Generative Neural Reparameterization for Differentiable PDE-Constrained Optimization

Archis S. Joglekar
Ergodic LLC, Seattle, WA
Pasteur Labs, Brooklyn, NY
Institute for Simulation Intelligence, Brooklyn, NY
archis@ergodic.io

Abstract

Partial-differential-equation (PDE)-constrained optimization is a well-worn technique for acquiring optimal parameters of systems governed by PDEs. However, this approach is limited to providing a single set of optimal parameters per optimization. Given a differentiable PDE solver, if the free parameters are reparameterized as the output of a neural network, that neural network can be trained to learn a map from a probability distribution to the distribution of optimal parameters. This proves useful in the case where there are many well performing local minima for the PDE. We apply this technique to train a neural network that generates optimal parameters that minimize laser-plasma instabilities relevant to laser fusion and show that the neural network generates many well performing and diverse minima.

1 Introduction

While Automatic Differentiation (AD) is not new (1) and scientific programmers have exploited implementations of it for some time now, AD tooling has become much more accessible as a byproduct of the deep learning revolution in machine learning. The maturation of AD tooling has led to the paradigm of differentiable programming, where general purpose programs can be written using an AD-capable framework and then benefit from natively supporting machine learning and gradient-descent. At this point, this paradigm has been exploited by many disciplines and it would be disingenuous to attempt to enumerate them here.

Partial-differential-equation (PDE) constrained optimization is pertinent to many scientific and engineering disciplines (2; 3). A key component of PDE-constrained optimization is the method by which the gradient is computed. When a numerical program that solves PDEs is written using an AD-capable framework, it immediately lends itself to PDE constrained optimization.

In this work, we will show that when PDE constrained optimization is performed using an AD-capable framework i.e. when the PDE solver is made differentiable, the benefit is not only access to a fast and accurate gradient with respect to the optimization parameters, but also the ability to learn distributions of optimal parameters via highly-parameterized neural networks. When the neural network is made 'generative', i.e. when it maps from a random vector, it can help capture degeneracy in PDE-constrained optimization problems with many local minima.

2 Generative PDE-constrained optimization

In this section, we explain our method, how it is an extension of previous work, and the advantages it offers in comparison to direct PDE constrained optimization. In the discussion that follows, $Y(T, \vec{x})$

D3S3: Data-driven and Differentiable Simulations, Surrogates, and Solvers @ NeurIPS 2024.

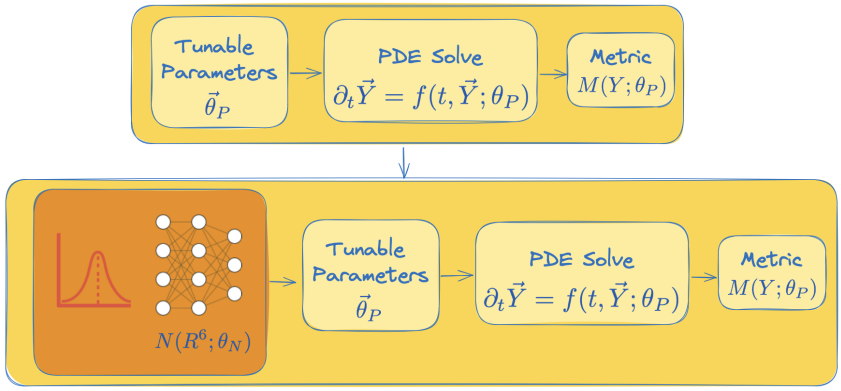

Figure 1: (a) The usual PDE constrained optimization problem where the parameters, θ_P, are optimized to minimize the metric value, M. The required gradient is $\partial M / \partial \theta_P$. (b) Generative neural PDE constrained optimization comprises a neural network that maps a random variable to the optimal parameter set θ_P. The relevant gradient is now $\partial M / \partial \theta_N$.

is the solution to a PDE parameterized by parameter set, θ_P. $\mathcal{M}$ is the function that computes the scalar quantity that is to be minimized. The minimization strategy is first-order gradient descent.

