Introduction to Digital Signal Processing

Dr. Gal Ben-David

Technion, 2012

Summarized by Ido Greenberg

References: lectures, book (including many helpful visualizations)

Contents

Signals in Fourier Space and Effects of Discretization (Digitation)

Fourier Transform (FT)

Transforming a signal from time-space to frequency-space is achieved by **continuous Fourier-Transform**, i.e. by converting from delta-functions-basis to (the orthogonal) **trigonometric-functions-basis**.

Note: time/frequency terminology is usually used, although the theory is more generic and can be applied, for example, to space/frequency as well (as in image processing).

- **Uncertainty principle of Heisenberg**: short support in time (i.e. fast decay in time) ➔ long support in freq (i.e. slow decay in freq).
  Note: in reality, all sampled signals are finite in time, thus infinite in freq.
- **Smoothness**: n bounded derivatives in time ➔ $1/\omega^{n+1}$-fast decay in freq (equivalently, jumps in time cause more significant high frequencies).
- **Convolution** in time ➔ multiplication in freq.
- **Derivation** in time ➔ $\cdot\, i\omega$ in freq.
- **Duality** of the relationship between time and freq: the roles of time and freq in the properties above can be swapped.

Discrete-Time Fourier Transform (DTFT)

DTFT is a FT of [a signal sampled in discrete times].

Two ways to see discretization in time (i.e. sampling):

1. Changing from integral ($\int$) to sum ($\sum$) over discrete points in time-space – which deserves change of variable notation: x(t) → x(n).
2. Multiplying the signal by "train" of delta functions (in time-space) before applying FT – which can keep us with homogeneous notation x(t).

DTFT with sampling interval T causes **duplication** in frequency-space of the signal, in freq intervals of $\Omega = 2\pi/T$.

- **Shannon frequency**: If the frequency is known to be bounded by $\Omega = 2\pi F$ (which is practically never true due to time being finite, see above), then sampling in $T < 1/(2F)$ (or $\boldsymbol{f > 2F}$) ensures that the supports of the duplications are disjoint, i.e. the duplications do not overlap, thus by slicing the frequency range $(-f, f)$ **the signal can be fully recovered from its (infinite) discrete sampling!**
  - Note: it is completely dual to a signal within bounded time interval [-T,T] being fully represented by discrete Fourier series.
- **Aliasing**: sampling in lower frequencies than Shannon frequency ($f < 2F$) causes impersonation of the (impossible-to-measure) high frequencies to lower frequencies ($f - F$). For example, f=80Hz-sampling of F=50Hz-signal would create fake frequency peak in f-F=80-50=30Hz.
- **Short band in high frequency**: for frequency band of **width B around a <u>known</u> central freq F>>B**, sampling of **f>2B is enough**. One can just eliminate the null frequencies f<F-B and recover the original signal.
  That's fortunate, since using small antennas requires small transmitted wavelengths, which requires multiplication of the transmitted signals by a high frequency.

Discrete Fourier Transform (DFT)

Approximating FT using digital computational tools requires **discretization** and **bounding** of both **time** and **frequency**. Applying all of these to FT results in the Discrete FT (**DFT**).

So that's what we do to a signal in order to digitally process it:

	Effect in time-space	**Effect in frequency-space**
Discretization in time (DTFT)	**Sampling**: multiplying by a train of deltas.	Convoluting with train of deltas – i.e. **infinite duplication** of the signal.
Bounding time	**Bounding**: multiplying by a rectangle.	Convoluting with Dirichlet Kernel (the DTFT of a rectangle, which is an infinite duplication of sync) – i.e. **leakage of frequencies**.
Discretization in frequency	Duplications in time are irrelevant since time is already bounded, thus **no additional side-effects**.	**Sampling**: multiplying by a train of deltas.
Bounding frequency	Practically already caused by discretization in time (since the duplication in frequency enforces us to look only at bounded frequencies anyway), thus **no additional side-effects**.	

