Institut für Theoretische Physik
der Universität zu Köln

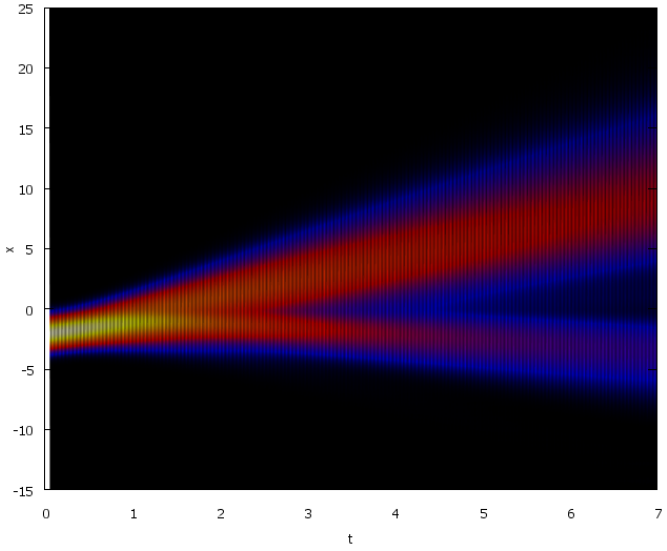
Sommersemester 2010

**Notizen zur Vorlesung**

# Computerphysik: Numerische Methoden zur Lösung physikalischer Probleme

Dozent: Prof. Dr. Johannes Berg
Mitschrift: Gerold Busch

Stand: 29. Juli 2010



Diese Vorlesungsmitschrift von Gerold Busch bezieht sich auf die Vorlesung ‘Computerphysik’, die im Sommersemester 2010 in Köln von Johannes Berg gehalten wurde. Die Mitschrift ist zum internen Gebrauch an der Universität Köln bestimmt. Die Abbildung auf der Titelseite stammt von Martin Sundermann und zeigt die Streuung eines quantenmechanischen Teilchens an einer Potentialschwelle, s. Übung 9. Vielen Dank an Matthias Sitte für Korrekturen. Weitere Fehler bitte direkt an `berg@thp.uni-koeln.de` weitergeben.

# Einleitung

> The purpose of computing is insight, not numbers.
>
> *(Richard W. Hamming)*

Die Computerphysik behandelt numerische Methoden zur Lösung physikalischer Probleme. Die Motivation dafür, numerische Verfahren einzusetzen ist einfach: Die meisten Probleme haben keine exakte analytische Lösung.
Die numerische Analyse beschäftigt sich mit Algorithmen, die numerische Werte (z.B. 3.175 für $\pi$) benutzen. Für jedes Problem muss das numerische Ergebnis neu berechnet werden, daher ist die Effizienz des Algorithmus wichtig. Im Gegensatz dazu kann man in eine analytische Lösung eines Problems beliebige Parameter einsetzen, ohne die Rechnung neu durchführen zu müssen.

Die numerische Analyse hat eine lange Geschichte. Die babylonische Mathematik vor 4000 Jahren konnte $\sqrt{2}$ schon recht genau bestimmen: in ihrem Zahlensystem mit Basis 60 nutzten sie

$$\sqrt{2} \approx 1 + \frac{24}{60} + \frac{51}{60^2} + \frac{20}{60^3} \ , \tag{1}$$

mit einer Abweichung von lediglich 0.003% vom exakten Wert.
Große Schritte in der numerischen Mathematik, von denen wir einige kennenlernen werden, machten zwischen 1700 und 1950 berühmte Mathematiker wie Newton, Gauß, Euler oder Lagrange. Dabei wurden die einzelnen Schritte eines Algorithmus von Hand durchgeführt. Seit etwa 1950 kommen elektronische Computer zum Einsatz. Computer können eine riesige Zahl von numerischen Rechenschritten in kurzer Zeit ausführen und ermöglichen so ein z.B. ein Wechselspiel zwischen Theoriebildung und numerischen Experimenten.

Es gibt allerdings (fast) keine Methoden, die für alle Varianten eines Problems funktionieren. Man braucht also ein gutes Gespür für die Grenzen einer Methode.
Diese Vorlesung gibt ein Einführung in einfache Algorithmen anhand von Problemen aus der Mechanik, Elektrodynamik und der Quantenmechanik.

# Inhaltsverzeichnis

# 1 Nullstellensuche

Ziel ist das Lösen einer Gleichung in einer Variablen. Motivation ist, dass zu linearen, quadratischen, kubischen und quartischen Gleichungen allgemeine Lösungen in geschlossener Form existieren, zu Polynomen höherer Ordnung allerdings nicht, von transzendenten Gleichungen ganz zu schweigen. Es sind also numerische Verfahren gesucht.

## 1.1 Bisektion

Angenommen, wir wissen, dass eine Nullstelle im Intervall $[x_0, x_1]$ liegt. Dann gibt es zwei Möglichkeiten:

1. Die Nullstelle liegt oberhalb von $\frac{x_0+x_1}{2}$
2. Die Nullstelle liegt unterhalb von $\frac{x_0+x_1}{2}$

Die Idee ist also, das Intervall zu halbieren und die Hälfte, die die Nullstelle enthält, als neues Intervall zu wählen. Wiederholte Anwendung dieses Verfahrens grenzt die Nullstelle beliebig stark ein. Ein einfacher Sonderfall, bei dem die Lösung genau in der Mitte zwischen den beiden Intervallgrenzen liegt ist leicht abzufragen, dann kann die Nullstellensuche sofort beendet werden. Die Intervalllänge $\Delta(i)$ nach $i$ Schritten ist

$$\Delta(i) = \Delta(0) \cdot 2^{-i} \tag{1.1}$$

Man spricht von linearer Konvergenz des Algorithmus (bei logarithmischem Maßstab).

## 1.2 Newton-Raphson-Methode

Die Idee der Newton-Raphson-Methode ist die Funktion $f(x)$ mit einer Taylorreihe um einen Startwert $x_0$ bis zu einer gegebenen Ordnung zu entwickeln und die entstandene *polynomiale* Gleichung zu lösen. Diese Lösung nutzt man dann als neuen Startwert.
In erster Ordnung hat man z.B.:

$$f(x) = f(x_0) + (x - x_0)f'(x_0) + \mathcal{O}((x - x_0)^2) \approx f(x_0) + (x - x_0)f'(x_0) \tag{1.2}$$

Wir suchen nun die Nullstelle der Taylor-Entwicklung in erster Ordnung und erhalten den ersten Iterationsschritt: $x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$. Führt man dies weiter, so erhält man die iterative Regel:

$$\boxed{x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)}} \tag{1.3}$$

Wir betrachten nun die Konvergenz der Newton-Raphson-Methode. Sei $\epsilon_i = x_i - x$ der (unbekannte) Abstand der $i$-ten Iteration $x_i$ von der Nullstelle $x$. Die Iterationsregel ergibt dann:

$$\epsilon_{i+1} = \epsilon_i - \frac{f(x_i)}{f'(x_i)} \tag{1.4}$$

Wir führen nun eine Taylor-Entwicklung von $f(x_i)$ und $f'(x_i)$ um $x$ herum durch.

$$\begin{aligned} f(x_i) &= \underbrace{f(x)}_{=0} + (x_i - x)f'(x) + \frac{1}{2}(x_i - x)^2 f''(x) + \dots \\ &= \epsilon_i f'(x) + \frac{1}{2}\epsilon_i^2 f''(x) + \dots \\ f'(x_i) &= f'(x) + (x_i - x)f''(x) + \dots \\ &= f'(x) + \epsilon_i f''(x) + \dots \end{aligned} \tag{1.5}$$

Dies führt auf die Iterationsvorschrift für $\epsilon_i$

$$\begin{aligned} \epsilon_{i+1} &\approx \epsilon_i - \frac{\epsilon_i f'(x) + \frac{1}{2}\epsilon_i^2 f''(x)}{f'(x) + \epsilon_i f''(x)} \\ &= \epsilon_i \left[ 1 - \frac{1 + \frac{1}{2}\epsilon_i \, f''/f'}{1 + \epsilon_i f''/f'} \right] \\ &= \epsilon_i \left[ 1 - \left( 1 + \frac{1}{2}\epsilon_i f''/f' \right) \left( 1 - \epsilon_i f''/f' \right) + \mathcal{O}(\epsilon^3) \right] \\ &\approx \frac{1}{2}\epsilon_i^2 \frac{f''}{f'} \end{aligned} \tag{1.6}$$

Jeder Iterationsschritt bringt also ein $\epsilon$ an Genauigkeit. Ein Vergleich zwischen der Intervallhalbierung und der Newton-Raphson-Methode zeigt
Intervallhalbierung: $\frac{\Delta(i+1)}{\Delta(i)} = \frac{1}{2}$
Newton-Raphson: $\frac{\epsilon_{i+1}}{\epsilon_i} = \frac{1}{2}\epsilon_i \frac{f''(x)}{f'(x)}$

Die Konvergenz unter der Newton-Raphson-Methode ist also um so schneller, je kleiner $\epsilon_i$ ist.
Grundsätzlich ist allerdings Vorsicht geboten. Die Konvergenz hängt von der Wahl des Startpunktes ab, ein ungeschickt gewählter Startpunkt kann dazu führen, dass der Abstand von der gesuchten Lösung divergiert.
Die Newton-Raphson-Methode lässt sich verallgemeinern: Zum Beispiel mit quadratischer Entwicklung um $x$.

# 2 Numerische Integration

Eine Vielzahl physikalischer Probleme führt auf Integrale, die nicht analytisch lösbar sind. Selbst wenn $f(x)$ in geschlossener Form vorliegt, muss das nicht für die Stammfunktion $F(x)$ mit $f(x) = \frac{dF(x)}{dx}$ der Fall sein. Ein Beispiel:

$$\int_y^\infty dx\, e^{-\frac{1}{2}x^2} \tag{2.1}$$

Wir lassen uns im Folgenden von der Definition des Riemannschen Integrals inspirieren:

$$\int_a^b dx\, f(x) = \lim_{\Delta\to 0} \Delta \cdot \sum_{i=0}^{(b-a)/\Delta} f(a + i\Delta) \tag{2.2}$$

Das Integral ergibt sich im Grenzfall $\Delta \to 0$ aus einer Summe unendlich vieler Terme. Aufgabe der numerischen Integration ist es nun mit möglichst wenigen Termen in einer endlichen Summe eine möglichst gute Näherung für das Integral $I = \int dx\, f(x)$ zu erhalten:

$$I_{\text{numerisch}} = \sum_{i=1}^{N} \omega_i f_i \overset{!}{\approx} I_{\text{exakt}} \tag{2.3}$$

## 2.1 Integration über kleine Intervalle

Wir betrachten zunächst kleine Intervalle, über die eine Funktion zu integrieren sei. Sei $[x_0, x_1]$ das Integrationsintervall, und $f_0, f_1$ die Werte der zu integrierenden Funktion bei $x_0$ und $x_1$. Wir nähern das Integral durch:

$$I \approx \omega_0 f_0 + \omega_1 f_1 \tag{2.4}$$

und fordern, dass die Näherung für die beiden Funktionen $f(x) = 1$ und $f(x) = x$ exakt ist.

$$\begin{aligned} \int_{x_0}^{x_1} 1\, dx &= x_1 - x_0 \overset{!}{=} \omega_0 + \omega_1 \\ \int_{x_0}^{x_1} x\, dx &= \frac{1}{2}\left(x_1^2 - x_0^2\right) \overset{!}{=} \omega_0 x_0 + \omega_1 x_1 \end{aligned} \tag{2.5}$$

Die zwei resultierenden Gleichungen in den zwei Unbekannten $\omega_0$ und $\omega_1$ lassen sich elementar lösen. Wir erhalten für $\omega_0$ und $\omega_1$:

$$\omega_0 = \omega_1 = \frac{x_1 - x_0}{2} = \frac{h}{2} \tag{2.6}$$

mit $h = x_1 - x_0$. Wir haben also für kleine Intervalle:

$$\int_{x_0}^{x_1} dx\, f(x) \approx \frac{h}{2}\,(f_0 + f_1) \tag{2.7}$$

Für große Intervalle erhält man durch Zusammensetzen aus kleinen Intervallen die sogenannte Trapezregel:

$$I = \frac{h}{2} f_0 + h \cdot f_1 + h \cdot f_2 + \ldots + h \cdot f_{n-1} + \frac{h}{2} f_n = \frac{h}{2}(f_0 + f_n) + \sum_{k=1}^{k-1} h f_k \tag{2.8}$$

Natürlich können pro Intervall auch mehrere Stützstellen verwendet werden, z.B. $I \approx \omega_0 f_0 + \omega_1 f_1 + \omega_2 f_2$. Wir fordern nun, dass die Näherung für $f(x) = 1$, $f(x) = x$ und $f(x) = x^2$ exakt ist und erhalten das Gleichungssystem:

$$\begin{aligned}
\int dx &= x_2 - x_0 \stackrel{!}{=} \omega_0 + \omega_1 + \omega_2 \\
\int x\, dx &= \frac{1}{2}\left(x_2^2 - x_0^2\right) \stackrel{!}{=} \omega_0 x_0 + \omega_1 x_1 + \omega_2 x_2 \\
\int x^2\, dx &= \frac{1}{3}\left(x_2^3 - x_0^3\right) \stackrel{!}{=} \omega_0 x_0^2 + \omega_1 x_1^2 + \omega_2 x_2^2
\end{aligned} \tag{2.9}$$

Lösen nach $\omega_0$, $\omega_1$ und $\omega_2$ ergibt:

$$\omega_0 = \frac{1}{3}h,\ \omega_1 = \frac{4}{3}h,\ \omega_2 = \frac{1}{3}h \tag{2.10}$$

Einsetzen ergibt die Simpson-Regel:

$$\boxed{\int_{x_0}^{x_2} dx\, f(x) \approx \frac{h}{3}\,(f_0 + 4 \cdot f_1 + f_2)} \tag{2.11}$$

Diese sogenannte Simpson-Regel ist exakt für Polynome 2. Ordnung. Sie entspricht geometrisch einer Näherung von $f(x)$ an 3 Stützstellen durch eine Parabel.
Die Konstruktion von Integrationsregeln auf mehr Stützstellen pro kleinem Intervall und die Näherung der Funktion durch entsprechende Polznome höherer Ordnung ist einfach.

