Die Dichte-Matrix Renormierungsgruppe

Carsten Güttler

20.01.2005

Einleitung

Die DMRG ist ein numerisches Verfahren, mit dem man sehr gute Lösungen zur Approximation von niedrigen Energiezuständen in stark wechselwirkenden Quantensystemen erreichen kann. Das Verfahren wurde erst 1991 von Steven White [1] entwickelt und hat wegen der enormen Genauigkeit großes Aufsehen erregt. Man kann beispielsweise einen relativen Fehler von 10^{-10} für das Heisenberg Modell erreichen und die DMRG wurde seitdem in vielen Bereichen der Physik erprobt. Ich möchte dieses Verfahren hier anhand eines einfachen Problems, und zwar das eines Teilchens im eindimensionalen unendlichen Potentialwall, vorstellen. Es ist eine Verallgemeinerung auf wechselwirkende Probleme möglich, sie soll hier aber nicht behandelt werden.

Literatur

[1] Steven White, Phys. Rev. Letters 69. 2863 - 2866 (1992)

[2] Peschel, Wang, Kaulke, Hallberg (Eds.): "Density-Matrix Renormalization - A New Numerical Method in Physics", Kapitel 2 von Reinhard M. Noack und Steven R. White, Springer Verlag (1999)

[3] M A Martín-Delgado et al: "Density-Matrix Renormalization Group applied to single-particle Quantum Mechanics", J. Phys. A: Math. Gen. 32 6079 - 6090 (1999)

1 Das diskretisierte Grundproblem

Wir wissen eine Menge über das Teilchen im Kasten, deshalb ist es ein geeignetes Beispiel, um ein neues Verfahren zu erproben. Wir kennen beispielsweise die Wellenfunktion ψ, die wir sofort aufschreiben können:

$$\psi_i = \sin\left(\frac{i \cdot \pi}{L+1} \cdot k\right) \quad \begin{aligned} i &= 1, 2, ..., L \\ k &= 1, 2, ... \end{aligned} \tag{1}$$

Dieses ist schon eine diskretisierte (nicht normierte) Version der Wellenfunktion. Der Index k gibt die Anregung des Zustandes an und der Index i den Ort, der im Gegensatz zu der allgemein bekannten Form mit x nur ganzzahlige Werte annehmen kann. Wir betrachten die Wellenfunktion somit nur für diskrete Punkte auf einem eindimensionalen Gitter. Es ergeben sich für $k = 1$ und $k = 2$ die ersten beiden Grundzustände in Abbildung 1. Es ist noch zu bemerken, daß die Randwerte an denen der Sinus null wird in Formel (1) nicht enthalten sind. Das wären die Gitterplätze $i = 0$ und $i = L + 1$.

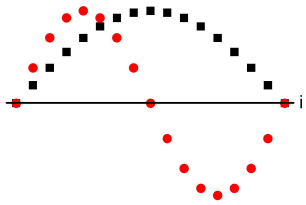

Abbildung 1: Die Wellenfunktion ψ_i laut Formel (1) für den Grundzustand und den ersten angeregten Zustand auf einem diskreten Gitter.

Zur Beschreibung des Problems benötigen wir noch den Hamiltonoperator, den wir uns zunächst in der diskretisierten Form erarbeiten müssen. Es geht im wesentlichen um die zweite Ableitung, die diskretisiert werden muß. Man erhält diese aus einer Taylorentwicklung um einen Punkt x, an dem die Wellenfunktion den Wert $\psi(x)$ annehmen soll. Man entwickelt um einen Gitterabstand h nach rechts und nach links

$$\begin{aligned} \psi(x+h) &= \psi(x) + \psi'(x)h + \frac{1}{2!}\,\psi''(x)h^2 + O(h^3) \\ \psi(x-h) &= \psi(x) - \psi'(x)h + \frac{1}{2!}\,\psi''(x)h^2 + O(h^3), \end{aligned}$$

vernachlässigt die dritte Ordnung und addiert die beiden Gleichungen, um sich der ersten Ableitung zu entledigen:

$$\psi(x+h) + \psi(x-h) = 2\,\psi(x) + \psi''(x)h^2$$

Diese Gleichung kann man nun auch formell diskretisieren, indem man zum Beispiel die Wellenfunktion am Punkt i mit ψ_i bezeichnet. Dann kann man die Gleichung nach ψ_i'' auflösen.

$$\begin{aligned} \psi_{i+1} + \psi_{i-1} &= 2\psi_i + h^2\psi_i'' \\ \Rightarrow \psi_i'' &= \frac{-2\psi_i + \psi_{i-1} + \psi_{i+1}}{h^2} \end{aligned} \tag{2}$$

Mit dieser Gleichung kann man schon arbeiten. Es gibt zum Beispiel die Möglichkeit, niedrig angeregte Energiezustände nach einem Monte-Carlo Verfahren zu bestimmen. Dafür setzt man sich auf einen beliebigen Gitterplatz und fragt ab, ob es eine energetische Verbesserung ergibt (wird die zweite Ableitung kleiner?), wenn man die Wellenfunktion anhebt. Wenn dieses der Fall ist, dann tut man es, wenn es nicht der Fall ist tut man es mit einer gewissen Wahrscheinlichkeit trotzdem. Danach setzt man sich auf einen anderen Gitterplatz und wiederholt die ganze Prozedur so lange, bis die gesamte Energie nicht mehr verringert werden kann. Das ist das typische Vorgehen bei der Monte-Carlo Methode. Es ist ein einfaches Anwendungsbeispiel, jedoch lassen die Lösungen noch zu wünschen übrig. Die Genauigkeit ist nicht besonders gut, man erreicht nur schwer angeregte Energieniveaus und im Grunde ist das Verfahren nur für das Grundproblem anwendbar. Es gibt Modifikationen und andere Möglichkeiten, jedoch soll es darum nicht gehen.

