The Compute-and-Forward Transform

Or Ordentlich, Uri Erez, Bobak Nazer

Tel Aviv University, Boston University

ISIT 2012

July 6, 2012
\[ \frac{1}{n} \| x_\ell \|^2 \leq \text{SNR}, \quad z \sim \mathcal{N}(0, I). \]
Theorem (Ahlswede ’71, Liao ’72)

The capacity region is the set of all rate pairs \((R_1, R_2)\) satisfying:

\[
R_1 < \frac{1}{2} \log(1 + h_1^2 \text{SNR}) \quad R_2 < \frac{1}{2} \log(1 + h_2^2 \text{SNR}) \\
R_1 + R_2 < \frac{1}{2} \log(1 + \|h\|^2 \text{SNR})
\]
Joint Typicality Decoding

$2^{nR_1}$ codewords

$2^{nR_2}$ codewords

$2^{n(R_1+R_2)}$ codewords
Successive Cancellation

\[ \frac{1}{n} \|x_\ell\|^2 \leq \text{SNR}, \quad z \sim \mathcal{N}(0, I). \]
Successive Cancellation

$$w_1 \rightarrow \mathcal{E}_1 \xrightarrow{x_1} h_1 \xrightarrow{z} y \xrightarrow{D} \hat{w}_1$$

$$w_2 \rightarrow \mathcal{E}_2 \xrightarrow{x_2} h_2 \xrightarrow{y} \hat{w}_2$$

$$\frac{1}{n} \| x_\ell \|^2 \leq \text{SNR}, \quad z \sim \mathcal{N}(0, I).$$

- Treat $x_2$ as noise and decode $x_1$, $R_1 < \frac{1}{2} \log \left( 1 + \frac{h_1^2 \text{SNR}}{1 + h_2^2 \text{SNR}} \right)$. 
Successive Cancellation

\[ \frac{1}{n} \| x_\ell \|^2 \leq \text{SNR}, \quad z \sim \mathcal{N}(0, I). \]

- **Treat** \( x_2 \) **as noise** and decode \( x_1 \), \( R_1 < \frac{1}{2} \log \left( 1 + \frac{h_1^2 \text{SNR}}{1 + h_2^2 \text{SNR}} \right) \).

- **Cancel** \( x_1 \) and decode \( x_2 \), \( R_2 < \frac{1}{2} \log \left( 1 + h_2^2 \text{SNR} \right) \).
Successive Cancellation

\[ \frac{1}{n} \| x_\ell \|^2 \leq \text{SNR}, \quad z \sim \mathcal{N}(0, I). \]

- Treat \( x_2 \) as noise and decode \( x_1 \), \( R_1 < \frac{1}{2} \log \left( 1 + \frac{h_1^2 \text{SNR}}{1 + h_2^2 \text{SNR}} \right) \).
- Cancel \( x_1 \) and decode \( x_2 \), \( R_2 < \frac{1}{2} \log \left( 1 + h_2^2 \text{SNR} \right) \).
- Switch decoding order for the other corner point.
Successive Cancellation

\[
\frac{1}{n} \|x_\ell\|^2 \leq \text{SNR}, \quad z \sim \mathcal{N}(0, I).
\]

- Treat \(x_2\) as noise and decode \(x_1\), \(R_1 < \frac{1}{2} \log \left(1 + \frac{h_1^2 \text{SNR}}{1 + h_2^2 \text{SNR}}\right)\).
- Cancel \(x_1\) and decode \(x_2\), \(R_2 < \frac{1}{2} \log (1 + h_2^2 \text{SNR})\).
- Switch decoding order for the other corner point.
- Achieves capacity when combined with time-sharing or rate-splitting (Rimoldi-Urbanke '96).
Compute-and-Forward

- Finite field messages: \( w_\ell \in \mathbb{F}_p^{k_\ell} \)
- Rates: \( R_\ell = \frac{k_\ell}{n} \log_2 p \)

- Decoder wants a linear combination of the messages with vanishing probability of error \( \lim_{n \to \infty} P(\hat{u} \neq u) = 0 \).

