AIMS CDT - Signal Processing

Michaelmas Term 2023

Xiaowen Dong

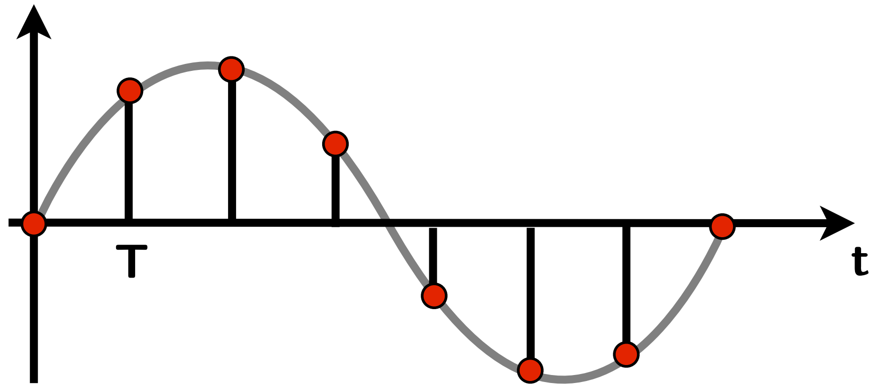
Department of Engineering Science

UNIVERSITY OF OXFORD

Representation of Signals

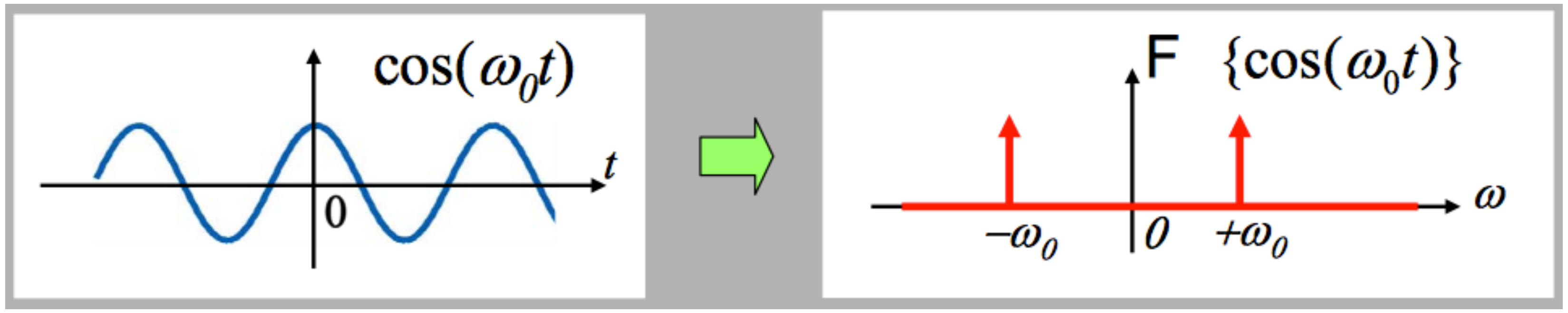
What is a representation of a signal?

- Sum of delta functions in time or space (sampling domain)
  - good for display or playback
  - not good for analysis (e.g., denoising, compression)



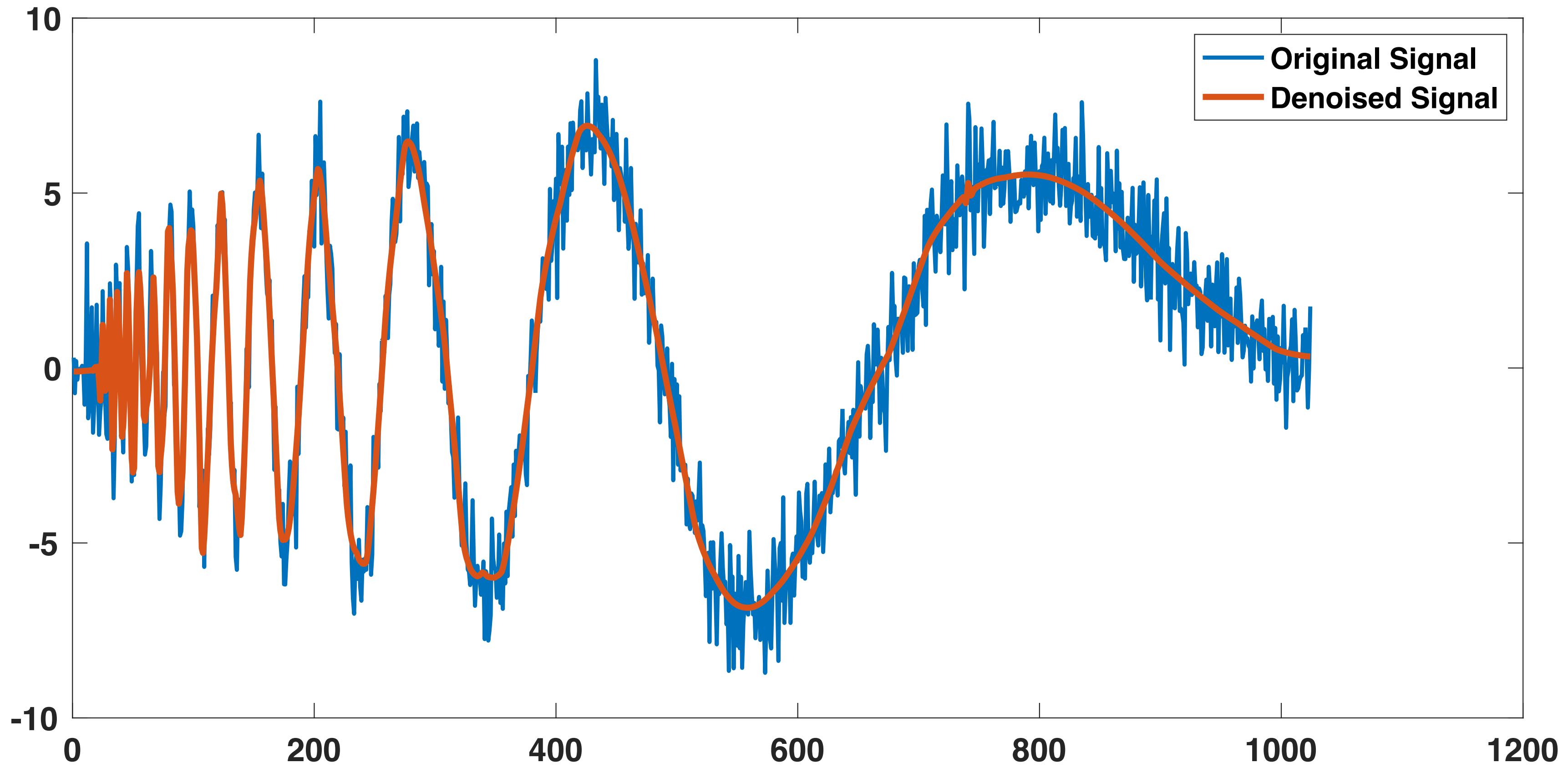
What is a representation of a signal?

- In a time-series setting, useful representation could be past samples
- More generally, it involves transformation of the signal into a new domain where signal characteristics are revealed
  - example: **Fourier coefficients** reveal **rate of change** of the signal



- Usefulness of the representation depends on the analysis goal
  - which may vary but all share the core desire for **simplification**

Example: Denoising



goal: recover signal from noisy observation

Example: Compression

original

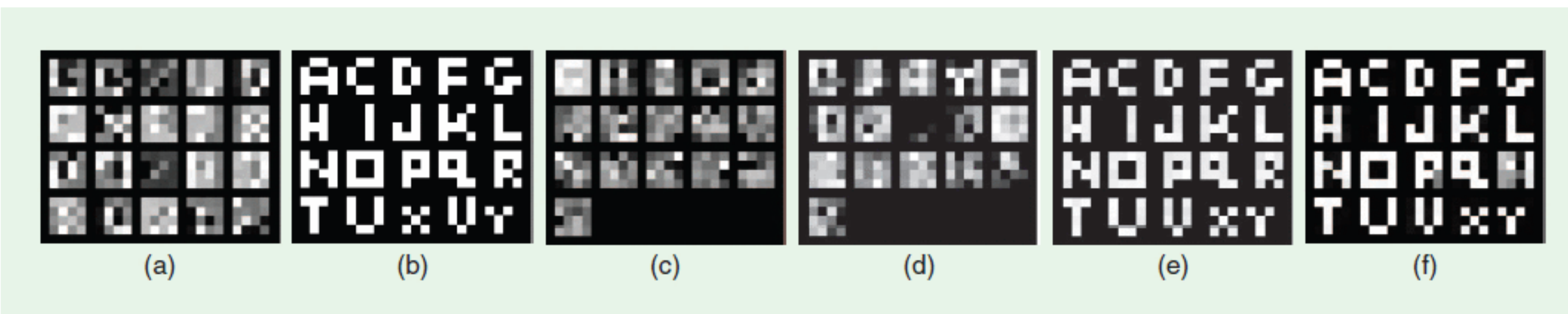
JPEG 2000 (10% in size)

JPEG 2000 (1% in size)

goal: compress signal without sacrificing quality

Example: Recognition

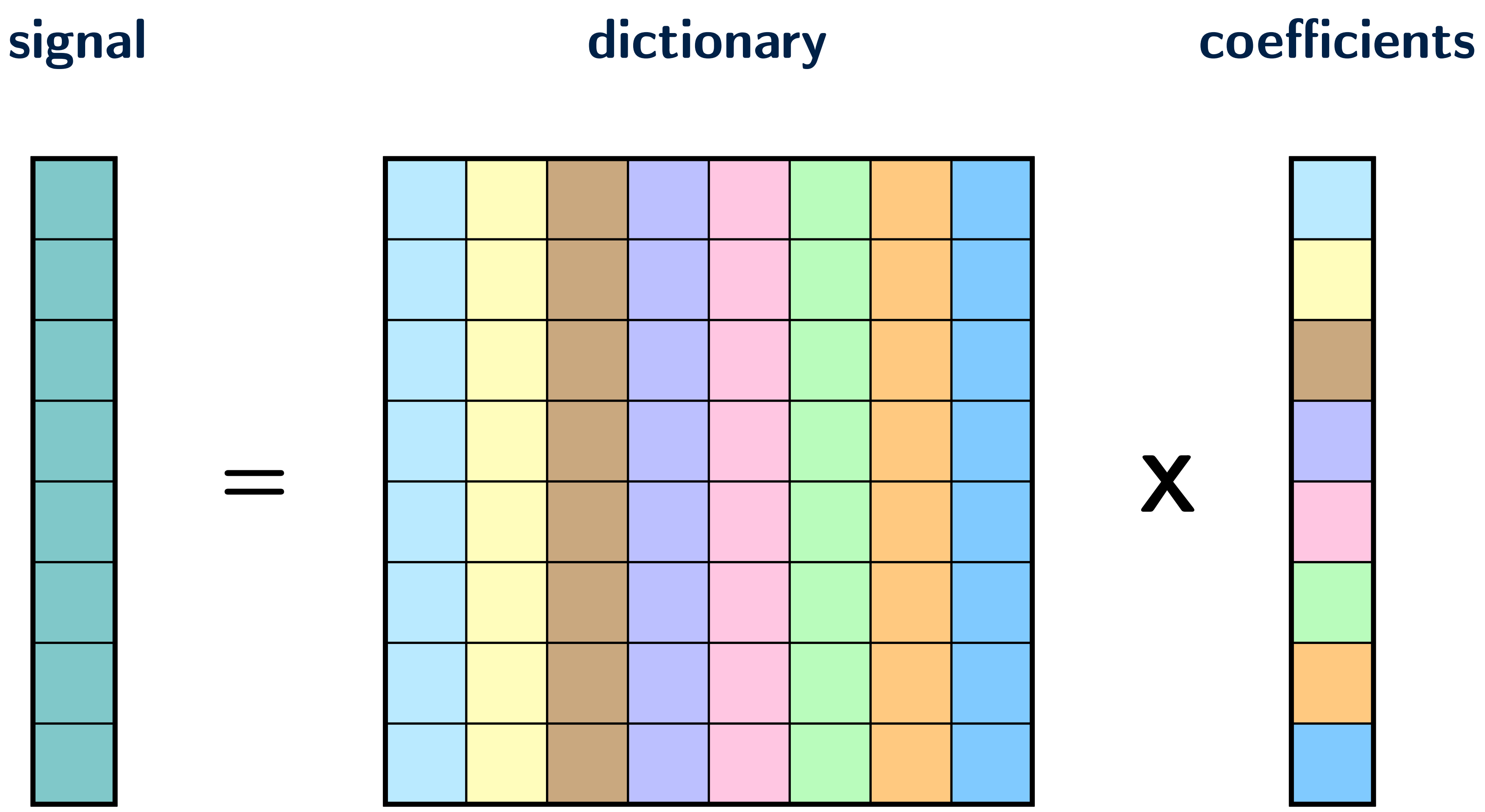
samples | **true "causes"** | **PCA** | **ICA** | **sparse coding** | **KSVD**



goal: capture true "causes" of signal

Tošić and Frossard, "Dictionary learning," IEEE SPM, 2011.

Signal representation via dictionaries

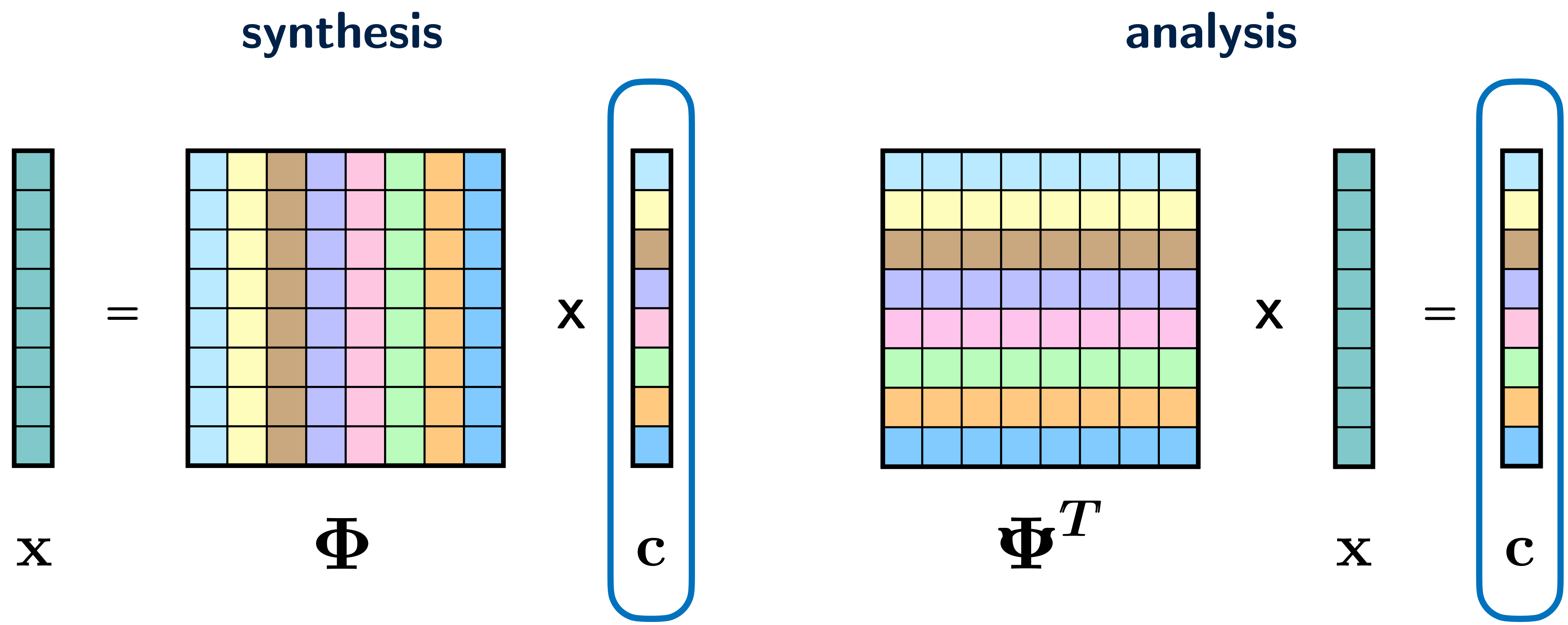


$\mathbf{x}$ $\quad$ $\boldsymbol{\Phi} = [\boldsymbol{\phi}_0 \ \boldsymbol{\phi}_1 \ \cdots \ \boldsymbol{\phi}_{N-1}]$ $\quad$ $\mathbf{c}$

"atoms"

Signal representation via dictionaries

- Complete dictionaries



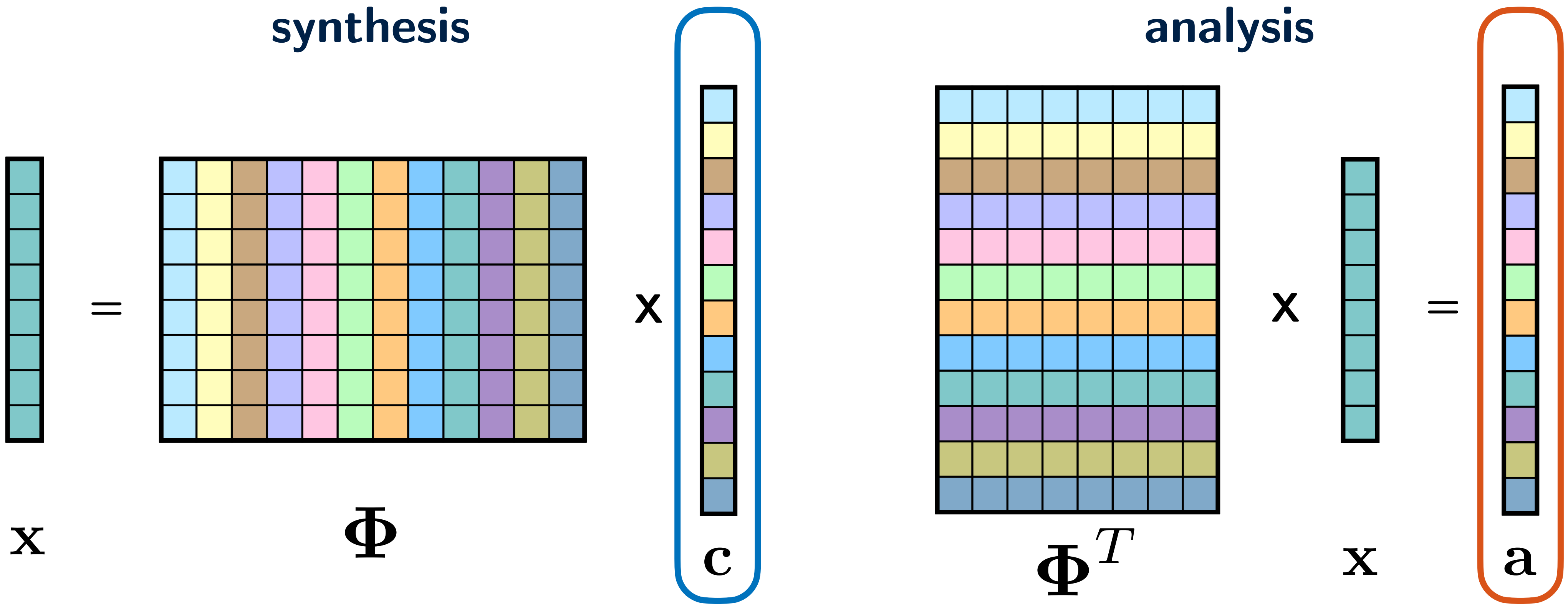
$$\mathbf{x} = \mathbf{\Phi}\mathbf{c} = \sum_n \mathbf{c}_n \phi_n$$

$$\mathbf{c}_n = \boldsymbol{\phi}_n^T \mathbf{x}$$

equivalent for complete dictionaries

Signal representation via dictionaries

- Over-complete dictionaries

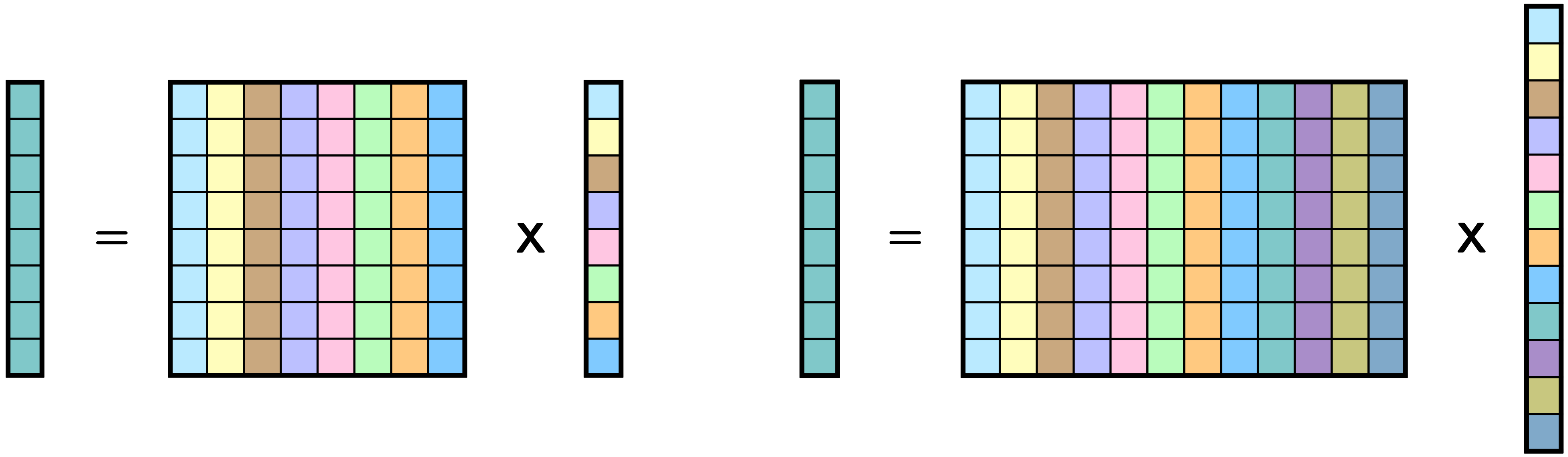


$$\mathbf{x} = \mathbf{\Phi}\mathbf{c} = \sum_n \mathbf{c}_n \phi_n$$

$$\mathbf{a}_n = \phi_n^T \mathbf{x}$$

not equivalent for over-complete dictionaries

Signal representation via dictionaries



two sources of dictionary design

- mathematical modelling of data (transforms/analytic dictionaries)
- a set of realisations of data (dictionary learning)

