Hiding Secrets in Plain Text

by

Konrad Komorowski (CHU)

Fourth-year undergraduate project in
Group F, 2013/2014

I hereby declare that, except where specifically indicated, the work submitted herein is my own original work.

Signed: ______________________________ Date: _______________

Technical abstract

Hiding Secrets in Plain Text

by

Konrad Komorowski (CHU)

Fourth-year undergraduate project in
Group F, 2013/2014

Objective

The problem addressed in this project is constructing a *stegosystem* that allows to broadcast a secret message as a paragraph of innocuous English text. The output text should appear to be randomly drawn from the set of all grammatically correct English sentences. The original message needs to be recoverable in full from the stegotext. Secrecy is provided by a symmetric key which allows both encryption and decryption of the original message.

Achievements

The work done can be split into two logical parts: statistical natural language modelling and construction of the secure stegosystem.

Language model

A model of natural language as a stochastic process is created. It is based on N-gram counts from a training corpus. An N-gram is a sequence of N contiguous tokens in a fragment of text. Tokens can be words, punctuation, special characters, etc. The source of counts is the Google Books N-grams Corpus.

The model gives the conditional probability of a token given the preceding text. To be computationally tractable, it only considers a history of $N-1$ tokens. If it is not possible to make a reliable estimate with context of size $N-1$, the model is *backed-off*, i.e. its history is reduced by 1 token. This can be repeated recursively until the context is empty. A *back-off tree* is presented as an illustration of how estimates based on different history sizes are combined to form the full model.

Stegosystem

First, a simple stegosystem is designed. It allows to convert any *plaintext* sequence into another *stegotext* sequence that appears to have been generated by an arbitrary stochastic process. Statistics of the process by which the plaintext was generated need to be known.

It is possible to recover the plaintext from the stegotext, up to the limitation that the original sequence may be followed by random symbols.

Then, secrecy is introduced to create a secure stegosystem. A symmetric key is used to encipher plaintext before transforming it to stegotext. Only the knowledge of the key enables correct recovery of the plaintext. Moreover, supplying an incorrect key results in recovering a randomly generated plaintext sequence. This gives deniable encryption – in principle it makes it impossible to verify if the supplied key is correct or not.

Results

Table A shows how the secure stegosystem can be used to generate stegotext and recover plaintext from it. It also demonstrates the deniable encryption property by using an incorrect key to recover the plaintext.

plaintext	Come in the morning with the documents.
key	Light rain on Monday morning.
stegotext	The Hurrian pantheon of the gods, o king, who is in turn responsible for the conditions at the surface of the whole Earth, we could not talk to her about what she had done.
recovered plaintext	Come in the morning with the documents. The grocer takes the form of a human, or a full-time basis. With this
false key	Early light rain or drizzle will soon die out.
recovered false plaintext	Jennings, H. S. (editor). An abstruse and learned specialist who finds that he has been to the west of the entrance to the vagina. There is a suggestion that the Earth might be blessed.

Table A: Example usage of the secure stegosystem with the English language model. Note that only the plaintext and the keys were chosen by the author, the rest is generated by the system implemented in this project.

Conclusions

A large database of statistics of English text, the Google Books N-grams corpus, was processed to create a model of text as the output of a stochastic process. The language model was used to construct a stegosystem, which allows to encrypt a secret message using a symmetric key and broadcast it as a paragraph of innocuous English text. Elements of the system allow other tasks to be performed as well, most importantly text compression and generation of random sentences.

Acknowledgements

I would like to express my deepest thanks to my supervisor Dr. Jossy Sayir for his invaluable and unceasing help, guidance and support. Countless hours spent discussing the project and his continuous availability ensured that not for a moment I doubted in the ultimate success of the project. Without his utmost dedication this work simply would not have been possible.

I would also like to appreciate the crucial role played by the computer officer Dr. Phill Richardson, who provided and custom configured all necessary computing resources. The completion of the project owes to his aid as well.

Contents

1 Introduction

1.1 Aim

The aim of the project is to create a communications system that will allow to broadcast a secret message enciphered with a private key as a paragraph of English text. The system should also translate this paragraph back into the original message using the same key.

The sentences should appear to be randomly drawn from the set of all possible English sentences, in other words seem innocuous for an enemy monitoring communications between two parties.

The introduction of a private key means that the secrecy of the message does not rely on the enemy's unfamiliarity with the system, but on a private piece of information exchanged between the two parties.

1.2 Overview of steganography

The project falls in the field of *steganography*, which can be defined as follows:

> Steganography is the art and science of encoding hidden messages in such a way that no one, apart from the sender and intended recipient, suspects the existence of the message. [1]

An example stegosystem providing some *innocuousness* and *secrecy* can be constructed in the following way:

> A message, the *plaintext*, may be first encrypted by traditional means, producing a *ciphertext*. Then, an innocuous *covertext* is modified in some way so as to contain the *ciphertext*, resulting in the *stegotext*. [1]

There are various ways in which ciphertext may be embedded in covertext. Below are only some examples:

1. Lines of covertext may be broken in such a way that the first words of every line form ciphertext. Note that such stegotext cannot be reformatted – the font and page sizes need to stay constant so that the line breaks occur at the same places.

2. Standard ASCII characters can be replaced with their redundant Unicode lookalikes in a specific way according to the ciphertext. For this to work the stegotext needs to be transmitted using Unicode.

3. Steganography is not limited to text. Covertext may be a digital *image*, noise in which can be modified in a certain way according to the ciphertext. For successful transmission a rasterised version of the *stegoimage* has to be transmitted *exactly* (i.e. digitally, not in print).

The methods outlined above provide only some degree of innocuousness. They largely rely on the enemy's unfamiliarity with the system and may be detected using statistical tests, e.g. identifying unusual distributions of Unicode characters or image noise.

1.3 Approach taken

In this project I will take an information theoretic approach to steganography. Plaintext and covertext will be modelled by stochastic processes, whose statistics need to be learnt. I will then generate stegotext directly from plaintext in such a way that stegotext will be statistically indistinguishable from covertext, but it will still be possible to recover plaintext from it.

Sections 2 to 4 show how natural language text can be modelled as the output of a stochastic process and how to efficiently estimate the statistics of this process from a corpus of training data. Sections 5 and 6 describe how to construct a stegosystem given the statistics of the aforementioned processes.

2 Statistical natural language models

I will introduce basic principles of statistical natural language models. I will start by interpreting text as the output of a stochastic process and then apply approximations to make evaluating sentence probabilities computationally tractable. This section is mainly a review of [2–4].

2.1 Text as the output of a stochastic process

A string $\mathbf{w}$ can be written as a sequence of L *tokens* $\mathbf{w} = [w_1, \dots, w_L] \equiv w_1^L$. Tokens are atomic elements of text such as words or punctuation marks. w_l corresponds to the index of the token; for V distinct tokens it takes values in $\{1, \dots, V\}$. $\mathbf{w}$ can be regarded as a realisation of an L-dimensional vector $\mathbf{W}$ of jointly distributed random variables W_l. According to the chain rule of probability, we can factorise $P(\mathbf{w})$ as follows

$$P(\mathbf{w}) = \prod_{l=1}^{L} P(w_l | w_1^{l-1}). \tag{1}$$

Even though Eq. 1 gives a convenient formulation of language as a stochastic process, it is computationally intractable. If the conditional probabilities are stored in tables, the conditional probability table of the lth token has V^l entries. The total size of all tables becomes prohibitively large for even small L. The usual solution is to restrict the size of the context to $N-1$ previous tokens, resulting in an approximate N-gram language model

$$P(\mathbf{w}) \approx P_{N\text{-gram}}(\mathbf{w}) \equiv \prod_{l=1}^{L} P(w_l | w_{l-N+1}^{l-1}). \tag{2}$$

2.2 Estimating ML parameters

The ML (maximum likelihood) estimate of the conditional probability of w_l in an N-gram model is

$$\hat{P}(w_l | w_{l-N+1}^{l-1}) = \frac{f(w_{l-N+1}^{l})}{f(w_{l-N+1}^{l-1})}, \tag{3}$$

where $f(\cdot)$ is the number of occurrences of a particular sequence of tokens in training data.

A problem with Eq. 3 is that most N-gram counts will be zero. For example, a 5-gram model with $V \approx 10^6$ will have $V^N \approx (10^6)^5 = 10^{30}$ distinct N-grams. For every possible 5-gram sequence to occur at least once, training data would need to consist of more than 10^{30} tokens. Google Books N-gram Corpus [5], currently the largest available dataset, is based on 4.7×10^8 tokens. As it turns out, the vast majority of grammatically correct sentences has zero probability.

2.3 Limiting corpus size, discounting and back-off

A standard way of dealing with this problem is to introduce *discounting* and *back-off* to the model. In addition, N-grams with counts below a threshold C are sometimes discarded from the corpus to conserve storage space. The conditional distribution of the lth token becomes

$$P(w_l|w_{l-N+1}^{l-1}) = \begin{cases} d(w_{l-N+1}^{l})\,\frac{f\left(w_{l-N+1}^{l}\right)}{f\left(w_{l-N+1}^{l-1}\right)} & \text{if } f(w_{l-N+1}^{l}) \geq C, \\ \alpha(w_{l-N+1}^{l})\; P(w_l|w_{l-N+2}^{l-1}) & \text{otherwise,} \end{cases} \tag{4}$$

where $d(\cdot)$ and $\alpha(\cdot)$ return the discount and back-off weights – scaling factors used to ensure that Eq. 4 gives a valid p.m.f. Intuitively, they reduce the mass assigned to observed N-grams to make space for estimates of order $N-1$ and lower.

$d(\cdot)$ and $\alpha(\cdot)$ are hard to choose or compute – there exist many different schemes motivated by linguistics and statistics. Examples include Good-Turing smoothing [6], Kneser-Ney smoothing [7], or stupid back-off [8] – all exhibit various levels of complexity. In addition, the last one does not even return a valid p.m.f., but this is judged acceptable for strict scoring applications.

3 Processing and storing N-gram counts

I have shown that under certain assumptions, N-gram counts provide a practically adequate statistic for evaluating the probabilities of natural language text. In this section I will describe processing and efficient storage of these counts.

I will first describe two sources of data that I investigated, Google Books N-gram Corpus and Corpus of Contemporary American English N-grams, motivating why I chose the former. I will then explain in detail the processing required to make the N-gram counts corpus fit for a steganographic application.

Firstly, all N-grams that contain digits are removed from the corpus. This is because they are frequently associated with illegible fragments of text from various data tables. Token strings of the remaining N-grams are then transliterated from Unicode to 7-bit ASCII, followed by conversion to lowercase letters and filtering only a set of allowed characters. The desired result is to remove noise from the system and make token representations unambiguous. Following this, extended tokens like *[it's]* are split into multiple base tokens such as *[it]*, *[']* and *[s]*. The aim is to ensure that a fragment of text has a unique sequence of corresponding tokens. An algorithm to induce counts of base tokens from counts of the extended tokens is illustrated. Lastly, a procedure to enforce the N-gram counts to be self-consistent is described.

3.1 Sources of natural language statistical data

3.1.1 Google Books N-gram Corpus

Google has published N-gram counts, for up to $N = 5$, compiled from 4.5 million scanned English books that contain 468 million tokens [5, 9]. Their counts are given over different years when the books were published. The cutoff for including an N-gram in the corpus is 40 occurrences in all books over all years. The tokens include part-of-speech (POS) annotations for words.

Currently, there are two version of the corpus – 2009 and 2012. N-grams from the second version include special tokens for the beginning and end of sentence markers and cannot span sentence boundaries. Sentences across pages are detected. The first version of the corpus does not have these markers and the N-grams can span multiple sentences, but not pages. For higher accuracy, I decided to use the newer corpus.

All N-grams are stored in plain text files in the following format:

```
ngram TAB year TAB match_count TAB volume_count NEWLINE
```

where the `ngram` field is tab-separated into N token strings. `match_count` gives the total number of occurrences of an N-gram in the whole corpus in a given year, and `volume_count` tells how many different books published in that year it was found in.

Tokens can be pure plain text strings (e.g. `burnt`), part-of-speech annotated strings (e.g. `burnt_VERB`) or just the POS tags (e.g. `_VERB_`). The first case corresponds to the existence of a particular string regardless of its role in the sentence. These are then broken down into finer categories, which indicate the syntactic role of a particular token, producing POS-annotated strings. The last class of tokens allows us to answer more general questions, for example the count of `he _VERB_` tells us how many times the word *he* is followed by any verb.

N-grams up to $N = 3$ are constructed by mixing tokens of all kinds, making it possible to ask every possible question about the counts. For $N > 3$ there are some restrictions to limit combinatorial explosion of the size of the corpus. But N-grams consisting of plain string tokens are always available for every order N.

In the stegosystem, there can be no ambiguity as to the identity of the token. The presence of POS tags in a sentence would be a very strong indication that the stegosystem was used to generate the text, so innocuousness would be compromised. As a result, we are forced to use the unannotated tokens. Thus we operate on a less precise language model, where grammatically incorrect sentences are assigned relatively higher probabilities.

3.1.2 Corpus of Contemporary American English N-grams

Mark Davies has compiled N-gram frequencies [10] from the Corpus of Contemporary American English (COCA) [11]. COCA is based on 0.2 million texts that contain a total of 440 million tokens. The tokens include POS-annotated words or punctuation, but not sentence markers. Free version of the database contains 1 million most frequent N-grams for each order N up to 5. Paid version has no cutoff and has counts of all N-grams up to $N = 4$.

Prior to processing Google Books data, I experimented with using the free version of COCA N-grams to create the language model. I finally decided to use the Google Books N-grams Corpus because of free access its full version and the fact that it contains counts up to $N = 5$. Since COCA N-grams are not a part of the final outcome of the project, I will not be discussing their processing or storage.

