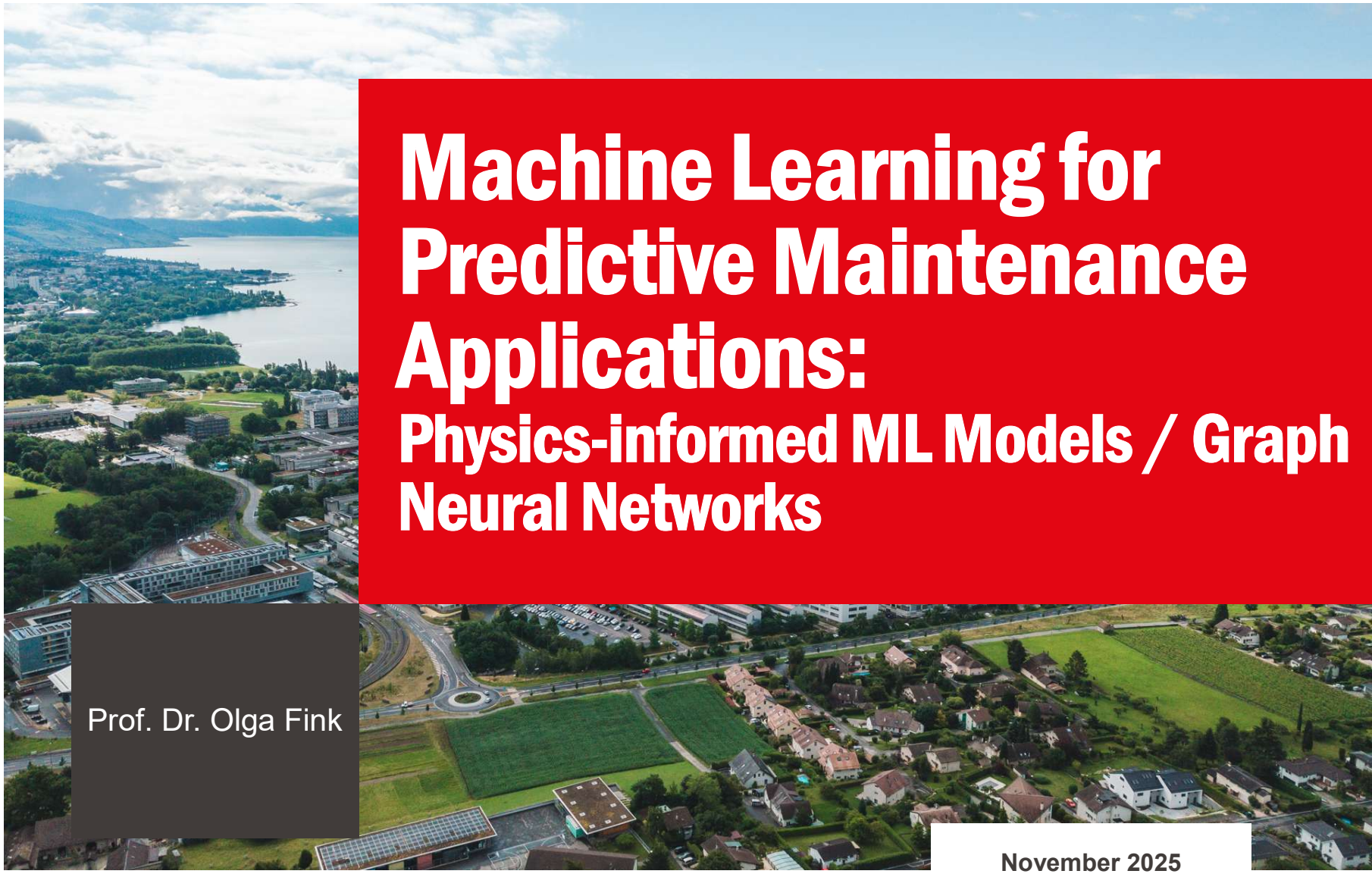


# Machine Learning for Predictive Maintenance Applications:

## Physics-informed ML Models / Graph Neural Networks

Prof. Dr. Olga Fink

École polytechnique fédérale de Lausanne

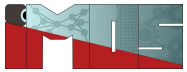
November 2025

model was developed at Factory A u
high-resolution data from newly instal
equipment. The model reliably detects
early bearing wear and motor imbalance.



When the same PHM model is deployed at Factory B and Factory C, its performance noticeably degrades:

At Factory B, the model raises many false alarms during normal production peaks.



# Digital Twins

# 1. Definition: Digital twins

- A digital twin is defined as “a living model of the physical asset or system, which continually adapts to operational changes based on the collected online data and information, and can forecast the future of the corresponding physical counterpart”.

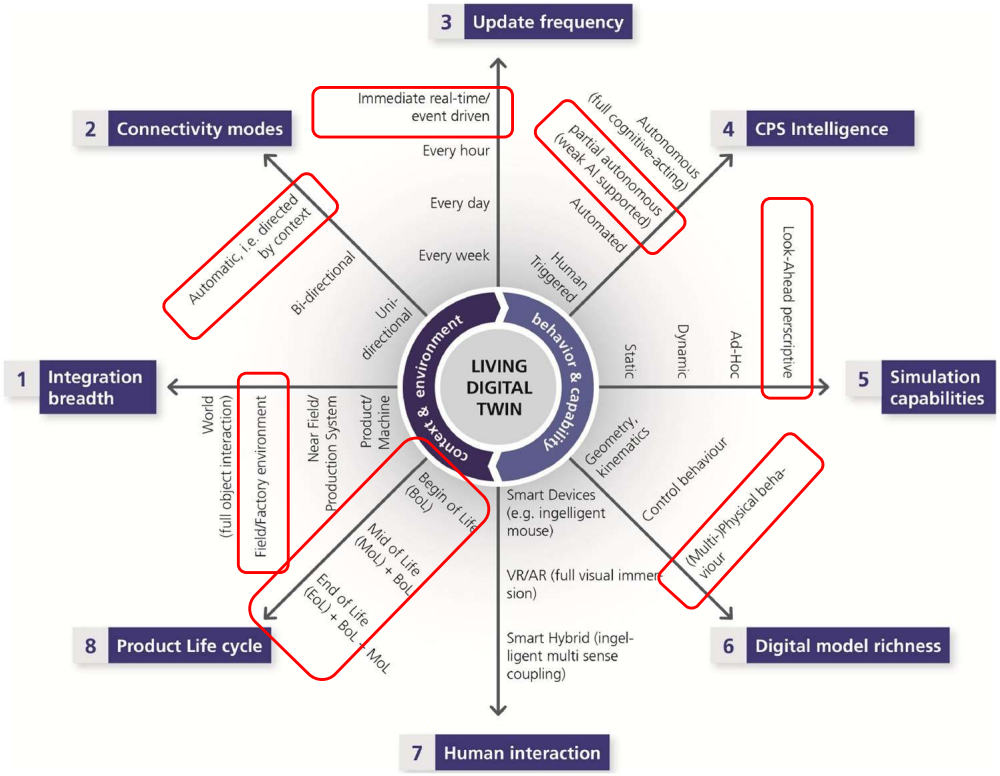
M. Grieves and J. Vickers, “Digital twin: Mitigating unpredictable, undesirable emergent behavior in complex systems,” in *Transdisciplinary Perspectives on Complex Systems: New Findings and Approaches*, Springer International Publishing, 2016, pp. 85–113.

# 2. Definition: Digital twins

- *A digital twin is a set of virtual information constructs that mimics the structure, context, and behavior of a natural, engineered, or social system (or system-of-systems), is dynamically updated with data from its physical twin, has a predictive capability, and informs decisions that realize value. The bidirectional interaction between the virtual and the physical is central to the digital twin.*

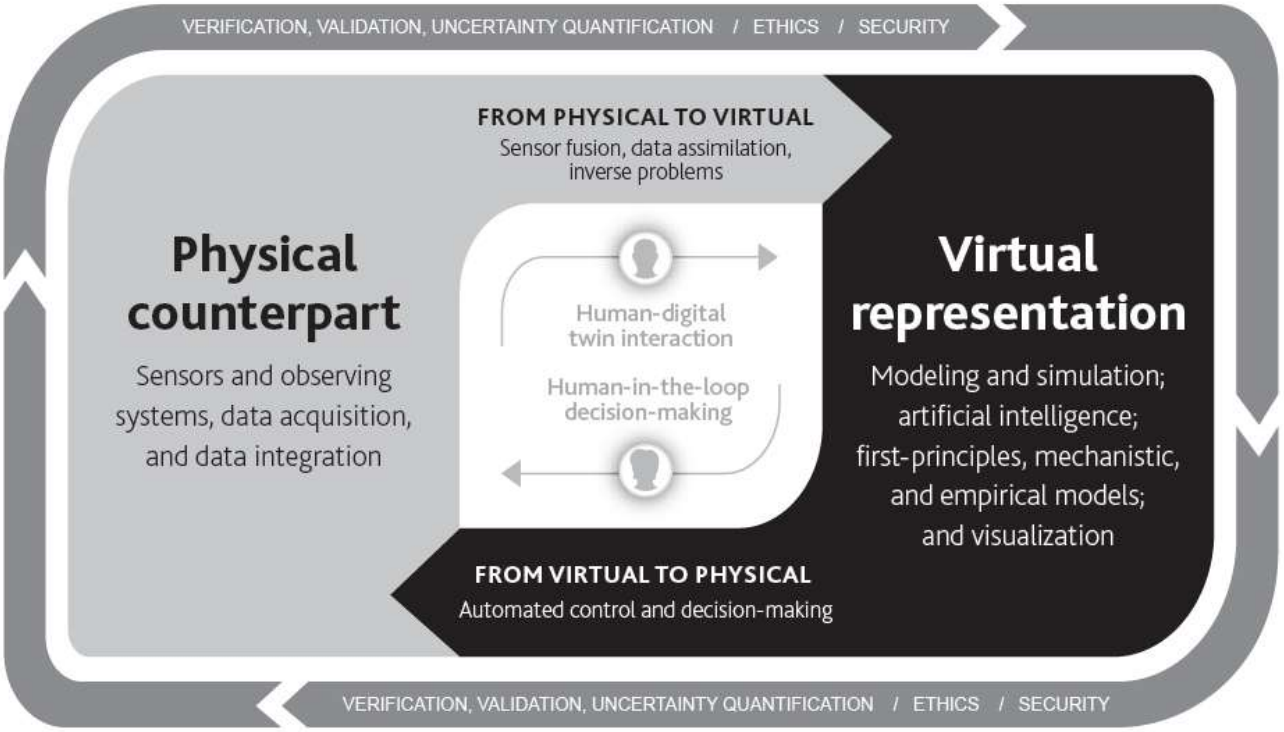
National Academies 2023: Foundational Research Gaps and Future Directions for Digital Twins

# Digital Twin integration



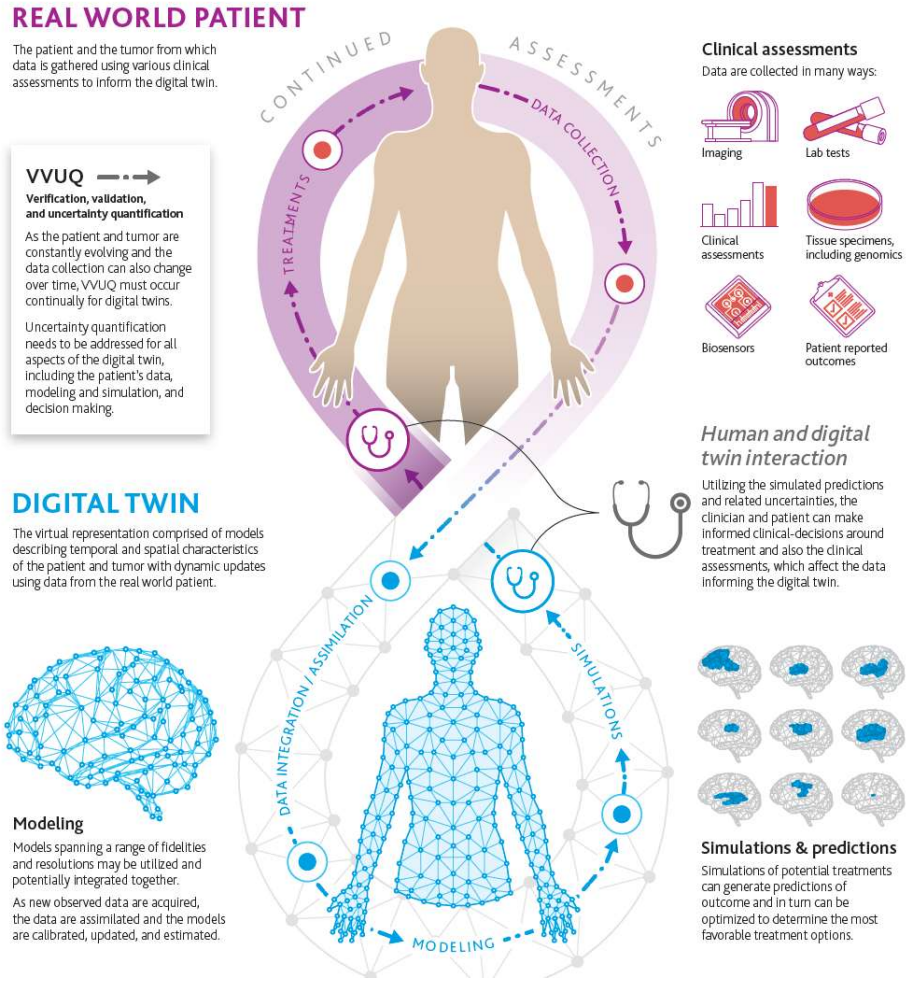

R. Stark, C. Fresemann, and K. Lindow, "Development and operation of Digital Twins for technical systems and services," *CIRP Ann.*, vol. 68, no. 1, pp. 129–132, Jan. 2019.

# Elements of a digital twin




National Academies 2023: Foundational Research Gaps and Future Directions for Digital Twins

# Digital twin of a patient

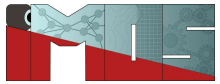



National Academies 2023: Foundational Research Gaps and Future Directions for Digital Twins

# Verification, Validation, Uncertainty Quantification

- **Verification:** Does a computer program correctly solve the equations of the mathematical model?
- **Validation:** To what degree is a model an accurate representation of the real world, from the perspective of the intended model uses?
- **Uncertainty Quantification:** What are uncertainties in model calculations of quantities of interest?

# Application fields



- **Manufacturing**
- Smart Cities
- Health care
- Energy Sector
- Aerospace and Defense
- Automotive Industry
- Building Energy Systems
- Agriculture
- Climate
- …

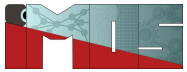
# Application cases of DT

- Product development
- Design + Testing
- PHM / Maintenance
- Optimizing product life-cycle management
- Process Optimization
- Prescriptive Operation

# A Digital Twin should be Fit for Purpose

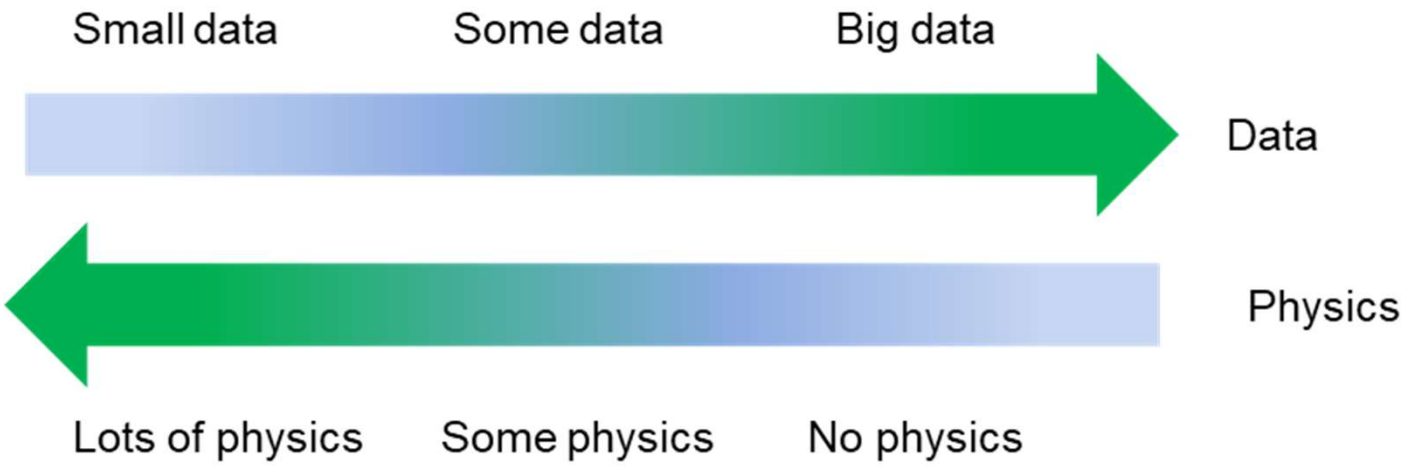
- **Balancing required fidelity for prediction, available resources, and acceptable costs**
- Different digital twin purposes drive different fitness requirements related to modeling fidelity, data availability, visualization, time-to-solution, etc.
- For many potential use cases, achieving fitness-for-purpose is currently intractable

National Academies 2023: Foundational Research Gaps and Future Directions for Digital Twins



# Hybrid Models

# Space of potential solutions

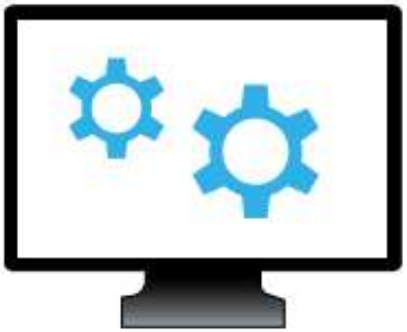


# Different ways to integrate physics and prior knowledge

Observational bias

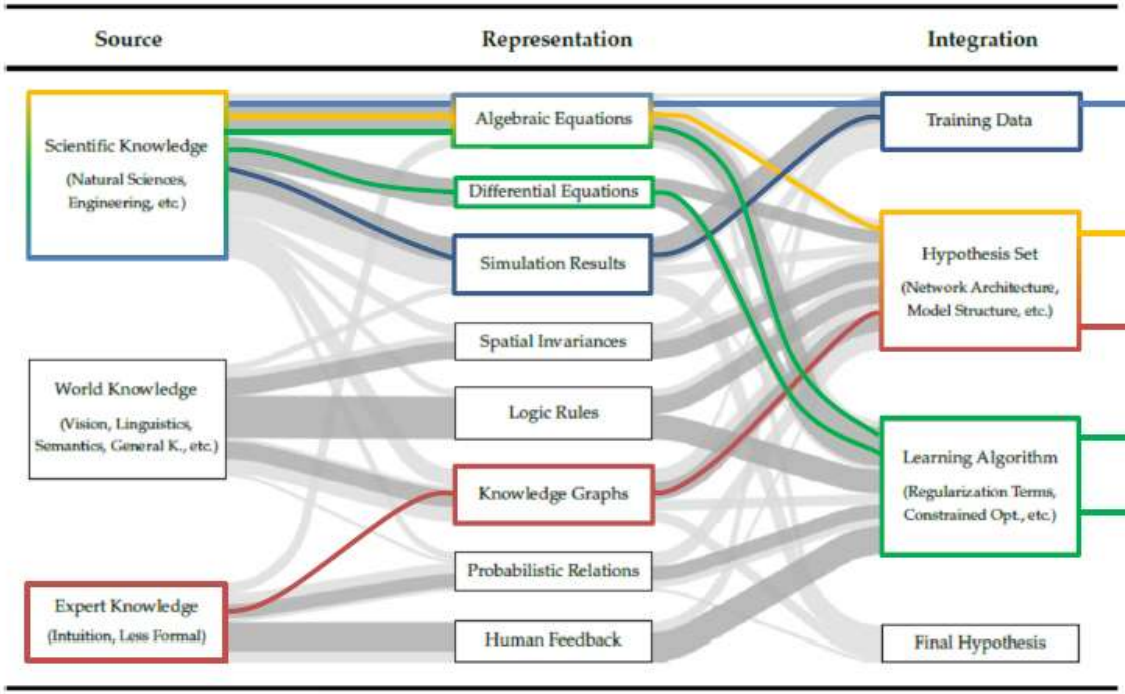
Inductive bias

Learning bias



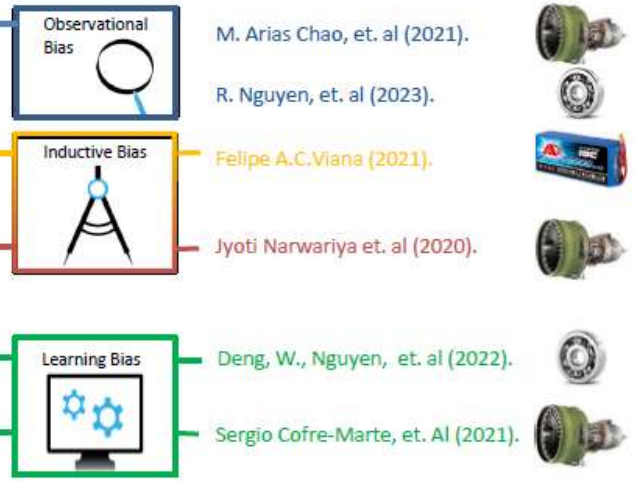
Karniadakis, G.E., Kevrekidis, I.G., Lu, L., Perdikaris, P., Wang, S. and Yang, L., 2021. Physics-informed machine learning. *Nature Reviews Physics*, *3*(6), pp.422-440.

# Physics-informed ML for prognostics and health management (PHM)




L. von Rueden *et al.*, (2021)

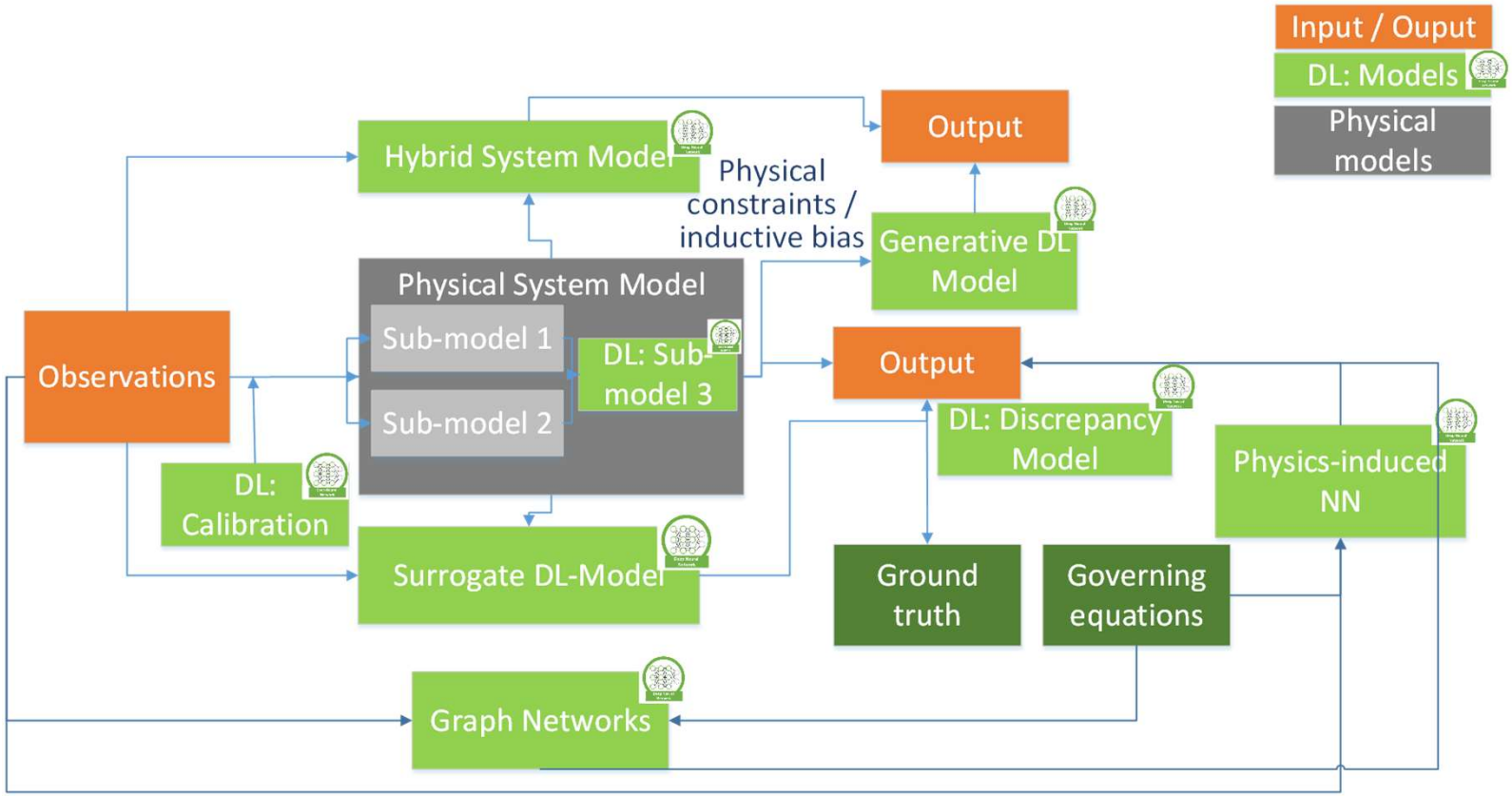
PIML methods in PHM:




George Em Karniadakis, Ioannis G. Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. Nature Reviews Physics 2021

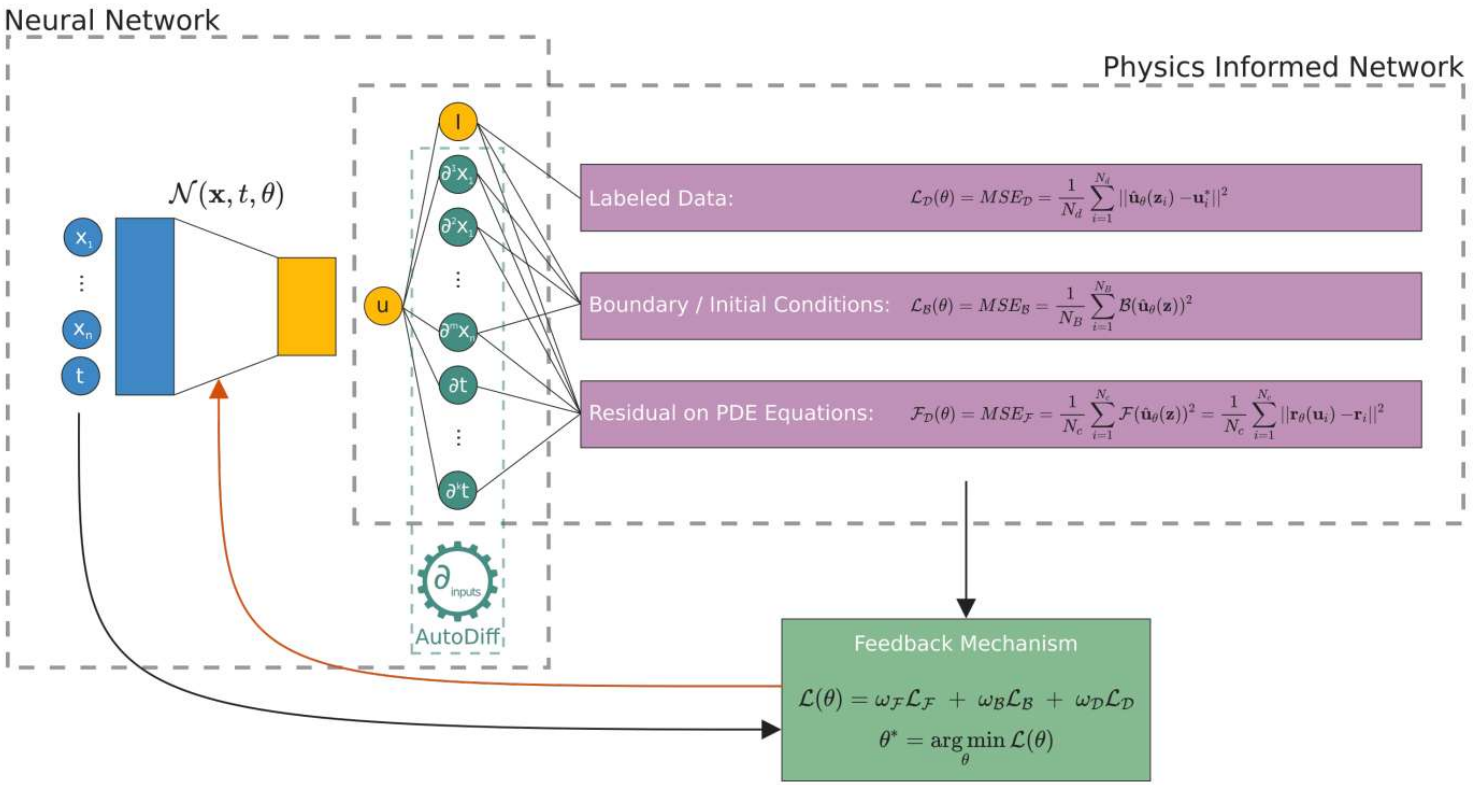
Source: M. Arias

# Different ways of fusing DL and physics



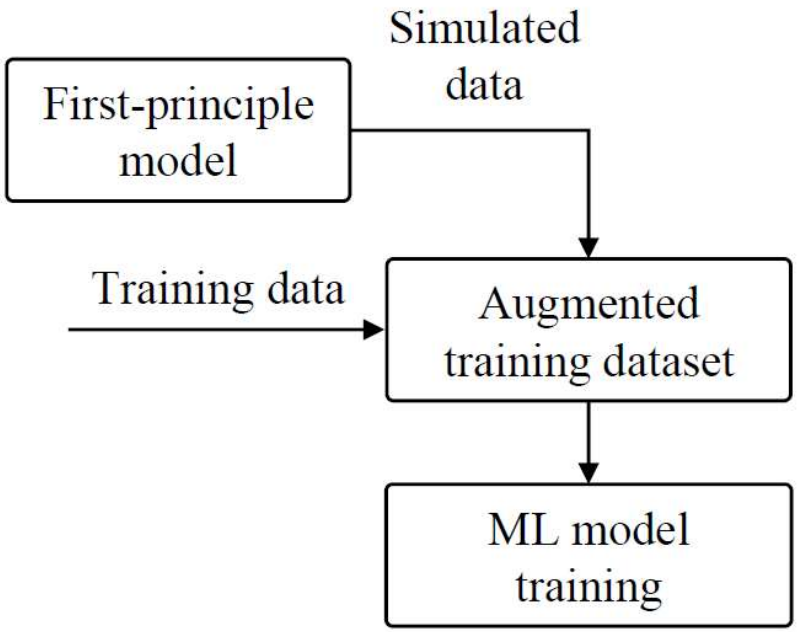
# Combining Physics-Based and Deep Learning Models

