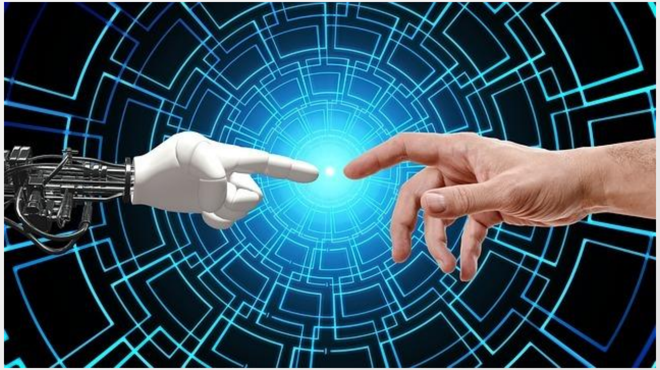
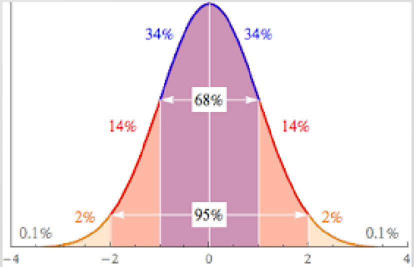
Ben-Gurion University of the Negev

Model-Based AI in Communications

Nir Shlezinger

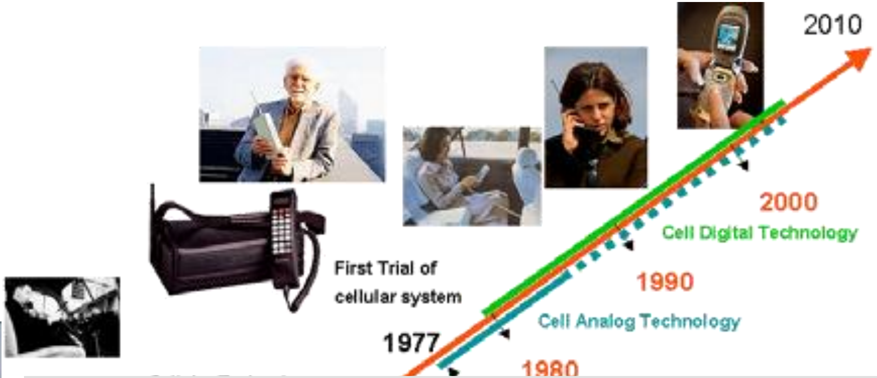
December 2022

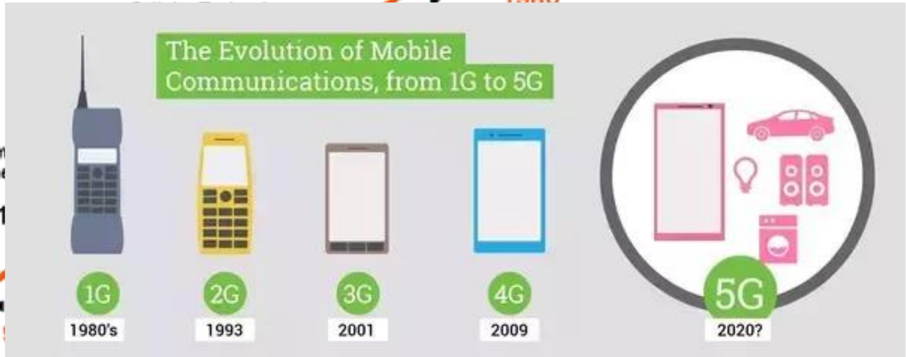




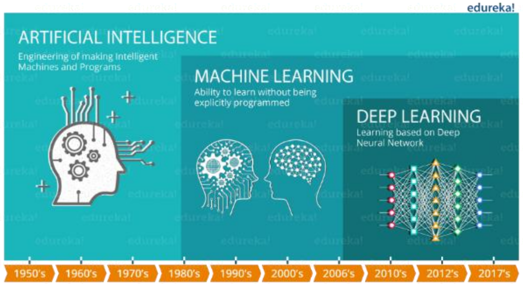
Wireless Communications

- Wireless communication era
- Major enabler technology
- Constantly increasing demands

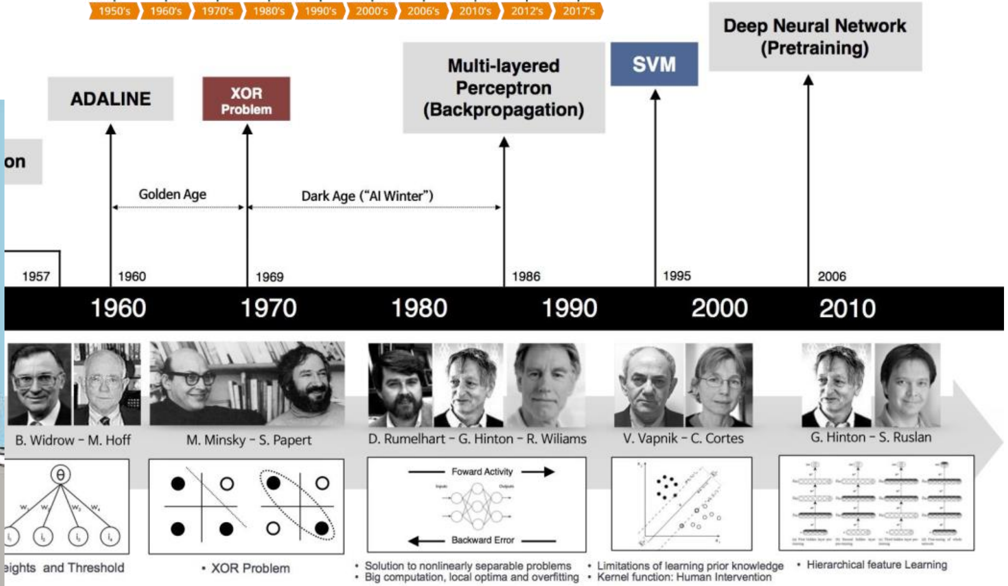


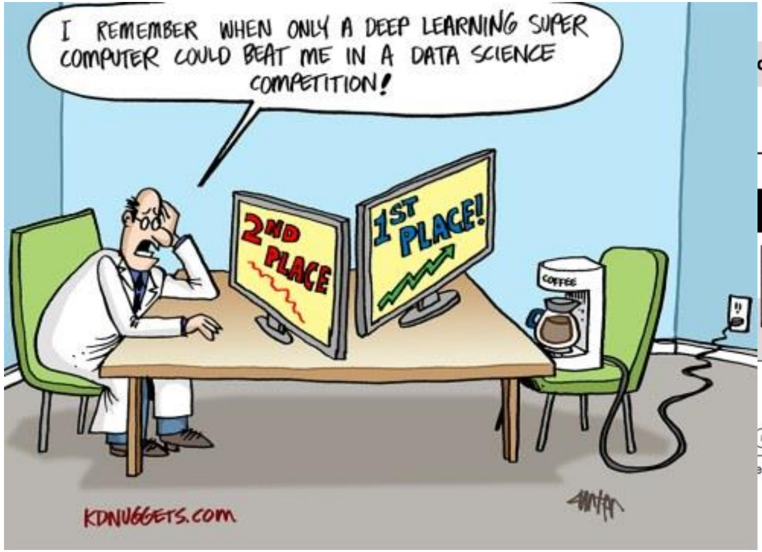


Machine Learning (ML)



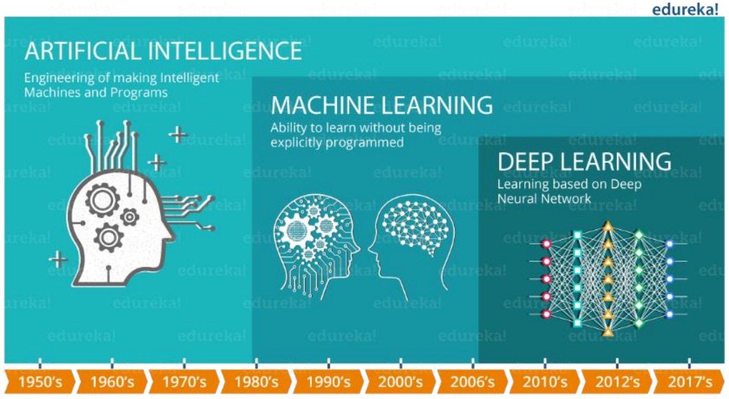
- A rapidly growing technology
- Revolutionized many fields
- Rely on data
- Unprecedent empirical success

