quadrat-integrierbaren Funktionen**. Den folgenden Satz entnehmen wir [2, (3.6.3) Satz]. Sein Beweis gelingt vollständiger Induktion und mit Hilfe des **Gram-Schmidt Orthogonalisierungsverfahrens** (siehe Vorlesung).

Satz 5.7. Es gibt zu $j = 0, 1, 2, \dots$ eindeutig bestimmte Polynome $p_j \in \Pi_j$ mit höchstem Koeffizienten 1 mit

$$(p_i, p_j) = 0 \text{ für } i \neq j. \tag{111}$$

Diese Polynome erfüllen die Rekursionsformel

$$p_0(x) = 1,\ p_1(x) = \left(x - \frac{(xp_0, p_0)}{(p_0,p_0)}\right) p_0(x) \text{ und} \tag{112}$$
$$p_{i+1}(x) = \left(x - \frac{(xp_i, p_i)}{(p_i,p_i)}\right) p_i(x) - \frac{(p_i,p_i)}{(p_{i-1},p_{i-1})} p_{i-1}(x) \text{ für } i \geq 1.$$

Wir kommen jetzt zum Hauptresultat.

Satz 5.8.

i. Mit den Nullstellen $x_1, \dots x_n$ von p_n bezeichnen $w_1 \dots, w_n$ die Lösung des linearen Gleichungssystems

$$\sum_{i=1}^n p_k(x_i) w_i = \begin{cases} (p_0,p_0), & \text{falls } k = 0, \\ 0, & \text{falls } k \geq 1. \end{cases} \tag{113}$$

Dann gilt $w_i > 0$ für $i = 1, \ldots, n$ und

$$\sum_{i=1}^{n} w_i p_i(x_i) = \int_a^b p(x)dx \text{ für alle } p \in \Pi_{2n-1}, \tag{114}$$

d.m. Polynome vom Grad $\leq 2n-1$ werden von

$$Q(f) := \sum_{i=1}^{n} w_i f(x_i) \tag{115}$$

exakt integriert.

ii. Gilt für reelle Zahlen x_i, w_i $(i = 1, \ldots, n)$ (114) für alle $p \in \Pi_{2n-1}$, so sind die x_i Nullstellen von p_n und die reellen Zahlen w_i erfüllen das Gleichungssystem (113).

iii. Es gibt keine reellen Zahlen x_i, w_i $(i = 1, \ldots, n)$, so daß (114) für alle $p \in \Pi_{2n}$ richtig ist.

Der Beweis dieses Satzes ist etwa in [2, (3.6.12) Satz] zu finden und benutzt die nachfolgend genannten Eigenschaften der Orthogonalpolynome p_i.

Hilfsatz 5.9. Es gilt

i. $(p, p_n) = 0$ für alle $p \in \Pi_{n-1}$,

ii. die Nullstellen $x_1, \ldots, x_n$ von p_n sind reell, einfach und liegen im offenen Intervall (a, b),

iii. für beliebige $t_1 < t_2 < \cdots < t_n$ ist die $n \times n$ Matrix

$$\begin{bmatrix} p_0(t_1) & \ldots & p_0(t_n) \\ \vdots & & \vdots \\ p_{n-1}(t_1) & \ldots & p_{n-1}(t_n) \end{bmatrix} \tag{116}$$

nicht singulär. Das meint, daß $p_0, p_1, p_2, \ldots$ ein **Tschebyscheff System** bilden. Die Bedingung (116) heißt **Haar'sche Bedingung**.

Beweis.

i. $p \in \Pi_{n-1}$, dann p Linearkombination der $p_0, \ldots, p_{n-1}$, also $p \perp p_n$.

ii. Seien $a < x_1 \cdots < x_l < b$ die Nullstellen von p_n, an denen ein Vorzeichenwechsel stattfindet. Gilt $l < n$, so ist mit

$$q(x) := \prod_{i=1}^{l} (x - x_i) \in \Pi_l$$

wegen i. $(q, p_n) = 0$. Es ändert aber qp_n auf (a, b) sein Vorzeichen nicht, also kann wegen der Definition des Skalarproduktes $(\cdot, \cdot)$ diese Orthogonalität nicht erfüllt sein. Widerspruch.

