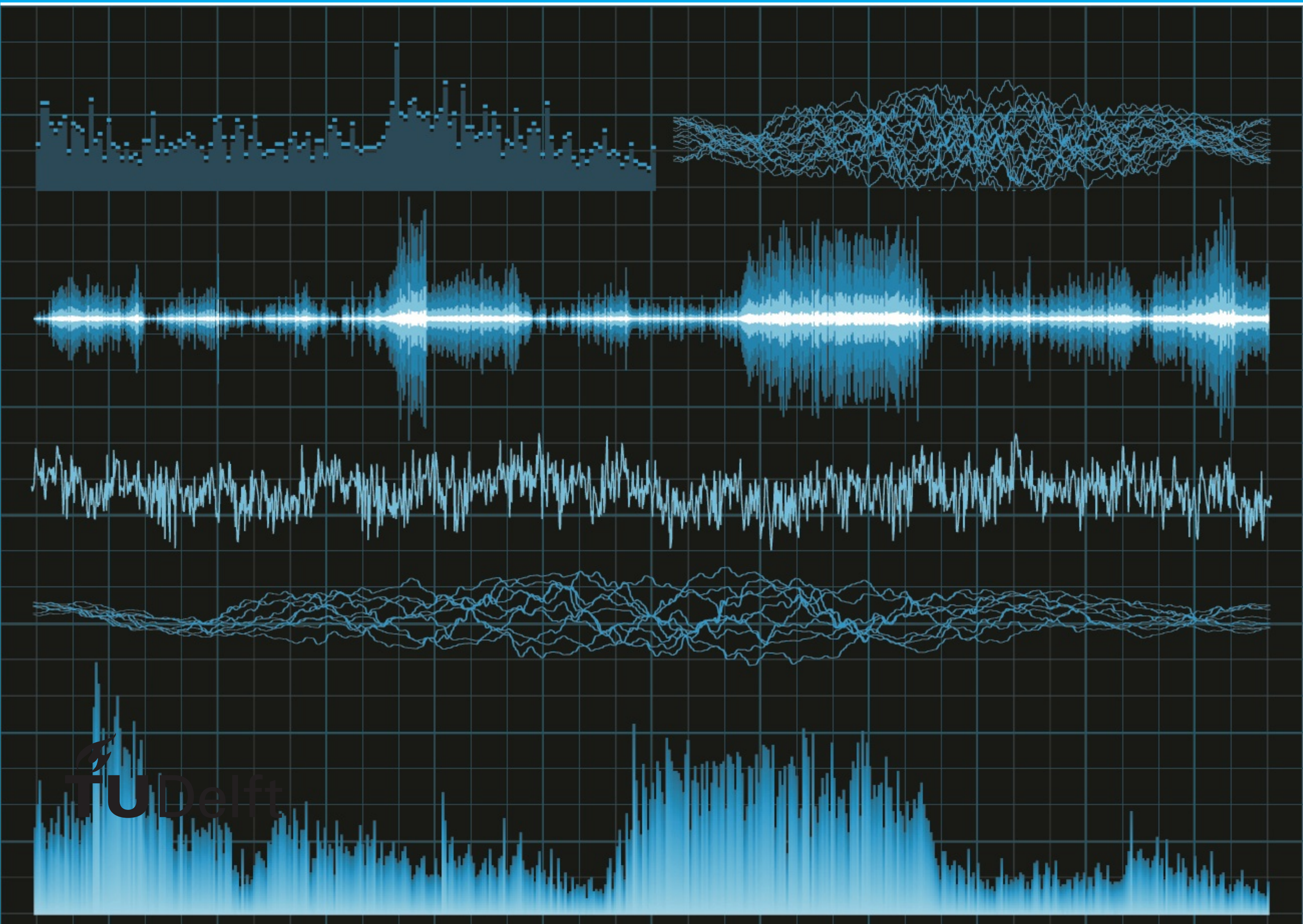
The Electrical Network Frequency Criterion

Determining The Time And Location Of Digital Recordings

T. Baksteen

Delft University of Technology



The Electrical Network Frequency Criterion

Determining The Time And Location Of Digital Recordings

by

T. Baksteen

in partial fulfillment of the requirements for the degree of

Master of Science
in Applied Mathematics

at the Delft University of Technology,

to be defended publicly on Wednesday November 11, 2015 at 15:30.

Student Number:	1519166	
Supervisor:	Dr. N.V. Budko	
Thesis committee:	Prof. dr. ir. C Vuik	Delft University of Technology
	Dr. N.V. Budko	Delft University of Technology
	Dr. J.W. van der Woude	Delft University of Technology
	T. van Mechelen	NCIM-groep

Preface

The topic of this thesis is the Electrical Network Frequency Criterion as a method to determine both the time of recording and the location of an audio file when no or corrupted meta-data is available. This may be possible when the recording equipment is mains powered or when it is situated in an electromagnetic field. Then the voltage signal is unintentionally recorded as well and its frequency is uniquely changing over time and location. It is found that a time of recording estimation can be done under reasonable circumstances and that a rough location estimation of the recording is technically possible but will have a very limited applicability.

The research topic was provided by NCIM-groep who has employees working at the NFI (the Dutch forensic institute). This topic is particularly of interest to forensic researchers for its applications in audio tempering detection. The validity of digital evidence can be checked by analyzing existing ENF traces. Audio cuts can be observed and a time of recording can be established to confirm or reject an alibi.

My thanks goes out to NCIM-groep who gave me the opportunity to do this great project at their company and to my thesis supervisors Neil Budko and Thomas van Mechelen.

T. Baksteen
Delft, November 2015

Contents

1 Introduction

The amount of digital audio and video recordings has been increasing rapidly over the last few decades, and with that also tools to modify its content and meta-data like time of creation. Some edits seem unnoticeable and the need to verify the authenticity of digital video and audio emerges. In legal cases recordings are used to indicate a person's presence at a certain location at a certain time. One could forge an alibi by editing footage. For this reason the Electrical Network Frequency (ENF) Criterion has been developed to authenticate such recordings. Under certain circumstances this technique allows for detecting cuts in the audio (or video) and determining the time of recording. The frequency of the power grid is used as a unique fingerprint that could unintentionally be saved in the audio track. Also video recordings without audio can contain the ENF information in brightness oscillations of the frames. One of the first articles to address the possibility of validating digital recordings using the ENF is written bij Catalin Grigoras in 2003 [8]. The article was published in 2005. Since then the knowledge about the topic of the ENF criterion is rapidly expanding.

The key observation is that the ENF fluctuates around its nominal value in a semi-random way. In Europe the power grid runs on 50 Hz and the ENF is controlled to be within a band of 200 mHz around this mean value. Each power withdrawal in Europe effects the frequency, hence there are so many components influencing the ENF that the ENF signal becomes like a white noise around the 50 Hz value.

Secondly, the ENF is nearly identical on each location of the same interconnected power grid. The power generators rotate at a certain speed, producing the electricity at a frequency of approximately 50 Hz. These generators are synchronous machines, meaning that they are intended to produce at equal frequencies. The ENF signal is therefore almost identical over the entire grid. These properties allow for a timestamping of some digital audio files anywhere in a power grid when there is an ENF database available for pattern matching. An other powerful application of the ENF criterion is to detect edits in the audio recording [8]. These may be visible as large frequency discontinuities.

This report contains an overview of the progress in the research of the ENF criterion. The report starts with some important properties of the ENF and how to capture the ENF of the power grid. Next, all techniques for the ENF extraction from audio and video are treated. Some different techniques are proposed in order to reduce computation time and increase ENF estimation accuracy. The most used database matching procedure in literature is explained and verified. In this chapter a few aspects of the matching procedure are changed and an illustrative example is given. A rather new topic within the field of the ENF criterion is the application to digital video recordings. Using the properties of fluorescent light and the rolling shutter effect of CMOS sensors the possibilities of video ENF extraction are explored.

The main focus of this report is to answer the question whether it is also possible to accurately determine the location of a (video or audio) recording when the ENF information is available. The first attempt to localize a recording was done in 2013 [7]. Inspired by their findings a theoretical framework is proposed to gain insight into the observation that the ENF signal is not exactly equal at all locations of an interconnected grid. Next three different localization methods are compared. Finally the new findings are discussed and recommendations for further research are given.

The research questions of the thesis are the following:

1. Can the ENF extraction accuracy be improved by decimating the voltage signal to a higher sampling rate?
2. Can the ENF extraction be accelerated by using the DFT rather than the FFT?
3. Which frequency resolution is to be chosen in combination with quadratic interpolation?
4. Can the ENF signal also be accurately extracted from video recordings?
5. Is it possible to accurately estimate the location at which the recording was made?

2
The ENF signal

The ENF signal, $f(x,t)$ consists of the most prominent frequency value of AC oscillations at time t measured at some location x. In order to gain some understanding of the ENF signal it is important to explore the basic workings of the power grid.

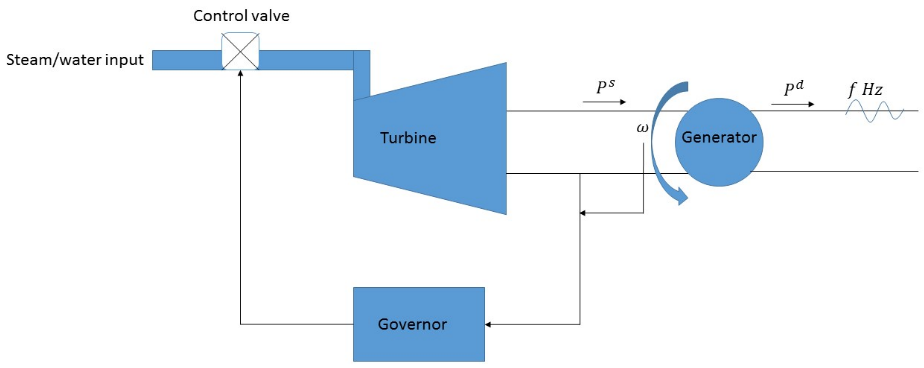


Figure 2.1: How the frequency is controlled

Figure 2.1 shows the schematics of electricity generation and power control. Turbines deliver kinetic energy to the generators, which convert it to electrical power. The supplied power (P^s) makes the generator rotate so that kinetic energy is converted into electrical energy to meet the power demand (P^d) of the grid. The angular velocity of the generator is denoted by ω which is directly related to the voltage frequency f, the ENF.

The frequency is constant whenever P^s equals P^d but since the power demand changes with every consumer switching a light on or off or use electric machines, the demand P^d is far from constant. The power demand must be met instantly. Since a change in power supply takes some time, the instantaneous power discrepancy is compensated for by the generators' rotational energy. So an increase in power demand results in a temporary decrease in ENF and vice versa [14]. The frequency may freely fluctuate within 20 mHz below and above the nominal frequency of 50 Hz. Outside this narrow band the primary control is activated meaning that a governor will control the turbine input to adjust P^s to stabilize the ENF. In case of more severe frequency deviations, over 200 Hz, more drastic measures are taken like intentionally shutting down generators causing a power outage to prevent more serious damage from happening.

The energy market also plays a big role in the power supply to the generators. Each hour the right to sell power to generators is auctioned but the allocation of the required power amongst the generators has to be decided on beforehand. Therefore the load must be predicted in advance each hour. If an increase in load is expected, the turbine power will be increased in advance resulting in an increased frequency to anticipate and prevent the coming frequency drop. Since there are many generators and turbines all over the interconnected grid, this applies to all generators. The generators are synchronous machines meaning that the frequency of the electricity produced by all generators is, if all runs correctly, nearly identical.

As for the dynamics of the ENF there is a simple equation describing the relation between power and rotation speed. Let f_i denote the frequency of the electric signal produced by generator i and let P_i^d and P_i^s be its power demand and power supply respectively. Furthermore we denote the nominal rotating energy $\frac{1}{2}J\omega_0^2$ of generator i by H_i. Here J_i is the moment of inertia of the ith generator.

The following approximate relation holds [14]:

$$\frac{df_i}{dt} = \frac{f_0}{2H_i}(P_i^s - P_i^d) \tag{2.1}$$

It can be deduced that indeed the frequency increases whenever the supply exceeds the demand. The surplus is stored as kinetic energy in the rotation of the generators and a shortage is drawn from the rotation. The relation is approximately linear.

The control mechanism pulls the ENF back to its nominal value f_0 by adjusting the power supply linearly to the frequency deficit, whenever the frequency deviates more than 20 mHz from its mean value. α_i is a constant related to the contribution of generator i to the grid's power. This control mechanism is called the droop speed control and the power supply dynamics for each generator i is given by:

$$\frac{dP_i^s}{dt} = \begin{cases} -\alpha_i(f_i - {f_0}_i) & \text{if } |f_i - {f_0}_i| \geq 20 \text{ mHz} \\ 0 & \text{if } |f_i - {f_0}_i| < 20 \text{ mHz} \end{cases} \tag{2.2}$$

2.1. Randomness of the ENF signal

As mentioned in the introduction, one of the goals of this report is to timestamp digital recordings using the ENF signal. The ENF signal of a recording is a list of frequency values around 50 Hz. This vector is compared to a reference database consisting of frequency values recorded over a long period of time. When the ENF vector of the recording is a close match to a specific database segment, then it can be concluded that the recording originates from the same time as that database segment, and the time of recording has been found. This procedure will be discussed in more detail later, but for now it is important to note that for this approach to work, different database segments must be unique. If this were not the case, then the matching may not yield a unique solution and the timestamp will be very uncertain.

We know that the frequency changes with power supply and demand, which are highly irregular quantities. There is, however, some form of periodicity present in the ENF. This periodicity is daily, weekly and even yearly and some of the effects are shown in Figure 2.2:

This carpet plot shows low frequency values in blue and high frequencies in red. Horizontally are the days number from September 1 to December 31 2014 and vertically the time of the day, starting at midnight and increasing 4 seconds per index. So element [20,5400] shows the frequency value on September 20 at 6 a.m. There are remarkable grid-like structures visible in the figure. The horizontally displayed patterns are lines at each multiple of 900 which is at each hour. The explanation for this is that each hour the generators rotation speed is adjusted to meet the demand forecast for the coming hour. For example the bright red line at index 5400 (and less bright at 4500 and 6300). This reveals a quick and short frequency increase at 6 a.m. daily. The generators speed is increased because many people in Europe wake up around that time and turn on the lights and other electrical machines. This increase in power demand decreases the spinning frequency which causes the ENF to even decline from the red to the blue values. A similar effect is visible in the afternoon at row 15300 and 16200, corresponding to 5 and 6 p.m. when many working people arrive at home and turn on their electrical devices. In the evening however blue lines are recognizable, indicating a that the next hour the power demand will decrease rapidly. This may be due to people going to bed or turning off their ovens and microwaves.

Vertically, yellow lines are visible which show a more constant frequency during the day. They show the weekend days on which people more gradually get up and go to sleep. Hence, the ENF also shows periodicity on different scales. On October 25 in the night a bright red vertical line is the result of daylight saving time which was applied in Europe. It is not really clear why this causes the network frequency to rise, but such an event is clearly visible in the frequency plots.

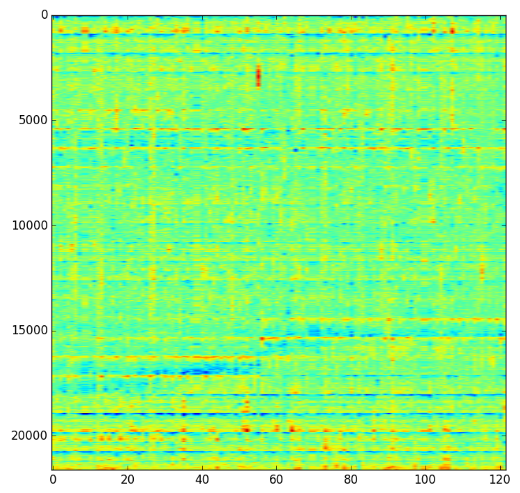


Figure 2.2: ENF variation over 3 autumn months

Although in all ENF literature it is stated that the frequency fluctuates randomly around its mean, there are some very characteristic patterns like a sudden ENF increase each day at 6 am, which drops to under the nominal frequency later that hour. This behavior is certainly not random, but the question that should be asked is whether the ENF signal of a certain duration is unique. This can be tested by taking an ENF database and calculating the error between two randomly chosen ENF vectors. To be able to get a unique timestamp on an extracted ENF signal, no two different database segments can be 'very much alike'. What that means exactly is discussed in section 4.1.

From the above shown data set, one million random segments of 5 minutes are compared and the 1-norm of the difference of the two vectors are shown in a histogram. In all those combinations of segments there are no two with a smaller sum of errors than 1.5 which is enough to conclude that the timestamping problem has a unique solution. This topic will be treated in section 4.2.

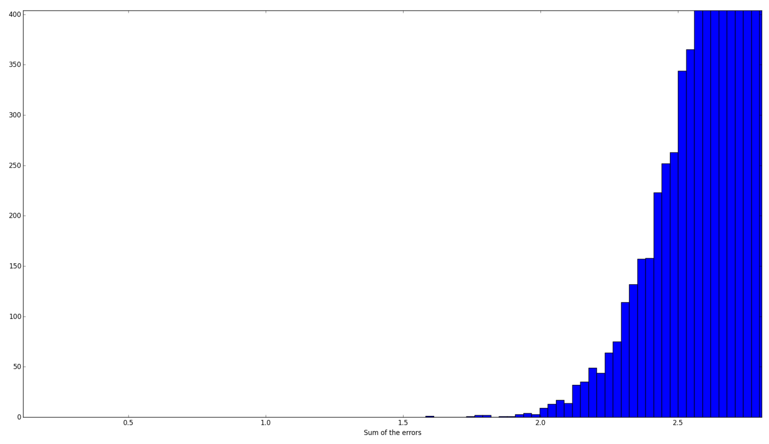


Figure 2.3: Errors of two different ENF vectors of 5 minutes

3 Obtaining the ENF

When mains powered equipment is used to create an audio file, the 50 Hz signal of the power grid could unintentionally find its way into the audio track, in the form of a 50 Hz sound. This frequency is almost undetectable for the human ear but it leaves a digital fingerprint on the file. This voltage signal can enter the audio track using battery powered electronics, when the microphone is in a electromagnetic field.

In this section the entire procedure from audio file to ENF signal will be discussed. Many techniques are used together to isolate the ±50 Hz signal and accurately determine which frequency is most prominently present in each second of audio signal. Most authors of articles about the ENF criterion have agreed upon a certain procedure with specific parameters. That procedure will partly be used in this report, with some adjustments to increase its computation speed and accuracy. The procedure consists of a decimation of audio signal to decrease the computational work, After that a so called bandpass filter is applied which isolates all signal components within a certain range around 50 Hz. All sounds outside this bandwidth is noise for our purposes. The resulting signal should mainly consist of the voltage signal coming from the power grid. Then the signal is divided into overlapping segments, each 10 second long and these segments are transformed to the frequency domain. The maximum of the frequency domain function is associated with the most prominent frequency value of that segment. This frequency value is chosen to be the ENF of the first second of the corresponding segment.

