OTFS Transceivers Design using Deep Neural Networks

OTFS Seminar Series

OTFS Special Interest Group

A. Chockalingam
Indian Institute of Science, Bangalore

Special thanks to Bharath S, Ashwitha N, Sandesh R M

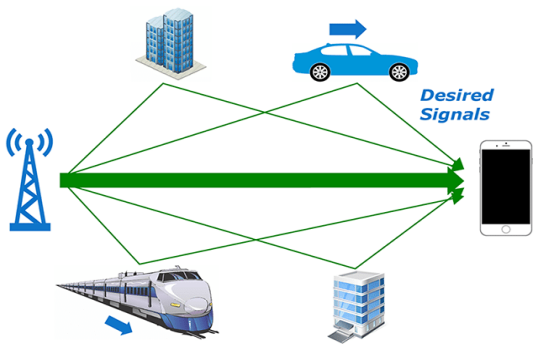
23 February 2022

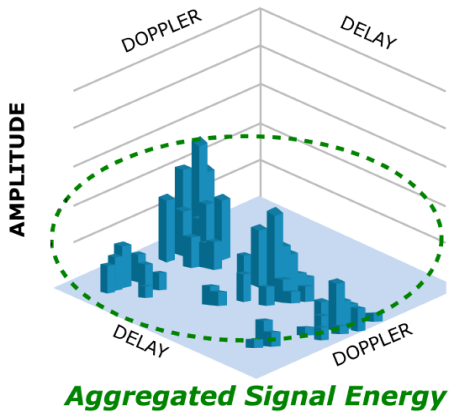
Outline I

OTFS modulation

Orthogonal Time Frequency Space (OTFS) modulation*

- A promising modulation scheme for doubly-selective channels
- Information is multiplexed in the delay-Doppler (DD) domain
  - Map information from DD domain to time domain and transmit
  - Direct approach:
    - use inverse Zak transform: DD domain $\rightarrow$ time domain
  - Two-step approach:
    - use ISFFT & Heisenberg transforms: DD domain $\rightarrow$ TF domain $\rightarrow$ time domain
- Channel is viewed/represented in DD domain
- Superior performance compared to OFDM

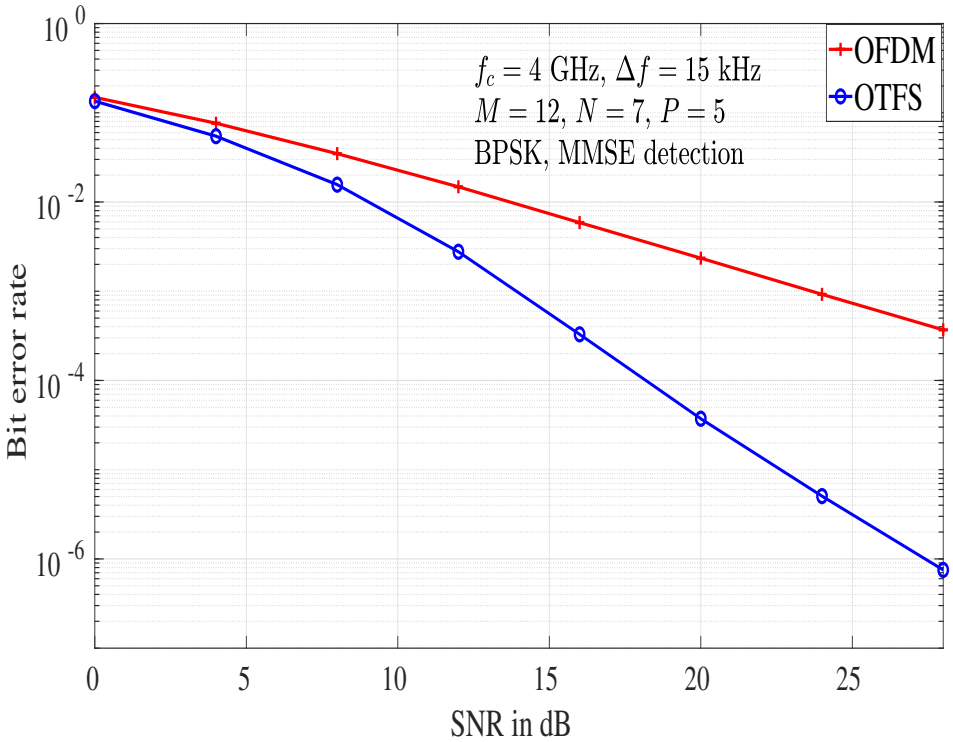




(*) R. Hadani, S. Rakib, M. Tsatsanis, A. Monk, A. J. Goldsmith, A. F. Molisch, and R. Calderbank, "Orthogonal time frequency space modulation," in *Proc. IEEE WCNC*, San Francisco, CA, USA, March 2017.

Why OTFS?

- OTFS vs OFDM performance



Parameter	Value
Carrier frequency (GHz)	4
Subcarrier spacing (kHz)	15
Frame size (M, N)	$(12, 7)$
Number of paths (P)	5
Delay profile	Exponential
Maximum speed (km/h)	500
Maximum Doppler (Hz)	1875
Modulation scheme	BPSK

