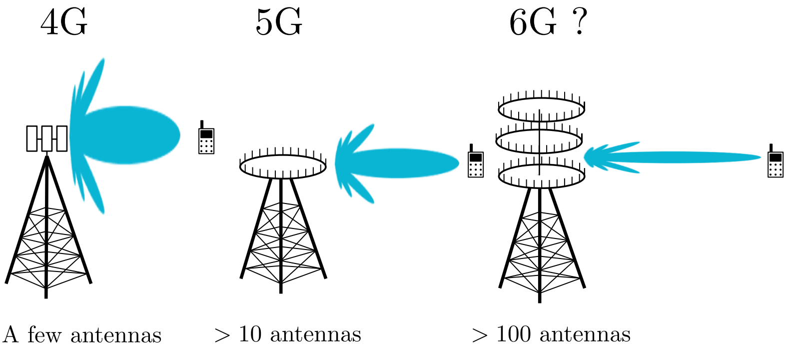
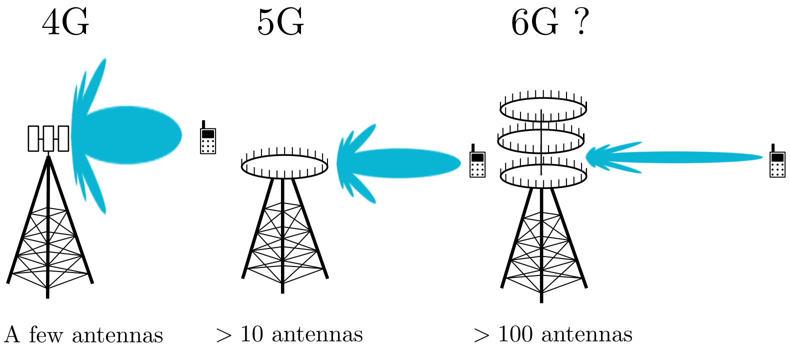
/ Online Unsupervised Deep Unfolding for SISO-OFDM channel estimation /

Baptiste CHATELIER[†,‡,⋆], Luc LE MAGOAROU[†], Getachew REDIETEAB[§,⋆]

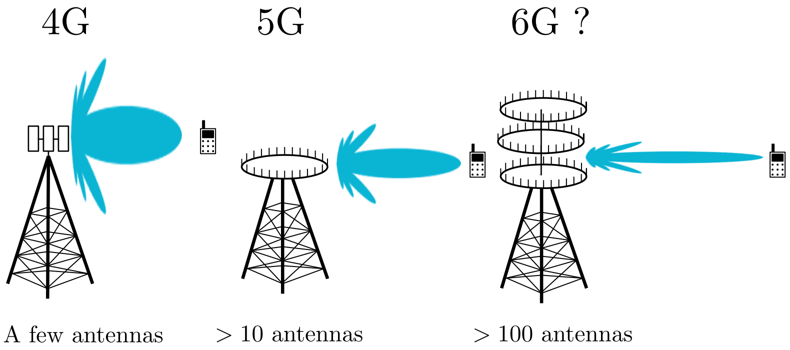
† Univ Rennes, INSA Rennes, CNRS, IETR-UMR 6164, Rennes, France
‡ Mitsubishi Electric R&D Centre Europe, Rennes, France
§ Orange Innovation, Rennes, France
⋆ b<>com, Rennes, France

baptiste.chatelier@insa-rennes.fr


4G
5G
6G ?
A few antennas
> 10 antennas
> 100 antennas

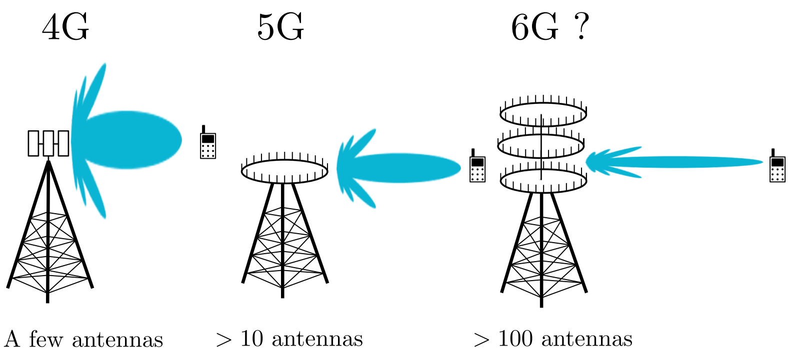


The dimension of channels increases



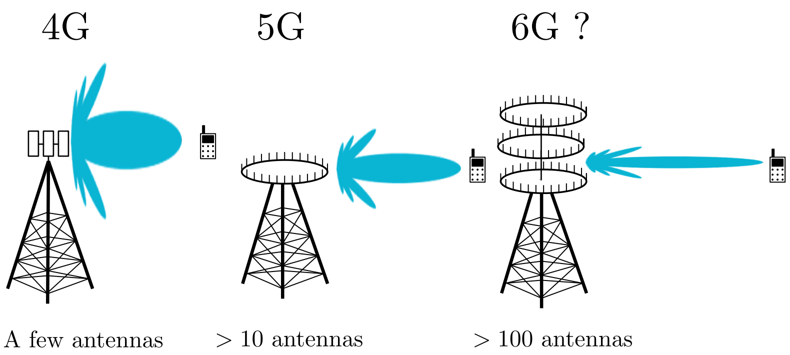
The dimension of channels increases

- Consequences:



The dimension of channels increases

- Consequences:
  - Channels are more and more difficult to estimate



The dimension of channels increases

- Consequences:
  - Channels are more and more difficult to estimate
  - Channels contain more and more information

- Typical data processing setting:
  - We observe a ***large*** number of ***correlated*** variables, explained by a ***small*** number of ***independent*** factors.