Notations:

- Continuous time $x(t)$ $-\infty < t < \infty$
- Discrete time $x(n)$ $0 \leq n \leq N-1$
- Continuous frequency $X^F(\omega)$ $-\infty < \omega < \infty$ also $F[x](\omega)$ or $\hat{x}(\omega)$
- Discrete frequency $X^D(k)$ $0 \leq k \leq N-1$

Padding with zeros: adding zeros to a finite signal allows increase of N without modifying the information of the signal, which improves the frequency resolution of the DFT (see windowing below).

Examples

	Rectangle $rect(t) = 1 \cdot (\lvert t\rvert \leq 1/2)$	**Single frequency** $\cos(\omega_0 t)$
FT	**Sync** $\frac{\sin(\omega/2)}{\omega/2}$	**Two symmetric deltas** $\frac{1}{2}\delta(\omega-\omega_0)+\frac{1}{2}\delta(\omega+\omega_0)$
DTFT	**Dirichlet-Kernel** $\frac{\sin(\omega N/2)}{\sin(\omega/2)}$	**Infinite train of pairs of deltas** $\sum_{K\in Z}\delta\big(\omega-(2\pi K\pm\omega_0)\big)$
DFT	**Dirichlet-Kernel** (already bounded in time…)	**Two leaked deltas** $\delta(\omega\pm\omega_0) * D_N(\omega)$ Note: sampling in exactly $T = 2\pi/\omega_0$ would prevent the leakage since the samples in frequency would fall exactly in the roots of D_N.

Fast Fourier Transform (FFT)

An efficient algorithm for computation of DFT in $O(NlogN)$ rather than $O(N^2)$.

Computing the DFT with N samples naively requires N^2 multiplications (since $X^D(k)$ is a sum of N multiplications of $\{x(n)\}$ with $\{e^{ikn}\}$).

However, Fast Fourier Transform exploits the unique structure of the matrix that represents the computation – and in particular the fact that the twiddles $\{e^{ikn}\}$ can be represented as cumulative shifts by frequency – to apply divide-and-conquer approach that reduces the time-complexity of the computation to $NlogN$ multiplications.

Windowing

As explained above, DFT implements bounding in time through multiplication by rectangle, AKA rectangular window, which causes leakage of frequencies.

The leakage has two main properties:

1. **Frequency resolution**, which is determined by the width of the main lobe, which can be reduced by adding more time-samples (i.e. increasing N). Frequency resolution is essential for separate detection of adjacent frequencies.
2. **Magnitude resolution**, which is determined by the height of the side-lobes, and is an inherent property of the rectangular window. Magnitude resolution is essential for detection of weak (yet non-zero) frequencies.

The implementation of DFT can be generalized to finite windows different from the rectangular one. In general, the **various windows achieve various points in the tradeoff between frequency resolution (i.e. how large N is required for a certain width of the main lobe) and magnitude resolution (i.e. height of side lobes)**.

Windows learned in the course:

- **Rectangular window**.
- **Triangular window**: convolution of two rectangles in time, hence multiplication of two Dirichlet-Kernels in frequency ➔ squared magnitude resolution.
- **Hann window**: 0.5*(cosine + rectangle).
- **Hamming window**: similar to Hann, but with weights other than 0.5 & 0.5.
- **Kaiser window**: parametric window with complicated expression that allows control of the frequency resolution vs. magnitude resolution tradeoff.

Periodic (cyclic) convolution

Cyclic convolution which is useful in certain contexts, and in particular allows efficient implementation of standard convolution in certain cases. Recommended to read about in external sources.

