トーンマッピング
その１

コントラスト補正
及び
輝度階調（トーン）をマップする関数

コントラスト強調

- コントラストとは
  - 「対比・対象」を意味する.
  - 画像処理においては，明るさや色の対比を表す.
- 低コントラストな画像
  - 写真の露出量が適切でない場合，得られる画像は
    全体的に暗い（露出アンダー），または，明るい（露出オーバー）
    低コントラストな画像となる.

低コントラストな画像

補正画像

コントラストに関する画像特徴　その１

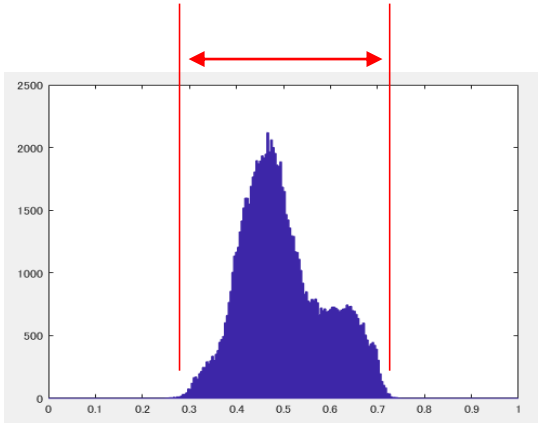
- 大域的なコントラスト：　輝度値の分布範囲が広い
  - 使用されている輝度値の範囲と頻度（すなわちヒストグラム）を調べると，良好な画像では幅広い範囲の輝度値を持つ.
  - 画像全体を大域的に大まかに見たとき，黒色から白色までの輝度が感じられる.

