

2024年度 複雑理工学実験概論 計測コース 生体計測グループ 第1回

篠田・牧野研究室 特任助教

鈴木 颯

2024年6月7日

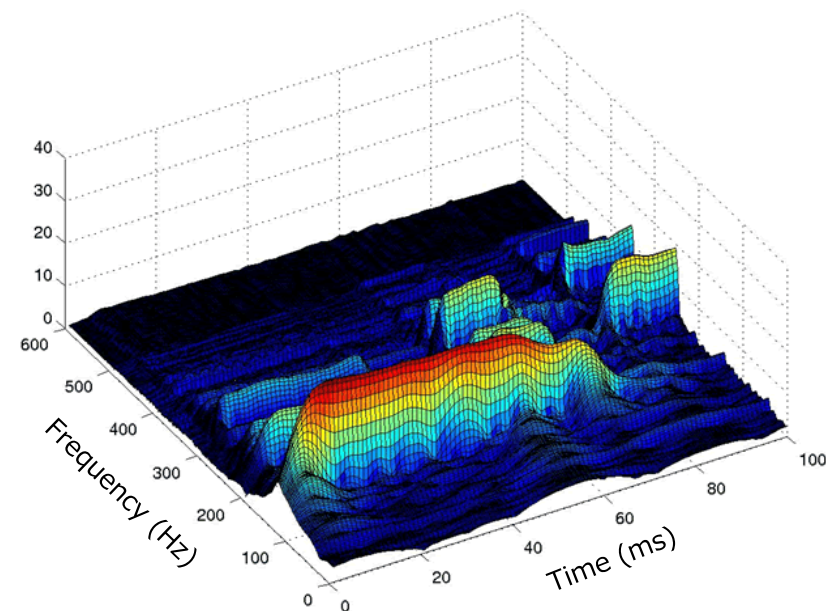
目次

1. 概要
2. 演習環境の準備
3. フーリエ変換
4. LTIシステム
5. フィルタの設計
6. フィルタの評価
7. まとめ

1. 概要

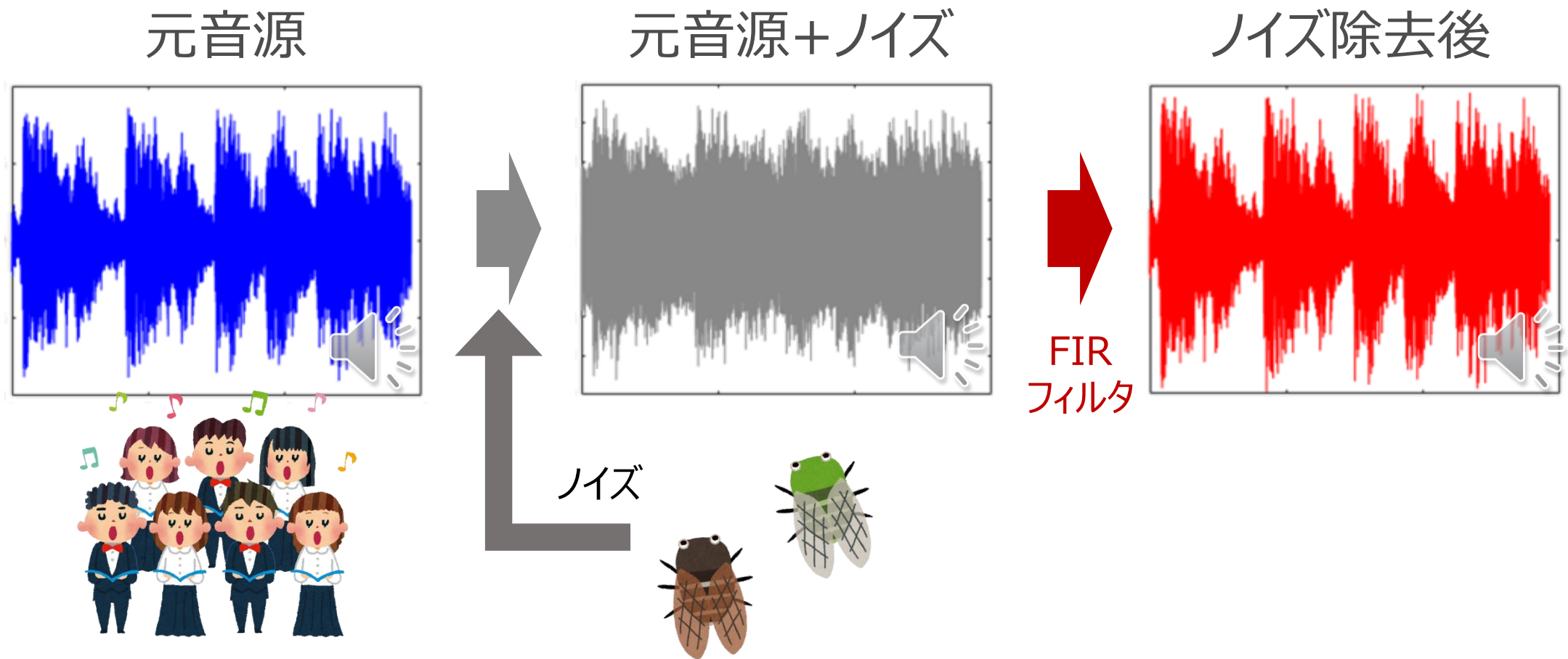
概要

- 「デジタル信号処理」の基礎を学習
 - 一例として音声・音響信号を対象にする
 - 信号の**周波数特性**を利用して情報を抽出する
 - FIRフィルタの設計 ← 今日
 - 信号の**統計的情報**を利用して情報を抽出する
 - 独立成分分析によるブラインド音源分離 ← 次回
- 演習主体
 - 実際にプログラミングして知識を身につける



何ができるようになるか

- FIRフィルタによるノイズ除去



今日の目標

- 時系列信号に対する「デジタルフィルタ」を設計できるようになる
 - 一例として音声・音響信号を対象にする
 - 動作が比較的予測しやすい FIR型を設計する

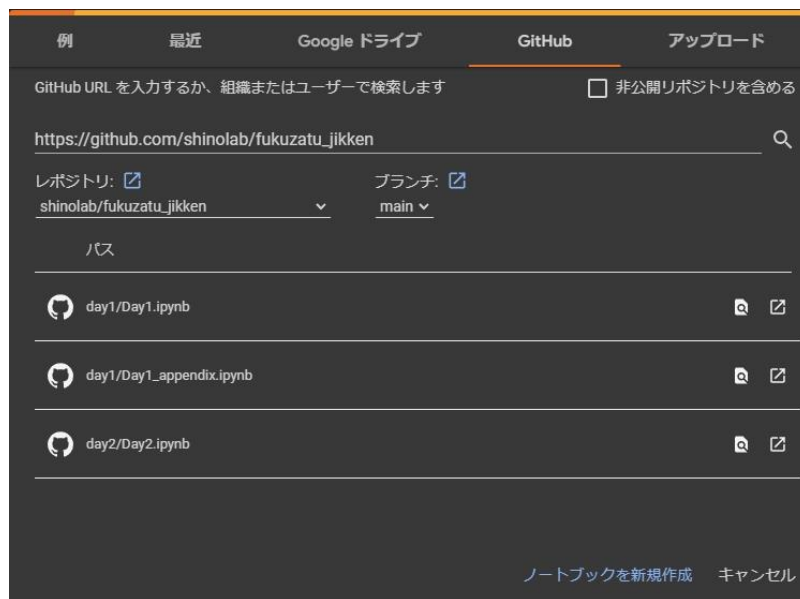
2. 演習環境の準備

Google Colaboratoryの紹介

- 本講義はGoogle Colaboratory 上のPythonを使用します
 - Jupyter Notebookを実行できるならローカルでも構いません
- Google Colaboratoryはブラウザ上でPythonのコーディング&実行が行えるツールです
 - 基本使用は無料です
 - Googleアカウントが必要になります

演習環境の準備

- <https://colab.research.google.com/> にアクセスし, 「GitHub」 タブの検索欄に「https://github.com/shinolab/fukuzatu_jikken」を入力し, 「Day1.ipynb」を選択してください。



- または, <https://colab.research.google.com/drive/1Jzv2zCACvyopg-a6PKJC7IdFN4EGvUZv?usp=sharing>

3. フーリエ変換

信号の解析

フーリエ変換とは

すべての関数は、
異なる周期の正弦波の
重ね合わせに分解できる！



Joseph Fourier
(1768~1830)

