Numerische Stabilitätsanalyse von Spektralmethoden für zeitabhängige partielle Differentialgleichungen

Masterarbeit

Zur Erlangung des akademischen Grades des Master of Science
Fachbereich Mathematik

Vorgelegt von Sven Leitgeber am 23.04.2021
Matrikelnummer: 2278610

Betreuer: Prof. Dr. Wilhelm Heinrichs

Fakultät für Mathematik der Universität Duisburg-Essen

Inhaltsverzeichnis

Einleitung

Als Hauptvorteil von Spektralmethoden gegenüber finite Differenzen oder finite Elemente, wird meistens die hohe Konvergenzordnung hervorgehoben, sofern die Lösung ausreichende Glattheit besitzt. Beim Lösen von zeitabhängigen partiellen Differentialgleichungen ist es allerdings allgemein effizienter Spektralmethoden nur auf die räumliche Dimension anzuwenden und die zeitabhängige Lösung iterativ von einem Zeitschritt zum nächsten zu berechnen. Dadurch kann jedoch bei der Berechnung in der Zeitdimension, bei ungeeigneter Wahl der Zeitschrittweite, der Fehler für jeden weiteren Zeitschritt immer weiter anwachsen, sogar in exponentieller Weise, wodurch die Konvergenz der numerischen Lösung nicht mehr gegeben ist. Obwohl es möglich ist die Zeitkoordinate auch spektral zu behandeln, ist die eben genannte Vorgehensweise deutlich günstiger als die Berechnung über das gesamte Raum-Zeit-Gebiet auf einmal. Somit bleibt das Problem der richtigen Zeitschrittwahl, das heißt eine, welche die Konvergenz der Lösung garantiert. In diesem Zusammenhang sei hier der Begriff der Stabilität verwendet.
Die Vorgehensweise der Diskretisierung der räumlichen Koordinaten auf ein System von gewöhnlichen Differentialgleichungen, die nur noch von der Zeitkoordinate abhängen, wird auch oft als Linienmethode bezeichnet. Bedingungen zu finden, welche für die Linienmethode eine konvergente Lösung sicherstellen, ist ein zentrales Thema dieser Arbeit. Dabei spielen unter anderem die Eigenwerte der Diskretisierung eine wichtige Rolle. Ein Nachteil von Spektralmethoden gegenüber finite Elemente/Differenzen ist die Vollbesetztheit der aus der Diskretisierung resultierenden Matrix, die im Regelfall auch nicht normal ist. In solchen Fällen kann es sein, dass die Eigenwertanalyse nicht mehr ausreichend gute Vorhersagen macht, mit Pseudospektren sind dann eventuell bessere Vorhersagen möglich.
Das auf ein System von gewöhnlichen Differentialgleichungen in der Zeit reduzierte Problem kann dann zum Beispiel mit einem Runge-Kutta-Verfahren oder einem linearen Mehrschrittverfahren gelöst werden. Ein wichtiges Werkzeug im Zusammenhang mit diesen Methoden sind Stabilitätsgebiete, die in Verbindung mit den Eigenwerten oder Pseudospektren der räumlichen Diskretisierung für entsprechende Restriktionsabschätzungen für die Zeitschrittwahl behilflich sein können.
Die Grundlagen zur Implementierung von oft benutzten Spektralmethoden werden im ersten Kapitel kurz erläutert, dabei werden überwiegend die Grundlagen aus „Spectral Methods“ von Shen,Tang,Wang [21] und „Spectral Methods in MATLAB“ von Trefethen [25] übernommen. Es handelt sich dabei hauptsächlich um die Berechnung von diskreten Ableitungen von Funktionen, um diese für die Diskretisierung von Differentialgleichungen zu verwenden. Die erste dieser Methoden besteht darin, auf die abzuleitende Funktion zuerst eine diskrete Fouriertransformation anzuwenden, um dann im Fourierraum mit einfachen Regeln eine Ableitung durchzuführen. Mit der anschließenden Rücktransformation erhält man dann die gesuchte diskrete Ableitung. Für die Anwendung wird im ersten Kapitel eine Ableitungsmatrix hergeleitet, so dass diese Methode mit einer Matrix-Vektor-Multiplikation umgesetzt werden kann. Eine weitere Methode um auch für die Diskretisierung einer Differentialgleichung die Randbedingungen mit zu berücksichtigen, führt über die Verwendung eines speziellen Diskretisierungsgitters, definiert über Chebyshev-Polynome. Auch hierfür wird eine Ableitungsmatrix vorgestellt. Auch wenn dies z.B. in MATLAB recht einfach zu implementieren ist, ist der Weg über Ableitungsmatrizen oft nicht der schnellste was Rechenzeit

betrifft. Daher wird alternativ sowohl für die Fourier- als auch die Chebyshevmethode eine effizientere Variante mittels schneller Fouriertransformation vorgestellt. Die Betrachtung von Ableitungsmatrizen macht trotzdem Sinn, da für die Überlegungen zur Stabilität in den darauf folgenden Kapiteln, die zugehörigen Eigenwerte von Bedeutung sein werden. Wie sich zeigen wird, ist vor allem das Auffinden der Eigenwerte der Chebyshev-Ableitungsmatrix zweiter Ordnung nicht trivial, so dass eine numerische Annäherung herangezogen werden muss.
Das Lösen einer partiellen Differentialgleichung durch Reduktion auf ein System von gewöhnlichen Differentialgleichungen, erfordert natürlich die Kenntnis über Lösungsmethoden für Letzteres. Daher werden ausgehend von einigen Standardwerken der Numerik von LeVeque [16], Rannacher [18] und Süli und Mayers [22] im zweiten Kapitel Einschrittverfahren und lineare Mehrschritt-verfahren zum Lösen von gewöhnlichen Differentialgleichungen vorgestellt. Dabei geht es vor allem um Konvergenzaussagen für diese Verfahren, insbesondere um Konvergenzbedingungen, die an die Wahl von Zeitschrittweiten gebunden sind. Diese Bedingungen sind vor allem für die konkrete Berechnung der Differentialgleichungen und nicht nur für theoretische Überlegungen von Bedeutung. Die Konvergenz des Verfahrens im Hinblick auf die Wahl der Zeitschritte, stellt die erste wichtige Art der Stabilität in dieser Arbeit dar. Dabei sind die Werkzeuge zur Auffindung von geeigneten Schrittweiten für Ein- und Mehrschrittverfahren nicht die gleichen. Für beide Arten von Verfahren spielen jedoch Stabilitätsgebiete eine bedeutende Rolle. Genauso der Zusammenhang dieser Gebiete mit den Eigenwerten der diskretisierten Gleichung. Stabilitätsgebiete grafisch darzustellen kann von Vorteil sein, wenn man eine grobe Orientierung dafür haben will, ob Eigenwerte innerhalb des Gebietes liegen. Daher werden hier einfache Algorithmen vorgestellt wie diese grafische Darstellung mit Hilfe von MATLAB umgesetzt werden kann. Im Allgemeinen ist es bekanntlich nicht so einfach das Stabilitätsgebiet von Hand zu bestimmen, sofern es sich nicht bloß um ein Intervall handelt.
Da wie schon eingangs erwähnt, die Betrachtung von Eigenwerten nicht immer ausreichend sind, benutzen wir Pseudospektren als eine Verallgemeinerung von Eigenwerten. Dies ist der Inhalt von Kapitel drei, das sich überwiegend an „Spectra and Pseudospectra“ von Lloyd N. Trefethen und Mark Embree [27] hält. Es gibt dabei verschiedene, in bestimmten Fällen zueinander äquivalente Definitionen von Pseudospektren, da je nach Situation eine jeweils andere Definition für Berechnungen oder Beweise hilfreich sein kann. Einige wichtige Eigenschaften von Pseudospektren finden sich ebenfalls in diesem Kapitel, insbesondere bezogen auf den Unterschied zwischen normalen und nicht normalen Matrizen. Dabei ist auch der Vergleich von Lage oder Größe des Pseudospektrums einer Matrix mit der Menge der Eigenwerte, also dem Spektrum, interessant. Die Menge des Pseudospektrums ist für gewöhnlich an einen gewissen Parameter ε gebunden. Ähnlich wie der Spektralradius einer Matrix, kann auch ein gewisser Radius eines ε-Pseudospektrums für ein bestimmtes ε berechnet und grafisch dargestellt werden. Damit kann nun ähnlich wie mit den Eigenwerten, die Entfernung eines ε-Pseudospektrums zu einem Stabilitätsgebiet bestimmt werden, was sich, wie sich herausstellen wird, zur Bestimmung von geeigneten Zeitschritten als nützlich erweist.
In Kapitel vier wird dazu eine nicht allgemeingültige, aber für Anwendungsfälle als oft ausreichend zu betrachtende Faustregel formuliert, die den Zusammenhang der Stabilitätsgebiete mit Eigenwerten oder bedarfsweise Pseudospektren beinhaltet. Diese Regel kann dann als eine Restriktionsbedingung für die Linienmethode zur Lösung von zeitabhängigen partiellen Differentialgleichungen betrachtet werden. Basierend auf den Erkenntnissen aus den vorherigen

Kapiteln wird dies präziser über einen Satz formuliert, der in [19,20] von Reddy und Trefethen bewiesen wurde. Dieser Satz gibt eine Abschätzung für stabile Zeitschritte für zeitabhängige partielle Differentialgleichungen an, auch für den allgemeineren Fall einer nicht normalen räumlichen Diskretisierung. Der Beweis dieses Satzes wird hier nach der Vorgehensweise wie Reddy und Trefethen es erstmals bewiesen haben, wiedergegeben. Der etwas technische Beweis erfordert ein paar Vorüberlegungen und stützt sich hauptsächlich auf einem Hilfsresultat, das eine Version des als Kreiss-Matrix-Theorem bekannten Satzes ist. Des Weiteren erfordert der Beweis ein paar Rückgriffe auf Resultate aus der Funktionentheorie, wie die Berechnung von Residuen, und vor allem die Verwendung des Dunford-Taylor-Integrals zur Darstellung einer Resolvente als Integral, die in dem umfangreichen Werk von Tasio Kato „Perturbation Theory for Linear Operators“ [14] nachgelesen werden können.
Abschließend findet sich in Kapitel fünf, mit etwas modifizierten Programmen aus dem 10. Kapitel des Buches „Spectral methods in MATLAB“ eine Nachstellung der Versuchsreihe, wie sie auch bei Trefethen durchgeführt wird, um zu erproben wie bei Anwendung einer Linienmethode die Eigenwerte der Diskretisierungsmatrix mit den Stabilitätsgebieten in Beziehung stehen müssen, um für zeitabhängige Probleme passende Zeitschritte zu finden. Dabei wird eine räumliche Diskretisierung mit den zuvor in Kapitel eins behandelten Spektralmethoden durchgeführt und geeignete Differenzenverfahren für die Zeitintegration verwendet. Als Erweiterung dazu, folgt ein Beispiel einer Wellengleichung erster Ordnung, als ein Fall der illustriert, wie Pseudospektren bei nicht ausreichen der Eigenwertanalyse eine geeignetere Abschätzung liefern können. Als kleines Unterthema wird zusätzlich die Methode des integrierenden Faktors vorgestellt, als eine Methode um Restriktionen an Zeitschritte zu lockern und damit Rechenzeit einzusparen. In Anlehnung an die Übungsaufgaben aus Kapitel 10 von „Spectral methods in MATLAB“, folgen weitere Anwendungsbeispiele der Linienmethode und des integrierenden Faktors auf ein paar bekannte zeitabhängige partielle Differentialgleichungen. Darunter befindet sich die Korteweg-de-Vries-Gleichung und die Kuramoto-Sivashinsky-Gleichung, sowie die Burgersgleichung. Die Berechnung dieser Gleichungen erfolgen wie in [21,25] mit einem integrierenden Faktor und einem Runge-Kutta-Verfahren vierter Stufe für die Zeitintegration.

1 Fourier und Chebyshev Spektralmethoden

Die grundlegende Idee von Spektralmethoden beruht auf der Annahme, dass eine unbekannte Funktion $u(x)$ durch eine Summe von $N+1$ Basisfunktionen $\phi_n(x)$ approximiert werden kann:

$$u(x) \approx u_N(x) = \sum_{n=0}^{N} a_n \phi_n(x)$$

Setzt man diese Summe in die Gleichung

$$Lu = f(x)$$

ein, wobei L der entsprechende Operator der Differential- oder Integralgleichung ist, erhält man daraus die sogenannte Residuumsfunktion

$$R(x; a_0, a_1, \ldots, a_N) = Lu_N - f.$$

Da die Residuumsfunktion für die exakte Lösung gleich null ist, besteht das Ziel darin, die Koeffizienten a_n so zu wählen, dass $R(x; a_0, \ldots, a_N)$ minimiert wird.
Spektralmethoden fallen in zwei große Kategorien, grob gesagt, die der interpolierenden und die der nicht interpolierenden. Die von der interpolierenden Art werden auch als pseudospektrale Methoden bezeichnet, bei denen die Basisfunktionen mit einem Gitter assoziiert werden. Die Koeffizienten a_n können dadurch gefunden werden, dass die Residuumsfunktion an jedem dieser Gitterpunkte gleich null wird. Das heißt eine pseudospektrale Methode fordert, dass die Differentialgleichung an diesen Punkten exakt erfüllt ist. Die entsprechenden Gitterpunkte heißen Interpolations- oder auch Kollokationspunkte. Da die Residuumsfunktion dazu gezwungen ist an den Gitterpunkten zu verschwinden, wird sie mit einer Erhöhung der Anzahl dieser Punkte, im Bereich zwischen diesen Punkten immer näher an null heranrücken, so dass $u_N(x)$ mit wachsendem N gegen $u(x)$ konvergiert [4].
Die hier Benutzten Methoden basieren zum einen auf Fourierreihen und zum anderen auf Chebyshev-Polynome, bei beiden auch unter Zuhilfenahme einer schnellen Fouriertransformation, auch FFT (fast Fourier transform) genannt.
Wir beginnen mit stetigen Fourierreihen. Zunächst sei das komplexe Exponential bezeichnet durch

$$e^{ikx} = \cos(kx) + i\sin(kx) = (\cos(x) + i\sin(x))^k.$$

Die Menge $\left\{e^{ikx} : k \in \mathbb{Z}\right\}$ bildet ein vollständiges Orthogonalsystem im komplexen Hilbertraum $L^2(0, 2\pi)$, ausgestattet mit dem Skalarprodukt

$$(u, v) = \frac{1}{2\pi}\int_0^{2\pi} u(x)\bar{v}(x)dx, \qquad \text{und der Norm} \qquad \|u\| = \sqrt{(u, u)}.$$

$\bar{v}$ ist dabei das komplex konjugierte zu v.

Für jede komplexwertige Funktion $u \in L^2(0, 2\pi)$ ist ihre Fourierreihe definiert durch

$$\mathcal{F}(u)(x) := \sum_{k=-\infty}^{\infty} \hat{u}_k e^{ikx},$$

wobei die sog. Fourierkoeffizienten gegeben sind durch

$$\hat{u}_k = (u, e^{iky}) = \frac{1}{2\pi} \int_0^{2\pi} u(x) e^{-ikx} dx. \tag{1.1}$$

Man sieht leicht, dass wenn u reellwertig ist, die Fourierkoeffizienten $\hat{u}_{-k} = \overline{\hat{u}}_k,\ k \in \mathbb{Z}$, erfüllen und $\hat{u}_0$ reell ist.
Zur Approximation an ein $u \in (0, 2\pi)$ benutzen wir das trigonometrische Polynom

$$\mathcal{F}_N(u) := \sum_{k=-N}^{N} \hat{u}_k e^{ikx}, \tag{1.2}$$

welches in der L^2-Norm gegen u konvergiert. Wenn u stetig, periodisch und von beschränkter Variation auf $[0, 2\pi]$ ist, dann konvergiert $\mathcal{F}_N(u)$ gleichmäßig gegen u [21].
Die Approximation (1.2) kann auch mit einer Faltung formuliert werden, und zwar

$$\mathcal{F}_N(u)(x) = (\mathcal{D}_N * u)(x) = \frac{1}{2\pi} \int_0^{2\pi} \mathcal{D}_N(x - t) u(t) dt,$$

wobei $\mathcal{D}_N$ den Dirichlet-Kern bezeichnet, welcher gegeben ist durch

$$\mathcal{D}_N(x) := \sum_{k=-N}^{N} e^{ikx} = 1 + 2 \sum_{k=1}^{N} \cos(kx) = \frac{\sin((N + 1/2)x)}{\sin(x/2)}.$$

Diskrete Fouriertransformation

Für positive N seien

$$x_j = jh = j\tfrac{2\pi}{N}, \qquad 0 \le j \le N - 1$$

die Fourier-Interpolationspunkte welche ein äquidistantes Gitter auf $[0, 2\pi)$ bilden. Die Fourierkoeffizienten in (1.1) könne im Regelfall nicht exakt ausgewertet werden, also greifen wir auf eine Quadraturformel zurück. Einfach zu berechnen und auch recht genau für 2π-periodische Funktionen ist die Rechteckregel

$$\frac{1}{2\pi}\int_0^{2\pi} v(x)dx \approx \frac{1}{N}\sum_{j=0}^{N-1} v(x_j), \qquad \forall v \in C[0,2\pi),$$

welche exakt ist für alle $u \in \mathsf{span}\left\{e^{ikx} : 0 \le |k| \le N-1\right\}$ [21].
Anwendung der Quadraturformel auf (1.1) ergibt die Approximationen

$$\hat{u}_k \approx \widetilde{u}_k := \frac{1}{N}\sum_{j=0}^{N-1} u(x_j)e^{-ikx_j}, \qquad k = 0, \pm 1, \ldots. \tag{1.3}$$

Man beachte dass $\widetilde{u}_k$ N-periodisch ist, das heißt

$$\widetilde{u}_{k\pm N} := \frac{1}{N}\sum_{j=0}^{N-1} u(x_j)e^{-i(k\pm N)x_j} = \frac{1}{N}\sum_{j=0}^{N-1} u(x_j)e^{-i(k)x_j}e^{\mp 2\pi ij} = \widetilde{u}_k$$

und damit gilt insbesondere für gerade N

$$\widetilde{u}_{-N/2} = \widetilde{u}_{N/2}. \tag{1.4}$$

Außerdem folgt für gerade N, dass auf dem Gitter $\{x_j\}_{j=0}^{N-1}$ nicht zwischen den Moden mit $k = \pm N/2$ unterschieden werden kann, da

$$e^{iNx_j/2} = e^{ij\pi} = e^{-ij\pi} = e^{-iNx_j/2}, \qquad 0 \le j \le N-1. \tag{1.5}$$

Um es einfacher zu halten, ist es für die diskrete Fouriertranformation sinnvoll ein gerades N zu benutzen und eine symmetrische endliche Menge von Wellenzahlen $-N/2 \le k \le N/2$. Wegen (1.4) und (1.5) muss daher die Approximation (1.3) umformuliert werden zu

$$\widehat{u}_k \approx \widetilde{u}_k := \frac{1}{Nc_k}\sum_{j=0}^{N-1} u(x_j)e^{-ikx_j}, \qquad k = -N/2, \ldots, N/2, \tag{1.6}$$

wobei $c_k = 1$ für $|k| < N/2$ und $c_k = 2$ für $k = \pm N/2$. Dieser Ausdruck ist die Diskrete Fouriertransformation (DFT).
Wir benutzen nun die Darstellung (1.2) als Interpolationspolynom. Mit

$$\mathcal{I}_N = \left\{ u = \sum_{k=-N/2}^{N/2} \widetilde{u}_k e^{ikx} : \widetilde{u}_{-N/2} = \widetilde{u}_{N/2} \right\}$$

sei $I_N : C[0,2\pi) \to \mathcal{I}_N$ der Interpolationsoperator definiert durch

$$(I_N u)(x) = p(x) = \sum_{k=-N/2}^{N/2} \widetilde{u}_k e^{ikx}. \tag{1.7}$$

Das folgende Lemma zeigt, dass $p(x_j) = u(x_j)$, das heißt die Interpolation ist exakt an den Interpolationspunkten.

Lemma 1.1: Für beliebiges $u \in C[0, 2\pi)$ gilt

$$p(x) = \sum_{j=0}^{N-1} u_j(x_j) h_j(x),$$

wobei
$$h_j(x) = \frac{1}{N} \sin\left[N\frac{x - x_j}{2}\right] \cot\left[\frac{x - x_j}{2}\right] \in \mathcal{I}_N \tag{1.8}$$

und dabei wird $h_j(x_k) = \delta_{jk}$ für alle $j, k = 0, 1 \ldots, N-1$ erfüllt.

Beweis: vgl. auch [21]. Mit Hilfe von (1.6) und (1.7) ergibt sich zunächst

$$\begin{aligned} p(x) &= \sum_{k=-N/2}^{N/2} \left(\frac{1}{N c_k} \sum_{j=0}^{N-1} u(x_j) e^{-ikx_j} \right) e^{ikx} \\ &= \sum_{j=0}^{N-1} \left(\frac{1}{N} \sum_{k=-N/2}^{N/2} \frac{1}{c_k} e^{ik(x-x_j)} \right) u(x_j) \\ &=: \sum_{j=0}^{N-1} h_j(x) u(x_j). \end{aligned}$$

Unter Verwendung des Dirichlet-Kerns und der Additionstheoreme rechnet man weiter

$$\begin{aligned} h_j(x) &= \frac{1}{N} \sum_{k=-N/2}^{N/2} \frac{1}{c_k} e^{ik(x-x_j)} \\ &= \frac{1}{N} \left(\sum_{k=-N/2+1}^{N/2-1} e^{ik(x-x_j)} \right) + \frac{1}{N} \left(\frac{1}{2} e^{iN/2(x-x_j)} + \frac{1}{2} e^{-iN/2(x-x_j)} \right) \\ &= \frac{1}{N} \left(\mathcal{D}_{N/2-1}(x - x_j) + \cos\left[N\frac{(x-x_j)}{2}\right] \right) \\ &= \frac{1}{N} \left(\frac{\sin[(N-1)\frac{x-x_j}{2}]}{\sin[\frac{x-x_j}{2}]} + \cos\left[N\frac{(x-x_j)}{2}\right] \right) \end{aligned}$$

$$= \frac{1}{N}\left(\frac{\sin[(N-1)\frac{x-x_j}{2}]}{\sin[\frac{x-x_j}{2}]} + \cos\left[N\frac{(x-x_j)}{2}\right]\right). \tag{1.9}$$

Schreibt man $h_j(x)$ wie in (1.9) dann ist wegen (1.5) $h_j(x) \in \mathcal{I}_N$. Des Weiteren ist leicht zu sehen, dass $h_j(x) = 0$ für $i \neq j$ und setzt man in (1.9) $x = x_j$ so erhält man $h_j(x) = 1$.

□

Aus der Interpolationsformel (1.7) und $p(x_j) = u(x_j)$ erhält man sofort die inverse diskrete Fouriertransformation

$$u(x_j) = \sum_{k=-N/2}^{N/2} \widetilde{u}_k e^{ikx_j}, \qquad j = 0, 1 \ldots, N-1. \tag{1.10}$$

Wegen (1.4) und (1.5) ist $\widetilde{u}_{-N/2} \cdot e^{-iNx_j/2} = \widetilde{u}_{N/2} \cdot e^{iNx_j/2}$, doch wegen dem Vorfaktor $1/2$ aus (1.6) kommt dieser Summand praktisch nur einmal in der Summe vor, so kann man die (inverse) DFT auch für $k = N/2 - 1, \ldots, N/2$ aufschreiben.

