Prediction and MDL for infinite sequences

Alexey Milovanov

AIXI lab

almas239@gmail.com

UAIASI reading group
9 November 2026

Motivation

Suppose an AI performs well in **99% of cases**, but very badly in the remaining **1%**.

Suppose also that we cannot tell in advance which tasks it will handle well and which will lead to failure.

If failure can mean **catastrophe**, this is a dangerous situation:

- It is psychologically hard to give up a tool that usually works well. Warnings alone may not be enough.
- A wrapper could make an AI refuse some requests. But it seems unrealistic to ensure that *every AI everywhere* has such a safeguard.

Motivation

Suppose an AI performs well in **99% of cases**, but very badly in the remaining **1%**.

Suppose also that we cannot tell in advance which tasks it will handle well and which will lead to failure.

If failure can mean **catastrophe**, this is a dangerous situation:

- It is psychologically hard to give up a tool that usually works well. Warnings alone may not be enough.
- A wrapper could make an AI refuse some requests. But it seems unrealistic to ensure that *every AI everywhere* has such a safeguard.

Motivation

Suppose an AI performs well in **99% of cases**, but very badly in the remaining **1%**.

Suppose also that we cannot tell in advance which tasks it will handle well and which will lead to failure.

If failure can mean **catastrophe**, this is a dangerous situation:

- It is psychologically hard to give up a tool that usually works well. Warnings alone may not be enough.
- A wrapper could make an AI refuse some requests. But it seems unrealistic to ensure that *every AI everywhere* has such a safeguard.

Motivation

Suppose an AI performs well in **99% of cases**, but very badly in the remaining **1%**.

Suppose also that we cannot tell in advance which tasks it will handle well and which will lead to failure.

If failure can mean **catastrophe**, this is a dangerous situation:

- It is psychologically hard to give up a tool that usually works well. Warnings alone may not be enough.
- A wrapper could make an AI refuse some requests. But it seems unrealistic to ensure that *every AI everywhere* has such a safeguard.

Motivation

Suppose an AI performs well in **99% of cases**, but very badly in the remaining **1%**.

Suppose also that we cannot tell in advance which tasks it will handle well and which will lead to failure.

If failure can mean **catastrophe**, this is a dangerous situation:

- It is psychologically hard to give up a tool that usually works well. Warnings alone may not be enough.
- A wrapper could make an AI refuse some requests. But it seems unrealistic to ensure that *every AI everywhere* has such a safeguard.

Motivation

Suppose an AI performs well in **99% of cases**, but very badly in the remaining **1%**.

Suppose also that we cannot tell in advance which tasks it will handle well and which will lead to failure.

If failure can mean **catastrophe**, this is a dangerous situation:

- It is psychologically hard to give up a tool that usually works well. Warnings alone may not be enough.
- A wrapper could make an AI refuse some requests. But it seems unrealistic to ensure that *every AI everywhere* has such a safeguard.

Challenge

Can we build a **safe AI** that:

1. performs at least as well as an ordinary AI;
2. avoids unexpectedly severe failures on otherwise typical tasks?

Challenge

Can we build a **safe AI** that:

1. performs at least as well as an ordinary AI;
2. avoids unexpectedly severe failures on otherwise typical tasks?

Challenge

Can we build a **safe AI** that:

1. performs at least as well as an ordinary AI;
2. avoids unexpectedly severe failures on otherwise typical tasks?

Challenge

Can we build a **safe AI** that:

1. performs at least as well as an ordinary AI;
2. avoids unexpectedly severe failures on otherwise typical tasks?

Sources, forecasts and prediction error

A probability measure P is specified by its cylinder probabilities:

$$P(\varepsilon) = 1, \qquad P(x0) + P(x1) = P(x).$$

$P(x)$ is the probability that the output starts with x.

At a history x with $P(x) > 0$, the target is

$$p(x) = P(1 \mid x) = \frac{P(x1)}{P(x)}.$$

For a predictor h, measure its total error along ω by

$$L_h(\omega, P) = \sum_{n \geq 0} \Big(h(\omega_{1:n}) - p(\omega_{1:n}) \Big)^2.$$

Sources, forecasts and prediction error

A probability measure P is specified by its cylinder probabilities:

$$P(\varepsilon) = 1, \qquad P(x0) + P(x1) = P(x).$$

$P(x)$ is the probability that the output starts with x.

At a history x with $P(x) > 0$, the target is

$$p(x) = P(1 \mid x) = \frac{P(x1)}{P(x)}.$$

For a predictor h, measure its total error along ω by

$$L_h(\omega, P) = \sum_{n \geq 0} \Big(h(\omega_{1:n}) - p(\omega_{1:n}) \Big)^2.$$

Sources, forecasts and prediction error

A probability measure P is specified by its cylinder probabilities:

$$P(\varepsilon) = 1, \qquad P(x0) + P(x1) = P(x).$$

$P(x)$ is the probability that the output starts with x.

At a history x with $P(x) > 0$, the target is

$$p(x) = P(1 \mid x) = \frac{P(x1)}{P(x)}.$$

For a predictor h, measure its total error along ω by

$$L_h(\omega, P) = \sum_{n \geq 0} \Big(h(\omega_{1:n}) - p(\omega_{1:n}) \Big)^2.$$

Sources, forecasts and prediction error

A probability measure P is specified by its cylinder probabilities:

$$P(\varepsilon) = 1, \qquad P(x0) + P(x1) = P(x).$$

$P(x)$ is the probability that the output starts with x.

At a history x with $P(x) > 0$, the target is

$$p(x) = P(1 \mid x) = \frac{P(x1)}{P(x)}.$$

For a predictor h, measure its total error along ω by

$$L_h(\omega, P) = \sum_{n \geq 0} \Big(h(\omega_{1:n}) - p(\omega_{1:n}) \Big)^2.$$

Solomonoff induction

A **tree semimeasure** may lose mass:

$$\nu(\varepsilon) \le 1, \qquad \nu(x0) + \nu(x1) \le \nu(x).$$

This allows a program to stop producing output. Choose a universal program mixture M with the dominance property: for every computable P,

$$M(x) \ge 2^{-K(P)} P(x).$$

$K(P)$ is the length of a shortest self-delimiting program computing P.

The normalized next-bit forecast is

$$m(1 \mid x) = \frac{M(x1)}{M(x0) + M(x1)}.$$

The raw forecast is $M(x1)/M(x)$. Both rules are noncomputable.

Solomonoff induction

A **tree semimeasure** may lose mass:

$$\nu(\varepsilon) \le 1, \qquad \nu(x0) + \nu(x1) \le \nu(x).$$

This allows a program to stop producing output. Choose a universal program mixture M with the dominance property: for every computable P,

$$M(x) \ge 2^{-K(P)} P(x).$$

$K(P)$ is the length of a shortest self-delimiting program computing P.

The normalized next-bit forecast is

$$m(1 \mid x) = \frac{M(x1)}{M(x0) + M(x1)}.$$

The raw forecast is $M(x1)/M(x)$. Both rules are noncomputable.

