Measuring complexity using information fluctuation: a tutorial[1]

By John E. Bates[2]

Wherein we seek a quantitative measure of complexity that is consistent with intuitive notions, following a path laid out in the early 1990's in the paper Measuring complexity using information fluctuation by Bates and Shepard.

1. Complex vs. complicated

By *complex* we do not mean *complicated*. Rather, we mean something interesting or meaningful resulting from relatively simple components that interact or combine in such a way as to produce emergent behaviors or properties. Our goal, then, is to formulate a mathematically precise definition of complexity that distinguishes it from complication.

To illustrate this distinction, picture the screen of a television tuned to a station that is not currently broadcasting. Such a screen is filled with an ever-changing random arrangement of pixels, reminiscent of snow. Two snapshots of such a screen, taken at different times, are shown in Figure 1.

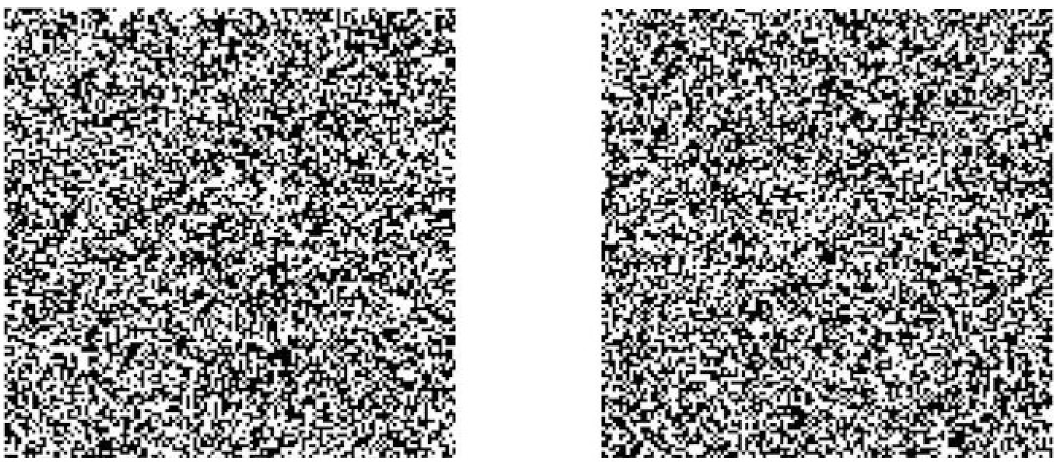

Figure 1: Two snapshots at different times of a television screen showing "snow".

The two "snowy" screens look similar: both are uninteresting and meaningless, containing no discernible pattern. Yet, they are complicated in the sense that much information is needed to accurately describe each one in detail. Essentially, the black/white state of every pixel must be specified; there is no briefer exact description.

But isn't one snowy screen as good as another? Describing all such screens simply as *snowy* requires much less information. And to create a generic snowy screen, all we need is a source of randomness such as a white noise signal or a random number generator. So, we consider randomness to be simple rather than complex.

Notice the important role that the concept of information is already playing in our search. It is a fundamental concept so that if we formulate a definition of complexity in terms of information it should be generally applicable.

[1] Revised November 23, 2024.

[2] In memory of Harvey K. Shepard – teacher, mentor, friend.

2. Information and probability

Information is defined in terms of probability. For example, tossing a two-sided coin generates exactly one *bit* of information because each side is equally likely to appear. The probability of the head side appearing is 50% or ½. If we let *1* denote that the outcome of a toss is a head and *0* denote a tail, then a coin toss can be encoded as a single binary digit or bit. Knowing the outcome of a coin toss is equivalent to knowing the value of the associated bit, be it 1 or 0. Thus, tossing a coin can be said to *generate* one bit of information.

The advantage of using a code is that many combinations can be encoded compactly. For example, suppose our code consists of two bits. There are four combinations of two bits: 00, 01, 10, 11. These combinations can encode anything that has four equally probable outcomes such as tossing a four-sided die in the shape of a tetrahedron. The number of possible outcomes increases exponentially with the number of bits. Three bits can encode $2^3 = 8$ possible outcomes, and so on.

