Deep Learning based Precoding for the MIMO Gaussian Wiretap Channel

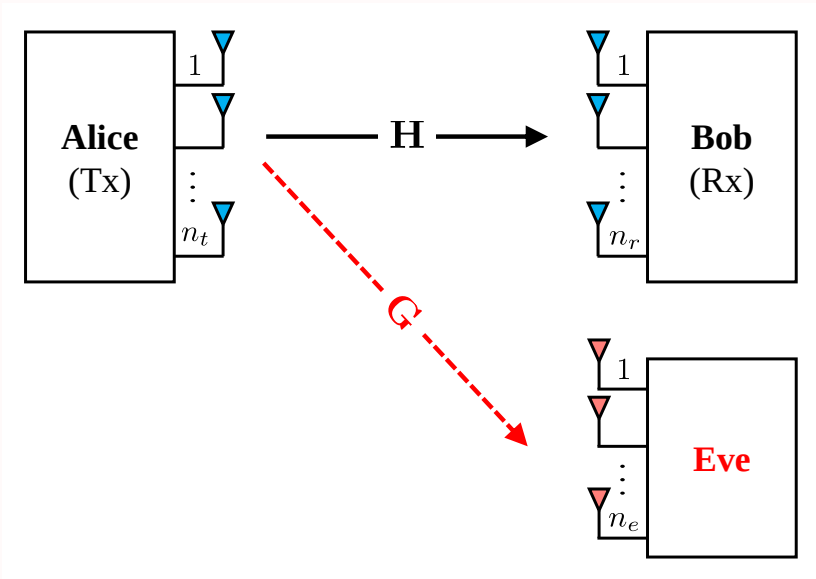
Xinliang Zhang and Mojtaba Vaezi

Department of Electrical and Computer Engineering

VILLANOVA UNIVERSITY

IEEE Global Communications Conference Workshop
Hawaii, USA
Dec. 13, 2019

Channel Model

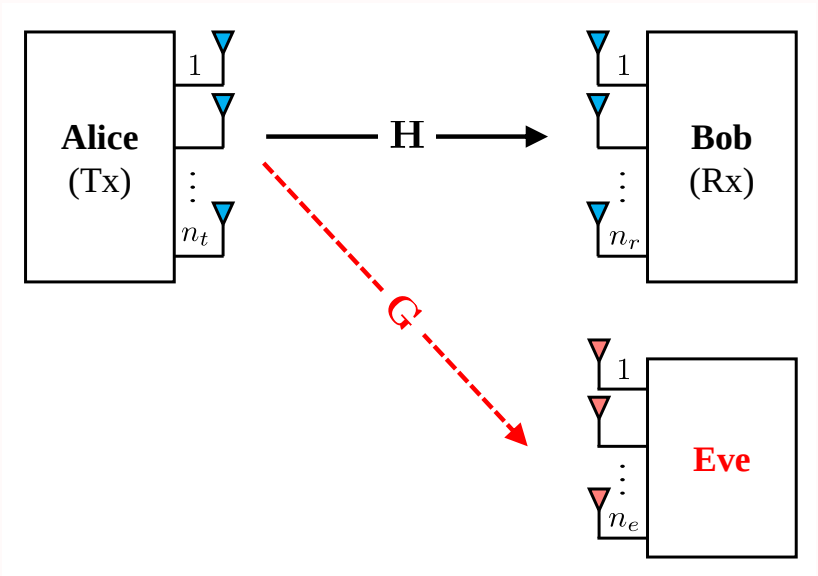


Multiple-input multiple-output (MIMO) Gaussian wiretap channel

$$\mathbf{y}_r = \mathbf{H}\,\mathbf{x} + \mathbf{w}_r$$
$$\mathbf{y}_e = \mathbf{G}\,\mathbf{x} + \mathbf{w}_e$$

- $\mathbf{w}_r$ and $\mathbf{w}_e$ are i.i.d. Gaussian noise vectors
- $\mathrm{tr}(\mathbb{E}\{\mathbf{x}\mathbf{x}^T\}) = \mathrm{tr}(\mathbf{Q}) \leq P$ (average power constraint)
- $\mathbf{H}$ and $\mathbf{G}$ are given