The first step is to downsample the given audio file. This is not actually required to extract the frequencies but it significantly speeds up the process and saves memory. 44.1 kHz is a common sampling rate for audio. However, this is far more than necessary. Section 3.4.1 performs numerical experiments to deduce a minimum required sampling rate for sufficiently high accuracy. Here 1 kHz shows to be sufficient, meaning that ENF estimates do not suffer from the decimation. However, when decimating the signal to lower than 1 kHz some accuracy loss may occur. Especially decreasing sampling rate to under 100 Hz is unadvised. 100 Hz is the Nyquist rate for a 50 Hz bandpass filtered signal, meaning that at least a 100 Hz sampling rate is required to detect a 50 Hz signal in a straightforward way. A low sample rate causes aliasing of the actual signal, which negatively influences the ENF estimation accuracy. Later on in this report ENF extraction from video is examined, where the topic of signal aliases will be treated more thoroughly.

Now a 1 kHz audio file remains consisting of all sorts of background noise, intentionally recorded audio of all frequencies and of course the ENF of about 50 Hz. For the ENF study all frequencies outside the bandwidth 49.5-50.5 are noise since the ENF will never deviate more than 0.5 Hz from the ideal value of 50. A bandwidth filter is applied to attenuate all sounds originating from sources producing other frequencies. However, all acoustic noise at 50 Hz still exists in the signal after the filter. Unfortunately there is not much that can be done about that. The transformation of the voltage signal from the time domain to the frequency domain is next. This is commonly done using a discrete Short Time Fourier Transformation.

3.1. Short Time Fourier Transform with zero-padding

The Short Time Fourier Transform (STFT) is a time domain to frequency domain transformation based on the Fast Fourier Transform (FFT), which is a modified version of the more basic Discrete Fourier Transform (DFT). The time domain signal is divided into several overlapping segments by multiplying the total signal with a sliding window function w. For this report only the discrete-time STFT is relevant.

Given our discrete signal $x[n]$ and a chosen symmetric window function $w[n]$ of finite length N, the STFT of $x[n]$ is given by the complex coefficients:

$$X(m,k) = \sum_{n=-\infty}^{\infty} x[n]w[n-m]e^{-i2\pi kn/N} \quad k = 0,1,\ldots N-1 \tag{3.1}$$

Here m determines the position of the window function and hence the time segment to be transformed. The time step, which is the time resolution, is set to one second in most literature, and therefore also in this report. k is the frequency bin corresponding to frequencies ranging from 0 to half the sampling rate (the Nyquist frequency). Equation 3.1 actually describes a size N Discrete Fourier Transform (DFT) of length N, which returns the exact same output as the FFT. The main difference is that the DFT is an $\mathcal{O}(n^2)$ algorithm where the FFT uses a divide and conquer algorithm to recursively divide the DFT into smaller segments, reducing the computational complexity to $\mathcal{O}(n\log_2 n)$. A STFT produces N equidistant Fourier coefficients for each segment m. In this report a window size corresponding to 10 seconds ($N = 10\cdot$ sampling rate) is commonly used, and the magnitude of each (complex) Fourier coefficient is considered.

The voltage signal, which is used as input for the STFT, is assumed to be approximately sinusoidal. The window function used in this report is the rectangular window. Figure 3.1 illustrates some properties of the Fourier Transform applied to a sine wave with a rectangular window. The voltage signal in the upper left needs to be transformed to the spectrum in the bottom right. Starting with the sine wave displayed in the upper left figure, the goal is to accurately estimate the top of the Fourier spectrum shown in the lower right image. In the time domain a rectangular window, the indicator function on a 10 seconds interval, is multiplied with the voltage signal. This results in a 10 second signal which can be transformed to the frequency domain using the FFT. It can be seen that a multiplication in the time domain corresponds to a convolution in the frequency domain.

In order to obtain an ENF estimate for each second of signal, the indicator window function is shifted by one second each step. When the signal is downsampled to 1 kHz that will be a shift of 1000 to represent the next second. So for the length N of the window this report uses ten times the sampling rate, i.e. 10,000. This means that the windows are 9 seconds overlapping. Overlapping of the windows is a common practice in signal processing since it reduces boundary artifacts and increases the continuity of the estimated ENF signal. Later on there will be more about window overlapping and continuity of the ENF signal. There is a wide variety of window functions to choose from. Most windows functions taper off at both ends to smoothen the frequency spectrum and decrease the height of the side lobes. There exists a trade-off in windowing properties. A window either has a narrow main lobe (which is 2 Hz in the example figure), or low side lobe amplitudes. In this case a narrow main lobe is favorable, because a pointier lobe allows for more accurate peak estimation. The height of the side lobes are irrelevant for this application.

10 seconds of perfect 50 Hz signal will have a discrete set of points from $sinc(f-50)$ as Fourier Transform. However, when studying real voltage signals the frequency will vary over time and the spectrum will not be an exact sinc function. In this variety of frequencies there will be one frequency f with the strongest presence in that segment of signal. The problem is that the frequency resolution is limited by the length of the signal from the time domain.

The following relation between time and frequency resolution holds: $\Delta f = \frac{1}{T}$ where T is the length of the signal segment in seconds [2]. So when Fourier Transforming 10 seconds of voltage signal, the frequency resolution will be $\frac{1}{10}$ Hz. This resolution is insufficient because frequency deviations in the order of 1 mHz need to be discerned. Simply increasing the window length to 1000 seconds if not an option because then the window overlap is so large that all ENF detail is lost. There are however artificial ways to increase frequency resolution. zero-padding is often called frequency domain interpolation [13]. It increases the frequency resolution by interpolating between existing frequency bins. zero-padding basically involves adding a certain number of zeros to the windowed signal segment. This increases the length of the segment, allowing the FFT to return more frequency magnitudes.

To improve the frequency resolution to 1 mHz this way, a zero-padding factor of 99 is needed. 99 times the original length of the windowed signal is the amount of zero's is then to be added to the signal. This is

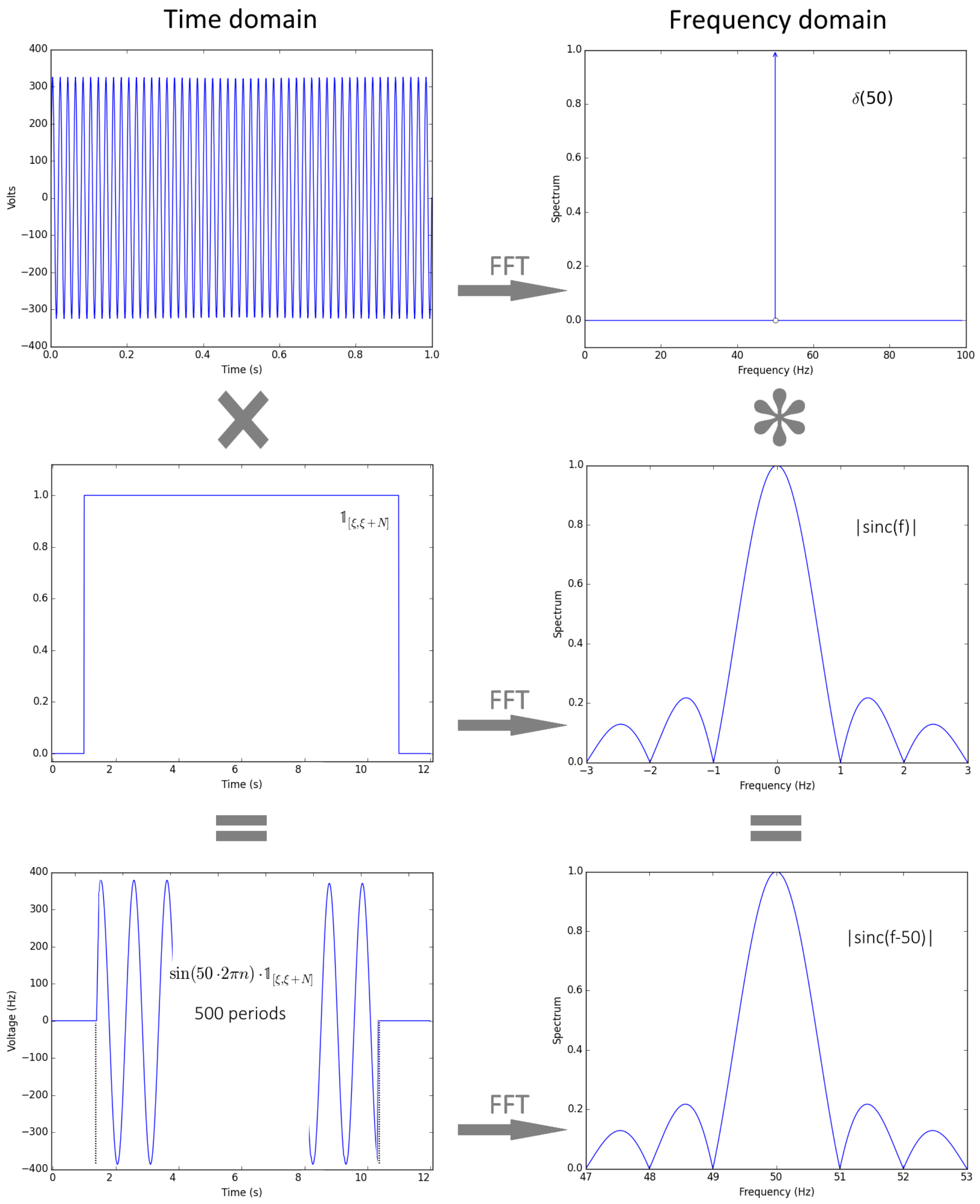


Figure 3.1: Transformation from the time-domain to the frequency-domain using a rectangular window

however not feasible due to memory and computation time. Some other technique should be used additionally. A good feasible ENF estimation can be obtained by a combination of mild zero-padding and quadratic interpolation in the power spectrum.

3.1.1. Quadratic Interpolation

Rife and Boorstyn show that when using a rectangular window, one can obtain an exact spectral peak estimator as the maximum likelihood estimator for a single sinusoid with additive white Gaussian noise by zero-padding to infinity [13]. The maximum likelihood estimator will asymptotically achieve the Cramer-Rao lower bound on the variance of the peak frequency estimator. Performing the STFT with too much zero-padding becomes computationally expensive and requires a lot of memory. So the commonly used spectral peak estimator is one which requires less memory, the Quadratically Interpolated FFT (QIFFT). The top of the main lobe of a sinc function is quite similar to a quadratic function. This can easily be deducted from the Taylor series of $sinc(f-50)$ around $f=50$ which is given by: $1-\frac{(f-50)^2}{6}+\frac{(f-50)^4}{120}+\mathcal{O}(f-50)^6)$. Decimating the series after the quadratic term leaves an error in the order of $\frac{(f-50)^4}{120}$ which is negligible.

So for the quadratic peak estimation let m denote the frequency bin with the largest FFT value. The most present frequency is then known to be in in one of the bins $m-1, m$ and $m+1$. To estimate the ENF more accurately than stating that it is somewhere in this vicinity, the quadratic interpolation is applied on those three FFT values.

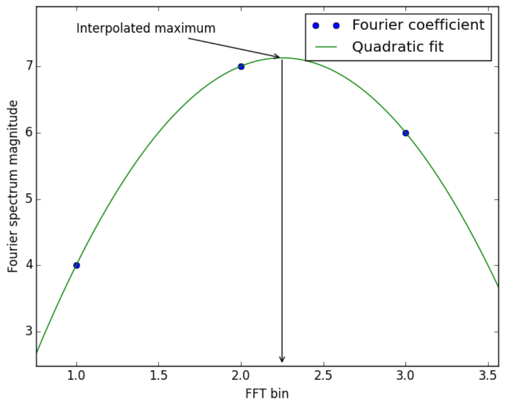


Figure 3.2: Quadratic interpolation on the three top FFT values

Given $\text{FFT}(m-1)=\alpha, \quad \text{FFT}(m)=\beta, \quad \text{FFT}(m+1)=\gamma$, the relative position of the interpolated top can be calculated as:

$$p=\frac{1}{2}\cdot\frac{\alpha-\gamma}{\alpha-2\beta+\gamma}$$

Meaning that the best estimation of the ENF value of that segment is located at 'bin' $m+p$. This procedure is computationally cheap and the use of QI reduces the need of zero-padding to a factor of 4 in stead of 999 which would be needed without the interpolation. Since a 10 second rectangular window is used in this report, the three values are spaced at 20 mHz distance. The quadratic interpolated top is about 0.1 mHz accurate which is more than enough accuracy for timestamping.

3.2. DFT

In this report an alternative approach for the transformation to the frequency domain is proposed. Surprisingly the more basic DFT is a faster ENF estimation technique. As stated in the previous section, the STFT is a sliding window FFT meaning that per second of signal an $\mathcal{O}(n\log_2 n)$ algorithm is performed to generate a large number of Fourier coefficients. The FFT approach uses a zero-padding factor of 4 and window overlap of 9 and the signal will be downsampled to at least 1 kHz. The STFT calculates all coefficients from -500 to 500 Hz, spaced at 20 mHz distance.Therefore the number of Fourier coefficients calculated in each step using the FFT is $50,000$. However, only the largest coefficient and its two direct neighbors will be used. Also, the index of the largest coefficient is already known to be within a very narrow band, since the ENF is always close to 50 Hz. Using this information the conventional DFT can be used to save memory and computation time. Even though the classic DFT is an $\mathcal{O}(n^2)$ algorithm, it is faster because the amount of coefficients to be calculated can be chosen explicitly and kept under 50. Also, no zero-padding is needed in this approach.

The DFT also is a linear transformation from the time domain to the frequency domain. Let $x[n]$ be the signal of length N. The computation of k chosen Fourier coefficients is given by the following k linear equations:

$$\begin{bmatrix} 1 & e^{\frac{-2\pi i\cdot 1\cdot f_1}{s}} & e^{\frac{-2\pi i\cdot 2\cdot f_1}{s}} & \cdots & e^{\frac{-2\pi i\cdot (N-1)\cdot f_1}{s}} \\ 1 & e^{\frac{-2\pi i\cdot 1\cdot f_2}{s}} & e^{\frac{-2\pi i\cdot 2\cdot f_2}{s}} & \cdots & e^{\frac{-2\pi i\cdot (N-1)\cdot f_2}{s}} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & e^{\frac{-2\pi i\cdot 1\cdot f_k}{s}} & e^{\frac{-2\pi i\cdot 2\cdot f_k}{s}} & \cdots & e^{\frac{-2\pi i\cdot (N-1)\cdot f_k}{s}} \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ \vdots \\ x[N-1] \end{bmatrix} = \begin{bmatrix} X[f_1] \\ X[f_2] \\ \vdots \\ X[f_k] \end{bmatrix} \tag{3.2}$$

The advantages of this approach can be summed op:

- Any amount of different frequency components can be calculated
- Frequency bins do not need to be equidistant.
- No zero-padding required

To estimate the first ENF value of a signal the frequency resolution can be chosen to be 10 mHz and the bins can be set to $[49.80, 49.81, \cdots, 50.19, 50.20]$ Hz after which the Quadratic Interpolation is applied. The FFT would calculate $2\cdot 10^5$ values but this small DFT only computes 41 values. For the next ENF values, even less coefficients have to be computed. Empirical studies have shown that the ENF does not change more than 40 mHz in one second, so knowing $f[i-1]$, the search band for $f[i]$ is narrowed down to 80 mHz. Let for instance $f[i-1] = 49.947$ Hz, then the DFT matrix consists of only 10 rows corresponding to the frequencies $[49.90, 49.91, \cdots, 49.99]$.

The main disadvantage is the higher complexity of $\mathcal{O}(n^2)$. Normally, the full DFT matrix, calculating the same frequency components as the FFT, is a square symmetrical matrix. The symmetry allows for a faster computation, which is how the FFT was developed. However, now that only a few coefficients are calculated the $k \times N$ DFT matrix is not square anymore since k is chosen such that $k \ll N$.

To compare the performance of both methods the ENF values of different length signals are calculated:

Signal length (s)	450	900	3600	7200
DFT duration (s)	4.2	7.8	23.3	44.2
FFT duration (s)	35.4	76.9	223.1	475.2

Table 3.1: DFT versus FFT

For all computations in this report Python has been used. For the code, please view the appendix. Both methods provide the exact same ENF values. However, the highly optimized FFT package takes about ten times the computation time the simple DFT needs. The longer the signal is, the more advantageous the DFT becomes. The DFT matrix has to be computed only once, after that only relatively small matrix vector multiplications are performed.

3.3. Noise

In the study of ENF, noise is considered to be any sound other than the approximately 50 Hz voltage signal of the power grid. Since all relevant recordings do not intend to record the ENF, the sound to noise ratio (SNR) is typically very low, which can result in bad ENF estimations. It is said that noise is the biggest limitation of the ENF criterion [4]. Hence, noise reduction is the main topic in ENF literature. Many techniques have been proposed for noise reduction like a threshold dependent median filtering [4], using multiple ENF harmonics [3] or autocorrelation models [5], and increasing the STFT window overlap. Only the topic of window overlap will be briefly discussed. This report will emphasize other topics such as location estimation.

To illustrate the effect of different STFT overlaps, Figures 3.3 to 3.5 show the ENF signal estimation of one recording using different window overlapping values.

All these images show an ENF signal estimation of 2 minutes of audio signal with 1 second resolution. In the first image, the ENF estimation using a window size of 5 seconds is shown. Note that this means that the windows overlap by 4 seconds. The signal is much more jagged than what the real ENF signal without noise would be. One property of the ENF is that the frequency value from 1 second to the next will not jump more than 40 mHz. This is due to the inertia of the generators, meaning that they will not all change rotational speed that fast in 1 second. So Figure 3.3 does not preview an accurate ENF signal. To smoothen this signal Cooper suggests to increase the window overlap.

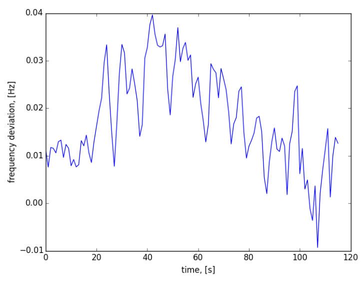


Figure 3.3: 4 sec overlap

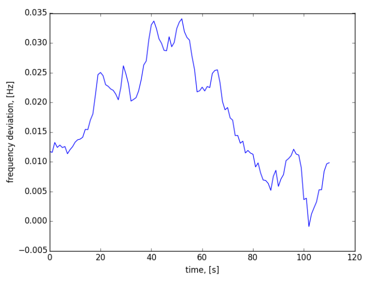


Figure 3.4: 9 sec overlap

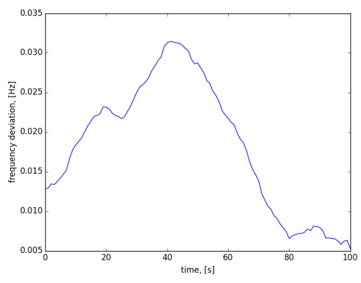


Figure 3.5: 19 sec overlap

Figure 3.4 shows the ENF estimation using a 9 second overlap in segments. As a result the frequencies are more plausible than before and a less jagged signal is plotted. One could even further increase the overlap to 19 seconds to get the result displayed in Figure 3.5. No article has proposed a convention concerning the overlap yet, neither has there been claims about the overlap factor producing the best results. 1 to 14 seconds overlap have all been used, where the latter works better for visual matching with the true ENF signal. In this report 9 seconds overlap will be used. This amount of overlap often shows the right measure of continuity in the ENF signal and still maintains enough detail.

