Universal learning: a view of a Bayesian

Ilya Nemenman

KITP, UCSB

nemenman@kitp.ucsb.edu

Contents

1. NFL theorems for learning, optimization, search.

2. Universal learning and continuous world.

3. Which complexity?

NFL for learning

Universal learning $\approx$ Bayesian learning with universal priors.

NFL for learning

Universal learning ≈ Bayesian learning with universal priors.

$$P_{\mathrm{est}}(\theta|\vec{x}) = \frac{P(\vec{x}|\theta)\mathcal{P}_{\mathrm{est}}(\theta)}{P_{\mathrm{est}}(\vec{x})}$$

NFL for learning

Universal learning ≈ Bayesian learning with universal priors.

$$
\begin{aligned}
P_{\text{est}}(\theta|\vec{x}) &= \frac{P(\vec{x}|\theta)\mathcal{P}_{\text{est}}(\theta)}{P_{\text{est}}(\vec{x})} \\
\mathcal{L}(\text{est}|\text{true}) &= \int d\theta P_{\text{true}}(\theta) \log \frac{P_{\text{true}}(\theta)}{P_{\text{est}}(\theta)}
\end{aligned}
$$

NFL for learning

Universal learning $\approx$ Bayesian learning with universal priors.

$$P_{\mathrm{est}}(\theta|\vec{x}) = \frac{P(\vec{x}|\theta)\mathcal{P}_{\mathrm{est}}(\theta)}{P_{\mathrm{est}}(\vec{x})}$$

$$\mathcal{L}(\mathrm{est}|\mathrm{true}) = \int d\theta P_{\mathrm{true}}(\theta) \log \frac{P_{\mathrm{true}}(\theta)}{P_{\mathrm{est}}(\theta)}$$

- NFL: if the real prior and the prior used for estimation are mismatched, one looses *on average*.

NFL for learning

Universal learning $\approx$ Bayesian learning with universal priors.

$$
\begin{aligned}
P_{\rm est}(\theta|\vec{x}) &= \frac{P(\vec{x}|\theta)\mathcal{P}_{\rm est}(\theta)}{P_{\rm est}(\vec{x})} \\
\mathcal{L}({\rm est}|{\rm true}) &= \int d\theta P_{\rm true}(\theta) \log \frac{P_{\rm true}(\theta)}{P_{\rm est}(\theta)}
\end{aligned}
$$

- NFL: if the real prior and the prior used for estimation are mismatched, one looses *on average*.
- NFL: if you are ignorant and indicate this by *uniform prior*, then the best average performance is achieved by exhaustively searching over all functions.

NFL for learning

Universal learning ≈ Bayesian learning with universal priors.

$$P_{\rm est}(\theta|\vec{x}) = \frac{P(\vec{x}|\theta)\mathcal{P}_{\rm est}(\theta)}{P_{\rm est}(\vec{x})}$$

$$\mathcal{L}({\rm est}|{\rm true}) = \int d\theta P_{\rm true}(\theta) \log \frac{P_{\rm true}(\theta)}{P_{\rm est}(\theta)}$$

- NFL: if the real prior and the prior used for estimation are mismatched, one looses *on average*.
- NFL: if you are ignorant and indicate this by *uniform prior*, then the best average performance is achieved by exhaustively searching over all functions.
- No such statements for *minimax* analysis.

NFL for learning: continuation

- Bayesian model selection and universal learning start with *well-defined, particular* priors over functions, which is an assumption.

NFL for learning: continuation

- Bayesian model selection and universal learning start with *well-defined, particular* priors over functions, which is an assumption.
- Probabilities are (roughly) uniform over sets of functions, and functions have nonuniform probabilities $1/||\mathrm{set}||$.

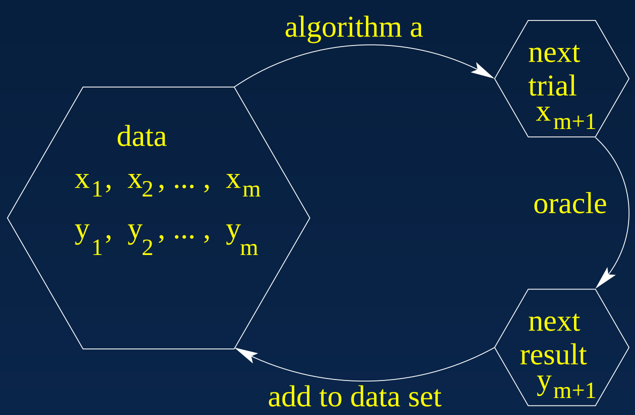
NFL for learning: continuation

- Bayesian model selection and universal learning start with *well-defined, particular* priors over functions, which is an assumption.
- Probabilities are (roughly) uniform over sets of functions, and functions have nonuniform probabilities $1/||\mathrm{set}||$.
- For Bayesian model selection and for universal learning to win, *on average*, the world must be simple (an assumption).

