

# SDOF linear oscillator

## Frequency Domain Analysis

Giacomo Boffi

Dipartimento di Ingegneria Strutturale, Politecnico di Milano

March 30, 2011

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Fast Fourier Transform

## Outline

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Fast Fourier Transform

### Fourier Transform

Extension of Fourier Series to non periodic functions  
Response in the Frequency Domain

### The Discrete Fourier Transform

The Discrete Fourier Transform  
Aliasing

### The Fast Fourier Transform

The Fast Fourier Transform

## Non periodic loadings

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

Extension of Fourier Series to non periodic functions

Response in the Frequency Domain

The Discrete Fourier Transform

The Fast Fourier Transform

It is possible to extend the Fourier analysis to non periodic loading. Let's start from the Fourier series representation of the load  $p(t)$ ,

$$p(t) = \sum_{-\infty}^{+\infty} P_r \exp(i\omega_r t), \quad \omega_r = r\Delta\omega, \quad \Delta\omega = \frac{2\pi}{T_p},$$

introducing  $P(i\omega_r) = P_r T_p$  and substituting,

$$p(t) = \frac{1}{T_p} \sum_{-\infty}^{+\infty} P(i\omega_r) \exp(i\omega_r t) = \frac{\Delta\omega}{2\pi} \sum_{-\infty}^{+\infty} P(i\omega_r) \exp(i\omega_r t).$$

Due to periodicity, we can modify the extremes of integration in the expression for the complex amplitudes,

$$P(i\omega_r) = \int_{-T_p/2}^{+T_p/2} p(t) \exp(-i\omega_r t) dt.$$

## Non periodic loadings (2)

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

Extension of Fourier Series to non periodic functions

Response in the Frequency Domain

The Discrete Fourier Transform

The Fast Fourier Transform

If the loading period is extended to infinity to represent the non-periodicity of the loading ( $T_p \rightarrow \infty$ ) then (a) the frequency increment becomes infinitesimal ( $\Delta\omega = \frac{2\pi}{T_p} \rightarrow d\omega$ ) and (b) the discrete frequency  $\omega_r$  becomes a continuous variable,  $\omega$ .

In the limit, for  $T_p \rightarrow \infty$  we can then write

$$p(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} P(i\omega) \exp(i\omega t) d\omega$$
$$P(i\omega) = \int_{-\infty}^{+\infty} p(t) \exp(-i\omega t) dt,$$

which are known as the inverse and the direct Fourier Transforms, respectively, and are collectively known as the Fourier transform pair.

# SDOF Response

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

Extension of Fourier Series to non periodic functions

Response in the Frequency Domain

The Discrete Fourier Transform

The Fast Fourier Transform

In analogy to what we have seen for periodic loads, the response of a damped SDOF system can be written in terms of  $H(i\omega)$ , the complex frequency response function,

$$x(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} H(i\omega) P(i\omega) \exp i\omega t dt, \quad \text{where}$$

$$H(i\omega) = \frac{1}{k} \left[ \frac{1}{(1 - \beta^2) + i(2\zeta\beta)} \right] = \frac{1}{k} \left[ \frac{(1 - \beta^2) - i(2\zeta\beta)}{(1 - \beta^2)^2 + (2\zeta\beta)^2} \right], \quad \beta = \frac{\omega}{\omega_n}.$$

To obtain the response *through frequency domain*, you should evaluate the above integral, but analytical integration is not always possible, and when it is possible, it is usually very difficult, implying contour integration in the complex plane (for an example, see Example **E6-3** in Clough Penzien).

# Discrete Fourier Transform

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Discrete Fourier Transform

Aliasing

The Fast Fourier Transform

To overcome the analytical difficulties associated with the inverse Fourier transform, one can use appropriate numerical methods, leading to good approximations.

Consider a loading of finite period  $T_p$ , divided into  $N$  equal intervals  $\Delta t = T_p/N$ , and the set of values  $p_s = p(t_s) = p(s\Delta t)$ . We can approximate the complex amplitude coefficients with a sum,

$$P_r = \frac{1}{T_p} \int_0^{T_p} p(t) \exp(-i\omega_r t) dt, \quad \text{that, by trapezoidal rule, is}$$
$$\approx \frac{1}{N\Delta t} \left( \Delta t \sum_{s=0}^{N-1} p_s \exp(-i\omega_r t_s) \right) = \frac{1}{N} \sum_{s=0}^{N-1} p_s \exp(-i \frac{2\pi r s}{N}).$$

