1. Special safety issues
   1. Algorithm’s area of use

Fetal monitoring is a part of health assessment methods for biological system, containing mother and her baby. The method described here receive a bit of information in the whole feature dimension. Fetal electrocardiogram as well as methods which include signals obtained from electrodes evaluate the work of organs and system in general.

Fetal electrocardiography is a promising alternative to cardiotocography continuous fetal monitoring. Robust extraction of the fetal signal from the abdominal mixture of maternal and fetal electrocardiograms presents the greatest challenge to effective fECG monitoring.

Fetal heart rate monitoring in its early form was based on the auscultation methods, i.e. intermittent observations of the fetal heart sounds. Progress in electronics and computers science brought to the introduction of the first fetal monitors based on phonocardiography in the middle of the 20th century. Yet these inventions were still challenged by the need to automatically distinguish between the maternal and fetal heart sounds. Consequently, in 1953, the first attempt was made to continuously monitor fetal heart rate by means of non-invasive fetal electrocardiography [1].

To the time, the number of successful attempts in fetal electrocardiography, fetal monitors is in the tens. Overall, they can be divided in two categories:

1. Stationary fetal monitors (ex. Meridian Mindchild,)
2. Mobile fetal monitors (ex. Monica AN24)

Main difference of categories in their number of features to be extracted, their quality and comfort of use. The idea of their use presented in figure 4.1. It is important to notice the size of calculation modules.

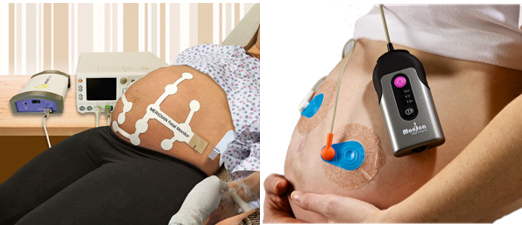


Figure 4.1 – Fetal monitors a) Meridian Mindchild stationary monitor

b) Monica AN24 Mobile monitoring system

While those are ready to use devices, my algorithm is intended to use in computer systems first of all. However, there are a lot of common parts, methods which can be used both in mobile, stationary or not lifetime systems.

For example, filter methods, including wavelet baseline removal, are ready to use in practically most of systems with little limitations. Fetal heart rate analysis, which is described in 2.3 intended mostly on telemedicine fetal monitors, while still can be used on non-lifetime data.

* 1. Software safety

Software to be developed do not complete any safety functions and activities which can directly hurt operator or patient. However, the field of biotechnology and medical science tightly communicated with human health.

Fetal heart rate analysis algorithm as well as other diagnosis tools perform assessment tasks and outputs biomedical indicators and decisions. Algorithm presented in current master’s thesis may harm people with imprecise data in case of unforeseen situations, which are mostly depend on the data acquired.

1. Complex high electrical interferences.
2. Continues artifacts with morphology similar to ECG
3. Non-compliant data acquisition

Algorithm as the list of instructions cannot misbehave, hence, difficult noises and problems described above should be eliminated in future releases.

* 1. Software ergonomics

Software ergonomics is a subcategory of ergonomics that concerns the software design, rather than the hardware design, of systems. Software ergonomics includes the determination of user needs, interface design, user support and usability testing.

Software-ergonomics standards contain guidance which assists both the specification of user requirements and the design and evaluation of the user interface of an interactive system. These standards do not aim at standardizing the user interface; rather, they give recommendations that should be applied in order to ensure the usability of the user interface of the product and eliminate design solutions which can be predicted to cause usability problems to users [2].

There is a list of fields in standards, which describes the most of features connected with software ergonomics

1. General guidance on software ergonomics (ISO 9241-110 to ISO 9241-119);
2. input, output and interaction (ISO 9241-120 to ISO 9241-129);
3. performance support (ISO 9241-130 to ISO 9241-139);
4. interaction techniques (ISO 9241-140 to ISO 9241-149);
5. topic-specific guidance (ISO 9241-150 to ISO 9241-159);
6. interface control components (ISO 9241-160 to ISO 9241-169);
7. cross-topic guidance on accessibility (ISO 9241-170 to ISO 9241-179).

