Towards the True Hybrid

Physics-informed Trainable Models for Prognostics and Health Management

Matteo Corbetta

KBR Inc., NASA Ames Research Center

ENHaNCE Training Week, April 22, 2021

Acknowledgments

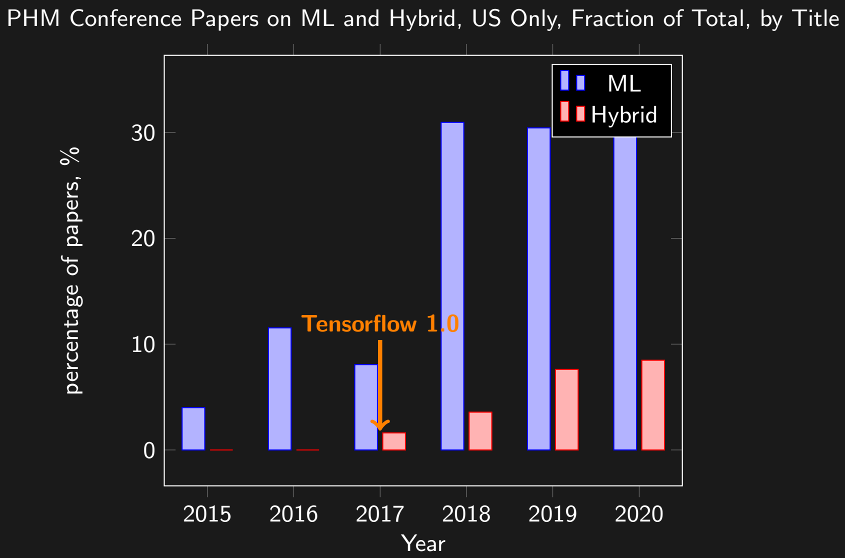
- Chetan Kulkarni, KBR, NASA Ames
- Stefan Schuet, NASA Ames
- Renato G. Do Nascimento, UCF
- Felipe Viana, UCF
- Michael Khasin, NASA Ames
- ENHANCE Training Week organizers, especially Manuel, Maria, and Juan, University of Granada, Spain
- Claudio Sbarufatti and Francesco Cadini, PoliMi, Italy

All authors of the papers we'll discuss here.

Agenda

- Recent Increase in Machine Learning Papers for PHM
- Why to Leverage Physics-Informed Learning in PHM
- How to do it: Overview of Current Research in Physics-Informed Machine Learning
  - Physics-Constrained Gaussian Processes
  - Physics-Informed Neural Networks
    - Implicit Constraints
    - Explicit Constraints
  - Data-Driven Discovery of Interpretable Dynamical Models
    - Sparse Identification of Non-Linear Dynamics (SINDy) for ODE
- Concluding Remarks

Recent Increase in Machine Learning Papers for PHM



Why to Leverage Physics-Informed Learning in PHM

Great achievements of ML in many scientific areas

AI system for breast cancer screening, Nature, Jan 2020
Generating "Art" by Learning About Styles and Deviating from Style Norms. ICCC, June 2017
DeepFace: Closing the Gap to Human-Level Performance in Face Verification, CCVPR, June 2014

Why to Leverage Physics-Informed Learning in PHM

Great achievements of ML in many scientific areas

AI system for breast cancer screening, Nature, Jan 2020
Generating "Art" by Learning About Styles and Deviating from Style Norms. ICCC, June 2017
DeepFace: Closing the Gap to Human-Level Performance in Face Verification, CCVPR, June 2014

"Data is the new oil"

The Economist, May 6th, 2017.

Why to Leverage Physics-Informed Learning in PHM

"Data is the new oil"

The Economist, cover story, May 6th, 2017.

... maybe.

Why to Leverage Physics-Informed Learning in PHM

"Data is the new oil"

The Economist, cover story, May 6th, 2017.

... maybe.

"Data is the new *snake*-oil; because when we'll have better *models*, we'll need *less data*."

Stuart Russel, UAI Conference, Monterey, CA, August 2018

Why to Leverage Physics-Informed Learning in PHM

Well-Known Drawbacks of Pure Data-Driven Methods

- Need of data we (typically) do not have in PHM
- Poor interpretability due to:
  - Large number of parameters
  - Lack of physical meaning: "good spot" in the parameter space w.r.t. $(y - \hat{y})$
- Validation and Generalization are hard when:
  - Cannot record or don't know we need to record external forcing
  - Model needs to extrapolate outside training domain, like in many degradation cases!

Why can be worth it

- Use the (partial) knowledge we have seems clever
- Restrict the search in the parameter space can save data and time
- Improve generalization and extrapolation to unseen input
- Improve *interpretability*

Why to Leverage Physics-Informed Learning in PHM

Questions before we get to the fun stuff? ;)

How to do it: Overview of Current Research in Physics-Informed Machine Learning

Physics-Constrained Gaussian Processes

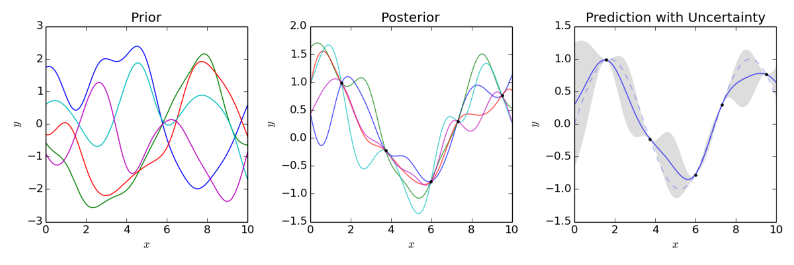
Thanks to Stefan Schuet for most of the work here.

