

RWTH Aachen University
Software Engineering Group

First line of title
Second line of title
Third line of title
Forth line of title

Bachelor Thesis/Master Thesis/Seminar Paper

presented by

Surname, Prenom

1st Examiner: Prof. Dr. B. Rumpe

2nd Examiner:

Advisor:

The present work was submitted to the Chair of Software Engineering

Aachen, January 12, 2022

Kurzfassung

Eine kurze Zusammenfassung der Arbeit.

Abstract

A short abstract of this thesis.

Contents

1	Introduction	1
2	Data	3
3	Background	5
3.1	Analysis tools	5
3.1.1	FieldTrip	5
3.1.2	openMNGlab	6
3.2	Data acquisition software	7
3.2.1	Spike2	7
3.2.2	Dapsys	8
3.2.3	OpenEphys	9
4	Software	11
4.0.1	Student 1	11
4.0.2	Student 2	12
4.0.3	Student 3	12
5	Conclusion	15
	References	17
A	z. B. Programmdokumentation	17

Chapter 1

Background

1.1 Analysis tools

When it comes to analysis software for microneurography data, there are multiple available that each have their own use-cases.

1.1.1 FieldTrip

Fieldtrip (fieldtriptoolbox.org) is a MATLAB toolbox developed at the Radboud University, Nijmegen, the Netherlands and offers a wide variety of analysis functions. It can analyse MEG, EEG, and iEEG and is an open-source software that has been in development since 2003. Its main strengths lie in the analysis of non-invasive and invasive electrophysiological data. It provides over 100 high-level and 800 low-level functions that can be used both by experimental users as well as developers. It does not feature a graphical user interface, but instead focuses on providing direct access to the high and low-level analysis functions that can be used in the command line or in scripts. This increases flexibility and is done because FieldTrip is not meant to be an application but a set of tools that can be mixed and matched for different requirements. With this approach the user needs to be somewhat familiar with MATLAB before starting their analysis, but after a potential initial time invest the flexibility of this approach yields great advantages. The analysis functions are meant to be combined and used as a sort of analysis protocol, where the output of functions is used to compute the next results. An example pipeline is lined out in the paper (put citation here). The first step after loading the data set of interest is to choose a data segment that is to be analysed by setting the boundaries of said segment. In order to free the data from noise and artifacts that could compromise the analysis. FieldTrip provides the corresponding functions for automatic removal of artifacts. As a next step the user can run a preprocessing function that loads the specified data into the MATLAB workspace and readies it for further analysis. Then the user can choose which specific analysis they want to perform. There is also the possibility of source reconstruction to get a visual representation of the source of the data.

An important feature of FieldTrip is the ability to save and reuse intermediate results. After each step in the analysis there is the possibility to save the current results for later use before further altering the data. This can be useful in many situations where the user wants to retrace certain steps in the analysis.

For visualization of results users can make use of standard MATLAB functionalities to plot the numeric results of FieldTrip analyses.

As an open-source software FieldTrip is also meant to be contributed to by developers who see opportunities for improvement. As opposed to other GUI-using software FieldTrips focus on having the direct access to functions lends itself to development and icontributions from various different experts, which only enhance the product.

This software package is very useful and widely used in the field, however it does not lend itself to our specific needs. The main problem with this software is its programming language. It is a MATLAB toolbox, however it would be preferred to use a software package in python or another programming language that slots better in the already existing structure that is used at the chair for medical informatics.

1.1.2 openMNGlab

The previously presented analysis tools could not be used for this thesis for different reasons. There are multiple data acquisition systems that are used at the chair of IMI that each produce different file formats.. The three main systems that need to be handled are Spike2, Dapsys and OpenEphys and are described in more detail in the next section.

We need a tool that that has the capability to load files from these different data acquisition softwares, put them in a compatible format and analyse them further. FieldTrip does not offer these kinds of capabilities as well as it being a MATLAB package, which would not be ideal for fitting into the rest of the analysis systems at IMI. Elephant is also lacking the importing tools that would be required for the software solution.

For this reason the Institute of medical informatics has started to develop their own software framework in python called openMNGlab. This framework aims to provide a solution for those issues with the different file types and combine them into a single usable format. In addition it provides analysis capabilities for microneurography data.

