ERROR CORRECTING CODES

EDDY LI

Abstract. Error correction is an important field of study in applied mathematics. This paper gives an overview of common error correction methods. We explore some intuitive error correcting codes, introduce some linear algebraic terminology, and finally prove that the famous $(7, 4)$-Hamming code is a valid and efficient way to correct errors in messages.

0. Introduction

Suppose that you want to send a message by transmitting a series of binary signals over a noisy channel that might corrupt 0 signal for a 1 signal, or vice versa. In some circumstances, the effects of even a single error would be disastrous, as an auxiliary encoding scheme might completely muddle the intended message. The natural question to ask would be: can we devise an algorithmic system that not only can detect an error in our transmission, but also correct it by restoring the original, intended message? In a nutshell, this question is the overarching motivation behind the whole study known as error correction.

In this paper, we will analyze several examples of well-known error correction methods. We first lay down the very basic groundwork in this section. In Section 1, we will look at the easiest way to correct errors in strings; we generalize this in Section 2 using linear algebra. In Section 3, we analyze the important $(7, 4)$-Hamming code and its validity. In Section 4, we will introduce some more terminology associated with error correction, using it to give another proof of the validity of the $(7, 4)$-Hamming code in Section 5. Finally, in Section 6, we consider some extensions of the previous methods.

As promised, we now set down the very basic terminology that we will refer to.

Definition 0.1. An *alphabet* A is any set of characters.

Definition 0.2. The set A^n consists of all strings of length n with characters in some set A.

Definition 0.3. Let a and b be nonnegative integers and let s be a string in A^a. An *error-correcting code*, or *ECC*, is a function $f : A^a \to A^b$ such that if any one character of $f(s)$ is erroneous, to create some string s'_f, we can always deduce s from s'_f.

Definition 0.4. The field $\mathbb{F}_2$ consists of the operations of additional and multiplication modulo 2 on the set $\{0, 1\}$.

Finally, from now on, we will always take our alphabet to be $\{0, 1\}$, the set corresponding to the field $\mathbb{F}_2$, where all operations are taken modulo 2. We also assume that there is at most one error in each transmitted string.

Date: March 20, 2023.

1. Intuitive ECCs

Now, we will analyze some of the most intuitive ECCs. Consider one such ECC below.

Theorem 1.1. *Suppose we wish to send a message* $\mathbf{v} \in \mathbb{F}_2$ *consisting of only one entry. Then the function* $f : \mathbb{F}_2 \to \mathbb{F}_2^3$ *given by*

$$f\left(\begin{bmatrix} x \end{bmatrix}\right) = \begin{bmatrix} x \\ x \\ x \end{bmatrix}$$

is a valid ECC.

Proof. Suppose we apply f to our vector

$$\mathbf{v} = \begin{bmatrix} x \end{bmatrix}$$

to get the transmitted vector $\mathbf{t} = f(\mathbf{v})$, and suppose that the vector $\mathbf{r} \in \mathbb{F}_2^3$ is received. Under the assumption that we have at most one corruption of $\mathbf{t}$, it follows that either these two vectors are equal or that the two vectors differs from each other by exactly one entry.

If we have the former, then the three entries of $\mathbf{r}$ will all equal x; if the latter, then $\mathbf{r}$ will have two entries that are equal to x, forcing the other entry to be equal to $x+1$ by definition of $\mathbb{F}_2 = \{0, 1\}$. Either way, at least two of the three entries of $\mathbf{r}$ equal x, so we can restore $\mathbf{v}$ simply by finding the character among the entries of $\mathbf{r}$ that appear the most. ■

Theorem 1.2. *Let* $f : \mathbb{F}_2^2 \to \mathbb{F}_2^5$ *be a function such that*

$$f\left(\begin{bmatrix} x \\ y \end{bmatrix}\right) = \begin{bmatrix} x \\ x \\ y \\ y \\ x+y \end{bmatrix}.$$