# Physics-informed ML



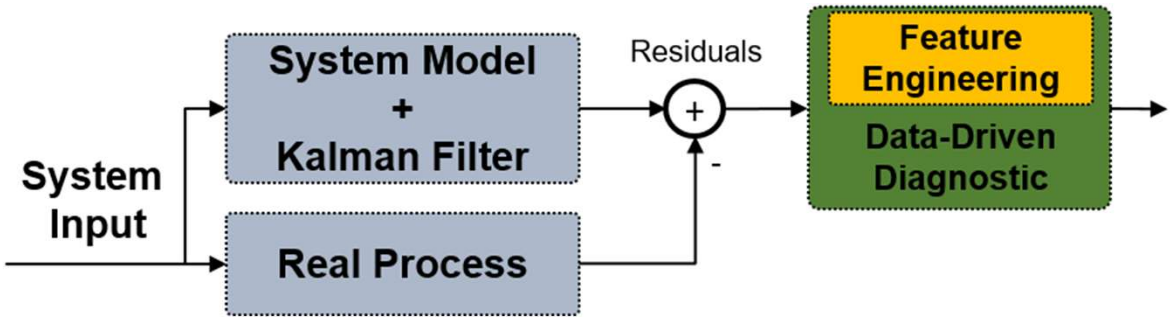

Thelen, A., X. Zhang, O. Fink, Y. Lu, S. Ghosh, B.D. Youn, M.D. Todd, S. Mahadevan, C. Hu, Z. Hu. "A Comprehensive Review of Digital Twin - Part 1: Modeling and Twinning Enabling Technologies]{A Comprehensive Review of Digital Twin - Part 1: Modeling and Twinning Enabling Technologies.“¨, Structural and Multidisciplinary Optimization journal

# Data Augmentation



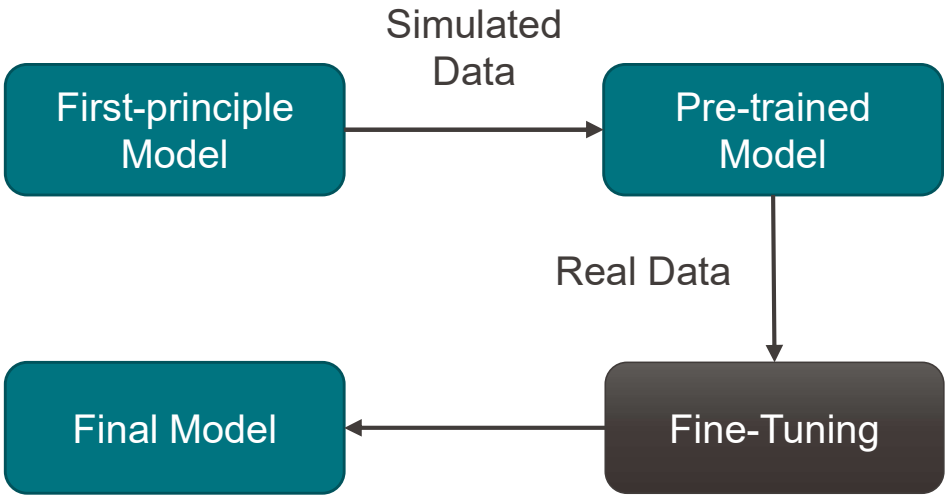

Thelen, A., X. Zhang, O. Fink, Y. Lu, S. Ghosh, B.D. Youn, M.D. Todd, S. Mahadevan, C. Hu, Z. Hu. "A Comprehensive Review of Digital Twin - Part 1: Modeling and Twinning Enabling Technologies]{A Comprehensive Review of Digital Twin - Part 1: Modeling and Twinning Enabling Technologies.“¨, accepted for publication in Structural and Multidisciplinary Optimization journal

# Residual-model




Rausch, Randal T., et al. "Integrated in-flight fault detection and accommodation: A model-based study." (2007): 962-969.

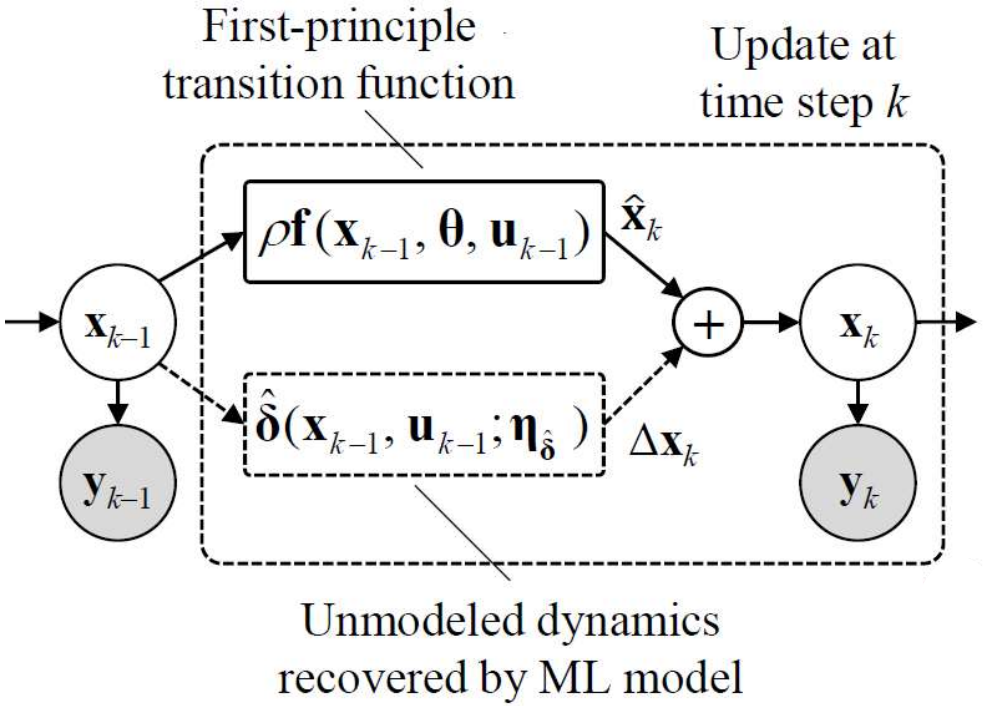
# Transfer learning






Thelen, A., X. Zhang, O. Fink, Y. Lu, S. Ghosh, B.D. Youn, M.D. Todd, S. Mahadevan, C. Hu, Z. Hu. "A Comprehensive Review of Digital Twin - Part 1: Modeling and Twinning Enabling Technologies]{A Comprehensive Review of Digital Twin - Part 1: Modeling and Twinning Enabling Technologies.“¨, accepted for publication in Structural and Multidisciplinary Optimization journal

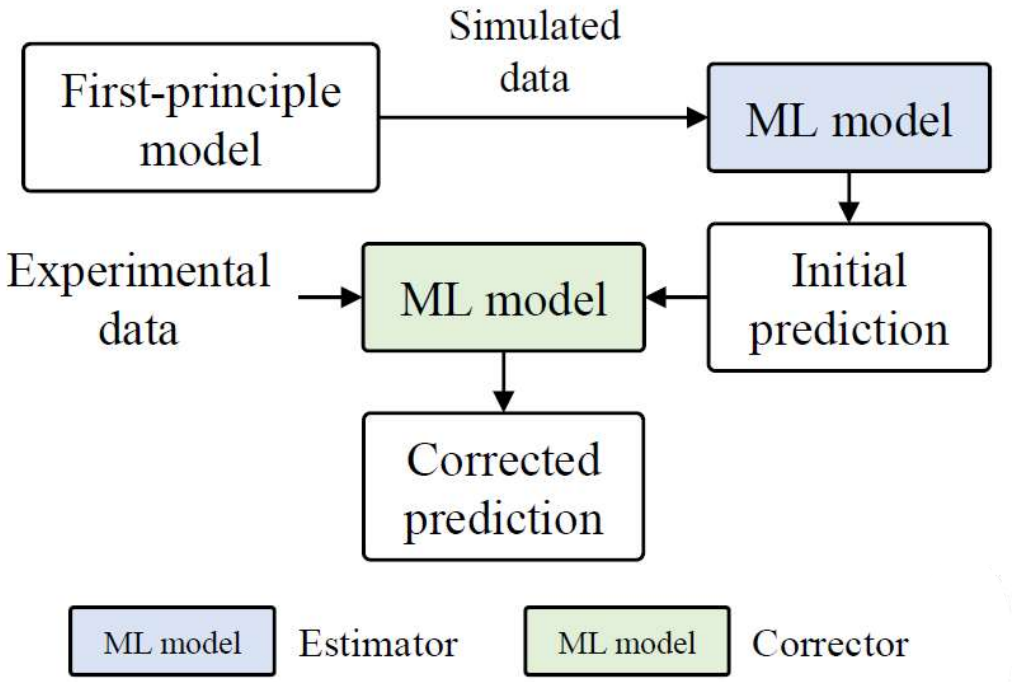
# Delta Learning (Missing Physics)





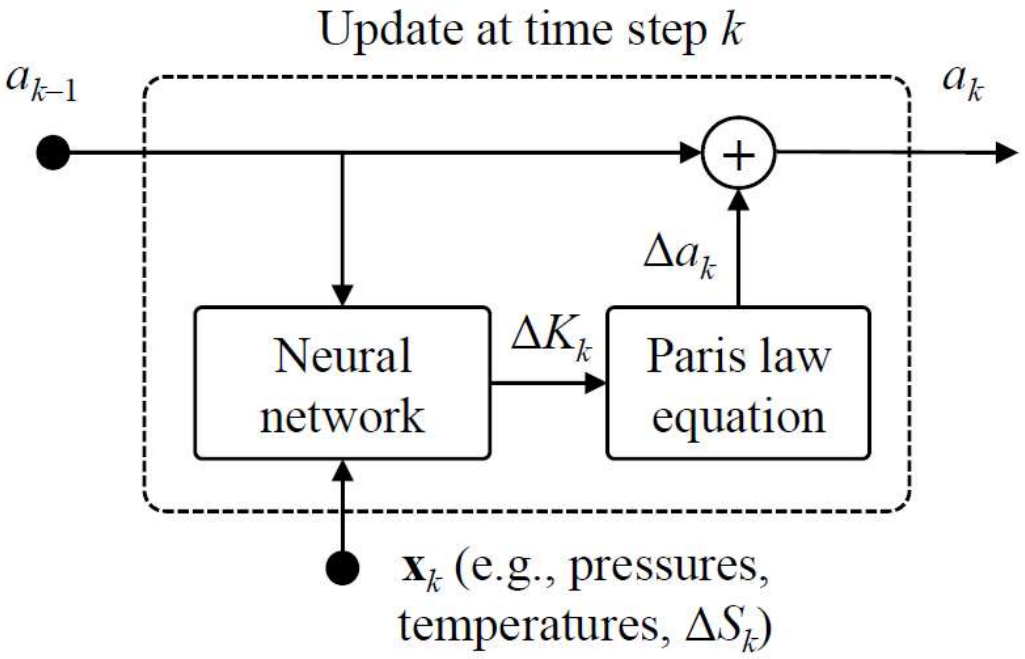

Thelen, A., X. Zhang, O. Fink, Y. Lu, S. Ghosh, B.D. Youn, M.D. Todd, S. Mahadevan, C. Hu, Z. Hu. "A Comprehensive Review of Digital Twin - Part 1: Modeling and Twinning Enabling Technologies]{A Comprehensive Review of Digital Twin - Part 1: Modeling and Twinning Enabling Technologies.“¨, accepted for publication in Structural and Multidisciplinary Optimization journal

# Delta Learning (ML Prediction)



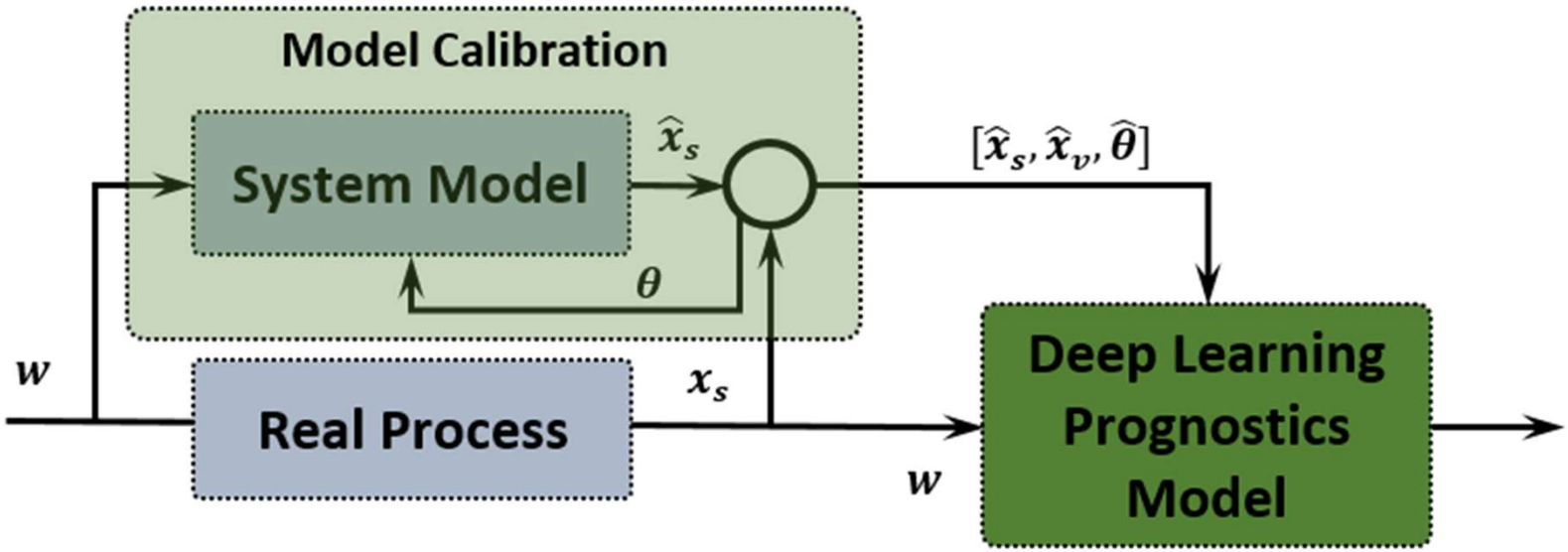

Thelen, A., X. Zhang, O. Fink, Y. Lu, S. Ghosh, B.D. Youn, M.D. Todd, S. Mahadevan, C. Hu, Z. Hu. "A Comprehensive Review of Digital Twin - Part 1: Modeling and Twinning Enabling Technologies]{A Comprehensive Review of Digital Twin - Part 1: Modeling and Twinning Enabling Technologies.“¨, accepted for publication in Structural and Multidisciplinary Optimization journal

# ML-Assisted Prediction

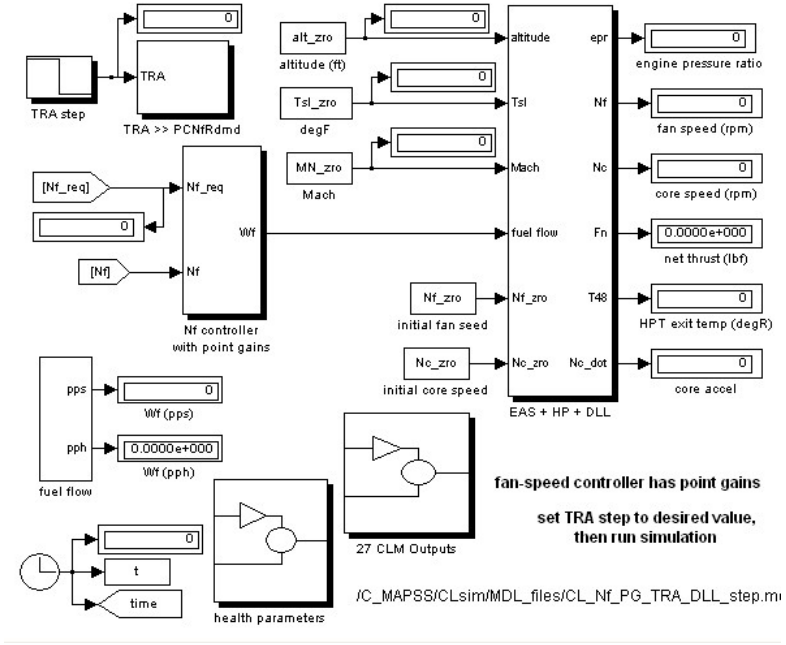



Thelen, A., X. Zhang, O. Fink, Y. Lu, S. Ghosh, B.D. Youn, M.D. Todd, S. Mahadevan, C. Hu, Z. Hu. "A Comprehensive Review of Digital Twin - Part 1: Modeling and Twinning Enabling Technologies]{A Comprehensive Review of Digital Twin - Part 1: Modeling and Twinning Enabling Technologies.“¨, accepted for publication in Structural and Multidisciplinary Optimization journal

# Fusing physical performance models and deep learning

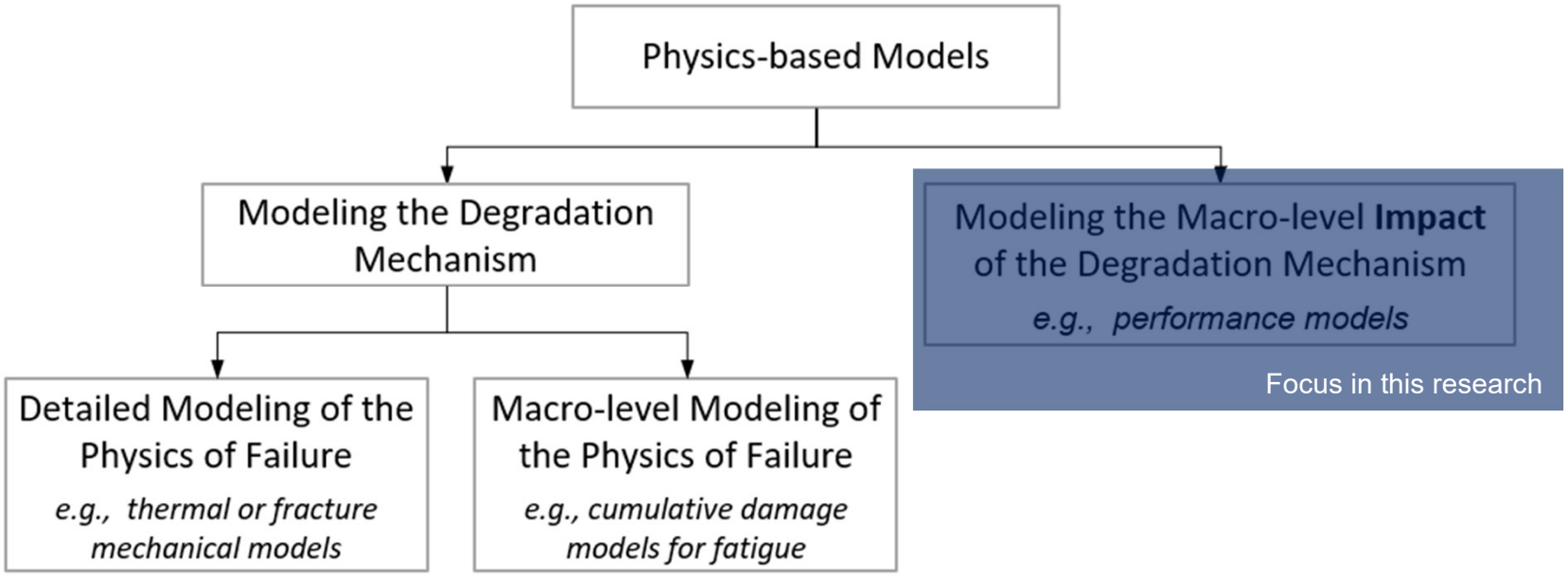



M. Arias Chao, C. S. Kulkarni, K. Goebel, and O. Fink, "Fusing Physics-based and Deep Learning Models for Prognostics," Reliability Engineering & System Safety.



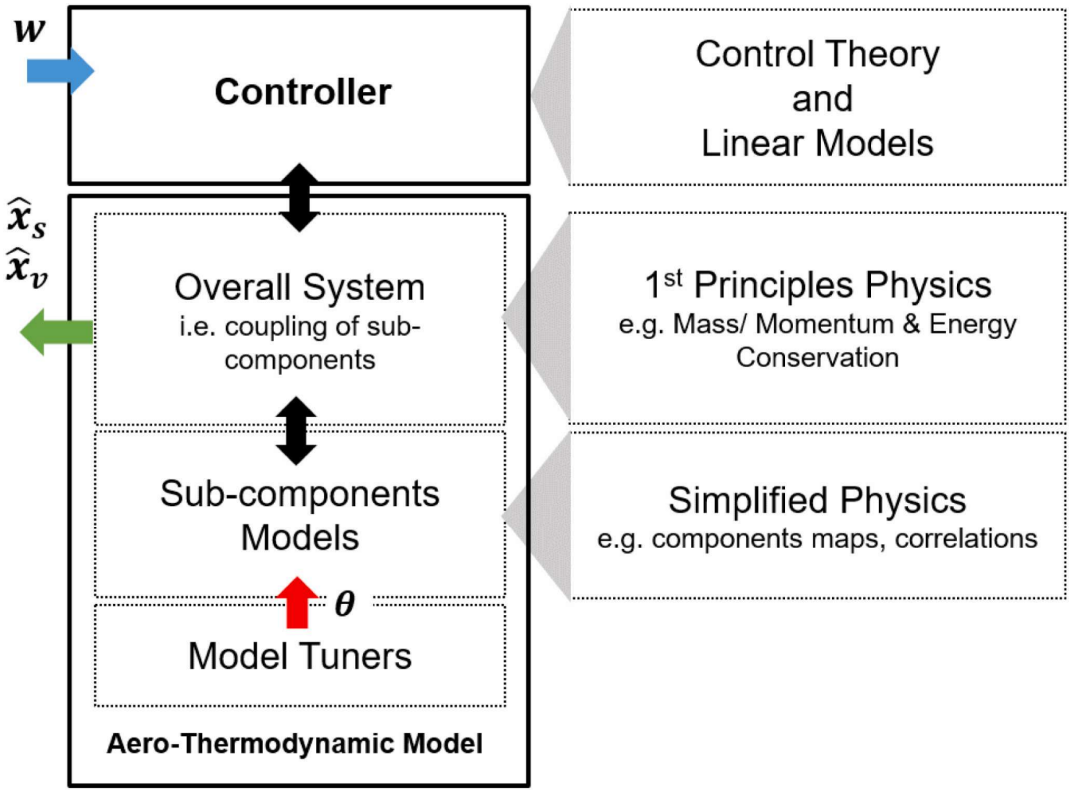

# Calibration-based hybrid framework for prognostics

# Physics-based models




M. Arias, Algorithm for Fleet Diagnostics and Prognostics Combining Deep Learning and Physics-based Performance Models (2021)

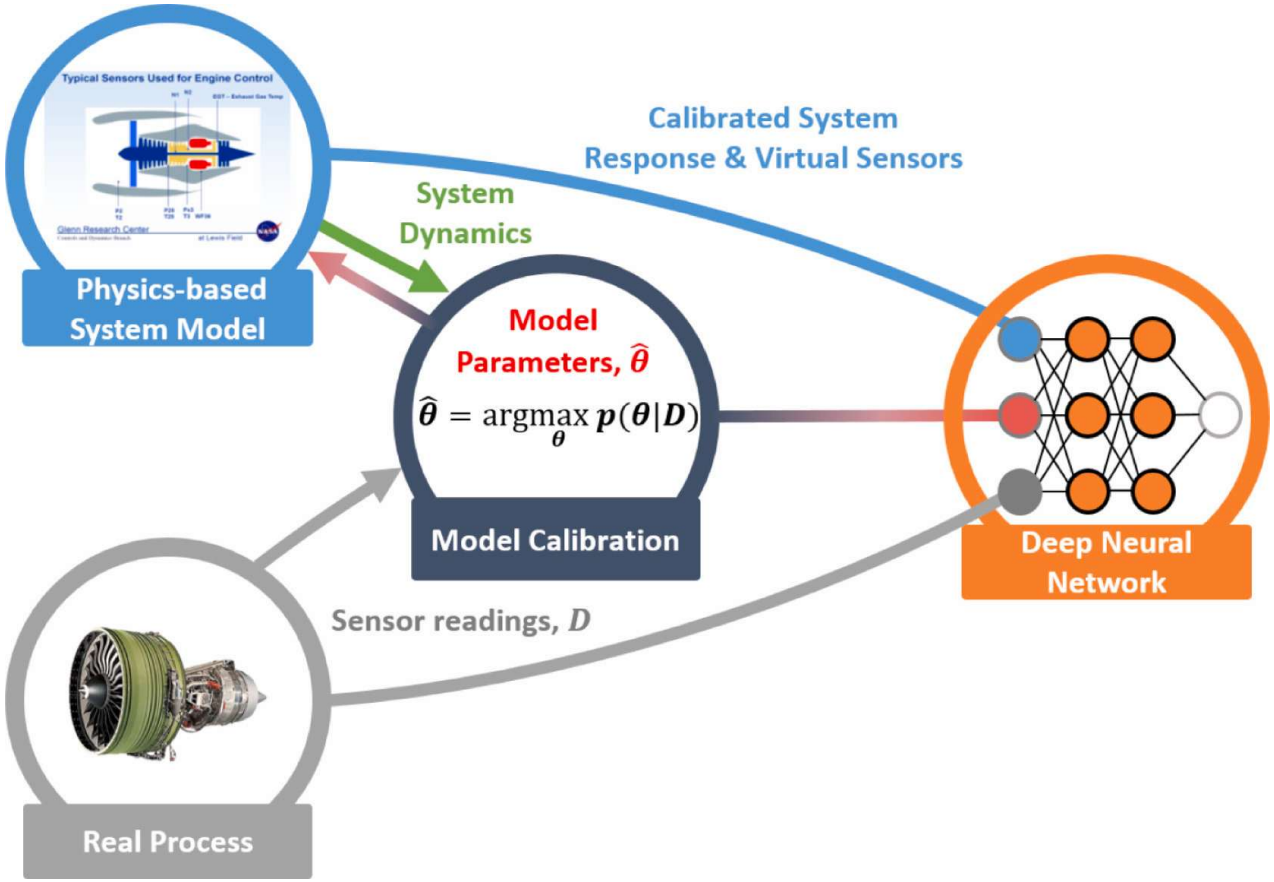
# General topology of an aero-thermodynamic performance model





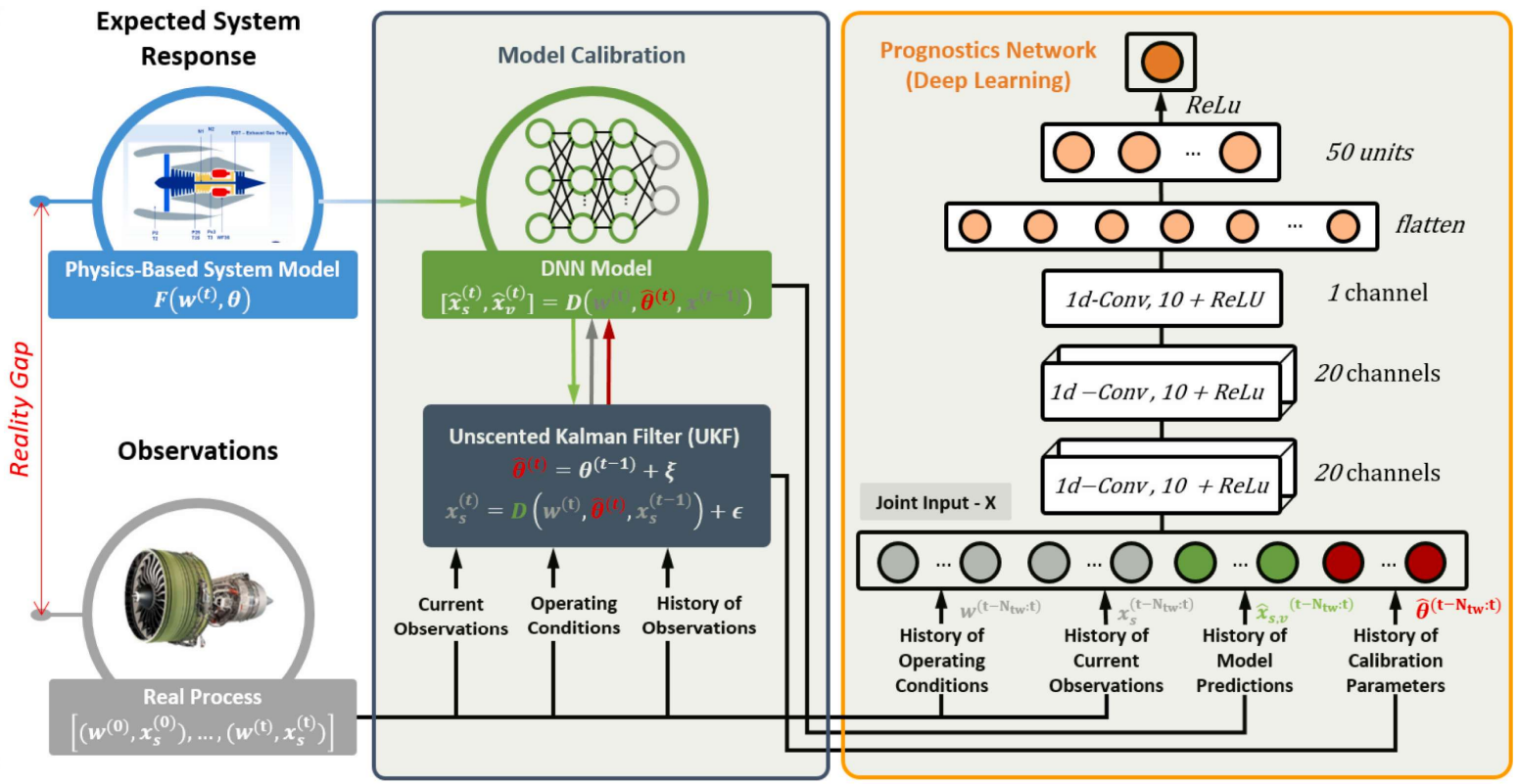

M. Arias Chao, C. S. Kulkarni, K. Goebel, and O. Fink, "Fusing Physics-based and Deep Learning Models for Prognostics," Reliability Engineering & System Safety.

