Advanced Transform Methods

Karhunen-Loeve Transform

Andy Watson

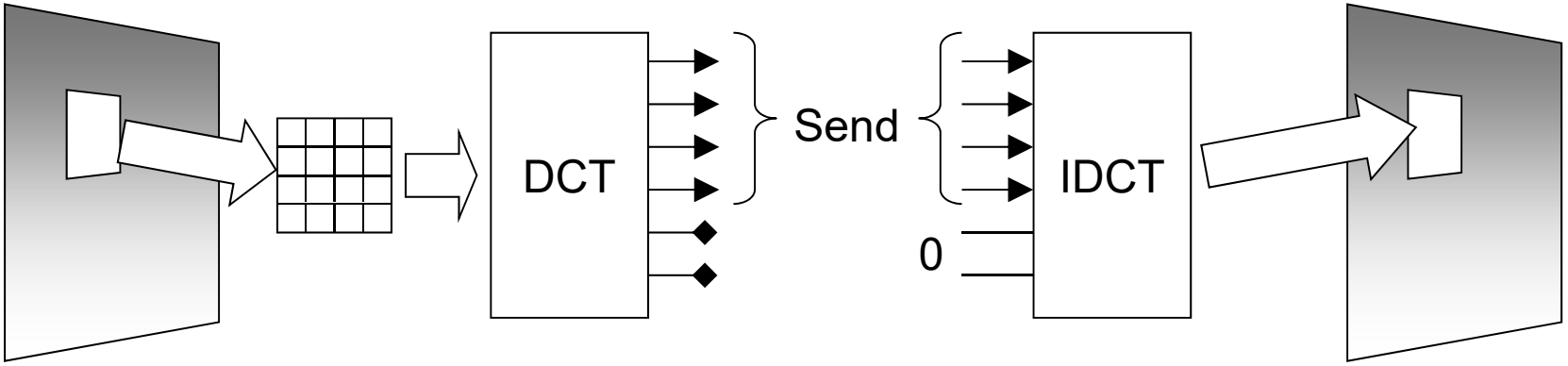
Karhunen Loeve Transform

Met Discrete Cosine Transform (DCT) in a previous lecture.

An advantage of DCT:

- most energy concentrated in a few coefficients, so
- can discard some coeffs, while keep most of signal

E.g. image compression:



Fourier, Wavelets also do this, depending on the signal.

But - what is the “best” transform for this?

Principal Components Analysis (PCA)

- Multivariate procedure
- Main use of PCA is to reduce dimensionality of a data set while retaining as much information as is possible.
- Finds a projection of the observations onto orthogonal axes contained in the space defined by the original variables.
- Correlated variables transformed into uncorrelated variables
  - Ordered by reducing variability.
  - Uncorrelated variables are linear combinations of original variables
- Computes compact, optimal description of data set.
- Rotates data so that maximum variabilities projected onto the axes
- Rotation of existing axes to new positions in the space defined by the original variables.

The new components/ axes/ variables

1. First principal component is the combination of variables that explains the greatest amount of variation
   - ✓ Contains the maximum amount of variation
2. Second principal component defines the next largest amount of variation and is independent to the first principal component
   - ✓ Contains the maximum amount of variation unexplained by and orthogonal to the first
3. Third axis contains the maximum amount of variation orthogonal to the first and second axis
   - ✓ Contains the maximum amount of variation unexplained by and orthogonal to the first and second axes