- Receiver uses its channel knowledge to match the equation coefficients \( a = [a_1 \ a_2]^T \) to the channel coefficients \( h = [h_1 \ h_2]^T \).
Compute-and-Forward: Effective Noise

\[ y = \sum_{\ell=1}^{K} h_{\ell} x_{\ell} + z \]

\[ = \sum_{\ell=1}^{K} a_{\ell} x_{\ell} + \sum_{\ell=1}^{K} (h_{\ell} - a_{\ell}) x_{\ell} + z \rightarrow a_{1}w_{1} \oplus a_{2}w_{2} \]

- How can we go between the integer combination of the real-valued codewords and the linear combination of the finite field messages?
- How do we cope with the self-noise?
- Use (dithered) nested lattice codes from *Erez-Zamir ’04.*
All users pick the same nested lattice code:
Choose messages over field $\mathbf{w}_\ell \in \mathbb{F}_q^k$:
Map $w_\ell$ to lattice point $t_\ell = \phi(w_\ell)$:
Transmit lattice points over the channel:

\[ w_1 \rightarrow \quad \begin{array}{cccccccc}
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\end{array} \]

\[ w_2 \rightarrow \quad \begin{array}{cccccccc}
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\
\end{array} \]

\[ h = [\ 1.4 \quad 2.1 \ ] \]

\[ a = [\ 2 \quad 3 \ ] \]
Transmit lattice points over the channel:

\[ w_1 \rightarrow \begin{array}{c}
\cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot \\
\end{array} \]

\[ w_2 \rightarrow \begin{array}{c}
\cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot \\
\cdot & \cdot & \cdot \\
\end{array} \]

\[ x_1 \rightarrow z \rightarrow y \]

\[ h = \begin{bmatrix} 1.4 & 2.1 \end{bmatrix} \]

\[ a = \begin{bmatrix} 2 & 3 \end{bmatrix} \]
Lattice codewords are scaled by channel coefficients:

\[ h = \begin{bmatrix} 1.4 & 2.1 \end{bmatrix} \]

\[ a = \begin{bmatrix} 2 & 3 \end{bmatrix} \]
Scaled codewords added together plus noise:

\[ \mathbf{h} = \begin{bmatrix} 1.4 & 2.1 \end{bmatrix} \]

\[ \mathbf{a} = \begin{bmatrix} 2 & 3 \end{bmatrix} \]
Scaled codewords added together plus noise:

\[ w_1 \quad w_2 \]

\[ x_1 \quad h_1 \quad z \quad x_2 \quad h_2 \]

\[ h = \begin{bmatrix} 1.4 & 2.1 \end{bmatrix} \]

\[ a = \begin{bmatrix} 2 & 3 \end{bmatrix} \]
Extra noise penalty for non-integer channel coefficients:

\[ w_1 \rightarrow \begin{array}{cc}
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\end{array} \]

\[ w_2 \rightarrow \begin{array}{cc}
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\cdot & \cdot \\
\end{array} \]

\[ x_1 \rightarrow z \rightarrow x_2 \]

\[ h_1 \rightarrow y \rightarrow h_2 \]

\[ h = [ 1.4 \ 2.1 ] \]

\[ a = [ 2 \ 3 ] \]

Effective noise: \( 1 + \text{SNR}\| h - a \|^2 \)
Scale output by $\alpha$ to reduce non-integer noise penalty:

$$\alpha \mathbf{h} = [ \alpha 1.4 \quad \alpha 2.1 ]$$

$$\mathbf{a} = [ 2 \quad 3 ]$$

Effective noise: $\alpha^2 + \text{SNR} \| \alpha \mathbf{h} - \mathbf{a} \|^2$
Scale output by $\alpha$ to reduce non-integer noise penalty:

$$\alpha h = \begin{bmatrix} \alpha 1.4 & \alpha 2.1 \end{bmatrix}$$

$$a = \begin{bmatrix} 2 & 3 \end{bmatrix}$$

Effective noise: $\alpha^2 + \text{SNR}\|\alpha h - a\|^2$
Decode to closest lattice point:

$$\alpha h = \begin{bmatrix} \alpha 1.4 & \alpha 2.1 \end{bmatrix}$$

$$a = \begin{bmatrix} 2 & 3 \end{bmatrix}$$

Effective noise: $$\alpha^2 + \text{SNR} \| \alpha h - a \|^2$$
Compute-and-Forward: Illustration

