Sparse Regression Codes

Andrew Barron
Yale University

Ramji Venkataramanan
University of Cambridge

Joint work with Antony Joseph, Sanghee Cho, Cynthia Rush, Adam Greig, Tuhin Sarkar, Sekhar Tatikonda

ISIT 2016

Outline of Tutorial

Sparse Superposition Codes or Sparse Regression Codes (SPARCs) for:

1. Provably practical and reliable communication over the AWGN channel at rates approaching capacity

2. Efficient lossy compression at rates approaching Shannon limit

3. Multi-terminal communication and compression models

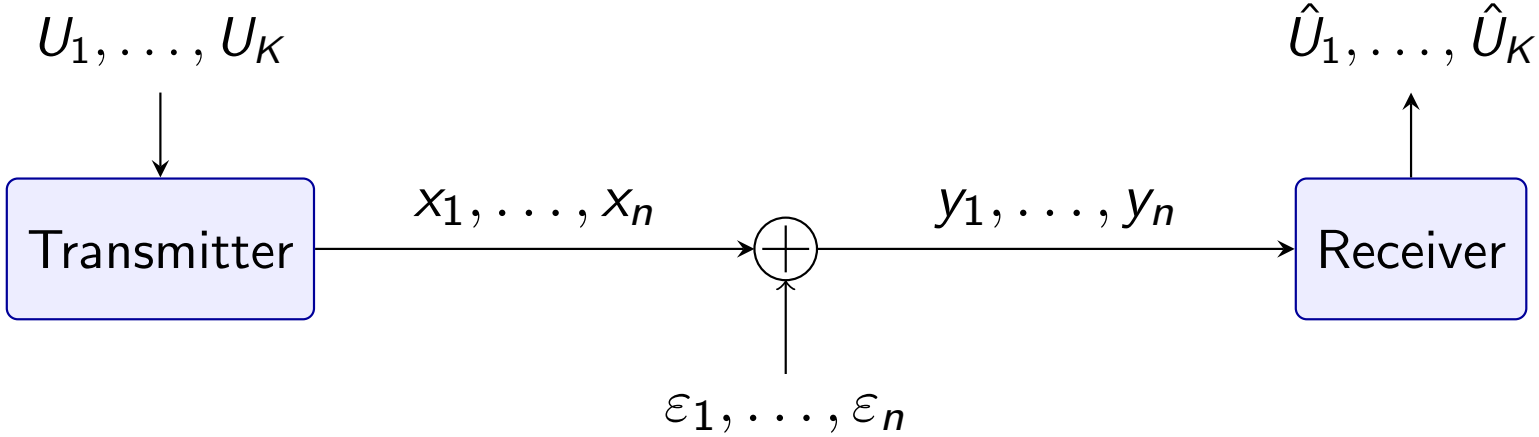
4. Open Questions

Part I: Communication over the AWGN Channel

Quest for Provably Practical and Reliable High Rate Communication

- The Channel Communication Problem
- Gaussian Channel
- History of Methods
- Sparse Superposition Coding
- Three efficient decoders:
  1. Adaptive successive threshold decoder
  2. Adaptive successive soft-decision decoder
  3. Approximate Message Passing (AMP) decoder
- Rate, Reliability, and Computational Complexity
- Distributional Analysis
- Simulations

The Additive White Gaussian Noise Channel



For $i = 1, \ldots n, \quad y_i = x_i + \varepsilon_i$ with:

- Average power constraint: $\frac{1}{n}\sum_i x_i^2 \leq P$
- Additive Gaussian noise: ε_i iid $\sim \mathcal{N}(0, \sigma^2)$
- Rate: $R = \frac{K}{n}$
- Capacity: $\mathcal{C} = \frac{1}{2}\log(1 + snr)$
- Reliability: Want small Prob$\{\hat{U} \neq U\}$ or reliably small fraction of errors for R approaching $\mathcal{C}$

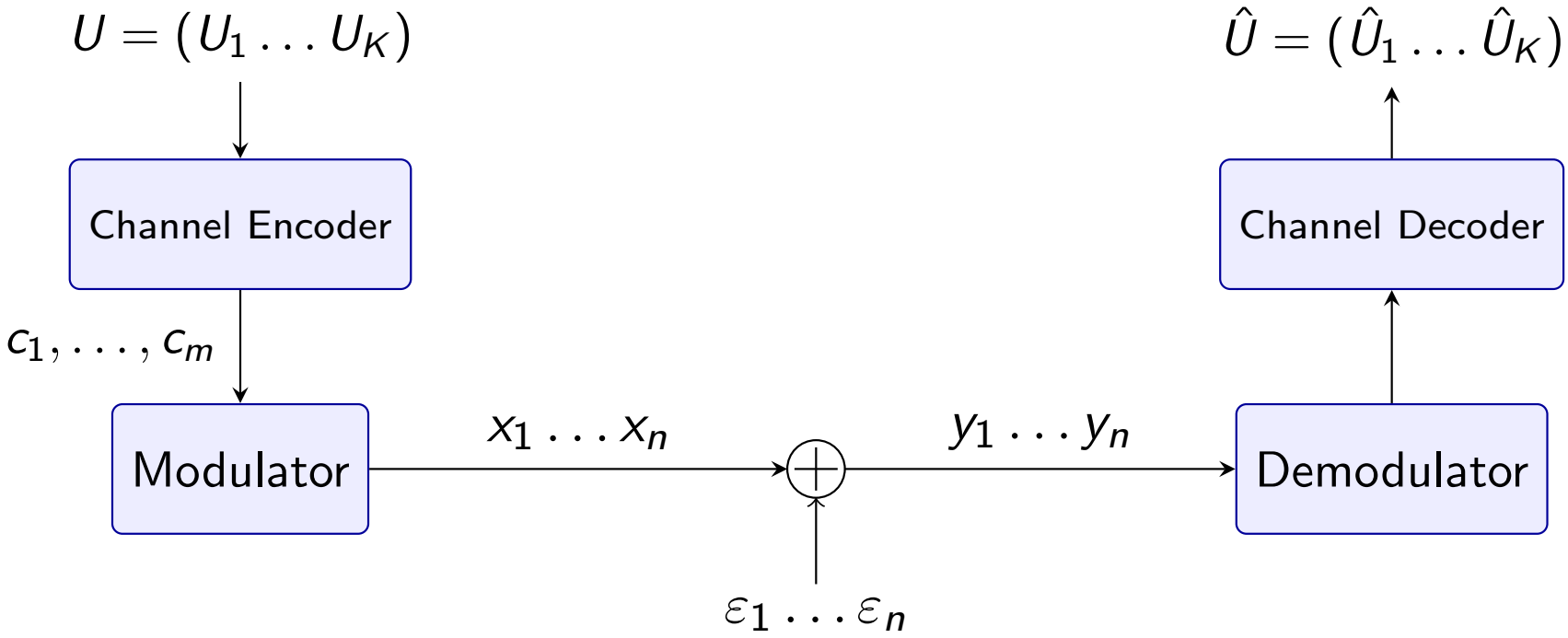
Capacity-achieving codes

For many *binary/discrete* alphabet channels:

- Turbo and sparse-graph (LDPC) codes achieve rates close to capacity with efficient message-passing decoding
- Theoretical results for spatially-coupled LDPC codes [Kudekar, Richardson, Urbanke '12, '13], ...
- Polar codes achieve capacity with efficient decoding [Arikan '09], [Arikan, Telatar], ...

But we want to achieve $\mathcal{C}$ for the AWGN channel. Let's look at some existing approaches ...

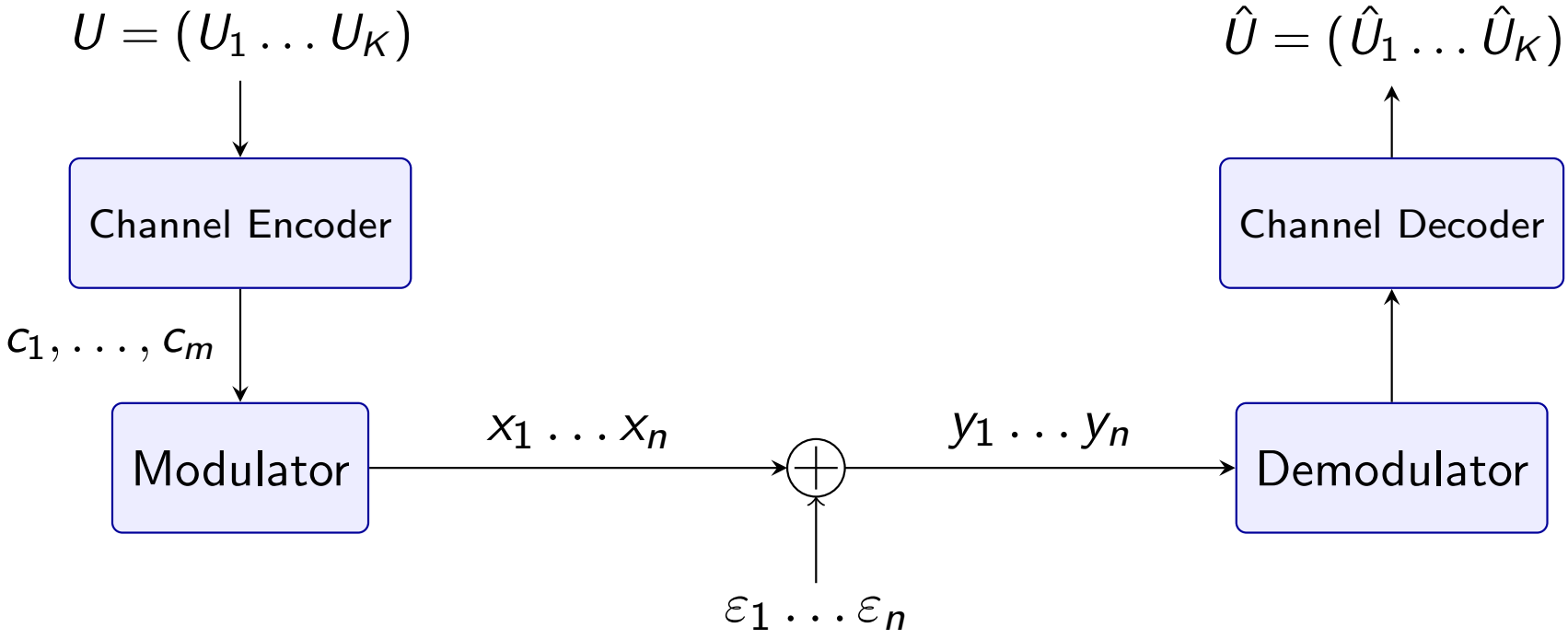
Existing Approaches: Coded Modulation



1. Fix a modulation scheme, e.g, 16-QAM, 64-QAM
2. Use a powerful binary code (e.g., LDPC, turbo code) to protect against errors
3. Channel decoder uses soft outputs from demodulator

Surveys:[Ungerboeck, Forney'98], [Guillen i Fabregas, Martinez, Caire'08]

Existing Approaches: Coded Modulation



Coded modulation works well in practice, but cannot provably achieve capacity with fixed constellation

Existing Approaches: Lattice Coding

Analog of linear codes in Euclidean space; provide coding and shaping gain

- *Achieving* $\frac{1}{2}\log(1+snr)$ *on the AWGN channel with lattice encoding and decoding*, [Erez, Zamir '08]
- *Low-Density Lattice Codes*, [Sommer-Feder-Shalvi '08]
- *Polar Lattices*, [Yan, Liu, Ling, Wu '14]

$\vdots$

Sparse Regression Codes (SPARC)

In this part of the tutorial, we discuss the basic Sparse Regression Code construction with power allocation + two feasible decoders

References for this part:

– A. Joseph and A. R. Barron, *Least-squares superposition codes of moderate dictionary are reliable at rates up to capacity,* IEEE Trans. Inf. Theory, May 2012

– A. Joseph and A. R. Barron, *Fast sparse superposition codes have near exponential error probability for $R < \mathcal{C}$*, IEEE Trans. Inf. Theory, Feb. 2014

– A. R. Barron and S. Cho, *High-rate sparse superposition codes with iteratively optimal estimates*, ISIT 2012

