

# インタラクティブシステム論 第5回

梶本裕之



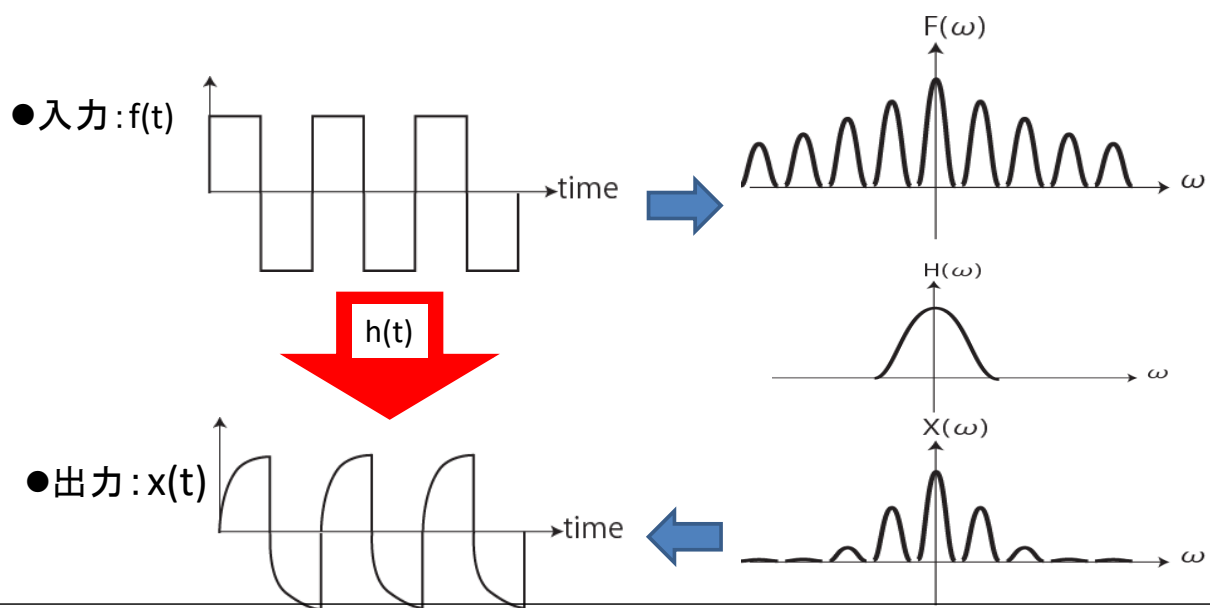
## 日程

講義番号	講義日	講義内容	pdf	video	レポート締め切り
1	4/9	イントロダクション	[ pdf] 2020年版	<a href="#">video</a>	4/16
		Scilab課題	[ pdf] 2020年版		↑
		上記資料のPython版	[ pdf] 2020年版		↑
2	4/16	フーリエ変換	[ pdf] 2020年版	<a href="#">video</a>	4/23
3	4/23	フーリエ変換と線形システム	[ pdf] 2020年版	<a href="#">video</a>	4/30
4	4/30	信号処理の基礎	[ pdf] 2020年版	<a href="#">video</a>	5/7
5	5/7	信号処理の応用1(相關)	[ pdf] 2020年版	<a href="#">video</a>	5/14
6	5/14	信号処理の応用2(画像処理)	[ pdf] 2020年版	<a href="#">video</a>	5/21
-	5/21	中間確認テスト準備 (自習)	[ pdf]2020年版		
-	5/28	中間確認テスト (現在は大学を予定)	[ pdf]2020年版		
7	6/4	ラプラス変換	[ pdf] 2020年版	<a href="#">video</a>	6/11
8	6/11	古典制御の基礎	[ pdf] 2020年版	<a href="#">video</a>	6/18
9	6/18	行列	[ pdf] 2020年版	<a href="#">video</a>	6/25
10	6/25	行列と最小二乗法	[ pdf] 2020年版	<a href="#">video</a>	7/2
11	7/2	ロボティクス	[ pdf] 2020年版	<a href="#">video</a>	7/9
-	7/9	期末テスト準備 (自習)	[ pdf]2020年版		
-	7/16	期末確認テスト (現在は大学を予定)			

日程およびテストを  
大学で行うかについ  
ては、**随時Google  
Classroom**や授業の  
ページを見てください。  
い。



# (復習) 周波数領域ではなく、 時間領域のまま議論できないか？



$X(\omega) = H(\omega) \times F(\omega)$ : 周波数領域で美しいのは分った。  
時間的な現象として何が起きているのか分からない

## (復習) 式で考えよう

$$X(\omega) = H(\omega) \times F(\omega)$$

フーリエ変換

$$F(\omega) = \int_{-\infty}^{\infty} f(t) \exp(-j\omega t) dt$$

逆フーリエ変換

$$f(t) = \int_{-\infty}^{\infty} F(\omega) \exp(j\omega t) d\omega$$

両辺を逆フーリエ変換すれば時間領域の信号に戻る。

$$\begin{aligned} x(t) &= \int_{-\infty}^{\infty} H(\omega) F(\omega) \exp(j\omega t) d\omega \\ &= \int_{-\infty}^{\infty} H(\omega) \left( \int_{-\infty}^{\infty} f(\tau) \exp(-j\omega \tau) d\tau \right) \exp(j\omega t) d\omega \\ &= \int_{-\infty}^{\infty} f(\tau) \left( \int_{-\infty}^{\infty} H(\omega) \exp(j\omega(t - \tau)) d\omega \right) d\tau \\ &= \int_{-\infty}^{\infty} f(\tau) h(t - \tau) d\tau \end{aligned}$$



# 逆順の計算もしておく（ふつうはこちら）

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

フーリエ変換

$$F(\omega) = \int_{-\infty}^{\infty} f(t)\exp(-j\omega t)dt$$

逆フーリエ変換

$$f(t) = \int_{-\infty}^{\infty} F(\omega)\exp(j\omega t)d\omega$$

両辺をフーリエ変換.

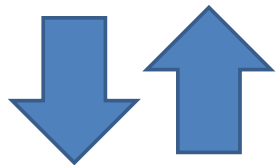
$$\begin{aligned} X(\omega) &= \int_{-\infty}^{\infty} \exp(-j\omega t)dt \int_{-\infty}^{\infty} f(\tau)h(t-\tau)d\tau \\ &= \int_{-\infty}^{\infty} \exp(-j\omega(\tau + (t-\tau)))dt \int_{-\infty}^{\infty} f(\tau)h(t-\tau)d\tau \\ &= \int_{-\infty}^{\infty} \exp(-j\omega\tau) \cdot \exp(-j\omega(t-\tau))dt \int_{-\infty}^{\infty} f(\tau)h(t-\tau)d\tau \\ &= \int_{-\infty}^{\infty} f(\tau)\exp(-j\omega\tau)d\tau \int_{-\infty}^{\infty} h(t-\tau)\exp(-j\omega(t-\tau))dt \\ &= \int_{-\infty}^{\infty} f(\tau)\exp(-j\omega\tau)d\tau \int_{-\infty}^{\infty} h(t')\exp(-j\omega t')dt' \\ &= F(\omega)H(\omega) \end{aligned}$$



## (復習) コンボリューション定理

$$X(\omega) = F(\omega)H(\omega) = H(\omega)F(\omega)$$

フーリエ逆変換



フーリエ変換

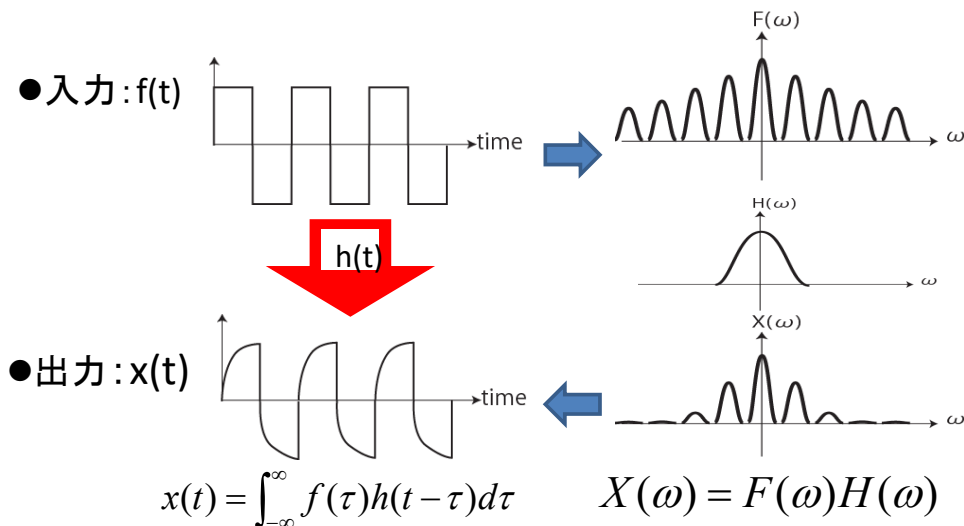
$$x(t) = \int_{-\infty}^{\infty} f(\tau)h(t-\tau)d\tau = \int_{-\infty}^{\infty} h(\tau)f(t-\tau)d\tau$$