- Typical data processing setting:
  - We observe a ***large*** number of ***correlated*** variables, explained by a ***small*** number of ***independent*** factors.

There are two complementary approaches to handle this situation:

- Typical data processing setting:
  - We observe a ***large*** number of ***correlated*** variables, explained by a ***small*** number of ***independent*** factors.

There are two complementary approaches to handle this situation:

- **Signal processing**
  - Model based (analytical description of the manifold)
  - Large bias
  - Low complexity

- Typical data processing setting:
  - We observe a ***large*** number of ***correlated*** variables, explained by a ***small*** number of ***independent*** factors.

There are two complementary approaches to handle this situation:

- **Signal processing**
  - Model based (analytical description of the manifold)
  - Large bias
  - Low complexity
- **Machine learning/Artificial intelligence**
  - Data based (sampling of the manifold)
  - Low bias
  - High complexity

- Typical data processing setting:
  - We observe a ***large*** number of ***correlated*** variables, explained by a ***small*** number of ***independent*** factors.

There are two complementary approaches to handle this situation:

- **Signal processing**
  - Model based (analytical description of the manifold)
  - Large bias
  - Low complexity
- **Machine learning/Artificial intelligence**
  - Data based (sampling of the manifold)
  - Low bias
  - High complexity

Hybrid approach: Model-based AI
Use models to structure, initialize and train learning methods

- Typical data processing setting:
  - We observe a ***large*** number of ***correlated*** variables, explained by a ***small*** number of ***independent*** factors.

There are two complementary approaches to handle this situation:

- **Signal processing**
  - Model based (analytical description of the manifold)
  - Large bias
  - Low complexity
- **Machine learning/Artificial intelligence**
  - Data based (sampling of the manifold)
  - Low bias
  - High complexity

Hybrid approach: Model-based AI
Use models to structure, initialize and train learning methods

- Make models more flexible: reduce bias of signal processing methods

- Typical data processing setting:
  - We observe a ***large*** number of ***correlated*** variables, explained by a ***small*** number of ***independent*** factors.

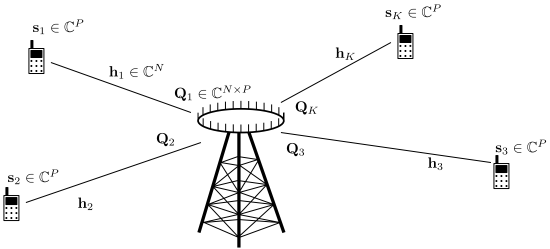
There are two complementary approaches to handle this situation:

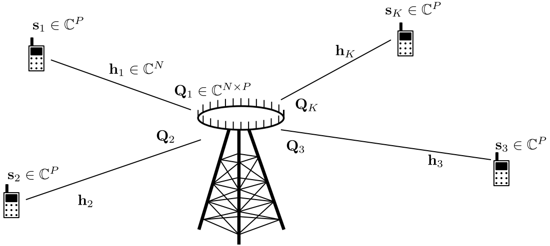
- **Signal processing**
  - Model based (analytical description of the manifold)
  - Large bias
  - Low complexity
- **Machine learning/Artificial intelligence**
  - Data based (sampling of the manifold)
  - Low bias
  - High complexity

Hybrid approach: Model-based AI
Use models to structure, initialize and train learning methods

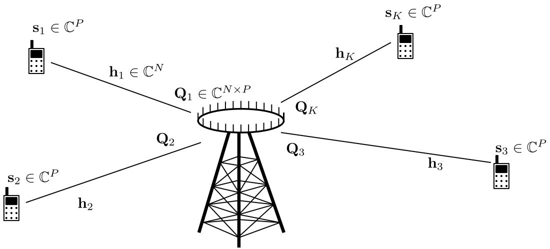
- Make models more flexible: reduce bias of signal processing methods
- Guide machine learning methods: reduce their complexity

Resilient channel estimation

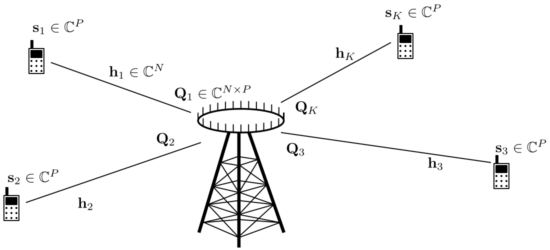

$\mathbf{s}_1 \in \mathbb{C}^P$
$\mathbf{h}_1 \in \mathbb{C}^N$
$\mathbf{Q}_1 \in \mathbb{C}^{N \times P}$
$\mathbf{s}_K \in \mathbb{C}^P$
$\mathbf{h}_K$
$\mathbf{Q}_K$
$\mathbf{Q}_2$
$\mathbf{s}_2 \in \mathbb{C}^P$
$\mathbf{h}_2$
$\mathbf{Q}_3$
$\mathbf{h}_3$
$\mathbf{s}_3 \in \mathbb{C}^P$



