Zustandsunabhängige Kontrolle eines Qubits

Markus Fleck, Florian Wodlei

Institut für Theoretische Physik

Juli 2007

Qubit Manipulationen

Zur Realisierung eines Quantencomputers benötigt man zwei Arten von Qubit-Manipulationen:

- Verschränkung der Qubits
- Manipulation der einzelnen Qubits

Diese Manipulationen werden durch unitäre Operatoren dargestellt.

Qubit Manipulationen

Qubit:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle$$

Allgemeine Operation:

$$U\,|\psi\rangle = \alpha'\,|0\rangle + \beta'\,|1\rangle$$

wobei U eine unitäre Matrix ist.

Hamilton und Propagator

$$H = H_o + \varepsilon_x(t)\sigma_x + \varepsilon_z(t)\sigma_z$$

wobei

$$H_o = \varepsilon_o \sigma_z$$

Daraus folgt die unitäre Transformation

$$U(t_f, 0) = T(e^{-\frac{i}{\hbar}\int_o^{t_f} dt' H(t')})$$

Ziel

$\varepsilon_x(t)$ und $\varepsilon_z(t)$ zu finden, für welches $U(t_f, 0) = \mathcal{O}$ wird. Ohne $\psi(t = 0)$ vorzugeben.
wobei $\mathcal{O}$ die gewünschte unitäre Transformation ist. (z.B. X-Gatter)

Optimierung

Zur Optimierung verwenden wir das Kostenfunktional

$$\mathcal{J} = -|Tr(\mathcal{O}^{\dagger} U(t_f, 0)|^2 + \alpha \int_0^{t_f} (\varepsilon_x^2(t) + \varepsilon_z^2(t)) dt$$

α ....Gewichtungsfaktor

Dynamik des Propagators

$U(t,0)$ wird in einem zwei-Niveau System durch eine komplexe 2x2 Matrix dargestellt.
Die Trajektorie ihrer 8 Variablen ist für gegebene Felder $\varepsilon_x(t)$, $\varepsilon_z(t)$ gegen durch:

$$i\hbar\frac{\partial}{\partial t}U = HU$$

mit der Anfangsbedingung

$$U(0,0) = \mathbb{1}$$

Nummerische Propagation

Dieses gekoppelte Differentialgleichungssystem kann z.B. mittels Runge Kutta Verfahren vierter Ordnung gelöst werden.

→RK4.c (Numerical Recipes)

Nummerische Optimierung

Durch die nötige Diskretisierung wird das Zeitintervall der Propagation in N Teilschritte zerlegt für die jeweils ein Startwert $\varepsilon_x(t_j)$, $\varepsilon_z(t_j)$ vorgegeben wird.
Diese 2N Variablen können nun mittels Conjugate Gradient Processing für das Kostenfunktional variiert werden. Dieses Verfahren benötigt sowohl den Wert des Kostenfunktionals wie auch dessen Ableitungen nach allen Variablen $\varepsilon_i t_j$).

$\longrightarrow$ frprmn.c (Numerical Recipes) Polak-Ribiere/Fletcher-Reeves.

frprmn.c

void frprmn(double p[], int n, double ftol, int *iter, double *fret,
double (*func)(double p[]), void (*dfunc)(double p[], double dp[]))

p[].....zu variierendes Variablenfeld (Startwerte initialisieren)
n.....Anzahl der Variablen
ftol....Genauigkeit
iter....Output der benötigten Iterationen
fret....Funktionswert am errechneten Minimum

func....Berechnung des Funktionswertes (Kostenfunktional)
dfunc...Ableitungen nach den Variablen

Kostenfuktional

$$\mathcal{J} = -\underbrace{|Tr(\mathcal{O}^\dagger U(t_f,0)|^2}_{|c|^2} + \alpha \underbrace{\sum_{i=0}^{N-1}(\varepsilon_x^2(t_i)+\varepsilon_z^2(t_i))\Delta t}_{d}$$

Variation:

$$\frac{\partial d}{\partial \varepsilon_i(t_k)} = 2\varepsilon_i(t_k)\Delta t$$

$$\frac{\partial |c|^2}{\partial \varepsilon_i(t_k)} = \frac{\partial c}{\partial \varepsilon_i(t_k)} c* + \frac{\partial c*}{\partial \varepsilon_i(t_k)} c$$

wobei

$$\frac{\partial c}{\partial \varepsilon_i(t_k)} = Tr\left(\mathcal{O}^\dagger \frac{\partial U}{\partial \varepsilon_i(t_k)}\right)$$

Kostenfuktional

$$U = \prod_{i=1}^{N} e^{-\frac{i}{\hbar}\Delta t \; H(t_i)}$$

D.h.

