Unsupervised Discovery of Inertial-Fusion Plasma Physics using Differentiable Kinetic Simulations

Anonymous Authors[1]

Abstract

Plasma supports collective modes and particle-wave interactions that leads to complex behavior in inertial fusion energy applications. While plasma can sometimes be modeled as a charged fluid, a kinetic description is useful towards the study of nonlinear effects in the higher dimensional momentum-position phase-space that describes the full complexity of plasma dynamics. We create a differentiable solver for the plasma kinetics 3D partial-differential-equation and introduce a domain-specific objective function. Using this framework, we perform gradient-based optimization of neural networks that provide forcing function parameters to the differentiable solver given a set of initial conditions. We apply this to an inertial-fusion relevant configuration and find that the optimization process exploits a novel physical effect that has previously remained undiscovered.

1. Introduction

1.1. Kinetic Plasma Physics

The study of kinetic plasma physics typically involves various formulations or realizations of the non-linear transport equation for electrons given by

$$\frac{\partial f}{\partial t} + v\frac{\partial f}{\partial x} - E\frac{\partial f}{\partial v} = \left[\frac{\delta f}{\delta t}\right]_{coll}, \tag{1}$$

where $E = \int f\ dv - 1$, and $f = f(t, x, v)$. Equation 1, along with Gauss's Law, is often termed the Vlasov-Poisson-Fokker-Planck (VPFP) equation set. Solving the VPFP set is often analytically intractable, even in one spatial and one velocity dimension (1D-1V). This is because the left-hand-side has a stiff linear transport term, has a non-linear term in $E\partial f/\partial v$, and can sustain wave propagation and other hyperbolic partial-differential-equation (PDE) behavior. Additionally, the right hand side is typically represented by a hyperbolic, advection-diffusion, partial-differential-equation. Making progress on kinetic plasma physics requires computational simulation tools.

Numerical solutions to the 1D-1V VPFP equation set have been applied in research on laser-plasma interactions in the context of inertial fusion plasma-based accelerators (Thomas, 2016), space physics (Chen et al., 2019), fundamental plasma physics (Pezzi et al., 2019), and inertial fusion (Strozzi et al., 2007; Fahlen et al., 2009; Banks et al., 2011)

In this work, we extend previous findings (Fahlen et al., 2009) in the context of inertial fusion using a gradient-based approach towards discovery of new physics.

1.2. Gradient-Based Optimization in Plasma Physics

There have been several recent applications of gradient-descent towards fusion plasma physics. Analytic approaches have resulted in the development of adjoint methods for shape derivatives of functions that depend on magnetohydrodynamics (MHD) equilibria (Antonsen et al., 2019; Paul et al., 2020). These methods have been used to perform optimization of stellarator design (Paul et al., 2021). In other work, by using gradients obtained from analytic (Zhu et al., 2017) and automatic differentiation (AD) (McGreivy et al., 2021), the FOCUS and FOCUSADD codes optimize coil shape. These advances are founded on the concept of performing a sensitivity analysis towards device design.

Our work here is, to our knowledge, the first application of AD towards learning physical relationships and discovering novel phenomena in the VPFP dynamical system. We do this by training neural networks through differentiable simulations that solve the 1D-1V version of eq. 1.

1.3. Differentiable Simulations

Differentiable simulations have been used in a variety of contexts for learning parameters for molecular dynamics (Schoenholz & Cubuk, 2019), for learning differencing stencils in PDEs (Bar-Sinai et al., 2019; Zhuang et al., 2020;

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the International Conference on Machine Learning (ICML). Do not distribute.

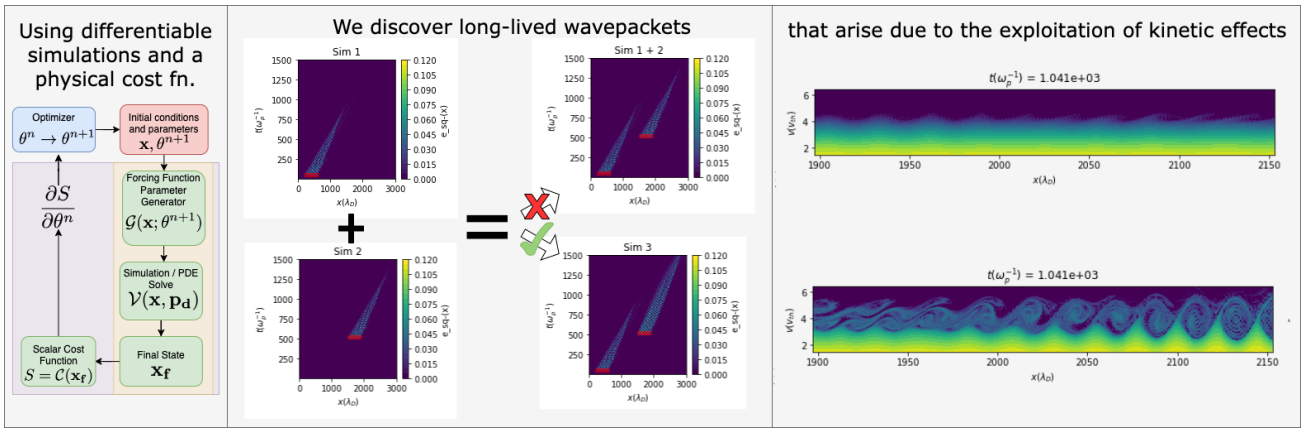


Figure 1. **Left** - We create a differentiable loop to learn forcing function parameters that optimize for a free energy and entropy metric. **Middle** - The optimization uncovers a superadditive process that results in a long-lived second wavepacket. **Right** - Deeper inspection shows that the learning process exploits novel kinetic plasma physics in order to minimize damping, where the second wavepacket interacts with the electrons from the first wavepacket.

