# Encryption Possessing Statistical Perfect Secrecy and Stealth*

**B. John Oommen† and Luis G. Rueda‡**

## 1 Highlights of the Invention

### 1.1 Patent Information

The invention discussed here has been patent protected and the details of the patent are as follows:

- **Title of Patent**: A Method For Encryption with Statistical Perfect Secrecy
- **Inventors**: B. John Oommen and Luis G. Rueda.
- **Owner of the Patent**: *Oommen Computer Consultants Inc.*, 5942 Third Line Road, North gower, ON: K0A2T0, Canada.
- **Associated Patents**:
  1. U.S. Patent No. 7,508,935 issued on March 24, 2009
  2. Canadian Patent No. 2,460,863 issued on April 26, 2011.
- **Significance of the Patent**: This patent solves a compression-based problem that was reported to be unsolved in the standard textbooks. Using this solution, we were able to invent an encryption which provides *Statistical Perfect Secrecy*. We believe that our solution is still the only reported one for this problem.

*The technology has been patent protected as explained in the text. The commercial history of the patent can be discussed separately with interested parties.

†Address of the first Inventor: *Chancellor's Professor* and *Fellow: IEEE; Fellow: IAPR*, School of Computer Science, Carleton University, 1125 Colonel By Dr., Ottawa, ON, K1S 5B6, Canada. E-mail: oommen@scs.carleton.ca.

‡Address of the second Inventor: *Professor*, School of Computer Science, University of Windsor, 401 Sunset Avenue, Windsor, ON, N9B 3P4, Canada. E-mail: lrueda@uwindsor.ca.

## 1.2 Overview

The hallmark of a "perfect" cryptosystem is the fundamental property called "Perfect Secrecy" [18]. Informally, this property means that for every input data stream, $\mathcal{X}$, the probability of yielding any given output data stream, $\mathcal{Y}$, is the same, and independent of the input. Consequently, there is no statistical information in the output data stream or ciphertext, about the identity and distribution of the input data or plaintext. A system possessing Perfect Secrecy yields an output sequence that is distributed like *random noise*. This means that an eavesdropper who examines the ciphertext output $\mathcal{Y}$, **cannot** deduce the input $\mathcal{X}$ from analyzing the statistical information in $\mathcal{Y}$.

The problem of attaining Perfect Secrecy was originally formalized by Shannon in 1949 [15]. Shannon [5, 15, 17, 18] showed that if a cryptosystem possesses Perfect Secrecy, then the length of the secret key must be at least as large as the Plaintext. This makes the development of a realistic perfect secrecy cryptosystem impractical, such as demonstrated by the Vernam One-time Pad. Developing a pragmatic encoding system that satisfies this property is an open problem that has been unsolved for many decades.

The invention described here guarantees *Statistical Perfect Secrecy* - a property closely related to the phenomenon of Perfect Secrecy. A system (including cryptosystems, compression systems, and in general, encoding systems) is said to possess *Statistical Perfect Secrecy* if all its contiguous output sequences of length $k$ are equally likely, for all values of $k$, independent of the input data stream, $\mathcal{X}$. Thus, a scheme that removes all statistical properties of the input stream also has the property of Statistical Perfect Secrecy. A system possessing Statistical Perfect Secrecy maximizes the entropy of the output computed on a symbol-wise basis.

It is easy to see that the phenomenon of *Statistical Perfect Secrecy* is related to the concept of Perfect Secrecy. It differs marginally from the former in that it is defined in terms of *all* possible subsequences of $\mathcal{Y}$ of length $k$ (for all $k$), and not in terms of the entire output sequence $\mathcal{Y}$. Additionally, since the property of Statistical Perfect Secrecy can characterize any system and not just a cryptosystem, there is no requirement of relating the size of the key to the size of the input, as required by Shannon's theorem.

## 1.3 Properties of the Invention

The invention which we present here has the following properties:

1. It is a stream cipher using an underlying philosophy that has not been previously

used in encryption. Indeed, the foundational mathematical tools used in solving the above open problems are those which are used in the design and analysis of stochastic Learning Automata (LA). It is pertinent to mention that this invention represents the pioneering use of LA in encryption.

2. The scheme encrypts guaranteeing Statistically Perfect Secrecy. Thus the output resembles random noise even for relatively small keys. The Statistical Perfect Secrecy phenomenon is not merely experimentally verified, but also formally and rigorously proven.

3. The cryptosystem passes all FIPS 140 tests on the standard benchmark corpuses.

4. The system also-demonstrates bit-wise, key-to-output, and input-output independence on the standard benchmark corpuses.

5. The output $\mathcal{Y}$ maximizes the entropy, $\mathcal{H}(\mathcal{Y})$, *independent* of *any* input data and *any* key.

6. The speed of the encryption decreases *linearly* with the key length, rendering it superior to many modern-day encryption schemes.

7. The scheme guarantees stealth and is thus ideal for steganography.

8. Since the scheme effectively removes all statistical information from the input stream, it does not lend itself to statistics-based cryptanalysis, and can thus be broken only by an exhaustive search of the entire key space.

9. One of the embodiments of the invention achieves on-line data compression, and thus it permits communication with an increased bandwidth.

## 1.4 The Kernel: The Solution to an Open Problem

To obtain a Statistical Perfect Secrecy encryption, we proceed from the first principles of coding theory, Shannon's entropy theory and the theory of stochastic learning, instead of using bit-wise operations (like XORs and the principles of feedback shift registers), elliptic functions, factoring of large numbers, or other NP-hard problems. The goal of the whole exercise is to develop a system generating "white-noise-like" output, where this output or Ciphertext is independent of the input or Plaintext. This is done by solving a problem which we refer to as the *Distribution Optimizing Data Compression (DODC)* problem, which has

been reported to be open even in modern-day textbooks [2]. This solution forms the kernel of our invention.

# 2 Open Problems

To show how the kernel is developed, we discuss in greater detail the open problem alluded to above.

When data has to be transmitted from one place to another or optimally stored, it is typically compressed and encrypted. The aim is to ensure that a much smaller compressed and secure file can be transmitted or stored, with no loss, to the receiver, who would then decode the encoded file to procure the original file. Viewed from this perspective, there are two fundamental problems involving

(*i*) *controlling the output probabilities*, and,

(*ii*) *achieving Statistical Perfect Secrecy.*

These two concepts are separately discussed in the next two sub-sections. We shall solve the former problem and utilize its solution in a cryptosystem possessing Statistical Perfect Secrecy and stealth.

## 2.1 Optimizing the Output Probabilities

Consider a system in which $\mathcal{X} = x[1] \dots x[M]$ is the plaintext data stream, where each $x[k]$ is drawn from a plaintext alphabet, $\mathcal{S} = \{s_1 \dots s_m\}$, and $\mathcal{Y} = y[1] \dots y[R]$ is the ciphertext data stream, where each $y[k] \in \mathcal{A}$ of cardinality $r$.

There are numerous schemes which have been devised for data compression/encoding. A survey of the field is found in [2, 14, 19].

The problem of obtaining arbitrary encodings of the output symbols has been studied by researchers for at least five decades. Many encoding algorithms (such as those of Huffman, Fano, Shannon, the arithmetic coding and others) have been developed using different statistical and structure models (e.g. dictionary structures, higher-order statistical models and others). They are all intended to compress data, but their major drawback is that they cannot control the probabilities of the output symbols.

This drawback can be explained as follows. Once the data encoding method has been specified, the user loses control of the contents of the encoded file. In other words, in all the

compression/encoding techniques known till today, the statistical properties of the output compressed/encoded file cannot be *controlled* by the user - they take on their values as a consequence of the statistical properties of the original file and the data encoding method in question.

