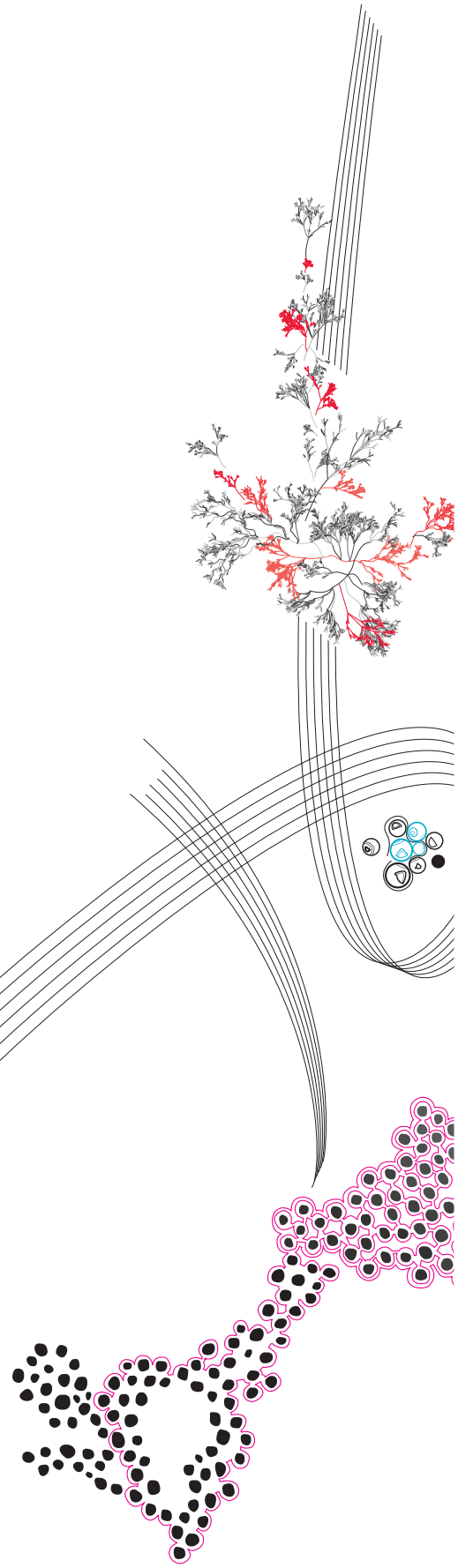


BSc Thesis Applied Mathematics

A study on constellations using random graphs

Justus Sleurink

Supervisor: dr. C. Stegehuis

February, 2021

Department of Applied Mathematics
Faculty of Electrical Engineering,
Mathematics and Computer Science

UNIVERSITY OF TWENTE.

Preface

I would like to thank my supervisor dr. C. Stegehuis for her guidance and her patience, as well as dr D. Bucur for sharing constellation data that she compiled and providing the Python code used to retrieve all relevant star data. Lastly, I would like to thank everyone that helped enable me to write this thesis.

A study on constellations using random graphs

Justus Sleurink

February, 2021

Abstract

In this paper, we develop a mathematical model that attempts to describe constellations. We present a random graph model that generates links between two stars based on their brightness and the angular distance between them, weighed against the brightness of and distance to other stars. We use the model to determine whether there is a mathematical mechanism behind constellations. If a suitable mathematical mechanism is found, it could help quantify the performance of the human mind on visualising structures. We measure the performance of the model based on the overlap between links in the constellation and links generated by the model. We find that on average slightly less than half of the links overlap. We also find that adjusting the attenuation function that determines the influence of the angular distance between the stars on the probability that a link is generated does not a□ect the results. Generally, the model performs slightly better for smaller constellations. We conclude from the results that the specific model presented with the way we implemented it does not imply that there is a mathematical mechanism behind constellations. We suggest that the model could be improved by increasing the influence of the interference factor, or requiring the generated links to form a connected component. Additionally, the model can also be applied to all visible stars in a bounded area.

Keywords: constellation, random graph, signal to interference ratio graph model, angular distance, attenuation

1 Introduction

Constellations have long been an object of interest within astronomy. The origins of constellations are generally unknown, but many of the well known constellations in western culture are originally Greek. Constellations are simply imaginary patterns in a group of visible stars, that are used to represent for instance animals or mythological beasts, as can be recognised from the given names of some constellations. Nowadays, the International Astronomical Union (IAU) has a list of 88 recognised constellations[9]. When talking about constellations, we usually mean one or multiple of the 88 constellations as compiled by the IAU. As constellations consist of a set of stars with a set of imaginary links between them, someone with basic knowledge of graph theory can quickly see that constellations can also be interpreted as graphs.

Previously, from the perspective of psychology, research has been done to inspect the patterns that people recognized in different point constellations.[3] These point constellations in fact corresponded to certain actual constellations. The goal of their research was however not to gain more insight into how constellations are formed, but instead to gain more insight into the performance of the human mind on the task of pattern recognition. However, constellations are essentially creations of the human imagination, which implies that looking for the logic behind the structure of constellations is indirectly also a research

into the human mind, just with a different approach. Through searching for literature we suspect there is not much precedent in looking for a mathematical mechanism behind the structure of constellations.

The content of this thesis is exactly that, as we research whether there is a mathematical mechanism behind constellations, by using a probabilistic mathematical model that generates random graphs. To be able to answer whether there is or is not a mathematical mechanism behind constellations, we have to answer some smaller questions first. Considering that constellations are originally based on stars as observed from the earth, how do we calculate the perceived distance between two stars as observed from earth? Consequently, how do we define probabilities that stars are connected based on their brightness and the perceived distance between them? And finally, once we have constructed a model that generates links based on those probabilities, how do we quantify the performance of this model on describing constellations?

In order to answer these questions we first introduce the concept of random graphs, which combines concepts of graph theory and probability theory. We then proceed by introducing an existing model, called the Signal to Interference Ratio Graph (STIRG) model, that is used in the context of wireless networks[2]. This model generates links between nodes in the plane if their signal strength is large enough compared to the interference from the signals of other nodes. We explain why the STIRG model can be used as a basis to construct a suitable random graph model for application on constellations, and how we adjust it to use the angular distance between stars, as it correlates closely with the perceived distance, and the brightness of stars. We make further adjustments to the model to calculate probabilities that any two stars in a set of stars are connected, such that it generates links between stars based on those probabilities. Furthermore, we explain how we control the expected size of the graph consisting of the generated links. By implementing the random graph model we created in Python, we generate results regarding all 88 constellations of the IAU based on the overlap between the links of the existing constellations and those generated by the model, and explain why the model we constructed does not perform well on the task of generating structures resembling constellations. At the end, we conclude based on these observations whether the model we constructed implies that there is a mathematical mechanism behind constellations, and we discuss possibilities for future research.