簡略化のため次のようにも表記される

$$x(t) = f(t) * h(t) = h(t) * f(t)$$



## (復習) コンボリューション定理の意味 (1)

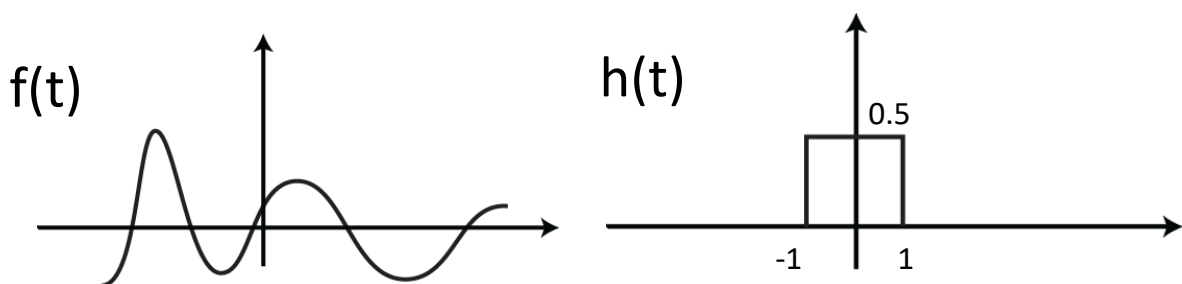


- $h(t)$ のフーリエ変換が $H(\omega)$ であるとする。
- 周波数領域でフィルタ $H(\omega)$ をかけることは、時間領域では、入力信号 $x(t)$ に対する関数 $h(t)$ の畳み込み積分(コンボリューション)として表現される。



## (復習) コンボリューション定理の意味 (2)

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



例えば,  $h(t)=0.5$  ( $-1 < t < 1$ )なら,

$$x(t) = \int_{-1}^1 0.5f(t-\tau)d\tau$$

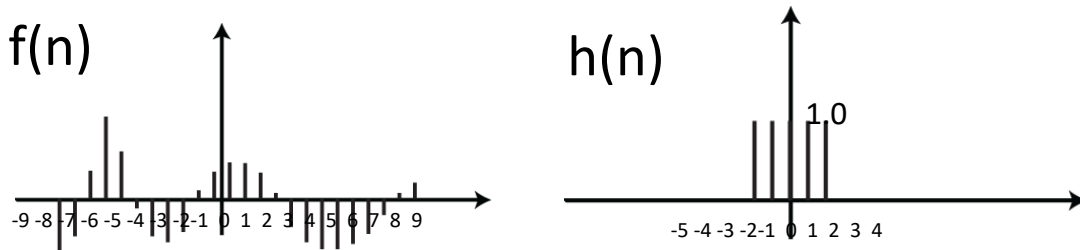
これは,  $f(t)$ を平均化していくフィルタ



# (復習) 離散化による理解

$$x(t) = \int_{-\infty}^{\infty} h(\tau) f(t - \tau) d\tau \quad \longrightarrow \quad x(n) = \sum_{i=-\infty}^{\infty} h(i) f(n - i)$$

$$x(n) = \dots + h(-4)f(n+4) + h(-3)f(n+3) + \dots + h(3)f(n-3) + h(4)f(n-4) + \dots$$



$h(n)$ が、 $n=-2 \sim 2$ の間だけ1の場合、

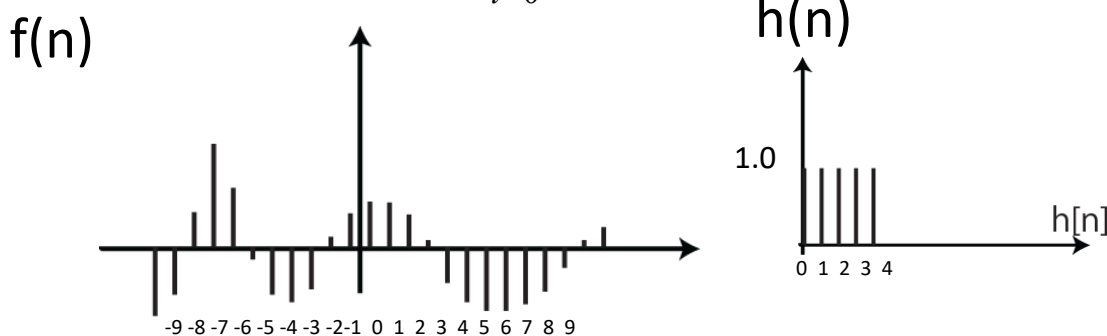
$$x(n) = f(n+2) + f(n+1) + f(n) + f(n-1) + f(n-2)$$

- この場合、出力 $x$ は、入力 $f$ の「平均化」になっている。
- つまりこの場合、 $h$ は平滑化フィルタである。



# (復習) FIRフィルタ

$$x(n) = \sum_{i=0}^{\infty} h(i) f(n - i)$$



$i=0$ から始める: 未来のデータが使えないことを意味する。  
この例は、元データ $f(n)$ を、4個平均して出力する。

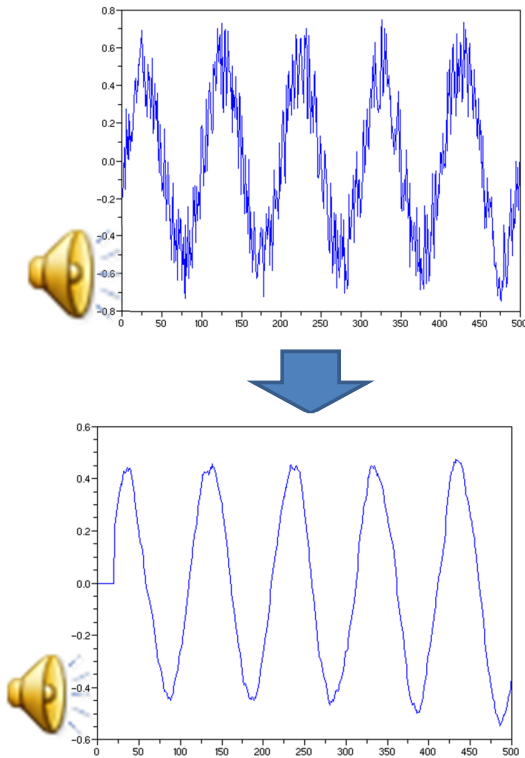
- 未来のデータが使えない例: リアルタイム制御
- 先のデータが使える例: 画像処理



# (復習) 平滑化フィルタの実例

メモリを20個持ったFIRフィルタによって平滑化

Scilabコード例



```
time = [0:0.01:100];
//振幅0.5の正弦波に最大振幅0.5のノイズが混入した信号
wave=0.5*sin(time*2*%pi) + 0.5*(rand(time)-0.5);

out=zeros(wave);

//20個を平均する.
for n=20:length(wave),
    for i=0:19,
        out(n)=out(n)+wave(n-i)/20;
    end
end

playsnd(out);
savewave('wave.wav',out);
plot(out(1:500));
```

# (復習) エコー

エコー＝時間遅れ信号の重畳.  
これはFIRフィルタで実装できる.