Laplace transform (Z-transform)

- Z-transform is a **generalization of Fourier transform for $\boldsymbol{s} \in \boldsymbol{C}$ rather than $\boldsymbol{i\omega} \in \boldsymbol{iR}$**:

$$L\{f\}(s) := \int_0^\infty f(t)e^{-st}dt$$

- Since we don't have $Re(s) = Re(i\omega) = 0$ anymore, and since for any $Re(s) \neq 0$ the exponent diverges rapidly in either ∞ or $-\infty$, then **the transform is defined only over $[\mathbf{0}, \infty)$**.
    - In many practical cases $f \sim e^{-at}$, and the transform converges only for $Re(s) \in (-a, \infty)$ ($\sim \int e^{-(s-a)t}$).
    - Due to the limits of the integration in the transform, **Laplace transform is useful for simplifying PDEs with one-sided bound condition**, whereas Fourier is useful for simplifying PDEs over a whole line (e.g. the heat equation $u_t = u_{xx}$ over the line $x \in R$).
- Laplace transform has many properties similar to Fourier transform. For our purposes note that:
    - Time shifting: $L\{f(t-a)u(t-a)\}(s) = e^{-as}L\{f\}(s)$
        - (u is a step function that removes anything that was shifted into $t < 0$)

Filters

Introduction

A general filter can be defined as an iterative processing of some input signal x, with the form:

$$y_n := \sum_{k=0}^{N_1} a_k x_{n-k} + \sum_{m=1}^{N_2} b_m y_{n-m}$$

where $\sum_{k=0}^{N_1} \boldsymbol{a_k x_{n-k}}$ **is named Moving Average (MA)** and $\sum_{m=1}^{N_2} \boldsymbol{b_m y_{n-m}}$ **is named Auto-Regression (AR)**, and together such a process is known as **ARMA**.

The filter can also be written as:

$$y_n - \sum_{m=1}^{N_2} b_m y_{n-m} = \sum_{k=0}^{N_1} a_k x_{n-k}$$

Any such filter can be represented (using time-shifting of Laplace transform) in the Frequency Space as:

$$\left(1 - \sum_{m=1}^{N_2} b_m e^{-ms}\right) Y = \left(\sum_{k=0}^{N_1} a_k e^{-ks}\right) X$$

Denoting $\boldsymbol{z} := \boldsymbol{e^s}$ (or $z = e^{i\omega}$ for real frequencies) and re-defining the $\{b_m\}$, we have:

$$Y = \frac{\sum_{k=0}^{N_1} a_k z^{-k}}{1 + \sum_{m=1}^{N_2} b_m z^{-m}} X$$

i.e. the filter can be represented in the Frequency Space as $\boldsymbol{Y} = \boldsymbol{HX}$ with $\boldsymbol{H} = \frac{\sum_{k=0}^{N_1} a_k z^{-k}}{1+\sum_{m=1}^{N_2} b_m z^{-m}}$.

- Usually we assume that the coefficients are real and that $N_1 \le N_2$.

Impulse response

A filter $H = \sum a_k z^{-k}$ (i.e. $y_n = \sum a_k x_{n-k}$) is actually described in terms of its response to a single impulse $x_n = \delta(n)$, since we have $y_n = a_n$.

In particular:

- If $b_m \equiv 0$ we have $\boldsymbol{H} = \sum_1^N \boldsymbol{a_k z^{-k}}$, or $\boldsymbol{y_n} = \sum_{k=0}^N \boldsymbol{a_k x_{n-k}}$.
  In such case the response to a single impulse would die out within N time steps, from which comes the name **Finite Impulse Response (FIR)**.
- If there is some $b_m \neq 0$, e.g. $y_n = x_n + by_{n-1}$, then any single impulse $\delta(t - t_0)$ would decay exponentially (e.g. $y(t - t_0) \sim e^{-b(t-t_0)}$), from which comes the name **Infinite Impulse Response (IIR)**.
  - It can also be written as $\boldsymbol{y_n} = \boldsymbol{x_n} + \boldsymbol{b y_{n-1}} = \sum_{k=1}^{\infty} \boldsymbol{b^k x_{n-k}}$.
    ➔ **stable** (finite-energy response to impulse) iff $b < 1$.
  - Equivalently, in the Frequency Space: $\boldsymbol{H} = \sum_{k=1}^{\infty} \boldsymbol{b^k z^{-k}} = \frac{\boldsymbol{1}}{\boldsymbol{1 - bz^{-1}}}$.
    ➔ $b < 1$ iff the pole is within the unit circle.
    - It turns out that the last attribute can be generalized: **a filter is stable iff all its poles are within the unit circle**.

