Interplay between data, physics and simulation models. Link to industrial use-cases.

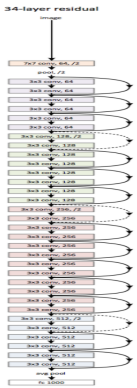
Mathilde Mougeot, ensIIE & Centre Borelli, ENS Paris-Saclay, France.

StatMathAppli, Fréjus, September 2^{nd} 2025

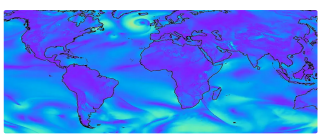
Motivation. Once calibrated on large sets, deep neural network models, have proven their efficiency and speed of execution. Subject to a good understanding of a physical phenomenon, simulation models can, in some cases, do without observational data, but their computational cost can be prohibitive. Motivated by industrial use cases, this talk presents some recent works on machine learning based on physics informed techniques, enabling efficient models to be obtained in cases where the volume of observational data available is low.

Data Sources & several Successes of "ML/AI" models
AI Foundations models

▶ Imagenet is a huge open source database :
14.10^6 labeled images for 10^3 **categories**,
... "quite expensive" labeling effort....
today pretrained models and DB are open source and may be used fo image classification, object detection...
AlexNet [Krizhevsky et al., 2012], ResNet [Huang et al., 2017]...



▶ GraphCast. This state-of-the-art model delivers 10-day weather predictions at unprecedented accuracy in under one minute based on 39 years (1979–2017) of historical data from ECMWF's ERA5 (21) reanalysis archive.



▶ DeepL. relied on the huge French-English Linguee dictionary.

▶ ChatGPT, LLM. trained on very large corpus of text data.

▶ Foundation models. **open-source and pretrained models** dedicated to various tasks under various software licences...

Top-performing deep architectures are trained on massive amounts of labeled data.

Learning a model from an algorithmic point of view...

 # The Train Data and your New, Target data
(XTrain,yTrain)= load(MyTrainDataBase);
(XNew,yNew)= load(MyNewData);

 # The NN model
from sklearn.neural_network import MLPClassifier
clf = MLPClassifier(solver='lbfgs', alpha=0.0001, hidden_layer_sizes=(5, 2), random_state=1)
clf.fit(XTrain, yTrain)

 # The Decision Tree model
from sklearn.tree impot DecisionTreeClassifier
treemod = DecisionTreeClassifier(
min_samples_split=10,min_samples_leaf=5,min_impurity_decrease=0)

 # Decision on new data
(XNew,yNew)=load(MyNewData)
pyNew=clf.predict_proba(Xnew)
treefit= treemod.fit (XTrain,YTrain)

 # Error Evaluation
pyNew=treefit.predict_proba(Xnew)>0.5

1. Input/ output (X, Y)
   (Features, labels set) defined by the operational need.
2. Data set. $S = \{(x_i, y_i)\}_{i=1}^{m} \sim \mathcal{D}^m$ a learning/training sample of m iid pairs.
   with $\mathcal{D}$ an unknown joint probability distribution on the product space $X \otimes Y$
3. Model $\mathcal{H} = \{h_\theta | h_\theta : X \to Y\}$ a hypothesis class, θ parameters, classifiers or regressors depending on the nature of Y.
4. Loss function $\ell(y, h_\theta(x))$ providing a cost of $h_\theta(x)$ deviating form the true output $y \in Y$.
   The best hypothesis is the one that minimizes the true risk, consequently, generalizes well :

$$R_{\mathcal{D}}{}^{\ell}(h_\theta) = \mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}}[\ell(h_\theta(x), y)]$$

   "Learning" consists of finding a good hypothesis function $h_\theta \in \mathcal{H}$ that captures in the best possible way the relationship between X and Y.

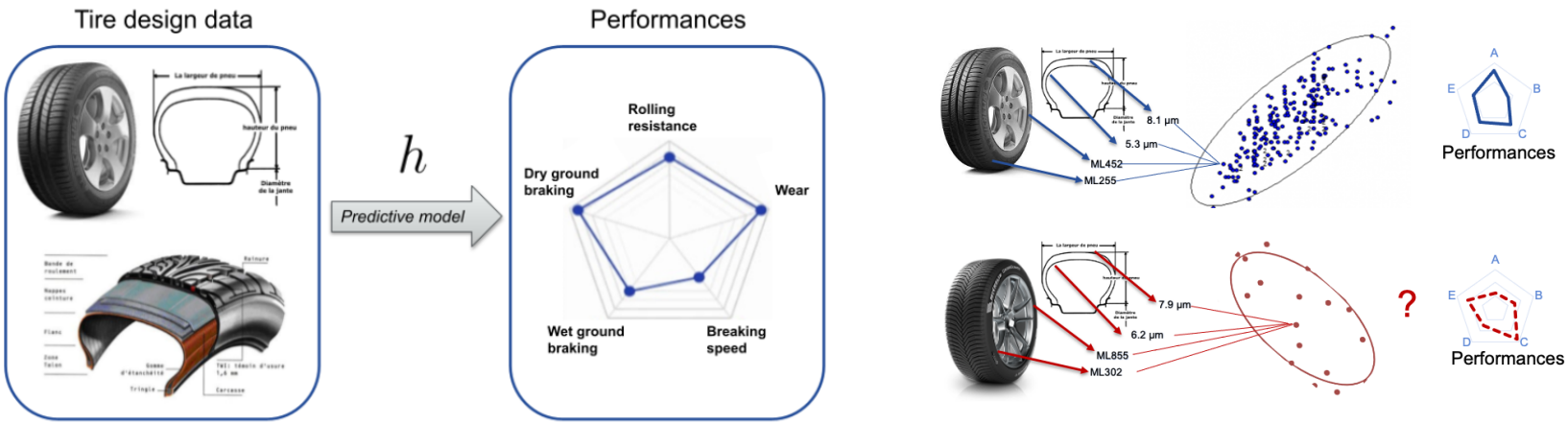
$$h_{\theta_{\text{opt}}} = \arg\min_{h_\theta} R_{\mathcal{D}}{}^{\ell}(h_\theta)$$

5. Optimization Procedure, hyperparameters...

Industrial motivations

Conception : product design.

- New products are regularly manufactured with a long and costly development.
- Relative small data sets are gathered during the development of products such as characteristics (X : color, shape, weight...) and performances (Y).



- Is-it possible to predict the performances (Y) of a new tire line given data previously gathered from **other** lines ? [de Mathelin, 2024]

"Usage" : ML for Automatic tire wear detection

Industrial objectives : Design an application to

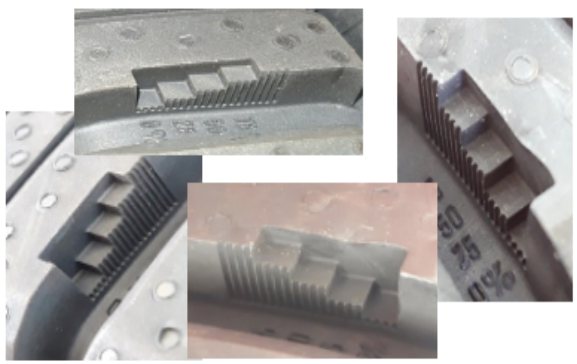
1. Detect and localize a "new generation" wear indicator in a tire image (X).
2. Estimate the wear level (Y)

Data base :
Gathering a **supervised** data base (X,Y) in costly, taking tire pictures (X) in various conditions (views, lighting ,-with and without- wear indicator...) then
evaluating "**by hand**" the wear quality (Y)
(raw information 0-100% , 4 grouped levels);

- **1000 labeled Tires images** were collected :

500 used for learning (shared with the competitors)

- 500 images for blind evaluation.

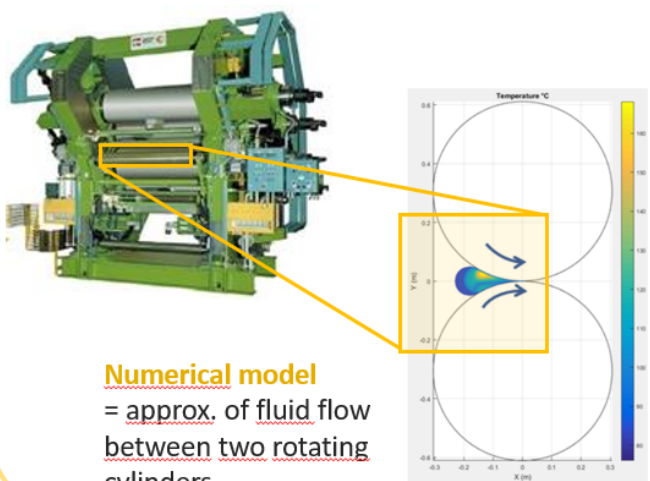


An AI Challenge was organized in 2019 for Startups by the Paris region and ENS Paris-Saclay in collaboration with Michelin company.

Production : decision making process.

- The final quality of the product strongly depends on raw materials (composition/proportion) and fabrication. Fabrication needs to be fully understood, for automatic decision making Process.
- As an illustration, the Calendering process smoothes out the rubber through contra-rotating cylinders. Physical problem : Assure that the rubber has the desired properties.

Example of Michelin's use-case:



Numerical model
= approx. of fluid flow between two rotating cylinders

• The rubber is assimilated as an incompressible non-Newtonian fluid flow.

$$\begin{cases} (2\eta(\vec{u},T)u_x)_x + \left(\eta(\vec{u},T)(u_y+v_x)\right)_y = p_x \\ \left(2\eta(\vec{u},T)v_y\right)_y + \left(\eta(\vec{u},T)(u_y+v_x)\right)_x = p_y \\ u_xT_x + u_yT_y = \dfrac{\lambda}{\rho C_p}(T_{xx}+T_{yy}) + \dfrac{\eta(\vec{u},T)}{\rho C_p}|\gamma(\vec{u})|^2 \\ u_x + v_y = 0 \end{cases}$$

where $\vec{u}$ is the velocity, p pressure and T temperature, $\bar{\bar{\varepsilon}}(\vec{u})$ is the strain rate tensor, $\gamma(\vec{u}) = \sqrt{2\sum \bar{\bar{\varepsilon}}_{i,j}^2}$, and the dynamic viscosity η:

$$\eta(\vec{u},T) = K|\gamma(\vec{u})|^{n-1}\exp\left(\frac{E_\alpha}{R}\left(\frac{1}{T}-\frac{1}{T_\alpha}\right)\right)$$

• Numerical simulations help to understand the process.

Machine Learning in the industry

Main observations :

- Often small, moderate, evolving database. Ex. manufacturing process.
- Few or not labeled data. Ex. Few production defaults.
- labeled-data is often difficult and time-consuming to acquire.
  Ex. Experimental design to help selecting costly observation outputs.
- In many real-world applications, historical (training) data and newly collected (test) data may often exhibit different statistical characteristics.
  However, in many ML scenarios, training and test samples are supposed to be generated by the same (unknown) probability distribution.
- Still Strong Needs for monitoring and diagnosis based on machine learning (ML) .
- Makes sense to re-use knowledge gained from related but distinct datasets, from physics, from conception.....

