D-MATH Numerische Methoden FS 2017
Dr. Vasile Gradinaru
Luc Grosheintz

Serie 12

Abgabedatum: keine Abgabe, ML wird hochgeladen.
Koordinatoren: Luc Grosheintz, HG G 46, `luc.grosheintz@sam.math.ethz.ch`
Webpage: `http://metaphor.ethz.ch/x/2017/fs/401-1662-10L`

1. *Zeitabhängige Schrödinger-Gleichung*

Wir betrachten die zeitabhängige Schrödinger-Gleichung mit Potential V:

$$\begin{cases} i\varepsilon \frac{\partial u}{\partial t} & = -\frac{1}{2}\varepsilon^2 \frac{\partial^2 u}{\partial x^2} + V(x)u, \\ u(x,0) & = g(x), \qquad x \in [-1,1] \end{cases} \tag{1}$$

wobei $\varepsilon = 0.01$ die Rolle von $\hbar$ übernimmt. Sei $N = 2^l$. Wir wählen eine Lösung:

$$u_N(x,t) = \sum_{k=-\frac{N}{2}}^{\frac{N}{2}-1} c_k(t) e^{2\pi i k x}, \qquad x \in [-1,1],$$

so dass (1) in $\underline{x} = [x_j] = [\frac{2j}{N}]$ mit $j = -\frac{N}{2}, \dots, \frac{N}{2}-1$ erfüllt ist. Sei $\underline{c} = [c_0, \dots, c_{\frac{N}{2}-1}, c_{-\frac{N}{2}}, \dots, c_{-1}]$ der Vektor mit den Fourier-Koeffizienten. Da $\underline{c} = \mathcal{F}_N[u_N(\underline{x})](\underline{k})$ erhalten wir:

$$i\dot{c}_k = \frac{1}{2}\varepsilon k^2 c_k + \left(\mathcal{F}_N V_N \mathcal{F}_N^{-1} \underline{c}\right)_k$$

mit:

$$V_N = \frac{1}{\varepsilon}\operatorname{diag}(V(\underline{x})),$$

oder:

$$i\underline{\dot{c}} = \frac{1}{2}\varepsilon D_N^2 \underline{c} + \mathcal{F}_N V_N \mathcal{F}_N^{-1} \underline{c},$$

mit:

$$D_N = \operatorname{diag}\left(0, \dots, \frac{N}{2}-1, -\frac{N}{2}, \dots, -1\right).$$

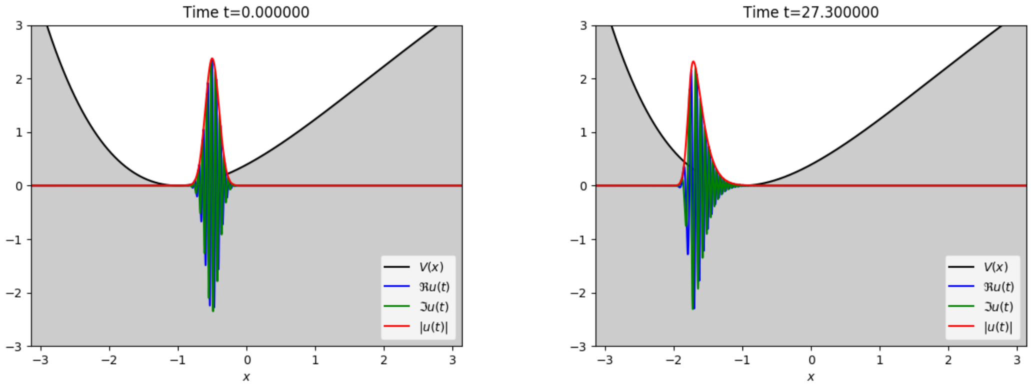
Deshalb gilt nun:

$$\boxed{\underline{\dot{c}} = -\frac{1}{2} i\varepsilon D_N^2 \underline{c} - i\mathcal{F}_N V_N \mathcal{F}_N^{-1} \underline{c}} \tag{2}$$

und wir haben:

$$e^{-ih\frac{\varepsilon}{2}D_N^2}\underline{c} = \left[e^{-\frac{1}{2}i\varepsilon h k^2} c_k\right], \quad \text{mit} \quad k = 0, \dots, \frac{N}{2}-1, -\frac{N}{2}, \dots, -1 \tag{3}$$

$$e^{-ihV_N} u_N(\underline{x}) = \left[e^{-ihV(x_j)} u_N(\underline{x})_j\right]_{j=-\frac{N}{2}}^{\frac{N}{2}-1}. \tag{4}$$



