Physica D 453 (2023) 133824

Contents lists available at ScienceDirect

Physica D

journal homepage: www.elsevier.com/locate/physd

Fourier transform bounded Kolmogorov complexity

Mohammed Terry-Jack *, Simon O'Keefe

University of York, Heslington, York, YO10 5DD, UK

ARTICLE INFO

Article history:
Received 23 May 2023
Received in revised form 4 June 2023
Accepted 8 June 2023
Available online 22 June 2023
Communicated by B. Hamzi

Keywords:
Discrete Fourier transform
Lossless compression
Quantisation
Binarisation
Kolmogorov complexity
Algorithmic information theory

ABSTRACT

The Discrete Fourier Transform (DFT) has been extended to lossless compression for binary images. Binarisation is key for DFT to compress losslessly because there exist lossy reconstructions (within a specific range of loss values) which are error-corrected during the binarisation step, effectively making the image lossless. In an ironic twist, the quantisation effect which usually introduces errors, has been utilised to remove noise from lossy reconstructions.

1. Introduction

Compression is useful for reducing the space and bandwidth requirements of image storage and transmission [1] and allows more efficient image processing by intelligent machines. The Fourier Transform [2] is a popular image compression technique able to discover hidden patterns via a convolutional property which often transforms seemingly arbitrary data into a more efficient description. It is also one of the most fundamental operations in digital signal processing [3] and has a wide range of applications from image reconstruction and noise reduction to quantum mechanics, etc.

Unfortunately, it does not compress losslessly which prevents it from being applied to certain use cases. For instance, lossless image compression is a legal requirement for medical images used in diagnosis [4,5] since loss of information in medical images can lead to a misjudgment of the disease [1]. Furthermore, a lossless Fourier-based compression opens up the potential for a Fourier Transform bounded (FT-bounded) Kolmogorov Complexity, which is immensely useful for a plethora of other theoretical and practical applications [6].

A recent advancement successfully extended the Discrete Fourier Transform (DFT) to lossless image compression [7] by innovatively utilising a quantisation effect (which usually introduces errors) for denoising (error-correct loss from noisy reconstructions). We review this error-correcting quantisation effect in more detail and present proofs that expand on its underlying principles.

The paper is organised as follows: Section 2 introduces concepts related to image compression while Sections 3 and 4 provide a detailed breakdown of how the DFT was extended for lossless compression. Section 5 briefly introduces Algorithmic Information Theory (AIT) and presents FT-bounded Kolmogorov Complexity. Section 6 summarises the findings, Section 7 presents the various strengths, limitations and future directions of this research and Section 8 discusses parallels and differences with related research.

2. Image compression

Compression is used to reduce the space and bandwidth required for image storage and transmission [1] and can lead to efficient representations useful for intelligent machines.

Definition 1 (*Binary Images*)**.** Let $\mathbb{B} = \{0, 1\}$ and let $S = \mathbb{B}^{W\times T}$ denote the finite set of all 2D binary images of width W and height T (The size of $|S| = |\mathbb{B}|^{WT} = 2^{WT}$).

Definition 2 (*Compression Encoder*)**.** Let $Z = \mathbb{C}^{U\times V}$ denote some representation space in which images are compressed, where $U, V \in \mathbb{Z}$. A compression encoder $C : S \to Z$ is responsible for transforming an arbitrary image $s \in S$ into a compressed representation $z \in Z$, such that:

$$z = C(s) \tag{1}$$

* Corresponding author.
E-mail addresses: mdtj500@york.ac.uk (M. Terry-Jack), simon.okeefe@york.ac.uk (S. O'Keefe).

https://doi.org/10.1016/j.physd.2023.133824

Definition 3 (*Compression Decoder*)**.** The corresponding decoder $C^{-1} : Z \rightarrow S$ is responsible for retrieving the image $s \in S$ from a compressed representation $z \in Z$, s.t.:

$$s = C^{-1}(z) \tag{2}$$

Definition 4 (*Lossless Compression*)**.** A compression is considered lossless if the decoder can restore the original image exactly from the compressed representation without incurring any loss, s.t. $\forall s \in S$:

$$C^{-1}(C(s)) = s \tag{3}$$

Remark 1 (*Examples of Lossless Compression*)**.** Lossless image compression is useful for the fast transmission of large archived images and legally required for medical images used for diagnostic purposes [4,5] since loss of crucial information in medical images can lead to a misjudgment of the disease by doctors [1] and medical technologies. There are many lossless image compression methods [8], such as: Entropy coding, Huffman coding, Bit-plane coding, Run-length coding, Lempel Ziv Welch coding, Gzip, Bzip2 (a block-sorting compressor based on the Burrows–Wheeler transform), PPMZ (prediction by partial matching), etc.

Definition 5 (*Lossy Compression*)**.** A lossy compression returns an image $C^{-1}(C(s))$ which approximates the original image s with some level of error (loss) $\varepsilon \in \Re^{W \times T}$, s.t.:

$$C^{-1}(C(s)) = s + \varepsilon \tag{4}$$

Remark 2 (*Examples of Lossy Compression*)**.** Lossy compressions can be useful in applications whereby a certain amount of error is an acceptable trade-off for increased compression performance. More than half of all files transmitted over the Internet consist of lossy compressed objects. Methods for lossy compression include: Fractal compression, Transform coding, Fourier-related Transform, Wavelet Transform and the classic JPEG algorithm (which is still the most popular algorithm for lossy image compression [4]).

