Design of LDPC Codes for the Two-User Gaussian Multiple Access Channel

Alexios Balatsoukas-Stimming

Technical University of Crete
Department of Electronic and Computer Engineering
Telecommunications Laboratory

June 25, 2012
Outline

1. Introduction
   - Motivation
   - Previous Work
   - LDPC Codes

2. Single User Asymptotic Analysis
   - MAP Decoding
   - Belief Propagation
   - Density Evolution
   - Gaussian Approximation
   - EXIT Charts
   - Code Design Using EXIT Charts

3. Two-User Asymptotic Analysis
   - MAP Decoding
   - Gaussian Approximation
   - EXIT Charts
   - Code Design Using EXIT Charts
   - Results
Motivation

- Interesting information and coding theoretic problem.

- **Application to relay channel**: Decode and Forward (DaF) relaying consists of two phases:
  1. Broadcast (BC) phase.
  2. Multiple Access (MAC) phase.

- Under certain conditions, DaF can achieve higher rate when the source and the relay transmit uncorrelated information in the MAC phase.
  - Unequal power two-user Multiple Access Channel.
Density Evolution based design has been considered by Amraoui et al. (2002).
- High computational complexity.

EXIT chart based design has been considered by Roumy & Declercq (2007).
- Constrained to the case where both users have equal power.

We generalize their approach to arbitrary user powers.
An LDPC code $\mathcal{C}$ is defined as the nullspace of an $m \times n$ \textbf{sparse} parity-check matrix $H$:

$$\mathcal{C} = \{c \in \{0, 1\}^n : Hc = 0\}.$$

An LDPC code can be associated with a bipartite graph, called a \textbf{Tanner graph}.

- \textbf{Variable nodes} correspond to codeword bits and \textbf{check nodes} correspond to the parity-check equations enforced by the code.

- Variable node $j$ is connected with check node $i$ if and only if $H_{ij} = 1$. 
Example ((7, 4) Hamming Code)

\[ H = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 & 1 & 1 \end{bmatrix} \]

\[ L(x) = \frac{3}{7}x + \frac{3}{7}x^2 + \frac{1}{7}x^3 \]
\[ \lambda(x) = \frac{3}{12} + \frac{6}{12}x + \frac{3}{12}x^2 \]
\[ \rho(x) = x^3 \]
LDPC Codes

- Variable node degree distribution from a **node perspective**:
  \[ L(x) = \sum_{i} L_i x^i \]

- Variable node degree distribution from an **edge perspective**:
  \[ \lambda(x) = \sum_{i} \lambda_i x^{i-1} \]

- Check node degree distribution from an **edge perspective**:
  \[ \rho(x) = \sum_{i} \rho_i x^{i-1} \]

- Code **design rate**:
  \[ R(\lambda, \rho) \triangleq 1 - \frac{\sum_i \rho_i / i}{\sum_i \lambda_i / i}. \]
1 Introduction
   - Motivation
   - Previous Work
   - LDPC Codes

2 Single User Asymptotic Analysis
   - MAP Decoding
   - Belief Propagation
   - Density Evolution
   - Gaussian Approximation
   - EXIT Charts
   - Code Design Using EXIT Charts

3 Two-User Asymptotic Analysis
   - MAP Decoding
   - Gaussian Approximation
   - EXIT Charts
   - Code Design Using EXIT Charts
   - Results
BPSK modulation of codeword $c \in C$:

$$x = 1 - 2c.$$ 

Transmission of $x$ over a memoryless AWGN channel:

$$y = x + w, \quad w \sim \mathcal{N}(0, \sigma_w^2 I).$$

$$n \sim \mathcal{N}(0, \sigma_w^2)$$

Figure: AWGN channel.

Low SNR regime: capacity is near unity, binary coding suffices.
MAP Decoding

- MAP estimate for codeword bit $i$:

$$\hat{x}_i = \arg \max_{x_i} p(x_i | y)$$

$$= \ldots \quad \text{(Bayes’ rule, conditional independence)}$$

$$= \arg \max_{x_i} \sum_{x_i} \left( \prod_{k=1}^{n} p(y_k | x_k) \right) \mathbf{1}_{[c \in C]}.$$

- In many cases, Belief Propagation provides a very good approximation of the MAP decoding rule.
Belief Propagation Decoding

- Iterative message-passing decoder.

