The Myth of 'The Myth of Hypercomputation'

Naveen Sundar G.[1] & Selmer Bringsjord[2]

Department of Computer Science[1,2]

Department of Cognitive Science [2]

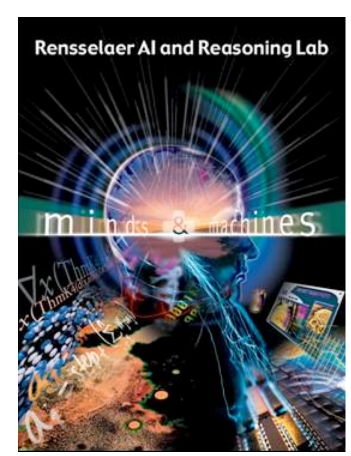
Lally School of Management [2]

Rensselaer Polytechnic Institute (RPI)

Troy NY 12180 USA

selmer@rpi.edu • govinn@rpi.edu

June 9 2011 Turku, Finland



The Myth of Hypercomputation

Martin Davis

Professor Emeritus, Courant Institute, NY University,
Visiting Scholar, Mathematics Department, University of California, Berkeley

Summary. Under the banner of "hypercomputation" various claims are being made for the feasibility of modes of computation that go beyond what is permitted by Turing computability. In this article it will be shown that such claims fly in the face of the inability of all currently accepted physical theories to deal with infinite-precision real numbers. When the claims are viewed critically, it is seen that they amount to little more than the obvious comment that if non-computable inputs are permitted, then non-computable outputs are attainable.

The Myth of Hypercomputation

Martin Davis

Professor Emeritus, Courant Institute, NY University,
Visiting Scholar, Mathematics Department, University of California, Berkeley

Summary. Under the banner of "hypercomputation" various claims are being made for the feasibility of modes of computation that go beyond what is permitted by Turing computability. In this article it will be shown that such claims fly in the face of the inability of all currently accepted physical theories to deal with infinite-precision real numbers. When the claims are viewed critically, it is seen that they amount to little more than the obvious comment that if non-computable inputs are permitted, then non-computable outputs are attainable.

The Myth of Hypercomputation

Martin Davis

Professor Emeritus, Courant Institute, NY University,
Visiting Scholar, Mathematics Department, University of California, Berkeley

MD: Hypercomputation is nothing but a myth.

Summary. Under the banner of "hypercomputation" various claims are being made for the feasibility of modes of computation that go beyond what is permitted by Turing computability. In this article it will be shown that such claims fly in the face of the inability of all currently accepted physical theories to deal with infinite-precision real numbers. When the claims are viewed critically, it is seen that they amount to little more than the obvious comment that if non-computable inputs are permitted, then non-computable outputs are attainable.

The real myth is that Davis' argument(s) is/are sound.

Hypercomputation *qua* field is a success if one or more of the following possibilities hold.

Hypercomputation *qua* field is a success if one or more of the following possibilities hold.

1. POSSIBILITY 1 or η_1: The Church-Turing is false; and it follows that there exist effective computations for functions that aren't Turing-computable.

Hypercomputation *qua* field is a success if one or more of the following possibilities hold.

1. POSSIBILITY 1 or η_1: The Church-Turing is false; and it follows that there exist effective computations for functions that aren't Turing-computable.
2. POSSIBILITY 2 or η_2: There are hypercomputational physical phenomena that may or may not be harnessable. In this case, the functions representing the dynamics of such phenomena are of course Turing-uncomputable.

Hypercomputation *qua* field is a success if one or more of the following possibilities hold.

1. POSSIBILITY 1 or η_1: The Church-Turing is false; and it follows that there exist effective computations for functions that aren't Turing-computable.

2. POSSIBILITY 2 or η_2: There are hypercomputational physical phenomena that may or may not be harnessable. In this case, the functions representing the dynamics of such phenomena are of course Turing-uncomputable.

3. POSSIBILITY 3 or η_3: There are hypercomputational cognitive phenomena that may or may not be harnessable. In this case, the functions representing the dynamics of such phenomena are of course Turing-uncomputable.

$$\phi \equiv \text{ Hypercomputation is true.}$$

$\phi \equiv$ Hypercomputation is true.

$$\phi \equiv \eta \equiv \eta_1 \vee \eta_2 \vee \eta_3$$

$\phi \equiv$ Hypercomputation is true.

$$\phi \equiv \eta \equiv \eta_1 \vee \eta_2 \vee \eta_3$$

Davis tries to establish $\neg\eta$ but fails.

$\phi \equiv$ Hypercomputation is true.

$$\phi \equiv \eta \equiv \eta_1 \vee \eta_2 \vee \eta_3$$

Davis tries to establish $\neg\eta$ but fails.

We don't establish η, but rather defend η against Davis.

One-Slide Encapsulation of the Situation ...

TT, CT, CTT

TT, CT, CTT

TT: A numerical (total) function is effectively computable by some algorithmic routine if and only if (= iff) it is computable by a Turing machine.

TT, CT, CTT

TT: A numerical (total) function is effectively computable by some algorithmic routine if and only if (= iff) it is computable by a Turing machine.

