Lectures on Philosophy and Computation

Cristian S. Calude

March–April 2014

Leibniz

Sans les mathématiques on ne pénètre point au fond de la philosophie.
Sans la philosophie on ne pénètre point au fond des mathématiques.
Sans les deux on ne pénètre point au fond de rien.

Without mathematics one cannot understand the fundamentals of philosophy.
Without philosophy we cannot reach the foundation of mathematics.
Without both (mathematics and philosophy) one cannot reach anything that is fundamental.

Bibliography (books)

- J. Barrow. *Impossibility—The Limits of Science and the Science of Limits*, Oxford University Press, Oxford, 1998.
- J. M. Borwein, D. Bailey, R. Girgensohn. *Experimentation in Mathematics: Computational Paths to Discovery*, A.K. Peters, Natick, MA, 2004.
- C. S. Calude. *Information and Randomness – An Algorithmic Perspective*, Springer-Verlag, 2002.

In early times, the concepts of chance and randomness were intertwined with that of fate.

Ancient peoples threw dice to determine fate.

The Chinese formalised odds and chance about 3,000 years ago.

The Greek philosophers discussed randomness at length, but only in non-quantitative forms.

It was only in the sixteenth century that Italian mathematicians began to formalise the odds associated with various games of chance.

Randomness means lack of pattern or predictability; it suggests a non-order or non-coherence in a sequence of symbols or steps, such that there is no intelligible pattern or combination.

Symptoms of randomness:

- typicality,
- lack of patterns,
- unpredictability,
- incompressibility,
- ...

The string

$$000000000000000000000000000000$$

is clearly non-random.

The string

$$001111000100011011010100010000 10$$

seems random.

Which strings are random?

01100110011001100110011001100110

00111100010001100011110001000110

00000000000000000011110001000110

00001010101000000010000000101000

00100100010000101001010001000110

The French mathematician Émile Borel, a pioneer of probability theory, argued that there is no way to formalise in an acceptable way the concept of randomness. His argument is based only one symptom of randomness, typicality and is known today as the *randomness paradox*.

A random bit-string should be "typical": it should not stand out from the crowd of other bit-strings.

Here is Borel's argument. Assume that there is a precise way to distinguish between "random bit-strings" and bit-strings which are "non-random". It does not matter how this criterion was found or operates.

Operationally, we have a precise **criterion** which can be applied to any bit-string and once a bit-string is given, we can say whether the bit-string is *random* or *non-random*.

Can the adopted **criterion** be consistent? The answer is *negative*.

Indeed, choose the first bit-string which **criterion** asserts it is random. This particular bit-string is

> *the first bit-string satisfying the property of being random,*

a property making it atypical, so non-random!

What are we left with?

- randomness does not exist,
- randomness is a matter of degree: we have a hierarchy of definitions of randomness,
- ?

Berry paradox

Name numbers in English.

Zero is the first number.

One is the first non-zero number.

One is the second number.

Find the value of the number defined by:

The first number not nameable in under ten words.

Russell's paradox, discovered by Bertrand Russell in 1901, showed that the naive set theory created by Georg Cantor leads to a contradiction. This generated a crisis in the foundations of mathematics.

According to naive set theory, any definable collection is a set.

Let R be the set of all sets that are not members of themselves.

If R is a member of itself, it would contradict its own definition as the set containing all sets that are not members of themselves.

If R is not a member of itself, it would qualify as a member of itself by the same definition, again a contradiction.

Give examples of:

1. sets which are not members of themselves,
2. sets which are members of themselves.

The main goal of Hilbert's program was to provide secure foundations for all mathematics. This includes:

- A formalisation of mathematics: all mathematical statements should be written in a precise formal language, and manipulated according to well defined rules (formal system).
- Completeness: a proof that all true mathematical statements can be proved in the adopted formal system.
- Consistency: a proof—preferably reasoning about finite mathematical objects only—that no contradiction can be obtained in the formal system.

- Decidability: construction of an algorithm for deciding, in the formal system, the truth or falsity of any mathematical statement.
- Conservation: a proof that any result about "real objects" obtained using "ideal objects" (such as uncountable sets) can be restated without using ideal objects.

By $N(P, v)$ we mean that the (Turing) program P will *never* halt when begun with input v.

For any particular program P and input v, $N(P, v)$ is a perfectly definite statement which is either true (in case P will never halt in the described situation) or false (in case P will eventually halt).

