Learning cure kinetics of frontal polymerization PDEs using differentiable simulations

Pengfei Cai[1], **Qibang Liu**[2], **Philippe Geubelle**[2], and **Rafael Gomez-Bombarelli**[1]

[1]Massachusetts Institute of Technology, {pengfeic, rafagb}@mit.edu
[2]University of Illinois at Urbana-Champaign, {qibang, geubelle}@illinois.edu

Abstract

Recent advances in frontal ring-opening metathesis polymerization (FROMP) offer a sustainable and energy-efficient alternative for the rapid curing of thermoset polymers compared to conventional bulk curing. To predict FROMP dynamics for different formulations and processing conditions, we require an accurate continuum model. The driving force for FROMP lies in the underlying cure kinetics, but our understanding of the mechanisms is limited and existing cure kinetics models fall short. Herein, we demonstrate that a differentiable simulator for partial differential equations (PDEs) enables learning of cure kinetics functions from video frames of the true solution. With a hybrid PDE solver, where learnable terms are parameterized by orthogonal polynomials or neural networks, we can uncover missing physics within the PDE by applying PDE-constrained optimizations and the adjoint method. Our work paves the way for learning spatiotemporal physics and kinetics from experimentally captured videos.

1 Introduction

1.1 Background

Thermoset polymers are integral in many industries due to their strong specific mechanical properties and thermo-chemical stability. However, current manufacturing of thermosets is energy-inefficient and unsustainable, requiring long curing times at high temperatures in large autoclaves. For instance, about 350 GJ is required to cure a section of Boeing 787's carbon fiber/epoxy fuselage over 8 hours and this process emits more than 80 tons of CO_2 [22]. Recent advancements in frontal ring-opening metathesis polymerization (FROMP) [18, 20] have enabled the rapid and stable curing of thermosets, particularly polydicyclopentadiene (pDCPD). Since the heat of polymerization released by the ring-opening metathesis reaction can propagate further FROMP reactions, only an initial thermal trigger is required. Thus, FROMP can be an energy-efficient and sustainable alternative for manufacturing thermosets at scale.

To discover formulations and processing conditions for stable and rapid FROMP, we need an accurate predictive model at the continuum scale. The dynamics of FROMP can be modeled as thermo-chemical PDEs in terms of the temperature T (in K) and the degree of cure α (dimensionless), as described in Eq. (1).

$$\rho C_p \frac{\partial T}{\partial t} = \kappa \nabla^2 T + \rho H_r \frac{\partial \alpha}{\partial t} \tag{1a}$$

$$\frac{\partial \alpha}{\partial t} = A \exp\left(-\frac{E}{RT}\right) f(\alpha) \tag{1b}$$

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

In the coupled PDE, the reaction term provides the heat source associated with the exothermic reaction and the heat diffusion term describes the heat transport ahead of the advancing polymerization front. The degree of cure α, a phenomenological quantity from 0 (uncured resin) to 1 (fully cured polymer), is the ratio of the amount of heat released to the total heat of polymerization ($\alpha = H/H_r$) [18].

$$\begin{cases} T(x,y,0) = T_0, & \\ \alpha(x,y,0) = \alpha_0, & \\ T(0,y,t) = T_{\text{trig}}, & 0 < t \leq t_{\text{trig}}, \\ \dfrac{\partial T}{\partial x}(0,y,t) = 0, & t > t_{\text{trig}}, \\ -\kappa \nabla T \cdot \mathbf{n} = h_L(T - T_0) \text{ or } 0, & y = \pm \dfrac{w}{2}. \end{cases} \tag{2}$$

The initial conditions and boundary conditions (BCs) are described in Eq. (2), where a trigger temperature T_{trig} is applied on one end $x = 0$ over t_{trig} (Dirichlet BC), followed by an adiabatic BC. Adiabatic or heat convection BCs (with heat transfer coefficient h_L) are imposed on other boundaries depending on problem settings, where w is the width of the domain.

1.2 Learning the complete PDE: Hybrid model with a differentiable PDE solver

In FROMP, the reaction kinetics coupled with different processing conditions are complex and not well understood. Existing cure kinetics functions $f(\alpha)$ are defined explicitly and parameters are obtained by nonlinear fitting of differential scanning calorimetry (DSC) curves, where experiments are performed at a controlled and low heating rate (typically $dT/dt \leq 20$ ^{o}C/min). However, the heating rate at the polymerization front is up to $\sim 10^5$ ^{o}C/min in FROMP - thus existing $f(\alpha)$ functions fitted from DSCs would not robustly predict the FROMP dynamics for different initial conditions and chemical formulations. To address this, we aim to learn unknown physics or kinetics by augmenting the existing PDE with learnable terms and learning the dynamics from experimentally observed spatiotemporal data.