Schnelle Fouriertransformation

Die Diskrete Fouriertransformation und ihre Inverse kann mittels Matrix-Vektor-Multiplikation berechnet werden, fordert jedoch $O(N^2)$ Rechenoperationen. Mit der schnellen Fouriertransformation nach Cooley and Tukey kann dies jedoch in nur $O(N \log_2 N)$ Operationen bewerkstelligt werden.
Mit der Substitution $w = \exp(-i2\pi/N)$ lässt sich (1.6) schreiben als

$$\widetilde{u}_k = \sum_{j=0}^{N-1} u(x_j) w^{kj}, \qquad k = -N/2 - 1, \ldots, N/2, \tag{1.11}$$

wobei N wieder als gerade vorausgesetzt wird. Hier zunächst ohne den Faktor $1/N$, dieser kann jedoch ohne großen Aufwand zur Transformation hinzugefügt werden. Die inverse Transformation lautet also dann

$$u(x_j) = \sum_{k=-N/2+1}^{N/2} \widetilde{u}_k e^{ikx} \qquad\qquad j = 0, 1 \ldots, N-1.$$

Die Grundidee ist jetzt, die Summe in zwei Teilsummen aufzuteilen, nach geraden und ungeraden Komponenten:

$$y_j = u(x_{2j}) \qquad\qquad z_j = u(x_{2j-1}) \qquad\qquad j = 1 \ldots, N/2,$$

mit ihren eigenen Transformationen

$$\widetilde{y}_k = \sum_{j=0}^{N/2} y_j w^{2kj} \qquad \text{und} \qquad \widetilde{z}_j = \sum_{j=0}^{N/2} z_j w^{2kj} \qquad k = 1\ldots, N/2.$$

Mit einer Matrixmultiplikation bräuchte man jeweils $(N/2)^2$ Operationen um diese beiden Transformationen zu bilden. Könnte man die gesuchte Transformation aus diesen beiden zusammensetzen, dann bräuchte man bloß $2(N/2)^2 = N^2/2$ Operationen, hätte also den Rechenaufwand halbiert. Es geht aber sogar noch besser. Dafür schreibt man (1.11) als

$$\begin{aligned}\widetilde{u}_k &= \sum_{j=1}^{N/2} u(x_{2j}) w^{k(2j)} + u(x_{2j-1}) w^{k(2j-1)} \\ &= \widetilde{y}_k + w^{-k}\widetilde{z}_k \qquad\qquad k = 1\ldots, N/2.\end{aligned}$$

Es fehlt noch die andere Hälfte der Transformation, das heißt für $k = -N/2+1, \ldots, 0$. Dafür nutzt man aus, dass $w^{-k+N/2} = -w^k$ und erhält damit

$$\widetilde{u}_{k-N/2} = \widetilde{y}_k - w^k \widetilde{z}_k, \quad k = 1, \ldots, N/2.$$

Somit lässt sich eine Transformation der Länge N aus zwei Transformationen der Länge $N/2$ berechnen. Für diese geht man dann genauso vor und setzt sie zusammen aus Transformationen der Länge $N/4$, und so weiter. Ist N eine Zweierpotenz, dann kommt man am Ende bei der trivialen Transformation der Länge eins an, so dass man die gesuchte Transformation der Länge N rekursiv aus den kürzeren erhält. Etwas genauer, ist $N = 2^n$ so fängt man mit 2^n Transformationen der Länge 1 an, die dann zu 2^{n-1} Transformationen der Länge 2 zusammengefügt werden, mit jeweils 2 Operationen. Dies führt man fort bis zur Transformation der Länge 2^n, zusammengeführt aus 2 Transformationen der Länge 2^{n-1}. Auf jeder Stufe s benötigt man also 2^{n-s} Zusammenführungen mit jeweils 2^s Operationen, somit $2^{n-s} \cdot 2^s = 2^n$ Operationen pro Stufe. Weil die Anzahl der Stufen n beträgt, kommt man insgesamt auf einen Rechenaufwand von $2^n n = N \log_2 N$ Operationen und das ganz ohne Matrixmultiplikation. Ist dazu noch u eine reelle Funktion, dann gilt $\widetilde{u}_{-k} = \overline{\widetilde{u}}_k$ und somit müssen nur die Hälfte der Koeffizienten der inversen Transformation gespeichert werden.
In allgemeineren Fällen wird in einzelne Transformationen mit jeweils der Länge der Primfaktoren von N zerlegt. Wobei dieses Vorgehen stets weniger Zeitersparnis bringt, je größer die Primfaktoren sind [4,12].
In MATLAB kann die FFT und ihre Inverse (iFFT) mit integrierten Softwarepaketen berechnet werden. Mit gegebenen Daten $\{u(x_j)\}$, gespeichert in einem Vektor v, gibt die Funktion $\widetilde{v} = \mathsf{fft}(v)$ den Vektor mit den Frequenzen aus Formel (1.6) zurück. Umgekehrt erhält man mit der Funktion $\mathsf{ifft}(\widetilde{v})$ die physikalischen Werte aus Formel (1.10). Zu beachten ist hierbei, dass die Elemente des Frequenzvektors in der Reihenfolge $k = 0, 1, \ldots, N/2, -N/2+1, \ldots, -1$ gespeichert werden.

Fourier spektrale Ableitung

Mit Hilfe der Fouriertransformation soll die diskrete Ableitung einer Funktion $u(x)$ bestimmt werden. Dafür nutzen wir die Darstellung aus Lemma 1.1 und schreiben

$$p(x) = \sum_{j=0}^{N-1} u_j(x_j) h_j(x), \tag{1.12}$$

durch bilden der m-ten Ableitung erhält man

$$p^{(m)}(x) = \sum_{j=0}^{N-1} u_j(x_j) h_j^{(m)}(x).$$

Dies kann man auch als Matrix-Vektor-Multiplikation aufschreiben

$$\mathbf{u}^{(m)} = D^{(m)}\mathbf{u}, \qquad m \geq 0,$$

wobei $D^{(m)} = (d_{kj}^{(m)} := h_j^{(m)}(x_k))_{k.j=0,\ldots,N-1}$ (1.13)

und $\mathbf{u} = (u(x_0), u(x_1).\ldots, u(x_{N-1}))^T$,

$\mathbf{u}^{(m)} = (u^{(m)}(x_0), u^{(m)}(x_1), \ldots, u^{(m)}(x_{N-1}))^T$

Lemma 1.2: Die Ableitungsmatrix erster Ordnung ist durch die folgenden Einträge gegeben

$$d_{kj}^{(1)} = h_j'(x_k) = \begin{cases} \frac{(-1)^{k+j}}{2}\cot(\frac{(k-j)\pi}{N}), & \text{falls} \quad k \neq j, \\ 0, & \text{falls} \quad k = j, \end{cases}$$

Beweis: vgl. [21]. Ableiten von (1.9) gibt

$$h_j'(x) = \frac{1}{2}\cos\left[N\frac{x-x_j}{2}\right]\cot\left[\frac{x-x_j}{2}\right] - \frac{1}{2N}\sin\left[N\frac{x-x_j}{2}\right]\csc^2\left[\frac{x-x_j}{2}\right].$$

Für $k \neq j$ wird der Sinus im rechten Term null und der linke Term wird offensichtlich zu $\frac{(-1)^{k+j}}{2}\cot(\frac{(k-j)\pi}{N})$. Für den Fall $k = j$ benutzen wir zur besseren Übersicht die Substitution $\theta = (x - x_j)/2$ und könne dann die obige Formel umschreiben zu

$$h_j'(x) = \frac{\cos(N\theta)\cos(\theta)\sin(\theta) - N^{-1}\sin(N\theta)}{2\sin^2(\theta)}.$$

Die einzelnen Terme im Zähler und Nenner mit einer Taylor-Entwicklung an der Stelle 0 dargestellt ergeben

$$\cos(N\theta)\cos(\theta)\sin(\theta) = \theta + O(\theta^3), \qquad N^{-1}\sin(N\theta) = \theta + O(\theta^3)$$

und

$$2\sin^2(\theta) = 2 + O(\theta^3).$$

Da für $x \to x_j$ auch $\theta \to 0$, folgt damit $\lim_{x \to x_j} h_j'(x) = 0$.

□

Zur besseren Unterscheidung zur später verwendeten Chebyshev-Ableitungsmatrix schreiben wir die Fourier-Ableitungsmatrix als $D_F = D^{(1)}$.
D_F ist eine reelle schiefsymmetrische Matrix, da $\cot(-x) = -\cot(x)$ und $d_{kk} = 0$, und eine Toeplitzmatrix, da $d_{kj} = d_{k+1,j+1}$. Diese beiden Eigenschaften vereinfachen das Erstellen der Matrix während der Programmierung. Die paarweise verschiedenen Eigenwerte von D_F sind $\{ik : k = N/2 + 1, \ldots, N/2 - 1\}$ und der Eigenwert 0 hat die Vielfachheit 2.
Ähnlich wie die Matrix D_F leitet man auch $D_F^{(2)}$ oder allgemeiner $D_F^{(m)}$ her. Es ist hier zu betonen, dass nicht einfach $D_F^{(2)} = D_F^2$ gilt.
Der Nachteil dieser Methode ist, dass die Matrixmultiplikation $O(N^2)$ Rechenoperationen benötigt. Eine effizientere Methode zur Ableitung erfolgt mit Hilfe der oben formulierten FFT, die mit einem Aufwand von nur $O(N \log_2 N)$ Operationen möglich ist.
Eine Funktion wie in (1.12) kann man wie schon gezeigt auch in der Form

$$p(x) = \sum_{k=-N/2}^{N/2} \widetilde{u}_k e^{ikx}$$

schreiben wobei wieder $\widetilde{u}_{-N/2} = \widetilde{u}_{N/2}$ gilt. Die Ableitung in den Punkten $x_j = \frac{2\pi j}{N}$ ist somit

$$p'(x_j) = \sum_{k=-N/2}^{N/2} ik\widetilde{u}_k e^{ikx_j},$$

für $j = 0, 1 \ldots, N-1$. In Kurzform kann man das Vorgehen zur Implementierung folgenderweise aufschreiben [25]:

- Mit gegebenen Eingangsdaten $\{u(x_j)\}$, berechne mittels FFT $\{\widehat{u}_k\}$.
- Setze $\widetilde{w}_k = ik\widetilde{u}_k$, da hier die Symmetrie $\widetilde{w}_{N/2} = -\widetilde{w}_{N/2}$ nicht gegeben ist, setze $\widetilde{w}_{N/2} = 0$.
- Berechne mittels iFFT $\{w_j\}$ von $\{\widetilde{w}_k\}$.

Für höhere Ableitungen funktioniert das sehr ähnlich, die m-te Ableitung erfolgt einfach durch eine Multiplikation durch $(ik)^m$, der zweite Schritt muss also ersetzt werden durch

- Setze $\widetilde{w}_k = (ik)^m\widetilde{u}_k$, mit Ausnahme $\widetilde{w}_{N/2} = 0$ falls m ungerade ist.

Chebyshev spektrale Ableitung

Die bisher betrachteten spektralen Ableitungen bezogen sich auf (2π-)periodische Funktionen. Für nicht periodische Probleme eignet sich eine andere Art von spektraler Ableitung.
Angenommen es liegt eine nicht periodische Funktion auf dem Intervall $[-1, 1]$ vor. Mit einer trigonometrischen Interpolation auf einem äquidistanten Gitter zu arbeiten, also die Fouriermethode wie im vorigen Abschnitt, könnte hier zu erheblichen Problemen führen. In Fällen bei denen die Lösung am Rand exponentiell auf null oder einen Konstanten Wert fällt, würde dieser Ansatz noch funktionieren, ansonsten führt die Nichtperiodizität zu Schwierigkeiten. Nimmt man beispielsweise ein glatte Funktion die periodisch erweitert wird, erhält man eine stückweise glatte Funktion die an den Periodenübergängen Sprungstellen aufweist. Aufgrund des Gibbs-Phänomens kommt es zu großen Ungenauigkeiten an den Sprungstellen, für die Interpolante als auch für ihre Ableitungen.
Eine Alternative wäre somit statt trigonometrische Poylnome, algebraische also von der Form $p(x) = a_0 + a_1 x + \cdots + a_N x^N$ zu benutzen. Diese auf ein äquidistantes Gitter anzuwenden führt jedoch zu dem bekannten Problem des Runge-Phänomens, welches sich dadurch auszeichnet, dass bei höheren Polynomgraden starken Oszillationen insbesondere am Rand auftreten. Um diese Unannehmlichkeit auch noch zu umgehen, wählt man ein Gitter mit nicht gleichmäßigen Abständen, dessen Punkte sich an den Rändern häufen. Es gibt dazu verschiedene solcher Gitter, sie werden normalerweise so gewählt, dass asymptotisch für $N \to \infty$ etwa $N/(\pi\sqrt{1-x^2})$ Punkte pro Längeneinheit vorliegen. Das heißt insbesondere, dass der mittlere Abstand zwischen Punkten nahe bei $x \approx \pm 1$ von der Ordnung $O(N^{-2})$ ist und $O(N^{-1})$ im Inneren, wobei der mittlere Abstand zwischen benachbarten Punkten in der Nähe von $x = 0$ asymptotisch an π/N liegt [25]. Die hier benutzten Punkte, welche die oben genanten Gittereigenschaften erfüllen, sind die Chebyshev-Punkte

$$x_j = \cos(j\pi/N), \qquad\qquad j = 0, \ldots, N,$$

welche auch Chebyshev-Extrempunkte genannt werden, da sie die Extremstellen des Chebyshev-polynoms $T_N(x) = \cos(N\theta)$ auf $[-1, 1]$ sind.
Betrachte man nun eine diskrete Funktion v definiert auf einem Gitter bestehend aus Chebyshev-Punkten. Weiter sie p das eindeutige Polynom vom Grad $\leq N$ mit $p(x_j) = v_j$. Eine diskrete Ableitung w erhält man nun durch die Ableitung von p in den Punkten x_j, also $w_j = p'(x_j)$. Da diese Operation linear ist, kann sie durch eine Multiplikation mit einer $(N+1) \times (N+1)$-Matrix repräsentiert werden. Anders als im Abschnitt über Fourier spektrale Ableitungen kann hier N auch ungerade sein. Für p wählen wir die Lagrangeinterpolation, das heißt

$$p(x) = \sum_{j=0}^{N} u(x_j) p_j(x), \qquad \text{mit} \qquad p_j(x) = \prod_{i \neq j}^{N} \frac{x - x_i}{x_j - x_i}\,.$$

Die Ableitung von p ist also

$$p'(x) = \sum_{j=0}^{N} u(x_j) p'_j(x).$$

Ähnlich wie in (1.13) kann dies mit einer Matrix-Vektor-Multiplikation ausgedrückt werden

$$\mathbf{u}' = D_C \mathbf{u}$$

mit
$$D_C = (d_{kj} := p'_j(x_k))_{k.j=0,\ldots,N+1},$$
$$\mathbf{u} = (u(x_0), u(x_1). \ldots, u(x_{N-1}))^T,$$
$$\mathbf{u}' = (u'(x_0), u'(x_1), \ldots, u'(x_{N-1}))^T.$$

Wir benutzen die Notation D_C um von der Fourier-Ableitungsmatrix D_F zu unterscheiden.
Die Einträge der Matrix D_C sehen dabei wie folgt aus [25]:

$$(D_C)_{00} = \frac{2N^2+1}{6}, \qquad (D_C)_{NN} = -\frac{2N^2+1}{6},$$
$$(D_C)_{jj} = -\frac{x_j}{2(1-x_j^2)}, \qquad j = 1, \ldots, N-1, \tag{1.13}$$

$$(D_C)_{kj} = -\frac{c_k(-1)^{k+j}}{c_j(x_k - x_j)}, \qquad k \neq j, \qquad k, j = 0, \ldots, N,$$

wobei
$$c_k = \begin{cases} 2, & k = 0 \text{ oder } N, \\ 1, & \text{sonst} \end{cases}.$$

Ein Weg um diese Matrix in MATLAB zu erstellen, könnte dann so erfolgen (vgl. [25]):

```
function [D,x] = cheb(N)

if N==0, D=0; x=1; return, end
x = cos(pi*(0:N)/N)';
c = [2; ones(N-1,1); 2].*(-1).^(0:N)';
X = repmat(x,1,N+1);
dX = X-X';
D = (c*(1./c)')./(dX+(eye(N+1)));      % Nebendiagonaleinträge
D = D - diag(sum(D'));                 % Diagonaleinträge
end
```

Die Funktion cheb( ) nimmt die Zahl N als Input und gibt ein Gitter mit $N+1$ Chebyshev-Punkten und die zugehörige Ableitungsmatrix zurück. Die Diagonaleinträge werden hier jedoch nicht einfach aus (1.13) bestimmt, sondern aus der Gleichung

$$(D_C)_{kk} = -\sum_{j \neq k}^{N} (D_C)_{kj},$$

wodurch das Programm etwas vereinfacht wird und sich die Genauigkeit für die Ableitung erhöht [3]. Anders als bei der Fouriermethode, ist es für höhere Ableitungen möglich, die entsprechende Potenz der Matrix zu verwenden. So kann man einfach die zweite Ableitung mit Hilfe der Matrix D_N^2 berechnen was $O(N^3)$ Rechenoperationen erfordert, alternativ benutzt man explizite Formeln von der Art wie in (1.13) oder Rekursionen welche mit einem Aufwand von $O(N^2)$ Operationen auskommen [25].
Eine einfache Möglichkeit um mit einer Chebyshev-Spektralmethode homogene Dirichlet-Randbedingungen zu behandeln, verläuft nach folgendem Schema. Angenommen wir wollen ein Randwertproblem der Art

$$u_{xx} = f(x), \quad x \in (-1, 1), \quad u(\pm 1) = 0,$$

lösen, wobei u_{xx} die zweite räumliche Ableitung bezeichnet. Nun nimmt man nur die inneren Chebyshev-Punkte als Gitter und wieder ein Polynom p mit Grad $\leq N$ und mit $p(x_j) = v_j$ für $j = 1, \ldots, N-1$ und setzt $w_j = p''(x_j)$. Da nun die Matrix D_C^2 eine $(N+1) \times (N+1)$-Matrix ist, die einen Vektor $v = (v_0, \ldots, v_n)^T$ auf einen Vektor $w = (w_0, \ldots, w_n)^T$ abbildet, müssen demnach die Komponenten v_0 und v_N null gesetzt werden und w_0 und w_N ignoriert werden. Ersteres hat zur Folge, dass die erste und letzte Spalte von D_C^2 keinen Effekt haben, da sie mit null multipliziert werden, man kann also diese beiden Spalten der Matrix einfach weglassen. Will man die erste und letzte Komponente von w ignorieren, zieht man einfach die erste und letzte Reihe von D_C^2 heraus. Insgesamt bleibt eine $(N-1) \times (N-1)$-Matrix übrig und es wird jetzt nur noch der Vektor $(v_1, \ldots, v_{N-1})^T$ auf $(w_1, \ldots, w_{N-1})^T$ abgebildet [25].

Chebyshev spektrale Ableitung mittels FFT

Um die Berechnungsgeschwindigkeit zu erhöhen, kann ähnlich wie bei Fourier spektralen Ableitungen die Ableitung im Frequenzraum mit Hilfe einer FFT erfolgen. Um dies für Chebyshev-Polynome zu bewerkstelligen, benötigt man eine Transformation auf äquidistante Stützstellen um nach einer Ableitung im Frequenzraum wieder auf das Chebyshev-Gitter zurück zu transformieren.
Dabei nutzt man aus, dass

$$\sum_{n=0}^{\infty} a_n T_n(x) = \sum_{n=0}^{\infty} a_n \cos(n\theta) = \frac{1}{2} \sum_{n=0}^{\infty} a_n (e^{i\Theta} + e^{-i\Theta}) \tag{1.14}$$

mit $x = \cos(\theta)$ und $T_n(x) = \cos(n\theta)$ gilt. Die Koeffizienten a_n sollen mit einer FFT bestimmt werden. Betrachten wir weiter die Approximationspolynome

$$P(\Theta) = \sum_{n=0}^{N} a_n \cos(n\theta) \quad \text{und} \quad p(x) = \sum_{n=0}^{N} a_n T_n(x),$$

wobei P eine Funktion $F(\Theta)$ definiert auf $\mathbb{R}$ approximiert, und p eine Funktion f, definiert auf $x \in [-1.1]$. Des Weiteren sei $F(\Theta) = f(cos\Theta)$. p abgeleitet ergibt

$$p'(x) = \frac{P'(\Theta)}{dx/d\Theta} = \frac{-\sum_{n=0}^{N} na_n \sin(n\theta)}{-\sin(\theta)} = \frac{\sum_{n=0}^{N} na_n \sin(n\theta)}{\sqrt{1-x^2}}. \tag{1.15}$$

Für $x = \pm 1$ berechnet man mit der Regel von l'Hospital die Werte

$$p'(1) = \sum_{n=0}^{N} n^2 a_n \qquad \text{und} \qquad p'(-1) = \sum_{n=0}^{N} (-1)^{n+1} n^2 a_n. \tag{1.16}$$

Die Vorgehensweise sieht nun wie folgt aus [25]:

- Mit gegebenen Daten $u(x_0), \ldots, u(x_N)$ an entsprechenden Chebyshev-Punkten, erweitert man diese Daten auf eine Anzahl von $2N$ durch hinzufügen der Werte $u(x_{2N-j}) = u(x_j)$ für $j = 1, \ldots, N-1$.
- Jetzt haben wir $2N$ äquidistante Punkte θ_j auf $[0, 2\pi]$ und können mit einer FFT die Koeffizienten $\widetilde{u}_k,\ k = -N+1, \ldots, N$ berechnen.
- Definiere nun $\widetilde{W}_k = ik\widetilde{u}_k$, mit der Ausnahme $\widetilde{W}_N = 0$.
- Mit der inversen FFT erhält man jetzt die Ableitung $W_j,\ j = 0, \ldots, 2N-1$ der Interpolanten P auf dem äquidistanten Gitter.

Mit Hilfe von (1.15) und (1.16) bestimmt man die Ableitung der Interpolanten p auf dem Chebyshev-Gitter:

$$w_j = \frac{W_j}{\sqrt{1-x_j^2}}, \qquad j = 1, \ldots, N-1, \qquad \text{für die inneren Punkte,}$$

und

$$w_0 = \sum_{n=0}^{N} n^2 a_n, \qquad w_N = \sum_{n=0}^{N} (-1)^{n+1} n^2 a_n \qquad \text{für die Endpunkte.}$$

Auf ähnliche Weise kann man höhere Ableitungen berechnen. Im dritten Schritt wird allgemein für die m-te Ableitung mit $(ik)^m$ multipliziert und für ungerade m setzt man $\widehat{W}_N = 0$ und für das weitere Vorgehen sind wieder Formeln notwendig um die Ableitungen auf dem äquidistanten und dem Chebyshev-Gitter in Beziehung zu setzen. Die zweiten Ableitungen sind z.B. verknüpft durch

$$p''(x) = \frac{x}{(1-x^2)^{3/2}} P'(\Theta) + \frac{1}{1-x^2} P''(\Theta).$$

Mit $W_j^{(2)}$ als zweite Ableitung auf dem äquidistanten Gitter, lautet die Formel für die zweite Ableitung auf dem Chebyshev-Gitter

$$w_j^{(2)}(x) = \frac{x_j}{(1-x_j^2)^{3/2}} W_j + \frac{1}{1-x_j^2} W_j^{(2)}, \qquad 1 \le j \le N-1.$$

Der Programmcode zu dem eben aufgeführten Algorithmus ist dann [25]:

```
 N = length(v)-1; if N==0, w=0; return, end
x = cos((0:N)'*pi/N);
k = 0:N-1;
v = v(:); V = [v; flipud(v(2:N))];          % transformiere x nach theta
U = real(fft(V));
W = real(ifft(1i*[k 0 1-N:-1]'.*U));
w = zeros(N+1,1);
w(2:N) = -W(2:N)./sqrt(1-x(2:N).^2);        % transformiere theta nach x
w(1) = sum(k'.^2.*U(k+1))/N + .5*N*U(N+1);
w(N+1) = sum((-1).^(k+1)'.*k'.^2.*U(k+1))/N + ...
.5*(-1)^(N+1)*N*U(N+1);
```

Zur Berechnung der Endpunkte w(1) und w(N+1) werden die Summen jeweils durch N geteilt, da MATLAB die Fourierkoeffizienten etwas anders ausgibt als in (1.6) definiert.

Eigenwerte der Ableitungsmatrizen

Da, wie sich im darauffolgenden Kapitel zeigen wird, die Eigenwerte der Diskretisierungsmatrix hilfreich für die Wahl der Zeitschrittweite ist, sollen in diesem Abschnitt die Eigenwerte der eben besprochenen Matrizen D_F, D_C und D_C^2 bestimmt werden, wenn auch zum Teil nur näherungsweise.
Die Eigenwerte von D_F sind rein imaginär und gleichmäßig über das komplexe Intervall $[-i\pi N/2, i\pi N/2]$ verteilt. Für gerades N hat der Eigenwert null die Vielfachheit zwei. Die Eigenvektoren sind komplexe Exponentialfunktionen.
D_C besitzt Eigenwerte mit negativem Realteil, die meisten davon verlaufen entlang eines Bogens ausgehend vom Punkt $-iN$ nach iN. Ein paar der Eigenwerte liegen jedoch weit außerhalb dieses Bogens und sind von der Größenordnung $O(N^2)$ für $N \to \infty$, ihre Anzahl steigt mit wachsendem N (siehe *Abb.5.5* im fünften Kapitel). Die Eigenvektoren zu diesen Ausreißern werden von starken Oszillationen dominiert. Dies erschwert die Zeitschrittwahl für Chebyshev Spektralmethoden [28].
Die Bestimmung der Eigenwerte der Chebyshev-Ableitungsmatrix zweiter Ordnung D_C^2 gestaltet sich etwas schwieriger. Genauer betrachten wir die Matrix D_C^2 nach Streichung der erste und letzten Zeile und Spalte, wie zur Behandlung von homogenen Randwertaufgaben. Die Eigenwerte des stetigen Ableitungsoperators zweiter Ordnung der Gleichung

$$\begin{aligned} &D^2 u(x) = \lambda u(x), \qquad -1 \le x \le 1, \\ &u(\pm 1) = 0, \end{aligned}$$

sind

$$\lambda_k = -\frac{k^2\pi^2}{4}, \qquad k = 1, 2, 3, \ldots$$

mit den zugehörigen normalisierten Eigenfunktionen [30]

$$u(x) = \begin{cases} \cos(\frac{1}{2}k\pi x), & k \text{ ungerade} \\ \sin(\frac{1}{2}k\pi x), & k \text{ gerade.} \end{cases}$$

Für eine Finite-Differenzen-Methode angewendet auf dieses Problem, sind die Eigenwerte der zugehörigen Matrix analytisch bestimmbar, für die spektrale Ableitungsmatrix kann man lediglich eine Abschätzung angeben. Hinzu kommt, dass für finite Differenzen und finite Elemente die Ableitungsmatrix normalerweise einen Spektralradius von $O(N^2)$ aufweist, für Spektralmethoden auf einem nicht periodischen Gebiet hingegen, eher von der Größenordnung $O(N^4)$. Dies hat zur Folge, dass für explizite Zeitintegrationsformeln sehr restriktive Zeitschrittwahlen der Ordnung $O(N^{-4})$ notwendig sind. In [7] wurde gezeigt, dass die Eigenwerte von D_C^2 reell, negativ und paarweise verschieden sind. Die kleinen Eigenwerte, numerisch bestimmt, sind nahe an denen des stetigen Problems. Diese Eigenwerte werden als „inliers“ bezeichnet, wobei die, welche die Eigenwerte des stetige Problems in keinster Weise approximieren, als „outliers“ (Ausreißer) bezeichnet werden. Die größten outliers wachsen sehr schnell von der Ordnung $O(N^4)$ und erzwingen die strikte Restriktion. Der Anteil der Eigenwerte die inliers sind, beträgt in etwa $2N/\pi$, bzw. nähert sich $2/\pi$ für $N \to \infty$. Die zugehörigen Eigenvektoren der inliers sind gute Approximationen von Sinus- und Kosinusfunktionen, wogegen die der outliers starken Oszillationen an den Rändern unterliegen. Eine erste Erklärung für dieses Verhältnis ist, dass für $k > 2N/\pi$ die Sinus- und Kosinusfunktionen, welche die Eigenfunktionen des stetigen Problems darstellen, weniger als zwei Punkte pro Wellenlänge im Zentrum des Chebyshev-Gitters zur Interpolation haben, wo die Abstände zwischen den Gitterpunkten etwa π/N beträgt. Im grobmaschigen Zentrum des Gitters können sie also nicht durch Polynominterpolation aufgelöst werden. Die zu den Ausreißern gehörenden Eigenvektoren gehören zu den nicht physikalischen Eigenmoden welche keine globalen Sinus- oder Kosinusfunktionen sind, aber stark in den Bereichen für $x = \pm 1$ konzentriert sind. *Abb.1.1* zeigt dieses Verhalten.
Für die Ermittlung von Stabilitätsrestriktionen, wäre die Kenntnis des größten Eigenwertes von D_C^2 hilfreich. Eine Abschätzung für den Spektralradius $\rho(D_C^2)$ ist [30]:

$$\limsup_{N\to\infty} \frac{\rho(D_C^2)}{N^4} \leq \sqrt{\frac{11}{4725}} \approx 0.0482, \tag{1.17}$$

was sich auch recht gut mit numerischer Bestimmung vereinbaren lässt. In *Abb.1.2* sind die Beträge der Eigenwerte der Größe nach in einer logarithmischen Skala eingetragen, die outliers sind gut von den inliers zu unterscheiden. Die Eigenwerte und Eigenvektoren sind mit der MATLAB-Funktion eig() bestimmt worden.