When $N(P, v)$ is false, this fact can always be demonstrated by running P on v.

No amount of computation will suffice to demonstrate the fact that $N(P, v)$ is true. We may still be able to prove that a particular $N(P, v)$ is true by a logical analysis of P's behaviour, but, as we proved in the undecidability of the halting problem, no such method works correctly in all cases.

What is a *proof*?

Suppose that certain strings of symbols (possibly paragraphs of a natural language (English, for example)) have been singled out—typically with the help of axioms and rules of inference—as proofs of particular statements of the form $N(P, v)$.

Operationally, we assume that we have an algorithm called **syntactic test** that can test an alleged proof Π that $N(P, v)$ is true and determine whether Π is or is not actually such a proof.

There are two basic requirements which it is natural to demand of our supposed rules of proof:

- *Soundness:* If there is a proof Π that $N(P, v)$ is true, then P will in fact never halt when begun with v on its tape.
- *Completeness:* If P will never halt when begun with v on its tape, then there is a proof Π that $N(P, v)$ is true.

Theorem [Gödel's incompleteness theorem]

No rules of proof can be both sound and complete.

Obviously, soundness cannot be sacrificed.

If a given set of rules of proof is sound, then, according to Gödel's incompleteness theorem, there is some true statement $N(P, v)$ which has no proof Π according to the given rules of proof.

Such a true unprovable statement is called *undecidable* since it will surely not be disprovable.

Suppose we had found rules of proof which were both sound and complete.

Suppose "proofs" according to these rules were particular strings of symbols on some specific finite alphabet.

Let $\Pi_1, \Pi_2, \Pi_3, \ldots$ be the quasi-lexicographic computable enumeration of all finite strings on this alphabet. This sequence includes all possible proofs, as well as a lot of other things (including a high percentage of total nonsense). But, hidden among the nonsense, are all possible proofs.

Now we show how we can use our supposed rules of proof to solve the halting problem—an impossibility.

So, we are given a program P and an input v and wish to test whether or not P will eventually halt when begun on v.

We run in parallel two computations:

- the computation of P on v,
- we generate the sequence $\Pi_1, \Pi_2, \Pi_3, \ldots$ of possible proofs and, as each Π_i is generated, we use the **syntactic test** to determine whether or not Π_i is a proof of $N(P, v)$.

If P will eventually halt on v we will find out in finite time.

If P will never halt on v, since our rules of proof are assumed to be complete, there will be a proof Π_i of $N(P, v)$ which we will discover via the enumeration $\Pi_1, \Pi_2, \Pi_3, \ldots$ of possible proofs.

Having obtained this Π_i we will be sure (because of soundness) that P will indeed never halt.

Thus, we have described an algorithm which would solve the halting problem, a contradiction!

Gödel's theorem does not indicate any particular pair P, v for which we will never be able to convince ourselves that $N(P, v)$ is true!

It says that for any given sound rules of proof, there will be a pair P, v for which $N(P, v)$ is true, but not provable *using the given rules.*

There may well be, and in fact there always are, other sound rules which decide this "undecidable" statement. But these other rules will in turn have their own undecidabilities.

Incompleteness and its consequences showed that most of the goals of Hilbert's program were impossible to achieve, at least if interpreted in the "most obvious way".

However, much of it can be salvaged by changing its goals slightly, and with the following modifications some of it was successfully completed:

- Although it is not possible to formalise all mathematics, it is possible to formalise essentially all the mathematics that "anyone uses". Zermelo-Fraenkel set theory, combined with first-order logic, gives a satisfactory and generally accepted formalism for essentially all current mathematics.

- Although it is not possible to prove completeness for systems at least as powerful as Peano arithmetic (if they have a computable set of axioms), it is possible to prove forms of completeness for many interesting systems.
  Gödel proved the completeness theorem for first-order logic before he proved the incompleteness theorems.

- The theory of algebraically closed fields of given characteristic is complete.
- Although there is no algorithm for deciding the truth of statements in Peano arithmetic, Tarski found an algorithm that can decide the truth of any statement in analytic geometry (more precisely, the theory of real closed fields is decidable). Given the Cantor-Dedekind axiom, this algorithm can decide the truth of any statement in Euclidean geometry.