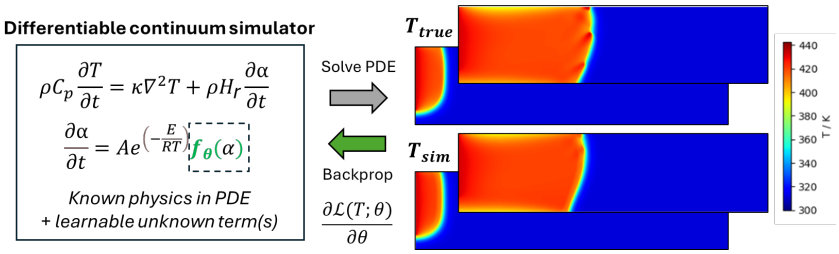

Figure 1: Differentiable hybrid PDE solver to learn unknown term(s) within the PDE

Herein, we adopt a hybrid solver approach where known terms form the base PDE and we focus on learning unknown physics from data. We demonstrate that we can recover the unknown physics from simulation videos by applying PDE-constrained optimization [26, 27] using a differentiable PDE simulator and differentiable programming in JAX [1]. The framework developed will eventually be useful to learn unknown physics and cure kinetics from thermal capture videos obtained experimentally, thus this work paves the way towards that goal. We need a PDE simulator that is end-to-end differentiable so we can update learnable terms within the PDE using gradient descent. The gradients of the loss function (between the true observed dynamics and the simulated PDE solutions) with respect to the parameters are backpropagated through the simulation time steps by solving the adjoint equations. For the numerical method, we choose the finite element method (FEM) [15, 25] due to its versatility and flexibility over different geometries. Unknown terms can be represented as neural networks or orthogonal bases, such as Legendre polynomials. For more details, refer to Appendix A.2.

2 Results and Discussion

We first demonstrate that the differentiable simulator can be used for the control and learning of parameters within the PDE (Appendix A.3).

2.1 Learning cure kinetics models

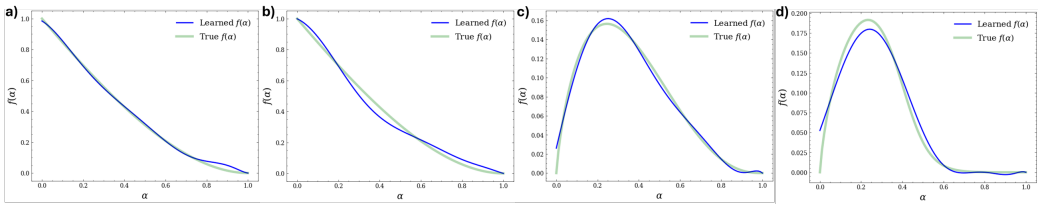

Figure 2: Learned cure kinetics functions $f(\alpha)$ for DCPD-GC1 by learning from a) degree of cure or b) temperatures. Learned $f(\alpha)$ for the FROMP of c) COD and d) DCPD-GC2 by learning from temperatures.

Eventually, we aim to learn unknown physics from experimental videos of FROMP. To this end, we demonstrate the ability to learn unknown cure kinetics functions from simulated FROMP videos (i.e. numerical PDE solutions). To test the robustness of the framework, we will examine different FROMP formulations, namely the polymerization of DCPD with Grubbs catalyst type 1 (DCPD-GC1), DCPD with Grubbs catalyst type 2 (DCPD-GC2) and the unstable FROMP of cyclooctadiene (COD) [14].

$$f_\theta(\boldsymbol{\alpha}) = (1-\alpha)\sum_{n=0}^{N=10} b_n P_n \tag{3}$$

$$\mathcal{L}(T_{\text{sim}}, T_{\text{true}}) = \sum_{i=1}^{N_t} \frac{\|T_{\text{sim},i} - T_{\text{true},i}\|_2}{\|T_{\text{true},i}\|_2} \tag{4}$$

For all cases, we parameterize the $f(\alpha)$ term with a linear combination of orthogonal polynomials, specifically with the first ten Legendre polynomials (Eq. (3)), and learn the eleven b_n coefficients that weigh each polynomial's contribution (including the 0-th order term) [27]. We choose orthogonal Legendre polynomials due to their high expressivity and interpretability while having less tunable parameters. We impose a prior of $(1-\alpha)$ since the reaction rate is 0 when the resin is fully cured. For the loss function (Eq. (4)), we use the relative L_2-norm between the simulated solutions (predictions) and the true solutions, summed over N_t frames. During training, the gradients of the loss with respect to the learnable parameters are backpropagated to update f iteratively via gradient descent.

Learning $f(\alpha)$ for the stable FROMP of DCPD-GC1. In the first case, we attempt to learn $f(\alpha)$ for the DCPD-GC1 system using 21 training samples. The true $f(\alpha)$ (Eq. (5)) is the n^{th}-order Prout-Tompkins model (PTn) [9] and this was used to solve the PDE to obtain the true solutions.

$$f(\alpha) = (1-\alpha)^{1.927}(1+0.365\alpha) \tag{5}$$

The Legendre polynomial parameters are initiated to be zeros and the PDE is solved forward for 100 steps (dt = 0.01). With a batch size of 3, the average gradients in each batch are backpropagated with respect to the 11 parameters to update them every batch. For this scheme, we learned from either degree of cure (Fig. 2a) or temperature (Fig. 2b). Evidently, it is easier to reproduce the true $f(\alpha)$ when we learn directly from the degree of cure as the cure kinetics function is a function of α. Our problem is a coupled time-dependent PDE where both temperature and degree of cure influence each solution in the next time step. In practice, we can only measure the spatiotemporal changes in temperature but not the degree of cure. As such, it is more meaningful to learn the $f(\alpha)$ from the sample's temperatures and solve for the degree of cure using the learned $f(\alpha)$ in each iteration.

