Filters in Image Processing

Introduction & Some revision

David Svoboda

email: svoboda@fi.muni.cz
Centre for Biomedical Image Analysis
Faculty of Informatics, Masaryk University, Brno, CZ

CBIA

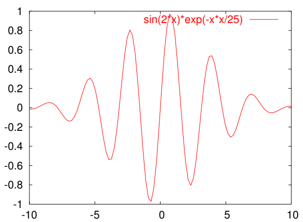
September 16, 2019

Outline

Introduction

Basic information

- Lecture + seminar = 2 + 2 hours per week.
- Final exam is written & spoken and is focused on your skills rather than knowledge.
- Basic knowledge of English and math (calculus, statistics, algebra) is highly recommended.
- Digital Image Processing (PV131) is highly recommended.
- Seminars take place in PC labs using MATLAB®
- The experience from seminars will be useful for completing a small team (two students) project written in MATLAB®, C/C++, Java (or the preferred language).
- At the end of each lecture you can find a list of questions you should be able to answer if you want to pass the final exam.



Introduction

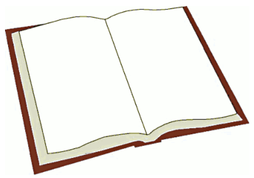
Structure of a lecture

1. Introduction & Some revision
2. Fourier transform, Fast Fourier transform
3. Image resampling, Texture filtering
4. Principal component analysis, Discrete cosine transform
5. Subband coding, Wavelet Transform, Discrete WT
6. Z-transform, recursive filtering
7. Edge detection
8. Image compression
9. Image descriptors (Haralick, SIFT, MPEG-7, ...)
10. Image restoration
11. Steerable filters

Introduction

Bibliography

- Gonzalez, R. C., Woods, R. E., Digital image processing / 2nd ed., Upper Saddle River: Prentice Hall, 2002, pages 793, ISBN 0201180758
- Bracewell, R. N., Fourier transform and its applications / 2nd ed. New York: McGraw-Hill, pages 474, ISBN 0070070156
- Jähne, B., Digital image processing / 6th rev. and ext. ed., Berlin: Springer, 2005, pages 607, ISBN 3540240357
- selected papers



Digital Filters in Image Processing

Some examples

- Linear mappings (e.g. Fourier or Wavelet transform)
- Denoising (e.g. median filtering)
- Point based transforms (e.g. thresholding, contrast, brightness)
- (Re)sampling (e.g. nearest neighbour, bilinear, Lanczos)
- Texture filtering (e.g. anisotropic filtering)
- Edge detection (e.g. Sobel, Canny)
- Quantization (common in lossy compression techniques)
- More ...

Signal Filter

Definition

Filter $\mathcal{F}$ is any system having its input/output:

$$h = \mathcal{F}(f)$$

- $f(\mathbf{x})$ or $f(\mathbf{m})$ ... input image/function/signal
- $h(\mathbf{y})$ or $h(\mathbf{n})$ ... output image/function/signal
- $\mathcal{F}$ ... filter (functor)
- $\mathbf{x}, \mathbf{y}$ ... continuous signal
- $\mathbf{m}, \mathbf{n}$ ... discrete sequence

Convolution

1D convolution

- Discrete: given two 1D signals $f(i)$ and $g(i)$:

$$(f * g)(i) \equiv \sum_k f(k)g(i-k)$$

- Continuous: given two 1D signals $f(x)$ and $g(x)$:

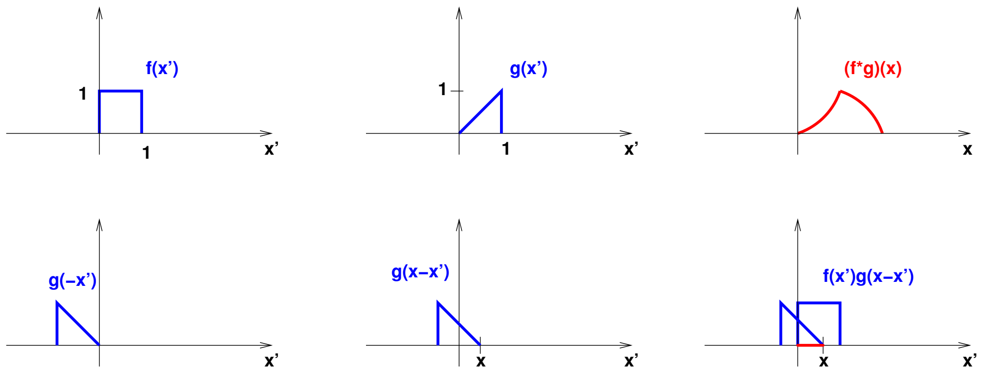
$$(f * g)(x) \equiv \int_{-\infty}^{\infty} f(x')g(x-x')dx'$$

