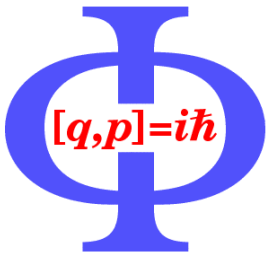
UNIVERSITÄT GÖTTINGEN



Institut für Theoretische Physik

Fakultät für Physik
Friedrich-Hund-Platz 1
37077 Göttingen

Skriptum zur Vorlesung

Computergestütztes Wissenschaftliches Rechnen

Andreas Honecker

Sommersemester 2007

(Stand: 20. Juli 2007)

Das Kleingedruckte: Dieses Skript erhebt nicht den Anspruch eines Lehrbuches. Insbesondere fehlt eine sorgfältige Durchsicht auf Fehler.

PDF-Fassung dieses Skripts (mit Hyperlinks etc.) unter

http://www.theorie.physik.uni-goettingen.de/~honecker/wr07

Inhaltsverzeichnis

0 Vorbemerkungen **1**

1 Klassische Mechanik **3**
1.1 Euler-Verfahren 4
1.2 Euler-Richardson-Verfahren 5
1.3 Runge-Kutta-Verfahren 8
1.3.1 Systeme $n$ter Ordnung 8
1.3.2 Runge-Kutta-Verfahren zweiter Ordnung 9
1.3.3 Runge-Kutta-Verfahren vierter Ordnung 10
1.3.4 Schrittweitenanpassung und Fehlerkontrolle 12
1.4 Kepler-Probleme und Sonnensystem 13
1.4.1 Einheiten im Sonnensystem 16

2 Diskrete Fourier-Transformation **17**
2.1 Binär-Zahlen 17
2.2 Schnelle Fourier-Transformation 19
2.3 Abtasteffekte 22

3 Partielle Differentialgleichungen: Statik **25**
3.1 Diskretisierung der Variablen 25
3.2 Potentialproblem 27
3.3 Lineare Gleichungssysteme 29
3.3.1 Conjugate Gradient Verfahren 32

4 Partielle Differentialgleichungen: Dynamik **37**
4.1 Explizite Lösungsverfahren 40
4.1.1 Explizites Euler-Verfahren 40
4.1.2 Lax-Verfahren 42
4.1.3 Diffusionsgleichung 44
4.2 Implizites Euler-Verfahren 2. Ordnung 47
4.2.1 Bi-Conjugate Gradient Verfahren 48
4.3 Zeitentwicklung in der Quantenmechanik 50
4.3.1 Differenzenverfahren 51
4.3.2 Operator-Splitting 54

4.3.3 Doppelmulden-Potential 57

5 Zufallsgeneratoren **61**
5.1 Pseudo-Zufallsgeneratoren 62
5.2 Andere Verteilungen 68
5.3 Monte-Carlo-Integration 70
5.3.1 Standardabweichung des Mittelwerts 72
5.3.2 Verbesserte Monte-Carlo-Integration 74

6 Importance Sampling: Metropolis-Algorithmus **77**
6.1 Ising-Modell 81

7 Quanten-Monte-Carlo **87**
7.1 Zufallsweg-Quanten-Monte-Carlo 88
7.2 Diffusions-Quanten-Monte-Carlo 92

8 Nachbemerkungen **97**

0 Vorbemerkungen

Unter der Überschrift „Computergestütztes wissenschaftliches Rechnen“ kann man verschiedene Dinge verstehen. Erstens mag man hier an Algorithmen (im Sinne der Informatik) denken. In dieser Vorlesung werden zwar auch Algorithmen diskutiert, jedoch nicht als Selbstzweck. Insbesondere werden wir klassische Themen der Computerwissenschaften wie z.B. Sortieren und Suchen oder die Nullstellensuche ausklammern. Selbstverständlich braucht auch ein Physiker ab und zu solche Algorithmen. Für diesen Fall sei als gute erste Anlaufstelle auf diverse Ausgaben des Buches „*Numerical Recipes*“ [24] verwiesen. Dieses Buch bietet eine gute Übersicht, allerdings ist zu beachten, dass es im Kern vor mehr als 15 Jahren entstanden ist und somit nicht immer auf dem neusten Stand ist. Insbesondere für rechenintensive Anwendungen ist daher immer eine Durchsicht neuerer Literatur zu empfehlen.

Zweitens fallen sicherlich auch Themen der Numerischen Mathematik in den Themenkomplex dieser Vorlesung. Allerdings wollen wir hier immer die Anwendung auf ein physikalisches Problem im Auge haben. Ferner werden wir im allgemeinen auf die mathematische Strenge verzichten, d.h. Beweise werden nicht immer vollständig angegeben und ggfs. wird eine Fehlerdiskussion auch nur angerissen. Für eine tiefergehende Diskussion dieser Aspekte der numerischen Mathematik sei auf die entsprechenden Vorlesungen der Mathematik verwiesen, wie auch die zahlreichen Bücher, die es zu diesem Thema gibt.

Schließlich kann man an eine Einführung in „Computational Physics“, einem vergleichsweise neuen Teilgebiet der Physik, denken. Dies wird auch die primäre Sichtweise dieser Vorlesung sein. Computational Physics wird manchmal neben Experiment und Theorie als ein drittes Standbein der Physik bezeichnet [10,11]. Zu betonen ist jedoch, dass Computational Physics als Bestandteil der Physik das Verständnis der Natur zum Ziel hat, wofür in diesem Fall eine numerische Methode verwendet wird. Eine grafische Darstellung ist oft unabdingbar, jedoch sollten bunte Bilder nicht mit Einsicht verwechselt werden. Auch kommt man ohne analytische Hilfsmittel nicht aus, um zum Beispiel die Probleme für eine Bearbeitung mit dem Computer geeignet auzubereiten, oder um die korrekte Funktion der Programme zu prüfen.

Da die genannten Themen und insbesondere Computational Physics inzwischen ein breites Gebiet darstellen, ist eine Themenauswahl unumgänglich. Eine

solche Auswahl spiegelt immer auch die perönlichen Präferenzen des Dozenten wieder. Auch wenn ein einigermaßen vollständiger Überblick über die wichtigsten Themen den Rahmen dieser Vorlesung sprengen würde, soll dennoch versucht werden, zumindest einen Eindruck von einigen der wichtigsten Verfahren und Vorgehensweisen zu vermitteln.

1 Klassische Mechanik

In der klassischen Mechanik wird der Zustand von N „Teilchen“ (Körpern) zur Zeit t beschrieben durch ihre Orte $\vec{r}_i(t)$, $i = 1, \ldots, N$ und die Geschwindigkeiten

$$\vec{v}_i(t) = \frac{\mathrm{d}}{\mathrm{d}t}\,\vec{r}_i(t) =: \dot{\vec{r}}_i(t)\,, \qquad i = 1, \ldots, N\,. \tag{1.1}$$

Diese beiden Größen sind äquivalent zu den aus der Theoretischen Mechanik bekannten Koordinaten und Impulsen.

Die Dynamik dieser Größen ist beschrieben durch die *Newtonschen Bewegungsgleichungen*

$$m_i\,\ddot{\vec{r}}_i = \vec{F}_i\left(\{\vec{r}_i(t)\}, \{\dot{\vec{r}}_i(t)\}, t\right)\,, \tag{1.2}$$

wobei m_i bzw. $\vec{F}_i$ die Masse bzw. die Summe aller auf das $i$te Teilchen wirkenden Kräfte sind. Gesucht sind die Bahnkurven $\vec{r}_i(t)$.

Es ist an dieser Stelle vielleicht nützlich sich zu erinnern, unter welchen Bedingungen die Lösung der Bewegungsgleichungen (1.2) geschlossen angegeben werden kann. Zunächst schränkt man sich gewöhnlich in der Theoretischen Mechanik auf konservative Kraftfelder ein, d.h. die Kräfte sind gegeben als Gradienten eines Potentials: $\vec{F}_i = \vec{\nabla}_i V\left(\{\vec{r}_i(t)\}, t\right)$. Aufgrund der Darstellung des Potentials V über Linienintegrale kann dieses nicht von den Geschwindigkeiten $\dot{\vec{r}}_i(t)$ abhängen, so dass geschwindigkeitsabhängige Kräfte ausgeschlossen werden. Insbesondere schließt dies Reibungskräfte aus, denn diese sind explizit geschwindigkeitsabhängig. Unter der Annahme konservativer Kraftfelder ist das Einteilchenproblem ($N = 1$) geschlossen lösbar, sowie translationsivariante Zweikörperprobleme ($N = 2$), da diese auf effektive Einteilchenprobleme zurückgeführt werden können. Bekanntlich ist aber bereits das Dreikörperproblem eine harte Nuß.

In den meisten Fällen können daher quantitative Ergebnisse nur mit Näherungsverfahren erzielt werden. So wurden z.B. bereits im 19. Jahrhundert Probleme der Himmelsmechanik mit Störungstheorie gelöst. Eine Alternative, mit der wir uns im folgenden beschäftigen wollen, sind numerische Verfahren. Auch diese sind nicht grundsätzlich neu (Anfänge sind z.B. bei Gauß und Euler zu finden), jedoch hat der Siegeszug der Computer ihre praktische Durchführbarkeit erheblich verbessert. Die folgenden Kapitel werden sich mit numerischen Lösungsverfahren für die Bewegungsgleichungen der klassischen Mechanik beschäftigen; für ergänzende Literatur sei dabei auf [10,11,29] verwiesen.

1.1 Euler-Verfahren

Das Euler-Verfahren ist eine (die wohl einfachste) Methode, Differentialgleichungen vom Typ (1.2) numerisch zu lösen. Zur Vereinfachung der Notation beschränken wir uns in der folgenden Diskussion auf den Fall eines Teilchens der Masse m.

Zunächst schreiben wir die Newtonsche Bewegungsgleichung (1.2) in der Form

$$\begin{aligned} \dot{\vec{r}}(t) &= \vec{v}(t)\,, \\ \dot{\vec{v}}(t) &= \frac{\vec{F}\,(\vec{r}(t),\vec{v}(t),t)}{m} =: \vec{a}\,(\vec{r}(t),\vec{v}(t),t)\ . \end{aligned} \tag{1.3}$$

Zweitens nähern wir die Zeitableitung durch den Differentialquotienten

$$\dot{\vec{r}}(t) \approx \frac{\vec{r}(t+\Delta t)-\vec{r}(t)}{\Delta t}\,, \qquad \dot{\vec{v}}(t) \approx \frac{\vec{v}(t+\Delta t)-\vec{v}(t)}{\Delta t}\,. \tag{1.4}$$

Setzt man für die linken Seiten hier die Bewegungsgleichungen (1.3) ein, so kann man nach $\vec{r}(t+\Delta t)$ und $\vec{v}(t+\Delta t)$ auflösen, d.h. für bekannte $\vec{r}(t)$ und $\vec{v}(t)$ (womit auch $\vec{a}\,(\vec{r}(t),\vec{v}(t),t)$ bekannt ist) liefern (1.3) und (1.4) eine Näherung dieser Größen zur Zeit $t+\Delta t$.

Diese Überlegungen erlauben die näherungsweise Berechnung einer diskreten Zeitentwicklung. Man betrachtet also diskrete Zeiten mit Abständen Δt

$$t_n = t_0 + n\,\Delta t\,, \qquad n = 0,1,2,\ldots \tag{1.5}$$

und definiert

$$\vec{r}_n = \vec{r}(t_n)\,, \qquad \vec{v}_n = \vec{v}(t_n)\,, \qquad \vec{a}_n = \vec{a}(\vec{r}_n,\vec{v}_n,t_n)\,.$$

Für diese Größen erhalten wir aus (1.3) und (1.4) die Iterationsvorschrift

$$\vec{v}_{n+1} = \vec{v}_n + \vec{a}_n\,\Delta t\,, \qquad \vec{r}_{n+1} = \vec{r}_n + \vec{v}_n\,\Delta t\,. \tag{1.6}$$

Diese Approximation an die Zeitentwicklung ist als *Euler-Verfahren* bekannt.

Im Euler-Verfahren sind die Werte $\vec{v}_{n+1}$, $\vec{r}_{n+1}$ am Ende des Zeitintervals $[t_n, t_{n+1}]$ ausschließlich durch die Werte an dessen Anfang $\vec{a}_n$, $\vec{v}_n$ und $\vec{r}_n$ bestimmt. Ferner geht die genäherte diskrete Zeitentwicklung im Limes $\Delta t \to 0$ formal in die exakte Zeitentwicklung über. Die Genauigkeit für endliches aber kleines Δt liest man aus der Taylor-Entwicklung ab

$$\vec{r}(t+\Delta t) = \vec{r}(t) + \Delta t\,\dot{\vec{r}}(t) + \mathcal{O}(\Delta t^2)\,. \tag{1.7}$$

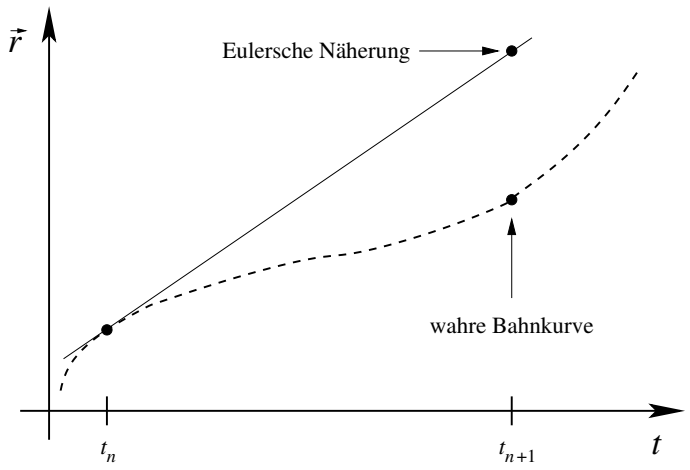


Abbildung 1.1: *Ein Zeitschritt Δt des Euler-Verfahrens.*

Folglich besteht ein Zeitschritt (1.6) im Euler-Verfahren in der linearen Näherung an die Bahnkurve zur Zeit t_n, wobei quadratische Termine in Δt sowie Terme höherer Ordnung vernachlässigt werden. Diese Eigenschaft eines Zeitschritts im Euler-Verfahren ist in Abb. 1.1 skizziert.

1.2 Euler-Richardson-Verfahren

Genau an der Stelle (1.7) kann man ansetzen, um das Verfahren zu verbessern, indem man Terme quadratischer (oder höherer Ordnung) in Δt in der diskretisierten Zeitentwicklung korrekt berücksichtigt. Die Euler-Richardson-Methode (eng verwandt mit dem Runge-Kutta-Verfahren, das im nächsten Kapitel diskutiert wird) ist eine solche Verbesserung.

Man führt zunächst einen mittleren Zeitpunkt $t_{\mathrm{mid}} = t + \Delta t/2$ ein, und bestimmt für diesen mit dem Euler-Verfahren eine Geschwindigkeit $\vec{v}_{\mathrm{mid}}$ und Position $\vec{r}_{\mathrm{mid}}$, sowie ferner $\vec{a}_{\mathrm{mid}} = \vec{F}\left(\vec{r}_{\mathrm{mid}}, \vec{v}_{\mathrm{mid}}, t_{\mathrm{mid}}\right)/m$. Mit den nun verfügbaren Daten $\vec{r}_n$, $\vec{v}_n$, $\vec{a}_n$ und $\vec{r}_{\mathrm{mid}}$, $\vec{v}_{\mathrm{mid}}$, $\vec{a}_{\mathrm{mid}}$ hat man nun ausreichend viele freie Parameter um $\vec{v}_{n+1}$ und $\vec{r}_{n+1}$ so zu konstruieren, dass diese genau sind bis einschließlich der Ordnung Δt^2. Diese Überlegungen führen auf das *Euler-Richardson-Verfahren*:

$$\begin{aligned}
\vec{a}_n &= \vec{F}\left(\vec{r}_n, \vec{v}_n, t_n\right)/m\,, \\
\vec{v}_{\mathrm{mid}} &= \vec{v}_n + \vec{a}_n \frac{\Delta t}{2}\,, \\
\vec{r}_{\mathrm{mid}} &= \vec{r}_n + \vec{v}_n \frac{\Delta t}{2}\,,
\end{aligned}$$

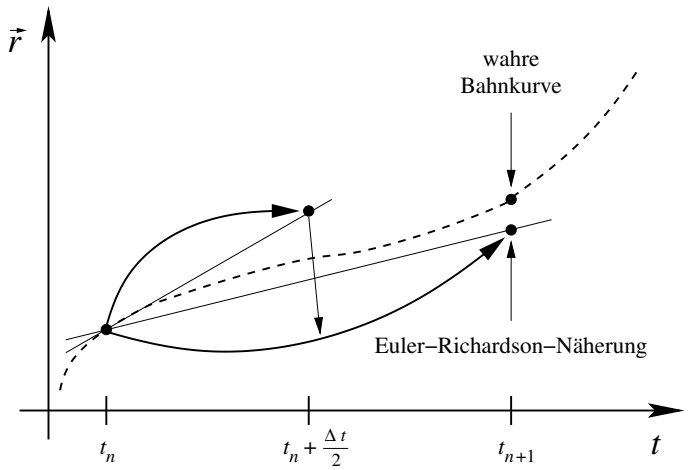


Abbildung 1.2: *Ein Zeitschritt Δt des Euler-Richardson-Verfahrens.*

$$\vec{a}_{\rm mid} \;=\; \vec{F}\left(\vec{r}_{\rm mid}, \vec{v}_{\rm mid}, t_n + \frac{\Delta t}{2}\right)/m \tag{1.8}$$

und

$$\begin{aligned} \vec{v}_{n+1} &= \vec{v}_n + \vec{a}_{\rm mid}\,\Delta t\,, \\ \vec{r}_{n+1} &= \vec{r}_n + \vec{v}_{\rm mid}\,\Delta t\,. \end{aligned} \tag{1.9}$$

Ein solcher Zeitschritt im Euler-Richardson-Verfahren ist in Abb. 1.2 skizziert.

Zwar müssen wir jetzt doppelt so viele Operationen pro Zeitschritt ausführen, meistens ist der Euler-Richardson-Algorithmus aber trotzdem deutlich schneller als der einfache Euler-Algorithmus, da wir größere Zeitschritte Δt verwenden können.

In der Tat sind die Abweichungen eines Zeitschritts des Euler-Richardson-Algorithmus nur von der Ordnung Δt^3. Um dies hier zumindest plausibel zu machen, verwenden wir eine Taylor-Entwicklung um $t_{\rm mid}$. Für die Geschwindigkeiten gilt

$$\begin{aligned} \vec{v}_n &= \vec{v}_{\rm mid} - \frac{\Delta t}{2}\,\dot{\vec{v}}_{\rm mid} + \frac{\Delta t^2}{8}\,\ddot{\vec{v}}_{\rm mid} + \mathcal{O}(\Delta t^3)\,, \\ \vec{v}_{n+1} &= \vec{v}_{\rm mid} + \frac{\Delta t}{2}\,\dot{\vec{v}}_{\rm mid} + \frac{\Delta t^2}{8}\,\ddot{\vec{v}}_{\rm mid} + \mathcal{O}(\Delta t^3)\,. \end{aligned} \tag{1.10}$$

Unter Beachtung von $\dot{\vec{v}}_{\rm mid} = \vec{a}_{\rm mid}$ findet man $\vec{v}_{n+1} - \vec{v}_n = \Delta t\,\vec{a}_{\rm mid} + \mathcal{O}(\Delta t^3)$. Mit analogen Argumenten folgt, dass $\vec{r}_{n+1} - \vec{r}_n = \Delta t\,\vec{v}_{\rm mid} + \mathcal{O}(\Delta t^3)$. Somit tritt

bei Verwendung von (1.9) tatsächlich kein quadratischer Korrekturterm in Δt auf, sondern die durch die Näherung verursachten Abweichungen sind von der Ordnung Δt^3.

Als ein Anwendungsbeispiel des Euler-(Richardson-)Algorithmus in der Physik wollen wir *Reibungskräfte* betrachten. Wie bereits erwähnt, sind Reibungskräfte geschwindigkeitsabhängig. Wir beschränken uns auf Flüssigkeiten und Gase. Dann gilt für nicht allzu große Geschwindigkeiten in guter Näherung

$$\vec{F}_R = -\lambda\, m\, \vec{v}\,, \tag{1.11}$$

wobei die Konstante λ von der Geometrie und Masse des Körpers abhängt und die Einheit einer inversen Zeit (s^{-1}) besitzt.

Für höhere Geschwindigkeiten stellt

$$\vec{F}_{R'} = -C\, m\, |\vec{v}|\, \vec{v} \tag{1.12}$$

eine bessere Beschreibung dar. Auch die Konstante C hängt von der Geometrie und Masse des Körpers ab und besitzt nun die Einheit einer inversen Länge (m^{-1}). Eine wichtige Eigenschaft der Reibungskraft (1.12) ist eine Kopplung der Bewegungsgleichung für die verschiedenen Koordinaten (über $|\vec{v}|$) – mit der Reibungskraft (1.11) entkoppeln die einzelnen Komponenten der Bewegungsgleichung.

Sowohl bei (1.11) als auch bei (1.12) handelt es sich um phänomenologische Formeln, die im Sinne einer Taylor-Entwicklung bzgl. $\vec{v}$ an komplexere physikalische Prozesse zu verstehen sind.

Wir betrachten ferner das Schwerefeld der Erde. Die Gravitationskraft kann in der Nähe der Erdoberfläche approximiert werden durch

$$\vec{F}_g = -m\, g\, \vec{e}_z\,. \tag{1.13}$$

Ohne Reibung, d.h. beim idealen freien Fall, sind die Bahnkurven Parabeln und die Geschwindigkeit $\vec{v}(t)$ wächst linear mit der Zeit. Die Reibung führt zu einer wichtigen Änderung im Vergleich zum idealen freien Fall. Nun können sich die Erdanziehung und Reibung aufheben! Dies führt zu einer *Grenzgeschwindigkeit* $\vec{v}_{\max}$.

Für den Fall (1.11) kann man die Grenzgeschwindigkeit leicht durch Nullsetzen der Summe von (1.11) und (1.13) herleiten. Man findet:

$$\vec{v}_{\max} = -\,\frac{g}{\lambda}\,\vec{e}_z\,. \tag{1.14}$$

Analog findet man für die zweite phänomenologische Reibungskraft (1.12) durch Nullsetzen der Summe von (1.12) und (1.13) ebenfalls eine maximale Geschwindigkeit $\vec{v}'_{\max}$. Diese ist gegen durch

$$\vec{v}'_{\max} = v'_{\max}\,\vec{e}_z\,, \qquad v'_{\max} = -\sqrt{\frac{g}{C}}\,. \tag{1.15}$$

1.3 Runge-Kutta-Verfahren

Bei den Newtonschen Bewegungsungleichungen (1.2) handelt es sich um Systeme gewöhnlicher Differentialgleichungen zweiter Ordnung. Im folgenden sollen die bisherigen Betrachtung verallgemeinert und ergänzt werden. So betrachten wir im folgenden den allgemeinen Fall gewöhnlicher Differentialgleichungen $n$ter Ordnung, werden eine neue Sichtweise der bisher dargestellten Verfahren kennenlernen und schließlich deren Ordnung weiter verbessern. Weiterführende Literatur zu diesem Thema findet man z.B. in Kapitel 16 des Buches [24] sowie Kapitel 4.1 des Buches [17].

1.3.1 Systeme $n$ter Ordnung

Gegeben sei ein System gewöhnlicher Differentialgleichungen $n$ter Ordnung

$$\frac{\mathrm{d}^n u_l}{\mathrm{d}t^n} = f_l\left(t, u_1, \ldots, u_m, \frac{\mathrm{d}\,u_1}{\mathrm{d}t}, \ldots, \frac{\mathrm{d}\,u_m}{\mathrm{d}t}, \ldots \frac{\mathrm{d}^{n-1}\,u_1}{\mathrm{d}t^{n-1}}, \ldots, \frac{\mathrm{d}^{n-1}\,u_m}{\mathrm{d}t^{n-1}}\right) \tag{1.16}$$

für die m Funktionen $u_l(t)$ $(l = 1, \ldots, m)$. Als konkretes Beispiel für (1.16) denken wir an die Newtonschen Bewegungsungleichungen (1.2), für die $n = 2$ und $m = 3\,N$ gilt.

Eine Differentialgleichungssystem vom Typ (1.16) läßt sich immer auf ein System gewöhnlicher Differentialgleichungen *erster* Ordnung zurückführen. Dazu führt man Variablen

$$z_{k,l}(t) = \frac{\mathrm{d}^k\,u_l(t)}{\mathrm{d}t^k} \tag{1.17}$$

mit $k = 0, \ldots, n-1$; $l = 1, \ldots, m$ ein. Mit diesen Variablen ist das System (1.16) äquivalent zu dem folgenden System 1. Ordnung:

$$\begin{aligned} \frac{\mathrm{d}\,z_{n-1,l}}{\mathrm{d}t} &= f_l\left(t, z_{0,1}, \ldots, z_{0,m}, z_{1,1}, \ldots, z_{1,m}, z_{n-1,1}, \ldots, z_{n-1,m}\right)\,, \\ \frac{\mathrm{d}\,z_{k,l}}{\mathrm{d}t} &= z_{k+1,l}\,, \qquad\qquad k = 0, \ldots, n-2\,. \end{aligned} \tag{1.18}$$

Die gesuchten Funktionen erhält man dann nach (1.17) als $u_l(t) = z_{0,l}(t)$. Aufgrund dieser Überlegung kann man sich bei der Konstruktion von numerischen Verfahren auf Differentialgleichungen 1. Ordnung beschränken:

$$\frac{\mathrm{d}\, x_r}{\mathrm{d}t} = f_r\left(t, x_1, \ldots, x_N\right) . \tag{1.19}$$

Genaugenommen haben wir dies sowohl beim Euler-Verfahren (1.6) wie auch beim Euler-Richardson-Algorithmus (1.8,1.9) bereits getan, indem wir die Variablen $\vec{v}$ (entsprechend $z_{1,l}$) und $\vec{r}$ (entsprechend $u_l = z_{0,l}$) verwendet haben.

1.3.2 Runge-Kutta-Verfahren zweiter Ordnung

Die einfachste Näherungslösung für eine Gleichung vom Typ (1.19) zu diskreten Zeiten $t_n = t_0 + n\,\Delta t$ ist durch das Euler-Verfahren gegeben

$$x_r(t_n) \quad \rightarrow \quad x_{r,n+1} = x_{r,n} + \Delta t\, f_r\left(t_n, x_{1,n}, \ldots, x_{N,n}\right) . \tag{1.20}$$

Dies ist offensichtlich eine Approximation, die genau ist bis zur Ordnung Δt. Nach Einsetzen der Variablen-Identifikationen erkennt man tatsächlich schnell die bereits früher angegebene Form (1.6) des Euler-Verfahrens.

Eigentlich gibt es keinen Grund, warum man bei der Berechnung von f_r auf der rechten Seiten von (1.20) ausgerechnet den Anfangspunkt des Intervalls $[t_n, t_{n+1}]$ verwendet. Tatsächlich kann man nicht nur andere Zeiten wählen, sondern durch Linearkombination die Genauigkeit des Verfahrens verbessern. Speziell können wir unter Verwendung von zwei Zeitpunkten den Ansatz machen, dass wir zwei Zeitschritte der Länge $\alpha\,\Delta t$ und $(1-\alpha)\,\Delta t$ nacheinander ausführen und das Ergebnis dann zu einem direkten Zeitschritt über Δt addieren. In diesem Sinn machen wir zunächst den Ansatz

$$\begin{aligned} k_r &= \Delta t\, f_r\left(t_n, x_{1,n}, \ldots, x_{N,n}\right) , \\ x_{r,n+1} &= x_{r,n} + \Delta t \Big(w_1\, f_r\left(t_n, x_{1,n}, \ldots, x_{N,n}\right) \\ &\qquad + w_2\, f_r\left(t_n + \alpha\,\Delta t, x_{1,n} + \alpha\, k_1, \ldots, x_{N,n} + \alpha\, k_N\right) \Big) \end{aligned} \tag{1.21}$$

mit freien Parametern α, w_1 und w_2. Dieser Ansatz soll nun die Taylor-Entwicklung für $x_r(t)$ bis zur zweiten Ordnung reproduzieren:

$$\begin{aligned} x_r(t+\Delta t) - x_r(t) &= \Delta t\, \frac{\mathrm{d}\, x_r}{\mathrm{d}t} + \frac{\Delta t^2}{2}\, \frac{\mathrm{d}^2\, x_r}{\mathrm{d}t^2} + \mathcal{O}(\Delta t^3) \\ &= \Delta t\, f_r + \frac{\Delta t^2}{2}\, \frac{\mathrm{d}\, f_r}{\mathrm{d}t} + \mathcal{O}(\Delta t^3) . \end{aligned} \tag{1.22}$$

Hierbei ist

$$\frac{\mathrm{d}\, f_r}{\mathrm{d}t} = \frac{\partial\, f_r}{\partial t} + \sum_s \frac{\partial\, f_r}{\partial x_s}\,\frac{\mathrm{d}\, x_s}{\mathrm{d}t} = \frac{\partial\, f_r}{\partial t} + \sum_s \frac{\partial\, f_r}{\partial x_s}\, f_s\,, \tag{1.23}$$

wobei wir im zweiten Schritt die Differentialgleichung (1.20) eingesetzt haben.

Andererseits lautet die Taylor-Entwicklung von (1.21)

$$\begin{aligned} x_{r,n+1} - x_{r,n} &= w_1\,\Delta t\, f_r + w_2\,\Delta t\, f_r \\ &\quad + w_2\,\Delta t^2\,\alpha\,\frac{\partial\, f_r}{\partial t} + w_2\,\Delta t^2\,\alpha \sum_s \frac{\partial\, f_r}{\partial x_s}\, f_s \\ &\quad + \mathcal{O}(\Delta t^3)\,, \end{aligned} \tag{1.24}$$

mit $f_r = f_r\,(t_n, x_{1,n}, \ldots, x_{N,n})$.

Durch Vergleich der Koeffizienten von (1.22) und (1.24) folgt

$$w_1 + w_2 = 1\,, \qquad w_2\,\alpha = \frac{1}{2}\,. \tag{1.25}$$

Diese Bedingungen besitzen eine einparametrige Schar von Lösungen, unter denen keine ausgezeichnet ist. Eine einfache Wahl ist

$$\alpha = \frac{1}{2}\,, \qquad w_2 = 1\,, \qquad w_1 = 0\,. \tag{1.26}$$

Wir erhalten damit das Runge-Kutta-Verfahren 2. Ordnung, das bis zur Ordnung Δt^2 genau ist:

$$\begin{aligned} k_r &= \Delta t\, f_r\,(t_n, x_{1,n}, \ldots, x_{N,n})\,, \\ x_{r,n+1} &= x_{r,n} + \Delta t\, f_r\left(t_n + \frac{\Delta t}{2}, x_{1,n} + \frac{k_1}{2}, \ldots, x_{N,n} + \frac{k_N}{2}\right)\,. \end{aligned} \tag{1.27}$$

Nach Transformation auf die Variablen $\vec{r}$ und $\vec{v}$ erkennt man den Euler-Richardson-Algorithmus (1.8,1.9) wieder.

1.3.3 Runge-Kutta-Verfahren vierter Ordnung

Unter Verwendung von mehr Zwischenschritten kann man ganz ähnlich wie in dem letzten Unterkapitel auch Runge-Kutta-Verfahren höherer Ordnung konstruieren. Ein häufig verwendetes Verfahren ist das Runge-Kutta-Verfahren vierter

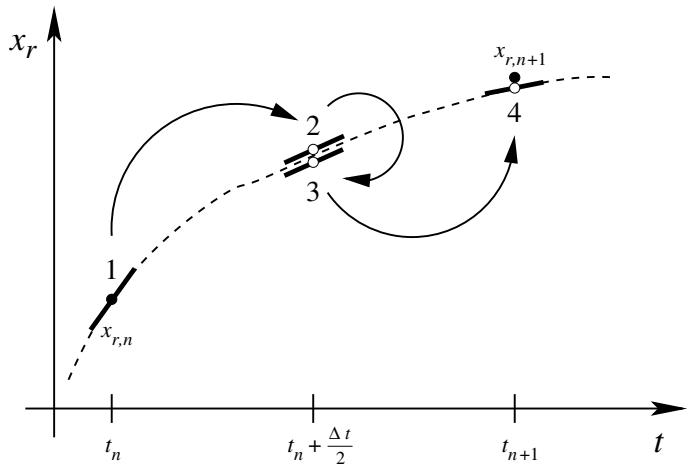


Abbildung 1.3: *Ein Zeitschritt Δt im Runge-Kutta-Verfahren 4. Ordnung.*

Ordnung, das bis zur Ordnung Δt^4 genau ist:

$$\begin{aligned}
k_{1,r} &= \Delta t\, f_r\left(t_n, x_{1,n}, \ldots, x_{N,n}\right) , \\
k_{2,r} &= \Delta t\, f_r\left(t_n + \frac{\Delta t}{2}, x_{1,n} + \frac{k_{1,1}}{2}, \ldots, x_{N,n} + \frac{k_{1,N}}{2}\right) , \\
k_{3,r} &= \Delta t\, f_r\left(t_n + \frac{\Delta t}{2}, x_{1,n} + \frac{k_{2,1}}{2}, \ldots, x_{N,n} + \frac{k_{2,N}}{2}\right) , \\
k_{4,r} &= \Delta t\, f_r\left(t_n + \Delta t, x_{1,n} + k_{3,1}, \ldots, x_{N,n} + k_{3,N}\right) , \\
x_{r,n+1} &= x_{r,n} + \frac{k_{1,r}}{6} + \frac{k_{2,r}}{3} + \frac{k_{3,r}}{3} + \frac{k_{4,r}}{6} .
\end{aligned} \tag{1.28}$$

In diesem Verfahren tritt ein Satz von vier Zwischengrößen $k_{i,r}$, $i = 1, \ldots, 4$ auf. Das Verfahren (1.28) soll hier nicht hergeleitet werden, stattdessen sei es in Abb. 1.3 grafisch veranschaulicht. Jeder der vier Zwischenschritte $k_{i,r}$ ergibt eine in der Ordnung Δt korrekte Näherung für $x_{r,n+1}$. Diese vier Näherungen werden in der letzten Zeile von (1.28) nun so linear kombiniert, dass sich die Korrekturterme der Ordnung Δt^2, Δt^3 und Δt^4 wegheben. Man beachte, dass die Summe der Koeffizienten eins ergeben muß, um die 1. Ordnung zu erhalten (in der Tat ist $1/6 + 1/3 + 1/3 + 1/6 = 1$).