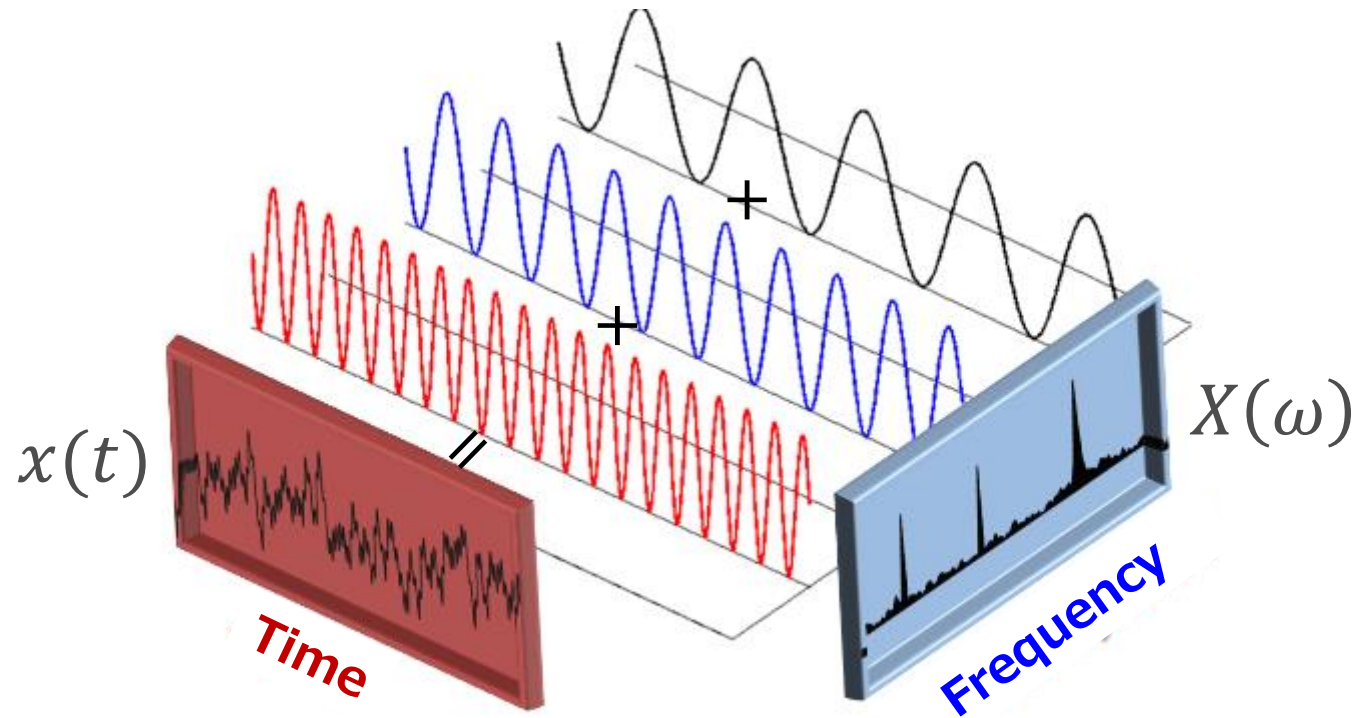
- 信号の「周波数表現」
 - FT (Fourier Transform)
- 連続時間信号 $x(t)$ のフーリエ変換は以下のとおり

フーリエ変換
$$X(\omega) = \mathcal{F}[x(t)] = \int_{-\infty}^{\infty} x(t)e^{-i\omega t} dt$$

逆フーリエ変換
$$x(t) = \mathcal{F}^{-1}[X(\omega)] = \frac{1}{2\pi} \int_{-\infty}^{\infty} X(\omega)e^{i\omega t} d\omega$$

フーリエ変換のイメージ

- 信号を別の側面から見る

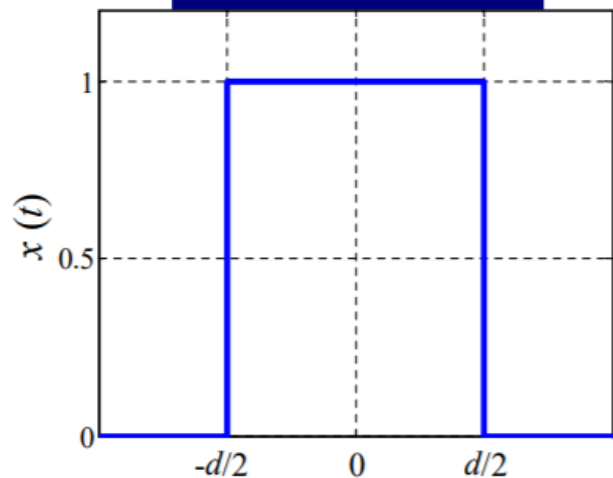


信号の時間表現

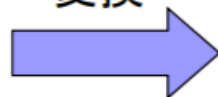
信号の周波数表現

フーリエ変換の例

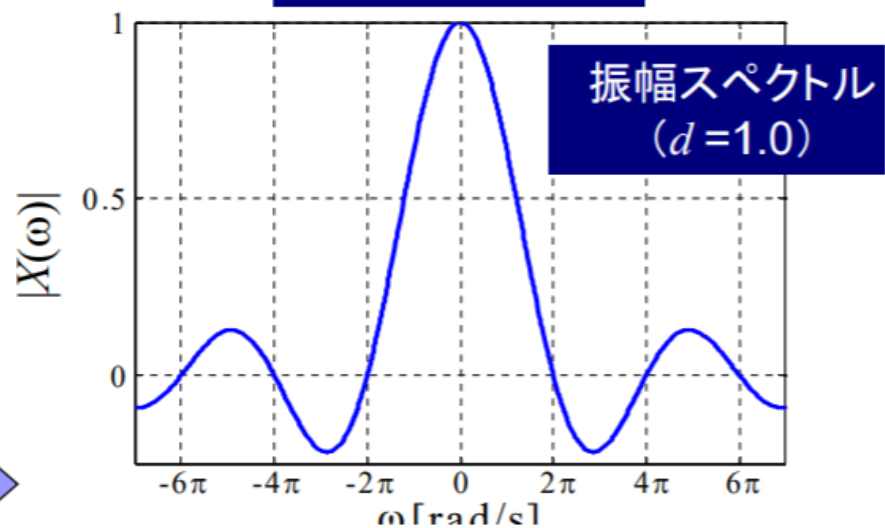
時間領域表現



フーリエ
変換



周波数領域表現



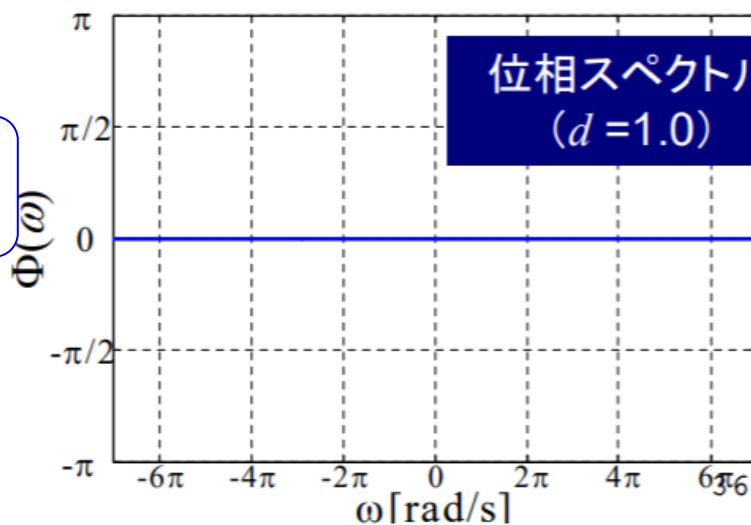
$$x(t) = \begin{cases} 1 & \left(|t| \leq \frac{d}{2} \right) \\ 0 & \left(|t| > \frac{d}{2} \right) \end{cases}$$

$$\text{sinc}(x) := \frac{\sin x}{x}$$

のフーリエ変換 $X(\omega)$ は

$$X(\omega) = d \cdot \frac{\sin(\omega d / 2)}{\omega d / 2} = d \cdot \text{sinc}\left(\frac{\omega d}{2}\right)$$

位相スペクトル
($d=1.0$)



フーリエ変換の性質

- 線形性

$$\mathcal{F}[af(t) + bg(t)] = a\mathcal{F}[f(t)] + b\mathcal{F}[g(t)]$$

- 畳み込み定理

$$\mathcal{F}[f(t) * g(t)] = \mathcal{F}[f(t)]\mathcal{F}[g(t)]$$

$$f(t) * g(t) = \int_{-\infty}^{\infty} f(t')g(t - t')dt' = \int_{-\infty}^{\infty} f(t - t')g(t')dt'$$

- 純実偶関数のフーリエ変換は純実偶関数
- 純実奇関数のフーリエ変換は純実奇関数

実関数のフーリエ変換

- 実関数のフーリエ変換は, その実部が「偶関数」 虚部が「奇関数」になる

$$\left(\begin{array}{l} x_{\text{even}}(t) = \left[\frac{x(t) + x(-t)}{2} \right] : \text{偶関数} \\ x_{\text{odd}}(t) = \left[\frac{x(t) - x(-t)}{2} \right] : \text{奇関数} \end{array} \right)$$

$$\begin{array}{ccc} & x(t) = x_{\text{even}}(t) + x_{\text{odd}}(t) & \\ \text{FT} \curvearrowright & & \curvearrowleft \text{IFT} \\ & X(\omega) = X_{\text{even}}(\omega) + iX_{\text{odd}}(\omega) & \end{array}$$

