Computability in Mathematics-Turing's Legacy

Rod Downey
Victoria University
Wellington
New Zealand

Downey's research is supported by the Marsden Fund, and this material also by the Newton Institute.

Heidelberg, June 2015

Plan

- There have been enormous technical advances in the pure theory on computability and complexity since their roots 90 years ago, and recently we we have seen some rich interactions with various areas of mathematics.
- I plan to outline some recent and not so recent uses of the theory of computation in mathematics.
- To some extent I will concentrate on some of my own contributions, not because I think they are so deep, but because I know them!
- These will be in algebra, algorithmic randomness, and analysis
- This is a "once over lightly" and I will not have time for proofs, and will only cover but a small fraction of the material.

History

- The explicit use of noncomputable methods in mathematics really began in the early 20th century.
- Several people at the time Grete Hermann (field theory), Kronecker (ring and field theory), Dehn (group theory), realized that there were algorithmic questions, for example, the word problem in finitely presented groups.
- von Mises, foundations of randomness theory.
- later, Bishop constructive analysis.
- All lacked a theory of computation.

Origins: Turing

- The equivalent of the "Nobel Prize" in computer science is the ACM Turing Award.
- It is for life work in computer science and worth about $1M.
- Why? This award was made up (1966) was well before Bletchley became public knowledge.
- (Aside) Prof. D. Ritchie (Codebreaker)-from "Station X, Pt 3"

  Alan Turing was one of the figures of the century. —— There were great men at Bletchley Park, but in the long hall of history Turing's name will be remembered as Number One in terms of consequences for mankind.

Logic

- Aristotle and other early Greeks then "modern" re-invention: Leibnitz (early 18th C), Boole, Frege, etc.
- Beginning of logic's treatment as language as data. Logic is the only part of mathematics taking language seriously.
- Simplest modern formal system propositional logic.
- Represent statements which are either possibly true or false.
- If You attend this lecture then you will know some logic. You attend this lecture. Therefore you know some logic.
- $((A \to K) \wedge A) \to K$.
- Question Given such a formula, can we decide if it is true or false no matter what we put in for the variables? (e.g. maybe you don't attend the lecture; the value of K is 0=False)

Decision Problems I

p	q	$p \wedge \neg(q \vee p)$	$(p \to q)$	$\to$	$\neg$	$(p \vee \neg q)$
1	1	0	1	0	0	1
1	0	0	0	1	0	1
0	1	0	1	1	1	0
0	0	0	1	0	0	1
				↑		

p	q	r	$(p$	$\to$	$(q \wedge r))$	$\to$	$(\neg r \to \neg p)$
1	1	1	1	1	1	1	1
1	1	0	0	0	0	1	0
1	0	1	0	0	0	1	1
1	0	0	0	0	0	1	0
0	1	1	1	1	1	1	1
0	1	0	1	0	0	1	1
0	0	1	1	0	0	1	1
0	0	0	1	0	0	1	1
					↑		

Want to earn some money?

- Notice that in the above each time we add a new variable we double the length of the table.
- One of the most important, if not the most important problems in mathematics/computer science,
- Is there any fast (polynomial time) way to figure out if there is a "true" line of the truth table? Or is essentially trying all possible ways the only way? P vs NP, \$1M Clay Prize.
- Note, with 100,000 variables, the time needed for such a search is approx $2^{100,000}$ estimated to be larger than the number of atoms in the universe.
- Note if you do find a fast way, please tell me. You will kill all modern banking security, and make millions of tasks exponentially easier. Revolutionize modern science and society.
- This is because we have ways of reducing many computation tasks to solving "satisfiability".
- Note also that all modern public cryptosystems work by arguing that "exponential searches are needed."

Predicate Logic

- Propositional Logic is too limited for many tasks, and we developed a richer logic using quantifiers
- $\forall$, $\exists$, etc.
- Remark that there are many logics and are the core of e.g. programme verification etc.