Outline

- A historical overview of dictionary design techniques
  - signal representation via stochastic models
  - transforms & analytic dictionaries
  - trained dictionaries (dictionary learning)
- Discussion
  - applications
  - connection with deep learning

1920s-30s: Stochastic models

- Stochastic models
  - examples of parametric models
  - describe how data were generated
  - provide a special representations of signal from a time-series viewpoint
- Typical examples
  - autoregressive (AR) models
  - moving average (MA) models
  - autoregressive moving average (ARMA) models

Autocorrelation

- Autocovariance: covariance between signal and lagged version of itself

$$\sigma_{xx}(T) = \frac{1}{N-1}\sum_{t=1}^{N}(x_{t-T} - \mu_x)(x_t - \mu_x)$$

lagged version by T samples

signal

- Autocorrelation: normalised autocovariance

$$r_{xx}(T) = \frac{\sigma_{xx}(T)}{\sigma_{xx}(0)} \qquad \text{where} \quad \sigma_{xx}(0) = \sigma_x^2$$

- Both are **symmetric** or **even** functions

Autocorrelation

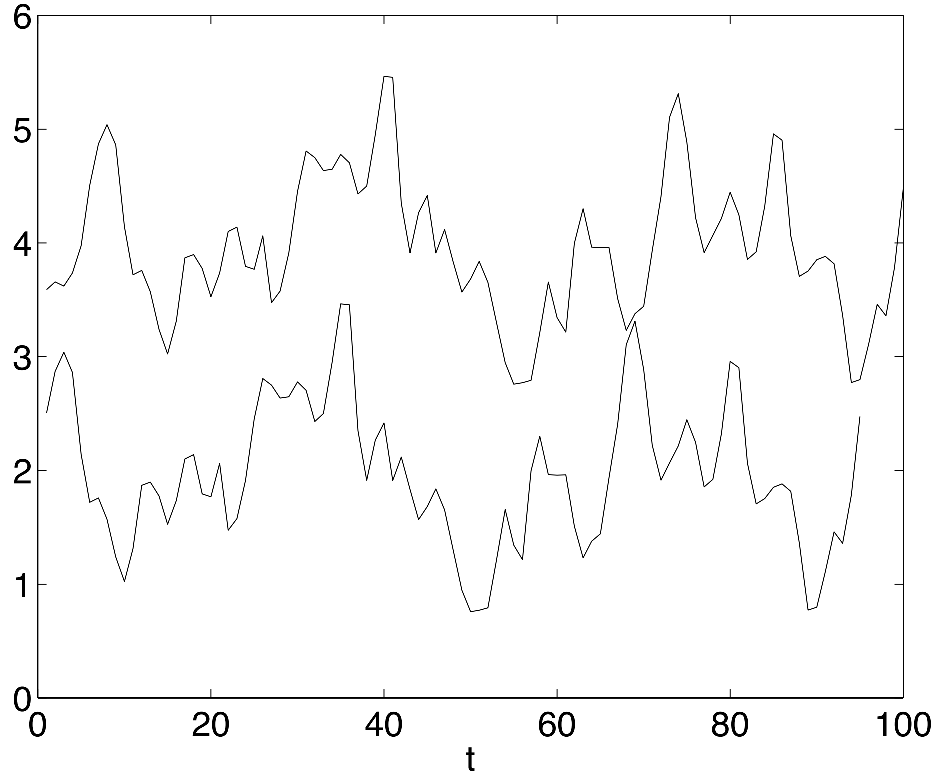


Figure 4.7: *Signal x_t (top) and x_{t+5} (bottom). The bottom trace* **leads** *the top trace by 5 samples. Or we may say it* **lags** *the top by -5 samples.*

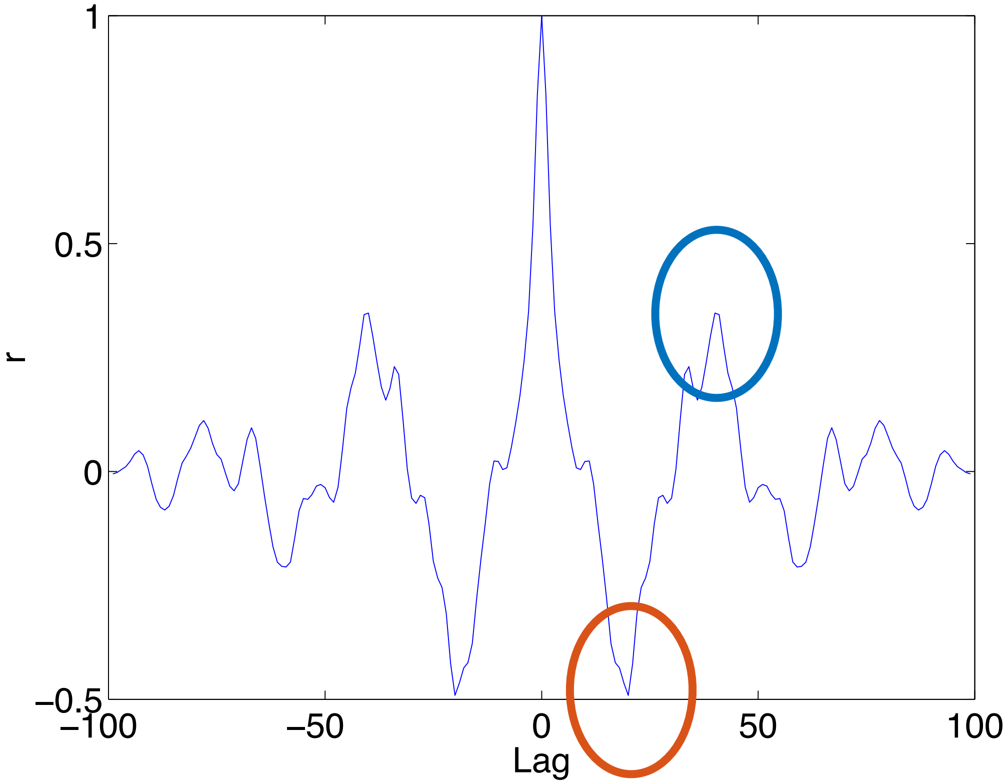


Figure 4.8: *Autocorrelation function for x_t. Notice the negative correlation at lag 20 and positive correlation at lag 40. Can you see from Figure 4.7 why these should occur?*

AR models

- An AR model **predicts** the value of a time-series from previous values

$$x_t = \sum_{i=1}^{p} x_{t-i} a_i + e_t$$

AR coefficients (a_i) **prediction error** (e_t) $e_t \sim \mathcal{N}(0, \sigma_e^2)$

- Matrix form

$$\mathbf{M} = \begin{bmatrix} x_4 & x_3 & x_2 & x_1 \\ x_5 & x_4 & x_3 & x_2 \\ .. & .. & .. & .. \\ x_{N-1} & x_{N-2} & x_{N-3} & x_{N-4} \end{bmatrix} \Rightarrow \begin{bmatrix} x_5 \\ x_6 \\ .. \\ x_N \end{bmatrix} = \begin{bmatrix} x_4 & x_3 & x_2 & x_1 \\ x_5 & x_4 & x_3 & x_2 \\ .. & .. & .. & .. \\ x_{N-1} & x_{N-2} & x_{N-3} & x_{N-4} \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix} + \begin{bmatrix} e_5 \\ e_6 \\ .. \\ e_N \end{bmatrix}$$

embedding matrix

$$\mathbf{x} = \mathbf{M}\mathbf{a} + \mathbf{e}$$

Estimation of AR: Least-squares method

- The AR model is a special case of the multivariate regression model
- It also provides a special **representation** of the signal
- To compute AR coefficients and predictions

$$\mathbf{x} = \mathbf{M}\mathbf{a} + \mathbf{e} \quad \Rightarrow \quad \begin{aligned} \hat{\mathbf{a}} &= (\mathbf{M}^T\mathbf{M})^{-1}\mathbf{M}^T\mathbf{x} \\ \hat{\mathbf{x}} &= \mathbf{M}\hat{\mathbf{a}} \\ \mathbf{e} &= \mathbf{x} - \hat{\mathbf{x}} \end{aligned}$$

Estimation of AR: Least-squares method

- Use an AR(4) model to analyse data shown before:

$$\hat{\mathbf{a}} = [1.46, -1.08, 0.60, -0.186]^T \qquad \sigma_e^2 = 0.079 \qquad \sigma_x^2 = 0.3882$$

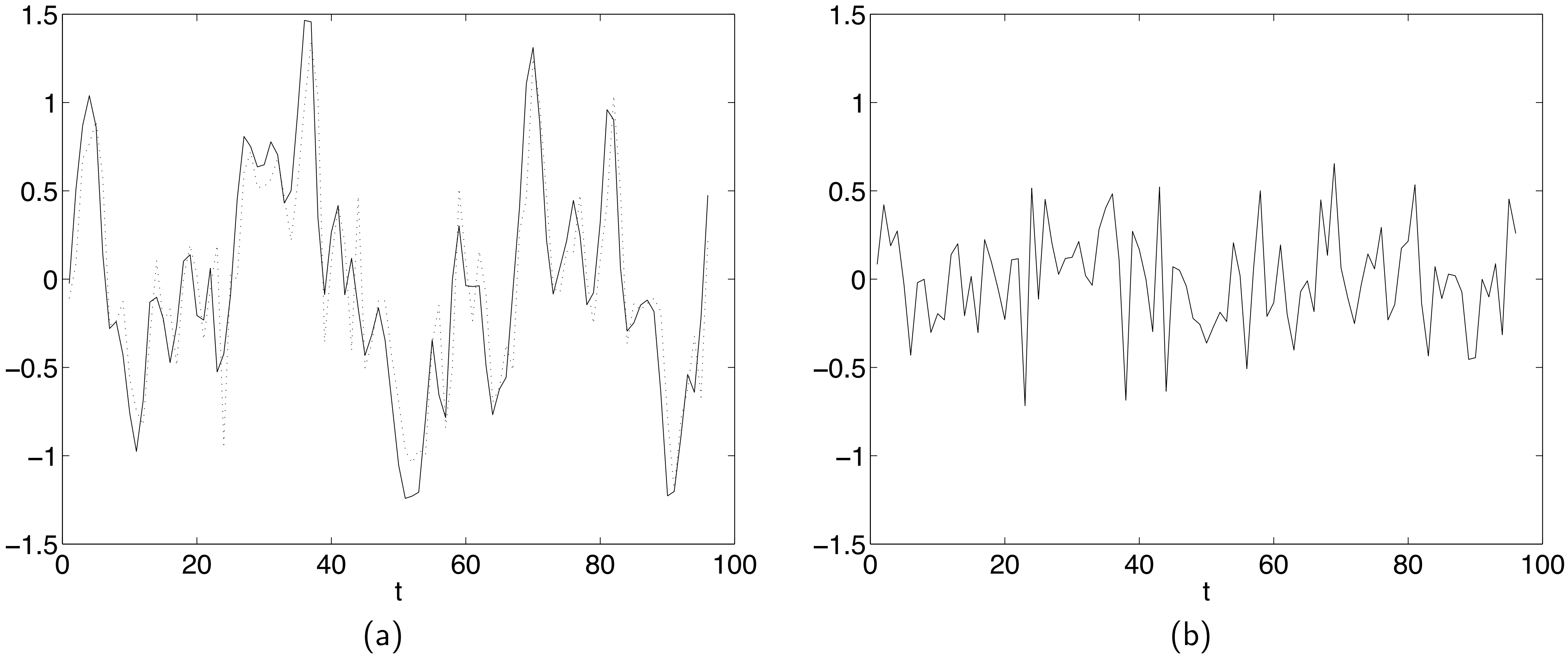


Figure 4.9: *(a) Original signal (solid line),* **X**, *and predictions (dotted line),* $\hat{\mathbf{X}}$, *from an AR(4) model and (b) the prediction errors,* **e**. *Notice that the variance of the errors is much less than that of the original signal.*

Estimation of AR: Yule-Walker method

- Relation to autocorrelation

$$x_t = a_1 x_{t-1} + a_2 x_{t-2} + ... + a_p x_{t-p} + e_t$$

⬇ multiply by x_{t-k}

$$x_t x_{t-k} = a_1 x_{t-1} x_{t-k} + a_2 x_{t-2} x_{t-k} + ... + a_p x_{t-p} x_{t-k} + e_t x_{t-k}$$

⬇ sum over t and divide by N-1

$$\sigma_{xx}(k) = a_1 \sigma_{xx}(k-1) + a_2 \sigma_{xx}(k-2) + ... + a_p \sigma_{xx}(k-p) + \sigma_{e,x}$$

⬇ divide by signal variance

$$r_{xx}(k) = a_1 r_{xx}(k-1) + a_2 r_{xx}(k-2) + ... + a_p r_{xx}(k-p)$$

Estimation of AR: Yule-Walker method

- For an AR(4) model

$$\begin{bmatrix} r_{xx}(1) \\ r_{xx}(2) \\ r_{xx}(3) \\ r_{xx}(4) \end{bmatrix} = \begin{bmatrix} r_{xx}(0) & r_{xx}(-1) & r_{xx}(-2) & r_{xx}(-3) \\ r_{xx}(1) & r_{xx}(0) & r_{xx}(-1) & r_{xx}(-2) \\ r_{xx}(2) & r_{xx}(1) & r_{xx}(0) & r_{xx}(-1) \\ r_{xx}(3) & r_{xx}(2) & r_{xx}(1) & r_{xx}(0) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ a_4 \end{bmatrix}$$

Yule-Walker relations

$$\mathbf{r} = \mathbf{R}\mathbf{a}$$

- More efficient way to estimate AR coefficients: $\mathbf{a} = \mathbf{R}^{-1}\mathbf{r}$

$$\mathbf{R} = \begin{bmatrix} 1 & r_{xx}(1) & r_{xx}(2) & r_{xx}(3) \\ r_{xx}(1) & 1 & r_{xx}(1) & r_{xx}(2) \\ r_{xx}(2) & r_{xx}(1) & 1 & r_{xx}(1) \\ r_{xx}(3) & r_{xx}(2) & r_{xx}(1) & 1 \end{bmatrix}$$

- autocorrelation matrix is symmetric and Toeplitz
- efficient computation via a recursive estimation technique (Levinson-Durbin)

Outline

- A historical overview of dictionary design techniques
  - signal representation via stochastic models
  - transforms & analytic dictionaries
  - trained dictionaries (dictionary learning)
- Discussion
  - applications
  - connection with deep learning

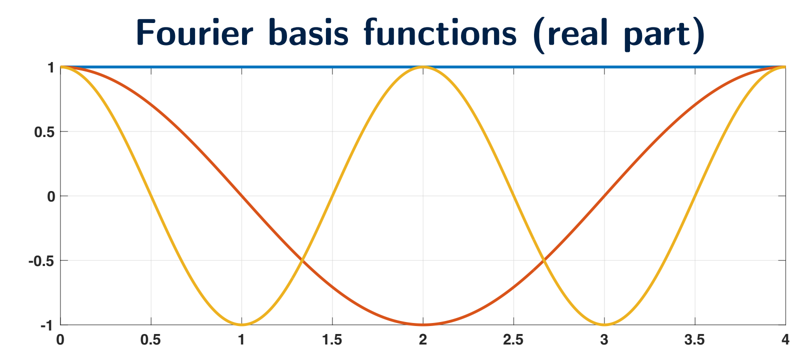
1960s: Fourier basis and DFT

discrete Fourier transform

1960s

- recall the LTI system

$$x(t) \longrightarrow \boxed{g(t)} \longrightarrow y(t)$$



$$y(t) = \int_{-\infty}^{\infty} e^{j\omega(t-\tau)} g(\tau) d\tau = e^{j\omega t} G(\omega)$$

Fourier basis diagonalises convolution operator

$$X(\omega) \longrightarrow \boxed{G(\omega)} \longrightarrow Y(\omega) = X(\omega)G(\omega)$$

1960s: Fourier basis and DFT

- Fourier basis describes a signal in terms of its **global** frequency content and hence is good at representing **uniformly smooth** signals
- discrete Fourier transform (DFT) provides an **orthogonal** dictionary: $\phi_n(k) = e^{j\frac{2\pi}{N}nk}$

$$\begin{pmatrix} x[0] \\ x[1] \\ x[2] \\ \vdots \\ x[N-1] \end{pmatrix} = \frac{1}{N} \begin{pmatrix} 1 & 1 & 1 & 1 & \dots & 1 \\ 1 & W & W^2 & W^3 & \dots & W^{N-1} \\ 1 & W^2 & W^4 & W^6 & \dots & W^{N-2} \\ 1 & W^3 & W^6 & W^9 & \dots & W^{N-3} \\ \vdots & & & & & \\ 1 & W^{N-1} & W^{N-2} & W^{N-3} & \dots & W \end{pmatrix} \begin{pmatrix} X[0] \\ X[1] \\ X[2] \\ \vdots \\ X[N-1] \end{pmatrix} \quad \text{with} \quad W = e^{j\frac{2\pi}{N}}$$

- fast Fourier transform (FFT) reduces complexity from $\mathcal{O}(N^2)$ to $\mathcal{O}(N\log N)$

1960s: Fourier basis and DFT

- DFT produces complex coefficients ("wasteful" for real signals)
- DFT assumes periodic extension (discontinuity at boundary)

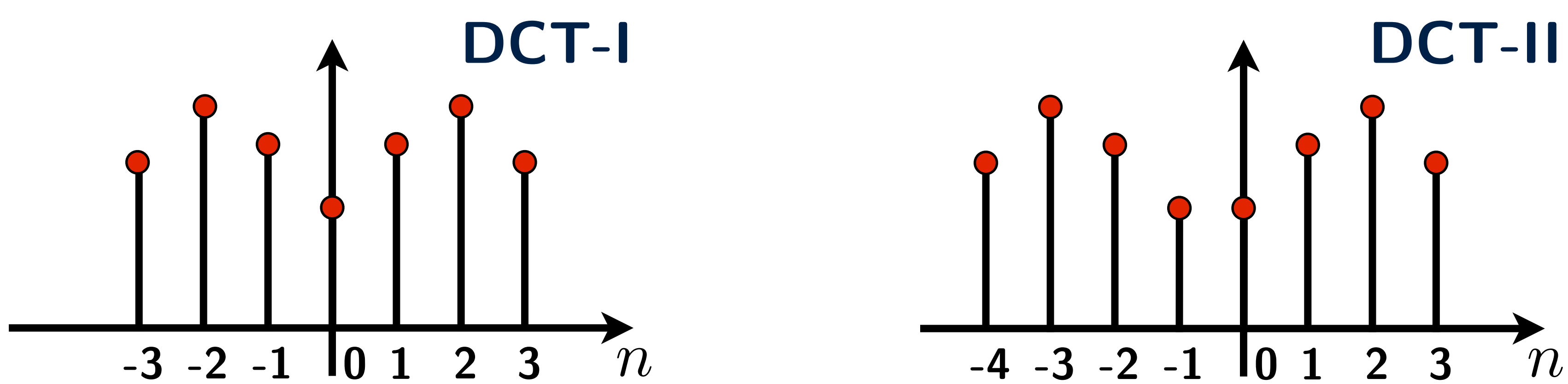
Fourier transform $X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-j\omega t}dt = \int_{-\infty}^{\infty} x(t)[\cos(\omega t) - j\sin(\omega t)]dt$

