




# AXIS DATA IN MATLAB

## USER GUIDE

February 2017

## Trademarks

Axion BioSystems, Inc. and  are trademarks of Axion BioSystems, Inc., and may not be used without the express written permission of Axion Biosystems, Inc.

All other brands, product names, company names, trademarks, and service marks are the properties of their respective owners.

## Restrictions and Liabilities

This document is provided “as is” and Axion BioSystems will assume no responsibility for any typographical, technical or other inaccuracies in this document. Axion BioSystems does not make any commitment to provide any changes, updates, or enhancements to this document within any time frame or at all.

This document might contain references to third-party sources of information, hardware or software, products or services and/or third-party websites (collectively the “Third-Party Information”). Axion BioSystems has no control over and is not responsible for any Third-Party Information, including without limitation the content, accuracy, copyright compliance, compatibility, performance, trustworthiness, legality, decency, links or any other aspect of Third-Party Information. The inclusion of Third-Party Information in this document does not imply endorsement with the use of Axion BioSystems technology.

## Conditions of Use

The user is responsible for understanding and performing the protocols that are described within. Axion BioSystems makes no guarantee for any results. These protocols are provided as a recommendation by Axion BioSystems based on use and experience.

## Copyright Notice

Copyright © 2017 Axion BioSystems, Inc. All rights reserved. This document may not be reproduced, distributed, modified or publicly displayed without the express written permission of Axion BioSystems.

# TABLE OF CONTENTS

1. Objective .....	1
2. Importing AxIS Data into MATLAB .....	1
2.1. Load all file data .....	1
2.1.1. Load an AxIS file .....	1
2.1.2. Load all raw data .....	2
2.1.3. Load all spike data .....	2
2.2. Load a portion of the data .....	3
2.2.1. Load data from a single electrode .....	3
2.2.2. Load specific data of interest .....	3
3. Isolating Data of Interest .....	4
3.1. Well and Electrode naming conventions .....	4
3.2. Retrieving voltage and time vectors from data array .....	4
3.2.1. Voltage and Time vectors from a single electrode .....	5
3.2.2. Voltage and Time vectors from multiple electrodes at once .....	5
4. Accessing Stimulation Event Tags in Matlab .....	5
5. Example Applications .....	6
5.1. Plot raw data .....	6
5.2. Plot spike data .....	7
5.3. Plot raster plot .....	8

## 1. OBJECTIVE

MATLAB® is an interactive environment for numerical computation, visualization, and programming that can be used to analyze data collected in AxIS. This document describes how to import raw and spike data into MATLAB, locate data of interest, and generate basic visualizations, including plots of raw voltage data and raster plots that show spike timing.

## 2. IMPORTING AXIS DATA INTO MATLAB

Axion provides MATLAB functions that load data from an AxIS file into Matlab. Axion Matlab functions can be found in C:\Program Files\Axion BioSystems, Inc\Axion Integrated Studio (AxIS)\Axion MATLAB Files. These functions import both AxIS raw files (".raw" extension) and AxIS spike files (".spk" extension).

The sample data referenced in this guide can be found in the folder Libraries/Documents/Public Documents/Axion BioSystems/Manuals and Documentation/Sample Data. This guide uses both the 48\_Well\_Neural\_Demo(000).raw and 12\_Well\_Cardiac\_Demo(000).raw files. A spike file can be generated from the raw neural data using AxIS (refer to the AxIS user guide, section 1.10, for detailed instructions).

### 2.1. LOAD ALL FILE DATA

Generally, it is helpful to load all of the file data into Matlab. Although it takes some time and memory to load all of the data at the same time, once the data is in Matlab, specific portions of the data are quickly and easily accessible.

#### 2.1.1. Load an AxIS file

To load an AxIS file into Matlab, be sure the Axion Matlab functions folder is in the Matlab path and type the following command at the Matlab prompt:

```
FileData=AxisFile('Filename.raw');
```

This loads a structure called FileData that contains the filename, experimental notes that were entered into AxIS at the time of data acquisition (Experiment ID, Investigator, and Description), the file version number of the file, the plate map, and any tag events associated with the file.