FIR vs. IIR:

- FIR advantages:
  - Linear phase (no frequencies dispersion)
  - Stability (due to finite response)
  - Relatively convenient for analytic research → advanced design methods are available
- IIR advantages:
  - Typically smaller filter order ($max(N_1, N_2)$), which determines both its **computational complexity** and its **time-delay**. This is critical, for example, in control-systems.

Finite Impulse Response (FIR)

- We denote $h[n] \coloneqq a_k$ and receive $H\left(z = e^{i\omega}\right) = \sum_{n=0}^N h[n] e^{-i\omega n}$.
- If we enforce a symmetry/anti-symmetry axis for $h[n]$ in $N/2$, then by taking $e^{-i\omega N/2}$ out of the sum $\boldsymbol{H(\omega)}$, it can be represented by:
  - A real amplitude $\boldsymbol{A(\omega)}$ consisting of:
    - A sum $\boldsymbol{G(\omega)}$ of cosines with real coefficients $g[k]$.
    - Some simple trigonometric function of the frequency $\boldsymbol{F(\omega)}$ (see below).
  - A phase which is linear in ω: $\boldsymbol{\phi(\omega)} = \phi_0 - \tau\omega$.
- The linear phase $-\omega\tau$ assures that each frequency – besides being multiplied by its real amplitude $A(\omega)$ – is delayed by the same time-difference τ:
  - $\cos\left(\omega(t - \tau)\right) = \cos(\omega t - \omega\tau)$.

- The constraint of real amplitude and 2π-periodicity yields $e^{i2\pi\tau} \in R$, hence $2\tau \in Z$, from which are derived the possible variations of $\boldsymbol{F(\omega)}$.
- The constraint of real impulse-response derives $h[n] \in R$, hence $H(\omega)$ is skew-symmetric (apparently some Fourier property), from which is derived $e^{i2\phi_0} = A(\omega)/A(-\omega)$:
  - $e^{i2\phi_0} \rightarrow \boldsymbol{\phi_0 \in \{0, \pi/2\}}$.
  - $\left|\frac{A(\omega)}{A(-\omega)}\right| = 1 \rightarrow A(\omega) = \pm A(\omega)$, with which we don't need the prior symmetry/ anti-symmetry assumption.

- For more detailed explanations, see lecture 15.
- In summary, **the general form of a linear-phase FIR filter** is determined by two factors:
  - The parity of the filter's order ($N\%2 \in \{0,1\}$).
  - The constant phase ($\phi_0 \in \{0, \pi/2\}$).
- In the context of discretely-sampling filter, we denote $\theta \coloneqq T\omega = 2\pi Tf$.

סיכום תכונות

$$H^f(\theta) = A(\theta)\, e^{j\left(\phi_0 - \frac{1}{2}\theta N\right)}$$
$$A(\theta) = F(\theta)\, G(\theta)$$
$$G(\theta) = \sum_{k=0}^{K} g[k] \cos(\theta k)$$

N – סדר המסנן.

Type	I	II	III	IV
Order סדר המסנן	even	odd	even	odd
Symmetry of $h[n]$	Symmetric $h[n]=h[N-n]$	Symmetric	Anti-Symmetric $h[n]=-h[N-n]$	Anti-Symmetric
Symmetry of $A(\theta)$	Symmetric	Symmetric	Anti-Symmetric	Anti-Symmetric
Period of $A(\theta)$	2π	4π	2π	4π
ϕ_o	0	0	0.5π	0.5π
$F(\theta)$ *in* (9.23)	1	$\cos(0.5\theta)$	$\sin\theta$	$\sin(0.5\theta)$
K *in* (9.23)	$N/2$	$(N-1)/2$	$(N-2)/2$	$(N-1)/2$
$g[n]$ *in* (9.23)	See (9.5)	See (9.10)	See (9.16)	See (9.21)
$H^f(0)$	abritrary	abritrary	0	0
$H^f(\pi)$	abritrary	0	0	abritrary
Uses	LP, HP, BP, BS, Multiband filters	LP, BP	Differentiators, Hilbert transformers	

