A quick and easy way to estimate entropy and mutual information for neuroscience

Mickael Zbili[1, 2], Sylvain Rama[3*]

[1], Lyon Neuroscience Research Center, INSERM U1028-CNRS UMR5292-Université Claude Bernard Lyon1, Lyon, France

[2], Now in Blue Brain Project, École polytechnique fédérale de Lausanne (EPFL) Biotech Campus, Geneva, Switzerland.

[3], UCL Queen Square Institute of Neurology, University College London, London, United Kingdom.

[*], Correspondence: Sylvain Rama, s.rama@ucl.ac.uk

Abstract

Calculations of entropy of a signal or mutual information between two variables are valuable analytical tools in the field of neuroscience. They can be applied to all types of data, capture nonlinear interactions and are model independent. Yet the limited size and number of recordings one can collect in a series of experiments makes their calculation highly prone to sampling bias. Mathematical methods to overcome this so called "sampling disaster" exist, but require significant expertise, great time and computational costs. As such, there is a need for a simple, unbiased and computationally efficient tool for reliable entropy and mutual information estimation. In this paper, we propose that application of entropy-coding compression algorithms widely used in text and image compression fulfill these requirements. By simply saving the signal in PNG picture format and measuring the size of the file on the hard drive, we can reliably estimate entropy through different conditions. Furthermore, with some simple modifications of the PNG file, we can also estimate mutual information between a stimulus and the observed responses into multiple trials. We show this using White noise signals, electrophysiological signals and histological data. Although this method does not give an absolute value of entropy or mutual information, it is mathematically correct, and its simplicity and broad use make it a powerful tool for their estimation through experiments.

1 Introduction

Entropy is the major component of information theory, conceptualized by Shannon in 1948 (Shannon, 1948). It is a dimensionless quantity representing uncertainty about the state of a continuous or discrete system or a collection of data. It is highly versatile as it applies to many different types of data, it can capture nonlinear interactions, and is model independent (Cover and Thomas, 2006). It has been widely used in the field of neurosciences, see (Borst and Theunissen, 1999; Timme and Lapish, 2018) for a more complete review of work; for example in the field of synaptic transmission (London et al., 2002), information rate of Action Potentials

(APs) (Bialek et al., 1991; Juusola and de Polavieja, 2003; Street, 2020) or connectivity studies (Ito et al., 2011; Vicente et al., 2011).

However, estimating the entropy of a signal can be a daunting task. The entropy H of a signal X is calculated with the well-known Shannon's formula:

$$\mathrm{H(X)} = -\sum_{i=1}^{N} p(x_i) \log_2 p(x_i) \qquad \text{(eq.1)}$$

Where $p(x_i)$ is the probability that the signal will take the x_i configuration in the probability distribution (x_1, x_2, x_3,..., x_N) of the signal. It is considered that if $p(x_i) = 0$, then $p(x_i)\log_2 p(x_i) = 0$, as $\lim_{x\to 0} x\,(\log_2 x) = 0$. And using a base 2 logarithm, entropy will be expressed in bits (Cover and Thomas, 2006; Shannon, 1948).

However, correctly estimating a probability distribution works only if each configuration happens many times. And by definition, one cannot know beforehand the number of needed experiments. This recording bias is even amplified by the fact that without making assumptions, there is no way to determine the relevant quantization and sampling of the data. The same recordings could be divided in any quantization bins and sampled by any interval, all giving different probability distributions and thus different entropy values.

As an example, let us consider the chain of characters A="04050405". It is unchanged with a quantization range "v" of 5, but will become "01010101" with a quantization range "v" of 2. If we now sample it with a bin "T" of 1 character, this will give a probability distribution of: p(0) = 0.5, p(4) = p(5) = 0.25 in the first scenario (v = 5) and: p(0) = p(1) = 0.5 in the second scenario (v = 2). We thus obtain different entropy values in the two scenarios: $H^{v=5,T=1} = 1.5$ and $H^{v=2,T=1} = 1$. Now, if we take a sampling bin of T = 2 we obtain p(04) = p(05) = 0.5 for v = 5 and p(01) = 1 for v = 2. The calculated entropies thus are: $H^{v=5,T=2} = 1$ and $H^{v=2,T=2} = 0$.

Without making assumptions on the data, there is no way to determine which value of entropy is the correct one. Therefore, quantization and sampling are crucial to determine the entropy of a signal. In an ideal case, we would need a method able to correct for the sample bias without making assumptions about the signal, meaning for any length of acquisition, any binning of width "T" and any quantization level "v" of the recorded data.

Thankfully there are multiple ways to use the direct formula and compensate for this bias, but none of them can be called trivial. There are for example the quadratic extrapolation method (Juusola and de Polavieja, 2003; de Polavieja et al., 2005; Strong et al., 1998), the Panzeri-Treves Bayesian estimation (Panzeri and Treves, 1996), the Best Universal Bound estimation (Paninski, 2003), the Nemenman-Shafee-Bialek method (Nemenman et al., 2004) or some more recent methods using statistic copulas (Ince et al., 2017; Safaai et al., 2018). Each method has its advantages and downsides (see (Panzeri et al., 2007) for a comparison of some of them), which leaves the experimenter puzzled and in dire need of a mathematician (Borst and Theunissen, 1999; Magri et al., 2009; Timme and Lapish, 2018).

However, there is another way to calculate the entropy of a signal, through what is called the Source Coding Theorem (Cover and Thomas, 2006; Larsson, 1996, 1999; Shannon, 1948; Wiegand, 2010) that to our knowledge has been used only once in the field of neurosciences, by (London et al., 2002) to calculate the information efficacy of a given synapse.

In signal processing, data compression is the process of encoding information using fewer bits than the original representation. In case of lossless compression, it does so by sorting parts of the signal by their redundancy and replacing them by shorter code words (Huffman, 1952; Shannon, 1948). However, the source coding theorem specifies that it is impossible to compress a signal of size N such that the length of the compressed signal is smaller than the entropy of the original signal multiplied by N. Therefore, with a perfect compression method the size of the compressed signal is proportional to the original signal entropy (Cover and Thomas, 2006; Larsson, 1996, 1999; Shannon, 1948; Wiegand, 2010).

When choosing this way of calculating entropy, the choice of the compression algorithm thus become critical as the compressed signal must be the smallest possible in order to represent the entropy of the original signal. It is of course possible to craft its own compression algorithm (see (London et al., 2002)), but thankfully this application has been broadly used in the domain of informatics, in order to compress text and images efficiently on the hard drive of a computer or before sending data through a network. In particular, this led to the development of two principal entropy-coding compression algorithms: the Huffman coding algorithm (Huffman, 1952) and the the Lempel–Ziv–Storer–Szymanski algorithm (Storer and Szymanski, 1982), both used to compress text and image files.

Portable Network Graphics (or PNG, see specifications at https://www.w3.org/TR/PNG/ or http://www.libpng.org/pub/png/) is a graphic file format supporting lossless data compression. Its high versatility and fidelity made it widely used for saving and displaying pictures. Its lossless compression is based on the combination of the Lempel–Ziv–Storer–Szymanski and Huffman algorithms and is called DEFLATE (Deutsch, 1996). Its great efficacy made it a reference for comparison with other entropy-encoding image compression methods (Bian et al., 2019; Cover and Thomas, 2006; Hou et al., 2020; Mentzer et al., 2020) and it is even used directly to estimate image entropy (Wagstaff and Corsetti, 2010).

In this paper, we propose that measurement of PNG file output size of neuroscientific data compressed using the PNG DEFLATE algorithm (in Bytes on the hard drive) is a reliable and unbiased proxy to estimate the entropy of different neuronal variables. From this simple step and by applying this method to electrophysiological and histological data we show that output data file size and entropy are related in a linear fashion and are robust enough to estimate entropy changes in response to different conditions. Further, with minimal modifications of the PNG file, we validate estimation of the mutual information between a stimulation protocol and the resulting experimental recording.

