Deep Weighted MMSE Downlink Beamforming

Lissy Pellaco, Mats Bengtsson, Joakim Jaldén

KTH Royal Institute of Technology, Stockholm, Sweden

IEEE ICASSP, 6-11 June, 2021
Introduction

- Multi-user multiple-input single-output (MU-MISO) interference downlink channel
Introduction

- Multi-user multiple-input single-output (MU-MISO) interference downlink channel
- Single base station with $M$ transmit antennas, $N$ single-antenna users

Received signal

$$y_i = h_i^H v_i x_i + \sum_{j=1, j \neq i}^{N} h_i^H v_j x_j + n_i$$

- $h_i$ ∈ $C^M$: channel
- $v_i$ ∈ $C^M$: beamformer vector
- $x_i$ ∼ $CN(0,1)$: transmitted symbol
- $n_i$ ∼ $CN(0, \sigma^2)$: noise

$V \equiv [v_1, v_2, ..., v_N]^T \in C^{N \times M}$: Beamformer matrix
Introduction

- Multi-user multiple-input single-output (MU-MISO) interference downlink channel
- Single base station with $M$ transmit antennas, $N$ single-antenna users
- Linear beamforming

Received signal $y_i = h_i h_i^T v_i x_i + \sum_{j=1, j \neq i}^{N} h_i h_i^T v_j x_j + n_i$

- $h_i \in \mathbb{C}^{M}$: channel
- $v_i \in \mathbb{C}^{M}$: beamformer vector
- $x_i \sim \mathcal{CN}(0, 1)$: transmitted symbol
- $n_i \sim \mathcal{CN}(0, \sigma^2)$: noise

Beamformer matrix $V = [v_1, v_2, ..., v_N]^T \in \mathbb{C}^{N \times M}$
Introduction

- Multi-user multiple-input single-output (MU-MISO) interference downlink channel
- Single base station with $M$ transmit antennas, $N$ single-antenna users
- Linear beamforming

\[ \text{Received signal} \quad y_i = h_i H v_i x_i + \sum_{j=1, j \neq i}^{N} h_i H v_j x_j + n_i \]

$h_i \in \mathbb{C}^M$: channel
$v_i \in \mathbb{C}^M$: beamformer vector
$x_i \sim \mathcal{CN}(0, 1)$: transmitted symbol
$n_i \sim \mathcal{CN}(0, \sigma^2)$: noise

Goals

$V \equiv [v_1, v_2, \ldots, v_N]^T \in \mathbb{C}^{N \times M}$: Beamformer matrix
Introduction

- Multi-user multiple-input single-output (MU-MISO) interference downlink channel
- Single base station with $M$ transmit antennas, $N$ single-antenna users
- Linear beamforming

**Goals**

1. Maximize **weighted sum rate** subject to transmit power constraint
Introduction

- Multi-user multiple-input single-output (MU-MISO) interference downlink channel
- Single base station with $M$ transmit antennas, $N$ single-antenna users
- Linear beamforming

Goals

1. Maximize **weighted sum rate** subject to transmit power constraint
2. Satisfy the **power consumption** and latency requirements at the base station

Received signal:

$$y_i = h_i^H v_i x_i + \sum_{j=1, j \neq i}^N h_i^H v_j x_j + n_i$$

$h_i \in \mathbb{C}^M$: channel
$v_i \in \mathbb{C}^M$: beamformer vector
$x_i \sim \mathcal{CN}(0, 1)$: transmitted symbol
$n_i \sim \mathcal{CN}(0, \sigma^2)$: noise

$V = [v_1, v_2, \ldots, v_N]^T \in \mathbb{C}^{N \times M}$: Beamformer matrix
Introduction

- Multi-user multiple-input single-output (MU-MISO) interference downlink channel
- Single base station with $M$ transmit antennas, $N$ single-antenna users
- Linear beamforming

**Received signal**

$$y_i = h_i^H v_i x_i + \sum_{j=1,j\neq i}^{N} h_i^H v_j x_j + n_i$$
Introduction

- Multi-user multiple-input single-output (MU-MISO) interference downlink channel
- Single base station with $M$ transmit antennas, $N$ single-antenna users
- Linear beamforming