## 2.2 Fehlerabschätzung am Beispiel der Trapezregel

Es ist hilfreich zu wissen, wie groß der Fehler ist, den wir durch die Integrationsregel einer gegebenen Ordnung, machen. Wir betrachten den Fall der Trapezregel (erste Ordnung). Wir entwickeln um den Punkt $x_{i+1/2} \equiv x_i + h/2$ zwischen zwei Stützstellen:

$$f(x_i + \frac{1}{2}h + \xi) = f_{i+1/2} + \xi\, f'_{i+1/2} + \frac{1}{2}\xi^2\, f''_{i+1/2} + \ldots \tag{2.12}$$

Sowohl das exakte Integral als auch das Ergebnis der Trapezregel lassen sich durch die Taylorreihe ausdrücken und dann vergleichen:

$$\begin{aligned} I_{\text{exakt}} &= \int_{-h/2}^{h/2} d\xi\, f(x_i + \frac{1}{2}h + \xi) \\ &= [\xi]_{-h/2}^{h/2} f_{i+1/2} + \left[\frac{1}{2}\xi^2\right]_{-h/2}^{h/2} f'_{i+1/2} + \frac{1}{2}\left[\frac{1}{3}\xi^3\right]_{-h/2}^{h/2} f''_{i+1/2} + \dots \\ &= h\, f_{i+1/2} + \frac{1}{3}\left(\frac{h}{2}\right)^3 f''_{i+1/2} + \mathcal{O}(h^4) \end{aligned} \tag{2.13}$$

Vergleich mit der Trapez-Regel:
Zunächst berechnen wir $f_i$ und $f_{i+1}$ aus der Entwicklung $f(x_i + h/2 + \xi)$ von oben:

$$\begin{aligned} f_i &= f(x_{i+1/2} - \frac{1}{2}h) \\ &= f_{i+1/2} - \frac{h}{2} f'_{i+1/2} + \frac{1}{2}\left(\frac{h}{2}\right)^2 f''_{i+1/2} + \dots \\ f_{i+1} &= f(x_{i+1/2} + \frac{1}{2}h) \\ &= f_{i+1/2} + \frac{h}{2} f'_{i+1/2} + \frac{1}{2}\left(\frac{h}{2}\right)^2 f''_{i+1/2} + \dots \end{aligned} \tag{2.14}$$

Damit haben wir:

$$I_{\text{Trapez}} = \frac{h}{2}\left(f_i + f_{i+1}\right) = h\, f_{i+1/2} + \frac{h^3}{8} f''_{i+1/2} + \mathcal{O}(h^4) \tag{2.15}$$

$I_{\text{exakt}}$ und $I_{\text{Trapez}}$ sind in erster Ordnung gleich. Sie unterscheiden sich in dritter Ordnung um:

$$\Delta I = \underbrace{\left(\frac{1}{8} - \frac{1}{24}\right)}_{1/12} h^3 f''_{i+1/2} + \mathcal{O}(h^4) \tag{2.16}$$

Die Fehlerabschätzung für das Integral bestehend aus $N = (b-a)/h$ kleinen Integralen ist dann:

$$\sum_i \Delta I_i = \frac{h^3}{12}\sum_i f''_{i+1/2} = \frac{h^3}{12} N \left\langle f''_{i+1/2}\right\rangle = \frac{h^2}{12}(b-a)\left\langle f''_{i+1/2}\right\rangle \le \frac{h^2}{12}(b-a)\max\{|\mathrm{f''(x)}|; \mathrm{a} \le \mathrm{x} \le \mathrm{b}\} \propto \mathrm{h}^2 \tag{2.17}$$

## 2.3 Romberg-Integration

Wir haben:

$$\begin{aligned} I = \int_a^b dx\, f(x) &= \sum_i \left(I^{(i)}_{\text{Trapez}} + \frac{h^2}{12} f''_{i+1/2} + \mathcal{O}(h^4)\right) \\ &= I^{(h)}_{\text{Trapez}} + \frac{h^2}{12}(b-a)\left\langle f''_{i+1/2}\right\rangle + \mathcal{O}(h^4) \end{aligned} \tag{2.18}$$

Die genaue Kontrolle der Fehler bei der Trapezregel (oder einem anderen Verfahren) erlaubt die Ergebnisse bei unterschiedlichen $h$ zu kombinieren, um Fehler unterschiedlicher Ordnung zu eliminieren. Diese Vorgehensweise ist unter dem Namen Romberg-Integration bekannt.
Rechnet man das Integral mit Schrittweite $h' = h/2$, so erhält man:

$$\begin{aligned} I = \ldots &= I^{h/2}_{\text{Trapez}} + \frac{(h/2)^2}{12}(b-a)\left\langle f''_{i+1/2}\right\rangle + \mathcal{O}(h^4) \\ &= I^{h/2}_{\text{Trapez}} + \frac{1}{4}\frac{h^2}{12}(b-a)\left\langle f''_{i+1/2}\right\rangle + \mathcal{O}(h^4) \end{aligned} \tag{2.19}$$

Wir kombinieren nun $I^h$ und $I^{h/2}$ so, dass sich die Fehler führender Ordnung in beiden Ausdrücken gegenseitig wegheben: dieser Fehler ist bei $I^{h/2}_{\text{Trapez}}$ viermal so klein wie bei $I^h_{\text{Trapez}}$. Aus der Differenz von $4I^{h/2}_{\text{Trapez}}$ und $I^h_{\text{Trapez}}$ erhalten wir daher einen Ausdruck für das Integral, bei dem der Fehler auf vierte Ordnung reduziert ist:

$$T^h_1 = \frac{4\, I^{h/2}_{\text{Trapez}} - I^h_{\text{Trapez}}}{3} = I + \mathcal{O}(h^4) \tag{2.20}$$

Bei erneuter Intervallhalbierung wird der Fehler von $T_1$ als Näherung von $I$ auf $(1/2)^4 = 1/16$ reduziert:

$$T^h_2 = \frac{16\, T^{h/2}_1 - T^h_1}{15} = I + \mathcal{O}(h^6) \tag{2.21}$$

Allgemein erhält man (3. Übung):

$$\boxed{T^h_n = \frac{4^n\, T^{h/2}_{n-1} - T^h_{n-1}}{4^n - 1} = I + \mathcal{O}(h^{2(n+1)})} \tag{2.22}$$

Iteration dieses Verfahrens und Extrapolation auf $h \to 0$ erlaubt eine sehr genaue Berechnung von $I$.

$$\begin{array}{ccccc} I^h & & & & \\ & \searrow & & & \\ I^{h/2} & \to & T^h_1 & & \\ & \searrow & & \searrow & \\ I^{h/4} & \to & T^{h/2}_1 & \to & T^h_2 \\ & \vdots & & & \end{array} \tag{2.23}$$

# 3 Differentialgleichungen

Physik ist in weiten Teilen Naturbeschreibung mit Differentialgleichungen:

- Klassische Mechanik: Newtonsche Bewegungsgleichung ist eine gewöhnliche Differentialgleichung 2. Ordnung.
- Elektrodynamik: Maxwellgleichungen sind partielle Dgln. erster Ordnung.
- Quantenmechanik: Schrödingergleichung ist eine partielle Differentialgleichung 2. Ordnung.

Wir beginnen mit dem einfachsten Fall, einer gewöhnlichen Differentialgleichung erster Ordnung.

## 3.1 Gewöhnliche Differentialgleichung 1. Ordnung

$$\frac{dy}{dx} = f(x, y) \tag{3.1}$$

Der einfache Fall $\frac{dy}{dx} = f(x)$ kann durch Integration $y(x) = y(x_0) + \int_{x_0}^{x} dx\, f(x)$ gelöst werden. Wir interessieren uns daher vor allem für den Fall, bei dem die rechte Seite von $y$ oder von $x, y$ abhängt.
Man kann die Differentialgleichung graphisch darstellen, indem man jedem Punkt $(x, y)$ seine Steigung $f(x, y)$ zuordnet. Die Differentialgleichung definiert also ein Vektorfeld, bei dem jedem Punkt ein Einheitsvektor zugeordnet ist, dessen Richtung durch die Steigung $f(x, y)$ definiert ist. (Das Verhältnis von $y$-Komponente zu $x$-Komponente ist also $f(x, y)$.)
Die Grundidee ist jetzt: Ausgehend von einer gegebenen Randbedingung $y(x = 0) = a$ 'folge man lokal der Steigung' und finde $y(x)$, so dass stets die Differentialgleichung erfüllt ist. Endliche Schritte entlang $x$ ergeben endliche Schritte entlang $y$, deren Größe von $f(x, y)$ bestimmt ist.
Zur Formulierung der Differentialgleichung auf einem diskreten Gitter in $x$-Richtung nutzen wir die Näherung

$$\left(\frac{dy}{dx}\right) \approx \frac{y_{i+1} - y_i}{x_{i+1} - x_i} = \frac{y_{i+1} - y_i}{h} \;, x_{i+1} = x_i + h \;. \tag{3.2}$$

Damit erhält man die Gleichung

$$\boxed{y_{i+1} = y_i + h \cdot f(x_i, y_i)} \tag{3.3}$$

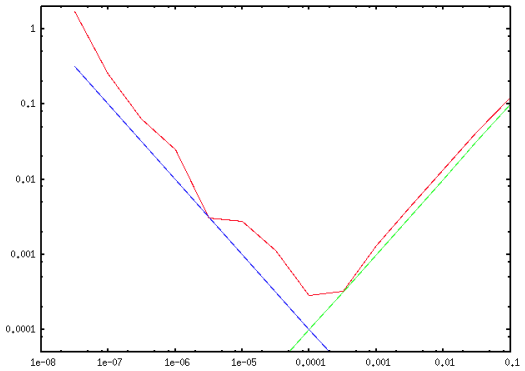



Abbildung 3.1: Fehler des Euler-Verfahrens in Abhängigkeit von der Schrittweite $h$ am Beispiel der Differentialgleichung $y\prime = y$ (rote Linie). Die geraden Linien (grün und blau) in dieser doppelt-logarithmischen Abbildung sind proportional zu $h$ und $h^{-1}$, entsprechen also Abbruchsfehler und Rundungsfehler.

bei der die rechte Seite vollständig durch die Werte von $(x_i, y_i)$ bestimmt ist. Gegeben eine Randbedingung $(x_0, y_0)$, kann man iterativ $y_1, y_2, \ldots$ bestimmen. Dies nennt man das Euler-Verfahren.

In der Praxis funktioniert das Euler-Verfahren leider oft nicht. Grundsätzlich gibt verschiedene Arten von Fehlern:

- Abbruchfehler: Die Taylor-Reihe wurde nach der ersten Ordnung abgebrochen.

$$\begin{aligned} y_{i+1} &= y_i + h\, f(x_i, y_i) + \frac{1}{2}h^2 \frac{d^2y}{dx^2} + \ldots \\ &= y_i + h\, f(x_i, y_i) + \mathcal{O}(h^2) \end{aligned} \tag{3.4}$$

  Der Fehler trägt schlimmstenfalls $\mathcal{O}(h^2) \cdot \frac{x - x_0}{h} = \mathcal{O}(h)$ zum Gesamtfehler bei.

- Rundungsfehler: Jede Addition wird mit endlicher Genauigkeit $\eta$ durchgeführt.

Damit erhalten wir den Gesamtfehler $\epsilon \approx \frac{\eta(x-x_0)}{h} + \beta h$. Das Minimum von $\epsilon$ ist die optimale Schrittweite. Wir minimieren $\epsilon$:

$$\frac{d\epsilon}{dh} = -\frac{\eta(x - x_0)}{h^2} + \beta \Rightarrow h \propto \sqrt{\eta} \tag{3.5}$$

Beide Arten von Fehlern können zu numerischen Instabilitäten führen: Das Ergebnis hat dann keinen Bezug mehr zur exakten Lösung. Ein Beispiel dafür ist:

$$\frac{dy}{dx} = -\alpha y \equiv f(y) \tag{3.6}$$

Das Euler-Verfahren gibt uns:

$$y_{n+1} = y_n + h\, f(x_n, y_n) = y_n - h\alpha y_n = (1 - h\alpha) y_n \tag{3.7}$$

Für $h > 1/\alpha$ oszilliert die Folge $y_n, \ldots$, für $h > 2/\alpha$ nimmt sogar die Amplitude dieser Oszillation zu.
Ein allgemeines $f(x, y)$ kann darüber hinaus an einer Stelle $x$ beliebig steil sein ($\partial f/\partial y|_x \gg 1$). Das heißt, Instabilitäten treten selbst bei beliebig kleinen $h$ auf.