Notice: 'g' is called *a convolution kernel (mask)*

Convolution

An example

1D convolution



Try: `http://www.jhu.edu/~signals/convolve/index.html`

Convolution

2D convolution

- Discrete: given two 2D signals $f(i,j)$ and $g(i,j)$:

$$(f * g)(i,j) \equiv \sum_{k,l} f(k,l) g(i-k, j-l)$$

- Continuous: given two 2D signals $f(x,y)$ and $g(x,y)$:

$$(f * g)(x,y) \equiv \int \int f(x',y') g(x-x', y-y') dx' dy'$$

Notice: If not necessary we will focus only on 1D (discrete) convolution.

Impulse symbol δ

Definition

Infinitely brief and infinitely strong unit-area impulse:

$$\delta(x) = 0 \quad x \neq 0$$

and

$$\int_{-\infty}^{\infty} \delta(x) dx = 1$$

- we call it Dirac delta function or impulse symbol
- it is NOT a function

Impulse symbol δ

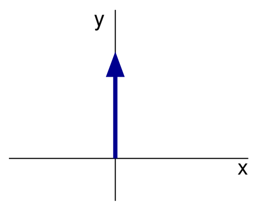
Some properties

Given 1D function f and $a \in \mathbb{R}$:

$$\int_{-\infty}^{\infty} \delta(x) f(x) dx = f(0)$$

$$\int_{-\infty}^{\infty} \delta(x-a) f(x) dx = f(a)$$

$\delta(x)$ plot:



Kronecker delta (function)

Kronecker delta function ...discrete counterpart to Dirac delta impulse.

$$\delta_{i,j} = \begin{cases} 1 & \text{if } (i = j) \\ 0 & \text{otherwise} \end{cases}$$

or

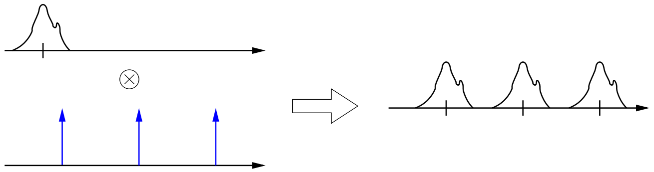
$$\delta(n) = \begin{cases} 1 & \text{if } (n = 0) \\ 0 & \text{otherwise} \end{cases}$$

Convolution

With some important functions

Convolution of any function f with:

- δ impulse shifts the origin of f to the nonzero response of δ
- δ impulses replicate the function f



- Gaussian shifts the origin of f to the position of the peak of the Gaussian and smooths

Convolution properties

Commutativity

Given two signals $f(i)$ and $g(i)$:

$$(f * g)(i) = (g * f)(i)$$

Proof:

$$\begin{aligned}(f * g)(i) &= \sum_{j=-\infty}^{\infty} f(j)g(i-j) = \sum_{j=-\infty}^{\infty} g(i-j)f(j) \\ & /\text{subst. } k = i - j; j = i - k/ \\ &= \sum_{k=-\infty}^{\infty} g(k)f(i-k) \\ &= (g * f)(i) \end{aligned}$$

$\square$

Convolution properties

Associativity

Given three signals $f(i)$, $g(i)$, and $h(i)$:

$$((f * g) * h)(i) = (f * (g * h))(i)$$

Proof:

$$\begin{aligned}((f * g) * h)(i) &= \sum_j (f * g)(j)h(i-j) = \sum_j \left[\sum_k f(k)g(j-k)\right] h(i-j) \\ &= \sum_j \sum_k f(k)g(j-k)h(i-j) \\ & /\text{subst. } l = j - k; j = k + l/ \\ &= \sum_l \sum_k f(k)g(l)h(i-k-l) = \sum_k f(k) \sum_l g(l)h(i-k-l) \\ &= \sum_k f(k)\left[(g * h)(i-k)\right] = (f * (g * h))(i) \quad \square \end{aligned}$$

Convolution properties

Separable kernels in 2D

2D kernel $g(i,j)$ is called *separable* if there exist two 1D vectors g_{row}, g_{col} such that:

$$g = g_{row} * g_{col}^T$$

Convolution with 2D separable kernel = two consecutive convolutions with 1D kernels:

$$\begin{aligned}(f * g)(i,j) &= (f * (g_{row} * g_{col}))(i,j) \\ & \quad /associativity/ \\ &= ((f * g_{row}) * g_{col})(i,j)\end{aligned}$$

Notice: 2D kernel is separable if the rank of its matrix is equal to 1.