Kochkov et al., 2021), and for controlling PDEs (Holl et al., 2020).

In the work here, we train a neural network that provides control parameters to the PDE solver. We choose physical parameters as inputs and control parameters as outputs of the neural network. This enables the neural network to learn a function that describes the physical relationship between the plasma parameters and the forcing function parameters e.g. the resonance frequency.

We train the neural network in an unsupervised fashion using a cost function based on the maximum entropy principle. This enables us to create self-learning plasma physics simulations, where the optimization process provides a physically interpretable function that can enable physics discovery.

2. Physics Discovery using Differentiable Simulations

In this section, we provide a step-by-step description of how a traditional simulation-based computational physics workflow may be modified to perform closed-loop optimization.

2.1. Open Loop - Manual Workflow

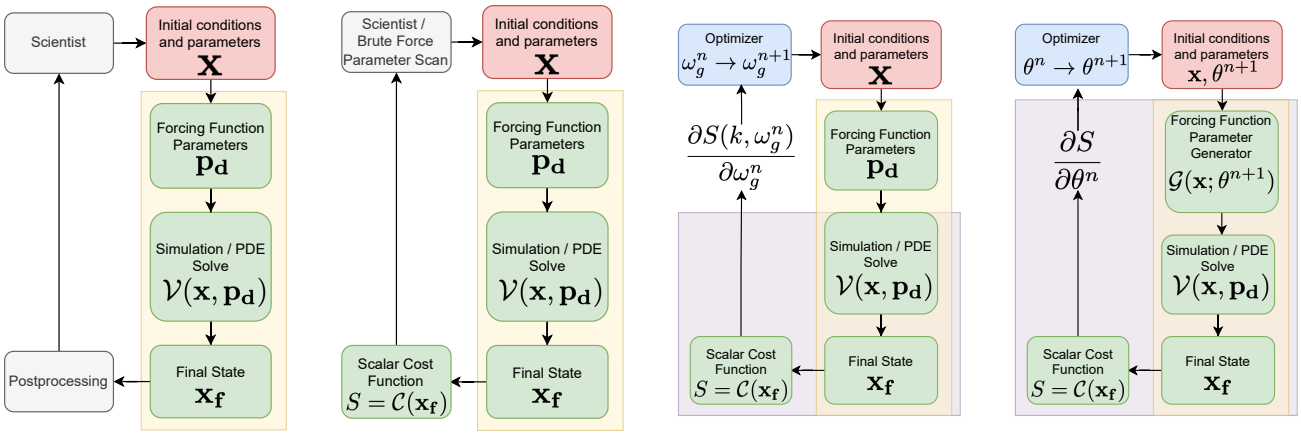
Figure 2a depicts a typical workflow of a computational scientist represented as a cyclic graph. The scientist defines the parametric inputs which create the state vector $\mathbf{x}$. This can contain any parameters that are used to define the simulation e.g. the grid size, the number of solver steps, etc. For didactic purposes, the physical parameters to the simulation may be separated into a different vector of inputs $\mathbf{p_d}$ e.g. the forcing function parameters, the viscosity coefficient etc.

Each of $\mathbf{x}$ and $\mathbf{p_d}$ is passed to the algorithm that solves the PDE which is represented by the function, $\mathcal{V}$. The output of these simulations is stored in the final state vector $\mathbf{x_f}$. The final state is postprocessed using a domain-specific set of algorithms devised by the scientist or otherwise. The results of the postprocessing are interpreted by the scientist who then determines the next set of inputs and parameters.

2.2. Closed Loop - Brute Force Parameter Scan

Figure 2b shows a more automated workflow. We replace the gray-box postprocessing step with the calculation of a scalar quantity S using a Cost Function $\mathcal{C}$ on the final state $\mathbf{x_f}$. This reduces the complexity of the interpretation of the postprocessing and enables a more rapid search in parameter space. The decrease in required human effort for completing one cycle enables the scientist to execute this loop as a brute force parameter scan over a pre-defined parameter space. At the end, the scientist can look up the minimum/maximum of the scalar cost function, and find the parameters which provide that minimum.

The parameter scan approach scales with the number of different unique parameters and the number of values of each parameter. e.g. a 2-D search in x and y requires $N_x \times N_y$ calculations. Therefore, the parameter scan approach quickly becomes inefficient when there are many parameters to scan, or when the required resolution in parameter space is very high. To search this parameter space efficiently, and to escape the linear scaling with each parameter, we can use gradient descent.



(a) Open Loop - Manual Workflow

(b) Closed Loop - Brute Force Parameter Scan

(c) Gradient-Descent-Based Parameter Learning

(d) Gradient-Descent-Based Function Learning

Figure 2. (a) A typical workflow of a computational scientist where the scientist provides the initial conditions and forcing function parameters to a partial-differential-equation (PDE) solve. The output of the solve is stored as the final state $\mathbf{x_f}$. The final state is analyzed by the scientist using domain-specific postprocessing algorithms. (b) A cost function and a parameter scan is introduced which enables a closed-loop search. (c) A gradient-descent based optimization algorithm replaces the parameter-scan to provide a more efficient search mechanism. This requires the components in the purple background to be written in an auto-differentiable framework (d) We add a neural network that generates the forcing function parameters as a function of other parameters. This generalizes the learnings from (c) and enables interpolation and extrapolation within the learned parameters.

