Communication against restricted adversaries: between Shannon and Hamming

IEEE ITSOC Distinguished Lecture, Chinese University of Hong Kong

Anand D. Sarwate

Rutgers University

24 July 2025

Reliable communication, revisited

- Alice wants to send a message $m \in [M]$ to Bob using a *codeword* of n symbols.

Reliable communication, revisited

- Alice wants to send a message $m \in [M]$ to Bob using a *codeword* of n symbols.
- The link between Alice and Bob is *unreliable*.

Reliable communication, revisited

- Alice wants to send a message $m \in [M]$ to Bob using a *codeword* of n symbols.
- The link between Alice and Bob is *unreliable*.
- How many possible messages M can Alice successfully send?

Reliable communication, revisited

- Alice wants to send a message $m \in [M]$ to Bob using a *codeword* of n symbols.
- The link between Alice and Bob is *unreliable*.
- How many possible messages M can Alice successfully send?

This problem has been studied to death! What more is there to understand?

Reliable communication, revisited

- Alice wants to send a message $m \in [M]$ to Bob using a *codeword* of n symbols.
- The link between Alice and Bob is *unreliable*.
- How many possible messages M can Alice successfully send?

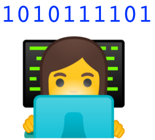
This problem has been studied to death! What more is there to understand?

Let's zoom in on binary channels with erasures.

Shannon theory: the channel is *random*

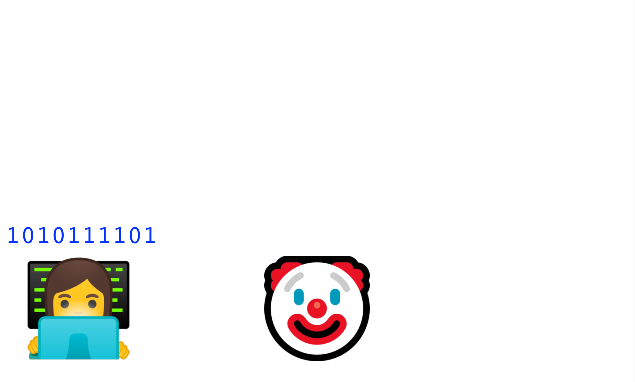
In the Shannon theory view, the channel acts *randomly*: $\approx pn$ positions are erased. The channel's actions are **oblivious** to the input.

Shannon theory: the channel is *random*



In the Shannon theory view, the channel acts *randomly*: $\approx pn$ positions are erased. The channel's actions are **oblivious** to the input.

Shannon theory: the channel is *random*



In the Shannon theory view, the channel acts *randomly*: $\approx pn$ positions are erased. The channel's actions are **oblivious** to the input.

Shannon theory: the channel is *random*

In the Shannon theory view, the channel acts *randomly*: $\approx pn$ positions are erased. The channel's actions are **oblivious** to the input.

Shannon theory: the channel is *random*

In the Shannon theory view, the channel acts *randomly*: $\approx pn$ positions are erased. The channel's actions are **oblivious** to the input.

Shannon theory: the channel is *random*

In the Shannon theory view, the channel acts *randomly*: $\approx pn$ positions are erased. The channel's actions are **oblivious** to the input.

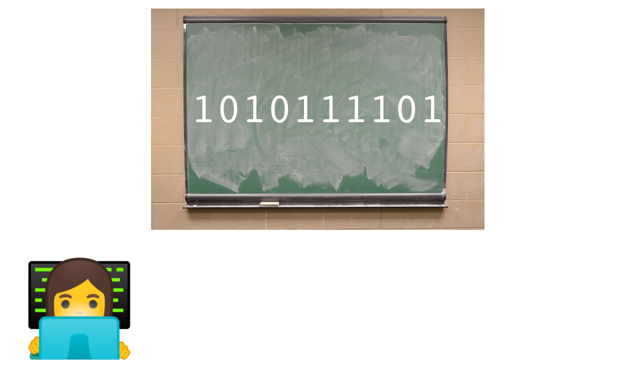
Coding theory ("Hamming"): the erasures are random

In the Shannon theory view, the channel acts *adversarially*: $\leq pn$ positions are erased. The channel's actions are **omniscient** w.r.t. to the input.

Coding theory (“Hamming”): the erasures are random

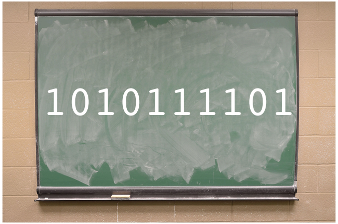
In the Shannon theory view, the channel acts *adversarially*: $\leq pn$ positions are erased. The channel’s actions are **omniscient** w.r.t. to the input.

Coding theory (“Hamming”): the erasures are random



In the Shannon theory view, the channel acts *adversarially*: $\leq pn$ positions are erased. The channel’s actions are **omniscient** w.r.t. to the input.

Coding theory ("Hamming"): the erasures are random



In the Shannon theory view, the channel acts *adversarially*: $\leq pn$ positions are erased. The channel's actions are **omniscient** w.r.t. to the input.

Coding theory ("Hamming"): the erasures are random

In the Shannon theory view, the channel acts *adversarially*: $\leq pn$ positions are erased. The channel's actions are **omniscient** w.r.t. to the input.

Coding theory (“Hamming”): the erasures are random

In the Shannon theory view, the channel acts *adversarially*: $\leq pn$ positions are erased. The channel’s actions are **omniscient** w.r.t. to the input.

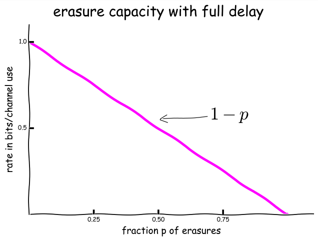
Coding theory (“Hamming”): the erasures are random

In the Shannon theory view, the channel acts *adversarially*: $\leq pn$ positions are erased. The channel’s actions are **omniscient** w.r.t. to the input.

Coding theory ("Hamming"): the erasures are random

In the Shannon theory view, the channel acts *adversarially*: $\leq pn$ positions are erased. The channel's actions are **omniscient** w.r.t. to the input.

The erasure channel: adversarial vs. random

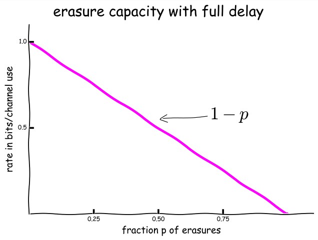


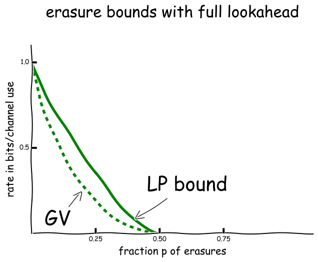
With the (Shannon-like) oblivious *average-case* model, the capacity is

$$C = 1 - p.$$

And we can achieve it many different ways.

The erasure channel: adversarial vs. random





With the (Shannon-like) oblivious *average-case* model, the capacity is

$$C = 1 - p.$$

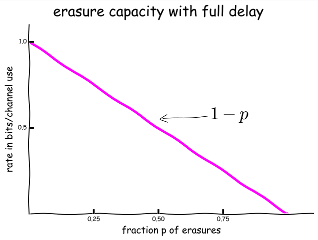
And we can achieve it many different ways.

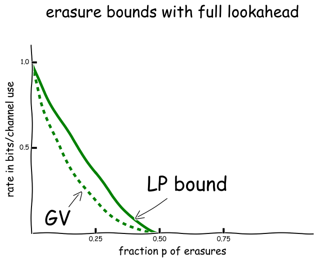
With the (Hamming-like) omniscient *worst-case* model, the capacity is

$$C \leq 1 - 2p.$$

We can get a lower bound using Gilbert-Varshamov (random) codes.

The erasure channel: adversarial vs. random





With the (Shannon-like) oblivious *average-case* model, the capacity is

$$C = 1 - p.$$

And we can achieve it many different ways.

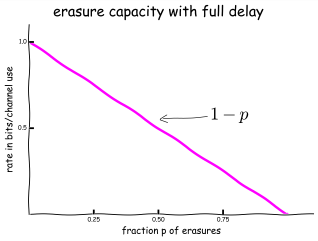
With the (Hamming-like) omniscient *worst-case* model, the capacity is

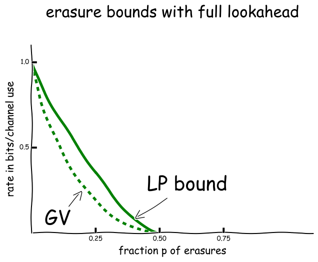
$$C \leq 1 - 2p.$$

We can get a lower bound using Gilbert-Varshamov (random) codes.

That's a big gap...

The erasure channel: adversarial vs. random





With the (Shannon-like) oblivious *average-case* model, the capacity is

$$C = 1 - p.$$

And we can achieve it many different ways.

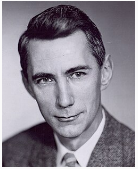
With the (Hamming-like) omniscient *worst-case* model, the capacity is

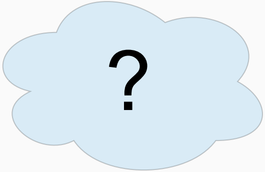
$$C \leq 1 - 2p.$$

We can get a lower bound using Gilbert-Varshamov (random) codes.

That's a big gap... where does it come from?

Why more is there to explore here?





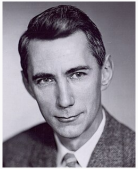
We are suggesting a different line of attack:

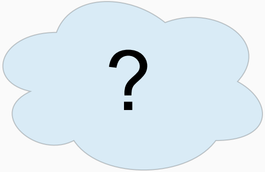
Why more is there to explore here?

We are suggesting a different line of attack:

1. Use **arbitrarily varying channels (AVCs)** to develop a **unified framework** for both the Shannon and Hamming models.

Why more is there to explore here?





We are suggesting a different line of attack:

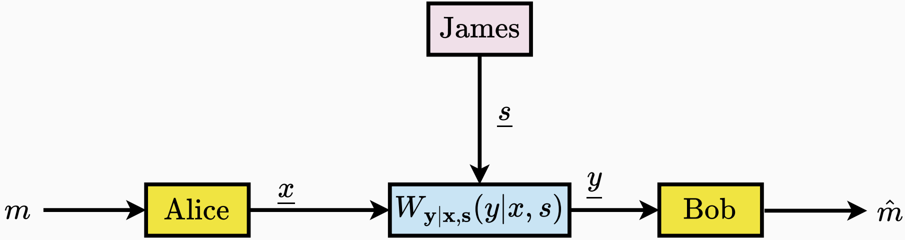
1. Use **arbitrarily varying channels (AVCs)** to develop a **unified framework** for both the Shannon and Hamming models.
2. Explore **intermediate models** to see **what lies in the the gap**.

Why more is there to explore here?

We are suggesting a different line of attack:

1. Use **arbitrarily varying channels (AVCs)** to develop a **unified framework** for both the Shannon and Hamming models.
2. Explore **intermediate models** to see **what lies in the the gap**.
3. Discover **coding strategies** and **new attacks/converses** to see what **resources are needed** to communicate reliably.

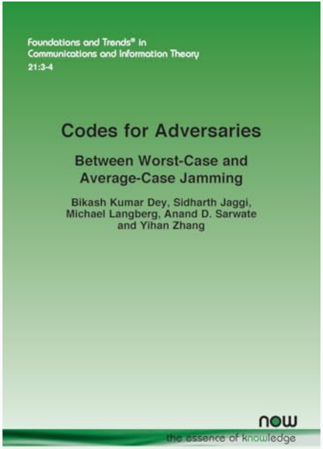
AVCs model channel “noise” as a state variable



In an **adversarial channel model**, **Alice** wants to communicate with **Bob** over a channel whose time-varying state is controlled by an adversarial **jammer** James.

- Alice and James may be **constrained** in how they communicate.
- Capacity depends on **what James knows** about m and $\underline{x}$.

Shameless self-promotion



We have a monograph (December 2024!) on coding against adversarial interference in a variety of settings using the framework of **arbitrarily varying channels**:

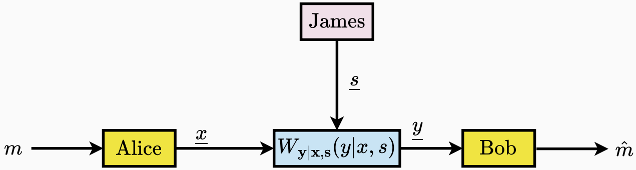
- ✔ Unified treatment of random noise (Shannon-theoretic) and worst-case noise (coding-theoretic).
- ✔ Intermediate models for jammers who can eavesdrop: online and myopic.
- ✔ Examples, open problems, and more!

