ITCT Lecture 10.1:

- Prof. Ja-Ling Wu
- Dept. CSIE and
  GINM,

National Taiwan University

Taipei, Taiwan

- Transform Coding Techniques

Transform Coding

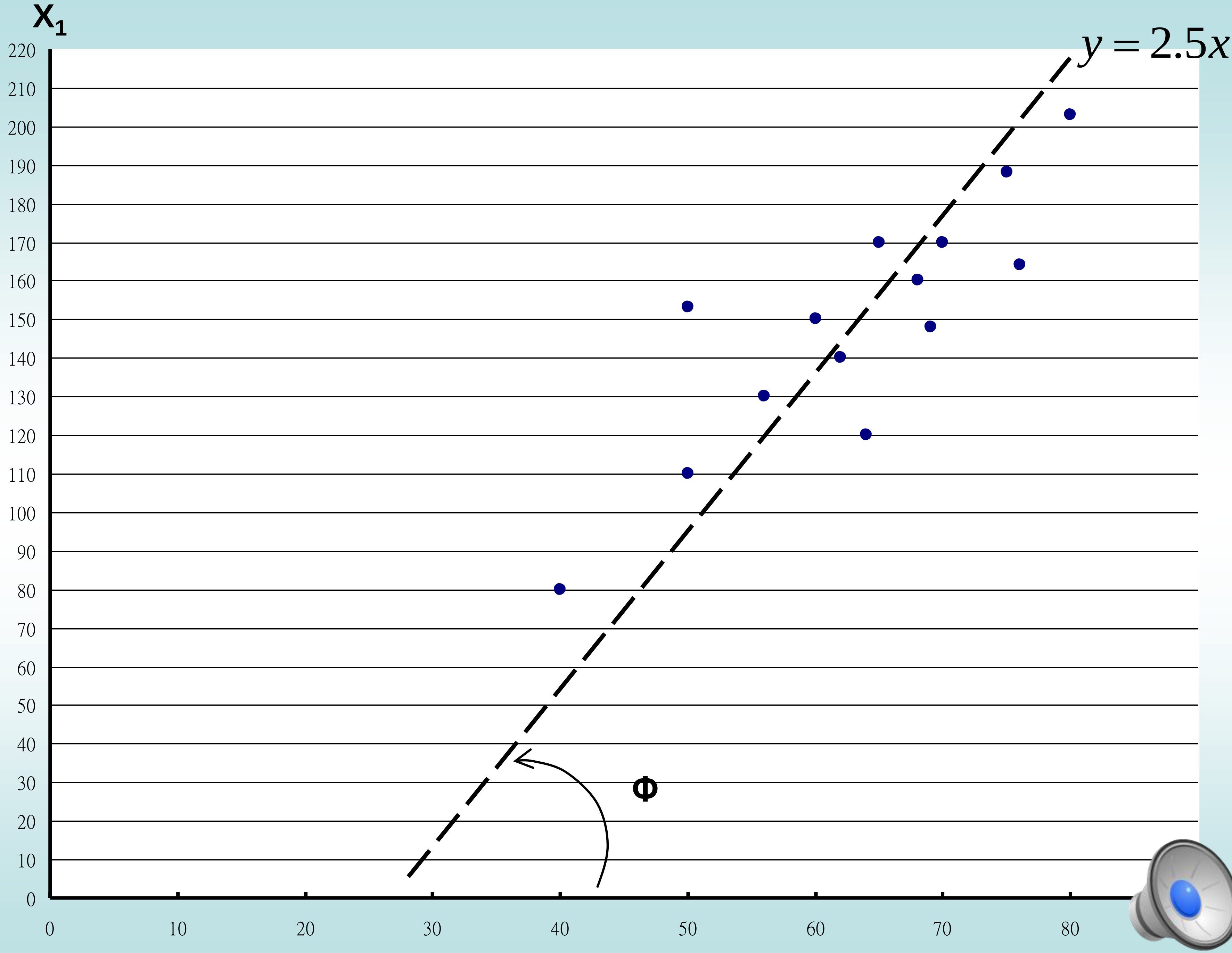
Example:

Original sequence $X=(x_0,x_1)^t$

Weight	Height
65	170
75	188
60	150
70	170
56	130
80	203
68	160
50	110
40	80
50	153
69	148
62	140
76	164
64	120

Transformed sequence $\theta=(\theta_0,\theta_1)^t$

Height	Weight
182	3
202	0
162	0
184	-2
141	-4
218	1
174	-4
121	-6
90	-7
161	10
163	-9
153	-6
181	-9
135	-15