# Discrete Fourier Transform (2)

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Discrete Fourier Transform

Aliasing

The Fast Fourier Transform

In the last two passages we have used the relations

$$p_N = p_0, \quad \exp(i\omega_r t_N) = \exp(ir\Delta\omega T_p) = \exp(ir2\pi) = \exp(i0)$$

$$\omega_r t_s = r\Delta\omega s\Delta t = rs \frac{2\pi T_p}{T_p N} = \frac{2\pi rs}{N}$$

Take note that the discrete function  $\exp(-i\frac{2\pi rs}{N})$ , defined for integer  $r, s$  is periodic with period  $N$ , implying that the complex amplitude coefficients are themselves periodic with period  $N$ .

$$P_{r+N} = P_r$$

Starting in the time domain with  $N$  distinct complex numbers,  $p_s$ , we have found that in the frequency domain our load is described by  $N$  distinct complex numbers,  $P_r$ , so that we can say that our function is described by the same amount of information in both domains.

# Aliasing

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

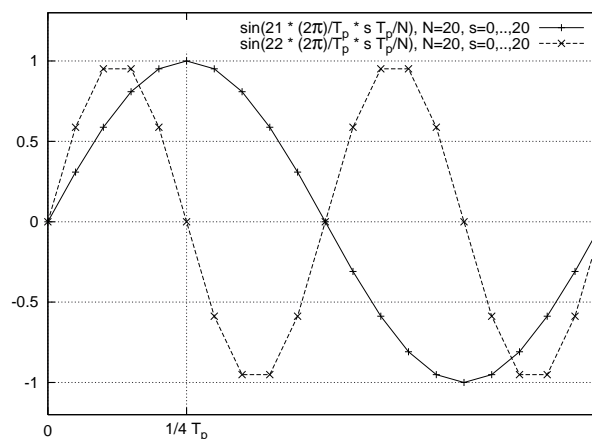
The Discrete Fourier Transform

Aliasing

The Fast Fourier Transform

Only  $N/2$  distinct frequencies ( $\sum_0^{N-1} = \sum_{-N/2}^{+N/2}$ ) contribute to the load representation, what if the *frequency content* of the loading has contributions from frequencies higher than  $\omega_{N/2}$ ? What happens is *aliasing*, i.e., the upper frequency contributions are mapped to contributions of lesser frequency.

See the plot above: the contributions from the high frequency sines, *when sampled*, are indistinguishable from the contributions from lower frequency components, i.e., are *aliased* to lower frequencies!



## Aliasing (2)

- ▶ The maximum frequency that can be described in the DFT is called the Nyquist frequency,  $\omega_{Ny} = \frac{1}{2} \frac{2\pi}{\Delta t}$ .
- ▶ It is usual in signal analysis to remove the signal's higher frequency components preprocessing the signal with a *filter* or a *digital filter*.
- ▶ It is worth noting that the *resolution* of the DFT in the frequency domain for a given sampling rate is proportional to the number of samples, i.e., to the duration of the sample.

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Discrete Fourier Transform

Aliasing

The Fast Fourier Transform

## Fast Fourier Transform (1)

The algorithm is the same for both direct and inverse DFT, so let us consider the direct transform,

$$A_r = P_r N = \sum_{s=0}^{N-1} p_s \exp(-i \frac{2\pi r s}{N}), \quad r = 0, 1, 2, \dots, N-1$$

A straightforward implementation requires about  $N^2$  complex products, becoming prohibitive for even moderately large  $N$ 's. The FFT is based on the decomposition of  $N$  in a product of its factors, but the algorithm was developed and is simpler to understand and implement if  $N = 2^\gamma$ . In this case each  $r, s$  in the interval  $0, N-1$  can be expressed in terms of binary (i.e., 0 or 1) coefficients

$$r = 2^{\gamma-1} r_{\gamma-1} + 2^{\gamma-2} r_{\gamma-2} + \dots + 2^0 r_0$$
$$s = 2^{\gamma-1} s_{\gamma-1} + 2^{\gamma-2} s_{\gamma-2} + \dots + 2^0 s_0$$

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Fast Fourier Transform

The Fast Fourier Transform

## Fast Fourier Transform (2)

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Fast Fourier Transform

The Fast Fourier Transform

With  $W_N = \exp(-i2\pi/N)$  we write

$$A(r_{\gamma-1}, r_{\gamma-2}, \dots, r_0) = \sum_{s_0=0}^1 \sum_{s_1=0}^1 \cdots \sum_{s_{\gamma-2}=0}^1 \sum_{s_{\gamma-1}=0}^1 p_0(s_{\gamma-1}, s_{\gamma-2}, \dots, s_0) W_N^{rs}$$

The subscript  $_0$  is added to the  $p$  coefficients for a reason that will become apparent as the algorithm develops.