Compute sum of lattice points modulo the coarse lattice:

\[ \alpha h = \begin{bmatrix} \alpha 1.4 & \alpha 2.1 \end{bmatrix} \]

\[ a = \begin{bmatrix} 2 & 3 \end{bmatrix} \]

Effective noise: \[ \alpha^2 + \text{SNR} \| \alpha h - a \|^2 \]
Map back to equation of message symbols over the field:

\[
\alpha h = \begin{bmatrix} \alpha_{1.4} & \alpha_{2.1} \end{bmatrix}
\]

\[
a = \begin{bmatrix} 2 & 3 \end{bmatrix}
\]

Effective noise: \( \alpha^2 + \text{SNR} \| \alpha h - a \|^2 \)
Theorem (Nazer-Gastpar ’11)

Achievable rate for decoding linear combination with coefficients $a$ from a MAC with coefficients $h$:

$$R_{comp}(h, a) = \max_{\alpha \in \mathbb{R}} \frac{1}{2} \log^+ \left( \frac{\text{SNR}}{\alpha^2 + \text{SNR} \| \alpha h - a \|^2} \right)$$
Compute-and-Forward: Achievable Rates

**Theorem (Nazer-Gastpar ’11)**

Achievable rate for decoding linear combination with coefficients \( a \) from a MAC with coefficients \( h \):

\[
R_{\text{comp}}(h, a) = \frac{1}{2} \log^+ \left( \frac{1}{a^T (I + \text{SNR} \ h h^T)^{-1} a} \right)
\]
Theorem (Nazer-Gastpar ’11)

Achievable rate for decoding linear combination with coefficients $a$ from a MAC with coefficients $h$:

$$R_{\text{comp}}(h, a) = \frac{1}{2} \log^+ \left( \frac{1}{a^T (I + \text{SNR} \ h h^T)^{-1} a} \right)$$

- Channel vector $h = [1 \ g]$.
- Plot maximum computation rate normalized by MAC sum capacity $\frac{1}{2} \log(1 + \|h\|^2 \text{SNR})$. 
Theorem (Nazer-Gastpar ’11)

Achievable rate for decoding linear combination with coefficients $a$ from a MAC with coefficients $h$:

$$R_{comp}(h, a) = \frac{1}{2} \log^+ \left( \frac{1}{a^T (I + \text{SNR} \; hh^T)^{-1} a} \right)$$

- Channel vector $h = [1 \; g]$.
- Plot maximum computation rate normalized by MAC sum capacity $\frac{1}{2} \log(1 + ||h||^2 \text{SNR})$. 
Theorem (Nazer-Gastpar ’11)

Achievable rate for decoding linear combination with coefficients $a$ from a MAC with coefficients $h$:

$$ R_{\text{comp}}(h, a) = \frac{1}{2} \log^+ \left( \frac{1}{a^T \left( I + \text{SNR} \ h h^T \right)^{-1} a} \right) $$

- Channel vector $h = [1 \ g]$.
- Plot maximum computation rate normalized by MAC sum capacity $\frac{1}{2} \log(1 + \|h\|^2 \text{SNR})$. 
Two Linear Combinations

\[ \hat{u}_1 = a_{11}w_1 \oplus a_{12}w_2 \]
\[ \hat{u}_2 = a_{21}w_1 \oplus a_{22}w_2 \]

- Decode two linearly independent equations.
Two Linear Combinations

\[ w_1 \rightarrow \mathcal{E}_1 \rightarrow x_1 \rightarrow h_1 \rightarrow z \]
\[ w_2 \rightarrow \mathcal{E}_2 \rightarrow x_2 \rightarrow h_2 \rightarrow y \rightarrow \hat{u}_1 \rightarrow \hat{u}_2 \]

- Decode two linearly independent equations.

\[
\begin{align*}
    u_1 &= a_{11}w_1 \oplus a_{12}w_2 \\
    u_2 &= a_{21}w_1 \oplus a_{22}w_2
\end{align*}
\]