What's coming up next

1. Arbitrarily varying channels (AVCs)
2. Some key ingredients
3. Causal adversarial models
4. Myopic adversarial models
5. Computationally efficient codes for causal adversaries
6. Looking forward

Arbitrarily varying channels (AVCs)

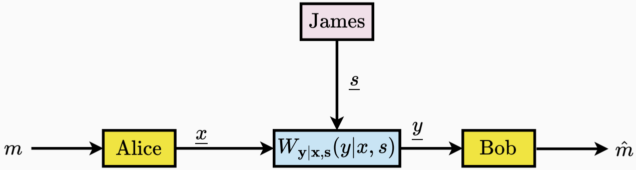
The basic channel model



Let $\mathcal{X}$, $\mathcal{S}$, and $\mathcal{Y}$ be discrete alphabets. An AVC is a discrete channel $W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s)$ such that

$$W_{\underline{\mathbf{y}}|\underline{\mathbf{x}},\underline{\mathbf{s}}}(\underline{y}|\underline{x},\underline{s}) = \prod_{i=1}^{n} W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y_i|x_i,s_i)$$

The basic channel model

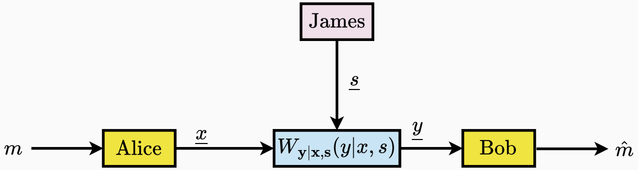


Let $\mathcal{X}$, $\mathcal{S}$, and $\mathcal{Y}$ be discrete alphabets. An AVC is a discrete channel $W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s)$ such that

$$W_{\underline{\mathbf{y}}|\underline{\mathbf{x}},\underline{\mathbf{s}}}(\underline{y}|\underline{x},\underline{s}) = \prod_{i=1}^{n} W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y_i|x_i,s_i)$$

The **state** $\underline{s} \in \mathcal{S}^n$ is controlled by an adversarial **jammer** (James).

The basic channel model



Let $\mathcal{X}$, $\mathcal{S}$, and $\mathcal{Y}$ be discrete alphabets. An AVC is a discrete channel $W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s)$ such that

$$W_{\underline{\mathbf{y}}|\underline{\mathbf{x}},\underline{\mathbf{s}}}(\underline{y}|\underline{x},\underline{s}) = \prod_{i=1}^{n} W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y_i|x_i,s_i)$$

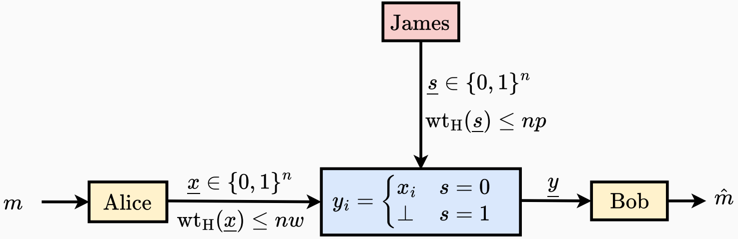
The **state** $\underline{s} \in \mathcal{S}^n$ is controlled by an adversarial **jammer** (James).
Examples: For binary channels $\underline{s}$ could be the error erasure pattern.

Input and cost constraints for AVCs

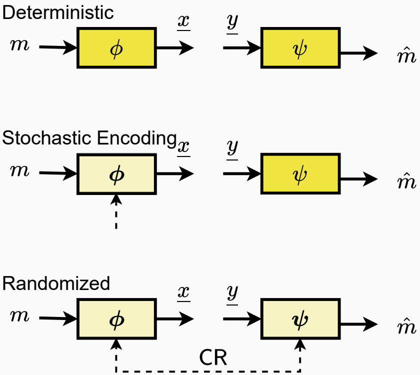
We impose that the types $T_{\underline{x}}$ and $T_{\underline{s}}$ of the codeword $\underline{x}$ and the state $\underline{s}$ lie be in convex subsets of the probability simplices $\Delta(\mathcal{X})$ and $\Delta(\mathcal{S})$:

$$T_{\underline{x}} \in \Gamma \subseteq \Delta(\mathcal{X})$$
$$T_{\underline{s}} \in \Lambda \subseteq \Delta(\mathcal{S})$$

Example: For binary channels $\underline{x}$ and $\underline{s}$ have bounded Hamming weight.



Defining codes and input constraints



An (n, M, Γ) code is

$$\phi : [M] \to \mathcal{X}^n \qquad \text{(encoder)}$$
$$\psi : \mathcal{Y}^n \to [M] \qquad \text{(decoder)}$$

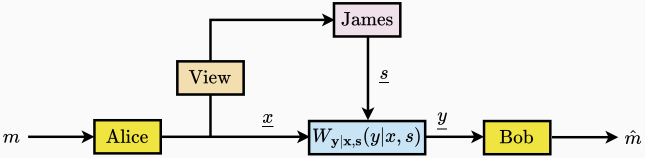
such that

$$T_{\phi(m)} \in \Gamma$$

The rate is $R = \frac{1}{n} \log_2(M)$.

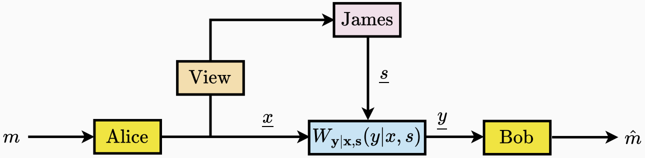
A **randomized code** lets Alice and Bob choose their code in secret. If Alice and Bob do not share common randomness, Alice can still use **stochastic encoding**.

What James knows: Shannon, Hamming, and in between



James wants to choose $\underline{s}$ to maximize the probability of error for **Bob**. What James can do depends on what he knows:

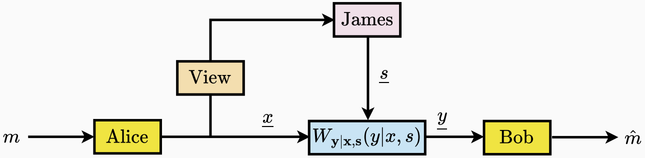
What James knows: Shannon, Hamming, and in between



James wants to choose $\underline{s}$ to maximize the probability of error for **Bob**. What James can do depends on what he knows:

- The message: target small **maximal (over messages) error**.

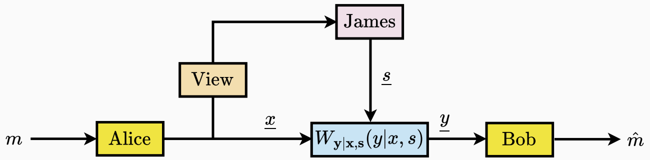
What James knows: Shannon, Hamming, and in between



James wants to choose $\underline{s}$ to maximize the probability of error for **Bob**. What James can do depends on what he knows:

- The message: target small **maximal (over messages) error**.
- The codeword (fully or partially).

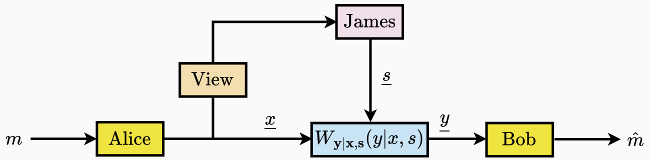
What James knows: Shannon, Hamming, and in between



James wants to choose $\underline{s}$ to maximize the probability of error for **Bob**. What James can do depends on what he knows:

- The message: target small **maximal (over messages) error**.
- The codeword (fully or partially).
- The randomness used by Alice (and/or Bob).

What James knows: Shannon, Hamming, and in between

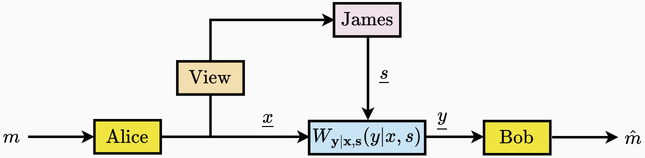


James wants to choose $\underline{s}$ to maximize the probability of error for **Bob**. What James can do depends on what he knows:

- The message: target small **maximal (over messages) error**.
- The codeword (fully or partially).
- The randomness used by Alice (and/or Bob).

These constrain the set of **strategies** James can use.

What James knows: Shannon, Hamming, and in between



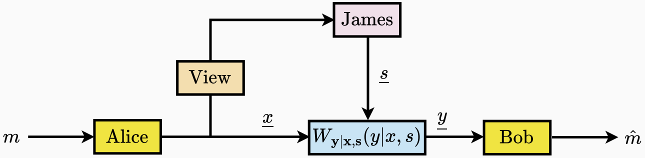
James wants to choose $\underline{s}$ to maximize the probability of error for **Bob**. What James can do depends on what he knows:

- The message: target small **maximal (over messages) error**.
- The codeword (fully or partially).
- The randomness used by Alice (and/or Bob).

These constrain the set of **strategies** James can use.

- **Oblivious** (Shannon): the message only.

What James knows: Shannon, Hamming, and in between



James wants to choose $\underline{s}$ to maximize the probability of error for **Bob**. What James can do depends on what he knows:

- The message: target small **maximal (over messages) error**.
- The codeword (fully or partially).
- The randomness used by Alice (and/or Bob).

These constrain the set of **strategies** James can use.

- **Oblivious** (Shannon): the message only.
- **Omniscient** (Hamming): the message and the codeword.

Maximal error and capacity

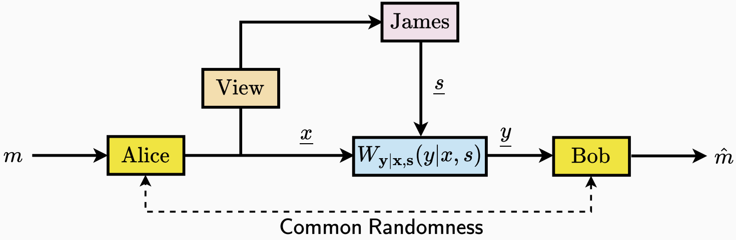
Maximal and average error:

$$P_{\mathrm{err}}(\phi, \psi) = \max_{\text{jamming strategies}} \frac{1}{M} \sum_{m=1}^{M} \sum_{\mathbf{x} \in \mathcal{X}^n} \mathbb{P}\left(\psi(\mathbf{y}) \neq m \mid \mathbf{x}\right) \mathbb{P}_{\phi}\left(\phi(m) = \mathbf{x}\right)$$

A rate R is achievable if for any $\epsilon > 0$ there exists an infinite sequence of rate R codes whose maximal probability of error is $< \epsilon$.
Let C_{obl} and C_{omni} be the capacities for oblivious and omniscient adversaries. In general

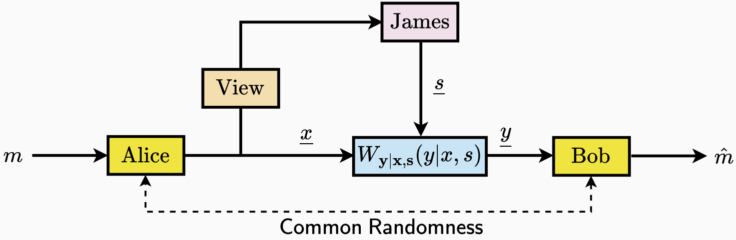
$$(\textit{Hamming}) \qquad C_{\mathrm{omni}} \qquad \leq \qquad C_{\mathrm{obl}} \qquad (\textit{Shannon})$$

Common randomness makes the problem easier



Blackwell et al. (1960) proposed the AVC model and studied **randomized codes**, where Alice and Bob share common randomness. James just minimizes the mutual information over equivalent DMCs:

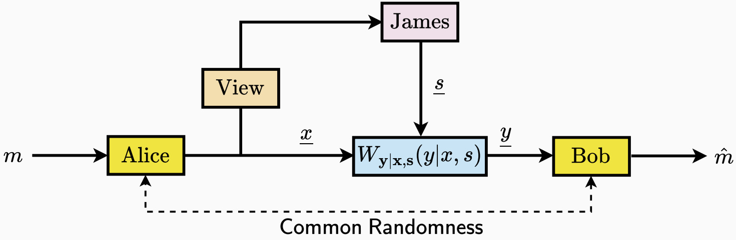
Common randomness makes the problem easier



Blackwell et al. (1960) proposed the AVC model and studied **randomized codes**, where Alice and Bob share common randomness. James just minimizes the mutual information over equivalent DMCs:

- **Oblivious:** find $\sum_s W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s)Q_{\mathbf{s}}(s)$ with lowest Shannon capacity.

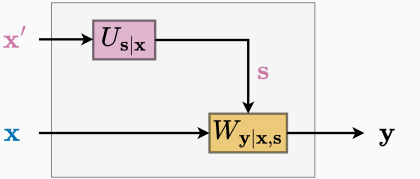
Common randomness makes the problem easier



Blackwell et al. (1960) proposed the AVC model and studied **randomized codes**, where Alice and Bob share common randomness. James just minimizes the mutual information over equivalent DMCs:

- **Oblivious:** find $\sum_s W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s)Q_{\mathbf{s}}(s)$ with lowest Shannon capacity.
- **Omniscient:** find $\sum_s W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s)U_{\mathbf{s}|\mathbf{x}}(s|x)$ with lowest Shannon capacity.

Deterministic codes and ECN Symmetrizability



An AVC is **Ericson-Csiszár-Narayan (ECN) symmetrizable** if James can spoof Alice's codeword. That is, for all $(\mathbf{y}, \mathbf{x}, \mathbf{x}')$, we have

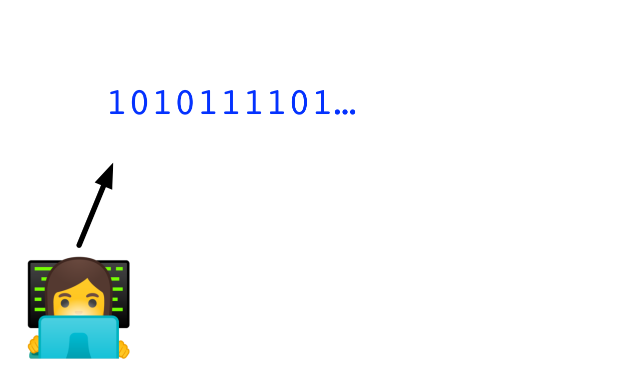
$$\sum_{s} U_{\mathbf{s}|\mathbf{x}'} W_{\mathbf{y}|\mathbf{x},\mathbf{s}} = \sum_{s} U_{\mathbf{s}|\mathbf{x}} W_{\mathbf{y}|\mathbf{x}',\mathbf{s}}.$$

Without common randomness, the capacity of a symmetrizable AVC $C_{\mathrm{obl}} = 0$.