Solomonoff induction

A **tree semimeasure** may lose mass:

$$\nu(\varepsilon) \le 1, \qquad \nu(x0) + \nu(x1) \le \nu(x).$$

This allows a program to stop producing output. Choose a universal program mixture M with the dominance property: for every computable P,

$$M(x) \ge 2^{-K(P)} P(x).$$

$K(P)$ is the length of a shortest self-delimiting program computing P.

The normalized next-bit forecast is

$$m(1 \mid x) = \frac{M(x1)}{M(x0) + M(x1)}.$$

The raw forecast is $M(x1)/M(x)$. Both rules are noncomputable.

Solomonoff induction

A **tree semimeasure** may lose mass:

$$\nu(\varepsilon) \le 1, \qquad \nu(x0) + \nu(x1) \le \nu(x).$$

This allows a program to stop producing output. Choose a universal program mixture M with the dominance property: for every computable P,

$$M(x) \ge 2^{-K(P)} P(x).$$

$K(P)$ is the length of a shortest self-delimiting program computing P.

The normalized next-bit forecast is

$$m(1 \mid x) = \frac{M(x1)}{M(x0) + M(x1)}.$$

The raw forecast is $M(x1)/M(x)$. Both rules are noncomputable.

Solomonoff: a strong guarantee in expectation

The total squared error of the normalized Solomonoff predictor is

$$L_m(\omega, P) = \sum_{n \geq 0} \Big[m(1 \mid \omega_{1:n}) - P(1 \mid \omega_{1:n}) \Big]^2.$$

Theorem 1 (Solomonoff, 1978)

For every computable probability measure P,

$$\mathbb{E}_P L_m = O\Big(K(P)\Big).$$

Consequently, the prediction error tends to zero on P-almost every sequence.

Solomonoff, Complexity-Based Induction Systems (1978), Theorem 3.

Solomonoff: a strong guarantee in expectation

The total squared error of the normalized Solomonoff predictor is

$$L_m(\omega, P) = \sum_{n \geq 0} \Big[m(1 \mid \omega_{1:n}) - P(1 \mid \omega_{1:n}) \Big]^2.$$

Theorem 1 (Solomonoff, 1978)

For every computable probability measure P,

$$\mathbb{E}_P L_m = O\Big(K(P)\Big).$$

Consequently, the prediction error tends to zero on P-almost every sequence.

Solomonoff, Complexity-Based Induction Systems (1978), Theorem 3.

Solomonoff: a strong guarantee in expectation

The total squared error of the normalized Solomonoff predictor is

$$L_m(\omega, P) = \sum_{n \geq 0} \Big[m(1 \mid \omega_{1:n}) - P(1 \mid \omega_{1:n}) \Big]^2.$$

Theorem 1 (Solomonoff, 1978)

For every computable probability measure P,

$$\mathbb{E}_P L_m = O\Big(K(P)\Big).$$

Consequently, the prediction error tends to zero on P-almost every sequence.

Solomonoff, Complexity-Based Induction Systems (1978), Theorem 3.

Solomonoff: a strong guarantee in expectation

The total squared error of the normalized Solomonoff predictor is

$$L_m(\omega, P) = \sum_{n \geq 0} \Big[m(1 \mid \omega_{1:n}) - P(1 \mid \omega_{1:n}) \Big]^2.$$

Theorem 1 (Solomonoff, 1978)

For every computable probability measure P,

$$\mathbb{E}_P L_m = O\Big(K(P)\Big).$$

Consequently, the prediction error tends to zero on P-almost every sequence.

Solomonoff, Complexity-Based Induction Systems (1978), Theorem 3.

Martin-Löf randomness: why we need a definition

Would these look like typical outcomes of fair-coin tossing?

$$000000\ldots \qquad \text{or} \qquad 010101\ldots,$$

Yet every n-bit string has the same probability 2^{-n}, and every individual infinite sequence has probability zero.

Correct frequencies are not enough either: $010101\ldots$ has a limiting frequency of ones equal to $1/2$, but is periodic.

Martin-Löf's idea

To define a typical sequence, exclude only null sets whose smallness can be demonstrated algorithmically.

Martin-Löf randomness: why we need a definition

Would these look like typical outcomes of fair-coin tossing?

$$000000\ldots \qquad \text{or} \qquad 010101\ldots,$$

Yet every n-bit string has the same probability 2^{-n}, and every individual infinite sequence has probability zero.

Correct frequencies are not enough either: $010101\ldots$ has a limiting frequency of ones equal to $1/2$, but is periodic.

Martin-Löf's idea

To define a typical sequence, exclude only null sets whose smallness can be demonstrated algorithmically.

Martin-Löf randomness: why we need a definition

Would these look like typical outcomes of fair-coin tossing?

$$000000\ldots \qquad \text{or} \qquad 010101\ldots,$$

Yet every n-bit string has the same probability 2^{-n}, and every individual infinite sequence has probability zero.

Correct frequencies are not enough either: $010101\ldots$ has a limiting frequency of ones equal to $1/2$, but is periodic.

Martin-Löf's idea

To define a typical sequence, exclude only null sets whose smallness can be demonstrated algorithmically.

Martin-Löf randomness: why we need a definition

Would these look like typical outcomes of fair-coin tossing?

$$000000\ldots \qquad \text{or} \qquad 010101\ldots,$$

Yet every n-bit string has the same probability 2^{-n}, and every individual infinite sequence has probability zero.

Correct frequencies are not enough either: $010101\ldots$ has a limiting frequency of ones equal to $1/2$, but is periodic.

Martin-Löf's idea

To define a typical sequence, exclude only null sets whose smallness can be demonstrated algorithmically.

Martin-Löf randomness: why we need a definition

Would these look like typical outcomes of fair-coin tossing?

$$000000\ldots \qquad \text{or} \qquad 010101\ldots,$$

Yet every n-bit string has the same probability 2^{-n}, and every individual infinite sequence has probability zero.

Correct frequencies are not enough either: $010101\ldots$ has a limiting frequency of ones equal to $1/2$, but is periodic.

Martin-Löf's idea

To define a typical sequence, exclude only null sets whose smallness can be demonstrated algorithmically.

Martin-Löf randomness: an individual-sequence question

A Martin-Löf test for P is an effective sequence of open sets U_j with

$$P(U_j) \leq 2^{-j}.$$

A sequence is **P-ML-random** if it avoids the intersection of every such test. The P-ML-random sequences have P-measure one.

Equivalently, fix a universal lower semicomputable semimeasure M and put

$$D(\omega \mid P) = \max \left\{ 0, \sup_n \log_2 \frac{M(\omega_{1:n})}{P(\omega_{1:n})} \right\}.$$

Then ω is P-ML-random exactly when $D(\omega \mid P) < \infty$. D is a randomness deficiency: a large value means that some prefix is much better explained by M than by P.

