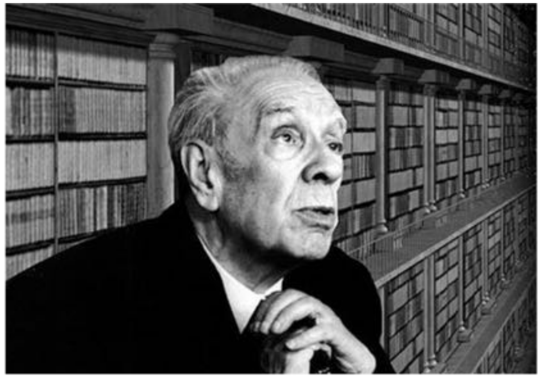
WHY THE WORLD IS SIMPLE

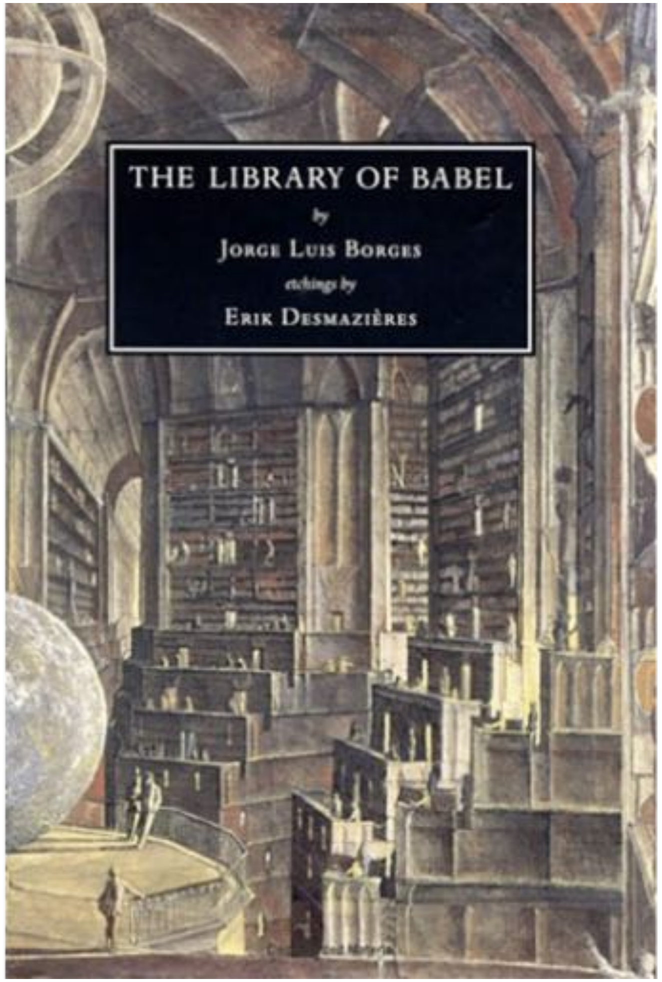
Ard Louis



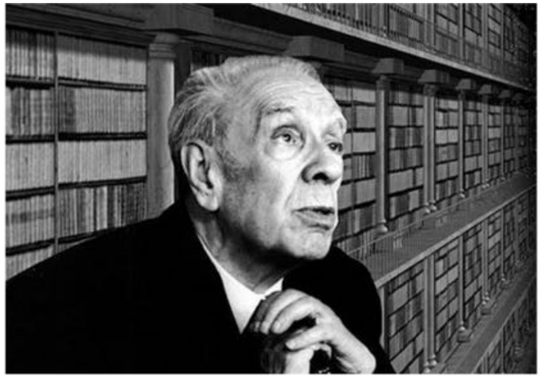
Borges' Library

- Each book is made up of 410 pages, each with 40 lines, each lines of 80 characters, using 22 letters of the alphabet, along with the period, space, and comma, for a total of 25 characters.

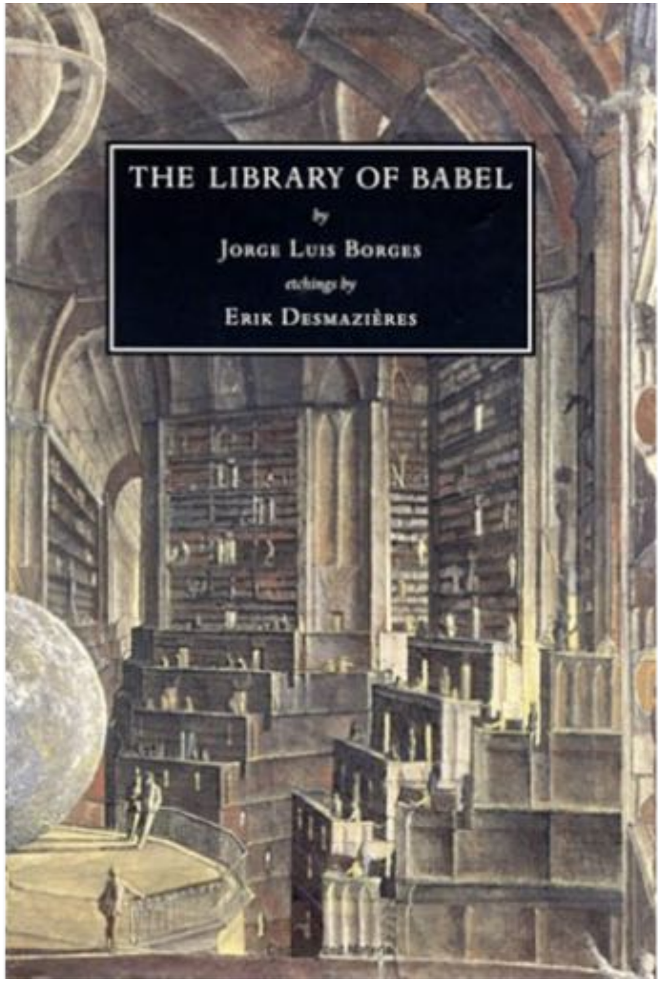
- **Every book is equally likely**



40X80X410=1,312,000 characters per book. – $10^{1,834,097}$ possible books



Borges' Library



- Each book is made up of 410 pages, each with 40 lines, each lines of 80 characters, using 22 letters of the alphabet, along with the period, space, and comma, for a total of 25 characters.

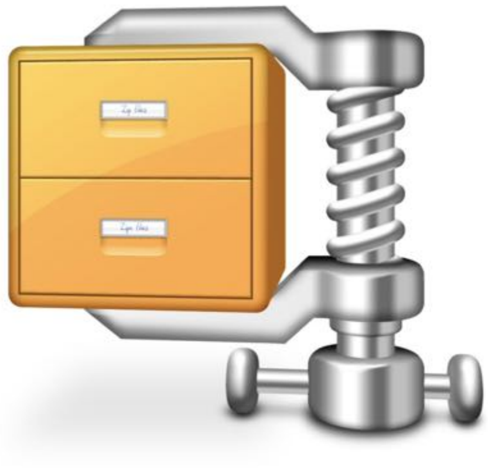
- **Every book is equally likely**

- **But, is every story equally likely?**

For fun: https://libraryofbabel.info/

$40 \times 80 \times 410 = 1{,}312{,}000$ characters per book. – $10^{1,834,097}$ possible books

Most strings are not compressible



number of binary strings

n=1: 0, 1
n=2: 00, 01, 10, 11
n=3: 000,001,010,011,100,101,110,111
2^n strings of length n
2^n-2 strings shorter than n

½ of strings compressed by 1 bit
¼ of strings compressed by 2 bits
$1/2^p$ " " " p bits

Examples:
about 1/1000t can be compressed by 10 bits
1/1,000,000 " " by 20 bits
etc…..

Highly compressible strings are extremely rare

Question: If most sequences are incompressible, why are most patterns we see in nature highly compressible?

AN INTUITION:

What is the probability that a monkey types out X digits of π on an N key typewriter ?

P(X) = (1/N)^(X+1)

3.14159265358979323846264338327950288419716939
93751058209749445923078164062862089986280348 2
53421170679821480865132823066470938446095505 8
22317253594081284811174502841027019385211055 5
964462294895493038196442

But what if the monkey types into C ?

P(X) $\lesssim$ (1/N)^133

133 character (obfuscated) C code to calculate first 15,000 digits of π

```
a[52514],b,c=52514,d,e,f=1e4,g,h;
main(){for(;b=c-=14;h=printf("%04d", e+d/f))
for(e=d%=f;g=--b*2;d/=g)d=d*b+f*(h?a[b]:f/5),a[b]=d%--g;}
```

$$\pi = \sum_{i=0}^{\infty} \frac{(i!)^2 2^{i+1}}{(2i+1)!}$$

C program due to Dik Winter and Achim Flammenkamp (See Unbounded Spigot Algorithms for the Digits of Pi, by Jeremy Gibbons (Oxford CS), Math. Monthly, April 2006, pages 318-328.)

Making monkey intuitions quantitative: Universal Turing Machines

Alan Turing
1912-1954

Universal Turing machine (UTM) can simulate anything that is computable. FAPP: C, Fortran, Pascal etc… are Turing Complete, with infinite resources they can be UTMs

Church-Turing thesis: a function on the natural numbers is computable if and only if it is computable by a Turing machine.