2.1 Direct method

Here, direct optimization refers to solving the optimization problem posed by

$$\min_{\theta_P} \quad \mathcal{L}\left[Y(T, \vec{x}; \theta_P)\right] \tag{1}$$

$$s.t. \quad \partial_t Y = f(t, \vec{x}, Y(t, \vec{x}); \theta_P). \tag{2}$$

This is a typical PDE constrained optimization problem. While effective, performing gradient descent to solve eq. 1 to obtain a distribution of optimal parameters can require many iterations and suffer from the curse of dimensionality because one optimization produces one set of optimal parameters.

2.2 Neural Reparameterization (NR)

In ref. (4), it was proposed that the parameter set be reparameterized using a neural network. For that particular application, the reparameterization enabled the optimization algorithm to uncover better performing minima. Formally, they solved the optimization problem posed by

$$\min_{\vec{g}, \theta_N} \quad \mathcal{L}\left[Y(T, \vec{x}; N(\vec{g}; \theta_N))\right] \tag{3}$$

$$s.t. \quad \partial_t Y = f(t, \vec{x}, Y(t, \vec{x}); N(g; \theta_N)), \tag{4}$$

where $\theta_P = N(\vec{g}; \theta_N)$ is the neural reparameterization (NR) of θ_P. Crucially, in that work, the input vector, $\vec{g}$, to the neural network is a learned quantity in addition to the weights of the neural network θ_N. Like with Direct method above, NR is also unable to obtain multiple local minima without suffering from the same problems.

2.3 Generative Neural Reparameterization (GNR)

Here, we propose that $\vec{g}$ is no longer learned, but is instead a random variable. This results in the neural network being responsible for learning a mapping from a random variable to an optimal set of parameters. Figure 1 provides a conceptual illustration of this method. Formally, the problem is stated by

$$\min_{\theta_N} \quad \mathcal{L}\left[Y(T, \vec{x}; N(G; \theta_N))\right] \tag{5}$$

$$s.t. \quad \partial_t Y = f(t, \vec{x}, Y(t, \vec{x}); N(G; \theta_N)), \tag{6}$$

where G is now a random vector from an N-dimensional unit normal vector where N is another hyperparameter for the neural network. Here, we choose $N = 6$. We find the training process to be relatively insensitive to this parameter within $4 < N < 16$. This addition to the NR method enables learning distributions of optimal parameters. The neural network is responsible for transforming the unit normal probability distribution to the probability distribution of optimal spectra.

If the optimization problem has many satisfactory local minima and many parameters where gradient-descent is necessary for the problem to be tractable, each instantiation of the direct method and NR will converge to a single minimum from the set of satisfactory minima. On the other hand, GNR learns distributions of optimal parameters. In addition to obtaining a generative function for inverse design, learning a distribution of optimal solutions can help with downstream analyses of the behavior of the solution at the optimal point.

In the following section, we apply the GNR method to a PDE-constrained minimization problem where local minima abound.

3 Application - Laser Plasma Instability Modeling

3.1 Background

Laser-driven inertial fusion has demonstrated promising recent results in the laboratory (5; 6) and efforts are underway to scale these experiments into those that can deliver enough power to serve as part of a fusion power plant. One of the persistent challenges with scaling the experiments and delivering large amounts of laser energy to the fusion target is the presence of laser plasma interactions. Plasmas are electromagnetic media which can and do interact with lasers. While laser plasma interactions have been used to make implosions more symmetric by leveraging plasma's ability to transfer energy between laser beams (7), they are typically detrimental in other aspects of the laser plasma physics that occurs during such experiments. For this reason, it is important to model the behavior of, and minimize the impact of, these laser plasma instabilities that plague laser-driven inertial fusion.

