Benign approximations, superspeedability, and randomness

Rupert Hölzl
Universität der Bundeswehr München

Based on joint works with Peter Hertling, Philip Janicki, Wolfgang Merkle, and Frank Stephan

1

Motivation

Left-computable and computable reals

1. **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

Left-computable and computable reals

1 **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

Left-computable and computable reals

1 **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

Left-computable and computable reals

1 **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

Left-computable and computable reals

1 **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

Left-computable and computable reals

1 **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

Left-computable and computable reals

1 **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

Left-computable and computable reals

1 Definition. A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

Left-computable and computable reals

1 **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

Left-computable and computable reals

1 **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

Left-computable and computable reals

1 **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

1 **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

Left-computable and computable reals

1 **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

Left-computable and computable reals

1 **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

Left-computable and computable reals

1 **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

Left-computable and computable reals

1 **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

Left-computable and computable reals

1 **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

2 The reason why this is not the same as *computable* is of course that there may be the type of "fake convergence" seen above.

Left-computable and computable reals

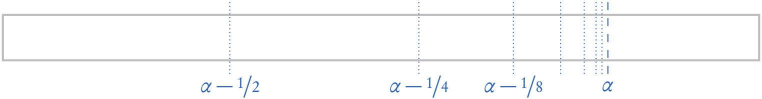
1. **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

2. The reason why this is not the same as *computable* is of course that there may be the type of “fake convergence” seen above.
3. **However:** The limit α *does* become computable if in the n-th approximation step we are guaranteed to be 2^{-n}-close to it:

Left-computable and computable reals

1. **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

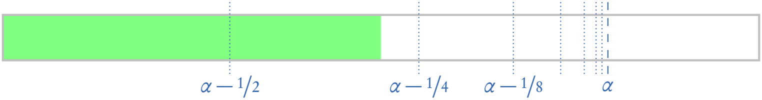
2. The reason why this is not the same as *computable* is of course that there may be the type of “fake convergence” seen above.
3. **However:** The limit α *does* become computable if in the n-th approximation step we are guaranteed to be 2^{-n}-close to it:



Left-computable and computable reals

1. **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

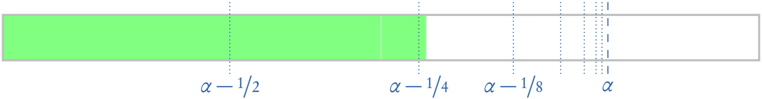
2. The reason why this is not the same as *computable* is of course that there may be the type of "fake convergence" seen above.
3. **However:** The limit α *does* become computable if in the n-th approximation step we are guaranteed to be 2^{-n}-close to it:



Left-computable and computable reals

1. **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

2. The reason why this is not the same as *computable* is of course that there may be the type of "fake convergence" seen above.
3. **However:** The limit α *does* become computable if in the n-th approximation step we are guaranteed to be 2^{-n}-close to it:

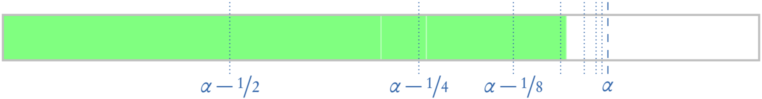


Left-computable and computable reals

1. **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

2. The reason why this is not the same as *computable* is of course that there may be the type of “fake convergence” seen above.

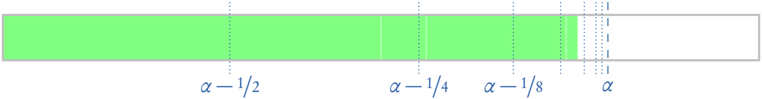
3. **However:** The limit α *does* become computable if in the n-th approximation step we are guaranteed to be 2^{-n}-close to it:



Left-computable and computable reals

1. **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

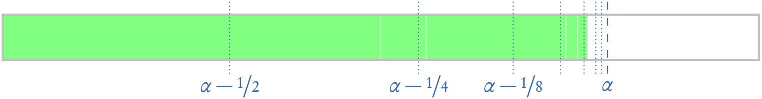
2. The reason why this is not the same as *computable* is of course that there may be the type of "fake convergence" seen above.
3. **However:** The limit α *does* become computable if in the n-th approximation step we are guaranteed to be 2^{-n}-close to it:



Left-computable and computable reals

1. **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

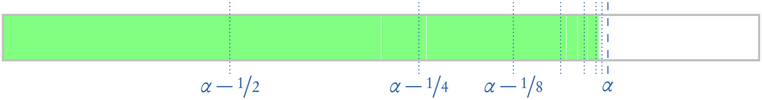
2. The reason why this is not the same as *computable* is of course that there may be the type of "fake convergence" seen above.
3. **However:** The limit α *does* become computable if in the n-th approximation step we are guaranteed to be 2^{-n}-close to it:



Left-computable and computable reals

1. **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

2. The reason why this is not the same as *computable* is of course that there may be the type of "fake convergence" seen above.
3. **However:** The limit α *does* become computable if in the n-th approximation step we are guaranteed to be 2^{-n}-close to it:

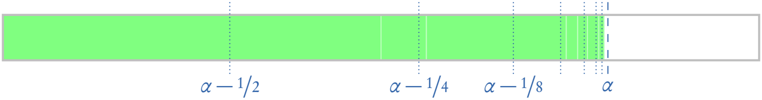


Left-computable and computable reals

1. **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

2. The reason why this is not the same as *computable* is of course that there may be the type of "fake convergence" seen above.

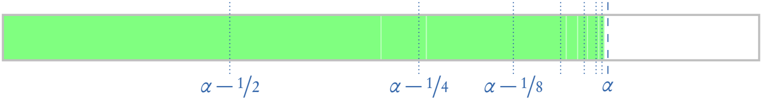
3. **However:** The limit α *does* become computable if in the n-th approximation step we are guaranteed to be 2^{-n}-close to it:



Left-computable and computable reals

1. **Definition.** A real number α is *left-computable* if there exists a computable, *increasing* sequence of rationals converging to it.

2. The reason why this is not the same as *computable* is of course that there may be the type of “fake convergence” seen above.
3. **However:** The limit α *does* become computable if in the n-th approximation step we are guaranteed to be 2^{-n}-close to it:



4. So far, so trivial.

Overview of the talk

1. We will only work inside the left-computable numbers.
2. We are interested in numbers having “benign” approximations
   - that are a little friendlier than just left-computable
   - but without (necessarily) implying computability.

Overview of the talk

1. We will only work inside the left-computable numbers.
2. We are interested in numbers having “benign” approximations
   - that are a little friendlier than just left-computable
   - but without (necessarily) implying computability.
3. We present three notions of such approximations, discuss some properties, and answer a question posed by Merkle and Titov.

Overview of the talk

1. We will only work inside the left-computable numbers.
2. We are interested in numbers having "benign" approximations
   - that are a little friendlier than just left-computable
   - but without (necessarily) implying computability.
3. We present three notions of such approximations, discuss some properties, and answer a question posed by Merkle and Titov.
4. We answer a question of Barmpalias about "uniformity," leading to a fourth notion of benign approximation.

Overview of the talk

1. We will only work inside the left-computable numbers.
2. We are interested in numbers having “benign” approximations
   - that are a little friendlier than just left-computable
   - but without (necessarily) implying computability.
3. We present three notions of such approximations, discuss some properties, and answer a question posed by Merkle and Titov.
4. We answer a question of Barmpalias about “uniformity,” leading to a fourth notion of benign approximation.
5. Finally, we inquire into the relationship with randomness.

2

Three types of benign approximations

Speedability

1. **Definition (Merkle & Titov).** α is *speedable* if there is
   - a $\rho \in (0,1)$ and
   - a computable left-approximation $(a_n)_n$ of α

   such that there are infinitely many $n \in \mathbb{N}$ with

$$\frac{a_{n+1} - a_n}{\alpha - a_n} \geq \rho.$$

(Merkle & Titov used a different, but equivalent formulation.)

Speedability

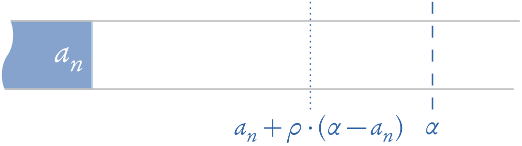
1 Definition (Merkle & Titov). α is *speedable* if there is

- a $\rho \in (0,1)$ and
- a computable left-approximation $(a_n)_n$ of α

such that there are infinitely many $n \in \mathbb{N}$ with

$$\frac{a_{n+1} - a_n}{\alpha - a_n} \geq \rho.$$

(Merkle & Titov used a different, but equivalent formulation.)



