Inverse Problems, Machine Learning and Digital Twins

Theory & Examples

Mark Asch — Université de Picardie Jules Verne

Contents

Digital Twin

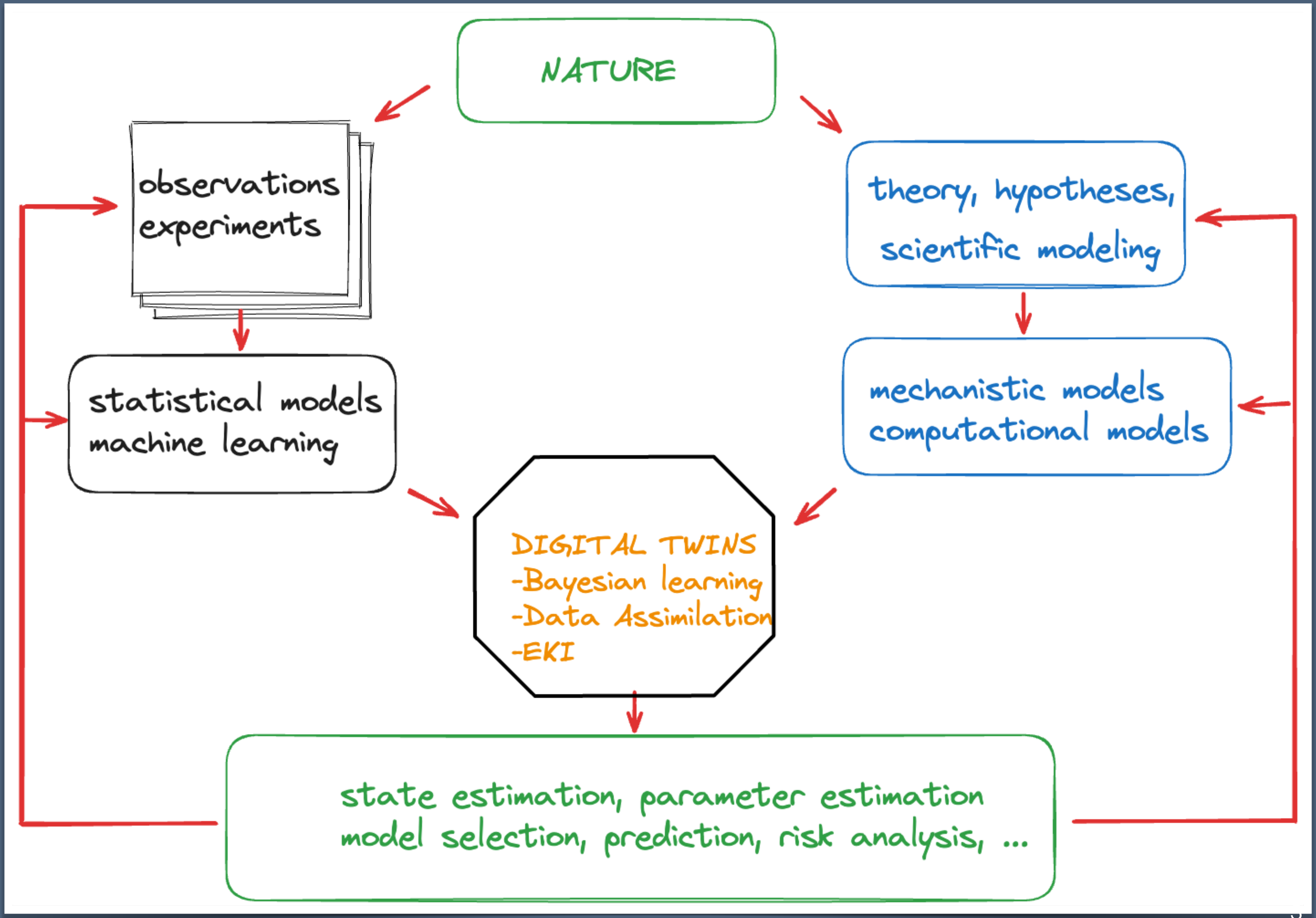
A mathematical
definition

- "A digital twin couples computational models with a physical counterpart to create a system that is dynamically updated through bidirectional data flows as conditions change." [NASEM]

Digital Twin

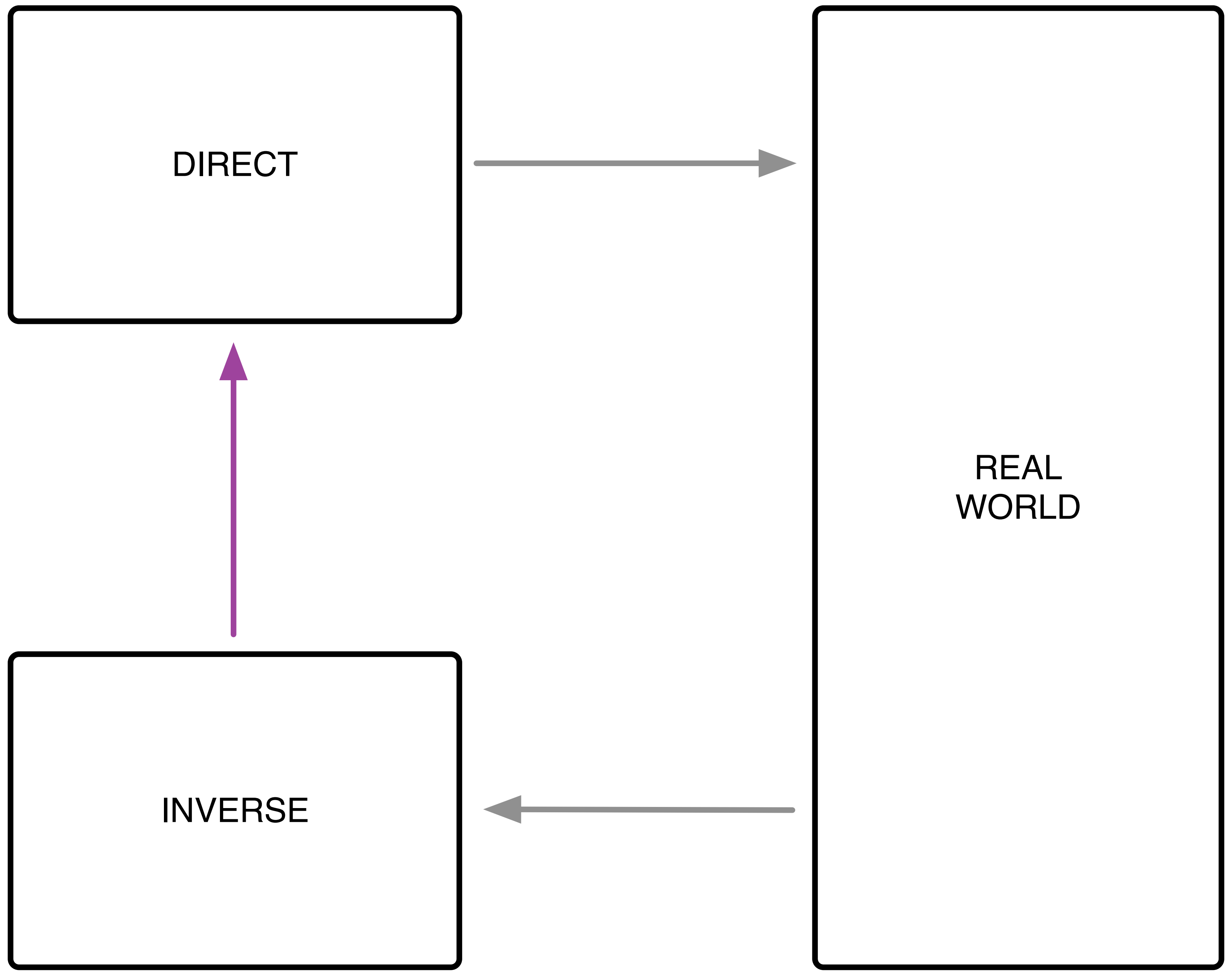
A mathematical definition

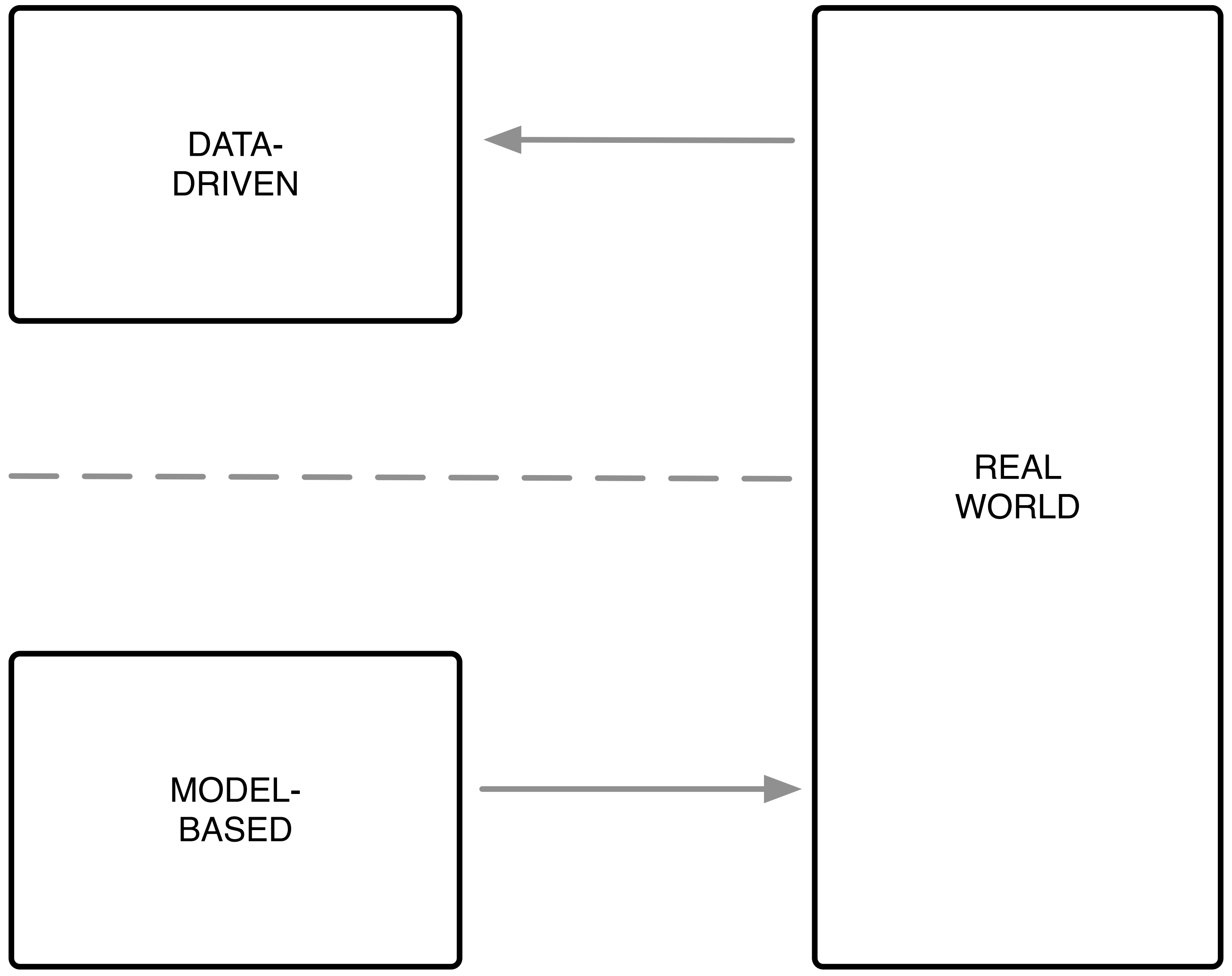
- "A digital twin couples computational models with a physical counterpart to create a system that is dynamically updated through bidirectional data flows as conditions change." [NASEM]



Digital Twin: definition

Definition: A digital twin can be defined as a computational model (or a set of models) that evolves over time to represent the structure and behaviour of a corresponding physical asset, by exchanging bi-directional data with it.



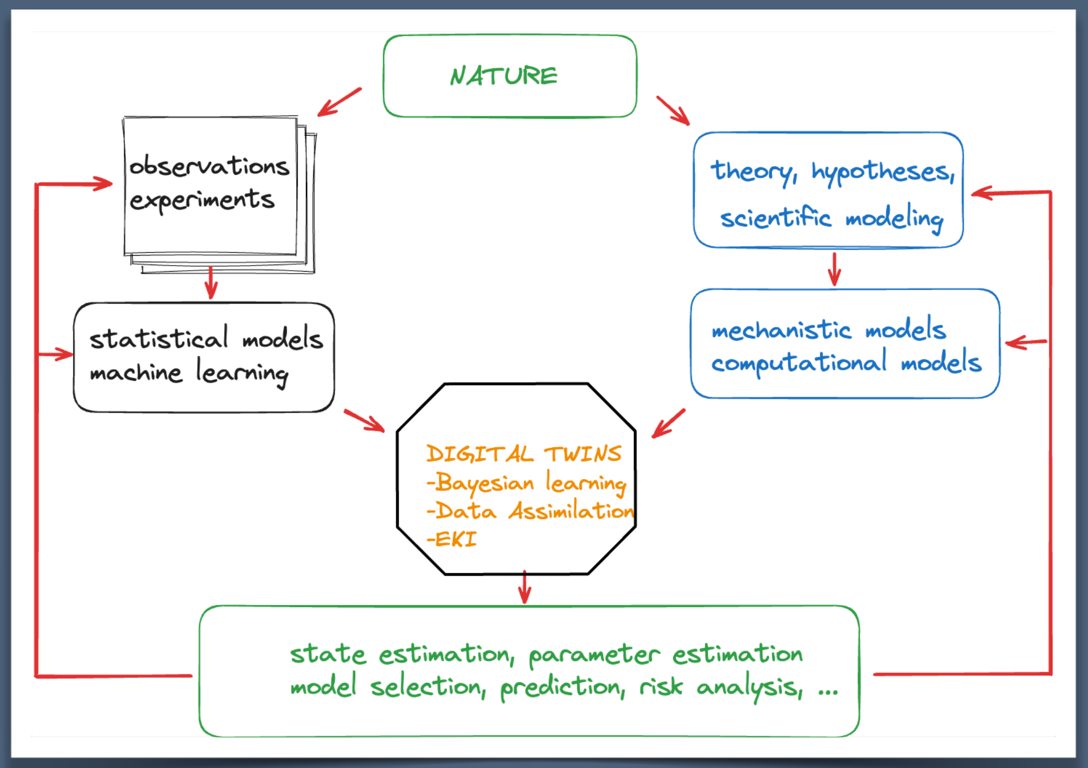


Challenge: For a given context, what is the best combination/synergy between data-driven/ML and model-based approaches?

SciML

Mathematics

Inverse Problems for DTs



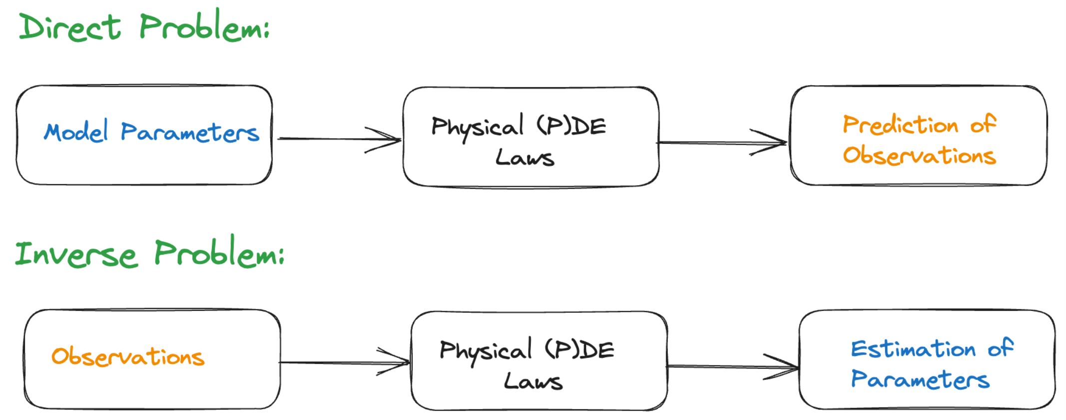
Dynamical System

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = g(t, \mathbf{u}; \boldsymbol{\theta}), \qquad \mathbf{u}(t_0) = \mathbf{u}_0,$$

with g known, $\boldsymbol{\theta} \in \Theta$, $\mathbf{u}(t) \in \mathbb{R}^k$.

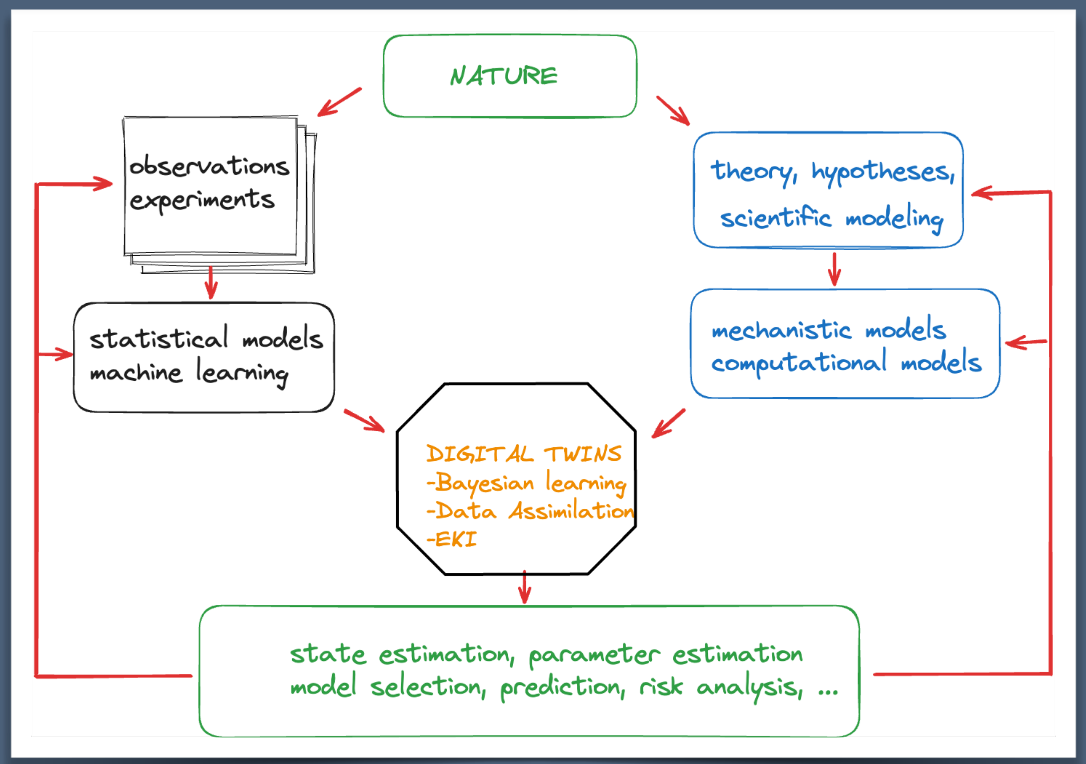
Direct: Given $\boldsymbol{\theta}$, $\mathbf{u}_0$, find $\mathbf{u}(t)$ for $t \geq t_0$.

Inverse: Given $\mathbf{u}(t)$ for $t \geq t_0$, find $\boldsymbol{\theta} \in \Theta$.



