Homework 2

AMATH 482, Winter 2025

Assigned Jan 30, 2025. Checkpoint due Feb 7th, 2025. Report and Code due on Feb 14th, 2025 at midnight.

Directions, Reminders and Policies

Read these instructions carefully: There are two stages for submitting the assignment.

- **You will be submitting a *Checkpoint* approximately one week after the assignment was published (see due date above). The checkpoint includes submission of your Report and Code in progress (at least 1/3) (2 points) and taking the checkpoint quiz (3 possible points).**

- The report should be a maximum of 6 pages long with references included. Minimum font size 10pts and margins of at least 1inch on A4 or standard letter size paper.
- Do not include your code in the report. Simply create a zip file of your main scripts and functions, without figures or data sets included, and upload the zip file to Canvas.
- Your report should be formatted as follows:
  - Title/author/abstract: Title, author/address lines, and short (100 words or less) abstract. This is not meant to be a separate title page.
  - Sec. 1. Introduction and Overview
  - Sec. 2. Theoretical Background
  - Sec. 3. Algorithm Implementation and Development
  - Sec. 4. Computational Results
  - Sec. 5. Summary and Conclusions
  - Acknowledgments (no more than four or five lines, also see the point below on collaborations)
  - References

- LaTeX(Overleaf is a great option) is recommended to prepare your reports. A template is provided on Canvas in Homework/Files. You are also welcome to use Microsoft Word or any other software that correctly typesets mathematical equations and properly allows you to include figures.
- Collaborations are encouraged; however, everything that is handed in (both your report and your code) should be your work. You are welcome to discuss your assignments with your peers and seek their advice but these should be clearly stated in the acknowledgments section of your reports. This also includes any significant help or suggestions from the TAs or any other faculty in the university. You don't need to give all the details of the help you received, just a sentence or two. A similar guideline applies to the use of Large Language Models (LLM). These are permitted for the study of topics and code presented in class and a better grasp of the problem and its solution. However, everything that is handed in (both your report and your code) should be your work and cannot be based on LLM content (modified or direct). Any use of external help should be specified in the acknowledgments section of the report.
- **Late reports are subject to a 2 points/day penalty up to five days. They will be no longer accepted afterwards. For example, if your report is three days late and you managed to get $16/20$, your final grade will be $16 - 6 = 10$, so be careful with late submission.**

Problem Description: Captured Motion Recognition

You are working on the next version of a humanoid robot OptimuS-VD which has built-in sensors that record the movements of 38 of its joints with rate of 60Hz. The movements of the joints are recorded as Euler angles and can be transformed to xyz coordinates. You have recorded 5 samples of each of the 3 movements that OptimuS-VD knows to perform: walking, jumping, and running. The samples are recorded for 1.4 secs (100 timesteps) and saved as a matrix of 114×100, where the first dimension records $x_1, ..x_{38}, y_1, ..y_{38}, z_1, ..z_{38}$ locations of the joints and the second dimension the timesteps. Your goal is to build a projection of the recordings to a lower dimension than the number of coordinates, visualize the movements and then based on it to design an algorithm that will be able to recognize which movement OptimuS-VD is performing in real-time. A test sample of each movement is available to you to test your approach.

You can download the data using the Google drive links on Canvas; either the data files `hw2datanpy.zip` for Python users, or `hw2datamat.zip` for MATLAB users.

These files contain two folders: train and test. Within each, each *npy* or *mat* file is a matrix of 114×100 in format explained above.

Some comments and hints

Here are some useful comments and facts to guide you along the way.

1. `sklearn` has functionalities for PCA and many other functions (e.g. computing accuracy) that could be useful for completing this homework. Using these can make your life easier.
2. Don't forget to center X_{train} before computing the PCA modes if you plan to use SVD. If you are using `sklearn`'s PCA function then you don't need to worry about this as it centers the data by default. Make sure to check the convention of `sklearn` in terms of rows and columns of X_{train} and transpose if necessary.
3. Recall that the projection of a given sample z onto k-PC modes is achieved by $z_k = U_k^T z$, where U_k^T is the transposed U matrix from SVD with first k column vectors kept intact and column vectors from $k+1$ and onward are set to be zero vectors.
4. The provided notebook will visualize a given sample in time as a skeleton in xyz coordinates. See also the GIF file that shows movements in time.
5. The data is a sub-sample of the CMU MoCap database http://mocap.cs.cmu.edu. There are additional datasets and benchmarks that capture motion such as (Human 3.6) http://vision.imar.ro/human3.6m/description.ph (NTU -RGBD+) https://github.com/shahroudy/NTURGB-D, etc.

Tasks

Below is a list of tasks to complete in this assignment and discuss in your report.

1. Compile all train samples into a matrix X_{train} and apply PCA such that the *PCA modes* are spatial modes and the *coefficients* are time-dependent coefficients. Investigate how many PCA spatial modes you need to keep to approximate X_{train} up to 70%, 80% , 90% , 95% in Frobenius norm (i.e., energy). Plot the cumulative energy to justify your results.
2. Truncate the PCA modes set to 2 and 3 modes and plot the projected X_{train} in the truncated PCA space as low dimensional 2D (PC1,PC2 coordinates) and 3D (PC1,PC2,PC3 coordinates) trajectories. Use colors for different movements and discuss visualization and your findings.
3. In order to classify each sample with type of movement establish the following ground truth. Create a vector of **ground truth labels** with an integer per class, e.g., 0 (walking), 1 (jumping), 2 (running) and assign an appropriate label to each sample in X_{train}. Then for each movement compute its **centroid** (mean) in k-modes PCA space.
4. Having the ground truth, preform the following training. Create another vector of **trained labels**. To assign these labels, for each sample in X_{train} compute the **distance** between the projected point in k-modes PCA space and each of the centroids. The minimal distance will determine to which class the sample belongs. Assign the label of the class of the centroid with minimal distance in the trained labels vector. Compute the trained labels for various k values of k-PCA truncation and report the accuracy of the trained classifier (the percentage of samples for which the ground truth and the trained labels match). You can use *accuracy_score* function in *sklearn* for this purpose. Discuss your results in terms of optimal k for the classifier accuracy.
5. To test how the classification performs on classification/recognition of new samples, load the given test samples and for each test sample assign the ground truth label. By projecting onto k-PCA space and computing the distance to the centroids, **predict the test labels**. Report the accuracy of the classifier on the test samples. Discuss and compare it with trained accuracy. Try various k values.
6. **Bonus (+2 points)**: Implement an alternative classifier based on k-PCA space and compare with your results above.