Learning $f(\alpha)$ for FROMP of COD and DCPD-GC2. In the next two cases, we learn the $f(\alpha)$ from $T(x,y,t)$ and demonstrate that with only 3 solutions for training, we can learn a good approximation of the $f(\alpha)$ over 50 epochs. Instead of mini-batching and updating with average gradients, we solve the PDE forward and update the parameters over each sample. COD polymerizes with an unstable FROMP profile with certain initial temperatures and pre-cure despite having a relatively simpler cure kinetics following the Prout-Tompkins (PT) model (Eq. (6)).

$$f(\alpha) = (1-\alpha)^{2.514}\alpha^{0.817} \tag{6}$$

We generate 3 solutions with chaotic fronts and attempt to learn the $f(\alpha)$ with only the first 5 frames of the PDE solution (Fig. 2c). Thus, with a total of only 15 frames of $T(x, y, t)$ solution, we can recover the $f(\alpha)$. This is possible because the $(1-\alpha)$ prior enforces $f(\alpha)$ to be 0 when α is 1, and the 3 training samples have 3 different initial conditions of α that give a sufficient range of trajectories for learning.

$$f(\alpha) = (1-\alpha)^{1.72}\alpha^{0.77}e^{-14.48(\alpha-0.41)} \tag{7}$$

Finally, we try to learn a more complicated $f(\alpha)$ involving the Prout-Tompkins model with a diffusion term (Eq. (7)), this model accounts for diffusion effects at higher temperatures [9]. For this case, we find that solving the PDE over longer trajectories (50 steps) is required for a more accurate recovery of $f(\alpha)$ (Fig. 2d).

For the experiments discussed, the trajectories of the learned $f(\alpha)$ over iterations and learning curves are in Appendix A.4. Using the learned $f(\alpha)$, we solve the PDE and plot the roll-out solutions for both T and α (Fig. 6, Fig. 7, Fig. 8) for a few test samples.

2.2 Comparing function representations

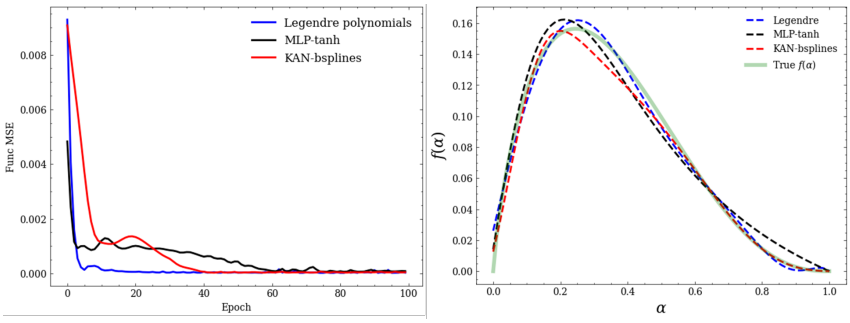

Figure 3: Comparison of validation loss curves and learned $f(\alpha)$ functions parameterized by Legendre polynomials, MLP, and KAN on the unstable COD FROMP case.

We compare parameterizations of $f(\alpha)$ using Legendre polynomials, MLPs, and Kolmogorov-Arnold Networks (KANs) [13] to learn the PT model (Eq. (6)) for COD (Fig. 3). MLPs with appropriate activation functions $(tanh)$ can converge, but are harder to train than orthogonal polynomials. Orthogonal bases like Legendre polynomials work well for smaller datasets due to fewer tunable parameters and better interpretability. With MLPs, inductive biases have to be enforced through the right selection of activation functions to accurately capture the physical dynamics. Nevertheless, neural networks would be more useful when we have larger datasets. In our example, KANs have shown promise in learning unknown terms in PDEs. Herein, we adopted a JAX re-implementation, based on b-splines as the underlying basis functions, of KANs. For future work, it could be meaningful to examine the effectiveness of KANs, particularly with other bases such as Legendre and Chebyshev polynomials, in learning PDEs.

3 Conclusion

With a hybrid differentiable PDE simulator, we have demonstrated that we can learn missing functions within the PDE by applying gradient-based PDE-constrained optimizations. We applied the approach to learn the cure kinetics models for three different FROMP systems - DCPD-GC1, DCPD-GC2, and the unstable FROMP of COD. With limited training samples and a few frames of PDE solutions, we can recover the true $f(\alpha)$ by iteratively updating the learnable function with gradient descent. Parameterizing the unknown functions with orthogonal polynomials give high accuracy and interpretability. Our work paves the way to uncover missing physics and cure kinetics from videos captured experimentally - thereby allowing end-to-end learning of continuum models from observed dynamics.

4 Acknowledgements

This work was supported as part of the Regenerative Energy-Efficient Manufacturing of Thermoset Polymeric Materials (REMAT), an Energy Frontier Research Center funded by the U.S. Department of Energy, Office of Science, Basic Energy Sciences under award DE-SC0023457. The authors thank the valuable discussions with Alex Cohen and Dr Ignacio Arretche.