Mathematics

Inverse Problems for DTs



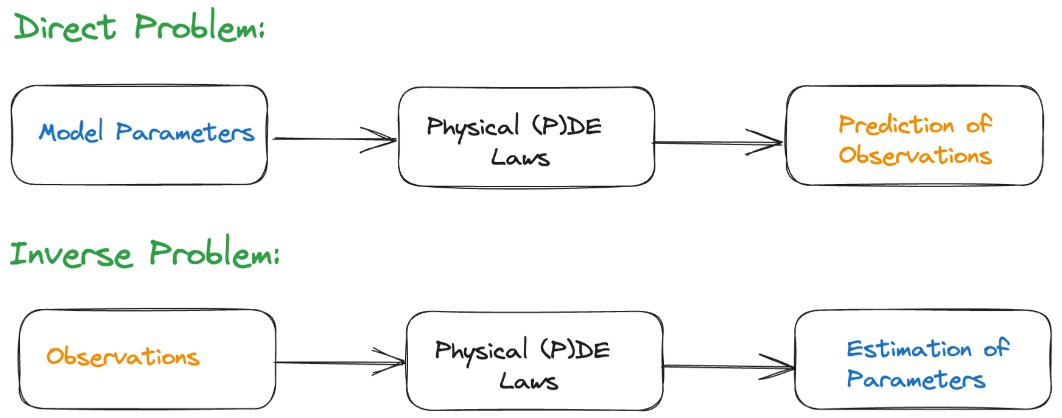
Dynamical System

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = g(t, \mathbf{u}; \boldsymbol{\theta}), \qquad \mathbf{u}(t_0) = \mathbf{u}_0,$$

with g known, $\boldsymbol{\theta} \in \Theta$, $\mathbf{u}(t) \in \mathbb{R}^k$.

Direct: Given $\boldsymbol{\theta}$, $\mathbf{u}_0$, find $\mathbf{u}(t)$ for $t \geq t_0$.

Inverse: Given $\mathbf{u}(t)$ for $t \geq t_0$, find $\boldsymbol{\theta} \in \Theta$.



Inverse Problems - Deterministic Case

In the deterministic case ($\eta = 0$), because of the ill-posedness of the inverse problem, we replace it by the least-squares optimization problem,

$$\underset{u \in X}{\operatorname{argmin}} \frac{1}{2} \|y - \mathcal{G}(u)\|_Y^2$$

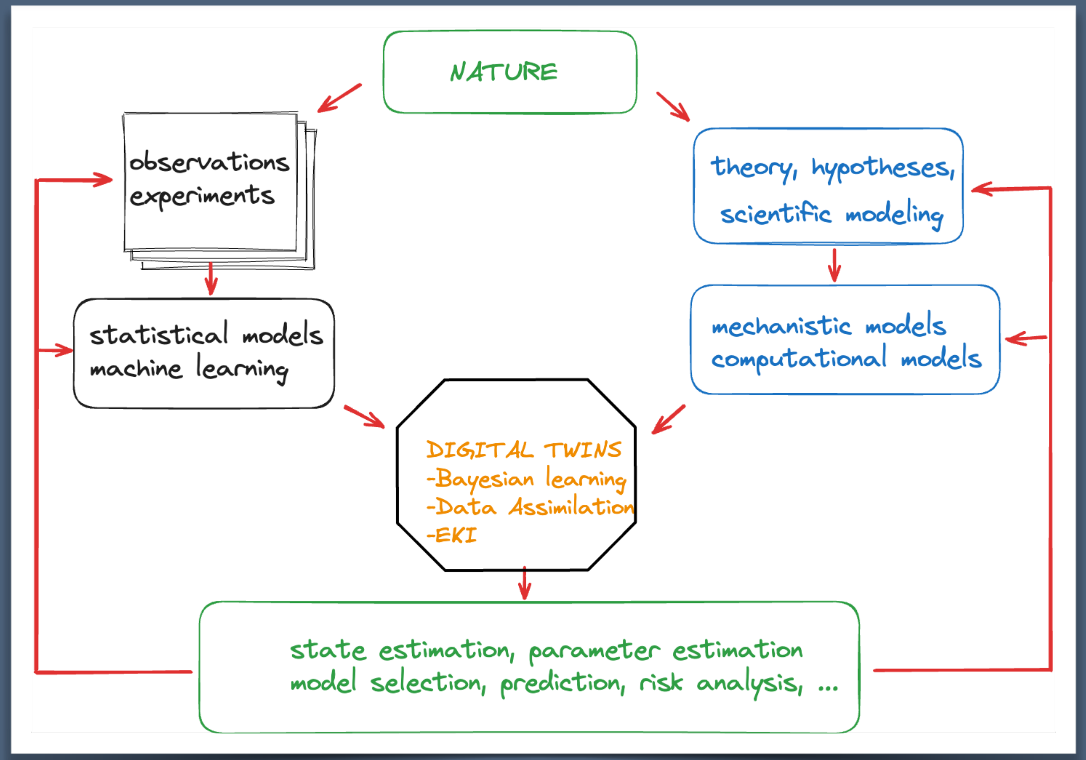
that is usually regularized as

$$\underset{u \in E}{\operatorname{argmin}} \frac{1}{2} \left(\|y - \mathcal{G}(u)\|_Y^2 + \frac{1}{2} \|u - m_0\|_E^2 \right)$$

for a given reference point $m_0 \in E$, with E, X, Y Banach spaces. The optimization requires a gradient (or adjoint).

Mathematics

Inverse Problems for DTs



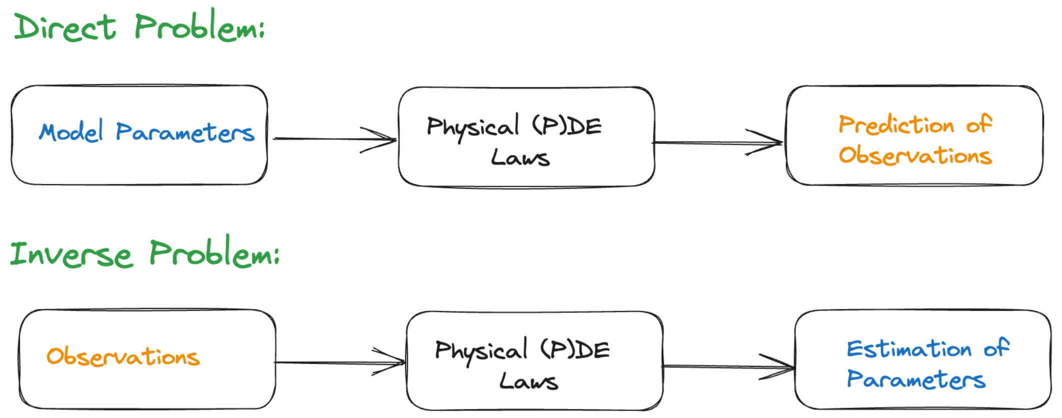
Dynamical System

$$\frac{d\mathbf{u}}{dt} = g(t, \mathbf{u}; \boldsymbol{\theta}), \qquad \mathbf{u}(t_0) = \mathbf{u}_0,$$

with g known, $\boldsymbol{\theta} \in \Theta$, $\mathbf{u}(t) \in \mathbb{R}^k$.

Direct: Given $\boldsymbol{\theta}$, $\mathbf{u}_0$, find $\mathbf{u}(t)$ for $t \geq t_0$.

Inverse: Given $\mathbf{u}(t)$ for $t \geq t_0$, find $\boldsymbol{\theta} \in \Theta$.



Inverse Problems - Deterministic Case

In the deterministic case ($\eta = 0$), because of the ill-posedness of the inverse problem, we replace it by the least-squares optimization problem,

$$\underset{u \in X}{\operatorname{argmin}} \frac{1}{2} \|y - \mathcal{G}(u)\|_Y^2$$

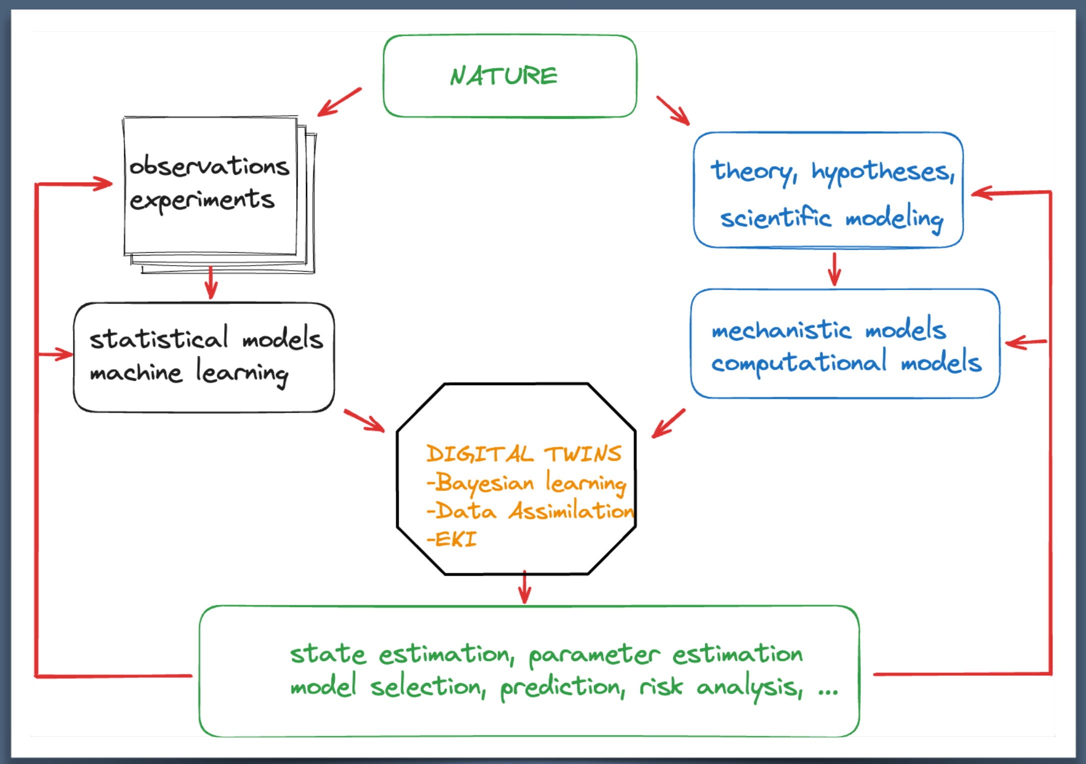
that is usually regularized as

$$\underset{u \in E}{\operatorname{argmin}} \frac{1}{2} \left(\|y - \mathcal{G}(u)\|_Y^2 + \frac{1}{2} \|u - m_0\|_E^2 \right)$$

for a given reference point $m_0 \in E$, with E, X, Y Banach spaces. The optimization requires a gradient (or adjoint).

Mathematics

Inverse Problems for DTs



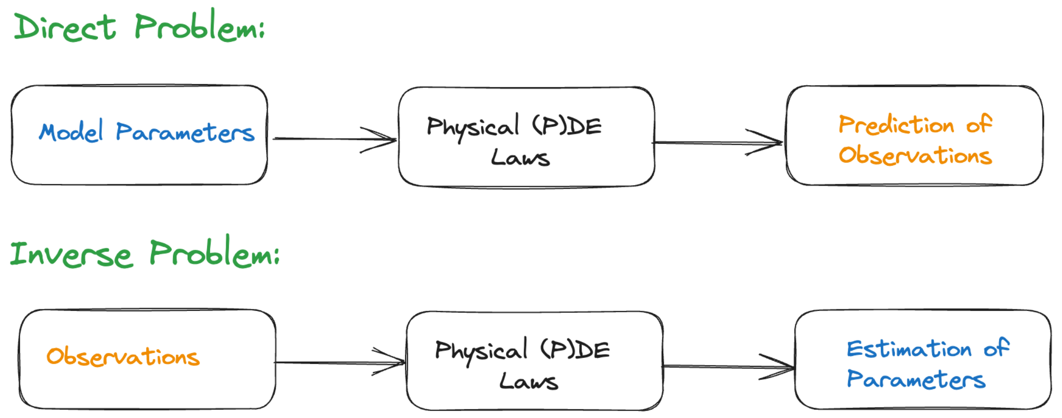
Dynamical System

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = g(t, \mathbf{u}; \boldsymbol{\theta}), \qquad \mathbf{u}(t_0) = \mathbf{u}_0,$$

with g known, $\boldsymbol{\theta} \in \Theta$, $\mathbf{u}(t) \in \mathbb{R}^k$.

Direct: Given $\boldsymbol{\theta}$, $\mathbf{u}_0$, find $\mathbf{u}(t)$ for $t \geq t_0$.

Inverse: Given $\mathbf{u}(t)$ for $t \geq t_0$, find $\boldsymbol{\theta} \in \Theta$.



Inverse Problems - Deterministic Case

In the deterministic case ($\eta = 0$), because of the ill-posedness of the inverse problem, we replace it by the least-squares optimization problem,

$$\operatorname*{argmin}_{u \in X} \frac{1}{2} \|y - \mathcal{G}(u)\|_Y^2$$

that is usually regularized as

$$\operatorname*{argmin}_{u \in E} \frac{1}{2} \left(\|y - \mathcal{G}(u)\|_Y^2 + \frac{1}{2} \|u - m_0\|_E^2 \right)$$

for a given reference point $m_0 \in E$, with E, X, Y Banach spaces. The optimization requires a gradient (or adjoint).

In reality, we have uncertainty (noise)

- in the model,
- in the parameters,
- in the observations.

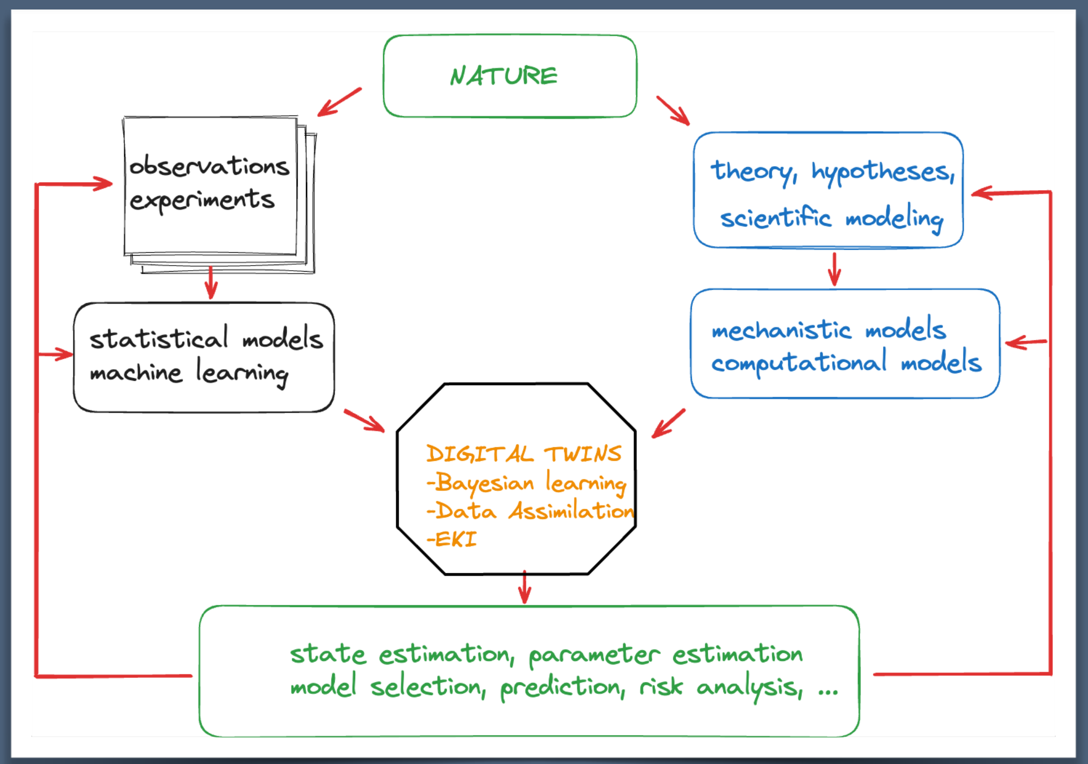
The dynamical system becomes

$$\mathbf{y} = \mathcal{G}(\mathbf{u}) + \eta,$$

where $\eta \sim \mathcal{N}(0, \Sigma)$.

Mathematics

Inverse Problems for DTs



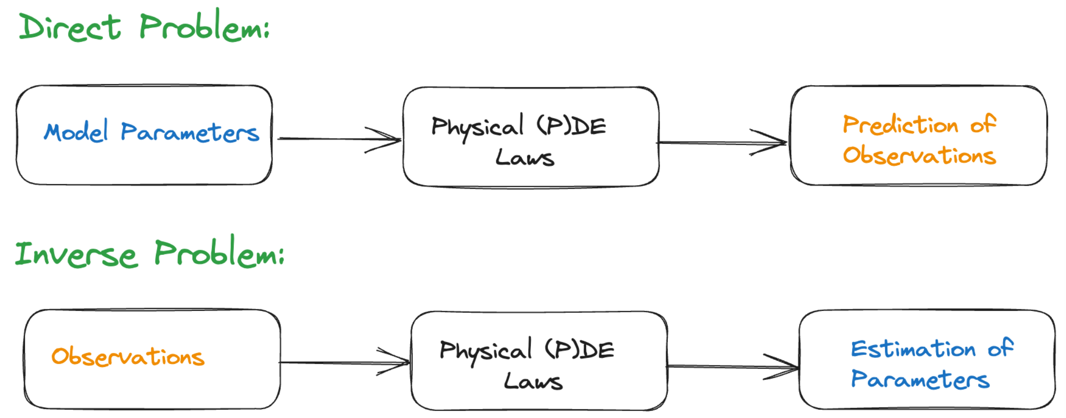
Dynamical System

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = g(t, \mathbf{u}; \boldsymbol{\theta}), \qquad \mathbf{u}(t_0) = \mathbf{u}_0,$$

with g known, $\boldsymbol{\theta} \in \Theta$, $\mathbf{u}(t) \in \mathbb{R}^k$.

Direct: Given $\boldsymbol{\theta}$, $\mathbf{u}_0$, find $\mathbf{u}(t)$ for $t \geq t_0$.

Inverse: Given $\mathbf{u}(t)$ for $t \geq t_0$, find $\boldsymbol{\theta} \in \Theta$.



In reality, we have uncertainty (noise)

- in the model,
- in the parameters,
- in the observations.

The dynamical system becomes

$$\mathbf{y} = \mathcal{G}(\mathbf{u}) + \eta,$$

where $\eta \sim \mathcal{N}(0, \Sigma)$.

Inverse Problems - Deterministic Case

In the deterministic case ($\eta = 0$), because of the ill-posedness of the inverse problem, we replace it by the least-squares optimization problem,

$$\operatorname*{argmin}_{u \in X} \frac{1}{2} \|y - \mathcal{G}(u)\|_Y^2$$

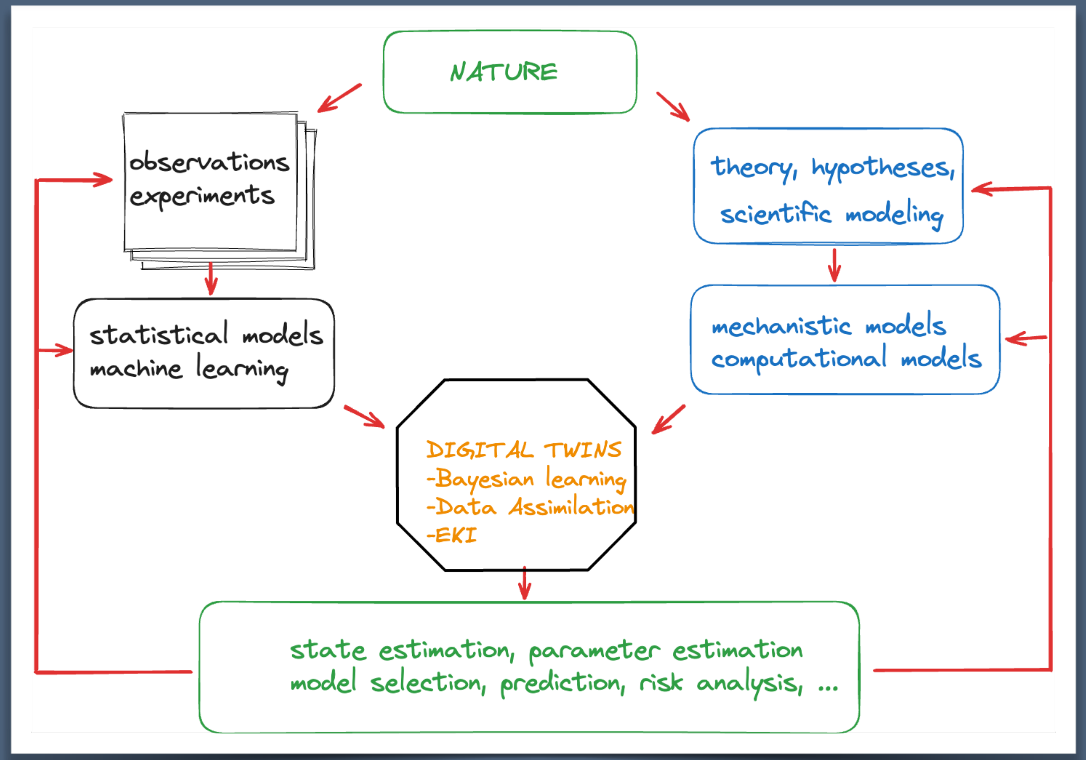
that is usually regularized as

$$\operatorname*{argmin}_{u \in E} \frac{1}{2} \left(\|y - \mathcal{G}(u)\|_Y^2 + \frac{1}{2} \|u - m_0\|_E^2 \right)$$

for a given reference point $m_0 \in E$, with E, X, Y Banach spaces. The optimization requires a gradient (or adjoint).