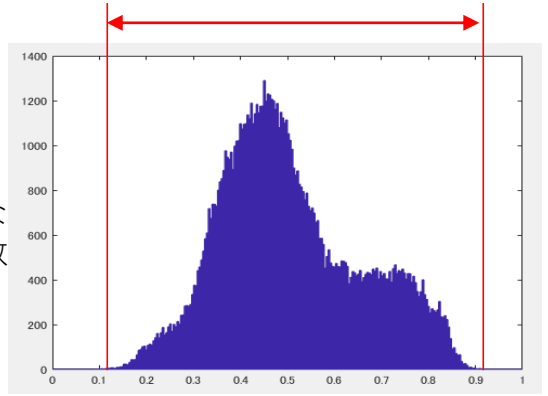
低コントラストな画像

輝度値（YCbCrのY）

良好な画像

輝度値（YCbCrのY）



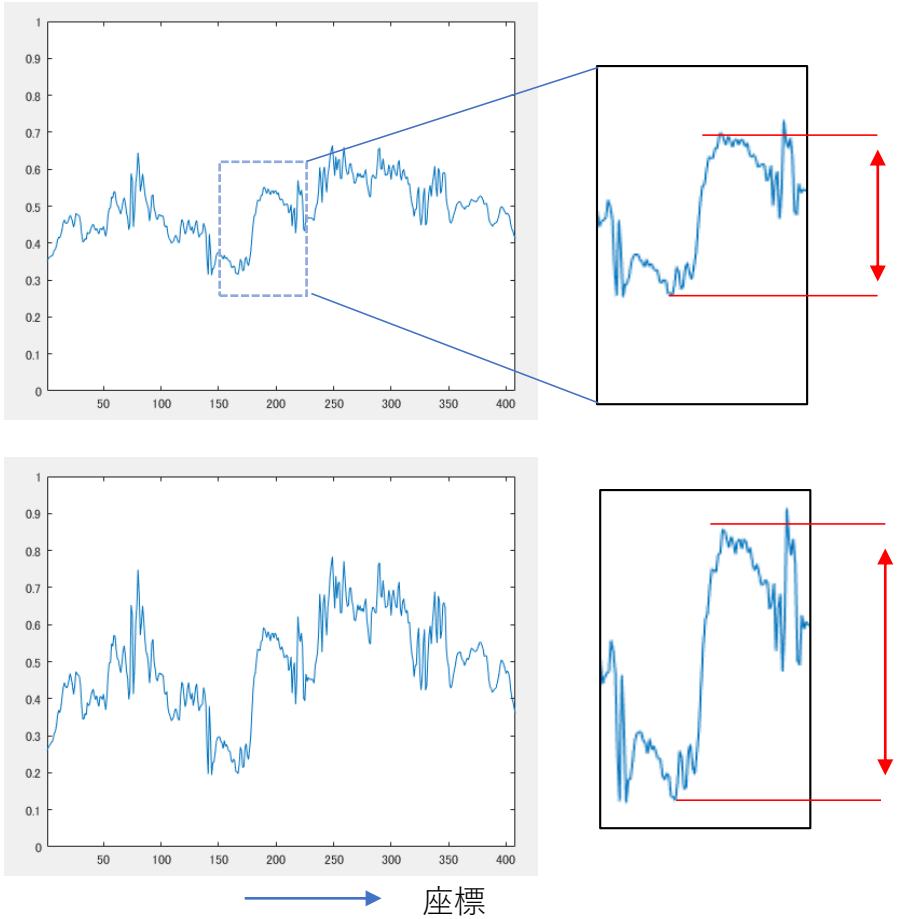


コントラストに関する画像特徴　その２

- 局所的なコントラスト：
  となりあう色領域の輝度差・色差が大きい
  - 図中の白線上の輝度値をプロットすると，良好な画像では輝度差が大きい．

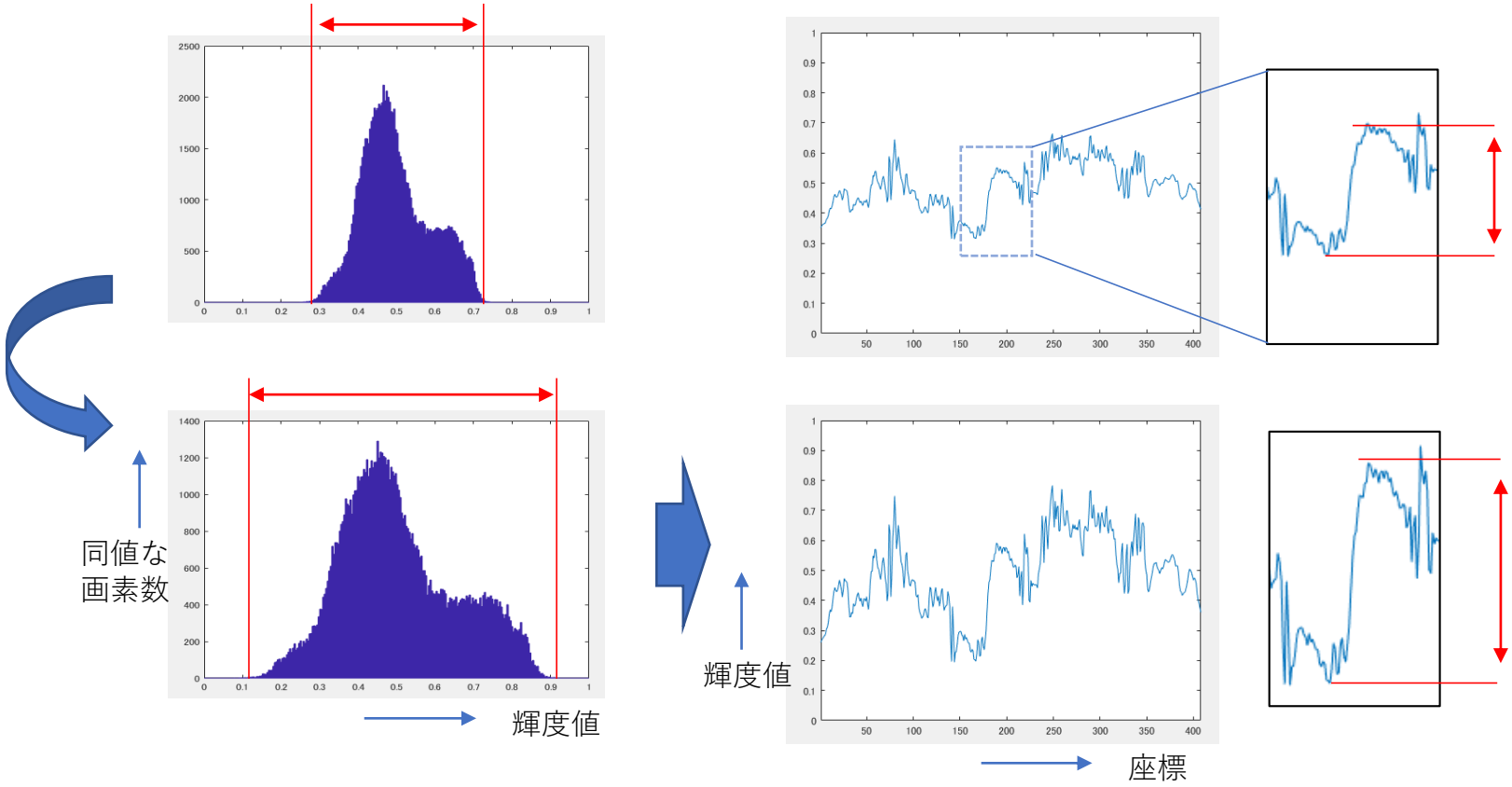
低コントラストな画像の
輝度値（YCbCrのY）

良好な画像の
輝度値（YCbCrのY）



コントラスト補正のアプローチ

- 特徴１を改善することで，特徴２もある程度改善される．
  - 特徴１での輝度範囲が広がれば，
    隣り合う色領域の色差も広がり，結果として，特徴２も改善される．
  - 特徴２を真に改善するには，今後に学習するフィルタリング処理が必要であり，
    未学習であるため，この演習では行わない．



この演習（３回）での目的

- ヒストグラムの幅（輝度値の範囲）を広げる処理について学ぶ.
- トーンマッピング関数
  - 入力画像の輝度値を x として,
    出力画像の輝度値を $f(x)$ として得る関数.
  - どのような関数にすれば良いのか？
    - 線形関数による線形濃度変換
  - 区分的多項式関数による表現.
- 応用
  - ２枚の画像間での色の交換
    一方の画像の色の変化の仕方を，他方の画像に写す.