## 3.2 Runge-Kutta-Verfahren

Die Idee ist wieder, $y_{i+1} = y(x_{i+1}) = y(x_i + h)$ um $x_i$ zu entwickeln. In erster Ordnung erhält man das Euler-Verfahren.
Entwickeln wir eine Ordnung weiter, so erhalten wir:

$$y_{i+1} = y_i + h \cdot f(x_i, y_i) + \frac{1}{2}h^2 y''(x_i, y_i) + \ldots \tag{3.8}$$

Betrachtet man $y''$ genauer, so erhält man:

$$\begin{aligned} y''(x_i, y_i) &= \left[\frac{d^2}{dx^2}y(x, y)\right]_{(x_i,y_i)} = \left[\frac{d}{dx}f(x, y)\right]_{(x_i,y_i)} \\ &= \left[\frac{\partial}{\partial x}f(x, y) + \frac{dy}{dx}\frac{\partial}{\partial y}f(x, y)\right]_{(x_i,y_i)} \\ &= \left[\frac{\partial}{\partial x}f(x, y) + f(x, y)\frac{\partial}{\partial y}f(x, y)\right]_{(x_i,y_i)} \end{aligned} \tag{3.9}$$

$$\tag{3.10}$$

Wir haben also:

$$y_{i+1} = y_i + h \cdot f(x_i, y_i) + \frac{1}{2}h^2\frac{\partial f}{\partial x}|_{x_i,y_i} + \frac{1}{2}h^2 f_i\frac{\partial f}{\partial y}|_{x_i,y_i} + \mathcal{O}(h^3) \tag{3.11}$$

Anstatt die partiellen Ableitungen numerisch zu berechnen, wählen wir schlau Punkte, an denen $f(x, y)$ ausgewertet wird, so dass das Ergebnis bis in 2. Ordnung mit der obigen Gleichung übereinstimmt. Ansatz:

$$y_{i+1} = y_i + \alpha_1 h f_i + \alpha_2 h f(x_i + \beta_1 h, y_i + \beta_2 h f_i) \tag{3.12}$$

Nun werden $\alpha_1, \alpha_2, \beta_1, \beta_2$ so angepasst, dass die beiden Gleichungen übereinstimmen. Taylorentwicklung ergibt:

$$f(x_i + \beta_1 h, y_i + \beta_2 h f_i) = f_i + h\beta_1\frac{\partial f}{\partial x} + h\beta_2 f_i\frac{\partial f}{\partial y} + \mathcal{O}(h^2) \tag{3.13}$$

Damit ist der Ansatz dann insgesamt:

$$y_{i+1} = y_i + \alpha_1 h f_i + \alpha_2 h f_i + \alpha_2\beta_1\left[\frac{\partial f}{\partial x}\right]_{x_1,y_i} h^2 + \alpha_2\beta_2\left[\frac{\partial f}{\partial y}\right]_{x_i,y_i} h^2 f_i \tag{3.14}$$

Ein Koeffizientenvergleich ergibt dann die Relationen $\alpha_1+\alpha_2 = 1, \alpha_2\cdot\beta_2 = \alpha_2\cdot\beta_1 = 1/2$. Man kann also z.B. $\alpha_1 = \alpha_2 = 1/2$ und $\beta_1 = \beta_2 = 1$ wählen. Dann hat man das Runge-Kutta-Verfahren 2. Ordnung (Heun-Verfahren):

$$\boxed{y_{i+1} = y_i + \frac{1}{2}hf_i + \frac{1}{2}hf(x_i + h, y_i + hf_i)} \tag{3.15}$$

Auch höhere Ordnungen kann man leicht berechnen, z.B. das Runge-Kutta Verfahren 4. Ordnung (RK4):

$$y_{i+1} = y_i + \frac{1}{6}(c_1 + 2c_2 + 2c_3 + c_4) \tag{3.16}$$

mit $c_1 = h \cdot f_i$; $c_2 = h \cdot f(x_i + \frac{1}{2}h, y_i + \frac{1}{2}c_1)$; $c_3 = h \cdot f(x_i + \frac{1}{2}h, y_i + \frac{1}{2}c_2)$; $c_4 = h \cdot f(x_i + h, y_i + c_3)$.

## 3.3 Differentialgleichungen höherer Ordnung

Viele Probleme in der Physik führen auf Differentialgleichungen zweiter Ordnung, z.B. die Newtonschen Bewegungsgleichungen. Durch geschicktes Einführen neuer Variablen kann eine Differentialgleichung höherer Ordnung in ein System *mehrerer* Differentialgleichungen niedrigerer Ordnung überführt werden.
Beispiel: Die Bewegungsgleichung $m\frac{d^2x}{dt^2} = F(x)$.
Wir führen als neue Variable den Impuls $p = m\frac{dx}{dt}$ ein. Dann lässt sich die Gleichung schreiben als:

$$\begin{array}{ll} \frac{dx}{dt} & = \frac{p}{m} \\ \frac{dp}{dt} & = F(x) \end{array} \tag{3.17}$$

Wir haben jetzt zwei gekoppelte Gleichungen erster Ordnung. Die Variablen $X = (x, p)$ lassen sich als Vektorelement des Phasenraums auffassen (Hamiltonmechanik).
Die vektorielle Darstellung hilft auch bei der numerischen Lösung:

$$\begin{array}{ll} x_{i+1} & = x_i + \frac{p(x_i)}{m}\Delta t \\ p_{i+1} & = p_i + F(x_i)\,\Delta t \end{array} \tag{3.18}$$

Die vektorielle Form lautet:

$$X_{i+1} = X_i + \begin{pmatrix} p_i/m \\ F(X_i) \end{pmatrix} \Delta t \tag{3.19}$$

Gewöhnliche Differentialgleichungen beliebig hoher Ordnung scheinen damit abgehandelt zu sein. Ihre Lösungen können trotz einfacher Form der Differentialgleichung jedoch sehr komplex sein. Dieser Effekt, der bei nichtlinearen Differentialgleichung auftreten kann, lässt sich anschaulich im Phasenraum des Problems diskutieren.

Wir betrachten Trajektorien im Phasenraum mit unterschiedlichen Anfangsbedingungen. Der Abstand ist zum Beispiel (euklidisch) $\Delta X = \sqrt{\Delta x^2 + \Delta p^2}$ mit der Zeitentwicklung $\Delta X(t) = e^{\lambda t}\,\Delta X(0)$. Dabei nennt man $\lambda$ den Ljapunov-Exponenten. Wir haben folgende Fälle:

1. Benachbarte Trajektorien nähern sich einander an: $\lambda < 0$

2. Benachbarte Trajektorien laufen parallel: $\lambda = 0$

3. Benachbarte Trajektorien laufen auseinander: $\lambda > 0$
   Kleine $\Delta X$ werden verstärkt. Eine Konsequenz ist, daß das Verhalten des Systems stark von den Anfangsbedingungen abhängt: Zwei Trajektorien, die lediglich kleine Unterschiede in den Anfangsbedingungen aufweisen, werden auf einer Zeitskala divergieren. Dabei kann (in der numerischen Berechnung) der Unterschied in Anfangsbedinungen durch Rundungsfehler entstehen, oder (im Experiment) in der Unkenntnis der exakten Anfangsbedingung begründet sein. Die ultimative Grenze ist hier durch die Unschärferelation gegeben.

   In Dimensionen des Phasenraums $d \geq 2$ kann weisen Trajektorien, die in unterschiedlicher Richtung benachbart sind, auch unterschiedliche Ljapunov-Exponenten auf. Die obige Analyse bezieht sich dann auf den größten Exponenten.

## 3.4 Partielle Differentialgleichungen (PDG)

Partielle Differentialgleichungen sind Differentialgleichungen, die Ableitungen in unterschiedlichen Variablen enthalten. Die Motivation, sich mit PDG zu beschäftigen, ist, dass viele grundlegende Gleichungen der Physik PDG sind. Beispiele sind die Poisson-Gleichung der Elektrostatik

$$\triangle\phi(\underline{x}) = \left(\partial_x^2 + \partial_y^2 + \partial_z^2\right)\phi(x,y,z) = \rho(\underline{x}) \ , \tag{3.20}$$

oder die Schrödinger-Gleichung der Quantenmechanik

$$i\hbar\,\partial_t\psi = -\frac{\hbar^2}{2m}\partial_x^2\psi(x,t) + V(x,t)\psi(x,t) \ . \tag{3.21}$$

PDGs treten immer dann auf, wenn klassische Felder in Raum und Zeit *lokal* beschrieben werden können; die Beziehung zwischen dem Feld an jedem gegebenen Raumzeitpunkt und Ableitungen der Felder an diesem Punk (lokale Grßen!) ergeben eine PDG. PDG treten daher auf bei der dynamischen Beschreibung von Flüssigkeiten, von chemische Reaktionen, der Dynamik der Atmosphäre oder der Dynamik von Sternenwolken.
Partielle Differentialgleichungen sind sowohl analytisch als auch numerisch oft sehr schwer zu lösen. Manchmal ist eine Rückführung auf mehrere gewöhnliche Differentialgleichungen durch Separation der Variablen möglich. Diese (einfachen) Fälle wollen wir im Folgenden allerdings nicht behandeln.
Grundsätzlich gibt es zwei Typen von PDG:

1. Anfangswertprobleme:
   Hier sind Randbedingungen und Anfangsbedingung, also der Zustand zu einem Zeitpunkt $t = t_0$ gegeben. Ein Beispiel hierfür ist eine schwingende Saite $y(x,t)$.
   Die Idee zur Lösung ist eine Rekursion von $t_0$ nach $t_0 + \Delta t$, von $t_0 + \Delta t$ nach $t_0 + 2\,\Delta t$ usw.

2. Randwertprobleme:
   Ein Beispiel hierfür ist die Elektro/Magnetostatik. Hier ist keine einfache rekursive Lösung möglich, da die Randwerte in allen Richtungen berücksichtigt werden müssen. Denn wie garantiert man, dass, wenn die Iterationen den Rand erreichen, auch der Randwert erreicht ist?

### 3.4.1 Anfangswertprobleme

Wir betrachten zunächst ein einfaches Beispiel

$$\frac{\partial u}{\partial t} = -v\frac{\partial u}{\partial x} \ . \tag{3.22}$$

Wie kennen die Lösung dieser Gleichung bereits, sie lässt sich allgemein lösen durch Funktionen der Form $f(x - vt)$.
Anhand dieses einfachen Beispiels wollen wir allgemeine Verfahren zur numerischen Lösung von PDG entwickeln. Dazu suchen wir zunächst nach einer geeigneten Darstellung von Funktionen (die ja prinzipiell jedem Punkt eines kontinuierlichen Raumes einen Wert zuordnen und daher unendlich viele Variablen zu ihrer Darstellung benötigen). Der erste Schritt besteht daher in einer Diskretisierung des Problems: wir führen ein Diskretisierungsgitter ein und suchen numerische Approximation der Lösung der PDG auf diesen Gitterpunkten. Zunächst approximieren wir die Ableitungen, die in der PDG auftreten durch von $u$ auf den Gitterpunkten:

$$\frac{\partial u}{\partial t} = \frac{u_j^{n+1} - u_j^n}{\Delta t} + \mathcal{O}(\Delta t^2) \tag{3.23}$$

$$\frac{\partial u}{\partial x} = \frac{u_{j+1}^n - u_{j-1}^n}{2\,\Delta x} + \mathcal{O}(\Delta x^2) \tag{3.24}$$

Die Diskretisierung nutzt Terme mit $(n + 1)$ und $n$: Einsetzen in (3.22) ergibt eine Iterationsformel:

$$u_j^{n+1} = u_j^n - \Delta t\, v\, \frac{u_{j+1}^n - u_{j-1}^n}{2\,\Delta x} \tag{3.25}$$

Ausgehen von $u(x, t = t_0)$ lässt sich durch Iteration das gesamte Gitter mit Werten für $u$ 'füllen'. Das Verfahren ist einfach, elegant, aber funktioniert (in diesem Fall, leider) nicht.
Eine Stabilitätsanalyse, entwickelt durch v. Neumann, ergibt:
In einer (linearen) PDG betrachten wir alle Koeffizienten als lokal konstant (dies entspricht einem sehr feinem Gitter).
Die Differenzmethode führt dann auf eine lineare Rekursionsvorschrift. Wir suchen nach Eigenmoden der Rekursionsvorschrift:

$$u_j^n = \xi^n \exp(ikj\,\Delta x) \tag{3.26}$$

mit Wellenvektor $k$. Für $|\xi| > 1$ haben wir ein exponentielles Wachstum der Amplitude und damit eine numerische Instabilität. Einsetzen von (3) in (2) ergibt:

$$\begin{aligned} \xi u_j^n &= u_j^n - \Delta t\, v \frac{e^{ik\,\Delta x} - e^{-ik\,\Delta x}}{2\,\Delta x} u_j^n \\ &= \left(1 - i\frac{\Delta t}{\Delta x}\sin(k\Delta x)\right) u_j^n \end{aligned} \tag{3.27}$$

$$\tag{3.28}$$

Also ist $\xi = \left(1 - i\frac{\Delta t}{\Delta x}\sin(k\,\Delta x)\right)$. Bei endlichen $\Delta t, k, \Delta x$ ist damit $|\xi| > 1$ und es liegt eine Instabilität vor.
Der einzige Ansatzpunkt, diese Katastrophe zu vermeiden, ist die Wahl der Diskretisierung einer partiellen Ableitung. So hätten wir z.B. statt der Diskretisierung $\partial_t u = \left(u_j^{n+1} - u_j^n\right)/\Delta t$ (asymmetrisch um $t$) die Wahl $\partial_t u = \left(u_j^{n+1} - u_j^{n-1}\right)/2\,\Delta t$ (symmetrisch) treffen können. Beide Diskretisierungsvorschriften sind in erster Ordnung von $\Delta t$ gleich.
Als Lax-Methode ist der folgende Vorschlag bekannt:

$$\frac{\partial u}{\partial t} = \frac{u_j^{n+1} - \frac{1}{2}\left(u_{j-1}^n + u_{j+1}^n\right)}{\Delta t} \tag{3.29}$$

Hier ist jetzt $|\xi| \leq 1$, es tritt keine Instabilität auf.

