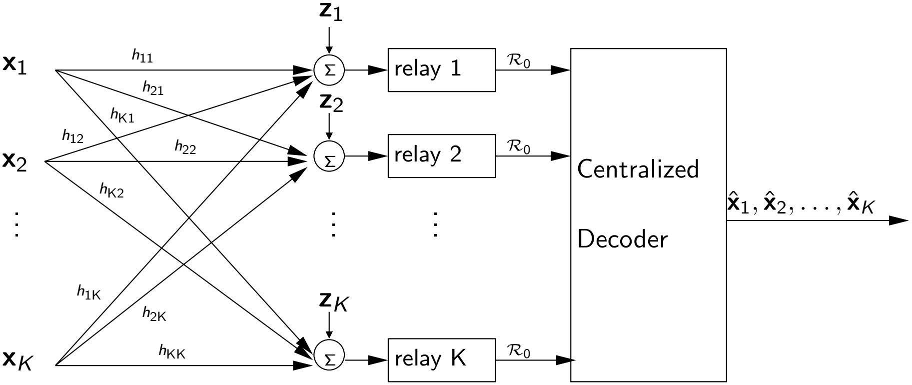
Practical Code Design for Compute-and-Forward

Or Ordentlich
Joint work with Jiening Zhan, Uri Erez, Michael Gastpar and Bobak Nazer

ISIT 2011
St. Petersburg
Russia

The linear Gaussian network



- K distributed users.
- K relays. Can cooperate only through a centralized decoder.
- Clean bit pipes of rate $\mathcal{R}_0$ between the relays and the decoder.
- Each relay sees a linear combination of all signals plus AWGN.
- Same power constraint for all users: $\frac{1}{n}\sum_{t=1}^{n} X_l^2[t] \leq \text{SNR}, \quad \forall l.$

Compute-and-Forward

- There are several approaches: Decode-and-Forward, Compress-and-Forward...

Compute-and-Forward [Nazer and Gastpar 2011]

- Each relay decodes a linear combination of the transmitted signals.
- The decoded linear combination is passed to the centralized decoder.
- Upon receiving a full-rank set of equations, the centralized decoder recovers the original messages.
- The scheme crucially depends on using linear codes.

- The scheme of [Nazer and Gastpar 2011] uses infinite dimensional nested lattice codebooks. Not possible for implementation...
- How can we approach the theoretical results with practical schemes?

Compute-and-Forward - practical implementation

Previous work

- Feng et al. (ISIT 2010) took an algebraic approach and showed promising simulation results with signal codes of block length 100.
- Hern and Narayanan (ISIT 2011) used multilevel codes, and decoded non-linear functions of the transmitted layers.

- In this work we seek a practical implementation that utilizes "off-the-shelf" encoders and decoders.
- Our scheme is essentially based on using linear q-ary codes with a "twist".

Compute-and-Forward with q-ary linear codes

- The received signal at relay k is

$$\mathbf{y}_k = \sum_{l=1}^{K} h_{kl}\mathbf{x}_l + \mathbf{z}_k .$$

- All users $\{\mathbf{x}_l\}_{l=1}^{K}$ encode their messages using the same linear codebook $\mathcal{C}$ over $\mathbb{Z}_q$.
- For $\mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C}$ the linearity of $\mathcal{C}$ implies

$$[a_1\mathbf{c}_1 + a_2\mathbf{c}_2] \bmod q \in \mathcal{C}, \quad \forall a_1, a_2 \in \mathbb{Z} \; .$$

- Relay k chooses a vector of integer coefficients $\mathbf{a}_k = \left[a_{k1} \; a_{k2} \ldots \; a_{kK}\right]^T \in \mathbb{Z}^L$, and attempts to decode

$$\mathbf{u}_k = \left[\sum_{l=1}^{K} a_{kl}\mathbf{x}_l\right] \bmod q \in \mathcal{C} \; .$$

Compute-and-Forward with q-ary linear codes

- Before decoding, relay k computes

$$\begin{aligned}\tilde{\mathbf{y}}_k &= [\alpha_k \mathbf{y}_k] \bmod q \\ &= \left[\sum_{l=1}^{K} a_{kl}\mathbf{x}_l + \sum_{l=1}^{K} (\alpha_k h_{kl} - a_{kl})\,\mathbf{x}_l + \alpha_k \mathbf{z}_k\right] \bmod q \\ &= [\mathbf{u}_k + \tilde{\mathbf{z}}_k] \bmod q.\end{aligned}$$

- α_k is chosen such as to optimize the tradeoff between decreasing the residual "self" noise and increasing the Gaussian noise.
- The decoded codeword $\hat{\mathbf{u}}_k$ is passed to the centralized unit along with the coefficients $\mathbf{a}_k$.