Mathematics

Inverse Problems for DTs



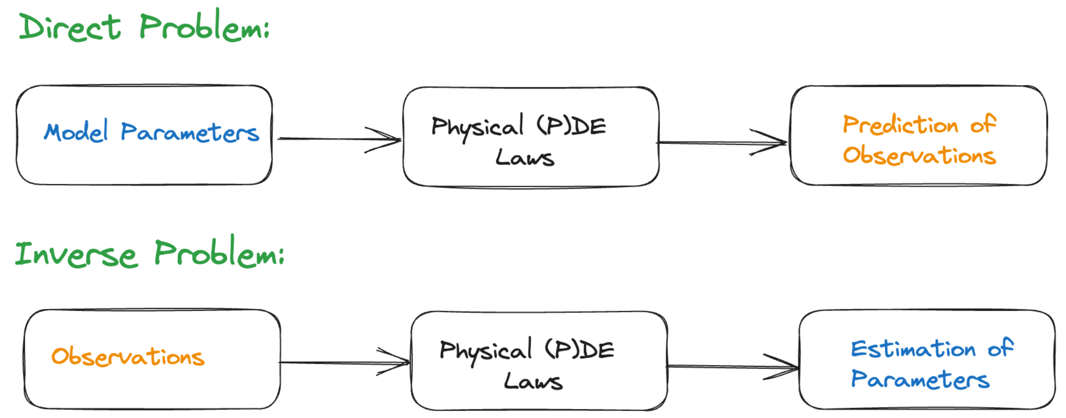
Dynamical System

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = g(t, \mathbf{u}; \boldsymbol{\theta}), \qquad \mathbf{u}(t_0) = \mathbf{u}_0,$$

with g known, $\boldsymbol{\theta} \in \Theta$, $\mathbf{u}(t) \in \mathbb{R}^k$.

Direct: Given $\boldsymbol{\theta}$, $\mathbf{u}_0$, find $\mathbf{u}(t)$ for $t \geq t_0$.

Inverse: Given $\mathbf{u}(t)$ for $t \geq t_0$, find $\boldsymbol{\theta} \in \Theta$.



Inverse Problems - Deterministic Case

In the deterministic case ($\eta = 0$), because of the ill-posedness of the inverse problem, we replace it by the least-squares optimization problem,

$$\operatorname*{argmin}_{u \in X} \frac{1}{2} \|y - \mathcal{G}(u)\|_Y^2$$

that is usually regularized as

$$\operatorname*{argmin}_{u \in E} \frac{1}{2} \left(\|y - \mathcal{G}(u)\|_Y^2 + \frac{1}{2} \|u - m_0\|_E^2 \right)$$

for a given reference point $m_0 \in E$, with E, X, Y Banach spaces. The optimization requires a gradient (or adjoint).

In reality, we have uncertainty (noise)

- in the model,
- in the parameters,
- in the observations.

The dynamical system becomes

$$\mathbf{y} = \mathcal{G}(\mathbf{u}) + \eta,$$

where $\eta \sim \mathcal{N}(0, \Sigma)$.

Inverse Problems - Stochastic Case

In the stochastic case, the solution of the inverse problem, "find u from y," is a posterior probability density function (ppdf).

Theorem (Bayes)

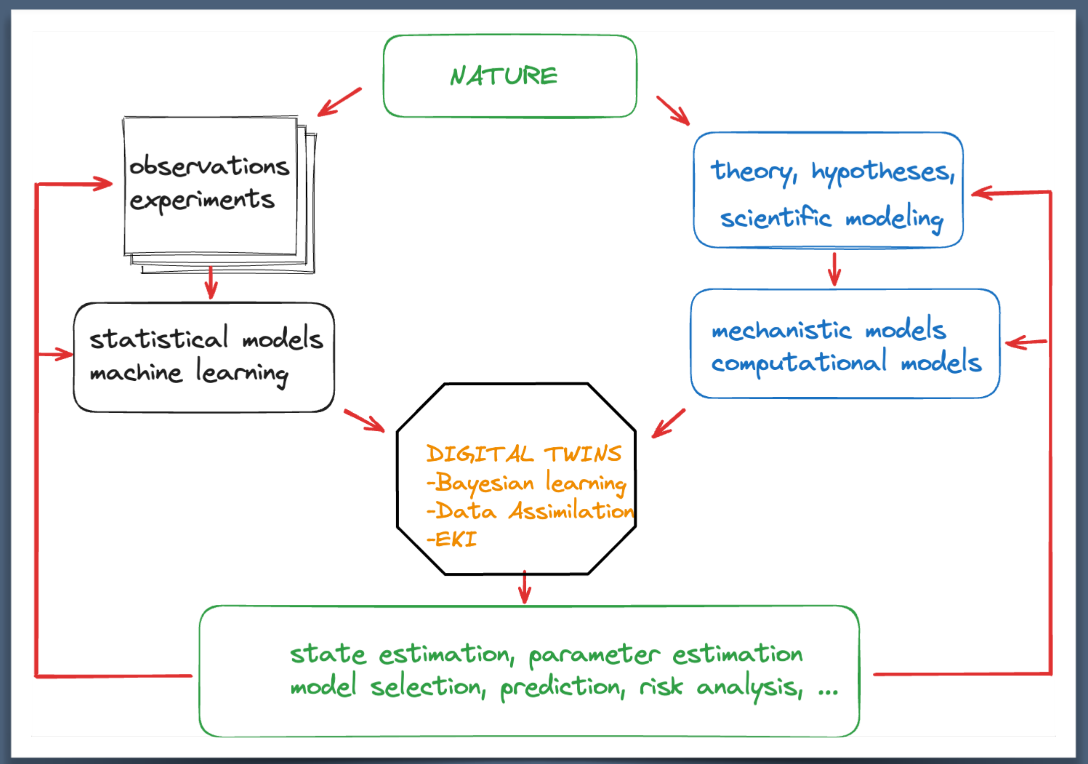
$$p(u|y) = \frac{p(y|u)p(u)}{p(y)},$$

or

$$p(\text{parameter}|\text{data}) \propto p(\text{data}|\text{parameter})p(\text{parameter}).$$

Mathematics

Inverse Problems for DTs



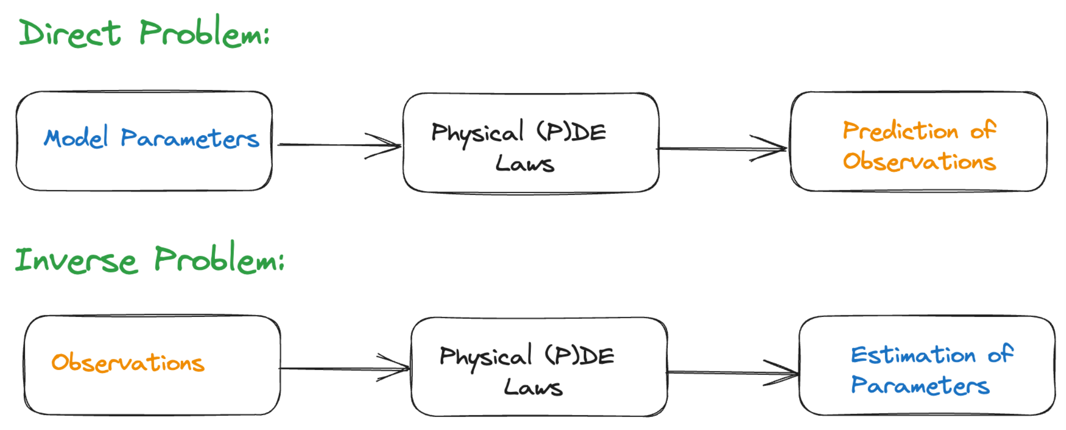
Dynamical System

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} = g(t, \mathbf{u}; \boldsymbol{\theta}), \qquad \mathbf{u}(t_0) = \mathbf{u}_0,$$

with g known, $\boldsymbol{\theta} \in \Theta$, $\mathbf{u}(t) \in \mathbb{R}^k$.

Direct: Given $\boldsymbol{\theta}$, $\mathbf{u}_0$, find $\mathbf{u}(t)$ for $t \geq t_0$.

Inverse: Given $\mathbf{u}(t)$ for $t \geq t_0$, find $\boldsymbol{\theta} \in \Theta$.



In reality, we have uncertainty (noise)

- in the model,
- in the parameters,
- in the observations.

The dynamical system becomes

$$\mathbf{y} = \mathcal{G}(\mathbf{u}) + \eta,$$

where $\eta \sim \mathcal{N}(0, \Sigma)$.

Inverse Problems - Deterministic Case

In the deterministic case ($\eta = 0$), because of the ill-posedness of the inverse problem, we replace it by the least-squares optimization problem,

$$\underset{u \in X}{\operatorname{argmin}} \frac{1}{2} \|y - \mathcal{G}(u)\|_Y^2$$

that is usually regularized as

$$\underset{u \in E}{\operatorname{argmin}} \frac{1}{2} \left(\|y - \mathcal{G}(u)\|_Y^2 + \frac{1}{2} \|u - m_0\|_E^2 \right)$$

for a given reference point $m_0 \in E$, with E, X, Y Banach spaces. The optimization requires a gradient (or adjoint).

Inverse Problems - Stochastic Case

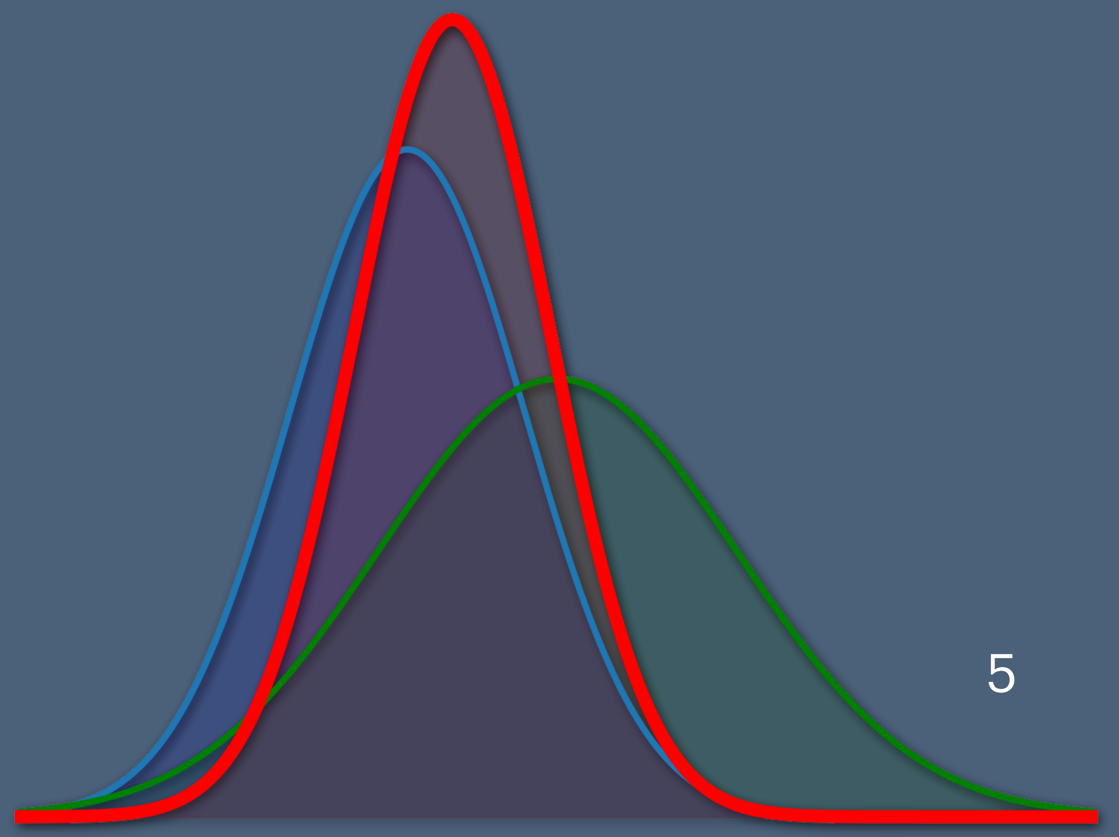
In the stochastic case, the solution of the inverse problem, “find u from y,” is a posterior probability density function (ppdf).

Theorem (Bayes)

$$p(u|y) = \frac{p(y|u)p(u)}{p(y)},$$

or

$$p(\text{parameter}|\text{data}) \propto p(\text{data}|\text{parameter})p(\text{parameter}).$$



ML/AI in a Nutshell

Finding a pattern.

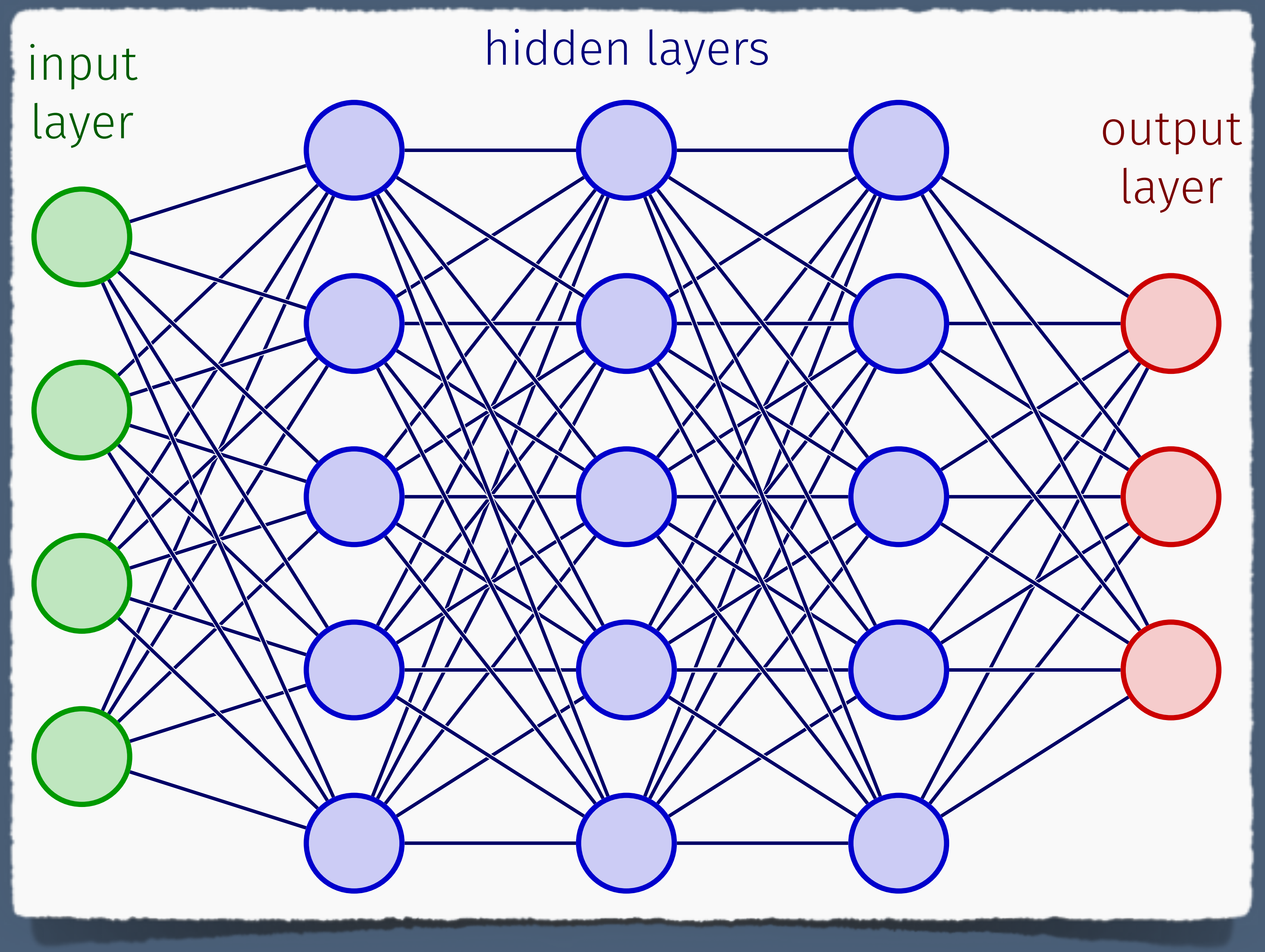
$$f : x \to y$$

- Input
- Features
- Parameters
- Causes

- Output
- Observations
- Outcomes
- Effects

ML/AI in a Nutshell

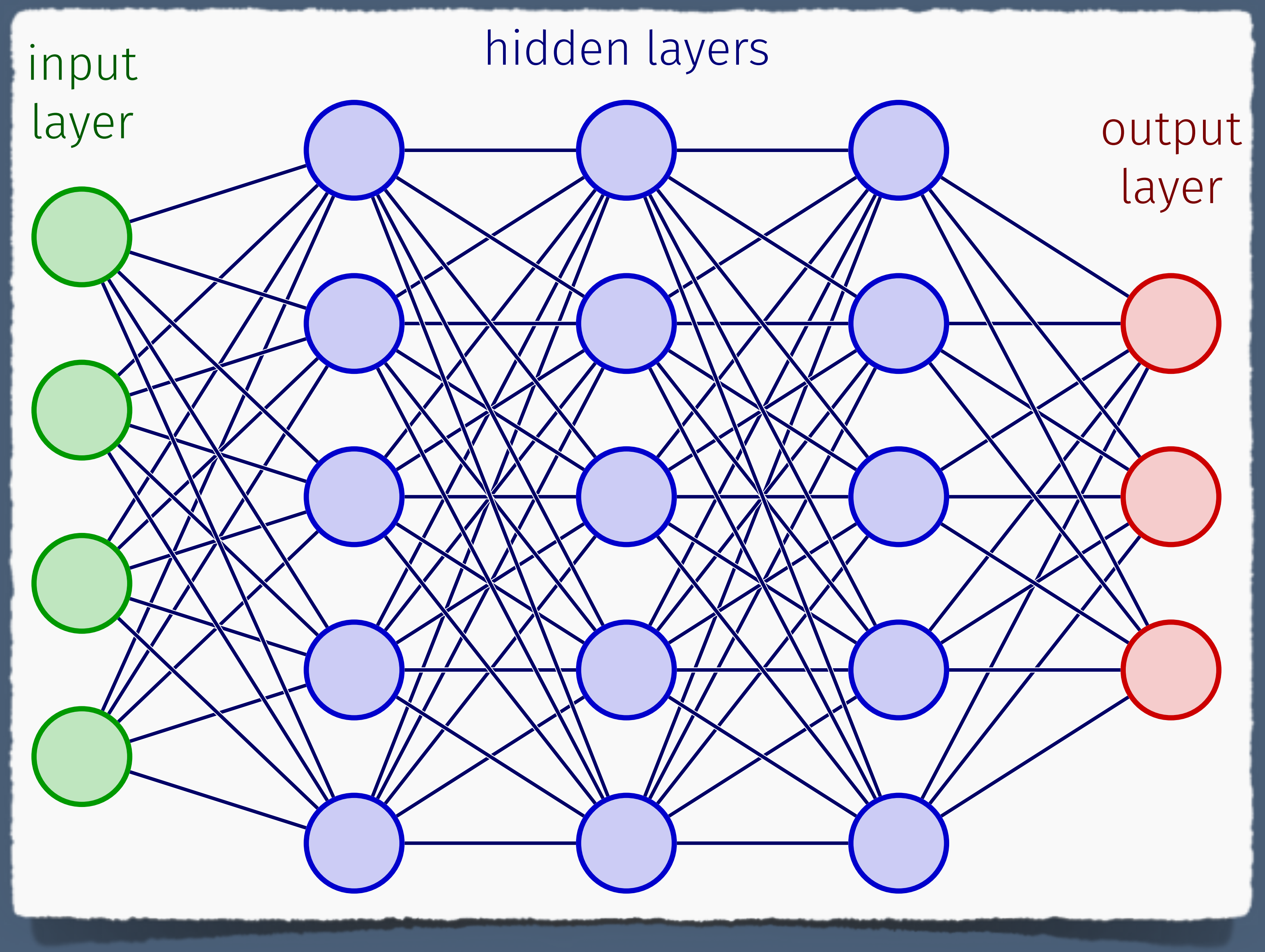
How is this done?