2.3. Gradient-Descent-Based Parameter Learning

Figure 2c includes two modifications. The scientist/parameter search graybox has been replaced with a gradient-descent-based optimization algorithm. This algorithm provides the updated parameters, e.g. ω_G, a guess for the resonant frequency of the system, for the next iteration of the loop. The gradient-descent algorithm requires the calculation of an accurate gradient. By writing our PDE solver $\mathcal{V}$ and the cost function $\mathcal{C}$ using a numerical framework that supports automatic differentiation, we are able to calculate $\partial S / \partial \omega_G$. Assuming a well-behaved solution manifold, performing gradient-descent tends to reduce the number of iterations required to find the minimum in comparison to a evenly-spaced parameter scan, especially when the required resolution is unknown (Nocedal & Wright, 2006). Put another way, gradient-descent effectively provides an adaptive stepping mechanism in-lieu of the pre-defined values that represent a parameter scan.

2.4. Gradient-Descent-Based Function Learning

In the final step, we can replace the lookup-like capability of the parameter optimization and choose to learn a blackbox function that can do the same. Through that process, we also gain the ability to interpolate and extrapolate with respect in the input space.

Here, we choose to use neural networks, with a parameter vector θ, to represent the black box function. This allows us to extend the gradient-descent based methodology and leverage existing numerical software to implement this differentiable programming loop.

3. Discovering Novel Physics - Long-lived Wavepackets

3.1. Background

When electrostatic waves are driven to large amplitude, electrons can become trapped in the large potential (O'Neil, 1965). Simulations of Stimulated Raman Scattering (SRS) in inertial confinement fusion (ICF) scenarios show that large-amplitude waves of finite extent are generated in the laser-plasma interaction, and that particle trapping is correlated with the transition to the high-reflectivity burst regime of SRS (Strozzi et al., 2007; Ellis et al., 2012).

Simulating wavepackets, similar to those generated in SRS, but in isolation, has illuminated kinetic dynamics where trapped electrons, with velocity $\approx v_{ph}$, transit the wavepacket which is moving at the slower group velocity v_g. The transit of the trapped electrons from the back of the wavepacket to the front results in the resumption of Landau damping at the back and the wavepacket is then damped away (Fahlen et al., 2009).

Recent work modeled the interaction of multiple speckles with a magnetic field acting as a control parameter. Since the effect of the magnetic field is to rotate the distribution in velocity space, the field strength serves as a parameter by which the authors control scattered particle propagation.

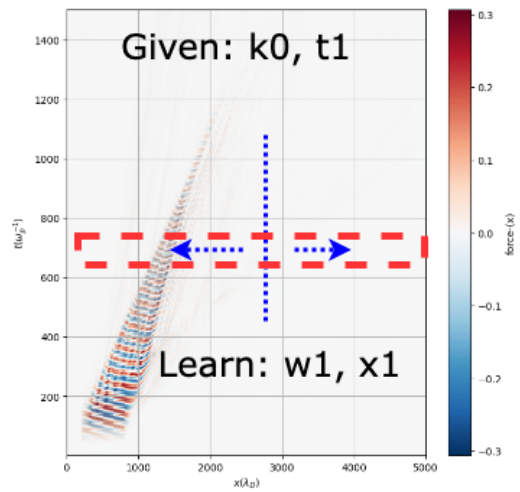


Figure 3. Given a first wavepacket with wavenumber k_0 and a desired time of second wavepacket excitation t_1, the task is to learn functions that give the optimal frequency ω_1 and spatial location x_1 of the second wavepacket.

Using this, along with carefully placed laser speckles, they show that scattered light and particles can serve as the trigger for SRS (Winjum et al., 2019).

Here, we seek the answer to a related question that interrogates the interaction mechanism directly

What is the best way to excite a wave that interacts with the escaped particles?

To answer this question, we reframe this as an optimization problem.

3.2. Problem Setup

We start with a large-amplitude, finite-length electrostatic wavepacket driven by a forcing function with parameters given by

$$\mathbf{p_0} = [x_0, \omega_0, t_0, k_0]\,, \tag{2}$$

where x_i is the location of excitation, ω_i is the frequency, t_i is the time of excitation, and k_i is the wavenumber, of the ith wavepacket.

Since we seek to excite a second wavepacket that can interact with the detrapped electrons, we set the wavenumber $k_1 = k_0$. We reparameterize the resonant frequency, ω_1, with a frequency shift, $\Delta\omega_1$ and the linear resonant frequency ω_0 such that $\omega_1 = \omega_0 + \Delta\omega_1$.

We use the time of excitation of the second wavepacket, t_1, as an independent variable along with k_0. For each t_1 and k_0, we seek to learn functions that produce x_1 and $\Delta\omega_1$ i.e. we seek to learn $x_1(t_1, k_0)$ and $\Delta\omega_1(t_1, k_0)$. The entire parameter vector for the second wavepacket is given by

$$\mathbf{p_0} = [x_1(t_1, k_0), \Delta\omega_1(t_1, k_0), t_1, k_0]\,. \tag{3}$$

This framing is also illustrated in fig. 3 where given k_0 and t_1, we seek functions for ω_1 and x_1.