Man beachte, dass Δt nicht beliebig groß gewählt werden kann, da die Potenzreihenentwicklung für große Δt gar nicht oder nur sehr langsam konvergiert. So sollte man bei Schwingungen oder Umläufen auch bei Verfahren hoher Ordnung mindestens größenordnungsmäßig 10 Zeitschritte je Periode machen. Daher

ist es zwar grundsätzlich möglich, Verfahren beliebig hoher Ordnung zu konstruieren; der zusätzliche Rechenaufwand wird aber irgendwann nicht mehr durch den Gewinn an Genauigkeit gerechtfertigt. Das Runge-Kutta-Verfahren vierter Ordnung ist aus diesem Blickwinkel ein guter Kompromiß und hat sich daher als Standard-Verfahren etabliert.

1.3.4 Schrittweitenanpassung und Fehlerkontrolle

Es gibt Probleme, wie z.B. der Flug einer Rakete zum Mond, in der sowohl Phasen mit schnellen Änderungen (Raketenstart, Einschwenken in Mondumlaufbahn, ...) solchen mit einer recht übersichtlichen Zeitentwicklung (z.B. Flugphase der Rakete von der Erde zum Mond) gegenüberstehen. In solchen Fällen ist es sinnvoll, nicht das gesamte Problem mit einer so kleinen Schrittweite Δt zu rechnen, dass zu allen Zeiten die gewünschte Genauigkeit erreicht wird, sondern eine Anpassung der Schrittweite im Verlauf des Verfahrens vorzunehmen.

Am elegantesten ist eine adaptive Steuerung der Schrittweite, die gleichzeitig auch eine Abschätzung des Fehlers erlaubt. Eine Möglichkeit den Fehler zu kontrollieren erhält man, wenn man jeden Schritt doppelt ausführt, und zwar einmal direkt und einmal als zwei halbe Schritte[1]:

$$\begin{aligned} x_{r,n} \quad &\rightarrow \quad x_{r,n+1} = X_1 \,, \\ x_{r,n} \quad &\rightarrow \quad x_r\left(t_n + \frac{\Delta t}{2}\right) \quad \rightarrow \quad x_{r,n+1} = X_2 \,. \end{aligned} \tag{1.29}$$

Für ein Verfahren $m$ter Ordnung gilt nun

$$\begin{aligned} x_r\left(t_n + \Delta t\right) &= X_1 + c\,\Delta t^{m+1} + \mathcal{O}\left(\Delta t^{m+2}\right) \\ &= X_2 + 2\,c\left(\frac{\Delta t}{2}\right)^{m+1} + \mathcal{O}\left(\Delta t^{m+2}\right) \,. \end{aligned} \tag{1.30}$$

Wir haben also

$$X_2 - X_1 \approx c\,\Delta t^{m+1}\left(1 - 2^{-m}\right) \,. \tag{1.31}$$

Wir können nun eine Schrittweite $\tilde{\Delta} t$ so bestimmen, dass der Fehler mit dieser Schrittweite in einem Schritt nach (1.29) einen vorgegebenen Wert ϵ annimmt

$$\left|\tilde{X}_2 - \tilde{X}_1\right| = \epsilon \,. \tag{1.32}$$

[1]Der einfache und doppelte Schritt teilen zumindest den Anfangspunkt. Durch Wiederverwenden solchen Überlapps kann der zusätzliche Rechenaufwand auf weniger als 50% reduziert werden.

Aus (1.31) folgt

$$\begin{aligned} \epsilon &= |c|\ \tilde{\Delta}t^{m+1}\ \left(1-2^{-m}\right)\ , \\ |X_2 - X_1| &= |c|\ \Delta t^{m+1}\ \left(1-2^{-m}\right)\ . \end{aligned} \tag{1.33}$$

Dies kann nach $\tilde{\Delta}t$ aufgelöst werden:

$$\tilde{\Delta}t = \Delta t\ \sqrt[m+1]{\frac{\epsilon}{|X_2 - X_1|}}\,. \tag{1.34}$$

War der Fehler größer als gewünscht, muß der letzte Schritt mit der Schrittweite (1.34) *wiederholt* werden. Ist der Fehler geringer als gefordert, kann im *nächsten* Schritt die größere Schrittweite (1.34) verwendet werden[2].

Da man c kennt, kann man außerdem eine sogenannte lokale Extrapolation ausführen, die den Fehlerterm der Ordnung $m+1$ korrigiert. Dazu kombiniert man (1.30) so, dass der Term $m+1$ter Ordnung eliminiert wird und verwendet als Schätzwert

$$x_{r,n+1} = X_2 + \frac{X_2 - X_1}{2^m - 1}\,. \tag{1.35}$$

Ob man dies tut oder läßt, ist Geschmackssache. In jedem Fall darf man nicht vergessen, dass man auch bei dieser Korrektur der $m+1$ten Ordnung nur den Fehler der $m$ten Ordnung kontrolliert.

1.4 Kepler-Probleme und Sonnensystem

Wir wollen in den Übungen astronomische Probleme (insbesondere das Sonnensystem) simulieren. Ergebnisse, die Sie hierzu benötigen, werden in diesem Kapitel zusammengefaßt.

Wir betrachten zunächst ein Zentralproblem in dem Gravitationspotential

$$V(r) = -\frac{G\,M\,m}{r}\,, \tag{1.36}$$

wobei M die Masse des Zentralkörpers ist. Aus dem Potential (1.36) ergibt sich eine Kraft

$$\vec{F}(\vec{r}) = -\frac{G\,M\,m}{r^3}\,\vec{r} \tag{1.37}$$

mit $r = |\vec{r}| = \sqrt{\vec{r}\cdot\vec{r}}$.

[2]Es empfiehlt sich, die Schrittweite Δt grundsätzlich etwas kleiner zu wählen, als die „genaue“ Formel nahelegt.

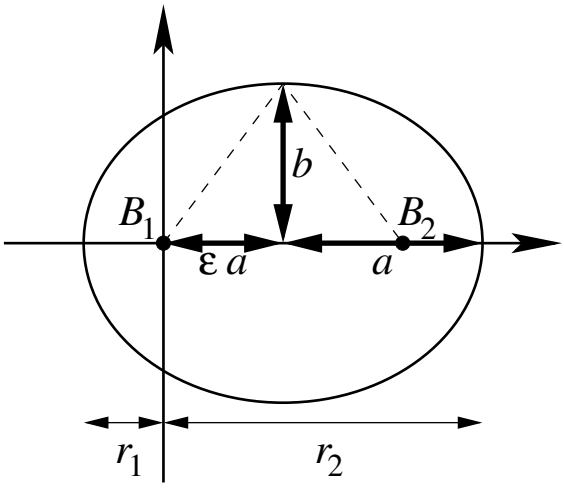


Abbildung 1.4: *Eine Ellipse. Die Bezeichnungen sind im Text erläutert.*

Für Energie $E < 0$ erhalten wir gebundene Bahnen. In dem Potential (1.36) handelt es sich hierbei um geschlossene Kurven, und zwar um Ellipsen. Wir fassen nun kurz einige bekannte Eigenschaften elliptischer Bahnen zusammen, die bei der Simulation von Kepler-Problemen nützlich sind. Dazu betrachten wir die in Abb. 1.4 dargestellte Ellipse. B_1 und B_2 sind die beiden Brennpunkte, a die große Halbachse und b die kleine Halbachse. $\epsilon \geq 0$ ist die sogenannte Exzentrizität. Für $\epsilon = 0$ findet man $a = b$ und die beiden Brennpunkte fallen zusammen, so dass sich ein Kreis ergibt. Eine mögliche Charakterisierung einer Ellipse ist, dass die Summe der Abstände von den beiden Brennpunkten B_1 und B_2 zu einem beliebigen Punkt auf ihr konstant ist. Mit den Bezeichnungen in Abb. 1.4 kann man daraus folgern, dass

$$b = a\sqrt{1-\epsilon^2}\,. \tag{1.38}$$

Aus dieser Formel folgert man weiter, dass $\epsilon < 1$ für eine eine Ellipse.

Das Kraftzentrum (und damit auch der Koordinatenursprung $r = 0$) befindet sich in dem Brennpunkt, den wir B_1 genannt haben. Die Orte mit den minimalen und maximalen Bahnradien (r_1 bzw. r_2) werden Apsiden genannt. Aus obiger Skizze liest man ab:

$$r_1 = (1-\epsilon)\,a \qquad r_2 = (1+\epsilon)\,a\,. \tag{1.39}$$

Dies ist übrigens auch konsistent mit der Definition der großen Halbachse $a = (r_1 + r_2)/2$.

Aus der Diskussion des Zentralpotentials (1.36) in der Theoretischen Mechanik folgt (siehe z.B. S. 57 von [27])

$$a = \frac{r_1 + r_2}{2} = -\frac{G\,M\,m}{2\,E}\,, \tag{1.40}$$

mit der Gesamtenergie

$$E = \frac{1}{2}\,m\,v^2 + V(r)\,. \tag{1.41}$$

Diese Definition von E kann man nun nach $v(r)$ auflösen. Man findet mit Hilfe von (1.36) und (1.40)

$$v(r) = \sqrt{G\,M\,\left(\frac{2}{r} - \frac{1}{a}\right)}\,. \tag{1.42}$$

Für eine Kreisbahn ist $r = a$. In diesem Fall reduziert sich (1.42) auf

$$v = \sqrt{\frac{G\,M}{r}}\,. \tag{1.43}$$

Letzteres Ergebnis folgt auch elementar durch Betrachtung des Gleichgewichts von „Fliehkraft“ und Gravitationskraft (1.37).

Da man außer dem Betrag der Geschwindigkeit auch ihre Richtung kennen muß, empfiehlt es sich, (1.42) auf Werte von r anzuwenden bei denen $\mathrm{d}\,r/\mathrm{d}t$ verschwindet, was $\vec{v} \perp \vec{r}$ bedeutet. Dies ist für eine Kreisbahn immer erfüllt, so dass für gegebenes r die zu einer Kreisbahn führende Geschwindigkeit aus (1.43) bestimmt werden kann. Im Fall einer Ellipse gilt $\vec{v} \perp \vec{r}$ insbesondere für die beiden Apsiden r_1 und r_2. Für eine gegebene elliptische Bahn kann man die Geschwindigkeit $\vec{v}$ daher mit Hilfe von (1.42) an den Apsiden besonders einfach bestimmen.

Das Potential (1.36) gilt auch für die relative Bewegung zweier Körper der Massen m und M. In diesem Fall ist $\vec{r}$ als Differenz der Ortsvektoren der beiden Körper zu interpretieren. Dies läßt sich leicht auf die gravitative Wechselwirkung von N Körpern der Massen m_i verallgemeinern:

$$V(\vec{r}_1, \ldots, \vec{r}_N) = -G \sum_{i<j} \frac{m_i\,m_j}{|\vec{r}_i - \vec{r}_j|}\,. \tag{1.44}$$

Hieraus folgen die Bewegungsgleichungen

$$\ddot{\vec{r}}_i = -G \sum_{\substack{j=1 \\ j\neq i}}^{N} m_j\,\frac{\vec{r}_i - \vec{r}_j}{|\vec{r}_i - \vec{r}_j|^3}\,. \tag{1.45}$$

1.4.1 Einheiten im Sonnensystem

Für eine numerische Lösung empfiehlt es sich grundsätzlich, an das Problem angepaßte Einheiten zu verwenden. Bei einem Zentralkraftproblem sollten diese so gewählt werden, dass der Zahlenwert von $G\,M$ in der Nähe von 1 liegt. Zur Beschreibung des Sonnensystems verwendet man als Längeneinheit die astronomische Einheit (AU), die gleich der großen Halbachse der Erdumlaufbahn (d.h. im wesentlichen deren mittleren Radius) gesetzt wird:

$$1\,AU = 1,496\,10^{11}\mathrm{m}\,. \tag{1.46}$$

Als Zeiteinheit verwendet man das Jahr, $1\,yr \approx 3,15\,10^{7}$s, d.h. die Umlaufdauer der Erde, die in guter Näherung auf einer Kreisbahn umläuft. Nun gilt für die Umlaufdauer $T = \frac{2\pi\,r}{v}$ auf einer Kreisbahn mit Radius r nach (1.43)

$$T^2 = \frac{4\pi^2}{G\,M}\,r^3\,. \tag{1.47}$$

Diese Beziehung ist die Aussage des 3. Keplerschen Gesetzes für den speziellen Fall einer Kreisbahn. Die Beziehung (1.47) kann man leicht nach dem Produkt von Gravitationskonstante und Sonnenmasse auflösen, und erhält in astronomischen Einheiten ($r = 1\,AU$, $T = 1\,yr$) den folgenden Wert:

$$G\,M = 4\pi^2\,\frac{AU^3}{yr^2}\,. \tag{1.48}$$

2 Diskrete Fourier-Transformation

An vielen Stellen benötigt man eine Fourier-Analyse. Dies ist z.B. bei der Bestimmung der in einer periodischen Bewegungen auftretenden Frequenzen der Fall (man denke z.B. an die Bahnkurven $\vec{r}_i(t)$ der Körper unseres Sonnensystems).

Hier erinnern wir kurz an die diskrete Fourier-Transformation (für eine etwas ausführlichere Dikussion vgl. z.B. Kapitel 12.1 von [24] oder Kapitel 9 von [11]). Gegeben seien Funktionswerte f_r an N äquidistanten Punkten $r = 0, 1, \ldots, N-1$. Dann definieren wir die komplexe diskrete Fourier-Transformation über

$$\hat{f}_s = \frac{1}{\sqrt{N}} \sum_{r=0}^{N-1} \mathrm{e}^{-2\pi i r s/N} f_r \,, \tag{2.1}$$

wobei s ebenfalls die ganzzahligen Werte von 0 bis $N-1$ annimmt. Die inverse Transformation ist gegeben durch

$$f_r = \frac{1}{\sqrt{N}} \sum_{s=0}^{N-1} \mathrm{e}^{2\pi i r s/N} \hat{f}_s \,. \tag{2.2}$$

Für die Vorfaktoren gibt es unterschiedliche Konventionen in der Literatur. In (2.1) und (2.2) wurden sie so gewählt, dass man unitäre Abbildungen erhält.

Nach (2.1) und (2.2) sind für die Berechnung der Transformation jeweils N Elemente zu berechnen, die ihrerseits aus N Summanden bestehen. In dieser Darstellung sind somit $\mathcal{O}(N^2)$ Operationen durchzuführen. Hierbei handelt es sich um einen erheblichen Aufwand: eine naive Verwendung von (2.1) oder (2.2) zur Berechnung des Frequenz-Spektrums einer nicht besonders langen Zeitreihe mit $N = 10^6$ Datenpunkten benötigt einen erheblichen Aufwand mit einer Größenordnung von 10^{12} Operationen! Für so etwas muß man selbst bei einem modernen Computer etwas Geduld aufwenden.

Häufig ist es jedoch möglich, die Zahl der benutzten Datenpunkte N zu variieren. Wir werden etwas später zeigen, wie man die Summen (2.1) und (2.2) insbesondere im Spezialfall, dass N eine 2er-Potenz ist, deutlich schneller berechnen kann. Dazu ist es jedoch nützlich, sich an Binär-Zahlen zu erinnern.

2.1 Binär-Zahlen

Der Computer arbeitet mit sognenannten „Bits“, d.h. Einheiten die die Zahlenwerte 0 und 1 annehmen. Dies führt auf natürliche Weise zur Binärdarstellung,

a_m	b_m	und a_m & b_m	oder a_m | b_m	exklusiv oder a_m ^ b_m
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

Tabelle 2.1: *Ergebnisse c_m einer „und“ ($c_m = a_m$ & b_m), „oder“ ($c_m = a_m$ | b_m) bzw. „exklusiv oder“ ($c_m = a_m$ ^ b_m) Verknüpfung zweier Bits a_m und b_m.*

d.h. einer Darstellung zur Basis 2. Für eine ganze Zahl B lautet die Binärdarstellung

$$B = \sum_{m=0}^{N} b_m \, 2^m \,, \tag{2.3}$$

wobei b_m die Bits der Zahl sind. Analog zur Dezimaldarstellung schreibt man die Binärdarstellung (2.3) als

$$b_N \, b_{N-1} \ldots b_1 \, b_0 \,, \tag{2.4}$$

wobei führende Nullen oft mit dargestellt werden. Die Binärdarstellung der Dezimalzahl 13 lautet z.B. 1101. 8 Bits werden zu einem „Byte“ zusammengefaßt. Mit einem Byte sind die ganzen Zahlen $0 \leq B \leq 2^8 - 1 = 255$ darstellbar.

Programmiersprachen unterstützen neben den Grundrechenarten auch binäre Operationen auf ganzen Zahlen. Dies sind zunächst die bitweise „und“- ($C = A$ & B), „oder“- ($C = A$ | B) bzw. „exklusiv oder“-Verknüpfung ($C = A$ ^ B) zweier Zahlen A und B. Hierbei haben wir die Symbole &, | bzw. ^ verwendet, die z.B. in C, C++ und Java für diese Operationen zum Einsatz kommen. Das Ergebnis C wird hierbei berechnet, indem jedes Bit c_m über die entsprechende Verknüpfung aus den zugehörigen Bits a_m und b_m von A und B berechnet wird. Die Ergebnisse für die Verknüpfungen von zwei Bits a_m und b_m sind in Tabelle 2.1 zusammengefaßt.

Eine weitere wichtige Klasse von Operationen auf der Binär-Darstellung sind die Bit-Shifts. Verwenden wir wieder die Symbole, die insbesondere in C, C++ und Java ür diese Operationen verwendet werden, wird ein Shift um n Bits nach links geschrieben als

$$B = A \mathbin{<<} n \tag{2.5}$$

sowie ein Shift um n Bits nach rechts als

$$B = A >> n\,. \tag{2.6}$$

Für die Bits bedeutet die Links-Verschiebung (2.5) die Zuordnung $b_{m+n} = a_m$ $(m \geq 0)$ und $b_k = 0$ $(k < n)$, wobei die Bits von A, für die im Ergebnis B kein Platz ist, wegfallen. Effektiv bewirkt die Verschiebung um n Bits nach links eine Multiplikation mit einer Potenz von 2, nämlich $B = 2^n\,A$ modulo der Kapazität des Datentyps.

Analog bedeutet die Rechts-Verschiebung (2.6) die Zuordnung $b_m = a_{m+n}$, wobei im Datentyp nicht vorhandene links Bits a_{m+n} auf 0 gesetzt werden. Die Bits a_k mit $k < n$ gehen hierbei verloren. Effektiv bewirkt die Verschiebung um n Bits nach rechts eine ganzzahlige Division ohne Rest durch eine Potenz von 2, nämlich $B = A/2^n$ ohne Rest.

2.2 Schnelle Fourier-Transformation

Im Spezialfall, dass $N = 2^m$ eine 2er-Potenz ist, bietet die ,,Schnelle Fourier-Transformation" (*Fast Fourier Transformation* (FFT)) eine Alternative zur Berechnung der Summen (2.1) und (2.2), die mit $\mathcal{O}(N \log_2 N)$ Operationen auskommt, also deutlich schneller ist (siehe z.B. Kapitel 12.2 von [24] und Anhang 9B von [11]). Die hier vorgestellte Formulierung ist als Danielson-Lanczos-Algorithmus bekannt.

Wir führen zuerst ein wenig Notation ein, und zwar die fundamentale Nte Einheitswurzel

$$W_N = \mathrm{e}^{2\,\pi\,i/N} \tag{2.7}$$

sowie eine etwas anders normierte Form der (transformierten) Daten

$$F_s = \frac{1}{\sqrt{N}}\,\hat{f}_s\,. \tag{2.8}$$

Dann kann man (2.2) zuerst mit diesen Größen ausdrücken und anschließend die Summe in die geraden bzw. ungeraden Terme zerlegen:

$$\begin{aligned} f_r = \sum_{s=0}^{N-1} W_N^{r\,s}\,F_s &= \sum_{s=0}^{N/2-1} W_N^{r\,(2\,s)}\,F_{2\,s} + \sum_{s=0}^{N/2-1} W_N^{r\,(2\,s+1)}\,F_{2\,s+1} \\ &= \sum_{s=0}^{N/2-1} W_{N/2}^{r\,s}\,F_{2\,s} + W_N^r \sum_{s=0}^{N/2-1} W_{N/2}^{r\,s}\,F_{2\,s+1} \\ &= F_r^e + W_N^r\,F_r^o\,. \end{aligned} \tag{2.9}$$

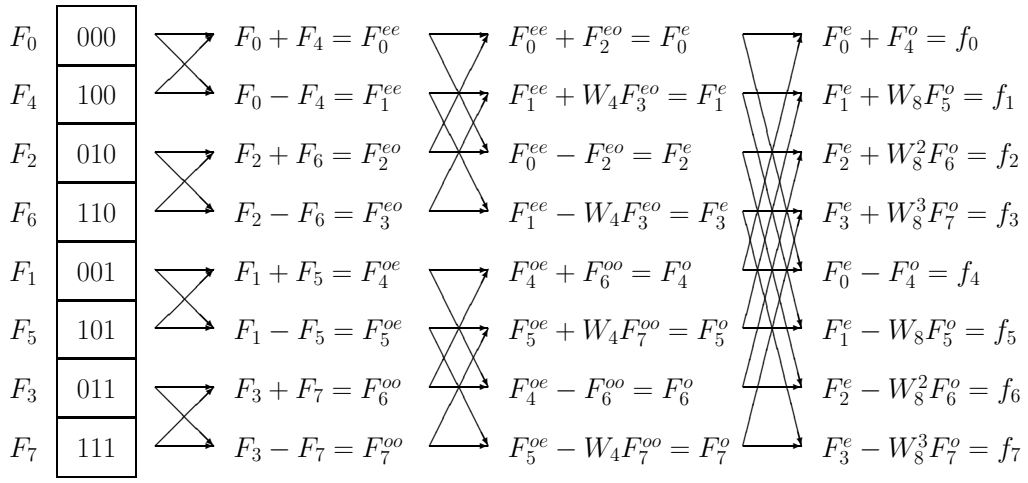


Abbildung 2.1: *Struktur der schnellen Fourier-Transformation (FFT) für $N = 8$ Datenpunkte.*

Neben der Zerlegung wurde $W_N^2 = W_{N/2}$ verwendet. In der vorletzten Zeile von (2.9) erkennt man dann die Fourier-Transformation der geraden bzw. ungeraden Hälften der Daten: $F_r^e = \sum_{s=0}^{N/2-1} W_{N/2}^{r\,s}\, F_{2\,s}$ bzw. $F_r^o = \sum_{s=0}^{N/2-1} W_{N/2}^{r\,s}\, F_{2\,s+1}$. Man beachte, dass in (2.9) $0 \le r \le N$ zu wählen ist, die Teil-Transformationen F_r^e und F_r^o jedoch Periode $N/2$ besitzen.

Die Zerlegung (2.9) kann nun rekursiv verwendet werden, und zwar solange bis die Unterfolgen nur noch aus einem Element bestehen. Für $N = 4 = 2^2$ findet man z.B.

$$\begin{aligned} f_r &= F_r^e + W_4^r\, F_r^o \\ &= F_r^{ee} + W_2^r\, F_r^{eo} + W_4^r\, (F_r^{oe} + W_2^r\, F_r^{oo}) \\ &= F_0 + (-1)^r\, F_2 + i^r\, (F_1 + (-1)^r\, F_3)\ . \end{aligned} \tag{2.10}$$

Hierbei wurde $W_2 = \mathrm{e}^{2\,\pi\,i/2} = -1$ und $W_4 = \mathrm{e}^{2\,\pi\,i/4} = i$ eingesetzt, sowie $F_r^{ee} = F_0$, $F_r^{eo} = F_2$, $F_r^{oe} = F_1$ und $F_r^{oo} = F_3$ identifiziert.

Im allgemeinen gelangt man nach $m = \log_2 N$ Schritten zu einem einzelnen Element F_t der Ursprungsdaten, wobei noch das richtige t zu identifizieren ist. Wir schreiben $F_t = F_r^{a_1\,a_2 \ldots a_m}$ mit $a_j = 0$ für e („even" – die gerade Unterfolge) bzw. $a_j = 1$ für o („odd" – die ungerade Unterfolge). Faßt man die Folge $(a_1\, a_2 \ldots a_m)$ nun als Binärdarstellung auf, so gilt $t = (a_m\, a_{m-1} \ldots a_1)$. t erhält man somit

einfach durch Bitumkehr! Dies ist insofern einsichtig, da man zuerst nach gerade bzw. ungerade fragt (a_1 ist also das niedrigste Bit von t) usw. bis schließlich nach oberer bzw. unterer Hälfte klassifiziert wird (d.h. a_m ist das höchste Bit von t).

Der gesamte Sachverhalt ist nochmals für $N = 8$, d.h. $m = \log_2 8 = 3$ in Abb. 2.1 dargestellt. Nach der Herleitung zerteilt man von rechts beginnend die Berechnung der Summanden in f_r unter Verwendung von (2.9) iterativ in jeweils zwei Unterhälften. Schließlich gelangt man zu den Ursprungsdaten F_s, die durch Bit-Umkehr identifiziert werden.

Es gibt verschiedenen Möglichkeiten, den beschriebenen Algorithmus zu implementieren, z.B. mit Hilfe von Rekursion, wobei allerdings Kopien der Daten angelegt werden müssen. Hier wollen wir eine Implementierung vorstellen, die Speicherplätze so oft wie möglich wiederverwendet. Dazu gehen wir bei der Berechnung genau umgekehrt wie oben geschildert vor: Wir führen zuerst die Bit-Umkehr aus, und fassen dann jeweils paarweise Summanden zusammen. Damit ergibt sich der folgende FFT Algoritmus:

1. Vertausche $F_r \leftrightarrow F_t$ für t Bit-Umkehr von r. In diesem Schritt kann auch der Normierungsfaktor in $F_r = \hat{f}_r/\sqrt{N}$ implementiert werden.

2. Schleife $m = 1$, $n = 2^m \leq N$, $m \to m + 1$

   Berechne W_n nach (2.7)[3].

   Schleife $k = 0$, $k < n/2$, $k \to k + 1$

   Schleife $l = k$, $l < N$, $l \to l + n$

   Berechne gleichzeitig[4]

   $$F_l + W_n^k \, F_{l+n/2} \to F_l$$
   $$F_l - W_n^k \, F_{l+n/2} \to F_{l+n/2}$$

   W_n^k sollte durch iterative Multiplikation berechnet werden:
   $W_n^0 = 1$, $W_n^{k+1} = W_n \, W_n^k$.

[3] An dieser Stelle läßt sich auch leicht das unterschiedliche Vorzeichen aus (2.1) bzw. (2.2) berücksichtigen, so dass die gleiche Routine sowohl zur Berechnung der Fourier-Transformation als auch ihrer Inversen verwendet werden kann.

[4] Man beachte, dass $W_n^{l+n/2} = -W_n^l$ ist, sowie $W_n^l = W_n^k$, da $l - k$ ein Vielfaches von n ist.

Beachtet man die korrekte Reihenfolge der (gleichzeitigen) Ersetzungen, kann das gleiche Array für die Eingangsdaten f_r (oder $\hat{f}_s$), die Zwischenergebnisse F_l sowie die Ausgangsdaten $\hat{f}_s$ (oder f_r) verwendet werden.

Die FFT kann zwar in Realteil und Imaginärteil aufgeteilt und dann reell implementiert werden. Es liegt jedoch nahe, komplexe Arithmetik zu verwenden.

Wir fassen zusammen: Durch geschickte Wiederverwendung von Zwischenergebnissen in der Berechnung reduziert die FFT den Aufwand der Berechnung der N Summen (2.1) oder (2.2) von $\mathcal{O}(N^2)$ auf $\mathcal{O}(N \log_2 N)$ Operationen, vorausgesetzt N ist eine Zweierpotenz. In vielen Anwendungen ist es kein Problem, die Zahl der Datenpunkte bzw. Stützstellen als Zweierpotenz zu wählen.

Man beachte, dass bei der Anwendung der FFT auf ein physikalisches Problem die Stützstellen, die in diesem Kapitel mit Abstand eins angenommen wurden, noch umzurechnen sind.

2.3 Abtasteffekte

In vielen Fällen liegen ursprünglich kontinuierliche Daten vor, wie z.B. ein Audiosignal oder der Bahnradius $r_i(t)$. Wir wollen daher kurz auf einige Effekte hinweisen, die bei der Diskretisierung solcher Daten auftreten.

Zunächst betrachten wir reelle Daten $f_r \in \mathbb{R}$. Aufgrund von $(\mathrm{e}^{2\pi i x})^\star = \mathrm{e}^{-2\pi i x}$ und $\mathrm{e}^{2\pi i} = 1$ gilt für die Nte Einheitswurzel (2.7): $(W_N^t)^\star = W_N^{-t} = W_N^{N-t}$ und damit für die Fourier-Transformierte (2.1)

$$\hat{f}_s^\star = \hat{f}_{-s} = \hat{f}_{N-s}\,. \tag{2.11}$$

Dies hat folgende Konsequenzen für die Abtastung eines kontinuierlichen Signals $f(t)$:

$$f_r = f(t_0 + r\,\Delta t)\,. \tag{2.12}$$

Bei einer solchen Abtastung sind nur Frequenzen bis zur *Nyquist-Frequenz*

$$f_C = \frac{1}{2\,\Delta t} \tag{2.13}$$

auflösbar. Die Nyquist-Frequenz (2.13) entspricht in unserer bisherigen Sprechweise $s = N/2$, der nach (2.11) höchstfrequenten Fourier-Komponente. Tatsächlich gilt nach dem Abtast-Theorem (siehe z.B. Kapitel 12.1 von [24]), dass $f(t)$

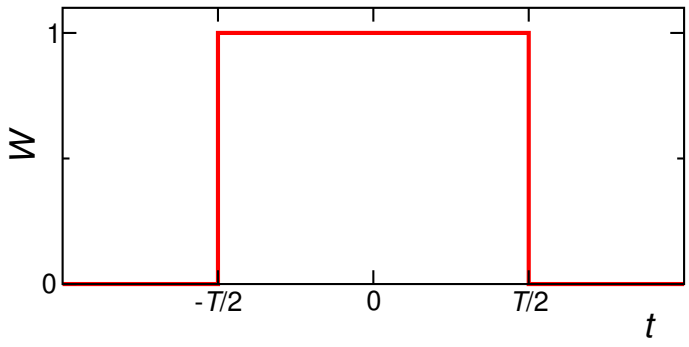


Abbildung 2.2: *Die Rechteck-Fenster-Funktion (2.14).*

eindeutig aus den abgetasteten Datenpunkten f_r rekonstruiert werden kann, vorausgesetzt in dem Signal $f(t)$ sind keine Frequenzen oberhalb der Nyquist-Frequenz f_C enthalten.

In dem Fall, dass höhere Frequenzen in dem Signal $f(t)$ vorliegen, tritt jedoch leider sogenanntes *Aliasing* auf. Dies bedeutet, dass höhere Frequenzen zu tiefen Frequenzen gefaltet werden. In der Tat gilt $\mathrm{e}^{2\pi i f_1 t_r} = \mathrm{e}^{2\pi i f_2 t_r}$ für alle Abtastpunkte $t_r = t_0 + r\,\Delta t$ genau dann, wenn sich die beiden Frequenzen f_1 und f_2 um ein ganzzahligen Vielfaches der Abtastfrequenz unterscheiden, d.h. genau dann wenn $f_1 - f_2 = \frac{n}{\Delta t}$ mit $n \in \mathbb{Z}$ (man beachte auch (2.11). Dies bedeutet, dass Δt so klein gewählt werden sollte, dass Beiträge mit Frequenzen jenseits der Nyquist-Frequenz im ursprünglichen Signal $f(t)$ vernachlässigt werden können.

Ein weiterer Effekt, den man beachten sollte, ist das sogenannte *Windowing* (vgl. z.B. Kapitel 13.4 von [24]). Man betrachtet nämlich meistens nicht das volle Signal $f(t)$, sondern schränkt es auf ein Zeitfenster der Länge T ein. Zunächst betrachten wir ein Rechteck-Fenster (vgl. Abb. 2.2)

$$W(t) = \Theta(t + T/2)\,\Theta(T/2 - t)\,, \tag{2.14}$$

wobei die Heaviside-Funktion

$$\Theta(t) = \begin{cases} 1 & \text{für } t > 0\,, \\ 0 & \text{für } t < 0 \end{cases} \tag{2.15}$$

auftritt. Die Fourier-Transformierte des Rechteck-Fensters (2.14) ist im kontinuierlichen Fall leicht berechenbar:

$$\hat{W}(s) = \int\limits_{-\infty}^{\infty} \mathrm{d}t\, \mathrm{e}^{-2\pi i s t}\, W(t) = \int\limits_{-T/2}^{T/2} \mathrm{d}t\, \mathrm{e}^{-2\pi i s t} = \frac{\sin(\pi\, s\, T)}{\pi\, s}\,. \tag{2.16}$$

Dies bedeutet eine Mischung benachbarter Frequenzen, die nur langsam ($\propto 1/s$) mit der Frequenz-Differenz s abfällt. Im diskreten Fall ändert sich das Ergebnis (2.16) etwas, die Schlußfolgerung bleibt jedoch qualitativ unverändert. Der Grund für dieses Verhalten ist das scharfe An- bzw. Ausschalten des Signals zur Zeit $t = -T/2$ bzw. $t = T/2$ (scharfe Kanten führen bekanntlich zu Überschwingern). Dieses Verhalten läßt sich somit verbessern, wenn man spezielle Fenster-Funktionen $W(t)$ verwendet, die einen glatten Anstieg und Abfall haben (siehe z.B. Kapitel 13.4 von [24]). Zumindest sollte man in Erinnerung behalten, dass die Frequenzauflösung durch die Länge des Zeitfensters begrenzt wird und eben nicht durch den Abstand der Frequenz-Punkte.

3 Partielle Differentialgleichungen: Statik

In diesem Kapitel werden wir „statische“ Lösungen partieller Differentialgleichungen zu vorgegebenen Randbedingungen diskutieren. In der Mathematik wird diese Fragestellung als Randwertproblem bezeichnet. Die Fragestellung wollen wir mit einem Beispiel aus der Elektrostatik illustrieren: Die Poisson-Gleichung lautet in CGS-Einheiten

$$-\Delta\,\Phi(\vec{x}) = -\sum_{i=1}^{d} \frac{\partial^2}{\partial x_i^2}\,\Phi(\vec{x}) = 4\,\pi\,\rho(\vec{x})\,. \tag{3.1}$$

Hierbei ist Δ der Laplace-Operator und d die räumliche Dimension. Naheliegende Fälle sind $d = 3$ (insbesondere in der Elektrostatik) sowie $d = 2$. Ferner sind in der Elektrostatik die Ladungsdichte $\rho(\vec{x})$ sowie Randbedingungen für Φ gegeben. Die Aufgabe besteht nun darin, ein Lösung der partiellen Differentialgleichung (3.1) zu finden und so das Potential $\Phi(\vec{x})$ zu bestimmen.