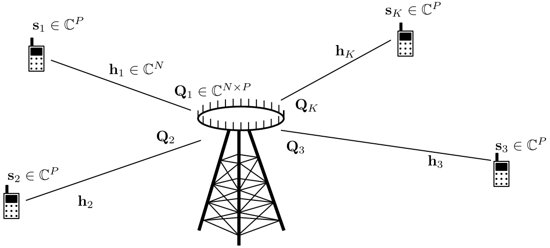
- Orthogonal pilot sequences: $\mathbf{s}_i^H \mathbf{s}_j = \delta_{i,j}$



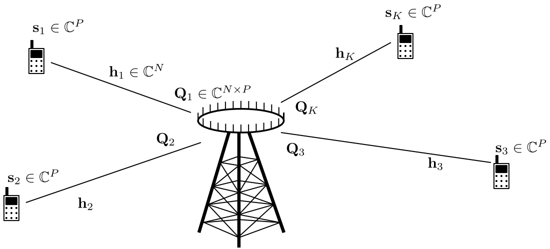
- Orthogonal pilot sequences: $\mathbf{s}_i^H \mathbf{s}_j = \delta_{i,j}$
- Signal due to the k-th UE: $\mathbf{Q}_k = \mathbf{h}_k \mathbf{s}_k^T$



- Orthogonal pilot sequences: $\mathbf{s}_i^H \mathbf{s}_j = \delta_{i,j}$
- Signal due to the k-th UE: $\mathbf{Q}_k = \mathbf{h}_k \mathbf{s}_k^T$
- Full observation at the BS: $\mathbf{Q} = \sum_{k=1}^{K} \mathbf{Q}_k + \mathbf{W}$



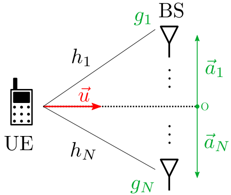
- Orthogonal pilot sequences: $\mathbf{s}_i^H \mathbf{s}_j = \delta_{i,j}$
- Signal due to the k-th UE: $\mathbf{Q}_k = \mathbf{h}_k \mathbf{s}_k^T$
- Full observation at the BS: $\mathbf{Q} = \sum_{k=1}^{K} \mathbf{Q}_k + \mathbf{W}$
- LS estimate: $\mathbf{x}_i = \mathbf{Q}\mathbf{s}_i^* = \mathbf{h}_i + \mathbf{n} \in \mathbb{C}^N$



- Orthogonal pilot sequences: $\mathbf{s}_i^H \mathbf{s}_j = \delta_{i,j}$
- Signal due to the k-th UE: $\mathbf{Q}_k = \mathbf{h}_k \mathbf{s}_k^T$
- Full observation at the BS: $\mathbf{Q} = \sum_{k=1}^{K} \mathbf{Q}_k + \mathbf{W}$
- LS estimate: $\mathbf{x}_i = \mathbf{Q}\mathbf{s}_i^* = \mathbf{h}_i + \mathbf{n} \in \mathbb{C}^N$

How to denoise the channels ?

SU-MIMO



Physical model:

Plane wave assumption:

$$h_O = \beta$$

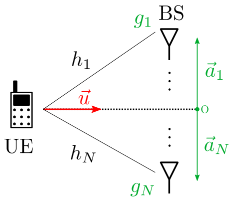
$$\Downarrow$$

$$h_i = \beta g_i \mathrm{e}^{-\mathrm{j}\frac{2\pi}{\lambda}\vec{a}_i \cdot \vec{u}}$$

$$\Downarrow$$

$$\mathbf{h} = \sum_{l=1}^{L_p} \beta_l \begin{bmatrix} g_1 \mathrm{e}^{-\mathrm{j}\frac{2\pi}{\lambda}\vec{a}_1 \cdot \vec{u}_l} \\ \vdots \\ g_N \mathrm{e}^{-\mathrm{j}\frac{2\pi}{\lambda}\vec{a}_N \cdot \vec{u}_l} \end{bmatrix}$$

SU-MIMO



Physical model:

Plane wave assumption:

$$h_O = \beta$$

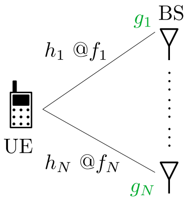
$$\Downarrow$$

$$h_i = \beta g_i \mathrm{e}^{-\mathrm{j}\frac{2\pi}{\lambda}\vec{a}_i \cdot \vec{u}}$$

$$\Downarrow$$

$$\mathbf{h} = \sum_{l=1}^{L_p} \beta_l \begin{bmatrix} g_1 \mathrm{e}^{-\mathrm{j}\frac{2\pi}{\lambda}\vec{a}_1 \cdot \vec{u}_l} \\ \vdots \\ g_N \mathrm{e}^{-\mathrm{j}\frac{2\pi}{\lambda}\vec{a}_N \cdot \vec{u}_l} \end{bmatrix}$$