Hilbert-Decision Problems II

- David Hilbert, 1900, asked for a decision procedure (like for propositional logic) for Predicate Logic.
- This is the (now) famous Entscheidungsproblem.
- I remark that in fact he believed that there was such a thing.

David Hilbert, 1912 — one of a group of portraits of professors which were sold as postcards in Göttingen

More than Mathematics

- Realize that it is not enough to try for a long time and find no method. Perhaps this is simply lack of brain power.
- What is being asked is to prove there is no method ; or to give one.
- To Prove NO The first thing you must do is to give something that models all such methods.
- Then prove that that model won't do it.
- Realize that the first part is essentially philosophical.
- How to model human thought? Or at least "human decision procedures"?

The confluence of ideas in 1936

- First Church, then Kleene, Turing and Post proposed models for decision procedures.
- We now know that all the proposed models are provably equivalent so that technically Church and Kleene first showed that that Entscheidungsproblem is undecidable.
- Church proposed his thesis that the λ-definable functions (too horrible to describe), and later partial recursive functions modeled all effectively computable processes.
- Post a Turing machine like model I torture 3rd years students with.
- Turing : Turing machine.
- At 24, whilst on a run at Grantchester Meadows, Turing devised a brilliantly convincing model.

Turing Machine

- Machine is a Box with a finite number of Internal States (i.e. mental states)
- Reads/writes on a two way potentially infinite tape.
- Action : can move Left, Right, or Print a symbol,
- Depending on (state, symbol)
- Here's the Yellow Brick Road
- This person is a simple person has two states : happy and unhappy.
- He's happy when he sees a yellow brick
- $\langle$ happy, $\square$, $\square$, unhappy$\rangle$, $\langle$ unhappy, $\square$, Y, happy $\rangle$, $\langle$ happy, Y, L , happy $\rangle$.
- What will this do?

Kleene Partial Recursive Functions

The **partial recursive function** are the smallest class of $\mathcal{C}$ of functions which are closed under the schemes (simplified version):

1. Zero function
$$Z(x) = 0$$
2. Successor function
$$S(x) = x + 1$$
3. Predecessor function (monus)
$$P(x) = x \hat{-} 1 = \begin{cases} x - 1 & x > 0 \\ 0 & x = 0 \end{cases}$$
4. Projection
$$P_j^m(x_1, \ldots, x_m) = x_j$$
5. Substitution

$$\text{If } f(x) \in \mathcal{P} \text{ and } g(x) \in \mathcal{P}, \text{ then } f(g(x)) \in \mathcal{P}$$

Not finished yet

6 Recursion

$$\text{If } g(\vec{x}, y) \in \mathcal{P} \text{ and } h(\vec{x}) \in \mathcal{P}, \text{ then } f(\vec{x}) \in \mathcal{P}$$
$$\text{where } f(0) = h(0)$$
$$f(n+1) = g(n, f(n))$$

7 Least number
If $g(\vec{x}, y) \in \mathcal{C}$ then $f(\vec{x}) \in \mathcal{C}$, where
$f(\vec{x}) = \mu y[(g(\vec{x}, y) \downarrow = 0) \text{ and } \forall z \leq y \; g(\vec{x}, y) \downarrow]$

I put the above model to show you that there is no obvious intuitive reason that the model above "captures all decision procedures." This is the genius of the Turing model.

Why Turing?

- Turing shows that a simple problem ("The halting problem") that can't be decided by the model.
- The Halting Problem problem is expressible in predicate logic. Eureka!
- The earlier proofs of Church etc not accepted at the time. See e.g. Davis, Gandy 1995, Soare 2012, Kleene 1995.
- First and foremost Turing has a conceptual analysis giving what many regard as a proof of the Church-Turing Thesis that TM's capture what is computable be a person.
- This analysis is the fundamental contribution of Turing's paper. (also a basis of "hard AI")
- See "The Universal Turing Machine: A Half Century Survey" R. Herken (ed) Springer 1995 (2nd Ed).