- Crossgain \( g \)

- Normalized Computation Rate

![Graph showing the relationship between crossgain and normalized computation rate]
Two Linear Combinations

\[ w_1 \rightarrow E_1 \xrightarrow{x_1} h_1 \rightarrow z \]
\[ w_2 \rightarrow E_2 \xrightarrow{x_2} h_2 \rightarrow y \]

\[ u_1 = a_{11} w_1 \oplus a_{12} w_2 \]
\[ u_2 = a_{21} w_1 \oplus a_{22} w_2 \]

- Decode two linearly independent equations.

```
0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
0 0.2 0.4 0.6 0.8 1
```

Crossgain $g$

Normalized Computation Rate

1st Equation

2nd Equation
Two Linear Combinations

\[ \mathbf{u}_1 = a_{11} \mathbf{w}_1 \oplus a_{12} \mathbf{w}_2 \]
\[ \mathbf{u}_2 = a_{21} \mathbf{w}_1 \oplus a_{22} \mathbf{w}_2 \]

- Decode two linearly independent equations.
Two Linear Combinations

\[ \mathbf{w}_1 \rightarrow \mathcal{E}_1 \xrightarrow{x_1} h_1 \rightarrow z \rightarrow y \rightarrow \mathcal{D} \xrightarrow{\mathbf{u}_1, \mathbf{u}_2} \]

\[ \mathbf{w}_2 \rightarrow \mathcal{E}_2 \xrightarrow{x_2} h_2 \rightarrow y \rightarrow \mathcal{D} \xrightarrow{\mathbf{u}_1, \mathbf{u}_2} \]

Decoding two linearly independent equations.

\[ \mathbf{u}_1 = a_{11} \mathbf{w}_1 \oplus a_{12} \mathbf{w}_2 \]

\[ \mathbf{u}_2 = a_{21} \mathbf{w}_1 \oplus a_{22} \mathbf{w}_2 \]

![Graph showing normalized computation rate vs. crossgain g with lines for Sum Rate, 1st Equation, and 2nd Equation.](image-url)
Two Linear Combinations

\[ w_1 \rightarrow E_1 \rightarrow x_1 \rightarrow h_1 \rightarrow z \rightarrow y \rightarrow D \rightarrow \hat{u}_1 \rightarrow \hat{u}_2 \]

\[ w_2 \rightarrow E_2 \rightarrow x_2 \rightarrow h_2 \rightarrow y \rightarrow D \rightarrow \hat{u}_1 \rightarrow \hat{u}_2 \]

- Decode two linearly independent equations.

\[
\begin{align*}
\mathbf{u}_1 &= a_{11} \mathbf{w}_1 \oplus a_{12} \mathbf{w}_2 \\
\mathbf{u}_2 &= a_{21} \mathbf{w}_1 \oplus a_{22} \mathbf{w}_2
\end{align*}
\]
• Looks as if the sum of computation rates is nearly equal to the MAC sum capacity. Why is this happening?

• Let $F = (I + \text{SNR } hh^T)^{-1/2}$. Then, each computation rate can be written as

$$R_{\text{comp}}(h, a_k) = \frac{1}{2} \log^+ \left( \frac{1}{\|F a_k\|^2} \right).$$

• Thus, decoding the best linear combinations is the same as finding the successive minima $\lambda_k(F)$ for the lattice $\Lambda(F) = F \mathbb{Z}^K$:

$$\lambda_k(F) \triangleq \inf \left\{ r : \dim \left( \text{span} \left( \Lambda(F) \cap B(0, r) \right) \right) \geq k \right\}$$
Successive Minima
Successive Minima
Successive Minima
Successive Minima
Successive Minima
Minkowski’s Theorem on Successive Minima

**Theorem (Minkowski)**

Let $\Lambda(F)$ be a lattice spanned by a full-rank $K \times K$ matrix $F$. Its successive minima $\lambda_k(F)$ satisfy

$$\prod_{k=1}^{K} \lambda_k^2(F) \leq K^K |\det(F)|^2.$$
Minkowski’s Theorem on Successive Minima

