What does $D_{KL}(\vec{p}(t) \,||\, \vec{q}(t))$ mean?

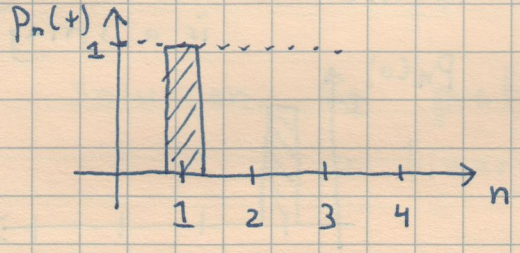
One interpretation is in terms of information.

Let us motivate a fundamental notion:

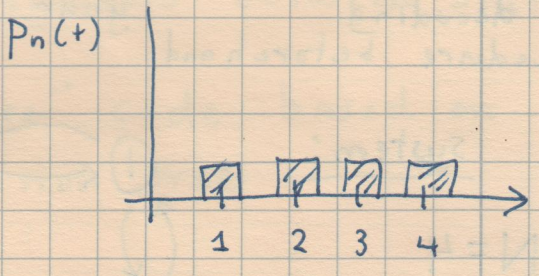
a probability $\vec{p}(t)$ associated with a system at time t contains ^potential information about the states of the system.

How much?

Naively: consider 4 state system



contains a lot of info: we know for certain the state is $n=1$



contains no info: the state can be any n with equal likelihood

Can we make this more quantitative?

Yes, thanks to the brilliance of Shannon [1948].

Imagine a scenario:

Alice (A) is a professor,
Bob (B) is her grad student.

They have an experimental protocol for a system: prepare it with init. dist. $\vec{p}(0)$, let it evolve until prob. is $\vec{p}(t)$, then take a measurement to determine the state at time t.

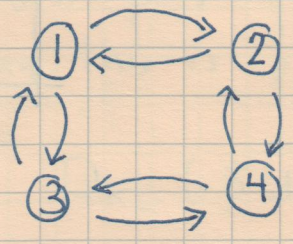
Alice is very smart, knows system, the matrix Ω, the prob. $\vec{p}(0)$ + hence can calculate $\vec{p}(t) = e^{-\Omega t}\vec{p}(0)$. She tells Bob what $\vec{p}(t)$ is, goes home, + assigns him a task:

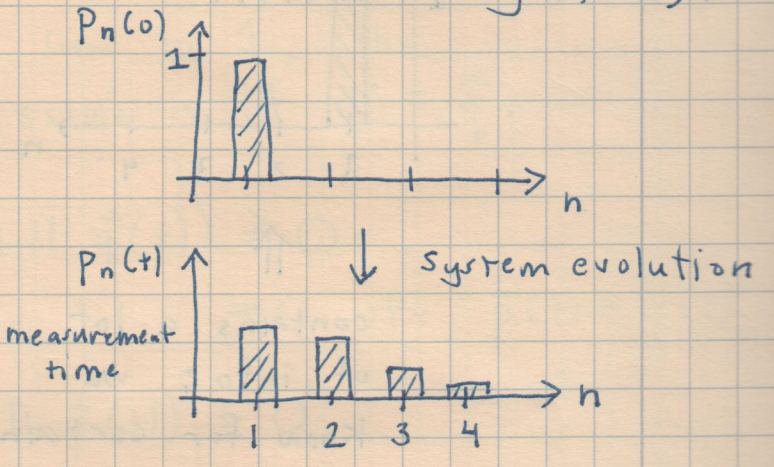
- do the protocol many times, + send me (digitally) the results of each measurement
- but be cheap about it: use as few bits as possible to communicate your results to me (assume digital message is a string of 0, 1's)

Bob can encode the info, + Alice knows the decoding procedure beforehand ⇐

system:

N = 4 states





Case I: Bob is an idiot! (or lazy) uses a fixed length code

state (measurement) result	code
1	00
2	01
3	10
4	11

mean message length per state communicated $\equiv S = \log_2 N = 2$ bits

Case II: Bob is a bit smarter! (no pun intended)