References

[1] 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: composable transformations of Python+NumPy programs, 2018. URL http://github.com/google/jax.

[2] Brandstetter, J., Worrall, D., and Welling, M. Message Passing Neural PDE Solvers, March 2022. URL http://arxiv.org/abs/2202.03376. arXiv:2202.03376 [cs, math].

[3] Cao, Q., Goswami, S., and Karniadakis, G. E. LNO: Laplace Neural Operator for Solving Differential Equations, May 2023. URL http://arxiv.org/abs/2303.10528. arXiv:2303.10528 [cs].

[4] Chen, R. T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. Neural Ordinary Differential Equations, December 2019. URL http://arxiv.org/abs/1806.07366.

[5] Crilly, A. J., Duhig, B., and Bouziani, N. Learning Closure Relations using Differentiable Programming: An Example in Radiation Transport.

[6] Kidger, P. On Neural Differential Equations, February 2022. URL http://arxiv.org/abs/2202.02435. arXiv:2202.02435 [cs, math, stat].

[7] Kochkov, D., Yuval, J., Langmore, I., Norgaard, P., Smith, J., Mooers, G., Klöwer, M., Lottes, J., Rasp, S., Düben, P., Hatfield, S., Battaglia, P., Sanchez-Gonzalez, A., Willson, M., Brenner, M. P., and Hoyer, S. Neural General Circulation Models for Weather and Climate, March 2024. URL http://arxiv.org/abs/2311.07222. arXiv:2311.07222 [physics].

[8] Kovachki, N., Li, Z., Liu, B., Azizzadenesheli, K., Bhattacharya, K., Stuart, A., and Anandkumar, A. Neural Operator: Learning Maps Between Function Spaces, April 2023. URL http://arxiv.org/abs/2108.08481. arXiv:2108.08481 [cs, math].

[9] Kumar, A., Gao, Y., and Geubelle, P. H. Analytical estimates of front velocity in the frontal polymerization of thermoset polymers and composites. *Journal of Polymer Science*, 59(11):1109–1118, 2021. doi: 10.1002/pol.20210155.

[10] Li, Z., Kovachki, N., Azizzadenesheli, K., Liu, B., Bhattacharya, K., Stuart, A., and Anandkumar, A. Fourier neural operator for parametric partial differential equations, 2021.

[11] Liu, Q., Abueidda, D., Vyas, S., Gao, Y., Koric, S., and Geubelle, P. H. Adaptive Data-Driven Deep-Learning Surrogate Model for Frontal Polymerization in Dicyclopentadiene. *The Journal of Physical Chemistry B*, 128(5):1220–1230, February 2024. ISSN 1520-6106. doi: 10.1021/acs.jpcb.3c07714. URL https://doi.org/10.1021/acs.jpcb.3c07714. Publisher: American Chemical Society.

[12] Liu, Z., Wu, Y., Huang, D. Z., Zhang, H., Qian, X., and Song, S. SPFNO: Spectral operator learning for PDEs with Dirichlet and Neumann boundary conditions, December 2023. URL http://arxiv.org/abs/2312.06980. arXiv:2312.06980 [cs, math].

[13] Liu, Z., Wang, Y., Vaidya, S., Ruehle, F., Halverson, J., Soljačić, M., Hou, T. Y., and Tegmark, M. KAN: Kolmogorov-Arnold Networks, April 2024. URL http://arxiv.org/abs/2404.19756. arXiv:2404.19756 [cond-mat, stat].

[14] Lloyd, E. M., Feinberg, E. C., Gao, Y., Peterson, S. R., Soman, B., Hemmer, J., Dean, L. M., Wu, Q., Geubelle, P. H., Sottos, N. R., and Moore, J. S. Spontaneous Patterning during Frontal Polymerization. *ACS Central Science*, 7(4):603–612, April 2021. ISSN 2374-7943, 2374-7951. doi: 10.1021/acscentsci.1c00110.

[15] Mitusch, S. K., Funke, S. W., and Dokken, J. S. dolfin-adjoint 2018.1: automated adjoints for fenics and firedrake. *Journal of Open Source Software*, 4(38):1292, 2019. doi: 10.21105/joss.01292.

[16] Pathak, J., Subramanian, S., Harrington, P., Raja, S., Chattopadhyay, A., Mardani, M., Kurth, T., Hall, D., Li, Z., Azizzadenesheli, K., Hassanzadeh, P., Kashinath, K., and Anandkumar, A. FourCastNet: A Global Data-driven High-resolution Weather Model using Adaptive Fourier Neural Operators, February 2022. URL http://arxiv.org/abs/2202.11214. arXiv:2202.11214 [physics].

[17] Raissi, M., Perdikaris, P., and Karniadakis, G. E. Physics informed deep learning (part i): Data-driven solutions of nonlinear partial differential equations, 2017.

[18] Robertson, I. D., Yourdkhani, M., Centellas, P. J., Aw, J. E., Ivanoff, D. G., Goli, E., Lloyd, E. M., Dean, L. M., Sottos, N. R., Geubelle, P. H., Moore, J. S., and White, S. R. Rapid energy-efficient manufacturing of polymers and composites via frontal polymerization. *Nature*, 557(7704):223–227, May 2018. doi: 10.1038/s41586-018-0054-x.