Natürlich gibt es ausgefeilte analytische Verfahren zur Lösung des Randwertproblems. Wir wollen uns hier jedoch auf numerische Verfahren konzentrieren. Partielle Differentialgleichungen müssen für eine Behandlung mit dem Rechner geeignet aufbereitet werden. Eine Möglichkeit ist die Entwicklung nach geeigneten Funktionensystemen. Bekannt ist z.B. die Fourier-Transformation, d.h. die Entwicklung nach ebenen Wellen, die u.a. den Laplace-Operator diagonalisiert (beachte: $\Delta\,\mathrm{e}^{i\,\vec{k}\cdot\vec{x}} = -\vec{k}^2\,\mathrm{e}^{i\,\vec{k}\cdot\vec{x}}$). Auf endlichen Gebieten ist es jedoch bei einem solchen Zugang nicht immer einfach, vorgegebene Randbedingungen zu berücksichtigen. Insbesondere für nichtlineare partielle Differentialgleichungen sind bei der Entwicklung räumliche Integrale zu berechnen, die insbesondere Matrixelemente zwischen verschiedenen Basisfunktionen bestimmen. Im letzteren Fall führt die Entwicklung auf ein Gleichungssystem, das es zu lösen gilt.

Wir wollen im folgenden einen anderen elementaren Weg beschreiten, der sich auch durch breite Anwendbarkeit auszeichnet.

3.1 Diskretisierung der Variablen

Eine Vorgehensweise zur Darstellung partieller Differentialgleichungen im Rechner basiert auf der Betrachtung der Funktion $f(\vec{x})$ an Stützstellen $\vec{x}_i$. Dieser Zugang ist auch analog zur Behandlung gewöhnlicher Differentialgleichungen in Kapitel 1. Die genannten Stützstellen können recht allgemein gewählt werden,

wodurch insbesondere bei adaptiver Wahl auch eine gute Anpassung an das Problem möglich ist. Allerdings ist eine allgemeine Stützstellen-Verteilung schwer zu handhaben.

Im folgenden soll ein reguläres Gitter von Punkten zur Diskretisierung einer partiellen Differentialgleichung für die Funktion $f(\vec{x})$ verwendet werden. Als Gitter wählen wir ein quadratisches bzw. (hyper-)kubisches Gitter

$$\vec{x}_{\vec{r}} = \vec{x}_0 + \sum_{i=1}^{d} \Delta x_i \; r_i \; \vec{e}_i \tag{3.2}$$

mit ganzzahligen r_i. Für die Funktionswerte an diesen Punkten schreiben wir

$$f_{\vec{r}} = f\left(\vec{x}_{\vec{r}}\right) \; . \tag{3.3}$$

Wir betrachten nun die partiellen Ableitungen $\frac{\partial^m}{\partial x_i^m} f(\vec{x})$ und tun der Einfachheit halber so, als ob dies die einzige Variable wäre. Für die erste partielle Ableitung an der Stelle $\vec{x}_r$ liegen zwei Definitionen nahe:

$$\begin{aligned} \frac{\partial}{\partial x_i} f(\vec{x}_r) &= \frac{f_r - f_{r-1}}{\Delta x_i} + \mathcal{O}(\Delta x_i) \\ &= \frac{f_{r+1} - f_r}{\Delta x_i} + \mathcal{O}(\Delta x_i) \, . \end{aligned} \tag{3.4}$$

Durch eine geschickte symmetrische Kombination der beiden Möglichkeiten (3.4) kann man die Ordnung verbessern:

$$\frac{\partial}{\partial x_i} f(\vec{x}_r) = \frac{f_{r+1} - f_{r-1}}{2\,\Delta x_i} + \mathcal{O}(\Delta x_i^2) \, . \tag{3.5}$$

Analog kann man die zweite partielle Ableitung konstruieren. Eine geschickte Wahl ist

$$\frac{\partial^2}{\partial x_i^2} f(\vec{x}_r) = \frac{f_{r+1} - 2\,f_r + f_{r-1}}{\Delta x_i^2} + \mathcal{O}(\Delta x_i^2) \, . \tag{3.6}$$

Wir müssen nun noch unsere Randbedingungen unterbringen. Bei *Dirichletschen Randbedingungen* (für eine partielle Differentialgleichung zweiter Ordnung) ist die Funktion f auf dem Rand des zu untersuchenden Gebiets bekannt. Dem läßt sich leicht Rechnung tragen, indem man für die Ableitungen direkt neben dem Rand in (3.5) oder (3.6) einfach für die entsprechenden $f_{r\pm1}$ die vorgegebenen Werte einsetzt. Bei *Neumannschen Randbedingungen* ist hingegen die Ableitung der Funktion f in Normalenrichtung des Randes auf diesem bekannt. Auch dies

läßt sich unterbringen, z.B. indem man $\frac{f_r - f_{r-1}}{\Delta x_i}$ in (3.6) an den entsprechenden Randpunkten durch die vorgegebene Funktion ersetzt. Man beachte jedoch, dass man hierbei nicht die optimale Näherung für die erste Ableitung verwendet (siehe (3.4)) und somit am Rand Genauigkeit verliert.

Krummlinige Koordinatensysteme lassen sich durch Variablentransformationen mit der geschilderten Vorgehensweise bearbeiten. So würde man z.B. in Kugelkoordinaten r, θ, φ diese Variablen jeweils äquidistant diskretisieren.

3.2 Potentialproblem

Der Spezialfall $\rho = 0$ der Poisson-Gleichung (3.1) führt auf die Laplace-Gleichung

$$\Delta \, \Phi = 0 \, . \tag{3.7}$$

Wie bereits erwähnt, ist Φ in der Elektrodynamik als elektrostatisches Potential zu interpretieren. Die Laplace-Gleichung tritt jedoch auch in anderen Zusammenhängen auf.

In $d = 2$ besteht z.B. eine Verbindung zwischen der Gleichung (3.7) und Minimalflächen auf einem endlichen Gebiet, wobei Φ nun als dritte Koordinate interpretiert werden kann. In diesem Fall korrespondiert $\frac{\partial^2}{\partial x_i^2} \Phi$ zu dem Krümmungsradius der Fläche an einem Punkt in Richtung i und die Laplace-Gleichung bedeutet, dass für eine Minimalfläche die Summe über die Krümmungsradien verschwindet. Wir wollen diesen Zusammenhang kurz etwas präzisieren. Dazu sei $\Phi(x, y)$ die z-Koordinaten der über dem Punkt (x, y) aufgespannten Fläche. Der infinitesimale Flächeninhalt an diesem Punkt ist

$$\mathrm{d}x \, \mathrm{d}y \, a\left(\frac{\partial \Phi}{\partial x}, \frac{\partial \Phi}{\partial y}\right) \quad \text{mit} \quad a\left(\frac{\partial \Phi}{\partial x}, \frac{\partial \Phi}{\partial y}\right) = \sqrt{\left(1 + \left(\frac{\partial \Phi}{\partial x}\right)^2\right)\left(1 + \left(\frac{\partial \Phi}{\partial y}\right)^2\right)} \, . \tag{3.8}$$

Damit ist die Oberfläche einer innerhalb eines vorgegebenen Randes aufgespannten Fläche $\Phi(x, y)$ gegeben durch

$$A[\Phi] = \int \mathrm{d}^2 x \, a\left(\frac{\partial \Phi}{\partial x}, \frac{\partial \Phi}{\partial y}\right) \, . \tag{3.9}$$

Eine Minimalfläche ist definiert als die Fläche mit der minimalen Oberfläche innerhalb des vorgegebenen Randes. Die Minimalfläche ist ein Extremum des

Funktionals $A[\Phi]$ und erfüllt somit das Variationsprinzip

$$\frac{\delta A}{\delta \Phi} = 0\,. \tag{3.10}$$

Bekanntlich führt die Variationsbedingung auf die Euler-Langrage-Gleichung

$$\frac{\partial}{\partial x}\,\frac{\delta a}{\delta \frac{\partial \Phi}{\partial x}} + \frac{\partial}{\partial y}\,\frac{\delta a}{\delta \frac{\partial \Phi}{\partial y}} - \frac{\delta a}{\delta \Phi} = 0\,. \tag{3.11}$$

Setzt man die explizite Form von a aus (3.8) in (3.11) ein, so ist $\frac{\delta a}{\delta \Phi} = 0$ und man findet

$$\begin{aligned}
0 &= \frac{\partial}{\partial x}\,\frac{1}{a}\,\frac{\partial \Phi}{\partial x}\left(1+\left(\frac{\partial \Phi}{\partial y}\right)^2\right) + \frac{\partial}{\partial y}\,\frac{1}{a}\,\frac{\partial \Phi}{\partial y}\left(1+\left(\frac{\partial \Phi}{\partial x}\right)^2\right) \\
&= \frac{1}{a}\left(\frac{\partial^2 \Phi}{\partial x^2}+\frac{\partial^2 \Phi}{\partial y^2}\right) \\
&\quad + \frac{1}{a}\left(\frac{\partial^2 \Phi}{\partial x^2}\left(\frac{\partial \Phi}{\partial y}\right)^2 + \frac{\partial^2 \Phi}{\partial y^2}\left(\frac{\partial \Phi}{\partial x}\right)^2\right) \\
&\quad + \frac{\partial \Phi}{\partial x}\,\frac{\partial}{\partial x}\,\frac{1}{a}\left(1+\left(\frac{\partial \Phi}{\partial y}\right)^2\right) + \frac{\partial \Phi}{\partial y}\,\frac{\partial}{\partial y}\,\frac{1}{a}\left(1+\left(\frac{\partial \Phi}{\partial x}\right)^2\right)\,.
\end{aligned} \tag{3.12}$$

Für den Fall dass $\frac{\partial \Phi}{\partial x}$ und $\frac{\partial \Phi}{\partial y}$ klein sind, kann man die Beiträge auf der zweiten und dritten Zeile im Ergebnis vernachlässigen; der erste Term führt dann auf die Laplace-Gleichung (3.7). Tatsächlich kann man für einen gegebenen Punkt (x, y) immer lokale Koordianten so wählen, dass $\frac{\partial \Phi}{\partial x} = \frac{\partial \Phi}{\partial y} = 0$ gilt. Für eine globale Sichtweise sind dann allerdings die Transformationen der Koordinatensysteme zu berücksichtigen und man erhält einen modifizierten Laplace-Operator. Diese Sichtweise führt in Richtung Differentialgeometrie und soll hier nicht weiter verfolgt werden. Wir halten lediglich fest, dass ein Zusammenhang zwischen der Laplace-Gleichung (3.7) und Minimalflächen besteht.

Kehren wir nun zur numerischen Behandlung der Laplace-Gleichung zurück. Die Diskretisierung (3.6) führt auf einem d-dimensionalen hyperkubischen Gitter mit Gitterabstand $\Delta x_i = \Delta x$ zu folgender Diskretisierung des Laplace-Operators

$$\begin{aligned}
\Delta\,\Phi\left(\vec{x}_{\vec{r}}\right) &= \frac{1}{\Delta x^2}\left(-2d\,\Phi\left(\vec{x}_{\vec{r}}\right) + \sum_{i=1}^{d}\left[\Phi\left(\vec{x}_{\vec{r}}+\Delta x\,\vec{e}_i\right)+\Phi\left(\vec{x}_{\vec{r}}-\Delta x\,\vec{e}_i\right)\right]\right) \\
&\quad + \mathcal{O}(\Delta x^2)\,.
\end{aligned} \tag{3.13}$$

Damit erhält man folgende Gitter-Variante der Laplace-Gleichung (3.7)

$$\Phi\left(\vec{x}_{\vec{r}}\right) = \frac{1}{2\,d}\sum_{i=1}^{d}\left\{\Phi\left(\vec{x}_{\vec{r}} + \Delta x\,\vec{e}_i\right) + \Phi\left(\vec{x}_{\vec{r}} - \Delta x\,\vec{e}_i\right)\right\} + \mathcal{O}(\Delta x^4)\,. \tag{3.14}$$

Dies bedeutet, dass das Potential am Platz $\vec{x}_{\vec{r}}$ gleich dem Mittelwert über seine Werte an den $2\,d$ Nachbarplätzen ist!

Diese Beobachtung legt ein einfaches Lösungsverfahren für die Laplace-Gleichung (3.7) nahe, das unter dem Namen Jacobi-Verfahren bekannt ist [10,11,29]:

1. Wähle eine beliebige Start-Potential-Konfiguration $\Phi\left(\vec{x}_{\vec{r}}\right)$, die allerdings die geforderten Randbedingungen erfüllt.

2. Berechne

$$\Phi_{\text{neu}}\left(\vec{x}_{\vec{r}}\right) = \frac{1}{2\,d}\sum_{i=1}^{d}\left\{\Phi\left(\vec{x}_{\vec{r}} + \Delta x\,\vec{e}_i\right) + \Phi\left(\vec{x}_{\vec{r}} - \Delta x\,\vec{e}_i\right)\right\}\,, \tag{3.15}$$

   bzw. bestimme $\Phi_{\text{neu}}\left(\vec{x}_{\vec{r}}\right)$ für Randplätze $\vec{x}_{\vec{r}}$ aus den Randbedingungen.

3. Berechne

$$\delta\Phi = \max_{\vec{x}_{\vec{r}}}\left|\Phi_{\text{neu}}\left(\vec{x}_{\vec{r}}\right) - \Phi\left(\vec{x}_{\vec{r}}\right)\right|\,. \tag{3.16}$$

4. Ersetze $\Phi\left(\vec{x}_{\vec{r}}\right)$ durch $\Phi_{\text{neu}}\left(\vec{x}_{\vec{r}}\right)$.

5. Fahre bei 2. fort, wenn $\delta\Phi$ eine vorgegebene Schwelle überschreitet. Andernfalls höre auf.

3.3 Lineare Gleichungssysteme

Wir kehren nun zurück zur Poisson-Gleichung (3.1). Mit der Diskretisierung (3.6) nimmt diese die Form

$$\sum_{i=1}^{d}\frac{\Phi\left(\vec{x}_{\vec{r}} + \Delta x_i\,\vec{e}_i\right) + \Phi\left(\vec{x}_{\vec{r}} - \Delta x_i\,\vec{e}_i\right) - 2\,\Phi\left(\vec{x}_{\vec{r}}\right)}{\Delta x_i^2} = -4\,\pi\,\rho\left(\vec{x}_{\vec{r}}\right) \tag{3.17}$$

an. Bringt man alle bekannte Information in (3.17) auf die rechte Seite und zählt die Stützstellen $\vec{x}_{\vec{r}}$ geeignet durch, so kann dies in die Form einer Gleichung

$$A\,\vec{\Phi} = \vec{b} \tag{3.18}$$

gebracht werden. Dies ist ein Spezialfall der allgemeineren Aussage, dass die Diskretisierung einer linearen partiellen Differentialgleichung auf ein lineares Gleichungssystem für die Werte von $\Phi(\vec{x}_{\vec{r}})$ an den inneren Punkten $\vec{x}_{\vec{r}}$ führt.

Zur Verdeutlichung betrachten wir den Fall von $d = 2$ räumlichen Dimensionen und ein quadratisches $N \times N$ Gitter. Die Stützstellen indizieren wir wie folgt:

$$\vec{x}_{\vec{r}} = \vec{x}_0 + r_1 \Delta x_1 \vec{e}_1 + r_2 \Delta x_2 \vec{e}_2 = \vec{x}_n \qquad \text{mit} \qquad n = N r_2 + r_1 \tag{3.19}$$

für $0 \leq r_i < N$. Der Einfachheit halber beschränken wir uns ferner auf Null-Randbedingungen ($\Phi = 0$ auf dem Rand). Die Vektoren in (3.18) nehmen dann folgende Form an:

$$\vec{\Phi} = \begin{pmatrix} \Phi(\vec{x}_0) \\ \Phi(\vec{x}_1) \\ \vdots \\ \Phi(\vec{x}_{N^2-1}) \end{pmatrix}, \qquad \vec{b} = -4\pi \begin{pmatrix} \rho(\vec{x}_0) \\ \rho(\vec{x}_1) \\ \vdots \\ \rho(\vec{x}_{N^2-1}) \end{pmatrix}. \tag{3.20}$$

Für allgemeine Randbedinungen würden die auf dem Rand vorgebenen Werte ebenfalls zu $\vec{b}$ beitragen.

Schließlich lesen wir die Matrix A aus (3.17) ab. Im Fall $\Delta x_i = \Delta x$ erhalten wir folgende $N^2 \times N^2$ Matrix:

$$A = \frac{1}{\Delta x^2} \begin{pmatrix} -4 & 1 & & & 1 & & & & \\ 1 & \ddots & \ddots & & 0 & \ddots & & & \\ & \ddots & \ddots & 1 & \vdots & \ddots & \ddots & & \\ & & 1 & -4 & 0 & \dots & 0 & 1 & \\ 1 & 0 & \dots & 0 & -4 & 1 & & & \\ & \ddots & & \vdots & 1 & \ddots & \ddots & & \ddots \\ & & \ddots & 0 & & \ddots & \ddots & 1 & \\ & & & 1 & & & 1 & -4 & \\ & & & & & \ddots & & & \ddots \end{pmatrix}. \tag{3.21}$$

Die meisten hier nicht explizit angegebenen Einträge der Matrix A sind 0. Praktisch empfiehlt es sich, die Matrix A nicht explizit zu programmieren, sondern das Gitter geeignet so zu durchlaufen, dass die Nachbarplätze zu einem gegebenen Punkt leicht bestimmt werden können, und dann anhand einer Zuordnung vom Typ (3.19) zu identifizieren, an welcher Stelle nicht-verschwindende Matrixelemente einzutragen sind.

Wir haben somit die Lösung einer diskretisierten linearen partiellen Differentialgleichung auf die Lösung eines linearen Gleichungssystems (3.18) zurückgespielt. Zur Lösung linearer Gleichungen existieren verschiedene Verfahren. So wurde z.B. in der Einführung in die Rechnerbedienung bereits das Gauß-Verfahren erwähnt [15] (Vorlesung vom 08.03.2007 und Aufgabe 2 von Übungsblatt 4 – siehe z.B. auch Kapitel 2.1 von [24] oder Kapitel 5 von [29]). Bei der Gauß-Elimination sind drei geschachtelte Schleifen zu durchlaufen, so dass man $\mathcal{O}(\dim^3)$ Operationen benötigt, wenn dim die Dimension des Vektorraums ist. Ferner muß bei diesem Verfahren die komplette Matrix A gespeichert werden, was zu einem Speicherbedarf $\propto \dim^2$ führt. Konkret betrachten wir wieder das Beispiel $d = 2$ und eine Genauigkeit, d.h. eine moderate Anzahl von Stützstellen $N = 100$. In diesem Beispiel ist $\dim = N^2 = 10^4$. Man benötigt somit $\mathcal{O}(10^{12})$ Operationen und fast 800MByte Speicher um die Matrixelemente als `double`-Zahlen (zu 8 Byte) zu speichern. Dies mag zwar auf einem modernen PC noch durchführbar sein; $N = 200$ sprengt allerdings definitiv die Möglichkeiten eines PCs. Hinzu kommt, dass direkte Lösungsverfahren wie das Gauß-Verfahren Rundungsfehler akkumulieren, so dass eine hinreichende Genauigkeit der Lösung insbesondere in den genannten hohen Dimensionen nicht sichergestellt ist.

Die genannten Probleme treten bereits bei der Diskretisierung der Laplace-Gleichung (3.7) auf, und zwar auch für Fälle, die sich mit dem Jacobi-Verfahren aus Kapitel 3.2 durchaus lösen lassen. Das einfache Jacobi-Verfahren ist in diesem Fall also deutlich effizienter als die Gauß-Elimination! Die Problematik ergibt sich daraus, dass wir spezielle Eigenschaften der Diskretisierung A von $-\Delta$ nicht nuten, wenn wir das Gleichungssystem mit dem Gauß-Verfahren lösen.

Tatsächlich hat A in (3.18) für viele partielle Differentialgleichungen zusätzliche nützliche Eigenschaften, insbesondere

1. Für viele Differentialgleichungen ist A symmetrisch (bzw. hermitesch)
$$A = A^\dagger \,. \tag{3.22}$$

2. A ist dünn besetzt, d.h. die meisten Matrixelemente a_{rs} von A verschwinden. Im Fall von (3.17) gilt in jeder Zeile r, dass $a_{rs} \neq 0$ nur für höchstens $2\,d+1$ Werte von s, und zwar unabhängig von der Größe des gewählten Gitters.

3. A ist positiv definit, d.h.
$$\vec{x} \cdot A\vec{x} > 0 \qquad \text{für} \quad \vec{x} \neq \vec{0} \,. \tag{3.23}$$

Diese Eigenschaft geht für die Diskretisierung A von $-\Delta$ darauf zurück, dass dieser Differentialoperator positiv definit ist (man sieht schnell, dass $-\int \mathrm{d}^d x\, f(\vec{x})^\star\, \Delta\, f(\vec{x}) = \int \mathrm{d}^d x\, \left|\vec{\nabla}\, f(\vec{x})\right|^2 > 0$ für geeignete Funktionen $f(\vec{x})$, wenn man Randterme ignoriert).

3.3.1 Conjugate Gradient Verfahren

Ziel dieses Kapitels ist es, eine Näherungslösung des linearen Gleichungssystems

$$A\,\vec{x} = \vec{b} \tag{3.24}$$

mit einer gewünschten Genauigkeit zu erhalten, unter der Voraussetzung, dass A die soeben genannten Eigenschaften 1-3 besitzt. Dieses Ziel wird mit einem iterativen Verfahren errreicht, das 1952 von Hestenes und Stiefel vorgeschlagen wurde [14] und unter dem Namen Conjugate Gradient Verfahren (Konjugierte Gradienten-Verfahren) bekannt ist (siehe z.B. Kapitel 2.7 von [24], Kapitel 11.9 von [16] oder Kapitel 10.2 von [8]).

Das Conjugate Gradient Verfahren basiert auf der Idee, die Funktion

$$f(\vec{x}) = \frac{1}{2}\,\vec{x}\cdot A\,\vec{x} - \vec{b}\cdot\vec{x} \tag{3.25}$$

zu minimieren. In einem Extremal- oder Sattelpunkt gilt für symmetrische Matrizen A

$$\vec{0} = \vec{\nabla}\, f(\vec{x}) = A\,\vec{x} - \vec{b}\,, \tag{3.26}$$

d.h. ein Extremal- oder Sattelpunkt $\vec{x}$ der Funktion (3.25) ist automatisch eine Lösung des linearen Gleichungssystems (3.24).

Beim Conjugate Gradient Verfahren nimmt man nun an, dass die Matrix A symmetrisch und positiv definit ist (beides gilt für Diskretisierungen des Operators $-\Delta$). Sei $\vec{x}_0$ die Lösung von (3.26), d.h. $A\,\vec{x}_0 - \vec{b} = \vec{0}$. Dann kann man unter Ausnutzung der Symmetrie von A zeigen, dass

$$f(\vec{x}_0 + \vec{x}') = \frac{1}{2}\,\vec{x}'\cdot A\,\vec{x}' + c \tag{3.27}$$

mit

$$c = \frac{1}{2}\,\vec{x}_0\cdot A\,\vec{x}_0 - \vec{b}\cdot\vec{x}_0\,. \tag{3.28}$$

Da A positiv definit ist, gilt $\vec{x}'\cdot A\,\vec{x}' > 0$ für $\vec{x}' \neq \vec{0}$. Folglich ist $\vec{x}' = \vec{0}$ das eindeutige Minimum der Funktion (3.27).

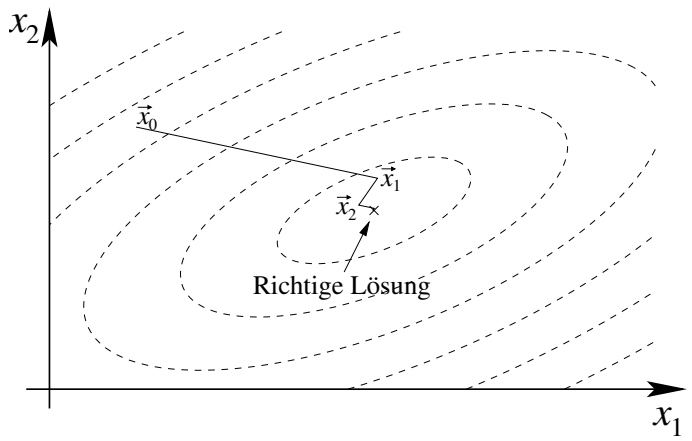


Abbildung 3.1: *Illustration des steilsten Abstiegs.*

Wir haben somit gezeigt, dass für symmetrische positiv definite Matrizen A die Lösung des Gleichungssystems (3.24) äquivalent ist zur Bestimmung des Minimums der Funktion (3.25).

Dieses Minimum kann nun iterativ bestimmt werden. Dies soll zuerst mit einem sehr einfachen Verfahren illustriert werden, der sogenannten Methode des *steilsten Abstiegs*. Hier läuft man jeweils die Funktion f entlang ihrer maximalen Steigung nach unten. Die Richtung der maximalen Steigung an einem Punkt $\vec{x}_k$ ist durch den Gradienten

$$\vec{\nabla} f(\vec{x}_k) = A\,\vec{x}_k - \vec{b} =: -\vec{r}_k \tag{3.29}$$

gegeben. Man bestimmt nun einen neuen Punkt

$$\vec{x}_{k+1} = \vec{x}_k + \alpha_k\,\vec{r}_k \tag{3.30}$$

so, dass $f(\vec{x}_{k+1})$ bzgl. α_k minimal ist. Aus der Bedingung $\partial f/\partial\alpha_k = 0$ leitet man leicht her, dass

$$\alpha_k = \frac{\vec{r}_k \cdot \vec{r}_k}{\vec{r}_k \cdot A\vec{r}_k} \tag{3.31}$$

zu wählen ist. Man beachte, dass aufeinanderfolgende Schritte voneinander unabhängig sind. Rundungsfehler, die unweigerlich immer auftreten, sammeln sich also nicht an, sondern werden in nachfolgenden Schritten wieder eliminiert.

Man beachte ferner, dass die Matrix A nur in (3.29) auftritt, und zwar als Matrix-Vektor-Multiplikation $A\,\vec{x}_k$. An dieser Stelle kann man berücksichtigen,

dass A dünn besetzt ist. Erstens kann man z.B. die Matrix-Vektor-Multiplikation direkt programmieren und damit darauf verzichten, die Matrix A zu speichern. Zweitens kann das Matrix-Vektor-Produkt in deutlich weniger als $\mathcal{O}(\dim^2)$ Operationen ausgeführt werden, wenn man verschwindende Einträge $a_{rs} = 0$ gar nicht erst betrachtet. Im Fall von (3.17) läßt sich der Rechenaufwand so auf $\mathcal{O}(\dim)$ Operationen reduzieren.

Abb. 3.1 illustriert das Verfahren des steilsten Abstiegs für ein zweidimensionales Beispiel. Man sieht, dass sich die Vektoren $\vec{x}_k$ allmählich dem Minimum entlang einer Zick-Zack-Bahn annähren, wobei auch bereits in Richtung des Minimums gelaufene Schritte teilweise wieder rückwärts gelaufen werden.

Eine bessere Wahl ist, wenn man nicht einfach immer in Richtung des Gradienten läuft, sondern eine Folge von Vektoren $\vec{p}_k$ wählt, die diese steilste Abstiegsrichtung möglichst gut unter der Nebenbedingung approximieren, dass die Vektoren $\vec{p}_k$ paarweise konjugiert sind. Hierbei bedeutet 'konjugiert', dass die Vektoren bzgl. A paarweise orthogonal sind:

$$\vec{p}_i \cdot A\,\vec{p}_j = 0 \tag{3.32}$$

für $i \neq j$.

Man kann zeigen [8,14], dass die folgende Iterationsvorschrift des *Conjugate Gradient* (CG) Verfahrens [16] die gewünschten Eigenschaften hat:

$$\begin{aligned}
\vec{u}_k &= A\,\vec{p}_k\,, \\
\alpha_k &= \frac{\vec{r}_k \cdot \vec{r}_k}{\vec{p}_k \cdot \vec{u}_k}\,, \\
\vec{x}_{k+1} &= \vec{x}_k + \alpha_k\,\vec{p}_k\,, \\
\vec{r}_{k+1} &= \vec{r}_k - \alpha_k\,\vec{u}_k\,, \\
\beta_k &= \frac{\vec{r}_{k+1} \cdot \vec{r}_{k+1}}{\vec{r}_k \cdot \vec{r}_k}\,, \\
\vec{p}_{k+1} &= \vec{r}_{k+1} + \beta_k\,\vec{p}_k\,.
\end{aligned} \tag{3.33}$$

Für einen gegebenen Startvektor $\vec{x}_0$ sind dabei die folgenden Anfangswerte zu wählen:

$$\vec{p}_0 = \vec{r}_0 = \vec{b} - A\,\vec{x}_0\,. \tag{3.34}$$

Beim CG Verfahren gilt außer (3.32) auch, dass die $\vec{r}_k$ paarweise orthogonal sind, d.h. $\vec{r}_i \cdot \vec{r}_j = 0$ für $i \neq j$ [8,14,16].

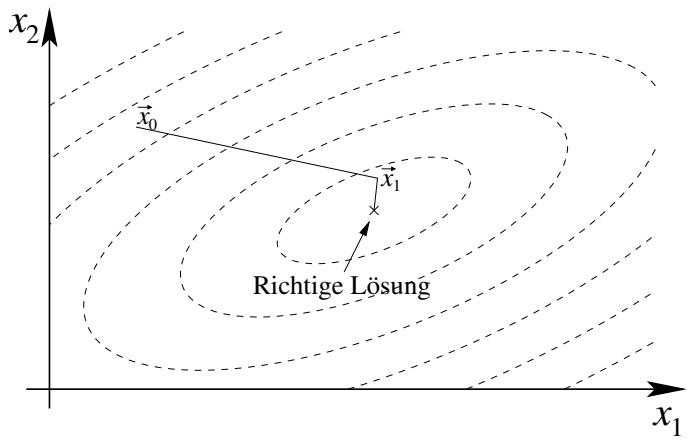


Abbildung 3.2: *Illustration des Conjugate-Gradient Verfahrens.*

Aufgrund der Orthogonalitätsrelation (3.32) der $\vec{p}_k$ ist bei exakter Arithmetik $\vec{p}_{\dim} = \vec{0}$, d.h. das CG Verfahren findet theoretisch die exakte Lösung in dim Schritten. Zur Veranschaulichung illustriert Abb. 3.2 das Verfahren an dem obigen zweidimensionalen Beispiel (dim = 2) mit demselben $\vec{x}_0$. In diesem Fall haben wir die richtige Lösung bereits nach dem 2. Schritt!

Für große Matrizen kommt man sogar meistens mit deutlich weniger als dim Schritten aus, um die Lösung mit einer gewünschten Genauigkeit zu approximieren. Man kann zeigen (vgl. Kapitel 11.10 von [16]), dass sich sowohl beim steilsten Abstieg als auch beim CG Verfahren das Fehlerquadrat im Schritt $k+1$ um den folgenden Betrag reduziert:

$$f(\vec{x}_k) - f(\vec{x}_{k+1}) = \alpha_k \, \vec{r}_k \cdot \vec{r}_k \, . \tag{3.35}$$

Die Iteration kann somit abgebrochen werden, wenn die rechte Seite von (3.35) eine vorgegebene Fehlerschwelle unterschreitet. Beim steilsten Abstieg verifiziert man (3.35) leicht, indem man die Definitionen (3.30) und (3.29) einsetzt. Beim CG Verfahren ist (3.35) eine Konsequenz der Tatsache, dass die Wahl von α_k in (3.33) für die gegebene Abstiegsrichtung $\vec{p}_k$ tatsächlich optimal ist [14].

Schließlich ist zu erwähnen, dass die Konvergenz des CG Verfahrens durch die 'Konditionszahl' von A

$$\kappa_A = \frac{\lambda_{\max}}{\lambda_{\min}} \tag{3.36}$$

bestimmt wird (vgl. Kapitel 5.3.6 von [6] und Theorem 10.2.6 von [8]), wobei $\lambda_{\max}$ und $\lambda_{\min}$ der größte bzw. kleinste Eigenwert von A sind. Dies kann man

sich dadurch plausibel machen, dass für $\kappa_A = 1$ alle Eigenwerte von A gleich sind. Dann muß $A = \lambda \mathbb{1}$ sein; der erste Abstiegsvektor $\vec{p}_0$ in (3.34) zeigt direkt auf das Minimum und das CG Verfahren liefert die gesuchte Lösung mit nur einem Schritt. Je größer die Konditionszahl κ_A ist (d.h. umso mehr sie von eins abweicht), desto mehr weichen die Abstiegsrichtungen $\vec{p}_k$ vom Minimum ab und umso mehr Iterationen sind erforderlich.

Auch beim CG Verfahren summieren sich die Rundungsfehler im Verlauf der Iteration. Dies hat zur Folge, dass die Orthogonalitätsrelation (3.32) umso schlechter erfüllt ist, je größer $|i - j|$ ist. Damit es diese Rundungsfehler vergißt, sollte man das CG Verfahren nach einigen Iterationen j 'neu starten', d.h. $\vec{x}_j$ als Startvektor betrachten und mit diesem neue Anfangsbedingungen nach (3.34) berechnen.

Das CG Verfahren kann im Vergleich zur hier vorgestellten Variante durchaus noch verbessert werden, z.B. durch den Einsatz eines Vorkonditionierers. Ein gut implementiertes CG Verfahren ist für die Lösung der Diskretisierung einer partiellen Differentialgleichung bei fester Genauigkeit normalerweise deutlich schneller als das Jacobi-Verfahren aus Kapitel 3.2. Das CG Verfahren und seine Verallgemeinerungen können darüber hinaus immer dann eingesetzt werden, wenn lineare Gleichungssysteme mit dünn besetzten Matrizen A auftreten.

Am Ende dieses Kapitels sei nur kurz erwähnt, dass beachtliche Leistungssteigerungen für die diskutierten Randwertprobleme durch den Einsatz sogenannter Mehrgitter-Verfahren erzielt werden können (siehe z.B. Kapitel 19.6 von [24], Kapitel 5.6 von [12], sowie [3]). Der Grundgedanke dieser Verfahren ist, schnellere Konvergenz durch die Kombination von Gittern mit unterschiedlicher Auflösung zu erzielen. Das Jacobi-Verfahren benötigt z.B. viele Iterationen, bis durch wiederholte Mittelung die Randbedingungen in das Innere des Gebiets propagiert werden. Es liegt somit nahe, dass man deutlich schneller zum Ziel gelangt, wenn man zuerst auf einem groben Gitter eine genäherte Lösung für das Gesamtsystem bestimmt, diese dann auf ein feineres Gitter überträgt, wo dann nur noch lokale Feinkorrekturen durchzuführen sind. Ähnliches gilt auch für das CG Verfahren: Lösungen auf einem Gitter anderer Auflösung können als Startvektoren $\vec{x}_0$ verwendet werden. Da diese bereits eine gute Näherung dastellen, konvergiert das CG Verfahren nun mit wenigen Iterationen gegen die gesuchte Lösung. Tatsächlich ist die Kombination von Mehrgitter- und CG Verfahren besonders leistungsfähig [3].