* Smallest resource block used in LTE: $M = 12$, $N = 7$

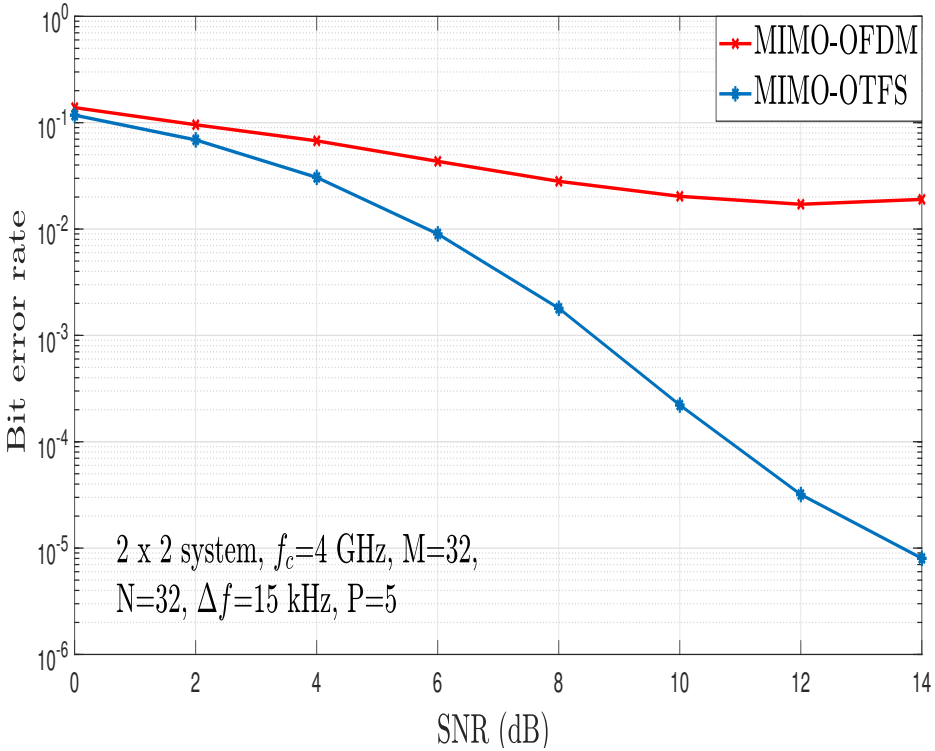
- MMSE detection

- OFDM performs poor due to Doppler induced ICI
- OTFS performs significantly better than OFDM

(*) G. D. Surabhi, R. M. Augustine, and A. Chockalingam, "On the diversity of uncoded OTFS modulation in doubly-dispersive channels," *IEEE Trans. Wireless Commun.*, vol. 18, no. 6, pp. 3049-3063, Jun. 2019.

Why OTFS?

- MIMO-OTFS vs MIMO-OFDM performance



Parameter	Value
Carrier frequency (GHz)	4
Subcarrier spacing (kHz)	15
Frame size (M, N)	$(32, 32)$
Modulation scheme	BPSK
MIMO configuration	2×2
Maximum speed (km/h)	507.6

Path index (i)	1	2	3	4	5
Delay (τ_i,μs)	2.1	4.2	6.3	8.4	10.4
Doppler (ν_i,Hz)	0	470	940	1410	1880

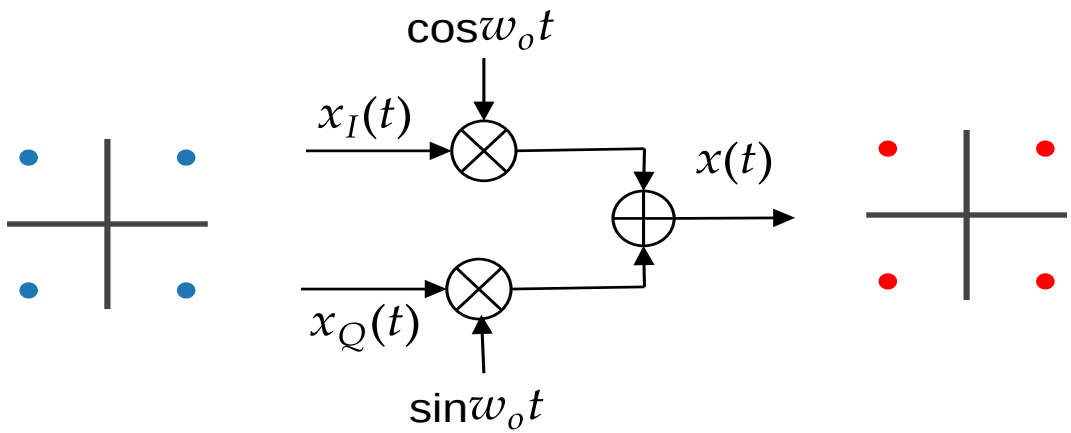
- Message passing detection

- MIMO-OTFS performs significantly better than MIMO-OFDM

(*) M. K. Ramachandran and A. Chockalingam, "MIMO-OTFS in High-Doppler Fading Channels: Signal Detection and Channel Estimation," *IEEE GLOBECOM'2018*, December 2018.

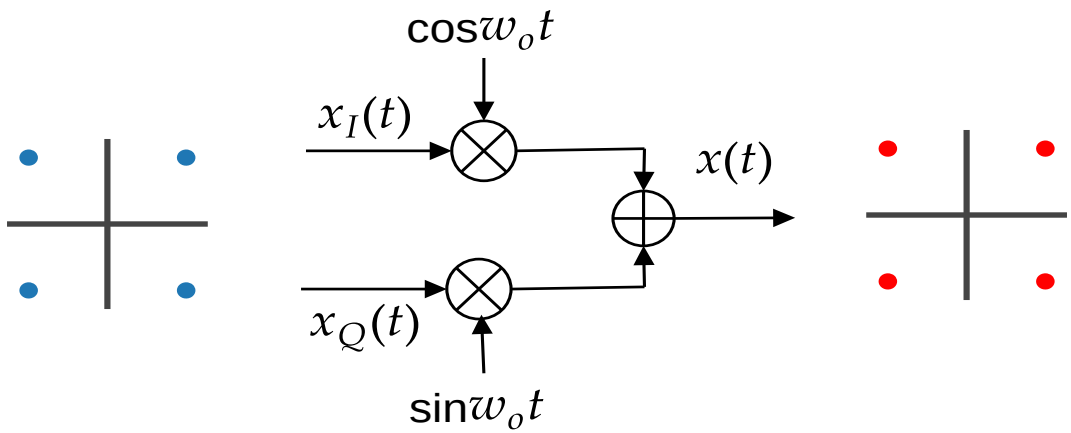
Why OTFS?