### 3.4.2 Randwertprobleme

Das Kernproblem ist, dass die Iteration in *einer* Richtung bei Randwertproblemen nutzlos ist. Die Werte am Rand müssen nämlich mit den finalen Iterationsschritte übereinstimmen. Statt iterativ in einer Richtung vorzugehen, müssen wir nun mit allen Gitterpunkten gleichzeitig rechnen. Die Idee ist also, ein Verfahren zu konstruieren, dass mit allen Punkten des Systems arbeitet. Ein Algorithmus berechnet nun aus einer Konfiguration an allen Gitterpunkte eine nächste Konfiguration. Die Iterationsvorschrift von einer Konfiguration zur nächsten ist dann so zu wählen, dass die Lösung der PDG Fixpunkt der Iterationsvorschrift ist. Dann können wir hoffen, dass der Algorithmus sich schrittweise der Lösung der PDG annähert. Zwei Probleme können auftreten:

1. Stabilitätsprobleme treten nur selten auf, da die Randwerte divergierende Lösungen verhindern.

2. Speicherplatz und Rechenzeit sind wichtiger. Es müssen nämlich stets alle Gitterpunkte berücksichtigt werden.

Ein Beispiel für Randwertprobleme ist die Poisson-Gleichung

$$\partial_x^2 u + \partial_y^2 u = \rho\ , \tag{3.30}$$

wobei z.B. das Potential $u = u(x, y)$ gesucht und die Ladungsdichte $\rho = \rho(x, y)$ sowie Randwerte gegeben sind. Der erste Schritt ist die Diskretisierung der Ableitungen durch

endliche Differenzen.
Wie diskretisiert man die zweite Ableitung? Betrachte $\frac{d^2 f}{dx^2}$ mit $x_i = x_0 + ih$. Nun machen wir eine Taylorentwicklung von $f$ an der Stelle $x_i$ und setzen $x_{i+1}$ und $x_{i-1}$ ein.

$$f(x_{i+1}) = f(x_i) + \underbrace{(x_{i+1} - x_i)}_{=(x_i+h)-x_i=h} f'(x_i) + \frac{1}{2}\underbrace{(x_{i+1} - x_i)^2}_{h^2} f''(x_i) + \mathcal{O}(h^2) \tag{3.31}$$

$$f(x_{i-1}) = f(x_i) + \underbrace{(x_{i-1} - x_i)}_{-h} f'(x_i) + \frac{1}{2}\underbrace{(x_{i-1} - x_i)^2}_{h^2} f''(x_i) + \mathcal{O}(h^2) \tag{3.32}$$

Addition ergibt:

$$f_{i+1} + f_{i-1} = 2f_i + f_i'' h^2 + \mathcal{O}(h^2) \tag{3.33}$$

$$\Rightarrow f''(x_i) = \frac{f_{i+1} + f_{i-1} - 2f_i}{h^2} + \mathcal{O}(h^2) \tag{3.34}$$

In zwei Dimensionen diskretisiert man dann:

$$x_i = x_0 + jh$$
$$y_i = y_0 + lh$$
$$u_{j,l} = u(x_j, y_l) \tag{3.35}$$
$$\tag{3.36}$$

Wobei $j = 0, \ldots, J$ und $l = 0, \ldots, L$.
Die Poisson-Gleichung, genähert durch endliche Differenzen:

$$u_{j+1,l} + u_{j-1,l} + u_{j,l+1} + u_{j,l-1} - 4u_{j,l} = h^2 \rho_{j,l} \tag{3.37}$$

Damit ist ein System von $L \cdot J$ (linearen) algebraischen Gleichungen in allen $u_{j,l}$ bis auf Randwerte (gegeben).
Konkrete Fälle:

$$j = l = 1: \; u_{2,1} + u_{0,1} + u_{1,2} + u_{1,0} - 4u_{1,1} = h^2 \rho_{1,1} \tag{3.38}$$

$$\Rightarrow u_{2,1} + u_{1,2} - 4u_{1,1} = h^2 \rho_{1,1} - u_{0,1} - u_{1,0} \tag{3.39}$$

$$j = 1, l = 2: \; u_{2,2} + u_{0,2} + u_{1,3} + u_{1,1} - 4u_{1,2} = h^2 \rho_{1,2} \tag{3.40}$$

$$\Rightarrow u_{2,2} + u_{1,3} + u_{1,1} - 4u_{1,2} = h^2 \rho_{1,2} - u_{0,2} \tag{3.41}$$

Um die Gleichungen kompakt aufzuschreiben, ist es nützlich einen Vektor zu definieren:

$$u_i = u_{j,l} \text{ mit } i = (j-1)L + l \tag{3.42}$$

Also: $\underline{u} = (u_{1,1}, u_{1,2}, \ldots, u_{1,L-1}, u_{2,1}, u_{2,2}, \ldots)^T$ Wir kommen damit auf eine Matrixgleichung:

$$\underline{\underline{A}}\,\underline{u} = \underline{b} \tag{3.43}$$

Für $(1,1)$ erhält man dann: $(\underline{\underline{A}})_{1,i'} = (-4,1,0,\ldots,1,0,\ldots,0,\ldots), b_1 = h^2\rho_{1,1} - u_{0,1} - u_{1,0}$.
Für $(1,2)$ hat man $(\underline{\underline{A}})_{2,i'} = (1,-4,1,0,\ldots,0,1,0,\ldots,0,\ldots), b_2 = h^2\rho_{1,2} - u_{0,2}$.
Man bekommt also Matrizen der Form:

$$\underline{\underline{A}} = \left(\begin{array}{cccc|cccc|cccc} -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & -4 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ \hline 1 & 0 & 0 & 0 & -4 & 1 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & -4 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & -4 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & -4 \end{array}\right) \tag{3.44}$$

(Lineare) PDG führen auf (große) lineare Gleichungssysteme.

- Terme wie $a(x,y)\,\partial_y^2 u(x,y)$ führen auf eine Matrix $\underline{\underline{A}}$ mit gleicher Struktur, allerdings sind alle Terme durch $a(x,y)$ moduliert.
- Nichtlineare PDG führen auf Systeme von nichtlinearen Gleichungen.

Haben wir die Dgl auf das lineare Gleichungssystem reduziert, so haben wir folgende Lösungsansätze (linearer Fall):

- Ist $\underline{\underline{A}}$ invertierbar, so existiert $\underline{\underline{A}}^{-1}$ und $\underline{u} = \underline{\underline{A}}^{-1}\,\underline{b}$. Hier kann man nutzen, dass $(\underline{\underline{A}})_{i,i'}$ für die meisten $(i,i')$ Null ist, was eine effiziente Invertierung ermöglicht.
- Wir spalten die Matrix in einen leicht invertierbaren Teil $\underline{\underline{E}}$ und einen Rest $-\underline{\underline{F}}$ auf:

  $$\underline{\underline{A}} = \underline{\underline{E}} - \underline{\underline{F}} \tag{3.45}$$

  In unserem Beispiel könnte $\underline{\underline{E}} = -4\underline{\underline{I}}$ eine gute Wahl sein. Es folgt dann: $\underline{\underline{E}}\,\underline{u} = \underline{\underline{F}}\,\underline{u} + \underline{b}$
- Wir führen nun wieder Iterationen durch:

  $$\boxed{\underline{u}^{n+1} = \underline{\underline{E}}^{-1}\left(\underline{\underline{F}}\,\underline{u}^n + \underline{b}\right)} \tag{3.46}$$

Diesen Iterationsschritt wiederholen wir bis zur Konvergenz. Konvergiert die Methode, so konvergiert sie auch gegen die Lösung, denn:

$$\underline{u}^{n+1} = \underline{u}^n \equiv \underline{u} \Rightarrow \underline{\underline{E}}\,\underline{u} = \underline{\underline{F}}\,\underline{u} + \underline{b} \Rightarrow (\underline{\underline{E}} - \underline{\underline{F}})\underline{u} = \underline{b} \Rightarrow \underline{\underline{A}}\,\underline{u} = \underline{b} \tag{3.47}$$

$\underline{u}^n$ relaxiert auf die (asymptotische) Lösung $\underline{u}$ hin. Die Methode trägt den Namen Relaxationsverfahren.

# 4 Lineare Algebra

Grundlegende Gleichungen der Physik sind oft linear. Auch nicht-lineare Probleme lassen sich meist "linearisieren", d.h. durch ein lineares Problem approximieren.
Es treten also lineare algebraische Gleichungen wie dieses Beispiel mit drei Gleichungen und drei Unbekannten auf:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}a_3 &= b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}a_3 &= b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}a_3 &= b_3 \end{aligned} \tag{4.1}$$

$M$ Gleichungen für $M$ Unbekannte lassen sich kompakt in Matrixform schreiben:

$$\underline{\underline{A}}\,\underline{x} = \underline{b} \tag{4.2}$$

Hier sind $(\underline{\underline{A}})_{ij} = a_{ij}$, $(\underline{x})_i = x_i$ und $(\underline{b})_i = b_i$.
Ein Beispiel war die Lösung der partiellen Differentialgleichung im letzten Kapitel. Weitere Beispiele sind:

1. Klassische Mechanik:
   Wir betrachten das Vibrationsspektrum eines komplexen "Moleküls", also ein Objekt aus $N/3$ Punktmassen, die miteinander wechselwirken und Auslenkungen $q_i$ um die Gleichgewichtslage $x_i^{GL}$ erfahren.

   $$x_i = x_i^{GL} + q_i \tag{4.3}$$

   Die potentielle Energie ist gegeben durch $U(q_1, q_2, \ldots, q_N) = \frac{1}{2}\sum_{i,j} q_i q_j a_{ij} + \mathcal{O}(q^3)$, die kinetische sei durch $T(q_i, \dot{q}_1, .., \dot{q}_N) = \frac{1}{2}\sum_{i,j} m_{ij}\dot{q}_i\dot{q}_j$ gegeben.
   Bewegungsgleichungen erhalten wir durch Einsetzen von $\mathcal{L} = T - U$ in die Euler-Lagrange-Gleichung $\frac{d}{dt}\frac{\partial}{\partial \dot{q}_i}\mathcal{L} = \frac{\partial}{\partial q_i}\mathcal{L}$:

   $$\sum_j m_{ij}\ddot{q}_j + \sum_j a_{ij}q_j = 0 \tag{4.4}$$

   Wir suchen oszillatorische Lösungen der Form $q_i(t) = y_i\, e^{-i\omega t}$:

   $$\sum_j a_{ij}y_i e^{-i\omega t} - \sum_j m_{ij}\omega^2 y_i e^{-i\omega t} = 0 \tag{4.5}$$

   Division durch $e^{-i\omega t}$ ergibt:

   $$(\underline{\underline{a}} - \omega^2\underline{\underline{m}})\underline{y} = 0 \tag{4.6}$$

Wenn $\underline{\underline{m}}$ invertierbar ist:

$$(\underline{\underline{m}}^{-1} \cdot \underline{\underline{a}})\, \underline{y} = \omega^2 \underline{y} \tag{4.7}$$

und wir erhalten eine Eigenwertgleichung für die Amplituden $\underline{y}$.
Wir finden $N$ Lösungen der Eigenwertgleichung. $\omega_1, \ldots, \omega_N$ beschreiben das Frequenzspektrum des Systems.
Die Dynamik des Systems lässt sich wiederum in Eigenmoden zerlegen:

$$\begin{aligned} \underline{q}(t) &= \sum_k b_k \underline{y}_k e^{-i\omega_k t} \\ \dot{\underline{q}}(t) &= \sum_k b_k \underline{y}_k (-i\omega_k) e^{-i\omega_k t} \end{aligned} \tag{4.8}$$

Bestimmen der Amplitude und Phase jeder Eigenmode $b_k \in \mathbb{C}$ aus den Anfangsbedingungen gibt uns die Dynamik $q(t)$ für alle Zeiten $t$. Bei $t = 0$ gilt zum Beispiel:

$$\begin{aligned} Re\left(\sum_k b_k \underline{y}_k\right) &= \underline{q}_0 \\ Re\left(\sum_k b_k (-i\omega_k) \underline{y}_k\right) &= \underline{v}_0 \end{aligned} \tag{4.9}$$

Wie zu erwarten haben wir also $2N$ lineare Gleichungen zur Bestimmung von $N$ komplexen Zahlen.