Intermediate model 1: James gets delayed information

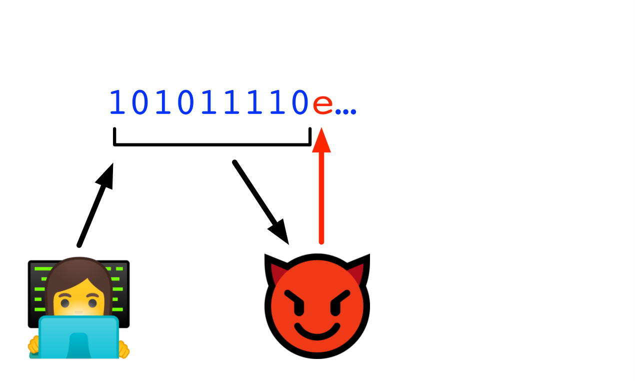
Delayed: James cannot use the full codeword in his strategy

Intermediate model 1: James gets delayed information



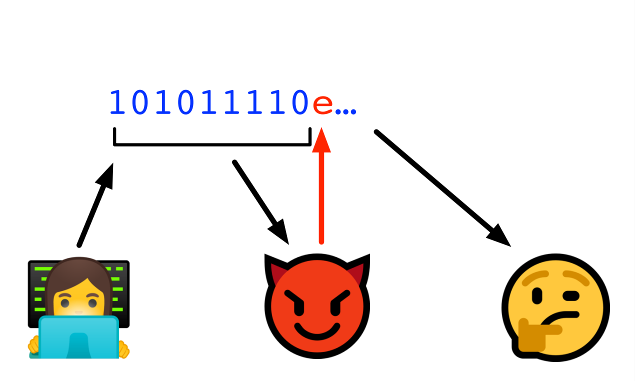
Delayed: James cannot use the full codeword in his strategy

Intermediate model 1: James gets delayed information



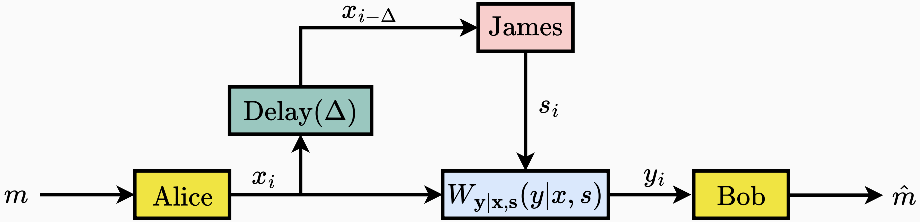
Delayed: James cannot use the full codeword in his strategy

Intermediate model 1: James gets delayed information



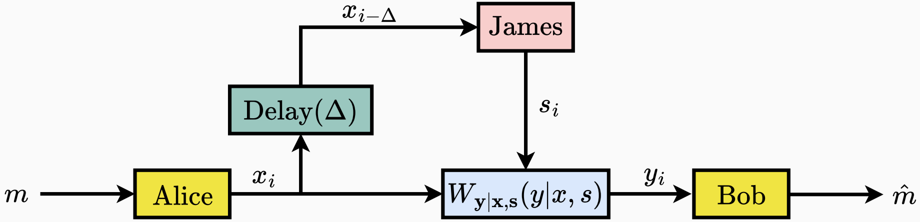
Delayed: James cannot use the full codeword in his strategy

Intermediate model 1: James gets delayed information

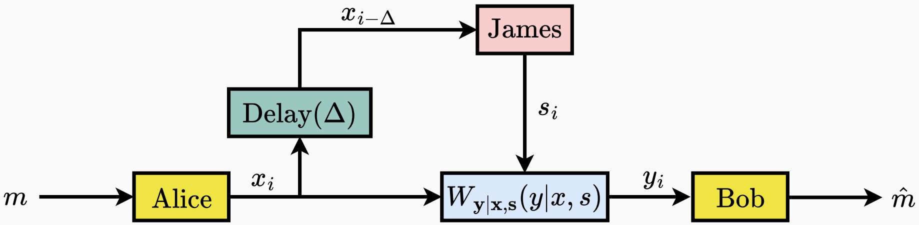


Delayed: James cannot use the full codeword in his strategy

The impact of delay in the erasure setting

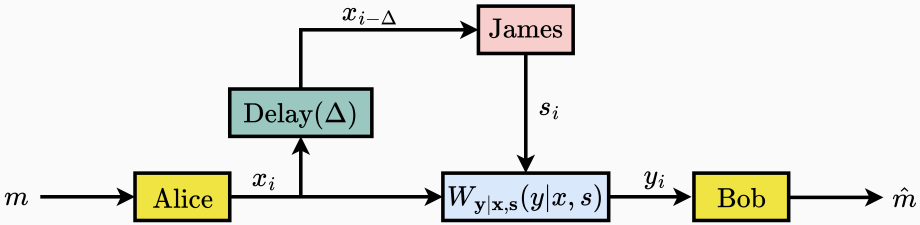


The impact of delay in the erasure setting



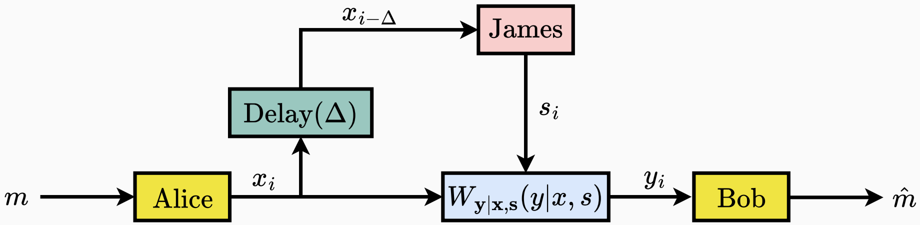
- $\Delta = n$ **(Oblivious):** capacity $= 1 - p$ ("Shannon")

The impact of delay in the erasure setting



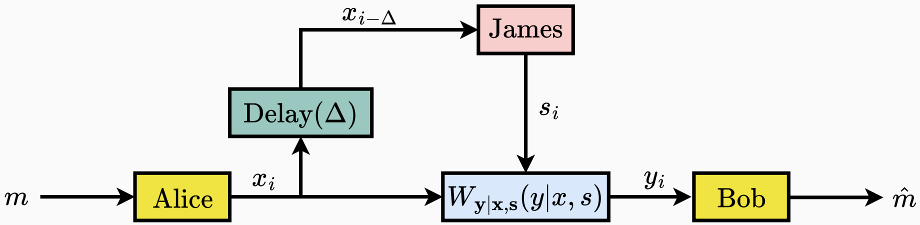
- $\Delta = n$ **(Oblivious):** capacity $= 1 - p$ ("Shannon")
- $\Delta = 1$ **("one bit delay"):** capacity $= 1 - p$

The impact of delay in the erasure setting



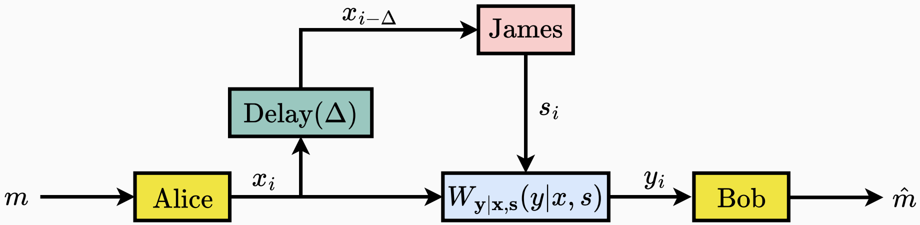
- $\Delta = n$ **(Oblivious):** capacity $= 1 - p$ ("Shannon")
- $\Delta = 1$ **("one bit delay"):** capacity $= 1 - p$
- $\Delta = 0$ **("causal"):** capacity $= 1 - 2p$

The impact of delay in the erasure setting



- $\Delta = n$ **(Oblivious):** capacity $= 1 - p$ ("Shannon")
- $\Delta = 1$ **("one bit delay"):** capacity $= 1 - p$
- $\Delta = 0$ **("causal"):** capacity $= 1 - 2p$
- $\Delta = -n$ **(Omniscient):** capacity $\leq 1 - 2p$ ("Hamming")

The impact of delay in the erasure setting



- $\Delta = n$ **(Oblivious):** capacity $= 1 - p$ ("Shannon")
- $\Delta = 1$ **("one bit delay"):** capacity $= 1 - p$
- $\Delta = 0$ **("causal"):** capacity $= 1 - 2p$
- $\Delta = -n$ **(Omniscient):** capacity $\leq 1 - 2p$ ("Hamming")

Knowing the current input gives James a lot of power!

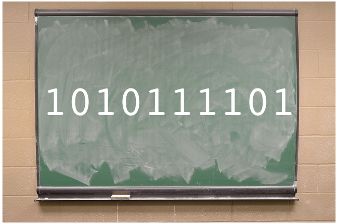
Intermediate model 2: Myopic adversarial models

Myopic: James gets a noisy view of the transmitted codeword.

Intermediate model 2: Myopic adversarial models

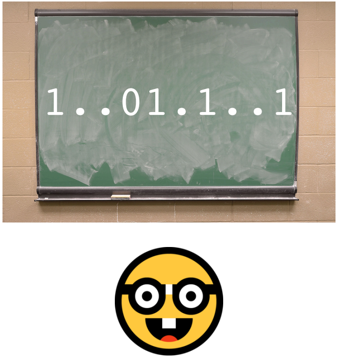
Myopic: James gets a noisy view of the transmitted codeword.

Intermediate model 2: Myopic adversarial models



Myopic: James gets a noisy view of the transmitted codeword.

Intermediate model 2: Myopic adversarial models



Myopic: James gets a noisy view of the transmitted codeword.

Intermediate model 2: Myopic adversarial models

Myopic: James gets a noisy view of the transmitted codeword.

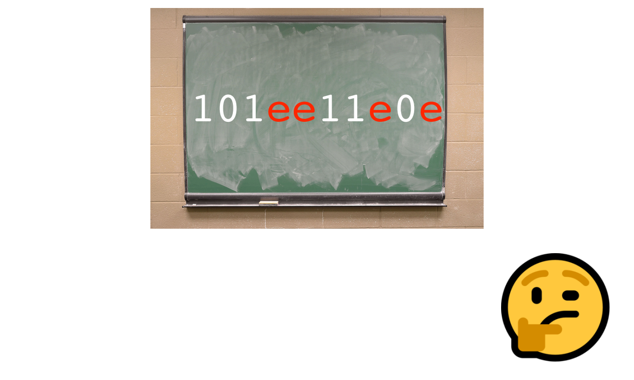
Intermediate model 2: Myopic adversarial models

Myopic: James gets a noisy view of the transmitted codeword.

Intermediate model 2: Myopic adversarial models

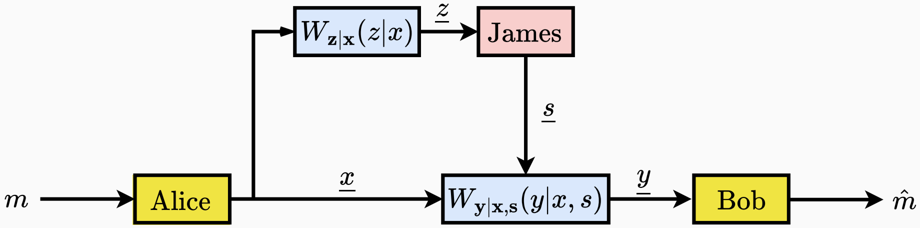
Myopic: James gets a noisy view of the transmitted codeword.

Intermediate model 2: Myopic adversarial models



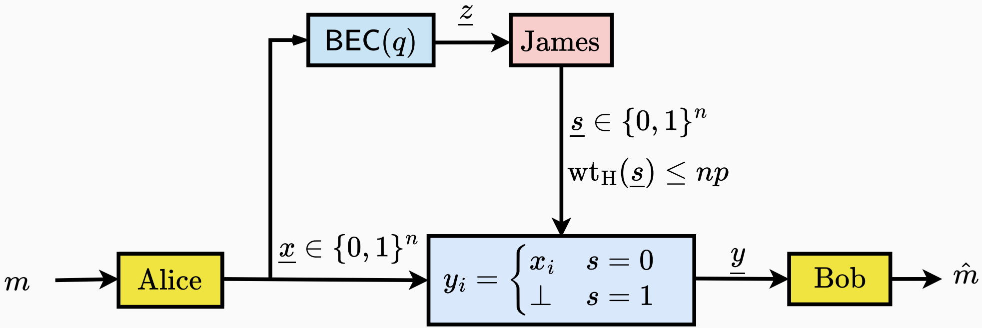
Myopic: James gets a noisy view of the transmitted codeword.

Intermediate model 2: Myopic adversarial models

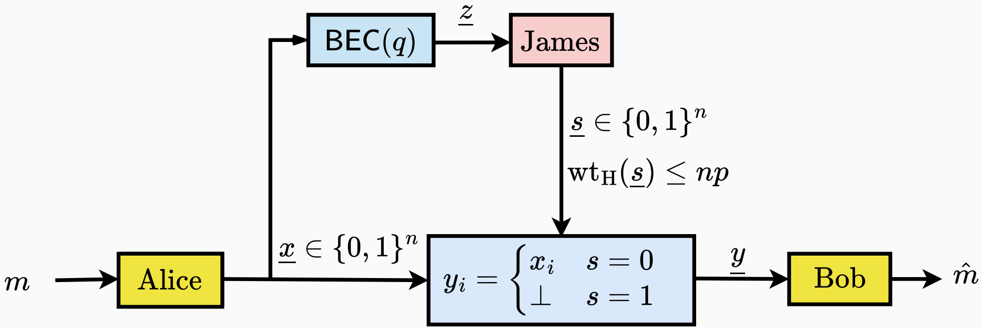


Myopic: James gets a noisy view of the transmitted codeword.