So, the information content of an outcome that occurs once every N tries is the exponent to which 2 is raised to, to produce N. In general, the information I contained in an outcome that occurs with probability $P = 1/N$ is

$$I = \log N = -\log P$$

where *log* is the logarithm operation; it is the inverse of exponentiation and so gives the exponent in question. The units of I are *bits* if the base of the logarithm is 2 and the code is binary, but they need not be. The minus sign tells us that the rarer an outcome is, the more information is generated when it does occur.

3. Order and chaos

An *orderly* system is insensitive to external stimuli and so is stable. In the simplest case, it is attracted to a stable state and tends to remain there regardless of disturbances. This behavior can be visualized with a *state diagram*, in which a circle represents a state and arrows represent transitions between states as in Figure 2.

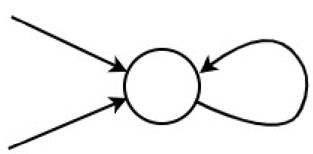

Figure 2: A fragment of a state diagram showing orderly behavior.

Only the stable state is shown. There are other transient states[3] not shown which all lead to the stable state which in turn leads back to itself. Upon entering its stable state, the system "forgets" how it arrived there. When multiple incoming arrows *converge* on a state, information is lost by the system during a transition to that state.

[3] We will only consider the long-term behavior of systems that have reached dynamic equilibrium, for which the concept of probability has meaning, so that we can study information flow. Transitions from transient states will be considered to have a negligible probability of occurrence and therefore will not contribute to our complexity measure. The system of Figure 2 exemplifies order only initially or in response to temporary disturbances but in the long term its probability of being in the stable state approaches one. For our purpose it will be considered to be an isolated state, a periodic system with period 1. As such, it will be classified as *simple* – purely periodic systems neither gain information from their environment nor lose information internally.

A *chaotic* system is sensitive to external stimuli and so is unstable. Small changes in initial conditions can result in drastically altered outcomes. Even slight disturbances can spread and possibly multiply. A chaotic system is not random in itself but can behave randomly if its external stimulus is random, as in the case of a television screen displaying the white noise signal it receives in the absence of a station signal as in Figure 1. Chaotic behavior is signified by *diverging* outgoing arrows in a state diagram, as in Figure 3.

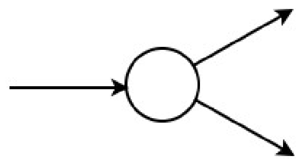

Figure 3: A fragment of a state diagram showing chaotic behavior.

To exit the state shown, information is needed to determine which transition arrow to follow. This information is generated by an external stimulus which the system then "remembers" by following a particular transition arrow, thus initiating a particular trajectory in state space. When multiple outgoing arrows *diverge* from a state, information is gained by the system during a transition from that state.

A classic example of chaos is the butterfly effect, such that a butterfly flapping its wings in China might result in a storm in North America days later. This could happen if the atmosphere has a chaotic aspect to its dynamics. But clearly the atmosphere also has an orderly aspect, otherwise weather would never be stable. The atmosphere is not extremely sensitive everywhere and at all times and so the butterfly stimulus would have to be in the right place at the right time, just where and when chaos is present. And so the atmosphere is actually an example of a *complex* system.

4. Complexity

A complex system is one in which both order and chaos are present and moreover, alternate in predominance: a dynamic balance of opposites. The state diagram of a complex system would therefore exhibit both converging and diverging transition arrows, alternating in predominance as the system evolves over time.