iii. Sei $c \neq 0$ mit $A^t c = 0$. Dann hat $q(x) := \sum_{i=0}^{n-1} c_i p_i(x)$ die n parweise verschiedenen Nullstellen t_i, verschwindet demnach identisch. Ebenso die Ableitungen bis zur Ordnung $n-1$. Der höchste Koeffizient der p_i ist jeweils 1, ihr Grad demnach i, also

$$0 = q^{(n-1)} = (n-1)!c_{n-1} \Rightarrow c_{n-1} = 0,\ q^{(n-2)} = 0 = (n-2)!c_{n-2} \Rightarrow c_{n-2} = 0,\ \ldots, \cdots \Rightarrow c_0 = 0.$$

Das ist ein Widerspruch zu $c \neq 0$.

Damit ist alles gezeigt. □

Bemerkung 5.10.
1. Für $[a,b]=[-1,1]$ gehen die Resultate aus Satz 5.8 auf Gauß zurück, die zugehörigen Orthogonalpolynome sind (bis auf Normierung) die **Legendre Polynome**

$$p_k(x) := \frac{k!}{(2k)!}\frac{d^k}{dx^k}\left(x^2-1\right)^k,\ k=0,1,\dots$$

2. Obige Ausführungen können auch auf Skalarprodukte der Form

$$(f,g) := \int_a^b f(x)g(x)w(x)dx.$$

mit positiven **Gewichtsfunktionen** *w verallgemeinert werden. Für spezielle Gewichtsfunktionen w und Intervalle sind die Orthogonalpolynome bekannt.*

- *$[a,b]=[-1,1]$, $w(x)=\frac{1}{\sqrt{1-x^2}}$. Orthogonalpolynome sind die Tschebyscheff Polynome T_n aus Definition 4.10.*
- *$[a,b]=[0,\infty]$, $w(x)=e^{-x}$. Orthogonalpolynome sind die* **Laguerre Polynome** *L_n.*
- *$[a,b]=[-\infty,\infty]$, $w(x)=e^{-x^2}$. Orthogonalpolynome sind die* **Hermite Polynome** *H_n.*

3. Die Gewichte w_i und Nullstellen x_i sind in den gängigen Tafelwerken tabelliert.

Verbleibt noch die praktische Berechnung der Nullstellen x_i und der Gewichte w_i in Formel (114). Dazu definieren wir zu gegebenen Skalarprodukt $(\cdot,\cdot)$ und Orthogonalpolynomen p_k

$$d_{i+1} := \frac{(xp_i,p_i)}{(p_i,p_i)} \text{ für } i\geq 0, \quad g_{i+1} := \begin{cases} 0 & \text{für } i=0 \\ \sqrt{\frac{(p_i,p_i)}{(p_{i-1},p_{i-1})}} & \text{für } i\geq 1. \end{cases}$$

Dann gilt [2, (3.6.20),(3.6.21) Satz]

Satz 5.11. Die Nullstellen x_i des n-ten Orthogonalpolynoms p_n sind gegeben durch die Eigenwerte der Tridiagonalmatrix

$$J_n := \begin{bmatrix} d_1 & g_2 & & & \\ g_2 & d_2 & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & g_n \\ & & & g_n & d_n \end{bmatrix},$$

die Gewichte w_i in (115) sind gegeben durch

$$w_k = \left(v_1^k\right)^2,\ k=1,\dots,n,$$

wobei $v^k=[v_1^k,\dots,v_n^k]^t$ den k-ten Eigenvektor zum Eigenwert x_k von J_n mit der Normierung $|v^k|_2^2=(p_0,p_0)$ bezeichnet.

Damit ist die Quadratur nach Gauß auf die Berechnung von Eigenwerten und Eigenvektoren von Tridiagonalmatrizen zurückgeführt.
Abschließend noch der Quadraturfehler bei der Gauß Integration.

Satz 5.12. Mit den oben eingeführten Bezeichnungen gilt für $f\in C^{2n}([a,b])$

$$\sum_{i=1}^n w_i f(x_i) - \int_a^b f(x)dx = \frac{f^{(2n)}(\xi)}{(2n)!}(p_n,p_n) \tag{117}$$

mit einem $\xi\in(a,b)$.

