MSEL
Modelling in Science Education and Learning

Modelling in Science Education and Learning
Volume 15 (1), 2022 doi: 10.4995/msel.2022.16224.
Instituto Universitario de Matemática Pura y Aplicada
Universitat Politècnica de València

Choreographies for musical fountains
Coreografías para fuentes musicales

Martin Bracke, Patrick Capraro
Technische Universität Kaiserslautern. Germany
bracke@mathematik.uni-kl.de, capraro@mathematik.uni-kl.de

Abstract

This paper describes different possibilities to answer the question of automatically creating a choreography for musical fountains using mathematical modeling. The possibilities for an implementation range from a short project in the middle school, which has large parts in mathematics, to long-term projects, which deal with the physical background, an implementation with the help of the computer and in the highest expansion stage even a technical realization. In the article different variants are described, supplemented by experiences from practice and extension possibilities are pointed out. All in all, the readers should be enabled to design and practically implement their own project according to the individual local conditions such as time frame, learning level of the target group and possibilities for interdisciplinary cooperation. (This article is a translation and update of the German first edition Bracke, M. & Capraro, P. (2021)).

Este artículo describe diferentes posibilidades para responder a la cuestión de la creación automática de una coreografía para fuentes musicales utilizando el modelado matemático. Las posibilidades de una implementación van desde un proyecto corto en la escuela media, que tiene grandes partes en matemáticas, hasta proyectos a largo plazo, que tratan el fondo físico, una implementación con la ayuda del ordenador y en la etapa de mayor expansión incluso una realización técnica. En el artículo se describen diferentes variantes, complementadas con experiencias de la práctica y se señalan las posibilidades de ampliación. En definitiva, los lectores deberían estar capacitados para diseñar y poner en práctica su propio proyecto de acuerdo con las condiciones locales individuales, tales como el marco de tiempo, el nivel de aprendizaje del grupo objetivo y las posibilidades de cooperación interdisciplinaria. (Este artículo es una traducción y actualización de la primera edición en alemán de Bracke, M. & Capraro, P. (2021))

Keywords: Mathematical modelling, mathematics education, mathematical concepts, tools and processes, school practice, German speaking.
Palabras clave: Modelización matemática, educación matemática, conceptos, herramientas y procesos matemáticos, práctica escolar, habla alemana.

1. Introduction

The project described in this paper was prompted by the following situation: In Seattle, the *International Fountain*, a fountain designed by Japanese architects Kazuyuki Matsushita and Hideki Shimizu, has existed since the World's Fair in 1962 (see Fig. 1). This fountain has a large number of individual jets that can be controlled individually or in small groups. A special feature of this fountain is that, to match different pieces of music, the water jets are controlled according to a choreography developed for the music in question. The audience is offered an impressive spectacle in which the emission and course of the water jets are *in unison with the music*.[1]

Figure 1 – International Foutain in Seattle (USA) (©Joe Mabel 2008, https://creativecommons.org/licenses/by-sa/3.0/legalcodeCC-BY-SA 3.0 license)

However, the respective control of the water jets for each individual piece of music is developed by hand in a very elaborate manner and adjusted until the desired choreography is achieved. Because of the elaborate process, there are only a few pieces of music that have been converted into a choreography specifically for this fountain.

A similar concept can be found elsewhere, for example in Hamburg, where there are regular water light concerts in the park *Planten un Blomen*, where an elaborately arranged play of water and light to matching music can be admired. There, two artists control or play a light organ or the water play, respectively, matching the music played in parallel.[2] If there are such concerts in combination with light and water plays in your area, this would be the ideal motivation and first source of information for the project described in the further course – but don't worry: It works without! [3].

Taking a closer look at such events, we can ask ourselves the following question: Can the high effort of creating a "fitting ", arrangement of water and/or light play to given music be simplified with the support of a computer? Or can we even create a beautiful choreography

[1] cf. e.g. https://youtu.be/QneeNTuQAWQorhttps://youtu.be/AXIBOUrp1wo
[2] cf. e.g. https://youtu.be/muqhQfEAdd0
[3] The original idea, from which some of the modeling projects carried out by the authors have emerged, came from Christof Wiedemair, who posed and supervised a problem on this topic at the Mathematical Modeling Week 2018 in Tramin/South Tyrol

completely automatically with the help of the computer? And what do these questions have to do with mathematical modeling?

In answering them, we will proceed step by step, approaching answers in different variations throughout this paper at different levels of complexity – mostly directly related to the sophistication of the mathematical concepts and tools used. In doing so, we will outline how a corresponding modeling project can be planned and implemented in mathematics classes – very much in interdisciplinary cooperation. Besides the target group, the available time frame and the intended focus on certain aspects of the project will be important. On the basis of suitable mathematical modeling, different focal points can be set, whereby the school subjects involved are in principle mathematics, physics, computer science and music. Ideally, even all these subjects are linked in a common project. Therefore, we offer ideas for implementations in many places, which allow interesting results and insights even without special prior knowledge or (time-) consuming work.

In the next section we make a mathematical classification of the initial question and collect some remarks on the didactical relevance. Subsequently, a basic idea for a mathematical model is presented, which can serve to answer the previously formulated central question. On this basis it will be possible to identify the connection to the addressed subjects and to motivate the need for a set of concepts and tools. We present the gradation in the level of detail of modeling and content implementation that is possible in our view in a tabular overview. In section 4 we describe a very strongly simplified model, which can already be implemented in the intermediate level. With this we want to illustrate that even with a small time budget and very little prior knowledge in mathematics as well as the other fields involved (music, physics, computer science) one can approach this topic in an interesting way. To do this, we use a *black-box* approach as often as possible, hiding complex content for the learners so that they can focus on essential aspects of the model and simple mathematical background.

Because initially a lot of simplifying assumptions were made and partial solutions are simply used in the form of black-box models, there is of course a lot of room for individual refinement and specialization. In section 5 we exemplarily present a number of such refined models which have been realized in the context of different implementations with students. In addition, we also take a look at the tools needed in each case from mathematics and computer science. This should provide a good overview of the diverse possibilities of this project and serve as a stimulus for own implementations in a school context. In section 6 we have compiled mathematical basics of acoustics in a compact form, which can be very helpful for a more in-depth treatment or supervision in the context of an upper school project. This is followed by a short section on the challenges posed by students' misconceptions about the performance of computers in the context of problem solving, as well as some suggestions for the technical implementation of the results obtained – less elaborate than with the original fountain, of course – for example, with a home-made light organ. All in all, the readers of this article have the choice of a whole range of own project implementations: It can be worked on with moderate time requirements already in the middle school, while at the other end certainly also a seminar course in the upper secondary school can easily be supplied with enough input. We would like to emphasize at this point that the project ideas described in this article are not intended to teach very specific (mathematically complex or demanding) content. Rather, the goal is to solve a complex user problem using mathematical modeling methods. The complexity of the mathematical tools used is not fixed – and it is not a criterion for the quality of a project implementation either! – but can be chosen by the implementing teacher and adapted to the existing competencies of the own learning group as well as organizational requirements (cf. also Bock, W. & Bracke, M. (2015)).

2. Mathematical classification and remarks on didactical relevance

Before we get into the concrete description of the presented project, we would like to do two things: First, we want to give a very brief classification of the problem from a research perspective, and second, we want to show the didactical relevance that the implementation of a project in the proposed thematic environment offers.

From a mathematical point of view, there are interesting and direct links to active research areas, which we summarize in bullet points using two prominent examples:

- **analysis and processing of signals:** The analysis of audio signals, for example by the very common *Fourier analysis*, is mathematically well described and has been known for a long time. Nevertheless, the fast processing required in our particular case – possibly even in real time, see below – is a special challenge, which is addressed in current research especially with respect to the technical implementation in the form of digital signal processors.

- **Automatic note-based processing of music in audio form:** A fairly new area of research is concerned with how complex audio signals – such as a performance by a classical symphony orchestra or a modern pop song – can be automatically divided into individual tracks that can be meaningfully separated from one another and then represented by a staff-based score. The desire for such a transformation stems from the fact that based on actually occurring notes (essentially represented by pitch, tone length, and volume), it is much easier to perform certain manipulations of an audio signal. After an appropriate manipulation, a back transformation into an appropriately modified audio signal often takes place again. A very prominent application is the so-called *Auto Tune* for vocals, which ideally can be computed and performed in real time so that it is not noticed by the listener. A good overview of current issues and applications can be found in the article by E. Benetos et al. (2019).

In addition, if we aim at a comprehensive solution to the original problem, there is another active research area that falls in the border area between mathematics and computer science – it can be well titled *Scientific Computing*. Even if we have an idea from the two previously mentioned areas of how to generate appropriate responses of a musical fountain from music in the form of audio files or live (analog/digital) signals that create the feeling of "unison of music and motion", in the viewer, the appropriate computations to control a fountain must be done live, i.e. in real time. In addition to the possibility of calculating the corresponding control offline in advance for known music, it would of course be spectacular if a suitable reaction of the fountain could also be calculated and technically implemented in real time for music played live. Since a latency between the music and the visual events can be disturbing from 10 milliseconds, the corresponding calculations must run extremely fast, which requires the development of special algorithms.

Furthermore, the mathematical modeling of the initial question including the corresponding simulation with realistic representation and in real time is in itself a task that poses a real challenge even to professional representatives of mathematical modeling.

Regarding the didactic relevance of the proposed topic, we do not aim at a comprehensive analysis here in any way – such an analysis would easily take the dimension of a university thesis due to the possibilities of variation in content as well as the adaptability to a very wide range of target groups – which is proclaimed at least by us. Therefore, we would like to focus on what we consider to be some important aspects. First and foremost, the implementation we propose is intended as project work integrated into the classroom with a strong action

orientation, which in itself has a high value (Herbert Gudjons (1986)). An additional addressed competence is that of mathematical modeling, which has its fixed place in all curricula for the different school types. Without effort, interdisciplinary teaching succeeds, and the corresponding demand for the treatment of interdisciplinary STEM content and projects has been the subject of public and professional discussion for some time. In this context, a special variety of cooperation possibilities is given in this project, ranging from mathematics – which we, as mathematicians, naturally see in a central and organizing role – to music, physics, computer science and technology.

The possible mathematical content includes such elementary things as the collection, analysis and presentation of data as well as the topic of meaningful accuracy and approximation in the representation of data by numbers. Statistical concepts, ranging from mean to measurement error to more complex measures such as standard deviation or variance can be included. Also the use of digital media, something for audio analysis or representation and processing of self-generated data is a profitable facet of an implementation. Due to the underlying highly complex scientific concepts already mentioned above, an extension of the level of difficulty for higher grade levels and longer project duration is almost arbitrarily possible. Finally, in our experience, allowing students to generate their own data and be creative in doing so provides an opportunity for lasting memory and learning – not to mention the corresponding motivation in working through the project, which can be co-created by the learners. This observation is underlined by recent subject didactic research, in which a combination of inquiry-based learning followed by brief and adaptive instruction suggests an advantage over classical instructional methods in terms of sustainability of the learned content (cf. Harald Lesch (2020)).