For the import of the different file types from the data acquisition packages and the internal structure of the data openMNGlab makes use of an already developed python module called Neo(neuralensemble.org/neo). This is a package that already deals with neurophysiological data in various different file formats. It is aimed towards combining these different sources while not providing any analysis capabilities of its own to reduce dependencies (cite website here). It also comes with templates to build importers for different new file formats should those be required and thus makes it easier to include more file formats at a later date.

OpenMNGlab has started as a project of the Institute of medical informatics and was initially developed by Fabian Schlebusch who developed openMNGlab 1.0 and setup the basic structure and developed the importers necessary for the file formats that we require. However, the 1.0 version of the framework is not enough in terms of analysis and also importing capabilities and needs to be improved.

At IMI there are people who work on varying aspects of MNG data analysis such as analyzing the response latency to electrical stimulation or analyzing the nerve activity in response to certain chemicals. I am focusing on analyzing spike trains occuring in mechanically and electrically stimulated nerve fibers.

With my bachelor thesis I aim to provide some more analysis functionalities for spike trains as well as give some ideas on the software engineering of the software framework in the future.

In the end FieldTrip and Elephant offer good solution for slightly different use-cases, but

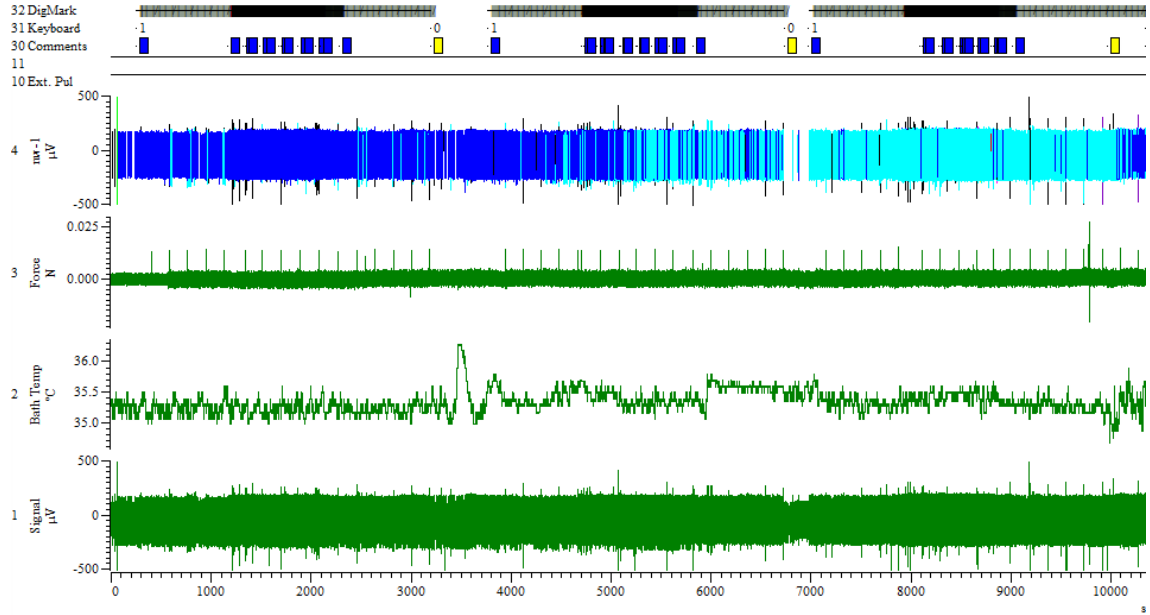


Figure 1.1: Typical mechanically and electrically stimulated recording in Spike2

were not a perfect fit for the analysis of microneurography data at the Institute of medical informatics. They are, however, a good inspiration for what should be included in such a software framework. In the case of Elephant, since it is also a python module and also makes use of the Neo structure, it could make for a good addition of more high-level analysis functionalities in the future.

1.2 Data acquisition software

A data acquisition system is a combination of software and hardware components that work together in order to control inputs towards and record data from different subjects. There are many different data acquisition systems for electrophysiological data. There are three systems that we are working with and that should be compatible with our chosen software analysis framework. These systems are Spike2, Dapsys and OpenEphys.

1.2.1 Spike2

Spike2 is a data acquisition and analysis software produced by Cambridge electronic design limited. It is a flexible tool that can be used in a variety of different ways.