- $x(t)$ が実関数になる必要十分条件は, $X(\omega)$ の実部が「偶関数」, 虚部が「奇関数」であること
→ デジタルフィルタの設計で重要

離散時間フーリエ変換

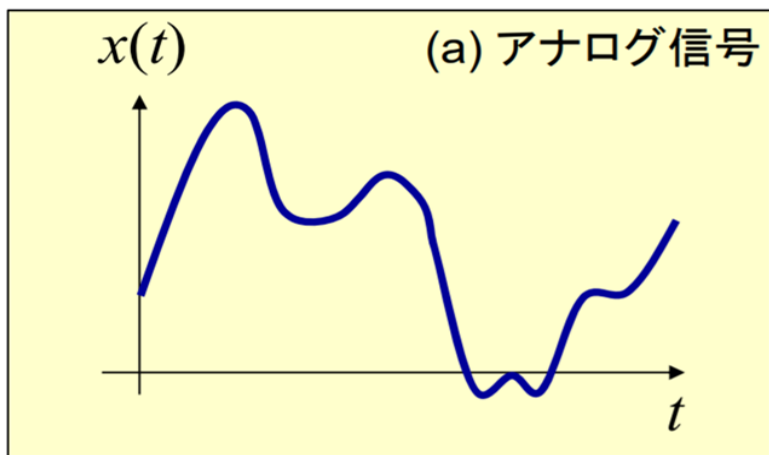
- 離散時間信号に拡張したフーリエ変換
 - DTFT (Discrete-Time Fourier Transform)
- 離散時間信号 $x[n] := x(Tn)$, (T は「サンプリング周期」, n は整数) に対しては以下の通り

離散時間フーリエ変換 $X(\Omega) = \text{DTFT}\{x[n]\} = \sum_{n=-\infty}^{\infty} x[n]e^{-i\Omega n}$

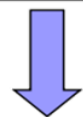
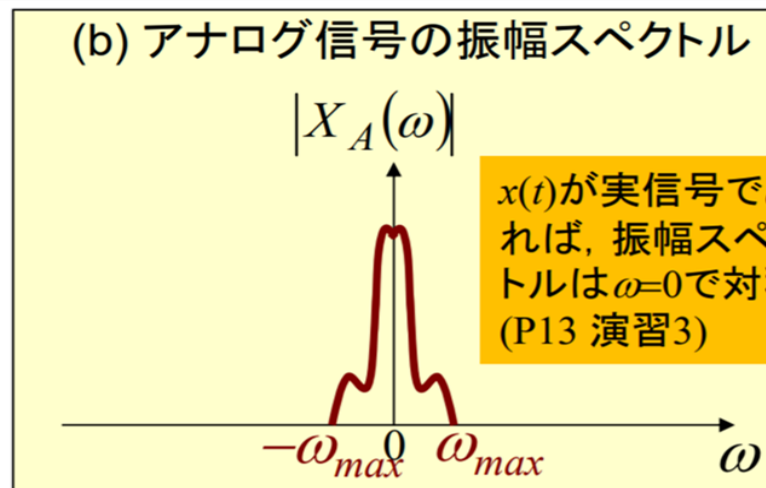
逆離散時間フーリエ変換 $x[n] = \text{DTFT}^{-1}\{X(\Omega)\} = \frac{1}{2\pi} \int_{-\pi}^{\pi} X(\Omega)e^{in\Omega} d\Omega$

($X(\Omega)$ は基本周期 2π の周期関数, $\Omega (= T\omega)$ は正規化周波数)

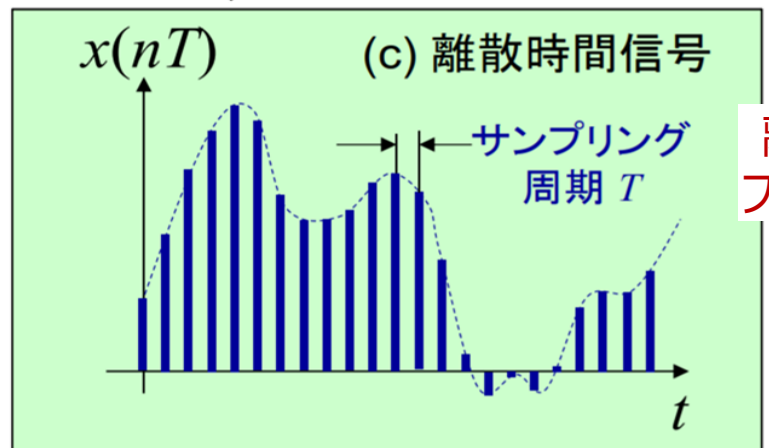
フーリエ変換と離散時間フーリエ変換



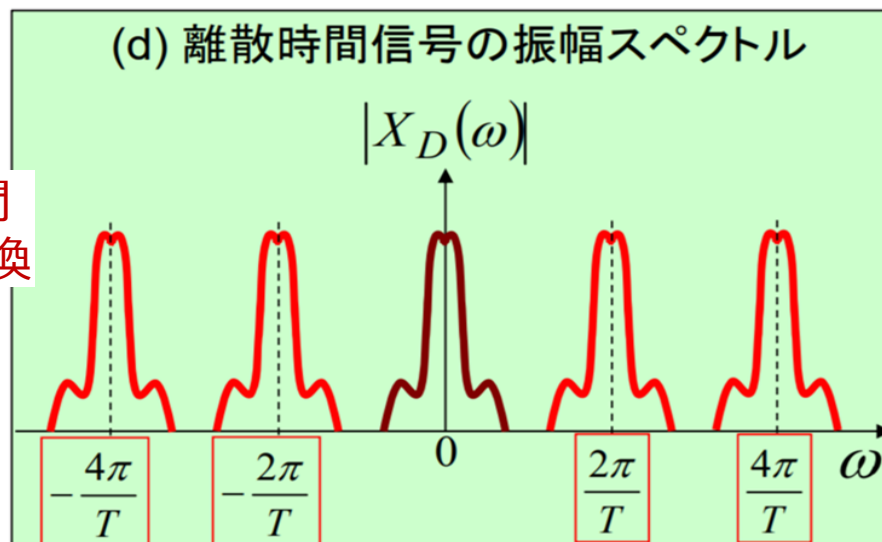
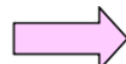
フーリエ
変換



周期 T でサンプリング



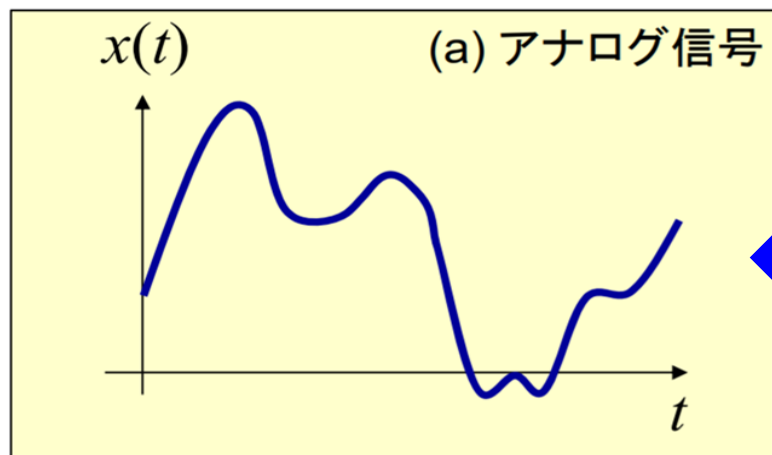
離散時間
フーリエ変換



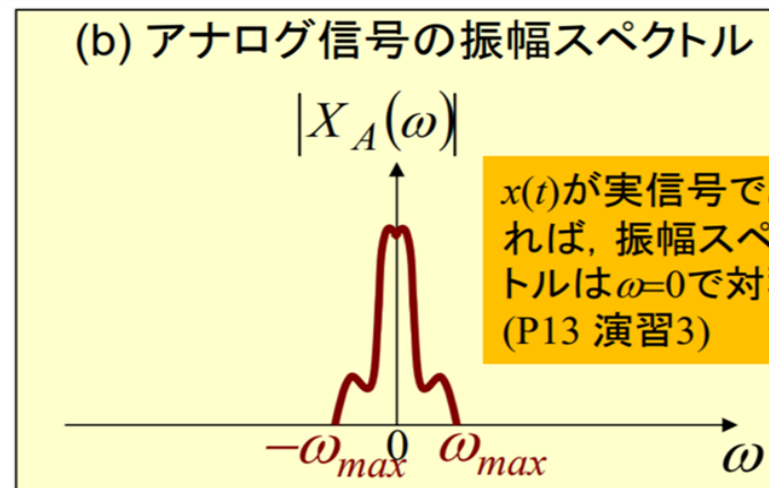
北海道大学、金井先生
『信号処理』
<http://sdmwww.ssi.ist.hokudai.ac.jp/lecture/signal/aboutus1.html>