$\xrightarrow{x(t) = x(-t)}$ $X(\omega) = \int_{-\infty}^{\infty} x(t)\cos(\omega t)dt$ **cosine transform**

- a real and even signal leads to a **real** cosine transform

1970s: DCT

- the discrete version is called the discrete cosine transform (DCT)
- several variants of symmetric extension, which all make the signal **even** and lead to **smoother boundary**



- DCT-II provides a real dictionary: $\phi_n(k) = \cos[\frac{\pi}{N}(n+\frac{1}{2})k]$
- DCT-II is behind the JEPG image compression standard

1970s-80s: KLT and PCA

- projection onto a fixed subset of DFT or DCT atoms leads to **compaction**

$$\mathbf{x} \approx \sum_{n \in \mathcal{S}_k} (\mathbf{\Psi}_n^T \mathbf{x}) \mathbf{\Phi}_n$$

- but **data** themselves can also be a source of compaction
- Karhunen-Loève transform (KLT) or principal component analysis (PCA) fits a low-dimensional subspace to data

$$\mathbf{\Sigma} = \mathbf{\Phi} \mathbf{\Lambda} \mathbf{\Phi}^T$$

known/empirical covariance

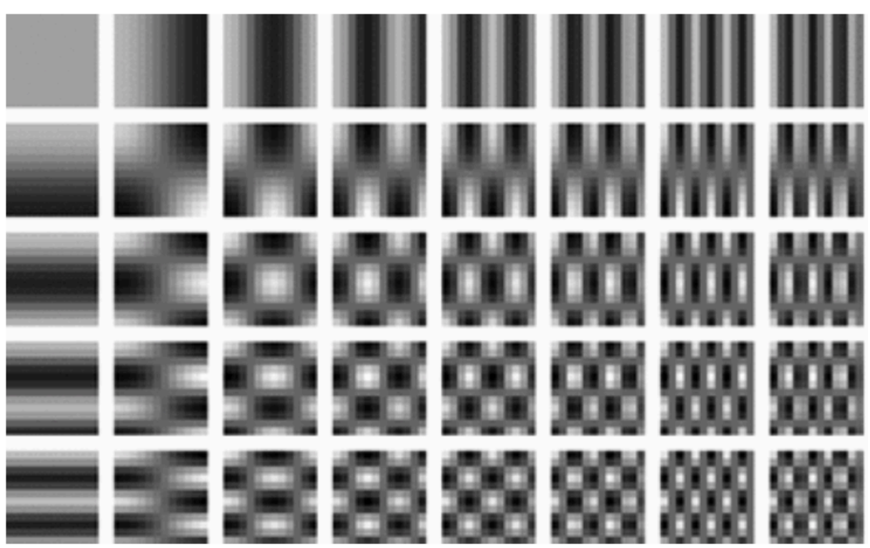
eigenvectors (dictionary atoms as k largest eigenvectors)

- representation is efficient (maximally compacts energy) but expensive to compute

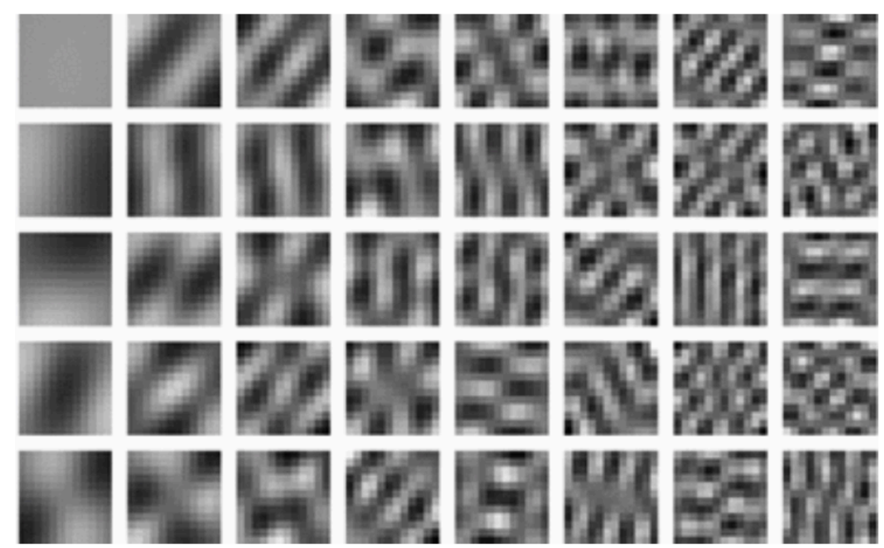
DCT vs KLT

- DCT atoms (12x12) vs. KLT atoms (trained using 12x12 image patches)

DCT



KLT



Rubinstein et al., "Dictionaries for sparse representation modeling," Proceedings of the IEEE, 2010.

The need for sparsity

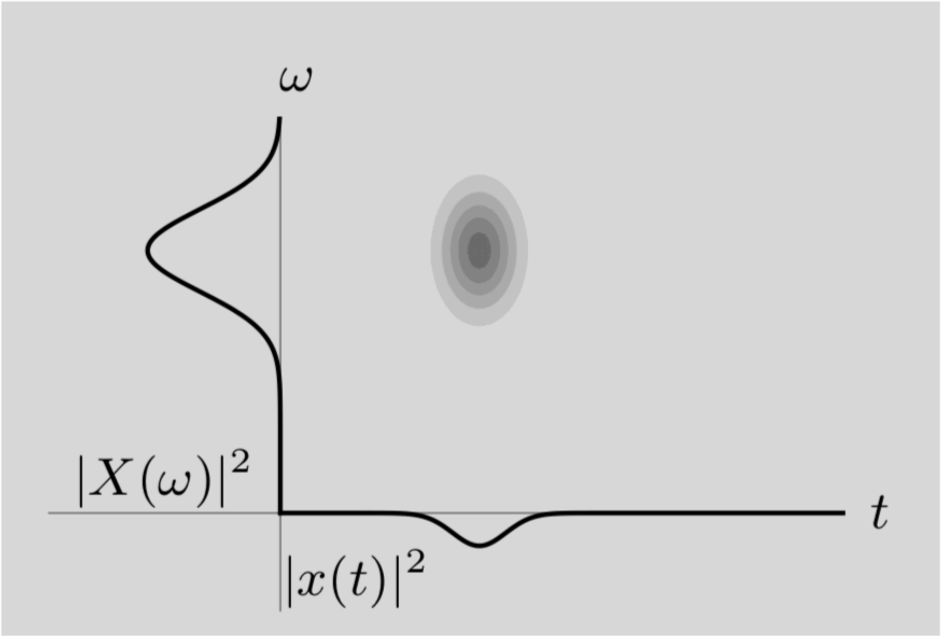
- simplicity motivates **sparsity**: signal as linear combination of a few atoms
- sparsity requires shift from linear to **nonlinear** approximation

$$\mathbf{x} \approx \sum_{\mathcal{K}} c_k \phi_k$$

subset of atoms (different for each x)

- sparsity requires **localisation:** atoms with concentrated supports
  - allow more flexible representations based on local characteristics
  - limit effects of irregularities (a main source of large coefficients)

Time-frequency representation



time-frequency tile (Heisenberg box)

time-frequency plane

time localisation

$$\mu_t = \frac{1}{||x||^2}\int_{-\infty}^{\infty} t|x(t)|^2 dt$$

$$\Delta_t = \left(\frac{1}{||x||^2}\int_{-\infty}^{\infty}(t-\mu_t)^2|x(t)|^2 dt\right)^{\frac{1}{2}}$$

frequency localisation

$$\mu_f = \frac{1}{2\pi||x||^2}\int_{-\infty}^{\infty} \omega|X(\omega)|^2 d\omega$$

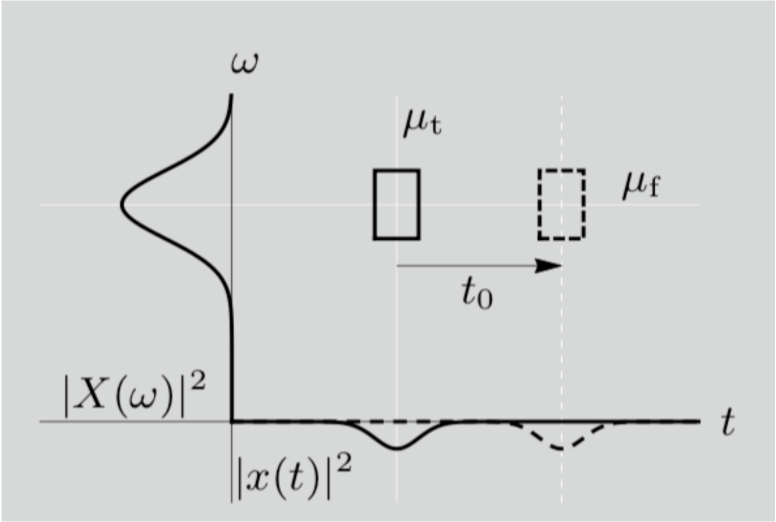
$$\Delta_f = \left(\frac{1}{2\pi||x||^2}\int_{-\infty}^{\infty}(\omega-\mu_f)^2|X(\omega)|^2 d\omega\right)^{\frac{1}{2}}$$

Vetterli et al., "Foundations of signal processing," Cambridge University Press, 2014.

Time-frequency representation

- Consider three basic operations

shift in time

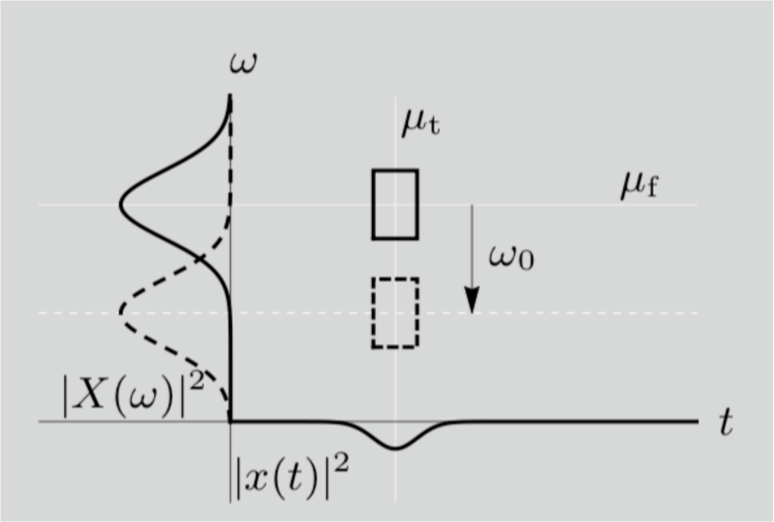


$$y(t) = x(t - t_0)$$

FT ⬇

$$Y(\omega) = e^{-j\omega t_0} X(\omega)$$

shift in frequency

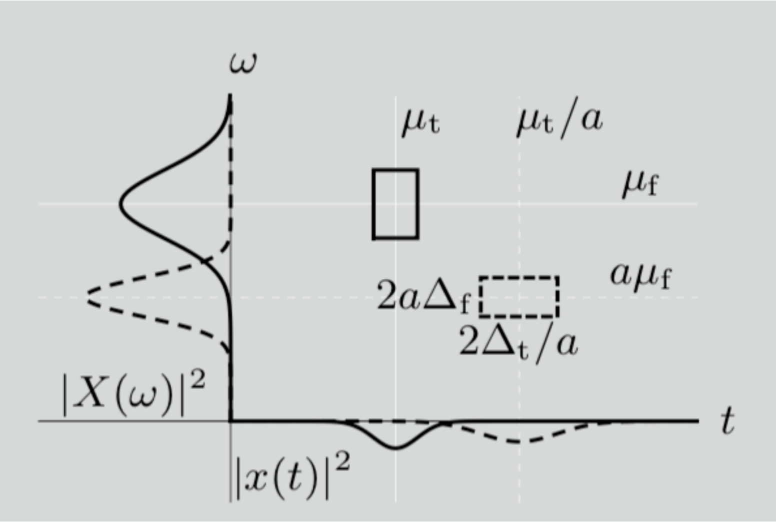


$$y(t) = e^{j\omega_0 t} x(t)$$

FT ⬇

$$Y(\omega) = X(\omega - \omega_0)$$

scaling in time



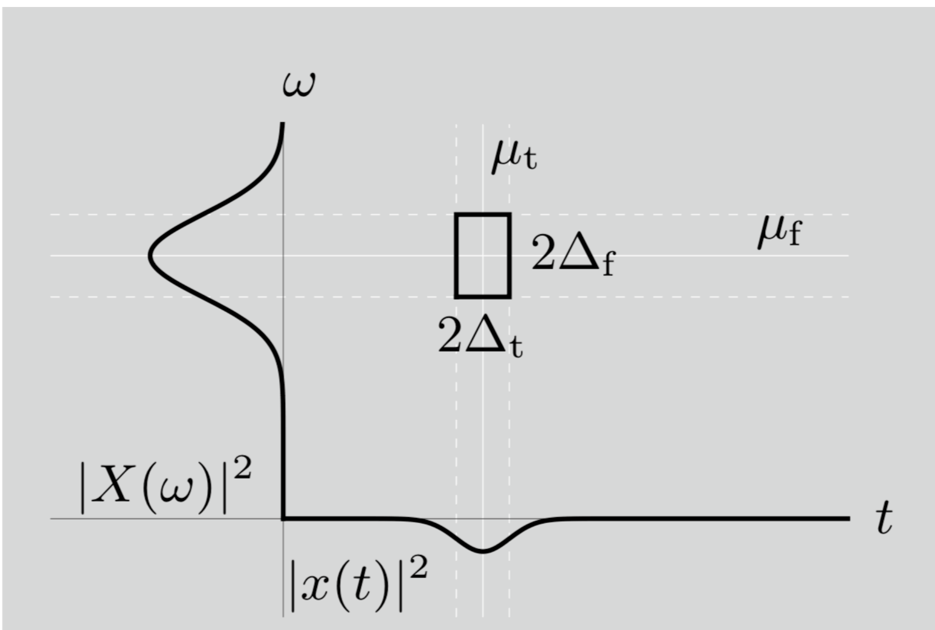
$$y(t) = \sqrt{a}x(at)$$

FT ⬇

$$Y(\omega) = \frac{1}{\sqrt{a}} X(\frac{\omega}{a})$$

Vetterli et al., "Foundations of signal processing," Cambridge University Press, 2014.

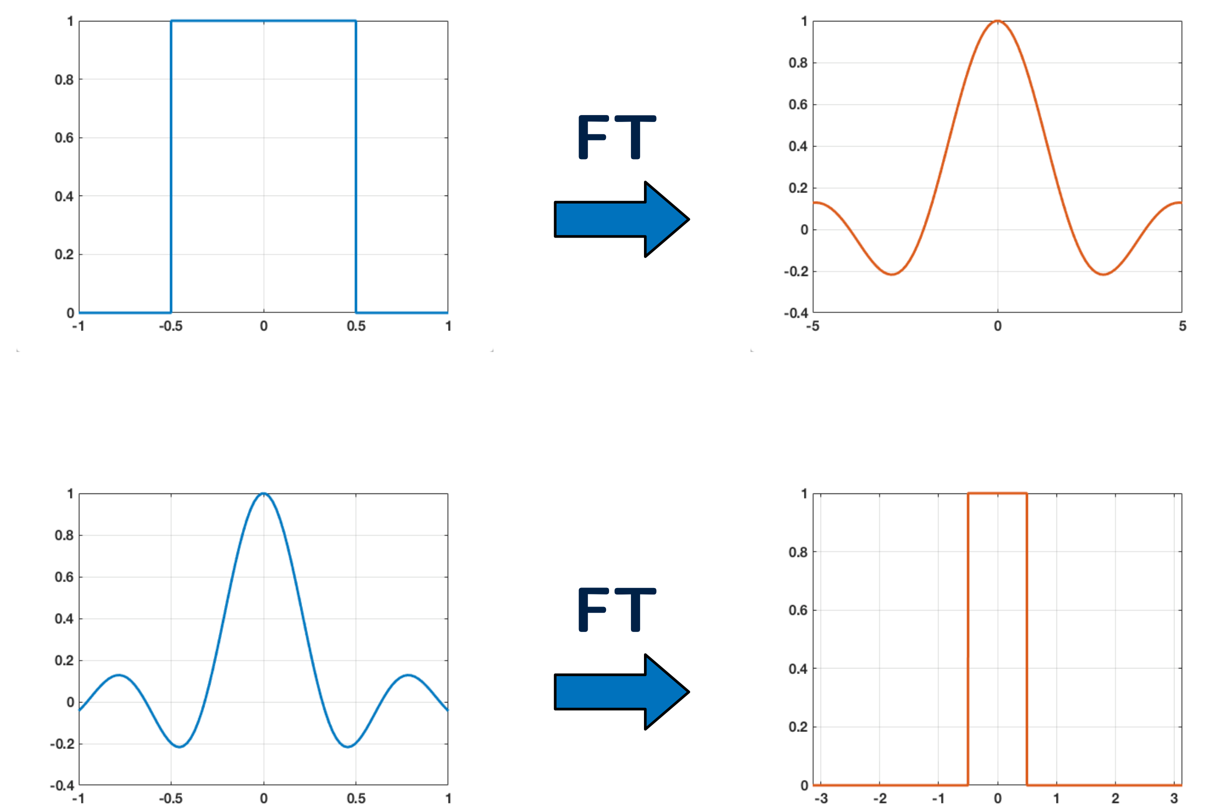
Time-frequency representation



Heisenberg's uncertainty principle

$$\text{Let } x \in \mathcal{L}^2(\mathbb{R}), \text{ then } \Delta_t \Delta_f \geq \frac{1}{2}$$

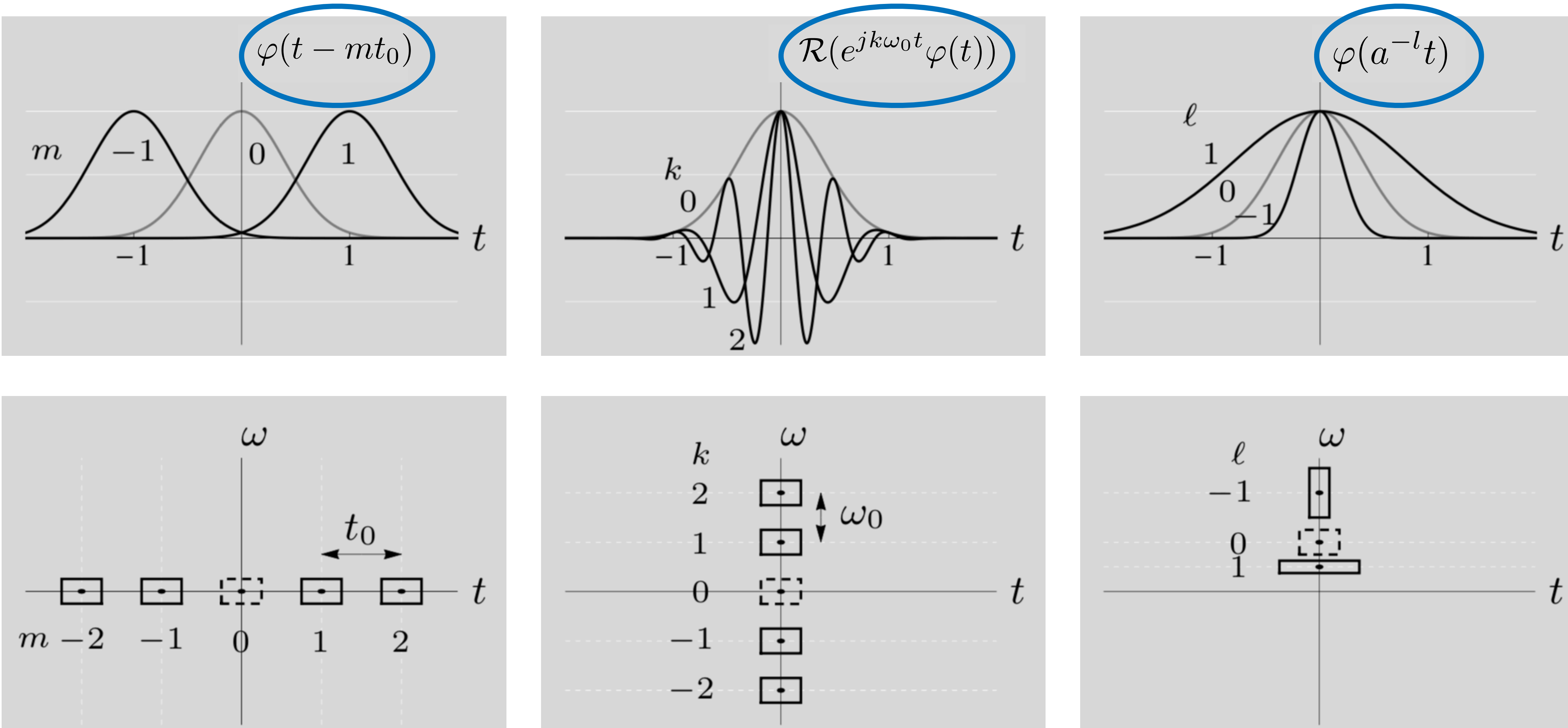
examples



Time-frequency representation

- Consider three structured sets of functions

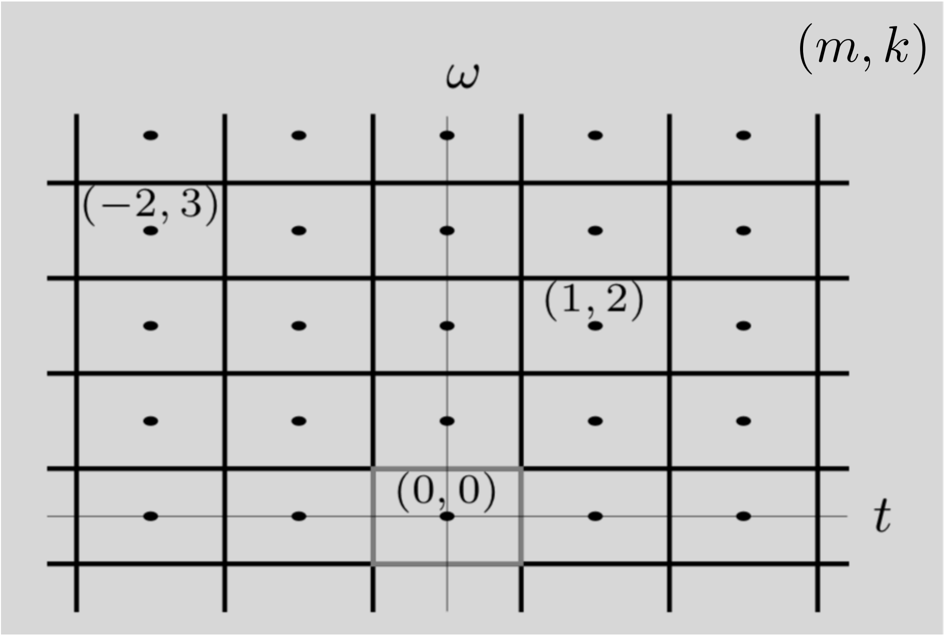
shift in time | **shift in frequency** | **scaling in time**



Vetterli et al., "Foundations of signal processing," Cambridge University Press, 2014.