Proof revisited

Proof: 1. a fact or thing that shows or helps to show that something is true or exists; 2. a demonstration of the truth of something, "in proof of my statement"; 3. the process of testing whether something is true or good or valid, "put it to the proof". To prove: to give or be proof of; to establish the validity of; to be found to be, "it proved to be a good theory"; to test or stay out. To argue: 1. to express disagreement, to exchange angry words; 2. to give reasons for or against something, to debate; 3. to persuade by talking, "argued him into going"; 4. to indicate, "their style of living argues that they are well off". Argument: 1. a discussion involving disagreement, a quarrel; 2. a reason put forward; 3. a theme or chain of reasoning. (Oxford American Dictionary)

A critique of the definition of proof

In all these statements, nothing is said about the means used "to show or help to show that something is true or exists", about the means used "in the process of testing whether something is true or good or valid".

In argumentation theory, various ways to argue are discussed, deductive reasoning being only one of them. We use suggestions, impressions, emotions, logic, gestures, mimicry, etc.

Mathematics: informal proofs

An *informal* (pen-on-paper) *proof* is a rigorous argument expressed in a mixture of natural language and formulae (for some mathematicians an equal mixture is the best proportion) that is intended to convince a knowledgeable mathematician of the truth of a statement, the theorem. Routine logical inferences are omitted. "Folklore" results are used without proof. Depending on the area, arguments may rely on intuition. Informal proofs are the standard of presentation of mathematics in textbooks, journals, classrooms, and conferences. They are the product of a social process.

Mathematics: formal proofs

A *formal proof* is written in a formal language consisting of certain strings of symbols from a fixed alphabet. Formal proofs are precisely specified without any ambiguity because all notions are explicitly defined, no steps (no matter how small) are omitted, no appeal to any kind of intuition is made. They satisfy Hilbert's criterion of mechanical testing:

> *The rules should be so clear, that if somebody gives you what they claim is a proof, there is a mechanical procedure that will check whether the proof is correct or not, whether it obeys the rules or not.*

By making sure that every step is correct, one can tell once and for all whether a proof is correct or not, i.e. whether a theorem has been proved.

Mathematical reality

Formal proof cannot be found in mathematical articles or books (except for a few simple examples).

However, most mathematicians believe that almost all "real" proofs, published in articles and books, can, with tedious work, be transformed into Hilbertian proofs. Why?

Because "real" proofs look *convincing* for the mathematical community. Going further, DeMillo, Lipton and Perlis argued that "real proofs" should be highly non-monolithic because they aim to be heard, read, assimilated, discussed, used and generalised by mathematicians—they are part of a social process.

Programming is the activity of solving problems with computers. It includes the following steps:

a) developing the *algorithm* to solve the problem,

b) writing the algorithm in a specific programming language (that is, coding the algorithm into a program),

c) assembling or compiling the program to turn it into machine language,

d) testing and debugging the program,

e) preparing the necessary documentation.

Ideally, at d) one should have written:

d) *proving the correctness of the algorithm*, testing and debugging the program.

We said, ideally, because correctness, although desired, is only practised in very few instances (for example, the programs involved in the proof of the Four-Color Theorem were not proved correct!)

Theorems and programs

It makes sense to prove the correctness of an algorithm, but *not*, the correctness of a program. Programs are analogues of mathematical models, they may be more or less adequate to code algorithms.

Adequacy is a property which depends on many factors, from pure formal/coding ones to physical and engineering ones.

One can even argue that a "correctness proof" for a program, if one could imagine such a thing, adds very little to the confidence in the program. In Knuth's words:

> *Beware of bugs in the above code: I have only proved it correct, not tried it.*

The role of proof in mathematical modelling is very small: *adequacy is the main issue!* Here is an illustration from Jack Schwartz:

> *. . . it may come as a shock to the mathematician to learn that the Schrödinger equation for the hydrogen atom . . . is not a literally correct description of this atom, but only an approximation to a somewhat more correct equation taking account of spin, magnetic dipole, and relativistic effects; that this corrected equation is itself only an ill-understood approximation to an infinite set of quantum field-theoretical equations; and finally that the quantum filed theory besides diverging, neglects a myriad of strange-particle interactions whose strength and form are largely unknown. . . .*

The modelling component of mathematics appears not only in applications, but also in the way mathematics develops new concepts.

Many important notions in mathematics reached an accepted definition only after a long process of modelling, from an intuitive, pre-mathematical notion to a more precisely defined, formal one. In the end, the accepted definition is adopted as a "thesis" claiming its adequacy.