3. Considerations for a basic modeling

In this section we would like to propose the cornerstones of a basic modeling, which on the one hand considers references to other subjects and on the other hand can serve as a basis for different implementations of the project, adapted to the target group and previous knowledge. As with very many mathematical models, there is no *correct* or *the* model, but we have many freedoms. Before making appropriate assumptions, therefore, it is useful to think about what we are trying to achieve: We cannot realistically expect a solution in a school project in which an arbitrary piece of music, e.g., in the form of an MP3 file is processed by the computer without further specification of the user. Or that a set of instructions is generated that controls the Seattle fountain mentioned in the introduction in such a way that the resulting choreography matches the music as desired. This realization may disappoint students at first, but finding reasons for this assertion together is, in our view, well worthwhile if you can devote the appropriate time. Why can't the goal be realized? There are at least the following partial aspects, which are almost certainly beyond the scope of a usual student project:

1. **translation music → appropriate formal description:** The choice of music track will not be able to be arbitrary, because alone the automatic translation of a digital audio file – for example, for a classical orchestral work – into the corresponding score is currently still a subject of research (cf. section 2). Even an average pop song with three to ten different tracks (vocals, instruments, percussion) overwhelms most fully automated (affordable) solutions.

2. **extraction of musical features:** From a complex score, even when available in a suitable digital format (i.e., ignoring all the problems in step 1), we will not be able to extract as comprehensive a set of meaningful musical features as possible without a sound knowledge

of music theory or a lot of practical experience. Furthermore, it must be investigated and guaranteed that a suitable implementation in terms of an action of the fountain is possible for the selected musical features.

3. **action of the fountain:** A complete implementation of a modeling project would include that at the end either the real fountain is controlled in a way specified by the model to match the selected music, or that its reaction is visualized by a simulation sufficiently close to reality. The real fountain is normally not available and for a simulation the actions that can be executed by the fountain must first be defined on the basis of the technically feasible possibilities – this will hardly be possible for the real fountain with almost 300 nozzles and selectable nozzle groups with the number of free parameters (time-dependent water pressure for each nozzle/nozzle group, thereby limitation by the maximum speed of changes) in the intended student project. In a modeling seminar at the University of Kaiserslautern, a seminar group in the master's program implemented a simulation using the scientific-technical software environment MATLAB to visualize water jets (bundles) from a total of 32 nozzles of the fountain as a three-dimensional animation. The physical model used was the inclined throw considering the air resistance. Even with the assumed simplifications and a small number of nozzles, such a realization is probably only feasible with the inclusion of year-long projects or youth research projects and is therefore not suitable for implementation in project or regular lessons at a school.

Even though it is already clear that a fully comprehensive solution is not achievable in a student project, the discussion of the aspects addressed can be very beneficial to better understand one's own limitations and technical challenges. Looking at the examples of existing choreographies of the fountains in Seattle and Hamburg linked in the introduction, one definitely gets a positive impulse as well: it can be seen that even in these professional implementations there is still a lot of room for improvement and that one's own results can even be really good in this light!

We propose the following simplifying assumptions for our basic model, referring to the sub-aspects discussed earlier:

1. **translation music → appropriate formal description:** The central question is which aspects of music we want to or can consider. In the simplest case, we can work with rhythmic audio signals, where we only consider the time of occurrence. We can easily generate such signals ourselves by tapping a rhythm on the table with our hand and recording this with a smartphone, for example. In the next section, we will show how we can arrive at an abstract mathematical description from such a recording, which can be done either completely manually or with the help of the freely available audio software *Audacity*. In more complex variants of the model, the frequency can also be added for rhythms. Another generalization would be to work with monophonic, simple melodies, and upwards there are no limits to the complexity here.

2. **extraction of musical features:** Once we have an abstract description of our "music", (which can also be a simple rhythm, see 1.), the next question is what features the fountain should respond to. One possibility would be to assign a reaction of the fountain to each event (i.e. rhythm beat) synchronously in time – of course, this depends on how we determine the available actions in the next step. As an extension, we can take into account the pitch or timbre of a rhythm beat, for example by assigning different tones/sounds to different actions of the fountain. Going one step further, we could combine multiple features at the data level into an overall event: An ascending sequence of sounds could be

appropriately assigned to a water fountain with increasing height – exciting: what does *appropriate* mean in this context?

3. **reaction of the fountain:** If we have decided on the points 1. and 2. in our model, the question naturally arises, how finally the assignment to the reaction possibilities of the fountain as a corresponding visualization can be done. We will quite certainly not have the real fountain at our disposal and an easily obtainable fountain model with simple possibilities of control is not known to us (alternatives for self-construction will be proposed later). A variant for visualization is the use of a simple light organ: This can consist of one or more lamps with the possibility to switch them on and off individually at defined times. Here, too, you will hardly be able to avoid building it yourself as a small technology project without investing a lot of money. What has proven successful in our implementations is the use of a computer program that practically simulates the light organ by means of different color fields that can be switched on and off. As part of a computer science project, something like this can be implemented on your own, but it is equally possible at this point to provide students with such a program with a simple interface – such as input via a simply structured text file.

In the next section, we will present an implementation in the form of a middle school modeling project based on the assumptions made. By focusing more on various aspects or taking them out of focus by providing materials and software tools, the time frame, subject focus, and content requirements can be very finely regulated.

4. Proposal for implementation in middle school

In this section we make a number of suggestions for an implementation of the project *Musical Fountain* in the middle school. As already indicated, we will make simplifications or cut back on the original objectives at various points. With the conception of an implementation in the form of project instruction we would like to refer gladly to the general remarks in Herbert Gudjons (1986). It should be noted that engaging in mathematical modeling requires a teacher to perform two tasks that may be difficult to reconcile: First, possible solutions must be checked for their technical appropriateness with respect to the chosen grade level and to the developmental level of the students. On the other hand, it should be based on the presumed motives and ideas of the learners and offer as much room for maneuver as possible for the realization of their own ideas. The goal, then, is to reconcile instructional goals and articulated action goals, which can be done in two ways in action-oriented instruction: "Either one manages to get the students to embrace the teaching goals as action goals (which is more likely to succeed the more the teaching situation contains problems/topics/tasks that are meaningful to the students), or else space is given to address differences of interest, to work out opposites, and to elaborate and pursue common action goals.", Herbert Gudjons (1986).

The first significant simplification along this path is that instead of a pop song in the form of an MP3 file, we use a more or less simple rhythm as the source material.[4] This has the advantage that, on the one hand, we can very easily generate the required audio file ourselves as a recording of a rhythm tapped by hand on a tabletop using a smartphone. Students can also do this themselves – and will probably get creative and find joy in the process. To kick things off, we would like to give the following rough directions:

- length of tapped rhythm piece: about 15-30 seconds

[4] cf. sample recordings can be requested from the authors

- reduce background noise as much as possible, i.e. microphone very close to the table or give up recording as homework
- create only one or very few *sounds* at the beginning
- a brief introduction to making your own recordings or provide a ready-made audio file
- before starting to work with the free audio software *Audacity*[5] give a short introduction to the features of the software that will be used

For the following example, we hit a simple rhythm on a cajon, but you obviously don't need a professional musical instrument for the project. Before starting the recording, it should be clarified that the results can be exported in one of the formats *.wav, .mp3, .m4a* and exported to the computer used for further processing. The examples *Cajon.m4a* and *Cajon 2.m4a*, which as already mentioned are available from the authors, can be used directly for experimentation. No rhythm "according to notes", was recorded here and the timing is anything but *tight*, but it is precisely from this that one can derive a few small investigations later in the analysis, which are interesting and can introduce into the matter.

We used the very good, powerful and free audio software *Audacity* for the analysis, which is available for all common operating systems. In addition, we recommend installing the *FFmpeg* libraries, whose installation is also described [6] – otherwise only *.wav* files can be opened and saved. These have a significantly larger memory requirement and, secondly, must be created in the first place if a smartphone's recording app does not offer this format.

After opening the self-recorded "piece of music", in Audacity we can see the waveform, i.e. the time course of the amplitude of the recording, which may look like in Figure 2. Smartphone recordings often have only one audio track, so they are mono recordings. If you have a stereo recording, we recommend converting it to a mono track: To the left of the waveform, in the small rectangular field with information about the audio file, next to the file name, you will find a black triangle with its tip pointing downward. Behind it is a drop-down menu in which you select *Split Stereo to Mono*, then select one of the two tracks by clicking on it, and then in Audacity's *Tracks* menu delete it with *Remove Tracks*.

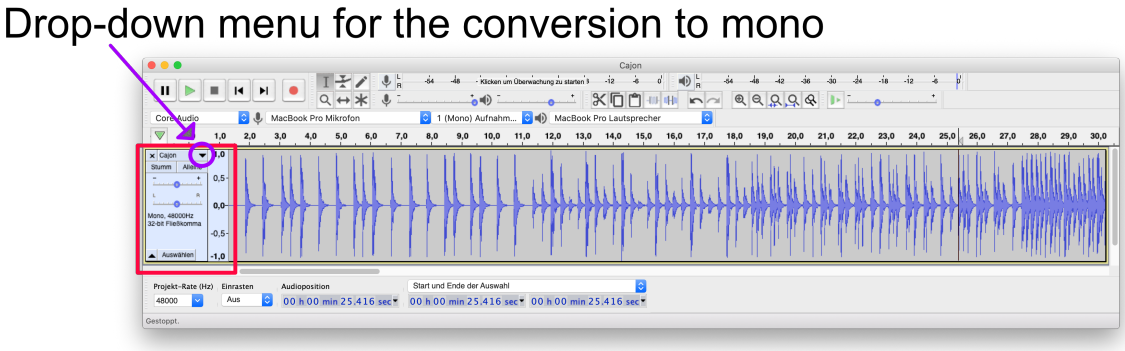

Figure 2 – Waveform of the first 30s of a simple rhythm (©Martin Bracke 2021)

The next step is to find out the *features* of the piece to which you want to react later in a suitable form with the fountain or another suitable output option. To do this, it must first be clear how the waveform is related to the audio signal. Since a vertical line runs through the diagram as a marker over time when listening, no elaborate or complicated explanations

[5] http://www.audacity.de
[6] https://www.audacity.de/empfehlungen/

are needed at this point; the students will be able to understand and use this intuitively. Of course, the background can be addressed depending on the time needed, the possibility of links to physics, and the interest of the learning group.

The next task for the students is now to determine the timing of the individual rhythm beats and note them down or – even better – record them in a spreadsheet. At this point, there are again opportunities for consolidation: Students can each work on their own piece and determine times. But one can also process a longer piece through human *parallelprocessing* by dividing the entire piece into several small time intervals, each of which is then best processed independently by several students. Indeed, this way one gets several data sets and can address the concepts of *measurement error/accuracy*, *mean*, and *median* as desired, as will be explained in more detail below. It is convenient to form pairs where one team member reads time points in Audacity and the other records the values on paper or in the spreadsheet.