Recent advances in laser technology have enabled the development of broadband high energy lasers. Adding bandwidth, i.e. where the laser is not monochromatic but has high energy and intensity light at many wavelengths, helps mitigate laser-plasma instabilities (8; 9; 10). For this reason, future laser fusion experiments and facilities will be designed with bandwidth. It will be important to determine exactly what kind of bandwidth spectra best serve the purpose of transferring laser energy to the target through the plasma. To do so, we will need bandwidth that helps minimize the interaction between the laser and the plasma. In the following section, we detail the PDE and the ensuing laser plasma instability minimization problem.

3.2 Equations

We model one such laser-plasma instability, called Two Plasmon Decay, using the PDE that describes electron plasma waves, also given in ref. (9), given by

$$\nabla\cdot\left[i\left(\frac{\partial}{\partial t}+\nu_e\circ\right)+\frac{3v_{te}^2}{2\omega_{p0}}\nabla^2+\frac{\omega_{p0}}{2}\left(1-\frac{n_b(x)}{n_0}\right)\right]\mathbf{E}_h = S_{\text{TPD}}(E_0)+S_h \tag{7}$$

where the source term, S_{TPD}, contains E_0, the laser field that can be optimized. The equations are described in more detail in ref. (9) and the appendix. It is the laser field that contains the free parameters that can be optimized to minimize laser-plasma instability activity. The laser field is a sum of N_c complex fields and is given by

$$E_0(t,x) = \sum_j^{N_c} a_j \exp(ik_j(x)x - i\omega_j t + \phi_j). \tag{8}$$

Formally, the optimization problem of interest is given by

$$\min_{a_j,\phi_j} \int_{15\text{ps}}^{20\text{ps}} \int\int d_t|\mathbf{E}_h(t,x,y;a_j,\phi_j,N_c=8)|^2\, dx\, dy\, dt, \tag{9}$$

where we are interested in minimizing the growth rate of the instability.

Performing GNR on eq. 9 gives

$$\min_{\theta_N} \quad \int_{15\text{ps}}^{20\text{ps}} \int \int d_t |\mathbf{E}_h(t, x, y; a_j, \phi_j, N_c = 8)|^2 \, dx \, dy \, dt \tag{10}$$

$$s.t. \quad a_j = f(N(G); \theta_N) \text{ and } \phi_j = g(N(G); \theta_N). \tag{11}$$

3.3 Results

In this section, we solve eq. 11[1] and show comparisons of the results of the optimization for $N_c = 8$ using GNR to the baseline case of a uniform amplitude, $a_j = 0.125$, spectrum with random phase $\phi_j = U[0, 2\pi]$, as in ref. (9). An example amplitude and phase pair from the uniform case, along with six instantiations of the generated spectra, are shown in fig. 2.

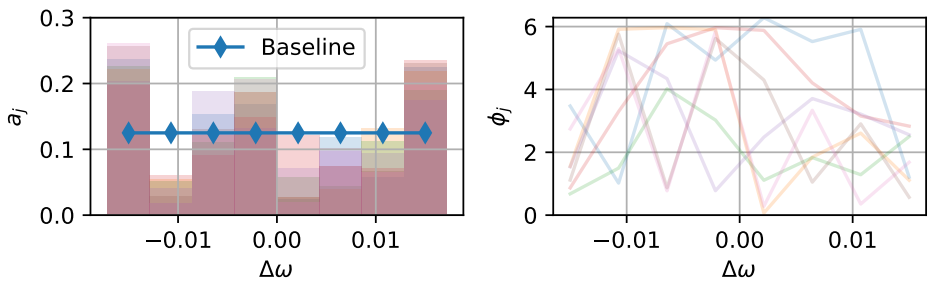

Figure 2: Example spectra from a trained GNR model

Figure 3 shows the comparison of growth rates of the absolute instability from simulations with generated spectra and from simulations using the baseline. The figure indicates that the generated

