ELEDIA@UniTN - University of Trento
Dept. of Civil, Environmental, and Mechanical Engineering
Via Mesiano 77, I-38123 Trento, Italy
E-mail: contact@eledia.org
Web: www.eledia.org/eledia-unitn

Introduction to ML and AI Methods (Part I)

Dr. Marco SALUCCI

Day 1 - August 29th, 2022

2022 ELEDIA@ICAM PhD Summer Schools
Machine Learning & AI Methods
Theory, Techniques, and Advanced Engineering Applications
29 Aug. - 02 Sept. 2022, Trento, Italy (Onsite and Online)

Copyright Notice

All rights to the content of this document are reserved. Any use, in whole or in part, of the contents included in this document, including the storage, reproduction, editing, dissemination or distribution of their content through any technology platform, support, or computer network is forbidden without the prior written permission from the author.

Tutti i diritti relativi ai contenuti del presente documento sono riservati. È vietato qualsiasi utilizzo, totale o parziale, dei contenuti inseriti nel presente documento, ivi inclusa la memorizzazione, riproduzione, rielaborazione, diffusione o distribuzione dei contenuti stessi mediante qualunque piattaforma tecnologica, supporto o rete telematica, senza previa autorizzazione scritta da parte dell'autore.

Notation Guidelines

Text Style

- ***Question***
- **Definition**
- **Positive Outcome**
- **Positive observation**
- **Negative Outcome**
- **Negative observation**
- **Neutral Outcome**
- **Neutral observation**
- **Final Result**
- **Theorem [1]**
- ***Hypotheses* [1]**
- **Open Issue/Drawback**
- **Technique/Approach**
- Key-parameters (Test cases)
- **Problem Statement**
- **Other important sentence**

Slide Background

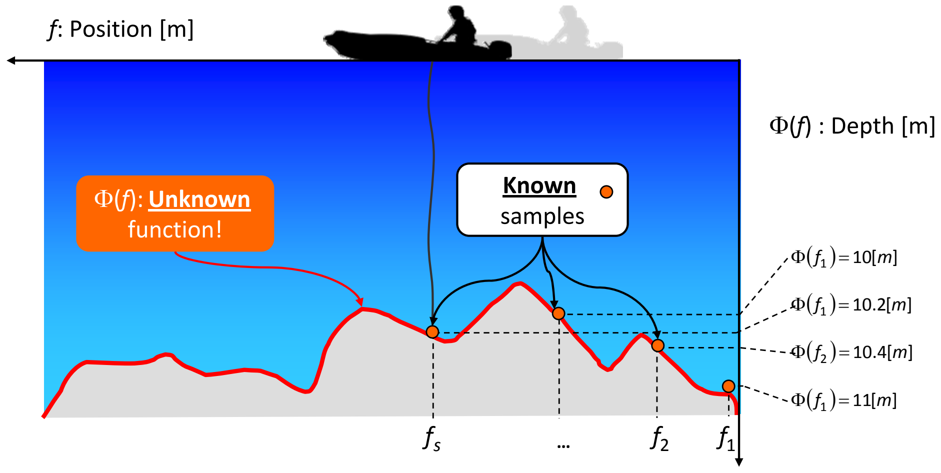
Light YELLOW – Numerical Examples	Light GRAY – Proofs/Maths
Light GREEN – Rules-of-Thumb	Light GBLU – Redundant
Light PINK – Resume/Shortening	Light ORANGE – Repetitions/Reminder
Light VIOLET – Glossary/Math Defs/Key	Light RED – Alternative Flow

Outline

- Introduction and Motivation
- Prediction Through **Interpolation Techniques**
  - ***Nearest Neighbor Interpolation***
  - ***Linear Interpolation***
- **3-Steps Learning-by-Examples** (***LBE***) Framework
  - Step 1: ***Dimensionality Reduction***
    - ***Sequential Feature Selection (SFT)***
    - ***Principal Component Analysis (PCA)***
    - ***Sammon Mapping***
    - ***Partial Least Squares (PLS)***

Introduction & Motivation (1/2)

What A men on a boat wants to measure **how deep** is a lake



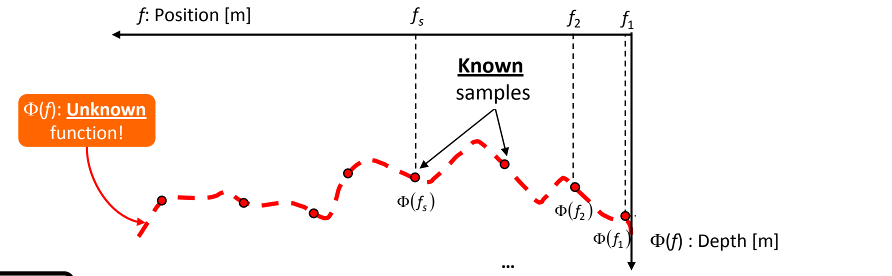
How He can **measure the depth** in a discrete number of **S positions**

$$\Rightarrow \{[f_i, \Phi(f_i)], i = 1, \dots, S\}$$

ELEDIA Research Center – All Rights Reserved

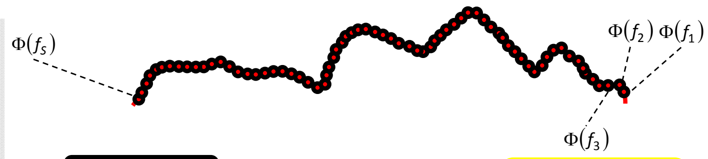
Introduction & Motivation (1/2)

How can he retrieve the depth $\Phi(f)$ for every position f ?



Approach Take **a lot** of very close consecutive **samples** of $\Phi(f)$

EXAMPLE:
Sample distance: 5 cm
Covered distance: 100 m
Required samples: 2000
Time per sample: 2 min
Total time: 2.78 days!!



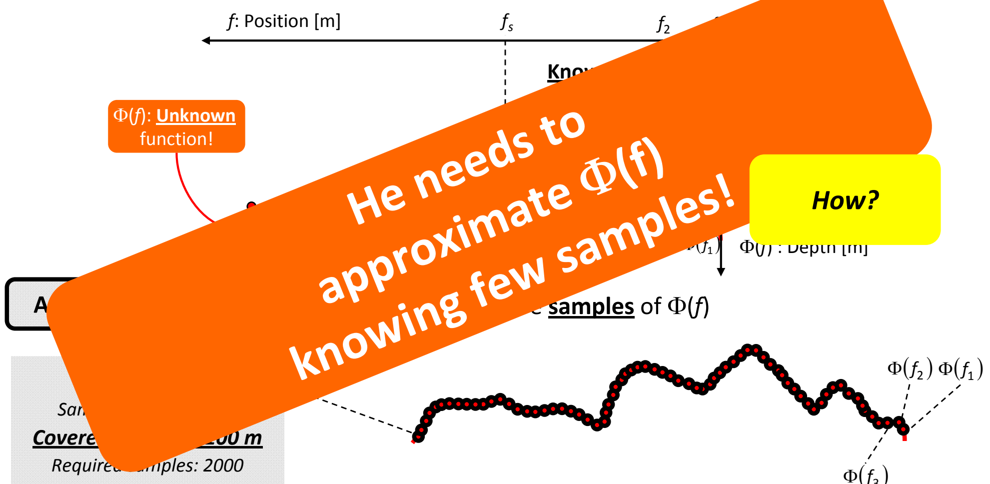
➡ **Issue** **Too expensive**!!

Faster Approach?

ELEDIA Research Center – All Rights Reserved

Introduction & Motivation (1/2)

How can he retrieve the depth $\Phi(f)$ for every position f ?



➡ **Issue** **Too expensive**!!

Faster Approach?

ELEDIA Research Center – All Rights Reserved

Efficient Evaluation of $\Phi(f)$

Open Issue Computationally-efficient methods

Fast Direct Solvers (small-medium problems)

- customized solvers with **limited** range of application ↓
- **no "training"** needed ↑
- can be applied for **small-medium** problems ↓

Approximate $\Phi(f)$ computation (large problems)

- $\Phi(f)$ is **predicted** ↑
- need of a **"training"** phase ↓
- can be applied **whatever** the problem size ↑

Which Approaches?

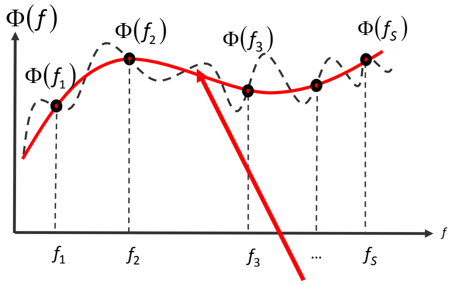
ELEDIA Research Center – All Rights Reserved

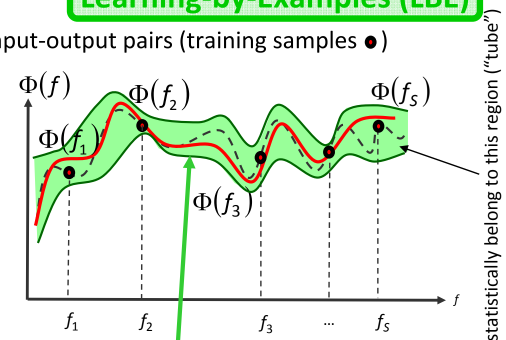
Efficient Evaluation of $\Phi(f)$

Approximate Φ(f) computation through a predictor ("Surrogate Model")

Interpolation | **Learning-by-Examples (LBE)**

$[\mathbf{f}_i; \Phi(\mathbf{f}_i)]\ i = 1,...,S$: Set of S **known** input-output pairs (training samples ●)





Approximation given by a **parametric function** with form is **postulated in advance** (e.g., polynomials, splines, ecc..)

Often implies exact fit at training points

Doesn't assume a parametric function a-priori but "**understands**" the **statistical behavior** of the underlying function.

Exact fit at training points not mandatory

ELEDIA Research Center – All Rights Reserved

Efficient Evaluation of $\Phi(f)$

Approximate Φ(f) computation through a predictor (large problems)

Interpolation

- Simple and efficient strategies ↑
- Requires to *a-priori* know/estimate functional dependency (e.g.: polynomial, exponential, etc.) ↓
- Low generalization capabilities ↓
- *Examples: **nearest neighbor interpolation**, **linear interpolation**, polynomial interpolation, spline interpolation*

Learning-by-Examples (LBE)

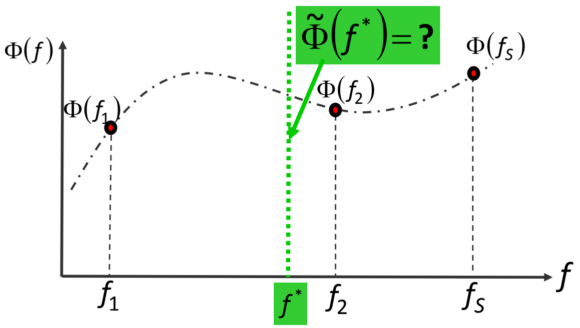
- Numerically efficient ↑
- No *a-priori* knowledge required ↑
- Good generalization capabilities ↑
- *Examples: Support Vector Machines (SVM), Artificial Neural Networks (ANN), Gaussian Processes (GP), Radial Basis Function (RBF)*

ELEDIA Research Center – All Rights Reserved

Prediction Through Interpolation

Given the function value $\Phi(f)$ at S N-dimensional samples, how to estimate the function value at f*?

EXAMPLE:
Number of variables: N=1



Solution **Interpolate** available data ("easiest" prediction)

How?
- **Nearest Neighbor (NN) interpolation** (A)
- Linear interpolation (B)

ELEDIA Research Center – All Rights Reserved

(A) Nearest Neighbor (NN) Interpolation

Approach The function value in **f*** is equal to the value assumed by the **nearest** training sample

Steps

(1) Compute the **Euclidean distance** between **f*** and $\mathbf{f}_i$, i=1,...,S

$$d(\mathbf{f}^*, \mathbf{f}_i) = \sqrt{\sum_{n=1}^{N} (f_{n,i} - f_n^*)^2}$$

N-dimensional

(2) Predict at **f***

$$\boxed{\tilde{\Phi}(\mathbf{f}^*)} = \Phi(\mathbf{f}_i) \quad \text{if} \quad d(\mathbf{f}^*, \mathbf{f}_i) < d(\mathbf{f}^*, \mathbf{f}_j) \quad \forall i, j = 1,...,S \quad i \neq j$$

ELEDIA Research Center – All Rights Reserved

(A) NN Interpolation - 1D Case

$$\tilde{\Phi}\left(\mathbf{f}^*\right)=\Phi\left(\mathbf{f}_i\right) \quad \text{if} \quad d\left(\mathbf{f}^*,\mathbf{f}_i\right)<d\left(\mathbf{f}^*,\mathbf{f}_j\right) \quad \forall i,j=1,\dots,S \quad i\neq j$$

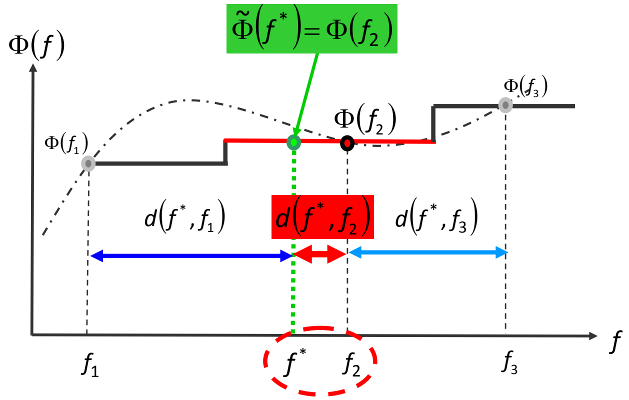
EXAMPLE:
Number of variables: N=1

$d\left(f^*,f_2\right)<d\left(f^*,f_1\right)$

$d\left(f^*,f_2\right)<d\left(f^*,f_3\right)$

The **nearest neighbor** of f^* is f_2

$\tilde{\Phi}\left(f^*\right)=\Phi\left(f_2\right)$

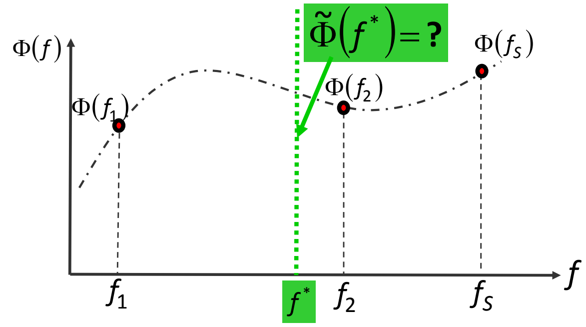


ELEDIA Research Center – All Rights Reserved

Prediction Through Interpolation

Given the function value $\Phi(f)$ at S N-dimensional samples, how to estimate the function value at f*?

EXAMPLE:
Number of variables: N=1



Solution **Interpolate** available data ("easiest" prediction)

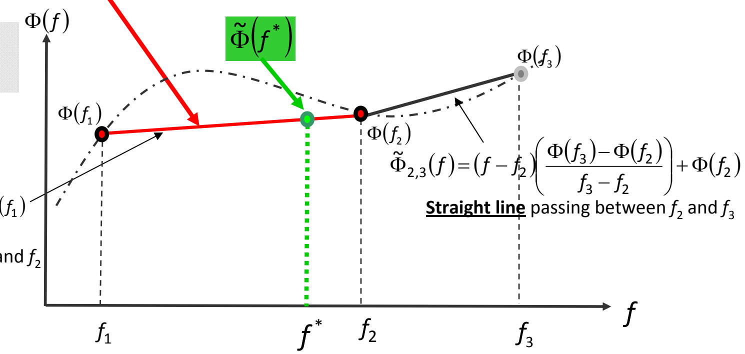
How?
- **Nearest Neighbor (NN) interpolation** (A)
- Linear interpolation (B)

ELEDIA Research Center – All Rights Reserved

(B) Linear Interpolation - 1D Case

Approach The function value in f^* is given by the equation of the **straight line** passing between the two neighboring points

EXAMPLE:
Number of variables: N=1



$\tilde{\Phi}_{1,2}(f)=\left(f-f_1\right)\left(\frac{\Phi(f_2)-\Phi(f_1)}{f_2-f_1}\right)+\Phi(f_1)$

Straight line passing between f_1 and f_2

$\tilde{\Phi}_{2,3}(f)=\left(f-f_2\right)\left(\frac{\Phi(f_3)-\Phi(f_2)}{f_3-f_2}\right)+\Phi(f_2)$

Straight line passing between f_2 and f_3

$$\boxed{\tilde{\Phi}\left(f^*\right)=\left(f^*-f_1\right)\left(\frac{\Phi(f_2)-\Phi(f_1)}{f_2-f_1}\right)+\Phi(f_1)}$$

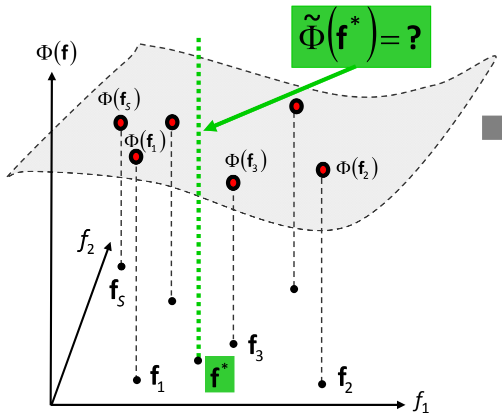
...And for N>1?