Martin-Löf randomness: an individual-sequence question

A Martin-Löf test for P is an effective sequence of open sets U_j with

$$P(U_j) \leq 2^{-j}.$$

A sequence is P-**ML-random** if it avoids the intersection of every such test. The P-ML-random sequences have P-measure one.

Equivalently, fix a universal lower semicomputable semimeasure M and put

$$D(\omega \mid P) = \max \left\{ 0, \sup_n \log_2 \frac{M(\omega_{1:n})}{P(\omega_{1:n})} \right\}.$$

Then ω is P-ML-random exactly when $D(\omega \mid P) < \infty$. D is a randomness deficiency: a large value means that some prefix is much better explained by M than by P.

Martin-Löf randomness: an individual-sequence question

A Martin-Löf test for P is an effective sequence of open sets U_j with

$$P(U_j) \leq 2^{-j}.$$

A sequence is **P-ML-random** if it avoids the intersection of every such test. The P-ML-random sequences have P-measure one.

Equivalently, fix a universal lower semicomputable semimeasure M and put

$$D(\omega \mid P) = \max \left\{ 0, \sup_n \log_2 \frac{M(\omega_{1:n})}{P(\omega_{1:n})} \right\}.$$

Then ω is P-ML-random exactly when $D(\omega \mid P) < \infty$. D is a randomness deficiency: a large value means that some prefix is much better explained by M than by P.

Martin-Löf randomness: an individual-sequence question

A Martin-Löf test for P is an effective sequence of open sets U_j with

$$P(U_j) \leq 2^{-j}.$$

A sequence is **P-ML-random** if it avoids the intersection of every such test. The P-ML-random sequences have P-measure one.

Equivalently, fix a universal lower semicomputable semimeasure M and put

$$D(\omega \mid P) = \max \left\{ 0, \sup_n \log_2 \frac{M(\omega_{1:n})}{P(\omega_{1:n})} \right\}.$$

Then ω is P-ML-random exactly when $D(\omega \mid P) < \infty$. D is a randomness deficiency: a large value means that some prefix is much better explained by M than by P.

Martin-Löf randomness: an individual-sequence question

A Martin-Löf test for P is an effective sequence of open sets U_j with

$$P(U_j) \leq 2^{-j}.$$

A sequence is **P-ML-random** if it avoids the intersection of every such test. The P-ML-random sequences have P-measure one.

Equivalently, fix a universal lower semicomputable semimeasure M and put

$$D(\omega \mid P) = \max \left\{ 0, \sup_n \log_2 \frac{M(\omega_{1:n})}{P(\omega_{1:n})} \right\}.$$

Then ω is P-ML-random exactly when $D(\omega \mid P) < \infty$. D is a randomness deficiency: a large value means that some prefix is much better explained by M than by P.

Why almost-sure convergence does not settle this

Almost sure	convergence outside a null set
On every ML-random sequence	convergence outside an effective null set

An exceptional set of probability zero can still contain Martin-Löf-random sequences.

Solomonoff's predictor is not computable. Its bad-prediction events do not automatically give an effective randomness test.

The stronger question

Does $m(1 \mid \omega_{1:n}) - P(1 \mid \omega_{1:n}) \to 0$ for every P-ML-random ω?

Why almost-sure convergence does not settle this

Almost sure	convergence outside a null set
On every ML-random sequence	convergence outside an effective null set

An exceptional set of probability zero can still contain Martin-Löf-random sequences.

Solomonoff's predictor is not computable. Its bad-prediction events do not automatically give an effective randomness test.

The stronger question

Does $m(1 \mid \omega_{1:n}) - P(1 \mid \omega_{1:n}) \to 0$ for every P-ML-random ω?

Why almost-sure convergence does not settle this

Almost sure	convergence outside a null set
On every ML-random sequence	convergence outside an effective null set

An exceptional set of probability zero can still contain Martin-Löf-random sequences.

Solomonoff's predictor is not computable. Its bad-prediction events do not automatically give an effective randomness test.

The stronger question

Does $m(1 \mid \omega_{1:n}) - P(1 \mid \omega_{1:n}) \to 0$ for every P-ML-random ω?

Why almost-sure convergence does not settle this

Almost sure	convergence outside a null set
On every ML-random sequence	convergence outside an effective null set

An exceptional set of probability zero can still contain Martin-Löf-random sequences.

Solomonoff's predictor is not computable. Its bad-prediction events do not automatically give an effective randomness test.

The stronger question

Does $m(1 \mid \omega_{1:n}) - P(1 \mid \omega_{1:n}) \to 0$ for every P-ML-random ω?

Why almost-sure convergence does not settle this

Almost sure	convergence outside a null set
On every ML-random sequence	convergence outside an effective null set

An exceptional set of probability zero can still contain Martin-Löf-random sequences.

Solomonoff's predictor is not computable. Its bad-prediction events do not automatically give an effective randomness test.

The stronger question

Does $m(1 \mid \omega_{1:n}) - P(1 \mid \omega_{1:n}) \to 0$ for every P-ML-random ω?

The negative results

Write $\lambda(x) = 2^{-|x|}$ for the fair-coin measure.

Theorem 2 (Hutter–Muchnik, 2004/2007)

For some universal lower semicomputable M and some λ-ML-random ω, the raw forecasts $M(\omega_n \mid \omega_{<n})$ do not converge to $1/2$.

Theorem 3 (Lattimore–Hutter, 2015)

Every universal mixture M has a λ-ML-random sequence on which its forecasts fail to converge to $1/2$. This holds for both raw and normalized forecasts.

Lattimore–Hutter, Theorems 6 and 13.

The negative results

Write $\lambda(x) = 2^{-|x|}$ for the fair-coin measure.

Theorem 2 (Hutter–Muchnik, 2004/2007)

For some universal lower semicomputable M and some λ-ML-random ω, the raw forecasts $M(\omega_n \mid \omega_{<n})$ do not converge to $1/2$.

Theorem 3 (Lattimore–Hutter, 2015)

Every universal mixture M has a λ-ML-random sequence on which its forecasts fail to converge to $1/2$. This holds for both raw and normalized forecasts.

Lattimore–Hutter, Theorems 6 and 13.

The negative results

Write $\lambda(x) = 2^{-|x|}$ for the fair-coin measure.

Theorem 2 (Hutter–Muchnik, 2004/2007)

For some universal lower semicomputable M and some λ-ML-random ω, the raw forecasts $M(\omega_n \mid \omega_{<n})$ do not converge to $1/2$.

Theorem 3 (Lattimore–Hutter, 2015)

Every universal mixture M has a λ-ML-random sequence on which its forecasts fail to converge to $1/2$. This holds for both raw and normalized forecasts.

Lattimore–Hutter, Theorems 6 and 13.

The negative results

Write $\lambda(x) = 2^{-|x|}$ for the fair-coin measure.

Theorem 2 (Hutter–Muchnik, 2004/2007)

For some universal lower semicomputable M and some λ-ML-random ω, the raw forecasts $M(\omega_n \mid \omega_{<n})$ do not converge to $1/2$.