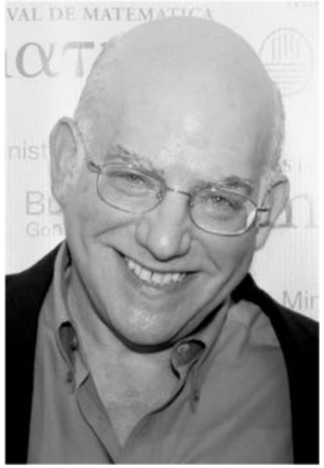
Halting Problem: There is no general algorithm that can always determine whether a program on a UTM will halt, or keep going on forever.

Entscheidungsproblem: David Hilbert's "decision" problem: Is mathematics decidable: Can an algorithm decide whether any given statement is provable from a fixed set axioms using the rules of logic? Godel & then Turing & Alonso Church proved that the answer is no.

*Turing, A.M. . "On Computable Numbers, with an Application to the Entscheidungs problem". Proceedings of the London Mathematical Society. 2 (1937) **42**: 230–265.*

Compression & Kolmogorov complexity of a single object

A.N. Kolgomorov
1903-1987



G.J. Chaitin
1947--

Kolmogorov complexity K(X) is the length in bits of the shortest program on a UTM that generates X

K is universal, (not UTM dependent) because you can always write a compiler . For example, if U and W are UTMs, then

K_U(X) = K_W(X) + O(1) ≈ K(X) asymptotically

K is not computable due to Halting problem.

simple

01010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101010101

complex

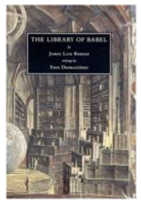
0111010100110010111101111011010100001000101110101010011010111110111010010100011101110110110111010

Warning: you don't know for sure that it is complex, t could be encoding π= 3.14159265358979323846 2 …..

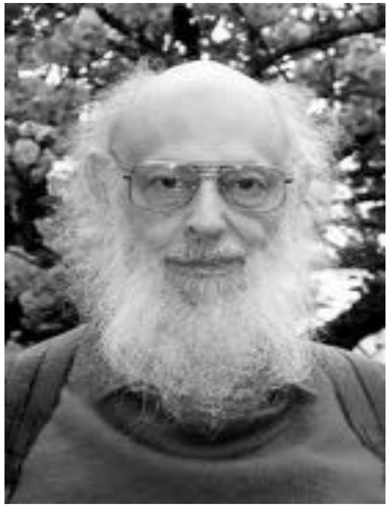
new intuitions from AIT

-- A random number is one for which K(X) ≳ |X|

-- The complexity of a set can be << than complexity of elements of the set
for example, Borges' library is very simple, even if the book describing your life is not.



Making the monkey intuition quantitative with AIT: Algorithmic probability



R. Solomonoff
1926-2009

$$P_U(X) = \sum_{l:U(l)=X} 2^{-l} = 2^{-K(X)} + \ldots.$$

First term is the biggest one

Sum all binary codes that generate X

Note: Solomonoff was heavily influenced by Carnap's program on induction

Intuitively: simpler (small K(X)) outputs are much more likely to appear

Solomonoff, R., "A Preliminary Report on a General Theory of Inductive Inference", Report V-131, Zator Co., Cambridge, Ma. Feb 4, 1960, revision, Nov., 1960.

Making the monkey intuition quantitative with AIT:

The coding theorem

L. Levin, 1948 --

$$2^{-K(x)} \leq P(x) \leq 2^{-K(x)+O(1)}$$

We should teach this much more widely!

Intuitively: simpler (small K(X)) outputs are much more likely to appear

L. A. Levin. Laws of information conservation (non-growth) and aspects of the foundation of probability theory. Problems of Information Transmission, 10:206–210, 1974.

Making the monkey intuition quantitative with AIT: The coding theorem

$$2^{-K(x)} \leq P(x) \leq 2^{-K(x)+O(1)}$$

We should teach this much more widely!

Serious problems for applying coding theorem more widely

1) Many systems are not Universal Turing Machines
2) Kolmogorov complexity K(x) is formally incomputable
3) Many systems not in the asymptotic limit , O(1) terms…

Coding theorem for non-universal maps

Kamal Dingle

(2 Dphils of work)

$$\boxed{P(x) \lesssim 2^{-a\tilde{K}(x)-b}}$$

NOTE: upper bound only!

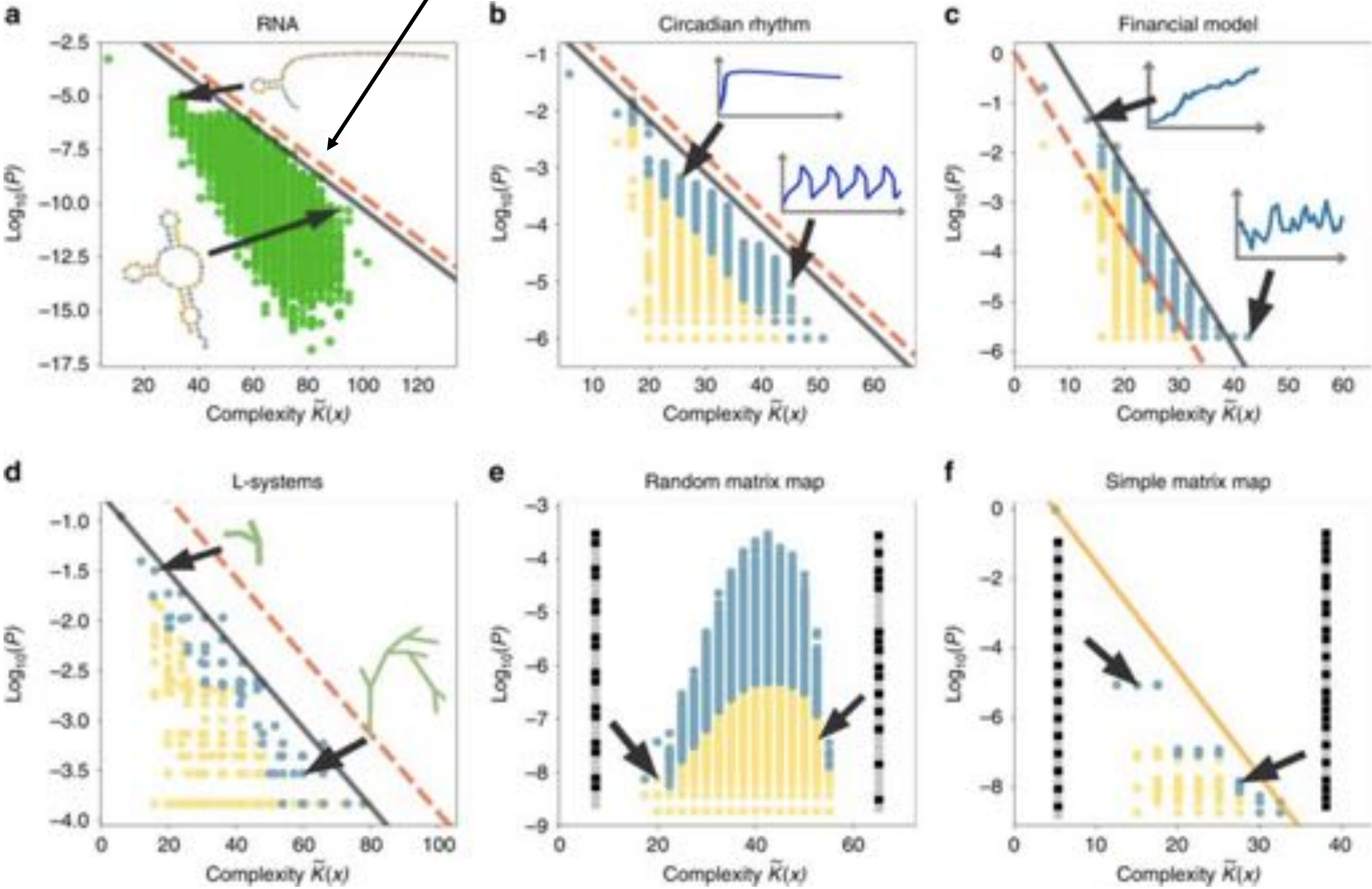
Chico Camargo

1) Computable input-output map **f**: I → O
2) Map **f** must be simple – e.g. K(f) grows slowly with system size
3) K(x) is approximated, for example by Lempel Ziv compression or some other suitable measure
4) Constants a and b depend on mapping only and can be approximated fairly easily.
5) Bound is tight for most inputs, but not most outputs.
6) Maps must be a) simple, b) redundant, c) non-linear, d) well-behaved (e.g. not a pseudorandom number generator)