Table 9.1 Properties and parameters of the four FIR filter types

הטבלה מתוך ספרו של B. Porat

Band-Pass filter

- Band-Pass filter (**BP**) keeps a band of frequencies while reducing other frequencies.
  - An ideal BP is the indicating function $\chi_{[f_1,f_2]}$.
- Most common cases are Low-Pass (**LP**, yielded by $f_1 = 0$) and High-Pass (**HP**, yielded by $f_2 = 2\pi$).
- FIR filter can only approximate the ideal BP filter. In practice, any such filter would consist of:
  - **Pass-band**: band of frequencies for which $|H(e^{i\omega}) - 1| < \delta$
  - **Transition-band**: band of frequencies for which $0 < H(e^{i\omega}) < 1$
  - **Stop-band**: band of frequencies for which $|H(e^{i\omega})| < \delta$
- Note:

  - The ideal filter is accepted for $\delta \to 0$ and no transition-band.
  - Shrinking the transition band requires high derivatives in frequency, which require many samples in time (i.e. large N).
- Ideal filter requires non-continuous jump in frequency, and accordingly infinite impulse response. However, this response would have finite energy (from Parseval) and thus can be approximated by the finite **Impulse-Response Truncation** (**IRT**).
  - Note that since the frequencies are orthogonal, then truncating the ideal IIR filter in order N is actually the **optimal FIR filter of order N in terms of L_2-norm**.
  - However, due to **Gibbs effect**, the best L2-approximation of a step-function is not optimal in terms of $||\cdot||_\infty$.
    - A partial solution is gradual truncation of the frequencies using **windowing**.
- **Remez method (Equi-ripple filters)**: **Optimizing a BP filter in terms of $||\cdot||_\infty$** (i.e. minimizing the maximal error out of the transition-band) is achieved using **Chebyshev Polynomials**, which minimize the maximal error using polynomials of N cosines which reach that maximal error exactly N+1 times (or so).
  - The polynomials are typically approximated using numerical iterative methods with random initial guess.

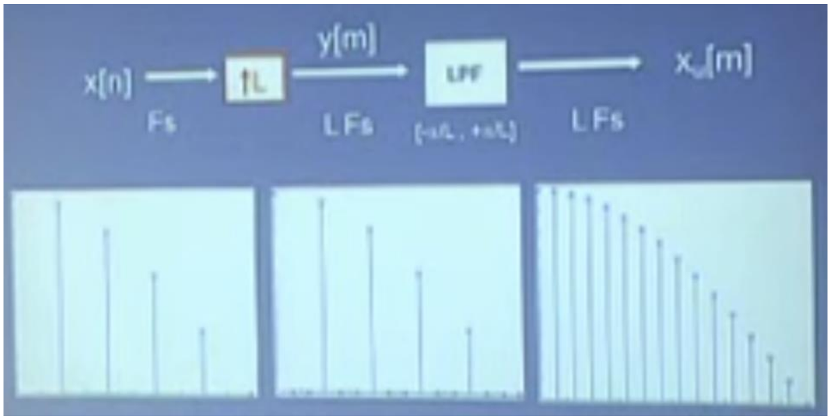
Infinite Impulse Response (IIR)

