Universität Bremen

Center for Industrial Mathematics (ZeTeM)

Faculty 03

Mathematics / Computer science

PCA-Networks

A Model Order Reduction Approach for Operator Learning

Janek Gödeke, Nick Heilenkötter, Tom Freudenberg

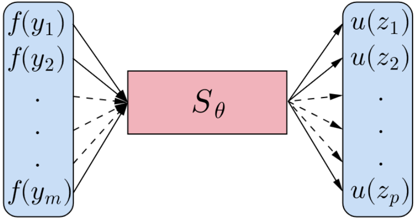
Renningen, 21.11.2025

Yesterday

One Neural Network S_θ for:

- Learning PDE-solutions u for diverse parameter functions f
- First example: solution operator $f \mapsto u$ of the problem

$$\partial_t u = f \quad \text{in } (0,1),$$
$$u(0) = 0.$$

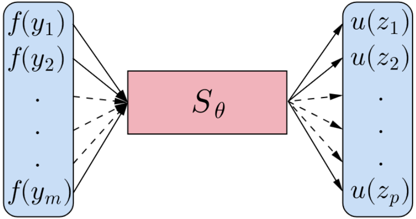


Yesterday

One Neural Network S_θ for:

- Learning PDE-solutions u for diverse parameter functions f
- First example: solution operator $f \mapsto u$ of the problem

$$\begin{aligned} \partial_t u &= f \quad \text{in } (0,1), \\ u(0) &= 0. \end{aligned}$$



Today: PCA-Nets and FNOs
$\rightarrow$ use more information about the problem

Foundation: The Concept of a Basis

$$\begin{pmatrix} 1.0 \\ 0.3 \\ -9.8 \end{pmatrix}$$

Foundation: The Concept of a Basis

$$\begin{pmatrix} 1.0 \\ 0.3 \\ -9.8 \end{pmatrix} = 1.0 \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + 0.3 \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} - 9.8 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

Foundation: The Concept of a Basis

Basis vectors

$$\begin{pmatrix} 1.0 \\ 0.3 \\ -9.8 \end{pmatrix} = 1.0 \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} + 0.3 \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} - 9.8 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

Basis coefficients

Foundation: The Concept of a Basis

Basis vectors

$$\begin{pmatrix} 1.0 \\ 0.3 \\ -9.8 \end{pmatrix} = 0.65 \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} + 0.35 \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix} - 9.8 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

Basis coefficients

Foundation: The Concept of a Basis

$$\begin{pmatrix} 1.0 \\ 0.3 \\ -9.8 \end{pmatrix} = 0.65 \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} + 0.35 \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix} - 9.8 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

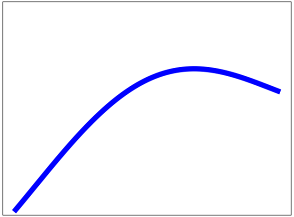
- This not only works for vectors, but also for functions!

Foundation: The Concept of a Basis

$$\begin{pmatrix} 1.0 \\ 0.3 \\ -9.8 \end{pmatrix} = 0.65 \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} + 0.35 \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix} - 9.8 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

This not only works for vectors, but also for functions!

Polynomial Basis: $f(x)$

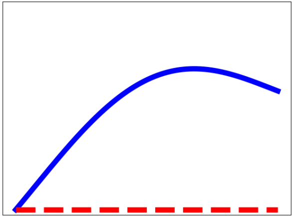


Foundation: The Concept of a Basis

$$\begin{pmatrix} 1.0 \\ 0.3 \\ -9.8 \end{pmatrix} = 0.65 \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} + 0.35 \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix} - 9.8 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

This not only works for vectors, but also for functions!

Polynomial Basis: $f(x) \approx f(0) \cdot 1$

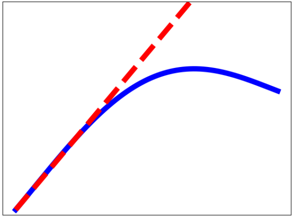


Foundation: The Concept of a Basis

$$\begin{pmatrix} 1.0 \\ 0.3 \\ -9.8 \end{pmatrix} = 0.65 \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} + 0.35 \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix} - 9.8 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

This not only works for vectors, but also for functions!

