NAVAL POSTGRADUATE SCHOOL

Compositional Features and Neural Network Complexity for Dynamical Systems

Wei Kang
Department of Applied Mathematics
Naval Postgraduate School
and
University of California, Santa Cruz

The 3rd Symposium on Machine Learning and Dynamical Systems
Fields Institute

Background and literature

Control and dynamical systems

Modeling: $\mathbf{f} : \mathbf{x} \in \mathbb{R}^{\mathrm{r}} \to \mathbb{R}^{\mathrm{d}}$

Feedback Control $\mathbf{u} : \mathbf{x} \in \mathbb{R}^{\mathrm{d}} \to \mathbb{R}^{\mathrm{m}}$

Stability $\mathbf{V} : \mathbf{x} \in \mathbb{R}^{\mathrm{d}} \to \mathbb{R}$

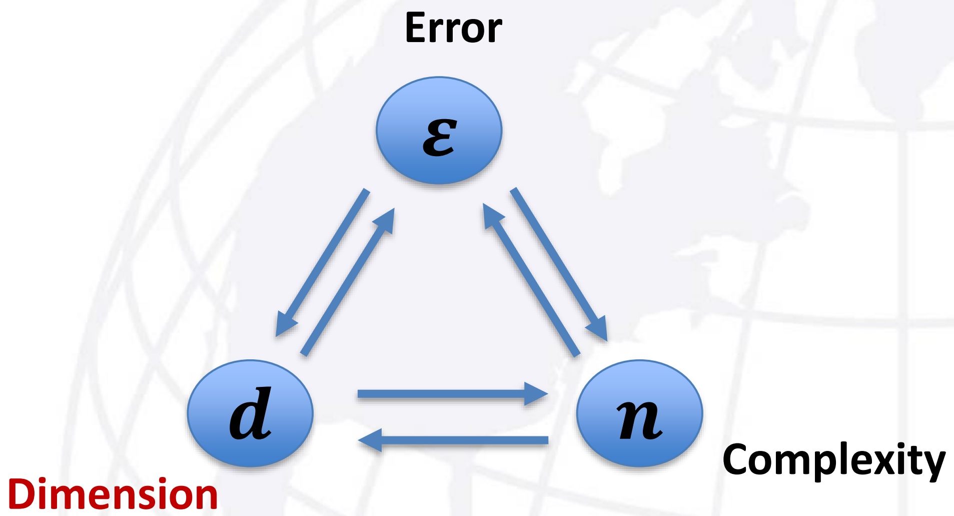
……… ………

Regression

$$f : \mathbb{R}^d \longrightarrow \mathbb{R}$$

$$f^{NN} \approx f\ ?$$

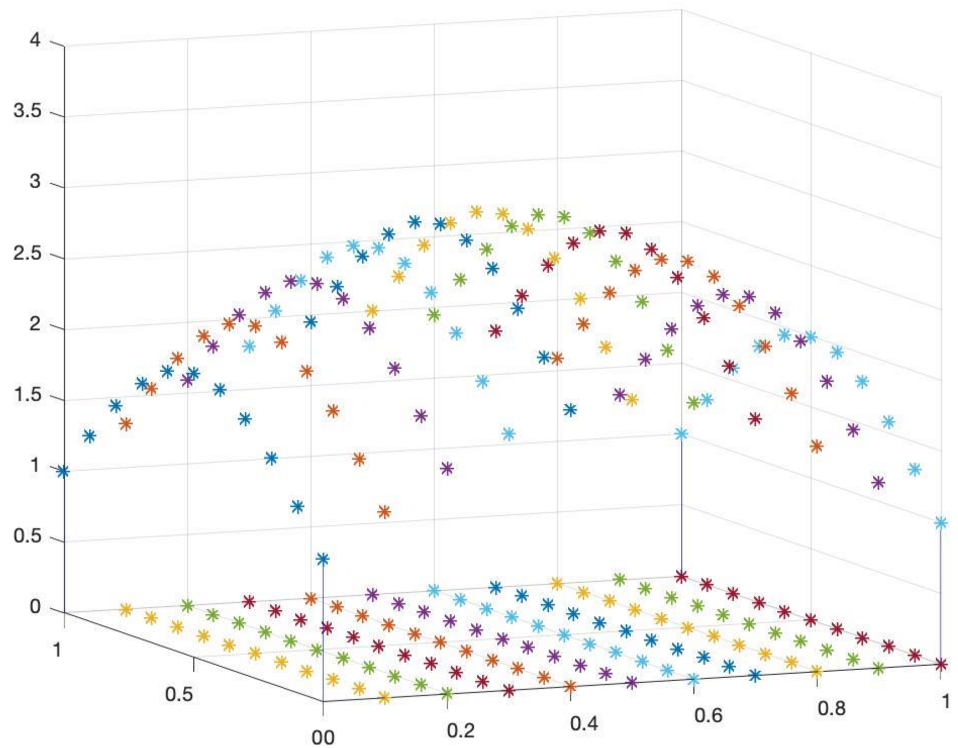
What's the big deal?



Background and literature

Curse-of-dimensionality: Given an error upper bound, the complexity increases exponentially with the dimension.

Dataset Size $= N^d$



Richard Ernest Bellman

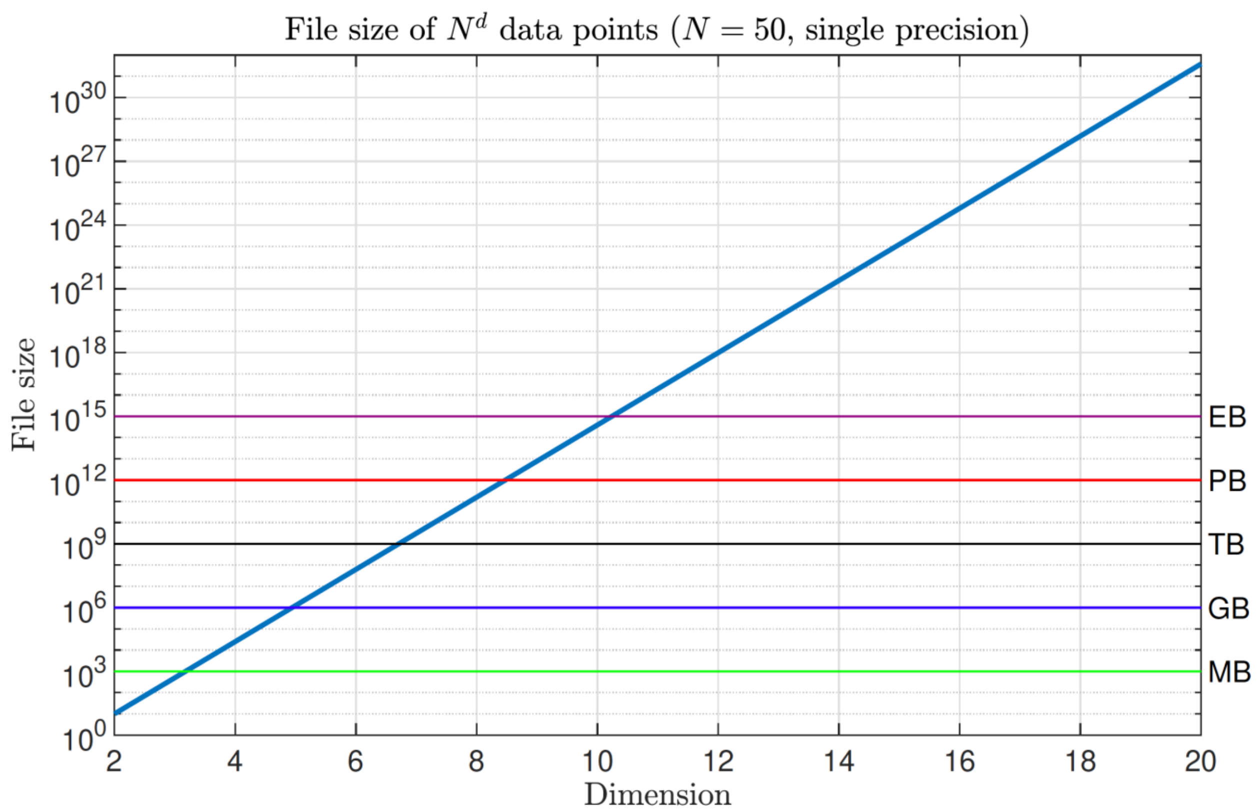
* Dynamic programming
* Curse of dimensionality
* HJB equation

……

Background and literature

Curse-of-dimensionality: Given an error upper bound, the complexity increases exponentially with the dimension.

Dataset Size $= N^d$



Background and literature

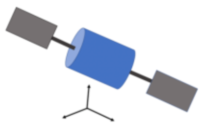
Empirical successes of machine learning

Solving high dimensional problems

- I. E. Lagaris, A. Likas, D. Fotiadis, *Artificial neural networks for solving ordinary and partial differential equations*, IEEE Trans. Neural Networks, 1998. (***d=2***)
- Y. Tassa and T. Erez, *Least square solutions of the HJB equation with neural network value-function approximations*, IEEE Trans. Neural Networks, 2007 (***d=4***)
- J. Han and W. E, *Deep learning approximation for stochastic control problems*, arXiv 2016 (***d=100***)
- J. Han, A. Jentzen, W. E., *Solving high-dimensional PDEs using deep learning*, arXiv 2017 (***d=100****)*
- J. Sirignano and K. Spiliopoulos, *DGM: A deep learning algorithm for solving PDEs*, arXiv 2018 (***d=200****)*
- M. Raissi, P. Perdikaris, G. Karniadakis, *Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear PDEs*, J. Comput. Phys., 2018 ($d_x=2, d_\lambda=2$)
- T. Nakamura-Zimmerer, Q. Gong, W. Kang, *Adaptive deep learning for high-dimensional HJB equations*, arXiv 2019 (***d=30***)
- D. Izzo, E. Öztürk, and M. Märtens, *Interplanetary transfers via deep representations of the optimal policy and/or of the value function*, arXiv 2019 (***d=7***)
- B. Azmi, D. Kalise, K. Kunisch, Optimal feedback law recovery by gradient-augmented sparse polynomial regression, arXiv 2020. (***d=80****)*

Some examples of dynamical systems

- T. Nakamura-Zimmerer, Q. Gong, W. Kang, *Adaptive deep learning for high-dimensional HJB equations*, SIAM J. Scientific Computing, 2021



(*d*=6, 30 ~ 10^6, 10^{30})

- T. Nakamura-Zimmerer, Q. Gong, W. Kang, QRnet: optimal regulator with LQR-augmented neural networks, IEEE Control Systems Letters, 2021.