Physics-Constrained Gaussian Processes

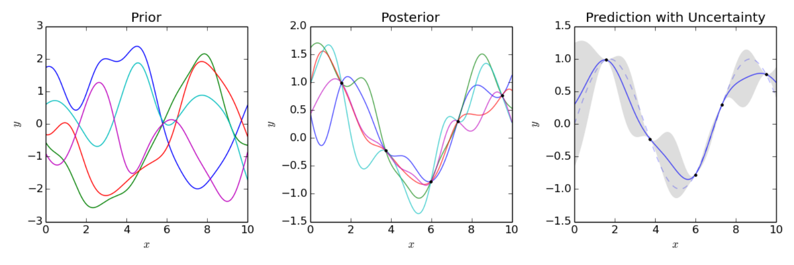
Gaussian Processes (from *Rasmussen & Williams, 2006*)

A Gaussian Process (GP) is a collection of random variables, any finite number of which has a joint Gaussian distribution.

Think of sampling functions from a function space instead of sampling variables:
(image from Wikipedia: wiki/Gaussian_process)



Physics-Constrained Gaussian Processes



$$\boldsymbol{y} \sim \mathcal{GP}\left(m(\boldsymbol{x}), \Sigma_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{x}')\right)$$

$$\begin{bmatrix} \boldsymbol{y} \\ \boldsymbol{y}_* \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} m(\boldsymbol{x}) \\ m(\boldsymbol{x}_*) \end{bmatrix}, \begin{bmatrix} k_{\boldsymbol{\theta},\boldsymbol{x}\boldsymbol{x}} + \sigma_n^2 I & k_{\boldsymbol{\theta},\boldsymbol{x}\boldsymbol{x}_*} \\ k_{\boldsymbol{\theta},\boldsymbol{x}\boldsymbol{x}_*}^\top & k_{\boldsymbol{\theta},\boldsymbol{x}_*\boldsymbol{x}_*} \end{bmatrix} \right)$$

$$\boldsymbol{y}_* | \boldsymbol{x}, \boldsymbol{y}, \boldsymbol{\theta} \sim \mathcal{GP}\left(\tilde{\mu},\ \tilde{\Sigma}\right)$$

Physics-Constrained Gaussian Processes

Use of Physics-Informed GPs

Surrogate models of complex physical phenomena

- Trainable with few data-points
- Fast to run w.r.t. FEM, CFD,
- Improve extrapolation

Physics-Constrained Gaussian Processes

From GPs to Physics-Informed GPs

Function we want to approximate (unknown)	$f(\boldsymbol{x}), \quad \boldsymbol{x} \in \mathbb{R}^{n \times 1}$
We also know from the physics that	$\mathcal{L}_{\boldsymbol{x}} f(\boldsymbol{x}) = 0$

Physics-Constrained Gaussian Processes

From GPs to Physics-Informed GPs

Function we want to approximate (unknown)	$f(\boldsymbol{x}), \quad \boldsymbol{x} \in \mathbb{R}^{n\times 1}$
We also know from the physics that	$\mathcal{L}_{\boldsymbol{x}} f(\boldsymbol{x}) = 0$
We want to approximate f with a GP	$f(\boldsymbol{x}) \approx \hat{f}(\boldsymbol{x}) \sim \mathcal{GP}$

Physics-Constrained Gaussian Processes

From GPs to Physics-Informed GPs

Function we want to approximate (unknown)	$f(\boldsymbol{x}), \quad \boldsymbol{x} \in \mathbb{R}^{n \times 1}$
We also know from the physics that	$\mathcal{L}_{\boldsymbol{x}} f(\boldsymbol{x}) = 0$
We want to approximate f with a GP	$f(\boldsymbol{x}) \approx \hat{f}(\boldsymbol{x}) \sim \mathcal{GP}$

Vector $[\hat{f}(x), \mathcal{L}_{\boldsymbol{x}} f(\boldsymbol{x})]^\top$ is also a $\mathcal{GP}$

- $\hat{f}$ is a $\mathcal{GP}$ (holds linear properties)
- Derivatives are linear operations
- The problem is to find the derivative of the $\mathcal{GP}$ mean function and covariance matrix

Physics-Constrained Gaussian Processes

$$\begin{bmatrix} \hat{f}(\boldsymbol{x}) \\ \frac{\partial \hat{f}(\boldsymbol{x})}{\partial x_i} \\ \frac{\partial^2 \hat{f}(\boldsymbol{x})}{\partial x_j^2} \\ \cdots \end{bmatrix} \sim \mathcal{GP}\left(\begin{bmatrix} m(x) \\ \frac{\partial m(x)}{\partial x_i} \\ \frac{\partial^2 m(x)}{\partial x_j^2} \\ \cdots \end{bmatrix}, \begin{bmatrix} \Sigma & \frac{\partial \Sigma}{\partial x_i}^\top & \frac{\partial^2 \Sigma}{\partial x_j^2}^\top & \cdots \\ \frac{\partial \Sigma}{\partial x_i} & \frac{\partial^2 \Sigma}{\partial x_i^2} & \frac{\partial^3 \Sigma}{\partial x_i \partial x_j^2} & \cdots \\ \frac{\partial^2 \Sigma}{\partial x_j^2} & \frac{\partial^3 \Sigma}{\partial x_i^2 \partial x_j} & \frac{\partial^4 \Sigma}{\partial x_j^4} & \cdots \\ \cdots & \cdots & \cdots & \end{bmatrix} \right)$$

