Geometric Learning

048865

HomeWork 4

By: Moshe Kimhi
ID: 203374293

1 Graph Signal Processing

1.1 Image Denoising

Let $x_0 \in R^n$ an image and $y = x_0 + n$ a noisy image when $n \sim \mathcal{N}(0, 0.01)$

classical approach

we need to solve

$$\min_x \|y - x\|_2^2 + \gamma x^T L x$$

we simply derive in quadratic form notation , and show that the given function to minimize is convex, so the solution when the derivative equals zero is minimum:

$$\frac{d}{dx} \min_x \|y - x\|_2^2 + \gamma x^T L x = 0 \tag{1}$$
$$2x^* - 2y + \gamma(L^T + L)x^* = 0 \tag{2}$$
$$2(I + \gamma L)x^* = 2y \tag{3}$$
$$x^* = (I + \gamma L)^{-1} y \tag{4}$$

for (3) we used the fact that the laplacian is symmetric, and for (4) we used the fact that L is positive-semi-definite, so in addition to I is invertible. the second derivative is: $\frac{d}{dx^2} = 2(I + \gamma L)$, since L is positive-semi-definit, $\gamma > 0$, so $\frac{d}{dx^2}$ positive-semi-definite.

conjugate gradient

We used the conjugate gradient method to denoise a given image in 2 ways:

convolution operation matrix:

with help from the supplied video, i created the operational matrix first, by creating 3 matrices A,B,C that corispond to the output of each column of the given filter with the image, and then use Kronecker product of those A,B,C to generate convolution op matrix. note that due to the size, it's crusial to use sparse.csr matrix

Listing 1: core of the operational matrix

```
"""
    :param h: hight of image
    :param w: width of image
    :param f_num: -1/8
"""
    basic_form = np.tril(np.ones((h, h)), 1) - np.tril(np.ones((h,
        h)), -2)
    basic_form[0, h - 1] = 1
    basic_form[h - 1, 0] = 1
    A = f_num * basic_form
    B = sparse.csr_matrix(A + np.eye(h) * (1 + -1*f_num))
    C = sparse.csr_matrix(A)
    A = sparse.csr_matrix(A)

    A_kron = np.tril(np.ones((w,w)),1) - np.tril(np.ones((w,w)),0)
    A_kron[w-1,0]=1
    A_kron = sparse.csr_matrix(A_kron)
    B_kron = sparse.csr_matrix(np.eye(w))
    C_kron = np.tril(np.ones((w,w)),-1) - np.tril(np.ones((w,w))
        ,-2)
    C_kron[0,w-1]=1
    C_kron=sparse.csr_matrix(C_kron)
    return sparse.kron(A_kron,A)+sparse.kron(B_kron,B) + sparse.
        kron(C_kron,C)
```

Bilateral filtering

same method of conjugate gradients, this time i created walk matrix with the given weight,

Listing 2: generate W

```
    n = len(image_cs)
    W_g = sparse.lil_matrix((n, n))
    weight_fun = lambda a,b: np.exp((-1*np.abs(a-b)**2)/(2*sigma
        **2))
    for i in range(n):
        neigbors = [i+1,i+h-1,i+h,i+h+1]
        for j in neigbors:
            if j < n:
                W_g[i,j] =W_g[j,i] = weight_fun(image_cs[i],
                    image_cs[i+1])
    return W_g
```

and then $L_g = D_g - W_g$ when D_g is simply the degree matrix

below, we'll explain the results of both methods:

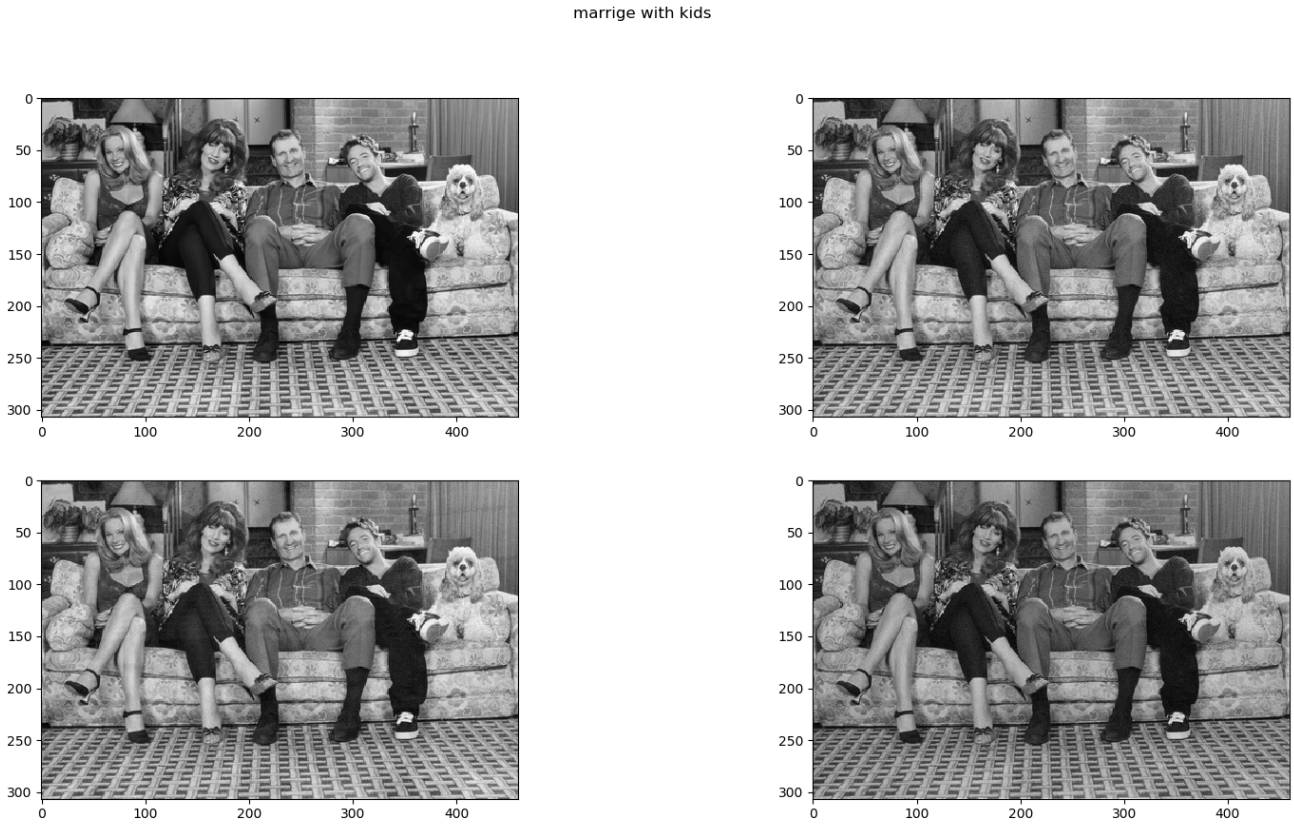

Figure 1: TOP left: original image. TOP right: noisy image.
Bottom left: denoised by conv op matrix. BOTTOM right: denoised by Bilateral filtering

note that added to this report, a higher resolution of this images. both denosied images improved a little the image while with the convolving kernel we see a bit of smearing towards the x direction (while when i tried to use row stack we can see the smear in the y direction).

the Bilateral filtering with right parametrization (sigma=0.001 for the given image) the reconstruction is very good. we can see the diffrance mostly on the blank surface such as Meg pants. note that with the low noise we can see good reconstruction with both. i used np.random.normal with scale twice as big as asked in the qustion.