2 Materials and Methods

1 Neuronal modeling

A single compartment model was simulated with NEURON 7.7 (https://www.neuron.yale.edu/neuron/). All simulations were run with 100-µs time steps and had duration of 5 seconds. The nominal temperature was 37°C. The voltage dependence of activation and inactivation of Hodgkin-Huxley–based conductance models were taken from (Hu et al., 2009) for g_{Nav} and g_{KDR}. The equilibrium potentials for Na^+, K^+, and passive channels were set to +60, −90 and −77 mV, respectively. The conductances densities were set to 0.04 S/cm^2, 0.01 S/cm^2 and 3.33*10^{-5} S/cm^2 for g_{Nav} and g_{KDR} and passive channels, respectively. The model was stimulated using various numbers of excitatory synapses using the AlphaSynapse PointProcess of the NEURON software. The time constant and reversal potential were the same for every synapses and were set to 0.5 ms and 0 mV, respectively. The size of EPSPs produced by the synapses were randomly chosen using a lognormal distribution of EPSPs amplitude experimentally described in L5 pyramidal neurons (Lefort et al., 2009). Each synapse stimulated the model once during a simulation and the time onset was randomly chosen.

For the simulations of Figures 3A and 3B, the number of synapses simulating the model depended on the spiking frequency desired. For example, to calculate the information rate in the case of a 1 Hz spike train, we simulated the model with 400 of the synapses described above. We ran 20 trials of the simulation with the same train of synapses (Figure 2A). In order to introduce some jitter in the spiking times, we also injected a small gaussian current with a mean of 0 nA and a standard deviation of 0.0005 nA during the 5 seconds of the simulation. We reproduced this whole protocol for others desired spiking frequency, using increasing number of synapses (for example: 1900 synapses for a 19 Hz spiking).

For the simulations of Figure 3C, we stimulated the model with 750 of the synapses described above to get a spiking frequency around 5Hz. The time onsets and the amplitude of the synapses were randomly chosen at each simulation. We also added one supplementary synapse (Syn_supp) which stimulates the model every 200 ms (i.e 25 times in 5 s). The size of the EPSP size produced by this synapse was called wSyn_supp. When wSyn_supp was weak, this synapse did not drive the spiking of the model (Figure 3C, up left). When wSyn_supp was strong, this synapse drove the spiking of the model (Figure 3C, down left). We ran 100 simulations for each wSyn_supp.

2 Direct calculation of information rate via the quadratic extrapolation method

We calculated the entropy and information rates using the direct method and quadratic extrapolation, as described by (Juusola and de Polavieja, 2003; Panzeri et al., 2007; de Polavieja et al., 2005; Strong et al., 1998).

The first step of this method is to calculate the different values of entropy of the signal with different portions "Size" of the signal, different quantizations "v" of the signal values and different samplings bin "T". As described in the introduction, each modification of "Size", "v" or "T" will give a different value for the entropy. For example, in figures 3A and B, the parameter "Size" was successively set as 1, 0.9, 0.8, 0.7, 0.6, 0.5; "v" successively set as 2, 4, 8, 16, 32, 64, 128, 256 and the parameter "T" was successively set as 1, 2, 3, 4, 5, 6, 7, 8, which produced 6 * 8 * 8 = 384 distinct values of entropy for every trial. Entropy values for different trials of the same condition were averaged together. These values were then plotted against 1/Size and the intersections to 0 estimated by quadratic fit of the data. This gave us the entropy values for every "v" and "T", corrected for infinite size of the recordings. These values were then quadratic fitted against 1/v. The intersection to 0 gave us entropy values for every parameters of "T", corrected for infinite Size and infinite number of bins "v". Finally, these new values were fitted against 1/T and the extrapolation to 0 was estimated, to obtain the entropy value for infinite Size of the signals, infinite number of quantization levels "v" and infinite number of combination bins "T" (example in Fig. 1B). By performing this triple extrapolation and dividing by the time sampling, we can estimate the entropy rate R of the signal for theoretical infinite size, infinite number of quantization levels and infinite combination of sampling bins as

$$R = \lim_{T\to\infty} \frac{1}{T} \lim_{T\to\infty} \lim_{T\to\infty} H^{T,v,Size}$$

To obtain R_N, the entropy rate of the noise, instead of calculating H along the length of the signal, we did it at every time point across the successive trials. This is equivalent to simply transpose the signal and re-applying the same method than for R_S. Finally, we obtained the information rate by subtracting R_N to R_S as

$$R = R_S - R_N = \lim_{T\to\infty} \frac{1}{T} \lim_{v\to\infty} \lim_{Size\to\infty} \left(H_S^{T,v,Size} - H_N^{T,v,Size}\right)$$

For Figure 3A, simulated recordings were down-sampled to 10 kHz before calculation of information rate. For Figure 3C, middle, simulated recordings were down-sampled to 3 kHz and binned as 0 & 1 depending of the presence of Action Potentials or not, similar to (London et al., 2002).

For figures 1B, 1C, 1D, 2, 3A and 3B, The parameter "Size" was successively set as 1, 0.9, 0.8, 0.7, 0.6, 0.5; "v" successively set as 2, 4, 8, 16, 32, 64, 128, 256 and the parameter "T" was successively set as 1, 2, 3, 4, 5, 6, 7, 8. For figure 3C, the parameter "Size" was successively set as 1, 0.9, 0.8, 0.7, 0.6, 0.5; "v" set as 2 and the parameter "T" was successively set as 1, 2, 5, 10, 20, 30, 40. R_S, R_N and Information rate were calculated by successive quadratic extrapolations, as described above.

All of this was done by custom scripts written in Python 3.7 Anaconda with Numpy, Pandas and pyABF modules. These scripts are available at https://github.com/Sylvain-Deposit/PNG-Entropy.

3 Export to PNG format

Export to PNG was made with 3 different softwares: i) Anaconda 3.7 (https://www.anaconda.com/) and the pyPNG package (https://pypi.org/project/pypng/) for Figures 1, 2A, 3A, 3B ; ii) Labview 2017 and Vision 2017 (National Instruments) for Figures 2B, 3C, 4A and 4B; iii) the FIJI distribution of ImageJ software (Rueden et al., 2017; Schindelin et al., 2012) for figure 4C. Signals were normalized to 256 values from 0 to 255 simply by subtracting the minimal value of the signal, then dividing by the maximal value and multiplying by 255. It was then saved as PNG format in 8-bits range (256 grey values). For figure 2B and 3C, as the signal was binnarized we saved it with a 1-bit range (2 grey values).

A minimal file of PNG format is composed of a header and several parts of data, named critical chunks (https://www.w3.org/TR/PNG/#5DataRep). To these minimum requirements it is possible to add ancillary chunks (https://www.w3.org/TR/PNG/#11Ancillary-chunks) containing various information such as Software name, ICC profile, pixels dimensions, etc… If useful, this is hindering the estimation of entropy as it represents an overhead to the final size of the file. To estimate this overhead for each of our software we saved an image of 100 * 100 values of zeros, which corresponds to black in 8-bits grey levels and has an entropy of 0. With pyPNG, Fiji and Labview we obtained three PNG files of size 90, 90 and 870 Bytes, respectively. When repeating the experiment of figure 1, we obtained really similar linear fits of slopes 1.21 (R^2 = 0.99), 1.18 (R^2 = 0.99) and 1.21 (R^2 = 0.99) respectively.

For Figure 4C, we used Fiji for every image of the collection and we: i) extracted the channel number 2 containing the MAP2 staining; ii) converted the file to 8-bits grey levels; iii) thresholded it to remove every intensity values under 10 to remove most of the background; iv) saved the new file as PNG format, v) checked the size of this new file and vi) divided the size in kBytes by the number of soma visible in the field.

3 Results

1 Entropy and application to Gaussian noise

To test the usability of the PNG format to represent entropy, we started by generating 10 000 points of uniform white noise with an amplitude N of 2. As we are using white noise, we fully know the probability distribution (in that case, $p(xi) = 1/N$) and we can apply the eq. 1 and obtain an entropy H of 1 bit. We then repeated this noise model progressively increasing the amplitude by squares of 2 until 256, which corresponds to a maximum entropy of 8 bits (Figure 1A). The entropy rate R is defined as $R = \frac{1}{T}$H with T being the sampling of the signal. In this model case, we can take T = 1 to finally obtain an entropy rate in bits per samples.

