Quadratically Convergent Optimal Control Algorithms: Newton-GRAPE

D.L. Goodwin[1], Ilya Kuprov[1]

[1]University of Southampton, UK

(Published in The Journal of Chemical Physics, 144 (20) 2016, 204107)

UNIVERSITY OF Southampton

SpinDynamics.org
Theoretical and Computational Spin Dynamics Group

Introduction

In quantum mechanics a popular formulation is to specify the desired state vector of the system and maximise the following fidelity functional with respect to the instrumentally controllable part of the Hamiltonian $\mathbf{H}_1(t)$:

$$J\left[\mathbf{H}_1(t)\right]=\mathrm{Re}\left\langle\boldsymbol{\delta}\right|\exp_{(\mathrm{O})}\left[-i\int_0^T\left(\mathbf{H}_0+\mathbf{H}_1(t)+i\mathbf{R}\right)dt\right]\left|\boldsymbol{\rho}_0\right\rangle-J_{\mathrm{RF}}\left[\mathbf{H}_1(t)\right]$$

where $\exp_{(\mathrm{O})}$ indicates a time-ordered exponential, $\mathbf{H}_0$ is the part of the system Hamiltonian that cannot be controlled, $\mathbf{R}$ is the relaxation operator, $\boldsymbol{\rho}_0$ is the initial state vector and J_{RF} is a penalty functional. The gradient ascent pulse engineering (GRAPE) method [1,2] proceeds by splitting the Hamiltonian into the uncontrollable part and a number of control operators with time-dependent coefficients:

$$\mathbf{H}(t)=\mathbf{H}_0+\sum_{k=1}^{K}c_k(t)\mathbf{H}_k$$

The control coefficients are then discretized on a finite grid of time points:

$$c_k(t)\quad\rightarrow\quad\mathbf{c}_k=\left[c_k(t_1)\quad c_k(t_2)\quad\cdots\quad c_k(t_N)\right],\qquad t_1<t_2<\ldots<t_N$$

With a piecewise-constant Hamiltonian, the time-ordered exponential becomes

$$\exp_{(\mathrm{O})}\left[-i\int_0^T\left(\mathbf{H}_0+\mathbf{H}_1(t)+i\mathbf{R}\right)dt\right]=\prod_n\exp\left[-i\left(\mathbf{H}_0+\sum_{k=1}^{K}c_{k,n}\mathbf{H}_k+i\mathbf{R}\right)\Delta t\right]$$

with a time-ordered product, and the equation itself acquires the following form:

$$J(\mathbf{c}_1,\ldots,\mathbf{c}_K)=\mathrm{Re}\left\langle\boldsymbol{\delta}\right|\mathbf{P}_N\ldots\mathbf{P}_2\mathbf{P}_1\left|\boldsymbol{\rho}_0\right\rangle-J_{\mathrm{RF}}(\mathbf{c}_1,\ldots,\mathbf{c}_K),\qquad\mathbf{P}_n=\exp\left[-i\left(\mathbf{H}_0+\sum_{k=1}^{K}c_{k,n}\mathbf{H}_k+i\mathbf{R}\right)\Delta t\right]$$

where the k index runs over the control channels and the n index runs over the time steps. A particular strength of the GRAPE method is that the gradient of the fidelity functional

$$\frac{\partial J}{\partial c_{k,n}}=\mathrm{Re}\left\langle\boldsymbol{\delta}\right|\mathbf{P}_N\cdots\mathbf{P}_{n+1}\frac{\partial\mathbf{P}_n}{\partial c_{k,n}}\mathbf{P}_{n-1}\cdots\mathbf{P}_1\left|\boldsymbol{\rho}_0\right\rangle$$

has the same numerical complexity scaling as the fidelity functional itself [1]. This also applies to the Hessian [3]:

$$\frac{\partial^2 J}{\partial c_{j,n}\partial c_{k,n}}=\mathrm{Re}\left\langle\boldsymbol{\delta}\right|\mathbf{P}_N\cdots\mathbf{P}_{n+1}\frac{\partial^2\mathbf{P}_n}{\partial c_{j,n}\partial c_{k,n}}\mathbf{P}_{n-1}\cdots\mathbf{P}_1\left|\boldsymbol{\rho}_0\right\rangle$$

$$\frac{\partial^2 J}{\partial c_{j,n}\partial c_{k,m}}=\mathrm{Re}\left\langle\boldsymbol{\delta}\right|\mathbf{P}_N\cdots\mathbf{P}_{n+1}\frac{\partial\mathbf{P}_n}{\partial c_{j,n}}\mathbf{P}_{n-1}\cdots\mathbf{P}_{m+1}\frac{\partial\mathbf{P}_m}{\partial c_{k,m}}\mathbf{P}_{m-1}\cdots\mathbf{P}_1\left|\boldsymbol{\rho}_0\right\rangle$$

