Physics guided neural networks with application to financial modeling

Bartłomiej Małkus

PhD candidate @ Jagiellonian University

24.11.2022

Outline

- Physics-based, data-driven and hybrid modeling
- Neural networks with embedded physics
- Approaches to embed physics in neural networks
- Example physics which may be solved with PGNN
- Financial applications

Physics-based, data-driven and hybrid modeling

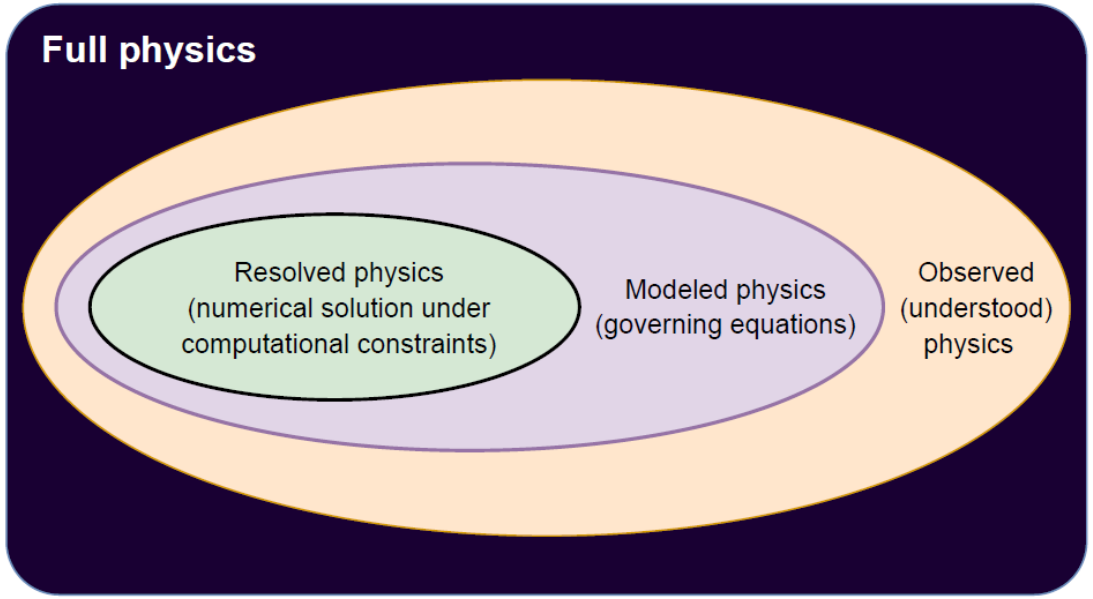
Physics-based modeling

Pros:

- Based on theory describing given phenomenon
- Interpretable
- Rather easy to analyze properties like stability and uncertainty

Cons:

- Uses only known and understood physics
- Often simplification and assumptions are made which may not hold
- Can be computationally ineffective for more complex problems



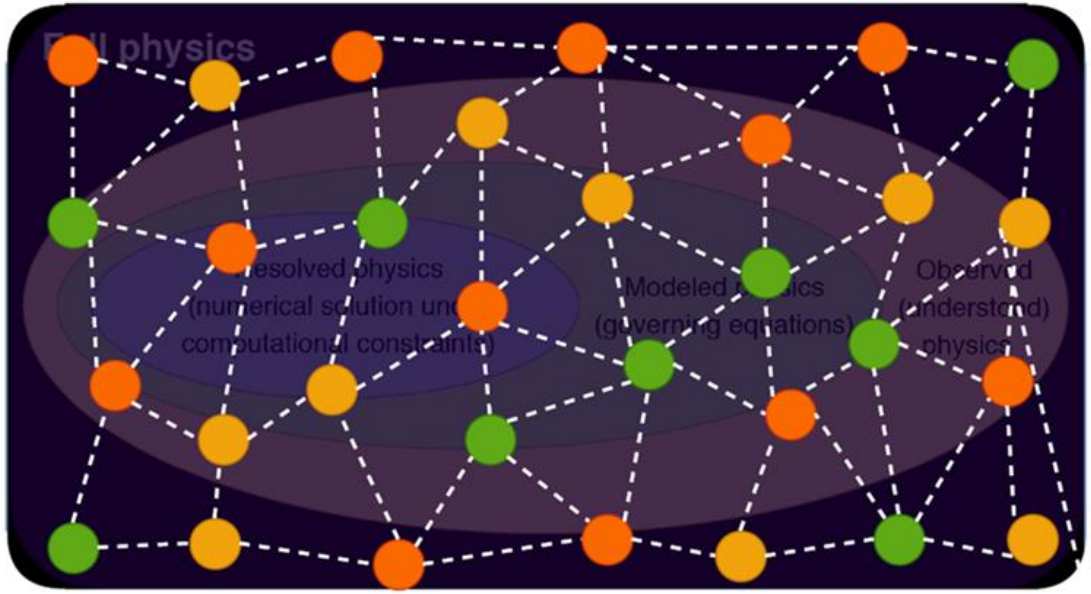
Data-driven modeling

Pros:

- Can model both known and unknown physics without prior knowledge of physical laws
- Good performance even on complex problem
- Machine learning outperforms conventional approaches in number of fields

Cons:

- Requires (lots of) data
- May not conform to known physics
- Rather hard to analyze properties like stability and uncertainty
- Harder to provide explanations



Hybrid modeling

The idea is to mix both approaches somehow. For example:

- Mixture of models – different parts of the problems are modeled using different approaches
- „Committee" models – modeling using both PB and DD models and combining the results (e. g. by averaging)
- Residual modeling – first PBM is used, and errors of such approach are modeled with the use of DDM
- Embedding known physics into data driven models

Neural networks with embedded physics

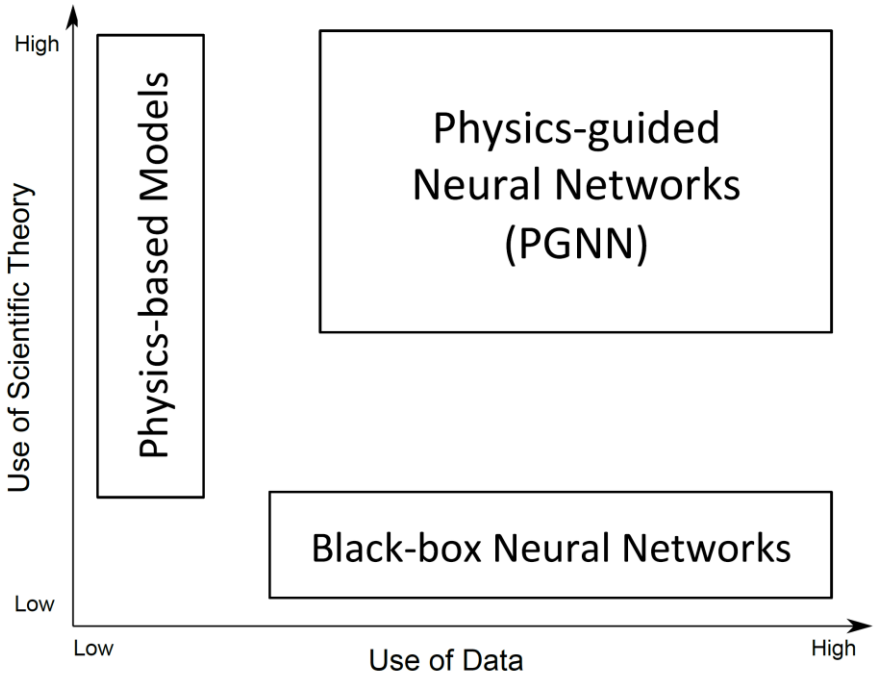
Physics guided neural networks

General idea is to make neural network „aware" of known physics

- physics may be „baked" into neural network
- it may be used as a soft constraint
- it may be used only to „guide" NN towards better solutions

In the literature terminology still seems to be a bit blurry. In some places term *physics-guided* appears interchangeably with *physics-informed* and *physics-encoded*, in others they are distinguished.

More about it after we introduce methods of embedding physics inside networks.



Objectives of embedding physics inside NN

- Improving predictions quality beyond that of PB models and DD models
  - Accounting for unknown physics/simplifications made in known physics
  - Improving data-efficiency
  - Improving generalization
  - Asserting scientific correctness
  - In finances, there is a lot of unknown physics and noise in the data + data is limited
- Parametrization of physics models
  - PB models are used to model processes, but they have to be calibrated
  - Calibration with the use of ML is computationally more efficient than traditional methods
  - In most cases black-box ML models are used to calibrate, it can be improved by embedding physics in them
  - In finances, models often have to be calibrated to market situation periodically or continuously

Objectives of embedding physics inside NN

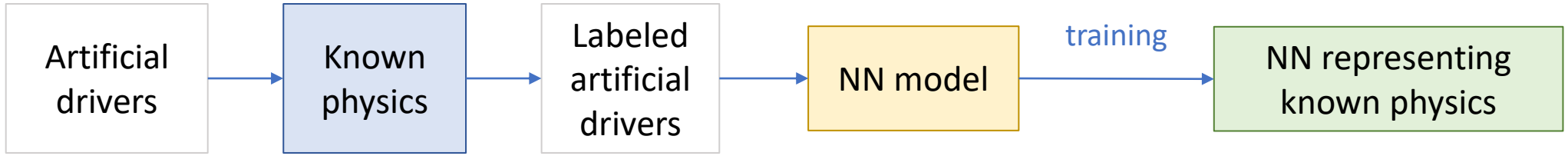
- Downscaling
  - predict finer-resolution variables from coarser-resolution variables
  - example: cloud-resolving models need to be run at sub-kilometer resolution to be effective, but it is not feasible for global climate models
- Reduced-Order Models
  - computationally inexpensive representations of more complex models
  - dimensionality reduction - projection of input space to lower-dimensional subspace
  - „controlled loss of accuracy”
  - ML-aided construction of ROMs