**Theorem (Minkowski)**

Let $\Lambda(F)$ be a lattice spanned by a full-rank $K \times K$ matrix $F$. Its successive minima $\lambda_k(F)$ satisfy

$$\prod_{k=1}^{K} \lambda_k^2(F) \leq K^K |\det(F)|^2.$$ 

**Theorem**

The sum of the $K$ best linearly independent computation rates satisfies

$$\sum_{k=1}^{K} R_{\text{comp}}(h, a_k) \geq \frac{1}{2} \log(1 + \|h\|^2 \text{SNR}) - \frac{K}{2} \log K.$$
Operational Interpretation: Multiple-Access

\[ u_1 = a_{11}w_1 \oplus a_{12}w_2 \]
\[ u_2 = a_{21}w_1 \oplus a_{22}w_2 \]

- Associate the rate of each equation to one message.
Operational Interpretation: Multiple-Access

\[ \begin{align*}
\mathbf{w}_1 &\rightarrow \mathcal{E}_1 \quad \mathbf{x}_1 \quad h_1 \\
\mathbf{w}_2 &\rightarrow \mathcal{E}_2 \quad \mathbf{x}_2 \quad h_2
\end{align*} \]

\[ z = h_1 \mathbf{x}_1 + h_2 \mathbf{x}_2 \]

\[ y = z \]

\[ \mathcal{D} \] (Decoding)

\[ \begin{align*}
\hat{\mathbf{u}}_1 &= a_{11} \mathbf{w}_1 \oplus a_{12} \mathbf{w}_2 \\
\hat{\mathbf{u}}_2 &= a_{21} \mathbf{w}_1 \oplus a_{22} \mathbf{w}_2
\end{align*} \]

- Associate the rate of each equation to one message.

- Decoding first equation: Succeeds since \( \max(R_1, R_2) < R_{\text{comp}, 1} \).
Operational Interpretation: Multiple-Access

\[
\begin{align*}
\mathbf{w}_1 &\rightarrow \mathcal{E}_1 \xrightarrow{\mathbf{x}_1} h_1 \xrightarrow{z} y \xrightarrow{\mathcal{D}} \hat{\mathbf{u}}_1 \hat{\mathbf{u}}_2 \\
\mathbf{w}_2 &\rightarrow \mathcal{E}_2 \xrightarrow{\mathbf{x}_2} h_2 \\
\mathbf{u}_1 &= a_{11}\mathbf{w}_1 \oplus a_{12}\mathbf{w}_2 \\
\mathbf{u}_2 &= a_{21}\mathbf{w}_1 \oplus a_{22}\mathbf{w}_2
\end{align*}
\]

- Associate the rate of each equation to one message.

- Decoding first equation: Succeeds since \(\max(R_1, R_2) < R_{\text{comp,1}}\).

- Decoding second equation runs into an issue: \(R_1 > R_{\text{comp,2}}\).
After decoding the first equation, the receiver knows

$$v_1 = [a_{11}t_1 + a_{12}t_2] \mod \Lambda.$$
After decoding the first equation, the receiver knows

\[ v_1 = [a_{11}t_1 + a_{12}t_2] \mod \Lambda. \]

The effective channel for the second equation is

\[ s_2 = [a_{21}t_1 + a_{22}t_2 + z_{\text{effec}}(h, a_2)] \mod \Lambda. \]
(Algebraic) Successive Cancellation

- After decoding the first equation, the receiver knows
  \[ \mathbf{v}_1 = [a_{11}t_1 + a_{12}t_2] \mod \Lambda. \]

- The effective channel for the second equation is
  \[ \mathbf{s}_2 = [a_{21}t_1 + a_{22}t_2 + z_{\text{effec}}(h, a_2)] \mod \Lambda. \]

- Using \( \mathbf{v}_1 \) we can cancel out \( t_1 \) from \( \mathbf{s}_2 \) without changing the effective noise.
  \[ \mathbf{s}_2^{SI} = [\mathbf{s}_2 - b_1 \mathbf{v}_1] \mod \Lambda \]
  \[ = [(a_{22} - b_1 a_{12})t_2 + z_{\text{effec}}(h, a_2)] \mod \Lambda. \]
(Algebraic) Successive Cancellation