Physics-Constrained Gaussian Processes

$$\begin{bmatrix} \hat{f}(\boldsymbol{x}) \\ \frac{\partial \hat{f}(\boldsymbol{x})}{\partial x_i} \\ \frac{\partial^2 \hat{f}(\boldsymbol{x})}{\partial x_j^2} \\ \dots \end{bmatrix} \sim \mathcal{GP}\left(\begin{bmatrix} m(x) \\ \frac{\partial m(x)}{\partial x_i} \\ \frac{\partial^2 m(x)}{\partial x_j^2} \\ \dots \end{bmatrix}, \begin{bmatrix} \Sigma & \frac{\partial \Sigma}{\partial x_i}^\top & \frac{\partial^2 \Sigma}{\partial x_j^2}^\top & \dots \\ \frac{\partial \Sigma}{\partial x_i} & \frac{\partial^2 \Sigma}{\partial x_i^2} & \frac{\partial^3 \Sigma}{\partial x_i \partial x_j^2} & \dots \\ \frac{\partial^2 \Sigma}{\partial x_j^2} & \frac{\partial^3 \Sigma}{\partial x_i^2 \partial x_j} & \frac{\partial^4 \Sigma}{\partial x_j^4} & \dots \\ \dots & \dots & \dots & \end{bmatrix} \right)$$

$$\begin{bmatrix} \hat{f}(\boldsymbol{x}) \\ \mathcal{L}_{\boldsymbol{x}} \hat{f}(\boldsymbol{x}) \end{bmatrix} \sim \mathcal{GP}(\dots, \dots)$$

Physics-Constrained Gaussian Processes

$$\begin{bmatrix} \hat{f}(\boldsymbol{x}) \\ \frac{\partial \hat{f}(\boldsymbol{x})}{\partial x_i} \\ \frac{\partial^2 \hat{f}(\boldsymbol{x})}{\partial x_j^2} \\ \dots \end{bmatrix} \sim \mathcal{GP}\left(\begin{bmatrix} m(x) \\ \frac{\partial m(x)}{\partial x_i} \\ \frac{\partial^2 m(x)}{\partial x_j^2} \\ \dots \end{bmatrix}, \begin{bmatrix} \Sigma & \frac{\partial \Sigma}{\partial x_i}^\top & \frac{\partial^2 \Sigma}{\partial x_j^2}^\top & \dots \\ \frac{\partial \Sigma}{\partial x_i} & \frac{\partial^2 \Sigma}{\partial x_i^2} & \frac{\partial^3 \Sigma}{\partial x_i \partial x_j^2} & \dots \\ \frac{\partial^2 \Sigma}{\partial x_j^2} & \frac{\partial^3 \Sigma}{\partial x_i^2 \partial x_j} & \frac{\partial^4 \Sigma}{\partial x_j^4} & \dots \\ \dots & \dots & \dots & \end{bmatrix} \right)$$

$$\begin{bmatrix} \hat{f}(\boldsymbol{x}) \\ \mathcal{L}_{\boldsymbol{x}} \hat{f}(\boldsymbol{x}) \end{bmatrix} \sim \mathcal{GP}(\dots, \dots)$$

$$\hat{f}(\boldsymbol{x}) | \mathcal{L}_{\boldsymbol{x}} \sim \mathcal{N}(\tilde{\mu}, \tilde{\Sigma})$$

Additive manufacturing: material properties as function of process parameters

Physics-Constrained Gaussian Processes

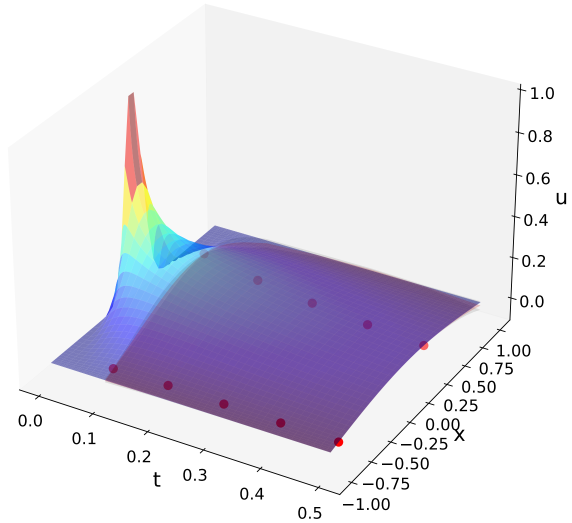
Example: Heat Equation

$$f(x,0) = \delta_{x,0}$$

$$\mathcal{L}_{\boldsymbol{x}} f(\boldsymbol{x}) = \frac{\partial f(x,t)}{\partial t} - \alpha \frac{\partial^2 f(x,t)}{\partial x^2} = 0$$