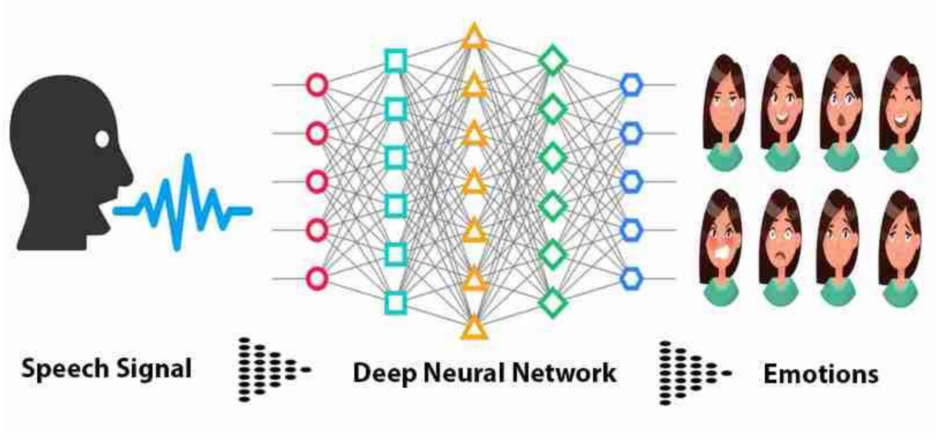


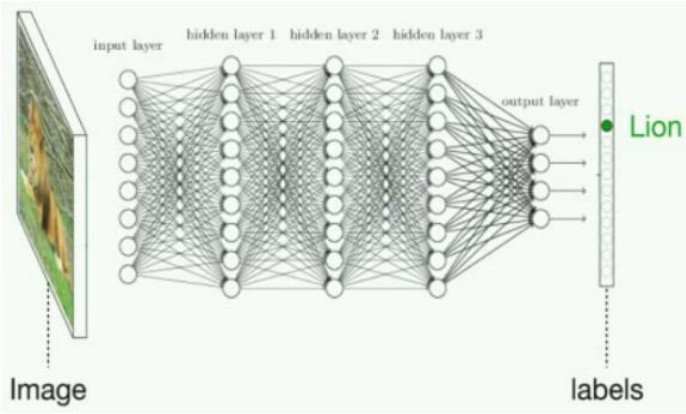


The Deep Learning Revolution

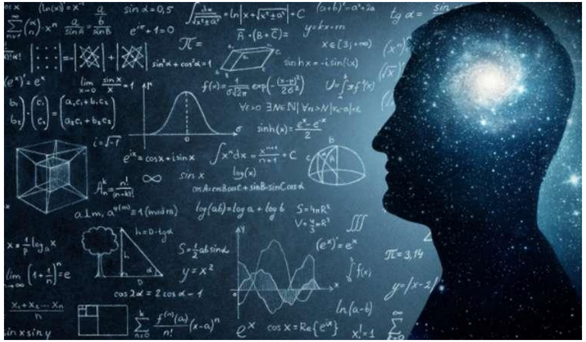
- Deep neural networks (DNNs) achieve superior performance in multiple areas:
  - Computer vision
  - Natural language processing
  - ….
  - Problems that are difficult to tackle using conventional optimization methods.





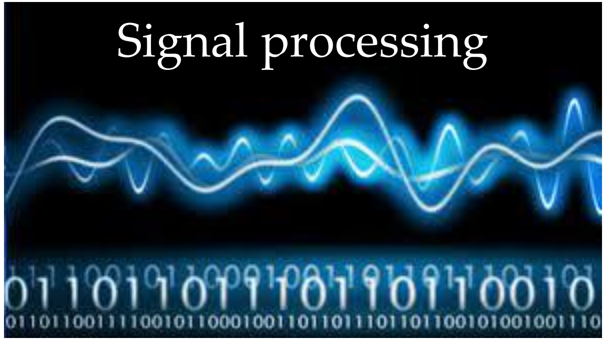


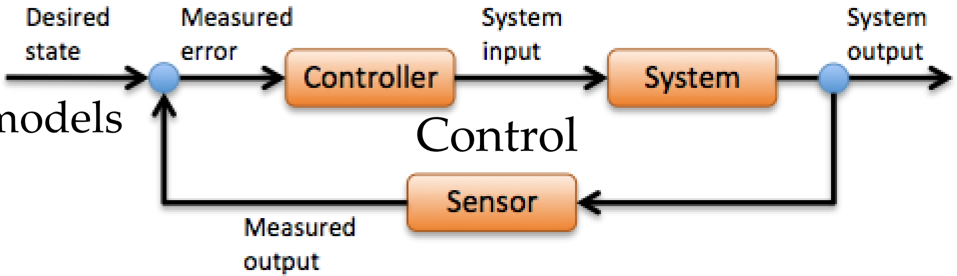
Model-Based Methods



- Classical methods rely on knowledge of principled mathematical models

- Established models arise in many applications

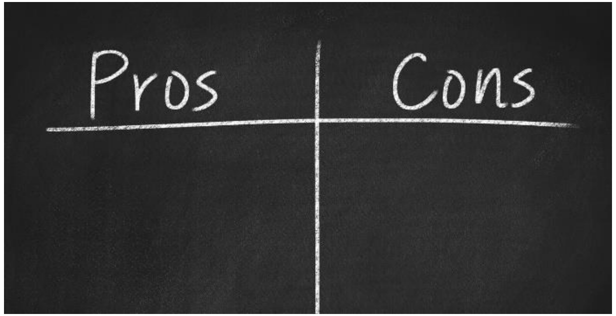




- Established algorithms designed for such models

- Sensitive to inaccuracies, complex, delay

Hybrid Methods

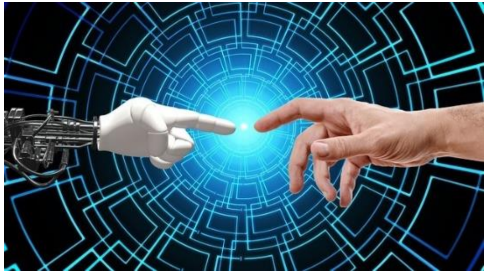


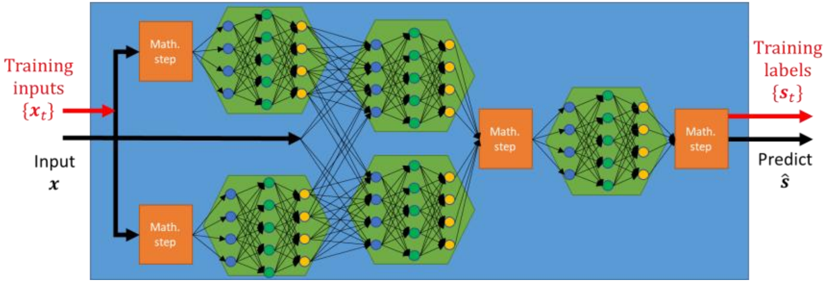
- Relying on data has its pros and cons
- Relying on knowledge has its pros and cons
- In signal processing and communications we often have both
- Hybrid techniques combining
  - Principled mathematical models
  - Data-driven systems

Model-Based Deep Learning

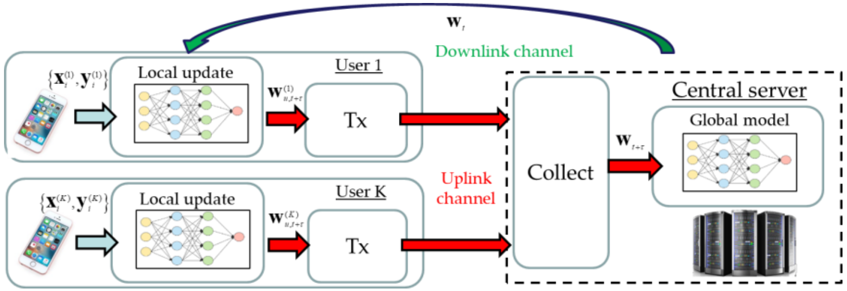
Talk Outline

- Overview and motivation
- What is model-based deep learning?
- (Only) two representative examples
  - KalmanNet
  - Unfolding for hyperparameter optimization
- The elephant in the room....