– A. R. Barron and S. Cho, *Approximate Iterative Bayes Optimal Estimates for High-Rate Sparse Superposition Codes*, WITMSE 2013

– S. Cho, *High-dimensional regression with random design, including sparse superposition codes*, PhD thesis, Yale University, 2014

Extensions and Generalizations of SPARCs

Spatially-coupled dictionaries for SPARC:

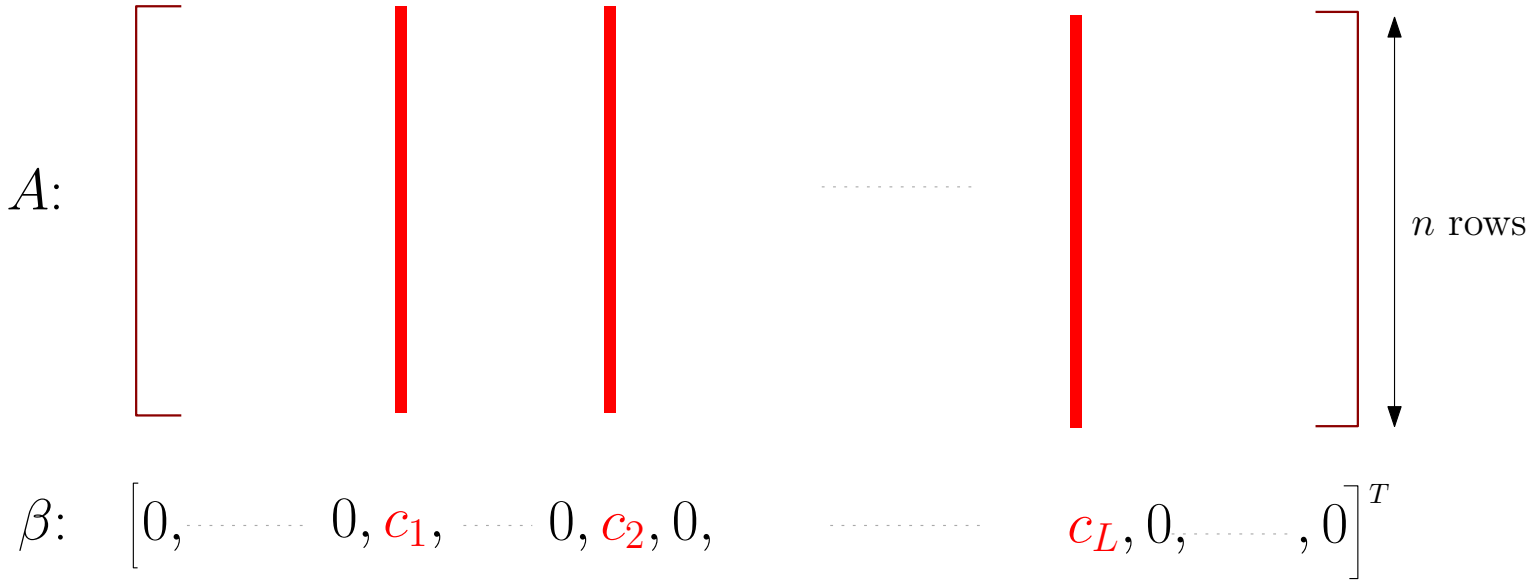
J. Barbier, C. Schülke, F. Krzakala, *Approximate message-passing with spatially coupled structured operators, with applications to compressed sensing and sparse superposition codes*, J. Stat. Mech, 2015
`http://arxiv.org/abs/1503.08040`

Bernoulli ± 1 dictionaries:

Y. Takeishi, M. Kawakita, and J. Takeuchi. *Least squares superposition codes with bernoulli dictionary are still reliable at rates up to capacity* , IEEE Trans. Inf. Theory, May 2014

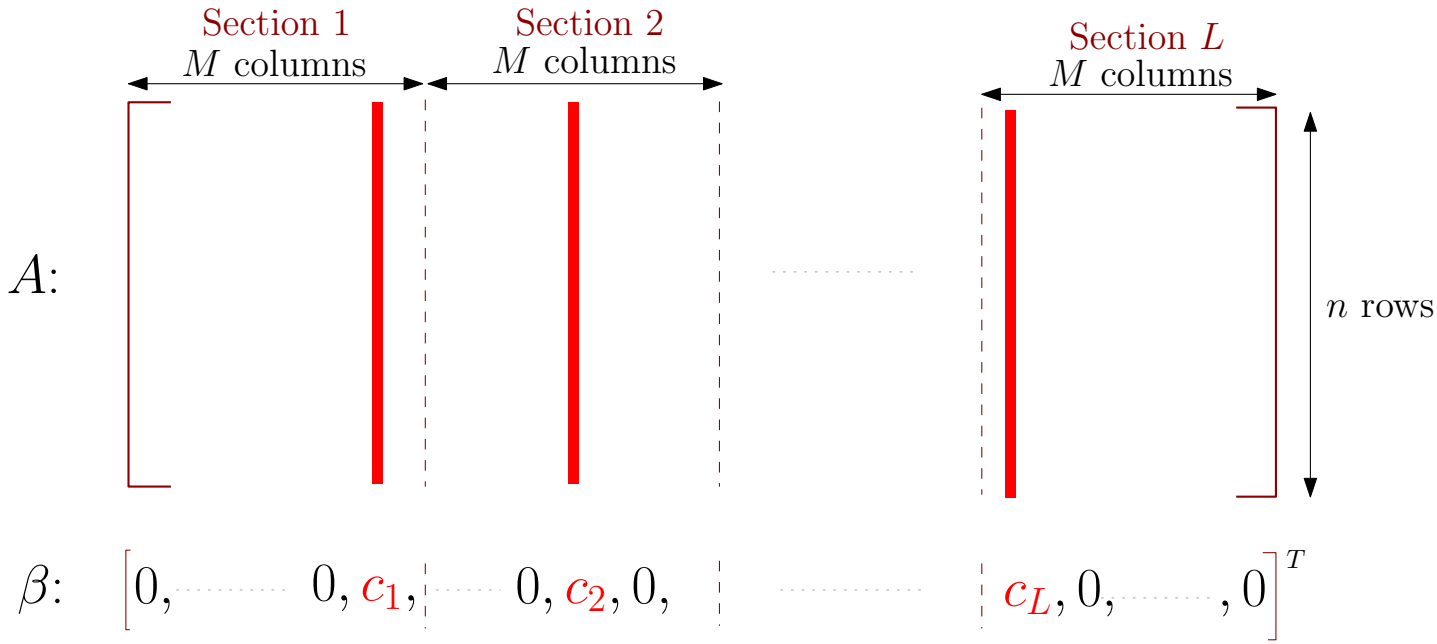
Tuesday afternoon session on Sparse Superposition Codes

Sparse Regression Code



- A: $n \times N$ design matrix with iid $\mathcal{N}(0, 1/n)$ entries
- Codewords $A\beta$: *sparse* linear combinations of columns of A with L out of N entries non-zero
- Message bits $U = (U_1, \ldots, U_K)$ determine the *locations* of the L non-zeros; values of non-zeros $c_1, \ldots, c_L$ are fixed a priori
- Blocklength of code $= n$; Rate $= K/n = \log \binom{N}{L}/n$
- Receiver gets $Y = A\beta + \varepsilon$; has to decode $\hat{\beta}, \hat{U}$ from Y, A

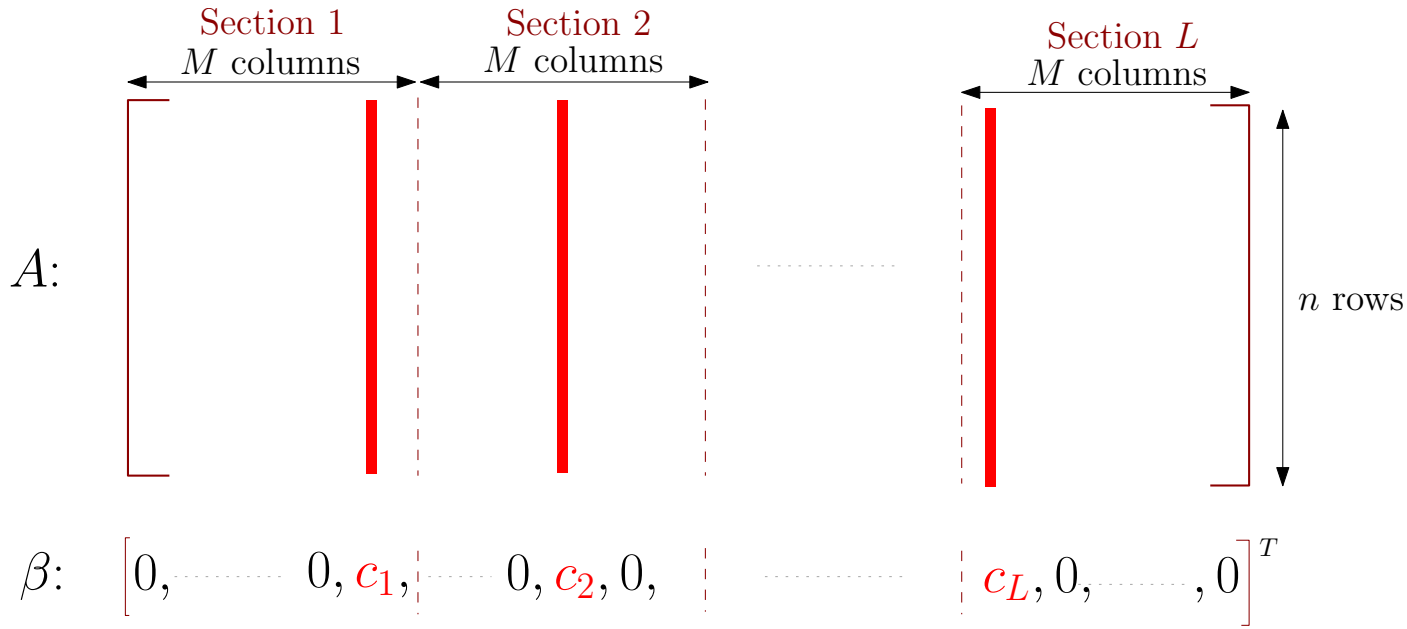
Partitioned Sparse Regression Code



Number of columns $N = ML$

- Matrix A split into L sections with M columns in each section
- β has exactly one non-zero in each section
- Total of M^L *codewords* $\Rightarrow$ Rate $R = \frac{\log M^L}{n} = \frac{L \log M}{n}$
- Input bits $U = (U_1, \ldots, U_K)$ split into L segments of $\log_2 M$ bits, with segment ℓ indexing location of non-zero in section ℓ
- Receiver gets $Y = A\beta + \varepsilon$; has to decode $\hat{\beta}$, $\hat{U}$ from Y, A

