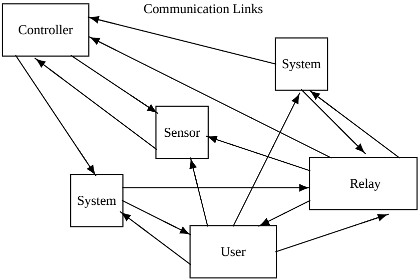
The connection between information theory and networked control

Anant Sahai
based in part on joint work with students:
Tunc Simsek, Hari Palaiyanur, and Pulkit Grover

Wireless Foundations
Department of Electrical Engineering and Computer Sciences
University of California at Berkeley

Tutorial Seminar at the
Global COE Workshop on Networked Control Systems
Kyoto University

Networked Control Systems



- Systems, sensors, and users connected with network links over noisy channels.
- Signals evolve in real time and the communication links carry ongoing and interacting streams of information.
- Holistic approach: overall cost function.

Ho, Kastner, and Wong (1978)

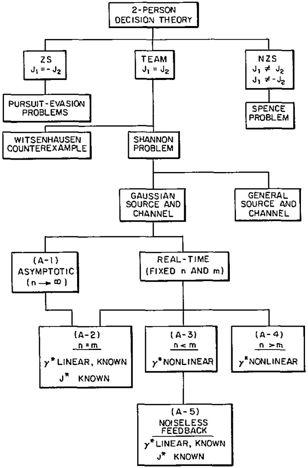


Fig. 1. Teams, signaling, and information theory.

"...sporadic and not too successful attempts have been made to relate Shannon's information theory with feedback control system design."

Shannon tells us

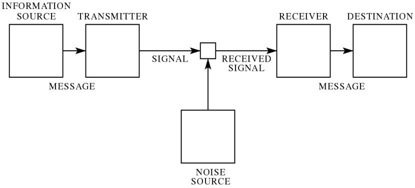


Fig. 1 — Schematic diagram of a general communication system.

- Separate source and channel coding

Shannon tells us

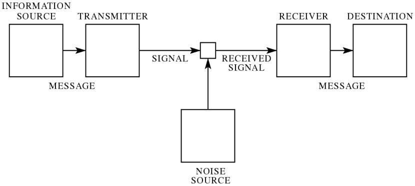


Fig. 1 — Schematic diagram of a general communication system.

- Separate source and channel coding
- But delay is the price of reliability.

Shannon tells us

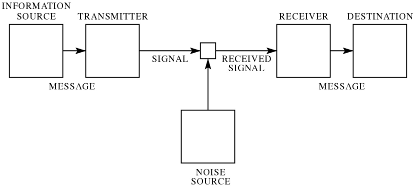


Fig. 1 — Schematic diagram of a general communication system.

- Separate source and channel coding
- But delay is the price of reliability.

"[The duality between source and channel coding] can be pursued further and is related to a duality between past and future and the notions of **control** *and knowledge. Thus we may have knowledge of the past and cannot control it; we may control the future but have no knowledge of it."* — *Claude Shannon 1959*

Shannon tells us

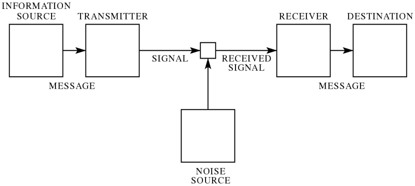


Fig. 1 — Schematic diagram of a general communication system.

- Separate source and channel coding
- But delay is the price of reliability.

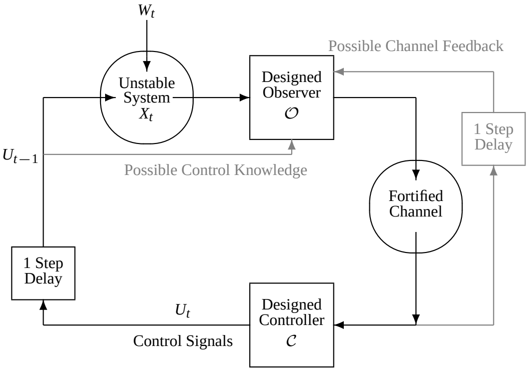
  "[The duality between source and channel coding] can be pursued further and is related to a duality between past and future and the notions of **control** *and knowledge. Thus we may have knowledge of the past and cannot control it; we may control the future but have no knowledge of it." — Claude Shannon 1959*

- What is this relationship since delays hurt control?

Outline

1. A bridge to nowhere? From control to information theory.
   - A simple control problem
   - A connection to information theory
   - Fixing information theory and filling in the gaps.
2. Coming back to the control problem
   - What is wrong with random coding
   - The role of noiseless feedback
3. Taking control thinking to the forefront of information theory.
   - The “holy grail” problem
   - Control thinking to the rescue!

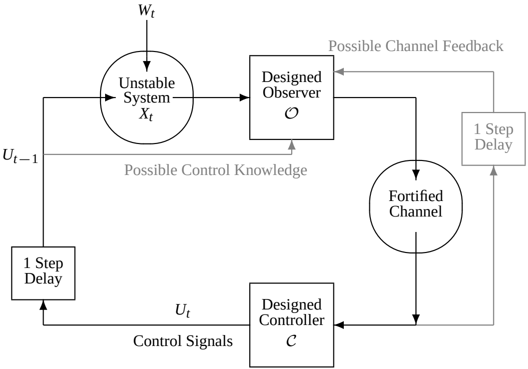
A simple distributed control problem



$$X_{t+1} = \lambda X_t + U_t + W_t$$

- Unstable $\lambda > 1$, bounded initial condition and disturbance W.

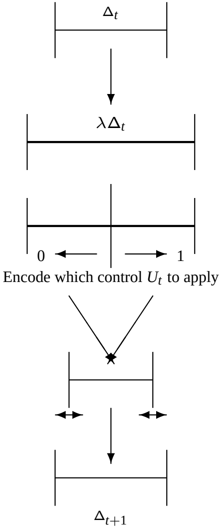
A simple distributed control problem



$$X_{t+1} = \lambda X_t + U_t + W_t$$

- Unstable $\lambda > 1$, bounded initial condition and disturbance W.
- Goal: Performance = $\sup_{t>0} E[\|X_t\|^\eta] \leq K$ for some target $K < \infty$.

Review: Entirely noiseless channel



Window known to contain X_t

will grow by factor of $\lambda > 1$

Sending R bits, cut window by a factor of 2^{-R}

grows by $\frac{\Omega}{2}$ on each side

giving a new window for X_{t+1}

As long as $R > \log_2 \lambda$, we can have Δ stay bounded forever.

The separation-principle oriented program

- Use entropy and mutual information

The separation-principle oriented program

- Use entropy and mutual information
  - Tatikonda's insight: directed mutual information captures causality

The separation-principle oriented program

- Use entropy and mutual information
  - Tatikonda's insight: directed mutual information captures causality
- Write out entropic inequalities
  - Key Inequality: Directed data-processing inequality

The separation-principle oriented program

- Use entropy and mutual information
  - Tatikonda's insight: directed mutual information captures causality
- Write out entropic inequalities
  - Key Inequality: Directed data-processing inequality
- Set up a mapping between bits and performance
  - You probably don't care about the entropy of the state.

The separation-principle oriented program

- Use entropy and mutual information
  - Tatikonda's insight: directed mutual information captures causality
- Write out entropic inequalities
  - Key Inequality: Directed data-processing inequality
- Set up a mapping between bits and performance
  - You probably don't care about the entropy of the state.
  - Lower bound performance assuming nested information
  - Equivalent to estimation

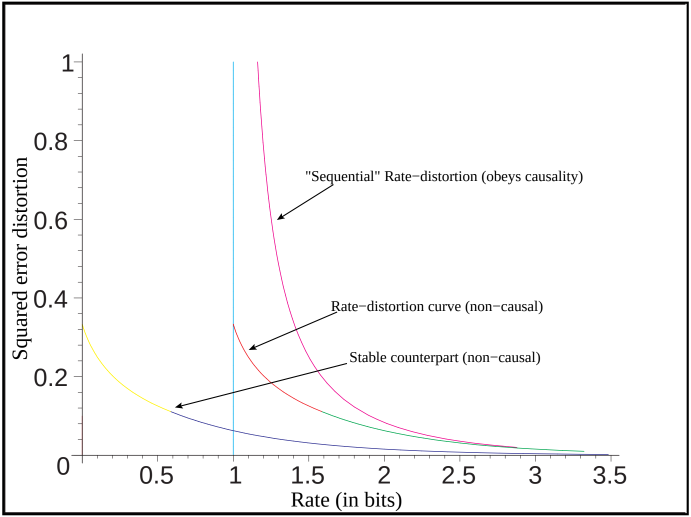
The separation-principle oriented program

- Use entropy and mutual information
  - Tatikonda's insight: directed mutual information captures causality
- Write out entropic inequalities
  - Key Inequality: Directed data-processing inequality
- Set up a mapping between bits and performance
  - You probably don't care about the entropy of the state.
  - Lower bound performance assuming nested information
  - Equivalent to estimation
  - Rate-distortion theory can be developed

The separation-principle oriented program

- Use entropy and mutual information
  - Tatikonda's insight: directed mutual information captures causality
- Write out entropic inequalities
  - Key Inequality: Directed data-processing inequality
- Set up a mapping between bits and performance
  - You probably don't care about the entropy of the state.
  - Lower bound performance assuming nested information
  - Equivalent to estimation
  - Rate-distortion theory can be developed
- Get tight upper bounds and architectures?

The rate-distortion part



How bad can entropic bounds be?