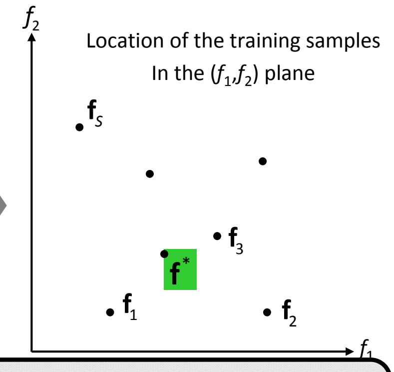
ELEDIA Research Center – All Rights Reserved

(B) Linear Interpolation - ND Case

Let be given a set of *S N*-dimensional training samples $\left[\mathbf{f}_i;\Phi\left(\mathbf{f}_i\right)\right] i=1,\dots,S$ •

EXAMPLE:
Number of variables: N=2





The prediction at f* will depend on the values assumed by its neighboring points

How to identify?

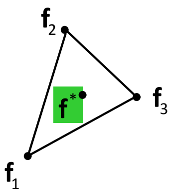
ELEDIA Research Center – All Rights Reserved

B ND Linear Interpolation - Steps

Approach The function value in **f*** is given by the **weighted sum** of its **neighboring points**

- Create the **Voronoi Diagram** $\mathcal{V}$(S)
- Create the **dual graph** $\mathcal{G}$(S) of $\mathcal{V}$(S)
- Convert $\mathcal{G}$ (S) to the **Delaunay diagram** $\mathcal{D}$ (S)
- Compute the **weights** α_i i=1,..,N+1
- Compute $\tilde{\Phi}(\mathbf{f}^*)$

Determine the **N+1 neighboring points** of **f***



Predict: $\tilde{\Phi}(\mathbf{f}^*) = \sum_{i=1}^{N+1} \alpha_i \Phi(\mathbf{f}_i)$

How?

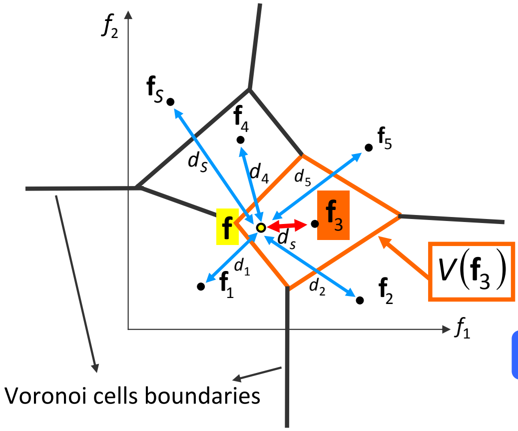
ELEDIA Research Center – All Rights Reserved

B Voronoi Cells/Diagrams

Def. **Voronoi cell** V($\mathbf{f}_i$) associated to sample $\mathbf{f}_i$:
Region of points whose Euclidean distance to $\mathbf{f}_i$ is lower than the distance to all other samples

EXAMPLE:
Number of variables: N=2

$$V(\mathbf{f}_i) = \{\mathbf{f} \mid d(\mathbf{f}, \mathbf{f}_i) \leq d(\mathbf{f}, \mathbf{f}_j)\, \forall i, j = 1, \dots, S\; i \neq j\}$$



f is closer to $\mathbf{f}_3$ than to all the other samples

$d_S < d_i \quad \forall i \neq 3$

$\Rightarrow \mathbf{f}_S \in V(\mathbf{f}_3)$

Def. **Voronoi diagram** $\mathcal{V}(S)$:
Complete set of S Voronoi cells

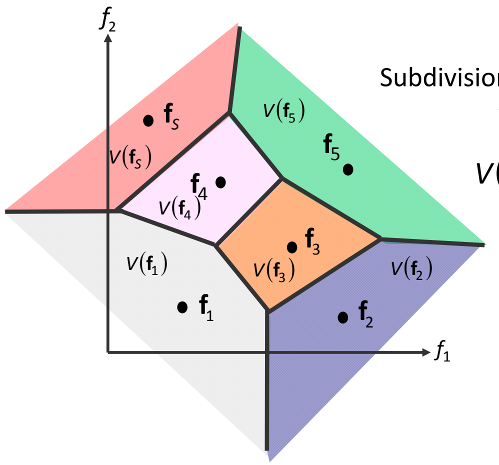
ELEDIA Research Center – All Rights Reserved

B ND Linear Interpolation - Step 1/5

- Create the **Voronoi Diagram** $\mathcal{V}$(S)
- Create the **dual graph** $\mathcal{G}$(S) of $\mathcal{V}$(S)
- Convert $\mathcal{G}$ (S) to the **Delaunay diagram** $\mathcal{D}$ (S)
- Compute the **weights** α_i i=1,..,N+1
- Compute $\tilde{\Phi}(\mathbf{f}^*)$

1 Create **Voronoi Diagram** $\mathcal{V}(S)$

EXAMPLE:
Number of variables: N=2



Subdivision of the plane into S Voronoi cells

$V(\mathbf{f}_i)\, \forall i = 1, \dots, S$

ELEDIA Research Center – All Rights Reserved

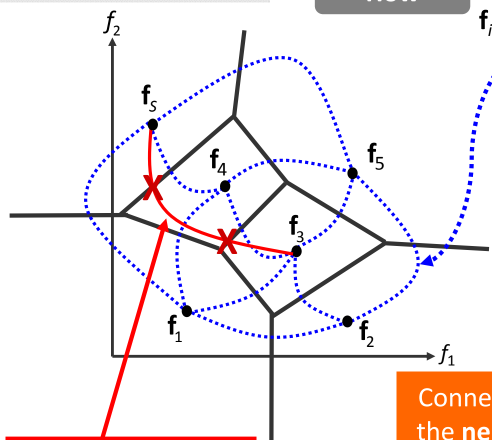
B ND Linear Interpolation - Step 2/5

- Create the **Voronoi Diagram** $\mathcal{V}$(S)
- Create the **dual graph** $\mathcal{G}$(S) of $\mathcal{V}$(S)
- Convert $\mathcal{G}$ (S) to the **Delaunay diagram** $\mathcal{D}$ (S)
- Compute the **weights** α_i i=1,..,N+1
- Compute $\tilde{\Phi}(\mathbf{f}^*)$

2 Create the **dual graph** $\mathcal{G}(S)$ of the Voronoi Diagram $\mathcal{V}(S)$

EXAMPLE:
Number of variables: N=2

How Connect two samples $\mathbf{f}_i$ and $\mathbf{f}_j$ if there exist a path crossing **only one Voronoi cell boundary**



Connection not allowed

(2 Voronoi buondaries crossed)

Connections will identify the **neighboring** samples for a given point **f***…

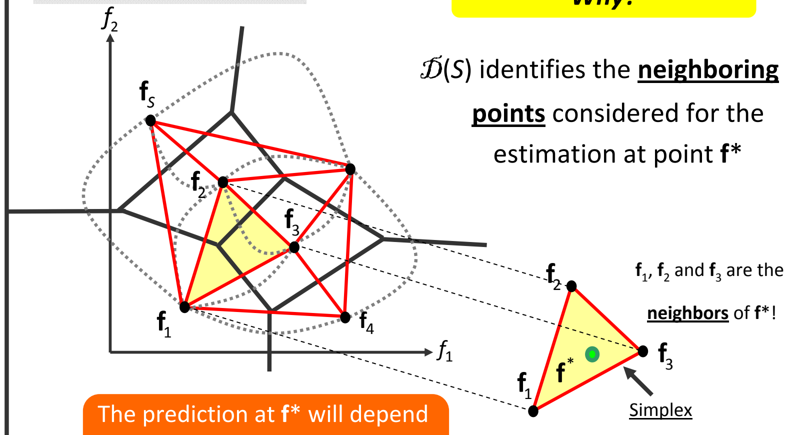
- All Rights Reserved

ⓑ ND Linear Interpolation - Step 3/5

- Create the **Voronoi Diagram** $\mathcal{V}(S)$
- Create the **dual graph** $\mathcal{G}(S)$ of $\mathcal{V}(S)$
- Convert $\mathcal{G}(S)$ to the **Delaunay diagram** $\mathcal{D}(S)$
- Compute the **weights** α_i i=1,..,N+1
- Compute $\tilde{\Phi}(\mathbf{f}^*)$

③ Create the **Delaunay Diagram** $\mathcal{D}(S)$: convert curved paths to **straight lines**

EXAMPLE:
Number of variables: N=2



Why?

$\mathcal{D}(S)$ identifies the **neighboring points** considered for the estimation at point **f***

The prediction at **f*** will depend on the values assumed by its **neighboring points**

ⓑ ND Linear Interpolation - Step 4/5

- Create the **Voronoi Diagram** $\mathcal{V}(S)$
- Create the **dual graph** $\mathcal{G}(S)$ of $\mathcal{V}(S)$
- Convert $\mathcal{G}(S)$ to the **Delaunay diagram** $\mathcal{D}(S)$
- Compute the **weights** α_i i=1,..,N+1
- Compute $\tilde{\Phi}(\mathbf{f}^*)$

④ Approximate the function at **f*** as a **weighted sum** of **N+1 independent samples** belonging to the same N-dimensional "simplex"

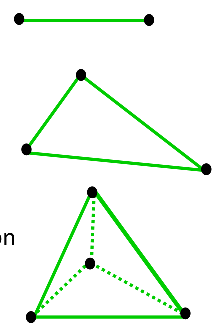
Prediction at **f***:

$$\tilde{\Phi}(\mathbf{f}^*) = \sum_{i=1}^{N+1} \alpha_i \Phi(\mathbf{f}_i) = \alpha_1 \Phi(\mathbf{f}_1) + \cdots + \alpha_{N+1} \Phi(\mathbf{f}_{N+1})$$

weights

Simplex

- N=1: Line
- N=2: Triangle
- N=3: Tetrahedron
- ...



How to **compute** the **weigths** α_n n=1,...,N+1?

ⓑ ND Linear Interpolation - Step 5/5

Weights Computation (1/2)

Def. **Volume** of a N-dimensional simplex: $Vol(\mathbf{f}_1,\ldots,\mathbf{f}_{N+1}) = \det\begin{pmatrix} \mathbf{f}_1 & \cdots & \mathbf{f}_{N+1} \\ 1 & \cdots & 1 \end{pmatrix}$

EXAMPLE:

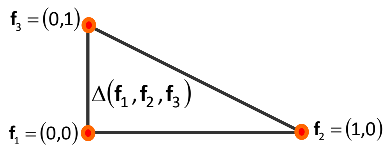
N=1

$$Vol(f_1,f_2) = \det\begin{pmatrix} f_1 & f_2 \\ 1 & 1 \end{pmatrix} = \boxed{f_1 - f_2} = -1$$

For N=1 volume = length

Signs depends on the ordering of the 2 points

N=2



$$Area(\Delta(\mathbf{f}_1,\mathbf{f}_2,\mathbf{f}_3)) = \frac{Base \cdot Height}{2} = \frac{1}{2}$$

$$Vol(\mathbf{f}_1,\mathbf{f}_2,\mathbf{f}_3) = \det\begin{pmatrix} \mathbf{f}_1 & \mathbf{f}_2 & \mathbf{f}_3 \\ 1 & 1 & 1 \end{pmatrix} = \det\begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 1 \end{pmatrix} = -1$$

$$Vol(\mathbf{f}_1,\mathbf{f}_2,\mathbf{f}_3) = \boxed{\pm 2 Area(\Delta(\mathbf{f}_1,\mathbf{f}_2,\mathbf{f}_3))}$$

Signs depends on the ordering of the 3 points

For N>1 volume ∝ area

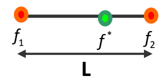
ⓑ ND Linear Interpolation - Step 5/5

Weights Computation (2/2)

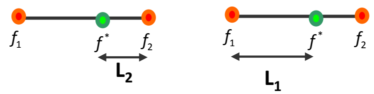
Weight for *i*-th neighbor $\mathbf{f}_i$: $\alpha_i = \dfrac{Vol(\mathbf{f}_1,\ldots,\mathbf{f}_{i-1},\mathbf{f}^*,\mathbf{f}_{i+1},\ldots,\mathbf{f}_{N+1})}{Vol(\mathbf{f}_1,\ldots,\mathbf{f}_{N+1})}$

EXAMPLE:

N=1



$$\alpha_1 = \frac{Vol(\mathbf{f}^*,\mathbf{f}_2)}{Vol(\mathbf{f}_1,\mathbf{f}_2)} = \frac{L_2}{L} \qquad \alpha_2 = \frac{Vol(\mathbf{f}_1,\mathbf{f}^*)}{Vol(\mathbf{f}_1,\mathbf{f}_2)} = \frac{L_1}{L}$$

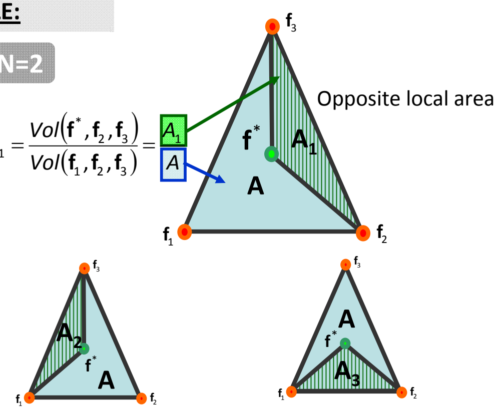


$$\boxed{\alpha_2 > \alpha_1}$$

$\tilde{\Phi}(f^*)$ will depend more on $\Phi(f_2)$

N=2

$$\alpha_1 = \frac{Vol(\mathbf{f}^*,\mathbf{f}_2,\mathbf{f}_3)}{Vol(\mathbf{f}_1,\mathbf{f}_2,\mathbf{f}_3)} = \frac{A_1}{A}$$



$$\alpha_2 = \frac{Vol(\mathbf{f}_1,\mathbf{f}^*,\mathbf{f}_3)}{Vol(\mathbf{f}_1,\mathbf{f}_2,\mathbf{f}_3)} = \frac{A_2}{A} \qquad \alpha_3 = \frac{Vol(\mathbf{f}_1,\mathbf{f}_2,\mathbf{f}^*)}{Vol(\mathbf{f}_1,\mathbf{f}_2,\mathbf{f}_3)} = \frac{A_3}{A}$$

B Linear Interpolation - Example

N=1

1. Volume calculation

$$Vol(\mathbf{f}_1,\mathbf{f}_2)=Vol(f_1,f_2)=\left|\det\begin{pmatrix} f_1 & f_2 \\ 1 & 1\end{pmatrix}\right|=f_2-f_1$$

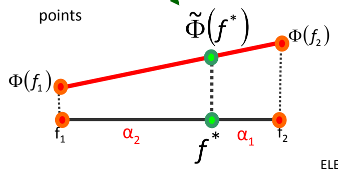
2. Weights calculation

$$\alpha_1=\frac{Vol(f^*,f_2)}{Vol(f_1,f_2)}=\frac{f_2-f^*}{f_2-f_1}\quad \alpha_2=\frac{Vol(f_1,f^*)}{Vol(f_1,f_2)}=\frac{f^*-f_1}{f_2-f_1}$$

3. Function approximation

$$\boxed{\tilde{\Phi}(f^*)}=\alpha_1\Phi(f_1)+\alpha_2\Phi(f_2)=(f^*-f_1)\left(\frac{\Phi(f_2)-\Phi(f_1)}{f_2-f_1}\right)+\Phi(f_1)$$

equation of a straight line passing between two points



N=2

1. Volume calculation

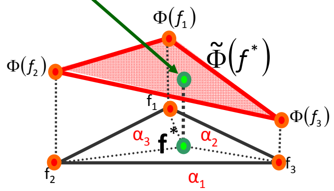
$$Vol(\mathbf{f}_1,\mathbf{f}_2,\mathbf{f}_3)=\left|\det\begin{pmatrix} \mathbf{f}_1 & \mathbf{f}_2 & \mathbf{f}_3 \\ 1 & 1 & 1\end{pmatrix}\right|=2Area(\Delta(\mathbf{f}_1,\mathbf{f}_2,\mathbf{f}_3))$$