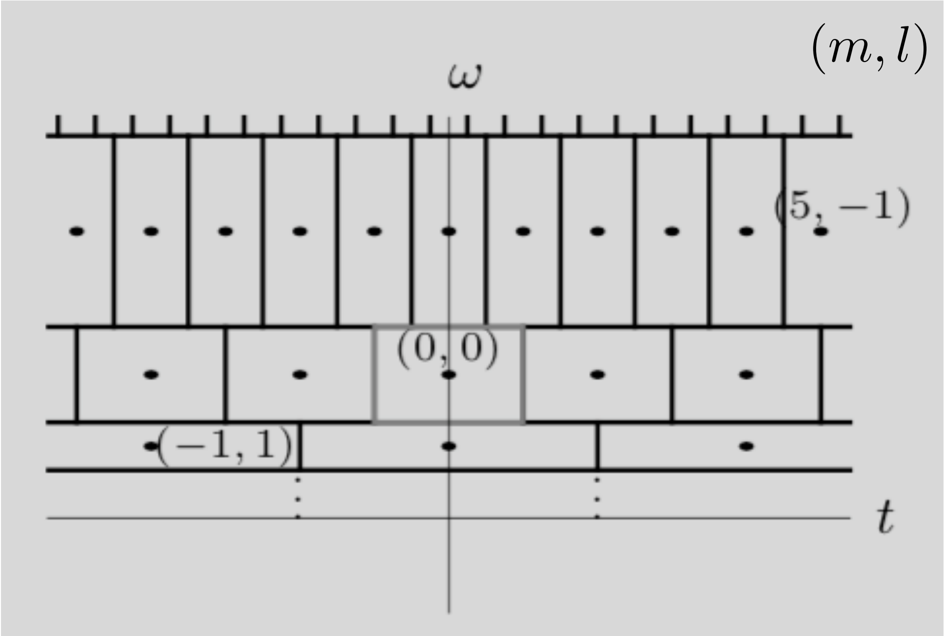
Time-frequency representation

time shift and modulation



$$\varphi_{k,m}(t) = e^{jk\omega_0 t}\varphi(t - mt_0) \quad k, m \in \mathbb{Z}$$

time shift and scaling



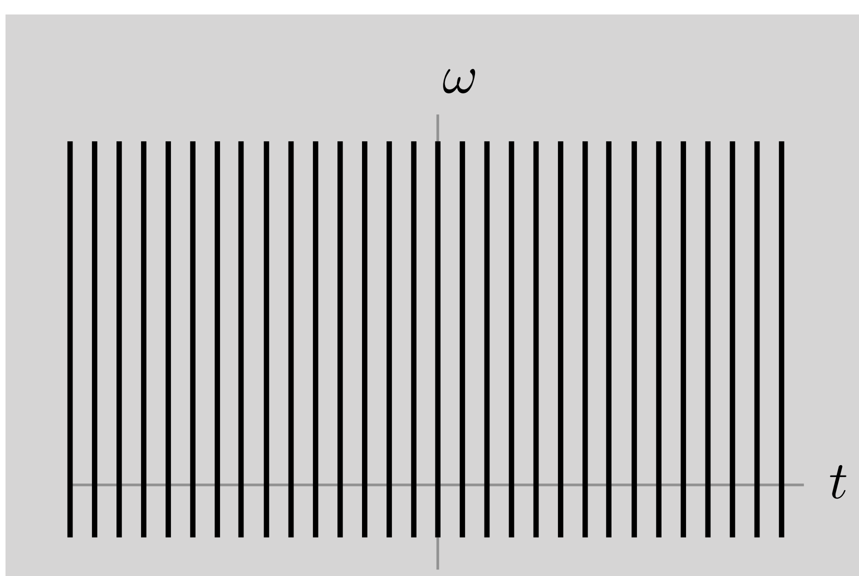
$$\varphi_{l,m}(t) = \varphi(a^{-l}t - mt_0) \quad l, m \in \mathbb{Z}$$

$$\varphi(a^{-l}(t - ma^l t_0))$$

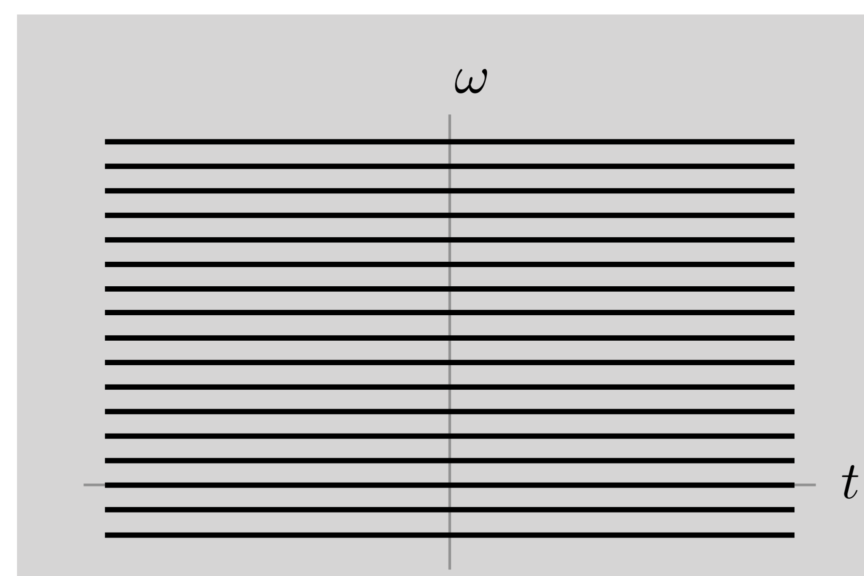
1970s-80s: STFT

- delta functions are not localised in frequency
- Fourier basis functions (complex exponentials) are not localised in time

time-domain



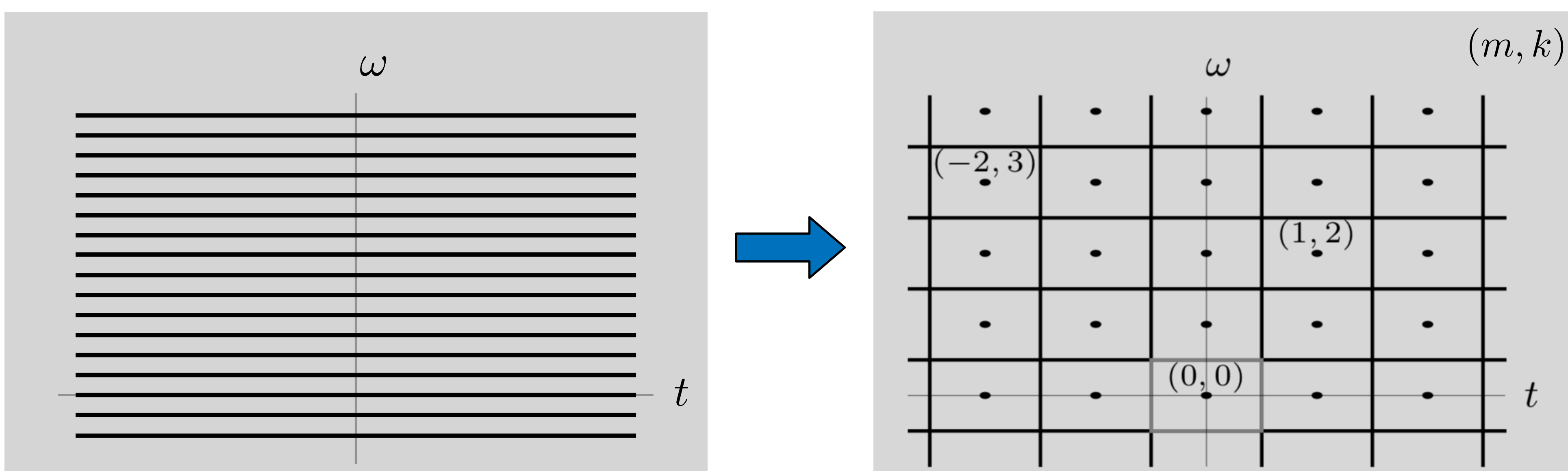
frequency-domain



1970s-80s: STFT

- consider a set of shifted and modulated versions of a low-pass function

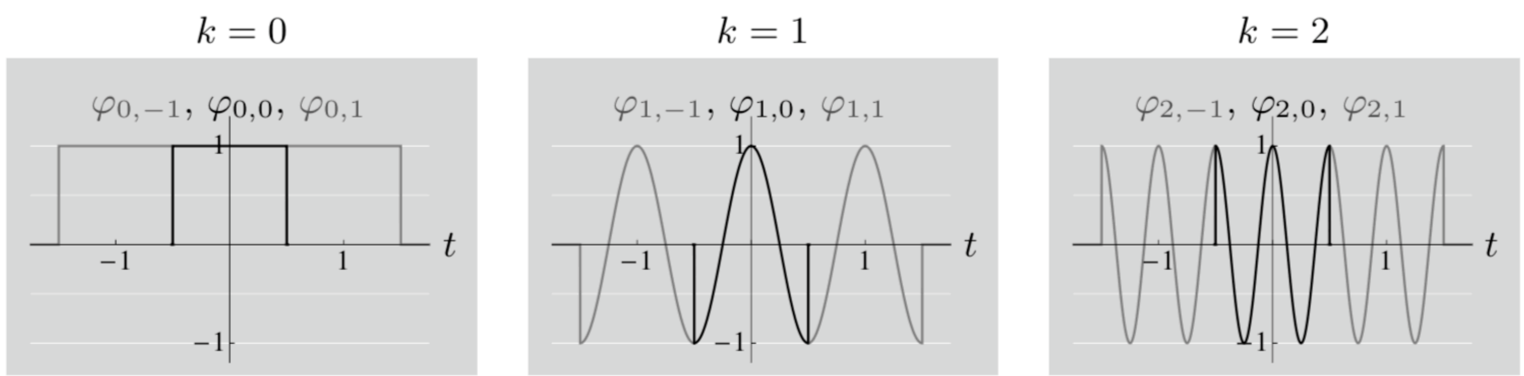
$$\varphi_{k,m}(t) = e^{jk\omega_0 t}\varphi(t - mt_0) \quad k, m \in \mathbb{Z}$$



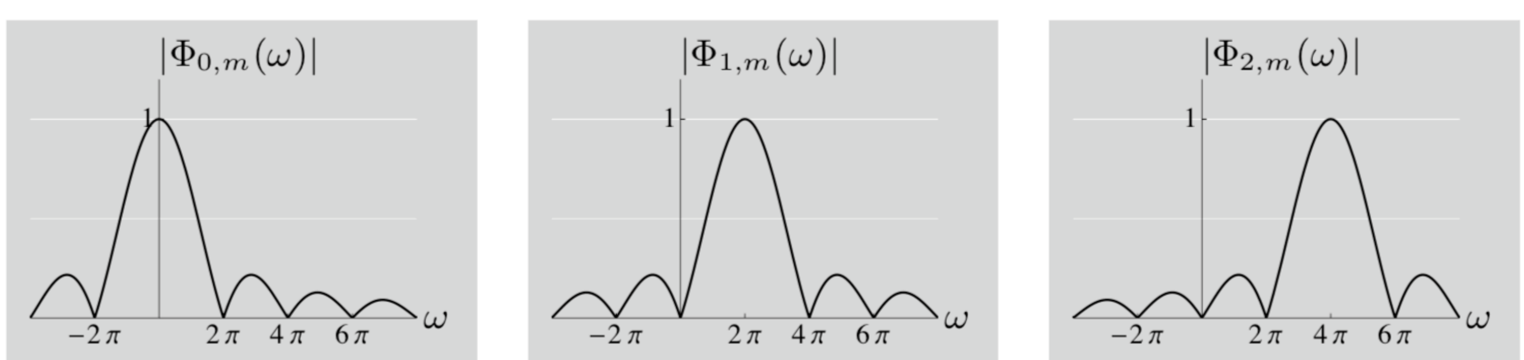
1970s-80s: STFT

- example: consider a box function and $t_0 = 1$, $\omega_0 = 2\pi$

$$\varphi_{k,m}(t) = e^{jk2\pi t}\varphi(t-m), \quad \varphi(t) = \begin{cases} 1, & \text{for } |t| \le \frac{1}{2}; \\ 0, & \text{otherwise.} \end{cases}$$



Basis functions (real parts only).



Magnitudes of the Fourier transform.

1970s-80s: STFT

- we can define the following transform

$$X_{k,m} = \int_{-\infty}^{\infty} x(t)\varphi_{k,m}^{*}(t)dt = \int_{-\infty}^{\infty} x(t)\varphi(t - mt_0)e^{-jk\omega_0 t}dt$$

$$X(\omega, \tau) = \int_{-\infty}^{\infty} x(t)\varphi_{\omega,\tau}^{*}(t)dt = \int_{-\infty}^{\infty} x(t)\varphi(t - \tau)e^{-j\omega t}dt$$

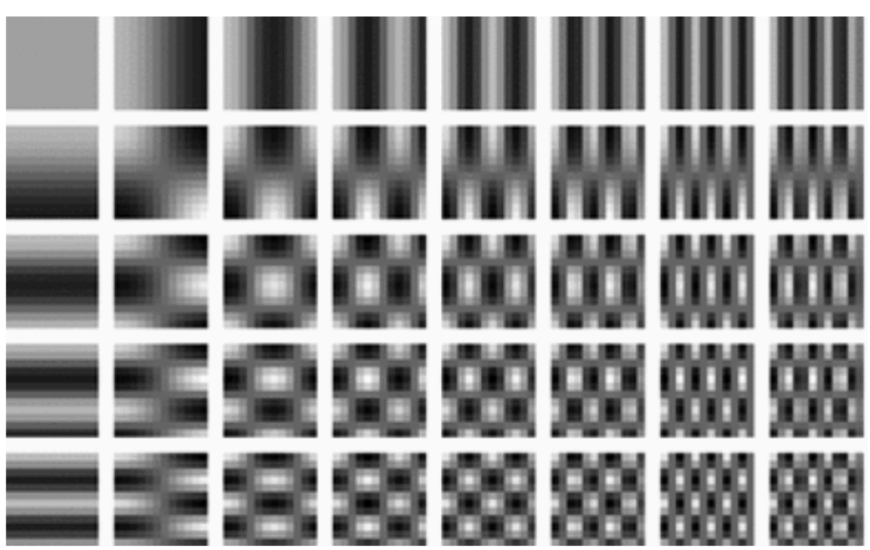
- applying **time-localised** window to the signal before taking Fourier transform: windowed or short-time Fourier transform (STFT)
- Gaussian window achieves **localisation in frequency**: Gabor transform
- STFT maps a 1-D function into a 2-D function (**redundant**)

DCT vs STFT

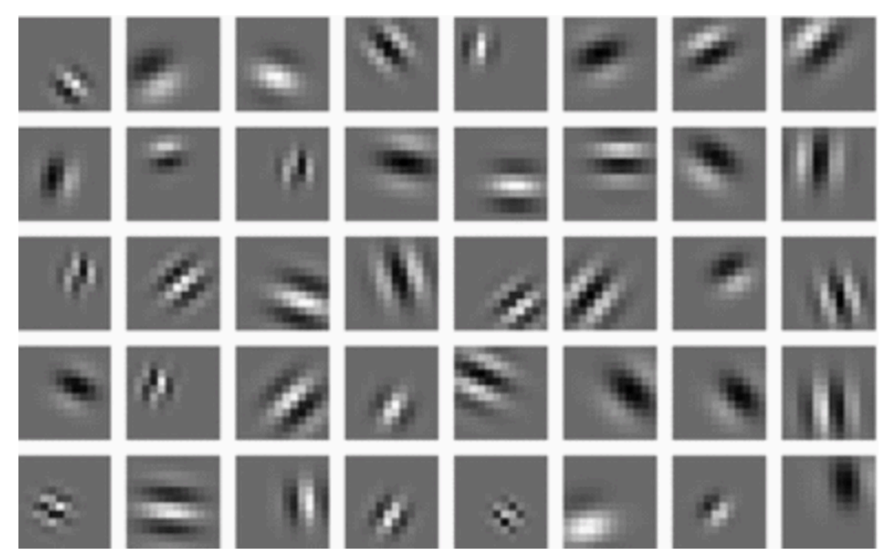
- discrete STFT provides an **over-complete** dictionary

$$\phi_{k,m}(n) = e^{j\frac{2\pi}{N}nk}\varphi(n - mN)$$

DCT

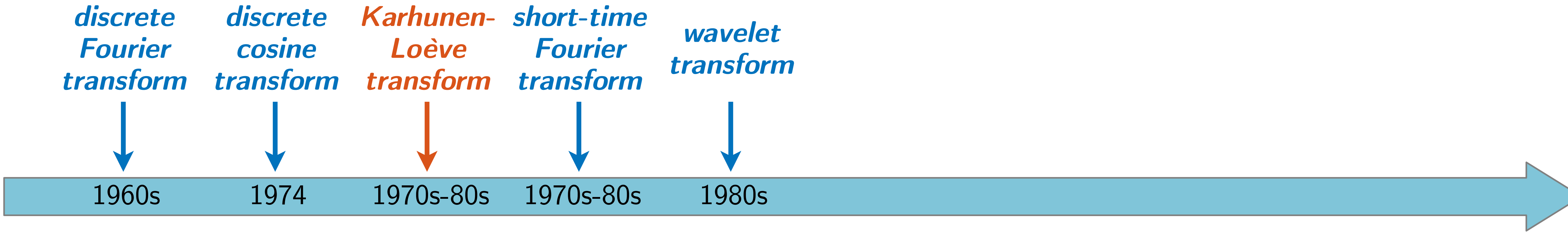


STFT

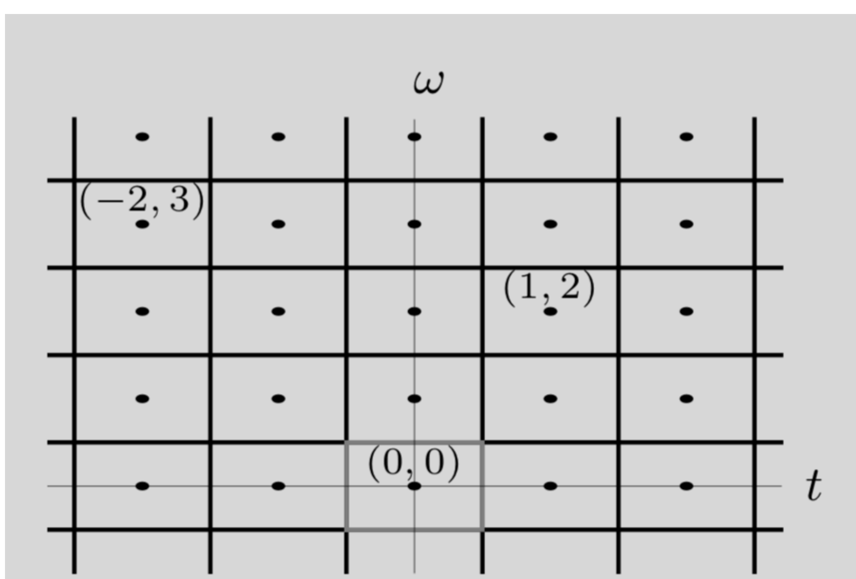


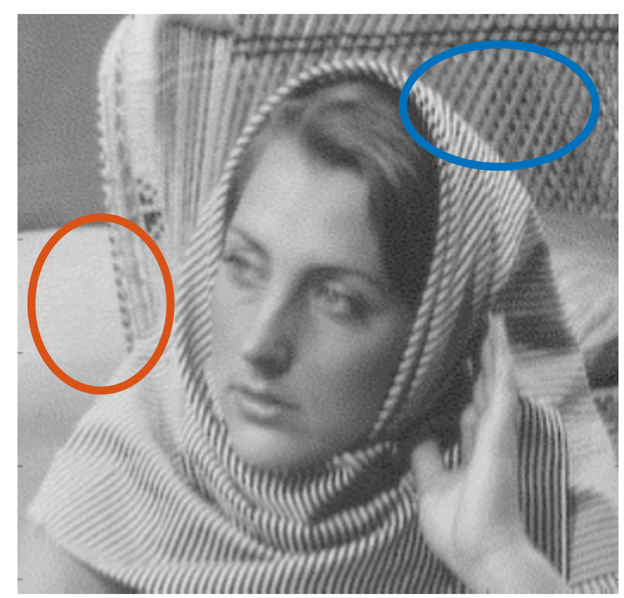
Rubinstein et al., "Dictionaries for sparse representation modeling," Proceedings of the IEEE, 2010.