4 Partielle Differentialgleichungen: Dynamik

Wir wollen uns nun der Zeitentwicklung von partiellen Differentialgleichungen zuwenden, d.h. wir nehmen an, dass eine ausgezeichnete Variable t existiert, die wir „Zeit“ nennen. Die Aufgabenstellung besteht nun darin, zu gegebenen Anfangsdaten für eine Zeit t_0 die Entwicklung des Systems für spätere Zeiten zu bestimmen. In der Physik tritt diese Aufgabe bei vielen zeitabhängigen Problemen z.B. in der Quantenmechanik oder der Elektrodynamik auf; in der Mathematik wird diese Fragestellung als Anfangswertproblem bezeichnet.

Das diskretisierte Anfangswertproblem wird in Abb. 4.1 veranschaulicht. Nun sind Anfangswerte für eine Vektor-wertige Funktion $\vec{u}$ gegeben und die Diskretisierung der partiellen Differentialgleichungen diktiert die Entwicklung der Funktion zu späteren Zeiten t. Auch hier benötigt man Randbedingungen, allerdings nur beim jeweils nächsten Zeitpunkt, um dort das Verhalten der Funktion festzulegen. Das Anfangswertproblem kann nacheinander für diskrete Zeitschritte gelöst werden, so dass andere Algorithmen als z.B. die in Kapitel 3 vorgestellten Verfahren für Randwertprobleme zum Einsatz kommen. Auch zu diesem Thema gibt es umfangreiche Literatur, insbesondere sei wieder auf Kapitel 19 von [24], sowie [12,25] und Kapitel 6 von [6] verwiesen; einige nützliche Bemerkungen findet man auch in Kapitel 10 von [29].

Wir setzen eine lineare Zeitabhängigkeit voraus und schreiben allgemein

$$\frac{\partial \vec{u}}{\partial t} = \mathcal{L}\left(\vec{u}(\vec{x},t),\vec{x},t\right) \ . \tag{4.1}$$

Hierbei ist $\mathcal{L}$ ein (möglicherweise nicht-linearer) partieller Differentialoperator

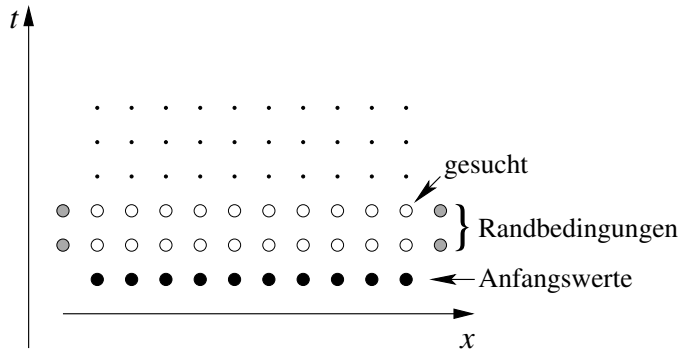


Abbildung 4.1: *Das diskretisierte Anfangswertproblem.*

bzgl. der räumlichen Komponenten $\vec{x}$.

Die Annahme einer ersten Zeitableitung in (4.1) ist tatsächlich keine Einschränkung. Ähnlich wie bei gewöhnlichen Differentialgleichungen (siehe Kapitel 1.3.1) können nämlich allgemeine Probleme höherer Ordnung in der Zeit t auf die Form (4.1) gebracht werden, sofern sie linear in der Zeit sind. Dies wollen wir an der Wellengleichung

$$\frac{\partial^2 \Psi(\vec{x},t)}{\partial t^2} = c^2\, \Delta \Psi(\vec{x},t) \tag{4.2}$$

illustrieren. Nebenbei sei angemerkt, dass c für die Ausbreitungsgeschwindigkeit steht, insbesondere ist c in der Elektrodynamik die Lichtgeschwindigkeit.

Wir schreiben

$$\vec{u} = \begin{pmatrix} \Psi \\ \frac{\partial}{\partial t}\, \Psi \end{pmatrix}$$

und können (4.2) dann tatsächlich in die Form

$$\frac{\partial}{\partial t}\, \vec{u} = \begin{pmatrix} 0 & 1 \\ c^2\, \Delta & 0 \end{pmatrix} \vec{u} = \mathcal{L}(\vec{u}) \tag{4.3}$$

bringen. Man beachte, dass aufgrund der zweiten Ableitung in (4.2) Anfangsbedingungen sowohl für die Funktion Ψ als auch für ihre Ableitung $\frac{\partial}{\partial t}\, \Psi$ vorgegeben werden müssen. Alternativ kann man bei der Wellengleichung (4.2) die Werte von Ψ zu einem Anfangszeitpunkt t_a und einem Endzeitpunkt t_e vorgeben. Man gelangt dann zurück zu einem Randwertproblem für $t_a \leq t \leq t_e$. Im folgenden wollen wir uns jedoch mit dem Anfangswertproblem beschäftigen.

Neben der räumlichen Diskretisierung (3.2) verwenden wir nun auch die diskreten Zeiten t_n, die bereits in (1.5) eingeführt wurden. Ferner führen wir die Notation

$$\vec{u}_{\vec{r}}^{\,(n)} = \vec{u}(\vec{x}_{\vec{r}}, t_n) \tag{4.4}$$

ein.

Das Ziel ist nun die Berechnung von $\vec{u}_{\vec{r}}^{\,(n+1)}$ aus bekannten $\vec{u}_{\vec{r}}^{\,(n)}$ (sowie ggfs. $\vec{u}_{\vec{r}}^{\,(n-1)}$, $\vec{u}_{\vec{r}}^{\,(n-2)}$, ...). Formal wird das Problem durch Integration von (4.1) gelöst:

$$\vec{u}_{\vec{r}}^{\,(n+1)} = \vec{u}_{\vec{r}}^{\,(n)} + \int_{t_n}^{t_{n+1}} \mathrm{d}t\, \mathcal{L}(\vec{u}) \tag{4.5}$$

Diese Integration ist in Abb. 4.2 veranschaulicht.

Praktisch muß das Integral in (4.5) näherungsweise ausgewertet werden. Einige einfache Möglichkeiten sind:

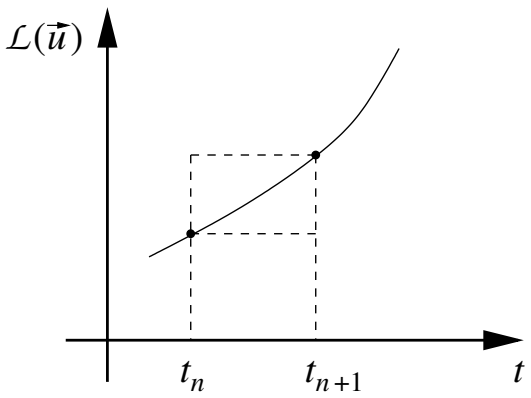


Abbildung 4.2: *Die in der diskreten Zeitentwicklung (4.5) auftretende Integration.*

i. Man nähert das Integral durch ein Rechteck mit der Höhe des linken Randpunkts:

$$\int_{t_n}^{t_{n+1}} \mathrm{d}t\, \mathcal{L}(\vec{u}) \approx \Delta t\, \mathcal{L}(\vec{u}^{(n)})\,.$$

Dies führt auf das *explizite Euler-Verfahren*. Nach (4.5) ist $\vec{u}^{(n+1)}$ mit dieser Näherung direkt („explizit“) für gegebenes $\vec{u}^{(n)}$ berechenbar

$$\vec{u}^{(n+1)} = \vec{u}^{(n)} + \Delta t\, \mathcal{L}(\vec{u}^{(n)})\,. \tag{4.6}$$

Diese Vorschrift zur Berechnung der Zeitentwicklung ähnelt stark dem Euler-Verfahren (1.20) zur Integration gewöhnlicher Differentialgleichungen.
Das explizite Euler-Verfahren (4.6) ist erster Ordnung in der Zeit, wie man leicht einsieht, wenn man $\vec{u}$ um t_n Taylor-entwickelt:

$$\left.\frac{\partial \vec{u}}{\partial t}\right|_{t_n} = \frac{\vec{u}^{(n+1)} - \vec{u}^{(n)}}{\Delta t} - \frac{1}{2}\,\Delta t\, \left.\frac{\partial^2 \vec{u}}{\partial t^2}\right|_{t_n} + \mathcal{O}(\Delta t^2)\,. \tag{4.7}$$

ii. Man kann auch den rechten Randpunkt verwenden, bzw. $\vec{u}$ um t_{n+1} Taylor-entwickeln:

$$\left.\frac{\partial \vec{u}}{\partial t}\right|_{t_{n+1}} = \frac{\vec{u}^{(n+1)} - \vec{u}^{(n)}}{\Delta t} + \frac{1}{2}\,\Delta t\, \left.\frac{\partial^2 \vec{u}}{\partial t^2}\right|_{t_{n+1}} + \mathcal{O}(\Delta t^2)\,. \tag{4.8}$$

Dies führt nun auf das *implizite Euler-Verfahren*

$$\vec{u}^{(n+1)} = \vec{u}^{(n)} + \Delta t\, \mathcal{L}(\vec{u}^{(n+1)})\,. \tag{4.9}$$

Das unbekannte $\vec{u}^{(n+1)}$ taucht nun auf beiden Seiten der Gleichung auf, es handelt sich also um eine implizite Gleichung für $\vec{u}^{(n+1)}$, d.h. ein (nicht-)lineares Gleichungssystem für die unbekannten $\vec{u}_{\vec{r}}^{(n+1)}$. Auch dieses Verfahren ist erster Ordnung in Δt.

iii. Schließlich kann man durch Kombination von (4.7) und (4.8) in bekannter Weise die Ordnung verbessern. Unter Berücksichtigung von

$$\left.\frac{\partial^2 \vec{u}}{\partial t^2}\right|_{t_{n+1}} = \left.\frac{\partial^2 \vec{u}}{\partial t^2}\right|_{t_n} + \mathcal{O}(\Delta t)$$

ist nämlich

$$\frac{1}{2}\left(\left.\frac{\partial \vec{u}}{\partial t}\right|_{t_{n+1}} + \left.\frac{\partial \vec{u}}{\partial t}\right|_{t_n}\right) = \frac{\vec{u}^{(n+1)} - \vec{u}^{(n)}}{\Delta t} + \mathcal{O}(\Delta t^2)\,. \tag{4.10}$$

Man erhält damit das *implizite Euler-Verfahren 2. Ordnung*

$$\vec{u}^{(n+1)} = \vec{u}^{(n)} + \frac{\Delta t}{2}\left(\mathcal{L}(\vec{u}^{(n)}) + \mathcal{L}(\vec{u}^{(n+1)})\right)\,. \tag{4.11}$$

4.1 Explizite Lösungsverfahren

In diesem Unterkapitel werden wir zuerst das explizite Euler-Verfahren genauer untersuchen und ein weiteres explizite Lösungsverfahren vorstellen, das Nachteile des Euler-Verfahrens behebt.

4.1.1 Explizites Euler-Verfahren

Das explizite Euler-Verfahren (4.6) ist leicht zu implementieren und sehr schnell. Es hat aber leider einen großen Nachteil: Es ist unbrauchbar.

Die Probleme des Euler-Verfahrens werden durch eine Stabilitätsanalyse offengelegt. Als Beispiel verwenden wir die partielle Differentialgleichung

$$\frac{\partial u}{\partial t} = -v\,\frac{\partial u}{\partial x} \tag{4.12}$$

Die Lösung dieser Differentialgleichung zu gegebenen Anfangsbedingungen $u(x, t_0) = u_0(x)$ kann sofort angegeben werden:

$$u(x,t) = u_0\left(x - v\,(t - t_0)\right)\,. \tag{4.13}$$

Dies bedeutet, dass sich ein vorgegebenes Profil der Funktion u mit konstanter Geschwindigkeit v unter Beibehaltung seiner Form nach rechts bewegt.

Die partielle Ableitung nach x in (4.12) diskretisieren wir gemäß (3.5) und erhalten für diesen Fall des expliziten Euler-Verfahrens

$$u_r^{(n+1)} = u_r^{(n)} - \frac{v\,\Delta t}{2\,\Delta x}\left(u_{r+1}^{(n)} - u_{r-1}^{(n)}\right) . \tag{4.14}$$

Man betrachtet nun Eigenmoden des Differenzenoperators mit festem k

$$u_r^{(n)} = g(n,k)\,\mathrm{e}^{i\,k\,r\,\Delta x} . \tag{4.15}$$

Durch Einsetzen in (4.14) überzeugt man sich, dass (4.15) tatsächlich auf eine Lösung führt, sofern die Koeffizienten $g(n,k)$ die folgende Rekursionsrelation erfüllen:

$$g(n+1,k) = g(n,k)\,\left(1 - i\frac{v\,\Delta t}{\Delta x}\sin(k\,\Delta x)\right) . \tag{4.16}$$

Diese Rekursionsrelation wird gelöst durch

$$g(n,k) = g(k)^n\,g(0,k) \qquad \text{mit} \qquad g(k) = 1 - i\frac{v\,\Delta t}{\Delta x}\sin(k\,\Delta x) . \tag{4.17}$$

Offensichtlich ist für $k \neq 0$

$$|g(k)|^2 = 1 + \left(\frac{v\,\Delta t}{\Delta x}\sin(k\,\Delta x)\right)^2 > 1 . \tag{4.18}$$

Die Amplitude $g(n,k)$ wächst also für alle $k \neq 0$. Dies ist in krassem Gegensatz zur bekannten Lösung (4.13) und bedeutet, dass das Verfahren immer instabil ist! Natürlich liefert (4.14) für kurze Zeiten trotzdem vernünftige Lösungen. Die genaue Zeitspanne hängt davon ab, wie groß der zweite Term in (4.18) ist, d.h. insbesondere wie groß k bzw. wie glatt die Funktion $u_0(x)$ ist. Früher oder später wird man aber immer eine nicht mehr sinnvolle Näherung erhalten.

Im allgemeinen kann man eine „von Neumann-Stabilitätsanalyse“ durchführen. Für eine gegebene nicht-lineare Differentialgleichung in $\vec{u}$ schreibt man $\vec{u} = \vec{u}_0 + \delta\vec{u}$ und linearisiert durch Entwicklung bis zur ersten Ordnung in $\delta\vec{u}$, wobei man annimmt, dass $\vec{u}_0$ bereits eine exakte Lösung der Differenzengleichung ist. Die linearisierte Gleichung für $\delta\vec{u}$ ist vom Typ (4.12), wobei der Koeffizient v natürlich vom Entwicklungspunkt $\vec{u}_0$ abhängt. Zu beantworten ist dann die Frage, ob eine instabile Eigenmode $\delta\vec{u}$ existiert. Die Existenz einer Instabilität bedeutet z.B., dass sich Rundungsfehler aufschaukeln werden und ist somit unerwünscht. Das oben vorgeführte Argument zeigt demnach, dass das explizite Euler-Verfahren immer (d.h. für allgemeine Differentialoperatoren $\mathcal{L}$ und allgemeine Anfangsbedingungen) instabil ist.

4.1.2 Lax-Verfahren

Die Instabilität des expliziten Euler-Verfahrens kann darauf zurückgeführt werden, dass sich insbesondere Schwankungen mit großen k schnell aufschaukeln. Dies legt eine Regularisierung des Verfahrens durch Glättung nahe. Man ersetzt in der Entwicklungsvorschrift (4.6) den Wert $\vec{u}_{\vec{r}}^{(n)}$ durch den Mittelwert seiner Nachbarpunkte. In einer räumlichen Dimension ersetzt man also $\vec{u}_r^{(n)}$ in (4.6) durch $\frac{1}{2}\left(\vec{u}_{r+1}^{(n)} + \vec{u}_{r-1}^{(n)}\right)$ und erhält so das *explizite Lax-Verfahren*

$$\vec{u}_r^{(n+1)} = \frac{1}{2}\left(\vec{u}_{r+1}^{(n)} + \vec{u}_{r-1}^{(n)}\right) + \Delta t\, \mathcal{L}(\vec{u}^{(n)})_r \,. \tag{4.19}$$

Die Stabilitätsanalyse führen wir wieder repräsentativ für die einfache Differentialgleichung (4.12) durch. Das Lax-Verfahren führt nun auf

$$u_r^{(n+1)} = \frac{1}{2}\left(u_{r+1}^{(n)} + u_{r-1}^{(n)}\right) - \frac{v\,\Delta t}{2\,\Delta x}\left(u_{r+1}^{(n)} - u_{r-1}^{(n)}\right) \,. \tag{4.20}$$

Wir betrachten wieder die Eigenmoden (4.15), setzen diese in (4.20) ein und erhalten mit $g(n,k) = g(k)^n\, g(0,k)$ folgende Lösung

$$g(k) = \cos(k\,\Delta x) - i\frac{v\,\Delta t}{\Delta x}\sin(k\,\Delta x)\,. \tag{4.21}$$

Man beachte, dass der erste Summand im Gegensatz zu (4.17) nun kleiner als eins ist. Für den Betrag gilt nun

$$|g(k)|^2 = 1 - \left(1 - \left(\frac{v\,\Delta t}{\Delta x}\right)^2\right)\sin^2(k\,\Delta x)\,. \tag{4.22}$$

Damit die Lösung nicht anwächst, sollte $|g(k)| \leq 1$ für alle k sichergestellt sein. Aus (4.22) liest man ab, dass in der Tat $|g(k)| \leq 1$ gilt, wenn die *Courant-Friedrichs-Lewy* Stabilitätsbedingung

$$|v|\,\frac{\Delta t}{\Delta x} \leq 1 \tag{4.23}$$

erfüllt ist.

Das Lax-Verfahren und insbesondere die Stabilitätsbedingung (4.23) ist in Abb. 4.3 veranschaulicht. Die Bedingung (4.23) bedeutet nun, dass der durch die Pfeile definierte Kegel den grau schaffierten physikalischen Ausbreitungskegel enthält, d.h. dass die numerische Ausbreitungsgeschwindigkeit $\frac{\Delta x}{\Delta t}$ größer ist als der Betrag der physikalischen Geschwindigkeit v: $\frac{\Delta x}{\Delta t} \geq |v|$.

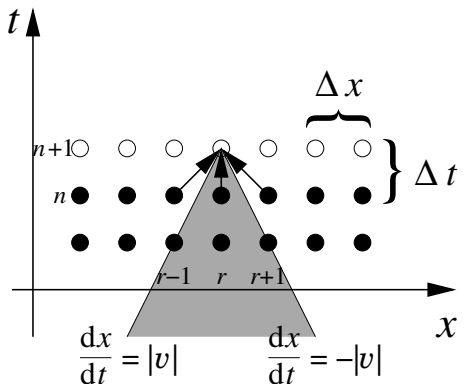


Abbildung 4.3: *Das Lax-Verfahren (4.19). Der Informationsfluß ist durch Pfeile angedeutet; der graue Bereich zeigt den physikalischen Ausbreitungskegel.*

Es ist ferner instruktiv, (4.19) in die Form des expliziten Euler-Verfahrens (4.6) umzuschreiben:

$$\begin{aligned} \vec{u}_r^{(n+1)} &= \vec{u}_r^{(n)} + \frac{1}{2}\left(\vec{u}_{r+1}^{(n)} - 2\vec{u}_r^{(n)} + \vec{u}_{r-1}^{(n)}\right) + \Delta t\, \mathcal{L}(\vec{u}^{(n)})_r \\ &=: \vec{u}_r^{(n)} + \Delta t\, \widetilde{\mathcal{L}}(\vec{u}^{(n)})_r\,. \end{aligned} \tag{4.24}$$

Den Übergang von (4.6) zu (4.19) kann man folglich durch Addition eines Terms zu dem partiellen Differentialoperator $\mathcal{L}(\vec{u})$ interpretieren, dessen Kontinuumsform

$$\widetilde{\mathcal{L}}(\vec{u}) = \mathcal{L}(\vec{u}) + \frac{1}{2}\,\frac{(\Delta x)^2}{\Delta t}\,\Delta\vec{u} \tag{4.25}$$

lautet. Dieser Zusatzterm hat die Form eines Diffusionsterms! Beim Übergang vom expliziten Euler- zum Lax-Verfahren wurde also eine numerische Dämpfung eingeführt (natürlich mit einem in Abhängigkeit von der Diskretisierung geeignet gewählten Koeffizienten). Dieser Zusatzterm dämpft insbesondere hohe k-Moden und verhindert auf diese Weise deren Anwachsen. Zwar werden auch die kleinen k-Moden gedämpft, so dass auch hier bei langen Zeiten Artefakte auftreten. Die Dämpfung ist bei kleinen k jedoch weniger stark. Mit dem Lax-Verfahren kann man somit langwellige Eigenschaften (kleine k) über Zeiträume verfolgen, bei denen im expliziten Euler-Verfahren die hohen k-Moden, die sowieso im Bereich von Diskretisierungs-Artefakten liegen, bereits stark angewachsen wären und die Lösung unbrauchbar gemacht hätten.

Tatsächlich handelt es sich um eine Standard-Vorgehensweise, numerische Verfahren durch Addition geeignet gewählter Dämpfungsterme zu stabilisieren (siehe z.B. [25]). Solche numerischen Verfahren können durchaus Lösungen liefern, die von einer mathematisch exakten Lösung der ursprünglichen partiellen Differentialgleichung abweichen. Derartige Abweichungen mögen jedoch vertretbar sein, denn die Natur regularisiert Singularitäten ebenfalls durch Einführung von Dissipation. Aus diesem Grund kann man sich vorstellen, dass ein regularisiertes numerisches Verfahren ein gegebenes physikalisches Problem sogar genauer beschreibt als die ursprüngliche idealisierte Beschreibung mit einer partiellen Differentialgleichung.

Das Lax-Verfahren (4.19) ist ein erstes Verfahren, das stabil und somit brauchbar ist. Leider ist es nur von erster Ordnung in Δt. Arbeitet man mit zweiter Ordnung Genauigkeit in Δx muß man also Δt entsprechend klein wählen, d.h. sehr viele Zeitschritte ausführen um die Entwicklung über einen Zeitraum zu berechnen, der einem räumlichen Gitterabstand entspricht. Nimmt man z.B. $\vec{u}^{(n-1)}$ zur Hilfe, so kann man explizite Verfahren konstruieren, die sowohl höherer Ordnung in Δt, stabil sowie ggfs. frei von Dämpfung sind. Ein verbreitetes Verfahren ist das „Leapfrog"- (Bocksprung)-Verfahren; es wird z.B. bei der Simulation von Tsunamis eingesetzt [9]. Wir wollen auf solche Verfahren hier jedoch nicht weiter eingehen, sondern verweisen auf die Literatur [12,24,25].

4.1.3 Diffusionsgleichung

Wir machen nun einen Einschub zu der bereits erwähnten Diffusionsgleichung. Die Diffusion beschreibt z.B. das Verhalten vieler Brownscher Teilchen. Dies kann man benutzen, um die Diffusionsgleichung aus der Brownschen Bewegung abzuleiten. Wie wir sehen werden, führt dies auf ein einfaches Beispiel für eine Zeitentwicklungs-Gleichung vom Typ (4.1).

Konkret betrachten wir die Bewegung eines Teilchens auf einem hyperkubischen Gitter in d Dimensionen mit Gitterabstand Δx. Diese sei dadurch charakterisisert, dass das Teilchen in jedem Zeitschritt *zufällig* auf einen seiner $2d$ Nachbarplätze gehe. Die Wahrhscheinlichkeit, dass das Teilchen auf einen gegebenen Nachbarplatz geht, ist demnach in jedem Zeitschritt $1/2d$. Nun kann man eine Funktion $n(\vec{x}, t)$ einführen, die die Wahrscheinlichkeit beschreibt, das Teilchen zur Zeit t auf einem Platz $\vec{x}$ zu finden. Für diese Wahrscheinlichkeitsdichte

gilt folgende Bilanzgleichung:

$$\begin{aligned} n(\vec{x}, t+\Delta t) &= n(\vec{x}, t) \\ &\quad + \frac{1}{2d}\left(-2d\, n(\vec{x}, t) + \sum_{i=1}^{d} \{n(\vec{x}+\Delta x\, \vec{e}_i, t) + n(\vec{x}-\Delta x\, \vec{e}_i, t)\}\right). \end{aligned} \tag{4.26}$$

Die zunächst etwas künstlich wirkende Darstellung der Null durch die ersten beiden Terme auf der rechten Seite macht deutlich, dass ein am Platz $\vec{x}$ sitzendes Teilchen diesen in jedem Fall verlassen muß. Die letzten beiden Summanden beschreiben Prozesse, in denen ein Teilchen von einem der $2d$ Nachbarplätze kommt. Andererseits erkennen wir in den runden Klammern die diskretisierte Form des Laplace-Operators (3.13) wieder. Wir können (4.26) somit als Euler-diskretisierte Form der Diffusionsgleichung interpretieren:

$$\frac{\partial}{\partial t} n(\vec{x}, t) = D\, \Delta n(\vec{x}, t) \tag{4.27}$$

mit der Diffusionskonstanten $D = \frac{(\Delta x)^2}{2d\, \Delta t}$. Andererseits werden Diskretisierungs-Details im Kontinuums-Grenzfall $\Delta x \to 0$, $\Delta t \to 0$ irrelevant. Somit liefert (4.27) generell die makroskopische Sichtweise für einen mikroskopischen Brownschen Prozess.

Es ist nun nicht besonders schwer, Lösungen der Diffusionsgleichung (4.27) zu untersuchen. Wir wollen dies kurz im Fall einer Dimension illustrieren. Zunächst betrachten wir das Integral

$$I(t) = \int_{x_0}^{x_1} \mathrm{d}x\, n(x, t)\,, \tag{4.28}$$

wobei die untere Grenze x_0 sowie die obere Grenze x_1 des Intervalls durchaus auch $\pm\infty$ sein können. Aufgrund (4.27) gilt

$$\frac{\partial}{\partial t} I(t) = \int_{x_0}^{x_1} \mathrm{d}x\, \frac{\partial}{\partial t} n(x, t) \overset{(4.27)}{=} D \int_{x_0}^{x_1} \mathrm{d}x\, \frac{\partial^2}{\partial x^2} n(x, t) = D\, \frac{\partial}{\partial x} n(x, t)\Big|_{x_0}^{x_1} = 0\,, \tag{4.29}$$

vorausgesetzt, $\frac{\partial}{\partial x} n(x, t)$ verschwindet am Rand. Dies ist z.B. der Fall, wenn man Neumannsche Randbedingungen auf einem endlichen Intervall betrachtet, oder aber auch für $x_0 = -\infty$ und $x_1 = +\infty$, da hier sowohl die Funktion $n(x, t)$ als auch deren räumlichen Ableitungen hinreichend schnell abfallen sollten. Physikalisch

bedeutet dies, dass Teilchen nicht über den Rand in das System hineinkommen oder es verlassen und somit die Teilchenzahl erhalten ist. Somit ist $I(t)$ konstant wie durch (4.29) ausgedrückt wird.

Analog kann man das erste Moment von x betrachten:

$$\begin{aligned}
\frac{\partial}{\partial t}\int_{x_0}^{x_1} \mathrm{d}x\, x\, n(x,t) &\overset{(4.27)}{=} D\int_{x_0}^{x_1} \mathrm{d}x\, x\, \frac{\partial^2}{\partial x^2} n(x,t) \\
&\overset{\text{partiell}}{=} D\, x\, \frac{\partial}{\partial x} n(x,t)\Big|_{x_0}^{x_1} - D\int_{x_0}^{x_1} \mathrm{d}x\, \frac{\partial}{\partial x} n(x,t) \\
&= D\, x\, \frac{\partial}{\partial x} n(x,t)\Big|_{x_0}^{x_1} - D\, n(x,t)|_{x_0}^{x_1} = 0\,, \qquad (4.30)
\end{aligned}$$

wiederum unter der Voraussetzung, dass die Randterme verschwinden.

Schließlich gilt für das zweite Moment von x (wir gehen wieder wie oben von verschwindenden Randtermen aus)

$$\begin{aligned}
\frac{\partial}{\partial t}\int_{x_0}^{x_1} \mathrm{d}x\, x^2\, n(x,t) &\overset{(4.27)}{=} D\int_{x_0}^{x_1} \mathrm{d}x\, x^2\, \frac{\partial^2}{\partial x^2} n(x,t) \\
&\overset{\text{partiell}}{=} D\, x^2\, \frac{\partial}{\partial x} n(x,t)\Big|_{x_0}^{x_1} - 2\, D\int_{x_0}^{x_1} \mathrm{d}x\, x\, \frac{\partial}{\partial x} n(x,t) \\
&\overset{\text{partiell}}{=} D\, x^2\, \frac{\partial}{\partial x} n(x,t)\Big|_{x_0}^{x_1} - 2\, D\, x\, n(x,t)|_{x_0}^{x_1} + 2\, D\int_{x_0}^{x_1} \mathrm{d}x\, n(x,t) \\
&\overset{(4.28)}{=} 2\, D\, I(t)\,. \qquad (4.31)
\end{aligned}$$

Als Konsequenz von (4.29), (4.30) und (4.31) folgt für die normierten Erwartungswerte

$$\begin{aligned}
\langle x\rangle(t) &= \frac{1}{I(t)}\int_{x_0}^{x_1} \mathrm{d}x\, x\, n(x,t) = \langle x\rangle(0)\,, \\
\langle x^2\rangle(t) &= \frac{1}{I(t)}\int_{x_0}^{x_1} \mathrm{d}x\, x^2\, n(x,t) = 2\, D\, t + \langle x^2\rangle(0)\,. \qquad (4.32)
\end{aligned}$$

Man erkennt das für Diffusion charakteristische Verhalten: ein Teilchen legt über den Zeitraum eine Strecke zurück, die mit $\sqrt{t}$ wächst; da der Weg jedoch in jede

beliebige Richtung gehen kann, gibt es im Mittel über alle Wege keine Fortbewegung ($\langle x \rangle(t)$ ist konstant).

Auch explizite Lösungen der Diffusionsgleichung (4.27) können beim Fehlen von Randtermen relativ leicht angegeben werden. Für den Fall einer Dimension und $x \in \mathbb{R}$ (d.h. $x_0 = -\infty$, $x_1 = +\infty$) kann die Lösung von (4.27) zu gegebenen Anfangsbedingungen $n(x,0)$ z.B. als Integral dargestellt werden:

$$n(x,t) = \frac{1}{\sqrt{4\,\pi\,D\,t}} \int\limits_{-\infty}^{\infty} \mathrm{d}x_0\, n(x_0,0)\, \mathrm{e}^{-\frac{(x-x_0)^2}{4\,D\,t}} \,. \tag{4.33}$$

4.2 Implizites Euler-Verfahren 2. Ordnung

Zu Beginn dieses Kapitels hatten wir auch das implizite Euler-Verfahren 2. Ordnung (4.11) vorgestellt, das in der Literatur auch unter dem Namen Crank-Nicholson-Verfahren bekannt ist (siehe Kapitel 19 von [24], Kapitel 10.4 von [29] und Kapitel 6.2 von [12]). Dieses wollen wir im folgenden ein wenig genauer diskutieren (siehe auch [25]).

Wir führen wieder eine Stabilitätsanalyse anhand der Differentialgleichung (4.12) durch. Für diese Gleichung führt (4.11) auf

$$u_r^{(n+1)} = u_r^{(n)} - \frac{v\,\Delta t}{4\,\Delta x} \left(u_{r+1}^{(n)} - u_{r-1}^{(n)} + u_{r+1}^{(n+1)} - u_{r-1}^{(n+1)} \right) \,. \tag{4.34}$$

Wie gewohnt setzt man die Eigenmoden (4.15) in (4.34) ein und findet

$$g(n+1,k) = g(n,k) - i\frac{v\,\Delta t}{2\,\Delta x} \sin(k\Delta x)\,(g(n,k) + g(n+1,k)) \,. \tag{4.35}$$

Auflösung nach $g(n+1,k)$ führt auf

$$g(n+1,k) = \frac{1 - i\frac{v\,\Delta t}{2\,\Delta x}\sin(k\Delta x)}{1 + i\frac{v\,\Delta t}{2\,\Delta x}\sin(k\Delta x)}\, g(n,k) \,. \tag{4.36}$$

Da der Vorfaktor vom Betrag eins ist, ändert sich die Amplitude $g(n,k)$ nicht:

$$|g(n+1,k)| = |g(n,k)| \,. \tag{4.37}$$

Diese Beobachtung erlaubt bereits eine Auflistung wichtiger Eigenschaften des impliziten Euler-Verfahrens 2. Ordnung:

- $\oplus$ Es ist immer stabil (die Amplitude $g(n,k)$ wächst nicht).

⊕ Es tritt keine Dämpfung auf (die Amplitude $g(n,k)$ nimmt auch nicht ab – siehe (4.37)).

⊕ Es ist nach Konstruktion 2. Ordnung in Δt.

⊖ Die guten Eigenschaften basieren auf einer simultanen Propagation der Information an allen Plätzen in einem Zeitschritt. Dies bedeutet umgekehrt eine unendliche numerische Ausbreitungsgeschwindigkeit von Information. Störungen an einer Stelle können also instantan Auswirkungen an beliebig weit entfernten Orten haben. Je nach Fragestellung kann ein solches Verhalten unphysikalisch sein.

⊖ $\vec{u}^{(n+1)}$ ist als Lösung einer impliziten Gleichung zu bestimmen. Dies macht implizite Verfahren aufwendiger als explizite Verfahren.

Für eine effiziente Implementierung benötigt man also insbesondere ein schnelles Verfahren zur Lösung linearer Gleichungssysteme. Es liegt nahe, an die Verwendung des aus Abschnitt 3.3.1 bekannten CG Verfahrens zu denken. Dies ist jedoch im allgemeinen nicht direkt möglich, wie z.B. die folgende Umordnung von (4.34) bereits im linearen Fall zeigt:

$$-\frac{v\,\Delta t}{4\,\Delta x}\,u_{r-1}^{(n+1)}+u_r^{(n+1)}+\frac{v\,\Delta t}{4\,\Delta x}\,u_{r+1}^{(n+1)}=\frac{v\,\Delta t}{4\,\Delta x}\,u_{r-1}^{(n)}+u_r^{(n)}-\frac{v\,\Delta t}{4\,\Delta x}\,u_{r+1}^{(n)}\,. \quad (4.38)$$

Diese Gleichung hat die Form $A\vec{u}^{(n+1)}=A^\dagger\vec{u}^{(n)}=\vec{b}$. Leider ist die Matrix A nicht-hermitesch (man beachte die unterschiedlichen Vorzeichen in (4.38) vor $u_{r-1}^{(n+1)}$ und $u_{r+1}^{(n+1)}$)! Eine der Voraussetzungen für die Anwendung des CG Verfahrens ist also nicht erfüllt.