明るさ（輝度値）のヒストグラム

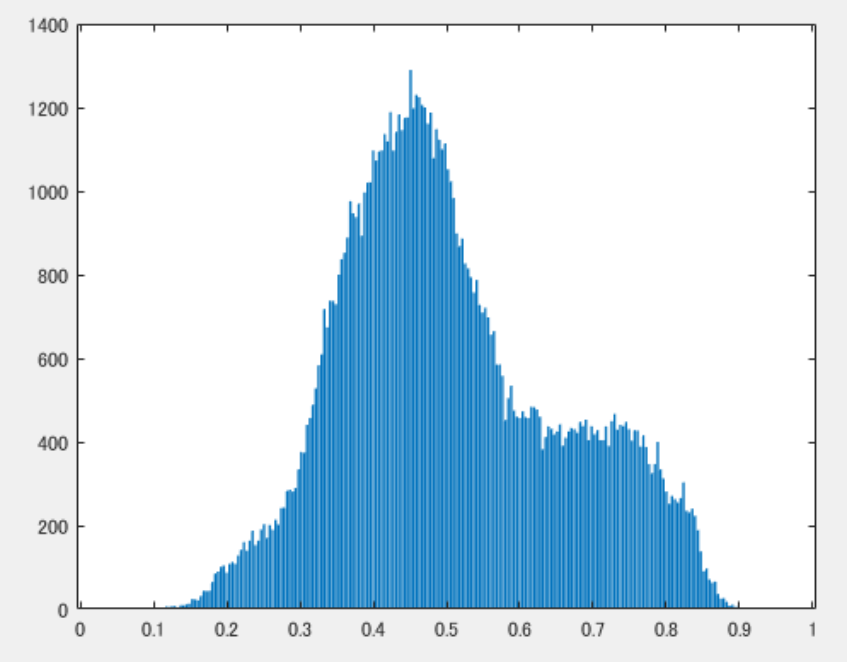
- カラー画像を読み込み，YCbCr 色成分へと変換し，輝度成分 Y を得る．
  - ```
    I = im2double( imread('../images/family.png' ) );
    Iycc = rgb2ycbcr( I );
    Y = Iycc(:,:,1);
    ```
- 輝度のヒストグラムを計算：
  使用輝度の基準値を表すビンを用意し，頻度を計算する．
  - ```
    bin_pos = 0:(1/255):1; % 0～1までを 256 等分
    H = hist( Y(:), bin_pos ); % Y(:) で２次元配列を１次元に変換
    ```
- 結果を表示
  - ```
    figure(1),  imshow( I );
    figure(2),  imshow( Y );
    figure(3),  bar( bin_pos, H );
    ```
    ```

実行結果

- 以下の３つの画像が表示される.

カラー画像

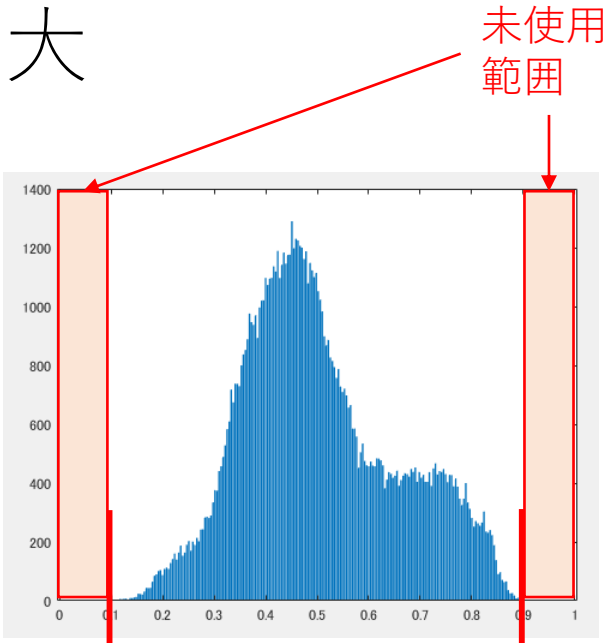
輝度画像
（YCbCr の Y 成分）



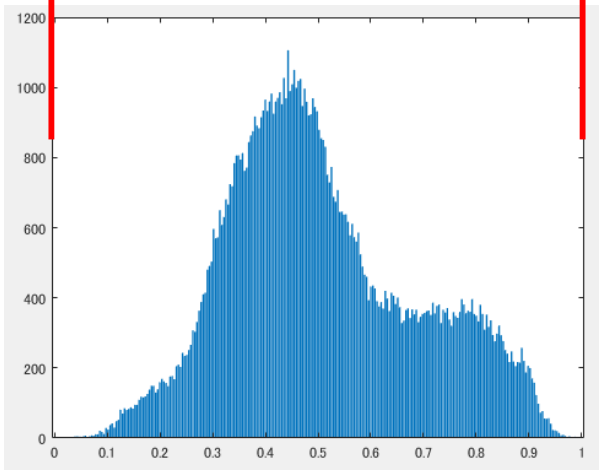
輝度画像のヒストグラム

輝度値の範囲の拡大

- 輝度値の範囲を変更したい
  - 読み込んだ画像（入力画像）では，
    0.1 ～ 0.9 の範囲の輝度値が使われている．
  - 0.1 以下，0.9 以上は未使用である．
  - 0 ～ 1 までの範囲を用いるように
    輝度値を補正し，
    出力画像としたい．
  - どうすれば良いか？



入力画像の
ヒストグラム



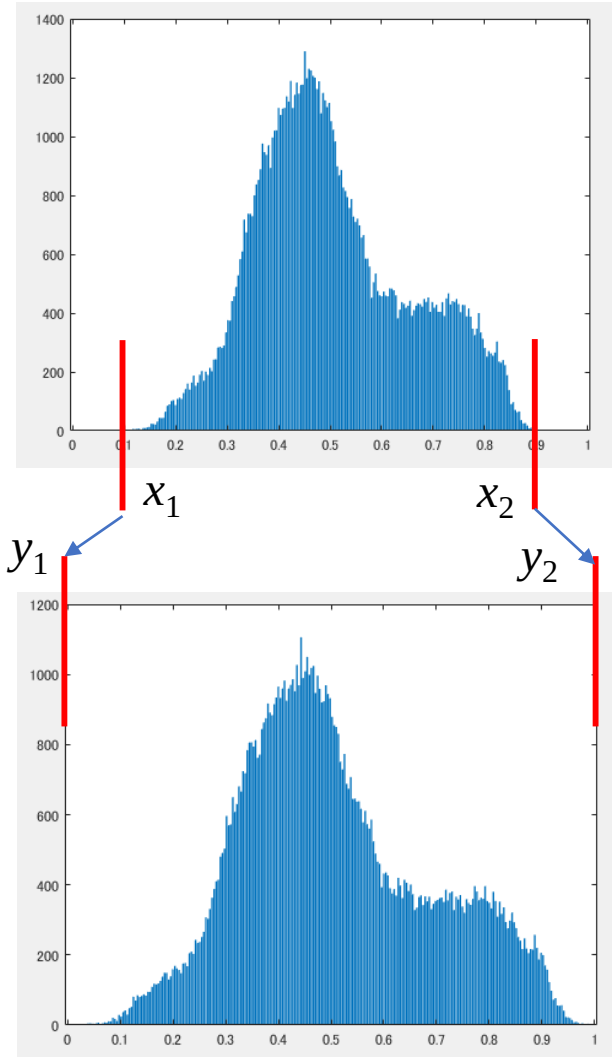
出力画像のヒストグラム

入力・出力輝度値のマッピング関数（１／４）

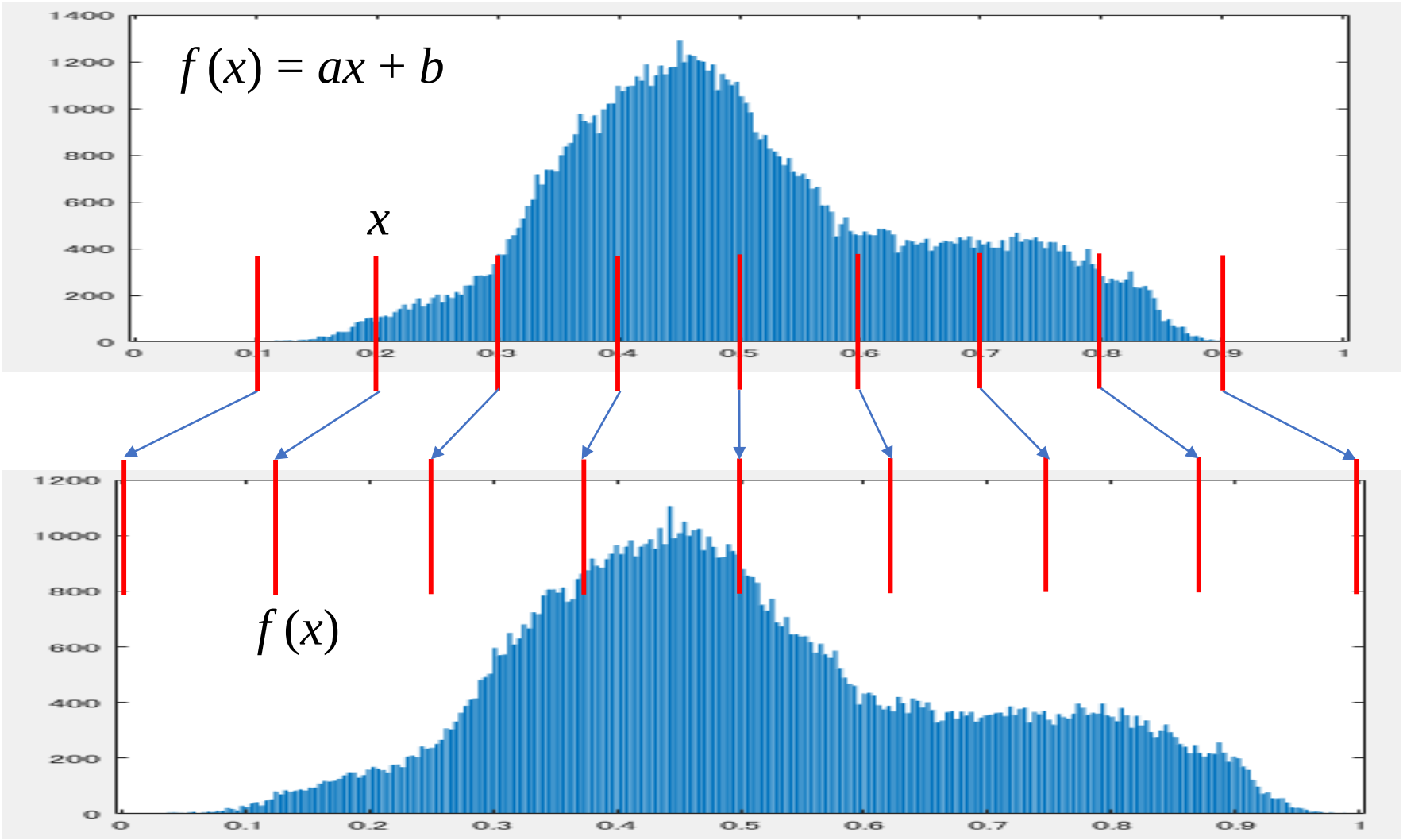
- 入力画像の輝度値　0.1 を
  出力画像では輝度値 0 へと変える.

- 入力画像の輝度値 0.9 を
  出力画像では輝度値 1 へと変える.

- そのような関数で表現できないか？
- 例：
  $x_1 = 0.1$ と $x_2 = 0.9$ および
  $y_1 = 0$ と $y_2 = 1$ が与えられたとき,
  $y = f(x) = a\,x + b$ となる関数.



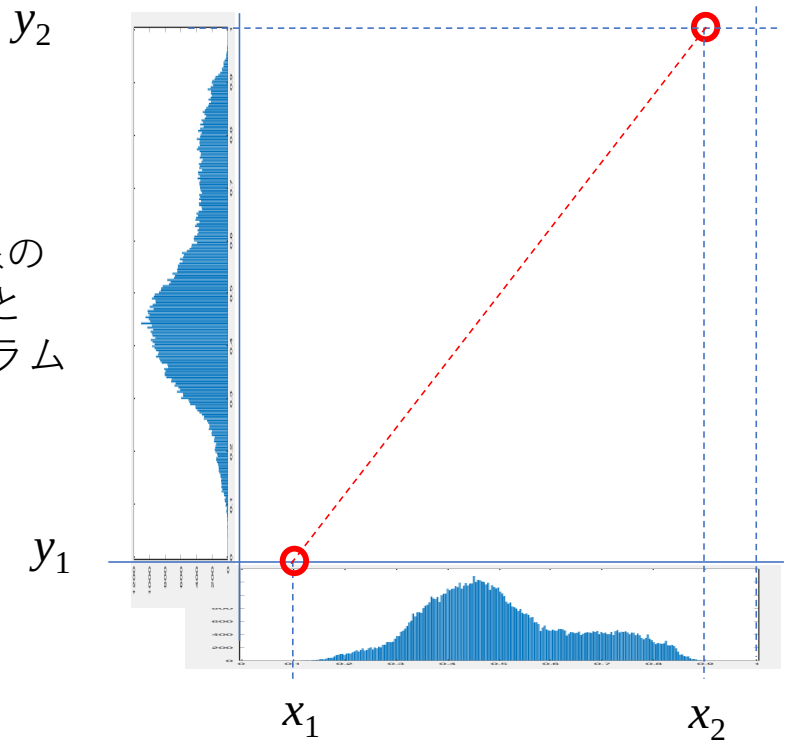
入力・出力輝度値のマッピング関数（２／４）



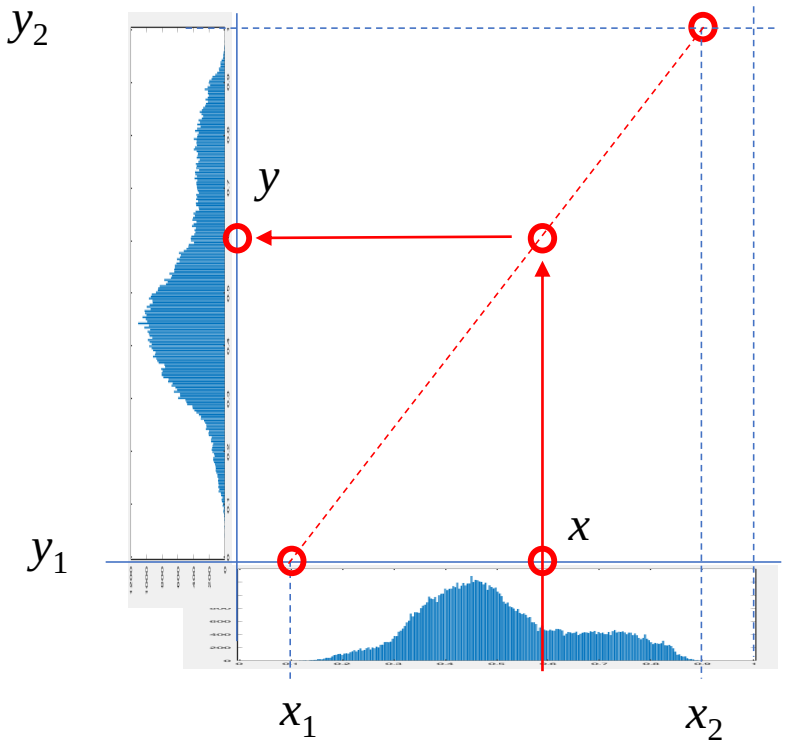
入力・出力輝度値のマッピング関数（３／４）

