BE 521: Homework 3

Feature extraction

Spring 2015

68 points

Due: Thursday, 2/12/2015 11:59pm

Objective: Extract features from data and build a simple detector

Homework Policy

1. Piazza should be used for peer discussion for all questions related to course material. Please also use Piazza to contact teaching staff for all questions. TA's will be available to help during office hours and occasionally on Piazza.

2. Submit LaTeX write-up (pdf) and Matlab code to Canvas as pennkey_hwx.pdf, .m before listed deadline.

3. Assignments will be returned electronically on Canvas.

4. Collaboration is encouraged but individual write-ups are required. Please list any collaborators. Honor code will be strictly enforced. Note: submitted code is routinely passed through a plagiarism checker.

5. Late Policy: 5% per day. **No homework is accepted after the 5th late day.** (e.g. If originally due Tuesday, 11:59PM, last day to turn in is Sunday, 11:59 PM).

1   Features and Simulations (39 pts)

As you learned in class, features are the backbone of almost all detection strategies, from seizures in EEG to faces in images. Features are usually defined in journal articles as an equation or set of equations, and the task for the reader—if she wants to use that feature—is to implement that feature in code. In this section, you will explore and implement some features commonly used in EEG analysis and test them on simulated time-series data.

1. Consider the following toy signal: 3 seconds of a 2 Hz sine wave with zero phase shift, sampled at 100 Hz

   (a) Plot the signal. (2 pts)

   (b) Using the Matlab functions for the difference, sum, and absolute value of the elements in a vector (look them up if you don't know them), create an anonymous function for the line-length feature

   $$LL(\mathbf{x}) = \sum_{i=2}^{n} |x_i - x_{i-1}|$$

in one line of Matlab code that uses no loops (i.e., the outputs of one function will be the inputs of another). Your function should look something like

```
LLFn = @(x) XXXXXX;
```

where **XXXXX** represents some use of the aformentioned functions and the input signal **x**. (4 pts)

(c) What is the line length of this signal? (2 pts)

2. Consider line length of the signal using a sliding window with a certain amount of window overlap (or, to think of it another way, displacement with each "slide"). Now, instead of having just one value for the line length, you will have a number of values.

(a) Given a signal **x** with sampling frequency **fs** and windows of length **winLen** and displacement **winDisp** (both in seconds), create an anonymous function called **NumWins** that calculates the number of possible (full) windows in your signal of length **xLen** (in samples), i.e.,

```
NumWins = @(xLen, fs, winLen, winDisp) XXXXXX;
```

where **XXXXXX** is the single-line expression for this value. You may assume that **winDisp** is a multiple of both **winLen** (as it usually is/should be) and the length (in seconds) of **x**. (4 pts)

(b) Use this function to calculate the number of windows for the signal described in Question 1.1 for a 500 ms window with 250 ms displacement, i.e., the expression

```
NumWins(length(x), fs, winLen, winDisp)
```

where **fs**, **winLen**, and **winDisp** are the appropriate values. (1 pt)

(c) Repeat the above calculation for 500 ms window displacement. (1 pt)

(d) Repeat the above calculation for 100 ms window displacement. (1 pt)

3. (a) Create a function (in another file) called **MovingWinFeats(x, fs, winLen, winDisp, featFn)** that returns a vector of the values of the feature on the signal **x** in all the possible windows, where **featFn** is a feature function like the one you wrote in Question 1.1.b. You may find it useful to use your **NumWins** function (or at least its expression). You may assume that the product of **winDisp** and the sampling rate **fs** is an integer. (6 pts)

Use the following Matlab code (in your script) to automatically load in the function's code (where we assume that the function is one directory up from the *.tex file). Windows users may need to change the forward-slash to a backslash.

```
%%
% <latex>
% \lstinputlisting{../MovingWinFeats.m}
% </latex>
```

(b) Using the signal you defined in Question 1.1 and the function you created in Question 1.1.b, calculate the line-length over windows of length 500 ms and displacement 250 ms. (2 pts)

(c) Add a unit-amplitude 5 Hz signal to your original signal and again calculate the line length over the same window and displacement. (2 pts)

4. Code the following 3 additional features in MINIMAL lines of code (hint: each can be implemented in one line using the anonymous function trick).