[19] Sadr, M., Tohme, T., and Youcef-Toumi, K. Data-Driven Discovery of PDEs via the Adjoint Method, January 2024. URL `http://arxiv.org/abs/2401.17177`. arXiv:2401.17177 [cs, math].

[20] Suslick, B. A., Hemmer, J., Groce, B. R., Stawiasz, K. J., Geubelle, P. H., Malucelli, G., Mariani, A., Moore, J. S., Pojman, J. A., and Sottos, N. R. Frontal polymerizations: From chemical perspectives to macroscopic properties and applications. *Chemical Reviews*, 123(6):3237–3298, 2023. doi: 10.1021/acs.chemrev.2c00686. PMID: 36827528.

[21] Takamoto, M., Praditia, T., Leiteritz, R., MacKinlay, D., Alesiani, F., Pflüger, D., and Niepert, M. Pdebench: An extensive benchmark for scientific machine learning, 2023.

[22] Timmis, A., Hodzic, A., Koh, L., Bonner, M., Soutis, C., Schafer, A., and Dray, L. Environmental impact assessment of aviation emission reduction through the implementation of composite materials. *The International Journal of Life Cycle Assessment*, 20:233–243, 02 2014. doi: 10.1007/s11367-014-0824-0.

[23] Xiao, X., Cao, D., Yang, R., Gupta, G., Liu, G., Yin, C., Balan, R., and Bogdan, P. Coupled multiwavelet neural operator learning for coupled partial differential equations, 2023.

[24] Xue, T., Liao, S., Gan, Z., Park, C., Xie, X., Liu, W. K., and Cao, J. JAX-FEM: A differentiable GPU-accelerated 3D finite element solver for automatic inverse design and mechanistic data science. *Computer Physics Communications*, 291:108802, October 2023. ISSN 0010-4655. doi: 10.1016/j.cpc.2023.108802. URL `https://www.sciencedirect.com/science/article/pii/S0010465523001479`.

[25] Yashchuk, I. Bringing PDEs to JAX with forward and reverse modes automatic differentiation. 2023.

[26] Zhao, H., Storey, B. D., Braatz, R. D., and Bazant, M. Z. Learning the Physics of Pattern Formation from Images. *Physical Review Letters*, 124(6):060201, February 2020. doi: 10.1103/PhysRevLett.124.060201.

[27] Zhao, H., Deng, H. D., Cohen, A. E., Lim, J., Li, Y., Fraggedakis, D., Jiang, B., Storey, B. D., Chueh, W. C., Braatz, R. D., and Bazant, M. Z. Learning heterogeneous reaction kinetics from X-ray videos pixel by pixel. *Nature*, 621(7978):289–294, September 2023. doi: 10.1038/s41586-023-06393-x.

A Appendix

A.1 Related works in scientific machine learning and neural PDEs

Neural PDE models aim to learn a data-driven PDE solver that can predict solutions for each time step autoregressively [2]. Notably, neural operators learn the mapping between function spaces [8]. Some examples include the Fourier [10], Laplace [3], Wavelet [23], and spectral neural operators [12]. Various NN and neural operator models have found successes in climate and weather forecasting where the models are trained on a large corpus of historic data [16]. However, for applications as surrogate PDE solvers, neural operators are usually trained on PDE solutions that are generated by a numerical solver [21]. Another direction involves Physics-Informed Neural Networks (PINNs) [17, 11] where NNs parameterize the underlying PDE solutions and incorporate the equations of the PDE to construct the loss function (i.e. with PDE residual, boundary conditions, initial conditions terms). In our case, we do not know the complete physics of the underlying PDE - our goal is to learn the PDE rather than to learn the solution or its operator.

Hybrid physics machine learning methods combine numerical methods with data-driven methods and these could be useful in multiscale closure modeling. This emerging direction has been applied to learn closure relations in PDEs [5], hybrid general circulation model of the atmosphere [7], and to learn kinetics of Lithium intercalation and pattern formations [26, 27]. Unknown physics within the differential equations, either due to unknown complex relationships or governed by higher-order behaviors not captured by the existing model assumptions, may be parameterized by learnable functions to recover the true physics. At the core of these methods is an end-to-end differentiable simulator enabled by the growing ecosystem of differentiable programming [7], differentiable PDE solvers [24], and the field of neural differential equations [4, 6]. In our work, we build on top of FEniCS-adjoint and JAX-FEniCS [15, 25], where FEM is the underlying PDE solver and the interface with JAX provides differentiable programming and neural networks support.

A.2 Methods

A.2.1 Numerical simulations with the Finite Element Method

To generate all the true numerical solutions examined in this paper, we solved the PDE using the finite element method implemented in FEniCS. The continuous Galerkin elements from the Lagrange family of function spaces are used to approximate the physical fields. An interval mesh and a rectangular mesh were used for 1D and 2D problems. For 2D problems, convective heat loss is applied on $y = \pm\frac{w}{2}$, where w is the width of the domain. For all examples, a Dirichlet boundary condition is imposed on one end of the domain $(x = 0)$ with $T = T_{trig}$.