Objectives of embedding physics inside NN

- Improving data generation capabilities
  - PB approaches are often computationally inefficient (and are limited to known physics)
  - generative ML models, like GANs or VAEs are feasible alternative
  - embedding physics in NN approaches can (and in some cases it already does) improve efficiency
- Efficiently Solving Partial Differential Equations
  - solving them using NN greatly improves computational efficiency
  - embedding physics may increase accuracy, impose physical invariants and increase trening effectiveness

Approaches to embedding physics inside NN

Training a NN to represent known physics

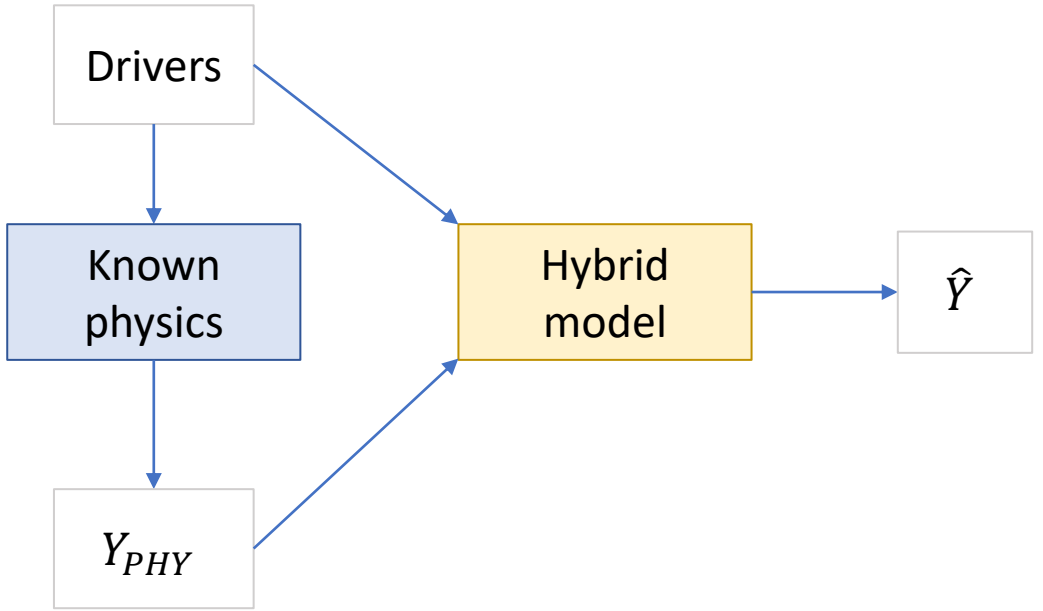


The oldest approach to „embed" physics inside neural network.

NN is just resembling the original physical model.

Used mostly to improve computational efficiency.

Hybrid physics-ML model



One of the simplest ways of incorporating physics in NN.

Does not impose any constraint on NN, there is no control on whether outputs will be physically correct. In some cases, NN may even ignore known physics part.

May be useful when known physics explain only small portion of data variance.

Physics-guided loss function

Training loss

Regularization term

$$Loss = Loss_{TRN}(Y, \hat{Y}) - \lambda R(W)$$

$$Loss = Loss_{TRN}(Y, \hat{Y}) - \lambda_R R(W) + \lambda_{PHY} Loss_{PHY}(\hat{Y})$$

Physics loss
Quantifies model inaccuracy
w. r. to known physics

Depending on λ_{PHY} may be effectively a soft or hard constraint.

Useful especially when there is hard physical constrain on the process.

Physics-guided loss function

Physical relationship between variable Y and other variables $\boldsymbol{Z}$ may be written using following equations ($\mathcal{G}$ and $\mathcal{H}$ are generic forms of physics equations):

$$\mathcal{G}(Y, \boldsymbol{Z}) = 0$$

$$\mathcal{H}(Y, \boldsymbol{Z}) \leq 0$$

$Loss_{PHY}(\hat{Y})$ can then take the following form:

$$Loss_{PHY}(\hat{Y}) = \left\| \mathcal{G}\left(\hat{Y}, \boldsymbol{Z}\right) \right\|^2 + ReLU\left(\mathcal{H}\left(\hat{Y}, \boldsymbol{Z}\right)\right)$$

Physics-guided architecture

The problem with previous approaches is that even though some physics is enforced in NN, or it's guided towards physics, the NN itself is still a black box.