Choosing M, L

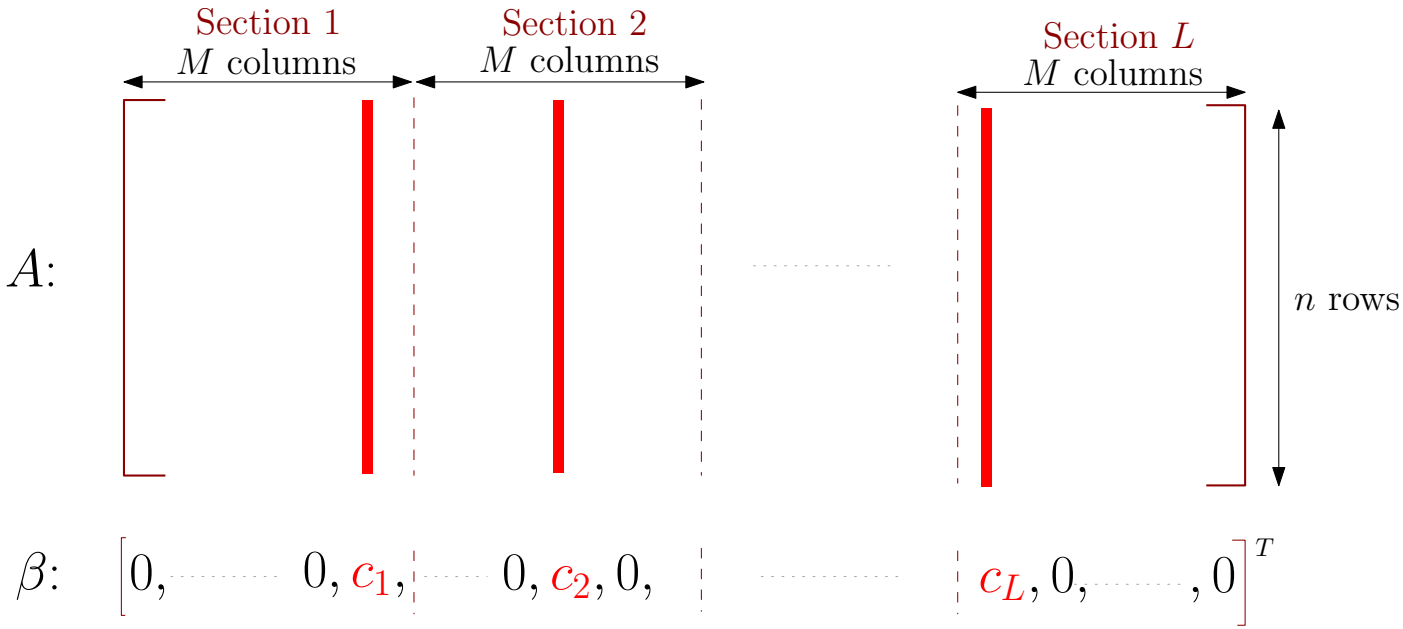


Block length n; Rate $R = (L \log M)/n$

Ultra-sparse case: Impractical $M = 2^{nR/L}$ with L constant

- Reliable at all rates $R < C$ [Cover 1972,1980]
- But size of A *exponential* in block length n

Choosing M, L

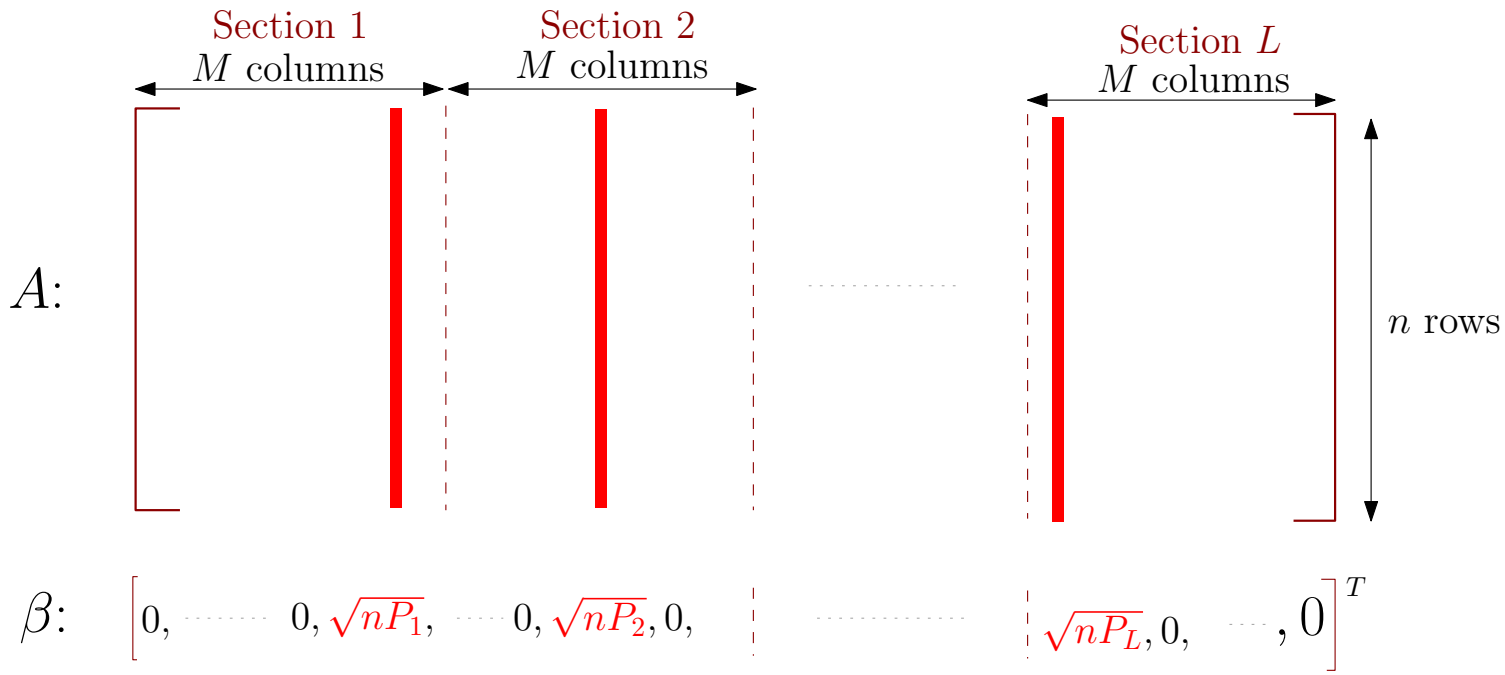


Block length n; Rate $R = (L \log M)/n$

Moderately-sparse: Practical $M = n^{\kappa}$ with $L = nR/\kappa \log n$

- size of A *polynomial* in block length n;
- Reliability: want small $\Pr\{\textit{Fraction section mistakes} \geq \epsilon\}$, for small ϵ
- Outer RS code: rate $1-\epsilon$, corrects remaining mistakes
- Overall rate: $R_{total} = (1-\epsilon)R$; *can achieve* R_{total} *up to* $\mathcal{C}$

Power Allocation



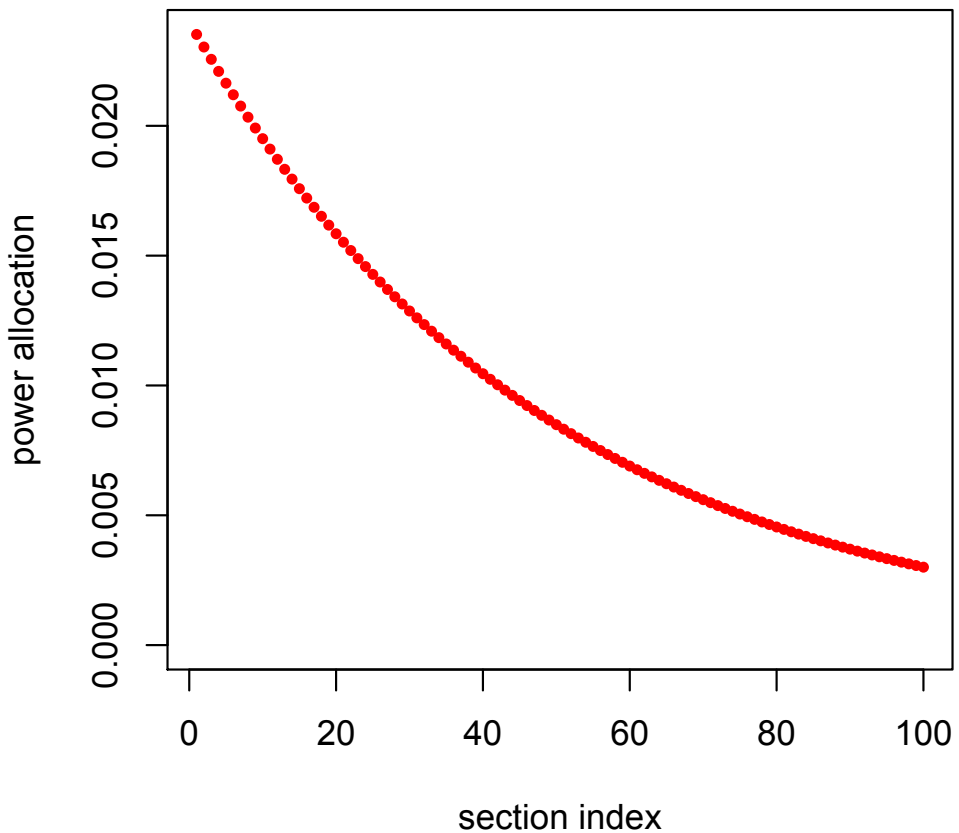
- Indices of nonzeros: $sent = (j_1, j_2, \ldots, j_L)$
- Coeff. values: $\beta_{j_1} = \sqrt{nP_1}$, $\beta_{j_2} = \sqrt{nP_2} \ldots \beta_{j_L} = \sqrt{nP_L}$
- Power Control: $\sum_\ell P_\ell = P$
  $\Rightarrow$ codewords $A\beta$ have average power P
- Examples: 1) Flat: $P_\ell = \frac{P}{L}$, $\ell = 1, \ldots, L$
  2) Exponentially decaying: $P_\ell \propto e^{-2C\ell/L}$

For all power allocations, $P_\ell = \Theta(\frac{1}{L})$, $\sqrt{nP_\ell} = \Theta(\sqrt{\log M})$ $\forall \ell$

Variable Power Allocation

- Power control: $\sum_{\ell=1}^{L} P_\ell = P \qquad\qquad \|\beta\|^2 = P$
- Variable power: P_ℓ proportional to $e^{-2\mathcal{C}\ell/L}$ for $\ell = 1, \ldots, L$

Example: P=7, $L = 100$



Variable Power Allocation

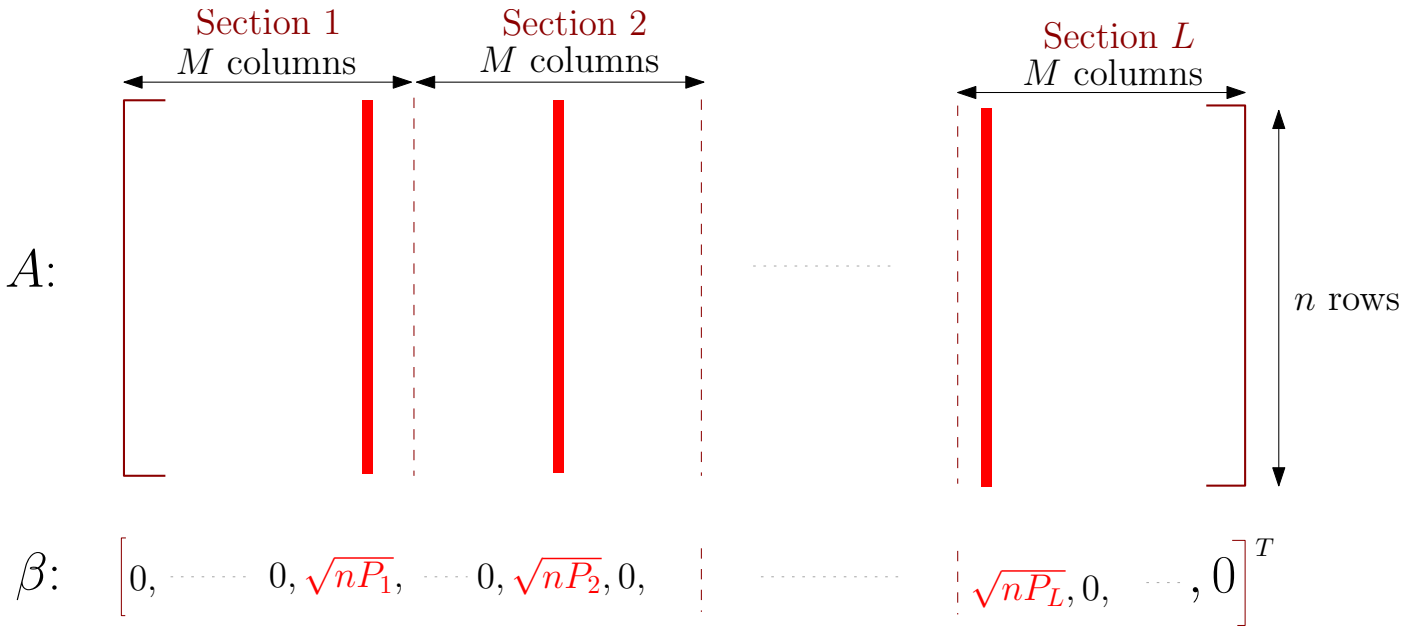
- Power control: $\sum_{\ell=1}^{L} P_\ell = P \qquad \|\beta\|^2 = P$
- For theoretical analysis, we use $P_\ell \propto e^{-2\mathcal{C}\ell/L}$ for $\ell = 1, \ldots, L$
- Successive decoding motivation
- Incremental capacity

$$\frac{1}{2}\log\left(1 + \frac{P_\ell}{\sigma^2 + P_{\ell+1} + \cdots + P_L}\right) = \frac{\mathcal{C}}{L}$$

matching the section rate

$$\frac{R}{L} = \frac{\log M}{n}$$

Decoding



GOAL: Recover *sent* terms in β, i.e., non-zero indices $j_1, \ldots, j_L$ from

$$Y = A\beta + \varepsilon$$

- Optimal decoder (ML): $\hat{\beta}_{ML} = \arg\min_{\hat{\beta}} \|Y - A\hat{\beta}\|^2$. But complexity exponential in n
- *Feasible* decoders: We will present three decoders, each of which iteratively produces estimates of β denoted $\beta^1, \beta^2, \ldots$

