# Towards a deeper understanding of the Visibility Graph algorithm

by

T.J. Alers

to obtain the degree of Master of Science
at the Delft University of Technology,
to be defended publicly on December 8, 2023 at 10:00 AM.


| | | |
|---|---|---|
| Student number: | 4585771 | |
| Project duration: | November 14, 2022 – December 8, 2023 | |
| Thesis committee: | Ir. R. Noldus, | Ericsson, TU Delft, supervisor |
| | Prof. dr. ir. R. Kooij, | TU Delft |
| | Dr. ir. H. Wang, | TU Delft |

An electronic version of this thesis is available at http://repository.tudelft.nl/.


TU Delft

# Preface

*We live in an increasingly data-driven world in which the ability to extract useful information from our data is crucial. One relatively new concept is to portray the data as a network of nodes that are connected based on the visibility between data points. Features of this 'visibility graph' can be analyzed in order to gain insight about the data. I would like to thank my supervisor Rogier Noldus for introducing me to this concept which, although able to produce accurate results in some use cases, is scarcely described mathematically. The goal of this thesis is to increase the understanding of which metrics are typical of a visibility graph, what sets them apart as a family of graphs, and how much information is retained in the visibility graph of a certain data set. For a duration of almost a year, I have delved into the abstract mathematical concept that is the VG. The curiosity-based objectives of this thesis are a far cry from the engineering projects that I have previously encountered in my studies. During my research I have written many MATLAB scripts to help me gain insight in the visibility graph algorithm. If anyone would like to continue the research on this topic (there certainly is room left to explore), do not hesitate to contact me at tim-alers@live.nl for my scripts thus far. I would like to thank my supervisor, my brother, and my parents for the support they have provided me and the input they gave me when discussing the content of my work.*

*T.J. Alers*
*Delft, December 2023*

# Abstract

The visibility graph has gained traction as a method for time series analysis. Through this method, it is possible to detect or quantify non-linear behavior in a myriad of fields and applications. This thesis explores the variations of visibility graphs, describes how they are used in literature and investigates what lies at the foundation of the transformation from time domain to graph domain. The algorithm for the construction of a visibility graph is used as the starting point. Through mathematical derivation, claims can be made regarding the degree metric of a visibility graph. The average degree for a discrete random time series is defined with high accuracy. Using the inequality sets that can be derived from the adjacency matrix of a visibility graph, certain patterns can be identified as being unattainable for a visibility graph. The information retention is investigated by comparing the Shannon Entropy of both a time series and the visibility graph that is constructed from it. The loss of information can be quantified, but only for short discrete series with low sampling depth. These findings are reported in this thesis, along with some insights that could be the subject of further research to deepen the understanding or enhance the use of the visibility graph algorithm.

Table 1: List of Abbreviations

| Abbreviation | Meaning |
|---|---|
| VG | Visibility Graph |
| A (matrix) | Adjacency matrix of a graph |
| NVG | Natural Visibility Graph |
| HVG | Horizontal Visibility Graph |
| CVG | Circular Visibility Graph |
| WVG | Weighted Visibility Graph |
| dVG | Directed Visibility Graph |
| LP | Limited Penetration |
| DP | Dual Perspective |
| (C)(LP)(DP)VG | (Circular) (Limited Penetration) (Dual Perspective) Visibility Graph |
| H | Hurst Exponent |
| (MF-)DFA | (Multifractal Detrended) Fluctuation Analysis |
| MLE | Maximum Likelihood Estimation |
| R/S (analysis) | Rescaled Range (Analysis) |
| fBm | Fractal Brownian Motion |
| QAM | Quardature Amplitude Multiplexing |
| BPSK | Binary Phase Shift Keying |
| SGN | Sub-Graph Network |
| SNR | Signal to Noise Ratio |
| $TV_G$ | Total Variation on Graph |
| ApEn | Approximate Entropy |
| PE | Permutation Entropy |
| KU | Kurtosis |
| RMS | Root Mean Square |
| DS | Degree Sequence |
| KLD | Kullback-Leiber divergence |
| LTE | Lower Triangle Entries |

Table 2: List of symbols, their names and meaning

| Symbol | Name | Meaning |
|---|---|---|
| $N$ (single value) | Nodes | The number of nodes in a graph |
| $\mathcal{N}$ (set) | Node set | The collection of nodes in a graph |
| $L$ (single value) | Links | The number of links in a graph |
| $\mathcal{L}$ (set) | Link set | The collection of links in a graph |
| $A$ | Adjacency matrix | Binary matrix which describes the connections in the graph |
| $a_{ij}$ | Adjacency matrix element | Binary element of the adjacency matrix where a $1$ indicates the existence of a link between nodes $i$ and $j$ |
| $W$ | Weighted Adjacency matrix | Matrix with non-binary elements that describes the connection in the graph |
| $\tau$ | Time delay | The time difference between samples that are used to create embedding vectors |
| $m$ | Embedding dimension | The length of an embedding vector |
| $\Theta(x)$ | Heaviside Function | Returns 0 for any negative $x$, returns 1 for any positive $x$ |
| $w_{ij}$ | Weight | The link weight attributed to the link between node $i$ and node $j$ |
| $k$ | Degree | The amount of neighbors a single node has |
| $P_G(k)$ | Degree Distribution | The distribution of degrees that are found in a graph G |
| $\gamma$ | Power of Scale-Freeness | The slope of the degree distribution of a scale-free graph |
| $cov(X,Y)$ | Covariance (X,Y) | The covariance between vector X and vector Y |
| $\sigma$ | Standard Deviation | The measure of how much entries differ from the mean of the group |
| $C$ | Coupling factor | The degree to which two chaotic systems influence each other |
| $f$ | Frequency | Number of oscillations per time unit |
| $f_s$ | Sampling Frequency | Frequency used to sample a continuous signal or process |
| $T$ | Period | Time between two identical points in a periodic signal |
| $E[X]$ | Expected Value | The expected value that is returned by stochastic process X |
| $C$ | Clustering Coefficient | The degree to which neighbors of a node are also neighboring each other |
| $H(X)$ | Shannon Entropy | A measure for the amount of information in a single realization of process X |
| $\rho_D$ | Degree Assortativity | The degree to which nodes tend to neighbor nodes with a similar degree |

# Contents

# 1 Introduction

In recent times, more and more events and processes are monitored for data collection. These measurements are used to make data-driven decisions or predictions that would otherwise not be possible. The collection of data over time can be stored as a string of values. A time series is such a collection of values. The values are in chronological order and are placed equidistant in time domain, i.e. the values are sampled or generated at a regular interval. The data are represented by vectors $T = [t_1, t_2, ..., t_N]$ and $X = [x_1, x_2, ..., x_N]$. In these vectors, $x_i$ represents the value of the time series at time $t_i$. Any sampled signal or regularly monitored process will result in such a series of data with certain values at different time instances.

There have been efforts to transform the data from the time domain to the graph domain in the form of Visibility Graphs and Recurrence Networks [1], [2]. A graph is a collection of nodes that are connected by edges. Data in a graph can be subject to analysis tools that can only be used in the graph domain, such as degree distribution, clustering coefficients, path analysis and others [3]. Transforming the data in a time series to the graph domain would allow for these analysis tools to provide new insights in the time series [1]. Such a transformation must define a set of nodes using the time series data, and then define some criterion for which two nodes are connected by an edge. A number of methods have been proposed for such a transformation.

This thesis delves into the Visibility Graphs (VGs), as a method of transforming a time series into a network of links and nodes. In these cases, each sampling point of the time series represents a node in the graph. Two nodes in the graph are connected by an edge if the two corresponding sampling points share a direct line of sight. The first VG method was proposed by Lacasa et al. in 2008 [1]. Since then, variations on this method have been proposed [4]–[6].

Since Lacasa's first work on the visibility graph, the theory has been used in a multitude of researches across various fields. There are many applications and processes in which data in a time series is analyzed. The advantage of transforming a time series to the graph domain is that the data can then be analyzed using graph analysis tools. The applications of the visibility graph to characterize data include brain scans, monitoring machine parts, and measurements of seismic activity [7]–[9]. The concept of the VG can also be used to accurately forecast time series in which non-linear or chaotic dynamics are present [10].

The fast adoption of the visibility graph in practice warrants further research. This thesis will explore the current state of the art for the transformation from time series to a visibility graph, and how the visibility graph plays a key role in the analysis of time series. The limitation of the current methods are highlighted in the literature review. This thesis will attempt to improve the understanding of the VG as a class of graphs. It investigates whether there are properties common to all VGs. Additionally, it is examined how much information about the time series is stored in the visibility graph. The reversibility of the transformation is investigated and alterations to the visibility graph algorithm and their implications to the retention of information are considered.

In Chapter 2, a selection of VG algorithms is introduced and the applications in scientific research are presented. It becomes clear that both the methods and fields of research in which VGs are used, are wide-ranging. Then some mathematical concepts that are important to the VG principle are detailed and visualized in Chapter 3, including which graph metrics are empirically found to be typical for differ-

ent VG implementations. Predictions for the degree metrics of a Natural VG (NVG), which is the method proposed by Lacasa et al. [1], are explored in Chapter 4. In Chapter 5, the gathered information about the NVG is used to find a method of identifying a graph as being the product of the NVG algorithm. Lastly, the goal of Chapter 6 is to find out how much information is lost in the transformation to the graph domain when using the NVG algorithm, and to what extent the transformation can be reversed. The thesis concludes with a summary of the findings and mentions some avenues of research that could be further explored in Chapter 7. Some suggestions for further research are presented in Chapter 8.

# 2

# Literature review

This chapter provides an overview of literature regarding the transformation of a time series to the graph domain. This includes variations of Recurrence Networks and of VGs. The chapter then further elaborates on the usage the VG as a tool to analyze a time series. The chapter is divided into three sections. First, multiple variations of the Recurrence Network and VG algorithms are presented. Secondly, a collection of work is analyzed in which the concept of the VG is used as a tool in time series in analysis. It will become evident that the VG tool is very flexible and can be used in numerous situations, although it often is only one part of a larger system of analysis. There are many pieces of scientific literature that make use of the relation between the degree distribution of the VG and the Hurst exponent of a time series, and this relation finds applications in numerous fields. After these are introduced, a number or publications that classify time series through various graph metrics of the VG are presented. Lastly, the limitations and recommendations found in these publications are summarized.

## 2.1. Recurrence networks

Recurrence networks are a method for representing data from a time series in a graph. The method allows for determining characteristics of chaotic time series created by dynamical systems [11]. The method is based on the notion that there are states within such systems that reoccur, and that such recurrences can be predicted by previous data.

### Recurrence plots

The structure of a recurrence network is based on the recurrence plot of a time series. For a discrete time series, there needs to be a phase space reconstruction to reconstruct the dynamical system variables. This is often done using the time delay method. Values of the time series that are a certain time $\tau$ apart, are gathered in a vector. These vectors indicate a trajectory of the series over time and are interpreted as a point in phase space. For a discrete time series $z(it)$, where $i = 1, 2, ..., M$ and $t$ is the time interval, vectors in phase space are created as follows [12]:

$$\mathbf{x_k} = [x_k(1), x_k(2), ..., x_k(m)] = [z(kt), z(kt+\tau), ..., z(kt+(m-1)\tau)] \tag{2.1}$$

Different vectors are created for $k = 1, 2, ..., N$ and $N = M-(m-1)\tau/t$. The values of $m$ (the embedding dimension) and $\tau$ (the time delay) have to be chosen carefully in order to obtain an accurate phase portrait. For the time delay, a suitable value can be found by determining the first local minimum in the average mutual information between the sets $z(it)$ and $z(it+\tau)$. This gives an indication of the amount of information about $z(it+\tau)$ that can be inferred from $z(it)$ for any $i$ [13]. The embedding dimension $m$ is chosen using the *false nearest neighbor* method, in which the embedding dimension is extended until the number of false neighbors drops to zero [14].

When the parameters have been chosen, the vectors $\mathbf{x_k}$ are created. The vectors are plotted as points in m-dimensional phase space, this is the recurrence plot. An example of how a periodic time series can be represented in phase space in a recurrence plot and how its recurrence network would look, is shown in Figure 2.1. The recurrence network can be represented using a matrix $R$. The construction of the matrix can be done using several methods, elaborated upon below.

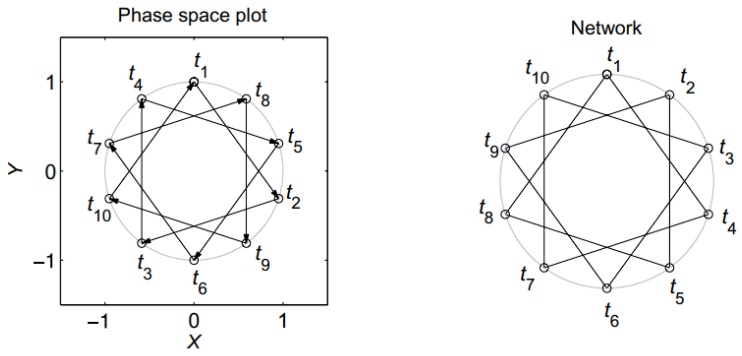



Figure 2.1: Example of a 2-D phase space representation, or recurrence plot, of time series (left) and its resulting Recurrence Network (right) that connects the points that are in close proximity in phase space [11]

### Recurrence networks

Each point in the recurrence plot is considered to be a node in the network. The resulting network will then carry information about the dynamical system that created the original time series. The requirements for two nodes to be connected can vary. Three common interpretations of recurrence plots are $\epsilon$, $k$ nearest neighbors, and adaptive nearest neighbors networks.

**$\epsilon$ recurrence networks** In this interpretation, $R$ is a binary matrix, with each element equal to [2]:

$$R_{i,j} = \Theta(\epsilon - ||\mathbf{x_i} - \mathbf{x_j}||) \tag{2.2}$$

Recall that the points in the recurrence plots are given by m-dimensional vectors. In essence, the element $R_{i,j}$ is equal to 1 if the distance between vector $i$ and vector $j$ is smaller than some $\epsilon$. The element is equal to 0, otherwise. This leads to a symmetric adjacency matrix and thus an undirected network.

**$k$ nearest neighbors recurrence networks** When using the $k$ nearest neighbors method, each node $i$ will be connected to the $k$ nodes that are closest to it, i.e. the first $k$ points in the recurrence plot when expanding from point $i$. This will lead to a regular graph, because all nodes will have the same degree by definition. The matrix will be asymmetrical in nature because if node $j$ is the nearest neighbor of node $i$, it is not necessarily so that node $i$ the nearest neighbor of node $j$ [15].

**Adaptive nearest neighbors recurrence networks** An adaptation to the $k$ nearest neighbors algorithm was proposed in order to obtain an undirected network [16]. In the adaptive nearest neighbors algorithm, a number ($E_0$) is chosen. Then an iterative process is started in which each node connects to its $E_0$ nearest neighbors. However, the links are bidirectional meaning that if $A(i,j) = 1$, then $A(j,i) = 1$. Furthermore, the link between nodes $i$ and $j$ only counts towards the maximum amount of allowed neighbors for one of the two nodes. This leads to an undirected network with $N \cdot E_0$ edges and nodes with a degree of at least $E_0$, but an average degree of $2 \cdot E_0$.

Recurrence networks capture trajectories over time within the time series and show when these trajectories re-occur. This makes them useful for determining characteristics of dynamical systems. Some properties of the dynamical system are captured in the topology of the network. For example, the local clustering coefficient can be used to identify dynamically invariant objects in the system [11].

## 2.2. VG variations

The VG is another way of transforming a time series to a graph, in which the mutual visibility between to data points determines whether connections in the graph are present. First presented in 2008 by Lacasa et al., the VG has been shown to inherit features from the time series it is generated from [1]. The definition for mutual visibility, which determines the existence of links in the graph, can be altered to achieve a variation on the algorithm that Lacasa et al. proposed. The method of Lacasa et al. will

henceforth be referred to as the Natural VG (NVG). In the following subsections, this algorithm and some variations to it will be presented briefly.

## Natural VG

The visibility graph proposed by Lacasa et al. generates a node for each entry of the time series $(x_i, t_i)$. Edges are placed between the nodes if the nodes can 'see' each other i.e. no peaks between the two time series entries obstruct the view. A pair of nodes corresponding to entries $(t_a, x_a)$ and $(t_b, x_b)$ are connected if

$$x_c < x_a + \frac{(x_b - x_a)(t_c - t_a)}{t_b - t_a} \tag{2.3}$$

holds for all entries $(t_c, x_c)$ with $(t_a < t_c < t_b)$. In other words, if all values $x_i$ are plotted against time $t$ and slopes are drawn between the peaks of all entries $(x_i, t_i)$, the entries $(x_a, t_a)$ and $(x_b, t_b)$ are connected in the graph if there is no peak that intersects or touches the slope.

## Horizontal VG

In the case of the horizontal visibility graph, lines of sight are only drawn horizontally. An advantage of this is the lower computational complexity. The criterion for two nodes to be connected in the graph is defined as

$$x_a, x_b > x_c, \tag{2.4}$$

in which $(x_c, t_c)$ are all entries in between entries $(x_a, t_a)$ and $(x_b, t_b)$ [5]. The lower complexity of this implementation allows for analytical solutions to be generated regarding average degree, degree distribution and clustering for HVGs of certain time series [17].

## Circular VG

Instead of drawing straight lines between peaks, one can choose to draw arcs to determine visibility between samples. This is the defining feature of Circular VGs. No line is drawn between the two samples, instead a circle is chosen that passes through both data points. The circle segment, or arc, that is between the two data points is used as the measure for visibility. If there are samples that intersect with the arc, visibility is obstructed. The circle system can be described by Eq. 2.5 [4].

$$f(t,x) = (t - t_a)(t - t_b) + (x - x_a)(x - x_b) + \alpha[(t - t_a)(x_b - x_a) - (x - x_a)(t_b - t_a)] \tag{2.5}$$

Where the subscripts $a$ and $b$ are indicative of samples $a$ and $b$ that are placed on the circle, and $\alpha$ determines the radius of the circle and the angle at which the arc is placed. For large $|\alpha|$, the radius of the circle grows. This means that for $|\alpha| \to \infty$ the line segment between the two samples will be a straight line. When $|\alpha| \to 0$ the line segment approaches the diameter of a circle. The values for $|\alpha|$ used by Xuan et al. are in the range [-100, 100] [4]. The sign of $\alpha$ determines whether the midpoint of the selected circle is above or below the line between the data points [4]. The circle systems for various $\alpha$ are visualized in Figure 2.2. In Figure 2.3, it is shown how these circles are used as lines of sight between samples. In this example there is a negative $alpha$, evident from the orientation of the arcs.

## Limited Penetrability VG

The concept of limited penetrability was first introduced in 2012 and was applied to the NVG [18]. When generating a VG with limited penetrability, the line of sight between two samples is no longer obstructed by a single sample that lies in between. Instead, a penetration distance is introduced which determines how many samples that are in conflict with the visibility criterion are allowed between two data points, while still having a link between the two corresponding nodes in the VG [18], [19]. This concept can be applied to the NVG, HVG and CVG presented above. Most commonly a penetration distance of $m = 1$ is used, meaning one sample is allowed to violate the visibility criterion [4], [18], [19]. Figure 2.3 shows limited penetration being applied to a CVG. The links that are formed as a result of the limited penetration are shown as dashed lines in the graph.

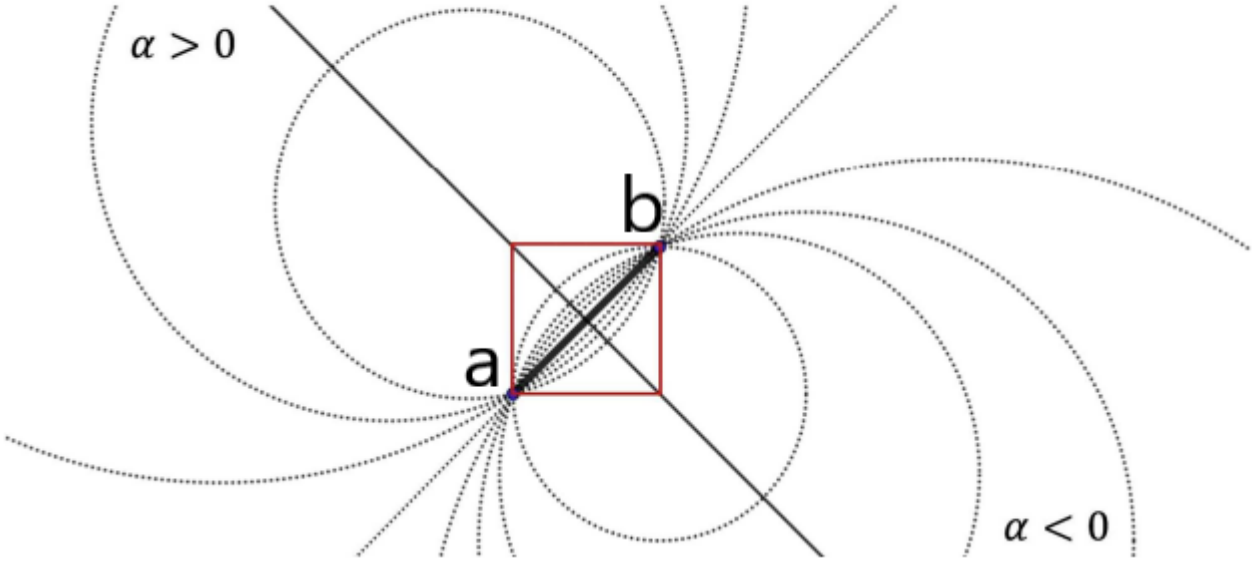



Figure 2.2: Depiction of circle system between two data points (at the lower left and upper right corner of the red square marked point $a$ and point $b$), for various values of $\alpha$ [4]

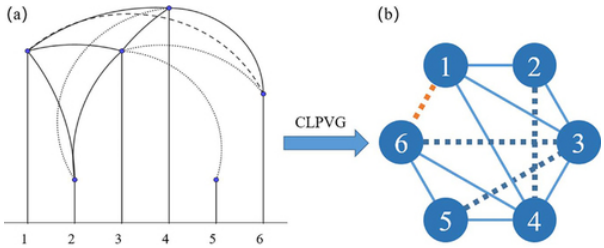



Figure 2.3: Example of how the concept of Limited Penetrability affects a CVG. In (a) the lines of sight are drawn on the time series. In (b) these lines of sight are represented by links in the graph. The blue dashed links are links that are added to graph as a result of the LP feature. The red dashed link is added as result of the lines of sight being drawn as arcs, as opposed to the straight lines of the NVG [4]

## Dual perspective VG

In 2021, Zheng et al. noticed a shortcoming of the NVG algorithm. Although large positive values are typically converted to hub nodes in the graph, the same is not true for large negative values [6]. This means that some extreme events in the time series can go unnoticed in the visibility graph. To solve this issue, the authors propose a dual perspective VG. In the dual perspective VG, the NVG of both the original time series ($s_t$) and the reflected time series ($-s_t$) are used to construct the graph. This allows extreme negative events in the time series to convert to nodes with extreme nodal properties [6]. These events are characterized by a large sudden drop in the time series. The values in the trough do not necessarily have to be negative, but very small compared to the other values in that region of the time series. Similar to limited penetrability, the concept of dual perspective can be applied to any VG algorithm although at the time of writing it has only been used by Zheng et al. [6] in combination with the NVG.

## Link weights

In the original algorithms for NVG, HVG, and CVG, the adjacency matrix of the resulting graph is binary [1], [4], [5]. Each link has a Boolean value $a_{ij} = 1$, regardless of which nodes $i$ and $j$ it connects. When no link is present, the entry in the adjacency matrix is a logical 0. There have been multiple works that attributed weights to the links in the VG, in different ways [20]–[23]. These methods include

weights based on the angle of the link and the (Euclidean) distance between the data points [6], [20]. The addition of the weight information means that more information from the time series is captured in the graph, which can lead to more robust results when deriving conclusions from the resulting VGs [20], [22].

### Directed VG

All VGs that have been introduced above are proposed as undirected graphs. This entails that a link between nodes works in both directions, i.e. if node $a$ is connected to node $b$ then the inverse is also true. However, directed versions of the algorithms have been proposed and used for specific use cases. To quantify the reversibility of a time series, the time directed HVG was proposed by Lacasa et al. [24]. In this version, any node $a$ can only connect to node $b$ when $t_a < t_b$. A directed version of the NVG has been used by Hu and Xiao [10] for improved prediction for time series based on random walks in the directed NVG.

## 2.3. Applications of the VG

The applications of VGs fall largely into two categories: analysis of fractal series and the classification of time series of which the underlying system is unknown. Works from both categories are described in this review.

Time series observed in many fields have a fractal nature. This entails that a small part of the time series is similarly shaped as a part of the time series on a larger scale [25]. Fractal time series can be modeled as a fractal Brownian motion (fBm). Models like these are used in various fields, ranging from geology, to biology and finance [26]–[28].

### Estimating the Hurst Exponent

Fractal Brownian motions are characterized by their Hurst exponent [29]. The Hurst exponent $(H)$ is indicative of whether the time series displays long memory, or long range dependence. It was first introduced by H.E. Hurst during his research of the reservoir control for the river Nile. The Hurst exponent is a measure of how the fluctuations between data points scale and how much these fluctuation deviate from a random-walk model. The value of $H$ lies between 0 and 1, where a value of 0.5 corresponds to a random walk, whereas values close to 0 and 1 correspond to anti-persistent and persistent time series, respectively. An anti-persistent process means it is more likely for a decreasing process to show an upward trend and vice versa. For the persistent process the opposite holds, and thus a decreasing process is likely to keep decreasing while an increasing process is likely to keep increasing [30].

The estimation of the Hurst exponent has attracted a lot of attention, as it indicates whether the future behavior of the time series is dependent on the past behavior. If this is the case, that means the possibility for making predictions on the future behavior exists [31]. Naturally, this has garnered interest in the financial markets as the prices fluctuations of stocks can be analyzed this way. However, there are other fields in which the Hurst exponent has become a point of interest [32]. There are multiple methods to estimate the Hurst exponent. A relatively new method was proposed by Lacasa et al. in the same paper where they introduce their VG algorithm [1]. This method rests upon the found relation between the degree distribution of the VG and the Hurst exponent of the time series. It is found that the slope of the distribution when plotted on a log-log scale, scales linearly with the Hurst exponent. Furthermore, this slope is indicative of the nature of the process that generated the time series and can be used to distinguish between stochastic and chaotic processes [33].

### Power of Scale-freeness of VG for heart rate dynamics

The visibility graph approach to determining the Hurst exponent of a time series has been used successfully in multiple fields. In 2016, Bhaduri and Ghosh compared it with the multifractal detrended fluctuation analysis method (MF-DFA) which had already been widely used. They opted for MF-DFA over DFA, as the latter requires longer sample sizes to be accurate [27].

The time series under investigation for their research is the instantaneous heart rate of subjects before and after they start meditating. Two data sets are used, one for Chi meditation and one for Kundalini yoga, with two time series each for a total of four. The metrics that are considered are the Power of

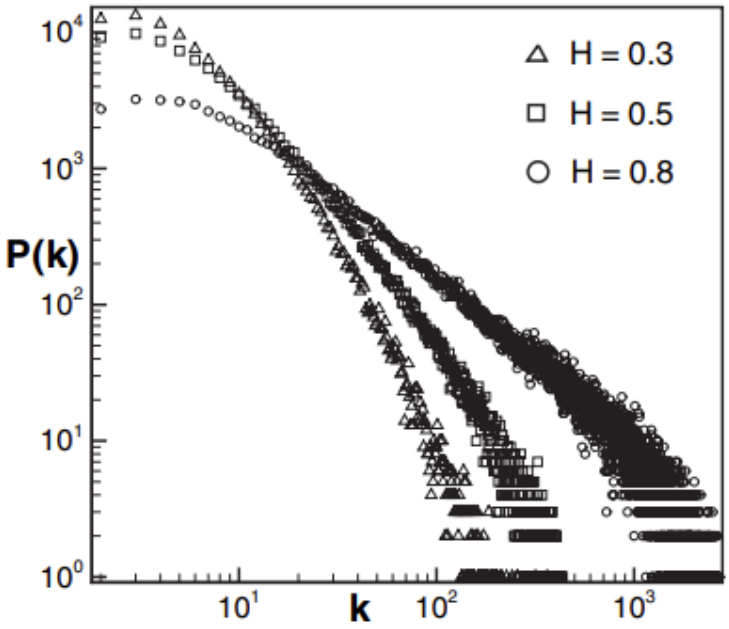



Figure 2.4: Degree distributions of three fractal Brownian motions with different Hurst exponents. Each visibility graph displays power law behavior in its degree distribution, but with varying exponents [29]

Scale-freeness of Visibility Graph (henceforth denoted as $\gamma$ in accordance with other literature) and the multi-fractal spectrum $f(\alpha)$. The two metrics cannot be directly compared as $\gamma$ measures fractality, while $f(\alpha)$ shows multi-fractality. Multi-fractal series are series that cannot be described with a single exponent, they require a spectrum of exponents. However, both methods do measure the complexity of the time series, which is enough for the purposes of differentiating between meditative and non-meditative states in the subjects.
The MF-DFA analysis yields results in which the difference between the spectra of the meditative and pre-meditative states are clearly noticeable with increases in width of 32.2-175%. However, the MF-DFA analysis does not reach conclusive results for the data of the subjects performing yoga. This is due to the insufficient length of the data. The value of $\gamma$ extracted from the VG, on the other hand, gives reliable results for both data sets. The value of $\gamma$ is consistently larger when a subject is either meditating or performing yoga when compared to the idle state. For the Chi meditation data set, a positive correlation between the two measured metrics is found. This means the method of using the visibility graph is a suitable way of determining the fractality of a time series, with the added benefit of being able to do so for short time series [27].
A similar research was conducted by Jiang et al. who also came to the conclusion that the visibility graph method can yield reliable results from a time series as short as 400 samples [34].