## Fast Fourier Transform (3)

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Fast Fourier Transform

The Fast Fourier Transform

In the previous slide we wrote  $W_N^{rs}$ , but we have to use the binary representation of  $r$  and  $s$ ,

$$W_N^{rs} = W_N^{(2^{\gamma-1}r_{\gamma-1} + 2^{\gamma-2}r_{\gamma-2} + \cdots + 2^0r_0)(2^{\gamma-1}s_{\gamma-1} + 2^{\gamma-2}s_{\gamma-2} + \cdots + 2^0s_0)}$$

As  $W_N^{a+b} = W_N^a W_N^b$ , we can expand the equation above to separate the contributions of the different binary indices in the binary representation of  $s$

$$\begin{aligned} W_N^{rs} &= W_N^{(2^{\gamma-1}r_{\gamma-1} + 2^{\gamma-2}r_{\gamma-2} + \cdots + 2^0r_0)(2^{\gamma-1}s_{\gamma-1})} \\ &\quad \times W_N^{(2^{\gamma-1}r_{\gamma-1} + 2^{\gamma-2}r_{\gamma-2} + \cdots + 2^0r_0)(2^{\gamma-2}s_{\gamma-2})} \\ &\quad \times \cdots \times W_N^{(2^{\gamma-1}r_{\gamma-1} + 2^{\gamma-2}r_{\gamma-2} + \cdots + 2^0r_0)(2^0s_0)} \end{aligned}$$

In the next slide we are going to rewrite each term in the above expression, to obtain a surprising simplification.

## Fast Fourier Transform (4)

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Fast Fourier Transform

The Fast Fourier Transform

The first term on the right in previous slide was

$$W_N^{(2^{\gamma-1}r_{\gamma-1}+2^{\gamma-2}r_{\gamma-2}+\dots+2^0r_0)(2^{\gamma-1}s_{\gamma-1})}$$

expanding the exponent

$$\begin{aligned} &= W_N^{2^{\gamma}(2^{\gamma-2}r_{\gamma-1}s_{\gamma-1})} \times W_N^{2^{\gamma}(2^{\gamma-3}r_{\gamma-2}s_{\gamma-1})} \times \dots \\ &\quad \dots \times W_N^{2^{\gamma}(2^1r_1s_{\gamma-1})} \times W_N^{2^{\gamma-1}(r_0s_{\gamma-1})} \\ &= W_N^{2^{\gamma-1}(2^0r_0)s_{\gamma-1}} \end{aligned}$$

because  $W_N^{2^{\gamma}(\text{integer})} = 1$ .

In a similar manner, we have

$$W_N^{(2^{\gamma-1}r_{\gamma-1}+2^{\gamma-2}r_{\gamma-2}+\dots+2^0r_0)(2^{\gamma-2}s_{\gamma-2})} = W_N^{2^{\gamma-2}(2^1r_1+2^0r_0)s_{\gamma-2}}$$

$$W_N^{(2^{\gamma-1}r_{\gamma-1}+2^{\gamma-2}r_{\gamma-2}+\dots+2^0r_0)(2^{\gamma-3}s_{\gamma-3})} = W_N^{2^{\gamma-3}(2^2r_2+2^1r_1+2^0r_0)s_{\gamma-3}}$$

...

$$W_N^{(2^{\gamma-1}r_{\gamma-1}+2^{\gamma-2}r_{\gamma-2}+\dots+2^0r_0)(2^0s_0)} = W_N^{2^0(2^{\gamma-1}r_{\gamma-1}+2^{\gamma-2}r_{\gamma-2}+\dots+2^0r_0)s_0}$$

## Fast Fourier Transform (5)

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Fast Fourier Transform

The Fast Fourier Transform

Substituting all  $W_N$  terms, in their reduced form,

$$\begin{aligned} A(r_{\gamma-1}, r_{\gamma-2}, \dots, r_0) &= \\ &\sum_{s_0=0}^1 \sum_{s_1=0}^1 \dots \sum_{s_{\gamma-2}=0}^1 \sum_{s_{\gamma-1}=0}^1 p_0(s_{\gamma-1}, s_{\gamma-2}, \dots, s_0) \times W_N^{2^{\gamma-1}(2^0r_0)s_{\gamma-1}} \\ &\quad \times W_N^{2^{\gamma-2}(2^1r_1+2^0r_0)s_{\gamma-2}} \times \dots \times W_N^{2^0(2^{\gamma-1}r_{\gamma-1}+2^{\gamma-2}r_{\gamma-2}+\dots+2^0r_0)s_0} \end{aligned}$$