(a) Wellenpaket φ am Anfang.

(b) Wellenpaket φ im Zeitschritt 273.

Abbildung 1 – Zeitentwicklung eines Wellenpakets

a) Verwenden Sie (3) und (4) um das Strang-Splitting für (1) via der formalen Lösung von Gleichung (2) zu implementieren.

Hinweis: benutzen Sie das Template `1_tdse.py`. Alle benötigten Parameter sind dort zu finden.

b) Bleiben die Norm und die Energie erhalten? Testen Sie mit folgenden Potentialen:

- Harmonisch: $V(x) = \frac{1}{2}x^2$
- Morse Potential: $V(x) = V_0\left(1 + e^{-2\beta(x+1)} - 2e^{-\beta(x+1)}\right)$ mit den Parametern $V_0 = 8$ und $\beta = \frac{1}{4}$.

und Anfangswerten ($\varepsilon = 10^{-2}$):

$$g_0(x) = \left(\frac{1}{\pi\varepsilon}\right)^{\frac{1}{4}} e^{-\frac{1}{2\epsilon}(x+\frac{1}{2})^2} e^{-i\frac{x}{\varepsilon}},$$

$$g_1(x) = \left(\frac{1}{\pi\varepsilon}\right)^{\frac{1}{4}} e^{-\frac{1}{2\epsilon}x^2}.$$

Plotten Sie das Potential und die Lösungen.

2. *Faltung*

Seien f und g zwei 1-periodische glatte Funktionen mit $\hat{f}(0) = 0$. Wir notieren:

$$I(x) = \int_0^x f(t)\mathrm{d}t$$
$$D(x) = g'(x)\,.$$

Sei die Faltung von f und g definiert als:

$$(f * g)(x) = \int_0^1 f(t)g(x-t)\mathrm{d}t\,. \tag{5}$$

Zeige, dass $I * D = f * g$ gilt. Begründe warum die Bedingung an f notwendig ist.

Bitte wenden!

3. *Diskrete Fouriertransformation und Faltung*

Wir wollen zu zwei Vektoren $\underline{x},\ \underline{y} \in \mathbb{R}^n$ die *periodische diskrete Faltung* berechnen, das heisst wir suchen einen Vektor $\underline{z} \in \mathbb{R}^n$ mit

$$z_k = \sum_{j=0}^{n-1} x_{k-j} y_j, \quad k = 0, \ldots, n-1\,. \tag{6}$$

Alle Indizes sind modulo n zu verstehen.

a) Sei $\hat{v}_l$ die l-te Komponente der diskreten Fouriertransformierten des Vektors $\underline{v} \in \mathbb{R}^n$, das heisst

$$\hat{v}_l = \sum_{k=0}^{n-1} v_k e^{-\frac{2\pi i k l}{n}}, \quad l = 0, \ldots, n-1\,. \tag{7}$$

Zeige, dass zwischen $\underline{x}$, $\underline{y}$ und $\underline{z}$ die folgende Beziehung gilt:

$$\hat{z}_l = \hat{x}_l \cdot \hat{y}_l, \quad l = 0, \ldots, n-1\,. \tag{8}$$

b) Implementiere eine `Python`-Funktion `def fft_conv(x,y)`, welche mit Hilfe der diskreten Fouriertransformation die periodische diskrete Faltung von $\underline{x}$ und $\underline{y}$ berechnet. Verwende `fft` und `ifft` aus `numpy.fft`.