For example, "Weierstrass' thesis" is the statement that the intuitive notion of continuity is extensionally equivalent to the notions yielded by the now standard definitions of continuous function.

Other examples include:

- "The function thesis": identification of a function with a set of ordered pairs,
- "Tarski's thesis": identification of Tarski's definition of truth in a formalised language with the intuitive notion of truth,
- "Church-Turing thesis".

None of these "theses" can be *proved*, but various analyses can conclude their degrees of plausibility/adequacy/applicability. Mathematics in both its practice and development is an "open-texture" .

Checking proofs

There are many "new" types of proofs, probabilistic, experimental or hybrid proofs (computation plus theoretical arguments).

Zeilberger has argued in favour of the transition from rigorous proofs to an "age of *semi-rigorous* mathematics, in which identities (and perhaps other kinds of theorems) will carry price tags" measured in computer and human resources necessary to prove them with a certain degree of confidence.

> *The real work of us mathematicians, from now until, roughly, fifty years from now, when computers won't need us anymore, is to make the transition from human-centric math to machine-centric math as smooth and efficient as possible.*

Communication and understanding

No theorem is validated before it is "communicated" to the mathematical community (orally and, eventually, in writing). Manin:

> *Proof is not just an argument convincing an imaginary opponent. Not at all. Proof is the way we communicate mathematical truth.*

However, as Rota pointed out:

> *One must guard, however, against confusing the presentation of mathematics with the content of mathematics.*

Proofs have to be written on paper, which means proofs are *physical*. From this perspective, proofs depend upon the physical universe (Calude and Chaitin).

Standards of rigour have changed throughout the history of mathematics and not necessarily from less rigour to more rigour.

B. Russell:

> *I wanted certainty in the kind of way in which people want religious faith.*

J.-P. Serre was quoted saying that mathematics is the only producer of "totally reliable and verifiable" truths.

D. Knuth:

> *. . . programming demands a significantly higher standard of accuracy. Things don't simply have to make sense to another human being, they must make sense to a computer.*

W. P. Thurston:

> *The standard of correctness and completeness necessary to get a program to work at all is a couple of orders of magnitude higher than the mathematical community's standard of valid proofs.*
> *When one considers how hard it is to write a computer program even approaching the intellectual scope of a good mathematical paper, and how much greater time and effort have to be put into it to make it "almost" formally correct, it is preposterous to claim that mathematics as we practice it is anywhere near formally correct.*

A conceptual period reaches its maturity under the form of an operational one, which, in its turn, is looking for a new level of conceptual attitude.

The whole treatise of Bourbaki is a conceptual reaction to an operational approach. Dirichlet's slogan asking to replace calculations with ideas should be supplemented with another, complementary slogan, requiring to detect an algorithmic level of concepts.

Can we expect a similar alternation of attitudes in respect to programming? Perhaps it is too early to answer, taking into account that the whole field is still too young. The question is not only academical as the project Flyspeck (to produce a formal proof of the Kepler Conjecture) reminds us.

Proof-assistants

In theory, each informal proof can be converted into a formal proof. However, this is rarely, almost never, done in practice.

A proof assistant (interactive theorem prover) is a software tool to assist with the development of formal proofs by man-machine collaboration.

Proof-assistants can be used not only to check the validity of a formalised proof of a known mathematical result, but also to interactively help to "prove" new theorems.

Hilbert's standard of proof is practicable, it's becoming reality

An impressive record of deep mathematical theorems formally proved:

- Gödel Incompleteness Theorem (1986),
- the Fundamental Theorem of Calculus (1996),
- the Fundamental Theorem of Algebra (2000),
- the Four Colour Theorem (2004),
- Jordan's Curve Theorem (2005),
- the Prime Number Theorem (2008).

The December 2008 issue of the *Notices of AMS* includes four papers on formal proof.

Coq

Produced by INRIA, free distributed under the GNU Lesser General Public Licence, `http://coq.inria.fr/`, Coq is a formal proof management system providing a formal language to write mathematical definitions, executable algorithms and theorems together with an environment for semi-interactive development of machine-checked proofs.

Typical applications include the formalisation of programming languages semantics, formalisation of mathematics (e.g. Four Colour Theorem) and teaching.

Isabelle

Isabelle is a generic proof assistant, free distributed under the BSD license, `http://www.cl.cam.ac.uk/research/hvg/Isabelle/`, allowing mathematical formulas to be expressed in a formal language and provides tools for proving those formulas in a logical calculus.