Speedability

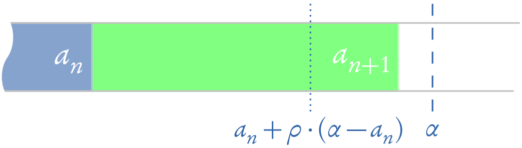
1 **Definition (Merkle & Titov).** α is *speedable* if there is

- a $\rho \in (0,1)$ and
- a computable left-approximation $(a_n)_n$ of α

such that there are infinitely many $n \in \mathbb{N}$ with

$$\frac{a_{n+1} - a_n}{\alpha - a_n} \geq \rho.$$

(Merkle & Titov used a different, but equivalent formulation.)



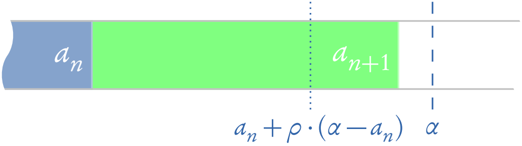
Speedability

1. **Definition (Merkle & Titov).** α is *speedable* if there is
   - a $\rho \in (0,1)$ and
   - a computable left-approximation $(a_n)_n$ of α

   such that there are infinitely many $n \in \mathbb{N}$ with

$$\frac{a_{n+1} - a_n}{\alpha - a_n} \geq \rho.$$

   (Merkle & Titov used a different, but equivalent formulation.)



2. **Theorem (Merkle & Titov).** Any $\rho \in (0,1)$ works equally.

   (But you need to nonuniformly replace the approximation by another one.)

Speedability & randomness

1. **Theorem (Merkle & Titov; implicit in Barmpalias & Lewis-Pye).** No Martin-Löf random can be speedable.

Speedability & randomness

1. **Theorem (Merkle & Titov; implicit in Barmpalias & Lewis-Pye).** No Martin-Löf random can be speedable.
2. **Question (Merkle & Titov).**
   - Does the inverse hold?
   - **That is:** Among the left-computables, are the randoms characterized by their non-speedability?

Approximations that catch up

1. While thinking about this, Philip and myself stumbled across another notion of benign approximation.

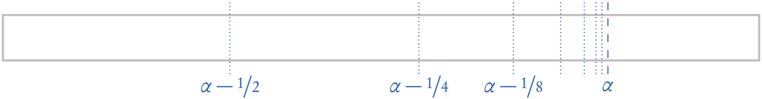
Approximations that catch up

1. While thinking about this, Philip and myself stumbled across another notion of benign approximation.
2. **Definition.** α is *regainingly approximable* if there is
   - a computable left-approximation $(a_n)_n$ of α
   - with $\alpha - a_n < 2^{-n}$ for infinitely many $n \in \mathbb{N}$.

Approximations that catch up

1. While thinking about this, Philip and myself stumbled across another notion of benign approximation.

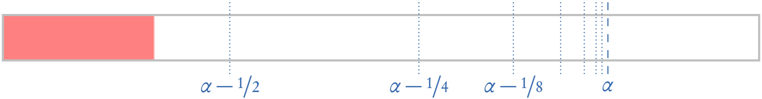
2. **Definition.** α is *regainingly approximable* if there is
   - a computable left-approximation $(a_n)_n$ of α
   - with $\alpha - a_n < 2^{-n}$ for infinitely many $n \in \mathbb{N}$.



Approximations that catch up

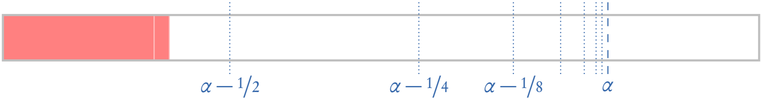
1. While thinking about this, Philip and myself stumbled across another notion of benign approximation.

2. **Definition.** α is *regainingly approximable* if there is
   - a computable left-approximation $(a_n)_n$ of α
   - with $\alpha - a_n < 2^{-n}$ for infinitely many $n \in \mathbb{N}$.



Approximations that catch up

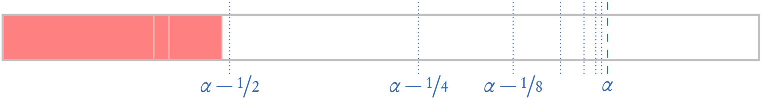
1. While thinking about this, Philip and myself stumbled across another notion of benign approximation.
2. **Definition.** α is *regainingly approximable* if there is
   - a computable left-approximation $(a_n)_n$ of α
   - with $\alpha - a_n < 2^{-n}$ for infinitely many $n \in \mathbb{N}$.



Approximations that catch up

1. While thinking about this, Philip and myself stumbled across another notion of benign approximation.

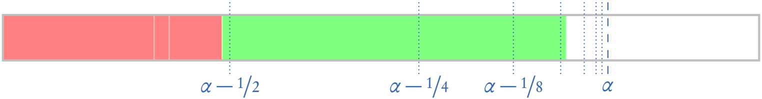
2. **Definition.** α is *regainingly approximable* if there is
   - a computable left-approximation $(a_n)_n$ of α
   - with $\alpha - a_n < 2^{-n}$ for infinitely many $n \in \mathbb{N}$.



Approximations that catch up

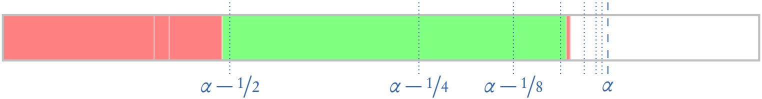
1. While thinking about this, Philip and myself stumbled across another notion of benign approximation.

2. **Definition.** α is *regainingly approximable* if there is
   - a computable left-approximation $(a_n)_n$ of α
   - with $\alpha - a_n < 2^{-n}$ for infinitely many $n \in \mathbb{N}$.



Approximations that catch up

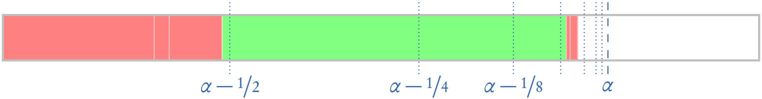
1. While thinking about this, Philip and myself stumbled across another notion of benign approximation.
2. **Definition.** α is *regainingly approximable* if there is
   - a computable left-approximation $(a_n)_n$ of α
   - with $\alpha - a_n < 2^{-n}$ for infinitely many $n \in \mathbb{N}$.



Approximations that catch up

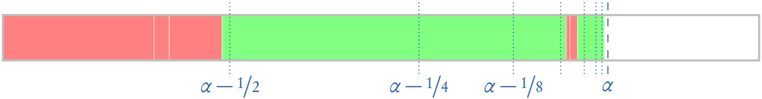
1. While thinking about this, Philip and myself stumbled across another notion of benign approximation.

2. **Definition.** α is *regainingly approximable* if there is
   - a computable left-approximation $(a_n)_n$ of α
   - with $\alpha - a_n < 2^{-n}$ for infinitely many $n \in \mathbb{N}$.



Approximations that catch up

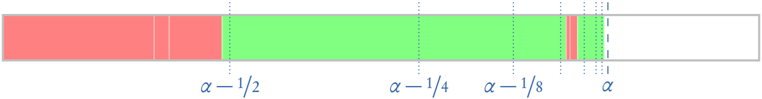
1. While thinking about this, Philip and myself stumbled across another notion of benign approximation.
2. **Definition.** α is *regainingly approximable* if there is
   - a computable left-approximation $(a_n)_n$ of α
   - with $\alpha - a_n < 2^{-n}$ for infinitely many $n \in \mathbb{N}$.



Approximations that catch up

1. While thinking about this, Philip and myself stumbled across another notion of benign approximation.

2. **Definition.** α is *regainingly approximable* if there is
   - a computable left-approximation $(a_n)_n$ of α
   - with $\alpha - a_n < 2^{-n}$ for infinitely many $n \in \mathbb{N}$.

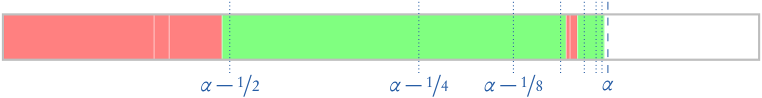


3. **Intuition.**
   - As in general left-computability, the approximation can "dawdle" arbitrarily, *but* infinitely often it must "catch up" to how fast computable numbers can be approximated.

Approximations that catch up

1. While thinking about this, Philip and myself stumbled across another notion of benign approximation.

2. **Definition.** α is *regainingly approximable* if there is
   - a computable left-approximation $(a_n)_n$ of α
   - with $\alpha - a_n < 2^{-n}$ for infinitely many $n \in \mathbb{N}$.



3. **Intuition.**
   - As in general left-computability, the approximation can "dawdle" arbitrarily, *but* infinitely often it must "catch up" to how fast computable numbers can be approximated.
   - Obviously, (in general) we do not know when these good moments occur; in case we do, α is again computable.

Approximations that catch up

1. Note that regaining approximability seems like a really natural notion. We expected to find previous work on this, but it seems no one looked at it before.
2. Thus, with Peter Hertling, we studied many of their properties. Let's mention only the ones most relevant for this talk.