$$\omega = \Omega/T$$

サンプリング定理

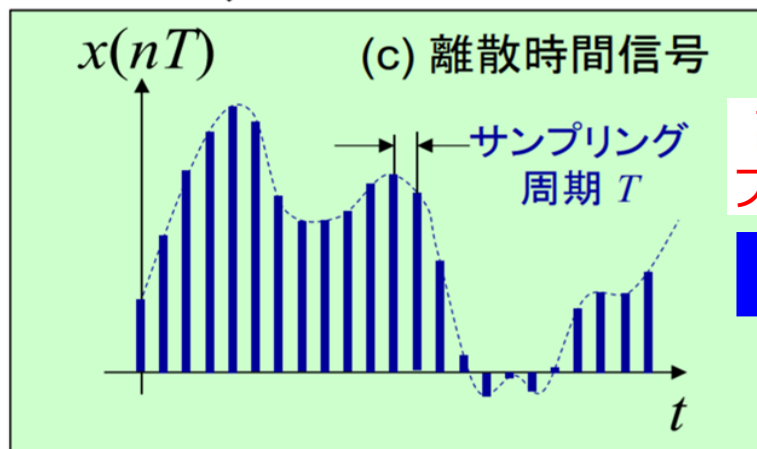


逆フーリエ
変換

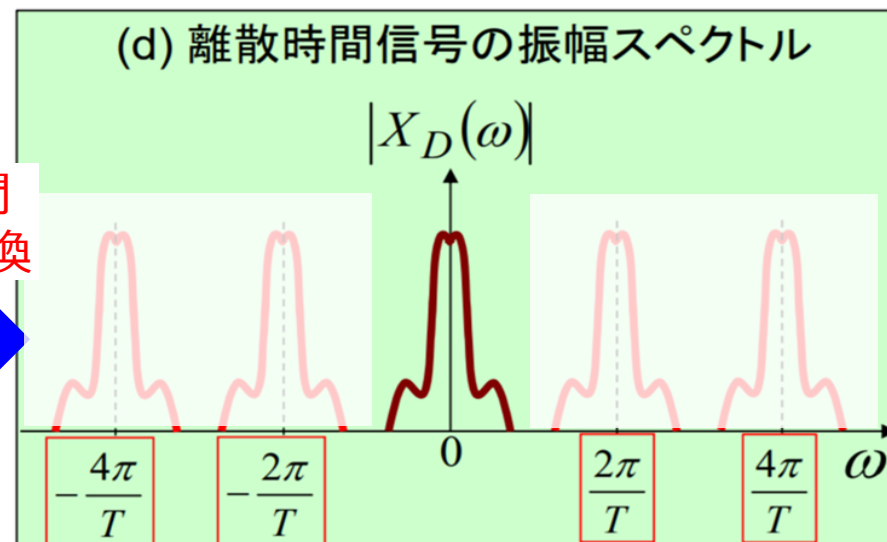


$\omega_{max} < \frac{\pi}{T}$ なら
離散時間信号からアナログ信号を
完全に復元可能

周期 T でサンプリング



離散時間
フーリエ変換



北海道大学、金井先生
『信号処理』
<http://sdmwww.ssi.ist.hokudai.ac.jp/lecture/signal/aboutus1.html>

$$\omega = \Omega/T$$

4. LTIシステム

最も基本的な「システム」

LTIシステムとは

- **線形性と時不変性**をもつシステム

- LTI (Linear & Time-Invariant, 線形時不変)

- ここではディジタルシステムを扱う (アナログシステムでも基本は同じ)

- 時刻パラメータが離散的な値 (整数 n)

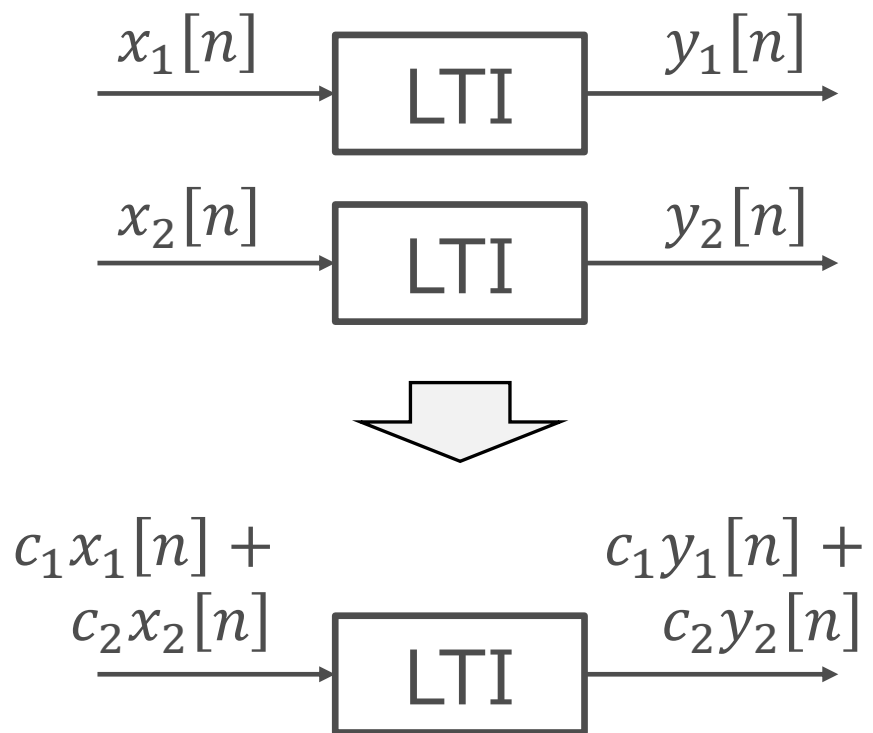


※「 $\dots, x[n-1], x[n], x[n+1], \dots$ 」の略記

→ 入力 $\{x[m]\}_{m=-\infty}^{\infty}$ に対して, 離散時刻 n の出力 $y[n]$ がただ一つ定まる

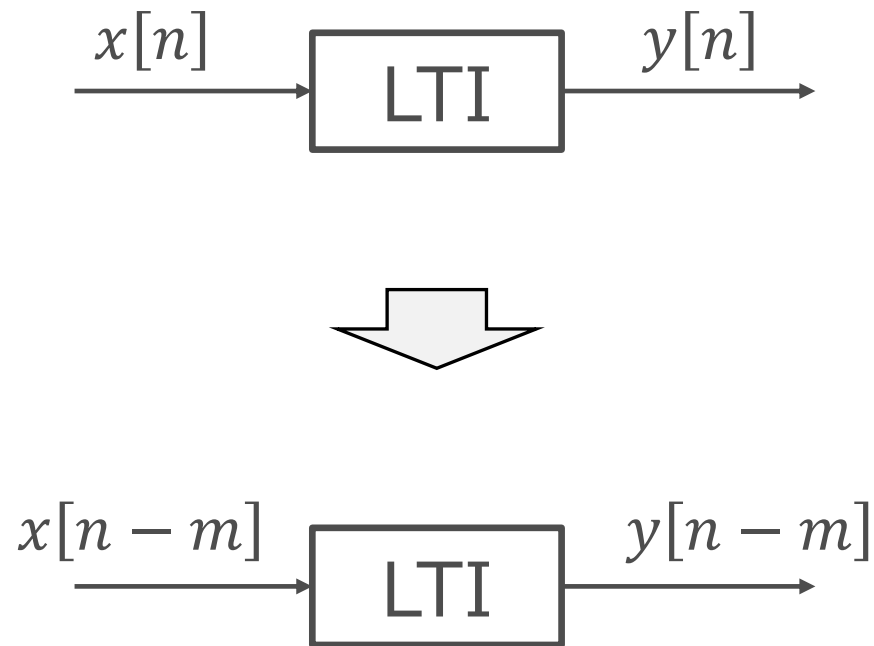
線形性と時不変性

- 線形性 (linearity)



※ c_1, c_2 は定数

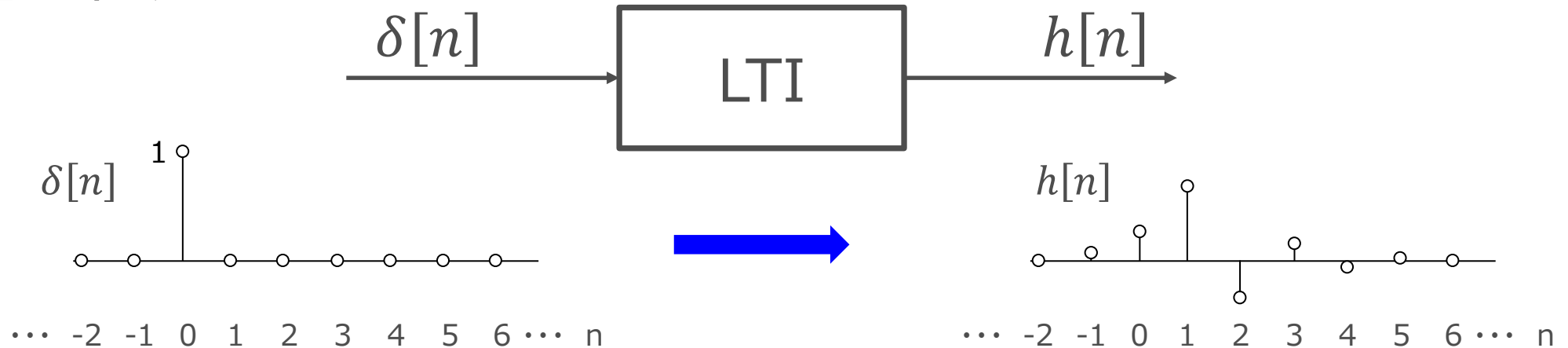
- 時不変性 (time-invariant)