CT: A numerical (total) function is effectively computable by some algorithmic routine if and only if (= iff) it is μ-recursive.

TT, CT, CTT

TT: A numerical (total) function is effectively computable by some algorithmic routine if and only if (= iff) it is computable by a Turing machine.

CT: A numerical (total) function is effectively computable by some algorithmic routine if and only if (= iff) it is μ-recursive.

CTT: The effectively computable total numerical functions are the μ-recursive/Turing computable functions.

TT, CT, CTT

TT: A numerical (total) function is effectively computable by some algorithmic routine if and only if (= iff) it is computable by a Turing machine.

CT: A numerical (total) function is effectively computable by some algorithmic routine if and only if (= iff) it is μ-recursive.

CTT: The effectively computable total numerical functions are the μ-recursive/Turing computable functions.

And, for a function *f* to be effectively computable, is for a human agent/computor/calculator/... to follow an algorithm in order to compute ... *f*.

$\vdash$ CTT

$\vdash \mathrm{CTT}$ Yes?

$\vdash \mathrm{CTT}$ Yes?

No.

$\vdash \mathrm{CTT}$ Yes?

No. $\nvdash \mathrm{CTT}$

$\vdash \mathrm{CTT}$ Yes?

No. $\nvdash \mathrm{CTT}$ But in the stipulative manner it *can* be done:

$\vdash \mathrm{CTT}$ **Yes?**

No. $\nvdash \mathrm{CTT}$ But in the stipulative manner it *can* be done:

$$\forall a \forall \mathcal{A}(C(a, \mathcal{A}) \leftrightarrow \delta)$$

$\vdash \mathrm{CTT}$ **Yes?**

No. $\nvdash \mathrm{CTT}$ But in the stipulative manner it *can* be done:

$$\forall a \forall \mathcal{A}(C(a, \mathcal{A}) \leftrightarrow \delta)$$

$$\forall f \forall a \forall \mathcal{A}(\mathit{Eff}(f) \leftrightarrow C(a, \mathcal{A}))$$

$\vdash \mathrm{CTT}$ **Yes?**

No. $\nvdash \mathrm{CTT}$ But in the stipulative manner it *can* be done:

$$\forall a \forall \mathcal{A}(C(a, \mathcal{A}) \leftrightarrow \delta)$$

$$\forall f \forall a \forall \mathcal{A}(\mathit{Eff}(f) \leftrightarrow C(a, \mathcal{A}))$$ ...

$\vdash \mathrm{CTT}$ Yes?

No. $\nvdash \mathrm{CTT}$ But in the stipulative manner it *can* be done:

$$\forall a \forall \mathcal{A}(C(a, \mathcal{A}) \leftrightarrow \delta)$$

$$\forall f \forall a \forall \mathcal{A}(\mathit{Eff}(f) \leftrightarrow C(a, \mathcal{A}))$$ ...

And hypercomputation?

$\vdash \mathrm{CTT}$ **Yes?**

No. $\nvdash \mathrm{CTT}$ But in the stipulative manner it *can* be done:

$$\forall a \forall \mathcal{A}(C(a, \mathcal{A}) \leftrightarrow \delta)$$

$$\forall f \forall a \forall \mathcal{A}(\mathit{Eff}(f) \leftrightarrow C(a, \mathcal{A}))$$ ...

And hypercomputation? $\vdash \Diamond \psi_{hyper}$

$\vdash \mathrm{CTT}$ Yes?

No. $\nvdash \mathrm{CTT}$ But in the stipulative manner it *can* be done:

$$\forall a \forall \mathcal{A}(C(a, \mathcal{A}) \leftrightarrow \delta)$$

$$\forall f \forall a \forall \mathcal{A}(\mathit{Eff}(f) \leftrightarrow C(a, \mathcal{A}))$$ ...

And hypercomputation? $\vdash \Diamond \psi_{hyper}$ Done!

$\vdash \mathrm{CTT}$ Yes?

No. $\nvdash \mathrm{CTT}$ But in the stipulative manner it *can* be done:

$$\forall a \forall \mathcal{A}(C(a,\mathcal{A}) \leftrightarrow \delta)$$

$$\forall f \forall a \forall \mathcal{A}(\mathit{Eff}(f) \leftrightarrow C(a,\mathcal{A}))$$ ...

And hypercomputation? $\vdash \Diamond \psi_{hyper}$ Done!

Too easy?

$\vdash \mathrm{CTT}$ Yes?

No. $\nvdash \mathrm{CTT}$ But in the stipulative manner it *can* be done:

$$\forall a \forall \mathcal{A}(C(a, \mathcal{A}) \leftrightarrow \delta)$$

$$\forall f \forall a \forall \mathcal{A}(\mathit{Eff}(f) \leftrightarrow C(a, \mathcal{A}))$$ ...

And hypercomputation? $\vdash \Diamond \psi_{hyper}$ Done!