- In opposed to FIR filters (this difference was not explicitly explained in the course), IIR filters are difficult to design directly in the digital space. Thus, **digital IIR filters are usually designed through conceptual discretization of analog filters**:
  - Requirements are expressed in terms of digital domain (since the filter applies in this domain).
  - The requirements are converted to the analog domain.
  - A theoretical analog filter is designed according to the requirements.
  - The analog filter is transformed into a digital filter.
- **Analog filter** is defined by its continuous response to impulse, or equivalently by its effect on every $s \in C$.
- Popular classes of analog filters:
  - **Butterworth**: basic analog filter.
  - **Chebyshev**: filter with ripples (i.e. non-monotonic behavior) in either the pass-band or the stop-band, and smooth, monotonic flow in the other band.
  - **Elliptical**: filter with ripples in both pass & stop bands, but with narrower transition band.
  - **Bessel**: IIR filter whose phase is intended to be as close to linear as possible.
  - **Kalman**: out of the scope of the course.
- Transformations from analog to digital filters:
  - **Impulse-invariance**: $h[n] \coloneqq Th(nT)$
    - Sampling the impulse-response h is intended to keep the impulse-response.
    - It can be shown that it's vanishes in $z = e^{i\pi}$, hence cannot implement high-pass.
  - **Step-invariance**: keep the response to step (which is important in many applications) by sampling it instead of sampling impulse-response: $h[n] \coloneqq L^{-1}\left\{\frac{H(s)}{s}\right\}(nT)$

- **Bilinear transform (most important in the course)**: $s \leftarrow \frac{2}{T}\frac{z-1}{z+1}$ (replacement of s in the representation of the continuous H)
  - The first two transformations apply sampling of the response of the filter to some input signal, which is essentially approximation of the convolution integral using the standard rectangles of Riemann. The Bilinear transform, however, can be represented as **Trapezoidal approximation of the convolution integral**.
  - Since the Bilinear transform does not involve integral transformations (or equivalently, does not apply sampling in time), it **does not suffer from Aliasing**.
  - The invert transformation is $z = \frac{1+sT/2}{1-sT/2}$, from which one can show that $[|z| < 1$ iff $Re(s) < 0]$ and $[|z| = 1$ iff $Re(s) = 0]$, i.e. **the complete frequency line $\boldsymbol{Re(s) = 0}$ is mapped to the unit circle $\boldsymbol{|z| = 1}$**. In particular, it can be shown that $\boldsymbol{\theta \coloneqq arg(z) = 2\tan^{-1}\frac{\omega T}{2}}$. Hence, **the bilinear transform maps all the frequencies $\boldsymbol{\omega \in R}$ into $\boldsymbol{\theta \in [-\pi, \pi]}$**.
  - Note that for $\omega \ll 1/T$ we have $\theta \approx \omega T$, but for larger ω's we get significant **frequency distortion**. Thus, the conversion of the requirements from digital to analog domain should take it into account in advance (**pre-wrap**): each requirement on a discrete frequency θ is rephrased for $\omega = \frac{2}{T}\tan\frac{\theta}{2}$ (rather than $\omega = \theta/T$).