X_1
220
210
200
190
180
170
160
150
140
130
120
110
100
90
80
70
60
50
40
30
20
10
0
0
10
20
30
40
50
60
70
80
$y = 2.5x$
Φ

Since the output values tend to cluster around the line $y = 2.5x$. We can rotate the set of original sequence values by the transform

$$\theta = AX$$

where X is the 2-D source output vector

$$X = \begin{bmatrix} x_0 \\ x_1 \end{bmatrix} = [x_0, x_1]^t$$

A is the rotation matrix

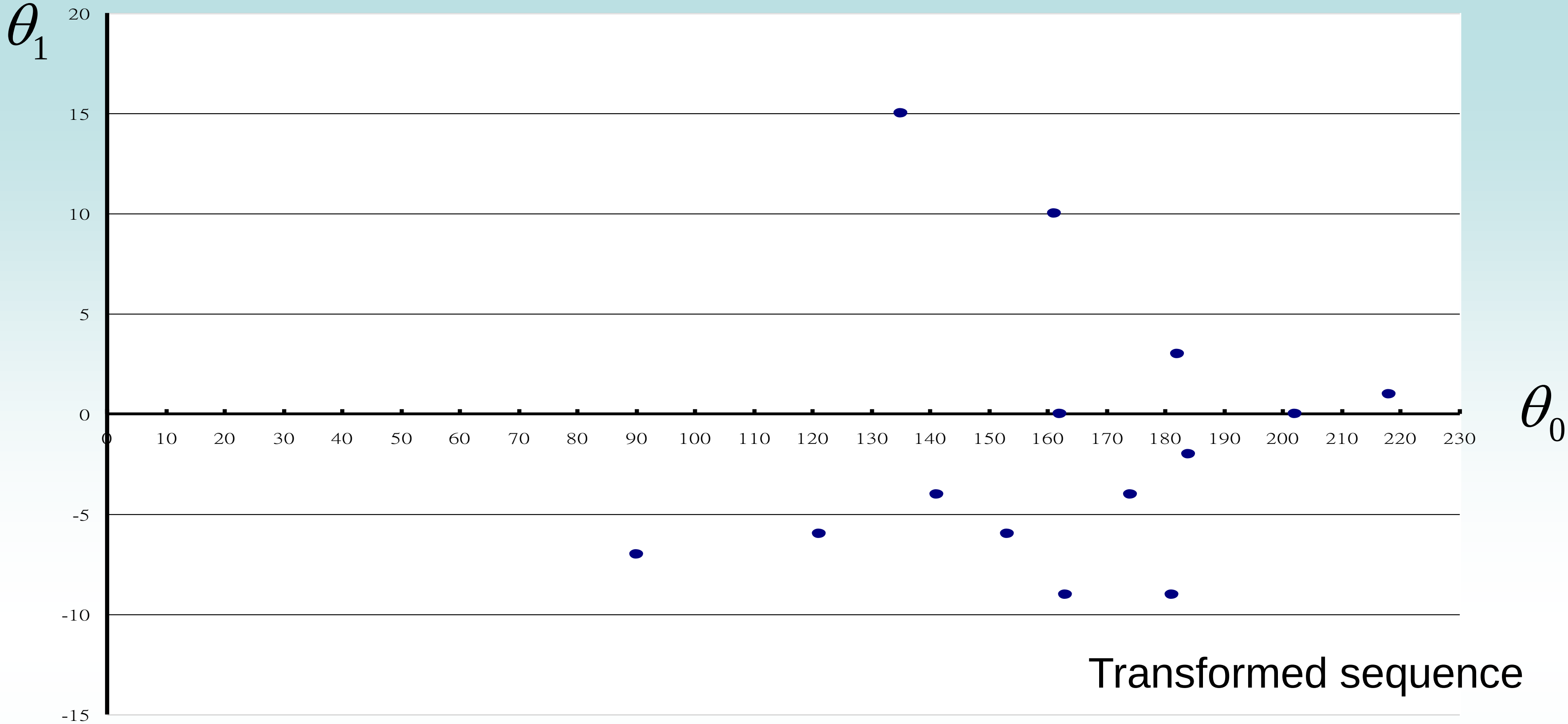
$$A = \begin{bmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{bmatrix}$$

ϕ is the angle between the x-axis and the $y = 2.5x$ line, and

$$\theta = \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix} = [\theta_0, \theta_1]^t$$

is the rotated or transformed set of values.
For this particular case

$$A = \begin{bmatrix} 037139068 & 0.92847669 \\ -0.92847669 & 0.37139068 \end{bmatrix}$$



Notice that for each pair of the **transformed values**, almost all the **energy** is **compacted into** the **first element** of the pair, while the 2nd element of the pair is significantly smaller.