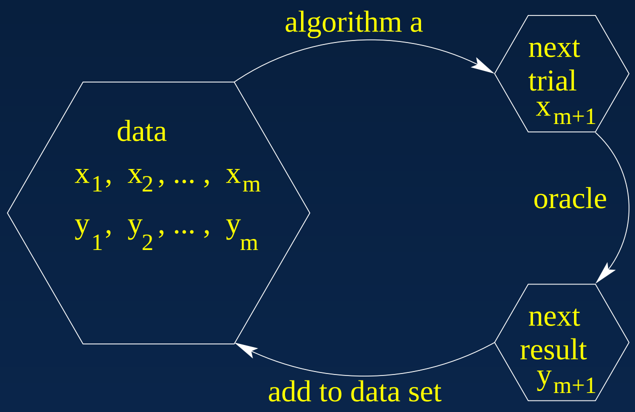
NFL for learning: continuation

- Bayesian model selection and universal learning start with *well-defined, particular* priors over functions, which is an assumption.
- Probabilities are (roughly) uniform over sets of functions, and functions have nonuniform probabilities $1/||\mathrm{set}||$.
- For Bayesian model selection and for universal learning to win, *on average*, the world must be simple (an assumption).
- But you won't loose too much if it is not, and in this case you are doomed anyway.

NFL for optimization

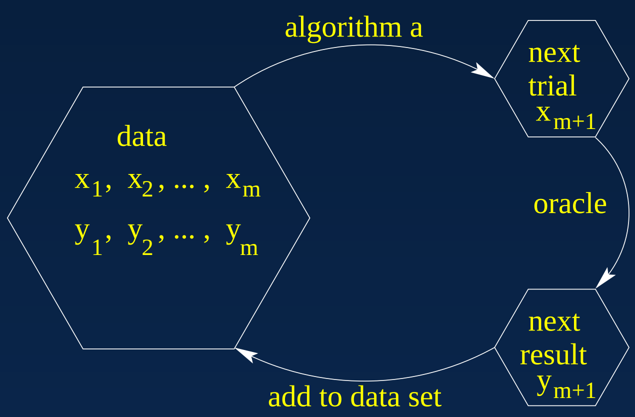


NFL for optimization



$$\sum_f P(y_m|f, a_1) = \sum_f P(y_m|f, a_2)$$

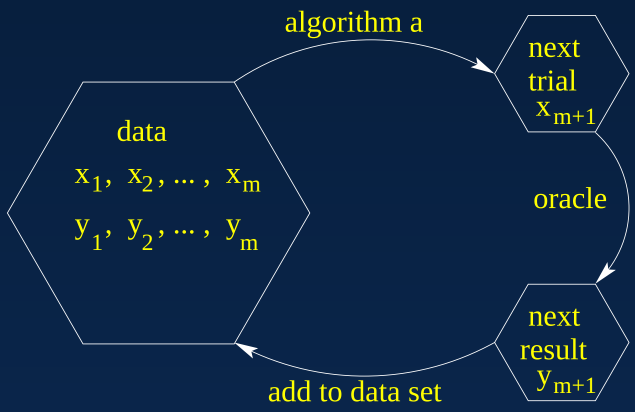
NFL for optimization



$$\sum_f P(y_m|f,a_1) \quad = \quad \sum_f P(y_m|f,a_2)$$

$$\sum_f \mathcal{L}(y_m)P(\mathcal{L}|f,a_1) \quad = \quad \sum_f \mathcal{L}(y_m)P(\mathcal{L}|f,a_2)$$

NFL for optimization

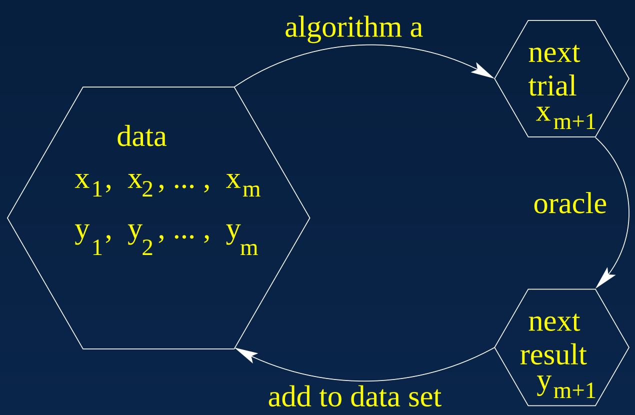


$$\sum_f P(y_m|f,a_1) = \sum_f P(y_m|f,a_2)$$

$$\sum_f \mathcal{L}(y_m)P(\mathcal{L}|f,a_1) = \sum_f \mathcal{L}(y_m)P(\mathcal{L}|f,a_2)$$

- Same holds for time–dependent optimization.

NFL for optimization

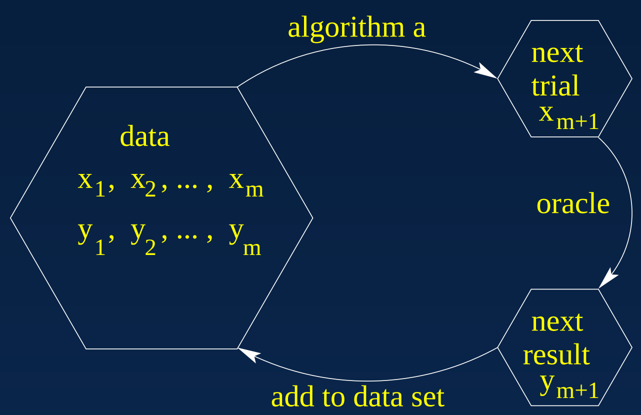


$$\sum_f P(y_m|f, a_1) = \sum_f P(y_m|f, a_2)$$

$$\sum_f \mathcal{L}(y_m) P(\mathcal{L}|f, a_1) = \sum_f \mathcal{L}(y_m) P(\mathcal{L}|f, a_2)$$

- Same holds for time–dependent optimization.
- Past performance is no guarantee of future performance.