Some selected properties

1. **Theorem.** The regainingly approximable numbers lie properly between the computable and the left-computable numbers.

Some selected properties

1. **Theorem.** The regainingly approximable numbers lie properly between the computable and the left-computable numbers.

2. **Fact.** All K-trivials are regainingly approximable.
3. **Fact.** All regainingly approximable α's are i.o. K-trivial.
   - **Idea.** For every n such that approximation step a_n "catches up", we just need to encode n to know α up to precision 2^{-n}, and thus to roughly know its first n bits.

Some selected properties

1. **Theorem.** The regainingly approximable numbers lie properly between the computable and the left-computable numbers.
2. **Fact.** All K-trivials are regainingly approximable.
3. **Fact.** All regainingly approximable α's are i.o. K-trivial.
   - **Idea.** For every n such that approximation step a_n "catches up", we just need to encode n to know α up to precision 2^{-n}, and thus to roughly know its first n bits.
4. **Question.** Does it coincide with K-triviality?

Regaining approximability is not K-triviality

1. **Theorem.** There is a regainingly approximable α such that $K(\alpha \upharpoonright n) > n$ for infinitely many n.

Regaining approximability is not K-triviality

1. **Theorem.** There is a regainingly approximable α such that $K(\alpha \upharpoonright n) > n$ for infinitely many n.
2. **Proof idea.**
   - Imitate a left-approximation of Ω by copying all its jumps.

Regaining approximability is not K-triviality

1. **Theorem.** There is a regainingly approximable α such that $K(\alpha \upharpoonright n) > n$ for infinitely many n.
2. **Proof idea.**
   - Imitate a left-approximation of Ω by copying all its jumps.
   - Once the resulting initial segment looks to be of high complexity, continue imitating Ω, but scale the jumps down by some factor.

Regaining approximability is not K-triviality

1. **Theorem.** There is a regainingly approximable α such that $K(\alpha \upharpoonright n) > n$ for infinitely many n.
2. **Proof idea.**
   - Imitate a left-approximation of Ω by copying all its jumps.
   - Once the resulting initial segment looks to be of high complexity, continue imitating Ω, but scale the jumps down by some factor.
   - That factor is chosen such that the sum of all future jumps is small enough not to violate regaining approximability.

Regaining approximability is not K-triviality

1. **Theorem.** There is a regainingly approximable α such that $K(\alpha \upharpoonright n) > n$ for infinitely many n.
2. **Proof idea.**
   - Imitate a left-approximation of Ω by copying all its jumps.
   - Once the resulting initial segment looks to be of high complexity, continue imitating Ω, but scale the jumps down by some factor.
   - That factor is chosen such that the sum of all future jumps is small enough not to violate regaining approximability.
   - In case we were deceived and the initial segment turns out to have low complexity after all, retroactively undo the scaling.

Regaining approximability is not K-triviality

1. **Theorem.** There is a regainingly approximable α such that $K(\alpha \upharpoonright n) > n$ for infinitely many n.
2. **Proof idea.**
   - Imitate a left-approximation of Ω by copying all its jumps.
   - Once the resulting initial segment looks to be of high complexity, continue imitating Ω, but scale the jumps down by some factor.
   - That factor is chosen such that the sum of all future jumps is small enough not to violate regaining approximability.
   - In case we were deceived and the initial segment turns out to have low complexity after all, retroactively undo the scaling.
   - This is allowed, as all we want is left-computability.

Regaining approximability is not K-triviality

1. **Theorem.** There is a regainingly approximable α such that $K(\alpha \restriction n) > n$ for infinitely many n.
2. **Proof idea.**
   - Imitate a left-approximation of Ω by copying all its jumps.
   - Once the resulting initial segment looks to be of high complexity, continue imitating Ω, but scale the jumps down by some factor.
   - That factor is chosen such that the sum of all future jumps is small enough not to violate regaining approximability.
   - In case we were deceived and the initial segment turns out to have low complexity after all, retroactively undo the scaling.
   - This is allowed, as all we want is left-computability.
   - Iterate. □

Regaining approximability implies speedability

1. Why might regaining approximability be relevant for us?

Regaining approximability implies speedability

1. Why might regaining approximability be relevant for us?
2. Because in order to catch up, a regaining approximation needs to make big jumps. **Question.** Are those moments of speedability?

Regaining approximability implies speedability

1. Why might regaining approximability be relevant for us?
2. Because in order to catch up, a regaining approximation needs to make big jumps. **Question.** Are those moments of speedability?
3. **Answer.** Almost, but not quite. Even a small jump could be the one finally catching up, if a large jump was made previously.

Regaining approximability implies speedability

1. Why might regaining approximability be relevant for us?
2. Because in order to catch up, a regaining approximation needs to make big jumps. **Question.** Are those moments of speedability?
3. **Answer.** Almost, but not quite. Even a small jump could be the one finally catching up, if a large jump was made previously.
4. But still, catching up requires making some big jump *somewhere*, and we can prove the following statement as a consequence.
5. **Proposition.** Every regainingly approximable α is speedable.

The converse is not true

1. **Proposition (Merkle & Titov).** Every left-computable α that is the binary expansion of a c.e. set is speedable.
2. **Theorem.** Not all such α are regainingly approximable.

So what?

1. The notion of regaining approximability requires something to have happened at some specific time.
2. Thus, making a number non-regainingly approximable looks easier than making it non-speedable.

So what?

1. The notion of regaining approximability requires something to have happened at some specific time.
2. Thus, making a number non-regainingly approximable looks easier than making it non-speedable.
3. But of course, due to the last slide, this is not good enough to negatively answer the open question yet.

So what?

1. The notion of regaining approximability requires something to have happened at some specific time.
2. Thus, making a number non-regainingly approximable looks easier than making it non-speedable.
3. But of course, due to the last slide, this is not good enough to negatively answer the open question yet.
4. Something is still missing, and this brings us to our third notion of benign approximability.

Nearly computable, left-computable numbers

1 **Definition (Hertling & Janicki).**

- $f : \mathbb{N} \to \mathbb{N}$ is a *modulus of convergence* of $(a_n)_n$ if for all $n \in \mathbb{N}$, and all $m \geq f(n)$, we have $|\alpha - a_m| < 2^{-n}$, where $\alpha = \lim a_n$.

Nearly computable, left-computable numbers

1. **Definition (Hertling & Janicki).**
   - $f : \mathbb{N} \to \mathbb{N}$ is a *modulus of convergence* of $(a_n)_n$ if for all $n \in \mathbb{N}$, and all $m \geq f(n)$, we have $|\alpha - a_m| < 2^{-n}$, where $\alpha = \lim a_n$.
   - $(a_n)_n$ *converges computably* if it has a computable such f.

Nearly computable, left-computable numbers

1 Definition (Hertling & Janicki).

- $f:\mathbb{N}\to\mathbb{N}$ is a *modulus of convergence* of $(a_n)_n$ if for all $n\in\mathbb{N}$, and all $m\geq f(n)$, we have $|\alpha - a_m| < 2^{-n}$, where $\alpha = \lim a_n$.
- $(a_n)_n$ *converges computably* if it has a computable such f.
- β is *nearly computable* if, for every computable increasing $(b_n)_n$ converging to it, $(b_{n+1}-b_n)_n$ converges computably to 0.

(This is a special case for left-computables; good enough for us.)

Nearly computable, left-computable numbers

1 Definition (Hertling & Janicki).

- $f : \mathbb{N} \to \mathbb{N}$ is a *modulus of convergence* of $(a_n)_n$ if for all $n \in \mathbb{N}$, and all $m \geq f(n)$, we have $|\alpha - a_m| < 2^{-n}$, where $\alpha = \lim a_n$.
- $(a_n)_n$ *converges computably* if it has a computable such f.
- β is *nearly computable* if, for every computable increasing $(b_n)_n$ converging to it, $(b_{n+1} - b_n)_n$ converges computably to 0.

(This is a special case for left-computables; good enough for us.)

2 Theorem (Downey & LaForte; reformulated). There are non-computable, left-computable, nearly computable numbers.

(In their original formulation, they showed the existence of a non-computable, left-computable number all of whose *presentations* via prefix-free c.e. sets are computable.)

3 Intuition. Knowing computable upper bounds on the *size* of individual jumps that may still be made doesn't "computably determine" their total *sum*.

How does this help answer the open question?

How does this help answer the open question?

1. **Recall:** We have that
   speedability implies regaining approximability.

How does this help answer the open question?

1. **Theorem.** Within the nearly computables,
   speedability coincides with regaining approximability.

How does this help answer the open question?

1. **Theorem.** Within the nearly computables,

   speedability coincides with regaining approximability.

2. So if we build something that is nearly computable and not regainingly approximable, it will not be speedable either.

How does this help answer the open question?

