Time and Spatial Series

- Data and Transform domains
- *Z*- and Fourier Transforms

- <u>Reading:</u>
  - Shearer, A5
  - Telford *et al.*, Sections 4.7.2-6, A.9

Data Representation 'Domains'

- Data domain:
  - Domain in which data are acquired.
    - Examples: Output of a geophone as a *function of time*, value of gravity at a point on a spatial grid
    - Time or space
- Transform domains:
  - Transformed for interpretation and understanding of certain aspects of the record as a whole.
  - Frequency, 'wave number’, velocity, *etc*….
- There are numerous transforms for continuous and discrete signal...
  - We are interested in *discrete, numerical transforms*

Z-Transform

- Consider a digitized record that is represented by a *series* of N readings: $U = \{u_0, u_1, u_2, \ldots, u_{N-1}\}$

How can we represent this series differently?

- The Z transform associates a *polynomial function* with this time series:

$$U(z) = u_0 + u_1 z + u_2 z^2 + u_3 z^3 + \ldots$$

  - For example, a 3-sample record of {1,2,5} is represented by a quadratic polynomial:

    $1 + 2z + 5z^2$.

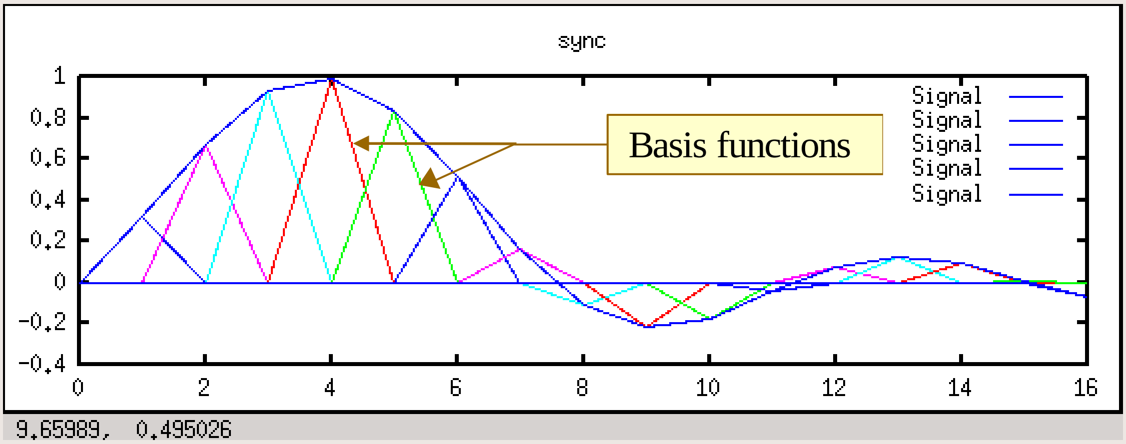
- In the Z-domain, the all-important operation of *convolution* of time series becomes simple multiplication of Z-transforms:

$$U_1 * U_2 \Leftrightarrow U_1(z) U_2(z)$$

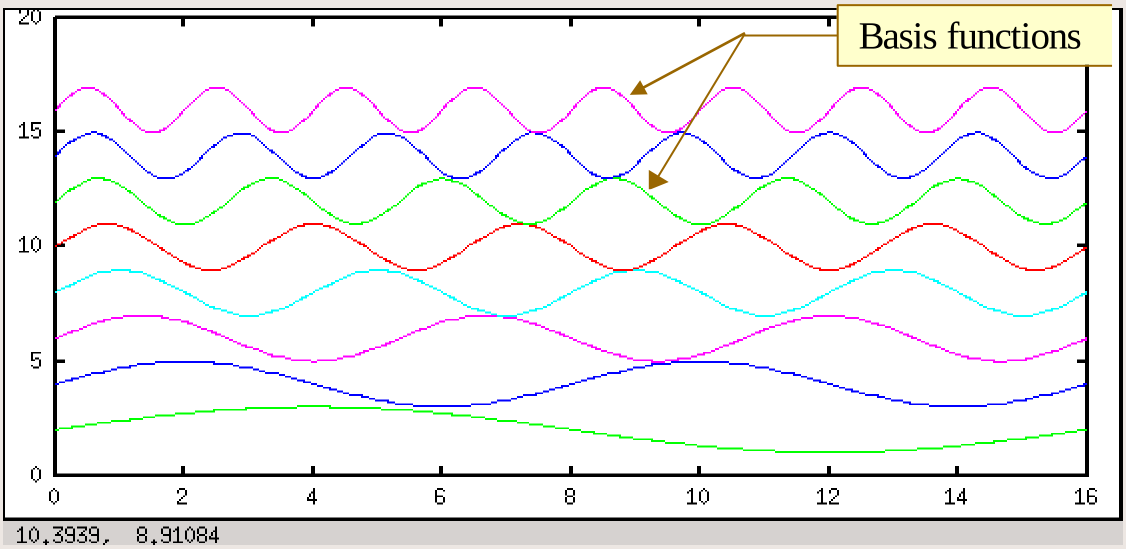
- We will return to this during the discussion of convolution.