NFL for optimization

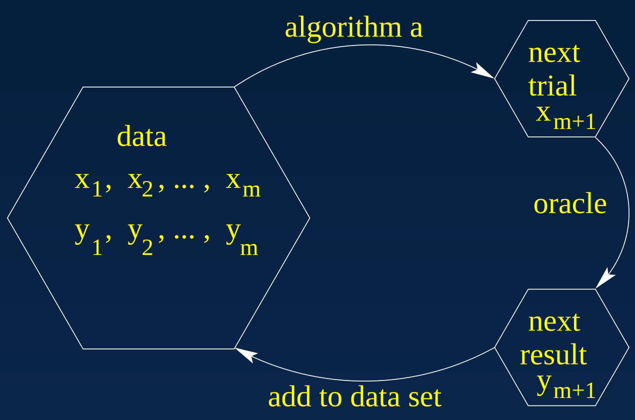


$$\sum_f P(y_m|f, a_1) = \sum_f P(y_m|f, a_2)$$

$$\sum_f \mathcal{L}(y_m) P(\mathcal{L}|f, a_1) = \sum_f \mathcal{L}(y_m) P(\mathcal{L}|f, a_2)$$

- Same holds for time–dependent optimization.
- Past performance is no guarantee of future performance.
- No similar minimax results.

NFL for optimization



$$\sum_f P(y_m|f, a_1) = \sum_f P(y_m|f, a_2)$$

$$\sum_f \mathcal{L}(y_m) P(\mathcal{L}|f, a_1) = \sum_f \mathcal{L}(y_m) P(\mathcal{L}|f, a_2)$$

- Same holds for time–dependent optimization.
- Past performance is no guarantee of future performance.
- No similar minimax results.

For uniform prior over f, what is won on some f is lost on the others.

NFL for universal search

Example: Jürgen Schmidhuber's OOPS. Spend half time searching solutions that start with the solution to a simpler problem, and half time on the rest.

NFL for universal search

Example: Jürgen Schmidhuber's OOPS. Spend half time searching solutions that start with the solution to a simpler problem, and half time on the rest.

Do not consider the interruption time, just the choice of the next guess.

NFL for universal search

Example: Jürgen Schmidhuber's OOPS. Spend half time searching solutions that start with the solution to a simpler problem, and half time on the rest.

Do not consider the interruption time, just the choice of the next guess.

Compared to	Universal prior	Uniform prior

NFL for universal search

Example: Jürgen Schmidhuber's OOPS. Spend half time searching solutions that start with the solution to a simpler problem, and half time on the rest.

Do not consider the interruption time, just the choice of the next guess.

Compared to	Universal prior	Uniform prior
Worst case loss	half the search time	all of search time

NFL for universal search

Example: Jürgen Schmidhuber's OOPS. Spend half time searching solutions that start with the solution to a simpler problem, and half time on the rest.

Do not consider the interruption time, just the choice of the next guess.

Compared to	Universal prior	Uniform prior
Worst case loss	half the search time	all of search time
Average loss	equivalent	equivalent

NFL for universal search

Example: Jürgen Schmidhuber's OOPS. Spend half time searching solutions that start with the solution to a simpler problem, and half time on the rest.

Do not consider the interruption time, just the choice of the next guess.

Compared to	Universal prior	Uniform prior
Worst case loss	half the search time	all of search time
Average loss	equivalent	equivalent

If the time till interrupt is also reweighted, things are more complicated.

NFL: discussion

- If prior is not incorporated into the algorithm, uniform prior is effectively chosen, and the results hold.

NFL: discussion

- If prior is not incorporated into the algorithm, uniform prior is effectively chosen, and the results hold.
- Uniform prior is just one of many; not necessarily the best choice to denote total ignorance.

NFL: discussion

- If prior is not incorporated into the algorithm, uniform prior is effectively chosen, and the results hold.
- Uniform prior is just one of many; not necessarily the best choice to denote total ignorance.
- But from this perspective universal prior is also one of many.

NFL: discussion

- If prior is not incorporated into the algorithm, uniform prior is effectively chosen, and the results hold.
- Uniform prior is just one of many; not necessarily the best choice to denote total ignorance.
- But from this perspective universal prior is also one of many.
- For *average* performance matching of algorithms to priors is crucial.

NFL: discussion

- If prior is not incorporated into the algorithm, uniform prior is effectively chosen, and the results hold.
- Uniform prior is just one of many; not necessarily the best choice to denote total ignorance.
- But from this perspective universal prior is also one of many.
- For *average* performance matching of algorithms to priors is crucial.
- For *minimax* properties it is not crucial.