… Last new axis which is the last amount of variation left

- ✓ Can be removed with minimum loss of real data

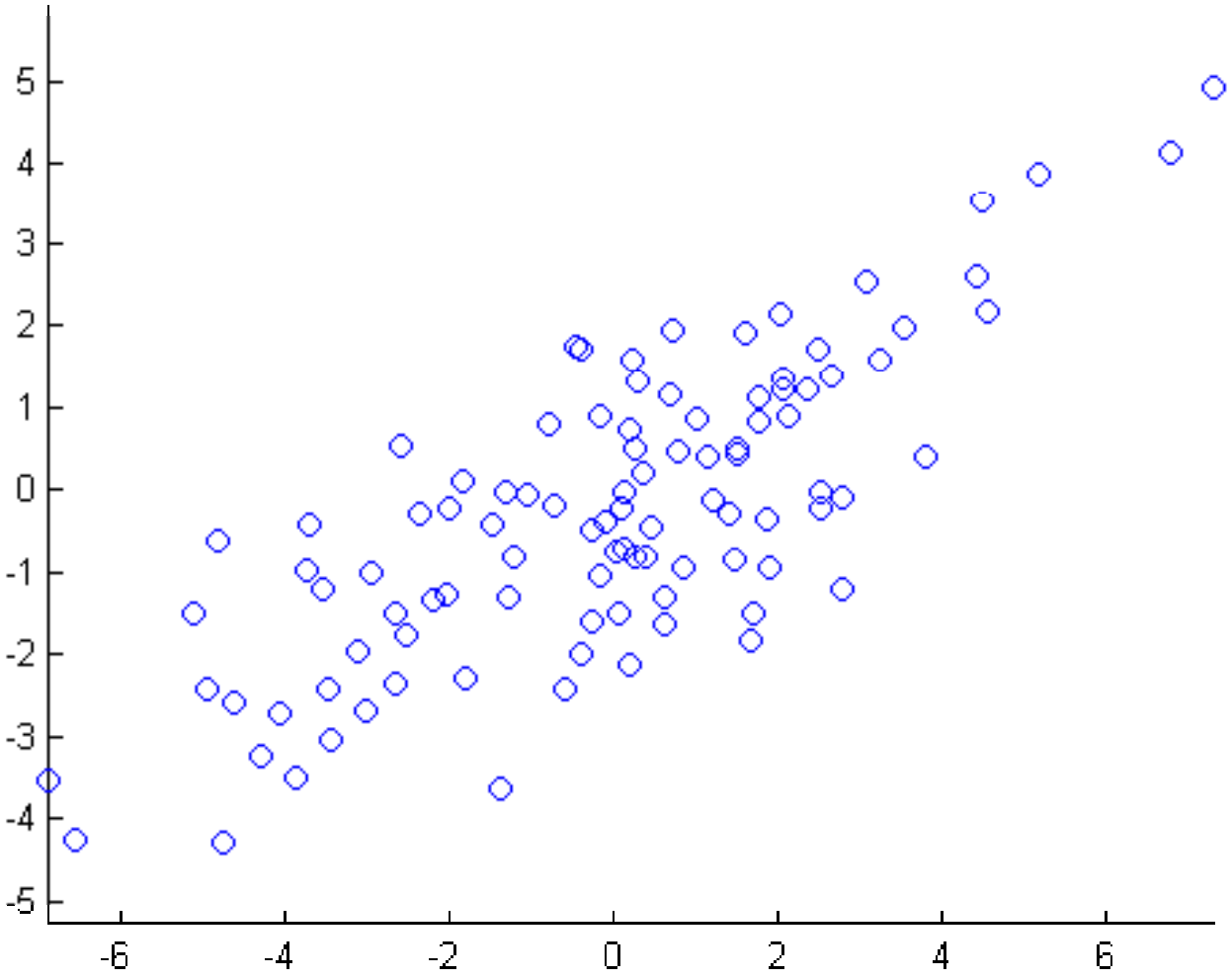
- ❑ Can be as many principal components as there are variables
- ❑ No correlation between the new variables defined by rotation

Example

- Set of 2-D data

x=

-3.0139	-2.6748
0.1810	-2.1227
-6.5559	-4.2424
1.5960	1.9161
2.7825	-1.2017
2.0292	2.1421
-2.5200	-1.7787
0.2340	-0.8122
0.8263	-0.9407
-4.6227	-2.5955
-4.3052	-3.2307
-2.6013	0.5435
3.5409	1.9755
5.1704	3.8629
.	.
.	.
.	.