Adaptive Successive Threshold Decoder (Hard-Decision)

Given $Y = A\beta + \varepsilon$, start with estimate $\beta^0 = 0$

Start [Step 1] :

- Compute the inner product of $Y/|Y|$ with each column of A ($|Y| = \|Y\|/\sqrt{n}$)
- Pick the ones above a *threshold* $\sqrt{2 \log M} + a$ to form β^1
- Form initial fit as weighted sum of columns: $Fit_1 = A\beta^1$

Adaptive Successive Threshold Decoder (Hard-Decision)

Given $Y = A\beta + \varepsilon$, start with estimate $\beta^0 = 0$

Start [Step 1] :

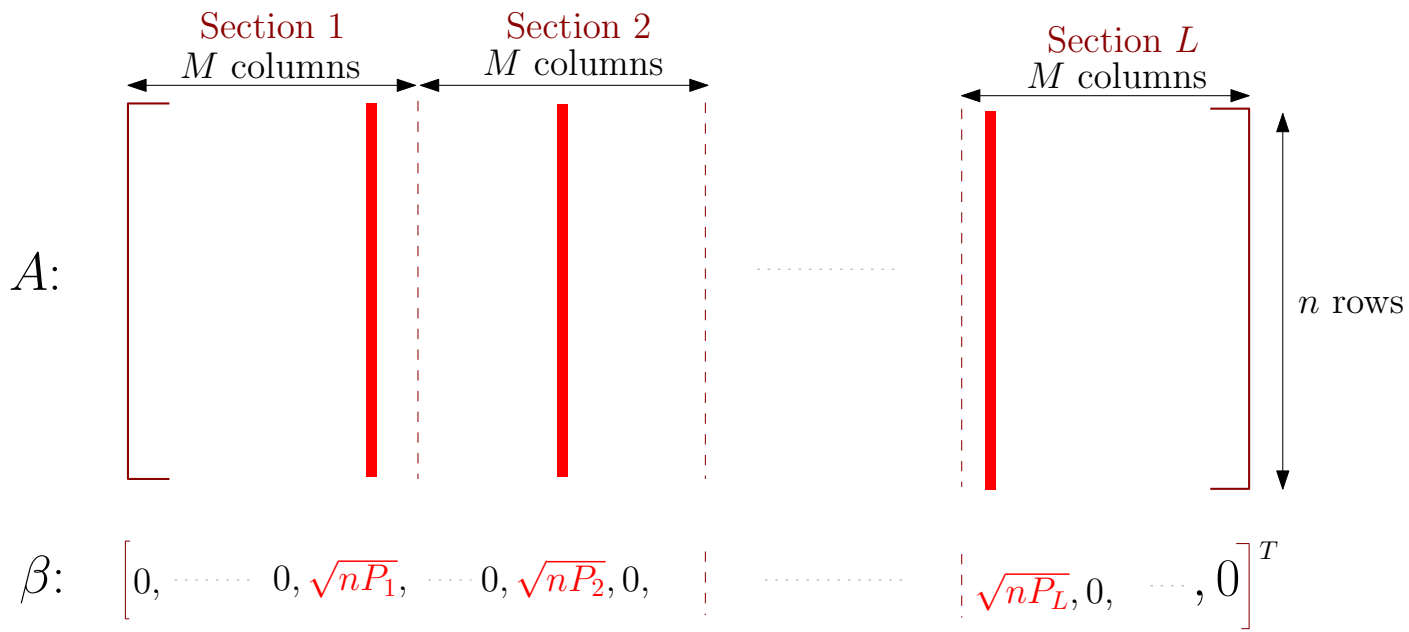
- Compute the inner product of $Y/|Y|$ with each column of A ($|Y| = \|Y\|/\sqrt{n}$)
- Pick the ones above a *threshold* $\sqrt{2\log M} + a$ to form β^1
- Form initial fit as weighted sum of columns: $Fit_1 = A\beta^1$

Iterate: [Step t, $t > 1$]

- Normalized residual $= (Y - Fit_{t-1})/|Y - Fit_{t-1}|$
- Compute the inner product of normalized residual with each *remaining* column of A
- Pick above threshold $\sqrt{2\log M} + a$ to form β^t
- $Fit_t = A\beta^t$

Stop: If there are no additional inner products above threshold, or after $snr \log M$ steps

Complexity of Adaptive Successive Threshold Decoder



Complexity in parallel pipelined implementation:

- Space: (use $T^* = snr \log M$ copies of the n by N dictionary)
  - $T^* nN = snr\, \mathcal{C}\, M\, n^2$ memory positions
  - $T^* N$ multiplier/accumulators and comparators
- Time: $O(1)$ per received Y symbol

Rate and Reliability

Result for Optimal ML Decoder [Joseph and Barron '12]: with outer RS decoder, and *flat* power allocation ($P_\ell = P/L, \forall \ell$)

Probability of error exponentially small in n for all $R < \mathcal{C}$:

$$\text{Prob}\{\textit{Error}\} \leq e^{-n(\mathcal{C}-R)^2/2V}$$

In agreement with the Shannon-Gallager exponent of optimal code, though with suboptimal constant V depending on *snr*

Performance of Adaptive Successive Threshold Decoder

Practical: Adaptive Successive Decoder, with outer RS code.

[Joseph-Barron]:

- Value $\mathcal{C}_{\mathcal{M}}$ approaching capacity:

$$\mathcal{C}_M = \mathcal{C}\left(1 - \frac{c_1}{\log M}\right)$$

Probability error exponentially small in L for $R < \mathcal{C}_M$

$$\text{Prob}\{\mathit{Error}\} \leq e^{-L(\mathcal{C}_M - R)^2 c_2}$$

- $L \sim n/(\log n)$
  $\Rightarrow$ Prob error exponentially small in $n/(\log n)$ for $R < \mathcal{C}$

Adaptive Successive Decoder (*Soft-Decision*)

Main idea: Instead of hard decision on sections that cross a threshold, *in each step update posterior probabilities of each column being the correct one in its section*

Start with $\beta^0 = 0$

Iterate: Assuming you have β^t, in step t compute:

- Inner product of *adjusted* residual with each column of A:

$$stat_{t,j} = \left(Y - A\beta^t_{-j}\right)^T A_j, \quad j = 1, \ldots, N$$

Adaptive Successive Decoder (*Soft-Decision*)

Main idea: Instead of hard decision on sections that cross a threshold, *in each step update posterior probabilities of each column being the correct one in its section*

Start with $\beta^0 = 0$

Iterate: Assuming you have β^t, in step t compute:

- Inner product of *adjusted* residual with each column of A:

$$stat_{t,j} = \left(Y - A\beta^t_{-j}\right)^T A_j, \quad j = 1, \ldots, N$$

We *want* $stat_{t,j}$ to be distributed as

$$stat_{t,j} = \beta_j + \tau_t Z_{t,j} = \underbrace{\sqrt{nP_\ell}\,\mathbf{1}_{\{j=j_\ell\}}}_{\beta_j} + \tau_t Z_{t,j}, \quad \text{for } j \in \text{section } \ell$$

  - j_ℓ is index of the true non-zero term in section $\ell \in \{1, \ldots, L\}$
  - $Z_{t,j}$ are iid $\mathcal{N}(0,1)$
  - $\sqrt{nP_\ell} = \Theta(\sqrt{\log M})$

Iteratively Bayes Optimal Estimates

The Bayes estimate based on $stat_t$ is $\beta^{t+1} = \mathbb{E}[\beta | stat_t]$ with:

- $stat_{t,j} = \sqrt{nP_\ell}\, \mathbf{1}_{\{j=j_\ell\}} + \tau_t\, Z_{t,j}, \quad$ for $j \in$ sec. ℓ
- Prior $j_\ell \sim$ *Uniform* on indices in sec. ℓ, $\quad Z_{t,j}$ iid $\sim \mathcal{N}(0,1)$

For $j \in$ section ℓ:

$$\beta_j^{t+1}(s) = \mathbb{E}[\beta_j | stat_t = s] = \sqrt{nP_\ell}\; Prob(j_\ell = j \mid stat_t = s)$$

$$= \sqrt{nP_\ell}\; \underbrace{\frac{\exp(\sqrt{nP_\ell}\, s_j/\tau_t^2)}{\sum_{k \in sec_\ell} \exp(\sqrt{nP_\ell}\, s_k/\tau_t^2)}}_{w_j(\tau_t)}$$

Weight $w_j(\tau_t)$ is the posterior probability (after step t) of column j being the correct one in its section

Desired Distribution

All of this is based on the *desired* distribution of $stat_t$:

$$stat_{t,j} = \underbrace{\sqrt{nP_\ell}\,\mathbf{1}_{\{j=j_\ell\}}}_{\beta_j} + \tau_t\, Z_{t,j}, \quad \text{for } j \in \text{section } \ell$$

What is the noise variance τ_t^2?

We can write

$$stat_{t,j} = (Y - A\beta^t_{-j})^T A_j = (Y - A\beta^t)^T A_j + \underbrace{\|A_j\|^2}_{\approx 1}\beta_j^t$$

Hence

$$stat_t = (Y - A\beta^t)^T A_j + \beta^t = \beta + \underbrace{A^T w}_{\mathcal{N}(0,\sigma^2)} + \underbrace{(\mathrm{I} - A^T A)}_{\approx \mathcal{N}(0,1/n)}(\beta^t - \beta)$$

If $(\beta - \beta^t)$ were independent of $(\mathrm{I} - A^T A)$, then ...