K. Dingle, C. Camargo and A.A.L, Nature Communications 9, 761 (2018)

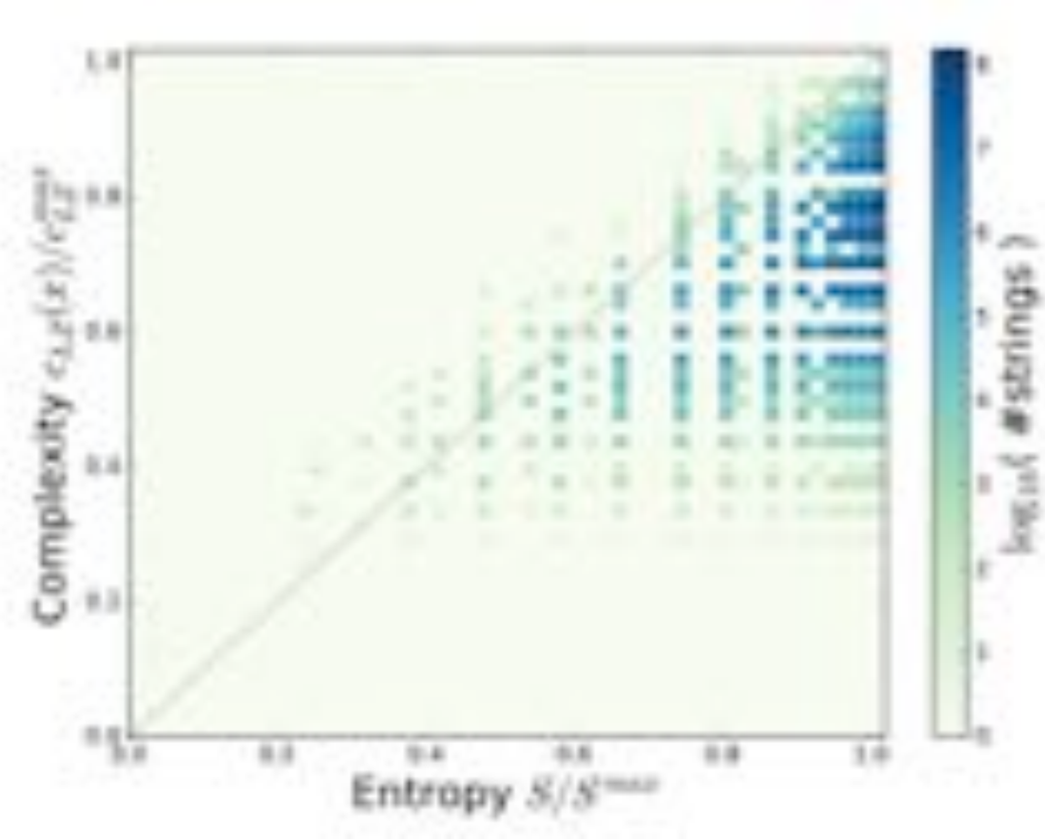
$$P(x) \lesssim 2^{-a\tilde{K}(x)-b}$$

Is black line (red dashed with b=0)



K. Dingle, C. Camargo and A.A.L, Nature Communications 9, 761 (2018)

Entropy versus KZ complexity



LZ Complexity v.s. Entropy S = p log p + (1-p) log (1-p) for binary strings of length 30

Question: most strings are close to maximally complex, Why do we see many compressible sequences in nature?

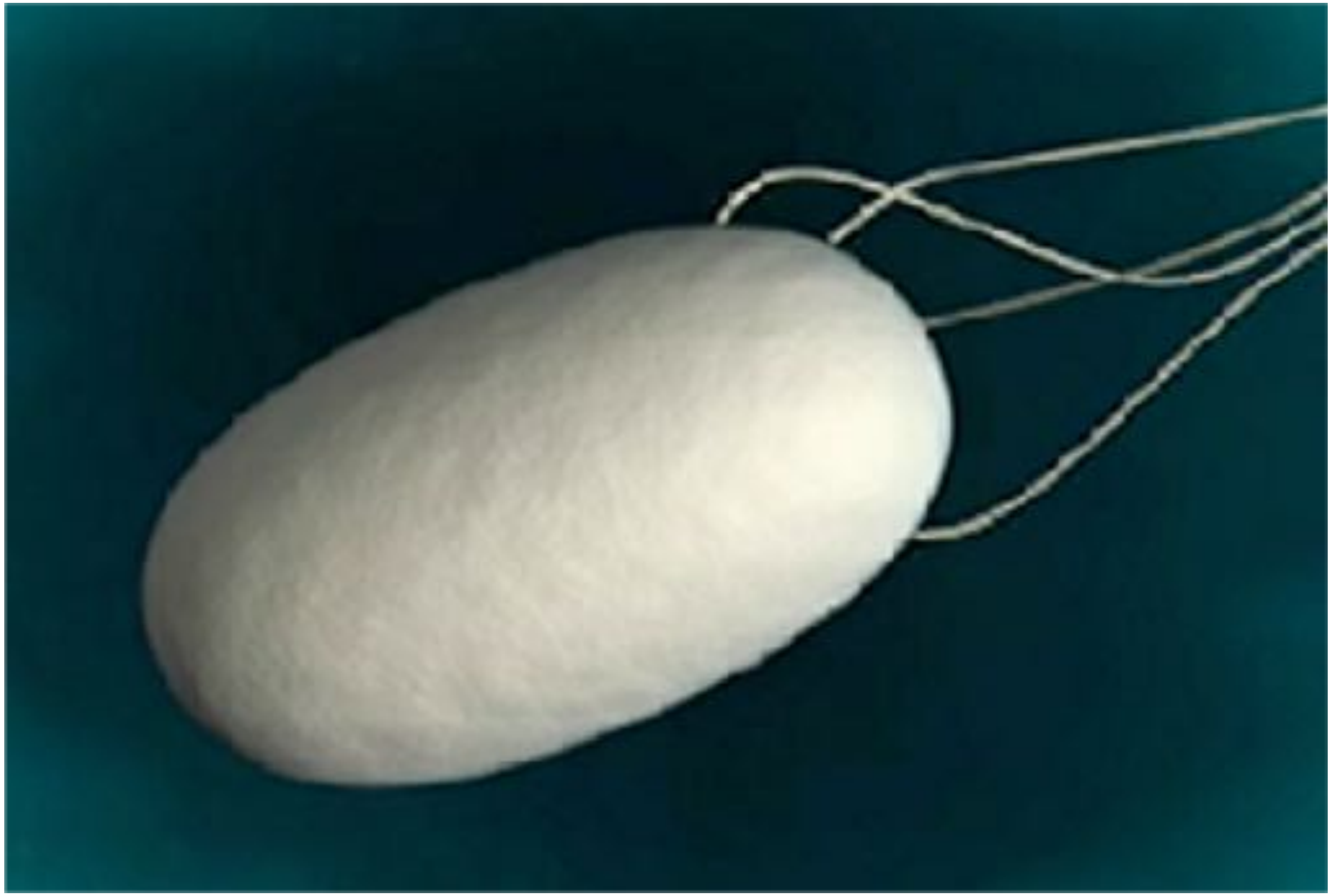
Potential answer: -- If patterns are caused by sampling algorithms then they will be exponentially biased towards low complexity outputs.

Applications of new coding theorem:

1. Evolution (the arrival of variation is biased towards simple phenotypes)

2. Machine learning with deep neural networks (biased towards simple functions)

Protein quaternary structure is self-assembled

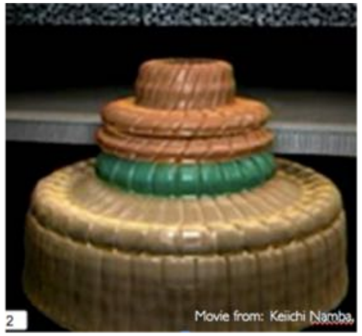


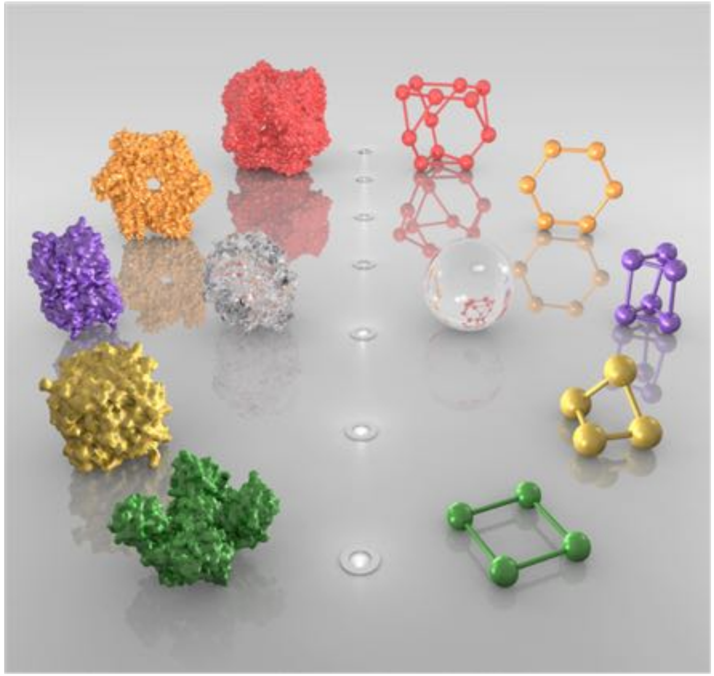
Self-assembly – can we understand, can we emulate?

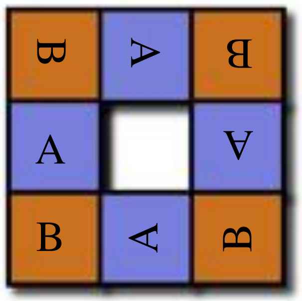
-- how does evolution design self-assembling structures?