(*d*=64)

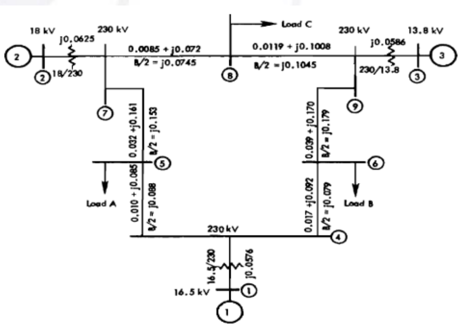
We combine the raw NN prediction (13) with the LQR value function (8) for the linearized dynamics (7) as

$$V^{\mathrm{NN}}(\boldsymbol{x}) = \frac{1}{c}\log\left[1 + cV^{\mathrm{LQR}}(\boldsymbol{x})\right] + W^{\mathrm{NN}}(\boldsymbol{x}), \qquad (14)$$

with a trainable parameter $c > 0$. Intuitively, LQR provides a

- W. Kang, Q. Gong, T. Nakamura-Zimmerer, F. Fahroo, Algorithms of data generation for deep learning and feedback design: A survey, Physica D: Nonlinear Phenomena, 2021

(*d*=30)



Background and literature

Why does deep learning work for so many high dimensional problems?

Background and literature

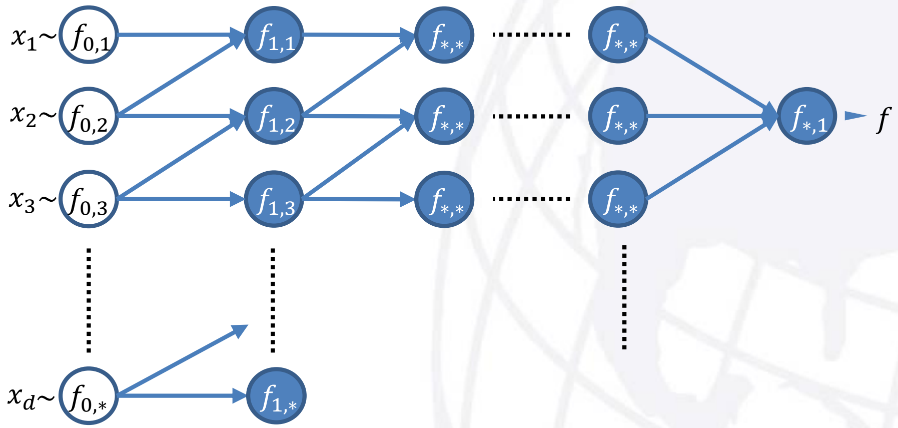
Approximation theory -

Rate of convergence, error upper bound, complexity, ……

- A. R. Barron, *Universal approximation bounds for superpositions of a sigmoidal function*, IEEE Trans. on Information Theory, 1993.
- W. E, C. Ma, S. Wojtowytsch and L. Wu, *Towards a mathematical understanding of neural network-based machine learning: what we know and what we don't*, arXiv 2020.
- P. C. Kainen, V. Kůrková, M. Sanguineti, *Approximating multivariable functions by feedforward neural nets*. In: Handbook on Neural Information Processing, Springer, 2013.
- T. Poggio, H. Mhaskar, L. Rosasco, B. Miranda, Q. Liao, *Why and when can deep - but not shallow - networks avoid the curse of dimensionality: a review*, arXiv 2017.
- H. N. Mhaskar and T. Poggio, *Deep vs. shallow networks: an approximation theory perspective*, arXiv 2016.
- W. Kang and Q. Gong, Feedforward *neural networks and compositional functions with applications to dynamical systems*, SIAM J. Control and Optim., 2022

Compositional function as a layered DAG

- W. Kang and Q. Gong, *Neural network approximations of compositional functions with applications to dynamical systems*, SIAM J. Control and Optimization, 2022.
- T. Poggio et al., *Why and when can deep – but not shallow – networks avoid the curse of dimensionality: A review*, arXiv 2017
- H. N. Mhaskar and T. Poggio, *Deep vs. shallow networks: an approximation theory perspective*, arXiv 2016.

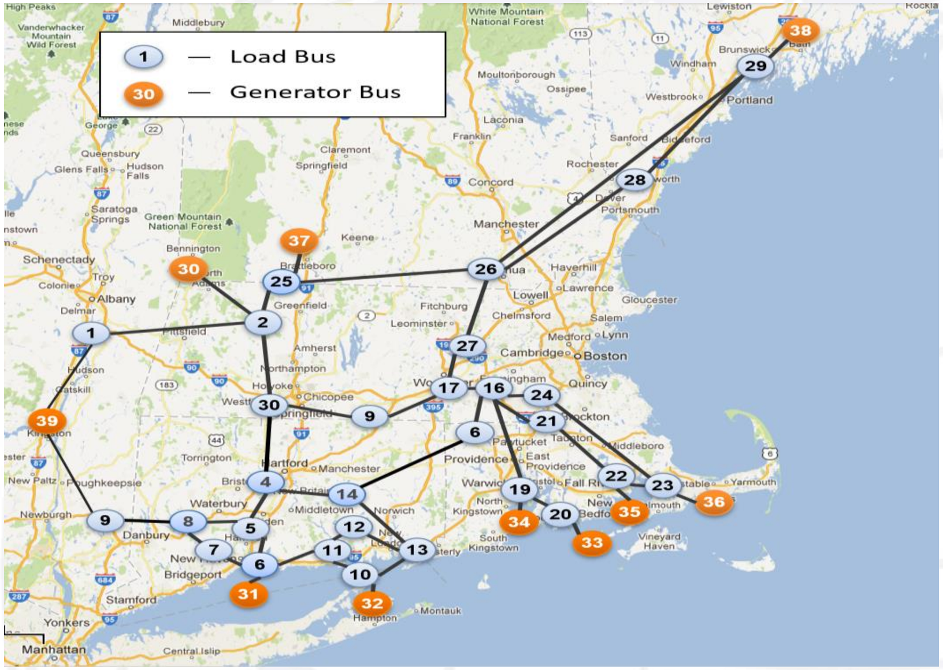


Layered directed acyclic graph (layered DAG)

Algebra of compositional functions

Observation: It is widely observed in science and engineering that complicated and high dimensional input-output information relations can be represented as compositions of functions with low input dimensions.

EI: A large electrical grid in North America. A simplified model has more than 25K buses, 28K lines, 8K transformers, and over 1,000 generators.

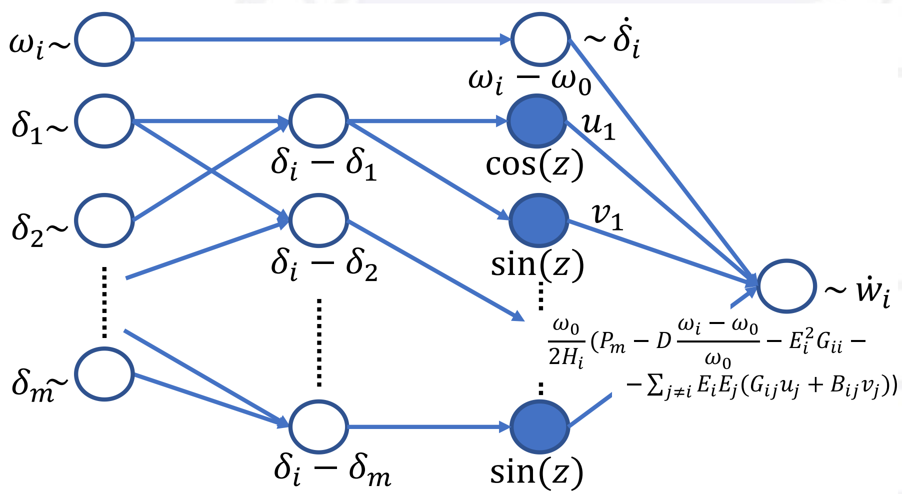


Algebra of compositional functions

Observation: It is widely observed in science and engineering that complicated and high dimensional input-output information relations can be represented as compositions of functions with low input dimensions.

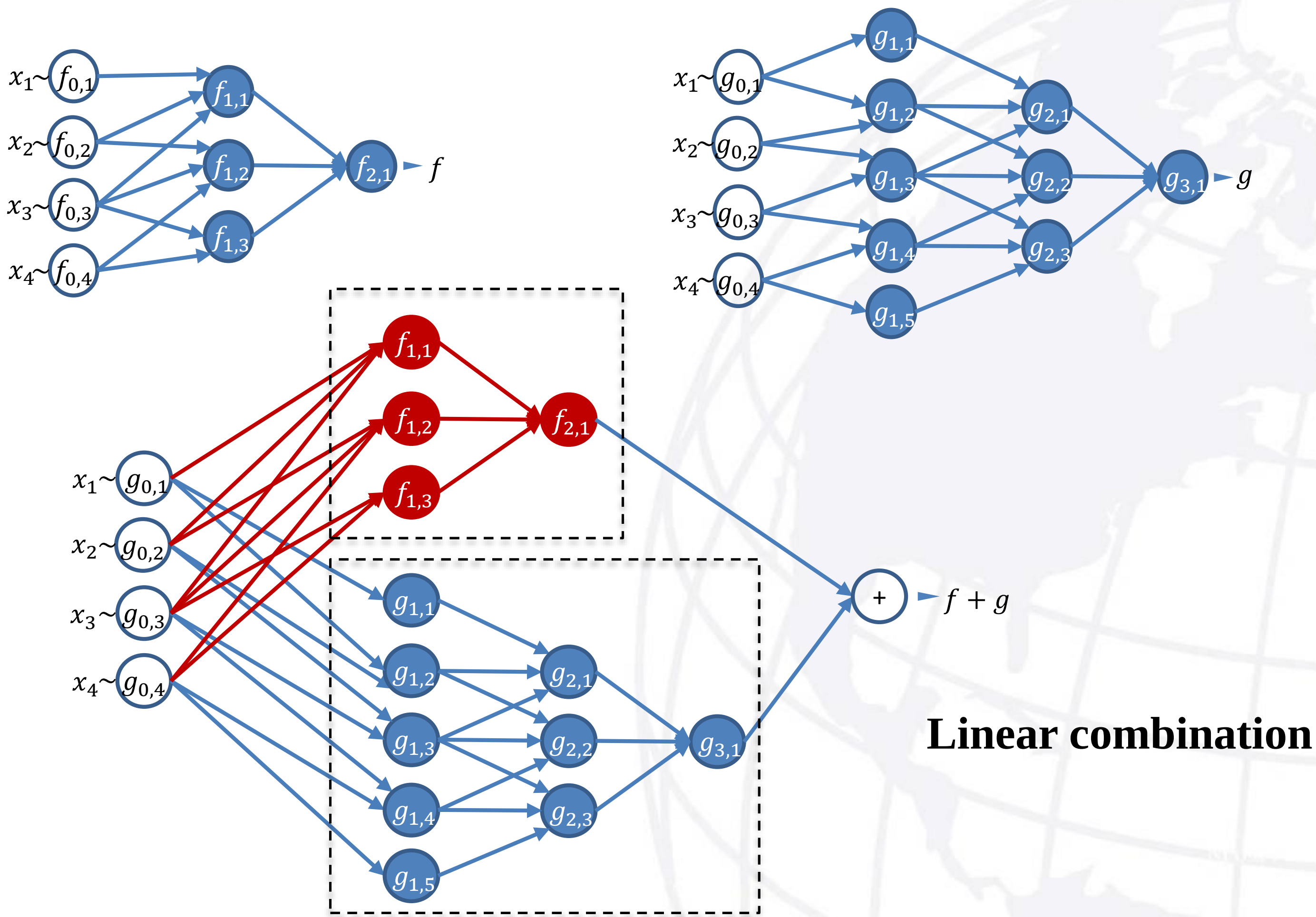
The swing equation of power systems

$$\frac{d\omega_i}{dt} = \frac{\omega_0}{2H_i}\left(P_m - D\frac{\omega_i - \omega_0}{\omega_0} - E_i^2 G_{ii} \cdots \right.$$
$$\left. - \sum_{j=1, j\neq i}^{10} E_i E_j [B_{ij}\sin(\delta_i - \delta_j) + G_{ij}\cos(\delta_i - \delta_j)]\right)$$
$$\frac{d\delta_i}{dt} = \omega_i - \omega_0,$$