Noise Variance τ_t^2

If $(\beta - \beta^t)$ were independent of $(\mathrm{I} - A^T A)$, then:

$$stat_t = \beta + \tau_t Z_t$$

where

$$\tau_t^2 = \sigma^2 + \frac{1}{n}\mathbb{E}\,\|\beta - \beta^t\|^2$$

Noise Variance τ_t^2

If $(\beta - \beta^t)$ were independent of $(\mathrm{I} - A^T A)$, then:

$$stat_t = \beta + \tau_t Z_t$$

where

$$\tau_t^2 = \sigma^2 \; + \; \frac{1}{n}\mathbb{E}\,\|\beta - \beta^t\|^2$$

Assuming $stat_1, \ldots, stat_{t-1}$ are all distributed as desired, then

$$\frac{1}{n}\,\mathbb{E}\,\|\beta - \beta^t\|^2 = \frac{1}{n}\,\mathbb{E}\,\|\beta - \mathbb{E}[\beta|\beta + \tau_{t-1}Z_{t-1}]\|^2 = P(1 - x_t(\tau_{t-1}))$$

where

$$x_t(\tau_{t-1}) = \sum_{\ell=1}^{L} \frac{P_\ell}{P}\mathbb{E}\left[\frac{\exp\left(\frac{\sqrt{nP_\ell}}{\tau_{t-1}}\left(U_1^\ell + \frac{\sqrt{nP_\ell}}{\tau_{t-1}}\right)\right)}{\exp\left(\frac{\sqrt{nP_\ell}}{\tau_{t-1}}\left(U_1^\ell + \frac{\sqrt{nP_\ell}}{\tau_{t-1}}\right)\right) + \sum_{j=2}^{M}\exp\left(\frac{\sqrt{nP_\ell}}{\tau_{t-1}}U_j^\ell\right)}\right]$$

$\{U_j^\ell\}$ are i.i.d. $\sim \mathcal{N}(0,1)$

Iteratively compute variances τ_t^2

$$\tau_0^2 = \sigma^2 + P$$
$$\tau_t^2 = \sigma^2 + P(1 - x_t(\tau_{t-1})), \quad t \geq 1$$

where

$$x_t(\tau_{t-1}) = \sum_{\ell=1}^{L} \frac{P_\ell}{P} \mathbb{E}\left[\underbrace{\frac{\exp\left(\frac{\sqrt{nP_\ell}}{\tau_{t-1}}\left(U_1^\ell + \frac{\sqrt{nP_\ell}}{\tau_{t-1}}\right)\right)}{\exp\left(\frac{\sqrt{nP_\ell}}{\tau_{t-1}}\left(U_1^\ell + \frac{\sqrt{nP_\ell}}{\tau_{t-1}}\right)\right) + \sum_{j=2}^{M} \exp\left(\frac{\sqrt{nP_\ell}}{\tau_{t-1}} U_j^\ell\right)}}_{w_{j_\ell}(\tau_{t-1})} \right]$$

With $stat_t = \beta + \tau_t Z$:

$\frac{1}{n}\mathbb{E}\|\beta - \beta^t\|^2 = P(1 - x_t)$ and $\frac{1}{n}\mathbb{E}[\beta^T \beta^t] = \frac{1}{n}\mathbb{E}\|\beta^t\|^2 = P x_t$

- x_t: Expected power-weighted fraction of *correctly decoded* sections after step t
- $P(1 - x_t)$: interference due to *undecoded* sections

Update Rule for Success Rate x_t

1) Update rule $x_t = g(x_{t-1})$ where

$$g(x) = \sum_{\ell=1}^{L} \frac{P_\ell}{P} \mathbb{E}\left[\underbrace{\frac{\exp\left(\frac{\sqrt{nP_\ell}}{\tau}\left(U_1^\ell + \frac{\sqrt{nP_\ell}}{\tau}\right)\right)}{\exp\left(\frac{\sqrt{nP_\ell}}{\tau}\left(U_1^\ell + \frac{\sqrt{nP_\ell}}{\tau}\right)\right) + \sum_{j=2}^{M} \exp\left(\frac{\sqrt{nP_\ell}}{\tau} U_j^\ell\right)}}_{w_{j_\ell}(\tau)} \right]$$

with $\tau^2 = \sigma^2 + P(1-x)$.

This is the success rate update function expressed as a *power-weighted sum of the posterior prob of the term sent*

Under the assumed distribution of $stat_t$:

2) The true empirical success rate is $x_t^* = \frac{1}{nP} \beta^T \beta^t$

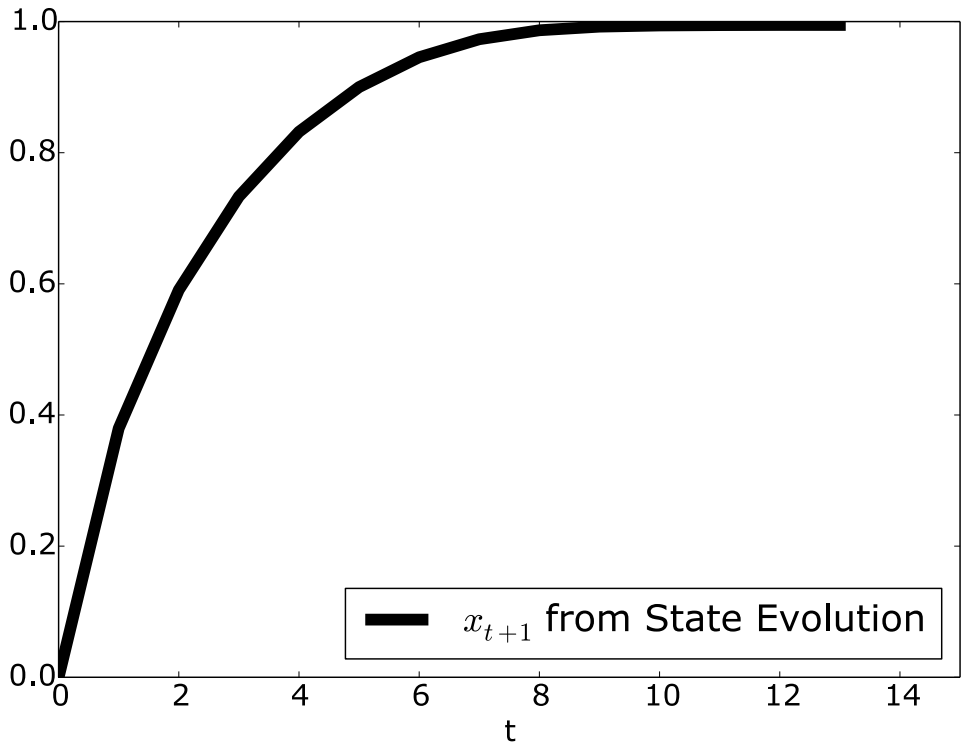
3) The decoder could also compute the empirical *estimated* success rate $\hat{x}_t = \frac{1}{nP} \|\beta^t\|^2$

x_t vs t

- With $\tau_t^2 = \sigma^2 + P(1 - x_t)$, we have iteratively computed $x_0 = 0,\ x_1, \ldots, x_t, \ldots$
- We want:

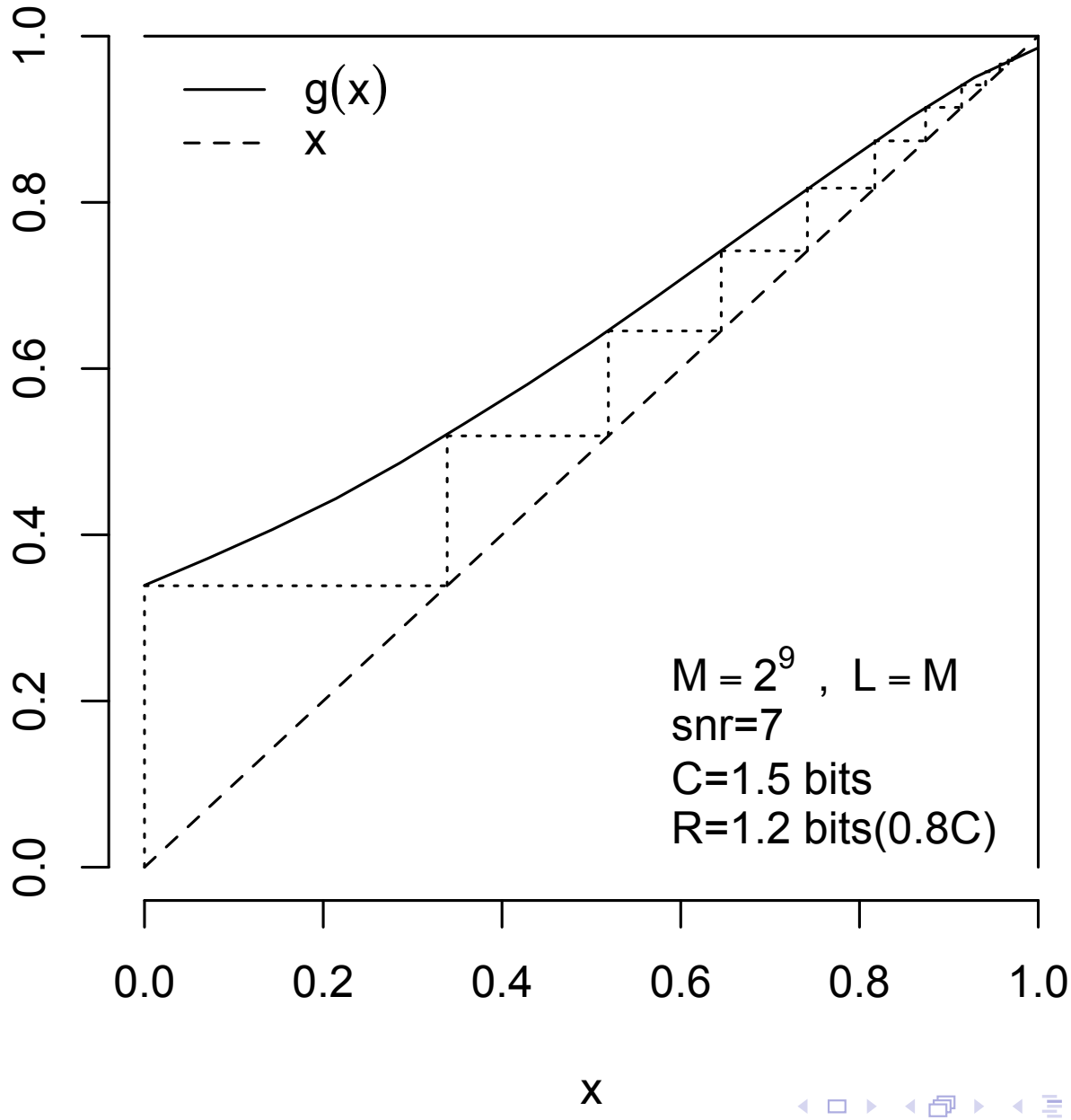
  x_t to $\nearrow$ monotonically with t up to a value close to 1 $\Leftrightarrow$
  τ_t^2 to $\searrow$ monotonically down to a value close to σ^2

SPARC: $M = 512, L = 1024, \text{snr} = 15, R = 0.7\mathcal{C}, P_\ell \propto 2^{-2\mathcal{C}\ell/L}$



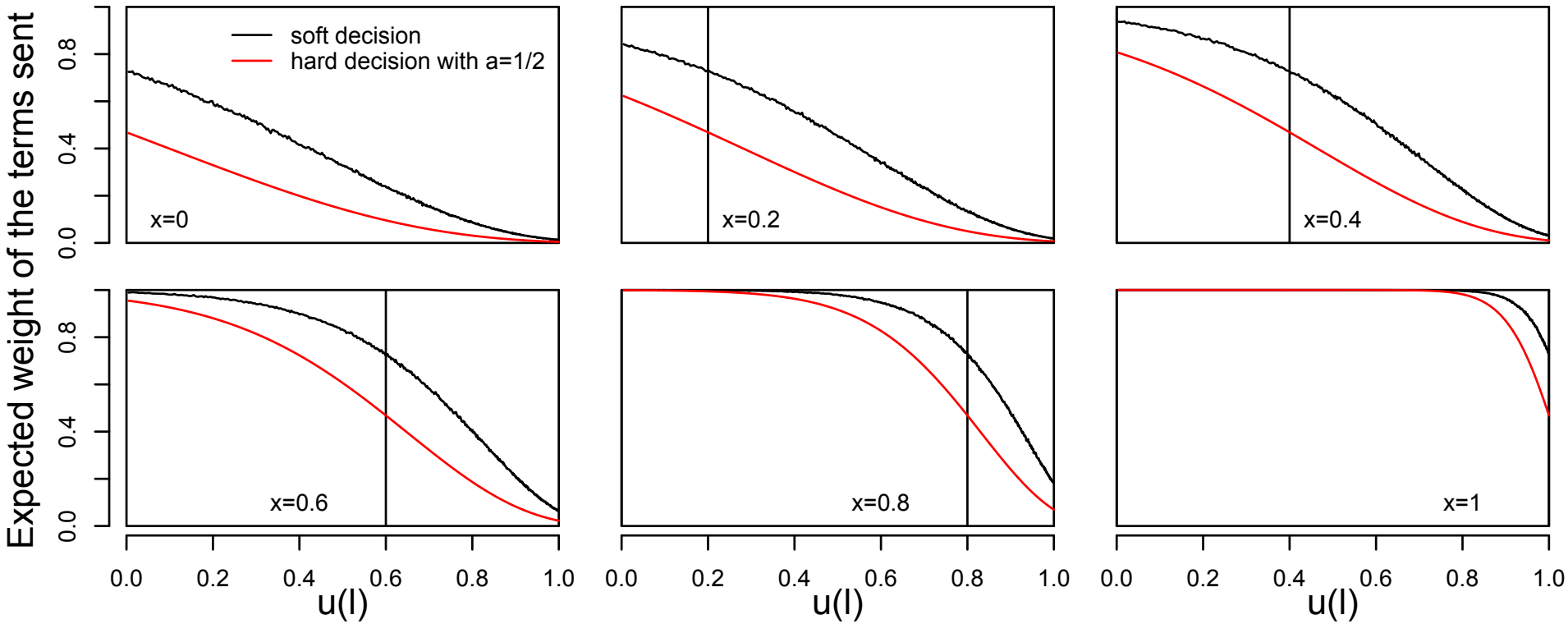
x_t: "Expected power-weighted fraction of correctly decoded sections in β^t"

Decoding Progression: $g(x)$ vs x



Success Progression Plots

SPARC with $M = 2^9$, $L = M$, $snr = 7, \mathcal{C} = 1.5$ bits and $R = 0.8\mathcal{C}$



The horizontal axis depicts $u(\ell) = 1 - e^{-2\mathcal{C}\ell/L}$ which is an increasing function of ℓ. Area under curve equals g(x).

