Module V, Lecture 03: Introduction to optimal control theory

Consider a physical system with the Hamiltonian partitioned in the following way:

$$\hat{H}(t)=\hat{H}^{(0)}+\sum_{n}c^{(n)}(t)\hat{H}^{(n)} \tag{1}$$

where $\hat{H}^{(0)}$ is the part that we cannot influence with our instrument (colloquially known as the "drift") and $c^{(n)}(t)$ are time-dependent coefficients in front of the interactions described by $\hat{H}^{(n)}$ (for example external electromagnetic fields) that we can control. These operators are known as "controls".

Our problem consists in finding such $\left\{c^{(n)}(t)\right\}$ as would take our system with the best possible accuracy from some initial state $\left|\psi_0\right\rangle$ to some destination state $\left|\sigma\right\rangle$ with (potentially) the following constrains:

1. The maximum amplitude of $c^{(n)}(t)$ may be fixed because instruments cannot output arbitrarily strong electromagnetic fields and sample heating must be avoided.
2. The maximum frequency present in $c^{(n)}(t)$ may be fixed because waveform synthesis hardware has a finite switching time.
3. A distribution with respect to $\hat{H}^{(0)}$ may exist in the ensemble because hardware cannot maintain perfect uniformity in the various experiment parameters across the sample.
4. Some generalised running cost may be present, for example

$$\int_{0}^{T}\lambda(t)c^{(n)}(t)\,dt \tag{2}$$

   that must be minimised at the same time as fidelity is maximised to achieve a balance.
5. The total time available to perform the transfer may be limited or may need to be minimised.

To solve this problem, consider the following functional:

$$J\left[\left\{c^{(n)}(t)\right\}\right]=\mathrm{Re}\left\langle\sigma\right|\exp_{(\mathrm{O})}\left[-i\int_{0}^{T}\left(\hat{H}^{(0)}+\sum_{n}c^{(n)}(t)\hat{H}^{(n)}\right)dt\right]\left|\psi_0\right\rangle \tag{3}$$

where $\exp_{(\mathrm{O})}$ denotes a time-ordered exponential propagator. If $\left|\psi_0\right\rangle$ is transported completely into $\left|\sigma\right\rangle$, this functional would reach the maximum value of 1, otherwise it would return the extent of the wavefunction overlap at the end of the experiment. We therefore need to find the global maximum of $J\left[\left\{c^{(n)}(t)\right\}\right]$ with respect to $\left\{c^{(n)}(t)\right\}$. At the same time, the cost should be kept to a minimum. Therefore, the quantity to be maximised is

$$J\left[\left\{c^{(n)}(t)\right\}\right]-K\left[\left\{c^{(n)}(t)\right\}\right] \tag{4}$$

where $K\left[\left\{c^{(n)}(t)\right\}\right]$ is the cost functional.

Gradient ascent pulse engineering (GRAPE)

The general variational problem described above is much simplified if the control sequences $\left\{c^{(n)}(t)\right\}$ are assumed to be piecewise-constant. For a piecewise-constant Hamiltonian:

$$\exp_{(\mathrm{O})}\left[-i\int_{0}^{T}\left(\hat{H}^{(0)}+\sum_{n}c^{(n)}(t)\hat{H}^{(n)}\right)dt\right]=\prod_{k}^{\rightarrow}\exp\left[-i\left(\hat{H}^{(0)}+\sum_{n}c^{(n)}(t_k)\hat{H}^{(n)}\right)\Delta t_k\right] \tag{5}$$

where the arrow indicates a time-ordered product and Δt_k is the duration of the *k*-th time interval. The simulation may therefore be carried out by sequentially multiplying the time slice propagators into the initial condition:

$$\begin{gathered}\left|\psi_k\right\rangle=\hat{U}_k\hat{U}_{k-1}\cdots\hat{U}_2\hat{U}_1\left|\psi_0\right\rangle \\ \hat{U}_k=\exp\left[-i\left(\hat{H}^{(0)}+\sum_n c^{(n)}\left(t_k\right)\hat{H}^{(n)}\right)\Delta t_k\right]\end{gathered} \tag{6}$$

Let us now change notation: we will put $c^{(n)}\left(t_k\right)=c_k^{(n)}$ and, for simplicity assume that all time intervals Δt_k are of the same duration Δt. The expression for our fidelity functional then becomes:

$$\begin{gathered}J=\left\langle\sigma\left|\hat{U}_k\hat{U}_{k-1}\cdots\hat{U}_2\hat{U}_1\right|\psi_0\right\rangle \\ \hat{U}_k=\exp\left[-i\left(\hat{H}^{(0)}+\sum_n c_k^{(n)}\hat{H}^{(n)}\right)\Delta t\right]\end{gathered} \tag{7}$$

The arrays $\left\{c_k^{(n)}\right\}$ are now vectors of finite dimension and we can use standard non-linear optimisation theory to find the maximum of J in their space. What we would need for that, however, is the gradient of J with respect to $\left\{c_k^{(n)}\right\}$. The task is therefore narrowed down to computing $\partial J/\partial c_k^{(n)}$.

