Assignment 4:
Graph Signal Processing, Spectral Clustering, and SPD Riemannian Manifolds

TA in charge of the exercise: Ya-Wei Eileen Lin (lin.ya-wei@campus.technion.ac.il)
Submission date: January 26^{th} (26/01/2021)
Relevant course material: Lectures 1-13 & Recitations 1-13

1. **Homework scale:**
   This assignment holds three questions -
   (a) **Single submission** – you are expected to present **one** questions.
   (b) **Submitted in pairs** – you are expected to present **two** questions.
2. **Submission Format.** Please submit your solutions to the appropriate homework submission box on Moodle as a ***zip*** file – with the file name ***HW4_ID.zip*** for single submission and ***HW4_ID1_ID2.zip*** for pair submission (where IDs are you and your partner's ID number).
   Your solution zip should contain:
   - A single PDF file with the name ***HW4_ID_Solution.pdf*** for single work and ***HW4_ID1_ID2_Solution.pdf*** for pair submission.
     Your report should be **typed** and presented in **English**. No other presentation format will be accepted. **The report should conclude all the results and the figures you obtained in the wet parts.**
   - Source code implementation for you wet part. Please make sure the code runs directly after uncompressing the zip file. Code must be written **solely** in the Python language unless directly stated otherwise. You may use any Python module you like, but we expect to see mostly your own coding here.
   - Additional miscellaneous (such as GIF files, images etc.) as requested in the assignment.
3. **Grading.** Please clearly show your work. Partial credit will be awarded for ideas *toward* the solution, so please submit your thoughts on an exercise even if you cannot find a full solution. *If you don't know where to get started with some of these exercises, just ask!* A great way to do this is to leave questions on our Piazza forum or to schedule reception hours with the TA in charge of the exercise. Maximal grade in this exercise is **120**, including all possible bonuses.
4. **Collaboration and External Resources.** You are encouraged to discuss all course material with your peers, including the written and coding assignments. You are especially encouraged to seek out new friends from other disciplines (CS, Math, EE, etc.) whose experience might complement your own. However, your final work must be your own, i.e., direct collaboration on assignments is prohibited. You are allowed to refer to any external resources - if you use one, cite such help on your submission. If you are caught cheating, you will get a zero for the entire course.
5. **Late Days.** Due to the large period of time given to this exercise's submission, we will accept no late day requests unless for official Technion reasons - reserve duty, hospitalization, or tragedy. Please be responsible with this
6. **Parenting Concessions.** By Technion regulations, parents are eligible for concessions with the homework. Please contact the Head TA for details if you are eligible.
7. **Warning!** With probability 1, there are typos in this assignment. If anything in this handout does not make sense (or is blatantly wrong), let us know!