- Consider a system with
  - $\lambda = 2$ for the dynamics
  - Real packet-drop channel ($C = \infty$)

$$Z_t = \begin{cases} Y_t & \text{with Probability } \frac{1}{2} \\ 0 & \text{with Probability } \frac{1}{2} \end{cases}$$

How bad can entropic bounds be?

- Consider a system with
  - $\lambda = 2$ for the dynamics
  - Real packet-drop channel ($C = \infty$)

$$Z_t = \begin{cases} Y_t & \text{with Probability } \frac{1}{2} \\ 0 & \text{with Probability } \frac{1}{2} \end{cases}$$

- No other constraints, so design is obvious: $Y_t = X_t$ and $U_t = -\lambda Z_t$

How bad can entropic bounds be?

- Consider a system with
  - $\lambda = 2$ for the dynamics
  - Real packet-drop channel ($C = \infty$)

$$Z_t = \begin{cases} Y_t & \text{with Probability } \frac{1}{2} \\ 0 & \text{with Probability } \frac{1}{2} \end{cases}$$

- No other constraints, so design is obvious: $Y_t = X_t$ and $U_t = -\lambda Z_t$

$$X_{t+1} = \begin{cases} W_t & \text{with Probability } \frac{1}{2} \\ 2X_t + W_t & \text{with Probability } \frac{1}{2} \end{cases}$$

How bad can entropic bounds be?

- Consider a system with
  - $\lambda = 2$ for the dynamics
  - Real packet-drop channel ($C = \infty$)

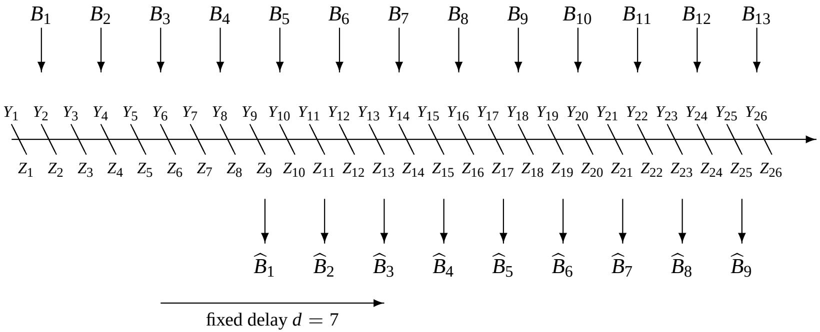
$$Z_t = \begin{cases} Y_t & \text{with Probability } \frac{1}{2} \\ 0 & \text{with Probability } \frac{1}{2} \end{cases}$$

- No other constraints, so design is obvious: $Y_t = X_t$ and $U_t = -\lambda Z_t$

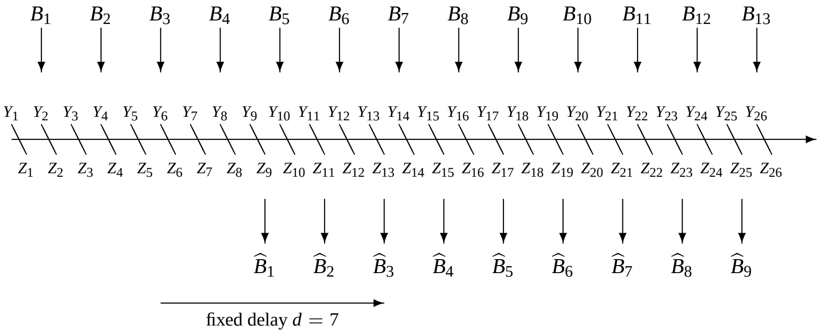
$$X_{t+1} = \begin{cases} W_t & \text{with Probability } \frac{1}{2} \\ 2X_t + W_t & \text{with Probability } \frac{1}{2} \end{cases}$$

- Under stochastic disturbances, the variance of the state is asymptotically infinite. (*St. Petersburg Lottery Style*)

Delay-universal (*anytime*) communication

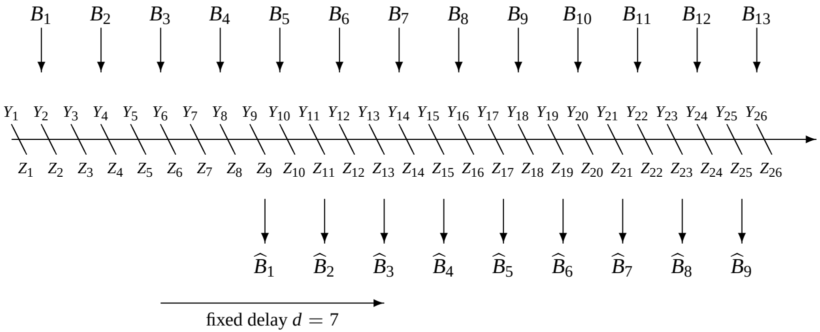


Delay-universal (*anytime*) communication



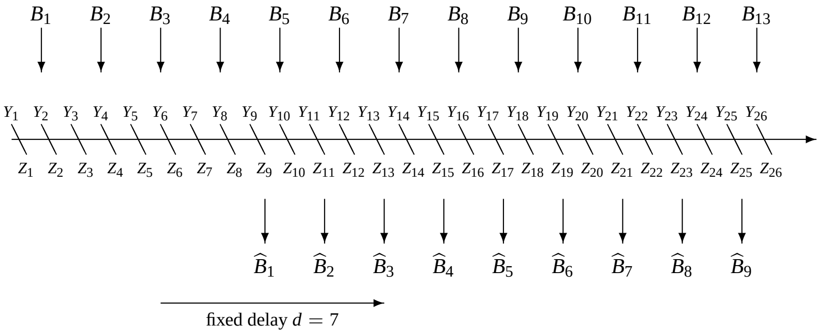
- Fixed-delay reliability α is achievable if there exists a sequence of encoder/decoder pairs with increasing end-to-end delays d_j such that $\lim_{j \to \infty} \frac{-1}{d_j} \ln P(B_i \neq \hat{B}_i^j) = \alpha$.

Delay-universal (*anytime*) communication



- α is achievable *delay-universally* or in an *anytime fashion* if a single encoder works for all sufficiently large delays d.

Delay-universal (*anytime*) communication



- The anytime capacity $C_{\text{any}}(\alpha)$ is the supremal rate at which reliability α is achievable in a delay-universal way.

Separation theorem for scalar control

Necessity: If a scalar system with parameter $\lambda > 1$ can be stabilized with finite η-moment across a noisy channel, then the **channel with noiseless feedback** must have

$$C_{\text{any}}(\eta \ln \lambda) \geq \ln \lambda$$

In general: If $P(|X| > m) < f(m)$, then $\exists K : P_{\text{error}}(d) < f(K\lambda^d)$

Separation theorem for scalar control

Necessity: If a scalar system with parameter $\lambda > 1$ can be stabilized with finite η-moment across a noisy channel, then the **channel with noiseless feedback** must have

$$C_{\text{any}}(\eta \ln \lambda) \geq \ln \lambda$$

In general: If $P(|X| > m) < f(m)$, then $\exists K : P_{\text{error}}(d) < f(K\lambda^d)$

Sufficiency: If there is an $\alpha > \eta \ln \lambda$ for which the **channel with noiseless feedback** has

$$C_{\text{any}}(\alpha) > \ln \lambda$$

then the scalar system with parameter $\lambda \geq 1$ with a bounded disturbance can be stabilized across the noisy channel with finite η-moment.

Separation theorem for scalar control

Necessity: If a scalar system with parameter $\lambda > 1$ can be stabilized with finite η-moment across a noisy channel, then the **channel with noiseless feedback** must have

$$C_{\text{any}}(\eta \ln \lambda) \geq \ln \lambda$$

In general: If $P(|X| > m) < f(m)$, then $\exists K : P_{\text{error}}(d) < f(K\lambda^d)$

Sufficiency: If there is an $\alpha > \eta \ln \lambda$ for which the **channel with noiseless feedback** has

$$C_{\text{any}}(\alpha) > \ln \lambda$$

then the scalar system with parameter $\lambda \geq 1$ with a bounded disturbance can be stabilized across the noisy channel with finite η-moment.

Captures stabilization only.

Some easy implications

- If we want $P(|X_t| > m) \leq f(m) = 0$ for some finite m, we require zero-error reliability across the channel. Also required (for DMCs) if we want the controller to be finite memory.

Some easy implications

- If we want $P(|X_t| > m) \leq f(m) = 0$ for some finite m, we require zero-error reliability across the channel. Also required (for DMCs) if we want the controller to be finite memory.
- For generic DMCs, anytime reliability with feedback is upper-bounded:

$$
\begin{aligned}
f(K\lambda^d) &\geq \zeta^d \\
f(m) &\geq K' m^{-\frac{\log_2 \frac{1}{\zeta}}{\log_2 \lambda}}
\end{aligned}
$$

A controlled state can have at best a power-law tail.

Some easy implications

- If we want $P(|X_t| > m) \leq f(m) = 0$ for some finite m, we require zero-error reliability across the channel. Also required (for DMCs) if we want the controller to be finite memory.
- For generic DMCs, anytime reliability with feedback is upper-bounded:

$$\begin{aligned} f(K\lambda^d) &\geq \zeta^d \\ f(m) &\geq K' m^{-\frac{\log_2 \frac{1}{\zeta}}{\log_2 \lambda}} \end{aligned}$$

  A controlled state can have at best a power-law tail.

- If we just want $\lim_{m \to \infty} f(m) = 0$, then just Shannon capacity $> \log_2 \lambda$ is required for DMCs.

