Communication by Regression: Sparse Superposition Codes

Andrew Barron
Department of Statistics, Yale University

Coauthors: Antony Joseph and Sanghee Cho

March 20, 2013, Brown University
Channel Communication Set-up

- **Input bits:** $U = (U_1, U_2, \ldots, U_K)$ indep Bern(1/2)
  
  $\downarrow$

- **Encoded:** $x = (x_1, x_2, \ldots, x_n)$
  
  $\downarrow$

- **Channel:** $p(y|x)$
  
  $\downarrow$

- **Received:** $Y = (Y_1, Y_2, \ldots, Y_n)$
  
  $\downarrow$

- **Decoded:** $\hat{U} = (\hat{U}_1, \hat{U}_2, \ldots, \hat{U}_K)$

- **Rate:** $R = \frac{K}{n}$
  
  Capacity $C$

- **Reliability:** Want small Prob $\{\hat{U} \neq U\}$ and small Prob $\{\text{Fraction mistakes} \geq \alpha\}$
Gaussian Noise Channel

- Input bits: \( U = (U_1, U_2, \ldots, U_K) \)
- Encoded: \( x = (x_1, x_2, \ldots, x_n) \)  
  \[ \frac{1}{n} \sum_{i=1}^{n} x_i^2 \approx P \]
- Channel: \( p(y|x) \)  
  \[ y = x + \varepsilon, \varepsilon \sim N(0, \sigma^2 I) \]
  \[ snr = \frac{P}{\sigma^2} \]
- Received: \( Y = (Y_1, Y_2, \ldots, Y_n) \)
- Decoded: \( \hat{U} = (\hat{U}_1, \hat{U}_2, \ldots, \hat{U}_K) \)
- Rate: \( R = \frac{K}{n} \)
  Capacity \( C = \frac{1}{2} \log(1 + snr) \)
- Reliability: Want small \( \text{Prob}\{\hat{U} \neq U\} \)
  and small \( \text{Prob}\{\text{Fraction mistakes} \geq \alpha\} \)
Sparse Superposition Code

- **Input bits:** $U = (U_1 \ldots U_K)$
- **Coefficients:** $\beta = (00 \times 0000000000 \times 00 \ldots 0 \times 00000)^T$
- **Sparsity:** $L$ entries non-zero out of $N$
- **Matrix:** $X$, $n$ by $N$, all entries indep Normal(0, 1)
- **Codeword:** $X\beta$, superposition of a subset of columns
- **Receive:** $Y = X\beta + \varepsilon$, a statistical linear model
- **Decode:** $\hat{\beta}$ and $\hat{U}$ from $X, Y$
Sparse Superposition Code

- **Input bits:** \( U = (U_1 \ldots U_K) \)
- **Coefficients:** \( \beta = (00 \ast 0000000000 \ast 00 \ldots 0 \ast 000000)^T \)
- **Sparsity:** \( L \) entries non-zero out of \( N \)
- **Matrix:** \( X, \ n \text{ by } N, \) all entries indep Normal(0, 1)
- **Codeword:** \( X\beta, \) superposition of a subset of columns
- **Receive:** \( Y = X\beta + \varepsilon, \) a statistical linear model
- **Decode:** \( \hat{\beta} \) and \( \hat{U} \) from \( X, Y \)
- **Rate:** \( R = \frac{K}{n} \) from \( K = \log \left( \frac{N}{L} \right), \) near \( L \log \left( \frac{N}{L} e \right) \)
Partitioned Superposition Code

- **Input bits**: \( U = (U_1 \ldots, \ldots, \ldots, \ldots U_K) \)
  
  \( L \) sections, each of size \( \log_2 M \)

- **Coefficients**: \( \beta = (00 \times 00000, 00000 \times 00, \ldots, 0 \times 000000) \)
  
  \( L \) sections, each of size \( M = N/L \), a power of 2

- **Sparsity**: 1 non-zero entry in each section

- **Indices of nonzeros**: \((j_1, j_2, \ldots, j_L)\) specified by \( U \) segments

- **Matrix**: \( X \), \( n \) by \( N \), splits into \( L \) sections

- **Codeword**: \( X\beta \), superposition of columns, one from each

- **Receive**: \( Y = X\beta + \varepsilon \)

- **Decode**: \( \hat{\beta} \) and \( \hat{U} \)

- **Rate**: \( R = \frac{K}{n} \) from \( K = L \log \frac{N}{L} = L \log M \)
Partitioned Superposition Code

- **Input bits:** \( U = (U_1 \ldots, \ldots, \ldots, \ldots, U_K) \)
  
  \( L \) sections, each of size \( \log_2 M \)

- **Coefficients:** \( \beta = (00 * 00000, 00000 * 00, \ldots, 0 * 000000) \)
  
  \( L \) sections, each of size \( M = N/L \), a power of 2

- **Sparsity:** 1 non-zero entry in each section

- **Indices of nonzeros:** \((j_1, j_2, \ldots, j_L)\) specified by \( U \) segments