2. Quantenmechanik:
Die Quantenmechanik ist eine lineare Theorie; aus dem Superpositionsprinzip ergibt sich der lineare Hilbertraum. Auch die Zeitentwicklung ist linear, denn $i\hbar\partial_t |\psi\rangle = H |\psi\rangle$.
Wir betrachten ein System mit beliebiger Hamiltonfunktion, z.B. $H = -\frac{\hbar^2}{2m}\partial_x^2 + V(x)$. Gesucht sind die Eigenzustände $|\psi_m\rangle$ von $H$: $H |\psi_m\rangle = E_m |\psi_m\rangle$.
Wir wählen nun eine Darstellung der $|\psi_m\rangle$, indem wir eine orthonormale Basis $|n\rangle$ aufstellen, z.B. seien $|n\rangle$ die Eigenzustände des harmonischen Oszillators

$$\langle x|n\rangle = \psi_n(x) = \left(\frac{1}{\pi}\right)^{1/4} \frac{1}{\sqrt{2^n n!}} H_n(x) e^{-\frac{1}{2}x^2} \tag{4.10}$$

mit den Hermite-Polynomen $H_n(x)$.
Wir wenden den Projektionsoperator $\sum_n |n\rangle \langle n| = 1$ auf die Hamiltongleichung an und erhalten

$$\sum_n |n\rangle \langle n|H|\psi_m\rangle = E_m \sum_n |n\rangle \langle n|\psi_m\rangle \tag{4.11}$$

Projektion auf einen der Basisvektoren $\langle p|$ ergibt:

$$\sum_n \langle p|H|n\rangle \langle n|\psi_m\rangle = E_m \sum_n \langle p|n\rangle \langle n|\psi_m\rangle \tag{4.12}$$

Links stehen nun ein Matrixelement von $H$, nämlich $H_{pn} = \langle p| H |n\rangle$ und eine Komponente von $\psi_m$ in Darstellung durch $|n\rangle$: $v_n^{(m)} = \langle n|\psi_m\rangle$. Rechts nutzen wir aus, dass die $|n\rangle$ eine Orthonormalbasis bilden und $\langle p|n\rangle = \delta_{pn}$ ist.
Wir erhalten damit die Eigenwertgleichung:

$$\sum_n H_{pn} v_n^{(m)} = E_m v_p^{(m)} \tag{4.13}$$

Die Werte $v_n^{(m)}$ bilden eine Matrix $U_{nm} \equiv v_n^{(m)}$, die von der Basis $|\psi_m\rangle$ in die Basis $|n\rangle$ transformiert.

$$|\psi\rangle = \sum_m a_m |\psi_m\rangle = \sum_{m,n} a_m |n\rangle \langle n|\psi_m\rangle \tag{4.14}$$

$$\langle p|\psi\rangle = \sum_{m,n} a_m \langle p|n\rangle \langle n|\psi_m\rangle = \sum_m a_m \langle p|\psi_m\rangle = \sum_m a_m v_p^{(m)} = \sum_m U_{pm} a_m \tag{4.15}$$

Die Basisdarstellung ist i.A. unendlichdimensional. Es gilt also im Prinzip die Eigenwerte einer unendlichen Matrix zu berechnen. In der Praxis kommt es dann darauf an, die Basis so zu wählen, dass sie möglichst an das Problem 'angepasst' ist und $U_{mn}$ für $|m-n| \gg 1$ möglichst klein ist. Dann reicht eine endliche, möglichst kleine Zahl von Basisvektoren aus, um die $|\psi_m\rangle$ zu approximieren.

## 4.1 Typen von (numerisch zu lösenden) Problemen der linearen Algebra

Die unterschiedlichen Probleme der linearen Algebra lassen sich grob wie folgt klassifizieren:

- Lösung der Gleichung $\underline{\underline{A}}\,\underline{x} = \underline{b}$ nach einem unbekannten Vektor $\underline{x}$.
- Lösung der Gleichungen $\underline{\underline{A}}\,\underline{x}^{(i)} = \underline{b}^{(i)}$
- Berechnung der Inversen $\underline{\underline{A}}^{-1}$ einer Matrix $\underline{\underline{A}}$. Dies ist ein Spezialfall vorangegangen Falles mit $b_i^{(j)} = \delta_{ij}$
$$\underline{\underline{A}}\,\underline{\underline{A}}^{-1} = \underline{\underline{1}} \tag{4.16}$$
- Berechnung der Determinante
- Berechnung von Eigenwerten und Eigenvektoren

Wir suchen nach einem numerischen Verfahren, das diese Probleme löst, und beginnen mit der ersten Klasse von Problemen. Dort betrachten wir ein Fall, der zunächst recht künstlich anmutet, letztendlich aber zur allgemeinen Lösung führt.

## 4.2 Dreiecksmatrizen

Eine Matrix der Form

$$\underline{\underline{A}} = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \cdots \\ 0 & a_{22} & a_{23} & \cdots \\ 0 & 0 & a_{33} & \cdots \\ \vdots & \vdots & \vdots & \ddots \end{pmatrix} \tag{4.17}$$

heißt obere Dreiecksmatrix; alle Enträge $a_{ij}$ mit $i > j$ sind null. Als konkretes Beispiel mit $N = 3$ betrachten wir das Gleichungssystem $\underline{\underline{A}}\,\underline{x} = \underline{b}$ mit

$$\underline{\underline{A}} = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22} & a_{23} \\ 0 & 0 & a_{33} \end{pmatrix} \tag{4.18}$$

Es ist sinnvoll mit der 3. Zeile zu beginnen: $a_{33}x_3 = b_3 \Rightarrow x_3 = b_3/a_{33}$ lasst sich leicht lösen. Dann 2. Zeile: $a_{22}x_2 + a_{23}x_3 = b_2 \Rightarrow x_2 = (b_2 - a_{23}x_3)/a_{22}$ usw.
Wir können so das Gleichungssystem von unten aufrollen und erhalten mit jeder Zeile die Lösung für eine weitere Variable.
Die Determinante lässt sich dann auch leicht berechnen, sie ist für $N = 2$:

$$\det \begin{pmatrix} a_{22} & a_{23} \\ 0 & a_{33} \end{pmatrix} = a_{22}a_{33} - 0 \cdot a_{23} = a_{22}a_{33} \tag{4.19}$$

und für $N = 3$:

$$\det \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ 0 & a_{22} & a_{23} \\ 0 & 0 & a_{33} \end{pmatrix} = a_{11} \det \begin{pmatrix} a_{22} & a_{23} \\ 0 & a_{33} \end{pmatrix} = a_{11}a_{22}a_{33} \tag{4.20}$$

Für Dreiecksmatrizen nimmt die Determinante also eine sehr einfache Form an: Sie ist das Produkt der Diagonalelemente.

## 4.3 Erzeugung von Dreiecksmatrizen

Die meisten Probleme führen nicht auf Gleichungen der Form $\underline{\underline{A}}\,\underline{x} = \underline{b}$, bei denen $\underline{\underline{A}}$ eine Dreiecksmatrix ist. Durch einfache Umformungen lässt sich $\underline{\underline{A}}$ jedoch in eine Dreiecksmatrix überführen, ohne die Lösung $\underline{x}$ zu verändern.
Beispiel: $N = 3, M = 3$:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}a_3 &= b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}a_3 &= b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}a_3 &= b_3 \end{aligned} \tag{4.21}$$

Wir können ohne die Lösung $\{x_1, x_2, x_3\}$ zu verändern:

- Gleichungen mit Konstanten $\neq 0$ multiplizieren: zeilenweise Multiplikation auf $\underline{\underline{A}}, \underline{b}$.
- Gleichungen addieren und subtrahieren bzw. zeilenweise/spaltenweise Addition von $\underline{\underline{A}}, \underline{b}$.
- Zeilen der Matrix $\underline{\underline{A}}$ und gleichzeitig Elemente von $\underline{b}$ vertauschen.
- Spalten von $\underline{\underline{A}}$ und gleichzeitig Zeilen von $\underline{x}$ vertauschen.

Beispiel $N = 2$:

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} \tag{4.22}$$

Wir ziehen von der 2. Zeile das $a_{21}/a_{11}$-fache der ersten Zeile ab:

$$\begin{pmatrix} a_{11} & a_{12} \\ a_{21} - \frac{a_{21}}{a_{11}} a_{11} & a_{22} - \frac{a_{21}}{a_{11}} a_{12} \end{pmatrix} = \begin{pmatrix} a_{11} & a_{12} \\ 0 & a_{22} - \frac{a_{21}}{a_{11}} a_{12} \end{pmatrix} \tag{4.23}$$

Für $N > 2$ führt der Schritt auf:

$$\left(\begin{array}{c|ccc} a_{11} & a_{12} & a_{13} & \cdots \\ \hline 0 & & & \\ 0 & & A^{(1)} & \\ 0 & & & \end{array}\right) \tag{4.24}$$

Nach dem ersten Schritt ist (lediglich) die erste Spalte null (bis auf $a_{11}$). Mit der restlichen $(N-1) \times (N-1)$-Submatrix wiederholen wir den Schritt, nach dem auch die zweite Spalte (bis auf die ersten zwei Einträge) null ist. Nach $(N-1)$ Iterationen erhält man dann eine Dreiecksmatrix.
Dieses einfache Verfahren (Gauß-Jordan-Verfahren) ist problematisch, da die diagonalen Elemente klein werden können (ohne, dass die Matrix $\underline{\underline{A}}$ selbst singulär oder fast singulär ist).
Dieses Problem lässt sich jedoch recht leicht lösen ("Pivotisierung"): Vor jedem Schritt vertauschen wir Zeilen der (verbleibenden) Matrix so, dass das betragsmäßig größte Element der Spalte auf der Diagonalen liegt (teilweise Pivotisierung).
Vollständige Pivotisierung sucht nach dem größten Element der gesamten Restmatrix und vertauscht dann Zeilen und Spalten. In der Praxis führt teilweise Pivotisierung meist zu genauso guten Resultaten.
Werden jedoch *alle* Elemente der Restmatrix klein, ist die Matrix $\underline{\underline{A}}$ fast singulär. Ist $\underline{\underline{A}}$ singulär, so existiert auch keine nichttriviale Lösung.
Das Gauß-Jordan-Verfahren mit Pivotisierung ist ein einfaches Verfahren, das in den meisten Fällen sehr gut funktioniert. Es gibt eine Vielzahl weitergehender Methoden zur Lösung linearer Gleichungssysteme. Ein wichtiger Spezialfall sind sogenannte 'sparse matrices', bei denen die meisten Elemente null sind und für die Algorithmen mit geringem Speicherbedarf und schneller Laufzeit entwickelt wurden.

Das folgende Beispiel zeigt, dass Dreiecksmatrizen auch bei der Matrixinvertierung nützlich sind.
Wir betrachten eine Matrix

$$\underline{\underline{A}} = \begin{pmatrix} 2 & -3 \\ -1 & 2 \end{pmatrix} \tag{4.25}$$

Gesucht ist die inverse Matrix $\underline{\underline{A}}^{-1}$. Dazu führen wir die folgenden Gleichungssysteme ein:

$$\underline{\underline{A}} \begin{pmatrix} x_1^{(1)} \\ x_2^{(1)} \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \underline{b}^{(1)}; \;\; \underline{\underline{A}} \begin{pmatrix} x_1^{(2)} \\ x_2^{(2)} \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \underline{b}^{(2)} \tag{4.26}$$

Eine kompakte Schreibweise ist:

$$\begin{bmatrix} 2 & -3 & 1 & 0 \\ -1 & 2 & 0 & 1 \end{bmatrix} \to \begin{bmatrix} 2 & -3 & 1 & 0 \\ 0 & 1/2 & 1/2 & 1 \end{bmatrix} \to \begin{bmatrix} 2 & 0 & 4 & 6 \\ 0 & 1/2 & 1/2 & 1 \end{bmatrix} \to \begin{bmatrix} 1 & 0 & 2 & 3 \\ 0 & 1 & 1 & 2 \end{bmatrix} \tag{4.27}$$

Im ersten Schritt wurde das 1/2-fache der ersten auf die zweite Zeile addiert. Im zweiten Schritt addiert man das 6-fache der zweiten auf die erste Zeile. Im dritten Schritt wiederum normiert man die ersten zwei Spalten, indem man die erste Zeile halbiert und die zweite verdoppelt.
Unsere inverse Matrix ist jetzt $\underline{\underline{A}}^{-1} = \begin{pmatrix} 2 & 3 \\ 1 & 2 \end{pmatrix}$.