Isabelle is developed at the University of Cambridge, Technische Universität München and Université Paris-Sud.

The first formal proofs in algorithmic information theory have been developed in Isabelle at UoA (C. Calude and N. Hay).

Data science

George Box: All models are wrong, but some are useful.

Models have been able to consistently, if imperfectly, explain the world around us.

Is there any choice?

Data science

Sixty years ago, digital computers made information readable.

Twenty years ago, the Internet made it reachable.

Ten years ago, the first search engine crawlers made it a single database.

Kilobytes are stored on floppy disks, megabytes are stored on hard disks, terabytes are stored in disk arrays, and petabytes are stored in the cloud. We leave in the *Petabyte Age*.

Data science: Google philosophy

We don't know why this page is better than that one. If the statistics of incoming links say it is, *that's good enough*. *No semantic or causal analysis is required*.

Operationalising this philosophy, Google

- can match ads to content without any knowledge or assumptions about the ads or the content;
- can translate languages without actually "knowing" them; it can translate Maori into Farsi as easily as it can translate French into English provided it has equal corpus data (see ▸ WIKI-LINKS).

Data science: Data deluge

According to ▸ C. ANDERSON, *Wired Magazine*, Google's research director Peter Norvig offered an update to George Box's dictum:

> *All models are wrong, and increasingly you can succeed without them.*

Unfortunately, Norvig ▸ DENIES:

> **That's a silly statement, I didn't say it, and I disagree with it.**

Data science: Data deluge

The big target here is science. The scientific method is built around testable hypotheses. The models are then tested, and experiments confirm or falsify theoretical models of how the world works. This is the way science has worked for hundreds of years.

▸ THE END OF THEORY: THE DATA DELUGE MAKES THE SCIENTIFIC METHOD OBSOLETE.

Welcome to data science! Long live the dead science!

Operationalising the idea

▸ THE END OF THEORY: THE DATA DELUGE MAKES THE SCIENTIFIC METHOD OBSOLETE.

In short, the more we learn about [[biology]], the further we find ourselves from a model that can explain it.

There is now a better way. Petabytes allow us to say: "Correlation is enough." We can stop looking for models. We can analyze the data without hypotheses about what it might show. We can throw the numbers into the biggest computing clusters the world has ever seen and let statistical algorithms find patterns where science cannot.

Data science: Data deluge

▸ NSF : *Computational and Data-Enabled Science and Engineering (CDS & E) is a new program. CDS & E is now clearly recognizable as a distinct intellectual and technological discipline lying at the intersection of applied mathematics, statistics, computer science, core science and engineering disciplines. . . . We regard CDS & E as explicitly recognizing the importance of data-enabled, data-intensive, and data centric science. CDS & E broadly interpreted now affects virtually every area of science and technology, revolutionizing the way science and engineering are done. Theory and experimentation have for centuries been regarded as two fundamental pillars of science. It is now widely recognized that computational and data-enabled science forms a critical third pillar.*

Data science: Data deluge

The "wave of the future"—as it was called—disposes of experiment and theory in science. It affects everything from astronomy to zoology, from medical sciences to social sciences.

History reminds us of similar exaggerations (recall chaos theory or catastrophe theory in mathematics): the "wave of the future" often washes away a number of worthy things and leaves a number of questionable items littering the shore.

Data science: Google cats

In June 2012, Google demonstrated the power of "data-oriented deep learning" with one of the largest neural networks containing more than a billion connections.

A Stanford University–Google team (lead by Andrew Ng and Jeff Dean) showed the system images from 10 million randomly selected YouTube videos. One simulated neuron in the model fixated on images of cats. Others focused on human faces, yellow flowers, and other discrete objects.

The model identified reasonable well these discrete objects even though no humans had ever defined or labeled them.

Data science: Deep learning...

in April 2013 ▸ JEFF DEAN PRESENTATION.

Data science: The good

Imagine there are two papers somewhere in the literature, one of which says that *A implies B*, and another that says *B implies C*. With the incredible growth of the scientific literature, it is likely that these two papers remain unrelated.

A program can find a way to stitch these two papers together, showing that *A implies C*, potentially an important discovery.

Data science: The price for the good

Is there a price in this finding? Cornell University program ▸ EUREQA is a free tool for detecting equations and hidden mathematical relationships in data with the goal "to identify the simplest mathematical formulas which could describe the underlying mechanisms that produced the data".