Beweis. Der Beweis speziell dieser Aussage findet sich in [2, (3.6.24) Satz]. □

Wir bemerken, daß die Quadraturformel $Q(f)$ aus (115) Polynome bis zum Grad $2n-1$ exakt integriert. Daher ergibt sich mit den gleichen Überlegungen wie nach dem Satz 5.2 für die Newton-Cotes Formeln für den Fehler in der Gauß Quadratur

$$R(f) = \int_a^b R_x \left(\frac{(x-t)_+^{2n-1}}{(2n-1)!} \right) f^{(2n)}(t) dt,$$

falls $f \in C^{2n}([a,b])$ gilt.

6 Ein- und Mehrschrittverfahren für Anfangswertaufgaben

6.1 Motivation

In vielen praktischen Anwendungen führt mathematische Modellierung auf Systeme von gewöhnlichen oder partiellen Differentialgleichungen. So hat uns etwa die makroskopische Modellierung von Transportprozessen unter Berücksichtigung des Gesetzes von Arrhenius auf die partielle Differentialgleichung (14)

$$\begin{cases} \rho_t(t,x) & = & K\Delta_x \rho(t,x) + \delta e^{\rho(t,x)} & \text{in } (\Omega)^T \\ RW(\rho)(t,x) & = & r(t,x) & \text{auf } \partial\Omega^T \\ \rho(0,x) & = & \rho_0(x) & \text{in } \Omega. \end{cases}$$

geführt. Hier sei bemerkt, daß diese Differentialgleichung nicht für alle Parameter $\delta > 0$ eine Lösung für alle Zeiten $t > 0$ zu besitzen braucht. Siehe hierzu das **Bratu Problem**. Diskretisieren wir die partielle Differentialgleichung zunächst in den örtlichen Variablen x, erhalten wir ein System nichtlinearer gewöhnlicher Differentialgleichungen. Dazu sei im Folgenden $\Omega := (0,1)$. Ferner bezeichne

$$x_i = i\Delta x, \ (i = 0, \ldots, n+1), \ \Delta x = \frac{1}{n+1}$$

ein äquidistantes Gitter über Ω mit Ortsauflösung Δx und $\rho_i(t)$ eine Approximation der Dichte $\rho(t,x)$ am Ort $x = x_i$, d.m.

$$\rho_i(t) \approx \rho(t, x_i).$$

Damit ergibt sich aus (14) (Ortsableitungen werden durch entsprechende Differenzenquotienten ersetzt und die dabei aufretenden Funktionswerte durch Approximationen $\rho_i(t)$) etwa

$$\frac{d}{dt}\rho_i(t) \ = \ K\frac{1}{\Delta x^2}\left(\rho_{i-1}(t) - 2\rho_i(t) + \rho_{i+1}(t)\right) + \delta e^{\rho_i(t)}, \quad \rho_i^0 := \rho_0(x_i) \ (i = 1, \ldots, n).$$

Mit der uns wohl bekannten Matrix

$$A = \frac{1}{(\Delta x)^2}\begin{bmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{bmatrix}, \quad \Delta x = \frac{1}{n+1},$$

erhalten wir schließlich die Anfangswertaufgabe nichtlinearer gewöhnlicher Differentialgleichungen

$$\frac{d}{dt}\rho = KA\rho + g(\rho), \ \rho(0) = \begin{bmatrix} \rho_0(x_1) \\ \vdots \\ \rho_0(x_n) \end{bmatrix},$$

wobei $g(\rho) := \delta[e^{\rho_1(t)}, \ldots, e^{\rho_n(t)}]^t$. Setzen wir noch $f(t,\rho) := KA\rho(t) + g(\rho(t))$, so erhalten wir aus (14) nach Diskretisierung im Ort ein System von gewöhnlichen Differentialgleichungen der Form

$$\rho'(t) = f(t, \rho(t)), \ \rho(0) = \rho_0.$$

Diese Form von Systemen soll im Folgenden untersucht werden.