※ m は任意の整数

LTIシステムの最大の特徴

- 「インパルス応答」で一意に特徴づけられる
- インパルス応答 $h[n]$ とは, 「インパルス信号 $\delta[n]$ 」を入力したときの出力

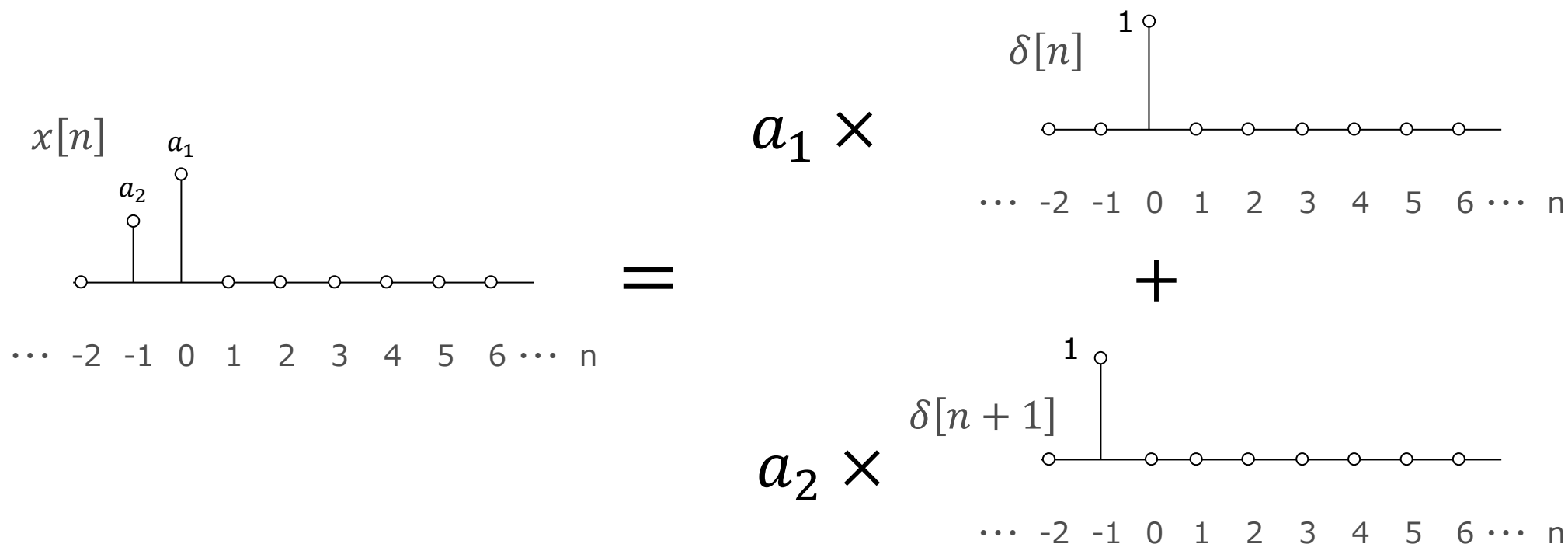


- インパルス応答 $h[n]$ から, 任意の入力 $x[n]$ に対する出力 $y[n]$ が計算可能

インパルス応答の重ね合わせ

- 任意の信号 $x[n]$ はインパルス入力の線形重ね合わせで表される

$$\text{e.g., } x[n] = a_1 \delta[n] + a_2 \delta[n + 1]$$



LTIシステムの入力

- 信号 $x[n] = a_1\delta[n] + a_2\delta[n+1]$ をインパルス応答 $h[n]$ のLTIに入力すると...

$$\begin{aligned}y[n] &= a_1 h[n] + a_2 h[n+1] \\&= x[0] h[n] + x[-1] h[n+1] \\&= \sum_{m=-1}^0 h[n-m] x[m]\end{aligned}$$

- より一般に

$$y[n] = \sum_{m=-\infty}^{\infty} h[n-m] x[m] = (h * x)[n]$$

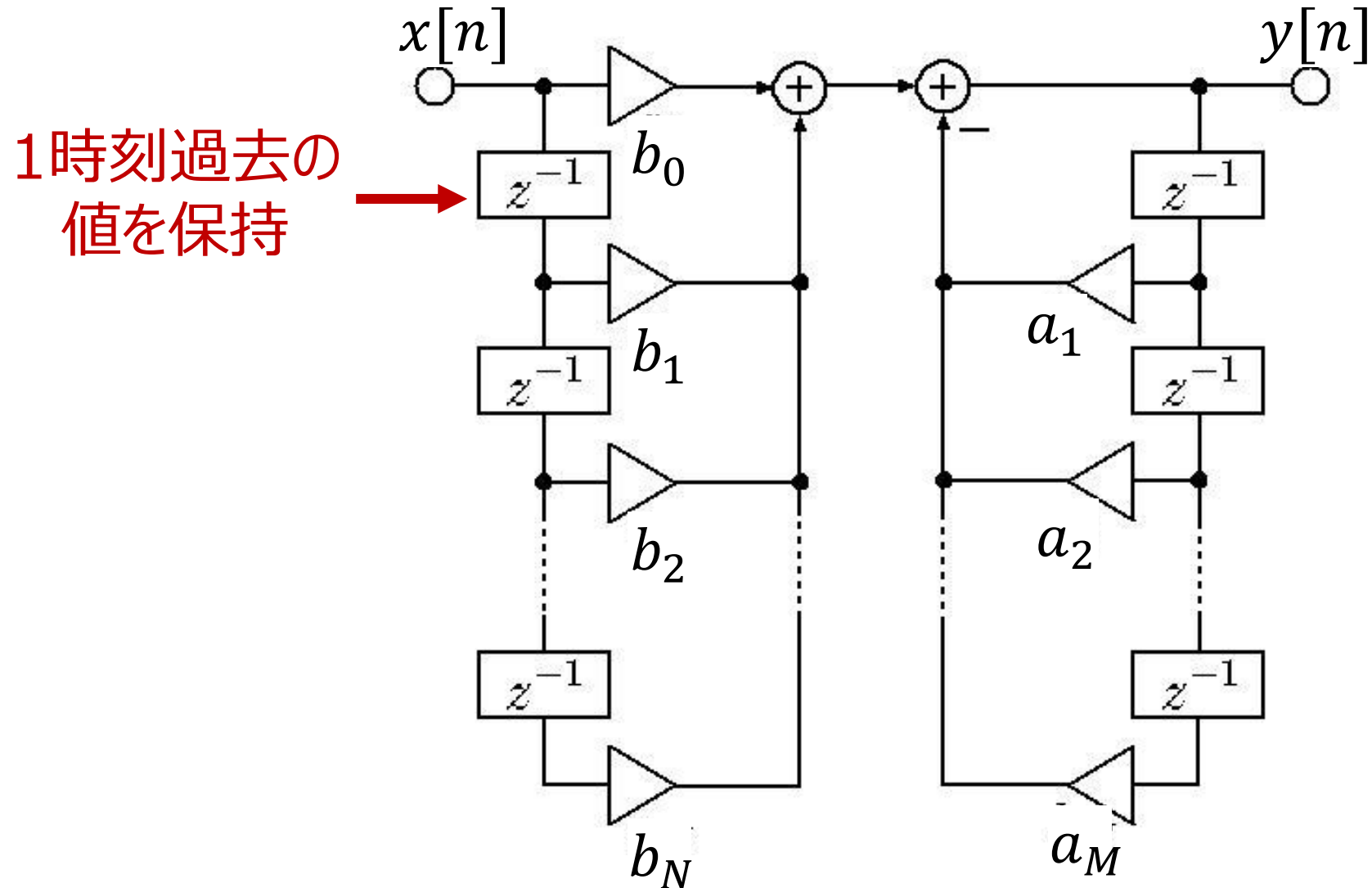
畳み込み

LTIシステムの各種表現

表現	例
インパルス応答	$h[n], n = \dots, -1, 0, 1, 2, \dots$
周波数応答	$H(\Omega) = \text{DTFT}\{h[n]\}, \quad \Omega < \pi$
伝達関数	$H(z) = \text{ZT}\{h[n]\} = \frac{b_0 + b_1 z^{-1} + \dots + b_N z^{-N}}{1 + a_1 z^{-1} + a_2 z^{-2} + \dots + a_M z^{-M}}, \quad z \in \mathbb{C}$
極・零点集合	$\{p_i\}, \{q_i\}$ s. t. $H(z) = \frac{b_0(1 - q_1 z^{-1})(1 - q_2 z^{-1}) \dots (1 - q_N z^{-N})}{(1 - p_1 z^{-1})(1 - p_2 z^{-1}) \dots (1 - p_M z^{-M})}$
差分方程式	$y[n] + a_1 y[n-1] + a_2 y[n-2] + \dots + a_M y[n-M]$ $= b_0 x[n] + b_1 x[n-1] + \dots + b_N x[n-N]$
ブロック線図	差分方程式に対応（次ページ参照）