The notes are stored in strings, which can be accessed from the loaded file by calling:

```
Investigator = FileData.Notes.Investigator;  
ExperimentID = FileData.Notes.ExperimentID;  
Description = FileData.Notes.Description;
```

### 2.1.2. Load all raw data

To load all of the raw data into Matlab, use the command

```
AllData=FileData.DataSets.LoadData;
```

Alternatively, if experimental information is not needed, load the data using the single-step command

```
AllData=AxisFile('Filename.raw').DataSets.LoadData;
```

Either command will load the file data into a 4-dimensional cell array which contains the raw data for each electrode in every well. The dimensions of the AllData cell array correspond to the number of (Well Rows) x (Well Columns) x (Electrode Columns) x (Electrode Rows). For raw data, there is only one signal per electrode. Each cell of AllData contains a waveform structure with the following variables:

1. Channel – information about the channel the data came from, including the well row and column, and the electrode row and column
2. Start – the time in the recording at which this data begins
3. Data – the voltage data points, unscaled and stored as an int16s
4. Source – a block vector set containing references to the acquisition parameters such as sampling frequency, voltage scale factor, and experiment settings.

### 2.1.3. Load all spike data

To load all data from a spike file into Matlab, type the following command at the Matlab prompt:

```
AllData=AxisFile('Filename.spk').DataSets.LoadData;
```

This will load the file data into a 4-dimensional cell array with dimensions corresponding to the number of (Well Rows) x (Well Columns) x (Electrode Columns) x (Electrode Rows). Spike data can contain multiple spike waveforms for each electrode, so each cell in AllData contains an array of waveform structures corresponding to individual spikes detected on each electrode. Each waveform structure contains the following variables:

1. TriggerSampleOffset – the number of samples that the waveform starting point precedes the detected spike time (the spike detection method and the time window surrounding spike are defined in AxIS at acquisition).
2. StandardDeviation – the voltage noise value (standard deviation of the voltage signal) determined by AxIS at the time the spike was detected.
3. ThresholdMultiplier – the value multiplied by the StandardDeviation noise value to determine the voltage threshold used for spike detection. Any voltages above this threshold were captured as spikes.
4. Channel – information about the channel the spike came from, including the well row and column, and electrode row and column

5. **Start** – the first time point of the spike waveform captured by AxIS. Generally this is several samples before the detected “spike time.” The precise timing, relative to spike time, was specified by the user in AxIS.
6. **Data** – the spike voltage waveform (waveform length specified by the user in AxIS), unscaled and stored as an int16.
7. **Source** – a block vector set containing references to the experiment information such as sampling frequency, voltage scale factor, and experiment settings.

## 2.2. LOAD A PORTION OF THE DATA

In order to save memory and increase processing speed, users may use the LoadData input arguments to specify specific portions of the data to load, rather than loading all of the data. The optional inputs include well, electrode, timespan, and dimension.

### 2.2.1. Load data from a single electrode

To load the data from Well A2 Electrode 31 from 10 s to 30 s use the command

```
Data=AxisFile('Filename.raw').DataSets.LoadData('A2','31',[10 30]);
```

This returns a 4-D cell array in which all cells are empty except for Data{1,2,3,1}, which contains a waveform structure containing the data collected from Well A2 Electrode 31, from 10 to 30 s.

### 2.2.2. Load specific data of interest

It is not necessary to specify all of the optional inputs – if inputs are missing, the function will default to *all*. For example, to load the data from all electrodes in Well B1, use the command

```
Data=AxisFile('Filename.raw').DataSets.LoadData('B1');
```

This returns a 4-D cell array in which all cells are empty except for Data{2,1,:,:), each of which contains a waveform structure as described above in Section 2.1.1.

To load the data from the entire recording for electrode 31 from all wells, use the command

```
Data=AxisFile('Filename.raw').DataSets.LoadData('31');
```

To load the data from 15-25 s for all wells and electrodes, use the command

```
Data=AxisFile('Filename.raw').DataSets.LoadData([15 25]);
```

### 3. ISOLATING DATA OF INTEREST

To analyze raw voltage data from a particular well or electrode, load the entire data file as described in Section 2.1 or load a single well and electrode as described in Section 2.2. To use the voltage data from a specific electrode, refer to the cell in the Data array corresponding to the desired {WellRow, WellColumn, ElectrodeColumn, ElectrodeRow} (i.e. Data{2,1,3,1} for Well B1 Electrode 31, described below in section 3.1).