1980s: wavelet transform

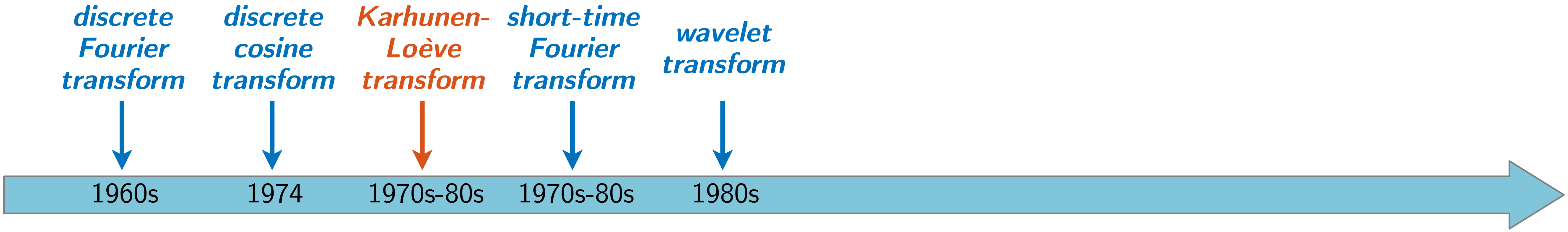


- STFT atoms have **fixed time-frequency resolution**
- often times a **multi-resolution** representation is needed to capture various scales in natural signals



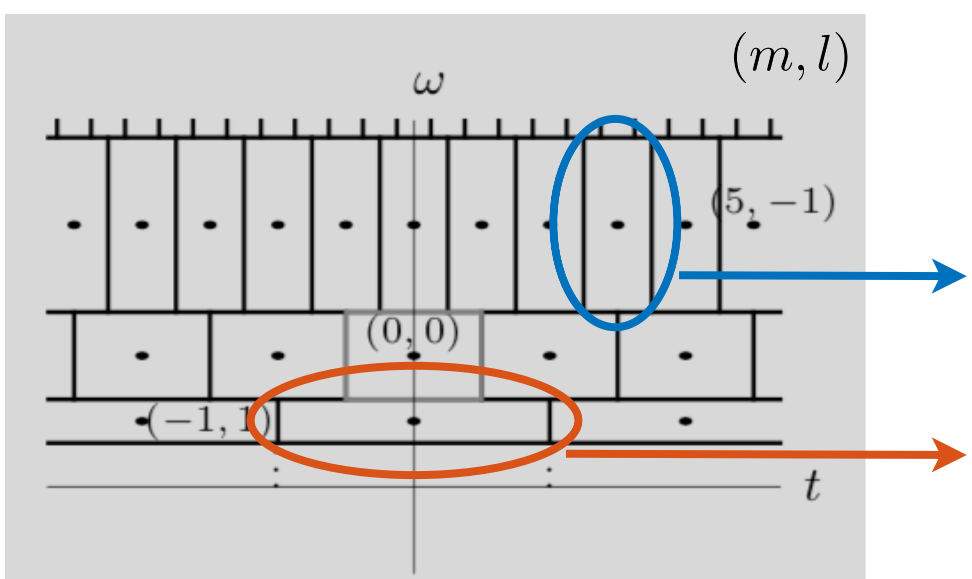


1980s: wavelet transform



- consider a set of shifted and scaled versions of a band-pass function

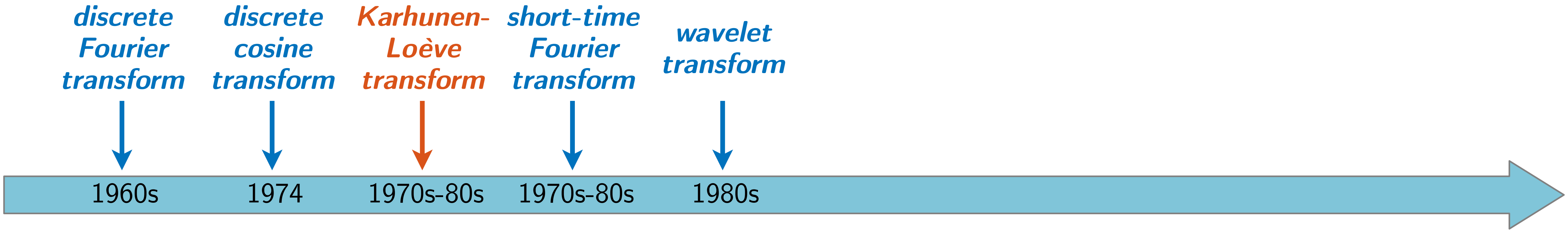
$$\varphi_{l,m}(t) = \varphi(a^{-l}t - mt_0) = \varphi(\frac{t - ma^l t_0}{a^l}) \quad l, m \in \mathbb{Z}$$



good time resolution (short-term)

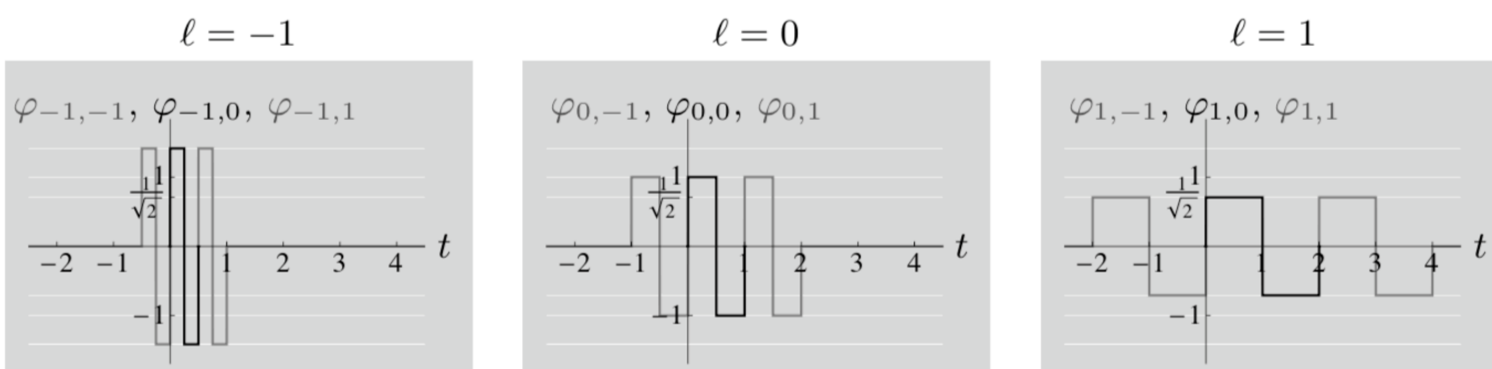
good frequency resolution (long-term)

1980s: wavelet transform

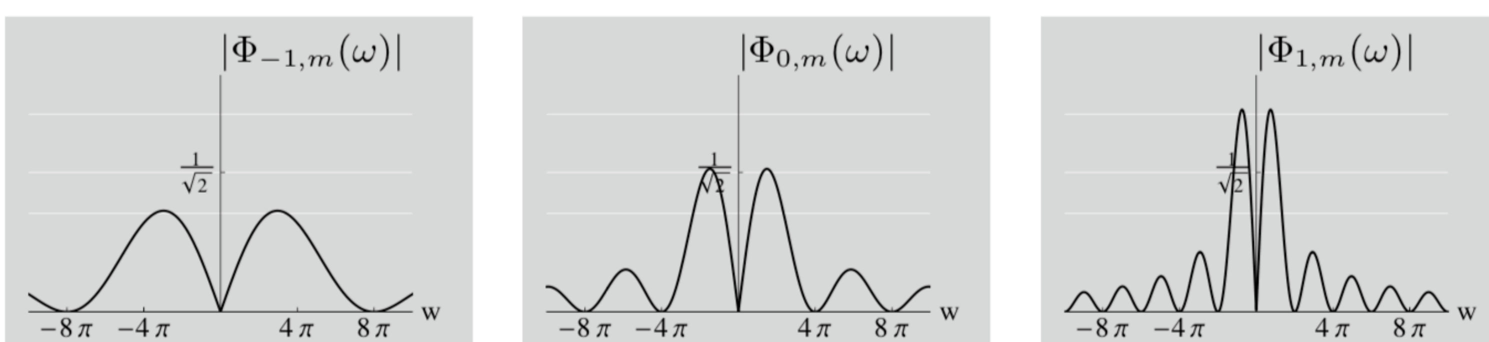


- example: consider a square wave function and $t_0 = 1$, $a = 2$

$$\varphi_{l,m}(t) = \varphi\left(\frac{t - 2^l m}{2^l}\right), \quad \varphi(t) = \begin{cases} 1, & \text{for } 0 \le t < \frac{1}{2}; \\ -1, & \text{for } \frac{1}{2} \le t < 1; \\ 0, & \text{otherwise.} \end{cases}$$



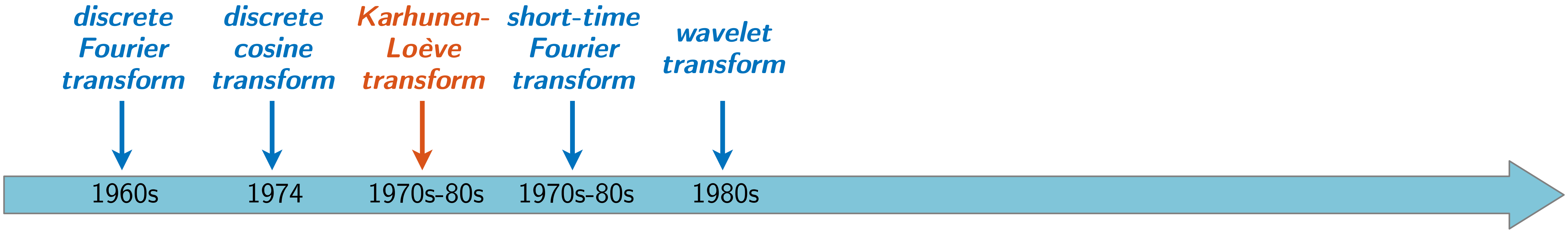
Basis functions.



Magnitudes of the Fourier transform.

Vetterli et al., "Foundations of signal processing," Cambridge University Press, 2014.

1980s: wavelet transform

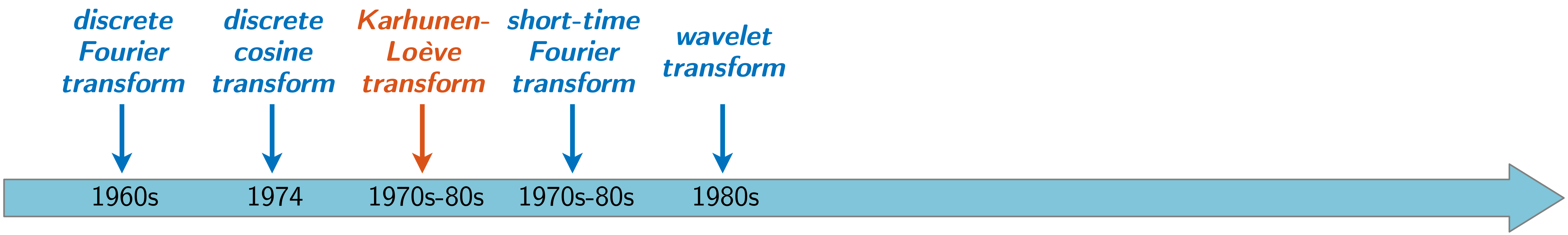


- consider a more general function and define the following transform

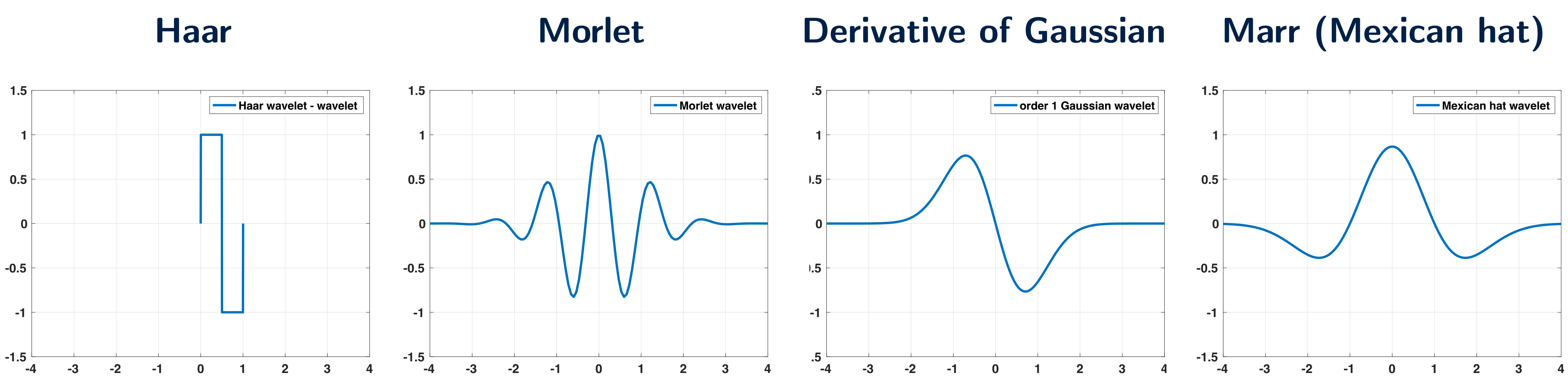
$$\psi_{s,\tau}(t) = \frac{1}{\sqrt{s}}\psi\left(\frac{t-\tau}{s}\right) \Rightarrow X(s,\tau) = \int_{-\infty}^{\infty} x(t)\psi^*_{s,\tau}(t)dt = \int_{-\infty}^{\infty} x(t)\frac{1}{\sqrt{s}}\psi^*\left(\frac{t-\tau}{s}\right)dt$$

- the prototype function $\psi(t)$
  - has a compact support (small or "-let")
  - is band-pass with zero mean ("wave"): $\int_{-\infty}^{\infty} \psi(t)dt = 0$
- this is called the **continuous wavelet transform (CWT)**
- CWT maps a 1-D function into a 2-D function (**redundant**)

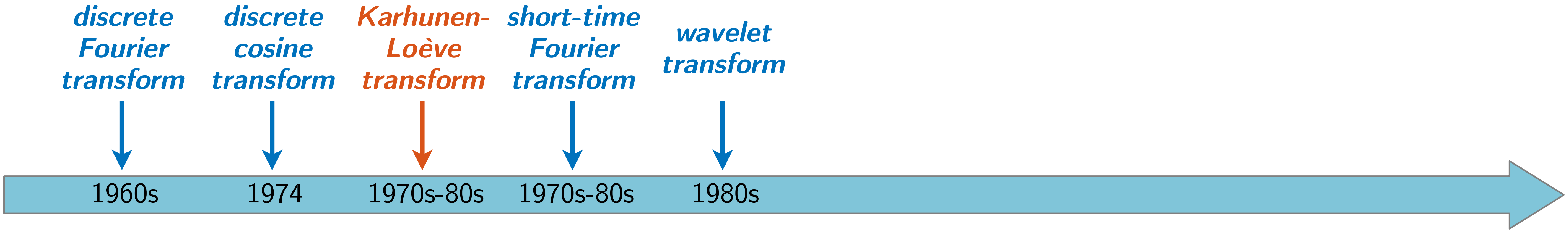
1980s: wavelet transform



- examples of prototype function (mother wavelet)

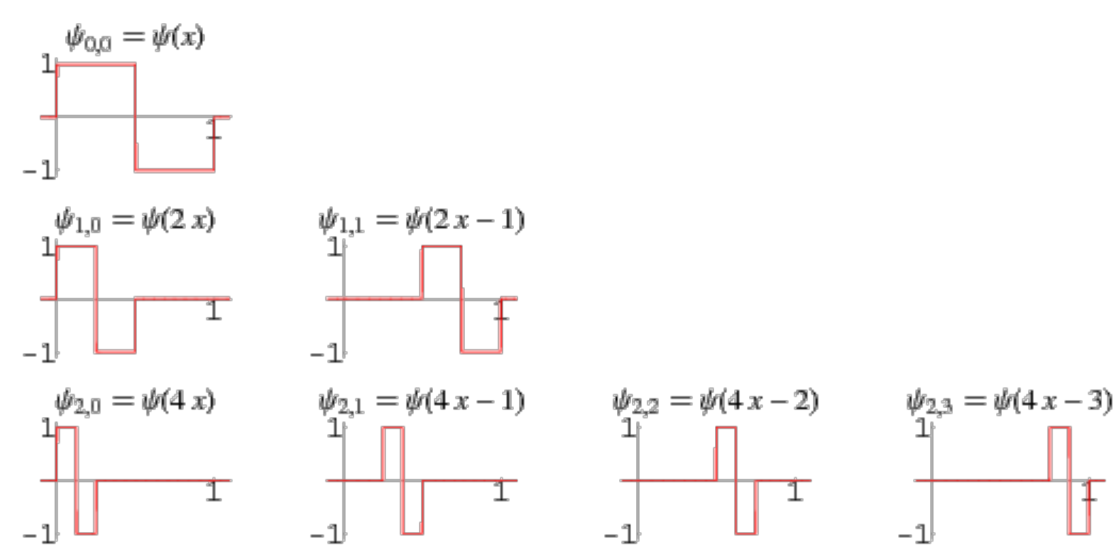


1980s: wavelet transform



- CWT is a redundant transform; however, unlike STFT, we can design an **orthogonal** wavelet transform through a **multi-resolution analysis**

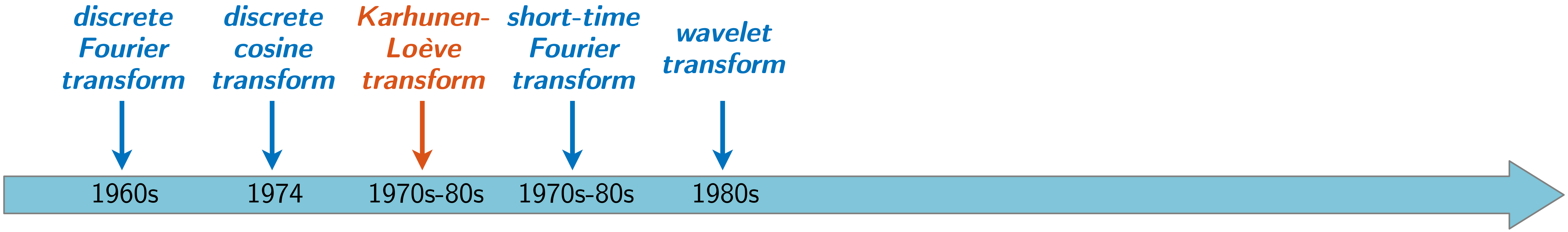
$$\psi_{l,m}(t) = \frac{1}{\sqrt{2^l}} \psi\left(\frac{t - 2^l m}{2^l}\right)$$