NFL: discussion

- If prior is not incorporated into the algorithm, uniform prior is effectively chosen, and the results hold.
- Uniform prior is just one of many; not necessarily the best choice to denote total ignorance.
- But from this perspective universal prior is also one of many.
- For *average* performance matching of algorithms to priors is crucial.
- For *minimax* properties it is not crucial.
- Usually, one is interested in *average* performance for problems that are “good” and *minimax* performance on “bad” problems. NFL theorems do not say anything about universal/Occam priors in this case.

Universal learning and continuous variables

- World (presumably) is continuous.

Universal learning and continuous variables

- World (presumably) is continuous.
- One has to quantize (discretize) it before supplying to a digital computer for analysis.

Universal learning and continuous variables

- World (presumably) is continuous.
- One has to quantize (discretize) it before supplying to a digital computer for analysis.
- There are bounds on universal learner performance for each discretization.

Universal learning and continuous variables

- World (presumably) is continuous.
- One has to quantize (discretize) it before supplying to a digital computer for analysis.
- There are bounds on universal learner performance for each discretization.
- How does learning depend on a choice of coordinates and/or discretization?

Universal learning and continuous variables

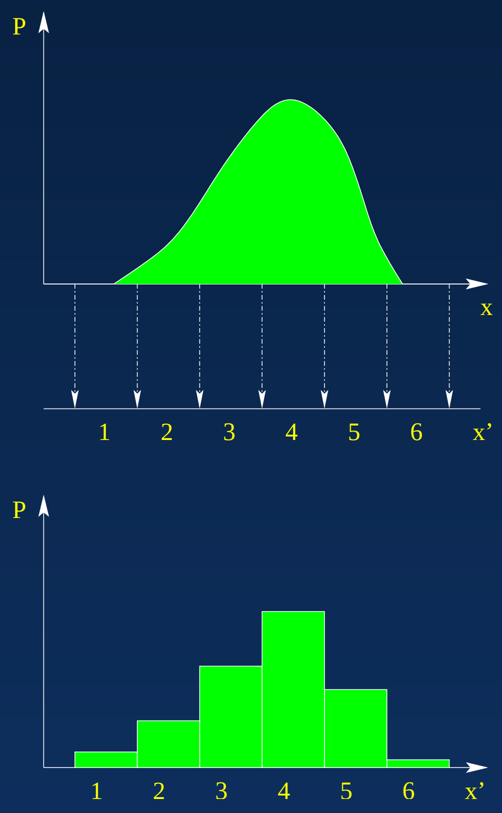
- World (presumably) is continuous.
- One has to quantize (discretize) it before supplying to a digital computer for analysis.
- There are bounds on universal learner performance for each discretization.
- How does learning depend on a choice of coordinates and/or discretization?
- Are there performance bounds uniform over all parameterizations and/or discretizations?

First problem: singular discretizations and coordinates

Uniform quantization

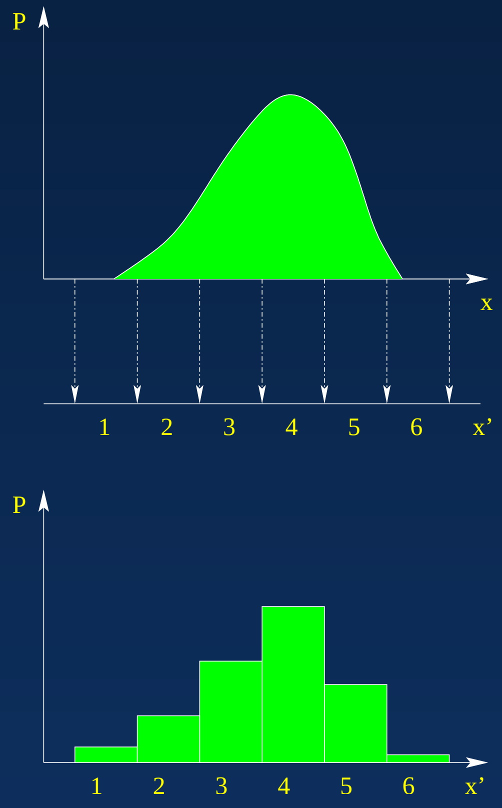
First problem: singular discretizations and coordinates

Uniform quantization



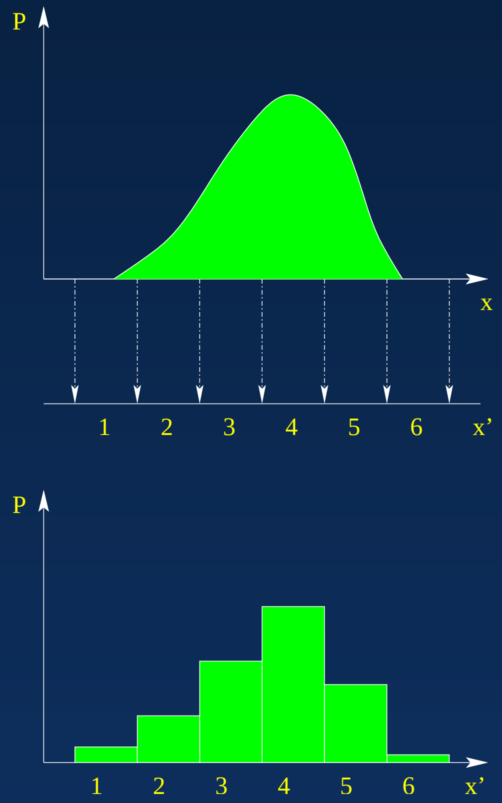
First problem: singular discretizations and coordinates

