Deep Weighted MMSE Downlink Beamforming

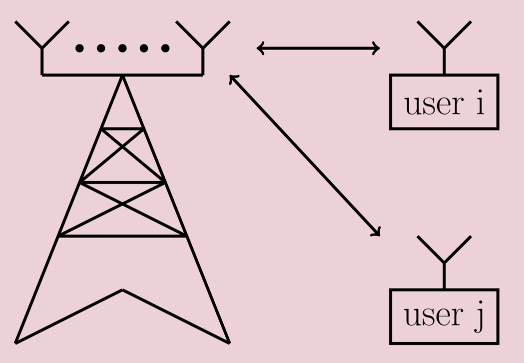
Lissy Pellaco, Mats Bengtsson, Joakim Jaldén
KTH Royal Institute of Technology, Stockholm, Sweden

Nr. 1795
pellaco@kth.se

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

Problem Formulation

- MU-MISO interference downlink channel
- Single base station with M transmit antennas
- N single-antenna users



Received signal of user i

$y_i = \boldsymbol{h}_i^H \boldsymbol{v}_i x_i + \sum_{j=1, j\neq i}^{N} \boldsymbol{h}_i^H \boldsymbol{v}_j x_j + n_i$

$\boldsymbol{h}_i \in \mathbb{C}^M$: channel, n_i: noise

$\boldsymbol{v}_i \in \mathbb{C}^M$: beamformer vector

x_i: transmitted symbol

$\boldsymbol{V} \triangleq [\boldsymbol{v}_1, \boldsymbol{v}_2, \ldots, \boldsymbol{v}_N]^T \in \mathbb{C}^{N\times M}$

- We address the **NP-hard** problem

$$\max_{\boldsymbol{V}} \sum_{i=1}^{N} \alpha_i \log_2(1+\mathrm{SINR}_i) \quad (1a)$$
$$\text{s.t.} \quad \mathrm{Tr}(\boldsymbol{V}\boldsymbol{V}^H) \leq P \quad (1b)$$

– $\log_2(1+\mathrm{SINR}_i)$ is the rate of user i

– α_i is the priority of user i (assumed to be known)

- We want to comply with the **power consumption** and **latency constraints** at the base station

WMMSE algorithm

- It works on an **equivalent reformulation**

$$\min_{\boldsymbol{u},\boldsymbol{w},\boldsymbol{V}} f(\boldsymbol{u},\boldsymbol{w},\boldsymbol{V}) \quad (2a)$$
$$\text{s.t.} \quad \mathrm{Tr}(\boldsymbol{V}\boldsymbol{V}^H) \leq P \quad (2b)$$

- f is jointly nonconvex over $(\boldsymbol{u},\boldsymbol{w},\boldsymbol{V})$
- f is **convex in each optimization variable**

PSEUDOCODE

repeat
  $\boldsymbol{u} = \mathrm{argmin}_{\boldsymbol{\xi}} f(\boldsymbol{\xi},\boldsymbol{w},\boldsymbol{V})$
  $\boldsymbol{w} = \mathrm{argmin}_{\boldsymbol{\xi}} f(\boldsymbol{u},\boldsymbol{\xi},\boldsymbol{V})$
  $\boldsymbol{V} = \mathrm{argmin}_{\boldsymbol{\xi}} f(\boldsymbol{u},\boldsymbol{w},\boldsymbol{\xi})$ s.t. $\mathrm{Tr}(\boldsymbol{\xi}\boldsymbol{\xi}^H) \leq P$
until convergence

- Guaranteed to **converge to a local optimum**
- **Relatively high computational complexity**

Deep unfolding

- **Goal**: trade off complexity and performance in presence of computational and latency constraints for iterative algorithms
- **Key idea**: build and train a neural network whose structure is determined by the iterative algorithm
- It **incorporates domain knowledge**

WMMSE - Deep unfolding

- The update equations of $\boldsymbol{u}$ and $\boldsymbol{w}$ can be **easily mapped to neural network layers**
- Conversely, the update of $\boldsymbol{V}$ is obtained by

$$\min_{\boldsymbol{\xi}} f(\boldsymbol{u},\boldsymbol{w},\boldsymbol{\xi}) \quad (3a)$$
$$\text{s.t.} \quad \mathrm{Tr}(\boldsymbol{\xi}\boldsymbol{\xi}^H) \leq P, \quad (3b)$$

with the method of Lagrange multipliers

- It leads to a **matrix inversion**, an **eigendecomposition**, and a **bisection search** which constitute an **obstacle to deep unfolding**