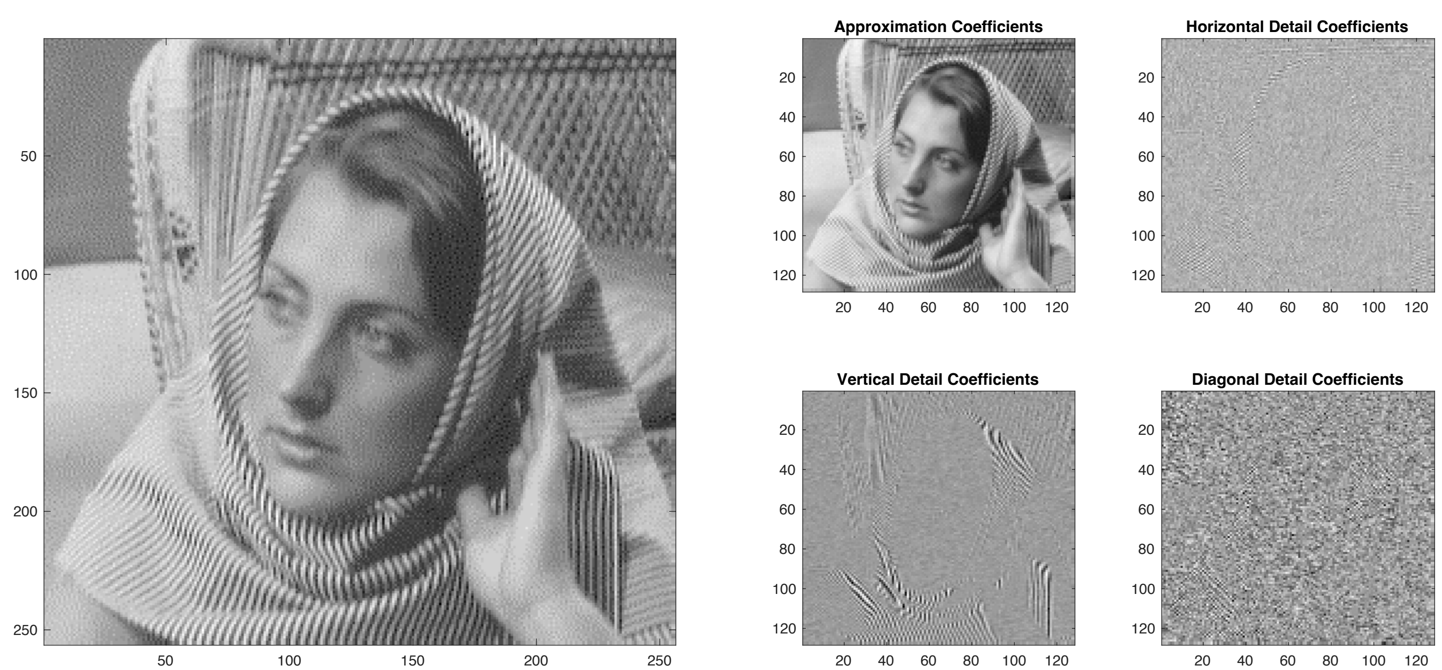
design principle

- functions at given scale l form an orthogonal basis of a space at l
- all functions across different scales are also orthogonal

1980s: wavelet transform

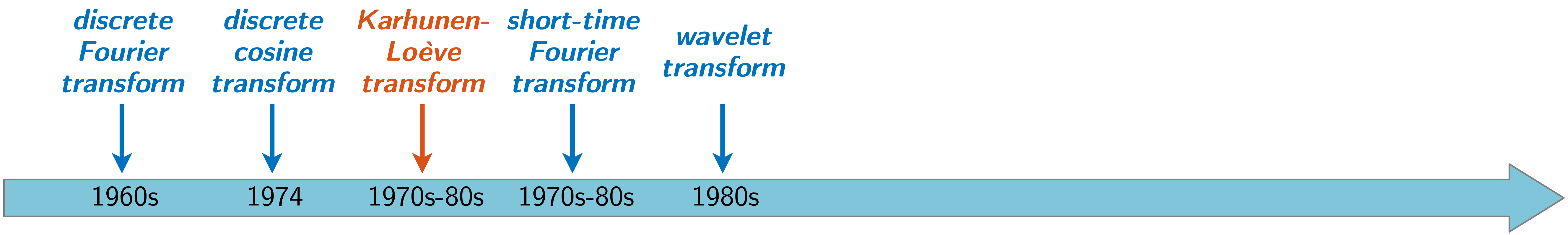


- this leads to the discrete wavelet transform (DWT) which provides an **orthogonal** dictionary

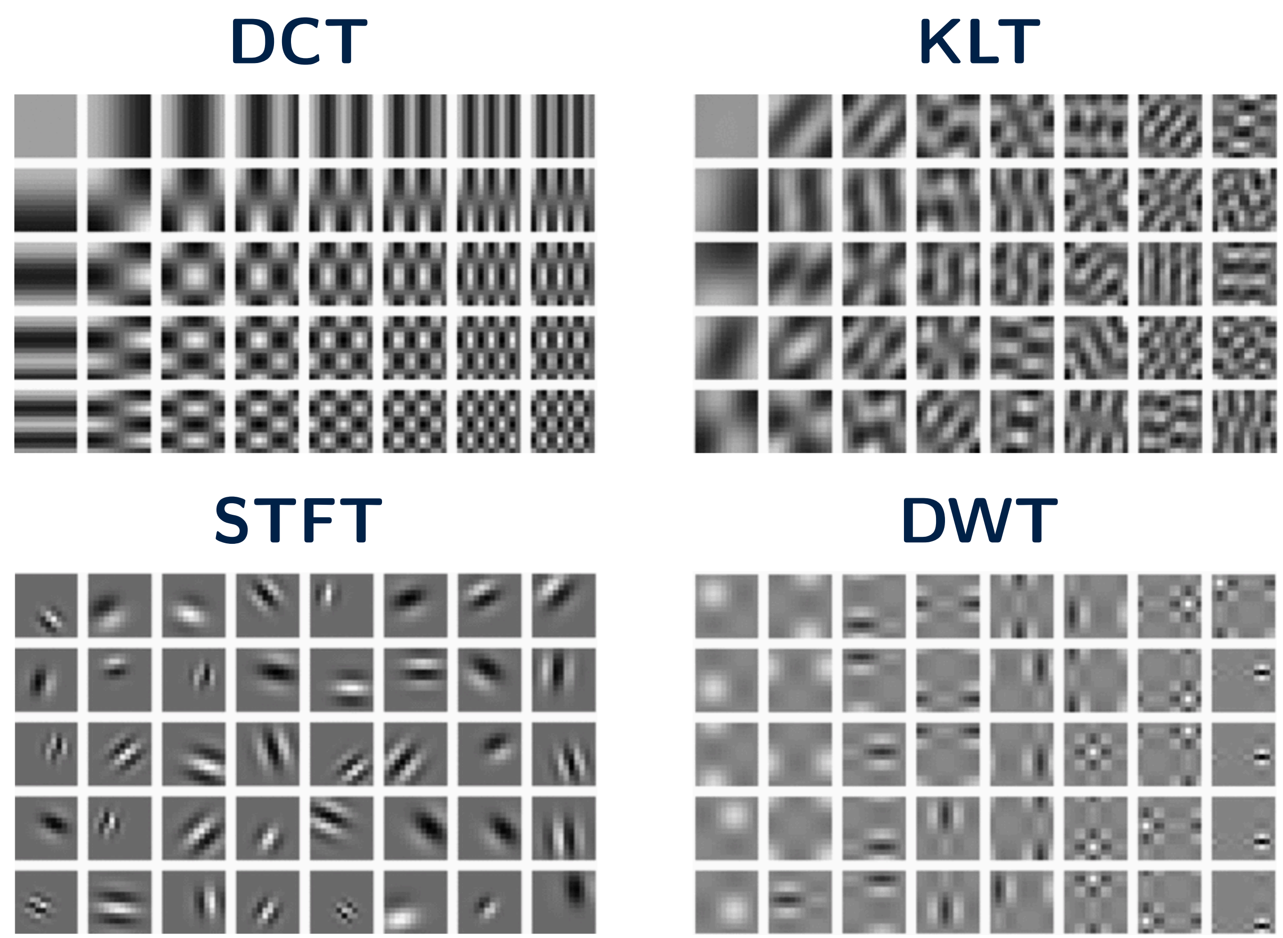


- DWT is behind the JEPG 2000 image compression standard

DCT vs KLT vs STFT vs DWT

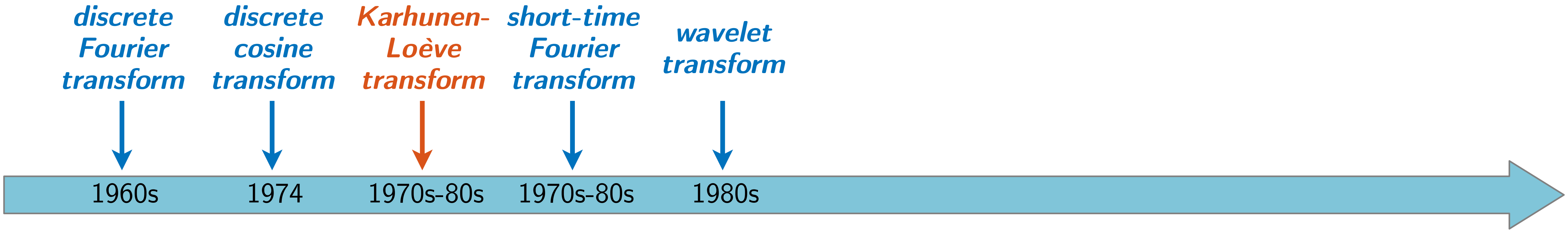


- comparison of the dictionaries we looked at so far



Rubinstein et al., "Dictionaries for sparse representation modeling," Proceedings of the IEEE, 2010.

Transform/analytic dictionary design

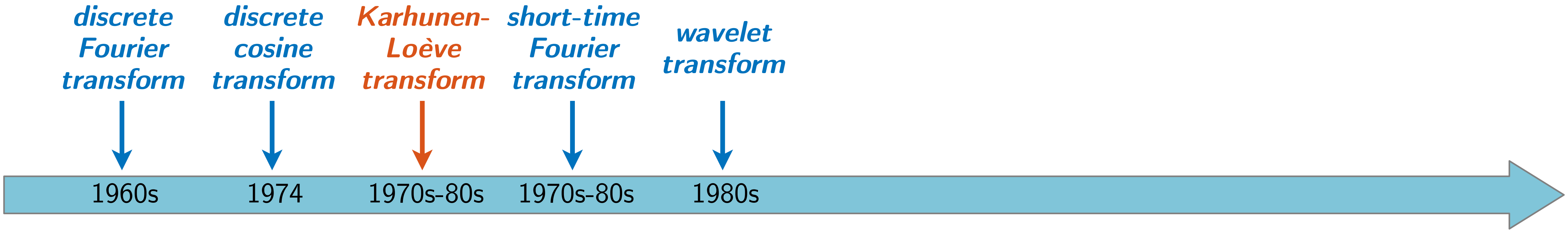


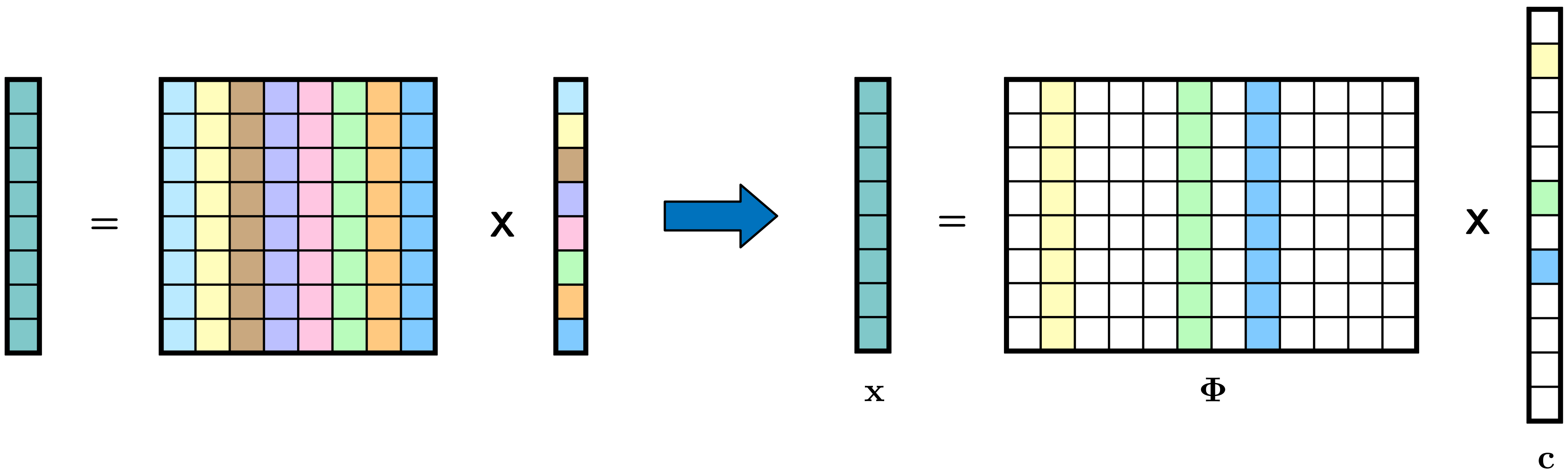
- summary
  - modelling data by a simpler class of mathematical functions
    - smooth functions (DFT, DCT)
    - piecewise-smooth functions (wavelets)
  - desired properties
    - localisation (STFT, wavelets)
    - multi-resolution (wavelets)
    - adaptivity (KLT, wavelet packets)
  - fast implementation is usually available
  - limited expressiveness

Outline

- A historical overview of dictionary design techniques
  - signal representation via stochastic models
  - transforms & analytic dictionaries
  - trained dictionaries (dictionary learning)
- Discussion
  - applications
  - connection with deep learning

A paradigm shift in dictionary design





orthogonal atoms

complete dictionary

all signals use all atoms

dense coefficients

mathematical modelling

non-orthogonal atoms

over-complete dictionary

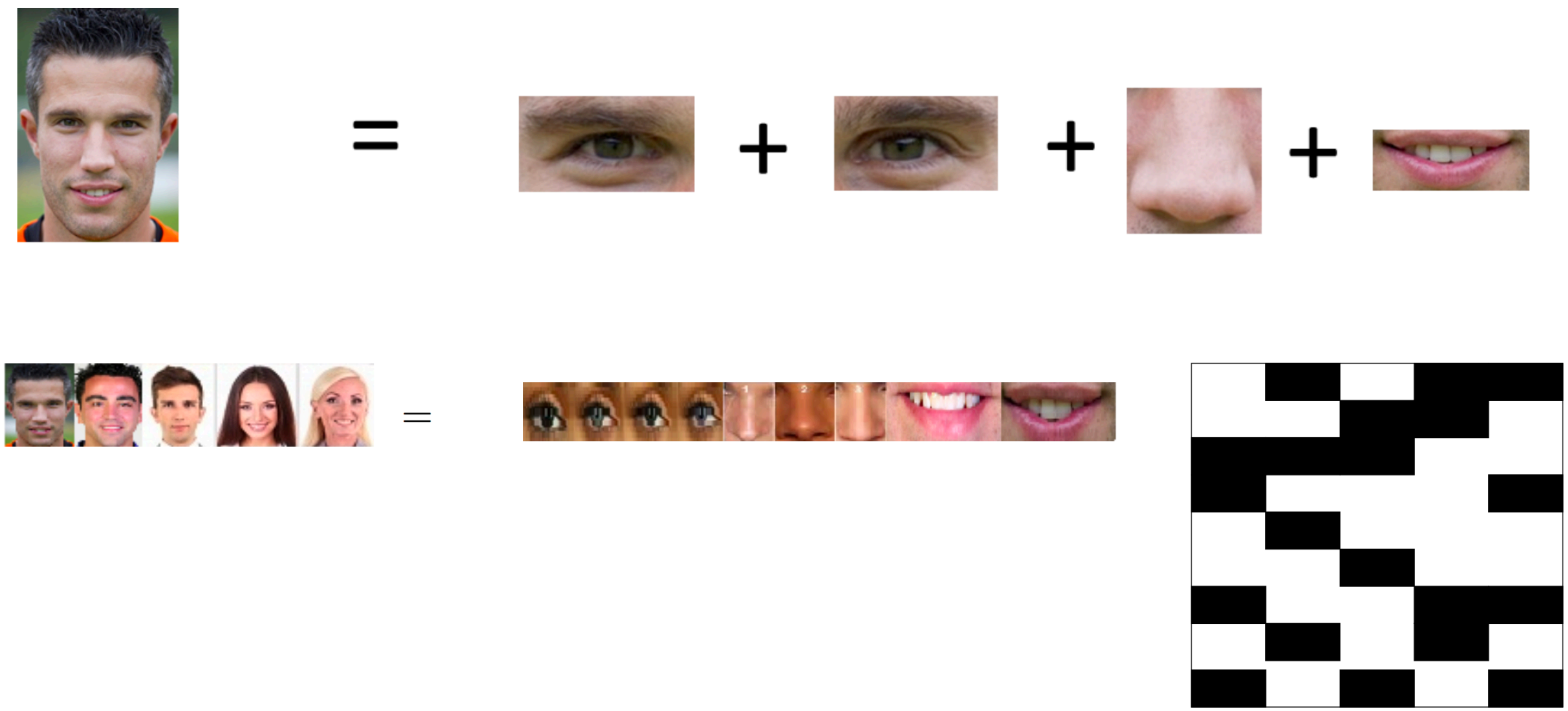
different signals use different atoms

sparse coefficients

adaptation to data realisations

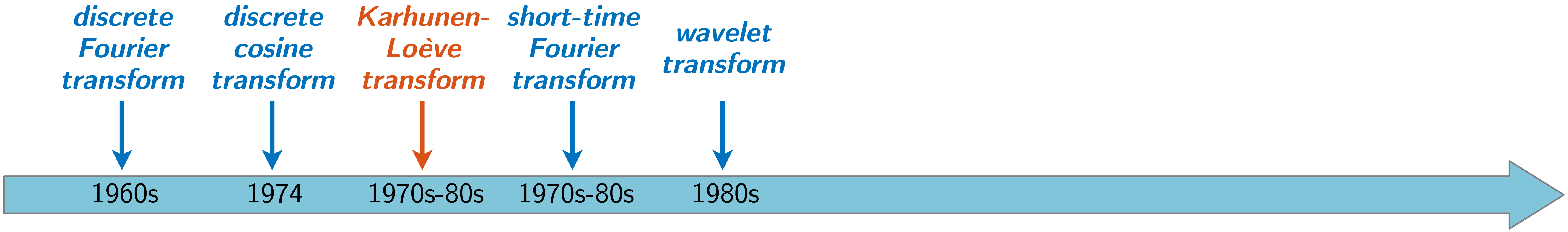
Illustrative example

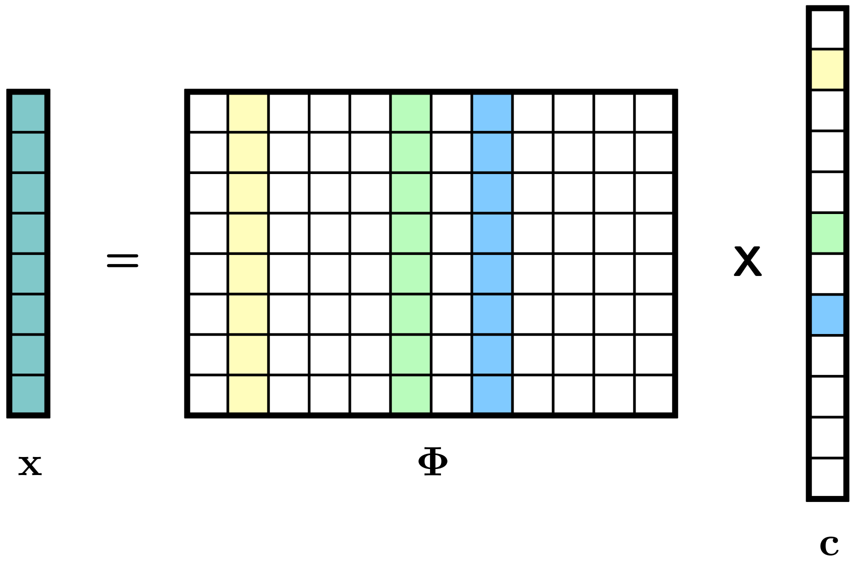
- **Modelling assumption:** Each data point is a combination of only a few (sparse) fundamental elements, i.e., dictionary atoms