Knowledge in ML modeling

Extra Prior knowledge can provide rich information not existing or hard to extract in limited training data and helps improve the ability to generalize, and the plausibility of resulting models.

- Data knowledge .
  1. **Augmentation.** Easy for Image classification tasks (symmetry, rotation...).
  2. **Feature engineering.** $x_{raw} \to x \to f_\theta(x) \sim y$
     Ex. Wavelet based scattering transform, Fourier transform.
     Ex : sounds classification for Delphin challenge classification (frequential data).
  3. **Foundation models.** Pre-trained models on alternative large data base, use of TL.

- Model Tailored knowledge [Features, architecture, function properties]
  1. **Design of specialized NN architecture** associated with a given predictive task. Symmetry groups as rotation, homothety, translation may implement an intrinsic geometry of f_θ
     $x \to f_\theta(x) \sim y$
     Ex : Convolutional NN [CNN] by crafty respecting **invariance** along the groups of symmetries. Pre-calibrated models (NN warm start).
  2. **Multi-Task Learning**
     Introduction of knowledge/ constraints in the cost function, in the optimisation process. Ex : *Physical Informed Neural Network*

Towards Transfer Learning, Domain adaptation, & Physics Informed Machine Learning..

Outline

Machine Learning models

&

Transfer learning

Learning a Machine.

ML models aim to provide models with a good generalization capability on a Target domain.
Learning data, vs Target data/same domains, Target data/different domains.

Target domain (ex I).

Same Domain $\{\mathcal{X}, P(X)\}$ &
task $\mathcal{T} = \{\mathcal{Y}, P(Y|X)\}$.

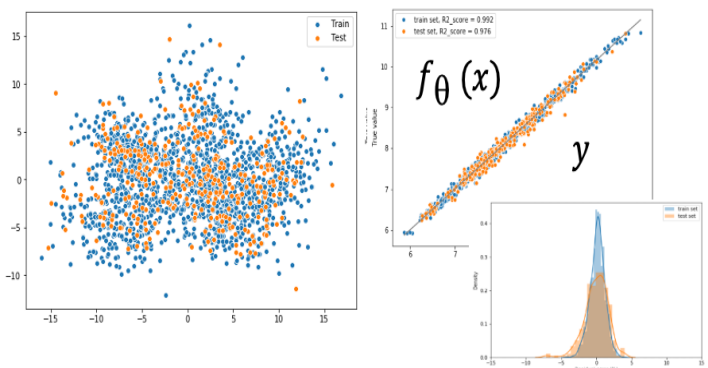


FIGURE – High Prediction capability.

Target domain (ex II).

Different Domain & same task.

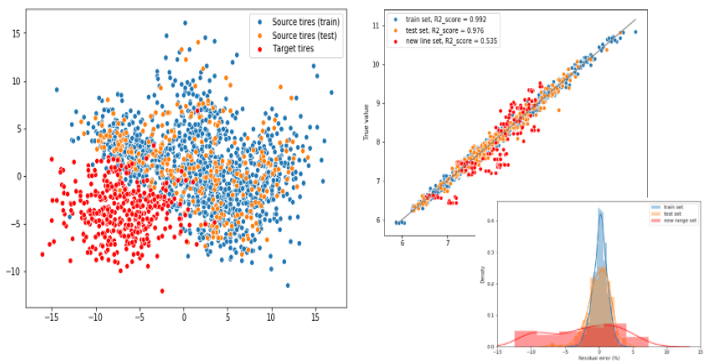


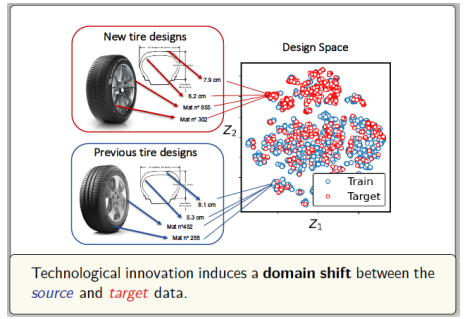
FIGURE – Low Prediction capability.

$P(x, y)$	=	$P(y/x)$	*	$P(x)$
Joint distribution differences		concept shift		covariate-shit

The Transfer learning framework

- **Data collections : Source & Target**
  1. **Source data** $\mathcal{S}$.
     $X_{\mathcal{S}} \otimes Y_{\mathcal{S}}$ the source input and output spaces associated with $\mathcal{S}$
     $\mathcal{S}_X$ the marginal distribution of $X_{\mathcal{S}}$, $t_{\mathcal{S}}$ the source learning task
  2. Target data $\mathcal{T}$
     $X_{\mathcal{T}} \otimes Y_{\mathcal{T}}$ the Target input and output spaces associated with $\mathcal{T}$
     $\mathcal{T}_X$ the marginal distribution of $X_{\mathcal{T}}$, $t_{\mathcal{T}}$ the Target learning task

  Δ Source and Target data are not drawn from the same distribution.



Technological innovation induces a **domain shift** between the *source* and *target* data.

- **Focus** on the Target Risk. $R_{\mathcal{T}}^{\ell}(h) = \mathbb{E}_{(x,y)\sim\mathcal{T}}[\ell(h(x), y)]$ with ℓ the loss function.
- **Supervised data or calibrated Model** available for the source domain (enough data).

Transfer learning aims to improve the learning of the target predictive function :
$f_{\mathcal{T}} : X_{\mathcal{T}} \to Y_{\mathcal{T}}$ for $t_{\mathcal{T}}$ using knowledge gained from $\mathcal{S}$ where $\mathcal{S} \neq \mathcal{T}$

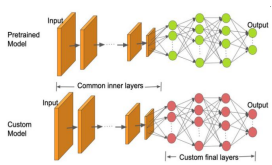
$\mathcal{S} \neq \mathcal{T}$ (joint distributions) implies several cases :

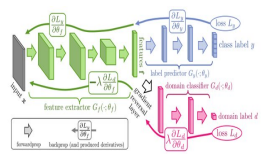
- $\mathcal{S}_{\mathcal{X}} \neq \mathcal{T}_{\mathcal{X}}$ i.e. $X_{\mathcal{S}} \neq X_{\mathcal{T}}$ (spaces) or $\mathcal{S}_X(X) \neq \mathcal{T}_X(X)$ (laws) or
- $t_{\mathcal{X}} \neq t_{\mathcal{T}}$ (i.e. $Y_{\mathcal{S}} \neq Y_{\mathcal{T}}$ (target task) or $\mathcal{S}(Y/X) \neq \mathcal{T}(Y/X)$ (conditional law)

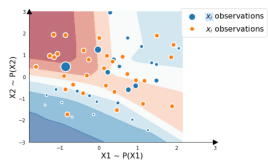
... Seems to be a hard problem...
Success stories ?... Theoretical guaranties ? Assumptions ?, Negative transfer ? ....
Answers to the industrial partners.... open source algorithms...

Transfer Learning & Domain adaptation Methods

- **Several approaches to transfer knowledge from Source to Target domain.**
  - Model-based. Transfer the model parameters learnt on the source data to the target model.
    - Train model available, not necessary the source data- .
    Ex. Image based tire wear estimation based on Deep architecture (Michelin) (Resnet...), Automatic fall detection based on decision trees/ RF (Tarkett).
  - Feature-based. Find a new representation space to bring feature spaces closer.
    -Source and Target Input data available-. *Ex. Domain adversarial neural networks (EDF, Michelin)*
  - Instance-based. Re-weight the source samples to bring the distributions closer.
    -Source and Target Input data available-.
    Ex. Multi-source domain adaptations for Product design (Michelin) or Electricity prediction (EDF)







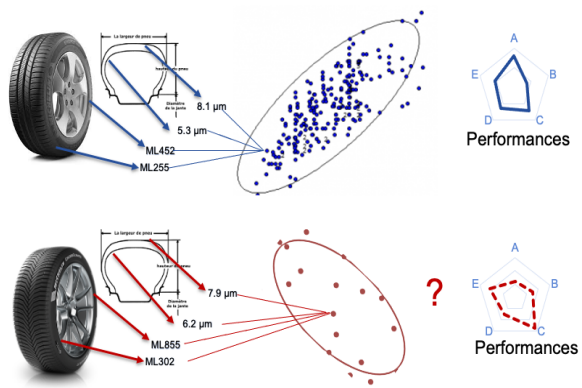
- **Theoretical guarantees on the Target Risk given the Source Risk and discrepancy**

$$\mathbf{R}_T(h) \leq \mathbf{R}_S(h, h_S^*) + \text{disc}_{\mathcal{H},\ell}(S, T) + \lambda$$

where $R_Q(h) = \mathbb{E}_Q(\ell(h(X), Y))$,
$\text{disc}_{\mathcal{H},L}(S, T) = \sup_{h,h' \in \mathcal{H}} \left| L_S(h, h') - L_T(h, h') \right|$, $\quad L_Q(h, h') = \mathbb{E}_Q(\ell(h(x), h'(x)))$, $h, h' \in \mathcal{H}$
$h_S^* = \arg\min_{h \in \mathbf{H}} \mathbf{R}_S(h)$, $\quad h_T^* = \arg\min_{h \in \mathcal{H}} \mathbf{R}_T(h)$, $\quad \lambda = \mathbf{R}_S(h_T^*) + \mathbf{L}_T(h_S^*, h_T^*)$

Transfer Learning & Domain adaptation Methods

- **Model-based.**
- **Feature-based.**
- Instance-based. Re-weight the source samples to bring the distributions closer.
  -Source and Target Input data available-.
  Ex. Multi-source domain adaptations for Product design (Michelin) or Electricity prediction (EDF)



K	10	50	100	200
Target-Only	8.37 (1.16)	4.76 (0.43)	3.63 (0.25)	2.81 (0.16)
Uniform	2.85 (0.10)	2.58 (0.08)	2.42 (0.07)	2.19 (0.09)
Balanced	3.12 (0.56)	2.60 (0.15)	2.32 (0.11)	2.21 (0.13)
WANN	**2.61 (0.10)**	**2.47 (0.15)**	**2.27 (0.10)**	**2.09 (0.08)**