The impact of myopia in the erasure setting

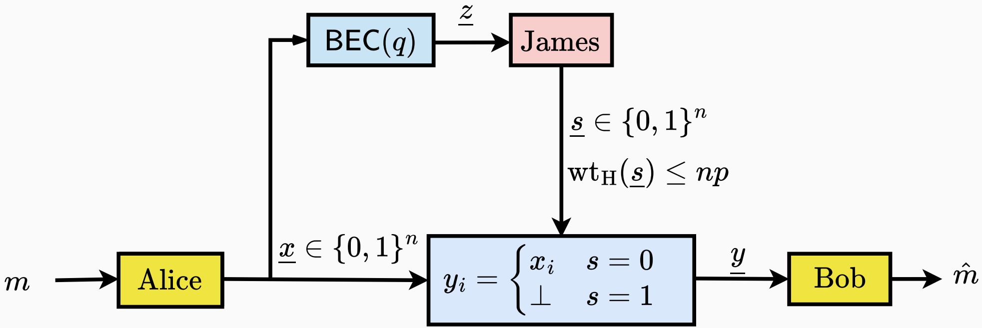


The impact of myopia in the erasure setting



- **Sufficiently myopic:** ($p < q$): capacity $= 1 - p$

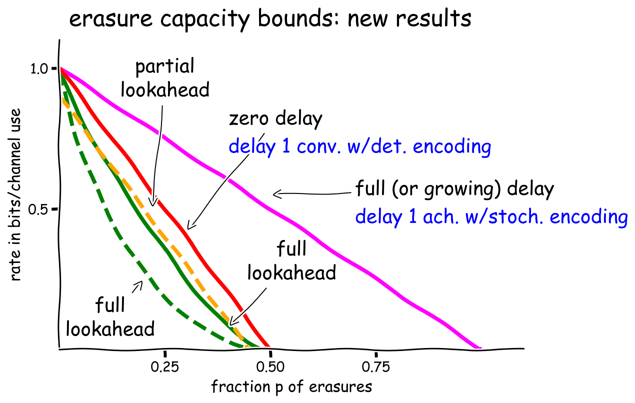
The impact of myopia in the erasure setting



- **Sufficiently myopic:** ($p < q$): capacity $= 1 - p$
- **Otherwise:** ($p > q$): it's more complicated...

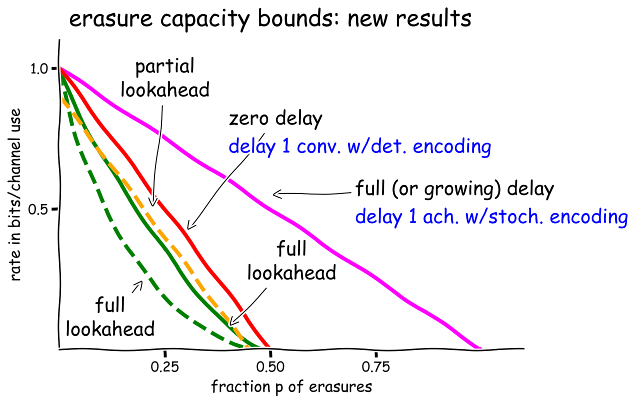
Some key ingredients

Ingredient 1: stochastic encoding



In **stochastic encoding**, Alice uses private randomness to create uncertainty for James

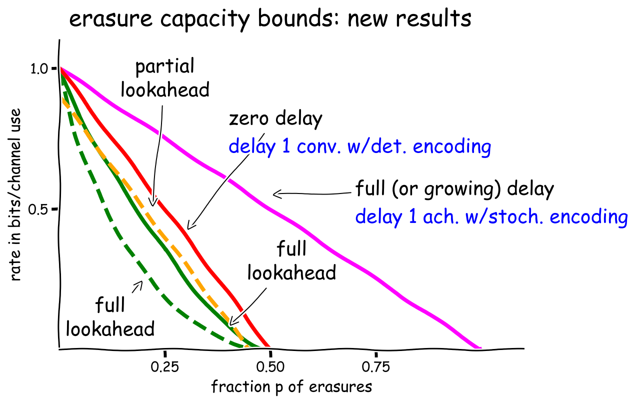
Ingredient 1: stochastic encoding



In **stochastic encoding**, Alice uses private randomness to create uncertainty for James

- Noise injection (c.f. secrecy).

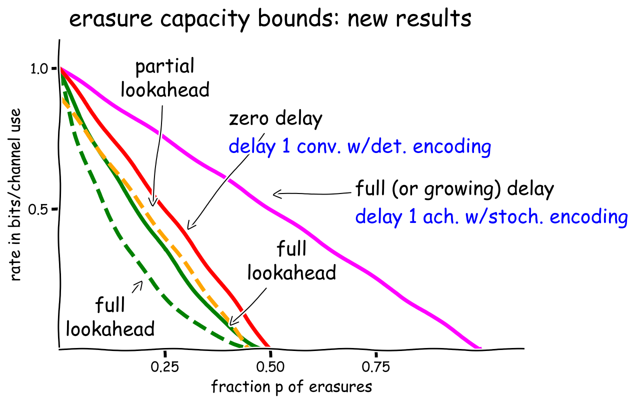
Ingredient 1: stochastic encoding



In **stochastic encoding**, Alice uses private randomness to create uncertainty for James

- Noise injection (c.f. secrecy).
- Low weight "fuzz" as a side channel.

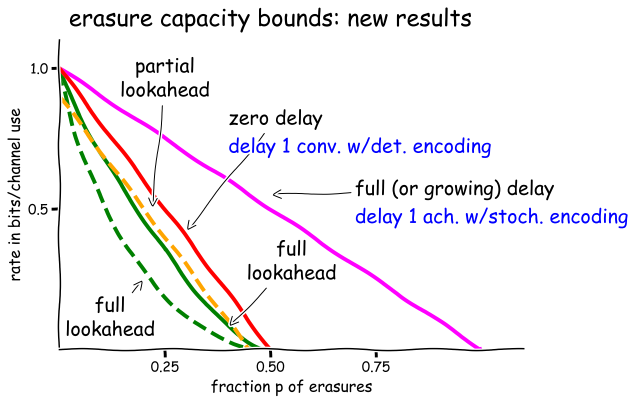
Ingredient 1: stochastic encoding



In **stochastic encoding**, Alice uses private randomness to create uncertainty for James

- Noise injection (c.f. secrecy).
- Low weight “fuzz” as a side channel.
- Select a codebook from a smaller “library”.

Ingredient 1: stochastic encoding

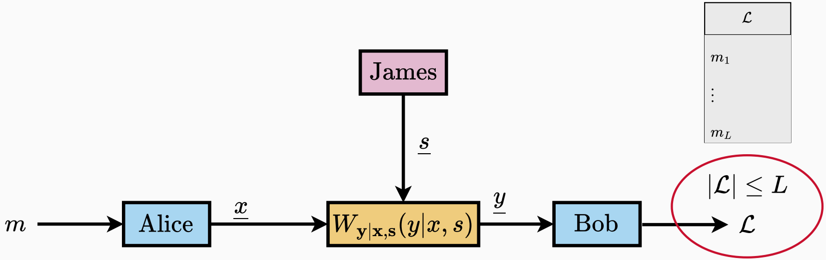


In **stochastic encoding**, Alice uses private randomness to create uncertainty for James

- Noise injection (c.f. secrecy).
- Low weight "fuzz" as a side channel.
- Select a codebook from a smaller "library".

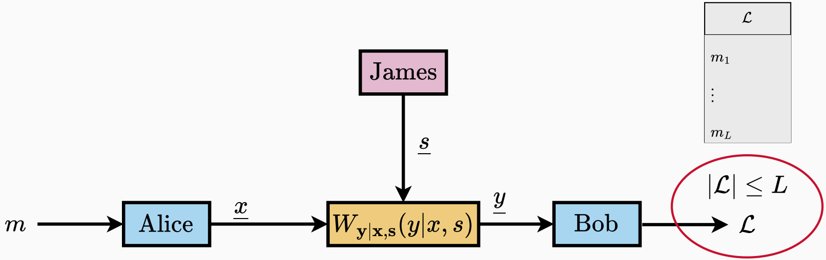
It can be **necessary**: deterministic erasure codes cannot do better than $1 - 2p$ against a James who has a single bit of delay.

Ingredient 2: list decoding



In **list decoding** we allow Bob to output a list $\mathcal{L}$.

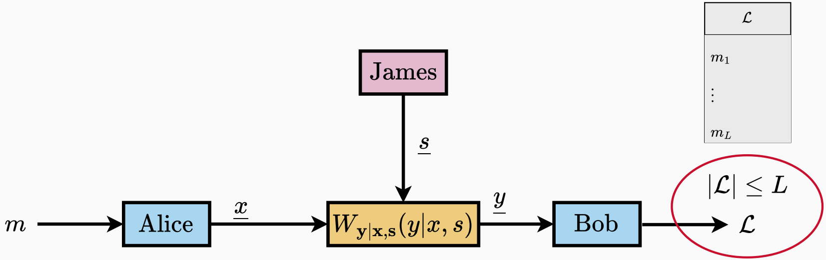
Ingredient 2: list decoding



In **list decoding** we allow Bob to output a list $\mathcal{L}$.

- Decoding is successful if the transmitted $m \in \mathcal{L}$.

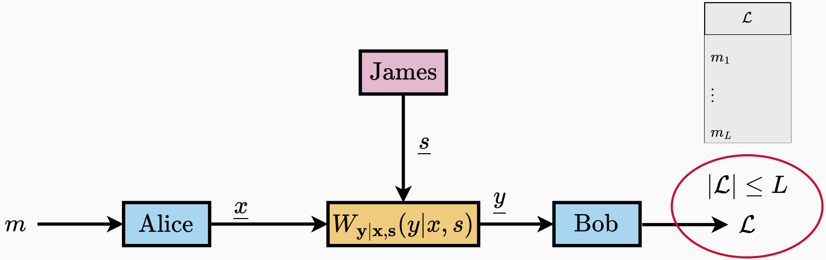
Ingredient 2: list decoding



In **list decoding** we allow Bob to output a list $\mathcal{L}$.

- Decoding is successful if the transmitted $m \in \mathcal{L}$.
- Require the list size is no larger than than L.

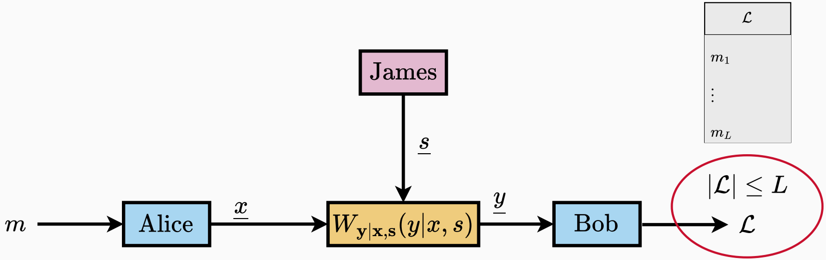
Ingredient 2: list decoding



In **list decoding** we allow Bob to output a list $\mathcal{L}$.

- Decoding is successful if the transmitted $m \in \mathcal{L}$.
- Require the list size is no larger than than L.
- Different L are useful in different cases: constant, $\mathrm{poly}(n)$, or $\exp(\epsilon n)$

Ingredient 2: list decoding



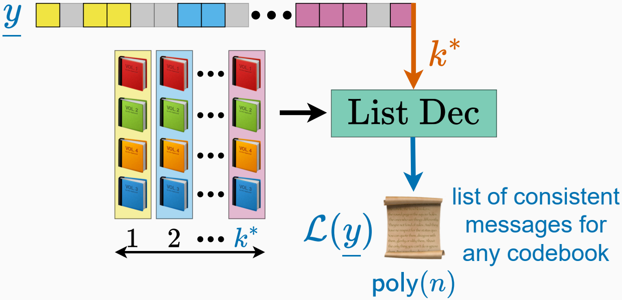
In **list decoding** we allow Bob to output a list $\mathcal{L}$.

- Decoding is successful if the transmitted $m \in \mathcal{L}$.
- Require the list size is no larger than than L.
- Different L are useful in different cases: constant, $\text{poly}(n)$, or $\exp(\epsilon n)$

In some cases the list decoding capacity can be **strictly larger**:

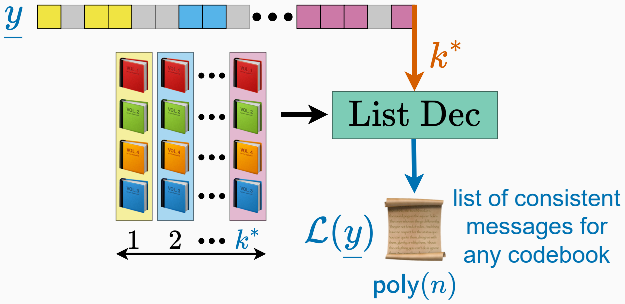
$$C_{\text{list}}(L) > C_{\text{obl}}.$$

List decoding can be useful in many ways



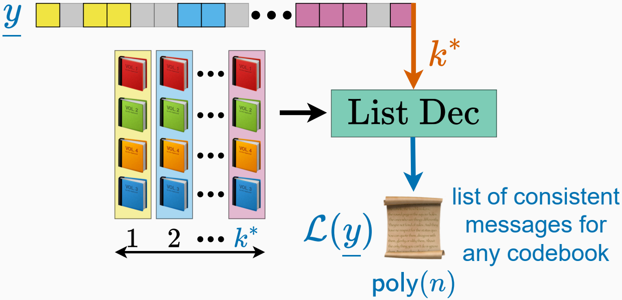
- Generalized notion of symmetrizability holds for list decoding to **constant list size.**

List decoding can be useful in many ways



- Generalized notion of symmetrizability holds for list decoding to **constant list size.**
- Alice/Bob can achieve the randomized coding capacity using $O(\log n)$ bits of common randomness using a list code with $L = \text{poly}(n)$.