6.2 Einschrittverfahren für Anfangswertaufgaben

Dazu sei $f : [0,T] \times \mathbb{R}^n \to \mathbb{R}^n$ hinreichend glatt (mindestens stetig) . Betrachtet wird das Anfangswertproblem

$$y'(t) = f(t, y(t)),\ t \in (0,T], \quad y(0) = y_0. \tag{118}$$

Zunächst zur Existenz und Eindeutigkeit von Lösungen der AWA (118). Die nachfolgenden Aussagen können etwa [20] entnommen werden.

Satz 6.1. Die Funktion f sei in $[0,T] \times \mathbb{R}^n$ stetig. Dann besitzt die AWA (118) eine Lösung y (**Existenzsatz von Peano**). Ist f zusätzlich Lipschitz stetig bzgl. des 2ten Argumentes, d.m.

$$\|f(t,y) - f(t,z)\| \le L\|y - z\| \text{ für alle } y, z \in \mathbb{R}^n \text{ und alle } t \in [0,T]$$

mit der Lipschitz Konstanten $L > 0$, so ist die Lösung y eindeutig bestimmt [**Picard/Lindelöf, Banach**].

Zur numerischen Diskretisierung definiere durch

$$t_0 := 0, \quad t_{j+1} := t_j + h_j,\ j = 0, \dots, m-1$$

ein Zeitgitter I_h , welches hier der Einfachheit halber zunächst als äquidistant, d.h., $h_j = h = \frac{T}{m}$ für alle j, vorausgesetzt wird.
Ziel ist es nun, eine Gitterfunktion

$$y_h : I_h \to \mathbb{R}^n$$

zu finden, welche die Lösung y von (A) in den Gitterpunkten t_j möglichst gut approximiert, d.h.,

$$y_h(t_j) - y(t_j) \stackrel{!}{=} \text{“klein“},$$

und für zunehmende Feinheit des Gitters in den Gitterpunkten gegen die Lösung konvergiert, d.h.,

$$y_h(t_j) - y(t_j) \stackrel{!}{\to} 0 \quad (h \to 0).$$

Zur Abkürzung schreibe

$$y_j \quad := \quad y_h(t_j).$$

Die in (118) auftretende Zeitableitung kann auf verschiedene Weisen diskretisiert werden. Bezeichnet t_j einen Zeitgitterpunkt, so gilt für die sogenannte **Vorwärtsdifferenz** (falls y einmal stetig differenzierbar ist)

$$y'(t_j) = \frac{y(t_{j+1}) - y(t_j)}{h} + o(1)(\text{ für } h \to 0),$$

für die **Rückwärtsdifferenz** gilt analog

$$y'(t_j) = \frac{y(t_j) - y(t_{j-1})}{h} + o(1)(\text{ für } h \to 0).$$

Es liegt daher nahe anzunehmen, daß

$$\frac{y_{j+1} - y_j}{h} \approx y'(t_j), \text{ bzw. } \frac{y_j - y_{j-1}}{h} \approx y'(t_j)$$

richtig ist und die in (118) auftretenden Ableitungen in einem numerischen Schema durch eine dieser Differenzen zu ersetzen.
Nach diesen Vorüberlegungen und Notationen definieren wir Einschrittverfahren wie folgt.

Definition 6.2. (Einschrittverfahren)
Ein Einschrittverfahren zur Bestimmung einer Näherungslösung y_h auf einem Gitter I_h hat die Form

$$\left\{ \begin{array}{lcll} y_0 : & = & y(0) & \\ t_{j+1} : & = & t_j + h_j & j = 0, 1, \dots, m-1 \\ y_{j+1} : & = & y_j + h_j \varphi(t_j, y_j, h_j). & \end{array} \right. \tag{119}$$

Dabei heißt

$$\varphi(\cdot,\cdot,h) :\ [0,T] \times \mathbb{R}^n \ \to \mathbb{R}^n$$

Verfahrensfunktion des zur Differentialgleichung $y' = f(t,y)$ bei der Schrittweite h gehörenden numerischen Verfahrens.

Beispiel 6.3.