Some easy implications

- If we want $P(|X_t| > m) \leq f(m) = 0$ for some finite m, we require zero-error reliability across the channel. Also required (for DMCs) if we want the controller to be finite memory.
- For generic DMCs, anytime reliability with feedback is upper-bounded:

$$
\begin{aligned}
f(K\lambda^d) &\geq \zeta^d \\
f(m) &\geq K' m^{-\frac{\log_2 \frac{1}{\zeta}}{\log_2 \lambda}}
\end{aligned}
$$

A controlled state can have at best a power-law tail.

- If we just want $\lim_{m\to\infty} f(m) = 0$, then just Shannon capacity $> \log_2 \lambda$ is required for DMCs.
- Almost-sure stabilization for $W_t = 0$ follows by simple time-varying transformation.

Asymptotic communication problem hierarchy

- The easiest: Shannon communication
  - Asymptotically: a single figure of merit C
  - Equivalent to most estimation problems of stationary ergodic processes with bounded distortion measures.
  - Feedback does not matter.

Asymptotic communication problem hierarchy

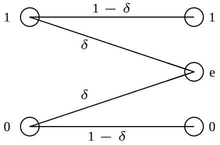
- The easiest: Shannon communication
  - Asymptotically: a single figure of merit C
  - Equivalent to most estimation problems of stationary ergodic processes with bounded distortion measures.
  - Feedback does not matter.

- Hardest level: Zero-error communication
  - Single figure of merit C_0
  - Feedback matters.

Asymptotic communication problem hierarchy

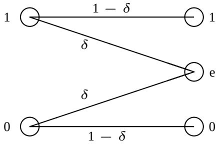
- The easiest: Shannon communication
  - Asymptotically: a single figure of merit C
  - Equivalent to most estimation problems of stationary ergodic processes with bounded distortion measures.
  - Feedback does not matter.
- Intermediate families: Anytime communication
  - Multiple figures of merit: $(\vec{R}, \vec{\alpha})$
  - Feedback case equivalent to stabilization problems
  - Related nonstationary estimation problems fall here also
  - Does feedback matter?
- Hardest level: Zero-error communication
  - Single figure of merit C_0
  - Feedback matters.

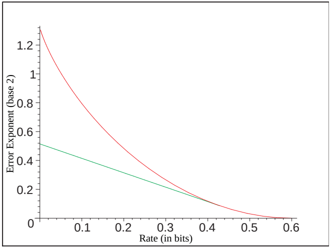
My favorite example: the BEC



- Simple capacity $1 - \delta$ bits per channel use
- With perfect feedback, simple to achieve: retransmit until it gets through
  - Time till success: Geometric$(1 - \delta)$
  - Expected time to get through: $\frac{1}{1-\delta}$

My favorite example: the BEC

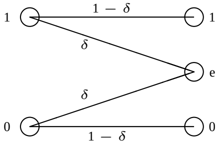


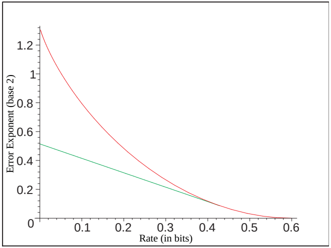


- Simple capacity $1 - \delta$ bits per channel use
- With perfect feedback, simple to achieve: retransmit until it gets through
  - Time till success: Geometric$(1 - \delta)$
  - Expected time to get through: $\frac{1}{1-\delta}$

- Classical bounds
  - Sphere-packing bound $D(1 - R||\delta)$
  - Random coding bound $\max_{\rho \in [0,1]} E_0(\rho) - \rho R$

My favorite example: the BEC





- Simple capacity $1 - \delta$ bits per channel use
- With perfect feedback, simple to achieve: retransmit until it gets through
  - Time till success: Geometric$(1 - \delta)$
  - Expected time to get through: $\frac{1}{1-\delta}$
- Classical bounds
  - Sphere-packing bound $D(1 - R||\delta)$
  - Random coding bound $\max_{\rho \in [0,1]} E_0(\rho) - \rho R$
- What happens with feedback?

BEC with feedback and fixed *blocks*

- At rate $R < 1$, have Rn bits to transmit in n channel uses.
- Typically $(1-\delta)n$ code bits will be received.

BEC with feedback and fixed *blocks*

- At rate $R < 1$, have Rn bits to transmit in n channel uses.
- Typically $(1 - \delta)n$ code bits will be received.
- Block errors caused by atypical channel behavior.
  - Doomed if fewer than Rn bits arrive intact.

BEC with feedback and fixed *blocks*

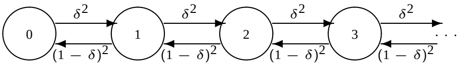
- At rate $R < 1$, have Rn bits to transmit in n channel uses.
- Typically $(1-\delta)n$ code bits will be received.
- Block errors caused by atypical channel behavior.
  - Doomed if fewer than Rn bits arrive intact.
  - *Feedback can not save us.*
  - $D(1-R||\delta)$

BEC with feedback and fixed *blocks*

- At rate $R < 1$, have Rn bits to transmit in n channel uses.
- Typically $(1-\delta)n$ code bits will be received.
- Block errors caused by atypical channel behavior.
  - Doomed if fewer than Rn bits arrive intact.
  - *Feedback can not save us.*
  - $D(1-R||\delta)$
- Dobrushin-62 showed that this type of behavior is common: $E^+(R) = E_{sp}(R)$ for symmetric channels.

BEC with feedback and fixed *delay*

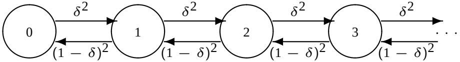
- $R = \frac{1}{2}$ example:



- Birth-death chain: positive recurrent if $\delta < \frac{1}{2}$

BEC with feedback and fixed *delay*

- $R = \frac{1}{2}$ example:

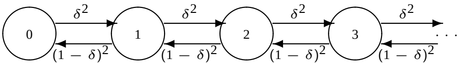


- Birth-death chain: positive recurrent if $\delta < \frac{1}{2}$
- Delay exponent easy to see:

$$P(D \geq d) = P(L > \frac{d}{2}) = K(\frac{\delta}{1-\delta})^d$$

BEC with feedback and fixed *delay*

- $R = \frac{1}{2}$ example:



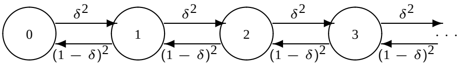
- Birth-death chain: positive recurrent if $\delta < \frac{1}{2}$
- Delay exponent easy to see:

$$P(D \geq d) = P(L > \frac{d}{2}) = K(\frac{\delta}{1-\delta})^d$$

- ≈ 0.584 vs 0.0294 for block-coding with $\delta = 0.4$

BEC with feedback and fixed *delay*

- $R = \frac{1}{2}$ example:



- Birth-death chain: positive recurrent if $\delta < \frac{1}{2}$
- Delay exponent easy to see:

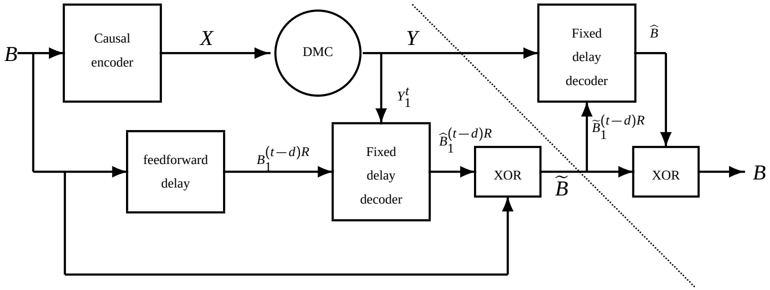
$$P(D \geq d) = P(L > \frac{d}{2}) = K(\frac{\delta}{1-\delta})^d$$

- ≈ 0.584 vs 0.0294 for block-coding with $\delta = 0.4$

Block-coding is misleading!

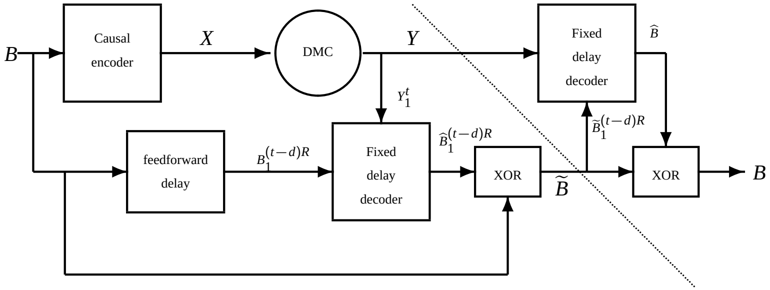
Pinsker's bounding construction explained

- Without feedback: $E^+(R)$ continues to be a bound.
- Consider a code with target delay d
  - Use it to construct a block-code with blocksize $n \gg d$
  - Genie-aided decoder: has the truth of all bits before i



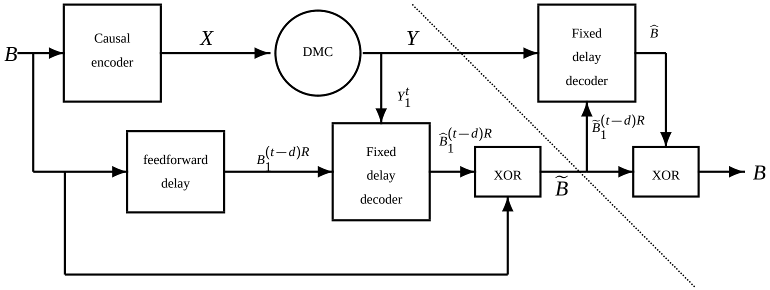
Pinsker's bounding construction explained