Hessian Regularisation

Hessians are normally expensive to compute, with computational complexity O(n^2). The recent quasi-Newton BFGS-GRAPE algorithm [2] avoids their calculation by recovering approximate second derivative information from the gradient history, scaling with O(n). The fact that the Hessian is cheap [3] suggests that Newton-Raphson type algorithms with the control sequence update rule at step s, $\mathbf{c}^{(s+1)}=\mathbf{c}^{(s)}-\left[\nabla^2J^{(s)}\right]^{-1}\nabla J^{(s)}$, (gradient ∇J and Hessian $\nabla^2 J$ of the fidelity functional $J(\mathbf{c})$) should be the next logical step.

Newton-Raphson and quasi-Newton methods (minimisation is assumed here) rely on the necessary conditions for Taylor's theorem and use a local quadratic approximation:

$$J(\mathbf{c}+\Delta\mathbf{c})\approx J(\mathbf{c})+\left\langle\nabla J(\mathbf{c})\right|\Delta\mathbf{c}\rangle+\tfrac{1}{2}\left\langle\Delta\mathbf{c}\right|\nabla^2J(\mathbf{c})\left|\Delta\mathbf{c}\right\rangle$$

First order necessary condition requires any minimiser $\tilde{\mathbf{c}}$ to be a stationary point ($\nabla J(\tilde{\mathbf{c}})=0$). The second order necessary condition is that the Hessian $\nabla^2 J$ should be positive definite at $\tilde{\mathbf{c}}$.

Far away from a minimiser, the Hessian is not actually expected to be positive definite. Small Hessian eigenvalues are also problematic, resulting in overly long steps (most fidelity functionals are not actually quadratic). A cheap way to detect an indefinite Hessian is to attempt Cholesky factorisation, which exists for any invertible positive definite matrix.

An eigenvalue shifting method (aka trust region method, TRM) can shift the eigenspectrum enough to ensure the Hessian is not near-singular. This method needs an explicit eigendecomposition [4], to make a precise estimate of the shifting value σ:

$$\left[\nabla^2J\right]=\mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^{-1},\qquad\sigma=\max(0,\delta-\min(\boldsymbol{\Lambda}_{ii})),\qquad\left[\nabla^2J\right]_{\mathrm{reg}}=\mathbf{Q}(\boldsymbol{\Lambda}+\sigma\mathbf{1})\mathbf{Q}^{-1}$$

where $\boldsymbol{\Lambda}$ is a diagonal matrix containing the eigenvalues of $\nabla^2 J$ and $\mathbf{Q}$ is the matrix with columns made up of corresponding eigenvectors. A user-specified positive value of δ is included to make the Hessian positive definite. A problem with this method is that the regularisation proceedure destroys much of the curvature information

The method that was found to perform best in our practical testing is known as rational function optimization (RFO) [5,6,7]. It replaces the Taylor expansion with a Padé approximant:

$$\Delta J=\frac{\left\langle\nabla J\right|\mathbf{c}\rangle+\frac{1}{2}\left\langle\mathbf{c}\right|\nabla^2J\left|\mathbf{c}\right\rangle}{1+\left\langle\mathbf{c}\right|\mathbf{S}\left|\mathbf{c}\right\rangle}$$

where S is a symmetric scaling matrix. This preserves derivative information, leaving the necessary conditions unchanged. The first order necessary condition with a uniform scaling matrix [7] $\mathbf{S}=\alpha^{-2}\mathbf{1}$, where $0<\alpha<1$, gives

$$\begin{pmatrix}\alpha^2\nabla^2J & \alpha\nabla J\\ \alpha\nabla J^T & \mathbf{0}\end{pmatrix}\begin{pmatrix}\mathbf{c}/\alpha\\ \mathbf{1}\end{pmatrix}=2\Delta J\begin{pmatrix}\mathbf{c}/\alpha\\ \mathbf{1}\end{pmatrix}$$

Rational function optimiSation proceeds in a similar way to eigenvalue shifting methods described above, except the shifting is applied to the augmented Hessian:

$$\left[\nabla^2J\right]^{\mathrm{aug}}=\begin{pmatrix}\alpha^2\nabla^2J & \alpha\nabla J\\ \alpha\nabla J^T & \mathbf{0}\end{pmatrix}=\mathbf{Q}\boldsymbol{\Lambda}\mathbf{Q}^{-1},\qquad\sigma=\max(0,-\min(\boldsymbol{\Lambda}_{ii}))$$

$$\left[\nabla^2J\right]^{\mathrm{aug}}_{\mathrm{reg}}=\frac{1}{\alpha^2}\mathbf{Q}(\boldsymbol{\Lambda}+\sigma\mathbf{1})\mathbf{Q}^{-1}$$