1. **Theorem.** Within the nearly computables,
   speedability coincides with regaining approximability.
2. So if we build something that is nearly computable and not regainingly approximable, it will not be speedable either.
3. **Question.** Can we do so while avoiding randomness, to answer the open question of Merkle & Titov?

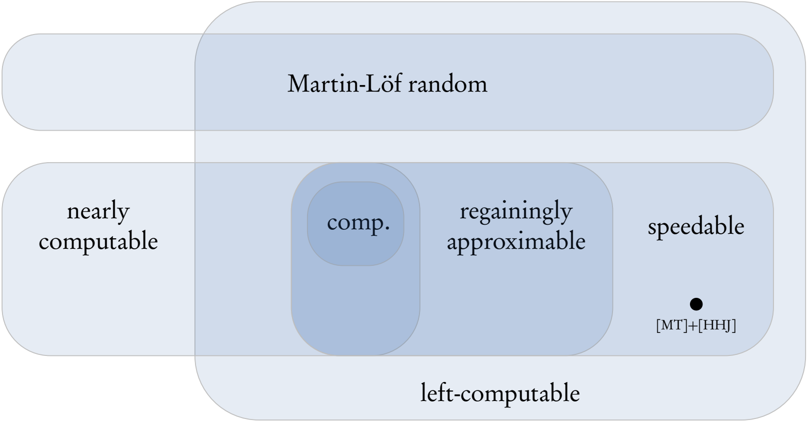
How does this help answer the open question?

1. **Theorem.** Within the nearly computables,
   speedability coincides with regaining approximability.
2. So if we build something that is nearly computable and not regainingly approximable, it will not be speedable either.
3. **Question.** Can we do so while avoiding randomness, to answer the open question of Merkle & Titov? **Yes!**

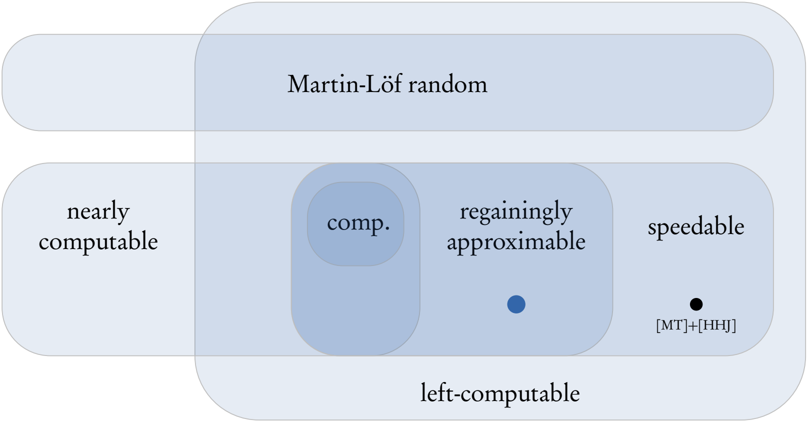
How does this help answer the open question?

1. **Theorem.** Within the nearly computables,
   speedability coincides with regaining approximability.
2. So if we build something that is nearly computable and not regainingly approximable, it will not be speedable either.
3. **Question.** Can we do so while avoiding randomness, to answer the open question of Merkle & Titov? **Yes!**
4. **Theorem (Stephan & Wu; reformulated).** Left-computable nearly computable numbers cannot be Martin-Löf random.

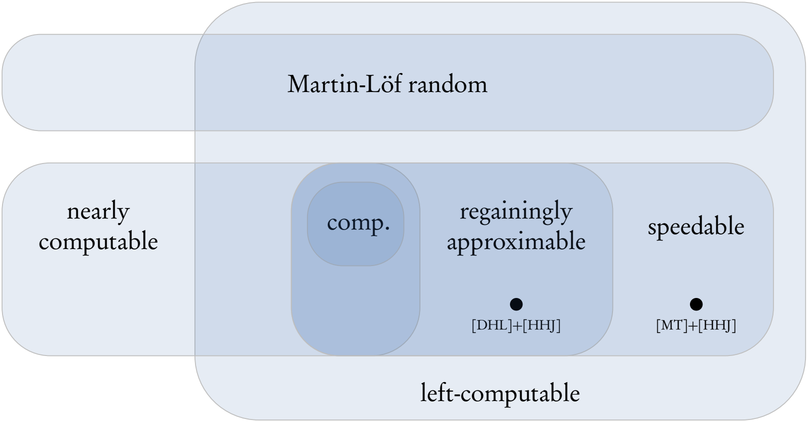
What we know so far & what remains to do



What we know so far & what remains to do

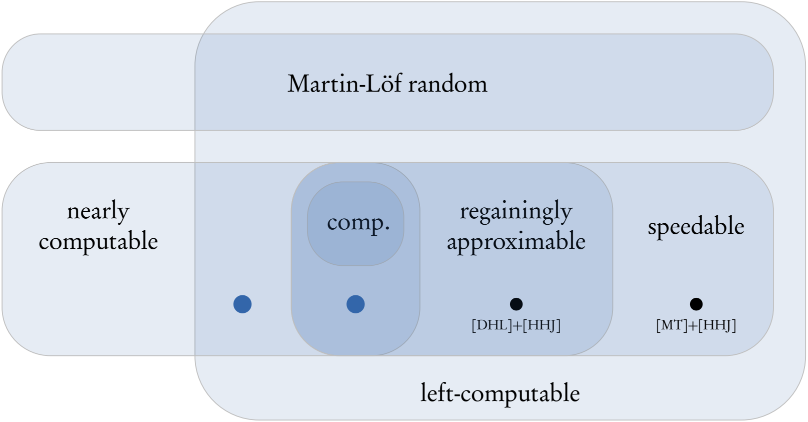


What we know so far & what remains to do



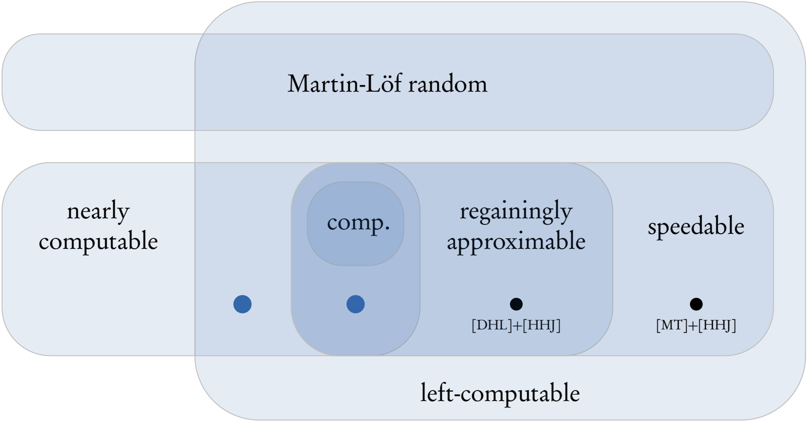
1 **Corollary.** Existence follows from a result of Downey, Hirschfelt and LaForte, combined with ours. (Details omitted.)

What we know so far & what remains to do



2 It remains to show that these two elements exist.

What we know so far & what remains to do



2 It remains to show that these two elements exist.

3 The left one then answers the open question negatively.

3 Constructing the missing points

Constructing the missing points

1. The proofs are inspired by Downey & LaForte's proof that non-trivial nearly computable numbers do exist.
2. But they are more complex because we need to satisfy more and more complex requirements.
3. We can only hint at some of the main ideas here.

Compatibility of the notions

1 **Theorem.** There is a non-computable α that is
regainingly approximable and nearly computable.

Compatibility of the notions

1. **Theorem.** There is a non-computable α that is
   regainingly approximable   and   nearly computable.

2. We construct α as the limit of some $(a_n)_n$ from the left.

Compatibility of the notions

1. **Theorem.** There is a non-computable α that is
   regainingly approximable and nearly computable.
2. We construct α as the limit of some $(a_n)_n$ from the left.
3. **To ensure regaining approximability:** see below.

Compatibility of the notions

1. **Theorem.** There is a non-computable α that is
   regainingly approximable and nearly computable.
2. We construct α as the limit of some $(a_n)_n$ from the left.
3. **To ensure regaining approximability:** see below.
4. **For non-computability:**

   $$\mathcal{N}_e: \ \varphi_e \text{ total and increasing} \Rightarrow (\exists m \in \mathbb{N})\ \alpha - a_{\varphi_e(m)} \geq 2^{-m}.$$

Compatibility of the notions

1. **Theorem.** There is a non-computable α that is

   regainingly approximable   and   nearly computable.