- Without feedback: $E^+(R)$ continues to be a bound.
- Consider a code with target delay d
  - Use it to construct a block-code with blocksize $n \gg d$
  - Genie-aided decoder: has the truth of all bits before i
  - Error events for genie-aided system depend only on last d

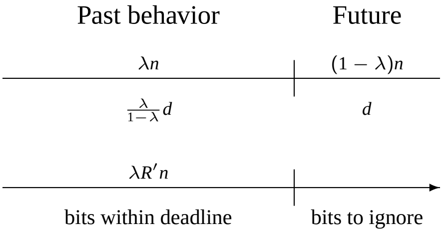


Pinsker's bounding construction explained

- Without feedback: $E^+(R)$ continues to be a bound.
- Consider a code with target delay d
  - Use it to construct a block-code with blocksize $n \gg d$
  - Genie-aided decoder: has the truth of all bits before i
  - Error events for genie-aided system depend only on last d
  - Apply a change of measure argument

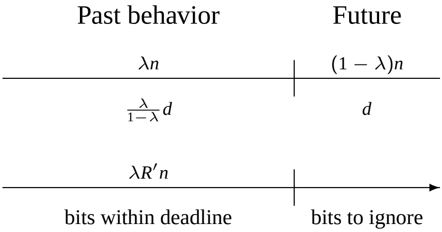


Using E_{sp} to bound α^* in general



- The block error probability is like $\exp(-\alpha(1-\lambda)n)$ which cannot exceed the Haroutunian bound $\exp(-E_{sp}(\lambda R)n)$

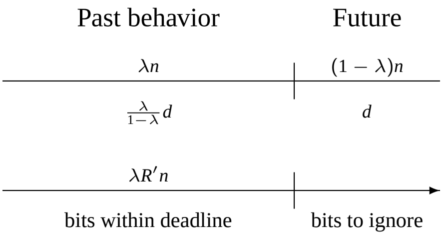
Using E_{sp} to bound α^* in general



- The block error probability is like $\exp(-\alpha(1-\lambda)n)$ which cannot exceed the Haroutunian bound $\exp(-E_{sp}(\lambda R)n)$

$$\alpha^*(R) \leq \frac{E_{sp}(\lambda R)}{1-\lambda}$$

Using E_{sp} to bound α^* in general



- The block error probability is like $\exp(-\alpha(1-\lambda)n)$ which cannot exceed the Haroutunian bound $\exp(-E_{sp}(\lambda R)n)$

$$\alpha^*(R) \leq \frac{E_{sp}(\lambda R)}{1-\lambda}$$

- The error events involve *both* the past and the future.

Uncertainty-focusing bound for symmetric DMCs

Minimize over λ for symmetric DMCs to sweep out frontier by varying $\rho > 0$:

$$\begin{aligned} R(\rho) &= \frac{E_0(\rho)}{\rho} \\ E_a^+(\rho) &= E_0(\rho) \end{aligned}$$

Using the Gallager function:

$$E_0(\rho) = -\max_q \ln \sum_j \left(\sum_i q_i p_{ij}^{\frac{1}{1+\rho}} \right)^{1+\rho}$$

Uncertainty-focusing bound for symmetric DMCs

Minimize over λ for symmetric DMCs to sweep out frontier by varying $\rho > 0$:

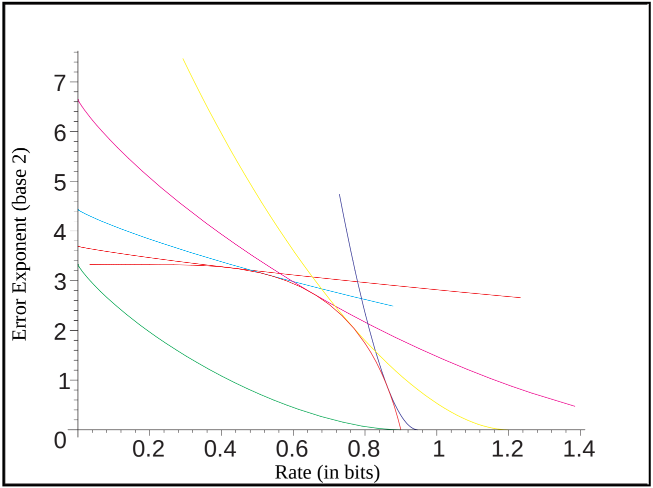
$$
\begin{aligned}
R(\rho) &= \frac{E_0(\rho)}{\rho} \\
E_a^+(\rho) &= E_0(\rho)
\end{aligned}
$$

Using the Gallager function:

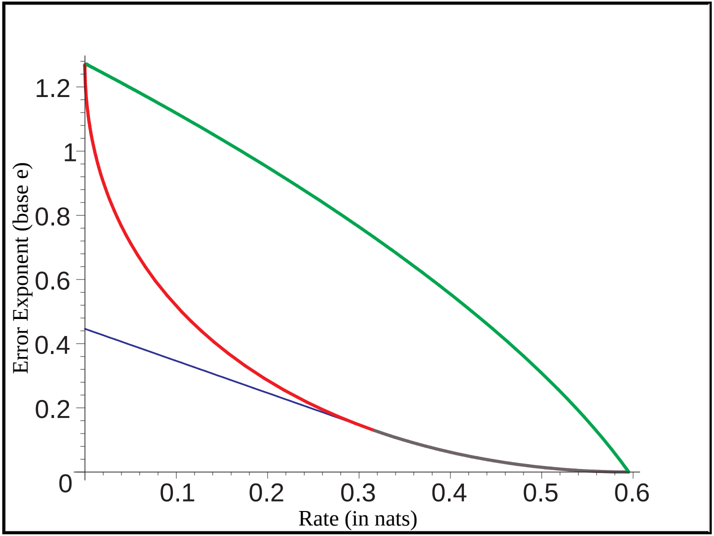
$$E_0(\rho) = -\max_q \ln \sum_j \left(\sum_i q_i p_{ij}^{\frac{1}{1+\rho}} \right)^{1+\rho}$$

Same form as Viterbi's "convolutional coding bound" for constraint-lengths, but a lot more fundamental!

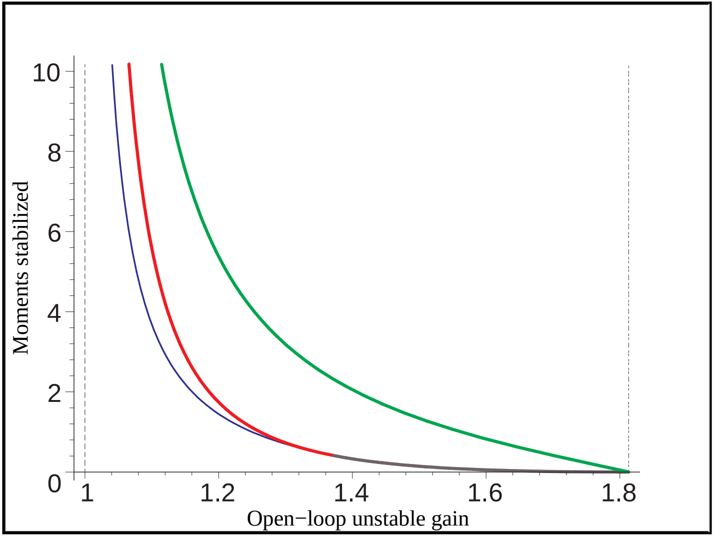
Upper bound tight for the BEC with feedback



Implications for scalar moment stabilization



Implications for scalar moment stabilization

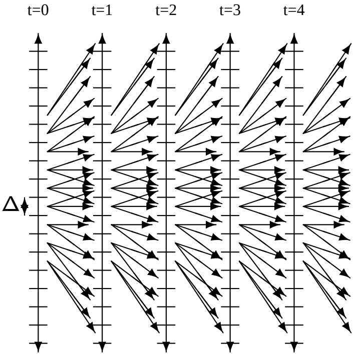


Outline

1. A bridge to nowhere?
   - A simple control problem
   - A connection to information theory
   - Fixing information theory and filling in the gaps.
2. Coming back to control
   - What is wrong with random coding
   - The role of noiseless feedback
3. Taking control thinking to the forefront of information theory.
   - The “holy grail” problem
   - Control thinking to the rescue!

Random coding bound is relatively easy to achieve

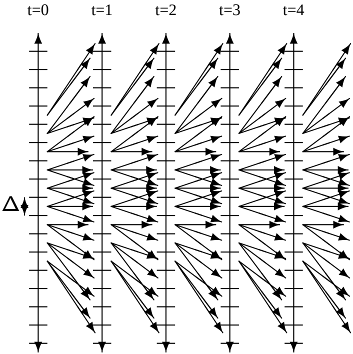
- Randomly label the uniformly quantized state!



$R = \log_2 3$

Random coding bound is relatively easy to achieve

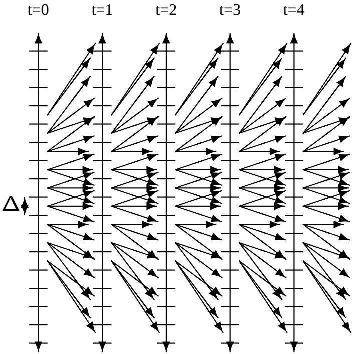
- Randomly label the uniformly quantized state!
- Stable system state "renews" itself.