Too easy? $\vdash \Diamond \Diamond_p \psi_{hyper}$

$\vdash \mathrm{CTT}$ **Yes?**

No. $\nvdash \mathrm{CTT}$ But in the stipulative manner it *can* be done:

$$\forall a \forall \mathcal{A}(C(a, \mathcal{A}) \leftrightarrow \delta)$$

$$\forall f \forall a \forall \mathcal{A}(\mathit{Eff}(f) \leftrightarrow C(a, \mathcal{A}))$$ ...

And hypercomputation? $\vdash \Diamond \psi_{hyper}$ **Done!**

Too easy? $\vdash \Diamond \Diamond_p \psi_{hyper}$

What is needed:

$\vdash \mathrm{CTT}$ Yes?

No. $\nvdash \mathrm{CTT}$ But in the stipulative manner it *can* be done:

$$\forall a \forall \mathcal{A}(C(a, \mathcal{A}) \leftrightarrow \delta)$$

$$\forall f \forall a \forall \mathcal{A}(\mathit{Eff}(f) \leftrightarrow C(a, \mathcal{A}))$$ ...

And hypercomputation? $\vdash \Diamond \psi_{hyper}$ Done!

Too easy? $\vdash \Diamond \Diamond_p \psi_{hyper}$

What is needed: $\vdash \Diamond_p \psi_{hyper}$

$\vdash \mathrm{CTT}$ Yes?

No. $\nvdash \mathrm{CTT}$ But in the stipulative manner it *can* be done:

$$\forall a \forall \mathcal{A}(C(a,\mathcal{A}) \leftrightarrow \delta)$$

$$\forall f \forall a \forall \mathcal{A}(\mathit{Eff}(f) \leftrightarrow C(a,\mathcal{A}))$$ ...

And hypercomputation? $\vdash \Diamond\psi_{hyper}$ Done!

Too easy? $\vdash \Diamond\Diamond_p\psi_{hyper}$

What is needed: $\vdash \Diamond_p\psi_{hyper}$ So ...

$\vdash \mathrm{CTT}$ Yes?

No. $\nvdash \mathrm{CTT}$ But in the stipulative manner it *can* be done:

$$\forall a \forall \mathcal{A}(C(a,\mathcal{A}) \leftrightarrow \delta)$$

$$\forall f \forall a \forall \mathcal{A}(\mathit{Eff}(f) \leftrightarrow C(a,\mathcal{A}))$$ ...

And hypercomputation? $\vdash \Diamond \psi_{hyper}$ Done!

Too easy? $\vdash \Diamond \Diamond_p \psi_{hyper}$

What is needed: $\vdash \Diamond_p \psi_{hyper}$ So ...

$$\mathcal{M} \models [\mathcal{A}] \cup [agents] \cup SpecRel \cup \{\exists f(\neg Tcomputable(f) \wedge \mathit{Eff}computable(f))$$

$\vdash \mathrm{CTT}$ Yes?

No. $\nvdash \mathrm{CTT}$ But in the stipulative manner it *can* be done:

$$\forall a \forall \mathcal{A}(C(a,\mathcal{A}) \leftrightarrow \delta)$$

$$\forall f \forall a \forall \mathcal{A}(\mathit{Eff}(f) \leftrightarrow C(a,\mathcal{A}))$$ ...

And hypercomputation? $\vdash \Diamond \psi_{hyper}$ Done!

Too easy? $\vdash \Diamond \Diamond_p \psi_{hyper}$

What is needed: $\vdash \Diamond_p \psi_{hyper}$ So ...

$\mathcal{M} \models [\mathcal{A}] \cup [agents] \cup SpecRel \cup \{\exists f(\neg Tcomputable(f) \wedge \mathit{Eff}computable(f))$ or:

$\vdash \mathrm{CTT}$ Yes?

No. $\nvdash \mathrm{CTT}$ But in the stipulative manner it *can* be done:

$$\forall a \forall \mathcal{A}(C(a, \mathcal{A}) \leftrightarrow \delta)$$

$$\forall f \forall a \forall \mathcal{A}(\mathit{Eff}(f) \leftrightarrow C(a, \mathcal{A}))$$ ...

And hypercomputation? $\vdash \Diamond \psi_{hyper}$ Done!

Too easy? $\vdash \Diamond \Diamond_p \psi_{hyper}$

What is needed: $\vdash \Diamond_p \psi_{hyper}$ So ...

$$\mathcal{M} \models [\mathcal{A}] \cup [agents] \cup SpecRel \cup \{\exists f(\neg Tcomputable(f) \wedge Effcomputable(f))$$ or:

$$\mathcal{M} \models [H\ machines] \cup SpecRel \cup \{\exists f(\neg Tcomputable(f) \wedge Hcomputable(f))$$

Attack on η_1

Attack on η_1

η_1: The Church-Turing is false; and it follows that there exist effective computations for functions that aren't Turing-computable.

Attack on η_1

Attack on η_1

Davis:

During the 1930s, as a result of the work of a number of logicians, it became possible to explain with full precision what it means to say for some given problem that an algorithm exists providing a solution to that problem. Moreover it then became feasible to prove for certain problems no such algorithm exists, that it is impossible to specify an algorithm that provides a solution to those problems.