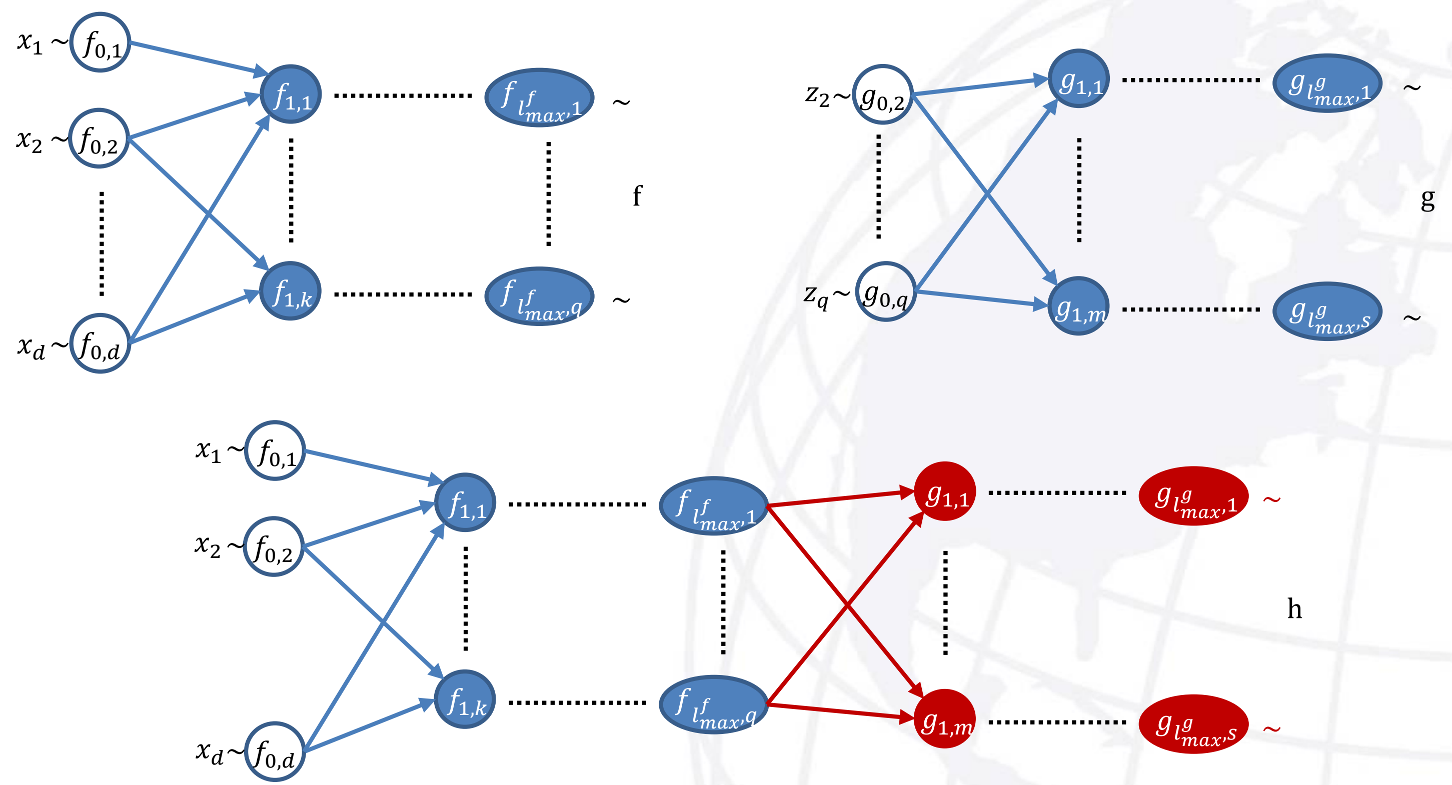
Algebra of compositional functions

Observation: It is widely observed in science and engineering that complicated and high dimensional input-output information relations can be represented as compositions of functions with low input dimensions.



Algebra of compositional functions

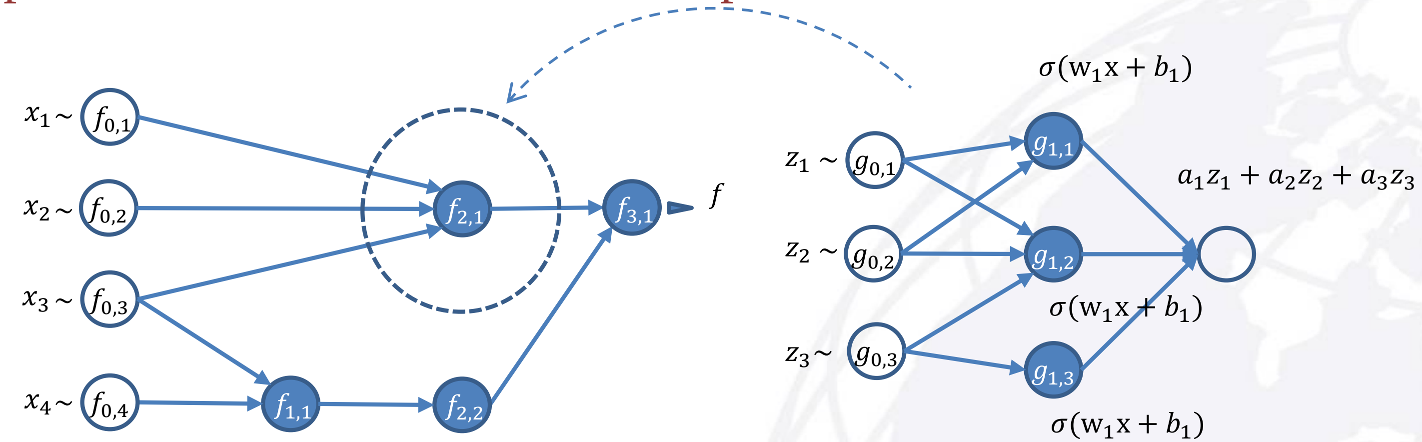
Observation: It is widely observed in science and engineering that complicated and high dimensional input-output information relations can be represented as compositions of functions with low input dimensions.

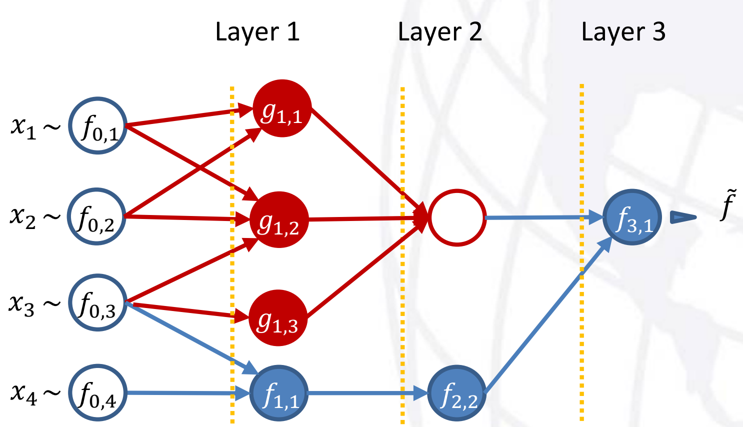


Composition of compositional functions

Algebra of compositional functions

Observation: It is widely observed in science and engineering that complicated and high dimensional input-output information relations can be represented as compositions of functions with low input dimensions.



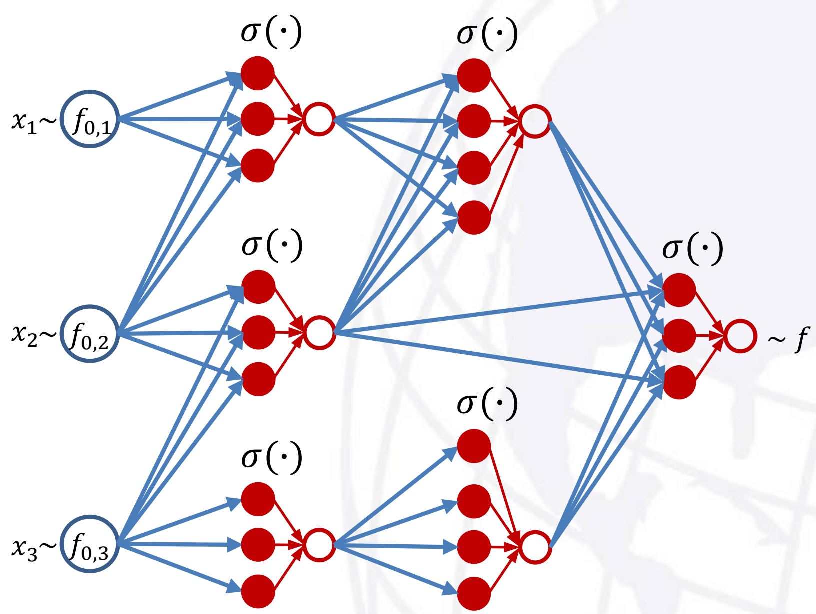


Substitution

Algebra of compositional functions

Observation: It is widely observed in science and engineering that complicated and high dimensional input-output information relations can be represented as compositions of functions with low input dimensions.