2. We construct α as the limit of some $(a_n)_n$ from the left.
3. **To ensure regaining approximability:** see below.
4. **For non-computability:**

   $$\mathcal{N}_e\colon\ \ \varphi_e \text{ total and increasing} \Rightarrow (\exists m \in \mathbb{N})\ \alpha - a_{\varphi_e(m)} \geq 2^{-m}.$$

5. **For near computability:**

   $$\mathcal{P}_e\colon\ \ \varphi_e \text{ total and increasing} \Rightarrow$$
   $$(a_{\varphi_e(t+1)} - a_{\varphi_e(t)})_t \text{ converges computably to } 0.$$

Compatibility of the notions

1. These two types of requirements seem to be in conflict:
   - The left-approximation of α we construct may need to satisfy negative requirements by performing large jumps rather late.
   - But for positive requirements, we need to commit at certain points to never again making jumps larger than some bound.

Compatibility of the notions

1. These two types of requirements seem to be in conflict:
   - The left-approximation of α we construct may need to satisfy negative requirements by performing large jumps rather late.
   - But for positive requirements, we need to commit at certain points to never again making jumps larger than some bound.
2. We need to carefully balance out these necessities:
   - If a low priority strategy wants to make a large jump, but can't due to a higher priority commitment, then that jump is divided into smaller jumps that are then scheduled for later execution.
   - Even stricter commitments by even higher priority strategies might lead to even further splitting.
   - A negative requirement is satisfied once all corresponding small jumps have been executed.

Compatibility of the notions

1. These two types of requirements seem to be in conflict:
   - The left-approximation of α we construct may need to satisfy negative requirements by performing large jumps rather late.
   - But for positive requirements, we need to commit at certain points to never again making jumps larger than some bound.
2. We need to carefully balance out these necessities:
   - If a low priority strategy wants to make a large jump, but can't due to a higher priority commitment, then that jump is divided into smaller jumps that are then scheduled for later execution.
   - Even stricter commitments by even higher priority strategies might lead to even further splitting.
   - A negative requirement is satisfied once all corresponding small jumps have been executed.
3. Our task is to ensure that all required jumps are executed eventually. This is hard because "φ_e is total and increasing" is a non-computable property, necessitating the use of infinite injury.

Compatibility of the notions

1. To also achieve regaining approximability, we want to use a similar idea as above when we were copying Ω:
   - At certain times, we want to scale down the entire game, so that the sum of all future jumps is sufficiently upper bounded.
2. This needs to be done very carefully, because the mechanism discussed on the last slide obliges us to make jumps of defined sizes waiting in queues for their turn. No scaling allowed!
3. Our way out is to make sure that there exist infinitely many *cut-off stages* when the queues are (in some sense) empty enough. □

Separating the notions

1 **Theorem.** There exists a left-computable α which is nearly computable and not regainingly approximable.

Separating the notions

1. **Theorem.** There exists a left-computable α which is
   nearly computable and not regainingly approximable.

2. We use the same positive requirements for near computability.

Separating the notions

1. **Theorem.** There exists a left-computable α which is
   nearly computable and not regainingly approximable.

2. We use the same positive requirements for near computability.
3. **To prevent regaining approximability:**

$$\mathcal{N}_e\colon\ \varphi_e \text{ total and increasing } \Rightarrow$$
$$(\exists m \in \mathbb{N})(\forall n \geq m)\ \alpha - a_{\varphi_e(n)} \geq 2^{-n}.$$

Separating the notions

1. **Theorem.** There exists a left-computable α which is

   nearly computable and not regainingly approximable.

2. We use the same positive requirements for near computability.
3. **To prevent regaining approximability:**

$$\mathcal{N}_e:\ \varphi_e \text{ total and increasing } \Rightarrow$$
$$(\exists m \in \mathbb{N})(\forall n \geq m)\ \alpha - a_{\varphi_e(n)} \geq 2^{-n}.$$

4. The construction is similar to the previous one, except
   - the infinitary negative requirements need different timing and a different initialisation strategy, and
   - this time we need not ensure the existence of cut-off stages, making the verification significantly easier. □