2 Random Graphs

Random graphs are essentially the result of combining concepts from the fields of graph theory and probability theory, as the name suggests. The term random graph is used generally, as random graphs can for example have randomness in the edges, when for instance each possible edge on a set of vertices appears independently of each other with a certain probability p, as in Erdős-Rényi random graphs[5]. Another, less prominent example is where the vertices of a graph appear at positions in the plane according to some stochastic process and whether vertices are connected by an edge or not is determined by some condition[1]. Applications of random graphs include a wide variety of modelling exercises, like in this paper. Another possible application is in the field of social sciences, for instance when representing human acquaintances in a graph, as that graph shares certain properties with Erdős-Rényi random graphs[6].

As we aim to construct a model that we can apply to sets of stars which have a fixed position, our model will be a random graph model where the randomness is present in the edges, with every possible edge having a certain probability that it is present assigned

to it. The existing model that we introduce in the next section, upon which we base our random graph model, instead places edges in a deterministic manner.

3 Adjusted Signal to Interference Ratio Graph Model

To be able to model constellations as a random graph based on properties of stars, we identify three key factors for determining the likelihood that two stars are connected. First of all, we assume that the perceived distance between two stars influences the probability, namely that the further away two stars are from each other, the lower the probability that the two stars are connected in a constellation is. The second assumption we make is that brighter stars have a higher probability to be used in the formation of a constellation. Finally, we assume that two stars are less likely to be connected if there are other visible stars situated in between or surrounding them. To that extent, neighbouring stars will also be taken into account when determining whether two stars are connected.

3.1 Input variables

Based on the key factors identified, our model requires as input the positional coordinates of a set of stars, as well as their brightness. If we imagine the starry sky as a hollow sphere centered around the earth with the inside of the sphere being an image with all the stars, what we would observe when looking up at night does not really change. This allows us to represent the position of stars on that sphere in a way similar to how we do with places on earth by latitude and longitude. This coordinate system is the system most often used to describe the positions of stars, and uses declination and right ascension corresponding to latitude and longitude respectively. The so-called celestial equator lies above the equator of the earth, implying that points on the celestial equator have a declination of 0 degrees, while the north and south celestial poles have a declination of 90 and -90 degrees respectively. Right ascension is measured from the spot the sun arrives at on the first day of spring, the vernal equinox, and is usually measured in hours. For our model, we instead use the right ascension in degrees, which means it is 0 degrees at the vernal equinox, and increases the further east around the earth we go, up to 360 degrees when returning to the vernal equinox. A more detailed explanation about celestial coordinates is available on the internet[7]. The measure that we will use for the perceived distance between two stars is the angular distance based on the celestial coordinates, as the angular distance is approximately the distance between two stars that we observe when looking up at the stars. We will denote the declination and right ascension of a star i as $_i$ and $_i$ respectively. Additionally, we denote the brightness of a star i by P_i, with P_i being larger the brighter a star is.

3.2 Signal to Interference Ratio Graph Model

An existing model that incorporates all three of the key components identified is the Signal to Interference Ratio Graph (STIRG) model[2], which is originally used in the context of wireless networks. The STIRG model operates on the basis that a node i can transmit data to a node j if the signal that j receives is strong enough, compared to interference from other signals and background noise. This condition is presented as

$$\frac{P_i L(\mathbf{x_i} \quad \mathbf{x_j})}{N_0 + \sum_{k \in i,j} P_k L(\mathbf{x_k} \quad \mathbf{x_j})} \quad : \tag{1}$$

Similarly, the condition that determines whether a node j can transmit to a node i is

$$\frac{P_j L(\mathbf{x_j} \quad \mathbf{x_i})}{N_0 + \sum_{k \in i;j} P_k L(\mathbf{x_k} \quad \mathbf{x_i})} \quad : \tag{2}$$

To simplify, they say that nodes i and j are directly connected only if both conditions are satisfied. The power of the signal transmitted from node i to node j is $P_i L(\mathbf{x_i} \quad \mathbf{x_j})$. Here, P_i represents the emitting power of a node i, and $\mathbf{x_i}$ and $\mathbf{x_j}$ represent the positions of nodes i and j in the plane. $L(\,)$ is the so-called attenuation function. Attenuation generally stands for a reduction in strength of a signal of some sort, so the purpose of the attenuation function here is to reduce the strength of the transmitted signal the larger the distance between the nodes in question is. Furthermore, N_0 is the power of the thermal background noise, and is a weight factor for how much different simultaneous signals interfere with each other. The threshold for the signal to interference ratio necessary for transmission is given by . In the original model, the attenuation function is defined as $L(\mathbf{x}) = l(jj\mathbf{x}jj)$ where they note that for $l(t) = t^{\ 2}$ no connection is possible for any > 0. Therefore the most common attenuation function used is $l(t) = t^{\ a}$, with a ranging from 3 to 6, with possible variations where the function is bounded.

In order to translate the model to suit the context of stars and constellations, we have instead defined P_i as the brightness of a star as mentioned previously. Also, the attenuation function is now based on the position vectors $\mathbf{x_i} = (\ _i;\ _i)$ and $\mathbf{x_j} = (\ _j;\ _j)$. We define the attenuation function to be $L(\mathbf{x_i} \quad \mathbf{x_j}) = l(jj\mathbf{x_i} \quad \mathbf{x_j}jj)$ with the norm equaling the angular distance, so

$$jj\mathbf{x_i} \quad \mathbf{x_j}jj = \arccos(\sin(\ _i)\sin(\ _j) + \cos(\ _i)\cos(\ _j)\cos(\quad));\tag{3}$$

where $= \ _i \quad \ _j$[4]. Furthermore, we similarly define $l(t) = t^{\ a}$ such that depending on the choice of a we can determine the degree to which the signal is weakened based on the angular distance between two stars. The limitation that no connection is possible for any > 0 when we use $a = 2$ becomes irrelevant in our adjusted model, as we will define probabilities for edges instead of setting a threshold. Lastly, the parameters N_0, and have lost their original meaning in the context of wireless networks, so we instead use them in the adjusted model to control the size of the generated graphs.