As a control way to calculate the entropy of our signals, we used the quadratic extrapolation method which compensate for the sampling disaster (Juusola and de Polavieja, 2003; Panzeri et al., 2007; de Polavieja et al., 2005; Strong et al., 1998). Briefly, this method requires slicing our signal by multiple factors “Size”, quantizing it into multiple levels “v” of amplitudes, creating the probability distribution of words made of multiple combinations “T” of bins and finally calculating the entropy of the signal from this probability distribution. We thus obtain a great number of entropy values that we plot successively against 1/Size, 1/v and 1/T in order to fit them with quadratic extrapolations and find the intersection to 0 (Figure 1B, see Methods). By this triple extrapolation, we can estimate the entropy rate of the signal for theoretical infinite size, infinite quantization levels and infinite combination of sample bins as

$$R = \lim_{T \to \infty} \frac{1}{T} \lim_{v \to \infty} \lim_{Size \to \infty} H^{T,v,Size} \quad \text{(eq.2)}$$

In our simple model, we can see that when using only 2 different values, the final extrapolation to 0 gives an entropy rate really close to 1 as expected (Figure 1B, right panel).

However, when increasing the number of possible values until 256 (entropy H of 8) we start to unravel the so-called “sampling disaster” (Figure 1C). When performing the last extrapolation (Figure 1C, right panel), even if the upper bound has the correct value of 8 (most right point), a linear fit can be done only on the 2 last points and the intersection to 0 gives a value of around 5.7, far from the expected one. This is easily explained by the number of data points we have in our model signal. The last extrapolation concerns the “T” combinations of values to calculate the entropy rate and when using Gaussian white noise and with T equal to 1, it means we need 2^8 = 256 samples to properly estimate the probability distribution. However, if T increases to 2 we then need 2^{8*2} = 65536 samples to estimate the probability distribution, when we had only 10000. The probability distribution is thus insufficient to properly estimate the entropy rate of this signal with 256 values. Even by using quadratic extrapolations to compensate for the sampling bias, we can see that the direct method gives a wrong result for high entropy values and few data points (red arrowheads in Figure 1D, left panel).

As a comparison, we simply saved the 10 000 samples data of increasing noise as PNG image file with a depth of 8 bits (and thus 256 possible values) and measured the space taken by these files on the hard drive (Figure 1D, right panel). We obtained a linear increase versus the ideal entropy value of equation ($y = 1.199\,x + 0.609$, R^2 = 0.99). The PNG compression algorithm works with linearized data, which means there is no difference when saving pictures as a 100 * 100 square format (figure 1D, right panel, squares plot) or saving a 10 000 single line (same panel, crosses plot). Noticeably, there is no sampling disaster when using this method. From this first test, we can conclude that the size of a PNG file of a model signal has a linear relationship with its entropy rate. And that this measure is not biased by the sampling disaster.

As another example to better illustrate the concept of entropy and the effect of PNG conversion, we choose to generate 10 000 points of uniform white noise of amplitude 4 (thus an entropy H of 2 bits. Figure 2A, top left) and compare it to the really same signal, but with sorted values (Figure 2A, bottom left). These two signals look drastically different but they have the same probability distribution and if we use the Shannon's formula (eq. 1), they both have the same entropy. However, when the signal is sorted the uncertainty of each value is drastically reduced. By knowing any value we have a good guess of what will be the next one, so its entropy should be close to 0. This is exactly what happens when using the quadratic extrapolation to correct the sampling bias: the signal made of uniform white noise keeps its entropy of 2 bits, but the sorted signal sees its entropy reduced to 0.0048 bits per pixel (Figure 2A, middle). We then multiplied the size of the signal by 2 and 4 (thus 20 000 and 40 000 points): this does not change the probability distribution of the values and thus the entropy values stay low (Entropy of 0.0001 and 0.0016 bits per pixel, respectively) (Figure 2A, middle).

The PNG algorithm works by removing statistical redundancy in the signal, but by definition there is no redundancy in a signal made of uniform white noise. As a result, when we increase the size of the random signal by a factor 2 or 4 we increase the size of the corresponding file by the same factor (File sizes of 3.2, 7.1 and 14 kB, respectively) (Figure 2A, right). If we use the sorted signal, the redundancy is maximal and thus the file size is much smaller, showing the decrease in entropy. If we increase the size of this signal, we increase the redundancy and thus the file size stays low (File sizes of 0.13, 0.12 and 0.16 kB, respectively) (Figure 2A, right).

The entropy of a signal is independent of the mean value of this signal, as it is calculated from a probability distribution of values and not from the values themselves. To demonstrate that the PNG format behaves in the same way, we generated 10 000 points of uniform white noise with 2 possible values (0 and 1) and progressively increased the percentage of 1 in this signal, from 0 to 100% (Figure 2B, left). If we calculate the entropy of this signal by the Shannon's formula (eq.1), we obtain the classical bell-shaped curve as described by (Shannon, 1948) (Figure 2B, right). We simply saved these signals to PNG format and measured the size of the file on the hard drive, and we could see that this size follows the same curve than the entropy (Figure 2B, right).

From these examples, we conclude that correcting the sampling bias is critical to capture the entropy of a signal. We showed that the PNG conversion method does capture the intrinsic organization of a signal, that it follows its entropy and is independent of the values of the signal itself. However, the different signals need to be of the same size if we want to estimate their entropy by PNG conversion.

2 Mutual Information and application to electrophysiological data

Most of the time, the experimenter is not interested in the entropy itself, but in the mutual information between two variables X and Y. The mutual information "I_{XY}" measures the statistical dependence by the distance to the independent situation (Cover and Thomas, 2006; Shannon, 1948) given by

$$I_{XY} = \sum_{i,j} p(x_i, y_j) \log_2 \left(\frac{p(x_i,y_j)}{p(x_i)p(y_j)}\right) \quad \text{(eq.3)}$$

Therefore, when X and Y are independent $p(x_i, y_j) = p(x_i) * p(y_j)$, so $I_{XY} = 0$ bits.

The mutual information can also be rewritten as the difference between the entropy of X and the conditional entropy of X given Y:

$$I_{XY} = H(X) - H(X/Y) \quad \text{(eq.4)}$$

Where H(X) is the entropy already described (eq.1) and

$$H(X/Y) = -\sum_j p(y_j) \sum_i p(x_i, y_j) * \log_2 p(x_i, y_j) \quad \text{(eq.5)}$$

Interestingly, if we consider a neuron stimulated repetitively by the same stimulus, we can define X as the response of the neuron to the stimulus and Y as the stimulus received by the neuron (Borst and Theunissen, 1999). In that case, H(X) can be interpreted as the total entropy (quantifying the average variability of the neuron response during one trial), also called H_S. H(X/Y) can be interpreted as the noise entropy H_N, quantifying the variability of the neuron response across the trials. In that case,

$$I_{SN} = H_S - H_N \quad \text{(eq.6)}$$

Where Hs is the average entropy of every trial and:

$$H_N = -\langle \textstyle\sum_i p(x_i)_\tau * \log_2 p(x_i)_\tau \rangle_\tau \quad \text{(eq.7)}$$

where $p(x_i)_\tau$ is the probability of finding the configuration x_i at a time τ over all the acquired trials of an experiment (Juusola and de Polavieja, 2003; de Polavieja et al., 2005; Strong et al., 1998). Finally, we can obtain the information rate R, by using the quadratic extrapolation method and dividing by the time sampling, as:

$$R = R_S - R_N = \lim_{T\to\infty} \frac{1}{T} \lim_{v\to\infty} \lim_{Size\to\infty} \left(H_S^{T,v,Size} - H_N^{T,v,Size}\right) \quad \text{(eq.8)}$$

In practical terms, this means to acquire multiple recordings of the same experiments, apply the quadratic extrapolation method first on each trial and average the results to obtain R_S. Then, to apply the same method across the trials for every time τ and average the results to obtain R_N. The information rate R is thus the mutual information between the stimulation protocol and the acquired trials.