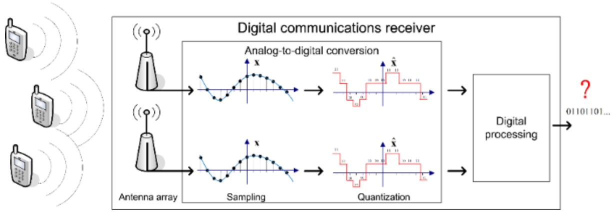


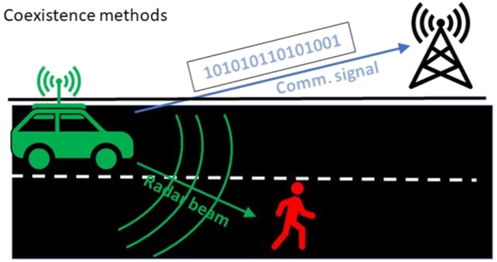


About Us



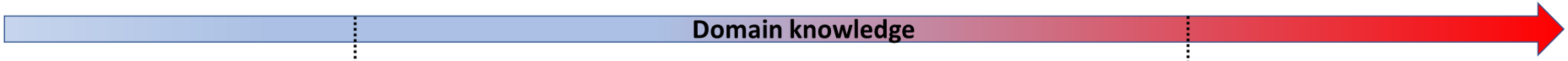
- PI: Nir Shlezinger
  - Research areas:
    - **Model-based AI**
    - Deep learning for communications
    - Learning over communication channels
    - Metasurfaces for communication
    - **Learning on the edge**
    - Joint radar-comm. systems
    - Analog-to-digital conversion

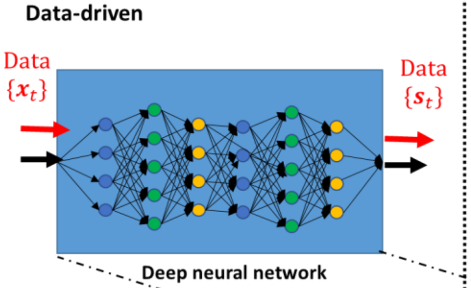




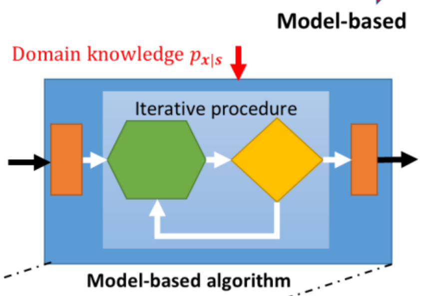
What is model-based deep learning?

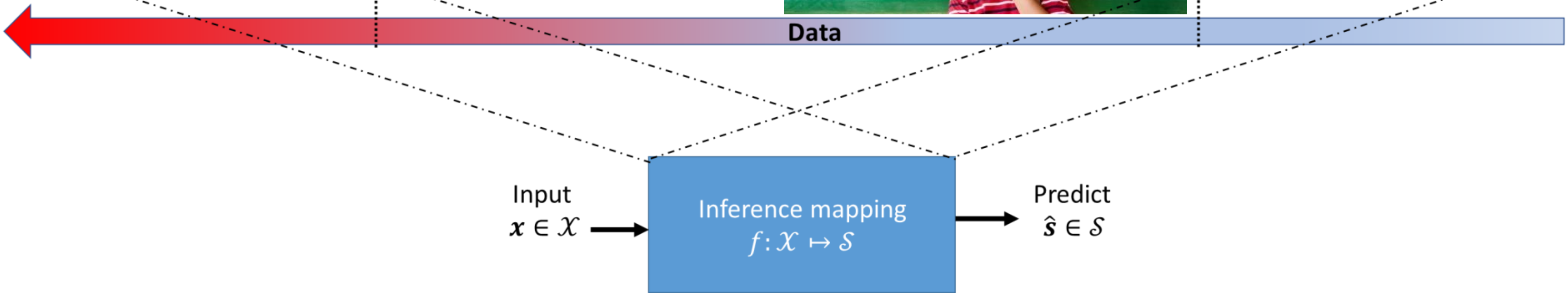
Is There a Middle Ground?





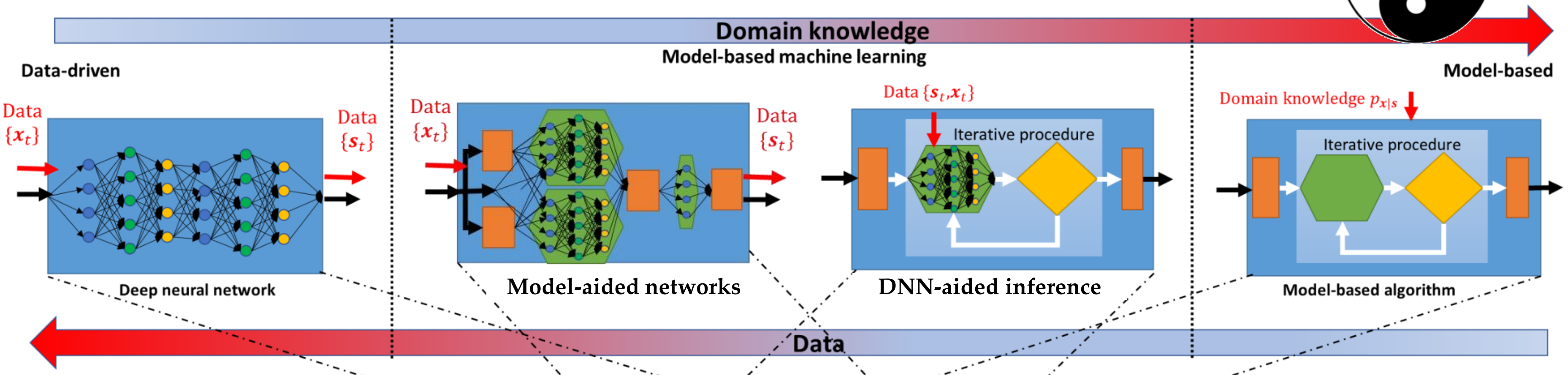
- Benefit from (partial) domain knowledge?
- Exploit (limited) data?
- How to combine?



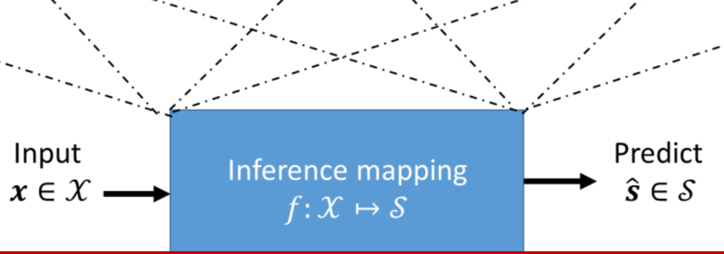


There is a Middle Ground!

Best Of Both Worlds



- Infer using a deep network
- Integrate model-based methods in architecture



- Infer using a model-based algorithm
- Augment algorithm with deep neural networks

Recent reviews:

N. Shlezinger, J. Whang, Y. C. Eldar, and A. G. Dimakis. “Model-Based Deep Learning." *arXiv 2012.08405*

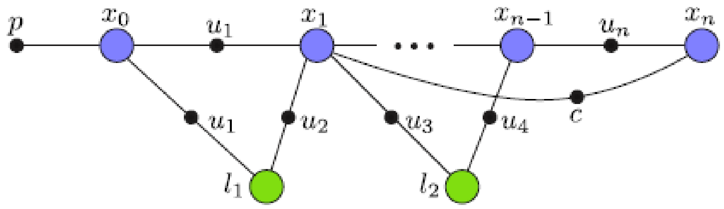
N. Shlezinger, Y. C. Eldar, and S. P. Boyd. “Model-Based Deep Learning: On the Intersection of Deep Learning and Optimization”, *IEEE Access ,2022*

(Only) two representative examples

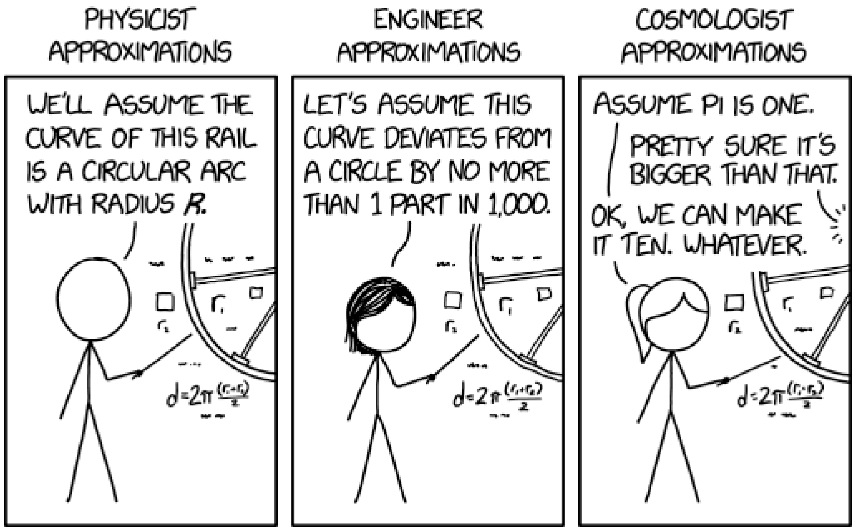
Structure-Oriented DNN-Aided Inference

Structure-oriented

- Often, we have knowledge of statistical structures
  - Physical characteristics
  - System operation
  - Understanding of underlying dynamics
  - Established approximations
- The model subtleties may be intractable…
- Structures are exploited by algorithms