Physicists should like evolution .. It is the fundamental law of biology

Self-assembling protein quaternary structure



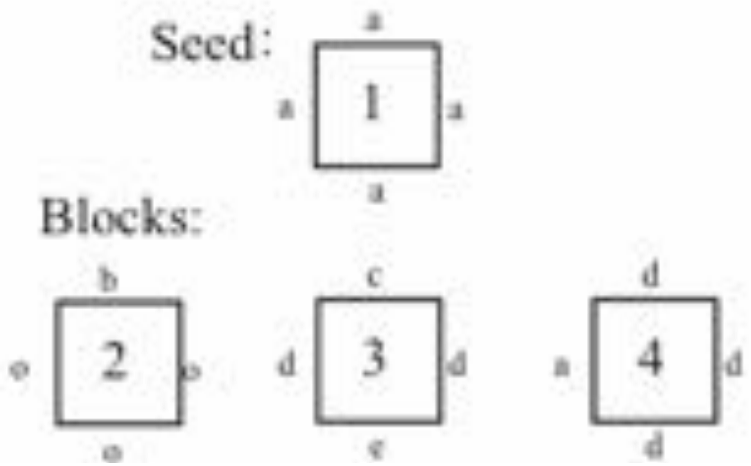




Polyomino model:
Self-assembling squares

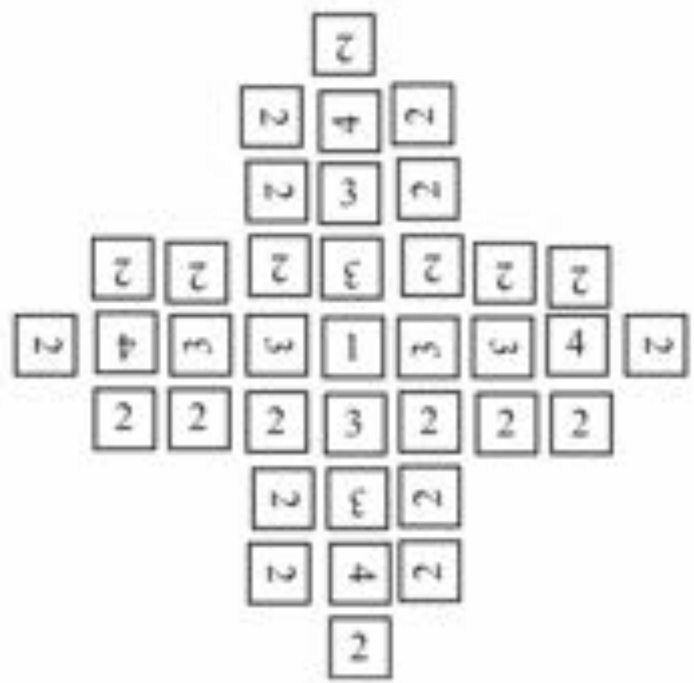
S. E. Ahnert, et al., Phys. Rev. E **82,** 02617 (2010), Iain G. Johnston, Phys. Rev. E. **83**, 066105 (2011)

Polyominoes: simplified self-assembly model



Rules:

	a	b	c	d	e	o
a	0	0	1	0	0	0
b	0	0	0	1	0	0
c	1	0	0	0	0	0
d	0	1	0	0	0	0
e	0	0	0	0	1	0
o	0	0	0	0	0	0



Deterministic self-assembly:
Always the same shape no matter what order you put blocks down

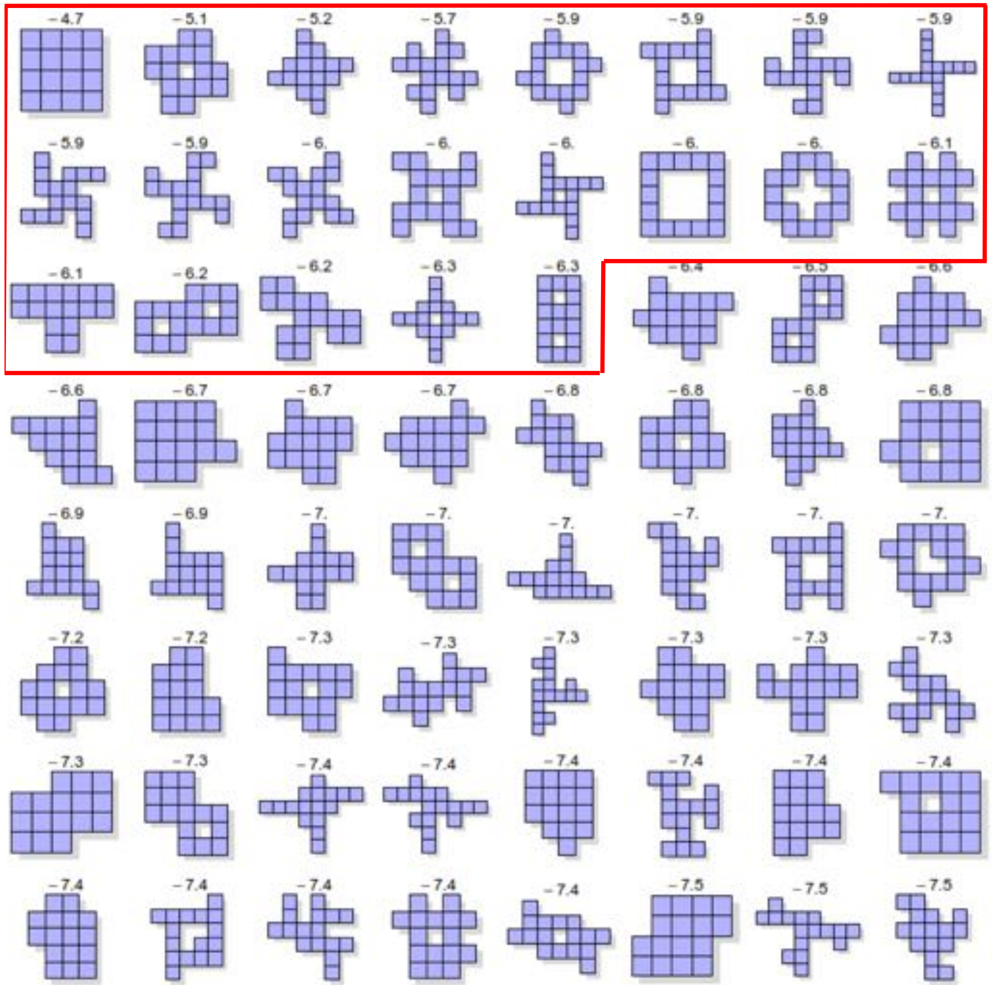
S. E. Ahnert, et al., Phys. Rev. E **82,** 02617 (2010), Iain G. Johnston, Phys. Rev. E. **83**, 066105 (2011)

Evolve to find rules to make 16-ominoes

There are 13,079,255 16-ominoes

Output of evolutionary runs:

Output is highly biased:
21 shapes =50% of genotypes

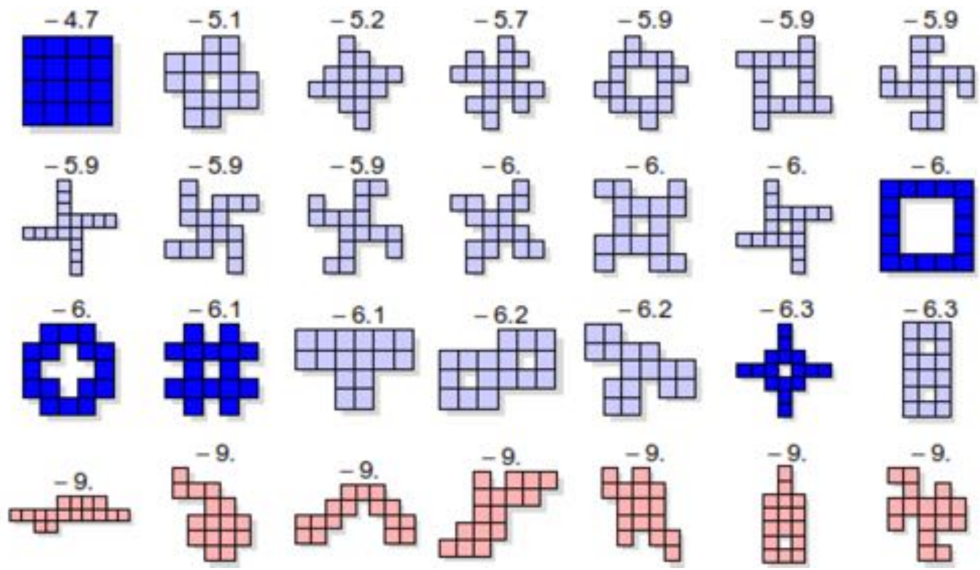


Iain Johnston

Symmetry spontaneously emerges from algorithmic nature of evolution

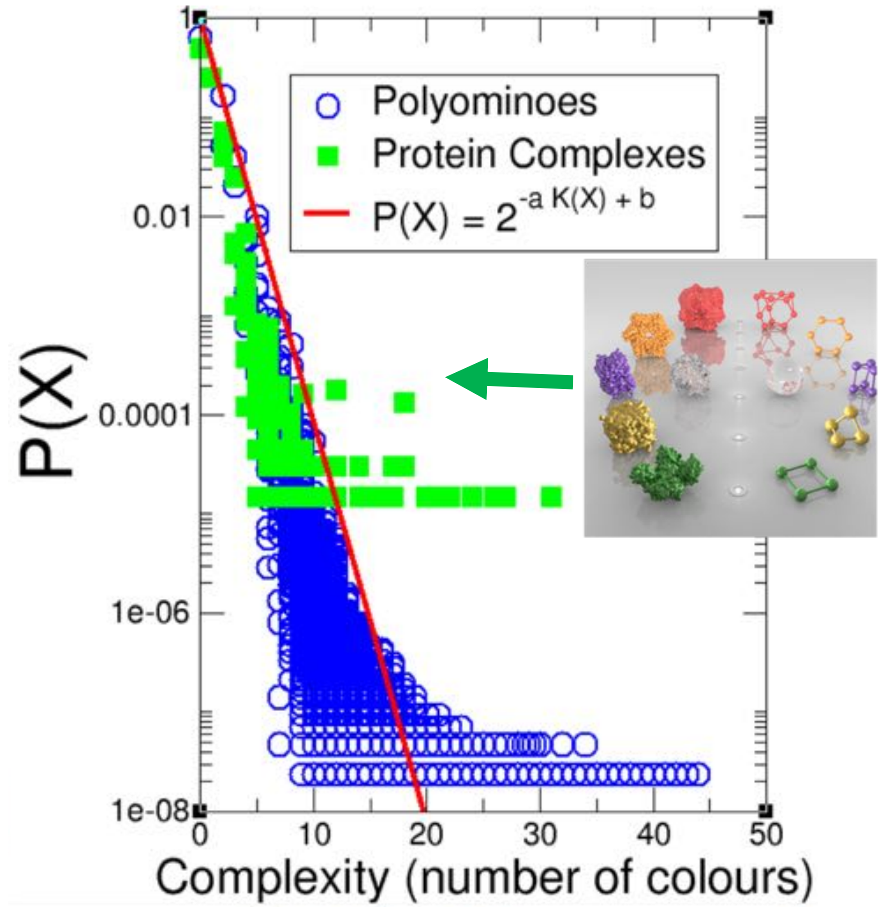
There are 13,079,255 16-ominoes

Output of 10^9 evolutionary runs:



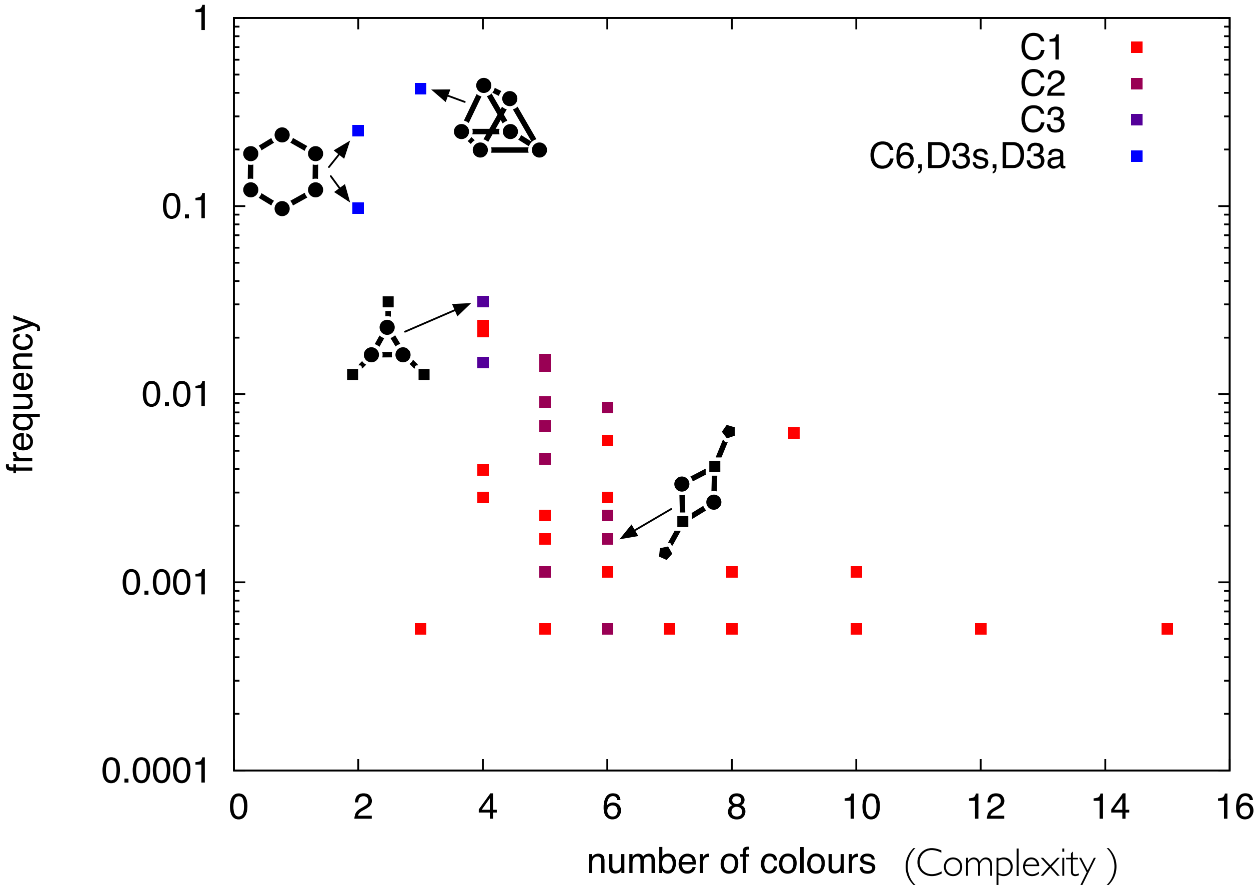
Only 5 D_4 symmetry 16-ominoes,
But they take up 35% of all genotypes

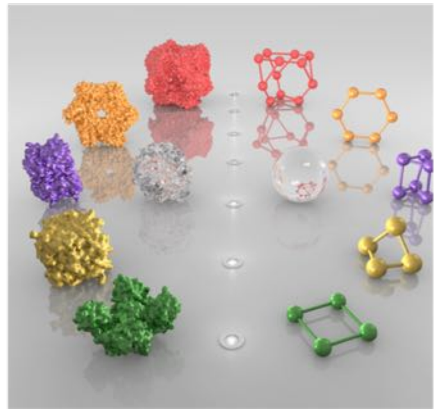
$$\boxed{P(x) \lesssim 2^{-aK(x)-b}}$$



Complexity measured as minimum information
Is needed to specify the assembling structure