Received signal
\[ y_i = h_i^H v_i x_i + \sum_{j=1,j\neq i}^N h_i^H v_j x_j + n_i \]

$h_i \in \mathbb{C}^M$: channel
Introduction

- Multi-user multiple-input single-output (MU-MISO) interference downlink channel
- Single base station with $M$ transmit antennas, $N$ single-antenna users
- Linear beamforming

Received signal

$$y_i = h_i^H v_i x_i + \sum_{j=1,j\neq i}^{N} h_j^H v_j x_j + n_i$$

$h_i \in \mathbb{C}^M$: channel
$v_i \in \mathbb{C}^M$: beamformer vector

$x_i \sim \mathcal{CN}(0, 1)$: transmitted symbol
$n_i \sim \mathcal{CN}(0, \sigma^2)$: noise
Introduction

- Multi-user multiple-input single-output (MU-MISO) interference downlink channel
- Single base station with $M$ transmit antennas, $N$ single-antenna users
- Linear beamforming

Revised signal

\[ y_i = h_i^H v_i x_i + \sum_{j=1, j \neq i}^N h_i^H v_j x_j + n_i \]

- $h_i \in \mathbb{C}^M$: channel
- $v_i \in \mathbb{C}^M$: beamformer vector
- $x_i \sim \mathcal{CN}(0, 1)$: transmitted symbol
Introduction

- Multi-user multiple-input single-output (MU-MISO) interference downlink channel
- Single base station with $M$ transmit antennas, $N$ single-antenna users
- Linear beamforming

Received signal

$$y_i = h_i^H v_i x_i + \sum_{j=1,j\neq i}^N h_i^H v_j x_j + n_i$$

$h_i \in \mathbb{C}^M$: channel

$v_i \in \mathbb{C}^M$: beamformer vector

$x_i \sim \mathcal{CN}(0, 1)$: transmitted symbol

$n_i \sim \mathcal{CN}(0, \sigma^2)$: noise
Introduction

- Multi-user multiple-input single-output (MU-MISO) interference downlink channel
- Single base station with $M$ transmit antennas, $N$ single-antenna users
- Linear beamforming

Received signal

$$y_i = h_i^H v_i x_i + \sum_{j=1,j\neq i}^{N} h_i^H v_j x_j + n_i$$

- $h_i \in \mathbb{C}^M$: channel
- $v_i \in \mathbb{C}^M$: beamformer vector
- $x_i \sim \mathcal{CN}(0, 1)$: transmitted symbol
- $n_i \sim \mathcal{CN}(0, \sigma^2)$: noise

$$V \triangleq [v_1, v_2, \ldots, v_N]^T \in \mathbb{C}^{N \times M}$$
Introduction

- Multi-user multiple-input single-output (MU-MISO) interference downlink channel
- Single base station with $M$ transmit antennas, $N$ single-antenna users
- Linear beamforming

Received signal

$$y_i = h_i^H v_i x_i + \sum_{j=1,j\neq i}^N h_j^H v_j x_j + n_i$$

$h_i \in \mathbb{C}^M$: channel
$v_i \in \mathbb{C}^M$: beamformer vector
$x_i \sim \mathcal{CN}(0, 1)$: transmitted symbol
$n_i \sim \mathcal{CN}(0, \sigma^2)$: noise

$$V \triangleq [v_1, v_2, \ldots, v_N]^T \in \mathbb{C}^{N \times M}$$
Introduction

- Multi-user multiple-input single-output (MU-MISO) interference downlink channel
- Single base station with $M$ transmit antennas, $N$ single-antenna users
- Linear beamforming