3.3 The Adjusted Model

The STIRG model was originally applied to a random scattering of nodes, and as noted, for our purposes the set of nodes always remains the same, as the stars each have a set position and brightness. Thus, if we would base which nodes are connected only on the conditions in equations 1 and 2, we would have one resulting graph for each set of stars used as input. In order to use the STIRG model to study constellations, we adjust it to be a probabilistic model, with probabilities based on the computed signal to interference ratios determining which stars are connected and which stars are not. We first define

$$p_{ij} := \frac{P_i L(\mathbf{x_i} \quad \mathbf{x_j})}{N_0 + \sum_{k \in i;j} P_k L(\mathbf{x_k} \quad \mathbf{x_j})} \quad \frac{P_j L(\mathbf{x_j} \quad \mathbf{x_i})}{N_0 + \sum_{k \in i;j} P_k L(\mathbf{x_k} \quad \mathbf{x_i})} \tag{4}$$

as a variable to base the probability that two stars i and j are connected, p_{ij}, on. Naturally, we must define p_{ij} in such a way that $0 \quad p_{ij} \quad 1$ for all pairs of nodes i and j. Additionally, we use the free variables N_0 and to control the size of the generated graphs. Note that is ignored, as it is originally used as a threshold and is thus omitted from the definition of p_{ij} in equation 4.

The average degree of the existing constellations d can be simply computed from the the number of edges e and the number of nodes n as $d = \frac{2e}{n}$. Similarly, the expected degree of the constellations we want to generate can be computed as

$$\mathrm{E}[D] = \frac{2}{n} \sum_{i;j} p_{ij}: \tag{5}$$

In order to ensure that the generated and original constellations have a similar number of edges, we then define p_{ij} in such a way that $\mathrm{E}[D] = d$. We note that β_{ij} already satisfies $0 \quad \beta_{ij}$, as all quantities in Equation 4 are positive. Simply defining $p_{ij} = \min(\beta_{ij}; 1)$ immediately satisfies $0 \quad p_{ij} \quad 1$, but in order to satisfy Equation 5 we still need to find suitable values of N_0 and $\quad$. To do so, we first define $\beta_{ij;1}$ as equaling β_{ij} with $N_0 = 1$ and $= 1$, so

$$\beta_{ij;1} := \frac{P_i L(\mathbf{x_i} \quad \mathbf{x_j})}{1 + \sum_{k \in i;j} P_k L(\mathbf{x_k} \quad \mathbf{x_j})} \quad \frac{P_j L(\mathbf{x_j} \quad \mathbf{x_i})}{1 + \sum_{k \in i;j} P_k L(\mathbf{x_k} \quad \mathbf{x_i})} \tag{6}$$

and set $N_0 = \quad = C$, such that we now have that $\beta_{ij} = \frac{\beta_{ij;1}}{C}$. This results in

$$\mathrm{E}[D] = \frac{2}{n} \sum_{i;j} p_{ij} = \frac{2}{n} \sum_{i;j} \min(\beta_{ij}; 1) = \frac{2}{n} \sum_{i;j} \min \quad \frac{\beta_{ij;1}}{C}; 1 \tag{7}$$

which, if we now equate the right-hand side of Equation 7 to d such that $\mathrm{E}[D] = d$, can then be rewritten as

$$f(C) := \frac{2}{nd} \sum_{i;j} \min(\beta_{ij;1}; C) = C \tag{8}$$

by multiplying both sides by $\frac{C}{d}$. The remaining issue is then to find a value of C such that $f(C) = C$. In other words, we need to find the fixed point of $f(C)$, if it exists. Finding the fixed point of a function can be achieved in multiple ways, the easiest achieved through fixed point iteration. Fixed point iteration is done by setting an initial point x_0 and iterating with

$$x_{n+1} = f(x_n); n = 0; 1; 2; : : : \tag{9}$$

resulting in the sequence $x_0; x_1; x_2; : : :$ which hopefully converges. If the sequence converges to a point x, that point satisfies $f(x) = x$ and thus is a fixed point of the function. Additionally, in order to increase the rate of convergence from linear to quadratic, Aitken's ^{2} process can be applied. Aitken's ^{2} process defines a new sequence with terms

$$y_n = \frac{x_n x_{n+1} \quad x_{n+1}^2}{x_n \quad 2x_{n+1} + x_{n+2}}; n = 0; 1; 2; : : : \tag{10}$$

where it is proven in its application on the sequence generated by fixed point iteration that the sequence $fy_n g_{n2\mathbb{N}_0}$ converges to the fixed point x faster than the sequence $fx_n g_{n2\mathbb{N}_0}$. The application of Aitken's ^{2} process to fixed point iteration is called Steffensen's method, of which an existing implementation in Python is available which significantly speeds up the computation time for the model. Based on experimentation, the sequence generated by Steffensen's method always converges for our model when using initial point $C_0 = 1$. Through finding a suitable value for C, we have achieved the definition of suitable connection probabilities p_{ij} for our random graph model based on the STIRG model.

4 Comparing Constellations and the Adjusted STIRG Model

Now that our model is defined, we need to consider what input data we use before we can determine our results.

4.1 Data

As input for our model we use star data provided by the SIMBAD astronomical database[10]. The data provides the right ascension and declination in degrees, which we have denoted by $_i$ and $_i$ respectively for a star i. The data also provides the apparent magnitude, denoted by m_i for star i, to determine the brightness of the stars. A lower apparent magnitude represents a brighter star. The relation between the apparent magnitude and brightness of a star P_i, is given by $P_i = (\frac{1}{2.512})^{m_i}$, considering that a difference of 5 magnitudes is a difference in brightness of a factor 100 and $2.512^5 \quad 100$[8].

4.2 Model Performance

Now that we are able to find connection probabilities for a given set of stars, we can generate sets of links between stars of existing constellations through an implementation of the model in Python and compare those generated links to the links that are part of the actual constellation. We compare our adjusted STIRG model to each of the 88 constellations as recognised by the IAU.