- Messages are in Log-Likelihood Ratio (LLR) form:

  $$m = \log \frac{p(x_i = +1|\tilde{y})}{p(x_i = -1|\tilde{y})},$$

  where $\tilde{y}$ is a RV which represents the information incorporated into $m$.

- Ideally, we would like the messages to evolve to:

  $$m = \log \frac{p(x_i = +1|y)}{p(x_i = -1|y)}, \forall i = 1, \ldots, n.$$
Belief Propagation Decoding

\[ m = \sum_{k=1}^{i-1} m_k \]

(a) Variable node update rule.

\[ m = 2 \tanh^{-1} \left( \prod_{k=1}^{i-1} \tanh \left( \frac{m_k}{2} \right) \right) \]

(b) Check node update rule.
Belief Propagation Decoding - Initialization

- $p(x_i)$ and $p(y_i)$ are known. So, given $\log \frac{p(y_i | +1)}{p(y_i | -1)}$, we can calculate $\log \frac{p(+1 | y_i)}{p(-1 | y_i)}$. 

\[
\log \frac{p(+1 | y_i)}{p(-1 | y_i)}
\]
Belief Propagation Decoding

If graph is cycle-free, after running BP: 
\[
\log \frac{p(x_i=+1|y)}{p(x_i=-1|y)}
\]

Otherwise, in many cases, BP provides a very good approximation of MAP decoding.
Belief Propagation Decoding

In practice, decoding proceeds in rounds:

Figure: Initialization.
Belief Propagation Decoding

In practice, decoding proceeds in rounds:

![Diagram showing variable-to-check messages](image)

**Figure**: Variable-to-check messages.
In practice, decoding proceeds in rounds:

**Figure:** Check-to-variable messages.
In the limit of infinite blocklength:

1. **Cycle-free** BP is guaranteed, thus BP is equivalent to MAP decoding.

2. The performance of the \((\lambda, \rho)\)-ensemble of LDPC codes under Belief Propagation decoding can be predicted by **Density Evolution (DE)**.

3. The performance of individual codes in the ensemble is close to the ensemble average performance.
It turns out to be convenient to rewrite check node update rule:

\[ m = \gamma^{-1} \left( \sum_{k=1}^{i-1} \gamma(m_k) \right), \]

where \( \gamma(x) \triangleq (\gamma_1(x), \gamma_2(x)) \triangleq (\text{sign}(x), -\ln \tanh \left| \frac{x}{2} \right|) \).

Let \( \Gamma \) denote the density transformation corresponding to \( \gamma \):

\[ f_{\gamma(x)}(\gamma_1(x), \gamma_2(x)) = \Gamma(f_X(x)). \]

The range space of \( \Gamma \) is endowed with a convolution operator (\( \otimes \)).

If \( m_k \) are independent and distributed according to \( a_{m_k} \), the density of \( m \), denoted \( a_m \), is:

\[ a_m = \Gamma^{-1} \left( \Gamma(a_{m_1}) \otimes \ldots \otimes \Gamma(a_{m_{i-1}}) \right). \]
It is sufficient (and convenient) to track the message densities under the **all-one BPSK codeword assumption**.

Let $a_0$ denote the density of $\log \frac{p(y_i | x_i = +1)}{p(y_i | x_i = -1)}$, given $X_i = +1$.

$$a_1 = a_0 \ast (b_0 \ast b_0)$$

$$b_0 = \Gamma^{-1}(\Gamma(a_0) \otimes \Gamma(a_0))$$
Recursive formula:

\[ a_{\ell+1} = a_0 \ast (\Gamma^{-1}(\Gamma(a_\ell) \otimes \Gamma(a_\ell))) \ast \Gamma^{-1}(\Gamma(a_\ell) \otimes \Gamma(a_\ell))). \]
Recursive formula:

\[ a_{\ell+1} = a_0 \ast \left( \Gamma^{-1} \left( \Gamma(a_{\ell}) \otimes \Gamma(a_{\ell}) \right) \right) \ast \left( \Gamma^{-1} \left( \Gamma(a_{\ell}) \otimes \Gamma(a_{\ell}) \right) \right). \]
Density Evolution

- Recursive formula:

\[ a_{\ell+1} = a_0 \ast \left( \Gamma^{-1}(\Gamma(a_{\ell}) \otimes \Gamma(a_{\ell})) \ast \Gamma^{-1}(\Gamma(a_{\ell}) \otimes \Gamma(a_{\ell})) \right). \]
Density Evolution