Definition 6 (*Compression Ratio*)**.** The Compression Ratio CR is defined as:

$$CR = \frac{|s|}{|C(s)|} \tag{5}$$

Remark 3 (*Lossy vs Lossless Compression Ratios*)**.** Compression quality is typically quantified by CR (where a higher CR it more desirable). CRs achieved by lossless image compression algorithms are typically not as good as those achieved by lossy algorithms [4]

3. Fourier transform based compression

The Fourier Transform is a very popular lossy image compression technique for its ability to decompose seemingly arbitrary data into efficient descriptions by discovering hidden patterns. Due to the efficiency of its convolutional property, the Discrete Fourier Transform (DFT) [2] is one of the most fundamental operations in digital signal processing [3] with a wide range of applications from image reconstruction and noise reduction to quantum mechanics, etc. Recently the DFT has also been extended to lossless image compression via quantisation [7]. Below we provide new proofs which expand on the definitions given in [7] to offer a precise explanation on how the DFT was extended to lossless compression by innovatively utilising quantisation to error-correct some lossy images.

Definition 7 (*Discrete Fourier Transform*)**.** Let $\mathcal{F} : \Re^{W \times T} \rightarrow \mathbb{C}^{U \times V}$ denote the 2D Discrete Fourier Transformation responsible for transforming an image s from the spatial domain into the frequency domain (by decomposing it into complex-valued coefficients with real and imaginary components), s.t. $\forall u \in [0, U \in \mathbb{Z})$, $\forall v \in [0, V \in \mathbb{Z})$:

$$\mathcal{F}(s)_{u,v} = \sum_{w=0}^{W-1} \sum_{t=0}^{T-1} s_{w,t} \cdot e^{-i2\pi(\frac{uw}{W} + \frac{vt}{T})} \tag{6}$$

Definition 8 (*Inverse Discrete Fourier Transform*)**.** Let $\mathcal{F}^{-1} : \mathbb{C}^{U \times V} \rightarrow \Re^{W \times T}$ denote the Inverse of the Discrete Fourier Transform, such that:

$$\mathcal{F}^{-1}(\mathcal{F}(s)) = s \tag{7}$$

and invertibility is guaranteed by orthogonality (the inverse of the DFT is just the conjugate transpose [3]).

Definition 9 (*Compression Size*)**.** The compression size of the Fourier representation $|\mathcal{F}(s)|$ is the number of nonzero coefficients (i.e. sparser representations will have a smaller compression size), s.t:

$$|\mathcal{F}(s)| = |\sum_{u=0}^{U-1} \sum_{v=0}^{V-1} (\mathcal{F}(s)_{u,v} \neq 0 + 0i)| \tag{8}$$

Remark 4 (*Compression Size in Bits*)**.** The data storage requirement is reduced if only the coefficients with a significant amount of energy are stored. Therefore, to store the transformed data in a more efficient manner, Fourier coefficients which equate to zero get ignored. The remaining non-zero coefficients will each be represented by an equal number of bits internally (irregardless of their varying values and differing significant figures) due to fixed-point arithmetic. The compression length in bits can therefore be obtained by multiplying the compression length by a constant b which represents the number of bits to encode a single complex value. The number of bits for a single floating point value is usually 32 bits (full-precision), however, 16 bits (half-precision) or 64 bits (double-precision) can also be used. For a single complex value (with a real and an imaginary component), twice as many bits are required.

Lemma 1. *The compression size will always be within the finite limits determined by the representation space* $Z = \mathbb{C}^{U \times V}$*:*

$$0 \leq |C(s)| \leq UV \tag{9}$$

Proof. Let $C^{min}(s)$ be the smallest compression size, such that $|C^{min}(s)| \leq |C(s)|$, and it is a compression where all coefficients are zero. Therefore, there will be no coefficients maintained in the representation according to Definition 9, s.t. $|C^{min}(s)| = |\emptyset| = 0$. Similarly, let $C^{max}(s)$ denote the largest compression size, s.t. $|C(s)| \leq |C^{max}(s)|$, which is a compression where all coefficients are nonzero. Therefore, all coefficients will be maintained in the representation according to Definition 9, s.t. $|C^{max}(s)| = UV$. Therefore: $|C^{min}(s)| \leq |C(s)| \leq |C^{max}(s)| \Rightarrow 0 \leq |C(s)| \leq UV$

Definition 10 (*Fourier-based Encoder*)**.** The encoding stage applies a binary mask $m^* \in \mathbb{B}^{U \times V}$ to optimally filter the Fourier coefficients into a minimal representation (i.e. by setting as many coefficients as possible to zero):

$$C^*(s) = m^* \mathcal{F}(s) \tag{10}$$

Remark 5 (*Fourier Filtering Causes Lossy Compression*)**.** $\mathcal{F}$ is an invertible function (Definition 8) and so, on its own, it does not

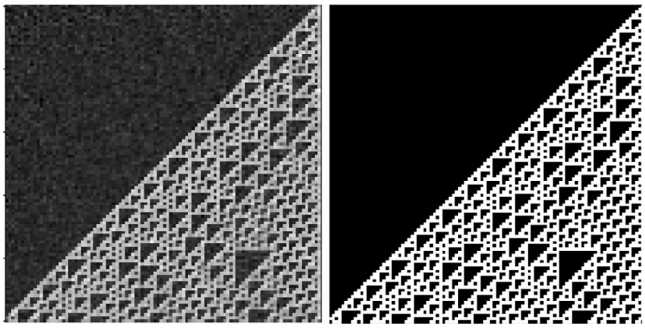

Fig. 1. An example of a lossy image with some noise (left) which gets error-corrected by the binarisation step to ultimately become lossless (right).