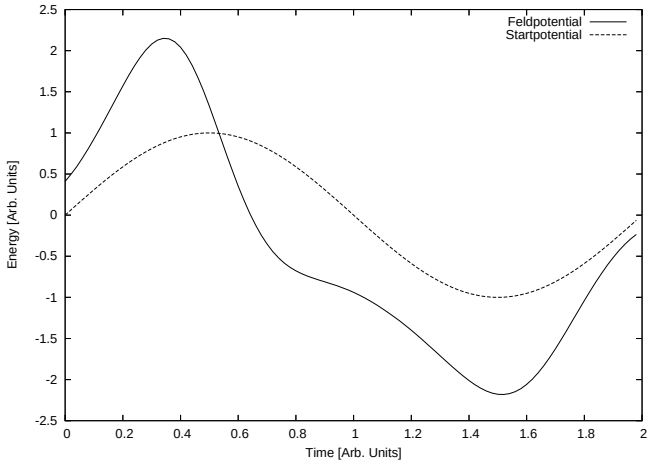
$$\frac{\partial U}{\partial \varepsilon_i(t_j)} = \underbrace{U(t_f, t_{j+1})}_{U(t_f,0)U^\dagger(t_j,0)} \left(-\frac{i\Delta t}{\hbar}\sigma_i\right) U(t_{j-1}, 0)$$

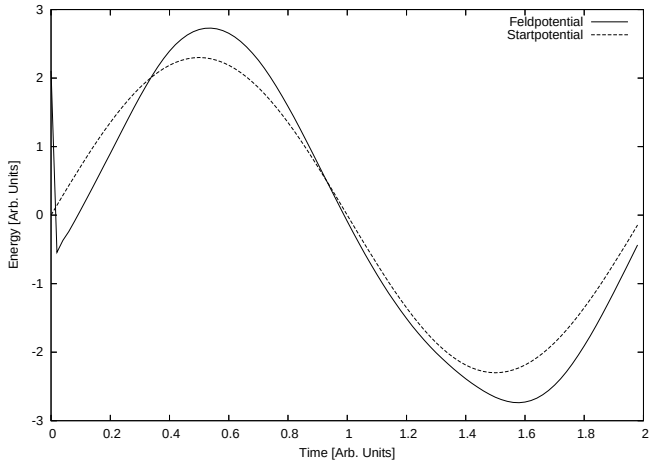
X-Gatter

$$\begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

$$\begin{pmatrix} 0,1+0,2i & 0,17-0,96i \\ -0,17-0,96i & 0,1-0,2i \end{pmatrix}$$

$J = 0.31$, $N = 100$, $t_f = 2$, $\alpha = 0.001$, $A_x = 1$, $A_z = 2.3$, $\varepsilon_o = 1$



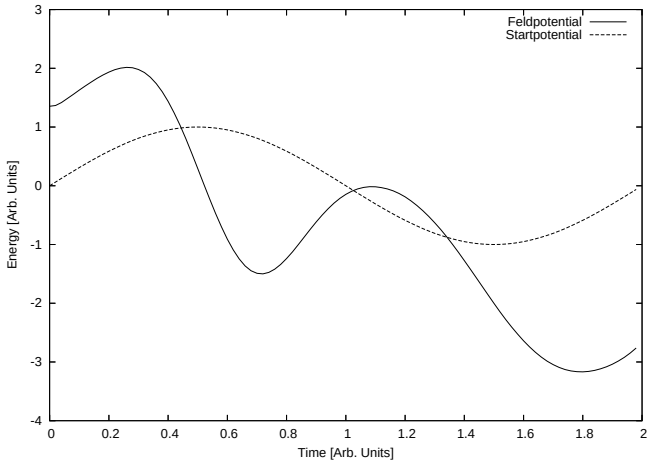


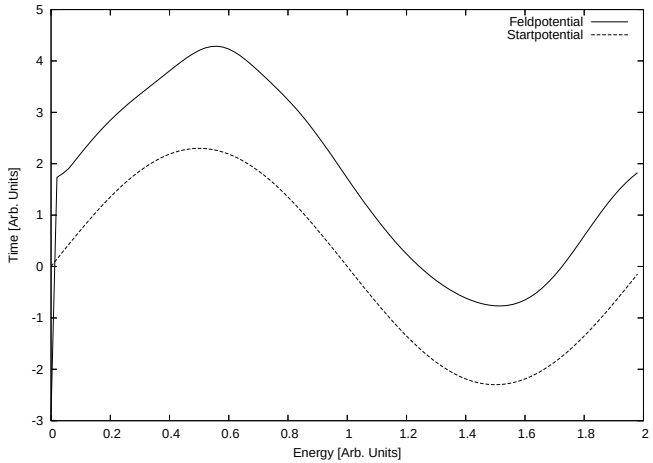
Y-Gatter

$$\begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}$$

$$\begin{pmatrix} -0,09 - 0,09i & 0,99 - 0,1i \\ -0,99 - 0,1i & -0,09 + 0,09i \end{pmatrix}$$

$J = 0.12$, $N = 100$, $t_f = 2$, $\alpha = 0.001$, $A_x = 1$, $A_z = 2.3$, $\varepsilon_o = 1$