- Recursive formula:

\[ a_{\ell+1} = a_0 \ast \left( \Gamma^{-1}(\Gamma(a_\ell) \otimes \Gamma(a_\ell)) \ast \Gamma^{-1}(\Gamma(a_\ell) \otimes \Gamma(a_\ell)) \right). \]

- More general: recursive formula for regular codes:

\[ a_{\ell+1} = a_0 \ast \left( \Gamma^{-1} \left( \Gamma(a_\ell) \otimes (d_c-1) \right) \ast (d_v-1) \right). \]

- Even more general: recursive formula for irregular codes:

\[ a_{\ell+1} = a_0 \ast \sum_i \lambda_i \left( \Gamma^{-1} \left( \sum_j \rho_j \Gamma(a_\ell) \otimes (j-1) \right) \ast (i-1) \right). \]
Density Evolution

- **Recursive formula:**

\[
a_{\ell+1} = a_0 \times \left( \Gamma^{-1} \left( \Gamma(a_{\ell}) \otimes \Gamma(a_{\ell}) \right) \right) \times \Gamma^{-1} \left( \Gamma(a_{\ell}) \otimes \Gamma(a_{\ell}) \right).
\]

- **More general:** recursive formula for regular codes:

\[
a_{\ell+1} = a_0 \times \left( \Gamma^{-1} \left( \Gamma(a_{\ell}) \otimes (d_c-1) \right) \right) \times ((d_v-1)).
\]

- **Even more general:** recursive formula for irregular codes:

\[
a_{\ell+1} = a_0 \times \sum \lambda_i \left( \Gamma^{-1} \left( \sum \rho_j \Gamma(a_{\ell}) \otimes (j-1) \right) \right) \times (i-1).
\]
The average probability of error at iteration $\ell$ is:

$$P_b^\ell(e) = \int_{-\infty}^{0} a_\ell(x) dx.$$ 

Condition for zero probability of error:

$$a_\ell \xrightarrow{\ell \to \infty} \Delta_\infty.$$ 

However, DE has very high computational complexity.
**Gaussian Approximation (GA):** we assume the messages are symmetric Gaussian random variables $\mathcal{N}(\mu, 2\mu)$ (Central Limit Theorem, experimental observations).

We must track **only the mean**.

**Variable node** of degree $i$ at iteration $\ell$:

\[
\mu_{v_i}^{(\ell)} = \mu_{ch} + \sum_{k=1}^{i-1} \mu_k^{(\ell-1)}
\]

\[
\mu_{v_i} = \mu_{ch} + (i - 1)\mu^{(\ell-1)}.
\]

**Average over $\lambda(x)$**:

\[
\mu_{v}^{(\ell)} = \sum_{i} \lambda_i \mu_{v_i}^{(\ell)}.
\]
Gaussian Approximation

- **Check node** of degree $i$ at iteration $\ell$:

\[
\mu_{u_i}^{(\ell)} = \phi^{-1} \left( 1 - \left(1 - \phi(\mu_{v}^{(\ell)})\right)^{i-1} \right)
\]

- $m = 2 \tanh^{-1} \left( \prod_{k=1}^{i-1} \tanh \left( \frac{m_k}{2} \right) \right)$

- **Average over $\rho(x)$**:

\[
\mu_u^{(\ell)} = \sum_i \rho_i \mu_{u_i}^{(\ell)}.
\]
We define:

\[ J(\sigma) = 1 - \frac{1}{\sqrt{2\pi\sigma}} \int_{-\infty}^{+\infty} e^{-\frac{(x-\sigma^2/2)^2}{2\sigma^2}} \log_2(1 + e^{-x}) \, dx. \]

\( J(\sigma) \) is the mutual information between a codeword bit and the corresponding message, assuming that the message is a symmetric Gaussian random variable with standard deviation \( \sigma \).

\( J^{-1}(I) \) is the standard deviation of the symmetric Gaussian distributed message which has mutual information \( I \) with a codeword bit.
$I_{EV}$ (resp. $I_{AV}$) is the average mutual information between the outgoing (resp. incoming) messages of variable nodes and the codeword bits.