List decoding can be useful in many ways



- Generalized notion of symmetrizability holds for list decoding to **constant list size.**
- Alice/Bob can achieve the randomized coding capacity using $O(\log n)$ bits of common randomness using a list code with $L = \text{poly}(n)$.
- Two-stage decoders which sequentially list decode in the first stage.

Ingredient 3: Completely Positive (CP) Couplings and the Plotkin Bound

A more technical ingredient which is particularly useful is the notion of **completely positive couplings**.

Ingredient 3: Completely Positive (CP) Couplings and the Plotkin Bound

A more technical ingredient which is particularly useful is the notion of **completely positive couplings**.

- Start with a marginal distribution $P_{\mathbf{x}} \in \Delta(\mathcal{X})$.

Ingredient 3: Completely Positive (CP) Couplings and the Plotkin Bound

A more technical ingredient which is particularly useful is the notion of **completely positive couplings**.

- Start with a marginal distribution $P_{\mathbf{x}} \in \Delta(\mathcal{X})$.
- A **self-coupling** is a joint distribution $P_{\mathbf{x},\mathbf{x}'}$ where each marginal is $P_{\mathbf{x}}$.

Ingredient 3: Completely Positive (CP) Couplings and the Plotkin Bound

A more technical ingredient which is particularly useful is the notion of **completely positive couplings**.

- Start with a marginal distribution $P_{\mathbf{x}} \in \Delta(\mathcal{X})$.
- A **self-coupling** is a joint distribution $P_{\mathbf{x},\mathbf{x}'}$ where each marginal is $P_{\mathbf{x}}$.
- A self-coupling is **completely positive** if it is a mixture of independent self-couplings:

$$P_{\mathbf{x},\mathbf{x}'}(x,x') = \sum_{i=1}^{|\mathcal{U}|} P_{\mathbf{u}}(i) P_{\mathbf{x}_i}(x) P_{\mathbf{x}_i}(x').$$

Generalizing the Plotkin bound

Question: can we have a codebook where all codewords have pairwise types that are ρ-far from a CP self-coupling?

$$\|T_{\underline{x},\underline{x}'} - P^{(\mathrm{CP})}_{\mathbf{x},\mathbf{x}'}\|_\infty > \rho \qquad \forall \underline{x}, \underline{x}', P^{(\mathrm{CP})}_{\mathbf{x},\mathbf{x}'} \tag{1}$$

Generalizing the Plotkin bound

Question: can we have a codebook where all codewords have pairwise types that are ρ-far from a CP self-coupling?

$$\|T_{\underline{x},\underline{x}'} - P^{(\mathrm{CP})}_{\mathbf{x},\mathbf{x}'}\|_\infty > \rho \qquad \forall \underline{x}, \underline{x}', P^{(\mathrm{CP})}_{\mathbf{x},\mathbf{x}'} \tag{1}$$

- It turns out that any codes with this property cannot be too large (for large n)!

Generalizing the Plotkin bound

Question: can we have a codebook where all codewords have pairwise types that are ρ-far from a CP self-coupling?

$$\|T_{\underline{x},\underline{x}'} - P^{(\mathrm{CP})}_{\mathbf{x},\mathbf{x}'}\|_\infty > \rho \qquad \forall \underline{x}, \underline{x}', P^{(\mathrm{CP})}_{\mathbf{x},\mathbf{x}'} \tag{1}$$

- It turns out that any codes with this property cannot be too large (for large n)!
- Compare this to the Plotkin bound: an upper bound on the size of binary codes with a given distance.

Generalizing the Plotkin bound

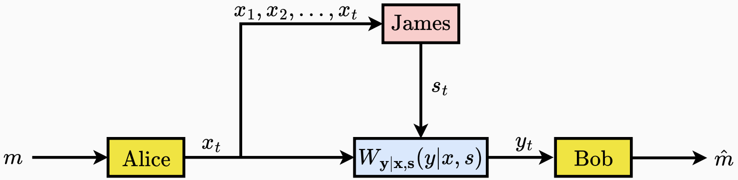
Question: can we have a codebook where all codewords have pairwise types that are ρ-far from a CP self-coupling?

$$\|T_{\underline{x},\underline{x}'} - P^{(CP)}_{\mathbf{x},\mathbf{x}'}\|_\infty > \rho \qquad \forall \underline{x}, \underline{x}', P^{(CP)}_{\mathbf{x},\mathbf{x}'} \tag{1}$$

- It turns out that any codes with this property cannot be too large (for large n)!
- Compare this to the Plotkin bound: an upper bound on the size of binary codes with a given distance.
- If our rate is too high, then there will a constant fraction of codeword pairs whose type is close to CP.

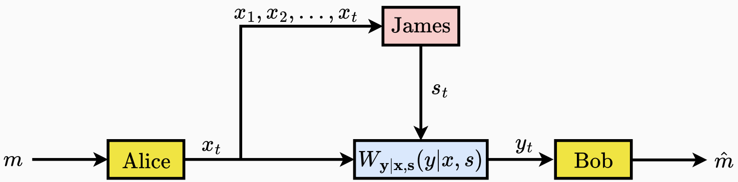
Causal adversarial models

Causal adversaries: James can see the current input



When can James "symmetrize" the channel and what does that mean? Think of James's constraint as a "power limitation":

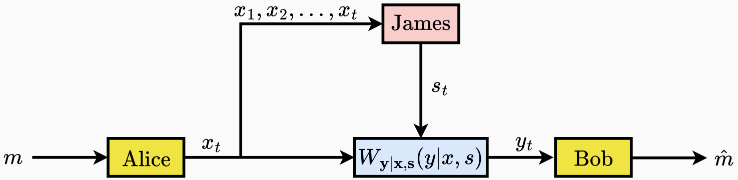
Causal adversaries: James can see the current input



When can James "symmetrize" the channel and what does that mean? Think of James's constraint as a "power limitation":

- Spend less power at the beginning to save it up and then push hard in the second half? Bob will get a better initial estimate.

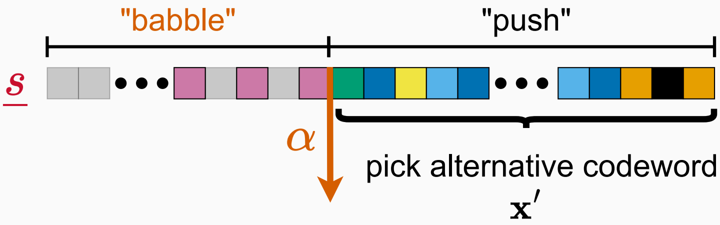
Causal adversaries: James can see the current input



When can James "symmetrize" the channel and what does that mean? Think of James's constraint as a "power limitation":

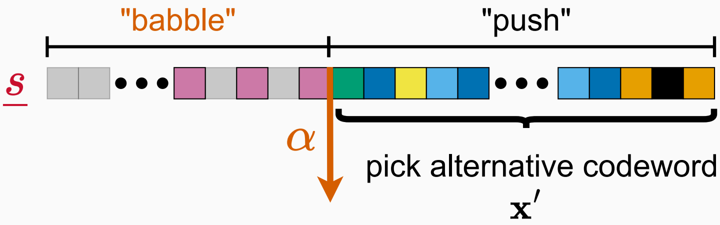
- Spend less power at the beginning to save it up and then push hard in the second half? Bob will get a better initial estimate.
- Spend more power at the beginning in the hope of leading Bob astray? But then the suffix might resolve Bob's uncertainty.

Babble and push: an attack for James



The main ideas in the converse, given a codebook $\mathcal{C}$ used by Alice and Bob:

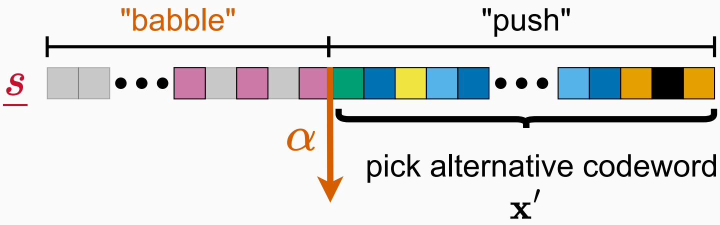
Babble and push: an attack for James



The main ideas in the converse, given a codebook $\mathcal{C}$ used by Alice and Bob:

1. James breaks the codewords into K chunks of lenfth $\epsilon_c n$ and “distills” a subcode which is a constant fraction of $\mathcal{C}$ and all codewords are approximately constant composition.

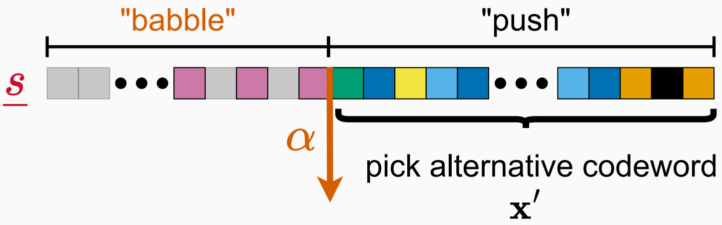
Babble and push: an attack for James



The main ideas in the converse, given a codebook $\mathcal{C}$ used by Alice and Bob:

1. James breaks the codewords into K chunks of lenfth $\epsilon_c n$ and “distills” a subcode which is a constant fraction of $\mathcal{C}$ and all codewords are approximately constant composition.
2. “Babble” by using a random attack $V_{\mathbf{s}|\mathbf{x},\mathbf{u}=u}$ for $u \leq \alpha K$.

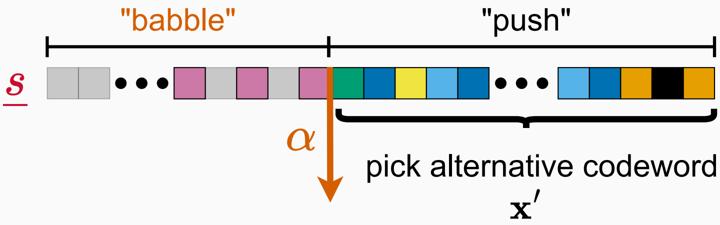
Babble and push: an attack for James



The main ideas in the converse, given a codebook $\mathcal{C}$ used by Alice and Bob:

1. James breaks the codewords into K chunks of lenfth $\epsilon_c n$ and "distills" a subcode which is a constant fraction of $\mathcal{C}$ and all codewords are approximately constant composition.
2. "Babble" by using a random attack $V_{\mathbf{s}|\mathbf{x},\mathbf{u}=u}$ for $u \leq \alpha K$.
3. Push by finding symmetrizing distributions $V_{\mathbf{s}|\mathbf{x},\mathbf{x}'\mathbf{u}=u}$ for the remaining chunks.

Babble and push: an attack for James

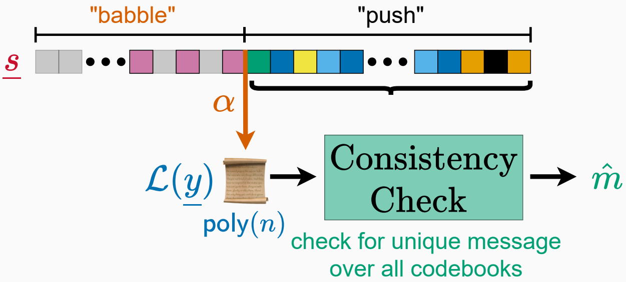


The main ideas in the converse, given a codebook $\mathcal{C}$ used by Alice and Bob:

1. James breaks the codewords into K chunks of lenfth $\epsilon_c n$ and "distills" a subcode which is a constant fraction of $\mathcal{C}$ and all codewords are approximately constant composition.
2. "Babble" by using a random attack $V_{\mathbf{s}|\mathbf{x},\mathbf{u}=u}$ for $u \leq \alpha K$.
3. Push by finding symmetrizing distributions $V_{\mathbf{s}|\mathbf{x},\mathbf{x}'\mathbf{u}=u}$ for the remaining chunks.

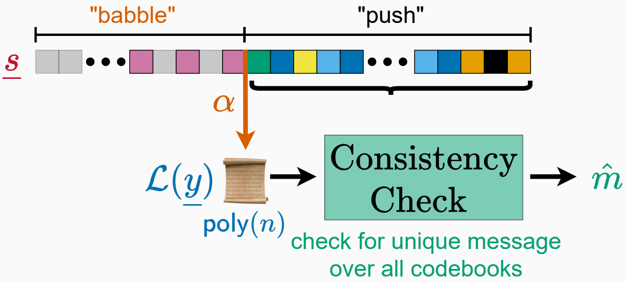
Use the generalized Plotkin bound (plus more) to show this will work.

Achievability



We can match the converse by using the same structure.