We reparameterize $\Delta\omega_1$ and x_1 with a neural network with a parameter vector, θ^*, that maximizes the electrostatic energy (minimizes the free energy) and maximizes the kinetic entropy i.e.

$$x_1 = x_1(t_1, k_0; \theta^*), \tag{4}$$
$$\Delta\omega_1 = \Delta\omega_1(t_1, k_0; \theta^*). \tag{5}$$

where,

$$\theta^* = argmin \;\; [-E_{\text{es}}(\mathbf{p}; \theta) - \Delta\mathcal{KE}(\mathbf{p}; \theta)]\,, \tag{6}$$

and

$$E_{\text{es}} = \sum_{t_i}^{t_f} \Delta t \sum_x \Delta x \; E_x^2 \tag{7}$$

$$\Delta\mathcal{KE} = \sum_{t_i}^{t_f} \Delta t \sum_x \Delta x \sum_v \Delta v \; (f \log(f) - f_{\text{MX}} \log(f_{\text{MX}})), \tag{8}$$

are the electrostatic energy, and entropy, terms in the loss function, respectively. $f_{\text{MX}} = f_{\text{MX}}(n, T), n = \int f dv, T = \int f v^2 dv$ where f_{MX} is the Maxwell-Boltzmann distribution.

3.3. Training and Simulation Details

We vary the independent variables such that

$$k_0 \in [0.26, 0.27, ..., 0.32], \tag{9}$$
$$t_1 \in [400, 500, ..., 800], \tag{10}$$

giving an input space of 35 samples from which we seek to learn these functions.

3.3.1. Architecture Details

We use a neural network with 2 hidden linear layers with 8 nodes activated with a ReLU function. The final layer is activated with a $\tanh$ function. The output is normalized such that $p_1 = p_{norm} \times p_{out} + p_{shift}$ where p_{out} is the output from the $\tanh$ function. We normalize the inputs between 0 and 1, and outputs with reasonable windows. For x_1, we allow the entire domain, and $\Delta\omega_1 \in [-0.06, 0.06]$.

We use the ADAM optimizer with a learning rate of 0.05.

3.3.2. Simulation Details

The training simulations are performed with $N_x = 6656, N_v = 160, N_t = 1200, t_{max} = 1100\omega_p^{-1}, x_{max} = 6600\lambda_D$. The amplitude of both drivers is 0.05. The spatial width is $400\lambda_D$ and temporal width is $50\omega_p^{-1}$.

We use a sixth order integrator in time (Casas et al., 2017) with operator splitting. Both operators are computed with exponential integrators with spectral discretizations in phase space. We use a spectral solver for Gauss's Law.

In order to more closely match realistic plasma conditions, we also implement a Fokker-Planck collision operator and use a non-zero collision frequency $\nu_{ee} = 10^{-4}$ for these simulations. More information on the solver is provided in the Appendix.

We implement absorbing boundaries by increasing the collision frequency of the Krook operator in a small localized region at the boundary (Strozzi et al., 2007).

3.4. Training Dynamics

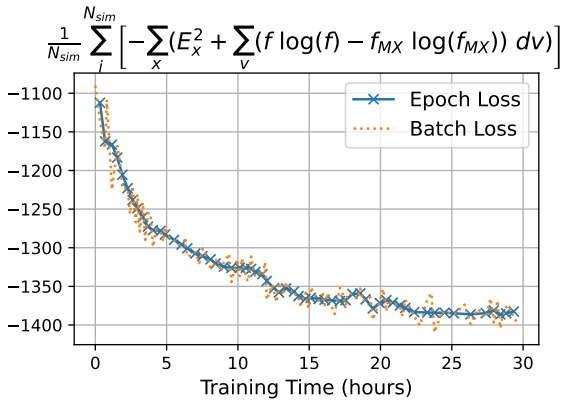


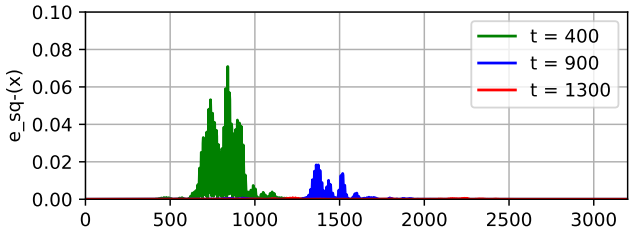
Figure 4. The loss given by eq. 12 is plotted as a function of time. Each cross represents an epoch, and batch-wise fluctuations are also displayed. The training converges after roughly 30 hours

Figure 4 shows that the loss value is reduced over the duration of the training process, roughly 30 hours, which runs for 60 epochs. The convergence in the loss metric suggests that we were able to train a overparameterized neural network with 35 samples of data in 60 epochs. We attribute this to the effect of having so-called 'physical' gradients from training through a PDE solver (Holl et al., 2021).