c) Implementiere eine `Python`-Funktion `poly_multfft(a,b)`, die die Koeffizienten c_n des Produktes $p(x)q(x) = \sum_{n=0}^{2n-2} c_n x^n$ zweier Polynome $p(x) = \sum_{i=0}^{n-1} a_i x^i$ und $q(x) = \sum_{i=0}^{n-1} b_i x^i$ mithilfe der periodischen diskreten Faltung (`fft_conv(x,y)`) bestimmt.
Validiere deine Implementierung anhand der naiven Implementierung in `poly_mult(a,b)` in `3_discrete_convolution.py`.

Hinweis: c_i ist die sogenannte *nicht-periodische Faltung* von $(a_0, \ldots, a_{n-1})$ und $(b_0, \ldots, b_{n-1})$.

d) Welche der beiden Implementierungen ist für grosse n effizienter? Begründe deine Antwort.

4. *Clenshaw-Curtis Quadratur*

Eine Funktion $f : [-1, 1] \to \mathbb{R}$ soll mit der Clenshaw-Curtis Quadraturregel integriert werden. Dazu substituieren wir $x = \cos\theta$:

$$\int_{-1}^{1} f(x)\,\mathrm{d}x = \int_{0}^{\pi} f(\cos\theta)\sin\theta\,\mathrm{d}\theta. \tag{9}$$

Da $f(\cos\theta)$ 2π-periodisch ist, lässt es sich in eine Fourier-Reihe entwickeln:

$$f(\cos\theta) \approx a_0 + \sum_{k=1}^{N} a_k \cos k\theta + b_k \sin k\theta \tag{10}$$

Wobei $b_k \equiv 0$, da $f(\cos\theta)$ gerade ist.

Einsetzen von (10) in (9) ergibt:

$$\int_{0}^{\pi} f(\cos\theta)\sin\theta\,\mathrm{d}\theta \approx \sum_{k=0}^{N} \hat{w}_k a_k \tag{11}$$

wobei

$$\hat{w}_k = \int_{0}^{\pi} \cos k\theta \sin\theta\,\mathrm{d}\theta = \frac{1+\cos k\pi}{1-k^2} = \begin{cases} \frac{2}{1-k^2} & k \text{ gerade} \\ 0 & \text{sonst} \end{cases}. \tag{12}$$

Und somit gilt für N gerade:

$$\int_{-1}^{1} f(x)\,\mathrm{d}x \approx \sum_{k=0}^{N/2} \frac{2}{1-(2k)^2}\, a_{2k} \tag{13}$$

a) Finde die reellen Fourierkoeffizienten a_k mithilfe der diskreten Fouriertransformation von $f(\cos\theta)$. Mache das wie folgt:

1. Die Funktion `numpy.fft.fft` berechnet die komplexen Fourier-Koeffizienten z_l mittels der schnellen Fouriertransformation.

$$f(\cos(\theta)) \approx \sum_{l=-\frac{N}{2}}^{\frac{N}{2}-1} z_l \exp(\imath l\theta) \tag{14}$$

   Finde heraus wie `numpy.fft.fft` die Frequenzen sortiert.
   Hinweis: Lies den Abschnitt *Implementation Details* in `help(numpy.fft)` aufmerksam.
2. Schreibe a_k in Abhängigkeit von z_l.
   Hinweis: Spiegle $f(\cos\theta)$ an der vertikalen Achse bei $\theta = \pi$.

b) Berechne das Integral (13) auf $[-1, 1]$ für folgende Funktionen:

- $|x|$
- e^{-x^2}
- $\frac{2}{100}\exp(3x)\sin(300x) + \tanh(20\sin(12x))$

Verwende Chebychev-Abszissen $x_k = \cos(\pi k/N)$ mit $k = 0, \ldots, N$ und $N = 2^{2,\ldots,10}$ für den Variabelwechsel und plotte den absoluten Fehler in einem logarithmischen Plot.

Hinweis: Template `4_clenshaw_curtis.py` verwenden.