## 4.4 Die Schrödingergleichung

Als Anwendung der Matrixinvertierung kehren wir zurück zum Problem partieller Differentialgleichungen. Wir betrachten die zeitabhängige Schrödingergleichung mit beliebigem Potential $V(x)$ in einer Dimension

$$i\frac{\partial \Psi(x,t)}{\partial t} = \left(-\frac{1}{2}\frac{\partial^2}{\partial x^2} + V(x)\right)\Psi(x,t) \tag{4.28}$$

wobei die Einheiten so gewählt sind, dass $\hbar = m = 1$. Der Ansatz

$$\Psi(x,t) = \exp(-iHt)\Psi(x,t=0) \equiv T_t\Psi(x,t=0) \tag{4.29}$$

mit $H = -\frac{1}{2}\frac{\partial^2}{\partial x^2} + V(x)$ löst die Schrödingergleichung. $T_t$ wird als Zeitentwicklungsoperator bezeichnet.
Die Zeitentwicklung $T_t$ ist ein unitärer Operator, die Norm $\langle\Psi(t)|\Psi(t)\rangle$ bleibt erhalten:

$$\begin{aligned} \langle\Psi(t)|\Psi(t)\rangle &= \int dx \Psi^*(x,t)\Psi(x,t) = \int dx \Psi^*(x,t=0)e^{iHt}e^{-iHt}\Psi(x,t=0) \\ &= \int dx \Psi^*(x,t=0)\Psi(x,t=0) = 1 \end{aligned} \tag{4.30}$$

Wir lösen die Differentialgleichung nun numerisch mit dem Differenzenverfahren. Auf dem Gitter $t_j = j\Delta t; x_i = i\Delta x$ erhalten wir

$$\begin{aligned}\partial_t \Psi(x,t) &= \frac{\Psi_i(t_{j+1}) - \Psi_i(t_j)}{\Delta t} + \mathcal{O}(\Delta t^2) \\ \partial_x^2 \Psi(x,t) &= \frac{\Psi_{i-1}(t_j) - 2\Psi_i(t_j) + \Psi_{i+1}(t_j)}{\Delta x^2} + \mathcal{O}(\Delta x^4)\end{aligned} \tag{4.31}$$

Wir setzen die Differenzen in die Schrödingergleichung ein und erhalten:

$$\begin{aligned}\Psi_i(t_{j+1}) &= \Psi_i(t_j) + \Delta t \frac{\partial \Psi_i}{\partial t} + \mathcal{O}(\Delta t^2) \\ &\approx \Psi_i(t_j) + \frac{1}{i}\Delta t \left( -\frac{1}{2} \frac{\Psi_{i-1}(t_j) - 2\Psi_i(t_j) + \Psi_{i+1}(t_j)}{\Delta x^2} + V(x_i)\Psi_i(t_j) \right) \\ &= \sum_k \left( \delta_{ik} - i\Delta t \mathcal{H}^d_{ik} \right) \Psi_k(t_j)\end{aligned} \tag{4.32}$$

Vergleich der letzten und vorletzten Zeile liefert die diagonalen Matrixelemente $\mathcal{H}^d_{ii} = V(x_i) + \frac{1}{\Delta x^2}$ und die offdiagonalen Elemente $\mathcal{H}^d_{ik} = -\frac{1}{2\Delta x^2}$ wenn $i = \pm k$ und 0 sonst. Dieser Ausdruck entspricht dem Hamiltonoperator in diskreter Schreibweise und Ortsraumdarstellung. Die iterative Vorschrift (4.32) ist leicht zu implementieren, ist aber aus folgendem Grund problematisch: Die Norm eines Quantenzustandes, hier $\int dx\, \Psi^*(x,t)\Psi(x,t)$, ist zu jedem Zeitpunkt eins; nur so kann das Betragsquadrat der Wellenfunktion als Wahrscheinlichkeit interpretiert werden.
Was passiert unter dem obigen Algorithmus mit der Normierung?

$$\begin{aligned}\langle \Psi(t+\Delta t)|\Psi(t+\Delta t)\rangle &= \langle \Psi(t)|(\underline{\underline{I}} - i\Delta t \mathcal{H}^d)^\dagger (\underline{\underline{I}} - i\Delta t \mathcal{H}^d)|\Psi(t)\rangle \\ &= \langle \Psi(t)|(\underline{\underline{I}} + i\Delta t \mathcal{H}^d)(\underline{\underline{I}} - i\Delta t \mathcal{H}^d)|\Psi(t)\rangle \\ &= \langle \Psi(t)|(\underline{\underline{I}} + \Delta t^2 (\mathcal{H}^d)^2)|\Psi(t)\rangle \\ &= \langle \Psi(t)|\Psi(t)\rangle + \Delta t^2 \langle \Psi(t)|(\mathcal{H}^d)^2|\Psi(t)\rangle\end{aligned} \tag{4.33}$$

Über $\propto 1/\Delta t$ viele Zeitschritte entsteht ein Fehler der Ordnung $\Delta t$.
Um die Norm eines Zustandes $|\Psi\rangle$ zu erhalten, muss der Zeitentwicklungsoperator $T = (\underline{\underline{I}} - i\Delta t \mathcal{H}^d)$ unitär sein, d.h. $T^\dagger T = \underline{\underline{I}}$. Das ist leider nicht der Fall, denn $T^\dagger T = (\underline{\underline{I}} - i\Delta t \mathcal{H}^d)^\dagger (\underline{\underline{I}} - i\Delta t \mathcal{H}^d) = \underline{\underline{I}} + \Delta t^2 (\mathcal{H}^d)^2 \neq \underline{\underline{I}}$. Mögliche Abhilfe:

- Fehler von Ordnung $\Delta t$ eliminieren (höhere Ordnung). Das Problem wird aber nur verschoben auf kleinere $\Delta t$. Für Entwicklungen über kleine Zeitintervalle könnte es ausreichen.
- $e^{-i\mathcal{H}t}$ darstellen (benötigt Eigenzustände von $\mathcal{H}$)
- Das sogenannte Crank-Nicolson-Schema, das im folgenden diskutiert wird.

Gesucht wird ein neuer Zeitentwicklungsoperator $T$, der in erster Ordnung mit dem obigen $T_1$ übereinstimmt, jedoch exakt unitär ist. Vorschlag:

$$T_2 = \left(\underline{\underline{I}} + i\frac{\Delta t}{2}\mathcal{H}^d\right)^{-1}\left(\underline{\underline{I}} - i\frac{\Delta t}{2}\mathcal{H}^d\right) \tag{4.34}$$

In erster Ordnung stimmen $T_1$ und $T_2$ überein:

$$T_2 \approx \left(\underline{\underline{I}} - i\frac{\Delta t}{2}\mathcal{H}^d\right)\left(\underline{\underline{I}} - i\frac{\Delta t}{2}\mathcal{H}^d\right) = \underline{\underline{I}} - i\Delta t\mathcal{H}^d + \mathcal{O}(\Delta t^2) = T_1 + \mathcal{O}(\Delta t^2) \tag{4.35}$$

Zur Unitarität: Mit $(AB)^\dagger = B^\dagger A^\dagger$ ist

$$T_2^\dagger T_2 = \left(\underline{\underline{I}} - i\frac{\Delta t}{2}\mathcal{H}^d\right)^\dagger\left(\underline{\underline{I}} + i\frac{\Delta t}{2}\mathcal{H}^d\right)^{-1\dagger}\left(\underline{\underline{I}} + i\frac{\Delta t}{2}\mathcal{H}^d\right)^{-1}\left(\underline{\underline{I}} - i\frac{\Delta t}{2}\mathcal{H}^d\right) \tag{4.36}$$

Da $\mathcal{H}^\dagger = \mathcal{H}$, kommutieren die vier Terme dieses Ausdrucks. Wir betrachten Term 1 und 3

$$\left(\underline{\underline{I}} - i\frac{\Delta t}{2}\mathcal{H}^d\right)^\dagger\left(\underline{\underline{I}} + i\frac{\Delta t}{2}\mathcal{H}^d\right)^{-1} = \left(\underline{\underline{I}} + i\frac{\Delta t}{2}\mathcal{H}^d\right)\left(\underline{\underline{I}} + i\frac{\Delta t}{2}\mathcal{H}^d\right)^{-1} = \underline{\underline{I}} \tag{4.37}$$

Adjungiert man nun die Terme 2 und 4 findet man dasselbe Ergebnis wie zu den Termen 1 und 3, $T_2$ ist also *exakt* unitär.

- $\left(\underline{\underline{I}} + i\frac{\Delta t}{2}\mathcal{H}^d\right)$ wird durch Matrixinvertierung berechnet.
- In der Praxis nutzt man, dass $\mathcal{H}^d$ nur auf der Diagonalen und auf den beiden off-Diagonalen Einträge hat. ($\rightarrow$ LU-Dekomposition, O(N))
- Man kann zeigen, dass der Operator $T_2$ ausserdem auch in zweiter Ordnung in $\Delta t$ korrekt ist.

## 4.5 Eigenwertprobleme

Eigenwertprobleme treten in allen Bereichen der Physik auf, z.B.:

- klassische Mechanik: Vibrationsspektren, Hauptachsen des Trägheitstensors, Spektrum von Liapunov-Exponenten
- Elektrodynamik: Moden des elektromagnetischen Feldes
- Quantenmechanik: stationäre Zustände
- Quantenfeldtheorie: Eigenmoden eines Feldes $\hat{=}$ Teilchen

Die allgemeine Form des Eigenwertproblems ist:

$$\underline{\underline{A}}\,\underline{x} = \lambda \underline{x} \tag{4.38}$$

wobei $\lambda$ und $\underline{x}$ unbekannt sind. $\underline{\underline{A}}$ ist häufig symmetrisch ($a_{ij} = a_{ji}$) oder hermitesch ($a_{ij} = a^*_{ji}$).
Meist ist die Ähnlichkeitstransformation gesucht, die $\underline{\underline{B}}^{-1}\,\underline{\underline{A}}\,\underline{\underline{B}}$ diagonalisiert [$\underline{x}' = \underline{\underline{B}}^{-1}\underline{x}$].
Beispiel: Der verschobene harmonische Oszillator.
Die Hamiltonfunktion kann geschrieben werden als:

$$\mathcal{H} = \underbrace{\hbar\omega(a^\dagger a + 1/2)}_{\mathcal{H}^0} + \Theta(\underbrace{a + a^\dagger}_{\sqrt{\frac{2m\omega}{\hbar}}x}) \tag{4.39}$$

Dabei sind $a, a^\dagger$ die Erzeuger und Vernichter von Zuständen des 1d harmonischen Oszillators $\mathcal{H}^0 = 1/2 \cdot m\omega^2 x^2 + 1/2 \cdot p^2/m$.
$\mathcal{H}$ beschreibt einen 1d harmonischen Oszillator, dessen Potential entlang der $x$-Achse verschoben ist. Eine einfache Transformation der Operatoren: $a = \overline{a} - \Theta/\hbar m$ und $a^\dagger = \overline{a}^\dagger - \Theta/\hbar m$ führt auf $\mathcal{H}_0 = \hbar\omega(\overline{a}^\dagger\overline{a} + 1/2)$.

Wir wollen das Problem nicht durch Verschiebung lösen, sondern suchen nach einer allgemeinen Methode, die für beliebige Potentiale anwendbar ist. Seien $|n\rangle$ die stationären Zustände von $\mathcal{H}_0 = \hbar\omega(\overline{a}^\dagger\overline{a} + 1/2)$. Dann gilt:

$$\begin{aligned}
\langle n|\mathcal{H}|p\rangle =& \langle n|\mathcal{H}_0|p\rangle + \Theta\,\langle n|a|p\rangle + \Theta\,\langle n|a^\dagger|p\rangle \\
=& E_p^0\,\langle n|p\rangle + \Theta\,\langle n|\sqrt{p}|p-1\rangle + \Theta\,\langle n|\sqrt{p+1}|p+1\rangle \\
=& E_p^0\delta_{np} + \Theta\sqrt{p}\delta_{n,(p-1)} + \Theta\sqrt{p+1}\delta_{n,(p+1)}
\end{aligned} \tag{4.40}$$

Wir erhalten also eine Matrix der Form:

$$\begin{pmatrix}
E_0^0 & \Theta & & & 0 & \cdots \\
\Theta & E_1^0 & \sqrt{2}\Theta & & & \\
 & \sqrt{2}\Theta & E_2^0 & \sqrt{3}\Theta & & \\
0 & & \sqrt{3}\Theta & E_3^0 & \sqrt{4}\Theta & \\
\vdots & & & & \ddots &
\end{pmatrix} \tag{4.41}$$

Diagonalisierung dieser Matrix ergibt dann das Energiespektrum von $\mathcal{H}$ (Hier identisch mit dem von $\mathcal{H}_0$).