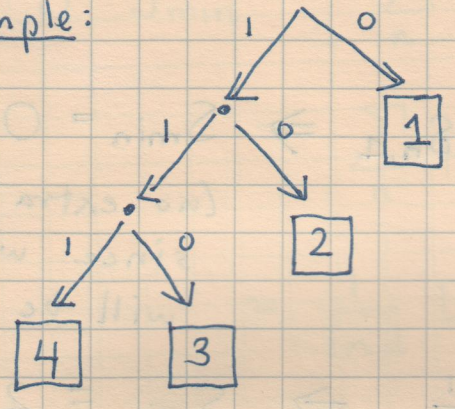
Realizes $\vec{p}(t)$ contains info about state of system, & hence can exploit it to send fewer bits to specify state.

clever idea: use fewer bits to encode more common states, economizing message length

⇒ a variable length code: but it needs a way of being decoded uniquely, without separator symbols between states

solution: a "prefix-free" code based on a binary decision tree

For example:



states are assigned to "leaves" of the tree.

Once you reach a leaf, you know the state, get ready for next transmission.

⇒

state	code
1	0
2	10
3	110
4	111

$S = 0.4 \times 1 + 0.35 \times 2 + 0.2 \times 3 + 0.05 \times 3$

$= 1.85$ bits

less than before!

In principle any binary tree w/ N leaves corresponds to some coding scheme.

Shannon source coding theorem: [1948]

among all possible codes, there is an "optimal" lower bound on avg. message length per state:

$$S \geq S_{min} = -\sum_{n=1}^{N} p_n(t) \log_2 p_n(t)$$

[measured in bits]

S_{min} tells us the smallest amount of "extra" info we need to send to specify a system state, if we know $p_n(t)$.

The smaller S_{min} is, the more informative $p_n(t)$ is, the less extra communication required.

If $p_n(t) = \delta_{n,1} \Rightarrow S_{min} = 0$ bits

(no extra required, since we know result will be 1 always)

If $p_n(t) = \frac{1}{4} \Rightarrow S_{min} = 2$ bits

(need at least 2 bits per ~~sy~~ state of extra comm.)

Proof: Consider a code scheme for N states

$\Rightarrow$ a binary tree w/ N leaves, where depth (code length) of state n is ℓ_n. In example above,

$$\ell_1 = 1, \ \ell_2 = 2, \ \ell_3 = 3, \ \ell_4 = 3$$

Define $q_n = 2^{-\ell_n} = \left(\frac{1}{2}\right)^{\ell_n}$. This is the "probability" of reaching leaf w/ state n if you start at top + choose next branch randomly w/ equal probability.

Hence $\sum_{n=1}^{N} q_n = 1$, so q_n is a true probability

(every leaf must be reached eventually)

Avg. message length per state for this code: $S = \sum_n p_n \ell_n$

$$= -\sum_n p_n \log_2 q_n$$

$$S - S_{min} = -\sum_n p_n \log_2 q_n + \sum_n p_n \log_2 p_n$$

$$= \sum_n p_n \log_2 \frac{p_n}{q_n}$$

$$= \frac{1}{\ln 2} D_{KL}(\vec{p} \,||\, \vec{q})$$

≥ 0 by properties of D_{KL}

$= 0$ iff $p_n = q_n$

$\Rightarrow \ell_n = -\log_2 p_n$ for all n (may not be possible since $-\log_2 p_n$ may not be integer)

[can be approx. true for large N]

optimal choice of code word length

$S \geq S_{min}$ always

But this also tells us how to interpret D_{KL}. Imagine Bob is sloppy & uses wrong probability $\vec{q}$ to build code instead of true one $\vec{p}$, then

$$\frac{1}{\ln 2} D_{KL}(\vec{p} \,||\, \vec{q}) = S - S_{min}$$

↑ conv. factor to bits

= extra bits needed on avg. for communicatio relative to optimal code where $\ell_n = -\log_2 p_n$

= how much less information $\vec{q}$ contains about system state, compared to $\vec{p}$

$D_{KL}(\vec{p}(t) \,||\, \vec{p}^{\,s})$ = how much more informative $\vec{p}(t)$ is about state of system at time t compared to $\vec{p}^{\,s}$

↓ decreases monotonically as $t \to \infty$