# Basic idea of the hybrid approach



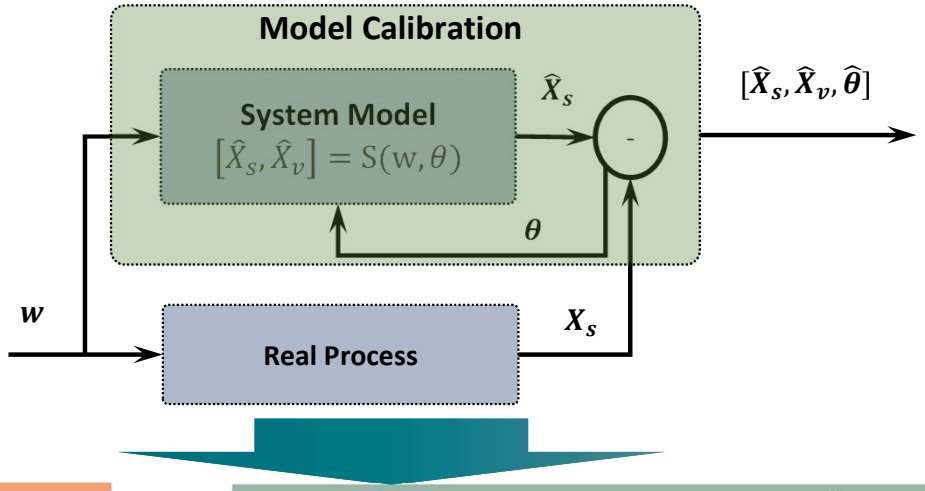

M. Arias Chao, C. S. Kulkarni, K. Goebel, and O. Fink, "Fusing Physics-based and Deep Learning Models for Prognostics," Reliability Engineering & System Safety.

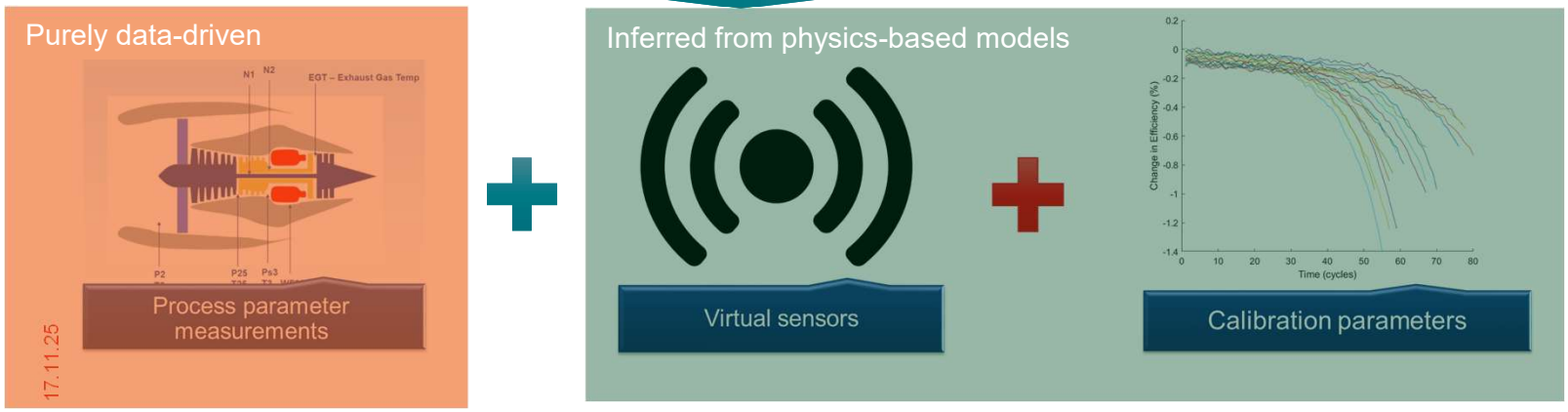
# Proposed framework




M. Arias Chao, C. S. Kulkarni, K. Goebel, and O. Fink, "Fusing Physics-based and Deep Learning Models for Prognostics," Reliability Engineering & System Safety.

# Enhancing the input space with inputs from physics-based models



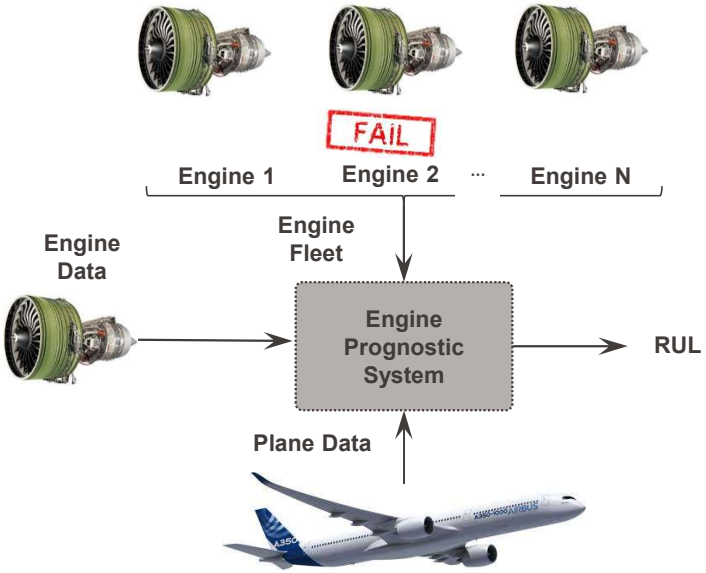


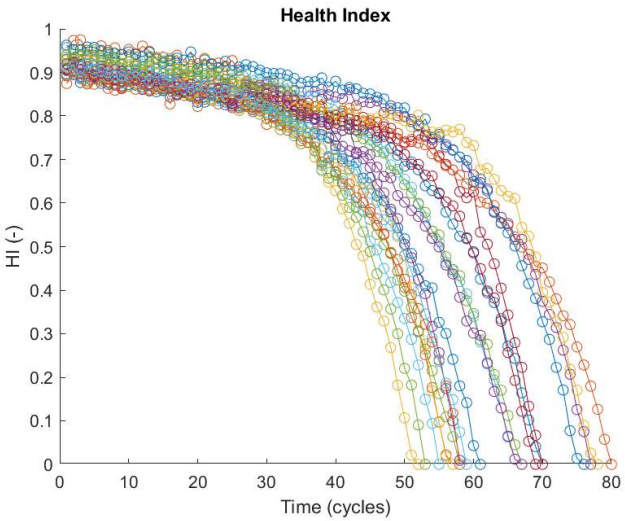

M. Arias Chao, C. S. Kulkarni, K. Goebel, and O. Fink, "Fusing Physics-based and Deep Learning Models for Prognostics," Reliability Engineering & System Safety.

# Estimation of the Remaining Useful Life

Remaining Useful Life (RUL) of a component

– the amount of time a component can be expected to continue operating within its stated specifications

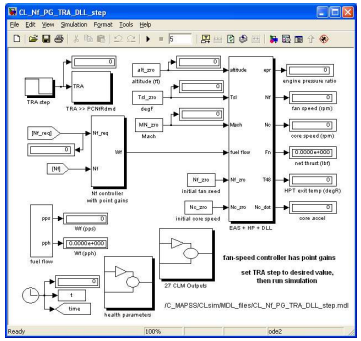


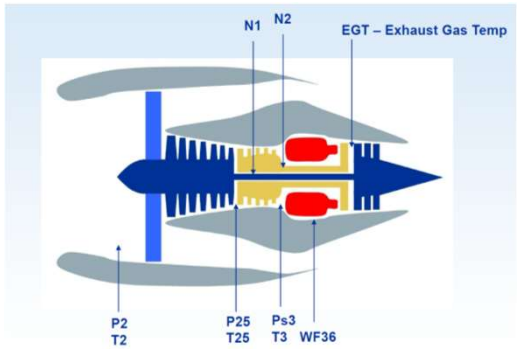




M. Arias Chao, C. S. Kulkarni, K. Goebel, and O. Fink, "Fusing Physics-based and Deep Learning Models for Prognostics," Reliability Engineering & System Safety.

# Test Case: Turbofan Engine

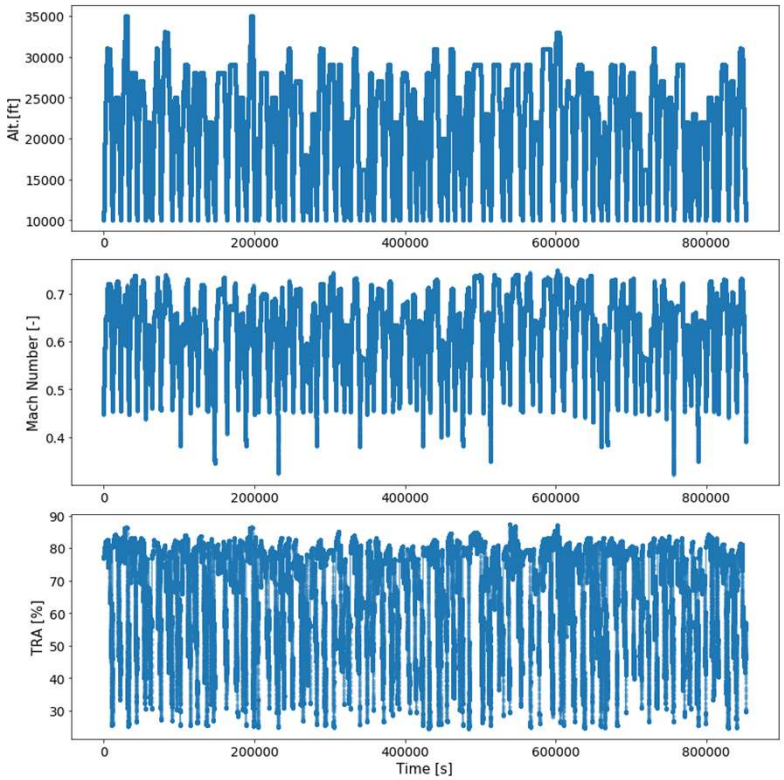
- **Simulation of a realistic large commercial turbofan engine**
  - **Simulated run to failure trajectories with the Commercial Modular Aero-Propulsion System Simulation (C-MAPSS) dynamical model**
- **Real flight conditions from a commercial jet**
  - NASA **DASHlink**
  - **~500 different (1-12h) flights**
  - **Recordings covering climb, cruise and descend**
  - **Operative conditions -** $w \in R^3$
- **Data from a fleet of 9 turbofan engines**
  - **Internal sensors -** $x_s \in R^{10}$
  - **Virtual sensors -** $x_v \in R^{15}$
  - **Model parameters -** $\theta \in R^{10}$






M. Arias Chao, C. S. Kulkarni, K. Goebel, and O. Fink, "Fusing Physics-based and Deep Learning Models for Prognostics," Reliability Engineering & System Safety.

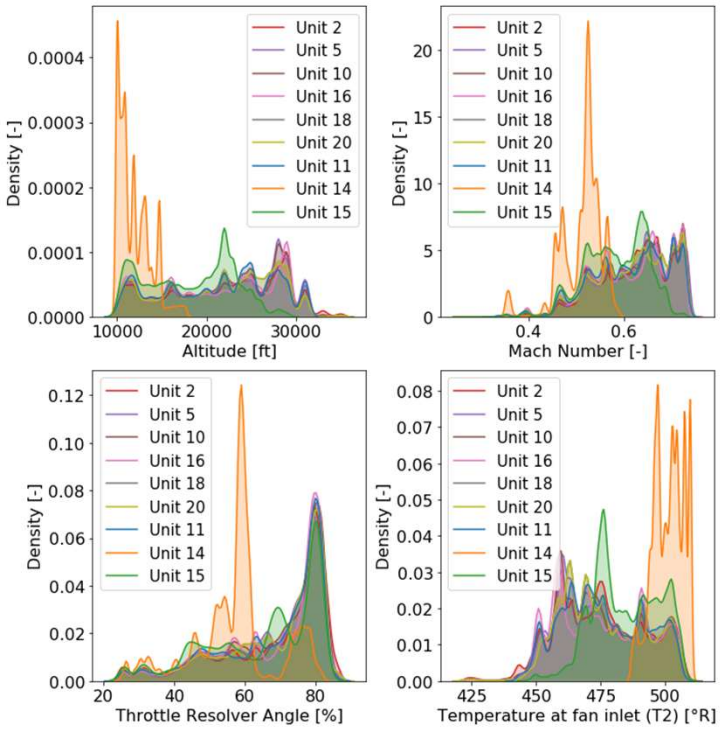
# Test Case: Turbofan Dataset




- **Failure modes:** 1) HPT degradation 2) HPT and LPT flow and efficiency degradation
- **Flight conditions:** Alt > 10000 [ft]
- **Training Data:**
  - **6 engines**
  - **50-90 cycles (flights)**
  - $\boldsymbol{w}, \boldsymbol{X_s}, \widehat{\boldsymbol{\theta}}$ **&** $RUL$
  - **5.5** x $10^6$ **samples**
- **Test Data:**
  - 3 engines
  - $\boldsymbol{w}, \boldsymbol{X_s}$ & $\widehat{\boldsymbol{\theta}}$
  - 1.2 x $10^6$ samples

M. Arias Chao, C. S. Kulkarni, K. Goebel, and O. Fink, "Fusing Physics-based and Deep Learning Models for Prognostics," Reliability Engineering & System Safety.

# Kernel density estimations of the simulated flight envelopes given by recordings of altitude



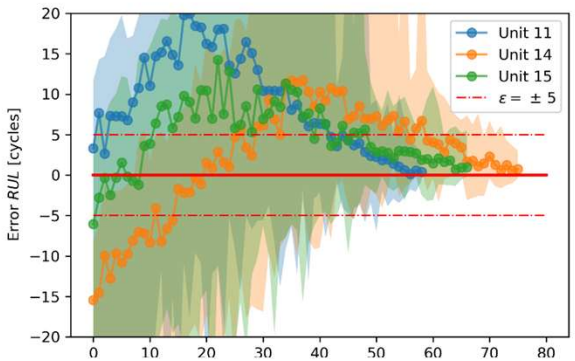

M. Arias Chao, C. S. Kulkarni, K. Goebel, and O. Fink, "Fusing Physics-based and Deep Learning Models for Prognostics," Reliability Engineering & System Safety.

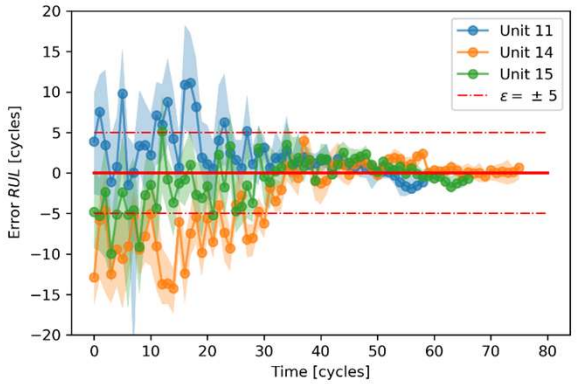
# RUL prediction results

Data-driven - $\boldsymbol{X} = [\boldsymbol{w}, \boldsymbol{X_s}]$

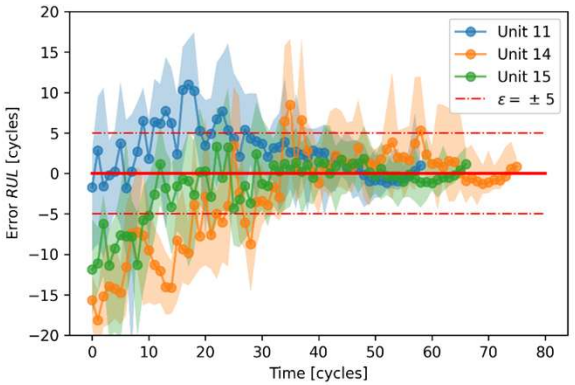
Hybrid - $\boldsymbol{X} = [\boldsymbol{w}, \hat{\boldsymbol{X}}_s, \hat{\boldsymbol{X}}_v, \hat{\boldsymbol{\theta}}]$

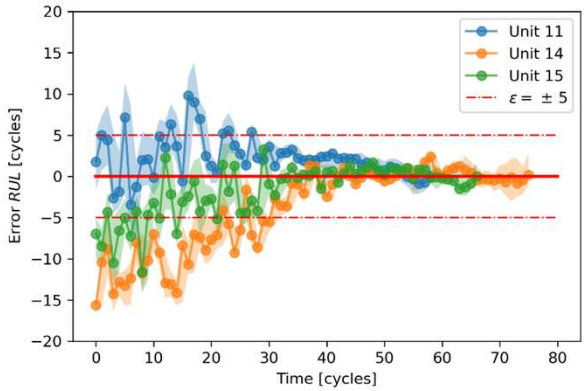
FFN




CNN




M. Arias Chao, C. S. Kulkarni, K. Goebel, and O. Fink, "Fusing Physics-based and Deep Learning Models for Prognostics," Reliability Engineering & System Safety.

# Detailed results RUL prediction

## FFN

| Metric | Data-Driven | Hybrid | Rel. Delta |
|---|---|---|---|
| RMSE [-] | $7.89 \pm 0.12$ | $4.22 \pm 0.10$ | -47% |
| $s \times 10^5$[-] | $1.39 \pm 0.04$ | $0.44 \pm 0.01$ | -68% |

## CNN

| Metric | Data-Driven | Hybrid | Rel. Delta |
|---|---|---|---|
| RMSE [-] | $4.95 \pm 0.15$ | $4.14 \pm 0.09$ | -16% |
| $s \times 10^5$[-] | $0.56 \pm 0.03$ | $0.44 \pm 0.02$ | -21% |

$$s = \sum_{j=1}^{m_*} exp(\alpha|\Delta^{(j)}|)$$

$$RMSE = \sqrt{\frac{1}{m_*}\sum_{j=1}^{m_*}(\Delta^{(j)})^2}$$



M. Arias Chao, C. S. Kulkarni, K. Goebel, and O. Fink, "Fusing Physics-based and Deep Learning Models for Prognostics," Reliability Engineering & System Safety.

# EOL prediction within the error bound of 5 cycles

## CNN

$t_{EOL} - t_{\epsilon_y<5}$ in [cycles]

| unit | Data-Driven | Hybrid | Rel. Delta |
|---|---|---|---|
| 11 | 11 | 31 | 195% |
| 14 | 15 | 43 | 197% |
| 15 | 24 | 37 | 54% |
| Fleet Avg. | 16 | 37 | 127% |

→ Prediction **horizon prolonged by 127%** on average (for some units ca. 200%)



M. Arias Chao, C. S. Kulkarni, K. Goebel, and O. Fink, "Fusing Physics-based and Deep Learning Models for Prognostics," Reliability Engineering & System Safety.

# Detailed results RUL prediction: smaller training dataset

## CNN: trained on units 2,5,10,16,18,20

| Metric | Data-Driven | Hybrid | Rel. Delta |
|---|---|---|---|
| RMSE [-] | $4.95 \pm 0.15$ | $4.14 \pm 0.09$ | -16% |
| $s \times 10^5$[-] | $0.56 \pm 0.03$ | $0.44 \pm 0.02$ | -21% |

$$s = \sum_{j=1}^{m_*} exp(\alpha|\Delta^{(j)}|)$$

$$RMSE = \sqrt{\frac{1}{m_*}\sum_{j=1}^{m_*}(\Delta^{(j)})^2}$$

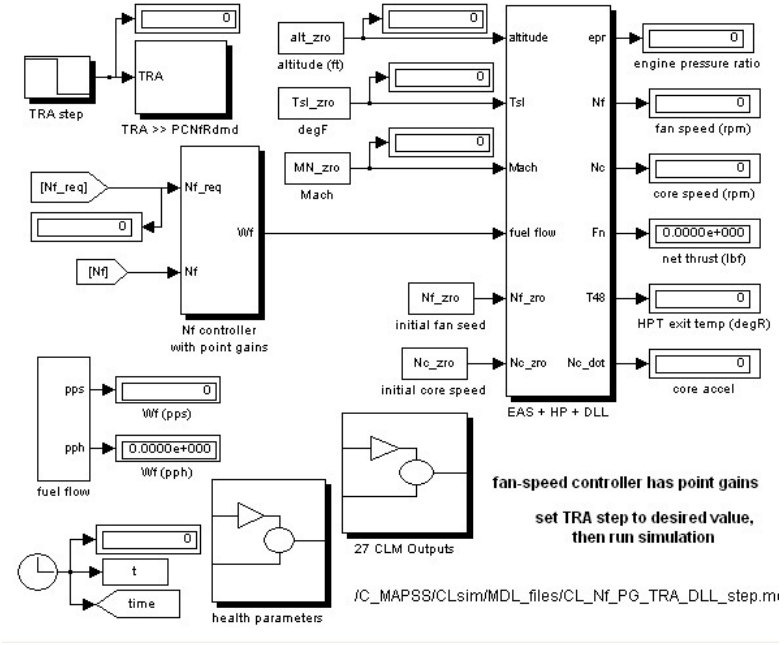
## CNN: trained on units 16,18,20

| Metric | Data-Driven | Hybrid | Rel. Delta |
|---|---|---|---|
| RMSE [-] | $5.97 \pm 0.37$ | $4.22 \pm 0.12$ | -29% |
| $s \times 10^5$[-] | $0.61 \pm 0.03$ | $0.43 \pm 0.02$ | -29% |
| rel. Delta RMSE [%] | 17% | 2% | |
| Rel. Delta s [%] | 8% | -2% | |

→ ca. 50% reduction of training data size

→ performance level maintained

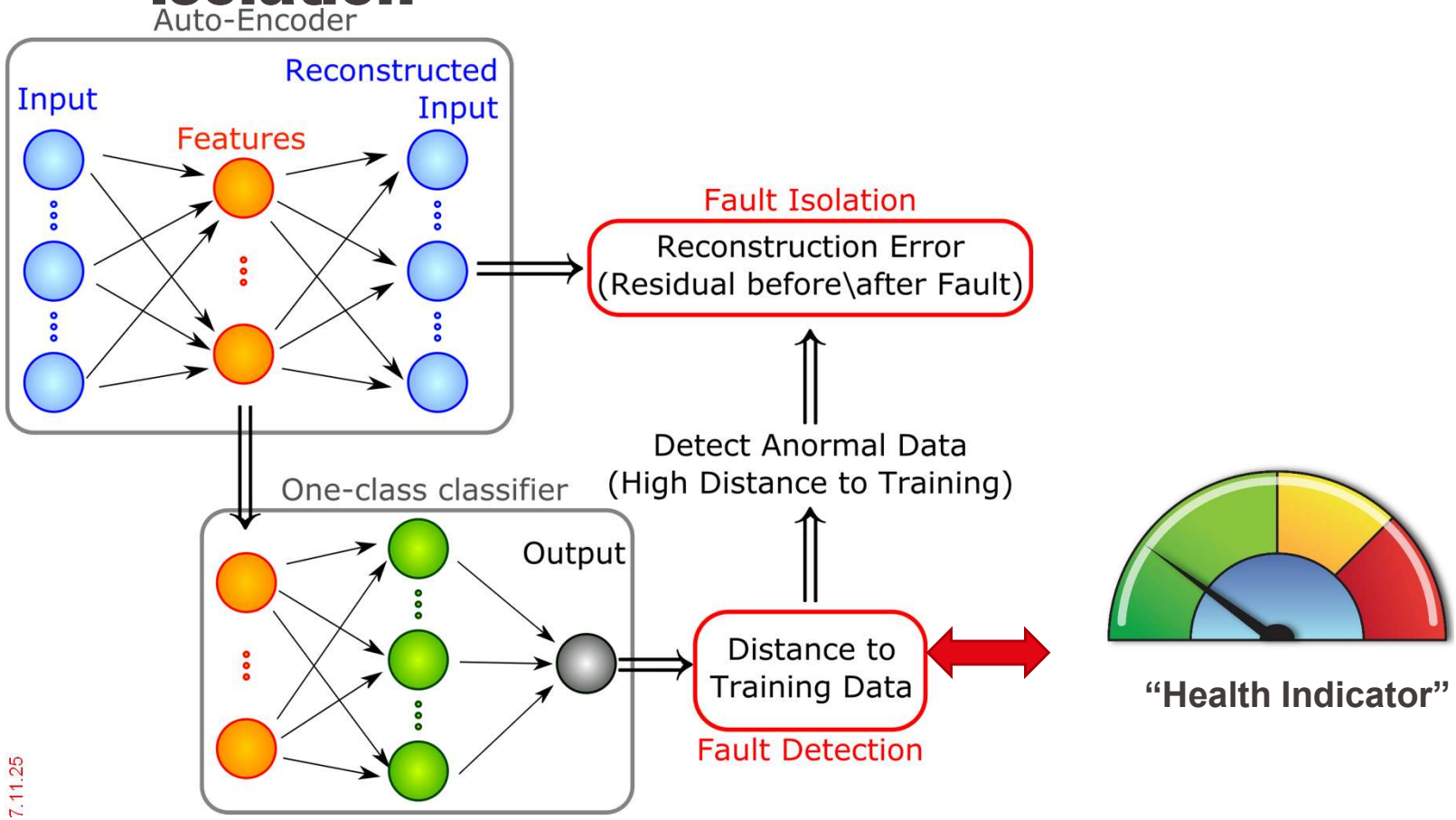
M. Arias Chao, C. S. Kulkarni, K. Goebel, and O. Fink, "Fusing Physics-based and Deep Learning Models for Prognostics," Reliability Engineering & System Safety.




