Support Tip

How to add methods to BrainVision Analyzer quickly and easily? An example of the interactive Matlab interface

by Dr. Roland Csuhaj

New signal processing methods are published virtually every day; BrainVision Analyzer 2 cannot have all of them. However, the Analyzer is not just a powerful tool for offline EEG data evaluation - it is also a flexible framework which can integrate signal processing functions from many different sources. Beside the macros and add-ins, the Analyzer offers an exciting possibility to export your data to Matlab, do some calculations there and import the results back to Analyzer in order to continue your work in its user friendly environment. Today I am going to show how quickly a new method can be added to Analyzer using the Matlab Transformation.

Let's look at the *fraction peak latency* method, which is a technique to detect onset latency of components. The fractional peak latency marks the time point, when a certain percentage of the peak amplitude (e.g. 50%) was reached in the backward direction. Although such a method is not yet implemented in Analyzer, we can take advantage of the existing transformations: 'Peak detection' can perform the first step, and identify the peak of the average nodes. The Matlab transformation can quickly send the data into Matlab in order to do the rest of the calculation there. (In this article I am not going to introduce all options of the transformation, only those which are important for our goal). Open an average node, where the peaks are already detected and start the Matlab transformation. In the first window, mark the 'Calculate Data on Creation of Node' radio button, in the second one activate the 'Export Markers'. The transformation executes the Matlab commands typed in the first window. You do not have to enter hundreds of lines here, it is enough to call the .m file(s). The 'Show Matlab Window' is pretty useful while the code is still polished. For those who are more familiar with Analyzer, this is similar to the semi-automatic mode: the data can be checked or even modified manually in Matlab before it is imported back to Analyzer. Since we have just started to deal with this transformation, simply mark the three mentioned boxes, but not the others and enter the following text to the Code Executed … field:

```
desktop;
```

It will only show you the familiar Matlab desktop instead of the command line. Once the data is sent to Matlab, a dialog with 'Press OK to continue in Analyzer' text will appear. Try to resist, and do NOT click on OK until you are sure you are ready to close Matlab and start the re-import phase. Once the Matlab main window is started, you can have a look at the data. Many different properties of the node were exported, for now only the 'EEGData' and the 'Markers' variables are important.

What should be done to mark the fractional peak? We need a loop which checks all markers. If a 'Peak' marker is found (type is stored in Markers.Type properties) the value of the corresponding data point is needed from the EEGData variable. The fractional value can be calculated easily and can be used as threshold. A second loop should be started that searches for the data point in the backward direction at which the threshold is reached. This is the point where the new marker has to be placed. During the re-import, the Analyzer will look for the variable name 'NewMarkers', its contents will be added to the existing marker list. Certainly it must have the same structure as the 'Markers' variable. So in the next run we can enter the following code into the Matlab transformation dialog:

```
Threshold = 50;
NbPeak = 1;
for i = 1 : size(Markers,2)
    if strcmp(Markers(1,i).Type,('Peak'))
        Pos = Markers(1,i).Position+1;
        PCh = Markers(1,i).Channel+1;
        PValue = abs(EEGData(Pos, PCh));
        Pos = Pos - 1;
        while Pos > 0
            CurrentValue = abs(EEGData(Pos, PCh));
            if CurrentValue <= PValue*Threshold*0.01
                NewMarkers(1,NbPeak) = Markers(1,i);
                NewMarkers(1,NbPeak).Description = ...
                strcat(Markers(1,i).Description,,_', num2str(Threshold));
                NewMarkers(1,NbPeak).Position = Pos - 1;
                NbPeak = NbPeak +1;
                break
            end
            Pos = Pos - 1;
        end
    end
end
desktop;
```

The percentage is defined by the 'Threshold' variable. The rest is quite straightforward, except for one point: the 'Markers' variable refers to the channel and data position according to the C# convention, where all indexing starts with 0. But the EEGData matrix cannot be formed according to this convention because in Matlab all indexing starts with 1. This difference has to be compensated for by the code when it looks for the corresponding data value (by adding 1) and also when it creates the new marker (by subtracting 1).

Once you are sure the code is running fine, the desktop command in the last line is not needed anymore and the 'Show Matlab Window' checkbox can be deactivated. The transformation will now run automatically. As you can see, only 21 lines were needed to add a new method to Analyzer. It is a fast and effective way to implement new functions.

The program codes are also available at www.brainproducts.com/downloads.php?kid=21 as .m file. This version contains more explanation as additional comment lines. ●