Channel Model



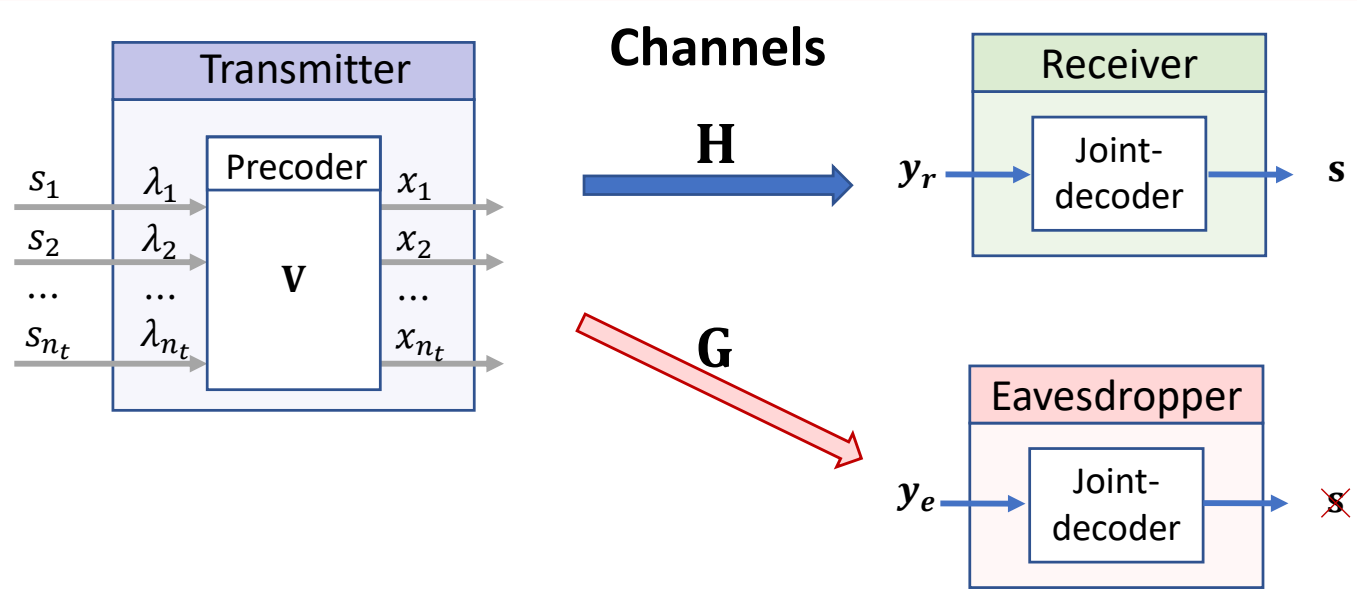
Multiple-input multiple-output (MIMO) Gaussian wiretap channel

$$\mathbf{y}_r = \mathbf{H}\,\mathbf{x} + \mathbf{w}_r$$
$$\mathbf{y}_e = \mathbf{G}\,\mathbf{x} + \mathbf{w}_e$$

- $\mathbf{w}_r$ and $\mathbf{w}_e$ are i.i.d. Gaussian noise vectors
- $\mathrm{tr}(\mathbb{E}\{\mathbf{x}\mathbf{x}^T\}) = \mathrm{tr}(\mathbf{Q}) \le P$ (average power constraint)
- $\mathbf{H}$ and $\mathbf{G}$ are given

Secrecy capacity [Khisti-Wornell'08] [Oggier-Hassibi'08]

$$\max_{\mathbf{Q}} \quad \frac{1}{2}\log\frac{|\mathbf{I}_{n_r} + \mathbf{H}\mathbf{Q}\mathbf{H}^T|}{|\mathbf{I}_{n_e} + \mathbf{G}\mathbf{Q}\mathbf{G}^T|}$$
$$\text{s.t.} \quad \mathbf{Q} \succeq \mathbf{0},\ \mathbf{Q} = \mathbf{Q}^T,\ \mathrm{tr}(\mathbf{Q}) \le P$$



Precoding for the three-nodes wireless system

The Q in precoding

$$\mathbb{E}\{\mathbf{s}\mathbf{s}^T\} = \mathbf{I}, \text{ (i.i.d)}$$

$$\mathbf{x} = \mathbf{V}\mathbf{\Lambda}^{\frac{1}{2}}\mathbf{s}$$

$$\mathbf{Q} \triangleq \mathbb{E}\{\mathbf{x}\mathbf{x}^T\} = \mathbf{V}\mathbf{\Lambda}\mathbf{V}^T$$