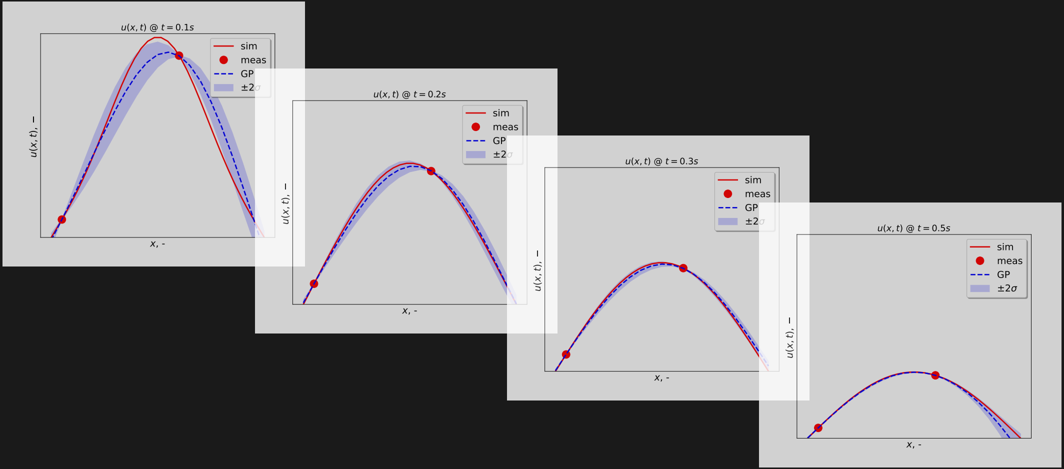
$$\begin{bmatrix} \hat{f}(x,t) \\ \hat{f}_t - \alpha \hat{f}_{xx} \end{bmatrix} \sim \mathcal{GP}(\tilde{m}, \tilde{\Sigma})$$

Simulated with Matlab (analytical solution)



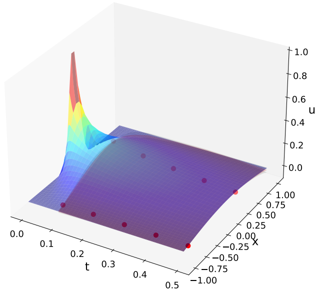
Physics-Constrained Gaussian Processes

Heat Equation



Physics-Constrained Gaussian Processes

Problems



1. Covariance matrices are big and built upon symbolic calculus. It's time consuming and creates hyper-parameter optimization issues.
2. Physical processes are characterized by varying lengthscales in time, space, or both (that's why we didn't fit the beginning!). In this example, a kernel function with a single lengthscale parameter cannot fit well the very sharp heat drop (start) and the slow heat diffusion (later) in the process. Need of a *non-stationary* covariance function for that.

$$k = \sigma^2 \exp\left(-\frac{||\mathbf{x} - \mathbf{x}'||}{2\ell^2}\right)$$

3. Solution to problem #2 makes the solution to problem #1 more challenging

Physics-Constrained Gaussian Processes

References and Q&A

1. http://www.gaussianprocess.org/gpml/ (Rasmussen & Williams)

2. Carl Edward Rasmussen. Gaussian processes in machine learning.
   In *Summer school on machine learning*, pages 63–71. Springer, 2003

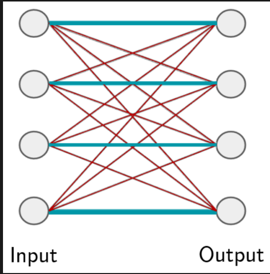
3. Emil M Constantinescu and Mihai Anitescu. Physics-based covariance models for gaussian processes with multiple outputs.
   International Journal for Uncertainty Quantification, 3(1), 2013

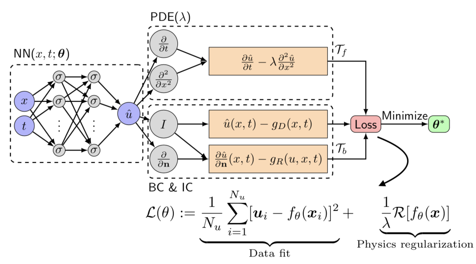
4. Carl Jidling, Niklas Wahlström, Adrian Wills, and Thomas B Schön. Linearly constrained gaussian processes.
   arXiv preprint arXiv:1703.00787, 2017

5. Markus Lange-Hegermann. Algorithmic linearly constrained gaussian processes.
   arXiv preprint arXiv:1801.09197, 2018

Physics-Informed Neural Networks

Implicit Constraints (a) Explicit Constraints (b)





(a) from Zaheer et al. Deep Sets, 2017. (b) from Prof. Paris Perdikaris, AAAI-MLPS 2020

Physics-Informed Neural Networks– Implicit Constraints

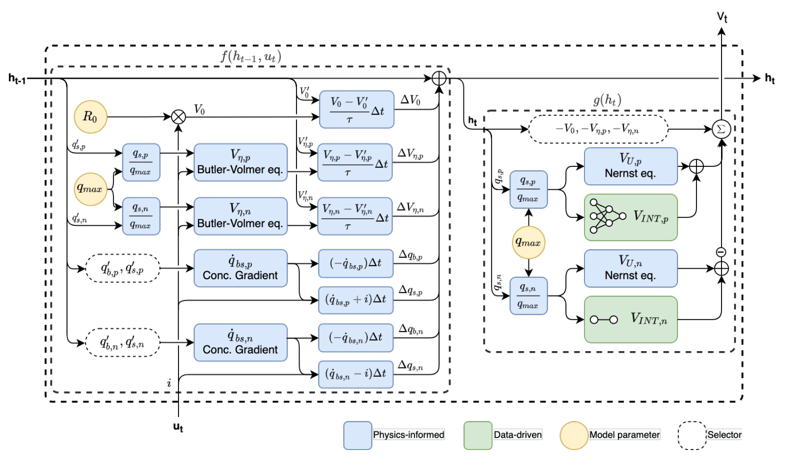
- **Network-Embedded Models**: insert neural networks (e.g., MLPs) within a bigger model to learn unknown physics.
- **Constrained-Connection Networks**: Build a trainable model with ad-hoc connections and equations, so that parameters are (at least partially) driven by physics.