The top left corner block of the regulariSed augmented Hessian is then used for the Newton-Raphson step [5,6,7]. Our practical experience with the scaling constant α indicates that it should be allowed to vary, for example:

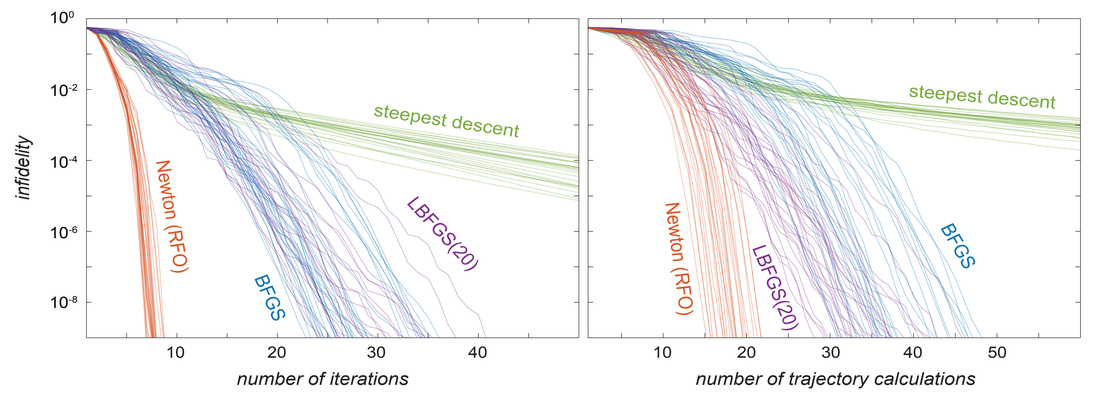
$$\alpha_{r+1}=\phi\alpha_r\qquad\text{while}\qquad\frac{\min(\boldsymbol{\Lambda}_{ii})}{\max(\boldsymbol{\Lambda}_{ii})}>\frac{1}{\sqrt[n]{\varepsilon}}$$

where ε is machine precision and $\alpha_0=1$. The factor ϕ is used to iteratively decrease the condition number of the Hessian. The n root of ε appearing is the strict limit for a line search method using polynomial interpolation of degree n.

In practice, at each optimisation step the function code attempts to compute the Cholesky decomposition. If that is successful then no regularisation is needed, otherwise the function proceeds to regularizs with the methods described above.

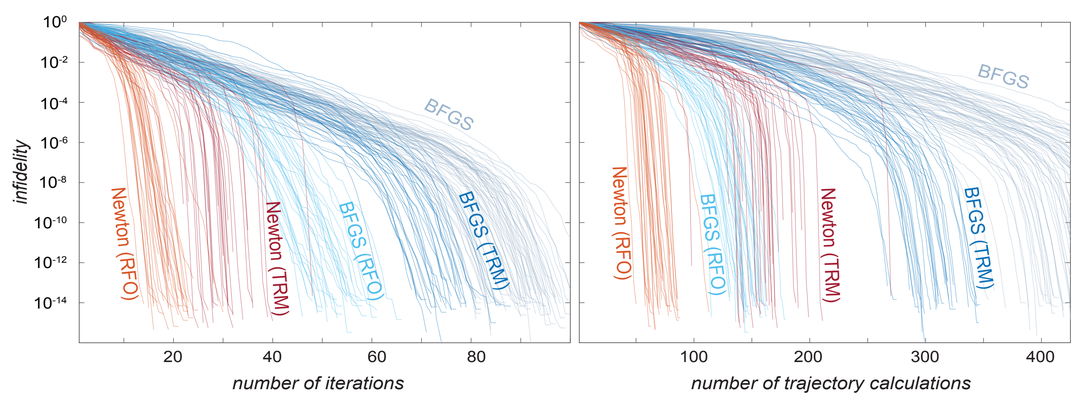
(Directly Below): Convergence profiles for the transfer of longitudinal magnetization into the singlet state for the two-spin system described in the main text. The same line search method in the predicted descent direction was used in all cases. Memory store for limited-memory BFGS was set to 20 gradients.