A theorem may be proven in this way, but no one person actually may understand the proof, though there may be reasons to believe it is correct.

Data science: The price for the good

So what does this all mean for the future of truth?

Is it possible for something to be true but not understandable? Is this bad?

Believing without finding reasons is controversial. However, if these findings motivate the search for more elegantly constructed, human-understandable, versions of these proofs, then they are good.

Data science: The signal problem

Data are assumed to accurately reflect the "real world"; however, significant gaps, with little or no signal coming from particular parts may exist.

Boston has a problem with potholes, patching approximately 20,000 every year. ▸ STREETBUMP smartphone app passively detects and instantly reports potholes to the City. *A clever approach which has a signal problem.* People in lower income groups are less likely to have smartphones, or to use StreetBump and this is particularly true of older residents, where smartphone penetration is as low as 16%.

Smartphone data sets miss inputs from significant parts of the population.

Data science: Ramsey theory or size vs content correlations

Ramsey theory, named after the British mathematician, logician and philosopher Frank P. Ramsey, is a branch of mathematics that studies the conditions under which order must appear.

Problems in Ramsey theory typically ask questions of the form:

> *How many elements of some structure must there be to guarantee that a particular property will hold?*

Data science: Party theorem

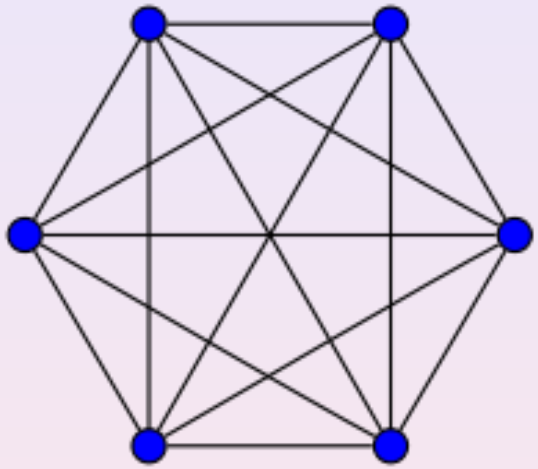
Suppose a party has six people. Consider any two of them.

They might be meeting for the first time—in which case we will call them *mutual strangers*; or they might have met before—in which case we will call them *mutual acquaintances*.

Theorem (Paul Erdös, Alfréd Rényi, Vera T. Sós): *In any party of six people either*

- *at least three of them are (pairwise) mutual strangers or*
- *at least three of them are (pairwise) mutual acquaintances.*

Data science: Party theorem



Party graph

This is the complete graph with six vertices in which every pair of vertices is joined by an edge. Every colouring of edges with red and blue, cannot avoid having either a red triangle or a blue triangle.

Proof. Choose any one vertex; call it P. There are five edges leaving P. They are each coloured red or blue. The *pigeonhole principle* says that at least three of them must be of the same colour.

Let A, B, C be the other ends of these three edges, all of the same colour, say blue. If any one of AB, BC, CA is blue, then that edge together with the two edges from P to the edge's endpoints forms a blue triangle. If none of AB, BC, CA is blue, then all three edges are red and we have a red triangle, namely, ABC.

When a set system is "quite big"?

Given a set X, a collection of subsets $S \subset 2^X$ is called *partition regular* if every set $A \in S$ has the property that, no matter how A is partitioned into finitely many subsets $A = C_1 \cup C_2 \cup \cdots \cup C_n$, at least one of the subsets C_i must belong to the collection S.

The infinite pigeonhole principle. Partition regularity asserts that every finite partition of an infinite set contains an infinite set.
Proof: take S the collection of all infinite subsets of the infinite set.

Data science: Ramsey theory

Ramsey theory proves that

> *Complete disorder is an impossibility. Every large set of numbers, points or objects necessarily contains a highly regular pattern.*

R. Graham, J. H. Spencer. ▸ RAMSEY THEORY, *Scientific American* 262 no. 7 (1990), 112–117.

Data science: Ramsey theory

- How to distinguish correlation from causation?
- How to distinguish content-correlations from Ramsey-type correlations?

Data science books

Two books on data science:

- CALVIN ANDRUS (2012). *DATA SCIENCE: AN INTRODUCTION.*
- JEFFREY M. STANTON (2012). *INTRODUCTION TO DATA SCIENCE.*