Covariance Matrix

- Find covariance matrix

$$\mathrm{cov}[\mathbf{X}] = \begin{bmatrix} 7.5649 & 3.8464 \\ 3.8464 & 3.2451 \end{bmatrix}$$

- Find principal components

$$\mathbf{pc} = \begin{bmatrix} 0.8630 & -0.5052 \\ 0.5052 & 0.8630 \end{bmatrix}$$

- Are they orthogonal?

$$\langle \psi_1, \psi_2 \rangle = \langle (0.8630, 0.5052), (-0.5052, 0.8630) \rangle$$

$$= 0.8630 \cdot (-0.5052) + 0.5052 \cdot 0.8630 = 0$$ **Yes**

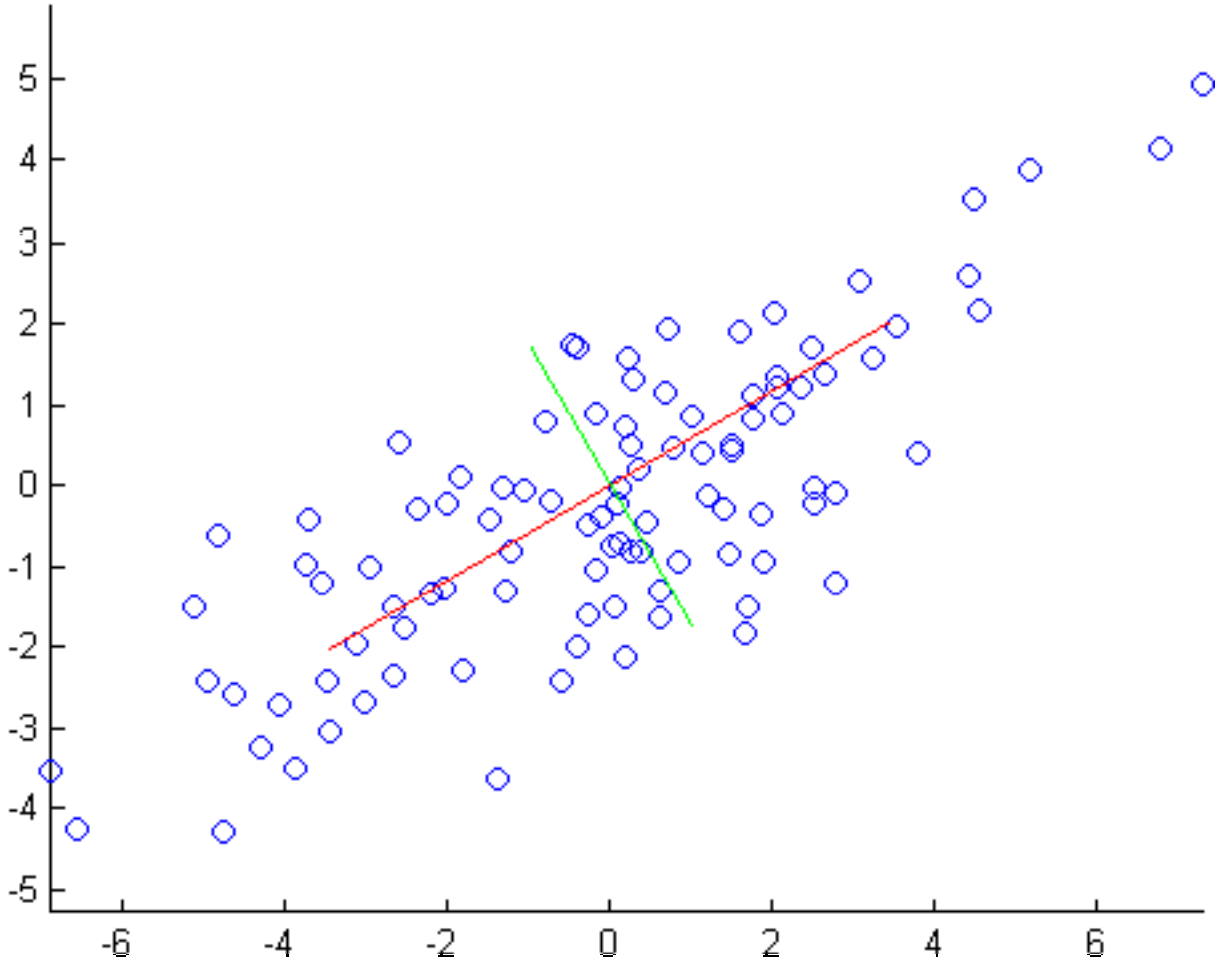
- Are they normalised?

$$\langle \psi_1, \psi_1 \rangle = 0.8630 \cdot 0.8630 + 0.5052 \cdot 0.5052 = 1$$

$$\langle \psi_2, \psi_2 \rangle = (-0.5052) \cdot (-0.5052) + 0.8630 \cdot 0.8630 = 1$$ **Yes**

