Signal Analysis and Processing. Project tasks

Topics for the First Project

1. Compare two algorithms: *Weighted Moving Average* (selecting weights based on a Gaussian function) and *Exponential Moving Average*. Using these algorithms, demonstrate the extraction of regular signal components (trends, periodicity or quasi-periodicity, etc.) in a signal, the possibilities of reducing existing or artificially added noises and possible failures. Also compare the results obtained by computing with different algorithm parameters K and L (in the case of weighted moving averaging) and α (in the case of exponential moving averaging). In your Report, provide a complete analysis (consisting of all the above mentioned studies) for only one selected signal (for which you consider the most representative results), For other signals studied, the Report may not include illustrations of all the studies (however, you may be asked to demonstrate the operation of the omitted studies on a computer screen during the discussion).

2. *Recurrence Plots*. Select the threshold distance parameter r according to the user's preferred percentage of black dots (pixels) in the Recurrence Plot. Demonstrate the algorithm's performance, giving examples of how the results can be interpreted in different cases. Compare the results obtained by computing with different algorithm parameters D and d, and with different metrics: L_2 (Euclidean); L_1 (Manhattan); L_∞ (maximum). In your Report, please provide a complete analysis (consisting of all the above mentioned studies) for only one signal of your choice (for which you think the most representative results are obtained), For other signals studied, the Report may not include illustrations of all the experiments (however, you may be asked to demonstrate the operation of the omitted studies on a computer screen during the discussion).

3. *Correlation Dimension*. Optimisation of the algorithm for calculating the correlation dimension is advisable (but not mandatory). The metric used – L_2 (Euclidean), L_1 (Manhattan) or L_∞ (maximum) – is at your choice. Demonstrate (with illustrations) the results of the algorithm for calculating the Correlation Dimension with the algorithm parameters D and d fixed. Investigate and show graphically how the resulting estimate of the Correlation Dimension depends on the algorithm parameter D. Compare the results obtained with different algorithm parameters d. In your Report, provide a complete analysis (consisting of all the above mentioned studies) for only one selected signal (for which you consider the most representative results). For the other signals studied, the Report may not include all the illustrations (however, you may be asked to demonstrate the operation of the omitted studies on a computer screen during the discussion).

4. *Cross-Correlation*. Demonstrate the performance of the algorithm, giving examples of how to interpret the results in different cases. Search for pairs of signals for which statistical similarity is recorded at different levels, in the presence or absence of real-time t synchronisation in the signal pair under study. For one selected pair of signals, artificially noise one or both of the signals with noise of varying intensity and investigate how this affects the resulting estimate of statistical similarity. When selecting the signal pairs to be processed, in at least one case it is possible (but not mandatory) to take some data and a cipher for it (recalling the cryptographic algorithms you have studied; in this case, the implementation of the cryptographic algorithm does not have to be of your own design). This is one way of assessing the robustness of a high-quality or low-quality encryption (e. g. Caesar cipher technique).

5. *Autocorrelation*. Demonstrate the performance of the algorithm, giving examples of how the results can be interpreted in different cases. Numerically estimate the duration of the inertia (memory) in the signal values (expressed in the same units of time, e. g. milliseconds, minutes, years, that define the measurement range of the signal values under investigation). Artificially noise one selected signal at

different intensities and investigate how this affects the resulting estimate of the inertia duration and the plot of the Autocorrelation function.

Topic for the Second Project

6. *Fourier Filter based on Cooley–Tukey Fast Fourier Transform (FFT) algorithm.* Implement an FFT-based algorithm (digital equalizer) for removing, attenuating or amplifying selected frequencies in a signal and demonstrate its performance.

   Modifiable frequency range (implement procedures for high frequencies, low frequencies, removing the selected band / attenuating / boosting) the user must control by specifying hertz (possibly kilohertz, depending on the specifics of the signal) but not harmonic numbers (array indices, which are important only to the programmer)!

   For this task it makes sense to process digitised audio signals, but there are other applications of practical value (e. g. in medicine, etc.).

   The implementation of the FFT (and inverse FFT) – with or without recursion – is at your choice, However, you must implement the implementation completely independently (from scratch) and *test* it, for at least one signal, comparing the results with those of the simpler but much slower Discrete Fourier Transform (not Fast Fourier Transform) for the same signal. You must also implement *Discrete Fourier Transform* (DFT) – it will take just a few lines of code.

   You must also demonstrate that your implementation of the FFT does indeed run significantly faster than the DFT – by comparing the FFT and the DFT *computation time* for the same signal (the computation time should be compared for a sufficiently long signal, possibly of synthetic origin). In addition to comparing the FFT and DFT results, further test the FFT implementation – by demonstrating that the FFT and the inverse of the FFT applied to the signal will give you the original signal (within rounding errors).