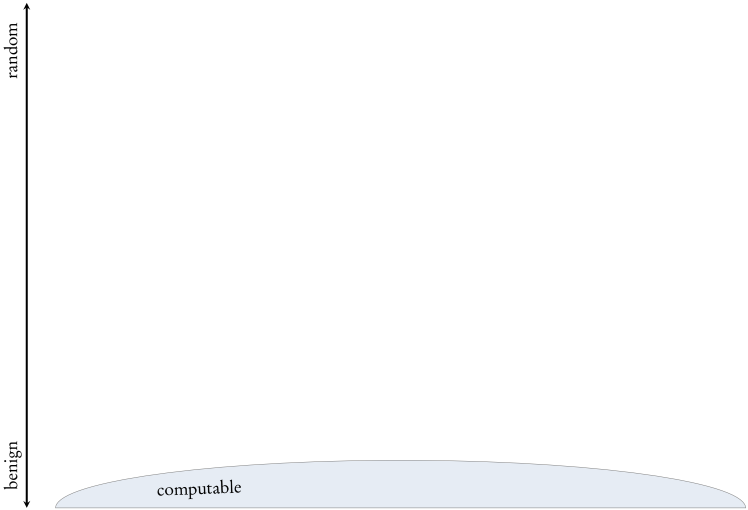
4

A different perspective

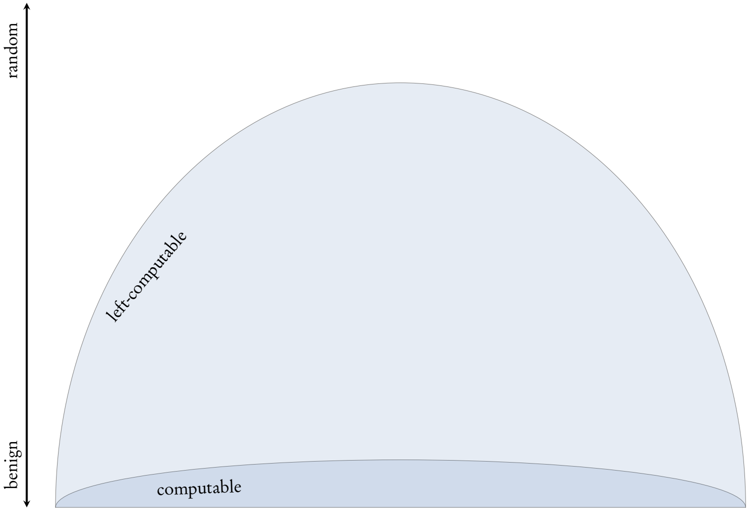
Benignness versus randomness

Benignness versus randomness

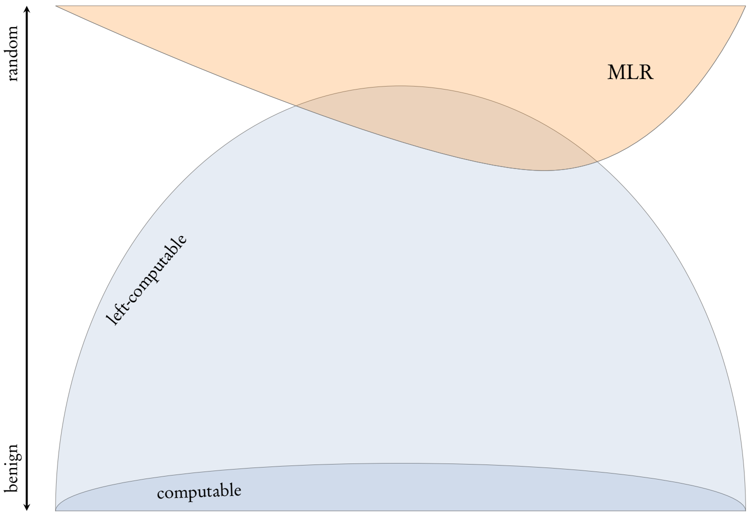
Benignness versus randomness



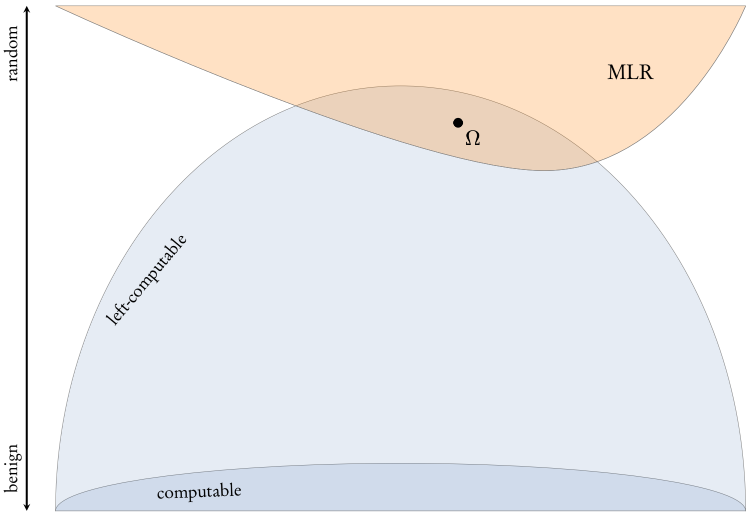
Benignness versus randomness



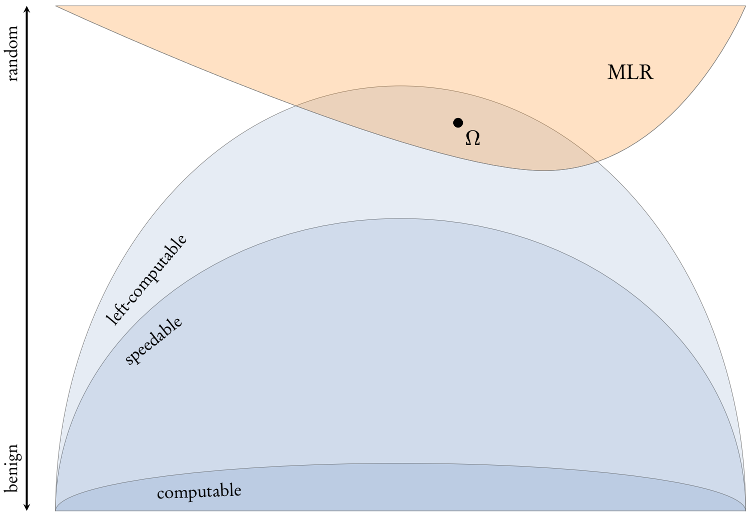
Benignness versus randomness



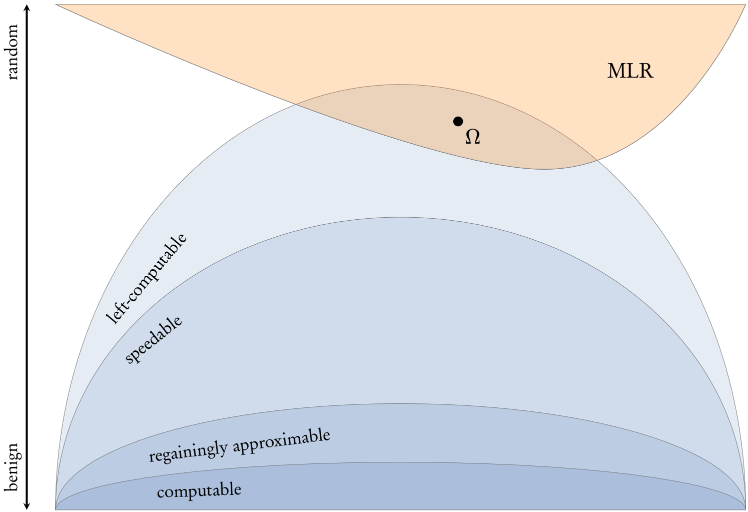
Benignness versus randomness



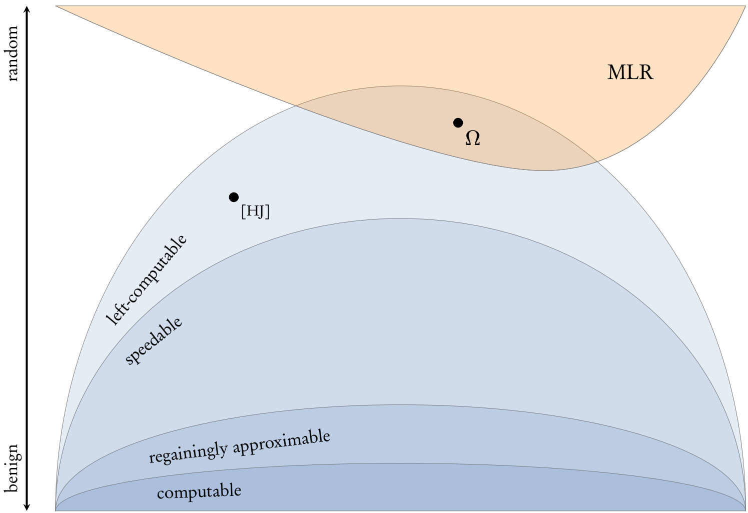
Benignness versus randomness



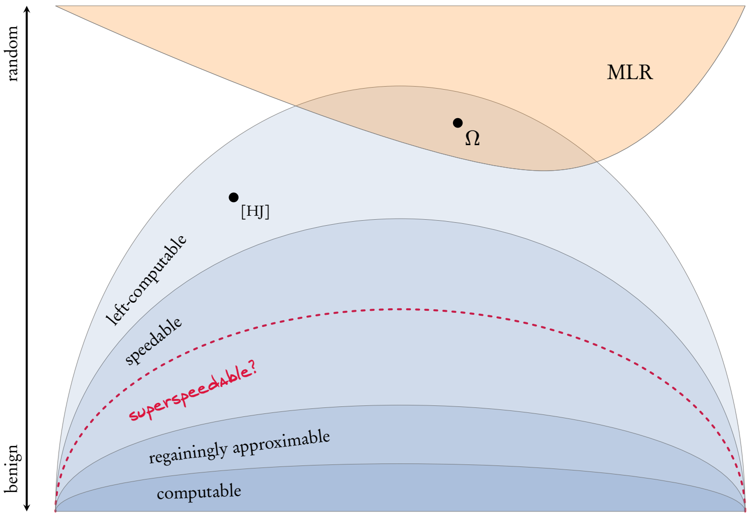
Benignness versus randomness



Benignness versus randomness



Benignness versus randomness



1 **Question (Barmpalias).** Do all speedables have a *single* approximation whose ρ goes to 1? Or is that a smaller set?

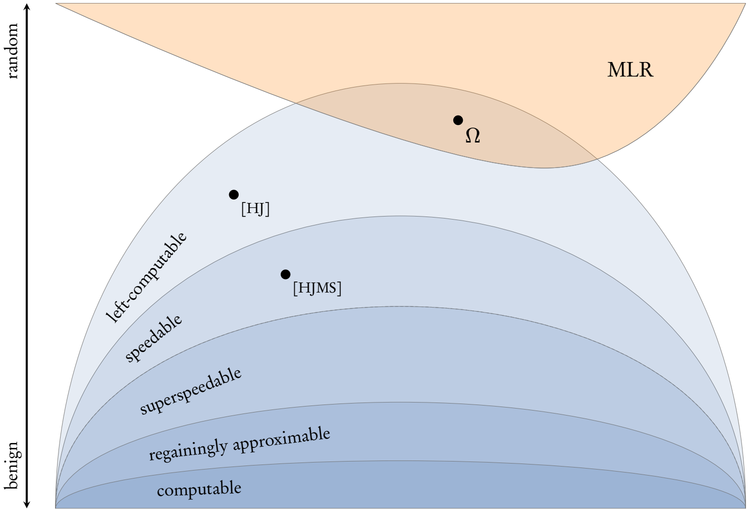
Superspeedability?

1. **Definition.** We call α *superspeedable* if there is a computable left-approximation $(a_n)_n$ of α such that
$$\limsup_{n\to\infty} \frac{a_{n+1}-a_n}{\alpha - a_n} = 1.$$
2. **Question (Barmpalias).** Is speedable $=$ superspeedable?

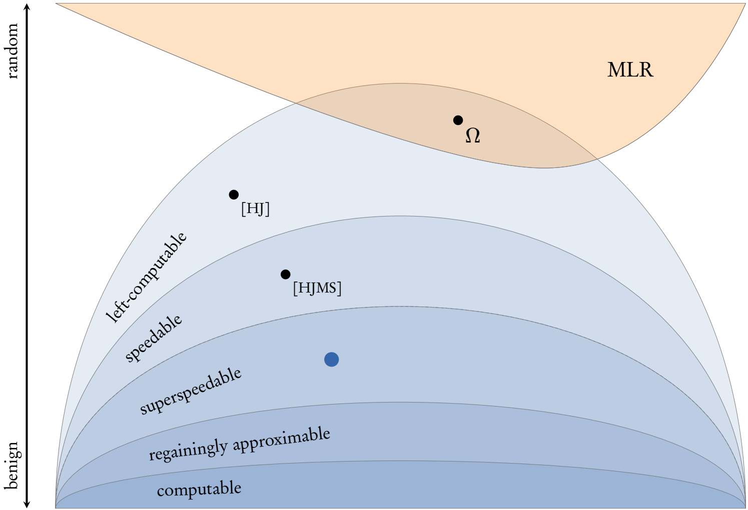
Superspeedability?

1. **Definition.** We call α *superspeedable* if there is a computable left-approximation $(a_n)_n$ of α such that
$$\limsup_{n\to\infty} \frac{a_{n+1} - a_n}{\alpha - a_n} = 1.$$
2. **Question (Barmpalias).** Is speedable = superspeedable?
3. **Theorem (Titov?).** For left-computable numbers,
   not immune implies speedable.
4. **Theorem.** There exists a left-computable number that is
   not immune and not superspeedable.