To illustrate the possibilities, in the following table we have determined the times we are looking for in several ways[7]:

Waveform, M1	1,904	2,460	3,081	3,359	3,723	4,322	4,921	5,456	5,755
Waveform, M2	1,925	2,460	3,124	3,402	3,701	4,300	4,878	5,477	5,755
Spectrum	1,925	2,460	3,081	3,380	3,701	4,322	4,878	5,456	5,755
Spectrum, zoom	1,894	2,458	3,078	3,367	3,692				

The first two lines could come from the measurements of two learners and one can discuss several things at this point:

- How many decimal places are useful for us?
- How accurately can we determine the time points?
- What role does the accuracy of an individual measurement (i.e., an individual learner) play in determining the data?
- Can we get an overall better result from multiple measurements? If so, how?

The accuracy with which the numbers should be recorded is certainly primarily related to the discrimination of the temporal offset of audio signal and, for example, a light signal that is later produced in response to selected features of the piece of music. At this point, you can pretend something, but you could also wait to resolve it until the end of the project, when learners can make the observation themselves and thus answer the question. Although we have recorded in the table the maximum 3 decimal places given by Audacity, hundredths should be accurate enough.

On the question of how to determine the timing as accurately as possible, here are some more tips: You might have a hunch that the time point you are looking for can be found exactly where the amplitude is greatest in the graph. This place is usually visible as a very thin line, which you can hit pretty accurately with the audio cursor in Audacity and then read off the corresponding time. If the question arises whether this visual feature actually matches the timing of the acoustic signal exactly, you can use two different methods to slow down the audio signal: At the top of the toolbar, in the right half, there is a second green *playbutton* next to which the playback speed can be changed. If you listen to the rhythm slowed down to 40-50 %, you can check the match of both features pretty well. The disadvantage with this method is that when slowing down with this option, the pitch changes and thus audio signals sometimes sound very funny. For rhythm pieces, this may still work quite well – but don't let your learning group experiment with their current favorite hits...

[7]The time of the event is recorded in seconds

An alternative is to slow down an audio signal while retaining its pitch: this can be done using the *Change Tempo* item in Audacity's *Effect* menu (s. Figure 3). In doing so, distracting artifacts appear when making very large changes, but the tool definitely serves the desired purpose. In higher grades, an interesting question can be raised at this point: How does the tempo change, which must be specified as a percentage, relate to the new length of the audio signal?

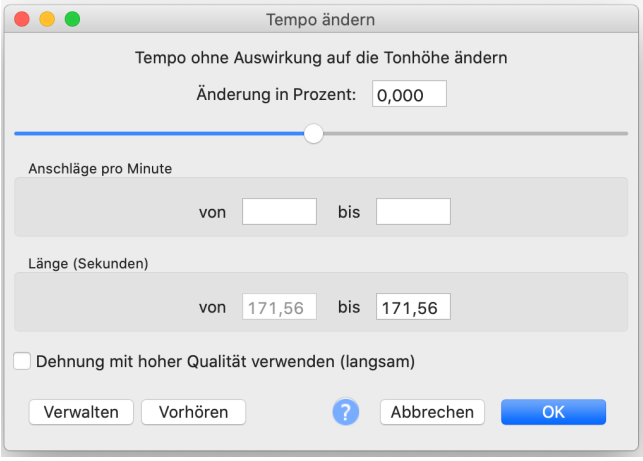

Figure 3 – Change tempo preserving pitch (©Martin Bracke 2021)

For reasons of space, we do not want to go into the possibilities of implementing the topics of accuracy and averaging here. It is relatively obvious from the application which possibilities are offered depending on the class level.

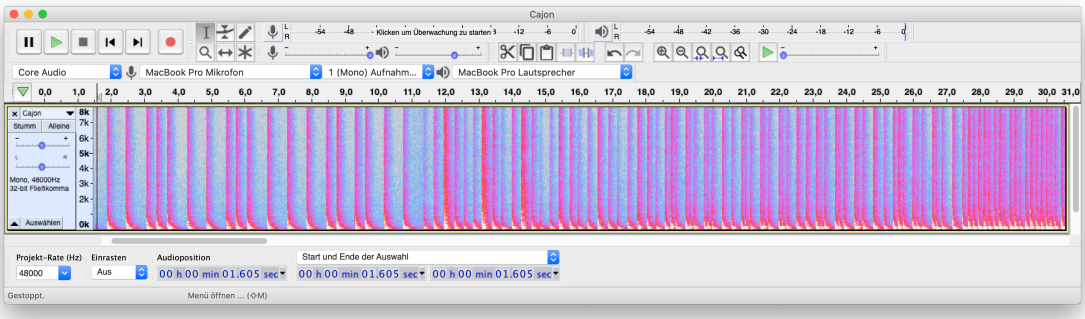

Figure 4 – Spectrogram of the first 30s of a simple rhythm (©Martin Bracke 2021)

This representation form allows possibly an even more exact selection of the event times. In combination with an enlargement (see Figure 5) one can also recognize and select the exact time very nicely over the volume coded by the color. Both variants have been applied additionally for the first time points in the table shown before for comparison.

As an interesting variant in the determination of the time points we would like to consider the possibility to look at the spectrum of the audio signal instead of the waveform. To switch, we select the item *spectrogram* in the drop-down menu at the front of the audio track, which we used earlier, and we get a display as in Figure 4.[8] As a conclusion and to control the obtained

[8]Optional details for the conversion can still be set under this menu item after the switch, for example the frequency range to be examined or the type of display (linear or logarithmic)

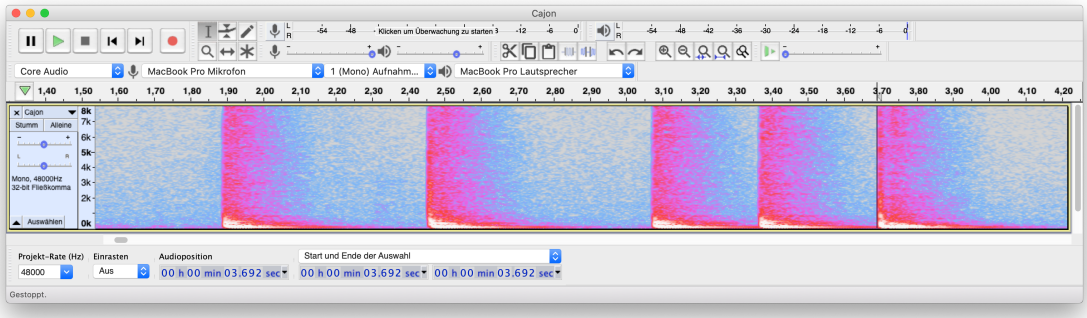

Figure 5 – Zoom into the first 4s of the spectrogram (©Martin Bracke 2021)

results, we are now missing a way to react to the features of a very simple piece of music determined in the form of discrete time points. Here we have two possibilities to offer, which are relatively easy to implement:

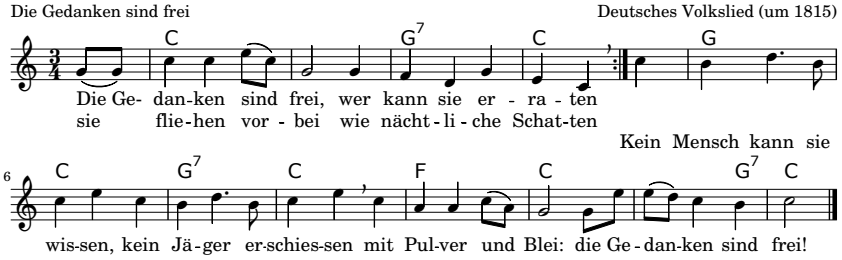

Figure 6 – Unanimous version of *Die Gedanken sind frei* (source: Richard Zillmann. (2020))

A version played on the piano can be requested from the authors. It is only meant to serve as an example here - more complex variants or a different choice of music are of course available in almost unlimited variety for your own implementation. Figure 8 shows the corresponding amplitude representation of this recording in Audacity.

- control of one or more LEDs with an *Arduino* microcontroller: with very little financial effort of $10-15$ euros a circuit can be built without using a soldering iron, which switches an LED on and off at specific points in time. The time points can be copied from the self-made excel table into the control app of the Arduino. After that, a small challenge is to start the music and control the LEDs so that both run synchronously, as this cannot be automated without a lot of effort.
- Using a simple Python program, colored geometric figures on the computer screen can be switched on and off in sync with the music. Again, the previously determined times serve as input data, and the program can be made available to the learning group.

Even if both variants are not comparable to a real fountain, where water heights of several nozzles change synchronously to the music, the result of the previous steps can be visualized very nicely, so that conclusions can also be drawn about the quality of the assumptions and simplifications made. Corresponding hints for implementation can be found further on in section 8.

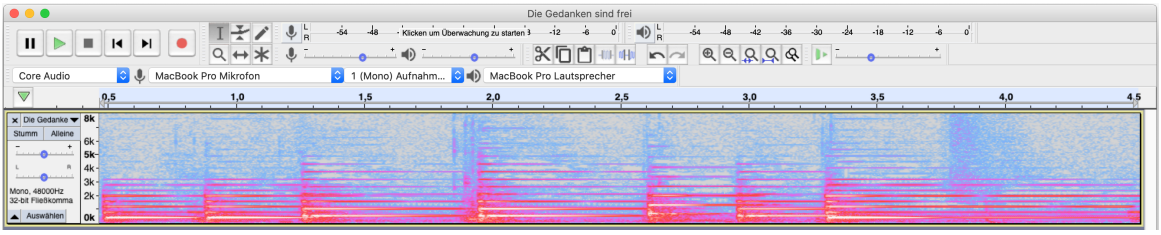

Figure 7 – Spectrogram of the beginning of *Die Gedanken sind frei* (©Martin Bracke 2021)

To conclude this project proposal for the middle school, we would like to point out that there are many exciting possibilities for expansion that can be considered if time is available, students are interested, for internal differentiation or the possibility of interdisciplinary teaching. In terms of source material, for example, monophonic pieces of music – sung, whistled, or played on an instrument and recorded as before with the smartphone – can greatly expand the range of useful features. One can thus react to sound duration, pitch, or even related events. In terms of technical implementation, multiple LEDs in different colors or the use of different geometric figures with variable color and brightness when using a Python program would be extensions that come to mind. As a concrete example, we have chosen here the old German folk song *Die Gedanken sind frei*, which was voted the most popular German folk song by listeners of MDR in 2011:

Without wanting to go into too much detail here, one can at least read off the times of the attack of the individual tones quite well from the representation and also make statements about the relative volume. This representation says nothing about the pitch. However, if we switch to the representation as a spectrogram as shown in Figure 9, we can also obtain information about the individual tones. This can be – independent of the exact assignment of tones – the fact that the pitch rises or falls at different times.