ML/AI in a Nutshell

How is this done?

1. Suppose the ML/AI model depends on parameters, θ.

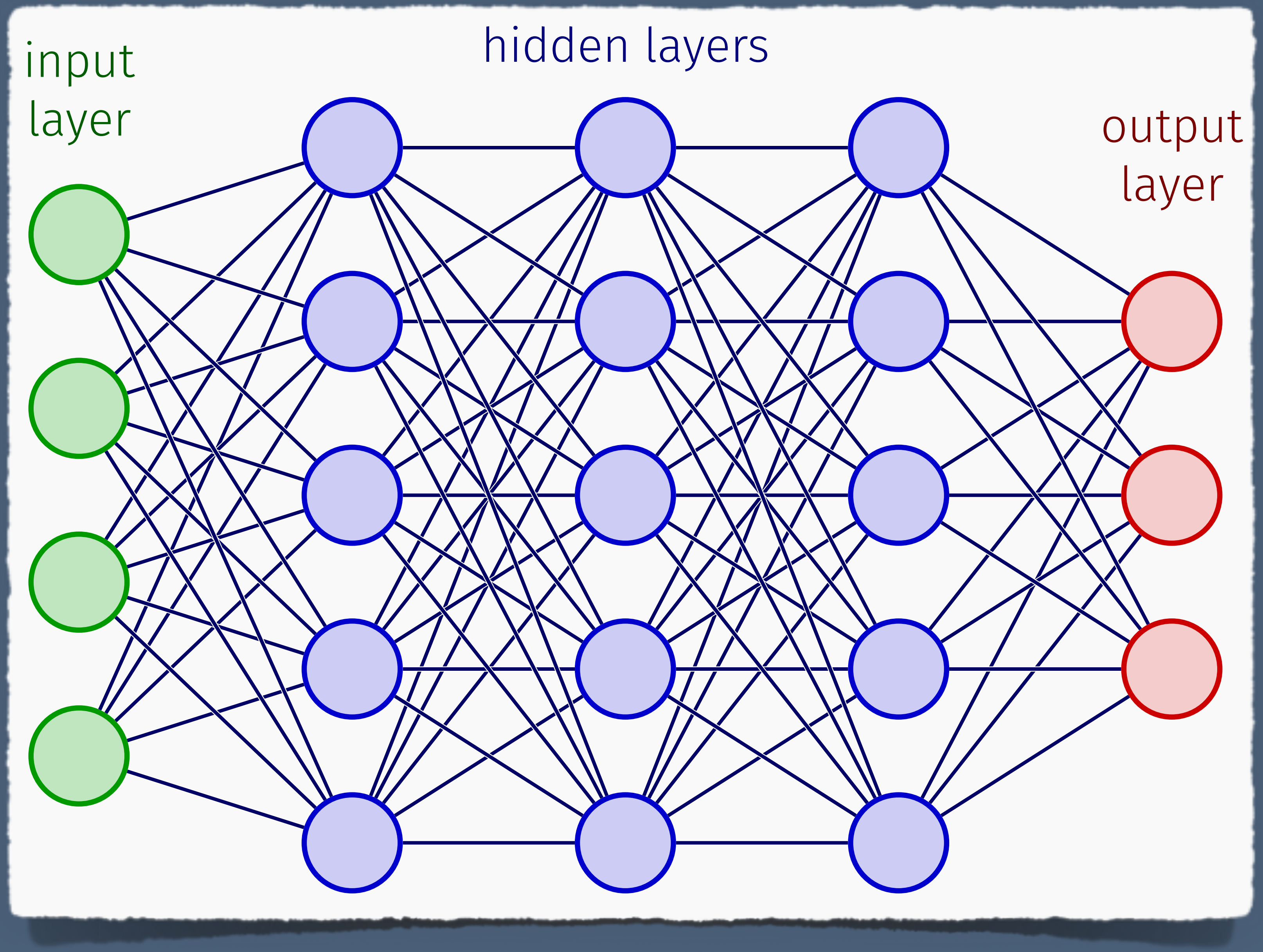


ML/AI in a Nutshell

How is this done?

1. Suppose the ML/AI model depends on parameters, θ.

2. Define a loss function $\mathscr{L}$ that measures the difference between model predictions and measured observations (eg. MSE, cross-entropy).



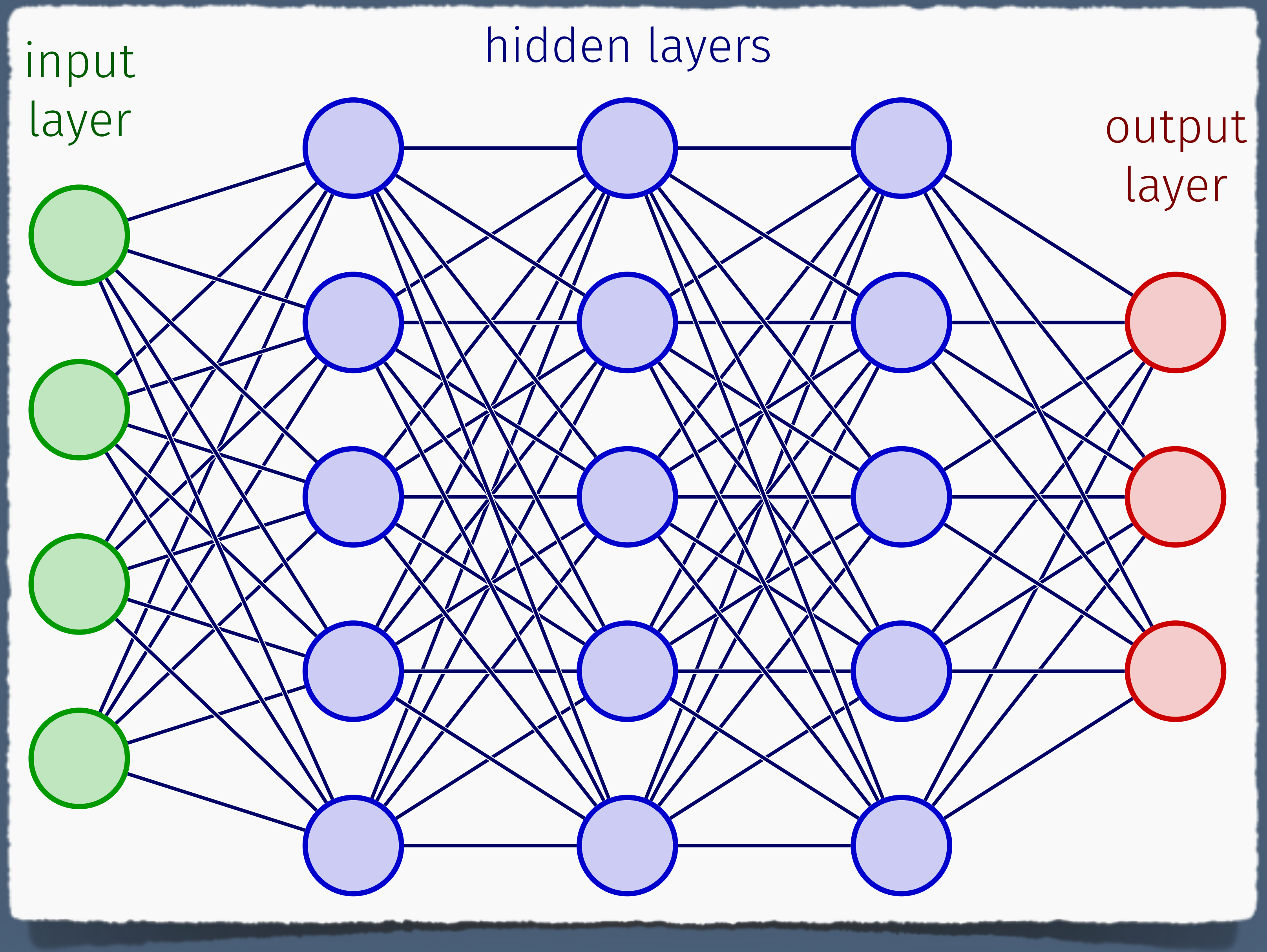
ML/AI in a Nutshell

How is this done?

1. Suppose the ML/AI model depends on parameters, $\boldsymbol{\theta}$.

2. Define a loss function $\mathscr{L}$ that measures the difference between model predictions and measured observations (eg. MSE, cross-entropy).

3. Find the optimal parameters $\boldsymbol{\theta}^*$ of the ML model that minimize the loss function.



ML/AI in a Nutshell

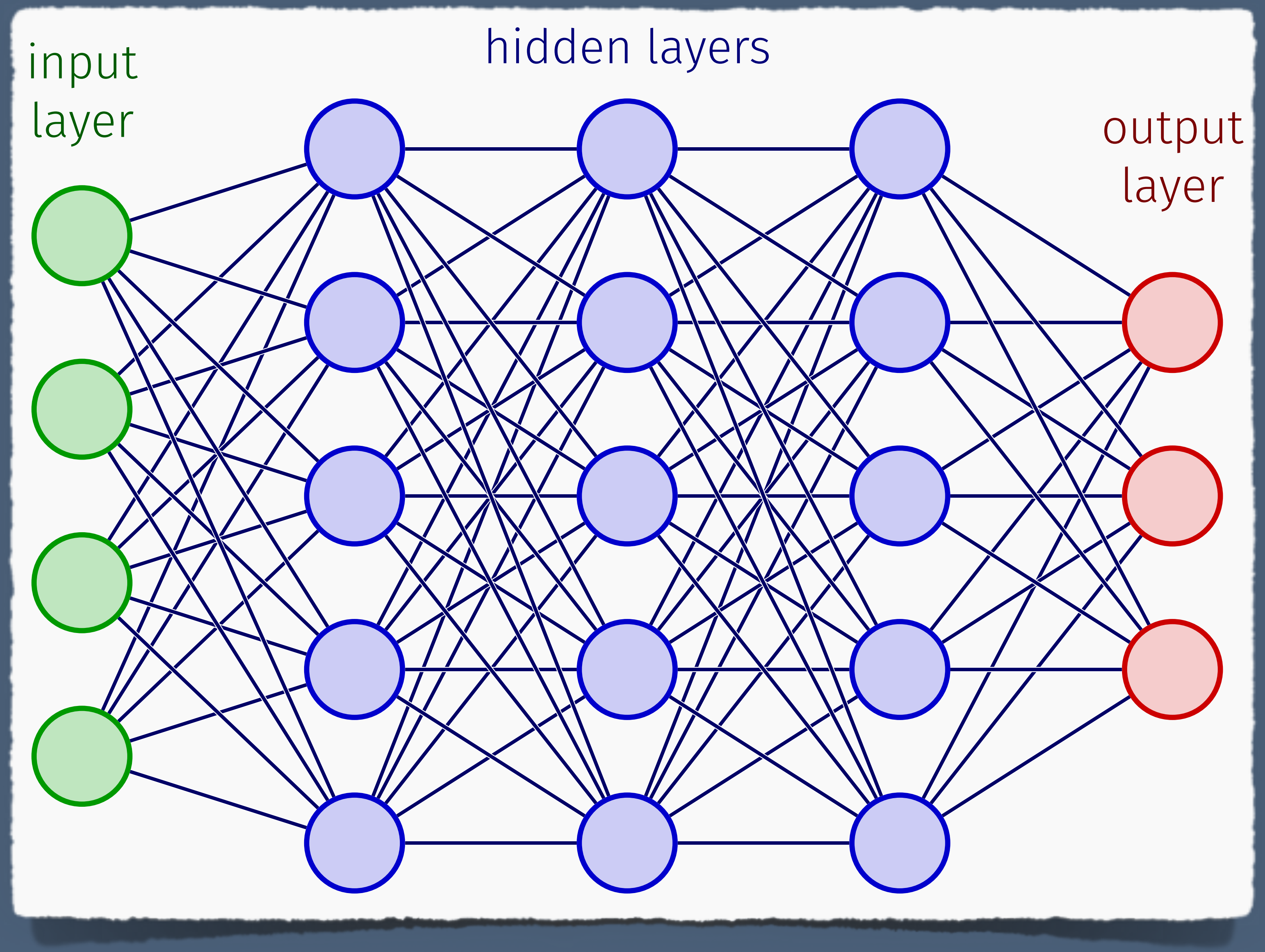
How is this done?

1. Suppose the ML/AI model depends on parameters, $\boldsymbol{\theta}$.

2. Define a loss function $\mathscr{L}$ that measures the difference between model predictions and measured observations (eg. MSE, cross-entropy).

3. Find the optimal parameters $\boldsymbol{\theta}^*$ of the ML model that minimize the loss function.

- *This is nothing more than an inverse problem.*



ML/AI in a Nutshell

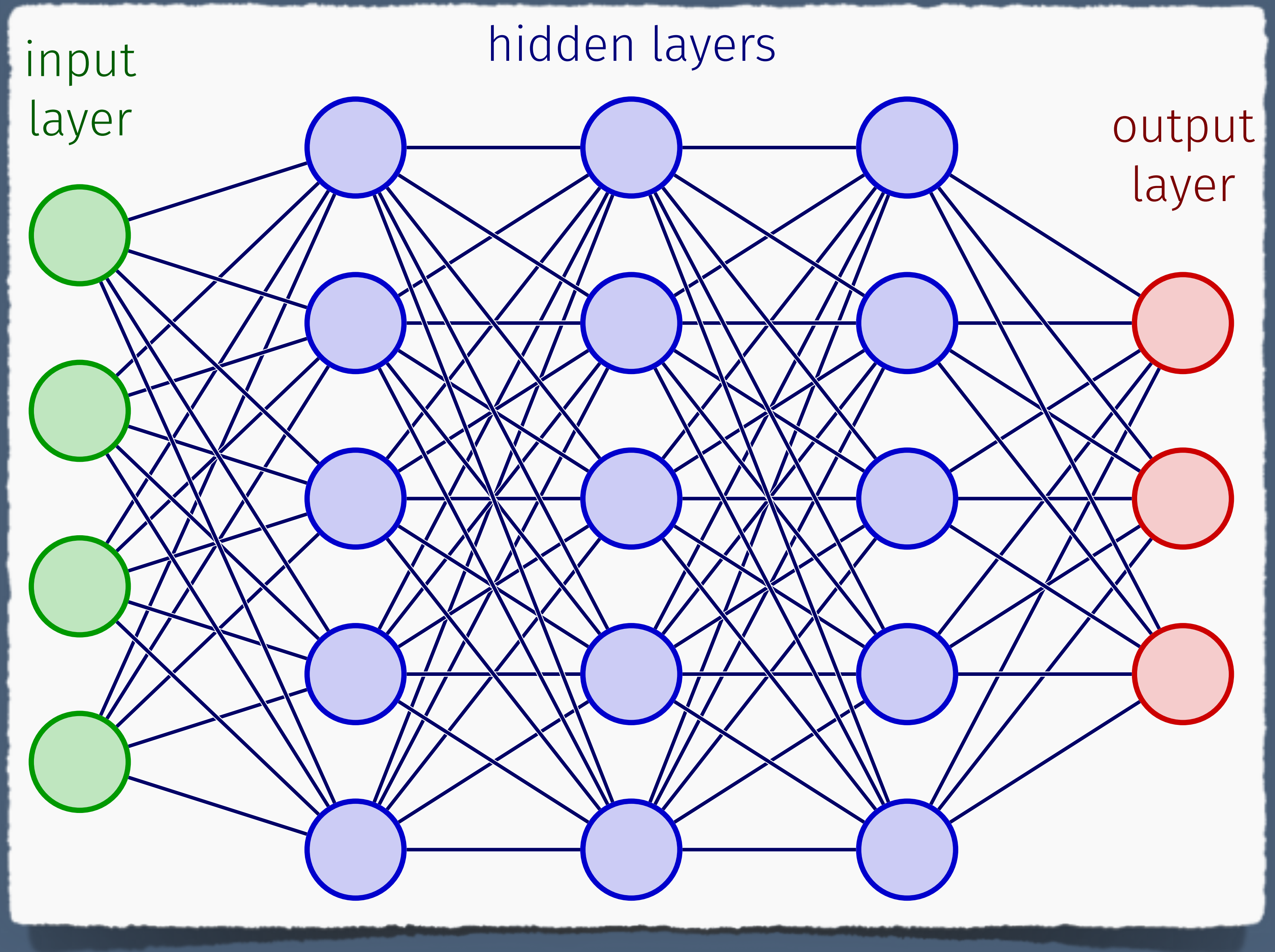
How is this done?

1. Suppose the ML/AI model depends on parameters, $\boldsymbol{\theta}$.

2. Define a loss function $\mathscr{L}$ that measures the difference between model predictions and measured observations (eg. MSE, cross-entropy).

3. Find the optimal parameters $\boldsymbol{\theta}^*$ of the ML model that minimize the loss function.

- *This is nothing more than an inverse problem.*
- *Inverse problems are at the core of* ***Digital Twins***.



ML/AI in a Nutshell

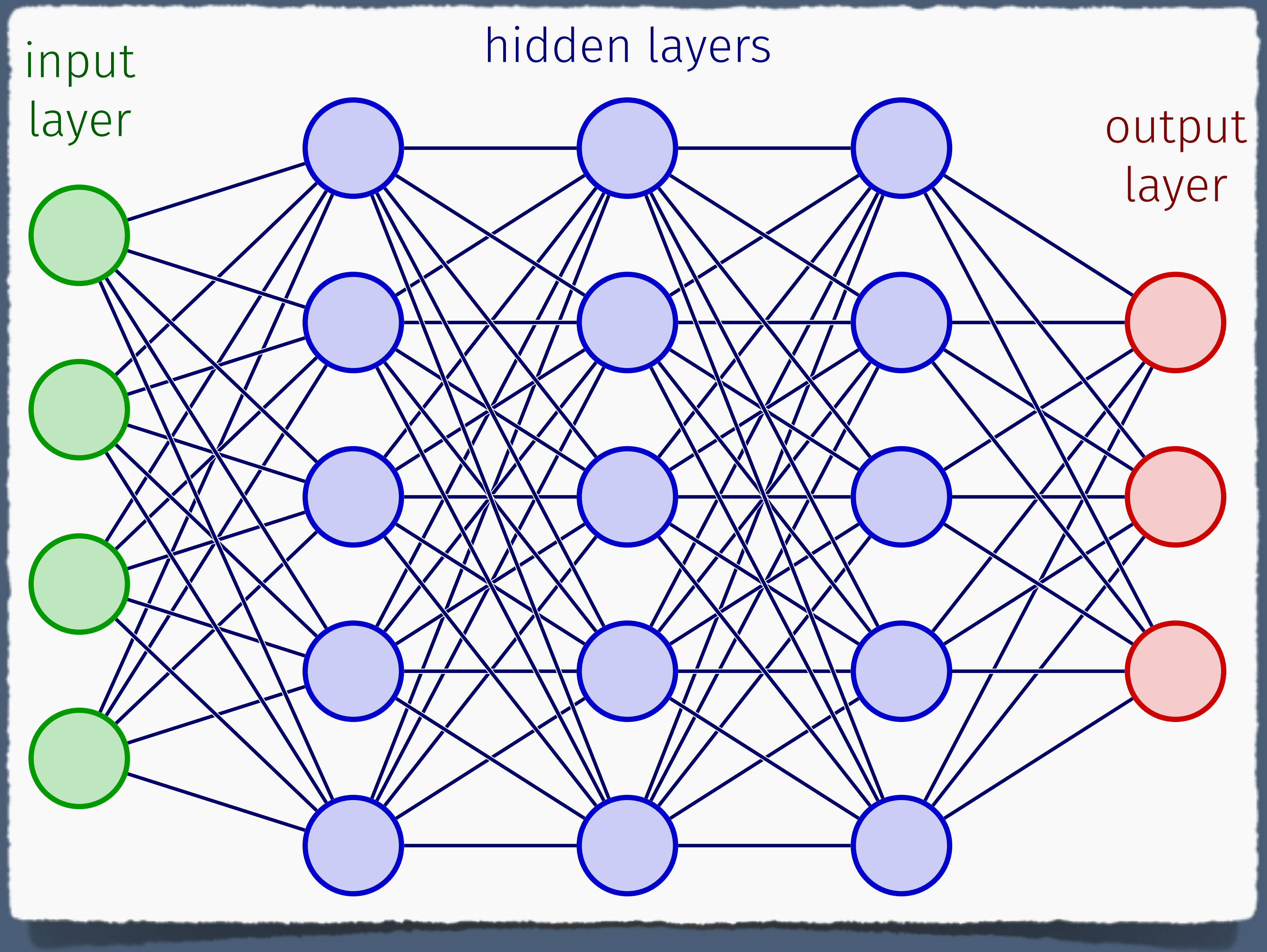
How is this done?