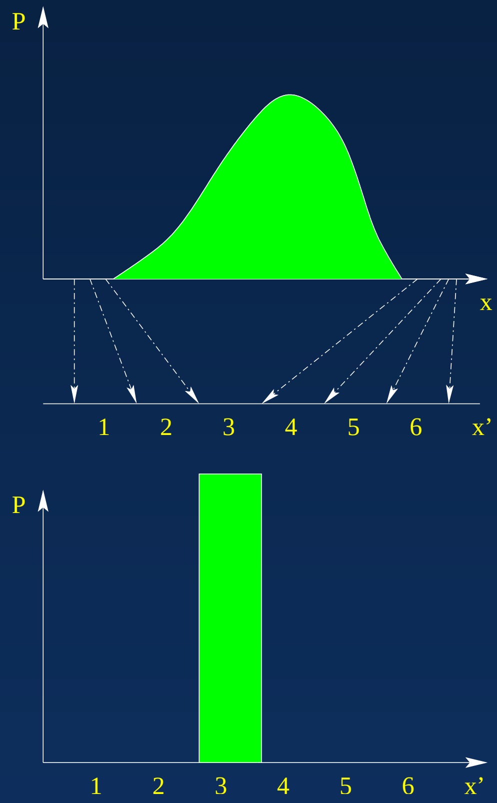
Uniform quantization Nonuniform quantization



First problem: singular discretizations and coordinates

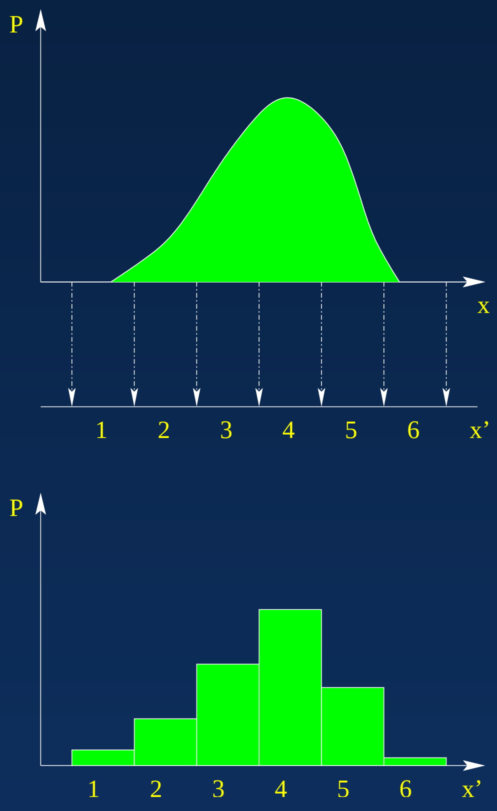
Uniform quantization   Nonuniform quantization