In order to apply this method to electrophysiological signals, we created a single compartment model in NEURON 7.7. The model was stimulated using various numbers of EPSPs with amplitudes chosen randomly in a log-normal distribution described in (Lefort et al., 2009). Each synapse stimulated the model once during a simulation and the time onset was randomly chosen. We ran 20 trials of the simulation with the same train of synapses (Figure 3A, left). In order to introduce some randomness in the spiking, a small Gaussian noise current was also injected (Figure 3A, left, See Methods). We calculated R_S and R_N using the quadratic extrapolation method. (Figure 3A, middle) and subtracting R_N to R_S to obtain the information rate (Figure 3A, right). We then reproduced this protocol with increasing numbers of synapses to obtain different spiking frequencies. As expected, when we increased the number of synapses we increased the spiking frequency and the mutual information between our stimulation and the response (Figure 3A, right). This measures follows a linear trend, really similar to what was already described in literature (Juusola and de Polavieja, 2003; de Polavieja et al., 2005).

As already described, the PNG format is line-wise. The compression algorithm will thus be sensitive to the orientation of the image we have to compress. To estimate the entropy of the signal, we started by converting our voltage signals to an 8-bits PNG image (256 levels of grey). As our signals are 20 trials of 5s at 10 kHz sampling, this yielded a 50 000 * 20 pixels of 256 grey scale image (Figure 3B, left). We saved this first version of the image and measured the size of the files on the hard drive. To estimate the entropy of the noise, we simply rotated the image 90 degrees and saved it again to PNG format. This constrains the algorithm to calculate the entropy through the acquired trials and not through the signal itself, thus estimating the entropy of the noise through every trials (Figure 3B, middle). We measured the size of the newly generated file and subtracted it to the previously measured for the signal entropy. As we can see, this difference of file size follows a linear behavior, increasing with AP frequency similarly to the direct measure of the information rate (Figure 3B, right).

As a second example, we reproduced the protocol made by (London et al., 2002) to estimate the information transfer between one synapse and the postsynaptic neuron. We stimulated the model with 750 of the synapses described above to get a spiking frequency around 5Hz and added a supplementary synapse stimulating the model every 200 ms. The EPSP size of this synapse (wSyn_supp) was modified at each simulation. When wSyn_supp was weak, this synapse did not drive the spiking of the model (Figure 3C, up left). When wSyn_supp was strong, this synapse drove the spiking of the model (Figure 3C, down left). We made 100 trials for each wSyn_supp and at each trial the onset time and amplitude of the others synapses were chosen randomly. We down-sampled our signal to 3 kHz and binarized it to 0 and 1, depending on the presence of APs or not similarly to (London et al., 2002). After calculating the Information rate using the quadratic extrapolation method, we obtained a sigmoid curve similar to what they described

(London et al., 2002) (Figure 3C, middle). In a similar way than above, we converted our voltage signals to a 1-bit PNG image (2 levels of grey). As our signals are 100 trials of 5s at 3 kHz sampling, this yielded an image of 15 000 * 100 pixels of 2 possible values. We measured the size of the file, then rotated the image 90 degrees, saved it again and measured the size of this new file. As expected, the difference between the sizes of these two files follows a sigmoid curve really similar to the one calculated by the quadratic extrapolation method (Figure 3C, right).

From this second test, we conclude that by saving multiple trials of the same experiment as a single PNG file, we can estimate the entropy of the signal. And by simply rotating this same file 90 degrees and saving it again, we can estimate the entropy of the noise. The difference between those two values follows the same behavior than measuring the Mutual Information between the stimulation protocol and the multiple recorded responses.

3 Application to 2D images

Another way to understand entropy is that it is a representation of complexity of a signal (Cover and Thomas, 2006). For example, (Wagstaff and Corsetti, 2010) used the PNG compression algorithm to evaluate the complexity of biogenic and abiogenic stromatolites. As a quick example, we took one of the famous drawings of a cortical column by Santiago Ramon y Cajal (Cajal, 1899) (Figure 4A, top, Wikimedia Commons). We saved each 1-pixel column of this picture as an 8-bits PNG file and measured its size on the hard drive. As we see, the size of the columns as PNG files changes with the different layers of the cortical stack, illustrating the differences in cell density and dendrites arborizations (Figure 4A, bottom).

In the same spirit, we used the ddAC Neuron example from the FIJI distribution of the ImageJ software (Schindelin et al., 2012). This reconstructed drosophila neuron (Figure 4Ba) is a classic example used for Sholl analysis (Ferreira et al., 2014; Sholl, 1953), (see https://imagej.net/Sholl_Analysis). This analysis estimates the complexity of an arborization by drawing concentric circles centered on the soma of the neuron and counting the number of intersections between those circles and the dendrites. The more intersections, the more complex is the dendritic tree. As we need PNG files with the same dimensions to be able to compare their sizes, we realized a cylindrical anamorphosis centered on the soma of the ddAC neuron (Figure 3Bb) and saved each column of this new rectangular image as PNG files. As a result, the size of those files grew with the distance from soma, reaching the same peak than a Sholl analysis made with default settings in Fiji (Figure 4Bc). Of course, it is also possible to simply tile the original image in smaller PNG files and save them independently. The size of these files will give an idea of the complexity of the area covered by the tile (Figure 4Bd).

In a final example, we used a group of images made by Dieter Brandner and Ginger Withers available in the Cell Image Library (http://cellimagelibrary.org/groups/3006). These images are under Creative Common Attribution Licence and show the growth of neuronal cultures from 2 to 7 days In-Vitro. They show two stainings, for tubulin and MAP2. They are suitable to our needs as all the images have the same dimensions and resolution. We kept only the MAP2 channel as it reveals the dendrite morphology, converted the images to 8-bits grey scale (256 grey levels) and thresholded them to remove the background (Figure 4C, left and middle). We then saved all the images to PNG, measured the size of the files on the hard drive and divided this number by the number of visible somas, in order to make a quick normalization by the culture density. As expected, this ratio File Size / Number of Cells increases with the number of days in culture, revealing the dendrite growth (Figure 4C, right).

From this third test, we showed that we can use the PNG format to estimate the entropy of 2D images as well, and this can be used to estimate dendrite growth or local complexity of an image.

4 Discussion

Entropy measurement can be a tool of choice in neuroscience, as it applies to many different types of data; it can capture nonlinear interactions, and is model independent. However, an accurate measure can be difficult as it is prone to a sampling bias depending of the size of the recorded signal, its quantization levels and its sampling. There are multiple ways to compensate for it, but none them trivial. In this paper, we showed that it is possible to estimate the entropy rate of neuroscience data simply by compressing them in PNG format and measuring the size of the file on the hard drive. The principle relies on the Source Coding Theorem specifying that one cannot compress a signal more than its entropy multiplied by its size. We showed first that the size of PNG files correlates linearly with the entropy of signals made of Uniform white noise. Then that we can estimate the information transfer rate between a stimulation protocol and the measured response simply by saving the responses as a PNG file, measuring the size and subtracting the size of the same file rotated 90 degrees. And finally, that we can generally use the PNG format to estimate complexity of two-dimensional data like neuronal arborization and in histological stainings.

1 Drawbacks

The main drawback of this method is that the PNG file size is not the absolute value of the entropy of the signal. Even if entropy bits and computer bytes do share similar names, in no cases should we exchange one for the other. A PNG file size is a way to estimate the evolution of entropy, considering all other parameters unchanged. As so, PNG files must be all of the same dimensions, of the same dynamic range and saved with the same software. A PNG file is composed of a header, critical chunks and non-essential ancillary chunks (See Methods). Different software will save different data in the ancillary chunks and thus will change the size of the file, independently of the compressed signal.

2 Advantages

The main advantage of this method is that it relies on previously developed compression algorithms that were already shown as optimal (Huffman, 1952). Moreover, it does not need any specialized software or any knowledge in programming language, as the PNG format is ubiquitous in informatics. For example, the ImageJ software is widely used in neuroscience and can export data as PNG.