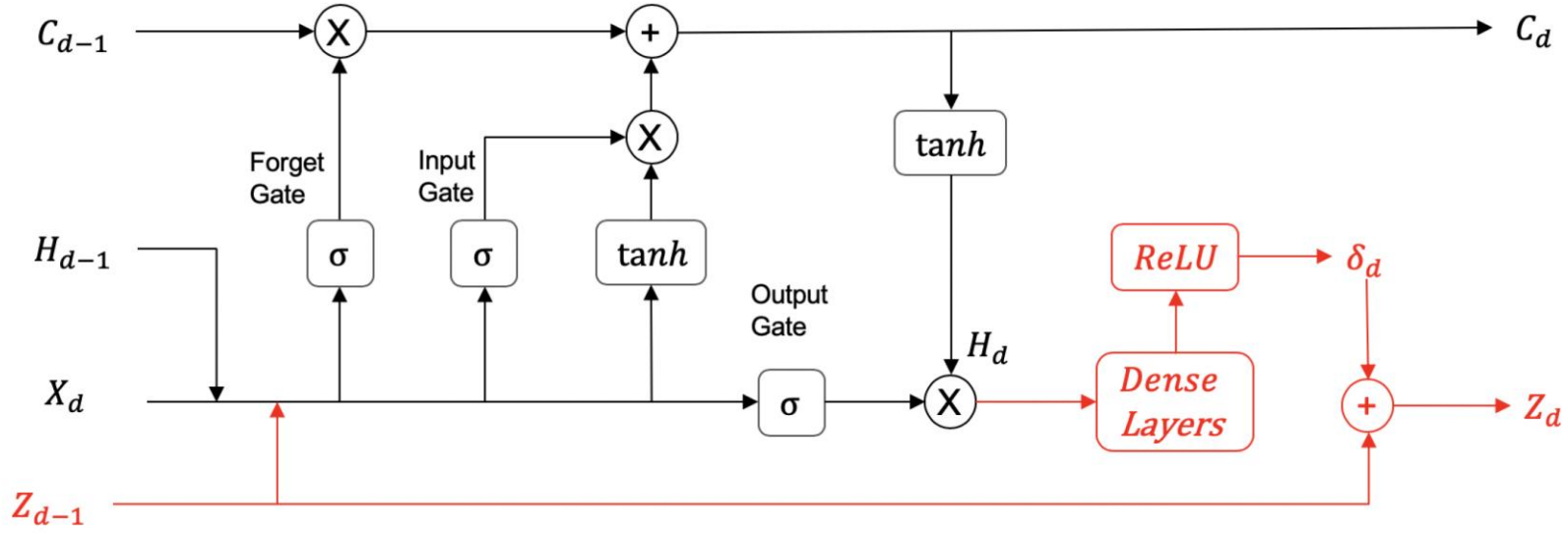
The idea is to change the architecture of NN in such a way that it's making use of the specifics of the problem solved.

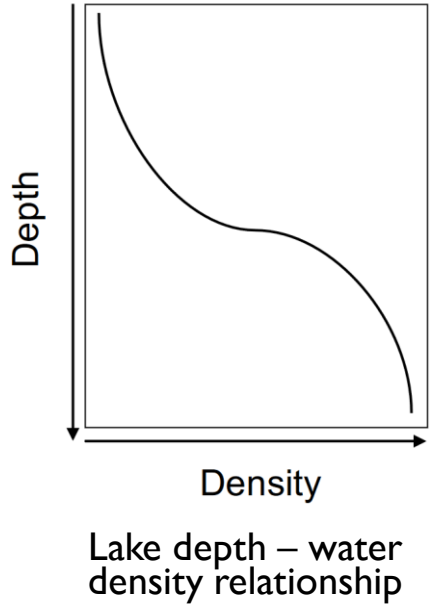
With such approach NN is truly aware of the physics (at least a part of it).

Some hard constraints can be imposed such way.

Physics-guided architecture - example

Lake temperature modeling: extension of classic LSTM cell.