2. Weights calculation

$$\alpha_1=\frac{Vol(\mathbf{f}^*,\mathbf{f}_2,\mathbf{f}_3)}{Vol(\mathbf{f}_1,\mathbf{f}_2,\mathbf{f}_3)}\quad \alpha_2=\frac{Vol(\mathbf{f}_1,\mathbf{f}^*,\mathbf{f}_3)}{Vol(\mathbf{f}_1,\mathbf{f}_2,\mathbf{f}_3)}\quad \alpha_3=\frac{Vol(\mathbf{f}_1,\mathbf{f}_2,\mathbf{f}^*)}{Vol(\mathbf{f}_1,\mathbf{f}_2,\mathbf{f}_3)}$$

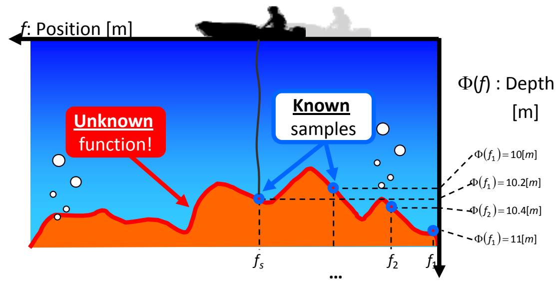
3. Function approximation

$$\boxed{\tilde{\Phi}(\mathbf{f}^*)}=\alpha_1\Phi(\mathbf{f}_1)+\alpha_2\Phi(\mathbf{f}_2)+\alpha_3\Phi(\mathbf{f}_3)$$

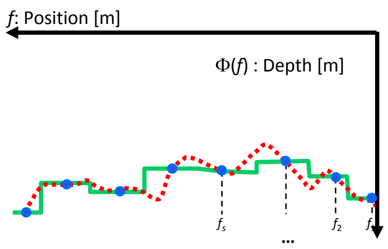


ELEDIA Research Center – All Rights Reserved

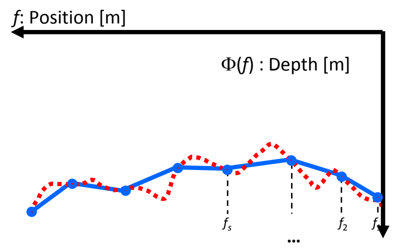
Nearest Neighbor Vs. Linear Interpolation (1/2)



Nearest Neighbor



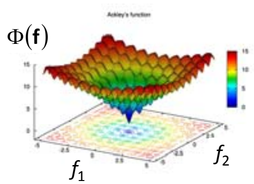
Linear Interpolation



ELEDIA Research Center – All Rights Reserved

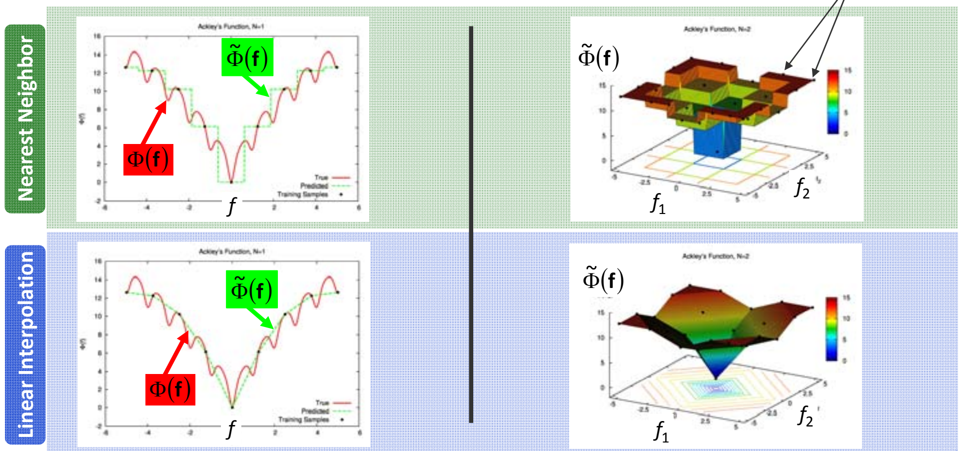
Nearest Neighbor Vs. Linear Interpolation (2/2)

Function: "Ackley"



N=1, S=9 **N=2, S=25**

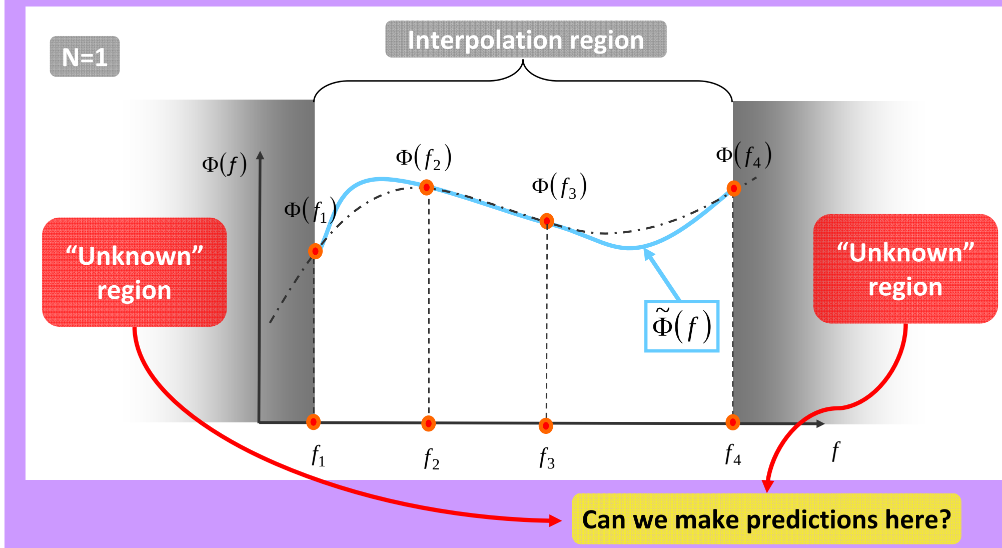
Training samples



ELEDIA Research Center – All Rights Reserved

Interpolation Vs. Extrapolation (1/2)

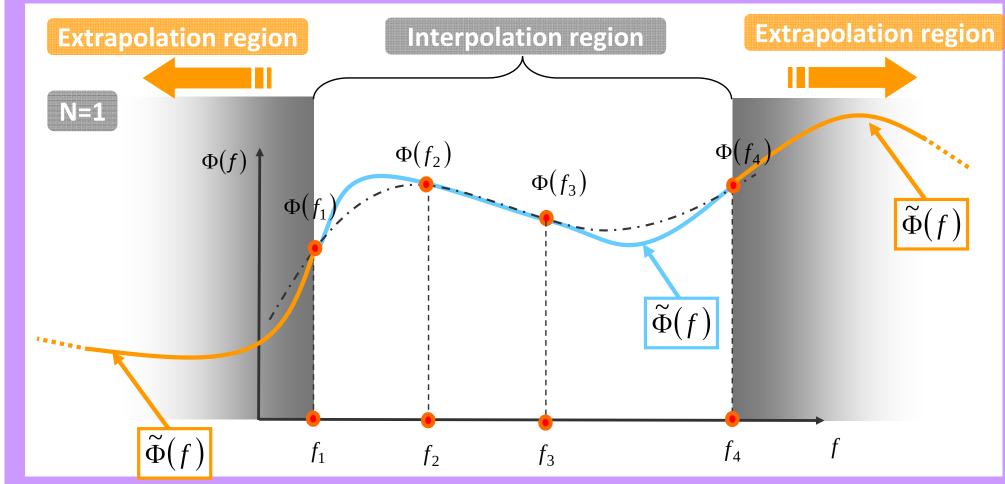
Interpolation Predict $\tilde{\Phi}(f)$ **between** observed points



ELEDIA Research Center – All Rights Reserved

Interpolation Vs. Extrapolation (2/2)

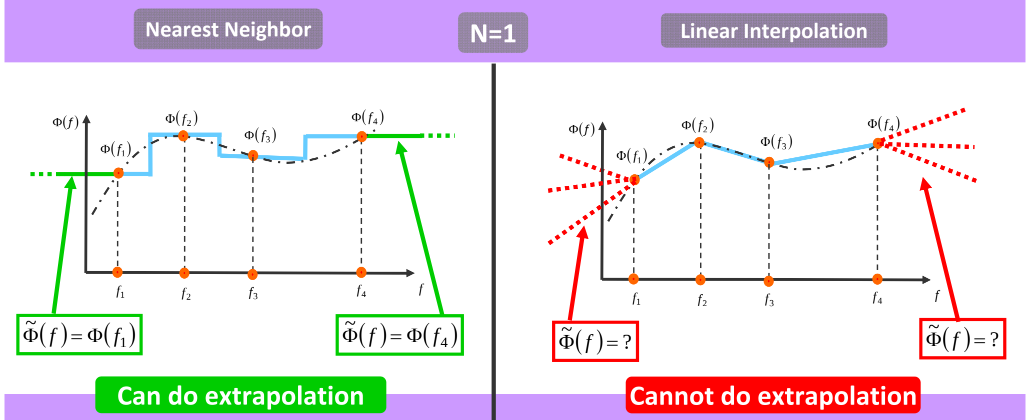
Extrapolation Predict $\tilde{\Phi}(f)$ **beyond** observed points



Always possible?

ELEDIA Research Center – All Rights Reserved

Extrapolation Capabilities



How to extrapolate?
- Nearest neighbor
- Extend the external straight lines

ELEDIA Research Center – All Rights Reserved

Efficient Evaluation of $\Phi(f)$

Approximate Φ(f) computation through a predictor (large problems)

Interpolation

- Simple and efficient strategies ↑
- Requires to *a-priori* know/estimate functional dependency (e.g.: polynomial, exponential, etc.) ↓
- Low generalization capabilities ↓
- *Examples: **nearest neighbor interpolation**, **linear interpolation**, polynomial interpolation, spline interpolation*

Learning-by-Examples (LBE)

- Numerically efficient ↑
- No *a-priori* knowledge required ↑
- Good generalization capabilities ↑
- *Examples: Support Vector Machines (SVM), Artificial Neural Networks (ANN), Gaussian Processes (GP), Radial Basis Function (RBF)*

ELEDIA Research Center – All Rights Reserved

How do LBE Methods Work?

Learning-By-Example Methods are formulated as two-step approaches

1 **Training** — Offline

f (known) → **LBE Algorithm** → Φ(f) (known)

Objective

Give the LBE method a set of known input-output relations (**[f, Φ(f)]** computed pairs) to train it

2 **Testing** — Online

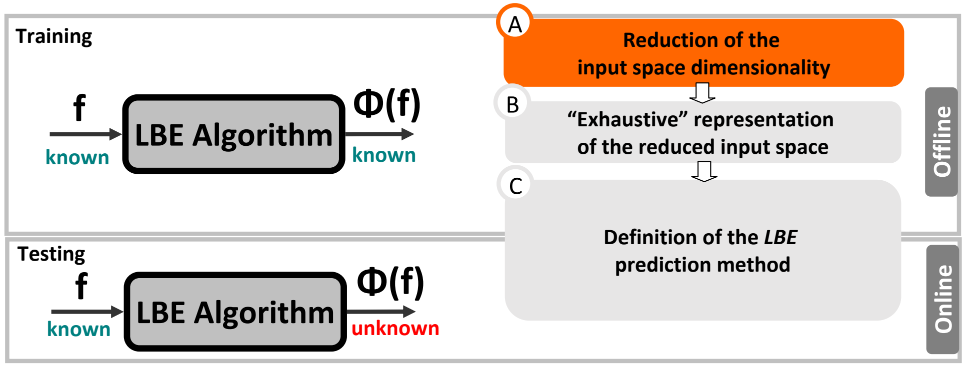
f (known) → **LBE Algorithm** → Φ(f) (unknown)

Objective

Give the LBE method a set of inputs [**f**] and let it predict the output [**Φ(f)**]

ELEDIA Research Center – All Rights Reserved

The 3-Steps LBE Framework



Problem How to deal with a <u>large number of variables</u> (N)?

Solution <u>**Reduce the DoFs of the functional space:**</u>
Look for the H<N variables showing the largest impact on the output ("*sensitivity analysis*")

How?

ELEDIA Research Center – All Rights Reserved

Reduction of the DoFs

Why? If the <u>**number of variables N is large**</u>, the number of required training samples <u>**S must be large**</u>

"Curse of dimensionality"

$N\uparrow \Rightarrow S\uparrow\uparrow\uparrow$

Num variables — Training size

- High <u>cost</u> of training phase
- High <u>learning complexity</u>

The number of required I/O samples (*S*) increases <u>**much faster**</u> than *N*!

How to mitigate it?

Map the N variables to a lower dimensionality space to reduce their number (and therefore, reduce S)

ELEDIA Research Center – All Rights Reserved

Mapping Samples to a Lower Dimensional Space

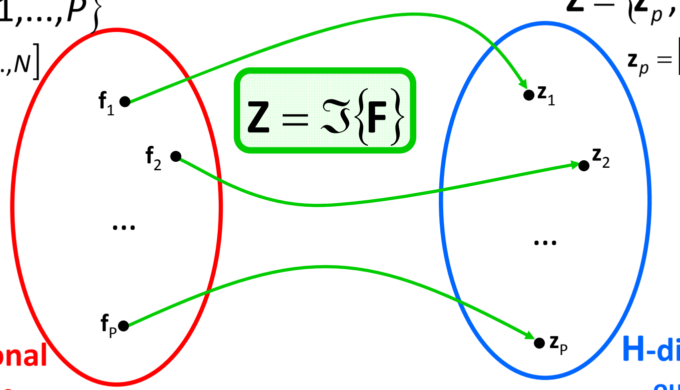
Approach Map P <u>**N-dimensional**</u> samples into P <u>**H-dimensional**</u> samples with **H<N**

$\mathbf{F}=\{\mathbf{f}_p, p=1,\ldots,P\}$

$\mathbf{f}_p=[f_{p,n}, n=1,\ldots,N]$

$\mathbf{Z}=\{\mathbf{z}_p, p=1,\ldots,P\}$

$\mathbf{z}_p=[z_{p,h}, h=1,\ldots,H]$



N-dimensional input space

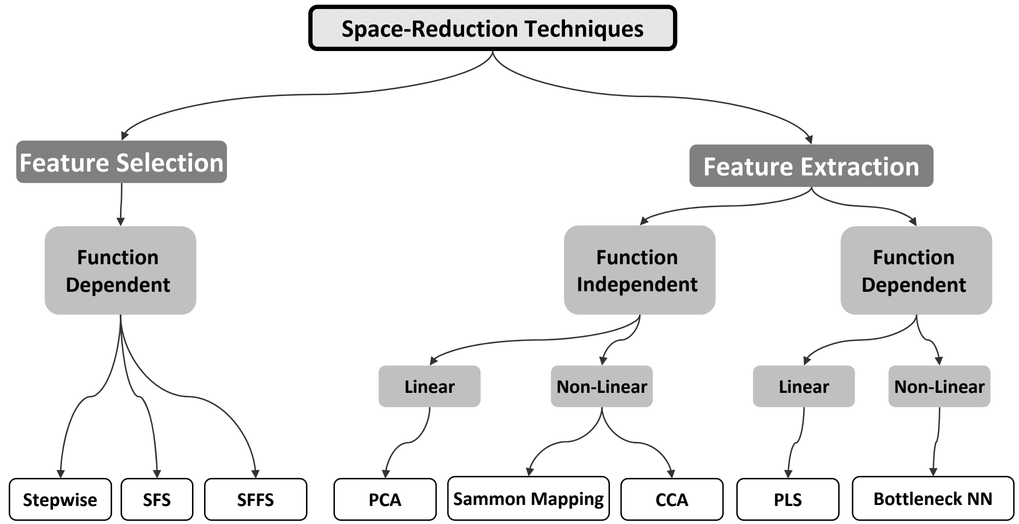
H-dimensional output space

H<N

How to map?

ELEDIA Research Center – All Rights Reserved

Space-Reduction: Overview



Selection Vs. Extraction?
Function Dependent Vs. Independent?
Linear Vs. Non-Linear?