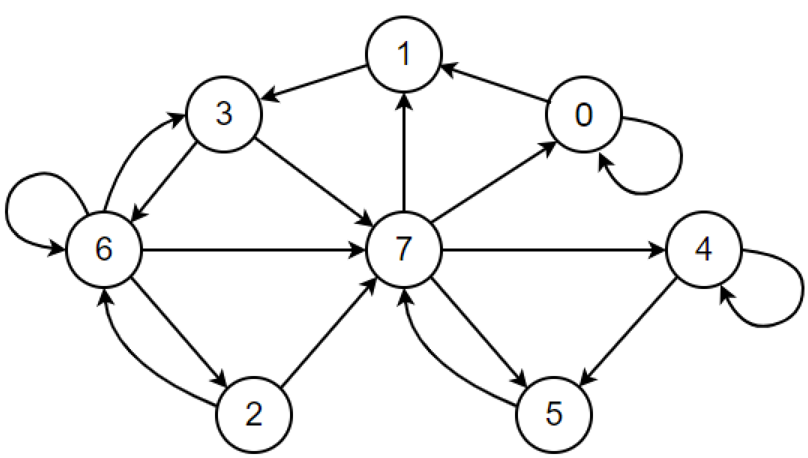

Figure 4: The state diagram of a complex system.

For example, consider the state diagram of a complex system as shown in Figure 4. States 0, 1, 4 and 5 have converging incoming transition arrows but no diverging outgoing arrows, and so order prevails in these regions of state space. State 2 has diverging outgoing arrows but no converging incoming arrows, and so chaos prevails there. States 3, 6 and 7 are mixed orderly/chaotic regions of state space. Overall, order and chaos alternate in predominance.

Consider the states 0 and 4. Both states can follow state 7, depending on the value of the external stimulus that governs which of the diverging arrows leaving state 7 is followed. Let us assume that the next state after state 7 is state 0. By being in state 0 and lingering there before proceeding to state 1, the system remembers the value of the stimulus that brought it there. Similarly, a different stimulus value might cause state 4 to follow state 7 where it may linger before proceeding to state 5, thus remembering the alternate value.

The memory may be temporary or quasi-permanent depending on how long the system lingers in state 0 or 4. Eventually, a certain combination of additional input stimuli may cause the system to return to state 7 by way of converging arrows, thus forgetting the original stimulus and ready to store a new memory.

The alternation of order and chaos makes possible an essential element of non-trivial computation: memory. It so happens that the state diagram of Figure 4 is that of an open-ended group of three adjacent cells, each of which follows the well-known rule 110 variant of the elementary cellular automaton which has been proven to be capable of universal computation. We will examine this system in detail in Section 7 below.

5 Measuring complexity

Now that we understand complexity as an alternation of the predominance of order (insensitivity to external stimulus) and chaos (sensitivity to external stimulus) in the state space of a system that evolves over time (a dynamic system), we can define the complexity of such a system as the *fluctuation of information* when the system is in dynamic equilibrium under the influence of random stimulation, i.e., when it is driven by a rich information source.[4] Let us formulate this definition precisely.

5.1 Fluctuation of net information gain

When several outgoing transition arrows diverge from a state in a state diagram, each one has a *forward conditional probability* $P_{i \to j}$, the probability that if the present state is i then the next state will be j. The external information needed to select between them is gained by the system when a transition actually occurs.

When several incoming arrows converge on a state, each one has a *reverse conditional probability* $P_{i \leftarrow j}$, the probability that if the present state is j then the previous state was i. The information about the previous state is lost when state j becomes the new present state.

The conditional probabilities can be estimated by observing the system over a long period of time and can also be determined analytically if the internal dynamics of the system are known.

We define the *net information gain* Γ_{ij} during a transition from a present state i to the next state j, as the information gained when leaving state i less the information lost when entering state j:

[4] The purpose of driving a system with a rich information source such as a random number generator or a white noise signal is to probe the internal dynamics of the system in much the same way as a frequency-rich impulse is used in signal processing.

$\Gamma_{ij} = -\log P_{i\to j} + \log P_{i\leftarrow j}$.

To capture the fluctuation of net information gain we can compute the root-mean-square deviation (standard deviation) σ_Γ of Γ about its mean, such that

$$\sigma_\Gamma^2 = \langle(\Gamma - \langle\Gamma\rangle)^2\rangle.$$

The symbol <> represents the weighted mean or average of whatever is enclosed. For example,

$$\langle\Gamma\rangle = \Sigma_{ij} P_{ij} \cdot \Gamma_{ij}$$

where P_{ij} is the probability that a transition from state i to state j occurs and where Σ_{ij} indicates a sum over all transitions.