## Improved Power of Scale-freeness VG for diagnosis of Autism Spectrum Disorder

Ahmadlou et al. [35] propose an addition to the method of calculating the Power of Scale-freeness by analyzing the time series at different scales. The original time series is divided into sequences of different lengths, spanning different time scales. The sequences are constructed as follows:

$$\mathbf{x}_m^k = [x_m, x_{m+k}, x_{m+2k}, \ldots, x_{m+\lfloor \frac{N-m}{k} \rfloor k}] \tag{2.6}$$

The sequence is constructed using samples in the data that are a distance $k$ apart, with an initial offset $m$ and $m = (1, 2, \ldots, k)$. This means for the initial value of $k = 1$, only one sequence is constructed, for $k = 2$, two sequences are constructed that each contain half the data points, and so on. For each individual sequence, the NVG is constructed and the exponent of the degree distribution of this graph,

$\gamma_m(k)$, is calculated. The values of $\gamma$ for each scale $k$ are averaged.

$$\gamma(k) = \frac{1}{k}\sum_{m=1}^{k}\gamma_m(k) \tag{2.7}$$

The value of $k$ is increased with increments of 1, until the resulting $\gamma(k)$ value falls within some predetermined distance of the previous value.
This method is applied to a generated time series with a known fractality. It is shown to be more sensitive to fractality of the time series than the method that simply calculates $\gamma$ using the whole time series, especially when noise is introduced on the signal. After this successful benchmark, the improved method has been applied to EEG scans for the diagnosis of autism. The improved method shows better results in discriminating between autistic and non-autistic test subjects [35].

### Seismic activity

The visibility graph has also been used to analyze data of seismic activity. It has already been shown that there is structure in the occurrences of earthquakes using, among other techniques, DFA analysis [26]. The same authors later investigate whether the visibility graph can give more insight into seismic sequences.
The research uses data of seismic activity in Italy from 2005 to 2010, with records of all activity of magnitude 1.9 and up on the Richter scale. A natural visibility graph is created using these data. The degree distribution of the graph is plotted on a log-log scale. The graph is found to be scale-free, since the degree distribution obeys a power-law $P(k) \sim k^{-\gamma}$. The exponent $\gamma$ is estimated using the Maximum Likelihood Estimate (MLE) and found to be almost exactly 3. Using the previously established relations, the authors conclude that there is long range dependence between the data points in the data sets [29]. After establishing the dependence, the data are transformed in several ways and the process is repeated. It is shown that the power-law behavior in the degree distribution persists even if the data points are shuffled and when a threshold value is introduced that discards a part of the data set [9].

### Stock market analysis

The interest in finding long range correlations is ever present in the world of finance. The visibility graph has been used to analyze behavior of time series in the stock market. In 2010, a research was conducted that constructed the visibility graph of 30 stock indices. In this case, weights are assigned to the links of the graph that correspond to the difference of price height over the time between the two data points. Using the resulting weighted graphs, three spanning trees are constructed for each graph. The minimum, maximum, and random spanning trees are constructed. It is investigated whether these trees exhibit allometric scaling behavior. The scaling behavior in the minimum spanning tree is found to be similar to the scaling behavior in the trees resulting from fractal Brownian motions. Furthermore it was found that the scaling factors are dependent on the length of the series and on the Hurst exponent of the Brownian motion [36].
In 2015, Stephen et al. [28] analyzed the behavior of stock indices using visibility graphs. Their work focuses on the evolutionary behavior of the daily returns of stocks over time. The time series under investigation are divided into short segments. The visibility graph for each of these segments is constructed and the collection of these graphs are represented in a new network. Each unique visibility graph is seen as a state, which is represented by a node in the new network. Two nodes $i$ and $j$ are connected by a uni-directional link if state $i$ is followed by state $j$ in the time series. This results in a state transfer network for the whole time series. In this network, the links that occur more often have a higher weight, and the degree of a node is representative of the occurring frequency of the state it represents. It is then investigated whether the transfer properties are dependent on the order of the data by repeating the same process for shuffled versions of the time series in which the samples are randomly shuffled although the authors fail to mention how the samples are shuffled. The ratio of occurring frequencies between the original time series and the shuffled time series are determined. The states that occur significantly more often in the original data are as seen the leading motifs for that series. The times that these states occur are taken as a new time series. It is investigated whether a pattern in these occurrences can be identified.
The described method is first applied to generated fBm series with different Hurst exponents. It is found

that the occurrence frequencies are significant against the frequencies in the shuffled series, and multiple motifs can be selected. Through R/S analysis it is shown that the intervals between the occurrences of the leading motifs have scaling exponents that are positively correlated to the Hurst exponent of the original time series. Similar results are found for the stock market data from NASDAQ and S&P500. It is found that the NASDAQ prices exhibit more fractality.
The contributions of Stephen et al. are twofold; the state transfer network yields insight into the evolutionary behavior of the daily stock returns over time, while the R/S analysis of the occurrence times between the leading motifs gives an indication of the fractality of the series. For a definitive conclusion on the latter subject, more data is needed [28].

### Classification of modulation schemes

A method for differentiating between different modulation schemes, such as Quardature Amplitude Modulation (QAM) and Binary Phase Shift Keying (BPSK), using the concept of the visibility graph was introduced by Xuan et al. in 2022 [4]. In the first part of their work, the CLPVG is introduced. This is the combination of a Circular VG with Limited Penetration as described in Section 2.2. It is shown that this implementation of the visibility graph is able to more accurately extract information of noisy signals in time domain when compared to the Limited Penetration VG (LPVG). For the first test, chaotic signals with and without additive white Gaussian noise are generated. The CLPVG and LPVG algorithms are used to obtain a degree distribution of the noisy and clean signals. It becomes evident that the degree distribution of the CLPVGs with noise are closer to the degree distribution of the original signal. The same can be said for other graph features such as average degree, average path length, and average clustering coefficient. It is concluded that with the right choice of parameters, the CLPVG is less sensitive to noise for accurate feature extraction compared to the LPVG [4].
In the second part of their research, Xuan et al. make use of sub-graph networks (SGN) for better feature extraction. The SGN is constructed by defining the sub-graphs in the network. This can be done in multiple ways, corresponding to different order SGNs. A first order SGN is created by taking all the links in the original network as nodes for the SGN. These nodes are then connected in the SGN if the links share a terminal node in the original network. This is also known as a line graph. The resulting network inherits structural features of the original network and can be used to augment the original network which can improve classification accuracy [37]. The work of Xuan et al on the CLPVG restricts itself to first order SGNs.
This method is then put to the test with a use case scenario that is inherently noisy: radio wave communication. A generated data set containing 11 different modulation types with varying noise levels is used. After some data pre-processing, the CLPVGs of the signals are constructed and subsequently, the first order SGN of the CLPVGs are constructed. Machine learning algorithms are used to interpret the graph structures as data and eventually label the signals with a certain modulation scheme. The whole process is visualized in Figure 2.5. It is found that the combination of SGN and CLPVG in the method for classifying signals, leads to the highest accuracy of determining the modulation type that was used. This holds especially for cases with high SNR. The method could be further optimized by tuning the parameters for the curvature of the arcs and the amount of allowed penetration [4].

### Rolling bearing fault diagnosis

The concept of the visibility graph can be used to monitor the status of an object under surveillance. Gao and Yu [8] applied this to fault diagnosis in rolling bearings in 2019. The vibrations in the machinery parts are recorded over time, forming a signal. The vibration signals can have non-linear and non-stationary properties, which limits the effectiveness of classical signal processing measures [8]. Using the algorithm for the Horizontal Visibility Graph (HVG), a visibility graph is constructed for the signal. This graph is then used to diagnose faults in the rolling bearings. Instead of analyzing typical graph metrics such as average degree, clustering coefficient, and others, Gao and Yu make use of graph signal processing. In this field of research, traditional signal processing methods are extended to be used on graphs [38]. In particular, Gao and Yu use the total variation on graph ($TV_G$), which measures the smoothness of graph signals. The graph signal, in this case, is the measured vibrations at the time steps that correspond to the nodes in the HVG. During normal operation of the rolling bearings, these signals should be smooth, i.e. the differences in measured vibration between connected nodes are relatively small. The vector containing all the measured vibration values and the adjacency matrix of the HVG are used to calculate $TV_G$. The outcome is then compared to previously obtained training

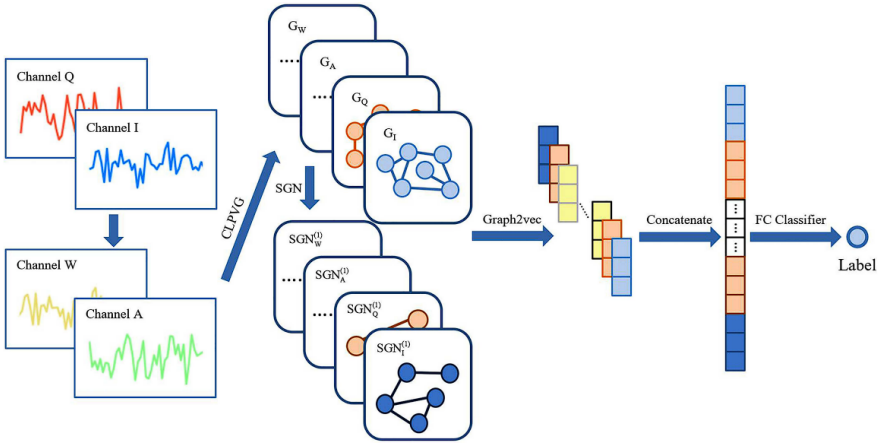



Figure 2.5: Flowchart demonstrating the classification process of signals based on CLPVG graph transformation and machine learning algorithms [4]

data where $TV_G$ values are assigned to known conditions of the rolling bearing. The classification of the measured state is done using the Mahalanobis distance, which is a measure that compares the difference between two vectors as compared to variation within the vectors themselves [39]. The measured state is classified as the known state that is closest according to Mahalanobis distance measure. The known states have been previously established using training data, the measured state is compared to these established states of operation of the rolling bearing.
This method is verified using data sets of two universities. It is found that the method can accurately distinguish the different states of the rolling bearing under supervision. The method is compared to classical time domain methods approximate entropy (ApEn), permutation entropy (PE), kurtosis (KU), and root mean square (RMS). From the experimental results, it is evident that the total variation on the HVG of the vibration signal outperforms those methods. The authors conclude by proposing investigation into monitoring systems over time with varying conditions [8],

## Classification of medical test results

In order to better understand temporal behavior in biological systems, Zheng et al. developed a method for community detection in networks that represent a time series [40]. A community in a network is a set of nodes that are more interconnected with themselves than with the rest of the network. Communities in a network that represents a time series, such as the visibility graph, are representative of a certain shared temporal behavior. A community in the network can thus be representative of the state of the biological system.
In order to prove this, Zheng et al. used both simulated time series and various biological data sets. The time series are converted to complex networks using their proposed Weighted Dual Perspective Visibility Graph (WDPVG). The advantages of this implementation are that the effects of uneven sampling are diminished, and that events related to troughs in the signal can also be detected. To find communities in the resulting graphs, the authors propose a community detection algorithm based on shortest paths in the network. In this algorithm, the shortest path $\{v_i^s\}$ of some length $k$ between the start node (corresponding to the first data point) and the end node (corresponding to the last data point) is found. In the next steps, all other nodes in the network are divided into hubs. Each hub corresponds to a node in path $\{v_i^s\}$. For each node in the network, it is determined which node in path $\{v_i^s\}$ is closest. The node is then added to the hub corresponding with the closest node in path $\{v_i^s\}$. If two nodes in path $\{v_i^s\}$ are equally close, the node that corresponds with the earliest data point is selected. Finally, hubs can be merged if the distance between the corresponding nodes in path $\{v_i^s\}$ is below a threshold value $\epsilon$. This results in all nodes in the graph being allocated to some community and the number of communities ranges from 1 to $k$, depending on the threshold $\epsilon$.
The method is first applied to the simulated time series, which are a sine wave, a block wave, and a saw tooth. All three are types of signals are generated with added noise and missing data points to test the robustness of the method. The method proposed by the authors proves to be better at identifying

the periods of the generated signals when compared to the Louvain and Girvan-Newman methods for community detection, especially when noise and missing samples are introduced. The same methods are then applied to the biological data sets. Here it is also found that the method proposed by Zheng et al. [40] can more accurately represent the states of the biological systems when compared to the Louvian and Girvan-Newman methods. The proposed method seems suitable for identifying different states within a time series, though more research is recommended. It should be explored how missing peak data would affect the method, especially in the case that the missing data is much larger than other values. The authors suggest a sliding time window provides smoothing. Furthermore, the process of hub merging can be further investigated [40].

## Synchronization of chaotic signals

VG-based methods for determining a measure of synchronization of chaotic systems was proposed and found to be successful under certain circumstances [41], [42]. The VG-based methods presented in [41] work as follows: the output of two systems, vectors $y1$ and $y2$, are used to construct two visibility graphs. The degree sequences (DS) of the visibility graphs are stored as one dimensional vectors. The similarity of the degree sequences is used as a measure for synchronization between the systems. The similarity is determined as follows:

$$S = |\frac{cov[DS_{y1}, DS_{y2}]}{\sigma_{DS_{y1}} \cdot \sigma_{DS_{y2}}}| \tag{2.8}$$

This results in a value between 0 and 1, where 0 corresponds to no synchronization and 1 corresponds to total synchronization. The visibility graphs are constructed as NVGs, HVGs as well as NVGs and HVGs of time series that were embedded in higher dimensions similar to the embedding described in Section 2.1. These methods were compared to the existing methods for detecting synchronization: cross-correlation, coherence, imaginary part of coherence, synchronization likelihood, phase coherence and phase lag index. Multiple noise conditions were used.
The test was performed for uncoupled non-identical systems, coupled non-identical systems, uncoupled identical systems, and coupled identical systems. Accurate measurements of the synchronization should show no synchronization for uncoupled identical systems and the level of synchronization should increase for increasing coupling factor (C) until full synchronization is achieved. For non-identical systems, the synchronization still increases from near zero for small C, but should never reach full synchronization for any C. The HVG method with no embedding was found to be the most accurate of the VG methods, outperforming all other methods except synchronization likelihood when determining the synchronization between two coupled Rössler systems in a noise free environment. When noise is introduced, however, the VG-based methods have a less accurate performance. They are no longer able to detect full synchronization between coupled identical systems. Ahmadi et al. [41] state that the contamination of the system with noise results in a degree distribution skewed towards higher degrees. This hinders the ability to detect synchronization between the systems, and therefore suggests that VG-based methods are not likely to be useful for detecting synchronization in a noisy environment [41].

## Quantify statistical time reversibility

Some processes are statistically time reversible. This means that for a process $X(t)$ an output $[X(1), X(2), \ldots, X(N-1), X(N)]$ is equally probable as the reverse output $[X(N), X(N-1), \ldots, X(2), X(1)]$. If a process is time irreversible, this indicates non-stationarity and the presence of chaotic or non-Gaussian dynamics. A thermodynamic process that falls out of equilibrium may start to exhibit such dynamics. Using a time directed version of the HVG, the time irreversibility of a time series can be quantified [24]. The method is proposed as folows. The time directed HVG (dHVG) is constructed from the time series. This means that the lines of sight only extend forwards in time from each sample in the series. Two degree distributions are formed as a result: the in-degree and out-degree. The in-degree of a node is the number of links that lead into that node, the out-degree is the number of links that extend from that node. The in-degree distribution of the time series $x(t)$ is by definition equal to the out-degree distribution of reversed time series $x'(t)$, and vice versa. The difference between the out-degree distributions $(P_{out}(k))$ of the time series x(t) and its reverse, is a measure of the irreversibility of the process. This difference is quantified using the Kullback-Leiber divergence (KLD).

$$KLD[P_{out}(k)||P_{in}(k)] = \sum_k P_{out}(k) log(\frac{P_{out}(k)}{P_{in}(k)}) \tag{2.9}$$

The distribution $P_{in}(k)$ of the original series is used instead of $P_{out}(k)$ of the reversed series, because the distributions are equal and this prevents the need for constructing the dHVG twice. The KLD is equal to 0 for identical distributions, indicating perfect reversibility. The larger the value for KLD, the higher the irreversibility of the process.
To verify the method, a process is chosen of which the reversibility can be tuned via parameter $V$. It is shown that the method is accurate at detecting the irreversibility as $V$ is increased beyond $V = 0$. The quantification of the irreversibility is improved when the distributions in Eq. (2.9) are replaced with the higher order distribution $P_{in}(k, k')$, which indicates the probability of a degree $k$ given that the previous node has degree $k'$. With this method, the time reversibility of a process can be accurately assessed without the need for coarse graining or making assumptions as is necessary for other methods. This can be used to monitor processes and recognize when they fall from their equilibrium state [24].

## Time series forecasting

A use for the time directed NVG is described by Hu and Xiao [10], who use a variation of the dNVG to forecast future entries of a time series. The dNVG of the time series is constructed, following the standard NVG visibility criterion but only allowing links that move forward in time. Then it departs from the dNVG method proposed by Lacasa et al. [24], as the nodes that are not connected by these forward or, as the authors call them, 'positive' links, are connected by 'negative' links that go backwards in time. The negative link set $E = L_n$ is thus by definition the complement of the positive link set $L_p$. An analysis of the graph based on random walks is conducted as follows. For the resulting graph $G(N, L_p \cup L_n)$, two transition probability matrices are generated. The matrix $P_p$ indicates the probability of a random walker to move from one node to another along a positive edge. The matrix $P_n$ indicates the probabilities for a random walker to move to another node using a negative edge. Using these matrices, an expression can be found for the likelihood of a random walker ending up at node $y$ when starting from node $x$ after one step. The expressions for each of these nodes are used to compare the similarity of each node in the network.
Take a series $y(t)$ of length $N$. A new time series is constructed $Y(T)$ to reflect the fluctuations between the samples in $y(t)$ so that for any sample entry $m$ in the new series the following holds:

$$Y_m = (y_{m+1} - y_m)/y_m \tag{2.10}$$

The new time series $Y(T)$ of length $N - 1$ is converted into a graph $G(n, L_p \cup L_n)$ using the aforementioned dNVG method, where $n$ is the set of nodes and $L_p \cup L_n$ is the set of links. The transition matrices are generated, where each element of the matrix is either 0, or $1/k$ with $k$ being the number of possible transitions from that node. The probabilities for a random walker to move from a node $x$ to a node within $n$ are given by a vector $\pi_x(t)$. The probabilities are found by

$$\pi_x(t) = P^T \pi_x(t-1) \tag{2.11}$$

where $\pi_x(0)$, the initial condition, is a vector of length $N - 1$ with all zero entries except a one at the $x^{th}$ entry. The probability for a random walker to move from node $x$ to node $y$ in $t$ steps and the probability for a random walker to move from node $y$ to node $x$ in the same amount of steps are combined to define a local similarity measure between the two nodes. The results for different values of $t$ are superimposed to arrive at the final similarity $S_{xy}^{SRW}$ where SRW stands for superimposed random walker.
To forecast the value of the next entry in the time series $y(N + 1)$, the similarity scores $S_i$ between $n_N - 1$ and all previous nodes $n_i$ are collected. The authors pose that nodes with a high similarity score are more likely to have a similar future [10]. Recall that the entries in $Y(T)$ reflect fluctuations in the time series $y(t)$. An estimation for $y_{N+1}$ can now be made by taking a weighted average of all previous entries in $Y(T)$, where each entry is weighted with the corresponding similarity score. Let the resulting expected fluctuation be $E_f$. Then the estimate for the next entry is defined as $y'_{N+1} = E_f y_N$.
This method has shown accurate results in experiments with real-world data, outperforming other forecasting methods by a small margin [10].

## 2.4. Summary

As demonstrated by the discussed papers in this section, wide variety of fields can make use of VGs in different ways. In the literature that uses the concept of the VG, a clear structure can be identified. A data set of some time series is taken as the starting point. From there, a type of VG is constructed. However, from the VG itself and its metrics, hardly any conclusions are drawn. There is another layer of analysis necessary in order to produce results from which conclusions on the time series can be asserted. The different materials and methods for all of these steps are shown in Figure 2.6.
This thesis will focus on the second column of Figure 2.6, i.e. the construction of the visibility graph. Although it is only one step in the process, it is the step that most fundamentally changes the way the information in the time series is presented and structured and there are still unanswered questions as to how this restructuring impacts the information [17]. In Chapter 3, this thesis will delve into the way that different VGs are constructed and attempt to give a mathematical interpretation that can be attributed to the VG.

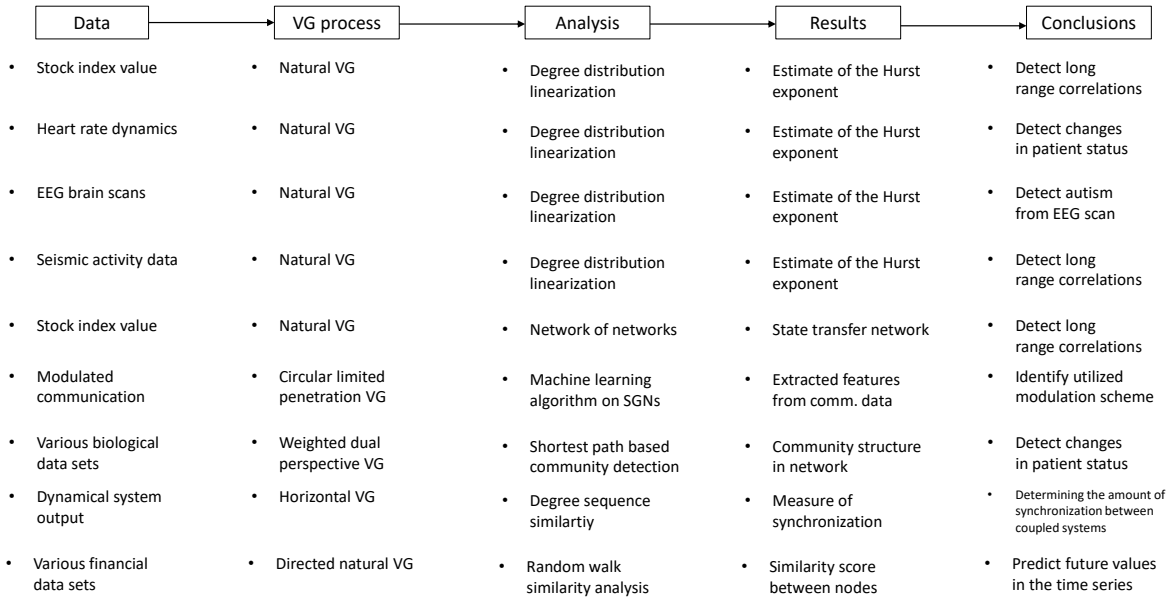



Figure 2.6: Overview of the structure found in literature that uses the concept of the VG

# 3

# VG Methodology and Metrics

Although many methods for constructing visibility graphs have been proposed, the understanding of what information the graph holds is still lacking. Past works have mostly focused on the practical applications of the visibility graph, for example to characterize states in a process. Despite the successes that have been achieved with these methods, none of the works as of the time of writing fully explain the relation between the time series and the visibility graph [17]. The transformation from a time series to a visibility graph is a ”lossy” one. Not all information from the time series data is captured in the graph, especially if the graph uses a binary adjacency matrix [17]. This prevents a possible inverse operation from correctly mapping the graph back to a series of values. Since this first publication, variations of the VG have been proposed, each with different criteria for a connection between two nodes. These result in different graphs which have characteristics that make them more suitable than other for certain scenarios. This chapter will introduce the original VG and its later variations briefly. The original VG and other variations are constructed from time series using MATLAB scripts. Although each variation has its own script, all follow the same steps described in Algorithm 1.

The variable $i$ denotes the sample that is evaluated, and for all values $a$ before and after that sample,

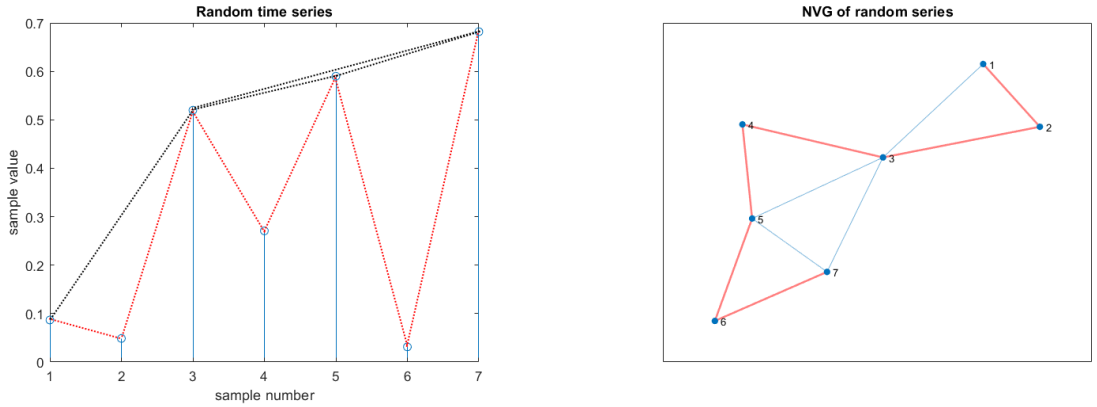
**Algorithm 1** General VG algorithm

```
for i = 1 : N do
    for a = 1 : i − 1 do
        if Neighboring node then
            Add link
        else
            for c = a + 1 : i − 1 do
                Count obstructing samples
            end for
            if Obstructions == 0 then
                Add link
            end if
        end if
    end for
    Repeat loop for a = i+1:N
end for
```

the visibility criterion is checked. If this criterion is not violated, i.e. Obstructions == 0, a link is added between node number $a$ and node number $i$. Not shown in the algorithm are a few tweaks that make the script run more efficiently by breaking the loop at certain points and adding the links in a sparse matrix manner to preserve memory. As discussed in Section 2.2, the different variations of VGs have their own visibility criteria or additional steps in their construction. This chapter shows an implementation of the different variations and investigates graph metrics for various types of times series and their VGs, in order to gain more insight into the VG algorithm.

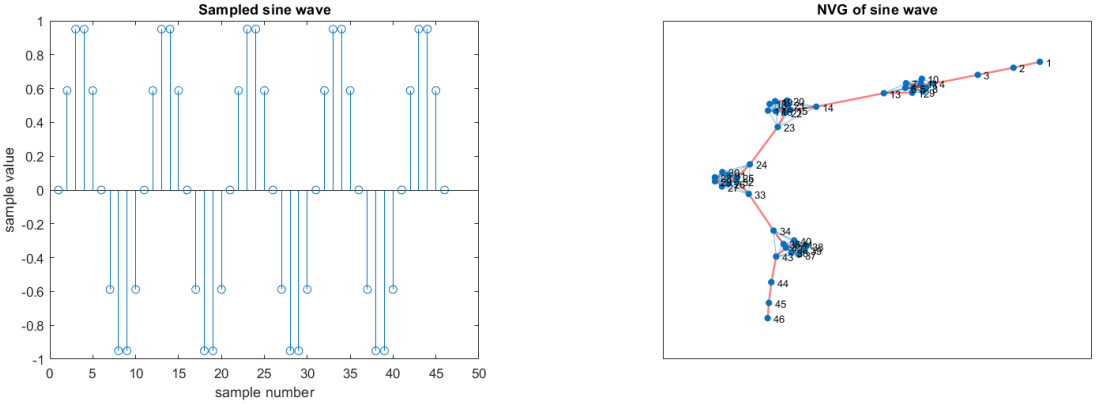
## 3.1. Natural visibility graph

This is the first implementation of the VG concept, proposed by Lacasa et al. in 2008 [1]. The visibility criterion is defined in Eq. 2.3. A visualization of how a time series is transformed is shown in Figure 3.1. The randomly generated time series of length $N = 10$ is plotted, and the lines of sight are drawn in gray between the peaks of each data point. The resulting VG is plotted next to it. When examining carefully, one can see the structure of the lines of sight between the peaks, is exactly the same as the structure of the links between the nodes. This is how the NVG is constructed.




(a) Random time series (b) NVG

Figure 3.1: In subfigure (a), a random sequence of 10 numbers on the interval (0,1) is plotted. All lines of sight between the data points are drawn. Subfigure (b) shows the resulting NVG for the sequence

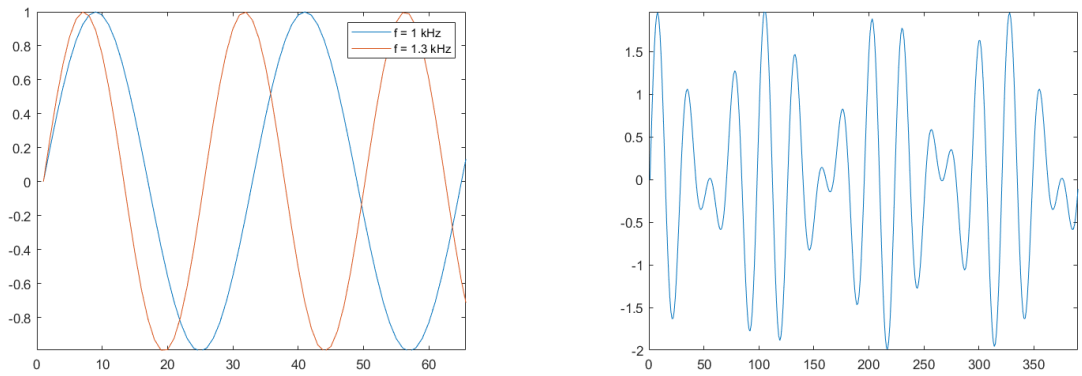



(a) Sampled sine wave (b) NVG

Figure 3.2: In subfigure (a), a sampled version of a sine wave is plotted. The lines of sight between the data points are ommitted for readability. Subfigure (b) shows the resulting NVG for the sampled sine wave

In Figure 3.2, the regular nature of the NVG of a periodic series becomes apparent. The phenomenon of VGs with a limited amount of repeating structures arising from periodic sequences has been described before [1]. When the periodic sequence is a sampled sine function, the relation between the sampling frequency $f_s$ and the frequency of the sine function $f$ is of importance for the resulting NVG. If the sampling frequency is chosen such that $f_s/f$ is not a rational number, this will cause the signal to not be sampled at the exact same point in each period of the sine function. This results in higher and lower peaks in the time series, and distorts the neat separation of periods. This effect can be seen in the Figure 3.4. The addition of two periodic sequences $t_1$ and $t_2$ results in another periodic sequence $t_{sum}$. The period of $t_{sum}$ is the lowest common multiple of the periods of $t_1$ and $t_2$ as shown in Figure 3.3. It is therefore expected that the NVG of such a sequence $t_{sum}$ exhibits the

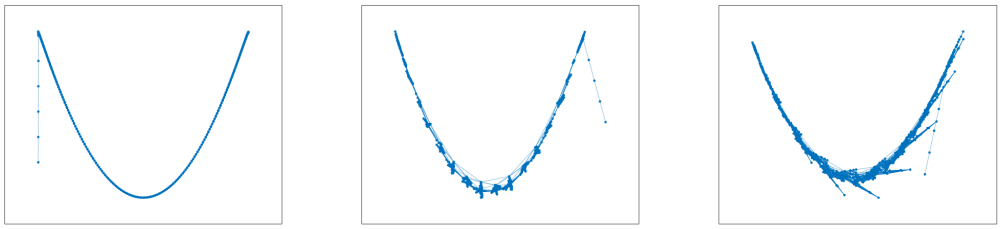
repetitive behavior as shown in Figure 3.2, but at a larger scale. This is confirmed in Figure 3.4 where the pattern in subfigure (c) is only repeated a few times.