Fetal heart rate algorithm analysis is mostly hand written and code is for personal use, but some parts could be extended and used by other people. Thus, there is a sense to describe ergonomic part of human-software communication in the way of updates and research.

In addition, the absence of a graphical user interface makes unnecessary item 3, performance support. Which describes element and field highlighting and other features for intuitive program use. Interface control components as well as graphical interface is completed in algorithm is optional and not realized, thus, items 5 and 6 do not relate to the project done.

* + 1. Dialogue principles

Seven principles have been identified as being important for the design and evaluation of interactive systems, which serve as a set of general goals for the design and evaluation of dialogues [3]:

1. Suitability for the task;
2. self-descriptiveness;
3. conformity with user expectations;
4. suitability for learning;
5. controllability;
6. error tolerance;
7. suitability for individualization.

Algorithm is presented as a number of intuitive functions gathered together in the order of doing the task step by step. However, there is no graphical user interface for people to easy follow these steps. The majority of functions are presented in the way of providing user comfort, default parameters are set, all function are called with clear understandable names.

In addition, items 4 and 5 are mostly depends on the development environment and still cannot be achieved by code perfection. While the program is intended for qualified people, for example, in programming language it is written in, items 6 and 7 may be performed.

* + 1. Individualization

Individualization is used in a wide variety of ways to enhance applications both for users. The wide variety of different implementations includes many instances where individualization creates considerable challenges for the users that it ought to be helping. This becomes an even greater challenge when users have to deal with different individualization approaches in each of the several applications that they use [4].

Individualization is a very delicate question. Algorithm for fetal extraction and analysis like a constructor, where signal flows through the independent blocks. To this time, researched can adjust block parameters, rework some of them or even exclude from the algorithm. However, huge changes should be performed with rules. First of all, dependencies must be fit and not violates in any way. Signal flow are not to be changed, because of algorithm interruption.

From the other hand, human factor may impact on the code changes badly, because of misunderstanding that is shown in [4] chapter 6.3. For example, a number of nonsense commands might lead to increase of computational cost and accessibility to other functions.

* + 1. Software accessibility

It is important to incorporate accessibility goals and features into the design as early as possible, when it is relatively inexpensive compared to the cost of modifying products to make them accessible once they have been designed.

The majority of graphs are used with *plotly* framework. Plotly is a technical computing company headquartered in Montreal, Quebec, that develops online data analytics and visualization tools. Plotly provides online graphing, analytics, and statistics tools for individuals and collaboration, as well as scientific graphing libraries for Python, R, MATLAB, Perl, Julia [5].

Despite they provide graphical user interface framework called *Dash,* current project was done only with offline version of graphs that were plot [6]. User interface presented in the figure 4.2.