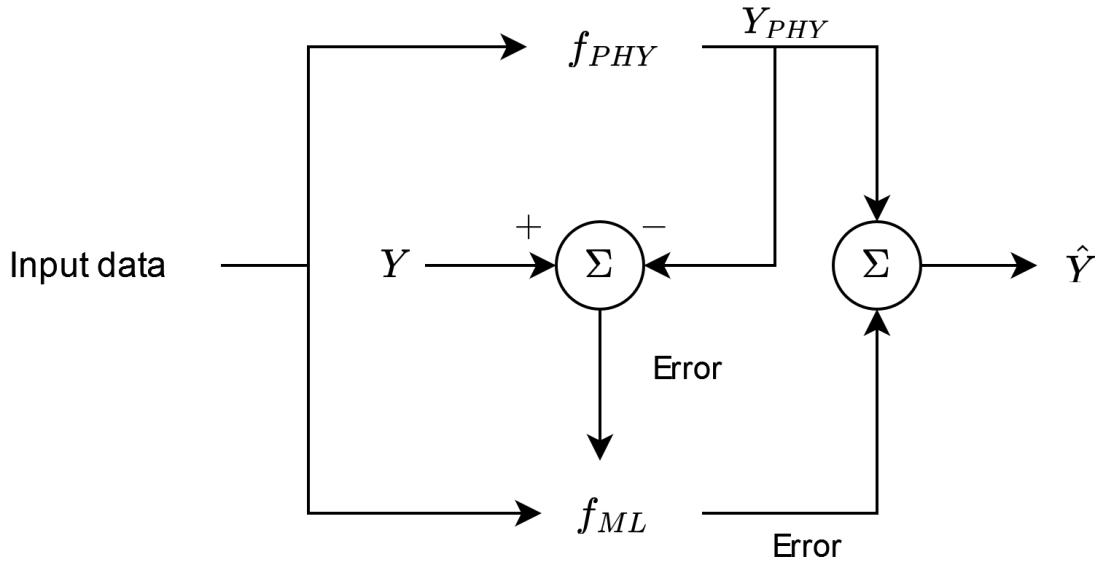


Lake depth – water density relationship

Monotonicity-preserving LSTM Architecture. Components in red represent physics-informed innovations in LSTM. Z_d represents the density values at a specific depth d, δ_d is the increase of density from layer $d-1$ to d. The value of δ is gauranteed to be non-negative as it is generated from an ReLU layer. Such an architecture ensures the monotonic increase of density values as depth increases.

Source: Daw, A., Thomas, R. Q., Carey, C. C., Read, J. S., Appling, A. P., & Karpatne, A. (2020). *Physics-Guided Architecture (pga) of neural networks for quantifying uncertainty in lake temperature modeling.*

Residual modeling

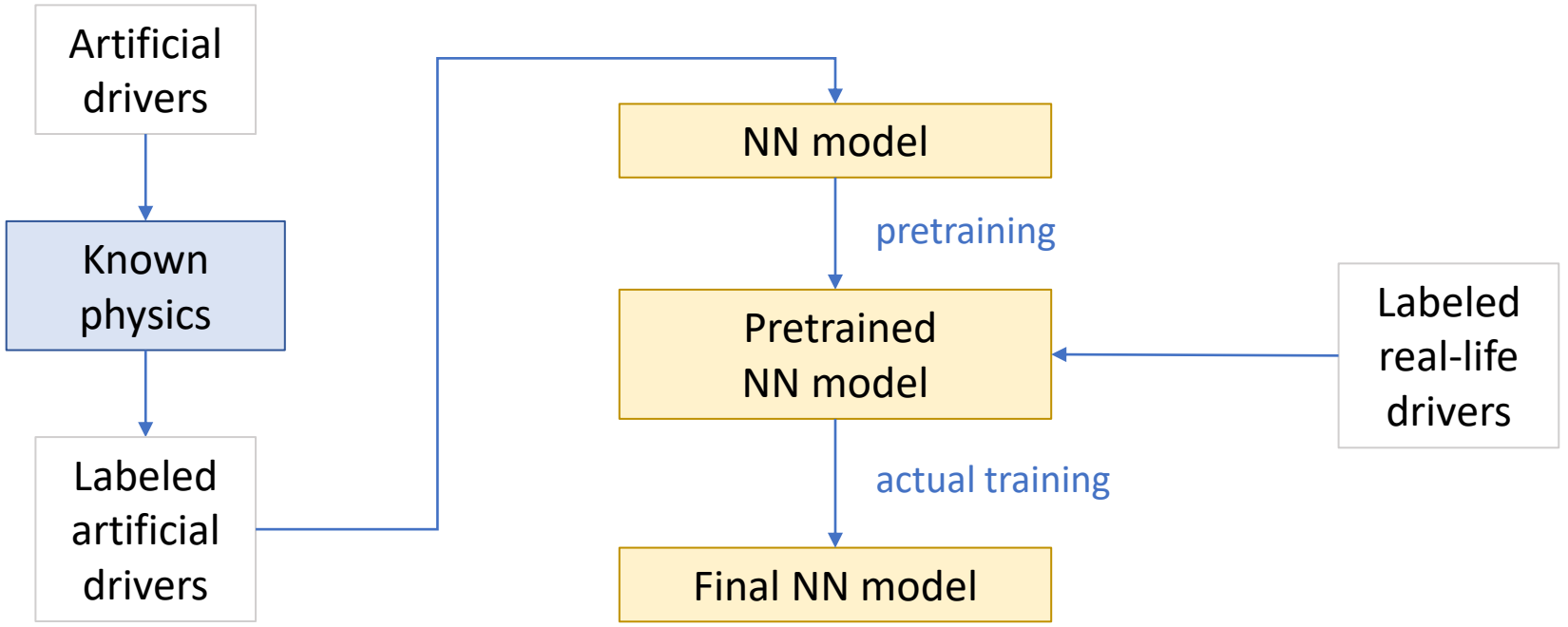


The idea here is to model only „errors" of physical models.