1. *Euler'sches Verfahren* $\varphi(t,y,h) = f(t,y)$
2. *Verbessertes Euler'sches Verfahren* $\varphi(t,y,h) = f(t+\frac{h}{2}, y+\frac{h}{2}f(t,y))$

Verfahrensfunktionen können auch implizit sein;

3. *Implizites Eulerverfahren*

$$\left.\begin{array}{l} y_0 = y(0), \\ t_{j+1} = t_j + h_j \\ y_{j+1} = y_j + h_j f(t_{j+1}, y_{j+1}), \end{array}\right\} j = 0,1,\ldots,m-1, \text{ und}$$

4. *Trapezverfahren*

$$\left.\begin{array}{l} y_0 = y(0) \\ t_{j+1} = t_j + h_j \\ y_{j+1} = y_j + \frac{h_j}{2}[f(t_j,y_j) + f(t_{j+1},y_{j+1})] \end{array}\right\} j = 0,1,\ldots,m-1.$$

Hier ist wie oben

$$T = t_m.$$

Ist z.B.

$$f(t,y) := \lambda y, \quad \lambda \in \mathbb{R},$$

so gilt in 3.

$$\varphi(t,y,h) = \frac{\lambda}{1-h\lambda}\, y,$$

in 4.

$$\varphi(t,y,h) = \frac{\lambda}{1-\frac{h\lambda}{2}}\, y.$$

Wichtige Begriffe sind Konsistenz und Konvergenz. Zu deren Erläuterung ist die Einführung des Verfahrensfehlers hilfreich.

Definition 6.4. (Verfahrensfehler)
Sei y die Lösung von (118) (welche als eindeutig existent vorausgesetzt wird), und sei φ die Verfahrensfunktion eines Einschrittverfahrens zur Integration von

$$y' = f(t,y), \quad y(0) = y_0.$$

Dann heißt

$$r(t,y(t),h) := \frac{y(t+h)-y(t)}{h} - \varphi(t,y(t),h)$$

lokaler Verfahrensfehler des Einschrittverfahrens an der Stelle $(t,y(t))$. Der lokale Verfahrensfehler gibt den Fehler an der entsteht, falls die exakte Lösung der AWA in das Verahren eingesetzt wird.

Definition 6.5. (Konsistenz)
Sei y die Lösung der Anfangswertaufgabe (118) und sei φ die Verfahrensfunktion eines Einschrittverfahrens zur Integration von (118). φ heißt konsistent, genau dann, wenn

$$\sup_{t\in[0,T]} |r(t,y(t),h)| \quad \to 0 \qquad (h \to 0), \tag{120}$$

und ist konsistent von der Ordnung $p > 0$ genau dann, wenn

$$\sup_{t\in[0,T]} |r(t,y(t),h)| \quad \leq \quad Kh^p \qquad (h \to 0). \tag{121}$$

Konsistenz meint demnach, daß das Verfahren angewendet auf die exakte Lösung mit feiner werdender Diskretisierung gegen die Differentialgleichung konvergiert.

Definition 6.6. (Konvergenz)
Ein numerisches Verfahren, daß zu jedem Gitter I_h eine Gitterfunktion y_h berechnet, heißt konvergent für die AWA (118), falls für

$$\varepsilon_h(t) := \; y(t) - y_h(t), \quad t \in [0,T]$$

die Beziehung

$$||\varepsilon_h||_h := \max_{t\in I_h} |y_h(t) - y(t)| \to 0 \quad (h \to 0)$$

gilt. Die Konvergenz ist von der Ordnung $p > 0$ genau dann, wenn

$$\|\varepsilon_h\|_h = \mathcal{O}(h^p) \quad (h \to 0).$$

Konsistenz eines Einschritt Verfahrens (119) reicht nicht aus, um Konvergenz der berechneten Gitterfunktion zu gewährleisten. **Stabilität** wird dazu benötigt. Zur Einführung dieses Begriffes (und der hier benötigten Form) schreiben wir das Verfahren (119) als Nullstellen Problem. Wir setzen für $v = [v_0, \ldots, v_m]^t \in \mathbb{R}^{n\cdot m}$

$$F^h_{j+1}(v) := \frac{v_{j+1} - v_j}{h_j} - \varphi(t_j, v_j, h_j) \text{ für } j = 0, \ldots, m-1 \text{ und } F^h_0(v) := v_0 - y_0. \tag{122}$$