Wir wollen den Hamiltonoperator ausgehend von Formel (2) etwas abstrahieren. Wir setzen zunächst das h und sämtliche anderen Konstanten des Hamiltonoperators so, daß sie zusammen 1 ergeben und nicht weiter stören. Die zweite Ableitung steht im Hamiltonoperator mit negativem Vorzeichen und wir erhalten:

$$H_{i,j} = \begin{cases} 2 + V(x_i) & \text{für } i = j \\ -1 & \text{für } i = j \pm 1 \\ 0 & \text{sonst} \end{cases} \tag{3}$$

Das Potential $V(x_i)$ wird zunächst nicht behandelt ($V = 0$). Diese Darstellung deckt sich mit Formel (2), außerdem läßt sich der Hamiltonoperator dadurch auch als Matrix schreiben, wenn wir unsere Wellenfunktion aus Formel (1) ebenso als Vektor schreiben. In den Komponenten von ψ stehen dann die Funktionswerte auf den entsprechenden Gitterplätzen.

$$H = \begin{pmatrix} 2 & -1 & & 0 \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ 0 & & -1 & 2 \end{pmatrix} \qquad \psi_k = \begin{pmatrix} \sin\left(\frac{1\cdot\pi}{L+1}\cdot k\right) \\ \sin\left(\frac{2\cdot\pi}{L+1}\cdot k\right) \\ \vdots \\ \sin\left(\frac{L\cdot\pi}{L+1}\cdot k\right) \end{pmatrix} \tag{4}$$

Man mache sich klar, daß diese Darstellung sinnvoll ist. In der Eigenwertgleichung

$$H\psi_k = E_k\psi_k$$

stände beispielsweise in jeder Komponente die Anwendung von Formel (2) auf den entsprechenden Gitterplatz. Dies wollen wir nun für eine Komponente i tun und daraus die Energieeigenwerte analytisch berechnen:

$$\begin{aligned}
(H\psi)_i &= -\sin\left((i-1)\frac{\pi}{L+1}k\right) + 2\sin\left(i\frac{\pi}{L+1}k\right) - \sin\left((i+1)\frac{\pi}{L+1}k\right) \\
&= \sin(ia)\left\{2 - \frac{\sin(ia-a)}{\sin(ia)} - \frac{\sin(ia+a)}{\sin(ia)}\right\} \qquad \text{mit } \; a = \frac{\pi}{L+1}k \\
&= \psi_i\left\{2 - \frac{\sin(ia)\cos(a) - \cos(ia)\sin(a) + \sin(ia)\cos(a) + \cos(ia)\sin(a)}{\sin(ia)}\right\} \\
&= \psi_i\left\{2 - \frac{2\sin(ia)\cos(a)}{sin(ia)}\right\} \\
&= \psi_i\left\{2 - 2\cos\left(\frac{\pi}{L+1}k\right)\right\}
\end{aligned}$$

Also ergeben sich die Eigenwerte, die wir hier rein analytisch und exakt, jedoch für das diskretisierte Grundproblem berechnet haben zu:

$$E_k = 2\left(1 - \cos\left(\frac{\pi}{L+1}\cdot k\right)\right) \tag{5}$$

Die Zustandsenergie kann aus der Hamiltonmatrix in Gleichung (4) auch numerisch mit großer Genauigkeit bestimmt werden, indem man die Matrix diagonalisiert. Die aufsteigenden Eigenwerte sind dann die aufsteigenden Zustandsenergien, die zugehörigen Eigenvektoren sind die Funktionswerte der Wellenfunktion. Wir wollen jedoch ein Näherungsverfahren entwickeln, da die Hamiltonmatrix im Realfall deutlich komplizierter und auch sehr groß ausfallen kann.

2 Die Wilson Renormierungsgruppe

Die Wilson Renormierungsgruppe ist ein modernes Näherungsverfahren, das sehr gute Lösungen für das Kondo-Problem geliefert hat. Es wird in letzter Konsequenz nicht für unsere Zwecke geeignet sein, jedoch liefert es gute Ansätze und die Grundidee für die DMRG. Also wollen wir dieses Verfahren erarbeiten, sehen woran es scheitert und die geeigneten Verbesserungen einfügen. Wir halten uns dabei eng an die Veröffentlichung [2].

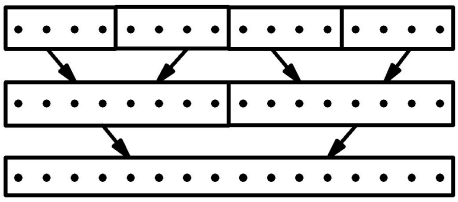

Abbildung 2: Zusammensetzen eines 16-Punkte-Gitters aus vier 4-Punkte-Gittern in zwei Schritten.

Um nicht das gesamte System in einem Schritt behandeln zu müssen wie bei einer exakten Diagonalisierung, nehmen wir zunächst das System auseinander und setzen es stückweise zusammen. Dies ist in Abbildung 2 verdeutlicht. In diesem Beispiel wird ein System aus sechzehn Gitterpunkten in vier Systeme mit jeweils vier Gitterpunkten aufgeteilt. In zwei Schritten werden dann jeweils zwei Systeme zusammengefügt, so daß am Ende das System aus sechzehn Gitterpunkten entsteht. Die Hamiltonmatrix für ein System aus vier Gitterpunkten ist die bekannte 4×4 Matrix

$$H_4 = \begin{pmatrix} 2 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{pmatrix}.$$

Diese Matrix wird diagonalisiert, wobei wir einen scheinbar komplizierten Weg gehen und ihn mit der Eigenvektormatrix (die Matrix, in deren Spalten die Eigenvektoren von H_4 stehen; Formel (7)) transformieren. Auf der Diagonalen stehen also schon die Energien, die dieses kleine System hätte, wenn es abgeschlossen wäre. Nun setzen wir aber gemäß

$$H_8 = \begin{pmatrix} \widetilde{H_4} & \widetilde{T_4} \\ \widetilde{T_4^\dagger} & \widetilde{H_4} \end{pmatrix} \tag{6}$$

zwei Blöcke zusammen. T_4 wäre dann die Matrix bei der nur im Eintrag links unten ein Wert steht. Das ist genau die -1, die beim Zusammensetzen zweier nicht transformierter Systeme fehlen würde. Die Systeme sind jedoch transformiert, was durch die Tilde gekennzeichnet sein soll. Also muß T_4 ebenfalls mit der Transformationsmatrix O_4 zu

$$\widetilde{T_4} = O_4^\dagger \begin{pmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ -1 & 0 & 0 & 0 \end{pmatrix} O_4$$

transformiert werden. $\widetilde{T_4}$ beschreibt die Wechselwirkung zwischen den beiden Teilsystemen.