Polynomial Basis: $f(x) \approx f(0) \cdot 1 + f'(0)x$

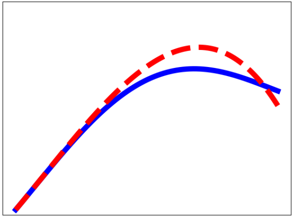


Foundation: The Concept of a Basis

$$\begin{pmatrix} 1.0 \\ 0.3 \\ -9.8 \end{pmatrix} = 0.65 \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} + 0.35 \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix} - 9.8 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

This not only works for vectors, but also for functions!

Polynomial Basis: $f(x) \approx f(0) \cdot 1 + f'(0)x + \frac{1}{2}f''(0)x^2$

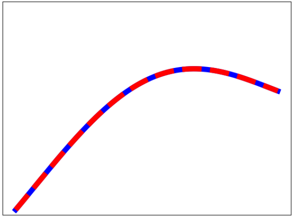


Foundation: The Concept of a Basis

$$\begin{pmatrix} 1.0 \\ 0.3 \\ -9.8 \end{pmatrix} = 0.65 \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} + 0.35 \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix} - 9.8 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

This not only works for vectors, but also for functions!

Polynomial Basis: $f(x) = f(0) \cdot 1 + f'(0)x + \frac{1}{2}f''(0)x^2 + \dots$



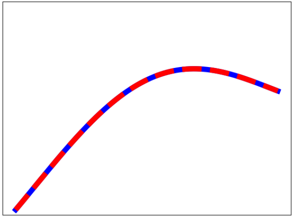
Foundation: The Concept of a Basis

$$\begin{pmatrix} 1.0 \\ 0.3 \\ -9.8 \end{pmatrix} = 0.65 \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} + 0.35 \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix} - 9.8 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

This not only works for vectors, but also for functions!

Polynomial Basis: $f(x) = f(0) \cdot 1 + f'(0)x + \frac{1}{2} f''(0)x^2 + \dots$

Polynomial basis:	1	x^1	x^2	x^3	$\dots$
Coefficient for f:	$f(0)$	$f'(0)$	$\frac{1}{2} f''(0)$	$\frac{1}{3!} f'''(0)$	$\dots$



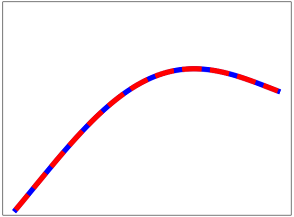
Foundation: The Concept of a Basis

$$\begin{pmatrix} 1.0 \\ 0.3 \\ -9.8 \end{pmatrix} = 0.65 \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} + 0.35 \begin{pmatrix} 1 \\ -1 \\ 0 \end{pmatrix} - 9.8 \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

This not only works for vectors, but also for functions!

Polynomial Basis: $f(x) = f(0) \cdot 1 + f'(0)x + \frac{1}{2}f''(0)x^2 + \dots$

Fourier Basis: $f(x) = c_0 + c_1 \cos(2\pi x) + c_2 \sin(2\pi x) + \dots$



Why Bases for Functions?

Polynomial basis:	1	x^1	x^2	x^3	x^4	x^5	$\dots$
Coefficient for f:	$f(0)$	$f'(0)$	$\frac{1}{2}f''(0)$	$\frac{1}{3!}f'''(0)$	$\frac{1}{4!}f^{(4)}(0)$	$\frac{1}{5!}f^{(5)}(0)$	$\dots$

Useful for compression:

Why Bases for Functions?