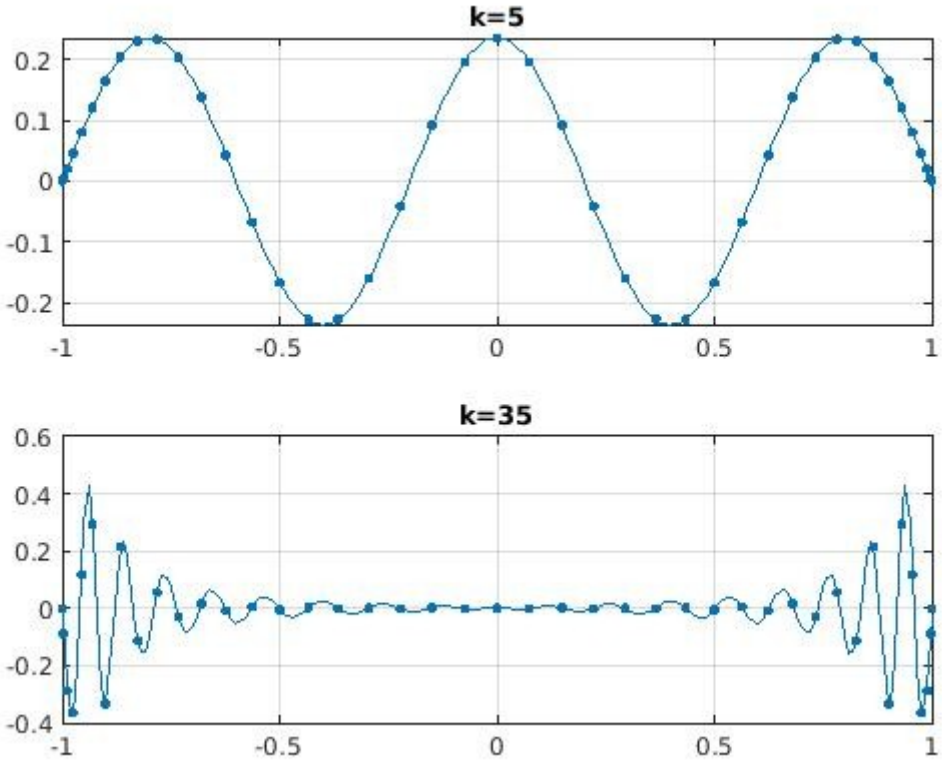

Abb.1.1: Interpolation der Eigenvektoren zum 5. und 35. Eigenwert (betragsmäßig von klein nach groß sortiert).

Eine genauere Begründung für den Interpolationsfehler der großen Eigenwerte und dem kritischen Wert $2\pi/N$ ergibt sich aus folgender Beobachtung. Sei $f_N(x) = e^{i\alpha N x}$ definiert auf $[-1, 1]$. Sei weiter $p_N(x)$ ein Polynom vom Grad N, das $f_N(x)$ an den Chebyshev-Punkten interpoliert. Dann ist eine ausreichende Bedingung für Konvergenz im Sinne von

$$\max_{-1\leq 1} |f_N(x) - p_N(x)| \to 0 \qquad \text{für} \qquad N \to \infty,$$

dass $\alpha < 1$ gilt [30]. Im Fall der Eigenfunktionen ist $(1/2)k\pi = |\alpha|\, N$, also muss $(1/2)k\pi < N$ gelten oder umgeformt $k < 2N/\pi$. Des Weiteren entspricht dies einer minimalen Wellenlänge von $2\pi/N$. Da der mittlere Abstand zwischen den Gitterpunkten $2/N$ beträgt, sind also im Durchschnitt mindestens π Gitterpunkte pro Wellenlänge notwendig um die Eigenfunktionen ausreichend auflösen zu können.

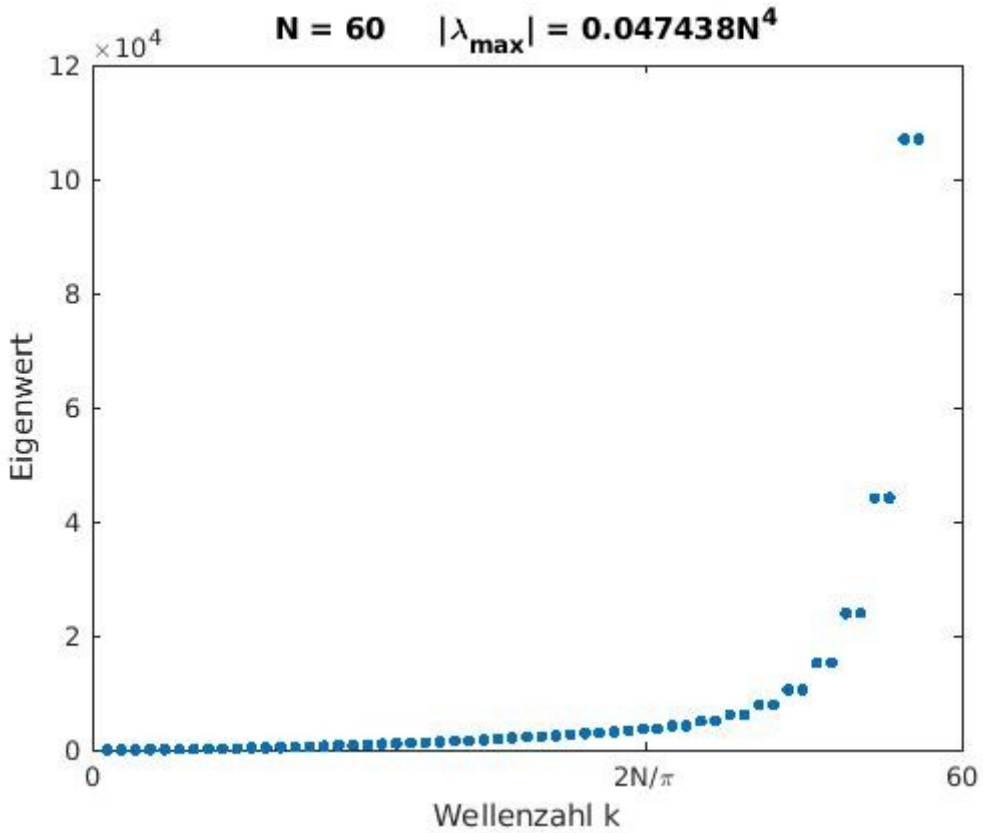

Abb.1.2: Die betragsmäßig nach Größe sortierten Eigenwerte von D_C^2. Der Anteil der Ausreißer befindet sich rechts von der Wellenzahl $2N/\pi$.

Programmcode zu *Abb.1.2*

```
N = 60; [D,x] = cheb(N); D2 = D^2; D2 = D2(2:N,2:N);
e = eig(D2); [e,ii] = sort(-e);
e_max = max(e);
plot(e,'.','markersize',10)
title(['N = ' int2str(N) '     |\lambda_{max}| = ' num2str(max(e)/N^4) 'N^4'])
axis([0 N 0 120000]),
xticks([0 2*N/pi N])
xticklabels({'0','2N/\pi',int2str(N)})
ylabel Eigenwert, xlabel 'Wellenzahl k'
```

2 Stabilitätsgebiete

Wir betrachten im Folgenden Systeme gewöhnlicher Differentialgleichungen erster Ordnung

$$\begin{aligned} u'(t) &= f(t, u(t)), \\ \text{mit Anfangsbedingung} \quad u(t_0) &= u_0, \end{aligned} \tag{2.1}$$

dabei seien $u(t) = (u_1(t) \ldots, u_N(t))^T$ und $f(t, x) = (f_1(t, x) \ldots, f_N(t, x))^T$ Vektorfunktionen. Ausgehend von einem Anfangspunkt $(t_0, u_0) \in \mathbb{R}^1 \times \mathbb{R}^N$ werden Lösungen $u(t)$ auf einem Intervall $[t_0, T]$ gesucht die $u(t_0) = u_0$ erfüllen. Die Existenz und Eindeutigkeit vorausgesetzt, soll es nun im weiteren Verlauf darum gehen, zu beschreiben, wann ein numerisches Lösungsverfahren für (2.1) stabil ist, das heißt kurz gesagt, wann sich Fehler im Laufe des Verfahrens nicht exponentiell aufsummieren. Als numerische Lösungsmethode für (2.1) betrachten wir Ein- und Mehrschrittverfahren.

Einschrittverfahren

Die allgemeine Form eines Einschrittverfahrens sieht wie folgt aus [18]

$$u_{n+1} = u_n + hF(h, t_n, u_n, u_{n+1}), \tag{2.2}$$

wobei u_n die Näherungslösungen zu $u(t_n)$ bezeichnen, $h = t_n - t_{n-1}$ die Schrittweite und F ist die sogenannte Verfahrensfunktion. Wenn die Näherungslösung nur von der vorausgehenden Näherung u_n abhängt, wird diese Methode explizit genannt. Hängt sie auch noch von u_{n+1} ab, dann spricht man von einem impliziten Verfahren und es ist für jeden Schritt eine Lösung eines im Allgemeinen nichtlinearen Gleichungssystems erforderlich.
Das einfachste Einschrittverfahren ist das Euler-Verfahren mit $F(h, t_n, u_n, u_{n+1}) = f(t_n, u_n)$, also

$$u_{n+1} = u_n + hf(t_n, u_n) \tag{2.3}$$

Zur Veranschaulichung der numerischen Stabilität von Einschrittverfahren, betrachten wir zuerst ein skalares Testproblem als Spezialfall von (2.1)

$$\begin{aligned} u'(t) &= \lambda u(t) \qquad \lambda \in \mathbb{C} \\ u(0) &= 1, \end{aligned} \tag{2.4}$$

und wenden darauf das Euler-Verfahren an. Man erhält also Näherungen $u_{n+1} = (1 + h\lambda)u_n$, zunächst für $\lambda \in \mathbb{R}$ und $\lambda \leq 0$. Es ergibt sich induktiv $u_n = (1 + h\lambda)^n u_0$. Man sieht hier, dass für $|1 + h\lambda| > 1$ die diskrete Lösung exponentiell wächst, also muss die Schrittweite h so gewählt werden, dass $|1 + h\lambda| \leq 1$ oder äquivalent dazu $h \leq |2/\lambda|$.

Im Allgemeinen wird zur numerischen Stabilität von Differenzenverfahren zunächst rein intuitiv gefordert, dass für eine festes h im Falle $\sup_{t>0} \|u(t)\| < \infty$ auch $\sup_{n\geq 0} \|u_n\| < \infty$ erfüllt ist.
Die Lösung der Testgleichung (2.4) ist $u(t) = e^{\lambda t}$. Das Verhalten dieser Lösung ist für $t \to \infty$ durch das Vorzeichen von $Re(\lambda)$ bestimmt.

- Für $Re(\lambda) < 0$ folgt $|u(t)| = e^{\lambda t} \to 0$.
- Für $Re(\lambda) = 0$ ist $|u(t)| \equiv 1$.
- Für $Re(\lambda) > 0$ folgt $|u(t)| = e^{\lambda t} \to \infty$.

Also Beschränktheit wenn der Realteil von λ kleiner oder gleich null ist. Da für die Näherungslösungen das gleiche Verhalten gefordert wird, kommt man so auf folgende Definition.

Definition 2.1: Ein Einschrittverfahren heißt absolut stabil für ein $\lambda h \neq 0$, wenn es angewendet auf das Testproblem (2.4) für $Re(\lambda) \leq 0$ beschränkte Näherungen erzeugt, das heißt: $\sup_{n\geq 0} \|u_n\| < \infty$.

Für das Euler-Verfahren liegt also genau dann absolute Stabilität vor, wenn für die sogenannte Stabilitätsfunktion $\omega(\lambda h) := 1 + \lambda h$ die Bedingung $|\omega(\lambda h)| \leq 1$ gilt. Um zu wissen für welche h für ein festes λ mit $Re(\lambda) \leq 0$ das Verfahren absolut stabil ist, definiert man mit Hilfe der Stabilitätsfunktion das Stabilitätsgebiet.

Defintion 2.2: Das Stabilitätsgebiet S eines Einschrittverfahrens ist die Menge aller Punkte λh in der komplexen Ebene für welche das Verfahren absolut stabil ist, oder anders ausgedrückt, für die $\omega(\lambda h) \leq 1$ gilt.

Gilt $\{z \in \mathbb{C} : \mathrm{Re}(z) \leq 0\} \subset S$, das heißt ist die ganze linke Seite der komplexen Halbebene in S enthalten, dann heißt das Verfahren A-stabil. Man kann zeigen, dass explizite Verfahren nicht A-stabil sein können [18], implizite jedoch schon.

Lineare Mehrschrittverfahren

Im Gegensatz zu Einschrittmethoden werden bei linearen Mehrschrittverfahren in jedem Iterationsschritt mehr als ein Wert aus den vorherigen Schritten benötigt. Allgemein hat eine s-Schritt-Methode die Form [16]:

$$\sum_{j=0}^{s} \alpha_j u_{n+j} = h \sum_{j=0}^{s} \beta_j f(u_{n+j}, t_{n+j}). \tag{2.5}$$

Dabei wird u_{n+s} aus den vorherigen Werten $u_{n+s-1}, u_{n+s-2}, \ldots, u_n$ und den zugehörigen Funktionswerten von f berechnet. Wenn $\beta_s = 0$ gilt, dann ist die Methode explizit, sonst implizit. Eine Multiplikation beider Seiten mit einer Konstanten ergibt im Wesentlichen die selbe Methode,

nur die Koeffizienten α_j und β_j würden sich ändern. Bevor auf die Bedingungen für Stabilität von Mehrschrittverfahren eingegangen wird, die etwas komplizierter sind als bei Einschrittverfahren, müssen erst die Begriffe Konvergenz und Konsistenz geklärt werden.

Definition 2.3 (Konvergenz): Eine lineares Mehrschrittverfahren heißt konvergent, wenn für jede Anfangswertaufgabe (2.1) gilt

$$\max_{0\leq n\leq N} \|u_n - u(t_n)\| \to 0 \quad \text{für } h \to 0,$$

mit der Voraussetzung, dass die Startwerte $u_0, \ldots, u_{s-1}$ konvergieren, das heißt

$$\max_{0\leq n\leq s-1} \|u_n - u(t_0)\| \to 0 \quad \text{für } h \to 0.$$

Um Konsistenz von linearen Mehrschrittverfahren zu definieren, braucht man zuerst den lokalen Diskretisierungsfehler (Abschneidefehler), den man durch Einsetzen in die Differenzengleichung (2.5) erhält:

$$\tau^h(t_n) := h^{-1} \sum_{j=0}^{s} \alpha_j u(t_{n+j}) - \sum_{j=0}^{s} \beta_j f(t_{n+j}, u(t_{n+j})).$$

Definition 2.4 (Konsistenz): Eine lineares Mehrschrittverfahren heißt konsistent, wenn

$$\max_{t_n\in[t_0,T]} \left\|\tau^h(t_n)\right\| \to 0 \qquad \text{mit } h \to 0,$$

und ist von der Ordnung $p > 0$, wenn für hinreichend glatte u gilt, dass $\max_{t_n\in[t_0,T]} \left\|\tau^h(t_n)\right\| = O(h^p)$.

Für weitere Betrachtungen sind die beiden charakteristischen Polynome hilfreich:

$$\rho(z) = \sum_{j=0}^{s} \alpha_j z^j \qquad \text{und} \qquad \sigma(z) = \sum_{j=0}^{s} \beta_j z^j.$$

ρ ist ein Polynom vom Grad s, genauso σ falls die Methode implizit ist, sonst ist der Grad von σ kleiner als s.

Wie man an der allgemeinen Formel (2.5) sieht, sind s Startwerte $u_0, \ldots, u_{s-1}$ notwendig, bevor eine s-Schritt-Methode auf das Anfangswertproblem (2.1) angewendet werden kann. Von diesen Startwerten ist u_0 bereits durch den Anfangswert gegeben, aber die anderen $u_1, \ldots, u_{s-1}$ müssen noch berechnet werden, z.B. durch eine passende Einschrittmethode. In jedem Fall werden die Startwerte numerische Fehler beinhalten und es ist wichtig wie diese weitere Approximationen u_n, $n \geq s$ berechnet durch (2.5), beeinflussen. Daher wird eine Art von Stabilität der numerischen

Methode in Bezug auf kleine Störungen der Startwerte benötigt. Dies führt auf die Definition der Nullstabilität.

Definition 2.5 (Nullstabilität): Eine lineare s-Schritt-Methode für die gewöhnliche Differentialgleichung (2.1) heißt nullstabil, wenn eine Konstante K existiert, so dass für zwei Folgen (u_n) und (w_n) generiert durch dieselbe Formel, aber mit jeweils verschiedenen Startwerten $u_0, \ldots, u_{s-1}$ und $w_0, \ldots, w_{s-1}$ gilt

$$|u_n - w_n| \leq K \; max \left\{ |u_0 - w_0|, |u_1 - w_1|, \ldots, |u_{s-1} - w_{s-1}| \right\}$$

für $t \leq T$ und $h \to 0$.

Man kann zeigen, dass es ausreicht, wenn man nur das Verhalten der Methode angewendet auf $u' = 0$ betrachtet. Deshalb auch die Bezeichnung Nullstabilität [22]. Obwohl die obige Definition die intuitive Vorstellung, dass kleine Störungen der Startwerte große Störungen im Ergebnis hervorrufen können, gut wiedergibt, wäre es zu mühsam nur unter Verwendung dieser Definition eine lineare Mehrschrittmethode auf Nullstabilität zu prüfen. Daher bedient man sich lieber der sog. Wurzelbedingung. Hierfür wird erst ein Hilfsresultat benötigt.

Lemma 2.1: Gegeben sei die homogene lineare Rekursionsgleichung

$$\alpha_s u_{n+s} + \cdots + \alpha_1 u_{n+1} + \alpha_0 u_n = 0, \qquad n = 0, 1, 2 \ldots, \tag{2.6}$$

mit $\alpha_s \neq 0, \; \alpha_0 \neq 0, \; \alpha_j \in \mathbb{R}, \; j = 0, 1, \ldots, s$, und dem zugehörigen charakteristischen Polynom

$$\rho(z) = \alpha_s z^s + \cdots + \alpha_1 z + \alpha_0.$$

Seien weiter z_r mit $1 \leq r \leq l, \;\; l \leq s$, die verschiedenen Nullstellen des Polynoms ρ und m_r die Vielfachheit der z_r, mit $m_1 + \cdots + m_l = s$. Wenn eine Folge (u_n) von komplexen Zahlen die Bedingung (2.6) erfüllt, dann gilt

$$u_n = \sum_{r=1}^{l} p_r(n) z_r^n \qquad \text{für alle } \; n \geq 0, \tag{2.7}$$

wobei p_r ein Polynom in n ist vom Grad $\; m_r - 1$. Sind insbesondere alle Nullstellen einfach, d.h. $m_r = 1, \; 1 \leq r \leq s$, dann sind die p_r alle konstant.

Beweis: siehe [22].

Jetzt kann die wesentliche Bedingung zur Überprüfung der Nullstabilität formuliert werden.

Satz 2.1 (Wurzelbedingung): Ein lineares Mehrschrittverfahren ist nullstabil für ein Anfangswertproblem der Form (2.1) genau dann, wenn alle Nullstellen des ersten charakteristischen Polynoms

des Verfahrens innerhalb der geschlossenen Einheitskreisscheibe in der komplexen Ebene liegen, mit allen auf dem Einheitskreis liegenden Nullstellen einfach.

Beweis: erfolgt mit obigem Lemma, siehe [22].

Ein wichtiger Satz der Konvergenz, Konsistenz und Nullstabilität in Verbindung bringt ist der folgende.

Satz 2.2 (Dahlquist-Äquivalenzsatz): Ein lineares s-Schrittverfahren, das konsistent ist mit der gewöhnlichen Differentialgleichung der Form (2.1) und für das alle Startwerte mit $h \to 0$ gegen den exakten Startwert $u(t_0)$ konvergieren, konvergiert genau dann, wenn es nullstabil ist.

Beweis: siehe z.B. [9].

Aufgrund des Äquivalenzprinzips gilt, dass wenn ein lineares Mehrschrittverfahren nicht nullstabil ist, ihr globaler Fehler nicht beliebig klein gemacht werden kann, indem man die Schrittweite beliebig klein macht für genügend genaue Anfangswerte. Genauer gesagt, wenn die Wurzelbedingung nicht erfüllt ist, dann existiert eine Lösung die in einem festen Zeitintervall beliebig schnell wachsen wird, wie genau die Startwerte auch sein mögen [22]. Aus diesem Grund ist die Nullstabilität von großer Wichtigkeit in praktischen Anwendungen.
Auch wenn eine konsistente nullstabile Methode konvergent ist, bezieht sich das im Allgemeinen auf eine gegen null konvergierende Schrittweite h. Bei realistischen Berechnungen kann es daher trotzdem zu Instabilitäten kommen, wenn die Schrittweite zu groß gewählt wird. Eine zu klein gewählte Schrittweite kann jedoch einen zu hohen Rechenaufwand erfordern. Es wäre also hilfreich zu wissen, ob eine bestimmte Schrittweite ohne zu große Fehler auskommt. Aus diesem Grund sind noch weitere Überlegungen zur Stabilität notwendig, die jedoch auf den bisherigen aufbauen.
Dafür betrachten wir wieder die Testgleichung $u' = \lambda u$, $\lambda \in \mathbb{C}$ und wenden darauf die Formel (2.5) für allgemeine lineare Mehrschrittverfahren an, das ergibt

$$\sum_{j=0}^{s} \alpha_j u_{n+j} = h \sum_{j=0}^{s} \beta_j \lambda u_{n+j},$$

umgeformt erhält man eine Differenzengleichung s-ter Ordnung

$$\sum_{j=0}^{s} (\alpha_j - \lambda h \beta_j) u_{n+j} = 0. \tag{2.8}$$

Das zugehörige charakteristische Polynom ist

$$\pi(z; \lambda h) = \sum_{j=0}^{s} (\alpha_j - h\lambda\beta_j) z^j. \tag{2.9}$$

Dies ist ein Polynom in z aber die Koeffizienten sind abhängig von λh. Es wird auch als das Stabilitätspolynom des linearen Mehrschrittverfahrens bezeichnet. Mit Hilfe des ersten und zweiten charakteristischen Polynoms kann man es auch schreiben als

$$\pi(z; \lambda h) = \rho(z) - \lambda h \sigma(z).$$

Nach Lemma 2.1 kann die Lösung der Differenzengleichung (2.8) in der Form

$$u_n = \sum_{r=1}^{l} p_r(n) z_r^n$$

dargestellt werden. Wenn hier eines der z_r betragsmäßig größer als eins ist, dann können die Näherungslösungen nicht mehr beschränkt sein. Wenn die Vielfachheit m_r von z_r größer als eins ist, dann muss für die Beschränktheit z_r sogar betragsmäßig kleiner als eins sein. Denn dann hat das zugehörige Polynom p_r nach Lemma 2.1 den Grad $m_r - 1$, ist also von der Form $z_r^n n^{m_r - 1} + \cdots + z_r^n n + z_r^n$. Selbst mit $|z_r| = 1$ folgt dann mit wachsendem n für die Näherungslösungen $u_n \to \infty$. Daher kann keine Konvergenz gegen die exakte Lösung gegeben sein, für die mit $t \to \infty$ eine Konvergenz gegen null zu erwarten ist.
Auf diesen Erkenntnissen basierend erhält man nun eine Wurzelbedingung für das Polynom (2.9), die dabei behilflich ist eine geeignete Zeitschrittweite zu wählen.

Definition 2.6: Ein lineares Mehrschrittverfahren heißt absolut stabil für einen gegebenen Wert $\lambda h \in \mathbb{C}$, wenn jede Nullstelle $z_r = z_r(\lambda h)$ des zugehörigen Stabilitätspolynoms $\pi(\ \cdot\ ; \lambda h)$ die Bedingung $|z_r(\lambda h)| \leq 1$ erfüllt, und für mehrfache Nullstellen $|z_r(\lambda h)| < 1$.