- Learn the missing model subtleties



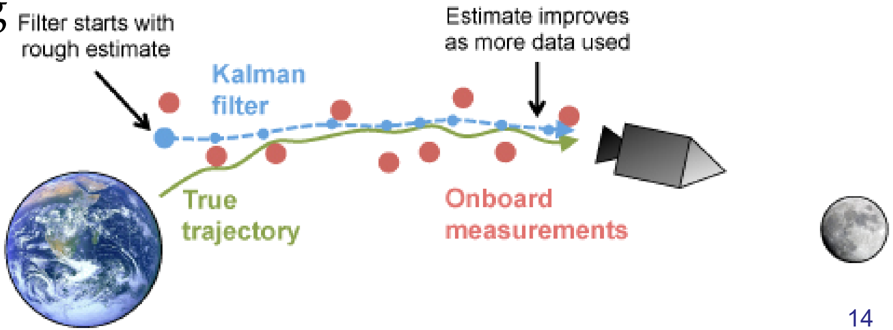
DNN-Aided Kalman Filter (1/5)

Real-time tracking of dynamic systems

- Observations: $\boldsymbol{x}_t \in \mathcal{R}^N$
- Desired state: $\boldsymbol{s}_t \in \mathcal{R}^K$

Input $\boldsymbol{x}_t \in \mathcal{R}^N$ → Inference mapping $f_t\colon \mathcal{R}^N \mapsto \mathcal{R}^K$ → Predict $\hat{\boldsymbol{s}}_t \in \mathcal{R}^K$

- State-space model:
  - State evolution model: $\boldsymbol{s}_t = \boldsymbol{f}(\boldsymbol{s}_{t-1}) + \boldsymbol{e}_t$
  - Observations model: $\boldsymbol{x}_t = \boldsymbol{h}(\boldsymbol{s}_t) + \boldsymbol{v}_t$
- Example applications: localization, tracking



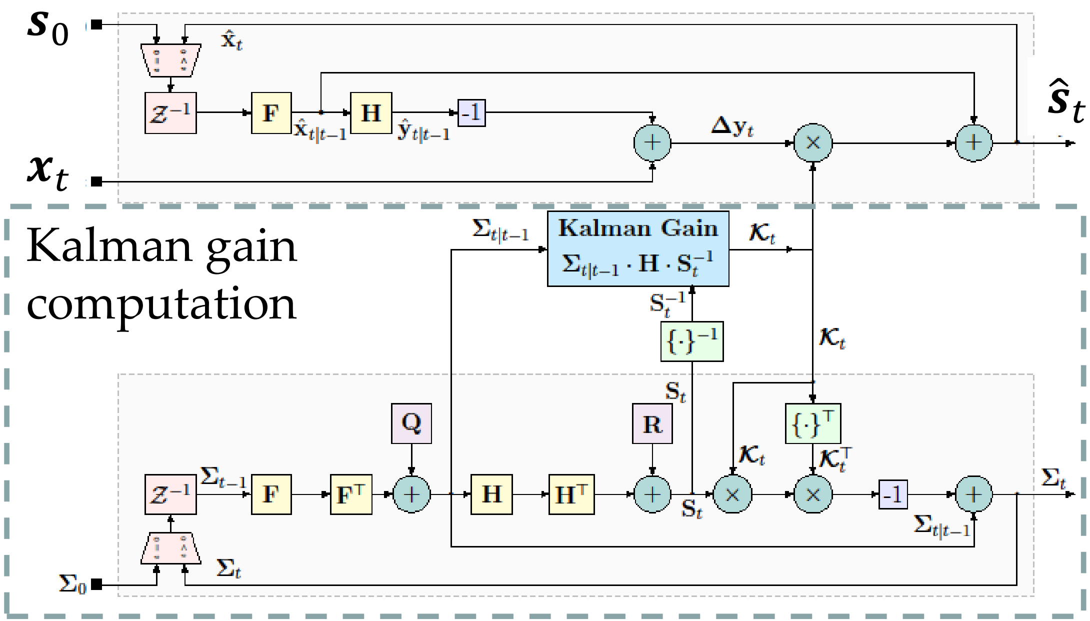
Classic Kalman Filter (2/5)

<u>Kalman Filter</u>: Linear Gaussian state-space model

$$s_t = F s_{t-1} + e_t \; (\sim \mathcal{N}(0, Q))$$

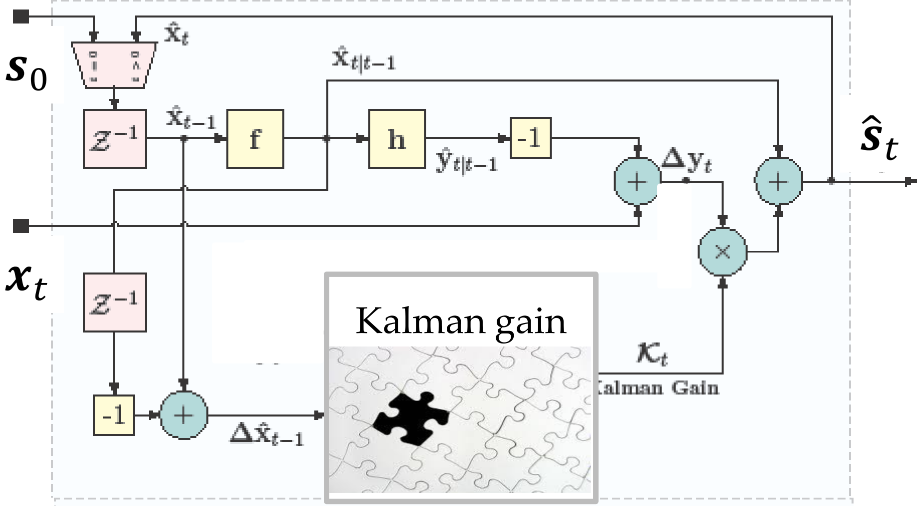
$$x_t = H s_t + v_t \; (\sim \mathcal{N}(0, R))$$

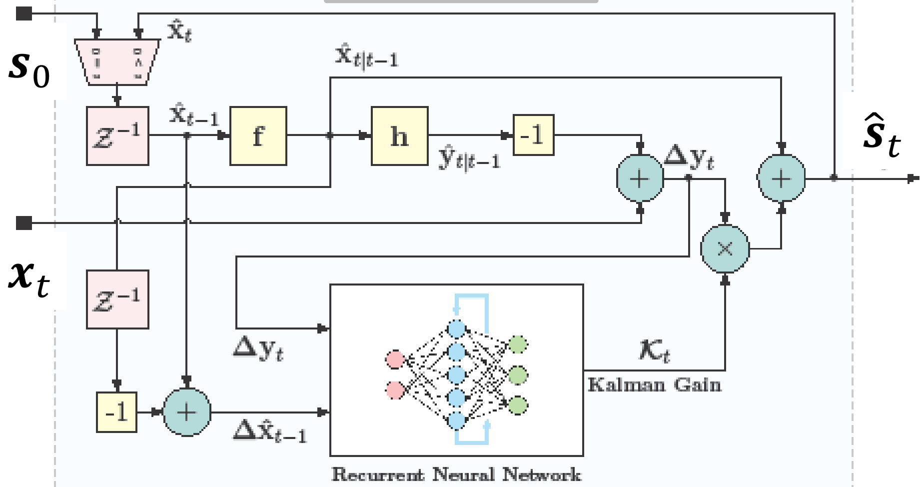
- Provides minimal MSE estimate of s_t given $\{x_\tau\}_{\tau=0}^{t}$
  - From current observation x_t
  - Previous estimate $\hat{s}_{t-1}$



KalmanNet (3/5)

- Kalman filter (and its variants) require knowledge of the state-space model
- Noise statistics dictate the Kalman gain
- In practice we are likely to only have an approximation of the state-space
- Learn the Kalman gain from data
  - Not enforcing a model on the noise
  - Learn to handle mismatches
  - Preserve Kalman operation





- G. Revach, N. Shlezinger, X. Ni, A. L. Escoriza, R. J. G. van Sloun, and Y. C. Eldar, "KalmanNet: Neural Network Aided Kalman Filtering for Partially Known Dynamics", *IEEE Transactions on Signal Processing*, 2022.