A problem that has been open for many decades [2] is that of devising an encoding scheme, which when applied on a data file, compresses the file and simultaneously makes the file appear to be random noise. Thus, if the alphabet is binary (as is typically the case) the *input* probability of '0' and '1' could be arbitrary. The problem of the user specifying the *output* probability of '0' and '1' in the encoded file has been open. Indeed, if the user specifies the stringent constraint that the output probabilities of '0' and '1' be arbitrarily close to 0.5, the consequences are very far-reaching, resulting in the erasure of statistical information.

We assume that we are given a code alphabet $\mathcal{A} = \{a_1, \ldots, a_r\}$, and that the user specifies the desired output probabilities or frequencies of each $a_j \in \mathcal{A}$ in the encoded file by $\mathcal{F}^* = \{f_1^*, \ldots, f_r^*\}$. Such a rendering will be called an "entropy optimal" rendering. On the other hand, the user also simultaneously requires optimal, or even sub-optimal *lossless* data compression. Thus, after compressing a file, we are required to recover **exactly** the same file after the specified decompression process has been invoked. This problem, which we refer to as the "Distribution Optimizing Data Compression" (DODC) problem[1], is known to be unsolved. Its formal statement is given in Appendix A.

It is interesting to note that the problem is *intrinsically* difficult because the data compression requirements and the output probability requirements can be contradictory. Informally speaking, if the occurrence of '0' is significantly lower than the occurrence of '1' in the original file, designing an encoding method which compresses the file and which simultaneously increases the proportion of '0's to '1's is far from trivial.

The importance of the problem is reflected in the following paragraph as originally stated in [2] (pp. 76-77):

> " It would be nice to have a slick algorithm to solve Problem 1, especially in the case $r = 2$, when the output will not vary with different definitions of $d(\mathcal{F}, \mathcal{F}^*)$. Also, the case $r = 2$ is distinguished by the fact that binary channels are in widespread use in the real world.
>
> We have no such algorithm! Perhaps someone reading this will supply one some day..."

[1] In the more general context of not just compressing the plaintext, but encoding it, the problem will be referred to as the Distribution Optimizing Data Encoding (or "DODE") problem.

## 2.2 Statistical Perfect Secrecy

The problem of erasing the statistical distribution from the input data stream and therefore the output data stream, has fundamental significance in cryptographic applications. It is well known that any good cryptosystem should generate an output that has random characteristics [5, 15, 17, 18].

A fundamental goal in cryptography is to attain Perfect Secrecy [18]. Indeed, as mentioned earlier, it is well known that it is quite impractical to develop a cryptosystem possessing Perfect Secrecy because such a system must have a secret key whose length is at least as large as the size of the plaintext.

The phenomenon of Statistical Perfect Secrecy eliminates the constraint between the key length and the length of the plaintext. This is achieved by defining the corresponding probabilities in terms of the subsequences of arbitrary lengths, as opposed to the entire output ciphertext. Thus, a system (including cryptosystems, compression systems, and in general, encoding systems) is said to possess *Statistical Perfect Secrecy* if all its contiguous output sequences of length $k$ are equally likely, for all values of $k$, independent of $\mathcal{X}$.

More formally, using the notation of the previous subsection, a system is said to possess Statistical Perfect Secrecy if for every input $\mathcal{X}$ there exists some integer $j_0 \geq 0$ and an arbitrarily small positive real number $\delta_0$ such that for all $j > j_0$,$\Pr[y_{j+1}...y_{j+k}|\mathcal{X}] = \frac{1}{r^k} \pm \delta_0$ for all $k$, $0 < k < R - j_0$.

As mentioned earlier, a system possessing this property is correctly said to display *Statistical* Perfect Secrecy. This is because for all practical purposes and for all finite-lengthed subsequences, statistically speaking, the system behaves as if it, indeed, possessed the stronger property of *Perfect Secrecy*.

# 3 Mathematical Tool: Stochastic Learning Automata

The foundational mathematical tools used in solving the above open problems are those which are used in the design and analysis of Stochastic Learning Automata[2].

Stochastic Learning Automata [6] are learning machines, which interact with random environments. The aim of the learning exercise is for the automaton to learn the best action that the environment offers. This is achieved by a sequence of feedback interactions between

[2]The references cited here are, by no means, comprehensive or exhaustive. They are listed to merely state the contributions that Oommen and his co-authors have made to the field. More specifically, Oommen was promoted to be a *Fellow of the IEEE* for his specific contributions to the area of automata learning. The references also demonstrate that these techniques can be applied in a variety of application domains.

the automaton and the environment.

Oommen[3] and his collaborators[4] have published extensively in this area. Indeed, the fastest and most accurate algorithms currently reported are due to him and his co-authors [1, 7, 8, 11]. Oommen and his co-authors have also used learning automata techniques to solve a number of applied problems, including cryptanalysis of substitution ciphers [13], and more recently in obtaining approximately-optimal solutions to NP-hard problems such as the graph partitioning problem [9], and the string taxonomy problem [10]. The most recent application involves using learning automata for the capacity assignment problem in network design [12].

### 3.1 Philosophy of Solution: Learning Automata Blending

To protect our patent rights, the formal technique by which the open problem has been solved cannot be disclosed here. The following statements are intended to provide the reader with a *flavour* of what is done, and hopefully convince him/her that such a solution exists. The novelty and uniqueness of the solution should be evident.

The present solution for DODC utilizes the principles of stochastic learning automata to blend the exponentially-large number of possible output distributions in an adaptive manner. This blending is done in such a way that the output probabilities follow a tightly constrained stochastic process. Furthermore, this blending is both adaptive and time varying, and can be rendered key-dependent.

We are not aware of any related solution in either the field of learning automata or in cryptography.

## 4 Solution Proposed to Open Problems

### 4.1 Highlights of the Solution

We have solved the open DODC problem stated as Problem 1 in the Appendix for the case when the input alphabet is any $r$-ary alphabet and the output alphabet is binary. This solution can easily be extended to the multi-symbol output alphabet case. Our current solution uses stochastic learning methodologies (explained presently) to produce *state-of-the-art*

[3] A complete list of Oommen's results in this area can be found at www.scs.carleton.ca/~oommen.

[4] The second inventor, Rueda has also worked in this field. Together with Oommen, he has recently used learning automata to devise weak estimators of the parameters of probability distributions, and these methods have been shown to have powerful applications in analyzing data obeying non-stationary distributions.

compression. The solution is both optimal and lossless. Additionally, the scheme is adaptive, and so the *input* probabilities need not be estimated. Finally, the output probabilities are controllable so that the output quickly converges to the specified distribution of '0's and '1's.

This algorithm achieves amazing results for *any* file type, and with *any* probability distribution. Although this is not the primary stipulation, the compression achieved can be comparable to the compression ratios of the best available compression algorithms.

## 4.2 Solution for General Encoding Systems

Apart from solving the reported open DODC problem (as stated as Problem 1 in the Appendix), we have also solved the more general problem when the output probabilities are requested by the user, but where s/he does not require the output to be *compressed*. Indeed, in the most general setting, the user may require that while the output is entropy optimal, the output size is simultaneously maintained the same, or even increased. This represents the general Distribution Optimizing Data Encoding (DODE) problem.

By extending our solution to the basic DODC problem, we have also solved the DODE problem. The latter solution involves a new data structure referred to as the Oommen-Rueda Structure. Adaptively blending the exponential number of Oommen-Rueda Structures leads to specific structure-based restructuring rules, which, in turn, lead to formal techniques for entropy-optimal encoding and decoding processes. Indeed, using these rules, we have been able to solve the DODE problem for the most general case when the input alphabet is of cardinality $m$, and the output alphabet is of cardinality $r$.

## 4.3 Uniqueness of The Solution

We have used the DODC solution (or in the most general setting, the DODE solution) as the kernel to our encryption. To get an overall perspective of the uniqueness and novelty of our contribution, we refer to the following statements from [5] (pp. 191-192):

($i$) There are relatively few fully specified stream ciphers in the literature. This is because most stream ciphers used, in practice, tend to be proprietary and confidential.