$R = \log_2 3$

Random coding bound is relatively easy to achieve

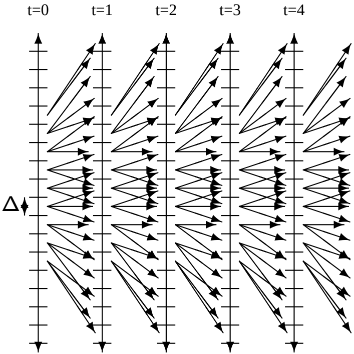
- Randomly label the uniformly quantized state!
- Stable system state "renews" itself.
- It diverges locally whenever the channel misbehaves.



$R = \log_2 3$

Random coding bound is relatively easy to achieve

- Randomly label the uniformly quantized state!
- Stable system state "renews" itself.
- It diverges locally whenever the channel misbehaves.
- Semi-reasonable implementation complexity.



$R = \log_2 3$

Controller and Computations

- All "false" disjoint paths through the trellis are pairwise independent with the true path.

Controller and Computations

- All “false” disjoint paths through the trellis are pairwise independent with the true path.
- Bound the number of distinct paths by assuming no remerging.

Controller and Computations

- All “false” disjoint paths through the trellis are pairwise independent with the true path.
- Bound the number of distinct paths by assuming no remerging.
- Gallager’s $E_r(R_{branch})$ emerges as the governing exponent.

Controller and Computations

- All “false” disjoint paths through the trellis are pairwise independent with the true path.
- Bound the number of distinct paths by assuming no remerging.
- Gallager’s $E_r(R_{branch})$ emerges as the governing exponent.
- Apply the control based on current ML state.

Controller and Computations

- All "false" disjoint paths through the trellis are pairwise independent with the true path.
- Bound the number of distinct paths by assuming no remerging.
- Gallager's $E_r(R_{branch})$ emerges as the governing exponent.
- Apply the control based on current ML state.
- Computational nightmare: effort grows exponentially with time.

Controller and Computations

- All "false" disjoint paths through the trellis are pairwise independent with the true path.
- Bound the number of distinct paths by assuming no remerging.
- Gallager's $E_r(R_{branch})$ emerges as the governing exponent.
- Apply the control based on current ML state.
- Computational nightmare: effort grows exponentially with time.
- Use "Stack-based" greedy search algorithm instead.
  - Log likelihoods are additive.
  - The score of a path is a random walk with drift.
  - Bias it so that the true path goes up and false ones down.

Controller and Computations

- All "false" disjoint paths through the trellis are pairwise independent with the true path.
- Bound the number of distinct paths by assuming no remerging.
- Gallager's $E_r(R_{branch})$ emerges as the governing exponent.
- Apply the control based on current ML state.
- Computational nightmare: effort grows exponentially with time.
- Use "Stack-based" greedy search algorithm instead.
  - Log likelihoods are additive.
  - The score of a path is a random walk with drift.
  - Bias it so that the true path goes up and false ones down.
- Classical results tell us that with appropriate bias, achieve $E_r(R_{branch})$ for error probability and hence power-law in state

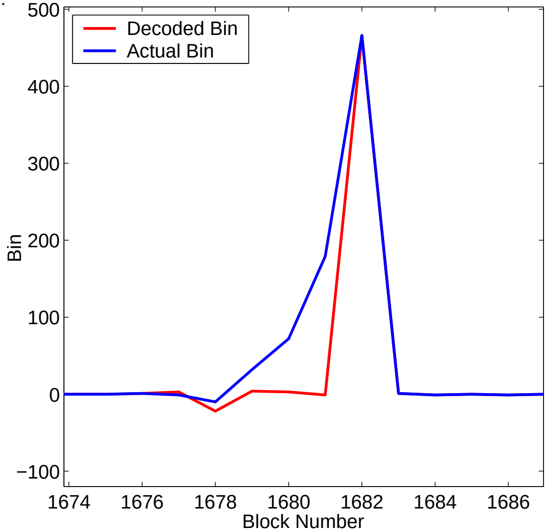
Controller and Computations

- All "false" disjoint paths through the trellis are pairwise independent with the true path.
- Bound the number of distinct paths by assuming no remerging.
- Gallager's $E_r(R_{branch})$ emerges as the governing exponent.
- Apply the control based on current ML state.
- Computational nightmare: effort grows exponentially with time.
- Use "Stack-based" greedy search algorithm instead.
  - Log likelihoods are additive.
  - The score of a path is a random walk with drift.
  - Bias it so that the true path goes up and false ones down.
- Classical results tell us that with appropriate bias, achieve $E_r(R_{branch})$ for error probability and hence power-law in state
- At the cost of only finite expected computation.

Catch up "all-at-once" phenomenon

Simulation Parameters:
λ = 1.1
ε = 0.05
Ω = 2.0
Δ = 5000.0
Bias = 0.55
T = 10
100,000 Blocks
17 seconds to run

Rate = 0.317
Capacity = 0.71



Truth in advertising: computation revisited

- Although we are doing better than exponential growth, we still have power laws on both sides.
- What if we needed a finite speed computer in the controller?

Truth in advertising: computation revisited

- Although we are doing better than exponential growth, we still have power laws on both sides.
- What if we needed a finite speed computer in the controller?
- Bad news:
  - Assume 0 control applied if we can not decode yet.

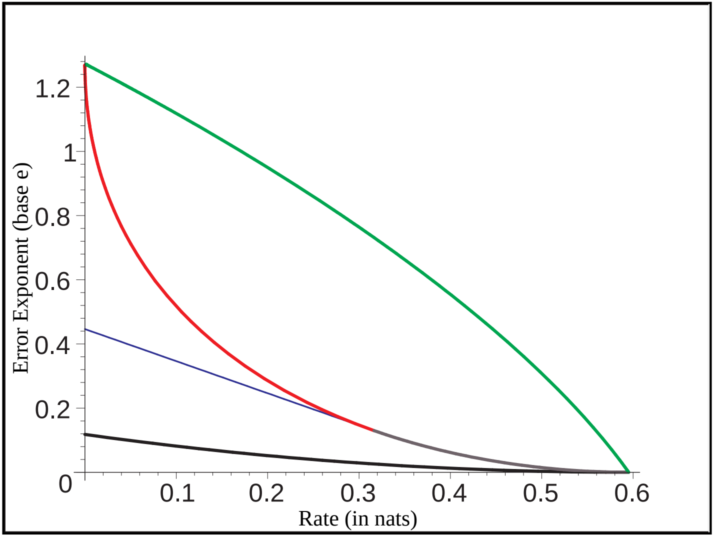
Truth in advertising: computation revisited

- Although we are doing better than exponential growth, we still have power laws on both sides.
- What if we needed a finite speed computer in the controller?
- Bad news:
  - Assume 0 control applied if we can not decode yet.
  - Power law for comp. implies power low for waiting.

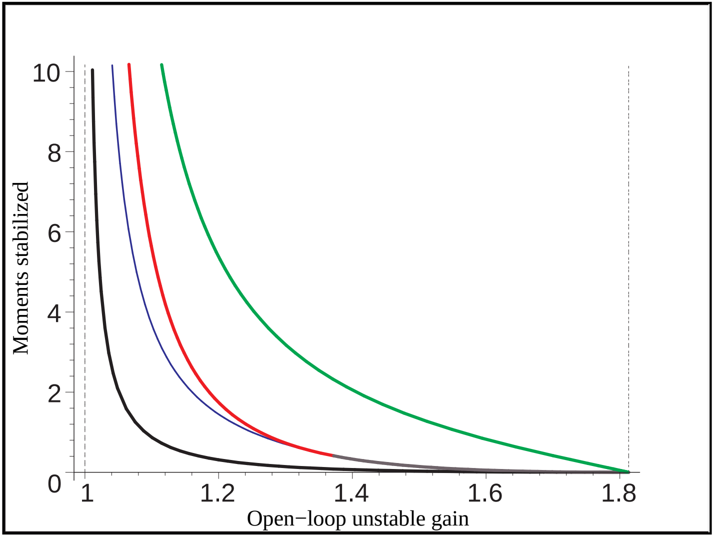
Truth in advertising: computation revisited

- Although we are doing better than exponential growth, we still have power laws on both sides.
- What if we needed a finite speed computer in the controller?
- Bad news:
  - Assume 0 control applied if we can not decode yet.
  - Power law for comp. implies power low for waiting.
  - Exponentially rare doubly exponentially bad states!

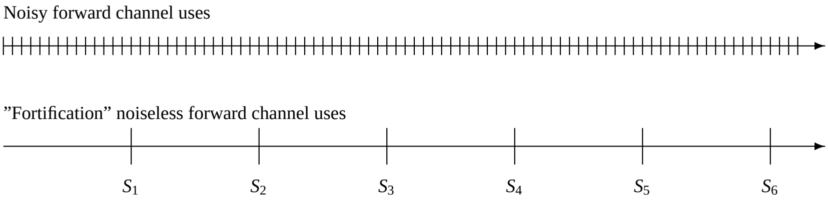
How to hit the higher bound?



How to hit the higher bound?

