INTELLIGENT SENSOR PROCESSING USING MACHINE LEARNING (1)

Dr TIAN Jing

tianjing@nus.edu.sg

Module objective

Module: Intelligent sensor processing using machine learning

Knowledge and understanding

- Understand the fundamentals of intelligent sensor processing using machine learning and its applications

Key skills

- Design, build, implement intelligent sensor processing using machine learning for real-world applications

Major reference

- [Introduction] MIT 6.S191: *Introduction to Deep Learning*, http://introtodeeplearning.com/
- [Intermediate] *Machine Learning for Signal Processing*, UIUC, https://courses.engr.illinois.edu/cs598ps/fa2018/index.html
- [Intermediate] *Neural Networks for Signal Processing*, UFL, http://www.cnel.ufl.edu/courses/EEL6814/EEL6814.php
- [Comprehensive] M. Hoogendoorn, B. Funk, *Machine Learning for the Quantified Self: On the Art of Learning from Sensory Data*, Springer, 2018, https://ml4qs.org

Topics

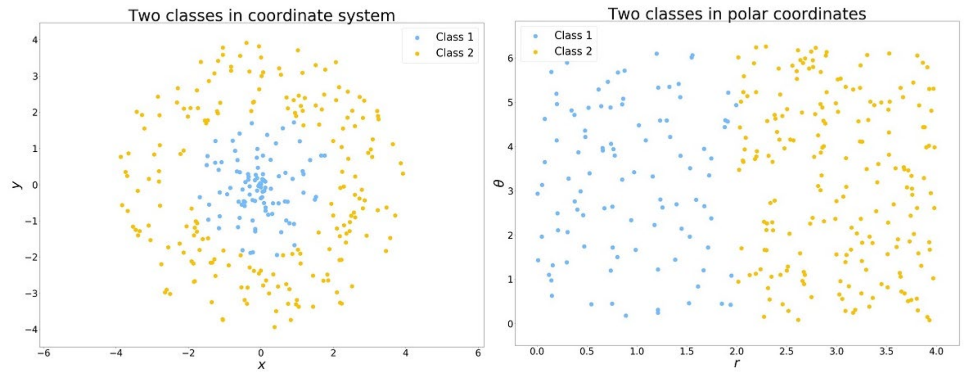
- Introduction to signal representation
- Data driven signal representation

Machine learning for signal processing

Sensor	Signal capture	Signal transmission	Feature extraction	Modeling
• Various types of sensors, audio, vision, IoT, etc	• Sensing • Denoising	• Source coding, channel coding, compression	• Deterministic features • Data-driven features • Feature representation learning	• Classification • Regression • Predication

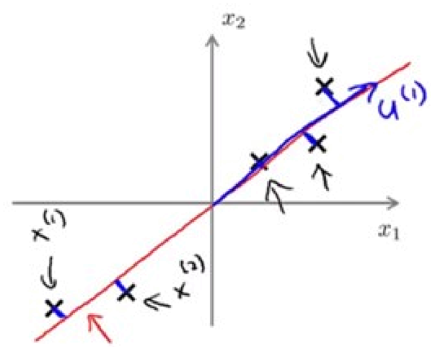
- Representation: How to represent signals for effective processing
- Modeling: How to model the systematic and statistical characteristics of the signal
- Classification: How do we assign a class to the data
- Prediction: How do we predict new or unseen values or attributes of the data

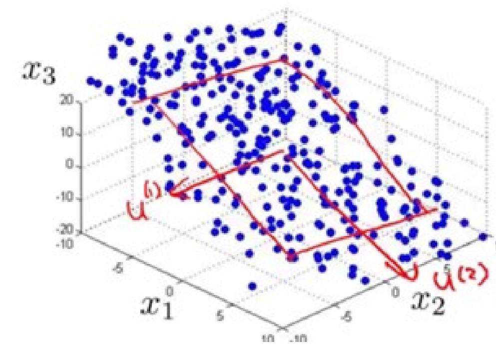
Signal representation



1. Signal representation can be manually designed (input-agnostic)

[Covered in previous class]

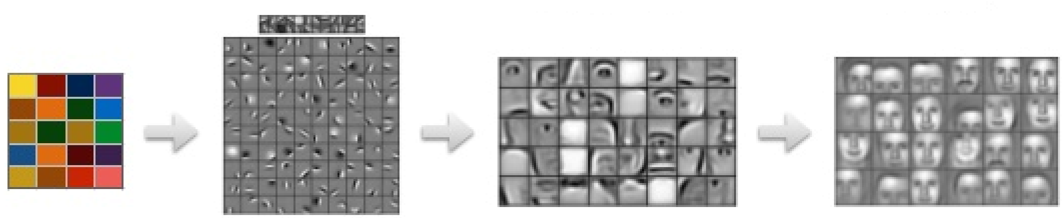




2. Signal representation can be adaptively to the input signal

[Covered in today class]

3. Signal representation can be learned from signal dataset



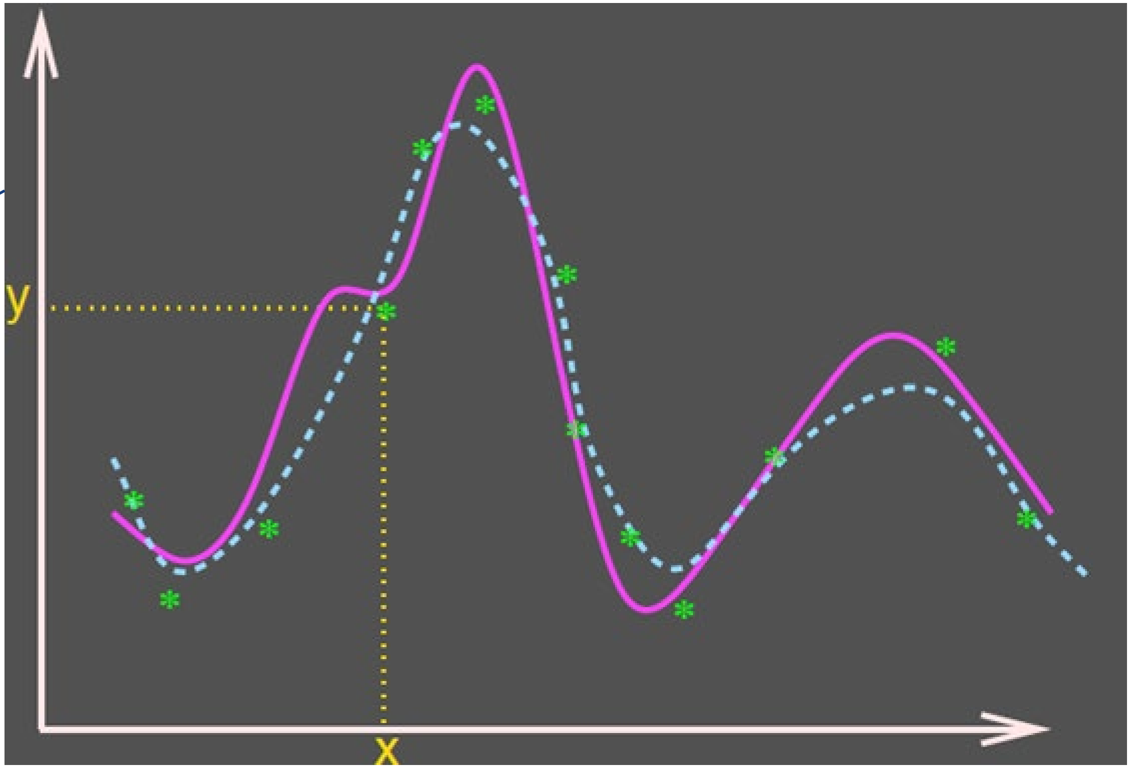
[Covered in today class]

Reference: https://www.kdnuggets.com/2018/12/feature-engineering-explained.html; https://www.dezyre.com/data-science-in-python-tutorial/principal-component-analysis-tutorial

Signal representation requirements

1. Smoothness

- By “smoothness” we mean that close inputs are mapped to close outputs (representations).
- To fit the data (green stars), the blue model is smoother than the purple model.