All these plots are based on the *assumption* that in each step t, the residual-based statistics of the soft-decision decoder

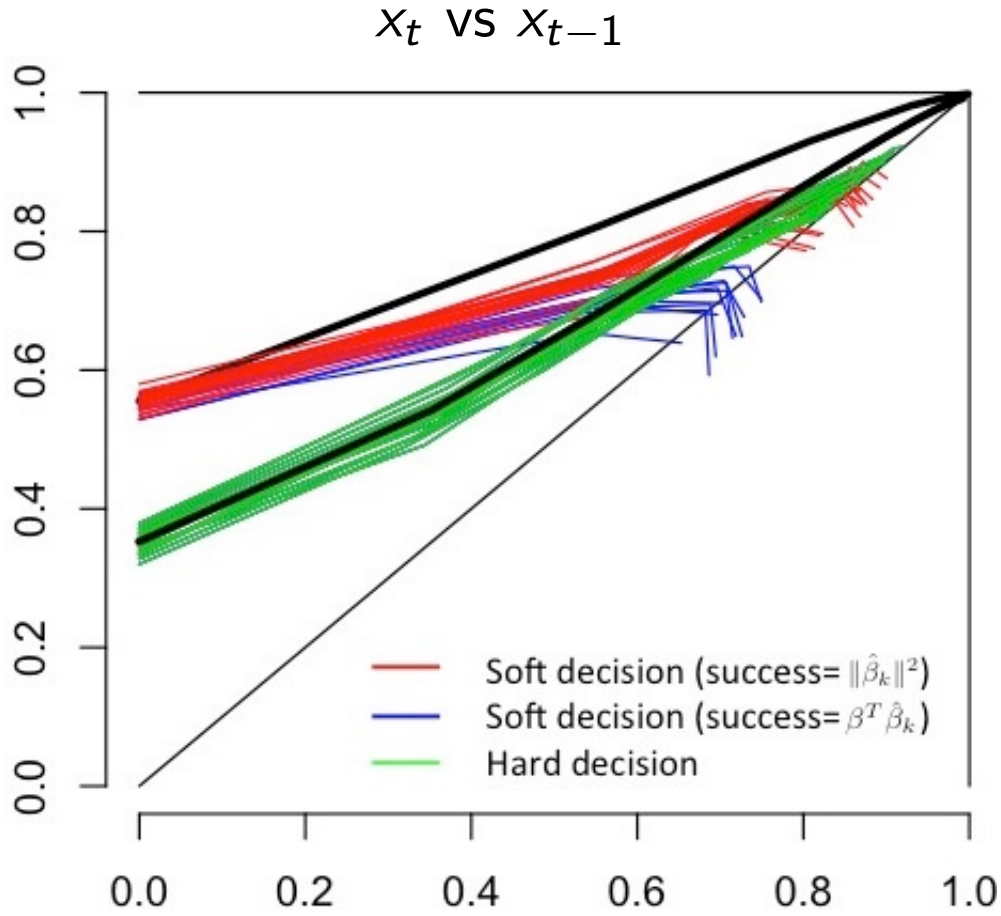
$$stat_{t,j} = \left(Y - A\beta^t_{-j}\right)^T A_j, \quad j = 1, \ldots, N$$

are distributed as $stat_t = \beta + \tau_t Z_t$

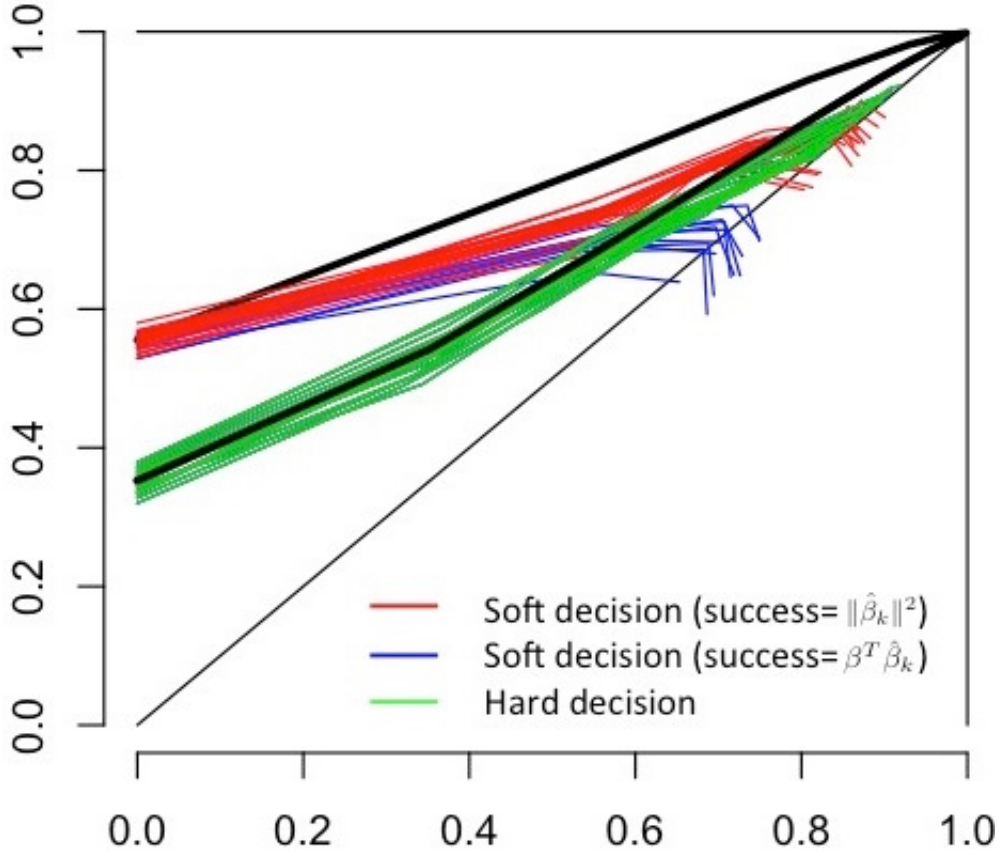
Is this valid ?

Simulation

SPARC: $L = 512, M = 64,\ snr = 15,\ \mathcal{C} = 2$ bits, $R = 1$ bit $(n = 3072)$



Green lines for hard thresholding; blue and red for soft decision decoder. Ran 20 trials for each method.



Clearly, empirical success rate x_t does not follow theoretical curve

$\Rightarrow$ Residual-based $stat_t$ *is not well-approximated* by $\beta + \tau_t Z$!

How to form $stat_t$ that is close to the desired representation?

Recall: Once $stat_t$ has the representation $stat_t = \beta + \tau_t Z_t$, it's easy to produce Bayes optimal estimate:

For $j \in$ section ℓ:

$$\beta_j^{t+1}(stat_t = s) = \mathbb{E}[\beta_j | stat_t = s] = \sqrt{nP_\ell} \underbrace{\frac{\exp(\sqrt{nP_\ell}\, s_j/\tau_t^2)}{\sum_{k \in sec_\ell} \exp(\sqrt{nP_\ell}\, s_k/\tau_t^2)}}_{w_j(\tau_t)}$$

General Framework of Iterative Statistics

For $t \geq 1$:

- Codeword fits: $F_t = A\beta^t$
- Vector of statistics: $stat_t =$ function of $(A, Y, F_1, \ldots, F_t)$
- e.g. $stat_{t,j}$ proportional to $A_j^T(Y - F_t)$
- Update β^{t+1} as a function of $stat_t$

General Framework of Iterative Statistics

For $t \geq 1$:

- Codeword fits: $F_t = A\beta^t$
- Vector of statistics: $stat_t =$ function of $(A, Y, F_1, \ldots, F_t)$
- e.g. $stat_{t,j}$ proportional to $A_j^T(Y - F_t)$
- Update β^{t+1} as a function of $stat_t$
  - Hard-thresholding: Adaptive Successive Decoder

    $\beta^{t+1,j} = \sqrt{nP_\ell}\, \mathbf{1}_{\{stat_{t,j} > thresh\}}$
  - Soft decision:

    $\beta^{t+1,j} = \mathbb{E}[\beta_j | stat_t] = \sqrt{nP_\ell}\, \hat{w}_{t,j}$

    with thresholding on the last step.

    KEY: We want $stat_t$ to distributed close to $\beta + \tau_t Z$

Orthogonal Components

- Codeword fits: $F_t = A\beta^t$
- Orthogonalization : Let $G_0 = Y$ and for $t \geq 1$

$$G_t = \text{part of fit } F_t \textit{ orthogonal} \text{ to } G_0, G_1, \ldots, G_{t-1}$$

- Components of statistics

$$\mathcal{Z}_{t,j} = \sqrt{n}\, A_j^T \frac{G_t}{\|G_t\|}, \qquad j = 1, \ldots, N$$

- Statistics such as residual-based $stat_{t,j}$ built from $A_j^T(Y - F_{t,-j})$ are linear combinations of these $\mathcal{Z}_{t,j}$

We now characterize the distribution of $\mathcal{Z}_t = (\mathcal{Z}_{t,1}, \ldots, \mathcal{Z}_{t,N})^T$

Distribution Evolution [Cho-Barron '12]

Lemma: Shifted Normal conditional distribution
Given $\mathcal{F}_{t-1} = (\|G_0\|, \ldots \|G_{t-1}\|, \mathcal{Z}_0, \mathcal{Z}_1, \ldots, \mathcal{Z}_{t-1})$,
$\mathcal{Z}_t = \sqrt{n}\, A^T G_t / \|G_t\|$ has the distributional representation

$$\mathcal{Z}_t = \frac{\|G_t\|}{\sigma_t} b_t + Z_t$$

- $\|G_t\|^2/\sigma_t^2 \sim$ Chi-square$(n-t) \approx \sqrt{n}$
- $b_0, b_1, \ldots, b_t$ the successive *orthonormal* components of

$$\begin{bmatrix} \beta \\ \sigma \end{bmatrix}, \begin{bmatrix} \beta^1 \\ 0 \end{bmatrix}, \ldots, \begin{bmatrix} \beta^t \\ 0 \end{bmatrix} \qquad (*)$$

- $Z_t \sim N(0, \Sigma_t)$ indep of $\|G_t\|$
- $\Sigma_t = I - b_0 b_0^T - b_1 b_1^T - \ldots - b_t b_t^T$
  $=$ projection onto space orthogonal to $(*)$
- $\sigma_t^2 = (\beta^t)^T \Sigma_{t-1} \beta^t$

Approximating Distribution for $\mathcal{Z}_t$

We approximate the distribution of $\mathcal{Z}_t = \sqrt{n}\, A^T G_t / \|G_t\|$ given $\mathcal{F}_{t-1}$ by

$$\mathcal{Z}_t \approx \sqrt{n}\, b_t \; + \; Z_t$$

where $Z_k \sim N(0, I - Proj_t)$ where $Proj_t$ is a projection matrix to the space spanned by $(\beta^1, \ldots, \beta^t)$.