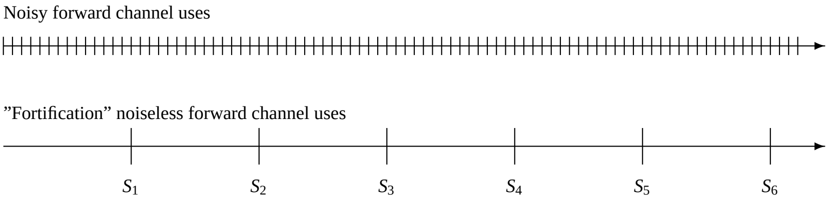


Fortified channels



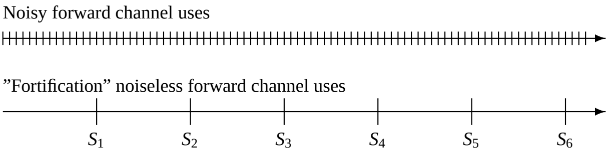
- Some mix of noisy and noiseless channels

Fortified channels



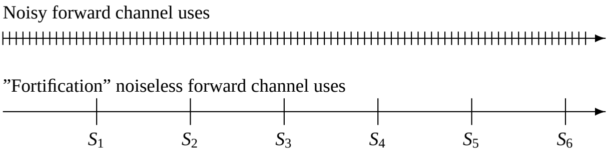
- Some mix of noisy and noiseless channels
- Is it all or nothing?

Noiseless channel can enable event-based sampling



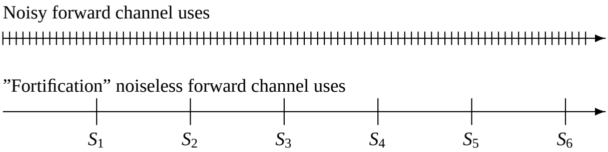
- Need to allow for gradual progress during bad periods.

Noiseless channel can enable event-based sampling

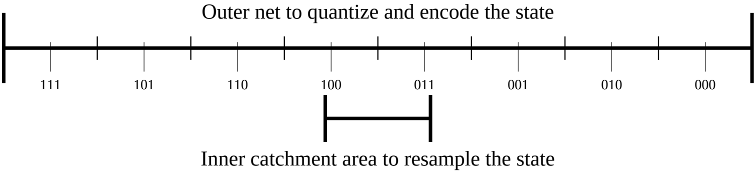


- Need to allow for gradual progress during bad periods.
- Use the noiseless channel for supervisory information:

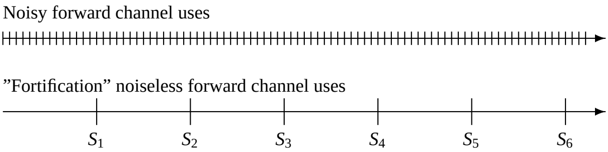
Noiseless channel can enable event-based sampling



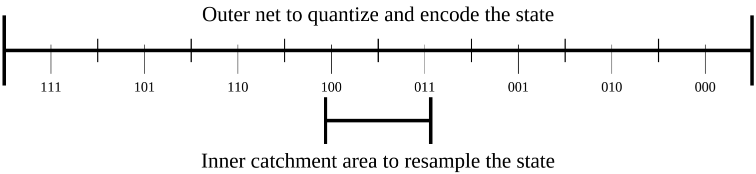
- Need to allow for gradual progress during bad periods.
- Use the noiseless channel for supervisory information:
  - Have the observer do event-based “sampling” of the state.



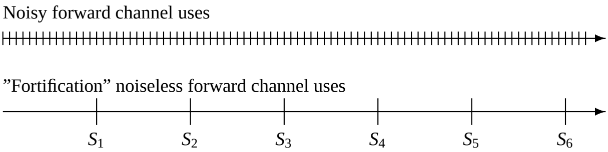
Noiseless channel can enable event-based sampling



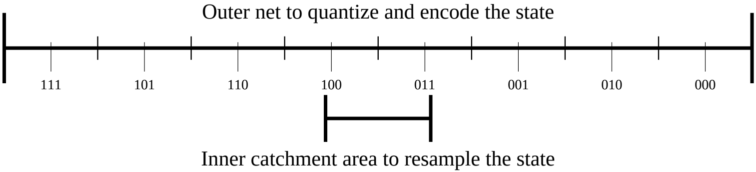
- Need to allow for gradual progress during bad periods.
- Use the noiseless channel for supervisory information:
  - Have the observer do event-based “sampling” of the state.
  - “Quantization net” grows as needed, but has only e^{nR} boxes.



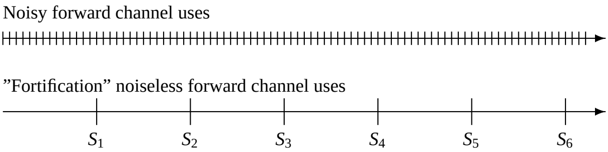
Noiseless channel can enable event-based sampling



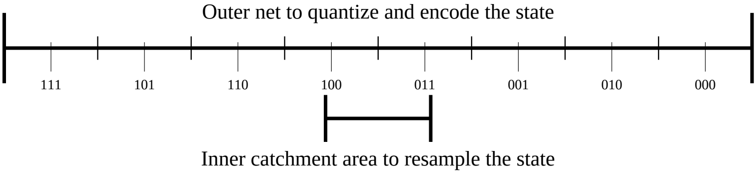
- Need to allow for gradual progress during bad periods.
- Use the noiseless channel for supervisory information:
  - Have the observer do event-based “sampling” of the state.
  - “Quantization net” grows as needed, but has only e^{nR} boxes.
  - Noiseless channel tells controller when it has “resampled.”



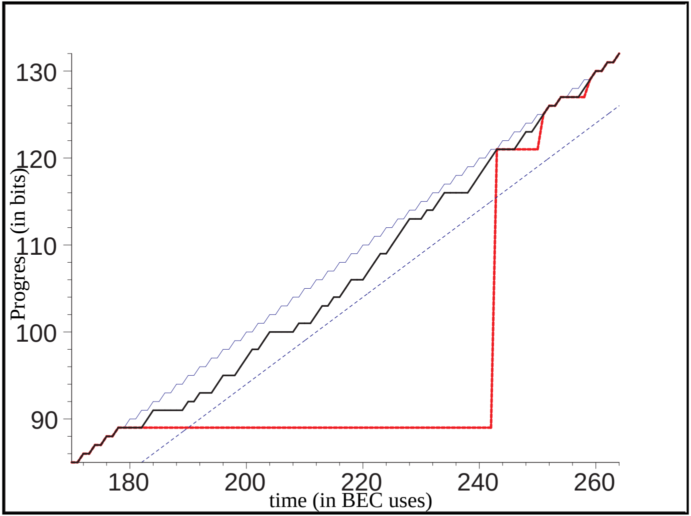
Noiseless channel can enable event-based sampling



- Need to allow for gradual progress during bad periods.
- Use the noiseless channel for supervisory information:
  - Have the observer do event-based “sampling” of the state.
  - “Quantization net” grows as needed, but has only e^{nR} boxes.
  - Noiseless channel tells controller when it has “resampled.”
- Use the noisy channel for variable-length block-coding.



Why gradual progress is better: intuition



Why this works: proof strategy

- Lift problem by using large nR

Why this works: proof strategy

- Lift problem by using large nR
  - Very few noiseless channel uses required

Why this works: proof strategy

- Lift problem by using large nR
  - Very few noiseless channel uses required
  - Stopping time for variable-length channel is like $n + \tilde{T}$, where $\tilde{T}$ is geometric $\exp(-E_0(\rho))$.

Why this works: proof strategy

- Lift problem by using large nR
  - Very few noiseless channel uses required
  - Stopping time for variable-length channel is like $n + \tilde{T}$, where $\tilde{T}$ is geometric $\exp(-E_0(\rho))$.
  - Interpret with $\ln \lambda < R = \frac{E_0(\rho)}{\rho} < \frac{E_0(\eta+\epsilon)}{\eta+\epsilon}$

Why this works: proof strategy

- Lift problem by using large nR
  - Very few noiseless channel uses required
  - Stopping time for variable-length channel is like $n + \tilde{T}$, where $\tilde{T}$ is geometric $\exp(-E_0(\rho))$.
  - Interpret with $\ln \lambda < R = \frac{E_0(\rho)}{\rho} < \frac{E_0(\eta+\epsilon)}{\eta+\epsilon}$
- Behaves like a “virtual” packet-erasure channel.

Why this works: proof strategy

- Lift problem by using large nR
  - Very few noiseless channel uses required
  - Stopping time for variable-length channel is like $n + \tilde{T}$, where $\tilde{T}$ is geometric $\exp(-E_0(\rho))$.
  - Interpret with $\ln \lambda < R = \frac{E_0(\rho)}{\rho} < \frac{E_0(\eta+\epsilon)}{\eta+\epsilon}$
- Behaves like a "virtual" packet-erasure channel.
  - Each packet carries $n(R - \ln \lambda)$ nats.

Why this works: proof strategy

- Lift problem by using large nR
  - Very few noiseless channel uses required
  - Stopping time for variable-length channel is like $n + \tilde{T}$, where $\tilde{T}$ is geometric $\exp(-E_0(\rho))$.
  - Interpret with $\ln \lambda < R = \frac{E_0(\rho)}{\rho} < \frac{E_0(\eta+\epsilon)}{\eta+\epsilon}$
- Behaves like a "virtual" packet-erasure channel.
  - Each packet carries $n(R - \ln \lambda)$ nats.
  - Disturbances grow by factor $O(\lambda^n)$

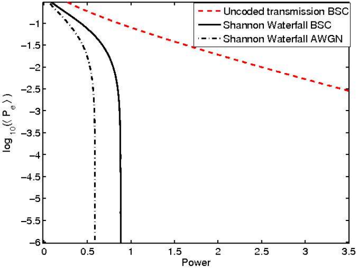
Why this works: proof strategy

- Lift problem by using large nR
  - Very few noiseless channel uses required
  - Stopping time for variable-length channel is like $n + \tilde{T}$, where $\tilde{T}$ is geometric $\exp(-E_0(\rho))$.
  - Interpret with $\ln \lambda < R = \frac{E_0(\rho)}{\rho} < \frac{E_0(\eta+\epsilon)}{\eta+\epsilon}$
- Behaves like a "virtual" packet-erasure channel.
  - Each packet carries $n(R - \ln \lambda)$ nats.
  - Disturbances grow by factor $O(\lambda^n)$
  - Erasure probability $exp(-E_0(\rho))$

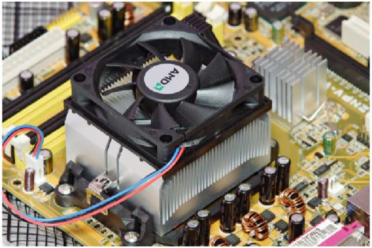
Outline

1. A bridge to nowhere?
   - A simple control problem
   - A connection to information theory
   - Fixing information theory and filling in the gaps.
2. Coming back to control
   - What is wrong with random coding
   - The role of noiseless feedback
3. Taking control thinking to the forefront of information theory.
   - The "holy grail" problem
   - Control thinking to the rescue!