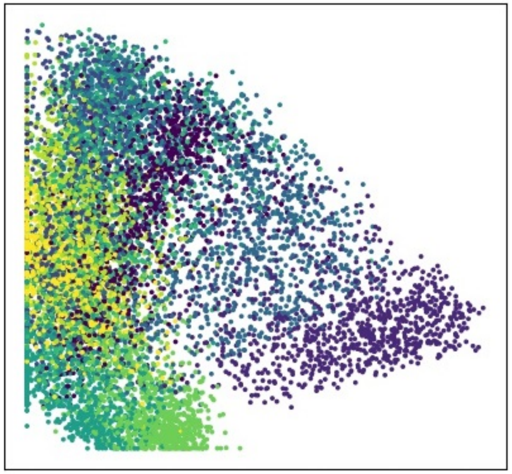
An example of model fitting
$$y = f(x)$$

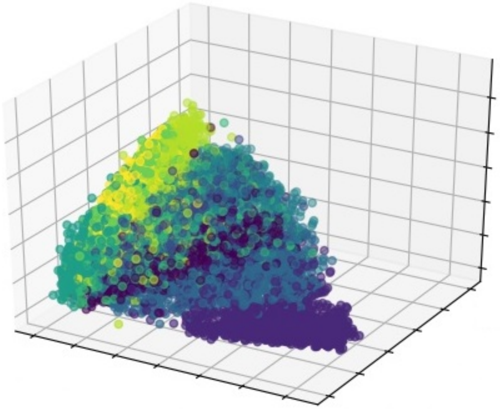
Source: ECE 8527, Introduction to Machine Learning and Pattern Recognition, https://www.isip.piconepress.com/courses/temple/ece_8527/

Signal representation requirements

2. Dimensionality

- Sometime, we need a representation with a lower dimension than the input dimension, to avoid complicated calculations or having too many configurations. But note that we should not loose too much important data.
- Sometime, we also need to increase dimensions of data to obtain richer representations.



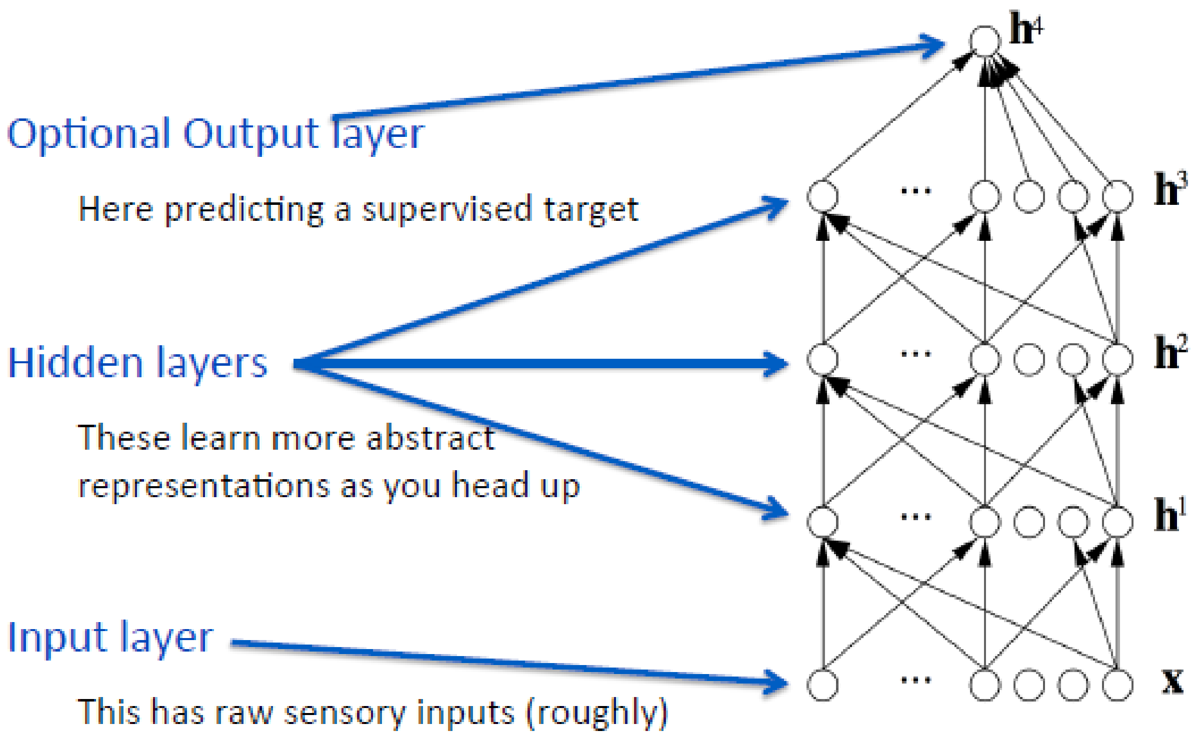


Source: ECE 8527, Introduction to Machine Learning and Pattern Recognition, https://www.isip.piconepress.com/courses/temple/ece_8527/

Signal representation requirements

3. Depth and abstraction

A good representation expresses high level and abstract features. In order to achieve such representations we can use deep architectures that allow reuse of low level features to potentially get more abstract features at higher layers.



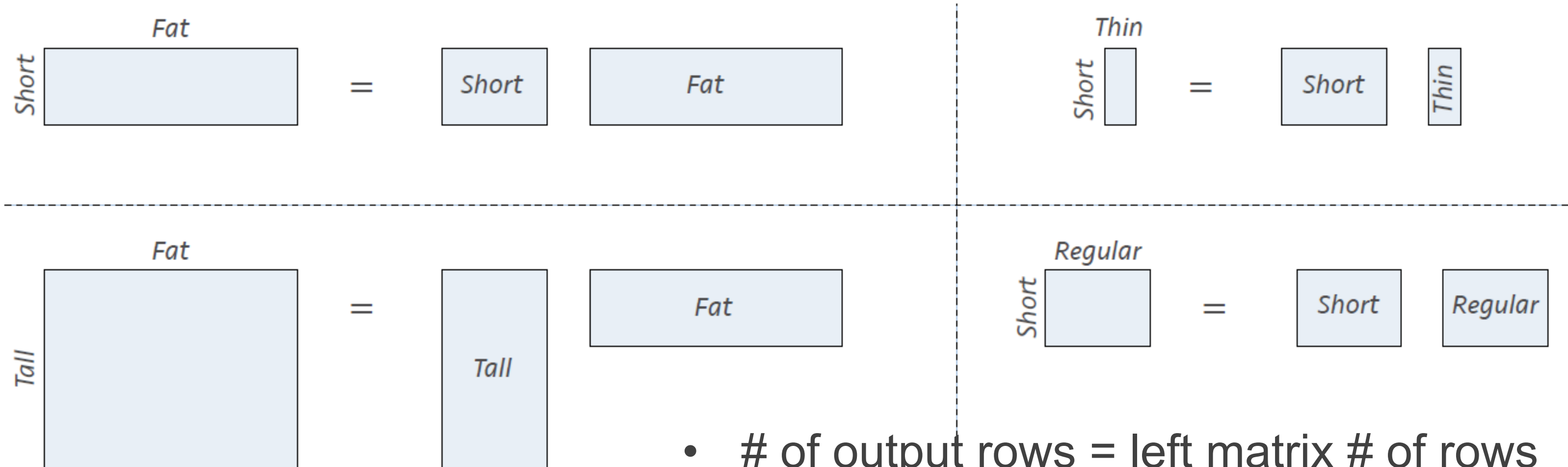
Source: ECE 8527, Introduction to Machine Learning and Pattern Recognition, https://www.isip.piconepress.com/courses/temple/ece_8527/