1 Graph Signal Processing (Dry + Wet)

1. **Image Denoising**
   Consider an image $\boldsymbol{x}_0 \in \mathbb{R}^n$ in a column-stack representation contaminated with additive noise such that
   $$\boldsymbol{y} = \boldsymbol{x}_0 + \boldsymbol{n}$$
   where $n(i) \sim \mathcal{N}(0, 0.01)$ are independent and identically distributed noise elements.
   (a) A classical approach to recover $\boldsymbol{x}_0$ is by solving the following optimization problem
   $$\min_{\boldsymbol{x}\in\mathbb{R}^n} ||\boldsymbol{y} - \boldsymbol{x}||_2^2 + \gamma \boldsymbol{x}^\top \boldsymbol{L} \boldsymbol{x}$$
   where $\boldsymbol{L}$ is the Laplacian kernel in the matrix form and $\gamma > 0$ is the weight of the regularization term $\boldsymbol{x}^\top \boldsymbol{L}\boldsymbol{x}$, promoting minimization of the Dirichlet energy, as seen in R8.
   Prove that the optimal solution is given by
   $$\boldsymbol{x}^* = (\boldsymbol{I} + \gamma \boldsymbol{L})^{-1} \boldsymbol{y}$$
   (b) Take any image of appropriate size. That is, find an image $\boldsymbol{Z} \in \mathbb{R}^{n_x \times n_y}$ in gray scale such that n_x and n_y are less than or equal to 512. Compute the column-stack representation $\boldsymbol{z_0} \in \mathbb{R}^n$, where $n = n_x n_y$. Add noise $\mathcal{N}(0, 0.01)$ to each pixel. Plot the noisy image.
   We denote the noisy image by $\boldsymbol{Y} \in \mathbb{R}^{n_x \times n_y}$ and its column-stack representation by $\boldsymbol{y}$.
   (c) Implement the following image reconstruction approach:
   - Construct the sparse matrix $\boldsymbol{L} \in \mathbb{R}^{n\times n}$ with the kernel $\boldsymbol{h}_8$ such that $\boldsymbol{L}\boldsymbol{y}$ is equivalent to convolving $\boldsymbol{Y}$ with $\boldsymbol{h}_8$ (up to column-stack representation), where $\boldsymbol{h}_8$ is given by
   $$\begin{bmatrix} -\frac{1}{8} & -\frac{1}{8} & -\frac{1}{8} \\ -\frac{1}{8} & 1 & -\frac{1}{8} \\ -\frac{1}{8} & -\frac{1}{8} & -\frac{1}{8} \end{bmatrix}$$
   To avoid boundary effects, set the diagonal of $\boldsymbol{L}$ such that the sum of each row is 0. You may use built-in functions to create sparse matrices.
   - Solve $(\boldsymbol{I} + \gamma\boldsymbol{L})\boldsymbol{x} = \boldsymbol{y}$ using the conjugate gradient method[1]. Set the hyper-parameter γ manually (try to reduce the noise while preserving the details sharp). Make sure the algorithm converges and clip the output image values between 0 and 1.
   - Plot the obtained reconstructed image.
   (d) [*Bilateral filtering*] Repeat the above construction with the graph-Laplacian matrix $\boldsymbol{L}_G$.
   - Construct a sparse matrix $\boldsymbol{W}_G \in \mathbb{R}^{n\times n}$ assuming that each pixel is connected to its 8 neighboring pixels on the image.
   - Set the weights by
   $$\boldsymbol{W}_G(k,l) = \begin{cases} \exp\left(-\frac{||\boldsymbol{y}(k) - \boldsymbol{y}(l)||^2}{2\sigma^2}\right), & \text{if pixel } k \text{ is the neighbor of pixel } l \\ 0, & \text{otherwise} \end{cases}$$
   where σ is a tunable hyperparameter. Set $\sigma = 0.1$ here.
   - Construct the graph Laplacian matrix $\boldsymbol{L}_G$=$\boldsymbol{D}_G$-$\boldsymbol{W}_G$.
   - Solve $(\boldsymbol{I} + \gamma\boldsymbol{L}_G)\boldsymbol{x} = \boldsymbol{y}$ using the conjugate gradient method. Set γ manually. Plot the obtained reconstructed image and compare to the result above.

[1] https://docs.scipy.org/doc/scipy/reference/generated/scipy.sparse.linalg.cg.html