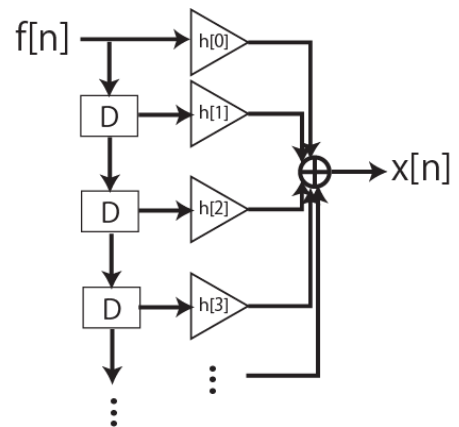
Scilabコード例

```
wave = loadwave('aiueo.wav');

out=zeros(wave);

//エコー(1000ステップ前の信号を重畳)
for n=1000:length(wave),
    out(n)=wave(n)+0.9*wave(n-999);
end

playsnd(out,11000); //11kHzサンプリングで再生
savewave('wave.wav',out,[11000]);
```



原音



1000ステップ前の信号を重畳



1000ステップ前+2000ステップ前の信号を重畳



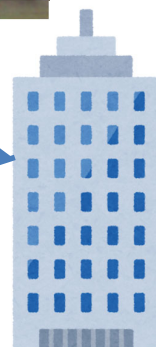
沢山重畳



# エコーは害（ゴースト障害／ゴースト現象）



<https://ja.wikipedia.org/wiki/%E3%83%B4%E3%83%99%E3%83%89%E3%83%99%E3%83%86%E3%83%9A>



相互相関関数  
自己相関関数



# エコーキャンセル (ゴーストリダクション)

FIRフィルタによってエコーの影響を低減することができる。

考え方:

(1) エコー成分のモデルを推定

$$\text{out}(n) = \text{wave}(n) + 0.5 * \text{wave}(n-100);$$

「100ステップ前の信号が0.5のゲインで重畳されている!」

(2) そのモデルに基づき、エコー成分と逆のゲインをかける

$$\text{out}(n) - 0.5 * \text{out}(n-100)$$

$$= \text{wave}(n) + 0.5 * \text{wave}(n-100) - 0.5 * (\text{wave}(n-100) + 0.5 * \text{wave}(n-200))$$

$$= \text{wave}(n) + 0.25 * \text{wave}(n-200)$$

⇒エコーが半分に低減!

(3) 当然もっと工夫すれば... (もっとメモリがあれば)

$$\text{out}(n) - 0.5 * \text{out}(n-100) - 0.25 * \text{out}(n-200)$$

$$= \dots = \text{wave}(n) + 0.125 * \text{wave}(n-300)$$

⇒無限にメモリがあれば完璧に消せる。



## エコーキャンセルの課題

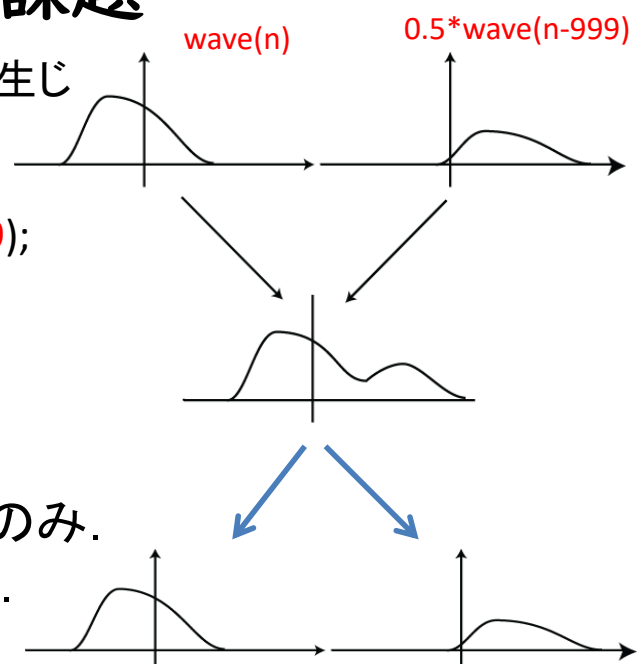
エコー成分が、どれだけ時間遅れを生じてやってくるかの**モデルを推定**

$$\text{out}(n) = \text{wave}(n) + 0.5 * \text{wave}(n-999);$$

<問題>

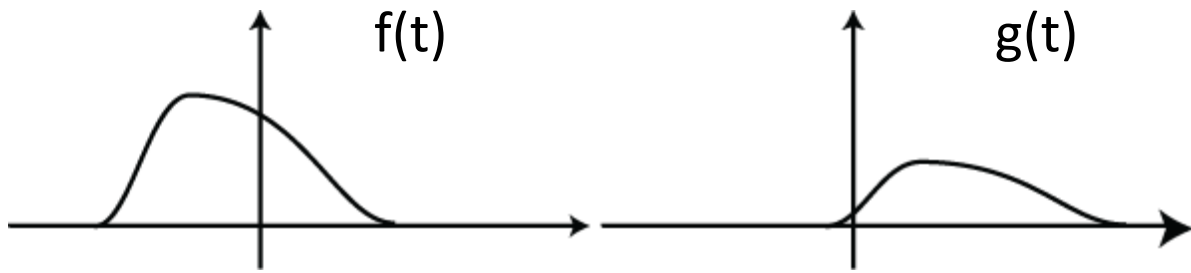
観測できるのは、  
エコーの「結果」としてのout(n)のみ。  
元の信号はわかっていない。

この信号からどのように、  
モデルを推定するのか?





# より簡単な問題から考えよう



- 二つの信号が、
- 時間的にどれだけずれているのか
  - 時間のずれを無視したらどれだけ似ているのか
- を測定したい。



## (復習)ベクトル空間と内積

ベクトル  $a=[a_x, a_y]$  のx成分は? . . . . .  $a_x$

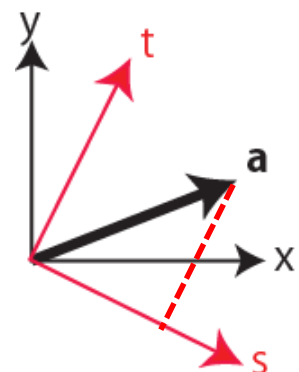
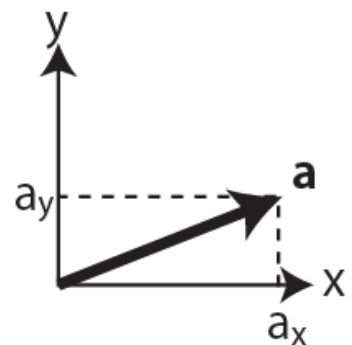
これはベクトル  $a$  とベクトル  $x=[1,0]$  との **内積** である。

$$a \cdot x = [a_x, a_y] \cdot [1, 0] = a_x$$

回転した座標軸,  $s, t$  を考える。  
ベクトル  $a=[a_x, a_y]$  の,  $s$  成分は?

これはベクトル  $a$  とベクトル  $s=[s_x, s_y]$  との **内積** である。

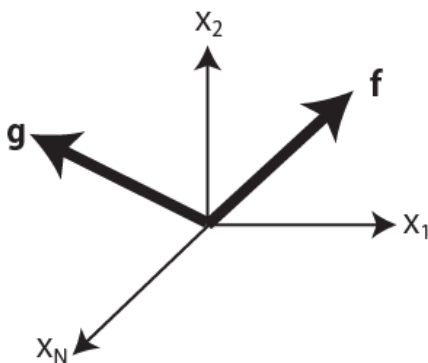
$$a \cdot s = [a_x, a_y] \cdot [s_x, s_y]$$



内積は、あるベクトルが別のベクトルの成分をどれだけ持つかを表す

# (復習) N次元空間では

N次元空間で、二つのベクトル  
 $f=[f_1, f_2, \dots, f_N], g=[g_1, g_2, \dots, g_N]$  を考える.