Compute-and-Forward with q-ary linear codes

- The centralized decoder gets a decoded equation and its coefficients from each relay.
- The centralized decoder has to solve

$$U = AX \bmod q.$$

- A has to be invertible over $\mathbb{Z}_q$.
- A is likely to be invertible if q is large, but large q means high complexity...
- Need to use small q, but than A is likely to be non-invertible.
- What should we do?

q-ary linear codes - Naive solution

- Decode the linear combinations over the reals, i.e. decode

$$\lambda_k = \sum_{l=1}^{L} a_{kl}\mathbf{x}_l$$

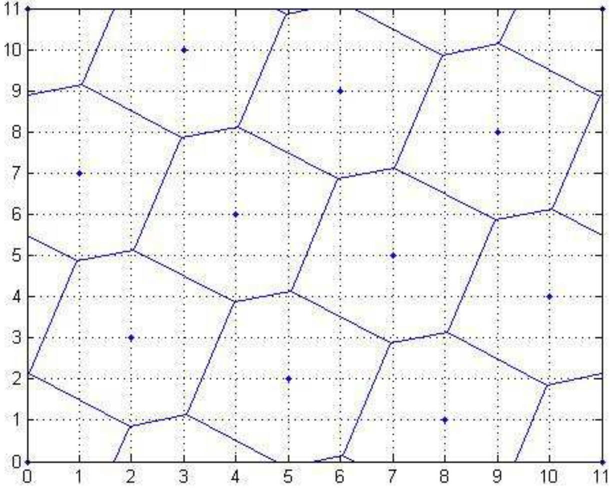
rather than

$$\mathbf{u}_k = \left[\sum_{l=1}^{K} a_{kl}\mathbf{x}_l\right] \bmod q.$$

- Now A only has to be invertible over $\mathbb{R}$ - an easier restriction.
- Can be done in a two-step procedure - first decode $\mathbf{u}_k$ and than use it for estimating λ_k.
- Results in the same error floor as in TCM.