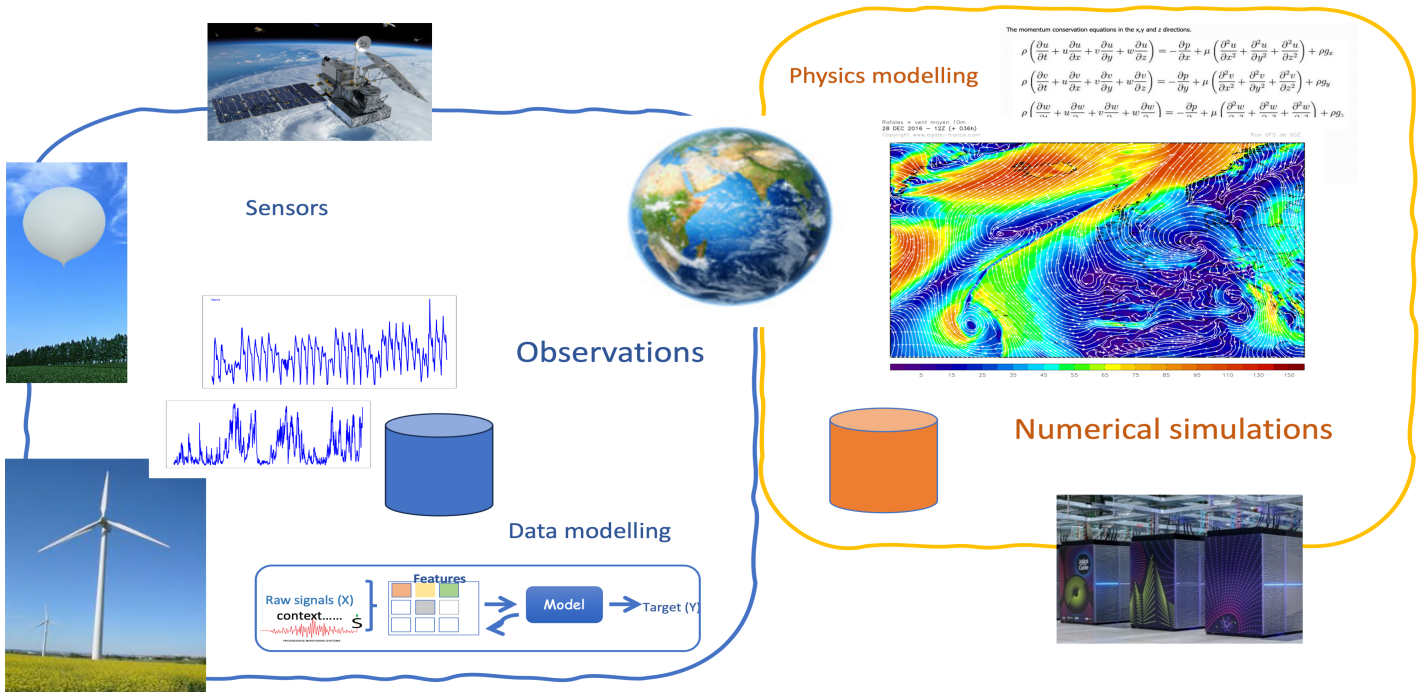
MAPE comparison for different K values.

FIGURE – Importance Weighting algorithm for the tire design use-case where source and target data respectively correspond to the observations recorded before and after 2018.
All methods are provided with the source dataset, augmented with the few target labeled data : $\mathcal{S} \cup \mathcal{T}^K$

[de Mathelin, 2024].

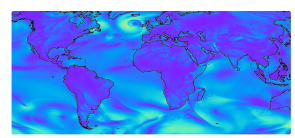
Machine Learning models for physics

Modeling. Observation data and Numerical simulations



▶ Surrogate Models. Meta Model. Machine Learning models and numerical simulations. (speed..)

▶ GraphCast. The state-of-the-art model delivers 10-day weather predictions at unprecedented accuracy in under one minute based on 39 years (1979–2017) of historical data from ECMWF's ERA5 (21) reanalysis archive.

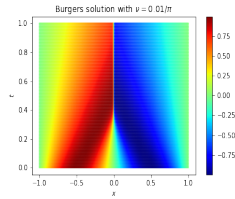


ML surrogate model

Illustration with Deep Neural Network on a toy example.

Burger's equation :input (x space , t *time*), output : speed of the fluid
given parameter : ν viscosity. Dirichlet boundary condition :

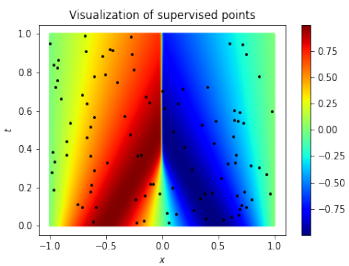
$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} - \nu\frac{\partial^2 u}{\partial x^2} = 0; u(0,x) = -sin(\pi x); u(t,-1) = u(t,1) = 0$$



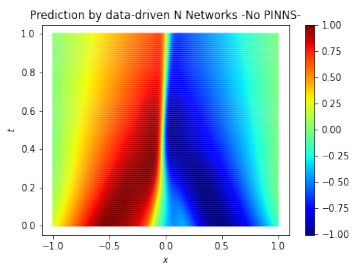
Neural network Surrogate model.

f_θ trained with $n = 100/1000$ supervised observations.
ML model : NN architecture : 2-4 (x50)-1.
Evaluation on a **Test data set**, regular grid of points (256 (x) , 100 (t)).
$E(u,\hat{u}) = ||u-\hat{u}||_2^2/||u||_2^2$ on a grid (256 (x) , 100 (t))

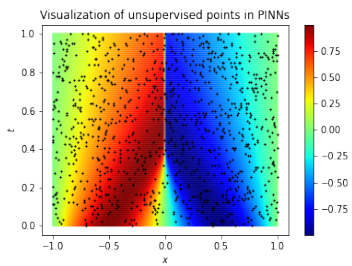
Supervised points



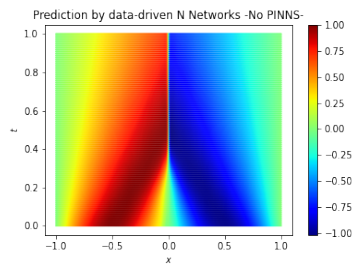
Surrogate model $E(u,\hat{u}) = 0.17$



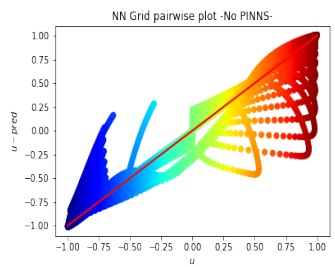
Supervised points



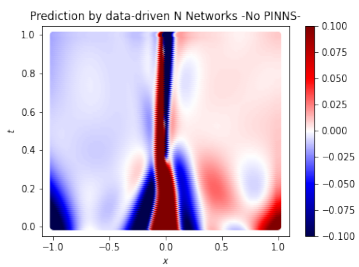
Surrogate model $E(y,\hat{y}) = 0.02$



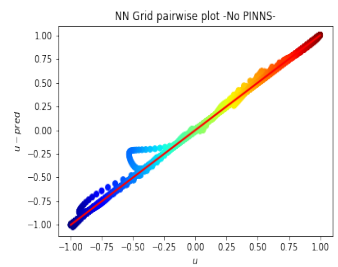
Pairwise Plot



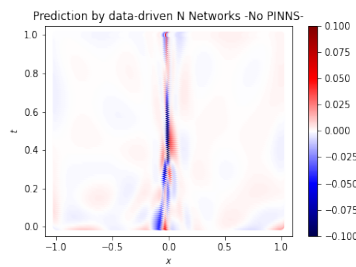
Error



Pairwise Plot



Error

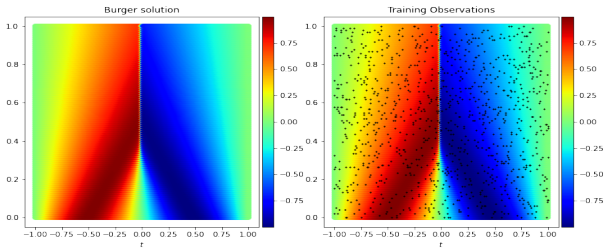


ML Surrogate model

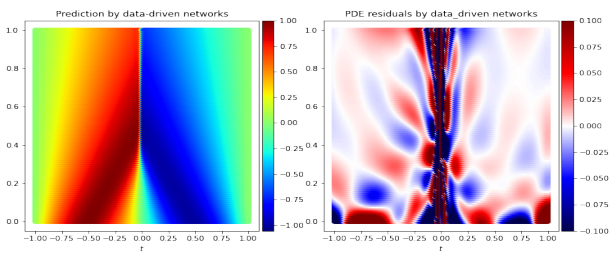
Illustration with Deep Neural Network on a toy example.
Focus on the Pde errors/residuals computed for Burger model (right/bottom figure) :

$$R = \frac{\partial \hat{u}(x,t)}{\partial t} + \hat{u}(x,t)\frac{\partial \hat{u}(x,t)}{\partial x} - \nu \frac{\partial^2 \hat{u}(x,t)}{\partial x^2} \ ?= 0\,;$$

$N_{\text{data}} = 1000$, $N_{\text{colloc}=0}$, NN architecture : 2-4 (x50)-1. 50 000 epoch, Adam optimizer.
$E(u, \hat{u}) = ||u - \hat{u}||_2^2 / ||u||_2^2$ on a grid (256 (x) , 100 (t))



PINNs model $E(u, \hat{u}) = 0.0163$



$\rightarrow$The pde constraints are not respected...
$\rightarrow$ The NN model mimics the input/output relation
but the underlaying physics is not caught.

- Especially, in a "small data regime", the vast majority of state-of-the-art machine learning techniques are lacking robustness.
- The cost of data acquisition may be prohibitive.
  Experimental design are proposed to chose the observations.
- ML fail to "model" the underlaying physics phenomena (pde constraints not respected).
- We are inevitably faced with the challenge of drawings conclusions and making decision under partial information.

Physics Informed Neural Networks

PINNs are neural networks trained to solve supervised learning tasks while respecting any given laws of physics described by general nonlinear partial differential equations. [Raissi et al., 2019].

Key ingredients :

1. A deep neural network model : $h_\theta \in \mathcal{H}$

2. A supervised data set. $\mathcal{D}_n = \{(X_i, y_i)\ i = 1..n,\ X \in \mathcal{X}, y \in \mathcal{Y}\}$ the training data (initial and boundary observations).

$$\boxed{\text{MSE}_{\text{data}} = \frac{1}{n}\sum_{i=1}^{n}[h_\theta(X_i) - y_i]^2}$$

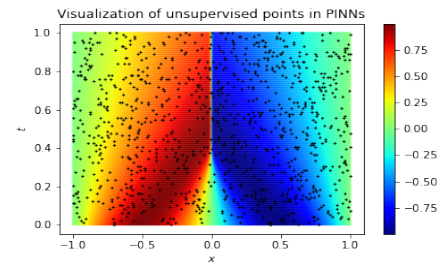
3. **Equations*** describing the physics.
Ex : Burger equation : Input : $X = (x, t)$ output : u :

$$R(u, t, x, \nu) = \frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} - \nu\frac{\partial^2 u}{\partial x^2} = 0$$

$$\boxed{\text{MSE}_{\text{pde}} = \frac{1}{n}\sum_{i=1}^{\text{Colloc}} R(u_i, t_i, X_i, \nu)}\ \textbf{, No mesh !}$$

pde residuals cost on random collocation points.