Das Stabilitätsgebiet ist nun über alle λh definiert, welche die Bedingungen aus Definition 2.6 erfüllen. Analog zu den Einschrittverfahren definiert man A-Stabilität.
Wir wollen nun noch zu einer allgemeineren Begründung für die Verwendung der Testgleichung (2.4) kommen. Bei gewöhnlichen Differentialgleichungen der Form $u' = f(t, u)$ wird die Analyse numerischer Stabilität oft über Eigenwerte vollzogen. Dafür reduziert man das Problem in drei Schritten auf ein skalares, lineares Problem mit konstanten Koeffizienten [11,18]. Da f im Allgemeinen nichtlinear ist, wird die Gleichung zuerst linearisiert. Für eine Lösung $u^*(t)$ die nahe an $u(t)$ liegt definiert man $w(t) = u(t) - u^*(t)$, mit kleinem $w(t)$. Dann linearisiert man f durch

$$f(t, u) = f(t, u^*) + A(t)w(t) + o(\|w(t)\|),$$

wobei $A(t)$ die Jacobi-Matrix von f ist, mit den partiellen Ableitungen in u:

$$[A(t)]_{jk} = \frac{\partial f}{\partial u_k}(t, u^*(t)), \qquad 1 \leq j, k \leq N.$$

Mit $w(t)$ klein, kann also (2.1) mit einem linearen Problem

$$u'(t) = f(t, u^*) + A(t)w(t) \tag{2.10}$$

approximiert werden.
Setzt man in (2.10) $f(t, u^*) = (u^*)'(t) = u'(t) - w'(t)$ ein, so erhält man

$$w'(t) = A(t)w(t).$$

Im nächsten Schritt werden die Koeffizienten „eingefroren“ indem man $A = A(t^*)$ setzt, für einen festen Zeitpunkt t^*. Die Idee dahinter ist, dass Instabilität ein lokales Phänomen ist, das zu bestimmen Zeitpunkten t^* auftritt. Es ergibt sich also das lineare Problem mit konstanten Koeffizienten

$$w' = Aw. \tag{2.11}$$

Für den letzten Schritt wird A als diagonalisierbar vorausgesetzt. Wir können also $AV = VD$ schreiben, wobei D die Diagonalmatrix mit den Eigenwerten von A auf der Hauptdiagonalen ist und V enthält Spaltenweise die zugehörigen Eigenvektoren.
Nun wenden wir auf (2.11) ein lineares Mehrschrittverfahren an (für Einschrittverfahren verläuft der Rest völlig analog) [15]:

$$\sum_{j=0}^{s} \alpha_j w_{n+j} = h \sum_{j=0}^{s} \beta_j A w_{n+j}. \tag{2.12}$$

w_n lässt sich nun als Linearkombination der Eigenvektorbasis aufschreiben, das heißt $w_n = V w_n$. Setzt man dies in (2.12) ein, so erhält man

$$\begin{aligned} &\sum_{j=0}^{s} \alpha_j V z_{n+j} = h \sum_{j=0}^{s} \beta_j A V z_{n+j} \\ \Leftrightarrow \quad &\sum_{j=0}^{s} \alpha_j V z_{n+j} = h \sum_{j=0}^{s} \beta_j V D z_{n+j}. \end{aligned}$$

Multiplikation mit V^{-1} ergibt schließlich

$$\sum_{j=0}^{s} \alpha_j z_{n+j} = h \sum_{j=0}^{s} \beta_j D z_{n+j}.$$

Schreibt man dies zeilenweise auf, dann folgt

$$\sum_{j=0}^{s} \alpha_j z_{n+j}^i = h \sum_{j=0}^{s} \beta_j \lambda_i z_{n+j}^i, \qquad i = 0, \ldots, N,$$

mit z_n^i die Komponenten von z_n, und λ_i die im Allgemeinen komplexen Eigenwerte von A.

Ist $Re(\lambda_i) \leq 0$, dann ist wegen $\lim_{n\to\infty} \|u_n\| = 0 \Leftrightarrow \lim_{n\to\infty} \|z_n\| = 0$ das Verfahren (2.12) stabil, wenn die lineare Mehrschrittmethode angewendet auf

$$\frac{dz_n^i}{dt} = \lambda_i z_n^i \qquad i = 1, \ldots, N,$$

stabil ist, das heißt wenn $\lambda_i h \in S$.
Generell sagt man dann, dass ein numerisches Verfahren für (2.1) stabil ist um eine Lösung bei t^* zu berechnen, wenn der Zeitschritt h klein genug ist, so dass für jeden Eigenwert λ von $A(t^*)$, λh in dem Stabilitätsgebiet des entsprechenden Verfahrens liegt. Ist A nicht normal, kann diese Eigenwertanalyse jedoch zu falschen Vorhersagen führen.
Dazu betrachten wir wieder das Problem $w' = Aw$, mit A eine konstante $N \times N$-Matrix und approximieren es mit einem expliziten R-stufigen Runge-Kutta-Verfahren mit Zeitschrittweite h. Das diskretisierte Problem ist dann $w_{n+1} = \varphi(hA)w_n$ wobei φ mit $R \leq 4$ ein Polynom vom Grad R ist. Die Näherungslösung für $w(t)$ zum Zeitpunkt nh lässt sich auch schreiben als $w_n = \varphi(hA)^n w_0$. Um das Verhalten von $\varphi(hA)^n$ mit wachsendem n (bzw. Wachsendem t) zu untersuchen, betrachtet man die Norm $\|\varphi(hA)^n\|$ induziert von einer beliebigen Vektornorm, dann erhält man die Abschätzung $\sigma(\varphi(hA))^n \leq \|\varphi(hA)^n\| \leq \|\varphi(hA)\|^n$ [27]. Wenn A nicht normal ist, kann die Lücke zwischen diesen Grenzen ziemlich groß sein [11]. Kurz gesagt, für kleine t nahe bei null ist die Abschätzung nach oben schärfer. Wohingegen die Eigenwerte wegen $\lim_{n\to\infty} \|\varphi(hA)^n\|^{1/n} = \sigma(\varphi(hA))$ eher für $t \to \infty$ eine Auskunft für das Verhalten von $\varphi(hA)^n$ geben [27]. Für kleine t ungleich null, ist eine Abschätzung mit Hilfe der ε-Pseudospektren genauer. Wie in Kapitel vier zu sehen sein wird, ist die Stabilität hier dadurch bestimmt, wie weit die ε-Pseudospektren von hA von S entfernt sind.

Plotten der Stabilitätsgebiete

Wie bereits erwähnt ist für Einschrittverfahren das Stabilitätsgebiet durch die Stabilitätsfunktion ω bestimmt, genauer gesagt werden alle Punkte z $\in \mathbb{C}$ gesucht, für die $|\omega(z)| \leq 1$ ist. Anders formuliert ist es die von den Punkten eingeschlossene Region, die $|\omega(z)| = 1$ erfüllen,. Diese $\omega(z)$ lassen sich demnach schreiben als $\omega(z) = e^{i\Theta}$, $\Theta \in [0, 2\pi)$. Wir wollen zunächst für explizite Runge-Kutta-Verfahren bis zur vierten Stufe, das heißt $R \leq 4$, die Stabilitätsgebiete, bzw. ihre Ränder, grafisch darstellen. Die Vorgehensweise richtet sich nach [5]. Für jeden Winkel $\Theta = 2\pi/n$ löst man $\omega(z) - e^{i\Theta} = 0$ zwischen 0 und $2\pi R$, unter Verwendung des Newton-Verfahrens. Dabei wird jeweils als Startwert das Ergebnis genommen, das für den vorherigen Winkel bestimmt wurde und null als Startwert für die allererste Approximation bei $\Theta = 0$.
Ein Problem mit dieser Vorgehensweise ist, dass für $|\omega(z)| = 1$ mehrere Lösungen möglich sind. Als Alternative zum Newton-Verfahren kann man auch für die Funktion $|\omega(z)|$ mit einem Konturplotter die Konturlinie mit dem Wert 1 zeichnen lassen. Diese Alternative erfordert jedoch normalerweise mehr Rechenzeit.
Es fehlt nun noch die Bestimmung der Stabilitätsfunktion für das Runge-Kutta-Verfahren. Die zugehörige Verfahrensfunktion ist [18]:

$$F(h,t,u_{n+1},u_n) = \sum_{r=1}^{R} b_r k_r(h,t_n,u_n), \qquad k_1 = f(t_n,u_n),$$

$$k_r = f(t_n + hd_r, u_n + h\sum_{s=1}^{r-1} a_{rs}k_s).$$

Angewendet auf das Testproblem (2.4) ergibt sich dann z.B. für $R = 2$ (Heun-Verfahren):

$$k_1 = \lambda u_n,$$
$$k_2 = \lambda(u_n + h\lambda u_n) = \lambda u_n + h\lambda^2 u_n.$$
$$\Rightarrow u_{n+1} = u_n + h\frac{1}{2}\lambda u_n + h\frac{1}{2}(\lambda u_n + h\lambda^2 u_n) = (1 + h\lambda + \frac{1}{2}h^2\lambda^2)u_n.$$

Analog erhält man für R-stufige Runge-Kutta-Verfahren mit $R \leq 4$

$$\omega(\lambda h) = \sum_{r=0}^{R} \frac{(\lambda h)^r}{r!}$$

als Stabilitätsfunktion.

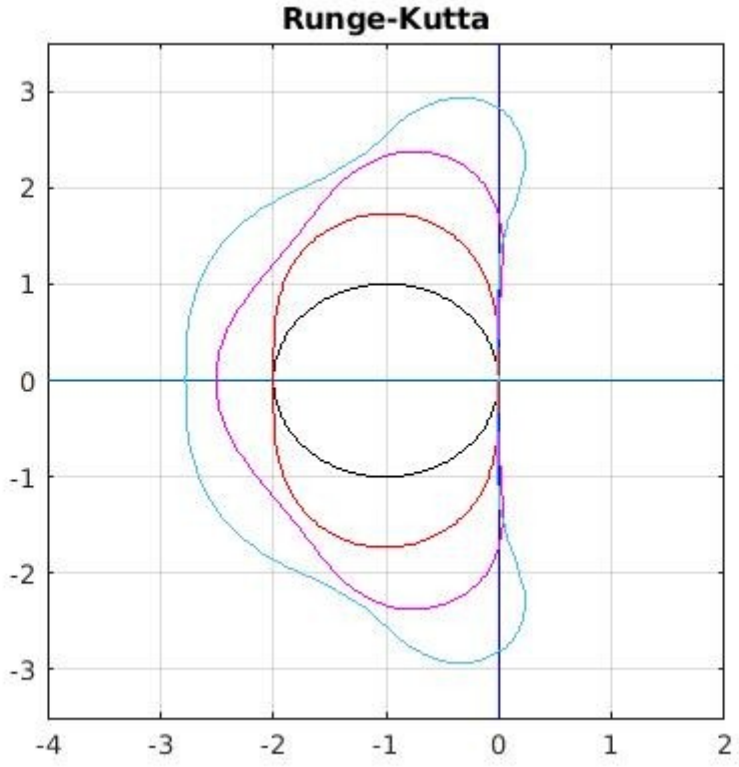

Abb.2.1: Die Ränder der Stabilitätsgebiete der Runge-Kutta-Verfahren 1. bis 4. Stufe. Mit steigender Ordnung wird das Gebiet größer.

Die Implementierung unter Verwendung des Newton-Verfahrens, ist z.B. in [25] einsehbar. Ergänzend wird hier die Version mit Hilfe des Konturplotters aufgeführt, da diese vor allem in MATLAB einfach umzusetzen ist.

```
plot ([-4 2],[0 0]) , hold on, plot ([0 0],[-4 4], 'b') % Koordinatensystem zeichnen
% 1. Ordnung
x = -3:0.1:1; y = -3:0.1:3; [X,Y] = meshgrid(x,y); w = X+1i*Y;
W = abs(1+w);
contour(x,y,W,[1,1],'black');
% 2. Ordnung
W = abs(1+w+.5*w.^2);
contour(x,y,W,[1,1],'r');
% 3. Ordnung
W = abs(1+w+.5*w.^2+w.^3/6);
contour(x,y,W,[1,1],'m');
% 4. Ordnung
W = abs(1+w+.5*w.^2+w.^3/6+w.^4/24);
contour(x,y,W,[1,1]);
toc
axis([-4 2 -3.5 3.5]),axis square, grid on, title Runge-Kutta
```

Ein Nachteil dieser Methode ist wie gesagt die längere Berechnungszeit. Mit einem gröberen Gitter kann sie verkürzt werden, ein zu grobes Gitter liefert jedoch ungenaue Konturlinien, die dann im Plot eckig erscheinen.
Im Falle von linearen Mehrschrittverfahren wurde schon gezeigt, dass ein Punkt ζ im Stabilitätsgebiet S liegt, wenn das Stabilitätspolynom $\pi(z;\zeta)$ die Wurzelbedingung erfüllt. Wir wollen wieder die Randpunkte von S suchen um S grafisch darzustellen [15,16]. Da die Nullstellen eines Polynoms stetig von den Koeffizienten abhängen, muss für einen Punkt ζ für den mindestens eine Nullstelle von $\pi(z;\zeta)$ den Betrag eins hat, dieser Punkt auf dem Rand von S liegen. Diese Nullstelle ist von der Form $z = e^{i\Theta}$ und es gilt $\pi(e^{i\Theta};\zeta) = \rho(e^{i\Theta}) - \zeta\sigma(e^{i\Theta}) = 0$, daraus folgt $\zeta = \rho(e^{i\Theta})/\sigma(e^{i\Theta})$.
Ist der Winkel Θ bekannt, kann ζ aus dieser Formel bestimmt werden. Man plottet also einfach die Kurve

$$\zeta(\Theta) = \frac{\rho(e^{i\Theta})}{\sigma(e^{i\Theta})}.$$

Um herauszufinden auf welcher Seite der Kurve das Innere von S ist, kann z.B. für einen Punkt auf einer der Seiten neben der Kurve geprüft werden, ob $\pi(z;\zeta)$ hier die Wurzelbedingung erfüllt.
Als Beispiel wird das Adams-Bashforth-Verfahren dritter Ordnung betrachtet

$$u_{n+3} = u_{n+2} + \frac{h}{12}(5f_n - 16f_{n+1} + 23f_{n+2}).$$

Hier ist

$$\frac{\rho(z)}{\sigma(z)} = \frac{z^3 - z^2}{\frac{23}{12}z^2 - \frac{16}{12}z + \frac{5}{12}}.$$

Für die Adams-Bashforth-Verfahren erster bis dritter Ordnung sind die Ränder der gesuchten Gebiete in *Abb.2.2* zu sehen.

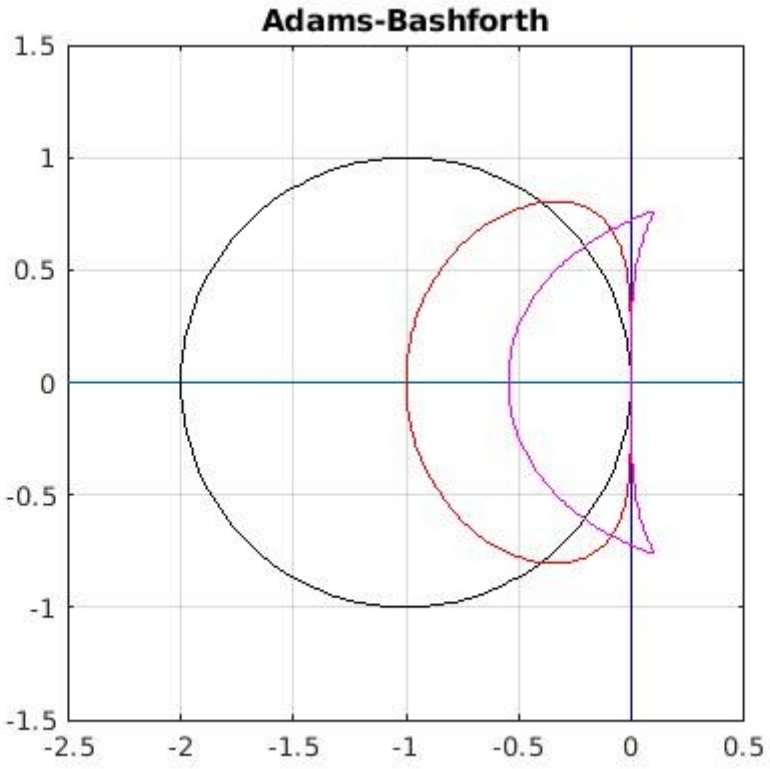

Abb.2.2: Das große Kreisförmige Gebiet ist das zur 1. Ordnung gehörende, mit steigender Ordnung werden die Stabilitätsgebiete kleiner.

Der zugehörige Programmcode ist:

```
plot ([-8 8],[0 0]), hold on, plot ([0 0],[-8 8],'b' )
z = exp(1i*pi*(0:200)/100);
% 1.Ordnung
zeta = z-1; plot(zeta,'black')
% 2.Ordnung
zeta = 2*(z.^2-z)./(3*z-1); plot(zeta,'r')
% 3.Ordnung
zeta = 12*(z.^3-z.^2)./(23*z.^2-16*z+5); plot(zeta,'m')
axis([-2.5 .5 -1.5 1.5]), axis square, grid on
title Adams-Bashforth
```

3 Pseudospektren

Wie schon im vorigen Kapitel angesprochen, spielt die Eigenwertanalyse eine wichtige Rolle für die Behandlung von zeitabhängigen Differentialgleichungen, wird jedoch in bestimmen Fällen eine nicht ausreichend genaue Vorhersage gewährleisten können. Bei gewissen Problemen bei denen sich gezeigt hat, dass Vorhersagen basierend auf der Auswertung von Eigenwerten nicht mit den Beobachtungen übereinstimmen, findet sich eine mögliche Anwendung von Pseudospektren. Dabei handelt es sich für gewöhnlich um Probleme bei denen Matrizen oder Operatoren involviert sind, die nicht normal sind. Wir beschränken uns im Folgenden nur auf Matrizen A in $\mathbb{C}^{N\times N}$ mit einer Norm induziert von einer Vektornorm auf $\mathbb{C}^N$. Ausweitungen auf lineare Operatoren in Banachräumen sind ebenfalls möglich (siehe z.B. [27]). Für nicht normale Matrizen ist ausschlaggebend, dass ihre Eigenvektoren nicht orthogonal sind. Darüber hinaus ist die Konditionszahl der Matrix der Eigenvektoren größer als 1 und möglicherweise sogar viel größer [23].
Ein Eigenwert einer Matrix A ist bekanntlich eine Zahl $z \in \mathbb{C}$ mit der $zI–A$ singulär ist. Eine Ausweitung dieser Definition kann so aussehen, dass man nicht nur untersucht ob $(zI–A)^{-1}$ existiert, sondern ob $\left\|(zI-A)^{-1}\right\|$ groß oder klein ist. Die Matrix $(zI–A)^{-1}$ wird dabei als die Resolvente von A im Punkt $z \in \mathbb{C}$ bezeichnet. Die Idee dahinter ist, dass eine kleine Störung von A einen Übergang von Singularität zu Nichtsingularität oder umgekehrt bedeuten kann. Einfach nur nach der Invertierbarkeit von $zI–A$ zu fragen, kann daher in gewissen Problemstellungen zu grob sein.
Sei nun $\sigma(A) \subset \mathbb{C}$ das Spektrum der Matrix A (für Matrizen ist $\sigma(A)$ endlich, diskret und nichtleer). Um die „Größe“ der Resolvente zu bestimmen, bietet sich nun folgende Definition an.

Definition 3.1: Sei $A \in \mathbb{C}^{N\times N}$. Für beliebiges $\varepsilon > 0$ bezeichnet man die Teilmenge von $\mathbb{C}$ definiert durch

$$\sigma_\varepsilon(A) = \left\{z \in \mathbb{C} : \left\|(zI-A)^{-1}\right\| > \varepsilon^{-1}\right\} \tag{3.1}$$

als das ε-Pseudospektrum von A.

In Anlehnung an die Definition von Eigenwerten schreiben wir $\left\|(zI-A)^{-1}\right\| = \infty$ falls $z \in \sigma(A)$. So ist das Spektrum $\sigma(A)$ Teilmenge des ε-Pseudospektrums für alle $\varepsilon > 0$. In Worten formuliert, kann man das ε-Pseudospektrum beschreiben als die offene Teilmenge der komplexen Ebene begrenzt durch die ε^{-1} Konturlinie der Resolventennorm. Diese Konturlinien macht man sich auch zu Nutze um die ε-Pseudospektren grafisch darzustellen. Des Weiteren sind die ε-Pseudospektren von A eine Folge von verschachtelten Mengen,

$$\sigma_{\varepsilon_1}(A) \subseteq \sigma_{\varepsilon_2}(A) \subseteq \ldots, \qquad 0 < \varepsilon_1 < \varepsilon_2 < \ldots$$

die für $\varepsilon \to \infty$ soweit anwachsen bis sie die ganze komplexe Ebene ausfüllen. Der Schnitt von allen Pseudospektren ist das Spektrum:

$$\bigcap_{\varepsilon>0} \sigma_\varepsilon(A) = \sigma(A).$$

Intuitiv könnte man annehmen, dass $\left\|(zI - A)^{-1}\right\|$ gerade dann groß ist, wenn z nahe an einem Eigenwert von A liegt. Dies ist tatsächlich für normale Matrizen mit der Spektralnorm $\|\cdot\|_2$ zutreffend. Für nicht normale Matrizen kann die Norm der Resolvente aber sogar groß sein wenn z weit vom Spektrum entfernt ist.
Eine weitere Definition von Pseudospektren wird im Sinne von Eigenwerten einer gestörten Matrix formuliert. Für eine beliebige Matrix $A \in \mathbb{C}^{N\times N}$ sei

$$\sigma_\varepsilon(A) = \big\{z \in \mathbb{C} : z \in \sigma(A + E) \text{ für } E \in \mathbb{C}^{N\times N} \text{ mit } \|E\| < \varepsilon\big\}. \tag{3.2}$$

In einer dritten Definition des ε-Pseudospektrums wird die Existenz eines zugehörigen Eigenvektors mit einbezogen:

$$\sigma_\varepsilon(A) = \big\{z \in \mathbb{C} : \|(zI - A)\mathbf{v}\| < \varepsilon \text{ für ein } \mathbf{v} \in \mathbb{C}^N \text{ mit } \|\mathbf{v}\| = 1\big\}. \tag{3.3}$$

Die Zahl z ist ein ε-Pseudoeigenwert von A und $\mathbf{v}$ ist der zugehörige ε-Pseudoeigenvektor. Gleichbedeutende Bezeichnungen für $\mathbf{v}$ sind Pseudoeigenfunktion, Pseudoeigenmode, und Pseudomode.

Satz 3.1: Für eine beliebige Matrix $A \in \mathbb{C}^{N\times N}$ sind die Definitionen (3.1) – (3.3) äquivalent.

Beweis: siehe [27].

$\mathbb{C}^N$ wird ausgerüstet mit dem Standardskalarprodukt $(\mathbf{u}, \mathbf{v}) = \mathbf{v}^*\mathbf{u}$ und der dadurch induzierten 2-Norm $\|\mathbf{v}\|_2 = \sqrt{\mathbf{v}^*\mathbf{v}}$, die zugehörige Matrixnorm ist die schon erwähnte Spektralnorm.
Eine vierte äquivalente Definition, die auch für die Berechnung der Pseudospektren angemessener ist, beinhaltet den kleinsten Singulärwert s_{min} der Resolvente. Für eine beliebige Matrix $A \in \mathbb{C}^{N\times N}$ sei

$$\sigma_\varepsilon(A) = \{z \in \mathbb{C} : s_{min}(zI - A) < \varepsilon\}. \tag{3.4}$$

In einem allgemeinen Banachraum sind nur die Definitionen (3.1) – (3.3) äquivalent aber nicht (3.4). Hier in $\mathbb{C}^N$ (Hilbertraum) und mit der Spektralnorm $\|\cdot\|_2$, wie es in Anwendungen meist angemessen ist, ist die Äquivalenz von (3.1) und (3.4), und damit auch zu den anderen Definitionen, leicht einzusehen: Die Spektralnorm einer Matrix ist gleich ihrem größten Singulärwert, und die Norm ihrer Inversen ist gleich die Inverse des kleinsten Singulärwertes. Damit gilt

$$\varepsilon^{-1} < \left\|(zI - A)^{-1}\right\|_2 = [s_{min}(zI - A)]^{-1} \qquad \Longleftrightarrow \qquad s_{min}(zI - A) < \varepsilon.$$

Um jetzt die Problemstellung der Nichtnormalität zu behandeln, sei noch mal die Definition einer normalen Matrix erwähnt.

Definition 3.2: Eine Matrix $A \in \mathbb{C}^{N\times N}$ ist normal, wenn sie eine vollständige Menge von orthogonalen Eigenvektoren besitzt, das bedeutet wenn sie unitär diagonalisierbar ist:

$$A = UDU^*, \tag{3.5}$$

wobei U eine unitäre Matrix und D eine Diagonalmatrix mit den Eigenwerten von A ist.