Battery Discharge and Aging Prediction with Implicit PINN – Tensorflow

Thanks to:
Felipe Viana (University of Central Florida),
Renato Giorgiani Do Nascimento (University of Central Florida), and
Chetan Kulkarni (KBR Inc, NASA Ames Research Center)

Physics-Informed Neural Networks– Implicit Constraints

Battery Discharge and Aging Prediction with Implicit PINN – Tensorflow



Predict voltage curve during operation and battery aging

- Parameters R_0 and q_{max} estimated after each discharge
- Interval voltages $V_{INT,\cdot}$ approximated using MLPs

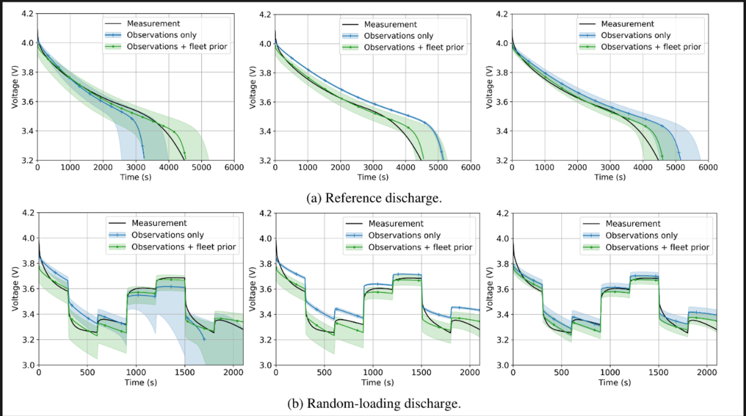
Physics-Informed Neural Networks– Implicit Constraints

Battery Discharge and Aging Prediction with Implicit PINN – Tensorflow

```python
class BatteryRNNCell(Layer):
    def __init__(self, q_max_model=None, R_0_model=None, curr_cum_pwh=0.0, initial_state=None, dt=1.0, qMobile=7600, mlp_trainable=True,
        super(BatteryRNNCell, self).__init__(**kwargs)

        self.initial_state = initial_state
        self.dt = dt
        self.qMobile = qMobile
        self.q_max_base_value = q_max_base
        self.R_0_base_value = R_0_base

        self.q_max_model = q_max_model
        self.R_0_model = R_0_model
        self.curr_cum_pwh = curr_cum_pwh

        self.initBatteryParams(batch_size, D_trainable)

        self.state_size  = tensor_shape.TensorShape(8)
        self.output_size = tensor_shape.TensorShape(1)

        self.MLPp = Sequential([
            # Dense(8, activation='tanh', input_shape=(1,), dtype=self.dtype, kernel_regularizer=tf.keras.regularizers.l2(0.001)),
            Dense(8, activation='tanh', input_shape=(1,), dtype=self.dtype),
            Dense(4, activation='tanh', dtype=self.dtype),
            Dense(1, dtype=self.dtype),
        ], name="MLPp")
```

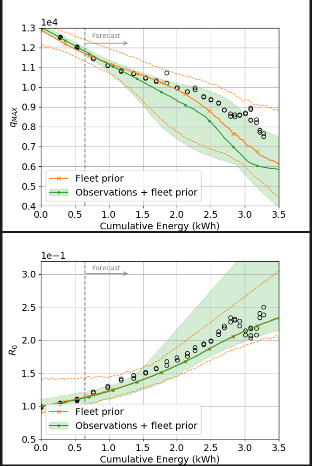
Physics-Informed Neural Networks– Implicit Constraints

Battery Discharge and Aging Prediction with Implicit PINN – Tensorflow



(a) Reference discharge.

(b) Random-loading discharge.



Physics-Informed Neural Networks– Implicit Constraints

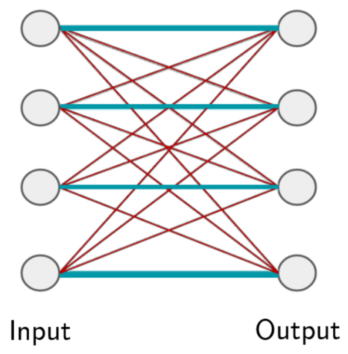
Constrained-Connection Networks

Idea is old! papers talking about it from 1988.
Impose symmetries, conservation laws, as well as other properties of the system to reduce the size of the set of latent space outputs

- Forcing parameter-sharing (Convolutional layers are a sub-category)
- Use functions invariant to specific properties (e.g., the sum is invariant to ordering)

Ex: Equivariant Layer (Zaheer et Al.)

$$\boldsymbol{f}_{\Theta}(\boldsymbol{x}) = \boldsymbol{\sigma}(\Theta\,\boldsymbol{x}),\;\; \Theta = \lambda\,\boldsymbol{I} + \gamma\left(\mathbf{1}\mathbf{1}^{\top}\right)$$



Physics-Informed Neural Networks– Implicit Constraints

Problems

Model-embedded Networks

- Careful tuning of initial value is still a thing
- Optimizers don't know physical limits of parameters $\rightarrow$ huge or small numbers trying to compensate for model errors. When you limit them, the optimization get stuck in local minima.
- Uncertainty estimate possible with *Variational layers*, but the Tensorflow framework still doesn't allow you to use those layers in every circumstance.