- Precoding matrix **V**
  Rotation model [Zhang-Qi-Vaezi'19]
  (arxiv.org/abs/1908.00994)
- Power allocation matrix

$$\mathbf{\Lambda} \triangleq \begin{bmatrix} \lambda_1^2 & & \\ & \ddots & \\ & & \lambda_{n_t}^2 \end{bmatrix}$$

Existing Solutions for **Q**

This problem is still difficult because it is non-convex and existing solutions are:

- not effective for all antenna settings
- iterative (time consuming)

Existing Solutions for **Q**

This problem is still difficult because it is non-convex and existing solutions are:

- not effective for all antenna settings
- iterative (time consuming)

Closed-form solution is known only for

- MISO case ($n_r = 1$) [Khisti-Wornell'10]
- $n_t = n_r = 2$ and $n_e = 1$ [Shafiee-Liu-Ulukus'09]
- Strictly degraded channel ($\mathbf{H}^H\mathbf{H} \succ \mathbf{G}^H\mathbf{G}$) only if $P > P_0$ [Loyka-Charalambous'16][Fakoorian-Swindlehurst'13]
- $n_t = 2$ [Vaezi-Shin-Poor'17]

Existing Solutions for **Q**

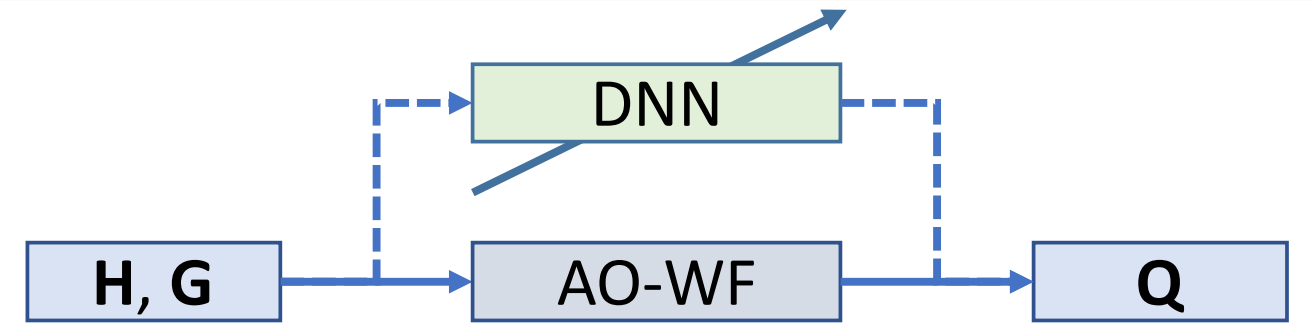
Existing numerical solutions

- Generalized singular value decomposition (GSVD) based beamforming [Fakoorian-Swindlehurst'12]
- Alternative optimization and water-filling (AO-WF) [Li-Hong-Wai-Liu-Ma-Luo'13]
- Rotation model [Zhang-Qi-Vaezi'19]

Problems

1. May provide a sub-optimal solution
2. Time cost usually high

This Talk

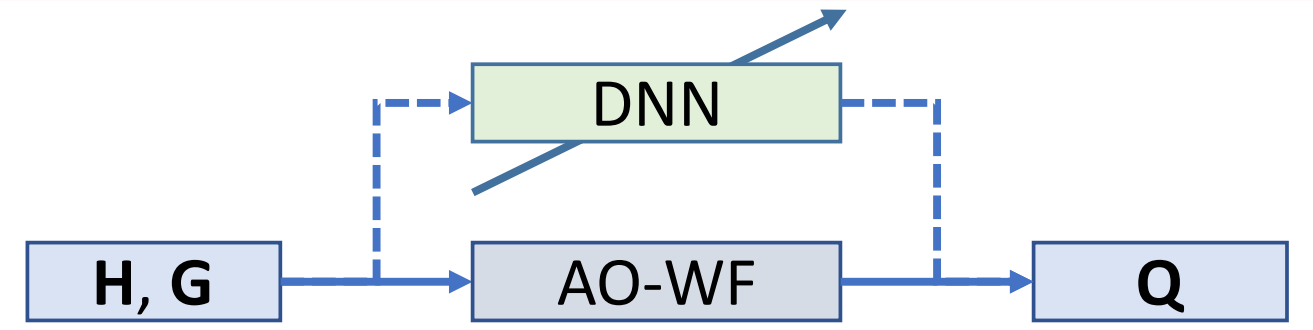


Training process

Why do we use deep neural network (DNN)?