In order to measure the performance of the adjusted STIRG model on generating links that exist in the constellations, we compute the average overlap between the generated links and the links in the constellations over multiple sets of generated links. This is done by computing both the proportion of links generated using the model that exist in the constellation, as well as the proportion of links of the constellation that are generated using the model. By denoting the number of links that are both generated (g) and in the constellation (c) by $l_{g;c}$, the number of links that are generated (g) but not in the constellation (n) by $l_{g;n}$ and the number of links that are generated (g) by l_g, we observe that

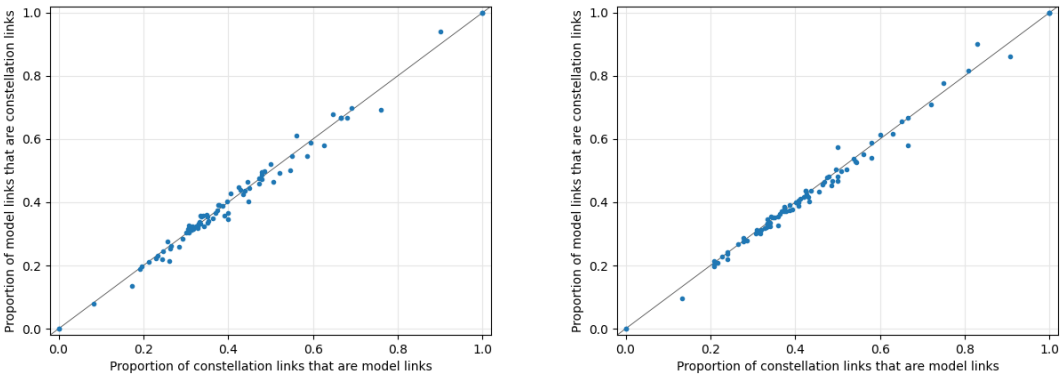
$$\mathsf{E}[l_{g;c}] + \mathsf{E}[l_{g;n}] = \mathsf{E}[l_g] = \frac{n\mathsf{E}[D]}{2} = \frac{nd}{2} \tag{11}$$

from Equation 5, where we note that $\mathsf{E}[l_g] = \sum_{i;j} p_{ij}$, and the construction of the model with $\mathsf{E}[D] = d$. Similarly, by denoting the number of links that are not generated (n) but in the constellation (c) by $l_{n;c}$ and the number of links in the constellation (c) by l_c, we have that

$$\mathsf{E}[l_{g;c}] + \mathsf{E}[l_{n;c}] = \mathsf{E}[l_c] = l_c = \frac{nd}{2} \tag{12}$$

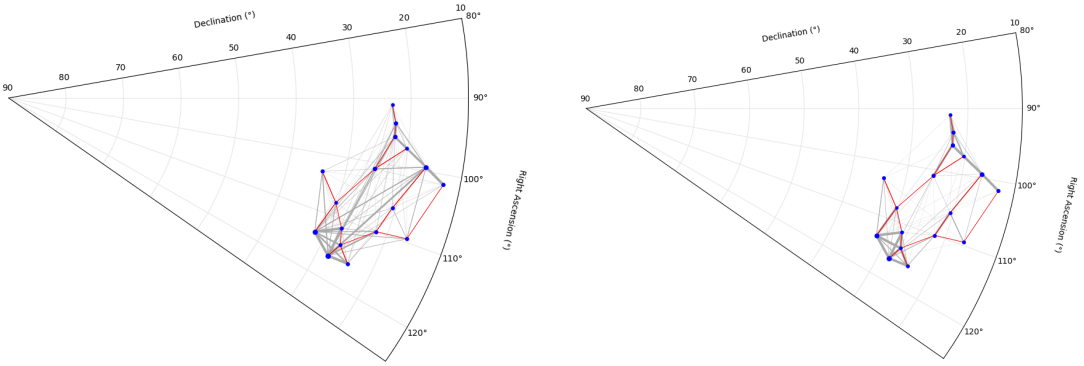
as a property of the constellation. From Equations 11 and 12 we see that $\mathsf{E}[l_{g;n}] = \mathsf{E}[l_{n;c}]$ and thus should be able to observe that $l_{g;n} \quad l_{n;c}$.

For every constellation, we generate links by using the model a multitude of times and average the proportions over the different simulations to retrieve a representative quantity for each constellation. The resulting average proportions over 25 simulations are plotted against each other in Figure 1. This confirms that $l_{g;n} \quad l_{n;c}$, which is why from here onward we denote by the proportion of overlap of a constellation the average between the proportion of constellation links that are links generated by the model and the proportion of links generated by the model that are constellation links. Figure 1 additionally shows us that for the majority of the constellations, the proportion of overlap is



(a) Attenuation function $l(t) = t^{-1}$ (b) Attenuation function $l(t) = t^{-2}$

Figure 1: The average proportion of overlap between existing constellation links and links generated by the adjusted STIRG model over 25 simulations per constellation



(a) Attenuation function $l(t) = t^{-1}$ (b) Attenuation function $l(t) = t^{-2}$

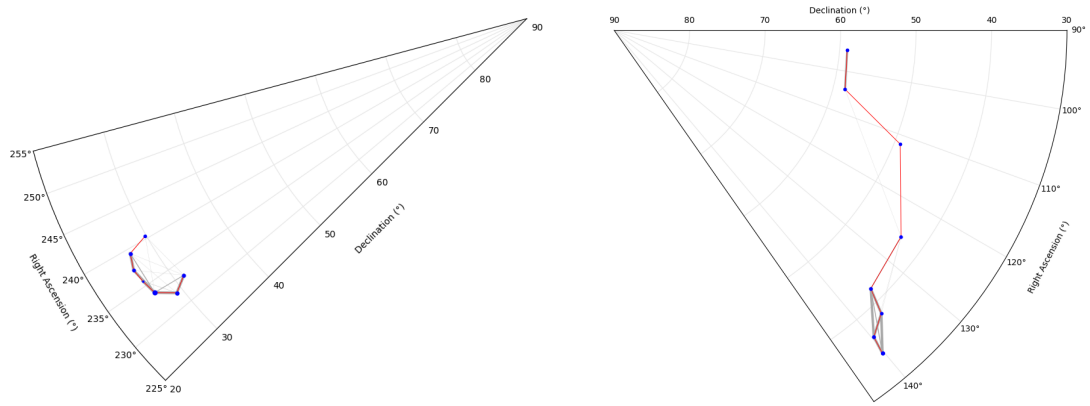
Figure 2: The Gemini constellation compared to links generated by the adjusted STIRG model

between approximately 0.2 and 0.5, which means that for the majority of constellations, at least half of the generated links are not present in the constellation. It should be noted that the constellations corresponding to a proportion of overlap of 0 are constellations consisting of a single star, and thus no links. The constellations corresponding to a proportion of overlap of 1 are constellations with two stars, and thus a single link that receives a probability of 1 according to the model.