(a) Two sine functions with frequencies 1 kHz and 1.3 kHz

(b) Sum of the two sine fuctions

Figure 3.3: In subfigure (a), two periodic sequences are plotted.Subfigure (b) shows the resulting sum of the two sequences



(a) NVG for f = 1 kHz

(b) NVG for f = 1.3 kHz

(c) NVG of the sum of the waves

Figure 3.4: The impact of the changing ratio $f_s/f$ is evident when comparing (a) and (b). In the sequence of 1 kHz, sampled at 32 kHz, the peak is always sampled exactly at the apex. This means there can be no lines of sight into another period. This changes when the frequency of the sine is 1.3 kHz and is still sampled at 32 kHz. This causes some of the clusters that correspond to the troughs of the sine are connected to troughs of another period. In subfigure (c) a repeated pattern can be seen but clearly at a larger scale than the other two NVGs. This is due to the much longer period that the summed sequence has

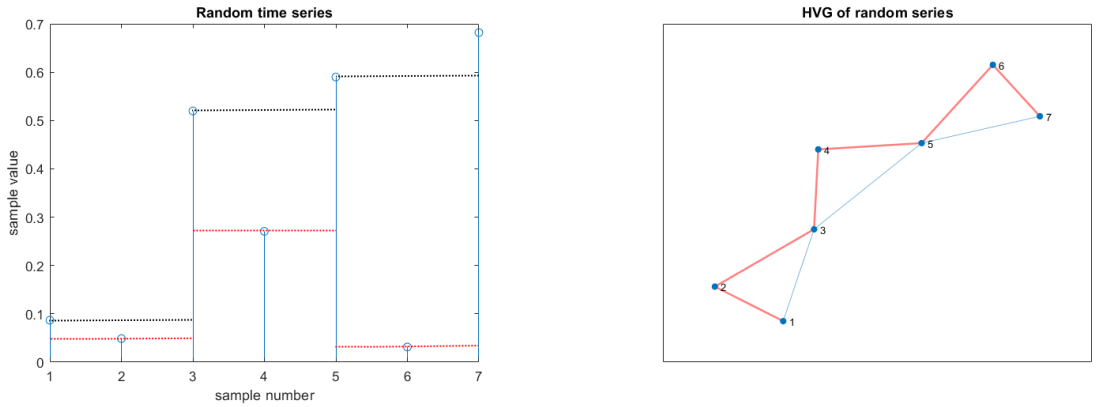
There are certain characteristics that are shared by all VGs as a result of how the algorithm constructs the graph from the time series. As the NVG was the first algorithm to be proposed, this was the first type of VG in which these features were discovered. Three characteristics that are often mentioned in literature are that VGs are [1], [17]:

- Connected
- Undirected (in all cases except the dVG)
- Invariant under affine transformations

The latter is not a graph characteristic, rather a characteristic of the function of the algorithm. In later work, Iacovacci and Lacasa add that by construction, a VG always contains a Hamiltonian path [43]. This is a path that visits all nodes in the graph exactly once. A VG always contains such a path, because neighboring data points are always connected in the graph. Therefore the path $p = 1, 2, ..., N$ is a Hamiltonian path for all VGs. These paths are indicated by red lines in the graph and the plots of the time series. Exact characterizations of NVG graph metrics are as of yet not described in literature, contrary to horizontal visibility graph's (HVG) metrics. This is due to the NVG's relative complexity [17].

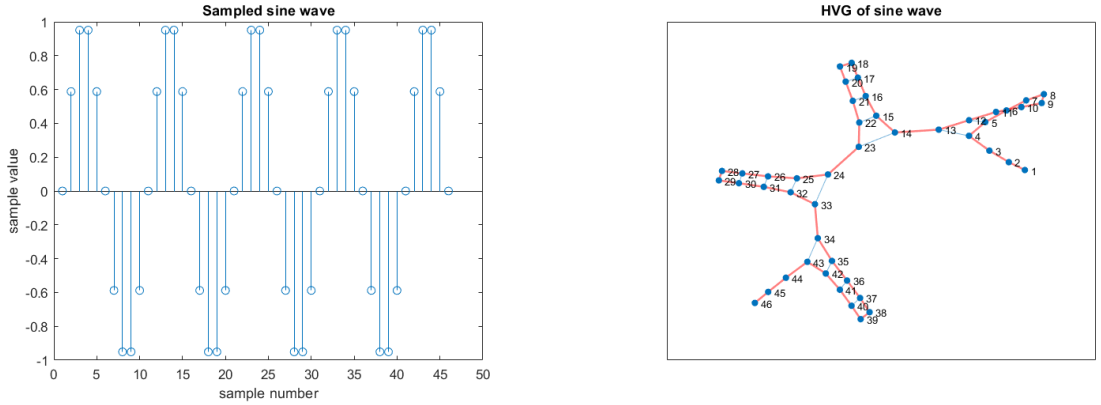
## 3.2. Horizontal visibility graph

For the HVG, lines of sight are only drawn horizontally. An advantage of this is the lower computational complexity. The criterion for two nodes to be connected in the graph is defined in Eq. 2.4. The MATLAB script is changed to reflect this altered visibility condition, but otherwise remains exactly the same as the implementation used for the NVG. An example of a horizontal visibility graph is shown in Figure 3.5. The same sequence of 7 random values between 0 and 1 is used as in Figure 3.1, and the lines of sight are drawn in black. In Figure 3.6, a sampled sine wave is shown along with its HVG. The regularity that forms in VGs of periodic sequences is very clear in this image.




(a) Random time series (b) HVG

Figure 3.5: In subfigure (a), a random sequence of 10 numbers on the interval (0,1) is plotted. All lines of sight between the data points are drawn. Subfigure (b) shows the resulting HVG for the sequence




(a) Sampled sine wave (b) HVG

Figure 3.6: In subfigure (a), a sampled version of a sine wave is plotted. The lines of sight between the data points are ommitted for readability. Subfigure (b) shows the resulting HVG for the sampled sine wave

The horizontal visibility graph can be used to identify randomness in a time series and to discriminate between chaotic series and independent and identically distributed theory. This is done using, among other metrics, the degree distribution, clustering coefficients, and average path length [5].

### Characteristics of HVGs for specific time series types

Shared characteristics of VGs are found in VGs that are constructed from similar time series. For example, the exponential behavior of the degree distribution in VGs constructed from fractal series has been well-documented [1]. The HVG, due to its lower complexity, lends itself well to find analytical results regarding the relation between time metrics and graph metrics, and between different graph

metrics [5]. It is possible to determine the mean degree of the HVG from the period T of the time series. The following theorem is found by Núñez et al. [44]:

$$\bar{k}(T) = 4(1 - \frac{1}{2T}) \tag{3.1}$$

This means the mean degree of a HVG is always within the bounds $2 \leq \bar{k} \leq 4$. The mean degree is minimal for a constant series, equivalent to period $T = 1$, and maximal for an aperiodic series which is equivalent to an infinite period length.
For a random time series it is found that the degree distribution of a HVG is always equal to Eq. (3.2)[5].

$$P(k) = \frac{1}{3}\left(\frac{2}{3}\right)^{k-2} \tag{3.2}$$

The degree distribution of the HVG of a random time series is thus not dependent on the probability distribution function that governs the height of the data points in the time series [5]. The proof arises from taking a generic data point and investigating the likelihood of that 'seed datum' (S) having a certain amount of neighbors. The neighbors are divided into two categories, 'bounding data' (B) and 'inner data' (I). The former are data points at either side of the seed that are higher than the seed datum. The latter are data points that are lower than the seed datum and lie in between the seed datum and the bounding datum on either side. The likelihood of a certain node having $k$ neighbors can then be described [5].

$$P(k) = \sum_{h=0}^{k-2} [S][B]^2[I]_h[I]_{k-2-h} \tag{3.3}$$

In this expression, all square brackets indicate the probability of that event occurring. The probability of each of these data points occurring is simply described by taking the integral of the probability distribution function $f(x)$ and setting appropriate boundaries [5]. Depending on the number of neighbors $k$, a multitude of configurations is possible. The inner nodes could be all on the left side, all on the right side, or divided over both sides of the seed node. The summation in Eq.(3.3) sums the likelihood of each of these configurations occurring, in which $h$ represents the number of inner data found on the left side of the seed node.
This illustrates why proofs of this nature are possible for HVGs, but similar relations have not been found for the NVG algorithm. Finding a bounding data point for a seed datum in an NVG is not as straightforward as finding the first data point that is higher than the seed.
The findings are consistent with the result found in Eq. (3.1), because $E[P(k)] = \sum_{k=2}^{\infty} k \cdot P(k) = 4$. As $P(k)$ is the degree distribution for random time series, this finding is in agreement with earlier statement regarding mean degree for aperiodic series. In later research, Lacasa investigated the effects on the degree distribution of an HVG when using an integer time series instead of a continuous series. A series of length $N = 10^5$ would be generated where each sample would be chosen from a set of integers $x \in [1, .., n]$. It was found that for sufficiently large $n$, the results converge to the findings of the continuous case in Eq. (3.2) [45].
The local clustering coefficient of the HVG of a random time series can also be neatly defined as a function of the local degree. The local clustering coefficient $C_i$ of node $i$ is a measure of the connectivity between the neighbors of node $i$. The relation between degree and clustering coefficient of a node is described in Eq. (3.4). Combined with the degree distribution for a random time series, this leads to the local clustering coefficient distribution given in Eq. (3.5).

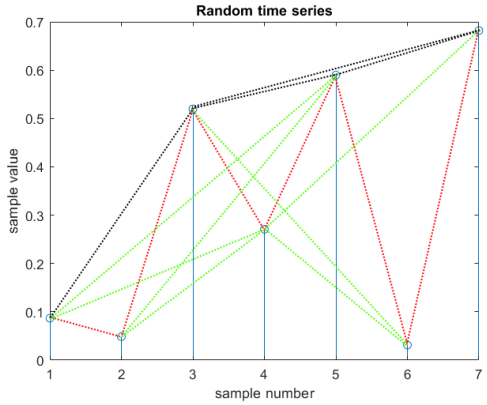
$$C(k) = \frac{k-1}{\binom{k}{2}} = \frac{2}{k} \tag{3.4}$$

$$P(C) = \frac{1}{3}\left(\frac{2}{3}\right)^{2/C-2} \tag{3.5}$$

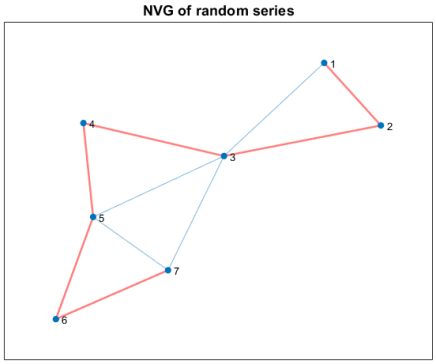
The above relations hold for an infinite random series and are again not dependent on the probability distribution that governs the height of the data points of the time series. When tested with finite random time series, the relations are confirmed [5]. The findings are used to discriminate between HVGs that are constructed from different types of time series, but not to discriminate between graphs that are a product of the HVG algorithm, and graphs that are not.

## 3.3. Effects of limited penetration

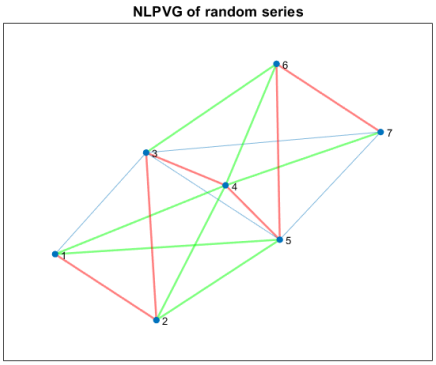
When limited penetration is applied to a VG algorithm, it changes the visibility criterion to allow a number of obstructing samples. The number of samples that is allowed to obstruct the line of sight, while still resulting in a link in the graph, is known as the penetration distance $m$. In Algorithm 1, the condition Obstructed $== 0$ is changed to Obstructed $<= m$ to effect the change in visibility criterion. An example of how this affects a VG, is shown in Figure 3.7. The same random sequence of length 7 is plotted as Figures 3.1 and 3.5. The original lines of sight according to the NVG algorithm are drawn in black, with the hamiltonian path highlighted red. The introduction of limited penetration with $m = 1$ allows for additional "lines of sight" to be added. These are drawn in red. The resulting NVG and Natural Limited Penetration Visibility Graph (NLPVG) are shown and the added links in the latter are highlighted green. In this case, not only neighboring nodes are connected by definition, but also the nodes next to the neighbors. Overall, the NLPVG is much more interconnected.




(a) Random sequence of integers




(b) NVG of integer series




(c) NLPVG of integer series

Figure 3.7: Comparison of the resulting VGs when limited penetration with m = 1 is applied

Exact effects of adding limited penetration to an algorithm have been described for the Horizontal Limited Penetration Visibility Graph (HLPVG). In this case, the HVG algorithm is used with the addition of some penetrable distance on a random time series. The exact results for the degree distribution, mean degree, and local clustering coefficient are found. The proofs follow a similar path as the ones described for the HVG, except the penetrated data point P is introduced into the configuration. Each seed node will now connect to two bounding data, and at least two penetrated data per definition of the HLPVG algorithm. The exact expression for the degree distribution $P(k)$ is now described as:

$$P(k) = \sum_{h=0}^{k-2(m+1)} (2m+1)^h [S][P]^{2m}[B]^2[I]_h[I]_{k-2(m+1)-h} \tag{3.6}$$

The expression is similar to Luque's findings for the HVG [5], with the addition of terms for the penetrated data occurring $2m$ times where $m$ is the penetration distance. Using the same method of finding the probability for each event with the bounded integrals of the probability distribution $f(x)$, the degree distribution of the HLPVG of a random series simplifies to the following equation [46].

$$P(k) = \frac{1}{2m+3}\left(\frac{2m+2}{2m+3}\right)^{k-2(m+1)} \tag{3.7}$$

From this finding, the average degree for a random series can be found by taking:

$$\bar{k} = \sum_{k=2(m+1)}^{\infty} kP(k) = 4(m+1) \tag{3.8}$$

The expression becomes more complicated for a series that exhibits periodicity. Wang et al. define a periodic sequence as '$X_t = \ldots, x_0, x_1, x_2, \ldots, x_T, x_1, x_2, \ldots$ where $x_0 = x_T$ with no repeated values in a period' [46]. Here, $T$ is the length of the period and the sample $x_T$ is the first sample of the next period. It should therefor have the same value as $x_0$, the first sample of the first period. Furthermore, the data points $x_0$ and $x_T$ are chosen as the highest points in the period. With these definitions, an iterative process of removing the lowest data point and its associated links is started. The number of links associated with the node of the lowest data point is always $2(m+1)$. This process is repeated $T-(2m+1)$ times, meaning $2(m+1)*(T-(2m+1))$ links are removed. The result is a complete graph which means there is a fixed number of remaining links. By counting the number of remaining and removed links, the number of links within a period can be found and thus the average number of links of a node within this period. It is found that the average degree of a node in an HLPVG of a periodic sequence is:

$$\bar{k}(T) = 2\left(\frac{L_{removed} + L_{remaining}}{T}\right) = 4(m+1)\left(1 - \frac{2m+1}{2T}\right), m << T \tag{3.9}$$

No exact expression for the local clustering coefficient is found, although lower and upper bounds have been defined and confirmed with numerical results [46].

## 3.4. Dual Perspective

Although only used in one paper, the Dual Perspective (DP) principle offers a very profound addition to the VG. It enables the identification of extreme events that present themselves as negative values in the time series. In Figure 3.8, this is illustrated. Sample number 4 has the largest absolute value ($x_4 = -5$), but in the NVG shown in Figure 3.8(c) node 4 is rather unremarkable with a degree of 2 and very low centrality measures. In the DPVG, the time series is reflected in the x-axis as shown in Figure 3.8(b). The NVG of this reflected series is constructed, and combined with the NVG of the original series. The DPVG is the result of the following operation:

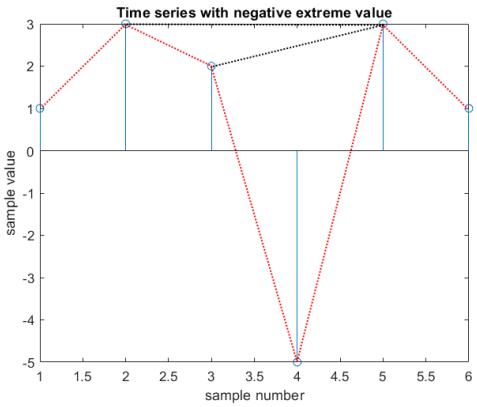
$$A_{DPVG} = or(A_{NVGoriginal}, A_{NVGreflected}) \tag{3.10}$$

The letter A indicates the adjacency matrix of the respective graphs. Because these are binary matrices, an OR operation can be used to combine them. In the resulting matrix, there will be a link when there is a link in either the original NVG, the NVG of the reflected series, or both. Figure 3.8(d) shows the DPVG, and node 4 now has the highest degree and is much more central to the graph.
The authors that proposed this method used the metric of shortest paths in the VG to do community detection [6]. This metric is heavily affected by the introduction of Dual Perspective, if there are large negative values in the time series.

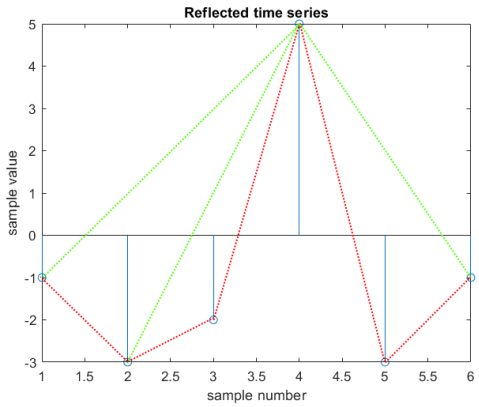
## 3.5. Metrics

Using MATLAB, various VG implementations are scripted and applied to identical sequences. The differences between the resulting graphs are measured using a collection of graph metrics, specifically average degree, clustering coefficient, average shortest path and assortativity.

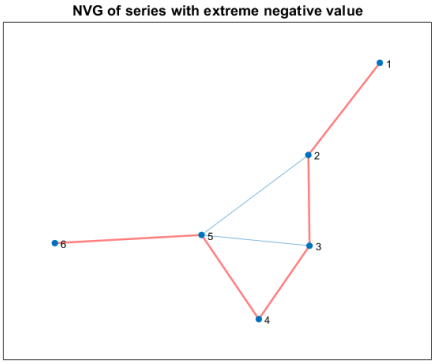
First, a random sequence of length $N = 10^3$ is generated with each entry chosen from a uniform distribution in the range $x_i \in [0, 1]$. The various VG algorithms are applied to this sequence. The



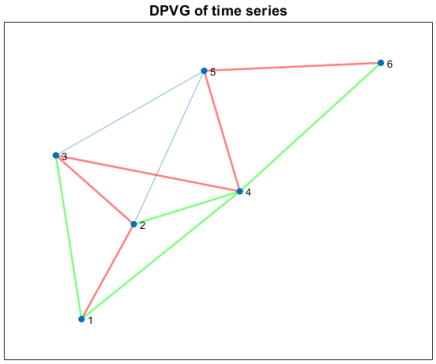

(a) Sequence with extreme value below 0




(b) The sequence in (a) reflected in the x-axis




(c) NVG of sequence in (a)




(d) DPVG constructed using sequences in (a) and (b)

Figure 3.8: The construction and effects of Dual Perspective when there is a negative extreme value present in the data

metrics of these graphs are compared in Table 3.1. Then another sequence is generated using a normal distribution for $x_i$, with $\mu = 0$ and $\sigma = 1$. The various VG algorithms are applied to that time series. The results for the time series of the normal distribution are shown in Table 3.2. Lastly, a periodic sequence is generated. One period of this sequence consists of the integers 1 through 10 in a random order. The period with the same random order is repeated 100 times to form the full sequence, resulting in a length of $N = 10^3$ samples. The metrics belonging to the graphs resulting from the periodic sequence are displayed in Table 3.3.

The values that are found for the nodal properties of HVGs are consistent with the findings in literature. The average degree $\bar{k}$ is almost equal to 4 for the random sequence. This confirms the relation given in Eq. (3.1) because for random sequences the period is equal to infinity. When changing the uniformly distributed random series to a normal distribution, the metrics change. All metrics, except for the average degree and clustering coefficient of the HVG and HLPVG, are somehow dependent on the probability distribution function of the time series. In previous sections it has been shown that the degree metrics and clustering coefficient are indeed independent of the distribution of the time series. However, Tables 3.1 and 3.2 show that this is not true for all metrics. The variance for these values over 100 realizations of the random time series also indicates that most NVG metrics are more sensitive to the stochastic process than the HVG metrics are. From the findings in Section 3.2, it was already concluded that the average degree and clustering coefficient of the HVG can be exactly determined, regardless of the probability density function of the time series. The low variance is therefore expected for the HVG and HLPVG. The length of the average shortest path is the exception. The HVG and HLPVG have higher significantly variances for this metric, compared to the NVG and NLPVG respectively. The variance for the clustering coefficient is negligible for all VG implementations, being in the order of $10^{-5}$ or lower.

Table 3.1: Various graph metrics (Average degree, clustering coefficient, average shortest path, assortativity) per VG type for the same uniformly distributed random sequence (m = 1 for all LP variants). The mean value of all 100 realizations and the variance are displayed.

| | avg. degree | | clustering c. | | avg. sp. | | assortativity | |
|---|---|---|---|---|---|---|---|---|
| | Mean | Var | Mean | Var | Mean | Var | Mean | Var |
| NVG | 5.44 | 0.005 | 0.748 | 0 | 5.82 | 0.060 | 0.178 | 0.0003 |
| HVG | 3.97 | 0.000 | 0.647 | 0 | 7.91 | 0.237 | 0.157 | 0.0002 |
| NLPVG | 10.87 | 0.014 | 0.728 | 0 | 4.43 | 0.013 | 0.218 | 0.0002 |
| HLPVG | 7.91 | 0.000 | 0.654 | 0 | 5.58 | 0.058 | 0.262 | 0.0001 |
| DPVG | 8.86 | 0.012 | 0.580 | 0 | 4.98 | 0.023 | 0.150 | 0.0002 |

Table 3.2: Various graph metrics (Average degree, clustering coefficient, average shortest path, assortativity) per VG type for the same normal distributed random sequence (m = 1 for all LP variants). The mean value of all 100 realizations and the variance are displayed.

| | avg. degree | | clustering c. | | avg. sp. | | assortativity | |
|---|---|---|---|---|---|---|---|---|
| | Mean | Var | Mean | Var | Mean | Var | Mean | Var |
| NVG | 5.86 | 0.006 | 0.752 | 0 | 5.12 | 0.052 | 0.165 | 0.0004 |
| HVG | 3.97 | 0.000 | 0.647 | 0 | 7.91 | 0.210 | 0.157 | 0.0003 |
| NLPVG | 11.64 | 0.012 | 0.734 | 0 | 3.99 | 0.012 | 0.177 | 0.0002 |
| HLPVG | 7.91 | 0.000 | 0.654 | 0 | 5.61 | 0.051 | 0.261 | 0.0001 |
| DPVG | 9.65 | 0.0155 | 0.581 | 0 | 4.39 | 0.014 | 0.120 | 0.0002 |

For the periodic sequence of period $T = 20$, the relation still holds as the average degree decreases to around 3.8. The average degree of the HLPVGs also confirms the previous findings. The VGs of the random series are used to compare the degree distributions that result from the different VG algorithms. The distributions are shown in Figure 3.10. The exponential decay of $P(k)_{HVG}$ that was predicted by Luque et al. is clearly visible here. Furthermore, the limited penetration leads to distributions that are skewed towards higher degrees as expected. The peak around degree 2 for the HVG is due to the fact that any sample that is between two larger samples will always lead to a degree of 2.
When the degree distribution corresponding to the periodic sequences are investigated, the regularity of the VGs becomes apparent. The degree sequences of all graphs are repeating integer sequences. The length of the repeating vectors in the degree sequences coincides with the length of the repeating vector in the sampled sine wave. This holds for all VG implementations, and allows for the estimation of the period of a series from the VG as has been previously noted in literature [44].

Table 3.3: Various graph metrics per VG type for an identical periodic sequence with period $T = 20$ (m = 1 for all LP variants)

| | Average degree | Clustering coefficient | Average shortest path | Assortativity |
|---|---|---|---|---|
| NVG | 4.3880 | 0.7264 | 35.2270 | -0.0598 |
| HVG | 3.7900 | 0.6731 | 35.6232 | -0.0965 |
| NLPVG | 9.1620 | 0.6980 | 18.3592 | 0.0481 |
| HLPVG | 7.3740 | 0.6130 | 18.8505 | 0.1430 |

If the resolution of a time series is lowered, and it is made up of a lower amount of unique values, this has an effect on some metrics of the time series. This is shown in Table A.1, where the sample depth $p$ is varied from $p = 16$ to $p = 1024$. Although the average degree and clustering coefficient remain fairly constant, the Average shortest path and assortativity change drastically when the sample depth is increased. The lower Average shortest path in combination with the higher assortativity suggests that a larger sampling depth leads to more connected hub nodes.
To investigate the effects of noise on the VG, a clean sine wave is generated $x_t = sin(4 * \pi * t)$. Multiple levels of additive white Gaussian noise are added to the signal. The resulting metrics are given in Table A.2. It stands out that the average degree for the HVG and HLPVG hardly varies. The VGs with limited penetration have metrics that change less as a result of noise than their non-penetration counterparts. This could explain why the feature extraction on noise signals by Xuan et al. [4] was more effective using a limited penetration method.

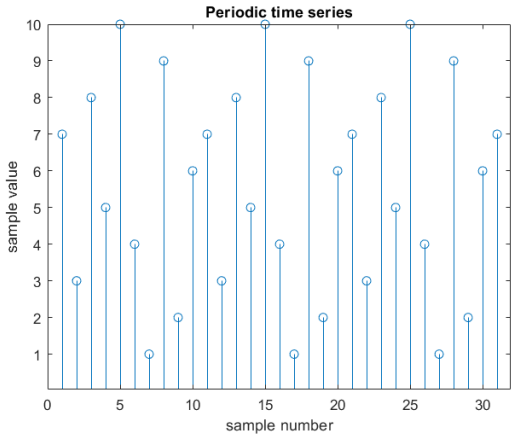



Figure 3.9: The first three periods of the generated periodic signal that is used for constructing the VGs of which the metrics are displayed in Table 3.3.

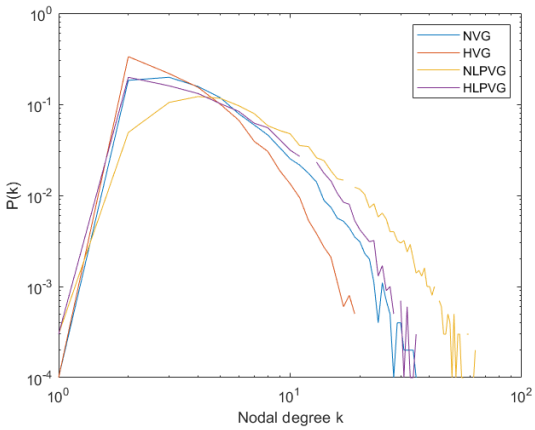



Figure 3.10: Comparison of 4 different degree distributions for the NVG, HVG, NLPVG and HLPVG algorithms on the same random sequence with values between 0 and 1 chosen uniformly.

## 3.6. Sequential motifs

Sequential motifs are defined as the subgraphs of a VG that are found along its natural Hamiltonian path, i.e. the Hamiltonian path that is formed by the consecutive samples in the time series. The set of nodes in a sequential motif is always of the form $s, s+1, ..., s+n-1$, where the nodes follow the order of the natural Hamiltonian path, $s$ is an integer between 1 and $N-n+1$, and $n$ is the length of the motif [43]. A template for a sequential motif of length 5 is shown in Figure 3.11. The red lines are links that are on the natural Hamiltonian path, with the dashed lines leading to nodes that are not part of the motif. The dotted black lines indicate possible links within the motif.