### 4.5.1 Numerische Behandlung von Eigenwertproblemen

Gegeben sei die Eigenwertgleichung

$$\underline{\underline{A}}\,\underline{x} = \lambda\underline{x}\ , \tag{4.42}$$

auch darstellbar als

$$(\underline{\underline{A}} - \lambda\underline{\underline{I}})\underline{x} = 0\ . \tag{4.43}$$

Die Gleichung sieht aus wie ein Standardproblem der linearen Algebra. Jedoch ist hier $\lambda$ unbekannt und $(\underline{\underline{A}} - \lambda \underline{\underline{I}})$ per Konstruktion singulär.
Für einen Eigenwert $\lambda$ muss $\det(\underline{\underline{a}} - \lambda \underline{\underline{I}}) = 0$ sein. Es ergibt sich also eine polynome Gleichung in den gesuchten Eigenwerten. Ein Beispiel ist:

$$\underline{\underline{A}} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \Rightarrow \det(\underline{\underline{A}} - \lambda \underline{\underline{I}}) = (\lambda^2 - 1) = 0 \tag{4.44}$$

Also sind die Eigenwerte $\lambda_{1,2} = \pm 1$.
Es ist natürlich verlockend, dieses Vorgehen zu verallgemeinern und Eigenwerte zu bestimmen, indem man das sogenannte charakteristische Polynom numerisch löst. Für sehr große Matrizen funktioniert das Verfahren aber sehr schlecht: Hier können die Eigenwerte 'dicht' liegen. Damit sind die Nullstellen des charakteristischen Polynoms dann sehr schwer zu finden.

### 4.5.2 Jakobi-Algorithmus für symmetrische Matrizen

Die meisten in der Praxis genutzten Verfahren basieren auf einfachen Transformationen, die mit jeder Iteration die Matrix ein wenig 'diagonaler' machen.
Idee: Sukzessive Drehungen um verschiedene Achsen des Koordinatensystems: Jede einzelne Drehung soll off-diagonale Terme möglichst klein machen.
Skizze eines Algorithmus: Wähle zwei Richtungen $q, p$. Diese definieren eine Ebene. Wie sehen Drehungen in dieser Ebene aus? Sei $p = 1, q = 2$:

$$\underline{\underline{B}}^{1,2} = \begin{pmatrix} c & s & & & 0 \\ -s & c & & & \\ & & 1 & & \\ & & & 1 & \\ 0 & & & & \ddots \end{pmatrix} \tag{4.45}$$

Dabei ist $c \equiv \cos\theta$ und $s \equiv \sin\theta$. Nach einer Iteration $A^{(1)} = (\underline{\underline{B}}^{pq})^{-1} \cdot \underline{\underline{A}} \cdot \underline{\underline{B}}^{pq}$ haben wir eine Matrix, die nur in den Komponenten mit $p$ und $q$ verändert ist. Genauer gesagt gilt:

$$\begin{aligned} a'_{pp} &= c^2 a_{pp} + s^2 a_{qq} - 2csa_{pq} \\ a'_{pq} &= (c^2 - s^2) a_{pp} + sc(a_{pp} - a_{qq}) \\ a'_{qq} &= c^2 a_{qq} + s^2 a_{pp} + 2sca_{pq} \end{aligned} \tag{4.46}$$

$a'_{qp}$ kann durch Lösen von $a'_{qp} = 0$ nach $\theta$ exakt null gesetzt werden. Dann wählen wir neue Richtungen $p, q$ und wiederholen den Schritt.

# 5 Zufallszahlen

Wofür brauchen wir Zufallszahlen?

- Simulationen des Zerfalls von Atomen, von nuklearen Kettenreaktionen etc.
- Simulation eines Systems, das an ein Wärmebad gekoppelt ist, dessen Dynamik also von zufälligen thermischen Fluktuationen abhängt
- Berechnung von Integralen mit komplizierten Randbedingungen und in hochdimensionalen Räumen. Einfachstes Beispiel: Wir betrachten einen in ein Quadrat bekannter Größe eingeschriebenen Kreis. Nun wählen wir zufällig Punkte im Quadrat (bei gleichförmiger Wahrscheinlichkeitsverteilung). Betrachtet man nun das Verhältnis der Zahlen der Punkte innerhalb und außerhalb des Kreises, so kann man die Fläche des Kreises abschätzen.
- allgemein: zufällige Dynamik modelliert deterministische Dynamik bei unvollständiger Information

## 5.1 Zufallszahlengeneratoren

Es gibt viele Möglichkeiten Zufallszahlen zu generieren:

- Physikalische Generatoren
  - Deterministische Systeme sind Systeme, die sensitiv von den Anfangsbedingungen abhängen: Würfel, Münzwurf, Lotteriemaschine, Rauschen im elektrischen Schaltkreis,...
  - Atomarer Zerfall und photoelektrischer Effekt folgen der Zufälligkeit der Quantenmechanik.

  Diese Generatoren sind meist zu langsam für praktische Anwendungen.
- Algorithmische Generatoren erzeugen Pseudozufallszahlen. Ein (etwas provokantes) Beispiel ist $3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5$. Dies sind die Stellen von $\pi$, die streng deterministisch berechnet werden können. Wer $\pi$ jedoch nicht kennt mag in dieser Zahlenfolge nichts regelmässiges finden. Wir stellen an solche (Pseudo)Zufallszahlen folgende Anforderungen:
  - Sie sollen einer wohldefinierten Verteilung gehorchen.
  - Sie sollen statistisch unkorreliert sein, also $P(x, y) = P(x)P(y)$ usw. in höheren Dimensionen.

- Sie sollen schnell und reproduzierbar berechenbar sein.
- Die Folge von Zufallszahlen soll nicht korreliert sein mit der Ausgabe des Programms, in dem wir die Zufallszahlen verwenden.

Der letzte Punkt macht klar, warum gute (Pseudo)Zufallszahlengeneratoren schwer zu konstruieren sind: Ein Generator der für eine bestimmte Anwendung zufriedenstellend ist, mag in einer anderen den letzten Punkt verletzen.

Idee: Wir suchen eine (einfache) Funktion $f(x)$, so dass die Sequenz $x_{i+1} = f(x_i)$ für alle Startwerte (seeds) die obigen Bedingungen möglichst gut erfüllt.
Für einen egebenen Startwert ist die Sequenz streng deterministisch. Wenn allerdings die Sequenz stark vom Startwert abhängt, können trotzdem die Anforderungen an einen guten Generator erfüllt sein.
Beispiel: Der 'Multiplicative linear congruential generator'(MLCG) ist zwar veraltet, doch ein Klassiker.

$$\boxed{x_{i+1} = (a\, x_i) \mod m} \tag{5.1}$$

mit $0 \leq x \leq m-1$. Man sieht schnell, dass es gute und schlechte Parameterwerte für $a$ und $m$ gibt. Eine schlechte wäre z.B. $a = 12$ und $m = 143$, denn: $x_{i+1} = 12x_i \mod 143$, $x_{i+2} = 12^2 x_i \mod 143 = x_i$. Eine gute Wahl ist $a = 16807, m = 2147483647$.

Als Test, wie gut ein (Pseudo-)Zufallszahlengenerator geeignet ist, kann man die Zufallszahl $x_{i+1}$ gegen ihren Vorgänger $x_i$ auftragen und beobachtet oft, dass die Punkte auf Linien liegen. Noch deutlicher wird es, wenn man $x_{i+2}$ gegen $x_{i+1}$ und $x_i$ plottet. Marsaglia stellte fest, dass diese auf höchstens $m$ Ebenen, bei ungeschickter Wahl auch weniger liegen (1968). Dies ist problematisch z.B. bei der Verteilung von Eigenwerten von Zufallsmatrizen. Marsaglia führte daher als zusätzliche Tests die 'Die hard battery of tests' ein, die ein guter Generator bestehen muss (1995).
Ein besserer multiplikativer Generator, der alle diese Tests besteht ist der 'Multiply with carry' (MWC, carry=Übertrag):

$$x_{i+1} = a + (x_i \& [2^{32} - 1]) + (x_i \gg 32) \tag{5.2}$$

Eine gute Wahl für den Parameter ist: $a = 4294957665$.

## 5.2 Verteilungen von Zufallszahlen

Die genannten Generatoren produzieren ganzzahlige (Pseudo-)Zufallszahlen, die gleichverteilt innerhalb eines Intervalls liegen (sollten). Damit lassen sich leicht auf einem Intervall $[0, 1]$ verteilte reelle Zahlen generieren (bis auf Diskretisierung durch endliche $m$).
Wie kommt man aber auf Gauß-, Poisson-, exponentiell, etc. verteilte Zufallszahlen? Wir suchen nach Verfahren um Zufallszahlen mit beliebige Verteilungen zu generieren.

- $x$ sei gleichverteilt in $[0,1]$. Wir suchen eine Abbildung $y = y(x)$, so dass die $y$ mit gegebener Verteilung $P(y)$ auftreten. Der Ansatz ist, dass die Wahrscheinlichkeiten $\int_x^{x+\alpha} dx\, P(x) = \int_{y(x)}^{y(x+\alpha)} dy\, P(y)$ gleich sein müssen. Damit erhält man:

$$P(y) = P(x) \left| \frac{dx}{dy} \right| \tag{5.3}$$

  Dies führt uns wegen $P(x) = 1$ auf die Differentialgleichung $dx/dy = P(y)/P(x) = P(y)$. Deren Lösung ist:

$$x(y) = \int_{-\infty}^{y} dy' P(y') \equiv F(y) \tag{5.4}$$

  und wir erhalten:

$$\boxed{y(x) = F^{-1}(x)} \tag{5.5}$$

  Dies funktioniert jedoch offenbar nur dann gut, wenn die Wahrscheinlichkeitsfunktion explizit integrierbar ist.
  Als Beispiel suchen wir exponentiell verteilte Zufallszahlen: $P(y) = e^{-y} \Rightarrow |dx/dy| = e^{-y} \Rightarrow x(y) = e^{-y} \Rightarrow y(x) = -\ln(x)$.

- Ist $F(y)$ nicht in geschlossener Form lös- und invertierbar, kann man $P(y)$ numerisch integrieren und invertieren.

- Die Verwerfungsmethode (rejection sampling, von Neumann-Methode) kann genutzt werden, wenn die Transformationsmethode aufgrund der Integration/Inversion nicht praktikabel ist.
  Die Idee ist, Punkte (2 dim.) zu generieren, die in der Fläche unter $P(y)$ gleichverteilt sind. Nimmt man jetzt nur die $y$-Komponenten, so erhält man neue Zufallszahlen $y$, die nach $P(y)$ verteilt sind.
  Wir implementieren dieses Verfahren, indem wir eine Funktion $g(y)$ so wählen, dass $g(y) \geq P(y)$ für alle $y$. Dann wählen wir Punkte, die unter $g(y)$ gleichverteilt sind. Nun 'akzeptieren' wir einen Punkt an der Stelle $y$ mit der Wahrscheinlichkeit $P(y)/g(y)$ und 'verwerfe' ihn sonst. Den Schritt 'mit einer bestimmten Wahrscheinlichkeit akzeptieren' kann man nur durch weitere Ziehung einer Zufallszahl realisieren.
  Dieses Verfahren funktioniert zwar immer, ist aber sehr ineffizient für $P(y) \ll g(y)$.

- Die Metropolis-Methode ähnelt der Verwerfungsmethode, produziert jedoch eine Sequenz korrelierter Zufallsvariablen, deren Verteilung (Ein-Punkt-Verteilung) gegen eine beliebige gegebene Verteilung $P(y)$ strebt. Es handelt sich um eine künstliche Dynamik, die (nach langen Zeiten) eine bestimmte statistische Verteilung erreicht: $y_{i+1}$ hängt von $y_i$ ab, $y_0, y_1, \ldots, y_N$ sind also (korrelierte) Zufallszahlen, die Verteilung von $y_N$ ist asymptotisch $P(y)$.

  Die Sequenz wird generiert durch $\bar{y}_{i+1} = y_i + h\xi|_{i+1}$, wobei die maximale Schrittweite $h$ und $\xi$ gleichverteilt in $[-1,1]$ sei. Der 'Kandidat' $\bar{y}_{i+1}$ wird akzeptiert mit

Wahrscheinlichkeit $\min(1, P(\bar{y}_{i+1})/P(y_i))$.
Im Detail: Wähle $x$ gleichverteilt im Intervall $[0,1]$. Ist $x \leq P(\bar{y}_{i+1})/P(y_i)$, dann wird der Schritt akzeptiert und $y_{i+1} = \bar{y}_{i+1}$. Andernfalls wird der Schritt verworfen und $y_{i+1} = y_i$.
Man kann zeigen, dass die Verteilung von $y_i$ für $i \to \infty$ gegen $P(y)$ geht:
Im Gleichgewicht gleicht der Fluss der Wahrscheinlichkeit von $y \to y'$ dem Fluss der Wahrscheinlichkeit von $y' \to y$. Damit ist die Wahrscheinlichkeit, das System im Zustand $y$, im nächsten Schritt aber in den Zustand $y'$ wechselnd zu finden. Das Gleichgewicht ist erreicht, wenn für alle $y, y'$ gilt:

$$P(y \to y')\, P_G(y) = P(y' \to y)\, P_G(y') \tag{5.6}$$

Daraus folgt, dass für die Gleichgewichtsverteilung $P_G$ gelten muss:

$$\frac{P_G(y)}{P_G(y')} = \frac{P(y' \to y)}{P(y \to y')} = \frac{\frac{P(y)}{P(y')}}{1} = \frac{P(y)}{P(y')} \tag{5.7}$$

Da $P(y)$ und $P_G(y)$ normiert sind, muss $P_G(y) = P(y)$ sein.

## 5.3 Monte-Carlo Methoden in der Physik

Der Ausdruck *Monte Carlo* (MC), benannt nach dem Mittelmeerort mit seinen Spielcasinos, bezieht sich auf die Verwendung von Zufallszahlen um deterministische Größen zu berechnen, wie z.B. die Integration der Kreisbahn zu Beginn dieses Kapitels. Im folgenden zeigen wir, dass MC-Methoden besonders mächtig zur Berechnung hochdimensionaler Integrale sind. Solche Integrale treten in Systemen mit einer großen Zahl von Freiheitsgraden auf.