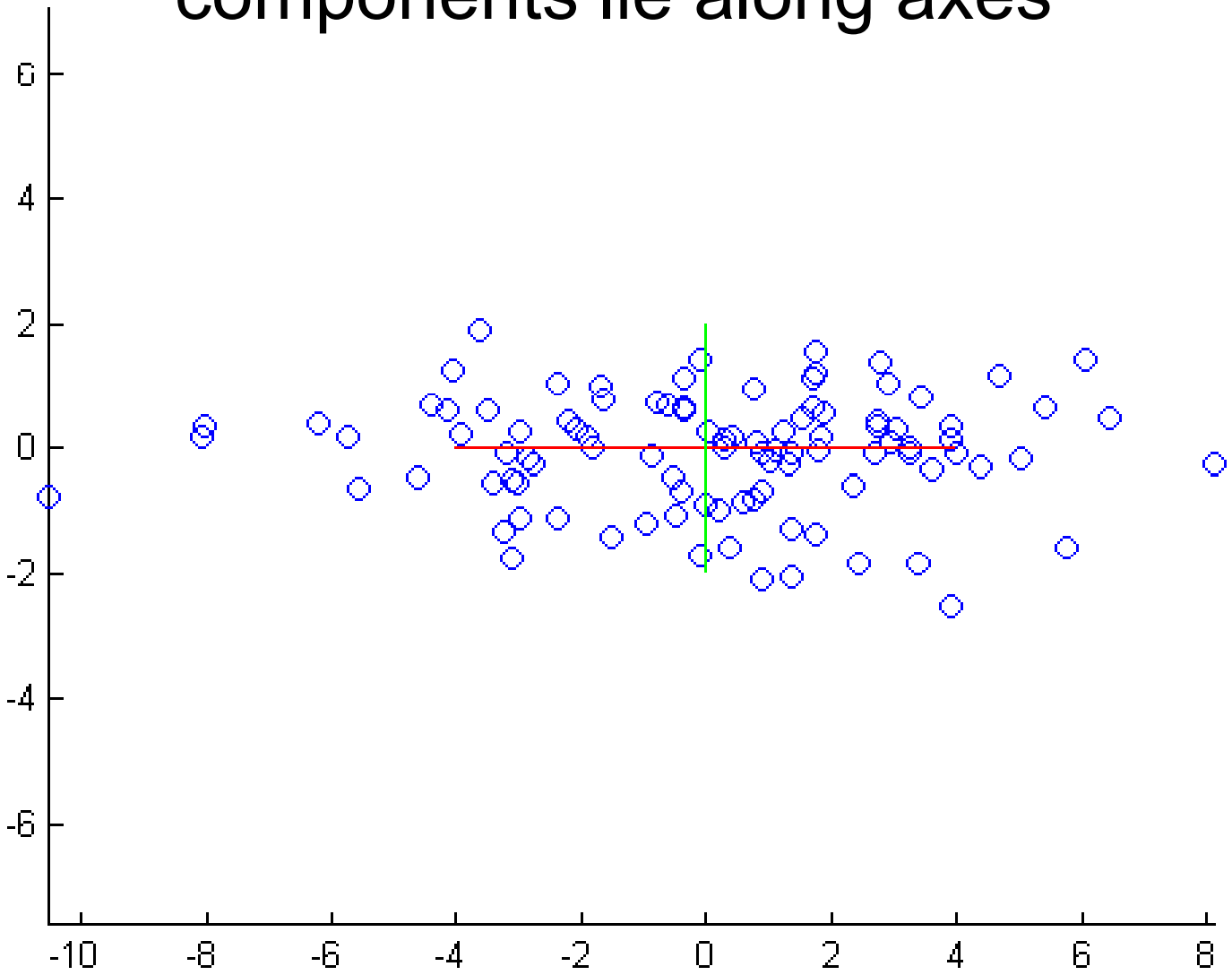
Principal components visualised

- red line represents direction of first principal component
  - line of greatest variation
- green line is direction of second principal component
  - perpendicular to red line.
- When there are more than 2 dimensions,
  - next component along line of next greatest variation

