デジタルメディア処理1

担当: 井尻 敬

（講義当日までもう少し修正を入れる予定です）

スケジュール

09/26　イントロダクション1：デジタル画像とは，量子化と標本化，Dynamic Range
10/03　イントロダクション2：デジタルカメラ，人間の視覚，表色系
10/10　フィルタ処理1：トーンカーブ，線形フィルタ
10/17　フィルタ処理2：非線形フィルタ，ハーフトーニング
10/24　フィルタ処理3：離散フーリエ変換と周波数フィルタリング
11/07　前半のまとめと中間試験
11/14　画像処理演習：python入門　(演習室)
11/21　画像処理演習：フィルタ処理 (演習室)
11/28　画像処理演習：フィルタ処理 (演習室)
12/05　画像処理演習：フィルタ処理 (演習室)
12/12　画像の幾何変換１：アファイン変換
12/19　画像の幾何変換２：画像の補間
01/16　画像復元：ConvolutionとDeconvolution
01/23　後半のまとめと期末試験

Deconvolution

σ =20のガウシアンブラーの例


手振れ・焦点ずれによりボケてしまった画像


ボケのない画像を復元

ボケを引き起こした**点広がり関数**が既知ならば綺麗な復元が可能

Deconvolution

w=20, θ=0.2π の線分状の点広がり関数の例


手振れ・焦点ずれによりボケてしまった画像


ボケのない画像を復元

ボケを引き起こした**点広がり関数**が既知ならば綺麗な復元が可能

復習：線形フィルタ（Convolution）

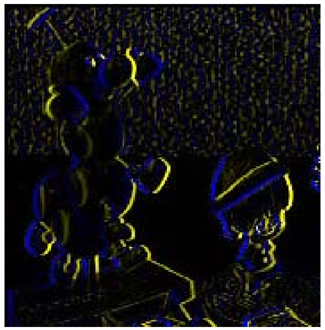
線形フィルタの例

ぼかす

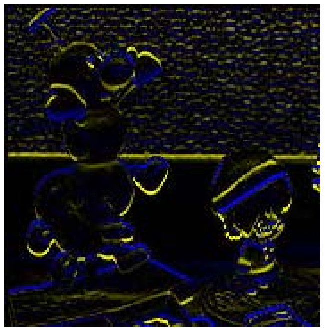
先鋭化

線形フィルタの例

エッジ抽出

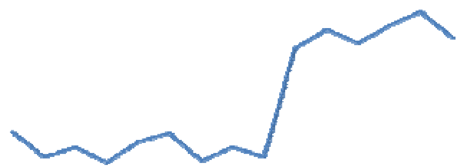


横方向



縦方向

線形フィルタの例 1D



33	15	22	11	26	32	12	22	15	94	108	98	111	121	102	102

27	23	16	20	23	23	22	16	44	72	100	106	110	111	108

※端ははみ出すので値をコピー（ほかの方法もある）

平滑化したい!

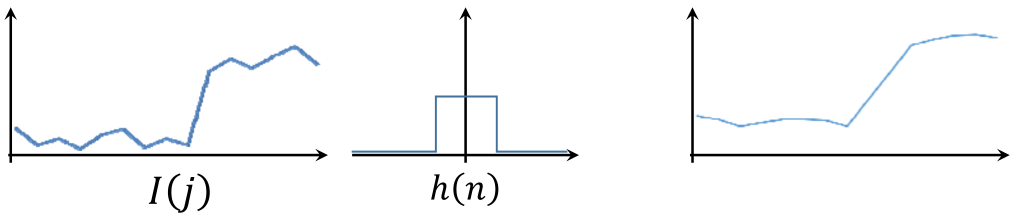
1/3	1/3	1/3

周囲３ピクセル
の平均を取る

線形フィルタ(1D)とは

出力画素値を周囲画素の重み付和で計算するフィルタ

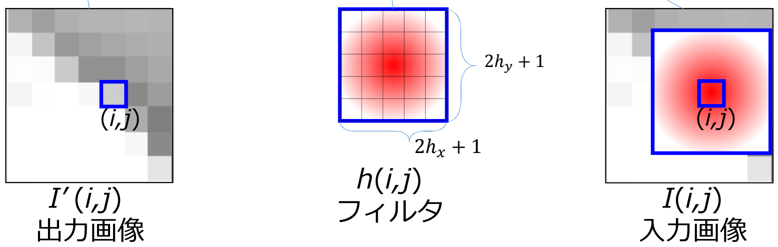
$$I'(j) = \sum_{n=-h_x}^{h_x} h(n)\ I(j+n)$$



線形フィルタ(2D)とは

出力画素値を周囲画素の重み付和で計算するフィルタ

$$I'(i,j) = \sum_{m=-h_y}^{h_y} \sum_{n=-h_x}^{h_x} h(m,n)\ I(i+m, j+n)$$



Convolution(畳み込み)とは

二つの関数 $f(t)$ $g(t)$ を重ね合わせる演算
$f(t)$ を固定し，$g(t)$ を平行移動しながら$f(t)$に掛けあわせ，得られた関数を積分

連続関数

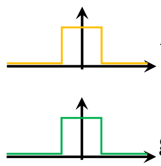
$$(f*g)(x) = \int_{-\infty}^{\infty} f(t)g(x-t)dt$$

離散関数

$$(f*g)(n) = \sum_{k=-\infty}^{\infty} f(k)g(n-k)$$

例題)
2関数の畳み込み積分を求めよ

$$(f*g)(x) = \int_{-\infty}^{\infty} f(t)g(x-t)dt$$

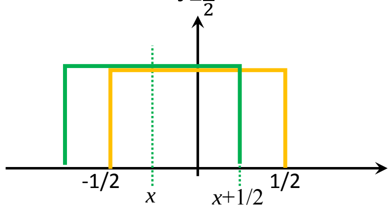


$$f(x) = \begin{cases} 1 & -\frac{1}{2} \le x \le \frac{1}{2} \\ 0 & otherwise \end{cases}$$

$$g(x) = \begin{cases} 1 & -\frac{1}{2} \le x \le \frac{1}{2} \\ 0 & otherwise \end{cases}$$

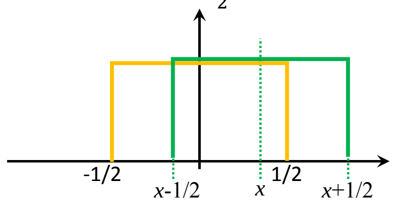
$-1 \le x \le 0$のとき

$$(f*g)(x) = \int_{-\frac{1}{2}}^{x+\frac{1}{2}} 1 \times 1dt = x+1$$



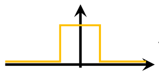
$0 \le x \le 1$のとき

$$(f*g)(x) = \int_{x-\frac{1}{2}}^{\frac{1}{2}} 1 \times 1dt = 1-x$$

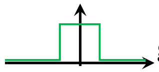


例題)
2関数の畳み込み積分を求めよ

$$(f*g)(x)=\int_{-\infty}^{\infty}f(t)g(x-t)dt$$

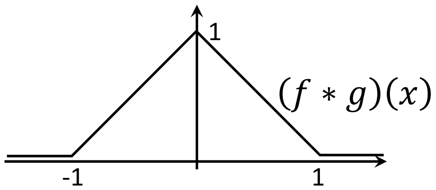


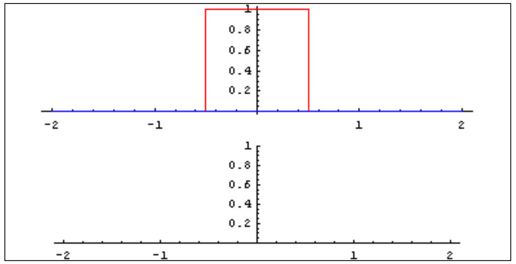
$$f(x)=\begin{cases}1 & -\frac{1}{2}\le x\le\frac{1}{2}\\ 0 & otherwise\end{cases}$$



$$g(x)=\begin{cases}1 & -\frac{1}{2}\le x\le\frac{1}{2}\\ 0 & otherwise\end{cases}$$

$$(f*g)(x)=\begin{cases}x+1 & -1\le x\le 0\\ 1-x & 0\le x\le 1\\ 0 & otherwise\end{cases}$$