1. **Fast**: finite matrices computation, no iterations
2. **Feasible to learn**: input/output mapping
3. **Easy data generation**: build data set including random channels and find corresponding optimal **Q**

This Talk



Training process

The proposed method

1. Build data set using numerical method (**off line**)
   − high time cost + close to the global optimal
2. Train the DNN (**off line**)
   − high time cost + accurate regression
3. Implement the well trained DNN (**on the fly**)
   + time efficient + close to the global optimal

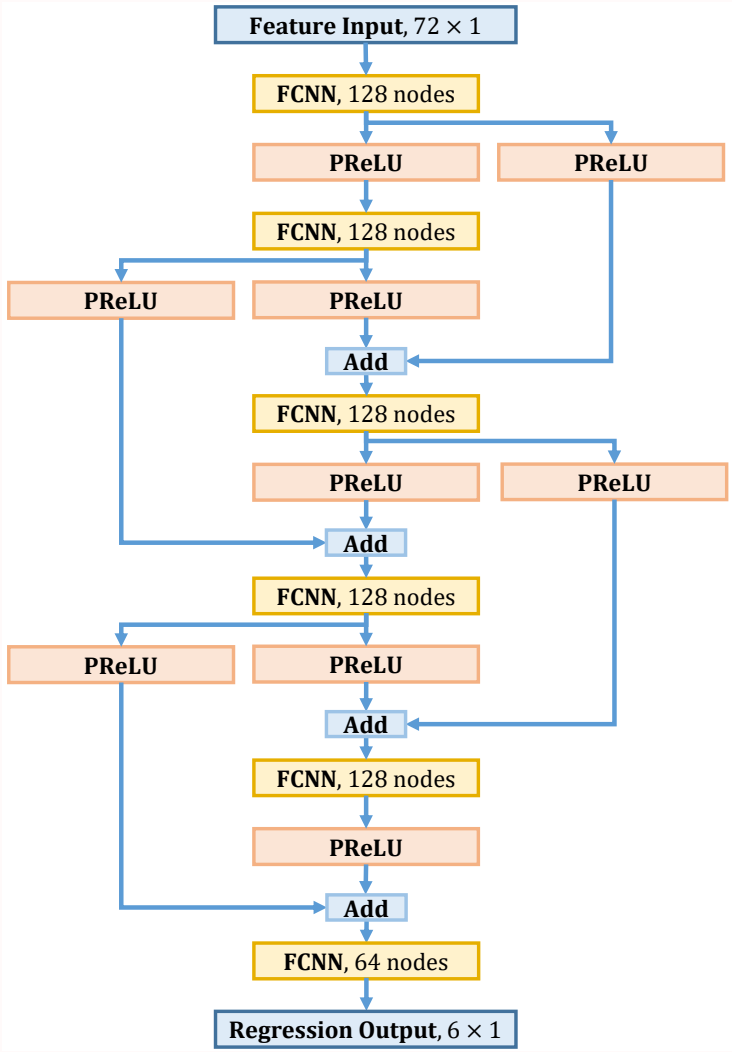
DNN design (for $n_t = 3$)

Components

- DNN architecture
  - Fully-connected neural networks (FCNN)
  - Activation functions
  - Residual connections (short cuts)

Ideas of the design

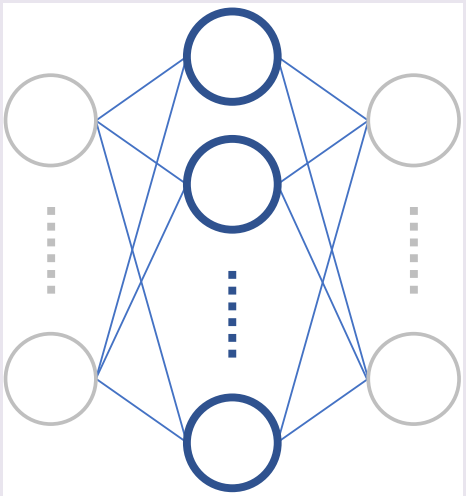
- **Non-linear**: to be able to learn the mapping
- **Simple**: small memory and computation, avoid over-fitting