Turing's analysis

- He considers an abstract human computor (1950's terminology)
- By limitations of sensory and mental apparatus we have
  (i) fixed bound for the symbols.
  (ii) fixed bound for number of squares
  (iii) fixed bound to the number of actions at each step
  (iv) fixed bound on the movement.
  (v) fixed bound on the number of states.
- This justifies TM's
- Gandy, Soare (and others) argue that Turing proves any function calculable by an abstract human is computable by a TM.

- Gandy (1995):

> *What Turing did, by his analysis of the processes and limitations of calculations of human beings, was to clear away, with a single stroke of his broom, this dependency on contemporary experience, and produce a characterization-within clearly perceived limits- which will stand for all time..... What Turing also did was to show that calculation can be broken down into the iteration (controlled by a "program") of extremely simple concrete operations; so concrete that they can easily be described in terms of (physical) mechanisms.*

(My emphasis)

The Universal Machine

- The other major contribution was the notion of a universal machine, a compiler.
- Turing has the first universal machine. The idea that there could be a single machine which interpreted programs to emulate any other machine.
- This revolutionary idea is the conceptual key to computers.

Post-history

- The idea that a computer could be universal was a long time penetrating.
- Howard Aitken (1956), a US computer expert of the time:

  > *If it should turn out that the basic logics of a machine designed for numerical solution of differential equations coincide with the logics of a machine intended to make bills for a department store, I would regard this as the most amazing coincidence that I have ever encountered.*

- Read more on this in Martin Davis' or Herken's books.

In Turing's words

Turing said in a lecture of 1947 with his design of ACE (automated computing engine)

> *The special machine may be called the universal machine; it works in the following quite simple manner. When we have decided what machine we wish to imitate we punch a description of it on the tape of the universal machine... . The universal machine has only to keep looking at this description in order to find out what it should do at each stage. Thus the complexity of the machine to be imitated is concentrated in the tape and does not appear in the universal machine proper in any way... . [D]igital computing machines such as the ACE ... are in fact practical versions of the universal machine.*

Reductions

- So we can enumerate the machines: $\{\varphi_x \mid x \in \mathbb{N}\}$.
- The Halting Problem : Input n, m:
  Question Does $\varphi_n(m)$ halt?
  is algorithmically undecidable.
- The principle idea is now to use this with reductions to show that other problems undecidable.
- $A \leq_T B$ means that with B as an oracle I can solve A.
- Think of reducing the question of a matrix being invertable to the computation of the determinant.

- Down thru the years many examples of problems of the same complexity as the halting problem.
- Hilbert's 10th Problem (Matiyasevich)
- Word problems in groups (Novikov-Boone)
- Homeomorphism problems in 3 space (Reubel)
- more recently DNA self assembly (Adelman, Lutz)
- boundaries of Julia Sets (Braverman, Yampolsky)
- Some general meta-theorems, e.g. Rice's Theorem, Markov Properties.

An application to differential geometry

- Sometimes what is needed is more intricate understanding of (c.e.) computably enumerable (Σ^0_1) sets for an application.
- The c.e. sets and their "degrees of unsolvability" each form extremely complex structures.
- For example, the c.e. degrees form a dense upper semilattice, where all distributive and some nondistributive lattice embed, but not all, and the question of which embed is still open after 60 years.
- At Chicago, Soare provided the computability needed for "settling times" of families of c.e. sets, for work on Riemannian metrics on a smooth manifold under reparameterization.
- This is denoted by $\mathrm{Met}(M) = \mathrm{Reim}(M)/\mathrm{Diff}(M)$.

Theorem (Nauktovsky and Weinberger-Geometrica Dedicata)

For every closed smooth manifold M of dimension $n > 4$, there are infinitely many local minimal of the diameter functional of the subset of $Met(M)$ consisting of the isometry classes of Riemannian metrics with curvature bounded in absolute value by 1. The minima are represented by Riemannian metrics of smoothness $C^{1,\alpha}$ for any $\alpha \in [0, 1)$. There is a constant $c(n)$ depending only on n, such that for any c.e. degree $\mathbf{b}$, the local minima of depth at least $\mathbf{b}$, are $\mathbf{b}$-dense in a path metric of the isometry classes, and the number of such mimina where the depth does not exceed d is not less that $e^{c(n)d^n}$.