   (a) Area, $A(\mathbf{x}) = \sum_{i=1}^{n} |x_i|$ (2 pts)

   (b) Energy, $E(\mathbf{x}) = \sum_{i=1}^{n} x_i^2$ (2 pts)

   (c) Zero-Crossings around mean,
   $ZX(\mathbf{x}) = \sum_{i=2}^{n} \mathbf{1}(\textbf{FromAbove})$ OR $\mathbf{1}(\textbf{FromBelow})$, where $\mathbf{1}(\cdot)$ denotes the indicator function, which returns a zero if its argument is false and a one if it is true, **FromAbove** denotes $(x_{i-1} - \overline{x} > 0)$ AND $(x_i - \overline{x} < 0)$, **FromBelow** denotes $(x_{i-1} - \overline{x} < 0)$ AND $(x_i - \overline{x} > 0)$, and $\overline{x}$ is the mean value of the elements in x. (4 pts)

   (d) Plot the values of the four features over a 500 ms window with 100 ms displacement in the first four cells of a 3x2 Matlab subplot. Using the right-aligned convention (where the "official" time of the feature is that of the last data point in the window), give the appropriate time axis for each window point. In addition, plot the original signal with the 2Hz and 5Hz components in the last two cells of the 3x2 subplot (to make comparing down the column easy). Ensure that the time axis in all of your plots is the same. (6 pts)

2   Feature Overlays (17 pts)

In this section, you will use a line-length feature overlay on a segment of EEG containing a seizure. This data is stored in `I521_A0003_D001`

1. What is the length using hours:minutes:seconds:milliseconds of the recording? (2 pts)

2. How many data points should we discard at the end if we want to clip the recording to the last full second? Do this clipping. (1 pt)

3. If we want to overlay a feature trace on the original signal, we have to interpolate that feature (which has been calculated over windows) for each data point of the original signal. One of the simplest methods of doing this is called zero-order interpolation, where we just hold the value constant until we get to the next calculated value. For example, if we had a moving window of 1 second with 1 second displacement, the zero-order interpolated feature vector would have the same value the entire first second, then the same for the entire second second, etc, where each second contains the same number of points as the sampling frequency of the original signal.

(a) Using the `repmat` and `reshape` functions, create an external function `zoInterp(x, numInterp` that copies each value of `x numInterp` times. You can implement this function in one line of code with no loops. Include the code for this function as you did in Question 1.3.a. (2 pts)

(b) Confirm that this function works correctly by expanding the length of the vector `1:5` by a factor of 10 and plotting with the command

```
plot(zoInterp(1:5,10),'-o')
```

where the `'-o'` option lets us see the individul points as well as the line that connects them. (2 pts)

4. Using a 4-second sliding window with 1-second displacement, calculate the line length feature over the entire signal. Normalize the line-length feature values to have a maximum twice that of the original EEG signal maximum. Plot the signal in blue and overlay the right-aligned line-length feature in green. N.B., you will need to pad the beginning of your feature vector and the end of your signal in order to get them two to line up correctly and be the same length. Put the units of your time axis in minutes, and be sure to add a legend in a location in the plot that does not cover up any signal or feature. (6 pts)

5. What threshold might you use on the raw line-length feature vector (not the normalized one used for plotting) in order to capture the 17 largest pre-seizure chirps that occur? (1 pt)

6. Using this threshold value, in another plot draw red vertical lines at the leading point in time where the threshold is crossed. Add these vertical lines on top of the plot you made in Question 2.4. These events should capture the pre-seizure chirps, the seizure onset, and some flickering during the end of the seizure. (3 pts)

3   Building a Detector (12 pts)

In this section, you will use the features you defined previously to build a seizure detector. Use the EEG data in the file `I521_A0003_D002` with channels `multiSz_1`, and `multiSz_2`.

1. Plot the signal in `multiSz_1` and draw vertical red lines at the times when you think the two seizures begin. (You should be able to do this without the need of any features.) (2 pts)

2. Produce feature overlay plots similar to that of Question 2.4 for each of the four features you have implemented along with the red vertical lines at each seizure. Use the same 4-second sliding window with 1 second displacement. (4 pts)

3. (a) Based on your plots in the previous question, which of the four features seems to give the largest signal (relative to the background) for when a seizure occurs? Explain. (3 pts)

   (b) What threshold would you use to determine if a seizure is occuring? (1 pt)

4. The signal in `multiSz_2` contains another seizure (whose location should again be fairly obvious). Plot the data along with the feature and threshold (horizontal black line, with

correct normalization for the signal in `data2`) you determined in the previous question. (2 pts)