Then f *is an ECC.*

Proof. Our function f is a slightly more complicated now, but we can use a similar method as above. Using similar notation as before, let $\mathbf{v}$ be the original vector, so that

$$\mathbf{v} = \begin{bmatrix} x \\ y \end{bmatrix},$$

let $\mathbf{t} = f(\mathbf{v})$ be the transmitted vector, and let $\mathbf{r}$ be the received vector. As before, $\mathbf{t}$ and $\mathbf{r}$ differ by at most one entry. Thus, let

$$\mathbf{r} = \begin{bmatrix} z_1 \\ z_2 \\ z_3 \\ z_4 \\ z_5 \end{bmatrix}$$

for some $a, b, c, d, e \in \mathbb{F}_2$. We now do casework on the entry in $\mathbf{r}$ that gets corrupted, if any.

First, if z_5 is corrupted or none of the characters are tampered with, then we can easily deduce that $x = z_1 = z_2$ and $y = z_3 = z_4$. If either z_1 or z_2 is tampered with, then we must have $y = z_3 = z_4$ and $x + y = z_5$, implying that $x = z_5 - y = z_5 - z_3$, from which we can restore both $\mathbf{v}$ and $\mathbf{t}$. Using a symmetrical strategy for the case where one of z_3 and z_4 is corrupted, it follows that we can always deduce $\mathbf{v}$ and $\mathbf{t}$ from $\mathbf{r}$, so f is a valid ECC. ■

Now, it would be helpful for us to introduce a metric to determine the efficiency of ECCs. Intuitively, the more efficient the ECC, the less space it should use to encode all the error detection machinery. Thus, it makes sense to define our efficiency of some ECC f as the ratio of the number of entries in our original vector $\mathbf{v}$ to the number of entries of $f(\mathbf{v})$, which is sophisticated enough such that it is immune to single-character tampering. This implies the following definition.

Definition 1.3. Suppose that an ECC takes in vectors from $\mathbb{F}_2^a$ and outputs vectors in $\mathbb{F}_2^b$, for fixed $a, b \in \mathbb{Z}$. Then the *efficiency* of this ECC is $\frac{a}{b}$.

In our above two examples, our (a, b) are respectively $(1, 3)$ and $(2, 5)$, thus yielding efficiencies of $\frac{1}{3} = 0.333$ and $\frac{2}{5} = 0.400$. Both are less than $\frac{1}{2}$, and it remains for us to see if there exists some ECC with higher efficiency.

2. Linear ECCs

Now, we broaden the types of ECCs to form the class of linear ECCs. As we shall soon see, both of our examples considered in the previous section count as linear ECCs.

Definition 2.1. An ECC $f : \mathbb{F}_2^a \to \mathbb{F}_2^b$ is linear if there exists some $b \times a$ matrix G, with entries in $\mathbb{F}_2$, such that

$$f(\mathbf{v}) = G\mathbf{v}$$

for all $\mathbf{v} \in \mathbb{F}_2^a$. We call G the *generator matrix*.

The above definition simplifies the task of describing our ECC system. Specifically, notice that the examples of ECCs that we have explored previously all output vectors whose entries are linear combinations of the entries of the input vector, giving us a clear motivation to represent our ECC as being equivalent to a matrix-vector product or a linear transformation.

However, after simplifying our notation for encoding $\mathbf{v}$ into $f(\mathbf{v}) = G\mathbf{v}$ using the ECC f, it would be great if we could have an equally simple and accurate way to restore the original message $\mathbf{v}$ from the received message $\mathbf{r}$, or at least to decide if the received message $\mathbf{r}$ was tampered with or not. We will deal with this question in the following definition.

Definition 2.2. Suppose that there exists some matrix H, also with entries in $\mathbb{F}_2$, such that if a message $\mathbf{r}$ was received, then $H\mathbf{r} = \mathbf{0}$ iff $\mathbf{r}$ was not corrupted by one character. We call the matrix H the *parity check matrix*.

The above definitions makes the notions of encryption and deduction much more mathematically concrete. However, our parity check matrix H is much more powerful than merely identifying if there is an error $\mathbf{r}$, as we shall illustrate below.