But does it matter?

- Various approaches to quantify the fact that undecidability is rare in practice.
- For example most group theoretical questions for finitely presented groups are generically decidable. (Karpovich, Myasnikov, Schupp and Shpilrain)
- Here one asks for an algorithm which is always right, but only halts on a set of Borel density 1.
- Similar questions arise about NP completeness. Why do SAT SOLVERS work so well?
- What is the topology of hard instances of real life problems... e.g. Parameterized complexity

No Invariants

- What is the use of an invariant, like e.g. dimension, Ulm invariants, etc.
- Arguably, they should make a classification problem easier.
- For example, one invariant for isomorphism type of a class of structures e.g. vector spaces over $\mathbb{Q}$ is the isomorphism type, but that's useless.
- We choose dimension as it completely classifies the type.
- How to show NO invariants?
- We give one answer in the context of computable mathematics, and mention some other approaches using logic.

Computable mathematics

- Mal'cev 1962 A computable abelian group is computably presented if we have $G = (G, +, 0)$ has $+$ and $=$ computable functions/relations on $G = \mathbb{N}$.
- When can an abelian group be computably presented? (Relative to an oracle) Is there any reasonable answer?
- Do different computable presentations have different computable properties?
- Mal'cev produced examples presentations of $\mathbb{Q}^\infty$ that were not computably isomorphic, as we see later.
- Along with Rabin and Frölich and Shepherdson, began the theory of presentations of computable structures, though arguably back to Emmy Noether, Kronecker as recycled in van der Waerden (first edition).
- See Matakides and Nerode "Effective Content of Field Theory".

Why should we care?

- If we are interested in actual processes on algebraic structures then surely we need to understand the extent to which they are algorithmic.
- Effective algorithmics requires more detailed understanding of the model theory. Witness the resurrection of the study of invariants despite Hilbert's celebrated "destruction" of the programme.
- The Hilbert basis (or nulstellensatz) theorem(s) are fine, but suppose we need to calculate the relevant basis.
- Examples of this include the whole edifice of combinatorial group theory. The theory of Gröbner bases etc. New constructions in combinatorics, algebra, etc.
- As we will see a backdoor into establishing classical results about the existence/nonexistence of invariants in mathematics. Computability is used to establish classical result.
- Establishing calibrations of complexity of algebraic constructions.... reverse mathematics.

While we are on the subject of logic

- Thanks to Moshe Vardi for this and the next quote (my highlighting).
- Cosma R. Shalizi, Santa Fe Institute (A famous US think-tank).

> *If, in 1901, a talented and sympathetic outsider had been called upon (say by a granting agency) to survey the sciences an name a branch that would the the least fruitful in the century ahead, his choice might well have settled upon mathematical logic, and exceedingly recondite field whose practitioners could all have fit into a small auditorium. It had no practical applications, and not even that much mathematics to show for itself: its crown was an exceedingly obscure definition of cardinal numbers.*

More recently

- Martin Davis (1988) Influences of mathematical Logic on Computer Science.

  When I was a student, even the topologists regarded mathematical logicians as living in outer space. Today the connections between logic and computers are a matter of engineering practice at every level of computer organization.

- Yuri Gurevich (Microsoft) quoted as saying engineers need logic not calculus!
- Read a somewhat dated but wonderful collection in the Bulletin of Symbolic Logic: On the Unusual Effectiveness of Logic in Computer Science (Halpern, Harper, Immerman, Kolaitis, and Vardi).
- Echoes Wigner's 1960 article "The unreasonable effectiveness of mathematics in the natural sciences," and Galileo's "The book of nature is writ in the language of mathematics."