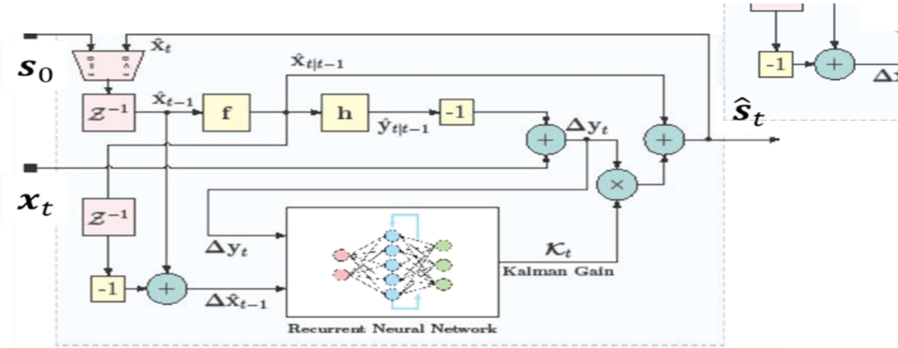
KalmanNet (4/5)

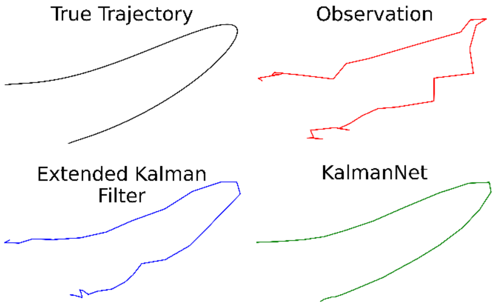
Model	MB KS	Benchmark [19]	RTSNet
MSE [dB]	-10.071	-15.346	-15.56
Inference time [sec]	9.93	30.5	5.007
Training time [hours/epoch]	N/A	0.4	0.16
Number of trainable parameters	N/A	41, 236	33, 270

- (Extended) Kalman filter

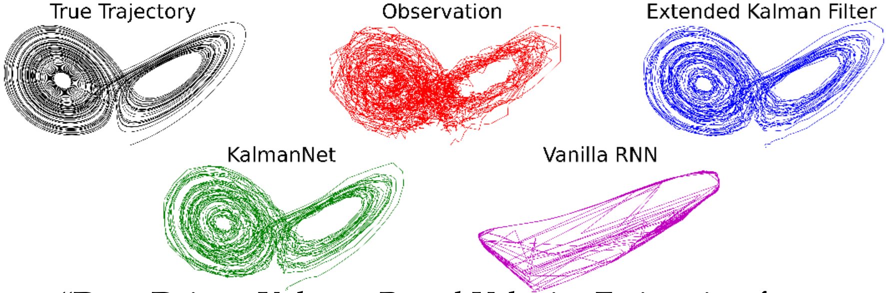
vs.

- KalmanNet





- Model-agnostic
  - Not enforcing a model on the noise
  - Learn to handle mismatches



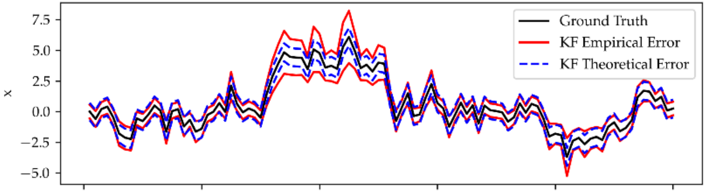
- Preserve Kalman real-time operation

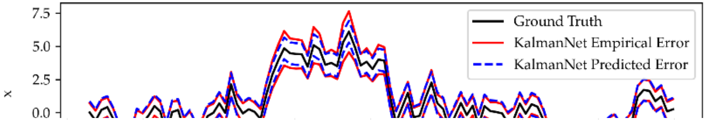
- A. L. Escoriza, G. Revach , N. Shlezinger, and R. J. G. van Sloun. "Data-Driven Kalman-Based Velocity Estimation for Autonomous Racing", *IEEE ICAS 2021*

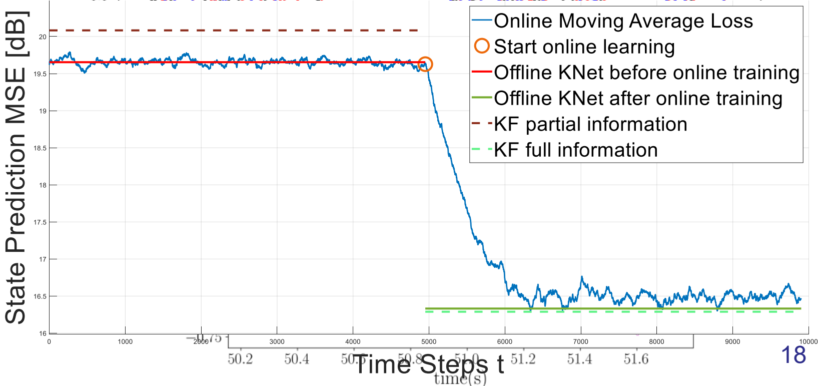
Gains of Model-Based Design (5/5)

› What do we gain from being interpretable?

1. Less complex network architectures
   - Applicable in real-time on limited hardware
2. Measure of *uncertainty*
   - Holds also under mismatched models
3. Can be trained in an unsupervised manner
   - Track variations in the state-space model
4. Extends naturally to other Kalman-type tasks



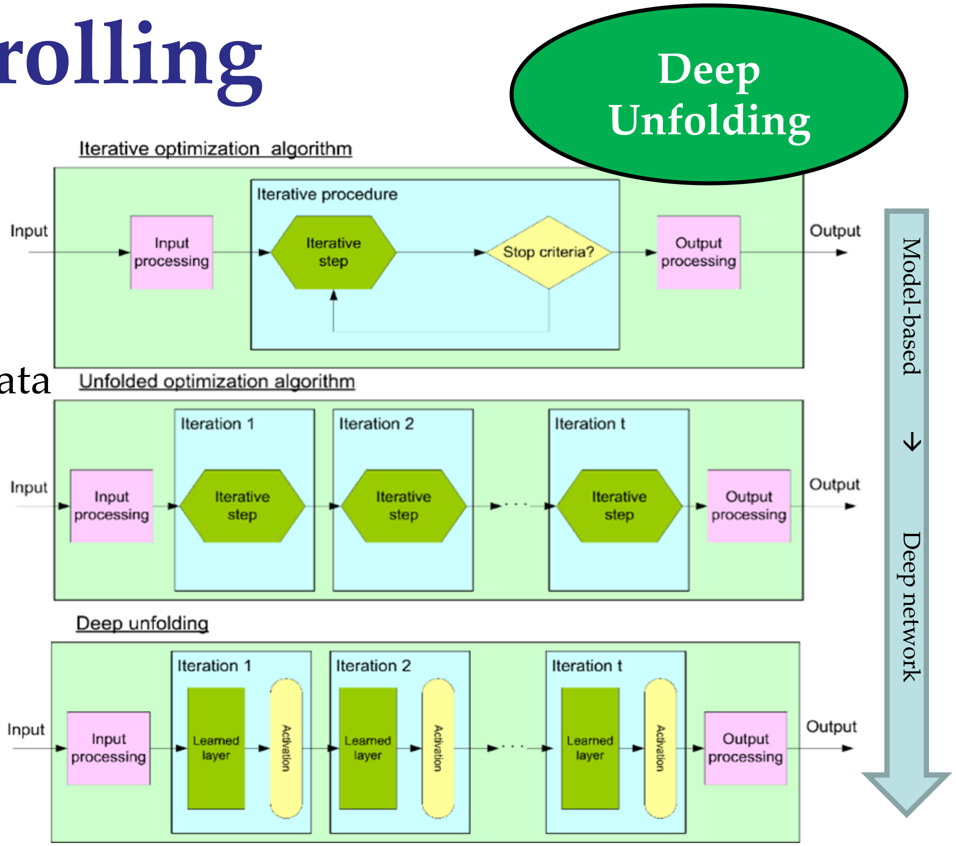




- A. L. Escoriza, G. Revach , N. Shlezinger, and R. J. G. van Sloun. “Data-Driven Kalman-Based Velocity Estimation for Autonomous Racing”, *IEEE ICAS 2021*
- I. Klein, G. Revach, N. Shlezinger, J. Mehr, R. J. G. van Sloun, and Y. C. Eldar, “ Uncertainty in Data-Driven Kalman Filtering for Partially Known State-Space Models”, *IEEE ICASSP 2022*
- G. Revach, N. Shlezinger, T. Locher, X. Ni, R. J. G. van Sloun, and Y. C. Eldar, “Unsupervised Learned Kalman Filtering”, *EUSIPCO 2022*
- X. Ni, G. Revach, N. Shlezinger, R. J. G. van Sloun, and Y. C. Eldar, “RTSNet: Deep Learning Aided Kalman Smoothing”, *IEEE ICASSP 2022*

Deep Unfolding/Unrolling

- Deep networks inspired by iterative model-based algorithm:
  - Unfold iterations into layers
  - Learn parameters of the layer from data
  - Model-driven network
- Benefits:
  - Faster convergence
  - Less trainable parameters
  - Interpretable network
  - Better performance, less training



LISTA for Sparse Recovery

- Sparse recovery via ℓ_1 relaxation

$$\boxed{\hat{\boldsymbol{s}}^* = \underset{\boldsymbol{s}}{\operatorname{argmin}} \|\boldsymbol{x} - \mathbf{H\Psi s}\|_F^2 + \rho \|\boldsymbol{s}\|_1}$$