1. Suppose the ML/AI model depends on parameters, $\boldsymbol{\theta}$.

2. Define a loss function $\mathscr{L}$ that measures the difference between model predictions and measured observations (eg. MSE, cross-entropy).

3. Find the optimal parameters $\boldsymbol{\theta}^*$ of the ML model that minimize the loss function.

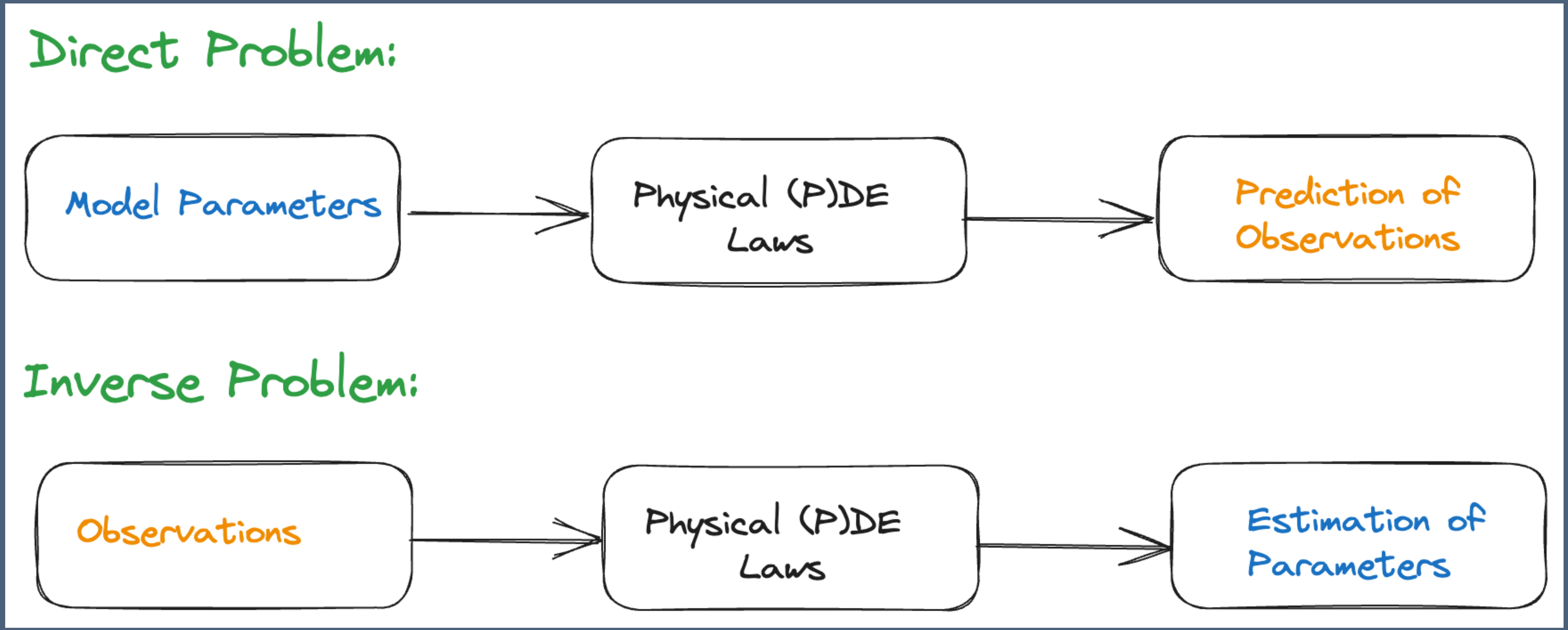
- *This is nothing more than an inverse problem.*
- *Inverse problems are at the core of* ***Digital Twins****.*
- *Scientific ML will improve certifiability—see below.*



Inverse Problems

Classes

- Linear - Nonlinear
- Deterministic - Statistical
- Static - Dynamic
- Well-posed - Ill-posed



Inverse Problems

Deterministic

- Given a physical relation

$$F(u;\boldsymbol{\theta}) = 0 \tag{2}$$

represented by an IBVP, or other functional relationship, with

⇒ u the physical quantity
⇒ $\boldsymbol{\theta}$ the (material/medium) properties/parameters

- Inverse Problem is defined as:

⇒ **Given** observations/measurements of u at the locations $\mathbf{x} = \{x_i\}$

$$\mathbf{u}^{\text{obs}} = \{u(x_i)\}_{i\in\mathcal{I}}$$

⇒ **Estimate** the parameters $\boldsymbol{\theta}$ by minimizing a loss/objective/cost function

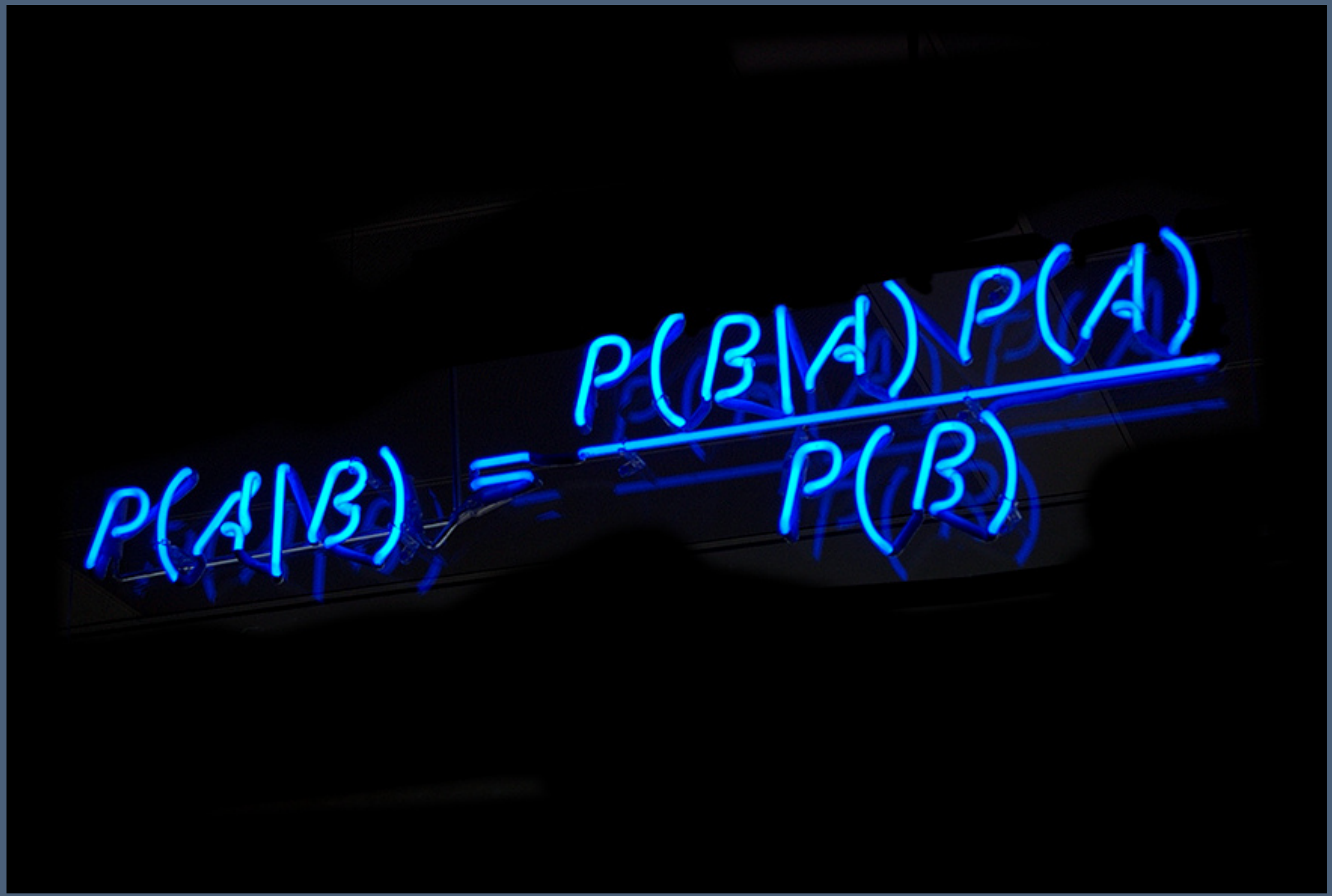
$$L(\boldsymbol{\theta}) = \left\| u(\mathbf{x}) - \mathbf{u}^{\text{obs}} \right\|_2^2$$

subject to (2).

Inverse Problems

Statistical (Stochastic)

- Idea: reformulate inverse problems as problems of statistical inference by means of Bayesian statistics.
  - All quantities and parameters are modeled as random variables.
  - Ill-posed IP becomes **well-posed** in this setting!
- From the perspective of statistical inversion theory, the solution to an inverse problem is the probability distribution of the quantity of interest when all information available has been incorporated in the model.
- This distribution, the posterior distribution, describes the degree of confidence about the quantity of interest, after the measurement has been performed.

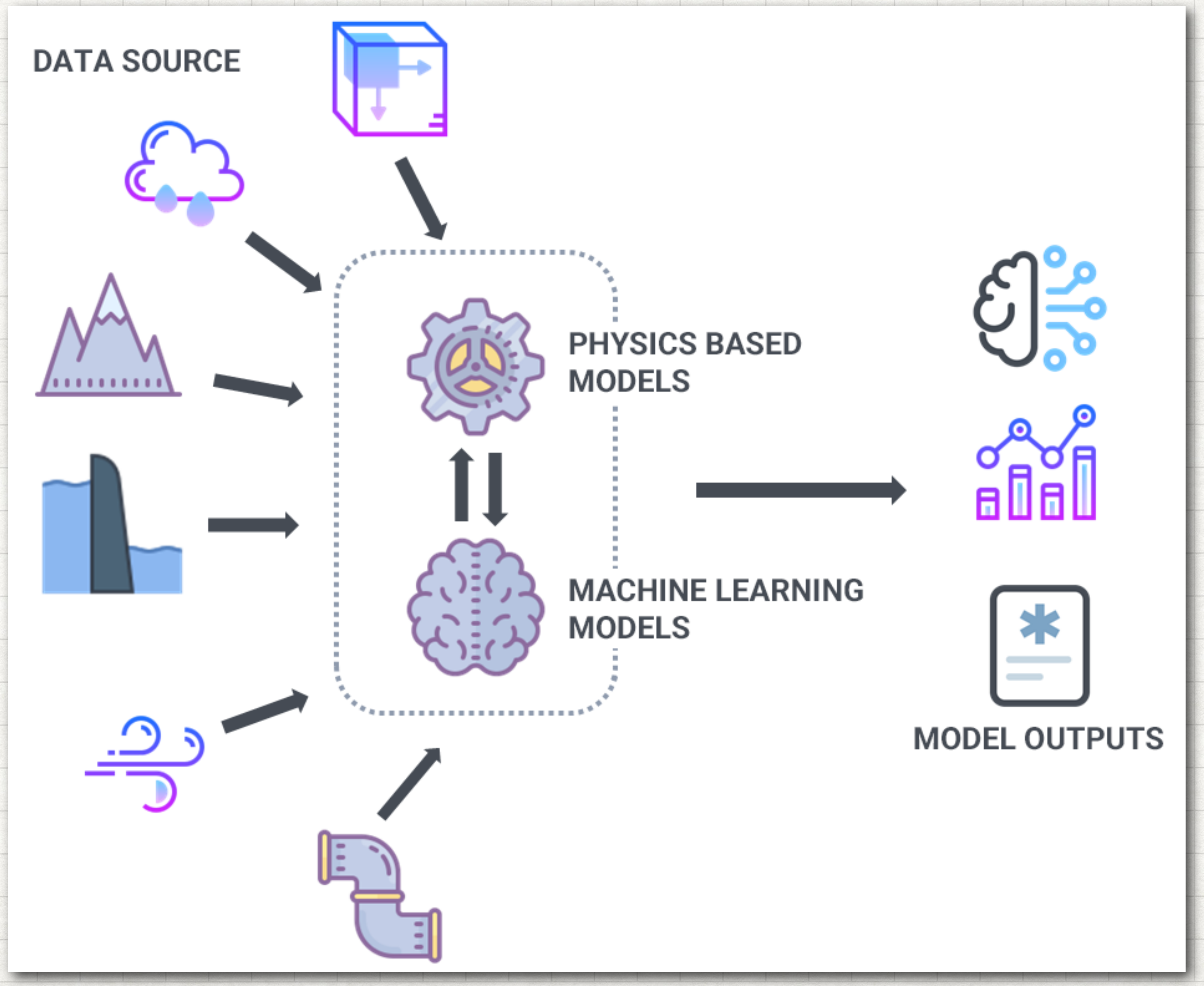


$$\boxed{f(\text{parameter} \mid \text{data}) \propto f(\text{data} \mid \text{parameter})\, f(\text{parameter})}$$

WHAT IS SCIENTIFIC ML?

TWO WORLDS UNITED

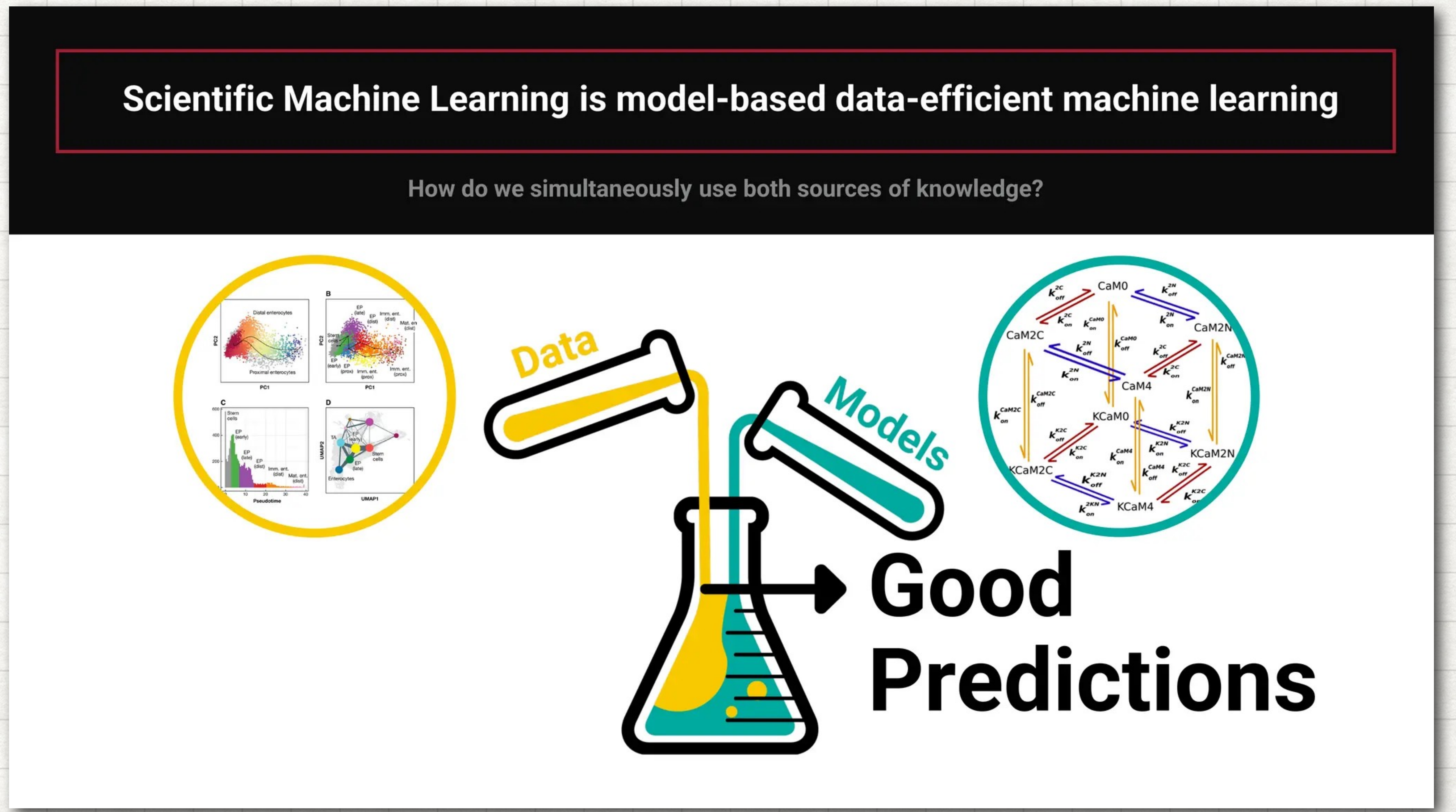
Scientific Machine Learning (SciML) is a field of research that combines traditional *scientific modeling* with *machine learning* techniques. It aims to develop new methods and tools for solving scientific problems that are more accurate, efficient, and generalizable than traditional methods.



WHAT IS SCIENTIFIC ML?

BLENDING

- SciML should exploit any available, underlying physical knowledge/principles:
  - Laws, equations.
  - Conserved quantities.
  - Symmetries, etc.
- Improves interpretability and certifiability—no more black boxes!
- Reduces needs for vast amounts of training data.
- Assists in DL network design (theory-inspired).
- Incorporates ML models into physical plants/ devices.