4. **A appropriate loss function** $\mathcal{L}(h_\theta, \mathcal{D}_n, \text{physics equations})$.



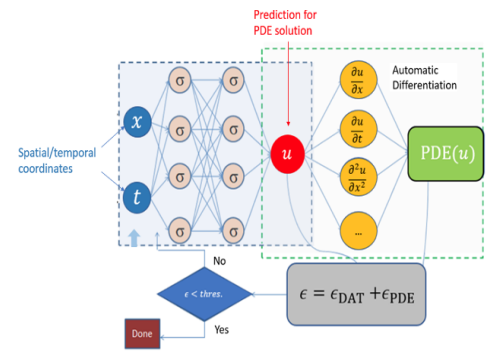


FIGURE – [Cuomo et al., 2022]

Shared parameters θ of the NN model $h_\theta(t, x)$ and $R_\theta^{\text{pde}}(t, x)$ are "learned" by minimizing : $\boxed{\text{MSE} = w_{\text{data}}\text{MSE}_{\text{data}} + w_{\text{pde}}\text{MSE}_{\text{pde}}}$

Physical Informed Neural Netwotks

Open questions with an important impact for PINNs models, September 2020

- Supervised data. Observations (number, localization, boundary or inside the domain).
- Unsupervised data. Collocation points : number, localization, on a mesh grid, at random ...
- Trade-off between the two data and residual pde losses : $\text{MSE} = w_{\text{data}}\text{MSE}_{\text{data}} + w_{\text{pde}}\text{MSE}_{\text{pde}}$
- PDE with given parameters (ν) are easily learned by PINNs.
  What about parameterized PDE ?
- Impact of the geometry of the domain (design) ?

Industrial PINNs Applications

- Surrogate modeling. Data-driven solution of pde. Fluid mechanics [Raissi et al., 2019],... , Solid mechanics,...
  Application to rubber calendering process (non-Newtonian fluid thermo-mechanical problems) [Nguyen et al., 2022]
- Identification of unknown parameters. Data discovery of pde. [Raissi et al., 2019], [Nguyen et al., 2022]
- Inverse problems.

Khoa Nguyen, PhD thesis, September 20^{th} 2024.
Development and assessment of physics-informed deep learning methods : towards mutliphysics simulation in industrial contexts.

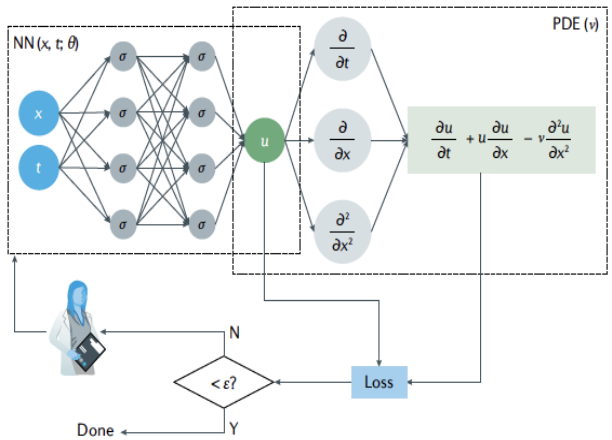
Outline

Physics Informed Neural Netwotks

[Raissi et al., 2019]

$f_\theta(t,x)$ is defined by : $f_\theta := u_t + \mathcal{N}[u]$

- $u(t,x)$ is approximated by a deep NN.
- Automatic differentiation helps to minimize $R(t,x)$ applying the chain rule for differentiating compositions of functions. Pytorch, TensorFlow



The shared parameters θ between the NN $f_\theta(t,x)$ and $R_\theta^{\text{pde}}(t,x)$ can be learned by minimizing the mean squared error.

$$\text{MSE} = w_{\text{data}}\text{MSE}_{\text{data}} + w_{\text{pde}}\text{MSE}_{\text{pde}}$$

- $\text{MSE}_{\text{data}} = \frac{1}{N_{\text{data}}}\sum_{i=1}^{N_{\text{data}}}[f_\theta(t_u^i, x_u^i) - u^i]^2$ for chosen training data (initial and boundary).
- $\text{MSE}_{\text{pde}} =$ is the **cost of pde residuals on random collocation points**. **No mesh !**

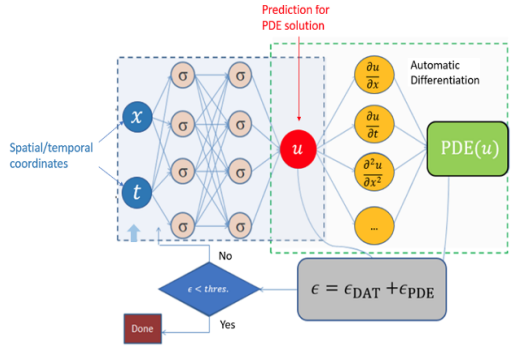
Supervised and Unsupervised Smart Mix

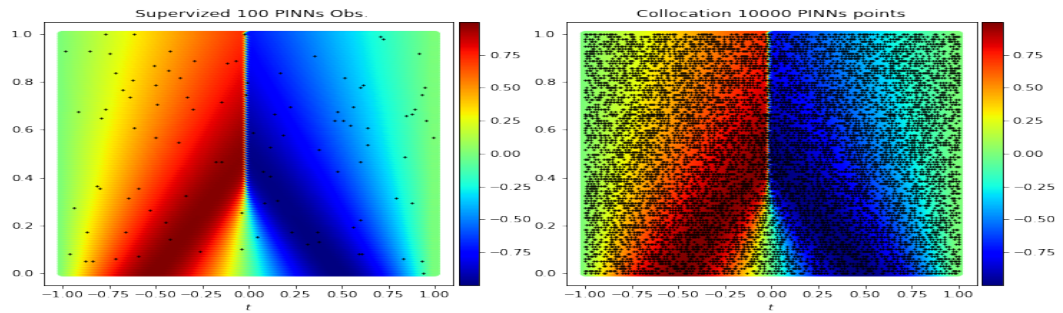
PINNs models

Importance of the Training observations.
Impact on the Pde errors computed for Burger model.

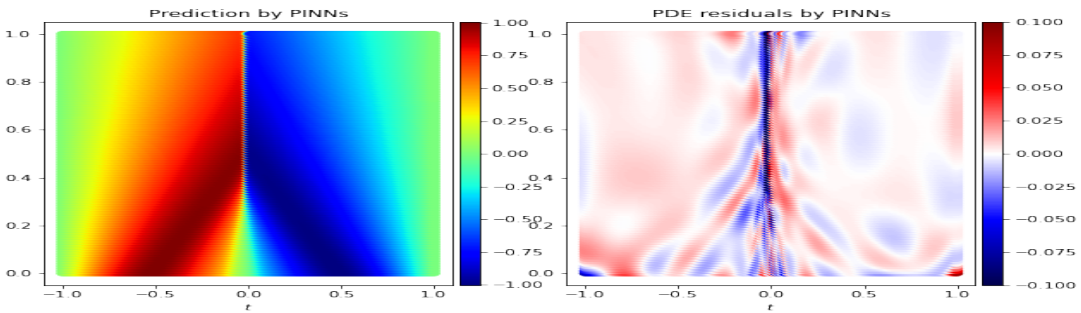
$N_{\text{data}} = 100$, $N_{\text{colloc}=10000}$,

ML model. NN architecture : 2-4 (x50)-1. 50 000 epoch, Adam optimizer.





PINNs model $E(u, \hat{u}) = 0.0157$

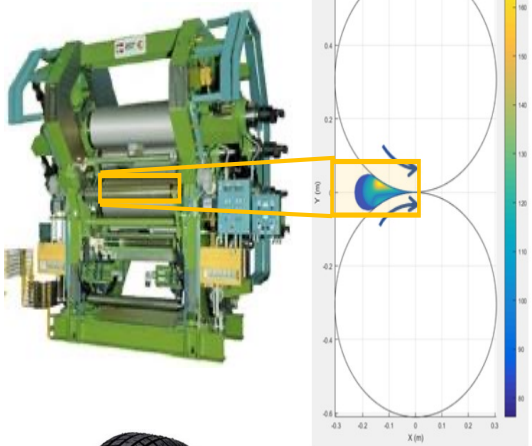


Motivation : Operationnal Use cases & Pinns

Context: Two use-cases...

... And common equations

MICHELIN

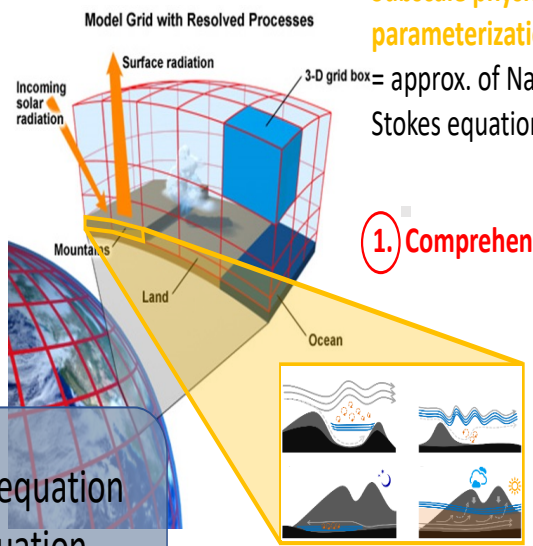


2. Decision making
Validate a fabrication decision

3. Conception
Create new products

cea

Subscale physical parameterizations
= approx. of Navier-Stokes equations



1. Comprehension

Momentum equation
Energy equation
Mass equation

High computational cost

Interest in **Machine learning**

3

Sampling methods for collocation points in PINNS

In the PINNS framework, PDE residuals are computed and minimized on a set of collocation points.

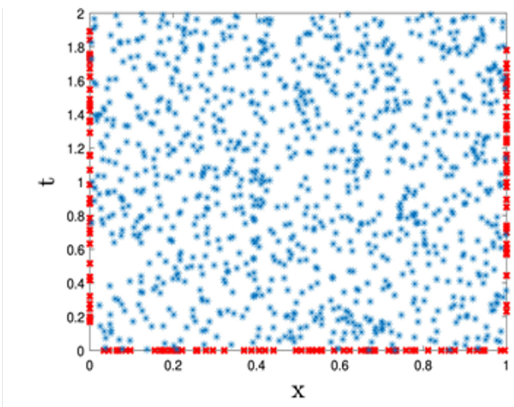
Various sampling strategies have been proposed for these training points, which can be divided into two main sub-classes :

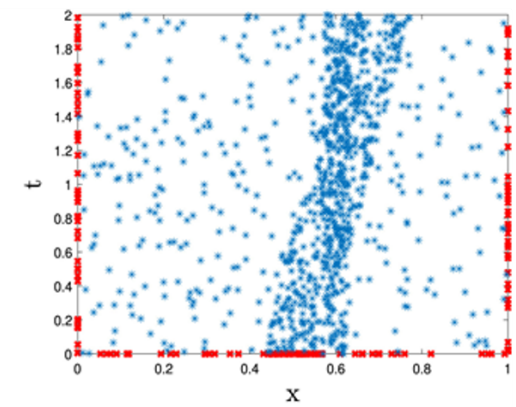
- the non-adaptive sampling approaches.
  - The collocation points are often uniformly and randomly distributed, as in the original work of [Raissi et al., 2019] that employed the Latin Hypercude Sampling (LHS), or
  - manually and non-uniformly based on prior knowledge of the physical phenomena, as in the work of [Mao et al., 2020] and [Nguyen et al., 2022] which is however problem dependent.
- the adaptive sampling approaches.

