PLOS | ONE

RESEARCH ARTICLE

Bayesian validation of grammar productions for the language of thought

Sergio Romano[1,2]*, **Alejo Salles**[3], **Marie Amalric**[4], **Stanislas Dehaene**[4], **Mariano Sigman**[5], **Santiago Figueira**[1,2]

1 Universidad de Buenos Aires. Facultad de Ciencias Exactas y Naturales. Departamento de Computación. Buenos Aires, Argentina, **2** CONICET-Universidad de Buenos Aires. Instituto de Investigación en Ciencias de la Computación (ICC). Buenos Aires, Argentina, **3** CONICET-Universidad de Buenos Aires. Instituto de Cálculo (IC). Buenos Aires, Argentina, **4** Cognitive Neuroimaging Unit, CEA DSV/I2BM, INSERM, Université Paris-Sud, Université Paris-Saclay, NeuroSpin center, 91191 Gif/Yvette, France, **5** CONICET-Universidad Torcuato Di Tella. Laboratorio de Neurociencia, C1428BIJ. Buenos Aires, Argentina

* sgromano@dc.uba.ar

Citation: Romano S, Salles A, Amalric M, Dehaene S, Sigman M, Figueira S (2018) Bayesian validation of grammar productions for the language of thought. PLoS ONE 13(7): e0200420. https://doi.org/10.1371/journal.pone.0200420

Editor: Zaid Abdo, Colorado State University, UNITED STATES

Received: August 28, 2017

Accepted: June 26, 2018

Published: July 10, 2018

Data Availability Statement: All relevant data are within the paper and its Supporting Information files.

Funding: This work was supported by "Secretaría de Ciencia y Técnica, Universidad de Buenos Aires" (UBACyT 20020150100002BA, http://www.uba.ar/secyt/contenido.php?id=42&s=50) (SR) and "Fondo para la Investigación Científica y Tecnológica, Agencia Nacional de Promoción Científica y Tecnológica" (PICT-2016-0215, http://www.agencia.mincyt.gob.ar/frontend/agencia/instrumento/24) (SF). In addition, part of the work

Abstract

Probabilistic proposals of Language of Thoughts (LoTs) can explain learning across different domains as statistical inference over a compositionally structured hypothesis space. While frameworks may differ on how a LoT may be implemented computationally, they all share the property that they are built from a set of atomic symbols and rules by which these symbols can be combined. In this work we propose an extra validation step for the set of atomic productions defined by the experimenter. It starts by expanding the defined LoT grammar for the cognitive domain with a broader set of arbitrary productions and then uses Bayesian inference to prune the productions from the experimental data. The result allows the researcher to validate that the resulting grammar still matches the intuitive grammar chosen for the domain. We then test this method in the *language of geometry*, a specific LoT model for geometrical sequence learning. Finally, despite the fact of the geometrical LoT not being a universal (i.e. Turing-complete) language, we show an empirical relation between a sequence's *probability* and its *complexity* consistent with the theoretical relationship for universal languages described by Levin's Coding Theorem.

Introduction

> It was not only difficult for him to understand that the generic term dog embraced so many unlike specimens of differing sizes and different forms; he was disturbed by the fact that a dog at three-fourteen (seen in profile) should have the same name as the dog at three-fifteen (seen from the front). (. . .)With no effort he had learned English, French, Portuguese and Latin. I suspect, however, that he was not very capable of thought. To think is to forget differences, generalize, make abstractions. In the teeming world of Funes, there were only details, almost immediate in their presence. [1]

done by one author (AS) was supported by Consejo Nacional de Investigaciones Científicas y Técnicas (PIP 11220130100384CO, http://convocatorias.conicet.gov.ar/investigacion-y-desarrollo/).

Competing interests: The authors have declared that no competing interests exist.

In his fantasy story, the writer Jorge Luis Borges described a fictional character, Funes, capable of remembering every detail of his life but not being able to generalize any of that data into mental categories and hence –Borges stressed– not capable of thinking.

Researchers have modeled these mental categories or conceptual classes with two classical approaches: in terms of similarity to a generic example or prototype [2–5] or based on a symbolic/rule-like representation [6–8].

Symbolic approaches like the *language of thought* (LoT) hypothesis [7], claim that thinking takes form in a sort of mental language, composed of a limited set of atomic symbols that can be combined to form more complex structures following combinatorial rules.

Despite criticisms and objections [9–12], symbolic approaches —in general— and the LoT hypothesis —in particular— have gained some renewed attention with recent results that might explain learning across different domains as statistical inference over a compositionally structured hypothesis space [13, 14].

The LoT is not necessarily unique. In fact, the form that it takes has been modeled in many different ways depending on the problem domain: numerical concept learning [15], sequence learning [16–18], visual concept learning [19], theory learning [20], etc.

While frameworks may differ on how a LoT may be implemented computationally, they all share the property of being built from a set of atomic symbols and rules by which they can be combined to form new and more complex expressions.

Most studies of LoTs have focused on the compositional aspect of the language, which has either been modeled within a Bayesian [13] or a Minimum Description Length (MDL) framework [16, 18, 21, 22].

The common method is to define a grammar with a set of productions based on operations that are intuitive to researchers and then study how different inference processes match regular patterns in human learning. A recent study [23] puts the focus on the process of how to empirically choose the set of productions and how different LoT definitions can create different patterns of learning. Here, we move along that direction but use Bayesian inference to individuate the LoT instead of comparing several of them by hand.

Broadly, our aim is to propose a method to select the set of atomic symbols in an inferential process by pruning and trimming from a broad repertoire. More precisely, we test whether Bayesian inference can be used to decide the proper set of productions in a LoT defined by a context free grammar. These productions are derived from the subjects' experimental data. In order to do this, a researcher builds a broader language with two sets of productions: 1) those for which she has a strong prior conviction that they should be used in the cognitive task, and 2) other productions that could be used to structure the data and extract regularities even if she believes are not part of the human reasoning repertoire for the task. With the new broader language, she should then turn the context free grammar that defines it into a probabilistic context free grammar (PCFG) and use Bayesian analysis to infer the probability of each production in order to choose the set that best explains the data.

In the next section we formalize this procedure and then apply it on the *language of geometry* presented by Amalric et al. in a recent study about geometrical sequence learning [16]. This LoT defines a language with some basic geometric instructions as the grammar productions and then models their composition within the MDL framework. Our method, however, can be applied to any LoT model that defines a grammar, independently of whether its compositional aspect is modeled using a Bayesian framework or a MDL approach.

Finally, even with the recent surge of popularity of Bayesian inference and MDL in cognitive science, there are –to the best of our knowledge– no practical attempts to close the gap between probabilistic and complexity approaches to LoT models.

The theory of computation, through Levin's Coding Theorem [24], exposes a remarkable relationship between the *Kolmogorov complexity* of a sequence and its *universal probability*, largely used in algorithmic information theory. Although both metrics are actually non-computable and defined over a universal prefix Turing Machine, we can apply both ideas to other non-universal Turing Machines in the same way that the concept of complexity used in MDL can be computed for specific, non-universal languages.

In this work, we examine the extent to which this theoretical prediction for infinite sequences holds empirically for a specific LoT, the *language of geometry*. Although the inverse logarithmic relationship between both metrics is proved for universal languages in the Coding Theorem, testing this same property for a particular non-universal language shows that the language shares some interesting properties of general languages. This constitutes a first step towards a formal link between probability and complexity modeling frameworks for LoTs.

Bayesian inference for LoT's productions

The project of Bayesian analysis of the LoT models concept learning using Bayesian inference in a grammatically structured hypothesis space [25]. Each LoT proposal is usually formalized by a context free grammar $\mathcal{G}$ that defines the valid functions or programs that can be generated, like in any other programming language. A program is a derivation tree of $\mathcal{G}$ that needs to be interpreted or executed according to a given semantics in order to get an actual description of the concept in the cognitive task at hand. Therefore, each concept is then represented by any of the programs that describe it and a Bayesian inference process is defined in order to infer from the observed data the distribution of valid programs in $\mathcal{G}$ that describes the concepts.

As explained above, our aim is to derive the productions of $\mathcal{G}$ from the data, instead of just conjecturing them using a priori knowledge about the task. Prior work on LoTs has fit probabilities of productions in a context free grammar using Bayesian inference, however, the focus has been put in integrating out the production probabilities to better predict the data without changing the grammar definition [23]. Here, we want to study if the inference process could let us decide which productions can be safely pruned from the grammar. We introduce a generic method that can be used on any grammar to select and test the proper set of productions. Instead of using a fixed grammar and adjusting the probabilities of the productions to predict the data, we use Bayesian inference to rule out productions with probability lower than a certain threshold. This allows the researcher to validate the adequacy of the productions she has chosen for the grammar or even define one that is broad enough to express different regularities and let the method select the best set for the observed data.

To infer the probability for each production based on the observed data, we need to add a vector of probabilities θ associated with each production in order to convert the context free grammar $\mathcal{G}$ into a probabilistic context free grammar (PCFG) [26].