The software can record multiple channels simultaneously. An example screenshot from a recording can be seen in Figure 3.1. This depicts a typical recording used for analysis in this bachelor thesis. The recording contains data from nerve fibers of rat cranial dura mater. The nerve fibers were stimulated using a mechanoelectrostimulator applying electrical and mechanical stimulation.

First of all it contains a channel for the recorded raw signal at the bottom. The next channel contains the temperature during the recording. In this example it fluctuates between 35°C and 36.5°C. In channel 3 we can observe the mechanical force that was

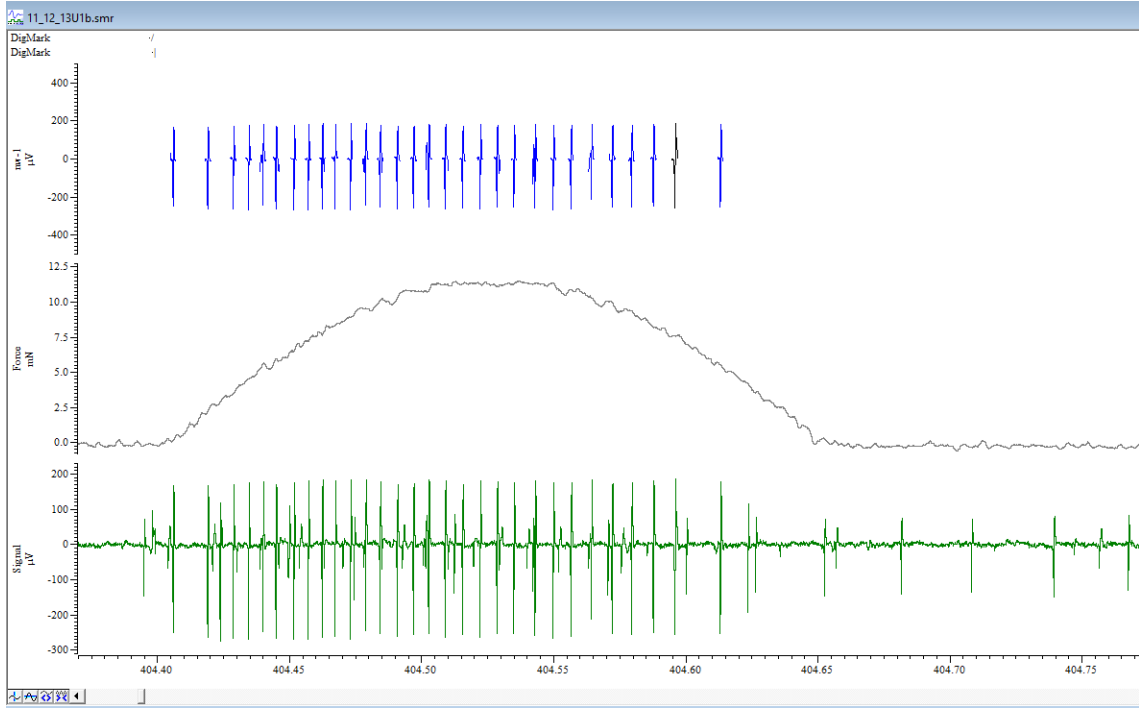


Figure 1.2: A single spike train in spike2

applied to the nerve fibers. In Figure 3.1 there are spikes in mechanical force whenever a mechanical stimulation occurs to evoke a spike train. For this experiment we want to collect the data of single nerve fibers. It is difficult, to record just a single nerve fiber in vitro, however. This is why in this experiment spike templates are applied to the raw signal to filter out specific fibers. These filtered fibers are then displayed in so called wavemark channels. In this example channel 4 is such a channel, where only specific action potentials are filtered. The filtering process is done by the experimenters and is based on certain features of the action potential shape.

The topmost channel in Figure 3.1 contains markers for the electrical and mechanical stimuli. Additionally there is a channel containing comments regarding the experiment. Comments can represent the experimental protocol and are filled in by the experimenters. In this example there are comments denoting a change electrical stimulation frequency. In other experiments for example, these could also denote the application of certain chemicals towards the recorded subject.

A more detailed view of a single spike train can be seen in Figure 3.2. Here the difference in electrical and mechanical event markers in the topmost channel can be seen. Mechanical markers are represented by a slash, while electrical markers are represented by a vertical line. Another thing that can be seen here is the channel containing only the spikes. This channel is ideal for the extraction of the spikes for later analysis as there is no noise in the channel anymore and the spikes can also be interpreted as simple events with a timestamp.