A bandpass filter is also applied by most. Its purpose is to attenuate all noise outside the 49.5-50.5 Hz band. This seems irrelevant because only the Fourier coefficients of the values within that band are condsidered. However, even a 55 Hz signal will influence the spectrum peak estimation. One of its side lobes will be added to the main lobe of the actual voltage spectrum, possibly yielding inaccurate ENF estimations.

A new noise countering technique has been developed in this report which is a post-processing Fourier based smoothing approach. The above figures clearly show that increasing the window size smoothens the ENF signal. Figure 5.6, later on in this report, shows that also with the same window size, the 'continuity' of the ENF signal is an indicator of the ENF estimation accuracy. When the SNR is high and a window size of 10 second is chosen, the ENF signal is 'smooth'. To quantify this term, the frequency spectrum of the ENF signal is studied. That is the Fourier transform of the signal consisting of maxima of Fourier spectra. When the spectrum of the ENF signal of a noise-free audio signal is studied, virtually no frequency components of larger than $1/windowsize$ Hz are observed. ENF signals from noisy audio are more jagged and do contain high frequency ENF oscillations.

A way to post-process bad ENF vectors is to remove all high frequency ENF oscillations according to the window size criterion. Using this approach, the error with respect to the noise-free ENF can be reduced with 10-20%, and database matching, which is discussed in Chapter 4, will be improved.

3.4. An ENF extraction procedure

One of the most important contributing articles about the ENF criterion is written by Alan J. Cooper who works as a forensic researcher. In this article a standard procedure to extract the ENF signal from an audio file is proposed. This procedure was then adopted by most others in the ENF research field. His procedure will be discussed in this section and after that a modified procedure is explicated to increase accuracy and save computation time.

Coopers choice of signal decimation to 300 Hz is not supported by any theory or experiments other than the fact that 300 Hz is well above the Nyquist frequency (twice the frequency of interest). However, in the next section it is shown that this likely leads to an ENF error of 0.1 mHz. This may influence the timestamping accuracy a bit, even though 0.1 mHz is small compared to the ENF errors caused by a low SNR. But this report is concerned with a location estimation of a voltage signal and for that all frequencies need to be accurate up to 0.1 mHz. Hence, every small error should be avoided.

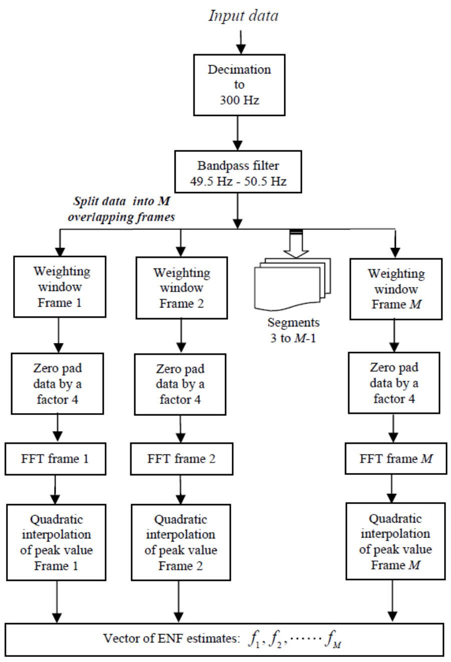

Input data
Decimation to 300 Hz
Bandpass filter 49.5 Hz - 50.5 Hz
Split data into M overlapping frames
Weighting window Frame 1
Weighting window Frame 2
Segments 3 to M-1
Weighting window Frame M
Zero pad data by a factor 4
FFT frame 1
FFT frame 2
FFT frame M
Quadratic interpolation of peak value Frame 1
Quadratic interpolation of peak value Frame 2
Quadratic interpolation of peak value Frame M
Vector of ENF estimates: $f_1, f_2, \cdots f_M$

Figure 3.6: Coopers ENF extraction procedure

The procedure of the bandpass filter, followed by the rectangular windowing with 9 seconds overlap has been discussed. Because most authors use the FFT, there is a need to use zero-padding. Zero-padding by a factor 4 results in 20 mHz frequency resolution, before the QI is applied. This chapter contains experiments to test the accuracy of a 20 mHz Fourier transform resolution. Using the DFT in stead of the FFT it is easy to adjust the frequency resolution to obtain a sufficient accuracy. The quadratic interpolation will be kept the same in the proposed modified procedure.

3.4.1. Sampling rate

As stated before, it is common practice to downsample the audio signal before applying the Fourier transformation, to save computational work [4]. The loss of information would not influence the ENF signal estimation accuracy. To verify this, a minute of sound with a reasonable SNR is recorded at 32768 Hz. This signal is then down-sampled to all different two powers down to 256 Hz. The ENF value of the same segment, with the different decimation factors is plotted in Figures 3.7 and 3.8.

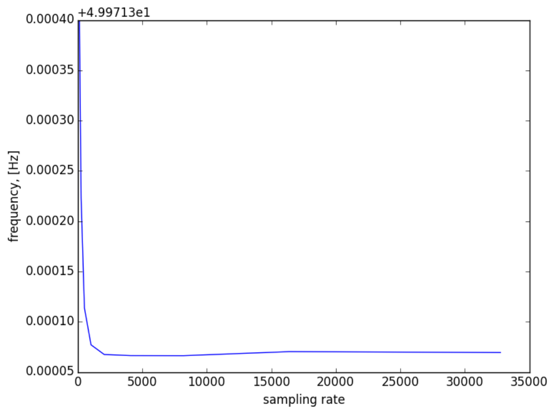


Figure 3.7: ENF convergence: sampling rate

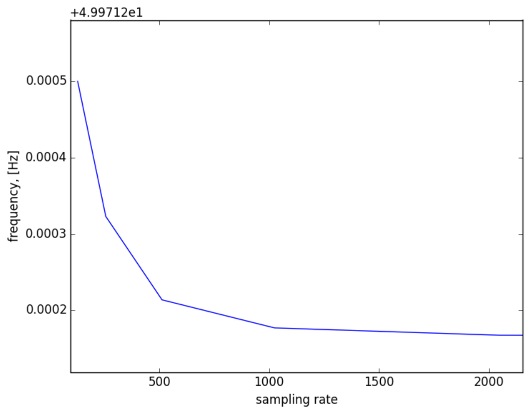


Figure 3.8: Zoomed in ENF convergence

The behavior shown in these figures is typical and applies to ENF estimations of audio files in general. It can be seen that the 300 Hz proposed by Cooper is adequate if a 0.1 mHz error is considered negligible.

However, a sampling rate of at least 1 kHz is advised for optimal accuracy. This may well be the difference between a wrong timestamp and the correct one. Using a sampling rate of 1 kHz increases both the computational work and accuracy. Since the computational work is not a limiting factor but ENF estimation accuracy is, the higher sampling rate is to be preferred.

The rectangular window indeed seems to be the right choice. As stated before, the rectangular window has the narrowest main lobe, making it pointier. This is a good property for peak estimation. The downside of the rectangular window is the height of the side lobes, but the ENF criterion is not concerned with side lobes of the voltage signal. The bandwidth filter is there to attenuate all side lobes from non-voltage sources. It is hard to optimally choose the window length. There exists a trade-off between accuracy and detail. A non-overlapping window of one second length would be preferred because only then the ENF of exactly that one second is estimated. But when this short window is applied, the DFT artifacts caused by the window boundaries become large and the signal length is too small. Even if the 1 second signal contains no noise, the short duration will result in wrong frequency estimations. Increasing the window length too much, to say 30 seconds, means that two consecutive windows have 29 seconds of overlap. As a result the two will have almost identical ENF values. The entire ENF signal will then be too smooth losing all detailed information like short term ENF changes. A 10 second window seems to be well balanced.

The greatest adjustment to Coopers procedure is the use of DFT in stead of FFT with zero-padding. Using the DFT saves a lot of computation time because the amount of coefficients to be calculated is reduced with a factor 1000. As a result the computation speed is increased with at least a factor 10. Also, the frequency resolution can be chosen freely. 10 mHz resolution in combination with the quadratic interpolation is sufficient. This is equivalent to an FFT with zero-padding factor 9, in combination with a 10 second window function. Coopers zero-padding factor of 4 again results in a 0.1 mHz estimation error. This will be illustrated in section 3.4.2.

3.4.2. Frequency resolution and QI

In the procedure suggested by Cooper, the STFT is used with a zero-padding factor of 4, in combination with the quadratic interpolation. Again, the question is asked whether this zero-padding factor with QI yields a frequency resolution which is high enough and whether the top estimation is accurate. Because in this report not the STFT but the DFT is used, it is important to note again that the DFT does not require zero-padding because the frequency resolution can be chosen freely. Coopers zero-padding factor of 4 (with 9 seconds overlap) results in a frequency resolution of 20 mHz. Which frequency resolution results in sufficient accuracy for the ENF estimation?

Using the same signal as in the previous section 3.4.1, different zero-padding factors (frequency resolutions) are used, with and without QI, to show the convergence of the ENF estimation to the correct value which is 49.9972 Hz. Figure 3.9 shows the process without the use of QI and Figure 3.10 and 3.11 do use QI.

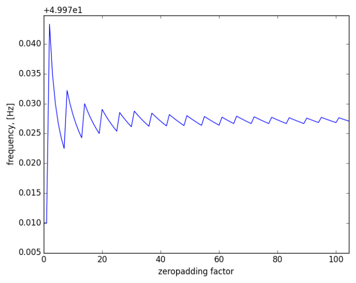


Figure 3.9: Without QI

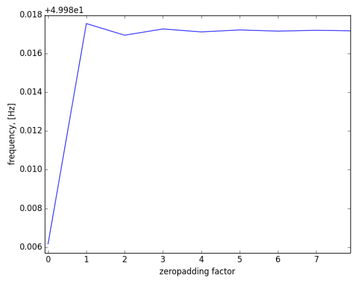


Figure 3.10: With QI

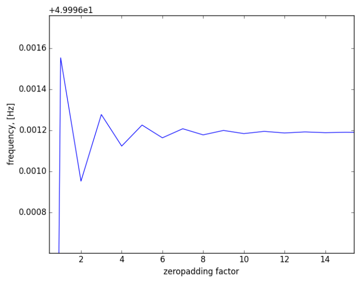


Figure 3.11: With QI zoomed in

The saw-pattern in Figure 3.9 clearly illustrates the lack of interpolation. The FFT resolution is given by $\frac{1}{1+zp}$ where zp is the zero-padding factor. As a result the maximum deviation of the estimation is $\frac{1}{2(1+zp)}$. Indeed the shape of the graph is like $\frac{1}{x}$ and $\frac{-1}{x}$. For each zero-padding factor there is a discrete set of points from which the maximum is determined. It may happen that a frequency bin is very close to the 49.9972 Hz for certain zero-padding factors, and in other cases the optimal value is located right in between two frequency bins.

Figure 3.10 shows a less jagged pattern because due to the QI a continuous set of ENF values can be

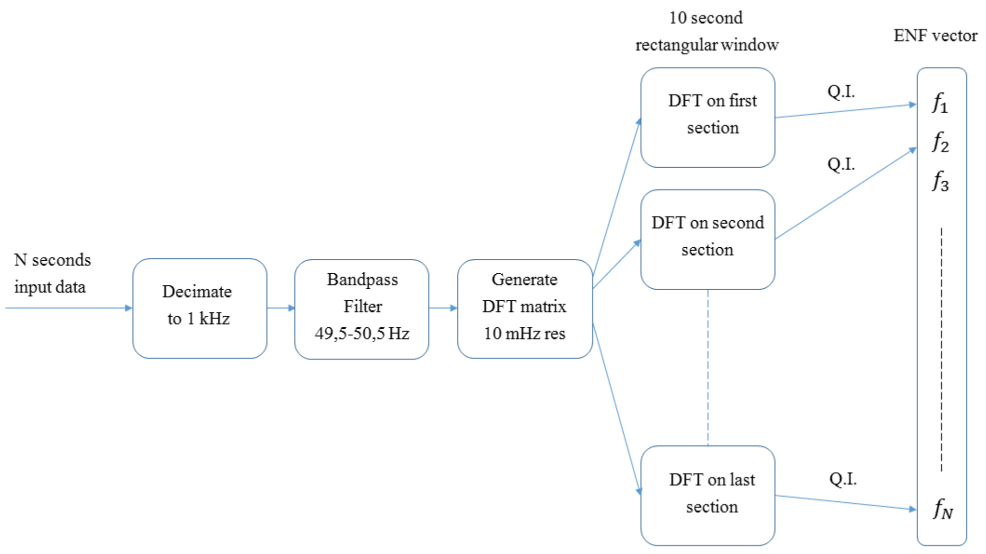


Figure 3.12: Improved ENF estimation procedure

determined. The convergence is fast and increasing the factor further than 4 does not appear to show much improvement. Figure 3.11 however shows that $zp = 9$ (10 mHz resolution) improves accuracy with 0.1 mHz compared to $zp = 4$. For this reason 10 mHz is the DFT frequency resolution used in the new procedure.

The preferred procedure for this report is illustrated in figure 3.12:

4
ENF database matching

For the purpose of timestamping it is essential to set up a database of ENF values with their corresponding times. This is normally done by using a step down voltage directly connected to the power grid and either feeding the signal to a computer sound card which runs the Fourier Transform or by counting zero crossings of the voltage signal.

The zero crossings method treats the signal as sinusoidal, and calculates the time τ between two consecutive zero crossings. The instantaneous frequency is then $f = \frac{1}{2\tau}$. The times of the zero crossings are determined by linear interpolation between consecutive samples that differ in sign. The frequency of one second of signal is the average of all frequency estimations obtained in that second. Such databases were already existing before the application of the ENF criterion because the power grid frequency is continuously being monitored. Note that the location where the database is created has never been considered important for timestamping as long as it is on the same grid as the recording that is to be timestamped.

A few minutes of extracted ENF is often sufficient to accurately timestamp the audio recording. Under two minutes the recording may be too short, resulting in the loss of uniqueness of the ENF signal. A wrong database match could be found [10]. Hence, the length of the recording is another important parameter for timestamping.

The database matching technique all authors use is straightforward: The estimated ENF of a recording of n seconds is compared to the first n seconds of database values, then to the segment shifted one second forward etc. The database segment which is the closest match to the extracted ENF signal may indicate the time of recording.

4.0.3. Cross Correlation versus Mean Squared Error

Now the question rises how the best match should be defined. Most literature uses one of two norms: the Cross Correlation (CC) or the Mean Squared Error (MSE). The CC of the two length N ENF vectors x and y is defined as $CC = \frac{\sum x \cdot y}{N \cdot \sigma_x \sigma_y}$, where σ is the variance of the ENF vector. To find a match the greatest correlation coefficient in the database has to be found. This is more or less equivalent to finding a minimum MSE where $MSE = \frac{\sum (x-y)^2}{N}$. Here x can be treated as the extracted ENF signal and y the selected database segment.

Many articles have compared the performance of Cross Correlation (CC) with Mean Squared Error (MSE). There have been claims that maximizing the CC matching should be preferred over minimizing the MSE since it would increase the chance of accurately timetamping recordings [10]. Others claim the MSE provides better results, when both vectors are zero mean [4].

The MSE distance norm without subtracting the vector mean may give bad results since some recording devices have a frequency offset due to its hardware [12] [11]. When the vectors are both zero mean, the frequency offset does not play any role in the matching process.

An article about weather forecasts by Barnston [1] shows a relation between the CC and the MSE. After some adjustments to fit the analysis to the ENF database matching, the equivalence of both matching scores can be shown. First define the ZMMSE to be the zero mean MSE, then the following relation holds:

$$
\begin{aligned}
ZMMSE &= \frac{\sum (x-y)^2}{N} \\
&= \frac{\sum x^2 + \sum y^2 - 2\sum x \cdot y}{N} \\
&= \sigma_x^2 + \sigma_y^2 - 2CC \cdot \sigma_x \sigma_y \\
&= 2\sigma^2 (1 - CC)
\end{aligned} \tag{4.1}
$$

Here the assumption is made that the standard deviation of the extracted ENF is equal to its match in the database, which is fair because the vectors are very much alike.

The relation between the correlation coefficient and the zero-mean mean squared error is one-to-one and linear. therefore it is unlikely that one would give better results than the other in terms of precision. Hence it is advised to use the ZMMSE approach. Not because it makes any difference in accuracy, but to minimize the computational work.

Minimizing $\sum (x-y)^2$ over all database segments seems to be an efficient matching algorithm. (Note that dividing by the constant vector length is superfluous). Now the question rises what score for the sum of squares should be considered a convincing match. Especially for noisy records, the lowest MSE score not always gives the right timestamp. Two matching score criteria will be discussed in this chapter.

4.1. Interdatabase error

The first technique is developed by Huijbregtse and Geradts [10]. They had an ENF database of 1.5 years of frequency values. From this data set two randomly picked non-overlapping segments of 600 ENF values (10 minutes) are compared. This is repeated one million times and the matching scores (CC or MSE) are stored and plotted.

The smallest root mean squared error between two random segments was found to be 4 mHz. Based on this observation they state that for the 10 minutes ENF vector of interest, the root mean squared error with respect to a database segment should be "well below" 4 mHz. Whenever this is not the case, the strength of the match is not good since a better match has been found between 2 random disjoint samples taken from that database. A false match is not unlikely for error scores around 4 mHz.