($ii$) Feedback Shift Registers (FSR), and, in particular, Linear Feedback Shift Registers (LFSR) are the basic building blocks in most stream ciphers.

(*iii*) LFSRs can be combined using either nonlinear combining functions, nonlinear filtering functions, or using the output of one or more LFSRs to control the clock of one or more LFSRs.

(*iv*) Clock-controlled stream ciphers are also reported in the literature, examples of which are the alternate step generator and the shrinking generator.

In contrast to these, our invention:

(*i*) Does not utilize XORs or FSRs in *any* form as a basic building block.

(*ii*) Is not clock-controlled in any manner.

(*iii*) Is based on the solution to the open DODC problem mentioned above, which till today has not been used in encryption. *This philosophy lends itself as the new paradigm based on which we have designed our novel stream ciphers.*

(*iv*) Although DODE and its cryptographic enhancements are not FSR-based, we believe that they can be easily implemented in hardware.

(*v*) Most reported stream ciphers claim to *experimentally* yield random noise characteristics. However, the strength of our invention is the *novel underlying philosophy and paradigm* which are distinct from the ones traditionally used, and the rigorous *formal* proofs that the schemes do yield Statistical Perfect Secrecy.

(*vi*) The formal details of these proofs can be found in the patent papers for the invention. These have already been submitted to protect our patent rights. As it stands now, the International Patent (PCT) Office has submitted a very positive report of the patent application. The PCT Office has determined that the closest result to our invention was a publication/invention in 1995, but even that solution does not solve the problem we have solved. *It is pertinent to mention that all the 47 claims we asked for were evaluated to be novel, inventive, and to possess industrial applicability.*

This affirms the uniqueness and the novelty of our invention!

# 5 Cryptographic Enhancements of DODE

By virtue of its intrinsic properties, we have been able to incorporate DODE in three encryption schemes which perform *on-line* encryption/decryption. These are listed below as DODE$^{\wedge}$, DODE$^{+}$ and DODE$^{*}$:

(a) DODE$^{\wedge}$ and DODE$^{+}$: Since a DODE solution achieves Statistical Perfect Secrecy, it can be used to enhance encryption in a straightforward manner. DODE$^{\wedge}$ and DODE$^{+}$ are obtained by sequentially cascading the DODE module with an existing encryption. This subsequent encryption can achieved by using a public key or a private key system, and could include schemes such as the RSA, the AES or any of the well-known encryption schemes. If the encryption scheme does not preserve the distribution of the DODE module, we refer to such a cryptosystem as DODE$^{\wedge}$. As opposed to this, if the encryption scheme preserves the distribution of the DODE module, (as any good substitution or permutation cipher would) we refer to the cryptosystem as DODE$^{+}$.

(b) DODE$^{*}$ is obtained by incorporating a key-based DODE and a novel encryption method in a non-sequential manner. In this case, the two processes intricately augment each other, so that it is not possible to decompose DODE$^{*}$ into two mutually exclusive processes. Whereas in DODC (or DODE), asymptotic random noise is achieved after a few hundred of bits, this transient drawback is completely eliminated in DODE$^{*}$. Breaking DODC$^{*}$ (or DODE$^{*}$) requires the exhaustive search of the entire key space. It should be emphasized that the key can be arbitrarily large.

# 6 Cryptanalysis of DODE$^{*}$

To visit the issue of cryptanalyzing DODE$^{*}$, we briefly cite the following points from [4]:

(a) The most typical use of a stream cipher for encryption is to generate a keystream in a way that depends on the secret key, and then to combine this (typically using bit-wise XORs) with the message being encrypted.

(b) It is imperative that the keystream "looks" random, and that this is verified by passing a battery of statistical tests.

(c) A good stream cipher utilizes a keystream which has a very large period. Thus, if the period of the keystream is too short, the adversary can apply discovered parts of the keystream to help in the decryption of other parts of the ciphertext.

(d) A more involved set of structural weaknesses might offer the opportunity of finding alternative ways to generate part, or even the whole of the keystream. This gives rise to the measure of security known as the *linear complexity* of a sequence, where the linear complexity is the size of the LFSR that needs to be used to replicate the sequence. The authors of [4] state that a necessary condition for the security of a stream cipher is that the sequences it produces have a high linear complexity.

(e) Other attacks attempt to recover part of the secret key that was used. Apart from the most obvious attack of searching for the key by brute force, a powerful class of attacks can be described by the term divide and conquer.

Bearing these in mind, it should be mentioned that DODE* is not vulnerable to the typical cryptanalytic methods used for stream ciphers for the following reasons:

(a) As opposed to the currently available stream ciphers, DODE* obtains the ciphertext by utilizing the key in effecting the structure-based restructuring rules, and thereafter operating on the plaintext to generate the ciphertext.

(b) It should be emphasized that this is achieved in a single process which cannot be decomposed. DODE* is indeed a *stream cipher* because it processes the plaintext in a symbol-wise manner.

(c) With regard to the rigorous statistical tests that measure the correlation of bits in the ciphertext, it will be shown presently that the output of DODE* effectively passes all the FIPS-140 tests.

(d) Divide-and-conquer methods of cryptanalysis will not be effective for DODE* since its output passes rigid input-output and key-output statistical independence tests.

(e) With regard to the *period* of DODE*, we observe that although we have not yet determined the period of ciphertexts generated by DODE*, it can be easily shown that this period is *strictly greater* than that of *any* of the cryptographically-secure pseudo-random number generators reported.

(f) The *linear complexity* of the ciphertext generated by DODE* has not been computed. Furthermore, since DODE* does not use FSRs in any form, but utilizes techniques that have not been used in designing stream ciphers, we believe that this complexity is not easily computable, either. However, since the period of the output of DODE* is very

large, and since the output passes the rigid input-output and key-output statistical independence tests, we contend that the linear complexity of the ciphertext generated by DODE* is also correspondingly high.

Thus, DODE* is not vulnerable to the traditional techniques for attacking stream ciphers.

On the other hand, although other well-known cryptanalytic techniques, such as *linear cryptanalysis* and *differential cryptanalysis* are not typically useful in breaking stream ciphers (but rather, block ciphers), we argue that they will not assist in cryptanalyzing DODE* either. Indeed, the axioms that warrant these methods imperatively utilize the statistical information resident in the ciphertext.

DODE* cannot be broken using linear cryptanalysis, because this technique is based on the non-uniformity of the output distribution. This non-uniformity is the characteristic that is erased using the DODE kernel and its enhancements.

On the other hand, differential cryptanalysis is also based on the non-uniformity of the distribution of the data streams resulting from performing XOR (or similar) operations. Since (a) the output of DODE* possesses the Statistical Perfect Secrecy property, (b) the output of DODE* demonstrates independence for higher-order Markovian models, (c) the blending achieved by the stochastic learning is adaptive, and (d) the output empirically demonstrates an independence of the input (as will be shown presently), the resulting output obtained from performing XOR (or similar) operations between output sequences will *also* demonstrate random noise characteristics.

Thus DODE* will not be vulnerable to differential cryptanalysis.

# 7 Testing Results: DODC as an Encoding Scheme

The set of files used to evaluate our algorithm is universally used as a standard test suite, and is traditionally known as the Calgary Corpus. This suite has been extended to constitute the Canterbury Corpus [19]. The files are widely used to test and compare the efficiency of different compression methods, and represent a wide range of different data sources such as executable programs, pictures, source code, postscript source, etc.