Attack on η_1

Davis:

During the 1930s, as a result of the work of a number of logicians, it became possible to explain with full precision what it means to say for some given problem that an algorithm exists providing a solution to that problem. Moreover it then became feasible to prove for certain problems no such algorithm exists, that it is impossible to specify an algorithm that provides a solution to those problems.

Attack on η_1

Davis:

During the 1930s, as a result of the work of a number of logicians, it became possible to explain with full precision what it means to say for some given problem that an algorithm exists providing a solution to that problem. Moreover it then became feasible to prove for certain problems no such algorithm exists, that it is impossible to specify an algorithm that provides a solution to those problems.

Circular reasoning by assuming the CTT.

And:

And:

- A false premise:

And:

- A false premise:
  - To explain something with "full precision" one presumably has a fully formal scheme at one's disposal; but because CTT (and all variants) has at its heart informal notions (e.g., effective computation) that have yet to be suitably formalized, "full precision" has not been obtain.

Attack on η_2

Attack on η_2

η_2: There are hypercomputational physical phenomena that may or may not be harnessable. In this case, the functions representing the dynamics of such phenomena are of course Turing-uncomputable.

Attacking Physical Hypercomputation

Attacking Physical Hypercomputation

- Finiteness Assumptions

Attacking Physical Hypercomputation

- Finiteness Assumptions
- Uncomputable Weights and Weights

Attacking Physical Hypercomputation

- Finiteness Assumptions
- Uncomputable Weights and Weights
- Abstractness and Approximations

Attacking Physical Hypercomputation

- Finiteness Assumptions
- Uncomputable Weights and Weights
- Abstractness and Approximations
- Necessity of Non-computable Real Numbers in Physical Theory

Attacking Physical Hypercomputation

- Finiteness Assumptions
- Uncomputable Weights and Weights
- Abstractness and Approximations
- Necessity of Non-computable Real Numbers in Physical Theory
- Science-based Arguments

Finiteness Assumptions

Finiteness Assumptions

- All hypercomputation models exploit infinite resources in some manner.

Finiteness Assumptions

- All hypercomputation models exploit infinite resources in some manner.
- Let η'_2 : There exists (in the mathematical universe) machines that can exploit infinite resources and such machines can be harnessed by people.

Finiteness Assumptions

- All hypercomputation models exploit infinite resources in some manner.
- Let η'_2 : There exists (in the mathematical universe) machines that can exploit infinite resources and such machines can be harnessed by people.
- We can say that $\eta_2 \rightarrow \eta'_2$

Finiteness Assumptions

- All hypercomputation models exploit infinite resources in some manner.
- Let η'_2 : There exists (in the mathematical universe) machines that can exploit infinite resources and such machines can be harnessed by people.
- We can say that $\eta_2 \rightarrow \eta'_2$
- Davis simply asserts $\neg\eta'_2$

Finiteness Assumptions

- All hypercomputation models exploit infinite resources in some manner.
- Let η'_2 : There exists (in the mathematical universe) machines that can exploit infinite resources and such machines can be harnessed by people.
- We can say that $\eta_2 \rightarrow \eta'_2$
- Davis simply asserts $\neg\eta'_2$
  - *"But it is worth noting that unlike the abstract algorithm that countenances no limitation on the size of the numbers being added, a machine implementing this algorithm, being a finite physical object, is constrained to accept only numbers smaller than some definite amount. (Davis 2004, 198)"*

Uncomputable Weights and Weights

Uncomputable Weights and Weights

- Davis claims that Turing-uncomputable inputs are **necessary** to produce Turing uncomputable outputs in **all** hypercomputational models.

Uncomputable Weights and Weights

- Davis claims that Turing-uncomputable inputs are **necessary** to produce Turing uncomputable outputs in **all** hypercomputational models.
  - *"When the claims are viewed critically, it is seen that they amount to little more than the obvious comment that if non-computable inputs are permitted, then non-computable outputs are attainable."*

Uncomputable Weights and Weights

Uncomputable Weights and Weights

- Davis discusses only Siegelmann's *Analog Neural Network Model.*

Uncomputable Weights and Weights

- Davis discusses only Siegelmann's *Analog Neural Network Model.*
- Furthermore, Siegelmann's models provide a model of computation in which one can <u>use and exploit real numbers</u>. This feature is already beyond the capability of Turing machines.

Uncomputable Weights and Weights

Uncomputable Weights and Weights

- There are other models which do not require Turing-uncomputable inputs to produce Turing-uncomputable outputs.

Uncomputable Weights and Weights

- There are other models which do not require Turing-uncomputable inputs to produce Turing-uncomputable outputs.
- Can be categorized as:

Uncomputable Weights and Weights

- There are other models which do not require Turing-uncomputable inputs to produce Turing-uncomputable outputs.
- Can be categorized as:
  - Infinite time Turing machines

Uncomputable Weights and Weights

- There are other models which do not require Turing-uncomputable inputs to produce Turing-uncomputable outputs.
- Can be categorized as:
  - Infinite time Turing machines
  - Super-task machines

Uncomputable Weights and Weights

- There are other models which do not require Turing-uncomputable inputs to produce Turing-uncomputable outputs.
- Can be categorized as:
  - Infinite time Turing machines
  - Super-task machines
  - Physical oracles

Abstractness and Approximations

Abstractness and Approximations

- This rather absurd attack goes as follows

Abstractness and Approximations

- This rather absurd attack goes as follows
  1. Even Turing machines are abstract models that can't be implemented fully.