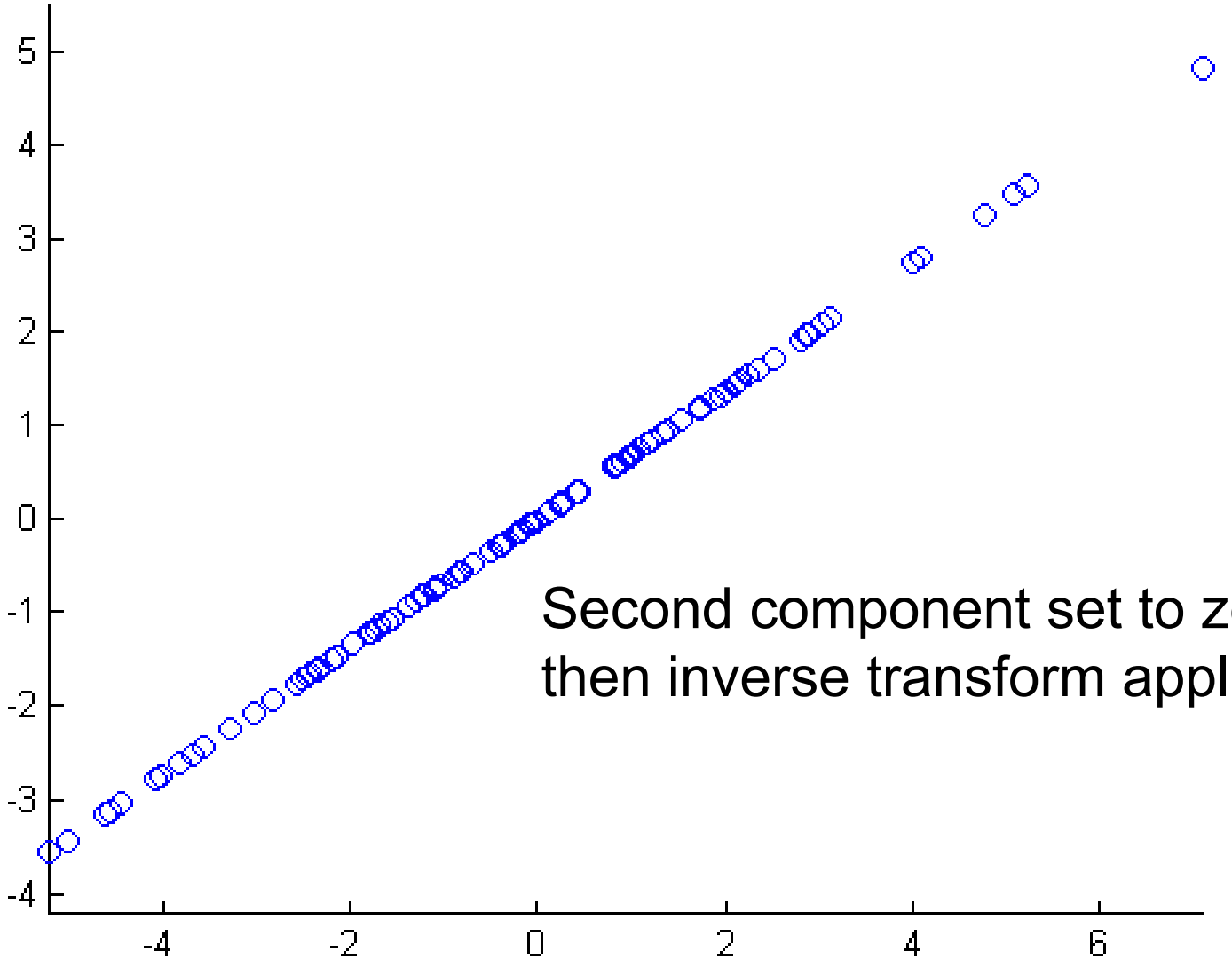


By multiplying original data by principal components, data rotated so that principal components lie along axes

- :



PCA is used to reduce data dimensionality while retaining the most information



Now let's give the theory

- The Karhunen Loève Transform is based on PCA
- Also known as the Hotelling Transform

Karhunen Loève Transform (KLT)