Convolution properties

Separable kernels in 2D

Examples

- Gaussian:

$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} * \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

- Sobel:

$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ -1 \end{bmatrix} * \begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

There are also *separable functions*. An example of such function is 2D Dirac impulse: $\delta(x,y) = \delta(x)\delta(y)$

Convolution properties

Linearity & Position invariance

If convolution kernel 'g' is fixed (which might be valid for optical systems, for example) then we can write:

$$f * g = O_g(f)$$

The operator O_g is:

- linear – given the images f_1, f_2, and any $\alpha, \beta \in \mathbb{R}$, it holds:

$$O_g(\alpha f_1 + \beta f_2) = \alpha O_g(f_1) + \beta O_g(f_2)$$

- position invariant – if $h(\mathbf{x}) = O_g(f(\mathbf{x}))$ then also

$$\forall \mathbf{y} : h(\mathbf{x} - \mathbf{y}) = O_g\left[f(\mathbf{x} - \mathbf{y})\right]$$

Convolution properties

Convolution theorem

$$\mathcal{F}(f * g) = \mathcal{F}(f) \cdot \mathcal{F}(g)$$

$$\mathcal{F}(f \cdot g) = \mathcal{F}(f) * \mathcal{F}(g)$$

where

- $\mathcal{F}$ . . . Fourier transform
- "·" . . . point-wise multiplication
- "$*$" . . . convolution
- f, g . . . images

Notice: Proof is coming soon.

Convolution properties

Why is it good to know/understand them?

Commutativity

- convolutions can be reordered in random sequence
- images becomes convolution kernels, and vice versa

Associativity

- parenthesis can be moved without affecting the result
- choosing the simpler way of evaluation – different position of parenthesis may change the complexity

Kernel separability

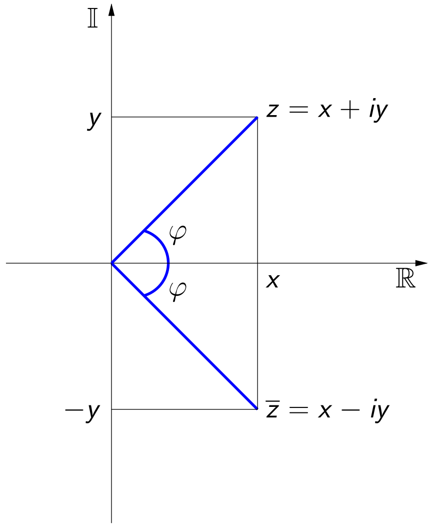
- convolution with 2D kernels . . . $O(n^2)$
- convolution with 1D kernels . . . $O(n)$

$\mathbb{C}$omplex numbers

Any $z \in \mathbb{C}$ can be written in one of the following ways:

- $z = x + iy$
- $z = |z|\,(\cos\varphi + i\sin\varphi)$
- $z = |z|e^{i\varphi}$

where $x, y \in \mathbb{R}$ and $i^2 = -1$ is a constant, $|z|$ is a magnitude and φ is a phase of z



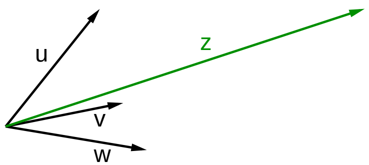
Properties:

- conjugate complex number: $\overline{z} = x - iy$

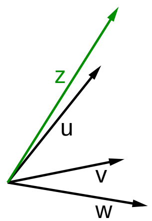
Vector composition

Let be given a Euclidean ($\mathbb{K} = \mathbb{R}$) or unitary ($\mathbb{K} = \mathbb{C}$) vector space $\mathbb{V} \subseteq \mathbb{K}^n$ and three vectors $\mathbf{u}, \mathbf{v}, \mathbf{w} \in \mathbb{V}$:

- Vector addition: $\mathbf{z} = \mathbf{u} + \mathbf{v} + \mathbf{w} \in \mathbb{V}$



- Linear combination of vectors: $\mathbf{z} = \frac{1}{2}\mathbf{u} + 3\mathbf{v} - 2\mathbf{w} \in \mathbb{V}$



Vector decomposition

Let be given Euclidean space $\mathbb{V} = \langle \mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_n} \rangle$, then each $\mathbf{v} \in \mathbb{V}$ can be written as:

$$\mathbf{v} = a_1\mathbf{u_1} + a_2\mathbf{u_2} + \cdots + a_n\mathbf{u_n}$$

where

- $(\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_n})$ is the basis of $\mathbb{V}$
- $\forall i = \{1, \ldots, n\} : a_i \in \mathbb{K}$
- vector $(a_1, a_2, \ldots, a_n)$ is unique.

Notes:

- two vectors $\mathbf{u}, \mathbf{v} \in \mathbb{V}$ are orthogonal, if $\mathbf{u} \cdot \mathbf{v} = 0$ ('$\cdot$' stands to inner product)
- basis $(\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_n})$ is orthonormal, if $\forall i, j = 1, \ldots, n : \mathbf{u_i} \cdot \mathbf{u_j} = \delta_{i,j}$ ($\delta_{i,j}$ stands for Kronecker delta)

Vector decomposition

Example

Given Cartesian coordinate system $\langle \mathbf{e_1}, \mathbf{e_2}, \mathbf{e_3} \rangle$ and a vector $\mathbf{v} = (3.4, -2, 7)$, we can write:

$$\mathbf{v} = 3.4\mathbf{e_1} - 2\mathbf{e_2} + 7\mathbf{e_3}$$

where

$$\begin{aligned} \mathbf{e_1} &= (1, 0, 0) \\ \mathbf{e_2} &= (0, 1, 0) \\ \mathbf{e_3} &= (0, 0, 1) \end{aligned}$$

Question: How to find the (linear combination) coefficients when we do not project the vector $\mathbf{v}$ onto standard basis?

Vector decomposition

Conversion to another basis

Given a vector $\mathbf{v} \in \mathbb{V}$ and "any" basis $(\mathbf{u_1}, \mathbf{u_2}, \ldots, \mathbf{u_n})$ in $\mathbb{V}$, we can write:

$$\mathbf{v} = a_1\mathbf{u_1} + a_2\mathbf{u_2} + \cdots + a_n\mathbf{u_n}$$

where

$$\forall i = \{1, \ldots, n\} : a_i = \frac{\mathbf{v} \cdot \mathbf{u_i}}{\mathbf{u_i} \cdot \mathbf{u_i}}$$

If the basis is orthonormal, it is sufficient to write: $a_i = \mathbf{v} \cdot \mathbf{u_i}$

Notice: Inner product $\mathbf{v} \cdot \mathbf{w}$ is a projection $\mathbf{v}$ onto $\mathbf{w}$. The result is a number.

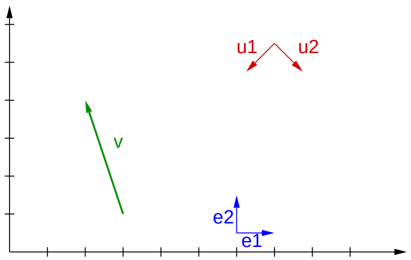
Vector decomposition

Example

- standard basis: $\langle \mathbf{e_1}, \mathbf{e_2} \rangle = \langle (1, 0), (0, 1) \rangle$
  $\mathbf{v}_{\langle \mathbf{e_1}, \mathbf{e_2} \rangle} = (-1, 3)$
- another basis: $\langle \mathbf{u_1}, \mathbf{u_2} \rangle = \langle (-0.7, -0.7), (0.7, -0.7) \rangle$ $(0.7 \doteq \frac{\sqrt{2}}{2})$

$$\begin{aligned} a_1 &= \frac{(-1, 3) \cdot (-0.7, -0.7)}{(-0.7, -0.7) \cdot (-0.7, -0.7)} \doteq -1.42 \\ a_2 &= \frac{(-1, 3) \cdot (0.7, -0.7)}{(0.7, -0.7) \cdot (0.7, -0.7)} \doteq -2.86 \end{aligned}$$