For examples involving the optimization of parameters and learning cure kinetics of DCPD-GC1, the PDEs of Eq. (1) are framed as a coupled scheme that solves T and α simultaneously using a nonlinear solver following Newton's method and the iterative linear solver of the generalized minimal residual (GMRES) method with an algebraic multigrid (AMG) preconditioner.

Due to convergence issues using the coupled scheme, for examples involving the learning of cure kinetics of DCPD-GC2 and unstable COD FROMP, Eq. (1) is solved in a decoupled scheme, in which an iterative linear solver with the conjugate gradient method and an AMG preconditioner are used to solve the diffusion PDE (Eq. (1a)) for T in an implicit Euler scheme, and an explicit Euler scheme is used to solve the reaction ordinary differential equation (ODE, Eq. (1b)) for α. The material properties shown in Eq. (1) for thermal conductivity κ (in W m^{-1} K^{-1}), specific heat capacity C_p (J kg^{-1} K^{-1}), density ρ (kg m^{-3}), enthalpy of polymerization H_r (J kg^{-1}), pre-exponent A (in s^{-1}) and activation energy E (in kJ mol^{-1}) for DCPD-GC1, DCPD-GC2 and COD can be found in [9].

A.2.2 Differentiable PDE solver

A differentiable PDE solver is required to allow end-to-end learning of unknown terms in the PDE. We apply PDE-constrained optimizations with the adjoint method [19, 26], and backpropagate the loss to update learnable parameters within the PDE. With reverse-mode auto-differentiation, the vector-Jacobian product functions solve the adjoint equations in FEniCS-adjoint [15]. In our time-dependent PDE, solving the PDE forward evaluates T (Eq. (8)) while the adjoint equation (Eq. (9)) is solved backward in time. The adjoint variable, λ, is the solution of the adjoint equation and θ is a set

of parameters that we are trying to learn. A loss function (or objective function) which depends on the state variable (i.e. temperature T) integrated over time ($L(T) = \int_0^t l(T)\, dt$), is constructed based on the relative L_2 norm between the true T and simulated T. Eq. (10) shows the gradient of the loss function with respect to the parameters where $\partial T/\partial \theta$ is the sensitivity.

$$M\frac{dT}{dt} = F(T, \theta) \tag{8}$$

$$-M^{\dagger}\frac{d\lambda}{dt} = \left(\frac{\partial F}{\partial T}\right)^{\dagger}\lambda + \frac{\partial l}{\partial T} \tag{9}$$

$$\frac{\partial L}{\partial \theta} = \int_0^t \left(\frac{\partial l}{\partial T}\right)^{\dagger}\frac{\partial T}{\partial \theta}dt \tag{10}$$

By interfacing with JAX [1], we can create end-to-end differentiable simulators and parameterize learnable functions within the PDE with versatile representations, such as orthogonal polynomials and neural networks. The ADAM optimizer is used to update parameters iteratively through gradient descent with weight decay. An L_2 regularization is imposed in the loss function to discourage larger values of the parameters.

A.3 Optimizing and learning parameters within the PDE

Optimizing material parameters towards high frontal velocities. We apply our approach to optimize material parameters that would steer the PDE solutions toward high frontal velocities. We solve the 1D problem of the PDE forward and calculate the frontal velocities (V_f) using the relative positions of the front with α=0.5. By setting a target V_f in the loss function, we optimize the material parameters within the PDE to control the PDE solutions to reach a high V_f by taking the gradients with respect to each parameter. Specifically, we optimize for the thermal conductivity (κ), specific heat capacity (C_p), and the enthalpy of polymerization (H_r). Intuitively, from the PDE, we know that a high κ, low C_p, and high H_r would lead to high V_f. With this toy problem, we demonstrate that known parameters within the PDE can be optimized (Fig. 4) to give high V_f, thus reproducing our physical intuition built within the PDE.

Learning initial conditions and thermal conductivity. Similarly, we can recover parameters and initial conditions with the same approach. We generate a 2D solution and use only the first 10 time steps (first 0.1s) for learning. Initializing the κ term and the initial temperature T_0 as zeros, we solve the PDE forward for 10 steps and backpropagate the loss (mean squared error for all 10 frames) to update both κ and T_0. With 300 iterations, the parameters converged to recover κ and T_0 as 0.1523 and 24.90, which are close to the true values of 0.152 W/m · K and 25.0 °C respectively (Fig. 4).

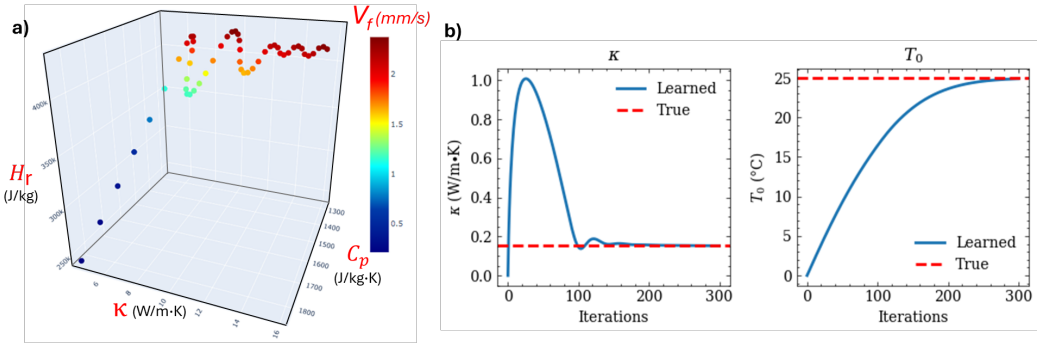

Figure 4: a) Optimizing material parameters for high FROMP frontal velocities. b) Learning thermal conductivity and initial temperature.