   You can (but are not required to) also test the FFT by comparing your implementation with, for example, Python's NumPy fft and ifft methods, corresponding MATLAB procedures, etc.; it is advisable to take synthetic short signals (consisting of as few as 8 or 16 values) for this testing. The testing and the comparison of FFT and DFT computation times will be briefly mentioned in the Report and demonstrated in detail on the computer during the debriefing.

   In your Report, give creative examples of the possible operation of the developed Fourier filter, modifying the ideas proposed below or proposing your own.

   Possible ideas:

   Artificially drown out one selected signal with a high intensity noise that only affects a specific frequency range (which will not be known to the signal filter, again remembering that all frequency manipulation must be done by means of hertz'es rather than harmonic indices), So, don't artificially noise the signal itself (which is likely to contaminate all frequencies), but its spectral components (which you will find with FFT). Suggest a way of knowing the energy of the original signal (spoofing – note Parseval's theorem, you just need to implement an elementary adder programmatically) to identify the frequency range of the noise effect in an automated approximation (without knowing it before the filtering), filter this noise and assess the robustness of the filtering. If the signal itself is quite regular (no or almost no oscillations in the graph), it is possible to simply compare the graphs of the original and the reconstructed signal, plot them side by side or graph the relative error (divided by the signal maximum, thus adapting to the scale of the values being tested) or a fragment of the corresponding graphs (if the signal is of long duration and the graphs simply do not fit into a single illustration). In the presence of irregularities in the original signal, the error at individual points (at particular points in time) may be large, but if it can be achieved, to remain relatively small over large time spans – the quality of the filtering (globally speaking) can be quite good, so in that case, might it make sense to calculate the mean and standard deviation of the relative error?

Demonstrate how well (or not) it works to separate two pre-combined signals (e. g. two audio clips) each operating on a different (e. g. human speech and the sound of a door creaking open) or in similar frequency ranges (e. g. female and male voices). The quality of the separation can be quantified through the mean and standard deviation of the relative error of the reconstructed signal, as well as by listening to the audio files (during the discussion). The signals to be combined may be both real (e. g. sounds) and synthetic (e. g. a sine wave of a given frequency).

From data of medical origin (for example, a specific electroencephalogram or electrocardiogram channel) recorded using electrical devices, remove the so-called “utility frequency” (which partially obscures important information in the signal, e. g. making it difficult to detect epileptic peaks) of 50 Hz.

Highlighting of frequency components in irregular (highly oscillatory) data (e. g. seasonality in signals of economic or meteorological origin).

Give examples of both successful and unsuccessful operation of the Fourier Filter (thus further demonstrating its capabilities, nuances and limits).

Assignment description and Report requirements

From freely available digital signal databases (see Additional References at the end of this assignment description, or from your own findings / available data) experiment with several signals. The area of origin and the specific signals are up to you, but also take into account the specifics of the algorithm when choosing them. For example, in the Recurrence Plot method, the signal length must be sufficiently small (but you can preprocess the downloaded signals additionally). Additional signal preprocessing may also be needed for the Cooley-Tukey Fast Fourier Transform algorithm. When studying the Cross-Correlation, one should choose a pair of signals that have something in common (for example, fluctuations in the exchange rates of different currencies, but this is not the only appropriate choice).

If you can find examples of interesting signals in the public domain – you can use them (please cite sources).

For each assignment, you must process at least 5 different signals (or 5 pairs of signals, if you deal with cross-correlation) from more than one source (e. g. digital signal databases). Most (an exception may be made for one signal – for algorithm testing purposes, to demonstrate that your results are consistent with independently obtained results) of the signals you analyse do not have to match the examples discussed in lectures, also desirable, they should be of different origin (e. g. medical, financial, meteorological, sound, etc.) and should not repeat analysis done by other students.

The entire algorithm must be programmed by you, *without employing additional libraries, code or pseudocode found on the internet, including code written by other students*. However, you can use tools or libraries that you have not developed yourself for visualisation of results, arithmetic of complex numbers, data input. You can also use external tools or libraries (e. g. for computing the Fast Fourier Transform) to test your own code. For testing your results, it is convenient (but not mandatory) to use computational and simulation tools such as SageMath (free), GNU Octave (free), R (free), MATLAB (VU MIF has a license), Wolfram Mathematica (15-day trial available for download), Maple (VU MIF has a license, but not the latest version), etc., most of the algorithms can be implemented in a sequence of one or more commands. You can also test how your computer tool works with the signals discussed during the lectures. If in doubt about what can and cannot be used, please ask.

Requirements

- Submit a short Report (5 pages or more, including illustrations, but excluding appendices, font size 12pt, line spacing 10% larger than single spacing; write more if motivated) a description of the work carried out – in PDF format.