- Effect of IQ imbalance at the Tx and Rx
- Ideal Tx chain

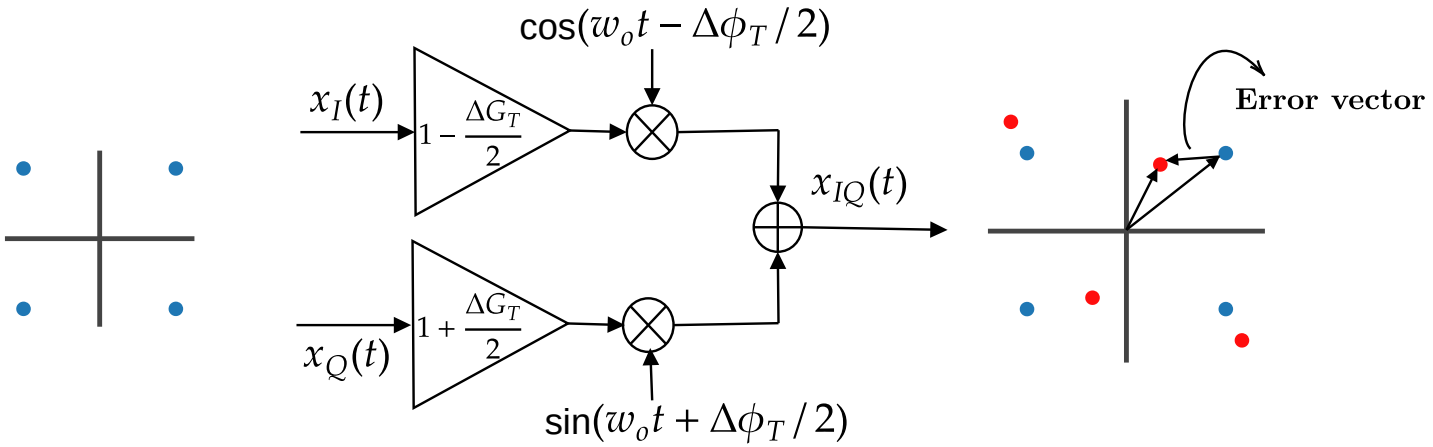


Why OTFS?

- Effect of IQ imbalance at the Tx and Rx
- Ideal Tx chain

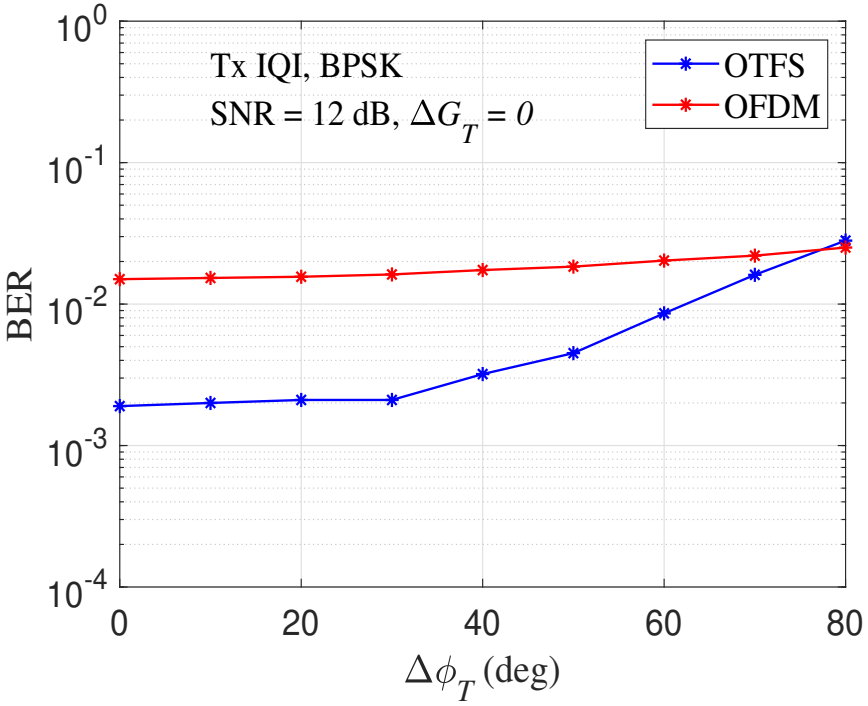


- Gain and phase imbalance in Tx chain $(\Delta G_T, \Delta\phi_T)$

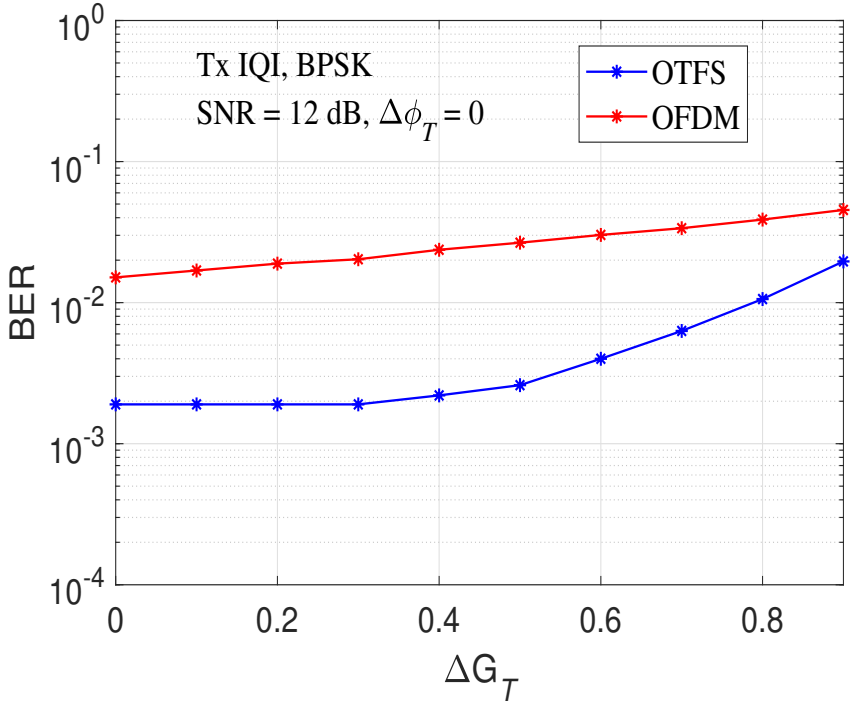


Why OTFS?