Theorem 3 (Lattimore–Hutter, 2015)

Every universal mixture M has a λ-ML-random sequence on which its forecasts fail to converge to $1/2$. This holds for both raw and normalized forecasts.

Lattimore–Hutter, Theorems 6 and 13.

MDL: choose an explanation, then predict with it

MDL balances two costs:

$$\underbrace{K(Q)}_{\text{describe the model}} + \underbrace{-\log_2 Q(x)}_{\text{describe the data using the model}}.$$

For prediction, choose a *computable probability measure* Q explaining the observed history x.

We allow a stronger penalty on the model:

$$Q_x^{(\alpha)} \in \arg\min_Q \left[\alpha K(Q) - \log_2 Q(x)\right], \qquad \alpha \geq 1.$$

Then predict $h_\alpha(x) = Q_x^{(\alpha)}(1 \mid x)$.

The model is reselected after every observed bit.

MDL: choose an explanation, then predict with it

MDL balances two costs:

$$\underbrace{K(Q)}_{\text{describe the model}} + \underbrace{-\log_2 Q(x)}_{\text{describe the data using the model}} .$$

For prediction, choose a *computable probability measure* Q explaining the observed history x.

We allow a stronger penalty on the model:

$$Q_x^{(\alpha)} \in \arg\min_Q \left[\alpha K(Q) - \log_2 Q(x)\right], \qquad \alpha \geq 1.$$

Then predict $h_\alpha(x) = Q_x^{(\alpha)}(1 \mid x)$.

The model is reselected after every observed bit.

MDL: choose an explanation, then predict with it

MDL balances two costs:

$$\underbrace{K(Q)}_{\text{describe the model}} + \underbrace{-\log_2 Q(x)}_{\text{describe the data using the model}} .$$

For prediction, choose a *computable probability measure* Q explaining the observed history x.

We allow a stronger penalty on the model:

$$Q_x^{(\alpha)} \in \arg\min_Q \left[\alpha K(Q) - \log_2 Q(x)\right], \qquad \alpha \geq 1.$$

Then predict $h_\alpha(x) = Q_x^{(\alpha)}(1 \mid x)$.

The model is reselected after every observed bit.

MDL: choose an explanation, then predict with it

MDL balances two costs:

$$\underbrace{K(Q)}_{\text{describe the model}} + \underbrace{-\log_2 Q(x)}_{\text{describe the data using the model}}.$$

For prediction, choose a *computable probability measure* Q explaining the observed history x.

We allow a stronger penalty on the model:

$$Q_x^{(\alpha)} \in \arg\min_Q \left[\alpha K(Q) - \log_2 Q(x)\right], \qquad \alpha \geq 1.$$

Then predict $h_\alpha(x) = Q_x^{(\alpha)}(1 \mid x)$.

The model is reselected after every observed bit.

MDL: choose an explanation, then predict with it

MDL balances two costs:

$$\underbrace{K(Q)}_{\text{describe the model}} + \underbrace{-\log_2 Q(x)}_{\text{describe the data using the model}}.$$

For prediction, choose a *computable probability measure* Q explaining the observed history x.

We allow a stronger penalty on the model:

$$Q_x^{(\alpha)} \in \arg\min_Q \left[\alpha K(Q) - \log_2 Q(x)\right], \qquad \alpha \geq 1.$$

Then predict $h_\alpha(x) = Q_x^{(\alpha)}(1 \mid x)$.

The model is reselected after every observed bit.

What is the difference between $\alpha = 1$ and $\alpha > 1$?

Let x be an incompressible n-bit string, so $K(x) \approx n$. Compare the fair-coin measure λ with a singleton model Q_x that outputs $x000\ldots$ with probability one.

Ignoring logarithmic terms, their scores are:

Model	$K(Q)$	$-\log_2 Q(x)$	$\alpha = 1$	$\alpha > 1$
Fair coin λ	$O(1)$	n	$\approx n$	$\approx n$
Singleton Q_x	$\approx n$	0	$\approx n$	$\approx \alpha n$

At $\alpha = 1$, the scores do not clearly distinguish the two explanations. At $\alpha > 1$, the simpler fair-coin model is preferred.

Intuitively, $\alpha > 1$ should give better predictions. **Can we support this intuition mathematically?**

What is the difference between $\alpha = 1$ and $\alpha > 1$?

Let x be an incompressible n-bit string, so $K(x) \approx n$. Compare the fair-coin measure λ with a singleton model Q_x that outputs $x000\ldots$ with probability one.

Ignoring logarithmic terms, their scores are:

Model	$K(Q)$	$-\log_2 Q(x)$	$\alpha = 1$	$\alpha > 1$
Fair coin λ	$O(1)$	n	$\approx n$	$\approx n$
Singleton Q_x	$\approx n$	0	$\approx n$	$\approx \alpha n$

At $\alpha = 1$, the scores do not clearly distinguish the two explanations. At $\alpha > 1$, the simpler fair-coin model is preferred.

Intuitively, $\alpha > 1$ should give better predictions. **Can we support this intuition mathematically?**

What is the difference between $\alpha = 1$ and $\alpha > 1$?

Let x be an incompressible n-bit string, so $K(x) \approx n$. Compare the fair-coin measure λ with a singleton model Q_x that outputs $x000\ldots$ with probability one.

Ignoring logarithmic terms, their scores are:

Model	$K(Q)$	$-\log_2 Q(x)$	$\alpha = 1$	$\alpha > 1$
Fair coin λ	$O(1)$	n	$\approx n$	$\approx n$
Singleton Q_x	$\approx n$	0	$\approx n$	$\approx \alpha n$

At $\alpha = 1$, the scores do not clearly distinguish the two explanations. At $\alpha > 1$, the simpler fair-coin model is preferred.

Intuitively, $\alpha > 1$ should give better predictions. **Can we support this intuition mathematically?**

What is the difference between $\alpha = 1$ and $\alpha > 1$?

Let x be an incompressible n-bit string, so $K(x) \approx n$. Compare the fair-coin measure λ with a singleton model Q_x that outputs $x000\ldots$ with probability one.

Ignoring logarithmic terms, their scores are:

Model	$K(Q)$	$-\log_2 Q(x)$	$\alpha = 1$	$\alpha > 1$
Fair coin λ	$O(1)$	n	$\approx n$	$\approx n$
Singleton Q_x	$\approx n$	0	$\approx n$	$\approx \alpha n$

At $\alpha = 1$, the scores do not clearly distinguish the two explanations. At $\alpha > 1$, the simpler fair-coin model is preferred.

Intuitively, $\alpha > 1$ should give better predictions. **Can we support this intuition mathematically?**

What is the difference between $\alpha = 1$ and $\alpha > 1$?

Let x be an incompressible n-bit string, so $K(x) \approx n$. Compare the fair-coin measure λ with a singleton model Q_x that outputs $x000\ldots$ with probability one.