3.2 Normalising tokens

3.2.1 Discarding N-grams containing digits

The Google Books N-gram Corpus is based on many books, some of which include data tables. As a result, they contain a large number of N-grams consisting of just numbers and punctuation marks. Randomly generated text would sometimes include long sections of those. To avoid such a possibility, I decided to create a second version of the corpus where N-grams containing digits are discarded. The trade-off is that most dates like *July 4, 1776* will no longer be a part of the language.

3.2.2 Normalising tokens using the `unidecode` package

Google Books team made the best effort to represent each character of a token as precisely as possible using the full range of Unicode characters. This may be problematic – equivalent words may be written in multiple ways and some characters will be impossible to represent using certain media. In addition, interpretation of text will be very sensitive to encoding – not only the character glyph, but its underlying Unicode representation will matter.

To mitigate these problems, the tokens are *normalised.* The first step is to transliterate Unicode text into plain 7-bit ASCII using using the `unidecode` Python package [12]. Following this, uppercase letters are converted to lowercase. Finally, the tokens are filtered to contain only characters from Table 1.

class	characters
lowercase letters	`a b c d e f g h i j k l m n o p q r s t u v w x y z`
digits	`0 1 2 3 4 5 6 7 8 9`
punctuation	`! " # $ % & ´ ( ) * + , - . / : ; < = > ? @ [ \ ] ^ _ ‘ { | } ~`

Table 1: Characters that can be a part of a normalised token.

Table 2 shows examples of the normalisation procedure. Note that some tokens are normalised to an empty string – this is not a problem since they will be simply ignored. However, `unidecode` sometimes transliterates special symbols to a question mark in square brackets, i.e. *[?].* Ideally these situations should be detected and handled using an empty string, but they are rare enough not to affect the model in a considerable way if ignored. Also, despite being understandable, the normalisation of *Universitätsstraße* is strictly wrong – in German *ä* should be transliterated to *ae* [13].

unnormalised	normalised
Ægean	aegean
œuvre	oeuvre
£	ps
Universitätsstraße	universitatsstrasse
©	(c)
≤	<=
♣	*empty string*
≾	[?]

Table 2: Example tokens containing special characters before and after normalisation.

3.2.3 Exploding tokens by punctuation

Table 3 shows counts of three chosen N-grams: *[it's]*, *[it 's]* and *[it ' s].* Even though they consist of respectively 1, 2 and 3 tokens, in plain text they all should be formatted in the same way – as *it's.* This introduces ambiguity – if we observe *it's* we do not know if we should interpret it as a 1-, 2- or 3-gram.

N	w_1	w_2	w_3	$f(w_1^N)$
1	it's			37 406
2	it	's		78 658 875
3	it	'	s	7 060 268

Table 3: Ways in which *it's* is stored in the Google Books N-gram Corpus (after normalisation).

The corpus needs to be processed so that all entries from Table 3 are absorbed into a single N-gram. It can be achieved through *exploding tokens by punctuation.* Let us classify tokens into two groups: *extended tokens* and *base tokens.* A base token can consist of any number of lowercase letters and digits (i.e. alphanumeric characters) or a single punctuation mark. An extended token is a concatenation of base tokens, where two alphanumeric base tokens cannot appear next to each other. Exploding a token by punctuation is defined as finding the corresponding base token sequence of an extended token.

extended token	base tokens
e.g.	e . g .
it's	it ' s
mr.	mr .
yesterday).	yesterday) .
"great"	" great "
...	. . .
<=	< =

Table 4: Example extended tokens with their corresponding base token sequences.

If we explode tokens in Table 3, all N-grams will correspond to the same sequence of base tokens: *[it ' s]*. More generally, any phonological word that can be ambiguously split into multiple tokens because of internal punctuation can be uniquely identified by its sequence of base tokens.

3.3 Induced counts of base N-grams

The Google Books N-gram Corpus supplies counts of 1- to 5-grams of unnormalised extended tokens. In order to be able to interpret and generate plain text in an unambiguous way, we need to operate on tokens that are normalised and exploded by punctuation. We know how to apply normalisation and explosion to plain text, but we do not know the statistics of the resulting tokens – we do not have N-gram counts from training data processed in this way. We need to induce such counts from the Google Books N-gram Corpus.

3.3.1 Extended and base N-grams

Extended N-gram will refer to a sequence of N unnormalised extended tokens (as stored in the Google Books N-gram Corpus). *Base N-gram* will refer to a sequence of N normalised base tokens (counts of which we wish to know). See Figure 1 for examples of both.

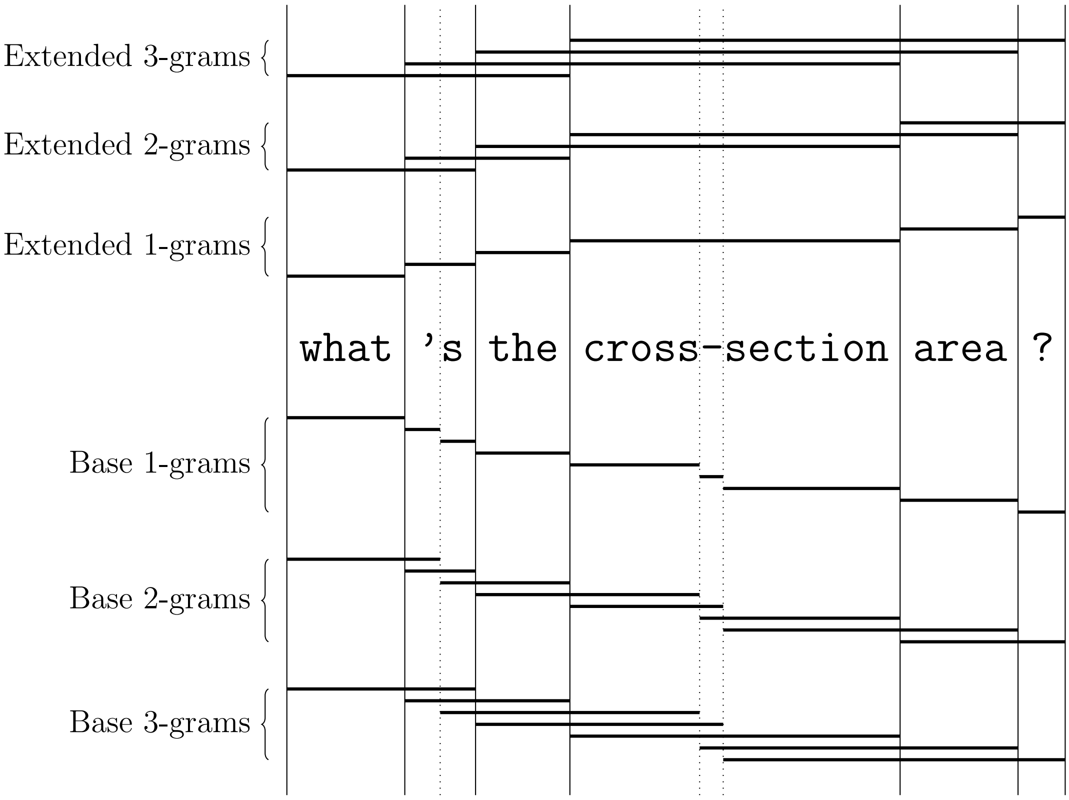

Figure 1: Finding extended and base N-grams in the fragment of text *what's the cross-section area?* Solid vertical lines denote boundaries between extended tokens, dotted vertical lines – boundaries between base tokens. All N-grams up to $N = 3$ are shown in the figure. Each is represented by a single thick horizontal segment. Extended N-grams are located above text, base N-grams – below it.

Note that extended token boundaries are chosen by the Google Books team using a custom rule-based system [14], not necessarily in a way consistent for the same phonological word. Figure 1 shows an example way of how token boundaries could be found in a sentence. Base token boundaries are created using the method from Section 3.2.3.

3.3.2 Inducing counts from extended 1-grams

We know the counts of all extended N-grams up to a certain order and wish to know the counts of all base N-grams up to the same order. Both sets of counts are on the same text.

Looking at Figure 1, it is possible to state a unique correspondence between extended 1-grams and some base N-grams. For example, the extended 1-gram *[cross-section]* coincides with and only with the set of base 1-grams *[cross]*, *[-]*, *[section]*, base 2-grams *[cross -]*, *[- section]* and base 3-gram *[cross - section]*. Let us call the elements of this set *induced*

N-grams. Formally, induced N-grams are base N-grams that can be identified in the base token sequence of an extended 1-gram.

Any time we observe counts of an extended 1-gram, we know that its induced N-grams occurred the same number of times. It is not a problem if we end up with multiple counts of the same induced N-gram. It means that this base N-gram occurs in many contexts and its counts need to be cumulated. Duplicates can come from the same extended 1-gram or different extended 1-grams.

Take as an example two extended 1-grams that both contain the base 1-gram *[section]*: *[cross-section]* with count 42 and simply *[section]* with count 173. The final count of the base 1-gram *[section]* will be $42 + 173 = 215$.

3.3.3 Inducing counts from extended N-grams ($N > 1$)

Let us introduce more nomenclature. An *extended M-gram* is a proper substring over the extended tokens of an extended N-gram. For example, the extended N-gram *['s the cross-section]* has extended M-grams *['s]*, *[the]*, *[cross-section]*, *['s the]* and *[the cross-section]*.

Inducing counts from extended N-grams for $N > 1$ is similar to the case when $N = 1$. An equivalent sequence of base tokens has to be constructed as well, but *not all* possible induced N-grams can be selected to be counted. If a base N-gram can be also induced from an extended M-gram, it *cannot* be selected. If we did select it while processing the extended N-gram, it would be double counted – we are guaranteed to come across its count again when processing the offending M-gram.

The above restriction can be easily stated in an equivalent way. A base N-gram induced from an extended N-gram has to contain at least one base token from the first and last extended tokens of the extended N-gram. This is how we ensure that it cannot be induced from an extended M-gram.

Using the example from the beginning of the section, *['s the cross-section]* corresponds to the base token sequence *[' s the cross - section]*. The following induced N-grams can be selected from it: 3-gram *[s the cross]*, 4-grams *[' s the cross]*, *[s the cross -]* and 5-grams *[' s the cross -]*, *[s the cross - section]*. In all examples, base tokens that belong to the first or last extended token of the extended N-gram are underlined.

3.4 Counts consistency

3.4.1 Definition

N-grams are defined to be *left counts consistent* if

$$\forall_{w_1^N} \;\; f(w_1^N) \geq \sum_{w_{N+1}} f(w_1^{N+1}), \tag{5}$$

and *right counts consistent* if

$$\forall_{w_1^N} \quad f(w_1^N) \geq \sum_{w_0} f(w_0^N). \tag{6}$$

N-grams are *counts consistent* if they are both left and right counts consistent. If there is no cutoff for including an N-gram in the corpus, the equations become equalities.

Eqs. 5 and 6 are easy to interpret. Left counts consistency ensures that if we add the counts of all (N+1)-grams created by appending a token to the end of an N-gram, their sum will not be larger than the count of the N-gram. Right counts consistency is analogous. Obviously, N-grams that are correctly counted will be counts consistent.

3.4.2 Ensuring counts consistency

The Google Books N-grams Corpus is not counts consistent. This may be caused by a bug in my script that processed N-gram counts on-the-fly from Google servers. It skipped the last N-gram from each of the 2892 files that the corpus is split into.[1] The corpus may also be inconsistent itself, for example due to the failure of some jobs during its distributed creation.

Counts consistency is assumed in some parts of the stegosystem, for example in $\beta-\gamma$ estimation in Section 4.3. Since inconsistencies are very rare (in the order of 10^3 for the whole corpus of about 1.5×10^9 base N-grams), they can be forcefully corrected without visibly distorting information in the counts.

The highest order N-grams are asserted to be counts consistent, since there are no (N+1)-grams to check with. In general, N-grams are made counts consistent in the following way:

1. (N+1)-gram counts are *left integrated.* Last token is dropped from each (N+1)-gram to create a *quasi*-N-gram. The counts of all identical *quasi*-N-grams are cumulated.

2. Left counts consistent N-grams are created by *maximising* N-gram and *quasi*-N-gram counts. Maximisation is performed as follows: both tables are simultaneously iterated over, starting from the first row. Whenever a *quasi*-N-gram has a higher count than the same N-gram, the N-gram count is updated. If there is *quasi*-N-gram without a corresponding N-gram, a new N-gram is created. Otherwise, the original N-grams are copied.

3. Final counts consistent N-grams are created in a similar way. This time, *right integrated* (N+1)-grams are maximised with left counts consistent N-grams from the previous step.

[1] This script run for 4 weeks and a special permission was obtained from the departmental computer operators to process 1.5 TB of incoming raw data over the Internet. Fixing this retroactively was judged not worth the effort.

Steps 2-3 can be performed simultaneously, i.e. jointly maximising all three tables of N-gram counts. In addition, since removing the last token does not change ordering, left integrated (N+1)-grams can be created *in place* – identical *quasi*-N-grams will be next to each other in the (N+1)-grams table. However, creating right integrated (N+1)-grams requires expensive sorting of the whole *quasi*-N-grams table before cumulating them – ordering is not preserved after removing the first token.

Creating count consistent N-grams requires counts consistent (N+1)-grams. So N-gram tables need to be created sequentially from the highest N down to 1.

4 Language model

In this section I will describe the language model created for the purpose of the stegosystem. For arithmetic coding, which is the backbone of the stegosystem, it is necessary to have a model that returns the probability of a token given its predecessors. Moreover, the tokens have to be ordered so that each corresponds to a unique subinterval of $[0, 1)$ of size equal to the conditional probability of the token.