- Performance of OTFS and OFDM in the presence of Tx IQI



((a)) $\Delta\phi_T$ sensitivity



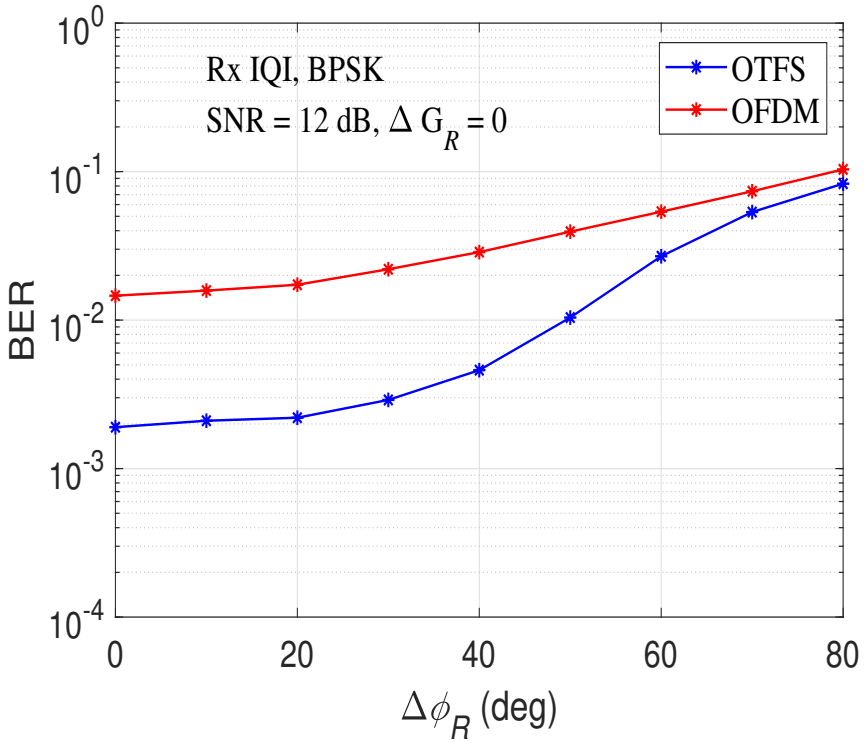
((b)) ΔG_T sensitivity

- OTFS is more robust to Tx IQI than OFDM

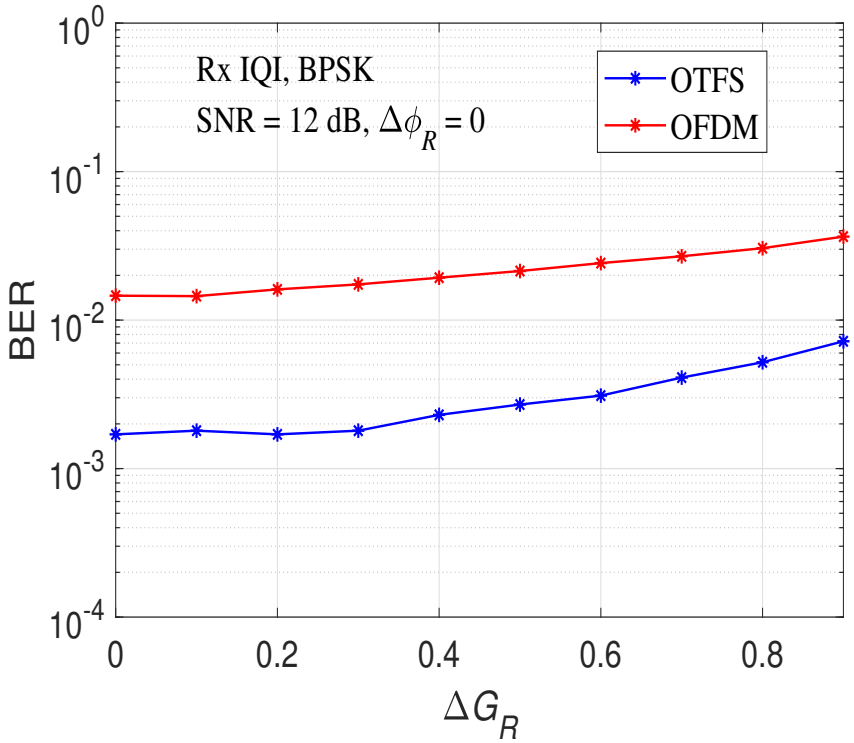
[1] Ashwitha Naikoti and A. Chockalingam, "A DNN-Based OTFS Transceiver With Delay-Doppler Channel Training and IQI Compensation," *IEEE PIMRC'2021*, September 2021.

Why OTFS?

- Performance of OTFS and OFDM in the presence of Rx IQI



((c)) $\Delta\phi_R$ sensitivity

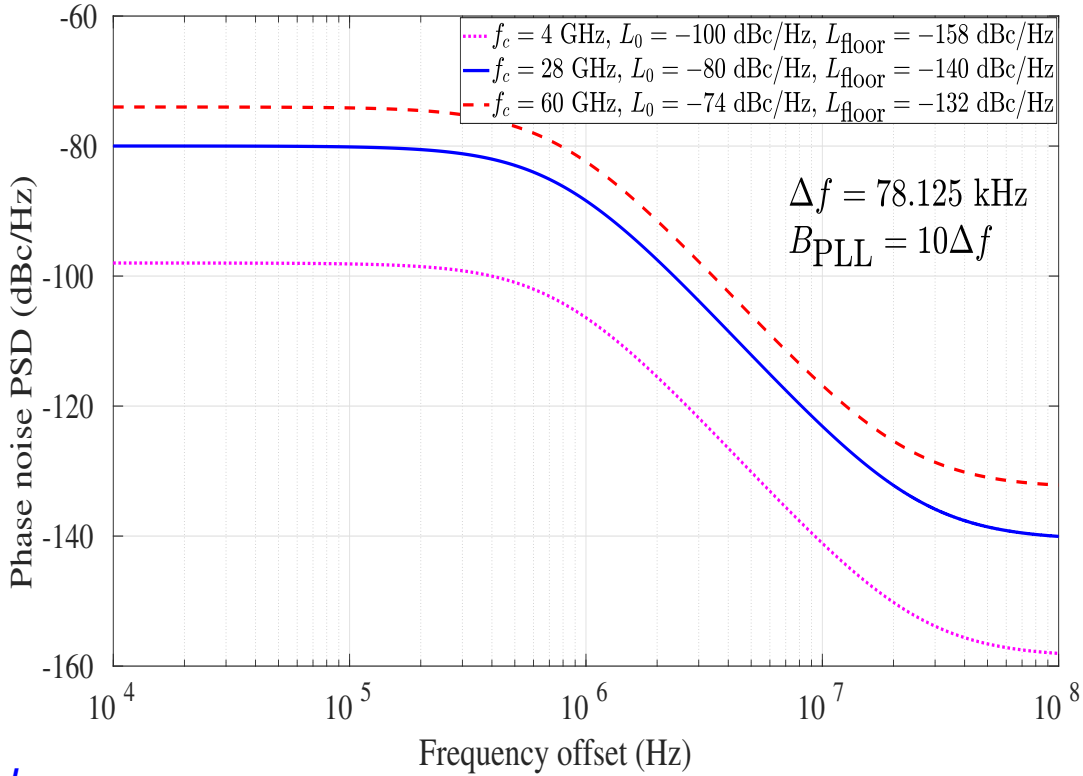


((d)) ΔG_R sensitivity

- OTFS is more robust to Rx IQI than OFDM

Why OTFS?