However, since a complex instrument like a piano not only produces pure sine oscillations, but also offers a number of harmonics in addition to the fundamental frequencies, the matter is a bit more confusing than one might first assume: The zoom in figure 7 on the beginning of the piece with the first notes to the song text *Die GEdanken sind frei,..* gives a deeper insight and offers a view on possible deepenings like *Determination of the fundamental frequencies and frequencies of the overtones, estimation of note lengths and rests, etc.*.

Another very interesting side topic that deviates a bit from the original goal of the music fountain can be the following variation: students try to keep as good a timing as possible when tapping their rhythms and control the quality by means of the tables with time events they have created themselves as before. Here, some arrangements may well have to be negotiated in advance – ideally through dialogue – and the task can create a motivating competition. Instead of getting a rhythm as good as possible by feeling, the task can also be to tap along to a played piece of music as *tight* as possible and to examine the result afterwards for its quality.

5. Selected options for specialization

So far, the project has been implemented in several realizations of the KOMMS[9] as well as very related modeling content, such as the construction and control of a light organ, with high school students. In the previous section, we suggested how reduction at various points

[9]The **Ko**mpetenzzentrum für mathematische **M**odellierung in **M**INT-Projekte in der **S**chule was founded in 2014 as a scientific institution of the University of Kaiserslautern and deals, among others. with the design and implementation of real-world application projects for students, as well as with accompanying research,

Figure 8 – Waveform of *Die Gedanken sind frei* (©Martin Bracke 2021)

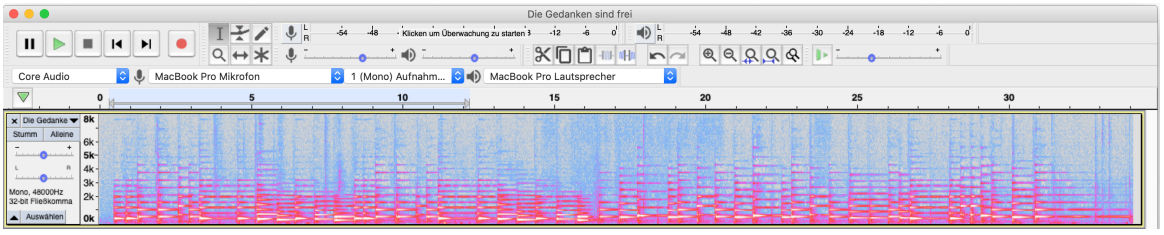

Figure 9 – Spectrogram of *Die Gedanken sind frei* (©Martin Bracke 2021)

can enable implementations at the middle school level. Many technical details can be easily hidden and treated as a black box. With sufficient project time or as a possibility for internal differentiation, it is nevertheless possible to address this content. The following table provides an overview of the implementations carried out so far:

Event	Grade	Time scope	Focus
STEM exchange	11 & 12	10h	Analysis (Audacity)
Project day	12	6h	Sound waves & Fourier analysis
Talent School I	10-12	12h	Analysis (Python)
Talent School II	10-12	20h	Analysis (Python)
Modeling Week	11 & 12	30-35h	Building a light organ

Except for the project day of a mathematics class, this was project work in which only little structuring input was given and the focus was on an independent and autonomous form of work. The content ranged from a focus on modeling (What are the features of a piece of music?) and implementation with the help of Audacity and provided Python scripts to Fourier analysis of pieces of music with self-written Python programs.

The project day of a mathematics class of the 12th grade was designed and implemented in the context of a master thesis (see Meyer, S. (2019)). Here the time schedule was much tighter with structured inputs: At the beginning, after the introduction of the topic, there was an introduction to audio analysis (*What is music?*, *What are sound waves?*, *Basic idea Fourier analysis*) with an overview of the most important functions of the software Audacity, which was followed by a group work phase with a final presentation.

5.1. Mathematical and physical specialization options

If one would like to increase the mathematical demand, then a part of the work, which was taken over before manually and/or by Audacity, can be taken in hand: We would like to use a self-written computer program to automatically transfer an audio file into the targeted Excel spreadsheet. For this we need a way to extract from the data contained in an audio file the features we would like to use: Time of occurrence of a signal, possibly with duration and pitch –

if it is a sound. At this point, the opportunity arises to make very concrete connections to physics and to perform experiments for illustration. Such a connection was planned and implemented, for example, for a series of lessons in physics for grade 7 of middle school, which is presented in detail in the master's thesis by Saygushev, D. (2016). This shows that the inclusion of relatively complex mathematical tools in middle school is possible and gives further ideas for own implementations of the music fountain project in middle school.

If we don't want to completely hide the mathematical background in the process, or if we want to provide a basic understanding in a computer-based implementation, we have to deal with a mathematical description for musical events, very specifically with the basics of acoustics for the representation of tones. Since this is only an optional component in our own modeling projects, we will deal with it in the separate section 6, which can be skipped if you read it.

5.2. Preliminary remarks on the use of the computer with self-developed algorithms

If you want to work with real data in the form of pieces of music, the project as described before can only be implemented very laboriously by hand, if only because of the abundance of data. However, the necessary programming knowledge for the use of the computer is manageable. Besides the important basics (loops, branches) one has to work a lot with arrays or lists, the provision of program fragments is possible and lowers the entry hurdle here.

Although working with Excel would in principle also be possible with more detailed models than in section 4, we would like to advise against it for the following reasons: Already a piece of music with a length of one minute and a sampling frequency of 44.1 kHz (corresponding to CD quality) is described as a stereo signal by a good 5 million floating point numbers. The maximum number of rows/columns that can be used in Excel is already below this with $2^{20} = 1048567$, so that only very short audio snippets would be analyzable – not to mention the clarity and processing speed. Since tools like Fourier analysis are also not available as standard in Excel and would have to be installed later, one would have to be very familiar with Excel on the one hand and have really good reasons for using it on the other.

In our student projects we have been using the Python programming language for several years. Here we recommend the free distribution *Anaconda*, which is available for all common operating systems and already comes with the required libraries (`https://www.anaconda.com`). We usually give the students some program examples for handling audio data directly and explain possible operations by way of example. In our experience, interested learners with no prior knowledge of the code have thus quickly become familiar with it and were able to continue working independently after a short time. The examples can be found in section 6.7. However, it is a prerequisite that the supervising teacher has at least a basic knowledge of Python, which is especially advantageous for the debugging that is certainly due.

Of course, there are a number of options when it comes to implementing your own ideas. At this point, we find the following criteria important when choosing an appropriate language:

- free (free of charge) availability;
- easy use & flat learning curve;
- wide range of existing standard algorithms;
- availability for all common operating systems; and
- use without internet access possible.

For us, Python is a very good compromise in this respect – and the language is now also already widely used in schools. But depending on the individual circumstances and the own learning group, it can of course make sense to choose other tools (computer algebra systems) or programming languages.

5.3. Processing waveforms in Python

As mentioned in the previous section, the musical fountain project was implemented during two *Talent Schools* of the Fraunhofer Society. The first implementation was an extracurricular learning opportunity for mathematically gifted and interested girls of the upper secondary level of schools of the MINT-EC network. Four work phases of three hours each were available, divided over two days. The results were presented on the third day.

The main criteria followed in the project were oriented to the terms *loudness* and *rhythm*. Since no frequency term was initially necessary for this analysis, the students worked with the original data, i.e. the waveforms.

One of the problems that emerged was in the nature of the data: Using amplitude as a measure of loudness initially yields a loudness-time function whose values vary widely. The students therefore developed a method to smooth the data (see section 5.3).

Another subgroup worked on an algorithm that should be able to recognize the rhythm of a song. This problem was partially solved in the fourth phase of work. By clever choice of parameters, the code was able to recognize the rhythm of a fixed song excerpt. However, it remained open what the approach should be for the general case (see section 5.3).

Smoothing the volume-time function

To be able to capture the dynamics of the song, the students first looked at the amplitude image of the data. The basic idea was to use the time-amplitude function (see figure 10) as a basis.

To smooth the data, the students developed the following algorithm:

1. Consider the magnitude of the time-amplitude function.
2. Using `localMax` (see Listing 3.1, find the local maxima and write them into a list `locmax`.
3. For each entry in `locmax`, together with the subsequent 180 entries, average them using `mova` (see Listing 3.2); write these averages into a new list `locmaxAverage`.
4. For each entry in `locmaxAverage`, together with the following 30 entries, take the mean, again using `mova`; write these means into a new list `locmaxAverage2`.

We thus obtain an iterative procedure (with two iteration steps) capable of smoothing the function (see Figure 11). Finally, the resulting volume-time function should be used to regulate the pressure (and thus the height) of the fountain.

Regarding the smoothing procedure, one of the first steps (finding the local maxima in Listing 3.1) changes the time scale. Since many intermediate values are dropped from the value table, the piece naturally becomes shorter. Moreover, one must assume that the time shortening is nonlinear, since the local maxima are not necessarily equidistant.

One can deal with this circumstance in different ways. The easiest way is to linearly stretch the time scale at the very end to restore the original length of the piece of music. Under certain circumstances, the small errors that are made in the process are hardly noticeable. If you want to be more precise, you can use the indices of the determined values, which are always carried