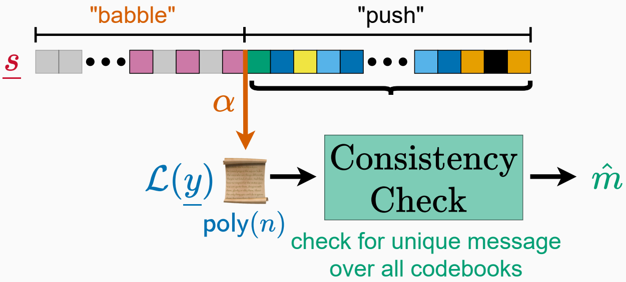
Achievability



We can match the converse by using the same structure.

- Encode m using independent randomness in each chunk.

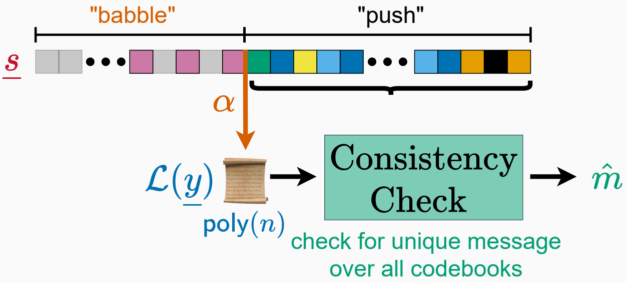
Achievability



We can match the converse by using the same structure.

- Encode *m* using independent randomness in each chunk.
- After each chunk, Bob tries to list decode by sequentially assuming James is using some random attacks $\{V_{\mathbf{s}|\mathbf{x},\mathbf{u}=u}\}$.

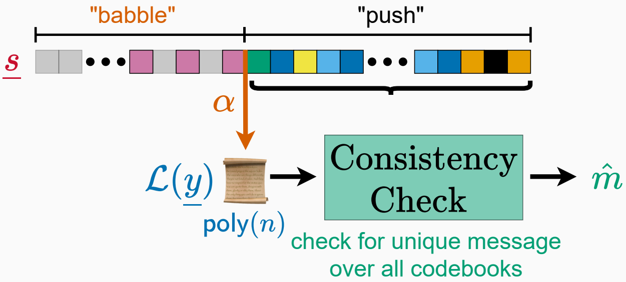
Achievability



We can match the converse by using the same structure.

- Encode m using independent randomness in each chunk.
- After each chunk, Bob tries to list decode by sequentially assuming James is using some random attacks $\{V_{\mathbf{s}|\mathbf{x},\mathbf{u}=u}\}$.
- If there is a message $\hat{m}$ and $\underline{s}$ such that the assumed attack and observed $\underline{y}$ he has seen so far are "feasible" then decode. Otherwise try another attack.

Achievability



We can match the converse by using the same structure.

- Encode m using independent randomness in each chunk.
- After each chunk, Bob tries to list decode by sequentially assuming James is using some random attacks $\{V_{\mathbf{s}|\mathbf{x},\mathbf{u}=u}\}$.
- If there is a message $\hat{m}$ and $\underline{s}$ such that the assumed attack and observed $\underline{y}$ he has seen so far are "feasible" then decode. Otherwise try another attack.

Basically have to define what "feasible" means in this setting (quite involved).

A multi-letter block characterization

Pros and cons:

A multi-letter block characterization

$$C := \limsup_{K\to\infty} \max_{\substack{P_{\mathbf{x}|\mathbf{u}}\in\Delta(\mathcal{X}|[1:K]) \\ \left[\mathrm{Unif}([K])P_{\mathbf{x}|\mathbf{u}}\right]_{\mathbf{x}}\in\Lambda_{\mathbf{x}}}} \min\Bigg\{ \min_{V_{\mathbf{s}|\mathbf{x},\mathbf{u}}\in\mathcal{F}(P_{\mathbf{x}|\mathbf{u}})} I(P_{\mathbf{x}|\mathbf{u}}, V_{\mathbf{s}|\mathbf{x},\mathbf{u}}),$$

$$\min_{\substack{(\alpha,(V_{\mathbf{s}|\mathbf{x},\mathbf{u}^{\leqslant\alpha}},V_{\mathbf{s}|\mathbf{x},\mathbf{x}',\mathbf{u}^{>\alpha}}))\in\left\{0,\frac{1}{K},\frac{2}{K},\cdots,1\right\}\times\mathcal{F}_{\alpha}(P_{\mathbf{x}|\mathbf{u}}) \\ \forall u\in[\alpha K+1:K],\, V_{\mathbf{s}|\mathbf{x},\mathbf{x}',\mathbf{u}^{>\alpha}=u}\in\mathcal{V}}} I(P_{\mathbf{x}|\mathbf{u}^{\leqslant\alpha}}, V_{\mathbf{s}|\mathbf{x},\mathbf{u}^{\leqslant\alpha}})\Bigg\}.$$

Pros and cons:

- ✗ We end up with a multi-letter expression for the capacity.

A multi-letter block characterization

$$C := \limsup_{K\to\infty} \max_{\substack{P_{\mathbf{x}|\mathbf{u}}\in\Delta(\mathcal{X}|[1:K]) \\ \left[\mathrm{Unif}([K])P_{\mathbf{x}|\mathbf{u}}\right]_{\mathbf{x}}\in\Lambda_{\mathbf{x}}}} \min\left\{ \min_{V_{\mathbf{s}|\mathbf{x},\mathbf{u}}\in\mathcal{F}(P_{\mathbf{x}|\mathbf{u}})} I(P_{\mathbf{x}|\mathbf{u}}, V_{\mathbf{s}|\mathbf{x},\mathbf{u}}), \right.$$

$$\left. \min_{\substack{(\alpha,(V_{\mathbf{s}|\mathbf{x},\mathbf{u}^{\leqslant\alpha}},V_{\mathbf{s}|\mathbf{x},\mathbf{x}',\mathbf{u}^{>\alpha}}))\in\left\{0,\frac{1}{K},\frac{2}{K},\cdots,1\right\}\times\mathcal{F}_{\alpha}(P_{\mathbf{x}|\mathbf{u}}) \\ \forall u\in[\alpha K+1:K],\, V_{\mathbf{s}|\mathbf{x},\mathbf{x}',\mathbf{u}^{>\alpha}=u}\in\mathcal{V}}} I(P_{\mathbf{x}|\mathbf{u}^{\leqslant\alpha}}, V_{\mathbf{s}|\mathbf{x},\mathbf{u}^{\leqslant\alpha}}) \right\}.$$

Pros and cons:

- ✗ We end up with a multi-letter expression for the capacity.
- ✔ Significantly generalizes prior arguments to general channels.

A multi-letter block characterization

$$C := \limsup_{K\to\infty} \max_{\substack{P_{\mathbf{x}|\mathbf{u}}\in\Delta(\mathcal{X}|[1:K]) \\ \left[\mathrm{Unif}([K])P_{\mathbf{x}|\mathbf{u}}\right]_{\mathbf{x}}\in\Lambda_{\mathbf{x}}}} \min\left\{ \min_{V_{\mathbf{s}|\mathbf{x},\mathbf{u}}\in\mathcal{F}(P_{\mathbf{x}|\mathbf{u}})} I(P_{\mathbf{x}|\mathbf{u}}, V_{\mathbf{s}|\mathbf{x},\mathbf{u}}), \right.$$

$$\left. \min_{\substack{(\alpha,(V_{\mathbf{s}|\mathbf{x},\mathbf{u}^{\leqslant\alpha}},V_{\mathbf{s}|\mathbf{x},\mathbf{x}',\mathbf{u}^{>\alpha}}))\in\left\{0,\frac{1}{K},\frac{2}{K},\cdots,1\right\}\times\mathcal{F}_{\alpha}(P_{\mathbf{x}|\mathbf{u}}) \\ \forall u\in[\alpha K+1:K],\, V_{\mathbf{s}|\mathbf{x},\mathbf{x}',\mathbf{u}^{>\alpha}=u}\in\mathcal{V}}} I(P_{\mathbf{x}|\mathbf{u}^{\leqslant\alpha}}, V_{\mathbf{s}|\mathbf{x},\mathbf{u}^{\leqslant\alpha}}) \right\}.$$

Pros and cons:

- ✗ We end up with a multi-letter expression for the capacity.
- ✔ Significantly generalizes prior arguments to general channels.
- ✔ Plotkin results may be useful elsewhere.

A multi-letter block characterization

$$C := \limsup_{K\to\infty} \max_{\substack{P_{\mathbf{x}|\mathbf{u}}\in\Delta(\mathcal{X}|[1:K]) \\ \left[\mathrm{Unif}([K])P_{\mathbf{x}|\mathbf{u}}\right]_{\mathbf{x}}\in\Lambda_{\mathbf{x}}}} \min\left\{ \min_{V_{\mathbf{s}|\mathbf{x},\mathbf{u}}\in\mathcal{F}(P_{\mathbf{x}|\mathbf{u}})} I(P_{\mathbf{x}|\mathbf{u}}, V_{\mathbf{s}|\mathbf{x},\mathbf{u}}), \right.$$

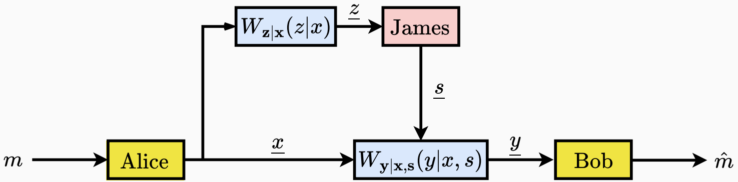
$$\left. \min_{\substack{(\alpha,(V_{\mathbf{s}|\mathbf{x},\mathbf{u}^{\leqslant\alpha}},V_{\mathbf{s}|\mathbf{x},\mathbf{x}',\mathbf{u}^{>\alpha}}))\in\left\{0,\frac{1}{K},\frac{2}{K},\cdots,1\right\}\times\mathcal{F}_\alpha(P_{\mathbf{x}|\mathbf{u}}) \\ \forall u\in[\alpha K+1:K],\, V_{\mathbf{s}|\mathbf{x},\mathbf{x}',\mathbf{u}^{>\alpha}=u}\in\mathcal{V}}} I(P_{\mathbf{x}|\mathbf{u}^{\leqslant\alpha}}, V_{\mathbf{s}|\mathbf{x},\mathbf{u}^{\leqslant\alpha}}) \right\}.$$

Pros and cons:

- ✗ We end up with a multi-letter expression for the capacity.
- ✔ Significantly generalizes prior arguments to general channels.
- ✔ Plotkin results may be useful elsewhere.
- ✗ Relies on some additional assumptions.

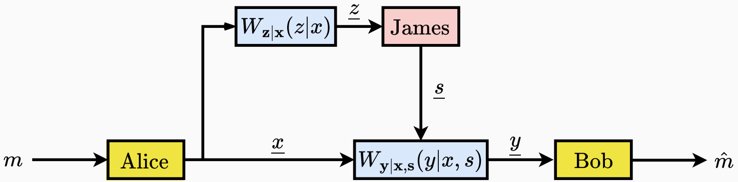
Myopic adversarial models

Myopic adversaries: James sees the whole codeword in noise



In a myopic AVC, James gets to see the entire codeword corrupted by a DMC $W_{\mathbf{z}|\mathbf{x}}$.

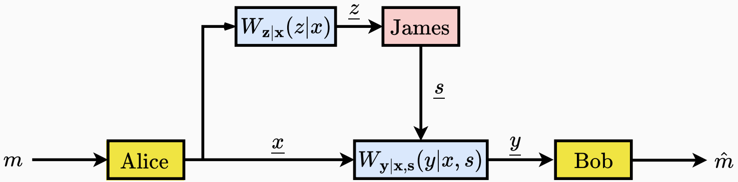
Myopic adversaries: James sees the whole codeword in noise



In a myopic AVC, James gets to see the entire codeword corrupted by a DMC $W_{\mathbf{z}|\mathbf{x}}$.

- Jamming strategies are maps $[M] \times \mathcal{Z}^n \to \mathcal{S}^n$.

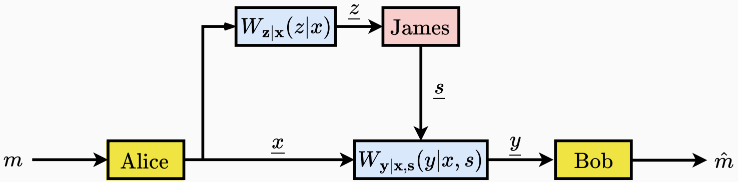
Myopic adversaries: James sees the whole codeword in noise



In a myopic AVC, James gets to see the entire codeword corrupted by a DMC $W_{\mathbf{z}|\mathbf{x}}$.

- Jamming strategies are maps $[M] \times \mathcal{Z}^n \to \mathcal{S}^n$.
- For randomized codes we can again look for the worst DMC $\sum_s W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s) W_{\mathbf{z}|\mathbf{x}} V_{\mathbf{s}|\mathbf{z}}(s|z)$.

Myopic adversaries: James sees the whole codeword in noise



In a myopic AVC, James gets to see the entire codeword corrupted by a DMC $W_{\mathbf{z}|\mathbf{x}}$.

- Jamming strategies are maps $[M] \times \mathcal{Z}^n \to \mathcal{S}^n$.
- For randomized codes we can again look for the worst DMC $\sum_s W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x, s) W_{\mathbf{z}|\mathbf{x}} V_{\mathbf{s}|\mathbf{z}}(s|z)$.
- By changing $W_{\mathbf{z}|\mathbf{x}}$ we can get the oblivious and omniscient settings.