Signals **Dictionary** **Coefficients**

Sparse representations





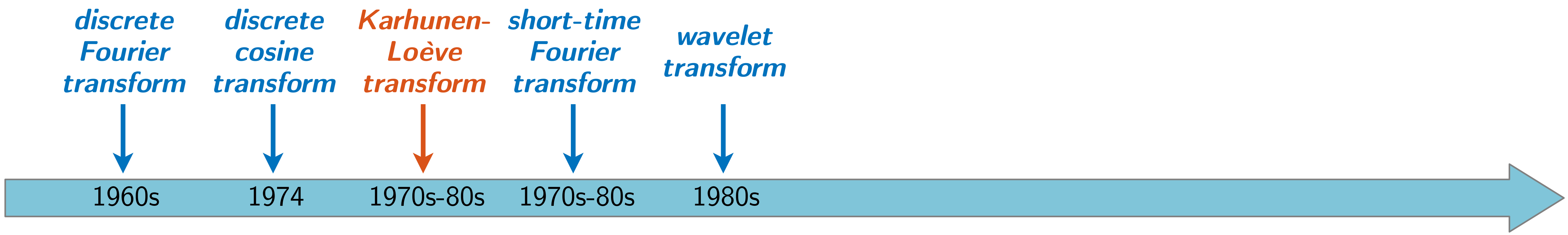
- given dictionary, express signal as linear combination of a small number of atoms

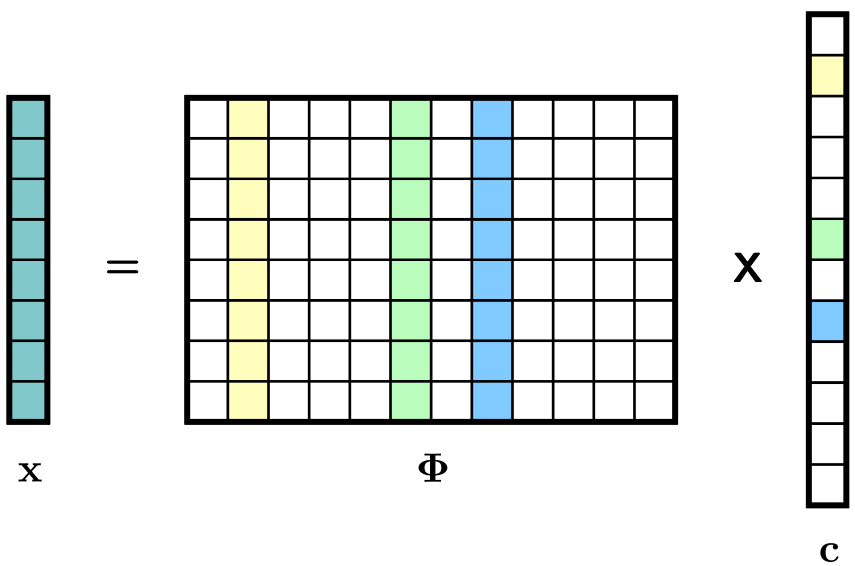
$$\min_{\mathbf{c}} ||\mathbf{c}||_0 \text{ subject to } \mathbf{x} = \boldsymbol{\Phi}\mathbf{c} + \boldsymbol{\eta} \text{ and } ||\boldsymbol{\eta}||_2^2 \leq \epsilon$$

- the problem is NP-hard
- two approximation algorithms
  - matching pursuit (MP)
  - least absolute shrinkage and selection operator (Lasso)

Mallat and Zhang, "Matching pursuits with time-frequency dictionaries," IEEE TSP, 1993.
Tibshirani, "Regression Shrinkage and Selection via the Lasso," Journal of the Royal Statistical Society: Series B, 1996.

Sparse representations

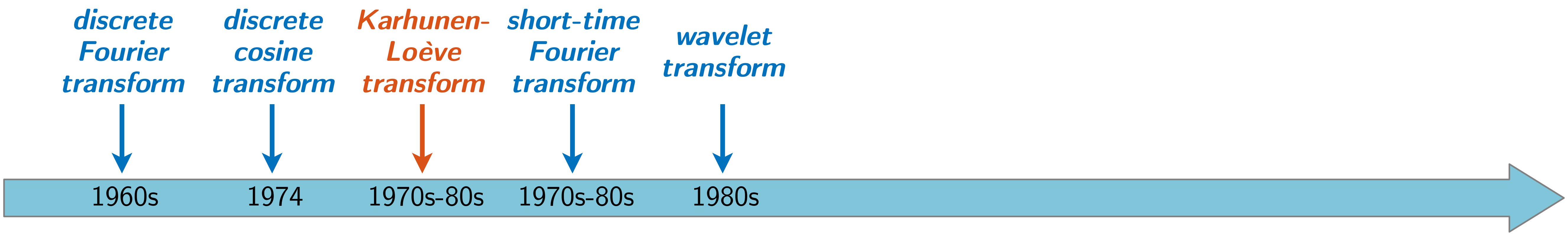


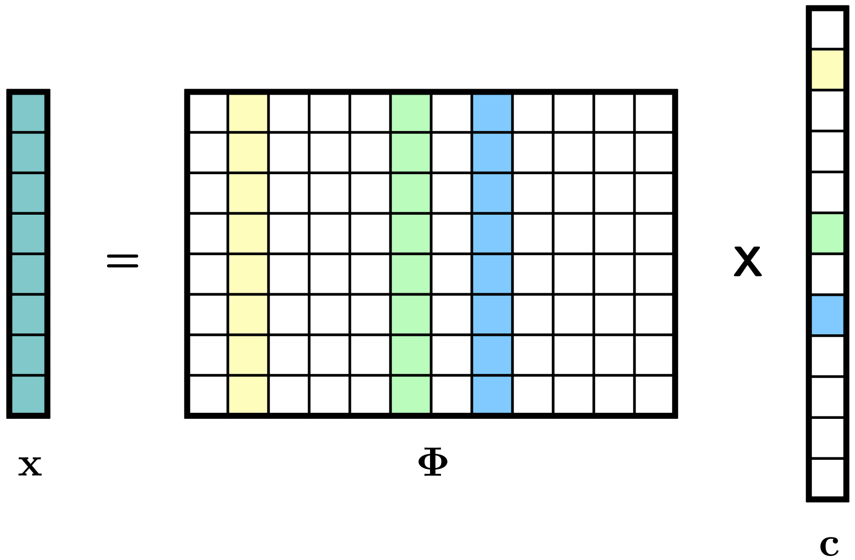


- MP
  - choose a subset of atoms from $\mathbf{\Phi}$
  - one atom at a time to maximally (greedily) reduce approximation error
- Lasso
  - solve a convex relaxation by replacing the 0-norm with 1-norm on **c**

$$\min_{\mathbf{c}} ||\mathbf{x} - \mathbf{\Phi}\mathbf{c}||_2^2 + \lambda||\mathbf{c}||_1$$

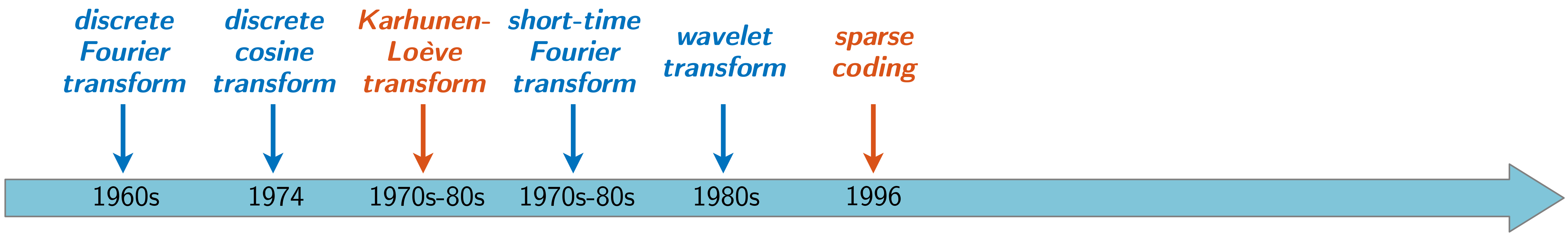
Sparse representations





- given dictionary, MP and Lasso can find a sparse approximation of the data
- the sparsity depends on not only data but also the **dictionary**
- finding optimised dictionaries is the goal of **dictionary learning**

Dictionary learning: Probabilistic approach



- probabilistic approach: maximum likelihood

$$\boldsymbol{\Phi}^* = \arg\max_{\boldsymbol{\Phi}} [\log\, P(\mathbf{x}|\boldsymbol{\Phi})]$$

$$= \arg\max_{\boldsymbol{\Phi}} [\log \int_{\mathbf{c}} P(\mathbf{x}|\mathbf{c}, \boldsymbol{\Phi}) P(\mathbf{c}) d\mathbf{c}]$$

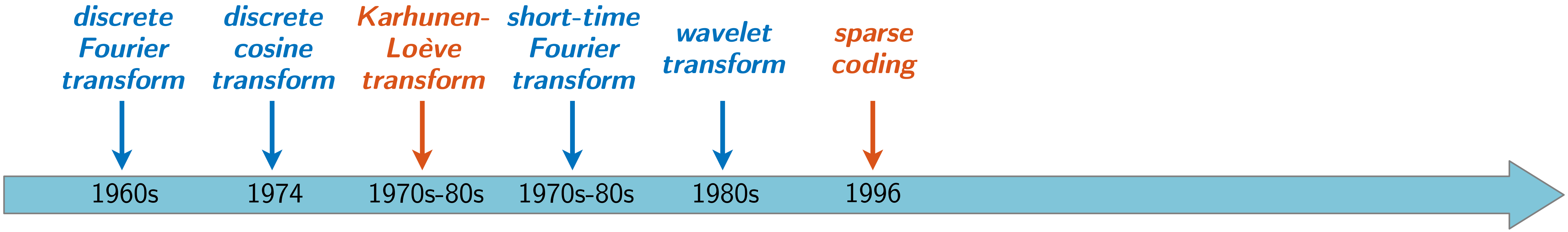
- assumption 1: Laplace distribution of coefficients $\mathbf{c}_i$
- assumption 2: Gaussian distribution of error $\boldsymbol{\eta}$

$$\boldsymbol{\Phi}^* = \arg\min_{\boldsymbol{\Phi},\mathbf{c}} -\log[P(\mathbf{x}|\mathbf{c}, \boldsymbol{\Phi}) P(\mathbf{c})]$$

$$= \arg\min_{\boldsymbol{\Phi},\mathbf{c}} ||\mathbf{x} - \boldsymbol{\Phi}\mathbf{c}||_2^2 + \lambda ||\mathbf{c}||_1$$

Olshausen and Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," Nature, 1996.

Dictionary learning: Probabilistic approach



- the problem is solved by iterating between two steps

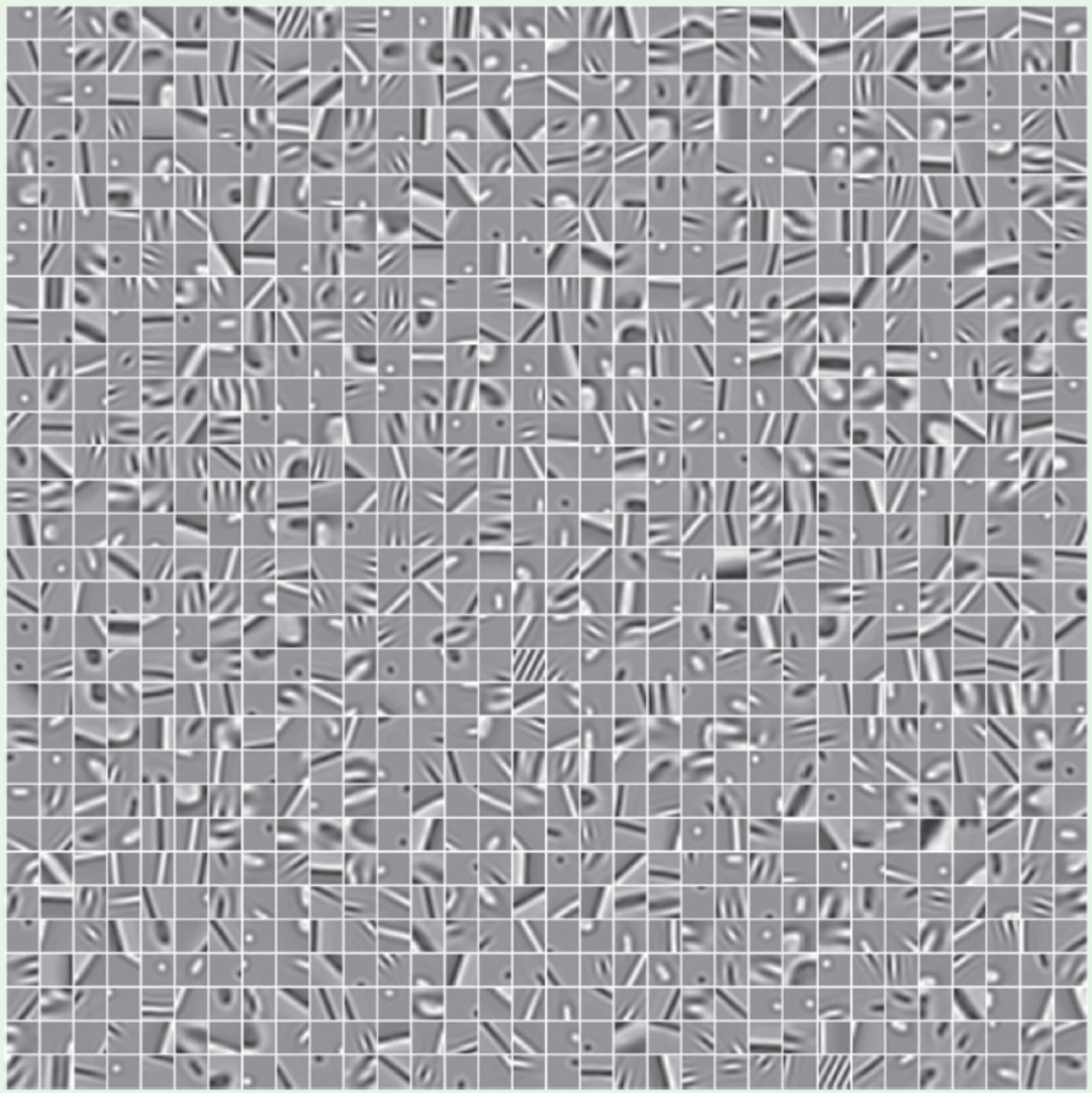
$$\min_{\boldsymbol{\Phi},\mathbf{c}} ||\mathbf{x} - \boldsymbol{\Phi}\mathbf{c}||_2^2 + \lambda||\mathbf{c}||_1$$

  - **sparse approximation:** given $\boldsymbol{\Phi}$, solve for **c** via Lasso
  - **dictionary update:** given **c**, update $\boldsymbol{\Phi}$ via gradient descent

- works at patch level for efficiency
- does not necessarily find global optimum
- trained atoms are remarkably similar to mammalian simple-cell receptive fields

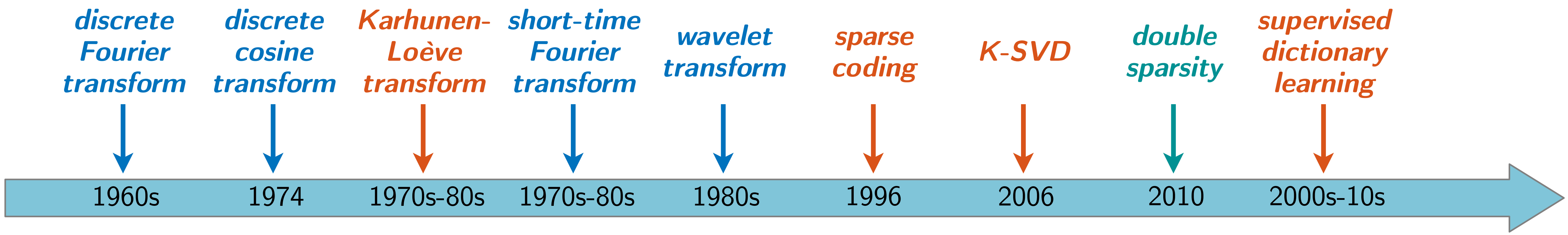
Olshausen and Field, "Emergence of simple-cell receptive field properties by learning a sparse code for natural images," Nature, 1996.

Dictionary learned with sparse coding



Olshausen et al., " "Learning real and complex overcomplete representations from the statistics of natural images," SPIE, 2009.

Dictionary learning

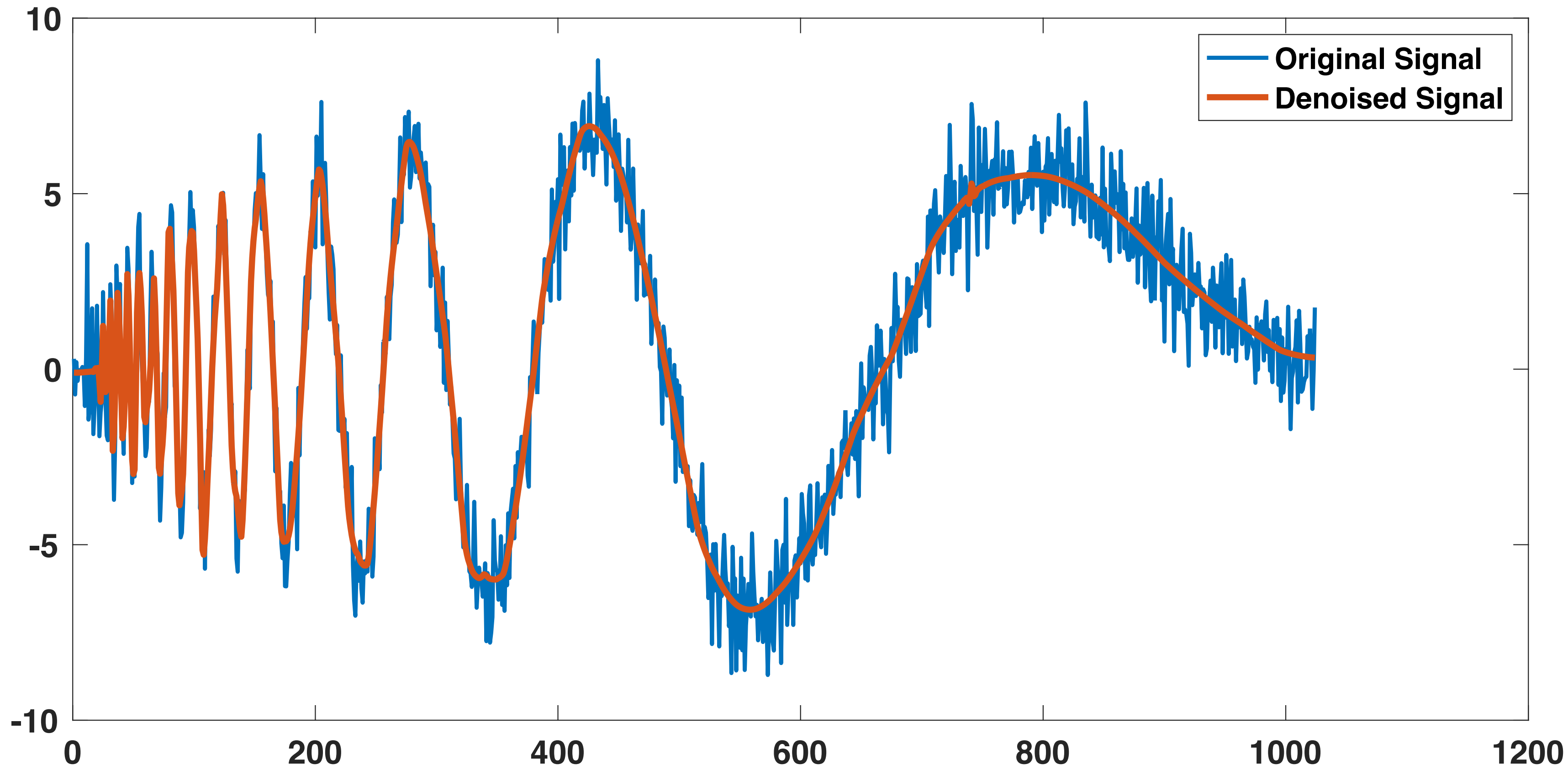


- summary
  - learning representations directly from data realisations
  - desired properties
    - over-completeness
    - sparse representations
    - efficiency in training
  - may be combined with analytical dictionary design
    - trained dictionary with structures (e.g., parametric dictionary learning)