Non adaptive Sampling methods

Vanilla PINNS with randomly distributed collocation points often fail to predict accurately the PDE solution in complex problems. If one dispose of **prior Physical knowledge** (discontinuity, coupled physics, interest zone, expert guided). Properly chosen locations can increase significantly the accuracy.

- Manual sampling in PINNs [Mao et al., 2020]. Error : left : $O(10^{-2})$; right : $O(10^{-5})$



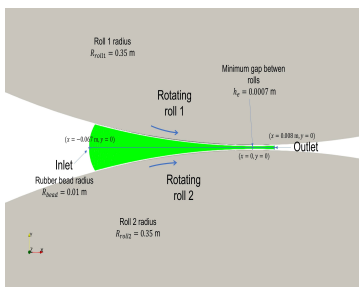


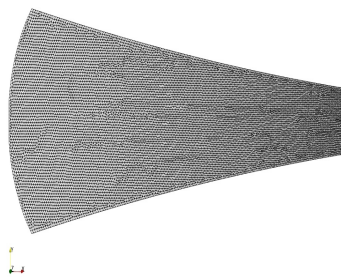
Non adaptive Sampling methods

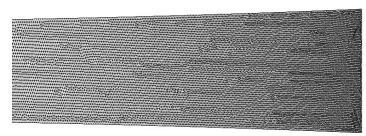
- Prior Mesh knowledge based Information [Nguyen et al., 2022]

The non-adaptive sampling approaches are widely used in PINNS literature : straightforward and efficient when dealing with various PDS problems, including PDEs providing that expert physical knowledge is available.

Illustration : calendar simulation process and pre-defined mesh grid.







→ However, physical knowledge is not always available...

Adaptive Sampling methods

Several adaptive collocation approaches have been developed.

- Residual-based Adaptive Refinement (RAR) approach, first introduced by [Lu et al., 2021], adds new training collocation points to the location where the PDE residual errors are large. Very efficient in enhancing the accuracy of the prediction in many complexe PDE problems.

  $\rightarrow$ leads to an uncontrollable amount of collocation points (∞) and computational cost at the end of the training process.

- Residual-based Adaptive Distribution (RAD) approach, introduced by [Wu et al., 2023]. computes/estimates a probability density function based on PDE residuals, and then resamples or adds new collocation points according to this pdf.

  $\rightarrow$ strong benefits compared to vanilla PINNS in both forward and inverse tasks. Generalizes several previous adaptive approaches [Peng et al., 2022], ...

- Residual-based Adaptive Refinement with Distribution (RAR-D) approach.
  After a certain number of iterations, m points are added to the set of training collocation points.

Adaptive Sampling methods

The Probability Density Function (PDF) $p(x)$, which is based on the PDE residuals, is defined as follows :

$$p(x) \propto \frac{\epsilon^k(x)}{\mathbb{E}[\epsilon^k(x)]} + c$$

where $\epsilon(x)$ denotes the PDE residual for any point x, $k \geq 0$ and $c \geq 0$ are two hyperparameters. Introduced by [Wu et al., 2023] :

- Residual-based Adaptive Distribution (RAD) : after a certain number of iterations, the collocation points are randomly resampled according to the PDF previously defined.
- Residual-based Adaptive Refinement with Distribution (RAR-D) : after a certain number of iterations, m points are randomly sampled according to the PDF previously defined. Then these points are added to the set of training collocation points.
- $\rightarrow$ only RAD allows the number of training points to be fixed during the training, while in RAR-D, this number gradually increases and may lead to higher computational costs.

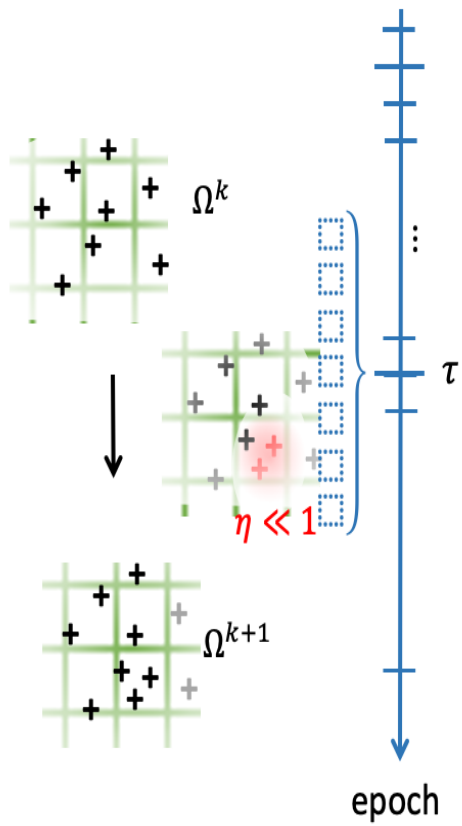
The performance of RAD and RAR-D also depends on the period of resampling method.
k and c hyperparameters of RAD and RAR-D, need to be adjusted.

Adaptive Sampling methods

Fixed-Budget Online Adaptive Learning (FBOAL) [Nguyen, 2024]
Principle : The domain is decomposed into different sub-domains. The collocation set is divided equally for each equation.

Notations : Ω_j : collocation set for eq. j. L^j_{pde} : PDE res. of eq. j.
x_{ji} : collocation points in Ω_j at loc. i. x'_{ji} : new collocation points.

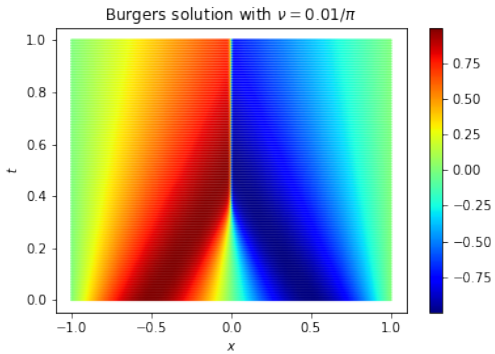
- The x_{ji} are distributed so as to maximize L^{ji}_{pde} i.e. Ω_j is updated by adding n new points x'_{ji}.
  Typically $n = \eta N_j$ $\eta \leq 1$ that yield larger residuals L^{ji}_{pde}.
  $\forall i \leq n \qquad L^j_{pde}(x'_{ji}) \geq \min_{x \in \Omega^k_j} L^j_{pde}(x)$
  Update rule : $\Omega^{k+1}_j = \{x'_{ji}\}_{i \leq n} \cup \Omega^k_j \setminus \{x_{ji}\}_{i \geq n+1}$
- The $N_j - n$ points of Ω^k_j with smallest residuals are discarded
- Updating is applied during the training process every τ epochs, which introduces two new parameters : η and τ.



Impact on collocation points/ Precision.

Burgers equation.

$$\begin{cases} u_t + uu_x - \nu u_{xx} = 0 \quad \text{for } x \in [-1, 1], t \in [0, 1] \\ u(x, 0) = -\sin(\pi x) \\ u(-1, t) = u(1, t) = 0 \end{cases}$$



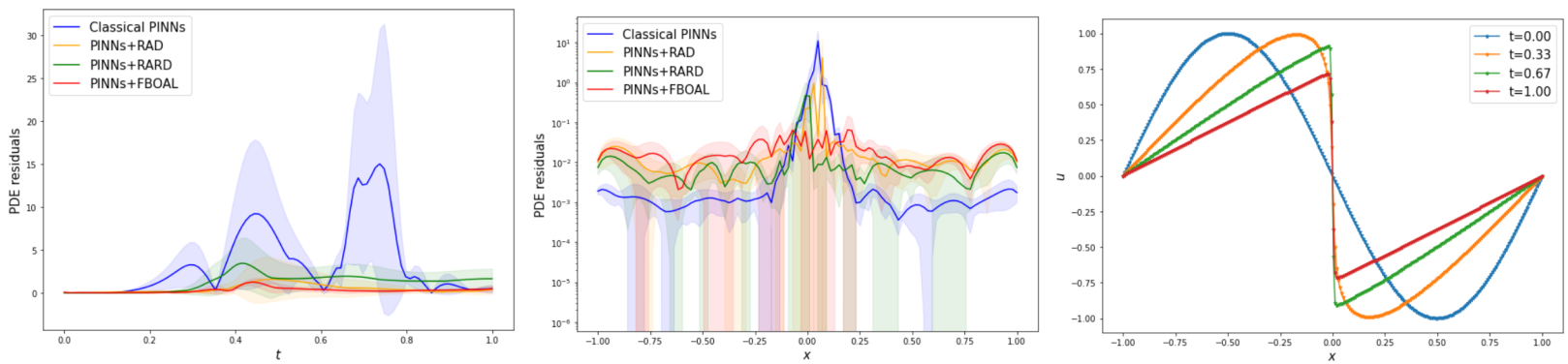
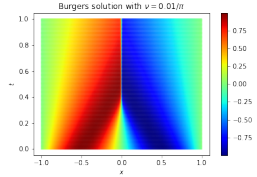


FIGURE – *Burgers equation : Absolute value of PDE residuals for* $\nu = 0.0025$ *after the training process for different approaches.* The curves and shaded regions represent the geometric mean and one standard deviation of five runs. On the line $x = 0$ (left), At instant $t = 1$ (right).