Let $D = (d_1, d_2, \ldots, d_n)$ denote the list of concepts produced by the subjects in an experiment. This means that each d_i is a concept produced by a subject in each trial. Then, $P(\theta \mid D)$, the posterior probability of the weights of each production after the observed data, can be calculated by marginalizing over the possible programs that compute D:

$$P(\theta \mid D) = \sum_{\text{Prog}} P(\text{Prog}, \theta \mid D), \tag{1}$$

where each $\text{Prog} = (p_1, p_2, \cdots, p_n)$ is a possible set of programs such that each p_i computes the corresponding concept d_i.

We can use Bayesian inference to learn the corresponding programs Prog and the vector θ for each production in the grammar, applying Bayes rule in the following way:

$$P(\text{Prog}, \theta \mid D) \propto P(D \mid \text{Prog})\ P(\text{Prog} \mid \theta)\ P(\theta), \tag{2}$$

Sampling the set of programs from $P(\text{Prog} \mid \theta)$ forces an inductive bias which is needed to handle uncertainty under sparse data. Here we use a standard prior for programs that is common in the LoT literature to introduce a syntactic complexity bias that favors shorter programs [25, 27]. Intuitively, the probability of sampling a certain program is proportional to the product of the production rules that were used to generate such program, and therefore inversely proportional to the size of the derivation tree. Formally, it is defined as:

$$P(\text{Prog} \mid \theta) \quad = \prod_{i=1}^{n} P(p_i \mid \theta), \tag{3}$$

where $P(p_i \mid \theta) = \prod_{r \in G} \theta_r^{f_r(p_i)}$ is the probability of the program p_i in the grammar, and $f_r(p_i)$ is the number of occurrences of the production r in program p_i.

In (2), $P(\theta)$ is a Dirichlet prior over the productions of the grammar. By using the term $P(\theta)$ we are abusing notation for simplicity. The proper term would be $P(\theta \mid \alpha)$ to express a Dirichlet prior with $\alpha \in \mathbb{R}^{\ell}$ its associated concentration vector hyper-parameter where ℓ is the number of productions in the grammar. This hierarchical Dirichlet prior has sometimes been replaced with a uniform prior on productions as it shows no significant differences in prediction results [15, 17]. However, here we will use the Dirichlet prior to be able to infer the production probabilities from this more flexible model.

The likelihood function is straightforward. It does not use any free parameter to account for perception errors in the observation. This forces that only programs that compute the exact concept are taken into account, and it can be easily calculated as follows:

$$P(D \mid \text{Prog}) \quad = \prod_{i=1}^{n} P(d_i \mid p_i), \tag{4}$$

where $P(d_i \mid p_i) = 1$ if the program p_i computes d_i, and 0 otherwise.

Calculating $P(\theta \mid D)$ directly is, however, not tractable since it requires to sum over all possible combinations of programs Prog for each of the possible values of θ. To this aim, then, we used a Gibbs Sampling [28] algorithm for PCFGs via Markov Chain Monte Carlo (MCMC) similar to the one proposed at [29], which alternates in each step of the chain between the two conditional distributions:

$$P(\text{Prog} \mid \theta, D) \quad = \quad \prod_{i=1}^{n} P(p_i \mid d_i, \theta). \tag{5}$$

$$P(\theta \mid \text{Prog}, D) \quad = \quad P_D(\theta \mid f(\text{Prog}) + \alpha). \tag{6}$$

Here, P_D is the Dirichlet distribution where the positions of the vector α were updated by counting the occurrences of the corresponding productions for all programs $p_i \in \text{Prog}$.

In the next section, we apply this method to a specific LoT. We add a new set of ad-hoc productions to the grammar that can explain regularities but are not related to the cognitive task. Intuitively, these ad-hoc productions should not be part of the human LoT repertory, still all of them can be used in many possible programs to express each concept.

So far, Probabilistic LoT approaches have been successful to model concept learning from few examples [13, 14]. However, this does not mean that Bayesian models would be able to infer the syntax of the model's grammar from sparse data. Here we test such hypothesis. If the method is effective, it should assign a low probability to the ad-hoc productions and instead favor the original set of productions selected by the researchers for the cognitive task. This would not only provide additional empirical evidence about the adequacy of the choice of the original productions for the selected LoT but, more importantly, about the usefulness of Bayesian inference for validating the set of productions involved in different LoTs.

The language of geometry: $\mathcal{G}eo$

The *language of geometry*, $\mathcal{G}eo$ [16], is a probabilistic generator of sequences of movements on a regular octagon like the one in Fig 1. It has been used to model human sequence predictions in adults, preschoolers, and adult members of an indigene group in the Amazon. As in other LoT domains, different models have been proposed for similar spatial sequence domains like the one in [17]. Although both successfully model the sequences in their experiments, they propose different grammars for their models (in particular, [16] contains productions for expressing symmetry reflections). This difference can be explained by the particularities of each experiment. On the one hand, [16] categorized the sequences in 12 groups based on their complexity, displayed them in an octagon and evaluate the performance of a diverse population to extrapolate them. On the other hand, [17] categorized the sequences in 4 groups, displayed them in an heptagon and evaluate the performance of adults not just to predict how the sequence continues, but to transfer the knowledge from the learned sequence across auditory and visual domains. Despite the domains not being equal, the differences in the grammars strengths the need for automatic methods to test and validate multiple grammars for the same domain in the LoT community.

The production rules of grammar $\mathcal{G}eo$ were selected based on previous claims of the universality of certain human geometrical knowledge [30–32] such as spatial notions [33, 34] and detection of symmetries [35, 36].

With these production rules, sequences are described by concatenating or repeating sequence of movements in the octagon. The original set of productions is shown in Table 1 and –besides the concatenation and repetition operators– it includes the following family of atomic geometrical transition productions: anticlockwise movements, staying at the same location, clockwise movements and symmetry movements.

The language actually supports not just a simple n times repetition of a block of productions, but it also supports two more complex productions in the repetition family: repeating with a change in the starting point after each cycle and repeating with a change to the resulting sequence after each cycle. More details about the formal syntax and semantics can be found in [16], though they are not needed here.

Each program p generated by the grammar describes a mapping $\Sigma \rightarrow \Sigma^+$, for $\Sigma = \{0, \ldots, 7\}$. Here, Σ^+ represents the set of all (non empty) finite sequences over the alphabet Σ, which can be understood as a finite sequence of points in the octagon. These programs must then be executed or interpreted from a starting point in order to get the resulting sequence of points. Let $p = [+1, +1]$ be a program, then $p(0)$ is the result of executing p starting from point 0 (that is, sequence 1, 2) and $p(4)$ is the result of executing the same program starting from point 4 in the octagon (sequence 5, 6).

Each sequence can be described with many different programs: from a simple concatenation of atomic productions to more compressed forms using repetitions. For example, to move through all the octagon clockwise one point at a time starting from point 0, one can use

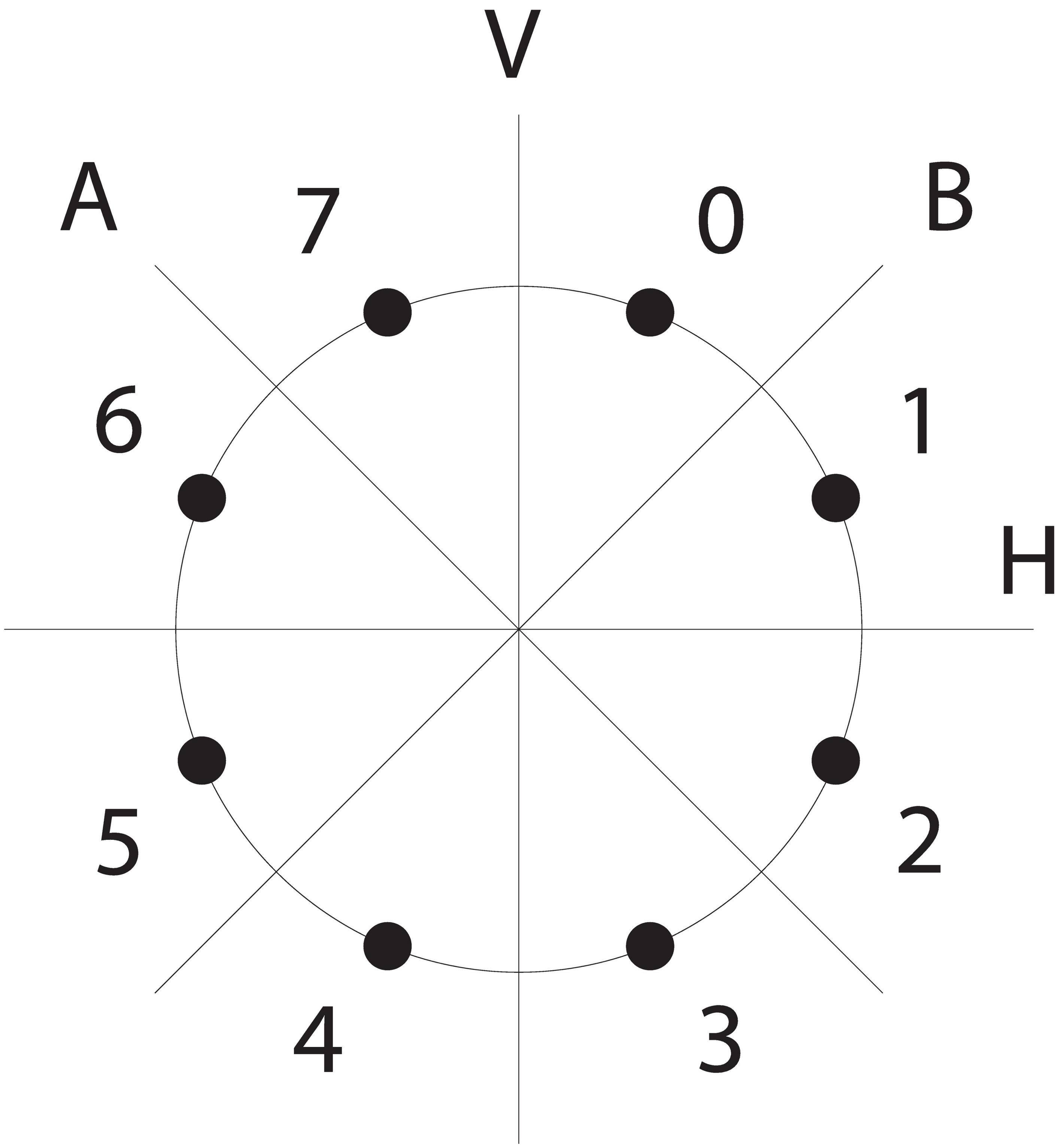

Fig 1. Possible sequence positions and reflection axes. Σ points around a circle to map current position in the octagon, and the reflection axes.