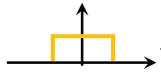




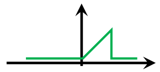
By Lautaro Carmona [CC-BY-SA] from wikipedia

例題)
2関数の畳み込み積分を求めよ

$$(f*g)(x)=\int_{-\infty}^{\infty}f(t)g(x-t)dt$$



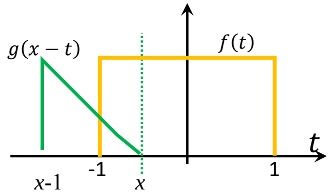
$$f(x)=\begin{cases}1 & -1\le x\le 1\\ 0 & otherwise\end{cases}$$



$$g(x)=\begin{cases}x & 0\le x\le 1\\ 0 & otherwise\end{cases}$$

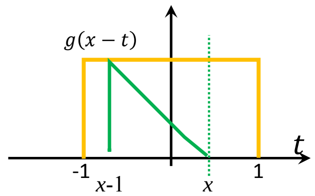
$-1\le x\le 0$のとき

$$(f*g)(x)=\int_{-1}^{x}(x-t)dt=\frac{(x+1)^2}{2}$$



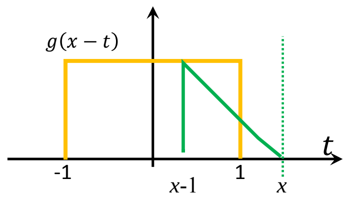
$0\le x\le 1$のとき

$$(f*g)(x)=\int_{x-1}^{x}(x-t)dt=\frac{1}{2}$$



$1\le x\le 2$のとき

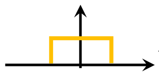
$$(f*g)(x)=\int_{x-1}^{1}(x-t)dt=-\frac{x^2}{2}+x$$



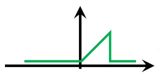
$g(x-t)$は，t軸に対して反転するので注意

例題)
2関数の畳み込み積分を求めよ

$$(f*g)(x)=\int_{-\infty}^{\infty}f(t)g(x-t)dt$$

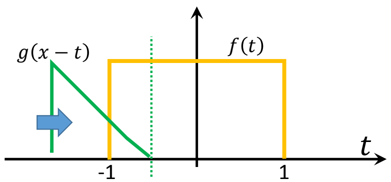


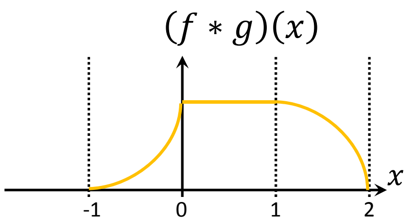
$$f(x)=\begin{cases}1 & -1\le x\le 1\\ 0 & otherwise\end{cases}$$



$$g(x)=\begin{cases}x & 0\le x\le 1\\ 0 & otherwise\end{cases}$$

$$(f*g)(x)=\int_{-1}^{x}(x-t)dt=\begin{cases}(x+1)^2/2 & -1\le x\le 0\\ 1/2 & 0\le x\le 1\\ -x^2/2+x & 1\le x\le 2\end{cases}$$





Convolution(畳み込み)とは

二つの関数 $f(t)$ $g(t)$ を重ね合わせる演算
$f(t)$ を固定し，$g(t)$ を平行移動しながら$f(t)$に掛けあわせ，得られた関数を積分

連続関数

$$(f*g)(x)=\int_{-\infty}^{\infty}f(t)g(x-t)dt$$

離散関数

$$(f*g)(n)=\sum_{k=-\infty}^{\infty}f(k)g(n-k)$$

2次元Convolution(畳み込み)

二つの関数 $f(x,y)$ $g(x,y)$ を重ね合わせる演算
$f(x,y)$ を固定し，$g(x,y)$ を平行移動しながら$f(x,y)$に掛けあわせ，得られた関数を積分

連続関数 $$(f*g)(x,y)=\int_{-\infty}^{\infty}\int_{-\infty}^{\infty}f(u,v)g(u-x,v-y)dudv$$

離散関数 $$(f*g)(n)=\sum_{j=-\infty}^{\infty}\sum_{i=-\infty}^{\infty}f(i,j)g(i-x,j-y)$$

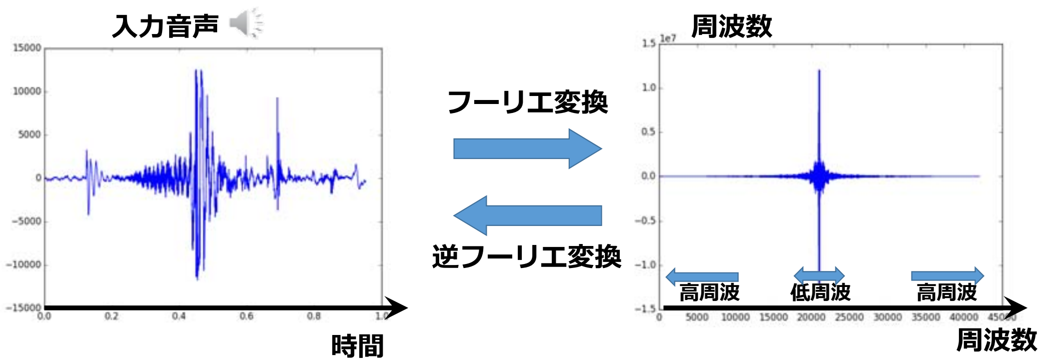
※ 長々と説明しましたが、教科書中の線形フィルタとの違いは，フィルタgの引数の一部がマイナスになっているところだけです。

復習：フーリエ変換

フーリエ変換とは

FourierSound.py

- 横軸が時間の関数を、横軸が周波数の関数に変換する手法

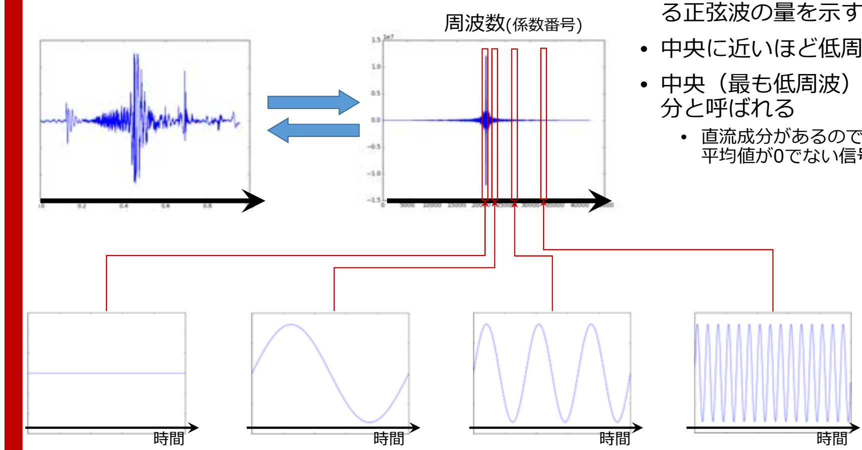


注) グラフ縦軸は音圧

注) グラフ横軸は係数番号（Hzではない）
グラフ縦軸は係数の実部

フーリエ変換とは

FourierSound.py



- フーリエ変換後の関数は元信号に含まれる正弦波の量を示す
- 中央に近いほど低周波，外ほどが高周波
- 中央（最も低周波）は，定数項で直流成分と呼ばれる
  - 直流成分があるので正弦波の組み合わせでも平均値が0でない信号を作れる

