4. Discrete Fourier Transform

1. The family of Fourier transforms
2. DFT for real-valued signals
3. Fourier transform properties
4. Applications of the DFT

Textbook: Smith, ch. 8, 9, 10

1. The family of Fourier transforms

1) Introduction

- **Fourier analysis** is a family of mathematical techniques, all based on decomposing signals into **sinusoids**.
- The **discrete Fourier transform (DFT)** is the family member used with *digitized* signals.
- **Real DFT** is a version of the discrete Fourier transform that uses real numbers to represent the input and output signals.
- The **complex DFT** uses complex numbers.

In 1807 **Jean Baptiste Joseph Fourier** (1768-1830) presented a paper that contained the controversial claim that **any continuous periodic signal** could be represented as **the sum** of properly chosen **sinusoidal waves**.

Why sinusoids ?

The goal of decomposition is to end up with something ***easier*** to deal with than the original signal.

For example:

- Impulse decomposition allows signals to be examined one point at a time, leading to the powerful technique of convolution.

The component sine and cosine waves are *simpler* than the original signal because they have a property of ***sinusoidal fidelity*** (sinusoids are the only waveform that have this useful property).

2) The Family of Fourier transforms

A signal can be either ***continuous*** or ***discrete***, and it can be either ***periodic*** or ***aperiodic*** --> **four categories** of the general term ***Fourier transform***:

1. Aperiodic-Continuous

A signal extends to both positive and negative infinity *without* repeating in a *periodic* pattern (e.g. Gaussian curve) → the **Fourier Transform**.

2. Periodic-Continuous

E.g. sine waves, square waves, and any waveform that repeats itself in a *regular* pattern from negative to positive infinity → **Fourier Series**.

3. Aperiodic-Discrete

A signal is defined at discrete points between positive and negative infinity, and is not periodic → **Discrete Time Fourier Transform.**

4. Periodic-Discrete

These are discrete signals that repeat themselves in a periodic fashion from negative to positive infinity → **Discrete Fourier Transform.**

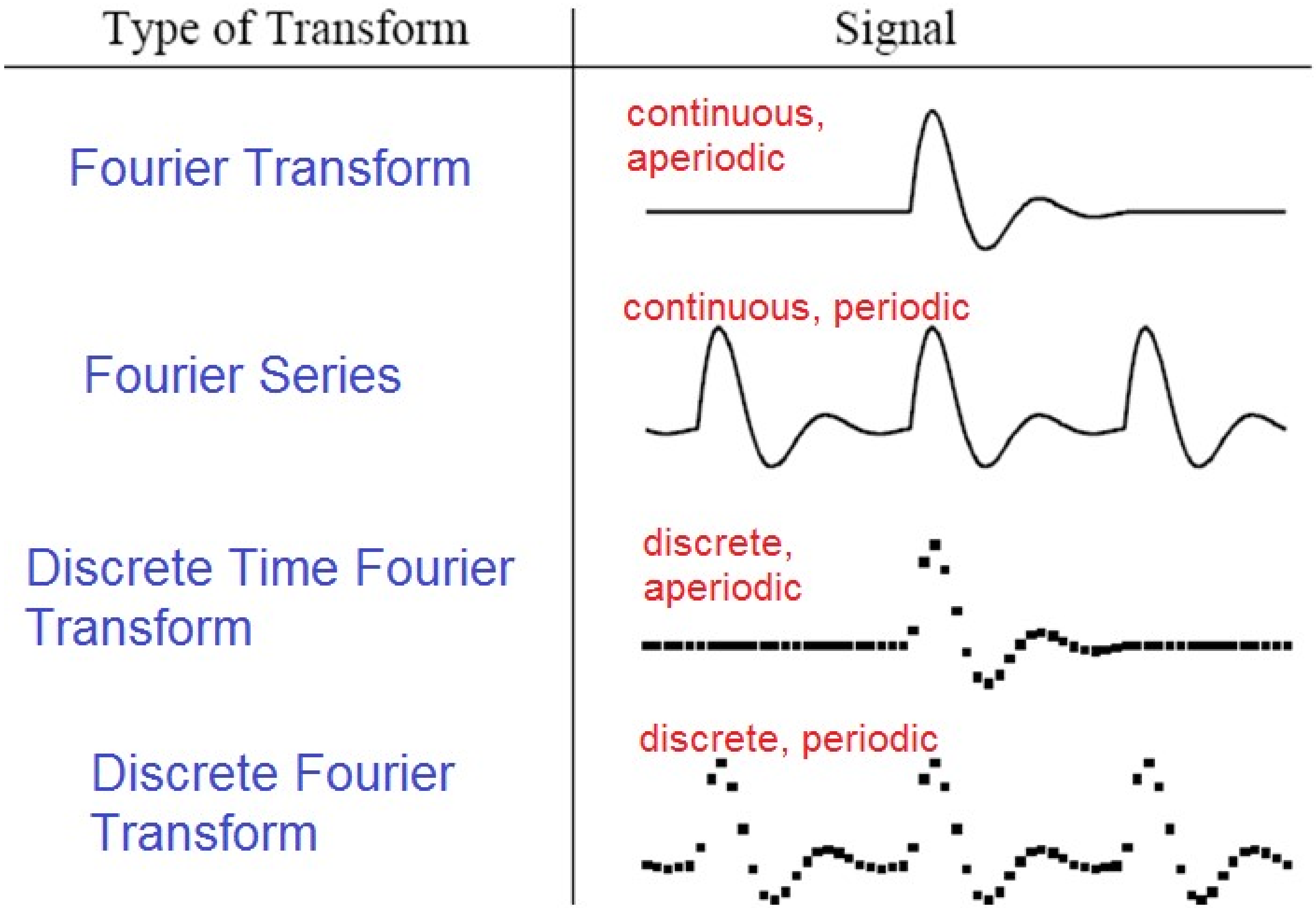


Fig. 1. The four Fourier transforms. A signal may be **continuous** or **discrete**, and it may be **periodic** or **aperiodic**.

3) Finite signals

All these 4 classes of signals extend to **positive** and **negative** ***infinity*.**

- But only a ***finite*** **number** of samples can be stored in a computer.

The **sine** and **cosine** waves are **infinitely** long.

One **cannot** use a group of ***infinitely long*** signals to **synthesize** a ***finite-length*** signal.

Idea: make the **finite** data ***look like*** an **infinite** length signal:

- Imagine that the signal has an infinite number of samples on the left and right of the actual points:
- If all these "imagined" samples have a **value of zero**, the signal looks ***discrete*** and ***aperiodic*** → **Discrete Time Fourier Transform** applies.

- Let the imagined samples be a **duplication** of the actual **N points**. In this case, the signal looks ***discrete*** and ***periodic***, with a period of N samples → use the **Discrete Fourier Transform**.

Observe that digital computers can only work with information that is ***discrete*** and ***finite*** in length.

- An ***infinite*** **number of sinusoids** are required to synthesize a signal that is ***aperiodic***.

This makes it **impossible** to calculate the **Discrete Time Fourier Transform** in a computer algorithm.

- The **only type** of Fourier transform that can be used in DSP is the **DFT**.

4) Real vs. complex numbers

Each of the four Fourier Transforms can be subdivided into:

- **real** and **complex** versions.

1. The real version (for example **real DFT**) is using ordinary numbers and algebra for the synthesis and decomposition.

2. The **complex** versions of the four **Fourier transforms** require the use of ***complex numbers***.

5) Transform

A function is an algorithm or procedure that changes one value into another value.

Transforms are a direct extension of this, allowing both the input and output to have ***multiple*** **values**.

2. DFT for real valued signals

The **discrete Fourier transform** changes an ***N* point** real-valued input signal into **two (*N*/2)** + 1 point output signals:

- The input signal is said to be in the **time domain** – because typically the signal for DFT consists of samples taken at regular intervals of *time.*
- The term **frequency domain** is used to describe the amplitudes of the sine and cosine waves.

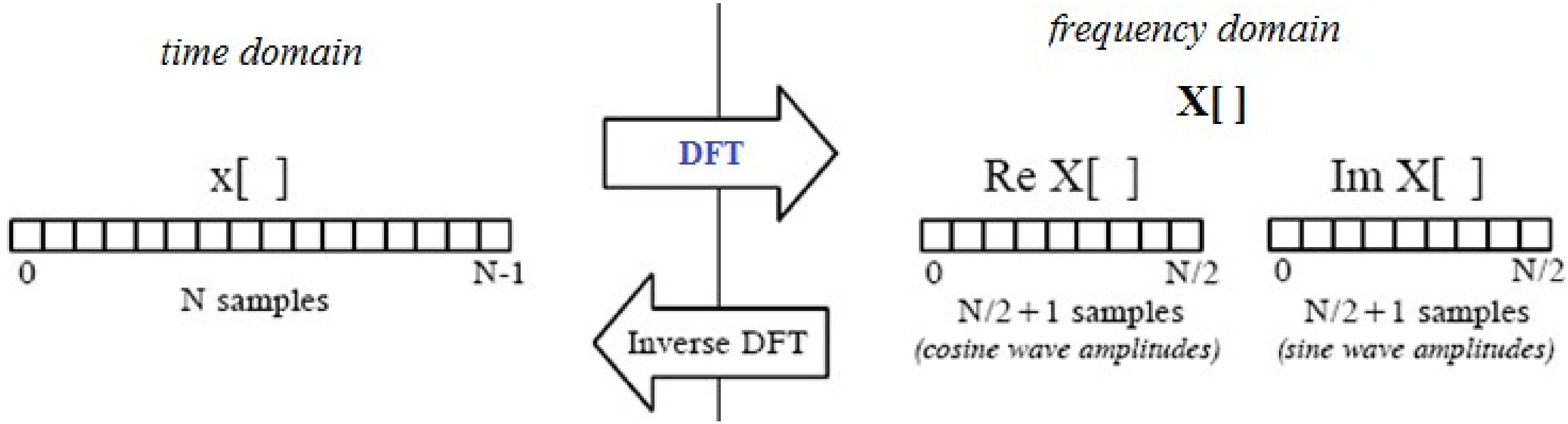


Fig. 2. DFT for real-valued signals

1) Terminology

- Given the time domain signal, the process of calculating the frequency domain is called **decomposition**, **analysis**, or simply **the DFT**.
- Knowing the frequency domain, calculation of the time domain is called **synthesis** or the **inverse DFT**.

<u>Notation</u>

$x[\ ]$ - an N point time domain signal,

$X[\ ]$ – the frequency domain of this signal, which consists of two parts, arrays of $(N/2)+1$ samples:

1. the **real part of** $\boldsymbol{X}[\ ]$ ($\boldsymbol{ReX}[\ ]$),
   $ReX[0], \ldots, ReX[N/2]$ - amplitudes of the cosine waves,

2. the **imaginary part of** $X[\ ]$ ($\boldsymbol{ImX}[\ ]$),
   $ImX[0], \ldots, ImX[N/2]$ - amplitudes of the sine waves.

Remark

The statement:

"The DFT changes an N point time domain signal into an N point frequency domain signal."

refers to the **DFT** when each "point" is a **complex** number.

2) The Frequency Domain Variables

The **horizontal axis** in frequency domain can be referred to in **4 ways**.

1. An **index** from $k=0$ to $k=N/2$;
2. A **fraction of the sampling rate** f - a value between 0 and 0.5; (f stands for **frequency**) , $f = k/N$.
3. A **frequency** f **multiplied by 2π**, denoted by ω.
   ω is called the **natural frequency**, and has the units of **radians**.
4. The **analog frequencies** used in a *particular* application (the frequency data has a *real world* meaning).

Example

A cosine wave is written:

1. using index k: $c_k(t) = \cos(2\pi k\, t / N)$,
2. using relative frequency f : $c_k(t) = \cos(2\pi f\, t)$,
3. using natural frequency ω: $c_k(t) = \cos(\omega\, t)$.

3) DFT Basis Functions (Fourier decomposition)

DFT basis functions- a set of **sine** and **cosine waves** with ***unity*** amplitude:

$$c_k(t) = \cos(\ 2\pi k\ t\ /N)$$
$$s_k(t) = \sin(\ 2\pi k\ t\ /N)$$

where the parameters, k, and N, determine the **frequency** of the wave.
In an N point DFT, $k \in [0,1,\ldots, N/2]$.

The **frequency parameter**, ***k***, is equal to the **number of complete cycles** that occur over the N points of the signal.

The result of the DFT is a **set of numbers** that represent **amplitudes**.