Symmetrizability for myopic AVCs

A myopic AVC is said to be symmetrizable under input distribution $P_{\mathbf{x}} \in \Gamma$ if there exists a channel $U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}$ such that for all x, x', y,

Symmetrizability for myopic AVCs

A myopic AVC is said to be symmetrizable under input distribution $P_{\mathbf{x}} \in \Gamma$ if there exists a channel $U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}$ such that for all x, x', y,

$$\sum_{z,s} P_{\mathbf{x}}(x) W_{\mathbf{z}|\mathbf{x}}(z|x) U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}(x', s|z) W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x, s)$$
$$= \sum_{z',s'} P_{\mathbf{x}}(x') W_{\mathbf{z}|\mathbf{x}}(z'|x') U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}(x, s'|z') W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x', s'),$$

Symmetrizability for myopic AVCs

A myopic AVC is said to be symmetrizable under input distribution $P_{\mathbf{x}} \in \Gamma$ if there exists a channel $U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}$ such that for all x, x', y,

$$\sum_{z,s} P_{\mathbf{x}}(x) W_{\mathbf{z}|\mathbf{x}}(z|x) U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}(x', s|z) W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x, s)$$
$$= \sum_{z',s'} P_{\mathbf{x}}(x') W_{\mathbf{z}|\mathbf{x}}(z'|x') U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}(x, s'|z') W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x', s'),$$

and the resulting state distribution given by

$$P_{\mathbf{s}}(s) = \sum_{x,z,x'} P_{\mathbf{x}}(x) W_{\mathbf{z}|\mathbf{x}}(z|x) U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}(x', s|z)$$

belongs to Λ.

Symmetrizability for myopic AVCs

A myopic AVC is said to be symmetrizable under input distribution $P_{\mathbf{x}} \in \Gamma$ if there exists a channel $U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}$ such that for all x, x', y,

$$\sum_{z,s} P_{\mathbf{x}}(x) W_{\mathbf{z}|\mathbf{x}}(z|x) U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}(x', s|z) W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x, s)$$
$$= \sum_{z',s'} P_{\mathbf{x}}(x') W_{\mathbf{z}|\mathbf{x}}(z'|x') U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}(x, s'|z') W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x', s'),$$

and the resulting state distribution given by

$$P_{\mathbf{s}}(s) = \sum_{x,z,x'} P_{\mathbf{x}}(x) W_{\mathbf{z}|\mathbf{x}}(z|x) U_{\mathbf{x}',\mathbf{s}|\mathbf{z}}(x', s|z)$$

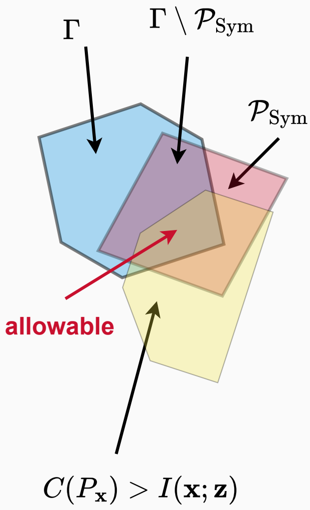
belongs to Λ. Let

$$\mathcal{P}_{\mathrm{Sym}} = \{P_{\mathbf{x}} \in \Gamma : P_{\mathbf{x}} \text{ is symmetrizable}\}.$$

Sufficient myopia and achievability

James can create an “effective DMC”

$$\mathcal{W} = \sum_{s} W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s) W_{\mathbf{z}|\mathbf{x}} V_{\mathbf{s}|\mathbf{z}}(s|z).$$



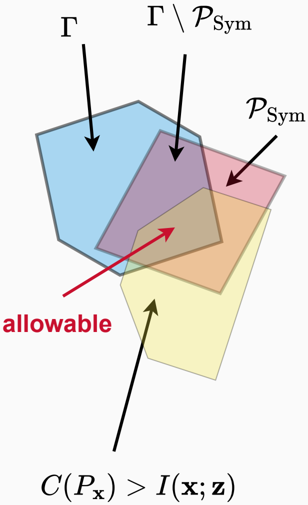
Sufficient myopia and achievability

James can create an "effective DMC"

$$\mathcal{W} = \sum_{s} W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s) W_{\mathbf{z}|\mathbf{x}} V_{\mathbf{s}|\mathbf{z}}(s|z).$$

Alice/Bob cannot use any $P_{\mathbf{x}} \in \mathcal{P}_{\mathrm{Sym}}$. For $P_{\mathbf{x}} \in \Gamma \setminus \mathcal{P}_{\mathrm{Sym}}$ they could target:

$$C(P_{\mathbf{x}}) = \min_{\mathcal{W}} I(\mathbf{x};\mathbf{y}).$$



Sufficient myopia and achievability

James can create an "effective DMC"

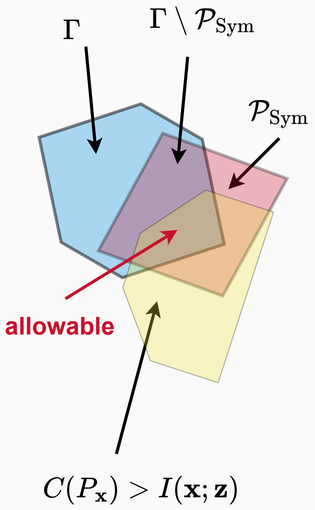
$$\mathcal{W} = \sum_{s} W_{\mathbf{y}|\mathbf{x},\mathbf{s}}(y|x,s) W_{\mathbf{z}|\mathbf{x}} V_{\mathbf{s}|\mathbf{z}}(s|z).$$

Alice/Bob cannot use any $P_{\mathbf{x}} \in \mathcal{P}_{\text{Sym}}$. For $P_{\mathbf{x}} \in \Gamma \setminus \mathcal{P}_{\text{Sym}}$ they could target:

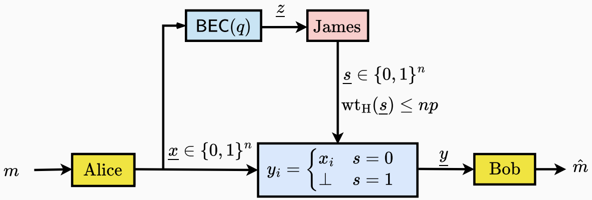
$$C(P_{\mathbf{x}}) = \min_{\mathcal{W}} I(\mathbf{x};\mathbf{y}).$$

If $I(\mathbf{z};\mathbf{x}) < C(P_{\mathbf{x}})$ we say James is **sufficiently myopic**. In that case we can achieve any rate

$$R < \max_{P_{\mathbf{x}} \in \Gamma \setminus \mathcal{P}_{\text{Sym}}} C(P_{\mathbf{x}}).$$



Myopic adversaries in the erasure setting

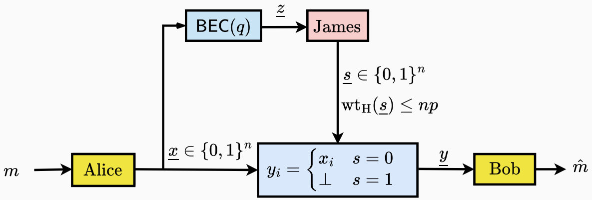


In the erasure setting the eavesdropping channel is a BEC(q) and James can erase at most pn bits. If $p < q$, James is **sufficiently myopic**.

Myopic adversaries in the erasure setting

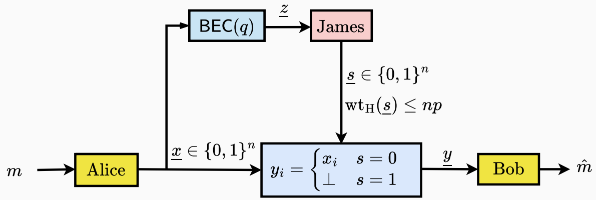
If $p < q$,

$C_{\text{obl}} = 1 - p$.



In the erasure setting the eavesdropping channel is a BEC(q) and James can erase at most pn bits. If $p < q$, James is **sufficiently myopic**.

Myopic adversaries in the erasure setting



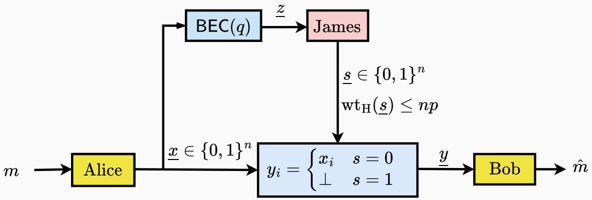
In the erasure setting the eavesdropping channel is a BEC(q) and James can erase at most pn bits. If $p < q$, James is **sufficiently myopic**.

If $p < q$,

$$C_{\mathrm{obl}} = 1 - p.$$

If $p > q$ we have two cases:

Myopic adversaries in the erasure setting



In the erasure setting the eavesdropping channel is a BEC(q) and James can erase at most pn bits. If $p < q$, James is **sufficiently myopic**.

If $p < q$,

$$C_{\mathrm{obl}} = 1 - p.$$

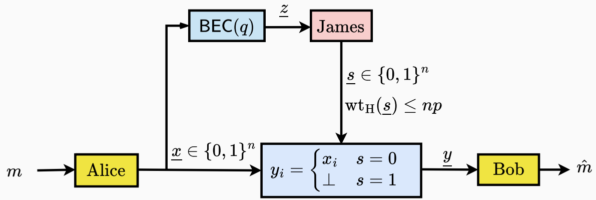
If $p > q$ we have two cases:

1. If $q > 2p - 1$,

$$C \in \left(0, (1-q)\bar{\alpha}\left(\frac{p-q}{1-q}\right)\right],$$

where $\bar{\alpha}$ is the LP bound for normalized distance.

Myopic adversaries in the erasure setting



In the erasure setting the eavesdropping channel is a BEC(q) and James can erase at most pn bits. If $p < q$, James is **sufficiently myopic**.

If $p < q$,

$$C_{\mathrm{obl}} = 1 - p.$$

If $p > q$ we have two cases:

1. If $q > 2p - 1$,

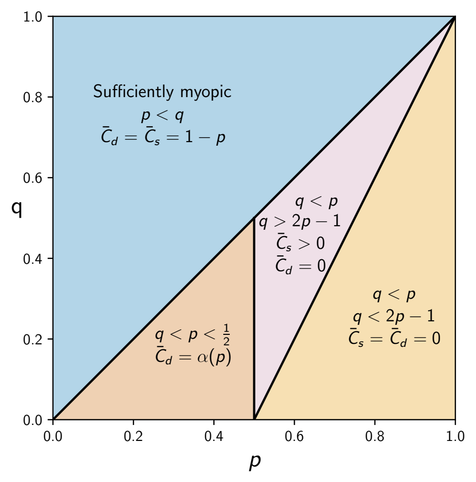
$$C \in \left(0, (1-q)\bar{\alpha}\left(\frac{p-q}{1-q}\right)\right],$$

   where $\bar{\alpha}$ is the LP bound for normalized distance.

2. If $q < 2p - 1$,

$$C = 0.$$

Myopic adversaries in the erasure setting



Computationally efficient codes for causal adversaries

Main questions we address

Can we design **efficient codes for causal and myopic models?**

By **efficient** we mean that they take **polynomial time** to encode, decode, and store.

Main questions we address

Can we design **efficient codes for causal and myopic models?**

By **efficient** we mean that they take **polynomial time** to encode, decode, and store.

- **random codes** are inefficient to decode but **linear codes** are too easy jam!
  $\longrightarrow$ use a **library of linear codebooks**.

Main questions we address

Can we design **efficient codes for causal and myopic models?**

By **efficient** we mean that they take **polynomial time** to encode, decode, and store.

- **random codes** are inefficient to decode but **linear codes** are too easy jam!
  $\longrightarrow$ use a **library of linear codebooks**.
- **common randomness** is unrealistic.
  $\longrightarrow$ use **limited encoder randomization** to **confuse the adversary**.

Main questions we address

Can we design **efficient codes for causal and myopic models?**

By **efficient** we mean that they take **polynomial time** to encode, decode, and store.

- **random codes** are inefficient to decode but **linear codes** are too easy jam!
  $\longrightarrow$ use a **library of linear codebooks**.
- **common randomness** is unrealistic.
  $\longrightarrow$ use **limited encoder randomization** to **confuse the adversary**.
- **minimum distance coding** is not efficient in general.
  $\longrightarrow$ use **list decoding** to permit **efficient decoding**.

“Efficient” coding schemes

To get **polynomial complexity**, use

- **a small amount of randomization** to select from a

“Efficient” coding schemes

To get **polynomial complexity**, use

- **a small amount of randomization** to select from a
- **library of random linear codes** and

“Efficient” coding schemes

To get **polynomial complexity**, use

- **a small amount of randomization** to select from a
- **library of random linear codes** and
- uses **list decoding** to reduce the search space

"Efficient" coding schemes

To get **polynomial complexity**, use

- **a small amount of randomization** to select from a
- **library of random linear codes** and
- uses **list decoding** to reduce the search space

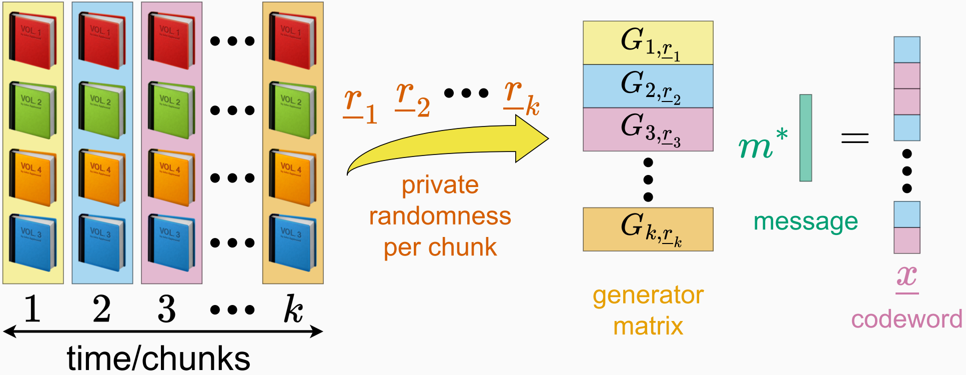
There are different types of complexity we would like to control:

- **Design**: how many bits do we need to generate the code?
- **Storage**: how many bits do we need to store the code?
- **Encoding**: how many operations are needed to encode a message?
- **Decoding**: how many operations are needed to decode the message?