Neural networks are compositional functions



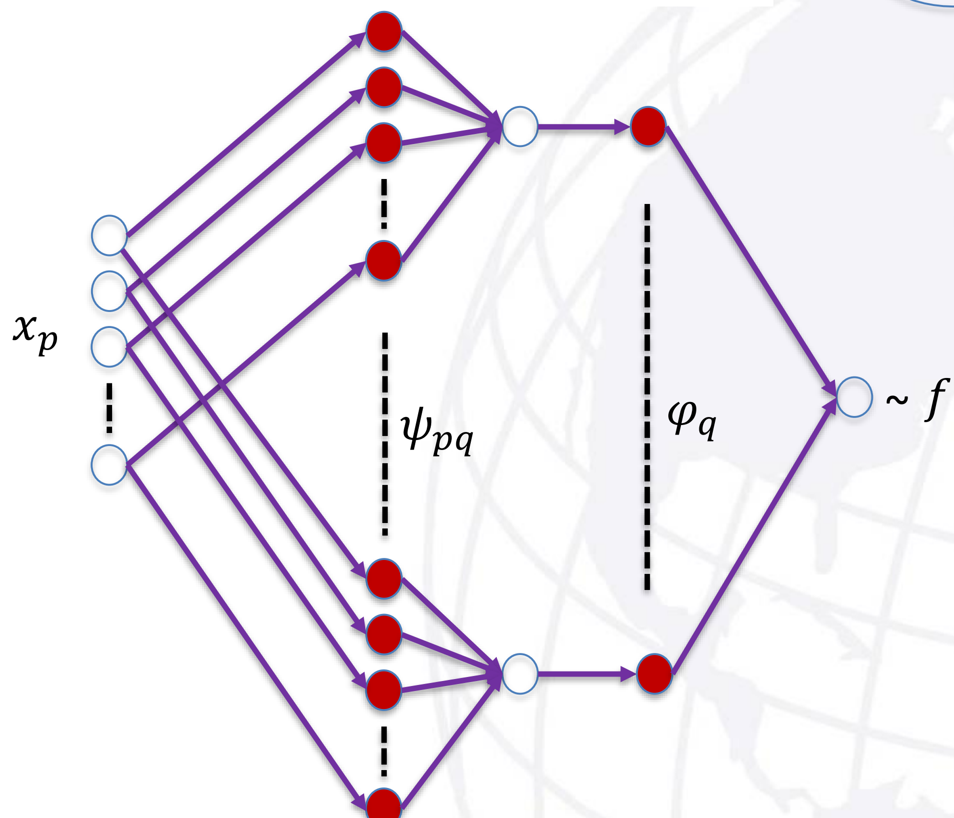
Wei Kang

Algebra of compositional functions

- Iterative computational algorithms are (deep) compositional functions.
- Kolmogorov-Arnold representation theorem

$$f(x_1, \cdots, x_d) = \sum_{q=1}^{2d+1} \phi_q \left(\sum_{p=1}^{d} \psi_{pq}(x_p) \right)$$

Every continuous function is a layered DAG



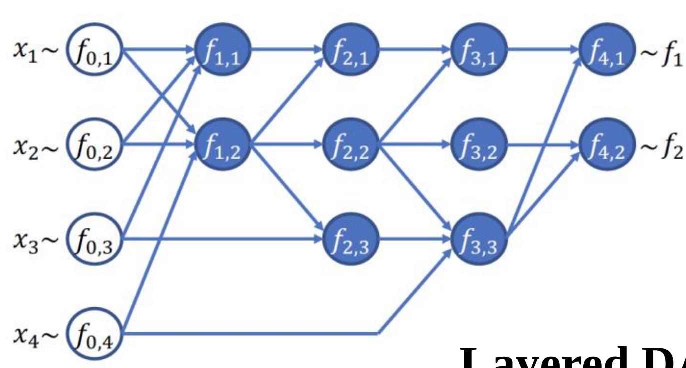
Compositional features

Dimension feature: $r^{\mathbf{f}}_{max}$ – the largest $d_{i,j}/m_{i,j}$

Volume feature: $\Lambda^{\mathbf{f}}$ – the largest $\max\{(R_{i,j})^{m_{i,j}}, 1\} \|f_{i,j}\|_{W^{\infty}_{m_{i,j},d_{i,j}}}$

Lipschitz constant feature: $L^{\mathbf{f}}_{max}$ – the largest $|L_{i,j}|$

Complexity feature: $|\mathcal{V}^{\mathbf{f}}_G|$– the total number of nodes



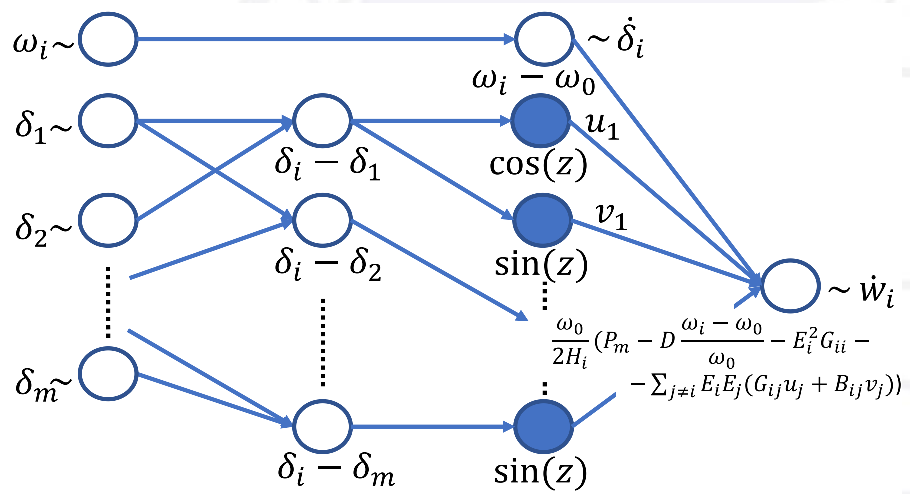
Layered DAG

Observation: It is widely observed in science and engineering that complicated and high dimensional input-output information relations can be represented as compositions of functions with low input dimensions.

The swing equation of power systems

$$\frac{d\omega_i}{dt} = \frac{\omega_0}{2H_i}\left(P_m - D\frac{\omega_i - \omega_0}{\omega_0} - E_i^2 G_{ii} \cdots - \sum_{j=1, j\neq i}^{10} E_i E_j [B_{ij}\sin(\delta_i - \delta_j) + G_{ij}\cos(\delta_i - \delta_j)]\right)$$

$$\frac{d\delta_i}{dt} = \omega_i - \omega_0,$$

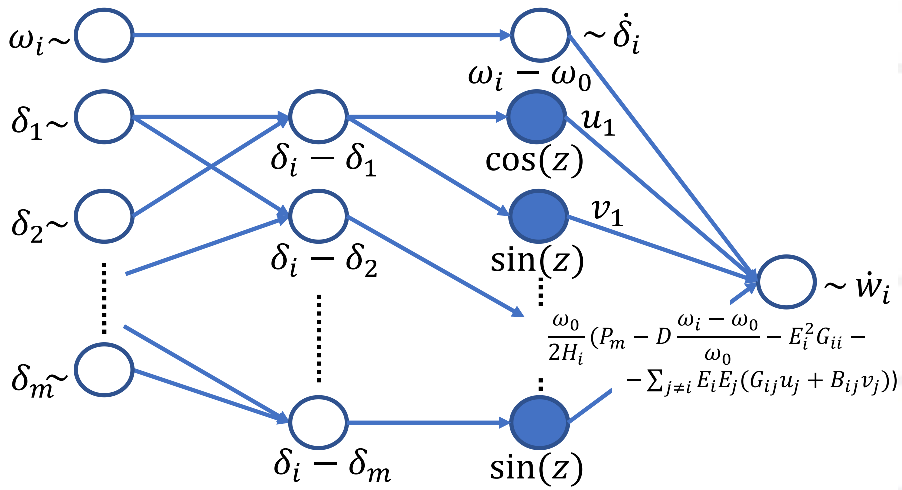


Examples

Observation: It is widely observed in science and engineering that complicated and high dimensional input-output information relations can be represented as compositions of functions with low input dimensions.

Compositional features

$$r_{max} = 1, \quad \Lambda = 4\pi, \quad |\mathcal{V}| = 2(N_g - 1)N_g,$$
$$L_{max} = \max_{1\leqslant i,j\leqslant N_g, i\neq j}\left\{\frac{\omega_0}{2H_i}E_iE_jG_{ij}, \frac{\omega_0}{2H_i}E_iE_jB_{ij}\right\}$$

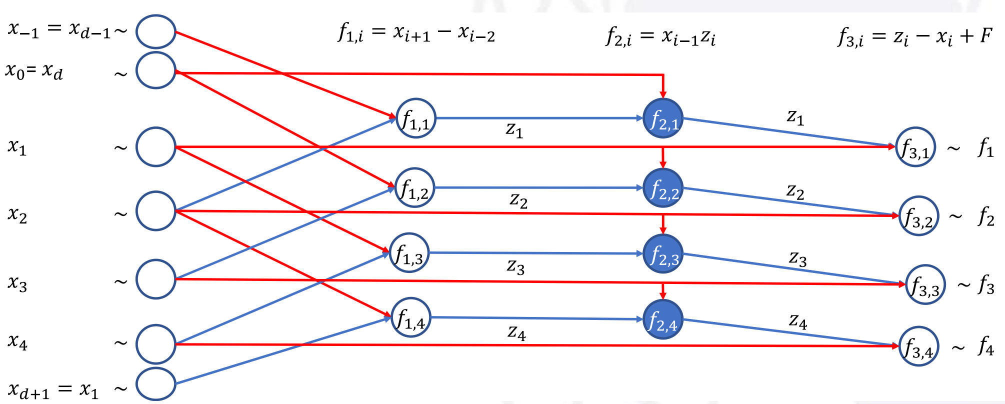


Examples

Observation: It is widely observed in science and engineering that complicated and high dimensional input-output information relations can be represented as compositions of functions with low input dimensions.

The Lorenz-96 model

$$\dot{\mathbf{x}} = \begin{bmatrix} x_0(x_2 - x_{-1}) - x_1 + F \\ x_1(x_3 - x_0) - x_2 + F \\ \vdots \\ x_{i-1}(x_{i+1} - x_{i-2}) - x_i + F \\ \vdots \\ x_{d-1}(x_{d+1} - x_{d-2}) - x_d + F \end{bmatrix}$$

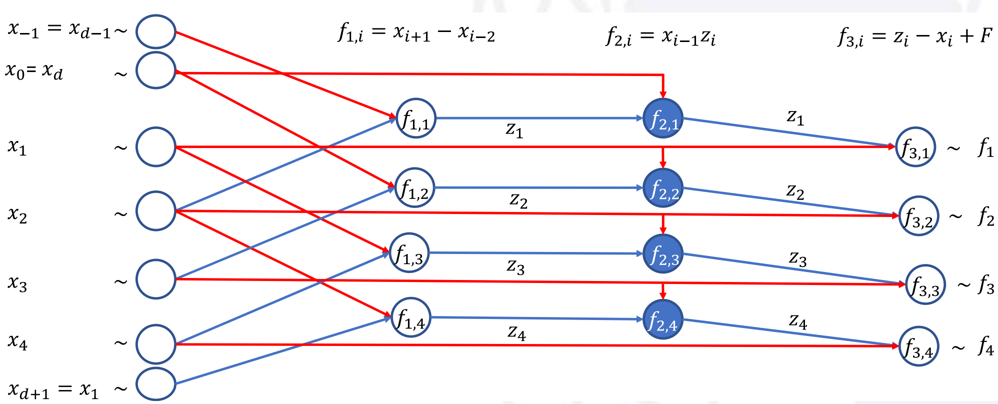


Examples

Observation: It is widely observed in science and engineering that complicated and high dimensional input-output information relations can be represented as compositions of functions with low input dimensions.