Unfoldable WMMSE algorithm

- We propose to solve (3) with the **projected gradient descent (PGD)** approach
- We **truncate** the sequence of PGD steps to K

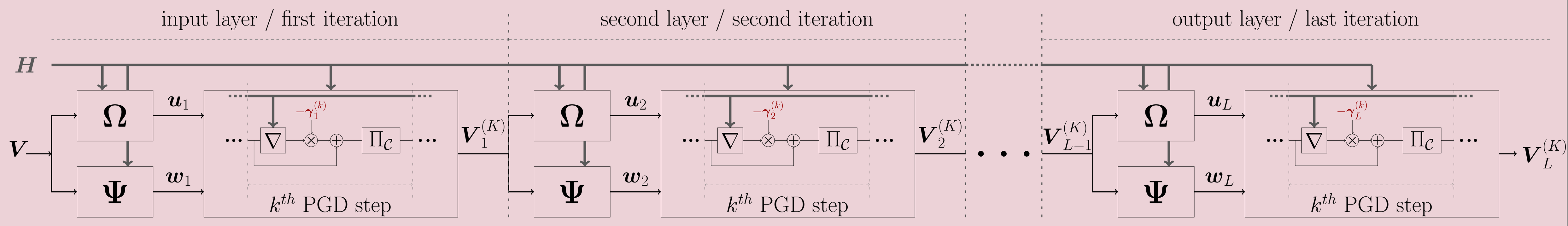
PSEUDOCODE

for $l = 1, \ldots, L$
  $\boldsymbol{u} = \mathrm{argmin}_{\boldsymbol{\xi}} f(\boldsymbol{\xi},\boldsymbol{w},\boldsymbol{V})$
  $\boldsymbol{w} = \mathrm{argmin}_{\boldsymbol{\xi}} f(\boldsymbol{u},\boldsymbol{\xi},\boldsymbol{V})$
  for $k = 1, \ldots, K$
    $\boldsymbol{V}^{(k)} = \Pi_{\mathcal{C}}\{\boldsymbol{V}^{(k-1)} - \gamma\nabla f(\boldsymbol{V}^{(k-1)})\}$

$$\Pi_{\mathcal{C}}\{\boldsymbol{V}\} = \begin{cases} \boldsymbol{V}, & \text{if } \mathrm{Tr}(\boldsymbol{V}\boldsymbol{V}^H) \leq P \\ \frac{\boldsymbol{V}}{\|\boldsymbol{V}\|}\sqrt{P}, & \text{otherwise.} \end{cases} \quad (4)$$

We prove the unfoldable WMMSE retains the **same convergence guarantees** of the WMMSE

Deep unfolded WMMSE



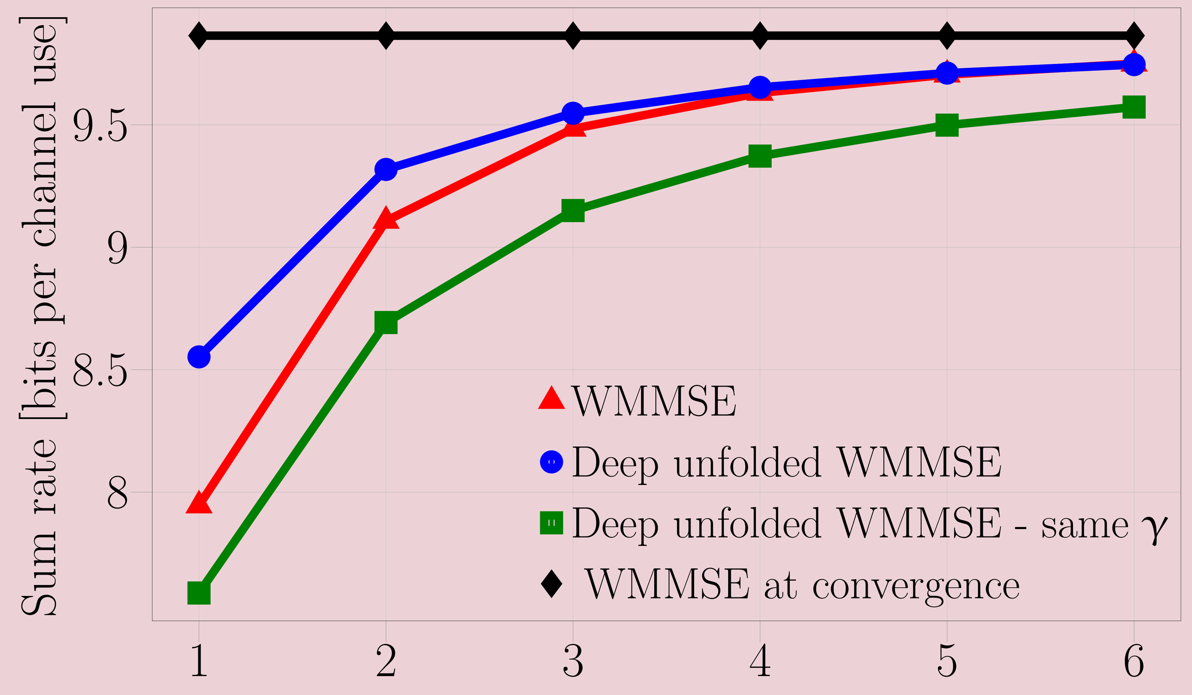
- The step sizes of the PGD ($\boldsymbol{\Gamma}$) are the trainable parameters, where $\boldsymbol{\Gamma} = (\boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2, \ldots, \boldsymbol{\gamma}_L)$
- We minimize the **global objective** loss function