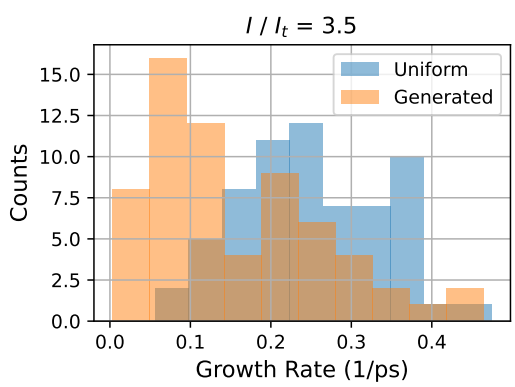

Figure 3: A comparison of the distribution of growth rates from simulations using the baseline uniform-random spectra and the spectra acquired using the GNR method shows a clear improvement over baseline.

distribution generally performs better than the baseline case in task of minimizing the growth rate. The analysis of the results and the underlying physics is beyond the scope of this workshop paper. The aim here is to demonstrate that the GNR method can capture many well performing local minima in PDE-constrained optimization.

[1]Training and solver details are given in the Appendix

4 Conclusions

PDE-constrained optimization is a common technique that can benefit from Automatic Differentiation and Differentiable Programming. In addition to the fast calculation of gradients, the ability to train neural networks in-line opens many possibilities. We highlight one such possibility here where a neural network can be trained to map from a random vector to a set of optimal parameters. This enables machine learning distributions of optimal parameters in the context of PDE-constrained optimization with many desirable local minima.

Acknowledgments and Disclosure of Funding

We acknowledge useful discussions with Dr. Peter Sharpe and Dr. Aidan Crilly.

This material is based upon work supported by the Department of Energy Office of Fusion Energy under Award Number(s) DE-SC0024863. This research used resources of the National Energy Research Scientific Computing Center, a DOE Office of Science User Facility supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231 using NERSC award FES-ERCAP0026741. This work also received support from Pasteur Labs and ISI.

The authors declare no competing interests.

References

[1] A. Griewank and A. Walther, *Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation*. USA: Society for Industrial and Applied Mathematics, second ed., 2008.

[2] L. T. Biegler, O. Ghattas, M. Heinkenschloss, and B. van Bloemen Waanders, "Large-scale pde-constrained optimization: an introduction," in *Large-scale PDE-constrained optimization*, pp. 3–13, Springer, 2003.

[3] J. C. De los Reyes, *Numerical PDE-constrained optimization*. Springer, 2015.

[4] S. Hoyer, J. Sohl-Dickstein, and S. Greydanus, "Neural reparameterization improves structural optimization," Sept. 2019. arXiv:1909.04240 [cs, stat].

[5] Indirect Drive ICF Collaboration, "Lawson Criterion for Ignition Exceeded in an Inertial Fusion Experiment," *Physical Review Letters*, vol. 129, p. 075001, Aug. 2022. Publisher: American Physical Society.

[6] A. B. Zylstra, O. A. Hurricane, D. A. Callahan, A. L. Kritcher, J. E. Ralph, H. F. Robey, J. S. Ross, and C. V. Young, "Burning plasma achieved in inertial fusion," *Nature*, vol. 601, pp. 542–548, Jan. 2022. Number: 7894 Publisher: Nature Publishing Group.

[7] E. L. Dewald, J. L. Milovich, P. Michel, O. L. Landen, J. L. Kline, S. Glenn, O. Jones, D. H. Kalantar, A. Pak, H. F. Robey, G. A. Kyrala, L. Divol, L. R. Benedetti, J. Holder, K. Widmann, A. Moore, M. B. Schneider, T. Döppner, R. Tommasini, D. K. Bradley, P. Bell, B. Ehrlich, C. A. Thomas, M. Shaw, C. Widmayer, D. A. Callahan, N. B. Meezan, R. P. J. Town, A. Hamza, B. Dzenitis, A. Nikroo, K. Moreno, B. Van Wonterghem, A. J. Mackinnon, S. H. Glenzer, B. J. MacGowan, J. D. Kilkenny, M. J. Edwards, L. J. Atherton, and E. I. Moses, "Early-Time Symmetry Tuning in the Presence of Cross-Beam Energy Transfer in ICF Experiments on the National Ignition Facility," *Physical Review Letters*, vol. 111, p. 235001, Dec. 2013. Publisher: American Physical Society.