DODC performs remarkably on the Calgary Corpus and the Canterbury Corpus. The erasing of the statistical properties in the data files is truly phenomenal. Also, as observed, the scheme is adaptive. Finally, there is no need to transmit any additional information to the decoder, other than the requested probability of 0 in the output, which can be specified beforehand. In our experiments, this value is set to be 0.5.

| File | $f_{PC}$ | $\hat{f}_0$ | $d(\hat{\mathcal{F}}, \mathcal{F}^*)$ |
|---|---|---|---|
| bib | 0.534628104 | 0.500002577 | 6.64064E-12 |
| book1 | 0.539274158 | 0.500000000 | 0 |
| book2 | 0.536233237 | 0.500173602 | 3.01376E-08 |
| geo | 0.524978249 | 0.499995693 | 1.85506E-11 |
| news | 0.534862968 | 0.500020293 | 4.11796E-10 |
| obj1 | 0.527373684 | 0.500007788 | 6.06479E-11 |
| obj2 | 0.528048693 | 0.500083078 | 6.90190E-09 |
| paper1 | 0.536035267 | 0.500030866 | 9.52716E-10 |
| pic | 0.737834157 | 0.500000587 | 3.44069E-13 |
| progc | 0.534078465 | 0.500057756 | 3.33578E-09 |
| progl | 0.531706581 | 0.499963697 | 1.31792E-09 |
| progp | 0.535076526 | 0.499992123 | 6.20480E-11 |
| trans | 0.532803448 | 0.499999065 | 8.73572E-13 |
| Average | 0.547575632 | 0.500025163 | 3.32360E-09 |

Table 1: Results obtained after running DODC and a traditional prefix coding Algorithm on files of the Calgary Corpus. Similar results are available for the Canterbury Corpus test suite.

To compare our method, we have also ranked our algorithm against a prefix coding scheme. Both the techniques have been tested using the same set of files.

The results obtained are detailed in Table 1. The column named $f_{PC}$ is the probability of '0's produced by the above-mentioned prefix coding scheme. The next column, named $\hat{f}_0$ represents the estimated probability of '0's produced by DODC. Observe that, for *all* the files tested, the probability of '0' is arbitrarily close to 0.5. The last column is the distance between the desired probabilities and those obtained by DODC. In *all* the cases, the distance is amazingly small. Moreover, the largest distance is not greater than 10E-08, even though the files were not artificially simulated, but "real life" files!

A plot of the distance (as defined in Problem 1) between the desired probability of '0', $f_0^* = 0.5$, and the actual probability of '0' obtained by DODC is depicted in Figure 1 as a function of the size of the output file. The x-axis, $n$, represents the number of bits produced in the output and the y-axis, $d(\hat{\mathcal{F}}, \mathcal{F}^*)$, represents the above-mentioned distance. Note that the distance decreases drastically with the number of bits produced in the output. It is clear that the probability of '0' tends to be arbitrarily close to 0.5 as the number of bits produced in the output increases indefinitely. Furthermore, the drop of this distance measure to values near the asymptotic value close to zero is also remarkable. *The scheme adaptively converges to the near-optimal statistical configuration extremely rapidly*!

# 8 Testing Results: DODE-based Cryptosystems

To demonstrate the power of systems involving DODE-based encryptions, two instantiations of DODE have been incorporated into a fully operational 48-bit-key prototype cryptosystem. Both of these have been rigorously tested for a variety of tests, and these results are cataloged below. The key management mechanism of this system allows the user to use a key composed of upper and lower-case characters, digits, and the blank space. In this prototype, by incorporating a special key-related filter, the transient behavior of DODC is erased. Thus DODC* generates a completely random-like output stream, even from the first bit! This can be observed in Figure 2, which depicts the plot of the distance after encrypting the file *paper1* (from the Calgary corpus) by using DODC*.

In all brevity, the properties of our prototype (and thus of a realistic instantiation of DODE*) are summarized below:

(*i*) The scheme yields $\Pr[y[j] = 0]$ and $\Pr[y[j] = 1]$ which are arbitrarily close to 0.5. Furthermore, we have done the following tests on the above-mentioned files:

*Frequency Test (monobit test)*: DODC* passes the chi-square test with a confidence level higher than 99%.

*Serial Tests (two-bit tests)*: DODC* passes the chi-square test with a confidence level higher than 99% for bits which are $k$-symbols-apart, where $k = 1, \ldots, 5$.

Furthermore, DODC* also passes all the FIPS-140 tests of randomness including the *Poker Test*, the *Runs Test* and the *Autocorrelation Test* [5] (pp. 181-183).

(*ii*) The speed of the encryption decreases *linearly* with the key length, rendering it superior to many modern-day encryption schemes.

(*iii*) The message obtained by "decrypting" using a key that differs from the true key by even a single bit, also has random noise characteristics. This is demonstrated by a suite of key-output tests. We also report the results of various input-output independence tests.

(*iv*) The scheme has been incorporated in a fully operational steganographic prototype by taking advantage of the random noise characteristics of the output.

(*v*) A speed enhanced prototype has recently been implemented. This prototype passes

all the FIPS 140-2 tests[5]. The enhanced prototype also passes the Markovian test, as well as the key/output and input/output independence tests for the increased key length. This prototype encrypts a file of **1 Mbye** in approximately **two** seconds on a Pentium 300 MHz system when using a key of length 64 bits[6]. It is written in the C programming language, and has been designed to be a stand-alone module, which can be easily invoked from any executable Windows program. It can also be compiled as a DLL library or a Unix utility. Implementing an Assembly-language module, which would run significantly faster, is a fairly straightforward task.

A more detailed catalogue of these results follow.

First of all, the results for the output probability obtained by testing DODC* on the Calgary Corpus is given in Table 2. Observe that the value of the distance between the attained distribution and the desired distribution is arbitrarily close to zero in every case. Similar results are available for the Canterbury Corpus test suite.

We have also obtained results for another instantiation which uses a solution for DODE* (as opposed to a solution for DODC*). These results, which use the DODE* kernel on the Calgary Corpus, are given in Table 3. Analogous results are available for the Canterbury Corpus test suite.

The variation of the distance as a function of the encrypted stream has also been plotted for various files. Figure 2 graphically plots the average distance obtained after encrypting file *bib* (from the Calgary Corpus) using the DODC* as the kernel. Observe that the graph attains their terminal value close to zero right from the outset – without any transient characteristics.

## 8.1 Output Markovian Modeling and Independence Analysis

The theoretical analysis of convergence and the empirical results discussed for DODC* from the above tables, were done by merely considering the probabilities of occurrence of the output symbols. The question of the dependencies of the output symbols is now studied using a Markovian and $\chi^2$ hypothesis testing analysis. To achieve this goal, the possibility of the output sequence obtained by DODC* being a *higher-order Markov model* is analyzed.

[5]The first FIPS 140-2 Draft was signed on May 25, 2001. These tests are more *restrictive* than those of the FIPS 140-1. Although the FIPS 140-2 have become the standard tests since only May 25, 2002, we have opted to use *these* test to show the power of the enhanced prototype. These results can be obtained from http://www.scs.carleton.ca/ oommen/papers/FIPS-140-2.pdf.