- $x_1 = 0.1$ と $x_2 = 0.9$ および $y_1 = 0$ と $y_2 = 1$ が与えられたとき，
  $y = f(x) = a\,x + b$ となる関数.
- $f(x) = ax$ を線形変換と呼び，$f(x) = ax + b$ はアフィン変換と呼ぶが，
  両者を基礎的な方法として線形濃度変換と呼ぶ.

出力画像の
輝度値と
ヒストグラム



入力画像の輝度値と
ヒストグラム



中間値もこの関数に従って写像
（マッピング）

入力・出力輝度値のマッピング関数（４／４）

- a と b の求解

  $\begin{matrix} y_1 = ax_1 + b \\ y_2 = ax_2 + b \end{matrix}$ を線形代数で表し $\begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} x_1 & 1 \\ x_2 & 1 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix}$,

  逆行列を両辺にかけ $\frac{1}{x_1 \cdot 1 - 1 \cdot x_2} \begin{pmatrix} 1 & -1 \\ -x_2 & x_1 \end{pmatrix} \begin{pmatrix} y_1 \\ y_2 \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix}$,

  式を整理して $\frac{1}{x_1 - x_2} \begin{pmatrix} y_1 - y_2 \\ -x_2 y_1 + x_1 y_2 \end{pmatrix} = \begin{pmatrix} a \\ b \end{pmatrix}$,

  $x_1 = 0.1$, $x_2 = 0.9$, $y_1 = 0$, $y_2 = 1$ を代入して

$$\begin{pmatrix} a \\ b \end{pmatrix} = \frac{1}{0.1 - 0.9} \begin{pmatrix} 0 - 1 \\ -0.9 \cdot 0 + 0.1 \cdot 1 \end{pmatrix} = \frac{1}{-0.8} \begin{pmatrix} -1 \\ 0.1 \end{pmatrix} = \begin{pmatrix} 5/4 \\ -1/8 \end{pmatrix}$$

- 別解

  $y_1 = 0$ と $y_2 = 1$ の場合に限り， $f(x) = (x - x_1) / (x_2 - x_1)$（オフセットを減算後スケールを正規化）

  $f(x) = ax + b$ と表す場合は， $a = 1 / (x_2 - x_1)$， $b = -x_1 / (x_2 - x_1)$

線形濃度変換でのマッピング関数