Wenn A normal ist, dann ist $\sigma_\varepsilon(A)$ genau die Menge von Punkten in $\mathbb{C}$ in einem Abstand $< \varepsilon$ von $\sigma(A)$. Wenn A nicht normal ist, kann diese Menge deutlich größer sein. Um dies in einem Satz zu beweisen sind noch ein paar Definitionen nützlich. Eine offene Kugel mit Radius ε sei folgendermaßen bezeichnet:

$$\Delta_\varepsilon = \{z \in \mathbb{C} : |z| < \varepsilon\}, \tag{3.6}$$

und die Menge von Punkten um das Spektrum mit einem Abstand kleiner ε durch

$$\begin{aligned} \sigma(A) + \Delta_\varepsilon &= \{z \in \mathbb{C} : z = z_1 + z_2,\ z_1 \in \sigma(A),\ z_2 \in \Delta_\varepsilon\} \\ &= \{z \in \mathbb{C} : \operatorname{dist}(z, \sigma(A)) < \varepsilon\}\,. \end{aligned} \tag{3.7}$$

Wobei dist die Entfernung von einem Punkt zu einer Menge in der komplexen Ebene bezeichnet.

Satz 3.2: Für eine Matrix $A \in \mathbb{C}^{N\times N}$ gilt

$$\sigma(A) + \Delta_\varepsilon \subseteq \sigma_\varepsilon(A) \qquad \forall \varepsilon > 0, \tag{3.8}$$

und falls A normal ist und die Spektralnorm benutzt wird, dann ist

$$\sigma(A) + \Delta_\varepsilon = \sigma_\varepsilon(A) \qquad \forall \varepsilon > 0. \tag{3.9}$$

Beweis: vgl. auch [27]. Wenn z ein Eigenwert von A ist und $\delta \in \mathbb{C}$ mit $|\delta| < \varepsilon$, dann gilt $z + \delta \in \sigma(A) + \Delta_\varepsilon$. Setze in (3.2) $E = \delta I$, dann ist $\|E\| = |\delta| < \varepsilon$ und damit $z + \delta \in \sigma(A + \delta I) = \sigma(A + E) = \sigma_\varepsilon(A)$.
Um (3.9) zu zeigen, sei A als normal vorausgesetzt, dann kann sie unter Benutzung der Spektralnorm als diagonal vorausgesetzt werden, mit den Eigenwerten λ_i als Diagonaleinträge, ohne dass sich ihre Norm verändert. Die Resolvente ist dann ebenfalls diagonal und es gilt

$$\left\|(zI - A)^{-1}\right\|_2 = \frac{1}{\min_i |z - \lambda_i|} = \frac{1}{\operatorname{dist}(z, \sigma(A))},$$

und mit (3.1) ist dann $\operatorname{dist}(z, \sigma(A)) < \varepsilon$ für ein ε-Pseudoeigenwert z, das heißt $z \in \sigma(A) + \Delta_\varepsilon$. Es folgt also $\sigma_\varepsilon(A) \subseteq \sigma(A) + \Delta_\varepsilon$ und zusammen mit (3.8) schließlich $\sigma(A) + \Delta_\varepsilon = \sigma_\varepsilon(A)$.

□

Sei jetzt A diagonalisierbar angenommen, aber nicht notwendig normal, das bedeutet $A = VDV^{-1}$ wobei $V \in \mathbb{C}$ die Matrix der Eigenvektoren von A ist. Die Konditionszahl der Basis der Eigenvektoren ist dann

$$\kappa(V) = \|V\|_2 = ||V^{-1}||_2 = \frac{s_{max}(V)}{s_{min}(V)}. \tag{3.10}$$

Man beachte, dass $\kappa(V)$ für eine gegebene Matrix A nicht eindeutig ist, da auch V nicht eindeutig bestimmt ist. Im Allgemeinen ist $\kappa(V)$ eine Zahl im Bereich $1 \leq \kappa(V) < \infty$ und der Wert $\kappa(V) = 1$ ist nur möglich, genau dann wenn A normal ist.
Mit der Konditionszahl kann nun auch eine obere Schranke für ein ε-Pseudospektrum angegeben werden.

Satz 3.3(Bauer-Fike): Sei $A \in \mathbb{C}^{N\times N}$ diagonalisierbar, das heißt $A = VDV^{-1}$. Dann gilt für alle $\varepsilon > 0$ unter Verwendung der Spektralnorm, dass

$$\sigma(A) + \Delta_\varepsilon \subseteq \sigma_\varepsilon(A) \subseteq \sigma(A) + \Delta_{\varepsilon\kappa(V)}. \tag{3.11}$$

Beweis: vgl. [27]. Die erste Inklusion gilt nach (3.8). Für die zweite berechnet man

$$(zI - A)^{-1} = (zI - VDV^{-1})^{-1} = [V(zI - D)V^{-1}]^{-1} = V(zI - D)^{-1}V^{-1},$$

woraus

$$\left\|(zI - A)^{-1}\right\|_2 \leq \kappa(V) \left\|(zI - D)^{-1}\right\|_2 = \frac{\kappa(V)}{\operatorname{dist}(z, \sigma(A))}$$

folgt. Mit (3.1) gilt dann für ein $z \in \sigma_\varepsilon(A)$ die Ungleichung $\operatorname{dist}(z, \sigma(A)) < \varepsilon\kappa(V)$ und somit $z \in \sigma(A) + \Delta_{\varepsilon\kappa(V)}$.

□

Zur Berechnung von Pseudospektren macht man sich Definition (3.4) zum Vorteil. Dazu wertet man $\sigma_{min}(zI - A)$ mit einer Singulärwertzerlegung (wie hier die in MATLAB vordefinierte) in den Punkten z eines Gitters in der komplexen Ebene aus. Mit den daraus resultierenden Zahlen erstellt man einen Konturplot der für ausgewählte Isolinien der Höhe ε die zugehörigen ε-Pseudospektren darstellt. Dies ist die grundlegende, einfach umzusetzende Vorgehensweise. Wenn A hermitesch ist, ist die Menge symmetrisch bezüglich der reellen Achse und man kann sich diese Symmetrie zu nutze machen um die Rechenzeit zu halbieren. Im Allgemeinen ist der Aufwand dieser Vorgehensweise jedoch hoch, auf einem $m \times m$- Gitter müssen m^2 Singulärwertzerlegungen durchgeführt werden, der Rechenaufwand für eine $N \times N$-Matrix ist somit von der Ordnung $O(m^2N^3)$. Verschiedene Methoden wurden bereits vorgeschlagen um die Berechnungsgeschwindigkeit gegenüber der oben genannten Methode zu beschleunigen. Einer dieser Algorithmen der eine deutliche Beschleunigung ermöglicht, ist die Methode der Fortsetzung (method of continuation). Dabei wird A mittels einer Schur-Zerlegung durch die Matrix T ersetzt, $(zI - T)$ $(zI - A)$

$(zI - T)$ hat nun die Eigenschaft, die gleichen Singulärwerte wie $(zI - A)$ zu besitzen, die jetzt jedoch mit weniger Aufwand berechnet werden können, da $(zI - T)$ eine obere Dreiecksmatrix ist. Der Gesamtaufwand wird so auf die Größenordnung $O(N^3 + m^2N^2)$ reduziert [24].
Im nächsten Kapitel wollen wir nun die Vorteile von Pseudospektren ausnutzen, um für bestimmte Fragen zur Stabilität Auskunft zu erteilen, wo sonst Eigenwerte dazu verwendet werden, jedoch auf Grund von Nichtnormalität die Eigenwertanalyse nicht ausreichend ist. Der Nachteil dieser Alternative ist jedoch, dass Pseudospektren nicht so einfach zu bestimmen und anzugeben sind wie Eigenwerte und man eher nur in Form von Abschätzungen Antworten erhält.

4 Stabilität der Linienmethode

Eine oft benutzte Methode um zeitabhängige partielle Differentialgleichungen zu lösen, besteht darin eine (spektrale) Diskretisierung der räumlichen Ableitungen vorzunehmen, in Verbindung mit einem Differenzenverfahren für die Zeitschritte. Letzteres kann mit einem der in Kapitel zwei aufgeführten Verfahren realisiert werden, das heißt Runge-Kutta, Adams-Bashforth oder Ähnliche. Nach der räumlichen Diskretisierung bleibt ein System von gewöhnlichen Differentialgleichungen übrig, darstellbar über eine Diskretisierungsmatrix für die dann eine Eigenwertanalyse gemacht werden kann, oder eben in speziellen Fällen eine Analyse der Pseudospektren.
Die Entkopplung von Raum und Zeit ist als die Linienmethode bekannt und hat vor allem den Vorteil, dass sie auf einem einfachen Prinzip beruht. Obwohl hier für ausreichende Genauigkeit eher kleine Zeitschritte h vonnöten sind, erhält man dennoch für gewöhnlich akzeptable Ergebnisse, da der Rechenaufwand mit kleiner werdendem h nur linear steigt und der Speicherbedarf gar nicht wächst [19]. Die numerische Stabilität von solch einer Linienmethode basierend auf spektralen räumlichen Ableitungen ist jedoch, wie sich zeigen wird, nicht immer unproblematisch.
Um die Linienmethode formal auszudrücken, betrachten wir eine Gleichung der Form

$$\begin{aligned} u_t &= \mathcal{L}u \qquad t \in [0,T] \\ u(x,0) &= f(x) \end{aligned} \tag{4,1}$$

wobei $\mathcal{L}$ ein zeitabhängiger linearer Differentialoperator ist und u eine skalare Funktion oder eine Vektorfunktion abhängig von der Zeit t und einer oder mehrerer Raumvariablen x. u_t bezeichnet die Ableitung nach der Zeit. Zuerst wird diese Gleichung mit finiten Differenzen, finiten Elementen oder wie hier eher von Interesse, mit einer Spektralmethode in Bezug auf die Raumvariablen approximiert. Damit wird die partielle Differentialgleichung (4.1) zu einem System von gewöhnlichen Differentialgleichungen

$$v_t = L_h v, \qquad v(0) = f_h, \tag{4.2}$$

wobei $v(t)$ ein Vektor der Dimension N_h, und L_h die Diskretisierungs- bzw. Ableitungsmatrix ist. Die Semidiskretisierung (4.2) wird dann mit einem linearen Mehrschrittverfahren, oder einem Einschrittverfahren bezüglich der Zeit t approximiert. Schreibt man $v_n \approx v(nh)$ dann lautet die vollständig diskretisierte Gleichung

$$\mathbf{v}_{n+1} = A_h \mathbf{v}_n = G(hL_h)\mathbf{v}_n, \tag{4.3}$$

mit geeigneten Anfangswerten. Für eine Einschrittmethode ist $\mathbf{v}_n = v_n$, wohingegen für ein lineares s-Schritt-Verfahren

$$\mathbf{v}_{n+1} = \begin{bmatrix} v_{n+s} \\ \vdots \\ v_{n+1} \end{bmatrix}, \qquad \mathbf{v}_n = \begin{bmatrix} v_{n+s-1} \\ \vdots \\ v_n \end{bmatrix}, \tag{4.4}$$

mit Vektoren jeweils der Länge sN_h. Die Funktion $G(w)$ charakterisiert die entsprechende Zeitintegrationsformel. Für eine lineare Mehrschrittmethode bildet sie eine Begleitmatrix mit affinen und rationalen Funktionen in w, sowohl für explizite als auch implizite Verfahren. Für Runge-Kutta- oder Einschrittverfahren ist G(w) ein Polynom oder rationale Funktion die e^w für $w \approx 0$ approximiert [20].
Wendet man (4.3) auf das altbekannte Modellproblem $u_t = \lambda u$ an, so erfüllt die Folge $\{\mathbf{v}_n\}$

$$\mathbf{v}_n = [G(\lambda h)]^n \mathbf{v}_0, \qquad n \geq 0. \tag{4.5}$$

Die Folge ist beschränkt wenn $[G(\lambda h)]^n$ für alle $n \geq 0$ beschränkt ist, dies gilt genau dann wenn alle Eigenwerte dieser Matrix kleiner gleich eins sind, mit nur einfachen Eigenwerten vom Betrag genau gleich eins [27]. Da es sich hier um eine Begleitmatrix handelt deren Eigenwerte genau die Nullstellen der charakteristischen Polynoms (2.9) sind, gilt hier also die Beschränktheit von (4.5) genau dann wenn λh im Stabilitätsgebiet S liegt.
Stabilitätsanalysen stellen sich hier für spektrale Methoden als schwieriger heraus als für finite Differenzen, da die beteiligten Operatoren nicht normal sind. Zudem sind die Grenzen für Zeitschrittweiten deutlich restriktiver, typischerweise $h \leq C/N^2$ für explizite Methoden von Problemen erster Ordnung und $h \leq C/N^4$ für solche zweiter Ordnung, auf einem Gitter mit N Punkten pro Raumdimension. Versucht man zu umgehen, bei expliziten Methoden geringe Schrittweiten zu wählen um Stabilität zu gewährleisten, indem man eine A-stabile implizite Methode wählt, ist man gezwungen unter großem Aufwand stark besetzte und schlecht konditionierte Matrizen zu invertieren.
Zum Problem der Stabilitätsanalyse sind in erster Linie die Eigenwerte der Diskretisierungsmatrix von Bedeutung. Ähnlich wie schon in Kapitel zwei formuliert gilt als Faustregel, dass die Linienmethode stabil ist, wenn die Eigenwerte der räumlichen Diskretisierungsmatrix, multipliziert mit h, in dem Stabilitätsgebiet der Zeitdiskretisierung liegen.
Anders ausgedrückt, sei L_h der räumliche Diskretisierungsoperator, dann müssen die Eigenwerte von hL_h in dem Stabilitätsgebiet der Zeitschrittformel liegen. Jedoch gilt auch hier wieder, dass L_h normal sein muss, andernfalls kann diese Vorhersagemethode versagen. Im Falle von Spektralmethoden kann L_h für kleine h weit von einer normalen Matrix abweichen, in dem Sinne, dass die Eigenvektoren dieser Matrix längst nicht orthogonal sind, oder wie es in Kapitel drei ausgedrückt wurde, die Konditionszahl der Matrix der Eigenvektoren ist sehr groß. Als Alternative, insbesondere bei Verdacht einer nicht normalen Matrix, ersetzt man in dieser Faustregel die Eigenwerte mit ε-Pseudospektren. Genauer gesagt muss das ε-Pseudospektrum in einem Abstand $O(\varepsilon) + O(h)$ von dem Stabilitätsgebiet liegen, für $\varepsilon \to 0$ und $h \to 0$.
Zu beachten ist auch, dass für eine feiner werdende räumliche Diskretisierung sich die Anzahl der Eigenwerte von hL_h erhöht und wenn sie von der Anzahl der Gitterpunkte abhängen, immer größer werden. Für die Konvergenz von (4.3) muss daher mit kleiner werdender räumlicher Gitterweite Δx auch die Zeitschrittweite mit einer passenden Rate kleiner werden.
Was wir nun noch brauchen ist ein für partielle Differentialgleichungen ähnlicher Begriff zur Nullstabilität.

Definition 4.1(Lax-Stabilität): Die volle Diskretisierung (4.3) heißt Lax-stabil, wenn

$$\|A_h^n\| \leq C, \qquad \forall\ n, h \text{ mit } 0 \leq nh \leq T,$$

für eine Konstante C und genügend kleine Zeitschritte h.

Ähnlich dem Dahlquist-Äquivalenzsatz für gewöhnliche Differentialgleichungen gibt es den Äquivalenzsatz von Lax der besagt, dass eine konsistente Methode der Form (4.3) für h gegen null konvergent ist, genau dann wenn sie Lax-stabil ist [20].
Die Bedingung $\sup_{n\geq 0} \|A^n\|$ ist äquivalent zu der Bedingung, dass alle Eigenwerte von A in der Einheitskreisscheibe liegen und alle Eigenwerte auf dem Einheitskreis einfach sind [27].
Diese Bedingung an die Eigenwerte gibt jedoch keine Auskunft darüber, wie groß $\sup_{n\geq 0} \|A^n\|$ sein könnte, für eine bestimmte Wahl von A, und auch nicht ob für eine Familie von Matrizen A_ν eine gleichmäßige Beschränktheit der Form $\sup_{n\geq 0} \|A_\nu^n\|$ unabhängig von ν gilt. Diese gleichmäßige Beschränktheit wird gerade für die Stabilität der Linienmethode benötigt. Der folgende Satz gibt dafür eine hinreichende und notwendige Bedingung, über die Entfernung der ε-Pseudospektren von der Einheitskreisscheibe D, an.

Satz 4.1: Wenn für die Familie von Matrizen $\{A_\nu\}$ gilt, dass

$$\|A_\nu^n\| \leq C \qquad \forall n \geq 0 \tag{4.6}$$

für eine Konstante C, unabhängig von ν, dann erfüllen die ε-Pseudospektren $\{\lambda_\varepsilon\}$ dieser Matrizen

$$\operatorname{dist}(\lambda_\varepsilon, D) \leq C\varepsilon \qquad \forall \varepsilon \geq 0. \tag{4.7}$$

Umgekehrt folgt aus (4.7)

$$\|A_\nu^n\| \leq 2eC \min\{N_\nu, n\} \qquad \forall n > 0. \tag{4.8}$$

Beweis: siehe [19,17].

Eine bessere Abschätzung ohne den Faktor 2 wird in [20] angegeben.
Betrachten wir nun eine s-Schritt lineare Mehrschrittmethode als Approximation zur Semidiskretisierung (4.2) gegeben durch

$$\sum_{j=0}^{s} \alpha_j v_{n+j} = h \sum_{j=0}^{s} \beta_j h L_h v_{n+j}, \tag{4.9}$$

mit der Konvention $\alpha_s = 1$. Umgeformt ergibt sich daraus

$$v_{n+s} = (I - \beta_s h L_h)^{-1} \left(\sum_{j=0}^{s-1} (\beta_j h L_h - \alpha_j) v_{n+j} \right). \tag{4.10}$$

Mit den Vektoren aus (4.4) erhält man dann für (4.3) die $sN_h \times sN_h$ - Matrix

$$A_h = G(hL_h) = \begin{bmatrix} a_{s-1} & \dots & a_1 & a_0 \\ I & & & \\ & \ddots & & \\ & & I & \end{bmatrix}, \tag{4.11}$$

mit $a_j = (I - \beta_s h L_h)^{-1}(\beta_j h L_h - \alpha_j I),\ 0 \le j \le s-1$ und I ist die Identitätsmatrix der Dimension N_h.
Wie auch schon für die grafische Darstellung der Stabilitätsgebiete verwendet, bezeichne

$$r(z) = \frac{\rho(z)}{\sigma(z)}$$

die rationale Funktion, welche die Einheitskreisscheibe D auf das Stabilitätsgebiet abbildet.
Die nächsten Resultate beziehen sich auf Mehrschrittverfahren, die folgende Voraussetzungen erfüllen:

(V.1) S ist beschränkt mit $r(z) \neq \infty$ für $z \in \partial D$.
(V.2) $r'(z) \neq 0$ für $z \in \partial D$.

Für explizite Methoden folgt schon aus $r(z) \neq \infty$ für $z \in D$, dass S beschränkt ist, für implizite Methoden reicht dies jedoch nicht aus. (V.2) impliziert, dass das Stabilitätsgebiet keine Spitzen aufweist [20].
Der folgende Satz ist das Hauptresultat für die Stabilität der Linienmethode auf unendlichen Zeitintervallen. Der Beweis erfolgt durch eine Übertragung von Satz 4.1, von der Einheitskreisscheibe auf ein beliebiges Stabilitätsgebiet. Zunächst für unendliche Zeitintervalle $[0, \infty)$.

Satz 4.2: Sei (4.9) die Diskretisierung von (4.1) mit einer linearen Mehrschrittmethode die (V.1) und (V.2) erfüllt. Wenn

$$\|A_h^n\| \le C_1, \qquad \forall n \ge 0, \tag{4.12}$$

dann erfüllen die ε-Pseudoeigenwerte $\{\mu_\varepsilon\}$ von $\{hL_h\}$

$$\operatorname{dist}(\mu_\varepsilon, S) \le C_2 \varepsilon \qquad \forall \varepsilon \ge 0. \tag{4.13}$$

Umgekehrt folgt aus (4.10)

$$\|A_h^n\| \le C_3 \min\{N_h, n\}, \qquad \forall n > 0. \tag{4.14}$$

Die Konstanten C_i sind unabhängig von h und von Δx.

Beweis: (vgl. [19]). Ersetzt man (4.13) mit

$$\operatorname{dist}(\lambda_\varepsilon, D) \leq C_2' \varepsilon, \tag{4.15}$$

dann folgt die Behauptung aus Satz 4.1. Also reicht es zu zeigen, dass (4.15) und (4.13) äquivalent sind. Aus der Definition für ε-Pseudoeigenwerte folgend, kann man (4.13) schreiben als

$$\left\|(\mu I - hL_h)^{-1}\right\| \leq \frac{C_2}{\operatorname{dist}(\mu, S)} \qquad \mu \in \mathbb{C} \setminus \bar{S}, \tag{4.16}$$

und (4.15) als

$$\left\|(\lambda I - A_h)^{-1}\right\| \leq \frac{C_2'}{\operatorname{dist}(\lambda, D)} \qquad \lambda \in \mathbb{C} \setminus \bar{D}, \tag{4.17}$$

wir zeigen also die Äquivalenz von (4.16) und (4.17).

(i) Wir zeigen zuerst, aus (4.16) folgt (4.17) und nehmen dafür an, dass λ nahe an D liegt und definieren den Ring $\Omega_z = \{z \in \mathbb{C} : 0 < \operatorname{dist}(z, D) < \tau_z\}$ für ein $\tau_z > 0$ klein genug, so dass $r(z) \neq \infty$ für alle $z \in \overline{\Omega}_z$, was nach (V.1) möglich ist. Sei $\lambda \in \Omega_z$ beliebig und definiere $\mu = r(\lambda)$. Da $A_h = G(hL_h)$, kann $(\lambda I - A_h)^{-1}$ als Resolventenintegral bezüglich w geschrieben werden,

$$(\lambda I - A_h)^{-1} = \frac{1}{2\pi i} \int_\Gamma (\lambda I - G(w))^{-1} \otimes (wI - hL_h)^{-1}\, dw. \tag{4.18}$$

Wobei Γ eine geschlossene Kurve in der w-Ebene ist, im Gebiet auf dem $(\lambda I - G(w))^{-1}$ analytisch ist und das Spektrum von hL_h umschließend [14]. Das Tensorprodukt $\otimes$ ergibt eine Matrix der Dimension sN_h aus einer Matrix von Dimension s und einer von Dimension N_h. $(\lambda I - G(w))^{-1}$ ist eine $s \times s$ matrixwertige Funktion von w, die analytisch ist in der w-Ebene, außer an einem Pol bei $\mu = r(\lambda)$. Dieser muss einfach sein, da wie schon erwähnt $(\lambda I - G(w))$ eine affine Funktion ist. Um das Integral auszuwerten, wählen wir Γ als eine Vereinigung einer Kurve Γ_1, ein kleiner Kreis der den Punkt μ in negativer Richtung umläuft, und Γ_2, ein Großer Kreis der μ und das Spektrum von hL_h umgibt. Das Residuum von $(wI - hL_h)^{-1}$ bei $w = \infty$ ist I und bezeichne $R(\mu)$ das Residuum von $(\lambda I - G(w))^{-1}$ bei $w = \mu$. Damit lassen sich die Teilintegrale einzeln auswerten und zwar ergibt sich $-R(\mu) \otimes (\mu I - hL_h)^{-1}$ für den Anteil von Γ_1 und $(\lambda I - G(\infty))^{-1} \otimes I$ für den Anteil von Γ_2. Wobei $(\lambda I - G(\infty))^{-1}$ ein Abkürzung für $\lim_{w\to\infty}(\lambda I - G(w))^{-1}$ ist. Insgesamt ergibt sich somit

$$(\lambda I - A_h)^{-1} = (\lambda I - G(\infty))^{-1} \otimes I - R(\mu) \otimes (\mu I - hL_h)^{-1}, \tag{4.19}$$

und daraus die Abschätzung

$$\left\|(\lambda I - A_h)^{-1}\right\| \leq C_4 + C_5 \left\|(\mu I - hL_h)^{-1}\right\|, \tag{4.20}$$

wobei $C_4 = \sup\limits_{z\in\Omega_z} \left\|(zI - G(\infty))^{-1}\right\|$ und $C_5 = \sup\limits_{z\in\Omega_z} \|R(r(z))\|$.
Beide Konstanten sind endlich, da die Suprema stetige Funktionen auf der Kompakten Menge $\overline{\Omega}_z$ beinhalten. Setzt man (4.16) in (4.20) ein, so erhält man

$$\left\|(\lambda I - A_h)^{-1}\right\| \leq C_4 + \frac{C_2 C_5}{\operatorname{dist}(\mu, S)} \tag{4.21}$$

Da S nach (V.1) beschränkt ist, gilt

$$\operatorname{dist}(\lambda, D) \leq C_6 \operatorname{dist}(\mu, S) \qquad \forall\ \lambda \in \Omega_z, \tag{4.22}$$

für eine Konstante C_6 unabhängig von λ. Da insbesondere noch $\operatorname{dist}(\lambda, D) \leq \tau_z$ für alle $\lambda \in \Omega_z$, ergibt sich aus (4.21) und (4.22)

$$\begin{aligned}
\left\|(\lambda I - A_h)^{-1}\right\| \operatorname{dist}(\lambda, D) &\leq C_4 \operatorname{dist}(\lambda, D) + \frac{C_2 C_5}{\operatorname{dist}(\mu, S)} \operatorname{dist}(\lambda, D) \\
&\leq C_4 \operatorname{dist}(\lambda, D) + \frac{C_2 C_5 C_6}{\operatorname{dist}(\mu, S)} \operatorname{dist}(\mu, S) \\
&\leq C_4 \tau_z + C_2 C_5 C_6. \\
&\Rightarrow \left\|(\lambda I - A_h)^{-1}\right\| \leq \frac{\widetilde{C}_2}{\operatorname{dist}(\lambda, D)} \qquad \forall \lambda \in \Omega_z,
\end{aligned}$$

mit $\widetilde{C}_2 = C_4 \tau_z + C_2 C_5 C_6$. Damit erhält man Gleichung (4.17) für $\lambda \in \Omega_z$. Um die restlichen Werte $\lambda \in \mathbb{C} \setminus \overline{D}$ zu behandeln, und zwar die mit $\operatorname{dist}(\lambda, D) \geq \tau_z$, schreibt man die Resolvente wie oben mit Hilfe des Dunford-Taylor-Integrals, diesmal in der z-Ebene

$$(\lambda I - A_h)^{-1} = \frac{1}{2\pi i} \int_\Gamma (\lambda - z)^{-1} (zI - A_h)^{-1}\, dz. \tag{4.23}$$

Wobei $\Gamma = \{z \in \mathbb{C} : |z| = 1 + \frac{1}{2}\tau_z\}$ eine geschlossene Kurze ist, die das Spektrum von A_h umschließt. Für $\operatorname{dist}(\lambda, D) \geq \tau_z$ und z auf Γ hat man die beiden Abschätzungen

$$\left|(\lambda - z)^{-1}\right| \leq \frac{2}{\operatorname{dist}(\lambda, D)}$$

und $$\left\|(zI - A_h)^{-1}\right\| \leq \frac{\widetilde{C}_2}{\operatorname{dist}(z, D)} = \frac{2\widetilde{C}_2}{\tau_z}.$$

Die Kurvenlänge von Γ ist gleich $2\pi(1 + \frac{1}{2}\tau_z)$, man erhält schließlich für das Integral (4.23) die Abschätzung

$$\left\|(\lambda I - A_h)^{-1}\right\| \leq \frac{4(1 + \frac{1}{2}\tau_z)\widetilde{C}_2}{\tau_z \operatorname{dist}(\lambda, D)} = \frac{C_2'}{\operatorname{dist}(\lambda, D)}.$$

Mit $C_2' = 4(1 + \frac{1}{2}\tau_z)\widetilde{C}_2/\tau_z$ gilt somit (4.17) auch für alle $\lambda \in \mathbb{C} \setminus \bar{D}$.