[C. Rackaukas]

Scientific Machine Learning - relation with DTs

- ML and DTs are essentially optimization problems, where a loss/mismatch function is minimised.
  - This is just a norm of the difference between model output and measured data.
  - It generates very ill-posed problems (very high dimensional, noisy, etc.).
- But, ML does it extremely well (optimisation + regularisation) with amazing tools.
  - Neural networks, Stochastic gradient methods
  - Back propagation is equivalent to a classical adjoint method.
  - Mini-batches (don't try to minimise over all the variables at once), dropout (UQ).
- **Q: Does the ML solution respect the physics?**
- **A: Use SciML!**

SciML-Theory

- Recall: there are two possible optimization strategies for constraining the NN (ML) to respect the physics
  1. Hard constraints
  2. Soft constraints

- Suppose we have a (P)DE of the form

$$\mathcal{F}(u(\mathrm{x},t)) = 0, \quad \mathrm{x} \in \Omega \subset \mathbb{R}^d, \quad t \in [0,T]\,,$$

  where

  ⇒ $\mathcal{F}$ is a differential operator representing the (P)DE
  ⇒ $u(\mathrm{x},t)$ is the state variable (i.e., quantity of interest), with $\mathrm{x},t)$ the space-time variables
  ⇒ T is the time horizon and Ω is the spatial domain (empty for ODEs)
  ⇒ initial and boundary conditions must be added for the problem to be well-posed

- Hard constraint: solve the contrained optimization problem

$$\min_\theta \mathcal{L}(u) \quad \text{s.t.} \quad \mathcal{F}(u) = 0,$$

  where

  ⇒ $\mathcal{L}(u)$ is the data (mismatch) loss term
  ⇒ $\mathcal{F}$ is the constraint on the residual of the (P)DE under consideration
  ⇒ as was amply discussed in the DA/inverse problem context, this type of (P)DE-constrained optimization is usually quite difficult to code and to solve

- Soft constraint: solve the regularized/penalized uncontrained optimization problem

$$\min_\theta \mathcal{L}(u) + \alpha_\mathcal{F}\mathcal{F}(u), \tag{3}$$

$$\mathcal{L}(u) = \mathcal{L}_{u_0} + \mathcal{L}_{u_b},$$

  where

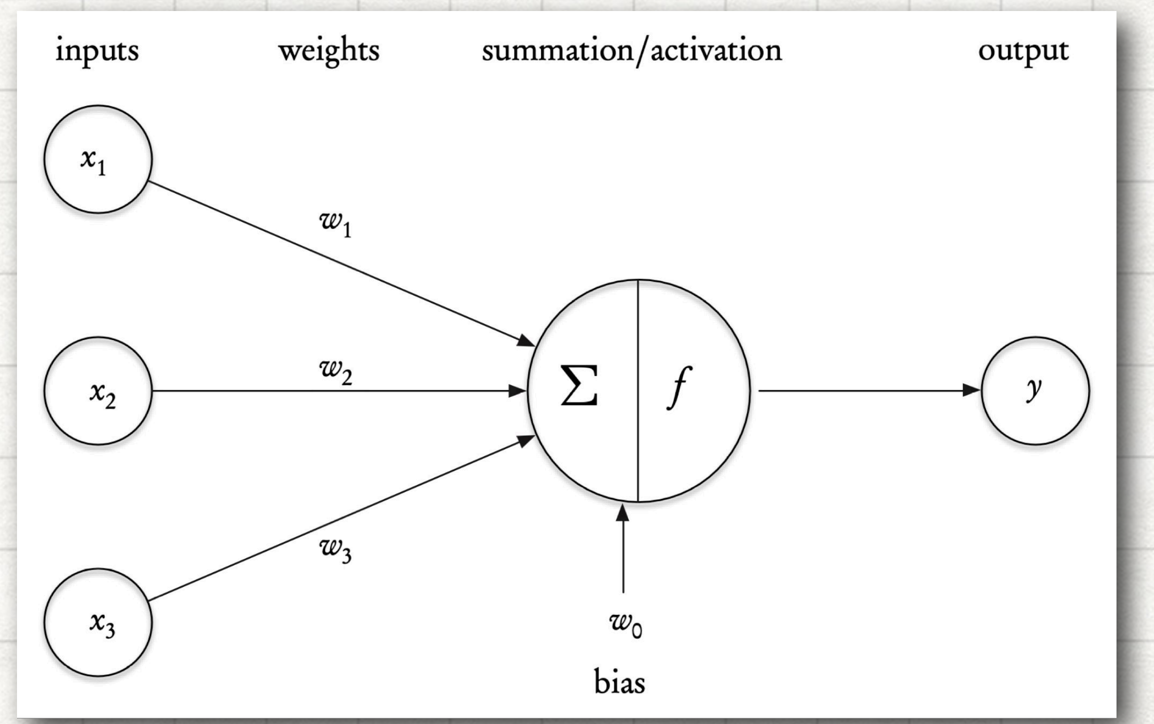
  ⇒ $\mathcal{L}_{u_0}$ represents the misfit of the NN predictions
  ⇒ $\mathcal{L}_{u_b}$ represents the misfit of the initial/boundary conditions

  ⇒ θ represents the NN parameters
  ⇒ $\alpha_\mathcal{F}$ is a regularization parameter that controls the emphasis on the PDE-based residual (which we ideally want to be zero)

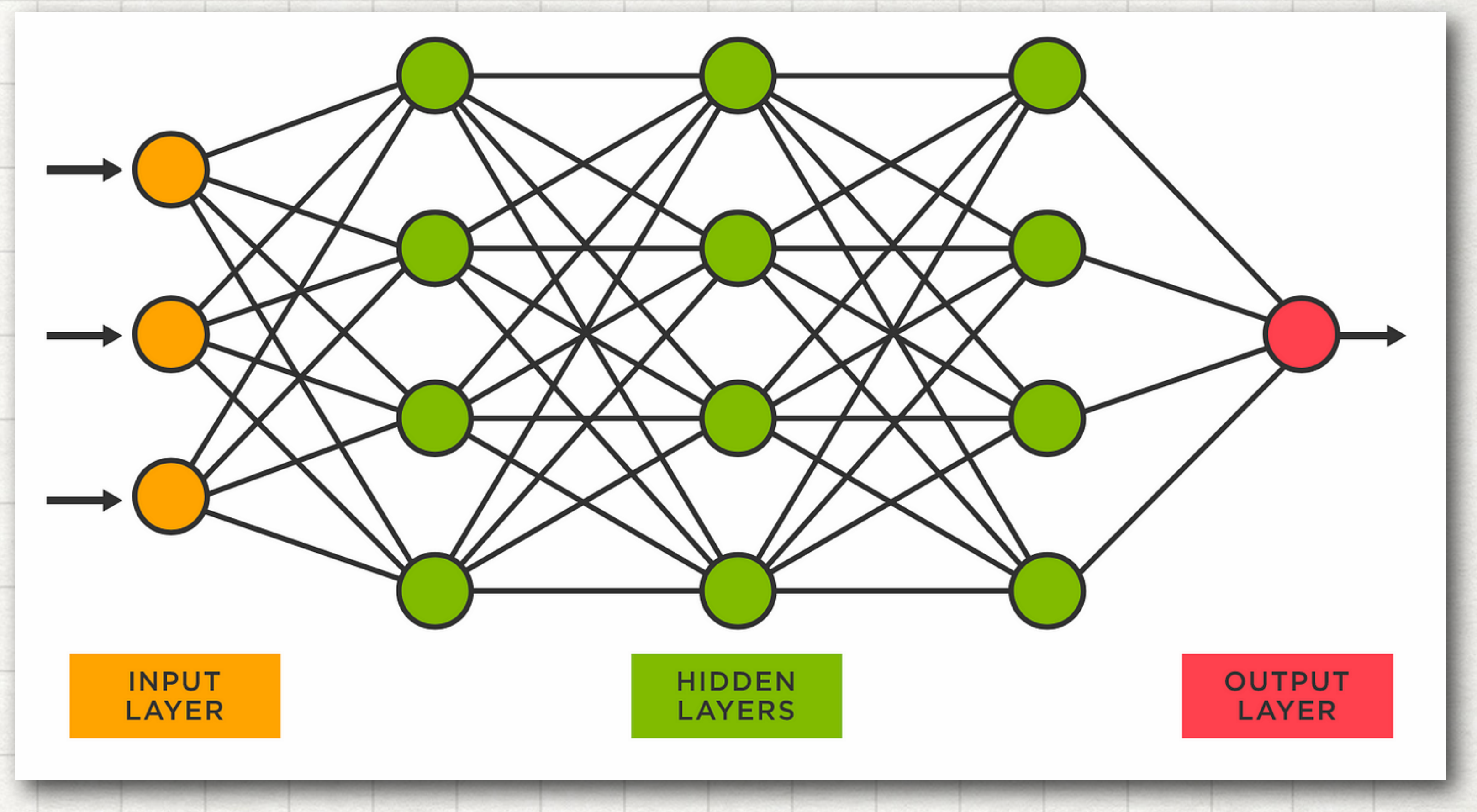
THE MATH BEHIND SCI-ML

APPROXIMATION THEORY

- Multi-layer perceptrons - 1950's - the basis.
- Universal Approximation Property - 1990's - the theory.
- *Differentiable programming - 2020's - makes it all possible! (see next slide)*



$$y = f\left(w_0 + \sum_{i=1}^{3} w_i x_i\right)$$



Theorem 1 (Cybenko 1989). *If σ is any continuous sigmoidal function, then finite sums*

$$G(x) = \sum_{j=1}^{N} \alpha_j \sigma\left(w_j \cdot x + b_j\right)$$

are dense in $C(I_d)$.

Theorem 2 (Pinkus 1999). *Let $\mathbf{m}_i \in \mathbb{Z}^d$, $i = 1, \ldots, s$, and set $m = \max_i \left|\mathbf{m}^i\right|$. Suppose that $\sigma \in C^m(\mathbb{R})$, not polynomial. Then the space of single hidden layer neural nets,*

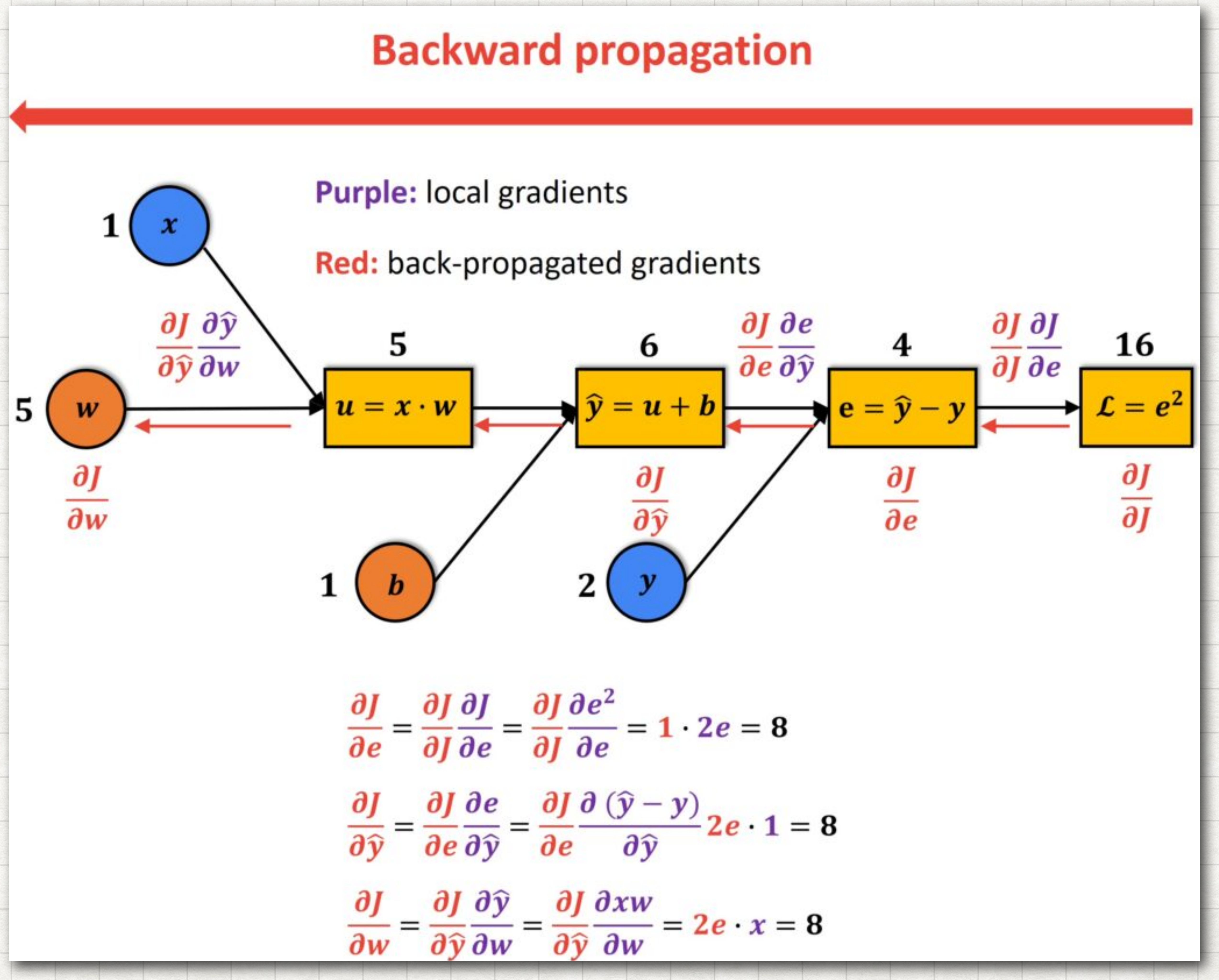
$$\mathcal{M}(\sigma) = \operatorname{span}\left\{\sigma(\mathbf{w} \cdot \mathbf{x} + b) \colon \mathbf{w} \in \mathbb{R}^d,\, b \in \mathbb{R}\right\},$$

is dense in $C^{\mathbf{m}^1,\ldots,\mathbf{m}^s}(\mathbb{R}^d) \doteq \cap_{i=1}^{s} C^{\mathbf{m}^i}(\mathbb{R}^d)$.

THE CODE BEHIND SCI-ML

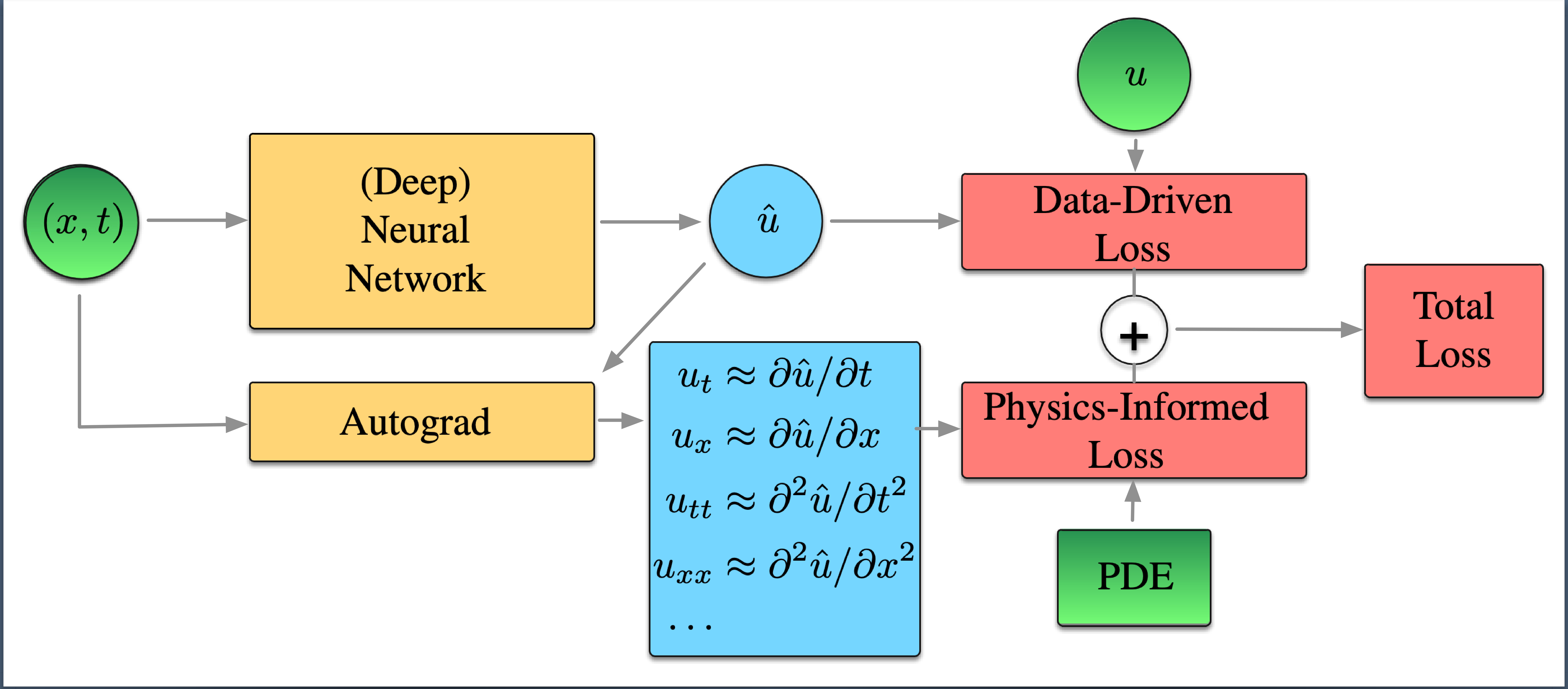
"AUTOGRAD"

- Differentiable programming - 2020's - makes it all possible!
- Based on:
  - Computational graphs
  - Chain-rule for differentiation



PINN

For any ODE/PDE



PINN Example

Example: IBVP for Diffusion Equation

Compute $u(\mathbf{x},t)\colon \Omega\times[0,T]\to\mathbb{R}$ such that

$$\frac{\partial u(\mathbf{x},t)}{\partial t} - \nabla\cdot(\lambda(x)\nabla u(\mathbf{x},t)) = f(\mathbf{x},t) \quad \text{in } \Omega\times(0,\mathrm{T}), \tag{6}$$

$$u(\mathbf{x},t) = g_D(\mathbf{x},t) \quad \text{on } \partial\Omega_{\mathrm{D}}\times(0,\mathrm{T}),$$

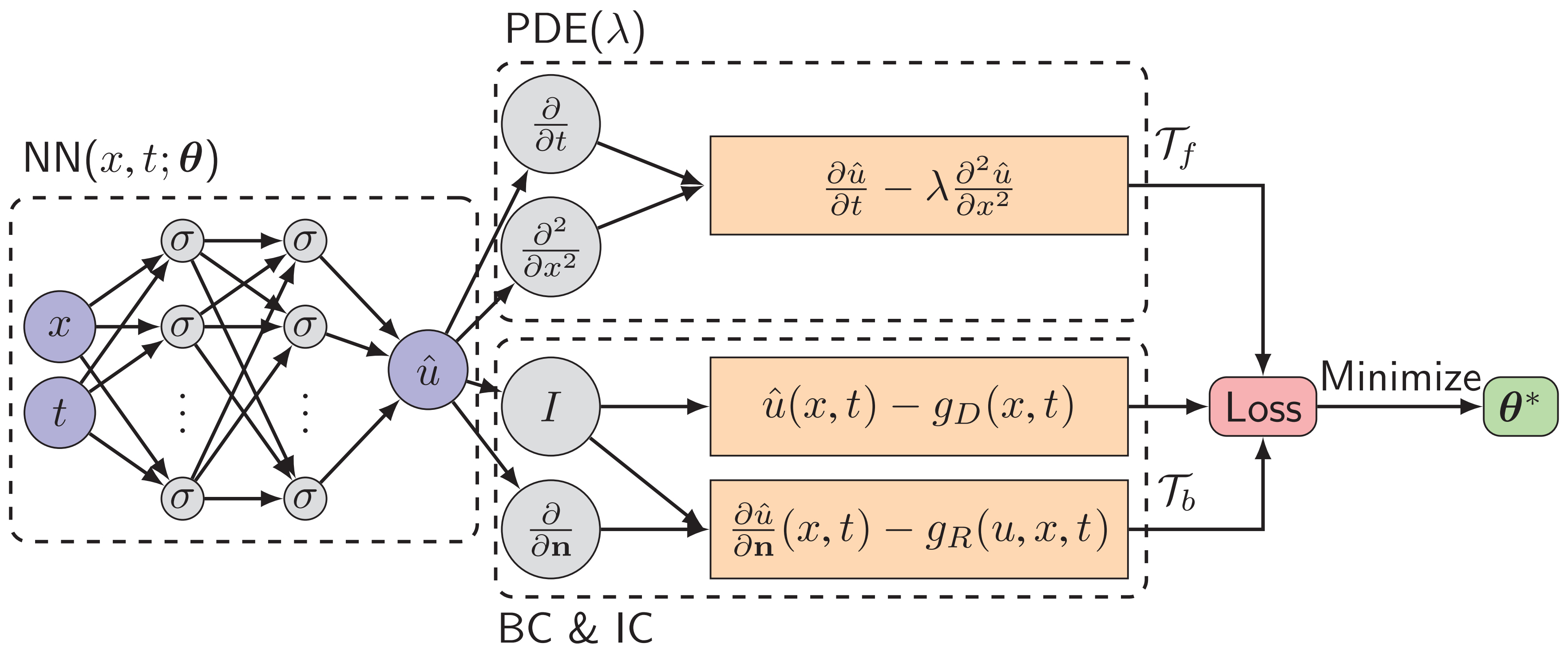
$$-\lambda(x)\nabla u(\mathbf{x},t)\cdot\mathbf{n} = g_R(\mathbf{x},t) \quad \text{on } \partial\Omega_{\mathrm{R}}\times(0,\mathrm{T}),$$

$$u(\mathbf{x},0) = u_0(\mathbf{x}) \quad \text{for x} \in \Omega.$$

Note that $\lambda(x)$ is, in general, a tensor (matrix) with elements λ_{ij}.

- **Direct problem**: given λ, compute u.
- **Inverse problem**: given u, compute λ.