The mean of the net information gain $\langle\Gamma\rangle$ is always zero[5]. We are left with

$$\sigma_\Gamma^2 = \langle\Gamma^2\rangle = \Sigma_{ij} P_{ij} \cdot \Gamma_{ij}^2$$

as a possible measure of complexity. This is a good first step; however, σ_Γ is limited by the fact that it is based on the net information gain of single transitions between pairs of states and thus does not take into account the cumulative effect of multiple transitions. This limitation is overcome in the next section.

5.2 Fluctuation of state information

The information content I_i of state i is given by

$$I_i = -\log P_i$$

where P_i is the probability that the system is in state i. With a little thought it is evident that the transition probabilities P_{ij} are related to the state and conditional probabilities by

$$P_{ij} = P_i \cdot P_{i\to j} = P_{i\leftarrow j} \cdot P_j.$$

Eliminating the conditional probabilities from the above definition of Γ_{ij} we have

$$\Gamma_{ij} = -\log (P_{ij} / P_i) + \log (P_{ij} / P_j)$$

and applying the rule for the logarithm of a ratio gives

$$\Gamma_{ij} = \log P_i - \log P_j = I_j - I_i.$$

So we see that the net information gain of a transition *only depends on the information content of the initial and final states* I_i *and* I_j. This is also true for a sequence of transitions regardless of the path between states i and j.[6]

---

[5] Proof that $\langle\Gamma\rangle$ is zero:
Combining $\langle\Gamma\rangle = \Sigma_{ij} P_{ij} \cdot \Gamma_{ij}$ and $\Gamma_{ij} = -\log P_{i\to j} + \log P_{i\leftarrow j}$ gives us $\langle\Gamma\rangle = \Sigma_{ij} P_{ij} \cdot (\log P_{i\leftarrow j} - \log P_{i\to j})$.
Since $P_{ij} = P_i \cdot P_{i\to j} = P_{i\leftarrow j} \cdot P_j$, $\langle\Gamma\rangle = \Sigma_{ij} P_{ij} \cdot (\log (P_{ij} / P_i) - \log (P_{ij} / P_j)) = \Sigma_{ij} P_{ij} \cdot (\log P_j - \log P_i)$.
Separating the double summation and swapping subscripts in the second term,
$\langle\Gamma\rangle = \Sigma_i \Sigma_j P_{ij} \cdot \log P_j - \Sigma_i \Sigma_j P_{ij} \cdot \log P_i = \Sigma_i \Sigma_j P_{ij} \cdot \log P_j - \Sigma_j \Sigma_i P_{ji} \cdot \log P_j$.
Finally, pulling out the j-summation gives us $\langle\Gamma\rangle = \Sigma_j \log P_j \cdot (\Sigma_i P_{ij} - \Sigma_i P_{ji})$.
But $\Sigma_i P_{ij} = \Sigma_i P_{ji} = P_j$ because the sum of transition probabilities must equal the state probability for both incoming and outgoing transitions; therefore, $\langle\Gamma\rangle = 0$.

[6] Proof that cumulative net information gain depends only on the endpoints, not on the path:
$\Gamma_{ik} = \Gamma_{ij} + \Gamma_{jk} = (I_j - I_i) + (I_k - I_j) = I_k - I_i$.

Therefore, if the system somehow transitions from a common state i (with low information content) to a rare state j (with high information content), it does so by gaining information from external stimuli. Similarly, transitioning from a rare state to a common state involves a loss of previously gained information. During a "round trip" from a common state to a rare state and back, external information is temporarily remembered then forgotten. We refer to such sequences as *memory loops* in state space.