Impact on collocation points/ feed-back on the density

Burgers equation. ML Model $u(x, t, \nu = \nu_0)$

$$\begin{cases} u_t + uu_x - \nu u_{xx} = 0 \quad \text{for } x \in [-1, 1], t \in [0, 1] \\ u(x, 0) = -\sin(\pi x) \\ u(-1, t) = u(1, t) = 0 \end{cases}$$



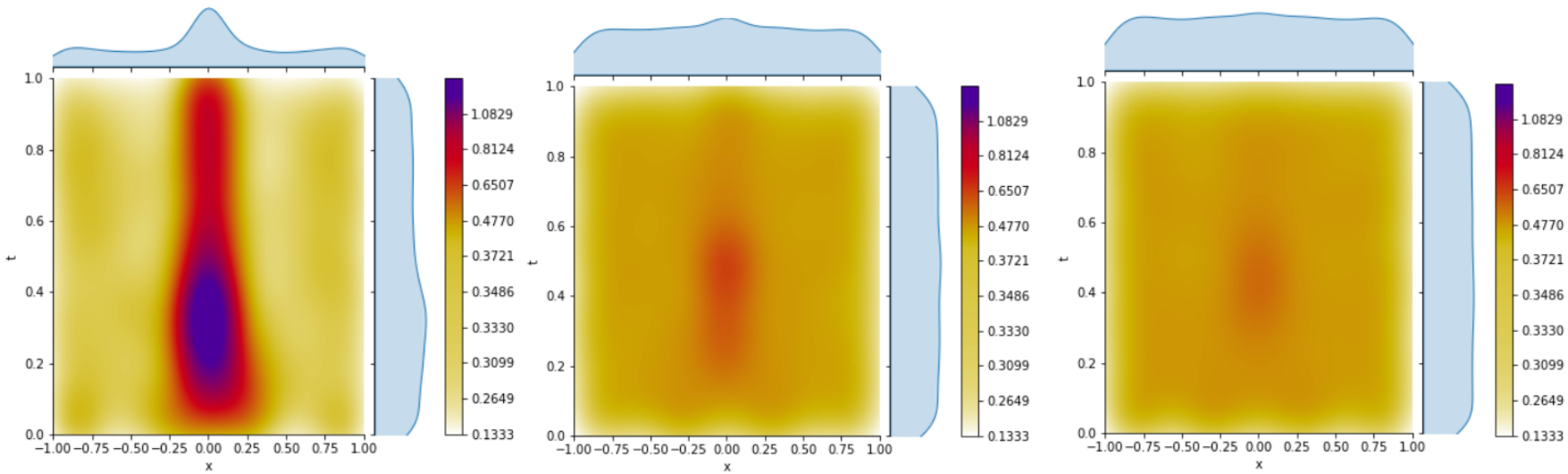
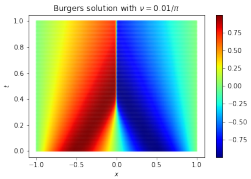


FIGURE – *Burgers equation : Density of collocation points after the training with FBOAL.*
$\nu = 0.0025$ (left); $\nu = 0.0076$ (center); $\nu = 0.0116$ (right)

Impact on collocation points/ Training cost benefits

Burgers equation.

$$\begin{cases} u_t + uu_x - \nu u_{xx} = 0 \quad \text{for } x \in [-1, 1], t \in [0, 1] \\ u(x, 0) = -\sin(\pi x) \\ u(-1, t) = u(1, t) = 0 \end{cases}$$



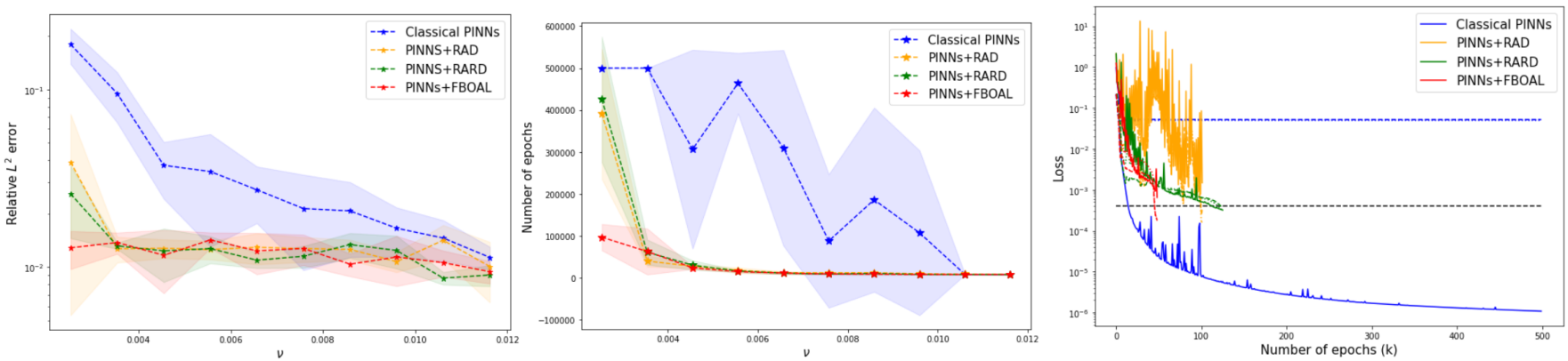


FIGURE – *Burgers equation : Performance of classical PINNs and PINNs with adaptive sampling approaches.* The curves and shaded regions represent the geometric mean and one standard deviation of five runs. In (c) the solid lines show the cost function during the training, the dashed lines show the errors on the testing mesh, and the black line shows the threshold to stop the training. Relative $\mathcal{L}^2$ error (left); Number of training iterations; Loss for $\nu = 0.0025$ (right).

	Classical PINNs	PINNs + RAR-D	PINNs + RAD	PINNs + FBOAL
Training time	33.7 ± 1.5	41.0 ± 5.8	38.8 ± 2.3	$\mathbf{21.5 \pm 2.7}$
Number of resampling	0 ± 0	201 ± 75	210 ± 77	$\mathbf{48 \pm 2}$

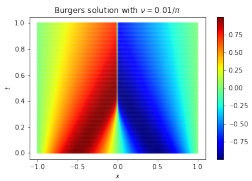
TABLE – *Burgers equation : Training time (in minutes) and the number of resampling for* $\nu = 0.0025$. The training is effectuated on an NVIDIA V100 GPU card.

Impact on collocation points/ ν Adaptation.

Parametrized Burgers equation. ML model $u(x, t, \nu)$

Training, 40 values of $\nu \in [0.0025, 0.0124]$.

$$\begin{cases} u_t + uu_x - \nu u_{xx} = 0 \quad \text{for } x \in [-1, 1], t \in [0, 1] \\ u(x, 0) = -\sin(\pi x) \\ u(-1, t) = u(1, t) = 0 \end{cases}$$



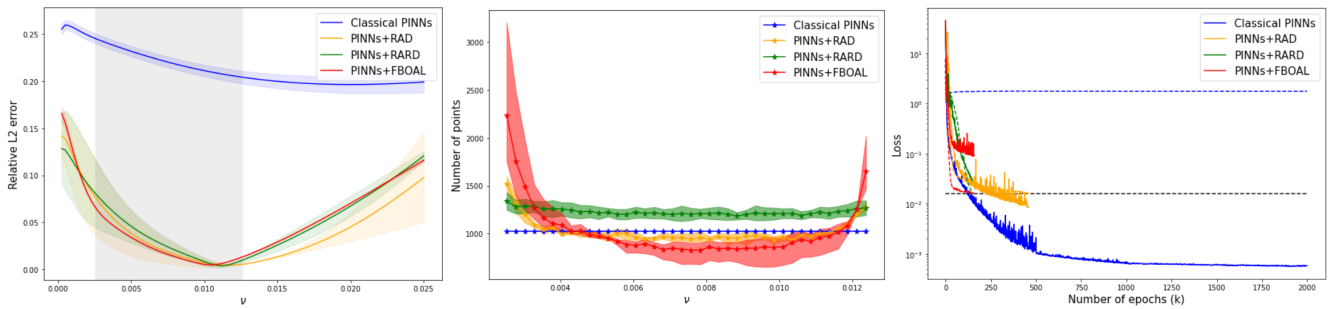


FIGURE – *Parametrized Burgers equation : Comparison of classical PINNs and PINNs with adaptive sampling approaches.* The zone in gray is the learning interval for ν (interpolation zone). The curves and shaded regions represent the geometric mean and one standard deviation of five runs. In (c) the solid lines show the cost function during the training, the dashed lines show the errors on the testing data set, and the black line shows the threshold to stop the training. Relative $\mathcal{L}^2$ error (left); Number of collocation points; Cost function during the training (right).

	Classical PINNs	PINNs + RAR-D	PINNs + RAD	PINNs + FBOAL
Training time	13.2 ± 0.0	$\mathbf{1.4 \pm 0.3}$	9.1 ± 3.5	7.8 ± 1.9
Number of resampling	0 ± 0	$\mathbf{34 \pm 8}$	204 ± 71	173 ± 25

TABLE – *Parametrized Burgers equation : Training time (in hours) and the number of resampling of each methodology. The training is effectuated on an NVIDIA A100 GPU card.*

PINNS Sampling methods

Dont't use PINNs methods without adaptation!

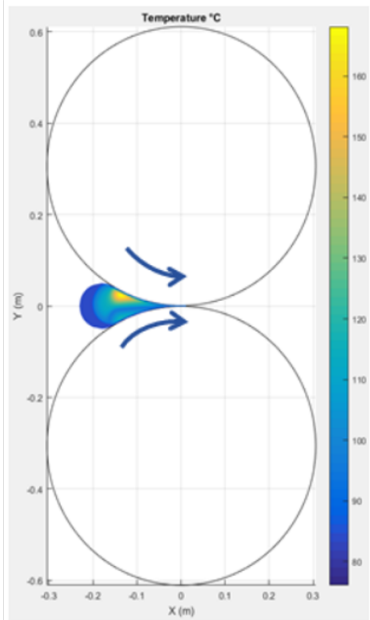
- FBOAL adaptively adds more points to important zones (coupling, discontinuity)
- With appropriate chosen hyperparameters, FBOAL provides better accuracy and faster convergence than vanilla PINNS with random points
- FBOAL can provide prior knowledge of high residuals.
  $\rightarrow$ Help conventional numerical solver in the construction of mesh.

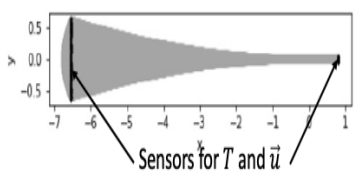
Limitations and Perspectives

- No theoretical results for convergence
- Optimal hyperparameters of FBOAL are problem-dependant
- The decomposition domain step can be further investigated (e.g. high-dimensional)
- The division of set points for multi-physics problems may not be optimal.