Topics

- Introduction to signal representation
- Data driven signal representation

Warm-up: How do we look at signal

- 1D signal (e.g. sound) will be vector
- 2D signal (e.g. image) will be matrix



- # of output rows = left matrix # of rows
- # of output columns = right matrix # of columns

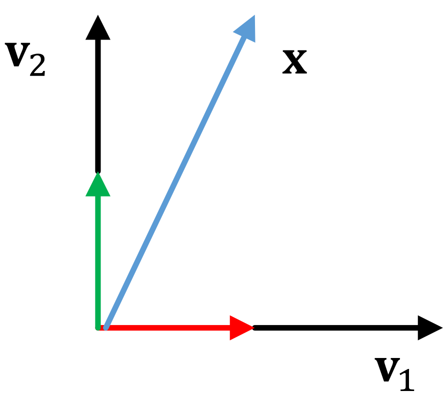
$$\begin{bmatrix} \$3 & \$4 & \$2 \end{bmatrix} \times \begin{bmatrix} 13 & 9 & 7 & 15 \\ 8 & 7 & 4 & 6 \\ 6 & 4 & 0 & 3 \end{bmatrix} = \begin{bmatrix} \$83 & \$63 & \$37 & \$75 \end{bmatrix}$$

$$\$3\times13 + \$4\times8 + \$2\times6$$

Source: https://www.mathsisfun.com/algebra/matrix-multiplying.html

Warm-up: Basis vectors

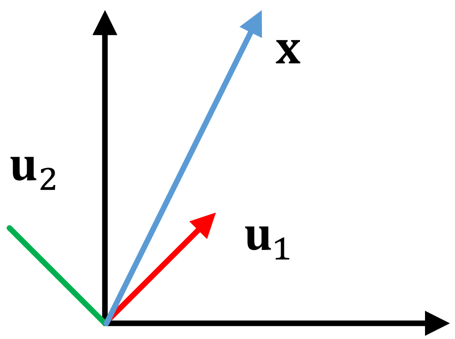
- A given vector value is represented with respect to a *coordinate system*.
- A coordinate system is defined by a set of linearly independent vectors forming the system *basis*.
- Any vector value is represented as a linear sum of the basis vectors.
- Key idea: The basis vector determines how the signal is represented. We can change basis vectors so that we can change signal representation.



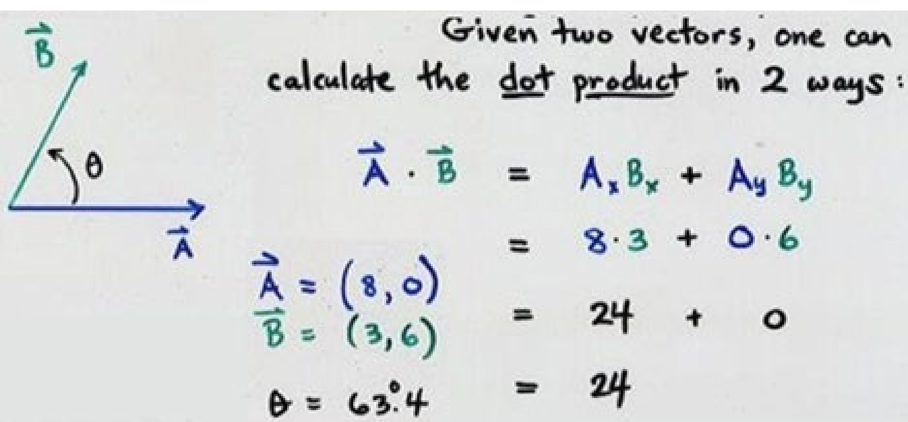
Signal	$\mathbf{x} = (1, 2)$
Basis vectors	$\mathbf{v}_1 = (1, 0), \mathbf{v}_2 = (0, 1)$
Signal representation coefficients	$\mathbf{w} = (1, 2)$
Justification	$\mathbf{x} = 1 \times (1, 0) + 2 \times (0, 1)$

Warm-up: Change of basis vectors

- **Question**: Given a vector $\mathbf{x}$, represented in an orthonormal basis vectors $\mathbf{v}_1, \mathbf{v}_2$, what is the representation of $\mathbf{x}$ in a different orthonormal basis vectors $\mathbf{u}_1, \mathbf{u}_2$?



Signal	$\mathbf{x} = (1, 2)$
Basis vectors	$\mathbf{u}_1 = \left(\sqrt{2}/2, \sqrt{2}/2\right), \mathbf{u}_2 = \left(-\sqrt{2}/2, \sqrt{2}/2\right)$
Signal representation coefficients	$\mathbf{w} = \left(3\sqrt{2}/2, \sqrt{2}/2\right)$
Justification	$\mathbf{x} = 3\sqrt{2}/2 \times \left(\sqrt{2}/2, \sqrt{2}/2\right) + \sqrt{2}/2 \times \left(-\sqrt{2}/2, \sqrt{2}/2\right)$



- Decompose signal $\mathbf{x}$

$$w_i = \langle \mathbf{x}, \mathbf{u}_i \rangle = \mathbf{x}^T \mathbf{u}_i = \sum_j x(j) u_i(j)$$

where $<\cdot>$ is the dot product of two vectors

- Reconstruct signal $\mathbf{x}$

$$\mathbf{x} = \sum_i w_i \times \mathbf{u}_i$$

Photo: http://spiff.rit.edu/classes/phys311.old/lectures/dot/dot.html

Appendix: Checkerboard basis

Question: All the basis we have considered so far are data agnostic. Checkerboards, Complex exponentials, Wavelets.. We use the same bases regardless of the data we analyze. How about data specific bases that consider the underlying data? Is there something better than checkerboards?

