The Journal of Open Source Software

# GRAPE.jl: Gradient Ascent Pulse Engineering in Julia

**Michael H. Goerz** [1], **Sebastián C. Carrasco** [1], **Alastair Marshall** [2], **and Vladimir S. Malinovsky** [1]

**1** DEVCOM Army Research Laboratory, United States **2** Universität Ulm, Institute for Quantum Optics, Germany

**DOI:** 10.xxxxxx/draft

**Software**

- Review
- Repository
- Archive

**Editor:** Nikoleta Glynatsi

**Reviewers:**

- @AICap23
- @szabo137

**Submitted:** 14 July 2025
**Published:** unpublished



## Summary

The `GRAPE.jl` package implements Gradient Ascent Pulse Engineering (Khaneja et al., 2005), a widely used method of quantum optimal control (Brif et al., 2010; Brumer & Shapiro, 2003; Sola et al., 2018). Its purpose is to find "controls" that steer a quantum system in a particular way. This is a prerequisite for next-generation quantum technology (Dowling & Milburn, 2003), such as quantum computing (Nielsen & Chuang, 2000) or quantum sensing (Degen et al., 2017). For example, in quantum computing with superconducting circuits (Koch et al., 2007), the controls are microwave pulses injected into the circuit in order to realize logical operations on the quantum states of the system (e.g., Goerz et al., 2017).

The quantum state of a system can be described numerically by a complex vector $|\Psi(t)\rangle$ that evolves under a differential equation of the form

$$i\hbar\frac{\partial|\Psi(t)\rangle}{\partial t} = \hat{H}(\epsilon(t))|\Psi(t)\rangle\,, \tag{1}$$

where $\hbar$ is the reduced Planck constant and $\hat{H}$ is a matrix whose elements depend in some way on the control function $\epsilon(t)$. We generally know the initial state of the system $|\Psi(t=0)\rangle$ and want to find an $\epsilon(t)$ that minimizes some real-valued functional $J$ that depends on the state at some final time $T$, as well as running costs on $|\Psi(t)\rangle$ and values of $\epsilon(t)$ at intermediate times. A common example is the square-modulus of the overlap with a target state.

The defining feature of the GRAPE method is that it considers $\epsilon(t)$ as piecewise constant, i.e., as a vector of values $\epsilon_n$, for the $n$'th interval of the time grid. This allows solving Equation 1 for each time interval, and deriving an expression for the gradient $\partial J/\partial\epsilon_n$ of the optimization functional with respect to the values of the control field. It results in an efficient numerical scheme for evaluating the full gradient (Goerz et al., 2022, fig. 1(a)). The scheme extends to situations where the functional is evaluated on top of *multiple* propagated states $\{|\Psi_k(t)\rangle\}$ with an index $k$, and multiple controls $\epsilon_l(t)$, resulting in a vector of values $\epsilon_{nl}$ with a double-index $nl$. Once the gradient has been evaluated, in the original formulation of GRAPE (Khaneja et al., 2005), the values $\epsilon_{nl}$ would then be updated by taking a step with a fixed step width $\alpha$ in the direction of the negative gradient, to iteratively minimize the value of the optimization functional $J$. In practice, the gradient can also be fed into an arbitrary gradient-based optimizer, and in particular a quasi-Newton method like L-BFGS-B (Qi & contributors, 2022; Zhu et al., 1997). This results in a dramatic improvement in stability and convergence (Fouquières et al., 2011), and is assumed as the default in `GRAPE.jl`. Gradients of the time evolution operator can be evaluated to machine precision following Goodwin & Kuprov (2015). The GRAPE method could also be extended to a true Hessian of the optimization functional (Goodwin & Kuprov, 2016), which would be in scope for future versions of `GRAPE.jl`.

The Journal of Open Source Software

## Statement of Need

There have been a number of implementations of the GRAPE method in different contexts. GRAPE was originally developed and adopted in the NMR community, e.g., as part of `SIMPSON` (Tošner et al., 2009) in C, and later as part of `Spinach` (Hogben et al., 2011) and `pulsefinder` (Ryan & contributors, 2013) in Matlab. More recent implementations in Python, geared towards more general purposes like quantum information, are found as part of the `QuTIP` library (Johansson et al., 2013), `C3` (Wittler et al., 2021), `QuOCS` (Rossignolo et al., 2023), and `QuanEstimation` (Zhang et al., 2022). The implementation of `GRAPE.jl` is also inspired by earlier work in the `QDYN` library in Fortran (2025). `GRAPE.jl` exploits the unique strengths of the Julia programming language (Bezanson et al., 2017) to avoid common shortcomings in existing implementations.