Soweit ist das ganze nur eine komplizierte Umformung, die zunächst das gleiche Ergebnis liefert wie die exakte Diagonalisierung. Die Wilson'sche Grundidee ist nun, die Eigenvektormatrix zu reduzieren und nur die Eigenvektoren $\vec{v}_i$ der $m < L$ kleinsten Eigenwerte mitzunehmen:

$$O_L = \begin{pmatrix} | & & | \\ \vec{v_1} & \cdots & \vec{v_m} \\ | & & | \end{pmatrix} \tag{7}$$

Diese Idee ist durchaus nicht unbegründet, man denkt sich die beiden Teilsysteme nicht als absolut wechselwirkend, sondern betrachtet die Wechselwirkung störungstheoretisch. Die beiden Systeme sollen sich also nur gegenseitig stören. In der

Störungstheorie geht der Energieterm mit $1/E$ ein und somit ist es zunächst keine schlechte Idee, nur die kleinsten Energien mitzunehmen. Der Vorteil den man sich dadurch erarbeitet hat liegt auf der Hand: Man muß in jedem Schritt nur eine $2m \times 2m$ Matrix diagonalisieren. Das verbraucht weniger Speicher und geht auch schneller, da einzelne Unterblöcke so wie in unserem Beispiel die gleiche Hamiltonmatrix haben können. Dieses Verfahren hat sehr gute Lösungen für das Kondo-Problem geliefert, versagt aber leider für allgemeine Aufgaben wie zum Beispiel unser Grundproblem. Die Methode liefert beispielsweise für ein System, das zehnmal verdoppelt wird einen gigantischen relativen Fehler von 10^4. Ein Programmbeispiel ist in Anhang A dargestellt. Dort wird nur eine Blocktransformation (einmal zusammensetzen) durchgeführt, man sieht in dem Beispiel aber sehr gut die Abhängigkeit des schon recht großen Fehlers von dem Verhältnis m zu L.

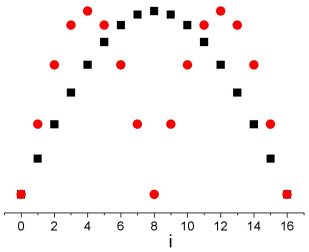

Abbildung 3: Die Wilson RG Methode liefert für kleine m (hier $m = L/2$) nicht die Grundzustadsenergie (schwarze Quadrate) sondern die Grundzustandsenergie der beiden einzelnen Systeme (rote Kreise). Es wird also ein Fehler beim Zusammensetzen der Blöcke gemacht.

Heute versteht man, woran es liegt, daß das Verfahren so ungenau wird. Man macht beim zusammensetzen von zwei Blöcken einen Fehler mit den Wechselwirkungstermen. Die Wechselwirkungterme $\widetilde{T_4}$ in Formel (6) wurden mit der beschnittenen Eigenvektormatrix transformiert. Für kleine m ist der Term so ungenau, daß man am Ende nicht die Grundenergie für das Gesamtsystem, sondern die Summe der Grundenergien für die einzelnen Blöcke berechnet hat. Dies wird in Abbildung 3 für sehr schwache Wechselwirkung (m sehr klein) verdeutlicht. Das System ist in der Mitte geteilt. Bei einem neuen Verfahren müssen wir also darauf achten, diese Wechselwirkung exakter zu behandeln. Genau das führt uns zur DMRG.

3 Die Dichte-Matrix Renormierungsgruppe

Bei der DMRG wird das ganze System in vier Blöcke statt in zwei aufgeteilt. Die mittleren beiden Blöcke beschreiben jeweils die Zustände für nur einen Gitterplatz, die äußeren Blöcke beschreiben alles, was links bzw. rechts davon ist. Eine Dar-

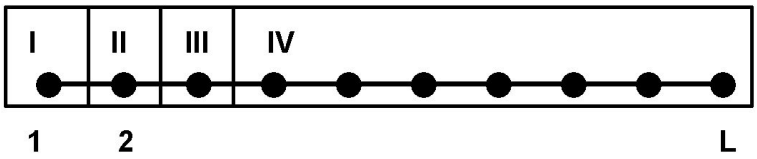

Abbildung 4: Bei der DMRG teilt man das gesamte System in vier Blöcke auf. Block II und III beschreiben jeweils nur einen Gitterplatz. Die Blöcke I bzw. IV beschreiben alles, was links bzw. rechts davon ist.

stellung findet sich in Abbildung 4. In diesem Fall beschreibt der linke Block nur einen Gitterplatz, was das Minimum ist, es dürften aber beliebig viele Punkte in einem äußeren Block vereint sein. Ein so definiertes System läßt sich durch die

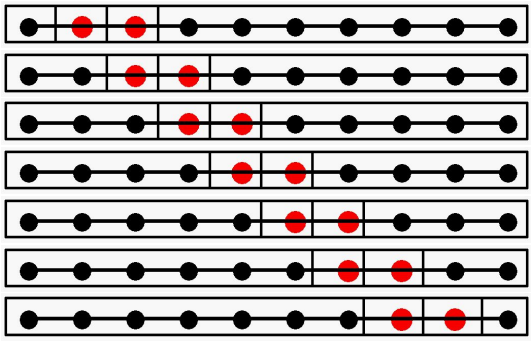

Abbildung 5: Der Algorithmus läuft hier von links nach rechts indem in jedem Schritt der erste mit dem zweiten Block verschmolzen wird und ein Gitterpunkt aus dem vierten Block herausgezogen wird.

Hamiltonmatrix

$$H_{DMRG} = \begin{pmatrix} H_{links} & T_{links} & 0 & 0 \\ T_{links}^{\dagger} & 2 & -1 & 0 \\ 0 & -1 & 2 & T_{rechts} \\ 0 & 0 & T_{rechts}^{\dagger} & H_{rechts} \end{pmatrix} \tag{8}$$

beschreiben. H_{li}, T_{li} bzw. H_{re}, T_{re} beschreiben also die Näherung für das linke bzw. rechte System mit den entsprechenden Wechselwirkungen. Die Aufgabe muß nun sein, für diese Werte bessere Näherungen zu finden als bei der Wilson RG. Dieses Problem ist lösbar und wird auch später behandelt, wir wollen es jedoch zunächst als gelöst betrachten und uns der Vorgehensweise des Algorithmus widmen.

Man kann beginnen wie in Abbildung 4, so daß der linke Block nur einen Zustand beschreibt. Nun wird so oft der erste Block mit dem zweiten verschmolzen (Abb. 5) und ein Punkt aus dem rechten Block herausgezogen, bis nur noch ein Punkt im rechten Block ist. Dabei wird in jedem Schritt die Genauigkeit von H_{li} und T_{li} verbessert, der Punkt mit dem er verschmolzen wird, ist ja gut bekannt: Er ist durch die linke 2 in der Matrix H_{DMRG} (Formel (8)) repräsentiert. Der rechte Block wird kleiner und somit unwichtiger; er wird verbessert, wenn wir - am Ende angekommen - wieder zurück von rechts nach links laufen. Die Genauigkeit von der Energie E_0 wird damit in jedem Schritt verbessert. Sie wird in jedem Schritt kleiner, jedoch niemals kleiner als die exakte Energie. Man kann zeigen, daß das Verfahren gegen die exakte Energie konvergiert.