2. **3D Point Clouds Denoising**
   - Load a 3D point cloud example from ModelNet40[2]. Plot this point cloud.
   - Add noise $\mathcal{N}(\mathbf{0}, 0.01s^2\boldsymbol{I})$ to the 3D position of the nodes. Let $\{\boldsymbol{x}_i\}_{i=1}^n$, where $\boldsymbol{x}_i \in \mathbb{R}^3$, be the 3D position of the noisy nodes and let $\mathbf{X}$ be the matrix form of them. Plot the noisy point cloud.
     Note that here s is an adaptive parameter, commonly set as 1. However, if your 3D position of the point cloud is in a smaller scale, adapt the noise with a smaller variance.
   - Construct a sparse affinity matrix with a Gaussian kernel such that
     $$\boldsymbol{W}(i,j) = \begin{cases} \exp\left(-\frac{||\boldsymbol{x}_i - \boldsymbol{x}_j||^2}{2\sigma^2}\right), \text{if } ||\boldsymbol{x}_i - \boldsymbol{x}_j||^2 \leq r \\ 0, \text{otherwise} \end{cases}$$
     where r and σ are tunable hyperparameters.
   - Compute the normalized graph Laplacian $\boldsymbol{N} = \boldsymbol{I} - \boldsymbol{D}^{-\frac{1}{2}}\boldsymbol{W}\boldsymbol{D}^{-\frac{1}{2}}$.
   - Compute the eigen-decomposition of $\boldsymbol{N}$ and plot the sorted eigenvalues (from the smallest to the largest). Denote the corresponding eigenpair by $\{\lambda_i, \psi_i\}_{i=1}^n$.
   - Plot the noisy 3D point cloud colored by eigenvectors $\psi_1, \psi_2, \psi_4, \psi_{10}$.
   - A low-pass graph filter is defined by
     $$\widehat{\boldsymbol{h}}(x) = \exp\left(-\frac{\tau x}{\lambda_{max}}\right)$$
     where τ is a tunable hyperparameter and λ_{max} is the largest eigenvalue of $\boldsymbol{N}$.
     Plot $\widehat{\boldsymbol{h}}(\boldsymbol{\lambda})$ with $\tau = 3, 5, 10$, where $\boldsymbol{\lambda} = [\lambda_1, \lambda_2, \cdots, \lambda_n]^\top$.
   - Denoise the noisy 3D point cloud by
     $$\mathbf{\Psi}\mathbf{\Lambda}_h\mathbf{\Psi}^\top\mathbf{X}$$
     where $\mathbf{\Psi}$ is the eigenbasis of $\boldsymbol{N}$ in a matrix form, $\mathbf{\Lambda}_h$ is a diagonal matrix whose diagonal elements are $\widehat{\boldsymbol{h}}(\boldsymbol{\lambda})$, and $\mathbf{X}$ is the 3D position of the nodes in the matrix form.
   - Experiment with τ values of $\{3, 5, 10\}$. Plot the resulting point clouds and compare between them.

[2] https://modelnet.cs.princeton.edu

2 Spectral Clustering (Dry + Wet)

1. Let G be a weighted graph with K components $A_1, A_2, \dots, A_K$. That is, if $x \in A_i$ and $y \in A_j$, then
$$\boldsymbol{W}(i,j) = 0$$
for all $i \neq j$. Prove that the multiplicity of the smallest eigenvalue of the Laplacian matrix is exactly K.
2. Let $G = (V, E)$ be a graph and $S \subseteq V$ be a subset of its vertices, where $|V| := n$. We define the graph induced by G on S as
$$G(S) = (S, \{(i,j) \in E, i \in S \text{ and } j \in S\})$$
Let $\boldsymbol{L}$ be the Laplacian matrix of G with the corresponding eigenpairs $\{\lambda_i, \psi_i\}_{i=1}^{n}$, where the eigenvalues are sorted from the smallest to the largest. For any $a \geq 2$, let
$$K_a = \{i \in V : \psi_a(i) \geq 0\}$$
Show that $G(K_a)$ has at most $a - 1$ connected components.
3. Let f be a function defined on the vertices, given by
$$f(i) = \begin{cases} \sqrt{\dfrac{|\bar{A}|}{|A|}}, i \in A \\ -\sqrt{\dfrac{|A|}{|\bar{A}|}}, i \in \bar{A} \end{cases}$$
Prove that:
   - The quadratic form satisfies
   $$f^{\top} \boldsymbol{L} f = |V| \cdot \text{RatioCut}(A, \bar{A})$$
   where $\boldsymbol{L}$ is the Laplacian matrix.
   - The function f is orthogonal to $\mathbf{1}$.
   - $\|\boldsymbol{f}\|^2 = |V|$.