Constrained-connection Networks

- Deep networks can represent symmetries in the depth-direction, not only within-layer (problem-dependent) $\rightarrow$ not as easy to define the implicit constraints.
- Initialization is crucial to be able to train effectively

Physics-Informed Neural Networks

References and Q&A

1. Arinan Dourado and Felipe AC Viana. Physics-informed neural networks for missing physics estimation in cumulative damage models: a case study in corrosion fatigue.
   Journal of Computing and Information Science in Engineering, 20(6), 2020

2. Renato Giorgiani Nascimento and Felipe AC Viana. Fleet prognosis with physics-informed recurrent neural networks.
   arXiv preprint arXiv:1901.05512, 2019

3. Kajetan Fricke, Renato Giorgiani do Nascimento, and Felipe Viana. Quadcopter soft vertical landing control with hybrid physics-informed machine learning.
   In *AIAA Scitech 2021 Forum*, page 1018, 2021

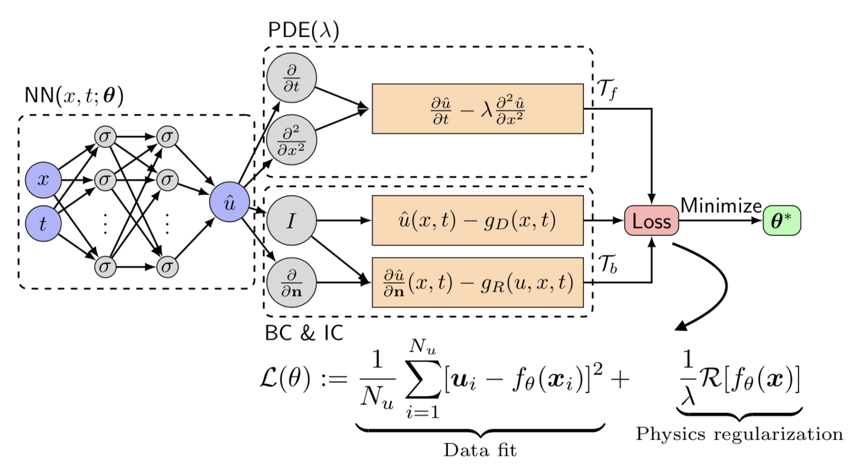
4. Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. Deep sets.
   arXiv preprint arXiv:1703.06114, 2017

5. Robert Gens and Pedro M Domingos. Deep symmetry networks.
   Advances in neural information processing systems, 27:2537–2545, 2014

Physics-Informed Neural Networks– Explicit Constraints

- Physics is explicity imposed in the output by adds-on to the loss function; if the solution does not satisfy physical constraints, the loss goes up
- It can easily handles initial and boundary conditions from ODE and PDE
- Automatic differentiation takes care of computing the derivative of the loss w.r.t. the input

Physics-Informed Neural Networks– Explicit Constraints



Pic from Prof. Paris Perdikaris, AAAI-MLPS 2020

Extensive testing of the approach came from:

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations.
Journal of Computational Physics, 378:686–707, 2019

Many papers solving PDE-related problems with the approach. Once again, taking advantage of DNN frameworks (e.g., Tensorflow or Pytorch):

Physics-Informed Neural Networks– Explicit Constraints

Snippet Code

Taken from Raissi 2019:

$u(t, x) \rightarrow$ unknown function

$u_t + u * u_{xx} - (0.01/\pi) * u_{xx} = 0 \rightarrow$ PDE (Burger's equation)

Code Snippet

```
def u(t, x):
    # NN output; neural_net is built using standard layers
    return neural_net(tf.concat([t, x],1), weights, biases)

def f(t, x):
    # PDE as f = u_t + u*u_{xx} - (0.01/pi) * u_{xx}
    uval = u(t, x)
    u_t  = tf.gradients(uval, t)[0] # compute necessary gradients
    u_x  = tf.gradients(uval, x)[0]
    u_xx = tf .gradients(u_x, x)[0]
    return u_t + u * u_x - (0.01/tf.pi) * u_xx
```

Physics-Informed Neural Networks– Explicit Constraints

Problems

- Many papers reiterating on academic examples and fundamental equations, some good CFD surrogate modeling, but little practical applications. Possible reasons:
  - Most of the time we *don't* know BC, or we have partial information
  - Generalization issues; does the trained NN work with any valid BC/IC once it has been trained?
- Extrapolation ability; prediction outside training domain while changing BC/IC?
- Data set size requirements still seems to be large, even if reduced

Data-Driven Discovery of Interpretable Dynamical Models

Data-Driven Discovery of Interpretable Dynamical Models

- Ideas came early (as always) – James P Crutchfield and BS McNamara. Equations of motion from a data series.

  Complex systems, 1(417-452):121, 1987

- Major contributions lately from Brunton, Kutz et al:

  - Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016
  - Samuel H Rudy, Steven L Brunton, Joshua L Proctor, and J Nathan Kutz.

    Data-driven discovery of partial differential equations. *Science Advances*, 3(4), 2017

Data-Driven Discovery of Interpretable Dynamical Models

Brunton, Kutz et. al., University of Washington

Goal: Utilize sparse regression methods to discover governing equations from data.
Can we *find* f $(\mathcal{N})$ having $x(t)$ $(u(t))$?