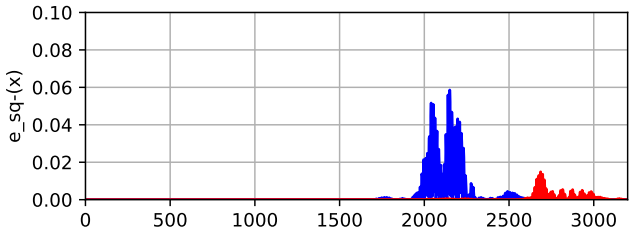
3.5. Long-lived wavepackets

3.5.1. Electric Field

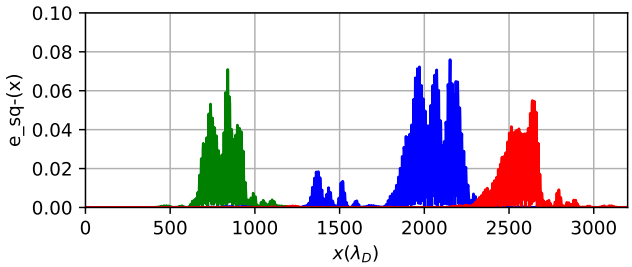
Figure 5 shows the electric field profile for three different simulations. In fig. 5a, only the first wavepacket is excited, and in fig. 5b, only the second wavepacket is excited. In fig. 5c, both wavepackets are excited.



(a) First wavepacket only



(b) Second wavepacket only



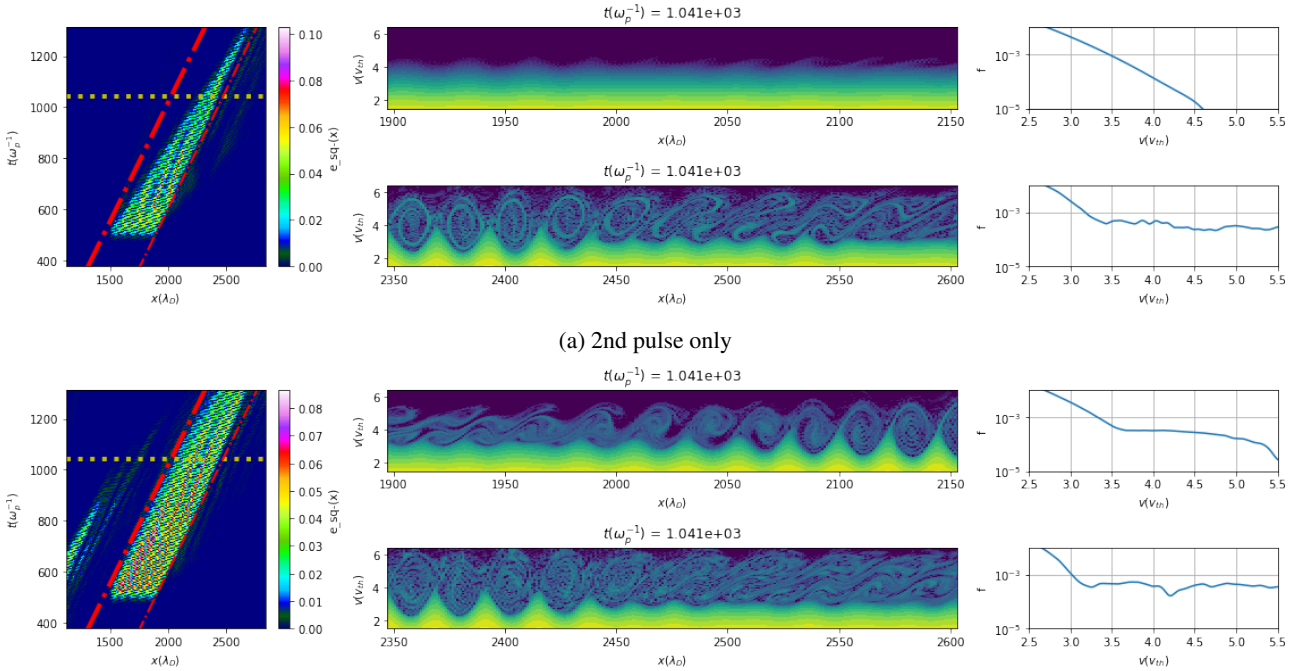
(c) Both wavepackets

Figure 5. Early in time (green), (c) is the same as (a). Later in time, $t = 900\omega_p^{-1}$ (blue), (a) and (c) show very similar magnitudes for the first wavepacket near $x = 1500\lambda_D$ but the second wavepacket excitation is larger in (c) than (b). At $t = 1300\omega_p^{-1}$ (red), it is clear that (c) is not a superposition of (a) and (b) because (b) has damped away, while (c) retains electrostatic energy suggesting the involvement of a superadditive process.

Early in time, $t = 400\omega_p^{-1}$ (green), when only the first wavepacket has been excited figs. 5a and 5c agree perfectly. The second wavepacket is excited at $t = 500\omega_p^{-1}$.

At $t = 900\omega_p^{-1}$, once some time has passed after the excitation of the second wavepacket, the first wavepacket has not fully damped away. It is visible as small bumps in figs. 5a and 5c. The second wavepacket is also present at this time and easily seen in fig. 5b. A larger amplitude wavepacket is seen in fig. 5c.

Late in time, the difference in amplitude between the second wavepacket in figs. 5b and 5c is obvious. The second wavepacket has nearly damped away in fig. 5b. In fig. 5c, the second wavepacket continues to persist, at nearly the same energy as it was at $t = 900\omega_p^{-1}$.