SU-SISO-OFDM



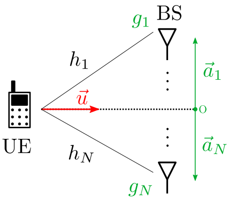
Physical model:

$$h_i = \beta g_i \mathrm{e}^{-\mathrm{j}2\pi f_i \tau}$$

$$\Downarrow$$

$$\mathbf{h} = \sum_{l=1}^{L_p} \beta_l \begin{bmatrix} g_1 \mathrm{e}^{-\mathrm{j}2\pi f_1 \tau_l} \\ \vdots \\ g_N \mathrm{e}^{-\mathrm{j}2\pi f_N \tau_l} \end{bmatrix}$$

SU-MIMO



Physical model:

Plane wave assumption:

$$h_O = \beta$$

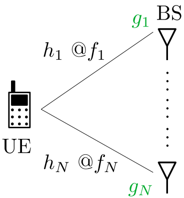
$$\Downarrow$$

$$h_i = \beta g_i \mathrm{e}^{-\mathrm{j}\frac{2\pi}{\lambda}\vec{a}_i \cdot \vec{u}}$$

$$\Downarrow$$

$$\mathbf{h} = \sum_{l=1}^{L_p} \beta_l \begin{bmatrix} g_1 \mathrm{e}^{-\mathrm{j}\frac{2\pi}{\lambda}\vec{a}_1 \cdot \vec{u}_l} \\ \vdots \\ g_N \mathrm{e}^{-\mathrm{j}\frac{2\pi}{\lambda}\vec{a}_N \cdot \vec{u}_l} \end{bmatrix}$$

SU-SISO-OFDM



Physical model:

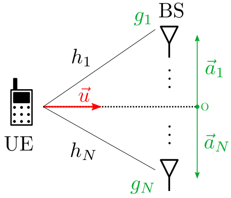
$$h_i = \beta g_i \mathrm{e}^{-\mathrm{j}2\pi f_i \tau}$$

$$\Downarrow$$

$$\mathbf{h} = \sum_{l=1}^{L_p} \beta_l \begin{bmatrix} g_1 \mathrm{e}^{-\mathrm{j}2\pi f_1 \tau_l} \\ \vdots \\ g_N \mathrm{e}^{-\mathrm{j}2\pi f_N \tau_l} \end{bmatrix}$$

- The BS has noisy estimates of the channels: $\mathbf{x} = \mathbf{h} + \mathbf{n}$, $\mathbf{n} \sim \mathcal{CN}\left(0, \sigma^2 \mathbf{Id}\right)$

SU-MIMO



Physical model:

Plane wave assumption:

$$h_O = \beta$$

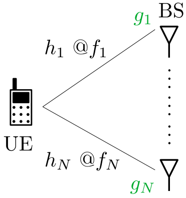
$$\Downarrow$$

$$h_i = \beta g_i \mathrm{e}^{-\mathrm{j}\frac{2\pi}{\lambda}\vec{a}_i \cdot \vec{u}}$$

$$\Downarrow$$

$$\mathbf{h} = \sum_{l=1}^{L_p} \beta_l \begin{bmatrix} g_1 \mathrm{e}^{-\mathrm{j}\frac{2\pi}{\lambda}\vec{a}_1 \cdot \vec{u}_l} \\ \vdots \\ g_N \mathrm{e}^{-\mathrm{j}\frac{2\pi}{\lambda}\vec{a}_N \cdot \vec{u}_l} \end{bmatrix}$$

SU-SISO-OFDM



Physical model:

$$h_i = \beta g_i \mathrm{e}^{-\mathrm{j}2\pi f_i \tau}$$

$$\Downarrow$$

$$\mathbf{h} = \sum_{l=1}^{L_p} \beta_l \begin{bmatrix} g_1 \mathrm{e}^{-\mathrm{j}2\pi f_1 \tau_l} \\ \vdots \\ g_N \mathrm{e}^{-\mathrm{j}2\pi f_N \tau_l} \end{bmatrix}$$

- The BS has noisy estimates of the channels: $\mathbf{x} = \mathbf{h} + \mathbf{n}$, $\mathbf{n} \sim \mathcal{CN}\left(0, \sigma^2 \mathbf{Id}\right)$

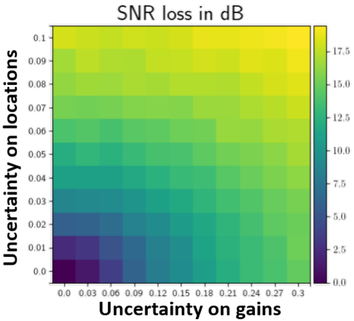
The physical model allows to denoise.

- The physical model can't be perfectly known:

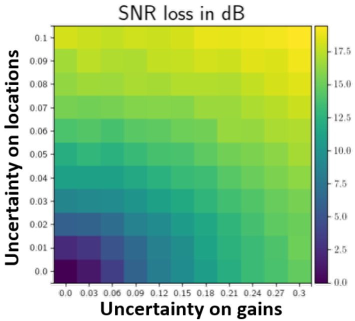
- The physical model can't be perfectly known:
  - Plane wave assumption $\rightarrow$ Only good for large distances

- The physical model can't be perfectly known:
  - Plane wave assumption $\rightarrow$ Only good for large distances
  - Antenna positions and gains are not exactly known, same for subcarriers frequencies.