1.2.2 Dapsys

The second data acquisition package that openMNGlab needs to support is called Dapsys. It is a hardware and software system that can record and analyse electrophysiological

data from animal or human sources and has been mainly used for studying the peripheral nervous system (cite website here). It has been in development for over 30 years since its earliest version and thus has a great history of usage in the field. The idea behind the conception was to build a system that could control stimulators and simultaneously acquire the data in real-time and display the data (cite the article).

Dapsys offers the capabilities of path tracking and comes with the benefit of much data being available from experiments conducted with the Dapsys software. It is used especially for electrophysiological recordings with human patients. As is the case for Spike2 it also comes with a visual representation of the data which can be seen in Figure (put ref to screenshot here). This graphical interface works real time while recording the data.

The version of Dapsys used in the experiments from Barbara Namer is specialized for microneurography and was configured in cooperation with Brian Turnquist.

OpenMNGlab already features an importer for Dapsys recordings, which makes many experiments conducted with this system readily available for analysis. The difference to Spike2 however, is that the importer does need an extra step from the user, since it does not import the raw Dapsys files. We need to export the raw data and templates for the tracks we want to analyse as csv files from the Dapsys software, before we can import them into openMNGlab. This process is especially cumbersome when dealing with a larger number of recordings. In the future the importer functionality of openMNGlab should be improved, so that it works on the raw Dapsys files in order to make the analysis workflow easier.

1.2.3 OpenEphys

The last data acquisition system I want to focus on is called OpenEphys. It is an open-source electrophysiology data acquisition system originally developed by a nonprofit based in Cambridge, Massachusetts. It sets a big focus on modularity and flexibility so that it can fit many different needs from a variety of users.

The idea behind OpenEphys came after an increased popularity of closed-loop experiments in neuroscience, in which the results of the recorded system has an influence on the system itself. With proprietary systems it is somewhat difficult to share the details of such experiments and to replicate them. The introduction of OpenEphys, an open-source system is supposed to make it easier to develop and share analysis and details of such closed-loop experiments.

OpenEphys makes use of inexpensive open-source hardware such as Intan chips to make it easier for small labs to get started with analysing electrophysiological data. The heart of the software part of the system is a plugin-based graphical user interface (GUI). Here the user can add different modules to the processing pipeline such as modules for communicating with the hardware or modules for analysing the data.

Different labs often have widely varying sets of requirements towards data analysis and therefore data acquisition systems. At the start of a project they are confronted with the question of whether to use a proprietary system or develop their own systems, which will result in much more effort before the real work can begin. Proprietary systems also often come with very limited customization possibilities, which makes it harder to get the system that fits the users needs exactly. This is where OpenEphys comes in with its plugin-based software. It is designed so that everyone can mix and match the processing modules and

end up with the exact analysis pipeline that they require. In addition to a wide array of already available processing modules, it is possible to develop your own modules that fit seamlessly into the structure of the system. OpenEphys is completely developed in C++ and based on a library for audio applications. This makes it easy to develop new modules by making use of the class inheritance capabilities of C++.

OpenEphys comes with many advantages such as low cost, transparency and flexibility due to the modularity, but there are also drawbacks to using the system. The start is also a little harder than in proprietary systems, where there is often a straight forward way of getting started with your first experiments. The modular nature of the system might also effect the performance of the analysis, which is why commercial systems might be a better choice, if they fit the analysis needs.

In this thesis we are considering OpenEphys because it is used by a collaborating research team in Bristol, which also wants to make use of openMNGlab.

Chapter 2

Software

- Use-cases:
 - opening data (importing)
 - latency study (Alina)
 - chemical data study (Jessica)
 - mechanically evoked spike trains (Alexander)
 - experimental researchers (Barbara)
- results in a list of requirements
- Use-cases lead to software engineering approach
- necessary steps for my analysis
- how I implemented it

To begin this section I want to present a couple different use-cases of various users of openMNGlab. I will describe the specifics of each use-case and then give a few remarks on the current state of openMNGlab and the potential improvements necessary for a smoother experience for all users.