Received signal

$$y_i = h_i^H v_i x_i + \sum_{j=1,j\neq i}^{N} h_i^H v_j x_j + n_i$$

- $h_i \in \mathbb{C}^M$: channel
- $v_i \in \mathbb{C}^M$: beamformer vector
- $x_i \sim \mathcal{CN}(0, 1)$: transmitted symbol
- $n_i \sim \mathcal{CN}(0, \sigma^2)$: noise

$$V \triangleq [v_1, v_2, \ldots, v_N]^T \in \mathbb{C}^{N \times M}$$

Beamformer matrix
We address the weighted sum rate (WSR) maximization problem

\[
\max_V \sum_{i=1}^{N} \alpha_i \log_2 (1 + \text{SINR}_i) \quad (1a)
\]

s.t. \( \text{Tr}(VV^H) \leq P \quad (1b) \)
Beamforming problem

We address the weighted sum rate (WSR) maximization problem

\[
\max_{\mathbf{V}} \sum_{i=1}^{N} \alpha_i \log_2 (1 + \text{SINR}_i) \quad (1a)
\]

s.t. \( \text{Tr}(\mathbf{V} \mathbf{V}^H) \leq P \quad (1b) \)

- \( \log_2 (1 + \text{SINR}_i) \) is the rate of user \( i \)
Beamforming problem

We address the weighted sum rate (WSR) maximization problem

\[
\max_{\mathbf{V}} \sum_{i=1}^{N} \alpha_i \log_2 (1 + \text{SINR}_i)
\]

s.t. \( \text{Tr}(\mathbf{V}\mathbf{V}^H) \leq P \)  

- \( \log_2 (1 + \text{SINR}_i) \) is the rate of user \( i \)
- \( \text{SINR}_i = \frac{|h_i^Hv_i|^2}{\sum_{j=1,j\neq i}^{N} |h_i^Hv_j|^2 + \sigma^2} \) (signal-to-interference-plus-noise ratio of user \( i \))
Beamforming problem

We address the weighted sum rate (WSR) maximization problem

\[
\max_{\mathbf{V}} \quad \sum_{i=1}^{N} \alpha_i \log_2 (1 + \text{SINR}_i) \tag{1a}
\]

\[
\text{s.t.} \quad \text{Tr}(\mathbf{VV}^H) \leq P \tag{1b}
\]

- \(\log_2 (1 + \text{SINR}_i)\) is the rate of user \(i\)
- \(\text{SINR}_i = \frac{|h_i^H v_i|^2}{\sum_{j=1,j\neq i}^{N} |h_i^H v_j|^2 + \sigma^2}\) (signal-to-interference-plus-noise ratio of user \(i\))
- \(\alpha_i\) is the priority of user \(i\) (assumed to be known)
We address the weighted sum rate (WSR) maximization problem

$$\max_{\mathbf{V}} \sum_{i=1}^{N} \alpha_i \log_2 (1 + \text{SINR}_i)$$ \hspace{1cm} (1a)

s.t. \hspace{1cm} \text{Tr}(\mathbf{VV}^H) \leq P \hspace{1cm} (1b)

- $\log_2 (1 + \text{SINR}_i)$ is the rate of user $i$
- $\text{SINR}_i = \frac{|h_i^H v_i|^2}{\sum_{j=1,j\neq i}^{N} |h_i^H v_j|^2 + \sigma^2}$ (signal-to-interference-plus-noise ratio of user $i$)
- $\alpha_i$ is the priority of user $i$ (assumed to be known)

Problem (1) is known to be **NP-hard**\(^1\)

---

The weighted minimum mean square error (WMMSE) algorithm\textsuperscript{2} finds a **local optimum** by working on an **equivalent reformulation** of the WSR problem, i.e.,

\[
\begin{align}
\min_{u, w, V} & \quad f(u, w, V) \\
\text{s.t.} & \quad \text{Tr}(V V^H) \leq P
\end{align}
\]  

The weighted minimum mean square error (WMMSE) algorithm\textsuperscript{2} finds a local optimum by working on an equivalent reformulation of the WSR problem, i.e.,

\[
\begin{align*}
\min_{u, w, V} & \quad f(u, w, V) \\
\text{s.t.} & \quad \text{Tr}(VV^H) \leq P
\end{align*}
\]  

At each iteration of the WMMSE:

---

The weighted minimum mean square error (WMMSE) algorithm\(^2\) finds a **local optimum** by working on an **equivalent reformulation** of the WSR problem, i.e.,

\[
\min_{u, w, V} \quad f(u, w, V) \tag{2a}
\]

\[
\text{s.t.} \quad \text{Tr}(V V^H) \leq P \tag{2b}
\]