ELEDIA Research Center – All Rights Reserved

Feature Selection vs. Feature Extraction

Feature Selection

Goal **Select** a **subset** of H < N **most relevant** variables (on Φ)

$$\mathbf{f}_p = \left[f_{p,1}, f_{p,2}, f_{p,3}, f_{p,4}, \ldots, f_{p,n}, \ldots f_{p,N} \right]$$

Less relevant on Φ | Less relevant on Φ

More relevant (informative) on Φ

$$\mathbf{z}_p = \left[z_{p,1}, z_{p,2}, \ldots, z_{p,h}, \ldots z_{p,H} \right]$$

Most informative measured features are kept

Sensitivity analysis required to understand which features must be selected

Feature Extraction

Goal **Generate** H **new variables** as a **combination** of the original ones

$$\mathbf{f}_p = \left[f_{p,1}, f_{p,2}, f_{p,3}, f_{p,4}, \ldots, f_{p,n}, \ldots f_{p,N} \right]$$

Feature "combination" $\mathbf{z}_p = \Im\left\{ \mathbf{f}_p \right\}$

$$\mathbf{z}_p = \left[z_{p,1}, z_{p,2}, \ldots, z_{p,h}, \ldots z_{p,H} \right]$$

All measured features used to extract ***new features***

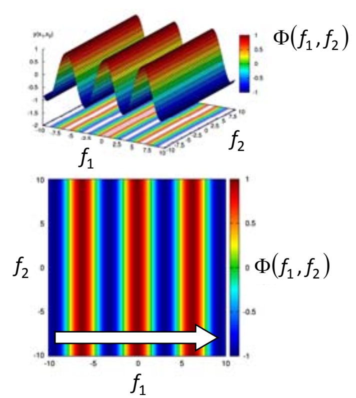
No sensitivity analysis required

ELEDIA Research Center – All Rights Reserved

When Feature Selection and When Extraction?

Feature Selection

Example $\Phi(f_1, f_2) = \cos(f_1)$



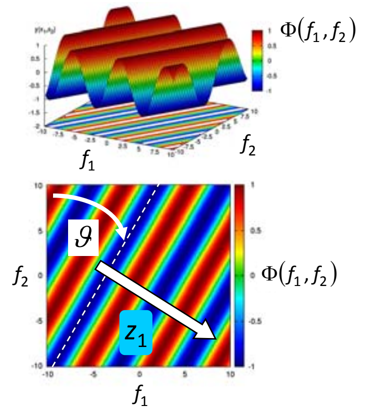
$\Phi(f_1, f_2)$ **depends only** on f_1

$\Phi(f_1, f_2) = \Phi(f_1)$

$$z_1 = f_1$$

Feature Extraction

Example $\Phi(f_1, f_2) = \cos\{f_1 \cos(\vartheta) - f_2 \sin(\vartheta)\}$



$\Phi(f_1, f_2)$ **can be expressed** as $\Phi(z_1)$

$$z_1 = \Im\{f_1, f_2\} = f_1 \cos(\vartheta) - f_2 \sin(\vartheta)$$

ELEDIA Research Center – All Rights Reserved

Function Independent vs. Function Dependent

Function Independent

Idea Reduction of the DoFs is based only on the **analysis** of the **input** samples

$\mathbf{F} = \{\mathbf{f}_p, p = 1, \ldots, P\}$ Old input variables

$$\mathbf{Z} = \Im\{\mathbf{F}\}$$

Most suitable for enhancing visualization of input data

Function Dependent

Idea Reduction of the DoFs is based on the **analysis** of the **input-output relationship**

Old input variables: $\mathbf{F} = \{\mathbf{f}_p, p = 1, \ldots, P\}$

Corresponding output: $\mathbf{\Phi} = \{\Phi(\mathbf{f}_p), p = 1, \ldots, P\}$

$$\mathbf{Z} = \Im\{\mathbf{F}; \mathbf{\Phi}\}$$

Most suitable for the prediction of Φ

$$\mathbf{f}_p = \left[f_{p,n}, n = 1, \ldots, N \right]$$
$$\mathbf{z}_p = \left[z_{p,h}, h = 1, \ldots, H \right]$$
$$\mathbf{Z} = \{\mathbf{z}_p, p = 1, \ldots, P\}$$

ELEDIA Research Center – All Rights Reserved

Linear vs. Non-Linear Feature Extraction

Linear Feature Extraction

Def. The new H variables are **linear combinations** of the old N variables

New variables / Old variables

$$\mathbf{Z} = \mathbf{F}\mathbf{W}$$

$[P \times H]$ $[P \times N]$ $[N \times H]$

Linear transformation matrix

Non-Linear Feature Extraction

Def. The new H variables are NOT linear combinations of the old N variables

New variables / Old variables

$$\mathbf{Z} = \Im\{\mathbf{F}\}$$

$[P \times H]$ $[P \times N]$

Non-linear transformation function

h-th component of *p*-th transformed sample

$$z_{p,h} = \sum_{n=1}^{N} f_{p,n} w_{n,h}$$

Linear combination of the old N variables

$$z_{p,h} = \Im\{f_{p,n}; n = 1 \ldots, N\}$$

P: Number of transformed samples

ELEDIA Research Center – All Rights Reserved

Sequential Feature Selection (SFS)

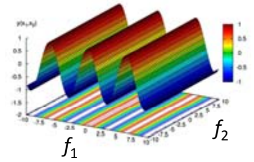
- Space-Reduction Techniques
  - **Feature Selection**
    - Function Dependent
      - Stepwise
      - **SFS**
      - SFFS
  - Feature Extraction
    - Function Independent
      - Linear
        - PCA
      - Non-Linear
        - Sammon Mapping
        - CCA
    - Function Dependent
      - Linear
        - PLS
      - Non-Linear
        - Bottleneck NN

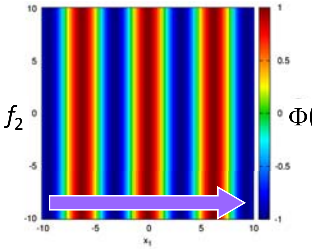
Sequential Feature Selection (SFS) - Basic Idea

Idea Given a set of N variables, select the subset of **H<N** variables having the largest impact on the function

Example $\Phi(f_1, f_2) = \cos(f_1)$

$\Phi(f_1, f_2)$ **depends only** on f_1 $\boxed{\Phi(f_1, f_2) = \Phi(f_1)}$





Select only f_1

$z_1 = f_1$

How to select f_1?

SFS – Block Diagram

Idea Given a set of N variables, select the subset of **H<N** variables having the largest impact on the function

Start from an empty set of selected variables

↓ q=0

Add the variable that optimizes singularly the selected criterion

↓ q=1

Add the variable that, together with those already selected, optimizes the selected criterion

↓ q=q+1

q==H? — NO → back to "Add the variable that, together with those already selected…"; YES → STOP

q=Number of selected variables

H=Number of variables to select

SFS – How to Select Variables (1/4)

Start from an empty set of selected variables

↓ q=0

Add the variable that optimizes singularly the selected criterion

↓ q=1

Add the variable that, together with those already selected, optimizes the selected criterion

↓ q=q+1

q==H? — NO → back; YES → STOP

EXAMPLE:

Dimension of the input space : N = 4

Number of variables to select: H = 3

$\Phi(f_1, f_2, f_3, f_4) = 10 f_2 + f_3 + \cos(f_4) + 0.1 f_1$

Less relevant on Φ | More relevant on Φ

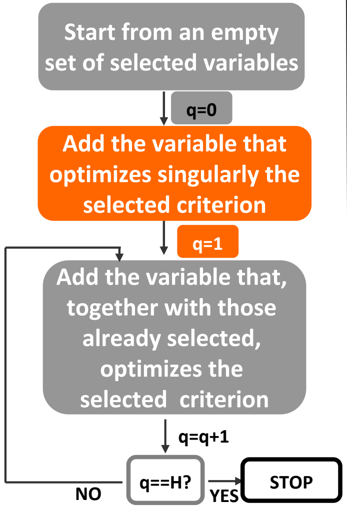
$\underbrace{f_1 \quad f_2 \quad f_3 \quad f_4}_{\text{Consider all possible variables}}$

Number of selected variables: q=0

$\mathbf{z} = \{\ \}$

Which is the first variable to select?

SFS – How to Select Variables (2/4)

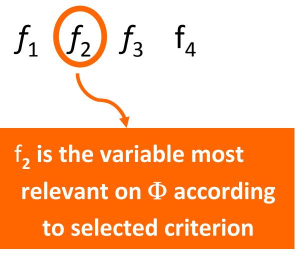


EXAMPLE:

Dimension of the input space : N = 4

Number of variables to select: H = 3

$$\Phi(f_1, f_2, f_3, f_4) = 10 f_2 + f_3 + \cos(f_4) + 0.1 f_1$$



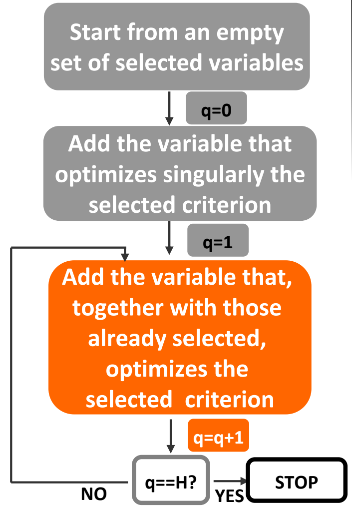
q=1

$$\mathbf{z} = \{f_2\}$$

How to select the remaining H-1 variables?

ELEDIA Research Center – All Rights Reserved

SFS – How to Select Variables (3/4)

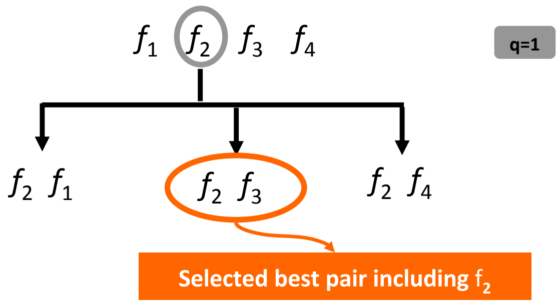


EXAMPLE:

Dimension of the input space : N = 4

Number of variables to select: H = 3

$$\Phi(f_1, f_2, f_3, f_4) = 10 f_2 + f_3 + \cos(f_4) + 0.1 f_1$$

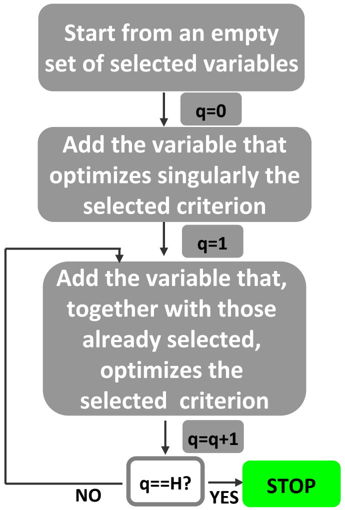


q=2

$$\mathbf{z} = \{f_2, f_3\}$$

ELEDIA Research Center – All Rights Reserved

SFS – How to Select Variables (4/4)

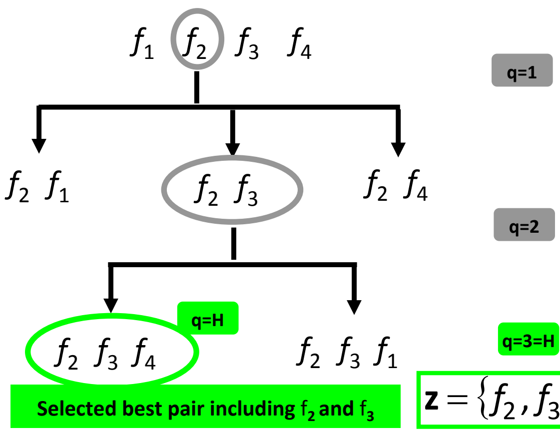


EXAMPLE:

Dimension of the input space : N = 4

Number of variables to select: H = 3

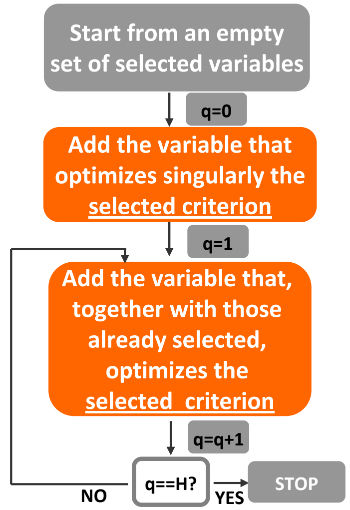
$$\Phi(f_1, f_2, f_3, f_4) = 10 f_2 + f_3 + \cos(f_4) + 0.1 f_1$$



$$\mathbf{z} = \{f_2, f_3, f_4\}$$

ELEDIA Research Center – All Rights Reserved

SFS – Selection Criterion?

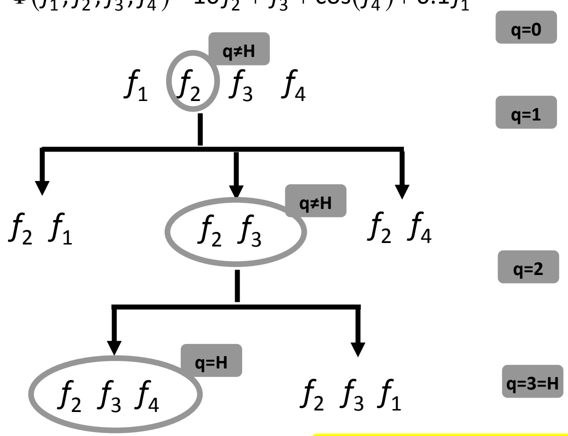


EXAMPLE:

Dimension of the input space : N = 4

Number of variables to select: H = 3

$$\Phi(f_1, f_2, f_3, f_4) = 10 f_2 + f_3 + \cos(f_4) + 0.1 f_1$$



Which criterion?

ELEDIA Research Center – All Rights Reserved

Feature Selection: Minimization of Criterion J(·)

EXAMPLE:
Dimension of the input space : N = 4
Number of variables to select: H = 3

$$\Phi(f_1, f_2, f_3, f_4) = 10 f_2 + f_3 + \cos(f_4) + 0.1 f_1$$

$\mathbf{f} = [f_1, f_2, f_3, f_4]$ → $\mathbf{z}_a = [f_1, f_2, f_3]$

$\mathbf{f} = [f_1, f_2, f_3, f_4]$ → $\mathbf{z}_b = [f_1, f_2, f_4]$

$\mathbf{f} = [f_1, f_2, f_3, f_4]$ → $\mathbf{z}_c = [f_1, f_3, f_4]$

$\mathbf{f} = [f_1, f_2, f_3, f_4]$ → $\mathbf{z}_d = [f_2, f_3, f_4]$

Criterion

$J(\mathbf{z}_a) = 7.5 \times 10^{-2}$ $\quad J(\mathbf{z}_b) = 5.4 \times 10^{-2}$ $\quad J(\mathbf{z}_c) = 8.9 \times 10^{-2}$ $\quad J(\mathbf{z}_d) = 3.6 \times 10^{-2}$

Lowest value of J(.)

$$\mathbf{z}_{opt} = \arg\left\{ \min_{\mathbf{z}_a, \mathbf{z}_b, \mathbf{z}_c, \mathbf{z}_d} (J(\mathbf{z})) \right\} = \mathbf{z}_d$$

How to comupte J(·)?

ELEDIA Research Center – All Rights Reserved

Feature Selection: How to Evaluate J(·)?