Abstractness and Approximations

- This rather absurd attack goes as follows

1. Even Turing machines are abstract models that can't be implemented fully.
2. Therefore, no other more powerful model can be implemented fully.

Abstractness and Approximations

- This rather absurd attack goes as follows
    1. Even Turing machines are abstract models that can't be implemented fully.
    2. Therefore, no other more powerful model can be implemented fully.

Abstractness and Approximations

- This rather absurd attack goes as follows

  1. Even Turing machines are abstract models that can't be implemented fully.

  2. Therefore, no other more powerful model can be implemented fully.

Abstractness and Approximations

- Going by the same argument:
  - This rather absurd attack goes as follows

    1. Even Turing machines are abstract models that can't be implemented fully.

    2. Therefore, no other more powerful model can be implemented fully.

Abstractness and Approximations

- Going by the same argument:
    - Since Turing computers can't be realized fully, Turing computation is now another "myth."

Abstractness and Approximations

- Going by the same argument:
  - Since Turing computers can't be realized fully, Turing computation is now another "myth."
- The problem is that Davis fails to recognize that a lot of the hypercomputational models are abstract models that no one hopes to build in the near future.

Necessity of Non-computable Reals

Necessity of Non-computable Reals

- Another point in Davis' argument is that almost all hypercomputation models require Physics to give them a Turing-uncomputable real number.

Necessity of Non-computable Reals

- Another point in Davis' argument is that almost all hypercomputation models require Physics to give them a Turing-uncomputable real number.
- This is false. Quite a large number of hypercomputation models don't require non-computable reals and roughly fall into the following categories

Necessity of Non-computable Reals

- Another point in Davis' argument is that almost all hypercomputation models require Physics to give them a Turing-uncomputable real number.
- This is false. Quite a large number of hypercomputation models don't require non-computable reals and roughly fall into the following categories
    - Infinite time Turing Machines

Necessity of Non-computable Reals

- Another point in Davis' argument is that almost all hypercomputation models require Physics to give them a Turing-uncomputable real number.
- This is false. Quite a large number of hypercomputation models don't require non-computable reals and roughly fall into the following categories
    - Infinite time Turing Machines
    - Zeus Machines