The most important downside is that NN is not aware of any known physics and there is no control on whether outputs will be physically correct.

Physics-guided initialization

- Variation on transfer learning
- Model is pre-trained on artificial data generated with the use of known physics
- Does not truly enforce physics, but may be used as helper for other approaches



Few words about the nomenclature

Some sources use *physics-guided* and *physics-informed* terms interchangeably (and *physics-encoded* sometimes)

Recent source proposed the following distinction:

Physics-guided neural networks – NN trained to represent known physics model, used mostly for performance purposes

Physics-informed neural networks – NN softly constrained by physics (e. g. by using physics-guided loss function)

Physics-encoded neural networks – NN hardly constrained by physics (physics is embedded in the architecture)

Example physics with application in finances

Lotka-Volterra model

It is best known for being used to describe predator-prey population relationship.

Generalized version can be used to describe relationship between two or more agents (including predator-prey, competition and mutualism relationships).

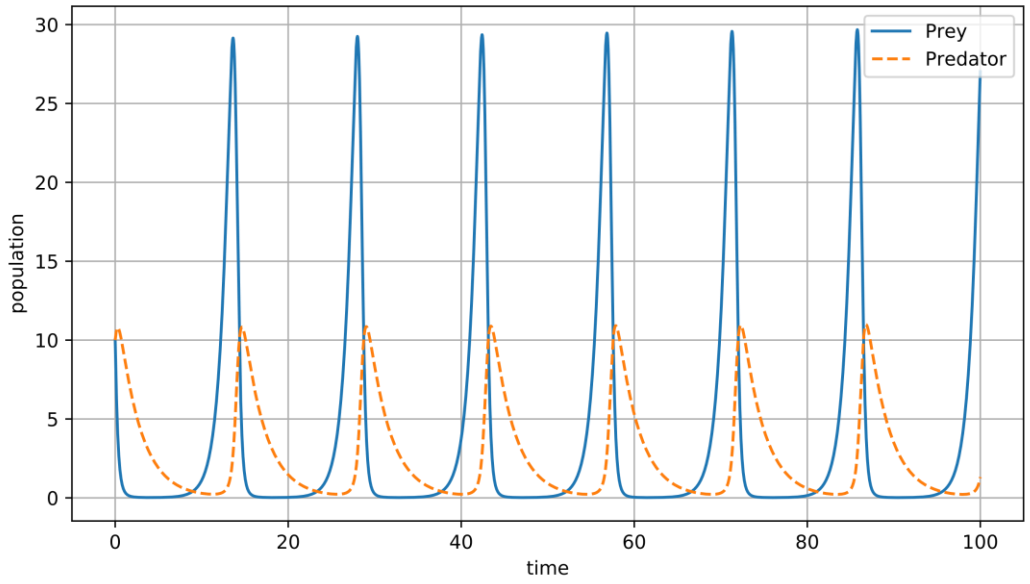
$$\frac{dx}{dt} = \alpha x + \beta yx$$
$$\frac{dy}{dt} = \gamma y + \delta xy$$

α - x growth rate (reproduction rate)

