19127517

Hồ Thiên Phước

1.1

```
while k < max freq :
    while n<N:  # means runs until n = N-1
        F = (tdom_flt[n]) *((cmath.exp((1j*-2*(math.pi)*
                                n*k)/N)))
        fdom_flt[k] = fdom_flt[k] + F
        n = n+1
    k = k+1
    n = 0  # reset npos to 0
           # npos > len(tdom_flt)
```

1.2 Miền không gian:

$$f(x,y) * h(x,y) = \frac{1}{MN} \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} f(x-m, y-n) . h(m,n)$$

Độ phức tạp $O(n^2 * m^2)$

Miền tần số:

Bước 1: $\mathcal{J}(f) = F(u,v)$: $O(n^2 \log_2 n)$: FFT

Bước 2: $\mathcal{J}(h) = H(u,v)$: $O(n^2)$

Bước 3: $F(u,v) . H(u,v)$: $O(n^2)$

Bước 4: $\mathcal{J}^{-1}\{F \cdot H\}$: $O(n^2 \log_2 n)$

Độ phức tạp: $O(n^2 \log_2 n)$

Kết luận: Phương pháp nào nhanh hơn sẽ phụ thuộc vào giá trị m, n.

3.1 Function K-means()

- Chia lưới ảnh
- Mỗi ô lưới là 1 vector đặc trưng
- Khởi tạo vector đặc trưng khởi thuỷ bằng cách lấy ngẫu nhiên vector đặc trưng của vùng từ ảnh đã bị chia ô lưới rồi tính trung bình cộng giá trị độ xám

$w_j = i_L, j \in \{1, \dots, k\}, L \in \{1, \dots, n\}$

Mỗi vùng đã phân đoạn liên kết với w_j

Loop

Kết nạp ứng viên vào vùng bằng cách tính độ dị biệt của vector đặc trưng đang xét với mỗi vùng. (với điều kiện cluster (vùng) mình chọn để kết nạp phải làm cho đối số dị biệt của vector đặc trưng đang xét với vector đặc trưng khởi thuỷ là cực tiểu.)

Điều kiện cần của cực tiểu

$$\frac{\partial E(t)}{\partial t} = 0 \approx \frac{E(t) - E(t-1)}{t - (t-1)} = 0 \Rightarrow |E(t) - E(t-1)| < \varepsilon$$

```
For l in range (1, n):
    minDistance = distance (i_L, w_i)
    minIdx = 1
        For j* in range (1, K):
            If (distance (i_L, w_i) < minDistance):
                minDistance = distance (i_L, w_i)
                minIdx = j*
    cluster[minIdx].add (i_L)
```

For mỗi vùng, do

Cập nhật lại vector riêng cho tất cả các vùng với công thức trung bình cộng

prototype khởi thuỷ $w_j = \sum_{\text{vector đặc trưng } i_L \in \text{từng vùng } C_j} i_L / |C_j|$;

Tính tổng sai số

$$E = \sum_{j=1}^{k} \sum_{i_L \in C_j} |i_L - w_j|^2$$

3.2

elb

Đkiện dừng của loop khi các vùng đã ổn định

```
stop_condition:
    bool flag = true;
    for (int j = 1 ; j <= k ; j++) {
        If (w_j^k != w_j^(k-1)) flag = false;
        Else continue;
    }.
    If(flag == true) break;
    Else { update_prototype();
        goto stop_condition;
    }
```

3.2

```
from sklearn.cluster import KMeans.
from matplotlib import pylot as plt.
import numpy as np
from sklearn.decomposition import PCA
```

elbow method

```
dataRaw = []
DataFile open
    while True:
        theline = DataFile.readline()
        if len(theline == 0): break
        readdata = theline.split(" ") # space 2 columes
        for pos in range(len(readdata)):
            readdata[pos] = float(readdata[pos])
        dataRaw.append(readdata)
DataFile close
X = np.array(dataRaw)
distorsions = []
  for k in range(2, 20):
```