1.2 3D Point Clouds Denoising

the Pointcloud i choose is chair0015

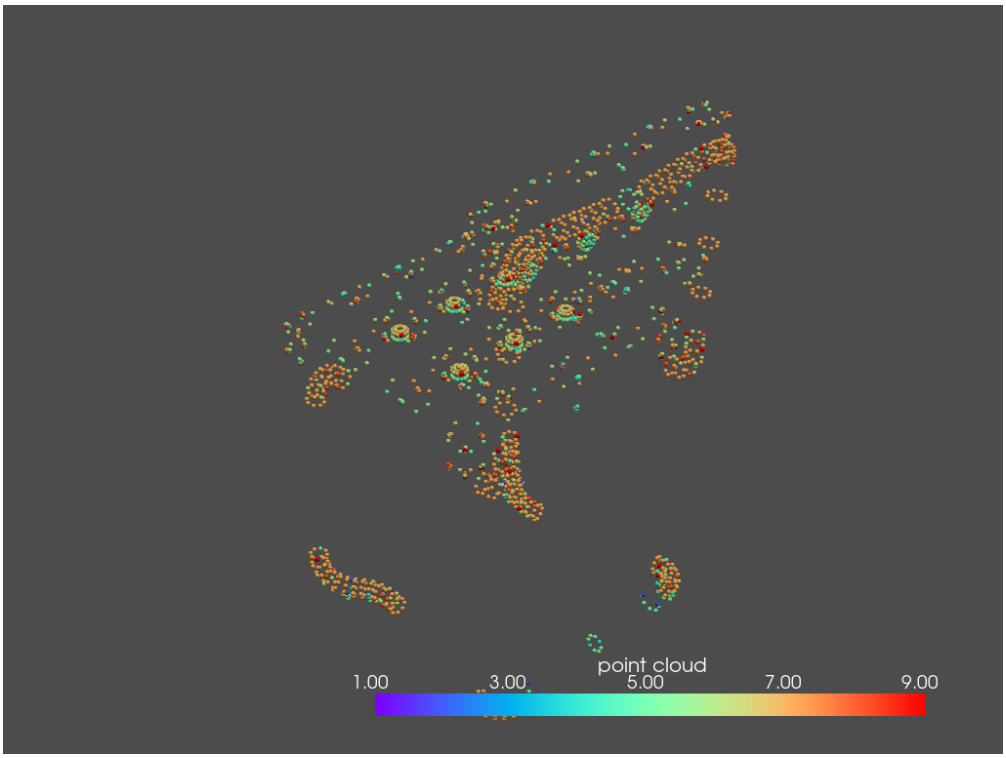

Figure 2: chair colored by vertex degree

After computed W (with help from scipy cdist), with r=25, i got the eigenvalues and vectors. first 10 eigen values:

8.495586581936169729e-05

7.083531418562660992e-04

1.040212360737288386e-02

1.779146072344694671e-02

6.241477880632263775e-02

9.326431529653538000e-02

2.681110970193160692e-01

3.241556080371286730e-01

3.327608664917803827e-01

4.024290444256259369e-01

all of the values in the eigen-valus file attached.