As a compiled language geared towards scientific computing, Julia delivers numerical performance similar to that of Fortran, while providing much greater flexibility due to the expressiveness of the language. The numerical cost of the GRAPE method is dominated by the cost of evaluating the time evolution of the quantum system. `GRAPE.jl` delegates this to efficient piecewise-constant propagators in `QuantumPropagators.jl` (Goerz & contributors, 2025b) or the general-purpose `DifferentialEquations.jl` framework (Rackauckas & Nie, 2017).

`GRAPE.jl` builds on the concepts defined in `QuantumControl.jl` (Goerz & contributors, 2025a) to allow functionals that depend on an arbitrary set of "trajectories" $\{|\Psi_k(t)\rangle\}$, each evolving under a potentially different $\hat{H}_k$. In contrast to the common restriction to a single state $|\Psi\rangle$ or a single unitary $\hat{U}$ as the dynamical state, this enables ensemble optimization for robustness against noise (e.g., Goerz, Halperin, et al., 2014). The optimization over multiple trajectories is parallelized. This makes the optimization of quantum gates more efficient, by tracking the logical basis states instead of the gate $\hat{U}(t)$. Each $\hat{H}_k$ may depend on an arbitrary number of controls $\{\epsilon_l(t)\}$ in an arbitrary way, going beyond the common assumption of linear controls, $\hat{H} = \hat{H}_0 + \epsilon(t)\hat{H}_1$.

Julia's core feature of multiple dispatch allows the user to define custom, problem-specific data structures with performance-optimized linear algebra operations. This gives `GRAPE.jl` great flexibility to work with any custom data structures for quantum states $|\Psi_k(t)\rangle$ or the matrices $\hat{H}_k(\{\epsilon_l(t)\})$, and enables a wide range of applications, from NMR spin systems to superconducting circuits or trapped atoms in quantum computing, to systems with spatial degrees of freedom (e.g., Dash et al., 2024). This also includes open quantum systems, as the structure of Equation 1 holds not just for the standard Schrödinger equation, but also for the Liouville equation, where $|\Psi_k\rangle$ is replaced by a (vectorized) density matrix and $\hat{H}$ becomes a Liouvillian super-operator (Goerz, Reich, et al., 2014).

The rise of machine learning generated considerable interest in using the capabilities of frameworks like Tensorflow (Abadi et al., 2016), PyTorch (Paszke et al., 2019), or JAX (Bradbury et al., 2018) for automatic differentiation (AD) (Griewank & Walther, 2008) to evaluate the gradient of the optimization functional. This has the benefit that it allows for arbitrary functionals (Abdelhafez et al., 2019, 2020; Leung et al., 2017, 2021). In contrast, the GRAPE method and all of its existing implementations are formulated only for a "standard" set of functionals that essentially measure the overlap of a propagated state with a target state. Unfortunately, AD comes with a large numerical overhead that makes the method impractical. Goerz et al. (2022) introduced the use of "semi-automatic differentiation" that limits the numerical cost to exactly that of the traditional GRAPE scheme. It does this by employing AD only for the evaluation of the derivative $\partial J/\partial\langle\Psi_k(T)|$, and only if that derivative cannot be evaluated analytically. `GRAPE.jl` is built on the resulting generalized GRAPE scheme. As necessary, it can use any available AD framework in the Julia ecosystem to enable the minimization of non-analytical functionals, such as entanglement measures (Goerz et al., 2015; Watts et al., 2015).

The Journal of Open Source Software

## Acknowledgements

Research was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement Numbers W911NF-23-2-0128 (MG) and W911NF-24-2-0044 (SC). The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation herein.

## References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., … Zheng, X. (2016). TensorFlow: A system for large-scale machine learning. *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, 265. https://www.tensorflow.org/

Abdelhafez, M., Baker, B., Gyenis, A., Mundada, P., Houck, A. A., Schuster, D., & Koch, J. (2020). Universal gates for protected superconducting qubits using optimal control. *Phys. Rev. A*, *101*, 022321. https://doi.org/10.1103/physreva.101.022321

Abdelhafez, M., Schuster, D. I., & Koch, J. (2019). Gradient-based optimal control of open quantum systems using quantum trajectories and automatic differentiation. *Phys. Rev. A*, *99*, 052327. https://doi.org/10.1103/PhysRevA.99.052327

Bezanson, J., Edelman, A., Karpinski, S., & Shah, V. B. (2017). Julia: A fresh approach to numerical computing. *SIAM Rev.*, *59*, 65. https://doi.org/10.1137/141000671

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., & Zhang, Q. (2018). *JAX: Composable transformations of Python+NumPy programs*. GitHub. http://github.com/jax-ml/jax

Brif, C., Chakrabarti, R., & Rabitz, H. (2010). Control of quantum phenomena: Past, present and future. *New J. Phys.*, *12*, 075008. https://doi.org/10.1088/1367-2630/12/7/075008

Brumer, P., & Shapiro, M. (2003). *Principles and applications of the quantum control of molecular processes*. Wiley Interscience.