A second advantage is the speed of execution. As an example, the Information Rate of neuronal signals (Figure 3A & B) took a bit more than 2 hours of calculation for the quadratic extrapolation method. Saving the same signals in PNG took less than 30 seconds.

As so, this method is extremely easy, quick, and does not need any knowledge in mathematics for correcting the sampling bias. It is interesting to note that an experimenter will often acquire multiple recordings of the same protocol in order to infer proper statistics. This means that most of the times no supplementary experiments are needed to calculate the entropy of a signal, or the information transfer rate between a stimulation protocol and its recorded result.

In conclusion, we propose this method as a quick-and-easy way to estimate the entropy of a signal or the information transfer rate between stimulation and recorded signals. It does not give the exact value of entropy or information, but it is related to it in a linear way and its evolution through different parameters follows a linear trend as well. And it is not affected by the sampling bias inherent to the direct way of calculating entropy.

3 Developments

We see multiple ways to improve this method. First, we saved our data as 8-bits PNG files, which limits the dynamic range of the file to 256 values. However, it is possible to save PNG natively as 1, 4, 8, 16 and 24 bits range, thus greatly increasing the dynamic range of the saved signal. Second, with some programming skills it is possible to remove the header and ancillary chunks of the PNG format, thus removing the size overhead (but the file will be unreadable by standard softwares). Finally, one possible way of improving the estimation of entropy rate would be to choose a better compression algorithm. We choose the PNG format as it is widely used by common softwares and it is based on LZSS and Huffman algorithms, which have been proven optimal. However, some algorithms may give a better compression rate depending on the quality of the data. As an example, the Rice compression algorithm was originally developed for the NASA Voyager missions (Rice and Plaunt, 1971). It is suboptimal but is better suited for noisy signals of low values.

In a more general direction, it is important to note that this method works with any entropy-coding compression algorithm, as long as they are loss-less. This is the case of GZip algorithms for example, used in many compression softwares such as WinRAR, PKZIP, ARJ, etc… It is thus not limited to pictures in PNG, although this format is useful for rotating the file and estimating the mutual information easily. Moreover, we apply these algorithms to 2D images, when actually the algorithm linearizes the data and works only in linear way on one dimension. There are some attempts to generalize Shannon entropy to 2D space (Larkin, 2016), but they are out of the scope of this paper.

5 References

Bialek, W., Rieke, F., Steveninck, R. de R. van, and Warland, D. (1991). Reading a neural code. Science *252*, 1854–1857.

Bian, N., Liang, F., Fu, H., and Lei, B. (2019). A Deep Image Compression Framework for Face Recognition. ArXiv190701714 Cs.

Borst, A., and Theunissen, F.E. (1999). Information theory and neural coding. Nat. Neurosci. *2*, 947–957.

Cajal, S.R. (1899). Comparative study of the sensory areas of the human cortex.

Cover, T., and Thomas, J. (2006). Elements of Information Theory, 2nd Edition | Wiley.

Deutsch, P. (1996). DEFLATE Compressed Data Format Specification version 1.3.

Ferreira, T.A., Blackman, A.V., Oyrer, J., Jayabal, S., Chung, A.J., Watt, A.J., Sjöström, P.J., and van Meyel, D.J. (2014). Neuronal morphometry directly from bitmap images. Nat. Methods *11*, 982–984.

Hou, Y., Zheng, L., and Gould, S. (2020). Learning to Structure an Image with Few Colors. ArXiv200307848 Cs.

Hu, W., Tian, C., Li, T., Yang, M., Hou, H., and Shu, Y. (2009). Distinct contributions of Nav1.6 and Nav1.2 in action potential initiation and backpropagation. Nat. Neurosci. *12*, 996–1002.

Huffman, D.A. (1952). A Method for the Construction of Minimum-Redundancy Codes. Proc. IRE *40*, 1098–1101.

Ince, R.A.A., Giordano, B.L., Kayser, C., Rousselet, G.A., Gross, J., and Schyns, P.G. (2017). A statistical framework for neuroimaging data analysis based on mutual information estimated via a gaussian copula. Hum. Brain Mapp. *38*, 1541–1573.

Ito, S., Hansen, M.E., Heiland, R., Lumsdaine, A., Litke, A.M., and Beggs, J.M. (2011). Extending Transfer Entropy Improves Identification of Effective Connectivity in a Spiking Cortical Network Model. PLoS ONE *6*.

Juusola, M., and de Polavieja, G.G. (2003). The rate of information transfer of naturalistic stimulation by graded potentials. J. Gen. Physiol. *122*, 191–206.

Larkin, K.G. (2016). Reflections on Shannon Information: In search of a natural information-entropy for images. ArXiv160901117 Cs Math.

Larsson, N.J. (1996). Extended Application of Suffix Trees to Data Compression. In In Data Compression Conference, pp. 190–199.

Larsson, N.J. (1999). Structures of String Matching and Data Compression. thesis/docmono. Lund University.

Lefort, S., Tomm, C., Floyd Sarria, J.-C., and Petersen, C.C.H. (2009). The excitatory neuronal network of the C2 barrel column in mouse primary somatosensory cortex. Neuron *61*, 301–316.

London, M., Schreibman, A., Häusser, M., Larkum, M.E., and Segev, I. (2002). The information efficacy of a synapse. Nat. Neurosci. *5*, 332–340.

Magri, C., Whittingstall, K., Singh, V., Logothetis, N.K., and Panzeri, S. (2009). A toolbox for the fast information analysis of multiple-site LFP, EEG and spike train recordings. BMC Neurosci. *10*, 81.

Mentzer, F., Van Gool, L., and Tschannen, M. (2020). Learning Better Lossless Compression Using Lossy Compression. ArXiv200310184 Cs Eess.

Nemenman, I., Bialek, W., and de Ruyter van Steveninck, R. (2004). Entropy and information in neural spike trains: progress on the sampling problem. Phys. Rev. E Stat. Nonlin. Soft Matter Phys. *69*, 056111.

Paninski, L. (2003). Convergence properties of three spike-triggered analysis techniques. Netw. Bristol Engl. *14*, 437–464.

Panzeri, S., and Treves, A. (1996). Analytical estimates of limited sampling biases in different information measures. Netw. Bristol Engl. *7*, 87–107.

Panzeri, S., Senatore, R., Montemurro, M.A., and Petersen, R.S. (2007). Correcting for the sampling bias problem in spike train information measures. J. Neurophysiol. *98*, 1064–1072.

de Polavieja, G.G., Harsch, A., Kleppe, I., Robinson, H.P.C., and Juusola, M. (2005). Stimulus history reliably shapes action potential waveforms of cortical neurons. J. Neurosci. Off. J. Soc. Neurosci. *25*, 5657–5665.

Rice, R., and Plaunt, J. (1971). Adaptive Variable-Length Coding for Efficient Compression of Spacecraft Television Data. IEEE Trans. Commun. Technol. *19*, 889–897.

Rueden, C.T., Schindelin, J., Hiner, M.C., DeZonia, B.E., Walter, A.E., Arena, E.T., and Eliceiri, K.W. (2017). ImageJ2: ImageJ for the next generation of scientific image data. BMC Bioinformatics *18*, 529.

Safaai, H., Onken, A., Harvey, C.D., and Panzeri, S. (2018). Information estimation using nonparametric copulas. Phys. Rev. E *98*.

Schindelin, J., Arganda-Carreras, I., Frise, E., Kaynig, V., Longair, M., Pietzsch, T., Preibisch, S., Rueden, C., Saalfeld, S., Schmid, B., et al. (2012). Fiji: an open-source platform for biological-image analysis. Nat. Methods *9*, 676–682.

Shannon, C.E. (1948). A Mathematical Theory of Communication. Bell Syst. Tech. J. *27*, 379–423.

Sholl, D.A. (1953). Dendritic organization in the neurons of the visual and motor cortices of the cat. J. Anat. *87*, 387–406.

Storer, J.A., and Szymanski, T.G. (1982). Data compression via textual substitution. J. ACM *29*, 928–951.