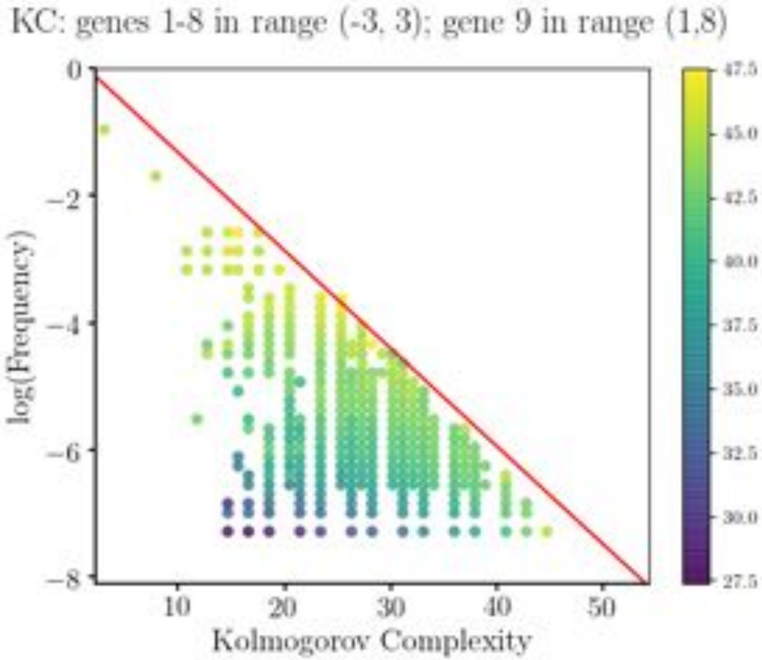
Symmetry of protein complex six-mers





Richard Dawkins' Biomorphs

Nora Martin



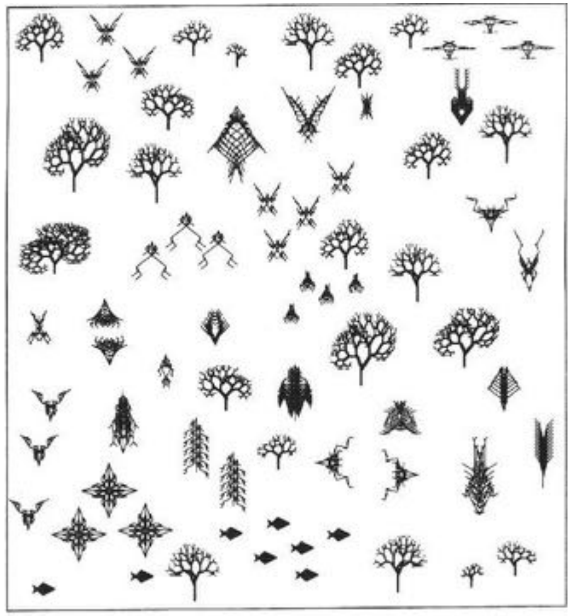
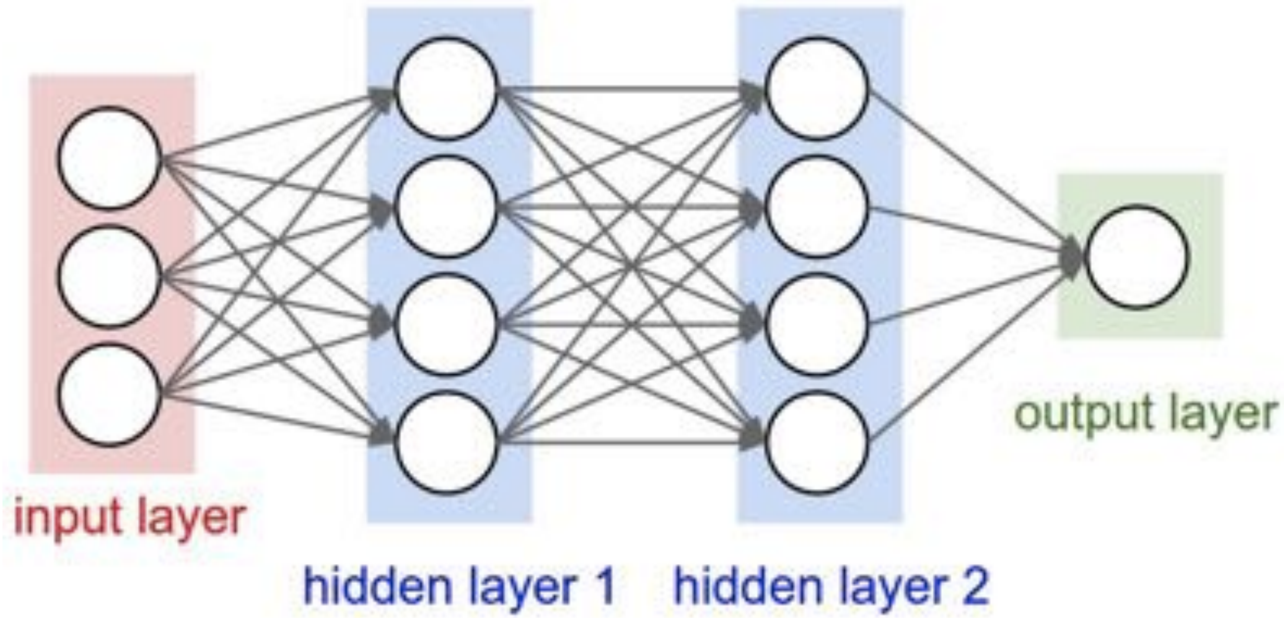


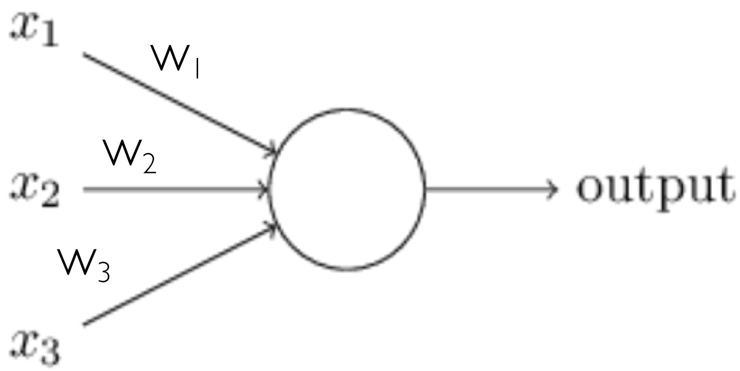
Figure 1.16 Safari park of black-and-white biomorphs, bred with the 'Blind Watchmaker' computer program.

Applications of new coding theorem:

1. Evolution (the arrival of variation is highly biased)
2. Machine learning with deep neural networks (biased towards simple functions)

Parameter-function map for deep learning

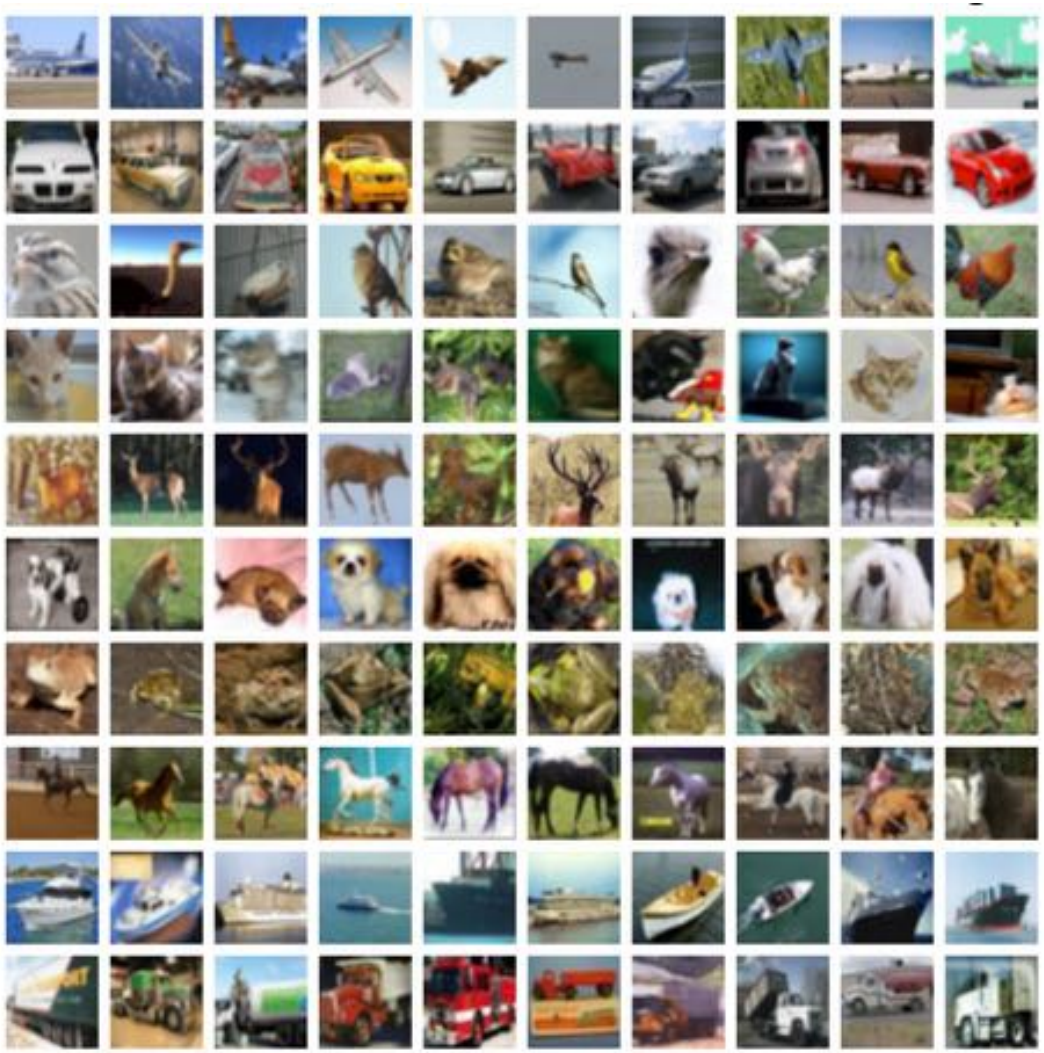




$$\text{output} = \begin{cases} 0 & \text{if } \sum_j w_j x_j \leq \text{ threshold} \\ 1 & \text{if } \sum_j w_j x_j > \text{ threshold} \end{cases}$$

f_θ is the function produced by the network with parameters θ.