PINN for the Diffusion Equation



[Credit: Lu, Karniadakis, SIAM Review, 2021]

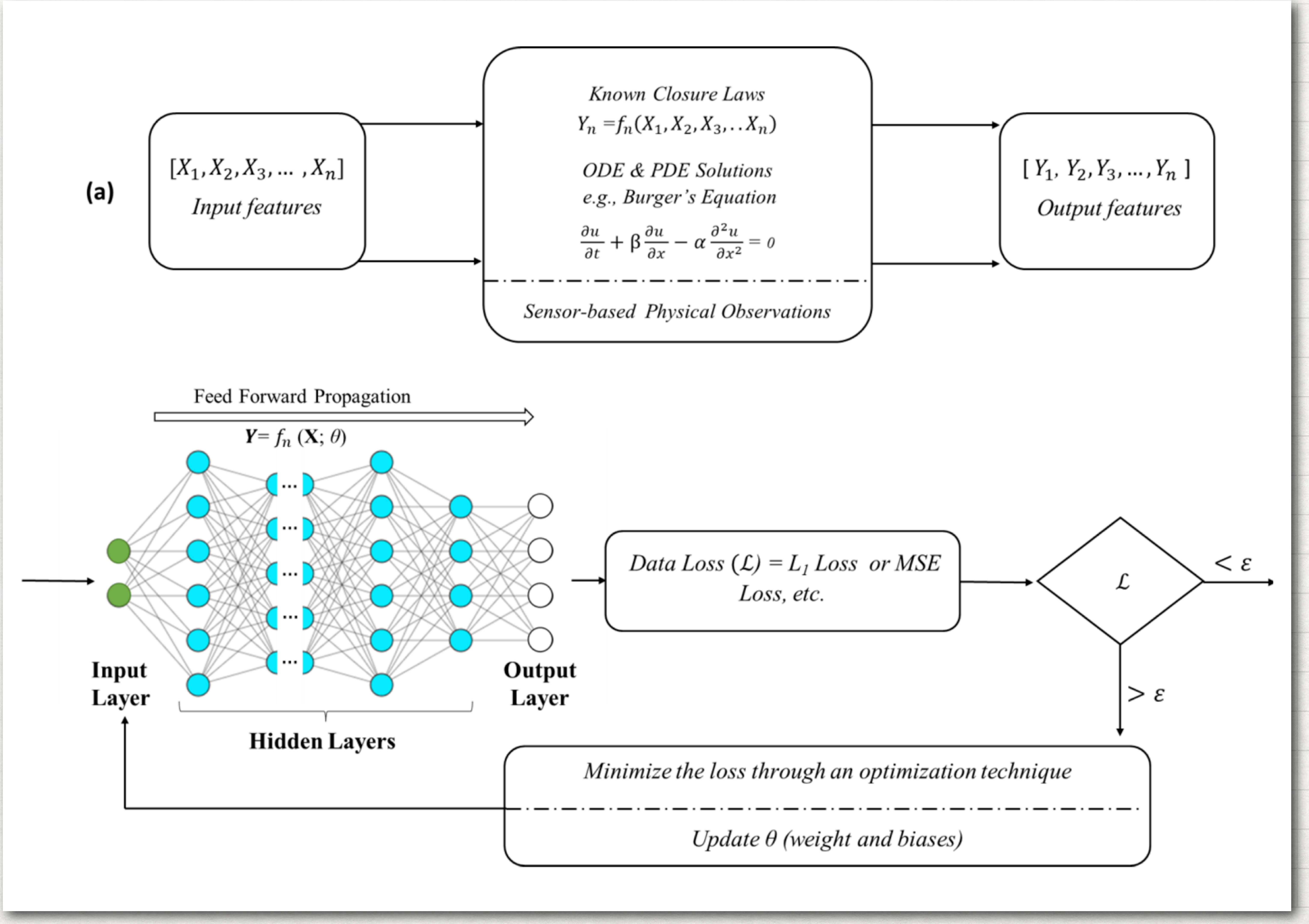
- Use FCNN to approximate u at the selected points x, with training data at residual points $\mathcal{T}_f$ and $\mathcal{T}_b$
- Use AD to compute derivatives for the PDE and the boundary/initial conditions
- Minimize the augmented, weighted loss function

HOW IS SCIENTIFIC ML DONE?

BLENDING

3 possible paths:

1. Physics-guided NNs
2. Physics-informed NNs
3. Physics-encoded NNs

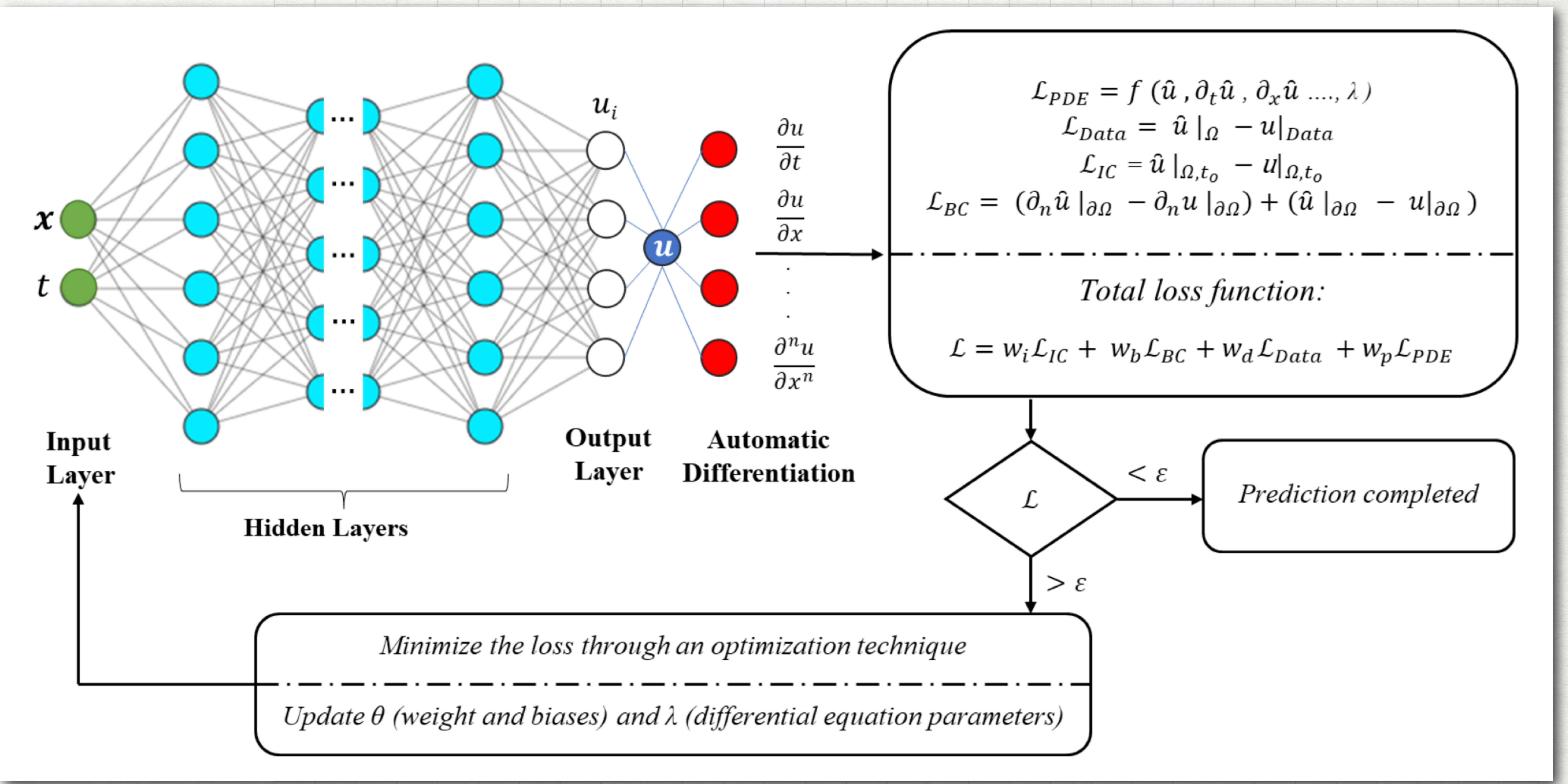


HOW IS SCIENTIFIC ML DONE?

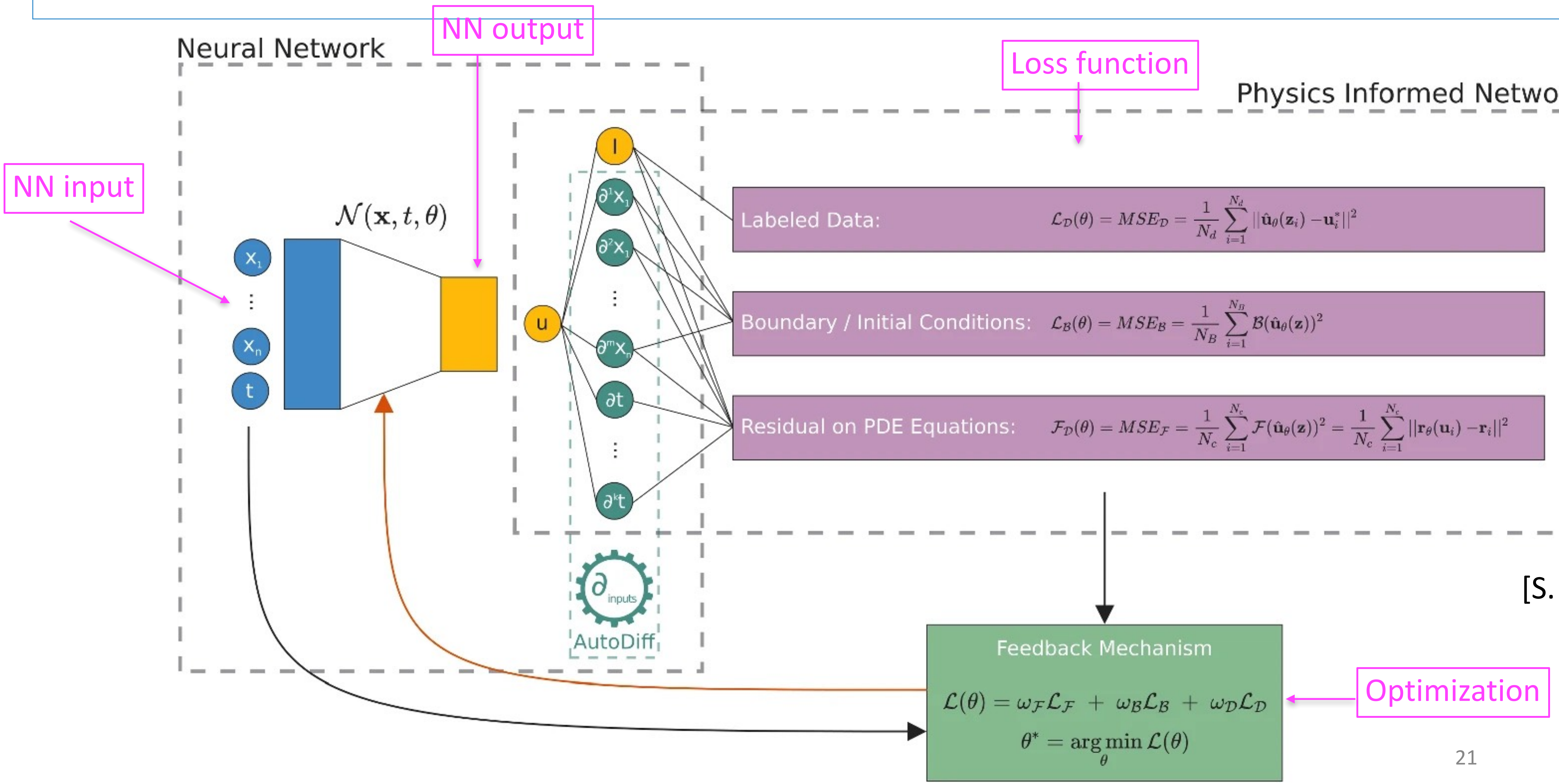
BLENDING

3 possible paths:

1. Physics-guided NNs
2. Physics-informed NNs
3. Physics-encoded NNs



SciML - PINN - Blending



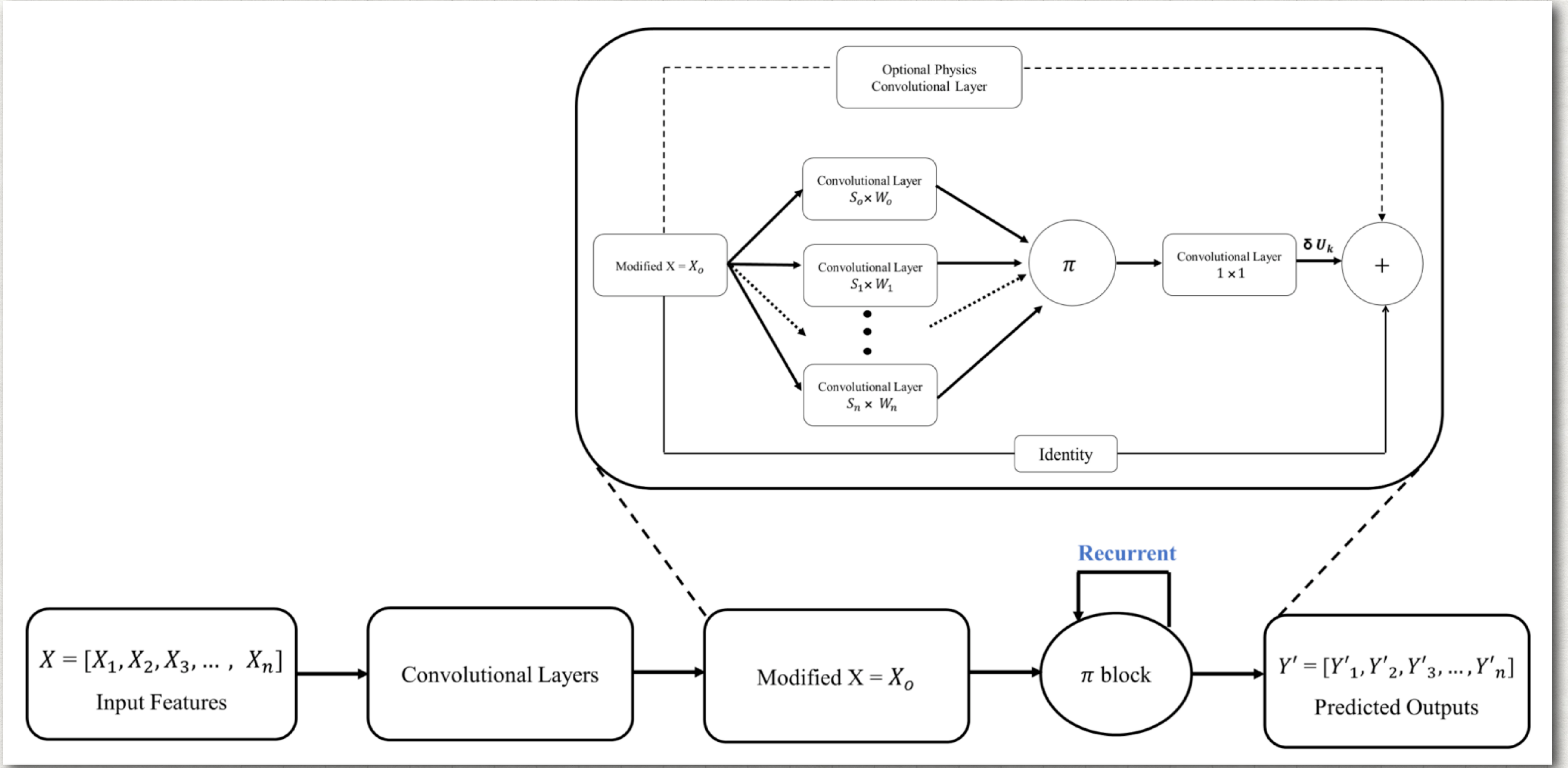
[S. Cuomo, et al]

HOW IS SCIENTIFIC ML DONE?

BLENDING

3 possible paths:

1. Physics-guided NNs
2. Physics-informed NNs
3. Physics-encoded NNs



Examples & Applications

Domains in biomedical sciences

- Epidemiology
- Omics
- Medicine
- Drug design

- COVID,
- Anthrax,
- HIV,
- Zika,
- Smallpox,
- Tuberculosis,
- Pneumonia,
- Ebola,
- Dengue,
- Polio,
- Measles.

Epidemiology with Scientific machine learning

- ML can be employed to study complex interactions between different biological systems, such as signaling pathways and metabolic networks, to advance our understanding of various biological phenomena and improve the diagnosis and treatment of diseases. These technologies have the potential to significantly impact biological research in a variety of areas, including infectious diseases and epidemiology.

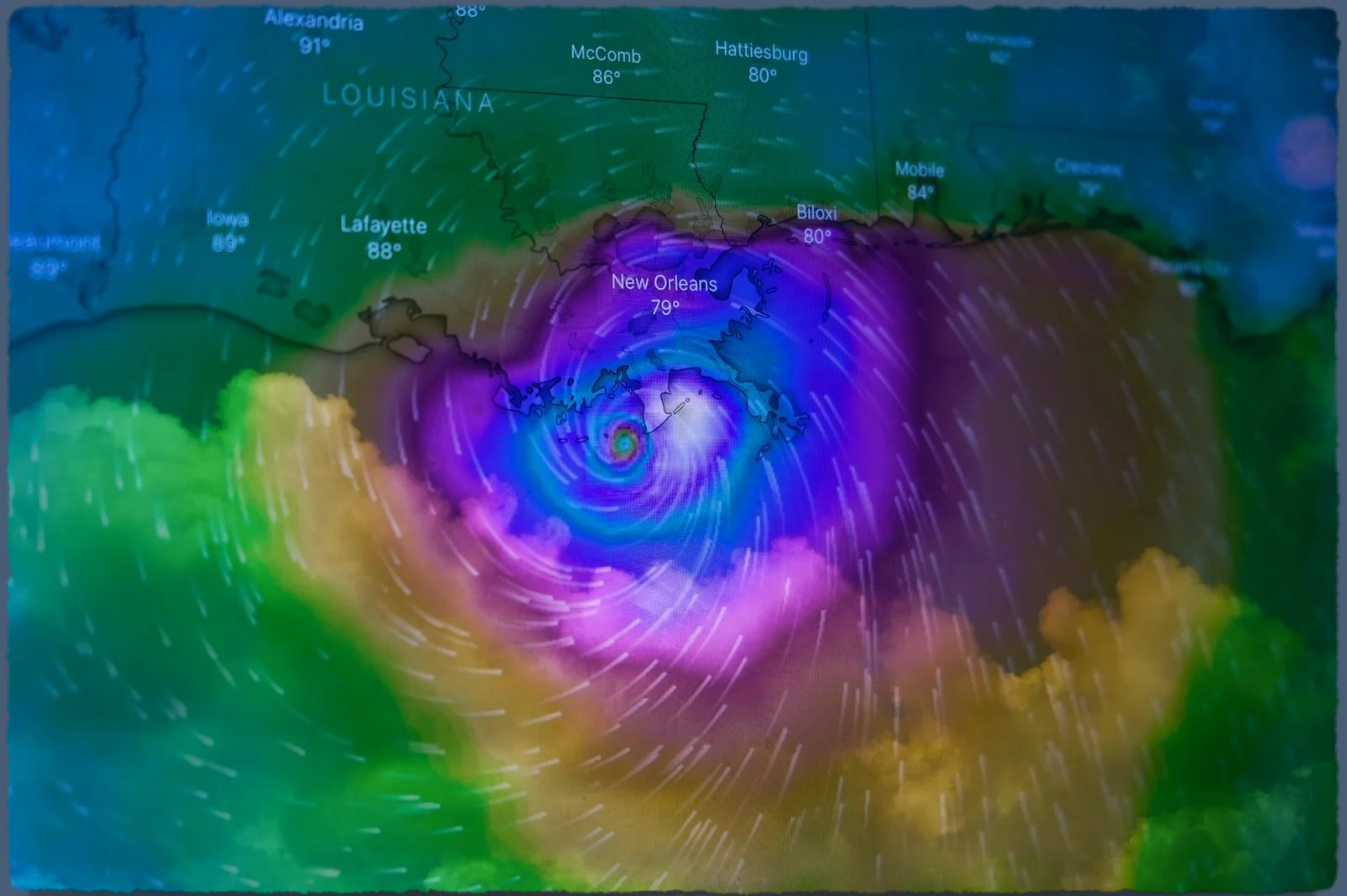
- ML can be used to analyze large datasets, such as genomic data, to identify patterns and trends relevant to the understanding and treatment of infectious diseases

- ML can be employed to predict the likelihood of certain outcomes, such as the spread of a disease, based on historical data and by analyzing datasets generated by epidemiological studies. This can aid epidemiologists in preventing or mitigating outbreaks of infectious diseases, such as influenza and HIV.