Definition 6.7. (Stabilität)
Das Verfahren (119) heißt **stabil**, falls eine Konstante $S > 0$ existiert mit

$$\|u - v\| \le S\|F^h(u) - F^h(v)\| \text{ für alle } u, v \in \mathbb{R}^{n\cdot m}. \tag{123}$$

Bezeichnet y die Lösung von (118), so bezeichnen wir

$$F^h_{j+1}(y) = \frac{y(t_{j+1}) - y(t_j)}{h_j} - \varphi(t_j, y(t_j), h_j) \text{ für } j = 0, \ldots, m-1 \text{ und } F^h_0(y) := y(0) - y_0.$$

Es gilt der fundamentale

Satz 6.8. Ist das Verfahren (119) konsistent (von der Ordnung p) und stabil, so ist es konvergent (von der Ordnung p).

Beweis. Bezeichne y die Lösung von (118). Konsistenz (der Ordnung p) impliziert $F^h(y) \to 0$ für $h \to 0$ ($\|F^h(y)\| \le Kh^p$). Bezeichne ferner y_h das Resultat des Verfahrens (119). Dann gilt $F^h(y_h) = 0$ nach Definition von F^h und mit (123) dann auch

$$\|y_h - y\|_h \le S\|F^h(y_h) - F^h(y)\| = S\|F^h(y)\| \to 0 \; (h \to 0)(\le SKh^p).$$

Damit ist alles bewiesen. □

Eine hinreichende Bedingung für die Stabilität des Verfahrens (118) ist die Lipschitz Stetigkeit der Verfahrensfunktion φ bezüglich des 2ten Arguments. Der Beweis ist Übungsaufgabe.

Satz 6.9. Die Verfahrensfunktion φ in (118) genüge der Lipschitz Bedingung

$$\|\varphi(t, v, h) - \varphi(t, w, h)\| \le L\|v - w\| \text{ für alle } t \in [0,T] \text{ und für alle } v, w \in \mathbb{R}^n.$$

Dann ist das zugehörige Verfahren (119) stabil.

Beispiel 6.10.
1. Euler explizit ist konvergent mit Ordnung 1 (falls $f \in C^2$).

$$\begin{aligned} r(t,y,h) &= \frac{y(t+h) - y(t)}{\Delta t} - \underbrace{f(t, y(t))}_{y'(t)} \\ &= \frac{h}{2}y''(t) + \mathcal{O}(h^2) \\ &= \mathcal{O}(h) \quad (h \to 0). \end{aligned}$$

2. Verbesserter Euler ist konvergent mit Ordnung 2 (falls $f \in C^3$).

$$r(t,y,h) = \mathcal{O}(h^2) \quad (h \to 0).$$

6.3 Runge-Kutta-Verfahren

Runge-Kutta-Verfahren sind wie folgt definiert.

Definition 6.11. (Runge-Kutta-Verfahren)
Sei $s \in \mathbb{N}$. Ein Runge-Kutta-Verfahren hat die Gestalt

$$\left.\begin{array}{ll} y_0 & = y(0) \\ t_{j+1} & = t_j + h \\ y_{j+1} & = y_j + h\varphi(t_j, y_j, h) \end{array}\right\} \; j = 0, 1, \ldots, m-1,$$

wobei $h = h_j$ und für $\kappa \in \{i-1, i, s\}$

$$\varphi(t, y, h) := \sum_{i=1}^{s} b_i v_i(t, y)$$

mit

$$v_i(t, y) := f\left(t + c_i h, y + h \sum_{k=1}^{\kappa} a_{ik} v_k(t, y)\right), \quad i = 1, \ldots, s.$$

Als Schema

$$\begin{array}{c|ccc} c_1 & a_{11} & \ldots & a_{1s} \\ \vdots & & & \\ c_s & a_{s1} & \ldots & a_{ss} \\ \hline & b_1 & \ldots & b_s. \end{array}$$

Das Verfahren heißt

implizit für $\kappa = s$,

semi-implizit für $\kappa = i$ und

explizit für $\kappa = i - 1$,

s bezeichnet die **Stufe** des Verfahrens.

6.3.1 Explizite Runge-Kutta Verfahren

Hier soll kurz illustriert werden, wie Runge-Kutta Verfahren konstruiert werden. Zunächst jedoch einige Beispiele für explizite Runge-Kutta Verfahren.