4.2.1 Bi-Conjugate Gradient Verfahren

Ist A in dem linearen Gleichungssystem (3.24) nicht hermitesch, so führt Multiplikation mit $A^\dagger$ auf

$$\tilde{A}\,\vec{x}=A^\dagger A\,\vec{x}=A^\dagger\vec{b}=\tilde{\vec{b}}\,, \quad (4.39)$$

d.h. auf ein lineares Gleichungssystem $\tilde{A}\,\vec{x}=\tilde{\vec{b}}$ mit hermiteschem $\tilde{A}=A^\dagger A$. $\tilde{A}$ ist außerdem positiv (semi-)definit, d.h. $\vec{x}\cdot\tilde{A}\vec{x}\geq 0$ für beliebige Vektoren $\vec{x}$.

Zur Lösung eines linearen Gleichungssystems (3.24) mit nicht-hermiteschem A könnte man also das CG Verfahren auf $\tilde{A}=A^\dagger A$ und $\tilde{\vec{b}}=A^\dagger\vec{b}$ anwenden.

Gegen diese Vorgehensweise sprechen jedoch Konvergenz-Eigenschaften. Wie in Unterkapitel 3.3.1 erwähnt, wird die Konvergenz durch die in (3.36) definierte Konditionszahl kontrolliert. Nun ist

$$\kappa_{\tilde{A}} = \kappa_A^\star \kappa_A \,. \tag{4.40}$$

Dies führt zu entsprechend ungünstigen Konvergenz-Eigenschaften des CG Verfahrens bei Verwendung der Matrix $\tilde{A}$.

Das Bi-Conjugate Gradient Verfahren (siehe z.B. Kapitel 2.7 von [24], Kapitel 11.11 von [16] und Kapitel 5.3.7 von [6]) liefert hier günstigeres Verhalten. Es handelt sich um eine Verallgemeinerung des CG Verfahrens, das mit einem doppelten Satz Vektoren $\vec{p}_i$, $\vec{r}_i$, $\vec{u}_i$, $\hat{\vec{p}}_i$, $\hat{\vec{r}}_i$, $\hat{\vec{u}}_i$ arbeitet. Die Vektoren $\hat{\vec{r}}_i$ und $\vec{r}_i$ sind paarweise „biorthogonal“:

$$\hat{\vec{r}}_i \cdot \vec{r}_j = 0 \qquad \text{für } i \neq j. \tag{4.41}$$

Ferner gilt als Verallgemeinerung von (3.32), dass die Abstiegsrichtungen $\hat{\vec{p}}_i$ und $\vec{p}_i$ zueinander „bikonjugiert“ sind:

$$\hat{\vec{p}}_i \cdot A\,\vec{p}_j = 0 \qquad \text{für } i \neq j. \tag{4.42}$$

Die Iterationsvorschrift des Bi-Conjugate Gradient Verfahrens lautet:

$$\begin{aligned}
\vec{u}_k &= A\,\vec{p}_k\,, & \hat{\vec{u}}_k &= A^\dagger\,\hat{\vec{p}}_k\,,\\
\alpha_k &= \frac{\hat{\vec{r}}_k \cdot \vec{r}_k}{\hat{\vec{p}}_k \cdot \vec{u}_k}\,,\\
\vec{x}_{k+1} &= \vec{x}_k + \alpha_k\,\vec{p}_k\,,\\
\vec{r}_{k+1} &= \vec{r}_k - \alpha_k\,\vec{u}_k\,, & \hat{\vec{r}}_{k+1} &= \hat{\vec{r}}_k - \alpha_k^\star\,\hat{\vec{u}}_k\,,\\
\beta_k &= \frac{\hat{\vec{r}}_{k+1} \cdot \vec{r}_{k+1}}{\hat{\vec{r}}_k \cdot \vec{r}_k}\,,\\
\vec{p}_{k+1} &= \vec{r}_{k+1} + \beta_k\,\vec{p}_k\,, & \hat{\vec{p}}_{k+1} &= \hat{\vec{r}}_{k+1} + \beta_k^\star\,\hat{\vec{p}}_k\,.
\end{aligned} \tag{4.43}$$

Schließlich sind für einen gegebenen Startvektor $\vec{x}_0$ Anfangswerte gemäß der folgenden Verallgemeinerung von (3.34) zu wählen:

$$\hat{\vec{p}}_0 = \hat{\vec{r}}_0 = \vec{p}_0 = \vec{r}_0 = \vec{b} - A\,\vec{x}_0\,. \tag{4.44}$$

Auf folgende Eigenschaften dieses Verfahrens sei besonders hingewiesen:

- Für hermitesche Matrizen $A^\dagger = A$ ist das Bi-Conjugate Gradient Verfahren äquivalent zum CG Verfahren.
- Die Konditionierung und damit die Konvergenz ist dem CG Verfahren gleich.
- Die zusätzlichen Operationen in (4.43) führen in etwa zu einer Verdopplung des Rechenaufwands im Vergleich zum CG Verfahren. Insbesondere benötigt man neben der Matrix-Vektor-Multiplikation für A auch eine für $A^\dagger$.

Bei der Anwendung im impliziten Euler-Verfahren bietet es sich offensichtlich an, $\vec{u}^{(n)}$ als Startvektor zur Bestimmung von $\vec{u}^{(n+1)}$ zu wählen. Ist die Änderung in einem Zeitschritt nicht groß, konvergiert das Verfahren entsprechend mit wenigen Iterationen gegen die gesuchte Lösung.

Zur Illustration einer möglichen Behandlung von nicht-Linearitäten betrachten wir die Burgers-Gleichung

$$\frac{\partial u}{\partial t} = -u\,\frac{\partial u}{\partial x}\,, \tag{4.45}$$

bei der $\mathcal{L}(u) = -u\,\frac{\partial u}{\partial x}$ ist. Nähert man in $\mathcal{L}(u^{(n+1)})$ den Funktionswert $u^{(n+1)} \approx u^{(n)}$, d.h. $\mathcal{L}(u^{(n+1)}) \approx -u^{(n)}\,\frac{\partial}{\partial x}\,u^{(n+1)}$, führt das implizite Euler-Verfahren (4.11) wieder auf eine lineare Gleichung[5]. Diese Vorgehensweise kann z.B. iterativ verbessert werden. Dazu betrachtet man eine Folge von Näherungen $\mathcal{L}(u^{(n+1,l+1)}) \approx -u^{(n+1,l)}\,\frac{\partial}{\partial x}\,u^{(n+1,l+1)}$ und beginnt mit $u^{(n+1,0)} = u^{(n)}$. Die skizzierte Vorgehensweise bedeutet Abbruch nach der Bestimmung von $u^{(n+1,1)} \approx u^{(n+1)}$. Man kann jedoch die Lösung des linearen Gleichungssystems mehrfach iterieren, bis die Approximationen $u^{(n+1,l)}$ eine gewünschte Genauigkeit erreicht haben.

4.3 Zeitentwicklung in der Quantenmechanik

Im Rest dieses Kapitels soll die zeitabhängige *Schrödingergleichung*

$$i\hbar\,\frac{\partial\,\Psi(\vec{r},t)}{\partial t} = \mathcal{H}\,\Psi(\vec{r},t) \tag{4.46}$$

numerisch behandelt werden. Durch Diskretisierung der Ortsvariablen wird aus dem Hamilton-Operator $\mathcal{H}$ eine hermitesche Matrix (bei der Diskretisierung ist

[5]Da die Änderungen in den Ableitungen größer sind als in der Funktion selbst, führt man diese Näherung besser für den Wert der Funktion als ihre Ableitung durch.

darauf zu achten, dass die Hermitizität erhalten bleibt). $\mathcal{H}$ besitzt somit einen vollständigen Satz von Eigenfunktionen Ψ_j zu Energie E_j:

$$\mathcal{H}\,\Psi_j = E_j\,\Psi_j\,. \tag{4.47}$$

Die Anzahl dieser Eigenvektoren ist endlich und durch die Dimension des diskretisierten Problems gegeben. Ferner sind die Eigenfunktionen paarweise orthogonal[6]

$$\int \mathrm{d}\vec{r}\,\Psi_j^\star\,\Psi_k = \delta_{j,k}\,. \tag{4.48}$$

Eine erste Möglichkeit zur Berechnung der Zeitentwicklung ist die explizite Bestimmung aller Eigenvektoren (4.47), d.h. die vollständige numerische Diagonalisierung von $\mathcal{H}$. Eine gegebene Anfangswellenfunktion kann dann (z.B. mit Hilfe von (4.48)) nach Eigenfunktionen entwickelt werden

$$\Psi(\vec{r},0) = \sum_j a_j\,\Psi_j\,. \tag{4.49}$$

Mit Hilfe von (4.47) kann die Lösung von (4.46) nun leicht angegeben werden

$$\Psi(\vec{r},t) = \sum_j a_j\,\mathrm{e}^{-i\,E_j\,t/\hbar}\,\Psi_j\,. \tag{4.50}$$

Man beachte, dass in einem solchen Zugang die Zeitentwicklung nicht diskretisiert werden muß. Allerdings bedeutet die vollständige Diagonalisierung von $\mathcal{H}$ einen erheblichen Aufwand. Trotzdem wird dieser Weg z.B. in [30] beschritten.

4.3.1 Differenzenverfahren

In den Unterkapiteln 4.1 und 4.2 wurden verschiedene numerische Verfahren zur Berechnung der diskretisierten Zeitentwicklung partieller Differentialgleichungen vorgestellt. Diese sollen nun zuerst mit Blick auf Ihre Anwendbarkeit für die Schrödingergleichung (4.46) diskutiert werden.

Zuerst stellt man fest, dass (4.46) in der Form (4.1) geschrieben werden kann

$$\frac{\partial\,\Psi(\vec{r},t)}{\partial t} = -\frac{i}{\hbar}\,\mathcal{H}\,\Psi(\vec{r},t) = \mathcal{L}\,(\Psi)\;. \tag{4.51}$$

Man beachte, dass die Quantenmechanik eine lineare Theorie ist, d.h. $\mathcal{L} = -\frac{i}{\hbar}\,\mathcal{H}$ ist hier ein linearer Operator! Nun betrachten wir die Wellenfunktion zu diskreten Zeiten t_n, wie sie insbesondere durch (1.5) gegeben sind. Eine wesentliche

[6] Hier und im folgenden wird das Skalarprodukt symbolisch als Integral geschrieben; in der konkreten Realisierung ist jedoch eine endliche Summe auszuführen.

Eigenschaft der Quantenmechanik ist die Unitarität der Zeitentwicklung, d.h. die Erhaltung von Wahrscheinlichkeiten. Für die diskretisierte Zeitentwicklung bedeutet dies, dass die Norm der Wellenfunktion erhalten bleiben muß

$$\|\Psi(t_{n+1})\| = \|\Psi(t_n)\| \;, \tag{4.52}$$

und zwar für alle $\Psi(t_n)$. Es reicht also nicht, Stabilität des Verfahrens sicherzustellen, sondern es ist die im allgemeinen stärkere Bedingung (4.52) zu erfüllen. Man beachte, dass es sich hierbei um eine nicht-lineare Bedingung handelt, da in der Quantenmechanik die Wellenfunktion selbst nicht meßbar ist, sondern nur ihr Quadrat (die Wahrscheinlichkeitsdichte).

Zuerst betrachten wir das *explizite Euler-Verfahren* (4.6)

$$\Psi(t_{n+1}) = \Psi(t_n) - \Delta t \, \frac{i}{\hbar} \, \mathcal{H} \, \Psi(t_n) \,. \tag{4.53}$$

Nun setzt man eine beliebige Eigenfunktion (4.47) ein $\Psi(t_n) = \Psi_j$. Dann gilt

$$\Psi(t_{n+1}) = \left(1 - \Delta t \, \frac{i}{\hbar} \, E_j \right) \, \Psi_j \,, \tag{4.54}$$

folglich

$$\|\Psi(t_{n+1})\|^2 = \left(1 + \left(\frac{\Delta t}{\hbar} \, E_j \right)^2 \right) \, \|\Psi_j\|^2 \;. \tag{4.55}$$

Der Vorfaktor auf der rechten Seite ist immer größer als 1, d.h. (4.52) ist immer verletzt. Das explizite Euler-Verfahren ist also auch in der Quantenmechanik kein brauchbares Verfahren.

Das erste stabile Verfahren war das *explizite Lax-Verfahren* (4.19). Aufgrund der räumlichen Mittelung kann es nicht allgemein analysiert werden. Wir betrachten somit ein freies Teilchen

$$\mathcal{H} = -\frac{\hbar^2}{2m} \, \Delta \,. \tag{4.56}$$

Man beachte, dass dies mit (3.6) für die Diskretisierung der zweiten Ableitung in einer Dimension auf

$$\left(\mathcal{H} \, \Psi \right)_r = -\frac{\hbar^2}{2m} \, \frac{\Psi_{r+1} - 2 \, \Psi_r + \Psi_{r-1}}{\Delta x^2} \tag{4.57}$$

führt. Die Eigenfunktionen lauten nun

$$\Psi_r^{(n)} = g(n,k) \, \mathrm{e}^{i \, k \, r \Delta x} \,. \tag{4.58}$$

Das explizite Lax-Verfahren (4.19) führt nun auf

$$\begin{aligned}\Psi_r^{(n+1)} &= \frac{1}{2}\left(\Psi_{r+1}^{(n)} + \Psi_{r-1}^{(n)}\right) - \Delta t\,\frac{i}{\hbar}\,\mathcal{H}\,\Psi_r^{(n)} \\ &= \left(\cos\left(k\Delta x\right) - \frac{\Delta t\, i}{\hbar}\,\frac{\hbar^2}{2m}\,\frac{2\left(\cos\left(k\Delta x\right)-1\right)}{\Delta x^2}\right)\,\Psi_r^{(n)}\,, \qquad (4.59)\end{aligned}$$

wobei in der zweiten Zeile (4.57) eingesetzt wurde. Für die Normen folgt

$$\left\|\Psi^{(n+1)}\right\|^2 = \left(\cos^2\left(k\Delta x\right) + \left(\frac{\Delta t\,\hbar}{m}\,\frac{\left(\cos\left(k\Delta x\right)-1\right)}{\Delta x^2}\right)^2\right)\,\left\|\Psi^{(n)}\right\|^2\,. \qquad (4.60)$$

Der Vorfaktor auf der rechten Seite ist im allgemeinen von 1 verschieden, d.h. (4.52) ist nicht erfüllt. Da die Zeitentwicklung des expliziten Lax-Verfahrens nicht einmal für ein freies Teilchen unitär ist, ist dieses Verfahren in der Quantenmechanik unbrauchbar.

Tatsächlich gibt es explizite Differenzenverfahren, die auch in der Quantenmechanik verwendet werden können. So wird z.B. in Kapitel 18.4 von [10] bzw. Kapitel 16.5 von [11] die Verwendung eines von Visscher vorgeschlagenen expliziten Verfahrens [28] empfohlen. Einerseits hat dieses Verfahren den Vorteil, dass es mit reeller Arithmetik implementiert werden kann, andererseits hat es Nachteile z.B. auf konzeptioneller Seite. Wir wollen daher keine neuen Differenzenverfahren jenseits den bereits bekannten einführen.

Andererseits haben wir das implizite Euler-Verfahren 2. Ordnung noch nicht auf seine Anwendbarkeit in den Quantenmechanik untersucht. Tatsächlich wird dieses implizite Verfahren an verschiedenen Stellen in der Literatur für die Anwendung auf die Zeitentwicklung in der Quantenmechanik empfohlen (vgl. z.B. Kapitel 19.2 von [24], Kapitel 7.5 von [18] bzw. [19] (ebenfalls Kapitel 7.5) und Kapitel 4.5 von [17]). Das *implizite Euler-Verfahren 2. Ordnung* (4.11) führt auf

$$\left(1 + \frac{\Delta t}{2\hbar}\,i\,\mathcal{H}\right)\,\Psi(t_{n+1}) = \left(1 - \frac{\Delta t}{2\hbar}\,i\,\mathcal{H}\right)\,\Psi(t_n)\,. \qquad (4.61)$$

Schreibt man dies als $\Psi(t_{n+1}) = U(\Delta t)\,\Psi(t_n)$, so erkennt man in $U(\Delta t) = \left(\mathbb{1} + \frac{\Delta t}{2\hbar}\,i\,\mathcal{H}\right)^{-1}\left(\mathbb{1} - \frac{\Delta t}{2\hbar}\,i\,\mathcal{H}\right)$ die Cayley-Transformierte von $\mathcal{H}$ wieder. Die durch (4.61) gegebene Zeitentwicklung ist somit unitär. Um dies auf einem elementaren Niveau nachzuvollziehen, betrachtet man wieder Eigenfunktionen $\Psi(t_n) = \Psi_j$. Die Relation (4.61) führt dann mit (4.47) auf

$$\Psi(t_{n+1}) = \frac{1 - \frac{\Delta t}{2\hbar}\,iE_j}{1 + \frac{\Delta t}{2\hbar}\,i\,E_j}\,\Psi(t_n)\,. \qquad (4.62)$$

Der Vorfaktor in (4.62) ist der Quotient zweier zueinander komplex konjugierter Zahlen, hat also den Betrag 1. Es folgt, dass die Vorschrift (4.61) *immer* unitär ist. Glg. (4.61) führt auf ein lineares Gleichungssystem (3.24) mit $A = \mathbb{1} + \frac{\Delta t}{2\hbar}\, i\, \mathcal{H}$. Auch wenn die Diskretisierung des Hamilton-Operators $\mathcal{H}$ hermitesch ist, ist der Operator A aufgrund des Faktors i manifest nicht-hermitesch. Das Gleichungssystem (4.61) ist also mit dem in Abschnitt 4.2.1 diskutierten Bi-Conjugate Gradient Verfahren zu lösen. Ferner kann die Diskretisierung $\mathcal{H}$ zwar häufig reell gewählt werden, jedoch führt der Faktor i auf eine explizit komplexe Matrix. Dieses Verfahren wird somit naheliegender Weise mit komplexer Arithmetik implementiert.

Zusammenfassend läßt sich sagen, dass weder das explizite Euler-Verfahren noch das Lax-Verfahren unitär sind, also für die Zeitentwicklung der Schrödingergleichung (4.46) untauglich sind. Das implizite Euler-Verfahren 2. Ordnung (4.61) kann hingegen auch in der Quantenmechanik für beliebige Δt verwendet werden, ohne dass prinzipielle Probleme auftreten (natürlich hängt die Genauigkeit der genäherten Zeitentwicklung von Δt ab).

4.3.2 Operator-Splitting

Im folgenden soll ein weiteres Verfahren vorgestellt werden, das konzeptionell anders arbeitet und somit ggfs. günstigere Merkmale aufweist als die bisher diskutierten. Dieses Verfahren wird z.B. in Kapitel 16.6 von [11] kurz erwähnt.

Formal können wir die zeitabhängige Schrödingergleichung (4.46) lösen, indem wir den Hamilton-Operator exponentieren:

$$\Psi(\vec{r}, t) = \mathrm{e}^{-i\,\mathcal{H}\,t/\hbar}\, \Psi(\vec{r}, 0)\,. \tag{4.63}$$

Hierbei ist $U(t) = \mathrm{e}^{-i\,\mathcal{H}\,t/\hbar}$ ein unitärer Operator.

Zur (näherungsweise) Berechnung der Exponentialfunktion bedienen wir uns eines Tricks. Um diesen zu erklären, betrachten wir zunächst die Exponentialfunktion einer Summe von zwei Operatoren A und B und entwickeln:

$$\begin{aligned}
\mathrm{e}^{\epsilon\,(A+B)} &= \mathbb{1} + \epsilon\,(A+B) + \frac{\epsilon^2}{2}\,(A+B)^2 + \mathcal{O}(\epsilon^3) \\
&= \mathbb{1} + \epsilon\,(A+B) + \frac{\epsilon^2}{2}\,(A^2 + A\,B + B\,A + B^2) + \mathcal{O}(\epsilon^3) \qquad (4.64) \\
&= \left(\mathbb{1} + \epsilon\,A + \frac{\epsilon^2}{2}\,A^2\right)\left(\mathbb{1} + \epsilon\,B + \frac{\epsilon^2}{2}\,B^2\right) - \frac{\epsilon^2}{2}\,A\,B + \frac{\epsilon^2}{2}\,B\,A + \mathcal{O}(\epsilon^3)\,.
\end{aligned}$$

Somit gilt:

$$\mathrm{e}^{\epsilon\,(A+B)} = \mathrm{e}^{\epsilon\,A}\,\mathrm{e}^{\epsilon\,B} - \frac{\epsilon^2}{2}\,[A,B] + \mathcal{O}(\epsilon^3) = \mathrm{e}^{\epsilon\,A}\,\mathrm{e}^{\epsilon\,B} + \mathcal{O}(\epsilon^2)\,. \tag{4.65}$$

Diese Identität ist dann nützlich, wenn e^A und e^B leichter zu berechnen sind als e^{A+B}.

Diese Überlegungen sollen im folgenden auf den Zeitentwicklungsoperator der Quantenmechanik für ein Intervall Δt angewendet werden:

$$U(\Delta t) = \mathrm{e}^{-i\,\mathcal{H}\,\Delta t/\hbar}\,. \tag{4.66}$$

In diesem Fall liegt die Aufteilung

$$\mathcal{H} = \mathcal{H}_0 + V = \frac{\vec{p}^2}{2\,m} + V(\vec{r}) \tag{4.67}$$

nahe. Die Näherung (4.65) führt dann mit der Identifikation $\epsilon = -i\,\Delta t/\hbar$ auf

$$U(\Delta t) = \mathrm{e}^{-i\,\mathcal{H}_0\,\Delta t/\hbar}\,\mathrm{e}^{-i\,V\,\Delta t/\hbar} + \mathcal{O}\left(\Delta t^2\right) = \widetilde{U}(\Delta t) + \mathcal{O}\left(\Delta t^2\right)\,. \tag{4.68}$$

Man erhählt also folgende Näherung für den Zeitentwicklungsoperator

$$\widetilde{U}(\Delta t) = \mathrm{e}^{-i\,\mathcal{H}_0\,\Delta t/\hbar}\,\mathrm{e}^{-i\,V\,\Delta t/\hbar}\,. \tag{4.69}$$

An diesem Ergebnis lassen sich bereits wesentliche Eigenschaften des resultierenden Verfahrens ablesen:

- $\oplus$ Laut (4.69) ist $\widetilde{U}(\Delta t)$ das Produkt zweier unitärer Operatoren, also manifest unitär. Dies gilt unabhängig von der Wahl von Δt. Man beachte, dass unter den bisher diskutierten Verfahren nur das implizite Euler-Verfahren unitär war.

- $\oplus$ Wie man (4.68) entnimmt, ist die Näherung zweiter Ordnung in Δt. Diese Eigenschaft teilt es mit dem impliziten Euler-Verfahren.

- $\oplus$ Der Fehler wird nach (4.65) durch $\langle\Psi \mid [\mathcal{H}_0, V] \mid \Psi\rangle$ kontrolliert. Im Fall $[\mathcal{H}_0, V] = 0$ ist $\widetilde{U}(\Delta t) = U(\Delta t)$, d.h. die Zeitentwicklung wird exakt beschrieben. Dies gilt insbesondere für ein freies Teilchen ($V = 0$). Man beachte, dass bei allen anderen bisher diskutierten Differenzenverfahren selbst in diesem einfachen Fall Abweichungen zweiter Ordnung in Δt auftreten! Im allgemeinen ist der Vorfaktor des Korrekturterms zweiter Ordnung mit der Näherung (4.69) evtl. kleiner, so dass für gewünschte Genauigkeit ein größeres Δt verwendet werden kann als bei Differenzenverfahren.

⊕ Die kinetische Energie $\mathcal{H}_0$ ist diagonal im Impulsraum. Somit kann in dieser Darstellung erstens $\mathrm{e}^{-i\,\mathcal{H}_0\,\Delta t/\hbar} = \mathrm{e}^{-i\,E(k)\,\Delta t/\hbar}$ leicht berechnet werden.
Ferner beobachten wir zunächst, dass man aus der Diskretisierung (4.57) leicht folgenden Ausdruck für die Energie eines freien Teilchens herleiten kann:

$$E(k) = \frac{\hbar^2}{m}\,\frac{1-\cos\left(k\Delta x\right)}{\Delta x^2}\,. \tag{4.70}$$

Betrachtet man nun $k\Delta x \ll 1$, so gilt $E(k) = \frac{\hbar^2}{2m}\,k^2 + \mathcal{O}\left((k\Delta x)^4\right)$. An dieser Stelle kann man im Operator-Splitting-Verfahren eine weitere Verbesserung einbauen, indem man den *exakten* Kontinuumsausdruck $E(k) = \frac{\hbar^2}{2\,m}\,k^2$ in die Zeitentwicklung einsetzt und so die Ordnung der Ortsraumdiskretisierung gegenüber dem Gitter-Ausdruck zweiter Ordnung (4.70) verbessert (die Ordnung hängt nun von der Gesamtzahl N der räumlichen Stützstellen ab). Bei hinreichend glattem Potential $V(\vec{r})$ kann dann die Zahl der Stützstellen verringert werden. Trotz des im folgenden diskutierten zusätzlichen Rechenaufwands zur Berechnug der Exponentialfunktionen kann das Verfahren bei geeigneten Problemen letzten Endes CPU-Zeit-effizienter sein als konkurrierende Verfahren.

- Das Potential V ist diagonal im Ortsraum, $\mathcal{H}_0$ im Impulsraum, so dass die Exponentialfunktionen in der jeweiligen Darstellung leicht berechnet werden können. Diese beiden Darstellungen der Wellenfunktion $|\,\Psi(t_n)\rangle$ können mit Hilfe einer *Fourier-Transformation* ineinander überführt werden. Es ergibt sich folgendes Verfahren für die Berechnung der Zeitentwicklung:

$$\begin{aligned}
|\,\Psi(t_n)\rangle \quad &\xrightarrow{\text{Ortsraum}} \quad \mathrm{e}^{-i\,V\,\Delta t/\hbar}\;|\,\Psi(t_n)\rangle \\
&\xrightarrow{\text{Fourier}} \quad \mathrm{e}^{-i\,V\,\widehat{\Delta t/\hbar}}\;|\,\widehat{\Psi}(t_n)\rangle \\
&\xrightarrow{\text{Impulsraum}} \quad \mathrm{e}^{-i\,E(k)\,\Delta t/\hbar}\,\mathrm{e}^{-i\,V\,\widehat{\Delta t/\hbar}}\;|\,\widehat{\Psi}(t_n)\rangle \;=\; |\,\widehat{\Psi(t_{n+1})}\rangle \\
&\xrightarrow{\text{inverse Fourier}} \quad |\,\Psi(t_{n+1})\rangle\;.
\end{aligned} \tag{4.71}$$

Eine etwas ausführlichere Diskussion dieses Verfahren findet man z.B. in Kapitel 2 von [4], sowie eine Zusammenfassung in Kapitel VI von [22].

Um (4.71) praktisch einzusetzen, wird ein effizientes Verfahren zur Berechnung der Fourier-Transformation benötigt, d.h. die FFT aus Unterkapitel 2.2. Bei der Anwendung der FFT auf ein physikalisches Problem sind die Stützstellen, die

in Unterkapitel 2.2 mit Abstand eins angenommen wurden, noch umzurechnen. Insbesondere gilt für die k-Punkte auf einem Intervall der Länge L

$$k_s = \frac{2\,\pi}{L}\,s\,. \tag{4.72}$$

Im Impulsraum können die Eigenwerte des Hamiltonoperators eines freien Teilchens (4.56) nun leicht angegeben werden:

$$\mathcal{H}_0\,\hat{\Psi}_s = E_s^{(0)}\,\hat{\Psi}_s\,. \tag{4.73}$$

Allerdings ist bei der Umrechnung des Bereiches $0 \le s < N$ aus der FFT noch zu beachten, dass der Wertebereich symmetrisch um das Minimum bei $k = 0$, d.h. $s = 0$ gewählt werden sollte. Man wählt also

$$E_s^{(0)} = \frac{\hbar^2\,k_s^2}{2\,m} = \begin{cases} \frac{\hbar^2\,(2\,\pi\,s)^2}{2\,m\,L^2} & \text{für } s \le N/2\,, \\ \frac{\hbar^2\,(2\,\pi\,(N-s))^2}{2\,m\,L^2} & \text{für } s \ge N/2\,. \end{cases} \tag{4.74}$$

4.3.3 Doppelmulden-Potential

Als Anwendungs-Beispiel wollen wir im folgenden das Doppelmulden-Potential untersuchen. Die Diskussion folgt Kapitel 2.5 und 2.6 von [30]. Gegeben sei der eindimensionale Hamilton-Operator

$$\mathcal{H} = \frac{p^2}{2\,m} + V(x) \tag{4.75}$$

mit

$$V(x) = m\,\omega^2\,\frac{(x - x_m)^2(x + x_m)^2}{8\,x_m^2}\,. \tag{4.76}$$

Der Vorfaktor ist hierbei so gewählt, dass $\frac{\partial^2}{\partial x^2}V\big|_{\pm x_m} = m\,\omega^2$ ist. In der Nähe von $\pm x_m$ hat man somit näherungsweise jeweils einen harmonischen Oszillator der gewohnten Form

$$V(x) \approx \frac{1}{2}\,m\,\omega^2\,(x \pm x_m)^2 \tag{4.77}$$

für $|x \pm x_m| \ll x_m$. Das Doppelmulden-Potential (4.76) ist in Abb. 4.4 skizziert. Die beiden Minima bei $\pm x_m$ sind durch eine Barriere der Höhe $V(0) = m\,\omega^2\,x_m^2/8$ getrennt. Für $x \to \pm\infty$ geht das Potential in den anharmonischen Oszillator $V(x) \sim x^4$ über.

Bei $\pm x_m$ findet man näherungsweise harmonische Oszillatoren der Frequenz ω. Als erste Näherung kann man bei einer hinreichend hohen Barriere, d.h. hinreichend großem Abstand x_m, die Niveaus dieser beiden harmonischen Oszillatoren

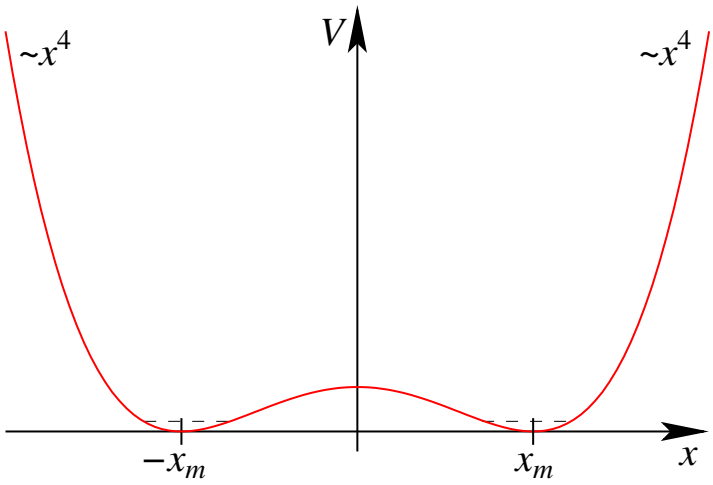


Abbildung 4.4: *Das Doppelmulden-Potential.*

betrachten. Die Grundzustandswellenfunktion eines harmonischen Oszillators hat die Form

$$\Psi_{x_z}(x) = C\,\mathrm{e}^{-\alpha(x-x_z)^2/2}\,. \tag{4.78}$$

Im Prinzip könnte man α und x_z in dieser Wellenfunktion für das Potential (4.76) optimieren. Dies ist jedoch mühselig, so dass wir näherungsweise die Grundzustandswellenfunktionen der harmonischen Oszillatoren (4.77) bei $\pm x_m$ verwenden:

$$x_z = \pm x_m\,, \qquad \alpha = \frac{m\,\omega}{\hbar}\,. \tag{4.79}$$

Demenstprechend finden wir für die Energie näherungsweise die Grundzustandsenergie eines harmonischen Oszillators:

$$E \approx \frac{1}{2}\,\hbar\,\omega\,. \tag{4.80}$$

In einer Simulation kann die Energie der Wellenfunktion (4.78) als Erwartungswert des Hamilton-Operators gemessen werden:

$$E = \langle \mathcal{H} \rangle = \frac{\langle \Psi \,|\, \mathcal{H} \,|\, \Psi \rangle}{\langle \Psi \,|\, \Psi \rangle}\,. \tag{4.81}$$

Wählt man als Anfangswellenfunktion $\Psi(x, t=0) = \Psi_{\pm x_m}(x)$, d.h. setzt man ein Teilchen bei $t = 0$ in eines der beiden Minima, so ist dies keine Eigenfunktion. Klassisch würde ein Teilchen natürlich im Minimum ('Grundzustand') bei

$\pm x_m$ liegen bleiben, quantenmechanisch kann es jedoch in das Nachbarminimum tunneln.

Eine verbesserte Näherung für die Eigenfunktionen ist folgende Kombination der Wellenfunktionen $\Psi_{\pm x_m}$:

$$\Psi_0 \approx \Psi_{x_m} + \Psi_{-x_m}\,, \qquad \Psi_1 \approx \Psi_{x_m} - \Psi_{-x_m}\,. \tag{4.82}$$

Die Berechnung der Energie dieser Wellenfunktionen für das Potential (4.76) ist nicht ganz trivial. Jedoch folgt aus dem Knotensatz, dass der Grundzustand symmetrisch ist und der erste angeregte Zustand antisymmetrisch, d.h. wir identifizieren Ψ_0 als Grundzustand.