For this thesis, a similar experiment is conducted with a 15 hour ENF measurement during which a short audio file is recorded. A time of recording is estimated and the quality of the match is expressed in terms of the interdatabase error. As stated before, the error score is the sum of the squared error: $\sum (x-y)^2$ where x and y are the two random segments, both transformed to zero-mean. They should however start more than 10 seconds after each other, else the segments are still too similar and not really two different segments. One million pairs of database segments are matched and the scores are plotted in a histogram:

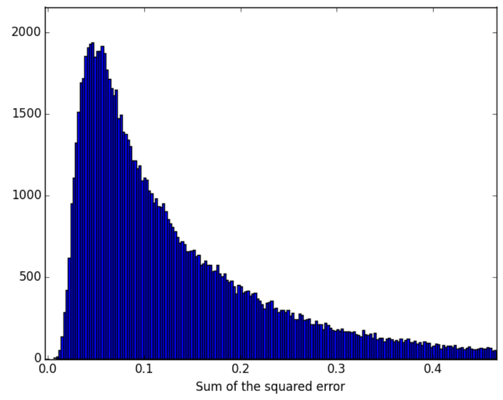


Figure 4.1: 5 minutes segments error

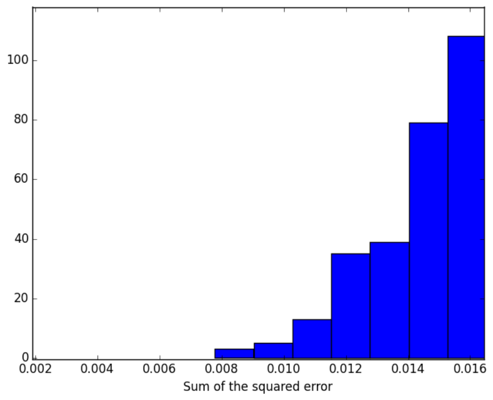


Figure 4.2: 5 minutes segments error, zoomed in

From these interdatabase errors it is observed that there are several random segment pairs which have an error as small as .008. Therefore the audio file ENF should have an error "well below" .008 for a strong match. This technique illustrates which error scores can be obtained by random matching and it gives an upper bound to the error of a true match. In this case the upper bound could be set to .006. When increasing the number of simulation it is possible to fit an exact probability distribution to the data. This probability distribution can then be used to set different match strength intervals. The shape of this distribution is always more or less equal, but what the lowest observed error is, depends on the number of simulations and the size of the database. A larger database in which over a million segments are compared is more likely to find low error scores than a small database like this one with only 15 hours of ENF values. A second criterion can be used in conjunction to this one:

4.2. Error distribution

The technique of the error distribution is described by Cooper [4]. The errors of the ENF extract with respect to all the database segments form a distribution. In certain experiments the authors found that the error is normally distributed and for some other the distribution was skewed. In case of a skewed distribution it can be transformed to a normal distribution. Now the point of smallest error is located a certain number of standard errors under the mean error. This is called the z-score and can be used to quantify the strength of the match. Furthermore, plotting all errors in one figure also graphically shows how small the error is compared to the other errors. When a few of the smallest errors are relatively close to each other, the match is questionable. So this is basically the actual matching process in which all errors are stored and the smallest error is compared to all other errors.

In Figure 4.3 and 4.4 the error distribution is plotted. Again, a skew to the left is observed. The idea of the scoring is based on a normal distribution, therefore the logarithm of the errors is used to transform the lognormal distribution to a normal one. Notice that the logarithmic transformation is order preserving meaning that the smallest squared error is also the smallest after transformation.

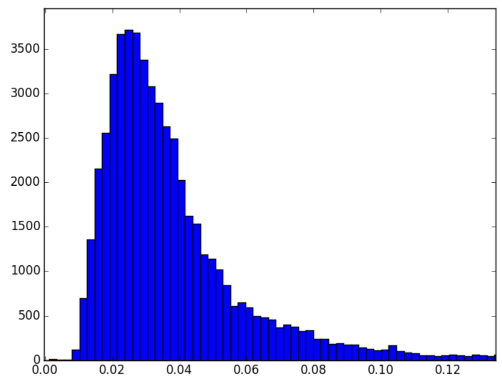


Figure 4.3: Squared error distribution

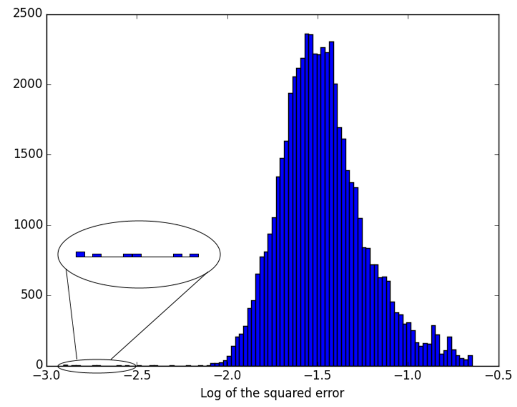


Figure 4.4: Logarithmic transformation to normal distribution

There appear to be multiple segments in the database with a log matching score of under -2.2. There are even multiple segments in the lowest error bin. That would mean that the match is far from certain and that especially the left tail does not fit a normal distribution well. The corresponding segments however are segments slightly shifted from the true match. If segment k is the real match, with the lowest error, then segments $k-10$ to $k+10$ all have low error scores. This can be seen in Figure 4.5 and 4.6.

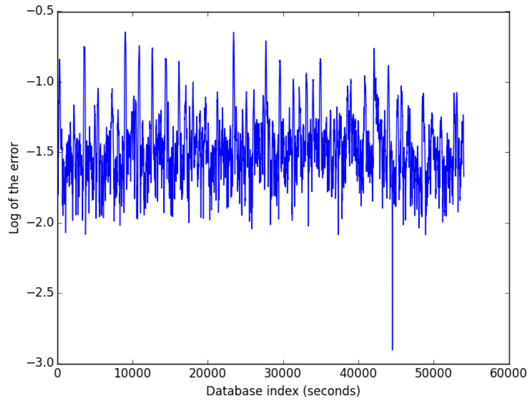


Figure 4.5: The log of all errors

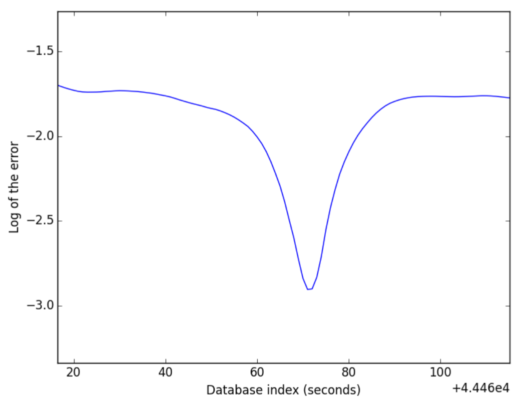


Figure 4.6: Errors around the match index

A way to go about this is to delete all 20 scores around the minimum score and then calculate the z-score of the lowest error. Figure 4.5 shows a single convincing minimum at 44531 seconds. The scores of the 20 surrounding segments are deleted from the normalized log error distribution and the left tail is shown in Figure 4.7. Now the distribution looks more like a normal distribution with one outlier value. The z-score of this lowest log error is -6.09 and it is very likely that the recording will be timestamped correctly. Recall that the lowest score of the interdatabase match which was 0.006. The squared error of the candidate match is 0.0012, so combining the fact that the error is significantly lower than the interdatabase lower bound with the z-score and the outlier-property, the found match is very likely the correct one.

The visual comparison is also an important aspect of the matching. From this it is observed that the first twenty seconds of the recording are very noisy: large ENF spikes are observable in the recording's ENF signal, where this is not the case in the reference database. Also, during the 5 minutes the audio's ENF signal graph is more jagged than the database signal. Different noise reduction techniques can be used to anticipate these undesired spikes to increase match certainty even further.

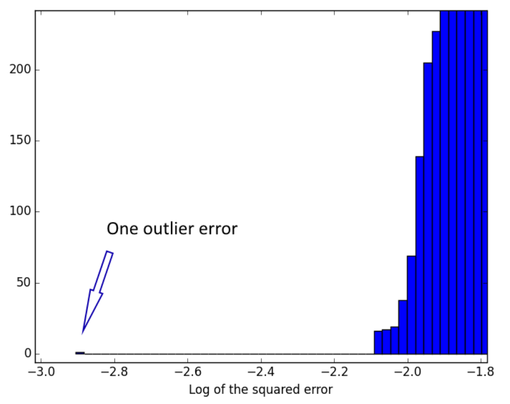


Figure 4.7: Squared error distribution

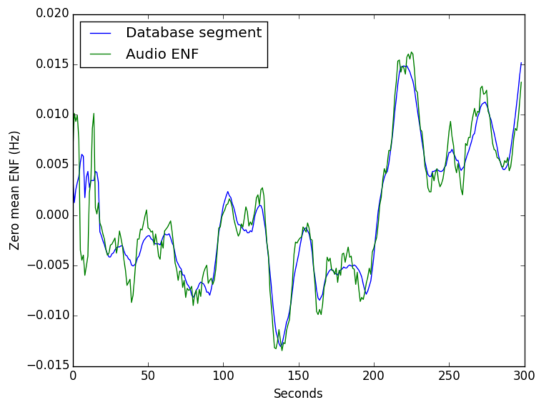


Figure 4.8: The match is found

5 ENF signal from video

This section discusses the estimation of the time of recording applied to video in stead of audio. The applicability of the timestamping theory to audio is limited to a small group of recordings. The audio recording has to be either mains powered or the microphone must be in a strong electromagnetic field. Secondly, 50 Hz sound must be recorded by the equipment. There are many recorders that have a built in high pass filter set at 200 Hz for example because almost all sounds of interest are above 200 Hz. With such a filter, there is no ENF recorded, even though the equipment may be mains powered or in a strong electromagnetic field. Finally, there is the obstacle of noise. If there are non-ENF sources of sound containing 50 Hz waves, the ENF extraction is seriously obstructed.

In some cases there is only video available without sound, like when security tapes are brought up as evidence in legal cases. The authors of "Seeing ENF" [6] examined the possibility of ENF estimation from video frames. The ENF is not only traceable via audio but also via light. Fluorescent light, as found in many offices, blinks at a rate of twice the ENF. Applying power at the nominal ENF of 50 Hz, its polarity changes at twice that rate and the current path is switched on and off at about 100 Hz. For example, as a stationary camera records a white wall with fluorescent light in the room the frames will all have different light intensity depending on the phase of the current. Our eyes process a white wall with constant brightness and can not register 100 Hz flickering, but the camera does. This property will be exploited by the ENF criterion for video recordings. Each frame's average pixel intensity is stored. This is the signal from which the ENF is extracted.

There is however one problem with the frames intensity registration and that is the low sampling rate of most video equipment. A frame rate of about 30 fps is a common setting and as stated before it is generally only possible to detect frequency components of up to half the frame rate, which would be 15 Hz in this case. Fortunately, the aliasing effect allows a trace back of the 100 Hz signal due to symmetry of the frequency spectrum around $f_s/2$ and duplications of the spectrum after each f_s Hz. This duplication of the complete Fourier spectrum is due to the fact that $\sin(\omega t)$ is indistinguishable from $\sin((\omega + f_s)t$, when sampled with f_s Hz.

Tests have been done using a camera with a frame rate of 29.97. The flickering signals that are picked up are not only the 100 Hz but also the some harmonics at multiples of 100 Hz. To illustrate the aliasing effect an ENF of 50 Hz is assumed with the second and third harmonic present. The Fourier Transform can only detect frequencies between 0 and $\frac{f_s}{2}$, so the fundamental 100 Hz and its harmonics' aliases within that range need to be determined. This is shown in the table below.

Frequency (Hz)	100	200	300
Frequency ($\cdot f_s$)	3.33667	6.67334	10.01001
Duplicate (mod 1)	.33667	.67334	.01001
Symmetry around $\frac{f_s}{2}$	.33667	.32666	.01001
Alias (Hz)	10.09	9.79	.30

In this set-up the fundamental and the second harmonic are projected very close to each other, at 10.09 and 9.79 Hz, causing the spectra to interfere with each other. A perfect 50 Hz ENF should result in a spectrum peak at 10.09 Hz, which is its alias peak frequency, but due to the presence of harmonics in combination with this frame rate of 29.97, the peak frequency will be lower. Hence the resulting ENF estimation will be incorrect. The following figure shows the spectrum of a signal consisting of the fundamental frequency of 100 Hz, in the presence of a weaker second and third harmonic, when the ENF is 50 Hz.

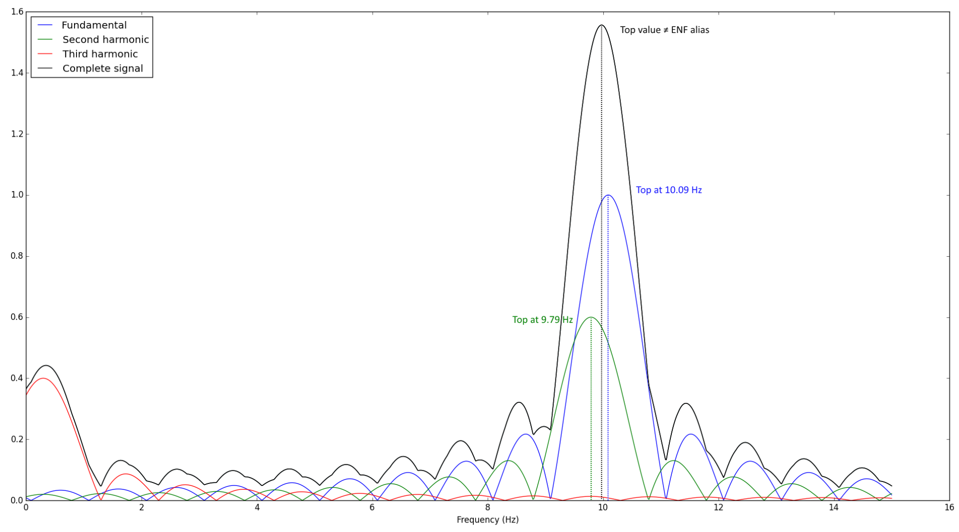


Figure 5.1: Interfering harmonic aliases of a signal in the FFT spectrum

Without harmonics there would be no problem because the top value of the spectrum would be located at 10.09 which is exactly the alias of a 100 Hz signal sampled at 29.97 Hz. In that case there is a one-to-one relation between the alias peak frequency and the ENF. However, especially the second harmonic causes the top of the spectrum to be located at some other frequency. There is no way to reconstruct the actual ENF value and the estimation would be too low.

It must also be noted that the ENF signal consisting of these second harmonic aliases has a doubled bandwidth, meaning that each ENF deviation from the mean 50 Hz will result in a doubled deviation from the 9.79 Hz value. Also, these values are mirrored since the positive frequencies around 9.79 are actually symmetries of the negative frequencies. For example, when the ENF is 49.98 Hz, the fundamental frequency of 99.96 Hz will be projected onto 10.07 Hz, and the the second harmonic is projected onto 9.83 Hz. In the same way, if the third harmonic is also present in the fluorescent light, the 49.98 Hz ENF will have its third harmonic alias at 0.24 Hz.

ENF extraction from video frames poses a real challenge but an image capturing artifact called the rolling shutter effect may offer a good alternative method for video ENF estimations. Most non-professional video recording equipment are equipped with CMOS sensors which do not capture one entire frame instantly but each row of a frame is saved subsequently. This means that the first row of pixels in a frame is recorded $\frac{1}{framerate}$ seconds before the last row of that same frame. This property causes the rolling shutter effect, which is noticeable in videos and images of very fast moving objects like the rotors of a helicopter, the spinning of car wheels or even lightning strikes. These objects are saved in a distorted way. The fast moving 'object' for the ENF application is fluorescent light. A two minutes 800x480 video is recorded in an office with fluorescent lighting using a regular smartphone. Figure 5.2 shows the rolling shutter effect in an individual frame.

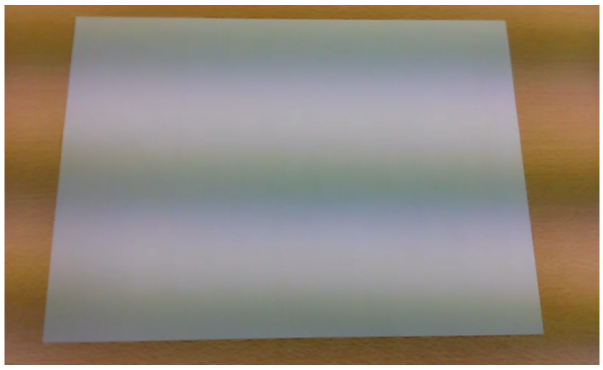


Figure 5.2: Rolling shutter effect in the office

Three dark rows are clearly noticeable in this image. The frame is saved in a time of almost three and a half light flickering periods. Combining this with the knowledge of the frame rate of this CMOS camera, the frequency of the light flickering can be determined as $f \approx \frac{\frac{1}{30}}{3.5}$. Hence $f \approx 100$ Hz which indicates that indeed each row is saved individually and the dark stripes are due to the flickering of the fluorescent light. The question becomes whether or not a decent ENF estimation can be determined using this artifact.

Again the pixel intensity signal will be studied, but this time it will not be the average pixel intensity per frame but the signal is the average pixel intensity *per row*. Each frame of the video is placed under the previous one and the rows are counted all the way down from row one in frame 1 to row h of frame $30 \cdot s$, where h is the frame height and s the video length in seconds. Treating the video as if each pixel row is one frame, the frame rate becomes $30 \cdot h$. The great advantage is that there is no more alias effect interfering with the 100 Hz signal. The intensity signal and spectrogram are given in figure 5.3 and 5.4.

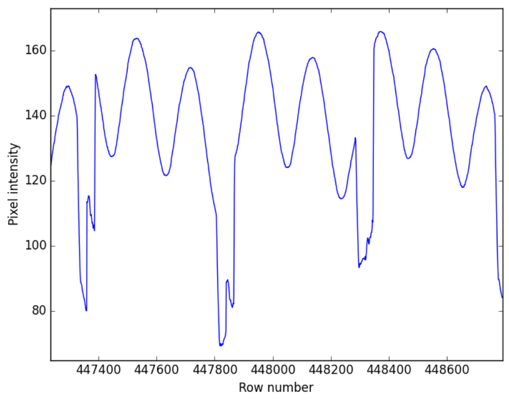


Figure 5.3: Pixel intensity per row

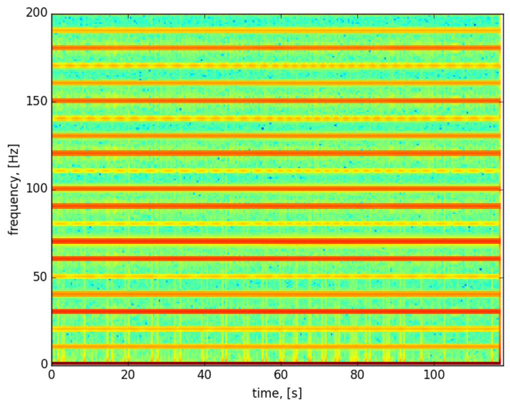


Figure 5.4: Pixel intensity spectrogram

There is a clear sinusoidal component in the 8-bit gray scale pixel intensity which corresponds to the 100 Hz light flickering, and each 480 rows there is an interruption caused by the top and bottom of each frame, where the white paper ends. The spectrogram reveals a strong presence of frequencies at almost each multiple of 10 Hz. These frequencies are all harmonics of 100 Hz and 30 Hz. The 30 Hz has a strong presence in the signal due to the similarity of each frame, which recurs thirty times a second. In the previous situation, the fundamental alias at 10.09 was very close to the alias of its second harmonic, at 9.79. Because there is no negative effect from aliasing at this high sampling rate, the 100 Hz can be accurately determined at its own frequency position. Simultaneous to the video, an audio recording was made with a high SNR. Figure 5.5 shows the great similarity between the two ENF signals. The video ENF shows more continuity at the same window size, which would indicate a more accurate estimation. Unfortunately there is no equipment available to serve as a ground truth ENF to verify this hypothesis. The largest difference between the two signals is measured to be 0.6 mHz indicating that using the rolling shutter artifact is a viable way to estimate the ENF. This first video was made under near ideal circumstances however, using a stationary camera recording a white paper. The experiment is repeated using a moving camera filming different objects in the office, more like a real-life video which might be encountered in forensic research. The result is a jagged ENF array which more or less follows the audio ENF. In some cases the dynamic video might offer accurate enough ENF estimations to give a time of recording, or to detect video editing, which was the goal of the ENF extraction process.