PHI TRÍ BẤT HƯNG

```
    kmeans = KMeans (n_clusters=k)
    kmeans.fit (X)
    distorsions.append (kmeans.inertia_)
fig = plt.figure (figsize = (15,5))
plt.plot (range(2,20), distorsions)
plt.grid (True)
```

2.1 $\underset{n\times 1}{y} = \underset{n\times n}{A^T}\underset{n\times 1}{x}$, $A^{-1} = A^T$

$$x = (A^T)^{-1} y = (A^{-1})^{-1} y = Ay = \begin{bmatrix} a_0(0) & a_1(0) & \cdots & a_{n-1}(0) \\ a_0(1) & a_1(1) & \cdots & a_{n-1}(1) \\ \vdots & \vdots & \ddots & \vdots \\ a_0(n-1) & a(n-1) & \cdots & a_{n-1}(n-1) \end{bmatrix} \cdot \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \end{bmatrix}$$

$$= y_0 \begin{bmatrix} a_0(0) \\ a_0(1) \\ \vdots \\ a_0(n-1) \end{bmatrix} + y_1 \begin{bmatrix} a_1(0) \\ a_1(1) \\ \vdots \\ a_1(n-1) \end{bmatrix} + \cdots + y_{n-1}^{m} \begin{bmatrix} a_{n-1}(0) \\ a_{n-1}(1) \\ \vdots \\ a_{n-1}(n-1) \end{bmatrix}$$

$$x = \sum_{i=1}^{n} y_i a_i$$

$$(x.a_j = (\sum_{i=1}^{n} y_i a_i) a_j = \sum_{i=1}^{n} y_i a_i a_j = y_j$$

Xấp xỉ $x \approx \hat{x} = \sum_{i=1}^{m} y_i a_i$, $(m < n)$

Sai số $Ss = \|x - \hat{x}\|^2 = \|\sum_{i=m}^{n-1} y_j a_i\|^2 = \sum_{i=m}^{n-1} y_i^2$

$$Ss = E(\|x - \hat{x}\|^2) = E[\sum_{i=m}^{n-1} y_i^2] = E[\sum_{i=m}^{n-1} (xa_i)^T xa_i$$

$$= \sum_{i=m}^{n-1} a_i^T E(xx^T) a_i + \begin{cases} a_i^T a_i = 1 \\ a_j^T a_j = 0 \ , \ i \neq j \end{cases}$$

$$= \sum_{i=m}^{n-1} \lambda_i$$

 Đ Kiện biên

Để cực tiểu sai số ta sắp λ_i theo thứ tự giảm dần rồi bỏ $n-1+m+1$ vector riêng λ_m -

Phương pháp nhân tử Lagrange

$$F(a_i, \lambda_i) = \sum_{i=m}^{n-1} a_i^T E(x^T x) a_i + \sum_{i=m}^{n-1} \lambda_i (1 - a_i^T a_i)$$

Đkiện cần:

$$\frac{\partial F}{\partial a_i} = 0 \Rightarrow a_i^T E(x^T x) - a_i^T \lambda_i = 0$$

$$\frac{\partial F}{\partial \lambda_i} = 0 \Rightarrow E(x^T x) \cdot a_i = \lambda_i a_i$$

(a_i vector riêng ma trận tương quan $E(x^T x)$)

$a_{n-1}(0)$
$a_{n-1}(1)$
$\vdots$
$a_{n-1}(n-1)$

$$\cdot \begin{bmatrix} y_0 \\ y_1 \\ \vdots \\ y_{n-1} \end{bmatrix}$$

PHI TRÍ BẤT HƯNG

$-m+1$ vector riêng $\lambda_m \to \lambda_{n-1}$ ứng với $n-1+1-m$ giá trị riêng bé nhất. (giữ lại $\lambda_0 \to \lambda_{m-1}$)