Polynomial basis:	1	x^1	x^2	x^3	x^4	x^5	$\dots$
Coefficient for f:	$f(0)$	$f'(0)$	$\frac{1}{2}f''(0)$	$\frac{1}{3!}f'''(0)$	$\frac{1}{4!}f^{(4)}(0)$	$\frac{1}{5!}f^{(5)}(0)$	$\dots$

Useful for compression:

- Given 5.000 functions f on grid of 100 points

  $\rightarrow$ Save 500.000 numbers!

Why Bases for Functions?

Truncated polynomial basis:	1	x^1	x^2	x^3
Coefficient for f:	$f(0)$	$f'(0)$	$\frac{1}{2}f''(0)$	$\frac{1}{3!}f'''(0)$

Useful for compression:

- Given 5.000 functions f on grid of 100 points

  $\rightarrow$ Save 500.000 numbers!

- Use first 20 basis functions to approximate f

  $\rightarrow$ Save only *basis + coefficients* $= 20 \cdot 100 + 20 \cdot 5000 = 102.000$ numbers!

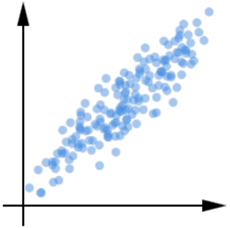
What are Good Bases?

- **Efficient compression:** Only few basis functions required
- **Computability:** Basis coefficients easy to compute
- **Noise-resilience**

Principle Component Analysis (PCA)

Given: Data functions f_j on N grid points

What is PCA?

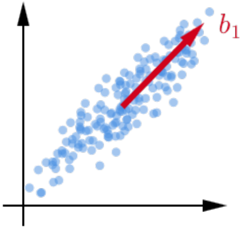


Principle Component Analysis (PCA)

Given: Data functions f_j on N grid points

What is PCA?

- First basis vector b_1 = direction of most variation

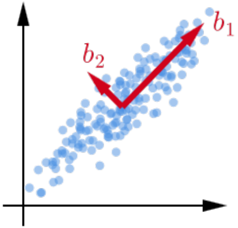


Principle Component Analysis (PCA)

Given: Data functions f_j on N grid points

What is PCA?

- First basis vector b_1 = direction of most variation
- Second basis vector b_2 = direction of second most variation

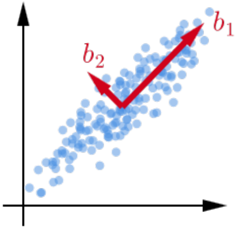


Principle Component Analysis (PCA)

Given: Data functions f_j on N grid points

What is PCA?

- First basis vector b_1 = direction of most variation
- Second basis vector b_2 = direction of second most variation
- PCA-basis $b_1, \ldots, b_N$

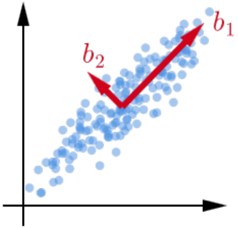


Principle Component Analysis (PCA)

Given: Data functions f_j on N grid points

What is PCA?

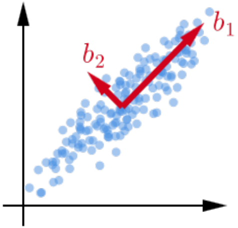
- First basis vector b_1 = direction of most variation
- Second basis vector b_2 = direction of second most variation
- PCA-basis $b_1, \ldots, b_N$



Compression: Select first $K < N$ principal components $b_1, \ldots, b_K$

Principle Component Analysis (PCA)

Given: Data functions f_j on N grid points



What is PCA?

- First basis vector b_1 = direction of most variation
- Second basis vector b_2 = direction of second most variation
- PCA-basis $b_1, \ldots, b_N$

Compression: Select first $K < N$ principal components $b_1, \ldots, b_K$

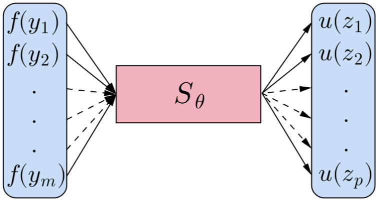
Basis coefficients easy to compute: $c_k = f \cdot b_k$

Reconstruction: $f \approx c_1 b_1 + \ldots + c_K b_K$

Benefit for Operator Learning

Reduce dimensionality of the problem:

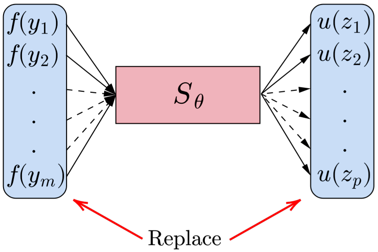
- Given discrete input function f on a 64×64 grid
- Default FCN has $64 \cdot 64 = 4096$ input neurons



Benefit for Operator Learning

Reduce dimensionality of the problem:

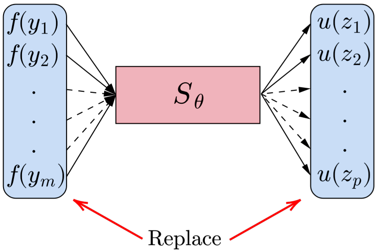
- Given discrete input function f on a 64×64 grid
- Default FCN has $64 \cdot 64 = 4096$ input neurons
- Replace samples $f(y_j)$ by $K \ll 4096$ PCA coeffs
  $\longrightarrow$ shorter training time



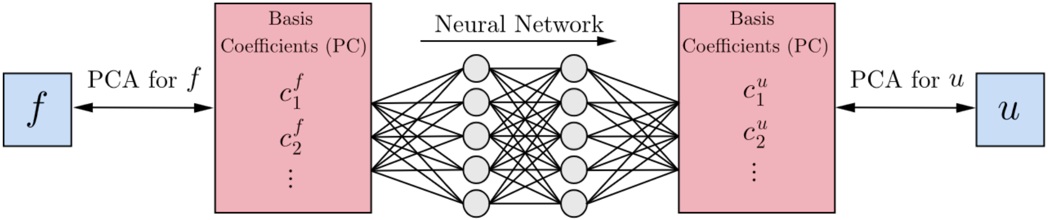
Benefit for Operator Learning

Reduce dimensionality of the problem:

- Given discrete input function f on a 64×64 grid
- Default FCN has $64 \cdot 64 = 4096$ input neurons
- Replace samples $f(y_j)$ by $K \ll 4096$ PCA coeffs
  $\longrightarrow$ shorter training time
- PCA also for output u

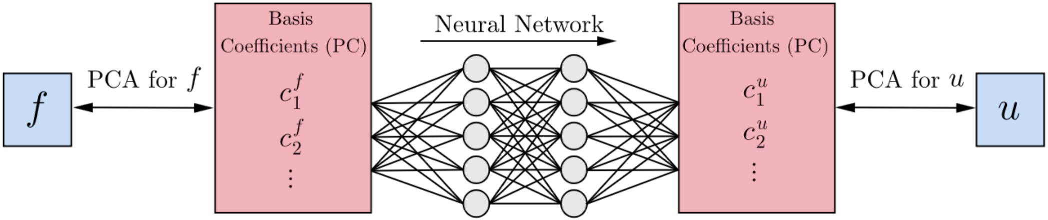


PCA-Networks[1]



[1] Bhattacharya et al, *Model reduction and neural networks for parametric PDEs*, 2020

PCA-Networks[1]



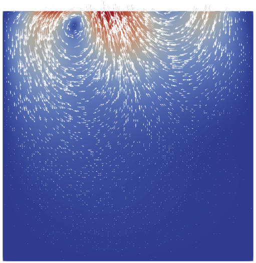
Additional advantages:

- PCA basis coeffients capture global information on f (and u)
- Reduction of noise (Exercise 7)

[1] Bhattacharya et al, *Model reduction and neural networks for parametric PDEs*, 2020

Using PCA-Networks in TorchPhysics

1. A joined exercise to see the general implementation:
   `Introduction_PCA-Nets.ipynb`
2. Solving the Stokes equations for different inflow profiles:
   `Exercise_6.ipynb`
3. Solving the inverse Allen-Cahn equation:
   `Exercise_7.ipynb`



Stokes solution