Dies alles hätte mit der Wilson RG ähnlich funktioniert. Der wesentliche Unterschied ist nun, ein gutes H_{li}, T_{li}, H_{re} und T_{re} zu finden; Wilson RG hätte dabei versagt. Der Knackpunkt ist hier, daß man versucht, nicht die Energie sondern die Wellenfunktion zu optimieren. Das heißt, man möchte das Betragsquadrat

$$S = \left| |\psi\rangle - |\bar{\psi}\rangle \right|^2$$

der Differenz einer exakten Wellenfunktion $|\psi\rangle$ und einer approximierten Wellenfunktion $|\bar{\psi}\rangle$ minimieren. Die Wellenfunktionen müssen zunächst geeignet dargestellt werden:

$$|\psi\rangle = \sum_i \sum_j \psi_{ij} |i\rangle |j\rangle \qquad \text{und} \qquad |\bar{\psi}\rangle = \sum_{\alpha=1}^{m} \sum_j \psi_{\alpha j} |u_\alpha\rangle |j\rangle \tag{9}$$

Der Zustand $|i\rangle$ repräsentiert die linke Hälfte, also Block I und Block II, der Zustand $|j\rangle$ repräsentiert den Rest (s. Abb. 6). Die ψ_{ij} und $\psi_{\alpha j}$ repräsentieren die Koeffizi-

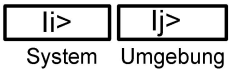

Abbildung 6: Die Zustände $|i\rangle$ bzw. $|j\rangle$ beschreiben das System (hier links) bzw. die Umgebung (hier rechts). Die Trennung wird zwischen den beiden mittleren (roten) Punkten in Abbildung 5 vollführt.

enten der Wellenfunktion. $|\bar{\psi}\rangle$ ist die genäherte Wellenfunktion mit dem genäherten Zustandsvektor $|u_\alpha\rangle$, in dem die Zustände von $|i\rangle$ optimal repräsentiert werden sollen, ohne alle Zustände mitzunehmen. Das m in der Summe ist also kleiner als die Anzahl der Gitterplätze im System (links).

Man kann nun zeigen, daß das Betragsquadrat S genau dann minimiert wird, wenn $|i\rangle$ und $|u_\alpha\rangle$ durch eine *Dichtematrixtransformation* miteinander verbunden sind. Der Beweis dafür wird hier nicht geführt.

Die Dichtematrix ist allgemein definiert als $\rho = \psi\psi^*$. Wir verwenden hier jedoch die *Reduzierte Dichtematrix*, in der die Zustände der Umgebung durch die Spur wegsummiert werden (Spur über j):

$$\rho_{red} = \mathrm{Spur}_j(\psi\psi^*) \qquad \text{oder analog} \qquad \rho_{red\ i'i} = \sum_j \psi_{i'j}\psi^*_{ij}$$

Da die Darstellung recht abstrakt ist, wollen wir die DMRG einmal an dem Grundproblem, das wir in Kapitel 1 erarbeitet haben, anwenden. Wir werden dort sehen, daß die Wellenfunktion und Dichtematrix sehr einfach sind woraus eine fast triviale Dichtematrixtransformation folgt.

4 Anwendung der DMRG am Grundproblem

4.1 Vorüberlegungen für das Grundproblem

Wir haben die Koeffizienten ψ_{ij} unserer Wellenfunktion in Formel (9) bereits mit zwei Indizes geschrieben. Das suggeriert, daß wir sie auch als Matrix schreiben können. Diese Matrix folgt zu:

$$\psi = \begin{pmatrix} 0 & \psi_{l+1} & \cdots & \psi_L \\ \psi_1 & & & \\ \vdots & & 0 & \\ \psi_l & & & \end{pmatrix}$$

Wir sehen, daß diese Matrix die Anforderungen erfüllt, die wir an die Koeffizienten gestellt haben. Nimmt der Index i einen bestimmten Wert an, so bewegen wir uns in der ersten Spalte, in der die Werte der Wellenfunktion im Systemblock stehen. Der Index j muß dann zwangsweise null sein, da wir ansonsten zwei Teilchen beschreiben würden. Wir haben ein Teilchen, also kann es sich nur entweder links oder rechts aufhalten. Die Null links oben beschreibt, den Fall, daß kein Teilchen im Kasten ist, die anderen Nullen beschreiben Mehrteilchenprobleme. Eine vereinfachte Darstellung, die jedoch analog ist, lautet:

$$\psi = \begin{pmatrix} 0 & \langle\psi_{re}| \\ |\psi_{li}\rangle & 0 \end{pmatrix}$$

Bilden wir aus dieser Matrix die Reduzierte Dichtematrix

$$\begin{aligned} \rho_{red} = \psi\psi^\dagger &= \begin{pmatrix} 0 & \langle\psi_{re}| \\ |\psi_{li}\rangle & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & \langle\psi_{li}| \\ |\psi_{re}\rangle & 0 \end{pmatrix} \\ &= \begin{pmatrix} \langle\psi_{re}|\psi_{re}\rangle & 0 \\ 0 & |\psi_{li}\rangle\langle\psi_{li}| \end{pmatrix} \end{aligned} \tag{10}$$

mit dem Eigenvektor $(1, 0, \cdots, 0)^T$ zum Eigenwert $w_r = \langle\psi_{re}|\psi_{re}\rangle$, der uns für den linken Block nicht interessiert. Der nächste Eigenvektor $(0, |\psi_{li}\rangle)^T$ zum Eigenwert $w_l = \langle\psi_{li}|\psi_{li}\rangle$ ist jedoch genau der Vektor, der uns interessiert, Eigenvektoren zu kleineren Eigenwerten nehmen wir nicht mit. Unsere Dichtematrixtransformation ist also für unser Grundproblem mit $|\psi_{li}\rangle$ sehr einfach und muß nur noch normiert werden.