# Hybrid Fault Detection and Diagnostics

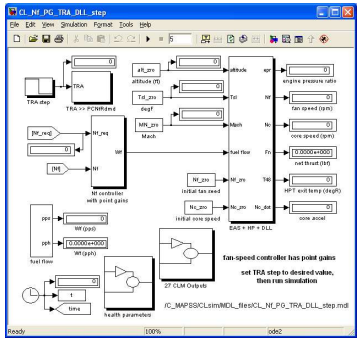
How to improve not only the performance but also the interpretability and generalizability?

# Analysing the reconstruction residuals for fault isolation



# Test Case: Turbofan Engine

- **Simulation of a realistic large commercial turbofan engine**
- **24 flight cycles**
- **Each flight contains ca. 175 snapshots of recordings covering climb, cruise and descend conditions**
- **Real flight conditions from a commercial jet**
- **Fault: HPC Efficiency with different magnitudes**

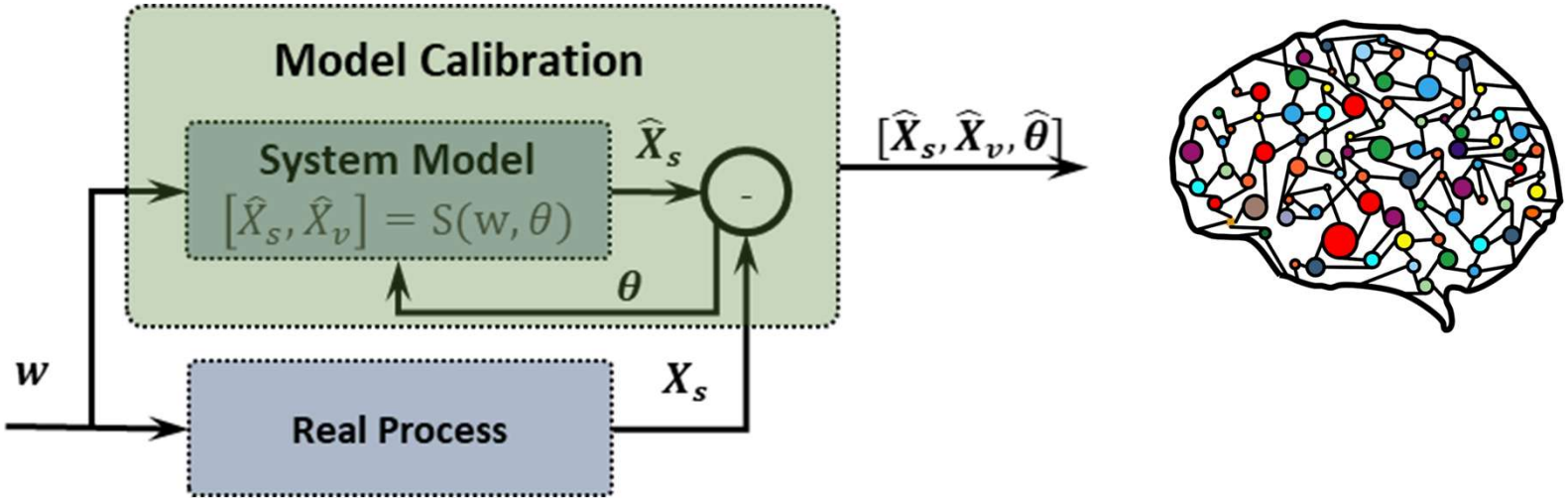







Arias Chao, M., Kulkarni, C., Goebel, K., & Fink, O. (2019). Hybrid deep fault detection and isolation: Combining deep neural networks and system performance modelsInternational Journal of Prognostics and Health Management

# Calibration-based hybrid framework



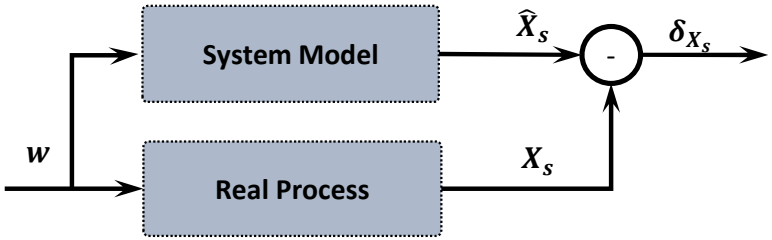



Arias Chao, M., Kulkarni, C., Goebel, K., & **Fink, O**. (2019). Hybrid deep fault detection and isolation: Combining deep neural networks and system performance modelsInternational Journal of Prognostics and Health Management

# System Model - Degradation Features

Degradation features provided by the system model:

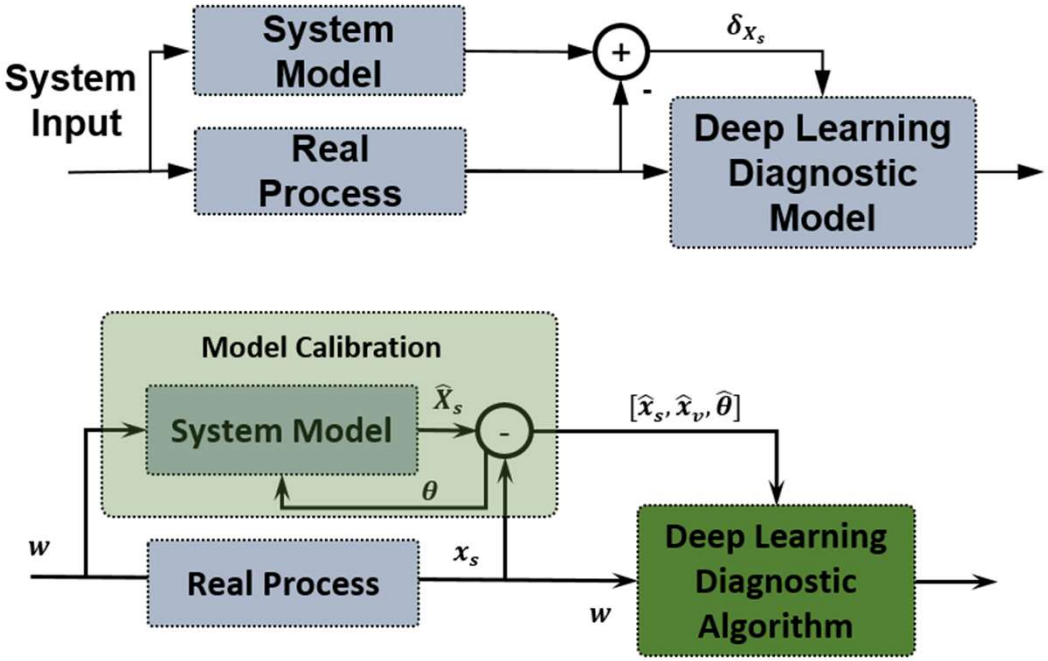
- Residual $\delta_{X_s} = \hat{X}_s - X_s$
  - Direct computation





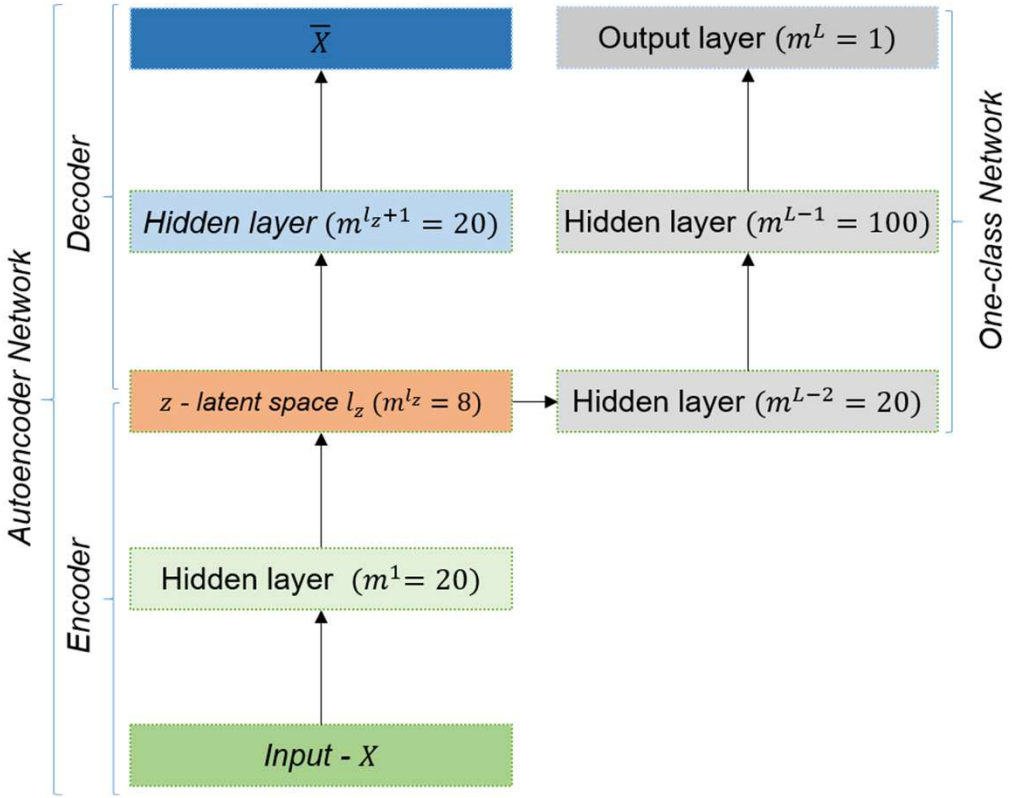

Arias Chao, M., Kulkarni, C., Goebel, K., & Fink, O. (2019). Hybrid deep fault detection and isolation: Combining deep neural networks and system performance modelsInternational Journal of Prognostics and Health Management

# Two tested setups

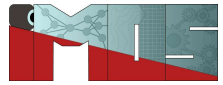



Arias Chao, M., Kulkarni, C., Goebel, K., & Fink, O. (2019). Hybrid deep fault detection and isolation: Combining deep neural networks and system performance modelsInternational Journal of Prognostics and Health Management

# Applied network architecture



# Detection Accuracy in [%]

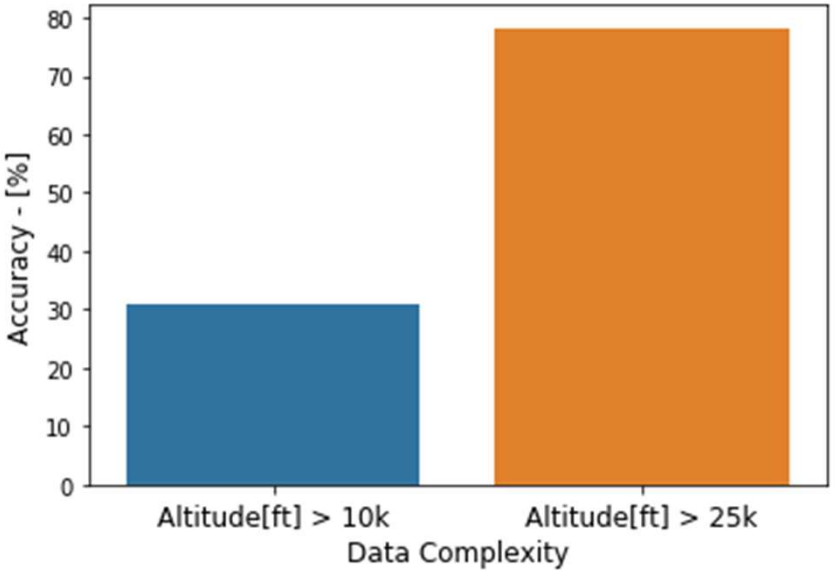


| | | AE | VAE | OC-SVM |
|---|---|---|---|---|
| Data-driven | $[W, X_S]$ | 24.5 | 12.9 | 10.0 |
| Residual-based | $[W, X_S, \delta_{X_S}]$ | 98.7 | 99.3 | 79.5 |
| Hybrid (calibration-based) | $[W, \widehat{X_S}, \widehat{X_V}, \hat{\theta}]$ | 100.0 | 97.5 | 96.2 |



Arias Chao, M., Kulkarni, C., Goebel, K., & Fink, O. (2019). Hybrid deep fault detection and isolation: Combining deep neural networks and system performance modelsInternational Journal of Prognostics and Health Management

# Influence of the data complexity (data-driven model)



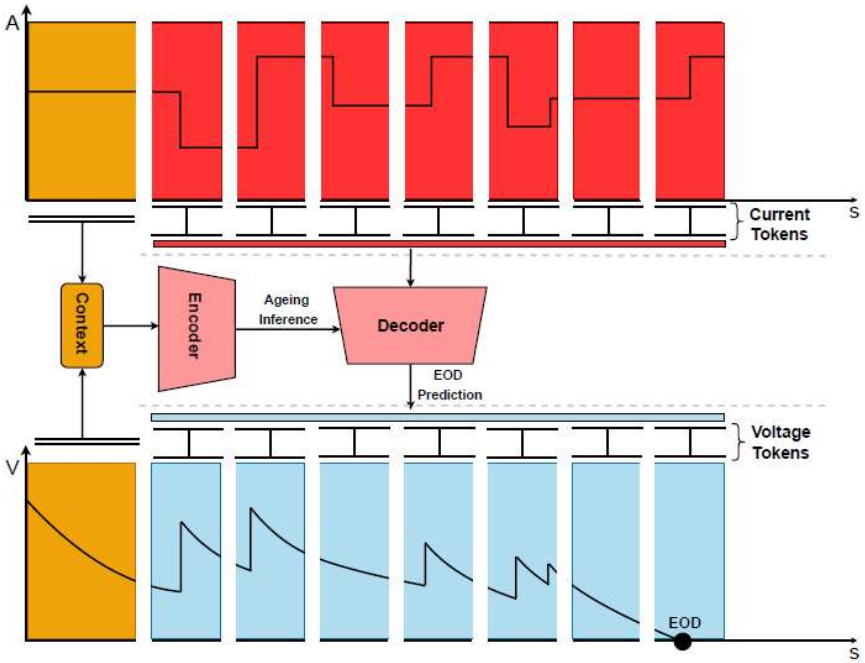

Arias Chao, M., Kulkarni, C., Goebel, K., & Fink, O. (2019). Hybrid deep fault detection and isolation: Combining deep neural networks and system performance modelsInternational Journal of Prognostics and Health Management

# Fault isolation (with a high fault intesity)

| | | AE |
|---|---|---|
| Data-driven | $[W, X_S]$ | Physical core speed<br>Static pressure at HPC outlet |
| Residual-based | $[W, X_S, \delta_{X_S}]$ | Delta physical core speed<br>Delta total temp. at HPC outlet<br>Delta total temp. at LPC outlet<br>Delta total temp. at HPT outlet<br>Delta total temp. at LPT outlet |
| Hybrid (calibration-based) | $[W, \widehat{X_S}, \widehat{X_V}, \hat{\theta}]$ | HPC efficiency modifier |



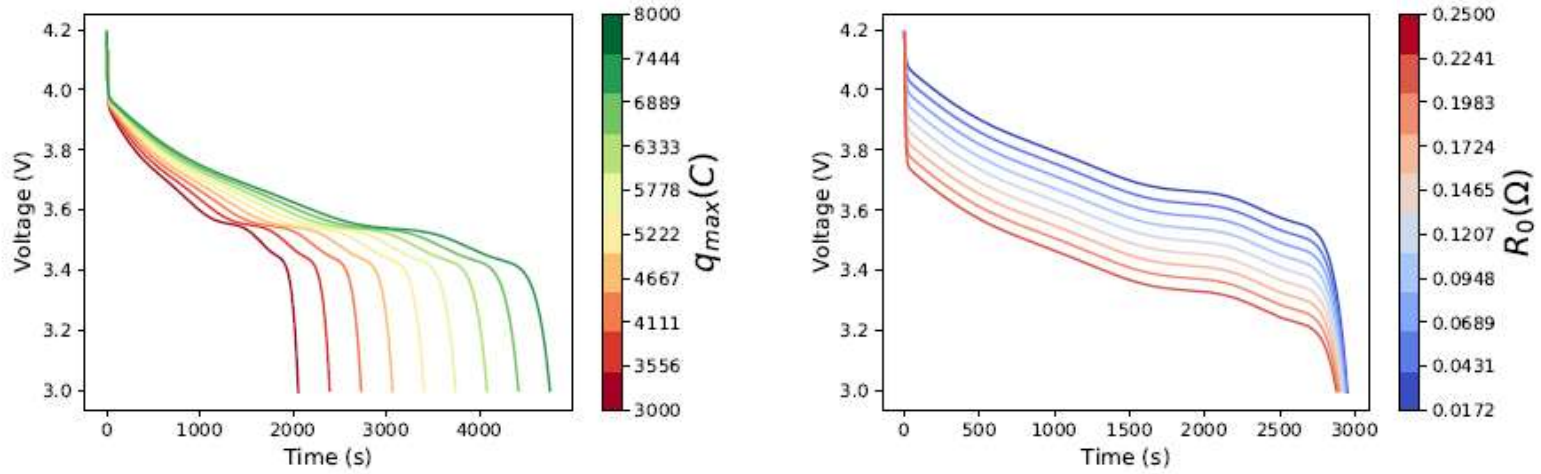
Arias Chao, M., Kulkarni, C., Goebel, K., & Fink, O. (2019). Hybrid deep fault detection and isolation: Combining deep neural networks and system performance modelsInternational Journal of Prognostics and Health Management




# Ageing-aware Battery Discharge Prediction

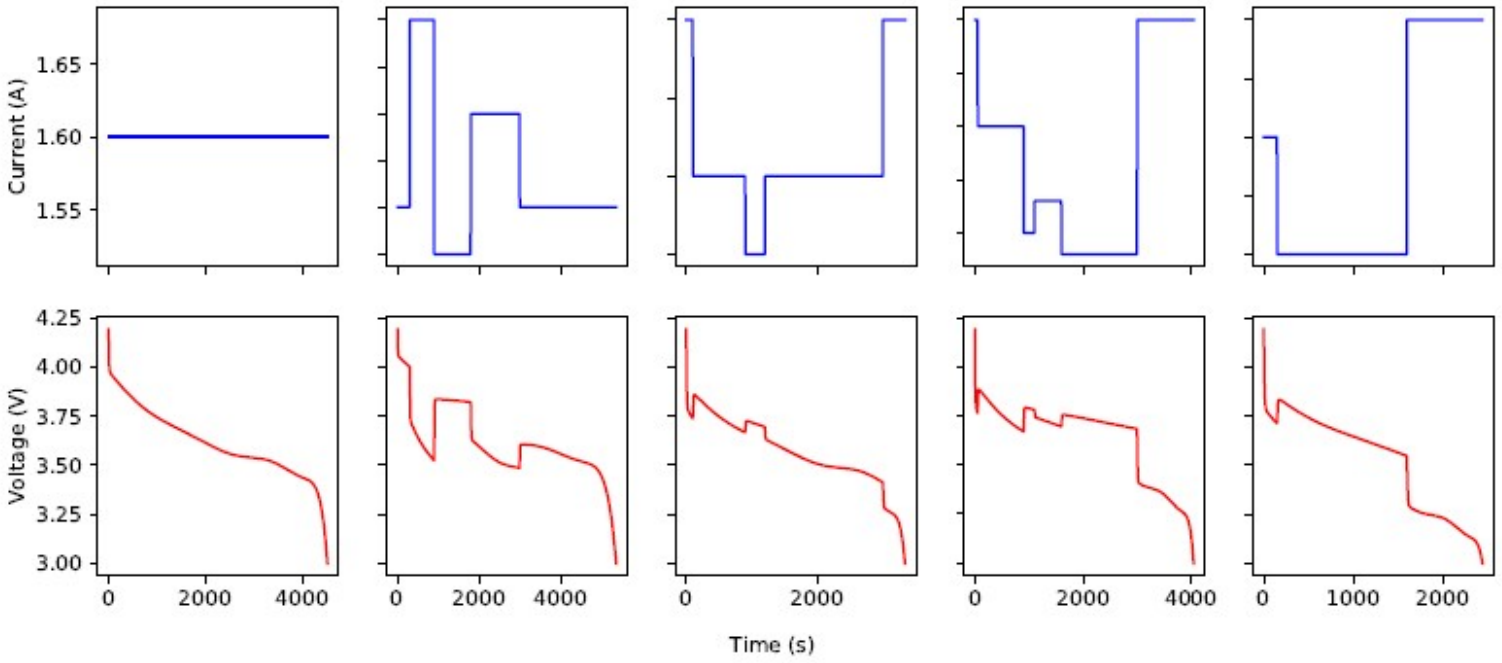
# Degradation of Li-Ion batteries → importance of precise planning

# Effect of varying the degradation parameters on the voltage discharge curve of a Li-ion battery



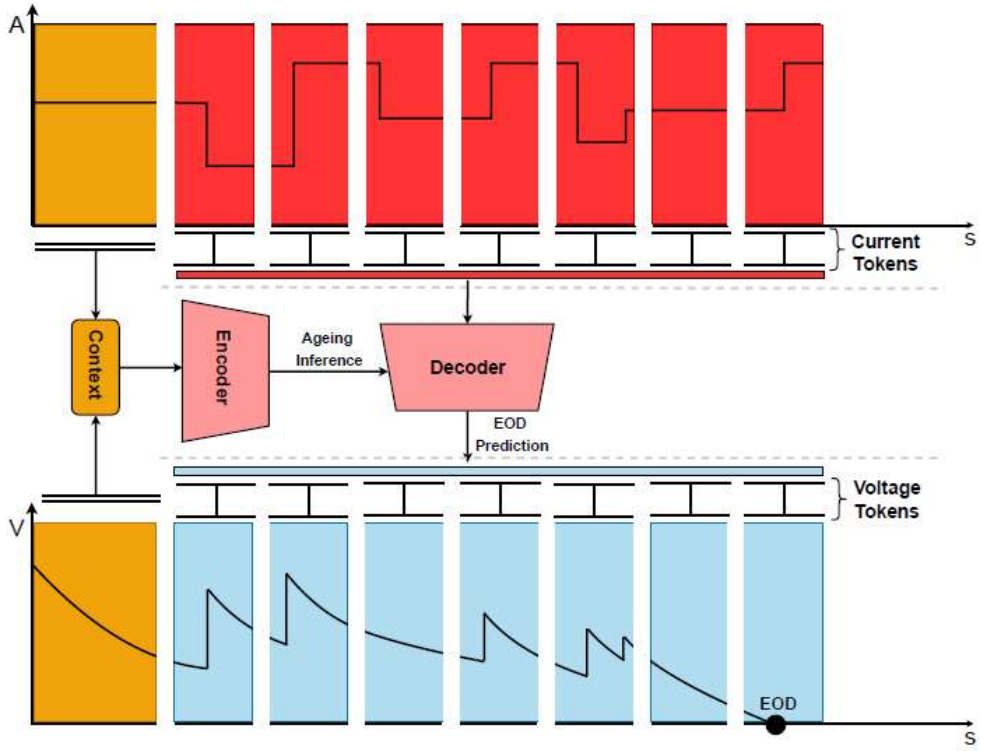

Biggio, L., Bendinelli, T., Kulkarni, C., & **Fink, O**. (2022). Dynaformer: A Deep Learning Model for Ageing-aware Battery Discharge Prediction, under review

# Discharge behaviors with respect to the different load profiles



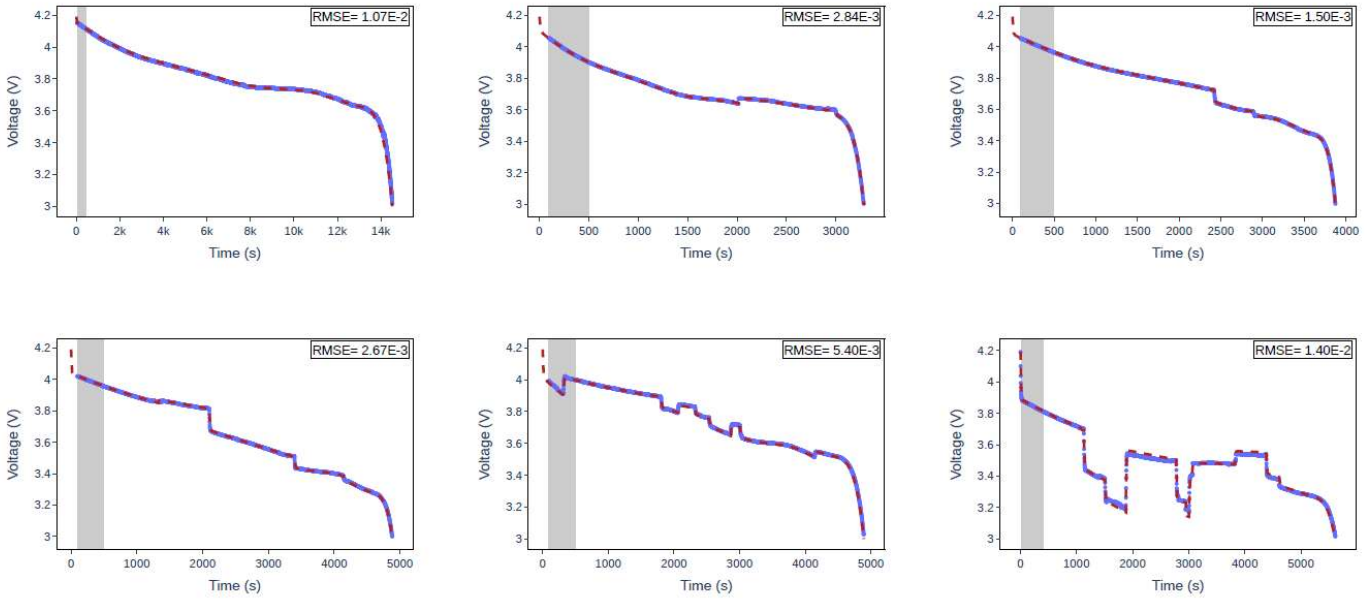

Biggio, L., Bendinelli, T., Kulkarni, C., & **Fink, O**. (2022). Dynaformer: A Deep Learning Model for Ageing-aware Battery Discharge Prediction, under review

# Transformer-based long-term battery discharge prediction → Dynaformer




Biggio, L., Bendinelli, T., Kulkarni, C., & **Fink, O**. (2022). Dynaformer: A Deep Learning Model for Ageing-aware Battery Discharge Prediction, under review

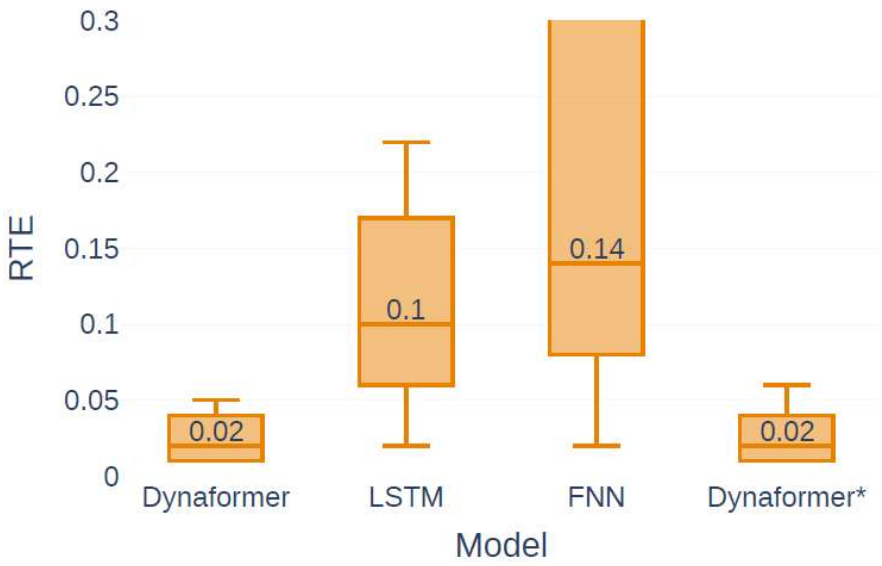
# Selected results



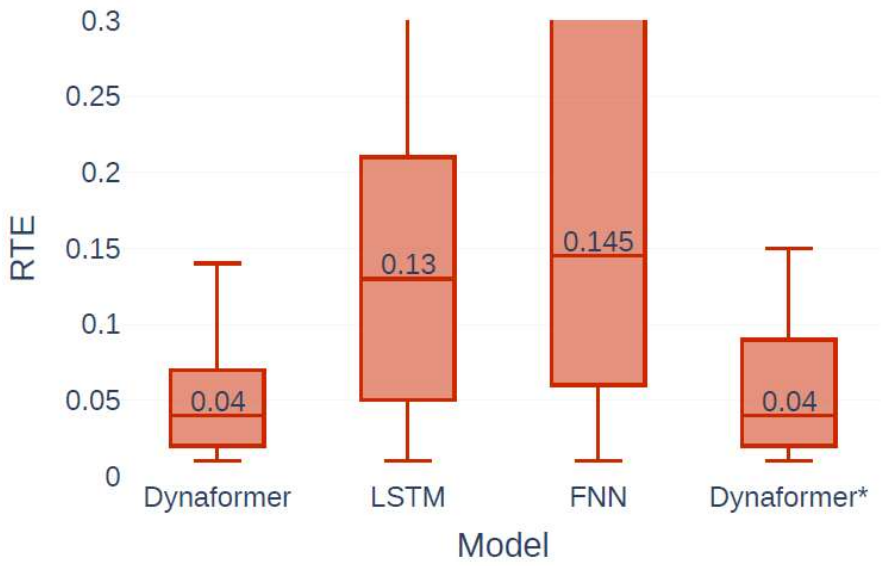

Biggio, L., Bendinelli, T., Kulkarni, C., & **Fink, O**. (2022). Dynaformer: A Deep Learning Model for Ageing-aware Battery Discharge Prediction, under review