At each iteration of the WMMSE:
- the update of \(u\) is the optimal solution of \(\min_{\xi} f(\xi, w, V)\)

---

WMMSE algorithm

- The weighted minimum mean square error (WMMSE) algorithm\(^2\) finds a **local optimum** by working on an **equivalent reformulation** of the WSR problem, i.e.,

\[
\begin{align*}
\min_{u,w,V} & \quad f(u, w, V) \\
\text{s.t.} & \quad \text{Tr}(VV^H) \leq P
\end{align*}
\]  

(2a)

(2b)

- At each iteration of the WMMSE:
  - the update of \(u\) is the optimal solution of \(\min_\xi f(\xi, w, V)\)
  - the update of \(w\) is the optimal solution of \(\min_\xi f(u, \xi, V)\)

The weighted minimum mean square error (WMMSE) algorithm finds a local optimum by working on an equivalent reformulation of the WSR problem, i.e.,

\[
\begin{align*}
\min_{u,w,V} & \quad f(u, w, V) \\
\text{s.t.} & \quad \text{Tr}(V V^H) \leq P
\end{align*}
\] (2a)

At each iteration of the WMMSE:

- the update of \( u \) is the optimal solution of \( \min_{\xi} f(\xi, w, V) \)
- the update of \( w \) is the optimal solution of \( \min_{\xi} f(u, \xi, V) \)
- the update of \( V \) is the optimal solution of \( \min_{\xi} f(u, w, \xi) \) s.t. \( \text{Tr}(\xi \xi^H) \leq P \)

It is guaranteed to converge to a local optimum.

It exhibits a relatively high computational complexity.

The weighted minimum mean square error (WMMSE) algorithm\(^2\) finds a **local optimum** by working on an **equivalent reformulation** of the WSR problem, i.e.,

\[
\begin{align*}
\min_{u, w, V} & \quad f(u, w, V) \\
\text{s.t.} & \quad \text{Tr}(VV^H) \leq P
\end{align*}
\]

At each iteration of the WMMSE:
- the update of \(u\) is the optimal solution of \(\min_{\xi} f(\xi, w, V)\)
- the update of \(w\) is the optimal solution of \(\min_{\xi} f(u, \xi, V)\)
- the update of \(V\) is the optimal solution of \(\min_{\xi} f(u, w, \xi)\) s.t. \(\text{Tr}(\xi\xi^H) \leq P\)

It is **guaranteed to converge** to a local optimum

---

WMMSE algorithm

The weighted minimum mean square error (WMMSE) algorithm\(^2\) finds a **local optimum** by working on an **equivalent reformulation** of the WSR problem, i.e.,

\[
\begin{align*}
\min_{u, w, V} & \quad f(u, w, V) \\
\text{s.t.} & \quad \text{Tr}(VV^H) \leq P
\end{align*}
\] (2a)

At each iteration of the WMMSE:
- the update of \(u\) is the optimal solution of \(\min_\xi f(\xi, w, V)\)
- the update of \(w\) is the optimal solution of \(\min_\xi f(u, \xi, V)\)
- the update of \(V\) is the optimal solution of \(\min_\xi f(u, w, \xi)\) s.t. \(\text{Tr}(\xi\xi^H) \leq P\)

It is **guaranteed to converge** to a local optimum

It exhibits a **relatively high computational complexity**

Deep unfolding

- It is a learning technique applicable to **iterative algorithms**
Deep unfolding

- It is a learning technique applicable to \textit{iterative algorithms}
- \textbf{Goal}: trade off complexity and performance in presence of constraints
Deep unfolding

- It is a learning technique applicable to **iterative algorithms**
- **Goal**: trade off complexity and performance in presence of constraints
- **Key idea**: build and train a neural network whose structure is determined by the iterative algorithm

- Map each iteration of the algorithm to a neural network layer
- Fix the number of layers of the network according to the complexity and latency constraints
- Select the trainable parameters
- Train the network with gradient-based methods and back-propagation

- It incorporates domain knowledge in the structure of the network
Deep unfolding