The Lorenz-96 model

$$\begin{array}{lllllll} r_{max} & = & 2, & \Lambda & = & \max\{(2R),1\}(2R+4R^2), \\ L_{max} & = & 1, & |\mathcal{V}| & = & d \end{array}$$



Wei Kang

Examples

Observation: It is widely observed in science and engineering that complicated and high dimensional input-output information relations can be represented as compositions of functions with low input dimensions.

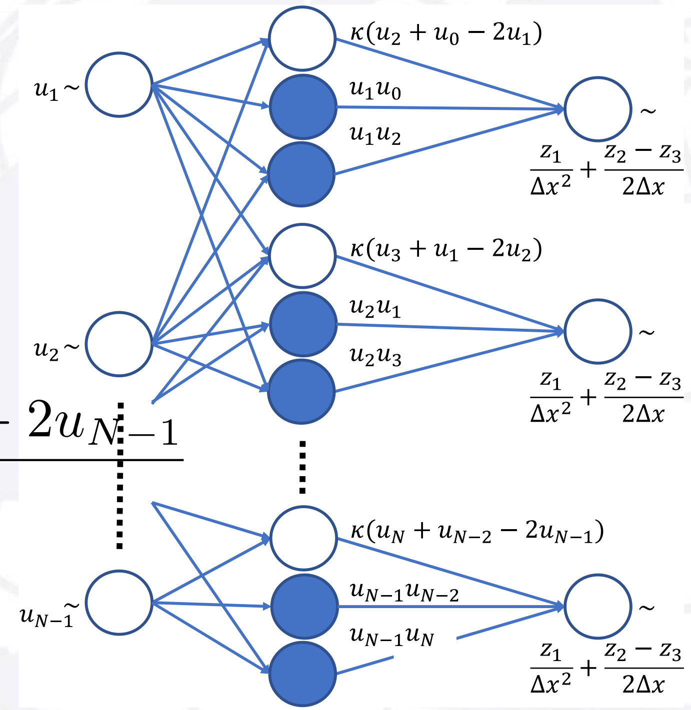
Discretization of PDEs

$$\dot{u}_1 = -u_1 \frac{u_2 - u_0}{2\Delta x} + \kappa \frac{u_2 + u_0 - 2u_1}{\Delta x^2}$$

$$\dot{u}_2 = -u_2 \frac{u_3 - u_1}{2\Delta x} + \kappa \frac{u_3 + u_1 - 2u_2}{\Delta x^2}$$

$$\vdots$$

$$\dot{u}_{N-1} = -u_{N-1} \frac{u_N - u_{N-2}}{2\Delta x} + \kappa \frac{u_N + u_{N-2} - 2u_{N-1}}{\Delta x^2}$$

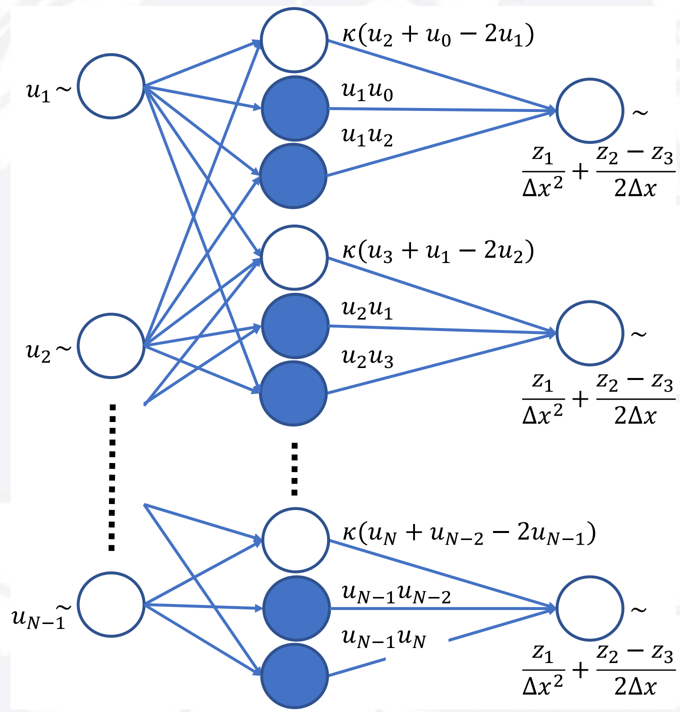


Examples

Observation: It is widely observed in science and engineering that complicated and high dimensional input-output information relations can be represented as compositions of functions with low input dimensions.

Discretization of PDEs

$$r_{max} = 2, \Lambda = \frac{1}{2}R^2 + R,$$
$$L_{max} = \frac{N}{L}, |\mathcal{V}_G| = 2N.$$

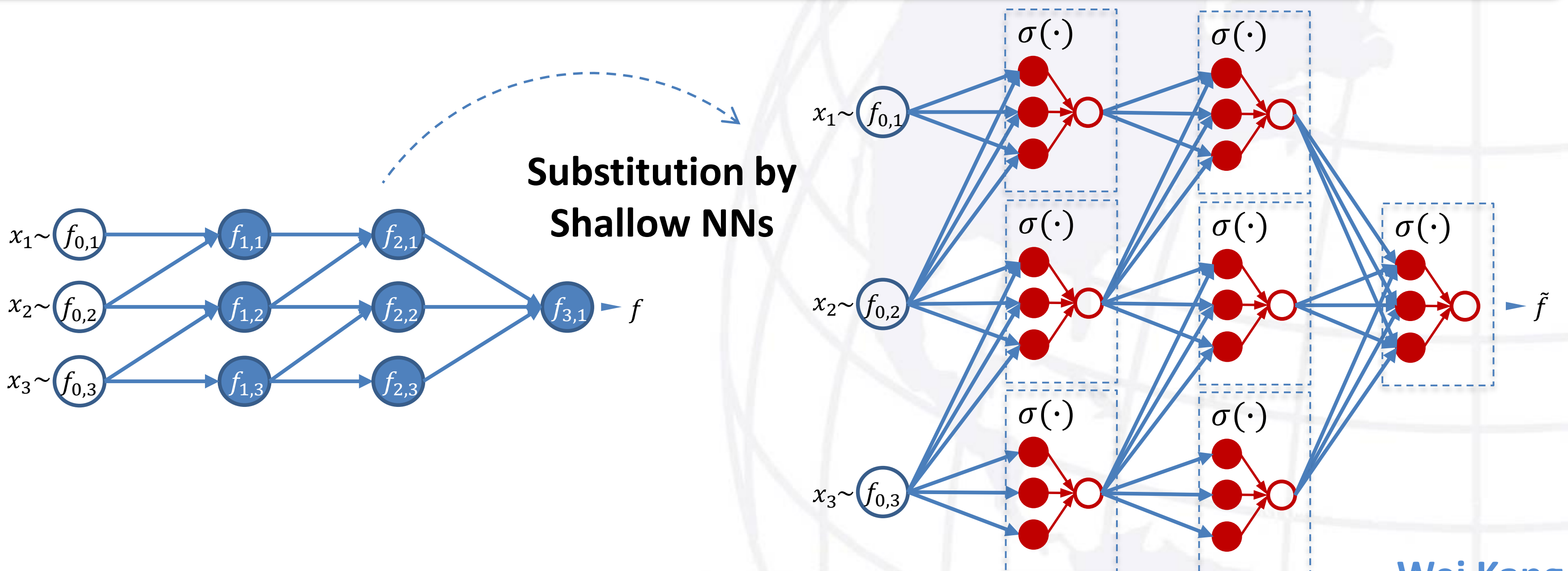


Error propagation - substitution

Theorem Under smoothness assumptions, the neural network approximation error is bounded by

$$\left\|\tilde{\mathbf{f}}(\boldsymbol{x}) - \mathbf{f}(\boldsymbol{x})\right\|_p \leq \sum_{i,j=1}^{K} L^{\mathbf{f}}_{i,j}\epsilon_{i,j}, \quad \text{for all } \boldsymbol{x} \text{ in the domain of } \mathbf{f}.$$

where $L_{i,j}$ is the Lipschitz constant associated with the node $f_{i,j}$, $\epsilon_{i,j}$ is the error of each node due to substitution.