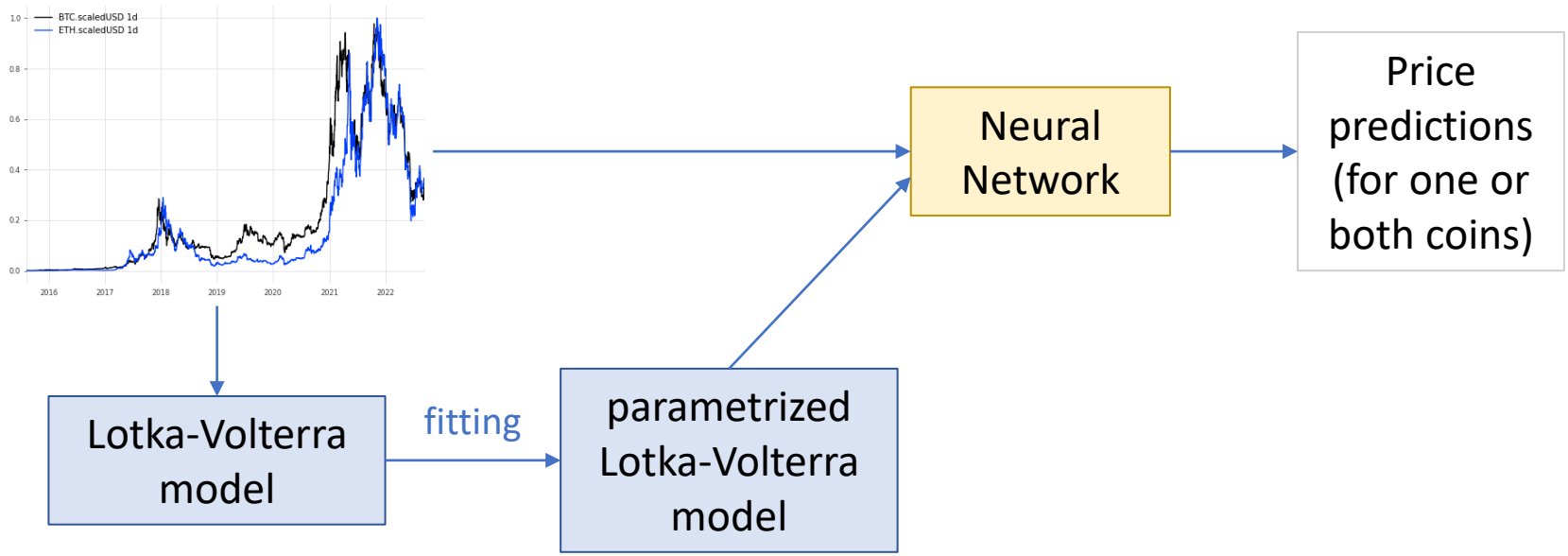
γ - y growth rate (reproduction rate)

β - influence of y on x

δ - influence of x on y



Lotka-Volterra model in crypto price modeling



Lotka-Volterra model was used to model pricing mechanisms on crypto market.

Such approach in long run did not turn out very well, in short term it was slightly better, but still not satisfactory.

Lotka-Volterra model in price mechanism modeling

What was done:

- modeling crypto prices/return rates using Lotka-Volterra model
- using such models to guide NN (only simple way of feeding model output into NN was used)

What can be done:

- Modeling other crypto parameters, like transactions count
- Modeling market participants/groups interactions (data availibility is a challenge)
- Try different ways of incorporating LV model into NN

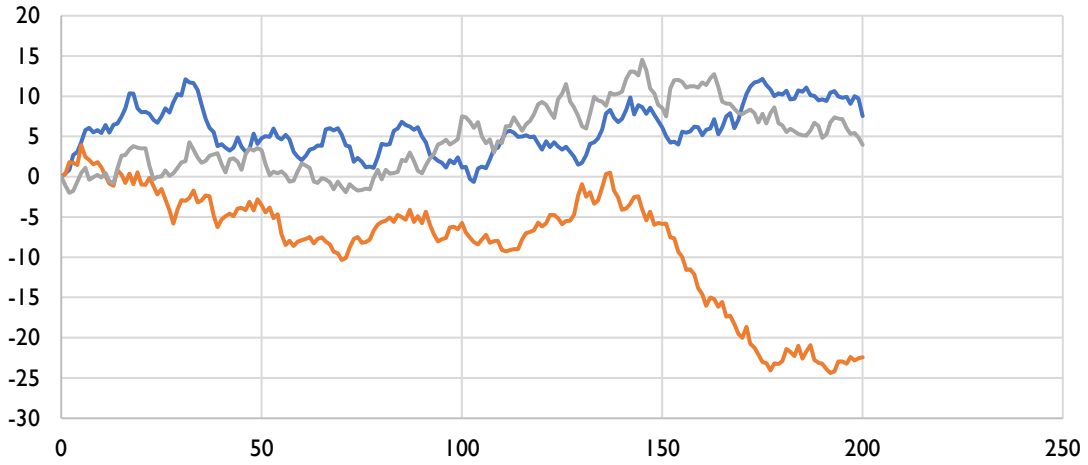
Another example application: Supporting model calibration with neural networks