- In DFT the base functions are **sampled**; points of the base functions with index k that contribute to the computation of the k-th output are:

$$c_k[n] = \cos(\ 2\pi k\ n\ /N)$$
$$s_k[n] = \sin(\ 2\pi k\ n\ /N)$$

where $n \in [0,1,\ldots,N\text{-}1]$

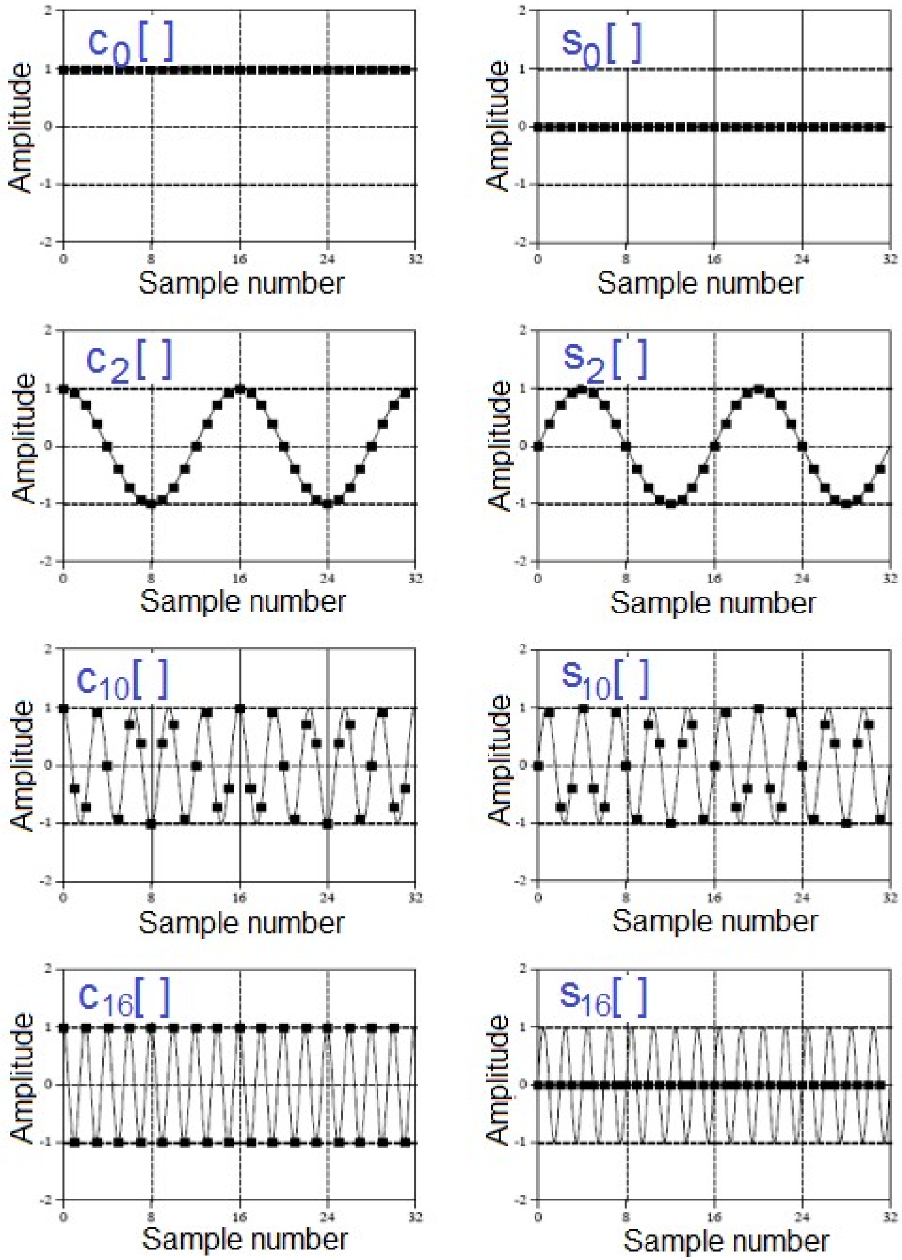
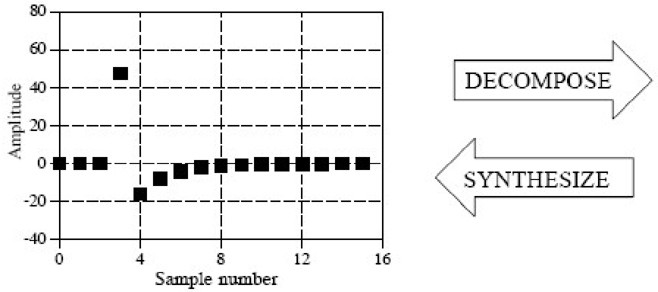
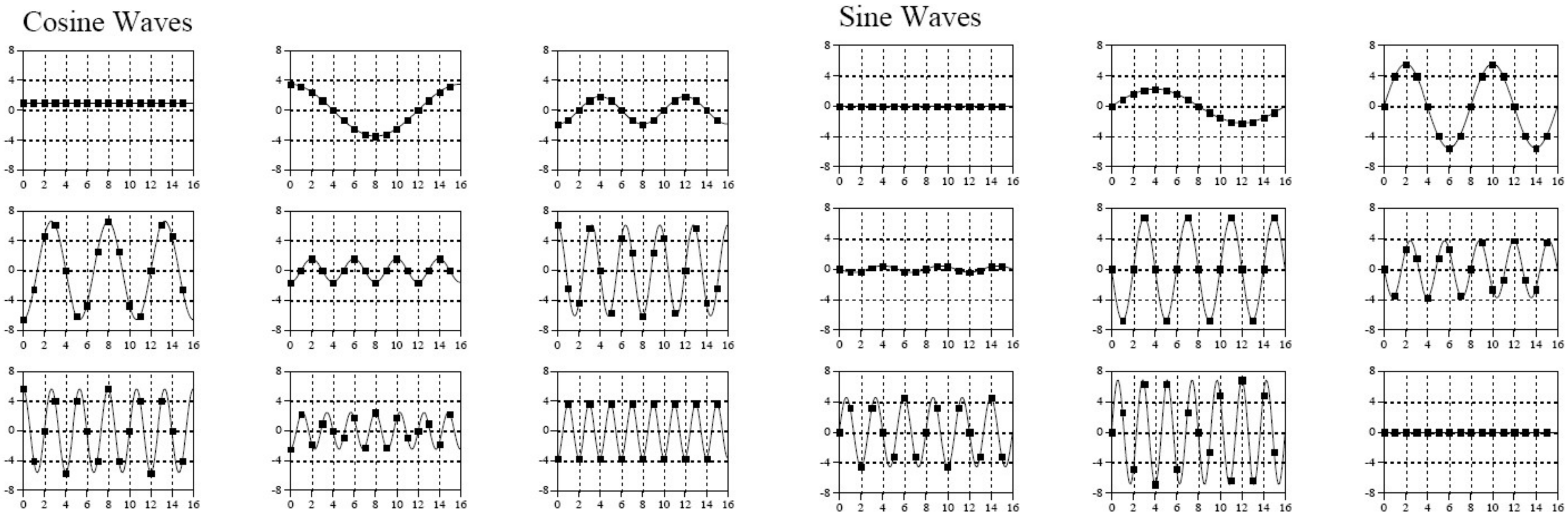


Fig. 3.
DFT basis functions. A 32 point DFT has 17 discrete cosine waves and 17 discrete sine waves for its basis functions: e.g. basis functions with indices 0, 2, 10 and 16.



(a) The signal to be decomposed.



(b) Cosine and sine waves

Fig. 4 Example of Fourier decomposition: (a) a 16 point signal is decomposed into (b) 9 cosine waves and 9 sine waves. The frequency of each sinusoid is fixed; only the amplitude is changed depending on the shape of the signal.

Remarks

1. The cosine wave c_0 is of zero frequency, and is a constant value of **one**. $ReX[0]$ is the average value of the time domain signal (the **DC offset**).
2. The sine wave of zero frequency, s_0, is composed of all ***zeros***. $ImX[0]$ is always set to **zero**.
3. The **highest** frequency basis functions are $c_{N/2}$ and $s_{N/2}$. The discrete cosine wave alternates in value **between 1 and -1**, i.e. a sampling of a continuous sinusoid at the *peaks*. The discrete sine wave contains all **zeros**, resulting from sampling at the *zero crossings*. $ImX[N/2] = ImX[0] = 0$.

Observe: there are N samples entering the DFT, and N+2 samples exiting, but two of the output samples contain *no* information ($ImX[0]=ImX[N/2]=0$).

4) Calculating the DFT

- **A correlation** *based* DFT if the DFT has less than 32 points;
- **A FFT** (Fast Fourier Transform) is preferred otherwise.

DFT by Correlation

To detect a **known waveform contained in another signal**, **multiply** the two signals and **add** all the points in the resulting signal → a measure of **how similar** the two signals are.

This procedure is formalized in the **DFT** (the **analysis equation)**:

$$\operatorname{Re} X[k] = \sum_{t=0}^{N-1} x[t] \cdot \cos(2\pi\, k\, t / N)$$

$$\operatorname{Im} X[k] = -\sum_{t=0}^{N-1} x[t] \cdot \sin(2\pi\, k\, t / N)$$

Each **sample** in the frequency domain is found by **multiplying** the time domain signal by the sine or cosine wave being looked for, and **adding** the resulting points.

Orthogonal base functions

Orthogonal property of the base functions:

each of them must be completely ***uncorrelated*** with all of the others; i.e. if any two of the base functions are multiplied, the sum of the resulting points will be equal to zero.

Example

Figure 5 illustrates the use of **correlation** to calculate $ImX[3]$:

- The signal, $x1[\ ]$, is composed of a sine wave that makes three cycles between points 0 and 63.
- $x2[\ ]$ is composed of several sine and cosine waves, *none* of which make three cycles between points 0 and 63.
- For $x1[\ ]$, the algorithm produces a value of 32, the amplitude of the sine wave present in the signal (modified by the scaling factors).
- For the other signal, $x2[\ ]$, a value of zero is produced, indicating that this particular sine wave is not present in this signal.

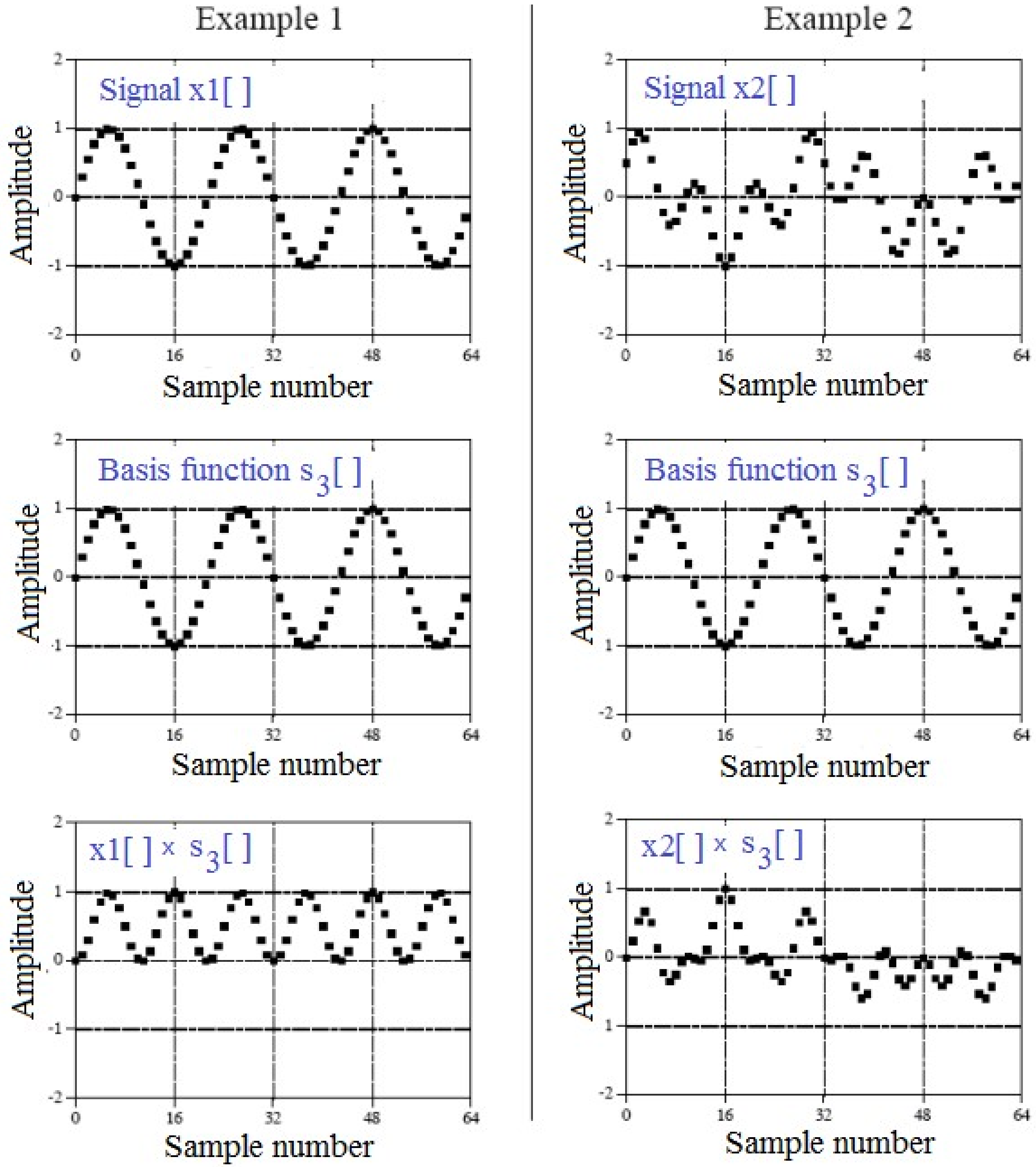


Fig. 5 Two example signals, x1[] and x2[], are multiplied by the specific basis function $s_3[]$:

x1[] × $s_3[]$ has an average of 0.5 (and sum= 32), indicating that *x*1[] contains the basis function with an amplitude of 1.0.

x2[] × $s_3[]$ has a zero average, indicating that *x*2[] does not contain the basis function.