This approximation is justified using the "ballpark method" . . .

The Ballpark Method

- A sequence $\mathbb{P}_L$ of true distributions of the statistics
- A sequence $\mathbb{Q}_L$ of convenient approximate distributions
- $D_\gamma(\mathbb{P}_L \| \mathbb{Q}_L)$, the Renyi divergence between the distributions

$$D_\gamma(\mathbb{P} \| \mathbb{Q}) = (1/\gamma) \log \mathbb{E}[(p(stat)/q(stat))^{\gamma-1}]$$

- A sequence A_L of events of interest

Lemma: If the Renyi divergence is bounded by a value D, then any event of exponentially small probability using the simplified measures $\mathbb{Q}_L$ also has exponentially small probabity using the true measures $\mathbb{P}_L$

$$\mathbb{P}(A_L) \leq e^{2D}[\mathbb{Q}(A_L)]^{1/2}$$

- With bounded D, allows treating statistics as Gaussian

Approximating distribution for $\mathcal{Z}_t$

We approximate the distribution for $\mathcal{Z}_t$ given $\mathcal{F}_{t-1}$ as

$$\mathcal{Z}_t = \sqrt{n} b_t + Z_t$$

where $Z_k \sim N(0, I - Proj_t)$ where $Proj_t$ is a projection matrix to the space spanned by $(\beta^1, \ldots, \beta^t)$.

Lemma. For any event A that is determined by the random variables $\{\|G_k\|, \mathcal{Z}_k\}$ for $k = 0, \ldots, t$, we have

$$\mathbb{P}A \leq \left((\mathbb{Q}A) e^{k(2+k^2/n+C)} \right)^{1/2}$$

Combining Components

Class of statistics $stat_t$ formed by combining $\mathcal{Z}_0, \ldots, \mathcal{Z}_k$:

$$stat_{t,j} = \tau_t \, \mathcal{Z}_{t,j}^{comb} + \beta_j^t, \qquad j = 1, \ldots, N$$

with $\mathcal{Z}_t^{comb} = \lambda_0 \, \mathcal{Z}_0 + \lambda_1 \, \mathcal{Z}_1 + \ldots + \lambda_t \, \mathcal{Z}_t$, $\sum_k \lambda_k^2 = 1$

Combining Components

Class of statistics $stat_t$ formed by combining $\mathcal{Z}_0, \ldots, \mathcal{Z}_k$:

$$stat_{t,j} = \tau_t \, \mathcal{Z}_{t,j}^{comb} + \beta_j^t, \qquad j = 1, \ldots, N$$

with $\mathcal{Z}_t^{comb} = \lambda_0 \, \mathcal{Z}_0 + \lambda_1 \, \mathcal{Z}_1 + \ldots + \lambda_t \, \mathcal{Z}_t$, $\sum_k \lambda_k^2 = 1$

Ideal Distribution of the combined statistics

$$stat_k^{ideal} = \beta + \tau_t Z_k^{comb}$$

where $\tau_t^2 = \sigma^2 + (1 - x_t)P$, and Z_k^{comb} i.i.d. $\sim \mathcal{N}(0, 1)$

What choice of λ_k's gives you the ideal ?

Oracle statistics

Choosing weights based on $(\lambda_0, \ldots, \lambda_t)$ proportional to

$$\left((\sqrt{n(\sigma^2 + P)} - b_0^T \beta^t),\ -(b_1^T \beta^t), \ldots, -(b_t^T \beta^t) \right)$$

Combining $\mathcal{Z}_k$ with these weights, replacing χ_{n-k} with $\sqrt{n}$, it produces the desired distributional representation

$$stat_t = \tau_t \sum_{k=0}^{t} \lambda_k \mathcal{Z}_k + \beta^t \approx \beta + \tau_t Z_t^{comb}$$

with $Z_t^{comb} \sim N(0, \mathrm{I})$.

But can't calculate these weights not knowing β: $b_0 = \beta / \sqrt{P + \sigma^2}$

Oracle weights vs Estimated Weights

Oracle weights of combination: $(\lambda_0, \ldots, \lambda_t)$ proportional to

$$\left((\sqrt{n(\sigma^2 + P)} - b_0^T \beta^t),\ -(b_1^T \beta^t), \ldots, -(b_t^T \beta^t) \right)$$

produces $stat_t$ with the desired representation $\beta + \tau_t Z_t^{comb}$

Estimated weights of combination: $(\lambda_1, \ldots, \lambda_t)$ proportional to

$$\left((\|Y\| - \mathcal{Z}_0^T \beta^t / \sqrt{n},\ -(\mathcal{Z}_1^T \beta^t)/\sqrt{n}, \ldots, -(\mathcal{Z}_k^T \beta^t)/\sqrt{n} \right)$$

produce the *residual-based* statistics previously discussed, which we have seen are *not* close enough to the desired distribution

Orthogonalization Interpretation of Ideal Weights

Estimation of $\big((\sigma_Y - b_0^T\beta^t),\ -(b_1^T\beta^t), \ldots, -(b_t^T\beta^t)\big)$:

These $b_k^T\beta^t$ arise in the QR-decomposition for $B = [\beta, \beta^1, \ldots, \beta^t]$

$$B = [b_0 \quad b_1 \quad \ldots \quad b_k] \begin{bmatrix} b_0^T\beta & b_0^T\beta^1 & \ldots & b_0^T\beta^t \\ 0 & b_1^T\beta^1 & \ldots & b_1^T\beta^t \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \ldots & b_t^T\beta^t \end{bmatrix}$$

Noting that $b_0, \ldots, b_t$ are orthonormal, we can express B^TB as ...

Cholesky Decomposition of $B^T B$

$$\begin{bmatrix} \beta^T\beta & \beta^T\beta^1 & \dots & \beta^T\beta^t \\ (\beta^1)^T\beta & (\beta^1)^T\beta^1 & \dots & (\beta^1)^T\beta^t \\ \vdots & \vdots & \ddots & \vdots \\ (\beta^t)^T\beta & \dots & \dots & (\beta^t)^T\beta^t \end{bmatrix} = R^T \begin{bmatrix} b_0^T\beta & b_0^T\beta^1 & \dots & b_0^T\beta^t \\ 0 & b_1^T\beta^1 & \dots & b_1^T\beta^t \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & b_t^T\beta^t \end{bmatrix}$$

Deterministic weights:

- Replace elements on LHS with deterministic values under desired representation: $\frac{1}{n}\mathbb{E}(\beta^k)^T\beta^t = \frac{1}{n}\mathbb{E}(\beta^k)^T\beta = x_k P$ for $k \leq t$
- Then perform Cholesky decomposition of LHS to get deterministic weights

Cholesky Decomposition of $B^T B$

With $\tau_k^2 = \sigma^2 + (1 - x_k)P$:

$$\begin{bmatrix} \tau_0^2 & x_1 P & \dots & x_t P \\ x_1 P & x_1 P & \dots & x_1 P \\ \vdots & \vdots & \ddots & \vdots \\ x_t P & \dots & \dots & x_t P \end{bmatrix} = R^T \begin{bmatrix} \tau_0 & \tau_0 - \tau_1^2\sqrt{\omega_0} & \dots & \sigma_Y - \tau_k^2\sqrt{\omega_0} \\ 0 & \tau_1^2\sqrt{\omega_1} & \dots & \tau_k^2\sqrt{\omega_1} \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \tau_t^2\sqrt{\omega_t} \end{bmatrix}$$

In the above, $\omega_0 = \frac{1}{\tau_0^2}$ and $\omega_k = \frac{1}{\tau_k^2} - \frac{1}{\tau_{k-1}^2}$ for $k \geq 1$

The last column gives the deterministic weights of combination

Deterministic weights of combination

For given $\beta^1, \ldots, \beta^t$ and $\tau_k^2 = \sigma^2 + (1 - x_k)P$

Combine $\mathcal{Z}_k = \sqrt{n}\, b_k + Z_k, \quad 0 \le k \le t$ with

$$\underline{\lambda}^* = \tau_t \left(\frac{1}{\tau_0}, -\sqrt{\frac{1}{\tau_1^2} - \frac{1}{\tau_0^2}}, \ldots, -\sqrt{\frac{1}{\tau_t^2} - \frac{1}{\tau_{t-1}^2}} \right)$$

yielding approximately optimal statistics

$$stat_t = \tau_t \sum_{k=0}^{t} \lambda_k^* \mathcal{Z}_k \; + \; \beta^t$$

Reliability under $\mathbb{Q}$

Lemma [Barron-Cho]: For $t \geq 1$,

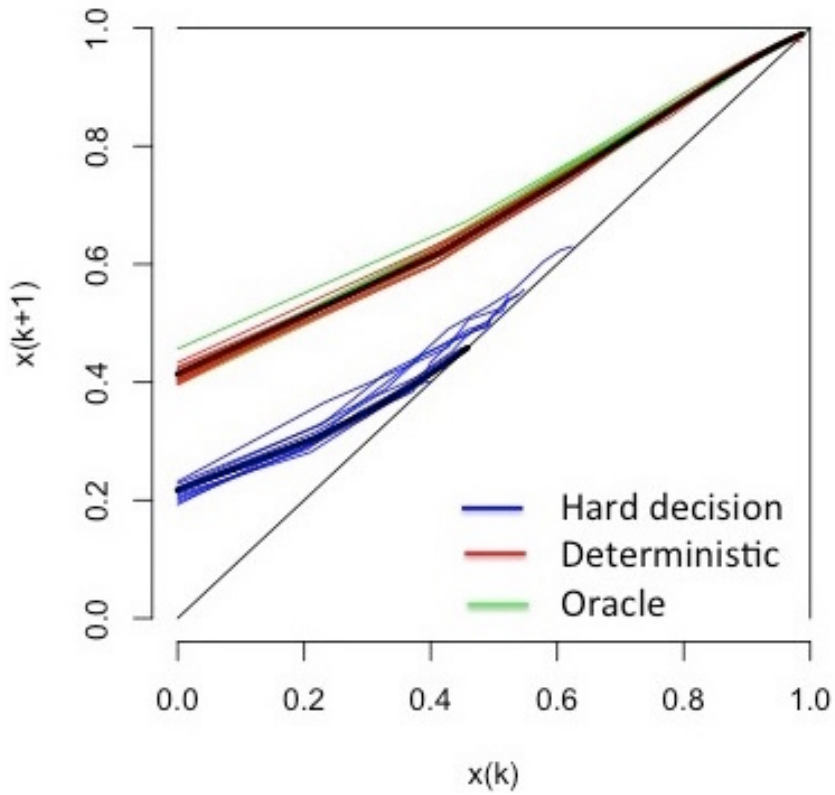
$$A_t = \left\{ \left| \frac{1}{nP} \beta^T \beta^t - x_t \right| > \eta_t \right\} \cup \left\{ \left| \frac{1}{nP} \|\beta^t\|^2 - x_t) \right| > \eta_t \right\}$$

with $\eta_t \sim (\log M)\, \eta_{t-1}$. Then, we have

$$\mathbb{Q}\{\cup_{k=1}^t A_k\} \lesssim \sum_{k=1}^t 6(k+1) \exp\left(-\frac{L}{4c^2} \eta_k^2 \right)$$

Update Plots for Deterministic and Oracle Weights

$L = 512,\ M = 64,\ snr = 15,\ C = 2\ bits,\ R = 0.7\mathcal{C},\ n = 2194$



Ran 10 trials for each method. We see that they follow the expected update function.

Cholesky decomposition based estimates

$$\begin{bmatrix} \beta^T\beta & \beta^T\beta^1 & \dots & \beta^T\beta^t \\ (\beta^1)^T\beta & (\beta^1)^T\beta^1 & \dots & (\beta^1)^T\beta^t \\ \vdots & \vdots & \ddots & \vdots \\ (\beta^t)^T\beta & \dots & \dots & (\beta^t)^T\beta^t \end{bmatrix} = R^T \begin{bmatrix} b_0^T\beta & b_0^T\beta^1 & \dots & b_0^T\beta^t \\ 0 & b_1^T\beta^1 & \dots & b_1^T\beta^t \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & b_t^T\beta^t \end{bmatrix}$$

Another Idea: Instead of replacing entire LHS with deterministic values, retain the entries we already know ...