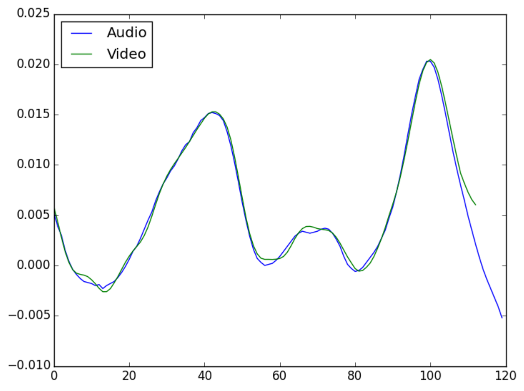


Figure 5.5: Frequency estimation from audio compared with video rows

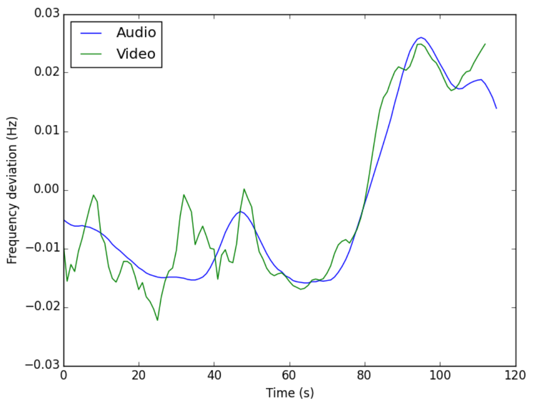


Figure 5.6: A second test has been done

Because video registers twice the ENF value at around 100 Hz, the frequency deviations are also doubled compared to the frequency extraction from audio. This should be considered when matching video ENF signals. Also, significant frequency offsets can be expected. The camera used here has a frequency offset of 96 mHz. It must however be noted that most camera's are equipped with an automatic cancellation of indoor lighting artifacts. This significantly reduces the applicability of video ENF estimation. It has not been checked yet whether it is possible to, after some editing, extract ENF from video images which are filtered for light flickering.

6 ENF signal propagation models

One most interesting topic in the study of the ENF criterion is whether it is possible to also accurately estimate the location of a recording.

By most researchers this question is answered with no. The ENF is assumed to be the same at each location of an interconnected power grid and therefore there should not be any location dependence of the ENF. Still one article has been written containing an attempt to locate an audio recording within an interconnected grid [7]. The authors claim that the ENF is non-uniformly distributed over an interconnected grid at a given time. The argument is that a local load change has an effect on the nearby generators and that its frequency deviation with respect to the other generators propagates through the grid at finite speed. Accurate simultaneous measurements back this theory up. However, no theoretical analysis is provided. The measurements done in that research were in the United States. Five ENF databases at 200 to 1200 kilometers apart were created. The ENF values of the different databases can differ 1 to 2 mHz at a certain time. Assuming the ENF vectors are more similar for locations nearby than for locations at greater distance, the authors established a location estimation procedure.

In this chapter a theoretical framework for the non-uniformity of the ENF signal will be given. In section 6.1 the conclusion will be drawn that the correlation between voltage signals decays exponentially over distance. Based on this result section 6.2 creates a model for frequency distribution over the grid. It seems that there should not be any significant difference in ENF values at different location if the grid is interconnected well. This property explaines why timestamping is actually possible without considering different locations on a grid. It is however not the case that the ENF is equal at each location and therefore the model may serve as a means to measure the quality of the network connectivity. Section 6.4 shows that a local load change results in a local power discrepancy. The nearby directly connected generators change in frequency and the generators which are not directly connected to the load are less sensitive to that power discrepancy. This way, the generators can run slightly out of sync. Knowing that the generators can run on slightly different frequencies, section 6.4 shows what frequency signal is to be expected at a certain location when its distance to different generators is known. Based on those findings, it is possible to estimate the location of a recording, using the ENF values at different locations. Chapter 7 concludes with three different models to estimate the location of a recording based on ENF databases at known locations.

6.1. Signal propagation

The Telegrapher equations describe the voltage and current on an electrical transmission line over time and distance. The grids resistance (R), inductance (I), capacitance (C) and conductance (G) are important factors. The coupled first order PDE's for a lossy transmission line in the time domain are given by:

$$\begin{aligned}\frac{\partial}{\partial x}V(x,t) &= -L\frac{\partial}{\partial t}I(x,t) - RI(x,t)\\ \frac{\partial}{\partial x}I(x,t) &= -C\frac{\partial}{\partial t}V(x,t) - GV(x,t)\end{aligned} \tag{6.1}$$

Alternatively, equations (6.1) can be formulated as two decoupled second order equations:

$$\begin{aligned}\frac{\partial^2}{\partial x^2}V(x,t) &= LC\frac{\partial^2}{\partial t^2}V(x,t) + (RC+GL)\frac{\partial}{\partial t}V(x,t) + GRV(x,t)\\ \frac{\partial^2}{\partial x^2}I(x,t) &= LC\frac{\partial^2}{\partial t^2}I(x,t) + (RC+GL)\frac{\partial}{\partial t}I(x,t) + GRI(x,t)\end{aligned} \tag{6.2}$$

For negligible isolation material conductance G, this reduces to:

$$\begin{aligned}\frac{\partial^2}{\partial x^2}V(x,t) &= LC\frac{\partial^2}{\partial t^2}V(x,t) + RC\frac{\partial}{\partial t}V(x,t)\\ \frac{\partial^2}{\partial x^2}I(x,t) &= LC\frac{\partial^2}{\partial t^2}I(x,t) + RC\frac{\partial}{\partial t}I(x,t)\end{aligned} \tag{6.3}$$

Which are wave equations with an extra first order term causing the signals to decay over time and distance.

Starting with the Telegrapher equations in the frequency domain:

$$\begin{aligned}\frac{\partial}{\partial x}V &= -(R+j\omega L)I\\ \frac{\partial}{\partial x}I &= -(G+j\omega C)V\end{aligned} \tag{6.4}$$

can be written as:

$$\begin{aligned}\frac{\partial^2}{\partial x^2}V &= \gamma^2 V\\ \frac{\partial^2}{\partial x^2}I &= \gamma^2 I\end{aligned} \tag{6.5}$$

where $\gamma = \sqrt{(R+j\omega L)(G+j\omega C)}$ is the propagation constant and ω the angular frequency.

Assuming relatively small losses R and G compared to the frequency we obtain an approximation for $V(x_2, t+\tau)$, given an input pulse $V(x_1, t)$:

$$V(x_2, t+\tau) \approx V(x_1, t)e^{-\frac{\sqrt{LC}}{2}\left(\frac{R}{L}+\frac{G}{C}\right)(x_2-x_1)} \tag{6.6}$$

τ is the travel time of the electric signal from x_1 to x_2. It holds that $\tau = \sqrt{LC}x$.

Equation (6.6) can be manipulated to investigate the behavior of the cross correlation of a signal with itself at a distance.

Multiplying both sides with $V(x_1, t)$ and time averaging results in:

$$\langle V(x_2, t+\tau)V(x_1, t)\rangle \approx \langle (V(x_1, t))^2\rangle\, e^{-\frac{\sqrt{LC}}{2}\left(\frac{R}{L}+\frac{G}{C}\right)(x_2-x_1)} \tag{6.7}$$

This shows that the correlation coefficient of the power grid signals measured at two locations is maximal when $x_1 = x_2$ and wears off exponentially with the increase of the distance.

6.2. A model for instantaneous spatial frequency distribution

Consider a general signal $s_i(t)$ at location i. In an interconnected grid the signal will be a combination of signals coming from different sources j. The weights are coefficients $a_{i,j}$, which are exponential in electric distance as concluded by the previous section. Alternatively, these coefficients can be approximated linearly. This results in the following model:

$$a_{i,i}s_i(t) = \sum_{j\neq i}^{k} a_{i,j}s_j(t) \tag{6.8}$$

The coefficient $a_{i,i}$ is set to 1. Then the sum of coefficients $\sum_{j\neq i}^{k} a_{i,j}$ is equal to 1 for each i. Next, the equation is Fourier transformed, and the spectrum's magnitude $v_i(\omega)$ is considered.

$$a_{i,i}\hat{f}_i(\omega) = \sum_{j\neq i}^{k} a_{i,j}\hat{f}_j(\omega) \tag{6.9}$$

$$a_{i,i}v_i(\omega) = \sum_{j\neq i}^{k} a_{i,j}v_j(\omega) \tag{6.10}$$

The second order Taylor approximation is taken around ω_j which the top of each of the frequency spectrum j. Furthermore define $v_i \equiv v_i(\omega_i)$ and $v_i' \equiv v_i'(\omega_i)$.

$$v_i(\omega) = v_i(\omega_i) + (\omega-\omega_i)v_i'(\omega_i) + \frac{1}{2}(\omega-\omega_i)^2 v_i'' + \mathcal{O}((\omega-\omega_i)^3) \quad i = 1,\dots k \tag{6.11}$$

Substituting equation (6.11) into (6.10) yields:

$$a_{i,i}\left(v_i + (\omega-\omega_i)v_i' + \frac{1}{2}(\omega-\omega_i)^2 v_i'' + \cdots\right) = \sum_{j\neq i}^{k} a_{i,j}\left(v_j + (\omega-\omega_j)v_j' + \frac{1}{2}(\omega-\omega_j)^2 v_j'' + \cdots\right)$$

Using the property of the maximum $v_i' = 0$ in ω_i we get:

$$a_{i,i}\left(v_i + \frac{1}{2}(\omega-\omega_i)^2 v_i'' + \cdots\right) = \sum_{j\neq i}^{k} a_{i,j}\left(v_j + \frac{1}{2}(\omega-\omega_j)^2 v_j'' + \cdots\right) \tag{6.12}$$

Differentiating both sides with respect to ω:

$$a_{i,i}(\omega-\omega_i)v_i'' + \cdots = \sum_{j\neq i}^{k} a_{i,j}(\omega-\omega_j)v_j'' + \cdots \tag{6.13}$$

Discretization of the problem using $\omega = \omega_i$ for $i = 1,\dots k$ yields the following equations:

$$0 = \sum_{j\neq i}^{k} a_{i,j}(\omega_i-\omega_j)v_j'' + \cdots \tag{6.14}$$

$$\sum_{j\neq i}^{k} a_{i,j}\omega_j v_j'' \approx \omega_i \sum_{j\neq i}^{k} a_{i,j}v_j'' \tag{6.15}$$

Hence for each i the following relation holds, with an error of $\mathcal{O}(\omega-\omega_i)^2$.

$$\frac{1}{\sum_{j\neq i}^{k} a_{i,j}v_j''}\sum_{j\neq i}^{k} a_{i,j}\omega_j v_j'' = \omega_i \tag{6.16}$$

This is written as a linear system

$$\frac{1}{\sum_{j\neq i}^{k} a_{i,j}v_j''}\begin{bmatrix} 0 & a_{1,2}v_2'' & a_{1,3}v_3'' & \cdots & a_{1,k}v_k'' \\ a_{2,1}v_1'' & 0 & a_{2,3}v_3'' & \cdots & a_{2,k}v_k'' \\ a_{3,1}v_1'' & a_{3,2}v_2'' & \ddots & & \\ \vdots & & & & \\ & & & & 0 \end{bmatrix}\begin{bmatrix}\omega_1\\ \omega_2\\ \vdots\\ \omega_k\end{bmatrix} = \begin{bmatrix}\omega_1\\ \omega_2\\ \vdots\\ \omega_k\end{bmatrix} \tag{6.17}$$

Note that this is a row-stochastic matrix with all non-negative entries. The ω-vector is the eigenvector corresponding to the eigenvalue 1. Whenever enough $a_{i,j}$ are non-zero, then the graph representation of this matrix is a strongly connected Markov chain, meaning that each vertex is reachable from every other vertex. This irreducibility property allows the use of the Perron-Frobenius theorem for irreducible matrices. The spectral radius of the above matrix is equal to 1 and the theorem states that this is also the largest eigenvalue.

This eigenvalue is also simple, meaning that its associated eigenspace is one-dimensional. The vector solving this system is a constant vector. It is concluded that according to this model, there cannot be any difference in frequency between different locations. This explains the fact that the ENF timestamping works, regardless of the location on the grid where the recording was made.

However, the ENF is slightly different at other locations in the same grid, as is concluded from accurate measuring. The university of Tennessee Knoxville has been monitoring the ENF in the United States at many different location. From this they observed that the ENF is certainly not constant over distance. Under normal circumstances the difference may be as large as 1 or 2 mHz. For power quality monitoring this is negligible, for timestamping it may cause an accuracy loss but for location estimation it is substantial enough to separate one city from another. What then causes the different ENFs simultaneously over the same grid? From a mathematical point of view it might be explained if the connectivity matrix is not irreducible, meaning that groups of $a_{i,j}$ need to be zero. If certain parts of the grid are not or badly connected to other parts the system may have more solutions than just the constant vector, which could be more like the small deviations which are measured. When the frequency of the signals produced by different generators are not identical this has to be due to a badly interconnected network with non-linear loads. This makes measuring frequencies at different locations a way to determine the quality of an electrical network's connectivity.

6.3. Generator frequency dynamics

In this section an attempt is made to model the frequency disturbance propagation of a power network. The question for this section is: What causes the ENF to be geographically non-uniform? It suggested that a local power discrepancy is locally compensated for. The frequency disturbance gradually spreads throughout the grid as a result of so called radiated power.

In 2011 an earthquake struck the East coast of the United States. The measurements of the Tennessee University showed that as a result the generators in that relatively small area suddenly dropped in rotation speed. Within seconds, the other generators in the grid drop in frequency since the generators in the earthquake area became a heavy load to the rest of the generators. The frequency drop spreads through the entire eastern interconnected grid within 3 seconds like a wave. The frequency of the generators in the area affected by the earthquake does not decrease monotonously but in an oscillating fashion. This oscillation is weaker for generators further away from the earthquake. During this event, the instantaneous frequency difference between generators in the same interconnection was up to 50 mHz.

In 2008 there was a generator trip in the south end of the area. One generator suddenly generated power at a frequency of 500 mHz higher than the rest of the generators. Again within 3 seconds the increased frequency reaches the other side of the grid. The overall space-time behavior of the frequency deviation is like a stone thrown in a pool that causes waves, spreading out, oscillating around the mean value and coming to a rest after 15 seconds. One could state that this behavior is very much like the PDE describing the voltage of a power system, equation 6.2, which is a wave equation with dissipation and a source term. However, the voltage propagates at nearly the speed of light, while the frequency disturbance travels at about only 1200 km per second. This lower speed is due to the generators' inertia and the fact that not every pair of generators is directly connected, so many connectivity coefficients $a_{i,j}$ are zero.

A start will be made to create a model for the dynamics of the frequency of generators in a network. Since this will be the first ENF literature to attempt such a model, the analysis will be a brief kick-off. Recall the two equations describing the generators' spinning frequency and the power control, equation 2.1, 2.2:

Consider multiple generators in a power network. Each hour the total mechanical power supplied to the generators is set equal to the forecast power demand:

$$\sum_i P_i^s = \sum_i P_i^{fd} = P^d \tag{6.18}$$

The notion of local power demand is also introduced. When a certain load increases, it will only directly draw power from the directly connected generators. Each generator now has a separate control equation:

$$\frac{d}{dt}\left(P_i^s - P_i^d\right) = \alpha_i (f_i - f_0) \tag{6.19}$$

The frequency f_i is the voltage frequency which is generated by generator i. The different simultaneous frequencies are now incorporated. Hence, there are multiple differential equations for the frequency as well:

$$\frac{df_i}{d_t} = \frac{f_0}{2H}\left(P_i^s - P_i^d\right) \tag{6.20}$$

Each local generator's frequency is now depending on local power supple and demand. The frequency will change when there is a discrepancy between the two. Because the generators are interconnected in a network, this local discrepancy influences the power demand of its neighboring generators [15], [9], [16]. In this report that is called the 'radiated power'. Denote by Ω_i the set of generator's index neighboring the ith generator. Then the following equation holds:

$$\frac{df_i}{dt} = \frac{f_0}{2H_i}\left(P_i^s - P_i^d + \sum_{j\in\Omega_i} a_{i,j}\left(P_j^s - P_j^d\right)\right) \tag{6.21}$$

where $a_{i,j}$ are again the signal weights corresponding to the connectivity between two location. Because all of the radiated power has to go to the adjacent nodes, the restriction

$$\sum_{i\in\Omega_j} a_{ij} = 1$$

holds. This models the conservation of power in a closed network.

Figure 6.1 is a very simplified illustration of the effect of a single load decrease on the different generators. The decreased load causes a power surplus for the directly connected generator. The frequency of that generator changes according to equation 6.21. Now the second layer of generators also get a power surplus due to the radiated power from the first generator. In the figure this is divided equally, corresponding to a uniform vector a_{ij}. The further away from the load change, the smaller the power mismatch becomes. This results in different frequencies over the grid.

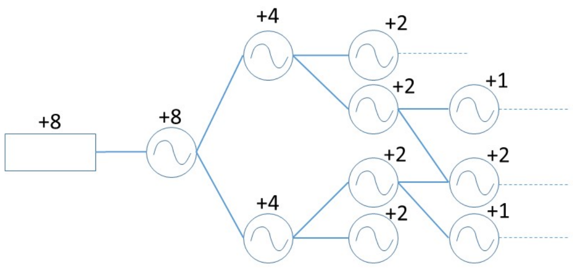


Figure 6.1: A simplified example of the spread of power discrepancy through the grid

In an article by L. Huang discusses the propagation speed of such a disturbance [9]. This propagation speed has indeed nothing to do with the speed of the voltage signal in between generators, but with the generators' inertia. Loosely speaking: It takes some time for a power disturbance to propagate through a generator. For that reason, a frequency disturbance propagates through the grid at a measurable speed, as is observed at the events described at the beginning of this section.

6.4. The ENF of a composed signal

The previous section showed that in an electrical network the ENF is not necessarily equal at different locations. The goal of this section is to gain insight in the location dependence of the ENF signal. Which ENF is measured at location X when multiple locations X_i and their ENF f_i are given? Since all generators are interconnected on the grid, and 'exchange' frequency differences by radiated power, two nearby generators run more in sync than two far away generators. Hence, the voltage signal measured at a certain location can be written as a combination of signals at other locations. The weights a_i correspond to the electrical distance $|X - X_i|$, which depends on the connectivity of the grid. The question is whether also the peak frequency (the estimated ENF) is a weighted combination of the known generators' ENF.