https://doi.org/10.1371/journal.pone.0200420.g001

Table 1. Original grammar.

Start production			
START	→	[INST]	start symbol
Basic productions			
INST	→	ATOMIC	atomic production
INST	→	INST,INST	concatenation
INST	→	REP[INST]^{n}	repeat family with $n \in [2, 8]$
REP	→	REP0	simple repeat
REP	→	REP1<ATOMIC>	repeat with starting point variation using ATOMIC
REP	→	REP2<ATOMIC>	repeat with resulting sequence variation using ATOMIC
Atomic productions			
ATOMIC	→	-1	next element anticlockwise (ACW)
ATOMIC	→	-2	second element ACW
ATOMIC	→	-3	third element ACW
ATOMIC	→	+0	stays at same location
ATOMIC	→	+1	next element clockwise (CW)
ATOMIC	→	+2	second element CW
ATOMIC	→	+3	third element CW
ATOMIC	→	A	symmetry around one diagonal axis
ATOMIC	→	B	symmetry around the other diagonal axis
ATOMIC	→	H	horizontal symmetry
ATOMIC	→	V	vertical symmetry
ATOMIC	→	P	rotational symmetry

https://doi.org/10.1371/journal.pone.0200420.t001

[+1, +1, +1, +1, +1, +1, +1, +1](0) or [REP[+1]^{8}](0) or [REP[+1]^{7}, +1](0), etc. To alternate 8 times between points 6 and 7, one can use a reflection production like [REP[A]^{8}](6), or [REP [+1, -1]^{4}](6).

Geo's original experiment

To infer the productions from the observed data, we used the original data from the experiment in [16]. In the experiment, volunteers were exposed to a series of spatial sequences defined on an octagon and were asked to predict future locations. The sequences were selected according to their MDL in the *language of geometry* so that each sequence could be easily described with few productions.

Participants. The data used in this work comes, except otherwise stated, from Experiment 1 in which participants were 23 French adults (12 female, mean age = 26.6, age range = 20 – 46) with college-level education. Data from Experiment 2 is later used when comparing adults and children results. In the later, participants where 24 preschoolers (minimal age = 5.33, max = 6.29, mean = 5.83 ± 0.05).

Procedure. On each trial, the first two points from the sequence were flashed sequentially in the octagon and the user had to click on the next location. If the subject selected the correct location, she was asked to continue with the next point until the eight points of the sequences were completed. If there was an error at any point, the mistake was corrected, the sequence flashed again from the first point to the corrected point and the user asked to predict the next location. Each $d_i \in \Sigma^8$ from our dataset D is thus the sequence of eight positions clicked in each subject's trial. The detailed procedure can be found in the cited work.

Table 2. Ad-hoc productions.

ATOMIC	→	DOUBLE	(location * 2) mod 8
ATOMIC	→	-DOUBLE	(location * − 2) mod 8
ATOMIC	→	SQUARE	(location^2) mod 8
ATOMIC	→	GAMMA	Γ(location+1) mod 8
ATOMIC	→	PI	location-th digit of π
ATOMIC	→	EULER	location-th digit of e
ATOMIC	→	GOLD	location-th digit of ϕ
ATOMIC	→	PYTH	location-th digit of $\sqrt{2}$
ATOMIC	→	KHINCHIN	location-th digit of Khinchin's constant
ATOMIC	→	GLAISHER	location-th digit of Glaisher's constant
ATOMIC	→	CHAITIN	location-th digit of Chaitin Omega's constant

https://doi.org/10.1371/journal.pone.0200420.t002

Extending $\mathcal{G}eo$'s grammar

We will now expand the original set of productions in $\mathcal{G}eo$ with a new set of productions that can also express regularities but are not related to any geometrical intuitions to test our Bayesian inference model.

In Table 2 we show the new set of productions which includes instructions like moving to the point whose label is the square of the current location's label, or using the current point location i to select the i^{th}digit of a well-known number like π or Chaitin's number (calculated for a particular universal Turing Machine and programs up to 84 bits long [37]). All digits are returned in arithmetic module 8 to get a valid point for the next position. For example, PI(0) returns the first digit of π, that is PI(0) = 3 mod (8) = 3; and PI(1) = 1.

Inference results for $\mathcal{G}eo$

To let the MCMC converge faster (and to later compare the concept's probability with their corresponding MDL), we generated all the programs that explain each of the observed sequences from the experiment. In this way, we are able to sample from the exact distribution $P(p_i \mid d_i, \theta)$ by sampling from a multinomial distribution of all the possible programs p_i that compute d_i, where each p_i has probability of occurrence equal to $P(p_i \mid \theta)$.

To get an idea of the expressiveness of the grammar to generate different programs for a sequence and the cost of computing them, it is worth mentioning that there are more than 159 million programs that compute the 292 unique sequences generated by the subjects in the experiment, and that for each sequence there is an average of 546,713 programs (min = 10, 749, max = 5, 500, 026, $\sigma = 693, 618$).

Fig 2 shows the inferred θ for the observed sequences from subjects, with a unit concentration parameter for the Dirichlet prior, $\alpha = (1, \ldots, 1)$. Each bar shows the mean probability and the standard error of each of the atomic productions after 50 steps of the MCMC, leaving the first 10 steps out as burn-in.

Although 50 steps might seem low for a MCMC algorithm to converge, our method calculated $P(p_i \mid d_i, \theta)$ exactly in order to speed up convergence and to be able to later compare the probability with the complexity from the original MDL model. In Fig 3, we show an example trace for four MCMC runs for θ_{+0}, which corresponds to the atomic production +0, but is representative of the behavior of all θ_i. (see S1 Fig for the full set of productions).

Fig 2 shows a remarkable difference between the probability of the productions that were originally used based on geometrical intuitions and the ad-hoc productions. The plot also