lead to any loss of information. However, during compression (i.e. when the coefficients get filtered by m^*), a loss of information does result. This loss is often negligible as most information gets encoded in only a small minority of coefficients, however, this is ultimately why traditional Fourier Transform based compression is lossy. In fact, the greater the compression the greater the resulting loss (highlighting the trade-off between compression size and loss)

Definition 11 (*Fourier-Based Decoder*)**.** The decoding stage includes the indicator function $\mathbb{1}_{>\theta}$ to binarise the reconstructed image returned by the inverse Fourier transformation $\mathcal{F}^{-1}$, s.t.

$$C^{-1}(z) = \mathbb{1}_{>\theta}(\mathcal{F}^{-1}(z)) \tag{11}$$

Definition 12 (*Binarisation Step*)**.** The indicator function $\mathbb{1}_{>\theta} : \Re^{W\times T} \rightarrow S$ is used to map real-values into binary-values using some real-valued threshold $\theta \in \Re$, s.t $\forall \hat{s} \in \Re^{W\times T}$:

$$\mathbb{1}_{>\theta}(\hat{s})_{w,t} = \begin{cases} 1 & \hat{s}_{w,t} > \theta \\ 0 & otherwise \end{cases} \tag{12}$$

Remark 6 (*Binarisation Step Error-Corrects Lossy Images*)**.** As the binarisation step $\mathbb{1}_{>\theta}$ maps real-values into binary-values, a quantisation effect occurs. However, rather than introducing further loss (as would be expected), this quantisation effect removes the noise from certain lossy compressions that have resulted from the filtering step (Remark 5). Therefore, there exists lossy reconstructions (within a specific range of loss values) which will be error-corrected during the binarisation step $\mathbb{1}_{>\theta}$ which effectively makes the reconstruction lossless (Fig. 1).

Lemma 2 (*Error-Correction*)**.** *There exist lossy images that will be error-corrected by the binarisation step $\mathbb{1}_{>\theta}$, s.t.*

$$\mathbb{1}_{>\theta}(s+\varepsilon) = s \tag{13}$$

where $\varepsilon \in \Re^{W\times T}$ denotes the loss in addition to image $s \in S$

Proof. Let $\theta = 1, s = \mathbf{1}^{W\times T}, \varepsilon = \mathbf{0.5}^{W\times T}$ and assume, by way of contradiction, $\mathbb{1}_{>\theta}(s+\varepsilon) \neq s$

$\Rightarrow \mathbb{1}_{>\theta}(\mathbf{1}^{W\times T} + \varepsilon) \neq \mathbf{1}^{W\times T}$

$\mathbb{1}_{>\theta}(\mathbf{1}^{W\times T} + \mathbf{0.5}^{W\times T}) \neq \mathbf{1}^{W\times T}$

$\mathbb{1}_{>\theta}(\mathbf{1.5}^{W\times T}) \neq \mathbf{1}^{W\times T}$

But by Definition 12 $\quad 1.5 > \theta : 1.5 \Rightarrow 1$

$\Rightarrow \mathbb{1}_{>\theta}(\mathbf{1.5}^{W\times T}) = \mathbf{1}^{W\times T}$

Contradiction

Lemma 3. *There exist lossy images which do not get error-corrected by the binarisation step $\mathbb{1}_{>\theta}$, s.t.*

$$\mathbb{1}_{>\theta}(s+\varepsilon) \neq s \tag{14}$$

Proof. Let $\theta = 1, s = \mathbf{1}^{W\times T}, \varepsilon = -\mathbf{1.5}^{W\times T}$ and assume, by way of contradiction, $\mathbb{1}_{>\theta}(s+\varepsilon) = s$

$\Rightarrow \mathbb{1}_{>\theta}(\mathbf{1}^{W\times T} + \varepsilon) = \mathbf{1}^{W\times T}$

$\mathbb{1}_{>\theta}(\mathbf{1}^{W\times T} + -\mathbf{1.5}^{W\times T}) = \mathbf{1}^{W\times T}$

$\mathbb{1}_{>\theta}(-\mathbf{0.5}^{W\times T}) = \mathbf{1}^{W\times T}$

But by Definition 12 $\quad -0.5 \leq \theta : -0.5 \Rightarrow 0$

$\Rightarrow \mathbb{1}_{>\theta}(-\mathbf{0.5}^{W\times T}) = \mathbf{0}^{W\times T}$

Contradiction

Lemma 4 (*Error-Correcting Limits*)**.** *The range of loss values which can be error-corrected by the binarisation step $\mathbb{1}_{>\theta}$ is bounded by the binarisation threshold θ, s.t:*

$$-1 + \theta < \varepsilon_{w,t} \leq \theta \tag{15}$$

Proof. We can derive the range of loss values which will be corrected by $\mathbb{1}_{>\theta}$ directly from the indication function's Definition 12, s.t. $\forall w \in [0, W), \forall t \in [0, T)$:

$s_{w,t} \in \mathbb{B} : s_{w,t} = 1 \Rightarrow s_{w,t} + \varepsilon_{w,t} > \theta$

$\Rightarrow 1 + \varepsilon_{w,t} > \theta$

$\varepsilon_{w,t} > -1 + \theta$

and:

$s_{w,t} \in \mathbb{B} : s_{w,t} = 0 \Rightarrow s_{w,t} + \varepsilon_{w,t} \leq \theta$

$\Rightarrow 0 + \varepsilon_{w,t} \leq \theta$

$\varepsilon_{w,t} \leq \theta$

Therefore:

$-1 + \theta < \varepsilon_{w,t} \leq \theta$

Lemma 5 (*Binarisation Threshold Limits*)**.** *The threshold must be bounded between 0 and 1, s.t. $\theta \in (0, 1] \subset \Re$ in order to ensure no additional loss is introduced when there is no loss to begin wih $\varepsilon = \mathbf{0}^{W\times T}$ (i.e. to ensure the quantisation error does not end up making a lossless image lossy).*

Proof. Let $\varepsilon_{w,t} = 0$. Now to keep $\varepsilon_{w,t} = 0$, according to Lemma 4, the threshold θ is bounded such that:

$-1 + \theta < 0 \Rightarrow \theta < 0 + 1 \Rightarrow \theta < 1$ $\quad$ is the upper bound

$\theta \geq 0 \Rightarrow \theta = 0$ $\quad$ is the lower bound

$\Rightarrow \theta \in [0, 1) \subset \Re$

Lemma 6 (*Binarising a Binarised Image*)**.** *When the binarisation threshold is bounded between 0 and 1, s.t. $\theta \in [0, 1)$, the binarisation step $\mathbb{1}_{>\theta}$ does not modify an image that is already binarised, such that $\forall s \in S$:*

$$\mathbb{1}_{>\theta}(s) = s \tag{16}$$

Proof. Let $\theta \in [0, 1) \subset \Re$ be some threshold and let $s \in S$ be a binary image. Now let us assume, by way of contradiction, that the indicator function introduces some non-zero loss ε to the image, s.t.

$\mathbb{1}_{>\theta}(s) = s + \varepsilon$

$\mathbb{1}_{>\theta}(s + \mathbf{0}^{W\times T}) = s + \varepsilon$

Now according to Lemma 4 $\forall w \in [0, W), \forall t \in [0, T)$:

$-1 + \theta < \mathbf{0}^{W \times T}_{w,t} \leq \theta$

$-1 + \theta < 0 \leq \theta$

$\Rightarrow \varepsilon = \mathbf{0}^{W \times T}$ by Lemma 5

Therefore, we find the resulting loss is zero:

$\Rightarrow \mathbb{1}_{>\theta}(s + \mathbf{0}^{W \times T}) = s + \mathbf{0}^{W \times T}$

ergo, no loss is introduced

$\mathbb{1}_{>\theta}(s) = s$

Contradiction

4. Lossless fourier filters

Lemma 7 (*Loss of a Filter*)**.** *The loss ε in terms of a given filter m is given by:*

$$\varepsilon = C^{-1}(m\mathcal{F}(s)) - s \tag{17}$$

Proof.

$C^{-1}(C(s)) = s + \varepsilon$ by Definition 5

$\varepsilon = C^{-1}(C(s)) - s$

$\varepsilon = C^{-1}(m\mathcal{F}(s)) - s$ by Definition 10

Definition 13 (*Lossless Compression Filters*)**.** Let $\mathcal{M}^* \subset \mathbb{B}^{U \times V}$ denote the set of lossless filters which modify the Fourier representation $\mathcal{F}(s)$ without preventing it from reconstructing any original image exactly (i.e. without loss), such that $\forall s \in S$:

$$\mathcal{M}^* = \{m \mid C^{-1}(m\mathcal{F}(s)) = s \quad \forall m \in \mathbb{B}^{U \times V}\} \tag{18}$$

Lemma 8. *All lossless filters necessarily have a loss of zero, s.t.* $\forall m \in \mathcal{M}^* : \varepsilon = \mathbf{0}^{W \times T}$

Proof. Let $m \in \mathcal{M}^*$ be a lossless filter. Now assume, by way of contradiction, it has a non-zero loss $\varepsilon \neq \mathbf{0}^{W \times T}$

$\varepsilon = C^{-1}(m\mathcal{F}(s)) - s$ by Lemma 7

$\varepsilon = s - s$ by Definition 13

$\varepsilon = \mathbf{0}^{W \times T}$

Contradiction

Definition 14 (*The Filter Yielding the Largest Compression Size*)**.** Let m^1 be a filter consisting entirely of ones, s.t.

$$m^1 = \mathbf{1}^{U \times V} \tag{19}$$

Lemma 9. *Filter m^1 gives the largest compression size, s.t.*

$$|m^1\mathcal{F}(s)| = max\{|m\mathcal{F}(s)| \mid \forall m \in \mathbb{B}^{U \times V}\} \tag{20}$$

Proof.

$|m^1\mathcal{F}(s)| = |\mathbf{1}^{U \times V}\mathcal{F}(s)|$ by Definition 14

$\Rightarrow |\mathcal{F}(s)| = UV$

(which is the largest compression size according to Lemma 1)

Lemma 10. *The loss of filter m^1 is $\varepsilon = \mathbf{0}^{W \times T}$*

Proof.

$\varepsilon = C^{-1}(m^1\mathcal{F}(s)) - s$ by Lemma 7

$\varepsilon = C^{-1}(\mathbf{1}^{W \times T}\mathcal{F}(s)) - s$ by Definition 14

$\varepsilon = C^{-1}(\mathcal{F}(s)) - s$

$\varepsilon = \mathbb{1}_{>\theta}(\mathcal{F}^{-1}(\mathcal{F}(s))) - s$ by Definition 11

$\varepsilon = \mathbb{1}_{>\theta}(s) - s$ by Definition 8

$\varepsilon = s - s$ by Lemma 6

$\varepsilon = \mathbf{0}^{W \times T}$

Lemma 11. *Filter $m^1 \in \mathcal{M}^*$ is a lossless filter*

Proof.