[8] R. K. Follett, J. G. Shaw, J. F. Myatt, J. P. Palastro, R. W. Short, and D. H. Froula, "Suppressing Two-Plasmon Decay with Laser Frequency Detuning," *Physical Review Letters*, vol. 120, p. 135005, Mar. 2018.

[9] R. K. Follett, J. G. Shaw, J. F. Myatt, C. Dorrer, D. H. Froula, and J. P. Palastro, "Thresholds of absolute instabilities driven by a broadband laser," *Physics of Plasmas*, vol. 26, p. 062111, June 2019.

[10] R. K. Follett, J. G. Shaw, J. F. Myatt, H. Wen, D. H. Froula, and J. P. Palastro, "Thresholds of absolute two-plasmon-decay and stimulated Raman scattering instabilities driven by multiple broadband lasers," *Physics of Plasmas*, vol. 28, p. 032103, Mar. 2021. Publisher: American Institute of Physics.

[11] A. Joglekar and A. Thomas, "Adept - automatic differentiation enabled plasma transport," in *ICML - SynS & ML Workshop (https://syns-ml.github.io/2023/contributions/)*, 2023.

A Solver Details

This solver is available as an option within ADEPT (11), an open-source differentiable simulation tool for plasma physics [2].

A.1 Equations

The PDE for Two Plasmon Decay (9) is given by

$$\nabla\cdot\left[i\left(\frac{\partial}{\partial t}+\nu_e\circ\right)+\frac{3v_{te}^2}{2\omega_{p0}}\nabla^2+\frac{\omega_{p0}}{2}\left(1-\frac{n_b(x)}{n_0}\right)\right]\mathbf{E}_h=S_{\text{TPD}}(E_0)+S_h \tag{12}$$

$$S_{\text{TPD}}\equiv\frac{e}{8\omega_{p0}m_e}\frac{n_b(x)}{n_0}\nabla\cdot[\nabla(\mathbf{E}_0\cdot\mathbf{E}_h^*)-\mathbf{E}_0\nabla\cdot\mathbf{E}_h^*]e^{-i(\omega_0-2\omega_{p0})t} \tag{13}$$

where the first term represents oscillations in the electron plasma, the second represents Landau damping, the third term represents the effect of a plasma density gradient, and the final term is the driving term that contains E_0, the laser field. It is the laser field that contains the free parameters that can be optimized to minimize laser-plasma instability activity. The laser field is given by

$$E_0(t,x)=\sum_j^{N_c}a_j\exp(ik_j(x)x-i\omega_j t+\phi_j) \tag{14}$$

A.2 Parameters

The simulation parameters are given in table 1.

$\Delta x, \Delta y$	20 nm	Δt	2 fs
L_x	140 μm	L_y	16 μm
t_{max}	20 ps	L_n	366 μm
T_e	3.92 keV	I_0	7.53×10^{14} W/cm^2

Table 1: Simulation Parameters, Geometry, and Discretization

B Training

B.1 Neural Network Details

Two separate neural networks were trained, one each for amplitude and phase. Their parameters are given in table 2.

Input	$\mathrm{N}(0,1)^6$	Optimizer	ADAM
Width	128	Learning Rate	0.001
Depth	3	Activation Function	$\tanh$
Output	8		

Table 2: Training details

It is important to note that the output of the neural network responsible for computing the amplitude spectrum was exponentiated. This was done to encourage higher contrast ratios between values. The output of the neural network was constrained to $[-3,3]$ and then exponentiated and normalized to maintain the desired total amplitude.

[2] https://www.github.com/ergodicio/adept