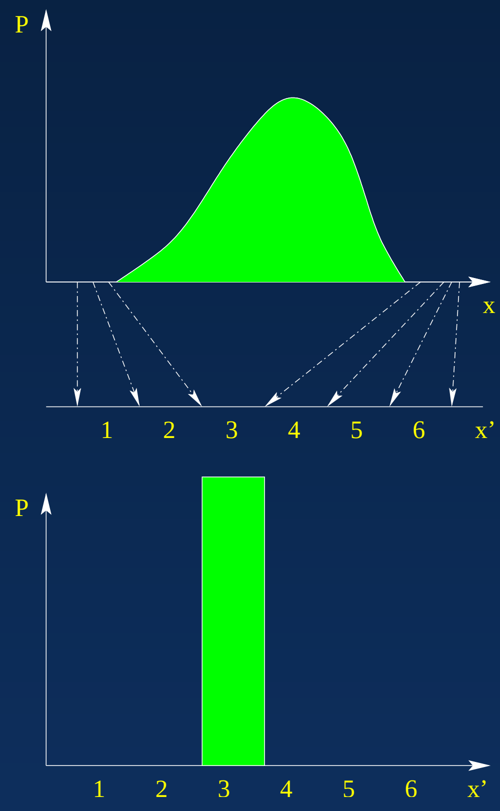


First problem: singular discretizations and coordinates

Uniform quantization



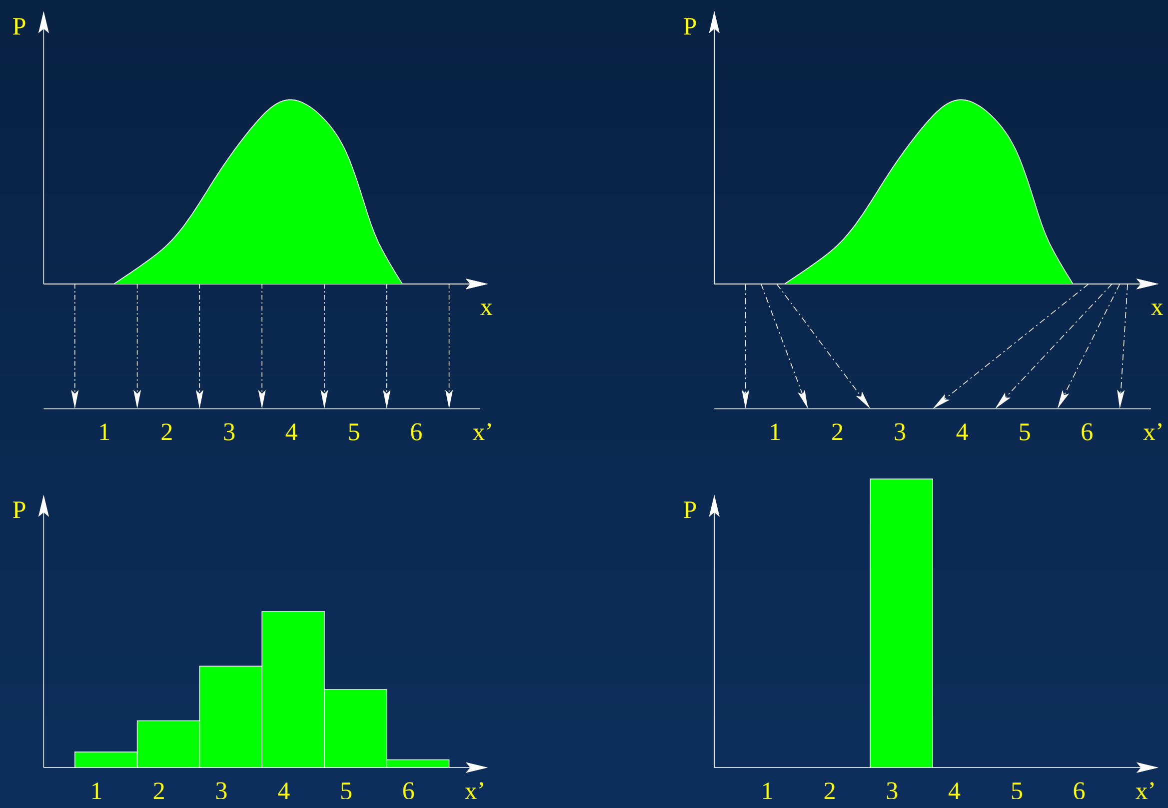
Nonuniform quantization



One will learn perfectly in the second case. But so what?

First problem: singular discretizations and coordinates

Uniform quantization   Nonuniform quantization



One will learn perfectly in the second case. But so what?

The question may be discretization dependent (e. g., finding the shortest path between two points).

Learning PDF's: covariance problem

There is a problem learning PDF's in a covariant way. Learning ($L : \{x_i\} \longrightarrow P(x)$) and reparameterization ($R_z : x \longrightarrow z(x)$) do not commute

$$[L, R_z] \neq 0$$

Learning PDF's: covariance problem

There is a problem learning PDF's in a covariant way. Learning $(L : \{x_i\} \longrightarrow P(x))$ and reparameterization $(R_z : x \longrightarrow z(x))$ do not commute

$$[L, R_z] \neq 0$$

Example: $z(x_i) = x_i$, and $z(x \neq x_i) \neq x$.

Learning PDF's: covariance problem

There is a problem learning PDF's in a covariant way. Learning $(L : \{x_i\} \longrightarrow P(x))$ and reparameterization $(R_z : x \longrightarrow z(x))$ do not commute

$$[L, R_z] \neq 0$$

Example: $z(x_i) = x_i$, and $z(x \neq x_i) \neq x$.

Similarly, quantization and learning do not commute.

Second problem: simplicity in complex coordinates

Coordinate system	x	z

Second problem: simplicity in complex coordinates

Coordinate system	x	z
PDF	P_x	$P_z = P_x \left\vert \frac{dx}{dz} \right\vert$

Second problem: simplicity in complex coordinates

Coordinate system	x	z
PDF	P_x	$P_z = P_x \left\lvert \frac{dx}{dz} \right\rvert$
K-complexity of coordinates	$K_z(x)$ huge	

Second problem: simplicity in complex coordinates

Coordinate system	x	z
PDF	P_x	$P_z = P_x \left\| \frac{dx}{dz} \right\|$
K-complexity of coordinates	$K_z(x)$ huge	$K_x(z) \asymp K_z(x)$ huge

Second problem: simplicity in complex coordinates

Coordinate system	x	z
PDF	P_x	$P_z = P_x \left\| \frac{dx}{dz} \right\|$
K-complexity of coordinates	$K_z(x)$ huge	$K_x(z) \asymp K_z(x)$ huge
K-complexity of PDF's		$K_z(P_z) \asymp 1$

Second problem: simplicity in complex coordinates

Coordinate system	x	z
PDF	P_x	$P_z = P_x \left\lvert \frac{dx}{dz} \right\rvert$
K-complexity of coordinates	$K_z(x)$ huge	$K_x(z) \asymp K_z(x)$ huge
K-complexity of PDF's	$K_x(P_x)$ huge	$K_z(P_z) \asymp 1$

Second problem: simplicity in complex coordinates

Coordinate system	x	z
PDF	P_x	$P_z = P_x \left\| \frac{dx}{dz} \right\|$
K-complexity of coordinates	$K_z(x)$ huge	$K_x(z) \asymp K_z(x)$ huge
K-complexity of PDF's	$K_x(P_x)$ huge	$K_z(P_z) \asymp 1$

To overcome analogs of NFL theorems, we must assume that P has small K-complexity in the coordinate system we have chosen.