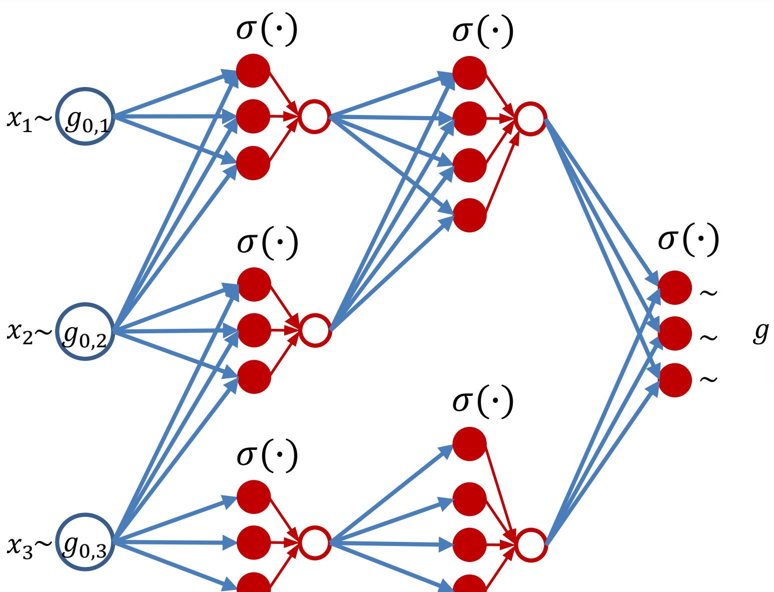


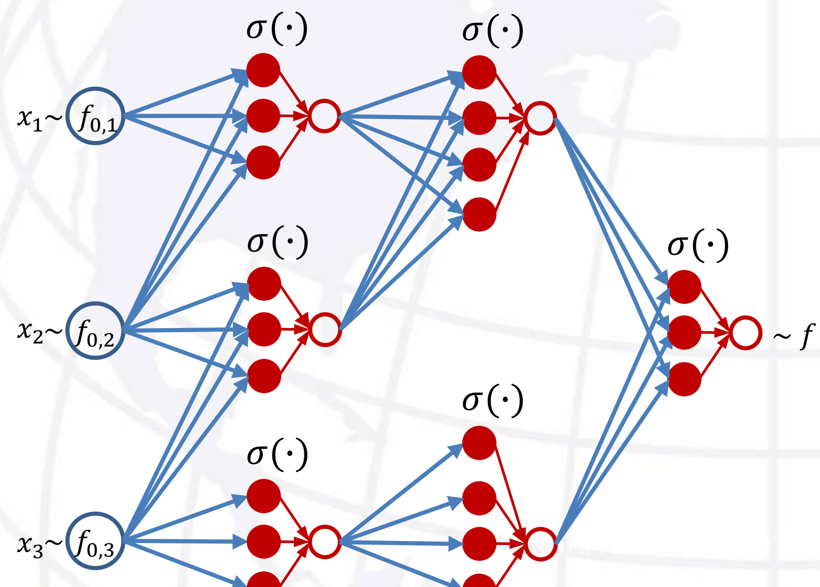
Error propagation - composition

Proposition Let $\tilde{\mathbf{f}}$, $\tilde{\mathbf{g}}$ and $\tilde{\mathbf{h}}$ be approximations of $\tilde{\mathbf{f}}$, $\tilde{\mathbf{g}}$ and $\tilde{\mathbf{h}}$ with errors bounded by ϵ_1, ϵ_2 and ϵ_3, respectively. Then

$$\left\|(\mathbf{f}(\cdot))^K \circ \mathbf{g}(\boldsymbol{x}) - (\tilde{\mathbf{f}}(\cdot))^K \circ \tilde{\mathbf{g}}(\boldsymbol{x})\right\|_p \leq \frac{(L^{\mathbf{f}})^K - 1}{L^{\mathbf{f}} - 1} e_1 + (L^{\mathbf{f}})^K e_2.$$

$$\left\|\mathbf{h} \circ (\mathbf{f}(\cdot))^K(\boldsymbol{x}) - \tilde{\mathbf{h}} \circ (\tilde{\mathbf{f}}(\cdot))^K(\boldsymbol{x})\right\|_p \leq L^{\mathbf{h}} \frac{(L^{\mathbf{f}})^K - 1}{L^{\mathbf{f}} - 1} e_1 + e_3.$$





Wei Kang

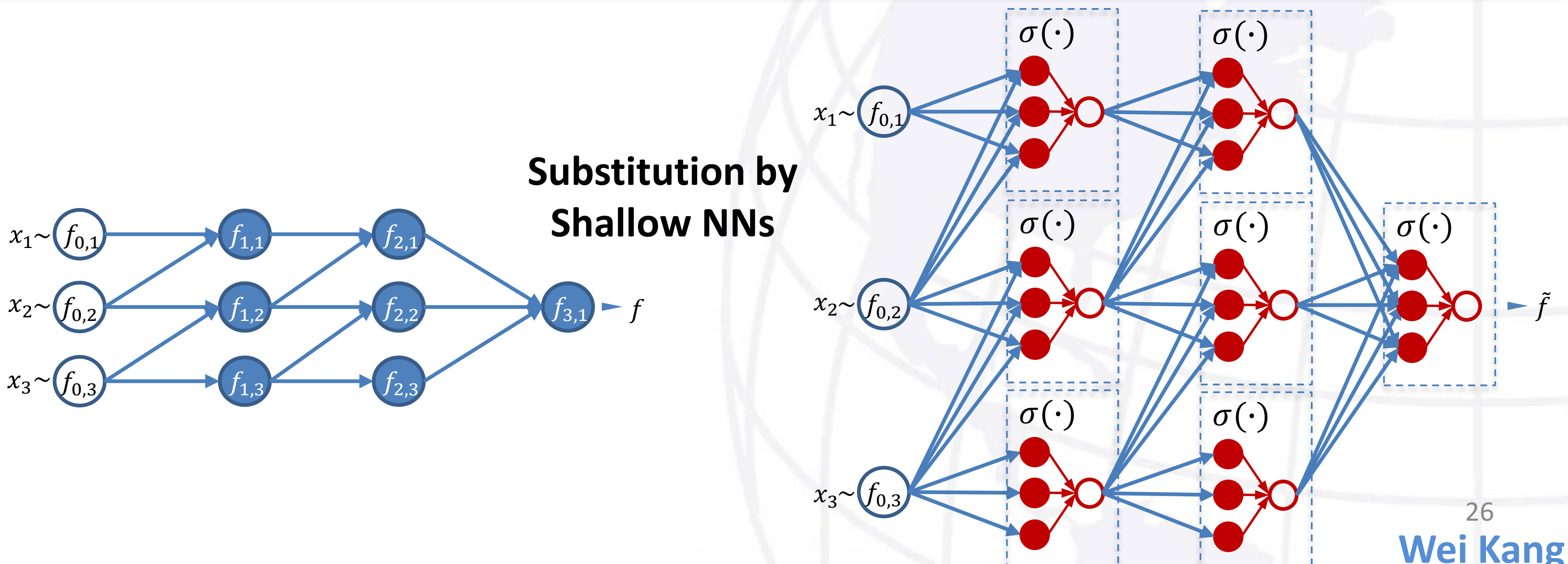
Approximation theory

Theorem Assume all nodes of a compositional function, $\mathbf{f}$, are C^1. For any integer $n_{width} > 0$, there exists a neural network, $\mathbf{f}^{NN}$, that has the following error upper bound,

$$\left\|\mathbf{f}(\boldsymbol{x}) - \mathbf{f}^{NN}(\boldsymbol{x})\right\|_p \leq C_1 L^{\mathbf{f}}_{max} \Lambda^{\mathbf{f}} \left|\mathcal{V}^{\mathbf{f}}_G\right| (n_{width})^{-1/r^{\mathbf{f}}_{max}}, \quad \text{for all } \boldsymbol{x} \in [-R, R]^d,$$

where C_1 is a constant determined by $\{d_{i,j}, m_{i,j}; \quad f_{i,j} \in \mathcal{V}^{\mathbf{f}}_G\}$. The complexity of $\mathbf{f}^{NN}$ is

$$n \leq \left|\mathcal{V}^{\mathbf{f}}_G\right| n_{width}.$$



Wei Kang

Trajectory of ODE

ODE

$$\dot{\boldsymbol{x}} = \mathbf{f}(\boldsymbol{x}), \quad \boldsymbol{x} \in \mathbb{R}^d, t \in [0, T]$$

Trajectory

$$\boldsymbol{x}(0) = \boldsymbol{x}$$
$$t \to \boldsymbol{\phi}(t; \boldsymbol{x})$$

Theorem Under C^1 assumption, for any $\epsilon > 0$, there exists a neural network approximation, $\boldsymbol{\phi}^{NN}$, satisfying

$$||\boldsymbol{\phi}(T; \boldsymbol{x}) - \boldsymbol{\phi}^{NN}(\boldsymbol{x}))||_p \leq \epsilon, \quad \boldsymbol{x} \in D^{\mathbf{f}}.$$

The complexity of $\boldsymbol{\phi}^{NN}$ is bounded by

$$n \leq C\epsilon^{-(r^{\mathbf{f}}+1)}$$

where C is a polynomial function of the compositional features of $\mathbf{f}$.

Optimal control

The problem of optimal control

$$\begin{cases} \min\limits_{U} J = \Psi(\boldsymbol{x}(T)), & \boldsymbol{x} \in \mathbb{R}^d \\ \dot{\boldsymbol{x}} = \mathbf{f}(\boldsymbol{x}, \boldsymbol{u}), & \boldsymbol{u} \in \mathbb{R}^q, \end{cases}$$

Zero-order hold control

$$U = \begin{bmatrix} \boldsymbol{u}(t_0)^T & \boldsymbol{u}(t_1)^T & \cdots & \boldsymbol{u}(t_{N_t}) \end{bmatrix}$$

Trajectory and terminal state

$$\boldsymbol{x}(T) = \boldsymbol{\phi}(T; \boldsymbol{x}_0, U)$$

Cost function

$$J = \Psi \circ \boldsymbol{\phi}(T; \boldsymbol{x}_0, U)$$

Theorem Under convexity and C^1 assumptions, for any $\epsilon > 0$ there exists a neural network approximation of the optimal control satisfying

$$||U^{*NN}(\boldsymbol{x}) - U^*(\boldsymbol{x})||_2 \leq 3\epsilon, \quad \boldsymbol{x} \in D.$$