Rubber calendering process

- Calendering process : smooth out the rubber through contra-rotating cylinders
- Physical problem : assure that the rubber has the desired properties





Rubber: assimilated as **an incompressible non-Newtonian fluid flow.**

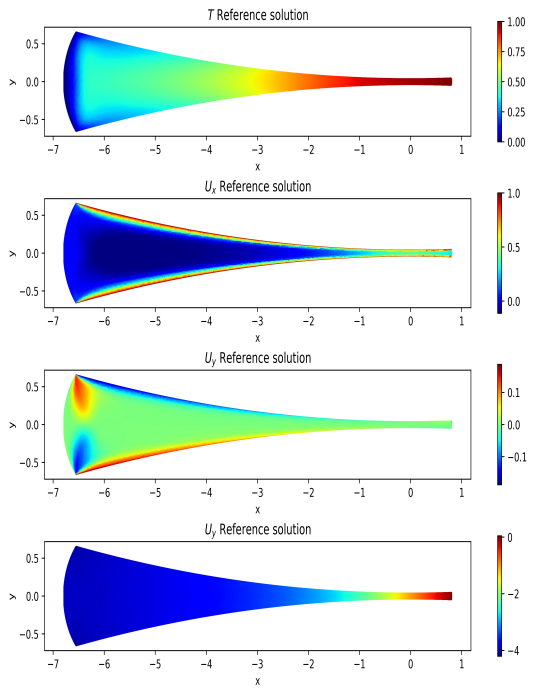
$$\begin{cases} (2\eta(\vec{u},T)u_x)_x + \left(\eta(\vec{u},T)(u_y+v_x)\right)_y = p_x \\ \left(2\eta(\vec{u},T)v_y\right)_y + \left(\eta(\vec{u},T)(u_y+v_x)\right)_x = p_y \\ u_xT_x + u_yT_y = \dfrac{\lambda}{\rho C_p}\left(T_{xx}+T_{yy}\right) + \dfrac{\eta(\vec{u},T)}{\rho C_p}|\gamma(\vec{u})|^2 \\ u_x + v_y = 0 \end{cases}$$

where $\vec{u}$ is the velocity, p pressure and T temperature, $\bar{\bar{\varepsilon}}(\vec{u})$ is the strain rate tensor, $\gamma(\vec{u}) = \sqrt{2\sum \bar{\bar{\varepsilon}}_{i,j}^2}$, and the dynamic viscosity η:

$$\eta(\vec{u},T) = K|\gamma(\vec{u})|^{n-1}\exp\left(\frac{E_\alpha}{R}\left(\frac{1}{T}-\frac{1}{T_\alpha}\right)\right)$$

Industrial Physics models.
Rubber calendering modeling

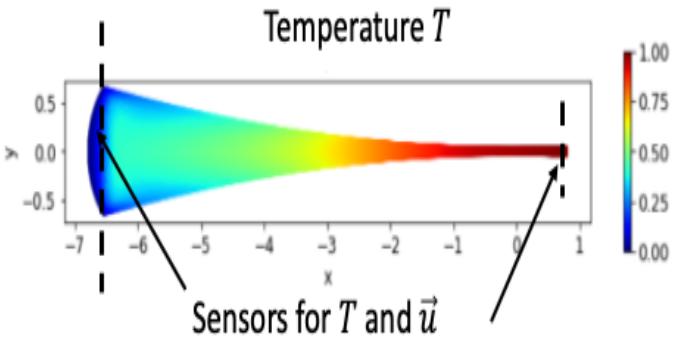
To generate reference High-Fidelity (HF) solutions of velocity, pressure, and temperature fields, the equations are discretized and solved using an in-house generic and multi-purpose finite element solver named MEF++ and co-developed by Laval University and Michelin. [?].

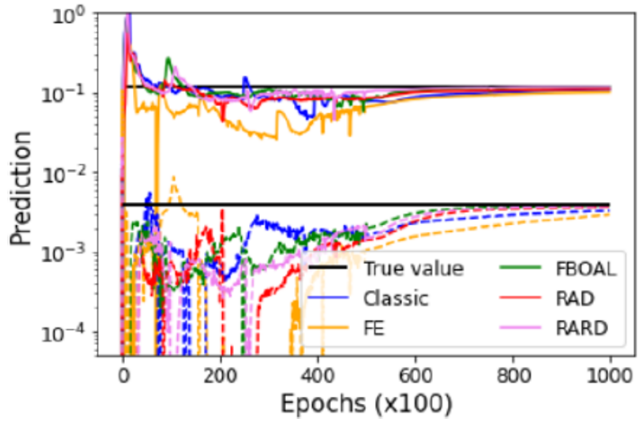


Rubber calendering process - inverse problem

Goal : identify the true value of the thermal conductivity from sensors measurements.
Dimensionless equation :

$$u_x T_x + u_y T_y = \frac{1}{Pe}(T_{xx} + T_{yy}) + \frac{Br}{Pe}\eta(\vec{u}, T)|\gamma(\vec{u})|^2$$



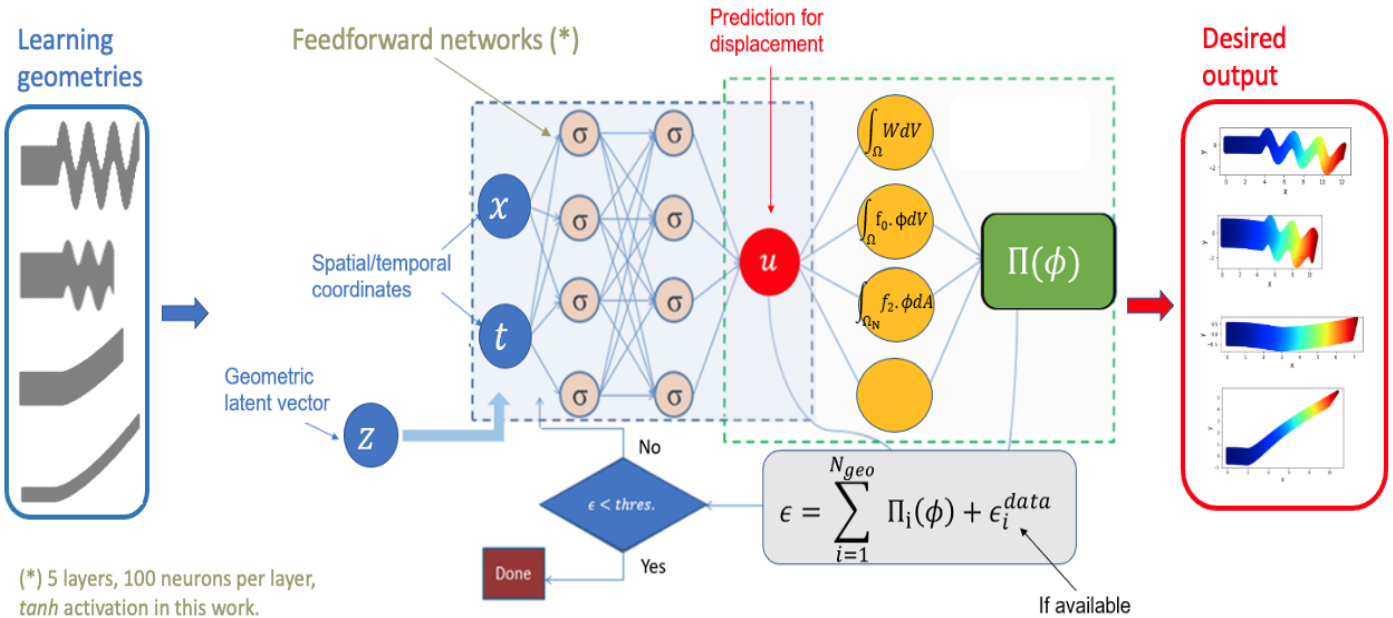


- Using only two lines of sensors, PINNs can identify accurately the true values of unknown parameters.
- Adaptive strategies outperform fixe strategies

Strategy	$\epsilon_{Br/Pe}$	$\epsilon_{1/Pe}$
Random points	6.50	12.9
FE mesh	12.2	24.7
FBOAL	**0.35**	8.05
RAD	0.71	7.00
RAR-D	**0.69**	**0.11**

Geometry Aware Deep Energy Method

- [Nguyen et al., 2024] proposed to encodes the geometric knowledge into the PINNS model and to minimize a loss function on the potential energy of the system and not on residual equations.
- The potential energy of the system is minimum at the equilibrium state and computed by the difference between internal and external energy, here approximated by Monte Carlo.
- The potential energy of the systems over all geometries are minimized.



- Focus is made of mechanical problem using weak formulation.

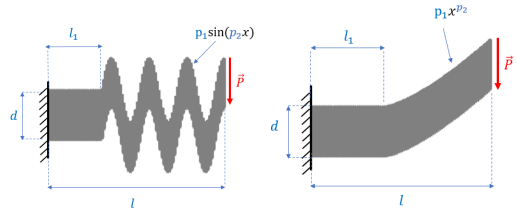
Geometry Aware PIML.

Displacement of a Beam with **Linear Elasticity**.

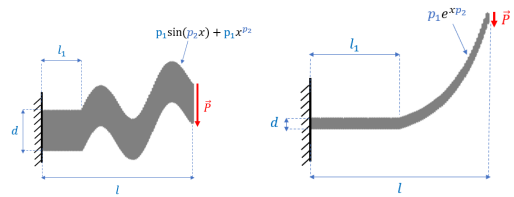
Experimental design.

- 5 parameters encoding the geometry
- LHS to compute 50 geometries
- The left side clamped, right side subjected to a traction $\vec{P} = (0, -1)N$.
- homogeneous, isotropic material with Young's modulus $E = 1000N/m^2$, and Poisson's ratio $\nu = 0.3$.
- Reference solution obtained by Finite Element Method and Fenics software

Ex. Training :

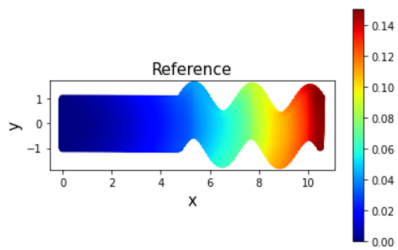


Ex. Testing



Coding the Geometry...

- **Parametric :** explicit parametric encoding.
- **PCA-Coord :** spatial coordinates for the geometric representation and PCA for the encoding (objects' boundaries).
- **VAE-Coord :** spatial coordinates for the geometric representation and VAE for the encoding.
- **PCA-Image :** images for the geometric representation and PCA for the encoding. (objects' images are available.)
- **VAE-Image :** images for the geometric representation and VAE for the encoding.



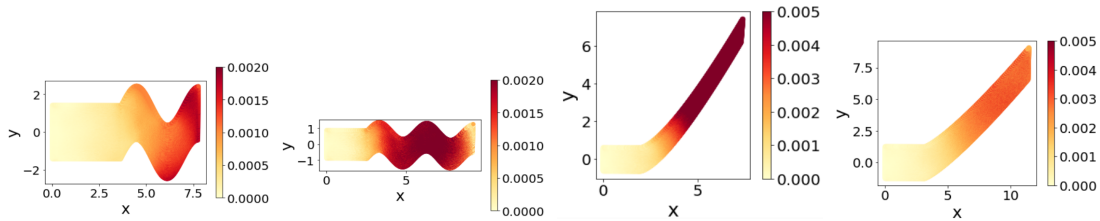
Geometry Aware PIML.

Displacement of a Beam with **Linear Elasticity**.

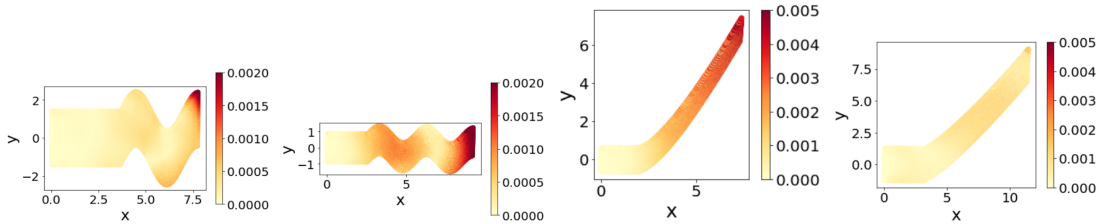
- **Importance of adaptive sampling**

Parametric encoding. Illustration.