(ii) Für die umgekehrte Richtung nehmen wir diesmal an, dass μ nahe an S liegt und definieren dazu noch den Ring $\Omega_w = \{w \in \mathbb{C} : 0 < \mathrm{dist}(w, S) < \tau_w\}$. Aus (V.1) kann man zeigen, dass man ein τ_w klein genug auswählen kann, so dass für jedes $\mu \in \Omega_w$ eine Zahl $\lambda \in \Omega_z$ existiert die $r(\lambda) = \mu$, erfüllt, genauer nimmt man die Nullstelle λ von $r(\lambda) = \mu$ mit höchstem Betrag. Für alle diese $\mu \in \Omega_w$ gilt dann

$$\mathrm{dist}(\mu, S) \leq C_7 \mathrm{dist}(\lambda, D). \tag{4.24}$$

Dieses paar μ, λ erfüllt dann wieder (4.19) und daraus erhält man die Abschätzung

$$\left\|(\mu I - hL_h)^{-1}\right\| \leq C_8(C_4 + \left\|(\lambda I - A_h)^{-1}\right\|, \tag{4.25}$$

mit

$$C_8 = [\inf_{z \in \Omega_z} \sup_{1 \leq i,j \leq s} |R(r(z))_{ij}|]^{-1} \tag{4.26}$$

und C_8 ist beschränkt, da $R(r(z))$ eine stetige Funktion auf Ω_z ist. Nach Multiplikation von (4.25) mit $\mathrm{dist}(\mu, S)$ und unter Benutzung von (4.17) und (4.24) folgt weiter

$$\left\|(\mu I - hL_h)^{-1}\right\| \leq \frac{\overline{C}_2}{\mathrm{dist}(\mu, S)} \qquad \forall \mu \in \Omega_w,$$

mit $\overline{C}_2 = C_8 C_7 (C_4 \tau_z + C_2')$. In gleicher Weise wie für die andere Richtung, benutzt man das Integral

$$(\mu I - hL_h)^{-1} = \frac{1}{2\pi i} \int_\Gamma (\mu - w)^{-1} (wI - hL_h)^{-1} \, dw,$$

auf der Kurve $\Gamma = \{w \in \mathbb{C} : \mathrm{dist}(w, S) = \frac{1}{2}\tau_w\}$, um die letzte Abschätzung auf alle $\mu \in \mathbb{C} \setminus \bar{S}$ auszuweiten. So ergibt sich schließlich die Abschätzung

$$\left\|(\mu I - hL_h)^{-1}\right\| \leq \frac{2L\overline{C}_2}{\tau_w \mathrm{dist}(\mu, S)} = \frac{C_2}{\mathrm{dist}(\mu, S)},$$

mit $C_2 = 2L\overline{C}_2/\tau_w$ wobei L die Kurvenlänge von Γ ist. Damit gilt also (4.16) für alle $\mu \in \mathbb{C} \setminus \bar{S}$. □

Für endliche Zeitintervalle $[0, T]$ kann ein sehr ähnlicher Satz formuliert werden, hier ohne Beweis (vgl.[20]).

Satz 4.3: Sei (4.9) die Diskretisierung von (4.1), mit einer linearen Mehrschrittmethode die (V.1) und (V.2) erfüllt und genügend kleinem Zeitschritt h. Wenn

$$\|A_h^n\| \leq C_1, \qquad 0 \leq nh \leq T, \tag{4.27}$$

dann erfüllen die ε-Pseudoeigenwerte $\{\mu_\varepsilon\}$ von $\{hL_h\}$

$$\operatorname{dist}(\mu_\varepsilon, S) \leq C_2\varepsilon + C_3 h \qquad \forall \varepsilon \geq 0. \tag{4.28}$$

Umgekehrt folgt aus (4.28)

$$\|A_h^n\| \leq C_4(T) \min\{N_h, n\}, \qquad 0 \leq nh \leq T. \tag{4.29}$$

Die Konstanten C_i sind unabhängig von h und von Δx.

Die beiden letzten Sätze gelten nicht für lineare Mehrschrittverfahren deren Stabilitätsgebiete Spitzen haben. Nimmt man als Beispiel eine Spitze $w_0 \in S$ die im Spektrum von hL_h enthalten ist, dann gibt es ein λ im Spektrum von A_h mit $r(\lambda) = w_0$ und λ ist ein Eigenwert mit Betrag eins, jedoch nicht einfach. Somit kann die Beschränktheit von A_h^n nicht erfüllt sein [20].
Für Einschrittverfahren schreibt man die Diskretisierung (4.3) als

$$v_{n+1} = A_h v_n = \phi(hL_h)v_n, \qquad v_0 = f_h \tag{4.30}$$

wobei $\phi(w)$ ein rationale Funktion ist. Das Stabilitätsgebiet ist hier definiert durch

$$S = \{w \in \mathbb{C} : \phi(w) \in \overline{D}\}.$$

An das S soll folgende Voraussetzung gebunden sein.

(V.3) S ist beschränkt und $\phi'(w) \neq 0$ für $w \in \partial S$.

Diese Bedingung schließt A-stabile und andere implizite Methoden mit unbeschränkten Stabilitätsgebieten aus. Die Bedingung an die Ableitung ist gleichbedeutend dazu, dass $|\phi(w)|$ keinen Sattelpunkt auf ∂S hat.

(V.4) Es existiert ein nicht leeres Gebiet $V \in \mathbb{C}$ und eine Konstante $M < \infty$, so dass $\left\|(\mu I - hL_h)^{-1}\right\| \leq M$ für alle $\mu \in V$ und alle h.

Mit (V.3) und (V.4) vorausgesetzt, lassen sich zu Satz 4.2 und 4.3 analoge Sätze für Einschrittverfahren formulieren.

5 Anwendungen und Beispiele

Die ersten drei Beispielprogramme in diesem finalen Kapitel behandeln zeitabhängige partielle Differentialgleichungen mit einer spektralen Diskretisierung in Raumkoordinaten und einer Leapfrog-Diskretisierung in der Zeit, um die Rolle der Zeitschrittwahl mit Hilfe von Stabilitätsgebieten zu erläutern. Dabei handelt es sich um die Durchführung der numerischen Experimente wie sie von Trefethen in [25] aufgezeigt wurden. Die später in diesem Kapitel aufgeführten Programme sind an den Übungsaufgaben aus Kapitel 10 von [25] motiviert.
Das Leapfrog-Verfahren (zentrale Differenzen) ist ein wegen seiner Einfachheit beliebtes Verfahren, definiert durch

$$u_{n+2} - u_n = 2hf(u_{n+1}, t_{n+1}), \tag{5.1}$$

also ein explizites 2-Schritt-Verfahren. Es hat gewisse Nachteile, zum Beispiel, dass die Lösung an ungeraden Zeitschritten immer weiter von der Lösung an den geraden Zeitschritten abweicht [4]. Für die hier gemachten Versuche ist das Verfahren jedoch ausreichend. Außerdem kann man das zugehörige Stabilitätsgebiet S recht einfach bestimmen. Die Programme zu den drei Gleichungen werden mehrfach ausgeführt, mit einer jeweils immer höheren Zeitschrittweite. Dabei wird dann verglichen ob die Eigenwerte der räumlichen Diskretisierung multipliziert mit dem gewählten Zeitschritt noch in S liegt oder nicht. Mit anderen Worten, ob die in Kapitel vier vorgestellte Faustregel erfüllt ist oder nicht. Die drei dafür verwendeten Modellprobleme sind:

1) Wellengleichung mit variablem Koeffizient:

$u_t + c(x)u_x = 0$ auf $[-\pi, \pi]$, wobei $c(x) = \frac{1}{5} + \sin^2(x - 1)$,

mit periodischen Randbedingungen und Startwert $u(x, 0) = \exp(-100(x - 1)^2)$.

Fourier Spektralmethode mit FFT für die räumliche Disktretisierung, $N = 128$.

2) Wellengleichung 2. Ordnung:

$u_{tt} = u_{xx}, \quad x \in [-1, 1], \quad t > 0, \quad u(\pm 1) = 0.$

Chebyshev Spektralmethode mit FFT, $N = 80$.

3) Wellengleichung 2. Ordnung in 2D:

$u_{tt} = u_{xx} + u_{yy}, \quad x, y \in [-1, 1], \quad t > 0, \quad u = 0$ auf dem Rand

und mit Anfangswerten $u(x, y, 0) = \exp(-40(x - 0, 4)^2 + y^2), \qquad u_t(x, y, 0) = 0.$

Chebyshev Spektralmethode mit Ableitungsmatrix, $N = 30$.

Programm 1

```
% Gitter, Koeffizientenfunktion, Startwerte:
N = 128; x = (2*pi/N)*(1:N); t = 0; dt = 1.89/N;
c = .2 + sin(x-1).^2;
u1 = exp(-100*(x-1).^2); u0 = exp(-100*(x-.2*dt-1).^2);

ppsec = 6;                    % plots pro Sekunde
tmax = 8;
nplot = round((1/ppsec)/dt);
steps = round(tmax/dt);
data = u1; tdata = 0;
for n = 1:steps
   t = t+dt;
   u_hat = fft(u1);
   w_hat = 1i*[0:N/2-1 0 -N/2+1:-1] .*u_hat;
   w = real(ifft(w_hat));
   u2 = u0 - 2*dt*c.*w; u0 = u1; u1 = u2;  % Leapfrog-Verfahren
   if max(u1) > 5
      break                      % Abbruch, wenn Fehler zu groß
   end
   if mod(n, nplot) == 0
      data = [data;u1]; tdata = [tdata t];     % Näherungslösungen die gezeichnet werden
   end
end

waterfall(x,tdata,data), colormap([0 0 0]), view(10,60)
axis([0 2*pi 0 tmax -1 3]), ylabel t, zlabel u, grid off
```

Programm 2

```
 % Gitter, Ableitungsmatrix und Zeitschritte
N = 80; x = cos(pi*(0:N)/N); t=0; dt = 9.1/N^2;
u0 = exp(-200*x.^2)';

%Euler-Schritt für Startwert
 u1 = u0 + dt*chebfft(u0); u1(1) = 0; u1(N+1) = 0;

ppsec = 15;                    % plots pro Sekunde
tmax = 4;
nplot = round((1/ppsec)/dt);
steps = round(tmax/dt);          % runden auf ganzzahlige Werte für Schleife
data = [u0';u1';zeros(floor(steps/nplot),N+1)];
tdata = [0 dt];
i=1;
for n = 1:steps
t = t+dt;
w = chebfft(chebfft(u1)); w(1) = 0; w(N+1) = 0;
u2 = 2*u1 - u0 + dt^2*w; u0 = u1; u1 = u2;
   if mod(n, nplot) == 0
      i=i+1;
      data(i,:) = u2' ; tdata = [tdata t];          % Näherungslösungen die gezeichnet werden
   end
end

waterfall(x,tdata,data)
axis([-1 1 0 tmax -2 2]), view(10,70), grid off
colormap([0 0 0]), ylabel t, zlabel u,
```

Programm 3

```
% Gitter und Anfangswerte
N = 30; [D,x] = cheb(N); D2 = D^2; D2 = D2(2:N,2:N);
y = x'; dt =6.4/N^2; [xx,yy] = meshgrid(x,y);

tmax=1;
steps = round(tmax/dt);
vv = exp(-40*((xx-.4).^2 + yy.^2));        % Startwert
vvold = vv;
for n = 1:steps
t = n*dt;
M=abs(min(vv,[],'all'));
if M > 10
      break   % Abbruch wenn Fehler zu groß
end
uxx = zeros(N+1,N+1); uyy = zeros(N+1,N+1);
ii = 2:N;
   for i = 2:N
   uxx(i,ii) = D2*vv(i,2:N)';
   end
   for j = 2:N
   % 2. Ableitung nach y in jeder Spalte
  uyy(ii,j) = D2*vv(2:N,j);
   end
  vvnew = 2*vv - vvold + dt^2*(uxx+uyy);
  vvold = vv; vv = vvnew;
end
% Plot nach 1 s oder nach Abbruch
  [xxx,yyy] = meshgrid(-1:1/16:1,-1:1/16:1);
   vvv = interp2(xx,yy,vv,xxx,yyy,'spline');
   mesh(xxx,yyy,vvv), axis([-1 1 -1 1 -1 1])
   colormap([0 0 0]), title(['t=' num2str(t)]), drawnow
```

In Programm 3 erfolgt die Ableitung über die Chebyshev-Ableitungsmatrix zweiter Ordnung, da dies für kleine N immer noch schneller ist als mit einer FFT, und der Programmcode dadurch deutlich übersichtlicher wird. Die gesuchte Lösung ist hier eine Funktion im $\mathbb{R}^2$, daher wird ein zweidimensionales Gitter gebildet, und anschließend entlang jeder Gitterlinie in x und y-Richtung abgeleitet. Für Programm 2 wird zur Veranschaulichung trotzdem mit einer Chebyshev-FFT gearbeitet.
Die Länge der Zeitschritte wird nun in jedem Programmdurchlauf sukzessive erhöht, bis die erste Instabilität auftaucht. Für Programm 1 ergibt das eine empirische Stabilitätsrestriktion von $h < 1.9N^{-1}$ und für Programm 2 und 3 $h < 9.2N^{-2}$ bzw. $h < 6.5N^{-2}$. In *Abb.5.3* sieht man das Ergebnis für Programm 1 nach Überschreiten der Zeitschrittrestriktion. Offensichtlich treten hier Instabilitäten auf, in der Weise, dass kleine Fehler exponentiell verstärkt werden. In den anderen Programmen ergeben sich sehr ähnliche Effekte bei Erreichen der oben angegebenen Zeitschrittweiten.
Um zu prüfen ob die in Kapitel vier aufgestellte Faustregel bei den oben genannten drei Beispielen tatsächlich die richtigen Vorhersagen macht, werden zunächst die entsprechenden Stabilitätsgebiete benötigt. Zuerst für das Problem der Wellengleichung mit variablem Koeffizient. Die Leapfrog-Formel auf das Testproblem $u_t = \lambda u$ angewendet, ergibt

$$u_{n+2} - u_n = 2h\lambda u_{n+1}. \tag{5.2}$$

Die zugehörige charakteristische Gleichung ist $z^2 - z^0 = 2\lambda h z$. Um diese Gleichung besser handhaben zu können formen wir zu $z - z^{-1} = 2\lambda h$ um. Nun suchen wir Nullstellen dieser Gleichung auf der geschlossenen Einheitskreisscheibe, wobei auf dem Einheitskreis nur einfache Nullstellen zugelassen sind. Mit Hilfe der p-q-Formel sieht man, dass für Gleichungen dieser Form gilt, wenn z_1 eine Nullstelle ist, dann ist $z_2 = -z_1^{-1}$ die zweite Nullstelle. Daraus ergibt sich, dass wenn $|z_1| < 1$ dann $|z_2| > 1$, womit z_2 außerhalb des Einheitskreises liegen würde. Also ist für die Stabilität $|z| = 1$ nötig und um mehrfache Nullstellen auf dem Einheitskreis zu vermeiden muss auch $z \neq -z^{-1}$ gelten, das bedeutet $z \neq \pm i$. Die gesuchten Werte liegen also auf dem Einheitskreis ohne $\pm i$. Da für komplexe Zahlen auf dem Einheitskreis die Inverse gleich dem komplex Konjugierten ist, ist $z - z^{-1}$ rein imaginär und genauer gilt hier $z - z^{-1} \in (-2i, 2i)$. Es herrscht also Stabilität wenn $2\lambda h$ in $(-2i, 2i)$ also λh in $(-i, i)$ und dieses Intervall ist das gesuchte Gebiet (genauer gesagt ist es das Stabilitätsintervall).
Nun werden noch die Eigenwerte der Diskretisierung der Wellengleichung benötigt. Zuerst ohne Koeffizienten $c(x)$ betrachtet, das heißt für $u_t + u_x = 0$. Die Eigenwerte der Fourier-Ableitungsmatrix D_F sind die Zahlen ik mit $k = N/2 + 1, \ldots, N/2 - 1$ und 0 mit der Vielfachheit zwei. Der größte Eigenwert $i(N/2 - 1)$ multipliziert mit h muss also nach der Faustregel im Intervall $(-i, i)$ liegen, das bedeutet $h(N/2 - 1) < 1$ muss erfüllt sein, bzw. für große N näherungsweise $h \leq 2N^{-1}$.
Für die Gleichung $u_t + c(x)u_x = 0$ muss jetzt noch der Koeffizient $c(x) = \frac{1}{5} + \sin^2(x - 1)$ mit berücksichtigt werden. $c(x)$ hat ein Maximum, wie man leicht nachrechnet, bei $\pi/2 + 1$ und $-\pi/2 + 1$ mit dem Wert $6/5$. Für große N werden also die Eigenwerte von $c(x_i) \cdot D_F$ etwa $6/5$ mal so groß sein wie von D_F. Das ergibt in etwa die Stabilitätsbedingung $h \leq \frac{5}{3}N^{-1}$, etwas strikter als die beobachteten $1.9N^{-1}$ im obigen Versuch. Das liegt daran, dass für kleine N die tatsächlichen Eigenwerte von den theoretisch vorhergesagten abweichen. Mit $N = 128$ beträgt der größte Eigenwert numerisch bestimmt etwa $68,5i$, das ergäbe eine Restriktion von $h \leq 1,86N^{-1}$, schon recht nahe an dem beobachteten Wert. Für größere N wird die Übereinstimmung jedoch besser. In *Abb.5.1* sieht man das Verhältnis des vorhergesagten größten Eigenwertes und des tatsächlichen, numerisch bestimmten, letzteres berechnet mit der eig()-Funktion in MATLAB. Mit wachsendem N nähern sich diese beiden Werte immer mehr aneinander an.

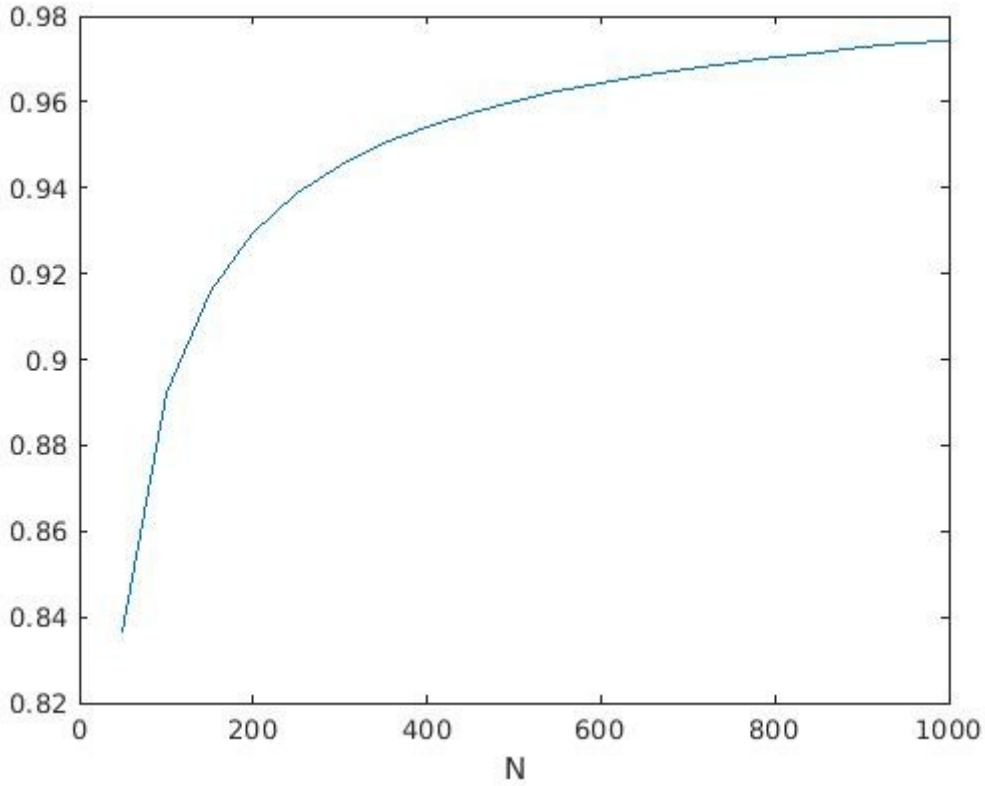

Abb. 5.1: Verhältnis vom vorhergesagten zum tatsächlichen Eigenwert der Diskretisierung von Programm 1.

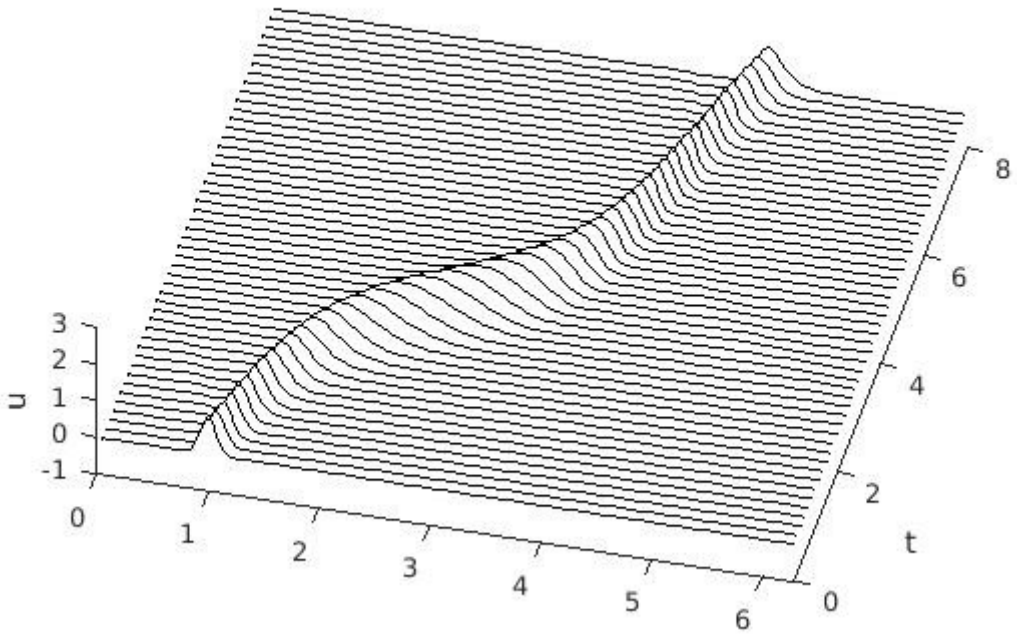

Abb.5.2: Lösung von Programm 1, stabil mit $h = 1.89/N$.

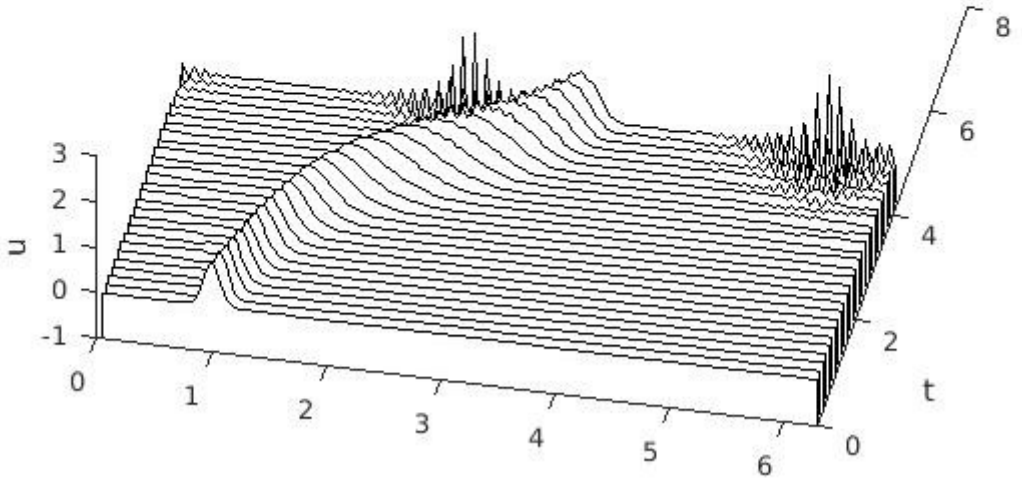

Abb.5.3: Programm 1 mit $h = 1,9/N$.