A.4 Supplementary figures for learning $f(\alpha)$ and roll-out solutions

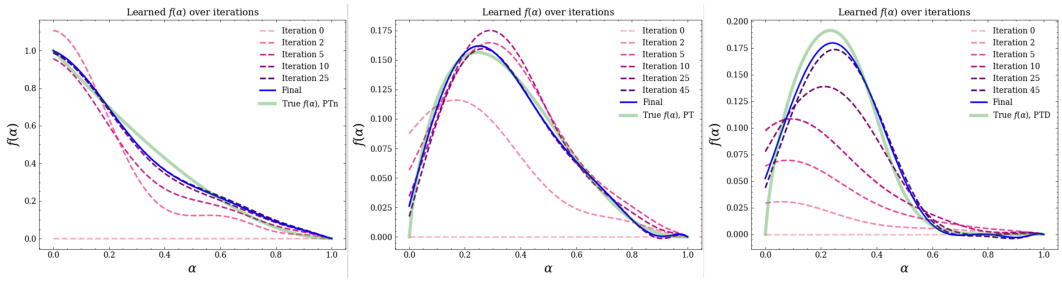

Figure 5: Evolution of learned $f(\alpha)$ over training iterations for DCPD-GC1, COD, and DCPD-GC2.

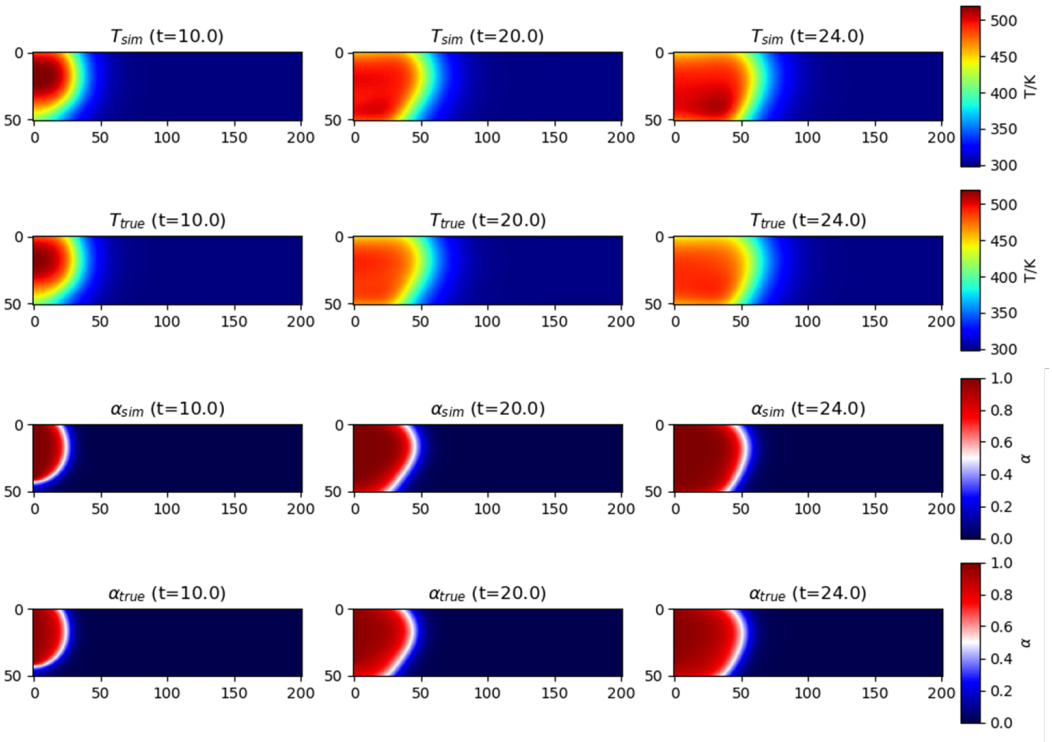

Figure 6: Comparisons of roll-out solutions with the learned $f(\alpha)$ and the true solutions for FROMP of DCPD-GC1. Both T and α solutions are plotted at different t (Test set example).