4. Load the ring5.npy data set $\{\boldsymbol{x}_i \in \mathbb{R}^3\}_{i=1}^{1000}$.
   (a) Apply K-Means[3] with $K = 5$ and plot $\{\boldsymbol{x}_i \in \mathbb{R}^3\}_{i=1}^{1000}$ colored by the obtained clusters.
   (b) Apply spectral clustering as follows
   - Construct the weight matrix $\boldsymbol{W}$ such that
   $$\boldsymbol{W}(i,j) = \begin{cases} \exp\left(-\dfrac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\sigma^2}\right), i \neq j \\ 0, \text{otherwise} \end{cases}$$
   where $\sigma = 0.2$.
   - Construct the Laplacian matrix by $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{W}$.
   - Compute eigen-decomposition of $\boldsymbol{L}$ and order the eigenvalues from the smallest to the largest. Take the first $(K+1)$ eigenpair $\{\lambda_i, \psi_i\}_{i=1}^{K+1}$.
   - Construct the following embedding of the vertices
   $$\boldsymbol{z}_i = \Phi(\boldsymbol{x}_i) = \begin{bmatrix} \dfrac{1}{\lambda_2}\psi_2(i) \\ \dfrac{1}{\lambda_3}\psi_3(i) \\ \vdots \\ \dfrac{1}{\lambda_{K+1}}\psi_{K+1}(i) \end{bmatrix} \in \mathbb{R}^K$$

[3] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.KMeans.html

- Apply K-Means to the embedding $\{\boldsymbol{z}_i\}_{i=1}^{1000}$

(c) Plot the first three coordinates of the embedding $\{\boldsymbol{z}_i\}_{i=1}^{1000}$ in $\mathbb{R}^3$.

(d) Plot $\{\boldsymbol{x}_i \in \mathbb{R}^3\}_{i=1}^{1000}$ colored by the obtained clusters.

5. Load the ring2.npy data set $\{\boldsymbol{x}_i \in \mathbb{R}^3\}_{i=1}^{600}$.

(a) Apply K-Means with $K = 2$ and plot the data colored by the obtained clusters.

(b) Apply spectral clustering as above. Because the two rings are close, set σ with a smaller value.

(c) Plot the first two coordinates of the embedding $\{\boldsymbol{z}_i\}$.

(d) Plot the original data set and color by the obtained clusters.

(e) Explain why the spectral clustering algorithm fails to cluster the two rings.
(Hint: You may like to check the degree matrix of the data set and compare it to the degree matrix of the ring5.npy data set.)

(f) Propose and implement an adjustment with the normalized Laplacian matrix so that the algorithm will cluster the two rings.

(g) Plot the new results.

3 SPD Riemannian Manifolds (Dry + Wet)

In this exercise, we consider the cone manifold of symmetric and positive definite (SPD) matrices, which is a convex half-cone in the vector space of $n \times n$ sysmetric matrices. Such a cone forms a differentiable Riemannian manifold $\mathcal{M}$ with the (affine-invariant) inner product

$$\langle \boldsymbol{S}_1, \boldsymbol{S}_2 \rangle_{\mathcal{T}_{\boldsymbol{P}}\mathcal{M}} = \langle \boldsymbol{P}^{-\frac{1}{2}} \boldsymbol{S}_1 \boldsymbol{P}^{-\frac{1}{2}}, \boldsymbol{P}^{-\frac{1}{2}} \boldsymbol{S}_2 \boldsymbol{P}^{-\frac{1}{2}} \rangle$$

where $\mathcal{T}_{\boldsymbol{P}}\mathcal{M}$ is the tangent space at $\boldsymbol{P} \in \mathcal{M}$, $\boldsymbol{S}_1, \boldsymbol{S}_2 \in \mathcal{T}_{\boldsymbol{P}}\mathcal{M}$, and $\langle \cdot, \cdot \rangle$ is the Euclidean inner product defined by $\langle \boldsymbol{A}, \boldsymbol{B} \rangle = \mathrm{Tr}(\boldsymbol{A}^\top \boldsymbol{B})$.