- It is a learning technique applicable to **iterative algorithms**
- **Goal**: trade off complexity and performance in presence of constraints
- **Key idea**: build and train a neural network whose structure is determined by the iterative algorithm
  - Map each iteration of the algorithm to a neural network layer
Deep unfolding

- It is a learning technique applicable to **iterative algorithms**
- **Goal**: trade off complexity and performance in presence of constraints
- **Key idea**: build and train a neural network whose structure is determined by the iterative algorithm
  - Map each iteration of the algorithm to a neural network layer
  - Fix the number of layers of the network according to the complexity and latency constraints

- It incorporates domain knowledge in the structure of the network
Deep unfolding

- It is a learning technique applicable to **iterative algorithms**
- **Goal**: trade off complexity and performance in presence of constraints
- **Key idea**: build and train a neural network whose structure is determined by the iterative algorithm
  - Map each iteration of the algorithm to a neural network layer
  - Fix the number of layers of the network according to the complexity and latency constraints
  - Select the trainable parameters
Deep unfolding

- It is a learning technique applicable to **iterative algorithms**
- **Goal**: trade off complexity and performance in presence of constraints
- **Key idea**: build and train a neural network whose structure is determined by the iterative algorithm
  - Map each iteration of the algorithm to a neural network layer
  - Fix the number of layers of the network according to the complexity and latency constraints
  - Select the trainable parameters
  - Train the network with gradient-based methods and back-propagation
Deep unfolding

- It is a learning technique applicable to **iterative algorithms**
- **Goal:** trade off complexity and performance in presence of constraints
- **Key idea:** build and train a neural network whose structure is determined by the iterative algorithm
  - Map each iteration of the algorithm to a neural network layer
  - Fix the number of layers of the network according to the complexity and latency constraints
  - Select the trainable parameters
  - Train the network with gradient-based methods and back-propagation
- It incorporates **domain knowledge** in the structure of the network
Deep unfolding

Advantages with respect to standard neural network solutions

- No architecture selection
- Explainability
- Fewer parameters to train
The WMMSE algorithm involves operations that are hard to map to neural network layers as acknowledged by Sun et al.\(^3\)

\(^3\)Sun et al., ”Learning to Optimize: Training Deep Neural Networks for Interference Management,” *IEEE Transactions on Signal Processing*, 2018
The WMMSE algorithm involves operations that are **hard to map to neural network layers as acknowledged by Sun et al.**

Sun et al., "Learning to Optimize: Training Deep Neural Networks for Interference Management,” *IEEE Transactions on Signal Processing*, 2018
The update equation of $V$ is obtained by solving

$$\min_{\xi} \quad f(u, w, \xi)$$

(3a)

subject to

$$\text{Tr}(\xi\xi^H) \leq P,$$

(3b)

with the method of Lagrange multipliers.
The update equation of $\mathbf{V}$ is obtained by solving

$$\min_{\xi} \ f(u, w, \xi)$$

subject to

$$\text{Tr}(\xi \xi^H) \leq P,$$

with the method of Lagrange multipliers.

It leads to a matrix inversion, an eigendecomposition, and a bisection search.
The update equation of $V$ is obtained by solving

$$\min_{\xi} \ f(u, w, \xi)$$  \hspace{1cm} (3a)

subject to

$$\text{Tr}(\xi \xi^H) \leq P,$$  \hspace{1cm} (3b)

with the method of Lagrange multipliers.

It leads to a matrix inversion, an eigendecomposition, and a bisection search.

We observe that

- The cost function is convex
- The constraint set is convex
WMMSE algorithm - Deep unfolding

- The update equation of $\mathbf{V}$ is obtained by solving
  \[
  \min_{\xi} \quad f(u, w, \xi) \quad \text{(3a)}
  \]
  \[
  \text{s.t.} \quad \text{Tr}(\xi\xi^H) \leq P, \quad \text{(3b)}
  \]
  with the method of Lagrange multipliers.

- It leads to a matrix inversion, an eigendecomposition, and a bisection search.

- We observe that
  - The cost function is convex
  - The constraint set is convex

- We propose to solve (3) with the **projected gradient descent (PGD)** approach.
The update equation of $V$ is obtained by solving

$$\min_{\xi} f(u, w, \xi)$$

subject to

$$\text{Tr}(\xi \xi^H) \leq P,$$

with the method of Lagrange multipliers.