State transfer from longitudinal polarization into a two-spin singlet state, while allowing for up to 20% miscalibration of the control channel power level. The spin system contains two 13C spins in a 14.1 T magnet with chemical shifts of 0.00 and 0.25 ppm and a J-coupling of 60 Hz. The system is prepared with 50 time discretization points, the nominal power of 60 Hz and the duration of 50 ms is optimized simultaneously for ten different power levels spaced equally between 80% and 120% of the nominal power. Weighted norm squared penalty functional was used with equal weights for all time points. The infidelity measure refers to the distance from the best possible magnetization transfer fidelity for the system in question. Trajectory analysis diagrams and the optimal control sequences are presented below the convergence profiles



(Directly Below): Convergence profiles for the state transfer within the 1H–13C–19F three-spin system described in the main text for the BFGS quasi-Newton method and the Newton-Raphson method using TRM or RFO Hessian regularization techniques. The same line search method in the predicted descent direction was used in all cases.

State transfer in a H–C–F group in a 9.4 T magnet with 1H isotope for hydrogen, 13C isotope for carbon and 19F isotope for fluorine, with the 1H–13C J-coupling of 140 Hz, 13C–19F J-coupling of −160 Hz and all three signals assumed to be on resonance with the transmitters on the corresponding NMR spectrometer channels. A six-channel (HX, HY, CX, CY, FX, FY) shaped pulse with a duration of 100 ms, a quadratic penalty for excursions outside the 10 kHz power envelope and 50 time discretization points was optimized to perform longitudinal magnetization transfer from 1H to 19F. This system was chosen because very high terminal fidelities are achievable with the parameters described above – it is a good test of terminal convergence behaviour for quadratic optimization algorithms in finite precision arithmetic



References

[1] Khaneja et. al., J Magn. Reson., 172, 296 (2005)

[2] de Fouquieres et. al., J Magn. Reson., 212, 412 (2011)

[3] Goodwin, Kuprov, J. Chem. Phys, 143, 084113 (2015)

[4] Greenstadt, Mathematics of Computation, 21, 360 (1967)

[5] Benerjee, Grein, Int. J. Quant. Chem., 10, 123 (1976)

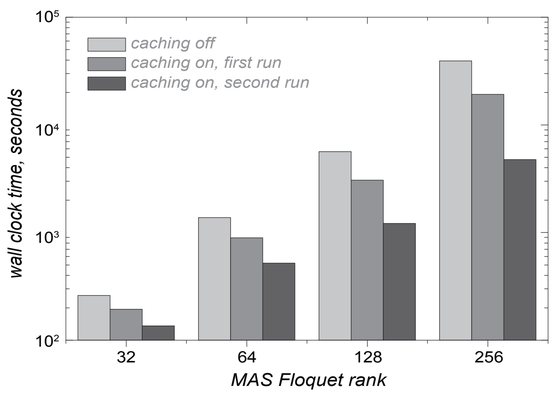
[6] Shepard, Shavitt, Simons, J. Chem. Phys, 76, 543 (1982)

[7] Banerjee et. al., J. Chem. Phys., 89, 52 (1985)

Acknowledgements

The authors are grateful to Sophie Schirmer, Stefan Stoll, Thomas Schulte-Herbrüggen and Steffen Glaser for useful discussions. This work was made possible by EPSRC through a grant EP/H003789/1 and a CDT studentship to DLG. The European Commission has facilitated this project through a coordination action grant (297861/QUAINT)..

iMR EPSRC spindynamics.org Spinach



Hashing and Caching can become very useful in optimisation, particularly when many similar evaluations are made of a particular function. A hash code is used as an identifier to be able to quickly retrieve results from a calculation that has already been evaluated.

Above is the wall clock time consumed by the simulation of the CN2D solid state NMR experiment for a 14N–13C spin pair in glycine with different matrix exponential caching settings.

Parallelisation

Our numerical implementations of gradient and Hessian calculations are parallelised with respect to the number of time. Gradient calculation uses 2 × 2 augmented exponentials [3] and Hessian calculation uses a 3 × 3 augmented exponential. The Hessian function has a further parallel loop when calculating $n\neq m$ blocks. At that stage, the first derivatives have already been calculated with the 3 × 3 augmented exponential and they are recycled.

The scaling depends on the number of time slices in the control sequence and the parallelisation is efficient all the way to the number of CPU cores being half the number of time slices (over which the parallel loop is running). Given the same computing resources, a Hessian calculation takes approximately 10 times longer than a gradient calculation. Because propagator derivatives are recycled during Hessian calculation, significant efficiency gains may be made by optimizing their storage and indexing (hashing and caching, left).

Amdahl's law parallelisation efficiency analysis for a problem with 6 control operators and 144 time slices