- The physical model can't be perfectly known:
  - Plane wave assumption → Only good for large distances
  - Antenna positions and gains are not exactly known, same for subcarriers frequencies.
- Impact on channel denoising:



- The physical model can't be perfectly known:
  - Plane wave assumption → Only good for large distances
  - Antenna positions and gains are not exactly known, same for subcarriers frequencies.
- Impact on channel denoising:



How to counter this performance loss ? Use of a neural network

- Unsupervised, online, SNR-adaptive neural network. Based on the *Deep-Unfolding*[1] approach.

[1] Balatsoukas-Stimming and Studer, “Deep Unfolding for Communications Systems: A Survey and Some New Directions”.

- Unsupervised, online, SNR-adaptive neural network. Based on the *Deep-Unfolding*[1] approach.
- MP algorithm (one iteration):

[1] Balatsoukas-Stimming and Studer, “Deep Unfolding for Communications Systems: A Survey and Some New Directions”.

- Unsupervised, online, SNR-adaptive neural network. Based on the *Deep-Unfolding*[1] approach.
- MP algorithm (one iteration):
    1. Correlation : $\mathbf{\Psi}^H\mathbf{x}$

[1] Balatsoukas-Stimming and Studer, “Deep Unfolding for Communications Systems: A Survey and Some New Directions”.

- Unsupervised, online, SNR-adaptive neural network. Based on the *Deep-Unfolding*[1] approach.

- MP algorithm (one iteration):
  1. Correlation : $\mathbf{\Psi}^H \mathbf{x}$
  2. Argmax search : $i^\star = \arg\max_i \left|\boldsymbol{\psi}_i^H \mathbf{x}\right|$

[1] Balatsoukas-Stimming and Studer, "Deep Unfolding for Communications Systems: A Survey and Some New Directions".

- Unsupervised, online, SNR-adaptive neural network. Based on the *Deep-Unfolding*[1] approach.

- MP algorithm (one iteration):
  1. Correlation : $\mathbf{\Psi}^H \mathbf{x}$
  2. Argmax search : $i^\star = \arg\max_i \left|\boldsymbol{\psi}_i^H \mathbf{x}\right|$
  3. Projection : $\hat{\mathbf{h}} = \boldsymbol{\psi}_{i^\star} \boldsymbol{\psi}_{i^\star}^H \mathbf{x}$

[1] Balatsoukas-Stimming and Studer, "Deep Unfolding for Communications Systems: A Survey and Some New Directions".

- Unsupervised, online, SNR-adaptive neural network. Based on the *Deep-Unfolding*[1] approach.

- MP algorithm (one iteration):
  1. Correlation : $\mathbf{\Psi}^H \mathbf{x}$
  2. Argmax search : $i^\star = \arg\max_i \left|\boldsymbol{\psi}_i^H \mathbf{x}\right|$
  3. Projection : $\hat{\mathbf{h}} = \boldsymbol{\psi}_{i^\star} \boldsymbol{\psi}_{i^\star}^H \mathbf{x}$

- mpNet:

[1] Balatsoukas-Stimming and Studer, “Deep Unfolding for Communications Systems: A Survey and Some New Directions”.

- Unsupervised, online, SNR-adaptive neural network. Based on the *Deep-Unfolding*[1] approach.

- MP algorithm (one iteration):
  1. Correlation : $\mathbf{\Psi}^H \mathbf{x}$
  2. Argmax search : $i^\star = \arg\max_i |\boldsymbol{\psi}_i^H \mathbf{x}|$
  3. Projection : $\hat{\mathbf{h}} = \boldsymbol{\psi}_{i^\star} \boldsymbol{\psi}_{i^\star}^H \mathbf{x}$

- mpNet:

$$\mathbf{x} = \mathbf{h} + \mathbf{n} \longrightarrow \boxed{\mathbf{W}^H} \longrightarrow \boxed{\mathrm{HT}_1} \longrightarrow \boxed{\mathbf{W}} \rightarrow \hat{\mathbf{h}}$$

[1] Balatsoukas-Stimming and Studer, "Deep Unfolding for Communications Systems: A Survey and Some New Directions".

- Unsupervised, online, SNR-adaptive neural network. Based on the *Deep-Unfolding*[1] approach.

- MP algorithm (one iteration):
  1. Correlation : $\mathbf{\Psi}^H \mathbf{x}$
  2. Argmax search : $i^\star = \arg\max_i \left|\boldsymbol{\psi}_i^H \mathbf{x}\right|$
  3. Projection : $\hat{\mathbf{h}} = \boldsymbol{\psi}_{i^\star} \boldsymbol{\psi}_{i^\star}^H \mathbf{x}$

- mpNet:

$$\mathbf{x} = \mathbf{h} + \mathbf{n} \longrightarrow \boxed{\mathbf{W}^H} \longrightarrow \boxed{\mathrm{HT}_1} \longrightarrow \boxed{\mathbf{W}} \rightarrow \hat{\mathbf{h}}$$

- Model-based AI: MIMO channel estimation[2], SISO-OFDM (***this paper***), MIMO-ISAC[3], MIMO-OFDM-ISAC-Multi-target[4].