Theorem 2.3. *Let f be some ECC with parity check matrix H. Suppose that the vector $\mathbf{r}$ is received. Then $H\mathbf{r} = \mathbf{0}$ iff $\mathbf{r}$ contains no error; otherwise, if the ith entry of $\mathbf{r}$ is erroneous, our check matrix output $H\mathbf{r}$ must equals the ith column of H.*

Proof. The first part of the statement, that $H\mathbf{r} = \mathbf{0}$ iff $\mathbf{r}$ has no error, immediately follows from the definition of a check matrix H.

Now suppose that $H\mathbf{r} \neq \mathbf{0}$, and suppose that the ith entry of $\mathbf{r}$ has been corrupted. It thus follows that $\mathbf{r} = \mathbf{t} + \mathbf{e}_i$, where $\mathbf{t}$ is the transmitted vector and $\mathbf{e}_i$ is the ith standard basis vector; this is due to the fact that the jth entry of $\mathbf{e}_i$ is 1 if $i = j$ and is 0 otherwise, so adding $\mathbf{e}_i$ to $\mathbf{t}$ toggles the ith entry and leaves everything else unchanged.

Since $\mathbf{t}$ is transmitted, it has never been corrupted, so $H\mathbf{t} = \mathbf{0}$ by definition. We thus have

$$\begin{aligned} H\mathbf{r} &= H(\mathbf{t} + \mathbf{e}_i) \\ &= H\mathbf{t} + H\mathbf{e}_i \\ &= \mathbf{0} + H\mathbf{e}_i \\ &= H\mathbf{e}_i. \end{aligned}$$

We observe that $H\mathbf{e}_i$ is the acquired sum when the ith column vector of H is given a weight of 1 all other columns are weighted by 0, implying that $H\mathbf{r} = H\mathbf{e}_i$ equals the ith column vector of H, as desired. ■

This theorem is quite powerful: not only does it mean that H can detect errors, we see that H can correct them too. It follows that our matrix H can have no column equal to the zero vector; it also cannot have two equal column vectors in distinct positions. In both of these cases, we would be unable to correct an error since it would be impossible to discern the incorrect bit in $\mathbf{r}$ without a guarantee of uniqueness, making f invalid. Finally, our above work also shows how HG must equal the zero matrix; we will leave this as an exercise for the reader.

Because the dimensions of G is $a \times b$, notice that finding an ECC with optimal efficiency can be visualized as finding some valid G whose dimensions are as close to a square matrix as possible.

3. Hamming Codes

Now that we have developed most of our necessary equipment, we will now introduce the famous $(7, 4)$-Hamming code in this section. This was developed by Richard Hamming in 1950, and our system takes in blocks of four bits and replaces them with blocks of seven, thus explaining the prefix $(7, 4)$. The precise definition, in terms of generator and parity check matrices G and H, are given below.

Definition 3.1. The $(7, 4)$-*Hamming code* has generator and parity check matrices

$$(G, H) = \left(\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \right).$$

Theorem 3.2. *The above system is a valid linear ECC.*

Proof. First, we will show that H is a valid parity check matrix corresponding to G. To do so, define the vector $\mathbf{t} \in \mathbb{F}_2^7$ to equal $G\mathbf{v}$, where we have some vector

$$\mathbf{v} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}.$$

Then, it follows that the transmitted vector $\mathbf{t}$ is given by

$$\begin{aligned} \mathbf{t} &= G\mathbf{v} \\ &= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix} \\ &= \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_2 + x_3 + x_4 \\ x_1 + x_3 + x_4 \\ x_1 + x_2 + x_4 \end{bmatrix}. \end{aligned}$$