Einen Ausgangszustand, bei dem sich das Teilchen links befindet, können wir also als

$$\Psi(x, t=0) = \Psi_{-x_m}(x) \approx \frac{1}{2}\left(\Psi_0(x) - \Psi_1(x)\right) = \varphi(x, t=0) \tag{4.83}$$

schreiben. Hierbei haben wir die Kombination der exakten Eigenfunktionen Ψ_0 und Ψ_1 als φ bezeichnet. Für deren Zeitentwicklung gilt:

$$\begin{aligned} \varphi(x,t) &= \frac{1}{2}\left(\mathrm{e}^{-i\,E_0\,t/\hbar}\,\Psi_0(x) - \mathrm{e}^{-i\,E_1\,t/\hbar}\,\Psi_1(x)\right) \\ &= \mathrm{e}^{-i\,E_0\,t/\hbar}\,\frac{1}{2}\left(\Psi_0(x) - \mathrm{e}^{-i\,(E_1-E_0)\,t/\hbar}\,\Psi_1(x)\right)\,. \end{aligned} \tag{4.84}$$

Man liest ab, dass $\varphi(x,t) \propto \Psi_{-x_m}(x)$ für $t = n\,\tau$, bzw. $\varphi(x,t) \propto \Psi_{x_m}(x)$ für $t = \left(n + \frac{1}{2}\right)\,\tau$ mit ganzzahligen n und

$$\tau = \frac{2\,\pi\,\hbar}{E_1 - E_0}\,. \tag{4.85}$$

Dementsprechend findet man Oszillationen z.B. in dem Erwartungswert des Ortes

$$\langle x\rangle(t) \approx -x_m\,\cos\left(\frac{2\,\pi\,t}{\tau}\right)\,. \tag{4.86}$$

In einer Simulation kann man nun z.B. $\langle x\rangle(t)$ messen, aus (4.86) die Periode τ bestimmen und schließlich aus (4.85) die Tunnel-Aufspaltung $E_1 - E_0$ ablesen. Beimischung von höheren angeregten Wellenfunktionen in $\Psi(x, t=0)$ führt zu höherfrequenten Beiträgen in den Oszillationen von $\langle x\rangle(t)$. Die kleinste Oszillationsfrequenz $\propto 1/\tau$ ist dennoch im allgemeinen gut abzulesen, so dass $E_1 - E_0$ ziemlich genau bestimmt werden kann.

Bisher haben wir gezeigt, wie Energie E und die Tunnel-Aufspaltung $E_1 - E_0$ aus einer zeitabhängigen Simulation bestimmt werden können. Zur Bestimmung der Grundzustandsenergie könnten wir nun im Sinne von (4.83) $E \approx \frac{1}{2}(E_0 + E_1)$ schätzen. Dies ist jedoch nicht besonders genau, so dass eine direkte Bestimmung von E_0 wünschenswert ist. Dies läßt sich beim Operator-Splitting-Verfahren mit einem kleinen Trick bewerkstelligen. Die Idee ist, den Beitrag einer Eigenfunktion Ψ_j mit einem Faktor $\mathrm{e}^{-E_j\,\eta\,t/\hbar}$ zu dämpfen. Normiert man $\Psi(x,t)$ jeweils neu, so bleibt für große t nur Ψ_0 übrig. Bei der Implementierung ersetzt man entsprechend in der Zeitentwicklung

$$\mathrm{e}^{-i\,\mathcal{H}\,t/\hbar} \quad \rightarrow \quad \mathrm{e}^{-i\,\mathcal{H}\,(t-i\,\eta)/\hbar}\,. \tag{4.87}$$

Man verwendet also den gewohnten Algorithmus für die Zeitentwicklung, wählt den Zeitschritt jedoch komplex $\Delta t \in \mathbb{C}$ mit $\Im\mathrm{m}(\Delta t) < 0$. Normiert man weiterhin $\Psi(x,t)$ nach jedem Zeitschritt neu, so konvergiert $\Psi(x,t) \rightarrow \Psi_0(x)$. Somit kann schließlich die Grundzustandsenergie als Erwartungswert des Hamilton-Operators bestimmt werden $\langle\mathcal{H}\rangle(t) \rightarrow E_0$.

5 Zufallsgeneratoren

Wir wollen uns im folgenden mit Monte-Carlo Verfahren beschäftigen. Hierzu werden Zufallszahlen benötigt. Später in diesem Kapitel werden wir zeigen, dass sich allgemeine Verteilungen $P(x)$ von Zufallszahlen x aus Gleichverteilungen herleiten lassen, so dass wir uns zunächst auf Gleichverteilungen in einem Intervall konzentrieren.

Wir suchen also Folgen von „Zufallszahlen“ x_i mit folgenden Eigenschaften:

(i) Die Zufallszahlen sollen auf einem gegebenen Intervall gleichverteilt sein. Für Fließkommazahlen verwendet man meistens eine Gleichverteilung auf dem Intervall $[0, 1]$, bei ganzen Zahlen ein Intervall $[0, n]$ mit einer festen ganzen Zahl n.

(ii) Zwischen den einzelnen x_i sollen keine nachweisbaren Korrelationen bestehen, d.h. sind x_0, ..., x_{i-1} bereits bekannt, soll keine Vorhersage für das Ergebnis der Zahl x_i möglich sein.

Bekannte Verfahren zur Erzeugung von „Zufallszahlen“ sind z.B. Würfel oder Roulette. Bei physikalischen Realisierungen mag man u.a. an (radioaktive) Zerfallsprozesse und thermisches Rauschen in elektronischen Schaltkreisen denken. Nun gilt es allerdings zu bedenken, dass moderne Computer Taktfrequenzen im GHz-Bereich aufweisen, so dass man durchaus bei einem Bedarf von $\geq 10^9$ zufälligen Bits in der Sekunde landet. Solche Raten sind schwer zu realisieren, wenn dabei gleichzeitig die Eigenschaften (i) und (ii) erfüllt sein sollen (d.h. Gleichverteilung und fehlende Korrelationen – beides mit hoher Genauigkeit).

Daher beschreitet man meistens einen anderen Weg und verwendet eine „chaotische“ (also möglichst nicht vorhersagbare), aber deterministische Berechnungsvorschrift. Derart erzeugte Zahlenfolgen sind natürlich nicht mehr zufällig, so dass man auch von „Pseudo-Zufallszahlen“ spricht.

Eine wichtige Referenz für Zufallsgeneratoren ist der Anfang von Kapitel 7 von [24]. Diskussionen von Pseudo-Zufallsgeneratoren findet man z.B. auch in Kapitel 12.6 von [10], in Kapitel 7.9 von [11], Kapitel 2.2.5 von [21] sowie Kapitel 4 von [30].

5.1 Pseudo-Zufallsgeneratoren

Wir wollen nun Beispiele für Pseudo-Zufallszahlsgeneratoren diskutieren. Wie bereits erwähnt, sind diese durch eine deterministische Berechnungsvorschrift charakterisiert, die ggfs. mit einem Gedächtnis arbeitet.

Dieser Zugang hat natürlich verschiedene Nachteile. Zunächst ist jedes x_i prinzipiell vorhersagbar (insbesondere dann, wenn man die Berechnungsvorschrift kennt). Ferner ist der Informationsgehalt der Berechnungsvorschrift zusammen mit dem Gedächtnis klein gegen eine Folge von echten Zufallszahlen mit einer typisch benötigten Länge. Genauer gesagt ist die einzige Möglichkeit, eine Folge echter Zufallszahlen zu reproduzieren das Speichern der kompletten Folge, d.h. aufgrund der Unvorhersagbarkeit der jeweils nächsten Zufallszahl existiert keine Berechnungsvorschrift, die (deutlich) kürzer ist als die Folge der Zufallszahhlen selbst. Hingegen benötigt man typischerweise nicht besonders viel Speicher, um die Berechnungsvorschrift und das Gedächtnis eines Pseudo-Zufallszahlsgenerators zu speichern. Pseudo-Zufallszahlszahlen sind aus diesen Gründen prinzipiell nicht wirklich zufällig, so dass man grundsätzlich immer testen muß, ob ein bestimmter Zufallsgenerator für einen gegebenen Zweck brauchbar ist.

Schließlich erzeugt jeder Pseudo-Zufallsgenerator die gleichen Folgenglieder x_i, wenn er sich im gleichen Zustand befindet. Da die Zahl der möglichen Zustände immer endlich ist, sind alle Pseudo-Zufallsgeneratoren periodisch. Für praktische Anwendungen sollte man darauf achten, dass die Periode lang genug ist, d.h. groß gegen die Anzahl der benötigten Zufallszahlen. Auf keinen Fall sollte man mehr Pseudo-Zufallszahlen verwenden als die Periodenlänge des Generators.

Ein positiver Randeffekt ist andererseits, dass die Ergebnisse reproduzierbar sind. Typischerweise befindet sich ein Pseudo-Zufallsgenerator beim Programmstart jeweils im gleichen Zustand, so dass das Verhalten eines Programms reproduzierbar sein sollte. Ein reproduzierbarer Programmablauf ist z.B. wichtig für die Fehlersuche.

Als erstes Beispiel wollen wir *linear kongruente Zufallsgeneratoren* diskutieren. Diese Generatoren verwenden eine Rekursionsrelation für ganze Zahlen I_j:

$$I_{n+1} = a\, I_n + c \pmod{m} \tag{5.1}$$

mit geeignet gewählten ganzzahligen a, c, m. Das „chaotische“ Verhalten wird in dieser Generatorklasse durch die modulo-Operation erzeugt. Je nach Wahl der

Parameter können linear kongruente Generatoren gut oder schlecht sein. Erfahrung zeigt, dass die Wahl $c = 0$ bei guter Wahl von a und m keine schlechteren Generatoren liefert als $c \neq 0$. Wir werden uns daher auf den Fall $c = 0$ konzentrieren.

Der Rekursionsrelation (5.1) sieht man leicht an, dass die Zahlenfolge periodisch ist, d.h. es existiert eine Periodenlänge k, so dass $I_{n+k} = I_n$ für alle n ist. Da I_n maximal m verschiedene Werte annehmen kann und I_n ferner alle weiteren Folgenglieder eindeutig bestimmt, ist die Periodenlänge eines linear kongruenten Generators maximal m, d.h. es gilt $k \leq m$. m sollte also möglichst groß gewählt werden.

Wie bereits erwähnt, ist die Güte der Zufallsgeneratoren zu testen. Zunächste sollte man die Gleichverteilung der erzeugten Zufallszahlen überprüfen; kritischer sind jedoch meist Korrelationen in der Folge der erzeugten Zufallszahlen.

Die Gleichverteilung und die Paarkorrelationen $\langle x_n\, x_{n-r}\rangle$ (x_n ist das auf $[0,1]$ normierte I_n) kann man einem einfachen geometrischen Test unterwerfen: Man zeichnet Punkte, deren x- bzw. y-Koordindaten durch zwei aufeinanderfolgende Zufallszahlen x_{n-1} und x_n gegeben ist. In diesem Test sollte die Ebene gleichmäßig, d.h. ohne erkennbares Muster, ausgefüllt werden.

Höhere Korrelationen können ebenfalls relevant sein. Die nächste Korrelationsfunktion ist die Tripletkorrelation $\langle x_n\, x_{n-r}\, x_{n-s}\rangle$. Für diese gilt in dem Fall, dass die Folgenglieder unkorreliert sind:

$$\langle x_n\, x_{n-r}\, x_{n-s}\rangle = \langle x_n\rangle\, \langle x_{n-r}\rangle\, \langle x_{n-s}\rangle = \langle x_n\rangle^3 = \left(\frac{1}{2}\right)^3 = \frac{1}{8}\,. \tag{5.2}$$

Man beachte, dass die Annahme von Unkorreliertheit an dieser Stelle insbesondere bedeutet, dass n, $n - r$ sowie $n - s$ paarweise verschieden sind, d.h. (5.2) gilt nur für $r \neq 0$, $s \neq 0$ und $r \neq s$.

Wir wollen nun ein paar Beispiele linear kongruenter Zufallsgeneratoren (5.1) diskutieren:

1. $a = 2^{15} - 1 = 32\ 767$, $m = 2^{16} - 1 = 65\ 535$, $c = 0$. Abb. 5.1 zeigt die von diesem Generator erzeugten Paare (x_{n-1}, x_n). Offensichtlich handelt es sich um einen sehr schlechten Generator. Ein Grund hierfür ist, dass m zu klein gewählt ist.

2. $a = 65\ 539$, $m = 2^{31} = 2\ 147\ 483\ 648$, $c = 0$ (dieses a ist eine Primzahl). Dieser Generator war zeitweise auf IBM-Großrechnern weit verbrei-

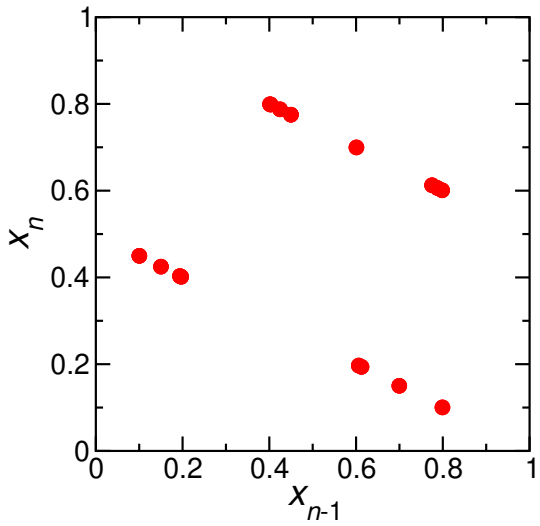


Abbildung 5.1: *Von dem linear kongruenten Zufallsgenerator mit* $a = 2^{15} - 1 = 32\ 767$, $m = 2^{16} - 1 = 65\ 535$, $c = 0$ *erzeugte Paare* (x_{n-1}, x_n).

tet. Tatsächlich diskutieren Binder und Stauffer (zwei Pioniere der Monte-Carlo Methode) diesen Generator in Kapitel 1.1.1 des Buches [2] als Hauptbeispiel. Binder und Stauffer bemerken selbst, dass dies kein besonders guter Generator ist. Der Kommentar in Kapitel 7.1 von [24] zu diesem Generator ist hingegen vernichtend: „... choices of m, a, and c have been botched“ („verpfuscht“). Ein Defizit dieses Generators ist leicht zu sehen: Aufgrund der Mulitplikation mit einer ungeraden Zahl mit anschließender modulo-Bildung mit einer geraden Zahl bleibt das unterste Bit der Folge I_n erhalten, d.h. ein ungerades I_0 führt zu einer Folge, deren Glieder I_n alle ungerade sind, ein gerades I_0 führt hingegen zu einer Folge, deren Glieder I_n ausschließlich gerade sind. Das Hauptproblem dieses Generators sieht man jedoch in Abb. 5.2: Faßt man drei aufeinander folgende Zufallszahlen x_{n-2}, x_{n-1}, x_n als Koordinaten eines Punktes in drei Dimensionen auf, so findet man nur eine sehr geringe Anzahl von Ebenen. Im allgemeinen gilt, dass die k-Tupel von k aufeinander folgenden mit einem linear kongruenten Generator erzeugten Zufallszahlen auf $(k-1)$-dimensionalen Hyperebenen liegen, wobei es höchstens $\sqrt[k]{m}$ solcher Hyperebenen gibt [24]. Im hier vorliegenden Fall $m = 2^{31}$, $k = 3$ wären im Prinzip etwa 1 290 Ebenen möglich – wie man Abb. 5.2 entnimmt, sind es tatsächlich deutlich weniger.

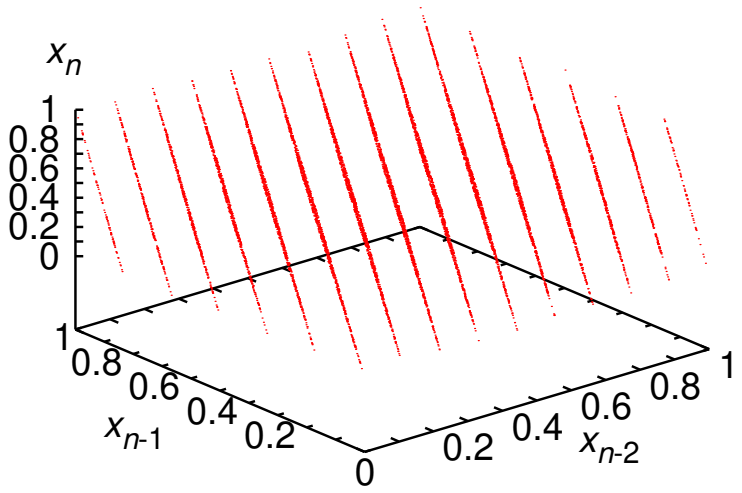


Abbildung 5.2: 10^4 *von dem linear kongruenten Zufallsgenerator mit* $a = 65\ 539$, $m = 2^{31} = 2\ 147\ 483\ 648$, $c = 0$ *erzeugte Triplets* (x_{n-2}, x_{n-1}, x_n).

3. $a = 7^5 = 16\ 807$, $m = 2^{31} - 1 = 2\ 147\ 483\ 647$, $c = 0$. Dies ist der „Minimal Standard“ Generator von [24]. Der Name „Minimal Standard“ soll dabei andeuten, dass dies zwar nicht unbedingt ein guter Generator ist, dass man aber keinen Generator verwenden sollte, der schlechter ist.

Auch ein guter linear kongruenter Generator hat Probleme. Zunächst sind die niedrigen Bits der I_j weniger zufällig als die hohen (ein besonders deutliches Beispiel hierfür war unser zweites Beispiel). Wird nur ein kleinerer Wertebereich benötigt, sollte man also *nie* die unteren, sondern immer nur die oberen Bits verwenden. Insbesondere sollte man auf die Zufallszahlen keine modulo-Operation anwenden. Ferner ist die Periode vergleichsweise kurz, d.h. man erhält typischerweise maximal von der Ordnung 10^8 verschiedene I_j, danach wiederholt sich die Zahlenfolge.

Schieberegister-Zufallsgeneratoren stellen eine zweite verbreitete Generator-Klasse dar. Diese Generatoren verwenden ein Register und folgende Rekursionsrelation

$$I_n = I_{n-p} \hat{} I_{n-q}\,. \tag{5.3}$$

Hierbei haben wir wie bereits in Unterkapitel 2.1 das Symbol $\hat{}$ für ein bitweises exklusives oder verwendet.

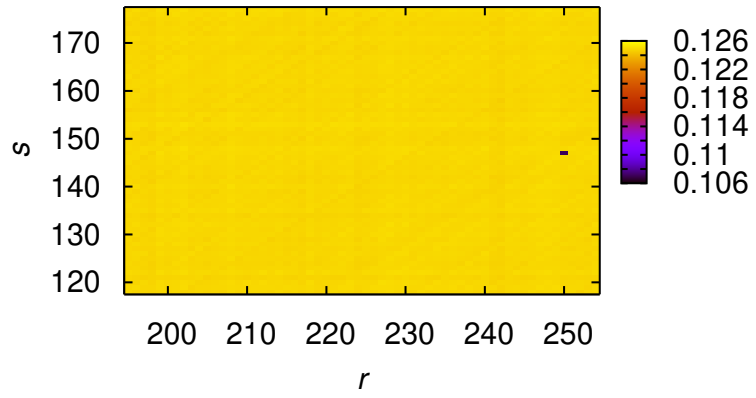


Abbildung 5.3: *Werte der Tripletkorrelation $\langle x_n\, x_{n-r}\, x_{n-s}\rangle$ in dem Schieberegister-Zufallsgenerator R250.*

Schieberegister-Zufallsgeneratoren besitzen verschiedene Vorteile. Zunächst stellt ein größeres Gedächtnis sehr lange Perioden sicher. Ferner sind alle Bits gleich zufällig, wenn die Anfangsbelegung des Registers gut gewählt ist, d.h. wenn die ersten $\max(p, q)$ Zahlen I_n geeignet gewählt werden.

Ein populärer Schieberegister-Generator ist unter dem Namen „R250“ bekannt. R250 ist durch die Parameter $p = 250$ und $q = 147$ definiert. Bei diesem Generator sind die Paarkorrelationen $\langle x_n\, x_{n-1}\rangle$ in Ordnung. Abb. 5.3 zeigt einen Ausschnitt der Tripletkorrelation $\langle x_n\, x_{n-r}\, x_{n-s}\rangle$ für R250. In den meisten Fällen beobachtet man den nach (5.2) erwarteten Wert von 0.125. Allerdings beobachtet man ein deutliches Signal bei $r = p$, $s = q$ (bzw. dem in Abb. 5.3 nicht gezeigten, aber äquivalenten Fall $r = q$, $s = p$): Für R250 findet man einen deutlich erniedrigten Wert $\langle x_n\, x_{n-250}\, x_{n-147}\rangle \approx 0.107$. Aufgrund des Bildungsgesetzes (5.3) ist diese Korrelation bei $r = p$ und $s = q$ nicht überraschend, sie kann aber zu systematischen Abweichungen im Ergebnis einer Simulation führen. Ein besonderes drastisches Beispiel findet sich in [26], hier wurden durch R250 verursachte systematische Fehler im Ergebnis von bis zu 20% beobachtet.

Im allgemeinen reicht auch kein „Durcheinanderwirbeln“ der Zufallszahlen durch den Algorithmus, um ein korrektes Ergebnis sicherzustellen. Dies ist spätestens seit der „Ferrenberg-Landau-Affäre“ bekannt [5]. In dieser Arbeit wurden Simulationen mit einem Algorithmus durchgeführt, der dem Augenschein nach nicht empfindlich auf vereinzelte Korrelationen reagieren sollte. Dennoch wurden

bei verschiedenen zu dieser Zeit beliebten Zufallsgeneratoren statistisch signifikante systematische Fehler im Ergebnis nachgewiesen [5]; auch für R250 wurden bereits hier Probleme berichtet, wobei der Grund für das Versagen des Generators weniger offensichtlich ist als in der späteren Arbeit [26].

Es gibt viele andere Pseudo-Zufallsgeneratoren sowie Methoden um Eigenschaften von Zufallsgeneratoren zu „verbesseren“. Man sollte sich dennoch immer daran erinnern, dass kein Pseudo-Zufallsgenerator perfekt sein kann, da er grundsätzlich immer weniger Information enthält als eine (hinreichend lange) Folge echter Zufallszahlen. Dies gilt auch für den in Kapitel 7.1 von [24] als „perfekt“ angepriesenen Zufallsgenerator `ran2()`. Tatsächlich ist `ran2()` für in der Physik verbreitete hochgenaue Simulationen ungeeignet. Der Grund hierfür liegt in einer Korrelation der Summen von Zufallszahlen. Konkret treten Probleme in solchen Situationen auf, in denen eine kleine Zufallszahl auf viele große Zufallszahlen folgt. Es sei dem Leser überlassen, diese Korrelation von `ran2()` mit einem statistischen Test nachzuweisen und das in [24] ausgelobte Preisgeld von $1000 einzufordern.

Ferner sei daran erinnert, dass Pseudo-Zufallsgeneratoren für gewöhnlich nach jedem Programmstart die gleiche „Zufalls“folge liefern. Will man also durch Wiederholung der Simulation die Statistik verbessern, so ist darauf zu achten, dass der Zufallsgenerator beim erneuten Programmstart geeignet initialisiert wird. Dies kann von Hand geschehen, oder z.B. über die Systemzeit[7]. Darüber hinaus stellt z.B. Linux unter `/dev/random` einen Pool von Zufallszahlen zur Verfügung [20]. Diese Verwaltung von Zufallszahlen mit Hilfe des Betriebssystems ist in erster Linie für kryptographische Zwecke gedacht, sie kann aber natürlich auch zur Initialisierung eines Pseudo-Zufallsgenerators verwendet werden – für eine direkte Verwendung in Monte-Carlo-Simulationen liefert das Betriebssystem Zufallszahlen allerdings nur mit einer deutlich zu niedrigen Geschweindigkeit, so dass wir hier auf Pseudo-Zufallsgeneratoren angewiesen sind.

[7] Bei einer Abfrage der Systemzeit ist natürlich darauf zu achten, dass evtl. mehrere Abfragen deutlich länger auseinander liegen als die Zeitauflösung der Uhr.

5.2 Andere Verteilungen

Eine Methode, andere Verteilungen zu erzeugen, basiert auf folgender Überlegung: Eine Gleichverteilung $P(x)$ auf $[0,1]$ ist charakterisiert durch

$$\int_0^1 \mathrm{d}x\, P(x)\, f(x) = \int_0^1 \mathrm{d}x\, f(x) \tag{5.4}$$

für jede beliebige Funktion $f(x)$. Nun wenden wir eine Substitution $y = y(x)$ auf (5.4)

$$\int_0^1 \mathrm{d}x\, P(x)\, f(x) \int_{y(0)}^{y(1)} \mathrm{d}y\, \frac{\mathrm{d}x}{\mathrm{d}y}\, P(y)\, f(y)\,. \tag{5.5}$$

Nun kann man

$$\tilde{P}(y) = \frac{\mathrm{d}x}{\mathrm{d}y}\, P(y) \tag{5.6}$$

als neue Verteilungsfunktion auffassen (vorausgesetzt, dass $\tilde{P}(y) \geq 0$ gilt; die Normierung $\int \mathrm{d}y\, \tilde{P}(y) = 1$ erbt $\tilde{P}(y)$ von $P(x)$). Wir können also durch Substitution $y = y(x)$ aus einer Gleichverteilung $P(x)$ eine andere Verteilung $\tilde{P}(y)$ erzeugen.

Ein wichtiges Beispiel für diese Konstruktion ist die Exponentialverteilung. Hier erzeugt man aus einer in $]0,1]$ gleichverteilten Zufallsvariablen x eine neue mittels

$$y(x) = -\ln x\,, \tag{5.7}$$

wobei $y \in [0,\infty[$ gilt. Die Umkehrfunktion von (5.7) ist $x(y) = \exp(-x)$. Die neue Zufallsvariable y ist also nach (5.6) exponentiell verteilt, denn

$$\tilde{P}(y) = \left|\frac{\mathrm{d}x}{\mathrm{d}y}\right| = \mathrm{e}^{-y}\,. \tag{5.8}$$

Der Betrag entsteht hier dadurch, dass wir die Integrationsgrenzen $y(0) = \infty$ und $y(1) = 0$ vertauscht haben. Diese Methode, aus einer gleichverteilten Zufallsvariablen x mit Hilfe der Transformation (5.7) eine exponentiell verteilte Zufallsvariable y zu erzeugen wird z.B. bei der Simulation von (radioaktiven) Zerfallsereignissen verwendet.

Mit der einfachen Transformationsmethode (5.6) erhält man allerdings nur solche Zufallsverteilungen $\tilde{P}(y)$, deren Umkehrfunktion existiert, die also insbesondere monoton sind. Eine Erweiterung ist die Verallgemeinerung der Transformation auf höhere Dimensionen, d.h. mehrere Variable. Ein wichtiges Beispiel

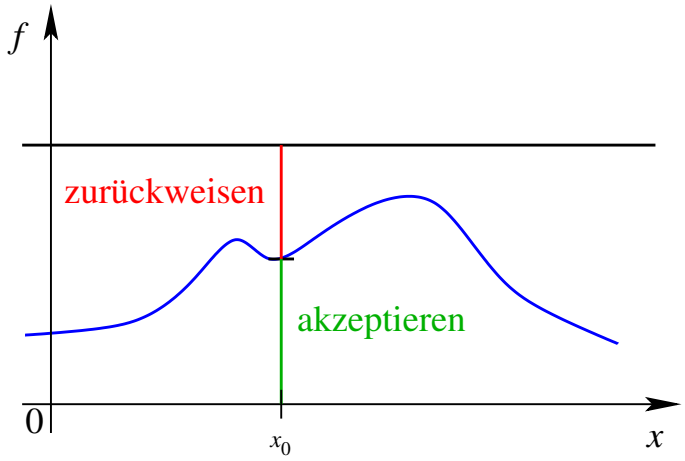


Abbildung 5.4: *Illustration der Verwerfungs- („Rejection“-) Methode zur Erzeugung einer beliebigen Zufallsverteilung $f(x)$.*

ist die „Box-Muller“-Methode zur Erzeugung einer Gaußverteilung. Man beginnt mit *zwei* Zufallsvariablen x_1, x_2, die beide in dem Intervall $[0, 1]$ gleichverteilt sind. Über die Transformation

$$y_1 = \sqrt{-2\,\ln x_1}\,\cos(2\pi\,x_2)\,, \qquad y_2 = \sqrt{-2\,\ln x_1}\,\sin(2\pi\,x_2) \tag{5.9}$$

führt man dann zwei neue Zufallszahlen y_1 und y_2 ein. Eine kurze Überlegung zeigt (siehe z.B. Kapitel 7.2 von [24]), dass y_1 und y_2 Gaußverteilt sind:

$$P(y_1) = P(y_2) = P(y) = \frac{1}{\sqrt{2\pi}}\,\mathrm{e}^{-y^2/2}\,. \tag{5.10}$$

Wir wollen diese Aussage hier nicht weiter begründen, sondern in den Übungen lediglich nachprüfen, dass mit (5.9) erzeugte Zufallszahlen tatsächlich der Gaußverteilung (5.10) genügen[8].

Auch mit einer mehrdimensionalen Substitution bekommt man allerdings nicht beliebige Verteilungen. Dieses Ziel erreicht man mit der Verwerfungs- („Rejection“-) Methode, die in Abb. 5.4 skizziert ist. Hier erzeugt man zunächst eine

[8] In [24] wird folgende „Verbesserung“ bei der Implementierung vorgeschlagen: Zur Vermeidung der trigonometrischen Funktionen ziehe man zwei Zufallszahlen v_1, v_2, so dass die Punkte (v_1, v_2) uniform im Einheitskreis verteilt sind (d.h. man zieht v_1 und v_2 uniform in $[-1, 1]$ und behält nur die Paare, die $0 < v_1^2 + v_2^2 < 1$ erfüllen). Hierbei ist der Aufwand zur Erzeugungen zusätzlicher Zufallszahlen gegen die Berechnung von sin und cos abzuwägen. Auf einem modernen Rechner und mit einem guten Zufallsgenerator wie z.B. `random()` in C kann die „verbesserte“ Auswertung aus [24] tatsächlich geringfügig langsamer sein als die direkte Auswertung von (5.9).

Zufallszahl x_0 und anschließend eine zweite Zufallszahl y. Die erste Zufallszahl x_0 wird nur dann akzeptiert, wenn $y < f(x_0)$ für die gewünschte Verteilung $f(x)$ gilt, andernfalls wird ein neues x_0 gezogen. Die Rejection-Methode wird vorzugsweise mit einer Transformation kombiniert. Auf diese Weise kann man dann z.B. die Gamma-, Poisson- und die Binomial-Verteilung erzeugen (siehe Kapitel 7.3 von [24]).

5.3 Monte-Carlo-Integration

Wir wollen nun eine erste Anwendung für unsere Zufallsgeneratoren kennen lernen, nämlich die Berechnung von Integralen. Es mag auf den ersten Blick ein wenig ungewohnt erscheinen, Integrale mit Zufallszahlen auszuwerten. Dennoch handelt es sich bei Integration um eine der ältesten Anwendungen für Monte-Carlo (MC) Verfahren (vgl. z.B. Kapitel 5 von [13] – die erste Fassung dieses Buchs wurde 1964 publiziert). Komplementäre Erklärungen zu der folgenden Diskussion findet man z.B. in den Kapiteln 7.6–7.8 von [24], Kapitel 11 von [10] bzw. [11], Kapitel 3.2 von [21], Kapitel 8.1 von [18] bzw. [19], sowie Kapitel 6 von [30].

Zur Motivation betrachten wir die Näherung eines Integrals über die Riemannsche Summe

$$I = \int \mathrm{d}V\, f \approx \sum_{\vec{x}_i \in \text{Gitter}} f(\vec{x}_i)\, \Delta V = \frac{V}{N} \sum_{i=1}^{N} f(\vec{x}_i) = V \langle f \rangle \tag{5.11}$$

mit dem Erwartungswert

$$\langle f \rangle = \frac{1}{N} \sum_{i=1}^{N} f(\vec{x}_i)\,, \tag{5.12}$$

wobei die Summe über die N Punkte $\vec{x}_i$ des Gitters auszuführen ist.

Man kann nun die Identität zwischen linker und rechter Seite von (5.11) beibehalten, wobei man allerdings im Erwartungswert (5.12) die reguläre Wahl der $\vec{x}_i$ durch eine zufällige ersetzt. Dies führt auf folgende *einfache Variante* der MC-Integration: Das Integral wird geschätzt durch

$$I = \int \mathrm{d}V\, f \approx V \langle f \rangle = V \frac{1}{N} \sum_{i=1}^{N} f(\vec{x}_i)\,, \tag{5.13}$$

wobei die Punkte $\vec{x}_i$ zufällig und gleichverteilt in V gewählt werden. Der Fehler von (5.13) wird abgeschätzt durch (zur Begründung vergleiche das folgende

Unterkapitel 5.3.1)

$$\delta I = V \sqrt{\frac{\langle f^2 \rangle - \langle f \rangle^2}{N}} \,. \tag{5.14}$$

Mit diesem Ergebnis können wir nun diskutieren, wann eine MC-Integration den bekannteren Gitterverfahren überlegen ist.

Betrachten wir zunächst ein *Gitter-Verfahren* $n$ter Ordnung in d Dimensionen. Haben wir insgesamt N Stützstellen zur Verfügung, so erzwingt eine reguläre Anordnung in d Dimensionen einen mittleren Abstand

$$\Delta x \propto N^{-\frac{1}{d}} \,. \tag{5.15}$$

Für den Fehler des Integrals I folgt nach Definition und mit der Überlegung (5.15)

$$\delta I \propto \Delta x^{n+1} \propto N^{-\frac{n+1}{d}} \,. \tag{5.16}$$

Andererseits gilt für den Fehler des *einfachen Monte-Carlo-Verfahrens* mit N Stützstellen gemäß (5.14)

$$\delta I \propto N^{-\frac{1}{2}} \,. \tag{5.17}$$

Man beachte, daß der Fehler in diesem Fall unabhängig von der Dimension d des Integrationsgebiets ist.

Durch Vergleich von (5.16) und (5.17) findet man, dass der MC-Fehler schneller abfällt als der des Gitter-Verfahrens, wenn

$$\frac{1}{2} > \frac{n+1}{d} \qquad \Leftrightarrow \qquad d > 2\,(n+1) \tag{5.18}$$

ist. Die MC-Integration ist also vor allem in hohen Dimensionen effizienter. Im Beispiel des Simpson-Verfahrens (siehe z.B. Kapitel 4.1 von [24]) ist die Ordnung $n = 4$, so dass gemäß der Ungleichung (5.18) die MC-Integration für $d > 10$ besser ist.

Neben der asymptotischen Betrachtung gilt zu beachten, dass in hohen Dimensionen bei einem Gitterverfahren nur sehr wenige Gitterpunkte je Richtung zur Verfügung stellen. Betrachten wir z.B. $d = 10$ und fordern, dass insgesamt höchstens $N \leq 10^9$ Stützstellen verwendet werden sollen, so sind maximal 7 Gitterpunkte je Richtung realisierbar. Bei einem enstpechend großen Δx ist man normalerweise nicht im asymptotischen Bereich und das Gitterverfahren verhält sich schlechter als aufgrund der asymptotischen Überlegung (5.16) zu erwarten

wäre. Deswegen sind MC-Verfahren oft auch in niedrigeren Dimensionen überlegen als die Abschätzung (5.18) nahe legt.

Zusammenfassend ergeben sich als Anwendungsgebiete für die MC-Integration zunächst hohe Dimensionen. Auch komplizierte Integrale in nicht besonders hohen Dimensionen (z.B. $d = 3$) werden gerne mit MC-Verfahren ausgewertet, da diese Verfahren z.B. unempfindlich gegenüber Singularitäten des Integranden sind (vorausgesetzt natürlich, die Singularitäten sind nicht zu böse, d.h. insbesondere integrierbar).