Proposed DNN architecture

DNN architecture

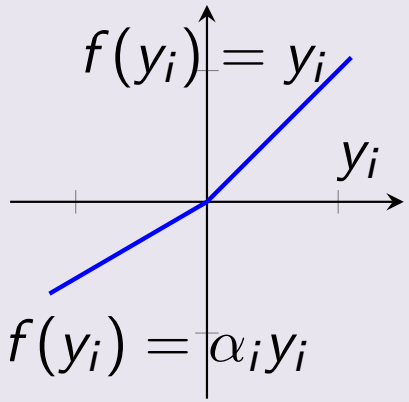
FCNN



Fully-connected neural network

Activation function

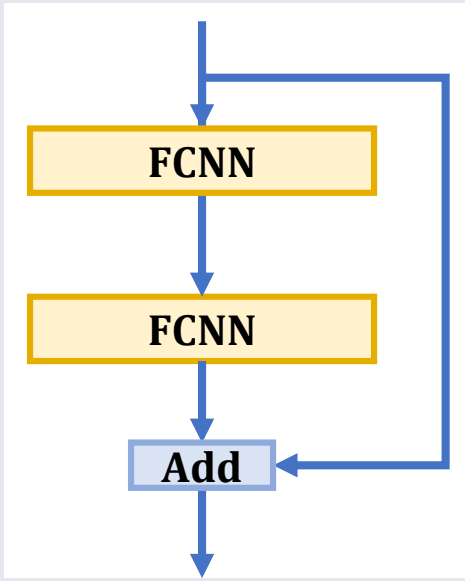
Parametric Rectified Linear Unit (PReLU)
[He-Zhang-Ren-Sun'15]