5) The Inverse DFT (the synthesis equation)

The **synthesis equation**

$$x[i] = \sum_{k=0}^{N/2} \operatorname{Re} \bar{X}[k] \cdot \cos(2\pi ki / N) + \sum_{k=0}^{N/2} \operatorname{Im} \bar{X}[k] \cdot \sin(2\pi ki / N) \qquad \text{(I 1)}$$

where:

$x[i]$ is the signal being synthesized, with the index $i \in [0,1,\ldots,N\text{-}1]$,
$Re\bar{X}[k]$ and $Im\bar{X}[k]$ hold the ***normalized*** amplitudes of the cosine and sine waves, respectively, with $k \in [0,1,\ldots, N/2]$.

The ***normalized*** amplitudes are:

$$\operatorname{Re} \bar{X}[k] = \frac{\operatorname{Re} X[k]}{N/2}; \quad \operatorname{Im} \bar{X}[k] = -\frac{\operatorname{Im} X[k]}{N/2} \qquad \text{(I 2)}$$

except for two cases:

$$\operatorname{Re} \bar{X}[0] = \frac{\operatorname{Re} X[0]}{N}; \quad \operatorname{Re} \bar{X}[N/2] = \frac{\operatorname{Re} X[N/2]}{N}$$

Why the negation of the imaginary part?

This is done to make the *real DFT* consistent with the *complex DFT*.

Example (fig. 6)

Why the amplitudes need to **be normalized**?

The difference occurs because the frequency domain is defined as a ***spectral density*** (Figure 7):

- S*pectral* ***density*** describes how much signal (amplitude) is present *per unit of bandwidth*.
- To convert the sinusoidal amplitudes into a spectral **density**, multiply each amplitude by the bandwidth it represents.

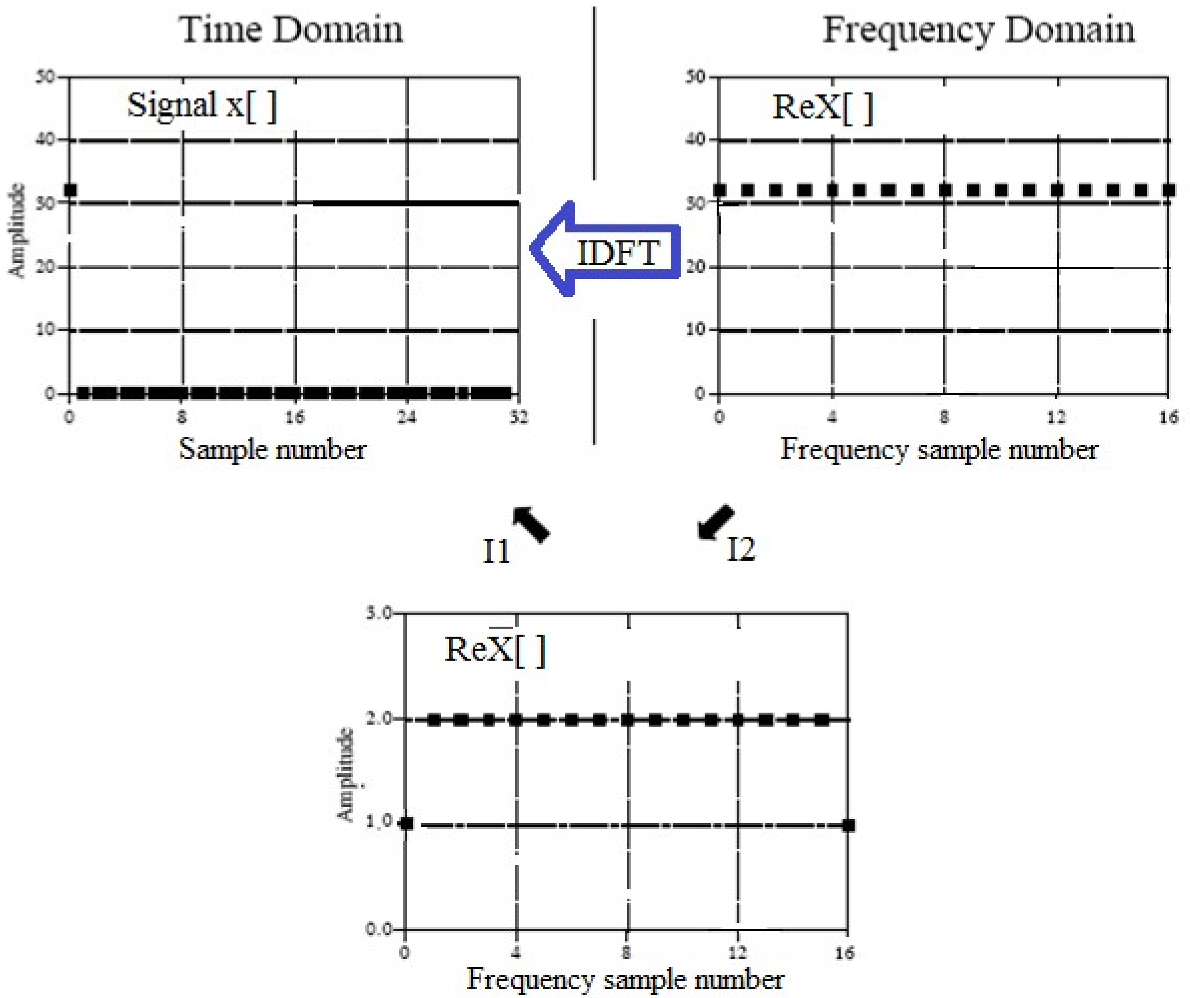


Fig. 6 The Inverse DFT: (a) an impulse at sample zero with an amplitude of 32; (b) the real part of the frequency domain of this signal, a constant value of 32 (the imaginary part is composed of all zeros); (c) the normalized amplitudes of the cosine waves = 2 (2 = 32/16).

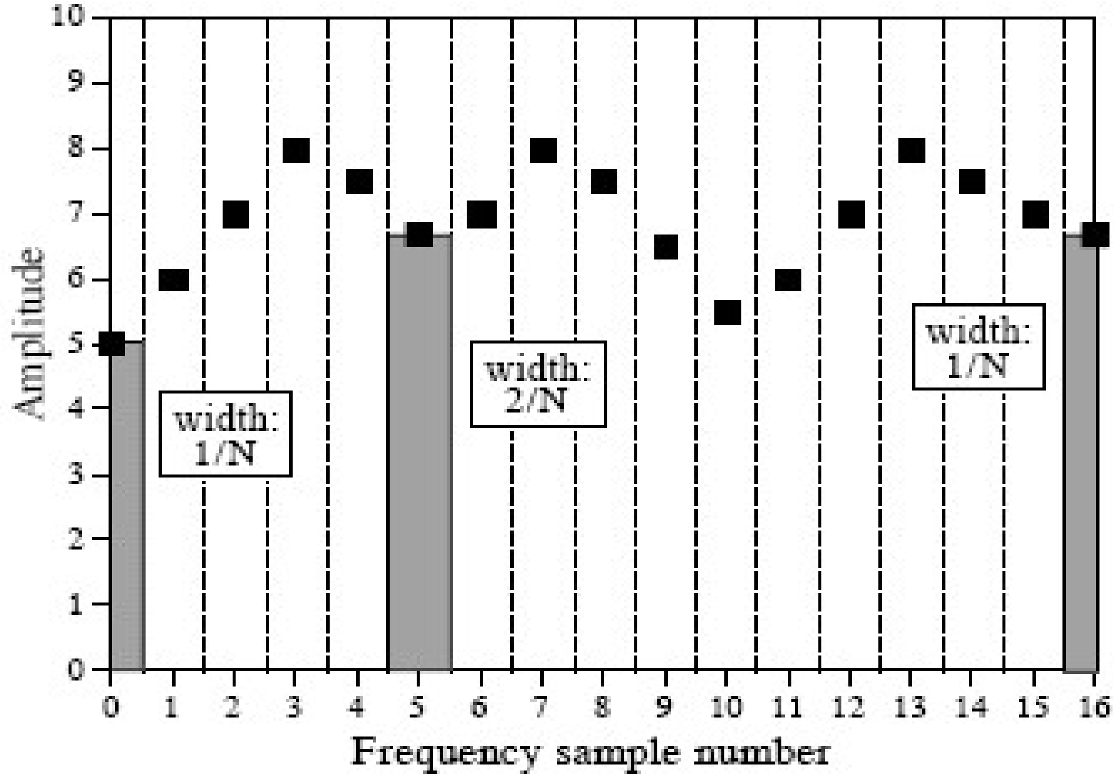


Fig. 7 Each sample in the frequency domain is contained in a frequency band of width $2/N$ (a fraction of the total bandwidth), except of the first and last samples (a bandwidth of $1/N$).

6) Duality

The **synthesis** and **analysis equations** are quite **similar**:

- to move from one domain to the other, the known values are **multiplied** by the basis functions, and the resulting products **added**.

The only **significant difference** between them:

- the result in the time domain is ***one*** signal of N points,
- the frequency domain has *two* signals of $N/2+1$ points.

The ***complex*** **DFT** expresses both the time and the frequency domains as complex signals of N points each:

- This makes the two domains completely symmetrical, and the equations for moving between them nearly *identical*.

The **duality** between the time and frequency domains:

- a point in the frequency domain corresponds to a sinusoid in the time domain.
- a point in the time domain corresponds to a sinusoid in the frequency domain.
- convolution in the time domain corresponds to multiplication in the frequency domain.
- convolution in the frequency domain corresponds to multiplication in the time domain.

7) Polar Notation

The frequency domain, given by a group of amplitudes of cosine and sine waves (*ReX*[ ] and *ImX*[ ])), is called **rectangular** notation.

The frequency domain can be expressed in **polar form** by:

- the **magnitude of** ***X***[ ] (***MagX***[ ]) and
- the **phase of** ***X***[ ] (***PhaseX***[ ]).

When we **add** a cosine wave and a sine wave of the same frequency, the result is a cosine wave of the same frequency, but with a new amplitude and a new phase shift:

$$A \cos(x) + B \sin(x) = M \cos(x + \theta)$$

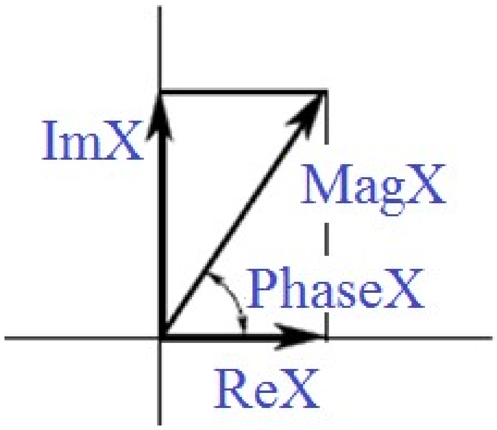


Fig. 8. Rectangular-to-polar conversion. The addition of a cosine wave and a sine wave (of the same frequency) follows the addition of simple vectors.