One important feature of openMNGlab that is needed in every use-case is the importing of new data. For new users it is important to be able to load sample data sets and get a feel for the framework by getting to know a few test analysis functions. For students or experimental researchers it is key to be able to load multiple files for a quick overview or even full analysis.

Because we are dealing with a couple of different sources when it comes to experimental data, which all need to be handled in a different way, this is where many issues can occur. When going through the use-cases we will see where already we encounter some difficulties regarding the data import.

2.0.1 Student 1

The first user is a student who does data analysis on mechanically and electrically stimulated Spike2 data. The goal is to perform latency analysis for spikes. The raw spike2 files feature a lot of information, not all of which is always needed. In this case, all that is required to analyse the latencies of the spikes are the timestamps of the spikes itself and the timestamps of the stimulation events. For the Spike2 software this means that we need to extract the DigMark channel which contains the event information as well as the corresponding wavemark channel which contains the information on the already presorted

spikes.

The relevant information can be imported using the importing function of openMNGlab. The data is then used to calculate the latency for the spikes corresponding to the latest electrical stimulus event as well as calculating the spike count.

Potential problems: The student started the analysis before the framework was working properly and therefore uses their own custom analysis notebook. However, all the information that is required is also available with the regular Spike2 importer in the current openMNGlab version. In recordings with both electrical and mechanical stimuli, the event markers for those stimuli share the same channel in Spike2. Although they can be distinguished in Spike2 itself, the imported channel in the Neo format does not distinguish between different event types. This needs to be addressed if we want to work with recordings that feature multiple types of stimulation. Additionally there might be some general import difficulties which I will elaborate on later in this section.

2.0.2 Student 2

The second user is also a student who uses mechanically and electrically stimulated Spike2 data. Their goal is to analyse spike trains resulting from mechanical stimulation. For this they do not need all of the information contained in the raw Spike2 file. They need the event information for mechanical and electrical stimuli as well as the mechanical force information for details of the mechanical stimuli. The event information can be found in the DigMark channel of the Spike2 file and gets extracted by the Spike2 importer of openMNGlab. The mechanical force is a continual signal but appears in the form of sin shapes when the nerve is stimulated. Just as the first user they also need the information regarding the spikes themselves, which are also imported by the framework.

From the gathered data the user then calculates various quantifiers for the mechanically induced spike trains such as spike count or more elaborate features. The user started this project while still using the old framework version where the channels from the Spike2 software were manually extracted to a csv file. With the updates to the framework in the meantime not all of the required information gets extracted as easily as before. This is why they use a hybrid version of the two versions of the framework where they mix and match the functions as they need them.

Potential problems: There are similar potential problems when it comes to the event channels and regular spike channels as in the case of the first user. Additionally there are problems when it comes to the import of the mechanical force. The user started the analysis with an old version of the framework which used other importers (csv files). With this technique they could extract the mechanical force throughout the recordings. The new Neo importer still has some bugs when it comes to importing the mechanical force from the Spike2 file, which requires fixing for future users who need this information.

2.0.3 Student 3

The third student user of the framework works on a project to analyse Spike2 data in which certain chemicals were applied to the test subject. In contrast to the other two users, this user started with the project when the current version of openMNGlab with the integration of the Neo module already in place. This means they only made use of the new importing functions and are not stuck with certain parts from the old framework. In

addition to the spiking activity, as in the previous use-cases, they also need information regarding the chemicals that are applied. They need the timings and doses of the specific chemicals. From this they need to specify a time frame where they want to monitor the spiking activity that results from the application of the chemicals.

Potential problems: The analysis of chemically stimulated data does bring with it its own new set of problems. There is no dedicated channel for chemical information in the Spike2 software. For this reason the chemical protocols are given in the general comment channel in Spike2. That means that the experimenter manually creates a comment everytime a new chemical is applied. The issue with this is that openMNGlab does not import the comment channel. In practice the comments do not have a strict naming scheme, which would also lead to difficulties with the automatic detection of specifics regarding the chemicals. This is a problem however that does not stem from the framework, but is something that has to be addressed on the side of the experimetal researchers.

These described problems lead to the current workflow of manually selecting a suitable timeframe for interesting spiking activity some time after the chemical applicaton; some times up to a minute (check the times again) after the chemicals where applied.

Chapter 3

Conclusion



Figure 3.1: Das SE Logo

Appendix A

z. B. Programmdokumentation