Σ_1^1-completeness?

- The halting problem is Σ_1^0. This means it can be described by an existential quantifier on numbers around a computable predicate. "There is a stage s where the e-th machine with input y halts in at most s steps"
- Showing that a problem A is Σ_1^0 complete means that there is a computable f such that for each instance I of a Σ_1^0 problem B, I can compute $f(I)$ which is an instance of A such that I is a yes for B iff $f(I)$ is a yes for A. A is the "most complex" Σ_1^0 problem.
- If a problem can be expressed as a finite number of alternations of number quantifiers, it is called arithmetical, "Δ_n^0" for some n.
- For example: is φ_x total? provably needs an alternation of quantifiers.

- Some problems are too complex for this. Classical isomorphism of infinite structures: "There is a function such that ...."
- If we allow function quantifiers, we put a "1" on top.
- Note now we are searching through the uncountably many possible functions $f : \mathbb{N} \to \mathbb{N}$.

$$\exists f \forall x, y (f(x + y) = f(x) + f(y)).$$

Consequences of Σ^1_1-completeness

- The idea of an invariant is that is ought to make the problem simpler.
- Classical isomorphism is always Σ^1_1.
- Invariants make this easier, you would expect. Dimension in a vector space makes the problem Δ^0_3.
- The point is that a Σ^1_1-completeness result result means that the cannot be reasonable invariants for the isomorphism problem.

Torsion-free abelian groups

- In general the isomorphism problem is very complex:

Theorem (Downey and Montalbán-J. Algebra)

The isomorphism problem for torsion-free abelian groups is Σ^1_1 complete.

- That is a computational "proof" that there cannot be invariants.
- As explained in the DM paper, group theorists try to understand finitely presented groups via spectral sequences, one called the integral homology sequence (Stallings etc)
- The above result, combined with one of Baumslag, Dyer and Miller shows that deciding if two finitely presented groups have the same 3rd members of this sequence is already Σ^1_1 complete!!
- This methodology understands invariant theory computationally.
- Also used by Slaman and Woodin, Friedman, Stanley, Knight and others in many other settings.
- There are other programmes like this as we now will see.

The Borel game

- This is related to work by the descriptive set theorists (a higher level of computability sharing techniques) who seek to have a notion of Borel cardinality for isomorphism types.
- One class $\mathcal{C}$ is reducible to another $\mathcal{D}$ if there is a Borel mapping injectively taking the isomorphism types of $\mathcal{C}$ into $\mathcal{D}$.
- For example, rank 3 torsion free groups are above rank 2 groups here.
- H. Friedman, Kechris, Thomas, Hjorth etc.
- Also miniaturized recently by Knight and her co-authors.
- A last approach pioneered by Shelah using classification theory.

Better algebraic classes

- Okay, so the general classification problem is intractable, then can we measure how complex the classically "understood" classes are.
- Recall that if G is a torsion-free then G embeds into $\oplus_{i \in F}(\mathbb{Q}, +)$. The cardinality of the least such F is called the (Prüfer) rank of G.
- Khisamiev proved that there is an effective embedding. (That is if G is a computable torsion-free abelian group then G can be computably embedded into a computable copy of $\oplus_{i \in F}(\mathbb{Q}, +)$.

Rank One Groups

- The only groups we understand well are the rank one groups (and certain mild generalizations) If $g \in G$, define $t(g) = (a_1, a_2, \dots)$ where $a_i \in \{\infty\} \cup \omega$ and represents the maximum number of times p_i divides g. Say that $t(g) = t(h)$ if they are $=^*$, meaning that they must be ∞ in the same places, but otherwise are finitely often different. Thus we can write $t(G)$.
- For example, a divisible group would have $(\infty, \infty, \dots)$ as its type.

Theorem (Baer, Levi)

For rank 1 torsion-free abelian groups, $G \cong H$ iff they have the same type.