(a) 2nd pulse only

(b) Both pulses

Figure 6. **Left** - Spacetime plot of the electrostatic energy shows the long-lived wavepacket in (b) where the field in (b) survives for a longer duration than in (a). The horizontal line indicates the timestamp of the snapshots in the middle and right. The diagonal dashed-dot lines indicate the spatial location of the snapshots in the middle and right.
Middle Phasespace plots at the back (top) and front (bottom) of the wavepacket. In (b), the phase space shows significant activity at the back of the wavepacket while in (a), the distribution function is nearly undisturbed.
Right The spatially averaged distribution function. This confirms the fact that the distribution function has returned to a Maxwell-Boltzmann at the back of the wavepacket in (a), while in (b), the distribution function remains flat at the phase velocity of the wave. This is the reason behind the loss of damping.

This superadditive behavior, where $f(x)+f(y) \leq f(x+y)$, suggests that there is a constructive interference. We observe this effect for all wavenumbers we model.

3.5.2. Phase Space

To determine the mechanism behind this phenomenon, we turn to the phase-space dynamics. In fig. 6, (i) is a space-time plot of the electric field. The two dashed-dot red lines at the front and back of the wavepacket are parallel and indicate the velocity of the wavefront. In fig. 6a(i), the front of the wavepacket propagates at a seemingly faster rate than the rear. This is due to the etching effect (Fahlen et al., 2009). In fig. 6b, the wave survives for a much longer time, as was also illustrated in fig. 5.

Figure 6 (ii) and (iii) are phase-space plots with their center indicated by the intersection of the horizontal timestamp line, and the dashed-dot line at the rear wavepacket red line at the back of the wavepacket. (iv) and (v) correspond to the intersection with the dashed-dot line at the front. (ii) and (iv) show the phase space within a window in x, while (iii) and (v) are the spatially averaged distribution function. (iii) and (v) serve as a proxy for approximating the propensity of Landau damping in that region.

In fig. 6a(ii) and (iii), we see that the rear of the wavepacket is Maxwellian. As previously shown, this is why the rear of the wavepacket damps faster than the front as in fig. 6a(i) (Fahlen et al., 2009).

In the simulations described here, fig. 6b show that the distribution function at the back of the wavepacket has trapped particle activity (fig. 6b(ii)) and near zero slope at the phase velocity of the wave (fig. 6b(iii)). Both plots show that the slope is negligible because of the arrival of streaming detrapped particles from the first wavepacket. Due to this effect, the remergence of Landau damping that occurs due to the loss of trapped particles in isolated wavepackets no longer occurs here. This results in a reduction of the etching and the wavepacket propagates freely for some time while the particles from the first wavepacket propagate and arrive at the rear of the second wavepacket.

4. Conclusion

In this preprint, we show how one may be able to discover novel physics using differentiable simulations by posing a physical question as an optimization problem. Here, this required domain expertise in determining which functional dependencies to learn using neural networks.

In fig. 2, we show how one may adapt an existing computational science workflow to the autodidactic process described here. In the work performed here, this process enabled the discovery of parameters in a 4D search space with known bounds but an unknown resolution. We trained the model over a coarse grid in k_0 and t_1, and learned functions for x_1 and ω_1. Using gradient descent here allows an escape from the curse of dimensionality and reduces the problem from a 4D search to a 2D search + 2D gradient descent.

In future work, we aim to dive deeper into the discovered phenomenon to better understand the mechanism and derive some scalings. Reduced models of wavepacket dynamics in SRS remain useful towards the development of inertial confinement fusion schemes where laser-plasma instabilities occur.

References

Antonsen, T., Paul, E. J., and Landreman, M. Adjoint approach to calculating shape gradients for three-dimensional magnetic confinement equilibria. *Journal of Plasma Physics*, 85(2), April 2019. ISSN 0022-3778, 1469-7807. doi: 10.1017/S0022377819000254. URL https://www.cambridge.org/core/journals/journal-of-plasma-physics/article/adjoint-approach-to-calculating-shape-gradients-for-three-dimensional-magnetic-confinement-equilibria/ED3C467C28CB98B2B7B036EA356120EB. Publisher: Cambridge University Press.

Banks, J. W., Berger, R. L., Brunner, S., Cohen, B. I., and Hittinger, J. A. F. Two-dimensional Vlasov simulation of electron plasma wave trapping, wavefront bowing, self-focusing, and sideloss. *Physics of Plasmas*, 18(5), 2011. ISSN 1070664X. doi: 10.1063/1.3577784. ISBN: 1070-664X.

Bar-Sinai, Y., Hoyer, S., Hickey, J., and Brenner, M. P. Learning data-driven discretizations for partial differential equations. *Proceedings of the National Academy of Sciences*, 116(31):15344–15349, July 2019. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.1814058116. URL https://www.pnas.org/content/116/31/15344. Publisher: National Academy of Sciences Section: Physical Sciences.

Bellan, P. M. *Fundamentals of Plasma Physics*, volume 173. ISBN 978-3-642-33847-2. doi: 10.2277/0521821169. ISSN: 00029505.

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S., and Zhang, Q. JAX: Autograd and XLA, 2018. URL https://github.com/google/jax. original-date: 2018-10-25T21:25:02Z.

Canosa, J. Numerical solution of Landau's dispersion equation. *Journal of Computational Physics*, 13(1):158–160, September 1973. ISSN 00219991. doi: 10.1016/0021-9991(73)90131-9. URL http://linkinghub.elsevier.com/retrieve/pii/0021999173901319.