To capture the fluctuation of state information we can compute the root-mean-square deviation of I, which we define as the complexity σ_I such that

$$\sigma_I{}^2 = <(I - <I>)^2>.$$

The symbol <> now represents the mean weighted by state probabilities. For example,

$$<I> = \Sigma_i P_i \cdot I_i$$

where the symbol Σ_i indicates a sum over all states. It is easy to verify that

$$<(I - <I>)^2> =<I^2> - <I>^2$$

and therefore

$$\sigma_I{}^2 = \Sigma_i P_i \cdot I_i{}^2 - (\Sigma_i P_i \cdot I_i)^2.$$

Finally,

$$\sigma_I{}^2 = \Sigma_i P_i \cdot log^2 P_i - (\Sigma_i P_i \cdot log P_i)^2.$$

5.3 Γ vs. I, σ_Γ vs. σ_I

Γ is defined as the net information gain of single transitions between directly connected pairs of states whereas I is defined as state information. We have shown that $\Gamma = \Delta I$, so I is like potential and Γ is like force. External information "pushes" a system "uphill" to a state of higher information potential to accomplish information storage, much like pushing a mass uphill to a state of higher gravitational potential stores energy. The amount of energy stored depends only on the final height, not the path up the hill. Likewise, the amount of information stored does not depend on the transition path between an initial and final state in state space.

σ_Γ is defined as the fluctuation of Γ whereas σ_I is defined as the fluctuation of I. Since σ_I accounts for the cumulative net information gain of sequences of transitions whereas σ_Γ only accounts for single transitions, σ_I can account for multi-transition memory loops in state space and should therefore be more indicative of the computational power of a system. Moreover, σ_I is easier to apply than σ_Γ because there can be many more transitions between states than actual states. Therefore, we take σ_I to be our primary *complexity measure* while bearing in mind that σ_Γ may also be a useful quantity in some cases.

6 Entropy

It is worth noting that $<I>$ is also known as entropy E:

$$E = <I> = - \Sigma_i P_i \cdot log P_i.$$

Since σ_I is defined as a fluctuation about $<I>$ we can also say that complexity is a fluctuation about entropy. And since both complexity and entropy tend to increase with system size, the dimensionless ratio σ_I/E or *relative complexity* may be useful for comparing systems of different sizes, with

$\sigma_I{}^2/E^2 = (\Sigma_i P_i \cdot \log^2 P_i / E^2) - 1.$

Entropy is a measure of *disorder* that can be linked to the notion of complication introduced in Section 1. The entropy of a system is maximum when all of its N states have the same probability $P_i = 1/N$ (as in the case of a television screen driven by white noise), in which case the entropy is the logarithm of the number of states N. The entropy is minimum when there is a single state with probability one, in which case the entropy is zero. At either extreme there can be no information fluctuation and therefore the complexity is zero, as intuitively expected.

7. Example: elementary cellular automaton rule 110

The elementary cellular automaton is a convenient and interesting system to study with our complexity measure. It consists of a line of adjacent cells each of which contains a single bit (0 or 1). The line may form a closed loop, be infinitely long or, as in our example, be open-ended with external inputs applied to the end cells. The next content of each cell depends on its present value and those of its immediate neighbors to the left and right. This dependency is specified by a rule such as rule 110 shown in Table 1.

3-cell group	111	110	101	100	011	010	001	000
Next center cell	0	1	1	0	1	1	1	0

Table 1: Elementary cellular automaton rule 110.

This rule is so named because the lower row as a binary number (01101110_2) is one hundred ten (110_{10}) as a decimal number. The upper row represents the contents of any 3-cell group along the line of cells and the lower row gives the next content of the center cell.

There are 2^8 or 256 possible rules. We have chosen rule 110 because it is known to be complex: it has been proven to be capable of universal computation. Essential to the proof is that this rule supports the existence of *gliders* as shown in Figure 5. Gliders are an example of emergent behavior associated with complex systems.