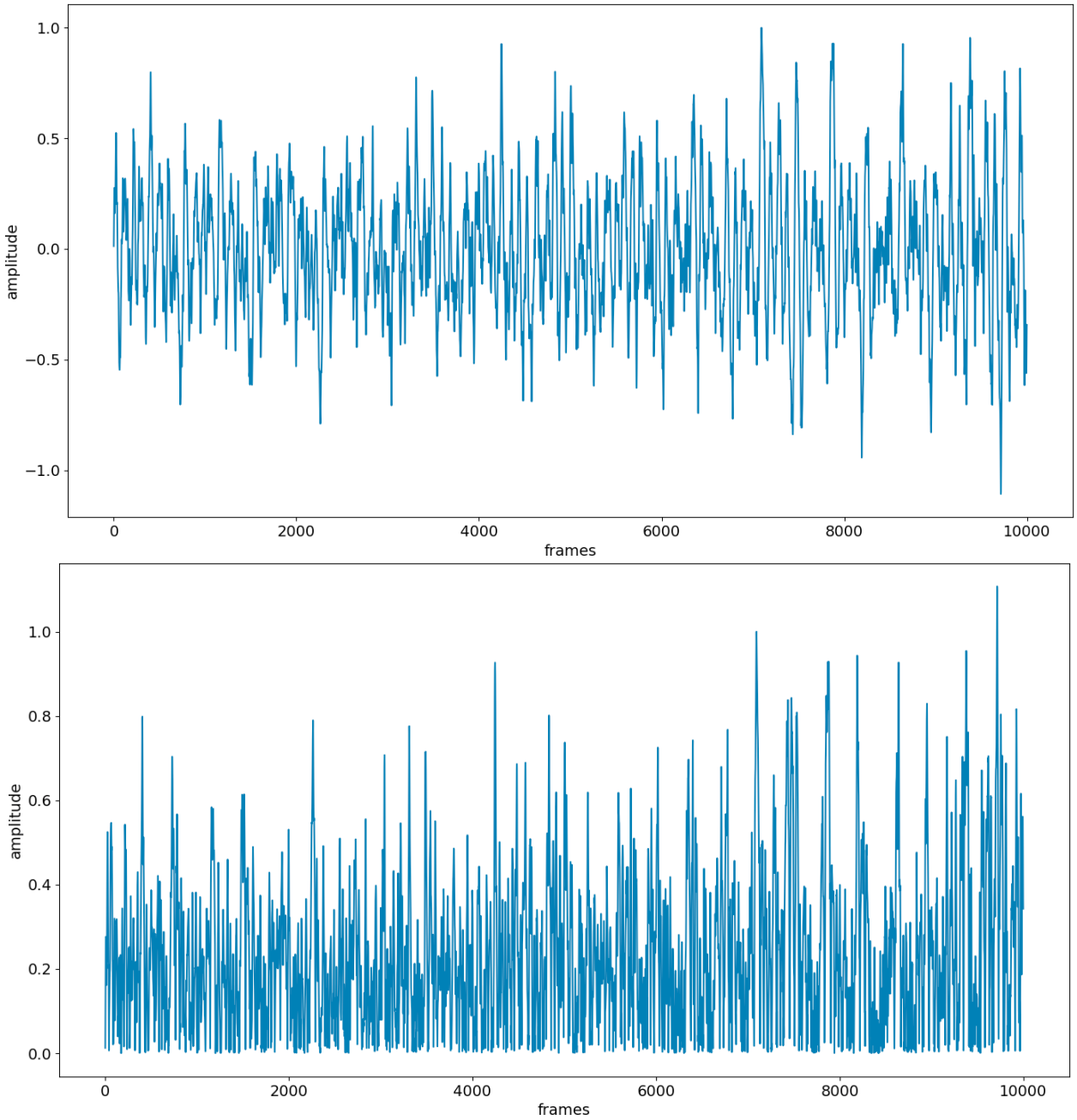

Figure 10 – Left: Section from the amplitude image; right: Amounts of the data. (©Patrick Capraro 2021)

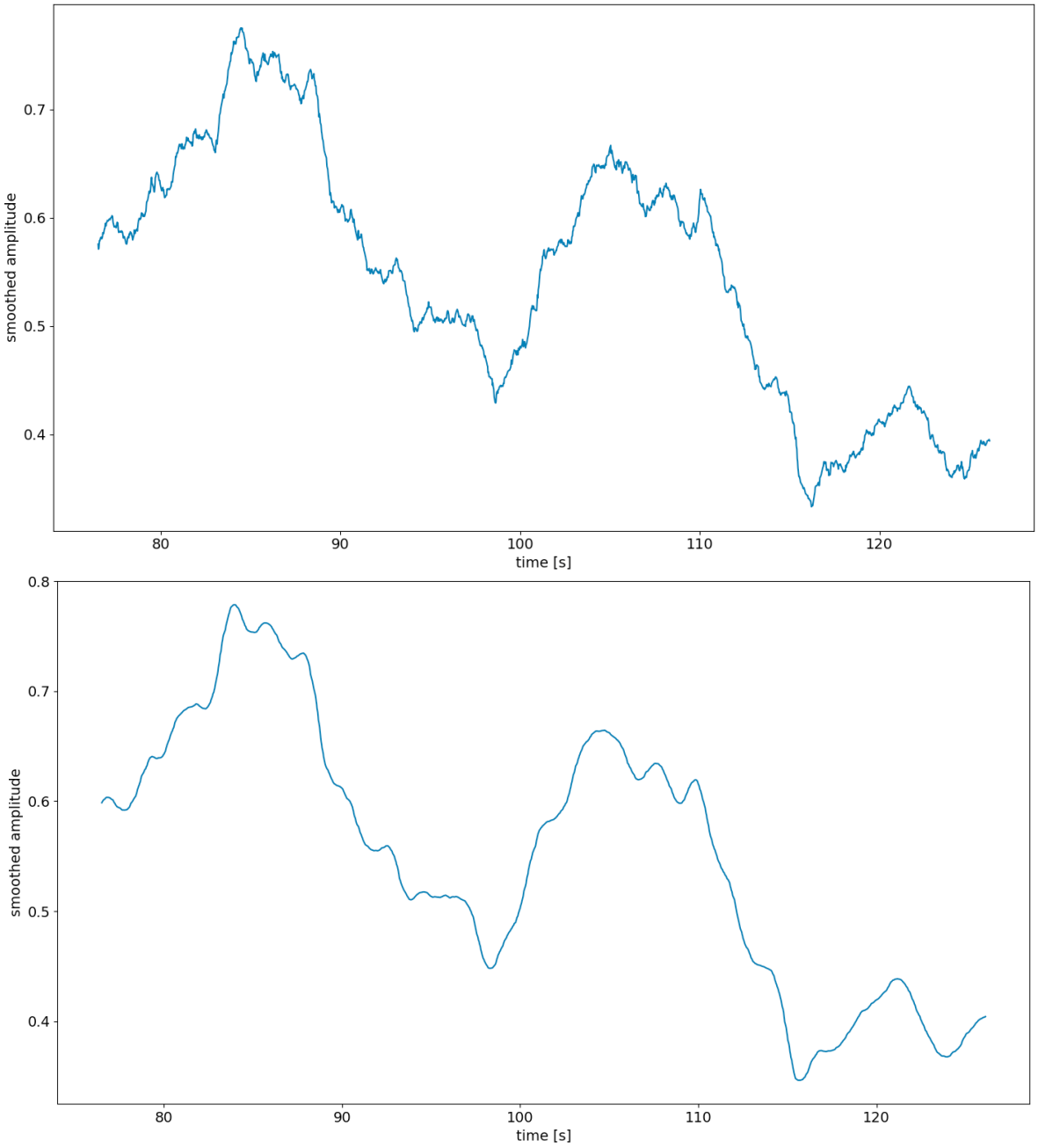

Figure 11 – First and second iteration step of the smoothing algorithm (©Patrick Capraro 2021)

by both functions (Listing 3.1 and Listing 3.2), to approximate the time scale to the initial state.

```
# look for local maxima, save them into a list
# INPUT:  audio data: list
# OUTPUT: list of value pairs consisting of
#         the indices and the values of the local maxima

def localMaxima(audioData):
  N=len(audioData)
  maxima=[]
  for i in range(1,N-1):
    if audioData[i]>audioData[i-1] and audioData[i]>audioData[i+1]:
      maxima.append([i,audioData[i]])
  return maxima
```

Listing 3.1 – The local maxima are determined. Note that the time axis may be changed nonlinearly.

```
# Iteration step of the smoothing algorithm
# INPUT:  local data obtained from the first step
#                 maxima: list, number of values for
#                 averaging: int
# OUTPUT: a list of value pairs consisting of the
#         indices and the mean values
def mova(maxima,noVals):
  listNew=[]
  for i in range(len(maxima)-noVals):
    t=maxima[i][0]
    sumY=0
    for j in range(noVals):
      sumY=sumY+maxima[i+j][1]
    averageY=sumY/noVals
    listNew.append([t,averageY])
  return listNew
```

Listing 3.2 – The function is smoothed by averaging on partial intervals. At each iteration, the list of data is truncated by the value `anz`.

Extract rhythm

The basic idea was to find areas within the song where the rhythm instruments sound as free as possible from background noise (e.g., other instruments) (see Figure 12). It was assumed that the local maxima here all have approximately the same value and that *loud* intervals and *quiet* intervals can be distinguished from each other. The differences in the times at which the loud intervals begin provide information about the rhythm.

The solution strategy can be illustrated as follows:

1. Automatically locate a suitable song section (was not solved).
2. Set threshold S for quiet intervals; background noise should be ignored if it is quieter than S.

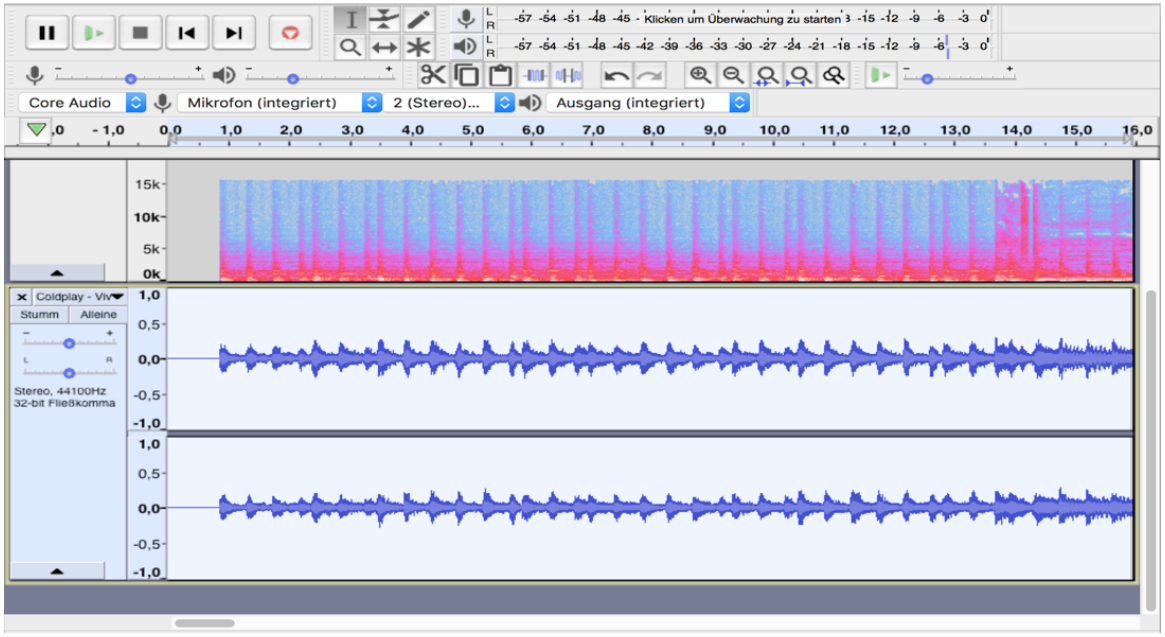

Figure 12 – Analysis of a song excerpt with the Audacity software (©Martin Bracke 2021)

3. Sudden increase of amplitude above value S marks the beginning of a loud interval
4. A loud interval ends as soon as the amplitude falls below the threshold S for a sufficiently long period of time ΔT.
5. Go back to item 3.

Given a *suitable song excerpt*, the described idea will successfully detect the timing of individual beats of the rhythm. So far, however, such an excerpt must be chosen by hand or a very simple piece of music/rhythm must be taken directly as a basis as in section 4. For a fully automatic treatment of complex music pieces, an extension of the described model approach is needed.