Let a voltage signal $s(t)$ at a fixed location X be a signal which is the composition of n signals at locations X_i, as in the previous section: $s(t) = \sum_{i=1}^{k} a_i s_i(t)$. Each $s_i(t)$ is a voltage signal produced by a generator. This signal is approximately sinusoidal and as stated before, the Fourier spectrum of such a rectangularly windowed signal is accurately estimated by a quadratic function, like it was done in equation (6.12). The error is only $\mathcal{O}(\omega - \omega_i)^4$ near the top. Therefore the spectrum of $f_i(t)$, $v_i(\omega)$, is approximated as a quadratic polynomial with its top at ω_i. All signals are again considered equal strength when leaving the generator, therefore all frequency spectra have equal maximal magnitude and the second derivative in the peak is the same for all signals. The following derivation of the peak estimation of the Fourier spectrum of a voltage signal is a QI interpretation of equations 6.8 to 6.16:

$$s(t) = \sum_{i=1}^{k} a_i f_i(t) \tag{6.22}$$

$$\hat{s}(\omega) = \sum_{i=1}^{k} a_i \hat{f}_i(\omega) \tag{6.23}$$

QI gives a general quadratic equation with its top in ω_j

$$\approx \sum_{i=1}^{k} a_i (c - b(\omega - \omega_i)^2) \tag{6.24}$$

$$= \sum a_i b \omega^2 - 2 \sum a_i b \omega_i \omega + C \qquad C \text{ constant} \tag{6.25}$$

Equation 6.25 is a quadratic equation with its top at $\omega = \frac{\sum a_i \omega_i}{\sum a_i}$ which is indeed a weighted average of the frequencies with weights according to the distance coefficients a_i. This means an ENF extract will resemble the nearby generators' frequency better than far away generators. This observation forms the basis for the following three models for localization of a recording.

7 Localization methods

7.1. Half-plane intersection

The first model to be discussed is the half-plane intersection as proposed by Garg, Hajj-Ahmad and Wu in their article about geo-location estimation from ENF [7]. This seems to be the first and to this date only published attempt to localize an audio recording using the ENF criterion. By means of experiments they have discovered that the correlation between ENF signals measured at different locations is dependent on the distance between the both locations. The measurements imply that if the correlation between the signals from X_i and X is stronger than between X_j and X, then the unknown X will be located closer to X_i than to X_j. The region $R_{i,j}$ describing this relation is a half-plane with its boundary separating X_i from X_j straight through the middle. However, it may happen that the two correlations are nearly identical in which case the uncertainty of a correct half-plane is very high. Thus, the half-plane will only be set when the correlations differ more than a certain ϵ.

Let $X_1, X_2 \cdots X_n$ be the known locations at which an ENF signal is available and let ρ_i be the correlation coefficient of the ENF signal from location X_i with respect to X. The equation for the half-plane between a pair of locations is:

$$R_{i,j} = \begin{cases} \{X : \|X - X_i\| > \|X - X_j\|\} & \text{if } \rho_j - \rho_i > \epsilon \\ \{X : \|X - X_i\| \le \|X - X_j\|\} & \text{if } \rho_j - \rho_i \le \epsilon \end{cases} \tag{7.1}$$

The correlation value of all known cities are used, resulting in $\frac{n(n-1)}{2}$ distinct pairs. The intersection of all half-planes is the region in which X is expected to be located at:

$$R = \cap_{i,j} R_{i,j}, \quad i, j \in \{1, 2, \ldots, n\}, \quad i \neq j \tag{7.2}$$

7.1.1. Example

The authors have gathered 10-hour long ENF signals at five different cities at the east coast of the US: Raleigh (R), Princeton (P), Champaign (Ch), College Park (CP) and Atlanta (A). The observation was that however the measured ENF values were very close to each other, significant differences were noticeable. All the cities are at least 200 kilometers apart.

Based on this data the half-plane method is put to the test. The ENF signal of each city is divided into 24 non-overlapping sections, and for each section all correlations were calculated. Figure 7.1 is given in the article [7] and shows the percentage of correct region estimates per section using different values of ϵ. To illustrate one such estimated region, the figure below shows the half-plane intersections corresponding to the average of all section's correlations with respect to CP.

In figure 7.2 $\epsilon = 0$ is chosen. For this value of ϵ 30% of the sections contain CP. As is to be expected with an accuracy of 30%, taking the average over all sections yields an incorrect region. The two constraints generated by city pairs Ch-A and R-P are fairly close to the actual location of CP. In each case where the calculated region does not contain CP, the error is due to a stronger correlation between CP and Ch than between CP and A. One of the main disadvantages is the large possibility of a wrong half-plane intersection as a result of cities at

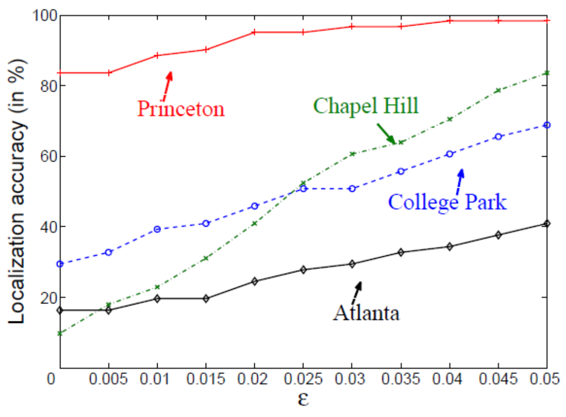


Figure 7.1: Correct region estimates

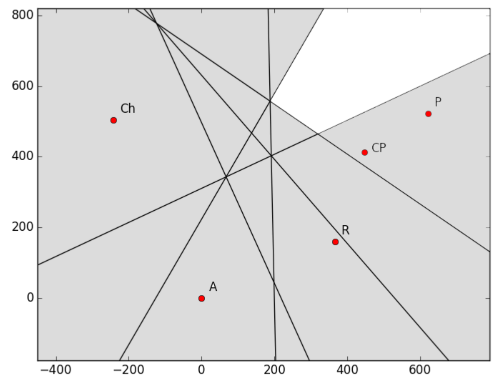


Figure 7.2: Half plane intersection region

comparable distance to the unknown location X. This is partially solved by increasing ϵ to neglect such half-planes, but this also drastically increases the size of the region. This way the model is gaining correctness but losing accuracy. Also, by discarding that half-plane entirely a lot of information is lost since according to the assumption, the city is estimated to be approximately at equal distance to the both cities. That knowledge is not used when a half-plane is discarded. The following two models do not discard that information and return a point estimation rather than an area.

7.2. Global proportionality model

All three models use the results of chapter 6. The fact that local load changes, affect nearby generators, means that ENF signals correlate less at larger distances. To be able to formulate this model as a least squares problem, the following relation is assumed: $c_i = a|X_i - X|$, where $X = (x, y)$ is the objective point. Here c denotes the *decorrelation* which is defined as one minus the correlation coefficient. So a decorrelation of zero, $c_i = 0$, indicates a perfect ENF correlation, which implies that X is located exactly at X_i. And hence, an increase in decorrelation is assumed to be the result of greater distance $|X_i - X|$ in a proportional way. For this model, a is the constant parameter which needs to be set. $a_{ij} = \frac{c_{ij}}{|X_i - X_j|}$ is calculated by using all pairs of decorrelation and distance values of the known ENF locations. a is then the average of all a_{ij}. Hence, this model is based on the assumption that decorrelation is directly proportional to the distance, with a constant factor a applying to all pairs of locations.

The derivation of the least squares problem is as follows:

$$c_i^2 = a^2|X_i - X|^2 = a^2(x_i - x)^2 + a^2(y_i - y)^2 \tag{7.3}$$

$$c_j^2 = a^2|X_j - X|^2 = a^2(x_j - x)^2 + a^2(y_j - y)^2 \tag{7.4}$$

$$c_i^2 - c_j^2 = a^2(x_i^2 - x_j^2 - 2x(x_i - x_j) + y_i^2 - y_j^2 - 2y(y_i - y_j)) \tag{7.5}$$

$$\frac{c_i^2 - c_j^2}{a^2} = x_i^2 - x_j^2 + y_i^2 - y_j^2 - 2(x_i - x_j)x - 2(y_i - y_j)y \tag{7.6}$$

This results in $\frac{n(n-1)}{2}$ equations for two unknowns.

$$2\begin{bmatrix} (x_2-x_1) & (y_2-y_1) \\ (x_3-x_1) & (y_3-y_1) \\ \vdots & \vdots \\ (x_n-x_1) & (y_n-y_1) \\ (x_3-x_2) & (y_3-y_2) \\ (x_4-x_2) & (y_4-y_2) \\ \vdots & \vdots \\ (x_n-x_{n-1}) & (y_n-y_{n-1}) \end{bmatrix}\begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} \frac{c_1^2-c_2^2}{a^2} - x_1^2 + x_2^2 - y_1^2 + y_2^2 \\ \frac{c_1^2-c_3^2}{a^2} - x_1^2 + x_3^2 - y_1^2 + y_3^2 \\ \vdots \\ \frac{c_1^2-c_n^2}{a^2} - x_1^2 + x_n^2 - y_1^2 + y_n^2 \\ \frac{c_2^2-c_3^2}{a^2} - x_2^2 + x_3^2 - y_2^2 + y_3^2 \\ \frac{c_2^2-c_4^2}{a^2} - x_2^2 + x_4^2 - y_2^2 + y_4^2 \\ \vdots \\ \frac{c_{n-1}^2-c_n^2}{a^2} - x_{n-1}^2 + x_n^2 - y_{n-1}^2 + y_n^2 \end{bmatrix} \tag{7.7}$$

Since the location X will be an approximation of the unknown location, the goal is to minimize the difference between the left-hand side vector and the right-hand side. E quation (7.7) is an overdetermined linear system $Ax = b$. To minimize $||Ax - b||$ one solves the so called normal equation $A^T Ax = A^T b$. The columns of A are linearly dependent if and only if all anchor cities are exactly on a straight line. This will not be the case in practice, hence the columns are linearly independent meaning that $(A^T A)$ is invertible and the best estimate location vector x is given by $(A^T A)^{-1} A^T b$.

7.2.1. Example

Using the data of Garg, Hajj-Ahmad and Wu from their article about geo-location estimation again [7], this model can be put to the test. The theory of section 7.2 describes how to determine the location X of one recording, given other locations X_i and all decorrelation coefficients c_i, assuming the relation $c_i = a|X - X_i|$. All correlation coefficients are given.

The following table shows the decorrelation coefficients between all signals from the different cities. The coefficients are averaged over all time segments. Furthermore, since the model assumes a linear relation between city distance and decorrelation, the figure below illustrates a linear and an exponential regression. It can be seen that the regression lines do not fit very precise, so significant estimation errors are to be expected. It may be interesting to test a model using the exponential fit, which would result in a non-linear model. However, the linear fit is a better approximation to this specific data set. Therefor the linear relation is used in this model. The exponential regression may give better results for large distance systems because it is unlikely that, as the linear regression suggests, the signals at 3300 km apart are completely uncorrelated.

	R	P	A	Ch	CP
R	0	.24	.12	.26	.13
P	.24	0	.36	.29	.07
A	.12	.36	0	.32	.27
Ch	.26	.29	.32	0	.24
CP	.13	.07	.27	.24	0

Table 7.1: 1 - Correlation coefficients of US east-coast ENF signals

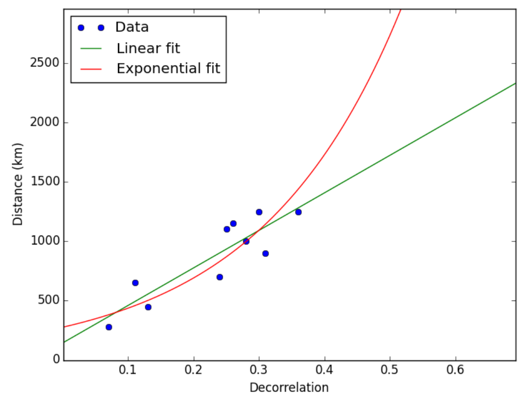


Figure 7.3: A linear fit to the decorrelation and distance data

As for the known locations of the anchor cities, let A be located at coordinates $(x_A, y_A) = (0,0)$ where x and y are in kilometers. Then the coordinates of the cities are: A = (0,0),
Ch = (-243,504), R = (366,161), P = (621,522) and CP = (447,412). For this experiment, each time one of the cities' location is assumed to be unknown, where all other locations and all decorrelations are known. This is

equivalent to having multiple ENF databases spread throughout the grid and one audio recording is provided, from which the location needs to be estimated. The location will be determined using system (7.7).

Figure 7.4 shows the actual locations of each of the cities in red and the estimated location based on the data in blue. From this it can be concluded that although the accuracy is far from good, each city is mapped roughly in the right area with respect to the four anchor cities. Also, it should be noticed that the location estimation error seems to be the largest for the outlier cities and that the cities in the middle (R and CP) are localized more accurately. In figure 7.5 a strong linear relation can be observed between the error of the estimation and the total distance of that city to the other four.

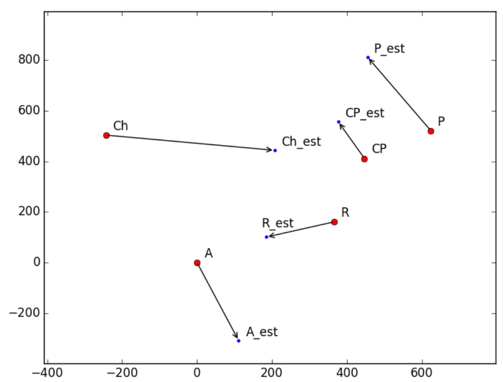


Figure 7.4: City location estimation using the Least Squares approach

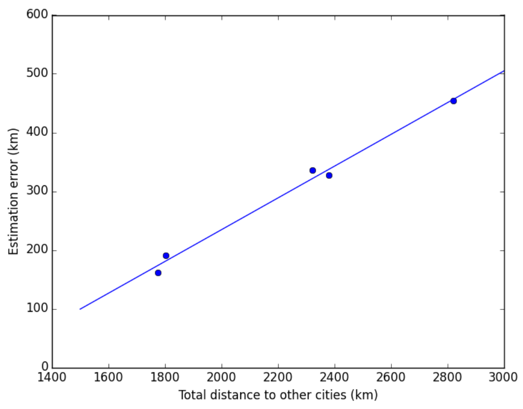


Figure 7.5: Error vs distance

A recommendation for applying this model is therefore to create ENF databases at many evenly spaced locations to increases the chances of have an ENF database close to the unknown location of the recording. Concerning the accuracy, the linear regression in figure 7.3 indicates what estimation error to expect. For this set-up the estimation errors are in the range of 160 to 450 km, where the anchor city distances are in the range of 200 to 865 km.

One weakness of this model may be that a constant parameters a is assumed, which implies that anywhere on the grid a certain distance between ENF measurement points translates to a correlation of $a \cdot |X - X_i|$. However, the physical distance between two locations may be something else than 'electrical' distance between them, due to the grid topology. The next model is a refinement of the previous two.

7.3. Apollonian circles localization

The Apollonian circles model is an improved recording localization model. Inspired by the half-plane intersection and improved on the global proportionality model, this model still assumes a linear relation between physical distance and decorrelation coefficients. However, each pair of known locations generates a set of possible points, which is a circle. A function is minimized to find the minimum distance between a point X and all generated circles.

This model also uses the decorrelation of each known anchor city with respect to the unknown locations' ENF. Considering two locations X_i and X_j and the unknown location X with decorrelations c_i and c_j, the decorrelation ratio $\frac{c_i}{c_j}$ is set equal to the ratio of the estimated distance $\frac{|X-X_i|}{|X-X_j|}$. The collection of points satisfying this relation is known to be the Apollonian circle, named after the Greek geometer who lived in the third century before Christ. Applying this model to all sets of anchor cities, there are $\frac{n(n-1)}{2}$ circles which in theory should all intersect in the desired point X. Because of noise and approximations, this will of course not be the case. To find a best solution to the problem, a function is minimized. The equation for one Apollonian circle is given by:

$$\frac{|X-X_i|^2}{|X-X_j|^2} = \left(\frac{c_i}{c_j}\right)^2 \tag{7.8}$$

Hence the least squared solution that minimizes the error of a point X with respect to all $\frac{n(n-1)}{2}$ Apollonian circles is given by:

$$\sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \left(\left(\frac{c_i}{c_j}\right)^2 \left((x_j-x)^2 + (y_j-y)^2\right) - \left((x_i-x)^2 + (y_i-y)^2\right) \right)^2 \tag{7.9}$$

Other metrics than the square norm have been studied. However, it was found that this norm yields slightly better results than for instance summing the absolute values in stead of the squares.

Equation 7.9 is not linear as was the case with the previous model. However, this forth degree polynomial with two variables can be solved using general non-linear solvers. To illustrate the Apollonian circle and the effect of a displacement with respect to the circle on the minimizing function, figure 7.6 shows such an example. There are two locations: $X_1 = (3,2)$ and $X_2 = (10,6)$. Furthermore, the decorrelation ratio is given to be $\frac{c_1}{c_2} = 2$. The Apollonian circle consists of all points that are twice as close to X_2 as to X_1. The colors indicate the value of the minimizing function, which is just one term in stead of the entire sum as above. The small values are shown in blue and large values are shown in dark red.

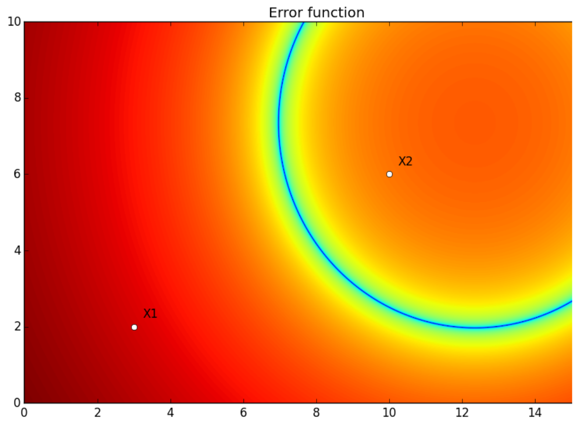


Figure 7.6: Heat map of the minimizing function showing a single Apollonian circle

7.3.1. Example

The same data set is used for localization of the five cities. Again, first the localization of CP will be attempted. This will also show the actual Apollonian circles . As a result it can easily be seen which circles give good locations and which circle do not contribute to an accurate result. Especially the largest circle, generated by the cities Ch and A, is a great distance off the correct location. This was also the case with the half-plane intersection model. Only in this case it only slightly affect the estimation, in stead of providing a wrong region all together.

Figures 7.7 and 7.8 show the six circles, which indeed do not all intersect in one point, but it is clear that certain regions are relatively close to all circles. On the right the heat map is shown, visualizing the value of the minimizing function.

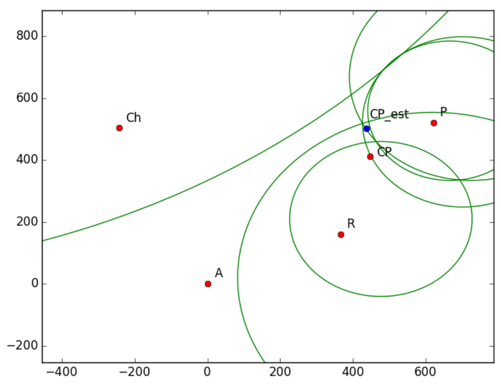


Figure 7.7: Apollonian circles to localize CP

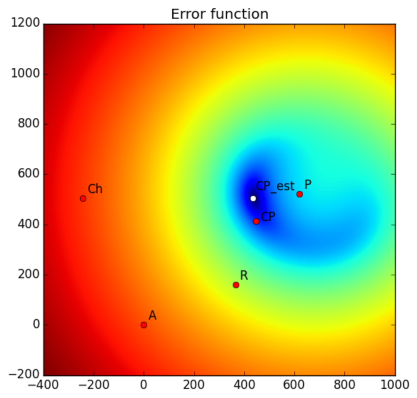


Figure 7.8: Heat map of the minimizing function on CP

The results of the localization of all ENF signals, using the other 4 cities as anchor locations are shown in figure 7.9.