Carrying out all summations in succession, we have

$$\sum_{s_{\gamma-1}=0}^1 p_0(s_{\gamma-1}, s_{\gamma-2}, \dots, s_0) \times W_N^{2^{\gamma-1}(2^0r_0)s_{\gamma-1}} \equiv p_1(r_0, s_{\gamma-2}, \dots, s_0)$$

$$\sum_{s_{\gamma-2}=0}^1 p_1(r_0, s_{\gamma-2}, \dots, s_0) \times W_N^{2^{\gamma-2}(2^1r_1+2^0r_0)s_{\gamma-2}} \equiv p_2(r_0, r_1, s_{\gamma-3}, \dots, s_0)$$

# Fast Fourier Transform (6)

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Fast Fourier Transform

The Fast Fourier Transform

Proceeding with the summations, we arrive at the last one, that gives us the coefficients  $A_n$

$$\sum_{s_0=0}^1 p_{\gamma-1}(r_0, r_1, \dots, r_{\gamma-2}, s_0) \times W_N^{2^0(2^{\gamma-1}r_{\gamma-1}+2^{\gamma-2}r_{\gamma-2}+\dots+2^0r_0)s_0}$$

$$\equiv p_{\gamma}(r_0, r_1, \dots, r_{\gamma-2}, r_{\gamma-1})$$

$$\equiv A(r_{\gamma-1}, r_{\gamma-2}, \dots, r_0)$$

The coefficients are computed in an order different from what intended (we have the so called *bit reversal*), but it's simple to reorder them. Our last remark, the number of multiplications using the Cooley-Tukey algorithm is in the order of  $N \log(N)$ , the savings even for moderately large  $N$  with respect to  $N^2$  are well worth the complication of the FFT algorithm.

FFT looks complex? Here it is an example of a simple, standard implementation of the FFT.

# Alternative Algorithm

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Fast Fourier Transform

The Fast Fourier Transform

Decimation in Time algorithm by Tukey and Cooley (1965), assume  $N$  is even, and divide the DFT summation to consider even and odd indices  $s$

$$X_r = \sum_{s=0}^{N-1} x_s e^{-\frac{2\pi i}{N} sr}, \quad r = 0, \dots, N-1$$

$$= \sum_{q=0}^{N/2-1} x_{2q} e^{-\frac{2\pi i}{N} (2q)r} + \sum_{q=0}^{N/2-1} x_{2q+1} e^{-\frac{2\pi i}{N} (2q+1)r}$$

collecting  $e^{-\frac{2\pi i}{N} r}$  in the second term and letting  $\frac{2q}{N} = \frac{q}{N/2}$

$$= \sum_{q=0}^{N/2-1} x_{2q} e^{-\frac{2\pi i}{N/2} qr} + e^{-\frac{2\pi i}{N} r} \sum_{q=0}^{N/2-1} x_{2q+1} e^{-\frac{2\pi i}{N/2} qr}$$

We have two DFT's of length  $N/2$ , the operations count is hence  $2(N/2)^2 = N^2/2$ , but we have to combine these two halves in the full DFT.



## Alternative 2

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Fast Fourier Transform

The Fast Fourier Transform

Say that

$$X_r = E_r + e^{-\frac{2\pi i}{N}r} O_r$$

where  $E_r$  and  $O_r$  are the even and odd half-DFT's, of which we computed only coefficients from 0 to  $N/2 - 1$ .

To get the full sequence we have to note that

1. the E and O DFT's are periodic with period  $N/2$ , and
2.  $\exp(-2\pi i(r + N/2)/N) = e^{-\pi i} \exp(-2\pi i r/N) = -\exp(-2\pi i r/N)$ ,

so that we can write

$$X_r = \begin{cases} E_r + \exp(-2\pi i r/N) O_r & \text{if } r < N/2, \\ E_{r-N/2} - \exp(-2\pi i r/N) O_{r-N/2} & \text{if } r \geq N/2. \end{cases}$$

The algorithm that was outlined can be applied to the computation of each of the half-DFT's when  $N/2$  were even, so that the operation count goes to  $N^2/4$ . If  $N/4$  were even ...