[6]In general, the speed of the enhanced prototype would decrease linearly with the key length. However, by taking into consideration certain machine-dependent issues, the encryption and decryption modules can be tuned so that the speed is essentially *independent* of the key length.

| File Name | $\hat{f}_{PC}$ | $\hat{f}_0$ | $d(\mathcal{F}, \mathcal{F}^*)$ |
|---|---|---|---|
| bib | 0.534890885 | 0.498926052 | 1.15336E-06 |
| book1 | 0.539382961 | 0.500425219 | 1.80811E-07 |
| book2 | 0.536287453 | 0.500349415 | 1.22091E-07 |
| geo | 0.526089998 | 0.501331183 | 1.77205E-06 |
| news | 0.534758020 | 0.499712212 | 8.28219E-08 |
| obj1 | 0.527805217 | 0.4997762 | 5.00864E-08 |
| obj2 | 0.527944809 | 0.499521420 | 2.29039E-07 |
| paper1 | 0.535224321 | 0.500632728 | 4.00345E-07 |
| progc | 0.535104455 | 0.499916321 | 7.00218E-09 |
| progl | 0.535598140 | 0.499278758 | 5.20190E-07 |
| progp | 0.535403385 | 0.499946669 | 2.84420E-09 |
| trans | 0.533495837 | 0.499784304 | 4.65248E-08 |
| Average | | | 2.74079E-07 |

Table 2: Empirical results obtained after encrypting files from the Calgary Corpus using DODC$^*$ as the kernel. The results are also compared with a traditional well-known encoding scheme. Similar results are available for the Canterbury Corpus test suite.

| File Name | $\hat{f}_{PC}$ | $\hat{f}_0$ | $d(\mathcal{F}, \mathcal{F}^*)$ |
|---|---|---|---|
| bib | 0.534890885 | 0.498730944 | 1.61050E-06 |
| book1 | 0.539382961 | 0.499440591 | 3.12938E-07 |
| book2 | 0.536287453 | 0.499593621 | 1.65144E-07 |
| geo | 0.526089998 | 0.500380707 | 1.44938E-07 |
| news | 0.534758020 | 0.499766932 | 5.43207E-08 |
| obj1 | 0.527805217 | 0.500416731 | 1.73665E-07 |
| obj2 | 0.527944809 | 0.499376045 | 3.89320E-07 |
| paper1 | 0.535224321 | 0.498987636 | 1.02488E-06 |
| progc | 0.535104455 | 0.500757891 | 5.74399E-07 |
| progl | 0.535598140 | 0.499958084 | 1.75695E-09 |
| progp | 0.535403385 | 0.500176402 | 3.11177E-08 |
| trans | 0.533495837 | 0.498818445 | 1.39607E-06 |
| Average | | | 3.5628E-07 |

Table 3: Empirical results obtained after encrypting files from the Calgary Corpus using a *DODE*-based kernel. The results are also compared with a traditional encoding scheme. Similar results are available for the Canterbury Corpus test suite.

This model, which considers conditional probabilities of a symbol given the occurrence of the $k$ previous symbols, is called a $k^{th}$-order model.

Consider the first-order model with the following conditional probabilities, $f_{0|0}(n)$ and $f_{1|0}(n)$ (where the latter is equal to $1 - f_{0|0}(n)$), $f_{0|1}(n)$, and $f_{1|1}(n)$ (which in turn, is $1 - f_{0|1}(n)$). These probabilities can also be expressed in terms of a $2 \times 2$ matrix form,

the underlying *transition matrix*. Since the asymptotic probabilities are $[0.5, 0.5]$, it is easy to see that this matrix has to be symmetric of the form $\begin{bmatrix} \delta & 1-\delta \\ 1-\delta & \delta \end{bmatrix}$, where $\delta = 0.5$ is the case when there is no Markovian dependence. The value of $\delta$ is estimated from the output sequence in a straightforward manner.

In order to analyze the independence of the symbols in the output, DODC* has been tested on files of the Calgary corpus and the Canterbury corpus. The requested probability of 0 in the output is $f^* = 0.5$. The estimated probability of 0 given 0 in a file of $R$ bits is obtained as follows: $\hat{f}_{0|0} = \frac{c_{0|0}(R)}{c_0(R)}$, where $c_{0|0}(R)$ is the number of occurrences of the sequence '00' in $\mathcal{Y}$, and $c_0(R)$ is the number of 0's in $\mathcal{Y}$. Analogously, $\hat{f}_{1|1}$ is calculated.

Assuming that $\mathcal{Y}$ is a sequence of random variables, the analysis of independence in the output is done by using a Chi-square hypothesis test of independence [16]. The empirical results obtained from executing DODC* on files of the Calgary corpus are shown in Table 4. Similar results are available for the Canterbury Corpus too.

| File name | $\frac{\hat{f}_{0\|0}+\hat{f}_{1\|1}}{2}$ | $\chi^2$ | Decision (98%) |
|---|---|---|---|
| bib | 0.4991533 | 0.00000726 | Indep. |
| book1 | 0.4997611 | 0.00000166 | Indep. |
| book2 | 0.4995139 | 0.00000196 | Indep. |
| geo | 0.4999662 | 0.00000096 | Indep. |
| news | 0.4992739 | 0.00000423 | Indep. |
| obj1 | 0.5008566 | 0.00000608 | Indep. |
| obj2 | 0.5005717 | 0.00000315 | Indep. |
| paper1 | 0.4995268 | 0.00000367 | Indep. |
| progc | 0.5009482 | 0.00001100 | Indep. |
| progl | 0.4977808 | 0.00005315 | Indep. |
| progp | 0.4994945 | 0.00002204 | Indep. |
| trans | 0.4999502 | 0.00000060 | Indep. |

Table 4: Results of the Chi-square test of independence in the output of the encryption that uses DODC* as the kernel. The encryption was tested on files of the Calgary corpus with $f^* = 0.5$. Similar results are available for the Canterbury Corpus test suite.

The second column represents the average between $\hat{f}_{0|0}$ and $\hat{f}_{1|1}$ which are calculated as explained above. The third column contains the value of the Chi-square statistic for $\hat{f}_{a_i|a_j}$, where $a_i, a_j$ are either 0 or 1, and the number of degrees of freedom is unity. The last column stands for the *decision* of the testing based on a 98% confidence level. Observe that for *all* the files of the Calgary corpus (and the Canterbury corpus) the output random variables are *independent*. This implies that, besides converging to the value 0.5 in the mean square sense,

the output of DODC* is also statistically independent as per the 98% confidence level.

Similar results are also available for higher-order Markovian models up to level five, in which the output of DODC* proves to be also statistically independent as per the 98% confidence level.

## 8.2 The FIPS 140 Statistical Tests of Randomness of DODE*

The results of the previous two subsections demonstrate the power of the encryptions for Statistical Perfect Secrecy (convergence to $f^* = 0.5$) and the Markovian independence. However, as pointed out in [4, 5], a good stream cipher (and in general, any good cryptosystem) must endure even the most stringent statistical tests. One such suite of tests are those recommended by the standard FIPS 140-1 ([5] pp. 181-183). These tests include the *frequency test*, the *poker test*, the *runs test*, and the *long runs test*. As suggested, these tests have been run for encrypted strings of length 20,000 bits, where the corresponding statistics for passing the tests are clearly specified. The empirical results obtained after encrypting the files of the Calgary Corpus using DODC* as the kernel are cataloged in Tables 5, 6, 7, 8, 9, and respectively. Notice that the encryption passes all the tests. Similar results are available for the Canterbury Corpus.