- Solve by ISTA

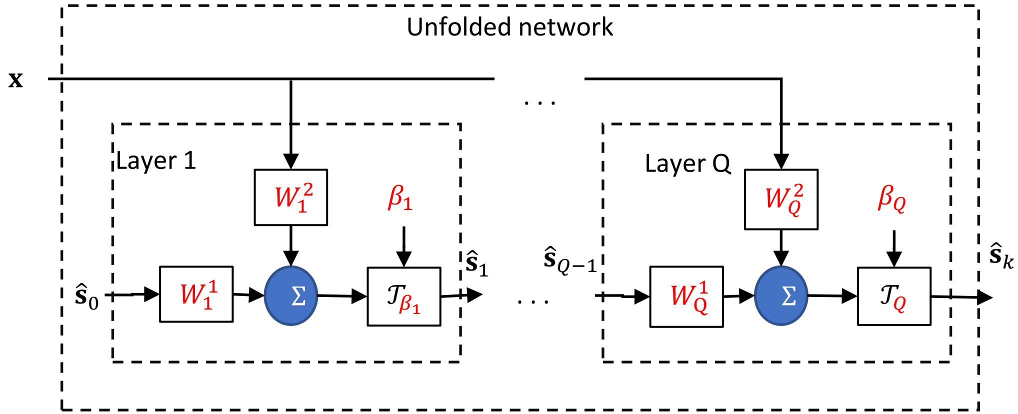
$$\boldsymbol{s}_{t+1} = \mathcal{T}_{\eta_t \rho}(\boldsymbol{s}_t - \eta_t \boldsymbol{\Psi}^T \boldsymbol{H}^T (\boldsymbol{H\Psi s}_t - \boldsymbol{x}))$$

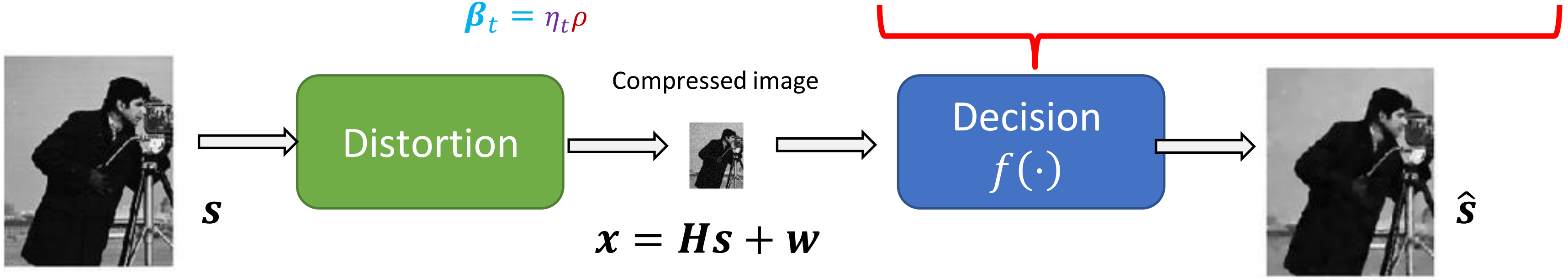
- Unfold Q iterations

$$\boldsymbol{s}_{t+1} = \mathcal{T}_{\boldsymbol{\beta}_t}(\boldsymbol{W}_t^1 \boldsymbol{s}_t + \boldsymbol{W}_e \boldsymbol{x}))$$

- Coincides when

$$\boldsymbol{W}_t^1 = \boldsymbol{I} - \eta_t \boldsymbol{\Psi}^T \boldsymbol{H}^T \boldsymbol{H\Psi}$$
$$\boldsymbol{W}_t^2 = \eta_t \boldsymbol{\Psi}^T \boldsymbol{H}^T$$
$$\boldsymbol{\beta}_t = \eta_t \rho$$





- K. Gregor and Y. LeCun. "Learning Fast Approximations of Sparse Coding." in *ICML* 2010

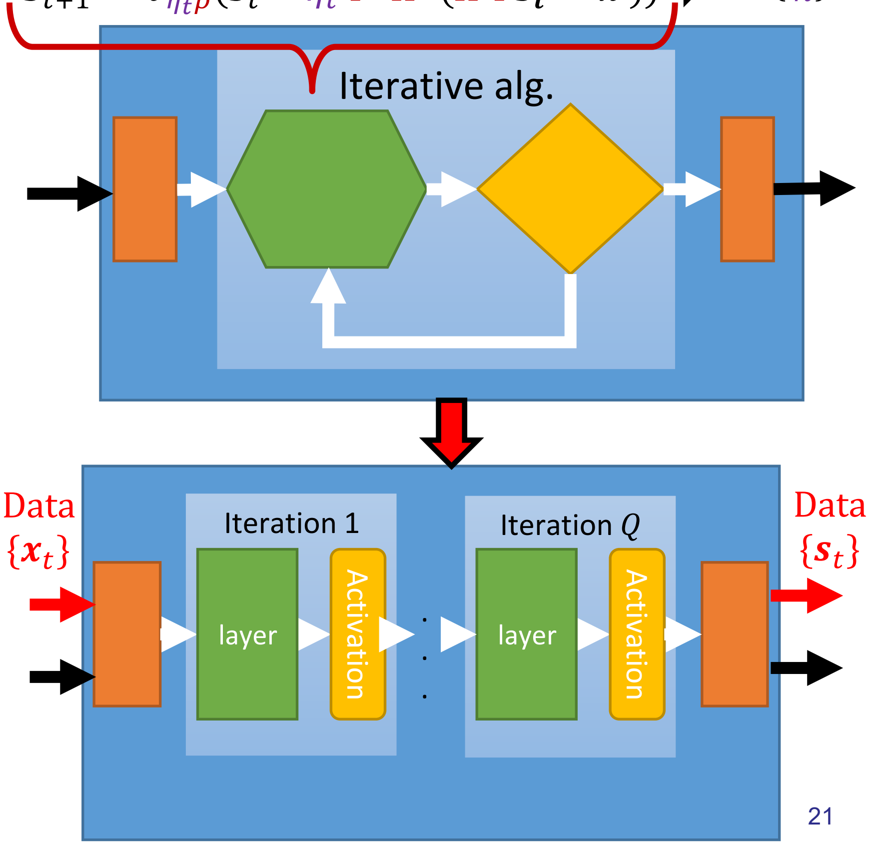
Deep Unrolling Options

- Iterative algorithms are also parameterized
  - Objective parameters ψ
  - Hyperparameters λ

- Three unfolding options:
  - A. Improve convergence speed
    - Optimize hyperparameters λ **for each iteration**
  - B. Learn abstract model
    - Set iteration parameters for both ψ and λ
    - **Different objective**
    - Use iterative solver as principled initialization
  - C. Convert into abstract deep network
    - Design neural layers that imitate the iterations
    - **Deep architecture inspired by algorithm**

$$\mathcal{L}_{\psi}(\boldsymbol{s}) = ||\boldsymbol{x} - \mathbf{H\Psi s}||_F^2 + \rho||\boldsymbol{s}||_1 \Rightarrow \boldsymbol{\psi} = \{\boldsymbol{H}, \boldsymbol{\Psi}, \rho\}$$

$$\boldsymbol{s}_{t+1} = \mathcal{T}_{\eta_t\rho}(\boldsymbol{s}_t - \eta_t \boldsymbol{\Psi}^T \boldsymbol{H}^T (\boldsymbol{H\Psi s}_t - \boldsymbol{x})) \Rightarrow \boldsymbol{\lambda} = \{\eta_t\}$$



Learned Hybrid Beamforming (1/5)

A. Improve convergence

Hybrid analog/digital receivers

Precoder in frequency-selective channels

Input $\{\widetilde{\boldsymbol{H}}_b\}$

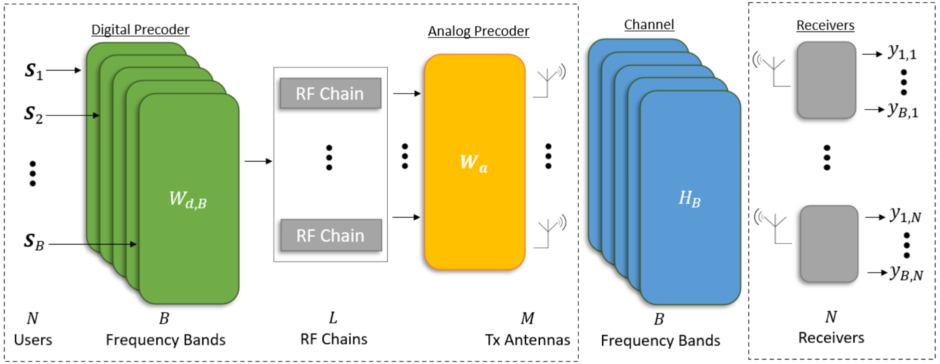
Inference mapping $f: \mathcal{R}^{B\times N\times M} \mapsto \mathcal{R}^{M\times L} \times \mathcal{R}^{B\times L\times N}$