#### 3.1. WELL AND ELECTRODE NAMING CONVENTIONS

Note that when loading data into MATLAB using AxisFile (described in Section 2), well names are specified as strings using the traditional letter-number combination, in which the letter indicates the well row and the number indicates the well column, beginning from the top left corner of the plate. For example, in a 12-well plate, the upper left well is 'A1', the upper right well is 'A4', the lower left well is 'C1' and the lower right well is 'C4'. Once the data has been loaded into the 4-D cell array, a particular well is identified using numbers for both well row and well column (not letters for well rows). The well rows are numbered from top to bottom of the plate. Therefore, row 'A' corresponds to row 1, row 'B' to row 2, etc. To isolate the waveform containing the data from all electrodes in Well B3, for example, the command will be Data{2,3,:}.

Electrodes are named using a number-number combination, in which the first number indicates the electrode column, and the second number indicates the electrode row, beginning from the bottom left corner of the well. For example, in a 12-well plate, which contains 64 electrodes per well, the bottom left electrode in the well is '11', the top left electrode is '18', the bottom right electrode is '81' and the top right electrode is '88'. These numbers correspond to the labels in the top left corner of the plots in AxIS. When loading data into Matlab using AxisFile (described in Section 2), electrode names are specified as strings. Once the data has been loaded into the 4-D cell array, a particular electrode is identified using the numbers for electrode column and electrode row separately. For example, to isolate the waveform containing the data from Well B3 Electrode 67, the command will be Data{2,3,6,7}.

#### 3.2. RETRIEVING VOLTAGE AND TIME VECTORS FROM DATA ARRAY

The data stored in the waveform for each electrode must be scaled by a voltage scaling factor found in the Source.Header of each waveform. Axion provides subfunctions to automatically scale the voltage data and generate the corresponding time vectors. These subfunctions include:

1. `GetVoltageVector` – returns the scaled voltage vector (spike waveform vector in .spk files)
2. `GetTimeVector` – returns a time vector based on the start time, length of data, and sampling frequency
3. `GetTimeVoltageVector` – returns both the time and voltage vectors

### 3.2.1. Voltage and Time vectors from a single electrode

To find the voltage recorded on Well C2 Electrode 34 (WellRow=3, WellCol=2, ElecCol=3, ElecRow=4), use the command

```
data=AllData{3,2,3,4}.GetVoltageVector;
```

This returns the voltage recorded from Well C2 Electrode 34. Likewise, to get the corresponding time vector associated with this voltage, use the command

```
time=AllData{3,2,3,4}.GetTimeVector;
```

To get both voltage and time vectors simultaneously, use the command

```
[time,data]= AllData{3,2,3,4}.GetTimeVoltageVector;
```

### 3.2.2. Voltage and Time vectors from multiple electrodes at once

To create a matrix of voltages from multiple electrodes (for example, all electrodes in well C2), first concatenate all of the waveforms into a single structure and then return the voltage data from these channels by using the two-step command

```
temp=[AllData{3,2, :, :}];  
data=temp.GetVoltageVector;
```

This returns the variable "data", which is a matrix with dimensions (Number of Samples) x (Number of Electrodes).

**Note:** Alternatively, get voltage data from multiple channels in one step by first specifying an optional input "dimension" to LoadData. See the AxisFile help for more information.

## 4. ACCESSING STIMULATION EVENT TAGS IN MATLAB

For custom-analysis routines, the stimulation tag timestamps may be accessed using the provided Matlab functions installed with AxIS. First, use the following command in Matlab to load an AxIS ".raw" or ".spk" file:

```
FileData = AxisFile(fileName);
```

The tag information is automatically extracted from the data file chosen. To access the tag time stamps, use the following command:

```
evts = sort([FileData.StimulationEvents(:).EventTime]);
```



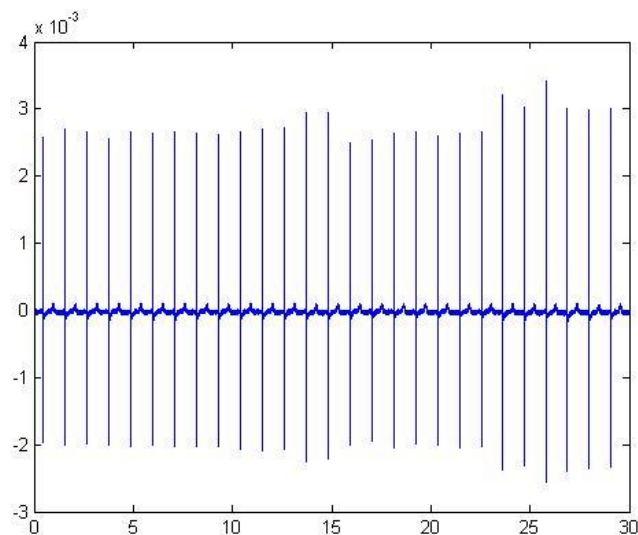
## 5. EXAMPLE APPLICATIONS

Matlab is useful for visualizing and analyzing data collected in AxIS. In this section, examples are provided describing how to generate basic plots of raw and spike data, as well as a more complex raster plot of neuronal firing.