Rectangular-to-polar conversion:

$$MagX[k] = \sqrt{(\operatorname{Re} X[k])^2 + (\operatorname{Im} X[k])^2}$$

$$PhaseX[k] = \arctan\left(\frac{\operatorname{Im} X[k]}{\operatorname{Re} X[k]}\right)$$

Polar-to-rectangular conversion:

$$\operatorname{Re} X[k] = MagX[k] \cdot \cos(PhaseX[k])$$

$$\operatorname{Im} X[k] = MagX[k] \cdot \sin(PhaseX[k])$$

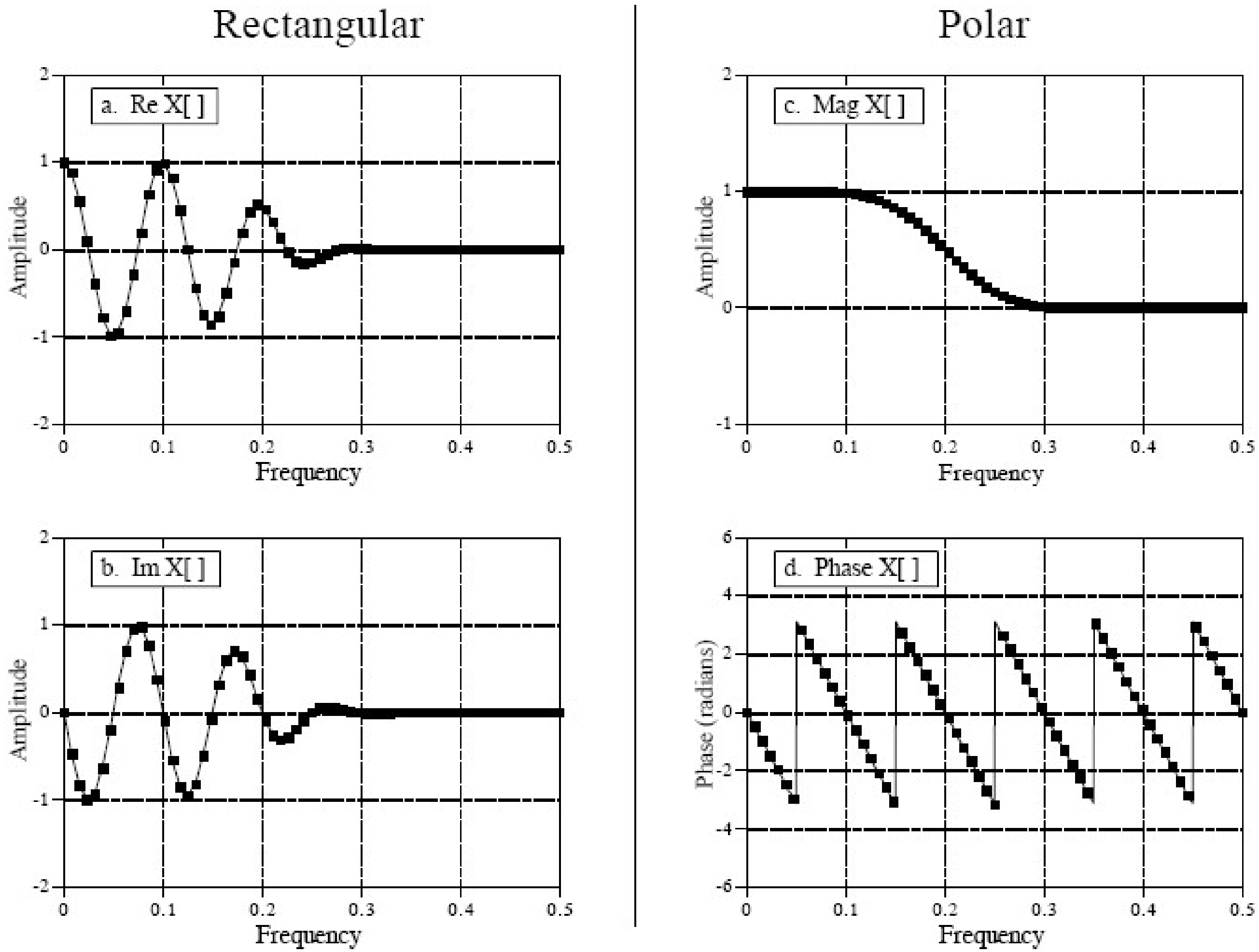


Fig. 9. Rectangular and polar frequency domains. The first and last samples in the phase must be zero, just as they are in the imaginary part. The polar curves are: only frequencies below about 0.25 are present, and the phase shift is linear.

Why is **it easier to understand** the frequency domain in **polar notation**?

- Recall the property of *sinusoidal fidelity*: if a sinusoid enters a linear system, the output will also be a sinusoid, and at exactly the same frequency as the input. Only the amplitude and phase can change.

- Polar notation directly represents signals in terms of the amplitude and phase of the component cosine waves. In turn, systems can be represented by how they modify the amplitude and phase of each of these cosine waves.

8) Magnitude- and phase-coded information

Polar notation usually gives a better understanding of the **characteristics of the signal**.
The phase contains information on the location of events in the time domain signal.

This is because an edge is formed when many sinusoids *rise* at the same location, possible only when their *phases* are coordinated.

Signals, like audio or speech, have their information encoded in the frequency domain.
The magnitude is most important for these signals, with the phase playing only a minor role.

Figure 10 demonstrates what information is contained in the *phase*, and what information is contained in the *magnitude*.

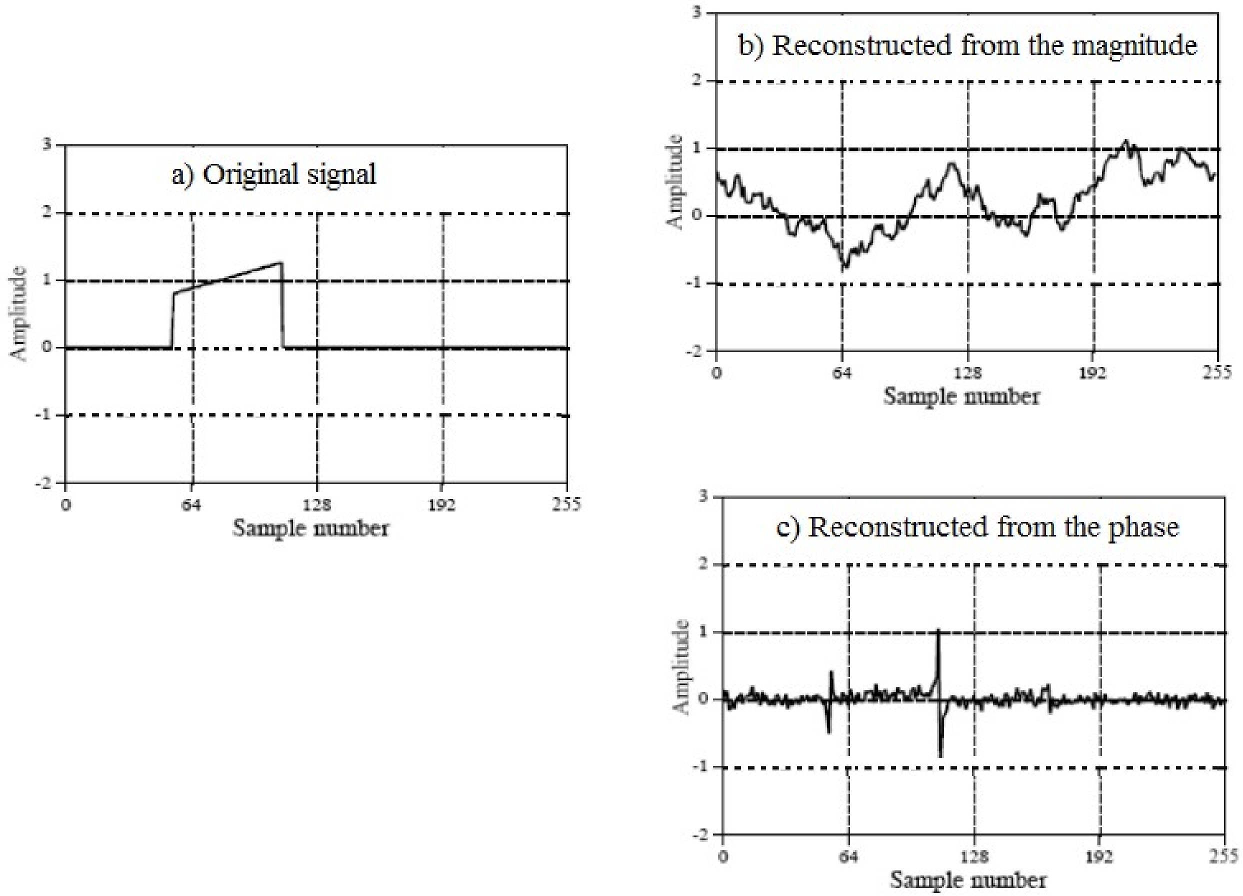


Fig. 10 Figure (a) is a pulse-like signal. The signal in (b) is created by taking the DFT of (a), replacing the *phase* with random numbers, and taking the Inverse DFT. The signal in (c) is found by taking the DFT of (a), replacing the *magnitude* with random numbers, and taking the Inverse DFT. The location of the *edges* is retained in (c), but not in (b).

9) Mathematics of frequency domain signals

The rectangular form is almost always used when mathematical computations of **summation and subtraction** are required, while for **multiplication and division** the polar form is preferred.

Multiplication of frequency domain signals:

$$Y[f] = X[f] \cdot H[f].$$

- In polar form,
  the magnitudes are multiplied: $MagY[f] = MagX[f] \cdot MagH[f]$,
  and the phases are added: $PhaseY[f] = PhaseX[f] + PhaseH[f]$.

- In rectangular form,

$$ReY[f] = ReX[f]\, ReH[f] - ImX[f]\, ImH[f]$$

$$ImY[f] = ImX[f]\, ReH[f] + ReX[f]\, ImH[f]$$

Division of one frequency domain signal by another:

$$H[f] = Y[f] \,/\, X[f] \,,$$

- In polar form, divide the magnitudes and subtract the phases, i.e.,

$$MagH[f] = MagY[f] \,/\, MagX[f],$$
$$PhaseH[f] = PhaseY[f] - PhaseX[f].$$

- In rectangular form,

$$\operatorname{Re} H[f] = \frac{\operatorname{Re} Y[f] \cdot \operatorname{Re} X[f] + \operatorname{Im} Y[f] \cdot \operatorname{Im} X[f]}{(\operatorname{Re} X[f])^2 + (\operatorname{Im} X[f])^2}$$

$$\operatorname{Im} H[f] = \frac{\operatorname{Im} Y[f] \cdot \operatorname{Re} X[f] - \operatorname{Re} Y[f] \cdot \operatorname{Im} X[f]}{(\operatorname{Re} X[f])^2 + (\operatorname{Im} X[f])^2}$$

3. Fourier transform properties

The Fourier transform is linear but ***not*** shift invariant, i.e. a shift in the time domain ***does not*** correspond to a shift in the frequency domain.

1) Linearity of the Fourier Transform

The **Fourier Transform** has the ***homogeneity*** and ***additivity properties***.

Homogeneity

If $x[\ ]$ and $X[\ ]$ are a **Fourier Transform pair**, then $kx[\ ]$ and $kX[\ ]$ are also a **Fourier Transform pair**, for any constant k.

Remark: If the frequency domain is represented in *polar* notation, $kX[\ ]$ means that the magnitude is **multiplied** by k, while the phase remains **unchanged**.