- Additional adaptive parameters
- Less computation

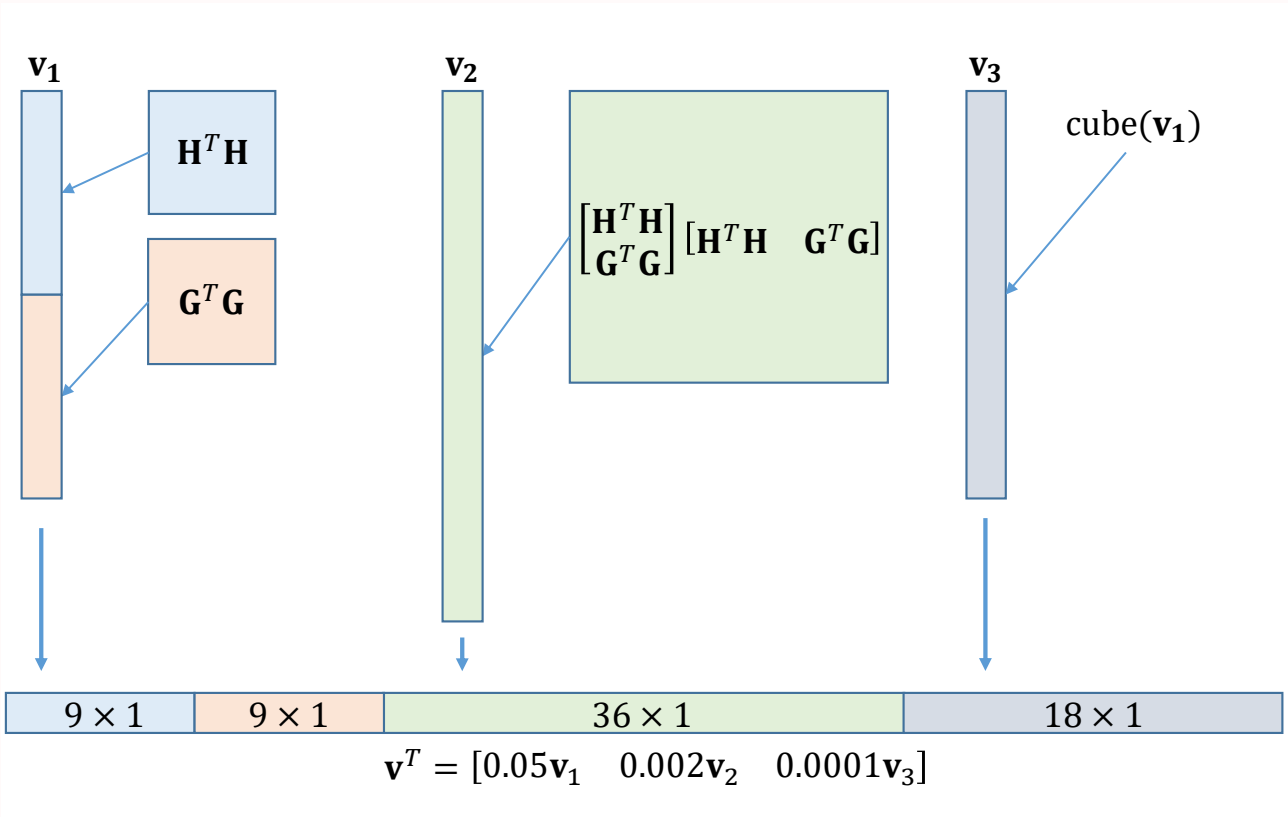
Residual connection

[He-Zhang-Ren-Sun'16]