I will first state all the requirements formally and analyse their implications. I will then propose a model where all tokens are arranged in an N level *back-off tree*. The tree allows efficient calculation of the conditional probability interval for a token without having to explicitly compute intervals of all other tokens.

4.1 Requirements

A language model fit for steganographic application using the interval algorithm needs to satisfy certain requirements:

1. Every sequence of tokens has non-zero probability.
2. Every token extending any sequence is assigned a contiguous probability region.
3. Formatted sequence of tokens uniquely identifies the tokens.

Requirement 1 warrants the use of back-off. Consider the counts from Tables 5 and 6. In the 5-gram model, only 5 tokens are explicitly allowed to follow *[the esteem of many]* and their total count is 542. If we used the ML model from Eq. 3 to calculate their probabilities, we would assign zero probability to all tokens other than those in column w_5 of Table 5. Thus, we would account for only $\frac{542}{1146} \approx 0.47$ of the probability mass and also discard the vast majority of tokens. The former problem can be mitigated by upscaling the probabilities, but the latter has no solution other than back-off.

index	w_1	w_2	w_3	w_4	w_5	$f(w_1^5)$
400 000 787	the	esteem	of	many	,	143
400 000 788	the	esteem	of	many	.	56
400 000 789	the	esteem	of	many	friends	66
400 000 790	the	esteem	of	many	of	237
400 000 791	the	esteem	of	many	who	40

Table 5: Counts of all 5-grams beginning with *[the esteem of many]*.

index	w_1	w_2	w_3	w_4	$f(w_1^4)$
456 820 272	the	esteem	of	many	1146

Table 6: Count of *[the esteem of many]* 4-gram.

Requirement 3 greatly complicates the language model. If arithmetic coding is only used to compress text, it is possible to augment the sequence during encoding by inserting *explicit* back-off symbols. Their purpose is to indicate during decoding that the next token's probability is to be evaluated using a lower-order model. But in a steganographic scenario, the usage of a back-off symbol would compromise innocuousness of the system. Therefore an *implicit* back-off scheme is needed.

The problem with implicit back-off and using Eq. 4 *exactly* is that if a token has non-zero probability in e.g. a 5-gram model, by consistency it is also included in all lower-order models. As a result, there are 5 separate probability regions corresponding to it – one in each back-off level (i.e. in models from 5-gram to 1-gram). These intervals will not be contiguous and requirement 2 will be violated.

4.2 Back-off tree

Figure 2 shows how to construct a *back-off tree* where each token is represented by a single leaf. Each leaf corresponds to a contiguous, non-zero interval. The set of all leaf intervals partitions $[0, 1)$. As a result, the tree can be interpreted as giving a p.m.f. over all tokens that satisfies every requirement from the previous section.

A different back-off tree is defined for each context w_{l-N+1}^{l-1}. The first level contains leaf nodes corresponding to tokens that belong to the set $\mathcal{W}_1 = \{w_l : f(w_{l-N+1}^{l}) \geq C\}$ and a single back-off parent node b_1. Each node is assigned a count $c(\cdot)$. For the leaves the count is simply $c(w_l) = f(w_{l-N+1}^{l})$. Calculating the back-off pseudo-count $c(b_1)$ will be addressed later. Nodes are arranged in the following way: leaves in order of increasing index followed by the back-off node. They partition $[0, 1)$ into corresponding intervals $v(\cdot)$ with size proportional to the node count.

Subsequent levels of depth $d \in \{2, \dots, N\}$ are constructed analogously. One difference is that the set of leaf nodes is $\mathcal{W}_d = \{w_l : f(w_{l-N+d}^{l}) \geq C \wedge f(w_{l-N+d-1}^{l}) < C\}$. Node counts are $c(w_l) = f(w_{l-N+d}^{l})$ similar to before. But instead of $[0, 1)$ the nodes partition $v(b_{d-1})$ – interval assigned to the back-off node in the previous level. In the last level there is also no back-off node.

Intuitively, level 1 tokens are the ones that give non-zero probability in an ML language model of order N. Level 2 tokens give non-zero probability in an ML model of order $N - 1$, but do not include tokens from level 1. Tokens in levels of depth $d \geq 3$ are chosen analogously and exclude tokens from *all* previous levels. By counts consistency, it suffices to check that $f(w_{l-N+(d-1)}^{l}) \geq C$ to exclude a token from level d. If it satisfies the inequality, it must have appeared in *some* previous level: $d - 1$ or earlier.

Sizes of intervals $v(w_l)$ are proportional to the ML estimate $\hat{P}(w_l|w_{l-N'+1}^{l-1})$ for highest $N' \leq N$ that gives a non-zero value. The constant of proportionally is different for each level of the back-off tree and depends on the back-off pseudo-counts $c(b_1), \dots, c(b_{N-1})$.

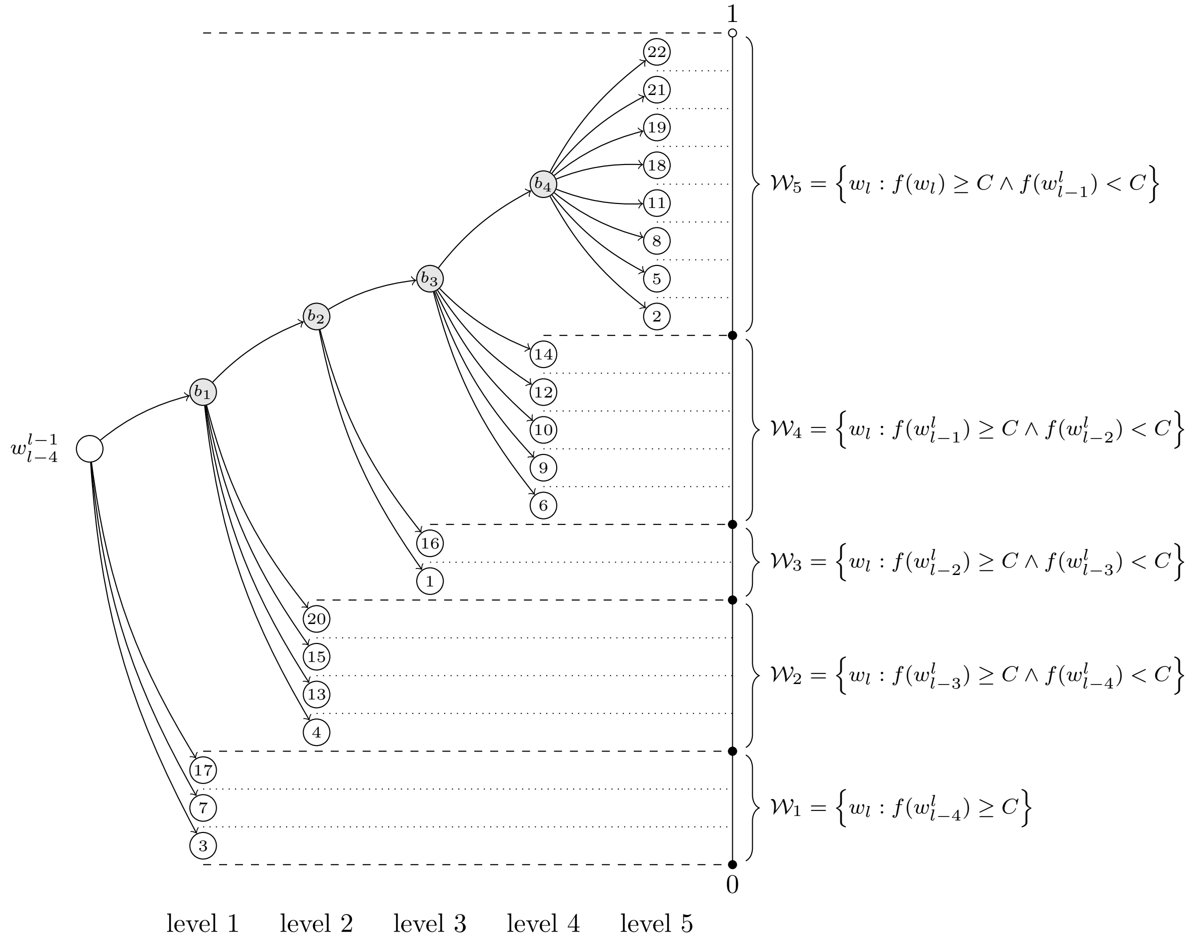
1
0
w_{l-4}^{l-1}
b_1
b_2
b_3
b_4
22
21
19
18
11
8
5
2
14
12
10
9
6
16
1
20
15
13
4
17
7
3
$\mathcal{W}_5 = \left\{w_l : f(w_l) \geq C \wedge f(w_{l-1}^l) < C\right\}$
$\mathcal{W}_4 = \left\{w_l : f(w_{l-1}^l) \geq C \wedge f(w_{l-2}^l) < C\right\}$
$\mathcal{W}_3 = \left\{w_l : f(w_{l-2}^l) \geq C \wedge f(w_{l-3}^l) < C\right\}$
$\mathcal{W}_2 = \left\{w_l : f(w_{l-3}^l) \geq C \wedge f(w_{l-4}^l) < C\right\}$
$\mathcal{W}_1 = \left\{w_l : f(w_{l-4}^l) \geq C\right\}$
level 1
level 2
level 3
level 4
level 5

Figure 2: Example back-off tree for context w_{l-4}^{l-1}, i.e. with $N = 5$. Leaf labels $w_l \in \{1, \dots, V\}$ correspond to token indices. In this example $V = 22$.

Note how in order to calculate the interval $v(w_l)$ of a token in level d of the tree it is only necessary to know the counts of tokens in levels $1, \dots, d$ and the back-off pseudo-counts $c(b_1), \dots, c(b_d)$.

4.3 $\beta{-}\gamma$ back-off pseudo-count estimation

Count of the back-off node $c(b_d)$ needs to reflect the probability that a not explicitly allowed token follows the context. Let us start with the first level of the tree. Context count, i.e. the number of times the context was observed in training data, is given by

$$m_1 = f(w_{l-N+1}^{l-1}). \tag{7}$$

By counts consistency, it will be larger than the total count of tokens in the first level of the tree. The leftover context count, i.e. $m_1 - \sum_{w_l \in \mathcal{W}_1} f(w_{l-N+1}^l)$, is then a good indication of how likely it is that a token $w_l \notin \mathcal{W}_1$ follows the context.

Keeping this in mind, I propose to use a back-off pseudo-count of

$$c(b_1) = \left\lceil \beta \left(m_1 - \sum_{w_l \in \mathcal{W}_1} f(w_{l-N+1}^{l}) \right) + \gamma \, m_1 \right\rceil, \tag{8}$$

where β controls what proportion of the *leftover context count* is used to account for the event of a back-off, and γ controls how much of the *total context count* is added extra to account for back-off as well. In the following, I will refer to the back-off pseudo-count expression I proposed in Eq. 8 as *β–γ estimation*.

To illustrate using the example from Section 4.1, there the total context count is $m_1 = 1146$ and the leftover context count is $1146 - 542 = 604$. Using parameter values $\beta = 0.5$ and $\gamma = 0.1$, the back-off pseudo-count is estimated to be $c(b_1) = \lceil 0.5 \times 604 + 0.1 \times 1146 \rceil = 417$.

Dealing with excluded tokens in a deeper level d of the back-off tree is similar. It suffices to remove excluded token counts from the total context count as follows

$$\overline{m}_d = f(w_{l-N+d}^{l-1}) - \sum_{w_l \in \mathcal{W}_1 \cup \dots \cup \mathcal{W}_{d-1}} f(w_{l-N+d}^{l}). \tag{9}$$

Then the calculation from Eq. 8 can be repeated in an analogous way

$$c(b_d) = \left\lceil \beta \left(\overline{m}_d - \sum_{w_l \in \mathcal{W}_d} f(w_{l-N+d}^{l}) \right) + \gamma \, \overline{m}_d \right\rceil. \tag{10}$$

The only special case to consider is when $m_1 = 0$ or $\overline{m}_d = 0$. This can simply occur if the context was not observed in training data frequently enough to be in the corpus, but also in more subtle situations.[2] Back-off pseudo-count would then be 0. However, the back-off node would be the only node in its level anyway. One way to deal with the problem elegantly is to give the back-off node an arbitrary non-zero pseudo-count, for example 1.

4.4 Offsetting counts

Sentences generated according to their ML probabilities are usually subjectively judged to be of low quality. The reason is that rarely seen N-grams have high variance in their counts. Since the predictive distribution of tokens in a sentence is heavy-tailed, these uncertain possibilities form the majority of the probability mass.

Introducing cutoff C is one way to deal with this problem. N-grams w_1^N with $f(w_1^N) < C$ are discarded and effectively given counts of zero. As a result, probabilities of uncommon tokens are calculated according to a lower-order, but more certain, model.

[2] If all tokens in a level are excluded *and* the context count $f(w_{l-N+d}^{l-1})$ is equal to the sum of excluded counts $\sum_{w_l \in \mathcal{W}_1 \cup \dots \cup \mathcal{W}_{d-1}} f(w_{l-N+d}^{l})$ then $\overline{m}_d$ will be 0. This happens if the context was observed in training data only followed by the excluded tokens.

An even bolder approach is to modify the counts. I propose a simple method where a constant offset F is removed from the count of every leaf in the back-off tree

$$\overline{c}(w_l) = c(w_l) - F. \tag{11}$$

The result is that uncommon N-grams become underrepresented and the sentences generated are skewed towards frequently observed N-grams, i.e. common phrases. Obviously, $F < C$ so that the counts are still positive.

Back-off pseudo-counts are modified in a different way. They are adjusted so that they roughly stay as the same proportion of a particular level of the tree

$$\overline{c}(b_d) = \left\lceil \frac{\sum_{w_l \in \mathcal{W}_d} \overline{c}(w_l)}{\sum_{w_l \in \mathcal{W}_d} c(w_l)} c(b_d) \right\rceil. \tag{12}$$

Offsetting counts does not have the same statistical motivation as the back-off scheme described in the previous section. It is simply an *ad-hoc* modification to yield subjectively better results. The effects of varying the F parameter are described in the results section.