Outline

- A historical overview of dictionary design techniques
  - signal representation via stochastic models
  - transforms & analytic dictionaries
  - trained dictionaries (dictionary learning)
- Discussion
  - applications
  - connection with deep learning

Signal denoising



denoising using the order 4 symlets wavelets

Image compression

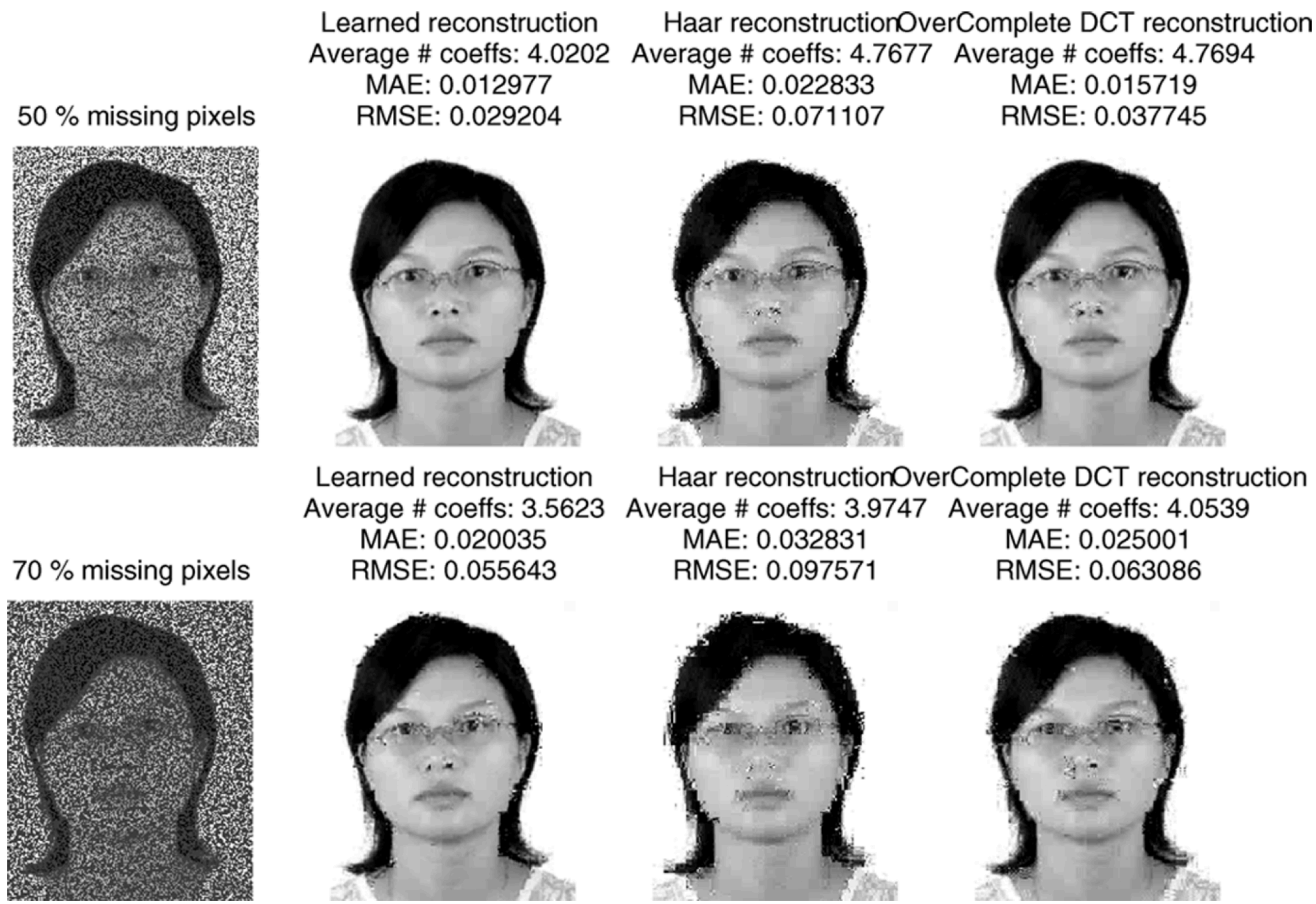
original

JPEG 2000 (10% in size)

JPEG 2000 (1% in size)

compression using the Cohen-Daubechies-Feauveau wavelets

Image reconstruction



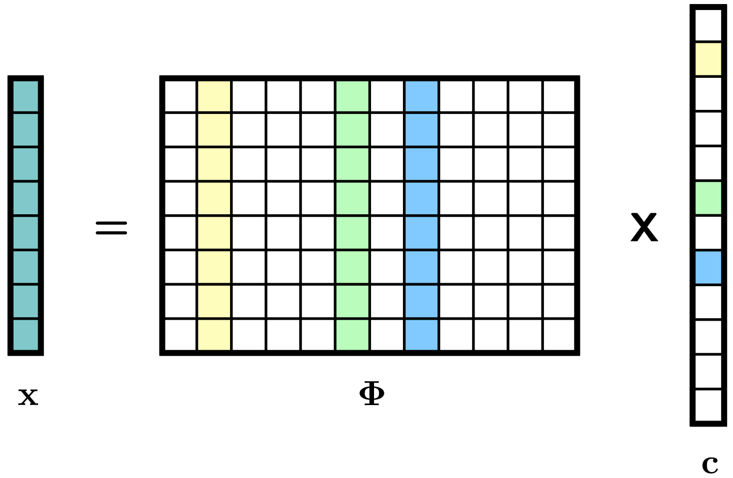
Aharon et al., "K-SVD: An algorithm for designing overcomplete dictionaries for sparse representation," IEEE TSP, 2006.

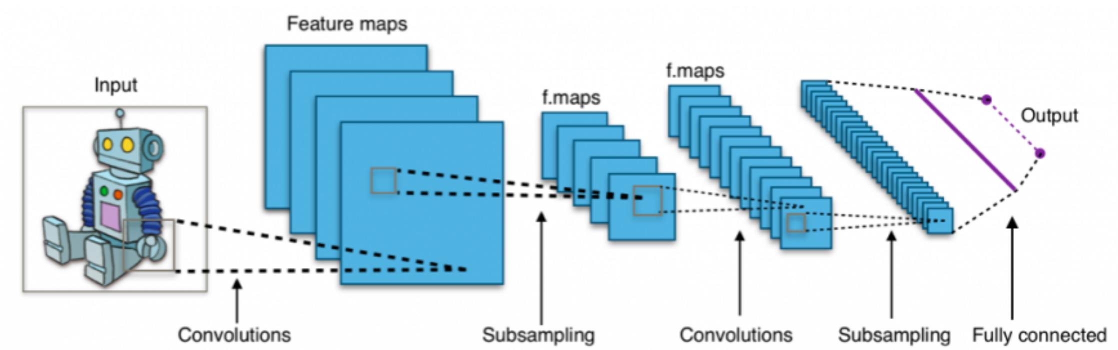
Image restoration

Mairal et al., "Sparse representation for color image restoration," IEEE TIP, 2008.

Connection with deep learning

- Dictionary learning vs. Deep learning
  - both extract **feature representations** from data realisations
  - both apply **sparsifying operations** such as shrinkage or rectified linear units
  - the former leads to representations that are not necessarily **hierarchical** (shallow model, no convolution operator, no pooling)
  - the former is normally for **reconstruction/approximation** (similar to autoencoders) while the latter is mainly for **classification**

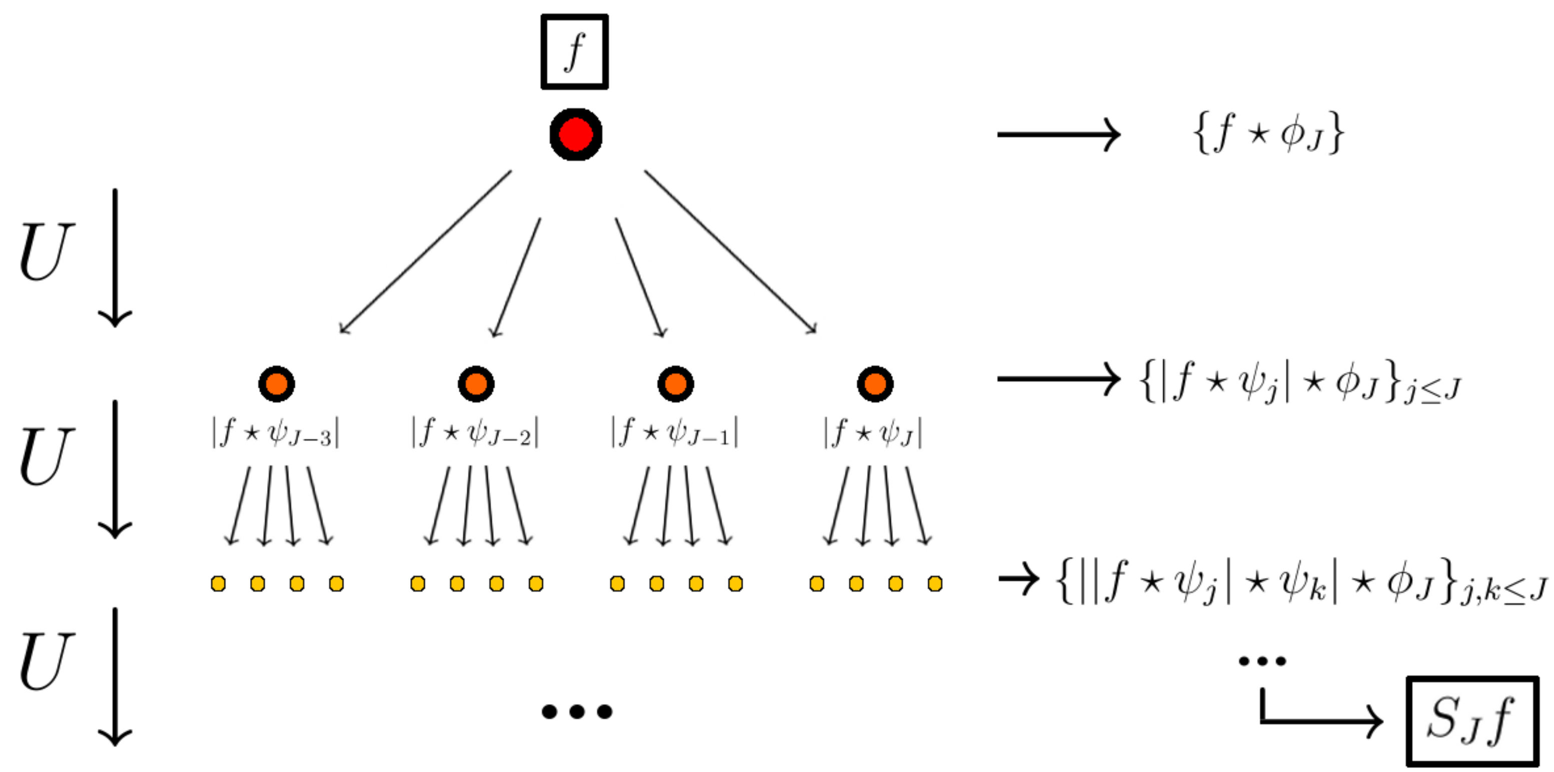




https://en.wikipedia.org/wiki/File:Typical_cnn.png

Dictionary-inspired deep architectures

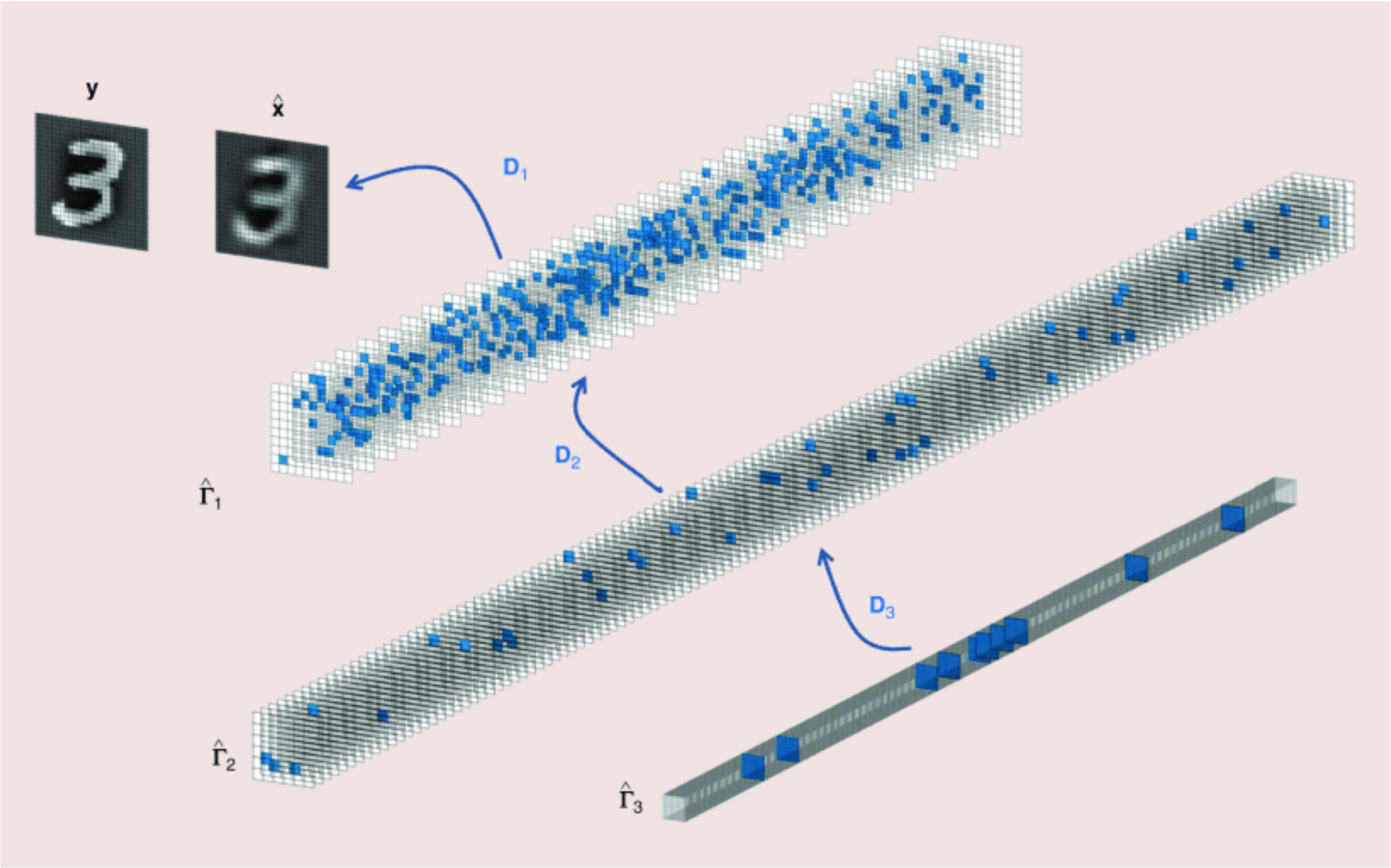
- Scattering transform



Bruna and Mallat, "Invariant scattering convolution networks," IEEE TPAMI, 2013.

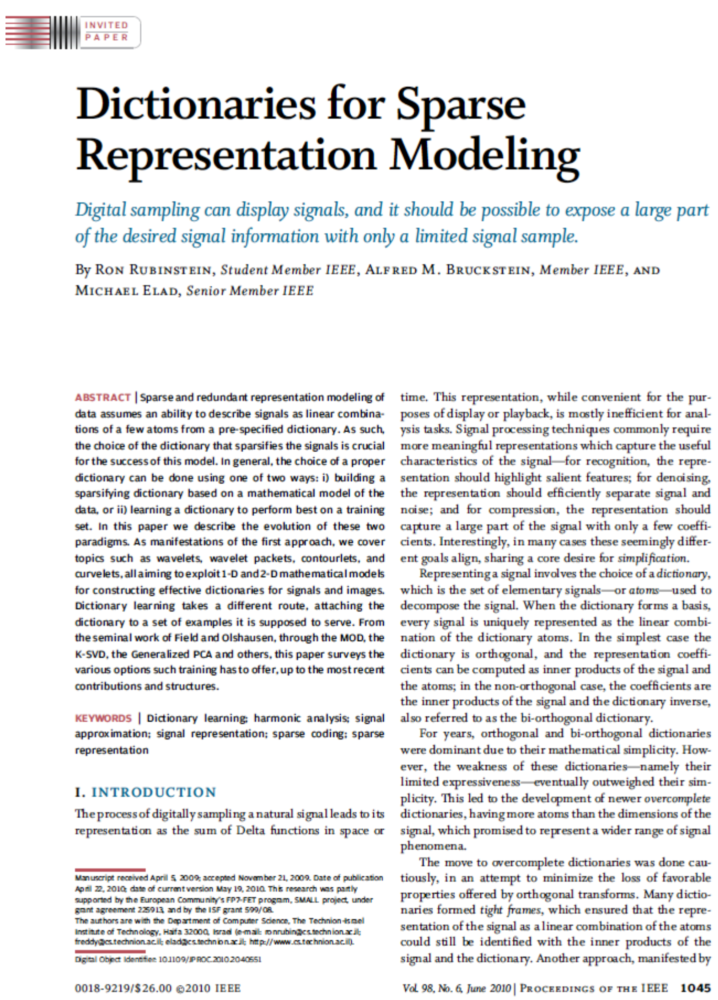
Dictionary-inspired deep architectures

- Multi-layer convolutional sparse coding



Papyan et al., "Theoretical foundations of deep learning via sparse representations," IEEE SPM, 2018.

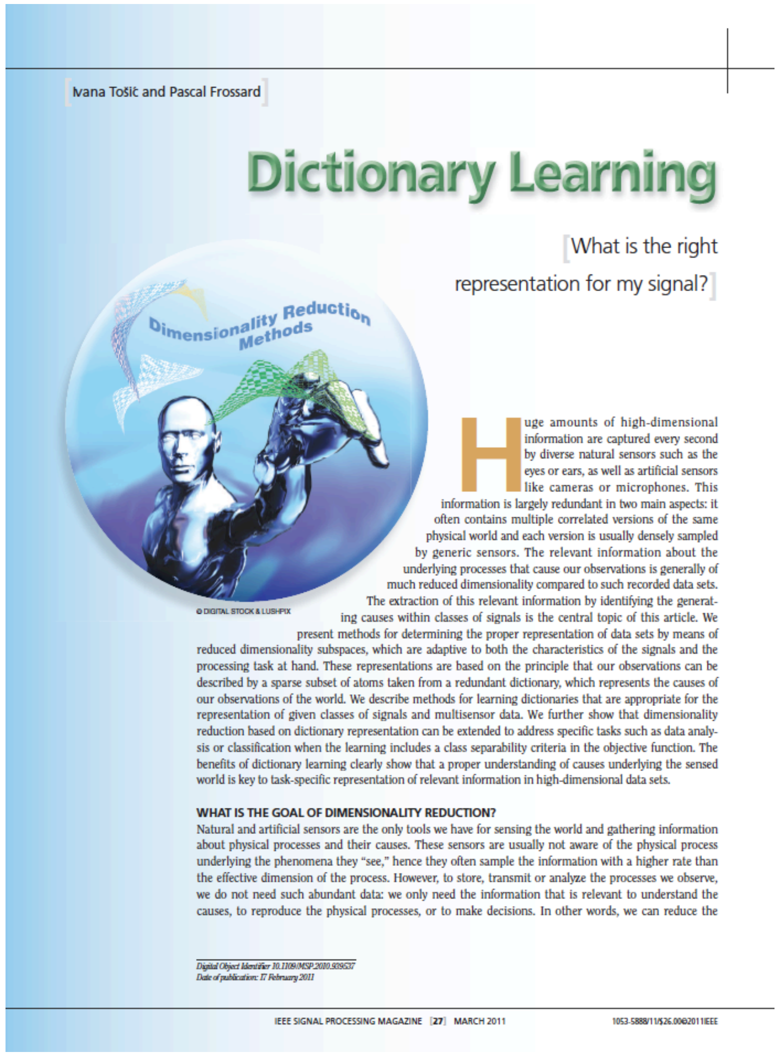
References



Dictionaries for Sparse Representation Modeling

Digital sampling can display signals, and it should be possible to expose a large part of the desired signal information with only a limited signal sample.

By Ron Rubinstein, *Student Member IEEE*, Alfred M. Bruckstein, *Member IEEE*, and Michael Elad, *Senior Member IEEE*



Ivana Tošić and Pascal Frossard

Dictionary Learning

What is the right representation for my signal?

Representation Learning: A Review and New Perspectives

Yoshua Bengio, Aaron Courville, and Pascal Vincent