- Basis functions are eigenvectors of the covariance matrix R_{XX} of the input signal.
  - This set of basis vectors is *not* fixed
  - Basis vectors depend on the data set
- KLT yields decorrelated transform coefficients (covariance matrix R_{YY} is diagonal).
- Optimal linear transform for keeping the subspace that has largest variance
  - Achieves optimum energy concentration.
- KLT maximizes coding gain

Procedure

1. Find mean vector for input data $\mathbf{X} = \left[\vec{x}_0, \vec{x}_1, ... \vec{x}_{N-1} \right]$

$$E(\vec{x}) = \frac{1}{N} \sum_{i=0}^{N-1} \vec{x}_i$$

2. Find covariance matrix

$$\mathbf{R_{XX}} = \frac{1}{N-1} \sum_{i=0}^{N-1} (\vec{x}_i - E(\vec{x}))(\vec{x}_i - E(\vec{x}))^T$$

3. Find eigenvalues of the covariance matrix

$$\left| \mathbf{R_{XX}} - \lambda \mathbf{I} \right| = 0$$

$\left|\mathbf{A}\right|$ means "Determinant of $\mathbf{A}$"

4. Find eigenvectors of the covariance matrix

$$\left[\mathbf{R_{XX}} - \lambda_i \mathbf{I} \right] \vec{\phi}_i = 0$$

Procedure

5. Normalise the eigenvectors

$$\left\langle \vec{\phi}_i, \vec{\phi}_i \right\rangle = 1$$

6. Transform the input

$$\mathbf{Y} = \phi^T \mathbf{X}$$

7. *To check*, find covariance matrix of $\mathbf{Y}$

$$\mathbf{R}_{\mathbf{YY}} = \frac{1}{N-1} \sum_{i=0}^{N-1} (\vec{y}_i - E[\vec{y}])(\vec{y}_i - E[\vec{y}])^T$$

8. *Optional,*

    1. set last row vector(s) of $\mathbf{Y}$ to 0

$$\mathbf{Y} \rightarrow \mathbf{Y'}$$

    2. Inverse transform this

$$\mathbf{X}' = \phi \mathbf{Y}'$$

Linear transform coding

Divide image (or signal) into P blocks of N pixels (samples).

The kth block is now an N - dimensional vector :

$$\mathbf{x}_k = (x_{1,k}, x_{2,k}, \ldots, x_{N,k})^T$$

The image (signal) is now a sequence of vectors $\{\mathbf{x}_k\}$.

We now transform each $\mathbf{x}$ by multiplying by a linear matrix

$$\mathbf{y} = \mathbf{A}\mathbf{x}$$

transmit the first M coefficients $\hat{\mathbf{y}} = (y_1, \ldots, y_M)^T$,

discarding the remaining $N - M$ coeffs $y_{M+1} \cdots y_N$

We then reconstruct the image block using another matrix

$$\hat{\mathbf{x}} = \mathbf{B}\hat{\mathbf{y}}$$

Linear transform coding (cont)

We measure the error introduced in $\hat{\mathbf{x}}$ as

$J = E(|\mathbf{x} - \hat{\mathbf{x}}|^2)$ *mean squared error* (MSE)

where $E(v)$ is the expected value (mean) of v.