```
x1 = 0.1,   x2 = 0.9;
y1 = 0.0,   y2 = 1.0;

T = [x1, 1; x2, 1];
Para = inv(T)*[y1; y2];
a = Para(1);  b = Para(2);

Y_adj = a * Y + b; % 線形濃度変換
Y_adj = max(0,min(1, Y_adj )); % [0,1] の範囲外の値を丸め込み

H_adj = hist( Y_adj (:), bin_pos );

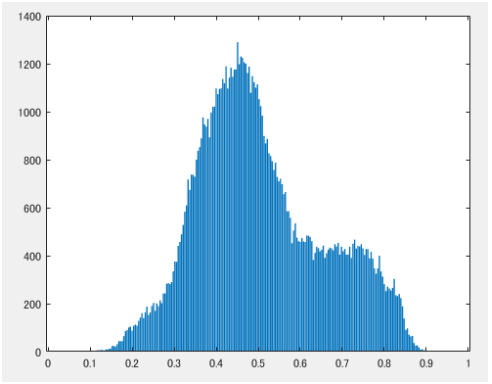
figure(4),  imshow( Y_adj );
figure(5),  bar( bin_pos, H_adj );
```

出力画像で，0 以下，1 以上の値が生じる場合，0以下の値は全て黒色，1以上の値は全て白色として扱うため，0 と 1 に丸め込む

実行結果

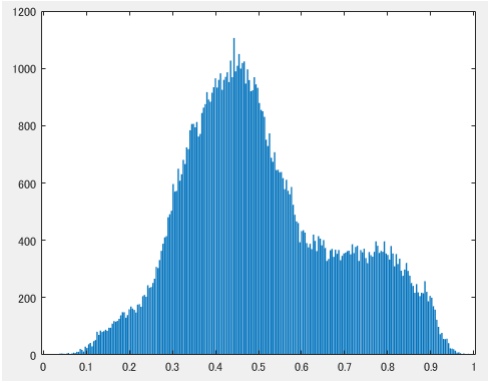
- 元々が良好な画像であるため，変化は感じにくいかもしれない

入力画像
（YCbCr の Y 成分）



入力画像のヒストグラム

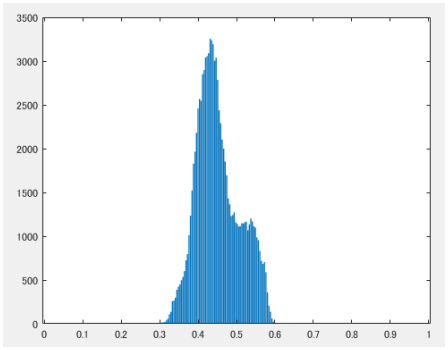
出力画像



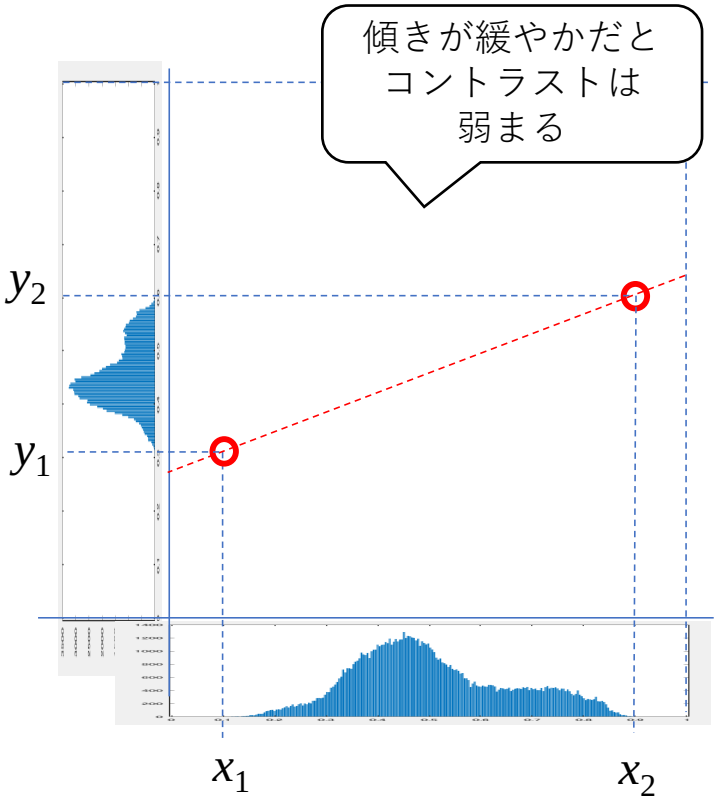
出力画像のヒストグラム

出力側の輝度値の範囲を狭めた場合

- プログラムを以下のように変更すると
- `x1 = 0.1,  x2 = 0.9;`
  `y1 = 0.3,  y2 = 0.6;`



出力画像およびヒストグラム



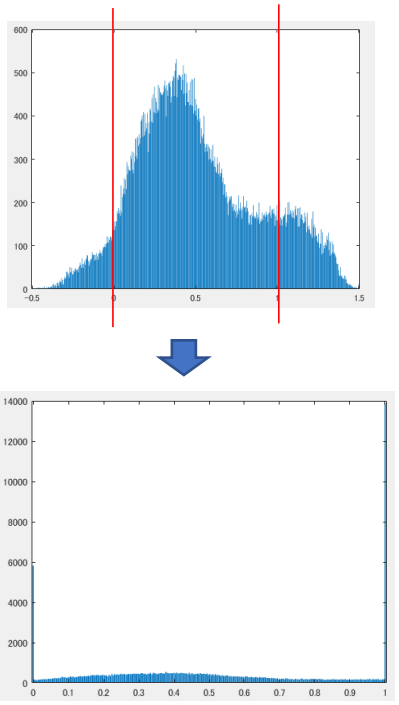
入出力関係

出側の輝度値の範囲を広げた場合

- プログラムを以下のように変更すると
- `x1 = 0.3,  x2 = 0.7;`
  `y1 = 0,  y2 = 1;`

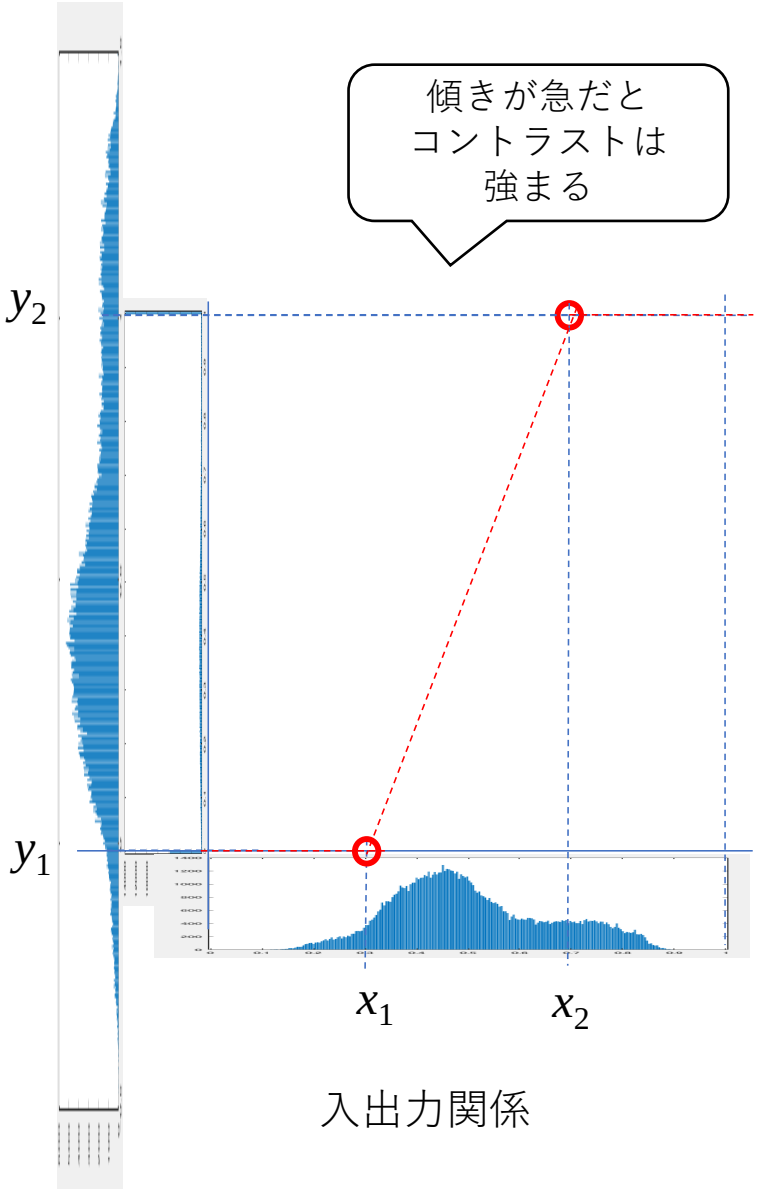
前述のように，出力画像では，0 以下，1以上の値が生じるため，本来のヒストグラムは右図のようになる．

出力画像およびヒストグラム



プログラムでは，0 以下，1以上の値を 0 と 1 に丸め込むため，右図のようなヒストグラムが得られる．

傾きが急だと
コントラストは
強まる



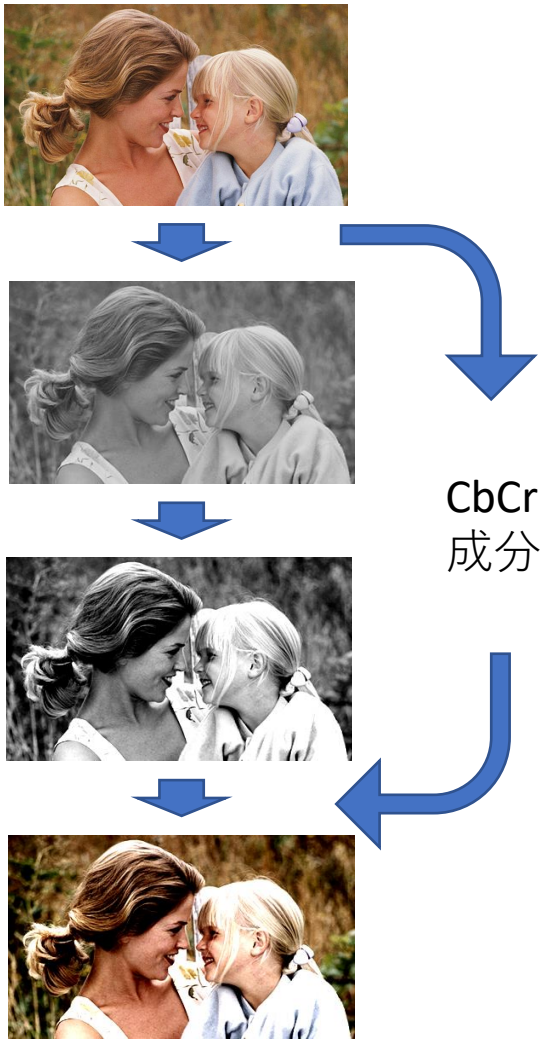
入出力関係

輝度の補正値をカラー画像に反映

- 入力された YCbCr 成分を出力用にコピーし，補正後の輝度成分 Y をコピーした後，RGB 成分へと変換する．
  - ```
    Jycc = Iycc;
    Jycc(:,:,1) = Y_adj;

    J = ycbcr2rgb( Jycc );

    figure(6), imshow( J );
    ```


    ```

YCbCr 変換を介さないとどうなるか？

- 彩度が変化する.
- R, G, B の各成分に対して,
  変換を行う場合
  - ```
    J = a * I + b;
    J = max(0,min(1,J));
    figure(6), imshow( J );
    ```
- YCbCr の各成分に対して
  変換を行う場合
  - ```
    Jycc = a * Iycc + b;
    Jycc = max(0,min(1, Jycc));
    J = ycbcr2rgb( Jycc );
    figure(6), imshow( J );
    ```
  - 結果は R, G, B とほぼ同じ.
    アフィン変換を含むため，完全に同一ではない.