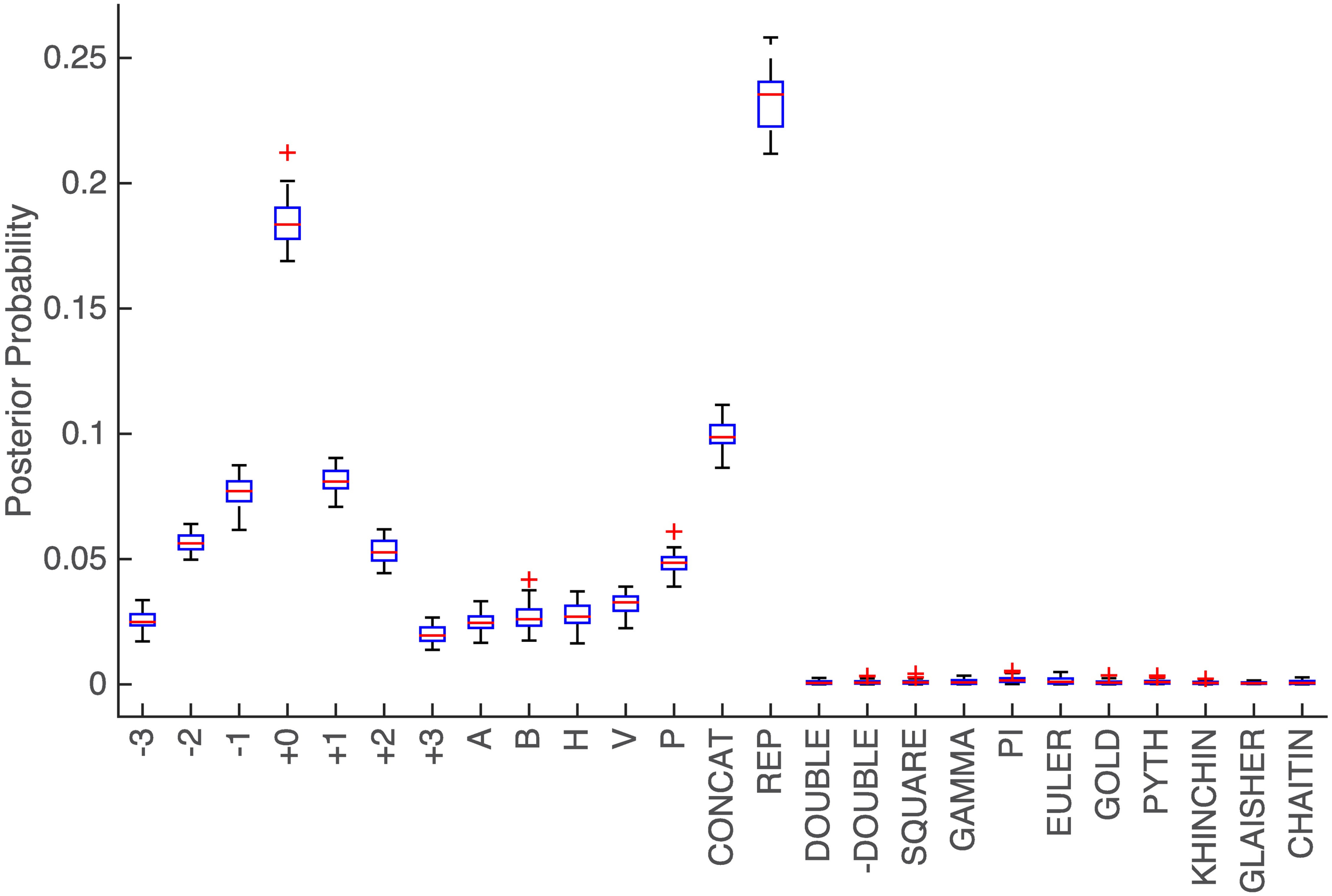

Fig 2. Inferred θ_i. Inferred probability for each production in the grammar.

https://doi.org/10.1371/journal.pone.0200420.g002

shows that each clockwise production has almost the same probability as its corresponding anticlockwise production, and a similar relation appears between horizontal and vertical symmetry (H and V) and symmetries around diagonal axes (A and B). This is important because the original experiment was designed to balance such behavior; the inferred grammar reflects this.

Fig 4 shows the same inferred θ but grouped according to production family. Grouping stresses the low probability of all the ad-hoc productions, but also shows an important difference between REP and the rest of the productions, particularly the simple concatenation of productions (CONCAT). This indicates that the *language of geometry* is capable of reusing simpler structures that capture geometrical meaning to explain the observed data, a key aspect of a successful model of LoT.

We then ran the same inference method using observed sequences from other experiments but only with the original grammar productions (i.e. setting aside the ad-hoc productions). We compared the result of inferring over our previously analyzed sequences generated by adults with sequences generated by children (experiment 2 from [16]) and the actual expected sequences for an ideal player.

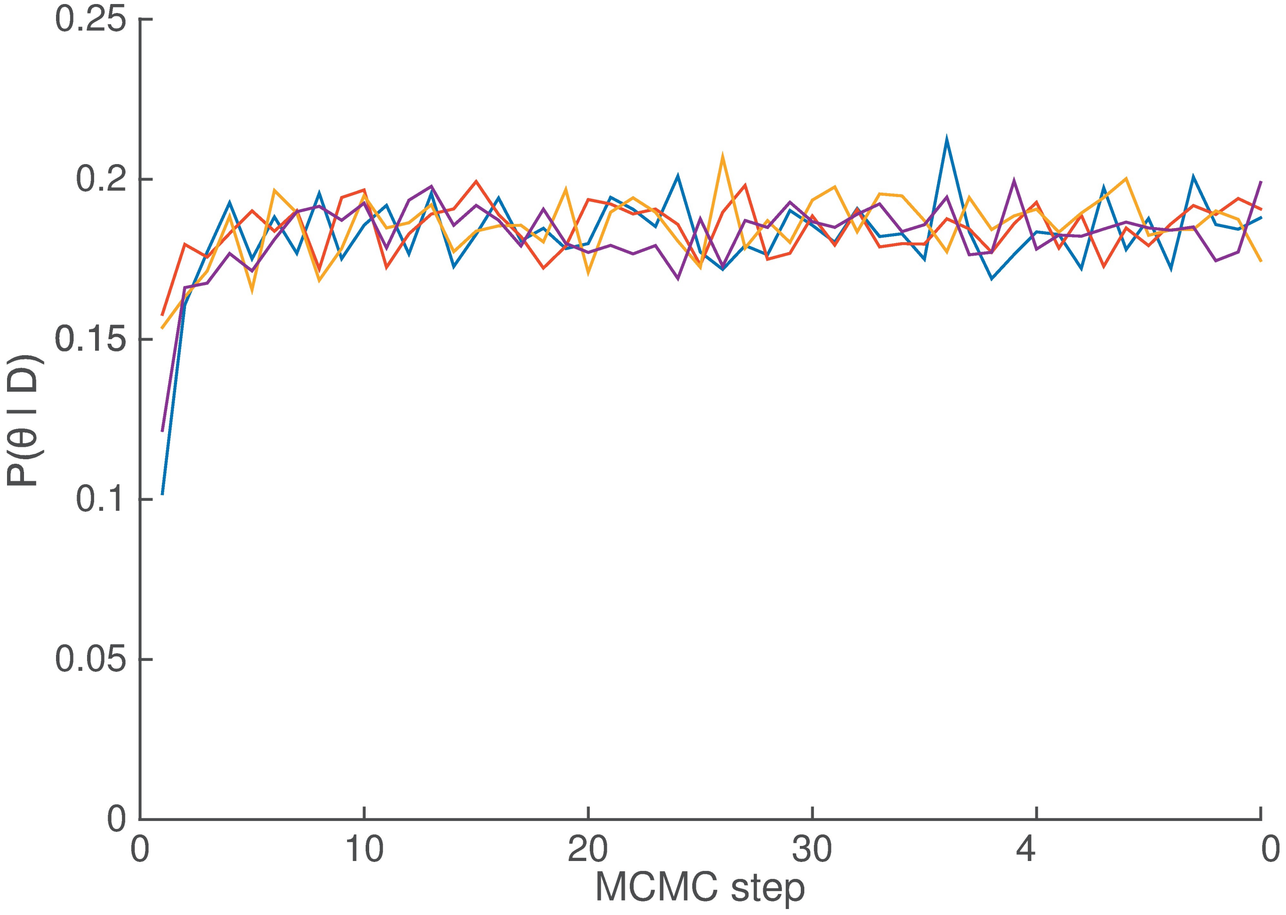

Fig 3. Inferred θ_{+0}. Inferred probability for +0 production at each step in four MCMC chains.

https://doi.org/10.1371/journal.pone.0200420.g003

Fig 5 shows the probabilities for each atomic production that is inferred after each population. The figure denotes that different populations can converge to different probabilities and thus different LoTs. Specifically, it is worth mentioning that the ideal learner indeed uses more repetition productions than simple concatenations when compared to adults. In the same way, adults use more repetitions than children. This could mean that the ideal learner is capable of reproducing the sequences by recursively embedding other smaller programs, whereas adults and children more so have problems understanding or learning the smaller concept that can explain all the sequences from the experiments, which is consistent with the results from the MDL model in [16].

It is worth mentioning that in [16] the complete grammar for the *language of geometry* could explain adults' behavior but had problems to reproduce the children's patterns for some sequences. However, they also showed that penalizing the rotational symmetry (P) could adequately explain children's behavior. In Fig 5, we see that the mean value of (P) for children is 0.06 whereas in adults it's 0.05 (a two-sample t-test reveals t = -12.6, p = 10–19). This might not necessarily be contradictory, as the model for children in [16] was used to predict the next symbol of a sequence after seeing its prefix by adding a penalization for extensions that use the rotational symmetry in the *minimal* program of each sequence. On the other hand, the Bayesian model in this work tries to explain the observed sequences produced by children