Complexity: strings or ensembles?

Average vs. particular cases

Complexity: strings or ensembles?

Average vs. particular cases

- nothing to encode (predict, reconstruct, describe) if only one string is possible

Complexity: strings or ensembles?

Average vs. particular cases

- nothing to encode (predict, reconstruct, describe) if only one string is possible
- atypical data is possible

Complexity: strings or ensembles?

Average vs. particular cases

- nothing to encode (predict, reconstruct, describe) if only one string is possible
- atypical data is possible
- purely random string (gas in the room) is simple, and meaningless.

Complexity: strings or ensembles?

Average vs. particular cases

- nothing to encode (predict, reconstruct, describe) if only one string is possible
- atypical data is possible
- purely random string (gas in the room) is simple, and meaningless.

Need to extract *meaningful* information.

Complexity: strings or ensembles?

Average vs. particular cases

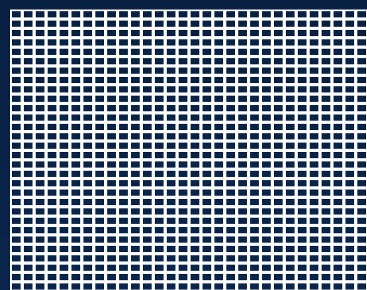
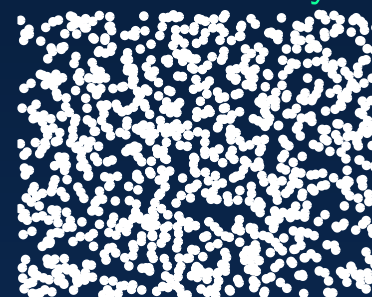
- nothing to encode (predict, reconstruct, describe) if only one string is possible
- atypical data is possible
- purely random string (gas in the room) is simple, and meaningless.

Need to extract *meaningful* information.

Individual (Martin-Löf) randomness, typicalities, etc. still require (possibly implicit) ensemble specification (or Bernoulli ensemble is assumed – but why should one work with this *worst* case?)

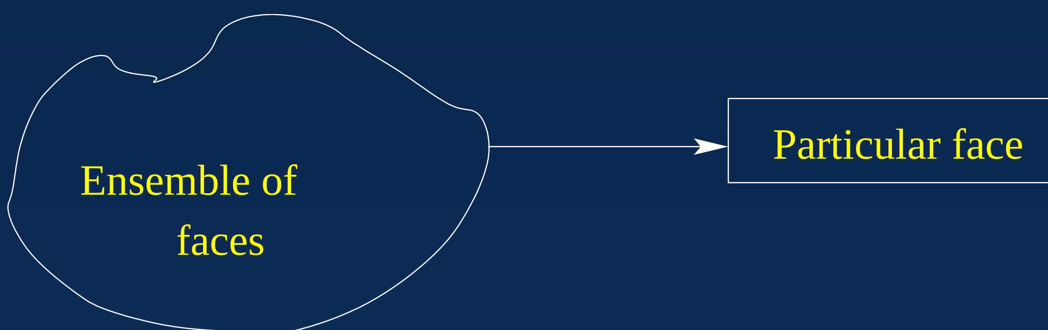
Example

Example: all pictures can be random, but we do not perceive them this way.

Example

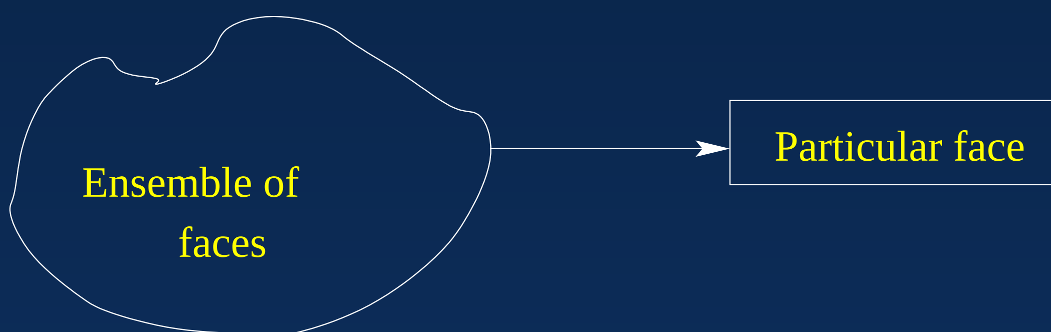
Example: all pictures can be random, but we do not perceive them this way.



For this ensemble the face is a random (typical) string.

Example

Example: all pictures can be random, but we do not perceive them this way.



For this ensemble the face is a random (typical) string.

Complexity is an ensemble (averaged) quantity, even if the ensemble is only implicit.