Wir wollen nun beispielhaft einen Programmschritt durchführen, in dem wir den linken Block verbessern und den rechten verkleinern. Daran soll die Dichtematrixtransformation veranschaulicht werden. Wir bilden die Matrix H_{DMRG} gemäß Formel (8) und diagonalisieren diese. Wir setzen $m = 1$ und benötigen somit nur den Eigenvektor

$$|\psi\rangle = \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{pmatrix} = \begin{pmatrix} |\psi_{li}\rangle \\ |\psi_{re}\rangle \end{pmatrix}$$

zum kleinsten Eigenwert. Nach der Wilson Methode könnten wir die Matrix schon mit diesem Vektor transformieren, erhielten dann aber nur einen Zustand des Gesamtsystems, anstatt eines approximierten Zustands für das linke Teilsystem. Schreiben wir also für diesen speziellen Fall einmal unsere reduzierte Dichtematrix nach Formel (10) auf:

$$\rho_{red} = \begin{pmatrix} a_3^2 + a_4^2 & 0 & \\ 0 & a_1^2 & a_1 a_2 \\ 0 & a_1 a_2 & a_2^2 \end{pmatrix}$$

Der relevante Eigenvektor ergibt sich zu $(a_1, a_2)^T$. Unsere Transformationsvorschrift für die Matrix H_{DMRG} vereint nun Block I und Block II und lautet in normierter Form:

$$\begin{pmatrix} a_1' \\ a_2' \end{pmatrix} = \frac{1}{\sqrt{a_1^2 + a_2^2}} \begin{pmatrix} a_1 \\ a_2 \end{pmatrix} \tag{11}$$

Wenn wir sie anwenden ergibt sich

$$\begin{aligned} H(l+1) &= (a_1', a_2') \cdot \begin{pmatrix} H(l) & T(l) \\ T(l) & 2 \end{pmatrix} \cdot \begin{pmatrix} a_1' \\ a_2' \end{pmatrix} \\ &= H(l) \cdot a_1'^2 + 2 \cdot T(l) \cdot a_1' \cdot a_2' + 2 \cdot a_2'^2 \end{aligned} \tag{12}$$

und $T(l+1) = -a_2'$. Das $H(l+1)$ beschreibt unseren linken Block für den nächsten Zustand, das $T(l+1)$ die entsprechende Wechselwirkung. Da wir zwei Blöcke miteinander verschmolzen haben, müssen wir aus dem rechten approximierten Zustand noch einen reinen Zustand (Block III bzw. roter Punkt) herausholen. Dies macht man jedoch einfach, indem man den Wert aus dem Speicher abfragt, den man beim letzten rechts-links Durchlauf für die entsprechende Größe $(L - (l+1) - 2)$ gespeichert hat. Man möchte den rechten Block in diesem Schritt ja nicht verbessern, sondern nur die Umgebung realistisch einbeziehen.

4.2 Der Ablauf des Programms

Mit den beiden Formeln für $H(l+1)$ und $T(l+1)$ kann man schon ein Programm schreiben, das die Grundzustandsenergie für das Grundproblem approximiert. Das Programm (Quellcode s. Anhang B) besteht im wesentlichen aus zwei Schleifen: Die Schleife, die von links nach rechts läuft und die, die wieder zurück läuft. Beide Schleifen funktionieren sehr ähnlich: Es wird zunächst eine geeignete Matrix H_{DMRG} definiert, die Einträge H_{li}, T_{li} bzw. H_{re}, T_{re} können ganz am Anfang quasi beliebig gewählt werden, sie werden später verbessert. Die Matrix wird diagonalisiert und man erhält das a_1' und a_2' gemäß Formel (11) um die Formel für

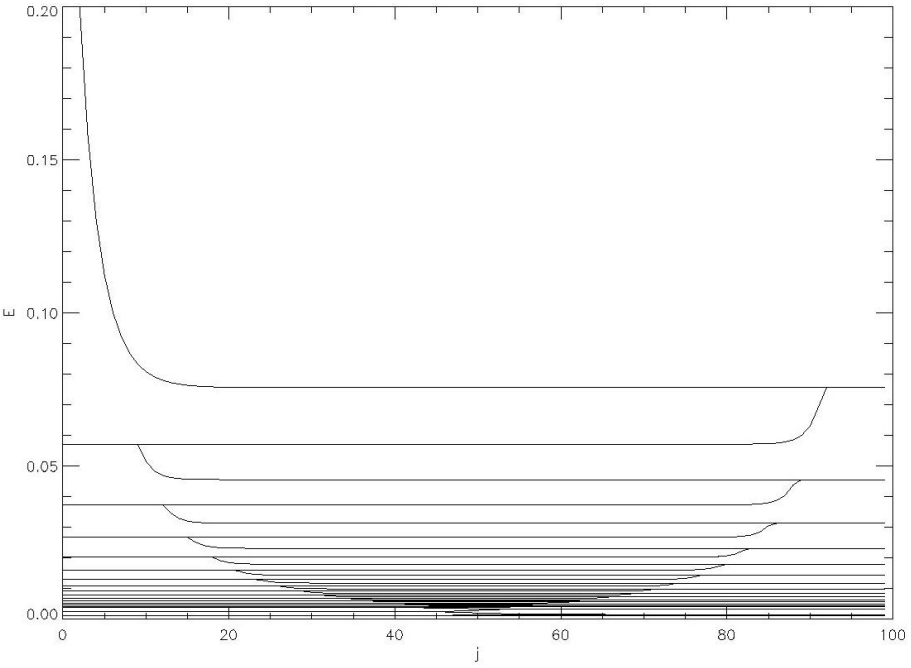

Abbildung 7: Die Energie wird in jedem links-rechts bzw. rechts-links Durchlauf verbessert. Sie konvergiert in 19 links-rechts-links Durchläufen für ein $L = 100$ Gitter gegen den exakten Wert $E_0 = 0.000967435$ ($V = 0$).