内積  $f \cdot g$  は、ベクトル  $f$  の、 $g$  軸成分(または逆)を表す.

$$\begin{aligned} &= [f_1 \quad f_2 \quad \dots \quad f_N] \bullet [g_1 \quad g_2 \quad \dots \quad g_N] \\ &= \sum_{i=1}^N f_i g_i \end{aligned}$$



# (復習) 波形 $f$ に波形 $g$ はどれだけ含まれるか

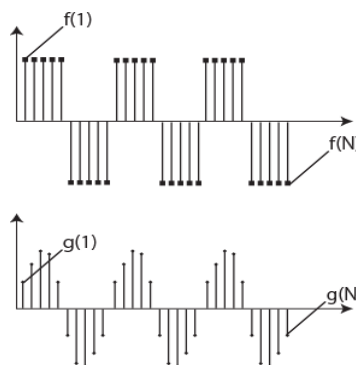
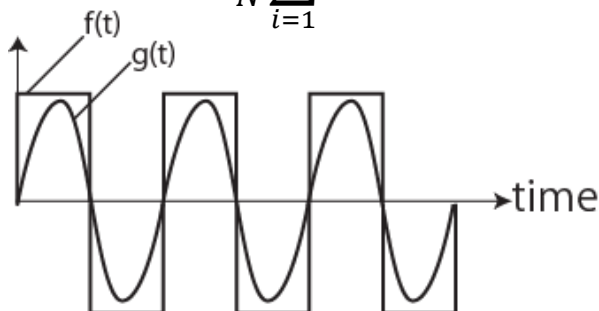
波形  $f$  中の、波形  $g$  の成分

$$= \frac{1}{T} \int_0^T \overline{f(t)} g(t) dt$$

(連続関数) 複素数の場合も考えて共役をとる

$$= \frac{1}{N} \sum_{i=1}^N \overline{f(i)} g(i)$$

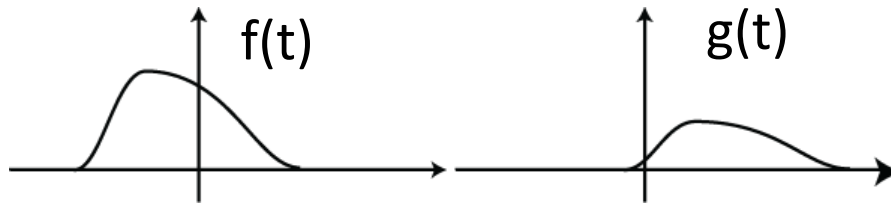
(離散化して考えた場合)



これは二つの波をベクトルと考えた時の内積に他ならない  
※内積を連続関数に対して定義



# 相互相関



<問題>

二つの信号が,

- 時間的にどれだけずれているのか
- 時間のずれを無視したらどれだけ似ているのかを測定したい。

内積を思い出せば,  
次の手順で測定すればよいことがわかる

- $g(t)$ を $\tau$ だけずらしてみる  $\Rightarrow$
- $f(t)$ との内積を取ってみる  $\Rightarrow$
- $\tau$ を変化させていく。

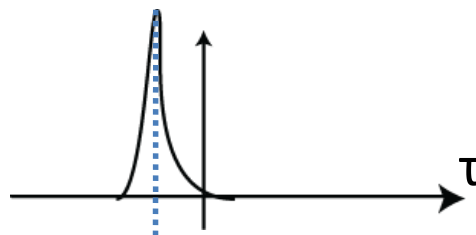
  


# 相互相関



$R_{fg}(\tau)$ : 二つの関数 $f(t)$ ,  $g(t)$ の, 相互相関関数

$$R_{fg}(\tau) = \int_{-\infty}^{\infty} \overline{f(t)} g(t + \tau) dt$$

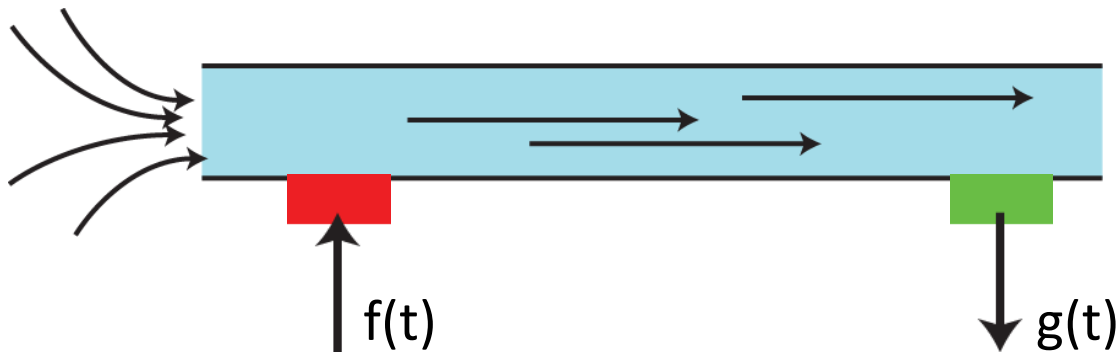


$R_{fg}(\tau)$ が最大の値をとる $\tau$  = 元の関数 $f(t)$ と $g(t)$ のズレ  
(ただし直流成分を取り除いた後)



# 相互相関の応用：速度計測

管内の流速を正確に測定したい  
ただし、管の中に接触してはいけない(液漏れ厳禁)

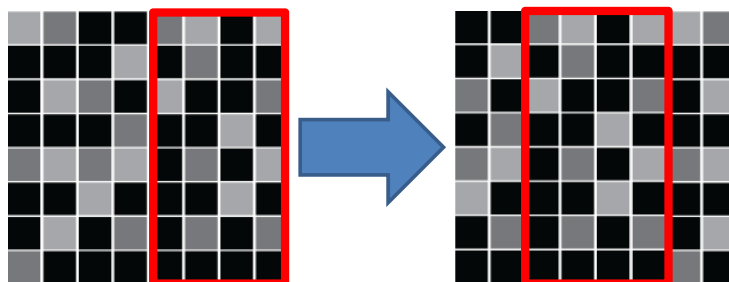


上流に熱源を置き,  $f(t)$  でランダムに変動させる.  
下流で温度を測定する.  $g(t)$

$f(t)$  と  $g(t)$  の相互相関関数が最大となる時間差  $\tau$  が,  
水流によって熱が移動するのに要する時間である.



# 相互相関の応用：速度計測



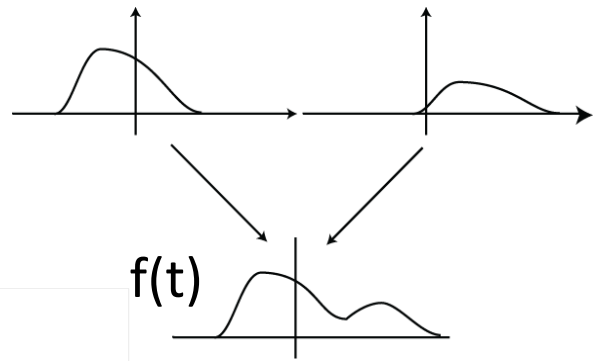
光学式マウスの中身 = 数十pixel角のCMOSカメラ。  
二つの画像 (= 2次元関数) 同士の相互相関を取ることで移動量を計測。



# 自己相関

二つの関数 $f(t)$ ,  $g(t)$ の代わりに、  
ひとつの関数 $f(t)$ の相関を取る。

$$R_{ff}(\tau) =$$

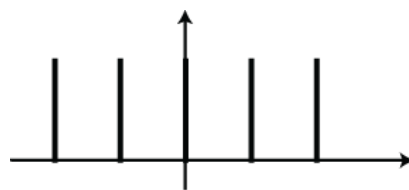


自己相関関数は、  
「どれだけずれたら自分自身に近い形になるか」  
を表す。  
すなわち、**エコー**を発見していることに他ならない。🔊

# 周期関数の自己相関



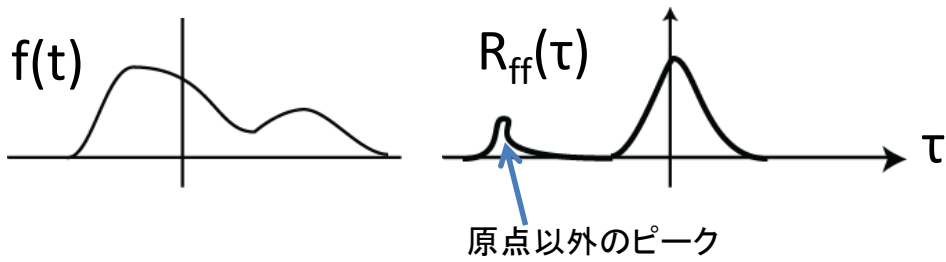
$$R_{ff}(\tau) = \int_{-\infty}^{\infty} \overline{f(t)} f(t + \tau) dt$$



周期関数の自己相関関数は沢山のピークが出る



# 自己相関まとめ

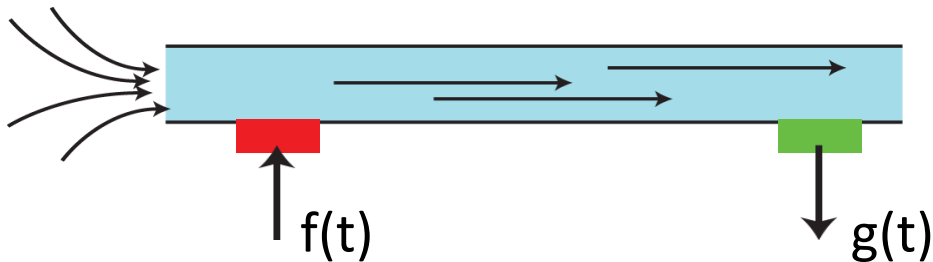


$R_{ff}(\tau)$ : 自分自身を $\tau$ ずらしたら、自分自身にどれだけ似ているか

- 当然,  $\tau=0$ で最大値を取る.
- $\tau \neq 0$ で大きな値をとった場合, その信号には $\tau$ の時間遅れ(エコー)成分が含まれている.
- 信号に潜む周期性を発見することもできる