※下の波はイメージ
※本来はもっともっと細かいです。

フーリエ級数

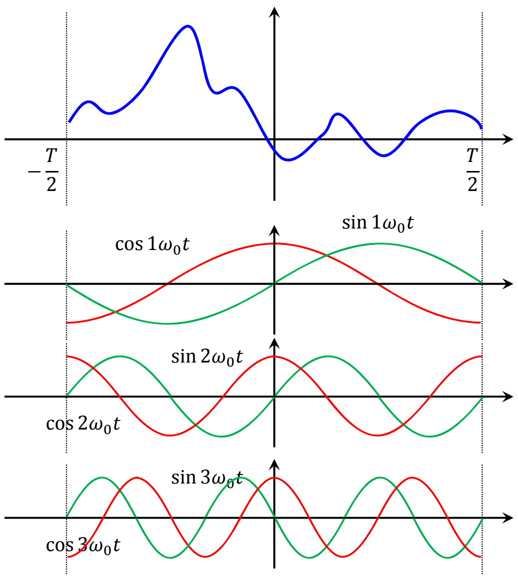
区間$\left[-\frac{T}{2},\frac{T}{2}\right]$上の連続関数$f(t)$は、

フーリエ級数で表現できる.

$$f(t)=\frac{a_0}{2}$$
$$+a_1\cos 1\omega_0 t+b_1\sin 1\omega_0 t$$
$$+a_2\cos 2\omega_0 t+b_2\sin 2\omega_0 t$$
$$+a_3\cos 3\omega_0 t+b_3\sin 3\omega_0 t$$
$$+\cdots$$

基本周波数

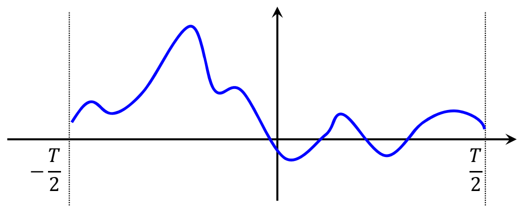
$$\omega_0=\frac{2\pi}{T}$$



フーリエ級数(**複素数表記**)

区間$\left[-\frac{T}{2},\frac{T}{2}\right]$上の連続関数$f(t)$は、

フーリエ級数で表現できる.



$$f(t)=\sum_{k=-\infty}^{\infty}C_k e^{ik\omega_0 t}$$

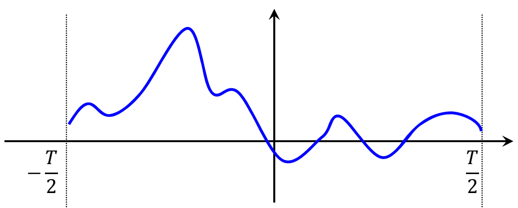
$$C_k=\frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}}f(t)e^{-ik\omega_0 t}dt$$

この複素数表記された
正弦波を重ね合わせていることは分かるんだけど、
$\cos k\omega_0 t\,,\sin k\omega_0 t$ に比べてイメージしにくい

フーリエ級数(**複素数表記**)

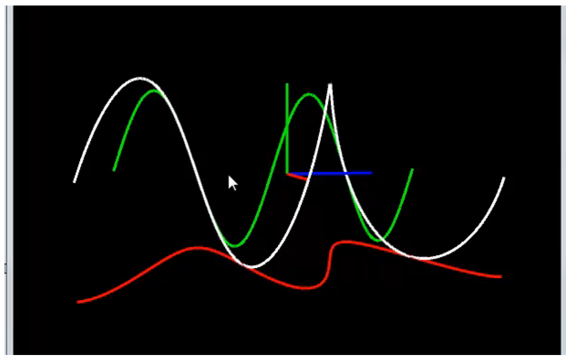
区間$\left[-\frac{T}{2},\frac{T}{2}\right]$上の連続関数$f(t)$は、

フーリエ級数で表現できる.



$$f(t)=\sum_{k=-\infty}^{\infty}C_k e^{ik\omega_0 t}$$

$$C_k=\frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}}f(t)e^{-ik\omega_0 t}dt$$



赤が実軸　　緑が虚軸　　青が時間軸

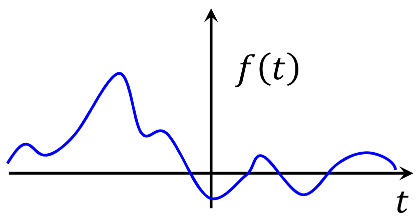
フーリエ変換とは

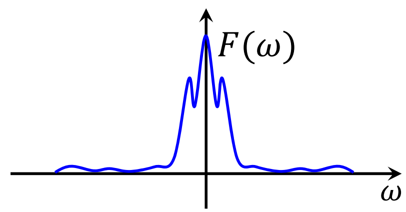
フーリエ変換：

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

逆フーリエ変換：

$$f(t)=\frac{1}{2\pi}\int_{-\infty}^{\infty}F(\omega)e^{i\omega t}\,d\omega$$





- 時間tの関数 $f(t)$ を、周波数ωの関数$F(\omega)$に変換する
- $f(t)$と$F(\omega)$は複素数関数である（$f(t)$は実数関数のことが多い）
- フーリエ級数展開において T → ∞ とすると導出できる

ガウス関数のフーリエ変換

導出も大切だけど、結論がとにかく大切なので覚えてほしい！

- ガウス関数をフーリエ変換するとガウス関数になる
- 標準偏差は逆数になる（裾の広い関数はとがった関数に）
- 虚部はゼロになる

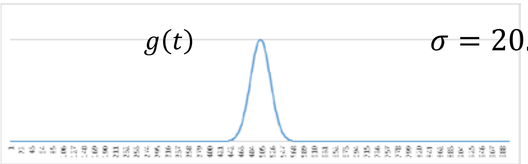
ガウス関数：

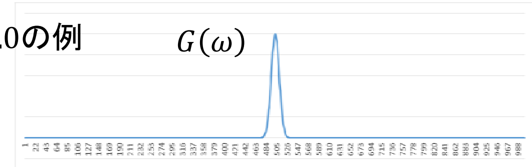
$$g(t) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{t^2}{2\sigma^2}}$$