フィルタ設計で重要

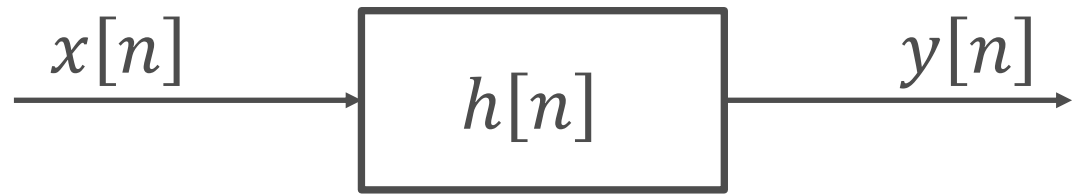
LTIシステムのブロック線図表現例



LTIシステムの出力（周波数領域表現）

- 「畳み込み定理」より出力信号の周波数特性 $Y(\Omega)$ は, LTIシステムの周波数応答 $H(\Omega)$ と入力信号の周波数特性 $X(\Omega)$ の積になる

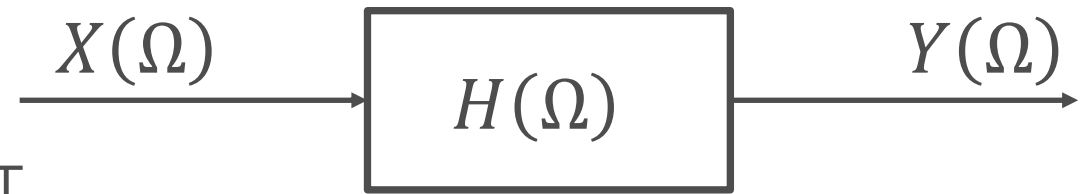
時間領域表現



$$y[n] = (h * x)[n]$$

$$= \sum_{m=-\infty}^{\infty} h[n-m]x[m]$$

周波数領域表現



$$Y(\Omega) = H(\Omega)X(\Omega)$$

ただし、 $X(\Omega) = \text{DTFT}\{x[n]\}$

$$H(\Omega) = \text{DTFT}\{h[n]\}$$

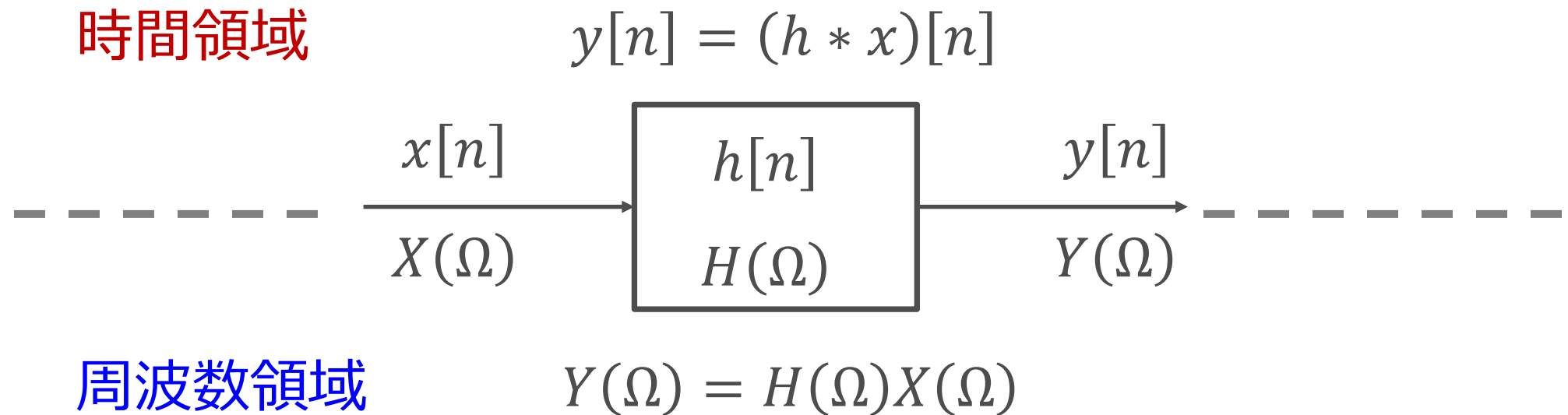
$$Y(\Omega) = \text{DTFT}\{y[n]\}$$

DTFT

IDTFT

信号処理における重要な問題

- 信号 $x[n]$ の特定周波数成分を阻止または通過させるLTIシステム $h[n]$ は設計できるか？



5. フィルタの設計

信号の加工

フィルタとは

- 信号に何らかの加工を行うシステム
- 今回は**線形時不変** (LTI) なデジタルフィルタを扱う
 - つまり, 以後「フィルタ」は「LTIシステム」と同義
- さらに, インパルス応答 $h[n]$ が**因果的**で有限長である**FIRフィルタ**を設計する
 - 因果的 (causal) : $h[n] = 0, (n < 0)$
 - FIR: Finite Impulse Response (対義語は IIR: Infinite ~)

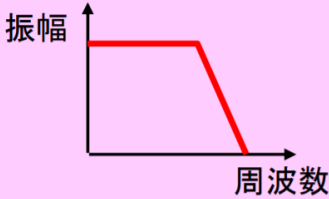
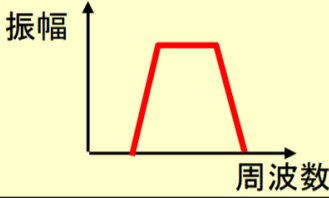
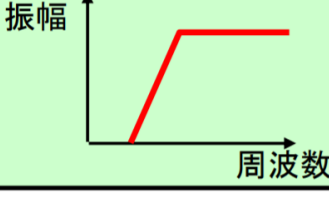


FIRフィルタとIIRフィルタの比較

	FIRフィルタ	IIRフィルタ
安定性	常に安定	安定とはならない場合がある.
伝達関数の次数	高い	低い
実現方法	有限個の遅延器、乗算器、加算器で実現可能	フィードバックをもつ有限個の要素で実現可能
直線位相特性	完全に実現可能	実現が困難

周波数選択フィルタ

- 信号の特定周波数成分の阻止, 通過, 増幅, 減衰などを行うフィルタ

特性	フィルタ名称	
	通過周波数帯域を基準とした場合	除去周波数帯域を基準とした場合
	低域通過フィルタ (Low Pass Filter, LPF)	高域除去フィルタ (High Cut Filter, HCF)
	帯域通過フィルタ (Band Pass Filter, BPF)	帯域除去フィルタ (Band Elimination Filter, BEF)
	高域通過フィルタ (High Pass Filter, HPF)	低域除去フィルタ (Low Cut Filter, LCF)

この他に, 全域通過フィルタ
(All Pass Filter, APF)
などもある

FIRフィルタを設計するには

- 所望の特性をもつフィルタの有限長インパルス応答 $\bar{h}[n]$ を求めればよい
 - 出力信号はインパルス応答を入力信号に畳み込むことで得られる
- (例) LPFの設計手順
 1. フィルタの周波数特性 $H(\Omega)$ を決める ($Y(\Omega) = H(\Omega)X(\Omega)$)
 2. 逆離散時間フーリエ変換で (無限長) インパルス応答 $h[n]$ を求める
 3. $h[n]$ を「フィルタ長 L 」で切り詰める
 4. 因果性を満たすように位相特性を変更する ← **今回はここまで**
 5. (切り詰めたインパルス応答に「窓関数 $w[n]$ 」をかける)