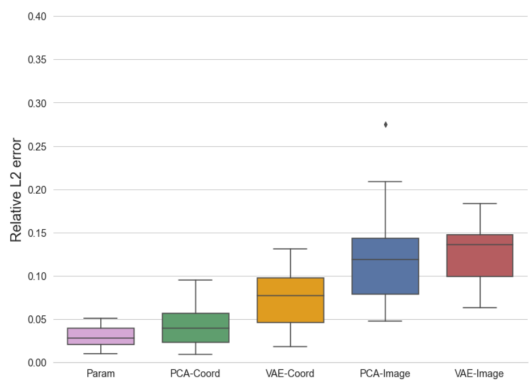
Without FBOAL (absolute error)



With FBOAL(absolute error)



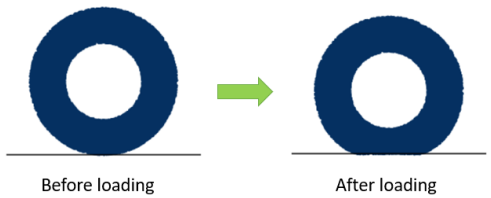
- **Impact of encoding.** Testing data set.



Geometry Aware PIML

Toy tire loading simulation with Hyperelasticity

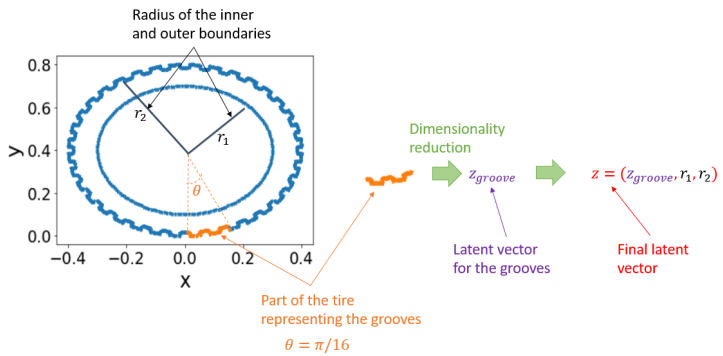
- Model the 2D displacement of toy tires composed of hyperelastic materials (rubber).
- Tires are made of rubber with a Young modulus $E = 21.10^6 Pa$ and a Poisson's ratio $\nu = 0.3$.
- Tires follow the Saint-Venant Kirchhoff hyperelastic model.



Sketch of a tire before and after loading.

About the geometry.

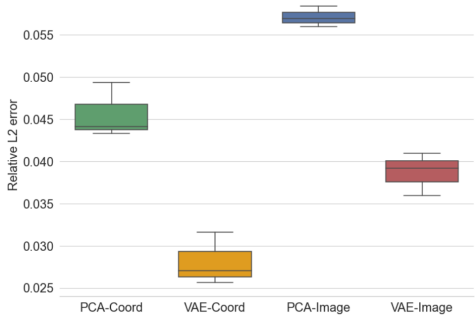
- 55 Tire geometries (5 x 11) given radius of the inner and outer boundaries, and the grooves.
- Different coding schemes : PCA-coord, VAE-coord, PCA-image, VAE-image



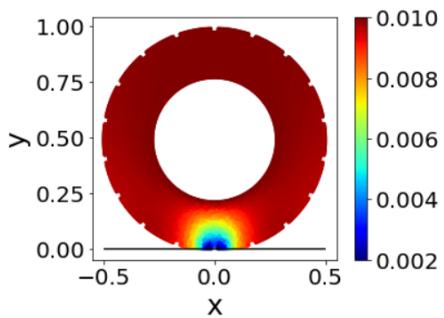
Geometry Aware PIML

Toy tire loading simulation with Hyperelasticity

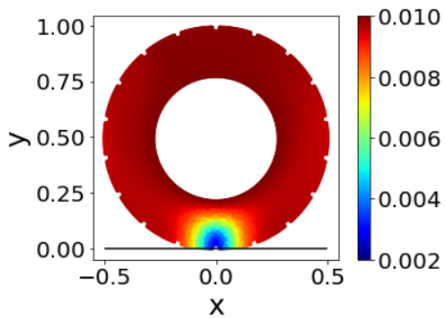
Performance of tire load simulation.
Best VAE-coord encoding (non linear)



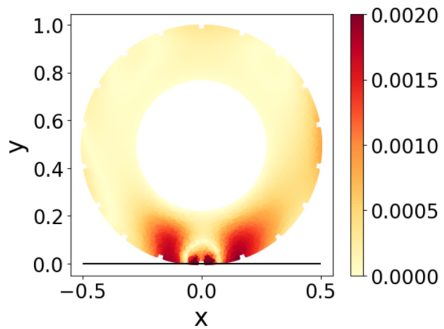
Improvement using adaptive sampling. Illustration



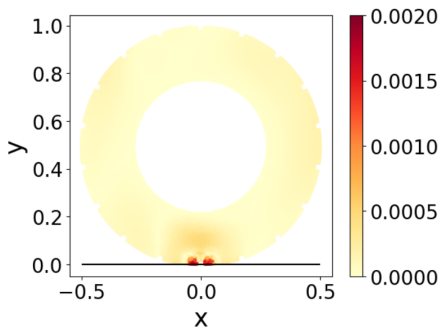
Truth



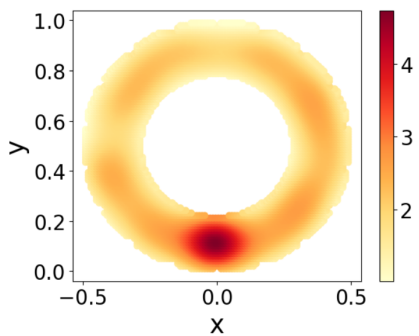
Pred Test



Err w/o FBOAL



Err w FBOAL



Density by FBOAL

Outline

A personal message to share...

- Collaborations
  between academics and “Z” partners often **raise original scientific questions** (Z = biologists, industrialists, socio-economists, etc.).
  But, Never forget we do not have the same jobs!

- Research time.
  Don’t try to solve first the "Z issue", but try to rephrase first a scientific question, and then bring an answer to this solution in you own field.
  Don’t leave your "scientific Island unless you know to find your way back! ... unless you want to change your work.

- Valorisation time. Your "field scientific answer" will bring an answer to the Z issue...(maybe with some adaptation)

- Innovation time.
  Your " scientific answer" applied to the Z needs will create a certain amount of added value (economical value).

Thank you!

A joint work

thanks to the Industrial Data Analytics and Machine Learning chair

Industrial Data Analytics & Machine Learning

with

- Antoine de Mathelin, Towards reliable machine learning under domain shift and costly labeling, with applications to engineering design Michelin & IDAML, Centre Borelli
- Khoa Nguyen, Development and assessment of physically informed learning methods : enhancement of multi-physical simulation in industrial contexts, CEA, Michelin & IDAML, Centre Borelli
- Fouad Oubari, Deep Generative design for Industrial Products Michelin & IDAML, Centre Borelli
- Rémy Vallot, Convergence acceleration of a nonlinear solver by statisticalphysically informed learning Michelin & IDAML, Centre Borelli, UTC
- CEA Team since 2018. C. Millet, G. Kluth, S. Oger, JC. Weill,... .
- Michelin Team since 2018. R. Décatoire, F. Deheeger, T. Dairay, R. Meunier,... .
- Borelli, ENS-Paris Saclay : Mounir Atiq, Ludovic Minvielle, Transfer learning for fall detection, Tarket, Sergio Peignier, Transfer learning, Guillaume Richard, Transfer learning for Temporal data, EDF , Nicolas Vayatis, Director Centre Borelli, ENS-Paris-Saclay,

Literature References

Cameron, S., Pretorius, A., and Roberts, S. (2024).

Nonparametric boundary geometry in physics informed deep learning.
Advances in Neural Information Processing Systems, 36.

Cuomo, S., Di Cola, V. S., Giampaolo, F., Rozza, G., Raissi, M., and Piccialli, F. (2022).

Scientific machine learning through physics–informed neural networks : Where we are and what's next.
Journal of Scientific Computing, 92(3) :88.

de Mathelin, A. (2024).

Towards reliable machine learning under domain shift and costly labeling with applications to engineering design.
PhD thesis, Université Paris-Saclay.

Gao, H., Sun, L., and Wang, J.-X. (2021).

Phygeonet : Physics-informed geometry-adaptive convolutional neural networks for solving parameterized steady-state pdes on irregular domain.
Journal of Computational Physics, 428 :110079.

Huang, G., Liu, Z., Van Der Maaten, L., and Weinberger, K. Q. (2017).

Densely connected convolutional networks.
In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4700–4708.

Krizhevsky, A., Sutskever, I., and Hinton, G. E. (2012).

Imagenet classification with deep convolutional neural networks.
Advances in neural information processing systems, 25.

Lu, L., Meng, X., Mao, Z., and Karniadakis, G. E. (2021).

Deepxde : A deep learning library for solving differential equations.
SIAM review, 63(1) :208–228.

Mao, Z., Jagtap, A. D., and Karniadakis, G. E. (2020).

Physics-informed neural networks for high-speed flows.
Computer Methods in Applied Mechanics and Engineering, 360 :112789.

Literature References

Nguyen, K. (2024).

Development and assessment of physically informed learning methods : towards multi-physical simulation in industrial contexts,.
PhD thesis, Université Paris-Saclay.

Nguyen, T. N. K., Dairay, T., Meunier, R., Millet, C., and Mougeot, M. (2024).

Geometry-aware framework for deep energy method : an application to structural mechanics with hyperelastic materials.
arXiv preprint arXiv :2405.03427.

Nguyen, T. N. K., Dairay, T., Meunier, R., and Mougeot, M. (2022).

Physics-informed neural networks for non-newtonian fluid thermo-mechanical problems : An application to rubber calendering process.
Engineering Applications of Artificial Intelligence, 114 :105176.

Oldenburg, J., Borowski, F., Öner, A., Schmitz, K.-P., and Stiehm, M. (2022).

Geometry aware physics informed neural network surrogate for solving navier–stokes equation (gapinn).
Advanced Modeling and Simulation in Engineering Sciences, 9(1) :8.

Peng, W., Zhou, W., Zhang, X., Yao, W., and Liu, Z. (2022).

Rang : A residual-based adaptive node generation method for physics-informed neural networks.
arXiv preprint arXiv :2205.01051.

Raissi, M., Perdikaris, P., and Karniadakis, G. E. (2019).

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.

Wu, W., Daneker, M., Jolley, M. A., Turner, K. T., and Lu, L. (2023).

Effective data sampling strategies and boundary condition constraints of physics-informed neural networks for identifying material properties in solid mechanics.
Applied mathematics and mechanics, 44(7) :1039–1068.