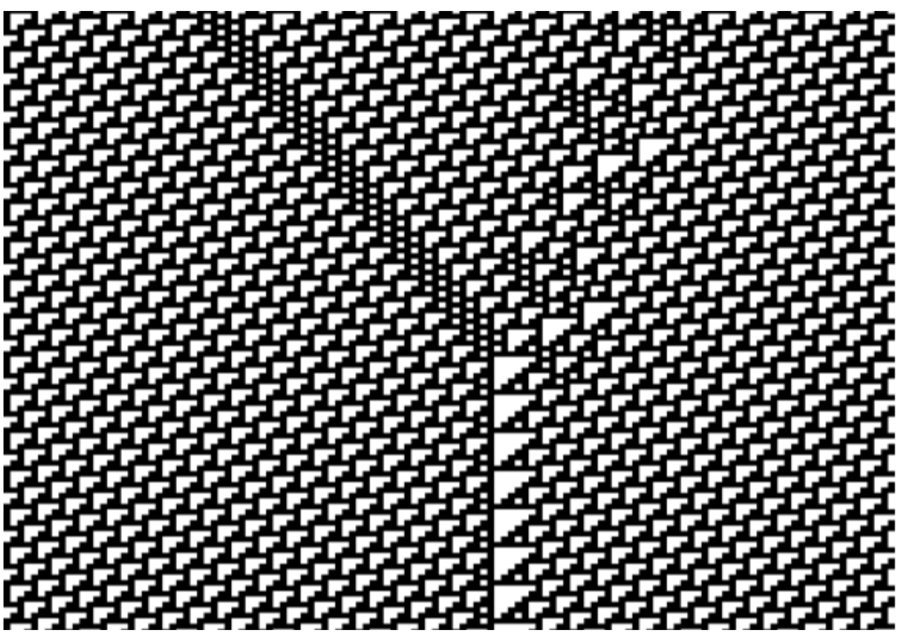

Figure 5: Rule 110—two gliders merging into a stationary object.

The pixels in each horizontal row of the image of Figure 5 represent the contents of the cells of the one-dimensional automaton at a moment in time. As time progresses downward, cell contents change according to the rule.

For coherent and self-perpetuating gliders to exist, small groups of adjacent cells must have the capacity to remember that a glider is passing through them for several generations. Memory is essential for higher levels of computation and for complexity. Furthermore, the glider interaction shown in Figure 5 can be interpreted as an elementary logical operation, bridging complexity theory and computation theory.

7.1 Analytical method

Now, let us compute the complexity of a small open-ended group of three adjacent cells each of which follows rule 110, as specified in Table 1. The table gives the next content of the center cell, but not that of the end cells. To determine the next contents of the end cells, we attach two driver cells, which we control, to the open ends of the 3-cell group:

driver → **end-center-end** ← driver.

The present state of our 3-cell **end-center-end** group can be represented as a 3-bit binary number or as its corresponding single octal digit, as in the state diagram of Figure 6.

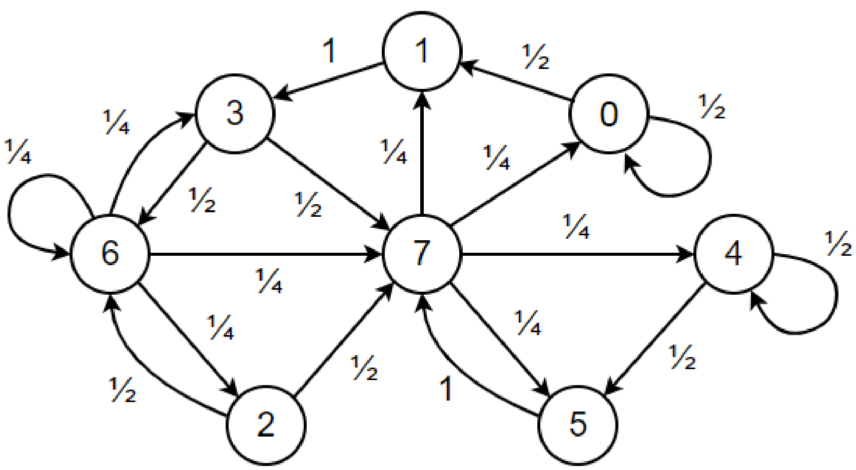

Figure 6: State diagram of three adjacent cells that obey rule 110, showing forward conditional transition probabilities.