Goal

Compute the **<u>prediction error</u>** when using only the set of selected variables **z**

$$\mathbf{f} = [f_1, f_2, f_3, f_4]$$

$$\mathbf{z}_d = [f_2, f_3, f_4]$$

Prediction error obtained when considering variables f_2,f_3,f_4 and discarding variable f_1

$$J(\mathbf{z}_d) = 3.6 \times 10^{-2}$$

Approach: Cross-Validation

ELEDIA Research Center – All Rights Reserved

Computation of Prediction Error: Cross-Validation

Let's consider *P* *H*-dimensional samples, where $\mathbf{z}_p = \underbrace{[f_{p,2}, f_{p,3}, f_{p,4}]}_{\text{Example}}; p = 1,\dots,P$

1. Subdivide the P samples into **V subsets** of T samples

$$\mathbf{Z} = \underbrace{\begin{bmatrix} f_{1,2} & f_{1,3} & f_{1,4} \\ f_{2,2} & f_{2,3} & f_{2,4} \\ \vdots & \vdots & \vdots \\ f_{P,2} & f_{P,3} & f_{P,4} \end{bmatrix}}_{[P \times H]} \underbrace{\begin{bmatrix} \Phi_1 \\ \Phi_2 \\ \vdots \\ \Phi_P \end{bmatrix}}_{[P \times 1]}$$

SUBSET 1 (T samples) — **USE FOR VALIDATION**

SUBSET V-1 (T samples)
SUBSET V (T samples) — **USE FOR TRAIN**

2. For each v-th subset train a predictor using the remaining V-1 subsets

3. Use the resulting model to check prediction accuracy on the v-th subset of selected samples

ELEDIA Research Center – All Rights Reserved

Computation of Prediction Error: Cross-Validation

4. Compute Mean Squared Error (η) of the feature subset **z**

For each v-th validation subset (v=1,...,V)

$$\eta_v(\mathbf{z}) = \frac{1}{T} \sum_{i=1}^{T} \left\{ \Phi_{i,v} - \underbrace{\tilde{\Phi}_{i,v}(\mathbf{z})}_{\text{Predicted output}} \right\}^2$$

Example

$$\underbrace{\mathbf{z} = \{f_2, f_3, f_4\}}_{\text{Selected features}}$$

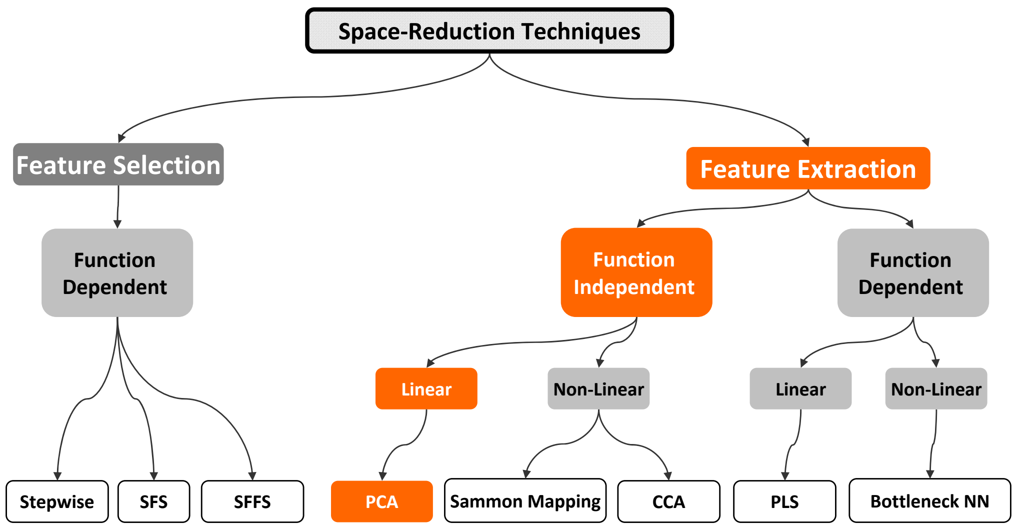
5. Average Mean Squared Error over all V validation subsets

Final criterion (to be minimized)

$$J(\mathbf{z}_d) = \underbrace{\boxed{\frac{1}{V} \sum_{v=1}^{V}}}_{\text{Average over all V subsets}} \frac{1}{T} \sum_{i=1}^{T} \left\{ \Phi_{i,v} - \tilde{\Phi}_{i,v}(\mathbf{z}) \right\}^2$$

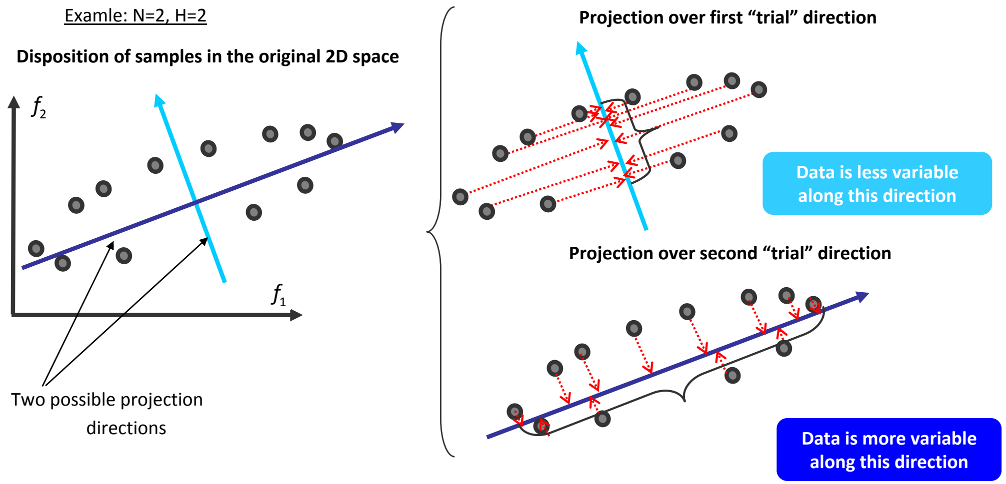
ELEDIA Research Center – All Rights Reserved

Principal Component Analysis (PCA)



PCA - Basic Idea (1/2)

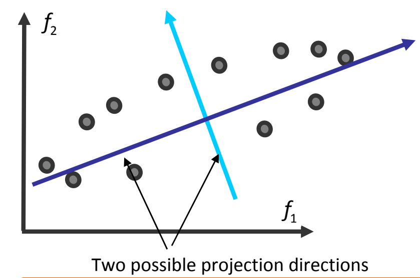
Idea Given P N-dimensional samples $\mathbf{F}=\{\mathbf{f}_p, p=1,\dots,P\}$ find the $H \le N$ **principal components** of data, i.e., the **directions** where there is the **largest variance**



PCA - Basic Idea (2/2)

Idea Given P N-dimensional samples $\mathbf{F}=\{\mathbf{f}_p, p=1,\dots,P\}$ find the $H \le N$ principal components of data, i.e., the **directions** where there is the **most variance**

Disposition of samples in the original 2D space



Which directions?

Compute the **N eigenvectors** of the **covariance matrix** in the original input space

How much variance?

Compute the **eigenvalues** associated to each eigenvector: larger eigenvalue means larger variance

Feature extraction: Project input samples onto the H<N eigenvectors {$\underline{W}_h$, h=1,…,H} showing the largest eigenvalues (the principal components)

$\underline{W}_h; h=1,\dots,H$ Basis of the J-dimensional subspace ***How to compute?***

PCA – Computation of The Basis

Linear Transformation

New variables, Old variables, Basis

$$\underset{[P\times H]}{\mathbf{Z}} = \underset{[P\times N]}{\mathbf{F}} \times \underset{[N\times H]}{\mathbf{W}} \qquad \underline{W} = \begin{bmatrix} w_{1,1} & w_{2,1} & \cdots & w_{H,1} \\ w_{1,2} & w_{2,2} & \cdots & w_{H,2} \\ \vdots & \vdots & \cdots & \vdots \\ w_{1,N} & w_{2,N} & \cdots & w_{H,N} \end{bmatrix}$$

$\underline{W}_1 \quad \underline{W}_2 \quad \dots \quad \underline{W}_H$

How to Compute? The basis vectors {$\underline{W}_h$; h=1,…,H} are the **eigenvectors** of the **covariance matrix** associated to the H **highest eigenvalues**

Covariance Matrix

$$\mathbf{C} = \frac{1}{P-1}\sum_{p=1}^{P}\left(\mathbf{f}_p - \overline{\mathbf{f}}\right)\left(\mathbf{f}_p - \overline{\mathbf{f}}\right)^T$$

Average vector

$$\overline{\mathbf{f}} = \frac{1}{S}\sum_{s=1}^{S}\mathbf{f}_p$$

Properties

- {$\underline{W}_h$; h=1,…,H} are called **principal components**
- {$\underline{W}_h$; h=1,…,H} are mutually **orthogonal**
- $\underline{W}_1$ has the **largest possible variance** (followed by $\underline{W}_2$, …, $\underline{W}_H$)

Covariance Matrix

$$\mathbf{C}=\begin{bmatrix} \mathrm{var}(f_1) & \mathrm{cov}(f_1,f_2) & \cdots & \mathrm{cov}(f_1,f_N) \\ \mathrm{cov}(f_2,f_1) & \mathrm{var}(f_2) & \cdots & \mathrm{cov}(f_2,f_N) \\ \vdots & \ddots & \ddots & \vdots \\ \mathrm{cov}(f_N,f_1) & \cdots & \mathrm{cov}(f_N,f_{N-1}) & \mathrm{var}(f_N) \end{bmatrix}$$

Diagonal: Variance of i-th (i=1,…,N) variables

$$\mathrm{var}(f_i)=\frac{1}{P-1}\sum_{p=1}^{P}\left(f_{p,i}-\bar{f}_i\right)^2$$

Average of i-th variable

$$\bar{f}_i=\frac{1}{P}\sum_{p=1}^{P}f_{p,i}$$

Off-Diagonal: Covariance between i-th and j-th variables

$$\mathrm{cov}(f_i,f_j)=\frac{1}{P-1}\sum_{p=1}^{P}\left(f_{p,i}-\bar{f}_i\right)\left(f_{p,j}-\bar{f}_j\right)$$

$\mathrm{cov}(f_i,f_j)=\mathrm{cov}(f_j,f_i) \Rightarrow$ **C** is symmetric

Meaning of cov(f_i,f_j)?

ELEDIA Research Center – All Rights Reserved

Covariance Between Two Variables

Covariance between i-th and j-th variables

$$\mathrm{cov}(f_i,f_j)=\frac{1}{P-1}\sum_{p=1}^{P}\left(f_{p,i}-\bar{f}_i\right)\left(f_{p,j}-\bar{f}_j\right) \qquad \mathrm{cov}(f_i,f_j)=\mathrm{cov}(f_j,f_i)$$

Meaning Measure of how much two variables f_i and f_j **change together**

- If f_j assumes **larger values** when f_i **increases**, covariance is **posivive**
- If f_j assumes **lower values** when f_i **increases**, covariance is **negative**

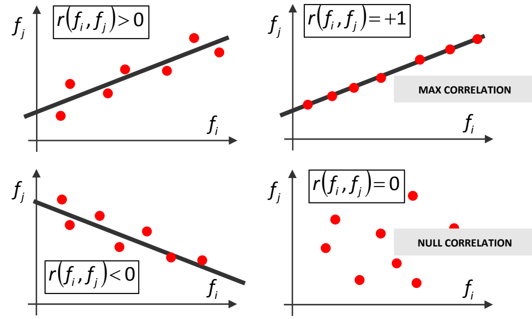
Correlation Coefficient r (Normalized Covariance)

$$r(f_i,f_j)=r(f_j,f_i)=\frac{\mathrm{cov}(f_i,f_j)}{std(f_i)\,std(f_j)}$$

$$-1\le r(f_i,f_j)\le +1$$

$$std(f_i)=\sqrt{\frac{1}{P-1}\sum_{p=1}^{P}\left(f_{p,i}-\bar{f}_i\right)^2}$$

Stadard deviation of i-th variable



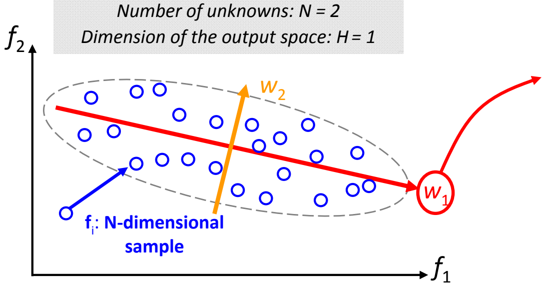
ELEDIA Research Center – All Rights Reserved

PCA – 2-D Example

EXAMPLE:

Number of unknowns: N = 2

Dimension of the output space: H = 1

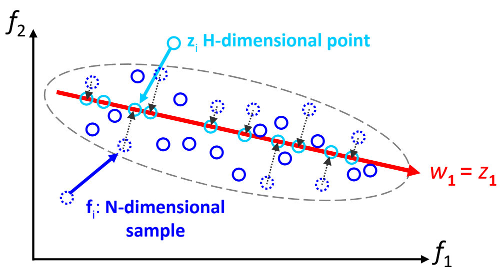


Eigenvector $\mathbf{w}_1$ corresponds to the highest eigenvalue

Greatest variance of the data lies on the first principal component $\mathbf{w}_1$

Project samples onto the principal component $\mathbf{w}_1$

- The total number of eigenvectors is equal to N
- The eigenvectors are mutually orthogonal



ELEDIA Research Center – All Rights Reserved

PCA - Example (1/7)

EXAMPLE:

Number of unknowns: N = 2

Number of samples: P = 10

Compute the covariance matrix C

↓

Compute the eigenvalues of C

↓

Compute the eigenvectors of C

↓

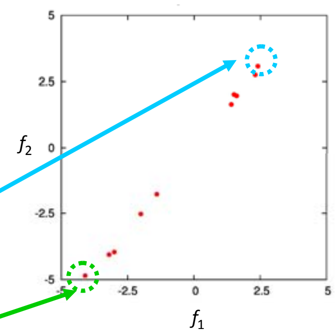
Project samples onto eigenvectors

Input 2-dimensional samples

$\mathbf{F} = [P\times N]$

Sample Index, *p*	f_1	f_2
1	1.4	1.65
2	1.6	1.975
3	-1.4	-1.775
4	-2.0	-2.525
5	-3.0	-3.95
6	2.4	3.075
7	1.5	2.025
8	2.3	2.75
9	-3.2	-4.05
10	-4.1	-4.85

Disposition of samples in the original space



ELEDIA Research Center – All Rights Reserved

PCA - Example (2/7)

EXAMPLE:
Number of unknowns: N = 2
Number of samples: P = 10

Compute the covariance matrix C → Compute the eigenvalues of C → Compute the eigenvectors of C → Project samples onto eigenvectors

$$\underset{[N\times N]}{\mathbf{C}}=\begin{bmatrix} Var(f_1) & Cov(f_1,f_2) \\ Cov(f_2,f_1) & Var(f_2) \end{bmatrix}$$

Variance of *i*-th variable

$$Var(f_i)=\frac{1}{P-1}\sum_{p=1}^{P}\left(f_{p,i}-\bar{f}_i\right)^2$$

Average of *i*-th variable

$$\bar{f}_i=\frac{1}{P}\sum_{p=1}^{P}f_{p,i}$$

Covariance between variables f_1 and f_2

$$Cov(f_1,f_2)=Cov(f_2,f_1)=\frac{1}{P-1}\sum_{p=1}^{P}\left(f_{p,1}-\bar{f}_1\right)\left(f_{p,2}-\bar{f}_2\right)$$

$$\bar{f}_1=-0.45 \qquad \bar{f}_2=-0.5675$$

$$Var(f_1)=6.4228$$
$$Var(f_2)=9.9528$$
$$Cov(f_1,f_2)=Cov(f_2,f_1)=7.9876$$

$$\Rightarrow \quad \mathbf{C}=\begin{bmatrix} 6.4228 & 7.9876 \\ 7.9876 & 9.9528 \end{bmatrix}$$

ELEDIA Research Center – All Rights Reserved

PCA - Example (3/7)

EXAMPLE:
Number of unknowns: N = 2
Number of samples: P = 10

Compute the covariance matrix C → **Compute the eigenvalues of C** → Compute the eigenvectors of C → Project samples onto eigenvectors

Find the eigenvalues of **C** by solving

$$\det\left(\mathbf{C}-\lambda\mathbf{I}\right)=0$$

$$\det\left(\mathbf{C}-\lambda\mathbf{I}\right)=\det\begin{bmatrix} Var(f_1)-\lambda & Cov(f_1,f_2) \\ Cov(f_2,f_1) & Var(f_2)-\lambda \end{bmatrix}=0$$

$$\mathbf{C}=\begin{bmatrix} 6.4228 & 7.9876 \\ 7.9876 & 9.9528 \end{bmatrix} \qquad Var(f_1)=6.4228 \quad Var(f_2)=9.9528$$

$$\lambda_1=16.368$$
$$\lambda_2=0.00746$$

Eigenvalues of the covariance matrix

Note The sum of the eigenvalues equals the sum of the variances

$$\lambda_1+\lambda_2=16.3756=Var(f_1)+Var(f_2)$$

ELEDIA Research Center – All Rights Reserved

PCA - Example (4/7)

EXAMPLE:
Number of unknowns: N = 2
Number of samples: P = 10

Compute the covariance matrix C → Compute the eigenvalues of C → **Compute the eigenvectors of C** → Project samples onto eigenvectors

For each *i*-th eigenvalue solve

$$\begin{cases} \left(\mathbf{C}-\lambda_i\mathbf{I}\right)\times\mathbf{w}_i=0 \Rightarrow \begin{bmatrix} Var(f_1)-\lambda_i & Cov(f_1,f_2) \\ Cov(f_2,f_1) & Var(f_2)-\lambda_i \end{bmatrix}\begin{bmatrix} w_{i,1} \\ w_{i,2} \end{bmatrix}=0 \\ \text{Subject to} \\ w_{i,1}^2+w_{i,2}^2=1 \end{cases}$$

Eigenvector associated to the *i*-th eigenvalue

$(\mathbf{C}-\lambda_1\mathbf{I})$ $\quad \lambda_1=16.368$

$$\begin{bmatrix} -9.9453 & 7.9876 \\ 7.9876 & -6.4153 \end{bmatrix}\begin{bmatrix} w_{1,1} \\ w_{1,2} \end{bmatrix}=0 \Rightarrow$$

First principal component

$$\underline{W}_1=\begin{bmatrix} 0.6262 \\ 0.7790 \end{bmatrix}$$

$$\begin{bmatrix} 6.41534 & 7.9876 \\ 7.9876 & 9.94534 \end{bmatrix}\begin{bmatrix} w_{2,1} \\ w_{2,2} \end{bmatrix}=0 \Rightarrow$$