Wiener process

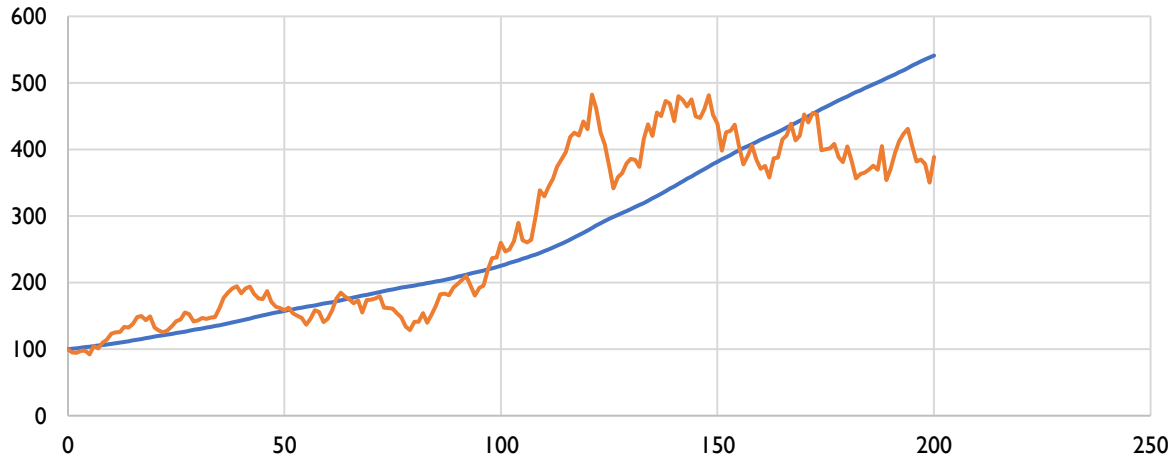
$$dS_t = \mu S_t dt + \sigma S_t dW_t$$

long-term trend

random fluctuations



Example implementations of the Wiener process



Example implementation of the geometric Brownian motion

Supporting model calibration with neural networks

As an example, let's take Heston model, Black-Scholes model extension. There are multiple parameters to calibrate.

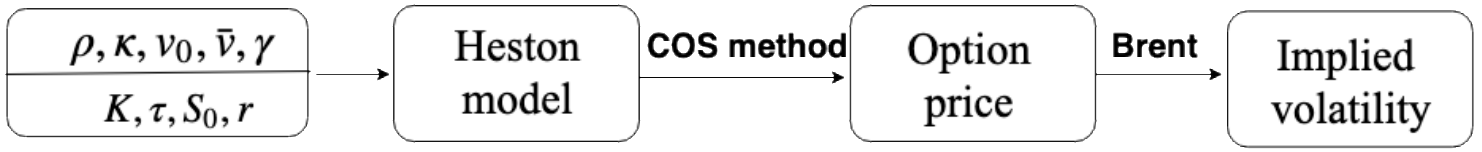
$$dS_t = \mu S_t dt + \sqrt{\nu_t} S_t dW_t^S + \left(e^{\alpha+\delta\epsilon} - 1\right) S_t dq, \qquad S_{t_0} = S_0$$

$$d\nu_t = \kappa(\bar{\nu} - \nu_t)dt + \gamma\sqrt{\nu_t} dW_t^\nu, \qquad \nu_{t_0} = \nu_0$$

$$dW_t^S dW_t^\nu = \rho dt$$

Supporting model calibration with neural networks

Classic option quotation using Heston model looks as follows:



The calibration of this model is performed with the use of non-gradient heuristics such as Differential Evolution or Particle Swarm Optimization.

Two bottlenecks arise - the time needed to evaluate the model in the calibration process and the pace of the calibration itself.

Supporting model calibration with neural networks

To eliminate the first bottleneck, neural network can be used, mapping the model parameters $\{\rho, \kappa, \nu_0, \bar{\nu}, \gamma; K, \tau, S_0, r\}$ to the price of an option V. The valuation process will be as follows:

Above NN are not physically constrained in any way though.

Improving on the second bottleneck is an open problem.

Source: Liu, S., Borovykh, A., Grzelak, L. A., & Oosterlee, C. W. (2019). *A neural network-based framework for financial model calibration.*

What else can they be used for?

- Markets synchronization (e. g. dependencies between stock, bond and bullion markets)
- Contagion effect on the markets (e. g. spread of crisis from one market/region to another)
- Data generation
- Model calibration
- Price discovery mechanisms

Thank you for your attention

Sources

Willard, J., Jia, X., Xu, S., Steinbach, M., & Kumar, V. (2020). Integrating physics-based modeling with machine learning: A survey. *arXiv preprint arXiv:2003.04919*, *1*(1), 1-34.

Robinson, H., Pawar, S., Rasheed, A., and San, O. (2022). Physics guided neural networks for modelling of non-linear dynamics. *arXiv preprint arXiv:2205.06858.*

Faroughi, S. A., Pawar, N., Fernandes, C., Das, S., Kalantari, N. K., & Mahjour, S. K. (2022). Physics-Guided, Physics-Informed, and Physics-Encoded Neural Networks in Scientific Computing. *arXiv preprint arXiv:2211.07377*.

Daw, A., Thomas, R. Q., Carey, C. C., Read, J. S., Appling, A. P., & Karpatne, A. (2020). Physics-guided architecture (pga) of neural networks for quantifying uncertainty in lake temperature modeling. In *Proceedings of the 2020 siam international conference on data mining* (pp. 532-540). Society for Industrial and Applied Mathematics.

Liu, S., Borovykh, A., Grzelak, L. A., & Oosterlee, C. W. (2019). A neural network-based framework for financial model calibration. *Journal of Mathematics in Industry*, *9*(1), 1-28.