Fourier Transform

- Time series represent the signal as a sum of *basis functions* – triangular pulses localized in time:



- Fourier transform represents the signal as a sum of sin(...), cos(...), or complex exp(...) basis functions with different *frequencies*:



Summary of Forward and Inverse Fourier Transforms

- *Forward Fourier Transform* (from time to frequency domain):

$$U_k = \sum_{m=0}^{N-1} e^{-i\frac{2\pi k}{N}m} u_m \qquad (1)$$

frequency $f = k\Delta f$ time $t = m\Delta t$

- The *Inverse Fourier Transform* (from frequency to tine domain) is given by a similar formula:

$$u_j = \frac{1}{N} \sum_{k=0}^{N-1} e^{i\frac{2\pi k}{N}j} U_k \qquad (2)$$

time $t = j\Delta t$ frequency $f = k\Delta f$

$$\Delta t = \frac{T}{N} = \frac{1}{f_s} \qquad \Delta f = \frac{f_s}{N} = \frac{1}{T}$$

Exercise: Prove this (plug (1) in (2) above)

Nyquist frequency

- Recall the frequency folding and aliasing phenomena we discussed before
- These phenomena are simply due to the fact that the time-domain signal $u(t)$ is real-valued, but the frequency-domain $U(f)$ is complex-valued.
  - This means that the $\{U_k\}$ series contain twice more numbers than $\{u_j\}$
  - Therefore, half of the values in $\{U_k\}$ must always be related to the other half. This is how they are related (this is the frequency folding):

$$U\left(f_s - f\right) = U^*\left(f\right)$$

- Thus, it is sufficient to know $U(f)$ only up to Nyquist frequency

$$f_N = \frac{f_s}{2} = \frac{N}{2}\Delta f = \frac{1}{2\Delta t}$$

- At $f > f_N$, the spectrum $U(f)$ is a "conjugate mirror image" of the spectrum below f_N