In *Abb.5.3* erkennt man, dass die Oszillation in dem Bereich auftritt wo $c(x)$ ihre Maxima annimmt. Die Berechnung wurde bei Eintreten der Instabilität gestoppt, für weitere Zeitschritte wächst der Fehler exponentiell an. Die ersten Moden die instabil werden, sind die mit der höchsten Wellenzahl, diese erzeugen das zu beobachtende Sägezahnmuster. *Abb.5.4* zeigt den Eigenvektor zum größten Eigenwert, man erkennt die Ähnlichkeit zur instabilen Mode in *Abb.5.3.*

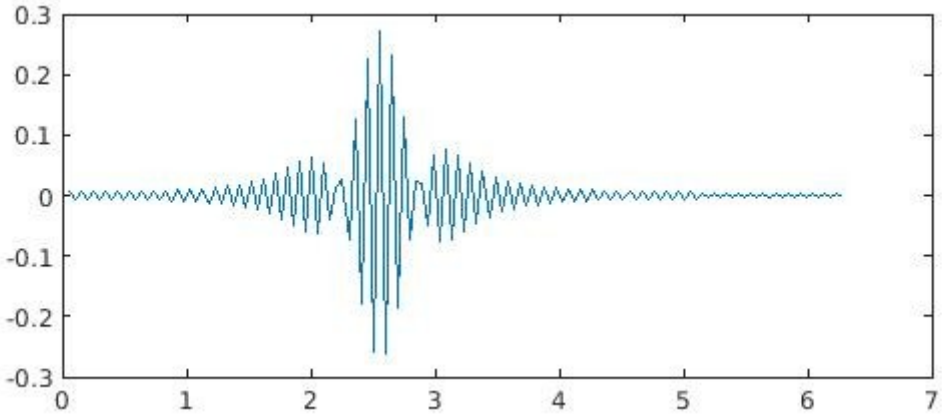

Abb.5.4: Eigenmode zum größten Eigenwert $i(N/2-1)$.

In den Programmen 2 und 3 wurde für die Zeitdiskretisierung die Leapfrog-Formel für die zweite Ableitung benutzt. Hierfür gilt es wieder die Stabilitätsgebiete zu bestimmen. Das Modellproblem

für Gleichungen zweiter Ordnung lautet $u_{tt} = \lambda u$, wendet man hierauf den zentralen Differenzenquotienten der zweiten Ableitung an, ergibt sich

$$u_{n+2} - 2u_{n+1} + u_n = h^2 \lambda u_{n+1}. \tag{5.3}$$

Die zugehörige charakteristische Gleichung schreiben wir wieder in der Form $z + z^{-1} = \lambda h^2 + 2$. Man sieht leicht ein, dass wenn z_1 eine Nullstelle ist, dann ist $z_2 = z_1^{-1}$ die andere. Mit sehr ähnlicher Argumentation wie oben erhält man, dass $|z| = 1$ und $z \neq \pm 1$ sein muss. Insgesamt ist also $z + z^{-1} \in (-2, 2)$ und damit $\lambda h^2 \in (-4, 0)$.
Die Eigenwerte von D_C^2 wurden bereits in Kapitel eins Angegeben. Für Programm 2 ist die Stabilitätsrestriktion demzufolge näherungsweise gegeben durch $-0,048N^4h^2 \leq -4$ bzw. $h \leq 9,2N^{-2}$, in Übereinstimmung mit dem beobachteten Ergebnis.
Die größten Eigenwerte in Programm 3 sind in etwa zweimal so groß wie die in Programm 2, das heißt die Stabilitätsbedingung ist hier zweimal so strikt für h^2 und somit $\sqrt{2}$ mal so strikt für h, also $h \leq 6,5N^{-2}$, in Übereinstimmung mit der empirisch bestimmten Restriktion.

Wellengleichung erster Ordnung

Ein Beispiel bei dem es nicht ausreicht wenn nur die Eigenwerte innerhalb von S liegen ist das folgende

$$u_t = u_x, \qquad x \in [-1, 1], \qquad 0 < t < 1,$$

mit Anfangsbedingung $u(x, 0) = \exp(-60(x - 1/2)^2)$ und Randwert $u(1, t) = 0$.
Zur Diskretisierung in Raumkoordinaten, wird die Chebyshev-Ableitungsmatrix D_C verwendet, ohne die erste Spalte und Zeile in D_C, um die Randbedingung mit einzubinden. Die Ausreißer unter den Eigenwerten dieser Matrix erschweren die Zeitschrittwahl und es wird nicht ausreichen wenn die Eigenwerte von hD_C in S liegen, um eine stabile Lösung zu erhalten. Hier werden wir also noch die Pseudospektren mit einbeziehen müssen. Für die Zeitintegration wird das Adams-Bashforth-Verfahren dritter Ordnung verwendet. Für die Initialisierungsphase müssen u_1 und u_2 vom Startwert u_0 durch eine andere Methode bestimmt werden, wir wählen dazu das einfach zu implementierende Euler-Vefahren.
Abb.5.5 zeigt die Eigenwerte der verwendeten Diskretisierungsmatrix für $h = 7/N^2, N = 60$ und die ε-Pseudospektren. Obwohl die Eigenwerte im Stabilitätsgebiet des Verfahrens liegen, befinden sich die Pseudospektren sogar für ε nahe an null, zum Teil recht weit außerhalb, wodurch sich die in *Abb.5.6* zu beobachtende Instabilität am Rand erklären lässt.

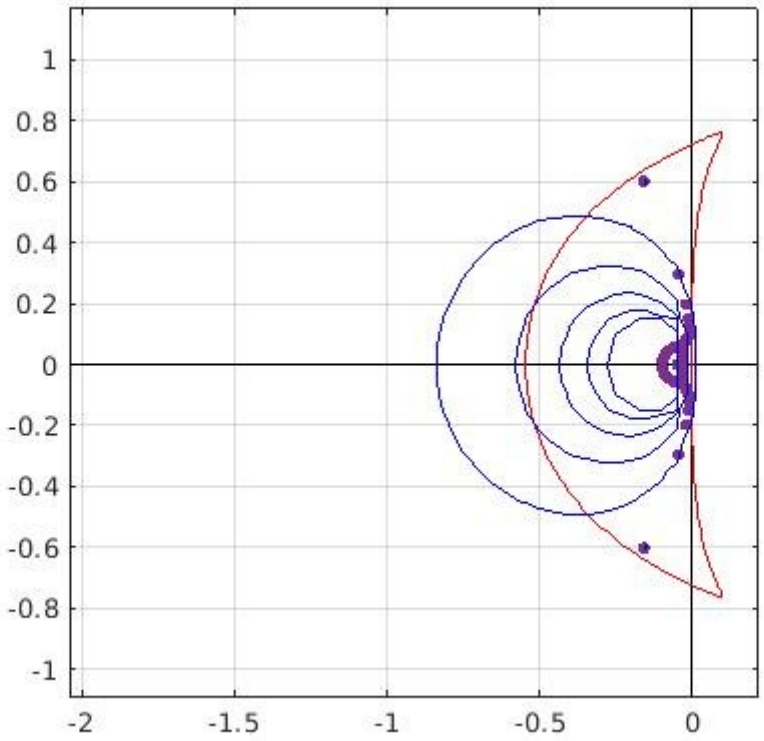

Abb.5.5: Die Eigenwerte sind hier als Punkte erkennbar, das Stabilitätsgebiet ist rot umrandet. Die blaue Kurven sind von außen nach innen die ε-Pseudospektren für $\varepsilon = 10^{-2}, 10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}$.

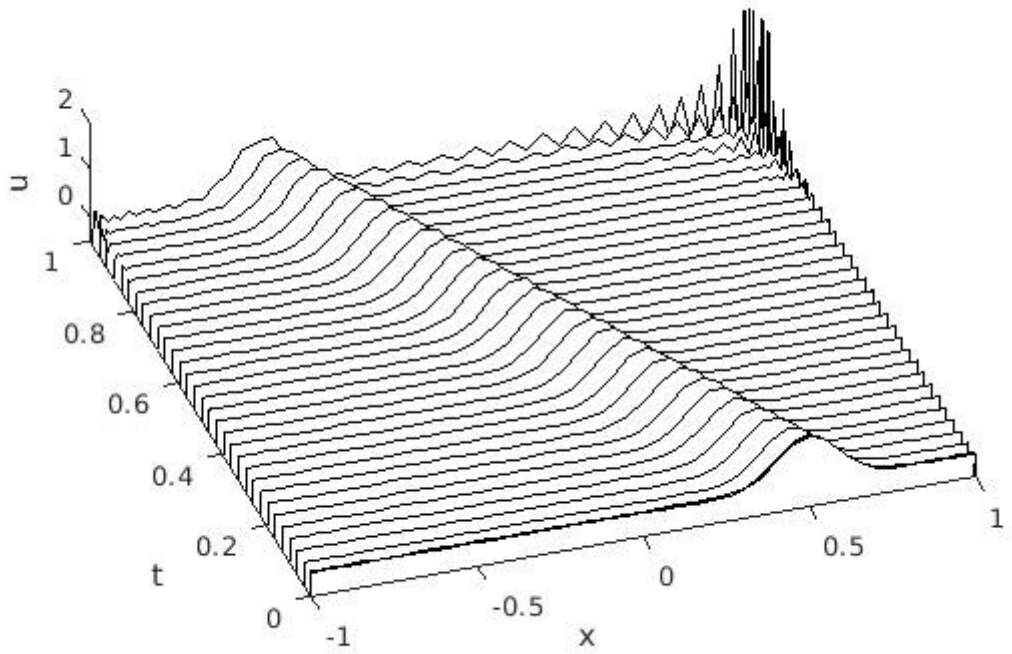

Abb.5.6

Programmcode Wellengleichung erster Ordnung

```
% Gitter und Ableitungsmatrix
N = 60; [D,x] = cheb(N); D = D(2:N+1,2:N+1); dt = 7/N^2;
e = eig(dt*D);

% Starwerte mit Euler-Verfahren
u0 = exp(-60*(x(2:N+1)-0.5).^2);
w0 = D*u0;
u1 = u0 + dt*w0;
w1 = D*u1;
u2 = u1 + dt*w1;
w2 = D*u2;

t = 0;
tmax = 1;
ppsec = 30;
nplot = round((1/ppsec)/dt);
steps = round(tmax/dt);
dim = floor(steps/nplot);
data = [[0; u0]'; [0; u1]'; [0; u2]';zeros(dim-2,N+1)];
tdata = [0 dt 2*dt];
j=3;
for n = 3:steps
   t = t+dt;
   u3 = u2 + 1/12*dt*(23*w2 - 16*w1 + 5*w0);
   u2 = u3; w2 = D*u2; w1 = w2;  w0 = w1;
     if mod(n, nplot) == 0
       j=j+1;
       data(j,:) = [0; u3]'; tdata = [tdata t];
      end
end

f1 = figure
% Stabilitätsgebiet
plot ([-8 8],[0 0],'black'), hold on, plot ([0 0],[-8 8],'black' )
z = exp(1i*pi*(0:200)/100); r = z-1;
s = (23-16./z+5./z.^2)/12; plot(r./s,'r')
axis([-2.5 .5 -1.5 1.5]), axis square, grid on

% Eigenwerte
plot(e,'.','markersize',10), hold on

% Pseudospektren
xx = -2:0.05:0.5; yy = -1:0.05:1; [X,Y] = meshgrid(xx,yy); Z = X+1i*Y;
I = eye(N); sigmin = zeros(length(yy),length(xx));
for j = 1:length(xx)
   for i = 1:length(yy), sigmin(i,j) = min(svd(Z(i,j)*I-dt*D)); end
end
contour(xx,yy,sigmin,10.^(-6:-2),'b')

f2=figure
waterfall(x,tdata,data), colormap ([0 0 0])
axis([-1 1 0 1 -.5 2])
xlabel x, ylabel t, zlabel u, grid off
pbaspect([5 5 1])
```

Integrierender Faktor

Um eine Lockerung der Restriktion an die Zeitschritte zu erhalten, bietet sich in bestimmten Fällen die Methode des integrierenden Faktors an. Dabei handelt es sich um eine Transformation der Unbekannten einer Differentialgleichung. Das meist bekannte Beispiel betrifft eine gewöhnliche Differentialgleichung erster Ordnung [1, 4]:

$$u_t + pu = f$$

Man multipliziert zuerst mit dem integrierenden Faktor $e^P,\ P = \int p\ dt$:

$$e^P u_t + e^P pu = e^P f.$$

Mit der Produktregel vereinfacht sich dies zu

$$(e^P u)_t = e^P f.$$

Durch Integration erhält man dann

$$e^P u = \int e^P f\ dt.$$

Um die Lösung u zu erhalten muss anschließend durch den integrierenden Faktor dividiert werden, bzw. mit e^{-P} multipliziert.
Für zeitabhängige partielle Differentialgleichungen

$$u_t = Lu + M(u)$$

ist die Methode ebenfalls anwendbar, wobei L ein linearer Differentialoperator mit nur räumlichen Ableitungen ist und M ein (möglicherweise nichtlinearer) Operator, der die restlichen Terme der Gleichung enthält. Der integrierende Faktor ist hier e^{-Lt} und die vereinfachte Gleichung lautet damit

$$(e^{-Lt} u)_t = e^{-Lt} M(u).$$

Der Vorteil besteht darin, dass der lineare Anteil nicht mehr vorhanden ist, außer im Exponentialteil. Das Ziel ist es, sehr kleine Zeitschritte zu vermeiden ohne auf z.B. implizite Methoden zurückzugreifen. Dies ist jedoch nur sinnvoll, wenn durch den integrierenden Faktor der (oft als steif bezeichnete) Teil der Gleichung eliminiert wird, das heißt derjenige der die kleinen Zeitschritte erfordert, um wirklich einen Geschwindigkeitsvorteil zu erhalten.

KdV-Gleichung

Um die eben behandelte Methode auszuprobieren, betrachten wir die Korteweg-de-Vries-Gleichung (KdV-Gleichung), eine Gleichung von der Form

$$\begin{aligned} &u_t + uu_x + u_{xxx} = 0 \qquad x \in (-\infty, \infty), \qquad t > 0, \\ &u(x,0) = u_0(x) \qquad x \in (-\infty, \infty), \end{aligned} \tag{5.4}$$

bestehend aus einem nichtlinearen hyperbolischen Term uu_x und einem linearen Term u_{xxx}. Unter den Lösungen dieser Gleichung gehören wandernde Wellen der Form [25]

$$u(x,t) = 3a^2 sech^2(\frac{1}{2}a(x - x_0) - a^3 t), \tag{5.5}$$

mit a und x_0 reell. Mit $sech$ ist die Funktion Sekans hyperbolicus gemeint. Die Amplitude dieser Welle ist $3a^2$ und die Geschwindigkeit $2a^2$, also ist die Geschwindigkeit proportional zur Amplitude, anders als bei linearen Wellengleichungen wo die Geschwindigkeit unabhängig von der Amplitude ist. Darüber hinaus ist zu beachten, dass u vom Punkt $x = x_0 + 2a^2 t$ schnell abnimmt, die Welle also örtlich konzentriert ist.
Die oben aufgeführten Lösungen sind sogenannte Solitone. Dies sind Lösungen die Wellen repräsentieren, die ihre Form während der Bewegung lange Zeit beibehalten und auch bei Interaktion mit anderen Solitonen die Form nicht verändern. Wellen die nur die erste Eigenschaft aufweisen, aber bei Interaktion mit anderen Wellen ihre Form verändern werden auch solitäre Wellen genannt. Wenn mehrere Solitone auf einem Gebiet mit unterschiedlichen Geschwindigkeiten auftreten, kann es zu Kollisionen kommen. Dabei vereinigen sie sich für einen kurzen Moment, während die schnellere Welle dann ihren Kurs fortsetzt, die langsamere also überholt. Beide Wellen bewegen sich dann weiterhin in die selbe Richtung wie vorher, wobei ihre Geschwindigkeiten und Form erhalten bleibt. Durch die Kollision kommt es jedoch zu einer Phasenverschiebung der Solitone, das heißt die Wellen verschieben ihre Position in horizontaler Richtung [8].
Wir wollen das Problem (5.4) mit Hilfe einer Fourier Spektralmethode lösen, da keine Randbedingungen gegeben sind und die vorliegenden Lösungen exponentiell zerfallen, ist dies angemessen. Zur Zeitdiskretisierung wird das 4-stufige Runge-Kutta-Verfahren benutzt. Die Nichtlinearität stellt kein großes Problem für dieses explizite Verfahren dar. Um Stabilität zu gewährleisten wäre jedoch trotzdem eine sehr kleine Zeitschrittweite notwendig. Der lineare Term in Gleichung (5.4) ist der steife Anteil, da dieser die hohen Frequenzen enthält und deshalb wäre hier eine geringere Schrittweite zur Auflösung notwendig. Um dies zu umgehen wollen wir die Gleichung unter Einsatz des integrierenden Faktors umformen, was eine fünf bis zehn mal größere Wahl der Zeitschrittweite ermöglicht, ohne Stabilitätsprobleme zu erwarten [25].
Da wie gesagt $u(x,t)$ für $|t| \to \infty$ der null annähernd exponentiell kleiner wird, kann das unendliche Intervall in ein endliches $(-\pi L, \pi L),\ L > 0$ verkürzt werden. So kann man die Randbedingungen mit den periodischen Randbedingungen approximieren. Um eine Fouriertransformation auf diesem Intervall durchführen zu können, benötigen wir eine Koordinatentransformation von $[-\pi L, \pi L]$ auf $[0, 2\pi]$ [21]:

$$y = \frac{x}{L} + \pi, \qquad y \in [0, 2\pi],$$
$$x = L(y - \pi), \qquad x \in [-\pi L, \pi L]$$

und schreiben $v(y,t) = u(x,t), \qquad v_0(y) = u_0(x).$

Die transformierte KdV-Gleichung lautet dann

$$v_t + \frac{1}{L} v v_y + \frac{1}{L^3} v_{yyy} = 0, \qquad y \in (0, 2\pi), \qquad t > 0,$$
$$v(y,0) = v_0(y), \qquad y \in (0, 2\pi)$$

und $v(\cdot, t)$ ist periodisch auf $[0, 2\pi)$.
Im Weiteren approximieren wir $v(\cdot, t)$ mit $v(x,t) \approx \sum_{-N/2}^{N/2} \widehat{v}_k(t) e^{iky}$, und zur Vereinfachung nutzen wir noch die Umformung $vv_y = \frac{1}{2}(v^2)_y$. Nimmt man nun das Skalarprodukt mit e^{iky} so erhält man mit Hilfe von (1.1)

$$\frac{d}{dt}(\widehat{v}_k) - \frac{ik^3}{L}\widehat{v}_k + \frac{ik}{2L}(\widehat{v^2})_k = 0, \qquad k = -N/2, \ldots, N/2,$$

mit der Anfangsbedingung

$$\widehat{v}_k(0) = (v_0(y), e^{iky}) = \frac{1}{2\pi}\int_0^{2\pi} v_0(y) e^{-iky} dy.$$

Die Fourierkoeffizienten $\widehat{v}_k$ und $(\widehat{v^2})_k$ werden mit einer FFT berechnet. Der integrierende Faktor ist hier $e^{-ik^3 t/L^3}$. Nach Multiplikation mit diesem und Umformung mittels Produktregel, kann die obige Gleichung also weiter umgeschrieben werden zu

$$\frac{d}{dt}(e^{-ik^3 t/L^3}\widehat{v}_k) = \frac{ik}{2L} e^{-ik^3 t/L^3}(\widehat{v^2})_k, \qquad k = -N/2, \ldots, N/2.$$

Der lineare Term ist somit verschwunden und das Problem ist nicht mehr steif. Der Nachteil ist jedoch der hinzu multiplizierte integrierende Faktor mit k^3 im Exponenten.
Damit noch $\widehat{u^2}$ aus der Gleichung eliminiert wird, benutzen wir eine Hintereinanderausführung von iFFT und FFT (man beachte dass $\widehat{u^2}$ die Fouriertransformierte von u^2 ist und nicht etwa das Quadrat der Fouriertransformierten $\hat{u}$). Sei $\widehat{V}$ der Vektor der als Komponenten die Fourierkoeffizienten $\widehat{v}_k$ enthält. Wir schreiben als Abkürzung $g = e^{-ik^3 t/L^3}$. Dann ist mit $\widehat{U} = g. * \widehat{V}$ die Gleichung als Programmcode geschrieben

$$\frac{d}{dt}\widehat{U} = -\frac{ik}{2}. * g. * \mathsf{fft}([\mathsf{ifft}(g^{-1}. * \widehat{U})]^2). \tag{5.6}$$

Wir lösen (5.6) mit dem 4-stufigen Runge-Kutta-Verfahren:

$$u_{n+1} = u_n + \tfrac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

mit $k_1 = f(t_n, u_n)$, $k_2 = f(t_n + \frac{h}{2}, u_n + \frac{h}{2}k_1)$,
$k_3 = f(t_n + \frac{h}{2}, u_n + \frac{h}{2}k_2)$, $k_4 = f(t_n + h, u_n + hk_3)$.

Dies jedoch nicht wie hier aufgeschrieben mit der Zeit t_n, sondern nur mit dem Zeitschritt h bzw. $h/2$ und aktualisieren die Formel von einem Zeitschritt zum nächsten. Dadurch erhöht man nochmals die Stabilität [13]. Die daraus resultierende Lösung muss nun mit der Inversen des integrierenden Faktors e^{ik^3t} multipliziert werden um $\widehat{V}$ zu erhalten. Da das ganze im Fourierraum berechnet wurde, muss wieder in den physikalischen Raum rücktransformiert werden, das geschieht entsprechend mit einer iFFT.
Der oben aufgeführte Runge-Kutta-Algorithmus ist eher kostspielig, da die 4-stufige Version eine Funktionsauswertung von f viermal für jeden Zeitschritt erfordert, somit ist diese Methode oft aufwendiger als Verfahren die ohne Auswertung von f auskommen. Nichtsdestotrotz hat dieses Verfahren erhebliche Vorteile. Zunächst einmal ist das 4-stufige Runge-Kutta-Verfahren von vierter Ordnung, das bedeutet es hat einen abnehmenden Fehler von der Ordnung $O(h^4)$ und normalerweise möchte man die Genauigkeit, die man erhält wenn man die räumliche Ableitung mit einer Spektralmethode bestimmt, nicht abgeben durch ein ungenaues Zeitschrittverfahren. Auch ist es im Vergleich zu anderen expliziten Methoden für die Berechnungen von Wellenausbreitung mit eher langen Zeitschritten stabil. Des Weiteren benötigen Runge-Kutta-Algorithmen zum starten keine anderen Verfahren um Startwerte zu generieren, wie es bei Mehrschrittverfahren der Fall ist [4].
Als Anwendung der KdV-Gleichung simulieren wir die Kollision zweier Solitone, die sich in die gleiche Richtung bewegen. Die Anfangsbedingung soll die Summe zweier Funktionen der Form (5.5) bei $t = 0$ sein:

$$3a^2 sech^2(\tfrac{1}{2}a(x+2)) + 3b^2 sech^2(\tfrac{1}{2}b(x+1)),$$

mit $a = 25$ und $b = 16$.
Die Welle dargestellt durch den linken Term hat die größere Amplitude und wegen oben erwähnter Proportionalität auch die höhere Geschwindigkeit. Die größere Welle startet im Punkt $x_0 = -2$ und die kleinere im Punkt $x_0 = -1$. Die größere (schnellere) Welle wird also die kleine irgendwann einholen und mit ihr kollidieren. Wie in *Abb.5.7* zu sehen ist, kommt es dadurch zu der schon erwähnten Phasenverschiebung.
Die Berechnung erfolgt mit einer Zeitschrittweite von $0.4/N^2$. Ohne integrierenden Faktor wäre aufgrund der Eigenwerte des Diskretisierungsoperators dritter Ordnung, eine Restriktion der Größenordnung $O(N^3)$ zu erwarten. Erst eine Erhöhung auf $0.43/N^2$ führt zu einem Zackenmuster, ähnlich wie zuvor in den Programmen 1 bis 3 zu beobachten.