For each state, we must determine the forward conditional probability of transitioning to each of the other states, given random external stimuli provided by the driver cells. For example, if the present state is 7 (**1–1–1**), we must determine the four next-states for each combination of driver cell contents:

0→**1–1–1**←0 → **1–0–1**,
0→**1–1–1**←1 → **1–0–0**,
1→**1–1–1**←0 → **0–0–1**,
1→**1–1–1**←1 → **0–0–0**.

Summarizing in octal form, 7 → 5,4,1,0. Accordingly, on the state diagram of Figure 6, four transition arrows exit state 7 going to states 5, 4, 1, 0. The driver cells are assumed to provide random stimulus so that each of the transition arrows is equally likely to be followed and therefore each is labeled with a forward conditional probability of ¼. Likewise, 6 → 6,7,2,3. Continuing: 5 → 7,7,7,7; 4 → 4,5,4,5; 3 → 7,6,7,6; 2 → 6,7,6,7; 1 → 3,3,3,3; and finally 0 → 0,1,0,1. States 0, 2, 3, and 4 have only two possible next states with equal probability ½. States 1 and 5 have only one possible next state with probability 1.

Now that we have determined the forward conditional probabilities, we can compute the state probabilities. They are related by

$$P_j = \Sigma_i P_i \cdot P_{i \to j}$$

and since all state probabilities must add to one,

$$\Sigma_i P_i = 1.$$

For the current example these equations are explicitly:

$$P_7 = ½ \cdot P_2 + ½ \cdot P_3 + P_5 + ¼ \cdot P_6$$
$$P_6 = ½ \cdot P_2 + ½ \cdot P_3 + ¼ \cdot P_6$$
$$P_5 = ½ \cdot P_4 + ¼ \cdot P_7$$
$$P_4 = ½ \cdot P_4 + ¼ \cdot P_7$$
$$P_3 = P_1 + ¼ \cdot P_6$$
$$P_2 = ¼ \cdot P_6$$
$$P_1 = ½ \cdot P_0 + ¼ \cdot P_7$$
$$P_0 = ½ \cdot P_0 + ¼ \cdot P_7$$
$$P_7 + P_6 + P_5 + P_4 + P_3 + P_2 + P_1 + P_0 = 1$$

They can be solved manually or with the aid of a computer program for solving linear algebraic equations. The solution is given in Table 2.

P_7	P_6	P_5	P_4	P_3	P_2	P_1	P_0
$\frac{4}{17}$	$\frac{2}{17}$	$\frac{2}{17}$	$\frac{2}{17}$	$\frac{5}{34}$	$\frac{1}{34}$	$\frac{2}{17}$	$\frac{2}{17}$

Table 2: State probabilities of three adjacent cells governed by rule 110.

Now we can compute the entropy, the complexity and the relative complexity:

$E = -\Sigma_i P_i \log_2 P_i$ = 2.86 bits,

$\sigma_I = \sqrt{\Sigma_i P_i \log_2^2 P_i - E^2}$ = 0.56 bits,

σ_I / E = 0.20.

Note that the maximum possible entropy for an eight-state system is 3 bits, which is the case when $P_i = ⅛$ for all i. Clearly then, rule 110 has a relatively high entropy or state utilization of 2.86 bits. However, this does not preclude a considerable fluctuation of information about the entropy and thus a considerable value of the complexity. Whereas, maximum entropy *would* preclude complexity.

7.2 Empirical method

One can imagine that the above computation might become unwieldy for much larger groups of adjacent cells than merely three. In such cases where an exact analytical solution is impractical an empirical method can be employed to approximate the state probabilities.

Simply drive the system at its inputs (the driver cells in the case of an elementary cellular automaton) with a random source and observe the probability of the system states over a long period of time. For example, some results of computer simulations of the rule 110 automaton for larger numbers of cells are shown in Table 3.