LPFの周波数特性

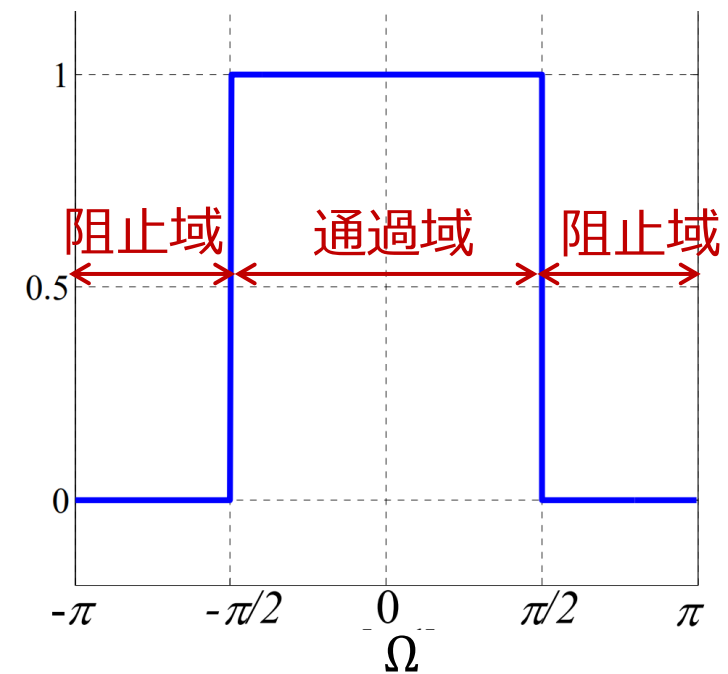
- 決めるべきは遮断周波数 (cutoff frequency) Ω_c
 - 物理的な遮断角周波数 ω_c に対し, $\Omega_c = T\omega_c$ の関係 (c.f. サンプリング定理)

- 所望の周波数特性 $H(\Omega) = \begin{cases} 1, & |\Omega| < \Omega_c \\ 0, & |\Omega| \geq \Omega_c \end{cases}$
 - インパルス応答を実数にするため, 純実偶関数

- 無限長インパルス応答

$$\begin{aligned} h[n] &= \text{DTFT}^{-1}\{H(\Omega)\} = \frac{1}{2\pi} \int_{-\pi}^{\pi} H(\Omega) e^{in\Omega} d\Omega \\ &= \frac{\Omega_c}{\pi} \text{sinc}(\Omega_c n) \end{aligned}$$

遮断周波数 $\Omega_c = \pi/2$ の場合



インパルス応答の切り詰め

- コンピュータで処理できるように、フィルタ長（インパルス応答が非0である長さ）を有限の値 L にする

$$h[n] = \frac{\Omega_c}{\pi} \text{sinc}(\Omega_c n)$$

$$\hat{h}[n] = \begin{cases} \frac{\Omega_c}{\pi} \text{sinc}(\Omega_c n), & (n \text{ が以下の範囲}) \\ 0, & (\text{otherwise}) \end{cases}$$

L が偶数:

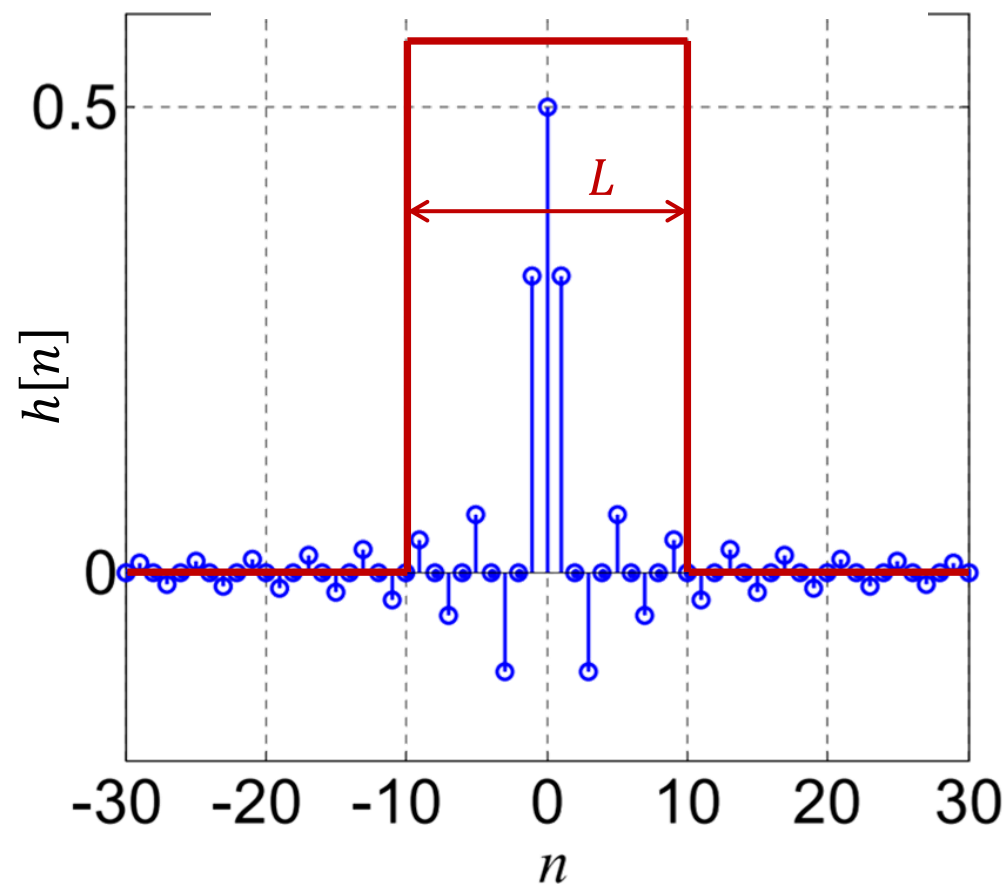
$$n = -L/2, \dots, -1, 0, 1, \dots, L/2 - 1$$

L が奇数:

$$n = -(L-1)/2, \dots, -1, 0, 1, \dots, (L-1)/2$$

※ L が長いほどフィルタ特性が良くなるが、計算時間が長くなる

遮断周波数 $\Omega_c = \pi/2$ の場合

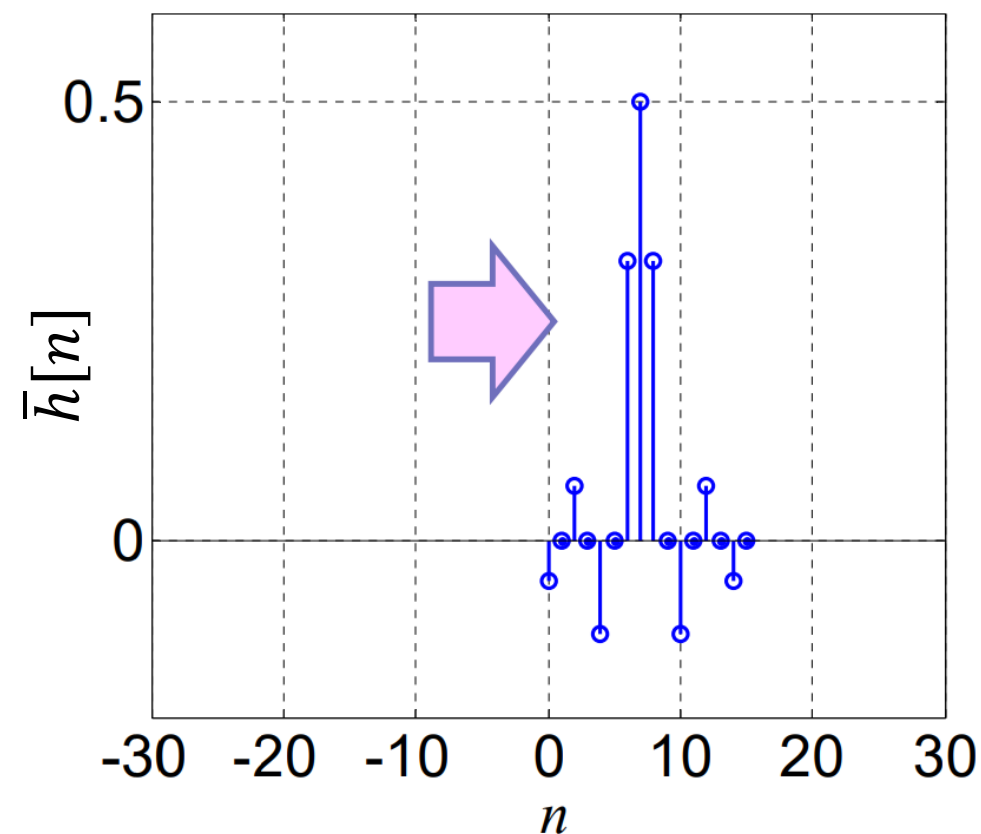


位相特性の変更

- 因果性 ($h[n] = 0, (n < 0)$) を満たすように $\hat{h}[n]$ をスライドさせる
 - DTFTの性質 (時間におけるシフト) により, フィルタの位相特性が線形に変化 (振幅特性には無関係)

$$\bar{h}[n] = \begin{cases} \hat{h}[n - L/2], & (L \text{ が偶数}) \\ \hat{h}[n - (L - 1)/2], & (L \text{ が奇数}) \end{cases}$$

↑
完成 !