Ignoring logarithmic terms, their scores are:

Model	$K(Q)$	$-\log_2 Q(x)$	$\alpha = 1$	$\alpha > 1$
Fair coin λ	$O(1)$	n	$\approx n$	$\approx n$
Singleton Q_x	$\approx n$	0	$\approx n$	$\approx \alpha n$

At $\alpha = 1$, the scores do not clearly distinguish the two explanations. At $\alpha > 1$, the simpler fair-coin model is preferred.

Intuitively, $\alpha > 1$ should give better predictions. **Can we support this intuition mathematically?**

MDL already has an expected-error guarantee

Let $\mathcal{C}$ be a countable class of probability measures, with weights $w_Q > 0$ and $\sum_Q w_Q \le 1$. At x, choose $Q_x \in \arg\max_Q w_Q Q(x)$ and use its conditional forecast.

Theorem 4 (Poland–Hutter, 2005)

For every true source $P \in \mathcal{C}$,

$$\mathbb{E}_P \sum_{n \ge 0} \sum_{b \in \{0,1\}} \Big[Q_{X_{1:n}}(b \mid X_{1:n}) - P(b \mid X_{1:n}) \Big]^2 \le \frac{21}{w_P}.$$

For $w_Q = 2^{-\alpha K(Q)}$, this is $O(2^{\alpha K(P)})$. It gives almost-sure convergence for every $\alpha \ge 1$.

Poland–Hutter, Asymptotics of Discrete MDL for Online Prediction, Corollaries 13–14.

MDL already has an expected-error guarantee

Let $\mathcal{C}$ be a countable class of probability measures, with weights $w_Q > 0$ and $\sum_Q w_Q \le 1$. At x, choose $Q_x \in \arg\max_Q w_Q Q(x)$ and use its conditional forecast.

Theorem 4 (Poland–Hutter, 2005)

For every true source $P \in \mathcal{C}$,

$$\mathbb{E}_P \sum_{n \ge 0} \sum_{b \in \{0,1\}} \Big[Q_{X_{1:n}}(b \mid X_{1:n}) - P(b \mid X_{1:n}) \Big]^2 \le \frac{21}{w_P}.$$

For $w_Q = 2^{-\alpha K(Q)}$, this is $O(2^{\alpha K(P)})$. It gives almost-sure convergence for every $\alpha \ge 1$.

Poland–Hutter, Asymptotics of Discrete MDL for Online Prediction, Corollaries 13–14.

MDL already has an expected-error guarantee

Let $\mathcal{C}$ be a countable class of probability measures, with weights $w_Q > 0$ and $\sum_Q w_Q \le 1$. At x, choose $Q_x \in \arg\max_Q w_Q Q(x)$ and use its conditional forecast.

Theorem 4 (Poland–Hutter, 2005)

For every true source $P \in \mathcal{C}$,

$$\mathbb{E}_P \sum_{n \ge 0} \sum_{b \in \{0,1\}} \Big[Q_{X_{1:n}}(b \mid X_{1:n}) - P(b \mid X_{1:n}) \Big]^2 \le \frac{21}{w_P}.$$

For $w_Q = 2^{-\alpha K(Q)}$, this is $O(2^{\alpha K(P)})$. It gives almost-sure convergence for every $\alpha \ge 1$.

Poland–Hutter, Asymptotics of Discrete MDL for Online Prediction, Corollaries 13–14.

MDL already has an expected-error guarantee

Let $\mathcal{C}$ be a countable class of probability measures, with weights $w_Q > 0$ and $\sum_Q w_Q \le 1$. At x, choose $Q_x \in \arg\max_Q w_Q Q(x)$ and use its conditional forecast.

Theorem 4 (Poland–Hutter, 2005)

For every true source $P \in \mathcal{C}$,

$$\mathbb{E}_P \sum_{n \ge 0} \sum_{b \in \{0,1\}} \Big[Q_{X_{1:n}}(b \mid X_{1:n}) - P(b \mid X_{1:n}) \Big]^2 \le \frac{21}{w_P}.$$

For $w_Q = 2^{-\alpha K(Q)}$, this is $O(2^{\alpha K(P)})$. It gives almost-sure convergence for every $\alpha \ge 1$.

Poland–Hutter, Asymptotics of Discrete MDL for Online Prediction, Corollaries 13–14.

MDL converges on every ML-random sequence

The candidate class is now **all computable probability measures**. Define the total squared prediction error along ω by

$$L_\alpha(\omega, P) = \sum_{n=0}^{\infty} \Big[h_\alpha(\omega_{1:n}) - P(1 \mid \omega_{1:n}) \Big]^2.$$

Write $k = K(P)$ and $D = D(\omega \mid P)$.

Theorem 5 (M.)

For every fixed $\alpha > 1$ and every P-ML-random ω,

$$L_\alpha(\omega, P) < \infty.$$

In particular, $h_\alpha(\omega_{1:n}) - P(1 \mid \omega_{1:n}) \to 0$.

MDL converges on every ML-random sequence

The candidate class is now **all computable probability measures**. Define the total squared prediction error along ω by

$$L_\alpha(\omega, P) = \sum_{n=0}^{\infty} \Big[h_\alpha(\omega_{1:n}) - P(1 \mid \omega_{1:n}) \Big]^2.$$

Write $k = K(P)$ and $D = D(\omega \mid P)$.

Theorem 5 (M.)

For every fixed $\alpha > 1$ and every P-ML-random ω,

$$L_\alpha(\omega, P) < \infty.$$

In particular, $h_\alpha(\omega_{1:n}) - P(1 \mid \omega_{1:n}) \to 0$.

MDL converges on every ML-random sequence

The candidate class is now **all computable probability measures**. Define the total squared prediction error along ω by

$$L_\alpha(\omega, P) = \sum_{n=0}^{\infty} \Big[h_\alpha(\omega_{1:n}) - P(1 \mid \omega_{1:n}) \Big]^2.$$

Write $k = K(P)$ and $D = D(\omega \mid P)$.

Theorem 5 (M.)

For every fixed $\alpha > 1$ and every P-ML-random ω,

$$L_\alpha(\omega, P) < \infty.$$

In particular, $h_\alpha(\omega_{1:n}) - P(1 \mid \omega_{1:n}) \to 0$.

MDL converges on every ML-random sequence

The candidate class is now **all computable probability measures**. Define the total squared prediction error along ω by

$$L_\alpha(\omega, P) = \sum_{n=0}^{\infty} \Big[h_\alpha(\omega_{1:n}) - P(1 \mid \omega_{1:n}) \Big]^2.$$

Write $k = K(P)$ and $D = D(\omega \mid P)$.

Theorem 5 (M.)

For every fixed $\alpha > 1$ and every P-ML-random ω,

$$L_\alpha(\omega, P) < \infty.$$

In particular, $h_\alpha(\omega_{1:n}) - P(1 \mid \omega_{1:n}) \to 0$.

Proof idea for $\alpha > 1$: two steps

Fix a P-ML-random sequence.