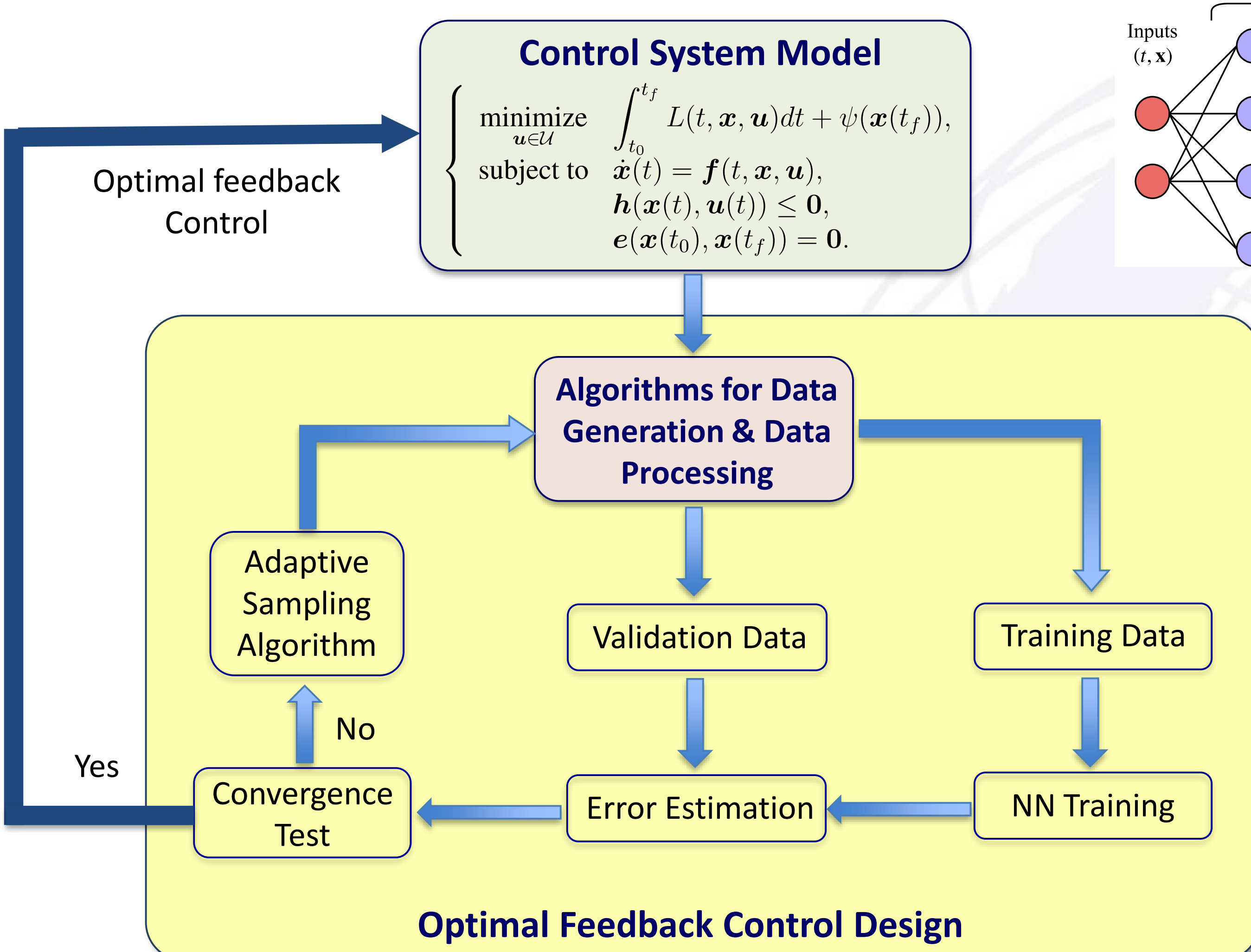
The complexity of U^{*NN} is bounded by

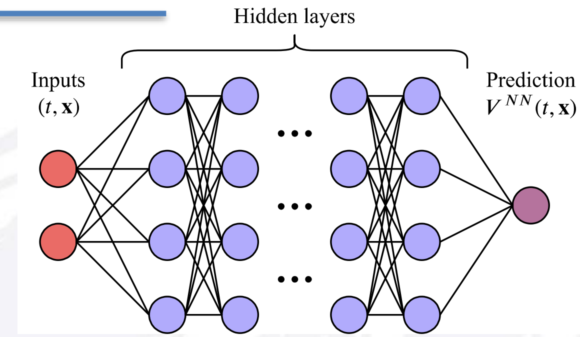
$$n \leq C\epsilon^{-(4r+1+\frac{4r}{r^{\mathbf{f}}_{max}})}$$

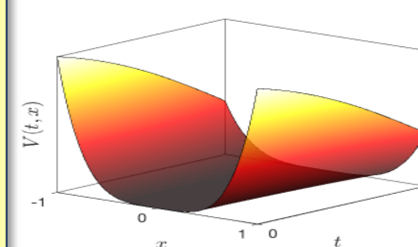
where $r = \max\{r^{\mathbf{f}}_{max}, r^{\Psi}_{max}\}$, C is a polynomial function of the compositional features of $\mathbf{f}$, Ψ, and the Hessian matrix of the cost function.

A model-based data-driven approach

❖ T. Nakamura-Zimmerer, Q. Gong, W. Kang, *Adaptive deep learning for high-dimensional HJB equations*, SIAM J. Scientific Computing, 2021.



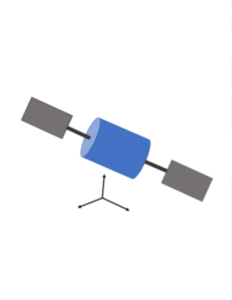




Example - rigid body optimal control

Problem formulation

$$\begin{cases} \min\limits_{\boldsymbol{u}(\cdot)} \int_0^{t_f} L(\boldsymbol{v},\boldsymbol{\omega},\boldsymbol{u})d\tau + \frac{W_4}{2}\|\boldsymbol{v}(t_f)\|^2 + \frac{W_5}{2}\|\boldsymbol{\omega}(t_f)\|^2, \\ \dot{\boldsymbol{v}} = \boldsymbol{E}(\boldsymbol{v})\boldsymbol{\omega}, \\ \boldsymbol{J}\dot{\boldsymbol{\omega}} = \boldsymbol{S}(\boldsymbol{\omega})\boldsymbol{R}(\boldsymbol{v})\boldsymbol{h} + \boldsymbol{B}\boldsymbol{u}. \end{cases}$$



State variables:

$\boldsymbol{v} = \begin{pmatrix}\phi & \theta & \psi\end{pmatrix}^T$ Euler angles

$\boldsymbol{\omega} = \begin{pmatrix}\omega_1 & \omega_2 & \omega_3\end{pmatrix}^T$ angular velocity

Running cost and weights:

$L(\boldsymbol{v},\boldsymbol{\omega},\boldsymbol{u}) = \frac{W_1}{2}\|\boldsymbol{v}\|^2 + \frac{W_2}{2}\|\boldsymbol{\omega}\|^2 + \frac{W_3}{2}\|\boldsymbol{u}\|^2,$

$W_1 = 1, W_2 = 10, W_3 = 0.5, W_4 = 1, W_5 = 1, t_f = 20.$

Matrices

$$\boldsymbol{E}(\boldsymbol{v}) := \begin{pmatrix} 1 & \sin\phi\tan\theta & \cos\phi\tan\theta \\ 0 & \cos\phi & -\sin\phi \\ 0 & \sin\phi/\cos\theta & \cos\phi/\cos\theta \end{pmatrix}, \qquad \boldsymbol{S}(\boldsymbol{\omega}) := \begin{pmatrix} 0 & \omega_3 & -\omega_2 \\ -\omega_3 & 0 & \omega_1 \\ \omega_2 & -\omega_1 & 0 \end{pmatrix},$$

$$\boldsymbol{R}(\boldsymbol{v}) := \begin{pmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ \sin\phi\sin\theta\cos\psi - \cos\phi\sin\psi & \sin\phi\sin\theta\sin\psi + \cos\phi\cos\psi & \cos\theta\sin\phi \\ \cos\phi\sin\theta\cos\psi + \sin\phi\sin\psi & \cos\phi\sin\theta\sin\psi - \sin\phi\cos\psi & \cos\theta\cos\phi \end{pmatrix}$$

$$\boldsymbol{B} = \begin{pmatrix} 1 & 1/20 & 1/10 \\ 1/15 & 1 & 1/10 \\ 1/10 & 1/15 & 1 \end{pmatrix}, \qquad \boldsymbol{J} = \begin{pmatrix} 2 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 4 \end{pmatrix}, \qquad \boldsymbol{h} = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}.$$

Example – rigid body optimal control

HJB equation and feedback law

Define the Hamiltonian

$$H(t, \boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{u}) = L(t, \boldsymbol{x}, \boldsymbol{u}) + \boldsymbol{\lambda}^T \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{u})$$

Solve the HJB equation to find the optimal feedback law

$$\begin{cases} V_t(t, \boldsymbol{x}) + \min\limits_{\boldsymbol{u} \in \mathcal{U}} H(t, \boldsymbol{x}, V_{\boldsymbol{x}}, \boldsymbol{u}) = 0, \\ V(t_f, \boldsymbol{x}) = \psi(\boldsymbol{x}), \end{cases}$$

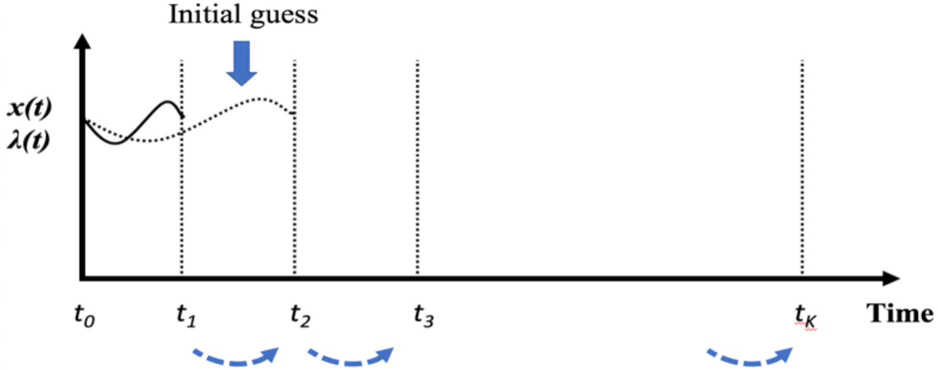
$$\boldsymbol{u}^*(t, \boldsymbol{x}) = \arg \min_{\boldsymbol{u} \in \mathcal{U}} H(t, \boldsymbol{x}, V_{\boldsymbol{x}}, \boldsymbol{u}).$$

Time-marching

Pontryagin's maximum principle

$$\begin{cases} \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{u}^*), & \boldsymbol{x}(0) = \boldsymbol{x}_0, \\ \dot{\boldsymbol{\lambda}}(t) = -H_{\boldsymbol{x}}(t, \boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{u}^*), & \boldsymbol{\lambda}(t_f) = F_{\boldsymbol{x}}(\boldsymbol{x}(t_f)), \\ \dot{v}(t) = -L(t, \boldsymbol{x}, \boldsymbol{u}^*), & v(t_f) = F(\boldsymbol{x}(t_f)). \end{cases}$$

where $\boldsymbol{u}^* = \arg\min_{\boldsymbol{u}} H(t, \boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{u})$



1. Choose a partition of $[t_0, t_f]$,
$$t_0 < t_1 < t_2 < \cdots < t_K = t_f.$$
2. In $[t_0, t_1]$, solve the TPBVP,
$$(\boldsymbol{x}^1(t), \boldsymbol{\lambda}^1(t)).$$
3. Extending the trajectory to $[t_0, t_2]$,
$$\boldsymbol{x}_0^2(t) = \begin{cases} \boldsymbol{x}^1(t), & \text{if } t_0 \le t \le t_1, \\ \boldsymbol{x}^1(t_1), & \text{if } t_1 < t \le t_2, \end{cases}$$
$\boldsymbol{\lambda}_0^2(t)$ is similarly defined.
4. $(\boldsymbol{x}_0^2(t), \boldsymbol{\lambda}_0^2(t))$ is used as an initial guess to solve the TPBVP over $[0, t_2]$.
5. Repeating the process until $t_K = t_f$.