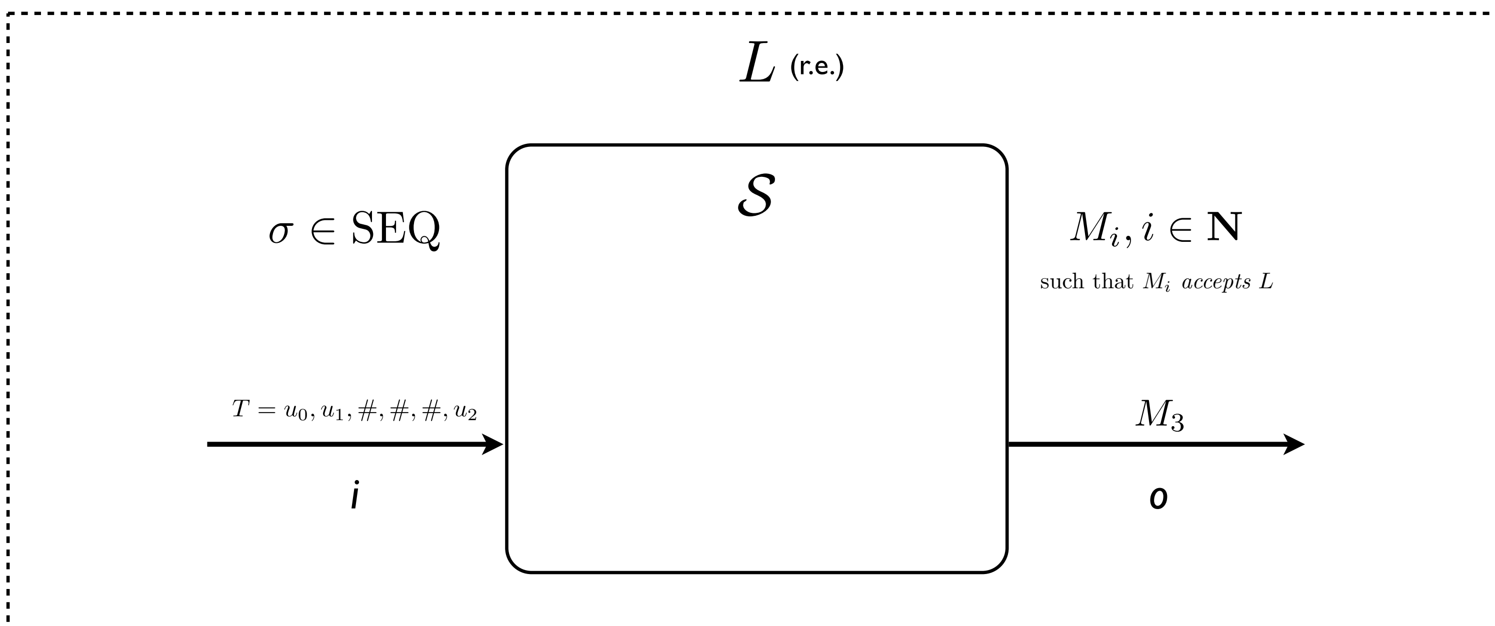
Necessity of Non-computable Reals

- Another point in Davis' argument is that almost all hypercomputation models require Physics to give them a Turing-uncomputable real number.
- This is false. Quite a large number of hypercomputation models don't require non-computable reals and roughly fall into the following categories
  - Infinite time Turing Machines
  - Zeus Machines
  - Kieu-type Quantum Computation

Science-based Arguments: A Meta Analysis of Davis and friends

Science-based Arguments: A Meta Analysis of Davis and friends

CLT-based Model of Science



SoS

SoS

- A language *L* is a set of finite strings from a finite alphabet Σ.

SoS

- A language L is a set of finite strings from a finite alphabet Σ.
- Nature represented by some language L_t.

SoS

- A language L is a set of finite strings from a finite alphabet Σ.
- Nature represented by some language L_t.
- The scientist is presented strings one by one from L_t by Nature.

SoS

- A language *L* is a set of finite strings from a finite alphabet Σ.
- Nature represented by some language L_t.
- The scientist is presented strings one by one from L_t by Nature.
- The scientist has to correctly identify L_t by output i such in some programming system *v*, $W_i^v = L_t$. W_i^v denotes the halting set of the *v*-program *i*.

SoS

SoS

- A particular infinite sequence of strings given by nature is called a *text*.

SoS

- A particular infinite sequence of strings given by nature is called a *text*.
- The scientist is said to identify a language *L* if he/she can produce the correct hypothesis after a finite number of mistakes for all texts of *L*.

SoS

- A particular infinite sequence of strings given by nature is called a *text*.
- The scientist is said to identify a language *L* if he/she can produce the correct hypothesis after a finite number of mistakes for all texts of *L*.
- The scientist is said to *identify* a class of languages $\mathcal{L}$ if he/she can identify all languages in $\mathcal{L}$.

Identifying machines in nature

Identifying machines in nature

- The formalism can be used to identify machines in nature.

Identifying machines in nature

- The formalism can be used to identify machines in nature.
- We are given a black box machine and the set of numbers the machine halts on.

Identifying machines in nature

- The formalism can be used to identify machines in nature.
- We are given a black box machine and the set of numbers the machine halts on.
- Our goal is to identify the language the machine halts on.

Identifying machines in nature

Identifying machines in nature

- Allow the *possibility* of the language K

Identifying machines in nature

- Allow the ***possibility*** of the language K
  - $K=\{i \mid i \notin W_i^v\}$, the non-recursive set of indices of all machines which do not halt on their own index.

Identifying machines in nature

- Allow the ***possibility*** of the language K
    - $K=\{i \mid i \notin W_i^v\}$, the non-recursive set of indices of all machines which do not halt on their own index.
- Assume a black-box machine *H*.

Identifying machines in nature

- Allow the *possibility* of the language K
  - *K*={i | i $\notin W_i^v$}, the non-recursive set of indices of all machines which do not halt on their own index.
- Assume a black-box machine *H*.
- Assume that for all the numbers *n* that the machine *H* has halted on, it has been proved that $n \in K$.

Identifying machines in nature

Identifying machines in nature

- Any rational scientist in this situation will admit the possibility $L_t = K$.

Identifying machines in nature

- Any rational scientist in this situation will admit the possibility $L_t = K$.
- The set of hypotheses for a rational scientist is then $\mathcal{L}_{hyp}^{+K} = \mathcal{L}_{hyp} + K$ where $\mathcal{L}_{hyp} = \{K_{fin} \mid K_{fin} \text{ is finite and } K_{fin} \subset K\}$

Two Lemmata

Two Lemmata

- SoS_1: Let $\mathcal{FIN}$ be the collection of all finite languages; and let $\mathcal{L}$ be an infinite class of languages. Then $\mathcal{L}+\mathcal{FIN}$ is not identifiable by any scientist.

Two Lemmata

- SoS_1: Let $\mathcal{FIN}$ be the collection of all finite languages; and let $\mathcal{L}$ be an infinite class of languages. Then $\mathcal{L}+\mathcal{FIN}$ is not identifiable by any scientist.
- SoS_2: A scientist is said to be self-monitoring if it can signal its own convergence, the point in the text when the scientist produces its final conjecture.