1. Selected models have bounded complexity.
A selected model must beat P in MDL score. A more complex model needs a larger likelihood advantage. ML-randomness limits this advantage, giving a uniform complexity bound along our sequence. Hence only finitely many models can be selected.

2. Each model contributes finite total squared error.
If a fixed model Q accumulates unbounded squared error relative to P, its likelihood relative to P tends to zero (Vovk, 1987). Eventually MDL stops selecting Q, so its contribution is finite.

Conclusion: finitely many models, each contributing finite squared error, give finite total squared error. Models may keep switching.

Proof idea for $\alpha > 1$: two steps

Fix a P-ML-random sequence.

1. Selected models have bounded complexity.
A selected model must beat P in MDL score. A more complex model needs a larger likelihood advantage. ML-randomness limits this advantage, giving a uniform complexity bound along our sequence. Hence only finitely many models can be selected.

2. Each model contributes finite total squared error.
If a fixed model Q accumulates unbounded squared error relative to P, its likelihood relative to P tends to zero (Vovk, 1987). Eventually MDL stops selecting Q, so its contribution is finite.

Conclusion: finitely many models, each contributing finite squared error, give finite total squared error. Models may keep switching.

Proof idea for $\alpha > 1$: two steps

Fix a P-ML-random sequence.

1. Selected models have bounded complexity.
A selected model must beat P in MDL score. A more complex model needs a larger likelihood advantage. ML-randomness limits this advantage, giving a uniform complexity bound along our sequence. Hence only finitely many models can be selected.

2. Each model contributes finite total squared error.
If a fixed model Q accumulates unbounded squared error relative to P, its likelihood relative to P tends to zero (Vovk, 1987). Eventually MDL stops selecting Q, so its contribution is finite.

Conclusion: finitely many models, each contributing finite squared error, give finite total squared error. Models may keep switching.

Proof idea for $\alpha > 1$: two steps

Fix a P-ML-random sequence.

1. Selected models have bounded complexity.
A selected model must beat P in MDL score. A more complex model needs a larger likelihood advantage. ML-randomness limits this advantage, giving a uniform complexity bound along our sequence. Hence only finitely many models can be selected.

2. Each model contributes finite total squared error.
If a fixed model Q accumulates unbounded squared error relative to P, its likelihood relative to P tends to zero (Vovk, 1987). Eventually MDL stops selecting Q, so its contribution is finite.

Conclusion: finitely many models, each contributing finite squared error, give finite total squared error. Models may keep switching.

Proof idea for $\alpha > 1$: two steps

Fix a P-ML-random sequence.

1. Selected models have bounded complexity.
A selected model must beat P in MDL score. A more complex model needs a larger likelihood advantage. ML-randomness limits this advantage, giving a uniform complexity bound along our sequence. Hence only finitely many models can be selected.

2. Each model contributes finite total squared error.
If a fixed model Q accumulates unbounded squared error relative to P, its likelihood relative to P tends to zero (Vovk, 1987). Eventually MDL stops selecting Q, so its contribution is finite.

Conclusion: finitely many models, each contributing finite squared error, give finite total squared error. Models may keep switching.

The boundary case $\alpha = 1$ also converges

Theorem 6 (recent work)

For ordinary MDL over all computable measures and every P-ML-random ω,

$$L_1(\omega, P) \leq C\, 2^{2k+D}.$$

Moreover, only finitely many distinct measures are selected on each such path.

So $\alpha > 1$ is *not necessary* for qualitative Martin-Löf convergence.

The new proof does not first bound the complexity of every possible winner. It builds an effective test for too many eligible candidates or too much error.

The expected-error statement at $\alpha = 1$ was already covered by Poland–Hutter. The additional conclusion is about every ML-random path.

The boundary case $\alpha = 1$ also converges

Theorem 6 (recent work)

For ordinary MDL over all computable measures and every P-ML-random ω,

$$L_1(\omega, P) \leq C\, 2^{2k+D}.$$

Moreover, only finitely many distinct measures are selected on each such path.

So $\alpha > 1$ is *not necessary* for qualitative Martin-Löf convergence.

The new proof does not first bound the complexity of every possible winner. It builds an effective test for too many eligible candidates or too much error.

The expected-error statement at $\alpha = 1$ was already covered by Poland–Hutter. The additional conclusion is about every ML-random path.

The boundary case $\alpha = 1$ also converges

Theorem 6 (recent work)

For ordinary MDL over all computable measures and every P-ML-random ω,

$$L_1(\omega, P) \leq C\, 2^{2k+D}.$$

Moreover, only finitely many distinct measures are selected on each such path.

So $\alpha > 1$ is *not necessary* for qualitative Martin-Löf convergence.

The new proof does not first bound the complexity of every possible winner. It builds an effective test for too many eligible candidates or too much error.

The expected-error statement at $\alpha = 1$ was already covered by Poland–Hutter. The additional conclusion is about every ML-random path.

The boundary case $\alpha = 1$ also converges

Theorem 6 (recent work)

For ordinary MDL over all computable measures and every P-ML-random ω,

$$L_1(\omega, P) \leq C\, 2^{2k+D}.$$

Moreover, only finitely many distinct measures are selected on each such path.

So $\alpha > 1$ is *not necessary* for qualitative Martin-Löf convergence.

The new proof does not first bound the complexity of every possible winner. It builds an effective test for too many eligible candidates or too much error.

The expected-error statement at $\alpha = 1$ was already covered by Poland–Hutter. The additional conclusion is about every ML-random path.

The boundary case $\alpha = 1$ also converges

Theorem 6 (recent work)

For ordinary MDL over all computable measures and every P-ML-random ω,

$$L_1(\omega, P) \leq C\, 2^{2k+D}.$$

Moreover, only finitely many distinct measures are selected on each such path.

So $\alpha > 1$ is *not necessary* for qualitative Martin-Löf convergence.

The new proof does not first bound the complexity of every possible winner. It builds an effective test for too many eligible candidates or too much error.

The expected-error statement at $\alpha = 1$ was already covered by Poland–Hutter. The additional conclusion is about every ML-random path.

Open question

For MDL over all computable measures, the known bounds on total squared prediction error are exponential.

Is the exponential dependence tight, or can it be improved to polynomial?

In expectation, we ask for a polynomial in $K(P)$.

On each P-ML-random sequence, we ask for a polynomial in $K(P) + D(\omega \mid P)$.

Open question

For MDL over all computable measures, the known bounds on total squared prediction error are exponential.

Is the exponential dependence tight, or can it be improved to polynomial?

In expectation, we ask for a polynomial in $K(P)$.

On each P-ML-random sequence, we ask for a polynomial in $K(P) + D(\omega \mid P)$.

Open question

For MDL over all computable measures, the known bounds on total squared prediction error are exponential.

Is the exponential dependence tight, or can it be improved to polynomial?

In expectation, we ask for a polynomial in $K(P)$.

On each P-ML-random sequence, we ask for a polynomial in $K(P) + D(\omega \mid P)$.