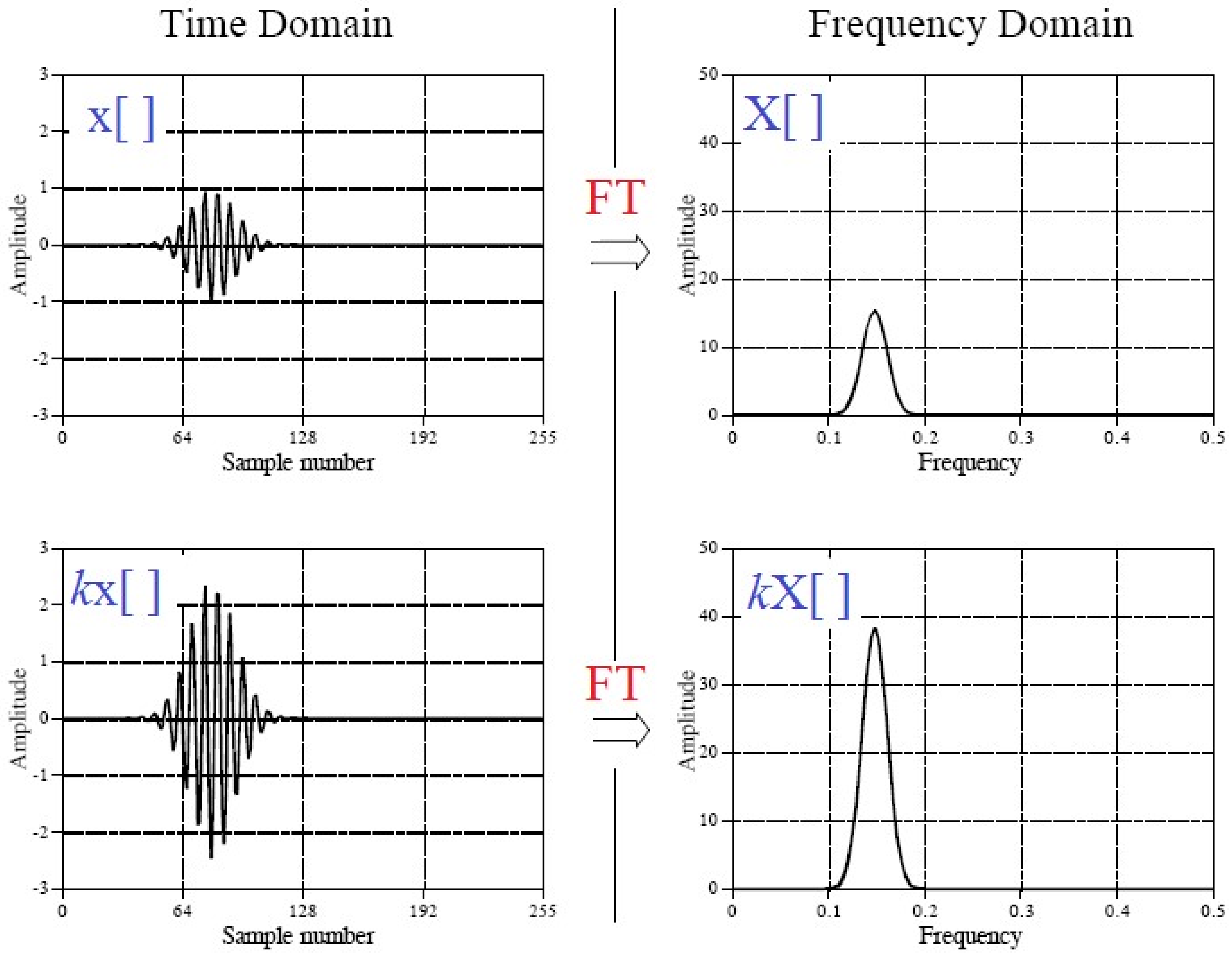


Fig. 11 Homogeneity of the Fourier transform - ***scaling*** in one domain corresponds to ***scaling*** in the other domain.

Additivity

Additivity of the Fourier transform means that ***addition*** in one domain corresponds to ***addition*** in the other domain.

Frequency spectra are added **in rectangular notation** by adding the real parts to the real parts and the imaginary parts to the imaginary parts.

Frequency spectra **in polar form cannot** be directly added:

1. they must be converted into rectangular notation,
2. next added, and
3. then reconverted back to polar form.

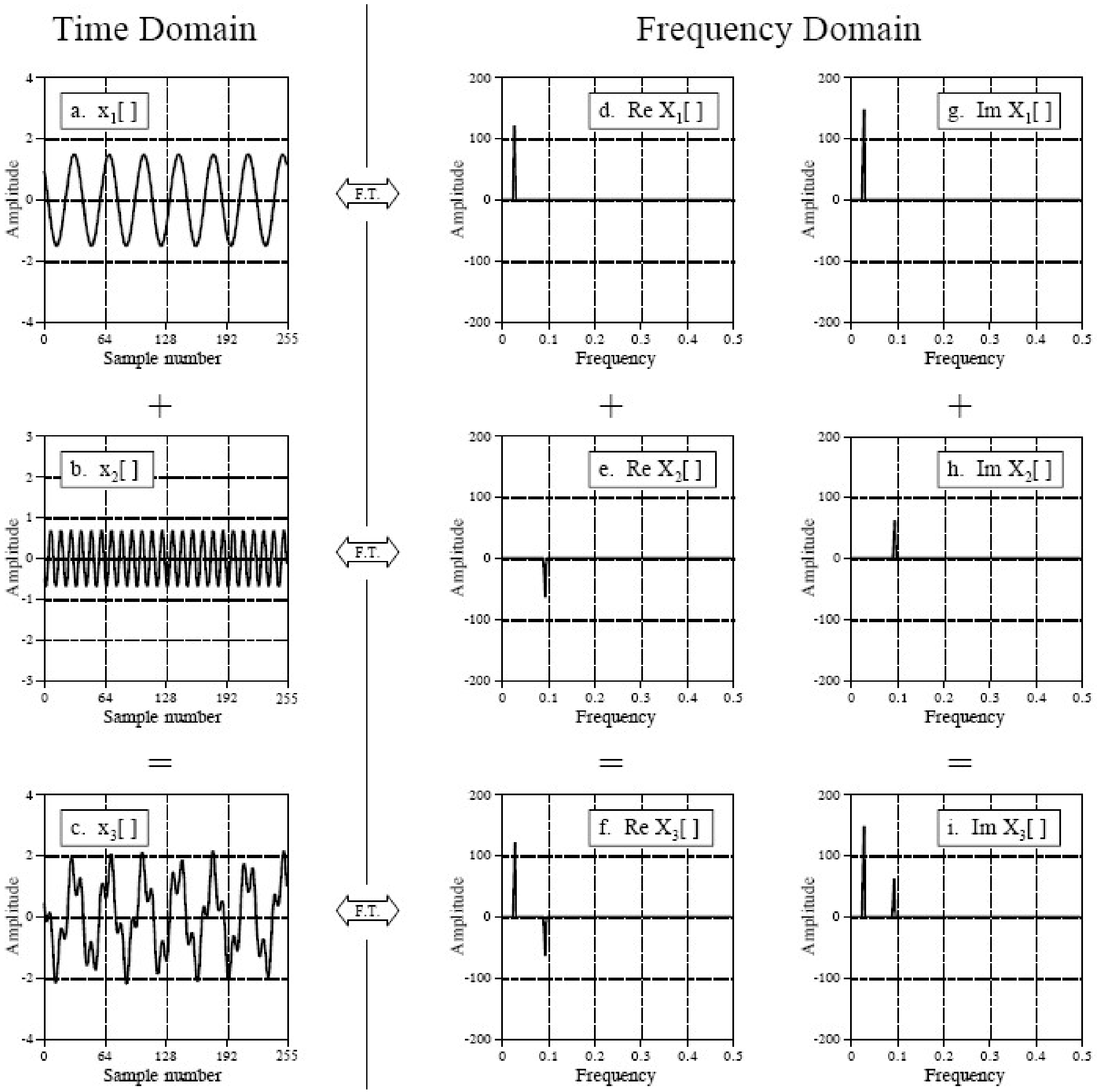


Fig. 12 Additivity of the Fourier transform.
The time domain signals in (a) and (b) are added to produce the signal in (c). This results in the corresponding real (d, e) and imaginary parts (g, h) of the frequency spectra being added (f, i).

2) Shift operation

If

$$x[n] \quad \leftrightarrow \quad (\, Mag\ X[f]\,,\ Phase\ X[f]\,),$$

then

a shift in the time domain results in:

$$x[n+s] \quad \leftrightarrow \quad (\, Mag\ X[f]\,,\ Phase\ X[f] + 2\pi\, s\, f\,),$$

an appropriate change in the phase, proportionally to the time shift and frequency (f is expressed as a fraction of the sampling rate).

A time domain signal that is **symmetrical** around a vertical axis has a ***linear*** *phase* - the phase of its frequency spectrum is a ***straight line***.

- When the time domain waveform is **shifted to the right**, the phase remains a straight line, but experiences a ***decrease*** in slope.
- When the time domain is shifted to the left, there is an ***increase* in the slope**.

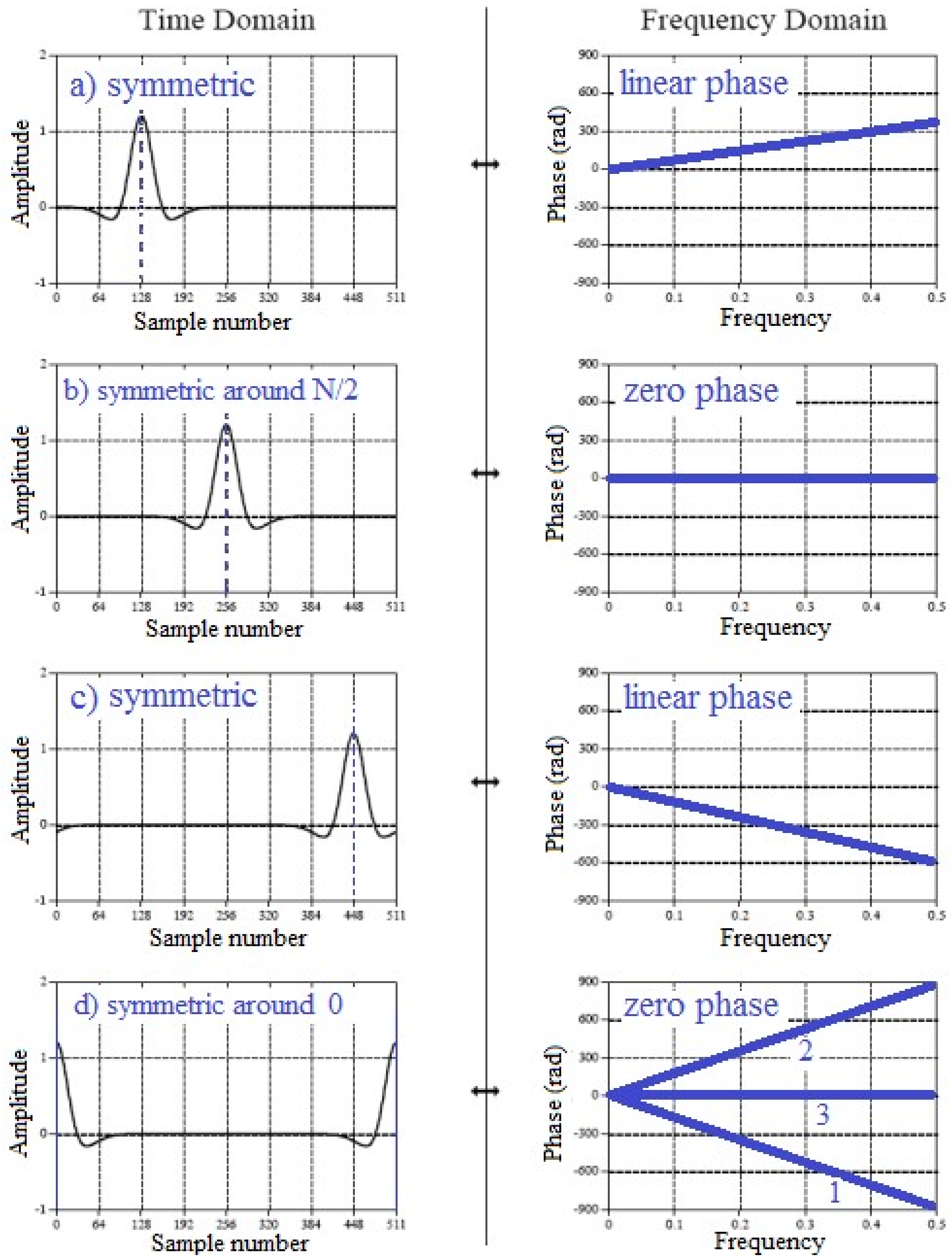


Fig. 13 Phase changes resulting from a time domain shift.
(a, c) The signal is symmetric – the phase is linear.
(b) The phase is entirely zero when the time domain signal is *symmetrical* around sample *zero*. A symmetry around sample *N*/2 is the same as zero.
In (d) all 3 lines are correct. Every sample in line 2 differs from the corresponding sample in line 1 by an integer multiple of 2π. Line 3 is related to lines 1 and 2 by π ambiguities.