$C^{-1}(C(s)) = s + \varepsilon$ by Definition 5

$C^{-1}(m^1\mathcal{F}(s)) = s + \varepsilon$ by Definition 10

$C^{-1}(m^1\mathcal{F}(s)) = s + \mathbf{0}^{W \times T}$ by Lemma 10

$C^{-1}(m^1\mathcal{F}(s)) = s \Rightarrow m^1 \in \mathcal{M}^*$ by Definition 13

Lemma 12. *Filter m^1 gives the largest compression size of all lossless filters, s.t.*

$$|m^1\mathcal{F}(s)| = max\{|m\mathcal{F}(s)| \mid \forall m \in \mathcal{M}^*\} \tag{21}$$

Proof. $m^1 \in \mathcal{M}^*$ is a lossless filter (Lemma 11) and the lossless filters are a subset of all binary filters $\mathcal{M}^* \subset \mathbb{B}^{U \times V}$ (Definition 13). Now since m^1 is the largest of all binary filters (Lemma 9), it has to be the largest of the lossless filters too $\Rightarrow |m^1\mathcal{F}(s)| = max\{|m\mathcal{F}(s)| \mid \forall m \in \mathcal{M}^* \subset \mathbb{B}^{U \times V}\}$

Remark 7 (*Binarisation is Key for DFT to Compress Losslessly*)**.** Filter $m^1 \in \mathcal{M}^*$ is the only truly lossless filter (i.e. lossless in the sense that it does not need error-correction via binarisation). However, m^1 is also trivially lossless (i.e. it has no loss by avoiding compression of the image, Lemma 9). This highlights the significance of the error-correcting binarisation step $\mathbb{1}_{>\theta}$ (Lemma 2), without which the Discrete Fourier Transform (DFT) would have no lossless filters that perform actual compression on the image (and thus not viable for lossless image compression at all).

Definition 15 (*The Filter Yielding the Smallest Compression Size*)**.** Let m^0 be a filter consisting entirely of zeros, s.t.

$$m^0 = \mathbf{0}^{U \times V} \tag{22}$$

Lemma 13. *Filter m^0 gives the smallest compression size, s.t.*

$$|m^0\mathcal{F}(s)| = min\{|m\mathcal{F}(s)| \mid \forall m \in \mathbb{B}^{U \times V}\} \tag{23}$$

Proof.

$|m^0\mathcal{F}(s)| = |\mathbf{0}^{U \times V}\mathcal{F}(s)|$ by Definition 15

$\Rightarrow |\mathbf{0}^{U \times V}| = 0$

(which is the minimum compression size according to Lemma 1)

Lemma 14. *The loss of filter m^0 is $\varepsilon = -s$*

Proof.

$\varepsilon = C^{-1}(m^0\mathcal{F}(s)) - s$ by Lemma 7

$\varepsilon = C^{-1}(\mathbf{0}^{U \times V}\mathcal{F}(s)) - s$ by Definition 15

$\varepsilon = C^{-1}(\mathbf{0}^{U \times V}) - s$

$\varepsilon = \mathbf{0}^{W \times T} - s$

$\varepsilon = -s$

Lemma 15. *Filter m^0 is not lossless, $m^0 \notin \mathcal{M}^*$*

Proof. Assume, by way of contradiction, filter m^0 is lossless, s.t. $m^0 \in \mathcal{M}^*$. It follows that it must have no loss $\varepsilon = \mathbf{0}^{W \times T}$ by

Lemma 8. But we know by Lemma 14 that m^0 does produce a loss. Contradiction.

Definition 16 (*The Optimal Filter*)**.** The optimal filter $m^* \in \mathcal{M}^*$ is the lossless filter resulting in the smallest Fourier representation, such that:

$$|m^*\mathcal{F}(s)| = min\{|m\mathcal{F}(s)| \mid \forall m \in \mathcal{M}^*\} \tag{24}$$

Lemma 16. *The smallest filter $m^0 \in \mathbb{B}^{U\times V}$ is not the optimal filter $m^* \in \mathcal{M}^*$*

$$m^* \neq m^0 \tag{25}$$

Proof. Let us assume, by way of contradiction, m^0 is the optimal filter, s.t. $m^* = m^0$. Therefore, $m^0 \in \mathcal{M}^*$ is lossless by Definition 16. But $m^0 \notin \mathcal{M}^*$ is lossy by Lemma 15. Contradiction

5. FT-bounded Kolmogorov complexity

Definition 17 (*Kolmogorov Complexity*)**.** The Kolmogorov Complexity K is defined as the length l of the shortest program p that can generate a given string s and halt when executed on a universal computer U, s.t.

$$K(s) = min\{l(p) \mid U(p) = s\} \tag{26}$$

and $K(s) = \infty$ if there are no such p [6]

Remark 8 (*Algorithmic Information Theory*)**.** In the 1950s, Kolmogorov generalised Shannon's information theoretic Entropy metric (which measures the average information communicated through objects produced by a random source) to be able to apply it to the study of deterministic dynamical systems [9–11]. The result was a precise and objective way to quantify of the amount of information in an individual object [12], commonly referred to as Kolmogorov Complexity [6]. It has since grown into an entirely new branch of computer science and mathematics, known as Algorithmic Information Theory (AIT) [6,13].

Lemma 17 (*Uncomputability*)**.** *Kolmogorov complexity K is a powerful and elegant definition of complexity. However K is not a computable function in general (it is only possible to bound K [11] and approximate it [14])*

Proof. The proof for the uncomputability of Kolmogorov Complexity is well-known but we provide a proof here for completeness. Let us assume, by way of contradiction, that $K(s^n) = n$ is a computable function with a program size at least as large as the complexity value $n \in \mathbb{Z}$ it returns for a given image $s^n \in S$, s.t. $|K| \geq |n|$, where the size of n is the number of digits used to represent it, s.t. $|n| \geq \log(n)$, giving us a lower-bound on the size of K, $\Rightarrow |K| \geq log(n)$.