The results for the encryption that uses DODE* as the kernel are identical, and not included here to avoid repetition.

| File name | $n_{1\min}$ | $< n_1 <$ | $n_{1\max}$ |
|---|---|---|---|
| bib | 9,654 | 10,007 | 10,646 |
| book1 | 9,654 | 9,974 | 10,646 |
| book2 | 9,654 | 9,991 | 10,646 |
| geo | 9,654 | 10,009 | 10,646 |
| news | 9,654 | 10,000 | 10,646 |
| obj1 | 9,654 | 10,053 | 10,646 |
| obj2 | 9,654 | 9,866 | 10,646 |
| paper1 | 9,654 | 9,958 | 10,646 |
| progc | 9,654 | 10,006 | 10,646 |
| progl | 9,654 | 10,072 | 10,646 |
| progp | 9,654 | 9,973 | 10,646 |
| trans | 9,654 | 10,046 | 10,646 |

Table 5: Results of the Monobit test in the output of the encryption that uses DODC* as the kernel. The encryption was tested on files of the Calgary corpus. Similar results are available for the Canterbury Corpus test suite.

| File name | $X_{3\min}$ | $< X_3 <$ | $X_{3\max}$ |
|---|---|---|---|
| bib | 1.03 | 6.25 | 57.4 |
| book1 | 1.03 | 28.74 | 57.4 |
| book2 | 1.03 | 10.33 | 57.4 |
| geo | 1.03 | 13.50 | 57.4 |
| news | 1.03 | 18.65 | 57.4 |
| obj1 | 1.03 | 9.28 | 57.4 |
| obj2 | 1.03 | 11.00 | 57.4 |
| paper1 | 1.03 | 21.72 | 57.4 |
| progc | 1.03 | 11.15 | 57.4 |
| progl | 1.03 | 13.67 | 57.4 |
| progp | 1.03 | 7.71 | 57.4 |
| trans | 1.03 | 14.30 | 57.4 |

Table 6: Results of the Poker test (when $m = 4$) in the output of the encryption that uses DODC$^*$ as the kernel. The encryption was tested on files of the Calgary corpus. Similar results are available for the Canterbury Corpus test suite.

| File name | $\ell_{\text{run}}$ | $B_i/G_{i\min}$ | $< G_i$ | $B_i <$ | $B_i/G_{i\max}$ |
|---|---|---|---|---|---|
| bib | 1 | 2,267 | 2,482 | 2,445 | 2,733 |
| | 2 | 1,079 | 1,241 | 1,214 | 1,421 |
| | 3 | 502 | 612 | 672 | 748 |
| | 4 | 223 | 293 | 327 | 402 |
| | 5 | 90 | 173 | 146 | 223 |
| | 6 | 90 | 146 | 164 | 223 |
| book1 | 1 | 2,267 | 2,540 | 2,539 | 2,733 |
| | 2 | 1,079 | 1,305 | 1,336 | 1,421 |
| | 3 | 502 | 628 | 574 | 748 |
| | 4 | 223 | 264 | 295 | 402 |
| | 5 | 90 | 138 | 150 | 223 |
| | 6 | 90 | 143 | 149 | 223 |
| book2 | 1 | 2,267 | 2,463 | 2,532 | 2,733 |
| | 2 | 1,079 | 1,266 | 1,184 | 1,421 |
| | 3 | 502 | 627 | 636 | 748 |
| | 4 | 223 | 328 | 322 | 402 |
| | 5 | 90 | 165 | 169 | 223 |
| | 6 | 90 | 170 | 170 | 223 |
| geo | 1 | 2,267 | 2,494 | 2,517 | 2,733 |
| | 2 | 1,079 | 1,226 | 1,251 | 1,421 |
| | 3 | 502 | 660 | 613 | 748 |
| | 4 | 223 | 320 | 289 | 402 |
| | 5 | 90 | 158 | 157 | 223 |
| | 6 | 90 | 152 | 161 | 223 |
| news | 1 | 2,267 | 2,532 | 2,579 | 2,733 |
| | 2 | 1,079 | 1,255 | 1,261 | 1,421 |
| | 3 | 502 | 643 | 610 | 748 |
| | 4 | 223 | 319 | 293 | 402 |
| | 5 | 90 | 156 | 138 | 223 |
| | 6 | 90 | 176 | 154 | 223 |
| obj1 | 1 | 2,267 | 2,534 | 2,510 | 2,733 |
| | 2 | 1,079 | 1,257 | 1,230 | 1,421 |
| | 3 | 502 | 620 | 637 | 748 |
| | 4 | 223 | 313 | 344 | 402 |
| | 5 | 90 | 149 | 158 | 223 |
| | 6 | 90 | 156 | 168 | 223 |

Table 7: Results of the Runs test in the output of the encryption that uses DODC* as the kernel. The encryption was tested on files of the Calgary corpus. Similar results are available for the Canterbury Corpus test suite.

## 8.3 Statistical Tests of Key-Input-Output Dependency

Apart from the FIPS 140 tests described above, the encryption which uses DODC* has also been rigorously tested to see how the ciphertext varies with changes in the key and the

| File name | $\ell_{\text{run}}$ | $B_i/G_{i\min}$ | $< G_i$ | $B_i <$ | $B_i/G_{i\max}$ |
|---|---|---|---|---|---|
| obj2 | 1 | 2,267 | 2,406 | 2,494 | 2,733 |
| | 2 | 1,079 | 1,277 | 1,241 | 1,421 |
| | 3 | 502 | 640 | 618 | 748 |
| | 4 | 223 | 313 | 287 | 402 |
| | 5 | 90 | 143 | 164 | 223 |
| | 6 | 90 | 177 | 135 | 223 |
| paper1 | 1 | 2,267 | 2,427 | 2,476 | 2,733 |
| | 2 | 1,079 | 1,328 | 1,223 | 1,421 |
| | 3 | 502 | 560 | 611 | 748 |
| | 4 | 223 | 306 | 335 | 402 |
| | 5 | 90 | 161 | 144 | 223 |
| | 6 | 90 | 161 | 151 | 223 |
| progc | 1 | 2,267 | 2,540 | 2,516 | 2,733 |
| | 2 | 1,079 | 1,250 | 1,268 | 1,421 |
| | 3 | 502 | 604 | 618 | 748 |
| | 4 | 223 | 300 | 292 | 402 |
| | 5 | 90 | 152 | 167 | 223 |
| | 6 | 90 | 159 | 169 | 223 |
| progl | 1 | 2,267 | 2,550 | 2,494 | 2,733 |
| | 2 | 1,079 | 1,202 | 1,259 | 1,421 |
| | 3 | 502 | 660 | 586 | 748 |
| | 4 | 223 | 287 | 333 | 402 |
| | 5 | 90 | 146 | 176 | 223 |
| | 6 | 90 | 147 | 142 | 223 |
| progp | 1 | 2,267 | 2,582 | 2,576 | 2,733 |
| | 2 | 1,079 | 1,291 | 1,288 | 1,421 |
| | 3 | 502 | 580 | 601 | 748 |
| | 4 | 223 | 323 | 311 | 402 |
| | 5 | 90 | 152 | 154 | 223 |
| | 6 | 90 | 147 | 131 | 223 |
| trans | 1 | 2,267 | 2,457 | 2,462 | 2,733 |
| | 2 | 1,079 | 1,228 | 1,245 | 1,421 |
| | 3 | 502 | 642 | 587 | 748 |
| | 4 | 223 | 316 | 312 | 402 |
| | 5 | 90 | 146 | 187 | 223 |
| | 6 | 90 | 177 | 161 | 223 |

Table 8: Results of the Runs test in the output of the encryption that uses DODC* as the kernel. The encryption was tested on files of the Calgary corpus. Similar results are available for the Canterbury Corpus test suite.

plaintext. In particular, the probability of observing changes in the ciphertext when the key changes by a single bit has been computed. This is tabulated in Table 10. Observe that

| File name | Max. run length | Runs $\geq 34$ |
|---|---|---|
| bib | 13 | 0 |
| book1 | 12 | 0 |
| book2 | 14 | 0 |
| geo | 14 | 0 |
| news | 16 | 0 |
| obj1 | 14 | 0 |
| obj2 | 13 | 0 |
| paper1 | 15 | 0 |
| progc | 17 | 0 |
| progl | 16 | 0 |
| progp | 17 | 0 |
| trans | 14 | 0 |

Table 9: Results of the Long Runs test in the output of the encryption that uses DODC* as the kernel. The encryption was tested on files of the Calgary corpus. Similar results are available for the Canterbury Corpus test suite.

as desired, the value is close to 0.5 for all the files of the Calgary Corpus. Similarly, the probability of changing subsequent ciphertext bits after a given plaintext symbol has been changed by one bit has also been computed. This value is also close to the desired value of 0.5 in almost *all* files! The results are tabulated in Table 11. Similar results are also available for files of the Canterbury Corpus.