- Effect of oscillator phase noise
- Oscillator phase noise spectrum at diff. carrier frequencies (4, 28, 60 GHz)[1]



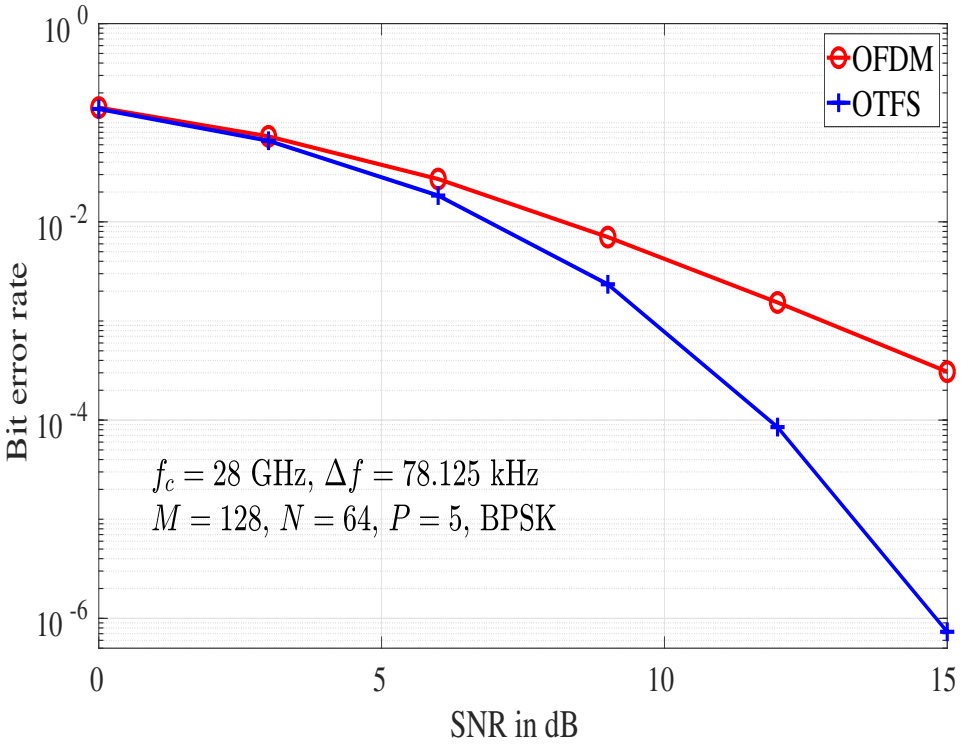
- $L(f) = \frac{B_{\text{PLL}}^2 L_0}{B_{\text{PLL}}^2 + f^2} + L_{\text{floor}}$

[1] 3GPP R1-163984: Discussion on phase noise modeling, 3GPP TSG RAN WG1 85, May 2016.

[2] L. Smaini, RF Analog Impairments Modeling for Communication Systems Simulation: Application to OFDM-Based Transceivers, Wiley, 2012

Why OTFS?

- OTFS and OFDM performance with phase noise



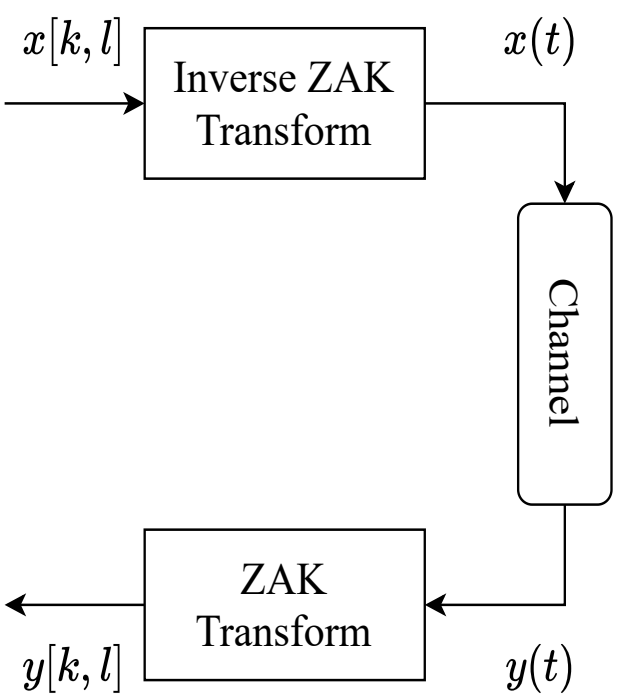
Parameter	Value
Carrier frequency (GHz)	28
Bandwidth (MHz)	10
Subcarrier spacing, Δf (kHz)	78.125
Frame size (M, N)	(128,64)
Rice factor (dB)	13
B_{PLL}	10 Δf
Modulation	BPSK
Number of taps, P	5

Path index (i)	1	2	3	4	5
Delay (τ_i,μs)	0.3	1	1.7	2.4	3.1
Doppler (ν_i,Hz)	0	−400	400	1220	1220