フーリエ変換：

$$G(\omega) = e^{-\frac{\omega^2\sigma^2}{2}}$$

$\sigma = 20.0$の例





ガウス関数のフーリエ変換(導出)

$$G(\omega) = \frac{1}{\sqrt{2\pi}\sigma}\int_{-\infty}^{\infty} e^{-\frac{t^2}{2\sigma^2}} e^{-i\omega t}\mathrm{d}t$$ （定義より）

$$\frac{dG(\omega)}{d\omega} = \frac{1}{\sqrt{2\pi}\sigma}\int_{-\infty}^{\infty} e^{-\frac{t^2}{2\sigma^2}} \frac{de^{-i\omega t}}{d\omega}\mathrm{d}t$$ （両辺微分）

$$= \frac{1}{\sqrt{2\pi}\sigma}\int_{-\infty}^{\infty} e^{-\frac{t^2}{2\sigma^2}} (-it) e^{-i\omega t}\mathrm{d}t$$ （微分実行）

$$= \frac{1}{\sqrt{2\pi}\sigma}\sigma^2\int_{-\infty}^{\infty} \left(-\frac{2t}{2\sigma^2}\right) e^{-\frac{t^2}{2\sigma^2}}\, ie^{-i\omega t}\mathrm{d}t$$ （整理・部分積分準備）

$$= \frac{1}{\sqrt{2\pi}\sigma}\sigma^2\left(\left[e^{-\frac{t^2}{2\sigma^2}}\, ie^{-i\omega t}\right]_{-\infty}^{\infty} - \int_{-\infty}^{\infty} e^{-\frac{t^2}{2\sigma^2}}\,\omega e^{-i\omega t}\mathrm{d}t\right)$$ （部分積分）

$$= -\,\omega\sigma^2 G(\omega)$$ （第一項はゼロ、第二項はG(w)なので）

$$\frac{dG(\omega)}{d\omega} = -\,\omega\sigma^2 G(\omega)$$

ガウス関数のフーリエ変換(導出)

$$\frac{dG(\omega)}{d\omega} = -\,\omega\sigma^2 G(\omega)$$ （これは一階の微分方程式なので変数分離で解ける）

$$\frac{dG(\omega)}{G(\omega)} = -\,\omega\sigma^2 d\omega$$ （変数分離）

$$\log G(\omega) = -\frac{1}{2}\omega^2\sigma^2 + C$$ （両辺を積分、Cは積分定数）

$$G(\omega) = e^C e^{-\frac{1}{2}\omega^2\sigma^2}$$ （整理）

$$G(0) = e^C = \frac{1}{\sqrt{2\pi}\sigma}\int_{-\infty}^{\infty} e^{-\frac{t^2}{2\sigma^2}}\mathrm{d}t = 1$$ （積分定数を決める、有名な積分公式利用）

$$G(\omega) = e^{-\frac{\omega^2\sigma^2}{2}}$$ （求まった積分定数を代入して，フーリエ変換が得られた）

畳み込み積分のフーリエ変換

導出も大切だけど、結論がとにかく大切なので覚えてほしい！

$$h(x) = \int_{-\infty}^{\infty} f(x-t)g(t)\mathrm{d}t \qquad H(\omega) = F(\omega)G(\omega)$$

$H(\omega)$, $F(\omega)$, $G(\omega)$は, $h(x)$, $h(x)$, $h(x)$のフーリエ変換

- 畳み込み積分のフーリエ変換は、フーリエ変換後の積になる
- 用途例
  - 畳み込みは処理時間がかかる → O(N^2)
  - フーリエ変換して（FFTならO($NlogN$)），周波数空間で積を計算し（O(N)），逆フーリエ変換（ O($NlogN$) ）→ O($NlogN$)
  - $(f * g)(x)$を計算するより $\mathcal{F}^{-1}\{\mathcal{F}[f]\,\mathcal{F}[g]\}$を計算したほうが速いことがある

畳み込み積分のフーリエ変換（導出）

導出も大切だけど、結論がとにかく大切なので覚えてほしい！

$$h(x) = \int_{-\infty}^{\infty} f(x-t)g(t)\mathrm{d}t$$ （畳み込み積分の定義）

$$H(\omega) = \int_{-\infty}^{\infty}\left(\int_{-\infty}^{\infty} f(x-t)g(t)\mathrm{d}t\right)e^{-ix\omega}\,dx$$ （hをフーリエ変換）

$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x-t)g(t)\,e^{-ix\omega}\,dtdx$$ （整理）

$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x-t)g(t)\,e^{-i(x-t)\omega}e^{-it\omega}\,dtdx$$ （さらに整理、少し技巧的）

$$= \int_{-\infty}^{\infty} f(x-t)e^{-i(x-t)\omega}dx\int_{-\infty}^{\infty} g(t)e^{-it\omega}\,\mathrm{d}t$$ （x関連とt関連の項に分離）

$$= F(\omega)G(\omega)$$

フーリエ変換（復習）のまとめ

フーリエ変換: 時間tの関数 $f(t)$ を周波数ωの関数$F(\omega)$に変換する変換

$$F(\omega) = \int_{-\infty}^{\infty} f(t)e^{-i\omega t}dt \qquad f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(\omega)e^{i\omega t}\,d\omega$$

ガウス関数をフーリエ変換するとガウス関数になる

$$g(t) = \frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{t^2}{2\sigma^2}} \qquad G(\omega) = e^{-\frac{\omega^2\sigma^2}{2}}$$

畳み込み積分のフーリエ変換はフーリエ変換の積になる

$$h(x) = \int_{-\infty}^{\infty} f(x-t)g(t)\mathrm{d}t \qquad H(\omega) = F(\omega)G(\omega)$$

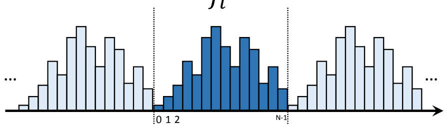
$H(\omega)$, $F(\omega)$, $G(\omega)$は, $h(x)$, $h(x)$, $h(x)$のフーリエ変換

離散フーリエ変換（1D）

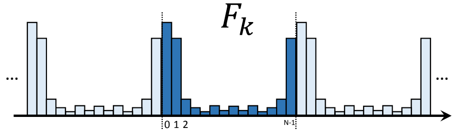
フーリエ変換 $$F_k = \frac{1}{N}\sum_{l=0}^{N-1} f_l e^{-i\frac{2\pi kl}{N}}$$

逆フーリエ変換 $$f_l = \sum_{k=0}^{N-1} F_k e^{i\frac{2\pi kl}{N}}$$

f_l



F_k



- 周期Nの離散値f_lを周期Nの離散値F_kに変換する
- f_lとF_kは複素数（**ただしf_lは実数列のことが多い**）
- f_lが実数の場合$F_k = \overline{F_{-k}}$が成り立つ（$F_{-k} = F_{N-k}$）

