Differentiable Physics Simulations for Deep Learning (... and Beyond)

Nils Thuerey



Physical Phenomena

Everywhere around us...

• Fluid Mechanics











Physical Phenomena

Everywhere around us...

- Fluid Mechanics
- Robotic Control
- Thermodynamics
- Plasma Physics
- Medical Simulations
- Many more...







Tremendous success of numerical simulations







Physical Phenomena

Everywhere around us...

- Fluid Mechanics
- Robotic Control
- Thermodynamics
- Plasma Physics
- Medical Simulations
- Many more...





00

Machine Learning: Exciting possibilities

Numerical Methods: Keep as much as possible X



Related Work

- Schnell et. al: Half-Inverse Gradients for Physical Learning
- Holl et. al: (Scale-invariant) Physical Gradients for Deep Learning
- Um et. al: Solver-in-the-Loop, Learning from Differentiable Physics to Interact with PDE-Solvers
- Bar-Sinai et. al: Learning data-driven discretizations for partial differential equations
- Raissi et. al: Hidden physics models: Machine learning of nonlinear partial differential equations
- Chen et. al.: Neural ordinary differential equations
- Morton et. al: Deep dynamical modeling and control of unsteady fluid flows







Differentiable Simulations in a Nutshell

Discretized PDE \mathscr{P} with phase space states s Learn via gradient $(\partial \mathcal{P} / \partial \mathbf{s})^T$ E.g., with loss L and $\mathbf{s} = NN(\mathbf{x} \mid \theta)$ Gradient is $-\eta \frac{\partial \mathbf{s}^T}{\partial \theta} \frac{\partial \mathcal{P}^T}{\partial \mathbf{s}} \frac{\partial L^T}{\partial \mathcal{P}}$

Requires differentiable physics simulator for \mathscr{P}

 \rightarrow Tight integration of numerical methods and learning process





Differentiable Simulations - Terminology

Differentiable PDE solver for $\mathscr{P} =$ "differentiable physics"

Equivalent:

- Adjoint method / differentiation
- Reverse-mode / backward differentiation
- Backpropagation

D. Kahneman: System 1 & 2



1.0 NO PINNs! 0.8 0.6 X2 - 1.6 0.4 - 1.2 0.2 -- 0.8 - 0.4 0.0 -0.4 0.6 0.0 0.2 0.8 1.0 x1 System 1 / System 1 / **NN Component NN Component** System 2 / System 2 / Simulation Simulation







Differentiable Physics in Action

Um et. al: Solver-in-the-Loop: Learning from Differentiable Physics to Interact with PDE-Solvers *List et. al*: Learned Turbulence Modelling with Differentiable Fluid Solvers



"Solver-in-the-Loop"





Um et. al: Solver-in-the-Loop, Learning from Differentiable Physics to Interact with PDE-Solvers

"Solver-in-the-Loop"



Um et. al: Solver-in-the-Loop, Learning from Differentiable Physics to Interact with PDE-Solvers



Shift of Input Feature Distributions





Learning via Differentiable Physics





A few more Details...

Unsteady Wake Flow in 2D

- Setup: Reference is 4x
- 3000 frames training data, Re \in {98 . . 3125}
- Test data: new Re Nr.s
- Source MAE: 0.146
- SOL₃₂ MAE: 0.013
- More than 10x reduction

Um et. al: Solver-in-the-Loop, Learning from Differentiable Physics to Interact with PDE-Solvers



Re=2343.8





Looking into the Future

Learning via a Large Number of Simulation Steps

Evaluation:

- MAE Improvement over Src
- Supervised training: 29%
- D.P. with 4 steps: 41%
- D.P. with 128 steps: 60%

r Improvement 0.0 .0 .0 .0 .0 Error 0.3 elocity 5.0 Ve 0.1 0.0







Generalization

Improved generalization due to varied, gradient-based training feedback

- Better performance for previously unseen inputs
- Flexible due to combination with source solver

Um et. al: Solver-in-the-Loop, Learning from Differentiable Physics to Interact with PDE-Solvers





Long-term Stability

Unsteady Wake Flow (250 time steps)

3D Test Case, Re=468.8



NON MAE=0.144

Um et. al: Solver-in-the-Loop, Learning from Differentiable Physics to Interact with PDE-Solvers















Um et. al: Solver-in-the-Loop, Learning from Differentiable Physics to Interact with PDE-Solvers



Reference



Differentiable Physics

Wide Range of Applications

- Error reduction for (generic) PDEs
- Control problems
- Plasma simulations
- Model completion (reacting flows)
- Turbulence





Turbulence: Spatial Mixing Layer

	32 -
	16 -
 Semi-implicit PISO solver 	> 0-
	-16 -
(2nd order in time)	-32 -
	32 -
 Shear layer with vorticity 	16 -
thickness $Re = 500$	> 0-
	-16 -
— I I I I I C	-32 -
 Evaluate on test set of 	32 -
unseen perturbation modes	52
	16 -
	> 0-
	-16 -
	-32 -
	0

List et. al: Learned Turbulence Modelling with Differentiable Fluid Solvers







Turbulence: Spatial Mixing Layer

Learned Simulator only:





List et. al: Learned Turbulence Modelling with Differentiable Fluid Solvers



Turbulence: Spatial Mixing Layer

Closely matches DNS turbulence statistics (steady state over 2500 steps)



List et. al: Learned Turbulence Modelling with Differentiable Fluid Solvers





Training via Differentiable Physics

Numerous Advantages:

- Generalization
- Runtime performance
- Improved accuracy





Improved Updates via Inversion

Holl et. al: Scale-invariant / Physical Gradients for Deep Learning Schnell et. al: Half-Inverse Gradients for Physical Learning



Improved Learning Updates

Motivation

So far taken for granted: deep learning paradigm of optimizing via $\partial \mathcal{P} / \partial s^T$ Has fundamental problems in physical settings

- Units are wrong: deep learning gradient
- Update should have units of s, as in Newton's method: $-\eta \left(\frac{\partial^2 L}{\partial s^2}\right)^{-1} \frac{\partial L^T}{\partial s}$

Scaling problems & unstable training

Holl et. al: Scale-invariant / Physical Gradients for Deep Learning



t for **s** was:
$$-\eta \frac{\partial \mathscr{P}^T}{\partial \mathbf{s}} \frac{\partial L^T}{\partial \mathscr{P}}$$



Improved Learning Updates

Inversion is Crucial

Steepest direction via
$$\frac{\partial L^T}{\partial s}$$
 not optimal

Example: steep dimension severely limits steps

Inversion accounts for rescaling, e.g.

Either *numerical or analytical* inversion

Holl et. al: Scale-invariant / Physical Gradients for Deep Learning







Physical Gradients

Leverage Inverse Solver for Update Step

Employ (custom) inverse solver 97-

Compute NN update step via proxy-L2 loss

 $\Delta \theta_{\rm PG} = -\eta \frac{\partial \mathbf{s}^{I}}{\partial \theta} \left(\mathbf{s} - \mathcal{P} \right)$ Update step integrates gradient w.r.t. outputs \mathbf{S}_{out}

Holl et. al: Scale-invariant / Physical Gradients for Deep Learning





Scale-Inverse Physics Gradients

NN Solving Inverse Problem with Heat Diffusion





Observation

Only difference: training method (Adam or Adam + PG)

Holl et. al: Scale-invariant / Physical Gradients for Deep Learning





Reconstruct Input







NN Solving Inverse Problem with Heat Diffusion



Holl et. al: Scale-invariant / Physical Gradients for Deep Learning

$X_{A + BFGS}$

 $X_{A + PG}$

Identical NNs!



Half-inverse Gradients

Joint Inversion of Physics and Network

Partially invert Jacobian from NN and simulator jointly

Resulting update step $\Delta \theta_{\text{HIG}} = -\eta$

Over all samples of a mini-batch

Update represents optimal & scale respecting first-order step

Schnell et. al: Half-Inverse Gradients for Physical Learning

$$\left(\frac{\partial \mathscr{P}}{\partial \theta}\right)^{-1/2} \left(\frac{\partial L}{\partial \mathscr{P}}\right)^{\mathsf{T}}$$



Half-inverse Gradients

Non-linear Oscillator

Classical problem setup with non-linear force term Backprop through 96 time integration steps (RK4)







Improved Gradients - Summary

Fundamentally improved learning directions

Yields neural network states that are unreachable with simpler methods

→ Illustrates potential gains from going beyond 1st-order gradients





Summary & Outlook



Summary

Differentiable Simulations and Inversion as Tools to bridge Physics & Learning 🤐

System 1 / Component

System 2 / Simulation System 1 / NN Component

System 2 / Simulation System 1 / NN Component

> System 2 / Simulation





Improved Updates

Turbulence Modeling





Error Correction



Physics-based Deep Learning - the Book

he general direction of Fil-

g our environment, and predictin key challenges of humankind. A

⊿ very active, quickly growing and exc.
 ⊿owing chapter will give a more thorough intro.
 ⊿blish the basics for following chapters.

es running the for. Load("./temp/burgers-g) .load("./temp/burgers-pin. .p.load("./temp/burgers-diff

ones([10,33])*-1. # we'll snea
stenate([sol_gt, divider, sol

ions Ground Truth (top), PINt eshape(sbs,[N*3+20,33,1]))





Physics-based Deep Learning - the Book

- Intro to physical simulations & deep learning
- Comprehensive overview
- Hands-on code examples, run on the spot
- Among others: supervised learning, tightly coupled differentiable simulations, reinforcement learning and uncertainty modeling...







se difference images clearly show that the optimization managed to align oper region of the plumes very well. Each original image (at the top) ear misalignment in terms of a black halo, while the states after parion largely overlap the target anoke configuration of the reference

duce' marker density out of the blue to match the target how the non-linear model tons charge the state of the system ever the course of 20 time steps. goal is guite difficult, and it is not possible to exactly. lation is this scenario ble at the sterrs of the sincke plumes, which still show a ck halo after the optimization. The optimization was not able to shift the wiposition, and hence needs to focus on aligning the upper regions of









They both start out with the same initial state at \(t=0\) (the downsamp solution from the reference solution manifold), and at \(t=20\) the solutions still share similarities. Over time, the source version strongly diffuses the structures in the flow and looses momentum. The flow behind the obstacles becomes straight, and lacks clear vortices.

The version produced by the hybrid solver does much better. It preserves the vortex shedding even after more than one hundred updates. Note that both outputs were produced by the same underlying solver. The second version just profits from the learned corrector which manages to revert the numerical errors of the source solver, including its overly strong dissipation







Thanks for Listening!



https://physicsbaseddeeplearning.org

Acknowledgements: P. Holl, K. Um, V. Koltun, R. Fei, B. List, L. W. Chen