Next, we compute $H\mathbf{t} = (HG)\mathbf{v}$, so that

$$\begin{aligned} H\mathbf{t} &= \begin{bmatrix} 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 1 & 0 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_2 + x_3 + x_4 \\ x_1 + x_3 + x_4 \\ x_1 + x_2 + x_4 \end{bmatrix} \\ &= 2 \begin{bmatrix} x_2 + x_3 + x_4 \\ x_1 + x_3 + x_4 \\ x_1 + x_2 + x_4 \end{bmatrix} \\ &= \mathbf{0}, \end{aligned}$$

implying that $H\mathbf{r}$ will equal the zero vector if there exists some $\mathbf{v}$ such that $\mathbf{r} = G\mathbf{v}$. This process is reversible, so it follows that $H\mathbf{r} = \mathbf{0}$ if and only if the received data $\mathbf{r}$ equals the transmitted data $\mathbf{t}$, and it follows that H is a valid parity check matrix for G.

Using our previous results, we have seen that if $H\mathbf{r} \neq \mathbf{0}$, then the value of $H\mathbf{r}$ must equal to one of the columns of H; if this column is the ith column, reading from left to right, then the ith entry of $\mathbf{r}$ has been tampered with. Observing that each of the seven columns are distinct from each other while also being nonzero, we can thus deduce which entry, if any, of $\mathbf{r}$ is corrupted by computing $H\mathbf{r}$, so we can thus restore $\mathbf{t}$ and therefore $\mathbf{v}$, implying that the Hamming code is a valid ECC. ■

The $(7,4)$-Hamming code has an efficiency of $\frac{4}{7} = 0.571$ as our $f(\mathbf{v}) = G\mathbf{v}$ sends vectors in $\mathbb{F}_2^4$ to $\mathbb{F}_2^7$. Compared with the earlier error codes that have efficiencies of 0.333 and 0.400, our $(7,4)$-Hamming code is a big improvement over the other ECCs in terms of efficiency. In fact, notice that $4 = 2^2$ and $7 = 2^3$, and the columns of H enumerate all the nonzero vectors in $\mathbb{F}_3$. This is not a coincidence, as the $(7,4)$-Hamming code compresses a wide range of information into a small amount of space.

4. Codewords and Hamming Distances

We have proved that the Hamming codes are a valid and efficient method of error correction. However, our proof does not answer the question of how we were able to get H from G, making our proof seem unmotivated and the fact that the columns of H enumerate $\mathbb{F}_2^3 \backslash \{\mathbf{0}\}$ seem somewhat coincidental. In this section, we will build up some fundamental definitions and corollaries that will give us a proof of the validity of the $(7,4)$-Hamming code without relying on H at all.

We first need to define a set consisting of the range of our matrix transformation G over all vectors in the field $\mathbb{F}_2^a$.

Definition 4.1. Let $f : \mathbb{F}_2^a \to \mathbb{F}_2^b$ be some ECC with generator matrix G, and let $C \subseteq \mathbb{F}_2^b$ be its range. Then C is the set of *codewords.*

In our previous work, we have intuitively seen that the farther apart the codewords are, the easier it is to determine the original message $\mathbf{t}$ from the received one. Think of it in this way: you have $|C|$ targets, each one representing an element of C, and someone throws an arrow, representing $\mathbf{r}$. Then the target closest to $\mathbf{r}$ is more likely to be the target $\mathbf{t}$ that they intended to hit. Intuitively, you would have an easier time determining $\mathbf{t}$ from $\mathbf{r}$ if the targets are farther apart rather than closely packed together. The worst possible situation would be that there exist several dartboards, each with the same probability that your friend picked it: you would then have no idea which dartboard it really was. The definitions below serve to quantify this conceptual idea.

Definition 4.2. The *Hamming distance* of two vectors $\mathbf{c}_1$ and $\mathbf{c}_2$ is equal to the number of bits in which the two differ. We denote the Hamming distance as $d_H(\mathbf{c}_1, \mathbf{c}_2)$.

Definition 4.3. The *minimum distance* δ associated with some set C of codewords is the minimum distance of any two elements in C. In other words, we have

$$\delta = \min_{\mathbf{c}_1, \mathbf{c}_2 \in C, \mathbf{c}_1 \neq \mathbf{c}_2} d_H(\mathbf{c}_1, \mathbf{c}_2).$$

Now, we prove a useful result relating codewords to error correction.