From comparison of Figures 1a and 1b we observe that the influence of adjusting the degree a of the attenuation function $L(\ ) = l(jj\ \ jj)$ with $l(t) = t^{\ a}$ on the general performance of the model is small. Averaging the proportion of overlap over all constellations results in an average proportion of overlap of 0.411 and 0.448 for $a = 1$ and $a = 2$ respectively. In order to illustrate the effect of adjusting the attenuation function, plots of the Gemini constellation and the model results with differing attenuation functions are shown in Figure 2. The figure shows the constellations links in red, and the generated links in grey. We generated links of the constellation 25 times and for each link counted how often it was present. The thickness of the grey lines correspond to how often the links

were present. Through comparing Figure 2a and 2b the effect of the attenuation function becomes clearer.

We observe that links that have a relatively low probability in the model when using attenuation function $l(t) = t^{a}$ with $a = 1$, which correspond to the links that were only generated a few times out of 25 as seen in Figure 2a, also have an even lower probability of being generated when using $a = 2$, considering those links are barely existent at all in Figure 2b. At the same time, links that are often chosen when using $a = 1$ are chosen almost every single time in the model with $a = 2$. This is most likely because of the difference in angular distance between stars i and j compared to the angular distance between the other stars in the constellation and star i and j. Noting that in the definition of $p_{ij;1}$ in Equation 6 we find $L(\mathbf{x_i} \quad \mathbf{x_j})$ in the numerator, and $L(\mathbf{x_k} \quad \mathbf{x_j})$ and $L(\mathbf{x_k} \quad \mathbf{x_i})$ for all $k \in i; j$ in the denominator, the difference in angular distance converts to a smaller effect of attenuation in the numerator compared to the denominator. This results in higher value of $p_{ij;1}$ and thus a high connection probability p_{ij}. When increasing a, the effect of attenuation for the larger angular distances in the denominator becomes even stronger compared to the angular distance in the numerator, which means $p_{ij;1}$ and thus p_{ij} also increase compared to when we use a smaller value of a. Connection probabilities that are small with $a = 1$ decreasing further when increasing a can be explained in a similar manner. The small and large connection probabilities with $a = 1$ becoming smaller and larger respectively when increasing a is exactly the effect that we observe in Figure 2. As noted before, the general performance of the model does not change significantly from adjusting the attenuation function, which is why from now on we will solely look at results where attenuation function $l(t) = t^{2}$ is used.



(a) The Corona Borealis constellation

(b) The Lynx constellation

Figure 3: The Corona Borealis and Lynx constellations compared to links generated by the adjusted STIRG model using attenuation function $l(t) = t^{2}$

To gain further insight into the performance of the model, we also look at the Corona Borealis and Lynx constellations as shown in Figure 3. Both constellations are relatively small compared to the Gemini constellation, but differ in structure. The links in the Corona Borealis constellation all have a similar length, and the stars are only connected to the stars that are closest to them, so the model manages to generate correct links more often than not. Though the Lynx constellation also has a relatively simple structure, the model performs worse as the links in the constellation have varying lengths and there is a cluster of stars at the bottom of the constellation. The model mostly generates links between stars in the cluster, as the length between any pair of stars in the cluster is rather

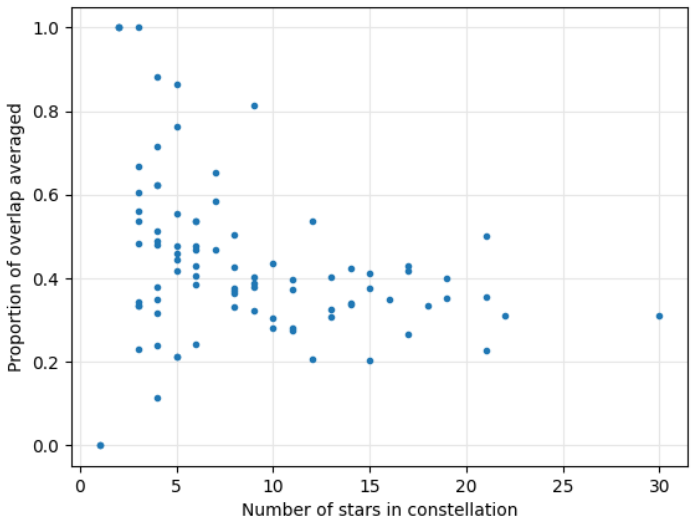


Figure 4: The number of stars in a constellation compared to the proportion of overlap between the constellation and model results with attenuation function $l(t) = t^{2}$

small, leading to the longer links in the constellation often being left out in the model. As the links between stars that are not in the same cluster are usually not generated by the model, the set of links generated rarely form a connected graph, while constellations always consist of a single, connected component. This is similar to what we see with the Gemini constellation in Figure 2b. Generally, the model often generates more links between stars that are clustered together than between stars that are not in a cluster and further apart from each other, which is expected from the construction of the model as it generally assigns a higher probability to those links.

Smaller constellations usually have less separate clusters of stars, simply because there are less stars in the constellation. For that reason, the model should theoretically perform better on smaller constellations. In Figure 4 we see that the proportion of overlap, which is the average of the proportion of constellation links that are model links and vice versa, is higher for some of the smaller constellations. This suggests that for some smaller constellation the model does perform better compared to larger constellations. However, applying the model to a constellation consisting of only a few stars does not guarantee that the generated graphs are similar to the constellation.