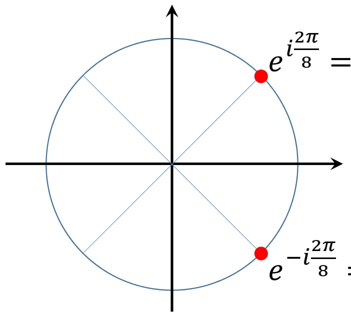
少し余談（FFTの簡単な説明）

N=8のときの離散フーリエ変換を考えてみる

$$F_k=\frac{1}{8}\sum_{l=0}^{7} f_l \boxed{e^{-i\frac{2\pi}{8}kl}}$$

この $e^{-i\frac{2\pi}{8}kl}$ は何？

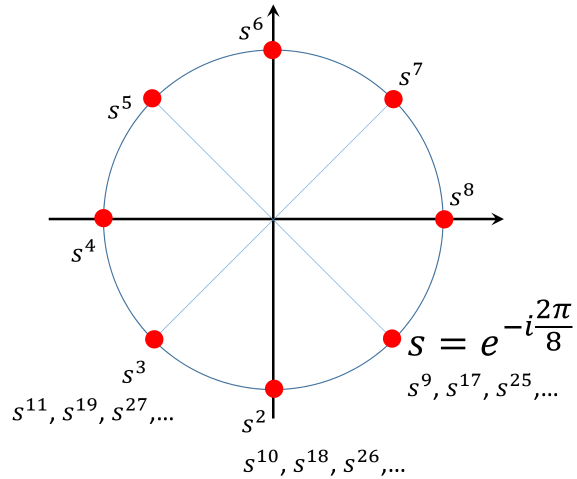
→$-kl$を無視した$e^{i\frac{2\pi}{8}}$は1の8乗根



$$e^{i\frac{2\pi}{8}}=cos\frac{2\pi}{8}+isin\frac{2\pi}{8}$$

$$e^{-i\frac{2\pi}{8}}=\cos\frac{2\pi}{8}-i\sin\frac{2\pi}{8}$$

$s=e^{-i\frac{2\pi}{8}}$とおくと…



$$F_k=\frac{1}{8}\sum_{l=0}^{7} f_l e^{-i\frac{2\pi}{8}kl}$$

N=8のときの離散フーリエ変換を、根性で書き下してみる

$s=e^{-i\frac{2\pi}{8}}$を利用すると以外に綺麗な形に

$$\begin{pmatrix}F_0\\F_1\\F_2\\F_3\\F_4\\F_5\\F_6\\F_7\end{pmatrix}=\frac{1}{8}\begin{pmatrix}s^0&s^0&s^0&s^0&s^0&s^0&s^0&s^0\\s^0&s^1&s^2&s^3&s^4&s^5&s^6&s^7\\s^0&s^2&s^4&s^6&s^8&s^{10}&s^{12}&s^{14}\\s^0&s^3&s^6&s^9&s^{12}&s^{15}&s^{18}&s^{21}\\s^0&s^4&s^8&s^{12}&s^{16}&s^{20}&s^{24}&s^{28}\\s^0&s^5&s^{10}&s^{15}&s^{20}&s^{25}&s^{30}&s^{35}\\s^0&s^6&s^{12}&s^{18}&s^{24}&s^{30}&s^{36}&s^{42}\\s^0&s^7&s^{14}&s^{21}&s^{28}&s^{35}&s^{42}&s^{49}\end{pmatrix}\begin{pmatrix}f_0\\f_1\\f_2\\f_3\\f_4\\f_5\\f_6\\f_7\end{pmatrix}$$

行列自体は前計算可能

行列と $(f_0, f_1, f_2,... f_7)$の積には，複素数の掛け算が 8×8回必用 → $O(N^2)$

Nが2のべき乗のとき，これを$O(N\ log\ N)$で計算できる!!

$$\begin{pmatrix}F_0\\F_1\\F_2\\F_3\\F_4\\F_5\\F_6\\F_7\end{pmatrix}=\frac{1}{8}\begin{pmatrix}s^0&s^0&s^0&s^0&s^0&s^0&s^0&s^0\\s^0&s^1&s^2&s^3&s^4&s^5&s^6&s^7\\s^0&s^2&s^4&s^6&s^8&s^{10}&s^{12}&s^{14}\\s^0&s^3&s^6&s^9&s^{12}&s^{15}&s^{18}&s^{21}\\s^0&s^4&s^8&s^{12}&s^{16}&s^{20}&s^{24}&s^{28}\\s^0&s^5&s^{10}&s^{15}&s^{20}&s^{25}&s^{30}&s^{35}\\s^0&s^6&s^{12}&s^{18}&s^{24}&s^{30}&s^{36}&s^{42}\\s^0&s^7&s^{14}&s^{21}&s^{28}&s^{35}&s^{42}&s^{49}\end{pmatrix}\begin{pmatrix}f_0\\f_1\\f_2\\f_3\\f_4\\f_5\\f_6\\f_7\end{pmatrix}$$

$s=e^{-i\frac{2\pi}{8}}$ なので

$s^{k+8n}=s^k$ が成り立つ

$$\begin{pmatrix}F_0\\F_1\\F_2\\F_3\\F_4\\F_5\\F_6\\F_7\end{pmatrix}=\frac{1}{8}\begin{pmatrix}s^0&s^0&s^0&s^0&s^0&s^0&s^0&s^0\\s^0&s^1&s^2&s^3&s^4&s^5&s^6&s^7\\s^0&s^2&s^4&s^6&s^0&s^2&s^4&s^6\\s^0&s^3&s^6&s^1&s^4&s^7&s^2&s^5\\s^0&s^4&s^0&s^4&s^0&s^4&s^0&s^4\\s^0&s^5&s^2&s^7&s^4&s^1&s^6&s^3\\s^0&s^6&s^4&s^2&s^0&s^6&s^4&s^2\\s^0&s^7&s^6&s^5&s^4&s^3&s^2&s^1\end{pmatrix}\begin{pmatrix}f_0\\f_1\\f_2\\f_3\\f_4\\f_5\\f_6\\f_7\end{pmatrix}$$

乗数が7以下になるよう変形

なんか繰り返しが見える…

$$\begin{pmatrix}F_0\\F_1\\F_2\\F_3\\F_4\\F_5\\F_6\\F_7\end{pmatrix}=\frac{1}{8}\begin{pmatrix}
s^0&s^0&s^0&s^0&s^0&s^0&s^0&s^0\\
s^0&s^1&s^2&s^3&s^4&s^5&s^6&s^7\\
s^0&s^2&s^4&s^6&s^0&s^2&s^4&s^6\\
s^0&s^3&s^6&s^1&s^4&s^7&s^2&s^5\\
s^0&s^4&s^0&s^4&s^0&s^4&s^0&s^4\\
s^0&s^5&s^2&s^7&s^4&s^1&s^6&s^3\\
s^0&s^6&s^4&s^2&s^0&s^6&s^4&s^2\\
s^0&s^7&s^6&s^5&s^4&s^3&s^2&s^1
\end{pmatrix}\begin{pmatrix}f_0\\f_1\\f_2\\f_3\\f_4\\f_5\\f_6\\f_7\end{pmatrix}$$

偶数列に注目して…
偶数列が先に、
奇数列が後に
なるよう入れ替える

$$\begin{pmatrix}F_0\\F_1\\F_2\\F_3\\F_4\\F_5\\F_6\\F_7\end{pmatrix}=\frac{1}{8}\begin{pmatrix}
s^0&s^0&s^0&s^0&s^0&s^0&s^0&s^0\\
s^0&s^2&s^4&s^6&s^1&s^3&s^5&s^7\\
s^0&s^4&s^0&s^4&s^2&s^6&s^2&s^6\\
s^0&s^6&s^4&s^2&s^3&s^1&s^7&s^5\\
s^0&s^0&s^0&s^0&s^4&s^4&s^4&s^4\\
s^0&s^2&s^4&s^6&s^5&s^7&s^1&s^3\\
s^0&s^4&s^0&s^4&s^6&s^2&s^6&s^2\\
s^0&s^6&s^4&s^2&s^7&s^5&s^3&s^1
\end{pmatrix}\begin{pmatrix}f_0\\f_2\\f_4\\f_6\\f_1\\f_3\\f_5\\f_7\end{pmatrix}$$

$$\begin{pmatrix}F_0\\F_1\\F_2\\F_3\\F_4\\F_5\\F_6\\F_7\end{pmatrix}=\frac{1}{8}\begin{pmatrix}
s^0&s^0&s^0&s^0&s^0&s^0&s^0&s^0\\
s^0&s^2&s^4&s^6&s^1&s^3&s^5&s^7\\
s^0&s^4&s^0&s^4&s^2&s^6&s^2&s^6\\
s^0&s^6&s^4&s^2&s^3&s^1&s^7&s^5\\
s^0&s^0&s^0&s^0&s^4&s^4&s^4&s^4\\
s^0&s^2&s^4&s^6&s^5&s^7&s^1&s^3\\
s^0&s^4&s^0&s^4&s^6&s^2&s^6&s^2\\
s^0&s^6&s^4&s^2&s^7&s^5&s^3&s^1
\end{pmatrix}\begin{pmatrix}f_0\\f_2\\f_4\\f_6\\f_1\\f_3\\f_5\\f_7\end{pmatrix}$$