The EXIT chart $I_{EV}$ describing the variable node function of degree $i$:

$$I_{EV}^i(I_{AV}, \sigma_{ch}^2) = J \left( \sqrt{(i - 1)J^{-1}(I_{AV})^2 + \sigma_{ch}^2} \right)$$

Average over $\lambda(x)$:

$$I_{EV}(I_{AV}, \sigma_{ch}^2) = \sum_i \lambda_i I_{EV}^i(I_{AV}, \sigma_{ch}^2),$$

with $\sigma_{ch}^2 = \frac{4}{\sigma_w^2}$, $\sigma_w^2$ is the noise variance.
EXIT Charts - Check Nodes

- $I_{EC}$ (resp. $I_{AC}$) is the average mutual information between the outgoing (resp. incoming) messages of check nodes and the codeword bits.

\[
m = 2 \tanh^{-1} \left( \prod_{k=1}^{i-1} \tanh \left( \frac{m_k}{2} \right) \right)
\]

- The EXIT chart $I_{EC}$ describing the check node function of degree $i$:

\[
l_{EC}^i(I_{AC}) \approx 1 - J \left( \sqrt{(i - 1)J^{-1}(1 - I_{AC})^2} \right)
\]

- Average over $\rho(x)$:

\[
l_{EC}(I_{AC}) \approx \sum_i \rho_i l_{EC}^i(I_{AC}).
\]
For every possible incoming average mutual information at variable nodes, we want the outgoing average mutual information to be larger:

\[ I_{EC}(I_{EV}(I_{AV})) > I_{AV} \]

Equivalently:

\[ I_{EV}(I_{AV}) > I_{EC}^{-1}(I_{AV}) \]

We assume \( \rho(x) \) is **concentrated**, i.e. \( \rho(x) = x^k, \ k \in \mathbb{N} \).

Then, the inverse of \( I_{EC} \), denoted \( I_{EC}^{-1} \) is:

\[ I_{EC}^{-1}(I_{AV}) \approx 1 - J \left( \frac{1}{\sqrt{k-1}} J^{-1}(1 - I_{AV}) \right) . \]
For given $\lambda(x)$ and $\rho(x)$, the code design rate is:

$$R(\lambda, \rho) = 1 - \frac{\sum_i \rho_i/i}{\sum_i \lambda_i/i}.$$  

A common approach is to maximize $R(\lambda, \rho)$ over $\lambda(x)$ for fixed $\rho(x)$. This gives rise to a continuous linear program:

$$\text{maximize}_{\lambda_i} \sum_{i=2}^{v_{\text{max}}} \lambda_i/i$$

subject to  

$$l_{EC}^{-1}(l_{AV}) < l_{EV}(l_{AV}) = \sum_{i=2}^{v_{\text{max}}} \lambda_i l_{EV}^i(l_{AV}), \quad \forall l_{AV} \in (0, 1)$$

$$\sum_{i=2}^{v_{\text{max}}} \lambda_i = 1, \quad \lambda_i \geq 0, \quad i = 2, 3, \ldots, v_{\text{max}}.$$  

We solve it by discretizing $l_{AV} \in (0, 1)$. 

1 Introduction
- Motivation
- Previous Work
- LDPC Codes

2 Single User Asymptotic Analysis
- MAP Decoding
- Belief Propagation
- Density Evolution
- Gaussian Approximation
- EXIT Charts
- Code Design Using EXIT Charts

3 Two-User Asymptotic Analysis
- MAP Decoding
- Gaussian Approximation
- EXIT Charts
- Code Design Using EXIT Charts
- Results
System Model

- Low SNR: maximal sum-rate is near unity, **binary coding suffices**.

- BPSK modulation of codewords $c[1] \in C_1$ and $c[2] \in C_2$:
  

- Let $P_1$ and $P_2$ denote the users’ **normalized** (wrt the noise variance) transmit powers.

$$y = \sqrt{P_1}x[1] + \sqrt{P_2}x[2] + w, \quad w \sim \mathcal{N}(0, I).$$
Two-user MAC capacity region:

\[
\begin{align*}
R^{[1]} & \leq C \left( \frac{P_1}{N} \right), \quad R^{[2]} \leq C \left( \frac{P_2}{N} \right), \\
R^{[1]} + R^{[2]} & \leq C \left( \frac{P_1 + P_2}{N} \right),
\end{align*}
\]

where \( C(x) = \frac{1}{2} \log_2(1 + x) \), \( P_i \) is the power of user \( i \), \( i = 1, 2 \), and \( N \) is the noise power.