Multi-rate digital signal processing

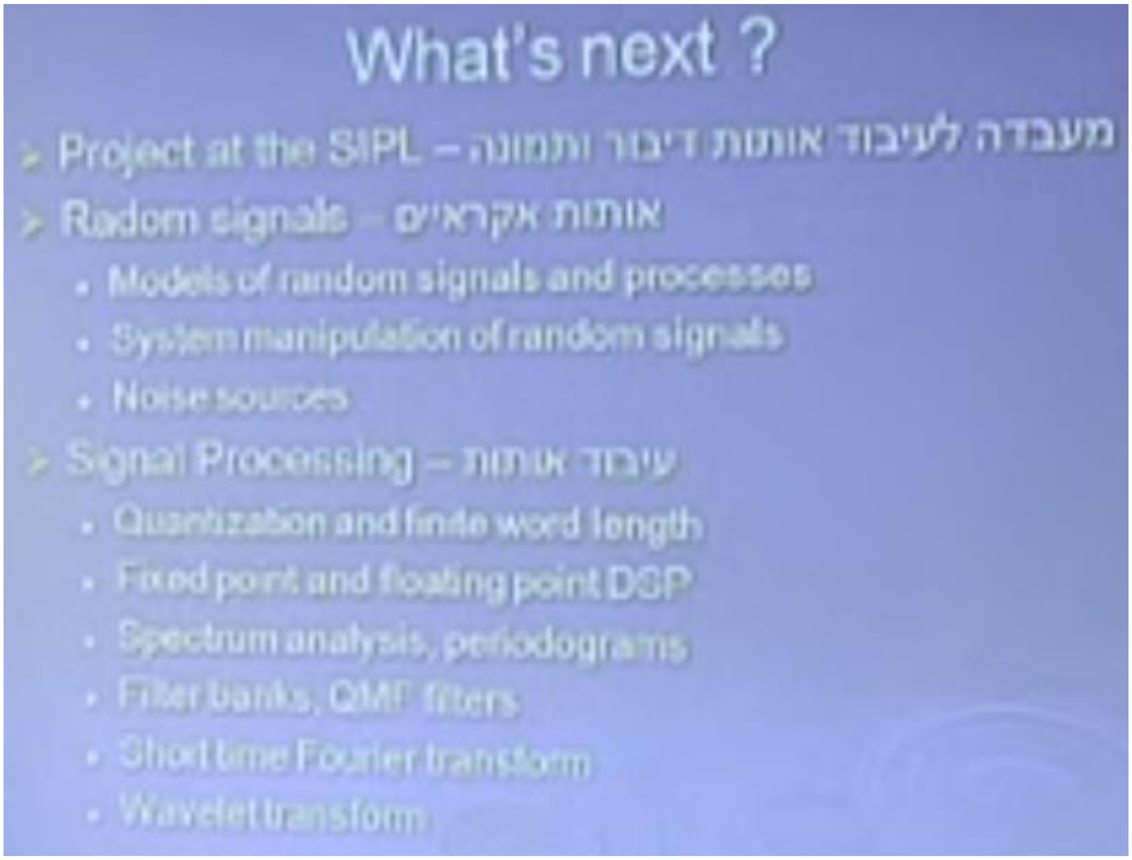
- **Change of sampling frequency from $\boldsymbol{f_1}$ to $\boldsymbol{f_2}$** can be implemented by approximating $\frac{f_2}{f_1} \approx q = m/n \in Q$, then upsampling and downsampling: $f_1 \to mf_1 \to \frac{mf_1}{n} \approx f_2$.
- **Upsampling (*interpolation*)** by integer factor:
  - Implementation:
    - **Upsampling**: add artificial zero samples (e.g. x1, 0, 0, x2, 0, 0, x3...).
    - **Reconstruction**: apply digital low pass filter.
      - Since LPF is essentially a discrete convolution with sync, it turns out that the LPF keeps the original non-zero samples and reconstructs the missing signal in the zero samples – just as in sampling & reconstruction.
      - Since ideal LPF requires infinite convolution and hence is not causal, in practice an approximating FIR or IIR filter is used.
  - The figure below demonstrates an original signal, the artificial upsampling and the reconstructed upsampled signal.