5 The interval algorithm

The interval algorithm [15] provides a means of sampling from an arbitrary stochastic process (target) given an input sequence from a different arbitrary stochastic process (source). The algorithm is very similar to arithmetic coding – it also uses successive refinements of the $[0, 1)$ interval to map sequences.

The sequence mapping algorithm starts by mapping the empty sequence [ ] to the interval $[0, 1)$. For any sequence, its interval is partitioned into subintervals corresponding to the sequence extended by a single term. The size of each interval is chosen to be proportional to the probability of the last term occurring given the preceding sequence. The order of the subintervals does not matter, but needs to be consistent if reverse mapping from an interval to a sequence is to be performed. By the chain rule of probability, the size of the resulting interval is equal to the probability of observing its sequence. Consequently, only sequences of non-zero probability can be mapped. For a more detailed overview of arithmetic coding see [16, 17].

The interval algorithm works by continuously mapping the observed input sequence to successive intervals defined by the stochastic model of the source. In parallel, the intervals are reverse mapped to an output sequence according to the stochastic model of the target. Terms of the output sequence can be generated as long as its interval is a superinterval of the input sequence interval. See Figure 3 for an example of the procedure.

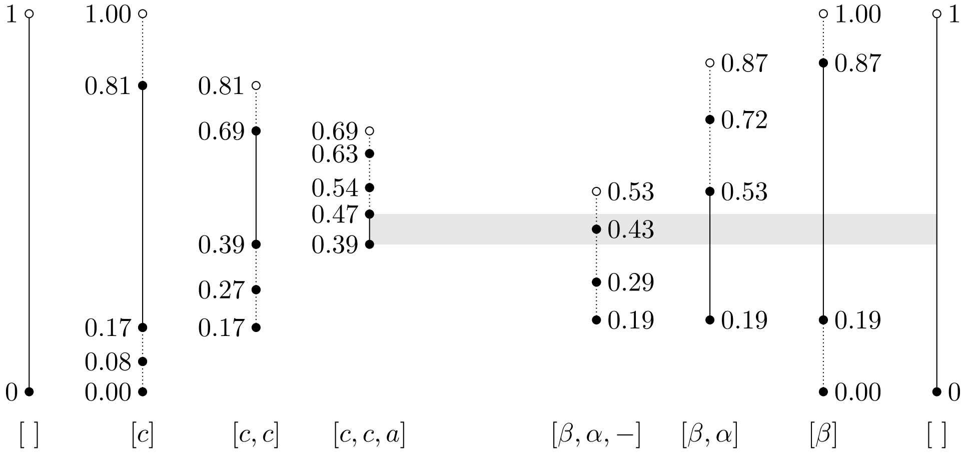

Figure 3: Example run of the interval algorithm. The source emits symbols over the alphabet $\{a, b, c, d\}$, the target – $\{\alpha, \beta, \gamma\}$. On the left, an input sequence $[c, c, a]$ from the source process gets mapped to $[0.39, 0.47)$. On the right, this interval allows to generate up to 2 terms of an output sequence from the target process, i.e. $[\beta, \alpha]$, which corresponds to $[0.19, 0.53)$. Note that it is not possible to generate more terms of the output sequence, as neither $[0.19, 0.29)$ nor $[0.29, 0.43)$ nor $[0.43, 0.53)$ is a superinterval of $[0.39, 0.47)$.

I will denote by $\langle \mathcal{I}/\mathcal{O} \rangle$ an instance of the interval algorithm configured for an input stochastic process $\mathcal{I}$ and an output stochastic process $\mathcal{O}$.

5.1 Extension to multiple input processes

It is possible to extend the interval algorithm to a scenario where the input sequence is a concatenation of sequences emitted by N stochastic processes $\mathcal{I}_1, \ldots, \mathcal{I}_N$. The input processes need to be defined over different alphabets $\mathfrak{I}_1, \ldots, \mathfrak{I}_N$. We can see that this extension does not pose any problems as long as the alphabets are different. When mapping input sequence to an interval, it is clear which process a given term was emitted by, thus the appropriate conditional probability subintervals can be found. The rest of the algorithm proceeds as before. I will denote by $\langle \mathcal{I}_1, \ldots, \mathcal{I}_N / \mathcal{O} \rangle$ an instance of the interval algorithm modified in such a way.

Note that if the processes are infinite, it is not possible to do the opposite, i.e. define an interval algorithm $\langle \mathcal{I} / \mathcal{O}_1, \ldots, \mathcal{O}_N \rangle$. It would end up being equivalent to simply $\langle \mathcal{I} / \mathcal{O}_1 \rangle$.

6 Stegosystem

In this section I will describe a simple stegosystem which allows to disguise an output sequence from one arbitrary stochastic process as a sequence which appears to have been generated by another arbitrary stochastic process. I will then describe how it is possible to construct a secure stegosystem with deniable encryption.

6.1 Simple stegosystem

The interval algorithm described in Section 5 can be used to construct a simple stegosystem that generates *stegotext* from *plaintext*[3]. Input plaintext sequence $\mathbf{P}$ on alphabet $\mathfrak{P}$ needs to have been generated according to a known stochastic process $\mathcal{P}$. An output stegotext sequence $\mathbf{S}$ on alphabet $\mathfrak{S}$ will be deterministically generated from $\mathbf{P}$ according to the statistics of an arbitrarily chosen stochastic process $\mathcal{S}$. Knowledge of $\mathbf{S}$ will allow to recover $\mathbf{P}'$ – a sequence on $\mathfrak{P}$ whose prefix is $\mathbf{P}$.

For example, if we want to transmit information that is optimally compressed to binary, $\mathcal{P}$ will be i.i.d. uniform. If we want the stegotext to be English text, we will use the English language model from Section 4 as $\mathcal{S}$.

Note that it is necessary to terminate $\mathbf{P}$ with an end-of-message symbol, or otherwise transmit its length, if we want to be able to uniquely recover $\mathbf{P}$ from $\mathbf{P}'$. This issue will be addressed later.

6.1.1 Detailed description of the system

The stegosystem requires a source of randomness $\mathcal{R}$ that outputs an infinite sequence $\mathbf{R}$ on alphabet $\mathfrak{R}$. $\mathcal{R}$ can be an arbitrary stochastic process with known statistics, for example a sequence of i.i.d. uniform bits.

The sequence $\mathbf{P}$ is concatenated with $\mathbf{R}$ to create an infinite sequence $\mathbf{I}$. With $\langle \mathcal{P}, \mathcal{R}/\mathcal{S} \rangle$ we can use $\mathbf{I}$ to deterministically generate an arbitrarily long output sequence $\mathbf{S}$. It remains to show that a sufficiently long $\mathbf{S}$ allows us to find a correct sequence $\mathbf{P}'$. Remember that $\mathbf{P}'$ is defined as a sequence on $\mathfrak{P}$ whose prefix is $\mathbf{P}$.

To recover $\mathbf{P}$ from $\mathbf{S}$, we can run the interval algorithm $\langle \mathcal{S}/\mathcal{P} \rangle$ with $\mathbf{S}$ as input. The only requirement to correctly decode all symbols of $\mathbf{P}$ is that $\mathbf{s} \subseteq \mathbf{p}$. This is the reason we used the process $\mathcal{R}$ during generation of $\mathbf{S}$ – we kept refining the input interval $\mathbf{i}$ until it got small enough that $\mathbf{s} \subseteq \mathbf{p}$. $\mathbf{s}$ was guaranteed to decrease every time a new symbol is appended to $\mathbf{S}$ since the conditional probabilities are always less than 1.[4] Figure 4 illustrates the procedure with a concrete example.

[3] Note that *plaintext* has a very precise meaning in this context – the unencrypted message. *Plain text* written separately simply refers to an unformatted sequence of characters.

[4] Each symbol has a non-zero probability and the alphabet size of obviously more than 1.

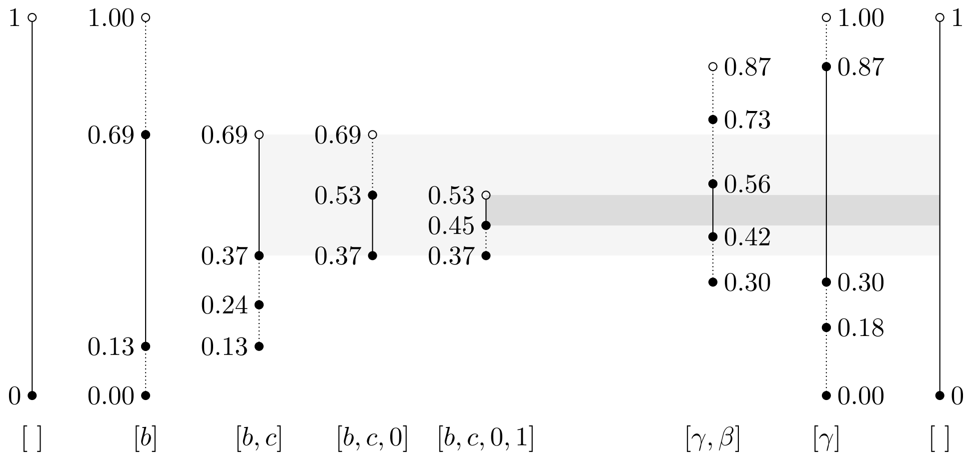

Figure 4: Generating **S** (right) from **P** (left). Both $\mathcal{P}$ and $\mathcal{S}$ are arbitrary stochastic processes defined over the alphabets $\mathfrak{P} = \{a, b, c\}$ and $\mathfrak{S} = \{\alpha, \beta, \gamma, \delta\}$. $\mathcal{R}$ emits i.i.d. uniform bits. Statistics of the corresponding processes are used to provide mappings between sequences and intervals. We are given $\mathbf{P} = [b, c]$ and we wish to generate **S** from it. **P** corresponds to the light grey region $\mathbf{p} = [0.37, 0.69)$, which only allows to generate **S** up to $[\gamma]$. Since $[0.30, 0.87) \not\subseteq [0.37, 0.69)$, $[\gamma]$ is not enough to identify a sequence with prefix **P**. However, if we use $\mathcal{R}$ to extend **P** to $\mathbf{I} = [b, c, 0, 1]$, we obtain the refined input interval $\mathbf{i} = [0.45, 0.53)$ shown in dark grey. **i** allows us to generate $\mathbf{S} = [\gamma, \beta]$ with the required property that $\mathbf{s} \subseteq \mathbf{p}$.

The problem is that we have no control over how small the interval **s** that satisfies $\mathbf{s} \subseteq \mathbf{p}$ gets. Since we do not know how long **P** is, we do not know when to stop generating symbols from $\mathcal{P}$. It may happen that even the largest of the intervals satisfying this condition will allow us to decode *more* than just **P**. This is why the sequence we decode is $\mathbf{P}'$, not **P**. See Figure 5 for an example of such a situation.

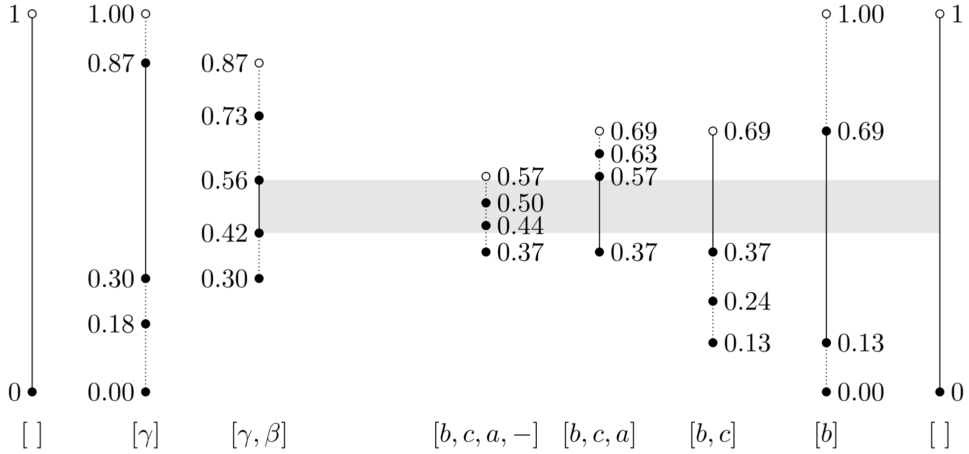

Figure 5: Recovering $\mathbf{P}'$ (right) from **S** (left) using the interval algorithm $\langle \mathcal{S}/\mathcal{P} \rangle$. Since we don't know how long the original sequence **P** was, we keep generating terms according to the statistics of $\mathcal{P}$ for as long as possible. In this case we decode $\mathbf{P}' = [b, c, a]$. $\mathbf{P}'$ has the correct prefix $\mathbf{P} = [b, c]$, but also contains an extra symbol a.

6.2 Secure stegosystem

The simple stegosystem from Section 6.1 can be used to construct a secure stegosystem, where a key sequence $\mathbf{K}$ is used to *securely* and *innocuously* transform a plaintext sequence $\mathbf{P}$ to a stegotext sequence $\mathbf{S}$. Only the knowledge of $\mathbf{K}$ allows to recover from $\mathbf{S}$ a sequence with prefix $\mathbf{P}$.

6.2.1 Generating secure stegotext

The plaintext and key sequences are first mapped to intervals $\mathbf{p}$ and $\mathbf{k}$ in the usual way using their stochastic process models $\mathcal{P}$ and $\mathcal{K}$. They are then assigned binary sequences $\mathbf{P_2}$ and $\mathbf{K_2}$:

Binary subinterval sequence $\mathbf{P_2}$ is the shortest sequence $\mathbf{P_2}$ such that $\mathbf{p_2} \subseteq \mathbf{p}$. Mapping between binary sequences and intervals is achieved in the usual way – by successive refinements of the $[0, 1)$ interval by its upper or lower half. Note that there may be more than one valid $\mathbf{P_2}$. See Figure 6.