In their work, Iaovacci and Lacasa use the relative frequency of these motifs to distinguish among different dynamics of the underlying time series [43]. The motifs of length $n = 3$ and $n = 4$ found in an HVG are shown in Figure 3.12. Next to the motif types, the corresponding inequality set is displayed. This is a set of rules that the time series has to fulfil, in order to result in the motif type that is shown. For example, the first motif of length $n = 3$ is found for all time series in which $x_1 > x_0$, for any $x_0$ and $x_2$. The motif can also be found for time series in which $x_1 < x_0$, if $x_2 < x_1$ also holds. The union of these two situations form the inequality set of that motif. The nature of the VG algorithm leads to a set of admissible motifs as displayed in Figure 3.12. However, the existence of admissible motifs also implies the existence of inadmissible motifs. These are motifs that cannot be a part of VG because the corresponding inequality set has conflicting rules. The lack of such motifs is thus a characteristic that all VGs share. The description of inadmissible motif warrants further research.

The process of finding the inequailty set that corresponds to a certain motif is shown in Figure 3.13.

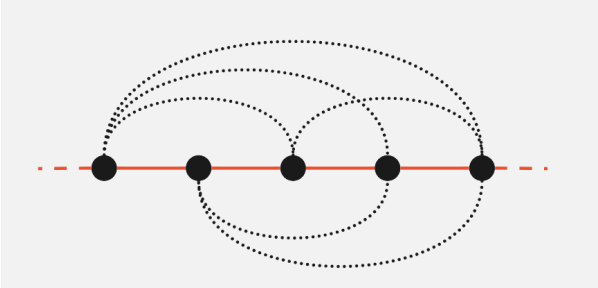


Figure 3.11: Template of a sequential motif where each red line indicates a link that is part of the Hamiltonian path and each dotted line represents a possible link within the motif

| Motif label | Motif type | Inequality set |
|---|---|---|
| 1 | | $\{\forall(x_0,x_2),x_1 > x_0\} \cup \{\forall x_0,x_1 < x_0,x_2 < x_1\}$ |
| 2 | | $\{\forall x_0,x_1 < x_0,x_2 > x_1\}$ |
| 1 | | $\{\forall(x_0,x_1),x_2 < x_1,x_3 < x_2\} \cup \{\forall(x_0,x_3),x_1 > x_0,x_2 > x_1\}$ |
| 2 | | $\{\forall x_0,x_1 < x_0,x_2 = x_1,x_3 > x_2\}$ |
| 3 | | $\{\forall x_0,x_1 < x_0,x_1 < x_2 < x_0,x_3 < x_2\} \cup \{\forall(x_0,x_3),x_1 < x_0,x_2 > x_0\}$ |
| 4 | | $\{\forall x_0,x_1 > x_0,x_2 < x_1,x_3 > x_2\} \cup \{\forall x_0,x_1 < x_0,x_2 < x_1,x_2 < x_3 < x_1\}$ |
| 5 | | $\{\forall x_0,x_1 < x_0,x_1 < x_2 < x_0,x_3 > x_2\}$ |
| 6 | | $\{\forall x_0,x_1 < x_0,x_2 < x_1,x_3 > x_1\}$ |

Figure 3.12: All admissible sequential motifs of length 3 and 4 for the HVG [43]

A motif of length 4 is extracted from an NVG. The nodes are numbered 1 through 4 and the motif can be considered an NVG of a series $x_t = [x_1, x_2, x_3, x_4]$. The adjacency matrix of the motif is shown. Certain entries in the matrix carry information about the time series. For example, the entry (3,1) is equal to 1. This means that the line of sight between $x_1$ and $x_3$ is not obstructed by $x_2$, i.e. $x_2 < x_1 + (x_3 - x_1)/2$. Similar deductions about the samples of the time series can be made using the other circled entries in the matrix.

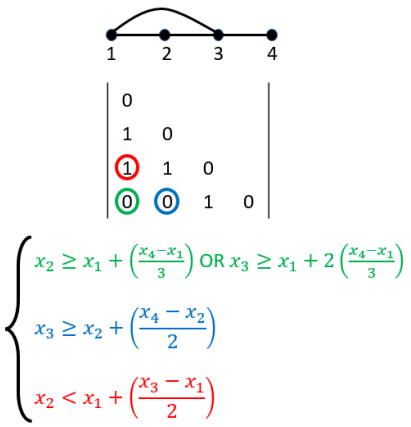



Figure 3.13: From top to bottom: A motif of length $n = 4$ extracted from an NVG, the adjacency matrix that corresponds to the motif, the inequality set corresponding to the adjancency matrix

## 3.7. Information in series and graphs

In practice, time series are mostly digital values in which each data point is represented by a number of bits. A series with length $N$, and sampling depth $p$ will have $p^N$ possible realizations. For a random time series, the contained information is the amount of samples multiplied by the number of bits per sample. The measure for the amount of information contained in a single random variable $x$ is known

as the entropy. Shannon's definition of entropy is displayed in Eq. (3.11).

$$H(X) = -\sum_{x \epsilon X} p(x) log_2(p(x)) \tag{3.11}$$

Here $X$ denotes the available alphabet, and $x$ the realization of a symbol within that alphabet. The log base 2 is taken to obtain a measure of information in bits. The expression $p(x)$ denotes the probability of realization $x$ being chosen out of alphabet $X$. For a completely random time series, all symbols are equally likely and thus entropy is maximized. For example, a variable with 256 equiprobable possible realizations constitutes 8 bits of information. A random time series with N values ranging between 1 and 256 uniformly thus contains $N \cdot 8$ bits of information. This is equivalent to stating there are $256^N$ possible realizations, and they are all equiprobable. More generally, the amount of information contained in a random time series is $N \cdot H(S)$, where $H(S)$ denotes the entropy of each individual sample.

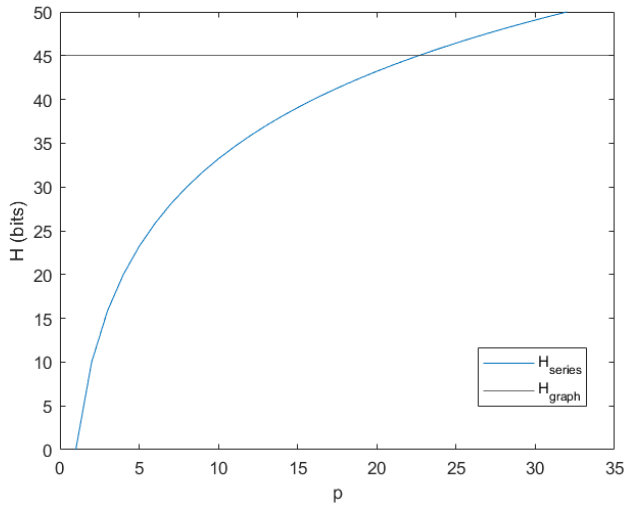



Figure 3.14: Theoretical comparison of available information in a uniform random time series of length $N = 10$ and a graph of 10 nodes. The intersection of the two lines occurs between $p = 22$ and $p = 23$.

A simple graph of size $N$ has $2^{N(N-1)/2}$ possible realizations. If each of these realizations were to be equiprobable, the graph would contain $\frac{N(N-1)}{2}$ bits of information. In this situation, each possible link can be seen as a bit of information. From the previous section, it would appear that as long as the entropy of a single sample $H(S) \leq \frac{(N-1)}{2}$ there need not be loss of information. In that case, a particular graph could ostensibly be mapped to particular realization of the time series and the transformation would be reversible. The threshold can be seen when the Shannon Entropy of both the time series and the graph are compared. Since the information content of the graph does not increase with the sampling depth of the time series, series with high enough $p$ will surpass the graph's information content. In Figure 3.14, the threshold is surpassed at $p = 23$, meaning a single sample of the uniform random time series of length $N = 10$ has an entropy of $H(S) = log_2(23) = 4.524 > \frac{(N-1)}{2}$. When larger values for $N$ are considered, the amount of possible graph realizations increases exponentially. For example, a graph of $N = 1000$ nodes has $2^{N(N-1)/2}$ realizations and thus $H_{graph} = 499500$ bits of information. That would lead to the conclusion that a time series of length $N = 1000$ could completely capture the information of a time series, as long as $H(S) \leq 499.5$ bits. However, the nature of the construction of the VG does not allow for a scenario in which all realizations of a graph with size $N$ are equiprobable or even possible at all. In Section 3.6, the existence of inadmissible motifs has been discussed. This means that not all thinkable graphs of size $N$ are eligible VGs, leading to further loss of information.

## 3.8. Shortcomings of current work

As is described in Chapter 2, there are various applications for using the concept of the VG to extract information from a time series in ways that are not possible using classical time series analysis methods. There have also been various alterations in the VG algorithm that have been proposed and used in different situations to improve the extraction of information from the time series. However, there are some fundamental features of the concept of the VG that have not been investigated in literature.

### Exact results for characteristics of the NVG

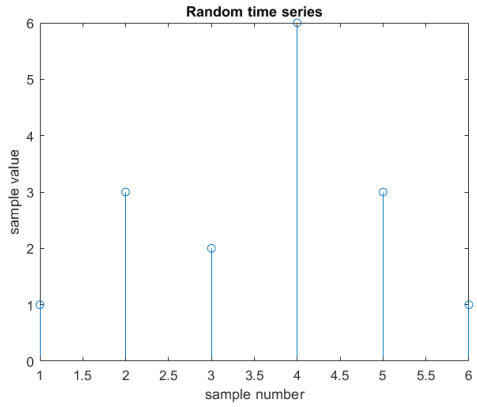
In the current literature, exact expressions for many metrics of the HVG are described. However, similar expressions for the NVG are still lacking due to the higher complexity of the NVG. Simplifications of the time series from which the NVG is constructed could lead to the possibility for exact expressions for NVG metrics too. This has not been sufficiently explored as of yet.

### Identifying a VG

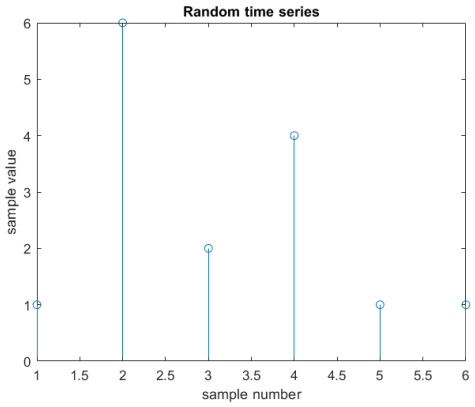
There are certain characteristics that are shared by all VGs due to the nature of their construction from a time series. These characteristics can vary depending on which variation of the algorithm is chosen. This thesis will mainly focus on the original interpretation of the VG, the NVG, as proposed by Lacasa et al. [1]. The VG can be seen as a class of graphs, but a specific definition for this class and its properties is lacking. This thesis attempts to work towards defining when a graph can be considered a VG, whether there are properties that are shared by all VGs or properties that cannot be present in a VG.

### Information retention in the VG

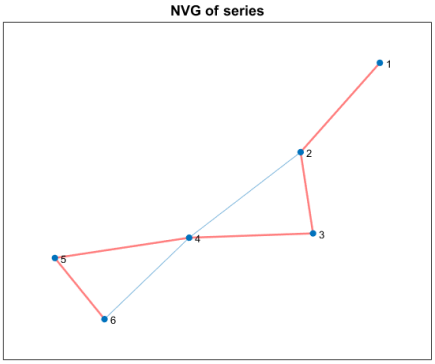
Although it is clear from previous works that information from the time series is inherited by the VG, it is not clear how the information is stored or exactly which information is stored in the VG [17]. By investigating time series with samples of limited precision, this thesis attempts to work towards a definition of how much information is retained in the VG and how this affects the possibility for an inverse transformation. It is obvious that in the current form, and inverse transformation is not possible because multiple time series can map to the same VG. This is shown in Figure 3.15. A measure, or even an estimate of the amount of lost information has not been the subject of research as of yet. This thesis will investigate the loss in information and attempt to quantify it.



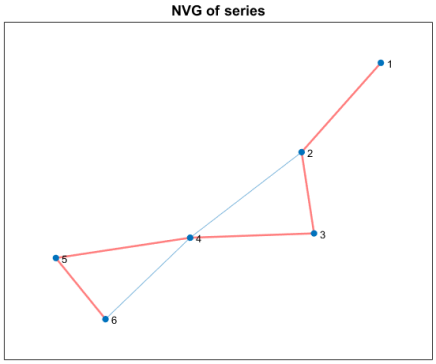

(a) First random time series




(b) Second random time series




(c) NVG of sequence in (a)




(d) NVG of sequence in (b)

Figure 3.15: An example of how two different time series can lead to the construction of the same graph

# 4

# Expected degree for NVG

Exact results have been derived for the degree properties in HVGs, however it has not been done for NVGs. It is attempted in the following section to find exact results for nodal properties of the NVG of an infinite uniformly distributed random time series. As a start, the probability of links occurring between the first node and following nodes is investigated. The condition for a link to occur between the first node and a node $i$, is that all samples $c$ for which $t_1 < t_c < t_i$ the sample value $x_c$ lies below the line of sight of $x_1$ and $x_i$. The probability of this happening can be found by integrating the probability density function $f(x)$ from bounds 0 to the line of sight which is given by Eq. (2.3). The principle is visualized in Figure 4.1.

| Links: | Probability of link occurring: |
|---|---|
| $x_1 \rightarrow x_2$ | $1$ |
| $x_1 \rightarrow x_3$ | $\int_0^{x_1+\frac{x_3-x_1}{2}} f(x_2)dx_2$ |
| $x_1 \rightarrow x_4$ | $\int_0^{x_1+\frac{x_4-x_1}{3}} f(x_2)dx_2 \cdot \int_0^{x_1+2\frac{x_4-x_1}{3}} f(x_3)dx_3$ |

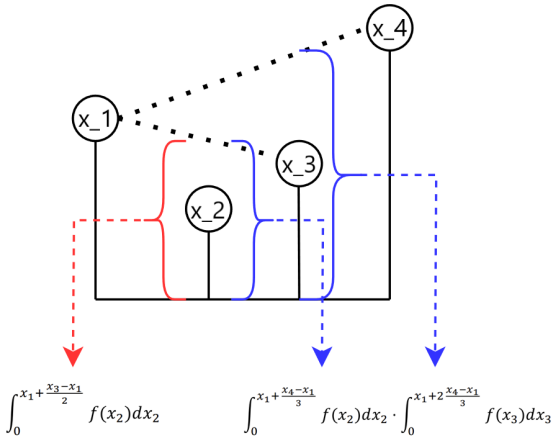



Figure 4.1: Visualization of how the probability for a link in an NVG is formulated. The region over which the probability distribution is integrated depends on the line of sight between two samples and the distance to the sample in between.

The sum of these probabilities amounts to the expected degree of the first node. To stay consistent with the notation of previous works, $x_b$ is the sample that may or may not have a line of sight with the sample under investigation while $x_c$ is a sample that lies in between the sample under investigation and $x_b$. Without loss of generality, the expected degree for the first sample can be written as:

$$E[k_1] = 1 + \sum_{b=3}^{N} \prod_{c=2}^{c=b-1} \int_0^{x_1 + \frac{x_b - x_1}{b-1}(c-1)} f(x_c) dx_c \tag{4.1}$$

Where $k_1$ indicates the degree of node 1. The issue with this expression, when compared to the case of the HVG, is that the upper bound of the integral depends on multiple values of the time series. This does not allow for the rewriting of the integral as is done by Luque et al. [5] for the HVG scenario in which the bound is only dependent on the first node. When the expected degree of a random node $a$ is investigated, the left and right hand side of time series both contribute to the degree. With a guarantee of one neighbor on each side, the expected degree can be described as:

$$E[k_a] = 2 + \sum_{b=1}^{a-2} \prod_{c=b+1}^{a-1} \int_0^{x_a + \frac{x_b - x_a}{b-a}(c-a)} f(x_c) dx_c + \sum_{b=a+2}^{N} \prod_{c=a+1}^{b-1} \int_0^{x_a + \frac{x_b - x_a}{b-a}(c-a)} f(x_c) dx_c \tag{4.2}$$

If the time series $x_t$ is assumed to be infinite, symmetry between the contributions of the left and right hand side of node $a$ can be expected and the expression of the expected degree becomes:

$$E[k_a] = 2 + 2 \sum_{b=a+2}^{\infty} \prod_{c=a+1}^{b-1} \int_0^{x_a + \frac{x_b - x_a}{b-a}(c-a)} f(x_c) dx_c \tag{4.3}$$

However, without a method of circumventing the multiple dependencies of the bounds, there is no clear path to an exact result for the expected degree in an NVG.

## 4.1. NVG of integer series

In order to arrive at an exact result, some assumptions on the time series have to be made. When the time series is defined such that the values of $x(t) \in [0, 1, 2, ..., p]$, more progress can be made. Using a simplified scenario where

$$f(x) = \begin{cases} x = 0 \\ x = 1 \end{cases} \tag{4.4}$$

it becomes possible to determine the expected degree of any node. The proof is similar to that in the HVG and is constructed as follows. A sample $x_0$ is taken as the seed sample. Its neighbors $x_{-1}$ and $x_1$ are always connected to the seed. Then, depending on the order of the samples, there can be two bounding samples $x_{b-}$ and $x_{b+}$ on respectively the left and right side of the seed sample. Between the seed and the bounding samples, there can be a number of inner samples denoted with $x_i$. All configurations are displayed in Figure 4.2, along with the degree of node 0. The expected degree of node 0 can be found by finding the probability of each configuration occurring, and multiplying that with the degree corresponding to that configuration. The sum of these values is the expected degree $E[k_0]$.

In configuration 1 ($C_1$), the neighbors of the seed node are simultaneously the bounds. The probability $P(x_0 = 1 \cap x_{-1} = 1 \cap x_1 = 1)$ is equal to $\frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{8} = P(C_1)$. With a degree of $k_{0(1)} = 2$, this configuration contributes $\frac{1}{4}$ to $E[k_0]$. The same holds for $C_2$. $C_3$ has a degree $k_{0(3)} = 4$ and the probability of it occurring is equal to $P(C_3) = P(x_0 = 0 \cap x_{-1} = 0 \cap x_1 = 0) = \frac{1}{8}$. Because this is an infinite time series, the probability of there being some bounding sample on either side is 1. Thus only the seed and the neighboring samples are of importance when calculating the probability. Next, it is important to recognize that $C_4$ has a symmetrical counterpart that is not shown in Figure 4.2, but the situation could of course be mirrored in the y-axis at the place of $x_0$. The probability for $C_4$ occurring should thus be doubled. This results in $P(C_4) = 2\frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$. In $C_5$, there are possible inner samples present. These

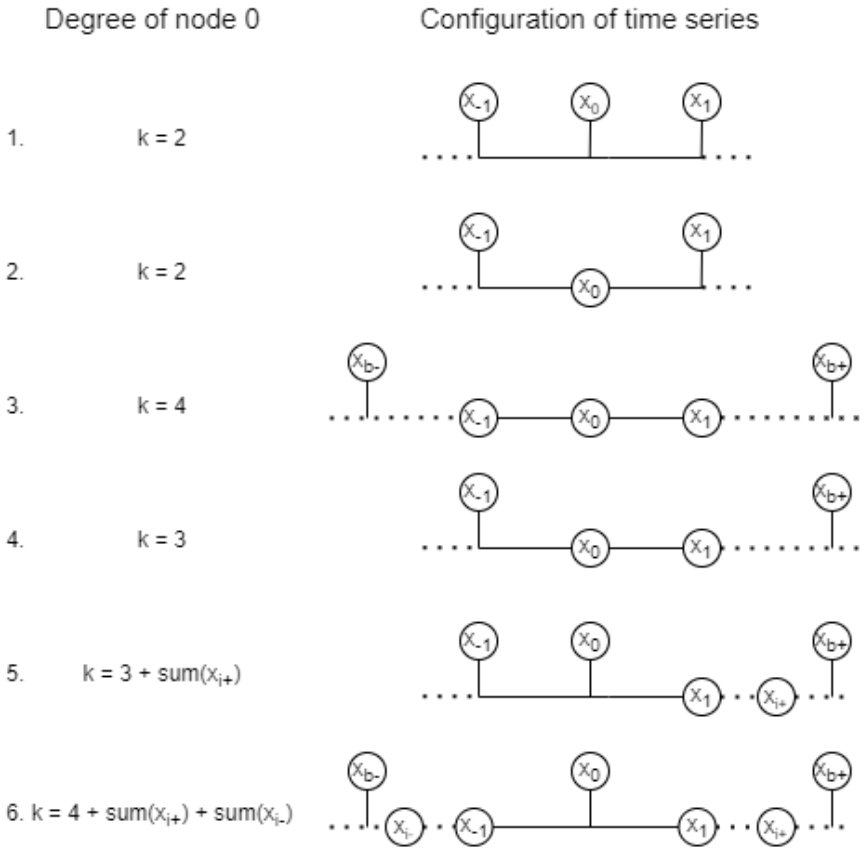



Figure 4.2: All possible configurations surrounding $x_0$ for a binary time series. The configurations are numbered and the degree of node 0 is listed per configuration. Configurations 4 and 5 can also occur in their mirrored version. The dotted lines indicate that there might be more samples in between. For configurations 5 and 6, this leads to an infinite number of permutations with different degrees for $x_0$. However, the probability of each of these permutations is known so an expected degree for $x_0$ can still be formulated.

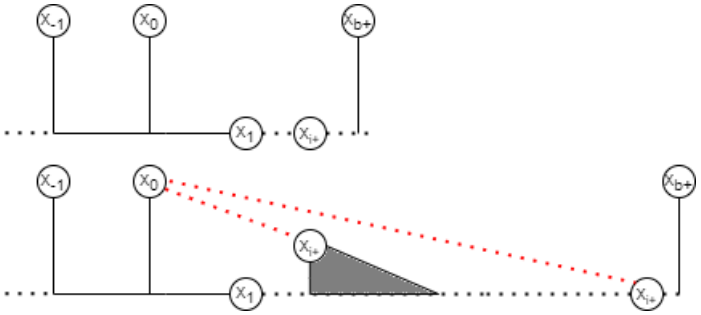


Figure 4.3: Configuration of time series where $x_{-1} = x_0 = 3$ and $x_1 = 0$, leading to inner samples between the seed and the bound. The illustration below depicts how this can lead to infinitely many configurations for an infinite time series.

are possibly infinite samples that are visible to $x_0$, but they all have to be equal to 0. Furthermore, similar to $C_4$, $C_5$ also has a symmetric counterpart. Therefore the probability $P(C_5) = 2\frac{1}{2}\cdot\frac{1}{2}\cdot\frac{1}{2} = \frac{1}{4}$. The expected degree, due to the 3 guaranteed visible samples in both neighbors and the bound and the presence of possibly infinite inner samples, is defined as $3 + \sum_{i=1}^{\infty}(\frac{1}{2})^i = 4$. The final configuration $C_6$ has inner samples on both sides of $x_0$, resulting in a degree $k_{0(6)} = 4 + 2\sum_{i=1}^{\infty}(\frac{1}{2})^i = 6$. The probability is equal to $P(C_6) = \frac{1}{2}\cdot\frac{1}{2}\cdot\frac{1}{2} = \frac{1}{8}$. Combining all these findings:

$$E[k_0] = P(C_1)\cdot k_{0(1)} + P(C_2)\cdot k_{0(2)} + P(C_3)\cdot k_{0(3)} + P(C_4)\cdot k_{0(4)} + P(C_5)\cdot k_{0(5)} + P(C_6)\cdot k_{0(6)} = 3.5 \quad (4.5)$$

This value is confirmed by generating a random series of N = 1000 ones and zeros in MATLAB and constructing its NVG. The average degree of the resulting NVG is equal to $\bar{k} = 3.514$. Similar proofs falter when the number of possible values for the samples in $x_t$ is increased. For example take the case where

$$f(x) = \begin{cases} x = 0 \\ x = 1 \\ x = 2 \end{cases} \quad (4.6)$$

And consider the configuration as displayed in Figure 4.3. While the configuration may seem comparable to $C_5$ in Figure 4.2, the key difference is that there is now an infinite amount of possible arrangements for the inner samples. If one of the samples $x_i = 1$, then subsequent samples $x_{i+1}$,$x_{i+2}$, etc. will not be visible if they are equal to 0, but a sample $x_{i+t} = 0$ may be visible for large enough $t$. Since the time series is infinite, it is no longer possible to define a finite number of configurations.
The expression for the expected degree as given in Eq. (4.3) changes in the discrete scenario. Because there is now a finite number of possible values for all samples of $x_t$, there is also a limited number of combinations $(x_a, x_b)$ and a limited number of values for $x_c$ that allow a line of sight. Let all samples of $x_t$ be an integer such that $x_t \in [1, 2, \ldots, p]$ where each integer has the same likelihood $\frac{1}{p}$ of occurring. For a given combination $(x_a, x_b)$, the probability of an intermediate sample $x_c$ not blocking the line of sight is now given as a sum instead of an integral:

$$Pr(x_c < x_a + \frac{x_b - x_a}{b - a}(c - a)) = \sum_{1}^{\lfloor x_a + \frac{x_b - x_a}{b-a}(c-a)\rfloor} \frac{1}{p} \quad (4.7)$$

The upper bound of the sum is the nearest integer below the line of sight. This can be calculated for each combination $(x_a, x_b)$, with each of these combinations being equiprobable at $Pr(x_a, x_b) = \frac{1}{p^2}$. The expression for the expected degree in the NVG of an infinite discrete time series then becomes:

$$E[k_a] = 2 + \frac{2}{p^2}\sum_{b=a+2}^{\infty}\sum_{x_a=1}^{p}\sum_{x_b=1}^{p}\prod_{c=a+1}^{b-1}\sum_{1}^{\lfloor x_a + \frac{x_b - x_a}{b-a}(c-a)\rfloor}\frac{1}{p} \quad (4.8)$$

The construction of the expression in Eq. (4.8) is further elaborated upon in Figure 4.4.

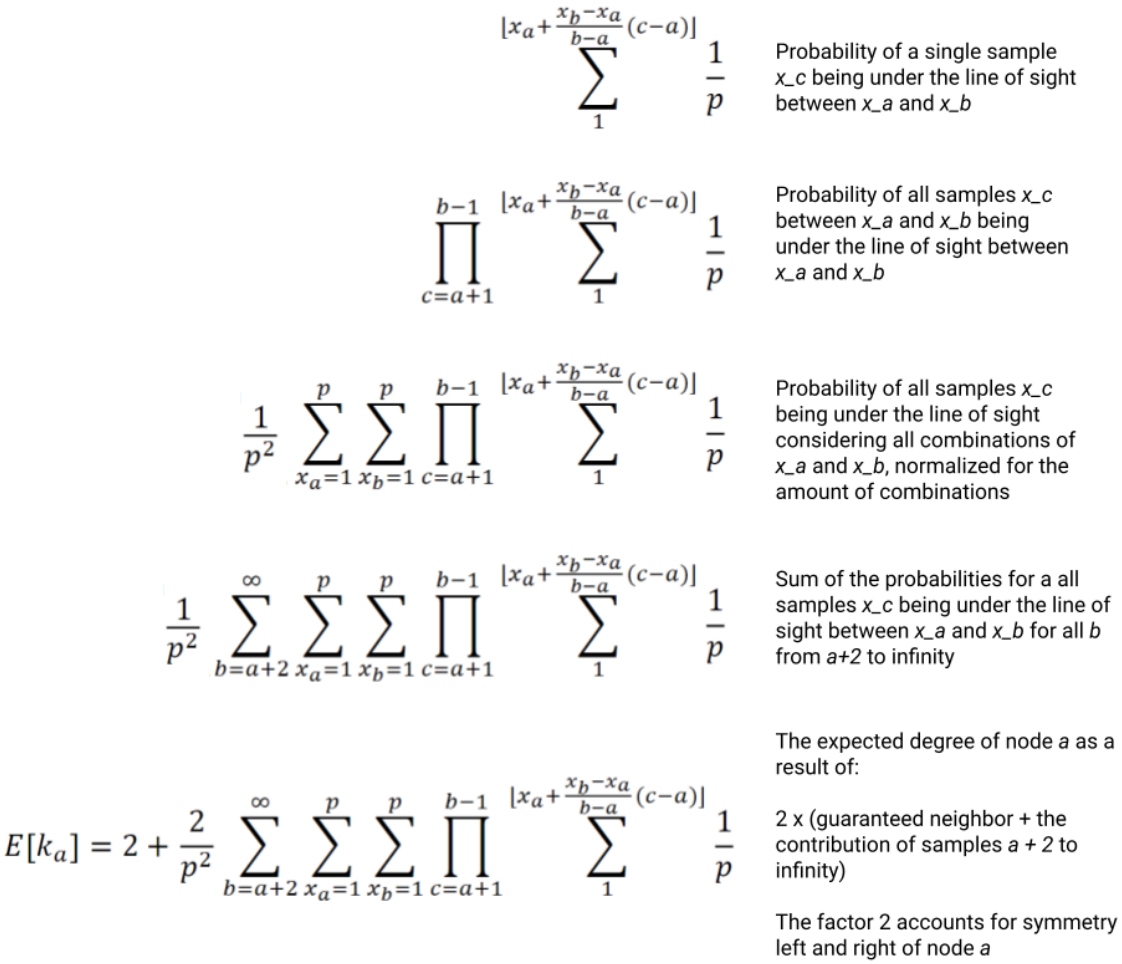



Figure 4.4: Step by step illustration of how the expression in Eq. (4.8) is formed. First a single sample is considered, then the collection of samples between seed $x_a$ and destination $x_b$. All combinations of values for samples $x_a$ and $x_b$ are then taken into account. Then all samples on the right hand side of $x_a$ are considered as possible destinations. The sum of the probabilities of these destinations having a line of sight with the seed, added to the guaranteed visible node next to the seed, form the expected right hand side degree of node $a$. This must be multiplied by 2 to find the total degree of node $a$, due to symmetry on the left hand side.

Since there are now a limited number of possible combinations, it is possible to estimate the expected degree of a random node $a$ in the NVG. To test the accuracy of the estimate given in Eq. (4.8), a MATLAB script is created that calculates the estimate for a given $p$. An infinite series is not feasible. For the sake of limiting computational complexity, a length of $N = 1000$ is selected. The script also generates a discrete time series with the same sampling depth $p$ and length $N$. The estimated average degree can be compared to the average degree that is found in the generated NVG of the discrete time series. The estimated value $\hat{E}[k]$ is deterministic and will always produce the same results for the same parameters $p$ and $N$, while the calculated average degree from a generated NVG will vary per realization of the time series. The variations are relatively small for large $N$.