Street, S. (2020). Upper Limit on the Thermodynamic Information Content of an Action Potential. Front. Comput. Neurosci. *14*, 37.

Strong, S.P., Koberle, R., de Ruyter van Steveninck, R.R., and Bialek, W. (1998). Entropy and Information in Neural Spike Trains. Phys. Rev. Lett. *80*, 197–200.

Timme, N.M., and Lapish, C. (2018). A Tutorial for Information Theory in Neuroscience. ENeuro *5*.

Vicente, R., Wibral, M., Lindner, M., and Pipa, G. (2011). Transfer entropy—a model-free measure of effective connectivity for the neurosciences. J. Comput. Neurosci. *30*, 45–67.

Wagstaff, K.L., and Corsetti, F.A. (2010). An evaluation of information-theoretic methods for detecting structural microbial biosignatures. Astrobiology *10*, 363–379.

Wiegand, T. (2010). Source Coding: Part I of Fundamentals of Source and Video Coding. Found. Trends® Signal Process. *4*, 1–222.

6 Funding

This work was funded by Marie-Curie Fellowship IF-746247 Astromodulation, by the Wellcome Trust Principal Fellowship (212251/Z/18/Z), and by the ERC Proof of Principle grant (767372 – NEUROCLOUD).

7 Acknowledgements

We thank Q. Bernard, C. Leterrier, V. Marra, T. Jensen, K. Zheng for helpful discussion.

Figure Legends

1 Figure 1: comparison of entropy and PNG file size on a model case.

A) Examples of 10 000 data points of uniform white noise with growing number of grey levels and growing entropy (here showed as square picture signals). Left: 2 possible grey values, or entropy of 1. Right: 256 possible grey values, or entropy of 8.

B) Direct calculation and quadratic extrapolations to 0 to calculate the entropy rate of the left signal in A). Left: Plotting all the entropy values to 1/Size and extrapolating to 0 to get the value for infinite size (white arrowhead). For clarity, only the condition for v = 2 is shown. Middle: plotting the obtained values at left to 1/v and extrapolating to 0 to get the value for infinite number of binning (white arrowhead). Right: plotting the final values to 1/T and extrapolating to 0 to get the value for infinite number of combinations of letters (black arrowhead). Note that this value is really close to 1, as expected when using a signal made of uniform white noise with 2 possible values.

C) Direct calculation and quadratic extrapolations to 0 to calculate the entropy rate of the right signal in A). Left: Only the condition for v = 2 is shown. Note that in the final graph (right), points do not follow a linear trend. When using the last 2 points for extrapolation to 0, the value is far from the expected value of 8 (red arrowhead).

D) Left: when plotted against the known entropy value, the quadratic extrapolation method shows examples of sampling disaster for high values of entropy (red arrowheads). Right: when simply saving all the signals described in A), the file size in kBytes shows a linear relationship with the signal entropy ($y = 1.199\,x + 0.609$, $R^2 = 0.99$). Not that this true for pictures made either of square signals (squares plot) or linearized signals (crosses plot) and there is no sampling disaster with this method.

2 Figure 2: Effect of redundancy and signal size on entropy and PNG file size.

A) Left, Top: Example of 10 000 data points of uniform white noise with 4 levels (entropy of 2 bits, here showed as square picture signals). Left, Bottom: the same signal, with sorted values. If we use the Shannons's formula for calculating entropy, both signals have an entropy of 2 bits. Middle: Effect of sampling bias correction and size of the signal on the entropy value. With the Shannon's formula, both random and sorted signals have the same entropy value (crosses plot). If we correct the sampling bias with the quadratic extrapolation method, the entropy of the sorted signal decreases dramatically (squares plot). In both cases, changing the size of the signal does not change the entropy value, as it does not change the probability distribution of each value in the signal. Right: As the PNG algorithm compresses by removing statistical redundancy in the signal, the file size will grow with the size of the random signal. A sorted signal has a maximal redundancy and thus its file size will stay almost constant when increasing the signal size.

B) Left: Example of multiple signals of 10 000 data points (here showed as square picture signals) with 2 possible grey levels (or 0 and 1) and a growing percentage of white (or 1). Right: Direct entropy calculation of these signals (grey dashed line) and normalized PNG file size as a comparison (black line).

3 Figure 3: Comparison of information rate and PNG file size on a neuronal model.

A) Left: example of 20 generated trials with the same synaptic activity. Due to the injection of a small Gaussian noise current, we obtain variability in the spiking of the different trials. Arrows show the direction used with the quadratic extrapolation method to calculate the signal entropy (R_S) and noise entropy (R_N). Middle: calculation of the Entropy Rate of the Signal (Rs) and the noise (Rn) for the full voltage of the cell for each condition. The Information transfer Rate R is the difference between Rs and Rn. Right: Information transfer Rate between the synaptic stimulation and the neuronal activity, versus the spiking frequency. This follows a linear trend ($y = 25.935\,x + 188.15$, $R^2 = 0.97$).

B) Left: Conversion of the modeled trials in A) as a 256 grey values PNG file. As the PNG conversion algorithm is line-wise, we have to save the image a first time, then rotate it 90 degrees and save it a second time (Middle). Arrows show the direction of compression. Right: the subtraction of the 2 images sizes follows a linear trend with the spiking frequency ($y = 2.555\,x - 4.12$, $R^2 = 0.98$).

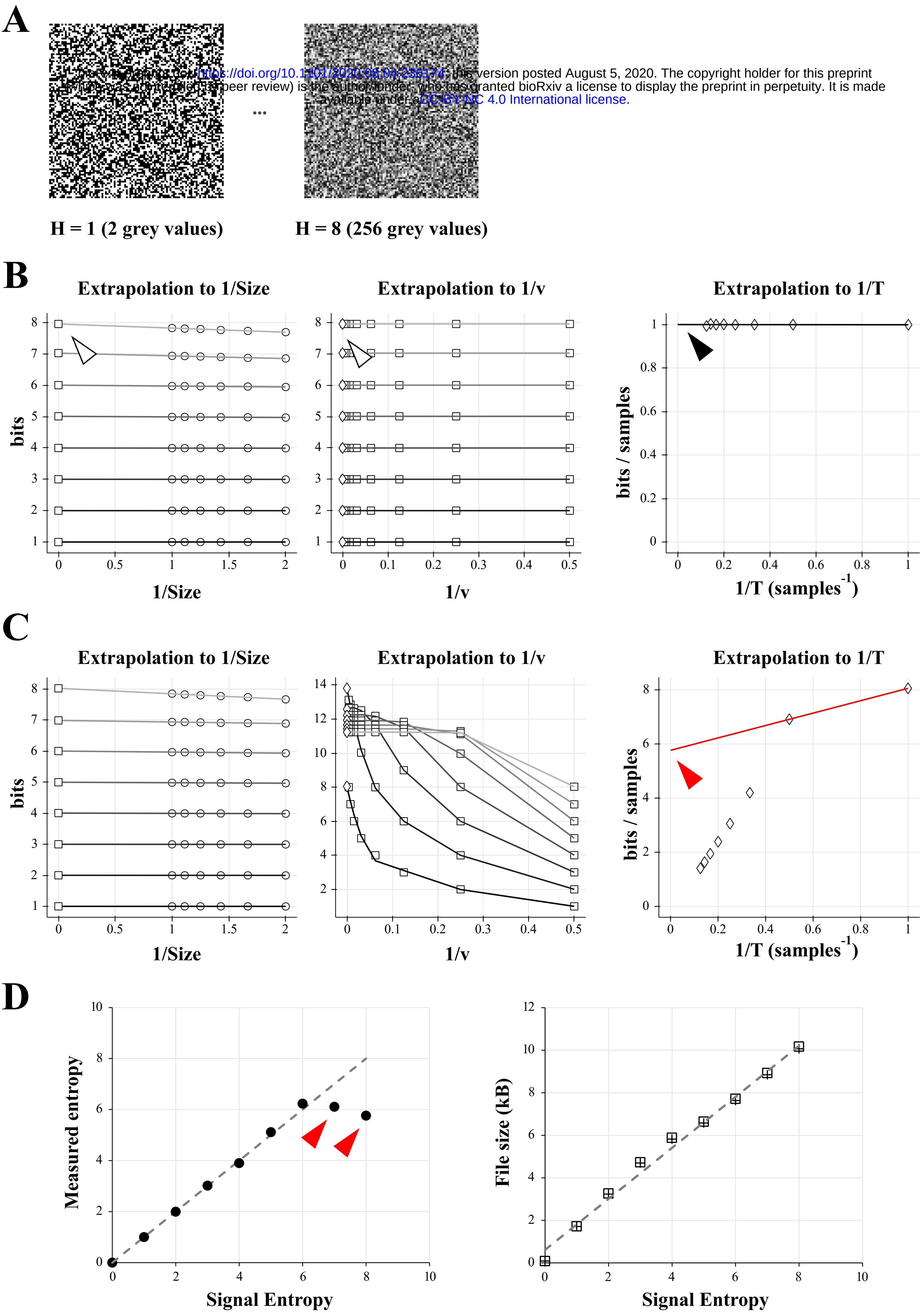
C) Left: examples of rastergrams showing the impact of the supplementary synapse on our neuronal model. With a low synaptic strength (wSyn_supp = 2 mV, top), this synapse barely drives the model. With a high synaptic strength (wSyn_supp = 6, bottom), the neuron spiking starts to synchronize with the occurrence of the synapse. Middle: Information transfer Rate between the activity of the supplementary synapse and the neuronal spiking. As expected, it follows a sigmoidal behavior. Right: rastergrams were saved as PNG files, rotated 90 degrees and saved again. The difference of the 2 files sizes follows a similar curve than the Information transfer Rate.