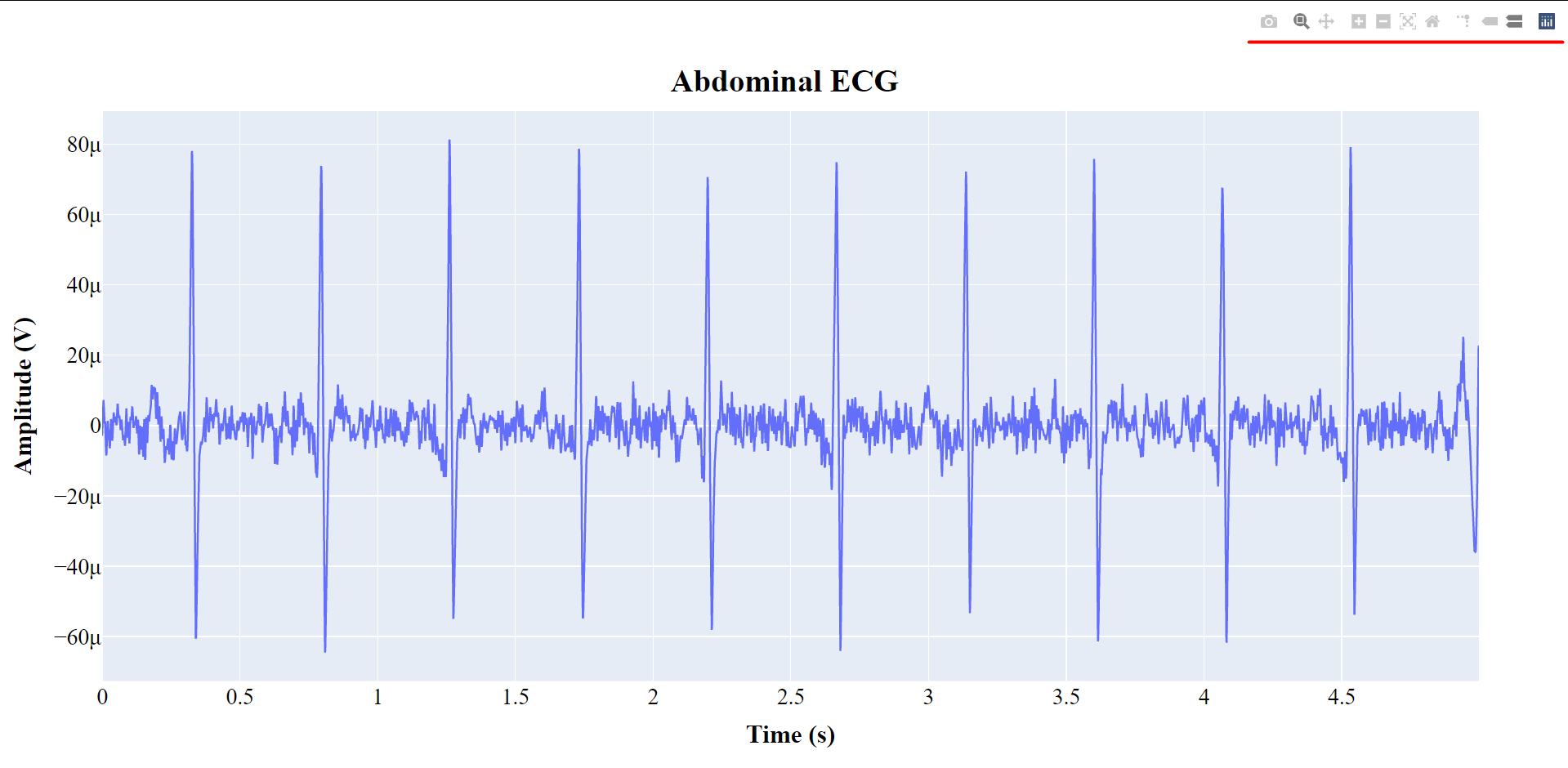


Figure 4.2 – Graph representation with plotly

Framework provide base intuitive buttons in the right top angle. There is a possibility to make screenshot, zoom the data and go through the plot with an ability to come back to the initial settings.

Standard ISO 9241-171 is based on the current understanding of the characteristics of individuals who have particular physical, sensory or cognitive impairments. However, accessibility is an issue that affects many groups of people. The intended users of interactive systems are consumers or professionals – people at home, at school, engineers, clerks, salespersons, Web designers, and so on. The individuals in such target groups vary significantly as regards physical, sensory and cognitive abilities and each target group will include people with different abilities.

Thus, people with disabilities do not form a specific group that can be separated out and then disregarded. The differences in capabilities can arise from a variety of factors that serve to limit the capability to engage in the activities of daily living, and are a “universal human experience”. Therefore, accessibility addresses a widely defined group of users including [7].

Plotly interface satisfies most of items in standard. Names are done with idea of being short, meaningful and uniqueness in the separated field they are placed. The whole name rules are presented in list below:

* Each element has a name.
* Names are unique in the field of use.
* Names have clear meaning.
* Names should be short.
* Names are available to assistive technologies.
* Names have proper position related to other elements.

For the graph display plotly interface provide different abilities to work with diagrams, they are presented in a list below:

* Support *Copy* (save) operation.
* Support *Undo* operation.
* Support elements to move the graph.