Signal at <u>standard</u> basis: $\mathbf{x} = \begin{bmatrix} 2 & 1 \\ 6 & 1 \end{bmatrix}$

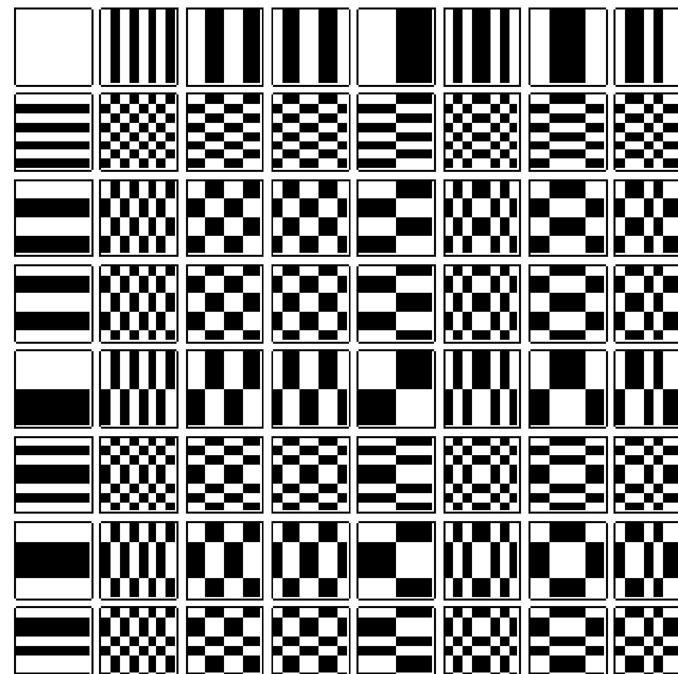
$$\begin{bmatrix} 2 & 1 \\ 6 & 1 \end{bmatrix} = 2 \times \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} + 1 \times \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} + 6 \times \begin{bmatrix} 0 & 0 \\ 1 & 0 \end{bmatrix} + 1 \times \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}$$

<u>New</u> basis

Signal at <u>new</u> basis: $\mathbf{x} = \begin{bmatrix} 5 & -2 \\ 2 & -3 \end{bmatrix}$

$$\mathbf{u}_1 = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix}/2 \qquad \mathbf{u}_2 = \begin{bmatrix} 1 & 1 \\ -1 & -1 \end{bmatrix}/2$$

$$\mathbf{u}_3 = \begin{bmatrix} -1 & 1 \\ 1 & -1 \end{bmatrix}/2 \qquad \mathbf{u}_4 = \begin{bmatrix} -1 & 1 \\ -1 & 1 \end{bmatrix}/2$$



Recall the formula

$$w_i = \langle \mathbf{x}, \mathbf{u}_i \rangle = \mathbf{x}^T \mathbf{u}_i = \sum_j x(j) u_i(j)$$

where $<\cdot>$ is the dot product of two vectors

$$\mathbf{x} = \sum_i w_i \times \mathbf{u}_i$$

Idea: Energy compaction property

Note: N might not be as same as the dimension of $\mathbf{x}$

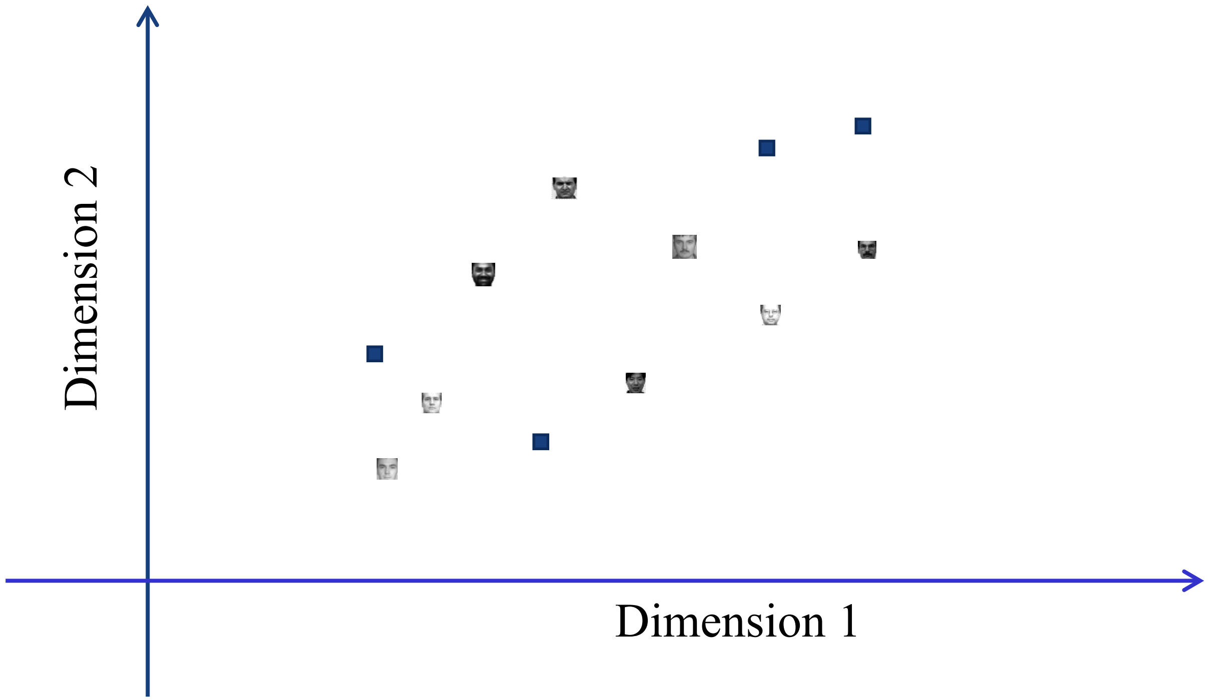
How to define better basis for signal representation?

- Given the signal representation as $\mathbf{x} = \sum_{k=1}^{N} w_k \mathbf{u}_k$
- The ideal is $\hat{\mathbf{x}} = w_1\mathbf{u}_1 + w_2\mathbf{u}_2 + w_3\mathbf{u}_3 + \cdots + w_N\mathbf{u}_N$ based on N basis components. Its error is defined as $\mathrm{Error}_N = ||\mathbf{x} - \hat{\mathbf{x}}||^2$
- If the signal representation is terminated at any point, we should still get most of the information about the data, that means $\mathrm{Error}_N < \mathrm{Error}_{N-1}$