When a shift is expressed in terms of a phase change, it becomes *proportional* to the frequency of the sinusoid being shifted.

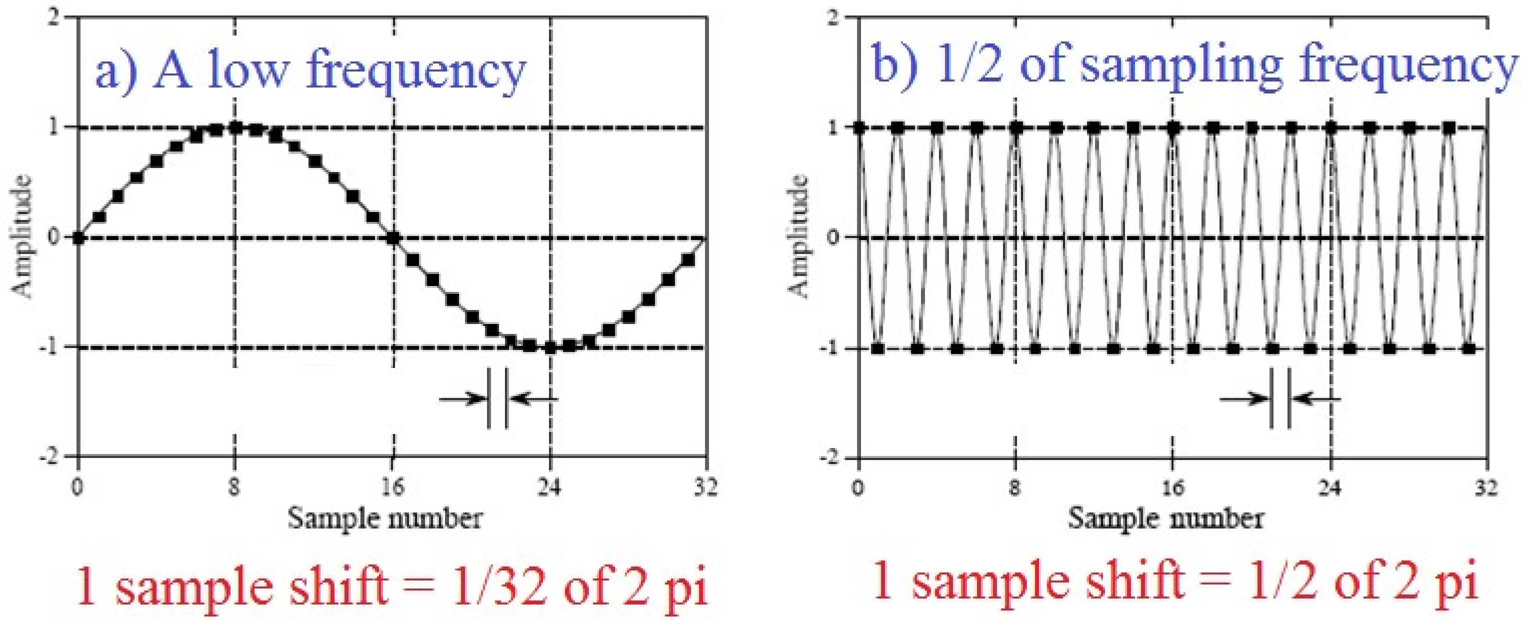


Fig. 14 The relationship between samples and phase: (a) a one sample shift is equal to 1/32 of a cycle; in (b), a one sample shift is equal to 1/2 of a cycle. The waveform changes the phase more at high frequencies than at low frequencies.

At zero frequency there is no phase shift and all of the frequencies between 0 and 0.5 (one-half of the sampling rate) follow in a straight line.

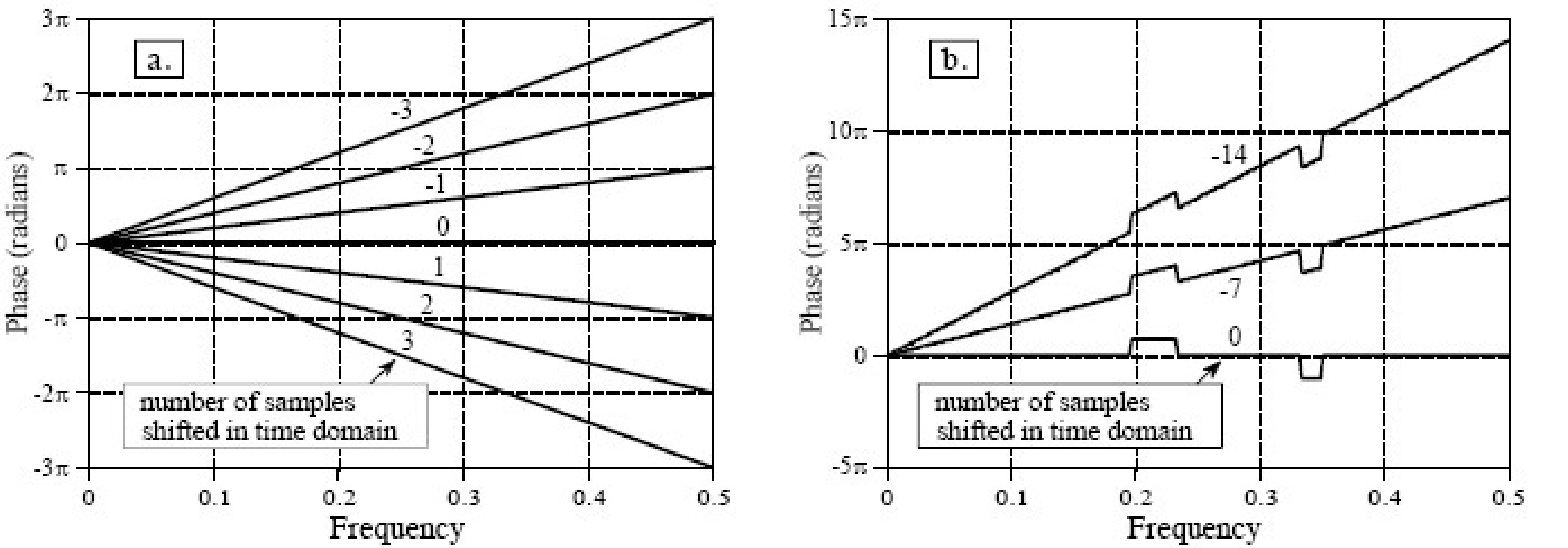


Fig. 15 For each sample that a time domain signal is shifted in the positive direction (i.e., to the right), the phase at frequency 0.5 (half of the sampling frequency) will decrease by π radians. For each sample shifted in the negative direction (i.e., to the left), the phase at frequency 0.5 will increase by π radians. Figure (a) shows this for a linear phase (a straight line), while (b) is an example using a nonlinear phase.

4) Complex conjugation and left-right symmetry

Why does left-right symmetry correspond to a **zero (or linear) phase**?

1. A linear phase appears *because* the nonlinear phase of the left half exactly cancels the nonlinear phase of the right half.

2. A left-right flip of the signal in the time domain does nothing to the magnitude, but changes the sign of every point in the phase.

3. Likewise, changing the sign of the phase flips the time domain signal left-for-right.
   (a) If the signals are continuous, the flip is around zero.
   (b) If the signals are **discrete**, the flip is around **sample zero** *and* **sample** $N/2$, simultaneously.

Figure 16 shows the phase characteristics for a left-right symmetric signal.

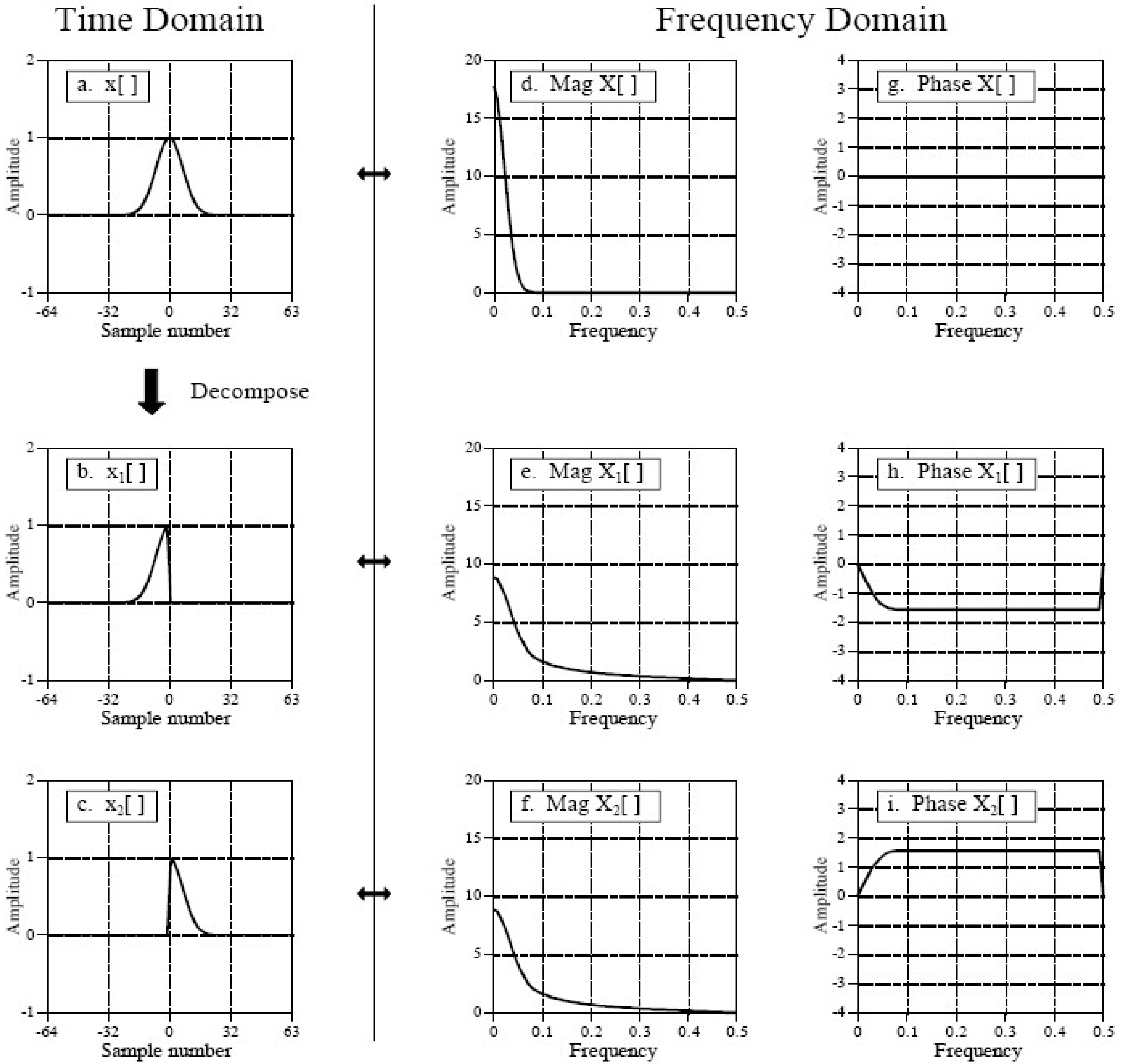


Fig. 16 Phase of left-right symmetry. A symmetric signal (a), can be decomposed into (b) a right half, and (c) a left half. The magnitudes of the two halves (e, f) are identical, while the phases (h, i) are the negative of each other.

Changing the sign of the phase is an operation called **complex conjugation**.

In rectangular notation, the complex conjugate is found by leaving the real part alone, and changing the sign of the imaginary part.

4. Applications of the DFT

1. The DFT can calculate a signal's *frequency spectrum*.
2. The DFT can find a system's frequency response from the system's impulse response, and vice versa.

   This allows systems to be analyzed in the *frequency domain*.
3. The DFT can be used as an intermediate step in signal processing.

   The classic example of this is *FFT convolution*, a fast algorithm for convolving signals.

1) Spectral Analysis of Signals

Many things oscillate in our universe. For example,

- speech is a result of vibration of the human vocal cords;
- ship's propellers generate periodic displacement of the water,

and so on.