$\Rightarrow$

we have rotated the original axes (x_0,x_1) to the new axes (θ_0,θ_1) by an angle of approximately $68° \approx \tan^{-1} 2.5$

Approximation Reconstruction:

S’pose we set all **the 2nd elements** of the transformation (i.e., θ_1) to **zero**. This reduces the number of elements that need to be coded by half.

What is the effect of throwing away half the elements of the sequence?

We can answer this question by taking the inverse transform (i.e., inverse rotation) of the reduced sequence.

$$\begin{bmatrix}\hat{x}_0 \\ \hat{x}_1\end{bmatrix} = \underbrace{\begin{bmatrix}\cos\phi & -\sin\phi \\ \sin\phi & \cos\phi\end{bmatrix}}_{A^{-1}} \begin{bmatrix}\theta_0 \\ 0\end{bmatrix}$$

Reconstructed sequence $[x_0, x_1]^t$

Weight	Height
68	169
75	188
60	150
68	171
53	131
81	203
65	162
45	112
34	84
60	150
61	151
57	142
67	168
50	125

Comparing the reconstructed sequence with the original sequence, we see that,

transmitted
memorized

even though we **<u>transformed</u>** only half the number of elements presented in the original sequence, the reconstructed sequence is very **<u>close to</u>** the original.

The reason there is so little error introduced in the sequence $\{x_n\}$ is that for this particular transformation (**linear and Invertible**), the error introduced into the $\{x_n\}$ sequence is equal to the error introduced into the sequence $\{\theta_n\}$. That is,

$$\sum (X - \hat{X})^2 = \sum (\theta - \hat{\theta})^2$$

We could reduce the number of samples we needed to code because most of the information (energy) contained in each pair of values was put into one element of each pair!! Because the other element of the pair contained very little information, we could discard it without a significant effect on the fidelity of the reconstructed sequence!!

From a vector pair to a **block** of source data:

By compacting most of the information (energy) in a source output sequence into a few elements of the transformed sequence using a **Reversible Transform**, and then discarding the elements of the sequence that do not contain much information, we can get a large amount of compression.

Statistical View of Transform Coding:

We can get the maximum amount of compaction if we use a transform that **Decorrelates** the input sequence; that is, the **sample-to-sample correlation** of the **transformed sequence** is **zero**!!

The first transform to provide decorrelation for discrete data was presented by **Hotelling** in the Journal of Education Psychology in 1933. He called his approach the method of **principal components**.

The analogous transform for continuous functions was obtained by **Karhunen** (1947) and **Loeve** (1948).

– Hotelling transform, principal component analysis, Karhunen-Loeve transform.

This decorrelation approach was first utilized for compression, in what we call transform coding, by Kramer and Mathews (1956) and Huang and Schultheiss (1963: IEEE Trans. on Communication Systems).

Transform coding consists of three steps:

(i) Data sequence $\{X_n\}$ is divided into blocks of size N. Each block is mapped into a transformed sequence $\{\theta_n\}$ using a reversible mapping

(ii) Quantizing the transformed sequence. The quantization strategy used will depend on three main factors:

- **the desired averaged bit rate**
- **the statistics of the various elements of the transformed sequence**
- **the effect of distortion in the transformed coefficients on the reconstructed sequence.**

In the previous example, we take all available bits to quantize the first coefficient, in more complex situations, the strategy used may be very different. ─

bit allocation problem!!

(iii) Encoding the quantized value using some binary encoding techniques.
─run-length coding, Huffman coding, arithmetic coding, …

All the transforms we used will be linear transforms; that is,

$$\theta_n = \sum_{i=0}^{N-1} x_i \, a_{n,i} \quad \text{… forward transform}$$

A major difference between the transformed sequence $\{\theta_n\}$ and the original sequence $\{x_n\}$ is that the characteristics of the elements of the q sequence are determined by their <u>position</u> within the sequence.

The transform domain is composed of a set of weighted axes, and therefore, the significance of transformed coefficients is index dependent.

A measure of the differing characteristics of the different elements of the transformed sequence $\{\theta_n\}$ is the <u>variance</u> σ_n^2 of each element. These variances will strongly influence how we encode the transformed sequence.