Idea: Energy compaction property

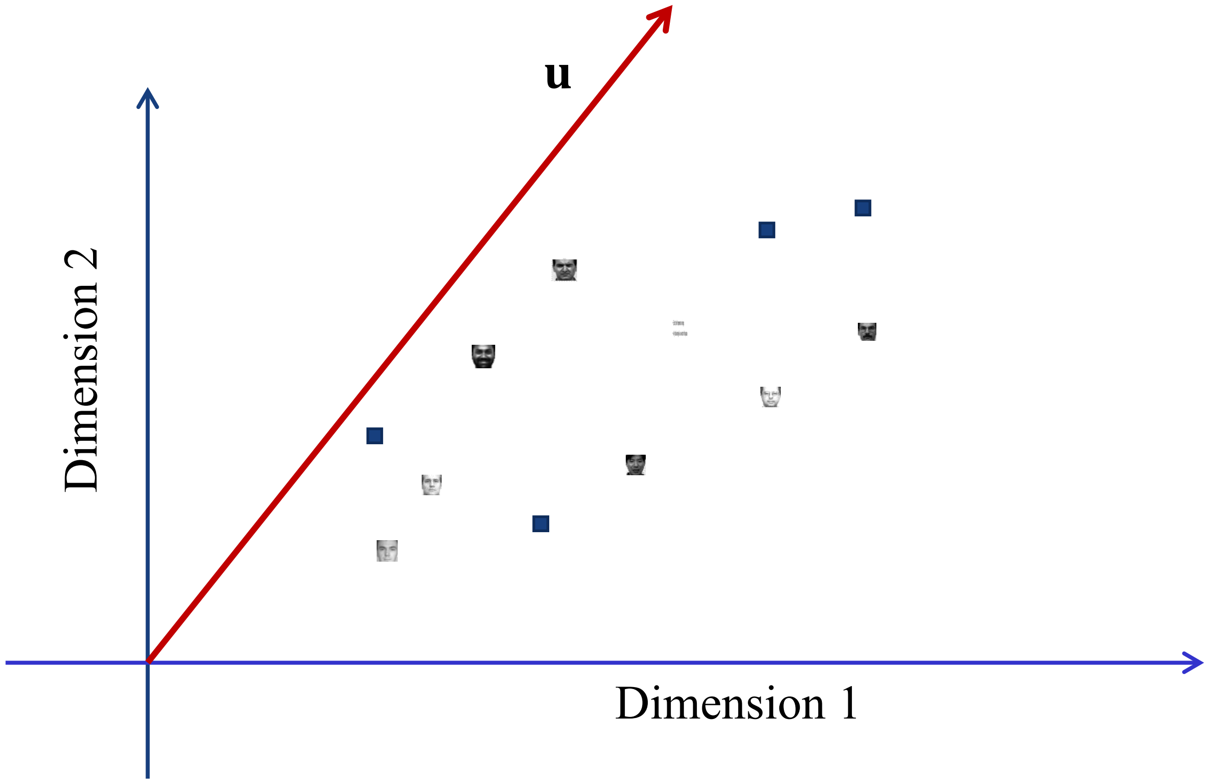
- Assumption: There are a set of N "typical" basis vectors that captures most of all (say, M) input data $\mathbf{x}_i$, where $i = 1, \cdots, M$.
- Approximate <u>every</u> data $\mathbf{x}_i$ as $\hat{\mathbf{x}}_i = w_{i,1}\boldsymbol{u}_1 + w_{i,2}\boldsymbol{u}_2 + \cdots + w_{i,N}\boldsymbol{u}_N$
    - $\mathbf{u}_2$ is used to "correct" errors resulting from using only $\mathbf{u}_1$.
    - $\left\|\mathbf{x}_i - \left(w_{i,1}\boldsymbol{u}_1 + w_{i,2}\boldsymbol{u}_2\right)\right\|^2 < \left\|\mathbf{x}_i - w_{i,1}\boldsymbol{u}_1\right\|^2$
    - $\mathbf{u}_3$ corrects errors remaining after correction with $\mathbf{u}_2$
    - $\left\|\mathbf{x}_i - \left(w_{i,1}\boldsymbol{u}_1 + w_{i,2}\boldsymbol{u}_2 + +w_{i,3}\boldsymbol{u}_3\right)\right\|^2 < \left\|\mathbf{x}_i - \left(w_{i,1}\boldsymbol{u}_1 + w_{i,2}\boldsymbol{u}_2\right)\right\|^2$
    - And so on
- Estimate $\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_N$ to minimize the squared error between the original signal and the reconstructed signal.

Find the first basis vector



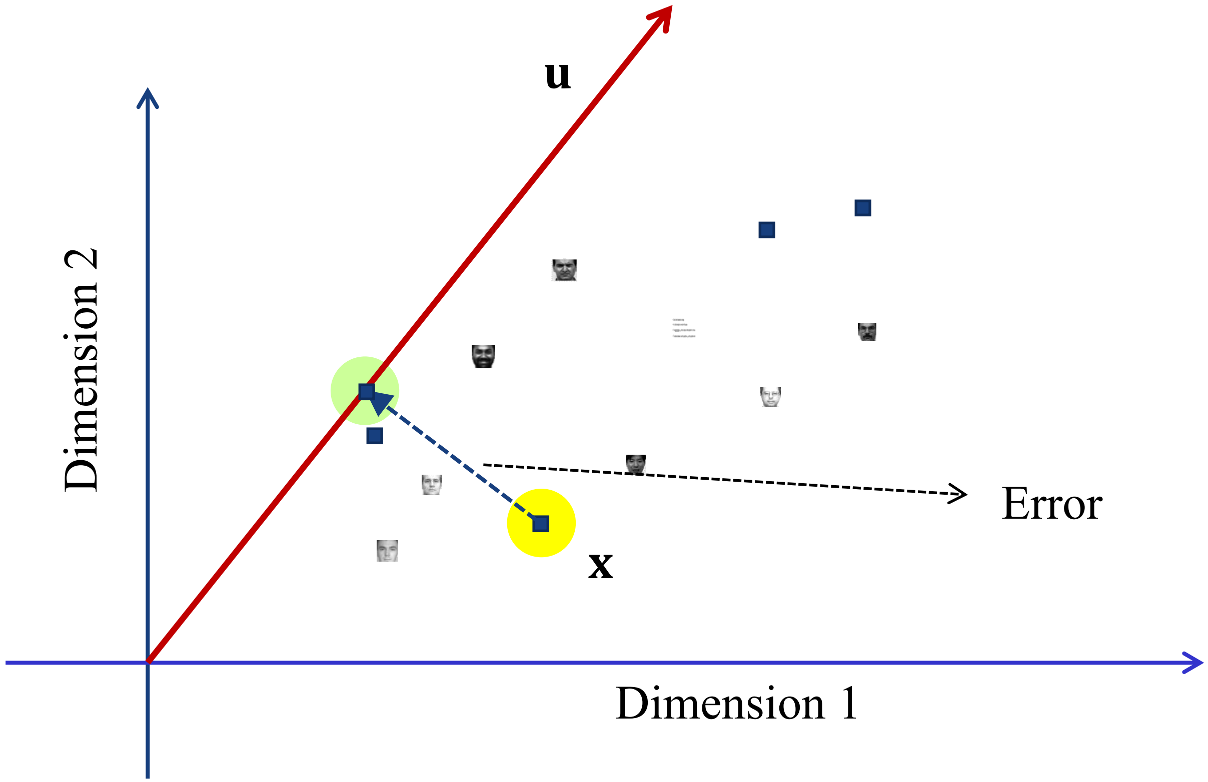
- Each "point" represents a signal data (displayed as image for visualization).

Find the first basis vector



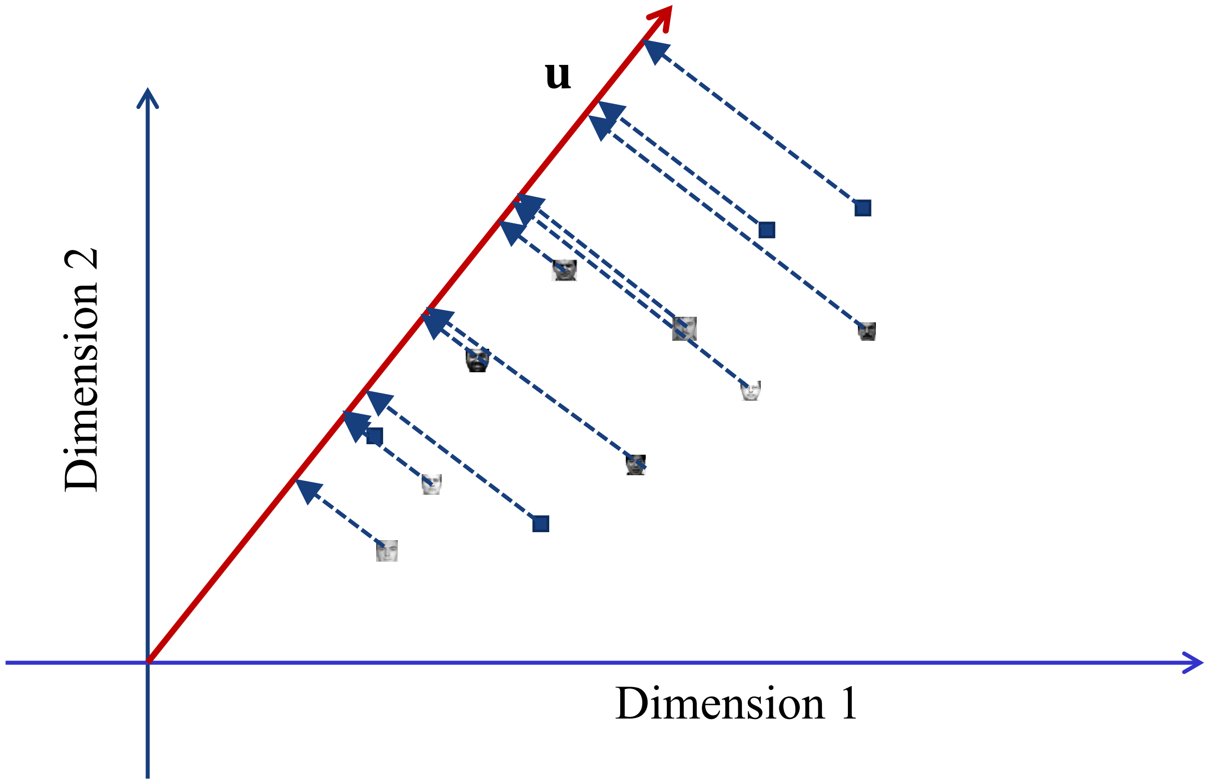
- Each "point" represents a signal data (displayed as image for visualization).
- Any "basis vector" **u** is a vector in this space.