Es sei noch eine Bemerkung ergänzt, nämlich dass die MC-Integration für ein konstantes $f(\vec{x})$ exakt ist. Formal folgt dies z.B. aus der für konstantes f gültigen Identität $\langle f^2\rangle = \langle f\rangle^2$, die mit (5.14) auf $\delta I = 0$ führt. Diese Beobachtung kann zur Fehlerreduktion verwendet werden. Kann z.B. eine Näherung an f analytisch integriert werden, so kann ein verbleibender Korrekturterm mit einem deutlich kleineren MC-Fehler berechnet werden. Eine andere Methode, mit Hilfe dieser Beobachtung den Fehler zu reduzieren, werden wir in Unterkapitel 5.3.2 kennen lernen.

5.3.1 Standardabweichung des Mittelwerts

Die Schätzung des Fehlers ist bei jedem MC-Verfahren wesentlich, und soll deswegen hier motiviert werden. Die Diskussion folgt Anhang 11B von [10] bzw. [11].

Wir gehen aus von der Schätzung eines Mittelwerts

$$\langle X\rangle = \frac{1}{N}\sum_{i=1}^{N} X_i \tag{5.19}$$

mit Hilfe von N unkorrelierten Messungen X_i. Zur Schätzung des Fehlers ist es allgemein hilfreich, $N = m\,n$ in m Sätze zu je n Ereignisse aufzuteilen, wobei wir für die folgende Diskussion $n \gg 1$ annehmen. $X_{\alpha,i}$ bezeichne nun das Ereignis $1 \le i \le n$ in Satz $1 \le \alpha \le m$.

Führen wir den Mittelwert über Satz α ein

$$\langle X\rangle_\alpha = \frac{1}{n}\sum_{i=1}^{n} X_{\alpha,i}\,, \tag{5.20}$$

so gilt

$$\langle X\rangle = \frac{1}{m}\sum_{\alpha=1}^{m}\langle X\rangle_\alpha = \frac{1}{m\,n}\sum_{\alpha=1}^{m}\sum_{i=1}^{n} X_{\alpha,i}\,. \tag{5.21}$$

Ferner können wir die Abweichungen der Mittelwerte und der Einzelmessungen einführen

$$e_\alpha = \langle X \rangle_\alpha - \langle X \rangle \tag{5.22}$$

$$d_{\alpha,i} = X_{\alpha,i} - \langle X \rangle \,. \tag{5.23}$$

Für diese gilt

$$e_\alpha \overset{(5.20)}{=} \frac{1}{n} \sum_{i=1}^{n} \left(X_{\alpha,i} - \langle X \rangle \right) \overset{(5.23)}{=} \frac{1}{n} \sum_{i=1}^{n} d_{\alpha,i} \,. \tag{5.24}$$

Mit diesen Definitionen berechnen wir die *Varianzen*. Zunächst gilt für die Einzelmessungen

$$\sigma^2 = \frac{1}{m\,n} \sum_{\alpha=1}^{m} \sum_{i=1}^{n} d_{\alpha,i}^2 \,. \tag{5.25}$$

Für die Varianz der Mittelwerte gilt hingegen

$$\begin{aligned} \sigma_\mathrm{m}^2 &= \frac{1}{m} \sum_{\alpha=1}^{m} e_\alpha^2 \overset{(5.24)}{=} \frac{1}{m} \sum_{\alpha=1}^{m} \sum_{i=1}^{n} \sum_{j=1}^{n} \left(\frac{1}{n}\, d_{\alpha,i} \right) \left(\frac{1}{n}\, d_{\alpha,j} \right) \\ &= \frac{1}{m\,n^2} \sum_{\alpha=1}^{m} \sum_{i=1}^{n} d_{\alpha,i}^2 + \frac{1}{m} \sum_{\alpha=1}^{m} \sum_{i \neq j} \left(\frac{1}{n}\, d_{\alpha,i} \right) \left(\frac{1}{n}\, d_{\alpha,j} \right) . \end{aligned} \tag{5.26}$$

In dem ersten Summanden erkennen wir bis auf einen Faktor n die Varianz der Einzelmessungen (5.25) wieder. Der zweite Term verschwindet für große n mit folgendem Argument: Die $d_{\alpha,i}$ haben Mittelwert 0, ferner sind $d_{\alpha,i}$ und $d_{\alpha,j}$ für $i \neq j$ unkorreliert. Die einzelnen Summanden haben somit mit gleicher Wahrscheinlichkeit positives und negatives Vorzeichen, so dass die Summe für $n \to \infty$ verschwindet (und zwar schneller als der erste Term in (5.26).

Aus diesen Überlegungen folgt

$$\sigma_\mathrm{m}^2 \overset{n \gg 1}{=} \frac{\sigma^2}{n} \,. \tag{5.27}$$

Nun kann man die Varianz der Mittelwerte σ_m^2 als Schätzung der Abweichung einer Mittelung über n Messungen vom exakten Ergebnis interpretieren. Damit erhalten wir den für jede MC-Methode relevanten Schätzer für den Fehler

$$\sigma_\mathrm{m} = \frac{1}{\sqrt{n}}\, \sigma = \frac{1}{\sqrt{n}} \sqrt{\langle X^2 \rangle - \langle X \rangle^2} \,. \tag{5.28}$$

Dieses Ergebnis gilt unter den Voraussetzung dass $n \gg 1$ und dass die Einzelmessungen X_i statistisch unabhängig (unkorreliert) sind.

Man beachte, dass auf der rechten Seite von (5.28) keine explizite Abhängigkeit von der Unterteilung in die m Sätze auftritt. Man kann daher zu $m = 1$ fortsetzen, d.h. $n = N$ einsetzen. Dies ist die Standardvorgehensweise um den Fehler eines Mittelwerts unkorrelierter Daten zu schätzen. Verwendet man (5.28) mit $n = N$ für das Integral (5.13), so erhält man (5.14).

Tatsächlich geht die Aussage etwas weiter. In dem Fall, dass die Meßwerte unkorreliert und Gauß-verteilt sind (dies gilt z.B. nach dem zentralen Grenzwertsatz nach Bildung hinreichend großer Sätze) gilt: Die Abweichung ist in ca. 68% der Fälle kleiner als die Schätzung $\sigma/\sqrt{N}$ für den Fehler, in etwa 95% der Fälle ist sie kleiner als $2\,\sigma/\sqrt{N}$ und nur in weniger als einem aus 10^4 Fällen sollte sie größer als $4\,\sigma/\sqrt{N}$ sein (die Zahlenwerte ergeben sich durch die entsprechenden Integrale der Gauß-Verteilung).

5.3.2 Verbesserte Monte-Carlo-Integration

Wir haben gesehen, dass der Fehler einer MC-Integration einer Funktion f durch die Varianz $\sigma^2(f) = \langle f^2\rangle - \langle f\rangle^2$ kontrolliert wird. Jede Verbesserung einer MC-Integration zielt daher auf eine Minimierung von $\sigma^2(f)$. Dies kann mit verschiedenen Strategien erzielt werden. Wir wollen hier kurz eine als „Stratified Sampling“ bekannte Strategie vorstellen (für weiterführende Bemerkungen vergleiche Kapitel 7.8 von [24] und Kapitel 5 von [13]).

Man teilt zunächst das Integrationsvolumen V in zwei *gleich große* Teilvolumina A und B auf. Die Mittelwerte sind

$$\langle f\rangle = \frac{1}{V}\int f\,, \qquad \langle f\rangle_{A/B} = \frac{2}{V}\int\limits_{A/B} f\,, \tag{5.29}$$

und die Varianzen lauten

$$\sigma^2(f) = \langle f^2\rangle - \langle f\rangle^2\,, \qquad \sigma^2_{A/B}(f) = \langle f^2\rangle_{A/B} - \langle f\rangle^2_{A/B}\,. \tag{5.30}$$

Durch Einsetzen und Umformen findet man

$$\begin{aligned}
\sigma^2(f) &= \langle f^2\rangle - \langle f\rangle^2 = \frac{1}{2}\left(\langle f^2\rangle_A + \langle f^2\rangle_B\right) - \frac{1}{4}\left(\langle f\rangle_A + \langle f\rangle_B\right)^2 \\
&= \frac{1}{2}\left(\sigma^2_A(f) + \sigma^2_B(f)\right) + \frac{1}{2}\langle f\rangle^2_A + \frac{1}{2}\langle f\rangle^2_B - \frac{1}{4}\left(\langle f\rangle_A + \langle f\rangle_B\right)^2 \\
&= \frac{1}{2}\left(\sigma^2_A(f) + \sigma^2_B(f)\right) + \frac{1}{4}\left(\langle f\rangle_A - \langle f\rangle_B\right)^2\,.
\end{aligned} \tag{5.31}$$

Die Varianz bei der Mittelung über das Gesamtvolumen V unterscheidet sich also von der Summe der Varianzen über die Teilvolumina A und B lediglich durch einen positiven Korrekturterm!

Man kann nun zur Berechnung des Integrals zwei Strategieen betrachten. Erstens kann man N zufällige $\vec{x} \in V$ zum Schätzen des Integrals wählen. Zweitens kann man je $N/2$ zufällige $\vec{x} \in A$ und $\vec{x} \in B$ verwenden. Das Ergebnis (5.31) zeigt, dass die Summe der Varianzen der zweiten Strategie nie größer ist als die Varianz der ersten Methode.

Es lohnt sich also im Zweifelsfall, das Integrationsgebiet in Teilgebiete aufzuteilen. Qualitativ ist dies auch einsichtig: Nach der Aufteilung wird f im allgemeinen in den Teilgebieten A und B jeweils weniger stark variieren als im Gesamtgebiet V, so dass man näher an dem idealen Fall eines konstanten f liegt, für das die MC-Integration exakt wird.

Die Unterteilung in Teilvolumina kann weiter geführt werden, solange in jedem Teilvolumen hinreichend viele Meßpunkte für eine sinnvolle Statistik verbleiben. Ferner kann man die Zahl der Meßpunkte N_U in einem Teilvolumen U abhängig von U wählen. Man kann zeigen, dass die *optimale Wahl*

$$N_U \propto \sigma_U(f) \tag{5.32}$$

ist. Dies bedeutet insbesondere, dass man mehr Meßpunkte in den Bereichen verwendet, in denen sich f sich am stärksten ändert. In Bereichen stark variierenden f wird der Fehler somit durch besonders viele Meßpunkte reduziert, in Bereichen mit schwachen Variationen von f führt hingegen auch eine geringere Anzahl von Meßpunkten zu einem hinreichend kleinen Fehler.

6 Importance Sampling: Metropolis-Algorithmus

Im vorangegangenen Kapitel haben wir eine Anwendung für Monte-Carlo (MC) Methoden kennengelernt, nämlich die numerische Integration. Im allgemeinen ist es offensichtlich günstig bevorzugt Zufallspunkte $\vec{x}_i$ in „wichtigen“ Gebieten zu erzeugen. Bei der Integration einer Funktion $f(\vec{x})$ ist z.B. eine Wahl der Wahrscheinlichkeit $p(\vec{x}_i) \propto f(\vec{x}_i)$ wünschenswert. Solche Verfahren sind unter dem Oberbegriff „Importance Sampling“ bekannt.

Wir wollen hier eine konkrete Variante von Importance Sampling vorstellen, die von Metropolis und Koautoren 1953 vorgeschlagen wurde [23] und daher als „Metropolis-Algorithmus“ bezeichnet wird. Wir wollen den Algorithmus zunächst allgemein vorstellen und anschließend auf das Ising-Modell anwenden. Für ergänzende Bemerkungen zu der folgenden allgemeinen Diskussion sei insbesondere auf Kapitel 11.8 von [10], Kapitel 11.7 von [11], Kapitel 8.3 von [18] bzw. [19], Kapitel 2.2.4 von [21], Kapitel 8.4 und 8.5 von [30] sowie Kapitel 9 von [13], verwiesen.

Das Ziel ist recht allgemein die Berechnung von Erwartungswerten

$$\langle f \rangle = \frac{\int \mathrm{d}^d x \, p(\vec{x}) \, f(\vec{x})}{\int \mathrm{d}^d x \, p(\vec{x})} \tag{6.1}$$

für eine beliebige Funktion f und eine gegebene „Wahrscheinlichkeitsdichte“ $p(\vec{x}) \geq 0$, deren Normierung $\int \mathrm{d}^d x \, p(\vec{x})$ allerdings nicht bekannt sein muß.

Man bedient sich nun einer Hilfskonstruktion. Und zwar wird eine Folge von Realisierungen $\vec{x}_i$ über einen Zufallsweg

$$\vec{x}_i \to \vec{x}_{i+1} \to \vec{x}_{i+2} \to \dots \tag{6.2}$$

konstruiert. Der Übergang wird dabei in jedem Schritt durch eine Übergangswahrscheinlichkeit $T(\vec{x}_i \to \vec{x}_j)$ zwischen $\vec{x}_i$ und $\vec{x}_j$ kontrolliert, die explizit vom aktuellen Zustand $\vec{x}_i$ abhängt, aber nicht von der Vorgeschichte. Ein Zufallsprozeß mit den genannten Eigenschaften wird *Markov-Prozeß* genannt.

Wir bezeichnen nun die Wahrscheinlichkeitsverteilung dafür, im $i$ten Schritt den Zustand $\vec{x}_i$ anzutreffen mit $P(\vec{x}_i, i)$. Ein *stationärer Zustand* wird nun angenommen, wenn diese Wahrscheinlichkeitsdichte unabhängig vom Schritt i wird, d.h.

$$P(\vec{x}, i+1) = P(\vec{x}, i) =: P(\vec{x}) \,. \tag{6.3}$$

Der Markov-Prozeß (6.2) sollte so konstruiert werden, dass ein solcher stationärer Zustand eindeutig ist. Dann wird er typischerweise zumindest näherungsweise nach einer hinreichend großen Anzahl von Schritten $i \gg 1$ angenommen, d.h. $P(\vec{x}, i \gg 1) \approx P(\vec{x})$.

Den Zusammenhang zu unserem ursprünglichen Problem, d.h. dem vorgegebenen $p(\vec{x})$ stellt nun eine Bedingung an die Übergangswahrscheinlichkeiten $T(\vec{x}_i \to \vec{x}_j)$ her, die als *detailliertes Gleichgewicht* bekannt ist

$$p(\vec{x}_i)\, T(\vec{x}_i \to \vec{x}_j) = p(\vec{x}_j)\, T(\vec{x}_j \to \vec{x}_i)\,. \tag{6.4}$$

Für nicht allzu pathologische Wahlen[9] von $T(\vec{x}_i \to \vec{x}_j)$ stellt (6.4) sicher, dass der stationäre Zustand der Markov-Kette die gesucht Verteilung $p(\vec{x})$ bis auf eine Normierungskonstante C reproduziert

$$P(\vec{x}) = C\, p(\vec{x})\,. \tag{6.5}$$

Zur Begründung dieses Ergebnisses betrachten wir die Entwicklung $P(\vec{x}, i)$ über einem Schritt:

$$\begin{aligned} P(\vec{x}, i+1) &= P(\vec{x}, i) + \sum_{\vec{y} \neq \vec{x}} \Big(T(\vec{y} \to \vec{x})\, P(\vec{y}, i) - T(\vec{x} \to \vec{y})\, P(\vec{x}, i) \Big) \\ &= P(\vec{x}, i) + \sum_{\vec{y}} \left(T(\vec{y} \to \vec{x})\, P(\vec{y}, i) - T(\vec{x} \to \vec{y})\, P(\vec{x}, i) \right)\,. \end{aligned} \tag{6.6}$$

Der erste Term bezeichnet hierbei den Gewinn von $\vec{x}$ durch den Übergang aus einem anderen Zustand $\vec{y}$ nach $\vec{x}$ und der zweite Term den Verlust an der Stelle $\vec{x}$ durch den Übergang in einen anderen Zustand $\vec{y}$. Offensichtlich macht es keinen Unterschied, ob der Fall $\vec{x} = \vec{y}$ in der Summe mitgenommen oder weggelassen wird.

Für den stationären Zustand (6.3) folgt

$$\begin{aligned} 0 = P(\vec{x}, i+1) - P(\vec{x}, i) &\overset{(6.6)}{=} \sum_{\vec{y}} \left(T(\vec{y} \to \vec{x})\, P(\vec{y}) - T(\vec{x} \to \vec{y})\, P(\vec{x}) \right) \\ &= \sum_{\vec{y}} P(\vec{y})\, T(\vec{x} \to \vec{y}) \left(\frac{T(\vec{y} \to \vec{x})}{T(\vec{x} \to \vec{y})} - \frac{P(\vec{x})}{P(\vec{y})} \right) \\ &\overset{(6.4)}{=} \sum_{\vec{y}} P(\vec{y})\, T(\vec{x} \to \vec{y}) \left(\frac{p(\vec{x})}{p(\vec{y})} - \frac{P(\vec{x})}{P(\vec{y})} \right)\,. \end{aligned} \tag{6.7}$$

[9] Ein triviale Lösung von (6.4) ist z.B. $T(\vec{x}_i \to \vec{x}_j) = \delta_{\vec{x}_i, \vec{x}_j}$. Der Markov-Prozeß (6.2) bleibt dann an einem einmal angenommenen Ort $\vec{x}$ stecken (d.h. $\vec{x}_i = \vec{x}$ für alle i) und enthält offensichtlich keine Information über $p(\vec{x})$.

Analog kann man ferner n Schritte im Markov-Prozeß betrachten. Für die Übergangswahrscheinlichkeit $T_n(\vec{x}_i \to \vec{x}_{i+n})$ gilt

$$T_n(\vec{x}_i \to \vec{x}_{i+n}) = \sum_{\vec{x}_{i+1},\ldots,\vec{x}_{i+n-1}} T(\vec{x}_i \to \vec{x}_{i+1}) \cdots T(\vec{x}_{i+n-1} \to \vec{x}_{i+n}) \,. \tag{6.8}$$

Man überzeugt sich leicht, dass auch T_n das detaillierte Gleichgewicht (6.4) erfüllt, wenn bereits T diese Bedingung erfüllt. Analog zu (6.7) folgt dann

$$0 = P(\vec{x}, i+n) - P(\vec{x}, i) = \sum_{\vec{y}} P(\vec{y})\, T_n(\vec{x} \to \vec{y}) \left(\frac{p(\vec{x})}{p(\vec{y})} - \frac{P(\vec{x})}{P(\vec{y})} \right) . \tag{6.9}$$

Sind die Übergangswahrscheinlichkeiten $T_n(\vec{x} \to \vec{y})$ hinreichend verschieden, so kann man deren Koeffizienten in (6.9) vergleichen, womit

$$0 = P(\vec{y}) \left(\frac{p(\vec{x})}{p(\vec{y})} - \frac{P(\vec{x})}{P(\vec{y})} \right) \tag{6.10}$$

und schließlich (6.5) folgt. An dieser Stelle wird normalerweise *Ergodizität* des Markov-Prozeßes gefordert, d.h. dass für alle Anfangs- bzw. Endzustände $\vec{x}$ bzw. $\vec{y}$ ein n existiert, so dass $T_n(\vec{x} \to \vec{y}) \neq 0$ ist. Ist der Markov-Prozeß nämlich nicht ergodisch, d.h. existieren disjunkte Untermengen, so gelangt man offensichtlich je nach Startbedingung $\vec{x}_0$ zu unterschiedlichen stationären Zuständen. Die gesuchte Identität (6.5) gilt in einem solchen nicht-ergodischen Fall allenfalls lokal, aber nicht global.

Die *Metropolis-Wahl* für die Übergangswahrscheinlichkeiten ist zunächst eine Faktorisierung in eine Vorschlags- und eine Akzeptanzwahrscheinlichkeit

$$T(\vec{x}_i \to \vec{x}_j) = \pi(\vec{x}_i, \vec{x}_j)\, W(\vec{x}_i \to \vec{x}_j) \,. \tag{6.11}$$

Hierbei wird die Vorschlagswahrscheinlichkeit symmetrisch $\pi(\vec{x}_i, \vec{x}_j) = \pi(\vec{x}_j, \vec{x}_i)$ und ohne Berücksichtigung der vorgegebenen Verteilungsfunktion $p(\vec{x})$ gewählt. Die Akzeptanzwahrscheinlichkeit ist bei der Metropolis-Wahl gemäß

$$W(\vec{x}_i \to \vec{x}_j) = \min \left(1, \frac{p(\vec{x}_j)}{p(\vec{x}_i)} \right) \tag{6.12}$$

aus $p(\vec{x})$ zu bestimmen. Man überzeugt sich leicht, dass die Wahl (6.11) mit (6.12) das detaillierte Gleichgewicht (6.4) erfüllt.

Der Markov-Prozeß wird nun über folgende Vorschrift für eine Metropolis-Monte-Carlo-Simulation realisiert:

1. Schlage im Schritt i ein neues $\vec{x}_j = \vec{x}_i + \delta\vec{x}$ vor, wobei zufällig eine (kleine) Änderung $\delta\vec{x}$ gewählt wird.
2. Berechne $w = \dfrac{p(\vec{x}_j)}{p(\vec{x}_i)}$.
3. Ist $w \geq 1$, so akzeptiere das vorgeschlagene $\vec{x}_j$, d.h. setze $\vec{x}_{i+1} = \vec{x}_j$.
4. Ist $w < 1$, erzeuge eine Zufallszahl r, die in $[0, 1]$ gleichverteilt ist.
   (a) Im Fall $r \leq w$ akzeptiere ebenfalls das vorgeschlagene $\vec{x}_j$, d.h. setze $\vec{x}_{i+1} = \vec{x}_j$.
   (b) Für $r > w$ ist der Vorschlag zu verwerfen, d.h. $\vec{x}_{i+1} = \vec{x}_i$ zu setzen.

Wir erinnern daran, dass die Änderungen $\delta\vec{x}$ im ersten Schritt so zu wählen sind, dass der Algorithmus erstens ergodisch wird und zweitens symmetrisch, d.h. dass für den Zustand $\vec{x}_j$ die Änderung $-\delta\vec{x}$ mit der gleichen Wahrscheinlichkeit auftritt wie eine Änderung $\delta\vec{x}$ im Zustand $\vec{x}_i$.

Im stationären Zustand gilt nun

$$\langle f \rangle \approx \frac{1}{n} \sum_{i=1}^{n} f(\vec{x}_i)\,, \tag{6.13}$$

d.h. das ursprüngliche Problem (6.1) wird gelöst, indem man bei hinreichend späten Zeiten der Metroplolis-Simulation eine gewünschte Anzahl n von Messungen durchführt. An dieser Stelle gilt zu beachten, dass die Folge der $\vec{x}_i$ per Konstruktion korreliert ist. Der Fehler darf deswegen *nicht* einfach über (5.28) geschätzt werden.

Eine wichtige Anwendung des Metropolis-Algorithmus ist die Thermodynamik. Hier sind die Wahrscheinlichkeiten durch die *Boltzmann-Gewichte* von Konfigurationen C mit Energie $E(C)$ gegeben

$$p(C) = \frac{\mathrm{e}^{-E(C)/(k_B\,T)}}{Z}\,. \tag{6.14}$$

Hierbei ist k_B die Boltzmann-Konstante. Die Normierungskonstante in (6.14) $Z = \sum_C \mathrm{e}^{-E(C)/(k_B\,T)}$ ist im allgemeinen unbekannt[10].

[10]Tatsächlich hat Z in der statistischen Mechanik die Bedeutung der Zustandssumme. Allein die Kenntnis von Z erlaubt bereits im wesentlichen die Ableitung der thermodynamischen Eigenschaften des Systems.

s_i	Magnet	Besetzung
$+1$	$\uparrow$	besetzt
-1	$\downarrow$	leer

Tabelle 6.1: *Mögliche physikalische Interpretationen für Ising-Variable* $s_i = \pm 1$.

Wir wollen nun analog zu (6.1) Erwartungswerte einer (oder mehrer) Größen A berechnen (eine spezielle meßbare Größe ist die Energie E)

$$\langle A \rangle = \frac{\sum\limits_C A(C)\, \mathrm{e}^{-E(C)/(k_B\,T)}}{\sum\limits_C \mathrm{e}^{-E(C)/(k_B\,T)}}\,. \tag{6.15}$$

Im allgemeinen können die Summen nicht exakt ausgeführt werden. In einem solchen Fall kann man Importance sampling, d.h. den oben eingeführten Metropolis-Algorithmus verwenden. Analog zu (6.13) schätzen wir die Erwartungswerte (6.15) über

$$\langle A \rangle \approx \frac{1}{n} \sum_{i=1}^{n} A(C_i)\,. \tag{6.16}$$

6.1 Ising-Modell

Eine der bekanntesten Anwendungen des Metropolis-Algorithmus ist das *Ising-Modell*. Entsprechende umfangreiche Literatur existiert zu diesem Thema (wir verweisen unter anderem auf Kapitel 16 und 17 von [10], Kapitel 15 von [11], Kapitel 8.4 von [18] bzw. [19], Kapitel 4.1 und 4.2 von [21], Kapitel 8 von [30] sowie Kapitel 5.5 von [17]).

Das Ising-Modell wird mit Hilfe von N „Spin“-Variablen s_i definiert, die die Werte $s_i = \pm 1$ annehmen können. Mögliche physikalische Interpretationen dieser beiden Werte im Rahmen eines uniaxialen Magneten oder Besetzungen von Gitterplätzen, die höchstens einfach besetzt werden können, sind in Tab. 6.1 angegeben.

Das ferromagnetische Ising-Modell ist auf einem Gitter über die Energie

$$E(s_1, \ldots, s_N) = -J \sum_{\langle i,j \rangle} s_i\, s_j \tag{6.17}$$

definiert. Hierbei erstreckt sich die Summe über Paare benachbarter Gitterplätze $\langle i, j \rangle$. Ein parallel eingestelltes Paar von Spins $(s_i, s_j) = (\uparrow, \uparrow)$ oder $(\downarrow, \downarrow)$ trägt

zu der Summe in (6.17) einen Beitrag $E = -J$ bei, ein antiparallel eingestelltes Paar von Spins $(s_i, s_j) = (\uparrow, \downarrow)$ oder $(\downarrow, \uparrow)$ ergibt hingegen einen Beitrag $E = +J$. Somit favorisiert die Energie (6.17) für $J > 0$ parallele Einstellungen des Spins; thermische Fluktuationen erzeugen für $T > 0$ allerdings auch Beiträge von antiparallelen Spin-Einstellungen.

Das Problem ist nun, dass die Anzahl der Konfigurationen von N spins $s_1, \ldots, s_N$ gleich 2^N ist. Betrachten wir z.B. ein 16×16 Quadratgitter, d.h. $N = 256$, so treten in der Summe (6.15) $2^{256} \approx 10^{77}$ Terme auf, was offensichtlich nicht explizit durchführbar ist (bereits ein 8×8 Gitter, d.h. $N = 64$, führt auf $2^{64} > 10^{19}$ Summanden, was auch praktisch nicht durchführbar ist). Hinzu kommt, dass vor allen bei niedrigen Temperaturen durch die Exponentialfunktion in (6.15) stark unterdrückt sind.

Somit kommt Importance-Sampling in der Form des Metropolis-Algorithmus zum Einsatz. Wir stellen hier die Einzel-Spin-Flip-Variante des Metropolis-Algorithmus für das Ising-Modell vor:

1. Wähle einen Anfangszustand.
2. Wähle einen Spin j zufällig. Schlage vor, diesen zu „flippen“, d.h. $s_j \to -s_j$ zu ersetzen (s_i mit $i \neq j$ bleibt unverändert).
3. Berechne $\Delta E = E_{\text{neu}} - E_{\text{alt}}$.
4. Ist $\Delta E \leq 0$, so behalte den geflippten Spin und fahre bei 7. fort.
5. Ist $\Delta E > 0$, so berechne $w = \exp(-\Delta E/(k_B\, T))$.
6. Erzeuge eine Zufallszahl r, die in $[0, 1]$ gleichverteilt ist.
   (a) Im Fall $r \leq w$ akzeptiere den Spin-Flip.
   (b) Für $r > w$ ist der Spin-Flip zu verwerfen, d.h. man muß man zum Ausgangzustand für s_j zurückkehren.
7. Fahre bei 2. fort, solange bis hinreichend viele Konfigurationen erzeugt sind.

Zunächst stellen wir fest, dass die Schritte 3.–6. die Metropolis-Wahl (6.12) realisieren, die in diesem Fall

$$W(\text{alt} \to \text{neu}) = \min\left(1, \frac{p_{\text{neu}}}{p_{\text{alt}}}\right) = \min\left(1, \mathrm{e}^{-\Delta E/(k_B\, T)}\right) \tag{6.18}$$

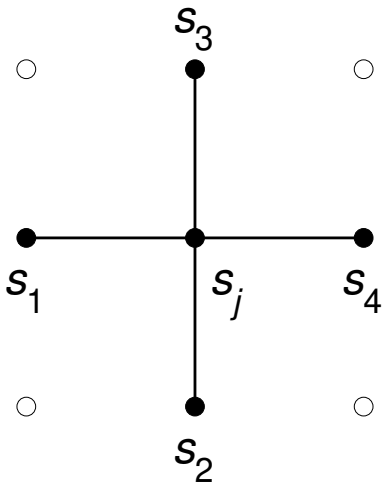


Abbildung 6.1: *Illustration der 4 Nachbarn eines Spins s_j im Ising-Modell auf dem Quadratgitter.*

lautet.

Genaugenommen hat man ein wenig Freiheit. So kann man die Gitterplätze statt der zufälligen Wahl im 2. Schritt auch geordnet durchgehen. Damit der Algorithmus ergodisch wird ist in jedem Fall darauf zu achten, dass alle Gitterplätze besucht werden.

Die Energie-Differenz ΔE im 3. Schritt sollte lokal berechnet werden. So hat der geflippte Spin s_j auf dem Quadratgitter nur 4 Nachbarn $s_1, \ldots, s_4$ (siehe Abb. 6.1). Bei dem Spin-Flip ändert sich nur die Wechselwirkung dieser vier Spins mit s_j, so dass sich mit (6.17) und $s_j^{\text{alt}} = -s_j^{\text{neu}}$

$$\Delta E = -2\,J\,(s_1 + s_2 + s_3 + s_4)\; s_j^{\text{neu}} \tag{6.19}$$

ergibt. Man kann noch eine weitere Optimierung durchführen: $(s_1 + s_2 + s_3 + s_4)\, s_j^{\text{neu}}$ kann lediglich die 5 verschiedenen Werte -4, -2, 0, 2 und 4 annehmen. Für diese 5 ganzzahligen Werte kann man leicht $w = \exp(-\Delta E/(k_B\,T))$ vorab berechnen und tabellieren, so dass die teure Exponentialfunktion im 5. Schritt entfällt.

Die gewünschten Messungen (6.16) kann man vor dem 7. Schritt durchführen. Allerdings sollte man dies nicht zu oft tun, da ein einzelner Schritt nur wenig an der Konfiguration ändert und Messungen meistens teuer sind. Typischerweise sollte man einen „Sweep“ abwarten, d.h. N^2 elementare MC-Schritte (bzw. im Mittel einen Flip-Versuch je Spin).

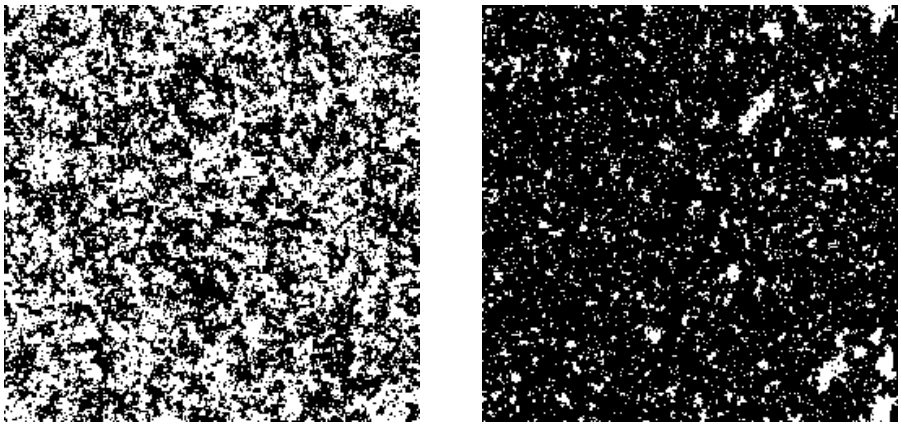


Abbildung 6.2: *Momentane Zustände des Ising-Modells auf einem* 256×256 *Quadratgitter in der ungeordneten Hochtemperatur-Phase* $T > T_c$ *(links) und der geordneten Tieftemperatur-Phase* $T < T_c$ *(rechts).*

Beispielsweise in einer solchen Simulation beobachtet man einen (magnetischen) Ordnungübergang bei eine kritischen Temperatur T_c. Für $T > T_c$ liegt ein (magnetisch) *un*geordneter Zustand vor (vgl. Abb. 6.2 links), für $T < T_c$ hingegen ein (magnetisch) geordneter Zustand (vgl. Abb. 6.2 rechts). Zur quantitativen Beschreibung dieser Physik führt man die Magnetisierung

$$M = \frac{1}{N} \sum_{i=1}^{N} s_i \tag{6.20}$$

ein. Aus Symmetrie-Gründen (gleichzeitige Inversion aller Spins, d.h. $s_i \to -s_i$) ist $\langle M \rangle = 0$ für ein endliches System und alle Temperaturen. Man betrachtet daher M^2. Für diese Größe gilt

$$\lim_{N \to \infty} \langle M^2 \rangle = M_0^2 > 0 \qquad \text{für} \quad T < T_c \,. \tag{6.21}$$

M_0 hat die Bedeutung einer „spontanen Magnetisierung“, die ein sehr großes System annimmt, wenn es nur auf kurzen Zeitskalen betrachtet wird. Die Spin-Inversions-Symmetrie ist also im thermodynamischen Limes $N \to \infty$ im geordneten Zustand ($T < T_c$) spontan gebrochen.

Auf folgende technische Details bei der Durchführung einer Simulation sei noch hingewiesen:

- Man verwendet gewöhnlich *periodische Randbedingungen*, um Randeffekte zu minimieren, die für endliche (insbesondere kleine) Systeme auftreten.

Dann gilt auch Translationsinvarianz und somit für alle i und j

$$\langle s_i \rangle = \langle s_j \rangle = \langle M \rangle \,. \tag{6.22}$$

- Zu Beginn der Simulation benötigt man einige Zeit, um in den stationären Zustand zu gelangen. Daher muß die Simulation „equilibriert“ werden, d.h. man muß sie einige Zeit laufen lassen, damit sie den willkürlich gewählten Anfangszustand vergißt. Erst nach dieser Equilibrations-Phase darf man Messungen von Gleichgewichts-Eigenschaften des Ising-Modells durchführen.