Block size N:

The size of the block <u>N</u> is dictated by practical considerations. In general, the <u>complexity</u> of the transform grows more linearly with N.

- Therefore, beyond a certain value of N, the computational costs overwhelm any marginal improvements that might be obtained by increasing N.
- In most real sources the statistical characteristics of the source output can change abruptly.

Silence

$\updownarrow$

Voiced speech

If N is large, the probability that the statistical characteristics change significantly within a block increases.

This generally results in a large number of the transform coefficients with large values, which in turn leads to a reduction in the compression ratio.

The original sequence $\{x_n\}$ can be recovered from the transformed sequence $\{\theta_n\}$ via the <u>inverse transform</u>

$$x_n = \sum_{i=0}^{N-1} \theta_i b_{n,i}$$

The transforms can be written in matrix form as

$$\theta = AX$$

$$X = B\,\theta$$

Where A and B are NxN matrices and the (i,j)-th element of the matrices are given by

$[A]_{i,j} = a_{i,j}$; $[B]_{i,j} = b_{i,j}$. **AB=BA=I**

2-D Transformations:

Let Xi,j be the (i,j)th pixel in an image. A general linear 2-D transform for a block of size NxN is given as

$$\Theta_{k,l} = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} x_{i,j} a_{i,j,k,l}$$

All 2-D transforms in use today are **separable** transforms; that is, we can take the transform of a 2-D block by first taking the transform along one dimension, then repeating the operation along the other direction. In terms of matrices, this involves first taking the 1-D transform of the rows, and then taking the column-by-column transform of the resulting matrix.

We can also reverse the order of the operations, i.e., column transforms first. The transform operation can be represented as

$$\Theta_{k,l} = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} x_{i,j} a_{k,i} \cdot a_{l,j}$$

Which in matrix terminology would be given by

$$\Theta = A \; X \; A^T$$

The inverse transform is given as

$$X = B \, \Theta \, B^T$$

All the transforms we deal with will be orthonormal transforms. An orthonormal transform has the property that the inverse of the transform matrix is simply its transpose:

$B = A^{-1} = A^{T}$.

Therefore, the inverse transform becomes:

$X = A^{T}\,\Theta\,A$

Orthonormal transforms are energy preserving:

The sum of the squares of the transformed sequence is the same as the sum of the squares of the original sequence.

Example : 1-D orthonormal transform

$$\sum_{i=0}^{N-1} \theta_i^2 = \theta^{\mathrm{T}} \theta$$

($A^TA = A^{-1}A = I$)

$$= (AX)^T AX$$

$$= X^T A^T AX = X^T X = \sum_{n=0}^{N-1} x_n^2$$

The efficiency of a transform depends on how much energy compaction is provided by the transform. One way of measuring the amount of energy compaction afforded by a particular transform is to take a ratio of the **arithmetic mean** of the **variances** of the transform coefficients to their **geometric mean**:

Transform coding Gain $\triangleq$ GTC = $$\frac{\frac{1}{N}\sum_{i=0}^{N-1}\sigma_i^2}{\left(\prod_{i=0}^{N-1}\sigma_i^2\right)^{\frac{1}{N}}}$$

Where σ_i^2 is the variance of the i-th coefficient θ_i .

Transforms can also be interpreted as a decomposition of the signal in terms of a basis set:

For example. Suppose we have a 2-D orthonormal transform A. The inverse transform can be written as

$$\begin{bmatrix} x_o \\ x_1 \end{bmatrix} = \begin{bmatrix} a_{00} & a_{01} \\ a_{10} & a_{11} \end{bmatrix} \begin{bmatrix} \theta_0 \\ \theta_1 \end{bmatrix}$$

$$= \theta_0 \begin{bmatrix} a_{00} \\ a_{01} \end{bmatrix} + \theta_1 \begin{bmatrix} a_{10} \\ a_{11} \end{bmatrix}$$

We can see that the transformed values are actually the coefficients of an expansion of the input sequence in terms of the columns of the transform matrix.

The columns of the transform matrix are often referred to as the basis vectors of the transform, and the elements of the transformed sequence are often called the transform coefficients.

Different transform $\longleftrightarrow$ Different Basis Vectors

Similarly, we can interpret 2-D transform as experience in terms of matrices that are formed by the outer product of the columns of the transform matrix.