Theorem 4.4. *An ECC whose codeword set has a minimum distance of δ can correct at most $\frac{\delta-1}{2}$ errors.*

Proof. Let $\mathbf{c}_1$ and $\mathbf{c}_2$ be two codewords such that their Hamming distance is minimized, so that $d_H(\mathbf{c}_1, \mathbf{c}_2) = \delta$. Now, suppose that our ECC can correct at most k errors, and suppose that $\mathbf{r}$ is a received message with $d_H(\mathbf{c}_1, \mathbf{r}) = k$. Then, we must have $d_H(\mathbf{c}_2, \mathbf{r}) > k$, or equivalently, $d_H(\mathbf{c}_2, \mathbf{r}) \geq k + 1$; otherwise, it is impossible for us to discern whether the intended transmitted message was $\mathbf{c}_1$ or $\mathbf{c}_2$.

A variant of the triangle inequality can be applied in this case: some of the changes from $\mathbf{c}_1$ to $\mathbf{r}$ and some of those from $\mathbf{r}$ to $\mathbf{c}_2$ might cancel out, implying that

$$\begin{aligned}\delta &= d_H(\mathbf{c}_1, \mathbf{c}_2)\\ &\geq d_H(\mathbf{c}_1, \mathbf{r}) + d_H(\mathbf{r}, \mathbf{c}_2)\\ &\geq k + (k+1)\\ &= 2k+1,\end{aligned}$$

implying that

$$k \leq \frac{\delta - 1}{2},$$

as desired. ■

A direct corollary is the following result, which is quite important because single-character corruptions are the focus for the $(7, 4)$-Hamming code.

Corollary 4.5. *An ECC that can correct single-character corruptions must have* $\delta \geq 3$.

Proof. This directly follows from our previous theorem: since some C with some certain δ can correct $\frac{\delta-1}{2}$ errors, and we must have $\frac{\delta-1}{2} \geq 1$, it follows that $\delta \geq 2 \cdot 1 + 1 = 3$. ■

5. Alternate Verification of the Hamming code

In this section, we will give an alternate way to prove that the Hamming code is valid, without relying on the parity check matrix H at all. This method is perhaps more motivated and intuitive, even if somewhat more tedious than the previous approach.

Theorem 5.1. *The set of codewords of the* $(7, 4)$*-Hamming code has* $\delta = 3$.

Proof. Consider two distinct vectors $\mathbf{v}$ and $\mathbf{w}$ in $\mathbb{F}_2^4$. From our previous work, it suffices to prove that $d_H(G\mathbf{v}, G\mathbf{w}) \geq 3$ for all vector pairs such that $\mathbf{v} \neq \mathbf{w}$. Hence, suppose that

$$(\mathbf{v}, \mathbf{w}) = \left(\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}, \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \end{bmatrix} \right),$$

which would imply that

$$(G\mathbf{v}, G\mathbf{w}) = \left(\begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_2 + x_3 + x_4 \\ x_1 + x_3 + x_4 \\ x_1 + x_2 + x_4 \end{bmatrix}, \begin{bmatrix} y_1 \\ y_2 \\ y_3 \\ y_4 \\ y_2 + y_3 + y_4 \\ y_1 + y_3 + y_4 \\ y_1 + y_2 + y_4 \end{bmatrix} \right).$$

Consider the value of $k = d_H(\mathbf{v}, \mathbf{w})$. Because $\mathbf{v} \neq \mathbf{w}$, we know that $k \in \{1, 2, 3, 4\}$. Also, since $\mathbf{v}$ and $\mathbf{w}$ are embedded into the first four entries of $G\mathbf{v}$ and $G\mathbf{w}$, we see that $d_H(G\mathbf{v}, G\mathbf{w}) \geq k$. If k is 3 or 4, then $d_H(G\mathbf{v}, G\mathbf{w}) \geq k \geq 3$, and our desired result immediately follows.