1. The *shape* of the time domain signal is not important in these signals.
2. The key information is in **the *frequency*, *phase*** and ***amplitude*** of the component sinusoids.
3. The DFT is used to extract this information.

Frequency spectrum resolution

The *length* of the signal limits the frequency resolution.

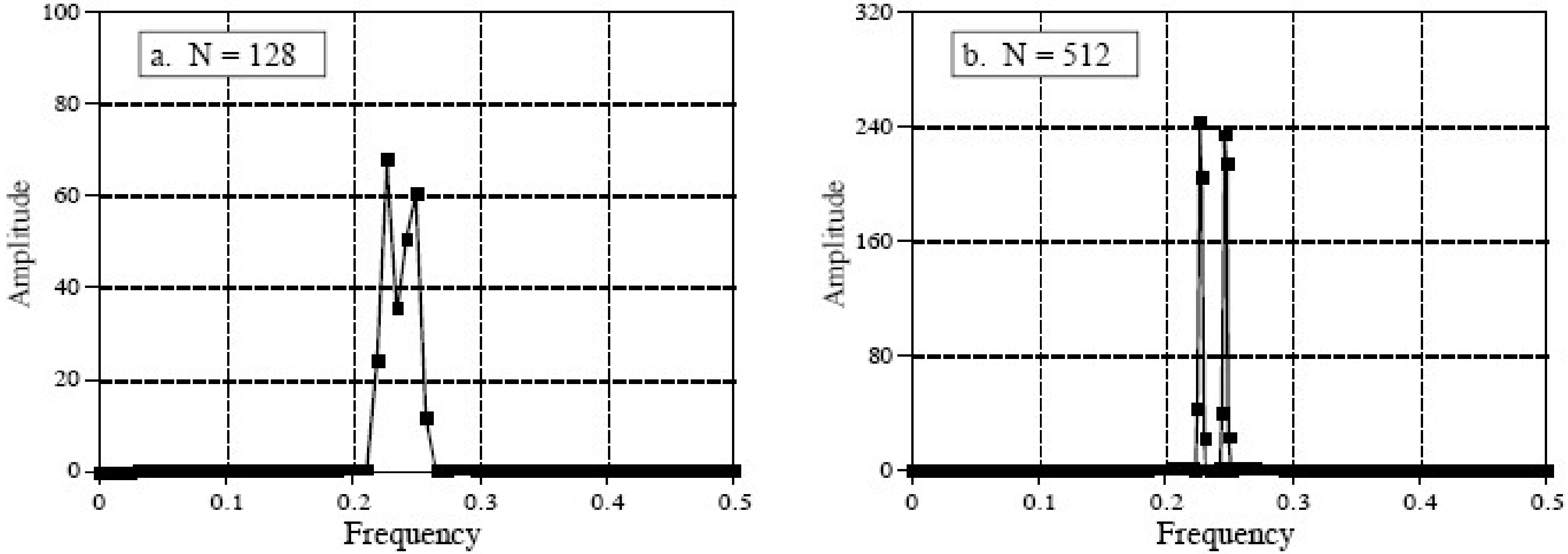


Fig. 17 Frequency spectrum resolution. The longer the DFT, the better the ability to separate closely spaced features. A 128 point DFT cannot resolve the two peaks, while a 512 point DFT can.

What happens if the input signal contains a sinusoid with a frequency *between* two of the basis functions? Since it cannot be represented by a single sample, it becomes a peak with **tails**.

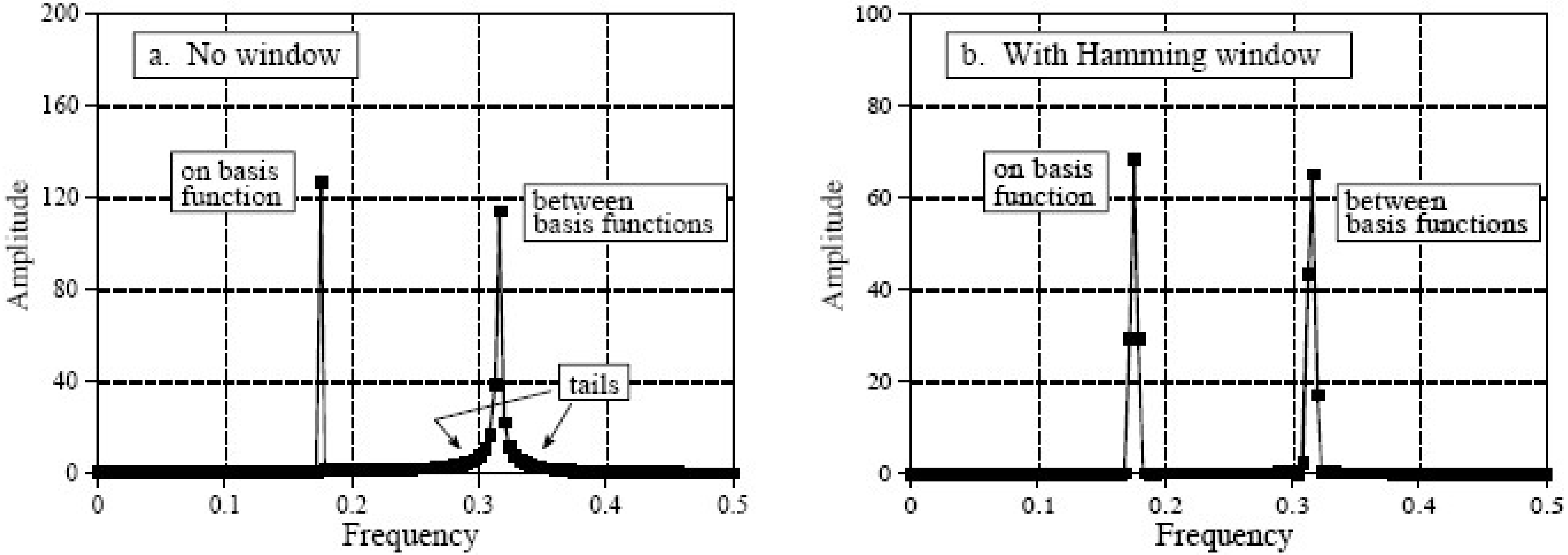


Fig. 18 Using **a window** in spectral analysis: the frequency spectrum of a signal consisting of two sine waves - one sine wave has a frequency exactly equal to a basis function, the other sine wave has a frequency *between* two of the basis functions, resulting in *tails* on the peak. The Hamming window applied before the DFT makes the peaks look the same and reduces the tails, but broadens the peaks.

Example (fig. 19)

256 samples are taken from an undersea microphone.
This signal (a) is multiplied by the Hamming window (b), resulting in the windowed signal in (c).
The frequency spectrum of the windowed signal is found using the DFT (d) (magnitude only).

Averaging 100 of these spectra **reduces the random noise**, resulting in the averaged spectrum (e).

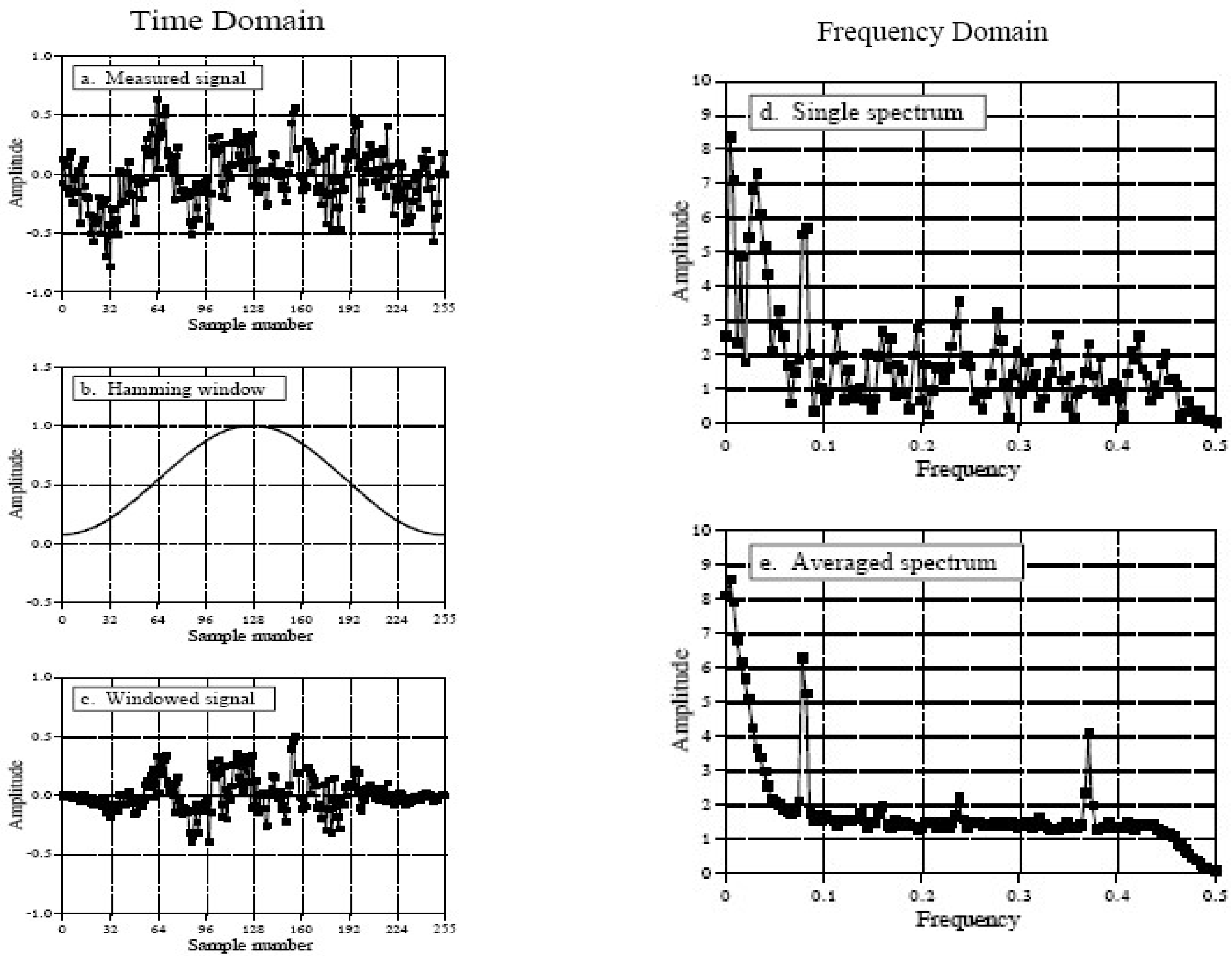


Fig. 19 An example of spectral analysis.

2) Windows

Windows provide a tradeoff between *resolution* (the width of the peak) and *spectral leakage* (the amplitude of the tails).

The use of a **window** changes the signal **in two ways**, both of which **distort** the true frequency spectrum:

Firstly, we *truncate* the signal, by multiplying it by a window.

1. For example, a 256 point *rectangular window* would allow 256 points to retain their correct value, while all the other samples in the infinitely long signal would be set to a value of zero.
2. The *Hamming window* would *shape* the retained samples, besides setting all points outside the window to zero.

The signal is still infinitely long, but with finite number of nonzero values.

Secondly, when two time domain signals are ***multiplied***, the corresponding frequency domains are ***convolved***.

- For example, if the spectrum of the signal is an infinitesimally narrow peak (i.e., a delta function), the spectrum of the windowed signal is the spectrum of the window shifted to the location of the peak.
- By changing the window shape, the amplitude of the side lobes can be reduced at the expense of making the main lobe wider.

Examples of windows

Figure 20 shows how the spectral peak of given signal would appear using 4 different windows:

1. The **rectangular window**, (a), has the narrowest main lobe but the largest amplitude side lobes.
2. The **Hamming window**, (b), and
3. the **Blackman window**, (c), have lower amplitude side lobes at the expense of a wider main lobe.
4. The **flat-top window**, (d), is used when the amplitude of a peak must be accurately measured. The shape for a **flat-top window** is *exactly the same* shape as the filter kernel of a **low-pass** filter.