The "holy grail:" understanding complexity

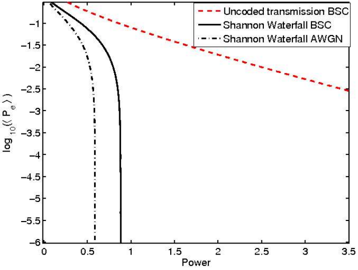


+

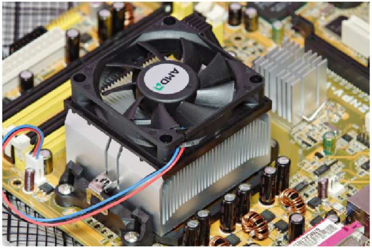


- Classical goal: arbitrarily low probability of error.
- Classical assumption: not delay sensitive at all.
- New twist: minimize **total power consumption**

The “holy grail:” understanding complexity

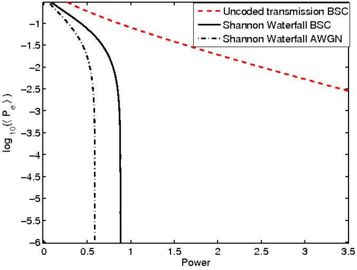


+

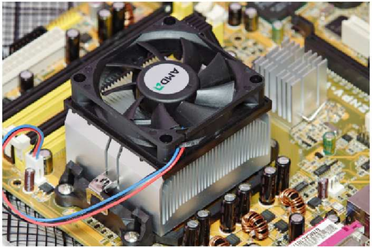


- Classical goal: arbitrarily low probability of error.
- Classical assumption: not delay sensitive at all.
- New twist: minimize **total power consumption**
- Important technology trends

The “holy grail:” understanding complexity

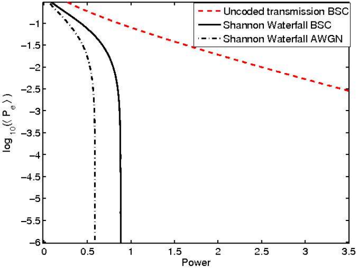


+

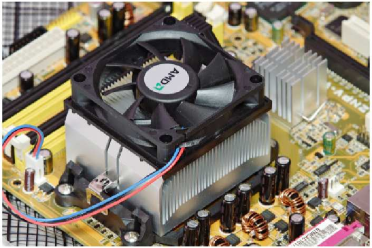


- Classical goal: arbitrarily low probability of error.
- Classical assumption: not delay sensitive at all.
- New twist: minimize **total power consumption**
- Important technology trends
  - “Moore’s law” allows billions of transistors, and but only mildly reduces power-consumption per transistor.

The “holy grail:” understanding complexity

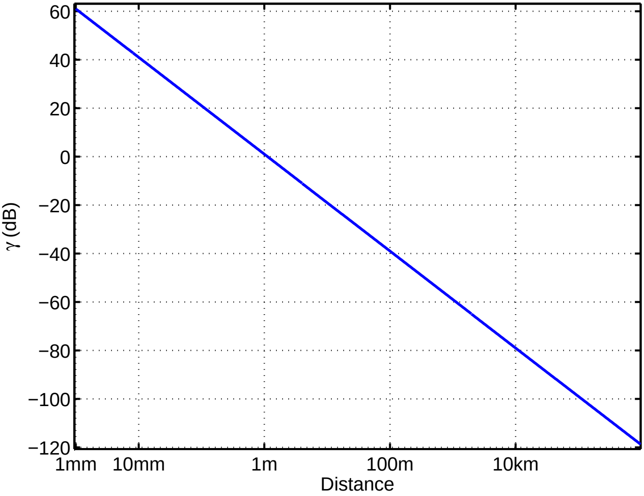


+

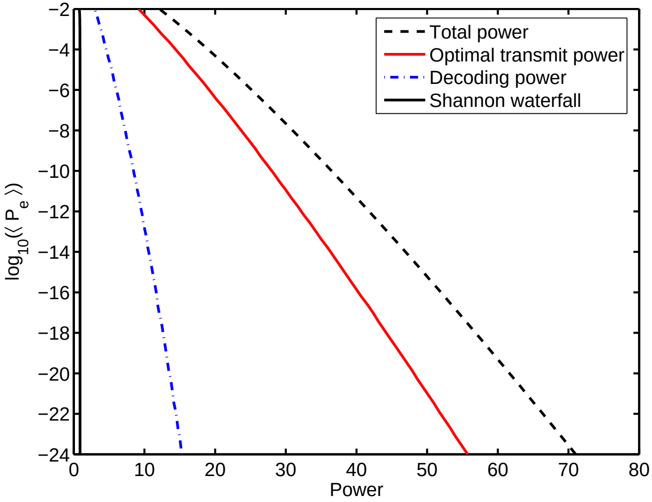


- Classical goal: arbitrarily low probability of error.
- Classical assumption: not delay sensitive at all.
- New twist: minimize **total power consumption**
- Important technology trends
  - “Moore’s law” allows billions of transistors, and but only mildly reduces power-consumption per transistor.
  - New short-range applications: swarm behavior, in-home networks, dense meshes, personal-area networks, UWB, between-chip communication, etc.

Decoding power vs communication range

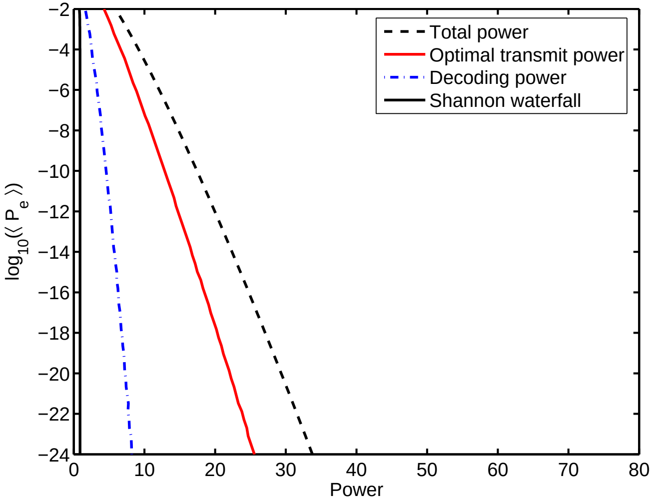


Dense linear codes with brute-force decoding



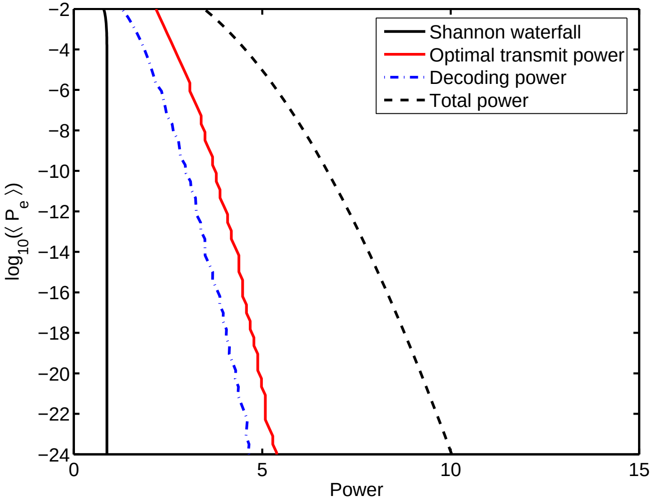
Decoding Power $nR2^{nR}$, Error Prob $2^{-E_{sp}(R,P)n}$

Convolutional codes with Viterbi decoding



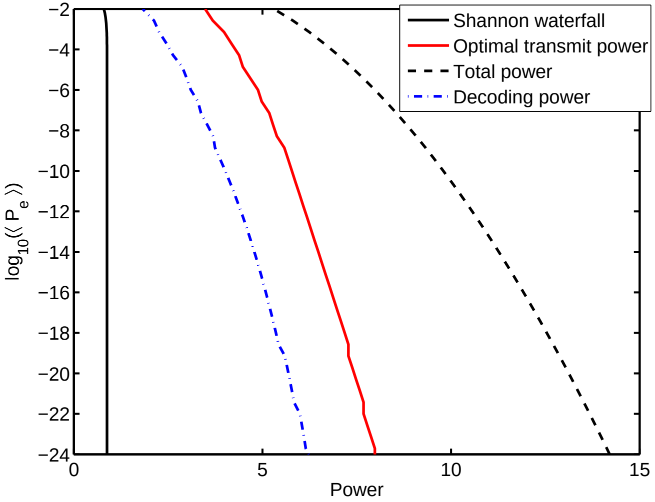
Decoding Power $L_c R 2^{L_c R}$, Error Prob $2^{-E_{conv}(R,P)L_c}$

Convolutional with “magical” sequential decoding



Decoding Power $L_c R$, Error Prob $2^{-E_{conv}(R,P)L_c}$

Dense linear codes with "magical" syndrome decoding



$$\text{Decoding Power } (1-R)nR\text{, Error Prob } 2^{-E_{sp}(R,P)n}$$

A new hope: iterative decoding

- Make assumptions about the decoder implementation rather than the code.