## Pseudocode for CT algorithm

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Fast Fourier Transform

The Fast Fourier Transform

```
def fft2(X, N):
    if N = 1 then
        Y = X
    else
        Y0 = fft2(X0, N/2)
        Y1 = fft2(X1, N/2)
        for k = 0 to N/2-1
            Y`k          = Y0`k + exp(2 pi i k/N) Y1`k
            Y`(k+N/2) = Y0`k - exp(2 pi i k/N) Y1`k
        endfor
    endif
return Y
```

# Dynamic Response (1)

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Fast Fourier Transform

The Fast Fourier Transform

To evaluate the dynamic response of a linear SDOF system in the frequency domain, use the inverse DFT,

$$x_s = \sum_{r=0}^{N-1} V_r \exp(i \frac{2\pi r s}{N}), \quad s = 0, 1, \dots, N-1$$

where  $V_r = H_r P_r$ .  $P_r$  are the discrete complex amplitude coefficients computed using the direct DFT, and  $H_r$  is the discretization of the complex frequency response function, that for viscous damping is

$$H_r = \frac{1}{k} \left[ \frac{1}{(1 - \beta_r^2) + i(2\zeta\beta_r)} \right] = \frac{1}{k} \left[ \frac{(1 - \beta_r^2) - i(2\zeta\beta_r)}{(1 - \beta_r^2)^2 + (2\zeta\beta_r)^2} \right], \quad \beta_r = \frac{\omega_r}{\omega_n}$$

while for hysteretic damping is

$$H_r = \frac{1}{k} \left[ \frac{1}{(1 - \beta_r^2) + i(2\zeta)} \right] = \frac{1}{k} \left[ \frac{(1 - \beta_r^2) - i(2\zeta)}{(1 - \beta_r^2)^2 + (2\zeta)^2} \right]$$

# Some words of caution

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Fast Fourier Transform

The Fast Fourier Transform

If you're going to approach the application of the complex frequency response function without proper concern, you're likely to be hurt.

Let's say  $\Delta\omega = 1.0$ ,  $N = 32$ ,  $\omega_n = 3.5$  and  $r = 30$ , what do you think it is the value of  $\beta_{30}$ ? If you are thinking  $\beta_{30} = 30 \Delta\omega / \omega_n = 30 / 3.5 \approx 8.57$  you're wrong!

Due to aliasing,  $\omega_r = \begin{cases} r\Delta\omega & r \leq N/2 \\ (r - N)\Delta\omega & r > N/2 \end{cases}$

note that in the upper part of the DFT the coefficients correspond to negative frequencies and, staying within our example, it is  $\beta_{30} = (30 - 32) \times 1 / 3.5 \approx -0.571$ .

If  $N$  is even,  $P_{N/2}$  is the coefficient corresponding to the Nyquist frequency, if  $N$  is odd  $P_{\frac{N-1}{2}}$  corresponds to the largest positive frequency, while  $P_{\frac{N+1}{2}}$  corresponds to the largest negative frequency.

## Transfer Functions

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Fast Fourier Transform

The Fast Fourier Transform

The response of a linear SDOF system to arbitrary loading can be evaluated by a convolution integral in the time domain,

$$x(t) = \int_0^t p(\tau) h(t - \tau) d\tau,$$

with the unit impulse response function

$h(t) = \frac{1}{m\omega_D} \exp(-\zeta\omega_n t) \sin(\omega_D t)$ , or through the frequency domain using the Fourier integral

$$x(t) = \int_{-\infty}^{+\infty} H(\omega) P(\omega) \exp(i\omega t) d\omega,$$

where  $H(\omega)$  is the complex frequency response function.

## Transfer Functions

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Fast Fourier Transform

The Fast Fourier Transform

These response functions, or *transfer* functions, are connected by the direct and inverse Fourier transforms:

$$H(\omega) = \int_{-\infty}^{+\infty} h(t) \exp(-i\omega t) dt,$$
$$h(t) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} H(\omega) \exp(i\omega t) d\omega.$$