We want to maximize the sum-rate, i.e. approach the dominant face.
MAP estimate for codeword bit $i$ of user 1:

$$\hat{x}_{i}^{[1]} = \arg \max_{x_{i}^{[1]}} p(x_{i}^{[1]} | y)$$

$$= \ldots \quad \text{(Bayes’ rule, conditional independence)}$$

$$= \arg \max_{x_{i}^{[1]}} \sum_{x_{i}^{[1]} \sim x_{i}^{[2]}} \sum_{x_{i}^{[2]}} \left( \prod_{k=1}^{n} p(y_{k} | x_{k}^{[1]}, x_{k}^{[2]}) \right) 1_{[c^{[1]} \in C_{1}]} 1_{[c^{[2]} \in C_{2}]}.$$ 

Analogous expression for codeword bit $i$ of user 2.
Two-User MAC Tanner Graph

**Asymptotically cycle-free**, like single user Tanner graph.
Variable-to-check, variable-to-state, and check-to-variable messages follow standard BP rules.

We need to derive the state node update rule:

\[ s_{v[1]}(y, v_{s[2]}) = \log \frac{e^{-\frac{(y-\sqrt{P_1}-\sqrt{P_2})^2}{2}} e^{vs[2]} + e^{-\frac{(y-\sqrt{P_1}+\sqrt{P_2})^2}{2}}}{e^{-\frac{(y+\sqrt{P_1}-\sqrt{P_2})^2}{2}} e^{vs[2]} + e^{-\frac{(y+\sqrt{P_1}+\sqrt{P_2})^2}{2}}} , \]

Analogous expression for messages towards user 2.
Belief Propagation Decoding

Figure: State-to-variable messages.
Belief Propagation Decoding

Figure: Variable-to-check messages.
Belief Propagation Decoding

![Diagram of Belief Propagation Decoding]

**Figure:** Check-to-variable messages.
Belief Propagation Decoding

Figure: Variable-to-state messages.
The all-one BPSK codeword assumption is not valid because the MAC channel is not output-symmetric w.r.t. each user:

\[ p(y|x^j) \neq p(-y| -x^j), \quad j = 1, 2. \]

Fortunately, we can restrict our analysis to the case where one user transmits the all-one BPSK codeword and the other user transmits a BPSK codeword of type one-half.

Proof is based on a typical codeword analysis and update rule symmetries.
Gaussian Approximation

- The mean of variable-to-check and check-to-variable messages follows **standard GA rules**.

- The mean of variable-to-state messages also follows standard GA rules, with the slight difference that averaging is done over $L(x)$, instead of $\lambda(x)$.

- For the mean of state-to-variable messages we need to consider **two types of state nodes**.

- One type is connected to two $+1$ variable nodes, while the other is connected to one $+1$ and one $-1$ variable node.
The update rules for the **mean** of the variable-to-state messages are:

\[
F_+^{[1]}(\mu) = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{+\infty} e^{-z^2} \log \left( \frac{1 + e^{\sqrt{4\mu + 8P_2z + \mu + 2P_2}}}{1 + e^{-\sqrt{4\mu + 8P_2z - \mu - 2P_2 - 4\sqrt{P_1\sqrt{P_2}}}}} \right) \, dz - \mu + 2(P_1 - P_2)
\]

\[
F_-^{[1]}(\mu) = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{+\infty} e^{-z^2} \log \left( \frac{1 + e^{-\sqrt{4\mu + 8P_2z - \mu - 2P_2}}}{1 + e^{\sqrt{4\mu + 8P_2z + \mu + 2P_2 - 4\sqrt{P_1\sqrt{P_2}}}}} \right) \, dz + \mu + 2 \left( \sqrt{P_1} - \sqrt{P_2} \right)^2
\]

\[
F_+^{[2]}(\mu) = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{+\infty} e^{-z^2} \log \left( \frac{1 + e^{\sqrt{4\mu + 8P_1z + \mu + 2P_1}}}{1 + e^{-\sqrt{4\mu + 8P_1z - \mu - 2P_1 - 4\sqrt{P_1\sqrt{P_2}}}}} \right) \, dz - \mu + 2(P_2 - P_1)
\]

\[
F_-^{[2]}(\mu) = \frac{1}{\sqrt{\pi}} \int_{-\infty}^{+\infty} e^{-z^2} \log \left( \frac{1 + e^{-\sqrt{4\mu + 8P_1z - \mu - 2P_1}}}{1 + e^{\sqrt{4\mu + 8P_1z + \mu + 2P_1 - 4\sqrt{P_1\sqrt{P_2}}}}} \right) \, dz - \mu - 2 \left( \sqrt{P_2} - \sqrt{P_1} \right)^2
\]
Variable-to-check, and check-to-variable messages follow standard EXIT chart rules.