# Performance without fine-tuning

Interpolation




Extrapolation

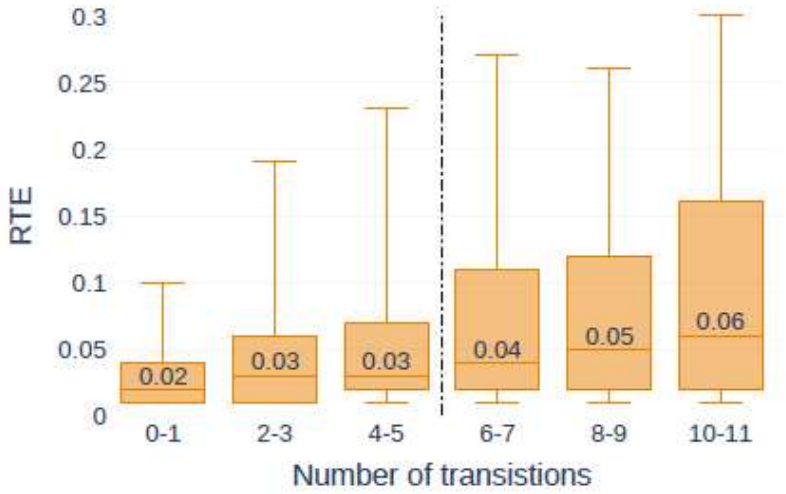



Dynaformer* → trained with variable current profiles

*RTE = relative temporal error

Biggio, L., Bendinelli, T., Kulkarni, C., & **Fink, O**. (2022). Dynaformer: A Deep Learning Model for Ageing-aware Battery Discharge Prediction, under review

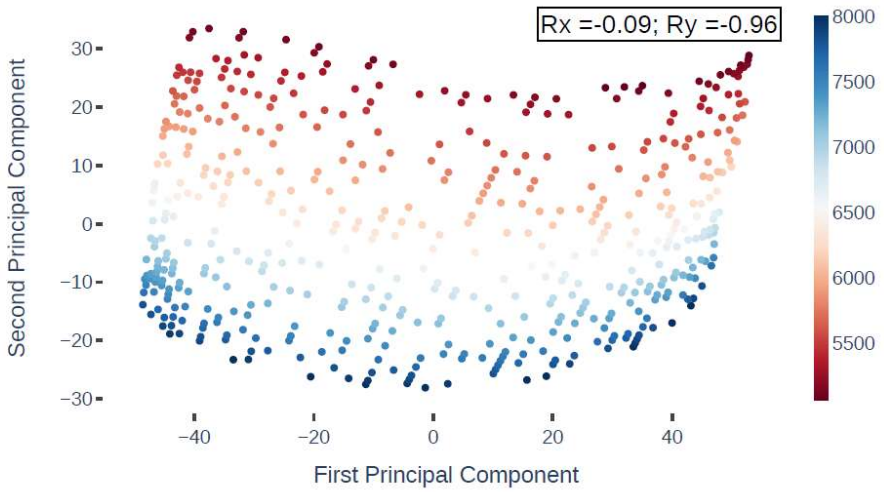
# Performance dependent on the complexity of the the load profiles



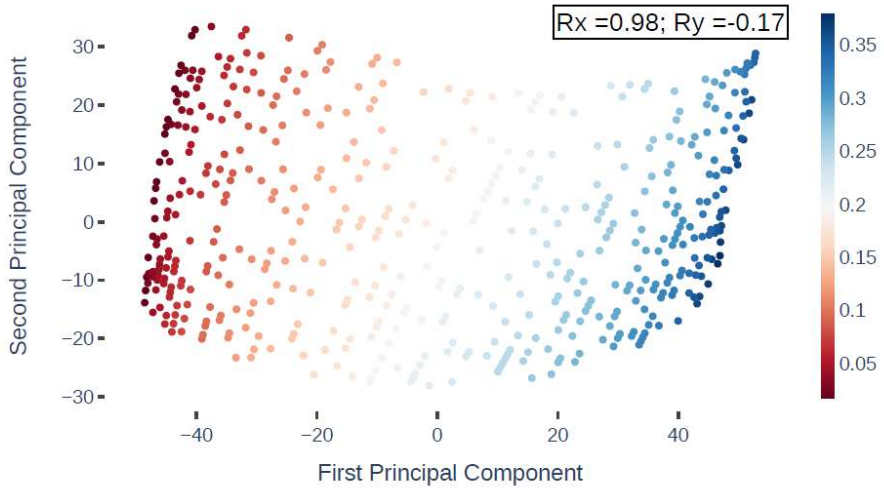

Biggio, L., Bendinelli, T., Kulkarni, C., & **Fink, O**. (2022). Dynaformer: A Deep Learning Model for Ageing-aware Battery Discharge Prediction, under review

# Implicit learning of the degradation parameters in the latent space

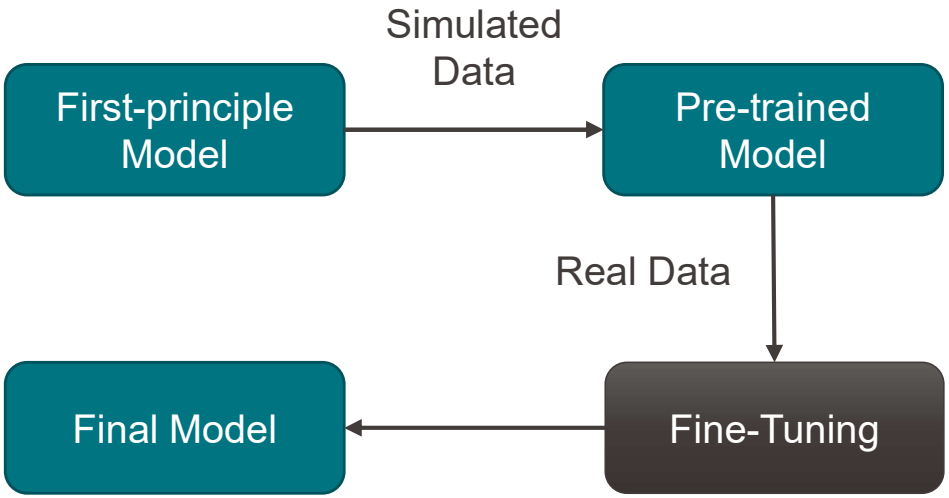
$q_{max}$



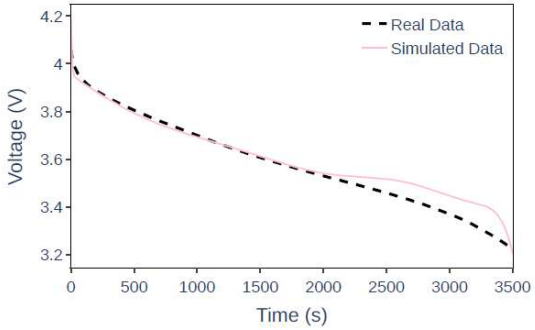

$R_0$

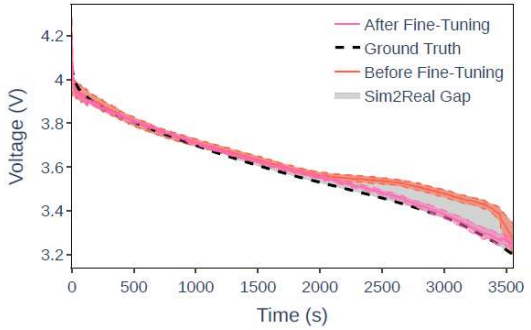


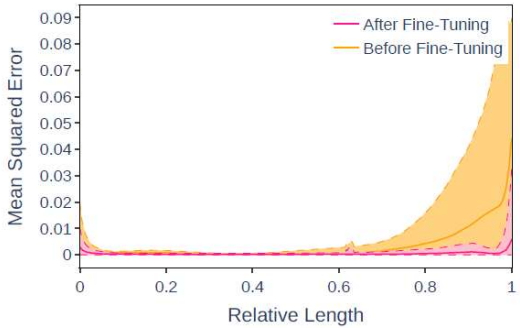
# Transfer learning

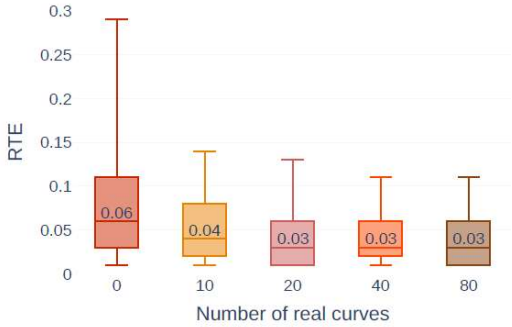


# Fine-tuning on real data




*RTE = relative temporal error

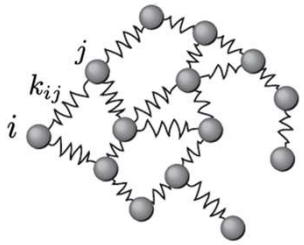
Biggio, L., Bendinelli, T., Kulkarni, C., & **Fink, O**. (2022). Dynaformer: A Deep Learning Model for Ageing-aware Battery Discharge Prediction, under review

# Graph Neural Networks

# Engineering Systems as Graphs

**Mechanical**

Spring–Mass, Bearings…

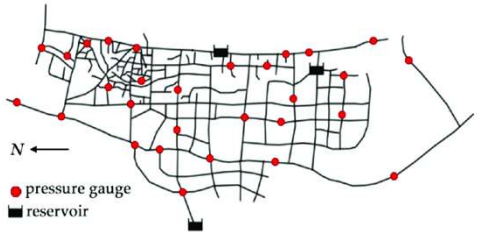


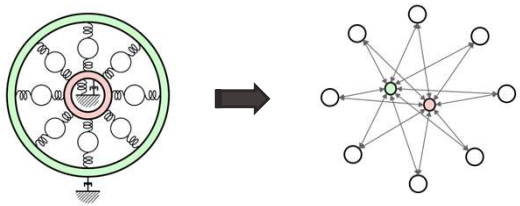

**Structural**

Bridges, Beams…

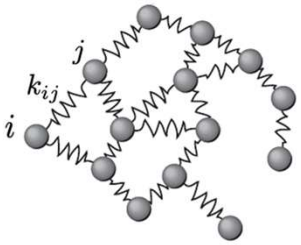
**Sensor networks**

Water, district heating…

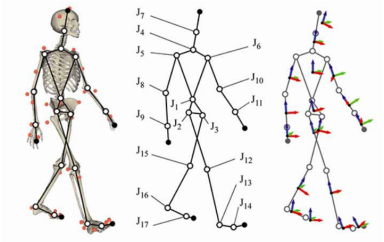


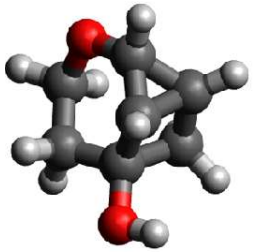
# Dynamical systems represented as graphs



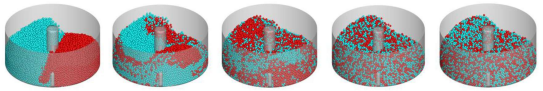

Spring-Mass Systems



Motion Dynamics



Molecular Dynamics



Particle Dynamics

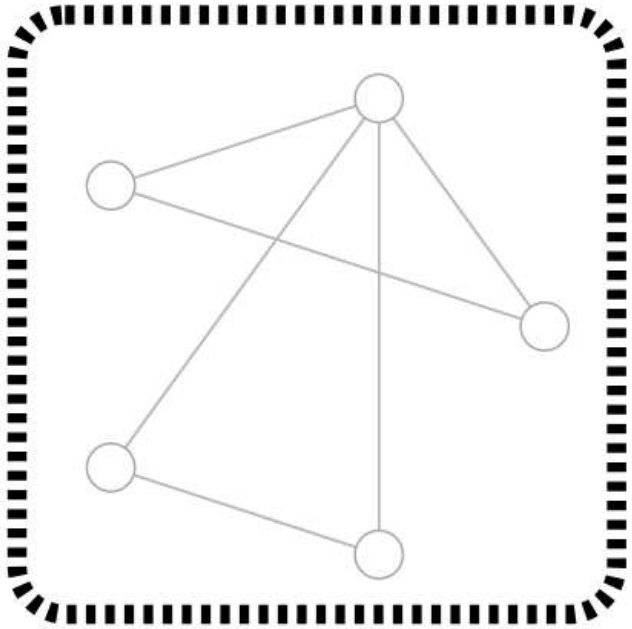
→ Simulate

→ Forecast

→ Detect anomalies / defects

→ Infer unobserved quantities

# Main properties of a graph



**V** Vertex (or node) attributes

e.g., node identity, number of neighbors

**E** Edge (or link) attributes and directions

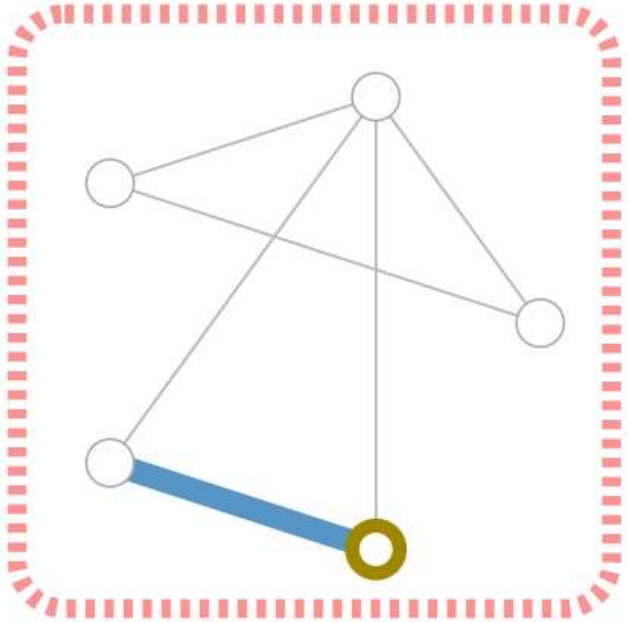
e.g., edge identity, edge weight

**U** Global (or master node) attributes

e.g., number of nodes, longest path

Source: B. Sanchez-Lengeling, E. Reif A. Pearce, A. B. Wiltschko

# Embeddings in a graph



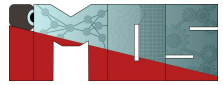
Vertex (or node) embedding

Edge (or link) attributes and embedding

Global (or master node) embedding

Source: B. Sanchez-Lengeling, E. Reif A. Pearce, A. B. Wiltschko

# Why Combine Graphs & Time?



- Many systems are dynamic: power grids, transportation, manufacturing
- Need to capture:
  - Spatial dependencies between components
  - Temporal evolution of system behavior

# Time-then-Graph

- Process:

1. Temporal model per node (e.g., LSTM, 1DCNN)
2. GNN to capture spatial structure

- Pros:
- Great for fixed graphs
- Captures rich temporal patterns

- Use case: Monitoring degradation signals at fixed sensor locations

# Graph-then-Time

- Process:

1. GNN applied at each timestep
2. Outputs passed to temporal model (e.g., LSTM/Transformer)

- Pros:
- Modular
- Handles dynamic graphs better

- Use case: Tracking how degradation spreads over time

# Joint Spatiotemporal GNNs

- Example Models: ST-GCN, TGAT

- Highlights:
- Learn over space and time simultaneously
- Graph includes spatial & temporal edges

- Use case: Complex interdependent systems (e.g., power networks, traffic)

# Spatiotemporal GNN Strategies

| Strategy | Description | Use Case |
|---|---|---|
| Time-then-Graph | Model time at nodes → then graph | Rich node dynamics |
| Graph-then-Time | Model graph snapshots → then time | Dynamic topology |
| Spatiotemporal GNNs | Jointly model space & time | Interdependent systems |
| Dynamic GNNs | Time-varying graph structure | Evolving network connections |

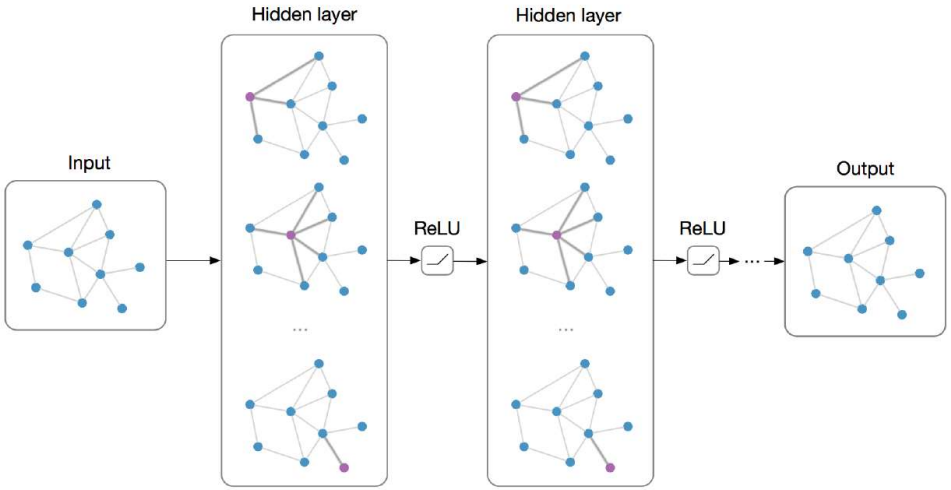
# Comparison of Spatiotemporal GNN Strategies

| Strategy | Spatial | Temporal | Graph Evolves | Best For |
|---|---|---|---|---|
| Time-then-Graph | ✔ After time | ✔ | ✘ | Rich node-level time dynamics |
| Graph-then-Time | ✔ First | ✔ | ✔ | Dynamic topologies |
| Spatiotemporal GNNs | ✔ Joint | ✔ Joint | ✔ | Highly coupled systems |
| Dynamic GNNs | ✔ Variable | ✔ | ✔ | Evolving graphs |

# Choosing the Right Architecture

- Considerations:
  - Static or dynamic topology?
  - Node-level or system-level reasoning?
  - Is interpretability important?
  - Data availability over time?

# Graph Neural Networks




**Main Idea**: Pass massages between pairs of nodes and agglomerate

**Alternative Interpretation**: Pass massages between nodes to refine node (and possibly edge) representations

Source: T. Kipf

# Message passing GNNs

1. **Receiving messages** from its neighbors.
2. **Aggregating** those messages.
3. **Updating** its state using a learnable function.

Formally, for each node $v$ at layer $t$:

$$\text{Message: } m_v^{(t)} = \text{AGGREGATE}\left(\{\phi(h_u^{(t-1)}, h_v^{(t-1)}, e_{uv})\}_{u \in \mathcal{N}(v)}\right)$$

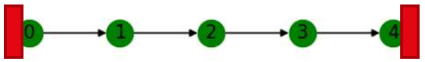
$$\text{Update: } h_v^{(t)} = \psi(h_v^{(t-1)}, m_v^{(t)})$$

- $h_v^{(t)}$: hidden state of node $v$
- $\mathcal{N}(v)$: neighbors of $v$
- $e_{uv}$: edge feature between nodes $u$ and $v$
- $\phi$, $\psi$: learnable functions (MLPs or simple linear layers)
- AGGREGATE: e.g., sum, mean, max

# Introduction to message passing GNNs →Graph Neural Simulators (GNS)

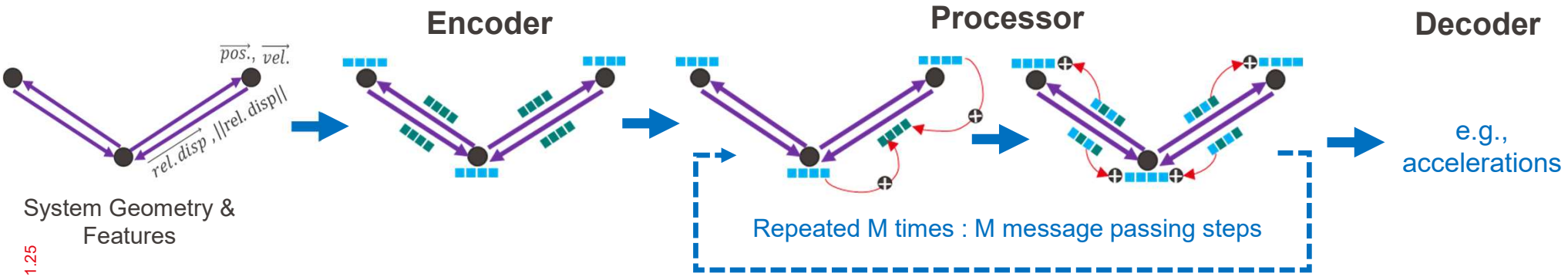
## Example : Spring Mass Systems

- Each node represents the mass
- Each edge represents the spring




## GNS : Message Passing on Graphs

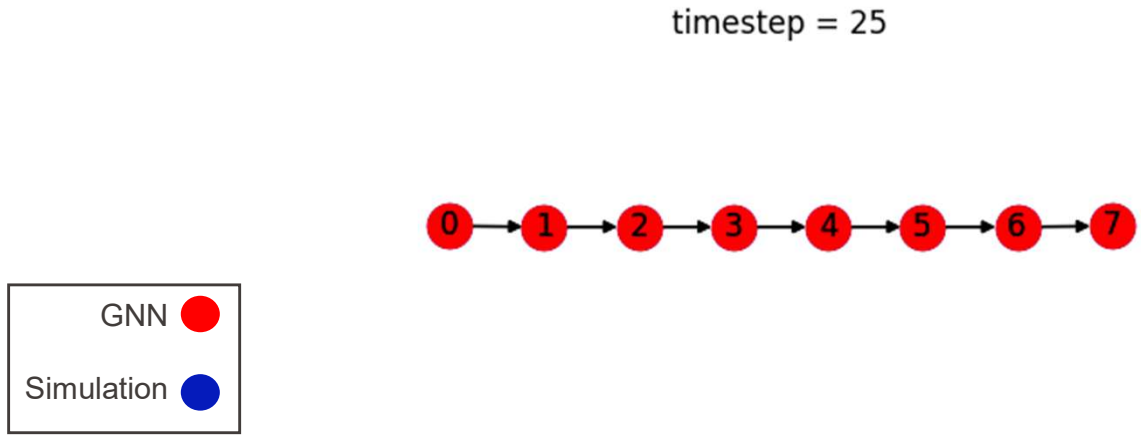
$$Loss = ||\vec{a_t}^{*} - \vec{a_t}^{g.t.}||$$




* Sanchez-Gonzalez et al. Learning to simulate complex physics with graph networks, ICML, 2020

# Learning dynamics of a Spring Mass System

**Example : Spring Mass Systems learned dynamics by GNN**

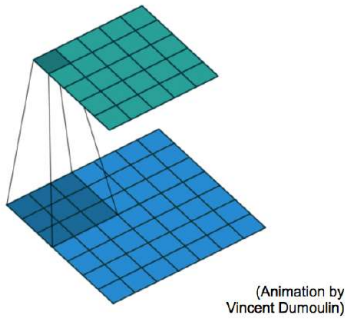



Train : 4, 5, 6, 7, 9, 10 masses
Test : 8 masses **(out of training)**

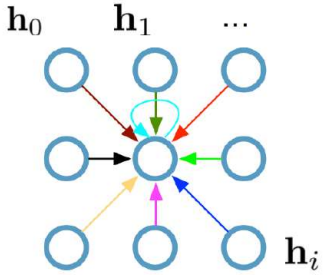
Source: V. Sharma

# Recap: Convolutional Neural Networks (CNNs) on Grids

**Single CNN layer with 3x3 filter:**



(Animation by Vincent Dumoulin)




**Update for a single pixel:**

- Transform messages individually $\mathbf{W}_i \mathbf{h}_i$
- Add everything up $\sum_i \mathbf{W}_i \mathbf{h}_i$

$\mathbf{h}_i \in \mathbb{R}^F$ are (hidden layer) activations of a pixel/node

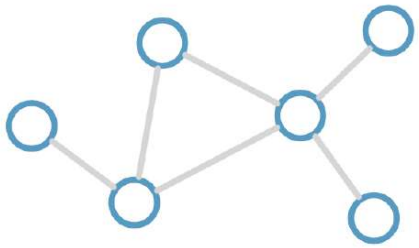
**Full update:**

$$\mathbf{h}_4^{(l+1)} = \sigma\left(\mathbf{W}_0^{(l)}\mathbf{h}_0^{(l)} + \mathbf{W}_1^{(l)}\mathbf{h}_1^{(l)} + \cdots + \mathbf{W}_8^{(l)}\mathbf{h}_8^{(l)}\right)$$

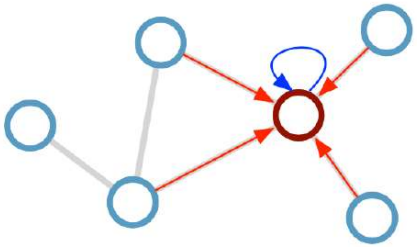
Source: T. Kipf

# Graph Convolutional Networks (GCNs)

Consider this undirected graph:



Calculate update for node in red:



**Desirable properties:**

- Weight sharing over all locations
- Invariance to permutations
- Linear complexity O(E)
- Applicable both in transductive and inductive settings

**Update rule:**

$$\mathbf{h}_i^{(l+1)} = \sigma\left(\mathbf{h}_i^{(l)}\mathbf{W}_0^{(l)} + \sum_{j\in\mathcal{N}_i}\frac{1}{c_{ij}}\mathbf{h}_j^{(l)}\mathbf{W}_1^{(l)}\right)$$

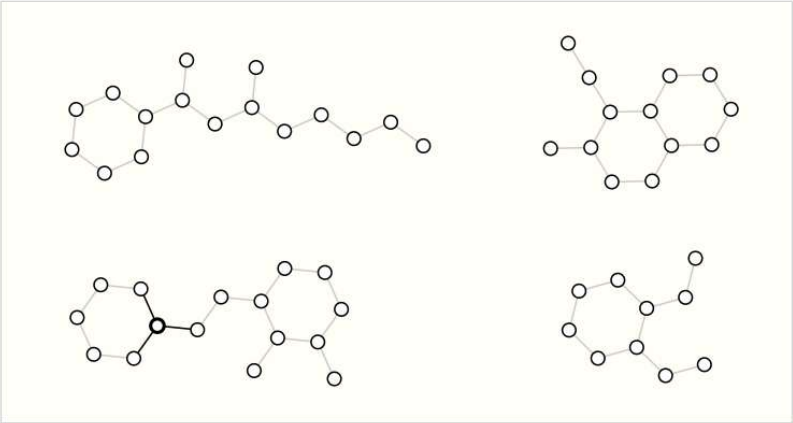
**Scalability: subsample messages** [Hamilton et al., NIPS 2017]

$\mathcal{N}_i$ : neighbor indices $c_{ij}$ : norm. constant (fixed/trainable)