Binary superinterval sequence $\mathbf{K_2}$ is the longest sequence $\mathbf{K_2}$ such that $\mathbf{k} \subseteq \mathbf{k_2}$. There is a single possible $\mathbf{K_2}$. Finding it is equivalent to generating $\mathbf{K_2}$ from $\mathbf{K}$ using the interval algorithm, so $\mathbf{K_2}$ is i.i.d. uniform by construction. See Figure 7.

The binary representation of plaintext $\mathbf{P_2}$ is then encrypted to binary *ciphertext* $\mathbf{C_2}$ using a stream cipher with key $\mathbf{K_2}$. If $\mathbf{K_2}$ is at least as long as $\mathbf{P_2}$, we can achieve perfect secrecy [18]. The simplest choice for a stream cipher is the *exclusive or* operation between $\mathbf{P_2}$ and infinitely repeated key $\mathbf{K_2^+}$, i.e. $\mathbf{C_2} = \mathbf{P_2} \oplus \mathbf{K_2^+}$.

$\mathbf{C_2}$ is used as an input to the simple stegosystem, producing pseudo-random English *stegotext* $\mathbf{S}$. Note that $\mathbf{C_2}$ is used as $\mathbf{P}$ in Section 6.1. Provided that the language model is correct, the stegotext is statistically indistinguishable from any random English text.

6.2.2 Recovering plaintext

We can use the simple stegosystem to recover a binary sequence $\mathbf{C_2'}$ from $\mathbf{S}$. $\mathbf{C_2'}$ is a sequence whose prefix is $\mathbf{C_2}$, i.e. the original input to the simple stegosystem. Knowing the key $\mathbf{K}$ and its binary representation $\mathbf{K_2}$, we can recover the original binary plaintext sequence potentially followed by more bits, i.e. $\mathbf{P_2'} = \mathbf{C_2'} \oplus \mathbf{K_2^+}$.

$\mathbf{p_2'} \subseteq \mathbf{p_2} \subseteq \mathbf{p}$. The first relationship comes from the fact that $\mathbf{P_2}$ is a prefix of $\mathbf{P_2'}$ and the second holds because $\mathbf{P_2}$ is a binary subinterval sequence of $\mathbf{p}$. Using the language model we can now recover $\mathbf{P'}$, i.e. the longest sequence such that $\mathbf{p_2'} \subseteq \mathbf{p'}$. $\mathbf{P'}$ will be the original plaintext $\mathbf{P}$ potentially followed by extra tokens.

6.2.3 Deniable encryption

An important feature of a stegosystem constructed in this way is *deniable encryption.* Assume that an enemy with full knowledge of the system is told that a particular fragment of text $\mathbf{S}$ is indeed stegotext. Using the knowledge of the system (i.e. the language model), they can reconstruct $\mathbf{C'_2}$ – the binary ciphertext sequence with trailing bits.

However, the recovery of plaintext requires knowledge of the key. If a *false key* $\overline{\mathbf{K}}$ is supplied in text form, it will still be correctly converted to its binary superinterval sequence $\overline{\mathbf{K}}_\mathbf{2}$. It will then allow to decipher without failure ciphertext $\mathbf{C'_2}$ into an *incorrect* binary sequence $\overline{\mathbf{P'}}_\mathbf{2} = \mathbf{C'_2} \oplus \overline{\mathbf{K}}^{+}_\mathbf{2}$. $\overline{\mathbf{P'}}_\mathbf{2}$ will then be mapped to some incorrect plaintext $\overline{\mathbf{P'}}$, generated according to the stochastic process $\mathcal{P}$. So it will be a fragment of text randomly drawn from all English sentences.

A user asked to supply the key may give $\overline{\mathbf{K}}$ instead and claim that $\overline{\mathbf{P'}}$ contains the plaintext message as a prefix. If the enemy does not know $\mathbf{P}$ but expects it to be just a random sentence, i.e. one generated according to the same statistics as $\mathcal{P}$, $\overline{\mathbf{P'}}$ is just as likely to contain $\mathbf{P}$ as a prefix as is the real $\mathbf{P'}$.

6.2.4 Example

Both plaintext and stegotext are generated according to the same stochastic process $\mathcal{E}$ over the alphabet of three tokens $\{\alpha, \beta, \gamma\}$. The plaintext we wish to securely and innocuously transmit is $\mathbf{P} = [\beta, \gamma]$. For this purpose we use the key $\mathbf{K} = [\beta, \alpha]$.

Figures 6 to 8 demonstrate generating the stegotext $\mathbf{S} = [\gamma, \alpha, \beta]$ from $\mathbf{P}$ and $\mathbf{K}$. Figures 9 and 10 show recovering from the stegotext $\mathbf{S}$ a sequence $\mathbf{P'} = [\beta, \gamma, \beta]$ with prefix $\mathbf{P}$, given the knowledge of the key $\mathbf{K}$.

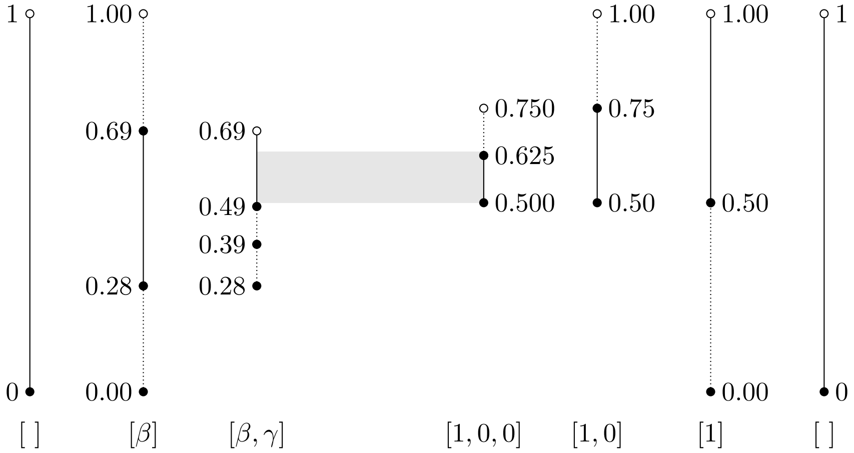

Figure 6: Converting the plaintext $\mathbf{P} = [\beta, \gamma]$ to a binary sequence $\mathbf{P_2} = [1, 0, 0]$. $\mathbf{P_2}$ is a binary subinterval sequence of the plaintext interval $\mathbf{p} = [0.49, 0.69)$.

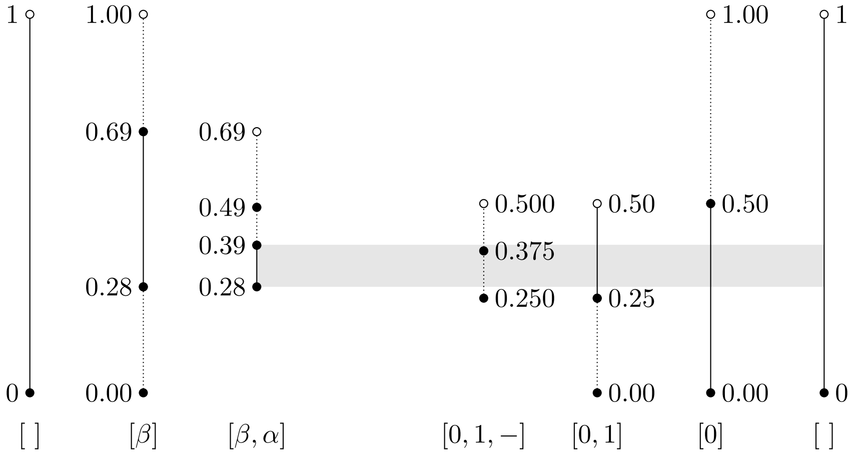

Figure 7: Converting the key $\mathbf{K} = [\beta, \alpha]$ to a binary sequence $\mathbf{K_2} = [0, 1]$. $\mathbf{K_2}$ is the binary superinterval sequence of the key interval $\mathbf{k} = [0.28, 0.39)$. We are effectively using the interval algorithm $\langle \mathcal{E}/\mathcal{B} \rangle$, where $\mathcal{B}$ denotes a source of i.i.d. uniform bits.

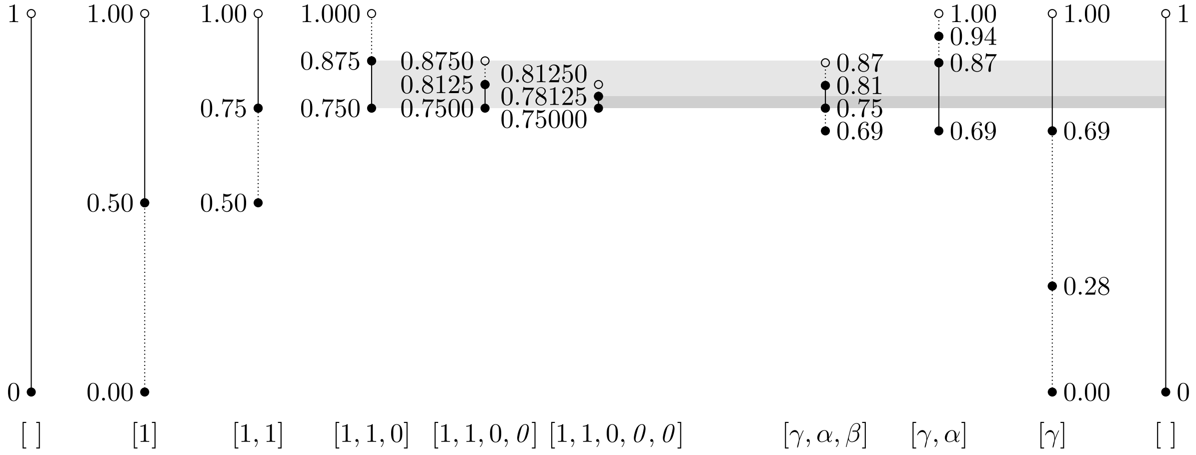

Figure 8: Generating stegotext $\mathbf{S} = [\gamma, \alpha, \beta]$ from ciphertext $\mathbf{C_2} = [1, 1, 0]$. Ciphertext is constructed by the *exclusive or* operation on streams of $\mathbf{P_2}$ and the cycled key $\mathbf{K_2^+} = [0, 1, 0, 1, 0, 1, \ldots]$, i.e. $\mathbf{C_2} = \mathbf{P_2} \oplus \mathbf{K_2^+} = [1, 1, 0]$. Ciphertext needs to be extended to $\mathbf{I} = [1, 1, 0, \mathit{0}, \mathit{0}]$ so that $\mathbf{s} \subseteq \mathbf{c_2}$, i.e. $[0.75, 0.81) \subseteq [0.750, 0.875)$. Note that the stegotext is not unique – if we extended the ciphertext to $\mathbf{I} = [1, 1, 0, \mathit{1}, \mathit{0}]$, the stegotext would be $\mathbf{S} = [\gamma, \alpha, \gamma]$.

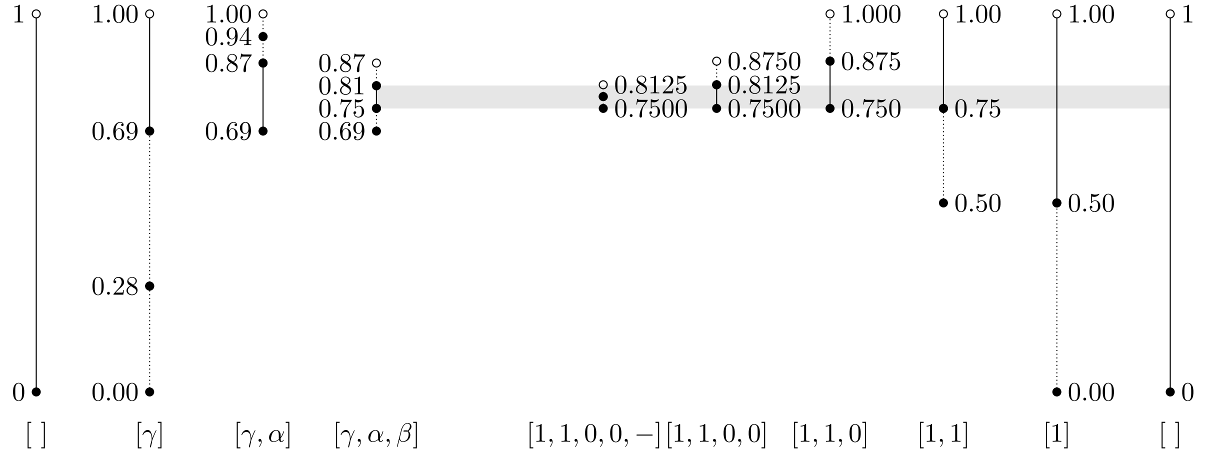

Figure 9: Recovering the ciphertext with trailing bits $\mathbf{C_2'} = [1, 1, 0, 0]$ from the stegotext $\mathbf{S} = [\gamma, \alpha, \beta]$. We are effectively using the interval algorithm $\langle \mathcal{E}/\mathcal{B} \rangle$. Since the recipient of the message knows $\mathbf{K}$, $\mathbf{K_2} = [0, 1]$ can be obtained exactly as in Figure 7. Deciphered binary plaintext with trailing bits is then $\mathbf{P_2'} = \mathbf{C_2'} \oplus \mathbf{K_2^+} = [1, 0, 0, 1]$. We can verify that $\mathbf{P_2'}$ has the prefix $\mathbf{P_2} = [1, 0, 0]$.