## 4.2. Verification of $E[k]$

In order to verify the proposed estimator for the expected degree in an NVG for an integer series, a MATLAB script is used that compares the results from the estimator to the results from 100 randomly generated NVGs. As is evident from the first two lines, Algorithm 2 loops through all but the last three

**Algorithm 2** Estimate of average degree in discrete time series

```
1:  for a = 1 : N − 3 do
2:      for b = a + 2 : N do
3:          for x_a = 1 : p do
4:              for x_b = 1 : p do
5:                  c = [a+1:b-1]
6:                  x_c_list = [ ]
7:                  for i = length(c) do
8:                      for x_c = 1 : ⌊x_a + ((x_b − x_a)/(b − a)) · (c(i) − a)⌋ − 10^−14 do
9:                          x_c_list(i) = x_c_list(i) + 1/p
10:                     end for
11:                     prob(x_a, x_b, b) = prod(x_c_list)
12:                 end for
13:             end for
14:             prob_link(b) = sum(prob(:,:,b)) / p^2
15:         end for
16:     end for
17:     E[k](a) = 2 + 2 ∗ sum(prob_link)
18:     prob = []
19:     prob_link = []
20: end for
21: avg_degree_estimate = mean(E[k])
22: data = randi(p, N, 1)
23: graph = NVG(data)
24: avg_degree = mean(degree(graph))
```

entries and only measures the contributions from the right hand side which it doubles, in accordance with Eq. 4.8. In lines 1 and 2, a pair of entries is chosen, represented by $a$ and $b$. In the subsequent two lines, all possible combinations of the values of these entries are considered by looping through 1 to $p$ for both $x_a$ and $x_b$. Then all entries between $a$ and $b$ are represented in the array $c$. For each entry between $a$ and $b$ and for each combination of $x_a$ and $x_b$, the probability of an entry $x_c$ being so that it does not obstruct the line of sight is calculated. An example for the process of the lines 7 to 11 is worked out for clarity. Take $p = 3$, $a = 1$ and $b = 4$. In this situation there are two entries $c = [c_1, c_2]$ in between $a$ and $b$. The first combination of $(x_a, x_b)$ is (1,1), in which case the limit in line 8 is immediately reached and the for loop is not entered. The for loop in line 8 is analogous with the summation $\sum_1^{\lfloor x_a + \frac{x_b - x_a}{b-a}(c-a) \rfloor} \frac{1}{p}$ and counts the number of possible values of an entry $c$ below the line of sight of $a$ and $b$. The chance of $a$ and $b$ being connected is equal to 0 for the combination (1,1) which is correct because both entries $x_{c1}$ and $x_{c2}$ are at least equal to 1, and therefore obstruct the line of sight. The

Table 4.1: Table of probabilities for a link occurring between $a$ and $b$ with two samples in between for the given combinations of $x_a$ and $x_b$ and $p = 3$.

| $x_a$ \ $x_b$ | 1 | 2 | 3 |
|---|---|---|---|
| 1 | 0 | $(1/3)^2$ | $(1/3)^2$ |
| 2 | $(1/3)^2$ | $(1/3)^2$ | $(2/3)^2$ |
| 3 | $(1/3)^2$ | $(2/3)^2$ | $(2/3)^2$ |

Table 4.2: Comparison of the estimated average degree $E[k]$ and calculated average degree from 100 realizations of time series $\bar{k}$.

| | $E[k]$ | $\bar{k}$ | % deviation |
|---|---|---|---|
| p = 2 | 3.4888 | 3.4860 | 0.08% |
| p = 4 | 4.2644 | 4.2694 | -0.12% |
| p = 8 | 4.7789 | 4.7776 | 0.027% |
| p = 16 | 5.0844 | 5.0850 | -0.012% |
| p = 32 | 5.2541 | 5.2577 | -0.068% |

next combination is (1,2), in which case there is exactly one value for $x_{c1}$ under the line of sight and exactly one value for $x_{c2}$ under the line of sight. The for loop in line 8 is now entered once for both $x_{c1}$ and $x_{c2}$. Because both entries have 1 possible value out of $p$ possible values to be below the line of sight, the chance of each of them individually being below the line of sight is equal to $1/p$. This is stored in the array x_c_list. But for the line of sight between $a$ and $b$ to hold, both events of $x_{c1}$ and $x_{c2}$ being below the line of sight have to happen simultaneously. This is why the product of all entries in x_c_list in line 11 is taken as the probability for the line of sight to hold for the given combination ($x_a$, $x_b$). For a given $a$, $b$ and $p$, the probabilities are stored in a $pxp$ array as shown in Table 4.1. Because each of these combinations is equally likely to occur, the sum of the probabilities normalized by $p^2$ can be taken as the probability of a link occurring between $a$ and $b$. This process is done for all pairs in the series, and thus the expected number of links, i.e. the expected degree, of each node can be estimated.

The estimate proves to be fairly accurate when $p << N$, as the estimate is based on the assumption of an infinite series. Due to the computational complexity of the algorithm, the length $N$ is chosen at $N = 1000$ and the maximum value for sampling depth at $p = 32$. The results of the estimation can be compared to the average degrees of actual NVGs that are generated under the same parameters $p$ and $N$[1]. For statistical accuracy, the average degrees of 100 realizations of each value $p$ are aggregated. The findings for these parameters are shown in Table 4.2. The estimate is very close to the measured value of the 100 realizations for each value of $p$. The estimator does not consistently over or under shoot and is always within 0.12% or less of the measured value. It can thus be concluded that the expected nodal degree of an NVG constructed from a discrete time series of random integers is given by the expression in Eq. (4.8).

## 4.3. Degree distribution of the NVG of integer series

Using the assumption of a limited integer series, the expressions for the likelihood of links are simplified. After the estimator for the average degree, investigating the degree distribution is a logical next step as this metric has also been worked out for the HVG and could be used as a stepping stone to more metrics as seen in Section 3.2. For the HVG, the degree distribution $P(k)$ is determined by identifying three different types of data from the perspective of some seed datum $S$. The first of these three types are the bound data $B$, which are the furthest data that the seed datum has a line of sight with. There is a bound datum at either side of the seed. The second type are the inner data, which are data points that share a line of sight with the seed, but do not block the lines of sight to all other data points behind it. The number of inner data is dependent on the degree of the seed node and the inner data can be both on the left and right hand side of the seed datum. Lastly, there are the hidden inner data which are data points that are between the seed and the bound but are hidden from the seed by some other

[1] Generating discrete time series and NVG for reference in lines 21-24 of Algorithm 2. The NVG is generated using 1 shown here as the function NVG(). The built-in MATLAB functions mean() and degree() provide the rest.

inner data point.
With the simplification of the limited integer time series, the likelihood of the occurrence each of these types of data can be expressed. For the bound data, the probability is denoted as [B], and is equal to the probability that the datum is higher than the lines of sight that all following data share with the seed datum. For example, if there is a seed datum $x_s$ and some datum $x_b$ on the right hand side of the seed, then $x_b$ is a bound datum for $x_s$ if $x_b > x_s + \frac{(x_n - x_s)(t_b - t_s)}{t_n - t_s}$ for all $c > b$ if $x_s$ and $x_b$ share a line of sight. A simpler interpretation is to state that all samples on the right hand side of the seed must be below the line of sight between $x_s$ and $x_b$. This is shown in Figure 4.5.

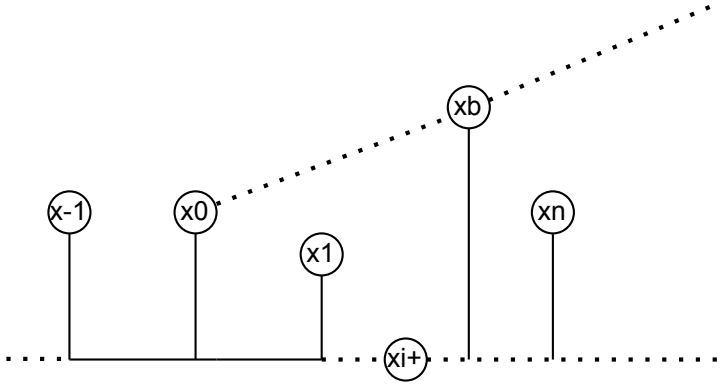



Figure 4.5: A seed datum $x_0$ and a datum $x_b$ are shown with their shared line of sight. If all data on the right hand side of $x_0$ are smaller than the dashed line of sight, $x_b$ is the right hand bound for $x_0$

This event can be described by extending the expression for the probability of line of sight. For a general expression, let there be a seed datum $x_s$ and some datum $x_d$ at a distance $d$ on the right hand side of the seed. All samples on the right hand side must be below the line of sight between $x_s$ and $x_d$. The probability of this event for some sample $x_d$ in for an infinite random discrete time series with sampling depth $p$ is shown in Eq. (4.9).

$$P(\mathrm{x_d \text{ bounds } x_s}) = \frac{1}{p^2} \sum_{x_s=1}^{p} \sum_{x_d=1}^{p} \prod_{\substack{n=s+1 \\ n \neq d}}^{\infty} \sum_{1}^{x_n = \lfloor x_s + \frac{x_n - x_s}{n-s}(d-s) \rfloor} \frac{1}{p} \tag{4.9}$$

When applying this to a time series of length $N = 1000$ and sampling depth $p$, the probability for blocking samples at a certain distances can be found. The first datum ($x_0$) of the time series is considered the seed datum. Then the probability is calculated for each datum to its right ($x_1$, $x_2$,...$x_N$) being the bound datum. The results for time series with varying $p$ are shown in Figure 4.6. The first entry to the right of the seed has a probability of approximately 37% to be the bound for the seed. The shape of the distribution does not seem to depend on the value of $p$. A verification of the result is done by finding the distances between seed and bound for each node in 100 NVG realizations of size $N = 1000$ and $p = 64$. In Figure 4.7, it is shown that the found distribution very closely resembles the predicted distribution in Figure 4.6. The distance-to-bound distribution also shows how many inner data there typically are, namely the distance between the bound and the seed. That moves one step closer to finding an expression for the degree distribution. What remains is an expression for the expected amount of inner data that are visible to the seed.

## 4.4. Conclusions

This chapter has shown that using certain limitations to the time series, which are not far fetched for real world scenarios, progress can be made towards finding exact metrics of the NVG. The average degree of a random limited integer time series can be accurately predicted. Strides are made towards finding an expression for the degree distribution $P(k)$. It is possible to find the distribution of distance at which a bounding datum occurs. This in turn means that there is a distribution of the number of inner data. What remains to be found is an expression for the amount of inner data that are visible to the seed.

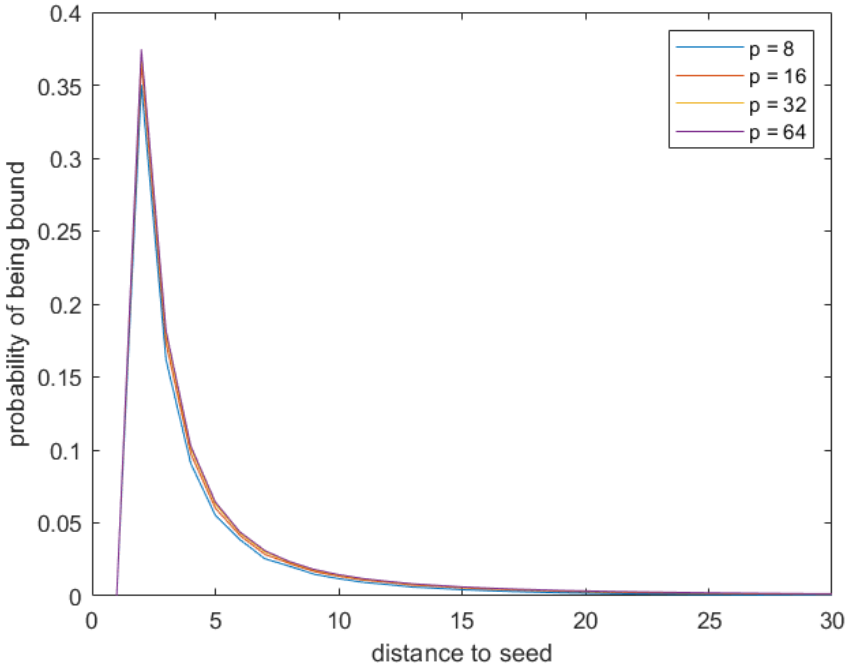



Figure 4.6: The distribution of distances from the seed datum to the right hand bounding datum for varying $p$ as a result of the expression in Eq. (4.9).

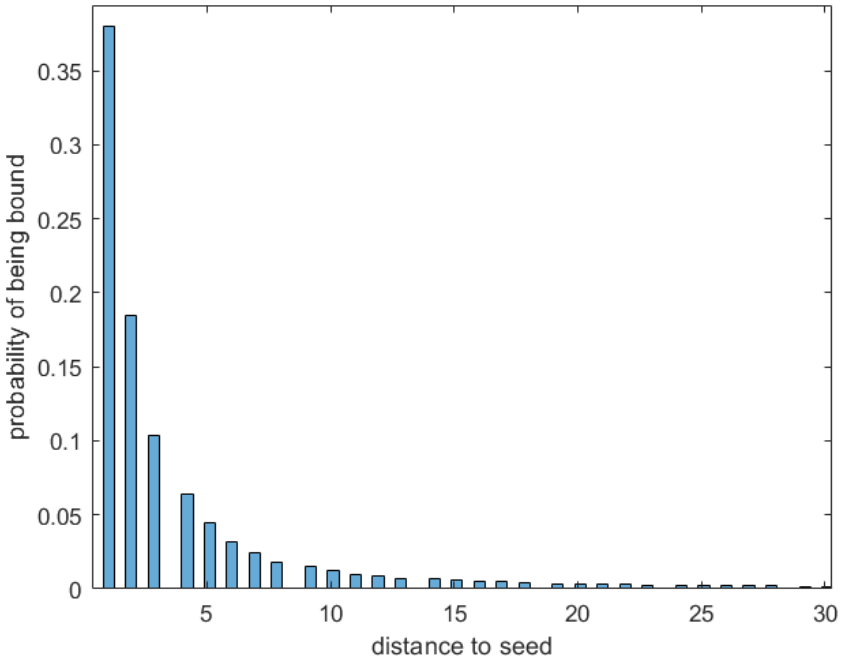



Figure 4.7: The distribution of distances from the seed datum to the right hand bounding datum as a result of 100 realizations of a uniformly distributed time series of length $N = 1000$ with sampling depth $p = 64$. The apparently missing bars are a result of the MATLAB implementation of histograms, not a reflection of distances that are never a bound for a seed.

# 5

# Identifying a VG

As described in Chapter 3, the graphs resulting from a VG algorithm share certain characteristics. The VGs can be said to form a class of graphs, although this class has not been strictly defined as of yet. This chapter works towards a definition of such class of graphs by investigating the features that identify a graph as an NVG. The chapter focuses on the NVG, because it is the most ubiquitous of the VG methods along with the HVG. On the latter there is already more knowledge such as a description of motifs. The admissible motifs described in Section 3.6 are used as a starting point. For the purposes of the identification of VGs, three types of motifs are defined: the admissible motif, the inadmissible motif, and the unavoidable motifs. The first type has already been described. The second type contains the motifs that, if found in a graph, disqualify that graph from belonging to the class of VGs. The last type are the motifs that will always occur in a VG.

## 5.1. Sequential motifs in the NVG

Expanding on the work done by Iovacci and Lacasa [43], this chapter investigates the motifs of any graph up to a length of $n = 5$. Then, it is investigated which information about the time series these motifs would carry, if they were constructed using the NVG algorithm. This information comes in the form of an inequality set, akin to the inequality sets in Figure 3.12. Similar to the motifs in Figure 3.12, only the sequential motifs (i.e. the motifs along the natural Hamiltonian path) are considered in this chapter. Using MATLAB, all sequential motifs of lengths $n = 3, 4, 5$ are generated, along with the inequality sets. The number of sequential motifs possible increases with the length of the motif. This becomes evident when examining the adjacency matrix of a generic NVG. The diagonal of a VG is all zeros, because there are no self loops. The second diagonal, i.e. (2,1), (3,2) ... (N,N-1), is all ones because two neighboring samples are always connected in the graph. Therefore, the triangle below[1] the second diagonal of the adjacency matrix is where differences can start to occur. This is visualized in Figure 5.1.

The number of conceivable sequential motifs will thus scale with the number of entries below the second diagonal. Let $LTE$ be the number of lower triangle entries. Then as $N$ grows, $LTE$ grows with

$$LTE = \frac{(N-2)(N-1)}{2}, \tag{5.1}$$

meaning that the number of sequential motifs will grow as

$$n_{seq.motifs} = 2^{LTE}. \tag{5.2}$$

The number of conceivable sequential motifs, scales quadratically with $LTE$ for unweighted graphs because each entry can only take the value of one or zero. The scaling properties are seen when the motifs for lengths $n = 3, 4, 5$ are examined in Figure 5.2. However, not all motifs that are conceivable in a graph, are also admissible in a VG. When generating VGs for a large number of sequences with

[1] For undirected graphs, the same holds for the upper triangle

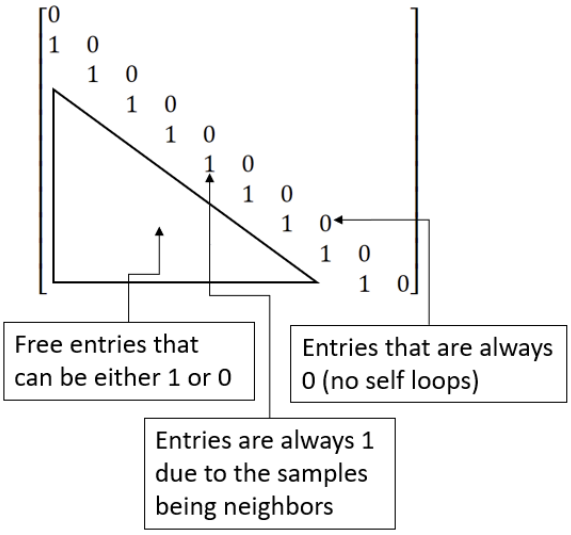



Figure 5.1: The adjacency matrix of any visibility (sub)graph

length $n = 4$, two motifs remain absent. These are the inadmissible motifs. For length $n = 5$, the number of inadmissible motifs is 49. Of the 64 motifs that are conceivable, only 25 (marked green in Figure 5.2) are a possible result of the NVG algorithm being applied to a time series of 5 entries. This can be deduced from their corresponding inequality sets.

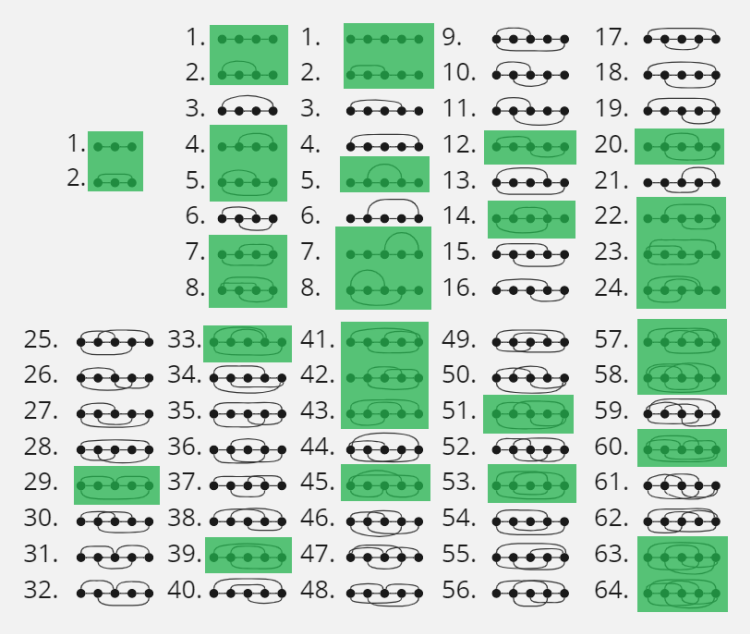


Figure 5.2: Collection of all sequential motifs theoretically possible for lengths $n = 3, 4, 5$, admissible motifs marked green

### Generating the inequality set

The inequality sets of the motifs are not shown in Figure 5.2, because they get increasingly more complex for longer motifs. For the two motifs of length $n = 3$ the inequality sets are just one line each: $x_2 \geq x_1 + (x_3 - x_1)/2$ and $x_2 < x_1 + (x_3 - x_1)/2$ for motifs one and two respectively. The adjacency matrix and inequality set for motif #62 of length $n = 5$ are shown. Such a set of inequalities can be derived from the adjacency matrix of the VG, as shown in Figure 3.13. Each entry in the lower triangle of the matrix provides one or more lines in the inequality set.

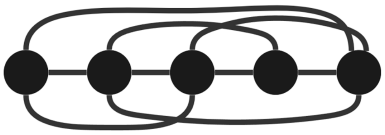


Figure 5.3: Motif number 62 of length $N = 5$ (above), and its Adjacency matrix (below)

$$\begin{bmatrix} 0 & 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 1 & 0 \end{bmatrix}$$

$$\begin{cases} x_2 \geq x_1 + \frac{x_4 - x_1}{3} || x_3 \geq x_1 + \frac{2(x_4 - x_1)}{3} \\ x_2 < x_1 + \frac{(x_3 - x_1)}{2} \\ x_2 < x_1 + \frac{(x_5 - x_1)}{4} \\ x_3 < x_2 + \frac{x_4 - x_2}{2} \\ x_3 < x_1 + \frac{2(x_5 - x_1)}{4} \\ x_3 < x_2 + \frac{(x_5 - x_2)}{3} \\ x_3 < x_2 + \frac{(x_5 - x_2)}{3} \\ x_4 < x_1 + \frac{3(x_5 - x_1)}{4} \\ x_4 < x_2 + \frac{2(x_5 - x_2)}{3} \\ x_4 < x_3 + \frac{(x_5 - x_3)}{2} \end{cases} \tag{5.3}$$

These sets can be deduced from inspecting the adjacency matrix of the VG, as shown in Figure 3.13. This process can be automated using a MATLAB script. The function takes the VG as input. Each entry in the lower triangle of the adjacency matrix is analyzed. If a 0 is found at entry (a,b), this means that there is at least one sample $x_c$ for which

$$x_c \geq x_a + \frac{(x_b - x_a)(t_c - t_a)}{t_b - t_a} \tag{5.4}$$

with $(t_a < t_c < t_b)$. This sample $x_c$ blocks the line of sight between $x_a$ and $x_b$. A string is stored in a cell. The string describes Equation (5.4) for each of the samples between $x_a$ and $x_b$. An OR operator is placed in between each of the inequalities to indicate that any only one of the inequalities needs to hold, but multiple may be true. The first line of set (5.3) is an example of such a string. The line of sight between $x_1$ and $x_4$ is blocked. This can be due to $x_2$ or $x_3$, which is described in the inequality set.
When a 1 is found at an entry (a,b), it means that all samples $x_c$ are lower than the line of sight between $x_a$ and $x_b$. This means that the following must hold for all samples $x_c$ that are in between $x_a$ and $x_b$

$$x_c < x_a + \frac{(x_b - x_a)(t_c - t_a)}{t_b - t_a} \tag{5.5}$$

The expression in (5.5) is generated as a string for each $x_c$, with AND operators in between because all expressions need to hold. The strings resulting from the 0 and 1 entries in the adjacency matrix can be combined as is done in set (5.3).

## 5.2. Inadmissible motifs

As discussed in Section 3.6, the existence of admissible motifs also implies the existence of inadmissible motifs. These are shapes or patterns that can never appear in a VG. In this section, these motifs are further investigated. By inspection of Figure 5.2, two common patterns are found in all inadmissible motifs, i.e. the motifs that are not marked. The patterns are further elaborated on in the following subsections.

### Overlapping pairs

The graph in Figure 5.4 is given and it is to be determined whether it could be a VG of a certain time series. The nodes are numbered in this graph, as the ordering of the nodes is important when determining whether the motif can be found in a VG. Just like the motifs shown in Figure 3.12, the

straight line across all nodes follows what would be the natural Hamiltonian path of the VG. The motif is henceforth referred to as 'overlapping pairs'.

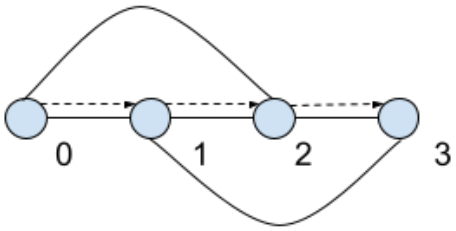



Figure 5.4: Graph of four nodes and five edges, the Hamiltonian path is indicated by the dashed arrows

Because node 0 and node 2 are connected, $x_1$ must lie beneath the line of sight between $x_0$ and $x_2$. The same can be concluded about $x_2$, with respect to the line of sight between $x_1$ and $x_3$. The lack of a connection between node 0 and node 3, implies that either $x_1$ or $x_2$ breaks the line of sight between $x_0$ and $x_3$. Thus the situation as described in Figure 5.4 leads to the sets of inequalities described in (5.6) and (5.7), one of which needs to hold for the time series $[x_0, x_1, x_2, x_3]$. If there is no series for which one of the inequality sets holds, it can be concluded that the graph cannot be a VG or a motif in a VG.

$$\begin{cases} x_1 < \frac{1}{2}x_0 + \frac{1}{2}x_2 \\ x_2 < \frac{1}{2}x_1 + \frac{1}{2}x_3 \\ x_1 \geq \frac{2}{3}x_0 + \frac{1}{3}x_3 \end{cases} \tag{5.6}$$

$$\begin{cases} x_1 < \frac{1}{2}x_0 + \frac{1}{2}x_2 \\ x_2 < \frac{1}{2}x_1 + \frac{1}{2}x_3 \\ x_2 \geq \frac{1}{3}x_0 + \frac{2}{3}x_3 \end{cases} \tag{5.7}$$

The case for inequality set (5.6) is examined first. The first and third term are combined to find a strict upper bound for $x_0$ in terms of $x_2$ and $x_3$.

$$\begin{aligned} \frac{2}{3}x_0 + \frac{1}{3}x_3 &< \frac{1}{2}x_0 + \frac{1}{2}x_2 \\ \frac{1}{6}x_0 &< \frac{1}{2}x_2 - \frac{1}{3}x_3 \\ x_0 &< 3x_2 - 2x_3 \end{aligned} \tag{5.8}$$

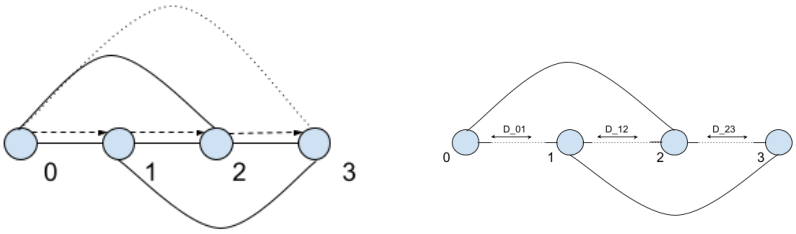
The expression for $x_1$ in the third inequality of (5.6) can be substituted in the second inequality, because if $x_1 \geq \frac{2}{3}x_0 + \frac{1}{3}x_3$, then substituting $\frac{2}{3}x_0 + \frac{1}{3}x_3$ for $x_1$ will never violate the upper bound for $x_2$. This yields the following:

$$\begin{aligned} x_2 &< \frac{1}{2}(\frac{2}{3}x_0 + \frac{1}{3}x_3) + \frac{1}{2}x_3 \\ x_2 &< \frac{1}{3}x_0 + \frac{2}{3}x_3 \end{aligned}$$

Some basic rearrangement to isolate $x_0$ leads to a strict lower bound:

$$x_0 > 3x_2 - 2x_3 \tag{5.9}$$

It is evident that Equations (5.8) and (5.9) cannot hold at the same time, and therefore the case in (5.6) is not possible. A similar proof can be performed for the inequalities in (5.7), meaning that a graph like




(a) Added link (b) Generalized case

Figure 5.5: Two additions to the previous graph: on the left a link is added, indicated by the dotted line. The addition of the link makes it so that it is no longer impossible for this graph to be a motif in a VG. On the right, the generalized case for this motif is presented, where there is room for other nodes to be in between the nodes that are part of the overlapping pairs

the one shown in Figure 5.4 cannot be a VG or a part of a VG. When an additional link is placed in the graph as shown in Figure 5.5, the motif does become possible.