Find the <u>first</u> basis vector



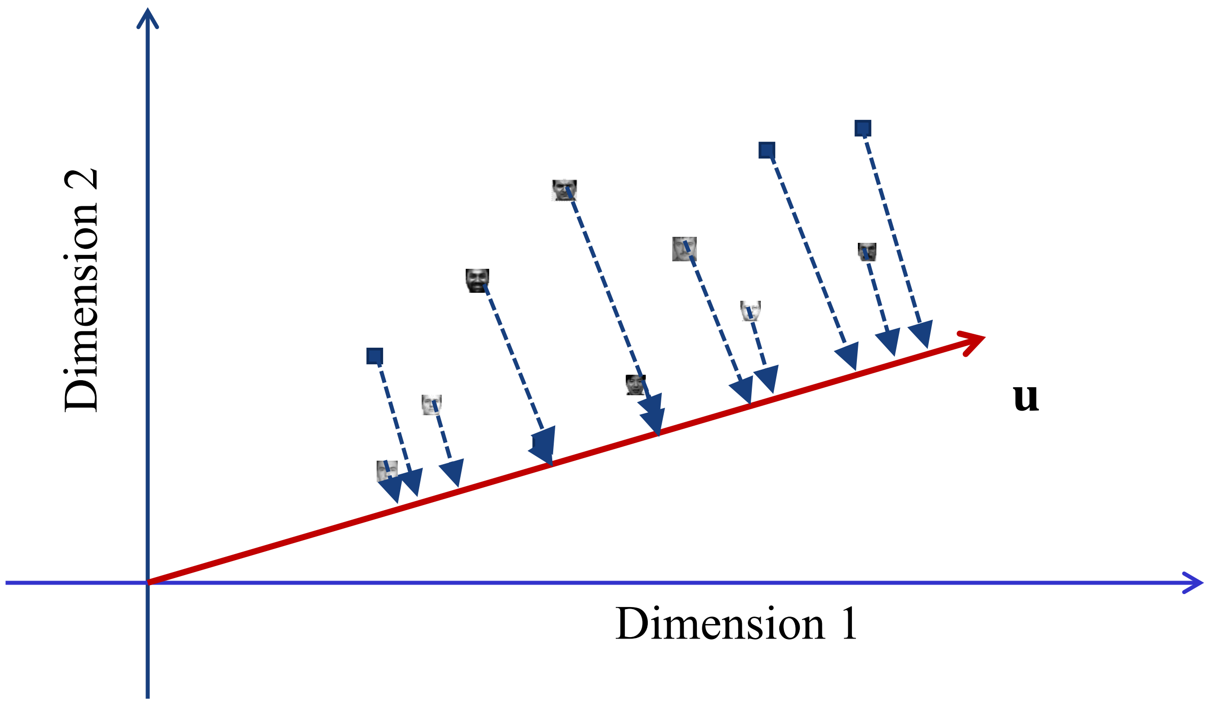
- Each "point" represents a signal data (displayed as image for visualization).
- Any "basis vector" $\mathbf{u}$ is a vector in this space.
- The approximation $\mathbf{u}\mathbf{u}^T\mathbf{x}$ for any signal $\mathbf{x}$ is the *projection* of $\mathbf{x}$ onto $\mathbf{u}$.
- The distance between $\mathbf{x}$ and its projection $\mathbf{u}\mathbf{u}^T\mathbf{x}$ is the *projection error.*

Find the first basis vector



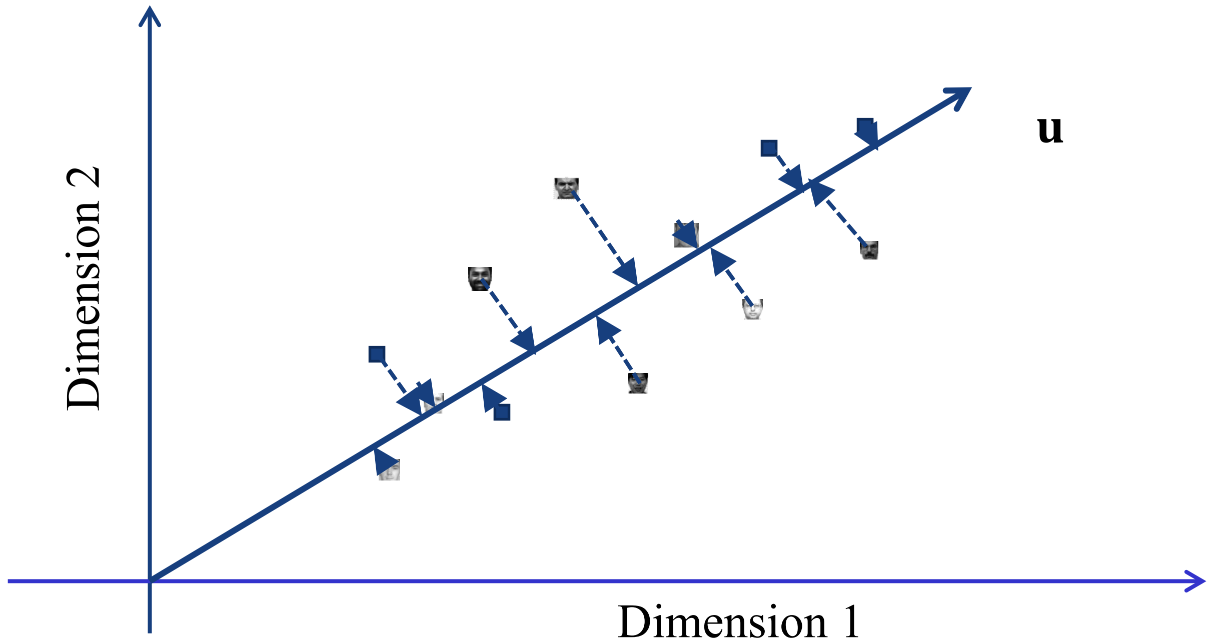
- *Every* signal data will suffer error when approximated by its projection on $\mathbf{u}$
- The total squared length of all error lines is the *total squared projection error.*
- The problem of finding the first basis vector: Find the $\mathbf{u}$ for which the total projection error is minimum!
- This "minimum squared error" $\mathbf{u}$ is our "best" first typical basis vector.