Predict $\boldsymbol{W}_a$, $\{\boldsymbol{W}_{d,b}\}$

- Observations: channel matrix in each bin
  - $\{\widetilde{\boldsymbol{H}}_b\}$, $b \in 1, \ldots, B$
- Output: hybrid precoding setting
  - Analog precoder $\boldsymbol{W}_a \in \mathcal{C}^{M\times L}$
  - Digital precoders $\boldsymbol{W}_{d,b} \in \mathcal{C}^{L\times N}$
- Problem formulation:

$$\underset{\boldsymbol{W}_a,\{\boldsymbol{W}_{d,b}\}}{\text{argmax}}\, R = \frac{1}{B}\sum \log\left|\boldsymbol{I} + \widetilde{\boldsymbol{H}}_b \boldsymbol{W}_a \boldsymbol{W}_{d,b} \boldsymbol{W}_{d,b}^H \boldsymbol{W}_a^H \widetilde{\boldsymbol{H}}_b^H\right|$$

$$\text{subject to } \frac{1}{B}\sum \|\boldsymbol{W}_a \boldsymbol{W}_{d,b}\|_F^2 \leq N$$



Learned Hybrid Beamforming (2/5)

- Set hybrid precoder for channel $\{\tilde{\boldsymbol{H}}_b\}$

$$\underset{\boldsymbol{W}_a,\{\boldsymbol{W}_{d,b}\}}{\operatorname{argmax}} R = \frac{1}{B}\sum \log\left|\boldsymbol{I} + \tilde{\boldsymbol{H}}_b \boldsymbol{W}_a \boldsymbol{W}_{d,b} \boldsymbol{W}_{d,b}^H \boldsymbol{W}_a^H \tilde{\boldsymbol{H}}_b^H\right|$$

$$\text{subject to } \frac{1}{\mathrm{B}}\sum ||\boldsymbol{W}_a \boldsymbol{W}_{d,b}||_F^2 \leq N$$

- Projected gradient ascent (PGA)

$$\boldsymbol{W}_a^{(k+1)} = \Pi\left(\boldsymbol{W}_a^{(k)} - \mu_a^{(k)} \nabla_{\boldsymbol{W}_a} R\left(\boldsymbol{W}_a^{(k)}, \left\{\boldsymbol{W}_{d,b}^{(k)}\right\}\right)\right)$$

$$\boldsymbol{W}_{\boldsymbol{d,b}}^{(k+1)} = \Pi\left(\boldsymbol{W}_{d,b}^{(k)} - \mu_{d.b}^{(k)} \nabla_{\boldsymbol{W}_{d,b}} R\left(\boldsymbol{W}_a^{(k+1)}, \left\{\boldsymbol{W}_{d,b}^{(k)}\right\}\right)\right)$$

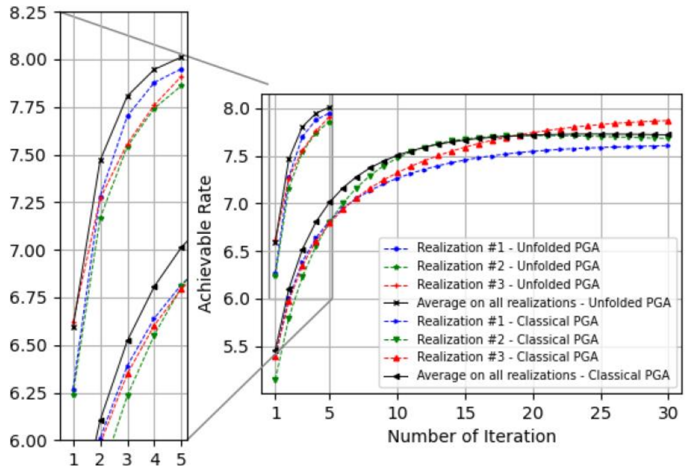
Projection operator

- Fix Q iterations

- Use past channels to optimize $\boldsymbol{\lambda} = \left\{\mu_a^{(k)}, \left\{\mu_{d.b}^{(k)}\right\}_{\boldsymbol{b}}\right\}$

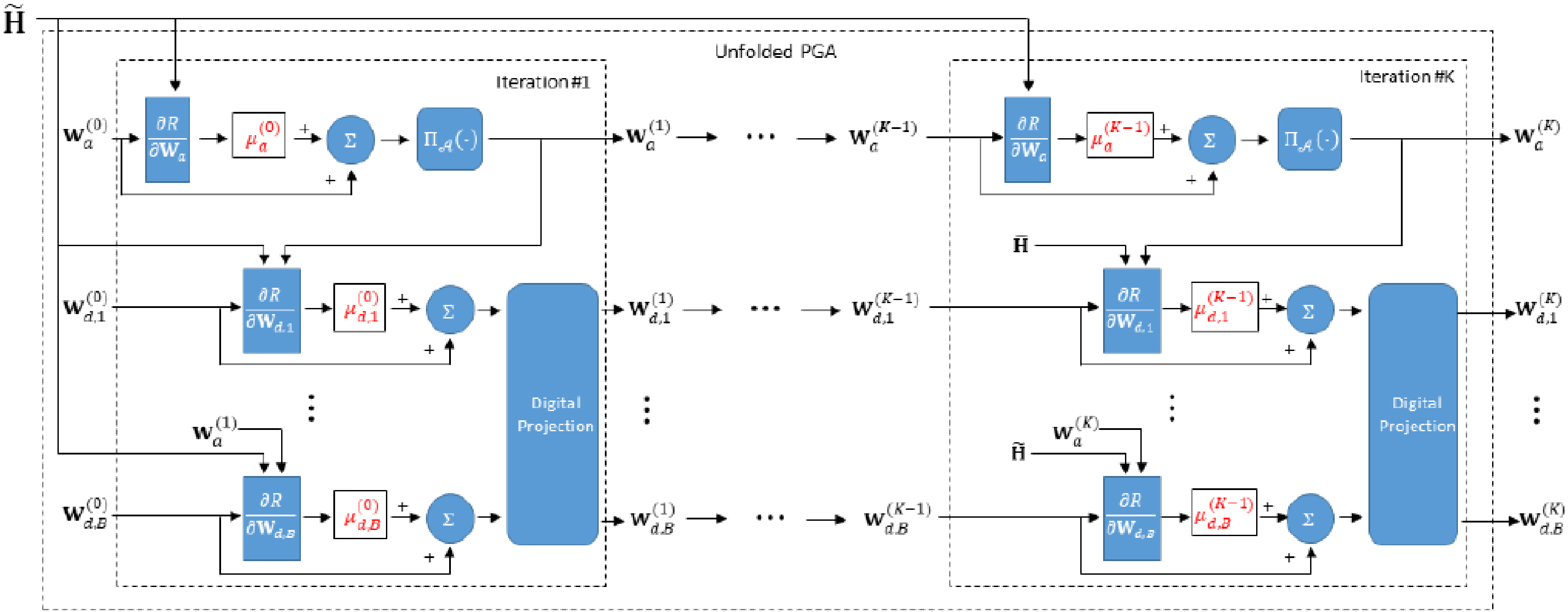
$$\boldsymbol{\lambda}^{\boldsymbol{k+1}} = \boldsymbol{\lambda}^{\boldsymbol{k}} - \eta \boldsymbol{\nabla}_{\boldsymbol{\lambda}} R\left(\boldsymbol{W}_a^{(Q)}, \left\{\boldsymbol{W}_{d,b}^{(Q)}\right\}; \boldsymbol{\lambda}^{\boldsymbol{k}}\right)$$

- Preserve pros of iterative optimizer
  - Suitability, interpretability, flexibility
- Improved convergence speed



- O. Agiv and N. Shlezinger, "Learn-to-Optimize for Rapid and Interpretable Hybrid Precoding", *IEEE SPAWC 2022*

Trainable Model (3/5)



Dealing with CSI Errors (4/5)

- Precoding requires channel $\{\widetilde{\boldsymbol{H}}_b\}$
- In practice…
  - Likely to observe noisy estimate $\{\widetilde{\boldsymbol{H}}_b+\boldsymbol{E}_b\}$

$$\underset{\boldsymbol{W}_a,\{\boldsymbol{W}_{d,b}\}}{\operatorname{argmax}} R = \frac{1}{B}\sum \log\left|\boldsymbol{I} + \widetilde{\boldsymbol{H}}_b\boldsymbol{W}_a\boldsymbol{W}_{d,b}\boldsymbol{W}_{d,b}^H\boldsymbol{W}_a^H\widetilde{\boldsymbol{H}}_b^H\right|$$
$$\text{subject to} \frac{1}{\mathrm{B}}\sum ||\boldsymbol{W}_a\boldsymbol{W}_{d,b}||_F^2 \leq N$$