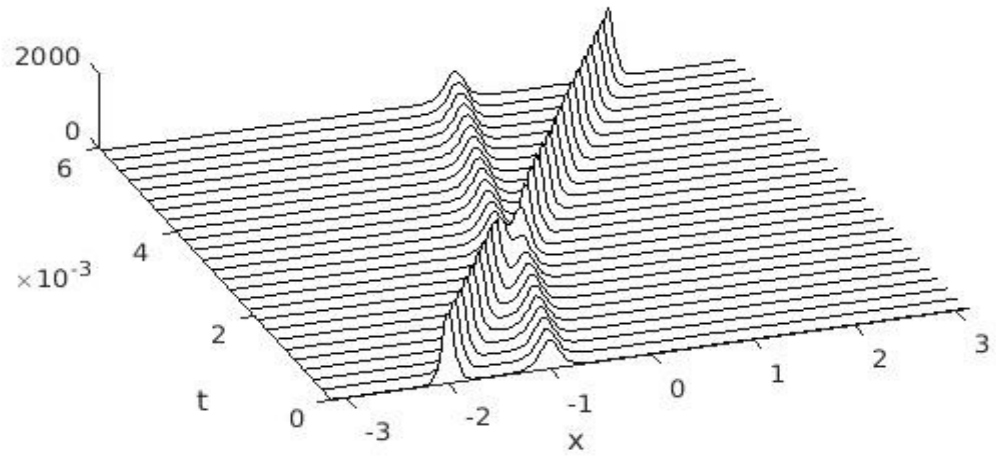

Abb.5.7

Programmcode KdV-Gleichung

```
% Erstelle Gitter und Anfangswerte:
N = 256; dt = 0.4/N^2; x = (2*pi/N)*(-N/2:N/2-1)';
A = 25; B = 16; clf, drawnow
u = 3*A^2*sech(.5*(A*(x+2))).^2 + 3*B^2*sech(.5*(B*(x+1))).^2;
v = fft(u); k = [0:N/2-1 0 -N/2+1:-1]'; ik3 = 1i*k.^3;

% Löse PDGl und plotte Lösung:
tmax = 0.006; nplt = floor((tmax/25)/dt); nmax = round(tmax/dt);
udata = u; tdata = 0;
for n = 1:nmax
  t = n*dt; g = -.5i*dt*k;
  E = exp(dt*ik3/2); E2 = E.^2;
  a = g.*fft(real( ifft(     v    ) ).^2);
  b = exp(-dt*ik3/2).*g.*fft(real( ifft(E.*(v+a/2) )).^2);     % Runge-Kutta
  c = exp(-dt*ik3/2).*g.*fft(real( ifft(E.*(v + b/2)) ).^2);    % 4. Ordnung
  d = exp(-dt*ik3).*g.*fft(real( ifft(E2.*(v+c)) ).^2);
  v = E2.*v + E2.*(a + 2*(b+c) + d)/6;
  if mod(n,nplt) == 0
    u = real(ifft(v));
    udata = [udata u]; tdata = [tdata t];
  end
end

waterfall(x,tdata,udata'), colormap([0 0 0]);
view(-20,25)
xlabel x, ylabel t, axis([-pi pi 0 tmax 0 2000]), grid off
pbaspect([1 1 .13])
```

Kuramoto-Sivashinsky-Gleichung

Die Kuramoto-Sivashinsky(KS)-Gleichung ist eine nichtlineare Gleichung vierter Ordnung, welche die interessante Eigenschaft besitzt chaotische Lösungen aufzuweisen. Aufgeschrieben lautet sie [21]:

$$u_t + (u^2)_x + u_{xx} + u_{xxxx} = 0 \qquad x \in (-\infty, \infty), \qquad t > 0,$$
$$u(x,0) = u_0(x) \qquad x \in (-\infty, \infty).$$

Wobei auch wieder uu_x statt $(u^2)_x$ geschrieben werden kann. Wir benutzen wieder einen integrierenden Faktor im Fourierraum. Mit einer exponentiell kleiner werdenden Funktion u können wieder periodische Randbedingungen eingeführt werden. Die Anfangsbedingung sei durch die Funktion $u(x,0) = \exp(-x^2)$ gegeben. Nach einer Koordinatentransformation und Approximation mittels einer Fourierreihe, erhält man in gleicher Weise wie bei der KdV-Gleichung folgende Unformung:

$$\frac{d}{dt}\widehat{v}_k - \frac{k^2}{L^2}\widehat{v}_k + \frac{k^4}{L^4}\widehat{v}_k = -\frac{ik}{L}\widehat{(v^2)}_k,$$

bzw.

$$\frac{d}{dt}\widehat{v}_k + \left(\frac{k^4}{L^4} - \frac{k^2}{L^2}\right)\widehat{v}_k = -\frac{ik}{L}\widehat{(v^2)}_k.$$

Der integrierende Faktor ist also $e^{(k^4/L^4 - k^2/L^2)t}$. Zur Implementierung muss nur der Programmcode der KdV-Gleichung entsprechend umgeschrieben werden.

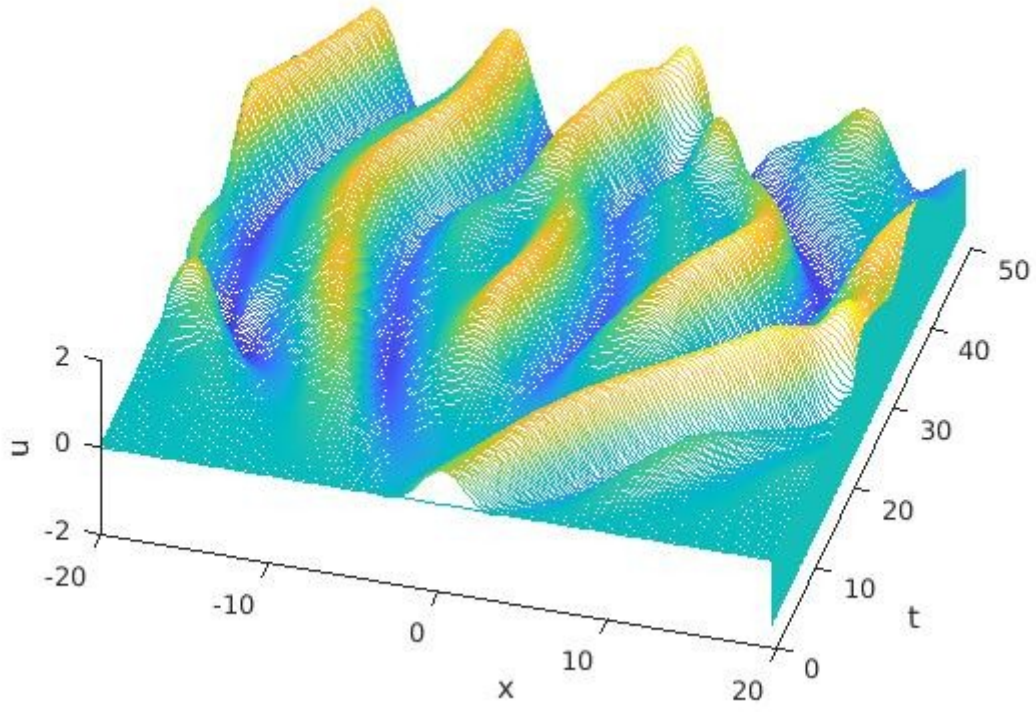

Abb.5.8: KS-Gleichung

Programmcode KS-Gleichung

```
L = 20/pi;
N = 128;
tmax = 50; dt = 1/10;
ppsec = 5;

x = (2*pi*L/N)*(-N/2:N/2-1);
u = exp(-x.^2);
v = fft(u);
k = [0:N/2-1 0 -N/2+1:-1]/L;
nplot = round((1/ppsec)/dt);
steps = round(tmax/dt);
dim = floor(steps/nplot);
[u; zeros(dim,N)];
tdata = 0;
j=1;
   for n = 1:steps
   t = n*dt; g = -1i*dt*k;
   E = exp(-dt*(k.^4-k.^2)/2); E2 = E.^2;
   a = g.*fft(real( ifft( v ) ).^2);
   b = g.*fft(real( ifft(E.*(v+a/2)) ).^2);  % Runge-Kutta
   c = g.*fft(real( ifft(E.*v + b/2) ).^2);   % 4. Ordnung
   d = g.*fft(real( ifft(E2.*v+E.*c) ).^2);
   v = E2.*v + (E2.*a + 2*E.*(b+c) + d)/6;
     if mod(n, nplot) == 0
      j=j+1;
      u = real(ifft(v));
      data(j,:) = u; tdata = [tdata t];
     end
   end

waterfall(x,tdata,data), colormap default
xlabel x, ylabel t, zlabel u, grid off
```

Burgersgleichung

Als weiteres Beispiel zur Verwendung des integrierenden Faktors und als ein Beispiel für eine Gleichung die Unstetigkeiten aufweisen kann, betrachten wir als nächstes die Burgersgleichung:

$$u_t + (u^2)_x = \epsilon u_{xx} \quad \text{auf} \quad [-\pi, \pi],$$

mit Anfangswerten

$$u(x,0) = \begin{cases} \sin^2(x) & \text{auf } [-\pi, 0] \\ 0 & \text{auf } [0, \pi]. \end{cases}$$

Der integrierende Faktor ist hier $e^{\epsilon k^2 t}$, die Wahl der Schrittweite $h = 1/100$ zeigt sich hier als ausreichend. Die Besonderheit an dieser Gleichung ist der Viskositätskoeffizient ϵ der für niedrige Werte nahe null Unstetigkeiten hervorrufen kann. Ein Koeffizient von $\epsilon = 0.25$ ist zunächst noch unproblematisch, *Abb.5.9* zeigt das Ergebnis mit $N = 256$.

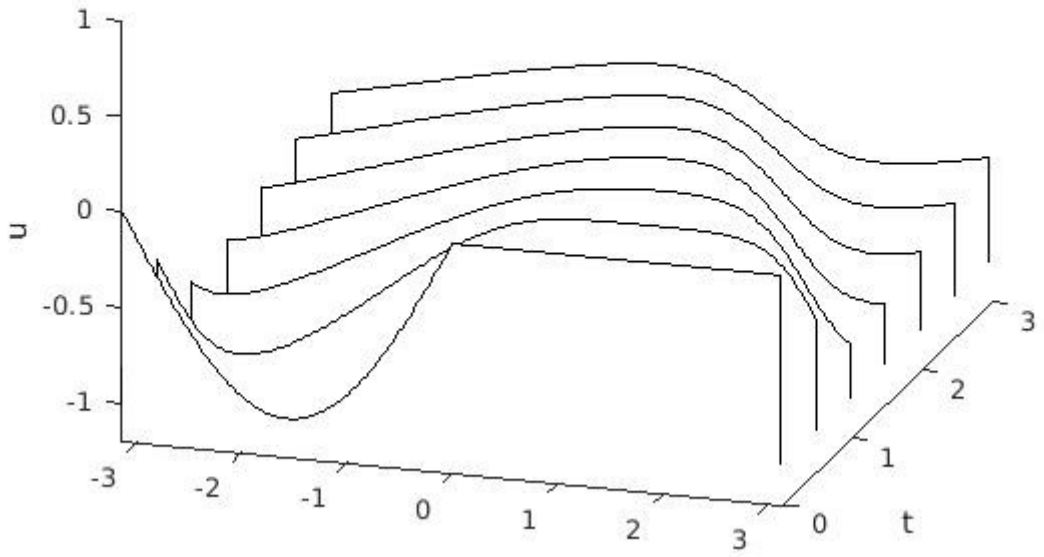

Abb.5.9: Bugersgleichung mit Viskositätskoeffizient $\epsilon = 0.25$

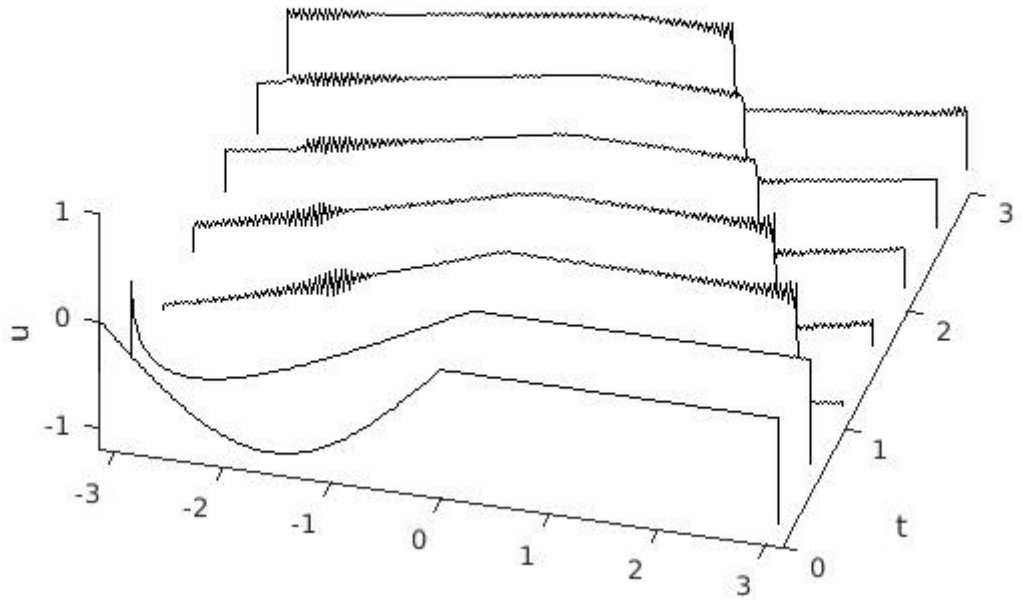

Abb.5.10: Burgersgleichung mit Viskositätskoeffizient $\epsilon = 0.001$

Mit immer kleiner werdendem Epsilon wird die Gleichung schwieriger zu simulieren und unabhängig davon wie klein die Schrittweite gewählt wird, treten aufgrund der Unstetigkeit Oszillationen auf. Dieser Effekt ist das bekannte Gibbs-Phänomen. Bei $\epsilon = 10^{-3}$ und $N = 256$ sieht das Ergebnis wie in *Abb.5.10* aus. Mit kleinerer Anzahl an Gitterpunkten treten die Oszillationen schon mit größerem Epsilon auf. Mit $N = 64$ schon bei $\epsilon = 0.02$ und mit $N = 128$ bei $\epsilon = 0.01$. Eine Verkleinerung der Schrittweite lässt die Oszillationen nicht verschwinden.

Programmcode Burgersgleichung

```
% Gitter und Anfangswerte
N = 256; dt = 1/100; x = (2*pi/N)*(-N/2:N/2-1);
u = [sin(x(1:N/2)) zeros(1,N/2)];
v = fft(u);
k = [0:N/2-1 0 -N/2+1:-1]; k2 = k.^2;
t = 0; tmax = 3;
ppsec = 2;
nplot = round((1/ppsec)/dt);
steps = round(tmax/dt);
data = u; tdata = 0;
eps = 0.25;
for n = 1:steps
t = n*dt; g = -1i*dt*k;
   E = exp(-dt*eps*k2/2); E2 = E.^2;
   a = g.*fft(real( ifft( v ) ).^2);
   b = g.*fft(real( ifft(E.*(v+a/2)) ).^2); % Runge-Kutta
   c = g.*fft(real( ifft(E.*v + b/2) ).^2); % 4. Ordnung
   d = g.*fft(real( ifft(E2.*v+E.*c) ).^2);
   v = E2.*v + (E2.*a + 2*E.*(b+c) + d)/6;
     if mod(n, nplot) == 0
      u = real(ifft(v));
      data = [data; u]; tdata = [tdata t];
     end
end

waterfall(x,tdata,data), view(10,70), colormap([0 0 0])
axis([-pi pi 0 tmax -1.2 1])
ylabel t, zlabel u, grid off
```

Reaktions-Diffusions-Gleichung

Die letzte hier aufgeführte Gleichung enthält einen linearen und einen nichtlinearen Teil:

$$u_t = u_{xx} + e^u, \qquad x \in [-1, 1], \quad t > 0,$$
$$u(\pm 1, t) = u(x, 0) = 0.$$

Zur räumlichen Diskretisierung wird eine Chebyshev Spektralmethode gewählt. Da wie bereits erwähnt die Chebyshev-Ableitungsmatrix zweiter Ordnung D_C^2 Eigenwerte von der Größenordnung $O(N^4)$ besitzt, müssten für ein explizites Zeitschrittverfahren eine sehr kleine Schrittweite gewählt werden. Um dies zu umgehen, wird für den linearen (steifen) Teil der Gleichung das A-stabile Crank-Nicolson-Verfahren eingesetzt und für den nichtlinearen Teil das Adams-Bashforth-

Verfahren zweiter Ordnung. Die Kombination dieser beiden ergeben ein implizit-explizites Verfahren der Form

$$u_{n+2} = u_{n+1} + \tfrac{3}{2}hf_{n+1} - \tfrac{1}{2}hf_n + \tfrac{1}{2}hg_{n+2} + \tfrac{1}{2}hg_n,$$

wobei $f = u_{xx}$ linear, $g = e^u$ nichtlinear [6].
Dadurch ist eine wesentlich größere Wahl der Zeitschrittweite möglich. Für den Startwert u_1 wird ein Euler-Schritt mit einer sehr kleinen Schrittweite durchgeführt.
Das Interessante an dieser Gleichung ist, dass die Lösung unter keinen Umständen konvergieren wird, sondern nach einer endlichen Zeit einen Wert nahe unendlich erreicht. Dieser Zeitpunkt des „blow-up“ ist in *Abb.5.11* veranschaulicht. Ab dem Zeitpunkt $t \approx 3,55$ ist die Lösung nicht mehr darstellbar, unabhängig von der Zeitschrittweite.

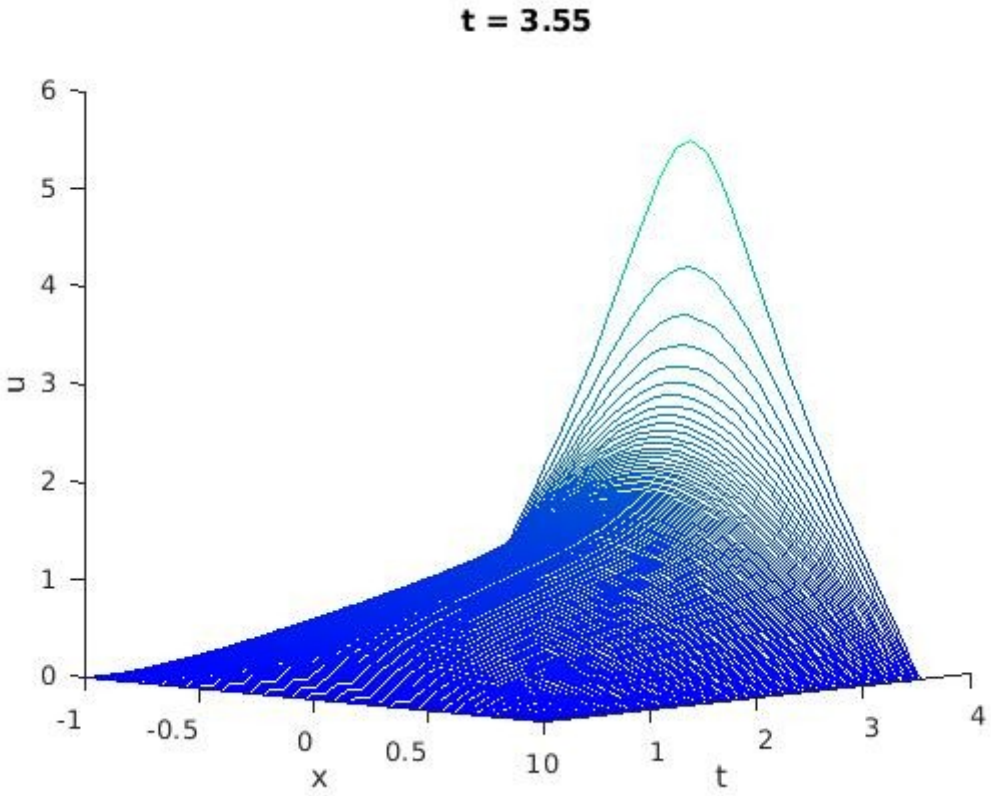

Abb. 5.11

Programmcode Reaktions-Diffusions-Gleichung

```
% Ableitungsmatrix 2.Ordnung und Gitter
N = 50; [D,x] = cheb(N); D2 = D^2; t=0;
D2 = D2(2:N,2:N);

u0=zeros(N-1,1);
I = eye(N-1,N-1);

% Ein Euler-Schritt für den Startwert
dt = 1/150000;
w0 = D2*u0; e0 = exp(u0);
u1 = u0 + dt*(w0 + e0);
w1 = D2*u1; e1 = exp(u1);

dt = 1/700;
tmax = 3.5; ppsec=50;
nplot = round((1/ppsec)/dt);
data = [[0; u0; 0]'; [0; u1; 0]']; tdata = [0 dt];
u2=zeros(1,N+1);
n=1;
while abs(u2(N/2)) < 1e+6
      t = t+dt; n=n+1;
      u2 = (I - 1/2*dt*D2)\(u1 + 3/2*dt*e1 - ...
               1/2*dt*e0  + 1/2*dt*w1);
      w0 = w1;    e0 = e1;
      u1 = u2; w1 = D2*u1;    e1 = exp(u1);
   if mod(n, nplot) == 0
      data = [data;[0; u2; 0]']; tdata = [tdata t];
   end

end

waterfall(x,tdata,data), colormap winter,
xlabel x,ylabel t, zlabel u, grid off
title(['t = ' num2str(t)])
```

Literaturverzeichnis

[1] **W.A. Adkins, M.G. Davidson.** Ordinary Differential Equations, Springer, New York Heidelberg (2012).

[2] **M. Basto, V. Semiao, F. Calheiros.** Dynamics in spectral solutions of Burgers equation, Journal of Computational and Applied Mathematics 205, 296 – 304 (2006).

[3] **A. Bayliss, A. Class, and B. J. Matkowsky.** Roundoff error in computing derivatives using the Chebyshev differentiation matrix, J. Comput. Phys. 116, 380-383 (1994).

[4] **John P. Boyd.** Chebyshev and Fourier Spectral Methods, second edition, Dover Publications Inc., Mineola (2000).

[5] **J. C. Butcher.** The Numerical Analysis of Ordinary Differential Equations: Runge-Kutta and General Linear Methods, Wiley, New York (1987).

[6] **J. Frank, W. Hundsdorfer, J.G. Verwer.** On the stability of implicit-explicit linear multistep methods, Applied Numerical Mathematics 25, 193-205, (1997).

[7] **D. Gottlieb and L. Lustman.** The spectrum of the Chebyshev collocation operator for the heat equation, SIAM J. Numer. Anal., 909-921 (1983).

[8] **G.W. Griffiths and W.E. Schiesser.** Linear and nonlinear waves, Scholarpedia, 4(7):4308 (2009).

[9] **E. Hairer, S. P. Nørsett and G. Wanner.** Solving Ordinary Differential Equations I. Nonstiff Problems. Springer-Verlag, Berlin, Heidelberg (1987).

[10] **E. Hairer, G. Wanner.** Solving Ordinary Differential Equations II. Stiff and Differential-Algebraic Problems. Second Revised Edition, Springer-Verlag, Berlin, Heidelberg (1996).

[11] **D.J. Higham and L.N. Trefethen.** Stiffness of ODEs, BIT Numerical Mathematics 33, 285-303 (1993).

[12] **A. Iserles**: A first course in the numerical analysis of differential equations, Cambridge University Press, 2. Auflage, New York (2009).

[13] **A.-K. Kassam and L.N. Trefethen.** Fourth-order ime-stepping for stiff PDEs, SIAMJ. Sci. Comput. Vol. 26, No. 4, 1214–1233 (2005).

[14] **T. Kato.** Perturbation Theory for Linear Operators, Corrected Printing of the Second Edition, Springer, Berlin Heidelberg (1995).

[15] **J. D. Lambert.** Numerical Methods for Ordinary Differential Systems: The Initial Value Problem, Wiley, New York (1991).

[16] **R.J. LeVeque.** Finite Difference Methods for Ordinary and Partial Differential Equations, Steady-State and Time-Dependent Problems, Society for Industrial and Applied Mathematics, Philadelphia, (2007).

[17] **R.J. LeVeque and L.N. Trefethen.** On the resolvent condition in the Kreiss matrix theorem, BIT 24, 584-591 (1984).

[18] **R. Rannacher.** Numerik 1, Numerik gewöhnlicher Differentialgleichungen, Heidelberg University Publishing, Heidelberg (2017).

[19] **S.C. Reddy, L.N. Trefethen.** Lax-stability of fully discrete spectral methods via stability regions and pseudo-eigenvalues. Comput. Meth. Appl. Mech. Eng. 80, 147-164 (1990).

[20] **S.C. Reddy, L.N. Trefethen.** Stability of the method of lines, Numer. Math. 62, 234-267

(1992).
[21] **J. Shen, T. Tang, L.-L. Wang.** Spectral Methods, Algorithms, Analysis and Applications, Springer Heidelberg New York (2011).
[22] **E. Süli and D. F. Mayers.** An Introduction to Numerical Analysis, Cambridge University Press, Cambridge (2003).
[23] **L.N. Trefethen.** Pseudospectra of linear operators. SIAM Review 39, no. 3, 383–406 (1997).
[24] **L.N. Trefethen.** Computation of Pseudospectra, Cambridge University Press, Acta Numerica, Volume 8, 247-295 (1999).
[25] **L. N. Trefethen.** Spectral Methods in MATLAB, Society for Industrial and Applied Mathematics, Philadelphia, (2000).
[26] **L.N. Trefethen.** Approximation Theory and Approximation Practice, Society for Industrial and Applied Mathematics, (2013).
[27] **L.N. Trefethen, M. Embree.** Spectra and Pseudospectra, The Behavior of NonnormaI Matrices and Operators, Princeton University Press, Princeton (2005).
[28] **L. N. Trefethen and M. R. Trummer.** An instability phenomenon in spectral methods , SIAM J. Numer. Anal. 24 , 1008-1023, (1987).
[29] **R.O. Weber, S.I. Barry.** Finite-Time Blow-Up in Reaction-Diffusion Equations, Mathematical and Computer Modelling, Vol.18, Issue 10, 163-168 (1993).
[30] **J. A. C. Weideman and L. N. Trefethen.** The eigenvalues of second-order spectral differentiation matrices, SIAM J. Numer. Anal. 25, 1279-1298, (1988).

Eidesstattliche Versicherung

Ich versichere an Eides statt durch meine Unterschrift, dass ich die vorstehende Arbeit selbständig und ohne fremde Hilfe angefertigt und alle Stellen, die ich wörtlich oder annähernd wörtlich aus Veröffentlichungen entnommen habe, als solche kenntlich gemacht habe, mich auch keiner anderen als der angegebenen Literatur oder sonstiger Hilfsmittel bedient habe. Die Arbeit hat in dieser oder ähnlicher Form noch keiner anderen Prüfungsbehörde vorgelegen.

_____ _____

Ort, Datum Unterschrift