For the variable-to-state messages at a degree $i$ variable node:

$$I_{EVS}^{[j],i}(I_{AV}^{[j]}) = J \left( \sqrt{i} J^{-1} \left( I_{AV}^{[j]} \right) \right).$$

Average over $L(x)$:

$$I_{EVS}^{[j]}(I_{AV}^{[j]}) = \sum_{i} L_i I_{EVS}^{[j],i}(I_{AV}^{[j]}).$$

The mutual information between the state-to-variable messages towards user 1 and this user's codeword bits is:

$$I_{ES}^{[1]}(I_{EVS}^{[2]}) = \frac{1}{2} J \left( \sqrt{2 F_+^{[1]} \left[ \frac{1}{2} J^{-1} \left( I_{EVS}^{[2]} \right)^2 \right]} \right) + \frac{1}{2} J \left( \sqrt{2 F_-^{[1]} \left[ \frac{1}{2} J^{-1} \left( I_{EVS}^{[2]} \right)^2 \right]} \right).$$
The overall EXIT chart for user 1 is:

\[ I^{[1]}_{EVC}(I^{[1]}_{AV}, I^{[2]}_{AV}) = \sum_i \lambda_i^{[1]} J \left( \sqrt{(i - 1)J^{-1} \left( I^{[1]}_{AV} \right)^2 + J^{-1} \left( I^{[1]}_{ES}(I^{[2]}_{AV}) \right)^2} \right). \]

An analogous expression holds for user 2.

If the inverse of \( I^{[j]}_{EC} \) lies below \( I^{[j]}_{EVC} \) for both users, then the probability of error of BP decoding becomes vanishingly small.

Again, we assume \( \rho^{[j]}(x) \) is concentrated, i.e. \( \rho^{[j]}(x) = x^{kj}, k_j \in \mathbb{N} \).
We fix the variable node degree distribution of one user and optimize the variable node degree distribution of the other user:

\[
\begin{align*}
\text{maximize} & \quad \sum_{i=1}^{v_{\text{max}}} \lambda_i[j]/i \\
\text{subject to} & \quad I_{EC}^{-1}[j](I_{AV}) < \sum_{i=1}^{v_{\text{max}}} \lambda_i[j]I_{EVC}(I_{AV}), \quad \forall I_{AV} \in (0,1), \\
& \quad \sum_{i=1}^{v_{\text{max}}} \lambda_i[j] = 1, \quad \lambda_i[j] \geq 0, \quad i = 2, 3, \ldots, v_{\text{max}}.
\end{align*}
\]

We proceed in an alternating fashion until convergence.

Repeated for several \((k_1, k_2)\), best result is kept.
Results

- $P_1 = 1.5$, $P_2 = 1$, $\nu_{\text{max}} = 200$.
- Degree distribution for user 1:

\[
\lambda^{[1]} = 0.2429x + 0.3595x^2 + 0.1433x^{21} + 0.0800x^{22} + 0.0631x^{97} + 0.1111x^{98}, \\
\rho^{[1]} = x^7.
\]

- Degree distribution for user 2:

\[
\lambda^{[2]} = 0.1853x + 0.2762x^2 + 0.0489x^{11} + 0.0705x^{12} + 0.0569x^{31} + 0.0567x^{32} + 0.3054x^{199}, \\
\rho^{[2]} = x^7.
\]
Results \( (P_1 = 1.5, \ P_2 = 1) \)

**Figure**: EXIT charts.
Results \((P_1 = 1.5, \ P_2 = 1)\)

**Figure:** Rate pair in capacity region and finite length simulation for \(n = 5 \cdot 10^4\).
Results \((P_1 = 1.5, \ P_2 = 1)\)

Figure: Convergence of alternating optimization.
Results \( (P_1 = 1.75, \ P_2 = 0.75) \)

**User 1 EXIT Chart**

**User 2 EXIT Chart**

**Figure:** EXIT charts.
Results \( (P_1 = 1.75, \ P_2 = 0.75) \)

**Figure:** Rate pair in capacity region and finite length simulation for \( n = 5 \cdot 10^4 \).
Results \( (P_1 = 1.75, \ P_2 = 0.75) \)

\[ \text{Figure: Convergence of alternating optimization.} \]
Questions?