Spectra

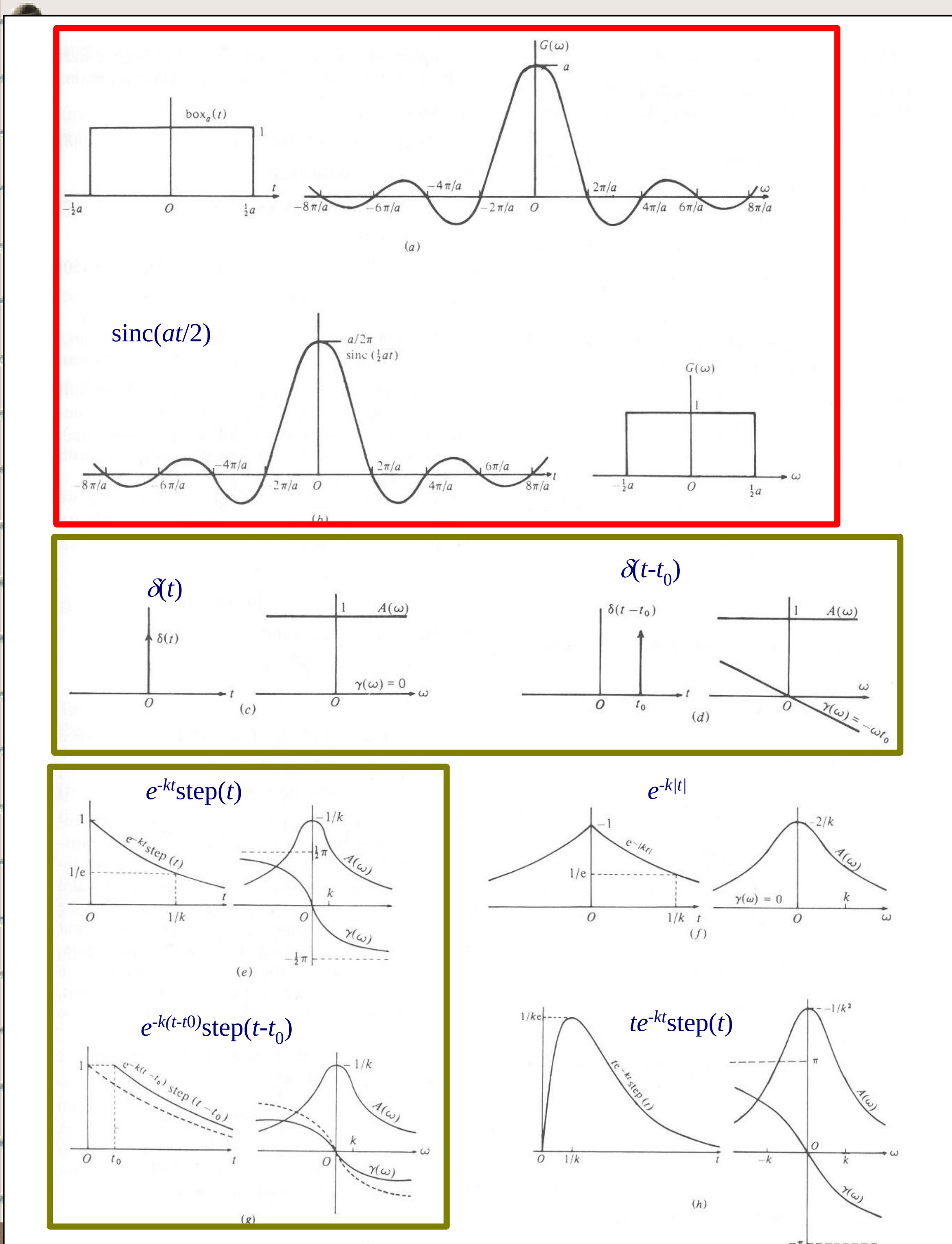
- In *frequency domain*, the signal $U(f)$ becomes complex-valued, and it varies with frequencies rather than times:

$$u(t) \Rightarrow U(f) = A(f)e^{i\theta(f)}$$

- $A(f)$ is called the *amplitude spectrum*, and $\theta(f)$ is the *phase spectrum* of the signal.
- $A(f)$ shows the amplitude of the particular *harmonic component* of the record, and $\theta(f)$ shows its relative phase
- $A(f)$ is measured in the same units as the amplitude, and $\theta(f)$ is dimensionless (or *radians*, often also expressed *in degrees*: 180° = π).