Y 成分のみ処理

R, G, B を処理

Y, Cb, Cr を処理

補足：トーンマップ関数を別のファイルにまとめる方法（１／３）

- 関数ファイルの作成
  - 関数名 + 拡張子 m のファイルを作成する.
    例えば，tone_map_01.m というファイルを作成する.

  - tone_map_01.m ファイルに以下のように記述する.
  - function Y = tone_map_01(X, a, b)

```
Y = a * X + b;
Y = max(0,min(1,Y)); % 値域を [0,1] に修正

end % 関数の終了．書かなくても良い.
```

補足：トーンマップ関数を別のファイルにまとめる方法（２／３）

- 関数ファイルに書いた関数の呼び出し
  - スクリプトファイルと関数ファイルが同じフォルダにあるか，
    もしくは，関数ファイルがパスの通ったフォルダにあれば，
    `imshow` などの関数同様に呼び出せる．

  - スクリプトファイルでは，
    線形濃度変換の計算式の代わりに，関数を記述．

```
% 計算式はコメントアウト
% Y_adj = a * Y + b; % 線形濃度変換
% Y_adj = max(0,min(1,Y_adj));

Y_adj = tone_map_01( Y, a, b ); % 自作関数
```

補足：トーンマップ関数を別のファイルにまとめる方法（３／３）

- **x1, x2, y1, y2** を入力とする場合は,
  関数ファイルを以下のように書き,
  - `function Y = tone_map_01( X, range_src, range_dist )`

```
x1 = range_src(1),   x2 = range_src(2);
y1 = range_dist(1),  y2 = range_dist(2);

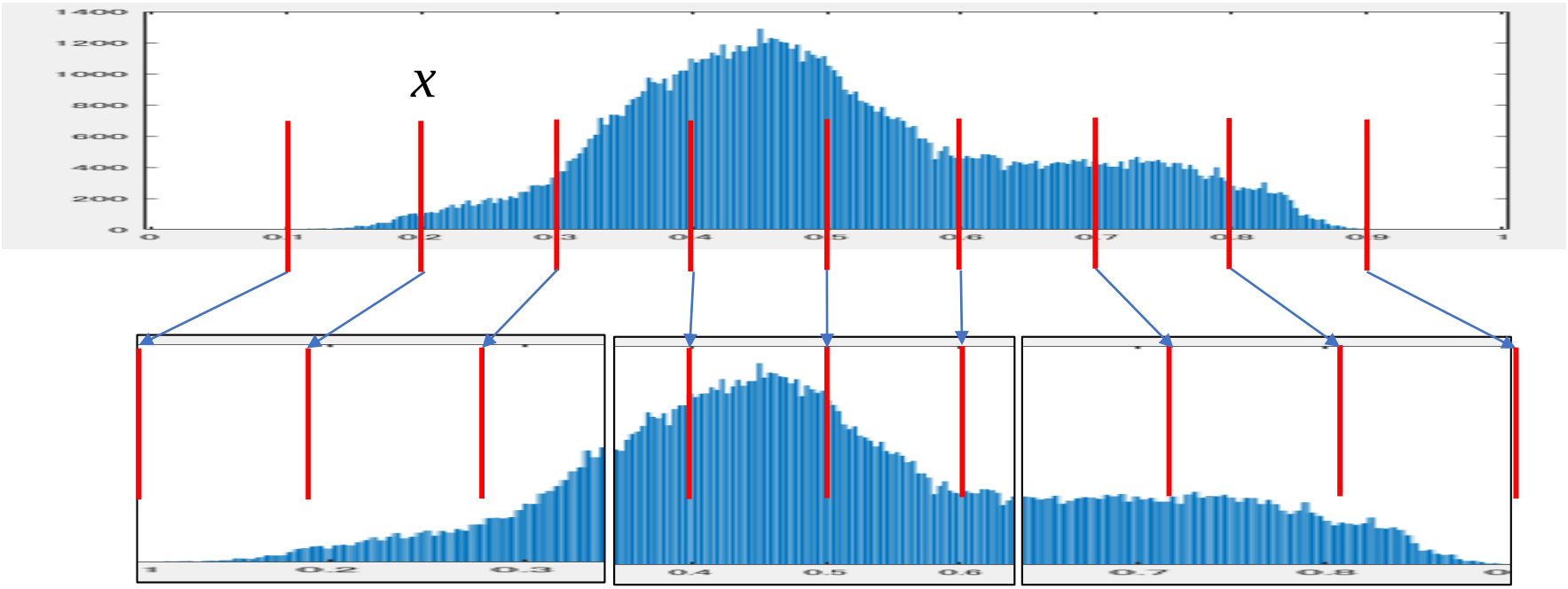
T = [x1, 1; x2, 1];
Para = inv(T)*[y1; y2];
a = Para(1);  b = Para(2);

Y = a * X + b;
Y = max(0,min(1, Y ));
end
```

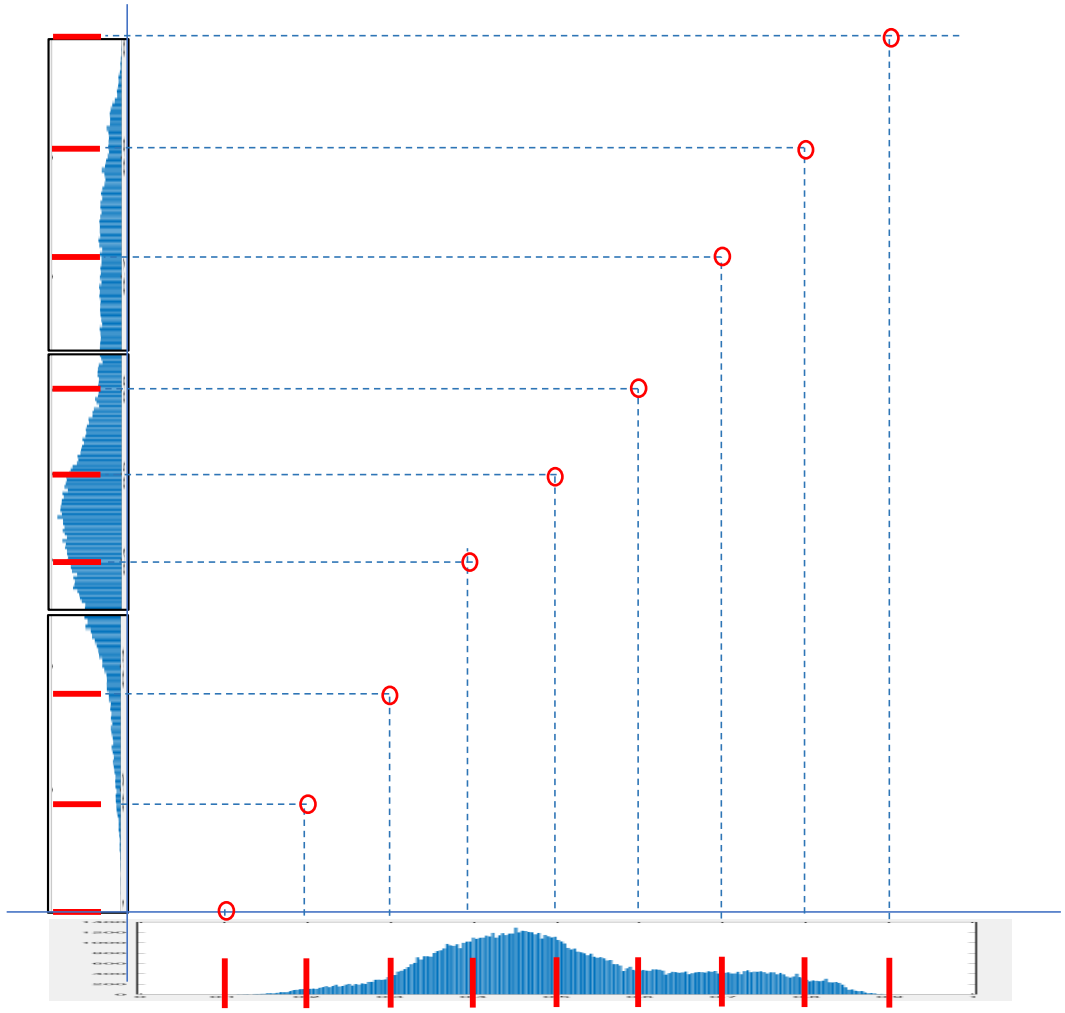
- スクリプトファイルを以下のように書くと良い
  - `Y_adj = tone_map_01( Y, [0.1,0.9], [0,1] );`