The property that makes GRAPE algorithm very efficient is the fact that the only thing in Equation (7) that depends on $c_k^{(n)}$ is $\hat{U}_k$. All other slice propagators do not depend on $c_k^{(n)}$. Therefore, the following very efficient scheme may be used for computing the gradient:

$$\begin{gathered}J=\left\langle\sigma\left|\hat{U}_K\hat{U}_{K-1}\cdots\hat{U}_{k+1}\hat{U}_k\right.\overbrace{\hat{U}_{k-1}\cdots\hat{U}_2\hat{U}_1\left|\psi_0\right\rangle}^{\text{propagate forward from source}}\right. \\ \uparrow \\ \left[\frac{\partial}{\partial c_k^{(n)}}U_k\right] \\ \uparrow \\ J=\underbrace{\left\langle\sigma\left|\hat{U}_K\hat{U}_{K-1}\cdots\hat{U}_{k+1}\right.\right.}_{\text{propagate backward from destination}}\hat{U}_k\hat{U}_{k-1}\cdots\hat{U}_2\hat{U}_1\left|\psi_0\right\rangle\end{gathered} \tag{8}$$

This essentially means that the system trajectory needs to be calculated forward (from the initial condition) and backward (from the destination state), and then scalar products between the pieces of the two trajectories should be taken with the propagator derivative at each step. Therefore, only two trajectory evaluations are necessary (K matrix exponentials because they can be re-used) and NK matrix exponential derivatives, where K is the number of points in the trajectory and N is the number of control operators. The computational cost of this remarkably efficient procedure should be compared to the cost of computing the same gradient with finite differences: at least NK trajectory evaluations, meaning NK^2 matrix exponentials. This efficiency is the primary reason why GRAPE procedure is so popular.

The remaining task is to calculate the derivatives of the propagators in Equation (7) with respect to the control coefficients. We will simplify notation somewhat and define

$$\hat{H}_k=\hat{H}^{(0)}+\sum_n c_k^{(n)}\hat{H}^{(n)} \tag{9}$$

We are looking to compute

$$\frac{\partial}{\partial c_k^{(n)}}\exp\left(-i\hat{H}_k\Delta t\right) \tag{10}$$

That is a straightforward procedure for which three popular methods exist:

1. Differentiate a power series (Taylor, Chebyshev, Newton, *etc.*). In the case of Taylor series:

$$\begin{aligned}\frac{\partial}{\partial c_k^{(n)}}\exp\left(-i\hat{H}_k\Delta t\right)&=\frac{\partial}{\partial c_k^{(n)}}\sum_{m=0}^{\infty}\frac{\left(-i\Delta t\right)^m}{m!}\hat{H}_k^m=\\&=\sum_{m=0}^{\infty}\frac{\left(-i\Delta t\right)^m}{m!}\frac{\partial}{\partial c_k^{(n)}}\underbrace{\left(\hat{H}_k\hat{H}_k\cdots\hat{H}_k\right)}_{m\text{ times}}=\\&=\sum_{m=0}^{\infty}\frac{\left(-i\Delta t\right)^m}{m!}\sum_{l=1}^{m}\left(\hat{H}_k\hat{H}_k\cdots\underbrace{\hat{H}^{(n)}}_{l^{\text{th}}\text{ position}}\ldots\hat{H}_k\right)\end{aligned} \tag{11}$$

   This expression has the downside of involving the double summation, this makes the evaluation procedure quite slow.

2. Use a commutator series. We can rotate summation indices in Equation (11) to get the following single-sum expression (the derivation is too long to be reproduced here):

$$\frac{\partial}{\partial c_k^{(n)}}\exp\left(-i\hat{H}_k\Delta t\right)=\exp\left(-i\hat{H}_k\Delta t\right)\sum_{m=0}^{\infty}\frac{\left(-i\Delta t\right)^m}{m!}\left[\hat{H}_k,\hat{H}^{(n)}\right]_m \tag{12}$$

   where $\left[A,B\right]_m$ denotes an *m*-fold nested commutator and $\left[A,B\right]_0=B$. This expression is faster to compute than Equation (11), but it is still inconvenient because a nested commutator series is involved.

3. The most convenient practical route to the matrix exponential derivative is to use auxiliary matrices. The following relations will be given here without proof:

$$\begin{pmatrix}\hat{U}_k & \dfrac{\partial\hat{U}_k}{\partial c_k^{(n)}} & \hat{S}_k^{(n,m)}\\ 0 & \hat{U}_k & \dfrac{\partial\hat{U}_k}{\partial c_k^{(m)}}\\ 0 & 0 & \hat{U}_k\end{pmatrix}=\exp\left[-i\begin{pmatrix}\hat{H}_k & \hat{H}^{(n)} & 0\\ 0 & \hat{H}_k & \hat{H}^{(m)}\\ 0 & 0 & \hat{H}_k\end{pmatrix}\Delta t\right],\qquad \frac{\partial^2\hat{U}_k}{\partial c_k^{(n)}\partial c_k^{(m)}}=\hat{S}_k^{(n,m)}+\hat{S}_k^{(m,n)} \tag{13}$$

   This method is currently the default in Spinach.

Whichever method is used, the result is the gradient of the fidelity functional with respect to the control sequence. Any of the numerous gradient-based optimisation methods (gradient descent, conjugate gradients, BFGS, *etc.*) may then be used to drive the fidelity to the maximum. Recent literature suggests that the Newton-Raphson method, which also uses second derivative information, performs best.

Instrument response functions

Another relevant problem is that the control sequence that the computer emits might not necessarily be the sequence that the sample actually experiences – the numerous cables and amplifiers inevitably introduce som distortions. In the linear response approximation, the "ideal" control sequence $s(t)$ and

the "real" control sequence $S(t)$ actually experienced by the sample is are related via a convolution with the instrument response function $g(t)$:

$$S(t) = g(t) * s(t) \tag{14}$$

In matrix form:

$$\vec{S} = \mathbf{G}\vec{s} \tag{15}$$

where $\mathbf{G}$ is the response matrix. It may be measured experimentally by submitting several linearly independent control sequences $\vec{s}_k$ from the computer and measuring the arriving electromagnetic fields $\vec{S}_k$ at the sample points, then solving the resulting system of equations for $\mathbf{G}$ in the least squares sense, for example using Matlab's backslash operation.