Two Lemmata

- SoS_1: Let $\mathcal{FIN}$ be the collection of all finite languages; and let $\mathcal{L}$ be an infinite class of languages. Then $\mathcal{L}+\mathcal{FIN}$ is not identifiable by any scientist.
- SoS_2: A scientist is said to be self-monitoring if it can signal its own convergence, the point in the text when the scientist produces its final conjecture.
    - No self-monitoring scientist identifies $\mathcal{FIN}$.

Two Lemmata

- SoS_1: Let $\mathcal{FIN}$ be the collection of all finite languages; and let $\mathcal{L}$ be an infinite class of languages. Then $\mathcal{L}+\mathcal{FIN}$ is not identifiable by any scientist.
- SoS_2: A scientist is said to be self-monitoring if it can signal its own convergence, the point in the text when the scientist produces its final conjecture.
  - No self-monitoring scientist identifies $\mathcal{FIN}$.
  - Since the SoS formalism lacks any notion of declarative statements, we take the notion of the self-monitoring signal to be a declaration of the statement that the scientist knows that the final conjecture has been produced.

Therefore,

Therefore,

- By SoS_1, No scientist can identify $\mathcal{L}_{hyp}{}^{+K}$

Therefore,

- By SoS_1, No scientist can identify $\mathcal{L}_{hyp}^{+K}$
- Even if a scientist a priori rejects the possibility of *K* (like Davis), by SoS_2 the scientist cannot be self-monitoring.

Therefore,

- By SoS_1, No scientist can identify $\mathcal{L}_{hyp}^{+K}$
- Even if a scientist a priori rejects the possibility of *K* (like Davis), by SoS_2 the scientist cannot be self-monitoring.

Therefore,

- By SoS_1, No scientist can identify $\mathcal{L}_{hyp}^{+K}$
- Even if a scientist a prior rejects the possibility of K (like Davis), by SoS_2 the scientist cannot be self-monitoring.

Therefore,

Any one, including Davis, who claims that $\mathcal{L}_{hyp}$ is the case, are <u>stating the absurd</u> when they also claim that all the dynamics behind *H* are known and finalized.

Our universe and *H*

Our universe and *H*

- Is represented by some element of $\mathcal{L}_{hyp}{}^{+K}$

Our universe and *H*

- Is represented by some element of $\mathcal{L}_{hyp}{}^{+K}$
- *H* represents <u>harnessable</u> hypercomputation.

Attack on η_3

Attack on η_3

η_3: There are hypercomputational cognitive phenomena that may or may not be harnessable. In this case, the functions representing the dynamics of such phenomena are of course Turing-uncomputable.

Our response:

Our response:

- Parallels that of the SoS argument.

Our response:

- Parallels that of the SoS argument.
- Replace *H* with a cognitive agent *C*

Digressions in the Myth Construction

Digressions in the Myth Construction

- Perpetual Motion Machines

Digressions in the Myth Construction

- Perpetual Motion Machines
- Other theories

Perpetual Machines

Perpetual Machines

- Perpetual machines are outright prohibited by existing laws of physics.

Perpetual Machines

- Perpetual machines are outright prohibited by existing laws of physics.
- Nothing in the existing laws prohibit hypercomputation.

Perpetual Machines

- Perpetual machines are outright prohibited by existing laws of physics.
- Nothing in the existing laws prohibit hypercomputation.
- A bit more precisely:

Perpetual Machines

- Perpetual machines are outright prohibited by existing laws of physics.
- Nothing in the existing laws prohibit hypercomputation.
- A bit more precisely:
  - Ψ_{hyper} = Hypercomputer computers exist

Perpetual Machines

- Perpetual machines are outright prohibited by existing laws of physics.
- Nothing in the existing laws prohibit hypercomputation.
- A bit more precisely:
  - Ψ_{hyper} = Hypercomputer computers exist
  - Ψ_{perp} = Perpetual motion machines exist

Perpetual Machines

Perpetual Machines

- Let Γ represent an axiomatization of all existing laws of physics

Perpetual Machines

- Let Γ represent an axiomatization of all existing laws of physics
- Then we have at least

Perpetual Machines

- Let Γ represent an axiomatization of all existing laws of physics
- Then we have at least

$$\Gamma \vdash \neg\psi_{perp}$$

$$\Gamma \vdash \diamond\psi_{hyper}$$

Theoretical Richness

Theoretical Richness

- Turing computation theory => Complexity theory & Turing degrees

Theoretical Richness

- Turing computation theory => Complexity theory & Turing degrees
- Hypercomputation => Mathematical models of hypercomputers, ``concretizes'' Turing degrees

Theoretical Richness

- Turing computation theory => Complexity theory & Turing degrees
- Hypercomputation => Mathematical models of hypercomputers, ``concretizes" Turing degrees
  - Hypercomputation could possibly be useful in formal learning theory

Theoretical Richness

- Turing computation theory => Complexity theory & Turing degrees
- Hypercomputation => Mathematical models of hypercomputers, ``concretizes'' Turing degrees
    - Hypercomputation could possibly be useful in formal learning theory

Theoretical Richness

- Turing computation theory => Complexity theory & Turing degrees
- Hypercomputation => Mathematical models of hypercomputers "concretizes" Turing degrees
    - Hypercomputation could possibly be useful in formal learning theory

Consideration of non-computable scientists thereby facilitates the analysis of proofs, making it clearer which assumptions carry the burden. (Jain et al. “Systems that Learn ”1999, 35)

Some Objections ...