It leads to a matrix inversion, an eigendecomposition, and a bisection search.

We observe that

- The cost function is convex
- The constraint set is convex

We propose to solve (3) with the **projected gradient descent (PGD)** approach.

We truncate the sequence of PGD steps to $K$. 
Unfoldable WMMSE algorithm

- At each iteration:
  - the update of $u$ is the optimal solution of $\min_\xi f(\xi, w, V)$
  - the update of $w$ is the optimal solution of $\min_\xi f(u, \xi, V)$
  - the update of $V$ is given by K PGD steps

Convergence

We can prove that the unfoldable WMMSE algorithm retains the same convergence guarantees of the original WMMSE.
Unfoldable WMMSE algorithm

▶ At each iteration:
  - the update of $u$ is the optimal solution of $\min_{\xi} f(\xi, w, V)$
  - the update of $w$ is the optimal solution of $\min_{\xi} f(u, \xi, V)$
  - the update of $V$ is given by $K$ PGD steps

Convergence

We can prove that the unfoldable WMMSE algorithm retains the same convergence guarantees of the original WMMSE
Deep unfolded WMMSE

We select the step sizes of the PGD (\(\Gamma\)) to be the trainable parameters

We minimize the following loss function

\[
L(\Gamma) = -\sum_{n=1}^{N_s} \sum_{l=1}^{N_s} f_{WSR}(H_n, V_l; \Gamma)
\]
We select the step sizes of the PGD (Γ) to be the trainable parameters.
We select the step sizes of the PGD (\(\Gamma\)) to be the trainable parameters

We minimize the following loss function

\[
\mathcal{L}(\Gamma) = -\frac{1}{N_s} \sum_{n=1}^{N_s} \sum_{l=1}^{L} f_{WSR}(H_n, V_l; \Gamma)
\]

where \(N_s\) is the size of the training set
Deep unfolded WMMSE

▶ We select the step sizes of the PGD ($\Gamma$) to be the trainable parameters
▶ We minimize the following loss function

$$\mathcal{L}(\Gamma) = -\frac{1}{N_s} \sum_{n=1}^{N_s} \sum_{l=1}^{L} f_{\text{WSR}}(H_n, V_l; \Gamma)$$  \hspace{1cm} (4)$$

where $N_s$ is the size of the training set  \hspace{1cm} Weighted Sum Rate
### Numerical results

- $M = 4$
- $N = 4$
- $\frac{P}{\sigma^2} = 10$ dB
- 4 PGD steps

<table>
<thead>
<tr>
<th>Number of iterations $L$</th>
<th>Sum rate [bits per channel use]</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8</td>
</tr>
<tr>
<td>2</td>
<td>8.5</td>
</tr>
<tr>
<td>3</td>
<td>9</td>
</tr>
<tr>
<td>4</td>
<td>9.5</td>
</tr>
<tr>
<td>5</td>
<td>9.5</td>
</tr>
<tr>
<td>6</td>
<td>9.5</td>
</tr>
</tbody>
</table>

- **Deep unfolded WMMSE**
- **WMMSE**
- **Deep unfolded WMMSE - same $\gamma$**
- **WMMSE at convergence**
Conclusion

- We addressed the **trade-off between complexity and performance** for the WSR maximization beamforming problem.
Conclusion

- We addressed the trade-off between complexity and performance for the WSR maximization beamforming problem.
- To this end, we provided a variant of the WMMSE algorithm that:
  - allows for the novel application of deep unfolding
  - retains the same convergence guarantees of the original WMMSE algorithm
Conclusion

▶ We addressed the trade-off between complexity and performance for the WSR maximization beamforming problem

▶ To this end, we provided a variant of the WMMSE algorithm that
  – allows for the novel application of deep unfolding
  – retains the same convergence guarantees of the original WMMSE algorithm

▶ Numerical results confirmed that the deep unfolded WMMSE successfully addresses the trade-off
Thank you for your attention!

https://github.com/lpkg/ WMMSE-deep-unfolding/tree/ICASSP2021

You can reach out to me at pellaco@kth.se