- **Matrix:** \( X \), \( n \) by \( N \), splits into \( L \) sections

- **Codeword:** \( X\beta \), superposition of columns, one from each

- **Receive:** \( Y = X\beta + \varepsilon \)

- **Decode:** \( \hat{\beta} \) and \( \hat{U} \)

- **Rate:** \( R = \frac{K}{n} \) from \( K = L \log \frac{N}{L} = L \log M \)

- **Ultra-sparse case:** Impractical \( M = 2^{nR/L} \) with \( L \) constant
  
  (reliable at all \( R < C \): Cover 1972, 1980)

- **Moderately-sparse:** Practical \( M = n \) with \( L = nR/\log n \)
  
  (still reliable at all \( R < C \))
Partitioned Superposition Code

- **Input bits:** \( U = (U_1 \ldots, \ldots, \ldots, \ldots U_K) \)
  - \( L \) sections, each of size \( \log_2 M \)
- **Coefficients:** \( \beta = (00 \ast 00000, 00000 \ast 00, \ldots, 0 \ast 000000) \)
  - \( L \) sections, each of size \( M = N/L, \) a power of 2
- **Sparsity:** 1 non-zero entry in each section
- **Indices of nonzeros:** \( (j_1, j_2, \ldots, j_L) \) specified by \( U \) segments
- **Matrix:** \( X, \) \( n \) by \( N, \) splits into \( L \) sections
- **Codeword:** \( X\beta, \) superposition of columns, one from each
- **Receive:** \( Y = X\beta + \varepsilon \)
- **Decode:** \( \hat{\beta} \) and \( \hat{U} \)
- **Rate:** \( R = \frac{K}{n} \) from \( K = L \log \frac{N}{L} = L \log M \)
- **Reliability:** small Prob\{Fraction mistakes \( \geq \alpha\} \) small \( \alpha \)
- **Outer RS code:** rate \( 1 - 2\alpha, \) corrects remaining mistakes
- **Overall rate:** \( R_{tot} = (1 - 2\alpha)R \)
- **Overall rate:** up to capacity
Power Allocation

- **Coefficients**: \( \beta = (00*00000, 00000*00, \ldots, 0*000000) \)
- **Indices of nonzeros**: \( \text{sent} = (j_1, j_2, \ldots, j_L) \)
- **Coeff. values**: \( \beta_{j\ell} = \sqrt{P_{\ell}} \) for \( \ell = 1, 2, \ldots, L \)
- **Power control**: \( \sum_{\ell=1}^{L} P_{\ell} = P \)
- **Codewords**: \( X\beta \), have average power \( P \)

**Power Allocations**

- **Constant power**: \( P_{\ell} = P/L \)
- **Variable power**: \( P_{\ell} \) proportional to \( e^{-2C_{\ell}/L} \)
- **Variable with leveling**
Decoding Steps (with thresholding)

**Start**: [Step 1]
- Compute the inner product of $Y$ with each column of $X$
- See which are above a threshold
- Form initial fit as weighted sum of columns above threshold

**Iterate**: [Step $k \geq 2$]
- Compute the inner product of residuals $Y - Fit_{k-1}$ with each remaining column of $X$
- See which are above threshold
- Add these columns to the fit

**Stop**:
- At Step $k = 1 + snr \log M$, or
- if there are no additional inner products above threshold
Complexity in parallel pipelined implementation

- **Space**: (use $k = snr \log M$ copies of the $n$ by $N$ dictionary)
  
  - $knN = snr C M n^2$ memory positions
  - $kN$ multiplier/accumulators and comparators

- **Time**: $O(1)$ per received $Y$ symbol
Adaptive Successive Decoder

Decoding Steps (with iteratively optimal statistics)

**Start**: [Step 1]
- Compute the inner product of $Y$ with each column of $X$
- Form initial fit

**Iterate**: [Step $k \geq 2$]
- Compute inner product of residuals $Y - Fit_{k-1}$ with each $X_j$.
- Adjusted form: $Z_{k,j}$ equals $(Y - X \hat{\beta}_{k-1,-j})^T X_j$
- Standardize by dividing it by $\| Y - X \hat{\beta}_{k-1} \|$.
- Form the new fit

$$\hat{\beta}_{k,j} = \sqrt{P_\ell} \ w_j(b) = \sqrt{P_\ell} \ \frac{e^b Z_{k,j}}{\sum_{j \in \text{sec}_\ell} e^b Z_{k,j}}$$

**Stop**:
- At Step $k = O(\log M)$ if $R < C$.
- At Step $k = M^\alpha$, with $0 < \alpha < 1$, if $R$ matches $C$ to within polynomially small amount.
Approximate distribution of $Z_{k,j}$:

$$Z_{k,j} = \frac{\sqrt{n} \beta_j}{\sqrt{\sigma^2 + E\|\hat{\beta}_{k-1} - \beta\|^2}} + Z_{k,j}$$

with $Z_{k,j}$ independent standard normal.

$$Z_{k,j} = b_{\ell,x_{k-1}} 1\{j \text{ sent}\} + Z_{k,j}$$

where

$$b = \sqrt{\frac{n \ell}{\sigma^2 + P(1 - x)}}$$

Here

$$E\|\hat{\beta}_{k-1} - \beta\|^2 = P(1 - x_{k-1})$$

Update rule $x_k = g(x_{k-1})$ where

$$g(x) = \sum_{\ell=1}^{L} (P_{\ell}/P) E[w_{j_{\ell}}(b_{\ell,x})].$$
Decoding progression

Figure: Plot of $g(x)$ and the sequence $x_k$. 