この行列をじっくり眺める

この部分は同じ！　この部分は…

$$\begin{pmatrix}
s^0&s^0&s^0&s^0\\
s^1&s^3&s^5&s^7\\
s^2&s^6&s^2&s^6\\
s^3&s^1&s^7&s^5\\
s^4&s^4&s^4&s^4\\
s^5&s^7&s^1&s^3\\
s^6&s^2&s^6&s^2\\
s^7&s^5&s^3&s^1
\end{pmatrix}\Rightarrow
\begin{matrix}
s^0(&s^0&s^0&s^0&s^0&)\\
s^1(&s^0&s^2&s^4&s^6&)\\
s^2(&s^0&s^4&s^0&s^4&)\\
s^3(&s^0&s^6&s^4&s^2&)\\
s^4(&s^0&s^0&s^0&s^0&)\\
s^5(&s^0&s^2&s^4&s^6&)\\
s^6(&s^0&s^4&s^0&s^4&)\\
s^7(&s^0&s^6&s^4&s^2&)
\end{matrix}$$

黄色い部分の
繰り返しが隠れている!

$$\begin{pmatrix}F_0\\F_1\\F_2\\F_3\\F_4\\F_5\\F_6\\F_7\end{pmatrix}=\frac{1}{8}\begin{pmatrix}
s^0&s^0&s^0&s^0&s^0&s^0&s^0&s^0\\
s^0&s^2&s^4&s^6&s^1&s^3&s^5&s^7\\
s^0&s^4&s^0&s^4&s^2&s^6&s^2&s^6\\
s^0&s^6&s^4&s^2&s^3&s^1&s^7&s^5\\
s^0&s^0&s^0&s^0&s^4&s^4&s^4&s^4\\
s^0&s^2&s^4&s^6&s^5&s^7&s^1&s^3\\
s^0&s^4&s^0&s^4&s^6&s^2&s^6&s^2\\
s^0&s^6&s^4&s^2&s^7&s^5&s^3&s^1
\end{pmatrix}\begin{pmatrix}f_0\\f_2\\f_4\\f_6\\f_1\\f_3\\f_5\\f_7\end{pmatrix}$$

先の繰り返し構造を利用して
以下の通り変形できる

$$=\frac{1}{8}\left(\begin{array}{cccc|cccc}
1&0&0&0&s^0&0&0&0\\
0&1&0&0&0&s^1&0&0\\
0&0&1&0&0&0&s^2&0\\
0&0&0&1&0&0&0&s^3\\ \hline
1&0&0&0&s^4&0&0&0\\
0&1&0&0&0&s^5&0&0\\
0&0&1&0&0&0&s^6&0\\
0&0&0&1&0&0&0&s^7
\end{array}\right)\begin{pmatrix}
s^0&s^0&s^0&s^0&0&0&0&0\\
s^0&s^2&s^4&s^6&0&0&0&0\\
s^0&s^4&s^0&s^4&0&0&0&0\\
s^0&s^6&s^4&s^2&0&0&0&0\\
0&0&0&0&s^0&s^0&s^0&s^0\\
0&0&0&0&s^0&s^2&s^4&s^6\\
0&0&0&0&s^0&s^4&s^0&s^4\\
0&0&0&0&s^0&s^6&s^4&s^2
\end{pmatrix}\begin{pmatrix}f_0\\f_2\\f_4\\f_6\\f_1\\f_3\\f_5\\f_7\end{pmatrix}$$

複素数
掛け算
回数

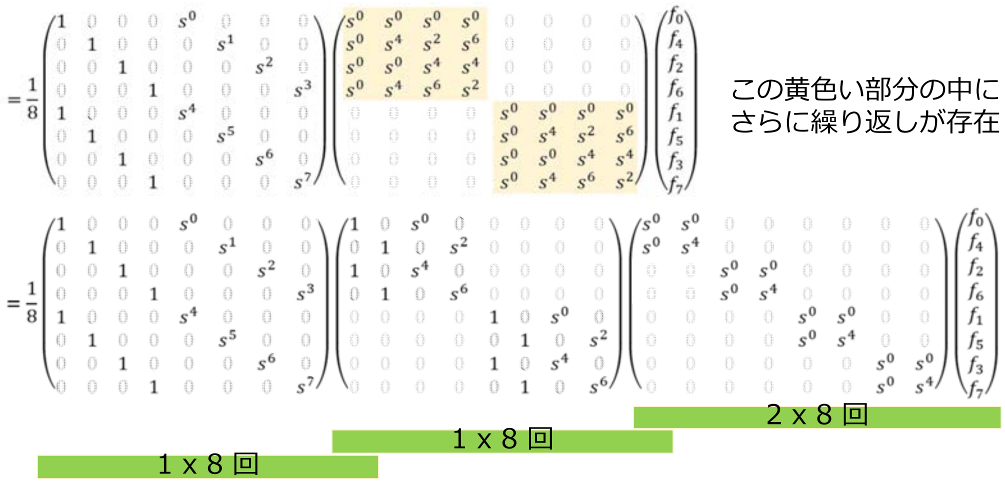
1 x 8 回(疎行列 x ベクトル)

4 x 8 回(疎行列 x ベクトル)

$$=\frac{1}{8}\left(\begin{array}{cccc|cccc}
1&0&0&0&s^0&0&0&0\\
0&1&0&0&0&s^1&0&0\\
0&0&1&0&0&0&s^2&0\\
0&0&0&1&0&0&0&s^3\\ \hline
1&0&0&0&s^4&0&0&0\\
0&1&0&0&0&s^5&0&0\\
0&0&1&0&0&0&s^6&0\\
0&0&0&1&0&0&0&s^7
\end{array}\right)\begin{pmatrix}
s^0&s^0&s^0&s^0&0&0&0&0\\
s^0&s^2&s^4&s^6&0&0&0&0\\
s^0&s^4&s^0&s^4&0&0&0&0\\
s^0&s^6&s^4&s^2&0&0&0&0\\
0&0&0&0&s^0&s^0&s^0&s^0\\
0&0&0&0&s^0&s^2&s^4&s^6\\
0&0&0&0&s^0&s^4&s^0&s^4\\
0&0&0&0&s^0&s^6&s^4&s^2
\end{pmatrix}\begin{pmatrix}f_0\\f_2\\f_4\\f_6\\f_1\\f_3\\f_5\\f_7\end{pmatrix}$$