Open question

For MDL over all computable measures, the known bounds on total squared prediction error are exponential.

Is the exponential dependence tight, or can it be improved to polynomial?

In expectation, we ask for a polynomial in $K(P)$.

On each P-ML-random sequence, we ask for a polynomial in $K(P) + D(\omega \mid P)$.

Open question

For MDL over all computable measures, the known bounds on total squared prediction error are exponential.

Is the exponential dependence tight, or can it be improved to polynomial?

In expectation, we ask for a polynomial in $K(P)$.

On each P-ML-random sequence, we ask for a polynomial in $K(P) + D(\omega \mid P)$.

A class whose hypotheses can be listed effectively

Suppose the unknown source P belongs to a family $\mathcal{A}$ of probability measures. We assume two properties:

1. **Effective enumeration**: the measures can be listed, with their rational cylinder probabilities computed exactly.
2. **Effective mixing**: from finitely many indices and rational weights, we can construct an index for their mixture.

Fix such a presentation $\mathcal{A} : e \mapsto P_e$, with $P_e(x)$ computable uniformly from e and x. Define

$$K_{\mathcal{A}}(Q) = \min_{e:\, P_e = Q} K(e).$$

MDL uses $\alpha K_{\mathcal{A}}(Q) - \log_2 Q(x)$. Now put $k = K_{\mathcal{A}}(P)$.

A class whose hypotheses can be listed effectively

Suppose the unknown source P belongs to a family $\mathcal{A}$ of probability measures. We assume two properties:

1. **Effective enumeration**: the measures can be listed, with their rational cylinder probabilities computed exactly.
2. **Effective mixing**: from finitely many indices and rational weights, we can construct an index for their mixture.

Fix such a presentation $\mathcal{A} : e \mapsto P_e$, with $P_e(x)$ computable uniformly from e and x. Define

$$K_{\mathcal{A}}(Q) = \min_{e:\, P_e = Q} K(e).$$

MDL uses $\alpha K_{\mathcal{A}}(Q) - \log_2 Q(x)$. Now put $k = K_{\mathcal{A}}(P)$.

A class whose hypotheses can be listed effectively

Suppose the unknown source P belongs to a family $\mathcal{A}$ of probability measures. We assume two properties:

1. **Effective enumeration**: the measures can be listed, with their rational cylinder probabilities computed exactly.
2. **Effective mixing**: from finitely many indices and rational weights, we can construct an index for their mixture.

Fix such a presentation $\mathcal{A} : e \mapsto P_e$, with $P_e(x)$ computable uniformly from e and x. Define

$$K_{\mathcal{A}}(Q) = \min_{e:\, P_e = Q} K(e).$$

MDL uses $\alpha K_{\mathcal{A}}(Q) - \log_2 Q(x)$. Now put $k = K_{\mathcal{A}}(P)$.

A class whose hypotheses can be listed effectively

Suppose the unknown source P belongs to a family $\mathcal{A}$ of probability measures. We assume two properties:

1. **Effective enumeration**: the measures can be listed, with their rational cylinder probabilities computed exactly.
2. **Effective mixing**: from finitely many indices and rational weights, we can construct an index for their mixture.

Fix such a presentation $\mathcal{A} : e \mapsto P_e$, with $P_e(x)$ computable uniformly from e and x. Define

$$K_{\mathcal{A}}(Q) = \min_{e:\, P_e = Q} K(e).$$

MDL uses $\alpha K_{\mathcal{A}}(Q) - \log_2 Q(x)$. Now put $k = K_{\mathcal{A}}(P)$.

A class whose hypotheses can be listed effectively

Suppose the unknown source P belongs to a family $\mathcal{A}$ of probability measures. We assume two properties:

1. **Effective enumeration**: the measures can be listed, with their rational cylinder probabilities computed exactly.
2. **Effective mixing**: from finitely many indices and rational weights, we can construct an index for their mixture.

Fix such a presentation $\mathcal{A} : e \mapsto P_e$, with $P_e(x)$ computable uniformly from e and x. Define

$$K_{\mathcal{A}}(Q) = \min_{e:\, P_e = Q} K(e).$$

MDL uses $\alpha K_{\mathcal{A}}(Q) - \log_2 Q(x)$. Now put $k = K_{\mathcal{A}}(P)$.

A class whose hypotheses can be listed effectively

Suppose the unknown source P belongs to a family $\mathcal{A}$ of probability measures. We assume two properties:

1. **Effective enumeration**: the measures can be listed, with their rational cylinder probabilities computed exactly.
2. **Effective mixing**: from finitely many indices and rational weights, we can construct an index for their mixture.

Fix such a presentation $\mathcal{A} : e \mapsto P_e$, with $P_e(x)$ computable uniformly from e and x. Define

$$K_{\mathcal{A}}(Q) = \min_{e:\, P_e = Q} K(e).$$

MDL uses $\alpha K_{\mathcal{A}}(Q) - \log_2 Q(x)$. Now put $k = K_{\mathcal{A}}(P)$.

Example: clocked programs and their finite mixtures

Fix a computable time limit $t(n)$ for histories of length n.

- Run each program within that limit to obtain a rational next-bit probability.
- Replace a late or invalid answer by $1/2$.
- Include all finite rational mixtures of the resulting measures.

This gives a uniformly computable class with effective mixing.

Example: clocked programs and their finite mixtures

Fix a computable time limit $t(n)$ for histories of length n.

- Run each program within that limit to obtain a rational next-bit probability.
- Replace a late or invalid answer by $1/2$.
- Include all finite rational mixtures of the resulting measures.

This gives a uniformly computable class with effective mixing.

Example: clocked programs and their finite mixtures

Fix a computable time limit $t(n)$ for histories of length n.

- Run each program within that limit to obtain a rational next-bit probability.
- Replace a late or invalid answer by $1/2$.
- Include all finite rational mixtures of the resulting measures.

This gives a uniformly computable class with effective mixing.

Example: clocked programs and their finite mixtures

Fix a computable time limit $t(n)$ for histories of length n.

- Run each program within that limit to obtain a rational next-bit probability.
- Replace a late or invalid answer by $1/2$.
- Include all finite rational mixtures of the resulting measures.

This gives a uniformly computable class with effective mixing.

Example: clocked programs and their finite mixtures

Fix a computable time limit $t(n)$ for histories of length n.

- Run each program within that limit to obtain a rational next-bit probability.
- Replace a late or invalid answer by $1/2$.
- Include all finite rational mixtures of the resulting measures.

This gives a uniformly computable class with effective mixing.

Example: clocked programs and their finite mixtures

Fix a computable time limit $t(n)$ for histories of length n.

- Run each program within that limit to obtain a rational next-bit probability.
- Replace a late or invalid answer by $1/2$.
- Include all finite rational mixtures of the resulting measures.

This gives a uniformly computable class with effective mixing.

For $\alpha > 1$, the bounds become quadratic

Assume the effective rational presentation and mixture closure just described.

Theorem 7 (M.)