- statistische Physik und Thermodynamik, Gase, Festkörper, ...,
- komplexe Moleküle, Polymere, ...,
- Quantenfelder
- Interdisziplinäre Anwendungen: Optimierungsprobleme, neuronale Netze,....

Warum sind nun MC-Methoden gerade in hohen Dimensionen so nützlich? Dazu betrachten wir erst ein eindimensionales Integral

$$S = \int_0^1 dx\, f(x) \tag{5.8}$$

Wir wenden die Trapezregel mit $M$ Punkten an und erhalten:

$$S = \frac{1}{M} \sum_{i=1}^{M} \omega_i f(x_i) + \mathcal{O}(1/M^2) \tag{5.9}$$

Mit der MC-Methode nehmen wir $M$ gleichmäßig verteilte Zufallszahlen und setzen sie in die Funktion ein:

$$S = \frac{1}{M}\sum_{i=1}^{M} f(x_i) + \mathcal{O}(1/\sqrt{M}) \tag{5.10}$$

$1/M \sum f(x_i)$ ist eine Zufallszahl mit Mittelwert

$$\frac{1}{M}\sum_n \int_0^1 dx_n f(x_n) = \frac{1}{M}\sum_n \langle f\rangle = \langle f\rangle = S \tag{5.11}$$

Wir wollen nun die Varianz bestimmen. Das zweite Moment ist:

$$\begin{aligned}\int_0^1 dx_1 \ldots dx_n \left(\frac{1}{M}\sum_n f(x_n)\right)^2 &= \int_0^1 dx_1 \ldots dx_n \frac{1}{M^2}\sum_{m,n} f(x_m)f(x_n) \\ &= \frac{1}{M^2}\int_0^1 dx_1 \ldots dx_n \left[\sum_n f^2(x_n) + \sum_{m\neq m} f(x_m)f(x_n)\right] \\ &= \frac{M}{M^2}\langle f^2\rangle + \frac{M(M-1)}{M^2}\langle f\rangle^2 \end{aligned} \tag{5.12}$$

Die Varianz erhält man dann, indem man vom zweiten Moment das Quadrat des Mittelwertes abzieht. Man erhält

$$\sigma^2 = \frac{1}{M}\left(\langle f^2\rangle - \langle f\rangle^2\right) \ . \tag{5.13}$$

In einer Dimension ist die Trapezregel (oder eine andere Methode) vorteilhafter, da der Fehler $\propto 1/M^2$ sehr viel kleiner ist als der der MC-Methode $\propto 1/\sqrt{M}$.
In höheren Dimensionen $d > 1$ ist der Fehler der Trapezregel

$$\mathcal{O}(h^2) = \mathcal{O}(1/M^{2/d}) \tag{5.14}$$

Der Fehler der MC-Methode ist aber immer $\mathcal{O}(1/\sqrt{M})$.
Die beiden Methoden sind etwa gleich mächtig bei der 'kritischen' Dimension $d = d_c$:

$$M^{2/d_c} = \sqrt{M} \Rightarrow d_c = 4 \ . \tag{5.15}$$

Oberhalb dieser Dimension sind MC-Methoden der Trapezregel überlegen.

### 5.3.1 Ein 'teaser' statistische Physik: Die Boltzmann-Verteilung

Die wichtigste Anwendung von MC-Methoden in der Physik liegt in der Beschreibung statistischer Systeme. Hierzu muss an dieser Stelle das Kernresultat der statistischen Physik vorweggenommen werden. Ein mechanisches System, das durch $N$ Variablen $(x_1, x_2, \ldots, x_N)$ beschrieben ist (Koordinaten, Impulse,...) befinde sich im Gleichgewicht mit einem Wärmebad mit Temperatur $T$. Die Wahrscheinlichkeitsdichte, das System in

einem Zustand mit Werten der Variablen gleich $x_1, x_2, \ldots, x_N$ anzutreffen, ist gegeben durch die Boltzmann-Verteilung

$$\boxed{P(x_1, \ldots, x_N) = \frac{e^{-\mathcal{H}(x_1,\ldots,x_N)/k_B T}}{Z}}\,. \tag{5.16}$$

Dabei ist $k_B \approx 1.38 \times 10^{-23} JK^{-1}$ die Boltzmann-Konstante, $Z$ eine Normierungskonstante und $\mathcal{H}(x_1, \ldots, x_N)$ die Hamiltonfunktion des Systems.
Unser Ziel ist nun die numerische Berechnung von Mittelwerten unter der Boltzmann-Verteilung von Größen $f(x_1, \ldots, x_N)$, die von dern Variablen $(x_1, x_2, \ldots, x_N)$ abhängen

$$\int dx_1 \ldots dx_N P(x_1, \ldots, x_N) f(x_1, \ldots, x_N) = \frac{1}{M} \sum_{n=1}^{M} f(x_1^n, \ldots, x_N^n) \tag{5.17}$$

wobei die $\underline{x}^n$ aus der Boltzmann-Verteilung gezogen werden.
Die Zufallszahlen $\underline{x}^1, \underline{x}^2, \ldots$, die der Boltzmann-Verteilung gehorchen, werden mittels der Metropolis-Methode generiert. Dazu gehen wir folgendermaßen vor:

1. Wähle eine Variable $x_i$ mit $i = 1, \ldots, N$ und bestimme $\bar{x}_i^{n+1} = x_i + h\xi^n$
2. Akzeptiere den Schritt mit Wahrscheinlichkeit

$$\frac{e^{-\beta \mathcal{H}(x_1^n, \ldots, \bar{x}_i^{n+1}, \ldots, x_N^n)}}{e^{-\beta \mathcal{H}(x_1^n, \ldots, x_i^n, \ldots, x_N^n)}} = e^{-\beta \Delta \mathcal{H}} \tag{5.18}$$

   wobei $\beta = 1/k_b T$ und $\Delta\mathcal{H} = \mathcal{H}(x_1^n, \ldots, \bar{x}_i^{n+1}, \ldots, x_N^n) - \mathcal{H}(x_1^n, \ldots, x_i^n, \ldots, x_N^n)$.
3. Wird der Schritt akzeptiert, ist $x_l^{n+1} = x_l^n$ für $l \neq i$ und $x_l^{n+1} = \bar{x}_i^{n+1}$.

Schritte, die die Energie des Systems verringern ($\Delta\mathcal{H} < 0$), werden immer akzeptiert; Schritte, die die Energie erhöhen, werden mit der Wahrscheinlichkeit $e^{-\beta\Delta\mathcal{H}} < 1$ akzeptiert.

### 5.3.2 Das Ising-Modell des Magnetismus

Die magnetischen Eigenschaften eines Festkörpers entstehen aus der gegenseitigen Wechselwirkung einer großen Zahl 'elementarer' magnetischer Momente.
Modell: Ein regelmäßiges Gitter aus 'Spins' $\{s_i\}_{i=1,\ldots,N}$, die jeweils die Werte $s_i = \pm 1$ annehmen können. In zwei Dimensionen haben wir z.B. $s = +1 \hat{=} \uparrow$ und $s = -1 \hat{=} \downarrow$. Nun nimmt man für $\uparrow\downarrow$ (Spins sind antiparallel) eine hohe Energie und für $\uparrow\uparrow$ und $\downarrow\downarrow$ (Spins sind parallel) eine niedrige Energie an.
Die Hamiltonfunktion ist gegeben durch:

$$\mathcal{H} = -J \sum_{\langle i,j \rangle} s_i s_j - B \sum_i s_i \tag{5.19}$$

dabei bezeichnet $\langle i, j \rangle$ die Summe über alle Paare von Gitterplätzen $(i, j)$, die auf dem Gitter benachbart sind. $J$ ist die Kopplungskonstante und gibt an, wie stark benachbarte Spins miteinander wechselwirken. $B$ ist ein externes magnetisches Feld.

### 5.3.3 Komplexe Optimierungsprobleme und 'simulated annealing'

Die Boltzmann-Verteilung $1/Z \cdot e^{-H/k_B T}$ weist Konfigurationen mit hoher Energie niedrige Wahrscheinlichkeiten und Konfigurationen mit niedriger Energie hohe Wahrscheinlichkeiten zu. Im Grenzfall $T \to 0$ und damit $\beta = 1/k_B T \to \infty$ ist liegt das gesamte Wahrscheinlichkeitsmaß auf den Konfigurationen mit minimaler Energie $H$ (Grundzustände).
Diese Idee kann zur Lösung komplexer Optimierungsprobleme genutzt werden. Ein klassisches Beispiel ist das sogennante 'Problem des Handlungsreisenden' ('traveling salesman problem', kurz TSP): Ein Handlungsreisender soll $N$ Städte besuchen, jede Stadt einmal, und dann zum Ausgangsort zurückkehren. Welche Route hat die minimale Gesamtstrecke?

| | | |
|---|---|---|
| $N$ Städte | $1, 2, 3, \ldots, N$ | |
| Route | $i_1, i_2, i_3, \ldots, i_N$ | $i_{N+1} = i_1$ |
| Distanzen zwischen den Städten | $d_i j$ | |
| Gesamtstrecke | $D = \sum_{n=1}^{N} d_{i_n, i_{n+1}}$ | |

Man stellt fest, dass die Zahl der möglichen Strecken $\propto N!$ ist. Stures Ausprobieren aller möglichen Reiserouten würde als zu einem Rechenaufwand führen der exponentiell mit der Zahl der Städte anwächst.
Dieses Problem gehört zu einer großen Klasse von Problemen, die aufeinander abbildbar sind und als $NP$-harte Probleme bekannt sind. Vermutlich existier für diese Probleme kein deterministischer Algorithmus mit einer Laufzeit die polynomial $N$ wächst. Existenz oder Nichtexistenz eines solchen Algorithmus ist Kernproblem der sogenannten Komplexitätstheorie.
Wir definieren die Hamiltonfunktion des Systems mit Variablen $i_1, i_2, \ldots, i_N$ als

$$H(i_1, \ldots, i_N) = D \ . \tag{5.20}$$

Die Variablen $i_1, i_2, \ldots, i_N$ geben die Reihenfolge der Städte einer bestimmten Route an. Routen mit kurzen Gesamtstrecken wird also eine niedrige Energie zugeordnet.
Wenn wir jetzt Konfigurationen zufällig generieren könnten, deren Verteilung der Boltzmann-Verteilung gehorcht, so erhalten wir bei $T \to 0$ einen Grundzustand. Der Grundzustand des Systems (5.20) entspricht einer Route mit der kürzesten Gesamtstrecke.
Der Metropolis-Algorithmus funktioniert hier folgendermaßen: Ein Kandidat wird generiert, indem 2 Städte (oder mehrere) einer Route vertauscht werden. Dann wird $\Delta H$ berechnet und mit der Wahrscheinlichkeit $\exp(-\beta \Delta H)$ akzeptiert. Dieser Schritt ist nicht deterministisch. $\beta$ wird dabei 'langsam' von 0 nach $\infty$ ($T \to 0$) geführt. Ein Applets im Internet erlauben mit verschiedenen Positionen der Städte etc. zu experimentieren.

# 6 Computerphysik in der Praxis

Im Kurs 'Computerphysik' haben wir die Programme für jedes Problem selbst programmiert. In der Praxis ist es wenig sinnvoll, für jedes Standardproblem (z.B. Matrixinvertierung) eigene Routinen zu schreiben. Statt dessen kann man Codes aus Standardbibliotheken übernehmen.

- Das Standardwerk Numerical Recipes enthält 'Kochrezepte' zu einem breiten Spektrum von numerischen Problemen, die recht genau beschrieben sind. Pädagogisch ist das Buch sehr wertvoll, aufgrund des Alters ist der Code jedoch teils suboptimal. Hinzu kommen potentielle Lizenz-Probleme, denn die Codes sind geschützt.
- Die GNU Scientific Library (GSL) ist die Open Source Variante der Numerical Recipes. Die Codes sind moderner und effizienter. Jedoch enthält das Buch keinerlei mathematischen Hintergrund.
- Matlab (MAtrix LABoratory) ist keine Sammlung von C-Algorithmen, sondern eine Programmiersprache, die auf Matrixoperationen zugeschnitten ist. In ihr sind die meisten Algorithmen wie Zufallszahlen, Eigenwertbestimmung etc. schon vorhanden. Zudem handelt es sich um eine interpretierte Programmiersprache, d.h. man muss nicht kompilieren. Herausragend sind die graphischen Fähigkeiten.
- Ein besonders mächtiges Programm ist Mathematica. Es zeichnet sich dadurch aus, dass es symbolische Manipulationen vornehmen kann, also z.B. Integrale in geschlossener Form ausrechnen. Damit löst es sich von rein numerischen Berechnungen.

Für diese Programme gibt es Campus-Lizenzen. Hat man einen Linux-Rechner, so kann man sogar per ssh darauf zugreifen, ohne das Programm zu installieren: `http://www.uni-koeln.de/rrzk/software/mathematik/afs-mathsoftware.html`. Im Cip-Pool sind sie ebenso verfügbar.