6. Background knowledge: Mathematical foundations of acoustics

As shown, many relevant aspects of the project can be discussed without computer science knowledge. However, if one wants to work with real data, there is no way around the computer, if only because the amount of data is too large. This is all the more true because in some places it is appropriate to use advanced computational methods (such as Fourier analysis).

In this section we want to present the mathematical basics, which especially the project supervisor should have in mind in order to be able to interpret the data correctly. As soon as not only rhythms but also tones are to be worked with, the pitch or frequency of a tone becomes important. If one uses software for Audacity for the conversion of audio data into the previously discussed abstract data (time, pitch), an intuitive understanding of the terms used is perfectly sufficient. However, if one wants to work with own programs, one will assign corresponding data in the frequency domain to a signal in the time domain, for example, with the help of the Fourier transformation. The representations calculated in this way are called spectrograms and are very similar to the representation of music by notes. Figure 13 shows a graphical representation of the first 30s of the song *Wanted (OneRepublic)*. Here, when listening, it is

much easier to associate the upper spectrogram view with the music than the lower waveform, from which one can essentially only read the volume directly. In order to be able to work with these mathematical descriptions and avoid mistakes, a basic understanding of the mathematical background is very helpful, as we have often experienced in our projects with students.

If you would like to deal with Fourier analysis in more detail, we recommend the book by Wickert, M. (2013). A clear introduction to the physics of musical instruments can be found in the textbook *Experimentalphysik 1* by Wolfgang (Demtröder, W. (2003), chapter 11.15) and another good introduction to acoustics can be found in the textbook by Lüders, K. and Pohl, R. O. (Eds.) (2017).

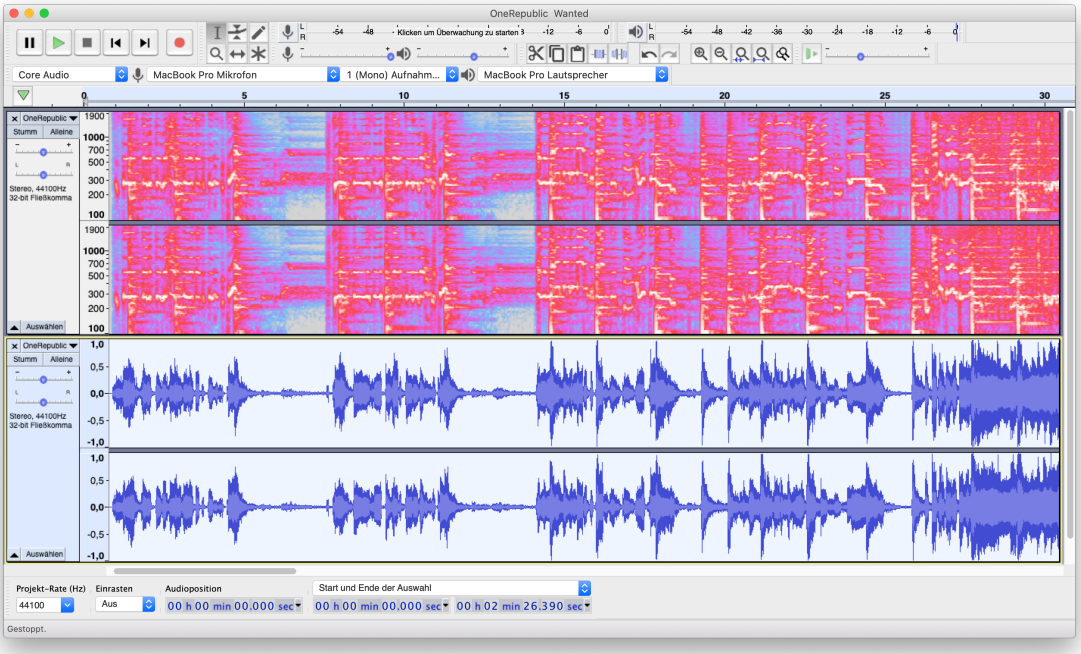

Figure 13 – Spectrum as well as waveform of a pop song in Audacity (©Martin Bracke 2021)

6.1. Sounds and tones

As is well known, sound waves can be described by trigonometric functions. But there are different kinds of acoustic oscillations. *Tone* is a harmonic oscillation which can be described by a sine or cosine curve. The important quantities here are known to be the frequency f and the amplitude A. In addition, there can be a phase shift φ, but this is not important for the interpretation, because the human hearing does not perceive this quantity.

In fact, the φ can be completely ignored, since every harmonic oscillation g can be interpreted as a sum of a sine and a cosine function of the form

$$g(t) = A\cos(2\pi ft) + B\sin(2\pi ft) \tag{1}$$

can be written. To obtain the total amplitude and the phase shift φ, one uses the addition theorem

$$A\sin(2\pi ft) + B\cos(2\pi ft) = \sqrt{A^2+B^2}\cos\left(2\pi ft - \arctan\frac{B}{A}\right) \tag{2}$$

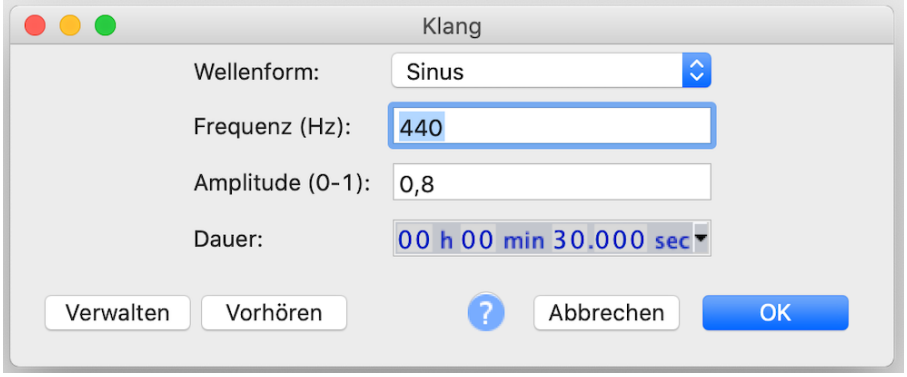

Figure 14 – Creating a simple sine wave with Audacity (©Martin Bracke 2021)

(see Bronstein, et al. (2013), equation 2.137). So it doesn't matter whether we represent the oscillation by a sine, a cosine or a superposition of the two.

As a rule, however, we are not dealing with tones (these also sound awful – test this with a sine generator in the physics collection of your school – or gladly with the corresponding function in Audacity: There, in the menu *generate*, you will find the sub-item *sound*, where you can select a sine oscillation, frequency and amplitude (see Figure 14)). The melodious sounds that surround us are superpositions of several tones – but in such a way that the resulting oscillation is still periodic. We call such oscillations *sounds*.

However, if too many oscillations overlap, then our hearing can no longer recognize a structure, we are then dealing with a *noise*. If frequencies from the very high to very low ranges of the human auditory system occur, then we speak of *white noise*. The human auditory system covers approximately the frequencies from 20 Hz to 20 kHz, whereby this varies of course from person to person and also depending on age.

6.2. Overtones and the Fourier series

People who play an instrument are often familiar with the concept of the overtone series. With instruments, you play a certain tone (the fundamental frequency f), but you produce sounds rather than pure tones. The reason for this is that the multiples of f (the so-called overtone frequencies) resonate. Since the intensities of the different multiples depend on the nature of the instrument, one can also infer the nature of the instrument from their amplitude ratios (see figure 15).

When these harmonics resound together, they produce a pleasant sound – this again has to do with the integer ratios and leads us into harmonic theory. The perceived frequency, however, is not disturbed by the harmonics, because the fundamental frequency of the superimposed wave is still f.

However, we can also see these phenomena mathematically if we familiarize ourselves with the concept of *Fourier series*: Any periodic function g with period length $1/f$ can be written – under certain conditions[10] – as an infinite series of sine and cosine functions. Thus it holds

$$g(t) = A_0 + \sum_{n=1}^{\infty} A_n \cos(2\pi n f t) + B_n \sin(2\pi n f t). \tag{3}$$

[10]There are different formulations of conditions that lead to different convergence behavior. For example, the theorem holds if g is Lipschitz continuous, or if g is locally square integrable.

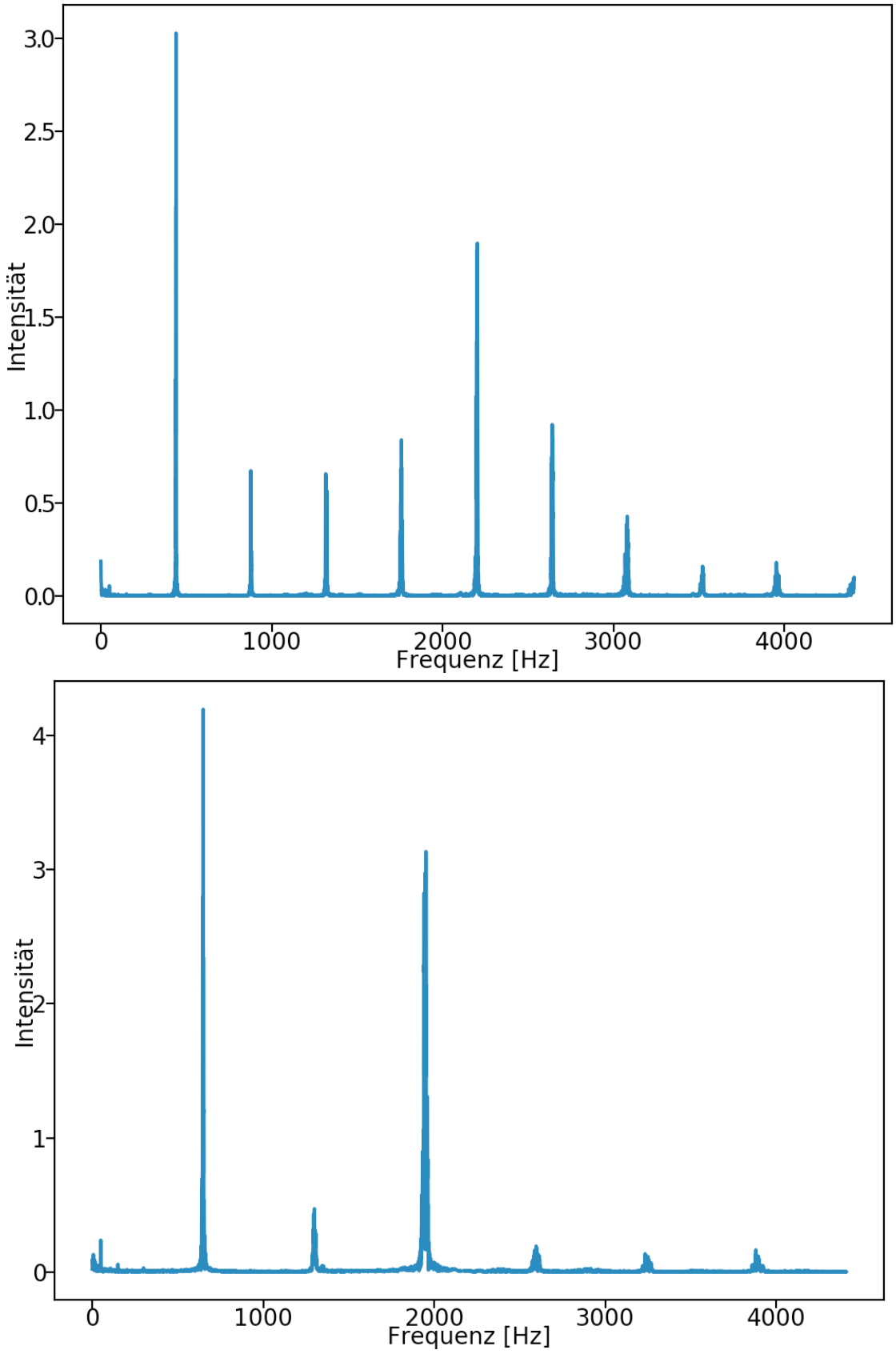

Figure 15 – Frequency spectra of two different instruments (©Patrick Capraro 2021). Up: Spectrum of a violoin. Down:Spectrum of a flute

We see here that we can assign an overtone series to pleasant sounds (namely the periodic oscillations). In the case of noise, however, the fundamental frequency f (the greatest common divisor of all occurring frequencies) is almost certainly outside our perceptible spectrum.