Now let K^{-1} be a program that recursively searches for an image with a specified complexity, s.t.

$$K^{-1}(n,i) = \begin{cases} \mathcal{S}_i & K(\mathcal{S}_i) = n \\ K^{-1}(n, i+1) & otherwise \end{cases}$$

where $\mathcal{S}$ is the infinite set of lexicographical ordered binary strings. So K^{-1} returns s^n when given n, s.t. $K^{-1}(n,0) = s^n$ as s^n is determined to have a complexity of n by K^{-1} (by way of K, s.t. $K(s^n) = n$). It necessarily follows that n is the smallest size for any program able to produce s^n, by Definition 17. Since K^{-1} is one such program able to produce s^n, we can lower-bound the size of K^{-1} to n, s.t. $|K^{-1}| \geq n$.

Further note that K is contained within K^{-1} and so the size of K^{-1} must at least be as large as K, s.t. $|K^{-1}| \geq |K|$. Therefore, $|K^{-1}| \geq \log(n)$ or $\Rightarrow |K^{-1}| = \log(n) + c$ where c is a constant denoting the fixed size overhead required by K^{-1} in addition to using K. Substituting $|K^{-1}| = \log(n) + c$ into $|K^{-1}| \geq n$ gives $\Rightarrow log(n) + c \geq n$ which is clearly a contradiction for any n exceeding c (or more precisely, we have a contradiction for any $c < n - \log(n)$).

Remark 9 (*Complexity vs Compression*)**.** In the algorithmic sense, higher complexity equates to less compressibility and vice-versa. Sequences can sometimes be quite long and even seem arbitrarily random, however, they may be hiding a simpler underlying pattern (an intrinsic order) which, if discovered (i.e. by the Fourier Transform), can be utilised to describe (or encode) the original sequence more efficiently, making for higher compressibility. Since regularities can be used to compress descriptions, the absence of regularities intuitively implies incompressibility [6].

Lemma 18 (*FT-bounded Kolmogorov Complexity*)**.** *The compression size of the optimal lossless compression C^* is equivalent to the Fourier Transform-bounded (FT-bounded) Kolmogorov Complexity $K_{\mathcal{F}}$, such that:*

$$K_{\mathcal{F}}(s) = |C^*(s)| \tag{27}$$

Proof. Let $s \in S$ be an arbitrary binary image (if so desired, this image can be read in row-by-row as a binary string). Let the program p be the filtered Fourier representation of s, s.t. $p = m\mathcal{F}(s)$ (Note that the program's length l becomes equivalent to the compression size, s.t. $l(p) = |m\mathcal{F}(s)|$). Finally, let the computer which executes p be the Fourier-based decoder, s.t. $U = C^{-1}$. Therefore:

$K(s) = min\{|m\mathcal{F}(s)| \mid C^{-1}(m\mathcal{F}(s)) = s\}$ by Definition 17
$\Rightarrow K_{\mathcal{F}}(s) = |m^*\mathcal{F}(s)|$ by Definition 16
$K_{\mathcal{F}}(s) = |C^*(s)|$ by Definition 10

Remark 10 (*Kolmogorov Complexity vs FT-bounded Kolmogorov Complexity*)**.** There are some significant differences between the pure definition Kolmogorov Complexity K (Definition 17) and FT-bounded Kolmogorov Complexity $K_{\mathcal{F}}$ (Lemma 18), which include:

- K takes in an infinite binary string, whereas $K_{\mathcal{F}}$ takes in a binary image from the set S with a finite size (Definition 1)
- K uses a Universal Computer U which has the ability to simulate one another and effectively execute any program [12]), whereas $K_{\mathcal{F}}$ replaces U with the Fourier-based Decoder (which is essentially the Inverse Fourier Transform) and so not a Turing Machine as it is constrained to running Fourier-based programs. As such, $K_{\mathcal{F}}$ can run any (Fourier-based) program it is given without risk of encountering the *halting problem*.

Lemma 19 (*Computability*)**.** *The additional constraints imposed on FT-bounded Kolmogorov Complexity $K_{\mathcal{F}}$ allow for it to be computable within the range $K_{\mathcal{F}} \in [0, UV)$ for binary images in S.*

Proof. Let $K_{\mathcal{F}} : S \times \mathbb{Z} \to \mathbb{Z}$ be the FT-bounded Kolmogorov Complexity computed by a greedy search algorithm, s.t. $\forall s \in S$:

$$K_{\mathcal{F}}(s,n) = \begin{cases} \infty & n \notin [0, UV) \\ n & \mathcal{M}^* \cup \mathcal{M}^n \neq \emptyset \\ K_{\mathcal{F}}(s, n+1) & otherwise \end{cases}$$

where $\mathcal{M}^*$ denotes the finite set of lossless filters and $\mathcal{M}^n$ denotes the finite set of filters with a resulting compression size of n, s.t. $\mathcal{M}^n = \{m \mid |m\mathcal{F}(s)| = n \quad \forall m \in \mathbb{B}^{U\times V}\}$. Therefore,

$K_{\mathcal{F}}(s, 0)$ will initially assume a compression size of 0 and then greedily enumerating all filters of that size and then doing the same for all filters with an associated compression size of 1, and then 2 and so on up until UV. The first lossless filter encountered in this way would have the smallest compression size and the algorithm would have found the FT-bounded Kolmogorov Complexity (Lemma 18). Since both the lossless filters $\mathcal{M}^*$ and all filters of a specified compression size $\mathcal{M}^n$ are ultimately subsets of the binary filters $\mathbb{B}^{U\times V}$, s.t. $\mathcal{M}^*, \mathcal{M}^n \subset \mathbb{B}^{U\times V}$, then all subsets used in $K_{\mathcal{F}}$ have a finite size (since $\mathbb{B}^{U\times V}$ has a finite size, s.t. $|\mathbb{B}^{U\times V}| = |\mathbb{B}|^{UV} = 2^{UV}$) and thus are enumerable and guaranteed to halt.