## Relationship of transfer functions

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Fast Fourier Transform

The Fast Fourier Transform

We write the response and its Fourier transform:

$$x(t) = \int_0^t p(\tau)h(t-\tau) d\tau = \int_{-\infty}^t p(\tau)h(t-\tau) d\tau$$

$$X(\omega) = \int_{-\infty}^{+\infty} \left[ \int_{-\infty}^t p(\tau)h(t-\tau) d\tau \right] \exp(-i\omega t) dt$$

the lower limit of integration in the first equation was changed from 0 to  $-\infty$  because  $p(\tau) = 0$  for  $\tau < 0$ , and since  $h(t-\tau) = 0$  for  $\tau > t$ , the upper limit of the second integral in the second equation can be changed from  $t$  to  $+\infty$ ,

$$X(\omega) = \lim_{s \rightarrow \infty} \int_{-s}^{+s} \int_{-s}^{+s} p(\tau)h(t-\tau) \exp(-i\omega t) dt d\tau$$

## Relationship of transfer functions

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Fast Fourier Transform

The Fast Fourier Transform

Introducing a new variable  $\theta = t - \tau$  we have

$$X(\omega) = \lim_{s \rightarrow \infty} \int_{-s}^{+s} p(\tau) \exp(-i\omega\tau) d\tau \int_{-s-\tau}^{+s-\tau} h(\theta) \exp(-i\omega\theta) d\theta$$

with  $\lim_{s \rightarrow \infty} s - \tau = \infty$ , we finally have

$$\begin{aligned} X(\omega) &= \int_{-\infty}^{+\infty} p(\tau) \exp(-i\omega\tau) d\tau \int_{-\infty}^{+\infty} h(\theta) \exp(-i\omega\theta) d\theta \\ &= P(\omega) \int_{-\infty}^{+\infty} h(\theta) \exp(-i\omega\theta) d\theta \end{aligned}$$

where we have recognized that the first integral is the Fourier transform of  $p(t)$ .

# Relationship of transfer functions

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Fast Fourier Transform

The Fast Fourier Transform

Our last relation was

$$X(\omega) = P(\omega) \int_{-\infty}^{+\infty} h(\theta) \exp(-i\omega\theta) d\theta$$

but  $X(\omega) = H(\omega)P(\omega)$ , so that, noting that in the above equation the last integral is just the Fourier transform of  $h(\theta)$ , we may conclude that, effectively,  $H(\omega)$  and  $h(t)$  form a Fourier transform pair.

```
from cmath import exp, pi

def d_fft(x,n):
    """Direct fft of x, a list of n=2**m complex values"""
    return _fft(x,n,[exp(-2*pi*1j*k/n) for k in range(n/2)])

def i_fft(x,n):
    """Inverse fft of x, a list of n=2**m complex values"""
    return [x/n for x in _fft(x,n,[exp(+2*pi*1j*k/n) for k in range(n/2)])]

def _fft(x, n, twiddle):
    """Decimation in Time FFT, to be called by d_fft and i_fft.
    x is the signal to transform, a list of complex values
    n is its length, results are undefined if n is not a power of 2
    tw is a list of twiddle factors, precomputed by the caller

    returns a list of complex values, to be normalized in case of an
    inverse transform"""

    if n == 1: return x # bottom reached, DFT of a length 1 vec x is x

    # call fft with the even and the odd coefficients in x
    # the results are the so called even and odd DFT's
    y_0 = _fft(x[0::2], n/2, tw[::2])
    y_1 = _fft(x[1::2], n/2, tw[::2])

    # assemble the partial results "in-place":
    # 1st half of full DFT is put in even DFT, 2nd half in odd DFT
    for k in range(n/2):
        y_0[k], y_1[k] = y_0[k]+tw[k]*y_1[k], y_0[k]-tw[k]*y_1[k]

    # concatenate the two halves of the DFT and return to caller
    return y_0+y_1
```

SDOF linear oscillator

Giacomo Boffi

Fourier Transform

The Discrete Fourier Transform

The Fast Fourier Transform

The Fast Fourier Transform

```
def main():
    """Run some test cases"""
    from cmath import cos, sin, pi

    def testit(title, seq):
        """utility to format and print a vector and the ifft of its fft"""
        l_seq = len(seq)
        print "-"*5, title, "-"*5
        print "\n".join([
            "%10.6f_::_%10.6f,_%10.6fj" % (a.real, t.real, t.imag)
            for (a, t) in zip(seq, i_ffft(d_fft(seq, l_seq), l_seq))
        ])

    length = 32

    testit("Square_wave", [+1.0+0.0j]*(length/2) + [-1.0+0.0j]*(length/2))
    testit("Sine_wave", [sin((2*pi*k)/length) for k in range(length)])
    testit("Cosine_wave", [cos((2*pi*k)/length) for k in range(length)])

if __name__ == "__main__":
    main()
```