5 Conclusion

In this paper, we have adjusted the Signal to Interference Ratio Graph model to a random graph model that we use to generate links between stars that we know are part of the same constellation. The model computes probabilities that two stars in a constellation are connected based on the brightness of the stars and the angular distance between them weighed against the brightness of and distance to other stars in the constellation. We use the angular distance as a measure for the perceived distance between stars when observed from earth and determine the influence of the angular distance between the stars on the generated probabilities with an attenuation function, which is a type of function that

strengthens or weakens the influence of its input argument on the result of the entire equation. Considering the aim of this paper is to determine whether the random graph model we constructed implies that there is a mathematical mechanism behind the structure of constellations, we proceeded by defining a measure for the performance of the model on constructing constellations based on star data. The measure produced for this is the overlap between links of the original constellation and the links generated by the model. Using an implementation of the model in Python we generated links 25 times for each of the 88 constellations listed by the International Astronomical Union, which resulted in an average overlap per constellation of 0.411 and 0.448 when using attenuation functions $l(t) = t^{1}$ and $l(t) = t^{2}$ respectively, with a large spread in overlap regardless of the attenuation function used as seen in Figure 1. On average, the model performs slightly better for smaller constellations as seen in Figure 4. From these results we conclude that the model does not perform good enough to imply that there is a mathematical mechanism behind constellations.

6 Discussion

In this paper, we have applied our model to different sets of stars that we know are part of an existing constellation. Visual examples of links generated by the model show that the model often generates too many links between stars that are clustered together, whereas links between those star clusters are often missing. This influences the performance of the model significantly, as constellations always consist of a single connected component. Therefore the model could still be improved by, for example, increasing the interference factor of stars in the neighbourhood. Concretely, this means that we suggest increasing the value of compared to N_0, as we currently set them equal to each other. This makes it less likely that a link is present if there are many other stars nearby, thus reducing the likelihood that the model generates densely connected clusters of stars. Another way to solve the issue that the model we presented generates too many links between stars that are clustered together and too few links between stars not in the same cluster, is by noting that this issue often causes graphs generated by the model to not be connected. Considering that we already apply our model to sets of stars of which we have the prior knowledge that they are part of a constellation and noting that constellations always consist of one connected component, the model might be improved by setting the requirement that the generated links must form a connected graph.

Another possible way to study constellations using our model, or a further adjusted version of it, is to apply it to all stars visible to the naked eye in an area where we know a constellation exists. The IAU has defined the 88 constellations in terms of an area in which they lie, in total spanning the entirety of the sky, which makes it an option for defining the areas on which to separately apply the model. In that case, conclusions might be drawn about constellation without using the prior knowledge of exactly which stars belong to which constellation. What should still be taken into account is that the computation time for the model scales quadratically in the number of stars used as input, which is the reason we refrained from applying our model to all visible stars in a certain area.

Also, the relevance of the brightness of the stars when applying the model to a set of stars that we know are part of a constellation is debatable, considering that the brightest stars in a constellation do not necessarily have a higher degree than other stars in the constellation, as illustrated by for instance the star Polaris in the Ursa Minor constellation. It can thus be argued that the brightness of the stars can be left out when applying the model to a set of stars that are known to form a constellation. The brightness of stars is

mostly relevant in identifying which stars are actually part of a constellation, as those stars are usually brighter than other stars around them, which means it is a necessary inclusion in the model if it would be applied to all visible stars in a certain area.

If a model that performs well in generating constellations is successfully created, it could be interesting to connect it to the field of psychology, in order to find out if and how the model and its results could provide support in quantifying how the human mind functions in terms of visualising structures.

References

[1] Jesper Dall and Michael Christensen. Random geometric graphs. *Physical Review E*, 66(1), Jul 2002.

[2] O. Dousse, F. Baccelli, and P. Thiran. Impact of Interferences on Connectivity in Ad Hoc Networks. *IEEE/ACM Transactions on Networking*, 13(2):425–436, 2005.

[3] M.J. Dry, D.J. Navarro, A.K. Preiss, and M.D Lee. The Perceptual Organization of Point Constellations. In *Proceedings of the 31st Annual Conference of the Cognitive Science Society*, pages 1151–1156, 2009.

[4] M.A. Earl. The Spherical Trigonometry vs. Vector Analysis. `http://www.castor2.ca/07_News/headline_062515.html`, 2015. Accessed: 23-11-2020.

[5] P. Erdős and A. Rényi. On random graphs. *Publ. Math. Debrecen*, 6:290–297, 1959.

[6] M. Kang and Z. Petrášek. Random graphs: theory and applications from nature to society to the brain. *Int. Math. Nachr., Wien*, 227(227):1–24, 2014.

[7] B. King. Right Ascension & Declination: Celestial Coordinates for Beginners. `https://skyandtelescope.org/astronomy-resources/right-ascension-declination-celestial-coordinates/`, 2019. Accessed: 2020-10-22.

[8] B. McClure. What is stellar magnitude? `https://earthsky.org/astronomy-essentials/what-is-stellar-magnitude`, 2017. Accessed: 26-11-2020.

[9] International Astronomical Union. The constellations. `https://www.iau.org/public/themes/constellations/`. Accessed: 14-02-2021.

[10] M. Wenger, F. Ochsenbein, D. Egret, P. Dubois, F. Bonnarel, S. Borde, F. Genova, G. Jasniewicz, S. Laloë, S. Lesteven, and R. Monier. The simbad astronomical database-the cds reference database for astronomical objects. *Astronomy and Astrophysics Supplement Series*, 143(1):9–22, 2000.

A Python code

A.1 Implemented methods

Here we show all methods that we have implemented in Python, including the implementation of the model.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import random
import csv
from scipy import optimize

# Imports from Doina Bucur's work
from get_star_metadata import read_star_metadata, compute_magnitude
from get_constellations import read_culture


# Function copied from Doina Bucur's fit.py
def get_all_stars():
    raw_df = read_star_metadata()

    # keep the columns MAIN_ID (index), RA_d, DEC_d
    df = pd.concat([raw_df['RA_d'], raw_df['DEC_d'], ], axis=1)

    # add magnitude Magn column computed out of 3 bands (FLUX_B FLUX_V FLUX_R)
    df['Magn'] = raw_df.apply(compute_magnitude, axis=1)

    return raw_df, df


# Calculates and returns the central angle (in degrees) between two stars
def central_angle(star1, star2):
    del1 = star1['DEC_d']
    del2 = star2['DEC_d']
    d_alf = star1['RA_d'] - star2['RA_d']
    angle = np.arccos(np.sin(np.radians(del1))  np.sin(np.radians(del2))
                      + np.cos(np.radians(del1))  np.cos(np.radians(del2))
                       np.cos(np.radians(d_alf)))
    return np.rad2deg(angle)