Casas, F., Crouseilles, N., Faou, E., and Mehrenberger, M. High-order Hamiltonian splitting for the Vlasov–Poisson equations. *Numerische Mathematik*, 135(3):769–801, 2017. ISSN 0029599X. doi: 10.1007/s00211-016-0816-z. arXiv: 1510.01841 Publisher: Springer Berlin Heidelberg.

Chen, C. H. K., Klein, K. G., and Howes, G. G. Evidence for electron Landau damping in space plasma turbulence. *Nature Communications*, 10(1):740, February 2019. ISSN 2041-1723. doi: 10.1038/s41467-019-08435-3. URL https://www.nature.com/articles/s41467-019-08435-3. Number: 1 Publisher: Nature Publishing Group.

Ellis, I. N., Strozzi, D. J., Winjum, B. J., Tsung, F. S., Grismayer, T., Mori, W. B., Fahlen, J. E., and Williams, E. A. Convective Raman amplification of light pulses causing kinetic inflation in inertial fusion plasmas. *Physics of Plasmas*, 19(11):112704, November 2012. ISSN 1070-664X. doi: 10.1063/1.4762853. URL https://aip.scitation.org/doi/10.1063/1.4762853. Publisher: American Institute of Physics.

Fahlen, J. E., Winjum, B. J., Grismayer, T., and Mori, W. B. Propagation and Damping of Nonlinear Plasma Wave Packets. *Physical Review Letters*, 102(24):245002, June 2009. ISSN 0031-9007, 1079-7114. doi: 10.1103/PhysRevLett.102.245002. URL https://link.aps.org/doi/10.1103/PhysRevLett.102.245002.

Hennigan, T., Cai, T., Norman, T., and Babuschkin, I. Haiku: Sonnet for JAX, 2020. URL https://github.com/deepmind/dm-haiku. original-date: 2020-02-18T07:14:02Z.

Holl, P., Koltun, V., and Thuerey, N. Learning to Control PDEs with Differentiable Physics. *arXiv:2001.07457 [physics, stat]*, January 2020. URL http://arxiv.org/abs/2001.07457. arXiv: 2001.07457.

Holl, P., Koltun, V., and Thuerey, N. Physical Gradients for Deep Learning. *arXiv:2109.15048 [physics]*, October 2021. URL http://arxiv.org/abs/2109.15048. arXiv: 2109.15048.

Joglekar, A. and Levy, M. VlaPy: A Python package for Eulerian Vlasov-Poisson-Fokker-Planck Simulations. *Journal of Open Source Software*, 5(53): 2182, September 2020. ISSN 2475-9066. doi: 10.21105/joss.02182. URL https://joss.theoj.org/papers/10.21105/joss.02182.

Kochkov, D., Smith, J. A., Alieva, A., Wang, Q., Brenner, M. P., and Hoyer, S. Machine learning accelerated computational fluid dynamics. *arXiv:2102.01010 [physics]*, January 2021. URL http://arxiv.org/abs/2102.01010. arXiv: 2102.01010.

McGreivy, N., Hudson, S. R., and Zhu, C. Optimized finite-build stellarator coils using automatic differentiation. *Nuclear Fusion*, 61(2):026020, January 2021. ISSN 0029-5515. doi: 10.1088/1741-4326/abcd76. URL https://doi.org/10.1088/1741-4326/abcd76. Publisher: IOP Publishing.

Nocedal, J. and Wright, S. J. *Numerical Optimization*. Springer, New York, NY, USA, 2e edition, 2006.

O'Neil, T. Collisionless Damping of Nonlinear Plasma Oscillations. *Physics of Fluids*, 8(12):2255, 1965. ISSN 00319171. doi: 10.1063/1.1761193. URL http://scitation.aip.org/content/aip/journal/pof1/8/12/10.1063/1.1761193. arXiv: 1011.1669v3 ISBN: 9788578110796.

Paul, E. J., Antonsen, T., Landreman, M., and Cooper, W. A. Adjoint approach to calculating shape gradients for three-dimensional magnetic confinement equilibria. Part 2. Applications. *Journal of Plasma Physics*, 86 (1), February 2020. ISSN 0022-3778, 1469-7807. doi: 10.1017/S0022377819000916. URL https://www.cambridge.org/core/journals/journal-of-plasma-physics/article/abs/adjoint-approach-to-calculating-shape-gradients-for-threedimensional-magnetic-confineme/0139F38720C83251917799020B5A74CE. Publisher: Cambridge University Press.

Paul, E. J., Landreman, M., and Antonsen, T. Gradient-based optimization of 3D MHD equilibria. *Journal of Plasma Physics*, 87(2), April 2021. ISSN 0022-3778, 1469-7807. doi: 10.1017/S0022377821000283. URL https://www.cambridge.org/core/journals/journal-of-plasma-physics/article/abs/gradientbased-optimization-of-3d-mhd-equilibria/4C3E81B14EB61D5FF2785D5B918E5B65. Publisher: Cambridge University Press.