### 5.1. PLOT RAW DATA

The following example demonstrates how to load and plot the first 30 seconds of the data from Well C2 Electrode 34 of the cardiac demo dataset. This sample dataset can be found at Libraries/Documents/Public Documents/Axion BioSystems/Manuals and Documentation/Sample Data and is called '12\_Well\_Cardiac\_Demo(000).raw', however any raw file may be used. Load and plot the data of interest using these commands:

```
Data=AxisFile('12_Well_Cardiac_Demo(000).raw').DataSets.LoadData...
    ('C2','34',[0 30]); %Load the data of interest
[t,v]=Data{3,2,3,4}.GetTimeVoltageVector; %Get time and voltage
    %vectors
plot(t,v); %Plot the data
```



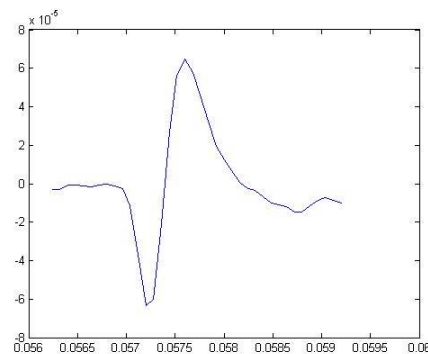
## 5.2. PLOT SPIKE DATA

The following example explains how to load and plot spikes that occurred on Well B1 Electrode 12 of the neural demo dataset. This raw sample neural dataset can be found at Libraries/Documents/Public Documents/Axion BioSystems/Manuals and Documentation/Sample Data and is called '48\_Well\_Neural\_Demo(000).raw'. Use AxIS to generate a .spk file from the raw file. Any other .spk file may also be used. First load all of the data in the file using the command:

```
AllData=AxisFile('NeuralDemoData_SpikeFile.spk').DataSets.LoadData;
```

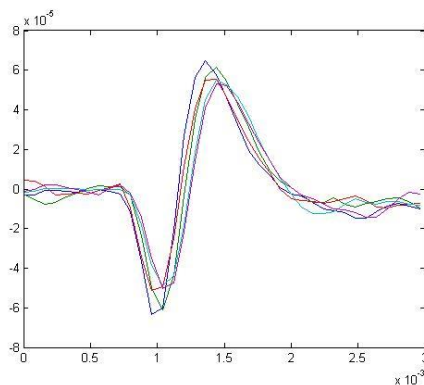
To plot the first spike detected on Well B8 Electrode 44, use the commands:

```
[t,v]=AllData{2,8,4,4}(1).GetTimeVoltageVector; %Get the time and
           %voltage vector for the first spike waveform only
plot(t,v); %Plot the data
```



To plot the first 5 spikes overlaid, use the commands:

```
spikes=AllData{2,8,4,4}(1:5).GetVoltageVector; %Get the voltage data
           %from the first 5 spikes
time=AllData{2,8,4,4}(1).GetTimeVector; %Get the time vector from the
           %first spike
plot_time=time-time(1)*ones(size(time)); %Shift the time vector back
           %so it starts from zero
plot(plot_time,spikes); %Plot the 5 spike waveforms overlaid
```



### 5.3. PLOT RASTER PLOT

Create a raster plot of the spike times for all electrodes in Well B1 of the neural demo dataset using the commands:

```
[nwr nwc nec ner] = size(AllData); %Find the dimensions of AllData
for i = 1:nec
    for j = 1:ner
        %Select the spike times from each electrode in the well
        if ~isempty(AllData{2,1,i,j})
            ts=[AllData{2,1,i,j}(:).Start];
            hold on;
            plot([ts;ts],[-(i-1)*ner-j+[0.75*ones(size(ts))...
                ;zeros(size(ts))],'k'); %Plot a vertical line at each
                %timepoint corresponding to a spike
            hold off;
        end
    end
end
```