## 白色雑音 (ホワイトノイズ) と自己相関

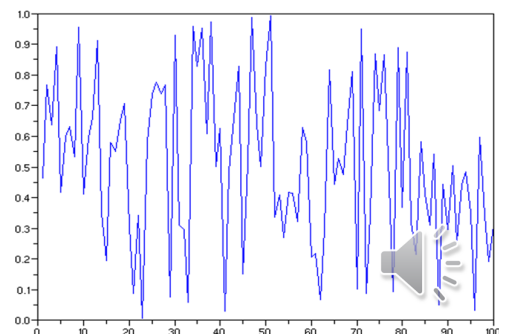


元の信号 $f(t)$ に「周期性」があったら,  $f(t)$ と $g(t)$ の相互相関は沢山の「ニセのピーク」が出てしまう.

元の入力 $f(t)$ はなるべく「でたらめ」であることが望ましい $\Rightarrow$ 白色雑音

<白色雑音の定義>

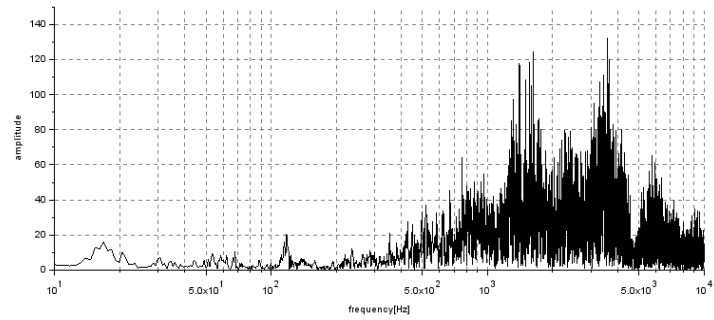
1. あらゆる周波数成分が均等に含まれる.
2. 自己相関関数が $\tau=0$ 以外では全て0  
(=エコー成分, 周期性を全く持たない)



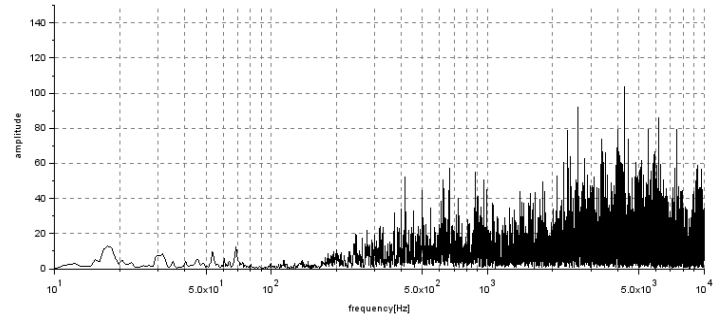
# インパルス応答とホワイトノイズ (屋内)



インパルス(拍手)



ホワイトノイズ



似ている？



## 自己相関とパワースペクトル

自己相関関数をフーリエ変換する  $R_{ff}(\tau) = \int_{-\infty}^{\infty} \overline{f(t)} f(t + \tau) dt$

$$\int_{-\infty}^{\infty} R_{ff}(\tau) e^{-j\omega\tau} d\tau = \int_{-\infty}^{\infty} \left[ \int_{-\infty}^{\infty} \overline{f(t)} f(t + \tau) dt \right] e^{-j\omega\tau} d\tau$$

=

=

=

=

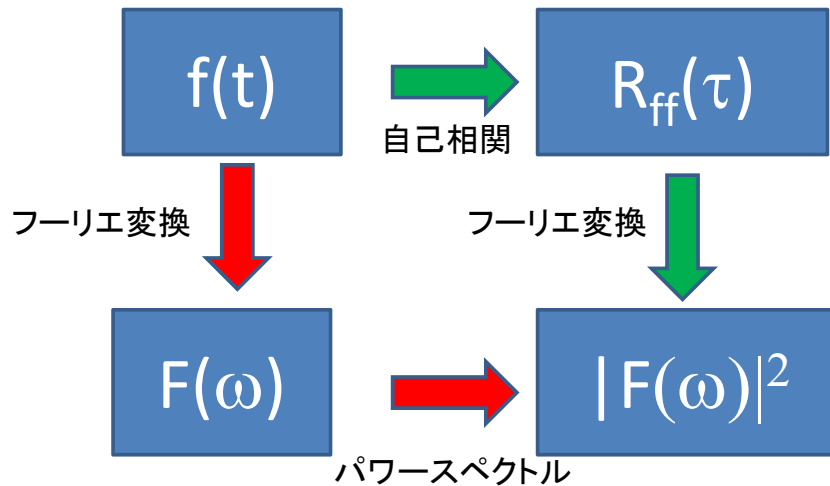
=



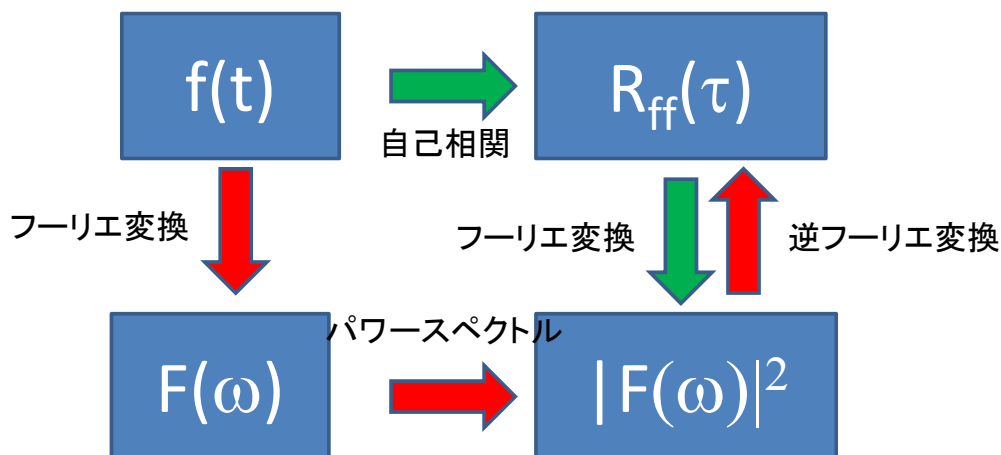
# 自己相関とパワースペクトル

$$\int_{-\infty}^{\infty} R_{ff}(\tau) e^{-j\omega\tau} d\tau = \|F(\omega)\|^2$$

## ウィナー・ヒンチンの定理



# 自己相関とパワースペクトル



自己相関は  
フーリエ変換⇒パワースペクトル⇒逆フーリエ変換  
によって間接的に計算できる

- 自己相関を直接計算した時の計算量:  $O(n^4)$
  - フーリエ変換(FFT)を使った時の計算量:  $O(n \log(n))$
- この方が計算が早いため多用される。





# レポート課題Ⅰ：自己相関

(1) 適当なwaveファイルに対してエコーを掛けてカラオケのようになる。(前回のレポート)

(2) その結果に対して定義通りの方法で自己相関関数を計算し、確かにエコー分の遅れのところでピークが出ることを確認する(エコーの検出)。

(3) ウイナー・ヒンチンの定理を用い、パワースペクトルの逆フーリエ変換によって同じ結果が出ることを確認する

(2)(3)のScilab (or python)ソースファイルの添付. waveファイルは添付不要。(2)(3)の処理にかかった時間についてコメント



# レポート課題：ヒント

(1) 適当なwaveファイルに対してエコーを掛けてカラオケのようになる。(前回のレポート)

```
[wave,f] = loadwave('aiueo.wav');  
//1000ステップごとのエコーを10回重ねる例  
out=[zeros(wave),zeros(1,10000)];  
for i=0:9,  
    out=out+[zeros(1,1000*i), wave, [zeros(1,10000-1000*i)]];  
end  
//plot(wave);  
savewave('foo.wav',out,f(3));
```




## レポート課題：ヒント

(2) その結果に対して定義通りの方法で自己相関関数を計算し、確かにエコー分の遅れのところでピークが出ることを確認する(エコーの検出).

```
time=length(out);  
auto_correlation=zeros(1,time);
```

```
for tau=1:time,  
    auto_correlation(tau) =   
end
```


```
plot(auto_correlation);
```

自己相関の定義式から、 $[\text{zeros}(1,\text{tau}), \text{out}]$  と  $[\text{out}, \text{zeros}(1,\text{tau})]$  の内積を取れば良い。(ベクトルの内積はどう書き表せるか?) 