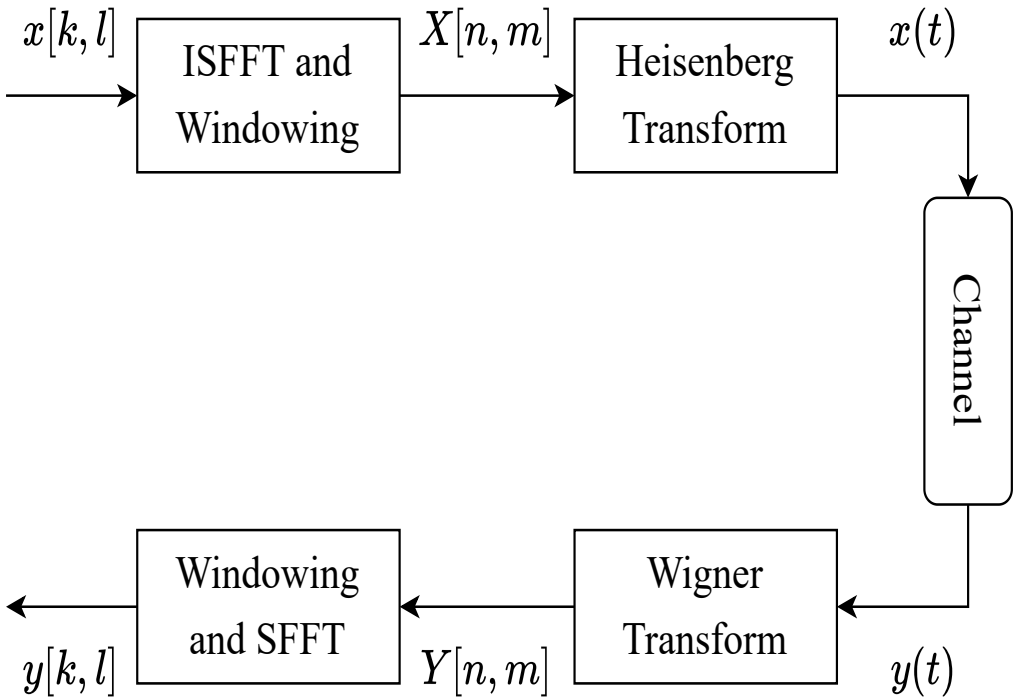
- Message passing detection
- OTFS is more robust to phase noise

G. D. Surabhi, M. K. Ramachandran, and A. Chockalingam, "OTFS modulation with phase noise in mmWave communications," *Proc. IEEE VTC'2019-Spring*, Kuala Lumpur, Apr. 2019.

OTFS - Signaling in DD domain



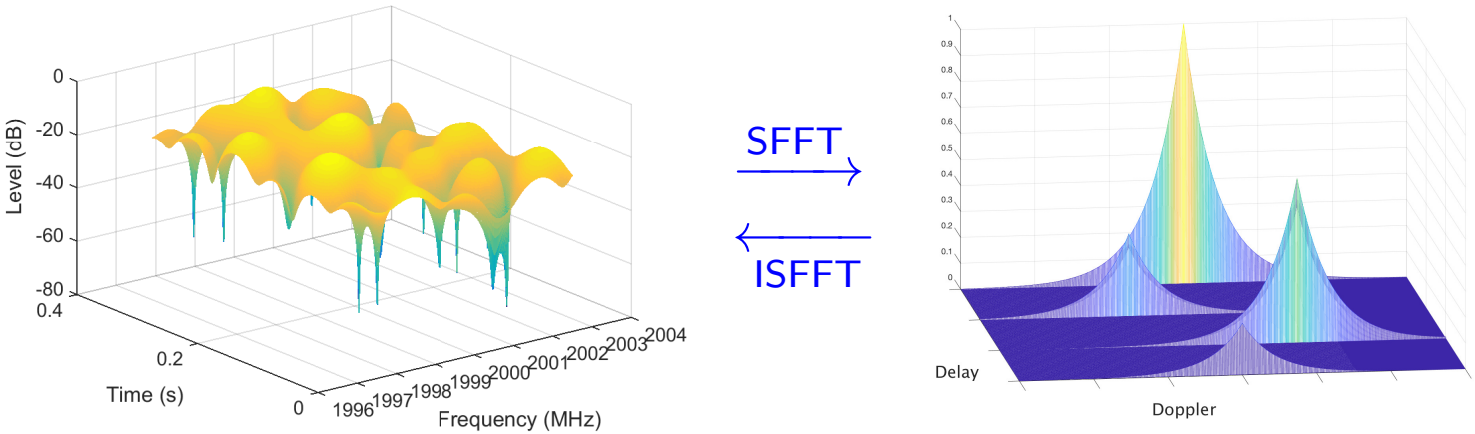
Direct approach



Two-step approach

Channel in DD domain

- DD domain impulse response $h(\tau, \nu)$ is compact, sparse, stable
  - channel taps in DD representation correspond to a cluster of reflectors with specific delay and Doppler values
  - the delay and Doppler values depend on reflectors' relative distance and relative velocity, respectively, with the transmitter and receiver
  - relative velocity and distance remain roughly constant for at least a few msecs