A more general case is considered when there is distance between the nodes of the overlapping pairs. This is shown in Figure 5.5, where the distances $d_{01}$, $d_{12}$, and $d_{23}$ are indicated between each of the nodes. Although the nodes are still labeled 0 through 3, they are not necessarily adjacent on the natural Hamiltonian path. However, their order of occurrence is still the same i.e. the data point corresponding to node 0 occurs before the data point corresponding to node 1 and so on. Although not all conditions for the time series are known, due to there being an unknown number of nodes in between nodes 0 through 3, it is possible to investigate a part of the inequality set for contradictions. There are now three possibilities for the lack of a connection between node 0 and node 3. Either $x_1$ obstructs the line of sight, or $x_2$ does. The third option is that there is some $x_i$ lying in between that obstructs the line of sight. First, the cases for $x_1$ and $x_2$ obstructing the line of sight between $x_0$ and $x_3$ are considered in inequality sets (5.10) and (5.11), respectively.

$$\begin{cases} x_1 < x_0 + \dfrac{x_2 - x_0}{d_{01} + d_{12}} \cdot d_{01} \\ x_2 < x_1 + \dfrac{x_3 - x_1}{d_{12} + d_{23}} \cdot d_{12} \\ x_1 \geq x_0 + \dfrac{x_3 - x_0}{d_{01} + d_{12} + d_{23}} \cdot d_{01} \end{cases} \tag{5.10}$$

$$\begin{cases} x_1 < x_0 + \dfrac{x_2 - x_0}{d_{01} + d_{12}} \cdot d_{01} \\ x_2 < x_1 + \dfrac{x_3 - x_1}{d_{12} + d_{23}} \cdot d_{12} \\ x_2 \geq x_0 + \dfrac{x_3 - x_0}{d_{01} + d_{12} + d_{23}} \cdot d_{12} \end{cases} \tag{5.11}$$

Using a similar proof as for the case in (5.6), two bounds for $x_0$ are formulated in terms of $x_2$ and $x_3$ and distances $d_{01}$, $d_{12}$, and $d_{23}$ as a result of the inequalities in (5.10). They are an upper and a lower bound, shown in (5.12) and (5.13), respectively.

$$x_0 < \frac{\frac{d_{01}}{d_{01}+d_{12}}}{\frac{d_{01}}{d_{01}+d_{12}} - \frac{d_{01}}{d_{01}+d_{12}+d_{23}}} \cdot x_2 - \frac{\frac{d_{01}}{d_{01}+d_{12}+d_{23}}}{\frac{d_{01}}{d_{01}+d_{12}} - \frac{d_{01}}{d_{01}+d_{12}+d_{23}}} \cdot x_3 \tag{5.12}$$

$$\begin{aligned} x_0 > & \frac{1}{1 - \frac{d_{01}}{d_{01}+d_{12}+d_{23}} - \frac{d_{12}}{d_{12}+d_{23}} + \frac{d_{01}}{d_{01}+d_{12}+d_{23}} \cdot \frac{d_{12}}{d_{12}+d_{23}}} \cdot x_2 \\ & - \frac{\frac{d_{01}}{d_{01}+d_{12}+d_{23}} + \frac{d_{12}}{d_{12}+d_{23}} - \frac{d_{01}}{d_{01}+d_{12}+d_{23}} \cdot \frac{d_{12}}{d_{12}+d_{23}}}{1 - \frac{d_{01}}{d_{01}+d_{12}+d_{23}} - \frac{d_{12}}{d_{12}+d_{23}} + \frac{d_{01}}{d_{01}+d_{12}+d_{23}} \cdot \frac{d_{12}}{d_{12}+d_{23}}} \cdot x_3 \end{aligned} \tag{5.13}$$

The bounds are in the form of $x_0 < ax_2 - bx_3$ and $x_0 > cx_2 - dx_3$. Through rearrangement it is found that in this case, $a = c$ and $b = d$. Therefore, the bounds cannot hold and the possibility of $x_1$ obstructing the line of sight between $x_0$ and $x_3$ is disqualified. A similar proof can be conducted for $x_2$ with the inequality set (5.11). This leaves the possibility for some $x_i$, located between $x_0$ and $x_3$ that

obstructs the line of sight between $x_0$ and $x_3$. This $x_i$ can be located either between $x_0$ and $x_1$, $x_1$ and $x_2$, or $x_2$ and $x_3$. Each of these locations for $x_i$ would lead to a new inequality set. The implications for $x_0 < x_i < x_1$ and $x_2 < x_i < x_3$ are the same due to symmetry so the latter option will not be included in the proof. This leads to the following sets of inequalities:

$$\begin{cases} x_1 < x_0 + \dfrac{x_2 - x_0}{d_{01} + d_{12}} \cdot d_{01} \\ x_2 < x_1 + \dfrac{x_3 - x_1}{d_{12} + d_{23}} \cdot d_{12} \\ x_i < x_0 + \dfrac{x_2 - x_0}{d_{01} + d_{12}} \cdot d_{0i} \\ x_i \geq x_0 + \dfrac{x_3 - x_0}{d_{01} + d_{12} + d_{23}} \cdot d_{0i} \end{cases} \tag{5.14}$$

$$\begin{cases} x_1 < x_0 + \dfrac{x_2 - x_0}{d_{01} + d_{12}} \cdot d_{01} \\ x_2 < x_1 + \dfrac{x_3 - x_1}{d_{12} + d_{23}} \cdot d_{12} \\ x_i < x_0 + \dfrac{x_2 - x_0}{d_{01} + d_{12}} \cdot (d_{01} + d_{1i}) \\ x_i < x_1 + \dfrac{x_3 - x_1}{d_{12} + d_{23}} \cdot (d_{01} + d_{1i}) \\ x_i \geq x_0 + \dfrac{x_3 - x_0}{d_{01} + d_{12} + d_{23}} \cdot (d_{01} + d_{1i}) \end{cases} \tag{5.15}$$

Starting with the set in Eq.(5.14), the third and fourth line are contradictory. From these lines it can be deduced that:

$$x_0 + \frac{x_2 - x_0}{d_{01} + d_{12}} \cdot d_{0i} > x_i \geq x_0 + \frac{x_3 - x_0}{d_{01} + d_{12} + d_{23}} \cdot d_{0i}$$

$$\frac{x_2 - x_0}{d_{01} + d_{12}} \geq \frac{x_3 - x_0}{d_{01} + d_{12} + d_{23}}$$

$$x_2 - x_0 \geq \frac{x_3 - x_0}{d_{01} + d_{12} + d_{23}} \cdot (d_{01} + d_{12}) \tag{5.16}$$

The result in Eq. (5.16) can be substituted back to replace the term $(x_2 - x_0)$ the third line of Eq. (5.14) because it describes the upper bound for $x_i$ and using the expression in Eq. (5.16) still satisfies this bound as all distances are greater than 0. Substitution yields the following:

$$x_i < x_0 + \frac{\frac{x_3 - x_0}{d_{01} + d_{12} + d_{23}} \cdot (d_{01} + d_{12})}{d_{01} + d_{12}} \cdot d_{0i}$$

This simplifies to:

$$x_i < x_0 + \frac{x_3 - x_0}{d_{01} + d_{12} + d_{23}} \cdot d_{0i}$$

Which directly contradicts the fourth line of the inequality set shown in Eq. (5.14). A similar proof can be conducted using the third and fifth lines of the inequality set shown in Eq.(5.15). It can therefore be concluded that there can be no such sample $x_i$ anywhere in between $x_0$ and $x_3$ that would give rise to a pattern of links as shown in Fig. 5.5. This is an inadmissible motif.

## Mutual visibility

When two non-adjacent data points share a line of sight, there must at least be one other data point in between them that they also both share a line of sight with. This means that a VG cannot be a cycle graph, like the one shown in Figure 5.6. Such patterns also cannot emerge as a motif in a VG. This can be proven by examining the set of inequalities that arises from such a graph. The set of inequalities belonging to the graph in Figure 5.6 is shown in (5.17). From the connection between node 0 and node 3, it can be concluded that both $x_1$ and $x_2$ are below the line of sight between $x_0$ and $x_3$, hence the first two terms of (5.17). Furthermore, the absence of a connection between node 0 and node 2 implies that $x_1$ interrupts the line of sight between $x_0$ and $x_2$. The same holds for $x_2$ with respect to the line of sight between $x_1$ and $x_3$. This leads to the last two terms in (5.17).

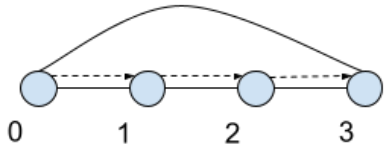



Figure 5.6: A cycle graph consisting of four nodes. The Hamiltonian path in this graph is indicated by the dashed arrows.

$$\begin{cases} x_1 < \frac{2}{3}x_0 + \frac{1}{3}x_3 \\ x_2 < \frac{1}{3}x_0 + \frac{2}{3}x_3 \\ x_1 \geq \frac{1}{2}x_0 + \frac{1}{2}x_2 \\ x_2 \geq \frac{1}{2}x_1 + \frac{1}{2}x_3 \end{cases} \tag{5.17}$$

The contradiction within this set of inequalities can quickly be found by substituting the fourth term expression for $x_2$ into the third term. Because the fourth term gives the lower bound for $x_2$, substituting that value into $x_1 \geq \frac{1}{2}x_0 + \frac{1}{2}x_2$ will not violate the lower bound for $x_1$. The substitution yields the following:

$$x_1 \geq \frac{1}{2}x_0 + \frac{1}{2}(\frac{1}{2}x_1 + \frac{1}{2}x_3)$$
$$x_1 \geq \frac{2}{3}x_0 + \frac{1}{3}x_3 \tag{5.18}$$

The expression in (5.18) is in direct conflict with the first term of (5.17), and therefore the graph cannot be a VG.

## 5.3. VG identification using MATLAB

Using the knowledge of VGs and their characteristics, it is possible to rule out whether a given graph is the product of the VG algorithm performed on some time series. Characteristics found in literature [1], [43], and the characteristics described in the previous section can be combined to make a set of criteria that a graph needs to obey for it to be a possible VG. So far, three criteria have been found that can help rule out whether a graph can be a VG. The first criterion is the existence of a Hamiltonian path, as all VGs have at least one such path. This immediately covers the criterion that the VG is connected. The second and third criteria are related to the inadmissible motifs from the previous section. If these are detected in the graph, it cannot be a VG.
The MATLAB script takes randomly generated graphs of size N and an amount of edges $e$. The edges will be placed randomly between the N nodes, so that a simple graph is formed using the 'graph' function in MATLAB. The graph is then tested for all three criteria and it becomes evident whether the graph can be a VG or not. For testing, a set of small graphs like the motifs in Figure 5.2 is used that goes up to size $n = 6$. All possible graphs with 5 or 6 nodes are generated and tested for being a VG. This is done in two ways, firstly by using the three mentioned criteria, secondly by generating the inequality set and finding a series of 6 samples that could fit the boundaries of the inequality set. Both methods give the same results, showing 25 VGs in all possible graphs of size $n = 5$ and 138 possible VGs in all graphs of $n = 6$. Because the complexity grows exponentially with size, it is not currently possible to examine larger graphs.

### The natural Hamiltonian path

Finding a Hamiltonian path on a graph is a computationally complex task, but there is a MATLAB function for it called 'hamiltonian'. This function takes three inputs, the adjacency matrix, a start node,

and a destination node. If a Hamiltonian path exists between start and destination, the function returns an array with the visited nodes in order. If no path exists, the function returns a string.
It is important that finding the presumed natural Hamiltonian path of the graph is the first step of the process, as this will induce a node order. When the graph is created, the nodes are numbered randomly. When a Hamiltonian path is found, the first node in the path is renamed node 1, the second node 2, and so on. This new node order is essential for the other criteria. It is possible that there are multiple Hamiltonian paths in the same graph, therefore the script follows the steps shown in Algorithm 3.

**Algorithm 3** Check for Hamiltonian path and VG criteria

```
for i = 1 : length(A) do
    for j = 1 : length(A) do
        Check for a Hamiltonian path between nodes i and j
        if Hamiltonian path exists then
            Reorder nodes to follow path order,
            Check mutual visibility criterion,
            Check the overlapping pair criterion
        else
            Return graph is not a VG
            if All criteria hold then
                Return graph can be a VG
            else
                Return graph is not a VG
            end if
```

The checks for the two criteria are done in separate scripts that are called by the script described above. The checks are described in the next paragraphs. If both criteria are met, the loop is broken and the script returns a TRUE value. If one or both of the criteria are not met for a certain Hamiltonian path, the script finds the next Hamiltonian path for which it can check the criteria, until all options have been explored.

## Overlapping pairs

The criterion for 'overlapping pairs' is related to the inadmissible motif described in Section 5.2. A MATLAB script is constructed to check for the occurrence of such motifs. When one is found, the graph in question is immediately dismissed as a possible VG. Let there be a graph of N = 10 nodes in which the links (3,7) and (5,9) exist. The adjacency matrix of this graph is shown in Matrix 5.19. If this graph is a visibility graph, the entry marked with an $x$ will always be 1. So in order to identify a graph as being the product of the NVG algorithm, the adjacency matrix of the graph under investigation can be used. If the matrix has an entry $A(i,j) = 0$ while also having entries $A(i,n) = 1$ and $A(m,j) = 1$ with $j < n < m$ and $n < m < i$, the graph in question cannot be the result of the NVG algorithm. In Figure 5.7, the graph that corresponds to the adjacency matrix is visualized. The link (3,7) corresponds with matrix entry $A(7,3) = 1$ and the link (5,9) corresponds with matrix entry $A(9,5) = 1$. There is an overlapping pair because in this case $n = 5$ and $m = 7$ while $i = 9$ and $j = 3$. Thus both $j < n < m$ and $n < m < i$ hold, and the entry $A(i,j) = 1$ is expected. The dotted line represents the missing link, the absence of which prohibits this graph from being the product of the NVG algorithm. A method of finding such an inadmissible pattern in a graph is described in 4.

**Algorithm 4** Steps to detect overlapping pairs

```
for i = 1 : length(A) do
    for j = 1 : i − 2 do
        if Link is found between non-neighboring nodes i and j then
            for b = j + 1 : i − 1 do
                Find nodes after node i that are connected to nodes between i and j
                if No connections are found then
                    Criterion holds
                else
                    Check the whether node j is also connected to the nodes that are
                    connected with the node b
                    if Connection does not exist then
                        Criterion does not hold
                        return
                    else
                        Criterion holds
                    end if
                end if
            end for
        end if
    end for
end for=0
```

$$\begin{bmatrix}
0 & & & & & & & & & \\
1 & 0 & & & & & & & & \\
 & 1 & 0 & & & & & & & \\
 & & 1 & 0 & & & & & & \\
 & & & 1 & 0 & & & & & \\
 & & & & 1 & 0 & & & & \\
 & & 1 & & & 1 & 0 & & & \\
 & & & & & & 1 & 0 & & \\
 & & x & & 1 & & & 1 & 0 & \\
 & & & & & & & & 1 & 0
\end{bmatrix} \tag{5.19}$$

Matrix 5.3.1: Adjacency matrix of a graph with some key links shown while omitting the rest for clarity. The entry marked with $x$ must be equal to 1 for this to be an adjacency matrix of a VG.

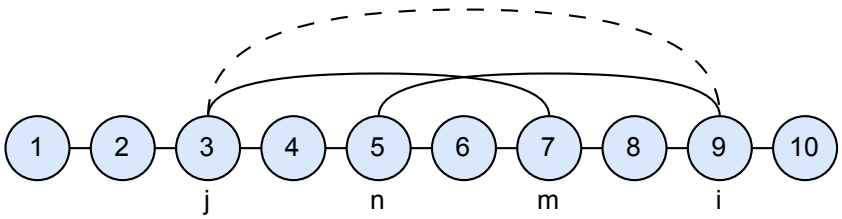



Figure 5.7: A case of overlapping pairs, placed as described in the adjacency matrix above. Similar to the adjacency matrix, only the links relevant to this example are shown. The absence of link (3,9), while links (3,7) and (5,9) are present, shows that this graph cannot be the result of the NVG algorithm.

## Mutual visibility

Similar to the previous criterion, the MATLAB script check for the occurrence of an inadmissible motif. When an edge is found that skips over more than one other node, so for example a connection between node 1 and node 5, it is checked whether nodes 1 and 5 are both connected to at least one of the intermediate nodes 2, 3 and 4. In the following example, there is a graph with N = 10 nodes. There

**Algorithm 5** Steps to detect mutual visibility between nodes

```
for i = 3 : length(A) do
    for j = 1 : i − 1 do
        if Non-neighboring nodes i and j are connected then
            Search for mutual neighbor in range b ∈ (j, i)
            if Mutual neighbor is not found then
                Criterion does not hold
                return
            else
                Criterion holds
            end if
        end if
    end for
end for
```

is a link between node 2 and node 8, as indicated in the adjacency matrix A. For the mutual visibility criterion to hold, at least one of the entries marked by a question mark have to be a 1. It is not possible for all the question mark entries to be 0. In terms of entries in the adjacency matrix shown in Matrix 5.20, it means that for an entry $A(i,j) = 1$, there must be at least one entry $A(i,n) = 1$ or $A(m,j) = 1$, with $j < n < i - 1$ and $j + 1 < m < i$. If one such entry is not present, the graph in question is not the result of the NVG algorithm. The algorithm that can identify this inadmissible pattern is shown in Algorithm 5.

$$\begin{bmatrix} 0 & & & & & & & & \\ 1 & 0 & & & & & & & \\ & 1 & 0 & & & & & & \\ & ? & 1 & 0 & & & & & \\ & ? & & 1 & 0 & & & & \\ & ? & & & 1 & 0 & & & \\ & ? & & & & 1 & 0 & & \\ & 1 & ? & ? & ? & ? & 1 & 0 & \\ & & & & & & & 1 & 0 \\ & & & & & & & & 1 & 0 \end{bmatrix} \tag{5.20}$$

Matrix 5.3.2: Adjacency matrix of a graph with some key links shown while omitting the rest for clarity. At least one of the entries marked with ? must be equal to 1 for this to be an adjacency matrix of a VG.

## 5.4. Conclusions

The central question to this chapter is whether, when presented with a graph, one can identify this graph as being the product of the NVG algorithm. Three indicators are found that can disqualify a graph from being an NVG. The first indicator, is the lack of a Hamiltonian path. The Hamiltonian path is the only characteristic that resembles an unavoidable motif. If no path is present that visits all the nodes exactly once, the graph under investigation cannot be an NVG. When there is such a path in the graph, the graph can be renumbered to have the node numbers coincide with the order of the Hamiltonian path. The nodes $n_1, n_2, ..., n_N$ now also coincide with samples $x_1, x_2, ..., x_N$ of a possible time series that could have been converted to the graph under investigation. It is in this form that the next two indicators can be used. Two motifs have been found to be inadmissible for an NVG. If either of these motifs is present anywhere in the graph, it cannot be the product of the NVG algorithm. These motifs are overlapping pairs and a lack of mutual visibility, described in Section 5.2. A mathematical description is provided of how these motifs were found to be inadmissible according to the algorithm that constructs the NVG. It is possible that there are more motifs or patterns that are not admissible, but they have not become apparent from the motifs for $n = 3, 4, 5$. For those lengths, the only motifs that do not appear in the generated NVGs are characterized by the overlapping pairs or lack of mutual visibility.
The chapter did not find a new feature that is an inherent or unique part of the NVG. No metric or pattern

has been found that can be used to definitively answer the question whether a given graph is the NVG of a time series, only patterns that disqualify a given graph from being an NVG. However, through the analysis of what makes the motifs inadmissible, some limitations in the adjacency matrix of an NVG are found. These limitations restrict the degree of freedom of how an NVG can be constructed, and thus also limit the amount of information that can be contained in an NVG. This concept is further explored in Chapter 6.

# 6

# Reversibility of the transformation

As stated in previous literature, the VG operation is non-reversible [17]. This is due to the loss of information that occurs when creating a VG from a time series. Reversibility of the algorithm could lead to new applications for the VG, for example in filtering [17]. To work towards such goals, it is important to distinguish how much information is lost, and how information is stored in the VG. The current chapter will examine the loss of information in the NVG algorithm and to what extent the transformation can be reversed.

## 6.1. Lossiness of transformation

In Chapter 3, it has been established how much information can be captured in a sampled time series. The number of possible graphs for size $N$ is also given and the amount of information that can be stored in the graph is deduced from this number. However, not all realizations of a graph of size $N$ are VGs, as was shown in Section (5.2). Each time series does not map to a unique VG. When using infinitely precise sample values, there is an infinite amount of time series that map to each VG realization which leads to a loss of information. When the data points in a time series are chosen from a finite pool of values, an alphabet $\mathcal{X}$, this limits the amount of time series that map to a particular VG. However, the amount of VGs of size $N$ will always remain smaller than the amount of time series of length $N$. Take for example a time series $x_t$ of length 2, where each data point $x_i \epsilon \mathcal{X}$ and $\mathcal{X} = [0, 1]$. There are 4 possible realizations of $t$ but they all result in the same VG; two connected nodes. The length of $t$ can be extended, resulting in more possible realizations. If $t$ is extended to length $N = 3$, there are two possible VGs; a line graph of length 3 and a triangle shape. To illustrate how quickly the amount of possible VGs increases as a function of both $N$ and the size of $\mathcal{X}$ (p), a MATLAB script was used to find the values for $N = 2$ through $N = 8$ and $p = 2$ through $p = 8$. The pseudocode of the MATLAB script is shown in Algorithm 6. By generating all possible sequences for a given length $N$ and sample depth $p$, it becomes too demanding very quickly for higher values. The results of this script are shown in Table 6.1 and Figure 6.1. For a constant length N and increasing sample depth $p$, the number of possible VGs grows towards a limit. For $N = 2, 3, 4$ this limit is reached immediately at 1,2 and 6 respectively. For $n = 5, 6$ the limits are found at $p = 4$ and $p = 6$ respectively. For $N > 6$, the limit is found at $p > 8$ and is therefore not visible in these results.

For $p = 2$, each increase in $N$ almost exactly doubles the amount of possible VGs. This relation starts at $N > 4$ ànd continues beyond $N = 8$. Overall it is evident that the amount of possible VGs does not increase as quickly as the number of possible time series for the values of $n$ and $p$. Furthermore, the increase in possible VGs as a function of $N$ for constant $p$ seems exponential, whereas the increase in possible VGs as a function of $p$ for constant $N$ grows towards a certain limit. The data are very limited due to the computational complexity of the problem. It is therefore not feasible to estimate a trend using these data in order to reach conclusions about the relation between sampling depth and the number of possible VGs. Furthermore, it is not clear to what extent the results for these integer series are indicative of the behaviour of continuous series. In previous work on VGs from integer series, it was concluded that certain graph metrics only converged to the continuous case for $p > 100$ [45].

**Algorithm 6** Algorithm to find all possible VGs for given length $N$ and sampling depth $p$

```
for n = 2 : N do
    for p = 2 : P do
        Generate all possible integer sequences for the values of n and p
        Construct NVG of the first sequence and use it as the first entry of NVG collection
        for i = 2 : number of sequences do
            Construct NVG_i of sequence i
            if NVG_i has not yet occurred in NVG collection then
                Add NVG_i to NVG collection
            end if
        end for
        Store length of NVG collection in location (n, p)
    end for
    Empty NVG collection
end for
```

Table 6.1: The number of unique possible VGs as a result of time series with length $N$ and entries chosen from an alphabet $\mathcal{X}$, where $\mathcal{X}$ = [1,2,..,p-1,p]

| N \ p | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|
| 2 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 3 | 2 | 2 | 2 | 2 | 2 | 2 | 2 |
| 4 | 6 | 6 | 6 | 6 | 6 | 6 | 6 |
| 5 | 12 | 24 | 25 | 25 | 25 | 25 | 25 |
| 6 | 24 | 82 | 124 | 134 | 138 | 138 | 138 |
| 7 | 48 | 250 | 542 | 759 | 889 | 925 | 945 |
| 8 | 99 | 730 | 2112 | 3774 | 5454 | 6667 | 7371 |

To determine the measure of information that is contained in the VGs, Shannon's definition of entropy in Eq. (3.11) is used. The probability of occurrence for each VG is calculated, and used to calculate the entropy. The steps in the MATLAB script that are taken to find the entropy at the given length and sample depths, are described in Algorithm 7.

The small graphs that are generated are similar to the motifs discussed in Chapter 5. There is a finite number of NVG motifs that can be generated for given length $N$ and sampling depth $p$. Not all motifs are equally likely to occur as a result of a random series of that length and sampling depth. As an example, the probability distributions of the motifs for $N = 2$ through $N = 6$ at $p = 6$ are shown in Figure A.1. These values are chosen because the maximum amount of unique VGs is reached for those lengths at $p = 6$ as is evident from Table 6.1. These probability distributions are used to calculate the entropy of the graphs that are constructed from each set of sequences. For example, only one graph is constructed from the set of 36 sequences with $N = 2$ and $p = 6$. This is shown in Figure A.1(a). For all the cases where $N = 2$, the entropy is equal to 0 because there is only one VG possible. From the set of $6^6$ sequences that can be formed with $N = 6$ and $p = 6$, 138 unique VGs can be constructed. Not all VGs are equally likely, as is shown in Figure A.1(e). Overall, it is clear that information is lost with the operation and the amount of information lost depends on the length of the signal as well as the sampling depth. The amount of lost information for each case is given in bits in Table 6.2.

For each increase in $p$, more information is lost because the number of VGs is not increasing as quickly as the number of combinations of the time series. For increases in $n$ more information is lost for sampling depths $p > 2$. However, at $p = 2$ the loss of information converges to 1.5 bits as the length of the series increases. At $p = 8$ and $N = 8$, the difference in entropy is largest with 14.4 out of 24 bits being lost in the transformation.

**Algorithm 7** Algorithm to find entropy in generated VGs

**for** $n = 2 : N$ **do**
  **for** $p = 2 : P$ **do**
    Generate all possible integer sequences for the values of $n$ and $p$
    Construct NVG of the first sequence and use it as the first entry of NVG collection
    Initialize counter to keep track of occurrences of identical NVGs
    **for** $i = 2$ :number of sequences **do**
      Construct $NVG\_i$ of sequence $i$
      **if** $NVG\_i$ has not yet occurred in NVG collection **then**
        Add $NVG\_i$ to NVG collection
      **else**
        Increase occurrence counter
      **end if**
    **end for**
    Store length of NVG collection in location $(n, p)$
  **end for**
  Determine probability distribution based on occurrence counter
  Empty NVG collection
**end for**
Calculate Shannon entropy ($H(X) = -\sum_{x \epsilon X} p(x) log_2(p(x))$)[1] using probability distribution

Table 6.2: The absolute difference ($H_{time\ series}$ - $H_{NVG}$) in the amount of information (bits) that is captured in the time series and the amount of information that is captured by the VG for values of N and p ranging from 2 to 8.

| $N$ \ $p$ | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|
| 2 | 2 | 3.1699 | 4 | 4.6439 | 5.1699 | 5.6147 | 6 |
| 3 | 2.0456 | 3.7798 | 5.0113 | 5.9736 | 6.7599 | 7.4259 | 8.0028 |
| 4 | 1.7256 | 3.9732 | 5.6244 | 6.9003 | 7.9485 | 8.8385 | 9.6090 |
| 5 | 1.6128 | 4.0291 | 6.0411 | 7.6247 | 8.9276 | 10.0343 | 10.9953 |
| 6 | 1.5564 | 4.0301 | 6.3273 | 8.2012 | 9.7555 | 11.0768 | 12.2273 |
| 7 | 1.5282 | 4.0419 | 6.5637 | 8.7058 | 10.5020 | 12.0339 | 13.3704 |
| 8 | 1.5141 | 4.0678 | 6.7694 | 9.1495 | 11.1749 | 12.9122 | 14.4331 |

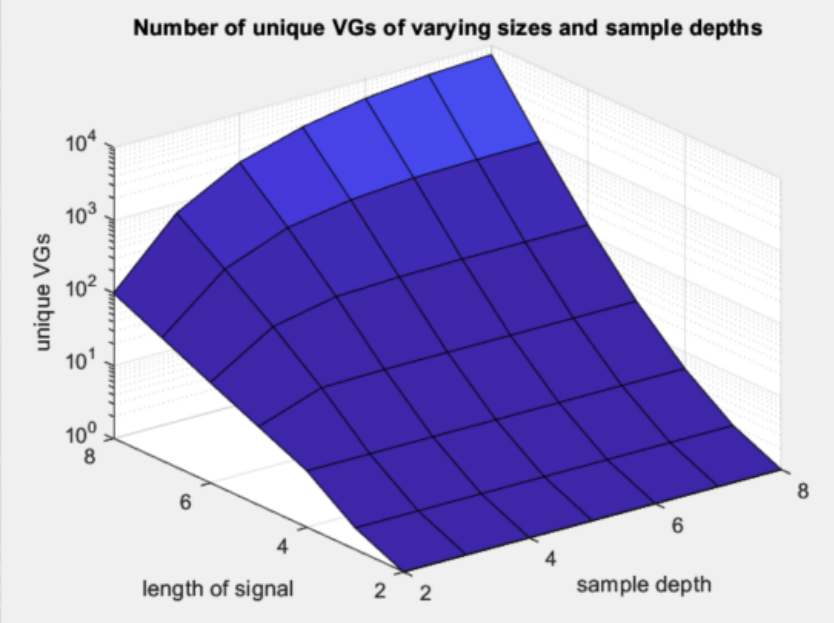



Figure 6.1: The number of unique possible VGs as a result of time series with length $N$ and entries chosen from an alphabet $\mathcal{X}$, where $\mathcal{X}$ = [1,2,..,p-1,p]. This is denoted on the axis labeled 'sample depth'.

Table 6.3: The absolute difference ($H_{\text{time series}}$ - $H_{\text{WNVG}}$) in the amount of information (bits) that is captured in the time series and the amount of information that is captured by the VG for values of N and p ranging from 2 to 7.

| $N$ \ $p$ | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|
| 2 | 0.5000 | 1.5850 | 2.4387 | 3.219 | 3.6866 | 4.1659 |
| 3 | 0.2500 | 1.4866 | 2.5682 | 3.5213 | 4.3282 | 5.0278 |
| 4 | 0.1250 | 1.4024 | 2.6378 | 3.7738 | 4.7739 | 5.6642 |
| 5 | 0.0625 | 1.3580 | 2.7043 | 3.9939 | 5.1547 | 6.2084 |
| 6 | 0.0313 | 1.3524 | 2.7855 | 4.2001 | 5.5057 | 6.7059 |
| 7 | 0.0156 | 1.3782 | 2.8937 | 4.4193 | 5.8553 | 7.1892 |

## Information in weighted VG

The results of Table 6.2 indicate the information lost when using the NVG algorithm as it was first proposed. However, literature has shown more robust results for the extraction of information from VGs when the algorithm is altered to include link weights [20], [22]. To put this to the test, a simple link weight addition is proposed. The link weights will indicate which is the largest of the samples corresponding to the connected nodes. The links will only take four possible values $w_{ij} \in [0, 1, 2, 3]$. An entry of 0 indicates no link, the weight of 1 indicates a link from a high sample to a low sample, the weight of 2 indicates that the samples are equal, and the weight of 3 indicates a link from a low sample to a high sample. For example, if node $a$ is connected to node $b$ and $x_a < x_b$, then entry $W(a, b) = 3$ in the weighted adjacency matrix $W$. This means that the entry $W(b, a) = 1$ by definition. Instead of a binary adjacency matrix, each entry now has 4 options and thus up to 2 bits of information. One could thus expect the information retained in the graph to be doubled. Using Algorithm 7 with the weighted NVG (WNVG) function, shows that the doubling of information is not achieved. The results of the lost entropy are shown in Table 6.3.