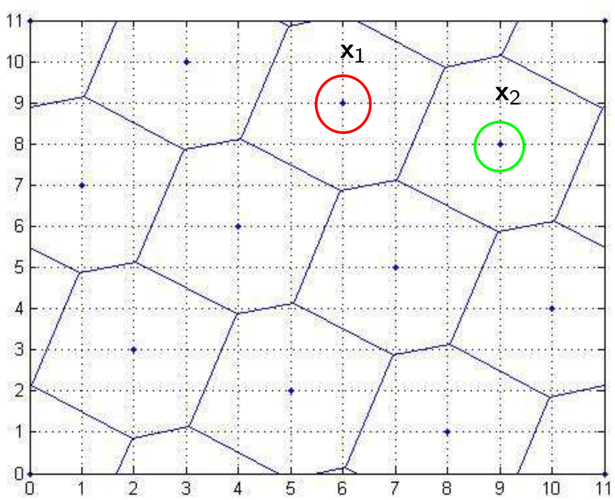
q-ary linear codes - Naive solution

Example: An 11-ary linear code



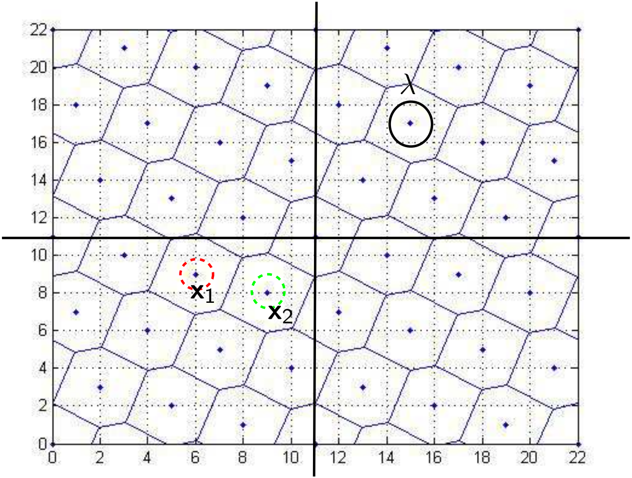
q-ary linear codes - Naive solution

Example: An 11-ary linear code

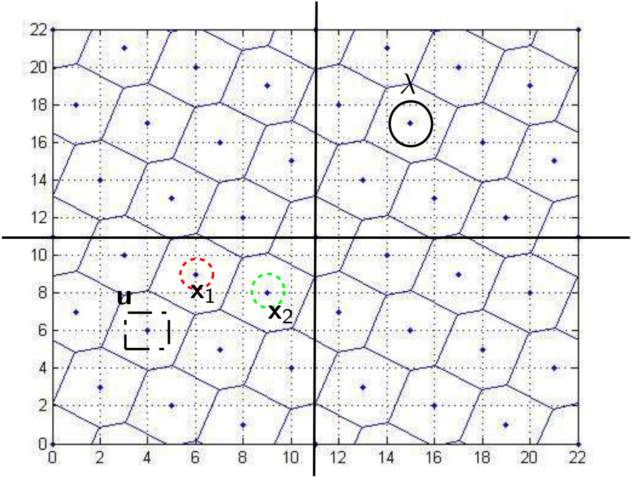


q-ary linear codes - Naive solution

$$\lambda = \mathbf{x}_1 + \mathbf{x}_2$$

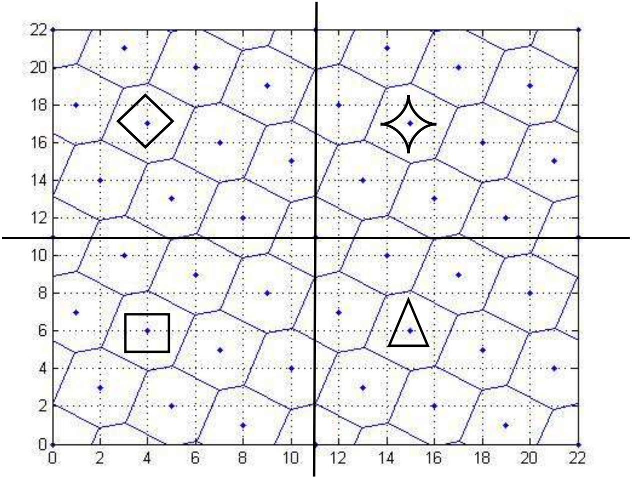


$$\mathbf{u} = [\lambda] \bmod {}_{11}$$



q-ary linear codes - Naive solution

Detecting the "uncoded bits" - An error floor is inevitible



q-ary linear codes - Preventing the error floor

- The centralized decoder has a set of equations over the reals, with some errors that result from the "uncoded bits"

$$\begin{pmatrix} \hat{\lambda}_1 \\ \hat{\lambda}_2 \\ \vdots \\ \hat{\lambda}_K \end{pmatrix} = \begin{pmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_K \end{pmatrix} + \begin{pmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \\ \vdots \\ \mathbf{e}_K \end{pmatrix} = AX$$

- Inverting the matrix A, rounding and reducing modulo q we have

$$\hat{X} = \left[X + \left[A^{-1} \begin{pmatrix} \mathbf{e}_1 \\ \mathbf{e}_2 \\ \vdots \\ \mathbf{e}_K \end{pmatrix} \right] \bmod q \right] \bmod q = [X + N] \bmod q$$

- We have a set of K DMCs.

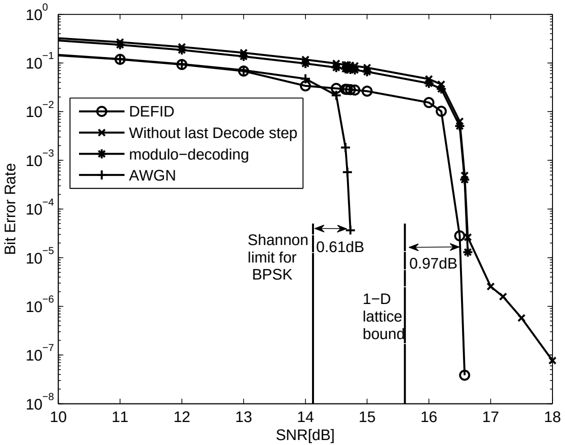
q-ary linear codes - Preventing the error floor

$$\hat{\mathbf{x}}_k = [\mathbf{x}_k + \mathbf{n}_k] \bmod q$$

- $\mathbf{n}_k$ is not zero only at locations where there was a detection error of the "uncoded bits" in one of the K relays.
- Should rarely happen if the "coded layer" was successfully decoded, the rate is not too small and the number of relays is not too big.
  $\Rightarrow$ The entropy of N_k is small.
- The q-ary linear codebook $\mathcal{C}$ should be good enough for the DMC.
  $\Rightarrow$ The error floor is prevented

Simulation results

- 2×2 Gaussian network. $\mathbf{h}_1 = [2/3\ 1/3]$, and $\mathbf{h}_2 = [0\ 1/3]$.
- Low SNR, binary LDPC code ($q = 2$).
- The relays decode the equations $\mathbf{a}_1 = [2\ 1]$, $\mathbf{a}_2 = [0\ 1]$.



Summary and conclusions

- We have proposed a simple q-ary implementation of Compute-and-Forward.
- Our implementation allows for small q while maintaining the weakest possible constraint on the invertibility of the set of integer coefficients.
- In the proposed scheme each relay decodes a linear combination over the reals.
- The crucial element in our scheme is an additional decode step which occurs at the centralized decoder.