Beispiel 6.12.

i. Es sei $s = 1$. Dann ist das zugehörige Schema gegeben durch

$$\begin{array}{c|c} 0 & \\ \hline & 1 \end{array}$$

und wir erhalten

$$\varphi(t, y, h) = v_1(t, y), \text{ mit } v_1(t, y) = f(t, y),$$

also das explizite Euler Verfahren aus 1. in Beispiel 6.3.

ii. Es sei $s = 2$. Verschiedene Schemata sind möglich. Wir wollen 2 Verfahren angeben, welche die Konvergenzordnung 2 besitzen. Zunächst das **Verfahren von Runge** *als Schema*

$$\begin{array}{c|cc} 0 & & \\ \frac{1}{2} & \frac{1}{2} & \\ \hline & 0 & 1 \end{array}$$

also

$$\varphi(t, y, h) = f\left(t + \frac{1}{2}h, y + \frac{1}{2}hf(t, y)\right).$$

Schließlich das **Verfahren von Heun**

$$\begin{array}{c|cc} 0 & & \\ 1 & 1 & \\ \hline & \frac{1}{2} & \frac{1}{2} \end{array}$$

also für die Verfahrensfunktion

$$\varphi(t,y,h) = \frac{1}{2}f(t,y) + \frac{1}{2}f\left(t+h, y+hf(t,y)\right).$$

iii. Es sei $s = 4$. Die maximale erreichbare Konvergenzordnung ist dann ebenfalls 4. Die bekanntesten Vertreter von Verfahren sind das **klassische Runge-Kutta Verfahren** *mit Schema*

$$\begin{array}{c|cccc} 0 & & & & \\ \frac{1}{2} & \frac{1}{2} & & & \\ \frac{1}{2} & 0 & \frac{1}{2} & & \\ 1 & 0 & 0 & 1 & \\ \hline & \frac{1}{6} & \frac{1}{3} & \frac{1}{3} & \frac{1}{6} \end{array}$$

und die $\frac{3}{8}$**-Regel**

$$\begin{array}{c|cccc} 0 & & & & \\ \frac{1}{3} & \frac{1}{3} & & & \\ \frac{2}{3} & -\frac{1}{3} & 1 & & \\ 1 & 1 & -1 & 1 & \\ \hline & \frac{1}{8} & \frac{3}{8} & \frac{3}{8} & \frac{1}{8} \end{array}$$

In diesen Beispielen sind die das Runge-Kutta Verfahren definierenden Koeffizienten gerade so gewählt, daß die Konsistenzordnung möglichst hoch ist. Die Tabelle 2 gibt an, welche welche Stufe $s_{\min}$ des Verfahrens mindestens notwendig ist, um eine vorgelegte Ordnung p zu erreichen, siehe etwa [14]. Am Beispiel

p	1	2	3	4	5	6	7	8	$p > 8$
$s_{\min}$	1	2	3	4	6	7	9	11	$> p+2$

Tabelle 2: Ordnung und Stufen bei expliziten Runge-Kutta Verfahren

einer **autonomen Differentialgleichung** (d.m. $y' = f(y)$) soll illustriert werden, wie die Koeffizienten zu wählen sind, damit Ordnung $p = 2$ bei einem 2-stufigen Verfahren Orndung erhalten wird. In der Vorlesung also aufpassen....

Einschrittverfahren benutzen nur Informationen vom vorherigen und aktuellen Zeitschritt. Verfahren, welche auch Informationen von mehreren vergangenen Zeitschritten nutzen, heißen Mehrschrittverfahren.

6.4 Lineare Mehrschrittverfahren

Hier werden nur lineare Mehrschrittverfahren definiert.