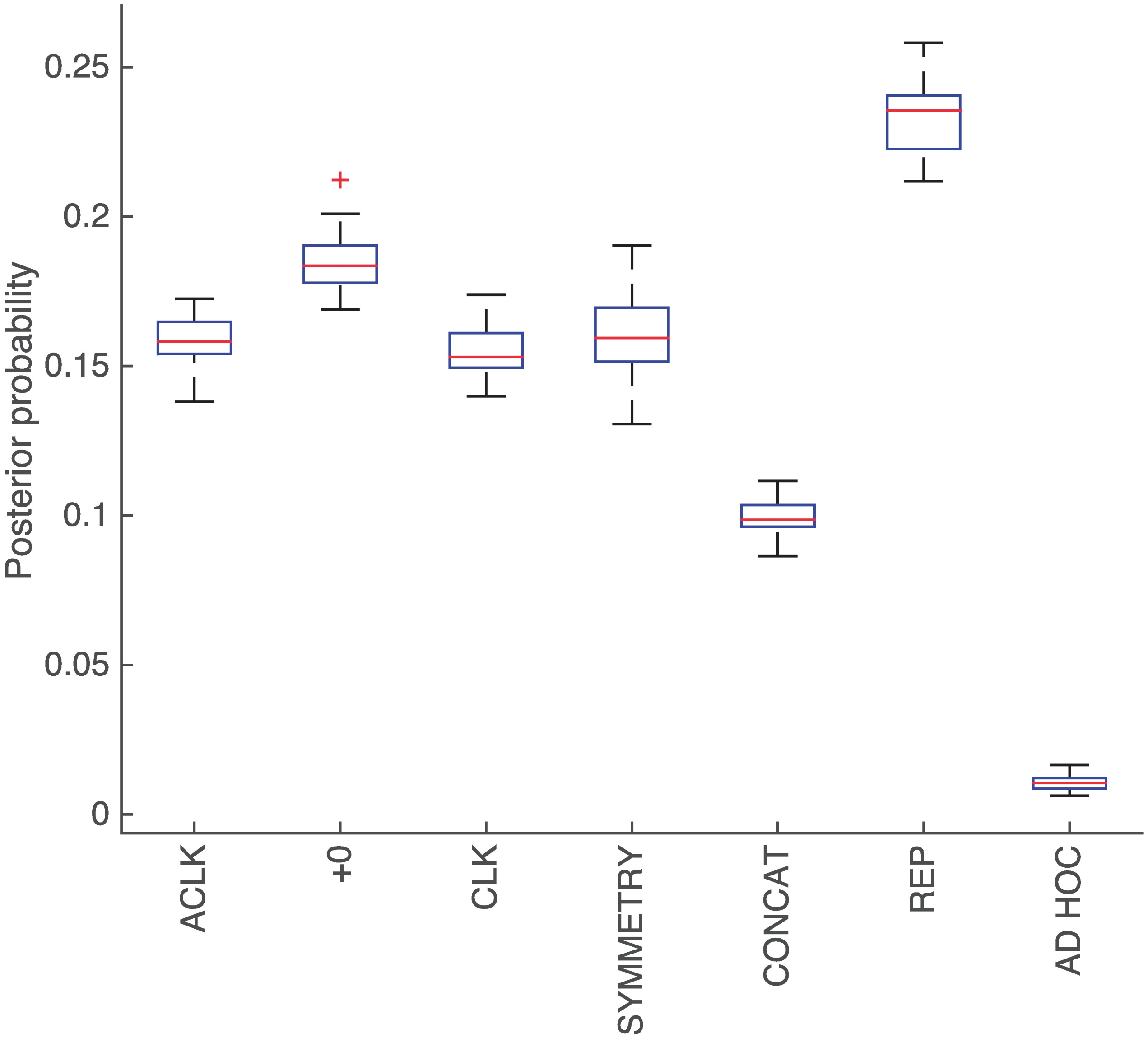

Fig 4. Inferred θ_i grouped by family. Inferred probability for each production in the grammar grouped by family.

https://doi.org/10.1371/journal.pone.0200420.g004

considering the probability of a sequence summing over *all* the possible programs that can generate it and not just the ones with minimal size. Thus, a production like (P) that might not be part of the minimal program for a sequence might not necessarily be less probable when considering the entire distribution of programs for that same sequence.

Coding Theorem

For each phenomenon there can always be an extremely large, possibly infinite, number of explanations. In a LoT model, this space is constrained by the grammar $\mathcal{G}$ that defines the valid hypotheses in the language. Still, one has to define how a hypothesis is chosen among all possibilities. Following Occam's razor, one should choose the simplest hypothesis amongst all the possible ones that explain a phenomenon. In cognitive science, the MDL framework has

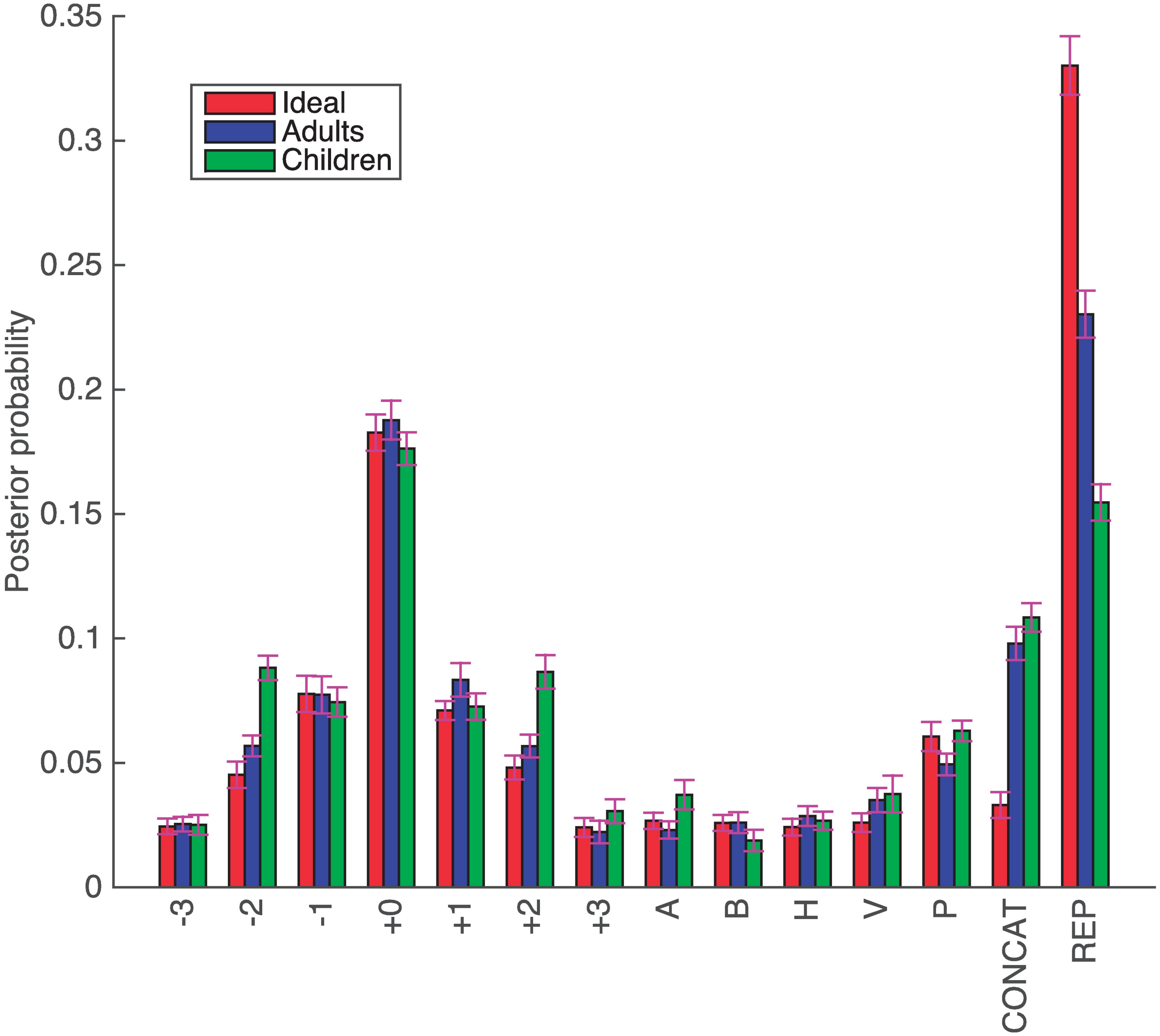

Fig 5. Inferred θ_i for ideal learner, adults and children. Inferred probability for each production in the grammar for different population data.

https://doi.org/10.1371/journal.pone.0200420.g005

been widely used to model such bias in human cognition, and in *the language of geometry* in particular [16]. The MDL framework is based on the ideas of information theory [38], Kolmogorov complexity [39] and Solomonoff induction [40].

Occam's razor was formalized by Solomonoff [40] in his theory of universal inductive inference, which proposes a universal prediction method that successfully approximates any distribution μ based on previous observations, with the only assumption of μ being computable. In short, Solomonoff's theory uses all programs (in the form of prefix Turing machines) that can describe previous observations of a sequence to calculate the probability of the next symbols in an optimal fashion, giving more weight to shorter programs. Intuitively, simpler theories with low complexity have higher probability than theories with higher complexity. Formally, this relationship is described by the Coding Theorem [24], which closes the gap between the concepts of Kolmogorov complexity and probability theory. However, LoT models that define a probabilistic distribution for their hypotheses do not attempt to compare it with a complexity measure of the hypotheses like the ones used in MDL, nor the other way around.

In what follows we formalize the Coding Theorem (for more information, see [41]) and test it experimentally. To the best our knowledge, this is the first attempt to validate these ideas for a particular (non universal) language. The reader should note that we are not validating the theorem itself as it has already been proved for universal Turing Machines. Here, we are testing whether the inverse logarithmic relationship between the probability and complexity holds true when defined for a specific non universal language.

The formal statement

Let M be a prefix Turing machine –by *prefix* we mean that if $M(x)$ is defined, then M is undefined for every proper extension of x. Let $P_M(x)$ be the probability that the machine M computes output x when the input is filled-up with the results of fair coin tosses, and let $K_M(x)$ be the *Kolmogorov complexity of x relative to M*, which is defined as the length of the shortest program which outputs x, when executed on M. The Coding Theorem states that for every string x we have

$$\log \frac{1}{P_U(x)} = K_U(x) \tag{7}$$

up to an additive constant, whenever U is a *universal* prefix Turing machine –by *universal* we mean a machine which is capable of simulating every other Turing machine; it can be understood as the underlying (Turing-complete) chosen programming language. It is important to remark that neither P_U, nor K_U are computable, which means that such mappings cannot be obtained through effective means. However, for specific (non-universal) machines M, one can, indeed, compute both P_M and K_M.