For each fixed $\alpha > 1$, every $P \in \mathcal{A}$ and every P-ML-random ω,

$$L_\alpha(\omega, P) = O\Big((k + D)^2\Big).$$

Consequently,

$$\mathbb{E}_P L_\alpha = O(k^2).$$

For $\alpha > 1$, the bounds become quadratic

Assume the effective rational presentation and mixture closure just described.

Theorem 7 (M.)

For each fixed $\alpha > 1$, every $P \in \mathcal{A}$ and every P-ML-random ω,

$$L_\alpha(\omega, P) = O\Big((k + D)^2\Big).$$

Consequently,

$$\mathbb{E}_P L_\alpha = O(k^2).$$

For $\alpha > 1$, the bounds become quadratic

Assume the effective rational presentation and mixture closure just described.

Theorem 7 (M.)

For each fixed $\alpha > 1$, every $P \in \mathcal{A}$ and every P-ML-random ω,

$$L_\alpha(\omega, P) = O\Big((k + D)^2\Big).$$

Consequently,

$$\mathbb{E}_P L_\alpha = O(k^2).$$

For $\alpha > 1$, the bounds become quadratic

Assume the effective rational presentation and mixture closure just described.

Theorem 7 (M.)

For each fixed $\alpha > 1$, every $P \in \mathcal{A}$ and every P-ML-random ω,

$$L_\alpha(\omega, P) = O\Big((k + D)^2\Big).$$

Consequently,

$$\mathbb{E}_P L_\alpha = O(k^2).$$

Proof idea: why only linearly many models can be selected

A complexity bound alone allows exponentially many models.

1. A complete block provides a better competitor.
Group descriptions of each length into blocks of 2^r. Their uniform mixture costs at most r extra bits for the data, but saves roughly r description bits. For $\alpha > 1$ and sufficiently large fixed r, the weighted saving wins: the mixture beats components whose descriptions in the block are shortest.

2. Only constantly many winners remain at each level.
Their shortest descriptions must lie in the final incomplete block, with fewer than 2^r descriptions. There are only $O(k + D + 1)$ levels, giving a **linear bound** on the number of selected models.

Combined with the earlier error bound per model, this gives quadratic total error.

Proof idea: why only linearly many models can be selected

A complexity bound alone allows exponentially many models.

1. A complete block provides a better competitor.
Group descriptions of each length into blocks of 2^r. Their uniform mixture costs at most r extra bits for the data, but saves roughly r description bits. For $\alpha > 1$ and sufficiently large fixed r, the weighted saving wins: the mixture beats components whose descriptions in the block are shortest.

2. Only constantly many winners remain at each level.
Their shortest descriptions must lie in the final incomplete block, with fewer than 2^r descriptions. There are only $O(k + D + 1)$ levels, giving a **linear bound** on the number of selected models.

Combined with the earlier error bound per model, this gives quadratic total error.

Proof idea: why only linearly many models can be selected

A complexity bound alone allows exponentially many models.

1. A complete block provides a better competitor.
Group descriptions of each length into blocks of 2^r. Their uniform mixture costs at most r extra bits for the data, but saves roughly r description bits. For $\alpha > 1$ and sufficiently large fixed r, the weighted saving wins: the mixture beats components whose descriptions in the block are shortest.

2. Only constantly many winners remain at each level.
Their shortest descriptions must lie in the final incomplete block, with fewer than 2^r descriptions. There are only $O(k + D + 1)$ levels, giving a **linear bound** on the number of selected models.

Combined with the earlier error bound per model, this gives quadratic total error.

Proof idea: why only linearly many models can be selected

A complexity bound alone allows exponentially many models.

1. A complete block provides a better competitor.
Group descriptions of each length into blocks of 2^r. Their uniform mixture costs at most r extra bits for the data, but saves roughly r description bits. For $\alpha > 1$ and sufficiently large fixed r, the weighted saving wins: the mixture beats components whose descriptions in the block are shortest.

2. Only constantly many winners remain at each level.
Their shortest descriptions must lie in the final incomplete block, with fewer than 2^r descriptions. There are only $O(k + D + 1)$ levels, giving a **linear bound** on the number of selected models.

Combined with the earlier error bound per model, this gives quadratic total error.

Proof idea: why only linearly many models can be selected

A complexity bound alone allows exponentially many models.

1. A complete block provides a better competitor.
Group descriptions of each length into blocks of 2^r. Their uniform mixture costs at most r extra bits for the data, but saves roughly r description bits. For $\alpha > 1$ and sufficiently large fixed r, the weighted saving wins: the mixture beats components whose descriptions in the block are shortest.

2. Only constantly many winners remain at each level.
Their shortest descriptions must lie in the final incomplete block, with fewer than 2^r descriptions. There are only $O(k + D + 1)$ levels, giving a **linear bound** on the number of selected models.

Combined with the earlier error bound per model, this gives quadratic total error.

Ordinary MDL: polynomial expected error survives

Also assume we can construct, from any finite word y, an index for the measure that prints y and then zeros.

Theorem 8 (recent work, $\alpha = 1$)

For a source $P \in \mathcal{A}$ of complexity $k = K_{\mathcal{A}}(P)$,

$$\mathbb{E}_P L_1 = O\left(k^2 \log_2^3 k\right).$$

Ordinary MDL: polynomial expected error survives

Also assume we can construct, from any finite word y, an index for the measure that prints y and then zeros.

Theorem 8 (recent work, $\alpha = 1$)

For a source $P \in \mathcal{A}$ of complexity $k = K_{\mathcal{A}}(P)$,

$$\mathbb{E}_P L_1 = O\left(k^2 \log_2^3 k\right).$$

Ordinary MDL: polynomial expected error survives

Also assume we can construct, from any finite word y, an index for the measure that prints y and then zeros.

Theorem 8 (recent work, $\alpha = 1$)

For a source $P \in \mathcal{A}$ of complexity $k = K_{\mathcal{A}}(P)$,

$$\mathbb{E}_P L_1 = O\left(k^2 \log_2^3 k\right).$$

Open questions

1. Exponential or polynomial?

In the general case of all computable probability measures, can the exponential error bounds be improved to polynomial bounds?

2. $\alpha = 1$ versus $\alpha > 1$.

Is there a substantial difference in prediction quality? Can the stronger preference for simple models at $\alpha > 1$ be justified by better error bounds?

Open questions

1. Exponential or polynomial?

In the general case of all computable probability measures, can the exponential error bounds be improved to polynomial bounds?

2. $\alpha = 1$ versus $\alpha > 1$.

Is there a substantial difference in prediction quality? Can the stronger preference for simple models at $\alpha > 1$ be justified by better error bounds?

Open questions

1. Exponential or polynomial?

In the general case of all computable probability measures, can the exponential error bounds be improved to polynomial bounds?

2. $\alpha = 1$ versus $\alpha > 1$.

Is there a substantial difference in prediction quality? Can the stronger preference for simple models at $\alpha > 1$ be justified by better error bounds?