$M = 2^9$, $L = M$
$\text{snr} = 7$
$C = 1.5 \text{ bits}$
$R = 1.2 \text{ bits (0.8C)}$
Figure: Comparison of update functions. Blue and light blue lines indicates \( \{0, 1\} \) decision using the threshold \( \tau = \sqrt{2 \log M} + a \) with respect to the value \( a \) as indicated.
Figure: Transition plots: $M = 2^9$, $L = M$, $C = 1.5$ bits and $R = 0.8C$. We used Monte Carlo simulation with replicate size 10000. The horizontal axis depicts $u(\ell) = 1 - e^{-2C\ell/L}$ which is an increasing function of $\ell$. 
Result for Optimal ML Decoder [Joseph and B. 2012], with outer RS decoder, and with equal power allowed across the sections

- Prob error exponentially small in $n$ for all $R < C$

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

- In agreement with the Shannon-Gallager exponent of optimal code, though with a suboptimal constant $V$ depending on the $\text{snr}$
Rate and Reliability of Fast Superposition Code

Practical: Adaptive Successive Decoder [B. and Joseph 2011]

- prob error exponentially small in $n/(\log M)^{1/2}$ for $R < C$
- Value $C_M$ approaching capacity

$$C_M = \frac{C}{1 + c_1/\log M}$$

- Probability error exponentially small in $L$ for $R < C_M$

$$\text{Prob}\{\text{Error}\} \leq e^{-L(C_M-R)^2c_2}$$

- Improves to $e^{-c_3L(C_M-R)^2(\log M)^{0.5}}$ using a Bernstein bound.
- Nearly optimal when $C_M - R$ is at least $C - C_M$.
- Our $c_1$ is near $(2.5 + 1/snr) \log \log M + 4C$
Some Relationships to Other Work

- Forney (1960): Concatenated codes
- Barg, Zémor (2002, 2004): Expander codes for the BSC:
  - exponential error bounds and linear complexity for $R < C$
- LDPC and turbo codes:
  - some theoretical analysis (Richardson, Urbanke 2008), yet obstacles remain for proof of rates up to capacity
- Arikan polar codes for Gaussian channel (Abbe, Bar. 2011):
  - $q$ quantization levels
  - $R < C_q$ with gap $C - C_q \leq \text{snr} / q$
  - Error bound from Hassani, Urbanke (2011) in $q = 2$ case:
    \[
    \text{Prob}\{\text{Error}\} \leq 2^{-n(1-\alpha)/2(C_q-R)^{1/2\alpha}} (1+o(1))
    \]
- Tropp 08 codes from compressive sensing; related work:
  - Wainwright; Fletcher, Rangan, Goyal; Zhang; others
  - $\ell_1$-constrained least squares practical, has positive rate
  - but not capacity achieving
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

- Currently studying rates $R_n$ approaching $C$, at a polynomial rate (slower than $1/\sqrt{n}$)
\[ w_j(b) = P[j_\ell = j \mid Z_k] = \frac{e^{b Z_{k,j}}}{\sum_{j \in \text{sec}_\ell} e^{b Z_{k,j}}} \]

The following quantities have the same expectation when \( j_\ell = 1 \) was sent for each section \( \ell \).

(i) \( 1 - w_1 \)

(ii) \( (1 - w_1)^2 + \sum_{j=2}^{m} w_j^2 = \| e_1 - w \|^2 \)

(iii) \( 1 - \sum_{j \in \text{sec}_\ell} w_j^2 \)

From these identities, we can estimate \( \beta^T \hat{\beta}_k \) by \( \| \hat{\beta}_k \|^2 \). Likewise, \( \| \beta - \hat{\beta}_k \|^2 \) and \( P - \| \hat{\beta}_k \|^2 \) have the same expectation.

Each of these is an average of \( L \) independent random variables bounded by 1, so the error of these estimates is accordingly small except in events of exponentially small probability.
Decoding progression

Figure: Plot of $g(x)$ and the sequence $x_k$. 

$M = 2^9$, $L = M$

$\text{snr}=7$

$C=1.5 \text{ bits}$

$R=1.2 \text{ bits}(0.8C)$

Andrew Barron  Communication by Regression  22/1
**Figure:** Comparison of update functions. Blue and light blue lines indicates 0, 1 decision using the threshold $\hat{a}2 \log M + a$ with respect to the value $a$ as indicated.
Figure: Transition plots: $M = 2^9$, $L = M$, $C = 1.5$ bits and $R = 0.8C$. We used Monte Carlo simulation with replicate size 10000. The horizontal axis depicts $u(\ell) = 1 - e^{-2C\ell/L}$ which is an increasing function of $\ell$. 