Quantifying averaged complexity

Shannon information theory: averaged meaningful information

Quantifying averaged complexity

Shannon information theory: averaged meaningful information

Quantifying averaged complexity

Shannon information theory: averaged meaningful information

T, N 0 T', N' x

past   now   future

$$
\begin{aligned}
\mathcal{I}_{\mathrm{pred}}(T, T') &= \left\langle \log_2 \left[\frac{P(x_{\mathrm{future}} | x_{\mathrm{past}})}{P(x_{\mathrm{future}})} \right] \right\rangle \\
&= S(T) + S(T') - S(T + T')
\end{aligned}
$$

Quantifying averaged complexity

Shannon information theory: averaged meaningful information

T, N — 0 — T', N' $\quad x$

past — now — future

$$\mathcal{I}_{\rm pred}(T, T') = \left\langle \log_2 \left[\frac{P(x_{\rm future} | x_{\rm past})}{P(x_{\rm future})} \right] \right\rangle$$

$$= S(T) + S(T') - S(T + T')$$

$$S(T) = \mathcal{S}_0 \cdot T + S_1(T)$$

Extensive component cancels in predictive information.

Quantifying averaged complexity

Shannon information theory: averaged meaningful information

$$\overset{T,N}{\text{past}} \quad \overset{0}{\text{now}} \quad \overset{T',N'}{\text{future}} \quad x$$

$$\begin{aligned} \mathcal{I}_{\text{pred}}(T,T') &= \left\langle \log_2 \left[\frac{P(x_{\text{future}}|x_{\text{past}})}{P(x_{\text{future}})} \right] \right\rangle \\ &= S(T) + S(T') - S(T+T') \\ S(T) &= \mathcal{S}_0 \cdot T + S_1(T) \end{aligned}$$

Extensive component cancels in predictive information.

Predictability is a deviation from extensivity!

Quantifying averaged complexity

Shannon information theory: averaged meaningful information

T, N | 0 | T', N' x

past | now | future

$$\mathcal{I}_{\rm pred}(T, T') = \left\langle \log_2 \left[\frac{P(x_{\rm future} | x_{\rm past})}{P(x_{\rm future})} \right] \right\rangle$$

$$= S(T) + S(T') - S(T + T')$$

$$S(T) = \mathcal{S}_0 \cdot T + S_1(T)$$

Extensive component cancels in predictive information.

Predictability is a deviation from extensivity!

$$I_{\rm pred}(T) \equiv \mathcal{I}_{\rm pred}(T, \infty) = S_1(T)$$

Complexity measure

Complexity measure

- some kind of entropy (we proclaim Shannon's postulates: monotonicity, continuity, additivity)

Complexity measure

- some kind of entropy (we proclaim Shannon's postulates: monotonicity, continuity, additivity)
- invariant under invertible temporally local transformations

($x_k \longrightarrow x_k + \xi x_{k-1}$: measuring device with inertia, article with misprints, same book in different languages – same universality class)

$$\log P_1(x) = \log P_2(x) + \text{loc. oper.} \;\Rightarrow\; C[P_1(x)] = C[P_2(x)]$$

This may present a problem in higher dimensions.

Complexity measure

- some kind of entropy (we proclaim Shannon's postulates: monotonicity, continuity, additivity)
- invariant under invertible temporally local transformations

($x_k \to x_k + \xi x_{k-1}$: measuring device with inertia, article with misprints, same book in different languages – same universality class)

$$\log P_1(x) = \log P_2(x) + \text{loc. oper.} \;\Rightarrow\; C[P_1(x)] = C[P_2(x)]$$

This may present a problem in higher dimensions.

The divergent (in T or N) part of subextensive entropy term measures complexity uniquely!

Relation to Kolmogorov complexity . . .

Relation to Kolmogorov complexity . . .

- partition all allowed strings into equivalence classes

Relation to Kolmogorov complexity ...

- partition all allowed strings into equivalence classes
- define Kolmogorov complexity $K_p(s)$ of a sequence s with respect to the partition as a length of the shortest program that can generate a sequence from the class s belongs to

Relation to Kolmogorov complexity ...

- partition all allowed strings into equivalence classes
- define Kolmogorov complexity $K_p(s)$ of a sequence s with respect to the partition as a length of the shortest program that can generate a sequence from the class s belongs to
- equivalence $=$ indistinguishable conditional distributions of futures

Relation to Kolmogorov complexity . . .

- partition all allowed strings into equivalence classes
- define Kolmogorov complexity $K_p(s)$ of a sequence s with respect to the partition as a length of the shortest program that can generate a sequence from the class s belongs to
- equivalence = indistinguishable conditional distributions of futures

If sufficient statistics exist, then $\langle K_p(s) \rangle \asymp I_{\rm pred}$. Otherwise $\langle K_p(s) \rangle \succ I_{\rm pred}$.

Relation to Kolmogorov complexity ...

- partition all allowed strings into equivalence classes
- define Kolmogorov complexity $K_p(s)$ of a sequence s with respect to the partition as a length of the shortest program that can generate a sequence from the class s belongs to
- equivalence = indistinguishable conditional distributions of futures

If sufficient statistics exist, then $\langle K_p(s) \rangle \asymp I_{\text{pred}}$. Otherwise $\langle K_p(s) \rangle \succ I_{\text{pred}}$.

$K_p(s)$ is basically the regular $K(s)$ without the random part (i. e. zero description cost for using a random number generator.)