Find the first basis vector



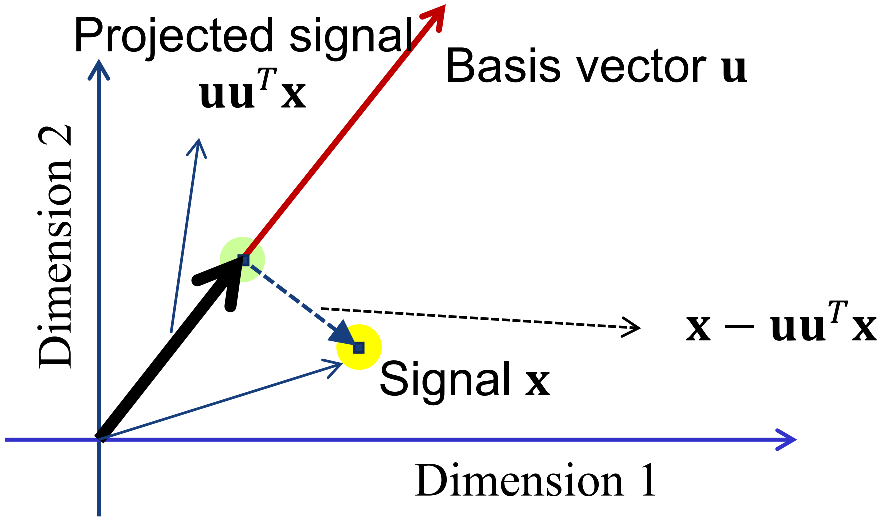
- *Every* signal data will suffer error when approximated by its projection on $\mathbf{u}$
- The total squared length of all error lines is the *total squared projection error.*
- The problem of finding the first basis vector: Find the $\mathbf{u}$ for which the total projection error is minimum!
- This “minimum squared error” $\mathbf{u}$ is our “best” first typical basis vector.

Find the <u>first</u> basis vector



- *Every* signal data will suffer error when approximated by its projection on $\mathbf{u}$
- The total squared length of all error lines is the *total squared projection error.*
- The problem of finding the first basis vector: Find the $\mathbf{u}$ for which the total projection error is minimum!
- This "minimum squared error" $\mathbf{u}$ is our "best" first typical basis vector.

Find the first basis vector



- Projection of a vector $\mathbf{x}$ on to a vector $\mathbf{u}$, which has unit length ($\|\mathbf{u}\| = \mathbf{1}$), $\hat{\mathbf{x}} = \mathbf{u}\mathbf{u}^T\mathbf{x}$

$$Error = \|\mathbf{x} - \hat{\mathbf{x}}\|^2 = \|\mathbf{x} - \mathbf{u}\mathbf{u}^T\mathbf{x}\|^2$$
$$= (\mathbf{x} - \mathbf{u}\mathbf{u}^T\mathbf{x})^T(\mathbf{x} - \mathbf{u}\mathbf{u}^T\mathbf{x})$$
$$= \mathbf{x}^T\mathbf{x} - \mathbf{x}^T\mathbf{u}\mathbf{u}^T\mathbf{x} - \mathbf{x}^T\mathbf{u}\mathbf{u}^T\mathbf{x} + \mathbf{x}^T\mathbf{u}\mathbf{u}^T\mathbf{u}\mathbf{u}^T\mathbf{x}$$
$$= \mathbf{x}^T\mathbf{x} - \mathbf{x}^T\mathbf{u}\mathbf{u}^T\mathbf{x}$$

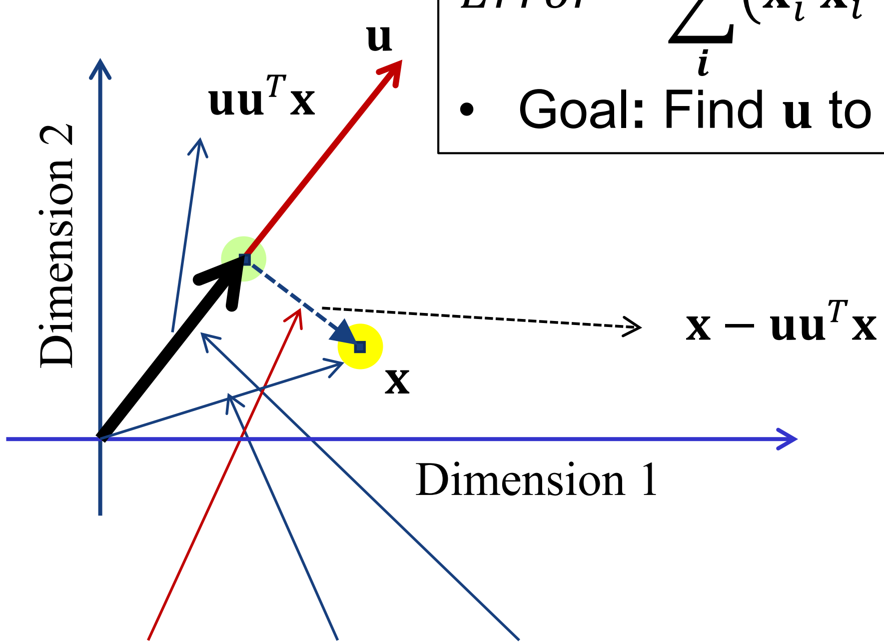
$\boxed{\mathbf{u}^T\mathbf{u}=1}$

Find the first basis vector

- Error for one signal vector $\mathbf{x}$ $Error = \mathbf{x}^T\mathbf{x} - \mathbf{x}^T\mathbf{u}\mathbf{u}^T\mathbf{x}$
- Error for all signal vectors $\mathbf{x}_i$

$$Error = \sum_{i}(\mathbf{x}_i^T\mathbf{x}_i - \mathbf{x}_i^T\mathbf{u}\mathbf{u}^T\mathbf{x}_i) = \sum_{i}\mathbf{x}_i^T\mathbf{x}_i - \sum_{i}\mathbf{x}_i^T\mathbf{u}\mathbf{u}^T\mathbf{x}_i$$

- Goal: Find $\mathbf{u}$ to minimize this error.



$$Error = \mathbf{x}^T\mathbf{x} - \mathbf{x}^T\mathbf{u}\mathbf{u}^T\mathbf{x}$$

Find the <u>first</u> basis vector

Find $\mathbf{u}$ to minimize the error $\sum_i \mathbf{x}_i^T\mathbf{x}_i - \sum_i \mathbf{x}_i^T\mathbf{u}\mathbf{u}^T\mathbf{x}_i$ subject to $\mathbf{u}^T\mathbf{u} = 1$

unit vector

Apply Lagrange multiplier α and set derivative (with respect to $\mathbf{u}$) to be zero

$$C = \sum_i \mathbf{x}_i^T\mathbf{x}_i - \sum_i \mathbf{x}_i^T\mathbf{u}\mathbf{u}^T\mathbf{x}_i + \alpha(\mathbf{u}^T\mathbf{u} - \mathbf{1})$$

$$-\mathbf{2}\sum_i \mathbf{x}_i\mathbf{x}_i^T\mathbf{u} + \mathbf{2}\alpha\mathbf{u} = \mathbf{0} \Rightarrow \boxed{\left(\sum_i \mathbf{x}_i\mathbf{x}_i^T\right)}\mathbf{u} = \alpha\mathbf{u}$$

- $\mathbf{u}$: eigenvector of matrix $\sum_i \mathbf{x}_i\mathbf{x}_i^T$
- α is eigenvalue