The essential insights for the project that can be derived from the Fourier series – namely, the equidistant frequencies – can be obtained very well through experimentation. Students can generate their own audio recordings (or search the Internet) and analyze the spectra.

6.3. The Fourier coefficients and the Fourier transform

Partial integration can be used to prove the orthogonality relations for trigonometric functions. These are[11]

$$\int_{-\pi}^{\pi} \cos(nt)\cos(kt)\mathrm{d}t = \int_{-\pi}^{\pi} \sin(nt)\sin(kt)\mathrm{d}t = \begin{cases} \pi, & n = k, \\ 0, & n \neq k, \end{cases} \tag{4}$$

and

$$\int_{-\pi}^{\pi} \cos(nt)\sin(kt)\mathrm{d}t = 0. \tag{5}$$

With a scaling of the time variables and equation (3) one obtains formulas for the calculation of the Fourier coefficients

$$A_n = \frac{2}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} g(t)\cos\left(2\pi n f t\right)\mathrm{d}t, \qquad B_n = \frac{2}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} g(t)\sin\left(2\pi n f t\right)\mathrm{d}t. \tag{6}$$

In the boundary transition from discrete to continuous frequencies, one finally obtains the formulas

$$G_{\mathrm{c}}(f) = \int_{-\infty}^{\infty} g(t)\cos(2\pi f t)\mathrm{d}t, \qquad G_{\mathrm{s}}(f) = \int_{-\infty}^{\infty} g(t)\sin(2\pi f t)\mathrm{d}t. \tag{7}$$

for the cosine and the sine part of the frequency spectrum, respectively. The integral transformations in equation (7) are called cosine and sine transformations, respectively.

From the addition theorem in equation (2) we can deduce that we can summarize both partial functions to the spectrum

$$G = \sqrt{G_{\mathrm{c}}^2 + G_{\mathrm{s}}^2}. \tag{8}$$

6.4. The Fourier transform and the complex numbers

In the scientific literature, one usually encounters the Fourier transform in the complex representation

$$G(f) = \int_{-\infty}^{\infty} g(t)\exp(-2\pi i f t)\mathrm{d}t, \tag{9}$$

which can be very easily related to (7) using the Euler formula $e^{iy} = \cos(y) + i\sin(y)\ \forall y \in \mathbb{R}$.

However, the complex structure of the Fourier transform can be disguised very well by identifying the complex numbers with $\mathbb{R}^2$ and using 2-tuples in the notation.

[11] Note that we assume $n, k \in \mathbb{N}$ and $f \cdot T = 1$ in what follows.

6.5. The discrete Fourier transform

The Fourier transform causes difficulties when working with real data for two reasons, since one

- does not have the data available for an infinitely long time interval;
- usually works with finitely many data (i.e., a table of values of g).

The solution is to move to the discrete Fourier transform, which is much more student-friendly than the daunting integrals in (7). This is obtained by approximating the integral by a sum and replacing the integration domain with a finite interval. If $g \in \mathbb{R}^n$ is the vector containing the data, we obtain with

$$G_{\mathrm{c},m} = \sum_{t=1}^{n} g_t \cos\left(\frac{2\pi \mathrm{i}}{n}(t-1)(m-1)\right), \quad G_{\mathrm{s},m} = \sum_{t=1}^{n} g_t \sin\left(\frac{2\pi \mathrm{i}}{n}(t-1)(m-1)\right) \tag{10}$$

the frequency spectrum, where $m = 1, \ldots, n$. Similar to (8), we can summarize the data and obtain the vector $G \in \mathbb{R}^n$ with $G_m = \sqrt{G_{\mathrm{c},m}^2 + G_{\mathrm{s},m}^2}$.

These formulas can be understood by students who have no knowledge of high school calculus. Therefore, the Fourier transform can be reduced to an upper intermediate level by reducing it to its discrete representation – at least when it comes to application with real data.

6.6. Interpretation of data

Digital audio data can be conceived as a vector, as we have seen above. In this case, we neglect the values of the time axis, since they are equidistant. The time step ΔT is obtained from the so-called *sampling rate* A_r (also *framerate* or *sampling rate*), which is simply the inverse of ΔT and has a value of $44,1\,\mathrm{kHz}$ for most digital audio formats.

If you also take the frequency data as a table of values and represent it in a coordinate system, you first have two problems:

- The values on the frequency axis are missing;
- If you plot the data, you get a symmetric spectrum.

Teorema 3.1 *(Sampling theorem). If the data $(g_1, \ldots, g_n) \in \mathbb{R}^n$ are real-valued and even and odd, respectively, then the following equalities for the spectrum in (10) hold:*

$$G_{s,1} = G_{s,\frac{n+2}{2}} = 0 \quad \textit{and} \quad G_{s,1} = 0, \ \textit{respectively, and}$$
$$G_{c,m} = G_{c,n+2-m}, \ G_{s,m} = -G_{s,n+2-m}, \ m = 2, \ldots, \lfloor \frac{n+1}{2} \rfloor. \tag{11}$$

If we interpret the tupel $G_m = (G_{c,m}, G_{s,m})$ as a complex number, the numbers G_m and G_{n+2-m} are complex conjugate to each other for $m = 2, \ldots, \lfloor \frac{n+1}{2} \rfloor$.

This theorem explains the symmetry of full frequency plots and confirms that the second half of the data is redundant – and this is why we included the theorem in the theoretical background. So we can restrict ourselves to the first half.

Teorema 3.2 *(Nyquist frequency). The largest frequency that can be represented in the discrete Fourier transform is the* Nyquist frequency $f_N = \frac{1}{2}A_r$. *This frequency corresponds to the data in the middle of the calculated spectrum G.*

With this statement, we have a means to choose the scale for the frequency axis. This will become necessary as the project progresses, as we need to decide which frequencies, or range of frequencies, to consider for defining appropriate features.

6.7. Simple implementation in Python

As we have seen before, audio data can be interpreted as vectors - albeit very extensive ones - after digitization. This means that in principle, if the sampling rate is not too large, it is even conceivable to process the data with a spreadsheet during digitization. However, with larger data sets the handling quickly becomes bulky and unclear, so that in our projects we prefer to work with small programs in the programming language *Python*. In doing so, many program parts can be provided to the students if time or appropriate programming skills are not available to a sufficient extent. In our experience, the interactive nature of the Python language makes it possible to work in a very intuitive way, focusing largely on the mathematical content.

The following templates were provided for project participants to experiment with audio data.

`scipy.io` can be used to read in `wav` files (see Listing 3.3). For stereo data, you get a two-dimensional array. The two columns of the data array then contain the two channels, which can be merged into one channel by adding them.

```
from scipy.io import wavfile
import numpy as np

# read audio data
[sampleRate,sampleData]=wavfile.read('example.wav')
sampleData=np.array(sampleData)

# combine stereo channels
if(len(sampleData.shape)==2):
  channel1=sampleData[:,0]
  channel2=sampleData[:,1]
  sampleData=channel1+channel2

# normalize amplitudes
maxabs=max(abs(sampleData))
if maxabs != 0:
  sampleDataNorm=sampleData/maxabs
else:
  sampleDataNorm=sampleData
```

Listing 3.3 – Read audio file

The function `fft` calculates the Fourier transform so that we get the frequency spectrum (see Listing 3.4). If we scale the time axis appropriately, we can read the frequencies in Hz.

Be careful when interpreting the data. The array G contains complex values. The functions `np.real()` and `np.imag()` can be used to decompose complex numbers into real and imaginary parts.

When plotting the data, the magnitude of the complex-valued entries is calculated to account for equation (8).

```
from scipy.fftpack import fft
```

```
from scipy.io import wavfile
import matplotlib.pyplot as plt

# read audio data  using the code from Listing 3
# so we assume to have audio data stored in a variable
# sampleData, as a normalized mono signal

# compute Fourier transform
G=fft(sampleData) # sample data in the frequency domain

# scale timeline
times=[i*sampleRate/len(sampleData) for i in range(len(sampleData))]

# plot frequencies; abs()=absulute value
p=plt.plot(times, abs(G))
plt.show()
```

Listing 3.4 – Compute and plot frequency spectrum

7. Misconceptions about algorithms and computers

This is a short, but from our point of view important section for a practical implementation of the presented project. We would like to point out something we observe in our work with students without programming experience relatively often when it comes to questions that have a strong component in computer science or writing their own software. In some cases, there are very clear misconceptions about problem solving with the help of computers. In concrete terms, this means that students encounter and formulate subproblems in their mathematical modeling, the processing and solution of which they do not see as their own task but think that there are already existing solutions for these subproblems that can simply be used. A computer can do so much more than we can – only unfortunately for some tasks, which apparently are very easy to solve for humans , there are so far only very complex or even no solutions at all. Let's think of "tasks as simple as", recognizing handwritten digits or the request to recognize living beings in the images delivered by a video camera and, if necessary, to estimate their approximate distance to the camera (something like this plays a major role in the development of robust solutions for autonomous driving). Even if partial successes already exist with the precondition of good or very good data, even the best currently available concepts can still be thrown out of step if, for example, the video signal is temporarily noisy or a sudden strong incidence of light – caused, for example, by a reflection of sunlight in a glass facade – causes a disturbance that is difficult to take into account.