- After decoding the first equation, the receiver knows

\[ v_1 = [a_{11}t_1 + a_{12}t_2] \mod \Lambda. \]

- The effective channel for the second equation is

\[ s_2 = [a_{21}t_1 + a_{22}t_2 + z_{\text{effec}}(h, a_2)] \mod \Lambda. \]

- Using \( v_1 \) we can cancel out \( t_1 \) from \( s_2 \) without changing the effective noise.

\[
 s_2^{S1} = [s_2 - b_1v_1] \mod \Lambda \\
= [(a_{22} - b_1a_{12})t_2 + z_{\text{effec}}(h, a_2)] \mod \Lambda.
\]

- Now, the receiver can decode since \( R_2 < R_{\text{comp}, 2} \).
Multiple-Access via Computation

\[ \begin{align*}
  w_1 &\rightarrow E_1 \quad x_1 \quad h_1 \quad \rightarrow \quad z \\
  w_2 &\rightarrow E_2 \quad x_2 \quad h_2 \quad \rightarrow \\
  \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \rightarrow \\
  \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \quad \rightarrow \\
  \hat{w}_1 &\quad \hat{w}_2 \\
\end{align*} \]
Multiple-Access via Computation

- **Successive cancellation** (without time-sharing or rate-splitting) achieves corner points.
• **Successive cancellation** (without time-sharing or rate-splitting) achieves corner points.
Successive cancellation (without time-sharing or rate-splitting) achieves corner points.
Successive cancellation (without time-sharing or rate-splitting) achieves corner points.

The compute-and-forward transform achieves another set of points near the sum rate boundary. Often closer to the symmetric capacity.
• **Successive cancellation** (without time-sharing or rate-splitting) achieves corner points.

• The **compute-and-forward transform** achieves another set of points near the sum rate boundary. Often closer to the symmetric capacity.
• **Successive cancellation** (without time-sharing or rate-splitting) achieves corner points.

• The **compute-and-forward transform** achieves another set of points near the sum rate boundary. Often closer to the symmetric capacity.

• Achieves $1/K$ DoF for almost all channel configurations. Proved using a strengthening of a DoF bound on compute-and-forward by **Niesen-Whiting ’11**.
Each transmitter wants to send a message to a single receiver.

Possibility of interference alignment Cadambe-Jafar ’08, Motahari et al. ’09.

Approximate capacity known in some special cases: two-user Etkin-Tse-Wang ’08, many-to-one and one-to-many Bresler-Parekh-Tse ’10, cyclic Zhou-Yu ’10.
Each transmitter wants to send a message to a single receiver.

Possibility of interference alignment Cadambe-Jafar ’08, Motahari et al. ’09.

Approximate capacity known in some special cases: two-user Etkin-Tse-Wang ’08, many-to-one and one-to-many Bresler-Parekh-Tse ’10, cyclic Zhou-Yu ’10.

Focus on the special case of symmetric cross-gains.
Lattice codes can enable alignment on the signal scale.

Each receiver sees an effective two-user multiple-access channel,

\[ y_k = x_k + g \sum_{\ell \neq l} x_\ell + z_k. \]

Idea: Successive cancellation. Decode and subtract interference \( \sum_{\ell \neq l} x_\ell \) before going after desired message.

Only optimal when the interference is very strong, Sridharan et al. ’08.

With the compute-and-forward transform we can approximate the sum capacity in all regimes.
Symmetric $K$-User Interference Channel

20dB

Sum-rate [bits/channel use] vs. $g$
Symmetric $K$-User Interference Channel

35dB

Sum-rate [bits/channel use] vs $g$ at 35dB
Symmetric $K$-User Interference Channel

50dB

Sum-rate [bits/channel use] vs $g$ for different SNR levels.
65 dB

Symmetric $K$-User Interference Channel

$\sum\text{-rate}[\text{bits/channel use}]$

$g$

$10^{-2}$ $10^{0}$ $10^{2}$
• **Basic Idea:** After decoding the first equation with coefficients $a$, we can create a new effective channel with coefficients $h + \beta a$ to make it easier to decode the second equation.
Decoding Multiple Equations: Can we do better?