Sample Fourier Transforms

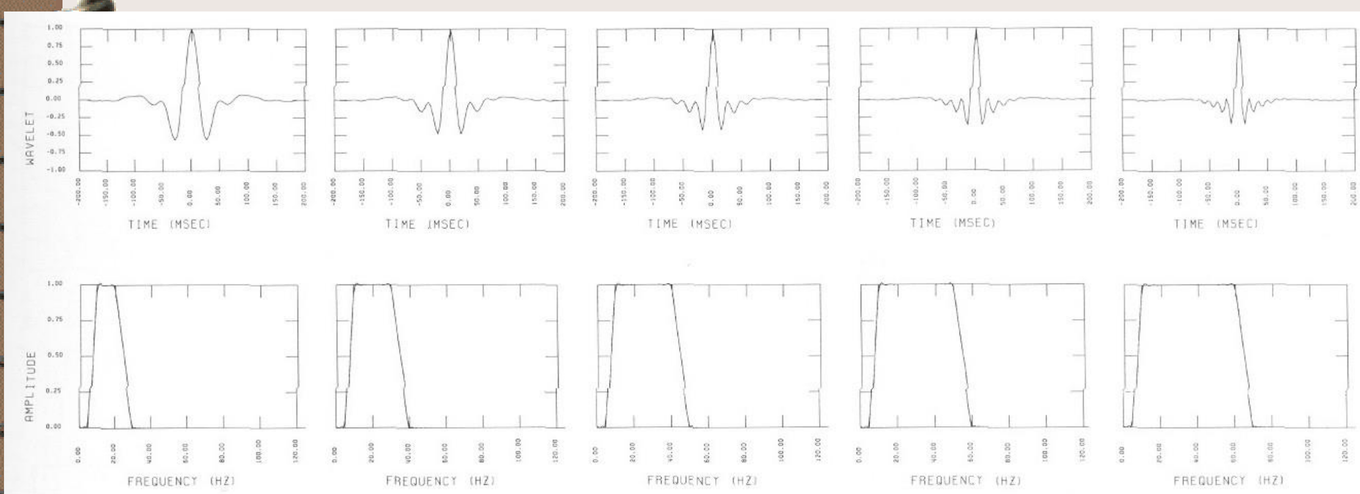
- Compare the transforms in the boxes



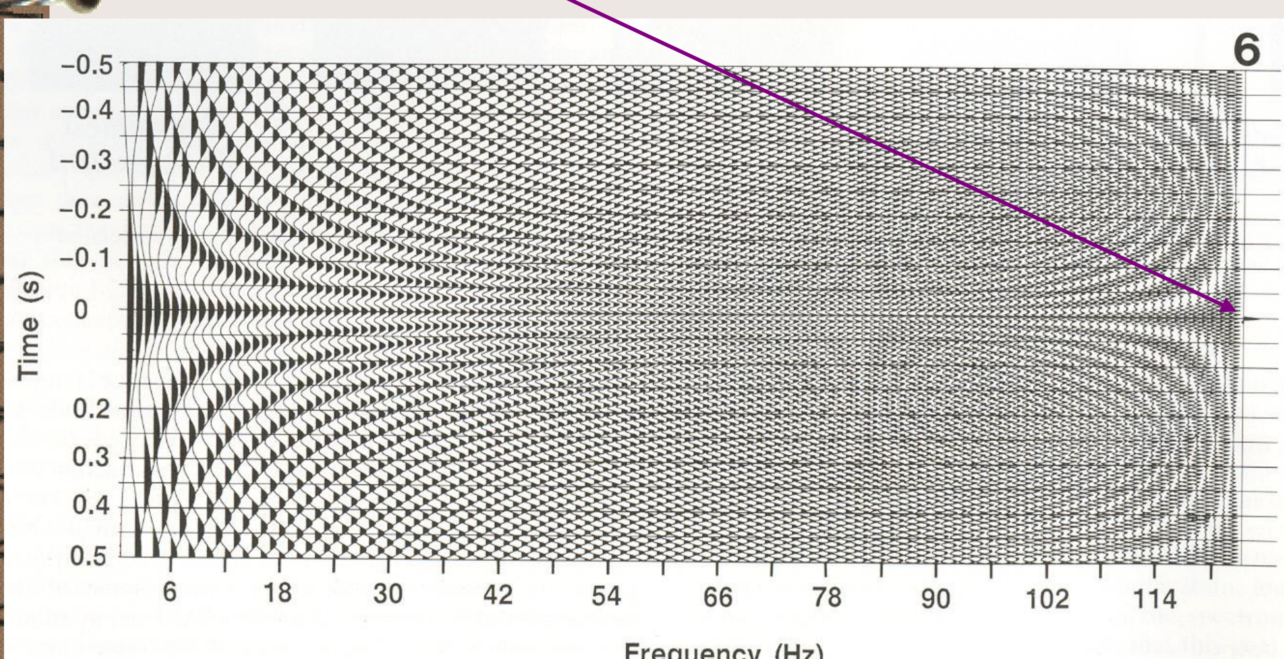
From Sheriff, Geldart, 1995

Spectra of Pulses

- For a pulse of width T s, its spectrum is about $1/T$ Hz in width:



- Equal-amplitude (co)sinusoids from 0 to f_N add up to form a spike:



From Yilmaz, 1987

Fast Fourier Transform

- The *Fast Fourier Transform* (*FFT*) is an efficient *algorithm* to compute the Fourier transforms
- It works with a series of N samples that can be efficiently *factorized* in terms of *prime factors*. The best-known, classic FFT uses $N = 2^n$.
- FFT utilizes trigonometric relations such as:

$$e^{-i2\alpha} = \left(e^{-i\alpha}\right)^2$$

  - Therefore, the sums computed for frequency f can be utilized to compute the FFT's at frequency $2f$, and so on.
  - As a result, FFT computes all frequency points in $\sim N\log_2 N$ steps instead of N^2
    - ~10 times speedup for $N = 1024$