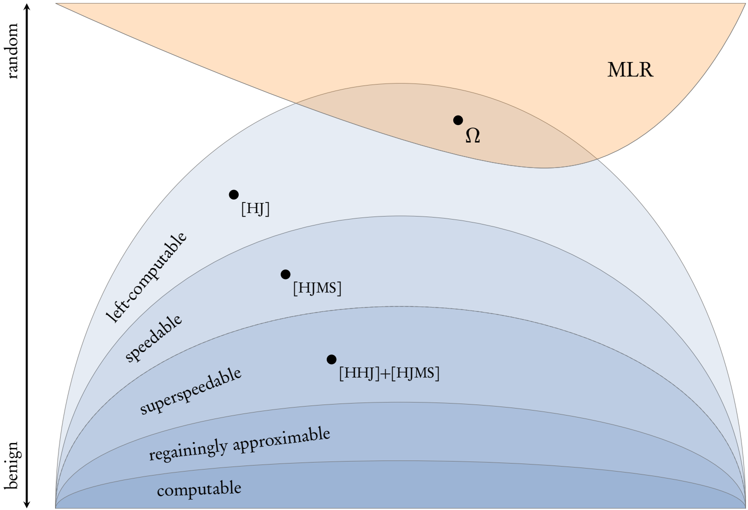
Benignness versus randomness



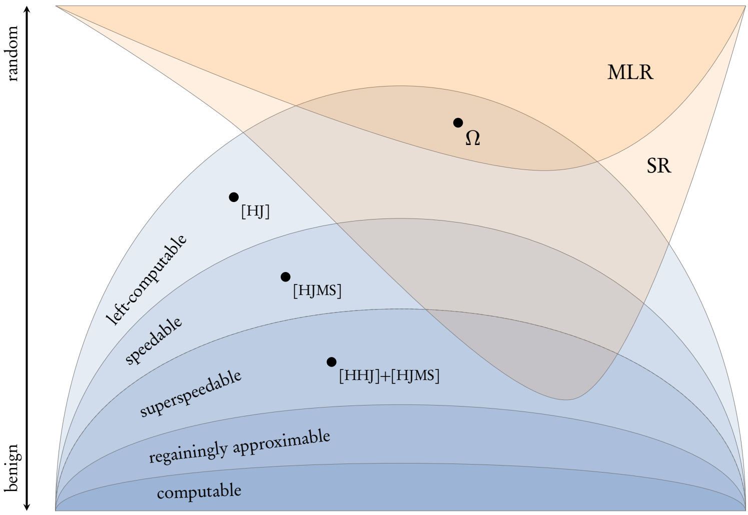
Benignness versus randomness



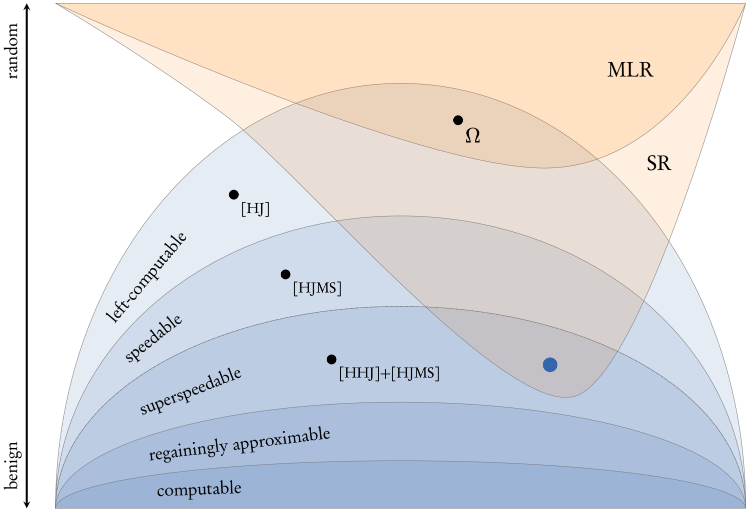
Benignness versus randomness



Benignness versus randomness



Benignness versus randomness



A superspeedable Schnorr random

1 **Theorem.** There is a superspeedable Schnorr random.

A superspeedable Schnorr random

1. **Theorem.** There is a superspeedable Schnorr random.

2. **Theorem (Franklin, Stephan).** The following are equivalent.
   - A is not Schnorr random;
   - there is a computable martingale d with the savings property and a computable function f such that $\exists^\infty n\ (d(A \restriction f(n)) \geqslant n)$.

   (That is, there is a computable lower bound on the winning speed of d.)

A superspeedable Schnorr random

1. **Theorem.** There is a superspeedable Schnorr random.

2. **Theorem (Franklin, Stephan).** The following are equivalent.
   - A is not Schnorr random;
   - there is a computable martingale d with the savings property and a computable function f such that $\exists^\infty n\ (d(A \upharpoonright f(n)) \geqslant n)$.

   (That is, there is a computable lower bound on the winning speed of d.)

3. For the proof, fix a c.e. A such that $\overline{A}$ is "incomputably thin" (dense simple) but whose binary expansion nonetheless contains arbitrarily long sequences of zeros.

A superspeedable Schnorr random

1. **Theorem.** There is a superspeedable Schnorr random.

2. **Theorem (Franklin, Stephan).** The following are equivalent.
   - A is not Schnorr random;
   - there is a computable martingale d with the savings property and a computable function f such that $\exists^\infty n \ (d(A \upharpoonright f(n)) \geqslant n)$.

   (That is, there is a computable lower bound on the winning speed of d.)

3. For the proof, fix a c.e. A such that $\overline{A}$ is "incomputably thin" (dense simple) but whose binary expansion nonetheless contains arbitrarily long sequences of zeros.

4. Define Ω_A bitwise via

$$\Omega_A(n) := \begin{cases} \Omega(m) & \text{if } n = p_A(m), \\ 0 & \text{else}; \end{cases}$$

   (That is, all bits of Ω are there, but they are stored at those positions that are in A.)

A superspeedable Schnorr random

Proof sketch.

A superspeedable Schnorr random

Proof sketch.

1. Ω_A is superspeedable:
   - Ω_A contains arbitrarily long blocks of $\mathtt{0}$'s, by choice of A.
   - Then the stages when, for the last time, a bit left of one of these blocks changes, witness superspeedability.

A superspeedable Schnorr random

Proof sketch.

1. Ω_A is superspeedable:
   - Ω_A contains arbitrarily long blocks of 0's, by choice of A.
   - Then the stages when, for the last time, a bit left of one of these blocks changes, witness superspeedability.
2. Ω_A is Schnorr random:

A superspeedable Schnorr random

Proof sketch.

1. Ω_A is superspeedable:
   - Ω_A contains arbitrarily long blocks of $\mathtt{0}$'s, by choice of A.
   - Then the stages when, for the last time, a bit left of one of these blocks changes, witness superspeedability.
2. Ω_A is Schnorr random:
   - If not, there is a computable d winning at computable speed f.
   - We turn this into a partial computable d' winning on Ω. ↯

A superspeedable Schnorr random

Proof sketch.

1. Ω_A is superspeedable:
   - Ω_A contains arbitrarily long blocks of 0's, by choice of A.
   - Then the stages when, for the last time, a bit left of one of these blocks changes, witness superspeedability.
2. Ω_A is Schnorr random:
   - If not, there is a computable d winning at computable speed f.
   - We turn this into a partial computable d' winning on Ω. ↯
   - If a true initial segment of Ω is input, then d' can approximate Ω until it sees a match. (This step causes the partiality of d'.)

A superspeedable Schnorr random

Proof sketch.

1. Ω_A is superspeedable:
   - Ω_A contains arbitrarily long blocks of 0's, by choice of A.
   - Then the stages when, for the last time, a bit left of one of these blocks changes, witness superspeedability.
2. Ω_A is Schnorr random:
   - If not, there is a computable d winning at computable speed f.
   - We turn this into a partial computable d' winning on Ω. ↯
   - If a true initial segment of Ω is input, then d' can approximate Ω until it sees a match. (This step causes the partiality of d'.)
   - By Kolmogorov complexity arguments, c.e. A settles much faster than random Ω. Thus, at the time of the match, d' knows $\overline{A}$.

A superspeedable Schnorr random

Proof sketch.