- Increases complexity
- Converges faster

DNN input feature design

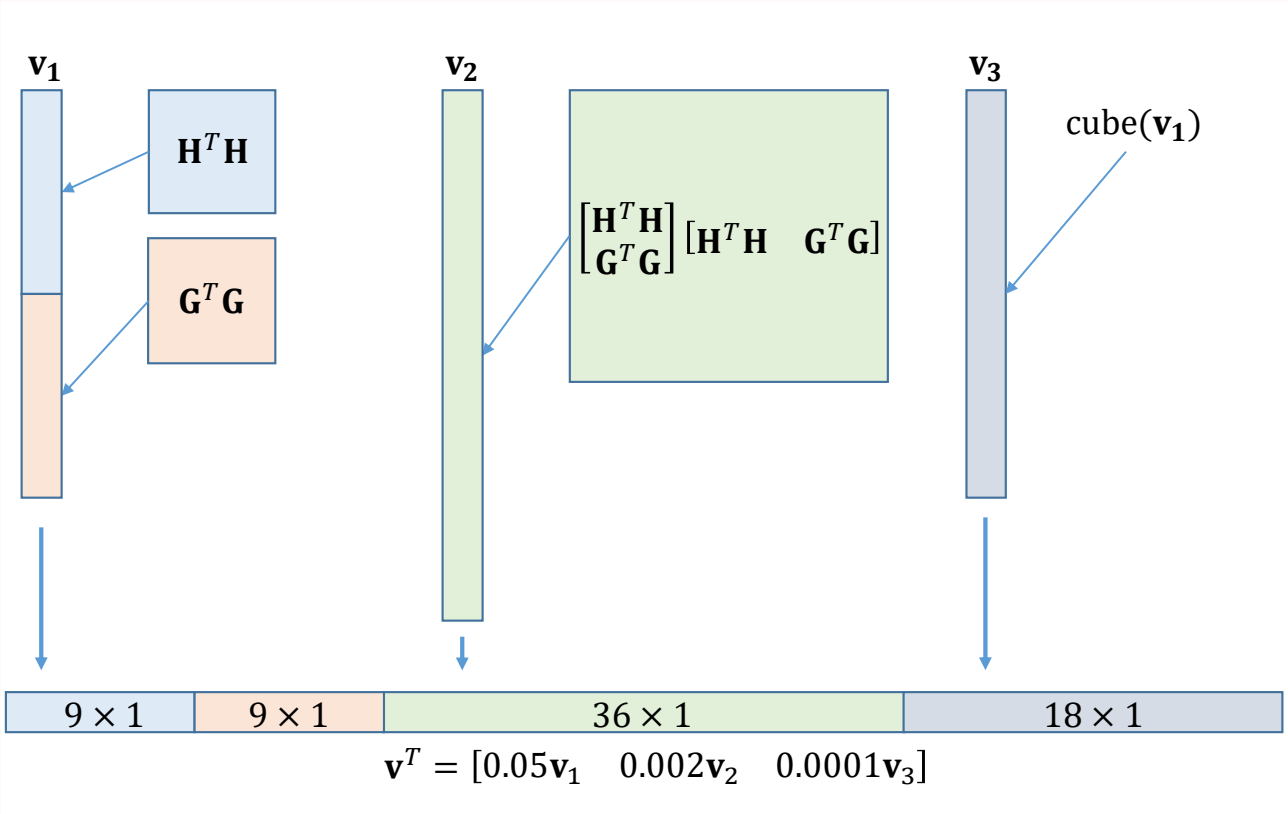


Secrecy capacity

$$\max_{\mathbf{Q}} \quad \frac{1}{2}\log\frac{|\mathbf{I}_{n_t}+\mathbf{H}^T\mathbf{H}\mathbf{Q}|}{|\mathbf{I}_{n_t}+\mathbf{G}^T\mathbf{G}\mathbf{Q}|}$$

$$\text{s.t.} \quad \mathbf{Q}\succeq\mathbf{0},\ \mathbf{Q}=\mathbf{Q}^T,\ \text{tr}(\mathbf{Q})\le P$$

DNN input feature design



Principles of input design

- The solution is only related to $\mathbf{H}^T\mathbf{H}$ and $\mathbf{G}^T\mathbf{G}$
- Nonlinear features can improve the performance of regression, since the DNN is a basicaly linear system

DNN output layer

Elements in the output vector (for $n_t = 3$)

Regression Output, 6×1

$$\mathbf{q} \triangleq [q_{11}, q_{22}, q_{33}, q_{12}, q_{23}, q_{13}]^T$$

$$\mathbf{Q}^* = \begin{bmatrix} q_{11} & q_{12} & q_{13} \\ q_{12} & q_{22} & q_{23} \\ q_{13} & q_{23} & q_{33} \end{bmatrix}$$

Symmetric matrix

Data set

Data Set construction

For each sample of data set,

- Generate **H** and **G** randomly (Gaussian)
- Find the optimal **Q** using AO-WF or rotation method
- Record input and output vectors

Table: Details of the data sets.

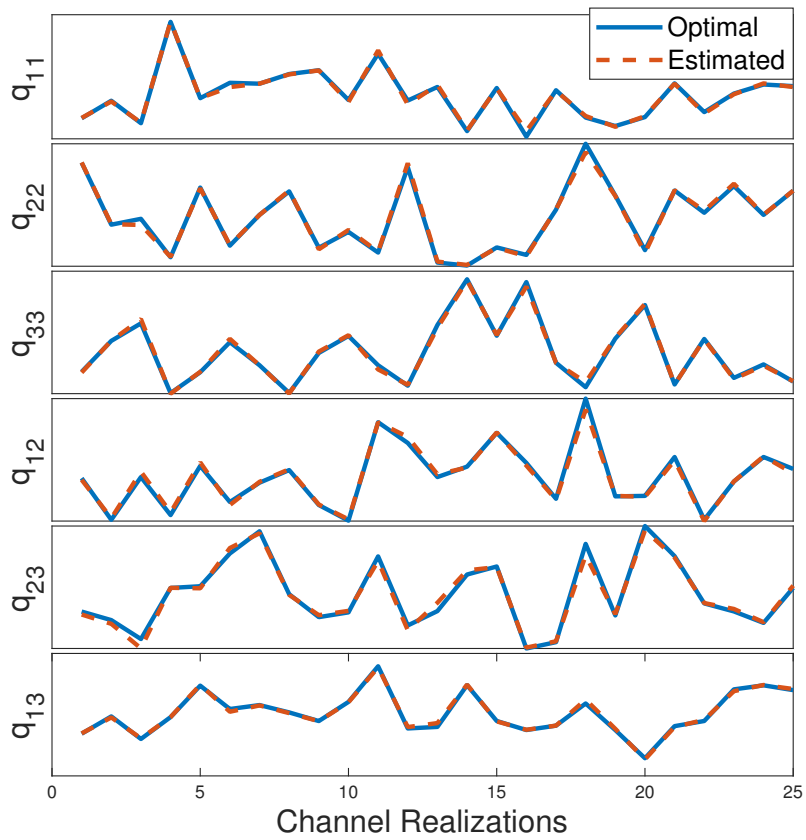
	n_t	n_r	n_e	number of samples
TrainingSet-I	3	2	1	2,000,000
TrainingSet-II	3	4	3	2,000,000
TrainingSet-III	Cascade of *TrainingSet-I* and *TrainingSet-II*			
TestSet-I	3	2	1	1000
TestSet-II	3	4	3	1000

Simulation Results

Accuracy of regression

Table: Mean square error (MSE) of the regression.