7 Quanten-Monte-Carlo

Zum Abschluß wollen wir noch einmal zur Quantenmechanik zurückkehren und Monte-Carlo-Verfahren für die Einteilchen-Schrödingergleichung vorstellen. Diese illustrieren Prinzipien allgemeinerer Quanten-Monte-Carlo (QMC)-Verfahren. Der Grundgedanke ist, das Quantensystem durch Einführung einer „imaginären Zeit“ auf ein klassisches Problem abzubilden, das dann mit einem Monte-Carlo-Verfahren simuliert werden kann. Die Diskussion in diesem Kapitel folgt den Kapiteln 18.6 und 18.7 von [10] bzw. Kapiteln 16.8 und 16.9 von [11]. Weitere nützliche Bemerkungen finden sich auch am Ende von Kapitel 8 von [18] bzw. [19]; im Übrigen sei auf die Original-Arbeiten von Anderson [1] verwiesen.

Grundsätzlich gehen wir von der zeitabhängigen Schrödinger-Gleichung (4.46) aus, mit der wir uns bereits im Unterkapitel 4.3 beschäftigt haben. Zunächst führen wir eine *imaginäre Zeit* ein:

$$\tau = \frac{i\,t}{\hbar}\,. \tag{7.1}$$

Setzen wir dies in (4.46) ein, so nimmt diese folgende Form an:

$$\frac{\partial\,\Psi\left(\vec{x},\tau\right)}{\partial\tau} = -\mathcal{H}\,\Psi\left(\vec{x},\tau\right) = \frac{\hbar^2}{2\,m}\,\Delta\Psi\left(\vec{x},\tau\right) - V\left(\vec{x}\right)\,\Psi\left(\vec{x},\tau\right)\,, \tag{7.2}$$

wobei wir auf der rechten Seite ferner die Orstdarstellung von (4.67) für den Hamilton-Operator eingesetzt haben.

Konzentrieren wir uns zunächst auf den ersten Term auf der rechten Seite von (7.2), d.h. wir setzen $V = 0$ und finden

$$\frac{\partial\,\Psi\left(\vec{x},\tau\right)}{\partial\tau} = \frac{\hbar^2}{2\,m}\,\Delta\Psi\left(\vec{x},\tau\right)\,. \tag{7.3}$$

Diese Gleichung kennen wir bereits aus Unterkapitel 4.1.3; es ist nichts anderes als die Diffusionsgleichung (4.27)! Durch Vergleich von (7.3) mit (4.27) ergibt sich die Diffusionskonstante zu

$$D = \frac{\hbar^2}{2\,m}\,. \tag{7.4}$$

Ignorieren wir hingegen den ersten Term auf der rechten Seite von (7.2) (formal können wir z.B. $\hbar = 0$ setzen), so erhalten wir folgende Differentialgleichung erster Ordnung in τ

$$\frac{\partial\,\Psi\left(\vec{x},\tau\right)}{\partial\tau} = -V\left(\vec{x}\right)\,\Psi\left(\vec{x},\tau\right)\,. \tag{7.5}$$

Für festes $\vec{x}$ erkennen wir auch hier etwas bekanntes wieder, nämlich je nach dem Vorzeichen von V am Ort $\vec{x}$ einen Zerfalls- oder Wachstums-Prozeß.

Diese beiden Beobachtungen legen es nahe, (7.2) zu simulieren, indem wir Zufallswege betrachten, bei denen Läufer nicht nur diffundieren, sondern auch sterben bzw. sich vervielfältigen können. Eine erste wesentliche Beobachtung ist in diesem Zusammenhang, dass wir nun die Wellenfunktion Ψ selbst (*nicht* $|\Psi|^2$) als Wahrscheinlichkeitsdichte interpretieren. Die Wellenfunktion muß somit reell und positiv sein, d.h. $\Psi \geq 0$. Dies gilt nun aber nur für den Grundzustand, so dass diese Art von QMC auf Grundzustands-Eigenschaften beschränkt ist.

Tatsächlich gilt nach (4.50) mit der imaginären Zeit (7.1)

$$\Psi(\vec{x}, \tau \to \infty) \approx a_0 \, \mathrm{e}^{-E_0 \, \tau} \, \Psi_0(\vec{x}) \tag{7.6}$$

und wir erhalten aus einer Simulation für große τ die Grundzustandswellenfunktion $\Psi_0(\vec{x})$.

7.1 Zufallsweg-Quanten-Monte-Carlo

Wir werden nun eine erste einfache Realisierung des skizzierten Algorithmus vorstellen, den wir als *Zufallsweg-Quanten-Monte-Carlo* bezeichnen. Im Prinzip betrachten wir N Läufer an verschiedenen Orten, die sowohl diffundieren als auch aussterben bzw. sich duplizieren können. Die Anzahl der Läufer n_i an einem „Ort“ $\vec{x}_i$ wird für große τ eine Näherung für die Grundzustandwellenfunktion ergeben.

Allerdings ergibt sich aus (7.6) das Problem, dass die Anzahl der Läufer N für $E_0 > 0$ im Verlauf der Simulation wächst, für $E_0 < 0$ hingegen abnimmt. Eigentlich ist es wünschenswert, die Zahl der Läufer (annährend) konstant zu halten. Dies kann erreicht werden, indem man ein Referenz-Potential $V_{\text{ref.}}$ einführt, d.h. (7.2) ersetzt durch

$$\frac{\partial \, \Psi \, (\vec{x}, \tau)}{\partial \tau} = \frac{\hbar^2}{2 \, m} \, \Delta \Psi \, (\vec{x}, \tau) - (V \, (\vec{x}) - V_{\text{ref.}}) \; \Psi \, (\vec{x}, \tau) \; . \tag{7.7}$$

Von der Physik her sind (7.2) und (7.7) äquivalent. Gelingt es uns, $V_{\text{ref.}} = E_0$ zu wählen, so ist unser Ziel erreicht und die Zahl der Läufer N wird während der Simulation annährend konstant bleiben. Wir müssen also die Grundzustandsenergie E_0 schätzen.

Wir werden nun zeigen, wie wir die Grundzustandsenergie messen können. Zunächst integrieren wir die Schrödingergleichung (7.7) über den d-dimensionalen

Ortsraum

$$\begin{aligned}\int \mathrm{d}^d x \, \frac{\partial \, \Psi \left(\vec{x}, \tau \right)}{\partial \tau} &= \int \mathrm{d}^d x \, \frac{\hbar^2}{2 \, m} \, \Delta \Psi \left(\vec{x}, \tau \right) - \int \mathrm{d}^d x \, V \left(\vec{x} \right) \, \Psi \left(\vec{x}, \tau \right) \\ &\quad + V_{\text{ref.}} \int \mathrm{d}^d x \, \Psi \left(\vec{x}, \tau \right) \\ &= - \int \mathrm{d}^d x \, V \left(\vec{x} \right) \, \Psi \left(\vec{x}, \tau \right) + V_{\text{ref.}} \int \mathrm{d}^d x \, \Psi \left(\vec{x}, \tau \right) \, , \quad (7.8)\end{aligned}$$

wobei der erste Term wegfällt, da die räumlichen Ableitungen $\partial \Psi \left(\vec{x}, \tau \right) / \partial x_i$ im Unendlichen verschwinden. Für $\tau \to \infty$ geht Ψ gegen die Grundzustandswellenfunktion. Nach (7.6) wird die Imaginärzeitentwicklung dann $\Psi(\vec{x}, \tau \to \infty) \approx a_0 \, \mathrm{e}^{(V_{\text{ref.}} - E_0) \, \tau} \, \Psi_0(\vec{x})$. Eingesetzt in die linke Seite von (7.8) führt dies auf

$$(V_{\text{ref.}} - E_0) \int \mathrm{d}^d x \, \Psi \left(\vec{x}, \tau \right) \approx - \int \mathrm{d}^d x \, V \left(\vec{x} \right) \, \Psi \left(\vec{x}, \tau \right) + V_{\text{ref.}} \int \mathrm{d}^d x \, \Psi \left(\vec{x}, \tau \right) \, . \quad (7.9)$$

Die Terme proportional zu $V_{\text{ref.}}$ heben sich weg, wir können nach E_0 auflösen und finden

$$E_0 = \lim_{\tau \to \infty} \frac{\int \mathrm{d}^d x \, V \left(\vec{x} \right) \, \Psi \left(\vec{x}, \tau \right)}{\int \mathrm{d}^d x \, \Psi \left(\vec{x}, \tau \right)} \approx \frac{\sum \, n_i \, V \left(\vec{x}_i \right)}{\sum \, n_i} = \langle V \rangle \, . \quad (7.10)$$

Die rechte Seite ergibt sich durch die Identifikation der Wellenfunktion mit der Anzahl n_i der Läufer am Punkt $\vec{x}_i$.

Diese Bemerkungen werden in folgendem Zufallsweg-QMC-Algorithmus umgesetzt:

1. Setze insgesamt N_0 Läufer an Anfangspositionen $\vec{x}_i$.
   Die Positionen $\vec{x}_i$ müssen nicht auf einem Gitter liegen.

2. Berechne das Referenz-Potential $V_{\text{ref.}} = \sum_i V \left(\vec{x}_i \right) / N_0$.

3. Für alle Läufer i

   (a) Wähle zufällig eine Einheitsrichtung $\pm \vec{e}_j$. Ersetze $\vec{x}_i \to \vec{x}_i \pm \Delta x \, \vec{e}_j$. Aufgrund des aus Unterkapitel 4.1.3 bekannten Zusammenhangs zwischen den Diskretisierungen und der Diffusionskonstanten D ist die feste räumliche Schrittweite $(\Delta x)^2 = 2d \, D \, \Delta \tau$ zu wählen, wobei $\Delta \tau$ die Länge eines Zeitschritts ist.
   In „natürlichen“ Einheiten $\hbar = m = 1$ ist nach (7.4) $D = 1/2$, so dass $(\Delta x)^2 = d \, \Delta \tau$ zu wählen ist.

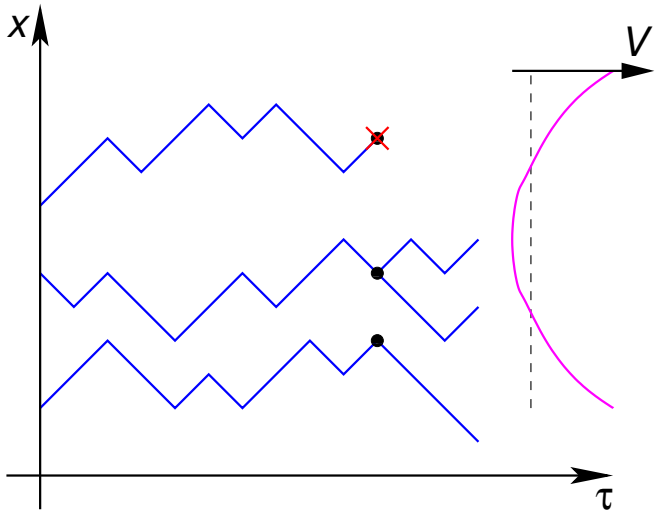


Abbildung 7.1: *Illustration des Zufallsweg-Quanten-Monte-Carlo-Algorithmus. Die Wege der Läufer sind durch Linien gekennzeichnet, Entscheidungen über Absterben oder Vervielfältigung der Läufer durch Punkte.*

(b) Berechne $\Delta V = V(\vec{x}_i) - V_{\text{ref.}}$ und eine Zufallszahl r, die im Intervall $[0, 1]$ gleichverteilt ist.

i. Ist $\Delta V > 0$ und $r < \Delta V \, \Delta\tau$, entferne den Läufer.

ii. Ist $\Delta V < 0$ und $r < -\Delta V \, \Delta\tau$, erzeuge einen weiteren Läufer am Punkt $\vec{x}_i$.

iii. Ansonsten bleibt der Läufer einfach am Punkt $\vec{x}_i$.

4. Berechne $\langle V \rangle$ nach (7.10) und die tatsächliche Anzahl der Läufer N. Adjustiere dann das Referenz-Potential zu

$$V_{\text{ref.}} = \langle V \rangle - \frac{a}{N_0 \, \Delta\tau} \, (N - N_0) \; . \tag{7.11}$$

Hierbei ist a so zu wählen, dass die Zahl der Läufer N näherungsweise konstant bleibt.
$a = 0$ ist eine mögliche Wahl. In diesem Fall kann aber die Zahl der Läufer stärker fluktuieren und sich ggfs. auch bei einem anderen Wert als N_0 einpendeln.

5. Wiederhole Schritte 3 und 4 bis sich die Grundzustandsenergie $\langle V \rangle$ auf einen festen Wert mit lediglich zufälligen Schwankungen eingependelt hat.

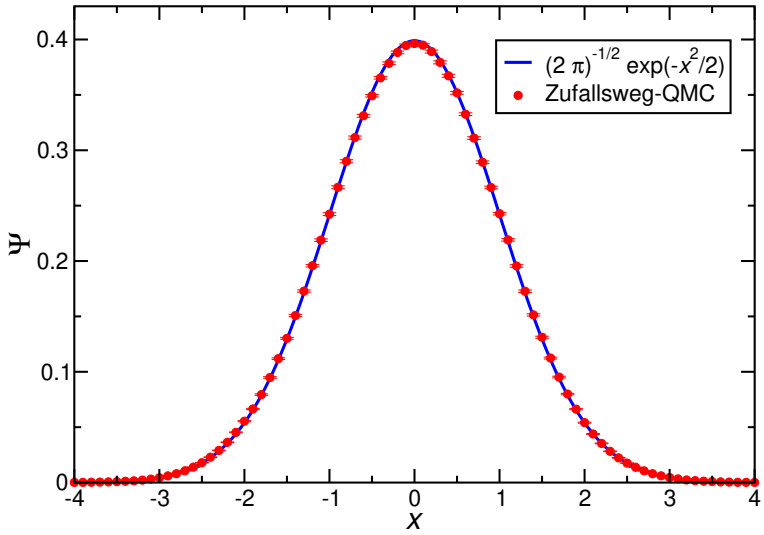


Abbildung 7.2: *Ergebnis einer Zufallsweg-QMC-Simulation für den eindimensionalen harmonischen Oszillator mit* $\hbar = m = \omega = 1$. *Die Parameter der Simulation waren* $\Delta x = 0.1$, $N_0 = 300$, $a = 1$. *Von der Simulation wurden die ersten 200 Schritte ignoriert und anschließend über 700 Schritte Daten gesammelt. Fehlerbalken wurden aus den Ergebnissen von 100 unabhängigen Simulationen geschätzt. Das Ergebnis für die Grundzustandsenergie war in dieser Simulation* $E_0 = 0.5034 \pm 0.0015$.

Mittele anschließend $\langle V \rangle$ über hinreichend viele Monte-Carlo-Schritte zur Bestimmung der Grundzustandsenergie. Schätze die Grundzustandswellenfunktion $\Psi_0(\vec{x})$ mit Hilfe einer analogen Mittelung für die Dichte der Läufer am Ort $\vec{x}$.

Dieser Algorithmus wird in Abb. 7.1 veranschaulicht. Linien zeigen die Diffusionsbewegung der Läufer (Schritt 3(a)), Punkte den Zerfalls- bzw. Wachstumsprozeß der Läufer (Schritt 3(b)). Die Wege der Läufer sind voneinander unabhängig (abgesehen vom Kopieren eines neuen Läufers sowie einer Rückkopplung über $V_{\text{ref.}}$).

Abb. 7.2 zeigt das Ergebnis einer solchen Zufallsweg-QMC-Simulation für den eindimensionalen harmonischen Oszillator, wobei „natürliche“ Einheiten $\hbar = m = \omega = 1$ verwendet wurden. Die Simulation wurde $n = 100$ mal wiederholt. Fehler dürfen dann mit Hilfe von (5.28) geschätzt werden, wobei die Statistik über die Mittelwerte der $n = 100$ einzelnen Simulationen auszuwerten ist. Die Ergebnisse sind konsistent mit bekannten Eigenschaften des harmonischen Oszillators: Für die Grundzustandswellenfunktion findet man eine Gauß-Kurve und

die geschätzte Grundzustandsenergie $E_0 = 0.5034 \pm 0.0015$ liegt in der Nähe des exakten Ergebnisses $E_0 = 1/2$.

7.2 Diffusions-Quanten-Monte-Carlo

Im Zufallsweg-QMC haben wir sowohl die Diffusion als auch den Zerfallsprozeß recht grob genähert. Die Ergebnisse sind somit nur für hinreichend kleine Δx und damit auch kleine $\Delta\tau$ genau. Tatsächlich zeichnen sich in Abb. 7.2 bereits systematische Abweichungen ab.

Ein Verbesserung dieser Näherungen führt auf den *Diffusions-Quanten-Monte-Carlo*-Algorithmus. Wir gehen von der Näherung (4.69) für den Zeitentwicklungsoperator aus. Angewandt auf (7.7) führt dies auf

$$\widetilde{U}(\Delta\tau) = \mathrm{e}^{-\mathcal{H}_0\,\Delta\tau}\,\mathrm{e}^{-(V-V_{\mathrm{ref.}})\,\Delta\tau}\,. \tag{7.12}$$

Tatsächlich ist es günstig, anstelle dieses Ausdrucks einen symmetrisierten zu verwenden, nämlich

$$\widehat{U}(\Delta\tau) = \mathrm{e}^{-\frac{V-V_{\mathrm{ref.}}}{2}\,\Delta\tau}\,\mathrm{e}^{-\mathcal{H}_0\,\Delta\tau}\,\mathrm{e}^{-\frac{V-V_{\mathrm{ref.}}}{2}\,\Delta\tau}\,. \tag{7.13}$$

$\mathrm{e}^{-\mathcal{H}_0\,\Delta\tau}$ ist der Zeitentwicklungsoperator über einen Zeitschritt $\Delta\tau$ für die Diffusionsionsgleichung. Deren Lösung kennen wir aus Unterkapitel 4.1.3 zumindest für eine Dimension. Die hier benötigte Verallgemeinerung von (4.33) auf d Dimensionen lautet

$$\Psi(\vec{x}, \tau + \Delta\tau|_{\mathrm{Diffusion}}) = \frac{1}{(4\,\pi\,D\,\Delta\tau)^{d/2}} \int \mathrm{d}^d x_0\,\mathrm{e}^{-\frac{(\vec{x}-\vec{x}_0)^2}{4\,D\,\Delta\tau}}\,\Psi(\vec{x}_0, \tau)\,. \tag{7.14}$$

Der Diffusionsprozeß kann somit verbessert simuliert werden, indem man einen Läufer am Punkt $\vec{x}_0$ jeweils um ein zufälliges $\Delta\vec{x} = \vec{x} - \vec{x}_0$ versetzt, das Gauß-verteilt ist mit Mittelwert $\vec{0}$ und Varianz $2\,D\,\Delta\tau$.

Die verbleibenden Faktoren in (7.13) implementiert man, indem man den Diffusionsschritt von $\vec{x}_0$ nach $\vec{x}$ mit

$$w(\vec{x}_0 \to \vec{x}, \Delta\tau) = \exp\left(-\left(\frac{V(\vec{x}) + V(\vec{x}_0)}{2} - V_{\mathrm{ref.}}\right)\,\Delta\tau\right) \tag{7.15}$$

gewichtet.

Die Umsetzung dieser Vorschriften erfolgt mit folgendem *Diffusions-QMC*-Algorithmus, den wir bewußt ähnlich zur Zufallsweg-QMC formulieren:

1. Setze insgesamt N_0 Läufer an Anfangspositionen $\vec{x}_i$.
   Es gibt kein Gitter, also sind die Positionen $\vec{x}_i$ kontinuierlich. Es ist günstig, die Läufer dort zu positionieren, wo die Wellenfunktion groß ist.

2. Berechne das Referenz-Potential $V_{\text{ref.}} = \sum_i V(\vec{x}_i)/N_0$.

3. Für alle Läufer i

   (a) *Diffusionsschritt*: Wähle zufällig einen Vektor $\vec{\xi}$, so dass dieser in alle Richtungen gleichverteilt ist und Varianz $2\,D$ besitzt. Setze den Läufer i an die neue Position $\vec{x}_i' = \vec{x}_i + \sqrt{\Delta\tau}\,\vec{\xi}$.
   In „natürlichen“ Einheiten $\hbar = m = 1$ ist nach (7.4) $D = 1/2$, so dass sich eine Varianz 1 für $\vec{\xi}$ ergibt. Dies kann erreicht werden, indem die *Komponenten* ξ_j von $\vec{\xi}$ zufällig so erzeugt werden, dass sie jeweils den Mittelwert 0 und Varianz 1 besitzen.

   (b) *Zerfalls- bzw. Wachstumsprozeß*: Wir müssen die Konfiguration nach (7.15) mit einem Faktor $w(\vec{x}_i \to \vec{x}_i', \Delta\tau)$ gewichten. Dazu gehen wir wie folgt vor: Wir erzeugen zunächst eine Zufallszahl r, die im Intervall $[0, 1]$ gleichverteilt ist. Damit berechnen wir

   $$n_i = \left[\exp\left(-\left(\frac{V(\vec{x}_i') + V(\vec{x}_i)}{2} - V_{\text{ref.}}\right)\Delta\tau\right) + r\right], \tag{7.16}$$

   wobei $[z]$ der ganzzahlig Anteil von z ist. Wir erzeugen schließlich insgesamt n_i Exemplare des Läufers am Punkt $\vec{x}_i'$.
   Mit Hilfe der Zufallszahl r und dem ganzzahligen Anteil erhalten wir ganze Zahlen n_i, die im Mittel (7.15) ergeben. Da $n_i = 0$ auch möglich ist, ergibt sich die Entfernung des Läufers an der Stelle $\vec{x}_i'$, ohne dass wir diesen Fall gesondert betrachten müssen.

4. Berechne $\langle V \rangle$ nach (7.10) und die tatsächliche Anzahl der Läufer N. Adjustiere dann das Referenz-Potential gemäß (7.11). Hierbei ist a wieder so zu wählen, dass die Zahl der Läufer N näherungsweise konstant bleibt.

5. Wiederhole Schritte 3 und 4 bis sich die Grundzustandsenergie $\langle V \rangle$ auf einen festen Wert mit lediglich zufälligen Schwankungen eingependelt hat. Mittele anschließend $\langle V \rangle$ über hinreichend viele Monte-Carlo-Schritte. Erzeuge ein Histogramm der Läufer an den Orten $\vec{x}_i$ zur Schätzung der Grundzustandswellenfunktion.

Die bisher vorgestellten Algorithmen können leider ineffizient werden. Wenn das Potential groß und negativ wird (wie z.B. beim Coulomb-Potential), kann die Anzahl der Kopien eines Läufers nämlich sehr groß werden. Die Effizienz der Algorithmen kann durch Importance-Sampling verbessert werden. Der Grundgedanke ist die Verwendung einer Testwellenfunktion Ψ_T, mit deren Hilfe die Läufer in die wichtigen Gebiete geführt werden. Um dies zu konkretisieren, betrachten wir die Funktion

$$f(\vec{x},\tau) = \Psi(\vec{x},\tau)\,\Psi_T(\vec{x})\,. \tag{7.17}$$

Durch Einsetzen von (7.7) kann man zeigen, dass die Funktion f der partiellen Differentialgleichung

$$\frac{\partial\, f\,(\vec{x},\tau)}{\partial\tau} = D\,\Delta f\,(\vec{x},\tau) - D\,\vec{\nabla}\cdot\Big(f\,(\vec{x},\tau)\;\vec{F}(\vec{x})\Big) - \left(E_L\,(\vec{x}) - V_{\text{ref.}}\right)\,f\,(\vec{x},\tau) \tag{7.18}$$

genügt, wobei die *treibende Kraft*

$$\vec{F}(\vec{x}) = \frac{2}{\Psi_T(\vec{x})}\,\vec{\nabla}\Psi_T(\vec{x}) \tag{7.19}$$

sowie die *lokale Energie*

$$E_L(\vec{x}) = V(\vec{x}) - \frac{D}{\Psi_T(\vec{x})}\,\Delta\Psi_T(\vec{x}) \tag{7.20}$$

auftreten. Die lokale Energie $E_L(\vec{x})$ übernimmt nun die Rolle des Potentials $V(\vec{x})$.

Die treibende Kraft (7.19) führt zu einen Drift-Term im Zufallsweg, d.h. sie kann in (7.14) mit der Ersetzung

$$\mathrm{e}^{-\frac{(\vec{x}-\vec{x}_0)^2}{4\,D\,\Delta\tau}} \;\longrightarrow\; \mathrm{e}^{-\frac{(\vec{x}-\vec{x}_0-D\,\Delta\tau\vec{F}(\vec{x}_0))^2}{4\,D\,\Delta\tau}} \tag{7.21}$$

berücksichtigt werden. Leider zerstört diese Ersetzung die Symmetrie zwischen $\vec{x}$ und $\vec{x}_0$. Diese kann wiederhergestellt werden, wenn wir die neue Position $\vec{x}$ nur nach einem Metropolis-Schritt akzeptieren. Wir führen also eine Akzeptanz-Wahrscheinlichkeit

$$W(\vec{x}_0\to\vec{x}) = \min\left(1, p(\vec{x}_0,\vec{x})\right) \qquad \text{mit} \qquad p(\vec{x}_0,\vec{x}) = \frac{\left|\Psi_T(\vec{x})\right|^2\;\mathrm{e}^{-\frac{(\vec{x}_0-\vec{x}-D\,\Delta\tau\vec{F}(\vec{x}))^2}{4\,D\,\Delta\tau}}}{\left|\Psi_T(\vec{x}_0)\right|^2\;\mathrm{e}^{-\frac{(\vec{x}-\vec{x}_0-D\,\Delta\tau\vec{F}(\vec{x}_0))^2}{4\,D\,\Delta\tau}}} \tag{7.22}$$

ein.

Der Diffusions-QMC-Algorithmus ändert sich damit wie folgt:

1. Wähle eine Testwellenfunktion $\Psi_T(\vec{x})$.

2. Setze insgesamt N_0 Läufer an Anfangspositionen $\vec{x}_i$.

3. Berechne das Referenz-Potential $V_{\text{ref.}} = \sum_i E_L(\vec{x}_i)/N_0$.

4. Für alle Läufer i

   (a) Wähle zufällig einen Vektor $\vec{\xi}$, so dass dieser in alle Richtungen gleichverteilt ist und Varianz $2\,D$ besitzt. Schlage eine neue Position $\vec{x}_i' = \vec{x}_i + D\,\Delta\tau\vec{F}(\vec{x}) + \sqrt{\Delta\tau}\,\vec{\xi}$ vor. Berechne $w = p(\vec{x}_i, \vec{x}_i')$ nach (7.22).

      i. Ist $w \geq 1$, so akzeptiere die neue Position $\vec{x}_i'$ und fahre bei (b) fort.

      ii. Ist $w < 1$, so erzeuge eine Zufallszahl R, die in $[0,1]$ gleichverteilt ist. Im Fall $R \leq w$ akzeptiere die neue Position $\vec{x}_i'$. Für $R > w$ ist der Schritt zu verwerfen, d.h. der Läufer muß zur Position $\vec{x}_i$ zurückkehren.

   (b) Erzeuge eine Zufallszahl r, die im Intervall $[0,1]$ gleichverteilt ist. Berechne

   $$n_i = \left[\exp\left(-\left(\frac{E_L(\vec{x}_i') + E_L(\vec{x}_i)}{2} - V_{\text{ref.}}\right)\zeta\,\Delta\tau\right) + r\right], \qquad (7.23)$$

   wobei ζ die mittlere Akzeptanzrate aus Schritt (a) ist (der effektive Zeitschritt $\zeta\,\Delta\tau$ ergibt sich dadurch, dass im Diffusionsschritt (a) einige Schritte verworfen werden). Erzeuge schließlich insgesamt n_i Exemplare des Läufers am Punkt $\vec{x}_i'$.

5. Berechne die tatsächliche Anzahl der Läufer N und $\langle V\rangle = \sum_i E_L(\vec{x}_i)/N$. Adjustiere dann das Referenz-Potential nach (7.11) mit geeignetem a.

6. Wiederhole Schritte 4 und 5 bis sich die Grundzustandsenergie $\langle V\rangle$ auf einen festen Wert mit lediglich zufälligen Schwankungen eingependelt hat. Mittele anschließend $\langle V\rangle$ über hinreichend viele Monte-Carlo-Schritte. Erzeuge ein Histogramm der Läufer an den Orten $\vec{x}_i$ zur Schätzung von $\Psi_0(\vec{x}_i)\,\Psi_T(\vec{x}_i)$, wobei Ψ_0 die Grundzustandswellenfunktion ist.

8 Nachbemerkungen

Dieses Skript sollte nicht losgelöst von den zugehörigen Übungen betrachtet werden. So werden teilweise hier Probleme vorbereitet, die dann in den Übungen gelöst wurden. Daher danke ich besonders Sebastian Fuchs und Jürgen Lampe für die Betreuung der Übungen. Ein Dank geht auch an alle Teilnehmer der Vorlesung und Übungen, die geduldig bis zum Ende durchgehalten haben, und vor allem diejenigen, die durch Fragen und Vorschläge zu Verbesserungen dieses Skripts beigetragen haben.

Die Literatur habe ich unter anderem unter dem Gesichtspunkt ihrer Verfügbarkeit an der Universität Göttingen zum aktuellen Zeitpunkt ausgewählt. Dies mag sich insbesondere bei den Online-Ressourcen in Zukunft ändern. Im Nachhinein habe ich gemerkt, dass auch das Buch [7] ganz gut zum Inhalt der Vorlesung paßt. Ich habe jedoch darauf verzichtet, im Verlauf des Skripts entsprechende Referenzen zu ergänzen.

Göttingen, 16. Juli 2007

Andreas Honecker

Literatur

[1] J.B. Anderson, *A Random-Walk Simulation of the Schrödinger Equation: H_3^+*, J. Chem. Phys. **63** (1975) 1499-1503; *Quantum Chemistry by Random Walk. $H\ ^2P$, $H_3^+\ D_{3h}\ ^1A_1'$, $H_2\ ^3\Sigma_u^+$, $H_4\ ^1\Sigma_g^+$, $Be\ ^1S$*, J. Chem. Phys. **65** (1976) 4121-4127; *Quantum Chemistry by Random Walk: Higher Accuracy*, J. Chem. Phys. **73** (1980) 3897-3899.

[2] K. Binder, D. Stauffer, *A Simple Introduction to Monte Carlo Simulation and Some Specialized Topics*, in: *Applications of the Monte Carlo Method*, K. Binder (Hrsg.), Springer Berlin (1984).

[3] M. Bollhöfer, *Numerische Behandlung partieller Differentialgleichungen*, Skript TU Berlin, Sommersemester 2001.

[4] M.D. Feit, J.A. Fleck, Jr., A. Steiger, *Solution of the Schrödinger Equation by a Spectral Method*, J. Computational Phys. **47** (1982) 412-433.

[5] A.M. Ferrenberg, D.P. Landau, Y.J. Wong, *Monte Carlo Simulations: Hidden Errors from "Good" Random Number Generators*, Phys. Rev. Lett. **69** (1992) 3382-3384.

[6] J.H. Ferziger, M. Perić, *Computational Methods for Fluid Dynamics*, Springer-Verlag, Berlin (2002).

[7] N.J. Giordano, H. Nakanishi, *Computational Physics*, Second Edition, Pearson/Prentice Hall, Upper Saddle River (2006).

[8] G.H. Golub, C.F. Van Loan, *Matrix Computations*, Johns Hopkins University Press, Baltimore (1996).

[9] C. Goto, Y. Ogawa, N. Shuto, F. Imamura, *IUGG/IOC TIME Project: Numerical Method of Tsunami Simulation with the Leap-Frog Scheme*, IOC Manual **35**, UNESCO (1997).

[10] H. Gould, J. Tobochnik, *An Introduction to Computer Simulation Methods*, Second Edititon, Addison-Wesley, Reading, Massachusetts (1996).

[11] H. Gould, J. Tobochnik, W. Christian *An Introduction to Computer Simulation Methods: Applications to Physical Systems*, Third Edition, Pearson/Addison-Wesley, San Francisco (2007).

[12] Ch. Großmann, H.-G. Roos, *Numerik partieller Differentialgleichungen*, B.G. Teubner, Stuttgart (1994).

[13] J.M. Hammersley, D.C. Handscomb, *Monte Carlo Methods*, Methuen & Co Ltd, London (1975).

[14] M.R. Hestenes, E. Stiefel, *Methods of Conjugate Gradients for Solving Linear Systems*, J. Res. Nat. Bur. Stand. **49** (1952) 409-435.

[15] A. Honecker, *Einführung in die Rechnerbedienung*, Blockkurs Universität Göttingen (2007).

[16] A. Jennings, J.J. McKeown, *Matrix Computation*, John Wiley, Chichester (1992).

[17] W. Kinzel, G. Reents, *Physik per Computer*, Spektrum Akademischer Verlag, Heidelberg (1996).

[18] S.E. Koonin, *Physik auf dem Computer 2*, Oldenbourg Verlag, München (1987).

[19] S.E. Koonin, D.C. Meredith, *Computational Physics. Fortran Version*, Westview Press, Boulder (1990).

[20] E.-K. Kunst, J. Quade, *Kern-Technik*, Folge 30, Linux-Magazin **11** (2006) 110-113.

[21] D.P. Landau, K. Binder, *A Guide to Monte Carlo Simulations in Statistical Physics*, Second Edition, Cambridge University Press, Cambridge (2005).

[22] C. Leforestier, R.H. Bisseling, C. Cerjan, M.D. Feit, R. Friesner, A. Guldberg, A. Hammerich, G. Jolicard, W. Karrlein, H.-D. Meyer, N. Lipkin, O. Roncero, R. Kosloff, *A Comparison of Different Propagation Schemes for the Time Dependent Schrödinger Equation*, J. Computational Phys. **94** (1991) 59-80.

[23] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, E. Teller, *Equation of State Calculations by Fast Computing Machines*, J. Chem. Phys. **21** (1953) 1087-1092.

[24] W.H. Press, S.A. Teukolsky, W.T. Vetterling, B.P. Flannery, *Numerical Recipes in C*, Cambridge University Press (1992).

[25] P.J. Roache, *Computational Fluid Dynamics*, Hermosa, Albuquerque (1982).

[26] F. Schmid, N.B. Wilding, *Errors in Monte Carlo Simulations Using Shift Register Random Number Generators*, Int. Jour. of Mod. Phys. **C6** (1995) 781-787 (Preprint-Version).

[27] K. Schönhammer, Mechanik-Skript.

[28] P.B. Visscher, *A Fast Explicit Algorithm for the Time-Dependent Schrödinger Equation*, Computers in Physics **5** (1991) 596-598.

[29] U. Wolff, O. Bär, O. Witzel, *Computational Physics I*, Skriptum HU Berlin, WS 2005/06.

[30] U. Wolff, O. Bär, O. Witzel, *Computational Physics II*, Skriptum HU Berlin, SS 2006.