[1] Balatsoukas-Stimming and Studer, “Deep Unfolding for Communications Systems: A Survey and Some New Directions”.

[2] Yassine and Le Magoarou, “mpNet: variable depth unfolded neural network for massive MIMO channel estimation”.

[3] Mateos-Ramos et al., “Model-Driven End-to-End Learning for Integrated Sensing and Communication”.

[4] Mateos-Ramos et al., “Model-Driven End-to-End Learning for Multi-Target Integrated Sensing and Communication”.

Contributions

- **Constrained dictionaries:**

- **Constrained dictionaries:**
  - Reducing the number of learning parameters

- **Constrained dictionaries:**
  - Reducing the number of learning parameters
  - *Without harming the model learning capabilities*

- **Constrained dictionaries:**
  - Reducing the number of learning parameters
  - *Without harming the model learning capabilities*
- **Hierarchical atom search:**

- **Constrained dictionaries:**
  - Reducing the number of learning parameters
  - *Without harming the model learning capabilities*

- **Hierarchical atom search:**
  - Exhaustive search over large dictionaries is computationally heavy

- **Constrained dictionaries:**
  - Reducing the number of learning parameters
  - *Without harming the model learning capabilities*

- **Hierarchical atom search:**
  - Exhaustive search over large dictionaries is computationally heavy
  - Speed-up the search of the most-correlated atom

- High number of learning parameters: long training time. How to reduce the learning parameters number ?

- High number of learning parameters: long training time. How to reduce the learning parameters number ?
- Non-constrained vs. constrained dictionary:

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & \cdots & w_{1,A} \\ \vdots & \ddots & \vdots \\ w_{N,1} & \cdots & w_{N,A} \end{bmatrix} \in \mathbb{C}^{N \times A} \tag{1}$$

- High number of learning parameters: long training time. How to reduce the learning parameters number ?
- Non-constrained vs. constrained dictionary:

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & \cdots & w_{1,A} \\ \vdots & \ddots & \vdots \\ w_{N,1} & \cdots & w_{N,A} \end{bmatrix} \in \mathbb{C}^{N \times A} \tag{1}$$

$$\overset{\diamond}{\mathbf{W}} = \begin{bmatrix} g_1 e^{-\mathrm{j}2\pi\left(f_1 - \frac{N}{2}\delta f\right)\tau_1} & \cdots & g_1 e^{-\mathrm{j}2\pi\left(f_1 - \frac{N}{2}\delta f\right)\tau_A} \\ \vdots & \ddots & \vdots \\ g_N e^{-\mathrm{j}2\pi\left(f_N + \frac{N}{2}\delta f\right)\tau_1} & \cdots & g_N e^{-\mathrm{j}2\pi\left(f_N + \frac{N}{2}\delta f\right)\tau_A} \end{bmatrix} \in \mathbb{C}^{N \times A} \tag{2}$$

- High number of learning parameters: long training time. How to reduce the learning parameters number ?
- Non-constrained vs. constrained dictionary:

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & \cdots & w_{1,A} \\ \vdots & \ddots & \vdots \\ w_{N,1} & \cdots & w_{N,A} \end{bmatrix} \in \mathbb{C}^{N\times A} \quad (1)$$

$$\overset{\diamond}{\mathbf{W}} = \begin{bmatrix} g_1 e^{-\mathrm{j}2\pi\left(f_1 - \frac{N}{2}\delta f\right)\tau_1} & \cdots & g_1 e^{-\mathrm{j}2\pi\left(f_1 - \frac{N}{2}\delta f\right)\tau_A} \\ \vdots & \ddots & \vdots \\ g_N e^{-\mathrm{j}2\pi\left(f_N + \frac{N}{2}\delta f\right)\tau_1} & \cdots & g_N e^{-\mathrm{j}2\pi\left(f_N + \frac{N}{2}\delta f\right)\tau_A} \end{bmatrix} \in \mathbb{C}^{N\times A} \quad (2)$$

  - From $2NA$ parameters to $2N+1$ parameters to learn

- High number of learning parameters: long training time. How to reduce the learning parameters number ?
- Non-constrained vs. constrained dictionary:

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & \cdots & w_{1,A} \\ \vdots & \ddots & \vdots \\ w_{N,1} & \cdots & w_{N,A} \end{bmatrix} \in \mathbb{C}^{N\times A} \tag{1}$$

$$\overset{\diamond}{\mathbf{W}} = \begin{bmatrix} g_1 e^{-j2\pi\left(f_1 - \frac{N}{2}\delta f\right)\tau_1} & \cdots & g_1 e^{-j2\pi\left(f_1 - \frac{N}{2}\delta f\right)\tau_A} \\ \vdots & \ddots & \vdots \\ g_N e^{-j2\pi\left(f_N + \frac{N}{2}\delta f\right)\tau_1} & \cdots & g_N e^{-j2\pi\left(f_N + \frac{N}{2}\delta f\right)\tau_A} \end{bmatrix} \in \mathbb{C}^{N\times A} \tag{2}$$

  - From $2NA$ parameters to $2N+1$ parameters to learn
  - Example: $N = 256$ subcarriers and $A = 990$ atoms $\Rightarrow 506,880$ to 513 parameters