Testing the Coding Theorem for $\mathcal{G}eo$

Despite the fact that P_M and K_M are defined over a Turing Machine M, the reader should note that a LoT is not usually formalized with a Turing Machine, but instead as a programming language with its own syntax of valid programs and semantics of execution, which stipulates how to compute a concept from a program. However, one can understand programming languages as defining an equivalent (not necessarily universal) Turing Machine model, and a LoT as defining its equivalent (not necessarily universal) Turing Machine $\mathcal{G}$. In short, machines and languages are interchangeable in this context: they both specify the programs/terms, which are symbolic objects that, in turn, describe semantic objects, namely, strings.

The Kolmogorov complexity relative to $\mathcal{G}eo$. In [16], the Minimal Description Length was used to model the combination of productions from the *language of geometry* into concepts by defining a Kolmogorov complexity relative to the *language of geometry*, which we denote $K_{\mathcal{G}eo}$. $K_{\mathcal{G}eo}(x)$ is the minimal size of an expression in the grammar of $\mathcal{G}eo$ which describes x. The formal definition of 'size' can be found in the cited work but in short: each of the atomic productions adds a fixed cost of 2 units; using any of the repetition productions to iterate n times a list of other productions adds the cost of the list, plus $\lfloor \log(n) \rfloor$; and joining two lists with a concatenation costs the same as the sum of the costs of both lists.

The probability relative to $\mathcal{G}eo$. On the other hand, with the Bayesian model specified in this work, we can define $P(x \mid \mathcal{G}eo, \theta)$ which is the probability of a string x relative to $\mathcal{G}eo$ and its vector of probabilities for each of the productions.

For the sake of simplicity, we will use $P_{\mathcal{G}eo}(x)$ to denote $P(x \mid \mathcal{G}eo, \theta)$ when θ is the inferred probability from the observed adult sequences from the experiment.

$$P_{\mathcal{G}eo}(x) = P(x \mid \mathcal{G}eo, \theta) \tag{8}$$

$$= \sum_{\text{prog}} P(x \mid \text{prog}, \theta) \tag{9}$$

$$\propto \sum_{prog} P(x \mid \text{prog}) P(\text{prog} \mid \theta). \tag{10}$$

Here, we calculate both $P_{\mathcal{G}eo}(x)$ and $K_{\mathcal{G}eo(x)}$ in an exact way (note that $\mathcal{G}eo$, seen as a programming language, is not Turing-complete). In this section, we show an experimental equivalence between such measures which is consistent with the Coding Theorem. We should stress, once more, that the theorem does not predict that this relationship should hold for a specific non-universal Turing Machine.

To calculate $P_{\mathcal{G}eo}(x)$ we are not interested in the normalization factor of $P(x \mid \text{prog}) P(\text{prog} \mid \theta)$ because we are just trying to measure the relationship between $P_{\mathcal{G}eo}$ and $K_{\mathcal{G}eo}$ in terms of the Coding Theorem. Note, however, that calculating $P_{\mathcal{G}eo}(x)$ involves calculating all programs that compute each of the sequences as in our previous experiment. To make this tractable we calculated $P_{\mathcal{G}eo}(x)$ for 10,000 unique random sequences for each of the possible sequence lengths from the experiment (i.e., up to eight). When the length of the sequence did not allow 10,000 unique combinations, we used all the possible sequences of that length.

Coding Theorem results

Fig 6 shows the mean probability $P_{\mathcal{G}eo}(x)$ for all sequences x with the same value of $K_{\mathcal{G}eo(x)}$ and length between 4 and 8 ($|x| \in [4, 8]$) for all generated sequences x. The data is plotted with a logarithmic scale for the x-axis, illustrating the inverse logarithmic relationship between $K_{\mathcal{G}eo}(x)$ and $P_{\mathcal{G}eo}(x)$. The fit is very good, with $R^2 = .99$, $R^2 = .94$, $R^2 = .97$, $R^2 = .99$ and $R^2 = .98$ for Fig 6A, 6B, 6C, 6D and 6E, respectively.

This relationship between the complexity $K_{\mathcal{G}eo}$ and the probability $P_{\mathcal{G}eo}$ defined for finite sequences in the *language of geometry*, matches the theoretical prediction for infinite sequences in universal languages described in the Coding Theorem. At the same time, it captures the Occam's razor intuition that the simpler sequences one can produce or explain with this language are also the more probable.

Figs 7 and 8 show the histogram of $P_{\mathcal{G}eo}(x)$ and $K_{\mathcal{G}eo}(x)$, respectively, for sequences with length = 8 to get a better insight about both measures. The histogram of the rest of the sequence's lengths are included in S2 and S3 Figs for completeness, and they all show the same behavior.

Discussion

We have presented a Bayesian inference method to select the set of productions for a LoT and test its effectiveness in the domain of a geometrical cognition task. We have shown that this

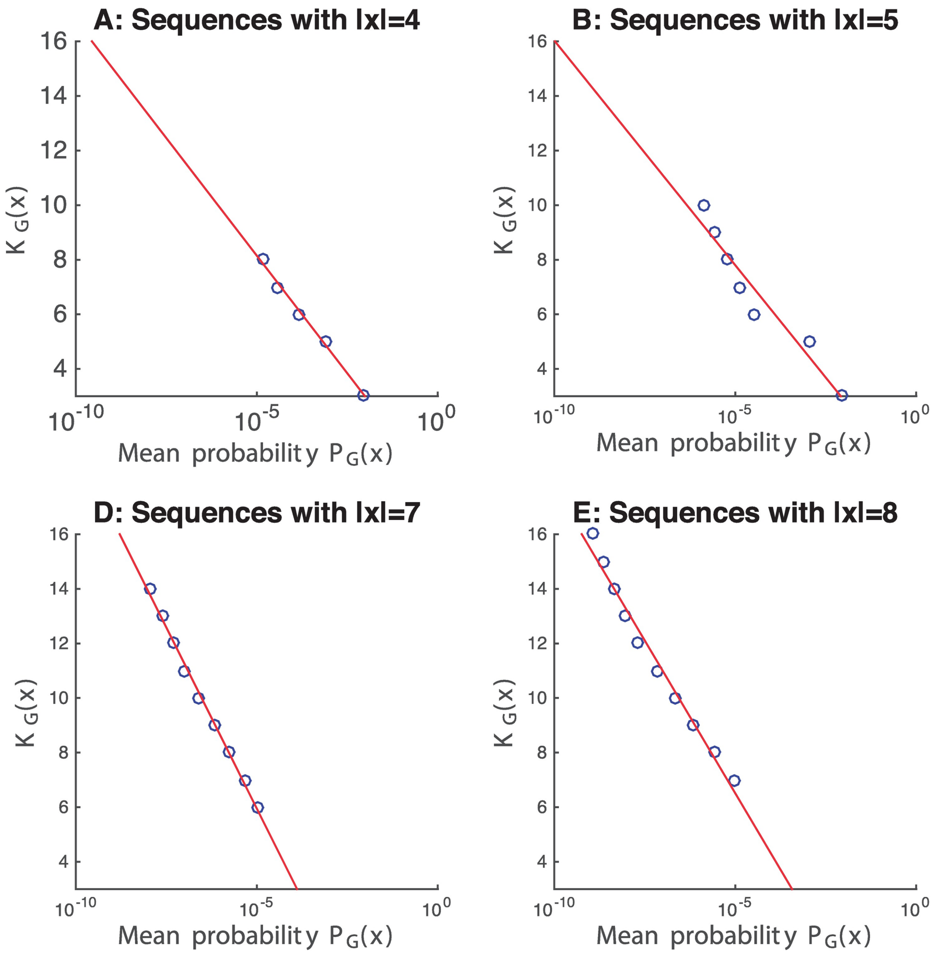

Fig 6. Mean probability $P_{\mathcal{G}eo}(\boldsymbol{x})$. Mean probability $P_{\mathcal{G}eo}(x)$ for all sequences x with the same complexity. Subfigure A: Sequences with $|x| = 4$. Subfigure B: Sequences with $|x| = 5$. Subfigure C: Sequences with $|x| = 6$. Subfigure D: Sequences with $|x| = 7$. Subfigure E: Sequences with $|x| = 8$.

https://doi.org/10.1371/journal.pone.0200420.g006

method is useful to distinguish between arbitrary ad-hoc productions and productions that were intuitively selected to mimic human abilities in such domain.

The proposal to use Bayesian models tied to PCFG grammars in a LoT is not new. However, previous work has not used the inferred probabilities to gain more insight about the grammar definition in order to modify it. Instead, it had usually integrated out the production probabilities to better predict the data, and even found that hierarchical priors for grammar productions show no significant differences in prediction results over uniform priors [15, 17].

We believe that inferring production probabilities can help prove the adequacy of the grammar, and can further lead to a formal mechanism for selecting the correct set of productions when it is not clear what a proper set should be. Researchers could use a much broader set of productions than what might seem intuitive or relevant for the domain and let the hierarchical Bayesian inference framework select the best subset.