- One corollary is that if we consider $T(G) = \{\langle x, y\rangle \mid x \leq t(G)_y\}$, then G is computably presentable iff $T(G)$ is Σ^0_1. (Mal'tsev)

Two Corollaries

- G is a computably categorical torsion-free abelian group iff it has finite rank.
- A structure is called computably categorical iff any two computable copies are computably isomorphic.

Definition

A structure $\mathcal{A}$ has a degree iff $\min\{\deg(\mathcal{B}) \mid \mathcal{B} \cong \mathcal{A}\}$ exists.

- Strictly speaking, we would mean the isomorphism type here.
- (Jockusch) Can define jump degree by replacing $\deg(\mathcal{B})$ by $\deg(\mathcal{B})'$. The same for α-th jump degree. Proper if no β-th jump degree for $\beta < \alpha$. The "jump" of a set is the halting problem with the set as an oracle.
- (Coles, Downey and Slaman-Bull LMS) Every torsion free abelian group of finite rank has first jump degree.
- (Anderson, Kach, Melnikov, Solomon-APAL) For each computable α and $\mathbf{a} > \mathbf{0}^\alpha$ there is a torsion-free abelian group with proper α-th jump degree $\mathbf{a}$.

The infinite rank case

- It could be hoped that if G has infinite rank, then $G \cong \oplus_{i \in \omega} H_i$ with H_i of rank one.
- Alas, this is not true, however, there is a class of groups for which this is true, called completely decomposable for which this does happen.
- What about categoricity for such groups?
- We cannot hope for computable categoricity, but can hope for things "higher up".

The homogeneous case

- If $G \cong \oplus H$ for a fixed H then G is called homogeneous

Theorem (Downey and Melnikov-J. Algebra)

Homogeneous computable torsion free abelian groups are Δ^0_3 categorical.

- The proof relies on a new notion of independence called S-independence generalizing a notion of Fuchs to sets S of primes.
- B, a set of elements, is S-independent (in G) iff for all $p \in S$ and $b_1, \ldots, b_k \in G$,

$$p | \sum_{i=1}^{k} m_i b_i \text{ implies } p | m_i \text{ for all } i.$$

- This bound is tight.

But when can it be Δ^0_2 categorical?

- Recall that a set S is called semilow if $\{e \mid W_e \cap S \neq \emptyset\} \leq \emptyset'$.
- Semilow sets allow for a certain kind of local guessing, and arose in (i) automorphisms of the lattice of computably enumerable sets (Soare) and in (ii) computational complexity as non-speedable ones. (Soare, Blum-Marques, etc.)

Theorem (Downey and Melnikov-J. Algebra)

G is Δ^0_2 categorical iff the type of H consists of only 0's and ∞'s and the position of the 0's is semilow.

- The proof is tricky and splits into 5 cases depending on "settling times".
- We remark that this is one of the very few known examples of when Δ^0_2 categoricity of structures has been classified.

The general completely decomposable case

Theorem (Downey and Melnikov-TAMS)

A completely decomposable G is Δ^0_5 categorical. The bound is tight.

The proof uses methods from the homogeneous case, plus some new ideas. The sharpness is a coding argument. For sharpness we use copies of $\oplus_{i\in\omega}\mathbb{Z} \oplus \oplus_{i\in\omega}\mathbb{Q}^{(p)} \oplus \oplus_{i\in\omega}\mathbb{Q}^{(q)}$., where $p \neq q$ primes and $\mathbb{Q}^{(r)}$ denotes the additive group of the localization of $\mathbb{Z}$ by r. Then a relation θ on this group which is decidable in one copy and very bad in another.
With some extra work we can also prove the following. We don't know if the bound is sharp here.

Corollary (Downey and Melnikov-TAMS)

The index set of completely decomposable groups is Σ^0_7.