- High number of learning parameters: long training time. How to reduce the learning parameters number ?
- Non-constrained vs. constrained dictionary:

$$\mathbf{W} = \begin{bmatrix} w_{1,1} & \cdots & w_{1,A} \\ \vdots & \ddots & \vdots \\ w_{N,1} & \cdots & w_{N,A} \end{bmatrix} \in \mathbb{C}^{N\times A} \tag{1}$$

$$\overset{\diamond}{\mathbf{W}} = \begin{bmatrix} g_1 e^{-\mathrm{j}2\pi\left(f_1 - \frac{N}{2}\delta f\right)\tau_1} & \cdots & g_1 e^{-\mathrm{j}2\pi\left(f_1 - \frac{N}{2}\delta f\right)\tau_A} \\ \vdots & \ddots & \vdots \\ g_N e^{-\mathrm{j}2\pi\left(f_N + \frac{N}{2}\delta f\right)\tau_1} & \cdots & g_N e^{-\mathrm{j}2\pi\left(f_N + \frac{N}{2}\delta f\right)\tau_A} \end{bmatrix} \in \mathbb{C}^{N\times A} \tag{2}$$

  - From $2NA$ parameters to $2N+1$ parameters to learn
  - Example: $N = 256$ subcarriers and $A = 990$ atoms $\Rightarrow 506,880$ to 513 parameters

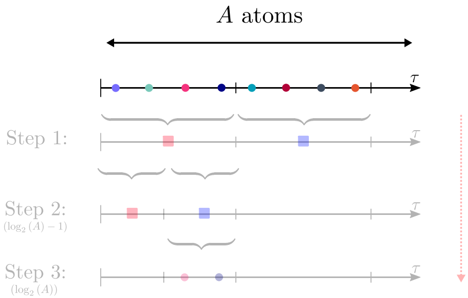
Learning parameter number is independent of the number of atoms

- Currently, correlation of the whole dictionary with the residual and argmax search.
  - Computationally intensive: How to speed up the process ?

- Currently, correlation of the whole dictionary with the residual and argmax search.
    - Computationally intensive: How to speed up the process ?
- New idea: Use a hierarchical approach.

- Currently, correlation of the whole dictionary with the residual and argmax search.
  - Computationally intensive: How to speed up the process ?

- New idea: Use a hierarchical approach.



Classical approach:

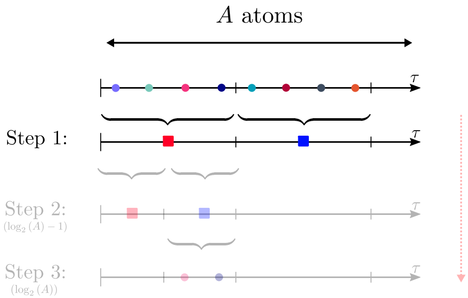
- 1 step
- A correlations

Hierarchical approach:

- $\log_2(A)$ steps
- $2\log_2(A)$ correlations

- Currently, correlation of the whole dictionary with the residual and argmax search.
  - Computationally intensive: How to speed up the process ?

- New idea: Use a hierarchical approach.



Classical approach:

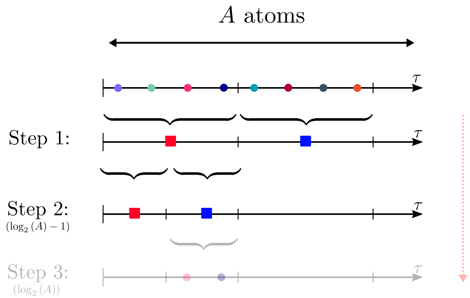
- 1 step
- A correlations

Hierarchical approach:

- $\log_2(A)$ steps
- $2\log_2(A)$ correlations

- Currently, correlation of the whole dictionary with the residual and argmax search.
  - Computationally intensive: How to speed up the process ?

- New idea: Use a hierarchical approach.



Classical approach:

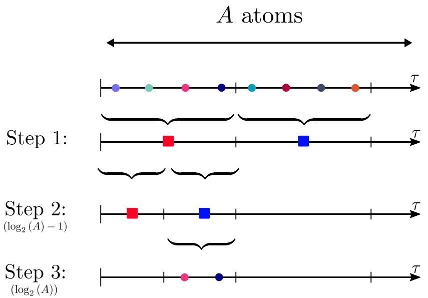
- 1 step
- A correlations

Hierarchical approach:

- $\log_2(A)$ steps
- $2\log_2(A)$ correlations

- Currently, correlation of the whole dictionary with the residual and argmax search.
  - Computationally intensive: How to speed up the process ?

- New idea: Use a hierarchical approach.



Classical approach:

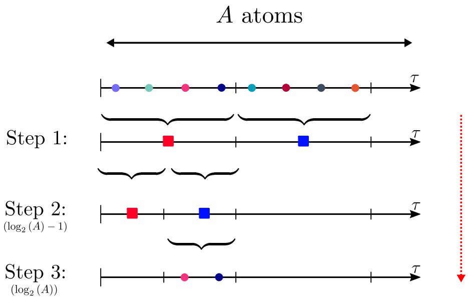
- 1 step
- A correlations

Hierarchical approach:

- $\log_2(A)$ steps
- $2\log_2(A)$ correlations

- Currently, correlation of the whole dictionary with the residual and argmax search.
  - Computationally intensive: How to speed up the process ?