| File name | $\hat{p}$ |
|---|---|
| bib | 0.499610 |
| book1 | 0.499955 |
| book2 | 0.500052 |
| geo | 0.499907 |
| news | 0.499842 |
| obj1 | 0.499463 |
| obj2 | 0.500309 |
| paper1 | 0.500962 |
| progc | 0.498403 |
| progl | 0.500092 |
| progp | 0.499961 |
| trans | 0.499244 |

Table 10: Statistical independence test between the key and the ciphertext performed by modifying the key in one single bit on files of the Calgary corpus. $\hat{p}$ is the estimated probability of change in the ciphertext. Similar results are available for the Canterbury Corpus test suite.

| File name | $\hat{p}$ |
|---|---|
| bib | 0.492451 |
| book1 | 0.467442 |
| book2 | 0.501907 |
| geo | 0.500798 |
| news | 0.500089 |
| obj1 | 0.477444 |
| obj2 | 0.496427 |
| paper1 | 0.481311 |
| progc | 0.484270 |
| progl | 0.471639 |
| progp | 0.490529 |
| trans | 0.491802 |

Table 11: Statistical independence test between the plaintext and the ciphertext performed by modifying one single bit in the *plaintext* on files of the Calgary corpus. $\hat{p}$ is the estimated probability of change of change in the ciphertext. Similar results are available for the Canterbury Corpus test suite.

# 9 Applications of DODE-based Solutions

With regard to applications, both DODC and DODE (and their various respective instantiations) can be used as compression/encoding schemes in their own right. They can thus be

used in both data transmission, data storage and in the recording of data in magnetic media. However, the major applications are those that arise from their cryptographic capabilities, and include:

- Traffic on the internet
- E-commerce/m-commerce/e-banking
- E-mail/ftp
- Video conferencing
- Audio/Video streaming
- secure communications, including Secure IP, Secure VOIP, Secure Satellite, secure mobile and wireless communications.
- Network security and File/database security. To clarify this, let us assume that the files stored in a network are saved using one of the schemes advocated by this invention. In such a case, even if a hacker succeeds in breaking the firewall and enters the system(s), he will not be able to "read" the files he accesses, because they will effectively be stored as "random noise". The advantages of this are invaluable.
- Watermarking, in which the watermark is generated by invoking a DODE-based encryption.
- Steganography. This application will be discussed in greater detail presently.

Steganography is the ancient art of hiding information, which dates back from around 440 B.C [3][7]. One of the most common steganographic techniques consists of hiding information in images. To experimentally demonstrate the power of DODC$^*$ (and in the most general case, of DODE$^*$), the latter has been incorporated into a steganographic application by modifying 256-gray scale images with the output of DODC$^*$. By virtue of the Statistical Perfect Secrecy property of DODC$^*$, its use in steganography is unique.

Rather than using a sophisticated scheme, to demonstrate a *prima-facie* case, the Least Significant Bit (LSB) substitution approach for image "carriers" has been used. This technique consists of replacing the least significant bit of the $j_k^{th}$ pixel (or, to be more specific, the

[7]Apart from [3], we refer the reader to the following sites which describe in greater detail many of the available solutions to steganography and the related problem of watermarking: www.jjtc.com/Steganography/, www.jjtc.com/stegdoc/stegdoc.htm, www.dlib.org/dlib/december97/ibm/12lotspiech.html, and www.lnt.de/ hartung/watermarkinglinks.html.

byte that represents it) by the $k^{th}$ bit of the output sequence. The set of indices, $\{j_1, \ldots, j_R\}$, where $R$ is the size of the output ($R \leq |I|$, and $|I|$ is the size of the carrier image), can be chosen in various ways, including the, so called, *pseudo-random permutations*. Details of these techniques can be found in [3].

Although this is a fairly simplistic approach to applying DODC* in steganography, it has been deliberately used so as to demonstrate its power. An example will clarify the issue. In an experimental prototype, the well known "Lena" image has been used as the carrier file, which carries the output of DODC*. The original Lena image and the image resulting from embedding the output obtained from encrypting the file *fields.c* of the Canterbury corpus are shown in Figure 23. Apart from the two images being visually similar, their respective histograms pass the similarity test with a very high level of confidence. Such results are typical.

Observe that once the output of DODC* has been obtained, it can be easily embedded in the carrier by the most sophisticated steganography tools currently available [3], and need not be embedded by the simplistic LSB pseudo-random permutation scheme described above.

# 10 Conclusions

In this report we have presented a novel method of encryption whose kernel is based on a solution to a problem which has been reported open, namely the Distribution Optimizing Data Encoding (DODE) problem. The invention guarantees *Statistical Perfect Secrecy* - a property which is closely related to the phenomenon of Perfect Secrecy. The invention utilizes the DODE solution to yield a stream cipher for which the output resembles random noise even for relatively small keys. The Statistical Perfect Secrecy phenomenon is not merely experimentally verified for all the FIPS 140 tests on the standard benchmark corpuses, but also formally and rigorously proven. The resulting cryptosystem also demonstrates bit-wise, key-to-output, and input-output independence on the standard benchmark corpuses.

The speed of the encryption decreases *linearly* with the key length, rendering it superior to many modern-day encryption schemes. One of the embodiments of the invention achieves on-line data compression, and thus permits communication with an increased bandwidth. Finally, since the scheme effectively removes all statistical information from the input stream, it does not lend itself to statistics-based cryptanalysis, and can thus be broken only by an exhaustive search of the entire key space.

The report also includes a list of the possible applications of the DODE solution and DODE-based encryptions.

# References


[1] M. Agache and B.J. Oommen. Continuous and Discretized Generalized Pursuit Learning Schemes. In *Proceedings of SCI-2000, the 4th World Multiconference on Systemics, Cybernetics*, pages VII:270–275, Orlando, USA, July 2000.

[2] D. Hankerson, G. Harris, and P. Johnson Jr. *Introduction to Information Theory and Data Compression*. CRC Press, 1998.

[3] S. Katzenbeisser and F. Peticolas. *Information Hiding Techniques for Steganography and Digital Watermarking*. Artech House, 2000.

[4] RSA Laboratories. *RSA Laboratories' Frequently Asked Questions About Today's Cryptography, Version 4.1*. RSA Security Inc., 2000.

[5] A. Menezes, P. van Oorschot, and S. Vanstone. *Handbook of Applied Cryptography*. CRC Press, 1996.

[6] K. Narendra and M. Thathachar. *Learning Automata. An Introduction*. Prentice Hall, 1989.

[7] B.J. Oommen. Stochastic Searching on the Line and its Applications to Parameter Learning in Nonlinear Optimization. *IEEE Transactions on Systems, Man and Cybernetics*, 27:733–739, 1997.

[8] B.J. Oommen and M. Agache. Continuous and Discretized Pursuit Learning Schemes: Various Algorithms and Their Comparison. *IEEE Transactions on Systems, Man and Cybernetics*, pages 277–287, 2001.

[9] B.J. Oommen and T. De St. Croix. Graph Partitioning Using Learning Automata. *IEEE Transactions on Computers*, 45(2):195–208, 1995.

[10] B.J. Oommen and T. De St. Croix. String Taxonomy Using Learning Automata. *IEEE Transactions on Systems, Man and Cybernetics*, 27:354–365, 1997.

[11] B.J. Oommen and G. Raghunath. Automata Learning and Intelligent Tertiary Searching for Stochastic Point Location. *IEEE Transactions on Systems, Man and Cybernetics*, 28B:947–954, 1998.

[12] B.J. Oommen and T. D. Roberts. Continuous Learning Automata Solutions to the Capacity Assignment Problem. *IEEE Transactions on Computers*, 49(6):608–620, 2000.