Algorithmic Randomness

- Can we give meaning to the notion of an individual random sequence?
- The idea is to use computability theory to define a real α to be random if no computable betting strategy (computable martingale) can succeed in making infinite capital.
- Can use simple martingales effective functions $f : 2^\omega \to \mathbb{R}^+ \cup \{0\}$,

$$f(\sigma) = \frac{f(\sigma 0) + f(\sigma 1)}{2}.$$

- Large amount of theory, and we know this correlates to e.g. initial segments being incompressible (Levin), or the real having no computably rare properties (Matin-Löf).
- The analog of the halting set is Chaitin's halting probability Ω, the Lebesgue measure of the collection of strings (names or programmes) which halt.

$$\Omega = \sum_{U(\sigma)\downarrow} 2^{-|\sigma|}.$$

A couple of recent theoretical advances

- Randoms split into two classes, "false randoms like Ω which can compute the halting problem", and those that are "really random and provable stupid in a technical sense."
- Using computable martingales s-gales with a weighting s to give a meaning to the effective Hausdorff dimension of an individual real. (Lutz, Schnorr)

- (Symbolic Dynamics) A d-dimensional *shift* of finite type is a collection of colourings of $\mathbb{Z}^d$ defined by local rules and a shift action (basically saying certain colourings are illegal). Its (Shannon) *entropy* is the asymptotic growth in the number of legal colourings.
- More formally, consider $G = (\mathbb{N}^d, +)$ or $(\mathbb{Z}^d, +)$, and A a finite set of symbols. We give A the discrete topology and A^G the product topology. The shift action of G on A^G is

$$(S^g x)(h) = x(h + g), \text{ for } g, h \in G \wedge x \in A^G.$$

A *subshift* is $X \subseteq A^G$ such that $x \in X$ implies $S^g x \in X$ (i.e. shift invariant).

Theorem (Simpson-J Ergodic Theory)

If X is a subshift (closed and shift invariant), then the effective Hausdorff dimension of X is equal to the classical Hausdorff dimension of X is equal to the entropy, moreover there are calculable relationships between the effective and classical quantities. (See Simpson's home page for his recent talks and more precise details.)

- Simpson use this to give new elementary proofs and then extensions to difficult results of Furstenberg.

Some other applications

- Braverman and Yampolsky showed that the halting probabilities correspond exactly to Julia sets from effective initial conditions. (JAMS)
- In symbolic dynamics again:

Theorem (Hochman and Meyerovitch, Ann Math)

The values of entropies of subshifts of finite type over $\mathbb{Z}^d$ for $d \geq 2$ are exactly the complements of halting probabilities.

- Recently Day used a similar method to give a very short proof of the Kolmogorov-Sinai theorem relating Shannon entropy to equivalence of Bernoulli systems from Ergodic theory.

Computable Analysis

- Roots go back to Turing's original paper!

Definition

f is computable it there is a uniform algorithm taking fast converging Cauchy sequences (i.e. $q_k \in B(q_n, 2^{-n})$ for all $k > n$) to fast converging Cauchy sequences.

(This is in. e.g. a separable metric space with a countable computable base, like the reals and the base $\mathbb{Q}$.)

- In fact "f continuous" $=$ "f computable relative to an oracle".

Theorem (Pour-E and Richards)

In this setting an operator is computable iff it is bounded.

- Thus there is a computable ode with computable initial conditions but no computable solution. (Myhill)

- Some things are perhaps surprisingly computable. For example, the graph G of a computable function on a closed interval is computable as a set, in the sense that the distance function $d(x, G)$ is computable from it.
- It is also possible to look at effective L^p-computability....

Derivatives

- If you look at e.g. the Dini derivative the correct way, then it looks like a Martingale. This observation was by Demuth. A

Theorem (Demuth-Brattka, Miller, Nies-TAMS)

A computable f of bounded variation is differentiable at each Martin-Löf random set, and this is tight.

- That is "differentiable=random"!!
- Westrick has shown that the differentiation/continuity hierarchy aligns exactly with the arithmetic/analytic hierarchy.
- Lots of new work on computational aspect of Ergodic Theorems, Brownian motion and the like. But no time.

Thank You