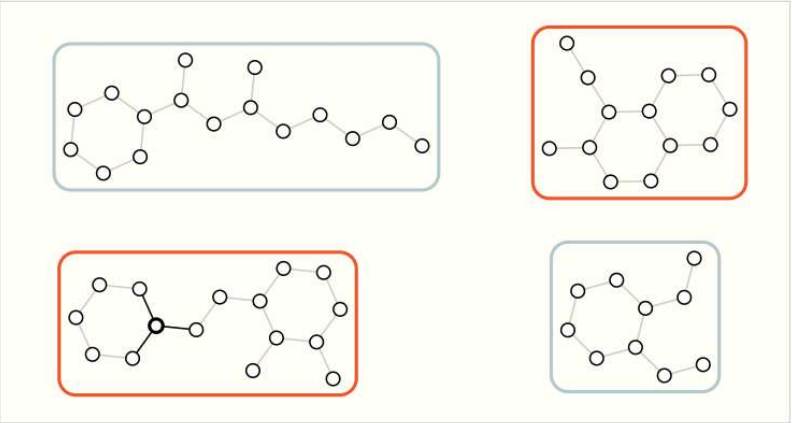
Source: T. Kipf

# Graph-level tasks

- Property of the entire graph



**Input:** graphs

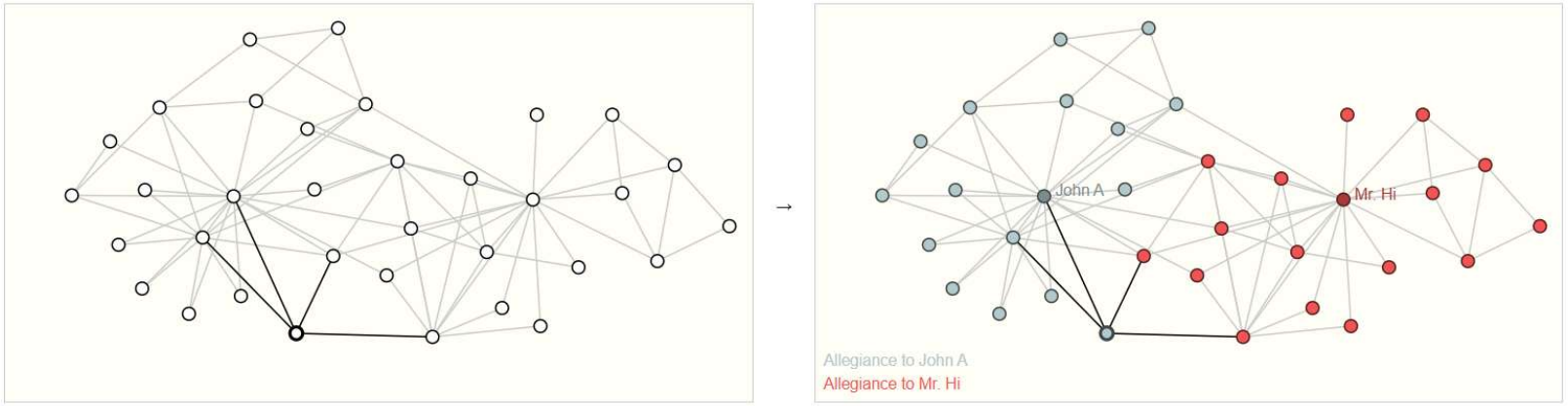


**Output:** labels for each graph, (e.g., "does the graph contain two rings?")

Source: B. Sanchez-Lengeling, E. Reif A. Pearce, A. B. Wiltschko

# Node-level tasks

- Predicting the identity or role of each node within a graph



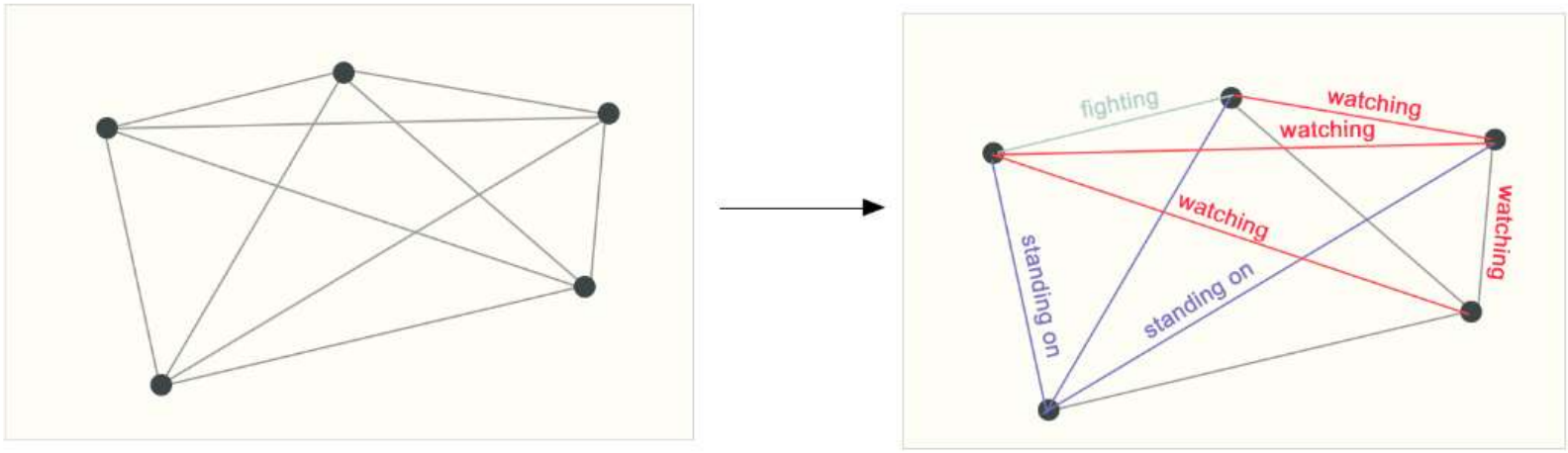

**Input:** graph with unlabled nodes

**Output:** graph node labels

Source: B. Sanchez-Lengeling, E. Reif A. Pearce, A. B. Wiltschko

# Edge-level tasks

- Edge prediction



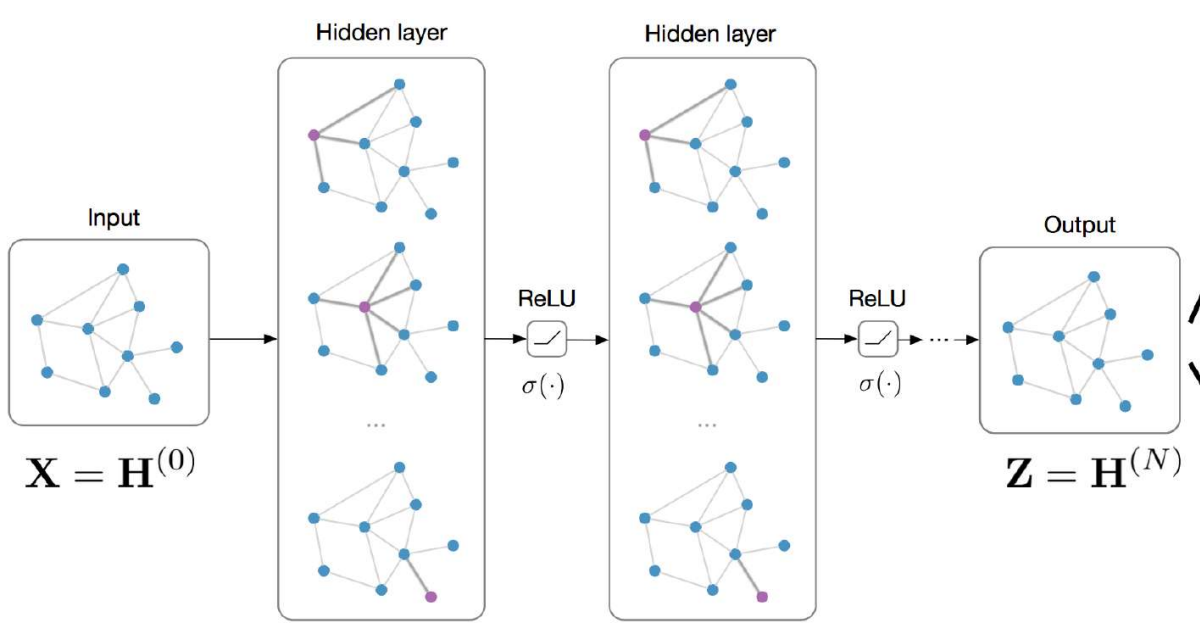

**Input:** fully connected graph, unlabeled edges

**Output:** labels for edges

Source: B. Sanchez-Lengeling, E. Reif A. Pearce, A. B. Wiltschko

# Classification and Link Prediction with GNNs / GCNs

**Input**: Feature matrix $\mathbf{X} \in \mathbb{R}^{N \times E}$, preprocessed adjacency matrix $\hat{\mathbf{A}}$




$$\mathbf{H}^{(l+1)} = \sigma\left(\hat{\mathbf{A}}\mathbf{H}^{(l)}\mathbf{W}^{(l)}\right)$$

**Node classification:**

$\mathrm{softmax}(\mathbf{z_n})$

e.g. Kipf & Welling (ICLR 2017)

**Graph classification:**

$\mathrm{softmax}(\sum_n \mathbf{z_n})$

e.g. Duvenaud et al. (NIPS 2015)

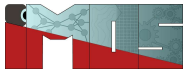
**Link prediction:**

$p(A_{ij}) = \sigma(\mathbf{z_i^T z_j})$

Kipf & Welling (NIPS BDL 2016)

**"Graph Auto-Encoders"**

Source: T. Kipf

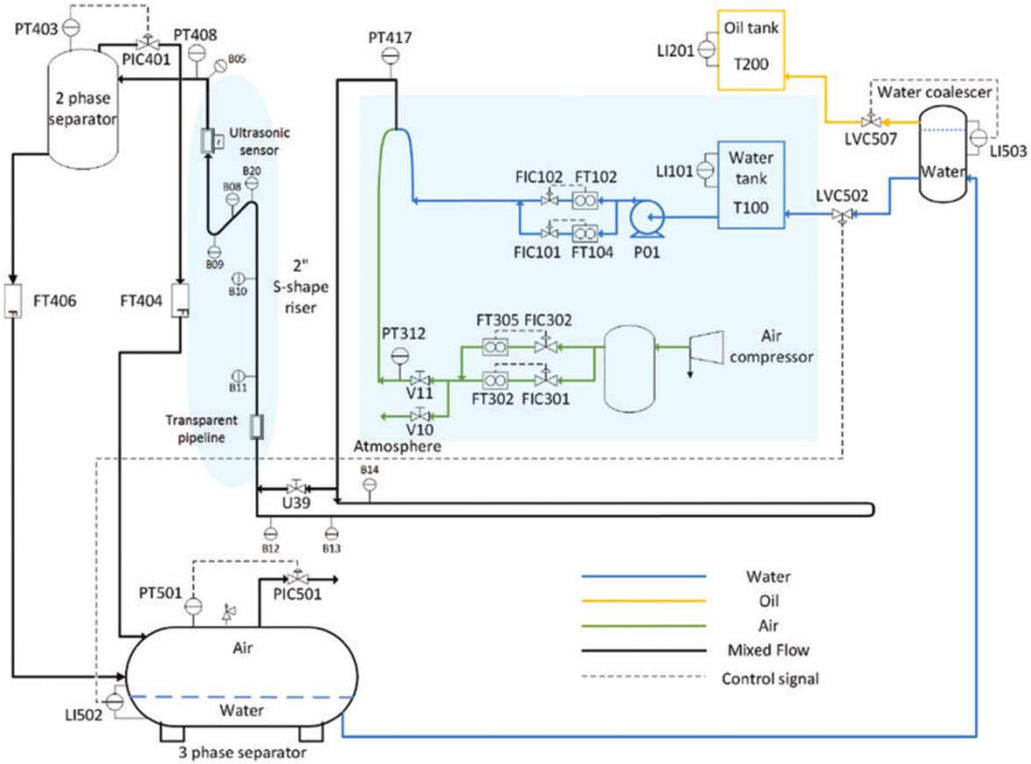


# Example: Industrial IoT systems

# Changing / evolving relationships between time series

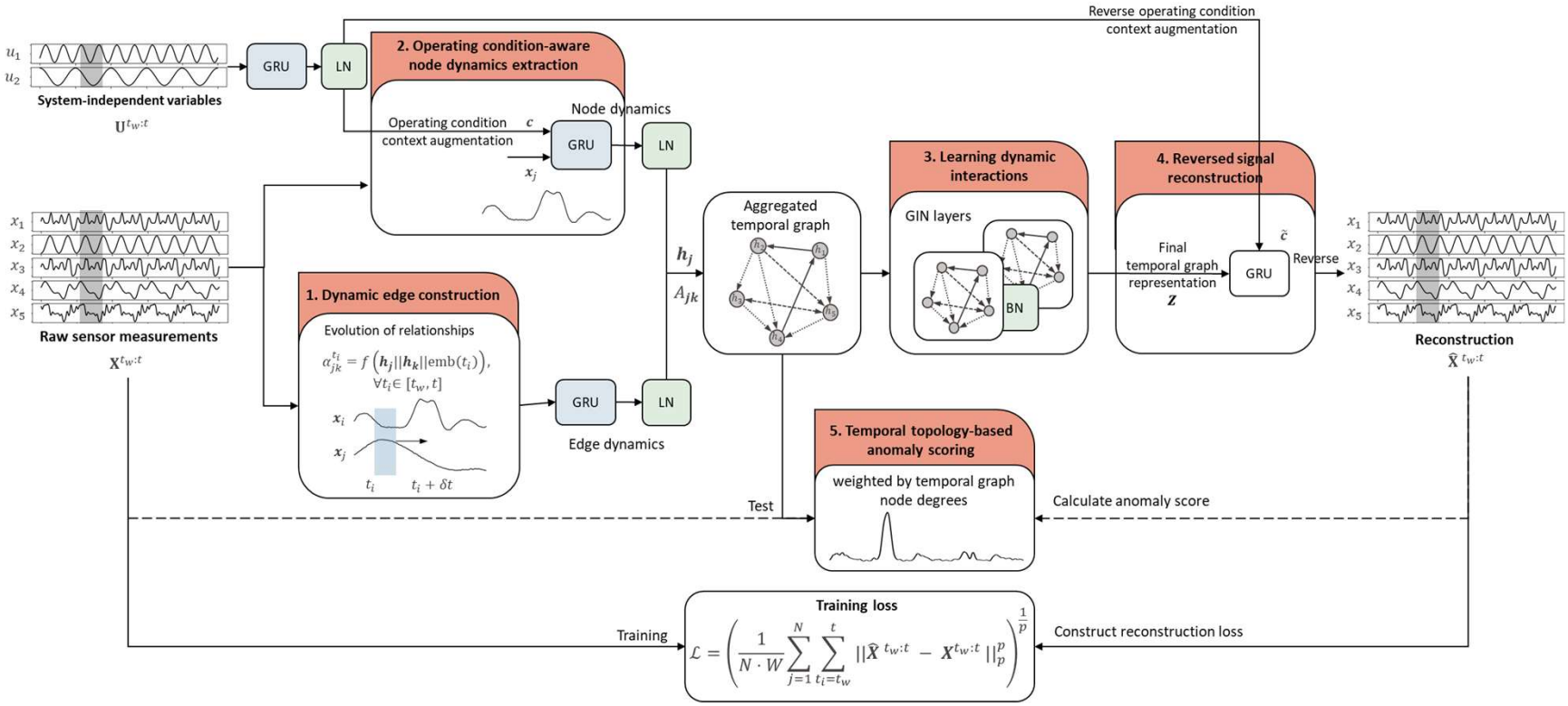
- Usually, graph topology between time series either given or derived once → static graph
- Changing environmental or operating conditions
- Impact of degradation
- Process optimization
- Maintenance
- …

# Anomaly detection in multi-flow facility



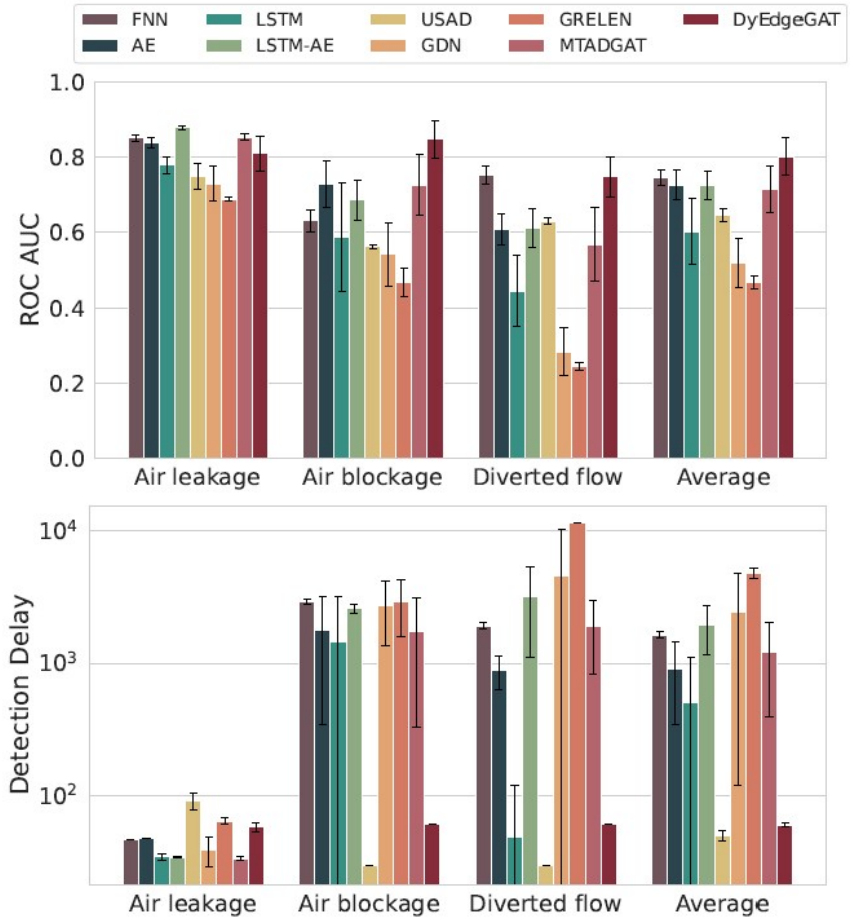

Source: Gedda, R., Beilina, L. and Tan, R., 2023. Change Point Detection for Process Data Analytics Applied to a Multiphase Flow Facility. *CMES-Computer Modeling in Engineering & Sciences*, *134*(3).

# DyEdgeGAT: Dynamic Edge via Graph Attention for Early Fault Detection in IIoT Systems



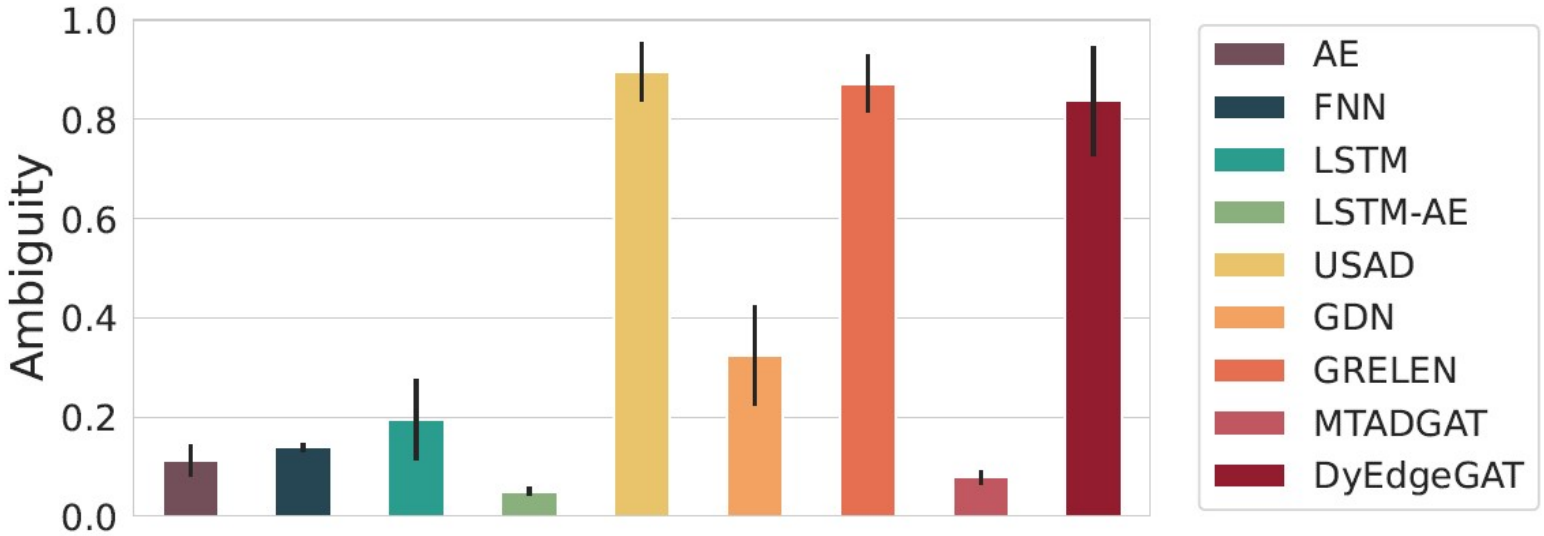

Zhao, M., & Fink, O. (2023). DyEdgeGAT: Dynamic Edge via Graph Attention for Early Fault Detection in IIoT Systems, IEEE Internet of Things Journal

# Dynamic Edge via Graph Attention (DyEdgeGAT)




Zhao, M., & Fink, O. (2023). DyEdgeGAT: Dynamic Edge via Graph Attention for Early Fault Detection in IIoT Systems, IEEE Internet of Things Journal

# Novel operating conditions



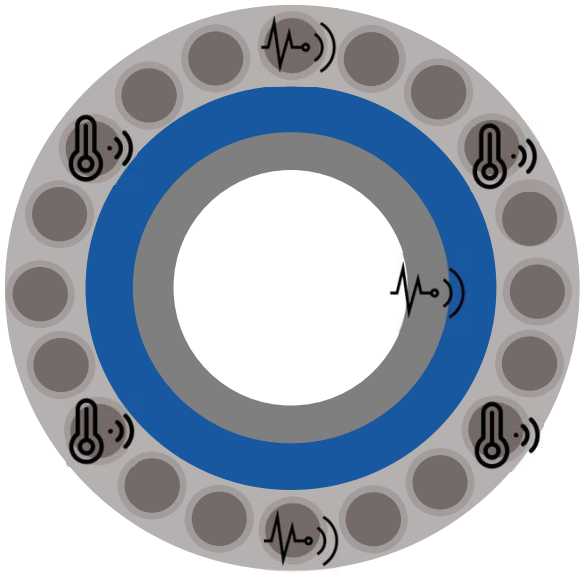

$$\text{Ambiguity} = 1 - 2 \cdot |\text{AUC} - 0.5|$$

Ambiguity quantifies the model's inability to differentiate between normal operations and novel conditions

Zhao, M., & Fink, O. (2023). DyEdgeGAT: Dynamic Edge via Graph Attention for Early Fault Detection in IIoT Systems, IEEE Internet of Things Journal

# Example: Heterogenous sensors

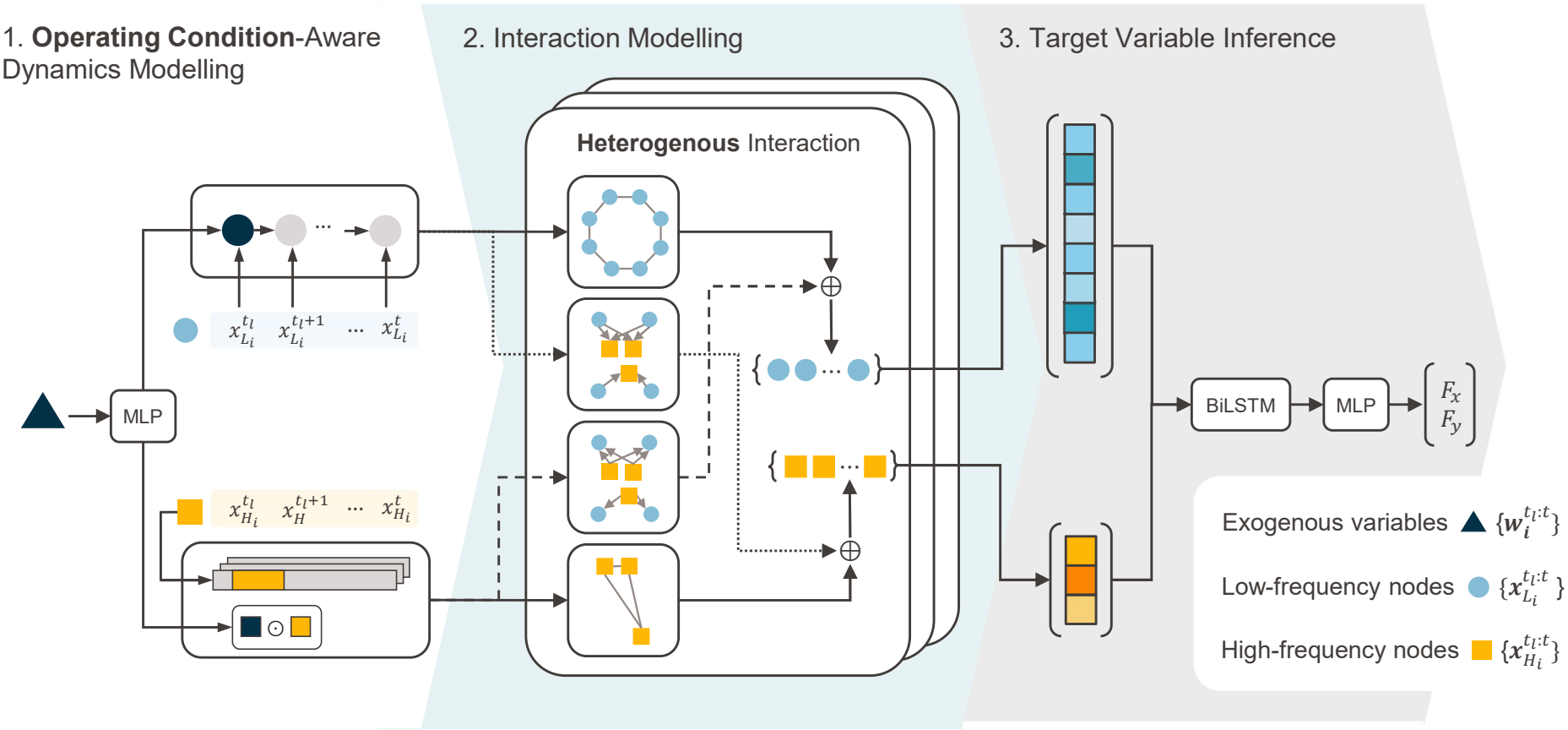
# Diverse sensing modalities provide complementary information



- **Vibration (high frequency)** → captures *instantaneous load variations* and *impacts*
- **Temperature (low frequency)** → reflects *gradual load accumulation* and *lubrication effects*

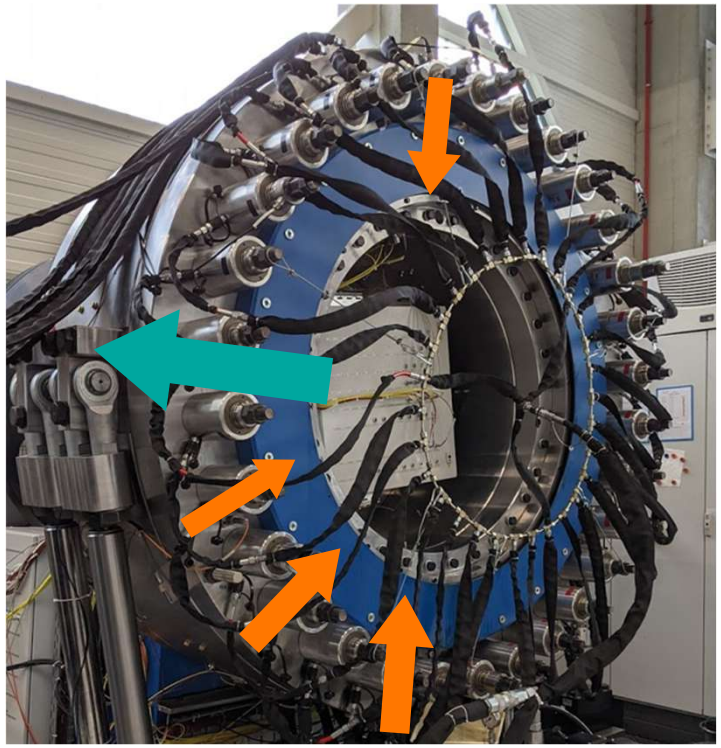
→ Combined, they enable **virtual inference of internal loads** otherwise unmeasurable

# Heterogeneous Temporal Graph Neural Network (HTGNN)




Zhao, M., Taal, C., Baggerohr, S. and Fink, O., 2025. Graph neural networks for virtual sensing in complex systems: Addressing heterogeneous temporal dynamics. Mechanical Systems and Signal Processing

# Test-rig Experimental Setup & Graph Construction



- RIR400 test rig
- Rotating inner ring
- Two single-row TRBs
  - OD: 2m
  - ID: 1.5m
- **55 operating conditions** with:
  - **Axial loads**: 1000kN – 8000kN
  - **Radial loads**: 100kN – 1000kN
  - Speeds: 10 rpm – 50 rpm