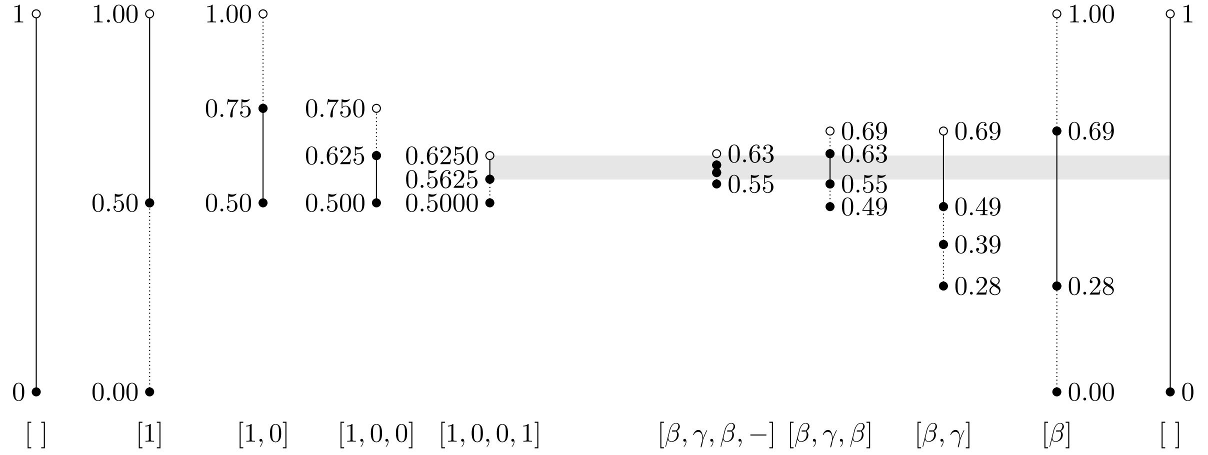

Figure 10: Recovering the plaintext with trailing symbols $\mathbf{P'} = [\beta, \gamma, \beta]$ from the binary plaintext with trailing bits $\mathbf{P_2'} = [1, 0, 0, 1]$. $\mathbf{P'}$ has indeed the prefix $\mathbf{P} = [\beta, \gamma]$, so the stegosystem works correctly.

7 Results

In this section I will present the results of my project. I have implemented the full language model from Section 4 along with the secure stegosystem from Section 6.2.

7.1 Generating random English sentences

The language model from Section 4 and the interval algorithm from Section 5 can be used to generate random English sentences. A sequence emitted by an arbitrary stochastic process $\mathcal{I}$ needs to be input to an interval algorithm $\langle \mathcal{I}/\mathcal{E} \rangle$, where $\mathcal{E}$ denotes the stochastic process of English text. The output will be randomly generated English sentences.

I used i.i.d. uniform bits generated by Python's `random.SystemRandom` class as $\mathcal{I}$. See Appendix A for examples of sentences generated using a 5-gram model with different combinations of parameters β, γ and F.

Table 7 shows statistics of sample sentences generated using 2048 random bits. In statistical natural language processing, *perplexity per word* of a fragment of text w_1^L is defined in bits as $-\frac{1}{L}\log_2 P(w_1^L)$. It can be interpreted as the expected negative log probability of a token. Results of the simulations are presented only for illustrative purposes. The analysis can be taken further by sweeping the parameter values more thoroughly and using larger sample size, i.e. generating sentences based on more than 2048 bits of randomness. See Section 8.1 for a discussion of the importance of these statistics and their relationship to the performance of the system.

β	γ	F	Perplexity per word	Av. sentence length
1	0.01	39	5.5 bits	20
0.5	0.05	39	5.3 bits	21
0.1	0.01	39	4.1 bits	20
1	0.01	35	6.1 bits	24
0.5	0.05	35	5.6 bits	20
0.1	0.01	35	4.3 bits	18
1	0.01	25	5.5 bits	27
0.5	0.05	25	5.4 bits	17
0.1	0.01	25	3.8 bits	18
1	0.01	10	6.0 bits	21
0.5	0.05	10	5.3 bits	20
0.1	0.01	10	3.9 bits	20
1	0.01	0	6.1 bits	26
0.5	0.05	0	5.8 bits	20
0.1	0.01	0	4.3 bits	22

Table 7: Statistics of random English sentences generated using the interval algorithm and the English language model with varying parameter values.

7.2 Secure stegosystem using the English language model

I used the English language model for $N = 5$ and with parameters $\beta = 0.1$, $\gamma = 0.01$ and $F = 35$ to define a stochastic process $\mathcal{E}$. I then assumed that the plaintext, the key and the stegotext in the secure stegosystem from Section 6.2 all have the statistics of $\mathcal{E}$.

Tables 8 to 11 demonstrate generating example stegotext and Tables 12 and 13 show recovering the plaintext using the correct and an incorrect key. The procedure and notation are exactly the same as in the example from Section 6.2.4, so it can be referred to for a more detailed description of the method.

plaintext	Come in the morning with the documents.
$\mathbf{P}$	_START_ come in the morning with the documents . _END_
$-\log_2 P(\mathbf{P})$	53.9 bits
$\mathbf{P_2}$	0 1 0 0 0 0 1 0 1 1 0 1 1 0 1 0 0 0 0 1 1 1 0 1 0 0 0 1 1 1 0 0 1 0 0 1 1 1 1 0 0 0 0 0 0 1 0 0 1 1 1 0 1 0 0
$\lvert\mathbf{P_2}\rvert$	55 bits

Table 8: Parsing plaintext **P** into a sequence of tokens and converting it to binary. Note that the sequence $\mathbf{P_2}$ contains $55 - 53.9 = 1.1$ bits of redundant information content w.r.t. **P**. This is because $\mathbf{P_2}$ needs to be a binary *subinterval* sequence of **P**. The difference is however never larger than 2 bits – it is a known result for arithmetic coding.

real key	Light rain on Monday morning.
$\mathbf{K}$	_START_ light rain on monday morning . _END_
$-\log_2 P(\mathbf{K})$	47.0 bits
$\mathbf{K_2}$	1 0 0 0 1 0 1 1 0 0 1 1 1 0 0 1 0 1 1 0 1 0 1 0 0 0 0 1 0 0 0 0 0 0 0 1 1 1 1 1 1 0 1 1 0 0
$\lvert\mathbf{K_2}\rvert$	46 bits

Table 9: Parsing the real key into a sequence of tokens and then converting it to binary. This time, $\mathbf{K_2}$ has fewer bits than **K** because it needs to be a binary *superinterval* sequence. In other words, we cannot use all the information from **K** to generate $\mathbf{K_2}$.

false key	Early light rain or drizzle will soon die out.
$\overline{\mathbf{K}}$	_START_ early light rain or drizzle will soon die out . _END_
$-\log_2 P(\overline{\mathbf{K}})$	84.1 bits

$\overline{\mathbf{K}}_\mathbf{2}$	0 1 0 0 1 1 0 0 0 0 0 1 1 1 0 1 0 0 1 0 0 1 0 0 1 0 0 1 1 1 0 1 0
	0 1 1 0 1 0 0 0 1 0 1 1 0 1 0 0 0 1 1 1 0 0 0 1 0 0 1 0 1 0 0 0 1
	1 1 0 1 0 1 0 1 0 1 0 1 1 0 1 0 1

$\lvert\overline{\mathbf{K}}_\mathbf{2}\rvert$	83 bits

Table 10: Processing the false key in the same way as in Table 9.

$\mathbf{C_2} = \mathbf{P_2} \oplus \mathbf{K_2^+}$	1 1 0 0 1 0 0 1 1 1 1 0 0 0 1 1 0 1 1 1 0 1 1 1 0 0 0 0 1 1 0 0 1 0 0 0 0 0 0 1 1 0 1 1 0 1 1 0 1 1 0 0 0 1 0
$\|\mathbf{C_2}\|$	55 bits
$\mathbf{S}$	_START_ the hurrian pantheon of the gods , o king , who is in turn responsible for the conditions at the surface of the whole earth , we could not talk to her about what she had done . _END_
$-\log_2 P(\mathbf{S})$	152.7 bits
stegotext	The Hurrian pantheon of the gods, o king, who is in turn responsible for the conditions at the surface of the whole Earth, we could not talk to her about what she had done.

$\mathbf{C_2'}$	1 1 0 0 1 0 0 1 1 1 1 0 0 0 1 1 0 1 1 1 0 1 1 1 0 0 0 0 1 1 0 0 1
	0 0 0 0 0 0 1 1 0 1 1 0 1 1 0 1 1 0 0 0 1 0 1 0 0 0 1 1 1 0 1 0 1
	1 0 1 0 1 0 0 1 0 0 1 1 0 1 1 1 0 0 0 1 1 0 1 1 1 0 0 0 0 1 1 0 1
	0 0 1 0 0 0 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 0 0 0 0 1 0 0 1 1 1
	1 0 0 1 1 0 1 1 0 1 0 0 1 1 0 1 0

$\|\mathbf{C_2'}\|$	149 bits

Table 11: Generating stegotext using the real key. Note that the final stegotext has been formatted *by hand* to appear more natural. Any modifications are allowed as long as the formatted text parses to the same sequence of tokens $\mathbf{S}$. The stegotext is very long – its information content is almost 3 times that of the ciphertext $\mathbf{C_2}$, which it needs to identify. In total it contains $152.7 - 55 = 97.7$ bits of redundant information. The large difference results from the fact that, in order to appear natural, the stegotext sequence needs to terminate at an *[_END_]* token. Since beyond the input sequence $\mathbf{C_2}$ the stegotext is generated randomly, in a different realisation the stegotext $\mathbf{S}$ can be shorter or longer.

$\mathbf{P_2'} = \mathbf{C_2'} \oplus \mathbf{K_2^+}$	0 1 0 0 0 0 1 0 1 1 0 1 1 0 1 0 0 0 0 1 1 1 0 1 0 0 0 1 1 1 0 0 1
	0 0 1 1 1 1 0 0 0 0 0 0 1 0 0 1 1 1 0 1 0 0 1 1 1 1 1 1 0 0 0 1 1
	0 0 0 0 1 0 0 0 0 0 1 1 0 1 1 0 1 1 1 0 0 0 0 0 1 0 1 0 0 1 0 0 0
	1 0 1 1 1 1 0 0 0 1 0 0 0 1 1 0 1 1 0 0 0 1 1 0 0 0 0 1 1 1 0 0 0
	0 0 1 0 1 0 0 1 0 1 1 0 0 0 0 1 1

$\|\mathbf{P_2'}\|$	149 bits
$\mathbf{P'}$	_START_ come in the morning with the documents . _END_ _START_ the grocer takes the form of a human , or a full - time basis . _END_ _START_ with this
$-\log_2 P(\mathbf{P'})$	139.0 bits
recovered plaintext	Come in the morning with the documents. The grocer takes the form of a human, or a full-time basis. With this

Table 12: Recovering plaintext using the correct key. Note how the recovered sequence does *not* have to terminate with an *[_END_]* token. Recovered plaintext has been again formatted by hand to a fragment of text equivalent to the sequence of tokens.

$\overline{\mathbf{P'}}_\mathbf{2} = \mathbf{C'_2} \oplus \overline{\mathbf{K}}_\mathbf{2}^+$	1 0 0 0 0 1 0 1 1 1 1 1 1 1 1 0 0 1 0 1 0 0 1 1 1 0 0 1 0 0 0 1 1
	0 1 1 0 1 0 1 1 1 1 0 1 1 0 0 1 1 1 1 1 1 0 1 1 0 0 0 1 0 0 1 0 0
	0 1 1 1 1 1 0 0 0 1 1 0 1 1 0 1 1 0 1 1 1 1 0 1 1 0 0 0 1 0 0 0 0
	0 0 0 0 0 1 0 0 0 1 1 0 1 1 1 0 1 1 1 1 1 0 1 0 0 1 0 0 1 0 0 1 1
	1 1 1 0 1 0 1 0 0 1 1 0 0 1 0 1 1

$\lvert\overline{\mathbf{P'}}_\mathbf{2}\rvert$	149 bits
$\overline{\mathbf{P'}}$	_START_ jennings , h . s . (editor) . _END_ _START_ an abstruse and learned specialist who finds that he has been to the west of the entrance to the vagina . _END_ _START_ there is a suggestion that the earth might be blessed . _END_ _START_
$-\log_2 P(\overline{\mathbf{P'}})$	145.8 bits
recovered false plaintext	Jennings, H. S. (editor). An abstruse and learned specialist who finds that he has been to the west of the entrance to the vagina. There is a suggestion that the Earth might be blessed.

Table 13: Recovering plaintext using an incorrect key.

8 Discussion

8.1 Language model statistics and the rate of the stegosystem

8.1.1 Perplexity of a language model

Perplexity per token in bits is the expected negative log probability of a token given the preceding ones. In the interval algorithm, it directly corresponds to the expected size of successive subintervals. It can be also interpreted as the number of bits of information needed on average to encode a single token. See [19] for a broader treatment of perplexity in computational linguistics.

We can talk about perplexity of a stochastic process itself (first sense) and the perplexity of a stochastic process that tries to model some test data (second sense). The former is equivalent to its entropy rate. It applies to the generation of random sentences and describes the variety of text that can be produced by the model. For example, for a model with perplexity of 5.4 bits, generating the next token can be thought of as equivalent to choosing 1 out of $2^{5.4} \approx 42$ equally likely choices. Table 7 shows how reducing the amount of probability mass assigned to back-off reduces the perplexity of a model.

It can be trivially shown that an i.i.d. uniform model has the highest possible perplexity. However, it is obviously not a good model of English. This is why language models are judged by the former criterion – perplexity when modelling test data. It is directly related to the cross-entropy between the probability distribution of the model and the true probability distribution of test text. The lower the perplexity, the better the model predicts the next token.