Main results

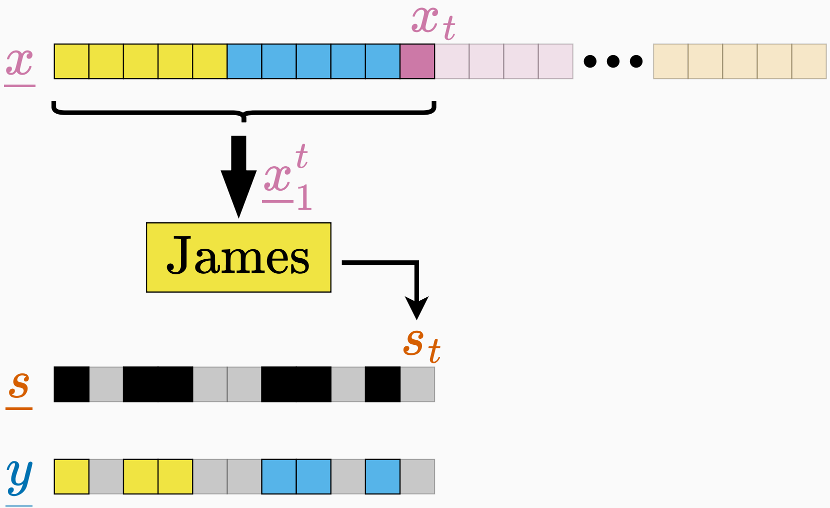
Model **rate**	**Randomness**	**Enc/Storage**	**Decoding**	$\mathbf{P}_{\text{error}}$
Myopic $p < q$ $\mathbf{1-p-\epsilon}$	$\lambda_{SM}\log(n)$	$O(n^{2+\lambda_{SM}})$	$O(n^{3+\lambda_{SM}})$	$O(n^{-\lambda_{SM}})$
Myopic $q < p$ **small rate**	$O(n\log\log n)$	$O(n^2\log\log n)$	$O(n^3\log\log n)$	$O(n^{-4/5})$
Causal $\mathbf{1-2p-\epsilon}$	$O\left(\frac{\gamma\log n}{\epsilon}\right)$	$O(n^3\log\log n)$	$O(n^{32/\epsilon})$	$O(n^{-(\gamma-1)})$

Encode splits block into a constant $k = \lceil \frac{n}{\epsilon} \rceil$ chunks

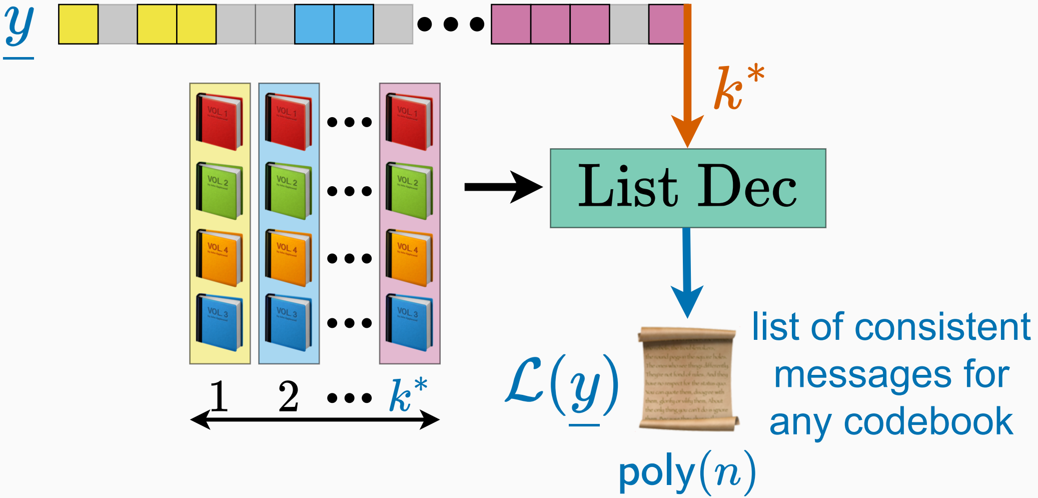


Generate a **library of linear codebooks** independently for each chunk.

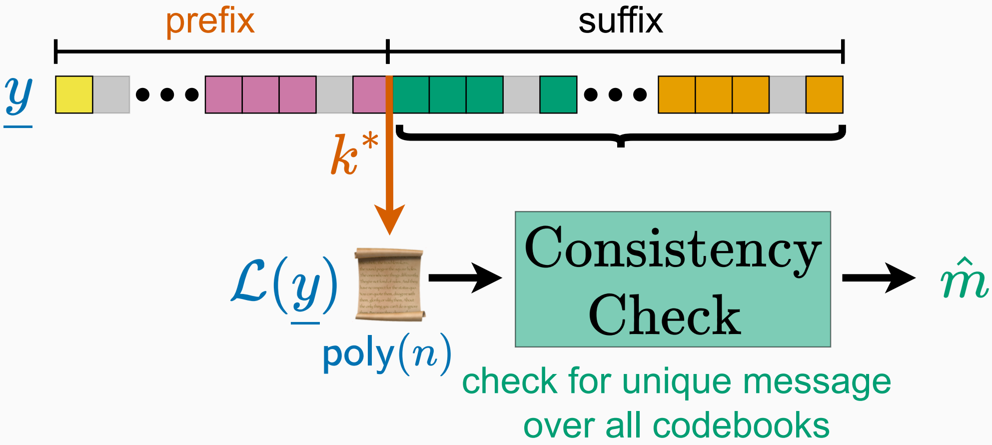
James can erase with causal information only



Bob decodes to a polynomial list



Bob uses suffix to disambiguate the list

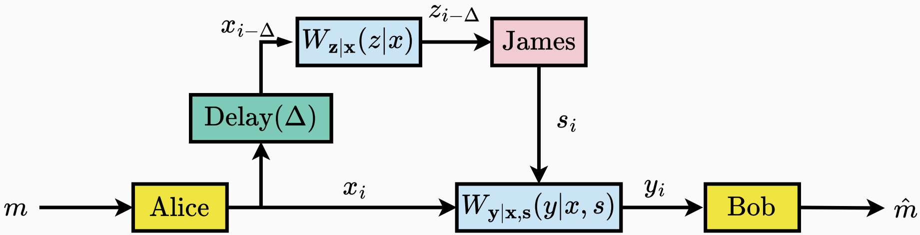


Why does this work?

1. Bob can **track James's erasure budget**.
2. List decoding creates **a smaller set of messages** to check for consistency.
3. James has a choice to **make the list larger** (erase more earlier, less later) or **conserve his budget** (erase less earlier, more later).
4. **Poor James, he can't win.**

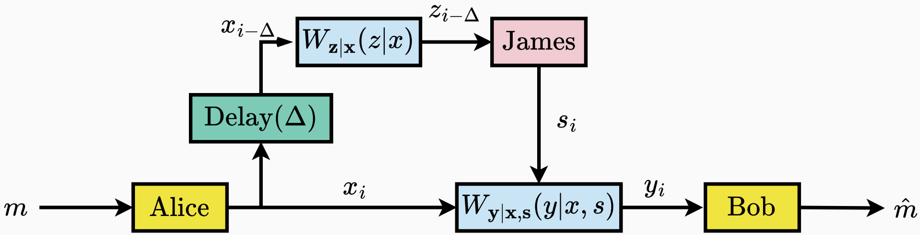
Looking forward

Other intermediate models



There are lots of other **intermediate models** one could look at:

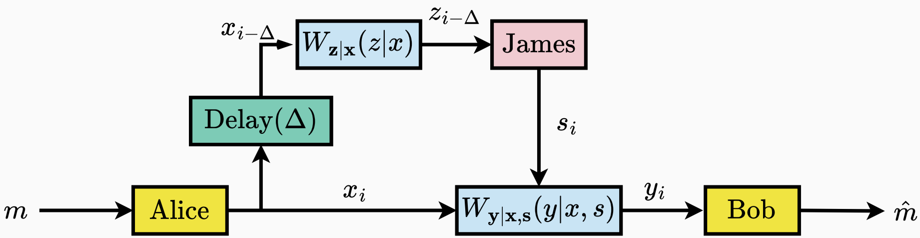
Other intermediate models



There are lots of other **intermediate models** one could look at:

- Causal and myopic together!

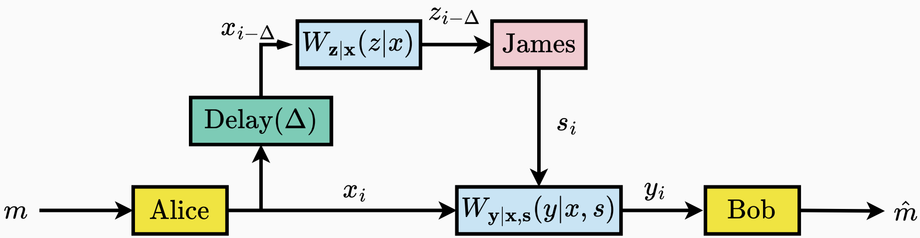
Other intermediate models



There are lots of other **intermediate models** one could look at:

- Causal and myopic together!
- Constraints that apply locally (sliding windows)

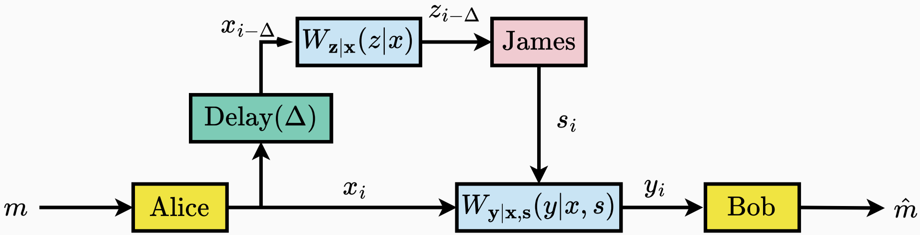
Other intermediate models



There are lots of other **intermediate models** one could look at:

- Causal and myopic together!
- Constraints that apply locally (sliding windows)
- Allow James to pick a fraction of locations to observe before acting.

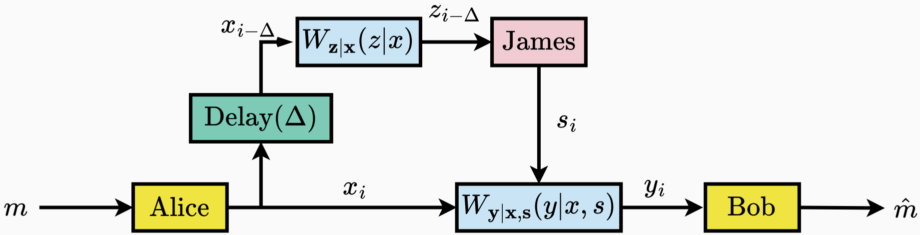
Other intermediate models



There are lots of other **intermediate models** one could look at:

- Causal and myopic together!
- Constraints that apply locally (sliding windows)
- Allow James to pick a fraction of locations to observe before acting.
- Etc. etc.

Other intermediate models



There are lots of other **intermediate models** one could look at:

- Causal and myopic together!
- Constraints that apply locally (sliding windows)
- Allow James to pick a fraction of locations to observe before acting.
- Etc. etc.

Each model will reveal something about what the **worst-case channel** looks like.

And for the theory folks...

Understanding AVCs has lots of connections (perhaps less well described here) to many interesting areas:

- zero-error capacity

And for the theory folks...

Understanding AVCs has lots of connections (perhaps less well described here) to many interesting areas:

- zero-error capacity
- high dimensional geometry

And for the theory folks...

Understanding AVCs has lots of connections (perhaps less well described here) to many interesting areas:

- zero-error capacity
- high dimensional geometry
- completely positive tensors and mixture models

And for the theory folks...

Understanding AVCs has lots of connections (perhaps less well described here) to many interesting areas:

- zero-error capacity
- high dimensional geometry
- completely positive tensors and mixture models
- adversarial machine learning

And for the theory folks...

Understanding AVCs has lots of connections (perhaps less well described here) to many interesting areas:

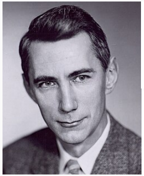
- zero-error capacity
- high dimensional geometry
- completely positive tensors and mixture models
- adversarial machine learning
- extremal graph theory

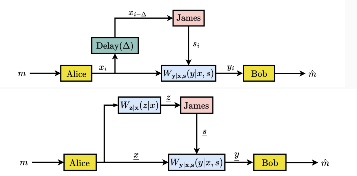
And for the theory folks...

Understanding AVCs has lots of connections (perhaps less well described here) to many interesting areas:

- zero-error capacity
- high dimensional geometry
- completely positive tensors and mixture models
- adversarial machine learning
- extremal graph theory
- other fun combinatorial problems

A final recap and takeaways





Proposed **arbitrarily varying channels** to explore the difference between average and worst-case channels.

- When James has to act causally, the capacity depends crucially on **what he knows about the current input**.
- When James has to act myopically, it depends on whether he **"decode"** or not: this creates many connections with the wiretap channel.

For emerging networked systems random noise models may be **too optimistic** and completely adversarial models may be **too pessimistic**. Strategies like **stochastic encoding** and **list decoding** can help!

Thank you!