

## CCM 原理

其实很简单，本质上就是调整饱和度。一个  $3 \times 3$  的矩阵，去乘以 RGB 组成的  $3 \times 1$  矩阵得到新的值；或者一个  $3 \times 4$  的矩阵，去乘以 RGB 再补上一个 1 的  $4 \times 1$  向量，得到了  $3 \times 1$  的新值。

通常用色卡来做：

1. 假设标准色卡的参考 RGB 为  $b$ ，其大小为  $3 \times 24$ ，每一列表示色卡的 RGB
2. 假设采集图片的色卡的 RGB 为  $x$ ，其大小为  $3 \times 24$ ，每一列表示色卡的 RGB。

通过求解  $Ax=b$  即可得到  $A$  这个  $3 \times 3$  矩阵了；那如果  $A$  是  $3 \times 4$  矩阵呢，那么  $x$  的大小是  $4 \times 24$ ，原来的  $x$  加上一列全 1，这样求解就是正确大小了。

为了便于讲述， $x$  选择  $3 \times 3$  的矩阵，其中的元素从左上角开始计数， $m_1$  到  $m_9$ 。可知：

$$\text{newR} = m_1 * R + m_2 * G + m_3 * B$$

$$\text{newG} = m_4 * R + m_5 * G + m_6 * B$$

$$\text{newB} = m_7 * R + m_8 * G + m_9 * B$$

有一个问题是会破坏白平衡，以白色块举例子， $R=G=B$ ，那么  $\text{newR} = R * (m_1 + m_2 + m_3)$ 、 $\text{newG} = G * (m_4 + m_5 + m_6)$ 、 $\text{newB} = B * (m_7 + m_8 + m_9)$ ，如果系数不进行约束，极大可能白色块  $\text{newR}$ 、 $\text{newG}$ 、 $\text{newB}$  不相等。

所以通常会有约束： $m_1 + m_2 + m_3 = m_4 + m_5 + m_6 = m_7 + m_8 + m_9 = \alpha$ ，其中  $\alpha$  是一个常数，通常是 1，当然你也可以取别的值。不过还是 1 最好，还是拿白色块思考，是 1 的话， $\text{newR} = R$ ，不会改变原来的值太多。

有约束了就不能直接在  $Ax=b$  用逆矩阵求解了。那就是一个最小化问题， $A$  相当于求下面矩阵。

$$\text{CCM} = \begin{bmatrix} 1 - x_1 - x_2 & x_1 & x_2 \\ x_3 & 1 - x_3 - x_4 & x_4 \\ x_5 & x_6 & 1 - x_5 - x_6 \end{bmatrix}$$

最小化就是最小化 **24 色卡的差距**，那么这个差距怎么算呢？每个预测色卡和参考色卡的 RGB 之差的平方和？nonono，有专门的颜色差异指标，又是一个很大的话题了，没必要细看。

可以去看 wiki 上的颜色差异，这里简单说一下，有两种指标：deltaE 和 deltaC，这两个都是需要将 RGB 转为 Lab 再进行比较。deltaE 比 deltaC 多考虑了 L 这个亮度指标，**不过本质上它们是一个东西（个人推测）**，你根据 deltaE 进行优化最后得到参数和 deltaC 优化是一样的，只不过最后的指标值会有不同罢了。比如相当于一个用 sum(x)，一个用 sum(x+1)。

deltaE 和 deltaC 又有很多变种，从一开始的直接平方差之和，到后来各种条件，没必要细看，知道这玩意就行了，比如这个图，一个是 76 年的，一个是 00 年，后面太复杂了额：

应用 $(L_1^*, a_1^*, b_1^*)$ 和 $(L_2^*, a_2^*, b_2^*)$ 两个 $L^*a^*b^*$ 色彩空间的颜色：

$$\Delta E_{ab}^* = \sqrt{(L_2^* - L_1^*)^2 + (a_2^* - a_1^*)^2 + (b_2^* - b_1^*)^2}$$

本公式对应的**最小可觉差**： $\Delta E_{ab}^* \approx 2.3$  [6]

#### CIEDE2000 [编辑]

鉴于1994年的公式并没有充分解决**感知非均匀特性**的问题，CIE再次修缮了定义，并加入了5个修订系数：[12][13]

- 色调旋转项( $R_T$ )，用来应对常出问题的蓝色区域（色相角度275°左右）；[14]
- 中性色调补偿（对应 $L^*C^*h$ 差异）
- 亮度补偿（ $S_L$ ）
- 色度补偿（ $S_C$ ）
- 色调补偿（ $S_H$ ）

$$\Delta E_{00}^* = \sqrt{\left(\frac{\Delta L'}{k_L S_L}\right)^2 + \left(\frac{\Delta C'}{k_C S_C}\right)^2 + \left(\frac{\Delta H'}{k_H S_H}\right)^2 + R_T \frac{\Delta C'}{k_C S_C} \frac{\Delta H'}{k_H S_H}}$$