Selecting a broader set of productions still leaves some arbitrary decisions to be made. However, it can help to build a more robust methodology that –combined with other ideas like testing grammars with different productions for the same task [23]– could provide more evidence of the adequacy of the proposed LoT.

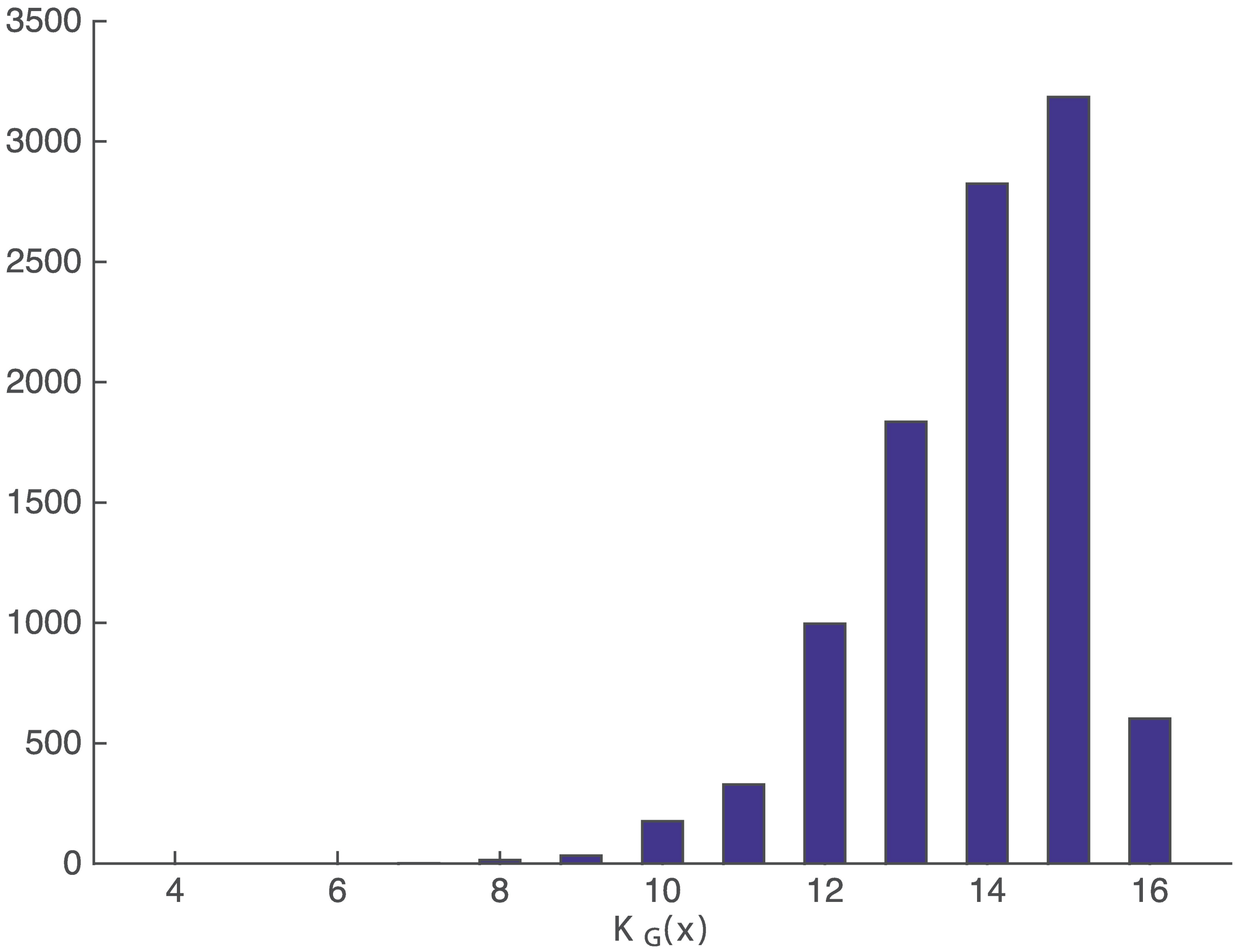

Fig 7. Histogram of complexity $K_{\mathcal{G}eo}(\boldsymbol{x})$. Histogram of complexity for sequences x with $|x| = 8$.

https://doi.org/10.1371/journal.pone.0200420.g007

Having a principled method for defining grammars in LoTs is a crucial aspect for their success because slightly different grammars can lead to different results, as has been shown in [23].

The experimental data used in this work was designed at [16] to understand how humans encode visuo-spatial sequences as structured expressions. As future research, we plan to perform a specific experiment to test these ideas in a broader range of domains. Additionally, data from more domains is needed to demonstrate if this method could also be used to effectively prove whether different people use different LoT productions as outlined in Fig 5.

Finally, we showed an empirical equivalence between the complexity of a sequence in a minimal description length (MDL) model and the probability of the same sequence in a Bayesian inference model which is consistent with the theoretical relationship described in the Coding Theorem. This opens an opportunity to bridge the gap between these two approaches that had been described ad complementary by some authors [42].

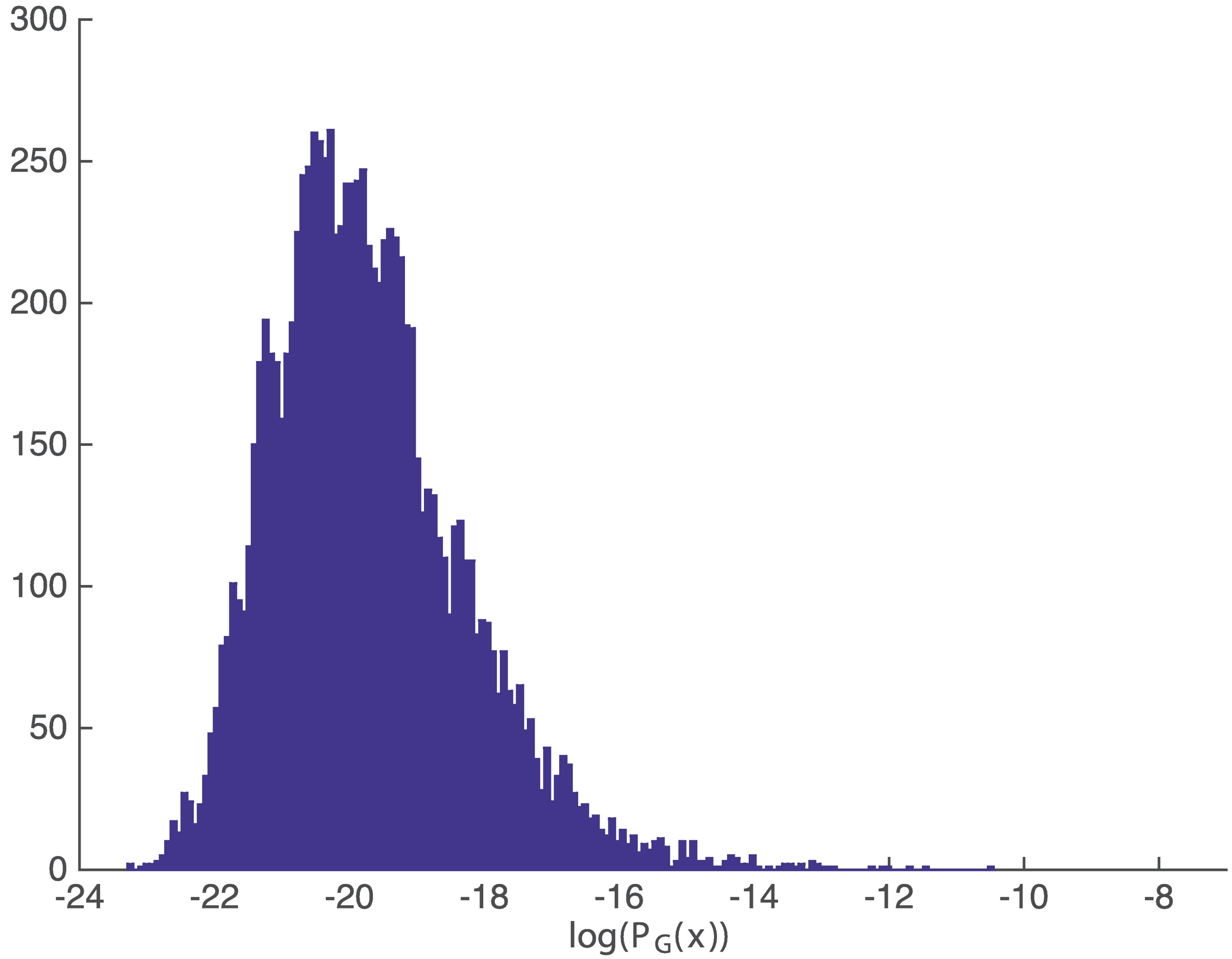

Fig 8. Histogram of probability $P_{\mathcal{G}eo}(x)$. Histogram of probability for sequences x with $|x| = 8$.

https://doi.org/10.1371/journal.pone.0200420.g008

Supporting information

S1 Fig. MCMC steps for $\mathcal{G}eo$'s productions. MCMC steps for the rest of $\mathcal{G}eo$'s grammar productions.
(EPS)

S2 Fig. Histograms of complexity $K_{\mathcal{G}eo}(x)$. Histograms of complexity for sequences with length between 4 and 8.
(EPS)

S3 Fig. Histograms of probability $P_{\mathcal{G}eo}(x)$. Histograms of probability for sequences with length between 4 and 8.
(EPS)

Author Contributions

Conceptualization: Sergio Romano, Stanislas Dehaene, Mariano Sigman, Santiago Figueira.