In our example, similar misunderstandings and overestimations of the ability of computers to solve a problem without appropriate input from the programmer occurred. In our observation, this seems to occur especially with topics to which one has a clear everyday connection and intuitive understanding. In the example of the musical fountain, for example, partial solutions were formulated very vaguely and delegated, as it were, to "the computer": *let the computer determine the rhythm.* The mathematical thinking power that must go into such an algorithm is often not obvious because the potential of computers is overestimated.

In this case, the thought processes of mathematical modeling help to demonstrate the central role of the algorithm in programming. To develop a computer program, the programmer must,

on the one hand, know the data with which he is working, and he must explain, step by step, what computational operations will be performed on the data.

As a reaction to unrealistic demands on the automatic available capabilities of a computer/computer program, it has proven useful in our practice to explicitly step into the role of the computer as a supervisor and to say whether one can execute something or not when asked or prompted by the learner. In the case just described of asking the computer to set the rhythm, it might look like this: We would quickly be able to show that a music file opened in Audacity can be displayed as a waveform or spectrogram, but in Audacity there is no way to automatically display the parts that are relevant to rhythm. What constitutes a rhythm anyway? Audacity does not know! One could, of course, as described in section 4, determine the times by hand, transfer it to a spreadsheet and continue working with this data. However, it is clear that such a manual step would make a fully automatic solution to the problem being worked on unattainable. So, if you have to formulate an algorithm yourself that determines the rhythm, you first have to define what constitutes the rhythm and from what information contained in the audio file the desired result – specifically, the times of the individual rhythm beats – can be determined and in what way. At the latest when not only the loudness is recognized as the decisive criterion, also the mathematical background information from the previous section gets a great relevance and is recognized as necessary.

Another, not so obvious example comes from a processing that took place in the context of a Talent School: At one point, the need arose to pick out the local maxima from a list of numbers – this involved working with waveform data to determine locally particularly noisy passages. Here one can answer that in Python the smallest or largest number of a list including its corresponding position can be determined – no more and no less. So the students first had to define the notion of a local extremum for a sequence of numbers and then describe how they would determine such positions themselves for a given, very long sequence of numbers. Answers of the kind "You can see that!", are of course not valid – but after some queries the principle usually becomes clear very quickly and a first algorithm can be formulated colloquially and then, with the help of the supervisor, transferred step by step into a corresponding computer program.

In this sense, a modeling project that works with real data can help eliminate these misconceptions about problem solving with computers. In the process, the importance of programming is put in a new light for many. And it is often helpful to differentiate even further between the development and formulation of algorithms and their implementation in a program of one's own. Indeed, in practice, one of the two components may not need to be done by oneself because either a suitable algorithm needs to be transferred from another context and only programmed, or existing program libraries can be used after formulating one's own algorithm.

8. Extension of the project by a technical implementation

As previously mentioned in a few places, the modeling project presented in this paper is eminently suitable for integrating technical aspects to almost any extent. On the one hand, this can be useful in case of special interest or existing skills of individual students, on the other hand, it can also be useful in case of an implementation in the upper grades of a vocational school, where the focus is on the technical side. Of course, as discussed earlier, the black box principle can be applied to exclude other, less relevant aspects such as the deep mathematical description from section 6. In this section, we present several approaches to such a technical implementation. For the construction of a light organ with the use of an Arduino outlined in section 8.2 as well as the solution presented in section 8.3 to process audio data in realtime

with a Raspberry Pi and to react to it with light signals, there is a handout available from the authors, which also includes two basic programs for the corresponding realizations.

8.1. Building a model fountain

For the technical implementation you need appropriate equipment (water pump, valves,. . .), which can be controlled electronically. For example, solenoid valves can be used as valves. These can be opened or closed with an Arduino microcontroller or a Raspberry Pi. The Raspberry Pi is suitable in that it has sufficient computing capacity to perform data analysis (e. g. in Python) directly and use the results directly for control. Since in many vocational branches the use of one or even both of these minicomputers is standard, the effort required is realistic in the context of an interdisciplinary modeling project.

The valves can then be controlled via the GPIO pins, to which a voltage of up to 5 V can be switched. Typically, solenoid valves switch at a much higher voltage (e. B. 12 V), so you may need transistors and an external voltage source to control the valves via the Raspberry Pi pins.

8.2. Visualization via a light organ

As indicated earlier, instead of using a real fountain or fountain model to visualize the features extracted from the music with your own model, you can use a simple light organ. So in the end, a light organ is controlled to match the music. This project was already successfully worked on during the mathematical modeling week of KOMMS. There was also a technical realization. If the goal is only to design a model of a light organ, the technical effort is much less, because one can work with LEDs, which get along with a voltage of about 3 V and therefore are controlled directly via the pins of the devices. Transistors and an external voltage source are then only needed if stronger lamps are actually to be controlled.

In the modeling week, the time frame was very flexible, so an introduction to Fourier analysis in the form of frontal teaching was omitted. The information from section 6 was interspersed in appropriate situations for which the students formulated appropriate questions.

8.3. Real-time audio data

Instead of reading the audio data from a file, you can also tap it directly from an audio cable. Here, however, a little insight into signal processing is helpful, which is why such an implementation is also suitable in vocational schools that have already addressed such topics: For example, if you want to tap the signal at a jack plug, you usually have a stereo cable with three wires – the ground and the two audio channels. Each of the audio channels carries an analog signal with voltages of about -2 V to 2 V.

The Raspberry Pi can only read digital signals, so you need an analog-to-digital converter to process the signal. The Arduino microcontroller, on the other hand, can also process analog signals. However, here you have to take into account that both devices only read in positive voltages. Therefore, a purely positive signal must be generated from the AC signal. This is achieved by an offset circuit, which adds the voltage 2,5 V to the signal (see figure 16).

9. Closing remarks

The goal of this paper was to show the wide variety of implementation possibilities for modeling a music fountain in school. To this end, after greatly simplifying the input data and reducing the aims, we have shown the kinds of questions that can be investigated in middle

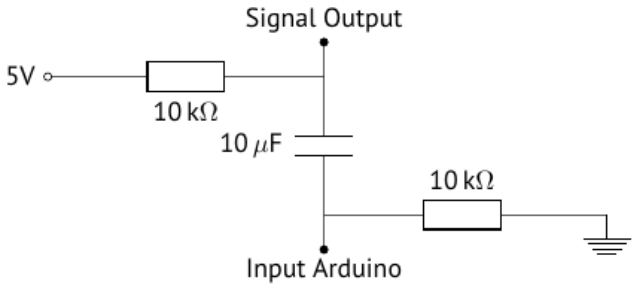

Figure 16 – Offset circuit for recording an analog audio signal (©Patrick Capraro 2021)

school mathematics classes. Many possibilities of variation make the adaptation to the level of different grades, choice of the temporal scope as well as interdisciplinary work possible. Due to this character, the project is also very well suited for is suitable for internal differentiation. Since, as shown, very nice modeling and results are possible in the context of the musical fountain on the basis of basic mathematical knowledge of the content areas *generating, collecting and analyzing one's own data*, *possibilities of representing data*, *calculation and interpretation of mean values (and, if applicable, further statistical parameters)*, implementations with learning groups of other school types are also very well imaginable. In addition to the purely mathematical competencies, the skills in mathematical modeling are also addressed when adapted to the prior knowledge of the own learning group, and also the proposed methodology of project work, especially in an interdisciplinary environment, offers a real added value (Herbert Gudjons (1986)).

An in-depth extension for high school students as well as supplementary mathematical background for the supervising teachers also make it seem possible to work on it as a long-term project, for example within a seminar course. In addition, there are some indications for setting a technical focus, which is particularly suitable for vocational schools with a corresponding orientation, since, if necessary, complex mathematical or physical relationships "can be very well hidden", behind technical, directly experienceable features. This is very useful in classes with a focus on electrical engineering and is suitable to show the connections between the technical realization and the corresponding theoretical background – also a nice opportunity for interdisciplinary teaching!

From our experience in numerous interdisciplinary projects of the kind described above, we can say that especially the various possibilities of linking the project to the contents of physics, music lessons and technical issues will ensure that students will remember such an independently carried out project in the later course of their school education, establish references and possibly even take up questions again and deepen them against the background of greater knowledge.

Finally, we hope that readers will find many ideas for their own classroom projects, which you and your students will hopefully enjoy and find exciting challenges. We welcome questions and feedback in the form of field reports!

Acknowledgements Some of the school projects described in the article were carried out as part of the SchuMaMoMINT project financially supported by the European Social Fund (ESF) of the state of Rhineland-Palatinate. We would like to thank all members of KOMMS involved in these implementations.

Bibliography

- Gudjons H.(1986).
  Action-oriented teaching and learning: Project teaching and student activity (in German).
  Klinkhardt.
- Bock W. & Bracke M. (2015).
  Applied School Mathematics – Made in Kaiserslautern.
  Currents in Industrial Mathematics: From Concepts to Research to Education. Neunzert, H. and Prätzel-Wolters, D. Eds.
  Springer.
- Benetos E., Dixon S., Duan Z. & Ewert S. (2019).
  Automatic Music Transcription: An Overview
  IEEE Signal Processing Magazine, 36 1, 20–30.
- Bracke M. & Capraro P. (2021).
  Choreografien für Musikbrunnen (in German).
  Neue Materialien für einen realitätsbezogenen Mathematikunterricht 8.
  Bracke, M. and Ludwig, M. and Vorhölter, K. (Eds)
  Springer Spektrum.
- Zillmann R. (2020).
  Die Notenschleuder.
  https://www.free-notes.net
- Demtröder W. (2003).
  Experimental Physics 1: Mechanics and Heat (in German)
  Springer.
- Saygushev D. (2016).
  Fourier analysis in the classroom – an application-based approach with implementation in school (in German)
  University of Kaiserslautern, Department of Mathematics.
- Butz T. (2011).
  Fourier transform for pedestrian (in German).
  Vieweg + Teubner.
- Bronstein I.N., Semendjajew K.A., Musiol G. & Mühlig H. (2013).
  Handbook of Mathematics.
  Springer.
- Meyer S. (2019).
  Modeling of a musical fountain – with implementation in the upper secondary school (in German).
  University of Kaiserslautern, Department of Mathematics.
- Lüders K. and Pohl R.O. (Eds.) (2017).
  Pohl's Introduction to Physics, Vol. 1: Mechanics, Acoustics and Thermodynamics.
  Springer Nature

Harald Lesch (2020).
School of the future – learning from the lockdown (in German)
https://www.zdf.de/wissen/leschs-kosmos/lernen-fuer-die-zukunft-100.html

Wickert M. (2013).
Signals & systems for dummies.
Wiley.

Kaiser G., Schukajlow S. and Stillman G. (2022).
A psyccologically - driven view on mathematical modelling processes.
Mathematical Thinking and Learning.

Winter H. (1995).
Mathematikunterricht und Allgemeinbildung.
Mitteilungen der Gesellschaft für Didaktik der Mathematik 61 37–46.

Modelling in Science Education and Learning
http://polipapers.upv.es/index.php/MSEL