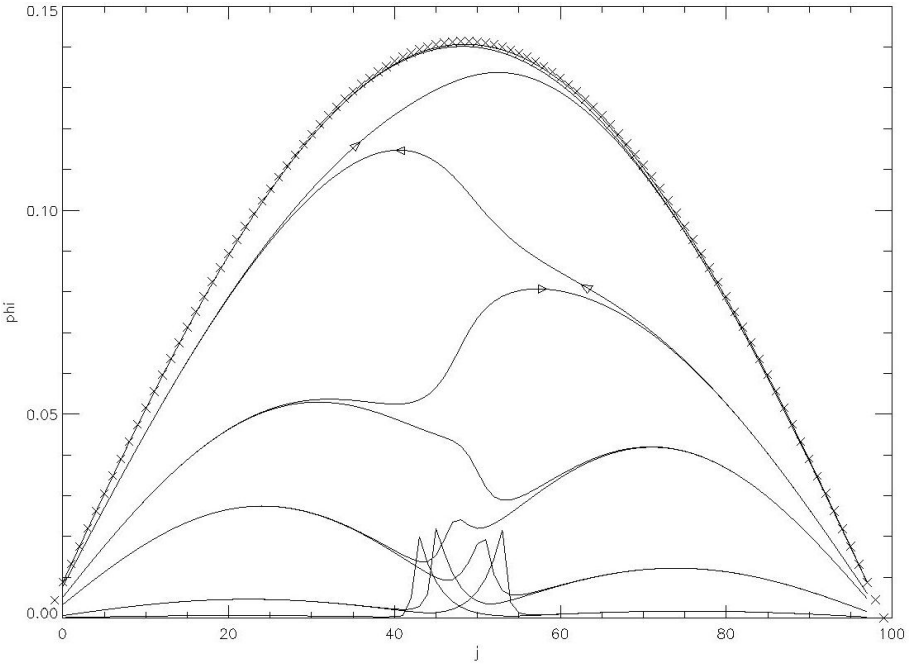

Abbildung 8: Die Wellenfunktion für das $L = 100$ Gitter konvergiert gegen die exakte Wellenfunktion (Kreuze). Aufgetragen werden kann die normierte Komponente a_2 oder a_3 des Eigenvektors zum kleinsten Eigenwert.

$H(l+1)$ und $T(l+1)$ anwenden zu können. Dies wird getan und die beiden Werte werden abgespeichert um die Schleife mit diesen Werten neu zu beginnen. Wenn die erste Schleife $L-2$ Mal durchgelaufen ist, also die Blöcke II und III (rote Punkte) bildlich am rechten Ende angekommen sind (letztes Bild in Abb. 5), so läuft man danach wieder zurück nach links. Nun werden die H_{re} und T_{re} verbessert und die H_{li} und T_{li} werden aus dem Speicher gelesen. Die Matrix ist also nun für jeden Schritt vom Algorithmus vorgegeben und wird in jedem Schritt verbessert. Man läßt die Schleifen so lange hin und her laufen bis das Verfahren die ausreichende Genauigkeit erzielt hat. Dabei muß man niemals eine größere Matrix als eine 4×4 Matrix diagonalisieren wodurch man Speicher und Rechenzeit spart. Die Erklärung eines Programms kann in diesem Umfang nur unzureichend sein, deshalb ist der Quellcode in Anhang B abgedruckt.

4.3 Ergebnisse des Programms

Das Programm läuft jedenfalls recht gut, die Energie konvergiert gegen den exakten Wert $E_0 = 0.000967435$ (s. Abb. 7). Die Genauigkeit ist nach der exakten Methode (Formel (5)) mit dem Taschenrechner in jeder Stelle gleich, außerdem kann man eine 100×100 Matrix noch exakt diagonalisieren und der Wert ist ebenfalls gleich. Die erhoffte Genauigkeit kann also tatsächlich erreicht werden. Aufgetragen ist in Abbildung 7 der kleinste Eigenwert der Matrix H_{DMRG} in jedem Schritt, also über der Größe des linken Blocks.

Analog kann man statt der Energie auch die Wellenfunktion plotten (Abb. 8). Dies ist sicherlich anschaulicher als ein abstrakter Zahlenwert einer Energie. Die Wellenfunktion ist die normierte Komponente a_2 oder a_3 des Eigenvektors zum kleinsten Eigenwert über dem zugehörigen Gitterplatz. Diese beiden Komponenten beschreiben ja gegenüber a_1 und a_4 nur einen Gitterplatz. Auch die Wellenfunktion konvergiert gegen die exakte Wellenfunktion (Formel (1)), die hier in normierter Form durch die Kreuze dargestellt ist.

5 Weitere Anwendungen der DMRG

Die Lösung dieses Grundproblems würde die Aufmerksamkeit, die wir der DMRG nun gewidmet haben nicht rechtfertigen. Tatsächlich kann man aber viele Aufgaben mit der DMRG lösen. Wir wollen uns hier noch kurz den zwei einfachsten Verallgemeinerungen widmen, die mit dieser numerischen Genauigkeit schon ein lohnenswertes Ziel der DMRG sind.

5.1 Das Grundproblem mit Potential

Wir wollten uns dem Grundproblem mit Potential widmen, wie es sich in ausführlicherer Darstellung auch in [3] findet. Wir haben ab Formel (3) zur Vereinfachung das Potential $V = 0$ gesetzt. Dies muß man nicht tun. Wenn man das Potential konsequent mit einbezieht ergibt sich eine veränderte Matrix

$$H_{DMRG} = \begin{pmatrix} H_{links} & T_{links} & 0 & 0 \\ T^{\dagger}_{links} & 2+V_l & -1 & 0 \\ 0 & -1 & 2+V_{l+1} & T_{rechts} \\ 0 & 0 & T^{\dagger}_{rechts} & H_{rechts} \end{pmatrix}.$$

Ebenso muß man das Potential in Formel (12) mitnehmen indem aus der 2 ebenso ein $2 + V_l$ wird. Diese zwei kleinen Programmänderungen reichen, um die Grundzustandsenergie und die entsprechende Wellenfunktion für einen Kasten mit einem zusätzlichen Potential zu erhalten. In den Quellcode wurde also zum Beispiel ein

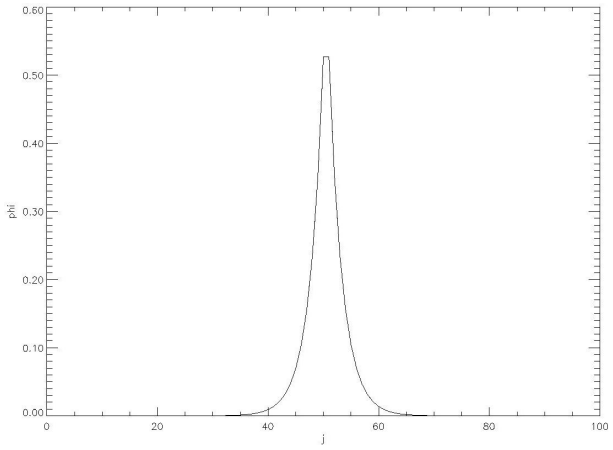

Abbildung 9: Die Wellenfunktion für ein $L = 100$ Gitter mit einem attraktiven δ-Potential an zwei Punkten: $V_{50} = -0.5$ und $V_{51} = -0.5$

attraktives δ-Potential an zwei Punkten ($V_{50} = -0.5$ und $V_{51} = -0.5$) eingefügt. Es ergibt sich eine erwartete abfallende Wellenfunktion. Die Aufenthaltswahrscheinlichkeit an den Punkten $j = 50$ und $j = 51$ ist deutlich erhöht und fällt zum Rand exponential ab, wo sie per Definition über die Randbedingungen null wird. Die Energie kann man für ein allgemeines Modell in den meisten Fällen nicht mehr analytisch behandeln, wenn dann mit sehr großem Aufwand. Für dieses Problem kann man jedoch noch die 100×100 Matrix numerisch diagonalisieren, und die Grundzustandsenergie stimmt wieder mit dem kleinsten Eigenwert überein.