$(\mathbf{C}-\lambda_2\mathbf{I})$ $\quad \lambda_2=0.00746$

Second principal component

$$\underline{W}_2=\begin{bmatrix} -0.7790 \\ 0.6262 \end{bmatrix}$$

ELEDIA Research Center – All Rights Reserved

PCA - Example (5/7)

EXAMPLE:
Number of unknowns: N = 2
Number of samples: P = 10

Compute the covariance matrix C → Compute the eigenvalues of C → Compute the eigenvectors of C → **Project samples onto eigenvectors**

Eigenvectors of the covariance matrix

$$\mathbf{Z}=\mathbf{F}_0\mathbf{W}=\left(\mathbf{F}-\bar{\mathbf{F}}\right)\left[\mathbf{w}_1 \quad \mathbf{w}_2\right]$$

Projected samples

Zero-mean input samples

$$\underset{[P\times H]}{\mathbf{Z}}=$$

$f_1-\bar{f}_1$	$f_2-\bar{f}_2$
1.85	2.2175
2.05	2.5425
-0.95	-1.2075
-1.55	-1.9575
-2.55	-3.3825
2.85	3.6425
1.95	2.5925
2.75	3.3175
-2.75	-3.4825
-3.65	-4.2825

×

$\mathbf{w}_1$	$\mathbf{w}_2$
0.6262	-0.7797
0.7797	0.6262

=

Projected samples

z_1	z_2
2.8874	-0.0538
3.2660	-0.0062
-1.5363	-0.0154
-2.4968	-0.0173
-4.2340	-0.1299
4.6246	0.0589
3.2424	0.1031
4.3086	-0.0667
-4.4372	-0.0366
-5.6245	0.1641

$[P\times H]$

$$\bar{f}_1=-0.45 \qquad \bar{f}_2=-0.5675$$

ELEDIA Research Center – All Rights Reserved

PCA - Example (6/7)

EXAMPLE:
Number of unknowns: N = 2
Number of samples: P = 10

$$\mathbf{Z} = \mathbf{F}_0\mathbf{W} = \left(\mathbf{F} - \overline{\mathbf{F}}\right)\left[\mathbf{w}_1 \quad \mathbf{w}_2\right]$$

Eigenvectors of the covariance matrix

Projected samples

Zero-mean input samples

Compute the covariance matrix C → Compute the eigenvalues of C → Compute the eigenvectors of C → **Project samples onto eigenvectors**

Z =

z_1	z_2
2.8874	-0.0538
3.2660	-0.0062
-1.5363	-0.0154
-2.4968	-0.0173
-4.2340	-0.1299
4.6246	0.0589
3.2424	0.1031
4.3086	-0.0667
-4.4372	-0.0366
-5.6245	0.1641

Projected samples

Note

$$Var(z_1) = 16.368 = \lambda_1$$
$$Var(z_2) = 0.00746 = \lambda_2$$

The variance of the projections onto the principal components is equal to the eigenvalues of the covariance matrix

The first eigenvector expresses around 99% of total variance

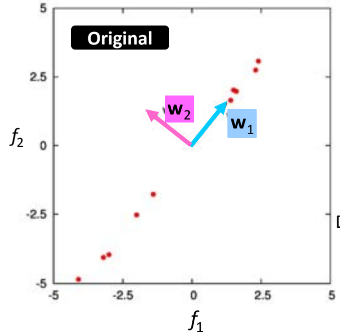
ELEDIA Research Center – All Rights Reserved

PCA - Example (7/7)

EXAMPLE:
Number of unknowns: N = 2
Number of samples: P = 10

Compute the covariance matrix C → Compute the eigenvalues of C → Compute the eigenvectors of C → **Project samples onto eigenvectors**

Disposition of samples in the original space (N=2)

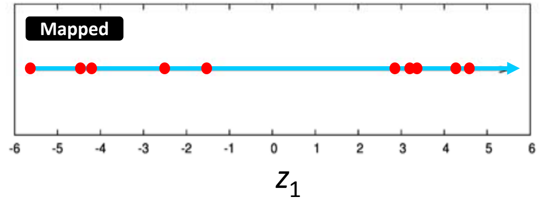


$\mathbf{w}_1$ is the **direction of maximum variance**

→ $\lambda_1 = 16.368$
$\lambda_2 = 0.00746$

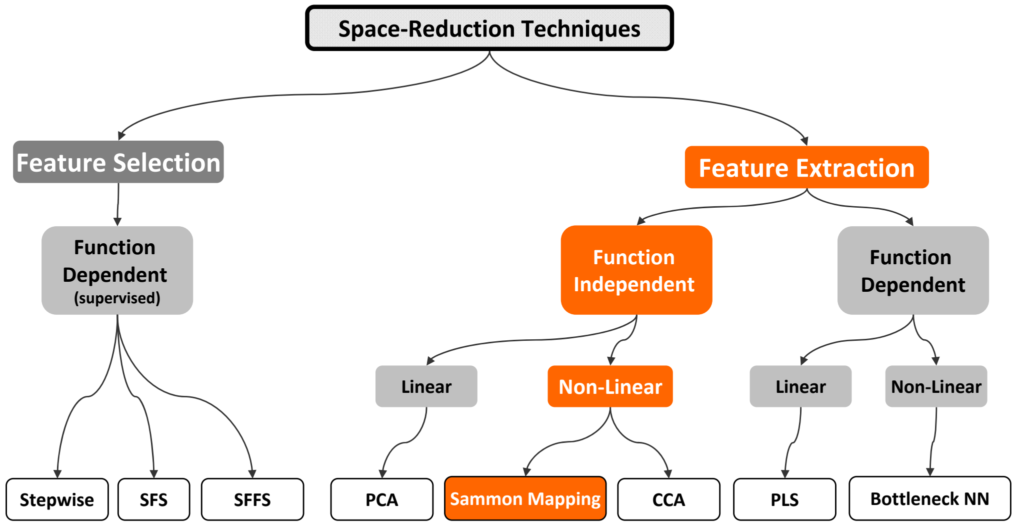
Data have a very limited variance along $\mathbf{w}_2$

Disposition of samples after projection onto first principal component (H=1)



ELEDIA Research Center – All Rights Reserved

Sammon Mapping



ELEDIA Research Center – All Rights Reserved

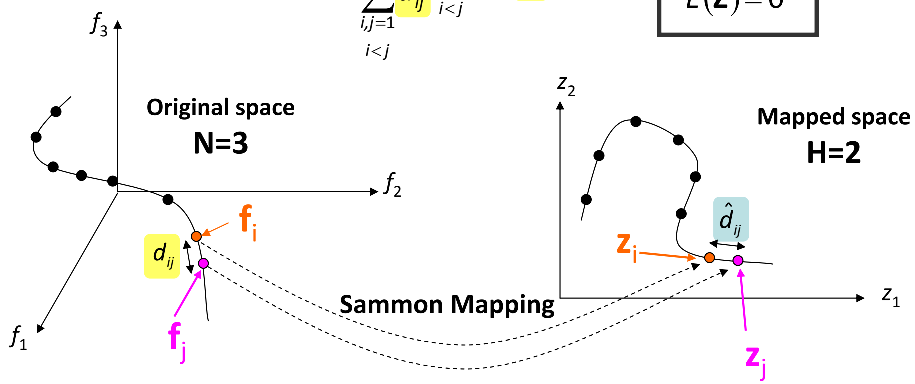
Sammon Mapping – Basic Idea

Goal **Preserve** the topological structure (**distance between samples**) between the two N-dimensional and H-dimensional (H<N) spaces

Mismatch between the distances in the two spaces (to be minimized)

$$E(\mathbf{Z}) = \frac{1}{\sum_{\substack{i,j=1 \\ i<j}}^{P} d_{ij}} \sum_{\substack{i,j=1 \\ i<j}}^{P} \frac{\left(d_{ij} - \hat{d}_{ij}\right)^2}{d_{ij}}$$

if $d_{ij} = \hat{d}_{ij}$ ⬇ $E(\mathbf{Z}) = 0$



$$d_{ij} = \sqrt{\sum_{n=1}^{N}\left(f_{n,i} - f_{n,j}\right)^2}$$

$$\hat{d}_{ij} = \sqrt{\sum_{h=1}^{H}\left(z_{h,i} - z_{h,j}\right)^2}$$

ELEDIA Research Center – All Rights Reserved

Sammon Mapping – Minimization of E(Z)

Goal **Minimize** the mismatch E(**Z**): $\mathbf{z}^{opt} = \arg\min_{\mathbf{z}} \{E(\mathbf{z})\} = \arg\min_{\mathbf{z}} \left\{ \frac{1}{\sum_{\substack{i,j=1\\ i<j}}^{P} d_{ij}} \sum_{\substack{i,j=1\\ i<j}}^{P} \frac{\left(d_{ij} - \hat{d}_{ij}\right)^2}{d_{ij}} \right\}$

How Gradient Descent Optimization

Initialization

$k=0 \quad \mathbf{Z}^{(0)}$

$k = k+1$

Update

$\mathbf{Z}^{(k+1)} = \mathbf{Z}^{(k)} + \Delta\mathbf{Z}$

How to Choose Z $\boldsymbol{^{(0)}}$***?***

ELEDIA Research Center – All Rights Reserved

Sammon Mapping - Initialization

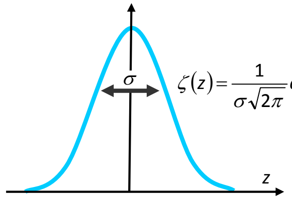
Initialization strategies for setting $\mathbf{Z}^{(0)}$

Random Initialization

Normal distribution with zero mean and variance equal to one

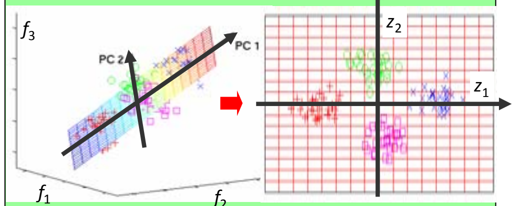
$$z_{p,h}^{(0)} = \mathcal{N}\left(\mu = 0; \sigma^2 = 1\right)$$

$h = 1,\ldots,H; \quad p = 1,\ldots,P$



$$\zeta(z) = \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-(z-\mu)^2}{2\sigma^2}}$$

Principal Component Analysis (PCA) Initialization

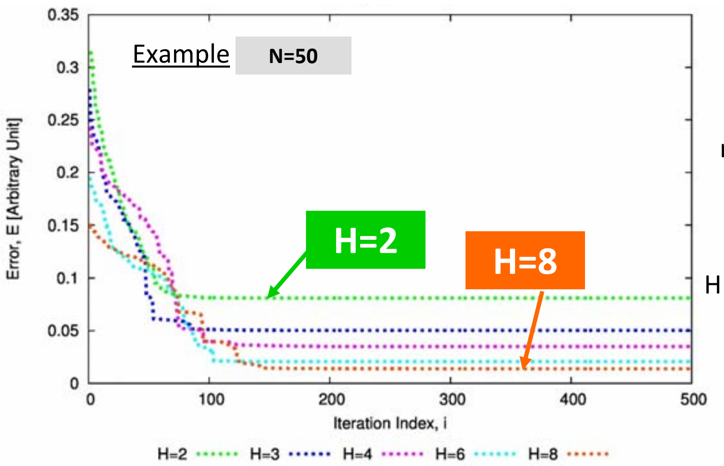


ELEDIA Research Center – All Rights Reserved

Sammon Mapping – Minimization of E(Z)

Goal **Minimize** the mismatch E(**Z**): $\mathbf{z}^{opt} = \arg\min_{\mathbf{z}} \{E(\mathbf{z})\} = \arg\min_{\mathbf{z}} \left\{ \frac{1}{\sum_{\substack{i,j=1\\ i<j}}^{P} d_{ij}} \sum_{\substack{i,j=1\\ i<j}}^{P} \frac{\left(d_{ij} - \hat{d}_{ij}\right)^2}{d_{ij}} \right\}$

How Gradient Descent Optimization



If H tends to N, the mismatch E tends to zero

H tipically user-defined

ELEDIA Research Center – All Rights Reserved

Partial Least Squares (PLS)

Space-Reduction Techniques

- **Feature Selection**
  - **Function Dependent** (supervised)
    - **Stepwise**
    - **SFS**
    - **SFFS**
- **Feature Extraction**
  - **Function Independent**
    - **Linear**
      - **PCA**
    - **Non-Linear**
      - **Sammon Mapping**
      - **CCA**
  - **Function Dependent**
    - **Linear**
      - **PLS**
    - **Non-Linear**
      - **Bottleneck NN**

ELEDIA Research Center – All Rights Reserved

PLS – Goal (1/2)

Goal Find a **new** set of **variables Z** such that the **input-output** relationship is **linearized** as much as possible

Zero-mean output · New variables · **Linear Approximation** $\tilde{\Phi}_0$

$$\Phi_0 = ZQ^T + E$$

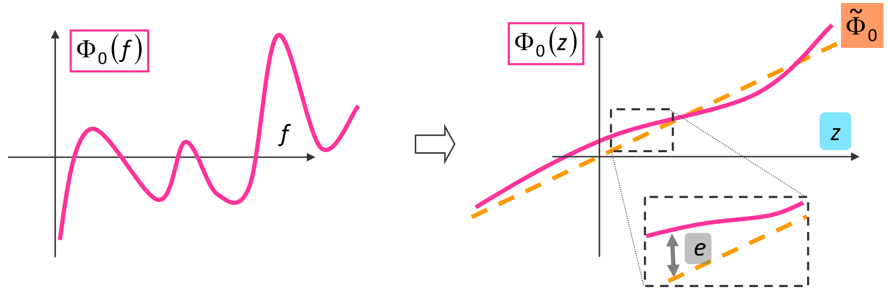
$$[P\times V] \quad [P\times H] \quad [H\times V] \quad [P\times V]$$

Approximation error $E = \Phi_0 - \tilde{\Phi}_0$

The lower the error, the better the approximation

P: Number of samples
H: # of new variables
V: Dimension of the output

Example: N=1, V=1, H=1



Input-output relation is **non-linear** w.r.t. the "old" variable f

Input-output relation is **almost linear** w.r.t. the "new" variable z

PLS – Goal (2/2)

Goal Find a **new** set of **variables Z** such that the **input-output** relationship is **linearized** as much as possible

Zero-mean output · New variables · **Linear Approximation** $\tilde{\Phi}_0$

$$\Phi_0 = ZQ^T + E$$

$$[P\times V] \quad [P\times H] \quad [H\times V] \quad [P\times V]$$

Approximation error $E = \Phi_0 - \tilde{\Phi}_0$

P: Number of samples
H: # of new variables
V: Dimension of the output

$$\begin{bmatrix} (\Phi_{1,1}-\bar{\Phi}_1) & (\Phi_{1,2}-\bar{\Phi}_2) & \cdots & (\Phi_{1,V}-\bar{\Phi}_V)\\ (\Phi_{2,1}-\bar{\Phi}_1) & (\Phi_{2,2}-\bar{\Phi}_2) & \cdots & (\Phi_{2,V}-\bar{\Phi}_V)\\ \vdots & \ddots & (\Phi_{p,v}-\bar{\Phi}_v) & \vdots\\ (\Phi_{P,1}-\bar{\Phi}_1) & \cdots & \ddots & (\Phi_{P,V}-\bar{\Phi}_V)\end{bmatrix} = \underbrace{\begin{bmatrix} z_{1,1} & z_{1,2} & \cdots & z_{1,H}\\ z_{2,1} & z_{2,1} & \cdots & z_{2,H}\\ \vdots & \ddots & z_{p,h} & \vdots\\ z_{P,1} & \cdots & \ddots & z_{P,H}\end{bmatrix}}_{H:\ \#\ \text{new variables}} \begin{bmatrix} q_{1,1} & q_{1,2} & \cdots & q_{1,V}\\ q_{2,1} & q_{2,1} & \cdots & q_{2,V}\\ \vdots & \ddots & q_{h,v} & \vdots\\ q_{H,1} & \cdots & \ddots & q_{H,V}\end{bmatrix} + \begin{bmatrix} e_{1,1} & e_{1,2} & \cdots & e_{1,V}\\ e_{2,1} & e_{2,1} & \cdots & e_{2,V}\\ \vdots & \ddots & e_{p,v} & \vdots\\ e_{P,1} & \cdots & \ddots & e_{P,V}\end{bmatrix}$$

v-th component of the function value for the p-th sample

$$\Phi_{p,v} - \bar{\Phi}_v = [\Phi_0(\mathbf{f}_p)]_v = \left(\sum_{h=1}^{H} z_{p,h} q_{h,v}\right) + e_{p,v}$$

Goal **Minimize** the approximation error **E**

How to find Z?