$\mathbf{v}_{\langle \mathbf{u_1}, \mathbf{u_2} \rangle} = (-1.42, -2.86)$



Vector decomposition

Matrix notation

Each orthonormal basis can form a square matrix A:

$$A = \begin{bmatrix} \mathbf{u_1} \\ \mathbf{u_2} \end{bmatrix} = \begin{bmatrix} -0.7 & -0.7 \\ 0.7 & -0.7 \end{bmatrix}$$

The projection is realized using matrix multiplication:

$$\mathbf{v}_{\langle \mathbf{u_1}, \mathbf{u_2} \rangle} = A\mathbf{v}_{\langle \mathbf{e_1}, \mathbf{e_2} \rangle}$$

Notice: Transform (mapping) from one basis onto another one is realized using matrix multiplication $\Rightarrow$ linear mapping.

Vector decomposition

matrix notation

Properties of transform matrix A

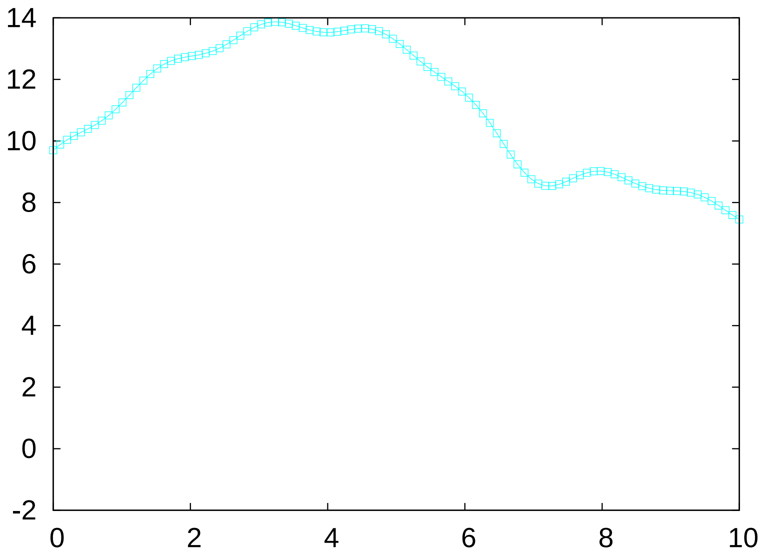
- A is unitary matrix, i.e. $A^{-1} = \overline{A}^T$.
- If $y = Ax$ is forward transform, then $x = A^{-1}y = \overline{A}^T y$ is inverse transform.

Let us do the same with functions
(n-dimensional vectors)

Function decomposition

An example

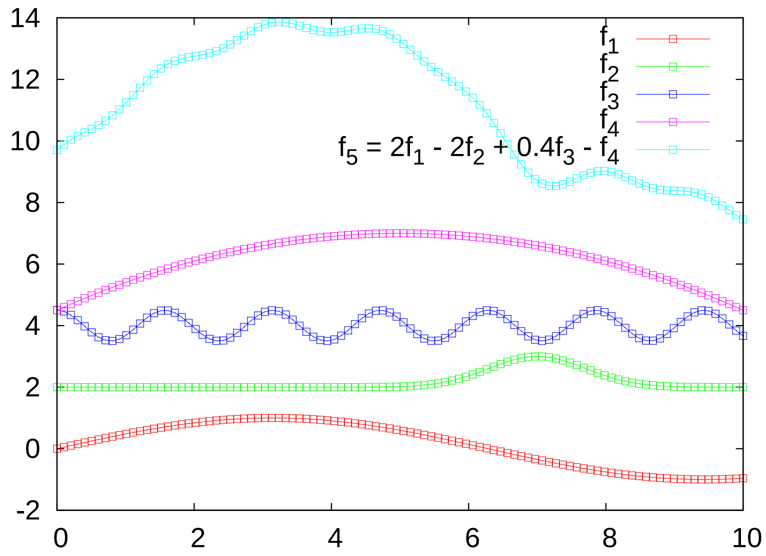
How can we decompose the following function?