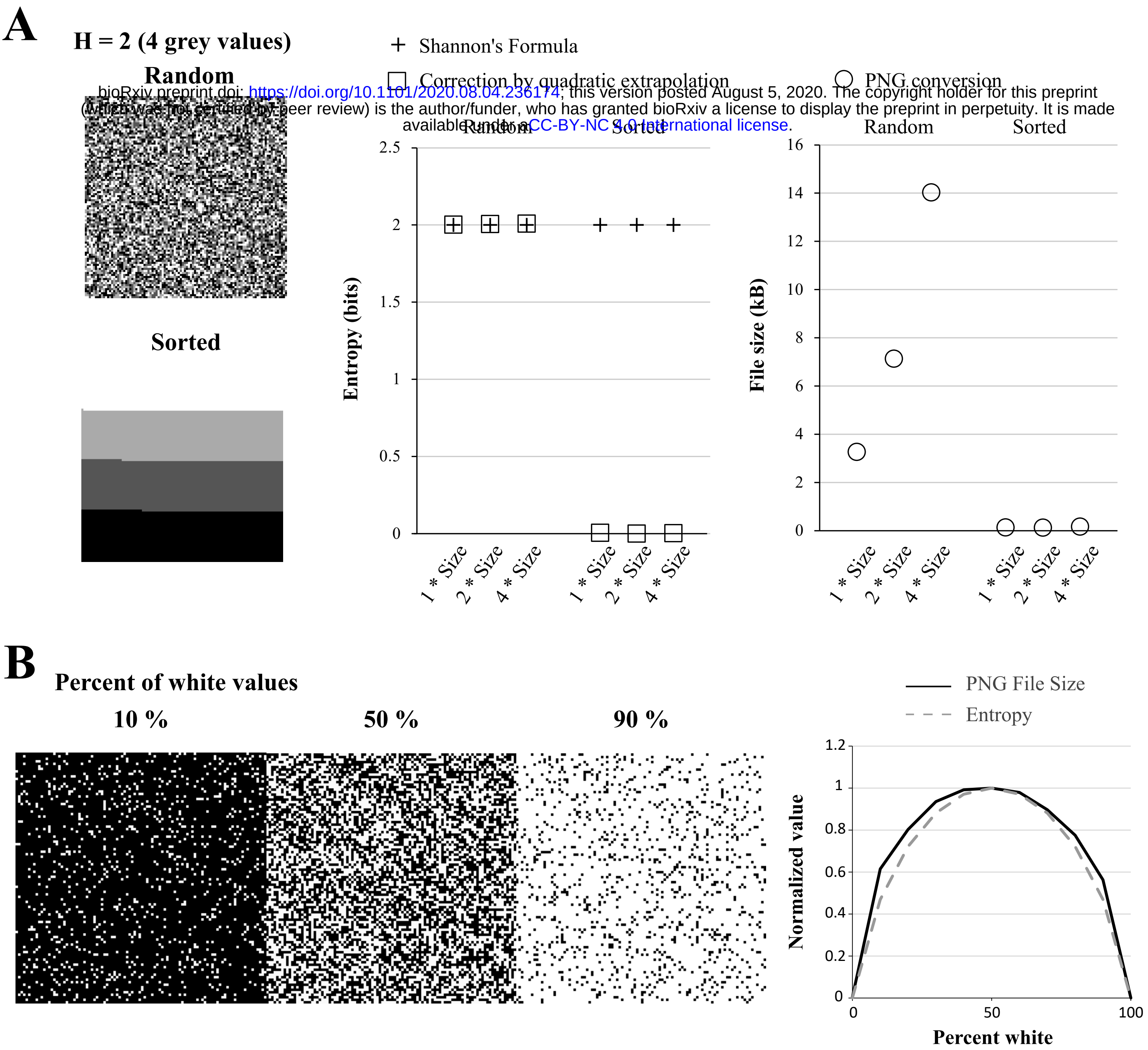
4 Figure 4: Application to 2D data.

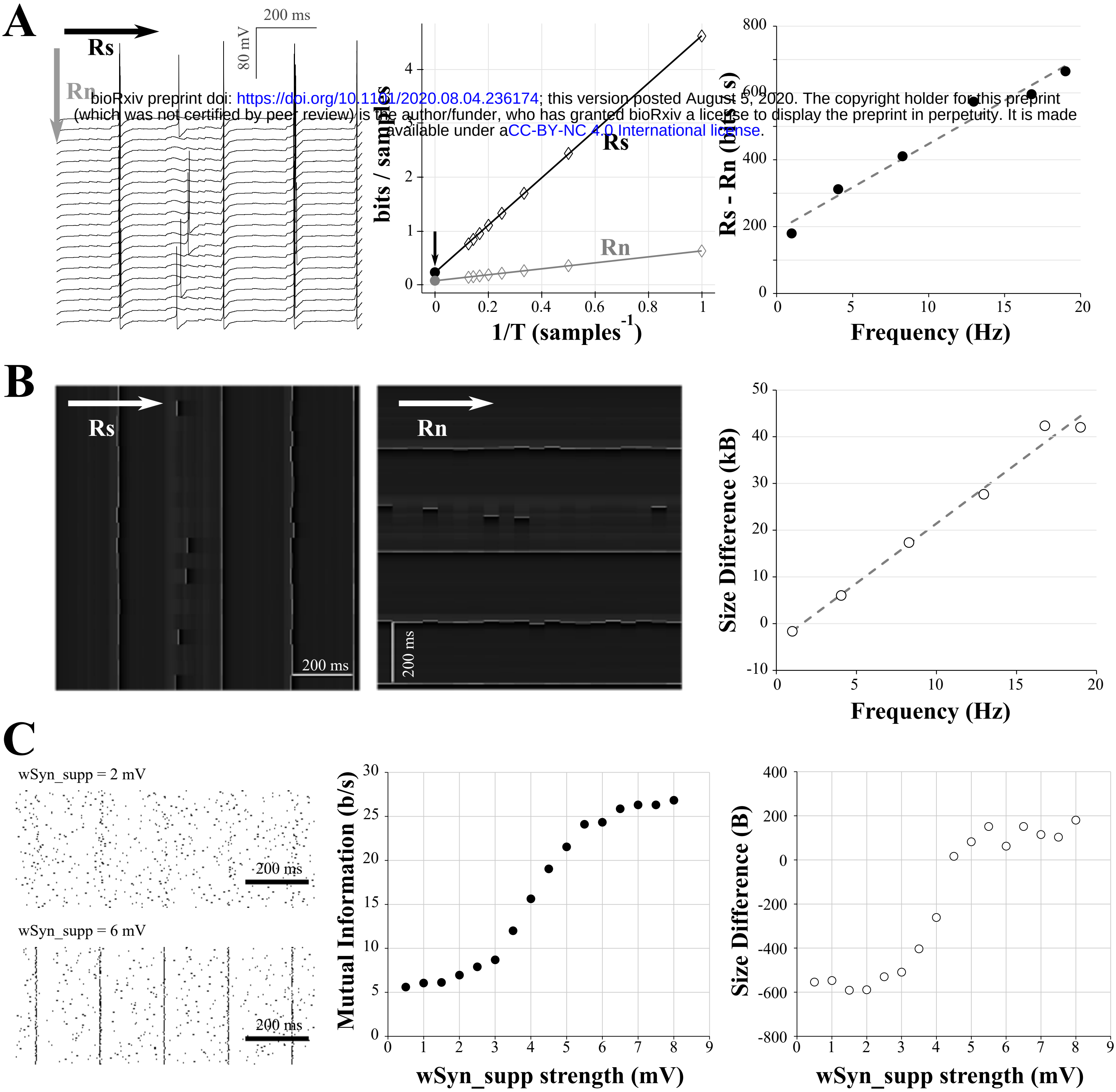
A) Top: original drawing of a cortical column by Santiago Ramon y Cajal. Arrow: each column of pixel was saved as a PNG file. Bottom: File size of each column as PNG, revealing the change in organization and complexity of the different layers.