[13] B.J. Oommen and J. Zgierski. A Learning Automaton Solution to Breaking Substitution Ciphers. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-15:185–192, 1993.

[14] K. Sayood. *Introduction to Data Compression*. Morgan Kaufmann, 2nd. edition, 2000.

[15] C. E. Shannon and W. Weaver. *The Mathematical Theory of Communications*. University of Illinois Press, 1949.

[16] G. Snedecor and W. Cochran. *Statitical Methods*. Iowa State University Press, 8th edition, 1989.

[17] W. Stallings. *Cryptography & Network Security: Principles & Practice*. Prentice Hall, 1998.

[18] D. R. Stinson. *Cyptography : Theory and Practice*. CRC Press, 2005 (Third Edition).

[19] I. Witten, A. Moffat, and T. Bell. *Managing Gigabytes: Compressing and Indexing Documents and Images*. Morgan Kaufmann, 2nd. edition, 1999.

# Appendix A: Formal Statement of the Problems

As stated in [2], the distribution optimizing data compression problem can be more formally written as follows:

**Problem 1.** Consider the source alphabet, $\mathcal{S} = \{s_1, \ldots, s_m\}$, with probabilities of occurrence, $\mathcal{P} = [p_1, \ldots, p_m]$, an input sequence, $\mathcal{X} = x[1] \ldots x[M]$, the code alphabet, $\mathcal{A} = \{a_1, \ldots, a_r\}$, and the optimal frequencies, $\mathcal{F}^* = \{f_1^*, \ldots, f_r^*\}$, of each $a_j$, $j = 1, \ldots, r$, required in the output sequence. The output is to be a sequence, $\mathcal{Y} = y[1] \ldots y[R]$, whose probabilities are $\mathcal{F} = [f_1, \ldots, f_r]$, generated by the encoding scheme $s_i \rightarrow w_i \in \mathcal{A}^+$ such that:

(*i*) The scheme must generate a *prefix* code, required for lossless compression.

(*ii*) The average code word length of the encoding scheme, $\overline{\ell} = \sum_{i=1}^{m} p_i \ell_i$, must be *minimal* among all the prefix codes, where $\ell_i$ is the length of $w_i$.

(*iii*) The distance, $d(\mathcal{F}, \mathcal{F}^*) = \sum_{j=1}^{r} |f_j - f_j^*|^\alpha, \alpha \geq 1$ must be *minimized*.

Typically, the distance is measured using the *mean square error*, by setting $\alpha = 2$.

The problem of achieving *Perfect Secrecy* is stated below.

**Problem 2.** Let $\mathcal{X} = x[1] \ldots x[M]$ be the plaintext, where each $x[k]$ is drawn from a source alphabet, $\mathcal{S} = \{s_1 \ldots s_m\}$. Let $\mathcal{Y} = y[1] \ldots y[R]$ be the ciphertext, where each $y[k] \in \mathcal{A}$, the ciphertext alphabet.

A cryptosystem has *Perfect Secrecy* if $\Pr[\mathcal{X}|\mathcal{Y}] = \Pr[\mathcal{X}]$.

By Bayes theorem, this is equivalent to requiring that $\Pr[\mathcal{Y}|\mathcal{X}] = \Pr[\mathcal{Y}]$.

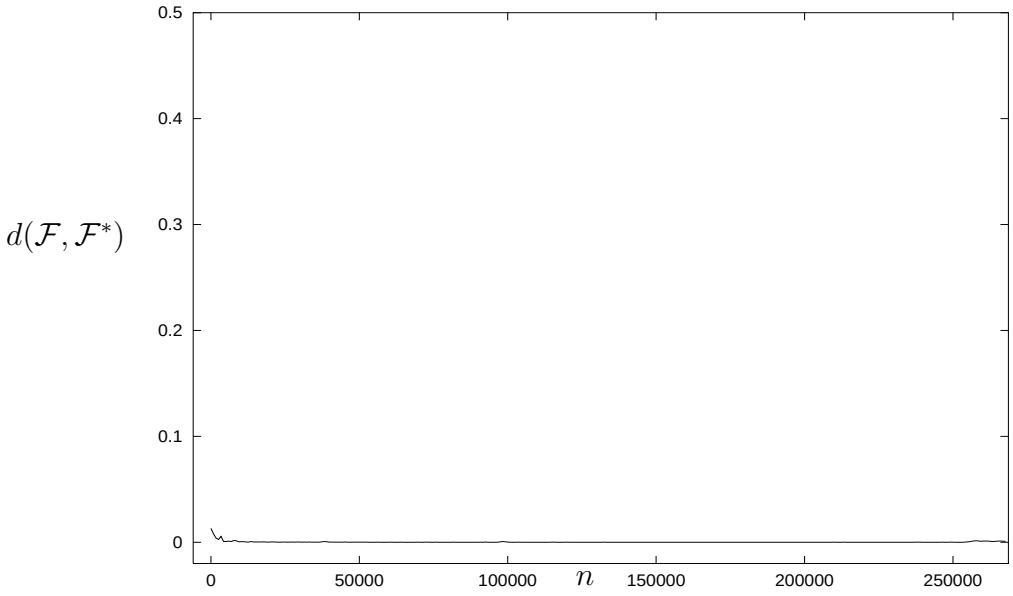


Figure 1: Distance obtained after encrypting the file *paper1* by using DODC, where the desired probability of zero is $f_0^* = 0.5$.

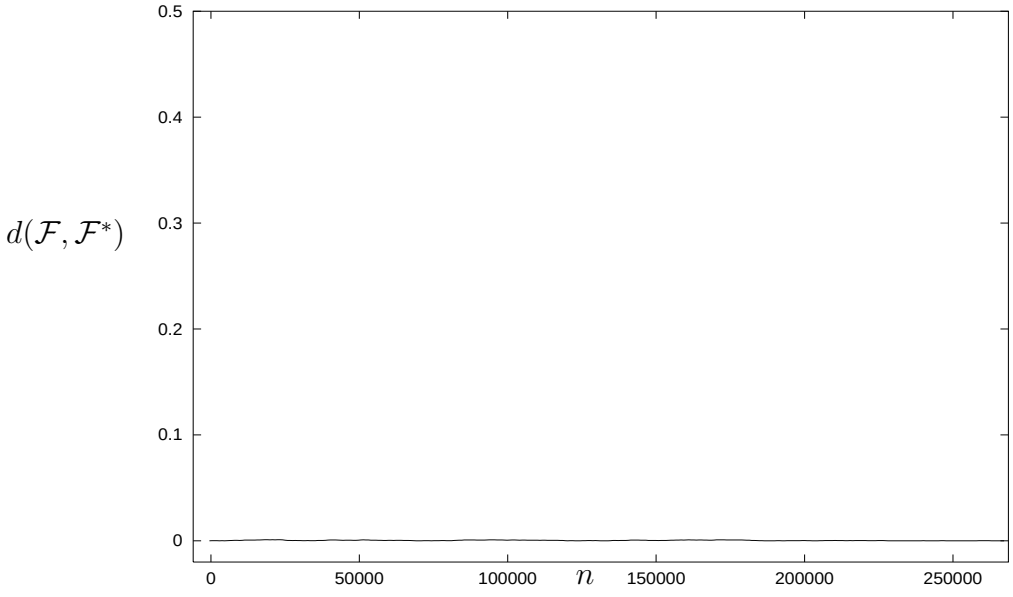


Figure 2: Plot of the distance (as defined in Problem 1) for the file *paper1* which was encrypted by using DODC$^*$. The requested probability of zero in the output is 0.5.

Figure 3: The original carrier image and the resulting image after applying steganographic techniques to the output of the file *fields.c* from the Canterbury corpus. The steganographic method used is the fairly straightforward Least Significant Bit approach.