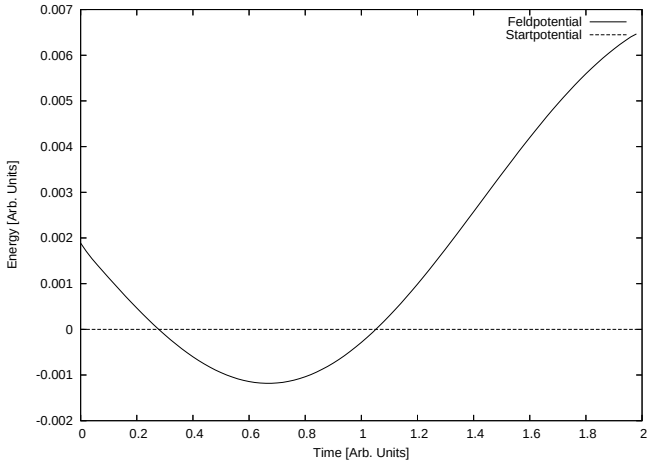


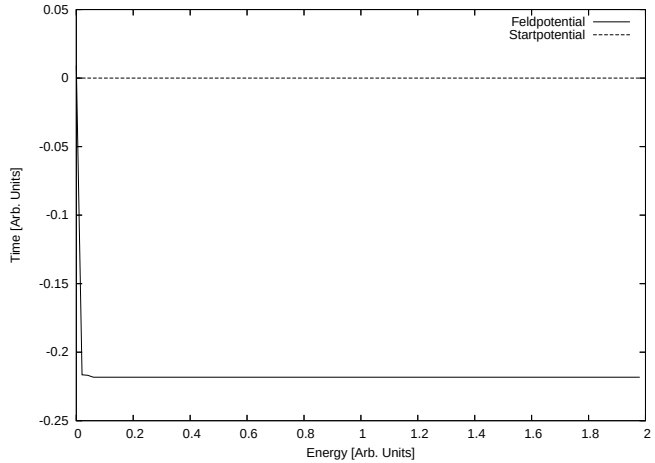
Z-Gatter

$$\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}$$

$$\begin{pmatrix} -0,001 - 0,999i & 0,003 - 0,0009i \\ -0,003 - 0,0009i & -0,001 + 0,999i \end{pmatrix}$$

$J = 0.00013$, $N = 100$, $t_f = 2$, $\alpha = 0.001$, $A_x = 0$, $A_z = 0$, $\varepsilon_o = 1$



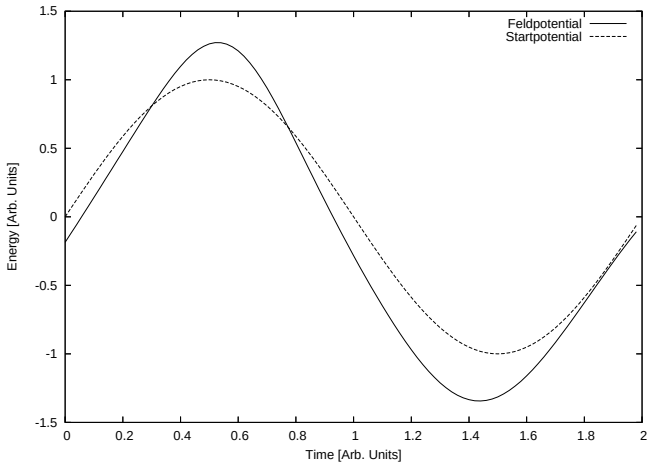


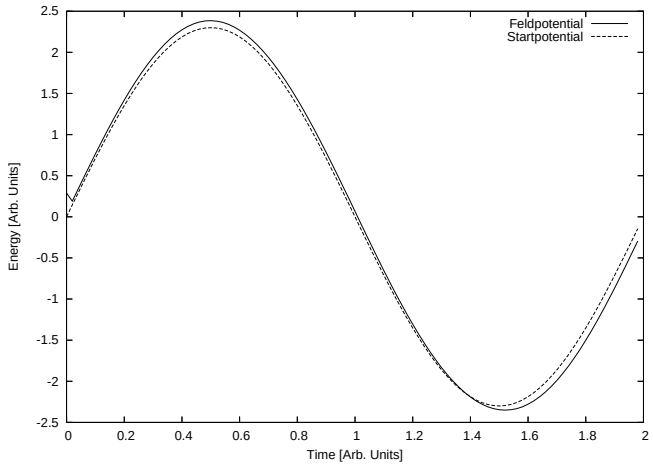
Hadamard-Gatter

$$\begin{pmatrix} \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & -\frac{1}{\sqrt{2}} \end{pmatrix}$$

$$\begin{pmatrix} -0,05 - 0,59i & -0,02 - 0,81i \\ 0,02 - 0,81i & -0,05 + 0,59i \end{pmatrix}$$

$J = 0.12$, $N = 100$, $t_f = 2$, $\alpha = 0.001$, $A_x = 1$, $A_z = 2.3$, $\varepsilon_o = 1$





Zusammenfassung

- Optimierung greift direkt am Propagator an (zustandsunabhängig)
- findet ohne Verwendung eines Lagrangen Multiplikators statt
- Runge Kutta kann im 2-Niveau System analytisch ersetzt werden.