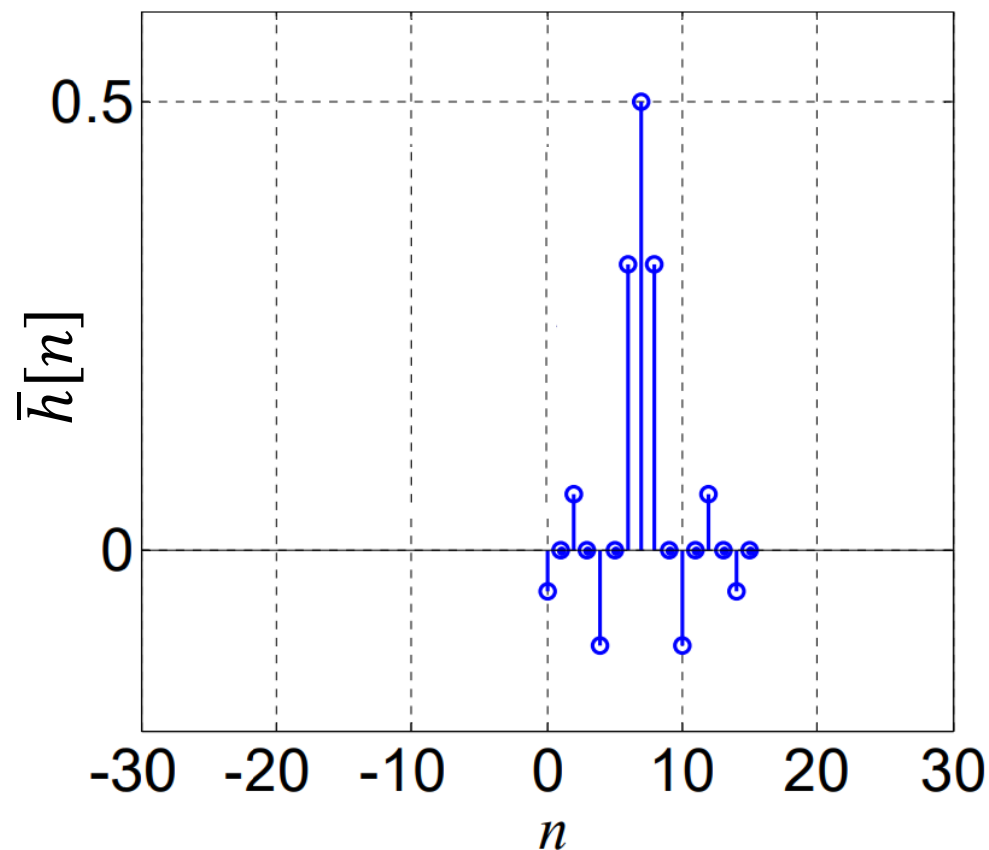
6. フィルタの評価

実データへの適用

設計したフィルタの適用

- 任意の入力信号 $x[n]$ に対し、下記の畳み込みでLPFをかけられる！

$$y[n] = (\bar{h} * x)[n] = \sum_{m=0}^{L-1} \bar{h}[m]x[m-n]$$



演習1

- Notebook上の「**lpf**」関数の中身を埋めてローパスフィルタを完成させてみよう
 - なお, numpyのsinc関数は本資料と異なり, 以下の定義を採用しているので注意: $\text{np.sinc}(x) = \frac{\sin \pi x}{\pi x}$
- ヒント
 - LPFフィルタ係数は右
 - 正規化周波数 $\Omega (= T\omega)$
 - 周期 $T = 1/f_s$, サンプルング周波数 f_s
- 所望の特性が得られているか?
 - 得られていなければ, なぜ得られていないか考えよう

$$\hat{h}[n] = \begin{cases} \frac{\Omega_c}{\pi} \text{sinc}(\Omega_c n), & (n \text{ が以下の範囲}) \\ 0, & (\text{otherwise}) \end{cases}$$

L が偶数:

$$n = -L/2, \dots, -1, 0, 1, \dots, L/2 - 1$$

L が奇数:

$$n = -(L-1)/2, \dots, -1, 0, 1, \dots, (L-1)/2$$

演習1解答例

```
def lpf(L, fc, fs):  
    fir = np.zeros(L, dtype=float)  
    T = 1/fs  
    Omega_c = T * 2 * np.pi * fc  
    for n in range(-(L-1)//2, (L-1)//2 + 1):  
        fir[n + (L-1)//2] = Omega_c / np.pi * np.sinc(Omega_c / np.pi * n)  
    return fir
```


演習2

- 次の周波数特性 $H(\Omega) = \begin{cases} 0, & |\Omega| < \Omega_c \\ 1, & |\Omega| \geq \Omega_c \end{cases}$ を持つハイパスフィルタの係数を計算してみよう
- また, Notebook上の「**hpf**」関数の中身を埋めてハイパスフィルタを完成させてみよう

演習2解答例

```
def hpf(L, fc, fs):
    fir = np.zeros(L, dtype=float)
    T = 1/fs
    Omega_c = T * 2 * np.pi * fc
    for n in range(-(L-1)//2, (L-1)//2 + 1):
        fir[n + (L-1)//2] = np.sinc(n) - Omega_c / np.pi * np.sinc(Omega_c / np.pi * n)
    return fir
```

$H^{\text{HPF}}(\Omega) = 1 - H^{\text{LPF}}(\Omega)$ なので, 以下に等しい

```
def hpf(L, fc, fs):
    fir = np.zeros(L, dtype=float)
    for n in range(-(L-1)//2, (L-1)//2 + 1):
        fir[n + (L-1)//2] = np.sinc(n)
    fir_lpf = lpf(L, fc, fs)
    fir = fir - fir_lpf
    return fir
```

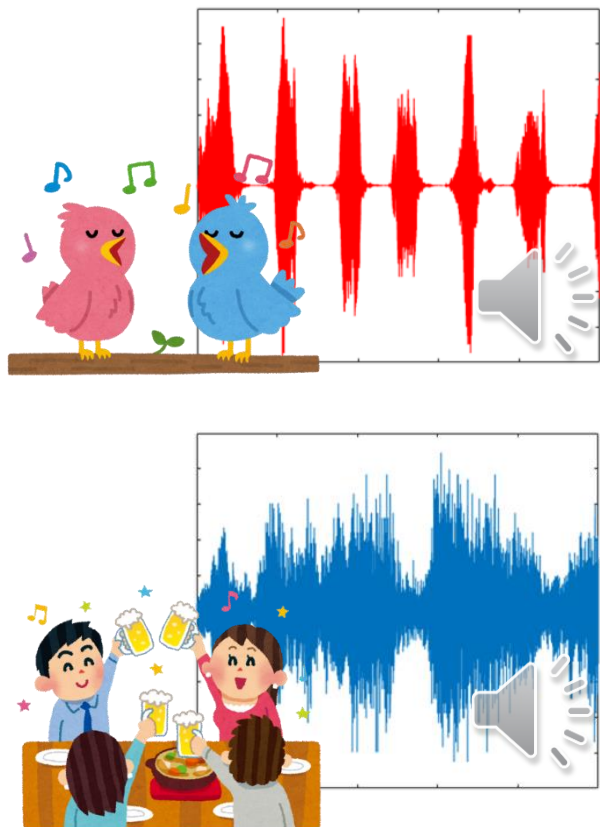
まとめ

1. 概要
2. フーリエ変換
 - 連続時間フーリエ変換
 - 離散時間フーリエ変換
3. LTIシステム
 - 線形性と時不変性
 - インパルス応答と周波数応答
 - 畳み込み
4. フィルタの設計
 - FIRフィルタとIIRフィルタ
 - フィルタ長と因果性
5. フィルタの評価

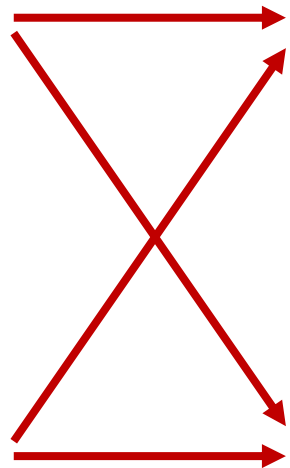
次回の予定

- 独立成分分析 (ICA) によるブラインド音源分離

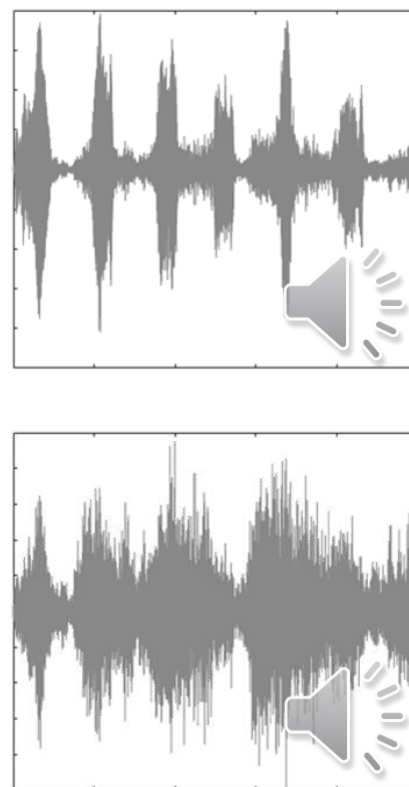
信号源



混合



観測信号



ICA



分離信号