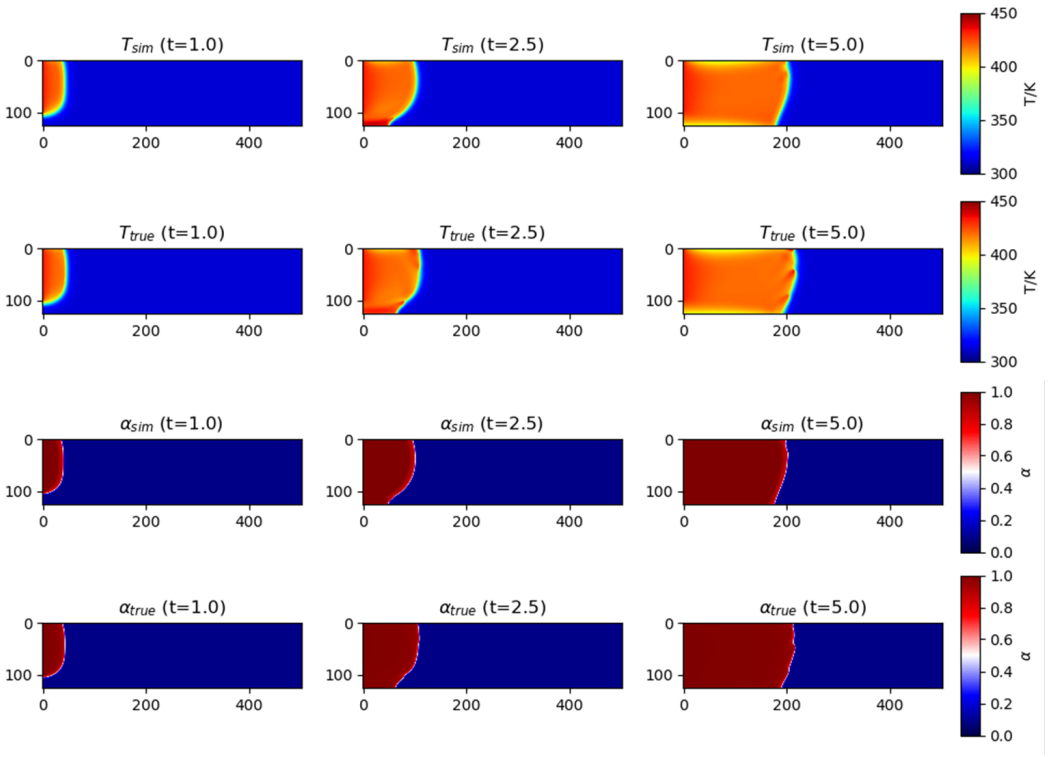

Figure 7: Comparisons of roll-out solutions with the learned $f(\alpha)$ and the true solutions for unstable FROMP of COD. Both T and α solutions are plotted at different t (Example 1, test set).

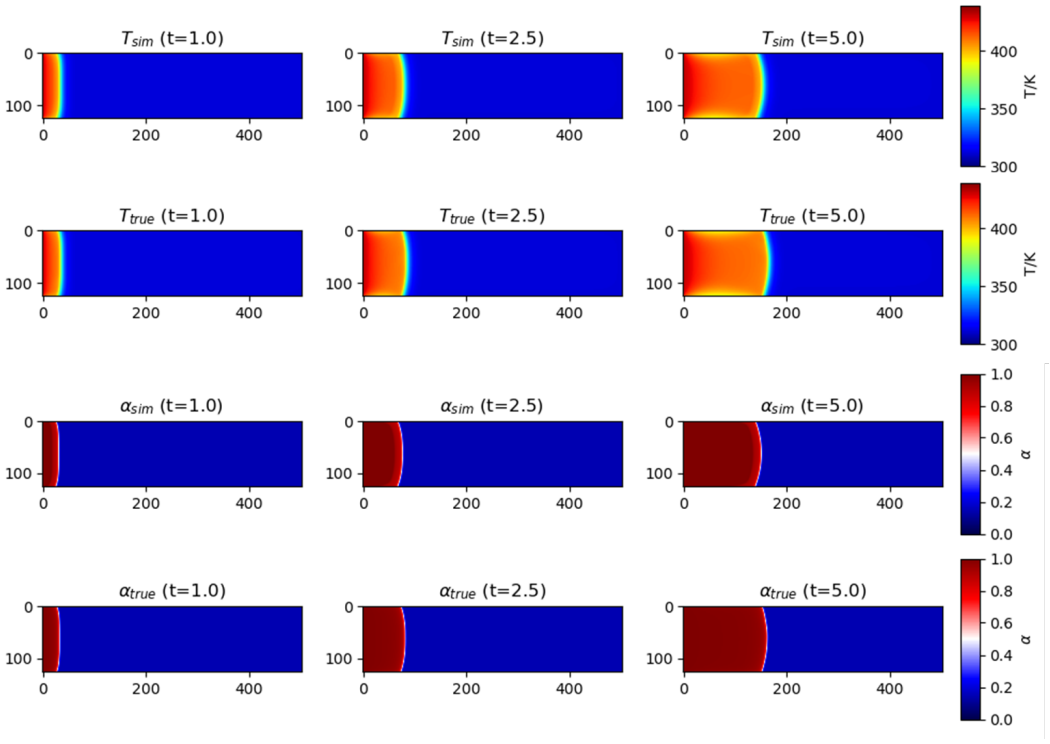

Figure 8: Comparisons of roll-out solutions with the learned $f(\alpha)$ and the true solutions for unstable FROMP of COD. Both T and α solutions are plotted at different t (Example 2, test set).