- Rich enough to capture LDPC, RA, Turbo, etc. codes.

A new hope: iterative decoding

- Make assumptions about the decoder implementation rather than the code.
  - Massively parallel computational nodes.
  - Each connected to at most $\alpha + 1$ other nodes.

- Rich enough to capture LDPC, RA, Turbo, etc. codes.

A new hope: iterative decoding

- Make assumptions about the decoder implementation rather than the code.
  - Massively parallel computational nodes.
  - Each connected to at most $\alpha + 1$ other nodes.
  - Each consumes E_{node} energy per iteration and can send *arbitrary* messages to its neighbors.

- Rich enough to capture LDPC, RA, Turbo, etc. codes.

A new hope: iterative decoding

- Make assumptions about the decoder implementation rather than the code.
  - Massively parallel computational nodes.
  - Each connected to at most $\alpha + 1$ other nodes.
  - Each consumes E_{node} energy per iteration and can send *arbitrary* messages to its neighbors.
  - Some nodes initialized with a single received codeword symbol.
  - Some nodes responsible for decoding a single message bit.

- Rich enough to capture LDPC, RA, Turbo, etc. codes.

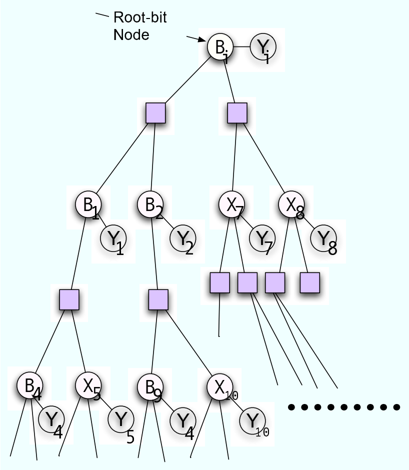
A new hope: iterative decoding

- Make assumptions about the decoder implementation rather than the code.
  - Massively parallel computational nodes.
  - Each connected to at most $\alpha + 1$ other nodes.
  - Each consumes E_{node} energy per iteration and can send *arbitrary* messages to its neighbors.
  - Some nodes initialized with a single received codeword symbol.
  - Some nodes responsible for decoding a single message bit.
  - Run for a fixed number of iterations i.
- Rich enough to capture LDPC, RA, Turbo, etc. codes.

A new hope: iterative decoding

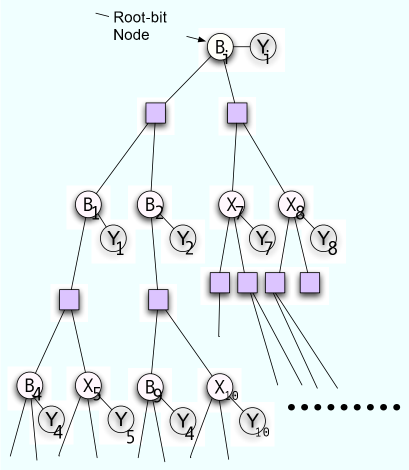
- Make assumptions about the decoder implementation rather than the code.
  - Massively parallel computational nodes.
  - Each connected to at most $\alpha + 1$ other nodes.
  - Each consumes E_{node} energy per iteration and can send *arbitrary* messages to its neighbors.
  - Some nodes initialized with a single received codeword symbol.
  - Some nodes responsible for decoding a single message bit.
  - Run for a fixed number of iterations i.
- Rich enough to capture LDPC, RA, Turbo, etc. codes.
- Power-consumption $\geq iE_{node}$ per received sample.

How to lower-bound the number of iterations?



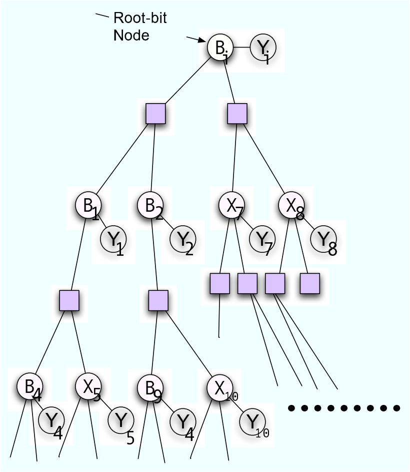
- Key concept: *decoding neighborhoods = information patterns*

How to lower-bound the number of iterations?



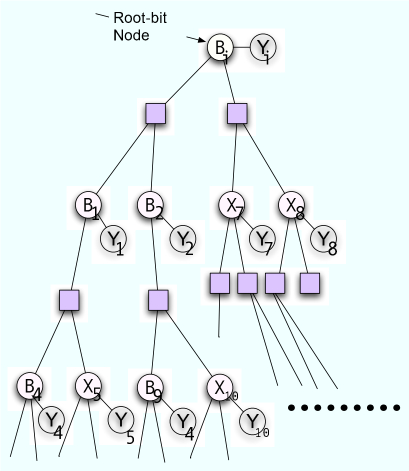
- Key concept: *decoding neighborhoods = information patterns*
- Decoding neighborhood size $n \leq 1 + (\alpha + 1)\alpha^{i-1} \approx \alpha^i$.

How to lower-bound the number of iterations?



- Key concept: *decoding neighborhoods = information patterns*
- Decoding neighborhood size $n \leq 1 + (\alpha + 1)\alpha^{i-1} \approx \alpha^i$.
- Need to lower-bound average probability of bit error in terms of n.

How to lower-bound the number of iterations?



- Key concept: *decoding neighborhoods = information patterns*
- Decoding neighborhood size $n \leq 1 + (\alpha + 1)\alpha^{i-1} \approx \alpha^i$.
- Need to lower-bound average probability of bit error in terms of n.
- Key insight: n is playing a role analogous to delay.

A **local** “sphere-packing” bound for the AWGN

Decoding neighborhood size $n \leq 1 + (\alpha+1)\alpha^{i-1} \approx \alpha^i$.

$\langle P_e \rangle \geq \sup_{\sigma_G^2 > \sigma_P^2 \mu(n):\ C(G) < R}$

$$\frac{h_b^{-1}(\delta(G))}{2} \exp\left(-nD(\sigma_G^2 || \sigma_P^2) - \frac{1}{2}\phi(n, h_b^{-1}(\delta(G)))\left(\frac{\sigma_G^2}{\sigma_P^2} - 1\right)\right)$$

- $C(G) = \frac{1}{2}\log_2(1 + \frac{P_T}{\sigma_G^2})$, $\delta(G)$: $1 - \frac{C(G)}{R}$

A **local** “sphere-packing” bound for the AWGN

Decoding neighborhood size $n \leq 1 + (\alpha+1)\alpha^{i-1} \approx \alpha^i$.

$\langle P_e \rangle \geq \sup_{\sigma_G^2 > \sigma_P^2 \mu(n):\, C(G) < R}$

$$\frac{h_b^{-1}(\delta(G))}{2} \exp\left(-nD(\sigma_G^2 || \sigma_P^2) - \frac{1}{2}\phi(n, h_b^{-1}(\delta(G)))\left(\frac{\sigma_G^2}{\sigma_P^2} - 1\right)\right)$$

- $C(G) = \frac{1}{2}\log_2(1 + \frac{P_T}{\sigma_G^2})$, $\delta(G)$: $1 - \frac{C(G)}{R}$
- $\mu(n) = \frac{1}{2}(1 + \frac{1}{T(n)+1} + \frac{4T(n)+2}{nT(n)(1+T(n))})$
- $T(n) = -W_L(-\exp(-1)(1/4)^{1/n})$
- $W_L(x)$ solves $x = W_L(x)\exp(W_L(x))$
- $\phi(n,y) = -n(W_L\left(-\exp(-1)(\frac{y}{2})^{\frac{2}{n}}\right) + 1)$

A **local** "sphere-packing" bound for the AWGN

Decoding neighborhood size $n \leq 1 + (\alpha + 1)\alpha^{i-1} \approx \alpha^i$.

$\langle P_e \rangle \geq \sup_{\sigma_G^2 > \sigma_P^2 \mu(n):\, C(G) < R}$

$$\frac{h_b^{-1}\left(\delta(G)\right)}{2} \exp\left(-nD(\sigma_G^2 || \sigma_P^2) - \frac{1}{2}\phi(n, h_b^{-1}\left(\delta(G)\right)) \left(\frac{\sigma_G^2}{\sigma_P^2} - 1\right)\right)$$

- $C(G) = \frac{1}{2}\log_2(1 + \frac{P_T}{\sigma_G^2})$, $\delta(G)$: $1 - \frac{C(G)}{R}$
- $\mu(n) = \frac{1}{2}(1 + \frac{1}{T(n)+1} + \frac{4T(n)+2}{nT(n)(1+T(n))})$
- $T(n) = -W_L(-\exp(-1)(1/4)^{1/n})$
- $W_L(x)$ solves $x = W_L(x)\exp(W_L(x))$
- $\phi(n, y) = -n(W_L\left(-\exp(-1)(\frac{y}{2})^{\frac{2}{n}}\right) + 1)$

Double-exponential potential return on investments in decoding power!

Waterslide curves for general AWGN case