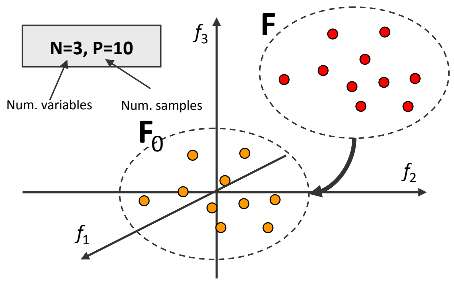
Centering F and Φ

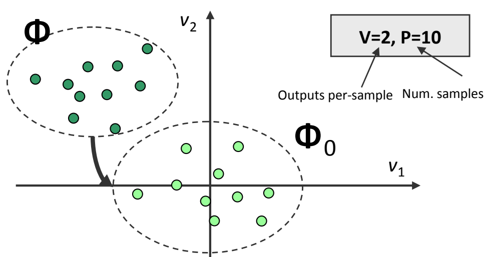
Centered input samples (N dimensional)

$$F_0 = (F - \bar{F}) = \{[(f_{p,n} - \bar{f}_n), n=1,\dots,N], p=1,\dots,P\}$$

Centered output values (V dimensional)

$$\Phi_0 = (\Phi - \bar{\Phi}) = \{[(\Phi_{p,v} - \bar{\Phi}_v), v=1,\dots,V], p=1,\dots,P\}$$





$$F_0 = \begin{bmatrix} (f_{1,1}-\bar{f}_1) & (f_{1,2}-\bar{f}_2) & \cdots & (f_{1,N}-\bar{f}_N)\\ (f_{2,1}-\bar{f}_1) & (f_{2,2}-\bar{f}_2) & \cdots & (f_{2,N}-\bar{f}_N)\\ \vdots & \ddots & (f_{p,n}-\bar{f}_n) & \vdots\\ (f_{P,1}-\bar{f}_1) & \cdots & \ddots & (f_{P,N}-\bar{f}_N)\end{bmatrix}$$

$$\Phi_0 = \begin{bmatrix} (\Phi_{1,1}-\bar{\Phi}_1) & (\Phi_{1,2}-\bar{\Phi}_2) & \cdots & (\Phi_{1,V}-\bar{\Phi}_V)\\ (\Phi_{2,1}-\bar{\Phi}_1) & (\Phi_{2,2}-\bar{\Phi}_2) & \cdots & (\Phi_{2,V}-\bar{\Phi}_V)\\ \vdots & \ddots & (\Phi_{p,v}-\bar{\Phi}_v) & \vdots\\ (\Phi_{P,1}-\bar{\Phi}_1) & \cdots & \ddots & (\Phi_{P,V}-\bar{\Phi}_V)\end{bmatrix}$$

Average of i-th variable (i=1,...,N) over P samples

$$\bar{f}_i = \frac{1}{P}\sum_{p=1}^{P} f_{p,i}$$

Average of v-th output (v=1,...,V) over P samples

$$\bar{\Phi}_v = \frac{1}{P}\sum_{p=1}^{P} \Phi_{p,v}$$

PLS – Extract The New Variables (Z)

Idea Find a **linear** transformation matrix $W = \Psi\{F;\Phi\}$ such that

Input · Output

(Zero-mean) old variables

New variables

$$Z = F_0 W$$

$$[P\times H] \quad [P\times N] \quad [N\times H]$$

P: Number of transformed samples
N: # old variables
H: # new variables

$$F_0 = (F - \bar{F}) = \{[(f_{p,n} - \bar{f}_n), n=1,\dots,N], p=1,\dots,P\}$$

Average of each n-th variable (n=1,...,N)

Such that the input-output relationship becomes **linear** w.r.t. **Z**

Value of the h-th new variable for the p-th sample

$$z_{p,h} = \sum_{n=1}^{N} f^0_{p,n} w_{n,h}$$

Value of the n-th old variable for the p-th sample (zero-mean)

The new variables are linear combinations of the old ones

How to find W?

Linear Decomposition of Input and Output Matrices

Basic HP Both input ($\mathbf{F}_0$) and output ($\mathbf{\Phi}_0$) can be expressed by **linear approximations**

Input

$$\underset{[P\times N]}{\mathbf{F}_0} = \underset{[P\times H]}{\mathbf{Z}}\underset{[H\times N]}{\mathbf{A}^T} + \underset{[P\times N]}{\mathbf{L}}$$

Output

$$\underset{[P\times V]}{\mathbf{\Phi}_0} = \underset{[P\times H]}{\mathbf{U}}\underset{[H\times V]}{\mathbf{Q}^T} + \underset{[P\times V]}{\mathbf{G}}$$

Errors made by the linear approximation

W is such that the transformed variables **(Z)** have **maximum correlation** with the transformed output **(U)**

⇓

If **Z** and **U** are **correlated**, we will be able to estimate the output directly from **Z**

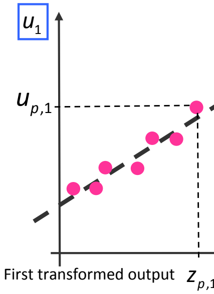
$$\mathbf{\Phi}_0 = \mathbf{Z}\mathbf{Q}^T + \mathbf{E}$$

PLS – Approach

Approach **Maximize** the **covariance** between corresponding columns of **Z** and **U**

Input $\mathbf{F}_0 = \mathbf{Z}\mathbf{A}^T + \mathbf{L}$

Output $\mathbf{\Phi}_0 = \mathbf{U}\mathbf{Q}^T + \mathbf{G}$



$\mathbf{z}_1$ has maximum covariance with $\mathbf{u}_1$,

$\mathbf{z}_2$ has maximum covariance with $\mathbf{u}_2$,

...

$\mathbf{z}_1$ First transformed variable

$\mathbf{u}_1$ First transformed output

$$\mathbf{Z} = \begin{bmatrix} z_{1,1} & z_{1,2} & \cdots & z_{1,H} \\ z_{2,1} & z_{2,2} & \cdots & z_{2,H} \\ \vdots & & \ddots & \vdots \\ z_{P,1} & \cdots & \ddots & z_{P,H} \end{bmatrix} \qquad \mathbf{U} = \begin{bmatrix} u_{1,1} & u_{1,2} & \cdots & u_{1,H} \\ u_{2,1} & u_{2,2} & \cdots & u_{2,H} \\ \vdots & & \ddots & \vdots \\ u_{P,1} & \cdots & \ddots & u_{P,H} \end{bmatrix}$$

P: # of samples

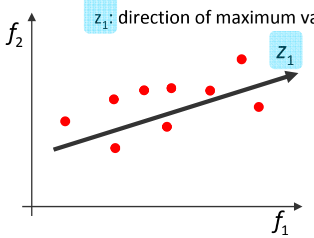
Difference w.r.t PCA?

PLS vs. PCA

PCA

PCA finds the **directions of maximum variation** in **F**

z_1: direction of maximum variance in **F**

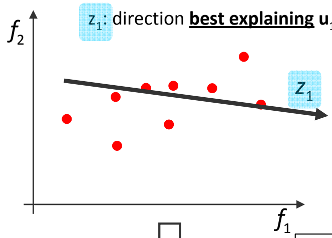


If applied to **Φ** it would find the directions of maximum variation in **Φ**

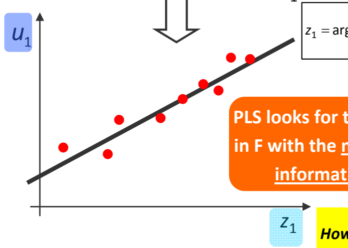
PLS

PLS finds the **directions** in **F** **best predicting** the output value **Φ**

z_1: direction **best explaining** $\mathbf{u}_1$



⇓



$$z_1 = \arg\left\{\max_{z_1}\left[r(z_1,u_1)\right]\right\}$$

PLS looks for the directions in F with the **most relevant information on Φ**

How to compute W?

Eigenvalue Decopomposition Algorithm (1/6)

- **Initialization: set h=0, Compute $\mathbf{F}_0$ and $\mathbf{\Phi}_0$**
- h=h+1
- **Compute the h-th column of W ($\mathbf{w}_h$)**
- **Compute the h-th column of Z ($\mathbf{z}_h$)**
- **Compute the h-th column of U ($\mathbf{u}_h$)**
- ***h==H?*** yes → **END**; no → **Update $\mathbf{F}_{h-1}$ and $\mathbf{\Phi}_{h-1}$** → h=h+1

Variables of the first sample $\mathbf{f}_1$

$$\mathbf{F} = \begin{bmatrix} f_{1,1} & f_{1,2} & \cdots & f_{1,N} \\ f_{2,1} & f_{2,2} & \cdots & f_{2,N} \\ \vdots & \ddots & & \vdots \\ f_{P,1} & \cdots & \ddots & f_{P,N} \end{bmatrix} \quad [P\times N]$$

Number of samples — Number of variables

Outputs of the first sample $\mathbf{\Phi}(\mathbf{f}_1)$

$$\mathbf{\Phi} = \begin{bmatrix} \Phi_{1,1} & \Phi_{1,2} & \cdots & \Phi_{1,V} \\ \Phi_{2,1} & \Phi_{2,2} & \cdots & \Phi_{2,V} \\ \vdots & \ddots & & \vdots \\ \Phi_{P,1} & \cdots & \ddots & \Phi_{P,V} \end{bmatrix} \quad [P\times V]$$

Number of outputs per sample

Average of *i*-th variable

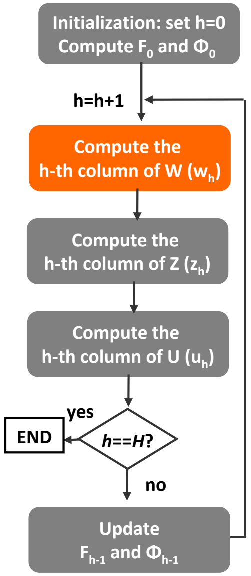
$$\bar{f}_i = \frac{1}{P}\sum_{p=1}^{P} f_{p,i} \Longrightarrow \mathbf{F}_0 = \mathbf{F} - \bar{\mathbf{F}} = \begin{bmatrix} (f_{1,1}-\bar{f}_1) & (f_{1,2}-\bar{f}_2) & \cdots & (f_{1,N}-\bar{f}_N) \\ (f_{2,1}-\bar{f}_1) & (f_{2,2}-\bar{f}_2) & \cdots & (f_{2,N}-\bar{f}_N) \\ \vdots & & \ddots & \vdots \\ (f_{P,1}-\bar{f}_1) & \cdots & \ddots & (f_{P,N}-\bar{f}_N) \end{bmatrix}$$

Subtract the average from both input and ouput

Average of *v*-th output

$$\bar{\Phi}_v = \frac{1}{P}\sum_{p=1}^{P} \Phi_{p,v} \Longrightarrow \mathbf{\Phi}_0 = \mathbf{\Phi} - \bar{\mathbf{\Phi}} = \begin{bmatrix} (\Phi_{1,1}-\bar{\Phi}_1) & (\Phi_{1,2}-\bar{\Phi}_2) & \cdots & (\Phi_{1,V}-\bar{\Phi}_V) \\ (\Phi_{2,1}-\bar{\Phi}_1) & (\Phi_{2,2}-\bar{\Phi}_2) & \cdots & (\Phi_{2,V}-\bar{\Phi}_V) \\ \vdots & & (\Phi_{p,v}-\bar{\Phi}_v) & \vdots \\ (\Phi_{P,1}-\bar{\Phi}_1) & \cdots & \ddots & (\Phi_{P,V}-\bar{\Phi}_V) \end{bmatrix}$$

Eigenvalue Decopomposition Algorithm (2/6)



At iteration *h*:

(1) Compute the matrix

$$\mathbf{M}_{h-1} = \mathbf{F}_{h-1}^T \mathbf{\Phi}_{h-1} \mathbf{\Phi}_{h-1}^T \mathbf{F}_{h-1}$$

$$[N\times N] \quad [N\times P] \quad [P\times V] \quad [V\times P] \quad [P\times N]$$

(2) Compute and sort the eigenvalues of $\mathbf{M}_{h-1}$

Solve $\det\left(\mathbf{M}_{h-1} - \lambda\mathbf{I}\right) = 0 \Rightarrow \lambda(M_{h-1})_1 > \lambda(M_{h-1})_2 > \ldots > \lambda(M_{h-1})_N$

(3) Compute $\mathbf{w}_h$ as the **eigenvector** associated to the **highest eigenvalue** ($\lambda(M_{h-1})_1$)

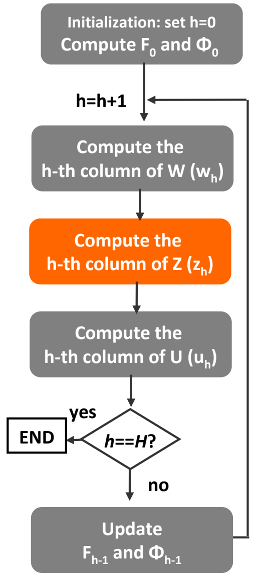
Solve $\left[\mathbf{M}_{h-1} - \lambda\left(\mathbf{M}_{h-1}\right)_1\right]\boxed{\mathbf{w}_h} = 0$

(4) Store $\mathbf{w}_h$ as the h-th column of matrix $\mathbf{W}$

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & \cdots & w_{1,h} & \cdots \\ w_{2,1} & \ddots & w_{2,h} & \\ \vdots & & \vdots & \\ w_{N,1} & \cdots & w_{N,h} & \cdots \end{bmatrix}$$

ELEDIA Research Center – All Rights Reserved

Eigenvalue Decopomposition Algorithm (3/6)



At iteration *h*:

(5) Compute the vector $\mathbf{z}_h$

$$\mathbf{z}_h = \mathbf{F}_{h-1}\mathbf{w}_h$$

$$\begin{bmatrix} z_{1,h} \\ z_{2,h} \\ \vdots \\ z_{P,h} \end{bmatrix} = \mathbf{F}_{h-1} \begin{bmatrix} w_{1,h} \\ w_{2,h} \\ \vdots \\ w_{P,h} \end{bmatrix}$$

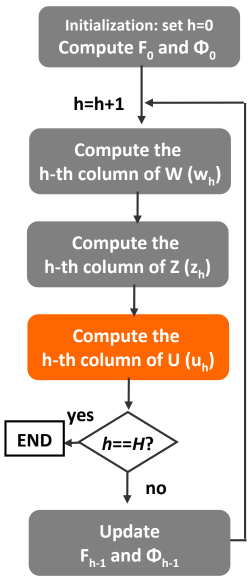
(6) Store $\mathbf{z}_h$ as the h-th column of matrix $\mathbf{Z}$

$$\mathbf{Z} = \begin{bmatrix} z_{1,1} & \cdots & z_{1,h} & \cdots \\ z_{2,1} & \ddots & z_{2,h} & \\ \vdots & & \vdots & \\ z_{P,1} & \cdots & z_{P,h} & \cdots \end{bmatrix}$$

Values of the *h*-th new variable for the *P* input samples

ELEDIA Research Center – All Rights Reserved

Eigenvalue Decopomposition Algorithm (4/6)



At iteration *h*:

(7) Compute the matrix

$$\mathbf{N}_{h-1} = \mathbf{\Phi}_{h-1}^T \mathbf{F}_{h-1} \mathbf{F}_{h-1}^T \mathbf{\Phi}_{h-1}$$

$$[V\times V] \quad [V\times P] \quad [P\times N] \quad [N\times P] \quad [P\times V]$$

(8) Compute and sort the eigenvalues of $\mathbf{N}_{h-1}$

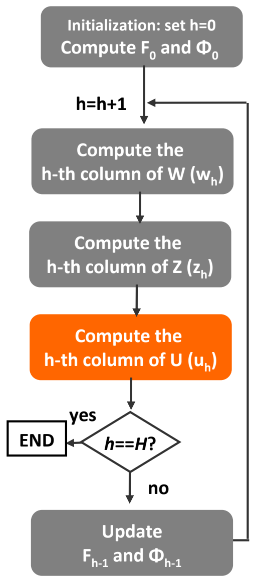
Solve $\det\left(\mathbf{N}_{h-1} - \lambda\mathbf{I}\right) = 0 \Rightarrow \lambda(\mathbf{N}_{h-1})_1 > \lambda(\mathbf{N}_{h-1})_2 > \ldots > \lambda(\mathbf{N}_{h-1})_N$

(9) Compute $\mathbf{c}_h$ as the **eigenvector** associated to the **highest eigenvalue** ($\lambda(\mathbf{N}_{h-1})_1$)

Solve $\left[\mathbf{N}_{h-1} - \lambda\left(\mathbf{N}_{h-1}\right)_1\right]\mathbf{c}_h = 0$

ELEDIA Research Center – All Rights Reserved

Eigenvalue Decopomposition Algorithm (5/6)



At iteration *h*:

(10) Compute the vector $\mathbf{u}_h$

$$\mathbf{u}_h = \mathbf{\Phi}_{h-1}\mathbf{c}_h$$

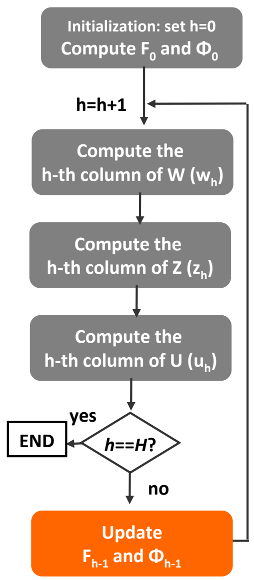
$$\begin{bmatrix} u_{1,h} \\ u_{2,h} \\ \vdots \\ u_{P,h} \end{bmatrix} = \mathbf{\Phi}_{h-1} \begin{bmatrix} c_{1,h} \\ c_{2,h} \\ \vdots \\ c_{V,h} \end{bmatrix}$$

(11) Store $\mathbf{u}_h$ as the h-th column of matrix $\mathbf{U}$

$$\mathbf{U} = \begin{bmatrix} u_{1,1} & \cdots & u_{1,h} & \cdots \\ u_{2,1} & \ddots & u_{2,h} & \\ \vdots & & \vdots & \\ u_{P,1} & \cdots & u_{P,h} & \cdots \end{bmatrix}$$

ELEDIA Research Center – All Rights Reserved

Eigenvalue Decopomposition Algorithm (6/6)



At iteration *h*:

(12) Update $\mathbf{F}_{h-1}$

$$\mathbf{F}_h = \mathbf{F}_{h-1} - \mathbf{z}_h \mathbf{z}_h^T \mathbf{F}_{h-1}$$

(13) Update $\mathbf{\Phi}_{h-1}$

$$\mathbf{\Phi}_h = \mathbf{\Phi}_{h-1} - \mathbf{z}_h \mathbf{z}_h^T \mathbf{\Phi}_{h-1}$$

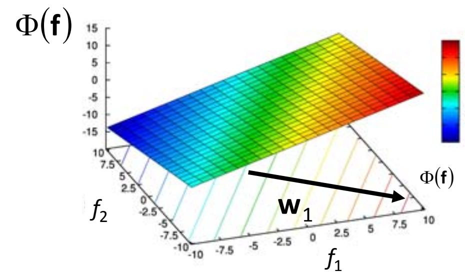
ELEDIA Research Center – All Rights Reserved

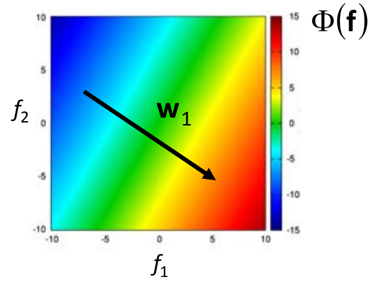
PLS Example (1/8)

EXAMPLE:
Number of variables: N = 2
Number of samples: P = 10
Dimension of the output: V=1
Number of new variables: H=1

"Rotated plane"

$$\Phi(f_1, f_2) = f_1 \cos(\vartheta_r) - f_2 \sin(\vartheta_r) \qquad \vartheta_r = 30[\text{deg}]$$



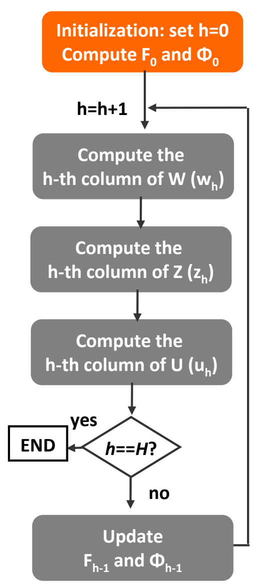


Observation Direction $\mathbf{w}_1$ is sufficient for computing $\Phi(\mathbf{f})$ univocally

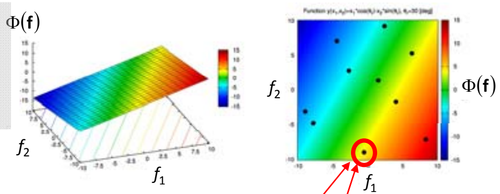
Goal Find $\mathbf{w}_1$ to **reduce the number of variables** from **N=2** to **H=1**

ELEDIA Research Center – All Rights Reserved

PLS Example (2/8)



EXAMPLE:
Number of variables: N = 2
Number of samples: P = 10
Dimension of the output: V=1
Number of new variables: H=1



$$\Phi(f_1, f_2) = f_1 \cos(\vartheta_r) - f_2 \sin(\vartheta_r)$$

$\mathbf{F} =$

f_1	f_2
-4.46	7.09
4.04	-1.72
-2.73	2.75
1.50	1.35
-9.00	-3.12
8.45	-7.13
2.40	9.24
-4.79	-8.97
6.34	5.30
-7.82	-4.80

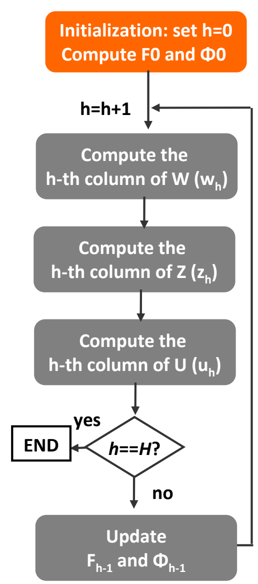
$\mathbf{\Phi} =$

$\Phi(f_1,f_2)$
-7.40
4.36
-3.74
0.62
-6.24
10.88
-2.55
4.07
2.84
-4.38

Samples of the function

ELEDIA Research Center – All Rights Reserved

PLS Example (3/8)



EXAMPLE:
Number of variables: N = 2
Number of samples: P = 10
Dimension of the output: V=1
Number of new variables: H=1

$$\bar{f}_1 = -0.178 \qquad \bar{\Phi} = -0.15$$
$$\bar{f}_2 = -0.002$$

Centered (zero-mean) version of both input and output samples

$\mathbf{F}_0 =$

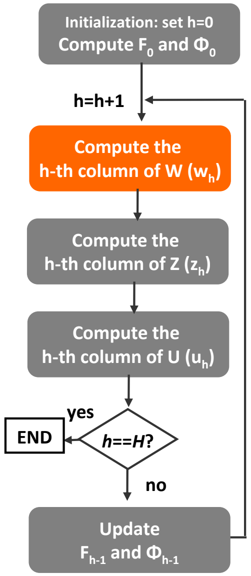
$f_1 - \bar{f}_1$	$f_2 - \bar{f}_2$
-4.28	7.09
4.22	-1.71
-2.56	2.75
1.67	1.35
-8.83	-3.11
8.63	-7.13
2.57	9.23
-0.30	-8.97
6.51	5.30
-7.65	-4.80

$\mathbf{\Phi}_0 =$

$\Phi - \bar{\Phi}$
-7.25
4.51
-3.59
0.78
-6.09
11.04
-2.39
4.23
2.99
-4.22

ELEDIA Research Center – All Rights Reserved

PLS Example (4/8)



EXAMPLE:
Number of variables: N = 2
Number of samples: P = 10
Dimension of the output: V=1
Number of new variables: H=1

Iteration h=1

1. Compute the matrix $\mathbf{M}_0 = \mathbf{F}_0^T \mathbf{\Phi}_0 \mathbf{\Phi}_0^T \mathbf{F}_0$

$$\underset{[N\times N]}{\mathbf{M}_0} = 10^4 \begin{bmatrix} 6.44 & -3.85 \\ -3.85 & 2.30 \end{bmatrix}$$

2. Compute and sort the eigenvalues of $\mathbf{M}_0$

Highest eigenvalue

Solve $\det(\mathbf{M}_0 - \lambda \mathbf{I}) = 0 \Rightarrow \lambda(M_0)_1 = 8.74 \times 10^4$

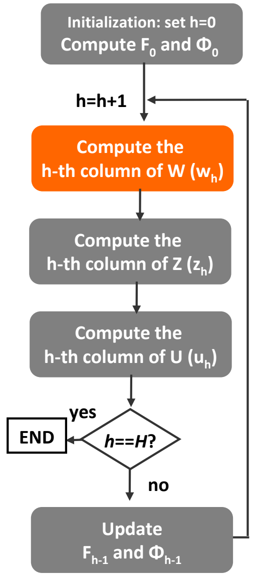
3. Compute $\mathbf{w}_1$ as the **eigenvector** associated to the **highest eigenvalue**

Solve $[\mathbf{M}_0 - \lambda(\mathbf{M}_0)_1]\mathbf{w}_1 = 0 \Rightarrow \mathbf{w}_1 = \begin{bmatrix} 0.86 \\ -0.51 \end{bmatrix}$

4. Store $\mathbf{w}_1$ as the 1-st column of matrix $\mathbf{W}$

$$\mathbf{W} = \begin{bmatrix} 0.86 & \cdots \\ -0.51 & \cdots \end{bmatrix}$$

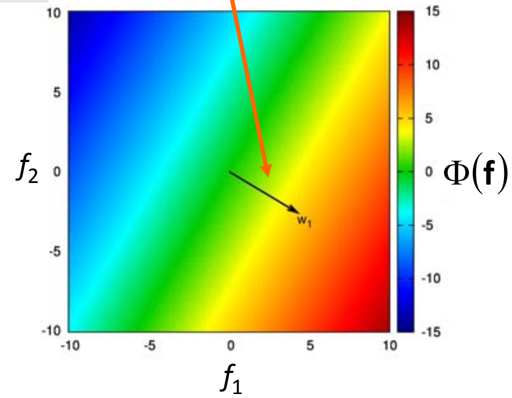
PLS Example (5/8)



EXAMPLE:
Number of variables: N = 2
Number of samples: P = 10
Dimension of the output: V=1
Number of new variables: H=1

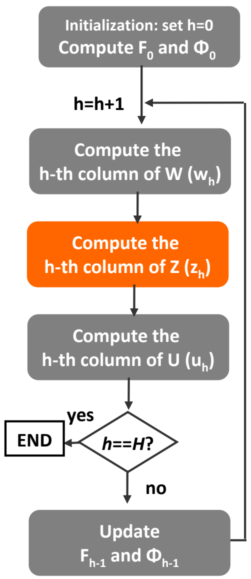
w1 is the direction we were looking for!

$$\mathbf{w}_1 = \begin{bmatrix} 0.86 \\ -0.51 \end{bmatrix}$$



$\Phi(\mathbf{f})$ is **linear** w.r.t. this direction!

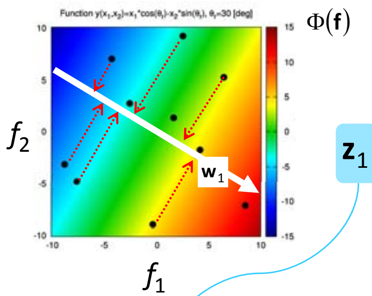
PLS Example (6/8)



EXAMPLE:
Number of variables: N = 2
Number of samples: P = 10
Dimension of the output: V=1
Number of new variables: H=1

Iteration h=1

5. Project samples onto $\mathbf{w}_1$: compute the vector $\mathbf{z}_1$



$\mathbf{z}_1 = \mathbf{F}_0 \mathbf{w}_1 =$

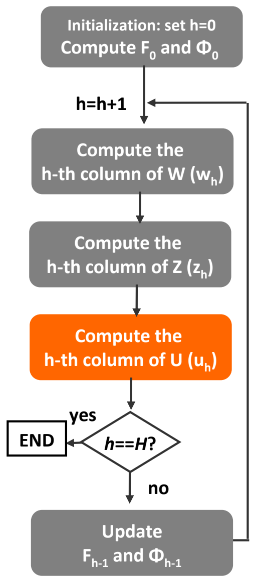
-7.31
4.50
-3.60
0.75
-5.98
11.06
-2.53
4.34
2.88
-4.11

6. Store $\mathbf{z}_1$ as the 1-st column of matrix $\mathbf{Z}$

$$\mathbf{Z} = \begin{bmatrix} z_{1,1} & \cdots & \cancel{z_{1,h}} & \cdots \\ z_{2,1} & \ddots & \cancel{z_{2,h}} & \\ \vdots & & \cancel{\vdots} & \\ z_{P,1} & \cdots & \cancel{z_{N,h}} & \cdots \end{bmatrix}$$

Knowledge of $z_{p,1}$ is **sufficient** for computing $\Phi(\mathbf{f})$

PLS Example (7/8)



EXAMPLE:
Number of variables: N = 2
Number of samples: P = 10
Dimension of the output: V=1
Number of new variables: H=1

Iteration h=1

7. Compute the matrix

$$\underset{[V\times V]}{\mathbf{N}_0} = \mathbf{\Phi}_0^T \mathbf{F}_0 \mathbf{F}_0^T \mathbf{\Phi}_0 = 8.74 \times 10^4$$

In this case V=1

8. Compute and sort the eigenvalues of $\mathbf{N}_0$

Solve $\det(\mathbf{N}_0 - \lambda \mathbf{I}) = 0 \Rightarrow \lambda(\mathbf{N}_{h-1})_1 = 8.74 \times 10^4$

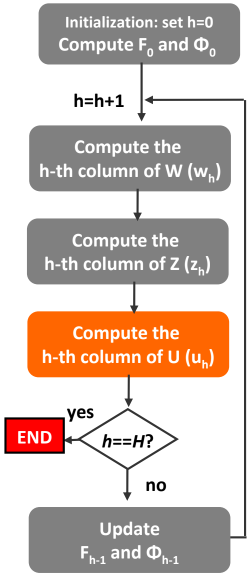
9. Compute $\mathbf{c}_1$ as the **eigenvector** associated to the **highest eigenvalue**

Solve $[\mathbf{N}_0 - \lambda(\mathbf{N}_0)_1]\mathbf{c}_1 = 0 \Rightarrow \mathbf{c}_1 = 1$

10. Compute the vector $\mathbf{u}_1$

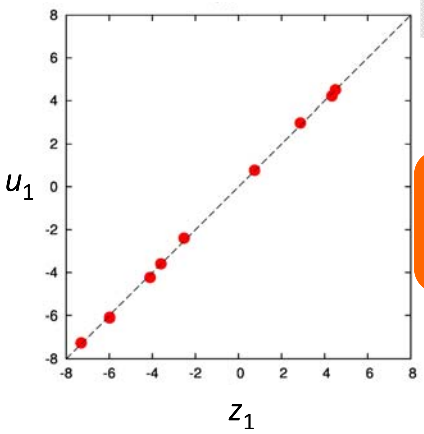
$$\mathbf{u}_1 = \mathbf{\Phi}_0 \mathbf{c}_1 = \mathbf{\Phi}_0$$

PLS Example (8/8)



Iteration h=1

EXAMPLE:
Number of variables: N = 2
Number of samples: P = 10
Dimension of the output: V=1
Number of new variables: H=1



Correlation between z_1 and u_1 is maximized

$$r(z_1,u_1)\approx 0.99$$

We can **use the new variable z_1 instead of f_1 and f_2** to compute/predict Φ

ELEDIA Research Center – All Rights Reserved

Conclusions

- Introduction and Motivation
- Prediction Through **Interpolation Techniques**
  - ***Nearest Neighbor Interpolation***
  - ***Linear Interpolation***
- **3-Steps Learning-by-Examples** (***LBE***) Framework
  - Step 1: ***Dimensionality Reduction***
    - ***Sequential Feature Selection (SFT)***
    - ***Principal Component Analysis (PCA)***
    - ***Sammon Mapping***
    - ***Partial Least Squares (PLS)***

ELEDIA Research Center – All Rights Reserved

ELEDIA@UniTN - University of Trento
Dept. of Civil, Environmental, and Mechanical Engineering
Via Mesiano 77, I-38123 Trento, Italy
E-mail: contact@eledia.org
Web: www.eledia.org/eledia-unitn

Introduction to ML and AI Methods (Part I)

Dr. Marco SALUCCI

Day 1 - August 29th, 2022

2022 ELEDIA@ICAM PhD Summer Schools
Machine Learning & AI Methods
Theory, Techniques, and Advanced Engineering Applications
29 Aug. - 02 Sept. 2022, Trento, Italy (Onsite and Online)

Additional Information

Contact Point: Dr. Marco SALUCCI
ELEDIA Research Center Board Member
Assistant Professor @ DICAM - University of Trento (Trento - Italy)
E-mail: marco.salucci@unitn.it
Web-site: www.eledia.org

summer-schools@eledia.org

- https://www.eledia.org
- https://twitter.com/ELEDIAResearch
- https://www.facebook.com/eledianet
- https://www.linkedin.com/company/eledianet
- https://www.instagram.com/eledianet
- WeChat ID: eledianet

ELEDIA Research Center – All Rights Reserved