注：下面的公式请使用角度而非弧度，尤其在 $R_T$ 上需要注意。

$k_L, k_C, k_H$ 一般取1。

$$\Delta L' = L_2^* - L_1^*$$

$$\bar{L} = \frac{L_1^* + L_2^*}{2} \quad \bar{C} = \frac{C_1^* + C_2^*}{2}$$

$$a_1' = a_1^* + \frac{a_1^*}{2} \left(1 - \sqrt{\frac{\bar{C}^7}{\bar{C}^7 + 25^7}}\right) \quad a_2' = a_2^* + \frac{a_2^*}{2} \left(1 - \sqrt{\frac{\bar{C}^7}{\bar{C}^7 + 25^7}}\right)$$

$$\bar{C}' = \frac{C_1' + C_2'}{2} \text{ and } \Delta C' = C_2' - C_1' \quad \text{where } C_1' = \sqrt{a_1'^2 + b_1'^2} \quad C_2' = \sqrt{a_2'^2 + b_2'^2}$$

$$h_1' = \text{atan2}(b_1', a_1') \mod 360^\circ, \quad h_2' = \text{atan2}(b_2', a_2') \mod 360^\circ$$

**Note:** The inverse tangent ( $\tan^{-1}$ ) can be computed using a common library routine `atan2(b, a)` which usually has a range from  $-\pi$  to  $\pi$  radians; color specifications are given in 0 to 360 degrees, so some adjustment is needed. The inverse tangent is indeterminate if both  $a'$  and  $b$  are zero (which also means that the corresponding  $C'$  is zero); in that case, set the hue angle to zero. See Sharma 2005, eqn. 7.

$$\Delta h' = \begin{cases} h_2' - h_1' & |h_1' - h_2'| \leq 180^\circ \\ h_2' - h_1' + 360^\circ & |h_1' - h_2'| > 180^\circ, h_2' \leq h_1' \\ h_2' - h_1' - 360^\circ & |h_1' - h_2'| > 180^\circ, h_2' > h_1' \end{cases}$$

**Note:** When either  $C_1'$  or  $C_2'$  is zero, then  $\Delta h'$  is irrelevant and may be set to zero. See Sharma 2005, eqn. 10.

$$\Delta H' = 2\sqrt{C_1' C_2'} \sin(\Delta h'/2), \quad \bar{H}' = \begin{cases} (h_1' + h_2' + 360^\circ)/2 & |h_1' - h_2'| > 180^\circ \\ (h_1' + h_2')/2 & |h_1' - h_2'| \leq 180^\circ \end{cases}$$

**Note:** When either  $C_1'$  or  $C_2'$  is zero, then  $\bar{H}'$  is  $h_1' + h_2'$  (no divide by 2; essentially, if one angle is indeterminate, then use the other angle as the average; relies on indeterminate angle being set to zero). See Sharma 2005, eqn. 7 and p. 23 stating most implementations on the internet at the time had "an error in the computation of average hue".

$$T = 1 - 0.17 \cos(\bar{H}' - 30^\circ) + 0.24 \cos(2\bar{H}') + 0.32 \cos(3\bar{H}' + 6^\circ) - 0.20 \cos(4\bar{H}' - 63^\circ)$$

$$S_L = 1 + \frac{0.015(\bar{L} - 50)^2}{\sqrt{20 + (\bar{L} - 50)^2}} \quad S_C = 1 + 0.045\bar{C}' \quad S_H = 1 + 0.015\bar{C}' T$$

$$R_T = -2\sqrt{\frac{\bar{C}'^7}{\bar{C}'^7 + 25^7}} \sin \left[ 60^\circ \cdot \exp \left( - \left[ \frac{\bar{H}' - 275^\circ}{25^\circ} \right]^2 \right) \right]$$

最后的最后，说一下代码实现，用的陈伟师兄写的代码，写的很好，最终实现的代码在 `color_matrix.py` 中。本质上用的 `scipy` 的 `minimize` 来做，他的好处是不用在计算式中将  $1-m_2-m_3$  显著地替换  $m_1$ ，而是在参数 `constraints` 中添加  $m_1+m_2+m_3=1$ ，这样看起来很直观。

不适合入门，但是回头看感觉很不错：<https://zhuanlan.zhihu.com/p/413851281>.