Minimize $C = \sum_i \mathbf{x}_i^T\mathbf{x}_i - \sum_i \mathbf{x}_i^T\mathbf{u}\mathbf{u}^T\mathbf{x}_i$	Recall $\mathbf{x}_i^T\mathbf{u} = \mathbf{u}^T\mathbf{x}_i$
$= \sum_i \mathbf{x}_i^T\mathbf{x}_i - \sum_i \mathbf{u}^T\mathbf{x}_i\mathbf{x}_i^T\mathbf{u}$	
$= \sum_i \mathbf{x}_i^T\mathbf{x}_i - \mathbf{u}^T\left(\sum_i \mathbf{x}_i\mathbf{x}_i^T\right)\mathbf{u} = \sum_i \mathbf{x}_i^T\mathbf{x}_i - \mathbf{u}^T\alpha\mathbf{u}$	Recall $\left(\sum_i \mathbf{x}_i\mathbf{x}_i^T\right)\mathbf{u} = \alpha\mathbf{u}$
$= \sum_i \mathbf{x}_i^T\mathbf{x}_i - \alpha\mathbf{u}^T\mathbf{u}$	Recall $\mathbf{u}^T\mathbf{u} = 1$
$= \sum_i \mathbf{x}_i^T\mathbf{x}_i - \alpha$	Choose the largest α to minimize this error

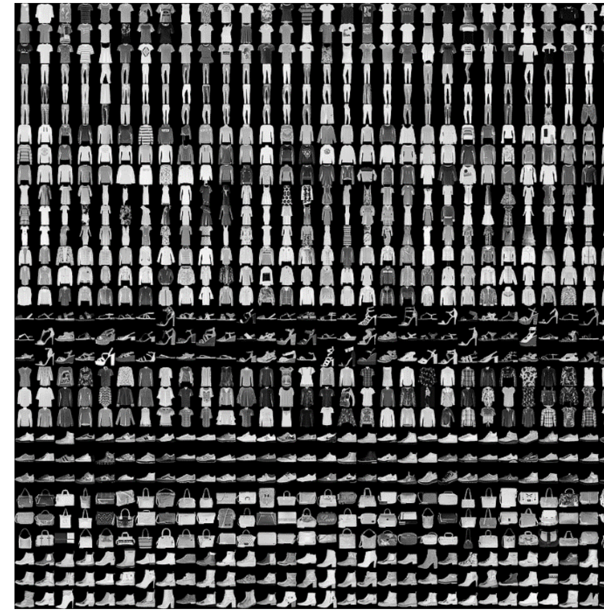
To find more basis

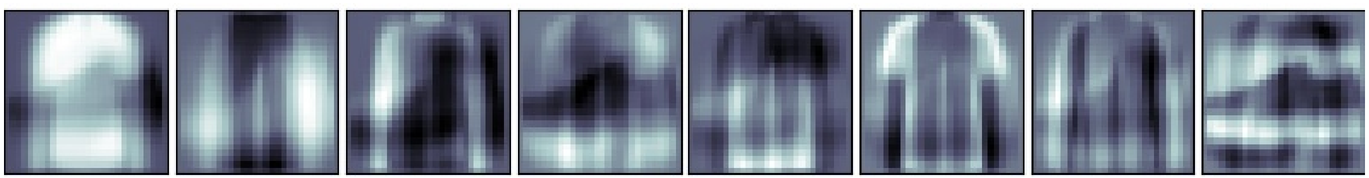
So far we already have the first basis, to get more basis, we need to

- Get the "error" signal $\mathbf{x} - \mathbf{u}_1 \mathbf{u}_1^T \mathbf{x}$ by subtracting the first-level approximation from the original signal.
- Treat the "error signal" as a new signal, and repeat the estimation on the "error" signal.
- Get the second-level "error" by subtracting the scaled second typical basis from the first-level error.
- Repeat the estimation on the second-level "error" signal.
- We can continue the process until we find N basis vectors.

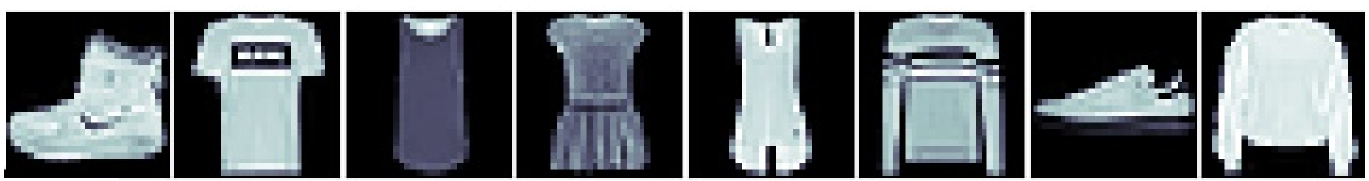
Example

Fashion-MNIST dataset,
https://github.com/zalandoresearch/fashion-mnist

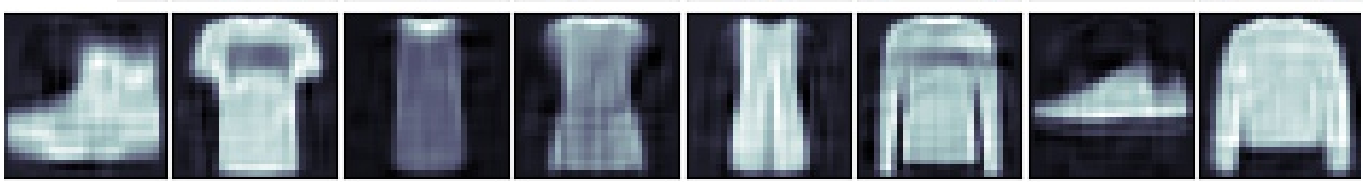




Examples of basis images



Original images



Reconstructed images (with 64 basis)

$$\hat{\mathbf{x}} = \sum_{k=1}^{64} w_k \mathbf{u}_k$$

Signal representation: Under-complete vs over-complete

Suppose we represent the signal $\mathbf{x}$ using N basis vectors as

$\mathbf{x} = \sum_{k=1}^{N} w_k \mathbf{u}_k$. Rewrite it into matrix form as $\mathbf{x} = [\mathbf{u}_1, \mathbf{u}_2, \cdots, \mathbf{u}_N] \begin{bmatrix} w_1 \\ w_2 \\ \vdots \\ w_N \end{bmatrix}$

Input data dimension $\mathrm{L} \times 1$

Basis vectors, each dimension $\mathrm{L} \times 1$

Number of basis vectors

- When #(Basis vectors) = dim(Input data), $N = L$, exact reconstruction
- When #(Basis vectors) < dim(Input data), $N < L$, under-complete, sparse
- When #(Basis vectors) > dim(Input data), $N > L$, over-complete, redundancy

While techniques such as Principal Component Analysis (PCA) allow us to learn a complete set of basis vectors efficiently, we wish to learn an **over-complete** set of basis vectors to represent input vectors $\mathbf{x} \in \mathbb{R}^n$ (i.e. such that $k > n$). The advantage of having an over-complete basis is that our basis vectors are better able to capture structures and patterns inherent in the input data. However, with an over-complete basis, the coefficients a_i are no longer uniquely determined by the input vector $\mathbf{x}$.

Quoted from Andrew Ng's tutorial, http://ufldl.stanford.edu/tutorial/unsupervised/SparseCoding

Summary

- Data driven signal representation
- Signal representation learning

Thank you!

Dr TIAN Jing
Email: tianjing@nus.edu.sg