5.2 Berechnung höher angeregter Energieniveaus

Selbstverständlich kann man mit dem Verfahren nicht nur die Grundzustandsenergie sondern auch angeregte Energien berechnen. Diese Verallgemeinerung ist nicht schwierig und betrifft nur Formel (12), da die Matrix nicht mehr nur mit der Grundzustandswellenfunktion transformiert wird.

$$\begin{aligned} \underline{\underline{H}}(l+1) &= \underline{\underline{A}}'^{\dagger} \cdot \begin{pmatrix} \underline{\underline{H}}(l) & \vec{T}(l) \\ \vec{T}^T(l) & 2 \end{pmatrix} \cdot \underline{\underline{A}}' \\ T_i(l+1) &= a'_{2,i} \;\; (i = 1, ..., k) \end{aligned}$$

Man nimmt also aus der Diagonalisierung die k kleinsten Eigenwerte und zugehörigen Eigenvektoren mit und erhält aus diesen die Transformationsvorschrift :

$$\underline{\underline{A}}' = \begin{pmatrix} \vec{a}'_{1,1} & \cdots & \vec{a}'_{1,k} \\ a'_{2,1} & \cdots & a'_{2,k} \end{pmatrix}$$

Eine tiefer gehende Behandlung für dieses Problem findet sich in [3]. Weitere Verallgemeinerungen sind generell denkbar, an dieser Stelle wollen wir es jedoch beim Einteilchen-Modell der Quantenmechanik belassen.

A Quellcode für Wilson RG

Dieses Programm setzt mit der Wilson Methode zwei Blöcke zusammen und berechnet die genäherte Grundzustandsenergie. Das Programm ist in IDL 5.2 geschrieben.

```
pro rg
```

```
L   = 10
m   = 10
H   = make_array(L,  L,  /DOUBLE, VALUE =0)
O   = make_array(L, L, /DOUBLE, VALUE = 0)
T   = make_array(L,  L,  /DOUBLE, VALUE = 0)
H_2 = make_array(2*L,2*L,/DOUBLE, VALUE = 0)
Ti  = make_array(L, L, /DOUBLE, VALUE = 0)

T[0,L-1]=-1

for i = 0, L-1 do $
    for j = 0, L-1 do $
        if i eq j then H[i,j]=2 else $
            if abs(i-j) eq 1 then H[i,j]=-1

O=H TRIRED, O, Eigenwerte, E    ;H in tridiagonale Form
bringen TRIQL, Eigenwerte, E, O ;EW und EV-Matrix von H aus tridiagonaler
                                ;Form berechnen
O=swappen(O,Eigenwerte,L)   ;EW und EV-Matrix nach Größe der EW ordnen

for i=m, L-1 do O[*,i]=0        ;"unwichtige" Einträge der EV-Matrix löschen

H = O##H##transpose(O) T = O##T##transpose(O)

for i = 0, m-1 do $
    for j = 0, m-1 do begin
        H_2[i,j]    =H[i,j]
        H_2[i+m,j+m]=H[i,j]
        H_2[i+m,j]  =T[i,j]
        H_2[i,j+m]  =T[j,i]     ;T[j,i] ist die transponierte zu T[i,j]
    endfor

TRIRED, H_2, Eigenwerte, E      ;H_2 in tridiagonale Form bringen
TRIQL, Eigenwerte, E, H_2       ;Eigenwerte aus der tridiagonalen Matrix
                                ;berechnen O=swappen(H_2,Eigenwerte,2*L)

print, Eigenwerte[2*L-2*m] ;
----------*-EXAKT-*----------
for i = 0, 2*L-1 do $
    for j = 0, 2*L-1 do $
        if i eq j then H_2[i,j]=2 else $
            if abs(i-j) eq 1 then H_2[i,j]=-1 else H_2[i,j]=0

TRIRED, H_2, Eigenwerte, E TRIQL, Eigenwerte, E, H_2
H_2=swappen(H_2,Eigenwerte,2*m) O=swappen(H_2,Eigenwerte,2*L)

print, ""
print, Eigenwerte[0]

end

function swappen, O, Eigenwerte, L
;*********************************************************************
;Diese Funktion ordnet die EW und EV aufsteigend nach der Größe der EW
;*********************************************************************
repeat begin
    noswap=1
    for i=0, L-2 do $
```

```
    if Eigenwerte[i] gt Eigenwerte[i+1] then begin
        noswap=0
        temp=Eigenwerte[i]
        Eigenwerte[i]=Eigenwerte[i+1]
        Eigenwerte[i+1]=temp
        temp_vec=O[*,i]
        O[*,i]=O[*,i+1]
        O[*,i+1]=temp_vec
        if i eq L-2 then i=0
    endif
endrep until noswap
return, O
end
```