- New idea: Use a hierarchical approach.



Classical approach:

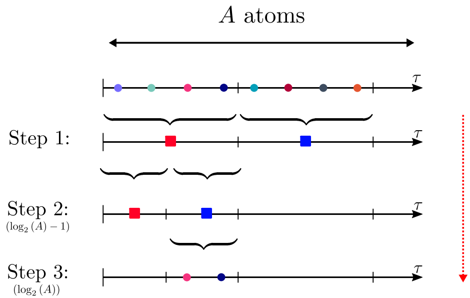
- 1 step
- A correlations

Hierarchical approach:

- $\log_2(A)$ steps
- $2\log_2(A)$ correlations

- Currently, correlation of the whole dictionary with the residual and argmax search.
  - Computationally intensive: How to speed up the process ?

- New idea: Use a hierarchical approach.



Classical approach:

- 1 step
- A correlations

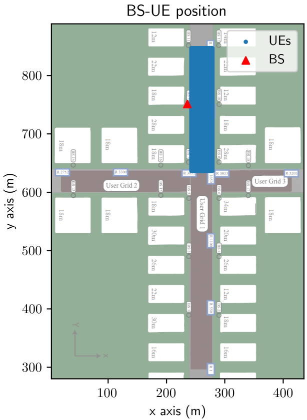
Hierarchical approach:

- $\log_2(A)$ steps
- $2\log_2(A)$ correlations

Correlation number is divided by $\frac{A}{2\log_2(A)}$

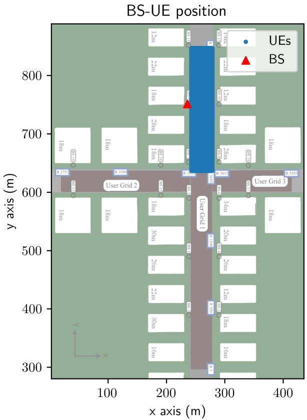
Empirical results

- DeepMIMO configuration
  - $f_0 = 3.4$GHz
  - BW = 50MHz
  - $N = 256$ subcarriers
  - Variable $\mathrm{SNR}_{\mathrm{in}}$

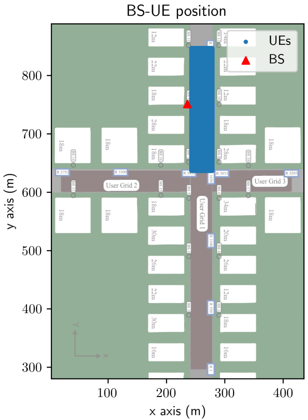


- DeepMIMO configuration
  - $f_0 = 3.4$GHz
  - BW = 50MHz
  - $N = 256$ subcarriers
  - Variable SNR_{in}

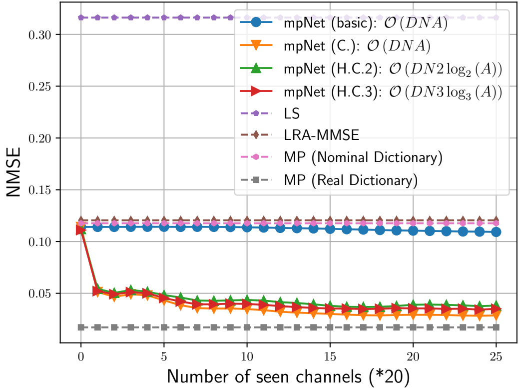
- Imperfection models:
  - SCO: $f_i = \tilde{f}_i + i\delta f$
  - Gain imperfection: $g_i = \tilde{g}_i + n_{g_i}, \, n_{g_i} \sim \mathcal{N}\left(0, \sigma_g^2\right)$



- DeepMIMO configuration
  - $f_0 = 3.4$GHz
  - BW = 50MHz
  - $N = 256$ subcarriers
  - Variable $\mathrm{SNR}_{\mathrm{in}}$
- Imperfection models:
  - SCO: $f_i = \tilde{f}_i + i\delta f$
  - Gain imperfection: $g_i = \tilde{g}_i + n_{g_i}$, $n_{g_i} \sim \mathcal{N}\left(0, \sigma_g^2\right)$
- Online (minibatch) learning
  - 10 channels per batch
  - 2000 test channels

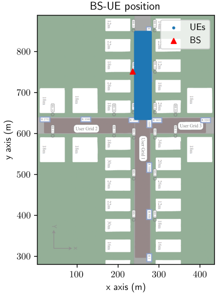


- DeepMIMO channels @3.4GHz, $N = 256$ subcarriers



SNR$_{\text{in}} = 5\,\text{dB}$, $\sigma_g^2 = 0.09$, $\xi = 40$ ppm

$$\text{NMSE} = \frac{\mathbb{E}\left[\left\|\hat{\mathbf{h}} - \mathbf{h}\right\|_2^2\right]}{\|\mathbf{h}\|_2^2}$$



- Contributions:
  - Sample complexity reduction: constrained dictionaries
  - Time complexity reduction: hierarchical search

- Link to paper: https://arxiv.org/pdf/2210.06588.pdf or QR-code:

Thank you!
Have you got any questions?

Thanks