Deep Neural Networks excel at pattern recognition



CIFAR-10 dataset

1) Train on a training set to fix the parameters
2) Test on a test set to see how well you predict unseen data
3) How well you do on unseen data is called generalization

Neural networks are highly over-parameterized: number of parameters >> number of data points

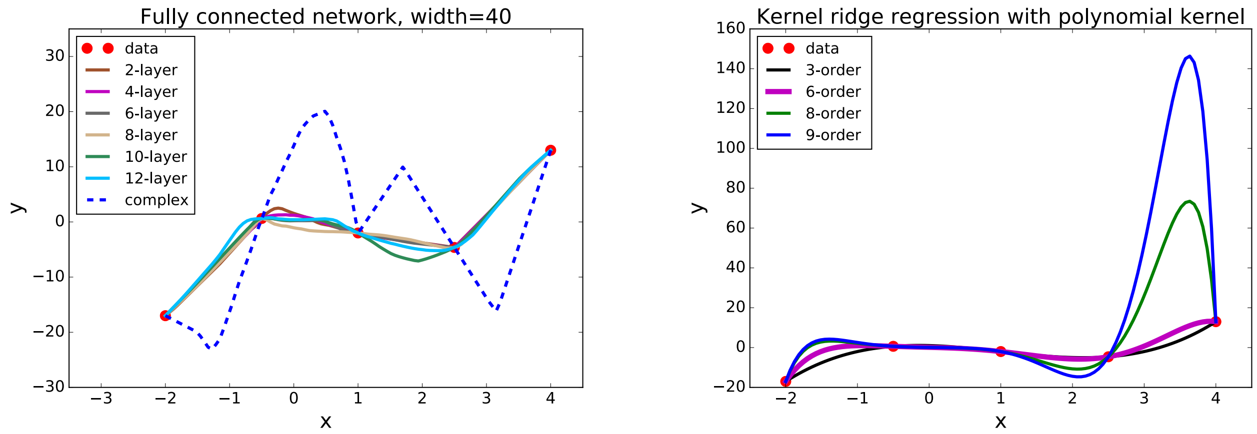


Figure 1: **(Left),** fitting results for 5 data points using FNNs with different number of layers; the overfitting solution with a high complexity (in dashed line) is intentionally constructed. **(Right),** fitting results by kernel regression with different orders of polynomial kernels.

Wu et al, arXiv:1706.10239

Understanding deep learning requires rethinking generalization, Zhang et al, arXiv:1611.03530 (2016)

Randomizing labels still leads to zero training error for a DNN , but no generalization …

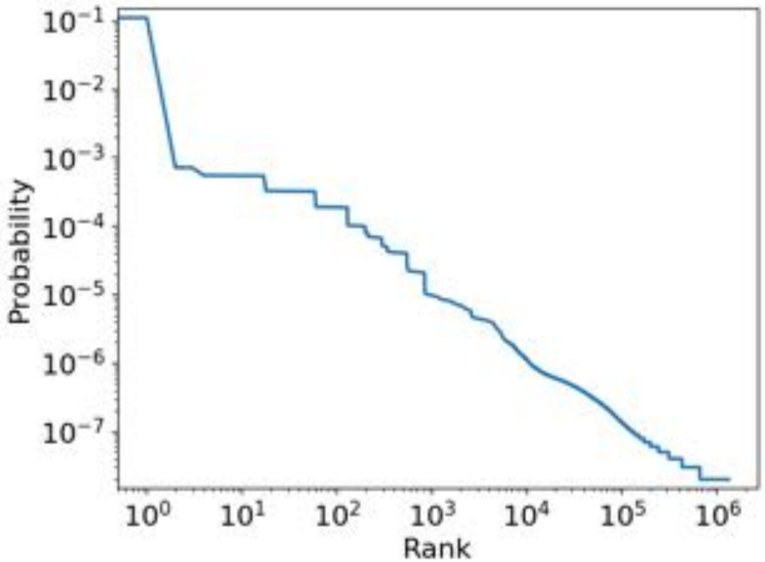
-- DNNs are very expressive (they can fit almost anything)

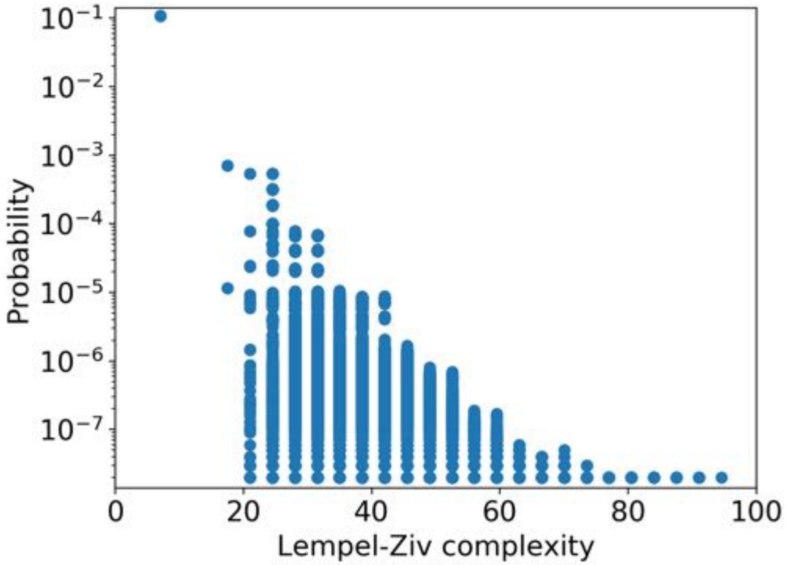
AI researchers allege that machine learning is alchemy
M Hutson - Science, 2018

DNN parameter-function map: If we randomly sample parameters θ, how likely are we to produce a particular function f?

Model problem for a 7 bit string, study all Boolean functions f.
There are $2^7 = 128$ different strings, and $2^{128} \simeq 10^{34}$ different functions.
You might expect a $1/10^{34}$ chance of finding any function.
Instead, we find strong simplicity bias.

Guillermo
Valle Perez

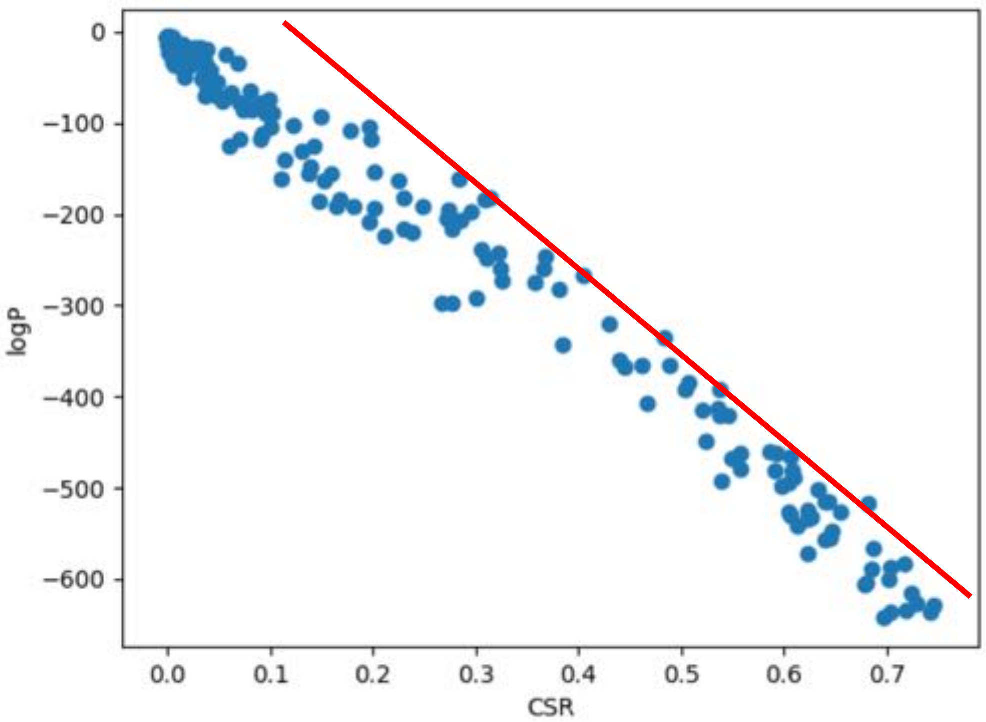


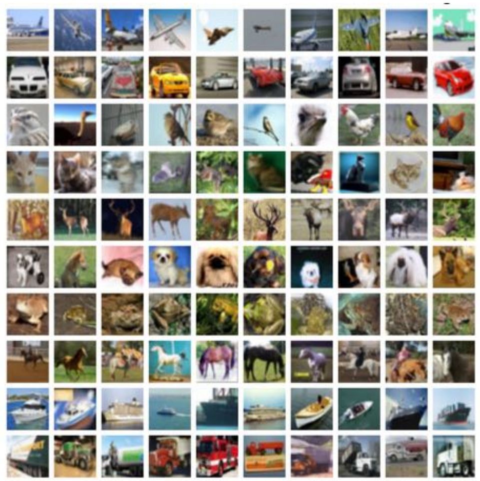


10^8 samples of parameters for (7,40,40,1) vanilla fully connected DNN system.