No. of cells	3	4	5	6	7	8	9	10	11	12	13
E **(bits)**	2.86	3.81	4.73	5.66	6.56	7.47	8.35	9.25	10.09	10.97	11.78
$\boldsymbol{\sigma_I}$ **(bits)**	0.56	0.65	0.72	0.73	0.79	0.81	0.89	0.90	1.00	1.01	1.15
$\boldsymbol{\sigma_I/E}$	0.20	0.17	0.15	0.13	0.12	0.11	0.11	0.10	0.10	0.09	0.10

Table 3: Results of rule 110 computer simulations.

We see that the relative complexity σ_I/E levels off to about 0.10 by 10 cells.

7.3. Technical notes

7.3.1. Figure 5: image of a rule 110 glider interaction

The software used to create this image is available on github.com.

This rule 110 automaton actually forms a closed loop: the leftmost cell is directly connected to the rightmost cell.

7.3.2. Tables 2 and 3: complexity computations

It is important to note that the subject of this example, the rule 110 variant of the elementary cellular automaton, has only one attractor in its state space – an attractor being an isolated subset of states that are only interconnected among themselves, in the steady-state, not including transient states. The attractor for rule 110 consists of eight states, the most possible for 3-cell automata, with none of them being transient states. Transient states begin to appear when the number of cells increases to 5; but at least up to 19 cells, there is only one attractor.

Some other rule variants have multiple attractors. For these rules, the complexity can be determined separately for each attractor; alternatively, representative complexity values could be determined by randomly varying the initial state of many simulations and averaging the results.

The software used to compute the results contained in Tables 2 and 3 is available on github.com. It can be used to study any of the rule variants and includes functions: to find all attractors in state space, to compute state probabilities for each attractor either by solving a set of linear equations such as those given in Section 7.1 or by simulating the automaton with random stimulation for many generations to approximate the probabilities as in Section 7.2, and to compute entropy and complexity from these state probabilities.

8. Applications

Of what use is our complexity measure? We address this question further below but it is already clear that it is useful in the study of the elementary cellular automaton. In the paper by Bates and Shepard, the complexity σ_I is computed for all elementary cellular automaton rules and it was observed that rules that exhibit slow-moving gliders and possibly stationary objects, as rule 110 does (see Figure 6), are highly correlated with large values of σ_I, as one might expect. σ_I could therefore be used as a filter for the selection of candidate rules for universal computation, which is challenging to prove.

Although our complexity measure is based on information fluctuations in a dynamic system, its definition is only given in terms of state probabilities and could therefore be made to apply to any probability distribution, including those derived from static images or text.

During the three decades since the publication of the aforementioned paper, it has been referred to by researchers in many diverse fields: complexity theory, complex systems science, complex networks, chaotic dynamics, many-body localization entanglement, environmental engineering, ecological complexity, ecological time-series analysis, ecosystem sustainability, air and water pollution, hydrological wavelet analysis, soil water flow, soil moisture, headwater runoff, groundwater depth, air traffic control, flow patterns and flood events, topology, economics, market forecasting of metal and electricity prices, health informatics, human cognition, human gait kinematics, neurology, EEG analysis, education, investing, artificial life and aesthetics.

The last item related to aesthetics is particularly intriguing as it suggests that it might be possible to understand even such abstract concepts as beauty and elegance as a dynamic balance of chaos and order, and furthermore to express such notions with a relatively simple mathematical formula.

9. A fundamental and universal concept

Fluctuations of order and chaos, loss and gain of information, stability and instability—are the essential components of complexity, as distinct from complication. Without chaos, there would be no evolution or adaptation; without order, there would be no persistence or resilience.

Wide-ranging applications of our information fluctuation complexity measure are to be expected considering the fundamental and universal understanding it is based on, an understanding of the dynamic balance of opposites so elegantly depicted by the ancients in the yin-yang symbol of Figure 7.

Figure 7: Yin-yang—dynamic balance of opposites.