# Test-rig Experimental Setup & Graph Construction

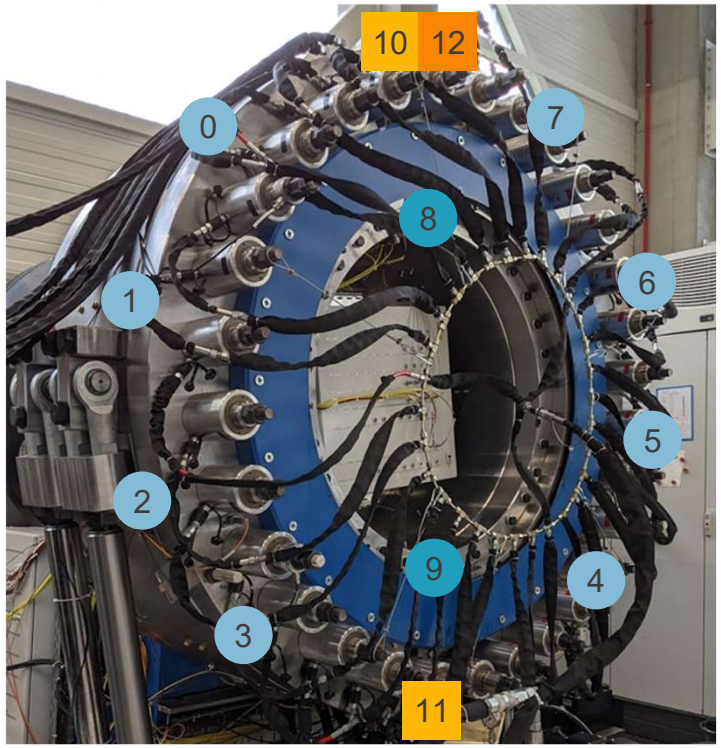
Graph construction based on **physical connectivity**

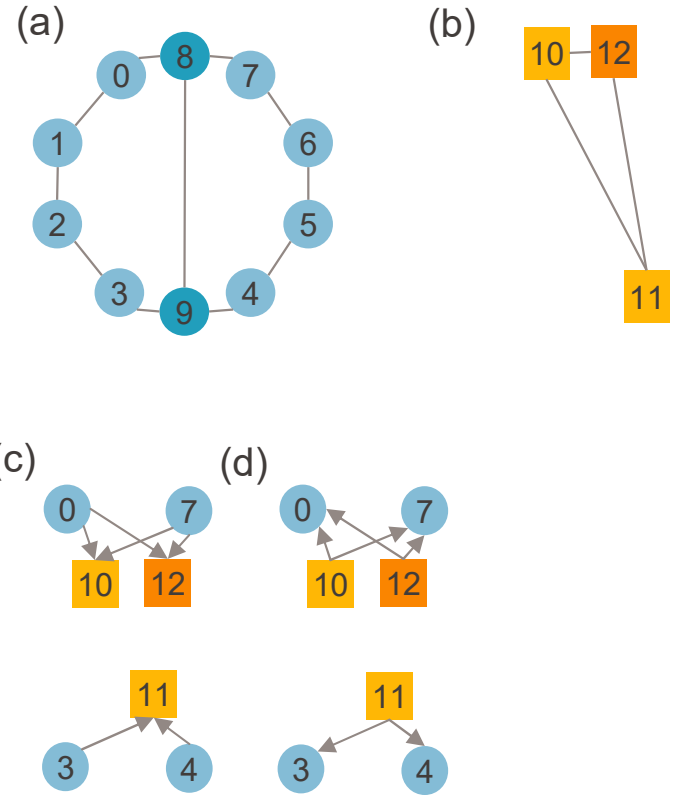
Temperature Outer Ring

Temperature Inner Ring

Axial Vibration

Radial Vibration



# Model performance (NRMSE) for bearing load predictions

| Model | Rotational speed | | | | | Avg. |
|---|---|---|---|---|---|---|
| | 10 | 20 | 30 | 40 | 50 | |
| BiLSTM | 0.021 ± 0.006 | 0.020 ± 0.008 | 0.019 ± 0.008 | 0.021 ± 0.008 | 0.015 ± 0.003 | 0.019 |
| 1D-CNN | 0.038 ± 0.008 | 0.037 ± 0.012 | 0.038 ± 0.011 | 0.025 ± 0.006 | 0.022 ± 0.004 | 0.032 |
| 1D-GCNN | 0.023 ± 0.005 | 0.029 ± 0.018 | 0.024 ± 0.008 | 0.017 ± 0.005 | 0.022 ± 0.005 | 0.023 |
| MTGAT | 0.024 ± 0.007 | 0.025 ± 0.009 | 0.020 ± 0.008 | 0.021 ± 0.009 | 0.015 ± 0.003 | 0.021 |
| **HTGNN (ours)** | **0.008** ± 0.004 | **0.004** ± 0.001 | **0.010** ± 0.010 | **0.005** ± 0.003 | **0.006** ± 0.002 | **0.007** |

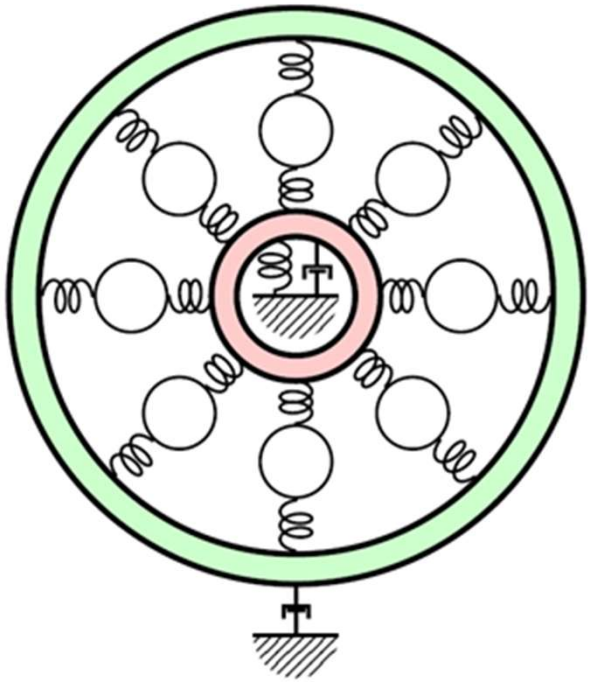
BiLSTM → Bidirectional Long Short-Term Memory network
1D-CNN → One-Dimensional Convolutional Neural Network
1D-GCNN → One-Dimensional Gated Convolutional Neural Network
MTGAT → Multivariate Time-series Graph Attention Network

Zhao, M., Taal, C., Baggerohr, S. and Fink, O., 2025. Graph neural networks for virtual sensing in complex systems: Addressing heterogeneous temporal dynamics. Mechanical Systems and Signal Processing

# Example: Modelling bearing as a graph

# Bearing Dynamic Model





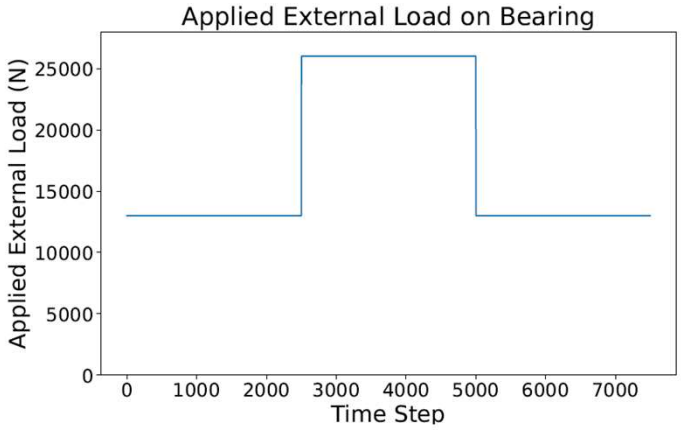
- 2D dynamic lumped bearing model, (based on P. K. Gupta, 1979)
- Differential equations of motion
- Lundberg & Palmgren model for Hertzian contact
- N209 CRB bearing (line contacts)
- Assumptions: rigid rings, zero-mass rolling elements

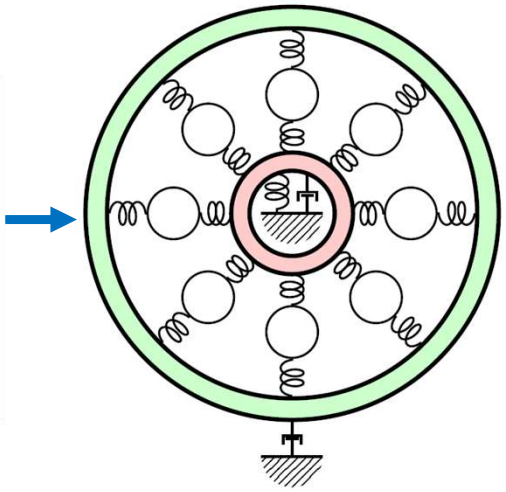
based on simple model but theory also applies to complex models!



Sharma, V., Ravesloot, J., Taal, C. and Fink, O., 2023. Graph Neural Networks for Dynamic Modeling of Roller Bearing, PHM Society conference 2023

# Bearing Dynamic Model: Training set



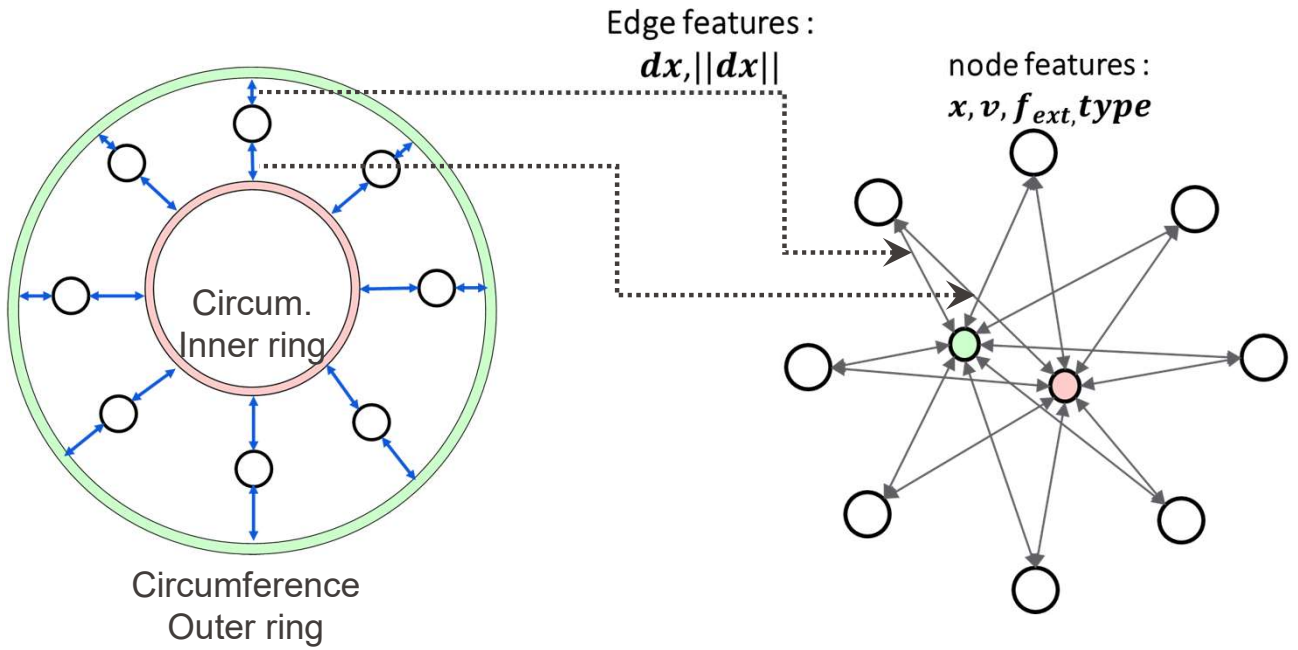



Positions $\vec{x}_i$
Velocities $\vec{v}_i$
Internal loads $\vec{f}_i$

- Load changes to generate dynamic step responses
- [1-20] kN OR loads, [13, 14, 15, 16] rollers
- (Rotation is not included yet)

Sharma, V., Ravesloot, J., Taal, C. and Fink, O., 2023. Graph Neural Networks for Dynamic Modeling of Roller Bearing, PHM Society conference 2023

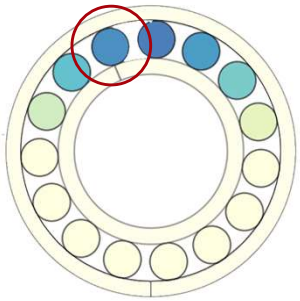
# GNN Inputs and Outputs

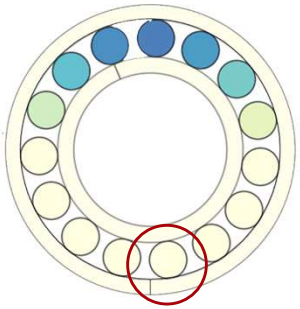


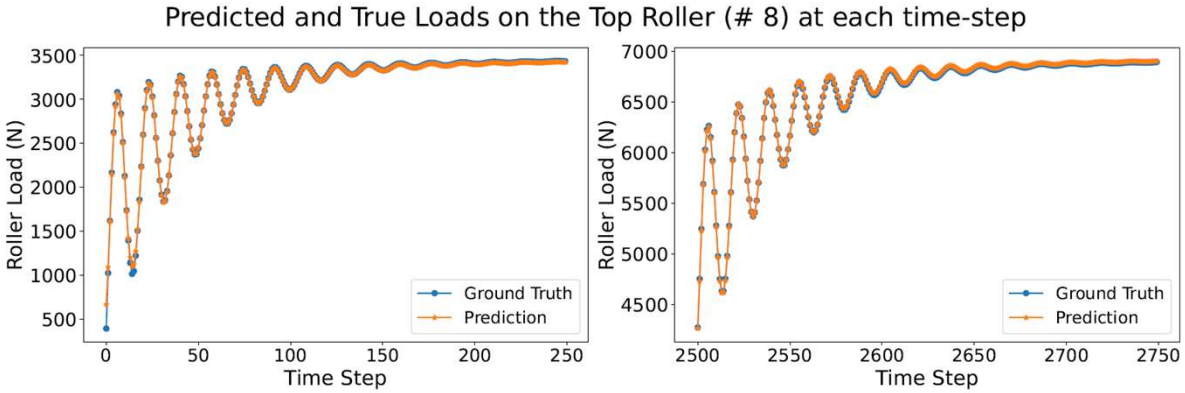

Sharma, V., Ravesloot, J., Taal, C. and Fink, O., 2023. Graph Neural Networks for Dynamic Modeling of Roller Bearing, PHM Society conference 2023

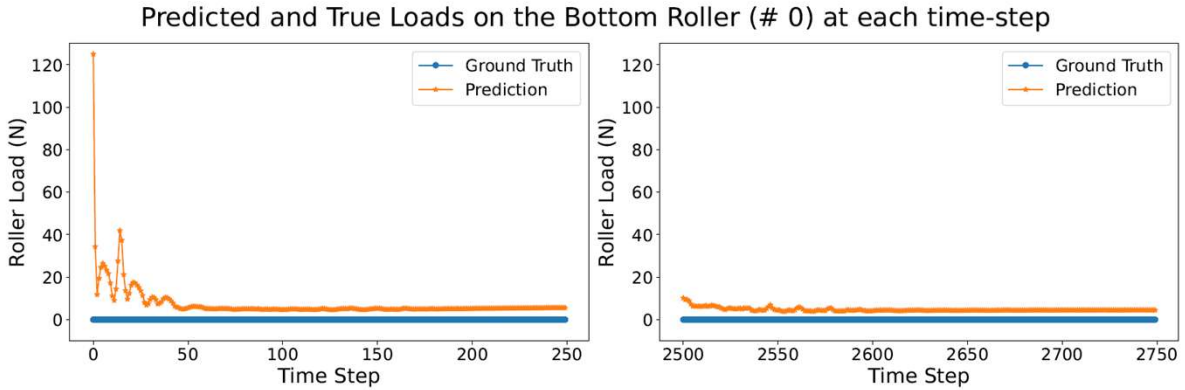
# Results: Generalizability

**Training : SKF N209 CRB (13, 14, 16 rollers)**
**Testing: 15 Rollers, 13kN**



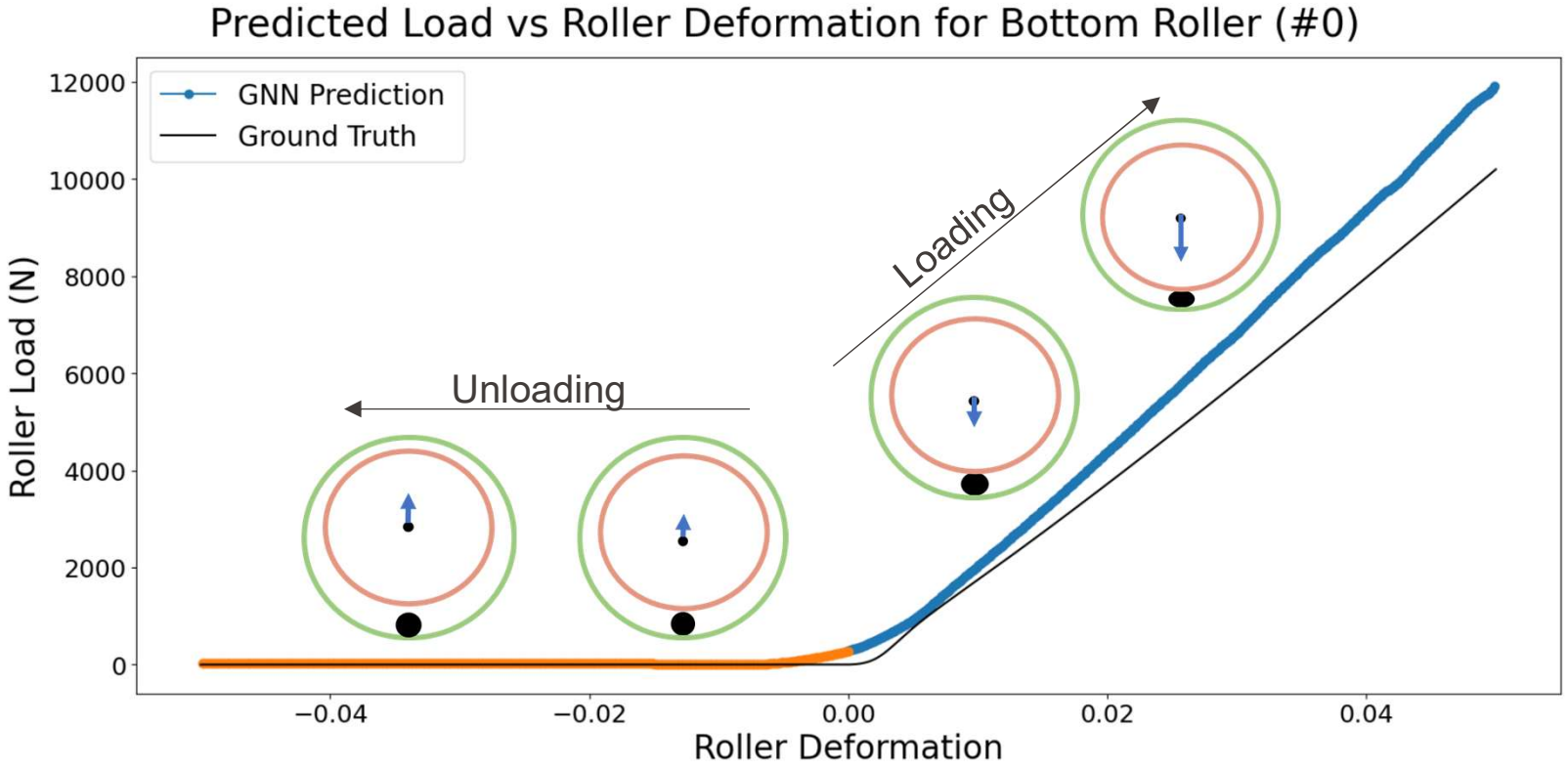





Sharma, V., Ravesloot, J., Taal, C. and Fink, O., 2023. Graph Neural Networks for Dynamic Modeling of Roller Bearing, PHM Society conference 2023

# Results: Interpretability




- **GNN can infer the unloading behavior without direct training. (Conventional Networks cannot infer this behavior)**

Sharma, V., Ravesloot, J., Taal, C. and Fink, O., 2023. Graph Neural Networks for Dynamic Modeling of Roller Bearing, PHM Society conference 2023

# Spectral Clustering

# Spectral clustering

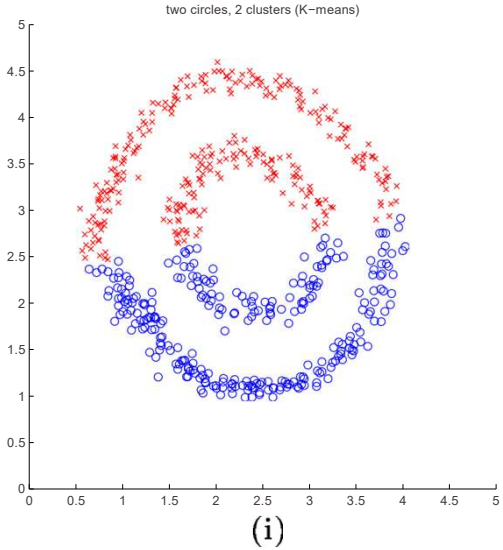
- Spectral clustering methods are attractive:
  - Easy to implement,
  - Reasonably fast especially for sparse data sets up to several thousands.

- Spectral clustering treats the data clustering as a graph partitioning problem without making any assumption on the form of the data clusters.

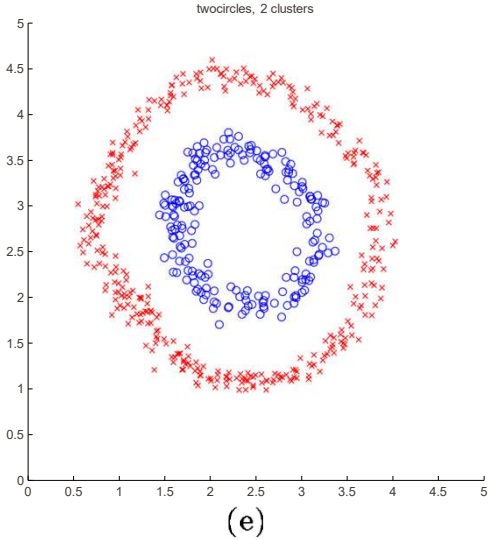
Source: Hamad & Biela

# Spectral clustering

K-means → compactness



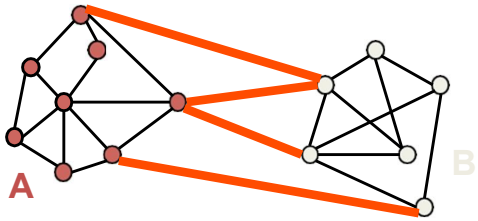

Spectral clustering → connectivity

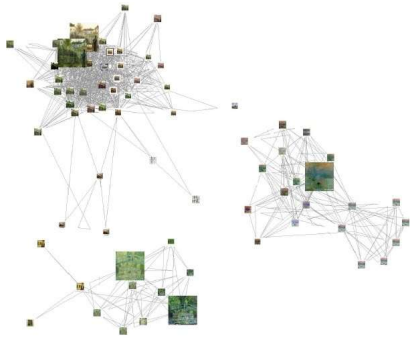



[Shi & Malik '00; Ng, Jordan, Weiss NIPS '01]

# Spectral clustering

Group points based on links in a graph



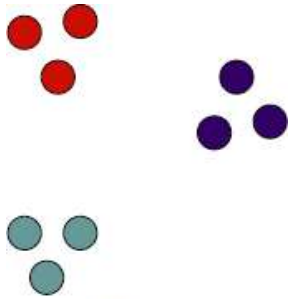
Source: James Hays

# Spectral Clustering

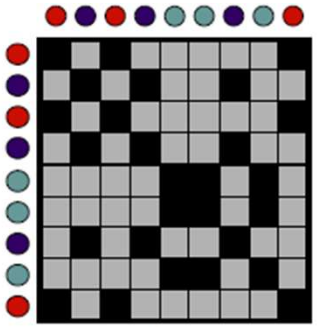
Goal: Given data points $X_1, \ldots, X_n$ and similarities $w(X_i, X_j)$, partition the data into groups so that points in a group are similar and points in different groups are dissimilar.

Similarity Graph: G(V,E,W)

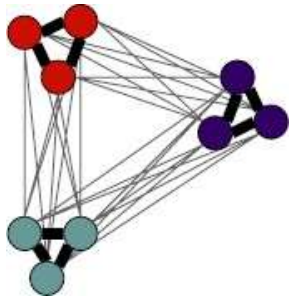
V – Vertices (Data points) E – Edge if similarity > 0
W - Edge weights (similarities)



Data



Similarities



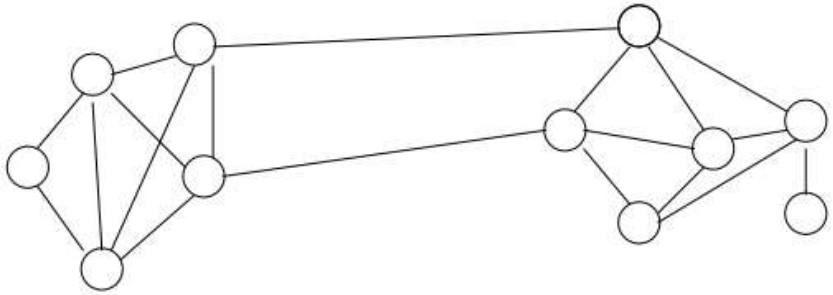
Similarity graph

Source: Hamad & Biela

Partition the graph so that edges within a group have large weights and edges across groups have small weights.

# Graph notation

- $W = (w_{ij})$ adjacency matrix of the graph
- $d_i = \sum_j w_{ij}$ degree of a vertex
- $D = diag(d_1, \ldots, d_n)$ degree matrix
- $|A|$ = number of vertices in $A$
- $\text{vol}(A) = \sum_{i \in A} d_i$



In the following: vector $f = (f_1, ..., f_n)$ interpreted as function on the graph with $f(X_i) = f_i$.

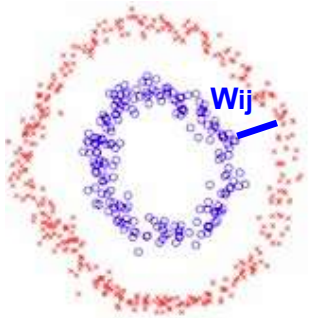
Source: Hein & Luxburg

# Similarity graph construction

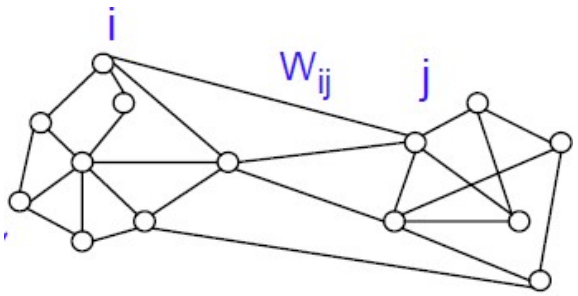
Similarity Graphs: Model local neighborhood relations between data points

E.g. Gaussian kernel similarity function

$$W_{ij} = e^{\frac{\|x_i - x_j\|^2}{2\sigma^2}}$$ → Controls size of neighborhood




Data clustering



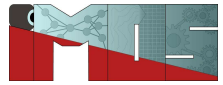

$G = \{V,E\}$

# Graph construction

- Different ways to construct a graph representing the relationships between data points :
  - Fully connected graph: All vertices having non-null similarities are connected to each other.
  - r-neighborhood graph: Each vertex is connected to vertices falling inside a ball of radius r where r is a real value that has to be tuned in order to catch the local structure of data.
  - k-nearest neighbor graph: Each vertex is connected to its k-nearest neighbors where k is an integer number which controls the local relationships of data.
  - r-neighborhood and k-nearest neighbor combined

Source: Hamad & Biela

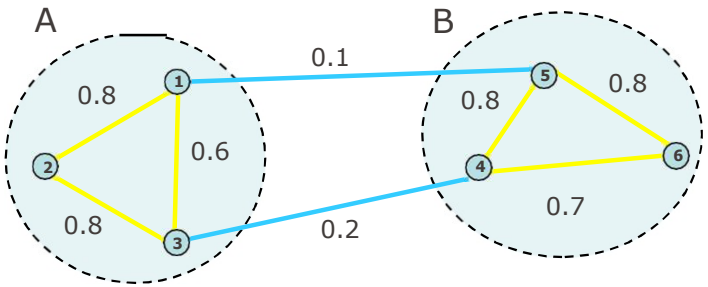
# 2-way Normalized Cuts



1. Compute the affinity matrix W, compute the degree matrix (D), D is diagonal and
   $D(i,i) = \sum_{jEV} W(i,j)$
2. Solve $(D - W)y = \lambda Dy$, where $D - W$ is called the Laplacian matrix
3. Use the eigenvector with the second smallest eigen-value to bipartition the graph into two parts.