Neural network warm start

$$\begin{cases} \dot{\boldsymbol{x}}(t) = \boldsymbol{f}(t, \boldsymbol{x}, \boldsymbol{u}^*), & \boldsymbol{x}(0) = \boldsymbol{x}_0, \\ \dot{\boldsymbol{\lambda}}(t) = -H_{\boldsymbol{x}}(t, \boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{u}^*), & \boldsymbol{\lambda}(t_f) = F_{\boldsymbol{x}}(\boldsymbol{x}(t_f)), \\ \dot{v}(t) = -L(t, \boldsymbol{x}, \boldsymbol{u}^*), & v(t_f) = F(\boldsymbol{x}(t_f)). \end{cases}$$

where $\boldsymbol{u}^* = \arg\min_{\boldsymbol{u}} H(t, \boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{u})$

1. Generate a first data set without initial guess (time-marching).
2. Train a neural network $V^{NN}(t, \boldsymbol{x})$.
3. Generate more data using warm start $\boldsymbol{\lambda}_0(t) = V^{NN}_{\boldsymbol{x}}(t, \boldsymbol{x})$.

Time-marching method

K	% BVP convergence	mean integration time
1	0.3%	0.37 s
2	38.7%	0.44 s
3	76.2%	0.40 s
4	92.9%	0.45 s
8	98.4%	0.53 s

Neural network warm start

μ	% BVP convergence	mean integration time
0	90%	0.44 s
10^{-3}	99.6%	0.41 s
10^{1}	100%	0.40 s

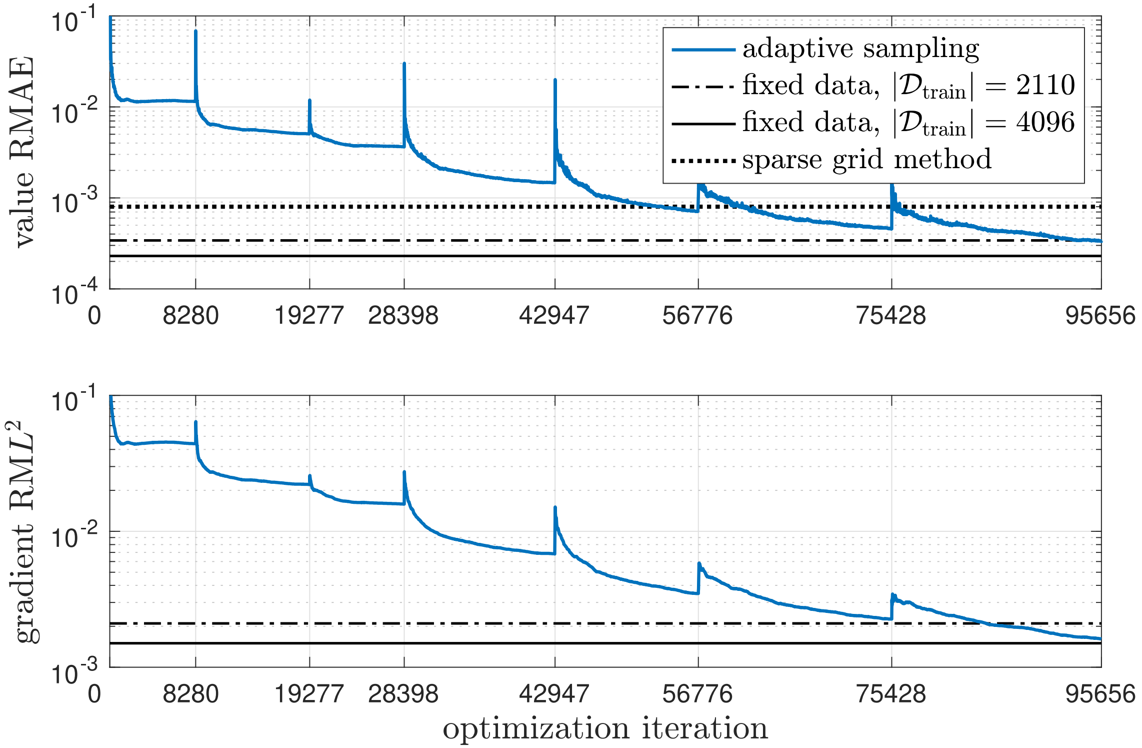
Rigid body optimal attitude control

NAVAL POSTGRADUATE SCHOOL

Example - rigid body optimal control

❖ T. Nakamura-Zimmerer, Q. Gong, W. Kang, *Adaptive deep learning for high-dimensional HJB equations*, SIAM J. Scientific Computing, 2021.

Neural network approximation of value function

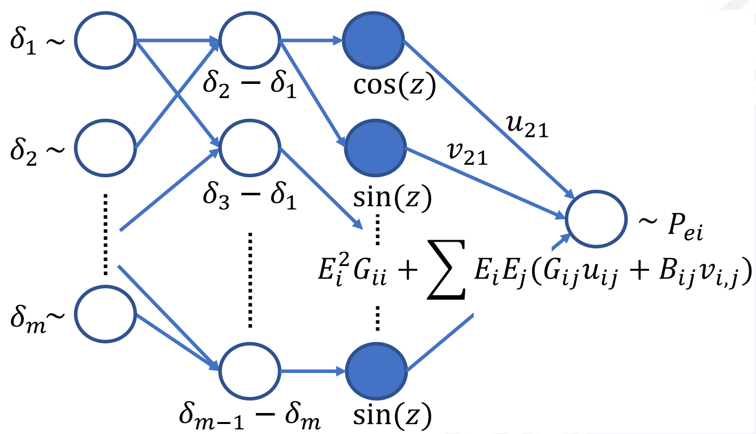


- Gradient loss weight $\mu = 10$; tolerance $\epsilon = 0.1$
- Training data set updated adaptively
- The final adaptive data set has 2110 points

Power system stability

Power system electric air-gap torque

$$(\mathbf{P}_e)_i = E_i^2 G_{ii} + \sum_{j=1, j\neq i}^{m} E_i E_j (G_{ij}\cos(\delta_i - \delta_j) + B_{ij}\sin(\delta_i - \delta_j)), \quad 1 \leq i \leq m,$$



$$\mathbf{r_{max}} = \mathbf{1},$$
$$\boldsymbol{\Lambda} = \mathbf{4}\pi,$$
$$|\mathcal{V}| = \mathbf{2}(\mathbf{N_g} - \mathbf{1})\mathbf{N_g},$$
$$\mathbf{L_{max}} = \max_{1\leq i,j\leq N_g, i\neq j}\left\{\frac{\omega_0}{\mathbf{2H_i}}\mathbf{E_iE_jG_{ij}}, \frac{\omega_0}{\mathbf{2H_i}}\mathbf{E_iE_jB_{ij}}\right\}$$

Power system domain of attraction

Theorem: Let $\mathcal{R} \subset \mathbb{R}^{2N_g}$ be a bounded set. Then, there exists a solution, $V(\mathbf{x})$, to Zubov's equation (a special Lyapunov function that characterizes the domain of attraction) and a neural network, $V^{NN}(\mathbf{x})$, that has n^{NN} hyperbolic tangent neurons. They satisfy

$$\left|V^{NN}(\mathbf{x}) - V(\mathbf{x})\right| < (C_1 N_g^2 + C_2)\frac{N_g}{\sqrt{n^{NN}}}$$

for $\mathbf{x} \in \mathcal{R}$, where C_1 and C_2 are constants independent of N_g.

Power system stability

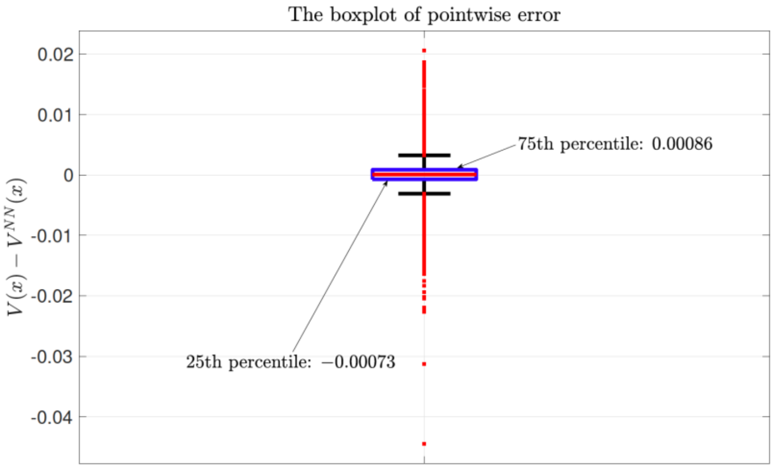
New England 10-generator 39-bus power system

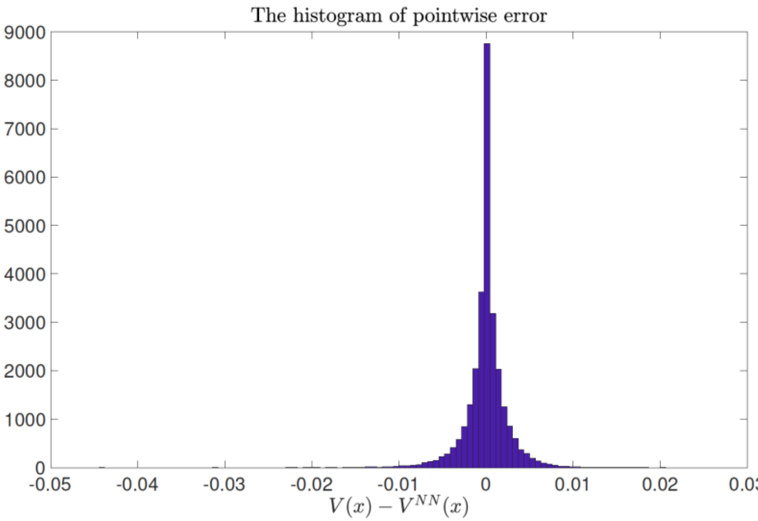
Characterization of the domain of attraction

--Neural network approximation of Zubov's equation

Activation:	hyperbolic tangent
Depth:	16
Width:	40

Error: $$RMSE = 2.0 \times 10^{-3}$$





Next step research

Future work

A lot more questions than answers

- The compositional features of nonsmooth problems, ReLU activation?
- How does compositional structure help to improve NN design and training?
- How does compositional structure help to validate a result, empirical risk vs population risk, L_2-norm vs infinity norm?
- Applications to dynamical systems (space dimension >5)
  - Deep filter and data assimilation (observability in unobservable systems)
  - Output regulation and FBI equation
  - Control Lyapunov function
  - Optimal control and viscosity solutions
  - Reachable set of control systems
  - The boundary of domain of attraction

THANK YOU

Problems appropriate for ML

What makes a good problem for machine learning?

- The overall problem does not have tractable solution.
  And
- You have access to lots of data.
  And
- The problem does not require highly accurate numerical solution (such as machine precision).

Give machine learning a try.

Deep learning for dynamics and control

Quantitative performance + **High dimension (not in normal form)** + **Real-time**

↓

Curse-of-dimensionality

Performance associated with PDEs

- HJB equation – optimal control
- HJI equation – differential game
- FBI equation – output regulation
- Zakai equation – optimal estimation
- Zubov's equation – the boundary of domain of attraction