$$\mathcal{L}(\boldsymbol{\Gamma}) = -\frac{1}{N_s}\sum_{n=1}^{N_s}\sum_{l=1}^{L} f_{\mathrm{WSR}}(\boldsymbol{H}_n, \boldsymbol{V}_l; \boldsymbol{\Gamma}) \quad (5)$$

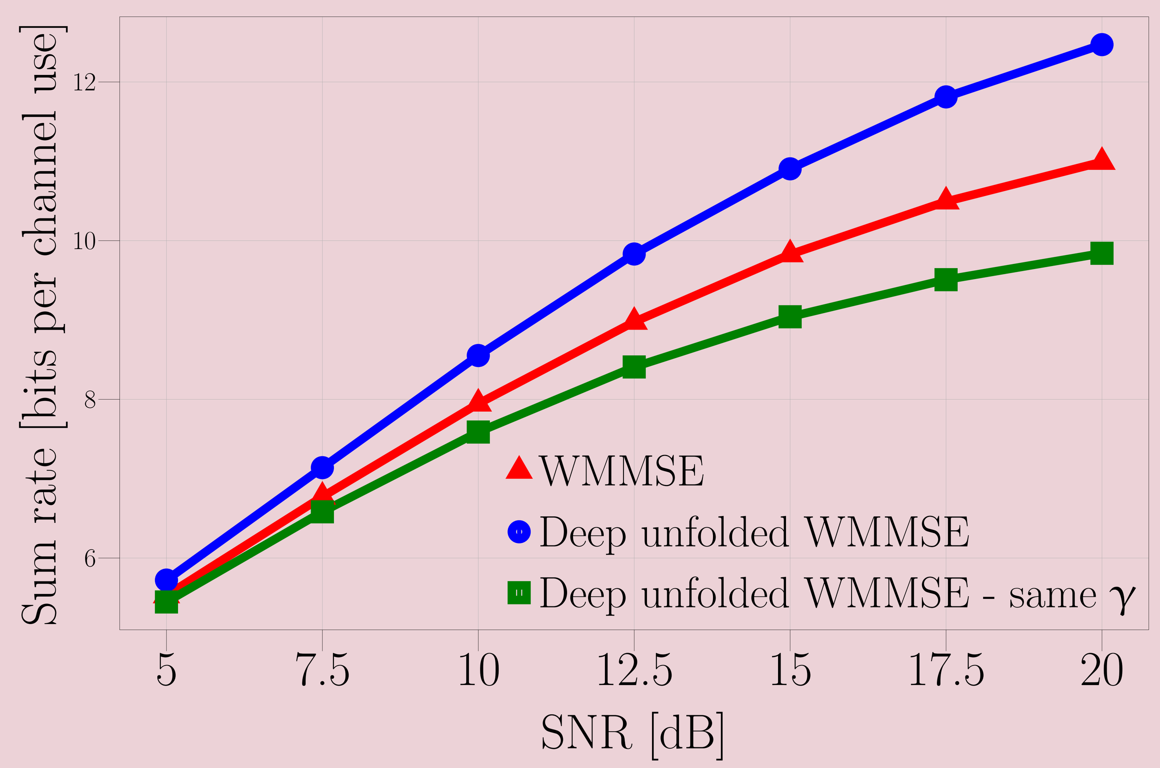
where N_s is the size of the training set