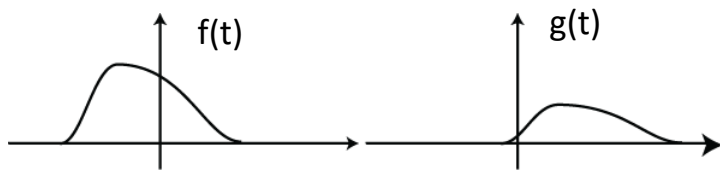
## レポート課題：ヒント

(3) ウィナー・ヒンチンの定理を用い、パワースペクトルの逆フーリエ変換によって同じ結果が出ることを確認する

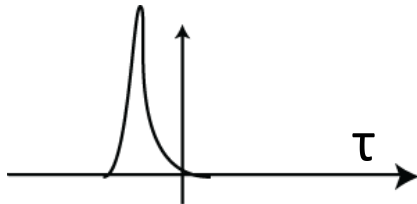
```
//フーリエ変換  
fourier =   
//過去のレポート課題を参照  
//パワースペクトル  
power_spec =   
//過去のレポート課題を参照  
//自己相関  
auto_correlation =   
//逆フーリエ変換  
  
plot(auto_correlation);
```

※(参考)(2)(3)の結果の微妙な違いは、(3)が信号の無限繰り返しを仮定しているために生じる。逆に(2)もそう仮定すれば(3)と全く同じ結果が得られる 

# (発展トピック) 相互相関と検波



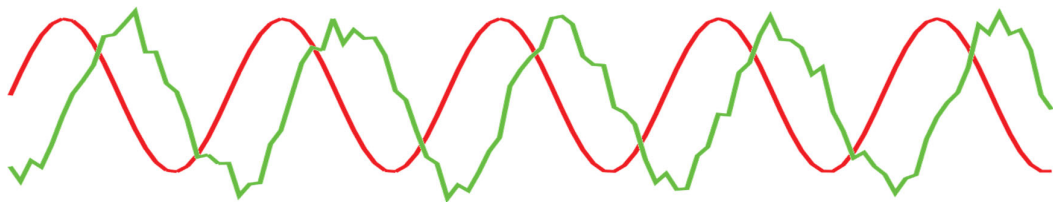
$$R_{fg}(\tau) = \int_{-\infty}^{\infty} \overline{f(t)} g(t + \tau) dt$$



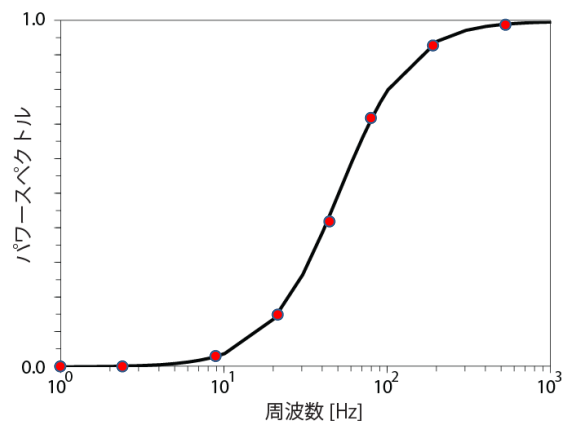
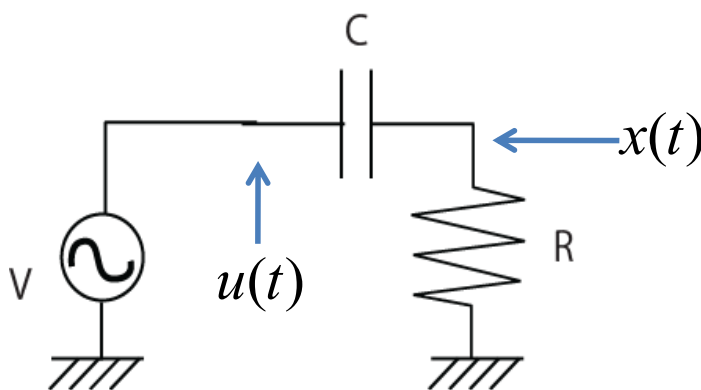
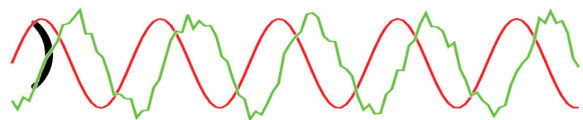
$R_{fg}(\tau)$ が最大の値をとる $\tau$   
 =元の関数 $f(t)$ と $g(t)$ のズレ  
 (ただし直流成分を取り除いた後)

実用上重要なケースの一つ:

- $f(t)$ が正弦波(入力)で,
- $g(t)$ がノイズが入って位相が遅れた正弦波(出力)



## 正弦波入出力の例 (I)



フィルタ回路: 入力 $u(t)$ , 出力 $x(t)$

電子回路, アンテナ, ロボットアーム, etc

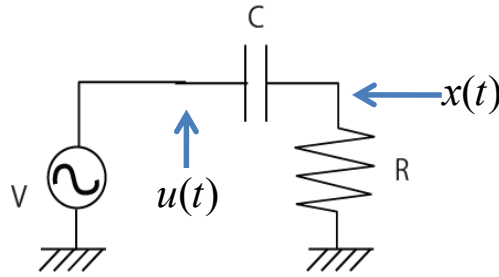
フィルタ特性を実測で求める場合

- $u(t)=\sin(\omega t)$ を**入力**し,
- 出力**を観察する.

$\omega$ を変化させることでフィルタ特性を得る.