Now suppose that $k = 1$, such that $x_j = y_j$ for all $j \in \{1, 2, 3, 4\}\backslash\{i\}$ for some i. Notice that x_1 is included 3 times in our expression for $G\mathbf{v}$, in the entries of x_1, $x_1 + x_2 + x_4$, and $x_1 + x_3 + x_4$. An equivalent expression holds for entries of $G\mathbf{w}$; thus, if $i = 1$, then $d_H(G\mathbf{v}, G\mathbf{w}) = 3$. Applying the same logic on other values of i, we find that the Hamming distance of $G\mathbf{v}$ and $G\mathbf{w}$ is 3 if $i \in \{1, 2, 3\}$ and is 4 otherwise, so our claim holds here.

The only other case is when $k = 2$, so that $x_j \neq y_j$ for only two values in the set $\{1, 2, 3, 4\}$. For instance, if these two values are 1 and 2, then $G\mathbf{v}$ and $G\mathbf{w}$ will differ on the entries corresponding to x_1, x_2, $x_1 + x_3 + x_4$, and $x_2 + x_3 + x_4$ only, so that $d_H(G\mathbf{v}, G\mathbf{w}) = 4$. Applying this to other values of j, we see that the Hamming distance $d_H(G\mathbf{v}, G\mathbf{w})$ equals 4 if $x_4 = y_4$, and equals 3 otherwise.

Hence, it follows that $d_H(G\mathbf{v}, G\mathbf{w}) \geq 3$ for all values of k, so $\delta \geq 3$. Since we have explicitly constructed various examples of cases where the Hamming distance of two codewords is 3, it follows that $\delta = 3$, as desired. ■

With our previous results, it follows that the $(7, 4)$-Hamming code can correct $\frac{\delta-1}{2} = 1$ error in codes, giving us a separate way to prove our desired results, without relying on the parity check matrix H at all. An immediate result of the above theorem is the following corollary.

Corollary 5.2. *In order to send a binary string of length n across a faulty system that corrupts at most one character per four characters, it is possible to retrieve the correct version of the original string with at most*

$$7\left\lceil\frac{n}{4}\right\rceil$$

characters being sent in total.

Proof. We simply exploit the $(7, 4)$-Hamming code: add $4\lceil\frac{n}{4}\rceil - n$ zeroes at the end of our string, and split this new string into $\lceil\frac{n}{4}\rceil$ groups of four characters each. Then apply the $(7, 4)$-Hamming code, which is designed exactly to detect and correct single-character errors in strings of length 4. Because this code replaces each four-character block with seven characters, we thus transmit $7\left\lceil\frac{n}{4}\right\rceil$ characters in total. ■

6. Conclusions

The $(7, 4)$-Hamming code is widely used due to its efficient and simple nature. A similar such code is the $(15, 11)$-Hamming code, whose corresponding H has a similar structure. In fact, there exist a whole family of codes, known as $(2^n - 1, 2^n - n - 1)$-Hamming codes; the efficient of this approaches the limit

$$\lim_{n\to\infty} \frac{2^n - n - 1}{2^n - 1} = 1,$$

which is even better than $\frac{4}{7} = 0.573$ or $\frac{11}{15} = 0.733$. However, using significantly large n increases the risk of having more than one error in a single string, and that would be disastrous to the whole Hamming code system. But without the $(7, 4)$-Hamming code, our technological life would be much more glitchy and slower than what we have now; this demonstrates the power of even a simple Hamming code, when $n = 3$.

Acknowledgements

The author would like to thank Simon Rubinstein-Salzedo for giving tons of opportunities, guidance, and insights, Sophia Sanchez for being an awesome and extremely helpful TA.

References

[Fen15] Aaron Fenyes. Matrix algebra and error-correcting codes, 2015.
[Goe15] Michel Goemans. Linear error-correcting codes, 2015.
[Jau12] Jeff Jauregui. Error correcting codes with linear algebra, 2012.

31415 Riemann Avenue, Gaussian Mountains, CA 92653
(You noticed that this was fake, didn't you?)
Email address: eddylihere@gmail.com