Assigned: 16 September 2021

Homework #2 – Numeric Computing

EE 541: Fall 2021

Due: Sunday, 26 September 2021 at 23:59. Late penalty: 10% per 24-hours before 28 September at 23:59. Submission instructions will follow separately on canvas.

1. Raman spectroscopy is a technique that uses inelastic scattering of light to identify unknown chemical substances. Spectral "peaks" indicate vibrational and rotational modes and are of special importance because they act like a chemical fingerprint. Raman spectroscopy measures photon intensity vs. *Raman shift*. The Raman shift relates the frequencies of the exciting laser and the scattered photons and is often reported as a wavenumber – the frequency difference in wavelengths per cm (*i.e.*, cm^{-1}).

    - Generate a molecular fingerprint using the spectroscopic data in `hw2p1.txt`. The file contains intensity vs. wavenumber data for an unknown chemical sample. Use the method below to estimate the wavenumbers of all spectral peaks. <u>You may use any standard `NumPy` or `SciPy` packages</u> or experiment with your own algorithms.

    - First detect peaks in the raw spectral data. Use the peak locations to focus on regions of interest apart from the complete dataset. For instance: if you detect peaks at x_1 cm^{-1} and x_2 cm^{-1} use regions of interest: $[x_1 - n_1,\ x_1 + n_1]$ and $[x_2 - n_2,\ x_2 + n_2]$. Experiment to find "good" widths n_1, n_2, etc. Then use a spline to interpolate intensity within each region of interest. Calculate zero-crossings of the derivative to estimate wavenumbers with <u>maximum</u> intensity.

        1) Print the wavenumber estimate for each spectral peak to `STDOUT` (in any order is OK).

        2) Create a figure that shows the Raman data (intensity vs. wavenumber) and mark each of the maximum intensity values.

        3) Produce a "zoomed-in" figure for at least four "regions of interest". Plot the raw spectral data and overlay your interpolating function. Use a marker to show wavenumber(s) with maximal intensity.

2. Unsupervised clustering algorithms are an efficient means to identify groups of related objects within large populations. Implement the following two clustering algorithms and apply them to the data in `mickey.csv`. The file contains data as: x, y, $class$. Use a regular expression to remove lines that are empty or that are invalid data. You can safely ignore any line that fails the regular expression.

    a. Use K-Means clustering with 3-clusters to label each (x, t) pair as "Head", "Ear_Right", or "Ear_Left". <u>You may use any standard `NumPy` or `SciPy` packages</u> or experiment with your own implementation.

    Produce a scatter plot marking each (x, y) pair as either BLUE (class = "Head"), RED (class = "Ear_Left") or GREEN (class = "Ear_Right"). Compare the K-means predicted labels to the true label and generate a confusion matrix showing the respective accuracies.

b. Gaussian Mixture Models (GMM) are a common method to cluster data from multi-modal probability densities. Expectation maximization (EM) is an iterative procedure to compute (locally) optimal GMM parameters – GMM cluster means μ_k, covariances Σ_k, and mixing weights w_k. EM consists of two-steps. The **E[**xpectation]-step uses the mixture parameters to update estimates of *hidden variables*. The "true" but unknown class is an example of a hidden variable. The **M[**aximization]-step then uses the new hidden variable values to update the mixture parameter estimates. This back-and forth updating provably increases the likelihood function and it eventually converges to a local maximum. The GMM update equations follow.

E-Step: use mixture parameters to estimate membership probabilities (*a.k.a.*, the hidden variables) for each sample, $\gamma_k(x_n)$,

$$\gamma_k(x_n) = \frac{w_k\, f(x_n;\, \mu_k, \Sigma_k)}{\sum_{j=1}^{K} w_j\, f\big(x_n; \mu_j, \Sigma_j\big)}$$

M-step: use new $\gamma_k(x_n)$ to update mixture parameter estimates,

$$\mu_k = \frac{\sum_{n=1}^{N} \gamma_k(x_n)\, x_n}{\sum_{n=1}^{N} \gamma_k(x_n)}$$

$$\Sigma_k = \frac{\sum_{n=1}^{N} \gamma_k(x_n)\,(x_n - \mu_k)(x_n - \mu_k)^{\mathrm{T}}}{\sum_{n=1}^{N} \gamma_k(x_n)}$$

$$w_k = \frac{1}{N}\sum_{n=1}^{N} \gamma_k(x_n)$$

for $k \in \{1, 2, \dots, K\}$ where $K \in Z^+$ is the (predefined) number of mixture components, x_n for $n \in \{1, 2, \dots, N\}$ are the data samples, and $f(x;\, \mu_k, \Sigma_k)$ is the d-dimensional jointly Gaussian pdf:

$$f(x;\, \mu_k, \Sigma_k) = \frac{\exp\left(-\frac{1}{2}(x - \mu_k)^{\mathrm{T}}\, \Sigma^{-1}\, (x - \mu_k)\right)}{\sqrt{(2\pi)^d\, |\Sigma_k|}}.$$

- Implement Expectation Maximization and use it to estimate mixture parameters for a 3-component GMM on the `mickey.csv` dataset. **<u>DO NOT</u>** use an EM implementation from `NumPy`, `SciPy`, or any other package. You must implement and use the above EM equations.

- Initialize $\gamma_k(x_n)$ for each sample using the K-means labels from part (a) as *"one-hot"* membership probabilities (i.e., initialize one of the probabilities is "1" and all others are "0"). Then compute initial μ_k and Σ_k for each component.

- Run EM until it has sufficiently converged. Assign each datapoint to the mixture component with the largest membership probability. Produce a Blue-Red-Green scatter plot as in part (a) and generate a confusion matrix showing the respective classification accuracies.

- Generate figures showing the class assignments during the first four iterations.
- Comment on the difference of the clustering result in (a) and (b). Describe any obvious difference between the plots and indicate which performs better.