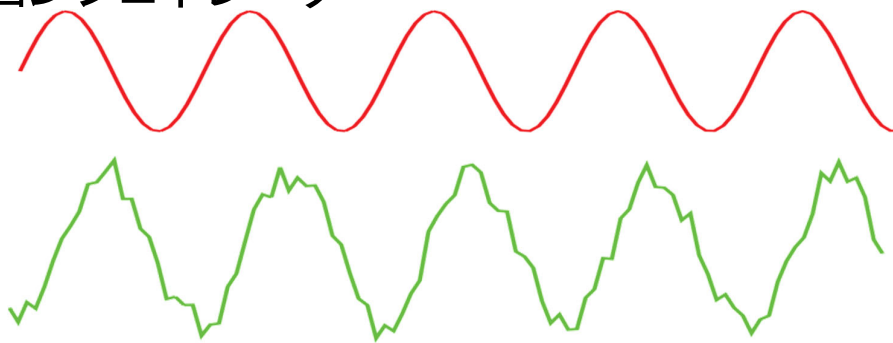


# 正弦波入出力の例（1）



ファンクションジェネレータ

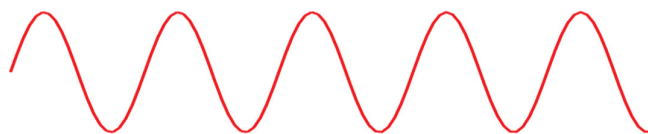
オシロスコープ



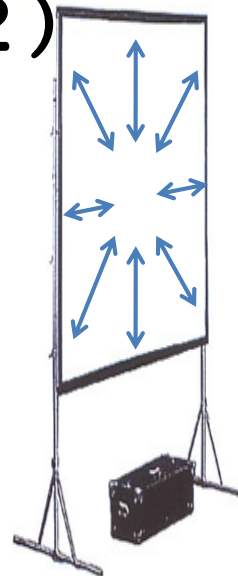
# 正弦波入出力の例（2）



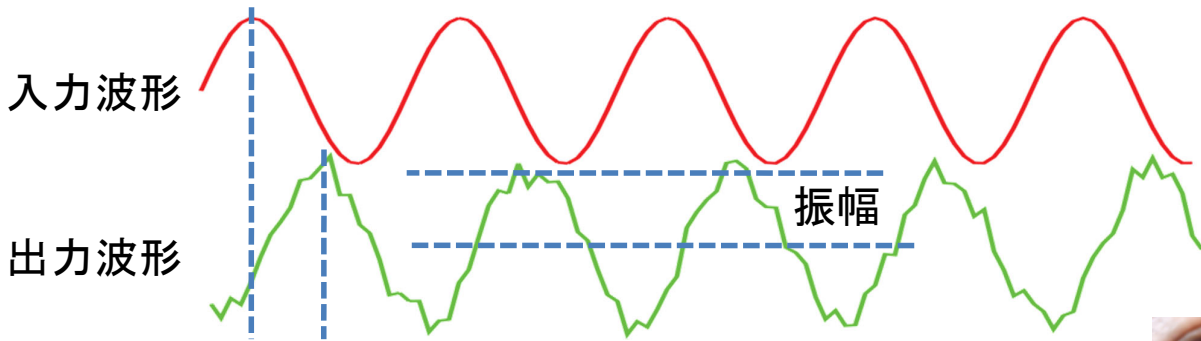
オプティカルフローの速度



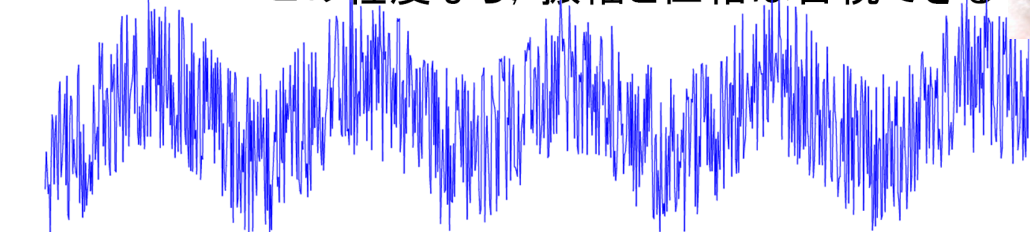
重心動揺



# 入力と出力の位相差，振幅比を求める



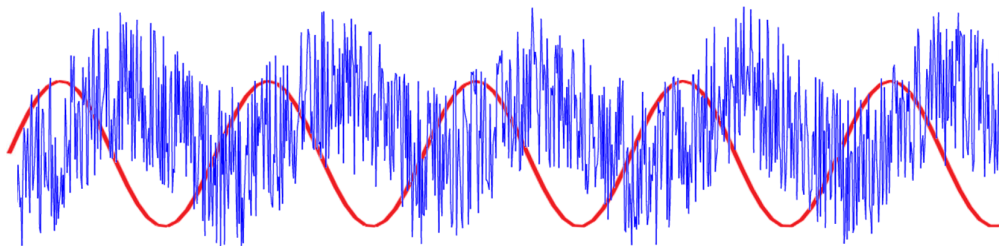
位相遅れ この程度なら，振幅と位相は目視できる



でもたいていこういう残念な感じ. ノイズがひどい  
⇒振幅と位相を，どうやって求めたら良い？  
方法1: バンドパスフィルタを使う  
方法2: 相関関数を考える



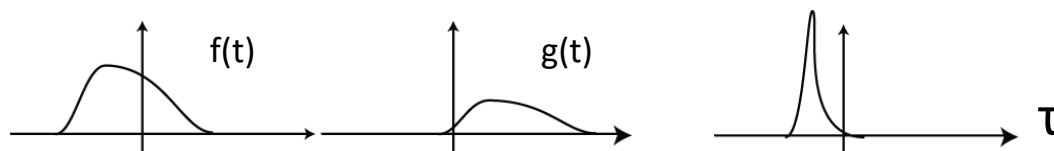
# 正弦波による相互相関



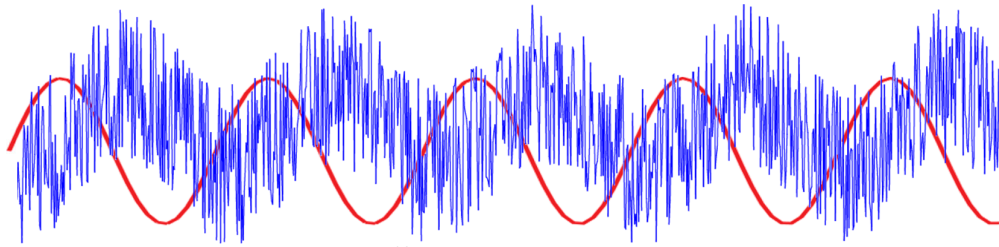
- 出力信号 $f(t)$ : ノイズだらけの正弦波. 位相不明. 周波数は既知.
- 入力信号 $g(t)$ : 周波数固定の正弦波

$g(t)$ を動かしていき，ピッタリと重なるところを見つければ良い  
⇒相互相関の計算にほかならない.

$$R_{fg}(\tau) = \int_{-\infty}^{\infty} \overline{f(t)}g(t + \tau)dt$$



# 正弦波による相互相関



$$R_{fg}(\tau) = \int_{-\infty}^{\infty} \overline{f(t)} g(t + \tau) dt$$

正弦波g(t)を動かす

$$R_{fg}(\tau) =$$

=

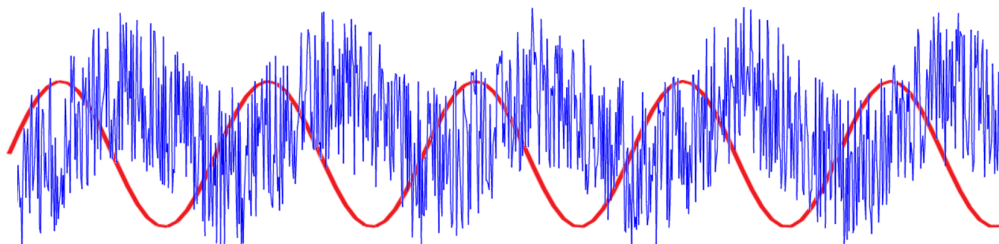
=

=

つまり正弦波の場合、積分の計算は2回やるだけでよい！！



# 直交検波



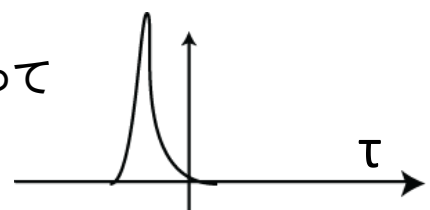
$$R_{fg}(\tau) = \cos(\omega\tau) \frac{1}{T} \int_0^T f(t) \sin(\omega t) dt + \sin(\omega\tau) \frac{1}{T} \int_0^T f(t) \cos(\omega t) dt$$

積分結果をS,Cと書いて

$$R_{fg}(\tau) = S \cos(\omega\tau) + C \sin(\omega\tau)$$

これが最大の値をとる場所τは、微分をとって

$$\frac{dR_{fg}(\tau)}{d\tau} = \quad = 0$$

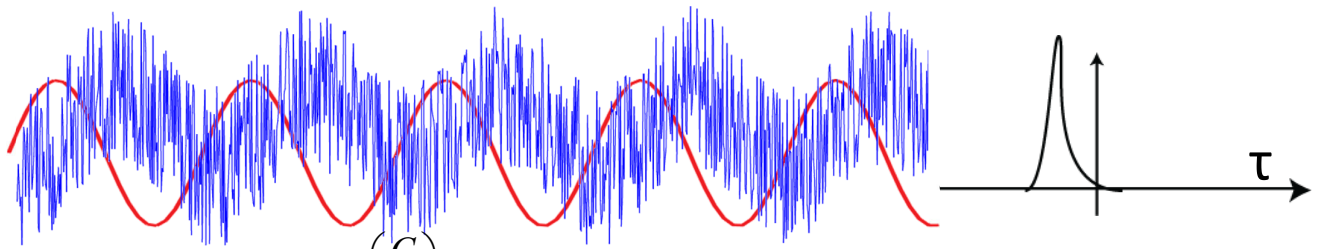


∴

⇒位相遅れτが求まった



# 直交検波



$$\therefore \arctan\left(\frac{C}{S}\right) = \omega\tau \Rightarrow \text{位相遅れ}\tau\text{が求まった}$$