  The Report must include:

  - Your name, the title of the algorithm implemented, the date of the Report,
  - a numbered list of the signals analysed, mentioning the field of origin of each of them (e. g. exchange rate fluctuations, electrocardiogram RR interval signal, etc.) and the full http address of the database from which it was downloaded or other source (e. g. if you process your own data),
  - whether the particular signal has been subjected to any additional preprocessing (e. g. truncation, snipping, noise reduction), giving details of how this has been done and the reason why this was necessary,
  - for each signal or signal pair analysed, an illustration(s) showing the performance of the algorithm(s) you have implemented. Depending on the situation, several graphs may (and where appropriate, should) be displayed in one illustration window (in different colours, or with different types of curves, e. g. – solid, dashed, dotted, etc.). Also provide a visualisation of at least one (possibly all) of the original (pre-processed) data (or, if the data are very large, a fragment of it) (if you are processing, for example, an audio signal, it may be more meaningful to show a graph of the spectrum of the signal rather than the signal itself). All illustrations must be numbered and must have a caption (a short text at the bottom of the illustration) mentioning the number of the signal in the list of signals analysed, the specific numerical values of all the parameters of the algorithm, a brief interpretation of the result seen (what conclusions can be drawn from this illustration, possibly comparing it with other illustrations, indicating the numbers of the compared illustrations). Pay attention to the proper layout of the illustrations – the size of the illustrations should not be unnecessarily reduced (70% – 100% of the page width should be used, 1 – 3 illustrations per line), the curves of the graphs must be sufficiently thick, and the illustration must not be overloaded with too many curves (two or more illustrations may be included instead of one if necessary), axis variables and gradations must be clear and easy to read (in the final PDF version of the Report, all text and figures within the illustration must be at least the same font size as the text of the Report), the dimensions of the axis variables must be mentioned (e.g. seconds, years, etc., a common mistake when the abscissa axis does not contain the time of the signal, but the indexes of an array of values of a discrete signal) or it must be stated that a specific quantity has no dimension. At the same time, in the preparation of this Report *you will gain the experience you are going to need to write your Thesis*. If necessary, you can also present additional results as numerical tables (although this is unlikely to be necessary) or in an Appendix to the Report.
  - the conclusions of the research carried out. If the algorithm(s) has parameters, write a commentary on the meaning of each of them and what exactly are their numerical values (or ranges) for a particular signal you found most meaningful, quoting the relevant illustrations. Please also write anything you consider significant – based on your experience in implementing this algorithm or algorithms. How can this algorithm be useful (in practical applications) and how do the illustrations confirm this? Are there any technical nuances in its implementation? Do you have any negative experiences – for example, for certain signals it is difficult to choose the parameters of the algorithm or to interpret the results? Mention this too – remember that negative results (provided, of course, that there are positive results) only enrich any study.

  - At the end of the Report (in an Appendix), include a listing of the essential (in your opinion) fragment or fragments of the code of the program you have written. Please devote 1 – 5 pages to this. These fragments should be 100% identical to the code you will compile and run to demonstrate (during the discussion) the program's performance.

- Demonstrate how your program works with various algorithm parameters, briefly describe your experience (what problems you may have encountered and how they were solved) and answer the tutor's questions;

- Provide the source code of the program you have developed (no compilation instructions are required, source code is requested only for plagiarism control purposes) and a PDF file of the report.

When writing the code, follow good programming style – the code must be clearly structured, the names of variables, functions and procedures must be systematic and meaningful, and key points should be commented on. This will also effect your assignment's grade.

Optional *reccomendation*: you can use LaTeX(a very user-friendly language for technical writing) for your report (and in the future also for your Thesis) and the LaTeXtemplate provided by the Computer Modelling Master study programme for Master Thesis (see "Darbų LaTex šablonas"):
`https://mif.vu.lt/lt3/studijos/magistrantams#kompiuterinis-modeliavimas`.
This is not compulsory, but it can be a very useful and fun experience!

Additional References

1. Rob J. Hyndman, Time Series Data Library,
   `https://pkg.yangzhuoranyang.com/tsdl`

2. Eastern Michigan University, Free Data Sources,
   `https://guides.emich.edu/data/free-data`

3. DataHub, Search for DataSets,
   `https://datahub.io/search`

4. PhysioNet. The Research Resource for Complex Physiologic Signals,
   `http://physionet.org`

5. Gareth Janacek, Practical Time Series Data for Analysis,
   `http://www.uea.ac.uk/~gj/book/data/datalist.html`

6. Sound Jay Sound Data Library,
   `http://www.soundjay.com`

7. Tadas Meškauskas, Signal Analysis and Processing,
   `http://mif.vu.lt/~meska/sap2024`

8. LaTeX, `http://en.wikipedia.org/wiki/LaTeX` (and many other references and examples)