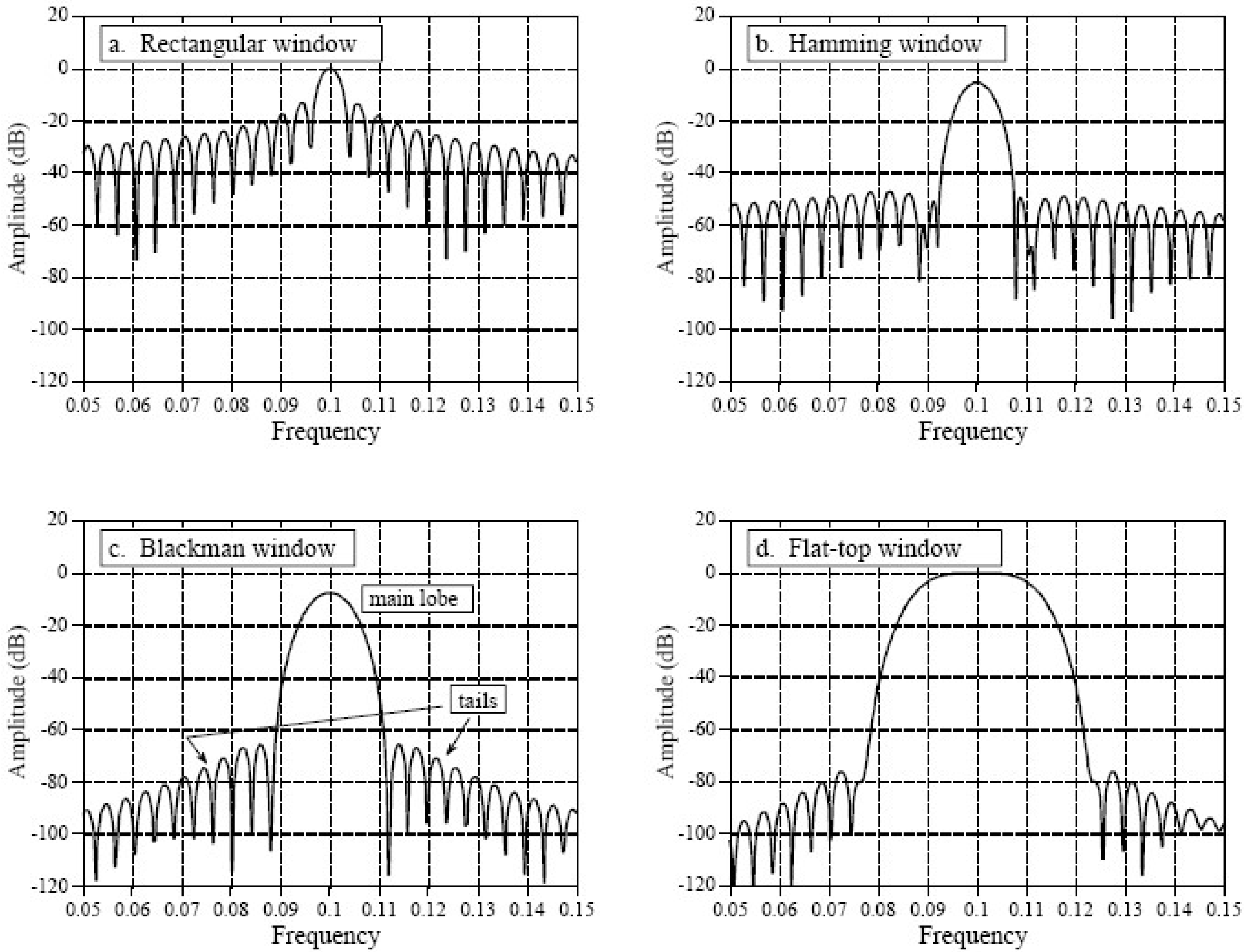


Fig. 20 Windows in the frequency domain. Each peak in the frequency spectrum of a windowed signal is a central lobe surrounded by tails formed from side lobes.

3) Frequency Response of Systems

Systems are analyzed in the *time domain* by using convolution. A similar analysis can be done in the *frequency domain* using the Fourier transform.

Any linear system can be *completely* described by how it changes the amplitude and phase of cosine waves passing through it. This information is called the system's **frequency response**.

Since both the **impulse response** and the **frequency response** contain complete information about the system, there must be **a one to-one correspondence** between the two.

A system's **frequency response** is the Fourier Transform of its **impulse response**.

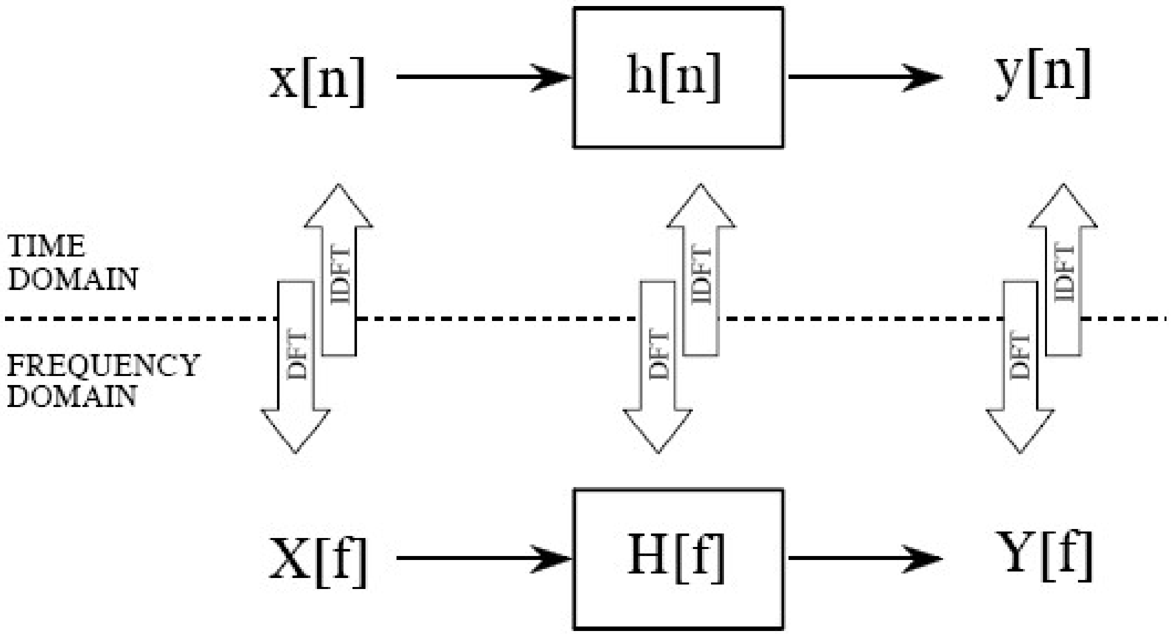


Fig. 21 The DFT and the Inverse DFT **relate the signals** (and **system responses**) in the two domains.

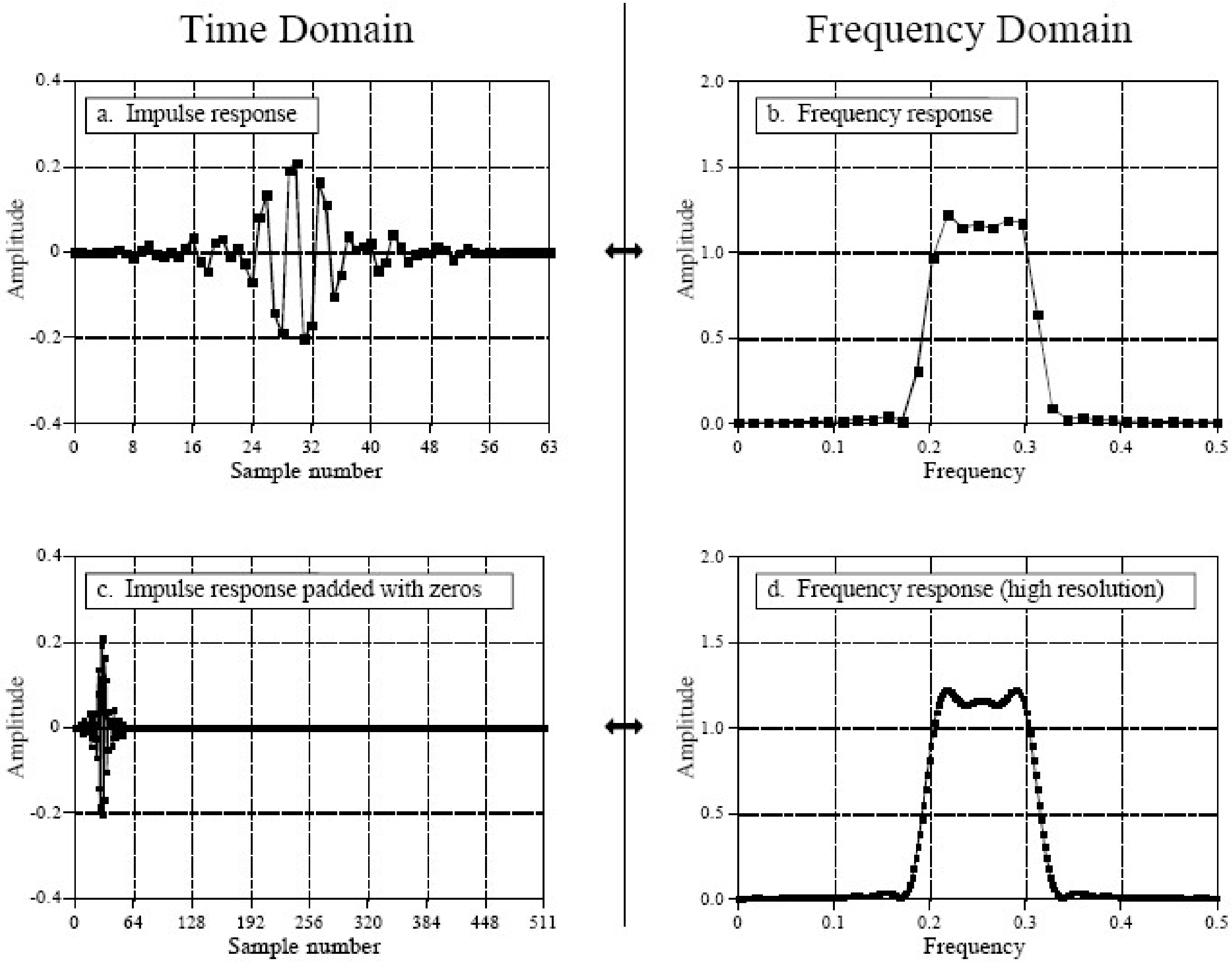


Fig. 22 Finding the frequency response from the impulse response. A system's impulse response (a) is transformed by DFT into the system's frequency response (b). By padding the impulse response with zeros (c), higher resolution can be obtained in the frequency response (d).

How much resolution can be obtained in the frequency response?

The answer is: ***infinitely* high**, if you pad the impulse response with an *infinite* number of zeros.

There is **nothing limiting the frequency resolution** except the length of the DFT.

Even though the impulse response is a *discrete* signal, the corresponding **frequency response** is ***continuous***.

An N point DFT of the impulse response provides $N/2+1$ *samples* of this continuous curve.

The DFT is used to calculate a ***sampling* of the true frequency response**.

4) Convolution via the Frequency Domain

Given **an input signal** and **impulse response**, we need to find the resulting output signal and do not want to do convolution:

1. Transform the two signals into the frequency domain,
2. Multiply them, and then
3. Transform the result back into the time domain.

This replaces one convolution with two DFTs, a multiplication, and an Inverse DFT.

The output of above algorithm is *identical* to the standard convolution algorithm.

Convolution is avoided for two reasons:

1. Convolution is *mathematically* difficult to deal with. Suppose you are given a system's impulse response, and its output signal.
   How to calculate the input signal, i.e. to make **deconvolution**?
   However, in the frequency domain deconvolution can be carried out as a simple *division*, the inverse of multiplication.

2. Convolution is *computationally slow*, because of the large number of multiplications and additions that must be calculated.
   To calculate the DFTs, the *Fast Fourier Transform* (FFT) is available.

Exercises 4

Exercise 4.1

Illustrate the DFT for real-valued signals and its inverse transform (analogical to the Fourier decomposition and synthesis schemas) for the following signal:

n	0	1	2	3	4	5	6	7
$\boldsymbol{x}[n]$	-4	-2	4	10	10	4	-2	-4

Exercise 4.2 Using the Fourier series compute the frequency-based decomposition of the following periodic function:

$$f(t) = \begin{cases} 1, & if\, 0 \le t < \pi/2 \\ -1, & if\ \pi/2 < t < 3\pi/2 \\ 1, & if\ 3\pi/2 < t \le 2\pi \end{cases}$$