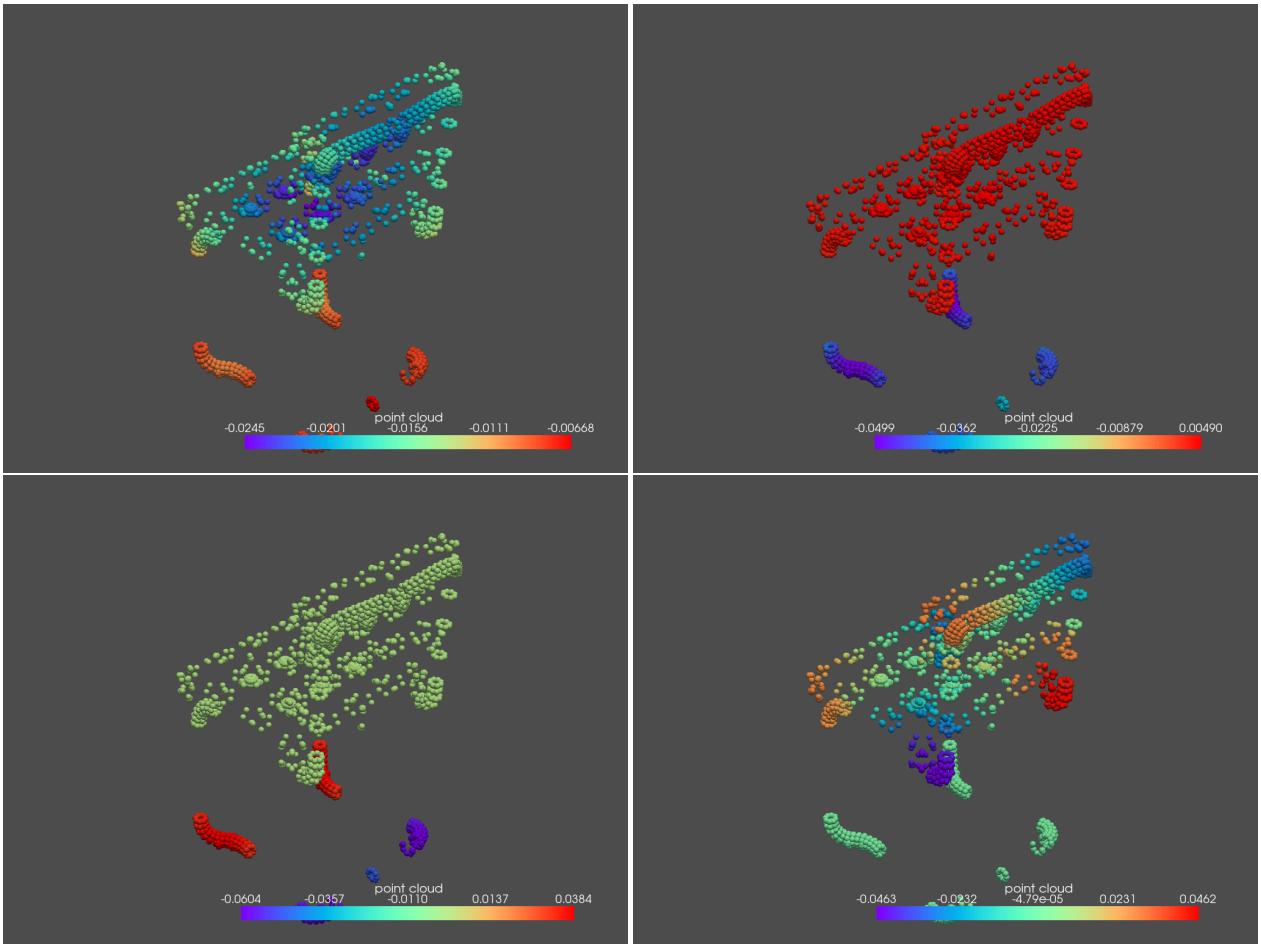

Figure 3: noisy chair PC colored by 1,2,4,10 eigen vectors

the top left is the first eigenvector (corispond to the smalles eigenvalue) top right is the second evector bottom left is the 4th and bottom right is the 10th.

Low Pass Filter

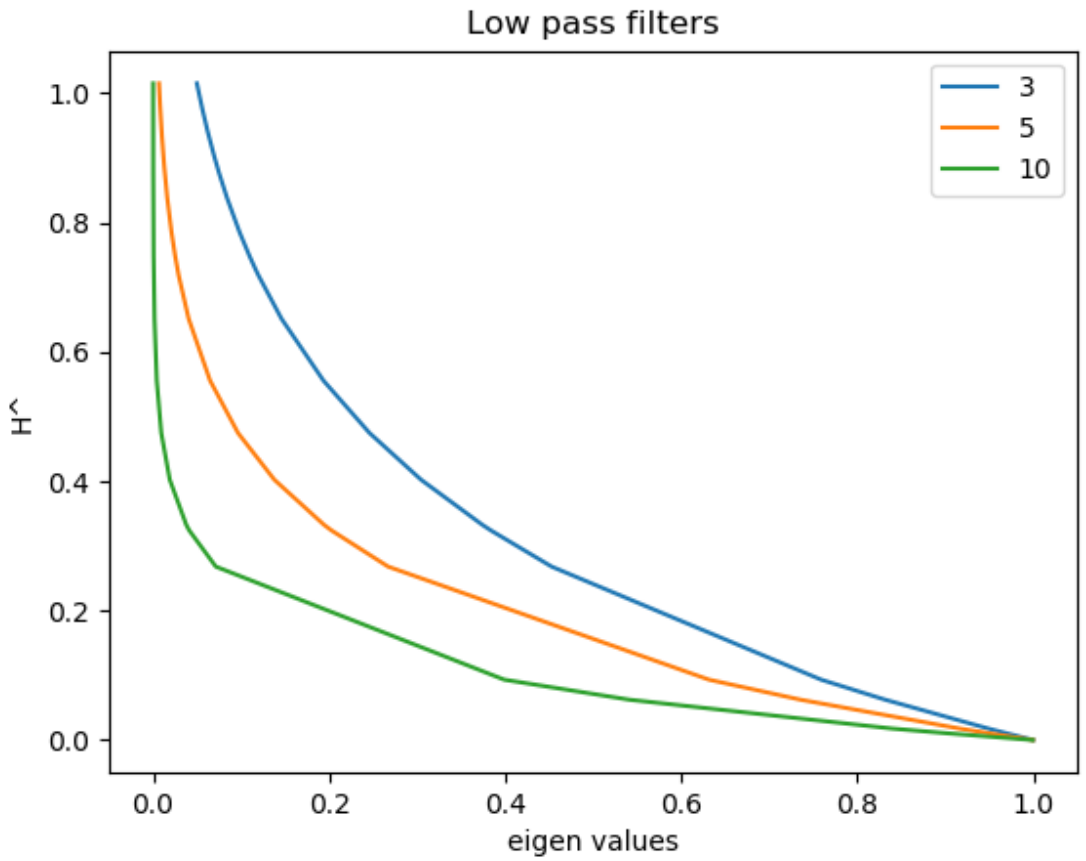

Figure 4: low pass as a function of tao

we can simply see that highter tao is a stronger curve of the LPF, with less frequncies to pass (tao is the exact same as Discharge of capacitor)