B Quellcode für DMRG

Dieses Programm implementiert den dargestellten DMRG Algorithmus. Es ist ebenfalls in IDL 5.2 geschrieben. Die benötigte Prozedur "swappen" findet sich in dem Programm zur Berechnung nach Wilson RG in Anhang A.

```
pro dmrg2
L           = 100
L           = L-3
H           = make_array(4,  4,/DOUBLE, VALUE = 0)
O           = make_array(4,  4,  /DOUBLE,VALUE = 0)
HLHR        = make_array(L,  4,  /DOUBLE, VALUE = 2)
V           = make_array(L+1,    /DOUBLE, VALUE = 2)
H_ex        = make_array(L+3,L+3,/DOUBLE, VALUE = 0)
Fehler      = 1e-11
ausgabe     = 1 ;0:E plotten,   1:phi plotten
                ;2:E schreiben, 3:phi schreiben
Durchlaeufe = 0
temp2       = 0

HLHR[*,1]=1
HLHR[*,3]=1

for i=0,2 do begin
    H[i  ,i  ]=2
    H[i+1,i  ]=-1
    H[i  ,i+1]=-1
endfor H[3,3]=2

;V[50]=1.5
;V[49]=1.5

if ausgabe gt 1 then begin
    set_plot, 'ps'
    device, /landscape, filename='c:\plot0.ps', /color
endif

x=findgen(L-1)/L/1000 & y=findgen(L-1)/L/1000 case 1 of
    (ausgabe mod 2 eq 0): $
        plot, x,y, xrange=[0,L-11], yrange=[0,0.2],$
            title='Energieverlauf für ein L=100 Gitter',$
            xtitle='j', ytitle='E'
    (ausgabe mod 2 eq 1): begin
        plot, x,y, xrange=[0,L+3], yrange=[0,0.15],$
```

```
            title='Wellenfunktion für ein L=100 Gitter',$
            xtitle='j', ytitle='phi'
        for j=1, L+4 do $
            plots, j-1, sin(!pi*j/(L+4))/sqrt(50), psym=7, symsize=1
    end
endcase

repeat begin
    Durchlaeufe=Durchlaeufe+1
    for j=0, L-2 do begin

        H[0,0]= HLHR[j,0]
        H[1,1]= V[j]
        H[2,2]= V[j+1]
        H[3,3]= HLHR[j,2]
        H[1,0]=-HLHR[j,1]
        H[0,1]=-HLHR[j,1]
        H[2,3]=-HLHR[j,3]
        H[3,2]=-HLHR[j,3]

        O=H                         ;O ist zunächst temporäre Matrix,
                                    ;später EV-Matrix
        TRIRED, O, Eigenwerte, E    ;H in tridiagonale Form bringen
        TRIQL, Eigenwerte, E, O     ;EW und EV-Matrix von H aus
                                    ;tridiagonaler Form berechnen
        O=swappen(O,Eigenwerte,4)   ;EW und EV-Matrix nach Größe der EW ordnen

        a_1 = O[0,0]/sqrt(O[0,0]*O[0,0]+O[1,0]*O[1,0])  ;Erste, normierte Komp.
                                                        ;des kleinsten EV
        a_2 = O[1,0]/sqrt(O[0,0]*O[0,0]+O[1,0]*O[1,0])  ;Zweite, normierte Komp.
                                                        ;des kleinsten EV

        HLHR[j+1,1]=a_2
        HLHR[j+1,0]=a_1*a_1*HLHR[j,0] + V[j]*a_2*a_2 - 2*a_1*a_2*HLHR[j,1]

        case 1 of
            (ausgabe mod 2 eq 0): begin
                x[j]=j & y[j]=Eigenwerte[0]
            end
            (ausgabe mod 2 eq 1): begin
                x[j]=j+1 & y[j]=abs(O[1,0])/sqrt(O[0,0]*O[0,0]+O[1,0]*O[1,0]$
                     +O[2,0]*O[2,0]+O[3,0]*O[3,0])
            end
        endcase

    endfor

    if temp2-Eigenwerte[0] lt 0 and durchlaeufe gt 1 then print, $
        "Warnung im ",durchlaeufe, ". Durchlauf (links)!"
    if durchlaeufe gt 17 or ausgabe mod 2 eq 0 then oplot, x, y
    temp=Eigenwerte[0]

    for j=2, L do begin

        H[0,0]= HLHR[L-j+1,2]
        H[1,1]= V[L-j+2]
        H[2,2]= V[L-j+1]
        H[3,3]= HLHR[L-j+1,0]
```

```
        H[2,3]=-HLHR[L-j+1,1]
        H[3,2]=-HLHR[L-j+1,1]
        H[1,0]=-HLHR[L-j+1,3]
        H[0,1]=-HLHR[L-j+1,3]

        O=H                         ;O ist zunächst temporäre Matrix,
                                    ;später EV-Matrix
        TRIRED, O, Eigenwerte, E    ;H in tridiagonale Form bringen
        TRIQL, Eigenwerte, E, O     ;EW und EV-Matrix von H aus
                                    ;tridiagonaler Form berechnen
        O=swappen(O,Eigenwerte,4)   ;EW und EV-Matrix nach Größe der EW ordnen

        a_1 = O[0,0]/sqrt(O[0,0]*O[0,0]+O[1,0]*O[1,0])  ;Erste, normierte Komp.
                                                        ;des kleinsten EV
        a_2 = O[1,0]/sqrt(O[0,0]*O[0,0]+O[1,0]*O[1,0])  ;Zweite, normierte Komp.
                                                        ;des kleinsten EV

        HLHR[L-j,3]=a_2
        HLHR[L-j,2]=a_1*a_1*HLHR[L-j+1,2] + V[j]*a_2*a_2 - 2*a_1*a_2*HLHR[L-j+1,3]

        case 1 of
            (ausgabe mod 2 eq 0): begin
                x[j-2]=L-j+4 & y[j-2]=Eigenwerte[0]
            end
            (ausgabe mod 2 eq 1): begin
                x[L-j]=L-j+3
                y[L-j]=abs(O[1,0])/sqrt(O[0,0]*O[0,0]+O[1,0]*O[1,0]$
                       +O[2,0]*O[2,0]+O[3,0]*O[3,0])
            end
        endcase
    endfor
    if durchlaeufe gt 17 or ausgabe mod 2 eq 0 then oplot, x, y

    if temp-Eigenwerte[0] lt 0 then print, "Warnung im ",durchlaeufe,$
            ". Durchlauf (rechts)!"
    temp2 = Eigenwerte[0]
    if durchlaeufe mod 100 eq 0 then print, durchlaeufe, ". Durchlauf beendet!"
endrep until abs(temp-Eigenwerte[0]) lt Fehler or durchlaeufe gt 199

if ausgabe gt 1 then begin
    device, /close
    set_plot, 'win'
endif

Print, "*****************************************************"
Print, "L=", L, ",", Durchlaeufe, " Durchläufe"
if abs(Eigenwerte[0]-temp) gt 1e-11 then Print, "E_o   = "$
     ,Eigenwerte[0], " +-", Eigenwerte[0]-temp $
else Print, "E_o   = ",Eigenwerte[0], " +- 1e-11"
H_ex[0,0]=2 & H_ex[1,0]=-1 & H_ex[0,1]=-1
for i=1,L+1 do begin
    H_ex[i  ,i  ]=V[i-1]
    H_ex[i+1,i  ]=-1
    H_ex[i  ,i+1]=-1
endfor
H_ex[L+2,L+2]=2
TRIRED, H_ex, Eigenwerte, E
TRIQL,Eigenwerte, E, H_ex
```

```
O=swappen(H_ex,Eigenwerte,L+3)
Print,"E_o,ex=", Eigenwerte[0]
Print,"*************************************************"

end
```