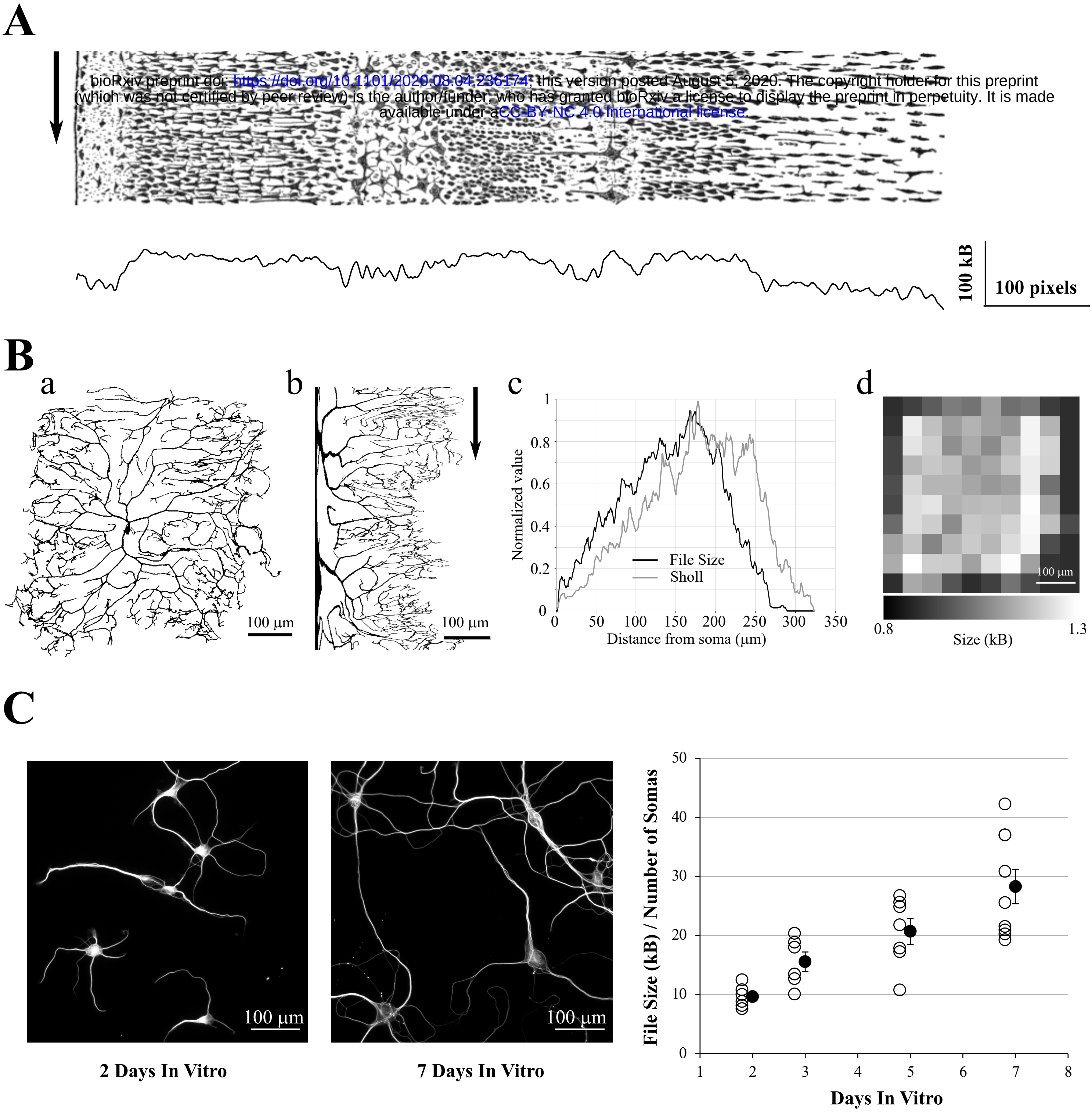
B) a: ddAC Neuron from Fiji examples. b: The same neuron after a circular anamorphosis centered on the soma. Note how the complexity of the dendritic arbor changes versus distance from soma. Arrow: each column of pixel was saved as a single PNG file. c: file size of these columns as PNG, showing the growth in complexity of the dendritic arborization (black line). As a comparison, we performed a Sholl analysis of the same image with default FIJI parameters (grey line). d: the original image was tiled in 10 squares, and each square saved as a PNG file. The sizes of these files reveal the heterogeneity of the dendritic arborization.

C) Left and Middle: examples of MAP2 stainings of Brandner and Withers neuronal cultures at 2 and 7 Days In Vitro. Note the growth in dendritic arborization through time. Right: Each image was saved into a PNG file, and the file size divided by the number of visible somas. This gives us a file size normalized by the density of the culture. This value increases with the number of Days In-Vitro, revealing the dendrite growth.


A
...
H = 1 (2 grey values)
H = 8 (256 grey values)
B
Extrapolation to 1/Size
Extrapolation to 1/v
Extrapolation to 1/T
bits
bits / samples
1/Size
1/v
1/T (samples^{-1})
C
Extrapolation to 1/Size
Extrapolation to 1/v
Extrapolation to 1/T
bits
bits / samples
1/Size
1/v
1/T (samples^{-1})
D
Measured entropy
Signal Entropy
File size (kB)
Signal Entropy


A
H = 2 (4 grey values)
Random
Sorted
Shannon's Formula
Correction by quadratic extrapolation
PNG conversion
Random
Sorted
Entropy (bits)
1 * Size
2 * Size
4 * Size
Random
Sorted
File size (kB)
B
Percent of white values
10 %
50 %
90 %
PNG File Size
Entropy
Normalized value
Percent white


A
Rs
Rn
200 ms
80 mV
bits / samples
1/T (samples^{-1})
Rs - Rn (bits/s)
Frequency (Hz)
B
Rs
Rn
200 ms
200 ms
Size Difference (kB)
Frequency (Hz)
C
wSyn_supp = 2 mV
200 ms
wSyn_supp = 6 mV
200 ms
Mutual Information (b/s)
wSyn_supp strength (mV)
Size Difference (B)
wSyn_supp strength (mV)


A
100 kB
100 pixels
B
a
100 µm
b
100 µm
c
Normalized value
File Size
Sholl
Distance from soma (µm)
d
100 µm
0.8
1.3
Size (kB)
C
100 µm
100 µm
2 Days In Vitro
7 Days In Vitro
File Size (kB) / Number of Somas
Days In Vitro