The unique geodesic curve between any two SPD matrices $\boldsymbol{A}, \boldsymbol{B} \in \mathcal{M}$ is given by

$$\phi(\boldsymbol{A}, \boldsymbol{B}, t) = \boldsymbol{A}^{\frac{1}{2}} \left(\boldsymbol{A}^{-\frac{1}{2}} \boldsymbol{B} \boldsymbol{A}^{-\frac{1}{2}} \right)^t \boldsymbol{A}^{\frac{1}{2}}$$

Its length induces a Riemannian distance on $\mathcal{M}$, which is given by

$$d_{\mathcal{R}}^2(\boldsymbol{A}, \boldsymbol{B}) = || \log(\boldsymbol{B}^{-\frac{1}{2}} \boldsymbol{A} \boldsymbol{B}^{-\frac{1}{2}}) ||_F^2 = \sum_{i=1}^{n} \log^2(\lambda_i(\boldsymbol{B}^{-\frac{1}{2}} \boldsymbol{A} \boldsymbol{B}^{-\frac{1}{2}}))$$

where $|| \cdot ||_F$ is the Frobenius norm, $\log(\boldsymbol{P})$ is the matrix logarithm, and $\lambda_i(\boldsymbol{P})$ is the i-th eigenvalue of $\boldsymbol{P}$.

The Logarithm map, projecting an SPD matrix $\boldsymbol{P_i} \in \mathcal{M}$ to the tangent space $\mathcal{T}_{\boldsymbol{P}}\mathcal{M}$ at $\boldsymbol{P} \in \mathcal{M}$, is defined by

$$\boldsymbol{S}_i = \mathrm{Log}_{\boldsymbol{P}}(\boldsymbol{P}_i) = \boldsymbol{P}^{\frac{1}{2}} \log(\boldsymbol{P}^{-\frac{1}{2}} \boldsymbol{P}_i \boldsymbol{P}^{-\frac{1}{2}}) \boldsymbol{P}^{\frac{1}{2}}$$

The Exponential map, project $\boldsymbol{S}_i \in \mathcal{T}_{\boldsymbol{P}}\mathcal{M}$ back to the manifold, is defined by

$$\boldsymbol{P}_i = \mathrm{Exp}_{\boldsymbol{P}}(\boldsymbol{S}_i) = \boldsymbol{P}^{\frac{1}{2}} \exp(\boldsymbol{P}^{-\frac{1}{2}} \boldsymbol{S}_i \boldsymbol{P}^{-\frac{1}{2}}) \boldsymbol{P}^{\frac{1}{2}}$$

1. Show that
   (a) $\phi(\boldsymbol{A}, \boldsymbol{B}, 0) = \boldsymbol{A}$ and $\phi(\boldsymbol{A}, \boldsymbol{B}, 1) = \boldsymbol{B}$.
   (b) $\phi(\boldsymbol{A}, \boldsymbol{B}, t) \preccurlyeq (1-t)\boldsymbol{A} + t\boldsymbol{B}$, where $\preccurlyeq$ is the Löwner partial order.
   (c) $\phi(\boldsymbol{A}, \boldsymbol{B}, t)^{-1} = \phi(\boldsymbol{A}^{-1}, \boldsymbol{B}^{-1}, t)$
2. [*Affine invariance*] Show that
   $$d_{\mathcal{R}}^2(\boldsymbol{A}, \boldsymbol{B}) = d_{\mathcal{R}}^2(\boldsymbol{X}\boldsymbol{A}\boldsymbol{X}^\top, \boldsymbol{X}\boldsymbol{B}\boldsymbol{X}^\top)$$
   where $\boldsymbol{X}$ is a real invertible $n \times n$ matrix.
3. Reconstruct the results in Section IV A in the paper[4]
4. **Bonus (10 points)** Reconstruct the results in Section IV B in the paper. The data is the Data set 2a in BCI competition IV[5]. Follow the instruction[6] to load the data sets.
   Apply a band-pass filter[7] with cutoff frequencies 8Hz and 30Hz before compute the covariance matrices.

[4] https://ronentalmon.com/wp-content/uploads/2019/03/TSP_Yair_2019.pdf
[5] http://www.bbci.de/competition/iv/
[6] http://www.bbci.de/competition/iv/desc_2a.pdf
Follow their Matlab code and save the data in .mat file. Load the .mat file to your code by `scipy.io.loadmat`
[7] https://docs.scipy.org/doc/scipy/reference/generated/scipy.signal.firwin.html