# computes the connection probabilities based on the signal to interference
# ratio for any set of two stars, and returns a set of links based on
# those connection probabilities
def calculate_SINR_links(df, nr_of_stars, avg_degree, attenuation_degree,
                         N_0=1.0, gamma=1.0):
    # calculate connection power Pi L(xi-xj) for all pairs of stars i, j
    # and save it to connection_powers[i][j]
    connection_powers = np.zeros((df.shape[0], df.shape[0]))
    for i in range(df.shape[0]):
        for j in range(df.shape[0]):
            star1 = df.iloc[i]
            star2 = df.iloc[j]
            if i != j:
                connection_powers[i][j] = (1 / 2.512)  star1['Magn']  \
```

```
                            (1 / central_angle(star1, star2)□□attenuation_degree)

    # compute the signal to interference ratios of star i to star j
    # and star j to i, with the (default) values of N_0 and gamma,
    # and multiplies them to retrieve ~p_{ij,1}
    prob_tilde = np.zeros((len(connection_powers), len(connection_powers)))
    interference_powers_i = np.zeros((len(connection_powers),
                                      len(connection_powers)))
    interference_powers_j = np.zeros((len(connection_powers),
                                      len(connection_powers)))
    for i in range(len(connection_powers)):
        for j in range(i + 1, len(connection_powers[0])):
            # compute the sum of all other connection powers that interfere with
            # the connection from i to j and j to i respectively
            for k in range(len(connection_powers)):
                if (k != i) & (k != j):
                    interference_powers_i[i][j] += connection_powers[k][j]
                    interference_powers_j[i][j] += connection_powers[k][i]

            SINR_i = connection_powers[i][j] / (N_0 + gamma □
                                                interference_powers_i[i][j])
            SINR_j = connection_powers[j][i] / (N_0 + gamma □
                                                interference_powers_j[i][j])

            prob_tilde[i][j] = SINR_i □ SINR_j

    # define the function of which we find the fixed point:
    # 2/nd \sum min(~p_{ij,1}, C)
    def fpi_func(x):
        value = 0
        for i in range(len(prob_tilde)):
            for prob_ij in prob_tilde[i]:
                value += min(prob_ij, x)
        return 2 □ value / (nr_of_stars □ avg_degree)

    # find the fixed point, with initial value 1 using Steffensen's method
    fixedpoint = optimize.fixed_point(fpi_func, 1)

    # calculate the final probabilities p_ij = min(~p_{ij,1} / C, 1)
    # and generate links based on those probabilities
    prob_final = np.zeros((len(connection_powers), len(connection_powers)))
    links = []
    for i in range(len(prob_tilde)):
        for j in range(len(prob_tilde[i])):
            prob_final[i][j] = min(prob_tilde[i][j] / fixedpoint, 1)
            rand = random.random()
            if rand <= prob_final[i][j]:
                links.append(df.index[i])
                links.append(df.index[j])
    return links


# Plots all stars of given dataframe in the northern hemisphere
# Optional arguments for formatting, and for filtering on a more specific area
def plot_stars_N(df, ax, formatting='b.', min_RA=0, max_RA=360,
                 min_DEC=0, max_DEC=90):
```

```
    # Filter the area based on function arguments
    df_filtered_area = df[(df['RA_d'] >= min_RA) & (df['RA_d'] <= max_RA) &
                          (df['DEC_d'] >= min_DEC) & (df['DEC_d'] <= max_DEC)]
    # Plot each star, with brighter stars (having a lower magnitude)
    # appearing larger
    for row in df_filtered_area.iterrows():
        ax.plot(np.radians(row[1]['RA_d']), row[1]['DEC_d'], formatting,
                           markersize=12 - (row[1]['Magn'] + 1.2))


# Plots all the connections between stars given by the links argument
# Requires all stars in the list links to be in the provided dataframe df
# Optional argument for formatting
def plot_constellation_N(df, ax, links, formatting='k-', alpha=1.0):
    # Stops if links has uneven length, as stars should be given in pairs
    if not len(links) % 2 == 0:
        print('Uneven number of stars in constellation links')
        return
    # For every two stars provided by links, plot a line between them
    for i in range(0, len(links), 2):
        stars = df.loc[[links[i], links[i + 1]], :]
        ax.plot(np.radians(stars['RA_d']), stars['DEC_d'], formatting,
                alpha, linewidth=0.7)


# Prepare the polar plot in which to plot the stars and constellations
def prep_polar_plot():
    ax = plt.subplot(1, 1, 1, projection='polar')
    ax.set_theta_direction(-1)
    ax.set_theta_zero_location('N')
    ax.set_ylim([0, 90])
    ax.grid(b=bool, which='major', axis='both', color='0.90')
    return ax


# return the list of stars given a list of constellation links
def stars_from_constellation(links):
    return list(set(links))


# compute and return the proportion of constellation links that are also
# links generated by the model as well as the proportion of links
# generated by the model that are also constellation links
# from a list of constellation links and a list of generated links
def calculate_overlap(constellation_links, SINR_links):
    nr_constellation_links = len(constellation_links) / 2
    nr_SINR_links = len(SINR_links) / 2
    nr_overlap_links = 0
    for i in range(0, len(constellation_links), 2):
        for j in range(i, len(SINR_links), 2):
            if (((constellation_links[i] == SINR_links[j]) &
                 (constellation_links[i + 1] == SINR_links[j + 1])) |
                ((constellation_links[i] == SINR_links[j + 1]) &
                 (constellation_links[i + 1] == SINR_links[j]))):
                nr_overlap_links += 1
```

```
    proportion_const_to_SINR = nr_overlap_links / nr_constellation_links
    if nr_SINR_links == 0:
        proportion_SINR_to_const = 0
    else:
        proportion_SINR_to_const = nr_overlap_links / nr_SINR_links

    return proportion_const_to_SINR, proportion_SINR_to_const
```

A.2 Example code for generating results

Here we list two code examples using the implemented methods shown above. An example for generating the results and representing them in plots as used for Figures 1 and 4 is provided first:

```
# puts all relevant star data in dataframe df
raw_df, df = get_all_stars()

# select a culture
culture = 'IAU'

# read the constellations data
constellations = read_culture('data/' + culture + '.txt')

# provide the degree 'a' of the attenuation function l(t) = t^(-a)
# to be used
attenuation_degree = 1

with open('overlap_values.csv', 'w', newline='') as csvfile:
    writer = csv.writer(csvfile, delimiter=',',
                        quotechar='"')
    writer.writerow(['Constellation', 'Number of stars',
                     'Average overlap const to SINR',
                     'Average overlap SINR to const'])

    nr_stars_list = []
    overlap_results_x = []
    overlap_results_y = []
    overlap_results = []
    overall_avg_overlap = 0
    # for each constellation, generate links a number of times
    # and compute the average overlap
    for constellation in constellations.keys():
        # setup a dataframe containing the stars of the constellation
        constellation_links = constellations[constellation]
        stars_list = stars_from_constellation(constellations[constellation])
        df_constellation = df.loc[stars_list, :]

        # determine the average degree and number of stars
        # of the constellation
        average_degree = len(constellation_links) / len(stars_list)
        number_of_stars = len(stars_list)
        nr_stars_list.append(number_of_stars)

        # set the number of iterations over which to average
        #the overlap per constellation
        nr_iterations = 25
```

```
        total_const_to_SINR = 0
        total_SINR_to_const = 0

        # generate links and compute the overlap between the
        # generated and constellation links
        for i in range(nr_iterations):
            SINR_links_constellation = \
                calculate_SINR_links(df_constellation, number_of_stars,
                                     average_degree, attenuation_degree)
            overlap = calculate_overlap(constellation_links,
                                        SINR_links_constellation)
            total_const_to_SINR += overlap[0]
            total_SINR_to_const += overlap[1]

        # compute the averages
        avg_const_to_SINR = total_const_to_SINR/nr_iterations
        avg_SINR_to_const = total_SINR_to_const/nr_iterations
        avg_overlap = (avg_const_to_SINR + avg_SINR_to_const)/2
        print(constellation)
        print((avg_const_to_SINR, avg_SINR_to_const))
        overlap_results_x.append(avg_const_to_SINR)
        overlap_results_y.append(avg_SINR_to_const)
        overlap_results.append(avg_overlap)
        overall_avg_overlap += avg_overlap

        # write the results to the csv file
        writer.writerow([constellation, number_of_stars,
                         avg_const_to_SINR, avg_SINR_to_const])
    overall_avg_overlap = overall_avg_overlap / len(constellations.keys())
print(overlap_results_x, overlap_results_y)
print('overall average overlap = ' + str(overall_avg_overlap))

# plot the proportions of overlap against each other
ax = plt.subplot(1, 1, 1)
ax.plot([0, 1], [0, 1], 'k', linewidth=0.6, alpha=0.7,
        transform=ax.transAxes)
ax.plot(overlap_results_x, overlap_results_y, '.')
ax.set_xlim([-0.02, 1.02])
ax.set_ylim([-0.02, 1.02])
ax.set_xlabel('Proportion of constellation links that are model links')
ax.set_ylabel('Proportion of model links that are constellation links')
ax.grid(b=bool, which='major', axis='both', color='0.90')

plt.show()

# plot the nr of stars to the average proportion of overlap
ax = plt.subplot(1, 1, 1)
ax.plot(nr_stars_list, overlap_results, '.')
ax.set_xlabel('Number of stars in constellation')
ax.set_ylabel('Proportion of overlap averaged')
ax.grid(b=bool, which='major', axis='both', color='0.90')

plt.show()
```

An example for generating the polar plots as seen in Figures 2 and 3, as used specifically to plot the Corona Borealis constellation as seen in Figure 3a is also provided:

```
# puts all relevant star data in dataframe df
raw_df, df = get_all_stars()

# select a culture and constellation
culture = 'IAU'
constellation_name = 'Corona Borealis'

# read the constellations links and put all stars of the constellation
# in a dataframe
constellations = read_culture('data/' + culture + '.txt')
constellation_links = constellations[constellation_name]
stars_list = stars_from_constellation(constellations[constellation_name])
df_constellation = df.loc[stars_list, :]

average_degree = len(constellation_links) / len(stars_list)
number_of_stars = len(stars_list)

# provide the degree 'a' of the attenuation function l(t) = t^(-a) to be used
attenuation_degree = 1

# Code for generating constellations + model results
# Plot settings:
main_plot = prep_polar_plot()
main_plot.set_ylim([20, 90])
main_plot.set_thetamin(225)
main_plot.set_thetamax(255)
main_plot.text(np.deg2rad(240), 15, 'Right Ascension ( )',
               rotation=-240-180, ha='center', va='center')
main_plot.text(np.deg2rad(215), 55, 'Declination ( )',
               rotation=45, ha='center', va='center')
main_plot.set_title('Corona Borealis constellation, attenuation degree 1')
plt.gca().invert_yaxis()

# generate links for the constellation a certain number of times,
# and for every possible link, count how often it is generated
nr_iterations = 25
occurrences_of_links = np.zeros((len(stars_list), len(stars_list)))
for i in range(nr_iterations):
    SINR_links_constellation = \
            calculate_SINR_links(df_constellation, number_of_stars,
                                 average_degree, attenuation_degree)
    for j in range(0, len(SINR_links_constellation), 2):
        for k in range(len(stars_list)):
            if SINR_links_constellation[j] == stars_list[k]:
                l = stars_list.index(SINR_links_constellation[j+1])
                occurrences_of_links[k][l] += 1

# plot the generated links with a thickness based on how often they are generated
for i in range(len(occurrences_of_links)):
    for j in range(i, len(occurrences_of_links)):
        if occurrences_of_links[i][j] != 0:
            stars = df.loc[[stars_list[i], stars_list[j]], :]
            main_plot.plot(np.radians(stars['RA_d']), stars['DEC_d'], 'darkgrey',
```

```
                                linewidth=3*occurrences_of_links[i][j]/nr_iterations)

    # plot the stars and constellation
    plot_constellation_N(df, main_plot, constellations[constellation_name], 'r-', 0.2)
    plot_stars_N(df_constellation, main_plot)

    plt.show()
```