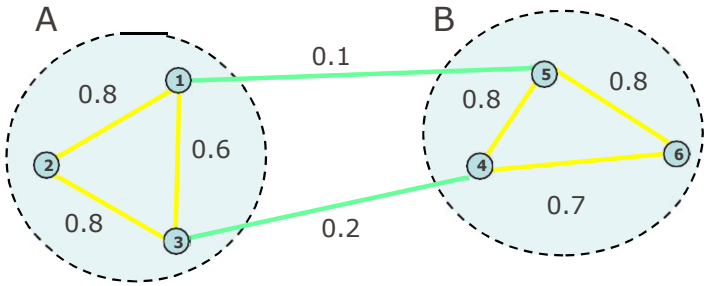
Source: D. Sonntag

# Illustrative example




Source: Hamad & Biela

# Graph and similarity matrix



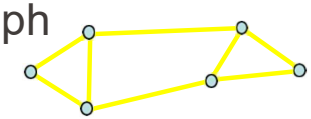

| | $x_1$ | $x_2$ | $x_3$ | $x_4$ | $x_5$ | $x_6$ |
|---|---|---|---|---|---|---|
| $x_1$ | 0 | 0.8 | 0.6 | 0 | 0.1 | 0 |
| $x_2$ | 0.8 | 0 | 0.8 | 0 | 0 | 0 |
| $x_3$ | 0.6 | 0.8 | 0 | 0.2 | 0 | 0 |
| $x_4$ | 0.8 | 0 | 0.2 | 0 | 0.8 | 0.7 |
| $x_5$ | 0.1 | 0 | 0 | 0.8 | 0 | 0.8 |
| $x_6$ | 0 | 0 | 0 | 0.7 | 0.8 | 0 |

Source: Hamad & Biela

# Illustrative example

Pre-processing

Build Laplacian matrix L of the graph



| | | | | | | |
|---|---|---|---|---|---|---|
| $x_1$ | 1.5 | -0.8 | -0.6 | 0 | -0.1 | 0 |
| $x_2$ | -0.8 | 1.6 | -0.8 | 0 | 0 | 0 |
| $x_3$ | -0.6 | -0.8 | 1.6 | -0.2 | 0 | 0 |
| $x_4$ | -0.8 | 0 | -0.2 | 2.5 | -0.8 | -0.7 |
| $x_5$ | -0.1 | 0 | 0 | 0.8 | 1.7 | -0.8 |
| $x_6$ | 0 | 0 | 0 | -0.7 | -0.8 | 1.5 |

Decomposition : Find

- eigenvalues $\Lambda$ and
- eigenvectors $X$ of matrix $L$

$\Lambda =$

| |
|---|
| 0.0 |
| 0.3 |
| 2.2 |
| 2.3 |
| 2.5 |
| 3.0 |

$X =$

| | | | | | |
|---|---|---|---|---|---|
| 0.4 | 0.2 | 0.1 | 0.4 | -0.2 | -0.9 |
| 0.4 | 0.2 | 0.1 | -0. | 0.4 | 0.3 |
| 0.4 | 0.2 | -0.2 | 0.0 | -0.2 | 0.6 |
| 0.4 | -0.4 | 0.9 | 0.2 | -0.4 | -0.6 |
| 0.4 | -0.7 | -0.4 | -0.8 | -0.6 | -0.2 |
| 0.4 | -0.7 | -0.2 | 0.5 | 0.8 | 0.9 |

- Map vertices to the corresponding components of 2nd eigenvector

| | |
|---|---|
| $x_1$ | 0.2 |
| $x_2$ | 0.2 |
| $x_3$ | 0.2 |
| $x_4$ | -0.4 |
| $x_5$ | -0.7 |
| $x_6$ | -0.7 |

How do we find the clusters?

Source: Hamad & Biela

# Spectral Clustering Algorithms

| | |
|---|---|
| $x_1$ | 0.2 |
| $x_2$ | 0.2 |
| $x_3$ | 0.2 |
| $x_4$ | -0.4 |
| $x_5$ | -0.7 |
| $x_6$ | -0.7 |

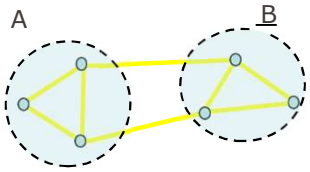
Split at value 0

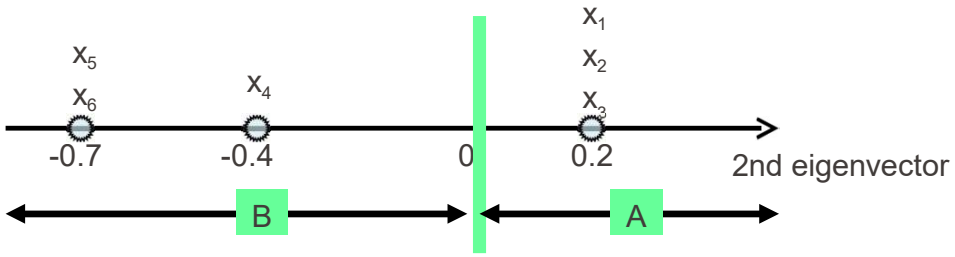
Cluster $A$: Positive points

Cluster $B$: Negative points

| | |
|---|---|
| $x_1$ | 0.2 |
| $x_2$ | 0.2 |
| $x_3$ | 0.2 |

| | |
|---|---|
| $x_4$ | -0.4 |
| $x_5$ | -0.7 |
| $x_6$ | -0.7 |



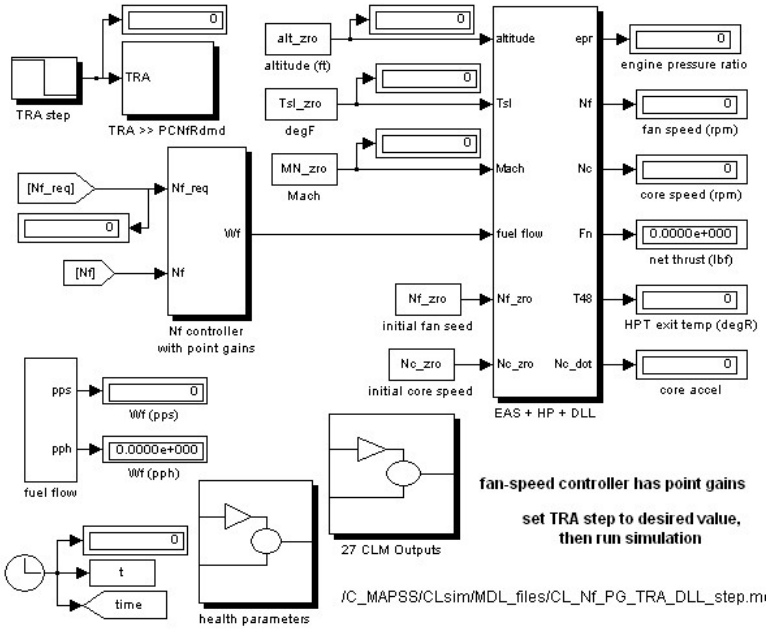




Source: Hamad & Biela

# Benefits of proposed approaches

- Fault generation based on domain expertise
- Bridge the sim2real gap with domain adaptation
- Overcome the imbalance problem
- Proposed approach independent of the imbalance level

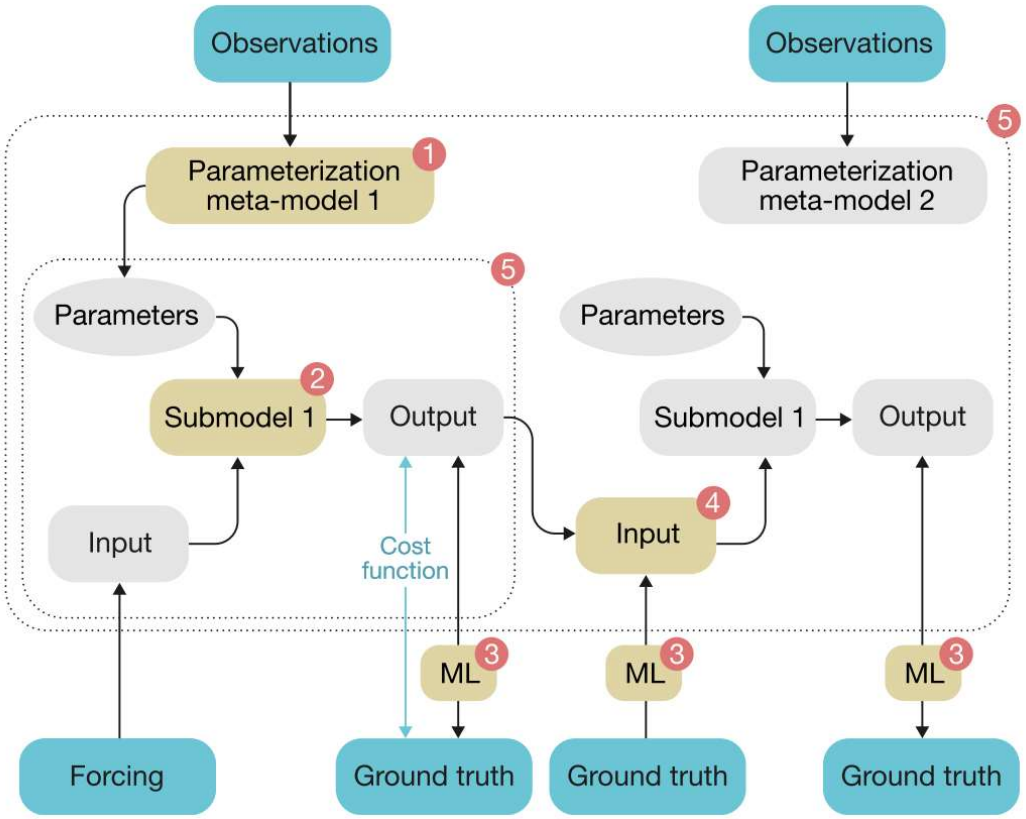
# Hybrid approaches: the best of two worlds




Deep Learning Models

Physics-based high fidelity models

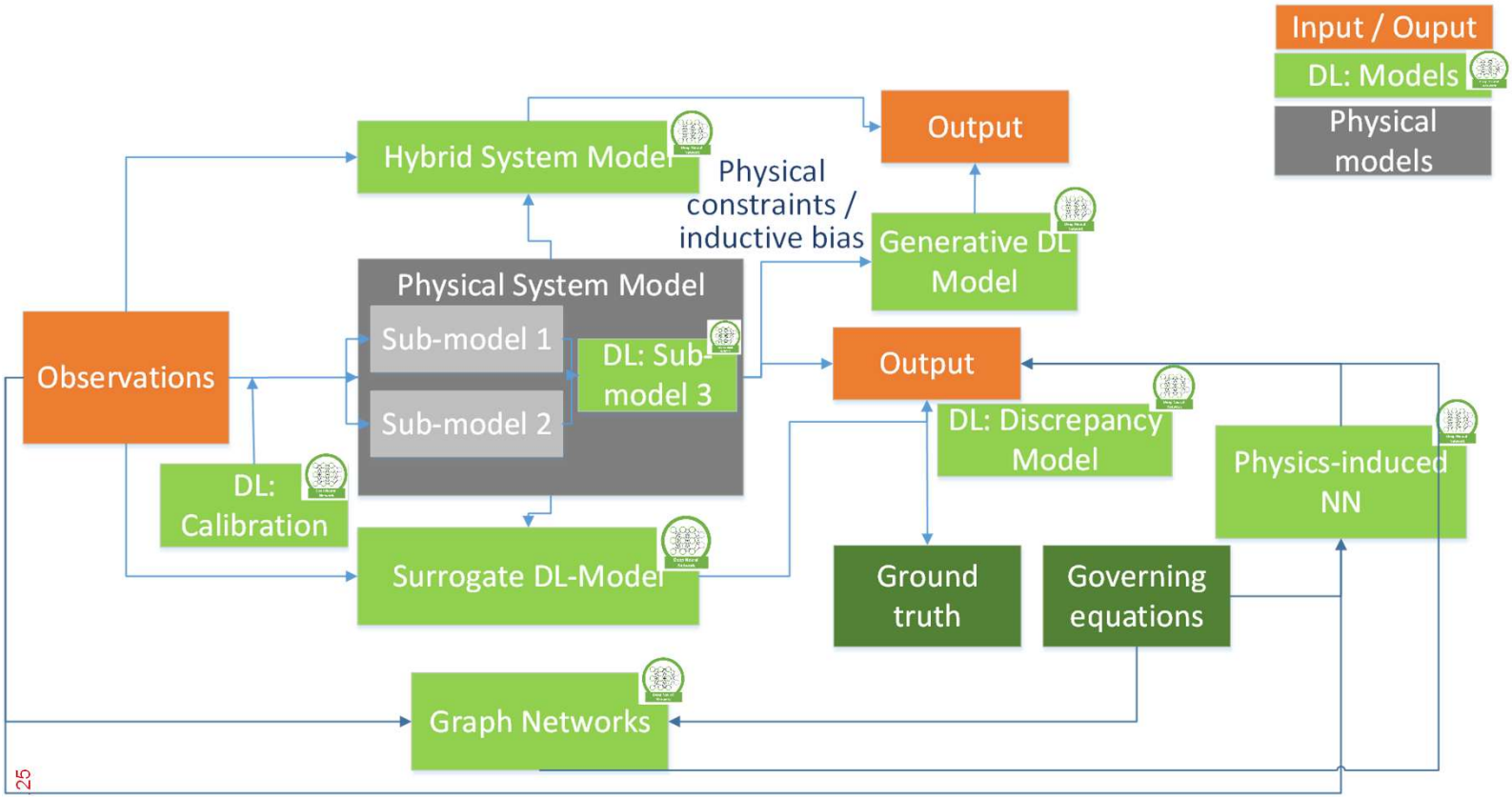
# Hybrid models at different stages



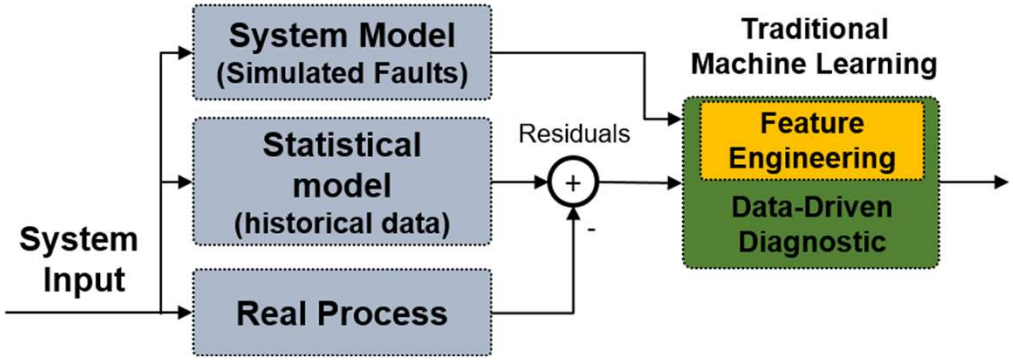

1. Improving parametrizations
2. Replacing a 'physical' sub-model with a machine learning model
3. Analysis of model–observation mismatch
4. Constraining submodels
5. Surrogate modelling or emulation

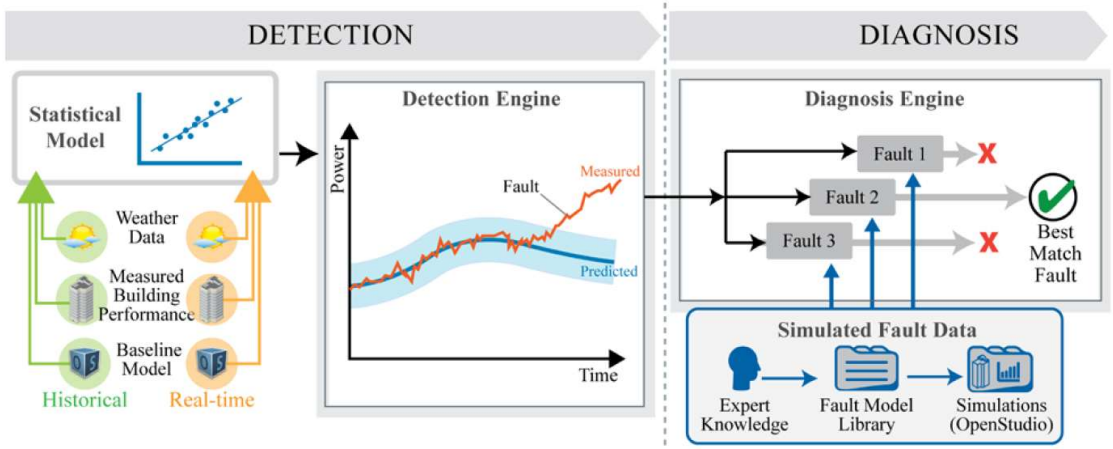
M. Reichstein *et al.*, "Deep learning and process understanding for data-driven Earth system science," *Nature*, vol. 566, no. 7743, pp. 195–204, 2019.

# Different ways of fusing DL and physics



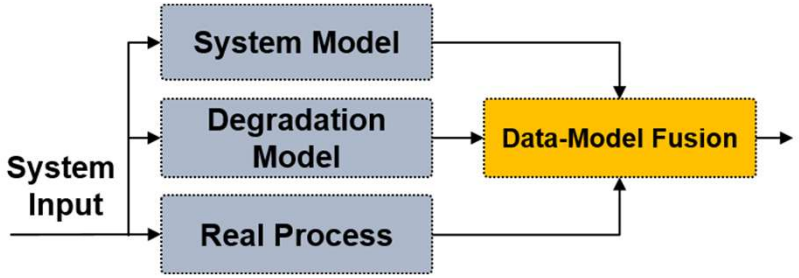
# Different types of hybrid models

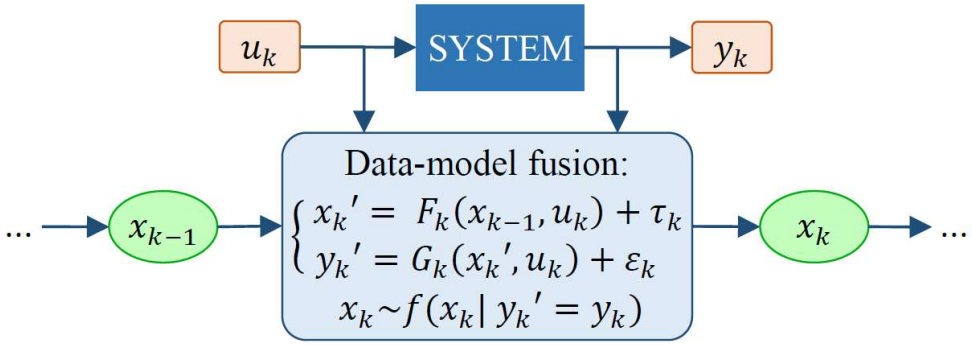



Frank, Stephen, et al. *Hybrid model-based and data-driven fault detection and diagnostics for commercial buildings*. No. NREL/CP-5500-65924. National Renewable Energy Lab.(NREL), Golden, CO (United States), 2016.

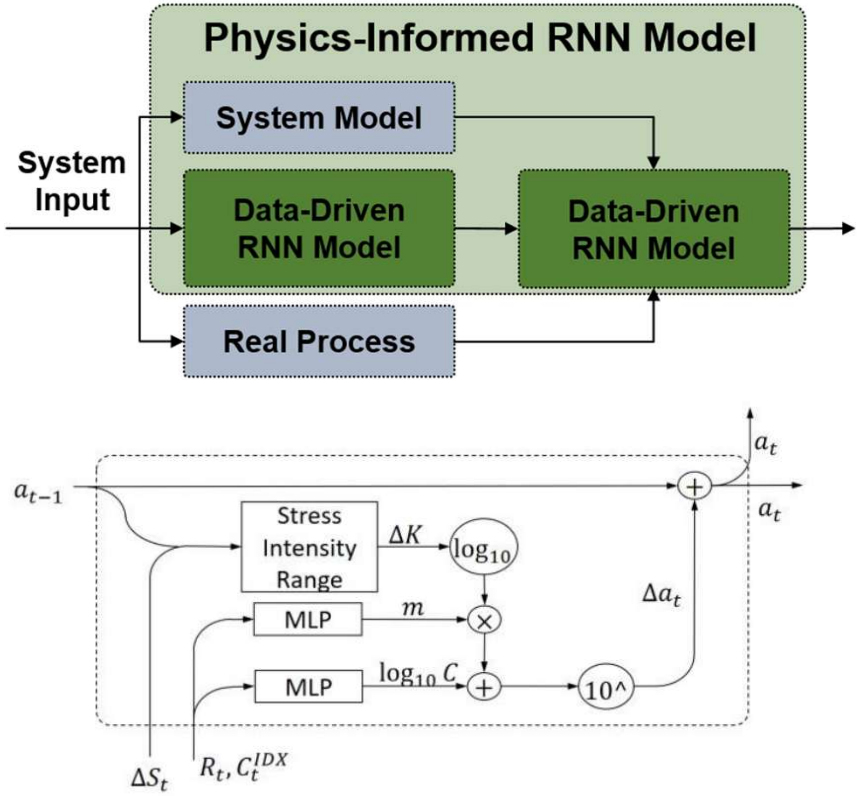
# Different types of hybrid models



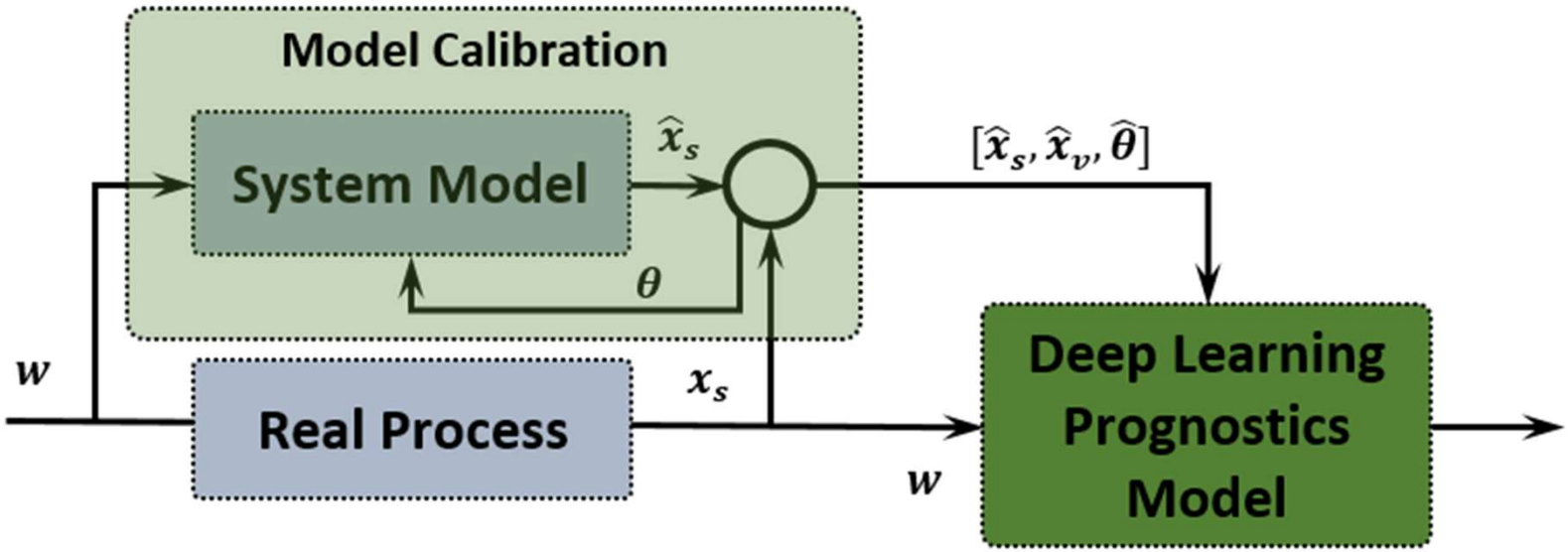

Hanachi, H., et al. "Hybrid physics-based and data-driven phm." *Canadian Machinery Vibration Association (CMVA) Annual Conference, Edmonton, Alberta, Canada*. 2017.

# Different types of hybrid models



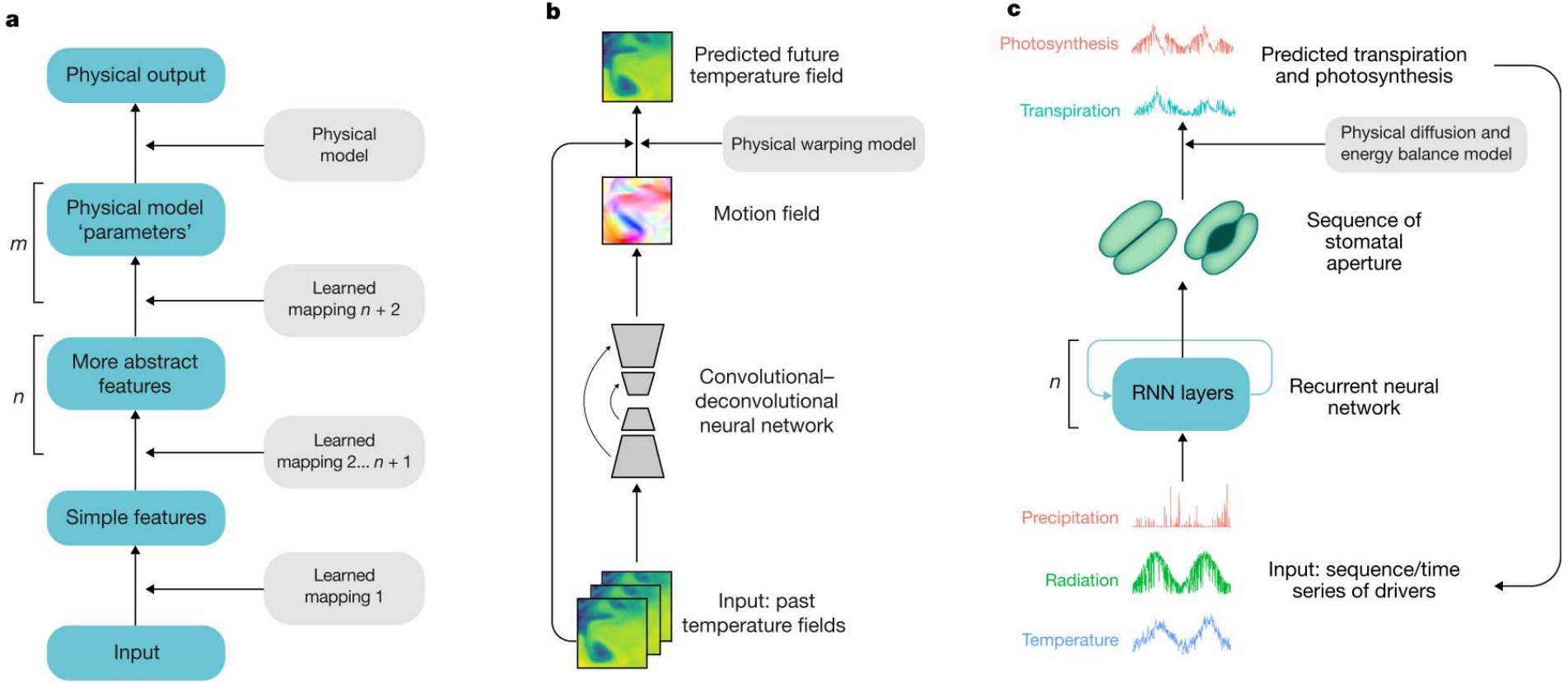

Dourado, Arinan, and Felipe AC Viana. "Physics-Informed Neural Networks for Corrosion-Fatigue Prognosis." *Proceedings of the Annual Conference of the PHM Society*. Vol. 11. No. 1. 2019.

# Fusing physical performance models and deep learning




Arias Chao, M., Kulkarni, C., Goebel, K., & Fink, O. (2020), Fusing Physics-based and Deep Learning Models for Prognostics

# Interpretation of hybrid modelling as deepening a deep learning architecture by adding one or several physical layers after the MLP




M. Reichstein *et al.*, "Deep learning and process understanding for data-driven Earth system science," *Nature*, vol. 566, no. 7743, pp. 195–204, 2019.