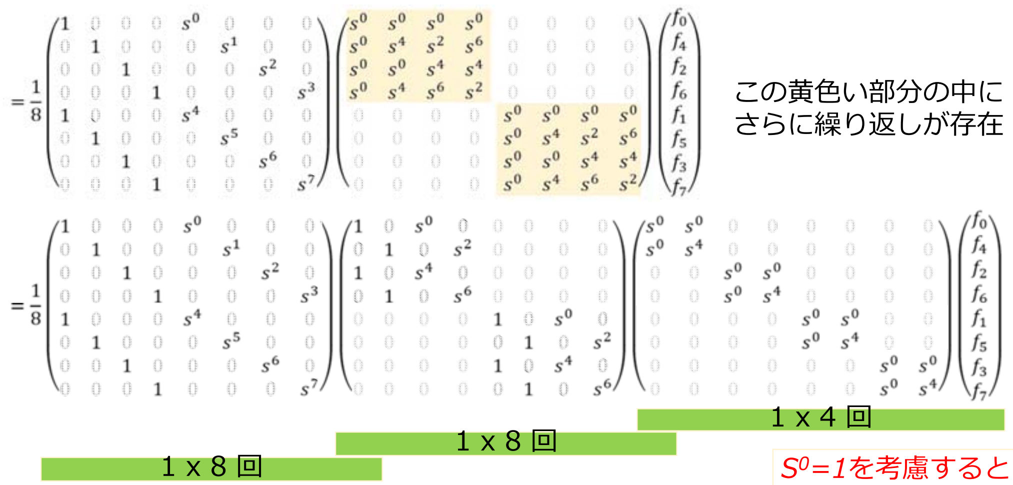
$$=\frac{1}{8}\left(\begin{array}{cccc|cccc}
1&0&0&0&s^0&0&0&0\\
0&1&0&0&0&s^1&0&0\\
0&0&1&0&0&0&s^2&0\\
0&0&0&1&0&0&0&s^3\\ \hline
1&0&0&0&s^4&0&0&0\\
0&1&0&0&0&s^5&0&0\\
0&0&1&0&0&0&s^6&0\\
0&0&0&1&0&0&0&s^7
\end{array}\right)\begin{pmatrix}
s^0&s^0&s^0&s^0&0&0&0&0\\
s^0&s^4&s^2&s^6&0&0&0&0\\
s^0&s^0&s^4&s^4&0&0&0&0\\
s^0&s^4&s^6&s^2&0&0&0&0\\
0&0&0&0&s^0&s^0&s^0&s^0\\
0&0&0&0&s^0&s^4&s^2&s^6\\
0&0&0&0&s^0&s^0&s^4&s^4\\
0&0&0&0&s^0&s^4&s^6&s^2
\end{pmatrix}\begin{pmatrix}f_0\\f_4\\f_2\\f_6\\f_1\\f_5\\f_3\\f_7\end{pmatrix}$$

偶数列を先に
奇数列を後になるよう
入れ替える



この黄色い部分の中に
さらに繰り返しが存在

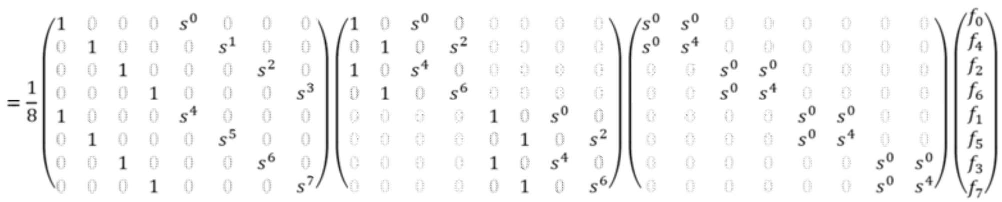
1x8 + 1x8 + 2x8 :
2分割の回数だけ1x8 が生じるので O(N log N) に



この黄色い部分の中に
さらに繰り返しが存在

$S^0=1$を考慮すると
掛け算は1x4回

1x8 + 1x8 + 2x8 :
2分割の回数だけ1x8 が生じるので O(N log N) に

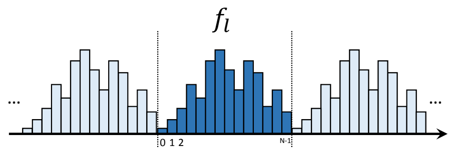


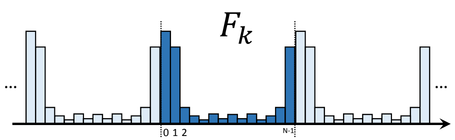
- 『 NxN 疎行列　×　N次元 ベクトル』の繰り返しである
- NxN 疎行列の各行には『1』と s^k が一つだけ含まれる
- 行列xベクトル 演算は,
  - ベクトルから2個選んで
  - 片方をそのまま、もう片方に s^k をかけて足し合わせる

→ バタフライ演算がうっすらと見えてきませんか？

まとめ: 離散フーリエ変換

フーリエ変換 $F_k = \frac{1}{N}\sum_{l=0}^{N-1} f_l e^{-i\frac{2\pi kl}{N}}$　　逆フーリエ変換 $f_l = \sum_{k=0}^{N-1} F_k e^{i\frac{2\pi kl}{N}}$



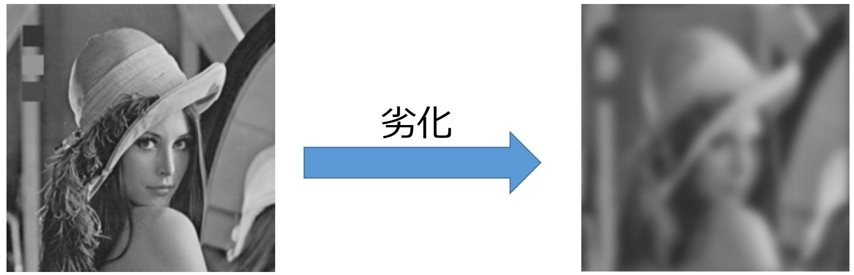


- 周期Nの離散値f_lを周期Nの離散値F_kに変換する
- Nが2のべき乗のとき高速フーリエ変換が適用可能 → O(N log N)に！

Decovolution

画像の劣化モデル（1）

手ブレ・ピンボケ・撮影機器のノイズ等のため劣化した画像が取得される
劣化前画像復元のため劣化課程をモデル化（数式表現）する



$f(x,y)$：劣化の無い理想画像
※ピンホールカメラ動きの無いシーンを撮影すると取得可能

$g(x,y)$：劣化画像
※ 手ブレ・ピンボケ・ノイズを含む

画像の劣化モデル（2）

画像の劣化モデルは以下の式で与えられる

$$g(x,y) = h(x,y) * f(x,y) + n(x,y)$$

$g(x,y)$ = $h(x,y)$ * $f(x,y)$ + $n(x,y)$

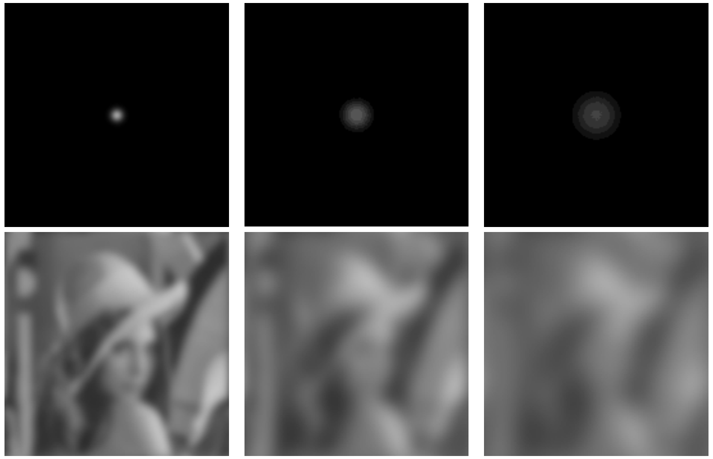
『元画像にカーネル$h(x,y)$を畳み込みノイズ$n(x,y)$が加算されて』画像が劣化する
$h(x,y)$は，ボケの様子を表し**点広がり関数**と呼ばれる

点広がり関数 (PSF: point spread function)

$g(x,y)$ = $h(x,y)$ * $f(x,y)$ + $n(x,y)$

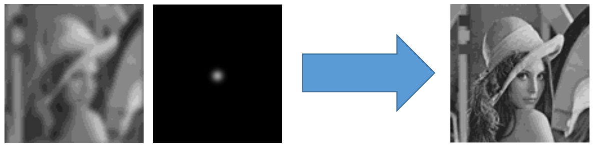
- 画像劣化時に元画像に畳み込まれている関数$h(x,y)$のこと
- 劣化の特性を表す
- 元画像が点光源のとき、劣化画像に表れる応答を表すため、点広がり関数（PSD）やインパルス応答と呼ばれる