Function decomposition

An example

We can express it as the following linear combination:

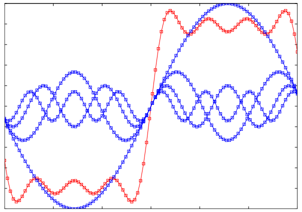
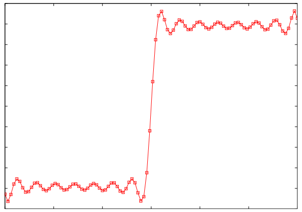


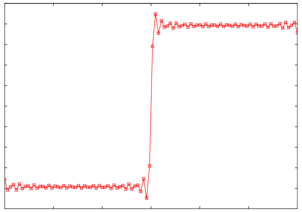
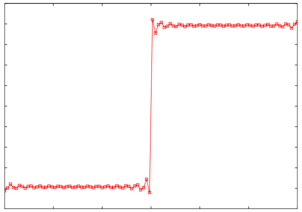
Function decomposition

Another example

A step function is defined as an infinite sum of sine waves:

$$f_z(m) = \sum_{n=0}^{z} \frac{sin\{(2n+1)m\}}{2n+1}$$

$f_3(m)$  $f_{10}(m)$ 

$f_{20}(m)$  $f_{35}(m)$ 

Function decomposition in 1D

Let be a discrete 1D function f of N samples:

- f is a point in some N-dim vector space $\mathbb{V} \subseteq \mathbb{K}^N$ $(\mathbb{K} = \mathbb{R}$ or $\mathbb{C})$
- f can be expressed as a linear combination of basis functions:

$$f(m) = \sum_{k=1}^{N} a_k \varphi_k(m)$$

where $a_k \in \mathbb{K}$ and $(\varphi_1, \varphi_2, \ldots, \varphi_N)$ form the orthonormal basis

The coefficients of linear combination are found in the common way:

$$\forall k = \{1, \ldots, N\} : a_k = f \cdot \varphi_k$$

i.e. using the projection (inner product)

Notice: $f \cdot \varphi_k = \sum_m f(m)\overline{\varphi_k(m)}$

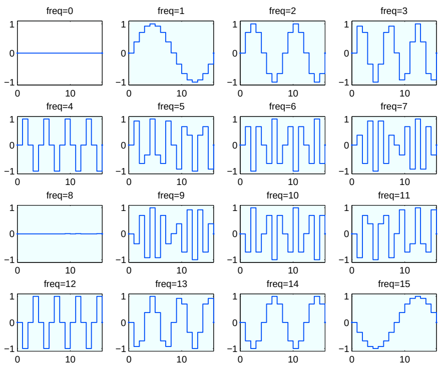
Basis functions

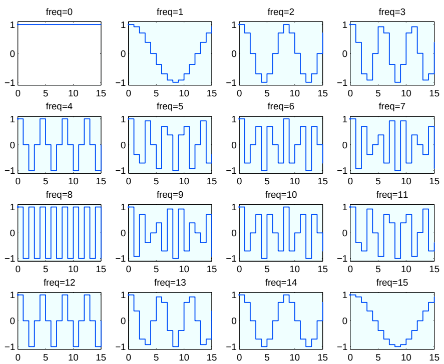
An example of sine & cosine waves sampled with $N = 16$

Common request:

- the basis is orthonormal, i.e. $\varphi_k \cdot \varphi_l = \delta_{k,l}$
- the basis functions for $N = 16$ are:

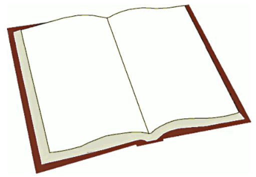
$$\varphi_k(m) = \frac{1}{\sqrt{N}} e^{\frac{-2\pi imk}{N}} = \frac{1}{\sqrt{N}} \left(\cos \frac{2\pi mk}{N} - i \sin \frac{2\pi mk}{N} \right)$$





Bibliography

- Gonzalez, R. C., Woods, R. E., Digital image processing / 2nd ed., Upper Saddle River: Prentice Hall, 2002, pages 793, ISBN 0201180758
- Bracewell, R. N., Fourier transform and its applications / 2nd ed. New York: McGraw-Hill, pages 474, ISBN 0070070156



You should know the answers . . .

- What happens if we convolve a function f with Gaussian located outside the origin?
- What is the result when convolving a function f with several δ impulses?
- Under which conditions is the convolution kernel separable?
- What is the *basis* and *vector space* generated by the given basis?
- What are the orthogonal vectors?
- What is the orthonormal basis?
- How can we simply convert a vector from one basis to another basis?
- What is the unitary/orthogonal matrix?
- What is the difference between *basis vector* and *basis function*?