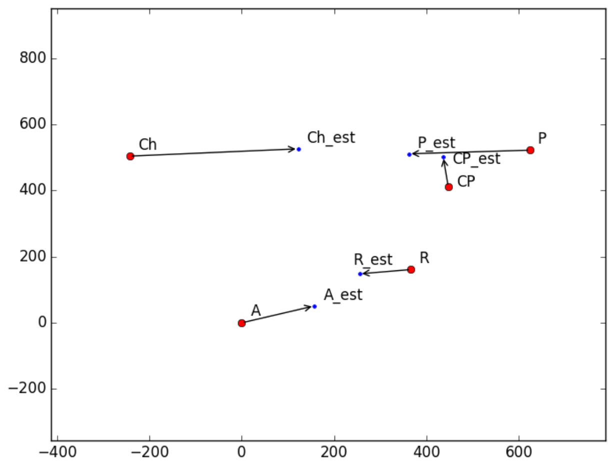


Figure 7.9: City location estimation using the Apollonian circles

The estimation errors are now in the range of 90 to 365 km, which is an improvement to the 160 to 450 km range of the global proportionality model. Using the Apollonian circles each city is actually placed nearer to its original location than using the global proportionality model. Still, the center cities are better localized than the cities on the outside.

8 Discussion

The ENF criterion is a fairly new field of research with only twelve years of history. On many different topics within the ENF study, the knowledge is still increasing and there are many topics left to research and optimize. Among those noise filtering may be the most important in order to increase timestamping accuracy. Fortunately there is already much knowledge about digital filters and such. Many techniques are developed which are not treated in this report. There were five research questions which are answered in this report. A brief summary of the findings will be given:

1. Can the ENF extraction accuracy be improved by decimating the voltage signal to a higher sampling rate? It is common practice to decimate the voltage signal from its original 44.1 kHz to a sampling rate of 300 Hz. It is always assumed this will not result in any significant loss of information. The results from this report however show that decimation does negatively influences the ENF estimation accuracy and the sampling rate of 1 kHz is proposed. Using a voltage signal of 1 kHz rather than 300 Hz consistently decreases the estimation error by .1 mHz.

2. Can the ENF extraction be accelerated by using the DFT rather than the FFT? The DFT with a complexity of $\mathcal{O}(n^2)$ is a more primitive algorithm than the FFT, which is an $\mathcal{O}(nlog(n))$ algorithm. For this reason the FFT is used in all literature. However, for estimating the ENF of one segment using the FFT 10^5 Fourier coefficients are calculated. The approach in this report uses the DFT which is able to obtain the same frequency estimate using only 10 to 41 Fourier coefficients. Due to this large difference in the number of calculated coefficients, a simple DFT implementation is approximately a factor 10 faster than an optimized FFT implementation. So indeed the extraction procedure has been accelerated significantly.

3. Which frequency resolution is to be chosen in combination with quadratic interpolation? The generally accepted procedure includes using the FFT on the voltage signal which is zero-padded with a factor 4. The quadratic interpolation technique is then used to enhance the frequency estimation performance. The procedure proposed in this report used the DFT, which renders the zero-padding obsolete. A frequency resolution can be chosen freely. The factor 4 zero-padded FFT approach yields a frequency resolution of 20 mHz. Experiments were conducted to investigate the results of enhancing the frequency resolution. It was found that a resolution of 10 mHz decreases the ENF estimation error by .1 mHz.

4. Can the ENF signal also be accurately extracted from video recordings? Most videos are recorded at a frame rate of about 30 fps. This frame rate is lower than the Nyquist frequency corresponding to the 100 Hz light flickering. This results in the alias effect which makes it very hard to estimate the ENF correctly from the average pixel intensity of the video frames. However, most cameras are equipped with CMOS sensors which writes frames one row at the time. This may result in the phenomenon known as the rolling shutter effect. Considering each pixel row individually the ENF can be very accurately extracted from the signal consisting of the average pixel intensity of each row. Just as with audio recordings, the video most be recorded under specific conditions. Furthermore, many video recorders automatically filter out the fluorescent light flickering. Therefore the applications of video ENF extraction are as limited as the applications of audio ENF extraction.

5. Is it possible to accurately estimate the location at which the recording was made? Previously to this report one localization technique has been developed, called the half-plane intersection. Multiple ENF databases are set up at different locations. The correlation coefficients of the ENF signal of the unknown source with respect to these databases are used to generate a region. The unknown location of the recording is then estimated to be within this region. The regions are large and often incorrect. In this report two methods are developed to return a point estimate of the location of a recording. Assuming that an inversely proportional relation between the correlation of two signals and the distance holds globally, a localization technique is proposed. Using only 4 reference databases, which are 250 to 1250 km apart, the unknown locations are estimated with an error in the range of 160 to 450 km. Further refinement of this technique resulted in the Apollonian circles method. The improvement in the Apollonian circles method is due to the fact that no longer a global proportionality is assumed. In rural areas the grid topology is different from the grid topology in urban areas. For each pair of databases, an Apollonian circle are drawn and the location estimation is the point that minimizes the distance to all circles. With the same data set, the results have improved to an estimation error of 90 to 365 km.

There is an ongoing debate whether to use the Cross Correlation or the Mean Squared Error as a database matching norm. But this report has shown that these two are equivalent when comparing two zero mean vectors, so that should conclude that discussion. In this report it is advised to use the sum of squared errors of the zero-mean vectors when executing the database matching, to minimize computation time.

It was attempted to gather accurate ENF data from multiple locations in Europe to verify the results of the localization techniques and to test the applicability of the theory. Due to a limited budget it was not possible to set up the required databases and cheaper alternatives were not accurate enough. It was concluded that estimating the geo-location of audio- and video recordings requires very high accuracy, even better than 1 mHz. This also holds for the recording which needs to be localized. Digital videos capturing fluorescent light may yield the most accurate ENF values in real cases.