Numerical Results

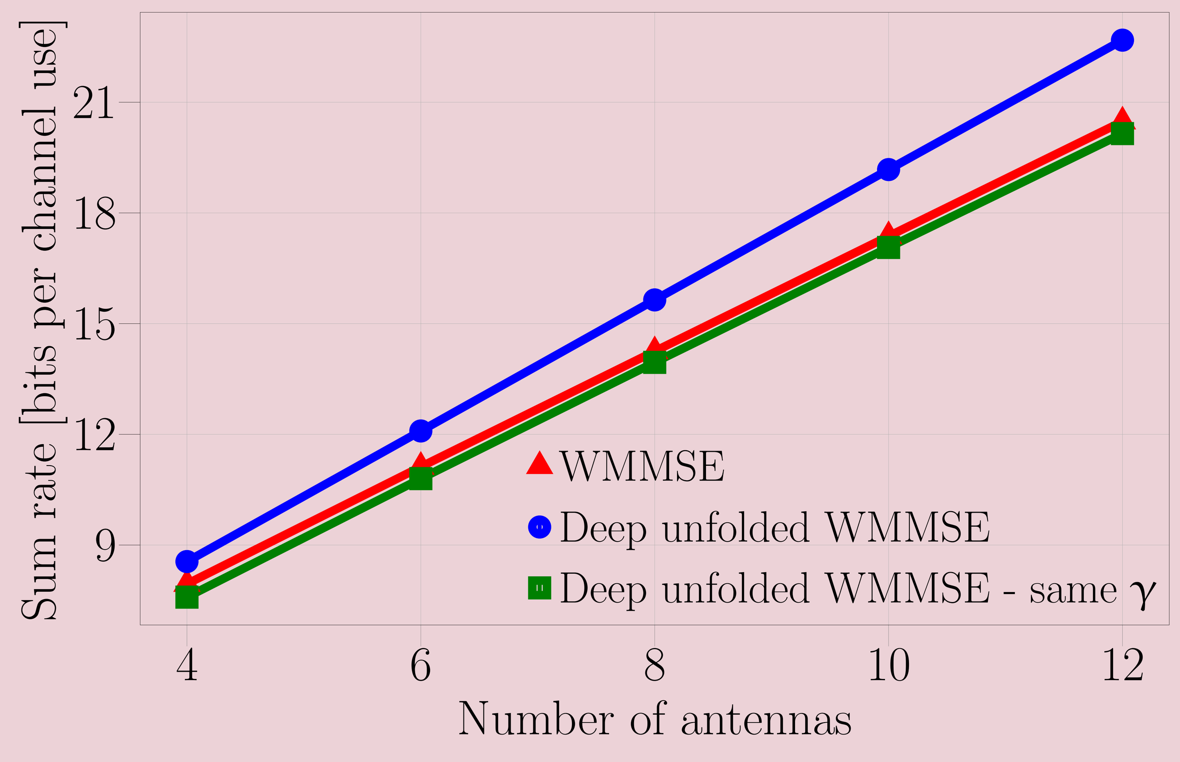
- We initialize $\boldsymbol{V}$ with matched filtering
- We initialize $\gamma_l^{(k)} = 1$ for $k = 1, \ldots, K$ and $l = 1, \ldots, L$
- We set $\alpha_i = 1$ for $i = 1, \ldots, N$
- $\boldsymbol{h}$ is drawn from a Rayleigh distribution
- For each combination of M, L, K, and SNR we train a different network
- We adopt the Adam optimizer
- 'same γ' - step sizes constrained to be equal across all PGD steps of the same layer: $\gamma_l^{(1)} = \ldots = \gamma_l^{(K)}$ for $l = 1, \ldots, L$



$N = M = 4$, $K = 4$, and SNR of 10 dB



$N = M = 4$, $L = 1$, and $K = 4$



$N = M$, $L = 1$, $K = 4$, and SNR of 10 dB