Objection 1

Objection 1

- *"You write that 'A rather large number of physical and mathematical models of hypercomputation have been put forward so far.' Well yes, but none of them, when actually physically implemented, can do anything that a Turing machine couldn't so far. If yes, show me the function. The burden of proof is of course on your side!"*

Our response

Our response

- Just a recapitulation of Davis' argument against η_2

Our response

- Just a recapitulation of Davis' argument against η_2
- Even if a physical hypercomputer is **impossible** in the actual world, the hypercomputation field **survives** via the abstract mathematical and cognitive domains (η_1 and η_3)

Objection 2

Objection 2

- *"You tell us that 'Davis ignores numerous other models of hypercomputation.' Yes, but because they all boil down to infinite resources in some form, infinite time, or some other wacky stuff."*

Our response: Flaw 1

Our response: Flaw 1

- The objection is a blanket statement that all hypercomputation models require infinitary processing.

Our response: Flaw 1

- The objection is a blanket statement that all hypercomputation models require infinitary processing.
  - Unwarranted statement without a proof.

Our response: Flaw 1

- The objection is a blanket statement that all hypercomputation models require infinitary processing.
  - Unwarranted statement without a proof.
  - From the CTT experience, difficult to prove such statements

Our response: Flaw 1

- The objection is a blanket statement that all hypercomputation models require infinitary processing.
  - Unwarranted statement without a proof.
  - From the CTT experience, difficult to prove such statements
  - Only a mythical belief exists in that all hypercomputation require infinite resources.

Our response: Flaw 2

Our response: Flaw 2

- Almost all formal sciences deal with infinite structures and processes.

Our response: Flaw 2

- Almost all formal sciences deal with infinite structures and processes.
- Our critic will then label infinitary logics as wacky.

Our response: Flaw 2

- Almost all formal sciences deal with infinite structures and processes.
- Our critic will then label infinitary logics as wacky.
    - Infinitary logics are essential for formal mathematics.

Our response: Flaw 2

- Almost all formal sciences deal with infinite structures and processes.
- Our critic will then label infinitary logics as wacky.
  - Infinitary logics are essential for formal mathematics.
    - E.g. characterizing abelian group properties.

Our response: Flaw 3

Our response: Flaw 3

- *“You tell us that ‘Davis ignores numerous other models of hypercomputation.’ Yes, but because they all boil down to infinite resources in some form, infinite time, or some other wacky stuff.”*

Our response: Flaw 3

- *"You tell us that 'Davis ignores numerous other models of hypercomputation.' Yes, but because they all boil down to infinite resources in some form, infinite time, or some other wacky stuff."*

Our response: Flaw 3

- *"You tell us that 'Davis ignores numerous other models of hypercomputation.' Yes, but because they all boil down to infinite resources in some form, infinite time, or some other wacky stuff."*

Our response: Flaw 3

- *“You tell us that ‘Davis ignores numerous other models of hypercomputation.’ Yes, but, because they all boil down to infinite resources in some form, infinite time, or some other wacky stuff.”*

So even Turing machines are “wacky” as they require an <u>infinite</u> tape!

Objection 3

Objection 3

- *“Do the authors really believe that the accelerating TM is a model worth mentioning as a physically plausible hypercomputer?!”*

Our response

Our response

- Yes. We believe it is **physically plausible**.

Our response

- Yes. We believe it is **physically plausible**.
    - This has little to do with flaws in Davis' reasoning.

Our response

- Yes. We believe it is **physically plausible**.
  - This has little to do with flaws in Davis' reasoning.
- Support for our belief:

Our response

- Yes. We believe it is **physically plausible**.
    - This has little to do with flaws in Davis' reasoning.
- Support for our belief:
    - Xia's result that, under certain conditions, *a body can be accelerated to infinite time*.

Our response

- Yes. We believe it is **physically plausible**.
    - This has little to do with flaws in Davis' reasoning.
- Support for our belief:
    - Xia's result that, under certain conditions, *a body can be accelerated to infinite time*.
- Even if we are wrong that the model is physically possible:

Our response

- Yes. We believe it is **physically plausible**.
    - This has little to do with flaws in Davis' reasoning.
- Support for our belief:
    - Xia's result that, under certain conditions, *a body can be accelerated to infinite time*.
- Even if we are wrong that the model is physically possible:
    - We are sure that it's **logically possible** that it's **physically possible**.

Our response

- Yes. We believe it is **physically plausible**.
    - This has little to do with flaws in Davis' reasoning.
- Support for our belief:
    - Xia's result that, under certain conditions, *a body can be accelerated to infinite time*.
- Even if we are wrong that the model is physically possible:
    - We are sure that it's **logically possible** that it's **physically possible**.
    - The above proposition is enough for our defense against Davis' stand.