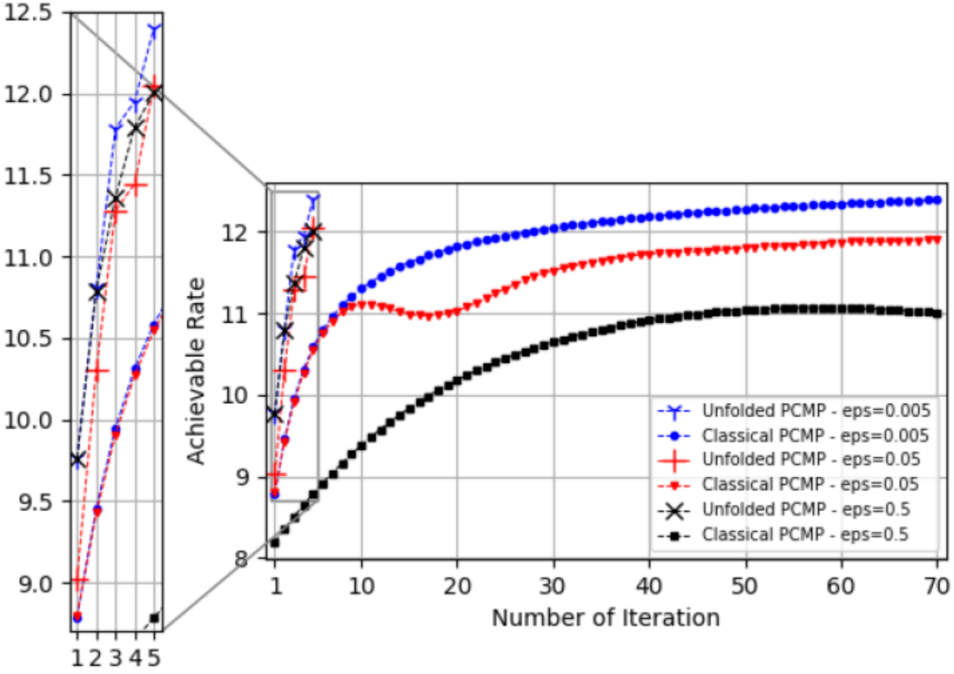
- How to cope?
  - Convert PGA objective to max-min

$$\underset{\boldsymbol{W}_a,\{\boldsymbol{W}_{d,b}\}}{\operatorname{argmax}} \ \underset{||\boldsymbol{E}_b||<\epsilon}{\min} \ R = \frac{1}{B}\sum \log\left|\boldsymbol{I} + (\widetilde{\boldsymbol{H}}_b+\boldsymbol{E}_b)\boldsymbol{W}_a\boldsymbol{W}_{d,b}\boldsymbol{W}_{d,b}^H\boldsymbol{W}_a^H(\widetilde{\boldsymbol{H}}_b^H + \boldsymbol{E}_b^H)\right|$$
$$\text{subject to} \frac{1}{\mathrm{B}}\sum ||\boldsymbol{W}_a\boldsymbol{W}_{d,b}||_F^2 \leq N$$

  - Minimax solver: projected conceptual mirror prox (PCMP)

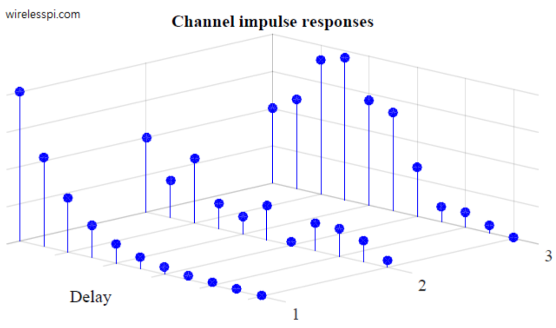
- Unfold algorithm

Results – QuadRiGA Channel (5/5)



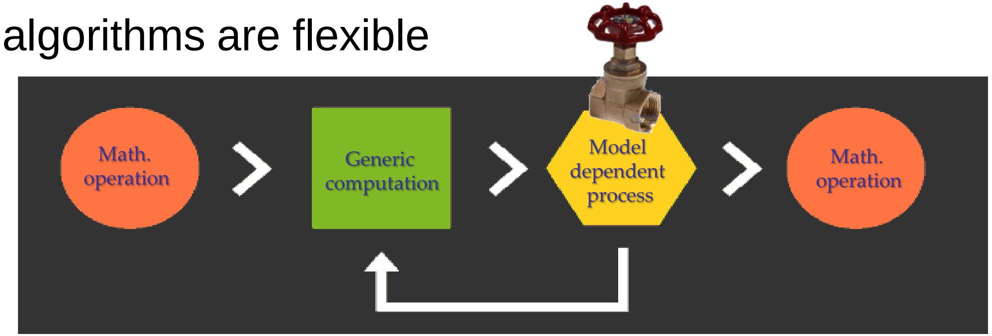
Coping with Dynamic Channels

Dynamic Channels



- Challenge: wireless channels statistics change in time
- Fundamental difference from traditional learning domains
- Model-based algorithms are flexible

Input x
(Measurement)

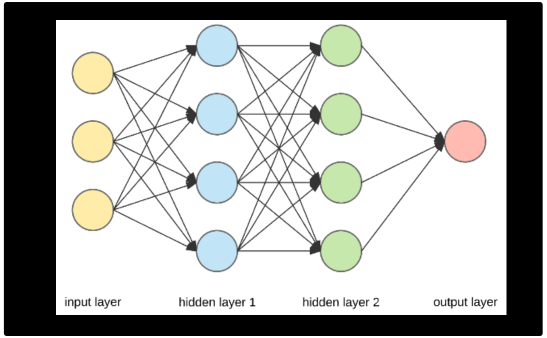


Output s

- Deep learning:

Many Inputs

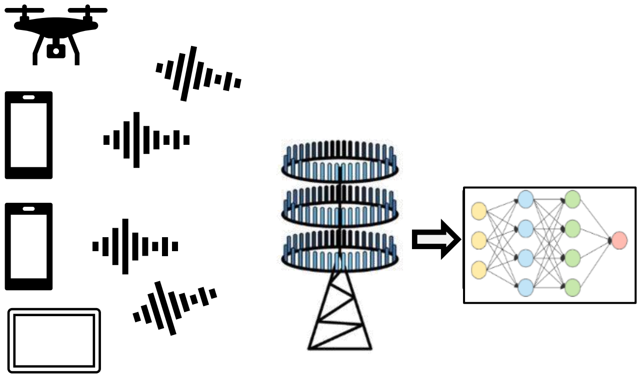
New x



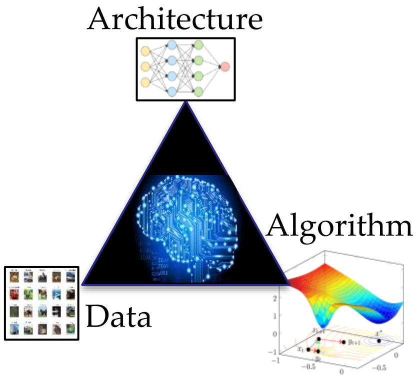
Many Outputs

s

Online Training of Deep Networks

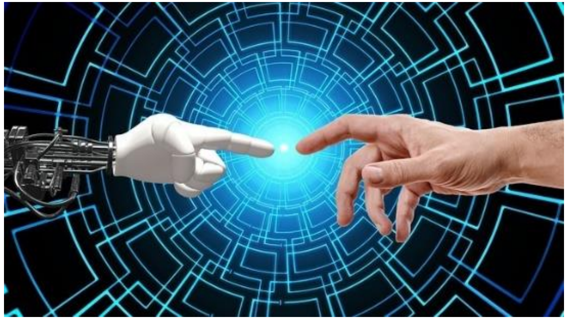


- Key aspects:
  - Architecture – apply on wireless devices
  - Training algorithm – adapt to dynamic channels
  - Data – train online with minimal overhead
- Architecture:
  - Instead of using "black boxes" → augment algorithms with AI
  - Use light-weight networks which learn model-based algorithms
- Training algorithm:
  - Allow rapid re-training to track dynamic variations
  - Meta-learning to optimize the optimizer with past data
  - Interpretable building blocks → train subsets of architecture
- Data:
  - Exploit inherent structures of communication protocols
  - Incorporate active learning



Conclusions

- Deep learning brings powerful data-driven tools
  - Yet, it's not magic
  - Highly parameterized optimization
  - Large labeled data sets
- Understand capabilities and limitations
- Model-based deep learning
  - Systematic framework for combining knowledge and data
  - Enhance established algorithms by learning
- Categorizing approaches to facilitate future design



Conclusions

- Many areas dominated by model-based algorithms
  - Communications
  - Radar
  - Control
  - Various signal processing applications

- The potential of model-based deep learning
  - The tip of the iceberg

- N. Shlezinger, J. Whang, Y. C. Eldar, and A. G. Dimakis. "Model-Based Deep Learning." *arXiv 2012.08405*
- N. Shlezinger, Y. C. Eldar, and S. P. Boyd. "Model-Based Deep Learning: On the Intersection of Deep Learning and Optimization", *IEEE Access 2022*