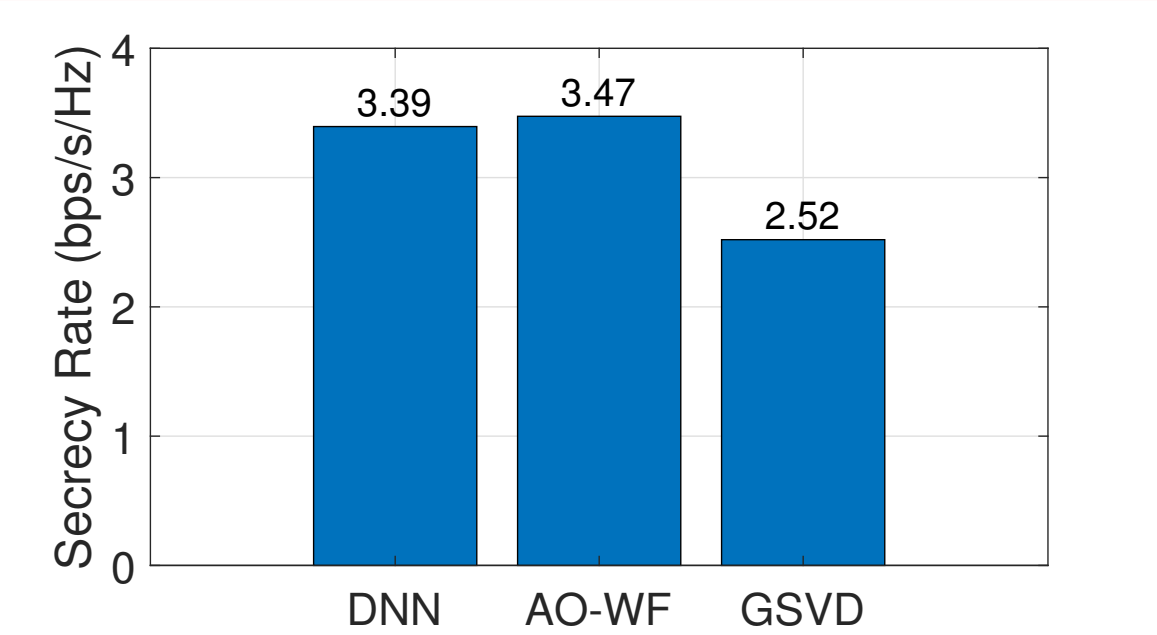
	TrainingSet-III	
	TestSet-I	*TestSet-II*
$\hat{q}_{11}$	0.1313	0.1564
$\hat{q}_{22}$	0.1234	0.1386
$\hat{q}_{33}$	0.1219	0.1364
$\hat{q}_{12}$	0.1526	0.1351
$\hat{q}_{23}$	0.1057	0.1596
$\hat{q}_{13}$	0.1384	0.1123



Comparison between the DNN output and the ideal values

Simulation Results

Achievable secrecy rate



Averaged time cost

Table: Average time for one realization.

	DNN	AO-WF	GSVD
Time Cost (ms)	**0.0255**	19.49	0.513

Simulation Results

Promising for the internet of things (IoT) devices

Limited computational ability and battery life
Require low delay and high speed

Averaged time cost

Table: Average time for one realization.

	DNN	AO-WF	GSVD
Time Cost (ms)	**0.0255**	19.49	0.513

Conclusions

Summary

- Problem: secure transmission over Gaussian wiretap channel
- Solution: DNN-based precoding
- Advantage: fast and reliable, efficient for IoT devices

Future Work

- Train for arbitrary n_r and n_e
- Generalize to complex channel
- Multi-task precoding realization

Thank you!