Channel in time-frequency $H(t, f)$ and delay-Doppler $h(\tau, \nu)$ domains

For a channel with P paths in the DD domain

$$h(\tau, \nu) = \sum_{i=1}^{P} h_i \delta(\tau - \tau_i)\delta(\nu - \nu_i)$$

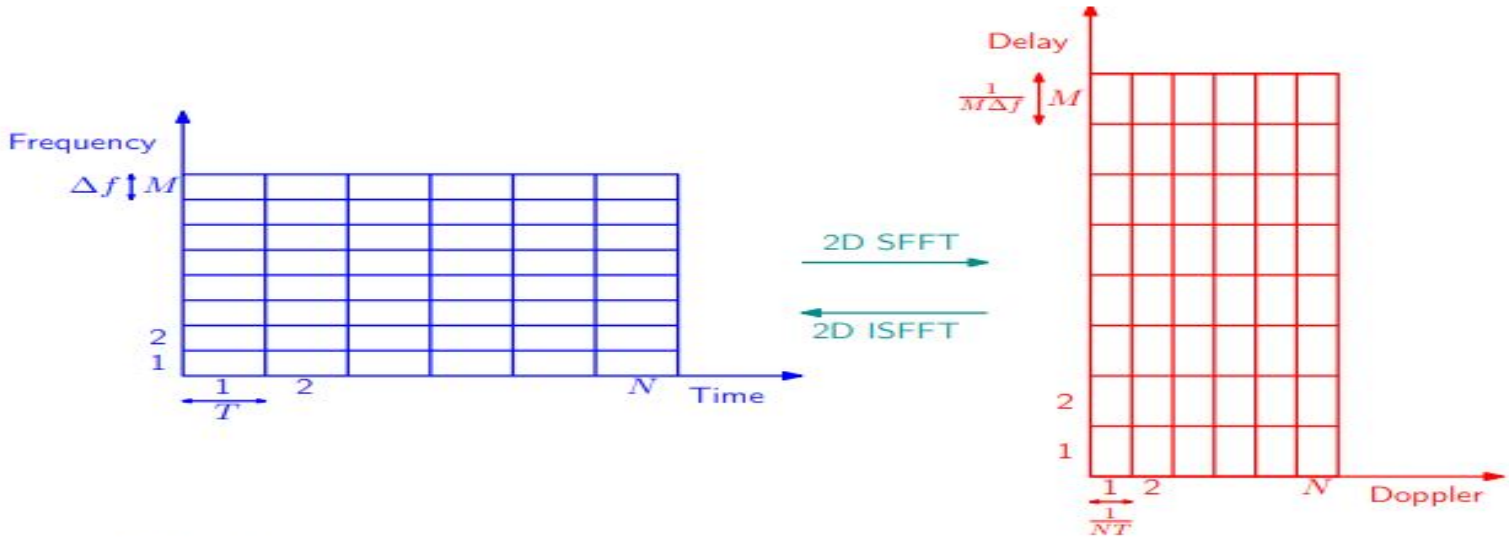
TF and DD grids

- TF grid, Λ: TF plane is sampled at intervals T and Δf, to obtain a 2D grid

$$\Lambda = \{(nT, m\Delta f), n = 0, \cdots, N-1, m = 0, \cdots, M-1\}$$

- DD grid Γ: reciprocal to Λ

$$\Gamma = \{(\tfrac{k}{NT}, \tfrac{l}{M\Delta f}), k = 0, \cdots, N-1, l = 0, \cdots, M-1\}$$



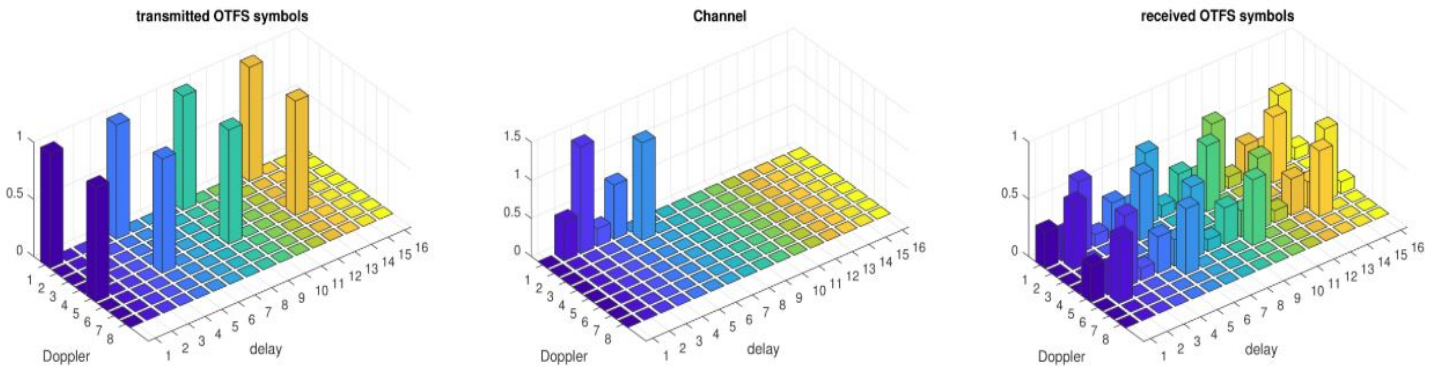
- In OTFS, information symbols are multiplexed in the DD grid (with Doppler resolution $\frac{1}{NT}$ and delay resolution $\frac{1}{M\Delta f}$)
- MN symbols are sent over a time duration of NT secs occupying a BW of $M\Delta f$

Input-output relation

- Received signal in DD domain:
  - for $\tau_i \triangleq \frac{\alpha_i}{M\Delta f}$ and $\nu_i \triangleq \frac{\beta_i}{NT}$, α_i and β_i are integers

$$y[k,l] = \sum_{i=1}^{P} h_i' \, x[(k-\beta_i)_N, (l-\alpha_i)_M] + v[k,l]$$

where $h_i' = h_i e^{-j2\pi\nu_i\tau_i}$, $h_i \sim \mathcal{CN}(0, 1/P)$.



- Let $\widehat{\mathbf{H}}$ denote the $N \times M$ channel matrix in the DD domain
- Let $\hat{h}(k,l)$ denote the (k,l)th element of $\widehat{\mathbf{H}}$, which is given by

$$\hat{h}(k,l) = \begin{cases} h_i' & \text{if } k = \beta_i \ \& \ l = \alpha_i \text{ for some } i \in \{1, 2, \cdots, P\} \\ 0 & \text{otherwise.} \end{cases}$$

Vectorized input-output relation

- The input-output relation can be vectorized as[1]

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{v},$$

where $x_{k+Nl} = x[k,l]$, $y_{k+Nl} = y[k,l]$, $v_{k+Nl} = v[k,l]$,
$k = 0, \cdots, N-1$, $l = 0, \cdots, M-1$

$\mathbf{H} \in \mathbb{C}^{MN \times MN}$: jth row ($j = k + Nl$) of $\mathbf{H}$, denoted by $\mathbf{H}[j]$, is given by

$$\mathbf{H}[j] = [\hat{h}((k-0)_N, (l-0)_M) \;\; \hat{h}((k-1)_N, (l-0)_M) \;\; \cdots \hat{h}((k-N-1)_N, (l-M-1)_M)].$$

- $\mathbf{H}$ is a block circulant matrix with circulant blocks, with each row having P non-zero elements
  - this can be exploited to devise low-complexity linear (MMSE) receivers[2,3]

[1] P. Raviteja, K. T. Phan, and E. Viterbo, "Interference cancellation and iterative detection for orthogonal time frequency space modulation," *IEEE Trans. Wireless Commun.*, vol. 17, no. 10, pp. 6501-6515, Oct. 2018.

[2] S. Tiwari, S. S. Das, and V. Rangamgari, "Low complexity MMSE receiver for OTFS," *IEEE Commun. Lett.*, vol. 23, no. 12, pp. 2205–2209, Dec. 2019.

[3] G.D. Surabhi andA. Chockalingam, "Low-complexity linear equalization for OTFS modulation," *IEEE Commun. Lett.*, vol. 24, no. 2, pp. 330–334, Feb. 2020.

Deep learning in communications

- Deep learning (DL) has been increasingly studied in wireless communications for designing intelligent communication systems[1,2]
- In the PHY layer, DL has been applied in two important ways
  - as a replacement to the existing communication blocks like channel coding[3], signal detection[4], channel estimation
  - for designing end-to-end communication systems without traditional communication blocks[2]
- Both approaches are found to be promising

---

[1] C. Jiang, H. Zhang, Y. Ren, Z. Han, K. C. Chen, and L. Hanzo, "Machine learning paradigms for next-generation wireless networks," *IEEE Wireless Commun.*, vol. 24, no. 2, pp. 98-105, Apr. 2017.