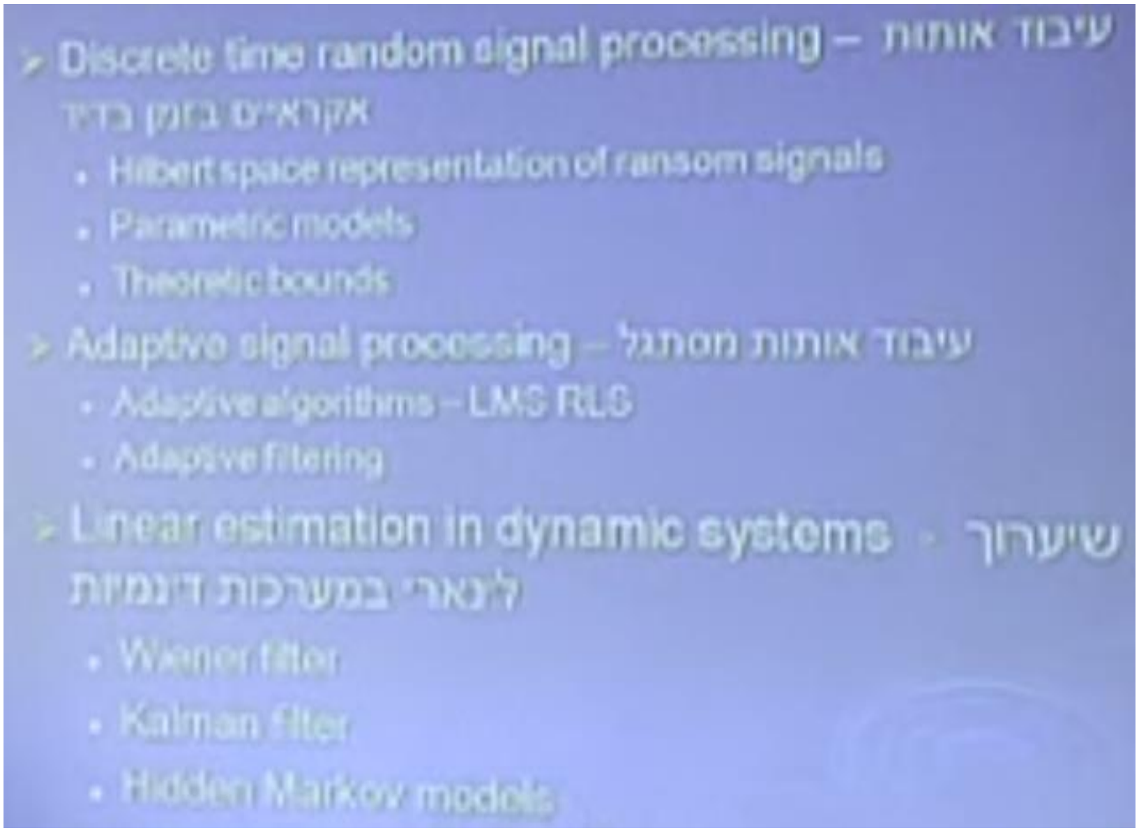


- Note: upsampler is linear, but not time-homogeneous.
- Note: the upsampling naturally **doesn't generate new information**, but rather reconstructs the "simplest" interpolation (in the notion of zeroized high frequencies).

- **Downsampling (*decimation*)** by integer factor:
  - Downsampling **decreases the Nyquist frequency** of the digital system, thus an input signal spectrum should correspond in advance to the Nyquist frequency of the downsampled system.
  - **Aliasing is prevented** by LPF that filters out the high frequencies in advance (and naturally **loses any information of frequencies higher than the new Nyquist**).
  - Implementation:
    - **Aliasing prevention**: apply low pass filter.
    - **Compression**: delete the unwanted samples (e.g. keep only x1, x4, x7, x10…).
- Both upsampling and downsampling involve computations in dimension higher than the "essential dimension" of the information carried by the output signal. Accordingly, it can be shown that many computations can be saved by appropriate design, as in the following examples:
  - Simple careful design of the order of the filter's components can directly spare many degenerated computations (e.g. multiplications by zero).
  - **Multi-stage**: decomposition of the sampling-rate conversion into two sequential processes (e.g. X50 X2 instead of X100) turns out to allow significant reduction of the computational complexity.
    - Furthermore, concatenating interpolation-and-then-decimation before a narrow-band (thus computationally-complex) filter can significantly reduce the complexity of the filter on account of extended transition band.
  - **Poly-phase** filters can implement efficient sampling-rate conversion through transforming N various frequencies into M frequencies with N/M various phases, and then applying the computation in the low-frequency (M) digital space.
- **Oversampling**: it is possible to use in advance higher sampling frequencies than required.
  - In IIR filters, it allows increase of the order of the filter and reduction of the noise without increasing the time delay.
  - It also allows **dynamic choice of the sampling frequency without losing information**.
    - For example, video sampling of 150hz allows watching both American movies with frame rate of 30hz, and European movies with 25hz.

List of advanced topics




Discrete time random signal processing – עיבוד אותות אקראיים בזמן בדיד
Hilbert space representation of ransom signals
Parametric models
Theoretic bounds
Adaptive signal processing – עיבוד אותות מסתגל
Adaptive algorithms – LMS RLS
Adaptive filtering
Linear estimation in dynamic systems - שיערוך לינארי במערכות דינמיות
Wiener filter
Kalman filter
Hidden Markov models