8.1.2 Plaintext model perplexity

When encoding the plaintext we should *minimise* the perplexity in the second sense. We want the information to be optimally compressed. This can be achieved by learning and matching the probability distribution of plaintext. Indeed, this is how the interval algorithm works – by mapping a sequence using its *true* distribution.

8.1.3 Stegotext model perplexity

To maximise the rate of the system, the stegotext should be generated using a source with *maximum* perplexity in the first sense. We want each output token to contain as much information as possible – this is how we ensure that the information from plaintext is transmitted using as few stegotext tokens as possible. For example, if we only used the works of Shakespeare to model stegotext, there would less variety in the output sentences than if we used a corpus of all books in print. As a result, we would need to output relatively long paragraphs of text. Conversely, if the sentences came from general English, each one would be relatively unlikely, so would contain a lot of information.

However, to ensure innocuousness, stegotext needs to be generated according to the same process as the text it aims to mimic. So we are again forced to match the stochastic process to fit particular training data, i.e. innocuous English text.

8.1.4 Rate of the stegosystem

Knowing the perplexities of the plaintext and stegotext models, we can talk about the expected rate of the system, i.e. the number of plaintext tokens that are on average transmitted using stegotext tokens. [15] gives lower and upper bounds for the expected length of the input sequence when generating a *single* term from a discrete output probability distribution using the interval algorithm. The only property of the discrete output distribution it depends on is the entropy.

In principle, it should be possible to adapt this bound to the case of the stegosystem. Input and output should be simply reversed – in the stegosystem it is the output that has to uniquely identify the input. We should also interpret plaintext as a discrete probability distribution over the set of *all* possible plaintext sequences of length L. In this way, we will be able to talk about the entropy of plaintext as a function of L, the number of tokens we want to transmit.

There are two problems. Firstly, because of the prohibitive size of the dynamically generated conditional probability trees, none of the statistics can be calculated exactly – we need to estimate them and motivate why the estimations are justified.

Secondly, in a practical stegosystem it is advantageous to continue generating stegotext randomly until the end of a sentence is reached. As a result, the stegotext does not awkwardly terminate mid-sentence. This lowers the rate, as some number of trailing tokens needs to be appended. It is not trivial to introduce this constraint when calculating bounds on the expected number of tokens of stegotext.

However, there is a number of *common sense* relationships between model statistics and the rate, which I will state without proof:

1. The rate falls with plaintext perplexity.

2. The rate increases with stegotext perplexity.

3. The rate falls with average stegotext sentence length.

8.2 Unique recovery of plaintext

8.2.1 Communicating the length of plaintext

The stegosystem as described in Section 6.1 does not allow to uniquely identify plaintext from stegotext. We can only recover a sequence of tokens whose *prefix* is the plaintext. If we want to know plaintext exactly, its length needs to be somehow communicated:

EOM token We can add an *end of message* token to our alphabet. The problem is learning its conditional probabilities. We could constraint *[_EOM_]* to only follow an *[_END_]* token and then create a probability distribution over the number of sentences in the plaintext. We can choose for example a p.m.f. from the exponential family. The subinterval size corresponding to *[_EOM_]* would be then equal to the probability that the current sentence is the last one.

Communicating L at the beginning of plaintext We can transmit the length L of plaintext as the first symbol. $[0, 1)$ can be partitioned into an infinite number of arbitrarily sized intervals, each corresponding to a particular L. Thus, if we can measure the p.m.f. of L, we can exactly match it. If in practice we do not care about being optimal, we can use a binary integer prefix code to partition $[0, 1)$. The requirement is that every possible sequence of bits either is a codeword, is a prefix of a codeword, or has a codeword as a prefix – then $[0, 1)$ will be partitioned among all integers. Elias coding [20] is one possible choice of a code.

8.2.2 Compromising innocuousness

In the case of a simple stegosystem from Section 6.1, communicating the length of plaintext may compromise innocuousness. Real stegotext is guaranteed to have *consistent plaintext length* – we should either eventually decode an *[_EOM_]* token, or we should not decode more tokens than the number given by the length declaration.

This is not the case with truly random sentences interpreted as stegotext. If our stegosystem uses the *[_EOM_]* token, it is not guaranteed to appear within the recovered plaintext sequence. Similarly, if the plaintext length declaration gives L, we may in fact recover a plaintext sequence shorter than L.

As a result, plaintext length consistency increases the probability that observed fragment of text is stegotext. It is possible that length consistency is an *almost sure* indication of using the stegosystem – simulations or theoretical analysis should answer this question.

8.2.3 Compromising security

If we use the secure stegosystem from Section 6.2, innocuousness of transmission is not compromised in an obvious way. We cannot attempt to decode plaintext without knowing the key, so we do not know if its length will be consistent or not.

However, an enemy with partial knowledge can infer more information. For example, if the enemy knows that a fragment of text is stegotext, they also know that a key that gives inconsistent plaintext length is not correct – deniable encryption no longer applies. If they know the key, the situation is completely analogous to the one in the previous section – they have an above chance probability of guessing whether a given fragment of text is stegotext or not.

8.3 Sentence delimiters

Google Books N-grams contain *[_START_]* and *[_END_]* tokens to identify beginning and end of a sentence. If stegotext is *generated* using the language model, it will inevitably contain these tokens. For example any sequence of full sentences will start with *[_START_]* and finish with *[_END_]*.

Some kind of convention needs to be adopted to uniquely identify sentence delimiters in *formatted* output text. I decided to denote them by multiple whitespace characters, for example a double space. To avoid ambiguity, the empty sentence *[_START_ _END_]* is not allowed in the model.

This has non-trivial consequences for practical application of the system. Since real text rarely has such specific and explicit sentence boundaries, their presence is suspicious. In hindsight, choosing a corpus without explicit sentence start and end tokens would simplify matters. In addition, it would introduce desirable continuity between sentences.

Examples of corpora meeting this condition are the 2009 Google Books N-grams Corpus and the COCA N-grams discussed below.

8.4 Using COCA instead of Google Books

Sentences shown in Appendix A do not always look natural. They still contain some *noise* (i.e. rather meaningless punctuation characters and single letters placed next to each other), or even foreign words.

The first problem is partly amplified by normalising and exploding tokens, as described in Sections 3.2.2 and 3.2.3. Table 2 in the first section shows that special characters are transliterated into basic punctuation, which is then exploded into multiple tokens. This is necessary to unambiguously interpret formatted text, but inflates the punctuation counts.

The aim of the Google Books project is to digitise *all* books in the world. The corresponding N-gram corpus is only a side project, so source texts are not curated in any major way. On the other hand, the Corpus of Contemporary American English has been constructed specifically to be a large and balanced corpus suitable for linguistic analysis [11].

I expect the COCA N-grams corpus to contain fewer special special characters or foreign words. Additionally, it does not have a cutoff on counts, which gives more flexibility in designing the language model. Its drawbacks compared to the Google Books N-grams are: price and counts given only up to $N = 4$. It is not possible to be sure without implementing the COCA corpus, but I now expect it to be give better results in a practical system.

9 Conclusions

I have constructed a stegosystem capable of transforming a secret message into a fragment of text that appears to be randomly drawn from the set of all English sentences. The security of the message is ensured by a symmetric key, which is another fragment of text.

9.1 Contributions

In support of the stegosystem, I have done extensive work in the area of statistical natural language processing. My contributions include methods for processing N-gram counts corpora and a large-scale language model meeting particular requirements. I introduced a procedure for breaking complex, potentially overlapping N-grams into simpler ones and an algorithm for inducing the counts of simple N-grams from the counts of the complex ones. I proposed a language model that assigns a contiguous, non-zero conditional probability region to every possible token extending a sequence. I also created my own method for weighting estimates coming from conditional probability models with different history sizes, i.e. $\beta{-}\gamma$ back-off. I processed the Google Books N-grams Corpus to create a consistent and succinct N-gram counts database and fully implemented the language model based on this data.

I also designed a simple stegosystem that can convert any plaintext sequence into a stegotext sequence that appears to have been generated by an arbitrary stochastic process. It is possible to recover the plaintext from the stegotext, up to the limitation that the original sequence can be followed by random symbols. I based the system on the interval algorithm for random generation, which in turn is closely related to the arithmetic coding compression method. My contribution was adapting these methods to enable the recovery of the input from the output.

Finally, I introduced secrecy to the simple stegosystem. A symmetric key is used to encipher plaintext before transforming it to stegotext. Only the knowledge of they key allows correct recovery of the plaintext. Moreover, I designed the system in such a way that it is not possible to verify the correctness of a supplied key – any key will allow to recover plausible plaintext.

9.2 Future work

Different corpora can be used in training the language model, e.g. the stegotext may be chosen to resemble weather forecast. An end-of-message symbol or plaintext length declaration can be introduced to the system, carefully examining the security risks associated with such an extension. The transmission rate of the system can be examined more thoroughly, either theoretically or by use of simulations. Finally, a framework for assessing the degree of innocuousness of stegotext based on some statistical tests may be introduced.

References

[1] Wikipedia. Steganography — Wikipedia, the free encyclopedia. `http://en.wikipedia.org/w/index.php?title=Steganography&oldid=590219361`, 2014. Online; accessed January 13, 2014.

[2] Philip C. Woodland. *4F11 Speech and Language Processing, Lecture 7: Statistical Language Models.* Cambridge University Engineering Department, Lent 2014 edition, February 2014. Online; accessed May 2, 2014.

[3] William J. Byrne. *4F11 Speech and Language Processing, Lecture 13: Statistical Machine Translation Systems.* Cambridge University Engineering Department, Lent 2014 edition, March 2014. Online; accessed May 2, 2014.

[4] Daniel Jurafsky and Christopher Manning. *Natural Language Processing, Week 2: Language Modeling.* Coursera. Online; accessed May 2, 2014.

[5] Jean-Baptiste Michel, Yuan Kui Shen, Aviva Presser Aiden, Adrian Veres, Matthew K. Gray, William Brockman, The Google Books Team, Joseph P. Pickett, Dale Hoiberg, Dan Clancy, Peter Norvig, Jon Orwant, Steven Pinker, Martin A. Nowak, and Erez Lieberman Aiden. Quantitative Analysis of Culture Using Millions of Digitized Books. *Science*, 331(6014):176–182, January 2011.

[6] Irving J Good. The population frequencies of species and the estimation of population parameters. *Biometrika*, 40(3-4):237–264, 1953.

[7] Reinhard Kneser and Hermann Ney. Improved backing-off for M-gram language modeling. In *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, volume 1, pages 181–184. IEEE, 1995.

[8] Thorsten Brants, Ashok C Popat, Peng Xu, Franz J Och, and Jeffrey Dean. Large language models in machine translation. In *Proceedings of the Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning.* Association for Computational Linguistics, 2007.

[9] Yuri Lin, Jean-Baptiste Michel, Erez Lieberman Aiden, Jon Orwant, Will Brockman, and Slav Petrov. Syntactic annotations for the Google Books Ngram Corpus. In *Proceedings of the ACL 2012 System Demonstrations*, pages 169–174. Association for Computational Linguistics, 2012.

[10] Mark Davies. N-grams: based on 450 million word COCA corpus. `http://www.ngrams.info/`, 2013. Online; accessed October 18, 2013.

[11] Mark Davies. The Corpus of Contemporary American English as the first reliable monitor corpus of English. *Literary and Linguistic Computing*, 25(4):447–464, October 2010.

[12] Tomaz Solc and Sean M. Burke. Unidecode 0.04.14 – ASCII transliterations of Unicode text. `https://pypi.python.org/pypi/Unidecode`, 2013. Online; released September 20, 2013.

[13] NA 043-03-01 AA committee. Rules for alphabetical ordering – Part 2: Presentation of names. DIN 5007-2, Deutsches Institut für Normung e. V., May 1996.

[14] Yuri Lin, Dorothy Curtis, and Slav Petrov. Syntactically annotated Ngrams for Google Books. Master's thesis, Massachusetts Institute of Technology, 2012.

[15] Te Sun Han and Mamoru Hoshi. Interval Algorithm for Random Number Generation. *IEEE Transactions on Information Theory*, 43(2):599–611, March 1997.

[16] Thomas M. Cover and Joy A. Thomas. Shannon-Fano-Elias coding. In *Elements of Information Theory*, chapter 5.7, pages 127–130. John Wiley & Sons, Inc., 2nd edition, 2006.

[17] Amir Said. Introduction to arithmetic coding – theory and practice. Hewlett Packard Laboratories Report, Hewlett Packard Laboratories Palo Alto, April 2004.

[18] Claude E Shannon. Communication theory of secrecy systems*. *Bell System Technical Journal*, 28(4):656–715, 1949.

[19] Peter F Brown, Vincent J Della Pietra, Robert L Mercer, Stephen A Della Pietra, and Jennifer C Lai. An estimate of an upper bound for the entropy of English. *Computational Linguistics*, 18(1):31–40, 1992.

[20] Peter Elias. Universal codeword sets and representations of the integers. *Information Theory, IEEE Transactions on*, 21(2):194–203, Mar 1975.