Pezzi, O., Cozzani, G., Califano, F., Valentini, F., Guarrasi, M., Camporeale, E., Brunetti, G., Retinò, A., and Veltri, P. ViDA: a Vlasov–DArwin solver for plasma physics at electron scales. *Journal of Plasma Physics*, 85(5):905850506, October 2019. ISSN 0022-3778, 1469-7807. doi: 10.1017/S0022377819000631. URL https://www.cambridge.org/core/product/identifier/S0022377819000631/type/journal_article.

Schoenholz, S. S. and Cubuk, E. D. JAX, M.D.: End-to-End Differentiable, Hardware Accelerated, Molecular Dynamics in Pure Python. *arXiv:1912.04232 [cond-mat, physics:physics, stat]*, December 2019. URL http://arxiv.org/abs/1912.04232. arXiv: 1912.04232.

SciPy 1.0 Contributors, Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, , Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., and van Mulbregt, P. SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature Methods*, 17(3):261–272, March 2020. ISSN 1548-7091, 1548-7105. doi: 10.1038/s41592-019-0686-2. URL http://www.nature.com/articles/s41592-019-0686-2.

Strozzi, D. J., Williams, E. A., Langdon, A. B., and Bers, A. Kinetic enhancement of Raman backscatter, and electron acoustic Thomson scatter. *Physics of Plasmas*, 14(1):013104, January 2007. ISSN 1070-664X. doi: 10.1063/1.2431161. URL http://aip.scitation.org/doi/10.1063/1.2431161. arXiv: physics/0610029.

Thomas, A. G. R. Vlasov simulations of thermal plasma waves with relativistic phase velocity in a Lorentz boosted frame. *Physical Review E*, 94(5): 053204, November 2016. doi: 10.1103/PhysRevE.94.053204. URL https://link.aps.org/doi/10.1103/PhysRevE.94.053204. Publisher: American Physical Society.

Winjum, B. J., Tableman, A., Tsung, F. S., and Mori, W. B. Interactions of laser speckles due to kinetic stimulated Raman scattering. *Physics of Plasmas*, 26(11):112701, November 2019. ISSN 1070-664X. doi: 10.1063/1.5110513. URL https://aip.scitation.org/doi/abs/10.1063/1.5110513. Publisher: American Institute of Physics.

Zhu, C., Hudson, S. R., Song, Y., and Wan, Y. New method to design stellarator coils without the winding surface. *Nuclear Fusion*, 58(1):016008, November 2017. ISSN 0029-5515. doi: 10.1088/1741-4326/aa8e0a. URL https://doi.org/10.1088/1741-4326/aa8e0a. Publisher: IOP Publishing.

Zhuang, J., Kochkov, D., Bar-Sinai, Y., Brenner, M. P., and Hoyer, S. Learned discretizations for passive scalar advection in a 2-D turbulent flow. *arXiv:2004.05477 [cond-mat, physics:physics]*, November 2020. URL http://arxiv.org/abs/2004.05477. arXiv: 2004.05477.

A. Simulation and Modeling details

A.1. Simulation Details

We reproduce the solvers and tests implemented in (Joglekar & Levy, 2020) using JAX (Bradbury et al., 2018) and Haiku (Hennigan et al., 2020) to allow the usage of automatic differentiation (AD). The validation tests cover Gauss's Law, Landau damping, and the energy and momentum conservation of the Fokker-Planck implementations when applicable depending on the operator.

A.2. Validating the Differentiable Simulator by Recovering Known Physics

We describe an additional test here which involves recovering the linear, small-amplitude electrostatic resonance using the gradient-based implementation enabled by this AD-capable implementation. This ensures that the gradients given by the AD system are physically reasonable and are representative of physical phenomenon.

A.2.1. Electrostatic Waves

A fundamental wave in plasma physics is the electrostatic wave in unmagnetized plasmas. The dispersion relation is given in numerous textbooks e.g. (Bellan), and reproduced here as

$$1 + \frac{\omega_e^2}{k^2} \int dv \frac{dg(v)/dv}{\omega - kv} = 0, \tag{11}$$

where ω_e is the plasma frequency, k is the wavenumber, ω is the resonant frequency, and v is the independent variable representing the velocity in the integral. $g(v)$ is the distribution function of the plasma particles. This equation has been solved numerically and a lookup table for ω as a function of k is provided in (Canosa, 1973).

We test the capability of our differentiable simulator framework by reproducing those calculations. To do so, we implement the functionality in fig. 2c. We choose a loss function that minimizes free energy and maximizes the entropy given by

$$\mathcal{C}(\mathbf{x_f}) = -\sum_x (E_x^2 + \sum_v (f \ \log(f) - f_{MX} \ \log(f_{MX}))\, dv). \tag{12}$$

Because collisions return the distribution to a Maxwell-Boltzmann distribution, it is not enough to simply maximize $f \log(f)$ if one is interested in retaining non-linear effects. Because of this, the entropy in term quantifies the deviation of the distribution function from a Maxwell-Boltzmann distribution.

In this test, we launch plasma waves using a ponderomotive driver in a box with periodic boundary conditions. For each optimization, the wavenumber k and the box size x_{max} change. We provide a lower bound of 0.5 and an upper bound of 1.5 to the optimization algorithm. We use the L-BFGS algorithm implemented in SciPy (SciPy 1.0 Contributors et al., 2020). We set $f_{\text{tol}} = r_{\text{tol}} = 10^{-12}$. We learn the ω that will minimize the value given by eq. 12 and ensure that it is within 2 decimal places of the previously computed numerical solution of eq. 11 (Canosa, 1973).