Definition 18 (*Upper Semi-Computable*)**.** A function f is upper semi-computable if f can be computably approximated from above [12]. More precisely, if there exists another function g, such that for some $x \in \mathbb{Q}$ and $\forall k \in \mathbb{Z}^+$:

$$g(x, k+1) \leq g(x, k) \qquad \text{and} \qquad \lim_{k\to\infty} g(x, k) = f(x) \tag{28}$$

Definition 19 (*Lower Semi-Computable*)**.** A function f is lower semi-computable if $-f$ is upper semi-computable

Definition 20 (*Computable*)**.** A function is computable if it is both upper semi-computable and lower semi-computable

Definition 21 (*Decidable*)**.** Let $\mathcal{A}$ be a countable set. $\mathcal{A}$ is decidable (i.e. recursive) if for any candidate a, there exists a Turing machine that can decide whether $a \in \mathcal{A}$ and halts.

Definition 22 (*Enumerable*)**.** Let $\mathcal{A}$ be a countable set. $\mathcal{A}$ is (recursively) enumerable if there exists a Turing machine that outputs all and only the elements of $\mathcal{A}$ without halting (e.g. the prime numbers are enumerable [12])

6. Summary

The extension of the DFT to lossless image compression is a recent advancement in [7]. We contribute theoretical explanations and proofs for a better understanding of the underlying principles related to lossless compression using DFT and the innovative utilisation of quantisation to perform error-correction. To summarise, the Fourier-based compression encoder uses a binary mask to filter the Fourier coefficients for a minimal representation (however, this produces a lossy compression as information is discarded by the filter — highlighting the trade-off between compression size and loss). Fortunately, the Fourier-based decoder further binarises the reconstructed image and this binarisation step introduces a quantisation effect which allows for certain lossy compressions to be error corrected. While it was shown that not all lossy images can be error-corrected by the binarisation step (and that it is ultimately bounded by the binarisation threshold), many lossy images can be perfectly recovered in this way. The corrected loss effectively bypasses the compression-loss trade-off and, for the first time, permits the DFT to perform non-trivial, lossless compression.

7. Future directions

Using the DFT for lossless compression was shown (on three different tests) to be superior to many other lossless compressors (including Huffman encoding, Run-length encoding, Lempel–Ziv–Markov-Chain Algorithm, ZLIB, GZIP, BZIP2, etc.) [7].

However, once the Fourier-based compression encoder obtains a set of coefficients for a given image, it is not trivial to filter it optimally. A search algorithm is required to find the optimal filter mask m^* which has the potential to make the compression computationally demanding (if the search is inefficient) or impair compression (if the search is suboptimal). While it is theoretically possible to find the optimal mask via a greedy search algorithm, like the one shown in proof 19, it would be inefficient and impractical due to the length of time it could take to enumerate all filters ($\mathcal{O}(UV|\mathbb{B}|^{UV})$). The search could be improved if the lossless filters $\mathcal{M}^*$ are cached and the binary filters $\mathbb{B}^{U\times V}$ are iterated through just once at the beginning to generate all the sets $\mathcal{M}^0, \mathcal{M}^1, \ldots$ before commencing the search ($\mathcal{O}(|\mathbb{B}|^{UV})$). An even better search heuristic was given in the appendix of [7] that improves the efficiency to $\mathcal{O}(UV)$ using the magnitude of the Fourier coefficients to guide the search. However, this search strategy would not guarantee the optimal filter is found.

The Sparse Fast Fourier Transform [15] is a recent algorithm which can compute the DFT with a sparse frequency domain in sub-linear runtime. This algorithm finds the minimal number of coefficients to represent a given image and could potentially provide a more efficient search for the optimal filter.

Alternatively, if the goal is to represent the original signal using a relatively small set of coefficients (to reduce the compression size), an alternative transformation may be desireable. The Discrete Cosine Transform (DCT) is a close relative of the Fourier transform and known to compact more energy into a fewer number of coefficients [16] (the DCT representation tends to have more of its energy concentrated in a smaller number of coefficients when compared to other transforms like the DFT). The DCT is frequently used in lossy data compression (such as the widely used JPEG compression [1]) and is very similar to the DFT except that the complex-valued coefficients are replaced with real-valued coefficients, thus further reducing the bits required to represent the compression (Remark 4).

JPEG2000 (a newer but less commonly used version of JPEG) is based on the Discrete Wavelet Transform (DWT) and is the most widely known algorithm for both lossy and lossless compression [4]. While the DWT may not necessarily offer increased compression gains over the DCT or DFT, it may improve the overall quality of the compression since the Fourier Transform is known not to represent abrupt changes efficiently. Consider an image containing a square wave (which can be compactly represented by a "half on, half off" signal). It would require an infinite number of frequencies to represent this square wave perfectly in Fourier space [9]. For any function that is smooth everywhere except at a few singular points, there is no way to localise these singularities in Fourier space [17] because the Fourier transform is inherently nonlocal (unlike wavelets which can be performed locally on a signal [17]).