It is clear that the WNVG includes more information about the time series. The losses are smaller than those for the NVG. For a binary series ($p = 2$), there is only a small of information at for $N > 2$. This is not surprising, since the weights indicate which samples are smaller and which are larger. When there are only two possible sample values, this is enough information to almost fully describe the time series. However, for a length of $N = 7$ this method uses 10 entries of 4 bits each to represent a series of 7 ones and zeros. The time series is not fully described partly because there is no way to differentiate between the constant time series $x_t = [1, \ldots, 1]$, $x_t = [0, \ldots, 0]$ and so on. For the situation where $p = 2$,

this uncertainty represents the only loss of information. As the likelihood of a constant time series decreases with larger $n$, the amount of entropy lost also decreases.

## 6.2. Attempt at the reverse operation

The NVG algorithm in its original form clearly loses information. Many time series can lead to the construction of the same VG, making it impossible to construct the original time series from its VG. If the original time series is not known, the VG by itself does not contain enough information to reconstruct it. The inequality set poses a set of limitations that the time series has to comply with. From these limitations, a set of time series can be identified as the original time series from which the VG was constructed. This is demonstrated in a MATLAB script in which a random series of length $n$ and sampling depth $p$ and its NVG are constructed. The inequality set is then generated as per the method detailed in Section 5.1. The inequality set is tested against all $p^N$ possible time series with length $n$ and sampling depth $p$ to find which series could be the original to the graph under investigation.
When testing for a time series with length $n = 8$ and $p = 8$, the scope of the number of possibilities becomes clear. The series $x_t = [6, 3, 8, 1, 4, 4, 7, 7]$ was randomly generated, along with its NVG. The graph ($NVG_x$) is shown in Figure 6.2. The goal is to find an estimate of $x_t$ using only the information in $G_x$. The inequality set is generated and 27 inequalities are found. These inequalities define the space in which the solution may lie. It forms a set of boundaries for each entry of the estimate $x'_t$. When generating all $p^N$ time series, it is found that 9819 of them are within the set boundaries. In the current form, there is no method of discriminating between all these options to find the original time series $x_t$.

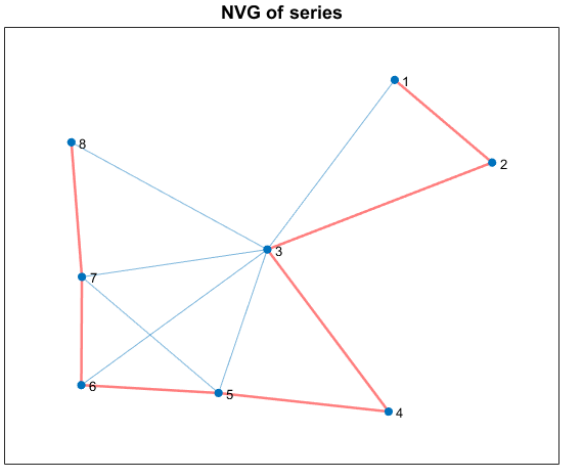



Figure 6.2: NVG associated with time series $x_t$

If the same series is transformed into a WNVG, as detailed in Section 6.1, the number of eligible time series is significantly reduced. From the 9819 valid time series in the unweighted, to 981 valid time series when the WNVG is used. This means that it is still not possible to reverse the operation, but using only a minimal addition of information the range of possible time series corresponding to a graph has been reduced tenfold. The number is not always reduced by exactly the same factor, but especially for low values of $p$ the reconstruction of the weighted graph works better than the unweighted version. As expected from the results in Table 6.3, for $p = 2$ the time series can be perfectly reconstructed barring the situations where all samples have the same value.

### The issue with link weights

It has been shown that the introduction of link weights allows for more information to be stored in the NVG. A theoretical threshold of information stored in the entries adjacency matrix to ensure complete description of the time series is possible. As long as the number of bits described in the adjacency matrix is larger than the number of bits in the time series, one could think an inverse operation is possible. The introduction of link weights adds more information to the adjacency matrix. As presented in Section

2.2, multiple methods of implementing link weights have been presented. These mostly revolve around the relations between samples in the form of height difference, Euclidean distance, or angle of the line of sight. The issue with measures like these with respect to information retention is twofold. Firstly, there is redundancy in the link weight information. Imagine a scenario where the link weight is the height difference of samples, and three consecutive samples are $x_t = [x_1, x_2, x_3] = [5, 0, 5]$. The links $W(1,2) = -5$ and $W(2,3) = 5$ already imply that link $W(1,3) = 0$. Although more information is now available about the time series, the effects on information retention of replacing the Boolean values of a binary adjacency matrix with infinite precision values, is limited. Secondly, taking another look at the example for $x_t = [x_1, x_2, x_3] = [5, 0, 5]$, it is clear that any version of $x_t$ that is wholly shifted in the vertical direction will produce the same adjacency matrix. The VG process being invariant under affine transformations also applies to weighted versions, even when using angles or Euclidean distance.
The issue can be circumvented by using the actual values of the samples in the link weights. For example, for a series of 8-bit samples, a link $l_{a.b}$ between node $a$ and node $b$ could be weighted such that $l^w_{a,b} = [x_a, x_b]$. So that the weighted link is a 16-bit word that consists of the two concatenated sample values $x_a$ and $x_b$. In this case, all information of the time series is captured in the graph. The use of such a graph is limited however, because the link weights are meaningless for metrics such as path length. The method does allow for an inverse transformation back to the time domain, but not in way that utilizes the concepts of the VG in a meaningful way.

## 6.3. Conclusions

All literature regarding the VG describe it as a 'lossy' process, in which some of the information in the time series is lost. However, a measure of the amount of lost information has not been presented. In this chapter, the Shannon definition of entropy has been used to attempt to quantify the information that is lost. It has been established that more information lost in the NVG transformation for time series with a higher precision. From the way an unweighted NVG is constructed, with $\frac{(N-1)(N-2)}{2}$ binary entries, it is clear that the graph cannot completely capture the information in the time series. However, because not all graphs can occur as VGs and some VGs are more likely to occur then others, even more entropy is lost when using the Shannon definition.
A trend that is seen in Figure 6.1, is that there is a practical maximum of unique VGs for a series with length $n$ and sampling precision $p$. The number of VGs that are found seem to grow asymptotically to some limit. Due to computational limitations, these limits are too high to assess, even for very short time series. The existence of a practical maximum of unique VGs of size $n$ being generated from an infinite amount of possible time series of length $n$ is not surprising, given how the binary adjacency matrix has a clear limit. However, what determines this practical maximum has not been found in this chapter.
Assigning weights based on sample height difference to the links alleviates the loss of information to some extent, but still fails to completely represent the information in the time series. As such, it is not possible to reverse the operation to regain the time series from a VG with the methods presented in this chapter.

# 7

# Discussion & Conclusions

This chapter recounts the findings of the thesis in chronological order. The knowledge gap in the current literature is framed, and it is assessed to what extent the findings of the thesis provide new information regarding the concept of the VG.

## 7.1. Discussion

As described in Chapter 2, there is a variety of VG algorithms with different conditions for visibility. The contexts in which the concept of the VG is used also vary widely from seismic activity, to stock market analysis, to diagnoses from biological data sets. In each of these researches, the VG serves as a way to convert the time series data to a new domain in which some form of analysis is done on the graph to find results. The VG is used as a means to an end, but the transformation itself is not always the subject of the research. The intrinsic characteristics of the HVG have been explored but the same cannot be said for the other VG implementations. The family of graphs that are VGs have been found to share some characteristics, but a formal definition or set of conditions for a graph to be a VG has not been put forward. Furthermore, the interpretation of the VG as a mathematical operation or transformation of information, is still lacking.

In Chapter 4 it is attempted to find out more about the intrinsic characteristics of the NVG in the form of degree metrics. It soon became clear that the same method that was used for the HVG would not suffice. A restriction had to be applied to the time series that is used for the construction of the NVG. The infinite uniformly distributed time series $x_t$ was bounded by the assumption that $x_t \in [1, 2, 3, ...p]$, i.e. the series becomes an integer series. The introduction of a limited number of sample values greatly simplifies the analysis of the degree metrics, while not being an unreasonable assumption. Any digitally sampled real-life data set will have a limited number of sample values depending on the precision of the digital measurement instrument. Using the simplification, it is possible to predict the average degree of an NVG given the length of the time series, the number of possible sample values and the probability distribution.

In order to gain more insight in the class of graphs that are VGs, Chapter 5 looks into the characteristics that define a VG. This process was started by inspection of a set of generated VGs for very short time series. It was evident that not all possible graphs of size $n$ appeared as a possible NVG of a time series with length $n$. Rather than finding a feature that all VGs have in common, two motifs were identified in the graphs that could not be a VG of a time series. One or both of these motifs are present in each of the non-VG graphs. By converting all graphs into inequality sets, it was shown that the inequality sets for the non-VG graphs were indeed contradictory. As such, two characteristics are found that can disqualify a graph from being a VG from analyzing the adjacency matrix. The way these restrictions present themselves in the adjacency matrix is interesting, as the entries at the intersection of rows and columns plays a part in both. However, no new characteristic common to all the VGs is identified to be added to a description of the VGs as a class of graphs.

The retention of information when transforming a time series to a VG has not been thoroughly investigated in literature. In Chapter 6, the widely used Shannon definition of entropy is used to try to quantify the loss of information during the transformation. Because of the relatively low number of unique VGs

for a time series of large length, the information that is retained in the VG is low by that definition. The time series used in this thesis were limited to integer series length $n = 8$ and a maximum sample value of $p = 8$. The limitations are due to the computational complexity of generating more than $8^8$ possible time series and identifying the unique VGs for each of them. The introduction of link weights alleviates the issue of information loss slightly with more unique VG configurations being possible. However, even with various link weight definitions, it is impossible to completely capture all the information in the VG. It is shown in literature that the VG can be used to make conclusions about time series, in some cases being more accurate than conventional time series analysis methods. It seems that the classical outlook of information theory that is put forward in this thesis is not adequate to describe how the VG captures structural information of the time series.
The thesis has taken some previously unexplored pathways into investigating the concept of the VG, with varying success. A definition for many metrics of the NVG are still lacking, there is no formal set of rules that can be used to identify a graph as being part of the class of VGs and the nature of the information that is stored in the VG is unclear.

## 7.2. Conclusions

It is found that, using the assumption of a limited integer series, metrics for the NVG can be expressed in an exact manner. This thesis has defined an expression for the average degree in the NVG constructed from a random limited integer time series. The findings are verified by comparing the outcome of the exact expression to an average of many NVG realizations.
The characteristics that make a graph part of the VG class are explored. Two motifs are found which, when present in a graph, disqualify that graph from being a VG. The proof that these motifs are indeed impossible in a VG is found in the inequality set that describes that graph. Using the two motifs, a MATLAB script is written that successfully returns whether a given graph is a VG or not. However, no conclusion is reached on a formal definition that describes all graphs in the VG class.
According to the Shannon definition of entropy, almost all information in a time series is lost when transforming it to a VG. This is due to there being a limited number of VGs of length $n$, while there are potentially infinite time series $x_t$ of length $n$ depending on how the sample values of $x_t$ are chosen.

## 7.3. Recommendations

Future work that might explore the concept of the VG further could focus on some aspects that were not fully worked out in this thesis. For example, the method of using a limited integer series for the analysis of the NVG can yield more exact results for different metrics. That would allow for a more extensive mathematical description of the NVG, as has been done for the HVG. From the expression of the degree distribution for the HVG, an expression for the clustering coefficient distribution was formulated by Luque et. al [5]. A similar thread for the NVG is worth exploring. When more definitions for the NVG of random signals are found, it could ostensibly be used to distinguish between stochastic and chaotic signals using different methods than the already established analysis of the degree distribution [33].
The patterns that are found in the adjacency matrices of NVGs in Chapter 5 were noted in this thesis, but not investigated further. This could be a starting point for a new research that analyses whether the a given adjacency matrix is the result of a VG operation on a time series and perhaps give insight into the class of graphs that are the VGs in that manner.

# 8

# Further Exploration

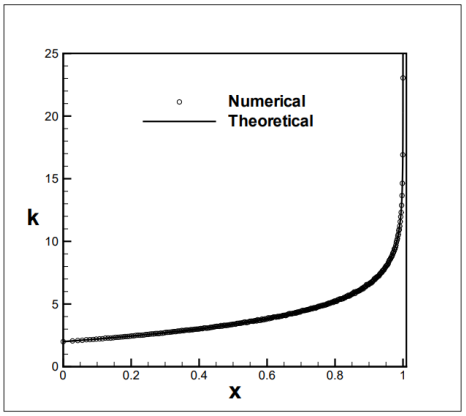
In addition to the recommendations given in the previous chapter, there are several other avenues to explore that were found during the making of this thesis. This chapter presents the basis of four topics that could be further explored in future works.

## 8.1. Sample height and (neighbor) degree correlation

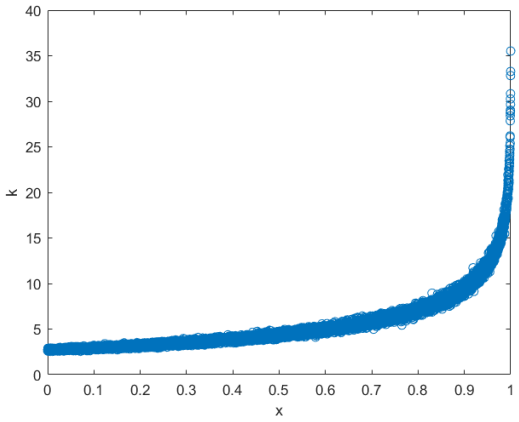
An exact relation between the height of a certain sample and the average degree of the corresponding node has been formulated for the HVG [5]. This is derived from the degree distribution of the HVG that is given in Chapter 6. When investigating the conditional probability of a node having degree $k$ for a given sample $x$ ($P_{HVG}(k|_x)$), the result can be described exactly. To find the average degree $\bar{k}$ of a node for a given sample $x$, it simply holds that;

$$\bar{k}(x) = \sum_{k=2}^{\infty} k \cdot P_{HVG}(k|_x) \tag{8.1}$$

The expression in Eq. (8.1) is verified by generating a random time series of length $N = 10^6$, constructing the HVG and taking the average of degrees of all the nodes that correspond to the samples with equal value in the time series. The results form this numerical approach suggest that the proposed expression is accurate, as shown in Figure 8.1(a).Since there is no singular expression for the degree distribution $P_{NVG}(k)$, the theoretical method of finding an average degree for a given sample $x$ is not yet possible. However, the numerical approach can be used to investigate whether a similar trend occurs in the degree and sample height correlation for the NVG.
A time series is created of length $N = 10^6$, with uniformly distributed random samples between 0 and 1 at 0.0001 intervals. The NVG is constructed for this time series and the degree sequence is extracted. In an Nx2 array, the degree sequence is stored together with the time series. For each unique sample height in the time series, the average corresponding degree is calculated. The findings are shown in Figure 8.1(b).
The relation between sample height and nodal degree is similar for the HVG and NVG. For both VGs, only the highest samples have the tendency to be hub nodes. From these results, it can be inferred that altering the higher samples of the time series has a larger effect on the topology of the graph when compared to altering the lower samples. The average nodal degree tends to an exponential curve in Figure 8.1(b), just as it does in Figure 8.1(a). However, the values in Figure 8.1(b) are more dispersed along the trend line. This could be due to the higher influence of stochastic events that is alluded to in Chapter 4, when working towards a description for $P_{NVG}(k)$. The similarity between $\sum_{k=2}^{\infty} k \cdot P_{HVG}(k|_x)$ and $\sum_{k=2}^{\infty} k \cdot P_{NVG}(k|_x)$ is indicative of some similarity between $P_{HVG}(k)$ and $P_{NVG}(k)$. This could be another entry point for the research towards the degree distribution of the NVG.
It is clear that higher sample values tend to lead to higher degree nodes. However, a high sample value is likely to block lines of sight from its neighbors to other samples. It can therefore be interesting to assess how the presence of high samples affects the average degree in the neighborhood of the high sample. This can also be done using MATLAB and leads to the distribution shown in Figure 8.2. There




(a) Relation between degree and sample height for HVG




(b) Relation between degree and sample height for NVG

Figure 8.1: Left: The relation $K(x)$ between the height of a sample and the average degree of the corresponding sample in the HVG constructed from a random time series. On the x-axis are all unique values that occur in the time series. On the y-axis is the average nodal degree for nodes that correspond with the samples of the height on the x-axis. In an HVG, the hub nodes are the nodes that correspond to high sample values and the average degree for a node that corresponds to a sample with value $x$ can be perfectly described using theory [5]. Right: The relation $K(x)$ between the height of a sample and the average degree of the corresponding sample in the NVG constructed from a random time series of length $N = 10^5$. On the x-axis are all unique values that occur in the time series. On the y-axis is the average nodal degree for nodes that correspond with the samples of the height on the x-axis.

is a large spread in the results, however it is clear that samples in the neighborhood of low samples on average have a degree $k > 4$. The size of the neighborhood is arbitrarily chosen as 8, a smaller neighborhood amplifies the effect that is seen in Figure 8.2 while a larger neighborhood dilutes it.

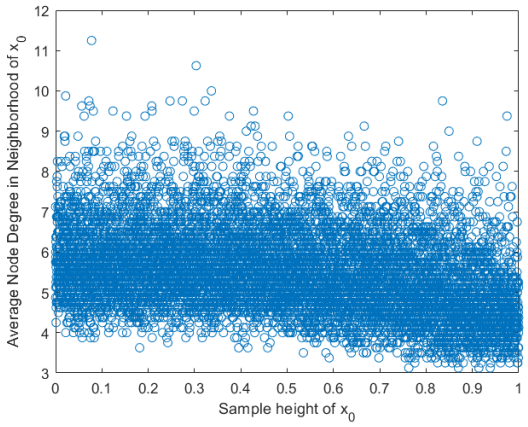



Figure 8.2: The relation between the height of a sample $x_0$ and the average degree of the neighborhood $[x_{-4}...x_4]$ (excluding $x_0$) in the NVG constructed from a random time series of length $N = 10^5$. On the x-axis are all unique values that occur in the time series. On the y-axis is the average nodal degree for nodes that correspond with the samples in the neighborhood of $x_0$, the height of which is shown on the x-axis.

## 8.2. Time domain assortativity

The concept of assortativity in a graph is the measure of how likely nodes with a certain degree are to connect to other nodes of a similar degree. If a graph has a positive assortativity, nodes of similar degree are likely to be connected to each other. For negative assortativity, nodes of high degree are likely to connect to nodes with low degree and vice versa. Through the NVG this concept can be extended to use the time domain data. The likelihood of data entries of a certain height seeing other data entries

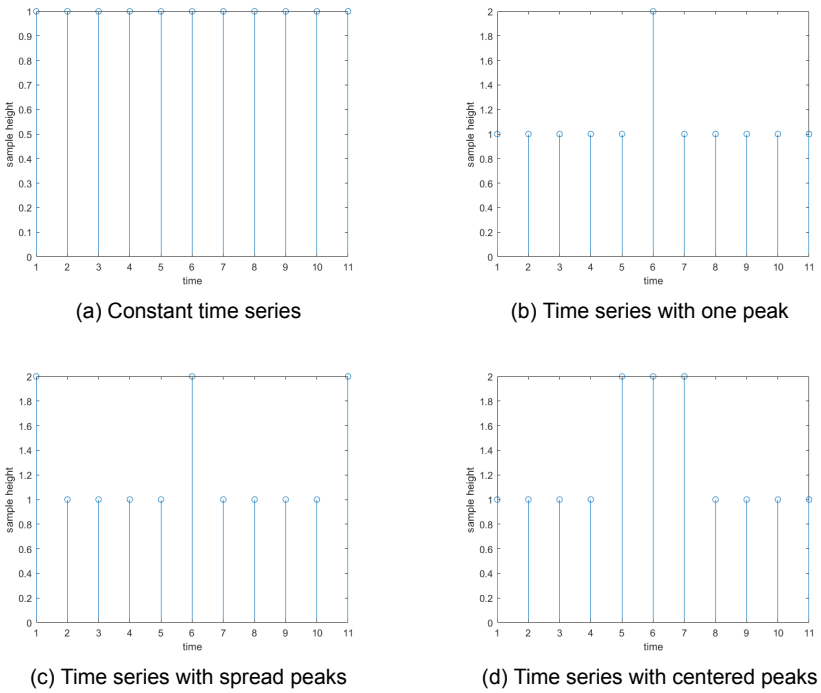
Table 8.1: Different assortativity values for graph and time domain of the four time series as presented in Figure 8.3

| | Degree assortativity | Time domain assortativity |
|---|---|---|
| subfigure (a) | -0.111 | NaN* |
| subfigure (b) | -0.4161 | -0.3846 |
| subfigure (c) | -0.3273 | -0.3714 |
| subfigure (d) | -0.5787 | -0.0667 |

of similar height would be the time domain interpretation of assortativity. This metric might be used to gain more insight in the positions of relative peaks in the time series. When high sample values are generally surrounded by relatively low sample values, this will lead to negative assortativity. Both the graph assortativity and the time domain assortativity will be negative in this case. Some variations of this scenario are depicted in Figure 8.3. In subfigure (a) there is a constant time series, the NVG of which is always a line graph. The degree assortativity for such a graph is given by Equation 8.2 [47].

$$\rho_D = -\frac{1}{N-2} \tag{8.2}$$

The degree assortativity for a line graph of length $N = 11$ should thus be equal to $-\frac{1}{9} = -0.11\bar{1}$. This is consistent with the value that is found using a MATLAB script. The found values for both degree assortativity and the time domain assortativity are shown in Table 8.1. The time domain assortativity for the line graph is undefined for MATLAB, it is trying to find the correlation between two perfectly identical sequences. There are discrepancies between the graph and time domain metrics, especially in the final scenario where peaks are clustered together. Further research could be devoted to how this new metrics can analyse the placement of high valued samples in a time series.




(a) Constant time series

(b) Time series with one peak

(c) Time series with spread peaks

(d) Time series with centered peaks

Figure 8.3: Four time series with a varying amount of peaks and different positioning of peaks, leading to different degree assortativity and time domain assortativity.

The different implementations of the VG also give varying results for time domain assortativity. The findings for the assortativity of VGs of a random time series of length $N = 1000$ are given in Table 8.2. It stands out that all values are positive for each VG implementation. This means that, on average, high degree nodes see other high degree nodes and low degree nodes see other low degree nodes.

Table 8.2: Different assortativity values for graph and time domain for graphs constructed from a random time series of length N = 1000 using varying VG implementations

| | Degree assortativity | Time domain assortativity |
|---|---|---|
| NVG | 0.1787 | 0.2122 |
| HVG | 0.1565 | 0.3112 |
| NLPVG | 0.2174 | 0.2038 |
| HLPVG | 0.2621 | 0.3116 |
| DPVG | 0.1461 | 0.4228 |

One might expect that the hubs of the graph, which tend to correspond with peaks in a random time series, can see many lower degree nodes around them. This is the case in the examples in Figure 8.3, yielding negative degree assortativity. The question remains how the assortativity can be positive in a random time series. For further investigation, a distinction is made between the degree assortativity of the low degrees and the high degrees. For each category, the assortativity is determined as shown in Algorithm 8.

**Algorithm 8** Assortativity difference between low and high degrees

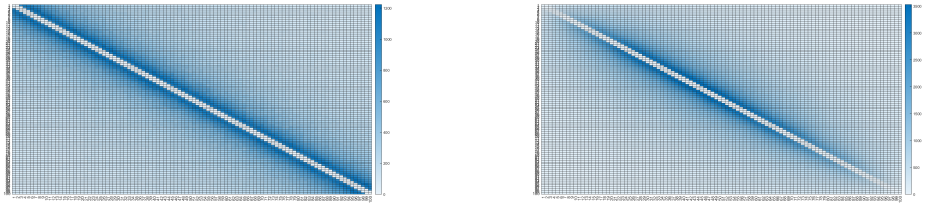
Construct NVG of a random time series
Generate link list (L) with rows $[start\_node, end\_node]$
Replace node numbers in L with degree of corresponding node such that rows are $[degree(start\_node), degree(end\_node)]$
Sort values in L from low to high in column 1 (or column 2, this is arbitrary) to obtain $L\_sorted$
Divide $L\_sorted$ in a lower half and upper half
Find assortativity of both halves

It is found that the degree assortativity in the lower half is close to 0, while the assortativity in the upper half is generally higher than the degree assortativity of the graph as a whole. This means that while low degree nodes have no clear preference for low or high degree nodes, the nodes with high degree are more likely to be connected with other nodes of high degree. When viewing this from the perspective of the time domain, and keeping in mind that high degree nodes are likely to correspond to high samples, this phenomenon can be explained. The low degree nodes are more likely to correspond to low value samples. These samples only have a line of sight with a few samples in their direct vicinity. Because the time series is completely random, the samples in the vicinity could be higher samples (corresponding to higher degree nodes) or lower samples (corresponding to lower degree nodes) and thus no clear preference can be discerned. On the other hand, high samples tend to have further reaching visibility and thus are not dependent on their direct neighbors as much as low samples are. For a high sample $x_h$ to see another sample $x_t$ at a distance, all samples $x_b$ in between have to be below the line of sight that is shared by $x_h$ and $x_t$. This bound is very strict if $x_t$ is a low value, and less strict if $x_t$ is a high value. Therefore it is more likely for high value samples to see other high values samples at a distance, causing a positive degree assortativity in the graph. This phenomenon is even more prominent for the time domain assortativity, hence the higher positive values. Limited penetration seems to lessen the gap between graph and time domain assortativity, while the dual perspective graph widens the discrepancy. Further research can be conducted to find the most suitable VG implementation to find the most relevant information from the new metric of time domain assortativity.

## 8.3. Effects of rewiring

The VG is a representation of the time series in the graph domain. Alterations in the time domain will lead to a different VG. However, it is unclear how alterations in the VG would change the set of time series that it represents. It is reasonable to assume that a single change in the graph will have an impact on the inequality set of the graph. This means the space in which the represented time series lie is changed. This could be better explored if there is a method of reversing the VG operation. However, there is another issue which is to determine the ways a VG can be changed to another VG that represents another time series. In Chapters 5 and 6, it became clear that relatively few graphs are actually visibility graphs. Finding ways to rewire such that a VG will be rewired into another VG could

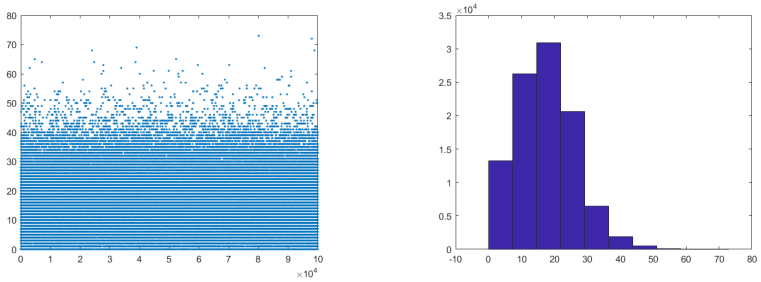
give more insight into the nature of VGs. The restrictions of the adjacency matrix that are posed in Chapter 5 could be a starting point to find legal rewirings.
Lacking a theoretical framework for legal rewirings, empirical results are another avenue worth investigating. A random time series ($x_{original}$) is generated, and its VG ($NVG_{original}$) is constructed. An alteration in the time series is applied by changing a single sample, leading to a time series $x_{altered}$. The VG of this time series, $NVG_{altered}$, is of the same size as the original but has different links. Therefore the difference between $NVG_{original}$ and $NVG_{altered}$ is an example of a legal rewiring by definition. In order to recognize patterns in the rewiring, the method is applied to a time series of length $N = 100$ with $x_{original}(t) \in [0, 1]$. The time series is then altered by setting a random sample to the maximum value, creating a peak at that position in the time series $x_{altered}$. The resulting $NVG_{altered}$ is constructed and the adjacency matrices of both VGs are compared using an XOR operation, which highlights where the matrices differ and so which links have appeared or disappeared. The changes are logged in a $100x100$ matrix named $A_{rewire}$. Because of the stochastic nature of this process, it is repeated $10^5$ times and all results are aggregated. The sum of all matrices $A_{rewire}$ accumulates all the changes that happen in the matrix as a result of legal rewirings. Figure 8.4(a) shows a heatmap of the changes that have occurred over $10^5$ runs. It is unsurprising that most rewirings happen close to the diagonal, as the effects of a changed sample is most noticeable for its near neighbors. However, no patterns can be discerned from the aggregated heatmap.
Instead of raising a random sample to the maximum value, another approach is to select the most connected node of $NVG_{original}$ and changing its corresponding sample. In $x_{altered}$, the entry that corresponds to the most connected node is set to 0. This will ostensibly induce a large rewiring as lower samples are more likely to have lower degree, and thus many links will disappear. The heatmap of aggregated results for this method is shown in Figure 8.4(b). In this instance, the changes occur around the diagonal and away from the first and last nodes of the graph. This is also unsurprising, as the affected entry is selected by highest degree. The node with the highest degree is likely to be more central in the graph than somewhere on the edge of the graph.



(a) Random sample rewiring (b) Highest degree rewiring

Figure 8.4: Left: Heatmap of the changes that occur as a result of rewiring by setting a random sample in $x_{original}$ to be equal to 1. Right: Heatmap of the changes that occur as a result of rewiring by setting the sample corresponding to the node with highest degree in $NVG_{original}$ to be equal to 0.