Dash, B., Goerz, M. H., Duspayev, A., Carrasco, S. C., Malinovsky, V. S., & Raithel, G. (2024). Rotation sensing using tractor atom interferometry. *AVS Quantum Science*, *6*, 014407. https://doi.org/10.1116/5.0175802

Degen, C. L., Reinhard, F., & Cappellaro, P. (2017). Quantum sensing. *Rev. Mod. Phys.*, *89*, 035002. https://doi.org/10.1103/RevModPhys.89.035002

Dowling, J. P., & Milburn, G. J. (2003). Quantum technology: The second quantum revolution. *Phil. Trans. R. Soc. A*, *361*, 1655. https://doi.org/10.1098/rsta.2003.1227

Fouquières, P. de, Schirmer, S. G., Glaser, S. J., & Kuprov, I. (2011). Second order gradient ascent pulse engineering. *J. Magnet. Res.*, *212*, 412. https://doi.org/10.1016/j.jmr.2011.07.023

Goerz, M. H., Carrasco, S. C., & Malinovsky, V. S. (2022). Quantum optimal control via semi-automatic differentiation. *Quantum*, *6*, 871. https://doi.org/10.22331/q-2022-12-07-871

Goerz, M. H., & contributors. (2025a). *QuantumControl.jl: Julia framework for quantum dynamics and control*. GitHub. https://github.com/JuliaQuantumControl/QuantumControl.jl

Goerz, M. H., & contributors. (2025b). *QuantumPropagators.jl: Propagators for quantum dynamics and optimal control*. GitHub. https://github.com/JuliaQuantumControl/

QuantumPropagators.jl

Goerz, M. H., Gualdi, G., Reich, D. M., Koch, C. P., Motzoi, F., Whaley, K. B., Vala, J., Müller, M. M., Montangero, S., & Calarco, T. (2015). Optimizing for an arbitrary perfect entangler. II. Application. *Phys. Rev. A*, *91*, 062307. https://doi.org/10.1103/PhysRevA.91.062307

Goerz, M. H., Halperin, E. J., Aytac, J. M., Koch, C. P., & Whaley, K. B. (2014). Robustness of high-fidelity Rydberg gates with single-site addressability. *Phys. Rev. A*, *90*, 032329. https://doi.org/10.1103/PhysRevA.90.032329

Goerz, M. H., Motzoi, F., Whaley, K. B., & Koch, C. P. (2017). Charting the circuit QED design landscape using optimal control theory. *Npj Quantum Inf*, *3*, 37. https://doi.org/10.1038/s41534-017-0036-0

Goerz, M. H., Reich, D. M., & Koch, C. P. (2014). Optimal control theory for a unitary operation under dissipative evolution. *New J. Phys.*, *16*, 055012. https://doi.org/10.1088/1367-2630/16/5/055012

Goodwin, D. L., & Kuprov, I. (2015). Auxiliary matrix formalism for interaction representation transformations, optimal control, and spin relaxation theories. *J. Chem. Phys.*, *143*, 084113. https://doi.org/10.1063/1.4928978

Goodwin, D. L., & Kuprov, I. (2016). Modified Newton-Raphson GRAPE methods for optimal control of spin systems. *J. Chem. Phys.*, *144*, 204107. https://doi.org/10.1063/1.4949534

Griewank, A., & Walther, A. (2008). *Evaluating derivatives* (Second). Society for Industrial; Applied Mathematics. https://doi.org/10.1137/1.9780898717761

Hogben, H. J., Krzystyniak, M., Charnock, G. T. P., Hore, P. J., & Kuprov, I. (2011). Spinach – a software library for simulation of spin dynamics in large spin systems. *J. Magnet. Res.*, *208*, 179. https://doi.org/10.1016/j.jmr.2010.11.008

Johansson, J. R., Nation, P. D., & Nori, F. (2013). QuTiP 2: A Python framework for the dynamics of open quantum systems. *Comput. Phys. Commun.*, *184*, 1234. https://doi.org/10.1016/j.cpc.2012.11.019