他のマッピング関数はないか？

- 部分ごとに広げ方を変えたい
  - 例：中央はそのままの輝度を概ね保持
    両端の輝度値を広げる
  - イメージ図



入出力関係から関数の形を想像する（１／３）



出力
輝度値

入力輝度値

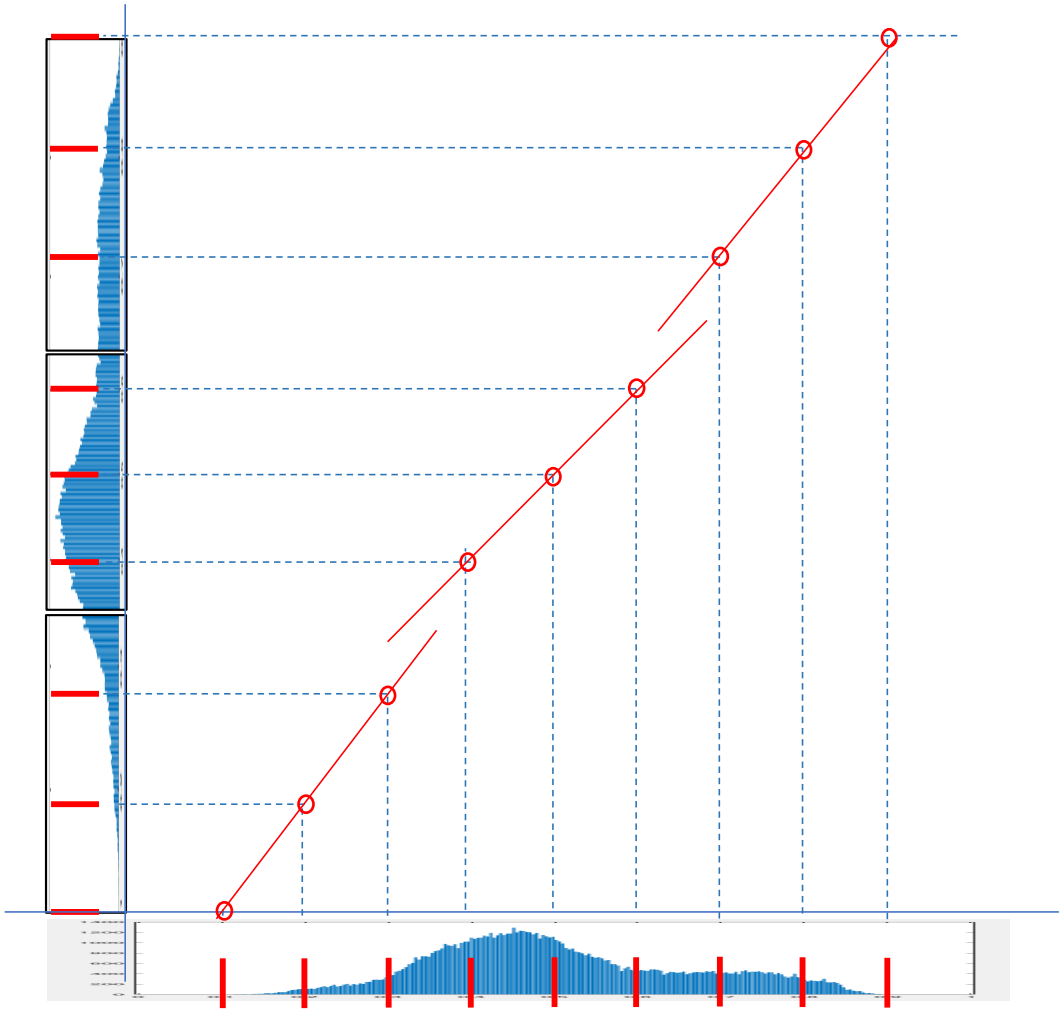
基準となる赤い印から
線を伸ばして・・・

交点をプロットしてみる
と・・・

関数の概形が見えてくる

入出力関係から関数の形を想像する（２／３）

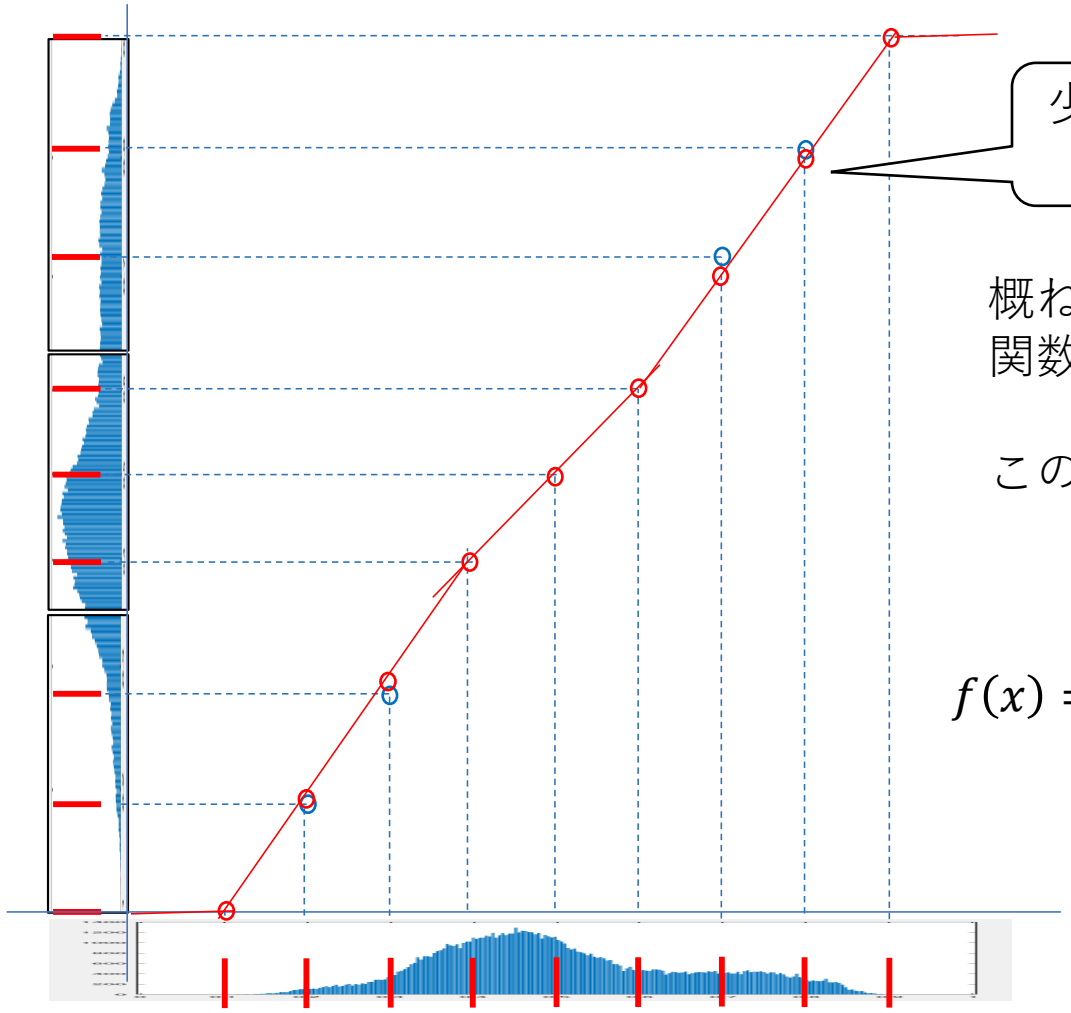
出力
輝度値



出力輝度値の見本を
適当に考えたため，
線が重なっていない
が・・・

入力輝度値

入出力関係から関数の形を想像する（３／３）



出力
輝度値

少し位置を
修正

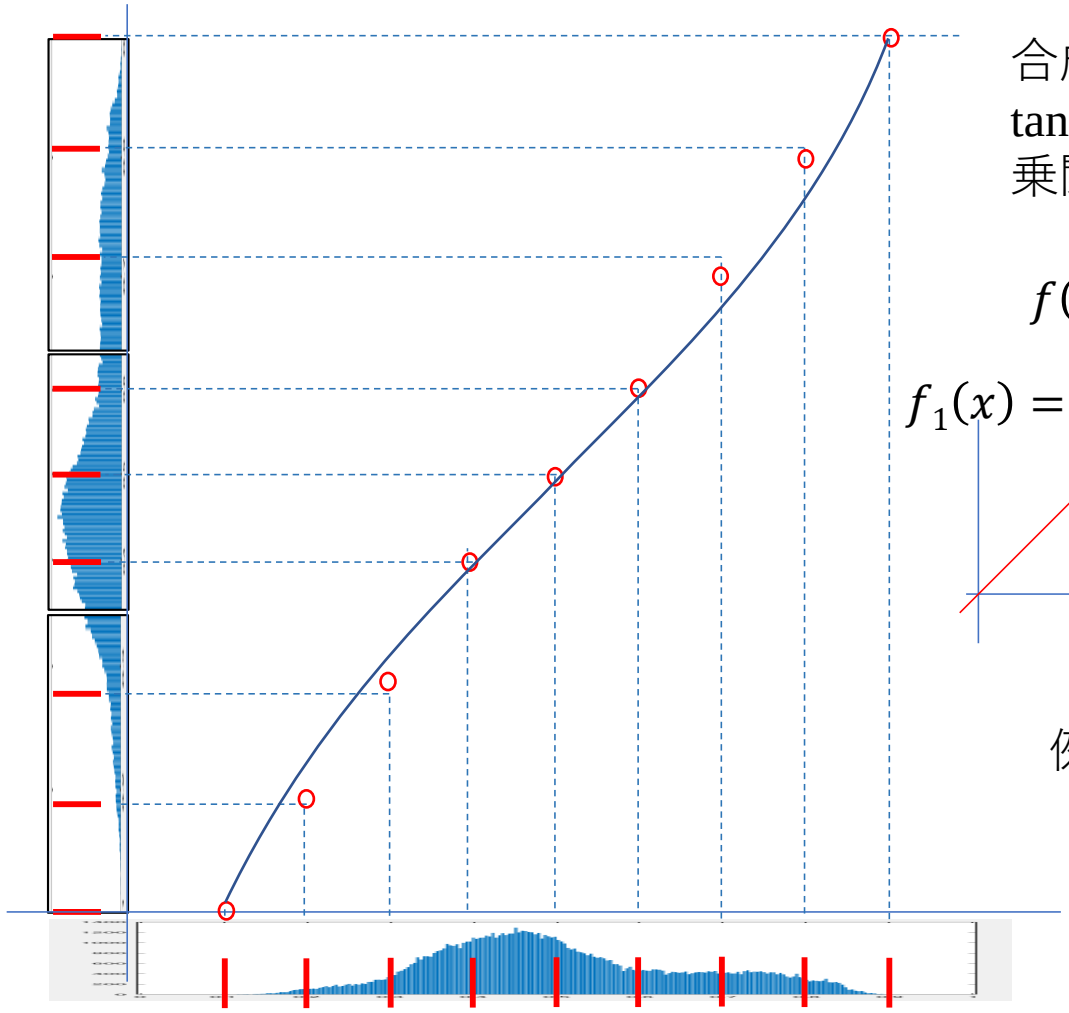
概ね，このような
関数になるのでは？

この区分関数の数式は

$$f(x)=\begin{cases}\frac{4}{3}x-\frac{0.6}{3} & x>0.6\\ x & x\in[0.4,0.6]\\ \frac{4}{3}x-\frac{0.4}{3} & x<0.4\end{cases}$$

入力輝度値

なめらかに形状が変化する関数で表す場合



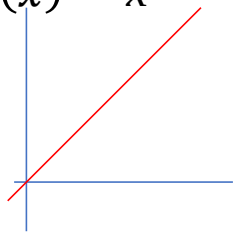
出力
輝度値

入力輝度値

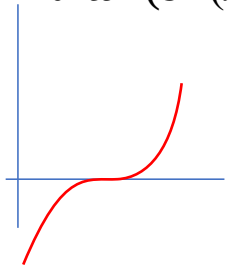
合成関数で設計する場合
tan 関数や $x^3 + x^5$ などの奇数乗関数の和で表現できる

$$f(x) = x + a\tan(b\pi(x-0.5))$$

$f_1(x) = x$



$f_2(x) = a\tan(b\pi(x-0.5))$



例えば，$a = 0.1,\ b = 0.9$

プログラムの関数で表すのであれば