There are no discernible patterns for either method. The number of changes in the VG is also seemingly random, as shown in Figure 8.5.



(a) Random sample rewiring: number of changes

(b) Distribution of number of changes

Figure 8.5: Left: Number of changes in the adjacency matrix as the result of a rewiring by changing a random sample. Right: Distribution of how often the number of changes occur.

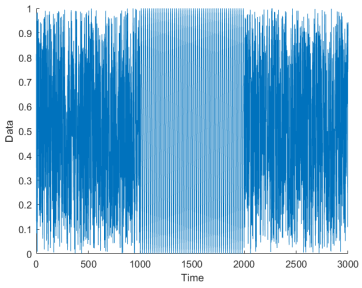
## 8.4. Sliding window

Some authors have chosen to analyze the time series as a set of sub-sequences. The series is divided into segments of length $n$ and a visibility graph of each segment is constructed. The segments are often defined by sliding a window with length $n$ across the whole time series. This can be used to express developments over time in the time series, as the constructed graphs will lose and gain one node at a time when the sliding window is moved. In one of his later works, Lacasa took a different route with the sliding window technique. Instead of finding a temporal pattern in states, Lacasa looked into the probability of states occurring and what these probabilities say about the process with which the initial time series was generated [43].

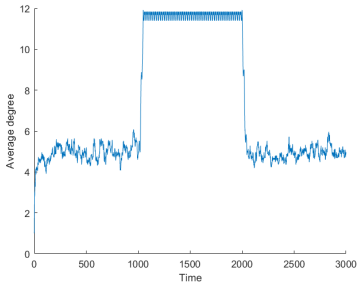
The sliding window is an essential concept for the use of real-time analysis of time series. As shown in Chapter 2, there are applications that can identify the state of a subject from a series of measurements and analysis with a VG. However, these are analysed after the fact. A method in which a time series is continually monitored is still lacking. A sliding window of size $s$ could be utilized to monitor a process. The latest samples $s$ of a time series are used to construct an NVG, and new samples continually replace the oldest ones. An example of this is shown in Figure 8.6. A time series that is formed by a concatenation of 1000 random samples, followed by a sine squared of 1000 samples, followed by another 1000 random samples is used. A sliding window of size $s = 50$ is implemented. It is visible that the average graph metrics change with the nature of time series, as expected. However, the change in metric only becomes completely obvious after a whole window length has passed as seen in Figure 8.7. Furthermore, as noted from the literature discussed in Chapter 2, these simple metrics are not necessarily the best indicators of the state of the time series. For example, in the paper on rolling bearing fault diagnosis [8], a graph function is added to the HVG that is created in order to identify faulty bearings. A topic for future research could be how such analysis methods work when the VGs are much shorter as a result of the window size $s$, and how the window size could be optimized. A longer window is more likely to produce an accurate representation when the time series is in a certain state. However, the transition between states will take longer to become evident if the window is longer. Finding an optimum for this balance will depend on the time series, graph implementation and graph analysis tools that are used.

When utilizing a sliding window for real-time applications, the complexity of the algorithm should also be considered. The algorithm should be able to calculate and analyze the VG of the current window before the next sample is received. In the current implementation of the VG that is used in the creation of Figure 8.6, the construction of the VG and finding the average metrics takes time about $t_w = 3ms$. This holds for a window length $s = 50$. When the window length is extended to $s = 500$, the time it takes to construct and analyze the VG of a full window increases to about $t_w = 0.15s$ on average, where the windows for the sine function take significantly longer to construct than the windows for the random function. This is due to some optimization in the algorithm where samples with low visibility are quicker to calculate. But it seems that time complexity scales quadratically with the length of the window, which is logical because every sample needs to be compared to every other sample, leading to $\mathcal{O}(s^2)$. A new implementation that uses the VG of the previous window as a base for the next window, with the oldest node dropping out and a new node being added, could be much more efficient. It is likely that such an

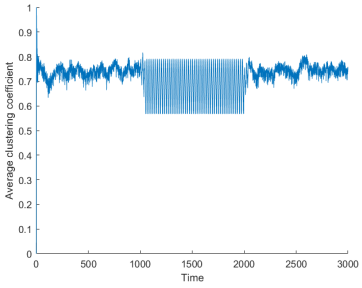
implementation only has a complexity $\mathcal{O}(s)$ as only one sample, the newest one, has to be compared with all other samples in the window. Such an implementation is a useful tool when considering the VG for the analysis of real-time data from a source that quickly generates new samples.



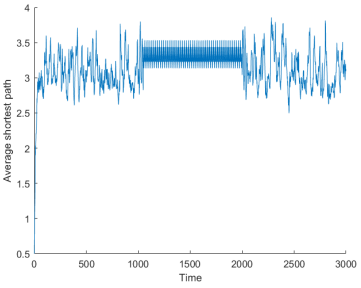

(a) Time series data




(b) Average degree in window

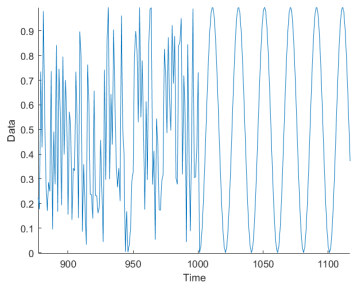



(c) Average clustering coefficient in window

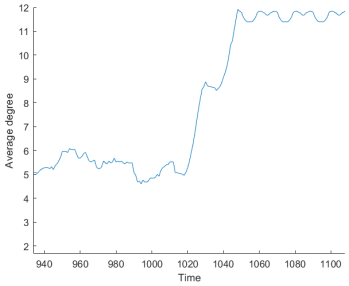



(d) Average shortest path in window

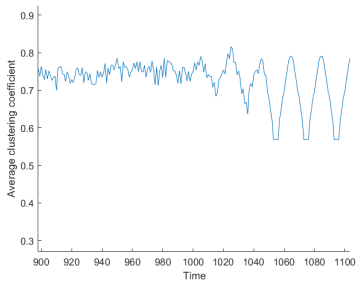
Figure 8.6: An example of how a time series formed by various processes can be monitored using a sliding window and constructing the NVG for each window



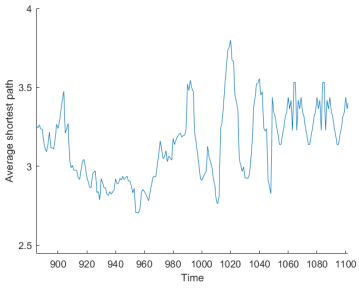

(a) Time series data




(b) Average degree in window




(c) Average clustering coefficient in window




(d) Average shortest path in window

Figure 8.7: An example of how a time series formed by various processes can be monitored using a sliding window of size $s = 50$ and constructing the NVG for each window (zoomed)

# A

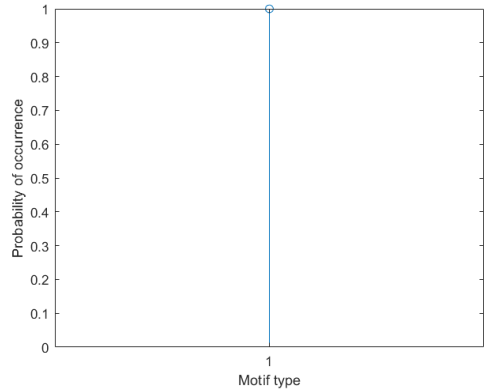
# Appendix A

Table A.1: Random time series of length $N = 10^4$ are generated, where each entry can take p values. This corresponds to samples with log2(p) precision. Some metrics exhibit a clear dependence on the value for p

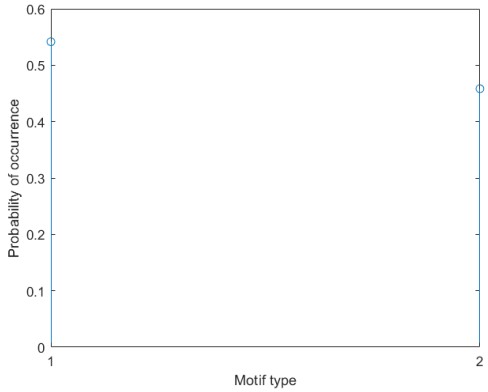
| | | Average degree | Clustering coefficient | Average shortest path | Assortativity |
|---|---|---|---|---|---|
| NVG | p = 16 | 5.1174 | 0.7429 | 207.8 | 0.0679 |
| | p = 32 | 5.2976 | 0.7462 | 111.0 | 0.1017 |
| | p = 64 | 5.3630 | 0.7479 | 54.10 | 0.1378 |
| | p = 128 | 5.3976 | 0.7476 | 29.66 | 0.1652 |
| | p = 256 | 5.4620 | 0.7482 | 19.86 | 0.1802 |
| | p = 512 | 5.4542 | 0.7486 | 14.36 | 0.1909 |
| | p = 1024 | 5.4602 | 0.7481 | 10.30 | 0.1947 |
| HVG | p = 16 | 3.5764 | 0.5559 | 209.5 | 0.0296 |
| | p = 32 | 3.3792 | 0.5976 | 113.0 | 0.0618 |
| | p = 64 | 3.8500 | 0.6242 | 56.30 | 0.0931 |
| | p = 128 | 3.9104 | 0.6333 | 32.00 | 0.1291 |
| | p = 256 | 3.9534 | 0.6403 | 22.39 | 0.1531 |
| | p = 512 | 3.9700 | 0.6451 | 17.48 | 0.1618 |
| | p = 1024 | 3.9828 | 0.6468 | 13.69 | 0.1657 |
| NLPVG | p = 16 | 10.3434 | 0.7268 | 104.8 | 0.1078 |
| | p = 32 | 10.6824 | 0.7257 | 56.50 | 0.1512 |
| | p = 64 | 10.8224 | 0.7267 | 28.27 | 0.1908 |
| | p = 128 | 10.8180 | 0.7267 | 16.18 | 0.2185 |
| | p = 256 | 10.9532 | 0.7270 | 11.44 | 0.2337 |
| | p = 512 | 10.9610 | 0.7265 | 8.86 | 0.2366 |
| | p = 1024 | 10.9800 | 0.7282 | 7.01 | 0.2450 |
| HLPVG | p = 16 | 7.1596 | 0.6286 | 105.8 | 0.1479 |
| | p = 32 | 7.4788 | 0.6407 | 57.77 | 0.1755 |
| | p = 64 | 7.7014 | 0.6467 | 29.57 | 0.2157 |
| | p = 128 | 7.8242 | 0.6522 | 17.59 | 0.2362 |
| | p = 256 | 7.9028 | 0.6519 | 13.03 | 0.2572 |
| | p = 512 | 7.9360 | 0.6529 | 10.67 | 0.2653 |
| | p = 1024 | 7.9620 | 0.6533 | 8.93 | 0.2764 |

Table A.2: Sine wave time series of length $N = 10^3$ generated with added white Gaussian noise at different levels. The series is generated as $x_t = sin(4 * \pi * t)$ with SNR levels 0 dB, 5 dB, 10 dB and 20 dB

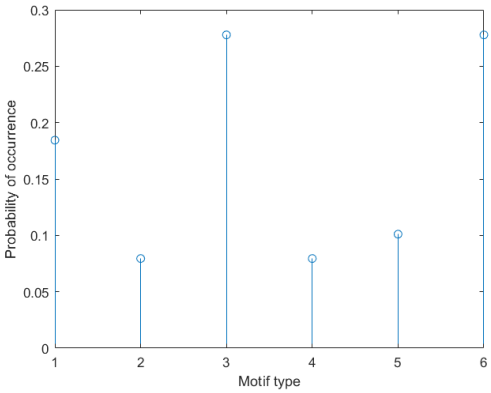
| | | Average degree | Clustering coefficient | Average shortest path | Assortativity |
|---|---|---|---|---|---|
| NVG | SNR = 0 | 6.0719 | 0.7537 | 5.0132 | 0.1914 |
| | SNR = 5 | 6.7712 | 0.7647 | 4.7743 | 0.1565 |
| | SNR = 10 | 7.3866 | 0.7543 | 5.1308 | 0.1734 |
| | SNR = 20 | 8.2098 | 0.7438 | 5.4982 | 0.2858 |
| HVG | SNR = 0 | 3.9660 | 0.6369 | 7.7137 | 0.1693 |
| | SNR = 5 | 3.9740 | 0.6168 | 7.7717 | 0.1816 |
| | SNR = 10 | 3.9740 | 0.5916 | 8.4360 | 0.2189 |
| | SNR= 20 | 3.9640 | 0.5601 | 9.1711 | 0.3297 |
| NLPVG | SNR = 0 | 11.8442 | 0.7495 | 3.9952 | 0.2185 |
| | SNR = 5 | 12.1399 | 0.7613 | 3.8642 | 0.2191 |
| | SNR = 10 | 12.3716 | 0.7805 | 3.9597 | 0.2509 |
| | SNR = 20 | 11.7163 | 0.7828 | 4.2771 | 0.3321 |
| HLPVG | SNR = 0 | 7.9121 | 0.6542 | 5.5992 | 0.2898 |
| | SNR = 5 | 7.9281 | 0.6600 | 5.6509 | 0.3612 |
| | SNR = 10 | 7.9321 | 0.6682 | 5.6311 | 0.4165 |
| | SNR = 20 | 7.8981 | 0.6842 | 6.0044 | 0.4724 |
| DPVG | SNR = 0 | 10.1818 | 0.5706 | 4.3233 | 0.1366 |
| | SNR = 5 | 11.6284 | 0.5823 | 4.1680 | 0.1423 |
| | SNR = 10 | 12.7712 | 0.5907 | 4.4229 | 0.1350 |
| | SNR = 20 | 14.5435 | 0.6172 | 4.3671 | 0.1843 |



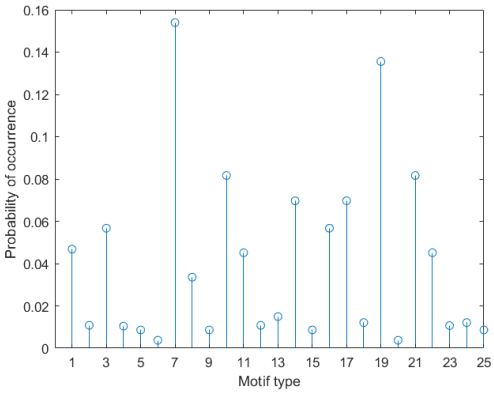

(a) Distribution of motifs for length $n = 2$



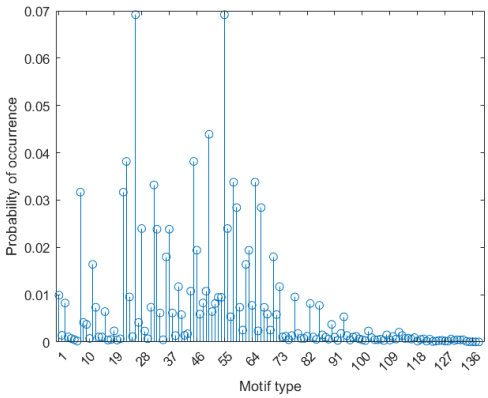

(b) Distribution of motifs for length $n = 3$




(c) Distribution of motifs for length $n = 4$




(d) Distribution of motifs for length $n = 5$




(e) Distribution of motifs for length $n = 6$

Figure A.1: On the x-axis of each of the plots, the number of unique VGs for that length $n$ is shown. The probability of the unique VG occurring as a result of a random sequence of length $n$ and sampling depth $p = 6$ is plotted

# References


[1] L. Lacasa, B. Luque, F. Ballesteros, J. Luque, and J. C. Nuño, "From time series to complex networks: The visibility graph," *Proceedings of the National Academy of Sciences*, vol. 105, no. 13, pp. 4972–4975, 2008. DOI: 10.1073/pnas.0709247105. [Online]. Available: https://www.pnas.org/doi/abs/10.1073/pnas.0709247105.

[2] N. Marwan, M. Carmen Romano, M. Thiel, and J. Kurths, "Recurrence plots for the analysis of complex systems," *Physics Reports*, vol. 438, no. 5, pp. 237–329, 2007, ISSN: 0370-1573. DOI: https://doi.org/10.1016/j.physrep.2006.11.001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0370157306004066.

[3] J. W. ESSAM and M. E. FISHER, "Some basic definitions in graph theory," *Rev. Mod. Phys.*, vol. 42, pp. 271–288, 2 Apr. 1970. DOI: 10.1103/RevModPhys.42.271. [Online]. Available: https://link.aps.org/doi/10.1103/RevModPhys.42.271.

[4] Q. Xuan, J. Zhou, K. Qiu, D. Xu, S. Zheng, and X. Yang, "Clpvg: Circular limited penetrable visibility graph as a new network model for time series," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 32, no. 1, p. 013 130, 2022. DOI: 10.1063/5.0048243. eprint: https://doi.org/10.1063/5.0048243. [Online]. Available: https://doi.org/10.1063/5.0048243.

[5] B. Luque, L. Lacasa, F. Ballesteros, and J. Luque, "Horizontal visibility graphs: Exact results for random time series," *Phys. Rev. E*, vol. 80, p. 046 103, 4 Oct. 2009. DOI: 10.1103/PhysRevE.80.046103. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevE.80.046103.

[6] M. Zheng, S. Domanskyi, C. Piermarocchi, and G. I. Mias, "Visibility graph based temporal community detection with applications in biological time series.," *Sci Rep*, vol. 11, pp. 201–208, 2021, ISSN: 0960-0779. DOI: https://doi.org/10.1038/s41598-021-84838-x.

[7] M. Diykh, Y. Li, and P. Wen, "Eeg sleep stages classification based on time domain features and structural graph similarity," *IEEE Transactions on Neural Systems and Rehabilitation Engineering*, vol. 24, no. 11, pp. 1159–1168, 2016. DOI: 10.1109/TNSRE.2016.2552539.

[8] Y. Gao and D. Yu, "Total variation on horizontal visibility graph and its application to rolling bearing fault diagnosis," *Mechanism and Machine Theory*, vol. 147, p. 103 768, 2020, ISSN: 0094-114X. DOI: https://doi.org/10.1016/j.mechmachtheory.2019.103768. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0094114X19323742.

[9] L. Telesca and M. Lovallo, "Analysis of seismic sequences by using the method of visibility graph," *Europhysics Letters*, vol. 97, no. 5, p. 50 002, Feb. 2012. DOI: 10.1209/0295-5075/97/50002. [Online]. Available: https://dx.doi.org/10.1209/0295-5075/97/50002.

[10] Y. Hu and F. Xiao, "A novel method for forecasting time series based on directed visibility graph and improved random walk," *Physica A: Statistical Mechanics and its Applications*, vol. 594, p. 127 029, 2022, ISSN: 0378-4371. DOI: https://doi.org/10.1016/j.physa.2022.127029. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378437122000978.

[11] R. V. Donner, Y. Zou, J. F. Donges, N. Marwan, and J. Kurths, "Recurrence networks—a novel paradigm for nonlinear time series analysis," *New Journal of Physics*, vol. 12, no. 3, p. 033 025, Mar. 2010. DOI: 10.1088/1367-2630/12/3/033025. [Online]. Available: https://dx.doi.org/10.1088/1367-2630/12/3/033025.

[12] Z. Gao and N. Jin, "Flow-pattern identification and nonlinear dynamics of gas-liquid two-phase flow in complex networks," *Phys. Rev. E*, vol. 79, p. 066 303, 6 Jun. 2009. DOI: 10.1103/PhysRevE.79.066303. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevE.79.066303.

[13] B. Kliková and A. Raidl, “Reconstruction of phase space of dynamical systems using method of time delay,” 2011.

[14] M. Kennel, R. B. R, and H. Abarbanel, “Determining embedding dimension for phase-space reconstruction using a geometrical construction.,” *Phys. Rev. E*, vol. 45, pp. 3403–3411, 6 Mar. 1992.

[15] J. Eckmann, S. Kamphorst, and D. Ruelle, “Recurrence plots of dynamical systems,” *Europhysics Letters*, vol. 4, pp. 973–977, 1987. [Online]. Available: http://dx.doi.org/10.1209/0295-5075/4/9/004.

[16] R. V. DONNER, M. SMALL, J. F. DONGES, *et al.*, “Recurrence-based time series analysis by means of complex network methods,” *International Journal of Bifurcation and Chaos*, vol. 21, no. 04, pp. 1019–1046, 2011. DOI: 10.1142/S0218127411029021. eprint: https://doi.org/10.1142/S0218127411029021. [Online]. Available: https://doi.org/10.1142/S0218127411029021.

[17] A. M. Nuñez, L. Lacasa, J. P. Gomez, and B. Luque, “Visibility algorithms: A short review,” in *New Frontiers in Graph Theory*, Y. Zhang, Ed., Rijeka: IntechOpen, 2012, ch. 6. DOI: 10.5772/34810. [Online]. Available: https://doi.org/10.5772/34810.

[18] Z. Ting-Ting, J. Ning-De, G. Zhong-Ke, and L. Yue-Bin, “Limited penetrable visibility graph for establishing complex network from time series,” *Acta Physica Sinica*, vol. 61, no. 3, 2012. DOI: 10.7498/aps.61.030506. [Online]. Available: https://wulixb.iphy.ac.cn/en/article/doi/10.7498/aps.61.030506.

[19] Z.-K. Gao, Q. Cai, Y.-X. Yang, W.-D. Dang, and S.-S. Zhang, “Multiscale limited penetrable horizontal visibility graph for analyzing nonlinear time series,” *Scientific Reports*, vol. 6, no. 1, 2016. DOI: 10.1038/srep35622. [Online]. Available: https://doi.org/10.1038/srep35622.

[20] B. A. Gonçalves, L. Carpi, O. A. Rosso, and M. G. Ravetti, “Time series characterization via horizontal visibility graph and information theory,” *Physica A: Statistical Mechanics and its Applications*, vol. 464, pp. 93–102, 2016.

[21] P. Xu, R. Zhang, and Y. Deng, “A novel visibility graph transformation of time series into weighted networks,” *Chaos, Solitons and Fractals*, vol. 117, pp. 201–208, 2018, ISSN: 0960-0779. DOI: https://doi.org/10.1016/j.chaos.2018.07.039. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0960077918307276.

[22] S. Supriya, S. Siuly, H. Wang, J. Cao, and Y. Zhang, “Weighted visibility graph with complex network features in the detection of epilepsy,” *IEEE Access*, vol. 4, pp. 6554–6566, 2016. DOI: 10.1109/ACCESS.2016.2612242.

[23] Z. Mohammadpoory, M. Nasrolahzadeh, and J. Haddadnia, “Epileptic seizure detection in eegs signals based on the weighted visibility graph entropy,” *Seizure*, vol. 50, pp. 202–208, 2017, ISSN: 1059-1311. DOI: https://doi.org/10.1016/j.seizure.2017.07.001. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S105913111730198X.

[24] L. Lacasa, A. Nunez, É. Roldán, J. M. Parrondo, and B. Luque, “Time series irreversibility: A visibility graph approach,” *The European Physical Journal B*, vol. 85, pp. 1–11, 2012.

[25] L. M. Sander, “Fractal growth processes,” in *Mathematics of Complexity and Dynamical Systems*, R. A. Meyers, Ed. New York, NY: Springer New York, 2011, pp. 429–445, ISBN: 978-1-4614-1806-1. DOI: 10.1007/978-1-4614-1806-1_28. [Online]. Available: https://doi.org/10.1007/978-1-4614-1806-1_28.

[26] L. Telesca and M. Lovallo, “Non-uniform scaling features in central italy seismicity: A non-linear approach in investigating seismic patterns and detection of possible earthquake precursors,” *Geophysical Research Letters*, vol. 36, no. 1, 2009.

[27] A. Bhaduri and D. Ghosh, “Quantitative assessment of heart rate dynamics during meditation: An ecg based study with multi-fractality and visibility graph,” *Frontiers in Physiology*, vol. 7, 2016, ISSN: 1664-042X. DOI: 10.3389/fphys.2016.00044. [Online]. Available: https://www.frontiersin.org/articles/10.3389/fphys.2016.00044.

[28] M. Stephen, C. Gu, and H. Yang, “Visibility graph based time series analysis,” *PLOS ONE*, 2015. [Online]. Available: https://doi.org/10.1371/journal.pone.0143015.

[29] L. Lacasa, B. Luque, J. Luque, and J. C. Nuño, “The visibility graph: A new method for estimating the hurst exponent of fractional brownian motion,” *Europhysics Letters*, vol. 86, no. 3, p. 30 001, May 2009. DOI: 10.1209/0295-5075/86/30001. [Online]. Available: https://dx.doi.org/10.1209/0295-5075/86/30001.

[30] M. D. Kale and F. B. Butar, “Fractal analysis of time series and distribution properties of hurst exponent,” Ph.D. dissertation, Sam Houston State University, 2005.

[31] M. Sánchez Granero, J. Trinidad Segovia, and J. García Pérez, “Some comments on hurst exponent and the long memory processes on capital markets,” *Physica A: Statistical Mechanics and its Applications*, vol. 387, no. 22, pp. 5543–5551, 2008, ISSN: 0378-4371. DOI: https://doi.org/10.1016/j.physa.2008.05.053. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0378437108004895.

[32] Y.-Z. Wang, B. Li, R.-Q. Wang, J. Su, and X.-X. Rong, “Application of the hurst exponent in ecology,” *Computers and Mathematics with Applications*, vol. 61, no. 8, pp. 2129–2131, 2011, Advances in Nonlinear Dynamics, ISSN: 0898-1221. DOI: https://doi.org/10.1016/j.camwa.2010.08.095. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0898122110006802.

[33] L. Lacasa and R. Toral, “Description of stochastic and chaotic series using visibility graphs,” *Physical Review E*, vol. 82, no. 3, Sep. 2010. DOI: 10.1103/physreve.82.036120. [Online]. Available: https://doi.org/10.1103%2Fphysreve.82.036120.

[34] S. Jiang, C. Bian, X. Ning, and Q. D. Ma, “Visibility graph analysis on heartbeat dynamics of meditation training,” *Applied Physics Letters*, vol. 102, no. 25, p. 253 702, 2013.

[35] M. Ahmadlou, H. Adeli, and A. Adeli, “Improved visibility graph fractality with application for the diagnosis of autism spectrum disorder,” *Physica A: Statistical Mechanics and its Applications*, vol. 391, no. 20, pp. 4720–4726, 2012.

[36] M.-C. Qian, Z.-Q. Jiang, and W.-X. Zhou, “Universal and nonuniversal allometric scaling behaviors in the visibility graphs of world stock market indices,” *Journal of Physics A: Mathematical and Theoretical*, vol. 43, no. 33, p. 335 002, 2010.

[37] Q. Xuan, J. Wang, M. Zhao, *et al.*, “Subgraph networks with application to structural feature space expansion,” *IEEE Transactions on Knowledge and Data Engineering*, vol. 33, no. 6, pp. 2776–2789, 2021. DOI: 10.1109/TKDE.2019.2957755.

[38] D. I. Shuman, S. K. Narang, P. Frossard, A. Ortega, and P. Vandergheynst, “The emerging field of signal processing on graphs: Extending high-dimensional data analysis to networks and other irregular domains,” *IEEE Signal Processing Magazine*, vol. 30, no. 3, pp. 83–98, 2013. DOI: 10.1109/MSP.2012.2235192.

[39] G. J. McLachlan, “Mahalanobis distance,” *Resonance*, vol. 4, no. 6, pp. 20–26, 1999.

[40] M. Zheng, S. Domanskyi, C. Piermarocchi, and G. I. Mias, “Visibility graph based temporal community detection with applications in biological time series,” *Scientific reports*, vol. 11, no. 1, pp. 1–12, 2021.

[41] N. Ahmadi, R. M. Besseling, and M. Pechenizkiy, “Assessment of visibility graph similarity as a synchronization measure for chaotic, noisy and stochastic time series,” *Social Network Analysis and Mining*, vol. 8, no. 1, pp. 1–17, 2018.

[42] M. Ahmadlou and H. Adeli, “Visibility graph similarity: A new measure of generalized synchronization in coupled dynamic systems,” *Physica D: Nonlinear Phenomena*, vol. 241, no. 4, pp. 326–332, 2012, ISSN: 0167-2789. DOI: https://doi.org/10.1016/j.physd.2011.09.008. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0167278911002491.

[43] J. Iacovacci and L. Lacasa, “Sequential visibility-graph motifs,” *Phys. Rev. E*, vol. 93, p. 042 309, 4 Apr. 2016. DOI: 10.1103/PhysRevE.93.042309. [Online]. Available: https://link.aps.org/doi/10.1103/PhysRevE.93.042309.

[44] A. NUÑEZ, L. LACASA, E. VALERO, J. P. GÓMEZ, and B. LUQUE, “DETECTING SERIES PERIODICITY WITH HORIZONTAL VISIBILITY GRAPHS,” *International Journal of Bifurcation and Chaos*, vol. 22, no. 07, p. 1 250 160, Jul. 2012. DOI: 10.1142/s021812741250160x. [Online]. Available: https://doi.org/10.1142%5C%2Fs021812741250160x.

[45] L. Lacasa, “Horizontal visibility graphs from integer sequences,” *Journal of Physics A: Mathematical and Theoretical*, vol. 49, no. 35, 35LT01, Jul. 2016. DOI: 10.1088/1751-8113/49/35/35LT01. [Online]. Available: https://dx.doi.org/10.1088/1751-8113/49/35/35LT01.

[46] M. Wang, A. L. M. Vilela, R. Du, *et al.*, *Exact results of the limited penetrable horizontal visibility graph associated to random time series and its application*, Mar. 2018. [Online]. Available: https://doi.org/10.1038/s41598-018-23388-1.

[47] R. Noldus and P. Van Mieghem, “Assortativity in complex networks,” *Journal of Complex Networks*, vol. 3, no. 4, pp. 507–542, Mar. 2015, ISSN: 2051-1310. DOI: 10.1093/comnet/cnv005. eprint: https://academic.oup.com/comnet/article-pdf/3/4/507/2328341/cnv005.pdf. [Online]. Available: https://doi.org/10.1093/comnet/cnv005.