Cholesky Decomposition-based Estimated Weights

$$\begin{bmatrix} \beta^T\beta & \beta^T\beta^1 & \dots & \beta^T\beta^t \\ (\beta^1)^T\beta & (\beta^1)^T\beta^1 & \dots & (\beta^1)^T\beta^t \\ \vdots & \vdots & \ddots & \vdots \\ (\beta^t)^T\beta & \dots & \dots & (\beta^t)^T\beta^t \end{bmatrix} = R^T \begin{bmatrix} b_0^T\beta & b_0^T\beta^1 & \dots & b_0^T\beta^t \\ 0 & b_1^T\beta^1 & \dots & b_1^T\beta^t \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & b_t^T\beta^t \end{bmatrix}$$

Under $\mathbb{Q}$, we have

$$\frac{\mathcal{Z}_k^T\beta^k}{\sqrt{n}} = (b_k^T\beta^k)$$

Based on the estimates we can recover the rest of the components of the matrix, which leads us to oracle weights under the approximating distribution, which we denote $\hat{\lambda}_t$.

Cholesky weights of combination

Combine $\mathcal{Z}_0, \ldots, \mathcal{Z}_t$ with weights $\hat{\lambda}_t$:

$$stat_t = \hat{\tau}_t \sum_{k=0}^{t} \hat{\lambda}_k \mathcal{Z}_k \; + \; \beta^t$$

where $\hat{\tau}_t^2 = \sigma^2 + \|\beta - \beta^t\|^2$, to get desired distributional representation under $\mathbb{Q}$:

$$stat_t \approx \beta + \tau_t Z_k^{comb}$$

Reliability under $\mathbb{Q}$

Lemma [Barron-Cho]: Suppose we have a Lipschitz condition on the update function with $c_{Lip} \leq 1$ so that

$$|g(x_1) - g(x_2)| \leq c_{Lip}|x_1 - x_2|.$$

For $t \geq 1$,

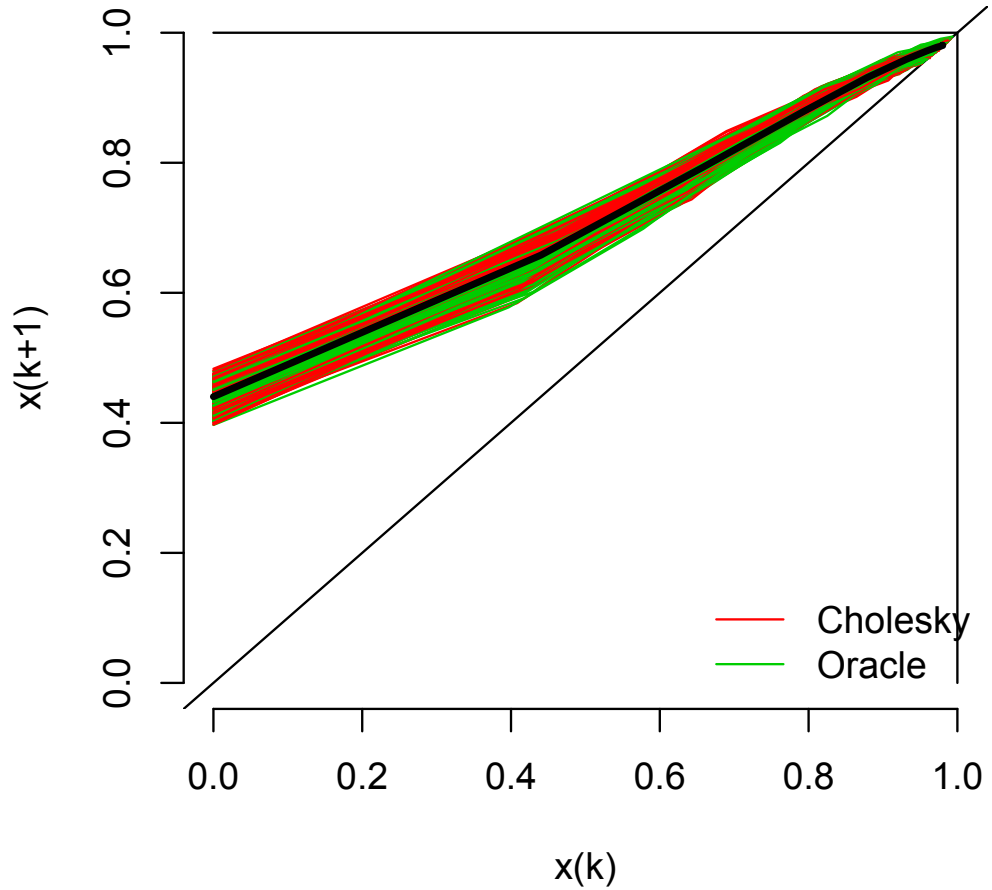
$$A_t = \left\{ \left| \frac{1}{nP} \beta^T \beta^t - x_t \right| > t\eta \right\} \cup \left\{ \left| \frac{1}{nP} \|\beta^t\|^2 - x_t \right| > t\eta \right\}$$

Then, we have

$$\mathbb{Q}\{\cup_{k=1}^{t} A_k\} \lesssim \exp\left(-\frac{L}{8c^2} \eta^2 \right)$$

Update Plots for Cholesky-based Weights

$L = 512,\ M = 64,\ snr = 7,\ C = 1.5\ bits,\ R = 0.7\mathcal{C},\ n = 2926$

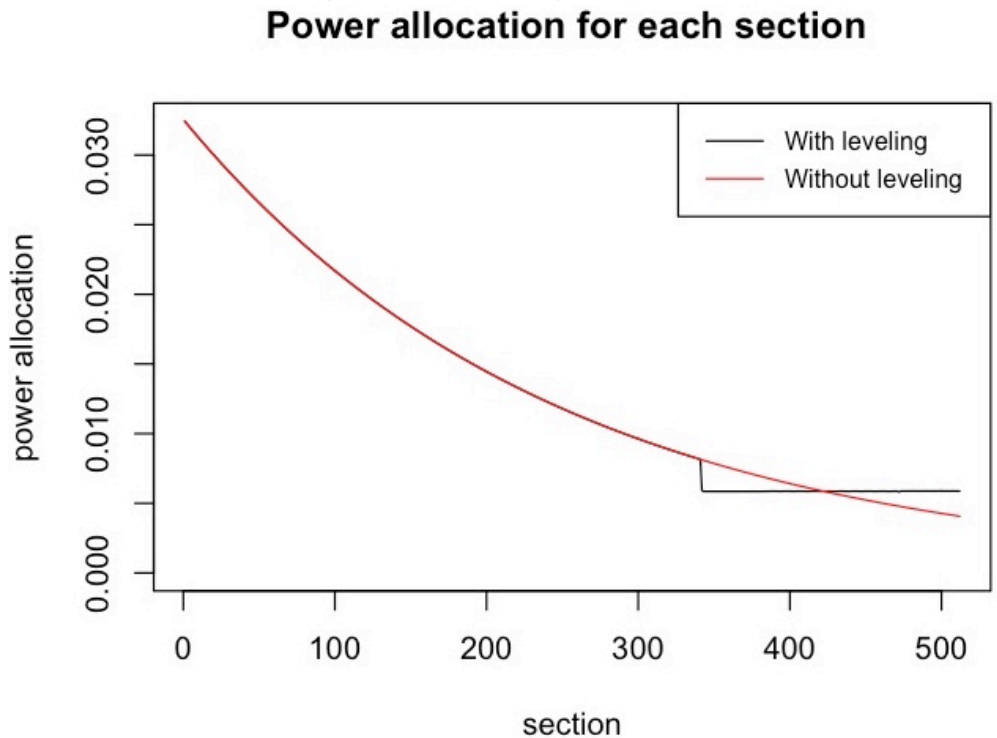


Red (cholesky decomposition based weights); green (oracle weights of combination). Ran 10 trials for each.

Improving the End Game

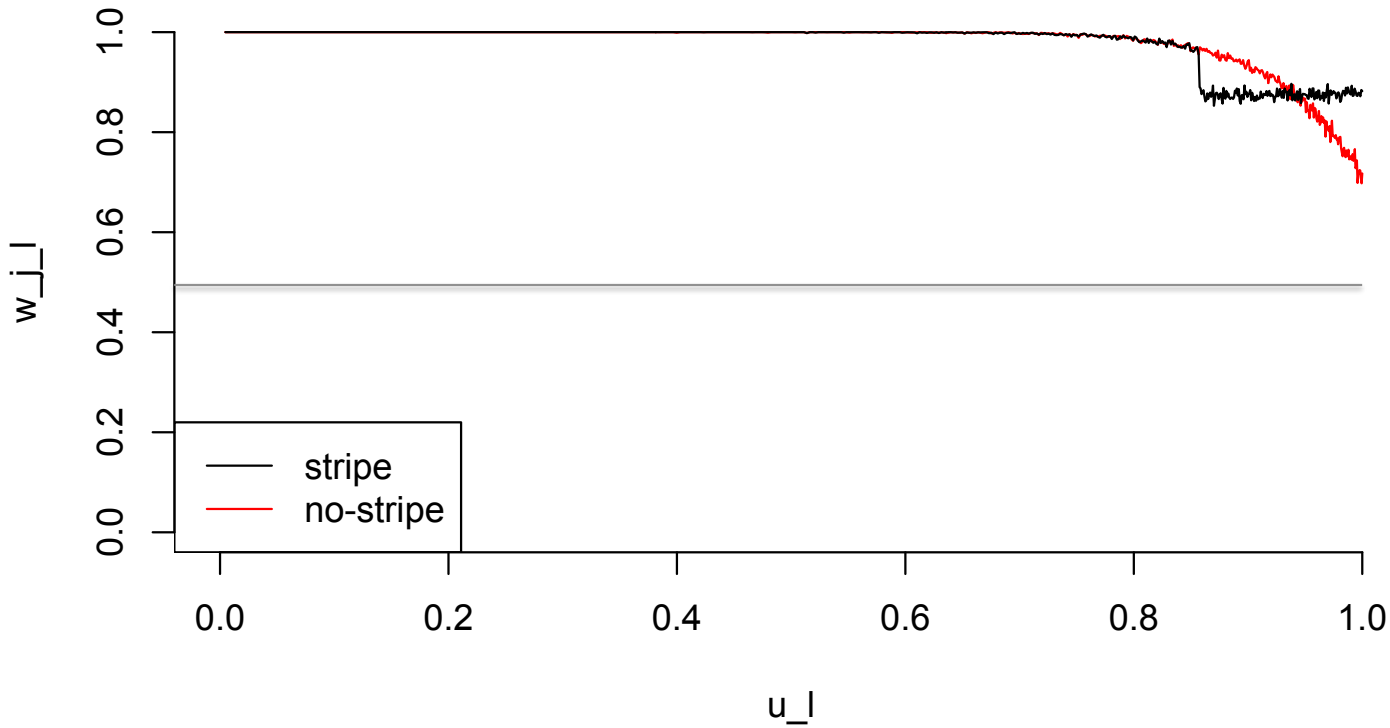
- Variable power: P_ℓ proportional to $e^{-2C\ell/L}$ for $\ell = 1, \ldots, L$
- When R is not close to C, say $R = 0.6C$, this allocates too much power to initial sections, leaving too little for the end
- We use alternative power allocation: constant leveling the power allocation for the last portion of the sections

$L = 512$, $snr = 7$, $C = 1.5$ bits



Progression Plot using Alternative Power Allocation

$L = 512,\ M = 64,\ snr = 7,\ C = 1.5$ bits, $R = 0.7\mathcal{C},\ n = 2926.$



Progression plot of the final step. The area under the curve might be the same, the expected weights for the last sections are higher when we level the power at the end

Summary

Sparse superposition codes with adaptive successive decoding

- Simplicity of the code permits:
  - distributional analysis of the decoding progression
  - low complexity decoder
  - exponentially small error probability for any fixed $R < C$
- Asymptotics superior to polar code bounds for such rates

Next . . .

- Approximate message passing (AMP) decoding
- Power-allocation schemes to improve finite block-length performance