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t))$$

$$u_t = \mathcal{N}(u, u_x, u_{xx}, \mu)$$

Data-Driven Discovery of Interpretable Dynamical Models

Brunton, Kutz et. al., University of Washington

Goal: Utilize sparse regression methods to discover governing equations from data.
Can we *find* f ($\mathcal{N}$) having $x(t)$ ($u(t)$)?

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t))$$

$$u_t = \mathcal{N}(u, u_x, u_{xx}, \mu)$$

f, $\mathcal{N}$ must be sparse w.r.t. the space of possible functions.

Data-Driven Discovery of Interpretable Dynamical Models

Brunton, Kutz et. al., University of Washington

Goal: Utilize sparse regression methods to discover governing equations from data.
Can we *find* f ($\mathcal{N}$) having $x(t)$ ($u(t)$)?

$$\dot{\boldsymbol{x}}(t) = \boldsymbol{f}(\boldsymbol{x}(t))$$

$$u_t = \mathcal{N}(u, u_x, u_{xx}, \mu)$$

f, $\mathcal{N}$ must be sparse w.r.t. the space of possible functions.

Potential use of sparse identification:

- Aid model discovery for complex phenomena (e.g., fluid dynamics)
- Learn interpretable models
- Improve extrapolation (inherent property)
- Learn new dynamics on-the-fly

Data-Driven Discovery of Interpretable Dynamical Models

Sparse Identification of Non-Linear Dynamics (SINDy) for ODE

$$\dot{x}(t) = f\,(x(t))$$

Approach

Find the nonlinear vector function f among a set of candidate functions.

- Generate a library of candidate functions
- Solve the regression problem by promoting sparsity
- Use the regression coefficients and the candidate functions to generate your model

Sparse Identification of Non-Linear Dynamics (SINDy) for ODE

Generation of the candidate function library

$$\boldsymbol{X}(t) = \begin{array}{c} \xrightarrow{\text{state dimension}} \\ \begin{bmatrix} | & | & \cdots & | \\ x_1(t_i) & x_2(t_i) & \cdots & x_n(t_i) \\ | & | & \cdots & | \end{bmatrix} \end{array} \Big\downarrow \text{time}$$

$$\boldsymbol{\Theta}(\boldsymbol{X}) = \begin{bmatrix} | & | & | & \cdots & | & | & \cdots \\ \mathbf{1} & \boldsymbol{X} & \boldsymbol{X}^{p_2} & \cdots & \sin \boldsymbol{X} & \cos \boldsymbol{X} & \cdots \\ | & | & | & \cdots & | & | & \cdots \end{bmatrix}$$

Sparse Identification of Non-Linear Dynamics (SINDy) for ODE

Solve the regression problem: ℓ_0 regularized regression

$$\xi_k = \arg\min_{\hat{\xi}_k} ||\dot{\boldsymbol{X}} - \boldsymbol{\Theta}\left(\boldsymbol{X}\right)\hat{\xi}_k||_2 + \lambda||\hat{\xi}_k||_0$$

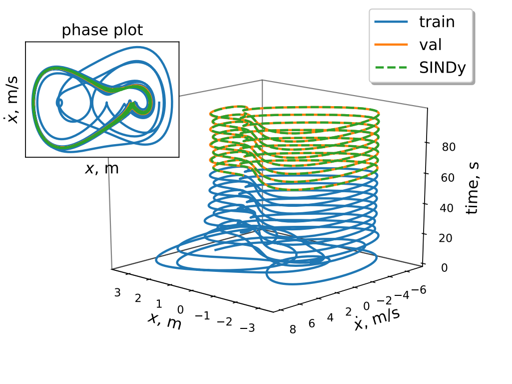
- Solved using *sequential thresholded least square* – give penalty for $\hat{\xi}_k \neq 0$
- One vector ξ_k for each state variable ($k = 1, \ldots, n$)
- λ is the sparsity promoter
- After solving for all ξ:

$$\dot{\boldsymbol{X}}(t) = \boldsymbol{\Theta}(\boldsymbol{X}(t))\boldsymbol{\Xi}$$

Sparse Identification of Non-Linear Dynamics (SINDy) for ODE

Simple tests: Modified Van Der-Pol Oscillator

$$\ddot{x} + f(c/m, \dot{x}, x^2) + f(k/m, x, x^3) = \frac{1}{m}F(t)$$



Θ-column	ξ_2	Reference	Error, %
1	0.0	0	-
x	-0.24938066	-0.25	0.25
$\dot{x}$	0.15618899	0.16	2.38
x^2	0.0	0.0	-
$x\dot{x}$	0.0	0.0	-
$\dot{x}^2$	0.0	0.0	-
x^3	-1.25049414	-1.25	0.04
$x^2\dot{x}$	-0.15509556	-0.16	3.0
$x\dot{x}^2$	0.0	0.0	-
$\dot{x}^3$	0.0	0.0	-
$F(t)$	0.05003716	0.05	0.07

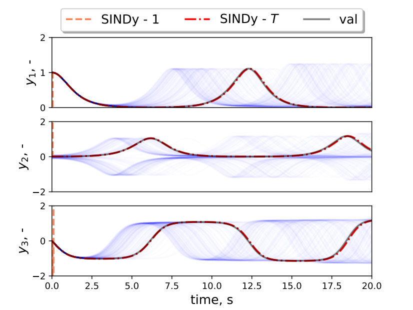
Sparse Identification of Non-Linear Dynamics (SINDy) for ODE