このときの相関値は,

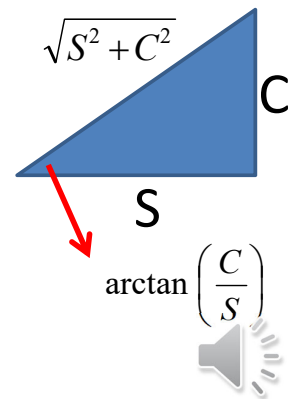
$$R_{fg}(\tau) = S \cos(\omega\tau) + C \sin(\omega\tau)$$

$$= S \cos(\quad) + C \sin(\quad)$$

$$= S \quad + C \quad$$

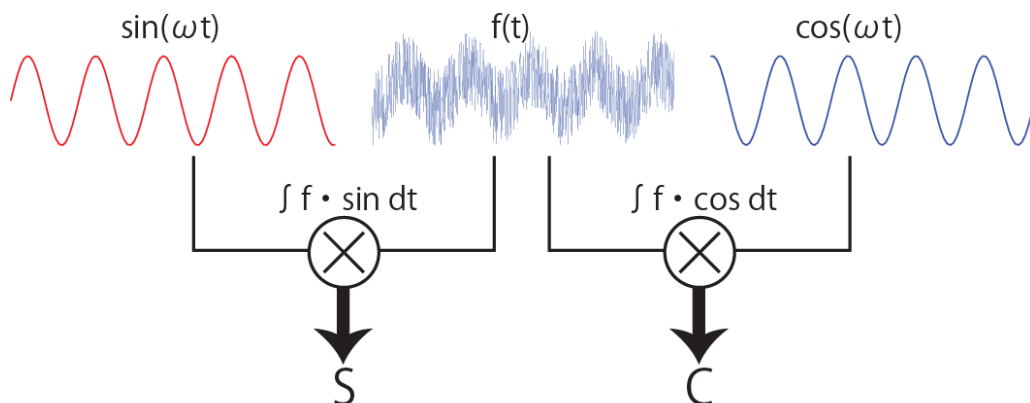
$$= \quad$$

$$= \quad \Rightarrow \text{振幅に相当する量}$$



# 直交検波まとめ

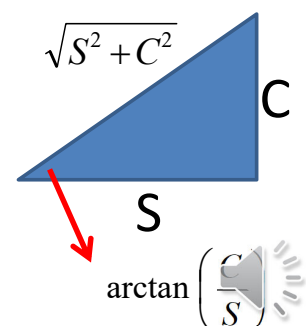
周波数 $\omega$ がわかっているノイズだらけの信号 $f(t)$ があったとき、振幅と位相は次のように計算できる。



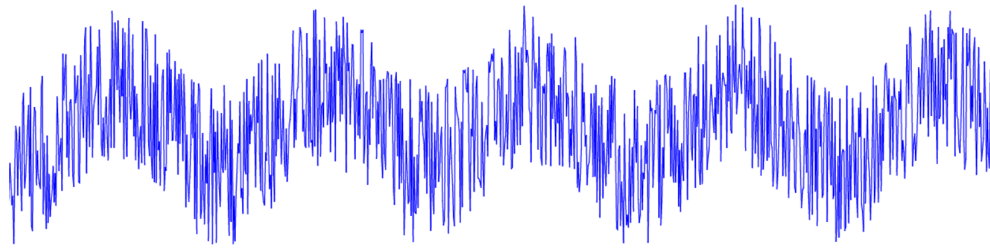
(1)  $\sin(\omega t)$ と $\cos(\omega t)$ との内積をとり、 $S$ 、 $C$ とおく。

(2) 位相遅れは $\arctan(C/S)$ となる。

(3) 振幅は $\sqrt{S^2+C^2}$ に比例する。



# 直交検波：もっと数式で理解する



問題を定式化

信号 $f(t)$ が,

$f(t) =$

と仮定できるとする. 周波数 $\omega$ はわかっている.

計測データから, 振幅 $A$ と, 位相ずれ $\phi$ を求めるには? 

# 直交検波：さらに数式で理解する

信号に $\sin(\omega t)$ をかけ, 積分する(=内積をとる)  
積分時間 $T$ は充分長い.

$$S = \frac{1}{T} \int_0^T (A \sin(\omega t + \phi) + \text{noise}(t)) \sin(\omega t) dt$$

$$= \frac{1}{T} \cdot$$

$$= \frac{1}{T} \cdot$$

$$=$$

$$=$$

$$=$$






# 直交検波：さらに数式で理解する

信号に $\cos(\omega t)$ をかけ、積分する(=内積をとる)  
積分時間 $T$ は充分長い。

$$\begin{aligned} C &= \frac{1}{T} \int_0^T (A \sin(\omega t + \phi) + \text{noise}(t)) \cos(\omega t) dt \\ &= \frac{1}{T} \int_0^T A \sin(\omega t + \phi) \cos(\omega t) dt + \int_0^T \text{noise}(t) \cos(\omega t) dt \\ &= \frac{1}{T} \int_0^T A (\sin(\omega t) \cos(\phi) + \cos(\omega t) \sin(\phi)) \cos(\omega t) dt \\ &= A \cos(\phi) \frac{1}{T} \int_0^T \sin(\omega t) \cos(\omega t) dt + A \sin(\phi) \frac{1}{T} \int_0^T \cos^2(\omega t) dt \\ &= A \sin(\phi) \frac{1}{T} \int_0^T \frac{1 + \cos(2\omega t)}{2} dt \end{aligned}$$

=  

# 直交検波：さらに数式で理解する

$$S = \frac{A}{2} \cos(\phi) \quad C = \frac{A}{2} \sin(\phi)$$

この二つの結果から、

位相差

$$\frac{C}{S} = \text{$$
$$\phi = \text{$$

振幅

$$S^2 + C^2 = \frac{A^2}{4} (\cos^2(\phi) + \sin^2(\phi))$$

$$= \text{$$

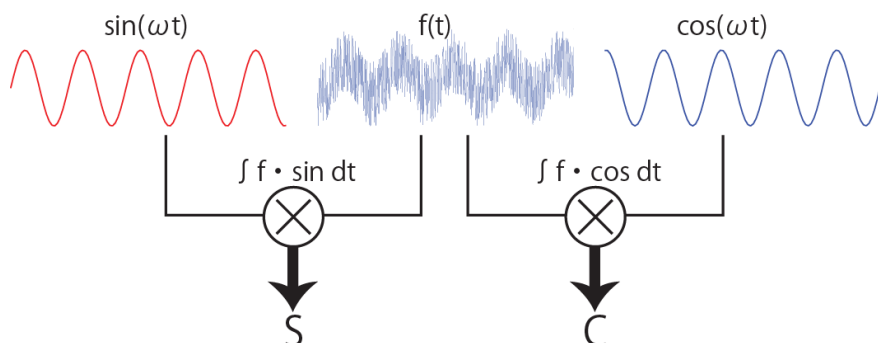
$$A = \text{$$

位相差と振幅が求まった



# (参考) AMラジオの同期検波

ラジオの信号は、典型的な  
「周波数 $\omega$ がわかっているノイズだらけの信号 $f(t)$ 」



復調方式の一つ:「同期検波」

直交検波をリアルタイムに行い、**ノイズに隠れた真の信号**を得る。

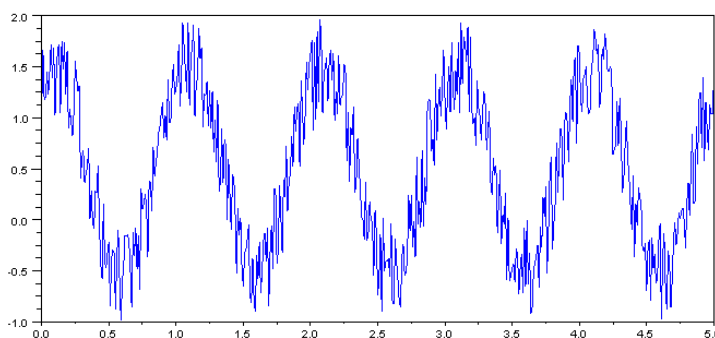
今ではPC上で計算可能(ソフトウェア無線技術)



## レポート課題2： ノイズ入り正弦波の位相を調べる

次のScilabソースで表されるノイズ入り正弦波の位相遅れを求め、妥当性をコメントせよ。

```
//ノイズ入り正弦波
t=[0:0.01:5]; //時刻
f=1.0; //周波数
amp=0.5; //振幅
phi=0.3*%pi; //位相遅れ
y=sin(2*%pi*f*t+phi) + rand(t);
plot(t,y);
```



```
S =  //yとsinの内積
C =  //yとcosの内積
ans_phi=atan(C,S) //検出された位相遅れ
```