- AI-based DTs can also be utilized to build predictive models that help researchers understand the relationships between different variables, such as gene expression and disease risk, interactions betwee pathogens and host organisms at the molecular level, and complex molecular interactions within biomolecules.

Other Scientific Domains

- Fluid dynamics, flow problems
  - Oceanography
  - Meteorology
  - Geophysical flows
  - Aerodynamics
- Solid mechanics
- Material science



USE-CASES

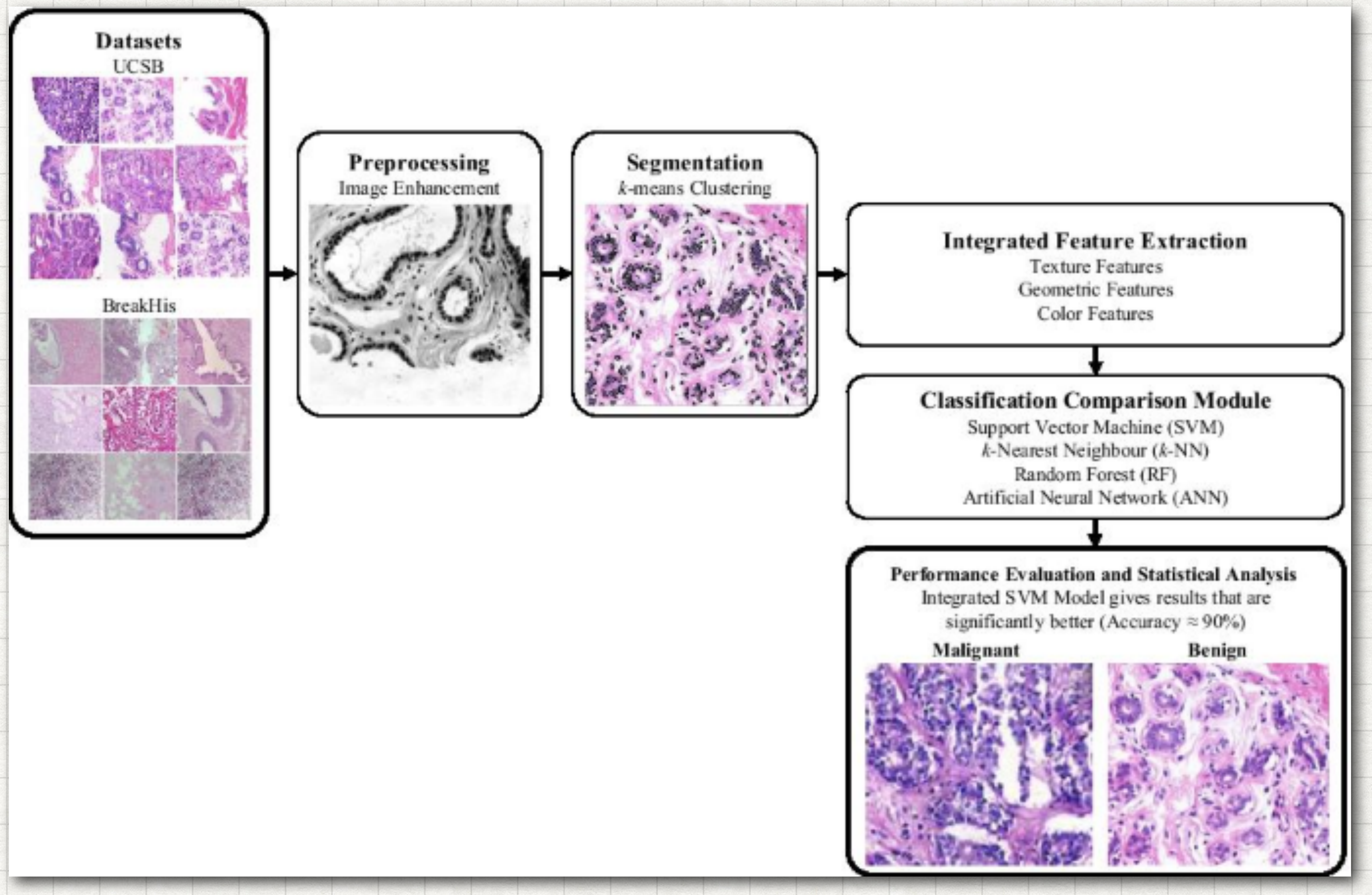
DISEASE DIAGNOSIS

PATTERN MATCHING

* ML algorithms, such as support vector machines (SVM) and random forests (RF), are trained on medical imaging data, such as X-rays or MRIs, to identify patterns associated with specific diseases.

* For example, a SVM model could be trained on a dataset of mammogram images labeled as "normal" or "cancerous" to create a computer-aided detection system for breast cancer.

* CNN's have exceptional pattern-recognition capabilities.



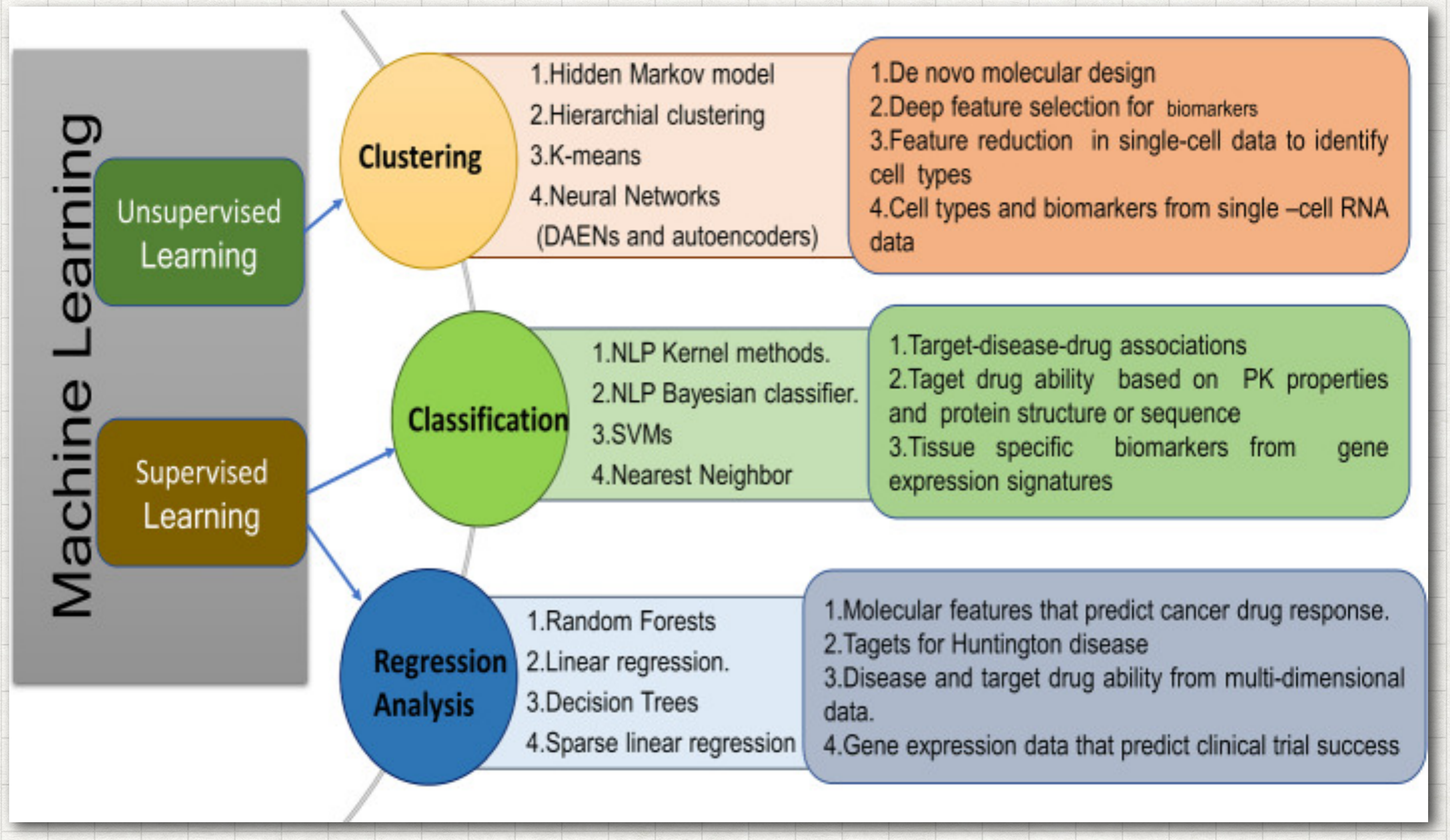
DRUG DISCOVERY

GRAPHS AND PROPERTIES

* ML is used to analyze large datasets of chemical compounds and their properties to predict which ones are likely to make effective drugs.

* For example, a deep learning model called a graph convolutional network (GCN) can be used to learn the structural features of molecules and predict their biological activity.

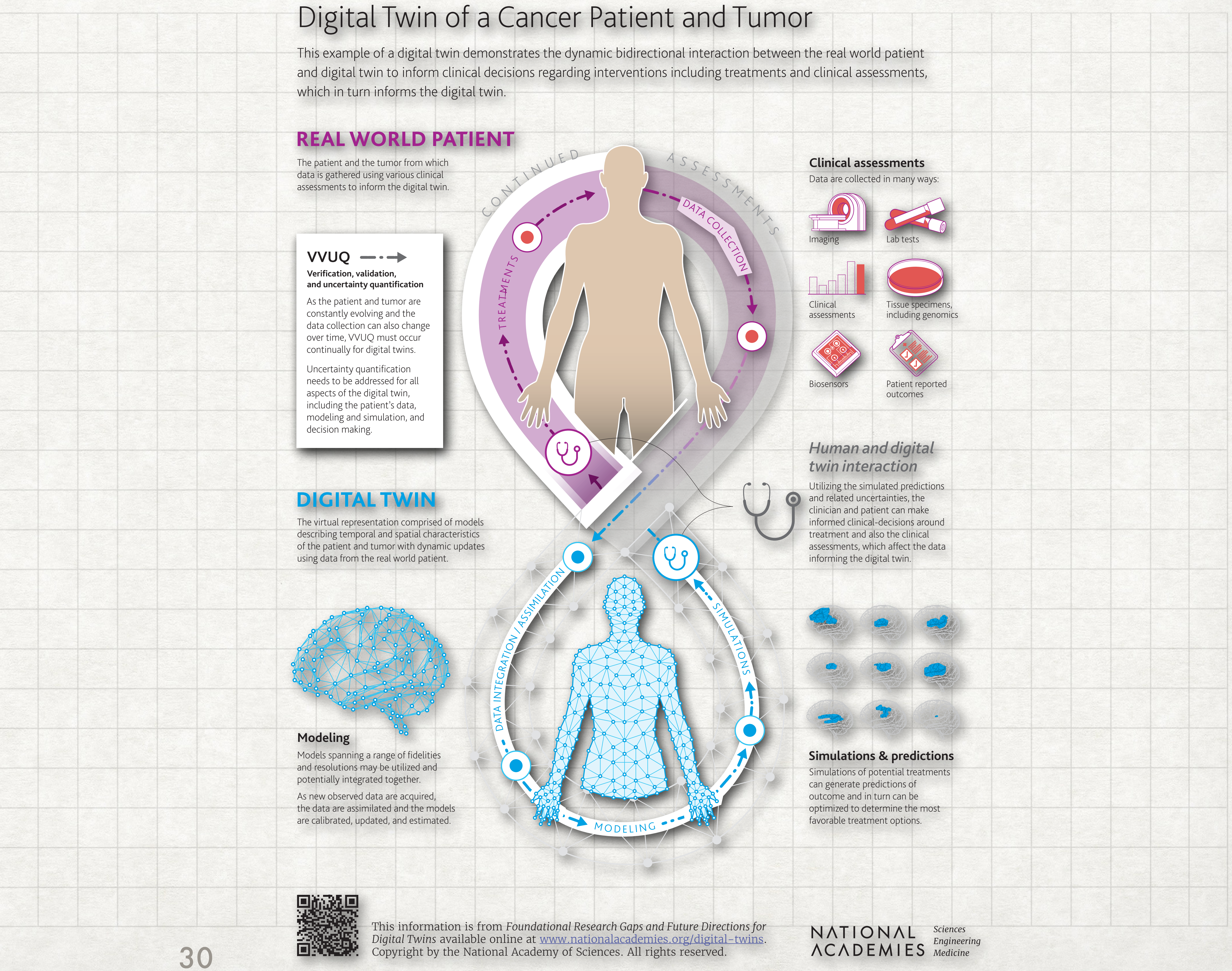
* CANDLE project solves large-scale machine learning problems for three cancer-related pilot applications: the drug response problem, RAS pathway problem, and treatment strategy problem (disease-drug) .



PERSONALIZED MEDECINE

LETS GET PERSONAL

* Machine learning algorithms used to analyze patients' genetic data and other health information to develop personalized treatment plans.
* For example, a decision tree model could be trained on a dataset of patients with similar characteristics to predict which medication will be most effective for a given patient.
* Moving towards a Digital Twin...



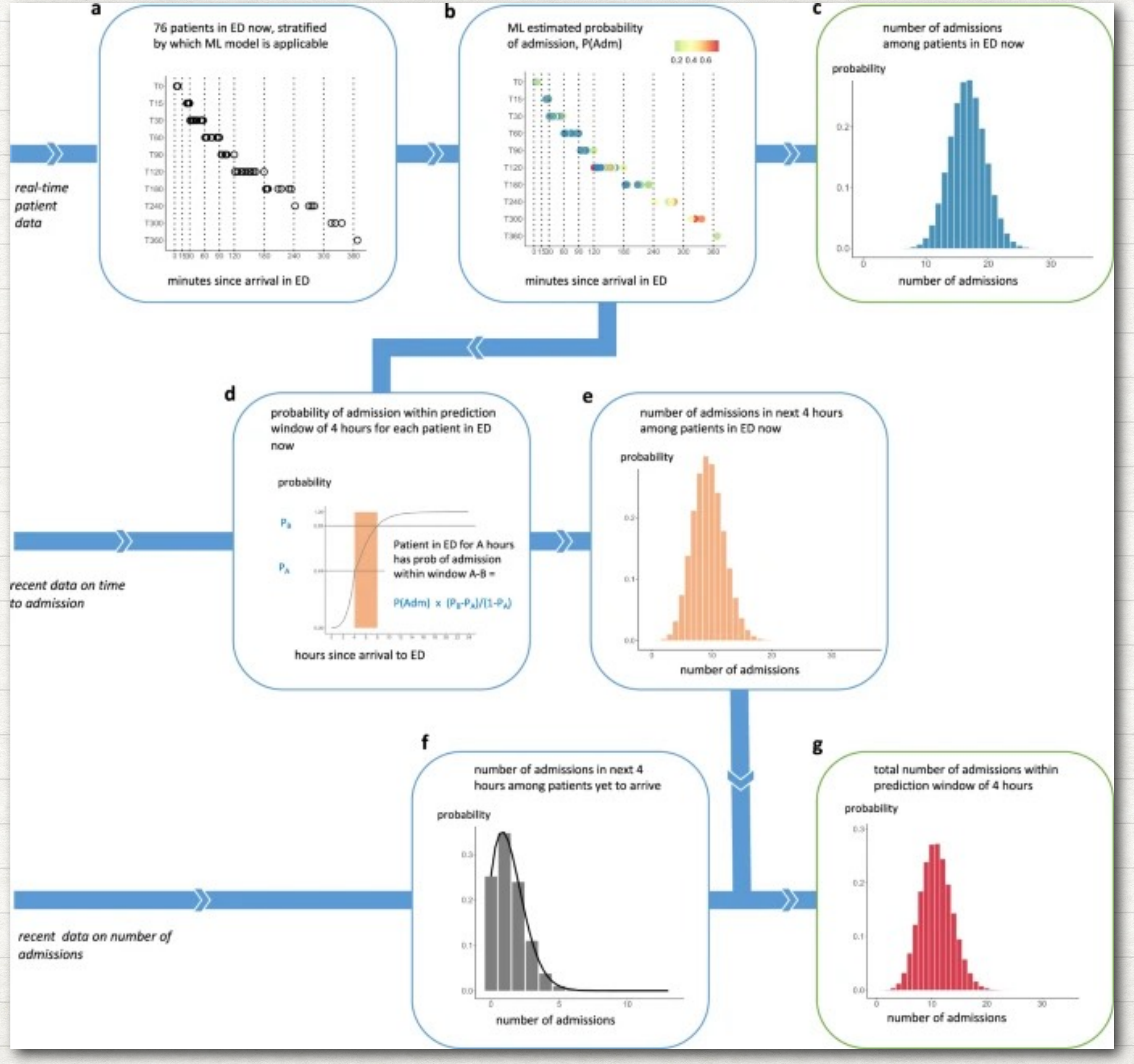
HOSPITAL PLANNING

ADMISSIONS AND READMISSIONS

* ML models, such as logistic regression and artificial neural networks, can be used to analyze patients' electronic health records (EHR) to identify factors that increase the risk of readmission.

* This information can then be used to develop interventions to reduce readmissions.

* Time-series (LSTM) and queuing theory to model admissions fluxes.



Conclusion

SciML for DTs

- Incorporating scientific knowledge (almost) always improves the performance of ML algorithms:

- Restricts the space of ML models.

- Stronger inductive bias.

- Can alleviate ML flaws (poor generalisation, optimisation difficulties, lack of interpretability, large amounts of training data).

Conclusion

SciML for DTs

- Incorporating scientific knowledge (almost) always improves the performance of ML algorithms:
- Restricts the space of ML models.
- Stronger inductive bias.
- Can alleviate ML flaws (poor generalisation, optimisation difficulties, lack of interpretability, large amounts of training data).

- Incorporating Machine Learning in the scientific workflow:
- Enhances the performance (efficiency, accuracy, insights, noise).
- Compensates for unknown/intractable physics.
- Takes advantage of large reservoirs of untapped data.

Thank You!

Contact

- mark.asch@u-picardie.fr
- https://www.linkedin.com/in/mark-asch-8a257130/
- https://github.com/markasch
- https://markasch.github.io/DT-tbx-v1/

REFERENCES

BOOKS & PAPERS

- M. Asch
- S A Faroughi, N Pawar, C Fernandes, M. Raissi, S. Das, N K Kalantari, S K Mahjour. Physics-Guided, Physics-Informed, and Physics-Encoded Neural Networks in Scientific Computing. arXiv, 2023. https://arxiv.org/pdf/2211.07377
- S. Cuomo, V. Schiano Di Cola, F. Giampaolo, G. Rozza, M. Raissi, F. Piccialli. Scientific Machine Learning Through Physics Informed Neural Networks: Where we are and What's Next. Journal of Scientific Computing (2022) 92:88.
- L Lu, P Jin, G Pang, Z Zhang, GE Karniadakis. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. Nature Machine Intelligence 3 (3), 218-229 (2021).
- L. Lu, X. Meng, Z. Mao, G. Karniadakis. DeepXDE: A Deep Learning Library for Solving Differential Equations. SIAM Review, 63, 1 (2021).
- E. Darve, K. Xu. Physics constrained learning for data-driven inverse modeling from sparse observations. J. of Computational Physics, 453. (2022).
- M. Raissi, P. Perdikaris, G. E Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. J. of Computational Physics, 378, pp. 686-707, 2021.
- N. Kovachki, Z. Li, B. Liu, K. Bhattacharya, A. Stuart, A. Anandkulmar. Neural Operator: Learning Maps Between Function Spaces With Applications to PDEs. J. of Machine Learning Research 24 (2023).
- A. Baydin, B. Pearlmutter, A. Radul, J. Siskind. Automatic differentiation in machine learning: a survey. Journal of Machine Learning Research, 18 (2017), article 153.