Finally, it should be noted that while utilising the error-correcting quantisation effect of a binarisation function does successfully extend the DFT to lossless image compression, it also limits it to binary images. Nevertheless, scaling this quantisation effect to nonbinary data should simply be a matter of increasing the number of quantisation buckets as required (we leave this for future work).

8. Discussion

A couple of parallels can be found in lifting-based lossless transforms [16] which use the Wavelet Transform and quantisation to produce lossless compressions. An input signal is transformed and the resulting coefficients are quantised (i.e. lifting steps are applied to round the values to integers). The quantisation step compresses the representation but, in so doing, introduces loss by corrupting the invertability of the transformation such that the inverse no longer guarantees an exact reconstruction. To preserve the invertability property and recover

the original signal perfectly, the same procedure is applied to both the forward and inverse transforms [3] (i.e. a reversible denoising and lifting step is used to remove the loss [18]). Therefore, quantisation is used to compress the representation instead of a filtering mask and a search is used to find an optimal denoising filter (to avoid distortions worsening the transformed component more than the noise reduction [18]) instead of a simple linear binarisation filter.

Another method is described in [12] which uses compression-based denoising to error-correct a lossy image. However, this method also relies on a search algorithm to remove noise (loss) from a noisy image, as opposed to the quantisation effect of a binarising threshold function. More precisely, the method uses a Genetic Algorithm to search for an optimal compression which can separate structure from noise by minimising the distortion of a target image (which is noiseless), thus denoising the (noisy) input image being compressed. Interestingly, the search objective here is almost the exact opposite of the search objective to find the optimal filter mask (which looks for a filter that effectively adds noise to the image – without exceeding the error-correcting range of the binarisation step to ensure the noise can later be removed – to minimise the compression size, as opposed to looking for a compression that minimises the noise).

Finally, an image compression method is described in [19] which uses the DFT in conjunction with quantisation. However, the compression method is not lossless. An image is divided into blocks and DFT is then applied to each block followed by a uniform quantisation to reduce the number of bits needed to represent the coefficients. A Matrix Minimisation algorithm is further applied to the high frequency components to reduce them to $\frac{1}{3}$ of the original size by contracting every three coefficients into a single value. The decompression stage uses a search to recover the original coefficients.

CRediT authorship contribution statement

Mohammed Terry-Jack: Conceptualization, Methodology, Software, Formal analysis, Writing – original draft, Editing. **Simon O'Keefe:** Writing – review.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

No data was used for the research described in the article.

References

[1] G. Xin, P. Fan, A lossless compression method for multi-component medical images based on big data mining, Sci. Rep. 11 (1) (2021) 1–11.
[2] J.W. Cooley, J.W. Tukey, An algorithm for the machine calculation of complex Fourier series, Math. Comp. 19 (90) (1965) 297–301.
[3] S. Oraintara, Y.-J. Chen, T.Q. Nguyen, Integer fast Fourier transform, IEEE Trans. Signal Process. 50 (3) (2002) 607–618.
[4] R. Starosolski, Hybrid adaptive lossless image compression based on discrete wavelet transform, Entropy 22 (7) (2020) 751.
[5] S. Dewitte, J. Cornelis, Lossless integer wavelet transform, IEEE Signal Process. Lett. 4 (6) (1997) 158–160.
[6] L. Ming, P.M. Vitányi, Kolmogorov complexity and its applications, in: Algorithms and Complexity, Elsevier, 1990, pp. 187–254.
[7] M. Terry-Jack, Novel lossless compression method based on the Fourier transform to approximate the Kolmogorov complexity of elementary cellular automata, J. Softw. Eng. Appl. 15 (10) (2022) 359–383.
[8] M.M.H. Chowdhury, A. Khatun, Image compression using discrete wavelet transform, Int. J. Comput. Sci. Issues (IJCSI) 9 (4) (2012) 327.
[9] J.P. Crutchfield, What lies between order and chaos? in: Art and Complexity, Elsevier, 2003, pp. 31–45.
[10] D.P. Feldman, J.P. Crutchfield, Measures of statistical complexity: Why? Phys. Lett. A 238 (4–5) (1998) 244–252.
[11] N. Israeli, N. Goldenfeld, Coarse-graining of cellular automata, emergence, and the predictability of complex systems, Phys. Rev. E 73 (2) (2006) 026203.
[12] P.M. Vitányi, Similarity and denoising, Phil. Trans. R. Soc. A 371 (1984) (2013) 20120091.
[13] P.D. Grünwald, P.M. Vitányi, et al., Algorithmic information theory, Handb. Philos. Inf. (2008) 281–320.
[14] J.-C. Dubacq, B. Durand, E. Formenti, Kolmogorov complexity and cellular automata classification, Theoret. Comput. Sci. 259 (1–2) (2001) 271–285.
[15] J. Schumacher, M. Püschel, High-performance sparse fast Fourier transforms, in: 2014 IEEE Workshop on Signal Processing Systems, SiPS, IEEE, 2014, pp. 1–6.
[16] J. Liang, T.D. Tran, Fast multiplierless approximations of the DCT with the lifting scheme, IEEE Trans. Signal Process. 49 (12) (2001) 3032–3044.
[17] M. Farge, Wavelet transforms and their applications to turbulence, Annu. Rev. Fluid Mech. 24 (1) (1992) 395–458.
[18] R. Starosolski, Reversible denoising and lifting based color component transformation for lossless image compression, Multimedia Tools Appl. 79 (17–18) (2020) 11269–11294.
[19] M.H. Rasheed, O.M. Salih, M.M. Siddeq, M.A. Rodrigues, Image compression based on 2D discrete Fourier transform and matrix minimization algorithm, Array 6 (2020) 100024.