Khaneja, N., Reiss, T., Kehlet, C., Schulte-Herbrüggen, T., & Glaser, S. J. (2005). Optimal control of coupled spin dynamics: Design of NMR pulse sequences by gradient ascent algorithms. *J. Magnet. Res.*, *172*, 296. https://doi.org/10.1016/j.jmr.2004.11.004

Koch, J., Yu, T. M., Gambetta, J., Houck, A. A., Schuster, D. I., Majer, J., Blais, A., Devoret, M. H., Girvin, S. M., & Schoelkopf, R. J. (2007). Charge-insensitive qubit design derived from the Cooper pair box. *Phys. Rev. A*, *76*, 042319. https://doi.org/10.1103/PhysRevA.76.042319

Leung, N., Abdelhafez, M., Koch, J., & Schuster, D. (2017). Speedup for quantum optimal control from automatic differentiation based on graphics processing units. *Phys. Rev. A*, *95*, 042318. https://doi.org/10.1103/PhysRevA.95.042318

Leung, N., Abdelhafez, M., & Schuster, D. (2021). *Quantum-optimal-control*. GitHub. https://github.com/SchusterLab/quantum-optimal-control

Nielsen, M., & Chuang, I. L. (2000). *Quantum computation and quantum information*. Cambridge University Press.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Köpf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., … Chintala, S. (2019). PyTorch: An imperative style, high-performance deep learning library. In H. M. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. A. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems 32* (pp. 8024–8035). Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019. http://papers.neurips.cc/paper/

9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

*QDYN: Fortran 95 library and utilities for quantum dynamics and optimal control*. (2025). https://www.qdyn-library.net

Qi, Y., & contributors. (2022). *LBFGSB: Julia wrapper for L-BFGS-B nonlinear optimization code*. GitHub. https://github.com/Gnimuc/LBFGSB.jl

Rackauckas, C., & Nie, Q. (2017). DifferentialEquations.jl – a performant and feature-rich ecosystem for solving differential equations in Julia. *J. Open Res. Softw.*, *5*. https://doi.org/10.5334/jors.151

Rossignolo, M., Reisser, T., Marshall, A., Rembold, P., Pagano, A., Vetter, P. J., Said, R. S., Müller, M. M., Motzoi, F., Calarco, T., Jelezko, F., & Montangero, S. (2023). QuOCS: The quantum optimal control suite. *Comput. Phys. Commun.*, *291*, 108782. https://doi.org/10.1016/j.cpc.2023.108782

Ryan, C. A., & contributors. (2013). *pulse-finder: Matlab code for GRAPE optimal control in NMR*. GitHub. https://github.com/caryan/pulse-finder/

Sola, I. R., Chang, B. Y., Malinovskaya, S. A., & Malinovsky, V. S. (2018). Quantum control in multilevel systems. *Adv. At. Mol. Opt. Phys.*, *67*, 151. https://doi.org/10.1016/bs.aamop.2018.02.003

Tošner, Z., Vosegaard, T., Kehlet, C., Khaneja, N., Glaser, S. J., & Nielsen, N. Chr. (2009). Optimal control in NMR spectroscopy: Numerical implementation in SIMPSON. *J. Magnet. Res.*, *197*, 120. https://doi.org/10.1016/j.jmr.2008.11.020

Watts, P., Vala, J., Müller, M. M., Calarco, T., Whaley, K. B., Reich, D. M., Goerz, M. H., & Koch, C. P. (2015). Optimizing for an arbitrary perfect entangler: I. Functionals. *Phys. Rev. A*, *91*, 062306. https://doi.org/10.1103/PhysRevA.91.062306

Wittler, N., Roy, F., Pack, K., Werninghaus, M., Roy, A. S., Egger, D. J., Filipp, S., Wilhelm, F. K., & Machnes, S. (2021). Integrated tool set for control, calibration, and characterization of quantum devices applied to superconducting qubits. *Phys. Rev. Applied*, *15*, 034080. https://doi.org/10.1103/physrevapplied.15.034080

Zhang, M., Yu, H.-M., Yuan, H., Wang, X., Demkowicz-Dobrzański, R., & Liu, J. (2022). QuanEstimation: An open-source toolkit for quantum parameter estimation. *Phys. Rev. Research*, *4*, 043057. https://doi.org/10.1103/physrevresearch.4.043057

Zhu, C., Byrd, R. H., Lu, P., & Nocedal, J. (1997). Algorithm 778: L-BFGS-B: Fortran subroutines for large-scale bound-constrained optimization. *ACM Trans. Math. Softw.*, *23*, 550. https://doi.org/10.1145/279232.279236