- tone_map_02.m などを作成し，以下を記述
  - function Y = tone_map_02(X, a, b)

```
Y = X + a * tan( b * pi * (X - 0.5) );
Y = max(0,min(1,Y)); % 値域を [0,1] に修正

end % 関数の終了．書かなくても良い
```

- スクリプトファイルでは，以下のように呼び出し
  - Y_adj = tone_map_02(Y, 0.1, 0.9);

自作関数の形状を調べたい場合

- 0〜1 まで，0.01 ずつ増加するベクトルを作成し，
  トーンマップ関数に入力し，
  出力結果を plot コマンドでプロットする．
  - ```
    X_sample = 0:0.01:1;
    Y_sample = tone_map_02( X_sample, 0.1, 0.9 );

    figure, plot( X_sample, Y_sample );
    xlim( [0,1] ); % 描画範囲を [0,1] に収めたい場合
    ylim( [0,1] );
    ```
    ```

代表的なマッピング関数

- トーンマッピングに有用な関数
  - ガンマ補正関数
    - $f(x) = x^a \ (a \geqq 0), \qquad f(x) = x^{-a} \ (a \geqq 0)$
  - sin や tan を用いる関数
    - $f(x) = x + a \sin(x) \ (a \geqq 0), \qquad f(x) = x - a \sin(x) \ (a \geqq 0)$

- ハイダイナミックレンジ画像用
  - Reinhard らの関数
    - $f(x) = \frac{1}{1+x}\left(1 + \frac{x}{a^2}\right)$
  - その他
    - $1 - \exp(ax)$
    - $\log(1 + ax)$