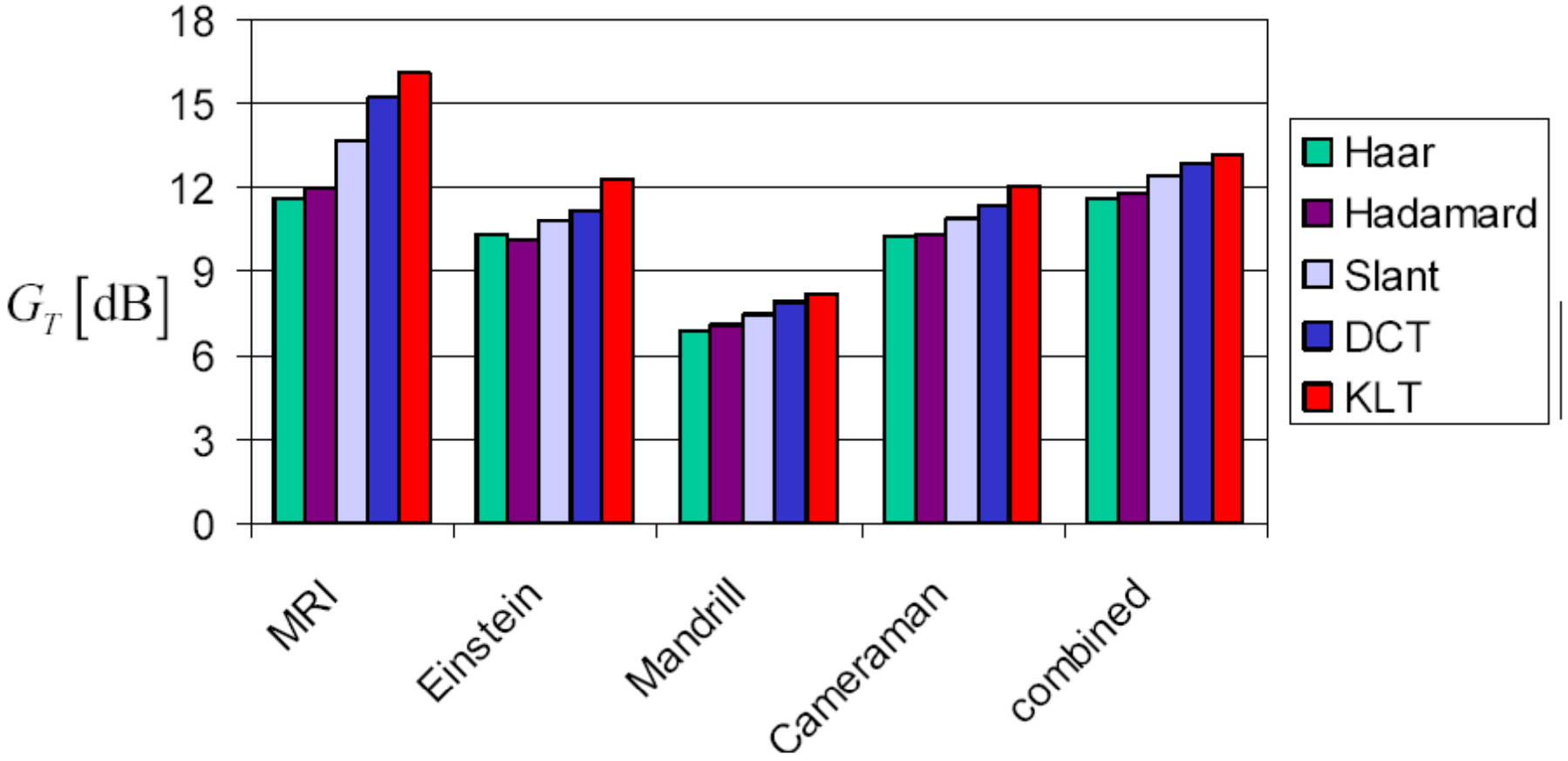
We want to choose $\mathbf{A}$ and $\mathbf{B}$ to minimize J.

Easy case: if we keep all the coefficients, we have

$\hat{\mathbf{y}} = \mathbf{y}$ so $\hat{\mathbf{x}} = \mathbf{B}\hat{\mathbf{y}} = \mathbf{B}\mathbf{y} = \mathbf{B}\mathbf{A}\mathbf{x}$

so $\hat{\mathbf{x}} = \mathbf{x}$ (making $J = 0$) if $\mathbf{B}\mathbf{A} = \mathbf{I}$ i.e. $\mathbf{B} = \mathbf{A}^{-1}$

Coding gain with 8x8 transforms



Drawbacks of KLT

KLT is theoretically optimal (in the MSE sense). (KLT maximises the coding gain, i.e. maximises the SNR after a given level of compression.)

BUT, it has practical difficulties:

- Estimate of correlation can be unwieldy
- Solution of eigenvector decomposition is computationally intensive (i.e. slow)
- Calculation of forward and inverse transforms is $O(MN)$ for each image block
- Transmission of data-dependent basis $\mathbf{A}$ is required
- The technique is linear, therefore any non-linear correlation between variables will not be captured.

In comparison, turns out that DCT is fixed, and

- a good approximation to KLT for typical images,
- needs no eigenvalue decomposition, and
- transform is $O(N \log N)$.