Recall that the outer product is given by

$$
XX^T = \begin{pmatrix}
x_0 x_0 & x_0 x_1 & \dots & x_0 x_{N-1} \\
x_1 x_0 & x_1 x_1 & \dots & x_1 x_{N-1} \\
. & . & . & . \\
. & . & . & . \\
. & . & . & . \\
x_{N-1}\, x_0 & x_{N-1}\, x_1 & \dots & x_{N-1}\, x_{N-1}
\end{pmatrix}
$$

For an NXN transform A, let $\alpha_{i.j}$ be the outer product of the ith and jth columns:

$$\alpha_{i.j} = \begin{pmatrix} a_{i0} \\ a_{i1} \\ \ldots \\ a_{iN-1} \end{pmatrix} \begin{pmatrix} a_{j0} \; a_{j1} \ldots a_{jN-1} \end{pmatrix}$$

:basis matrices

basis images

$$= \begin{pmatrix} a_{i0}a_{j0} & a_{i0}a_{j1} & \ldots & a_{i0}a_{jN-1} \\ a_{i1}a_{j0} & a_{i1}a_{j1} & \ldots & a_{i1}a_{jN-1} \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ a_{iN-1}\, a_{j0} & a_{iN-1}\, a_{j1} & \ldots & a_{iN-1}\, a_{jN-1} \end{pmatrix}$$

The transform values θ_{ij} can be viewed as the coeffs. of the expansion of $x_{i,j}$ in terms of the matrices $\alpha_{i.j}$

Karhunen-Loeve Transform

The columns of the Karhunen-Loeve transform, also known as the Hotelling transform, consists of the **eigenvectors** of the **autocorrelation matrix**.

The autocorrelation matrix for a random process X is a matrix whose (i,j)th element $[R]_{i,j}$ is given by

$$[R]_{i,j} = E[X_n \, X_{n+|i-j|}]$$

A transform constructed in this manner will minimize the **geometric mean** of the variance of the transform coeffs. Hence, the KLT provides the largest transform coding gain of any transform coding method.

$\longrightarrow$ **maximal Decorrelation process**

$$\theta_i = \sum_{n=0}^{N-1} x_n a_{n,i}$$

$$[A]^{-1} = [A]^t$$

$$X_n = \sum_{i=0}^{N-1} \theta_i a_{i,n}$$

Target = θ_i uncorrelated

A is a matrix whose columns are the normalized eigenvectors of the covariance matrix of the original pixels.

The covariance matrix of x_n:

$$C_x = E\{(x_n - E(x_n))\,(x_n - E(x_n))^t\}$$

Assume $E\{x_n\} = 0$ and set $x_n = \{x_0, x_1, \ldots, x_{N-1}\}$

$$C_x = \begin{pmatrix} E(x_0^2) & E(x_0 x_1) & \cdots & E(x_0 x_{N-1}) \\ E(x_1 x_0) & E(x_1^2) & \cdots & E(x_1 x_{N-1}) \\ . & . & . & . \\ . & . & . & . \\ . & . & . & . \\ E(x_{N-1} x_0) & E(x_{N-1} x_1) & \cdots & E(x_{N-1}^2) \end{pmatrix}$$

Let ϕ denote the eigenvectors of C_x:

$C_x \phi = \lambda \phi$ i.e., $\det[C_x - \lambda I] = 0$

Arrange λ's in decreasig order such that

$$\lambda_0 \geq \lambda_1 \geq \ldots \geq \lambda_{N-1}$$

Cx:Hermitian Matrix

And substitute into $(C_x - \lambda I)\phi = 0$ to solve for ϕ

When the matrix A (whose rows are the ϕ functions) is applied to Xn, the covariance of the resulting coeffs. θ_i is a diagonal matrix with diagonal elements

$$\lambda_0, \lambda_1, \ldots, \lambda_{N-1}$$

λ_{N-1}

θ_i is uncorrelated

That is

$$C_\theta = E\{(\theta - E(\theta))(\theta - E(\theta))^t\}$$

$$= E[\theta\theta^t]$$

: zero-mean assumption

$$= E\{(Ax)(Ax)^t\}$$

A: unitary matrix

$$= E\{Axx^t A^t\}$$

C_θ and C_x are similar

$$= AE\{xx^t\}A^t$$

$$= AC_x A^t = \begin{pmatrix} \lambda_0 & & & 0 \\ & \lambda_1 & & \\ & & \dots & \\ 0 & & & \lambda_{N-1} \end{pmatrix}$$

The KLT decorrelates the original input