Data curation: Marie Amalric.

Formal analysis: Sergio Romano, Alejo Salles, Santiago Figueira.

Funding acquisition: Mariano Sigman, Santiago Figueira.

Investigation: Sergio Romano, Mariano Sigman, Santiago Figueira.

Methodology: Sergio Romano, Alejo Salles, Marie Amalric, Santiago Figueira.

Project administration: Mariano Sigman, Santiago Figueira.

Resources: Marie Amalric, Santiago Figueira.

Software: Sergio Romano.

Supervision: Alejo Salles, Mariano Sigman, Santiago Figueira.

Validation: Sergio Romano, Alejo Salles, Stanislas Dehaene, Mariano Sigman, Santiago Figueira.

Visualization: Sergio Romano, Mariano Sigman.

Writing – original draft: Sergio Romano.

Writing – review & editing: Sergio Romano, Alejo Salles, Marie Amalric, Stanislas Dehaene, Mariano Sigman, Santiago Figueira.

References

1. Borges JL. Ficciones, 1935-1944. Buenos Aires: Sur; 1944.
2. Rosch E. Principles of categorization. Concepts: core readings. 1999; 189.
3. Nosofsky RM. Attention, similarity, and the identification–categorization relationship. Journal of experimental psychology: General. 1986; 115(1):39. https://doi.org/10.1037/0096-3445.115.1.39
4. Rosch E, Simpson C, Miller RS. Structural bases of typicality effects. Journal of Experimental Psychology: Human perception and performance. 1976; 2(4):491.
5. Rosch E, Mervis CB. Family resemblances: Studies in the internal structure of categories. Cognitive psychology. 1975; 7(4):573–605. https://doi.org/10.1016/0010-0285(75)90024-9
6. Boole G. An investigation of the laws of thought: on which are founded the mathematical theories of logic and probabilities. Dover Publications; 1854.
7. Fodor JA. The Language of Thought. Language and thought series. Harvard University Press; 1975.
8. Gentner D. Structure-mapping: A theoretical framework for analogy. Cognitive science. 1983; 7 (2):155–170. https://doi.org/10.1207/s15516709cog0702_3
9. Blackburn S. Spreading the Word: Grounding in the Philosophy of Language; 1984.
10. Loewer B, Rey G. Meaning in mind. Fodor and his Critics. 1991;.
11. Knowles J. The language of thought and natural language understanding. Analysis. 1998; 58(4):264–272. https://doi.org/10.1093/analys/58.4.264
12. Aydede M. Language of thought: The connectionist contribution. Minds and Machines. 1997; 7(1):57–101. https://doi.org/10.1023/A:1008203301671
13. Tenenbaum JB, Kemp C, Griffiths TL, Goodman ND. How to grow a mind: Statistics, structure, and abstraction. science. 2011; 331(6022):1279–1285. https://doi.org/10.1126/science.1192788 PMID: 21393536
14. Piantadosi ST, Jacobs RA. Four problems solved by the probabilistic Language of Thought. Current Directions in Psychological Science. 2016; 25(1):54–59. https://doi.org/10.1177/0963721415609581
15. Piantadosi ST, Tenenbaum JB, Goodman ND. Bootstrapping in a language of thought: A formal model of numerical concept learning. Cognition. 2012; 123(2):199–217. https://doi.org/10.1016/j.cognition.2011.11.005 PMID: 22284806
16. Amalric M, Wang L, Pica P, Figueira S, Sigman M, Dehaene S. The language of geometry: Fast comprehension of geometrical primitives and rules in human adults and preschoolers. PLOS

Computational Biology. 2017; 13(1):e1005273. https://doi.org/10.1371/journal.pcbi.1005273 PMID: 28125595

17. Yildirim I, Jacobs RA. Learning multisensory representations for auditory-visual transfer of sequence category knowledge: a probabilistic language of thought approach. Psychonomic bulletin & review. 2015; 22(3):673–686. https://doi.org/10.3758/s13423-014-0734-y

18. Romano S, Sigman M, Figueira S.: A language of thought with Turing-computable Kolmogorov complexity. Papers in Physics. 2013; 5:050001. https://doi.org/10.4279/pip.050001

19. Ellis K, Solar-Lezama A, Tenenbaum J. Unsupervised Learning by Program Synthesis. In: Advances in Neural Information Processing Systems; 2015. p. 973–981.

20. Ullman TD, Goodman ND, Tenenbaum JB. Theory learning as stochastic search in the language of thought. Cognitive Development. 2012; 27(4):455–480. https://doi.org/10.1016/j.cogdev.2012.07.005

21. Goldsmith J. Probabilistic models of grammar: Phonology as information minimization. Phonological Studies. 2002; 5:21–46.

22. Goldsmith J. Unsupervised learning of the morphology of a natural language. Computational linguistics. 2001; 27(2):153–198. https://doi.org/10.1162/089120101750300490

23. Piantadosi ST, Tenenbaum JB, Goodman ND. The Logical Primitives of Thought: Empirical Foundations for Compositional Cognitive Models. 2016;.

24. Levin LA. Laws of information conservation (nongrowth) and aspects of the foundation of probability theory. Problemy Peredachi Informatsii. 1974; 10(3):30–35.

25. Goodman ND, Tenenbaum JB, Feldman J, Griffiths TL. A Rational Analysis of Rule-Based Concept Learning. Cognitive Science. 2008; 32(1):108–154. https://doi.org/10.1080/03640210701802071 PMID: 21635333

26. Manning CD, Schütze H. Foundations of Statistical Natural Language Processing. MIT Press; 1999.

27. Overlan MC, Jacobs RA, Piantadosi ST. Learning abstract visual concepts via probabilistic program induction in a Language of Thought. Cognition. 2017; 168:320–334. https://doi.org/10.1016/j.cognition.2017.07.005 PMID: 28772189

28. Geman S, Geman D. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. Pattern Analysis and Machine Intelligence, IEEE Transactions on. 1984;(6):721–741. https://doi.org/10.1109/TPAMI.1984.4767596

29. Johnson M, Griffiths TL, Goldwater S. Bayesian Inference for PCFGs via Markov Chain Monte Carlo. In: HLT-NAACL; 2007. p. 139–146.

30. Izard V, Pica P, Dehaene S, Hinchey D, Spelke E. Geometry as a universal mental construction. Space, Time and Number in the Brain. 2011; 19:319–332. https://doi.org/10.1016/B978-0-12-385948-8.00019-0

31. Dehaene S, Izard V, Pica P, Spelke E. Core knowledge of geometry in an Amazonian indigene group. Science. 2006; 311(5759):381–384. https://doi.org/10.1126/science.1121739 PMID: 16424341

32. Dillon MR, Huang Y, Spelke ES. Core foundations of abstract geometry. Proceedings of the National Academy of Sciences. 2013; 110(35):14191–14195. https://doi.org/10.1073/pnas.1312640110

33. Landau B, Gleitman H, Spelke E. Spatial knowledge and geometric representation in a child blind from birth. Science. 1981; 213(4513):1275–1278. https://doi.org/10.1126/science.7268438 PMID: 7268438

34. Lee SA, Sovrano VA, Spelke ES. Navigation as a source of geometric knowledge: Young children's use of length, angle, distance, and direction in a reorientation task. Cognition. 2012; 123(1):144–161. https://doi.org/10.1016/j.cognition.2011.12.015 PMID: 22257573

35. Westphal-Fitch G, Huber L, Gómez JC, Fitch WT. Production and perception rules underlying visual patterns: effects of symmetry and hierarchy. Philosophical Transactions of the Royal Society of London B: Biological Sciences. 2012; 367(1598):2007–2022. https://doi.org/10.1098/rstb.2012.0098 PMID: 22688636

36. Machilsen B, Pauwels M, Wagemans J. The role of vertical mirror symmetry in visual shape detection. Journal of Vision. 2009; 9(12):11–11. https://doi.org/10.1167/9.12.11 PMID: 20053102

37. Calude CS, Dinneen MJ, Shu CK, et al. Computing a glimpse of randomness. Experimental Mathematics. 2002; 11(3):361–370. https://doi.org/10.1080/10586458.2002.10504481

38. Shannon C. A Mathematical Theory of Communication. Bell System Technical Journal. 1948; 27:379–423, 623–656. https://doi.org/10.1002/j.1538-7305.1948.tb01338.x

39. Kolmogorov AN. Three approaches to the quantitative definition of information*. International Journal of Computer Mathematics. 1968; 2(1-4):157–168. https://doi.org/10.1080/00207166808803030

40. Solomonoff RJ. A formal theory of inductive inference. Part I. Information and control. 1964; 7(1):1–22. https://doi.org/10.1016/S0019-9958(64)90223-2

41. Li M, Vitányi P. An introduction to Kolmogorov complexity and its applications. Springer Science & Business Media; 2013.

42. MacKay DJ. Information theory, inference and learning algorithms. Cambridge university press; 2003.