- **Basic Idea:** After decoding the first equation with coefficients $\mathbf{a}$, we can create a new effective channel with coefficients $\mathbf{h} + \beta \mathbf{a}$ to make it easier to decode the second equation.

- From another perspective, effective noise is often correlated across linear combinations.
Decoding Multiple Equations: Can we do better?

- **Basic Idea:** After decoding the first equation with coefficients $a$, we can create a new effective channel with coefficients $h + \beta a$ to make it easier to decode the second equation.

- From another perspective, effective noise is often correlated across linear combinations.

- We need the real sum of codewords $\sum_{\ell} a_{\ell} x_{\ell}$. 
Decoding Multiple Equations: Can we do better?

- **Basic Idea:** After decoding the first equation with coefficients $a$, we can create a new effective channel with coefficients $h + \beta a$ to make it easier to decode the second equation.

- From another perspective, effective noise is often correlated across linear combinations.

- We need the real sum of codewords $\sum_{\ell} a_\ell x_\ell$.

**Lemma (Nazer IZS ’12)**

*In the original compute-and-forward framework, if you can recover the modulo sum, you can also recover the real sum (with high probability).*
Successive Cancellation

• Receiver observes $y = \sum_{\ell=1}^{L} h_{\ell} x_{\ell} + z$

Successive cancellation:

• Decode $x_{i}$.

• Calculate $y - h_{i} x_{i}$.

• Receiver now has

$$\sum_{\ell \neq i} h_{\ell} x_{\ell} + z$$
Successive Cancellation Computation

- Receiver observes \( y = \sum_{\ell=1}^{L} h_\ell x_\ell + z \)

Successive cancellation:

- Decode \( x_i \).
- Calculate \( y - h_i x_i \).
- Receiver now has \( \sum_{\ell \neq i} h_\ell x_\ell + z \)

Successive computation:

- Decode \( \sum_{\ell=1}^{K} a_\ell x_\ell \).
- Calculate \( y + \beta \sum_{\ell=1}^{K} a_\ell x_\ell \).
- Receiver now has \( \sum_{\ell=1}^{K} (h_\ell + \beta a_\ell) x_\ell + z \)
Successive Computation

Theorem (Nazer IZS ’12)

The equations with coefficients $a_1$ and $a_2$ can be decoded if

$$R < \frac{1}{2} \log^+ \left( \left( \|a_1\|^2 - \frac{\text{SNR}(h^T a_1)^2}{1 + \text{SNR} \|h\|^2} \right)^{-1} \right)$$

$$R < \frac{1}{2} \log^+ \left( \left( \|a_2\|^2 - \frac{(a_1^T a_2)^2}{\|a_1\|^2} - \frac{\text{SNR} \left( (h - \frac{a_1^T h}{\|a_1\|^2} a_1) T a_2 \right)^2}{1 + \text{SNR} \left( \|h\|^2 - \frac{(a_1^T h)^2}{\|a_1\|^2} \right)} \right)^{-1} \right)$$

- **Geometric intuition**: Remove the contribution of the first equation $s_1$ from the channel output $y$ to get $y_{\perp}$. Create a new effective channel by summing the projections of $s_1$ and $y_{\perp}$ onto the desired equation.
Multiple-Access via Successive Computation

\[ w_1 \xrightarrow{\mathcal{E}_1} x_1 h_1 z \]
\[ w_2 \xrightarrow{\mathcal{E}_2} x_2 h_2 y \]

\[ \hat{w}_1 \xrightarrow{D} \hat{w}_2 \]

Diagram showing the multiple-access via successive computation process with variables and equations.
Multiple-Access via Successive Computation

- **Successive compute-and-forward** can often attain the exact MAC sum capacity.
Multiple-Access via Successive Computation

- **Successive compute-and-forward** can often attain the exact MAC sum capacity.

- But not always...
Successive compute-and-forward can often attain the exact MAC sum capacity.

But not always... working on a clean characterization.
Conclusions

• Connection between sum of computation rates and the multiple-access sum capacity.

• Another perspective on multiple-access communication.

• Story also holds in the MIMO case, i.e., integer-forcing.

• Useful for lattice interference alignment. Full paper on arXiv.

• Looking for more applications.