Now we use the reconstruction of the points by $\psi\eta_h\psi^T X$ where η_h is the diagonal matrix with h - the low pass filter as the diagonal

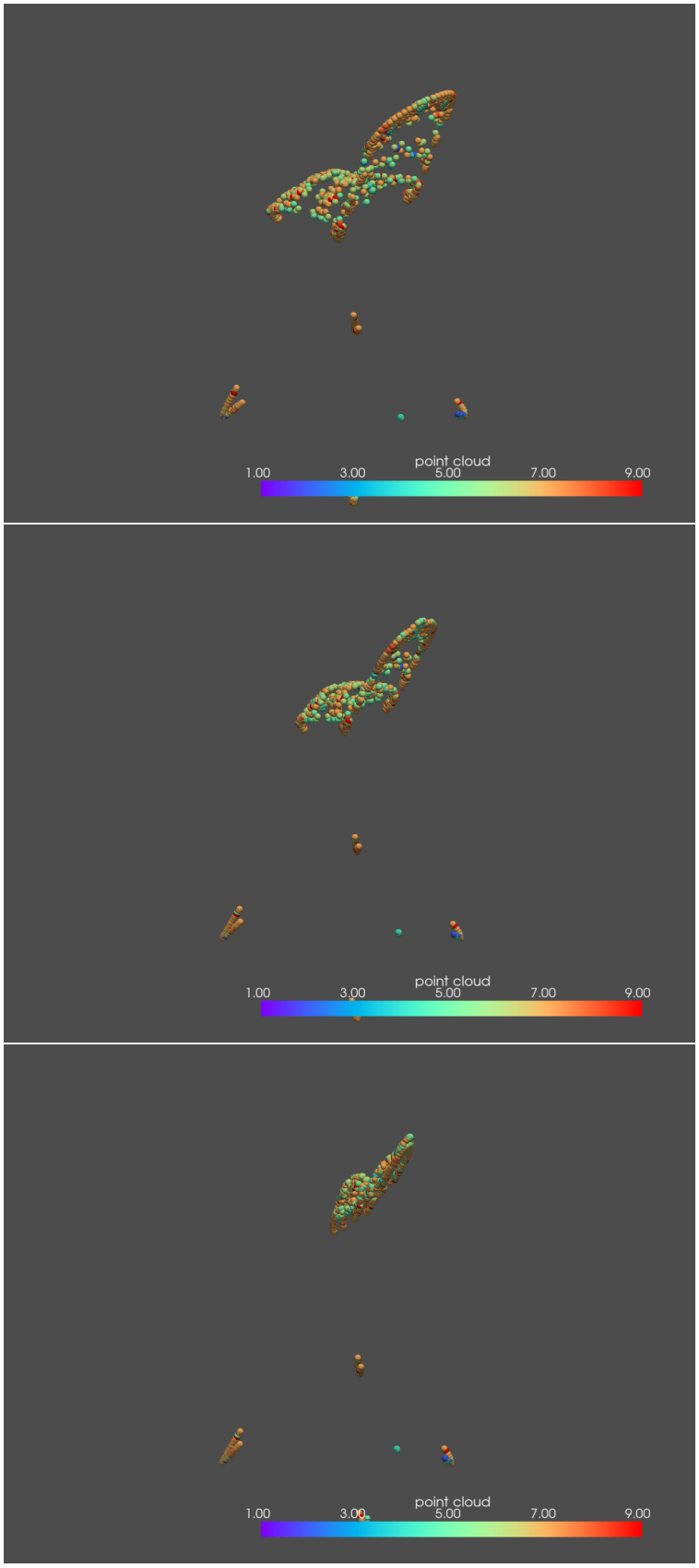

Figure 5: denoise by LPF with tao = 3,5,10

the reconstruction do make the surface of the chair, where the density of the points are high to look smooth and nice. with the Point cloud i choose, we see that some high frequencies are important, and as we take only low frequencies (such as in tao=10 or 5) we loose information about the original structure, as the chair looks more and more like a bar chair and not the original one. it's not a huge problem if we try to reconstruct a point cloud with low frequencies only. but noise can alias to point cloud information if we try to use higher frequencies as well. also, task dependent, if we just want to classify the image, even with tao=5 we can get a good idea of a chair in the image. for reconstruct face for example, as we learned in the last HW, we can not use this method at all since the face details depend on high frequencies.