Simple tests: Non-linear System with Random IC

$$\frac{\mathrm{d}y_1}{\mathrm{d}t} = y_1 y_3$$
$$\frac{\mathrm{d}y_2}{\mathrm{d}t} = -y_2 y_3$$
$$\frac{\mathrm{d}y_3}{\mathrm{d}t} = -y_1^2 + y_2^2$$

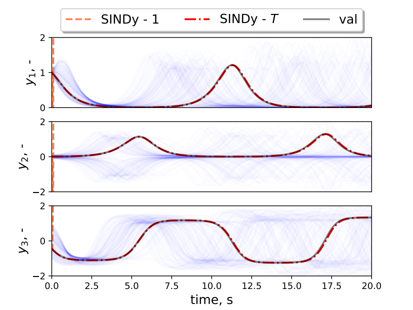
$y_2(0) = 0.1\,u$

200 samples (empirical)



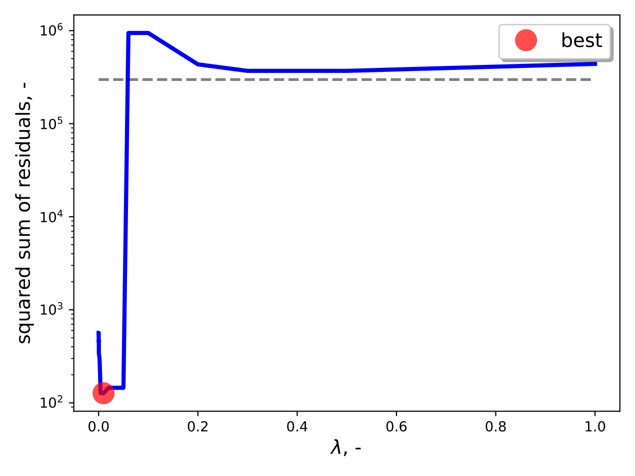
$y_2(0) = 0.1\,u\,,\ \ y_3(0) = u$

400 samples (empirical)



Sparse Identification of Non-Linear Dynamics (SINDy) for ODE

On the selection of λ



Sparse Identification of Non-Linear Dynamics (SINDy) for ODE

Problems

- Create the candidate function library; if we knew what should go in there ...
- Systems with control: works only for specific control logic.
  Model Predictive Control $\rightarrow$ yes
  LQR $\rightarrow$ no
- External forcing $\boldsymbol{u}$: must be included in the function library $\rightarrow$ we must measure it

Data-Driven Discovery of Interpretable Dynamical Models

References & Q&A

1. Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems.
   Proceedings of the National Academy of Sciences, 113(15):3932–3937, 2016

2. Samuel H Rudy, Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Data-driven discovery of partial differential equations.
   Science Advances, 3(4), 2017

3. Eurika Kaiser, J Nathan Kutz, and Steven L Brunton. Sparse identification of nonlinear dynamics for model predictive control in the low-data limit.
   Proceedings of the Royal Society A, 474(2219):20180335, 2018

4. Eurika Kaiser, J Nathan Kutz, and Steven Brunton. Data-driven discovery of koopman eigenfunctions for control.
   Machine Learning: Science and Technology, 2021


Thank you

References

Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016.

Emil M Constantinescu and Mihai Anitescu. Physics-based covariance models for gaussian processes with multiple outputs. *International Journal for Uncertainty Quantification*, 3(1), 2013.

James P Crutchfield and BS McNamara. Equations of motion from a data series. *Complex systems*, 1(417-452):121, 1987.

Arinan Dourado and Felipe AC Viana. Physics-informed neural networks for missing physics estimation in cumulative damage models: a case study in corrosion fatigue. *Journal of Computing and Information Science in Engineering*, 20(6), 2020.

Kajetan Fricke, Renato Giorgiani do Nascimento, and Felipe Viana. Quadcopter soft vertical landing control with hybrid physics-informed machine learning. In *AIAA Scitech 2021 Forum*, page 1018, 2021.

Robert Gens and Pedro M Domingos. Deep symmetry networks. *Advances in neural information processing systems*, 27:2537–2545, 2014.

Carl Jidling, Niklas Wahlström, Adrian Wills, and Thomas B Schön. Linearly constrained gaussian processes. *arXiv preprint arXiv:1703.00787*, 2017.

Eurika Kaiser, J Nathan Kutz, and Steven L Brunton. Sparse identification of nonlinear dynamics for model predictive control in the low-data limit. *Proceedings of the Royal Society A*, 474(2219):20180335, 2018.

Eurika Kaiser, J Nathan Kutz, and Steven Brunton. Data-driven discovery of koopman eigenfunctions for control. *Machine Learning: Science and Technology*, 2021.

Markus Lange-Hegermann. Algorithmic linearly constrained gaussian processes. *arXiv preprint arXiv:1801.09197*, 2018.

Renato Giorgiani Nascimento and Felipe AC Viana. Fleet prognosis with physics-informed recurrent neural networks. *arXiv preprint arXiv:1901.05512*, 2019.

Maziar Raissi, Paris Perdikaris, and George E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019.

Carl Edward Rasmussen. Gaussian processes in machine learning. In *Summer school on machine learning*, pages 63–71. Springer, 2003.

Samuel H Rudy, Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Data-driven discovery of partial differential equations. *Science Advances*, 3(4), 2017.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. Deep sets. *arXiv preprint arXiv:1703.06114*, 2017.