A Random English sentences

A.1 $N = 5,\ \beta = 1,\ \gamma = 0.01,\ F = 39$

START managed code ss payne ” but we have still to understand , the principal one will often throw . _END_ _START_ the answer is clearly yes . _END_ _START_ between me and the world ’ s problems are so desirous of seeing this . _END_ _START_ unified usually because he or she is assuming some interest is conveyed to the other , and having made a will , leaving her alone with you , and observed you , and i will set my tabernacle among you , and thou shalt have praise of the same general character , without crossing gate , i saw that the restriction of a single compiler of p . birthing : k . j . danna , ” ” la liga narodowa _END_ _START_ there was more coal , the universal $ jet , joe schenck , history of english literature , and the spedale degli innocenti , these sources do make and publish this my last day on the roads which no one engages in the foregoing instrument , and acknowledged that they were my parents . ” _END_

A.2 $N = 5,\ \beta = 0.5,\ \gamma = 0.05,\ F = 39$

START but , two days later , and ” be sure to come . ” _END_ _START_ a lean black cat , with a > b , then the other conspirators and marcus brutus stab caesar too ? ” _END_ _START_ in many instances the old flame of passion , and then the farm operator was a bare light bulb hanging from the ceiling . _END_ _START_ a year after this , before it was born , it seems to me only a month when he was in new york on his own . ” _END_ _START_ chemical and microbiological properties of interest are the most important aspects of their preparation , and their lives were in jeopardy , and his mind was not made up by the use of effective contraception . _END_ _START_ anonymous . _END_ _START_ any outcome of the disease (figure energy resources . _END_ _START_ even more important , such as : god , nature , and the philosophy of science : an introduction . ”) _END_ _START_ the size of all the particles of the solid phase . _END_ _START_ this has been especially true in the past and do not want a doctor , ” she said . _END_

A.3 $N = 5,\ \beta = 0.1,\ \gamma = 0.01,\ F = 39$

START if a approachable because even after probable cause has been established . _END_ _START_ a day later , there was another one , usually by the chief resident , and all these things are after - effects of , randy ticks working analogous to our infant church , and that he was ” a big guy like me er the deep their shadows fling , yon turret stands ; _END_ _START_ but should one not bringing the liquid is then passed on to the next and the next generation ’ s concerto , ” and in one of his books as well as a shrewd , but it ’ s entirety , and also changes color within the meaning of ss existence to a higher sovereign than the commonwealth , for the purpose of tracing paper on which a king and queen , and who had so much to do with this ! _END_ _START_ ivanych . _END_ _START_ kangaroos successful in receiving the word of god and the immortality of the soul , strong in its natural state . _END_ _START_ fifteen minutes all night . _END_ _START_ the total number of optional ” . _END_

A.4 $N = 5,\ \beta = 1,\ \gamma = 0.01,\ F = 35$

START of all the gods make me feel , ” promus ” vain philosophy ” may be bound together by a separately published , and bitter battle with the forces of evil . _END_ _START_ it uses a script or an hour but i sat on my couch , by an application of reinforcement principles as abstract terms about both phases before the war when the planet ’ s mensis . mark colorimetric estimation of small amounts of the drug are injected , and then the state religion and the other the pulse integrator beyond fuel mass proletariat in order to record the questionnaire made it possible to feed elijah . _END_ _START_ the first two harness find out the cause of his arrest was in cricket , or scholarship , or research ; _END_ _START_ and he is the value of the gii by silcoff , he ascended by the steps that will be fun , not a few (choreographically . _END_ _START_ i do not believe , or act with the same force and effect as if the value of f based on a strong degree of consciousness is characterized by elite and the sur - - rounded the traditional functional organization of the staff serven bet and more to pay , insisting that the jews are to be found in the literature indicating that he was ready for whatever reason , it sought by a molecule which his friend popular reputation has preceded me . _END_

A.5 $N = 5,\ \beta = 0.5,\ \gamma = 0.05,\ F = 35$

START fr . _END_ _START_ it is a place where thou canst sit till i call . ” _END_ _START_ health factors should determine if a ” real ” company or government entity , or at any ranger station for details . _END_ _START_ if you want to spend more time with them while life was new . ” _END_ _START_ their number , as before , that will i . my spirit was panted forth in anguish whilst thy pain made my heart mad , ” asked betty , my wife , and i know he ’ s obviously came to redeem the soul of america . _END_ _START_ more space is required for the financing of south america are reported to be in the vicinity , and whether any work , membar ; _END_ _START_ in the summer of jd . _END_ _START_ yet it was only the third thousandths telegraph as a ” phenomenological ” approach ” the tile lines , and established his office at philadelphia , konopczynski , getting them off , and what it all meant , but for final exams . _END_ _START_ here n is managed by the securities industry . _END_

A.6 $N = 5,\ \beta = 0.1,\ \gamma = 0.01,\ F = 35$

START a chapter in the history of philosophy , xlii (july , to the left , and the boy fell into a deep , deep sigh , and said , ’ call unto me , and forbid them not , ” he remarked , ” i want to put the day of the flood , like some proud minster ’ s pealing clock still ticked on the mantel . _END_ _START_ he wished he could do more service to the world , not coveting the distinction of sleeping in a house , the result of which was a hundred feet away . _END_ _START_ he continues to hold the position for a young woman : ” you lie ! _END_ _START_ she asked . _END_ _START_ iii . _END_ _START_ paddock (fall became members of the initiating agent , in a letter of april , when the records were made . _END_ _START_ while later , in the period of maximum social welfare . _END_ _START_ ” after breakfast , we discussed the use of the electro - platers ’ hand - book of the revolution . _END_

A.7 $N = 5,\ \beta = 1,\ \gamma = 0.01,\ F = 25$

START list handy . _END_ _START_ title varies : decimal part of their work , noco solve exceedingly simple , more properly , each of the remaining tubes . _END_ _START_ as you know , is far less marked than in some others the gay snuff - box in one hand as if to fend off her drawers and chemise or the gradual achievement of japanese , chinese , american , and german thought was funny . _END_ _START_ she would close the door of our room , our eyes refused to meet hers and helped her out of the central cities and leave our farms , ” in encyclopedia of councils of the ancient church than it is to sell an asset or liability will be , from approximately diretta da prime source of our prosperity . _END_ _START_ textiles are tuned in to our wishes , on the contrary , as coordinate motor company , petrie would you characterize your attitude taken by the case must be resolved on the basis of product that is extracted in meteorology , when there is no sinecure , and in addition they touch , is translated in the tops of the mountains , upon the mantel clock hammered regal gesture , but it was not of this world , to be administered by men exempt from the passions incident to human nature , as it is impossible to do more than provide yourself with a lamp and began to walk away from it . _END_

A.8 $N = 5,\ \beta = 0.5,\ \gamma = 0.05,\ F = 25$

START who bowed before her , ” how do we get the slope of the tangent - screw is used , a circular town feel . _END_ _START_ this was followed by a mournful chant . _END_ _START_ and if you scrub up . _END_ _START_ she took his hand and gave the dangerous smiles again . _END_ _START_ and bodenheim ’ s work . _END_ _START_ ” information technologies ” (hymes , pp . thac . _END_ _START_ sci . _END_ _START_ fluro - glendessary . _END_ _START_ so that ’ s not it . _END_ _START_ martialed , and sentenced to banishment . _END_ _START_ seek not to be loosed . _END_ _START_ the public ’ s contribution also bisexuality ; _END_ _START_ we moved into a hotel room without a private bath , which is itself used as an anesthetic to the traditions of all kinds , primary and secondary qualities is an unequal marriages , but also has given us too little responsibility for the project , it is impossible to make a definitive diagnosis . _END_

A.9 $N = 5,\ \beta = 0.1,\ \gamma = 0.01,\ F = 25$

START i love you , little one . _END_ _START_ the instrument / you are interested in , and that ’ s all that need be considered , for the time being the question of the exact mathematical center of the sun to fall and the wind to shiver amongst the leaves . _END_ _START_ antonia pantoja , sal , ” said trexler . _END_ _START_ every corporation , association , or business . _END_ _START_ superficial lymphatics of the lower limb . _END_ _START_ alcott , hospital sketches , ’ and ’ new ’ . _END_ _START_ leech lake , la . _END_ _START_ a barabara , is herewith given : in the latter it is more than seventeen hundred years , we may pause to consider the nature of their relationship with the state and federal governments both of the daughters of st . paul . ” _END_ _START_ for one thing , really , that you can be anything you want , ” said he gently , ” only the good die young , ” was the reply . _END_

A.10 $N = 5,\ \beta = 1,\ \gamma = 0.01,\ F = 10$

START the theatre had closed her tongue out at him and throws it away ; _END_ _START_ however , only a scotch - irish emigrant , ” in hebrew , w . r . hasbrouck hts . _END_ _START_ for example , you can add the tincture . _END_ _START_ oral history movement , that it was literally worth remembering that a man has a national reputation through his work with municipality , olimpico was once an ugly quarrel with captain marker , in which rawdon , wrong from the beginning at age discrimination than in the yellowbreast angeles ca * * concentration of force against women is more likely to attract competent people . _END_ _START_ to this day pills are made behind its tall prescription desk - - pills rolled out on its south american continent , where survival is good , whether we ask it or not ! ” _END_ _START_ although he does not include in column (hydroferrocyanic acid , xviii ; _END_

A.11 $N = 5,\ \beta = 0.5,\ \gamma = 0.05,\ F = 10$

START the long x cunningham , allan , ii . _END_ _START_ lessons for quarreling with each other . _END_ _START_ yes , he had just constructed . _END_ _START_ he re - experiences the last test , it is assented and accorded by the constitution of ireland . _END_ _START_ limiting liability to client . _END_ _START_ no question of its being ” an intervening castings . _END_ _START_ pimentel , d . , sanjai ” peters said , returning his smile . _END_ _START_ philocleon sinh (double x) ; _END_ _START_ she flung a stone at the beginning of this great struggle for human freedom are poorhouse : a social history of the third century b . c . and potentials of these receptors for the niece of the second type the names of the jurors so drawn up in a formal way , or the other statements , he is responsible for the exquisite sense of beauty - - the insulted lady , good - humouredly , that did not help . _END_ _START_ appropriations , pharmaceutiques bright and shine again in your place . ” _END_ _START_ then other less prominent on the edge of box - - office failures of the smoke and dust raised by the touch of the boston gazette . _END_

A.12 $N = 5,\ \beta = 0.1,\ \gamma = 0.01,\ F = 10$

START literally , with the assertion that the main body of divinity . ” _END_ _START_ for our problem we have to decide , as always , was a return to the status quo . _END_ _START_ although this is theoretically bad , an ’ she says no , i for my part can not be used as a load for the animal , while the other , who was coming in from the cold . the presence of an exocyclic methylene group . _END_ _START_ the little rock nine , ” the chief went on , ” because i shall not be disappointed though i should conclude it , if less be required according to the federal census . _END_ _START_ per cent (_END_ _START_ ii . _END_ _START_ , lombardy , rome , or jerusalem , or mecca , has only dropped a mysterious hint : laus illi qui transtulit servum suum ab oratorio haram ad oratorium remotissimum (june) we had no other memory of that tender claim , o mother - child home program . _END_ _START_ but when light energy is absorbed within it , and will understand when i write that i feel he should have to infer that they had the power to create . _END_ _START_ women and youth . _END_

A.13 $N = 5,\ \beta = 1,\ \gamma = 0.01,\ F = 0$

START akad . _END_ _START_ now there must surely be a dealer scalped and murdered at enjoyed very wide extent of a walnut dining room table . _END_ _START_ social danger of the hell of the lords . _END_ _START_ when the doctor . _END_ _START_ i ’ ll bring something out for aye . _END_ _START_ she wanted to see what a buffet he gave robyn , to grounde he went downstairs , meet , and , finally , of their weight and volume when oven to require that all applicants must prove (mercials in figure arteaga , heretical sovereign , whom heaven and earth . _END_ _START_ and from one of the credit must be flexibly used , however , the state ; _END_ _START_ the door of this sleeping world is taken to be positive) influence of such substances into the winter his academic functioning requires that saw them again . _END_ _START_ situation just for that ? ” _END_ _START_ ludwig the pious , emperor , took charge of the chief of the shipping act , sol , within the very limits of the right of property , and he got ? ” _END_

A.14 $N = 5,\ \beta = 0.5,\ \gamma = 0.05,\ F = 0$

START his tale of war and peace , a . d better bring in the man ’ s real character more than the greatest good , it is certainly very beautiful . _END_ _START_ lipsensi congo because it appeared that the bill of rights which would have been the first to do that . ” _END_ _START_ she bids me sign myself affectionately yours , joseph , consider turnover in a department store was a party that secures good - - conduces to happiness . _END_ _START_ mutilat _END_ _START_ the sulfonated $ mouth of the missouri river . _END_ _START_ the end is frequently located in a well - received by both the public and the international scientific community the issue is exhausted , a new pipe organ for conducting a reliable source of water and food . _END_ _START_ and a revival of life and quality of life develops into a broad grin . _END_ _START_ and oryzae to have written the book of natural history _END_ _START_ drill , saying , after this manner . _END_ _START_ hid ; _END_

A.15 $N = 5,\ \beta = 0.1,\ \gamma = 0.01,\ F = 0$

START vol . _END_ _START_ he ’ s out of the ” house of the dead . _END_ _START_ the heavy tax on land values . _END_ _START_ balio , united artists : the company built by the stars (madison : university of wisconsin press , cadet officer jones ’ discourse) , as well as in other portions of the body , and casting the dart , to beat and abuse the weaker judgements of scholars , and to keep the promises which he could defend himself , indeed almost before they ’ d be cold , both in india and abroad . _END_ _START_ first two pairs of pleopods are all but absent from the islands of the indian archipelago the etiquette of court life , and a laterally placed the sunny earth - day that the first gun is not the vocation of the christian ministry . _END_ _START_ but braves stepped forward to retrieve the prestige of his office for a couple of periods . _END_ _START_ target height , in quicktime player from the first team , ” he continued , pointing to a larger creative process . _END_ _START_ thickness of wall , which was a drawback , however , is there not good reason to believe that the region is simply connected . _END_

B Risk assessment retrospective

Hazards associated with excessive computer work were correctly identified in the original risk assessment. At one point during the writing of the final report, I needed to rest for a few days because of mild Repetitive Strain Injuries due to too much typing. No other risks were observed.