G. Valle-Perez, C. Camargo and A.A. Louis, arxiv:1805.08522

Probability v.s. complexity for a large image database called Cifar





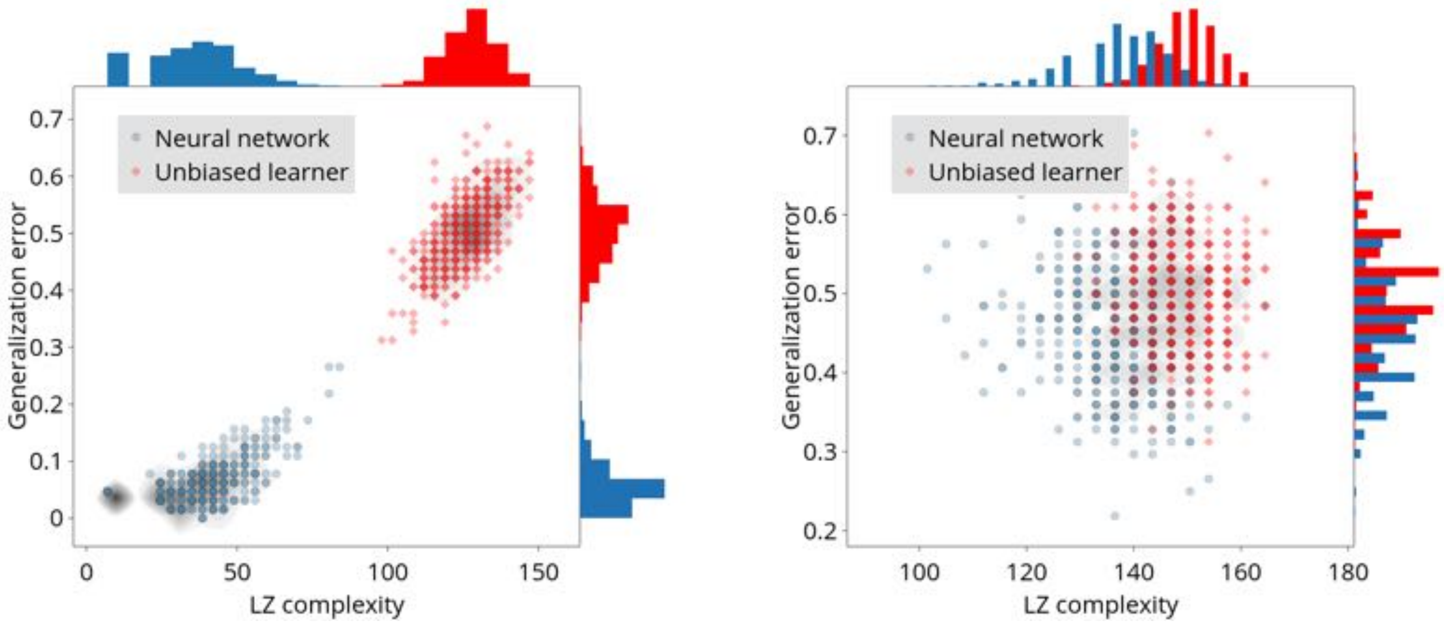
CIFAR

Sent to me this morning at 2:00 am.

Guillermo
Valle Perez

Test of generalization under supervised learning;

Training set = 64 strings – test set is other 64 strings

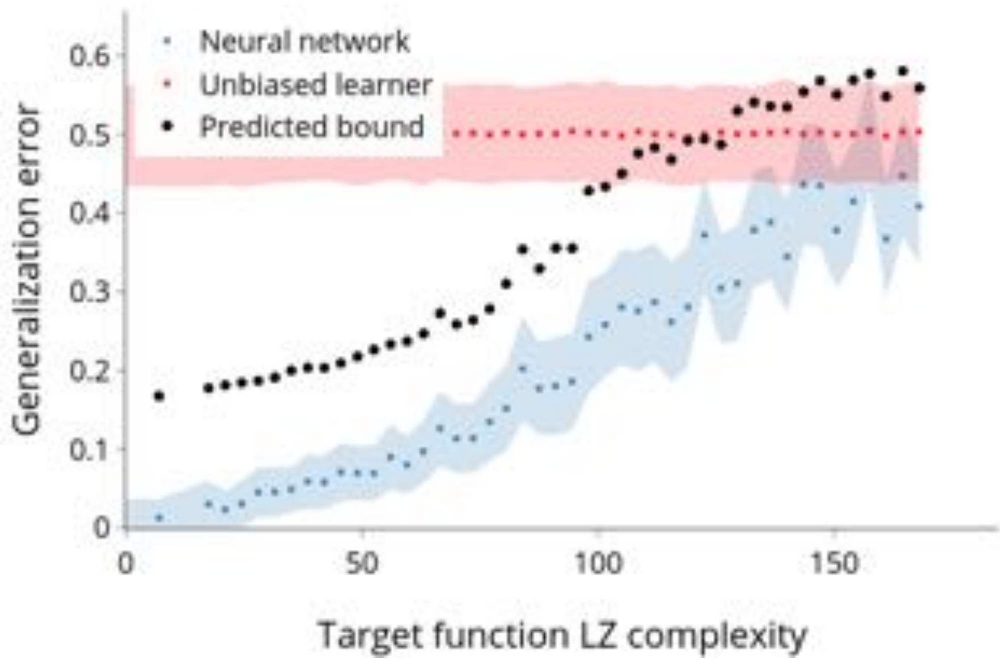


(a) Target function LZ complexity: 38.5 (b) Target function LZ complexity: 164.5

DNN does a much better job learning simpler functions than on complex functions

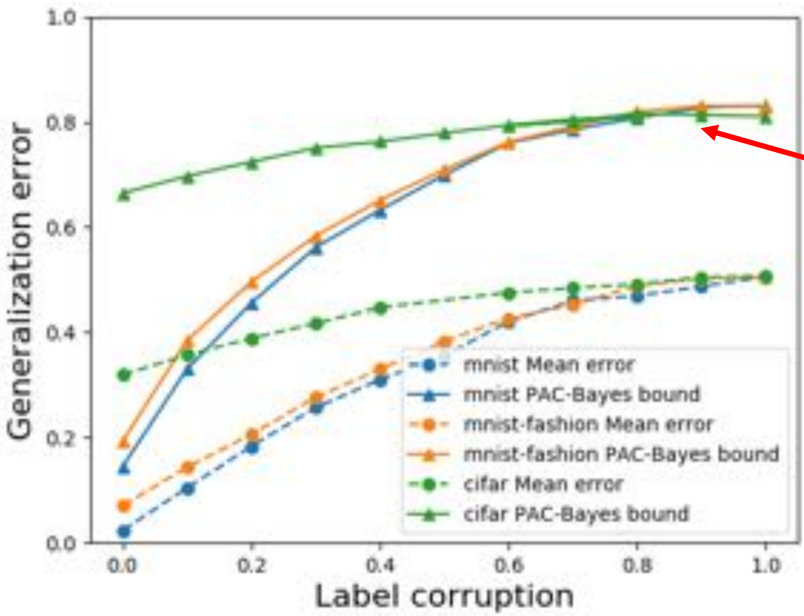
G. Valle-Perez, C. Camargo and A.A. Louis, arxiv:1805.08522

Generalization performance gets worse for more complex functions



G. Valle-Perez, C. Camargo and A.A. Louis, arxiv:1805.08522

Using bias to calculate PAC-Bayes bounds for CIFAR and MNIST



(a) for a 4 hidden layers convolutional network

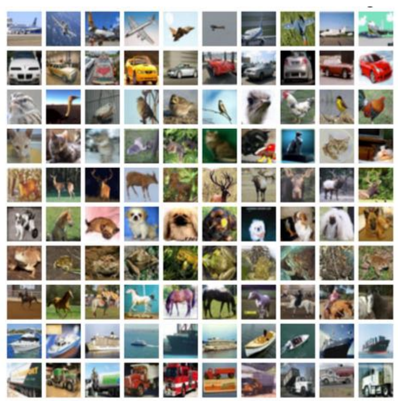
No longer alchemy?

Theorem 1. *(PAC-Bayes theorem [32]) For any measure P on any concept space and any measure on a space of instances we have, for $0 < \delta \leq 1$, that with probability at least $1-\delta$ over the choice of sample of m instances all measurable subsets U of the concepts such that every element of U is consistent with the sample and with $P(U) > 0$ satisfies the following:*

$$\epsilon(U) \leq \frac{\ln \frac{1}{P(U)} + \ln \frac{1}{\delta} + 2\ln m + 1}{m}$$

where $P(U) = \sum_{c \in U} P(c)$, and where $\epsilon(U) := E_{c\in U}\epsilon(c)$, i.e. the expected value of the generalization errors over concepts c in U with probability given by the posterior $\frac{P(c)}{P(U)}$. Here, $\epsilon(c)$ is the generalization error (probability of the concept c disagreeing with the target concept, when sampling inputs).

MNIST



Cifar

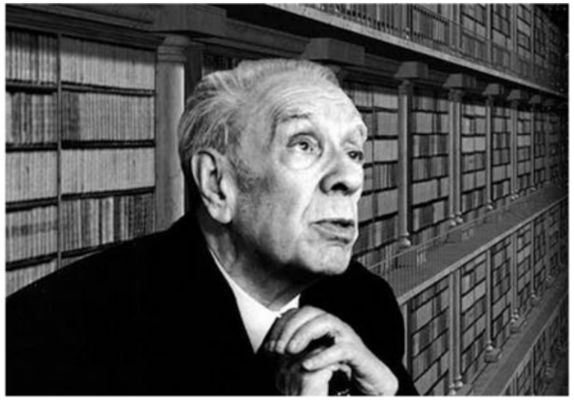
G. Valle-Perez, C. Camargo and A.A. Louis, arxiv:1805.08522

An algorithmic view of the world: Why is the world simple?

Algorithmic information theory, via the coding theorem, implies an exponential bias towards simplicity

Possibility spaces are not searched uniformly.

This may explain Occam's razor.



Every book is equally likely,
but every story is not