A
Python code

```
from __future__ import division
#**********************************************************************
#
# record.py records audio from the connection microphone and saves it
# as a .wav file. The time is synchronized so that an accurate
# time of recording is available
# Tom Baksteen (c) 2015
#
#**********************************************************************
import pyaudio
import wave
from pylab import *
import datetime
from time import sleep
import ntplib
import numpy as np

# Parameters
RECORD_SECONDS = 3*60
fs = 2**12

# recording sound from mic to wav file
CHUNK = 2**10
FORMAT = pyaudio.paInt32
CHANNELS = 1
RATE = fs
p = pyaudio.PyAudio()

raw_input('press Enter to start recording...')
print("recording", RECORD_SECONDS, 'seconds of sound at',RATE, 'Hz rate')
comments = ''
stream = p.open(format = FORMAT,
    channels = CHANNELS,
    rate = RATE,
    input = True,
    frames_per_buffer = CHUNK)
frames = []
now = datetime.datetime.now()
print(now.strftime("%m-%d-%H:%M:%S:%f"))
```

```
# Trying to establish a time synchronisation.
# Local computer time may deviate significantly from server time.
try:
    client = ntplib.NTPClient()
    response = client.request('pool.ntp.org', version=3)
    offset = response.offset
    print("Offset is:",offset)
    ms=int(now.microsecond+10**6*(offset % 1))
    s=int(now.second+int(offset)+(ms//10**6)-(offset<0))
    m=int(now.minute+s//60)
    h=int(now.hour+m//60)
    d=int(now.day+h//24)

    now = now.replace(microsecond=int(ms % 10**6),
                      second=int(s % 60),
                      minute=int(m % 60),
                      hour=int(h % 24),)
    comments += "Recording on "+ now.strftime("%m-%d-%H:%M:%S:%f")+" for "+\
                str(RECORD_SECONDS)+" seconds\n"
    sleep(1-now.microsecond/10**6)
    now = now.replace(microsecond=0,second=(now.second+1) % 60,
                     hour=now.hour+((now.second+1) // 60))  # time after sleeping
    timestr = now.strftime("%Y%m%d%H%M%S")
except:
    comments += "Could not connect with time server\n Time is NOT synced. " \
                "Computer time is "+ now.strftime("%m-%d-%H:%M:%S:%f")
    timestr = now.strftime("%Y%m%d%H%M%S")+"offset"
    print("Started on "+ timestr +" local computer time.")

WAVE_OUTPUT_FILENAME = "output_"+ timestr + ".wav"
print("Starting at",now.strftime("%m-%d-%H:%M:%S:%f"))
for i in range(0, int(RATE / CHUNK * RECORD_SECONDS)):
    data = stream.read(CHUNK)
    frames.append(data)
print("finished recording")
stream.close()
p.terminate()


# writing recorded wav file
print("Saving data")
data = b"".join(frames)
wf = wave.open(WAVE_OUTPUT_FILENAME, "wb")
wf.setnchannels(CHANNELS)
wf.setsampwidth(p.get_sample_size(FORMAT))
wf.setframerate(RATE)
wf.writeframes(data)
wf.close()

################################################################################
```

```
from __future__ import division
#*********************************************************************
#
# full_fft.py The ENF signal is calculated from a .wav file using the FFT
#
# Tom Baksteen (c) 2015
#
#*********************************************************************
from pylab import *
import numpy as np
import scipy as sp
from scipy.io.wavfile import read

# Parameters
timestr = "output_20150629191531_Overschie_ma"
filename = timestr+".wav"
record_seconds = 3600                    # How long is the recording?
bp = 0                                   # apply the bandpass filter?
lowcut = 49.8
highcut = 50.2
padding_factor = 9                                  # Paddingfactor.
overlap = 9
comments = ''

# reading recorded wav file
fs, signal = read(filename)
signal = signal/float(2**32)/2      # Convert signal to volts

# applying band-pass filter 49.5-50.5 Hz
if bp==1:
    ft = np.fft.fft(signal)
    new_ft = ft
    new_ft[:int(lowcut*record_seconds)] = 0
    new_ft[int(highcut*record_seconds):len(new_ft)-int(highcut*record_seconds)] = 0
    new_ft[len(new_ft)-int(lowcut*record_seconds):] = 0
    signal = np.fft.ifft(new_ft)

# STFT
print('Performing STFT')
hopsize = fs                                            # 1 second
framesize = (overlap+1) * hopsize
comments += "Overlap is "+str(framesize/hopsize-1)+' seconds\n'
total_segments = record_seconds-overlap                 # Throw away last bad seconds
w = ones(framesize)
comments += "Padding factor: "+str(padding_factor+1)+'\n'
padding = np.zeros(framesize*padding_factor)
ext_signal = np.append(signal, np.zeros(framesize))      # signal including zero's
freq = np.zeros((framesize*(padding_factor+1)/fs+1, total_segments), dtype=np.float32)
freq_val = zeros(total_segments)

for i in arange(total_segments):
    hop = i * hopsize
    segment = ext_signal[hop:hop+framesize]             # Rectangular window
    padded = np.append(segment, padding)
    spectrum = abs(np.fft.fft(padded))
    freq[:,i] = spectrum[49.5/fs*framesize*
```

```
                        (padding_factor+1):50.5/fs*framesize*(padding_factor+1)+1]

# Quatratic interpolation
print('Quadratic interpolation')
for i in arange(total_segments):
    m = np.argmax(freq[:,i])
    x = [m-1,m,m+1]
    if m!=0 and m!= len(freq)-1:    # Top has to be away from border, else freq is too much off
        y = freq[x,i]
        p = .5*(y[0]-y[2])/(y[0]-2*y[1]+y[2])
        freq_val[i] = 49.5+1./(framesize*(padding_factor+1))*fs*(m+p)
    else:
        freq_val[i]  = nan
        print("The",i,'\'th second was too noisy. Freq is set to nan')
        comments += "The "+str(i)+'\'th second was too noisy. Freq is set to nan\n'

# print("plotting data and spectrogram")
# NFFT = fs
# NOVERLAP = fs/2
# figure("spectogram")
# Pxx,freqs,bins,im = specgram(signal,NFFT=NFFT,Fs=fs,noverlap=NOVERLAP,
#                          cmap=get_cmap('jet'), interpolation='none', scale='linear')
# axis([0.0,record_seconds,40,60])
# ylabel('frequency, [Hz]')
# xlabel('time, [s]')
# # savefig("Saved_data/"+timestr+'_spec.pdf')

figure("Frequency deviation")
plot(freq_val-50)
ylabel('frequency deviation, [Hz]')
xlabel('time, [s]')
savefig("Saved_data/"+timestr+'_freq.jpg')

show()

# saving values
savetxt("Saved_data/"+timestr,freq_val,fmt='%.4f')
# writing comments
comment_file = open("Saved_data/"+timestr+'_comments','w')
comment_file.write(comments)
comment_file.close()

################################################################################
```

```
from __future__ import division
#*********************************************************************
#
# small_dft.py The ENF signal is calculated from a given .wav file using the DFT
#
# Tom Baksteen (c) 2015
#
#*********************************************************************
from pylab import *
import numpy as np
import scipy as sp
from scipy.io.wavfile import read
from pymedia import *
import datetime

# Parameters
timestr = "output_20150917122045"
filename = timestr+".wav"
record_seconds = 3*60                    # How long is the recording?
bp = 0                                   # apply the bandpass filter?
lowcut = 49.8
highcut = 50.2
# (overlap+1) needs to be a divisor of 100. Admitted are: 0,1,3,4,9,19
overlap = 9
# Paddingfactor. We don't zero pad, but freq resolution is 1/((pf+1)(overlap+1))
pf = 100/(overlap+1)-1
comments = ''


# Quatratic interpolation
def quadratic_interpolation(sec=0):
    m = np.argmax(coef)
    x = [m-1,m,m+1]
    if m!=0 and m!= len(coef)-1:  # Top has to be away from border, else freq is too much off
        y = coef[x]
        p = .5*(y[0]-y[2])/(y[0]-2*y[1]+y[2])
        if sec==0:
            freq_val[sec] = 49.8+.01*(m+p)
        else:
            freq_val[sec] = round(freq_val[sec-1],2)+.01*(m-5+p)
    else:
        freq_val[sec] = 50
        print("The",sec,'\'th second was too noisy. Freq is set to nan')
    return


# Defines the 11 freq bins to evaluate based on the assumption
# that the ENF never changes more than 50 mHz in 1 second.
def freq_range(i):
    return arange(int(round(freq_val[i-1]-49.8,2)*100-5),int(round(freq_val[i-1]-49.8,2)*100+6))

# reading recorded wav file
fs, signal = read(filename)
# signal = signal[:,0]                   # If needed, converts stereo sound to mono

# applying band-pass filter when preferred
```

```
if bp==1:
    from scipy.signal import freqz, butter, lfilter
    b, a = butter(3, [lowcut/(.5*fs), highcut/(.5*fs)], btype="band")
    g, h = freqz(b, a)
    signal = lfilter(b, a, signal)

print('Performing short time Small DFT')
hopsize = fs                                        # 1 second is the jump step
framesize = (overlap+1) * hopsize
N = framesize*(pf+1)                                # Capital N according to the DFT definition
comments += "Overlap is "+str(framesize/hopsize-1)+' seconds\n'
# The last few seconds are bad. Too short a framesize for good freq. estimation
total_segments = record_seconds-overlap
w = np.ones(framesize)                              # Rectangular window
freq_val = zeros(total_segments)
n = arange(framesize)
omega = arange(49.80,50.21,0.01)                    # frequency bins
A = np.exp(-2*np.pi*1j/fs*np.outer(omega,n))        # precompute the small_dft matrix

# First frequency estimation
f = w*signal[:framesize]
coef = np.abs(np.dot(A,f))
quadratic_interpolation(sec=0)

for i in arange(1,total_segments):
    hop = i * hopsize
    segment = signal[hop:hop+framesize]
    f = segment*w
    coef = np.abs(np.dot(A[freq_range(i),:],f))
    quadratic_interpolation(sec=i)

print("plotting data and spectrogram")
# NFFT = fs
# NOVERLAP = fs/2
#
# figure("Spectrogram")
# Pxx,freqs,bins,im = specgram(signal,NFFT=NFFT,Fs=fs,noverlap=NOVERLAP,
#                              cmap=get_cmap('jet'), interpolation='none', scale='linear')
# axis([0.0,record_seconds,0,500])
# ylabel('frequency, [Hz]')
# xlabel('time, [s]')

figure("Frequency deviation")
plot(freq_val-50)
ylabel('frequency deviation, [Hz]')
xlabel('time, [s]')
savefig("Saved_data/"+timestr+'_freq.pdf')

show()

# saving values
savetxt("Saved_data/"+timestr,freq_val,fmt='%.4f')
# writing comments
comment_file = open("Saved_data/"+timestr+'_comments','w')
comment_file.write(comments)
comment_file.close()
```

```
#####################################################################################

from __future__ import division
#*********************************************************************
#
# circles.py Apollonian circles are calculated and drawn. Estimated location is drawn.
#
# Tom Baksteen (c) 2015
#
#*********************************************************************
__author__ = 'TomNCIM'

import numpy as np
from matplotlib.pyplot import *
from numpy import arange
from numpy import meshgrid

cities = ["CP", 'Ch', 'A', 'R', 'P']

# City order is CP, Ch, A, R, P
xcoord = np.array([447,-243,0,366,621])
ycoord = np.array([412, 504,0,161,522])
f = np.array([.24,.27,.13,.07,.32,.26,.29,.12,.36,.24])  # all decorrelations
n = len(xcoord)

delta = 10
xrange = arange(-5000.0, 10000.0, delta)
yrange = arange(-5000.0, 12000.0, delta)
X, Y = meshgrid(xrange,yrange)

figure("Circles")

# Where is CP? Apollonian crcles in contour plot
for i in range(1,4):
    for j in range(i+1,5):
        F = (xcoord[i]-X)**2+(ycoord[i]-Y)**2
        G = (f[i-1]/f[j-1])**2*((xcoord[j]-X)**2+(ycoord[j]-Y)**2)
        contour(X, Y, (F - G), [0], colors='g')

real = plot(xcoord,ycoord, 'ro')
plot(436, 503, 'bo')

# annotate the cities
for i in range(len(cities)):
    if i ==0:
        annotate(cities[i], xy=(xcoord[i],ycoord[i]), xytext=(xcoord[i]+20,ycoord[i]))
    else:
        annotate(cities[i], xy=(xcoord[i],ycoord[i]), xytext=(xcoord[i]+20,ycoord[i]+20))
annotate("CP_est", xy=(426,518), xytext=(446,538))

show()

#####################################################################################

from __future__ import division
#*********************************************************************
```

```
#
# countour_cirlces.py Calculate and show the minimizing function for the circles model
#
# Tom Baksteen (c) 2015
#
#*******************************************************************
__author__ = 'TomNCIM'

import matplotlib
import numpy as np
import matplotlib.cm as cm
import matplotlib.mlab as mlab
import matplotlib.pyplot as plt

cities = ["CP", 'Ch', 'A', 'R', 'P']

# City order is CP, Ch, A, R, P
xcoord = np.array([447,-243,0,366,621])
ycoord = np.array([412, 504,0,161,522])

delta = 1
xrange = np.arange(-400, 1000.0, delta)
yrange = np.arange(-200, 1200.0, delta)
x, y = np.meshgrid(xrange,yrange)

Z = abs((.24/.27)**2*np.sqrt((0-x)**2+(0-y)**2)-np.sqrt((-243-x)**2+(504-y)**2))**2+\
    abs((.24/.13)**2*np.sqrt((366-x)**2+(161-y)**2)-np.sqrt((-243-x)**2+(504-y)**2))**2 +\
    abs((.24/.07)**2*np.sqrt((621-x)**2+(522-y)**2)-np.sqrt((-243-x)**2+(504-y)**2))**2 + \
    abs((.27/.13**2)*np.sqrt((366-x)**2+(161-y)**2)-np.sqrt((0-x)**2+(0-y)**2))**2 + \
    abs((.27/.07)**2*np.sqrt((621-x)**2+(522-y)**2)-np.sqrt((0-x)**2+(0-y)**2))**2 + \
    abs((.13/.07)**2*np.sqrt((621-x)**2+(522-y)**2)-np.sqrt((366-x)**2+(161-y)**2))**2

plt.figure()
im = plt.imshow(np.log10(Z), interpolation='bilinear', origin='lower',
                cmap=cm.jet, extent=(-400,1000,-200,1200))

plt.title('Error function')

real = plt.plot(xcoord,ycoord, 'ro')
plt.plot(436, 503, 'wo')

for i in range(len(cities)):
    if i ==0:
        plt.annotate(cities[i], xy=(xcoord[i],ycoord[i]), xytext=(xcoord[i]+20,ycoord[i]))
    else:
        plt.annotate(cities[i], xy=(xcoord[i],ycoord[i]), xytext=(xcoord[i]+20,ycoord[i]+20))
plt.annotate("CP_est", xy=(426,518), xytext=(446,538))

plt.show()

##########################################################################################
```

```
from __future__ import division
#*********************************************************************
#
# video_rows_dft.py The ENF signal is calculated from the rows pixel intensity
# of CMOS videos capturing fluoresent light. The DFT is used.
# Tom Baksteen (c) 2015
#
#*********************************************************************
__author__ = 'TomNCIM'

import numpy as np
import cv2
from matplotlib.pyplot import *
from pylab import *
import scipy as sp
from scipy.signal import butter, lfilter, freqs

xmin = 0
xmax = 640
ymin = 0
ymax = 360

name = "youtube"
filename = "Saved_data_video/"+name+".mp4"
record_seconds = 41*60+30
fs = 29.97                          # frame rate of the video
overlap = 9                         # window overlap
pf = 100/(overlap+1)-1              # virtual padding_factor. Yields 10 mHz DFT resolution
bp = 0                              # Apply the band-pass filter?
lowcut = 99.8
highcut = 100.2


# Quatratic interpolation
def quadratic_interpolation(sec=0):
    m = np.argmax(coef)
    x = [m-1,m,m+1]
    if m!=0 and m!= len(coef)-1:
        y = coef[x]
        p = .5*(y[0]-y[2])/(y[0]-2*y[1]+y[2])
        # if sec==0:
        freq_val[sec] = 99.5+.01*(m+p)
        # else:
        #     freq_val[sec] = round(freq_val[sec-1],2)+.01*(m-5+p)
    else:
        freq_val[sec] = 100
        print("The",sec,'\'th second was too noisy. Freq is set to nan')
    return

def butter_bandpass_filter(data, low, high, rate, order=5):
    b, a = butter(order, [low/(.5*rate), high/(.5*rate)], btype="band", analog=True)
    y = lfilter(b, a, data)
    return y


# Defines the 11 freq bins to evaluate based on the assumption that
# the ENF never changes more than 50 mHz in 1 second.
def freq_range(i):
    return arange(int(round(freq_val[i-1]-99.5,2)*100-5),int(round(freq_val[i-1]-99.5,2)*100+6))

# Read images from avi file
video = cv2.VideoCapture(filename)
fs *= ymax                      # From now on each row in a frame is considered a frame!
intensity = np.zeros(int(record_seconds*fs))
i=0

print("Calculating frame rows intensity. This may take a minute...")
while True:
    ret, frame = video.read()
    if ret:
        # Convert to grayscale
        frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        # Calculate average pixel intensity per row
        for j in range(ymin,ymax):
            intensity[i*(ymax-ymin)+j] = np.mean(frame[j,xmin:xmax])
    else:
        break
    i+=1
    # figure('horizontal slice')
    # plot(frame[100,:])
    # show()
    if cv2.waitKey(0) & 0xFF == ord('q'):
            break
video.release()
cv2.destroyAllWindows()

# Delete first few frame intensities which are often bad
intensity = intensity[3*fs:]
record_seconds -= 3

# saving values
savetxt("Saved_data_video/name",intensity,fmt='%.4f')

# Applying bandpass filter
if bp:
    from scipy.signal import freqz
    b, a = butter(3, [lowcut/(.5*fs), highcut/(.5*fs)], btype="band")
    w, h = freqz(b, a)
    intensity = lfilter(b, a, intensity)

print('Performing short time Small DFT')
hopsize = fs                                # 1 second is the jump step
framesize = (overlap+1) * hopsize
N = framesize*(pf+1)                        # Capital N according to the DFT definition
total_segments = record_seconds-overlap     # The last few seconds yield too short frame
freq_val = zeros(total_segments)

n = arange(framesize)
omega = arange(99.50,100.51,0.01)
A = np.exp(-2*np.pi*1j/fs*np.outer(omega,n))            # precompute the small_dft matrix

# First frequency estimation
```

```
f = intensity[:framesize]
coef = np.abs(np.dot(A, f))
quadratic_interpolation(sec=0)

for i in arange(1,total_segments):
    hop = i * hopsize
    segment = intensity[hop:hop+framesize]
    f = segment
    coef = np.abs(np.dot(A, f))
    # figure('coef')
    # plot(coef)
    # show()
    quadratic_interpolation(sec=i)

print("plotting data and spectrogram")
# NFFT = fs
# NOVERLAP = fs/2
#
# figure("spectogram")
# Pxx, freqs, bins, im = specgram(intensity,NFFT=NFFT, Fs=fs, noverlap=NOVERLAP,
#                                cmap=get_cmap('jet'), interpolation='none')
# axis([0.0,record_seconds,0,200])
# ylabel('frequency, [Hz]')
# xlabel('time, [s]')
# # savefig("Saved_data_video/"+timestr+'_spec.pdf')

figure("Frequency deviation on video")
plot(.5*(freq_val-100))
ylabel('frequency deviation, [Hz]')
xlabel('time, [s]')
# savefig("Saved_data_video/"+timestr+'_freq.pdf')

# saving values
savetxt("Saved_data_video/"+name, freq_val, fmt='%.4f')

show()

###############################################################################

from __future__ import division
#*********************************************************************
#
# interdatabase.py calculates the interdatabase error of a given database
# to determine an upper bound for the error of a match.
# Tom Baksteen (c) 2015
#
#*********************************************************************
__author__ = 'TomNCIM'

from numpy import *
from random import randint
from matplotlib.pyplot import *

data = loadtxt("sept")
segment_size = 300
N = 1000000
```

```
error = np.zeros(N)
i=0

while i<N:
    # random segment 1
    r1 = randint(0,len(data)-segment_size)
    segment1 = data[r1:r1+segment_size]
    # random segment 2
    r2 = randint(0,len(data)-segment_size)
    # segments must have at least 10 seconds of non-overlap
    if abs(r1-r2)>10:
        segment2 = data[r2:r2+segment_size]
        dif = abs(segment1-segment2)
        error[i] = (sum(dif))
        i+=1

hist(error,1000)
xlabel('Sum of the errors')
show()

# print error

##################################################################################

from __future__ import division
#*********************************************************************
#
# half_plane.py draws the half-planes to calculate a region in which
# the unknown location of a recording is supposed to be
# Tom Baksteen (c) 2015
#
#*********************************************************************
__author__ = 'TomNCIM'

from matplotlib.pyplot import *

cities = ["CP", 'Ch', 'A', 'R', 'P']

realx = [447,-243,0,366,621]
realy = [412,504,0,161,522]

estx = [436,122,157,255,362]
esty = [503,526,51,148,511]

n = len(cities)

fig = figure("Least Squares Approximation")
ax = fig.add_subplot(111)

real = ax.plot(realx, realy, 'ro')
for i in range(n):
    if i ==0:
        ax.annotate(cities[i], xy=(realx[i],realy[i]), xytext=(realx[i]+20,realy[i]))
    else:
        ax.annotate(cities[i], xy=(realx[i],realy[i]), xytext=(realx[i]+20,realy[i]+20))
```

```
for i in range(1,n−1):
    for j in range(i+1,n):
        dx = realx[i]−realx[j]
        dy = realy[i]−realy[j]
        a = −dx/dy
        b = realy[j]+dy/2−a*(realx[j]+dx/2)
        ax.plot([−1000,1000],[a*−1000+b,a*1000+b],'k')

show()

##################################################################################

from __future__ import division
#*********************************************************************
#
# matching.py matches an ENF segment with a given database and the error
# distribution is plotted to illustrate the match' strength
# Tom Baksteen (c) 2015
#
#*********************************************************************
__author__ = 'TomNCIM'

from scipy import interpolate
from numpy import *
from matplotlib.pyplot import *
import scipy
import pylab

# Load values
data = loadtxt("Saved_data/enf74")
freq_sample = data[1:300]
data2 = loadtxt("Saved_data/database.txt")
freq_db = data2

freq_sample = freq_sample−average(freq_sample)    # zero−mean

db_len = len(freq_db)
sample_len = len(freq_sample)
segment = zeros(sample_len)
err = zeros(db_len−sample_len+1)

# values not to be used in the matching process
# bad_sectors = arange(45,48)
# bad_sectors = append(bad_sectors,arange(620,624))
bad_sectors = []
p = 1                                         # set the distance norm

for i in range(0,db_len−sample_len+1):
    segment = freq_db[i:i+sample_len]
    segment = segment−average(segment)        # zero−mean db segment; can be optimized
    dif = abs(segment − freq_sample)
    dif[bad_sectors] = 0
    err[i] = (sum(dif**2))                    # p−norm of diff

norm_err = np.log10(err)                      # transformation to approximate normal dist.
```

```
j = argmin(norm_err)
norm_err[j-10:j] = average(norm_err)
norm_err[j+1:j+11] = average(norm_err)
st = std(norm_err)
m = mean(norm_err)
z = (m-norm_err[j])/st
print("The best match score is",err[j],"located at",j,"seconds with an error z-score of",z)

# # Horizontal shift refinement
# shifted_err = zeros(200)
# for x in arange(-1,0,.01):
#     match = append(freq_db[j],[(-x)*freq_db[j+i]+(1+x)*freq_db[j+i+1]
#                                   for i in range(sample_len-1)])
#     match -= mean(match)
#     shifted_err[int(100*(x+1))] = (sum(abs(match-freq_sample)**p))**(1/p)
#     print("Error at int was",err[j],"but shifted:",(sum(abs(match-freq_sample)**p))
#           **(1/p),"for x=",x)
# for x in arange(0,1,.01):
#     match = append([(1-x)*freq_db[j+i]+x*freq_db[j+i+1] for i in range(sample_len-1)],
#                    freq_db[j+sample_len-1])
#     match -= mean(match)
#     shifted_err[int(100*x+100)] = (sum(abs(match-freq_sample)**p))**(1/p)
#     print("Error at int was",err[j],"but shifted:",(sum(abs(match-freq_sample)**p))
#           **(1/p),"for x=",x)
#
# x = argmin(shifted_err)
# x = x/100-1
#
# if x>=0:
#     match = append([(1-x)*freq_db[j+i]+x*freq_db[j+i+1]
#                     for i in range(sample_len-1)],freq_db[j+sample_len-1])
# else:
#     match = append(freq_db[j],[(-x)*freq_db[j+i]+(1+x)*freq_db[j+i+1]
#                                   for i in range(sample_len-1)])
# match -= mean(match)
# print("Error at int was",err[j],"but shifted:",(sum(abs(match-freq_sample)**p))
#       **(1/p)/sample_len,"for x=",x)

figure('Error')
hist(err,100)
xlabel('Squared error')

figure('Log transform to normal')
hist(norm_err,100)
xlabel('Log of the squared error')

figure('Log error')
plot(norm_err)
xlabel('Database index (seconds)')
ylabel('Log of the error')
figure('Match')
plot(freq_db[j:j+len(freq_sample)]-average(freq_db[j:j+len(freq_sample)]),
     label='Database segment')                # normalized match
plot(freq_sample,label='Audio ENF')            # normalized sample
xlabel('Seconds')
```

```
ylabel('Zero mean ENF (Hz)')
legend(loc='upper left')

# Error histogram
'''fig = figure(2)
ax = fig.add_subplot(111)
ax.hist(norm_err,500)'''

show()

################################################################################

from __future__ import division
#*******************************************************************
#
# record_video.py records video from a webcam and synchronizes time with
# a server. The video is saved in gray-scale
# Tom Baksteen (c) 2015
#
#*******************************************************************
__author__ = 'TomNCIM'
import cv2
from pylab import *
import datetime as dt
import ntplib
from time import sleep

record_seconds = 600             # seconds
fs = 30

raw_input('press Enter to start recording...')
print("recording", record_seconds, 'seconds of video')
comments = ''

# Time synchronisation
now = dt.datetime.now()
print(now.strftime("%m-%d-%H:%M:%S:%f"))
try:
    client = ntplib.NTPClient()
    response = client.request('pool.ntp.org', version=3)
    offset = response.offset
    print("Offset is:",offset)
    ms=int(now.microsecond+10**6*(offset % 1))
    s=int(now.second+int(offset)+(ms//10**6)-(offset<0))
    m=int(now.minute+s//60)
    h=int(now.hour+m//60)
    d=int(now.day+h//24)

    now = now.replace(microsecond=int(ms % 10**6),
                      second=int(s % 60),
                      minute=int(m % 60),
                      hour=int(h % 24),)
    sleep(1-now.microsecond/10**6)
    now = now.replace(microsecond=0,second=(now.second+1) % 60,
                     hour=now.hour+((now.second+1) // 60))  # time after sleeping
    timestr = now.strftime("%Y%m%d%H%M%S")
    comments += "Recording on "+ now.strftime("%m-%d-%H:%M:%S:%f")\
                +" for "+str(record_seconds)+" seconds\n"
except:
    comments += "Could not connect with time server\n Time is NOT synced. " \
                "Computer time is "+ now.strftime("%m-%d-%H:%M:%S:%f")
    print("Started on "+ now.strftime("%m-%d-%H:%M:%S:%f")+" local computer time.")
    timestr = now.strftime("%Y%m%d%H%M%S")+"offset"

print("Starting at",now.strftime("%m-%d-%H:%M:%S:%f"))

# Capturing video
cap = cv2.VideoCapture(0)

fourcc = cv2.cv.CV_FOURCC(*'XVID')
writer = cv2.VideoWriter(filename="Saved_data_video/white_wall"+timestr+".avi",
                         fourcc=fourcc,fps=fs,frameSize=(640,480),isColor=0)

for i in arange(record_seconds*fs):
    # Capture frame-by-frame
    ret, frame = cap.read()

    # Gray please!
    frame = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)

    # Saving image to the video
    writer.write(frame)

    if cv2.waitKey(0) & 0xFF == ord('q'):
            break
# When everything done, release the capture
cap.release()
writer.release()
cv2.destroyAllWindows()

##################################################################################
```

Bibliography

[1] Anthony G Barnston. Correspondence among the correlation, rmse, and heidke forecast verification measures; refinement of the heidke score. *Weather and Forecasting*, 7(4):699–709, 1992.

[2] Math H Bollen and Irene Gu. *Signal processing of power quality disturbances*, volume 30. John Wiley & Sons, 2006.

[3] D. Bykhovsky and A. Cohen. Electrical network frequency (enf) maximum-likelihood estimation via a multitone harmonic model. *Information Forensics and Security, IEEE Transactions on*, 8(5):744–753, May 2013. ISSN 1556-6013. doi: 10.1109/TIFS.2013.2253462.

[4] Alan J Cooper. An automated approach to the electric network frequency (enf) criterion-theory and practice. *International Journal of Speech, Language and the Law*, 16(2):193–218, 2009.

[5] R. Garg, A.L. Varna, and Min Wu. Modeling and analysis of electric network frequency signal for timestamp verification. In *Information Forensics and Security (WIFS), 2012 IEEE International Workshop on*, pages 67–72, Dec 2012. doi: 10.1109/WIFS.2012.6412627.

[6] R. Garg, A.L. Varna, A. Hajj-Ahmad, and Min Wu. "seeing" enf: Power-signature-based timestamp for digital multimedia via optical sensing and signal processing. *Information Forensics and Security, IEEE Transactions on*, 8(9):1417–1432, Sept 2013. ISSN 1556-6013. doi: 10.1109/TIFS.2013.2272217.

[7] Ravi Garg, Adi Hajj-Ahmad, and Min Wu. Geo-location estimation from electrical network frequency signals. In *ICASSP*, pages 2862–2866, 2013.

[8] Catalin Grigoras. Digital audio recording analysis–the electric network frequency criterion. *International Journal of Speech Language and the Law*, 12(1):63–76, 2005.

[9] Liling Huang. Electromechanical wave propagation in large electric power systems. 2003.

[10] Maarten Huijbregtse and Zeno Geradts. Using the enf criterion for determining the time of recording of short digital audio recordings. In *Computational Forensics*, pages 116–124. Springer, 2009.

[11] Mateusz Kajstura, Agata Trawinska, and Jacek Hebenstreit. Application of the electrical network frequency (enf) criterion: A case of a digital recording. *Forensic science international*, 155(2):165–171, 2005.

[12] Alex Kantardjiev. *Determining the recording time of digital media by using the electric network frequency*. PhD thesis, Uppsala Universitet, 2011.

[13] D. Rife and R.R. Boorstyn. Single tone parameter estimation from discrete-time observations. *Information Theory, IEEE Transactions on*, 20(5):591–598, Sep 1974. ISSN 0018-9448. doi: 10.1109/TIT.1974.1055282.

[14] Birron Mathew Weedy, Brian John Cory, N Jenkins, JB Ekanayake, and G Strbac. *Electric power systems*. John Wiley & Sons, 2012.

[15] Yuehao Yan, Tianshu Bi, Liang Chen, and Qixun Yang. The cut-off frequency of disturbance propagation in discrete inertia model of power networks. In *Innovative Smart Grid Technologies Europe (ISGT EUROPE), 2013 4th IEEE/PES*, pages 1–5. IEEE, 2013.

[16] Yuehao Yan, Tianshu Bi, and Qixun Yang. The sectionalized homogeneous model of power systems and its analytical solution. In *Power and Energy Society General Meeting (PES), 2013 IEEE*, pages 1–5. IEEE, 2013.