劣化画像例



劣化画像例

劣化画像の復元　（単純な手法）



- 劣化画像と点広がり関数から元画像を取得する問題を考える

$g(x,y) = h(x,y) * f(x,y) + n(x,y)$　gとhは既知で f がほしい

$G(u,v) = H(u,v)F(u,v) + N(u,v)$　両辺をフーリエ変換（大文字で表現）

$G(u,v) \approx H(u,v)F(u,v)$　ちょっと強引だけどノイズの影響を無視

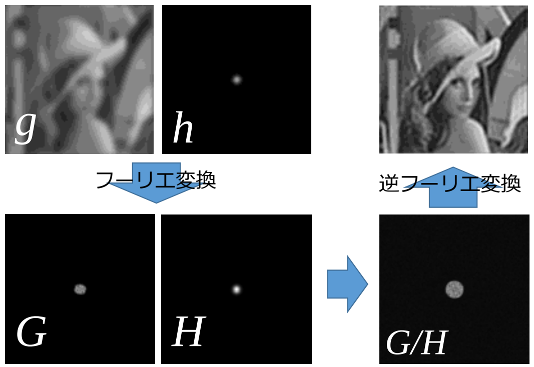
$F(u,v) \approx \frac{1}{H(u,v)} G(u,v)$　Fについて整理

$f(x,y) \approx \mathcal{F}^{-1}\left(\frac{1}{H(u,v)} G(u,v)\right)$　両辺逆フーリエ変換

劣化画像の復元　（単純な手法）

$$f(x,y) \approx \mathcal{F}^{-1}\left(\frac{1}{H(u,v)} G(u,v)\right)$$

f : 元画像
G : 劣化画像のフーリエ変換
H : 点広がり関数のフーリエ変換



この手法の問題
- ノイズを無視
- Hは高周波部分でほぼゼロ

→ 単純に G/H を実装するとノイズが強調される

左の例ではHを以下の通り拡張した

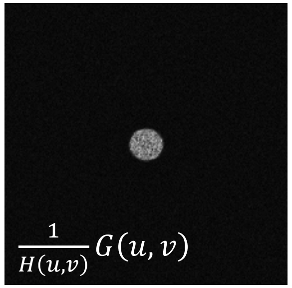
$$H'(u,v) = \begin{cases} t & H'^{(u,v)} < t \\ H(u,v) & otherwise \end{cases}$$

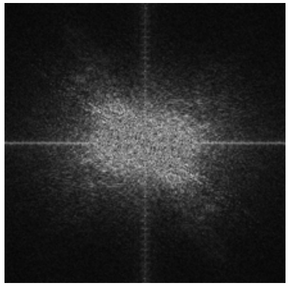
ここでは t=0.01を利用

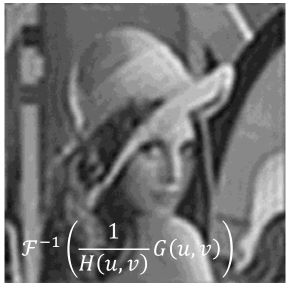
正解との比較

- 右が正解
- 左が復元手法

※*H*に対する閾値処理の影響が見られる
※これを行なわないと高周波成分に存在するノイズが強調され画像はうまく復元されない







劣化画像の復元：Weiner filter

$$f(x,y) \approx \mathcal{F}^{-1}\left(\frac{H^*(u,v)}{|H(u,v)|^2+\epsilon}G(u,v)\right)$$

f：元画像
G：劣化画像のフーリエ変換
H：点広がり関数のフーリエ変換
H^*：共役複素数

先の単純な手法の問題点（ノイズ無視・Hがゼロに近いときに困る）を改善
周波数空間において元画像$F(u,v)$と復元画像$F'(u,v)=M(u,v)G(u,v)$の平均二乗誤差を最小化

$$\underset{M}{\operatorname{argmin}} \sum_u \sum_v \left(F(u,v) - M(u,v)G(u,v)\right)$$

この解として以下が得られる

$$M(u,v) = \frac{H^*(u,v)}{|H(u,v)|^2+|N(u,v)|^2/|F(u,v)|^2}$$

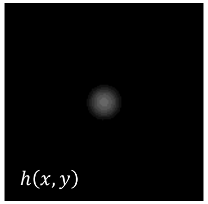
ここで、ノイズNと元画像Fは不明なので$|N(u,v)|^2/|F(u,v)|^2 = \epsilon$ 置いて上の式が得られる

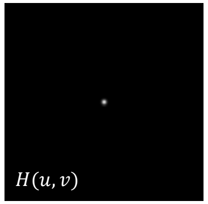
導出の参考 [http://web.engr.oregonstate.edu/~sinisa/courses/OSU/ECE468/lectures/ECE468_13.pdf]

点広がり関数 h のフーリエ変換 H について

hがガウシアンならHもガウシアン
広がりを表すσは逆数になる

$$h(x,y) = \frac{1}{2\pi\sigma^2}e^{-\frac{x^2+y^2}{2\sigma^2}} \quad H(u,v) = e^{-\frac{(u^2+v^2)\sigma^2}{2}}$$

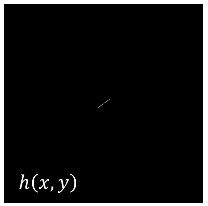


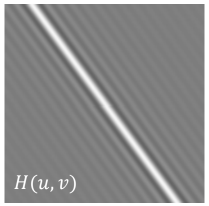


hが手ブレによる線分形状をもつとき
Hはsinc関数に

$$h(x,y) = \begin{cases} \frac{1}{2W} & if\ |x\cos\theta + y\sin\theta| \le w\ \&\ x\sin\theta = y\cos\theta \\ 0 & otherwise \end{cases}$$

$$H(u,v) = \frac{\sin(w(u\cos\theta+v\sin\theta))}{w(u\cos\theta+v\sin\theta)}$$





※ *w*: 線分の長さ、*θ*線分の傾き
※ *note*：実装の際は, *u*,*v*は画素位置に $u' = \frac{2\pi}{W}u,\ v' = \frac{2\pi}{H}v$ と正規化して利用する

Wiener filter 適用例

劣化画像

Wiener filter

単純な手法 1/H

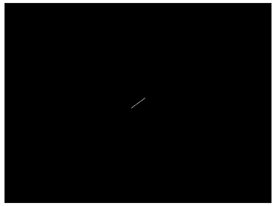
σ = 6のガウシアン
ε = 0.00001

Wiener filter 適用例

劣化画像

Wiener filter

単純な手法 1/H



w = 20, θ=0.8π の線分カーネル
ε = 0.00001

まとめ: Deconvolution

- Deconvolution（逆畳み込み）とは,
  - 『畳み込み(convolution)は，周波数空間では関数どうしの積になる』という特徴を利用し,
  - 畳み込み後の関数 g と **点広がり関数 h** から，劣化画像を復元する処理のこと
- 以下二つのフィルタを紹介した
  - 点広がり関数の逆数を利用した単純なフィルタ
  - ノイズを考慮し**復元画像と元画像の誤差を最小化する Wiener filter**
- 今回は点広がり関数を既知としたが，これも同時に推定する手法もある

$$f(x,y) \approx \mathcal{F}^{-1}\left(\frac{1}{H(u,v)} G(u,v)\right)$$

$$f(x,y) \approx \mathcal{F}^{-1}\left(\frac{H^*(u,v)}{|H(u,v)|^2+\epsilon} G(u,v)\right)$$

f : 元画像
G : 劣化画像のフーリエ変換
H : 点広がり関数のフーリエ変換
H^*: 共役複素数