Definition 6.13. (Lineare Mehrschrittverfahren)
Ein Verfahren der Form

$$y_{j+k} = y_{j+q} + h\sum_{i=0}^{k} b_i f_{j+i}, \quad t_{j+k} \in I_h = \{t_0, \ldots, t_m\}$$

mit $k \geq 1, 0 \leq q < k, f_{j+i} := f(t_{j+i}, y_{j+i})$, daß zu bekannten Werten $y_0, \ldots, y_{k-1}$, die Werte $y_k, \ldots, y_m$ zu berechnen gestattet, heißt lineares k-Schrittverfahren. Ist $b_k = 0$, so heißt es explizit, sonst implizit. Der lokale Abschneidefehler für Mehrschrittverfahren ist definiert durch

$$r(t, y(t), h) := \frac{y(t+kh) - y(t+qh)}{h} - \sum_{i=0}^{k} b_i f(t+ih, y(t+ih)).$$

Konsistenzordnung ist wie in (120) und (121) definiert.

MSV können über Integrationsformeln hergeleitet werden. Integration von (118) ergibt

$$y(t_{j+k}) = y(t_{j+q}) + \int_{t_{j+q}}^{t_{j+k}} f(t, y(t))dt. \tag{124}$$

Schreibe jetzt

$$\int_{t_{j+q}}^{t_{j+k}} f(t, y(t))dt \approx h \sum_{i=0}^{k} b_i f(t_{j+i}, y_{j+i}).$$

Um eine solche Näherung zu erhalten, wird etwa der Integrand durch ein Interpolationspolynom in den Stellen

$$(t_{j+i}, f_{j+i}) \begin{cases} i = 0, 1, \ldots, k-1 & \text{(expliziter Fall)} \\ i = 0, 1, \ldots, k & \text{(impliziter Fall)} \end{cases}$$

ersetzt und exakt ausgewertet.

Literatur

[1] Braess. *Methode der finiten Elemente*. Springer, 1994. 32

[2] Bulirsch/Stoer. *Numerische Mathematik I*. Springer, 1990. 18, 22, 47, 50, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64

[3] Bulirsch/Stoer. *Numerische Mathematik II*. Springer, 1990. 25, 26, 27

[4] G. Fischer. *Lineare Algebra*. Vieweg, 2000. 12. verbesserte Auflage. 24

[5] Großmann/Roos. *Numerik partieller Differentialgleichungen*. Teubner, 1994. 29

[6] Sabine Gutsch. Ein Vergleich CG-ähnlicher Verfahren zur Lösung indefiniter Probleme. Master's thesis, Institut für Informatik und Praktische Mathematik, Universität Kiel, 1994. 38

[7] Wolgang Hackbusch. *Iterative Lösung großer schwachbesetzter Gleichungssysteme*. Teubner, 1993. 32

[8] **Kelley, C.T.** *Iterative Methods for Linear and Nonlinear Equations*. SIAM, 1995. 46, 48

[9] Kroener. *Numerical Schemes for Conservation Laws*. Wiley/Teubner, 1997. 9

[10] H.M. Markowitz. Portfolio selection. *Journal of Finance*, 8:77–91, 1952. 13

[11] V.A. Barker O. Axelsson. *Finite Element Solution of Boundary Value Problems*. Academic Press, 1984. 31, 32

[12] René Pinnau. Mathematische Modellierung. Vorlesungskript, Fachbereich Mathematik, TU Darmstadt, 2000. 3

[13] Plato. *Numerische Mathematik Kompakt*. Vieweg, 2000. 18, 51, 52

[14] Strehmel/Weiner. *Numerik Gewöhnlicher Differentialgleichungen*. Teubner Studienbücher, 1995, 1995. 70

[15] H. Schwetlick und H. Kretzschmar. *Numerische Verfahren für Naturwissenschaftler und Ingenieurer*. Fachbuch Verlag Leipzig, 1991. 25

[16] R. Schaback und H. Werner. *Numerische Mathematik*. Springer Lehrbuch, 1992. 26, 58

[17] P. Deuflhard und R. Hohmann. *Numerische Mathematik 1*. De Gruyter, 2002. 45, 46

[18] A. Klar und R. Wegner. Enskog-like kinetic models for vehicular traffic, 1999. 9

[19] Barret; Chan; Demmel; Donato; Dongarra; Eijkhout; Pozo; Romine & van der Vorst. Templates for the solution of linear systems: Building blocks for iterative methods. preprint. 38

[20] Walter. *Gewöhnliche Differentialgleichungen*. Heidelberger Taschenbücher, Springer, 1976. 66