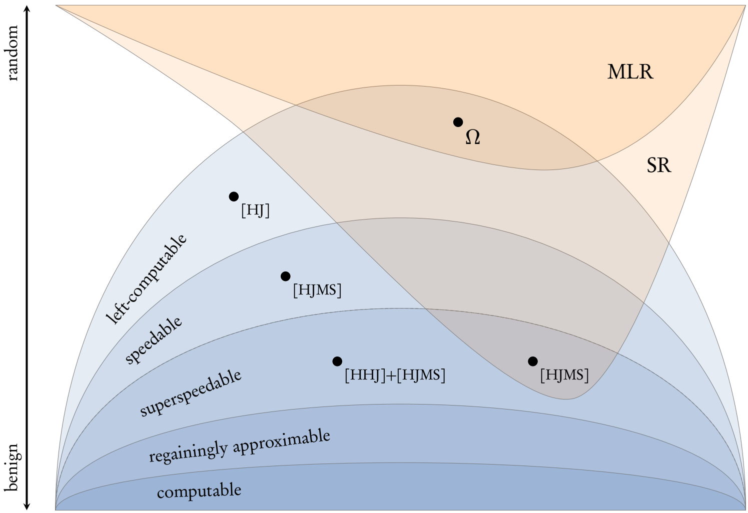
1. Ω_A is superspeedable:
   - Ω_A contains arbitrarily long blocks of 0's, by choice of A.
   - Then the stages when, for the last time, a bit left of one of these blocks changes, witness superspeedability.
2. Ω_A is Schnorr random:
   - If not, there is a computable d winning at computable speed f.
   - We turn this into a partial computable d' winning on Ω. ↯
   - If a true initial segment of Ω is input, then d' can approximate Ω until it sees a match. (This step causes the partiality of d'.)
   - By Kolmogorov complexity arguments, c.e. A settles much faster than random Ω. Thus, at the time of the match, d' knows $\overline{A}$.
   - Then omitting all bets on $\overline{A}$ leads to d' betting on Ω as d would have bet on the corresponding bits encoded in Ω_A.

A superspeedable Schnorr random

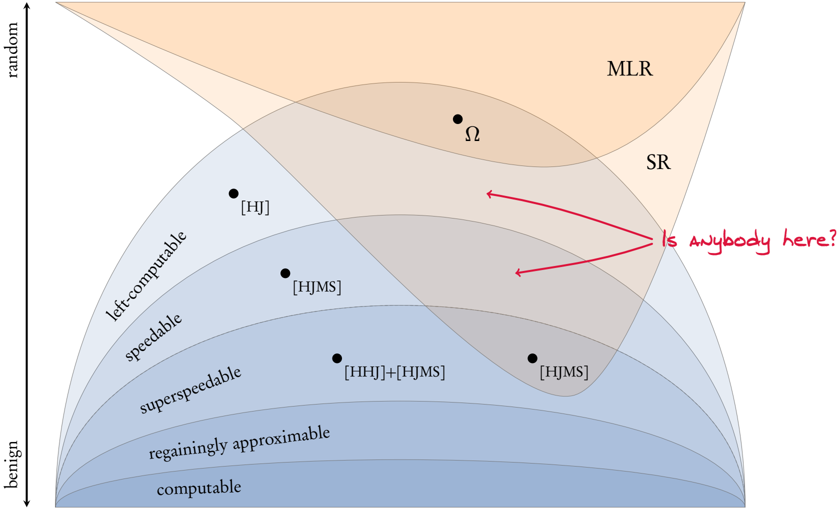
Proof sketch.

1. Ω_A is superspeedable:
   - Ω_A contains arbitrarily long blocks of 0's, by choice of A.
   - Then the stages when, for the last time, a bit left of one of these blocks changes, witness superspeedability.
2. Ω_A is Schnorr random:
   - If not, there is a computable d winning at computable speed f.
   - We turn this into a partial computable d' winning on Ω. ⚡
   - If a true initial segment of Ω is input, then d' can approximate Ω until it sees a match. (This step causes the partiality of d'.)
   - By Kolmogorov complexity arguments, c.e. A settles much faster than random Ω. Thus, at the time of the match, d' knows $\overline{A}$.
   - Then omitting all bets on $\overline{A}$ leads to d' betting on Ω as d would have bet on the corresponding bits encoded in Ω_A.
   - As $\overline{A}$ is incomputably thin, but d had a *computable* winning speed, an infinite portion of its winnings must have originated from the bits of Ω. Thus d' wins infinitely much on Ω. □

Open questions



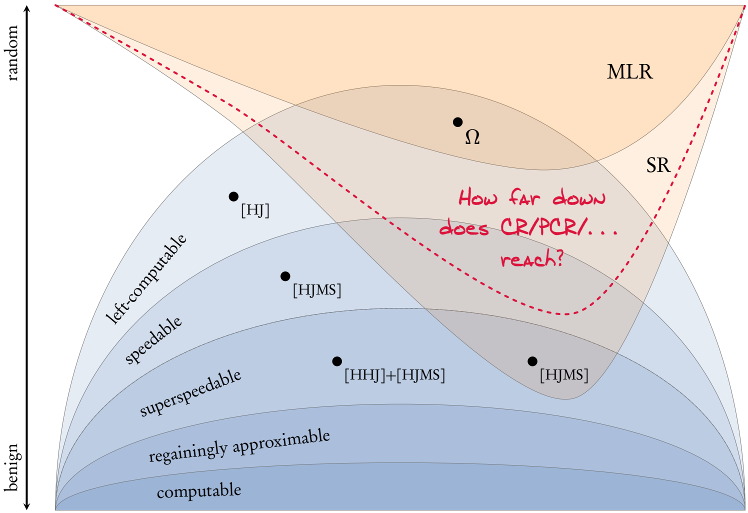
Open questions



1 **Open question.** Do numbers in the marked fields exist?

(It would be strange if not, but we could not construct any.)

Open questions



2 **Open question.** How benignly approximable can computable randoms, partial computable randoms, weak *s*-randoms,… be?

More open questions

1. **Open question.** Recall that we worked inside the nearly computables to obtain a counterexample to the question of Merkle and Titov. Are there counterexamples outside, too?

   Equivalently: Do the Martin-Löf random numbers, the nearly computable numbers, and the speedable numbers together form a covering of all left-computable numbers?

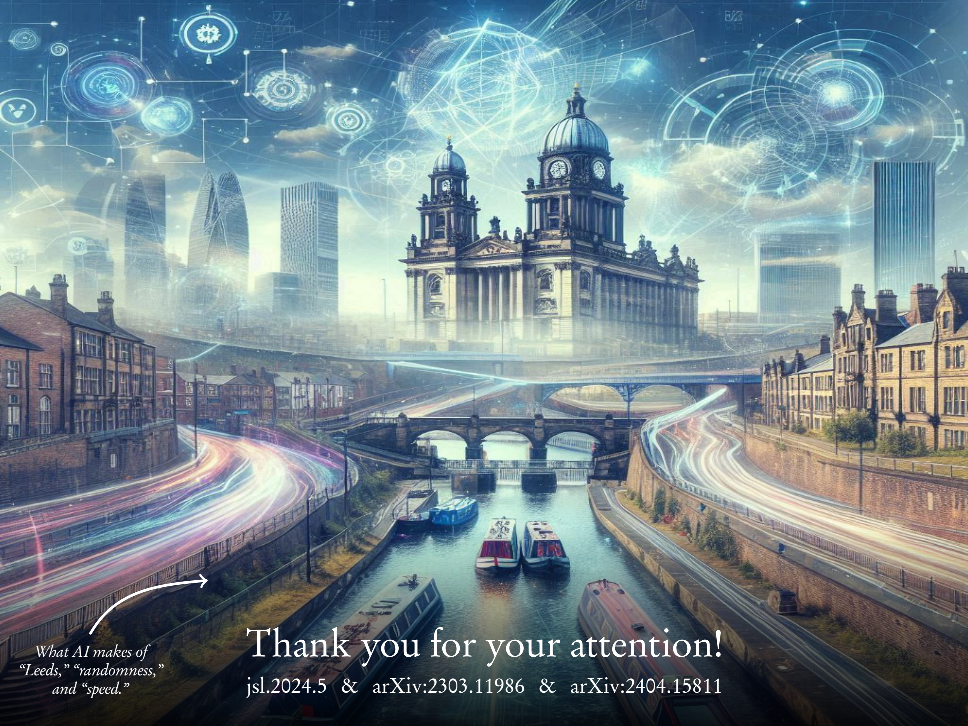
More open questions

1. **Open question.** Recall that we worked inside the nearly computables to obtain a counterexample to the question of Merkle and Titov. Are there counterexamples outside, too?

   Equivalently: Do the Martin-Löf random numbers, the nearly computable numbers, and the speedable numbers together form a covering of all left-computable numbers?

2. **Open question.** What are the Weihrauch degrees of incomputable tasks naturally arising in this area? For example,
   - given an approximation witnessing speedability and a desired ρ, find a sequence of stages where ρ is achieved;
   - for a speedable number and a desired ρ, determine another approximation of that number which achieves ρ;
   - for an approximation witnessing regaining approximability, find the n's at which the approximation "catches up;" etc.

What AI makes of “Leeds,” “randomness,” and “speed.”



What AI makes of "Leeds," "randomness," and "speed."

Thank you for your attention!

jsl.2024.5 & arXiv:2303.11986 & arXiv:2404.15811