To conclude, plotly framework provide standardized way to draw graphs and diagrams with a clear ability to save, move and zoom operations. Despite figure elements are done by default, graph elements can be adjusted only with program code. So, there is no graph inspector to change title, text style and so on in the completed figure, that can lead to some difficulties.

* + 1. Research comfort

There are lots for programming language that are known but all of them need to follow some strategy when they are implemented and this methodology/strategy is paradigms. Apart from varieties of programming language there are lots of paradigms to fulfil each and every demand. In general programming paradigms can be divided in two groups:

* Imperative.
* Declarative.

Main difference is in the way of performing the task. Imperative paradigm works by changing the program state through assignment statements. It performs step by step task by changing state. The main focus is on how to achieve the goal. The paradigm consists of several statements and after execution of all the result is stored [8].

While declarative paradigm focuses on what needs to be done rather how it should be done basically emphasize on what code is actually doing. It just declares the result we want rather how it has been produced.

Fetal heart rate algorithm has been done with imperative programming paradigm, or rather object-oriented programming. Actually, if I had a chance to make if from scratch I would build well defined class-method structure. However, there are no dedicated classes, just functions which usually take the common parameters:

* Sampling frequency
* Data

Right representation should contain methods which take these data as attributes of class and perform filters, processing algorithms, etc. So, it leads to disadvantage in code structure, makes it more complex and disorganized.

Another drawback of an algorithm as a research tool is the absence of graphical user interface or availability of pipeline implantation. There are a lot of parameters to change in order to influence on the signal in algorithm flow. For example, filters can have following parameters:

* Type (realization).
* Order.
* Frequency range (or single value for low/high filtration).

Graphical user interface could provide online change of parameters without relaunching the whole algorithm. Thus, computation speed for one test is much higher for the cost of a small amount of building time.

More complex processing and analysis parts would have the same amount of performing time. However, there is one significant interface that requires a lot of time compared with common methods. Graph building on the base of *plotly* framework requires two objects to show the outcome:

* Figure.
* Graph.

Figure is the place where graphs are located, it has the number of attributes like title text, alignment, styling elements and so on. Graph is the object that includes data and description within. Overall computational cost is distributed in these objects in the proportions of about 80 and 20%. Thus, predefined figure object with graphical user interface would decrease the amount of time to be used for research.

From the other hand, current program code contains several required comfort improvements. First of all, program is separated in four modules:

* Main.
* Dataset extraction.
* Processing.
* Analysis.

Strict structure sets the terms of use, for example, *analysis* can inherit variables and functions from processing, but not in another way. *Main* module inherits all other modules, while, *dataset extraction* is done only for data conversion in the form the whole algorithm performs. User module dependencies are shown in figure 4.3.

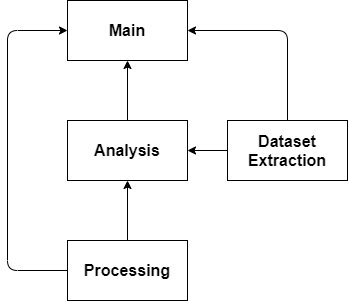


Figure 4.3 – User module dependencies

Second advantage is hidden in the input of the functions, despite the absence of right object-oriented paradigm, all of functions have predefined default values. For example, there is no need to set all parameters to make the project working, moreover, some values like *sampling frequency* are inherited from processing block and can be changed for all functions by default. This approach provides making functions independent for other applications with comfortable performance within project.

* 1. Conclusion

The project to be evaluated in terms of safety and ergonomics consists of two parts:

* Programming code
* Graph interface

Graphs and figures are only the part of chosen framework, thus, most of tools they provide couldn’t be changed or individualized. However, it satisfies standards of naming and alignment for comfortable use with clear element understanding.

Program code is mostly user developed, thus, it consists of a number of drawbacks in structure (complexity) as well as in repeatable research activity. However, there are a bunch of features done for convenience in the built structure. For example, some disadvantages from the absence of classes is neutralized with global variables and module inheritance.