[2] T. O'shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. Cognitive Commun. and Netw.*, vol. 3, no. 4, pp. 563-575, Dec. 2017.

[3] H. Kim, Y. Jiang, R. Rana, S. Kannan, and P. Viswanath, "Communication algorithms via deep learning," *Proc. ICLR'2018*, pp. 1-17, Apr. 2018.

[4] N. Farsad and A. Goldsmith, "Neural network detection of data sequences in communication systems," *IEEE Trans. Signal Process.*, vol. 66, no. 21, pp. 5663- 5678, Sep. 2018.

Why deep learning?

- Imperfections in/deviations from assumed models
- Deep learning has shown robustness to such imperfections/deviations
- Large amount of available data for training
- Training takes time, but once trained the weights can be stored
- Evolution of hardware to shorten training times

Deep neural networks

DNNs

- Neuron - A basic cell.
- $u_1, u_2, \ldots, u_n$ are the inputs to the neuron.
- $w_1, w_2, \ldots, w_n$ are the weights of the branches.
- b is called the bias.

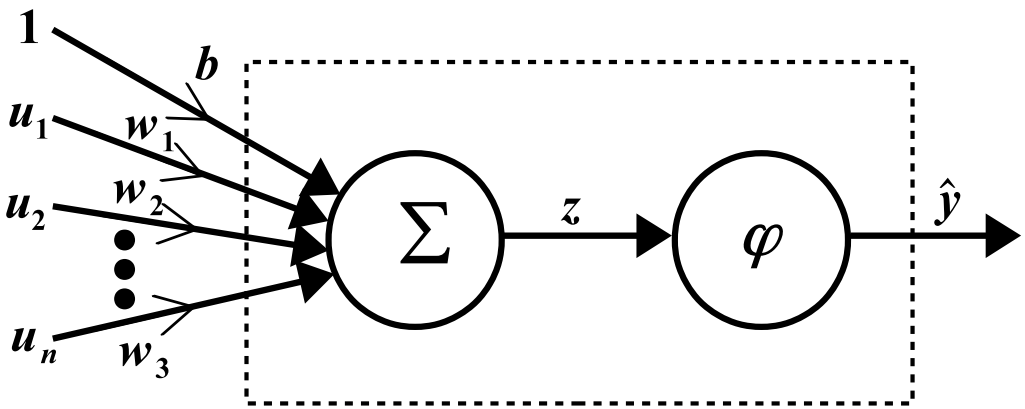


Figure: A single neuron

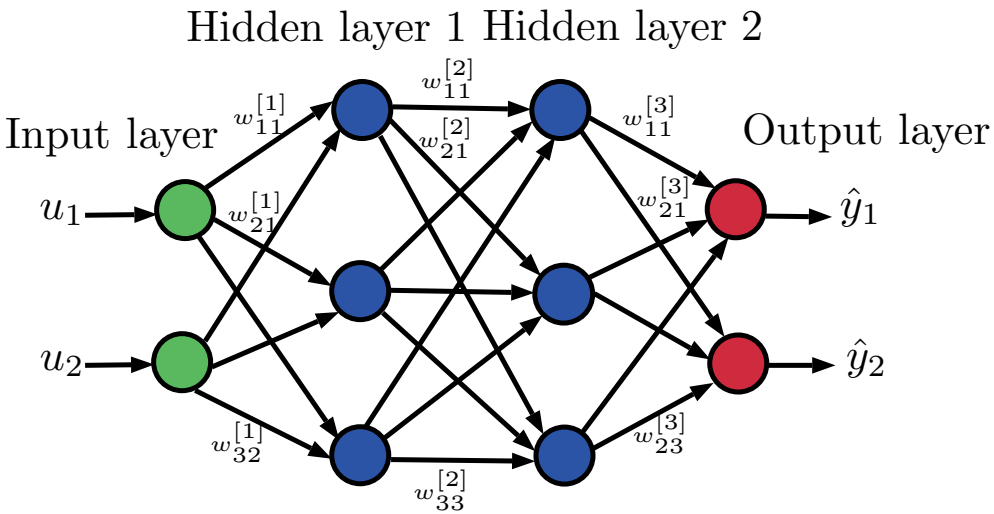
- z and $\hat{y}$ are computed as follows

$$z = \sum_{i=1}^{n} w_i u_i + b, \quad \text{and} \quad \hat{y} = \varphi(z)$$

- $\varphi(\cdot)$ is called the activation function. Examples are sigmoid, tanh, softmax.

Deep learning

- The goal is to update weights and bias in such a way that $\hat{y}$ is as close to the required y as possible, $y = f(u_1, \ldots, u_n) + c$.
- This is achieved in two steps.
  - Forward pass (finding $\hat{y}$ from u)
  - Back propagation (finding the weights from $\hat{y}$)
- Weights are calculated by minimizing loss function, $L(y, \hat{y})$. E.g., MSE, BCE
- In a typical neural network, many neurons are placed together to form a layer and such layers are interconnected.
- A neural network with three or more layers is called a deep neural network.



NN architectures

- Fully connected NN (simply called a deep neural network (DNN))
  - Each neuron in the current layer is connected to all the neurons in the previous layer
- Convolutional NN (CNN)
  - involves sparse connections and results in reduced complexity
  - commonly used for learning involving images
- Recurrent NN (RNN)
  - A special architecture used for learning which involves time series or sequence data

DL programming frameworks

- TensorFlow
  - developed by the Google Brain team
  - widely used deep learning framework
- Keras:
  - written in python and works on top of tensorflow
  - developed with a focus on quick experimentation
  - Beginner friendly
    - designed to minimize user actions and makes it easy to understand models
- PyTorch:
  - developed by Facebook
  - strong competitor for TensorFlow
- and many more...

OTFS signal detection using DNNs

OTFS signal detection using DNNs

- System model: $\mathbf{y}=\mathbf{H}\mathbf{x}+\mathbf{v}$
- Detection problem: Given $\mathbf{y}$ and $\mathbf{H}$, obtain an estimate of $\mathbf{x}$

OTFS signal detection using DNNs

- System model: $\mathbf{y}=\mathbf{H}\mathbf{x}+\mathbf{v}$
- Detection problem: Given $\mathbf{y}$ and $\mathbf{H}$, obtain an estimate of $\mathbf{x}$
- Two detection approaches using DNNs

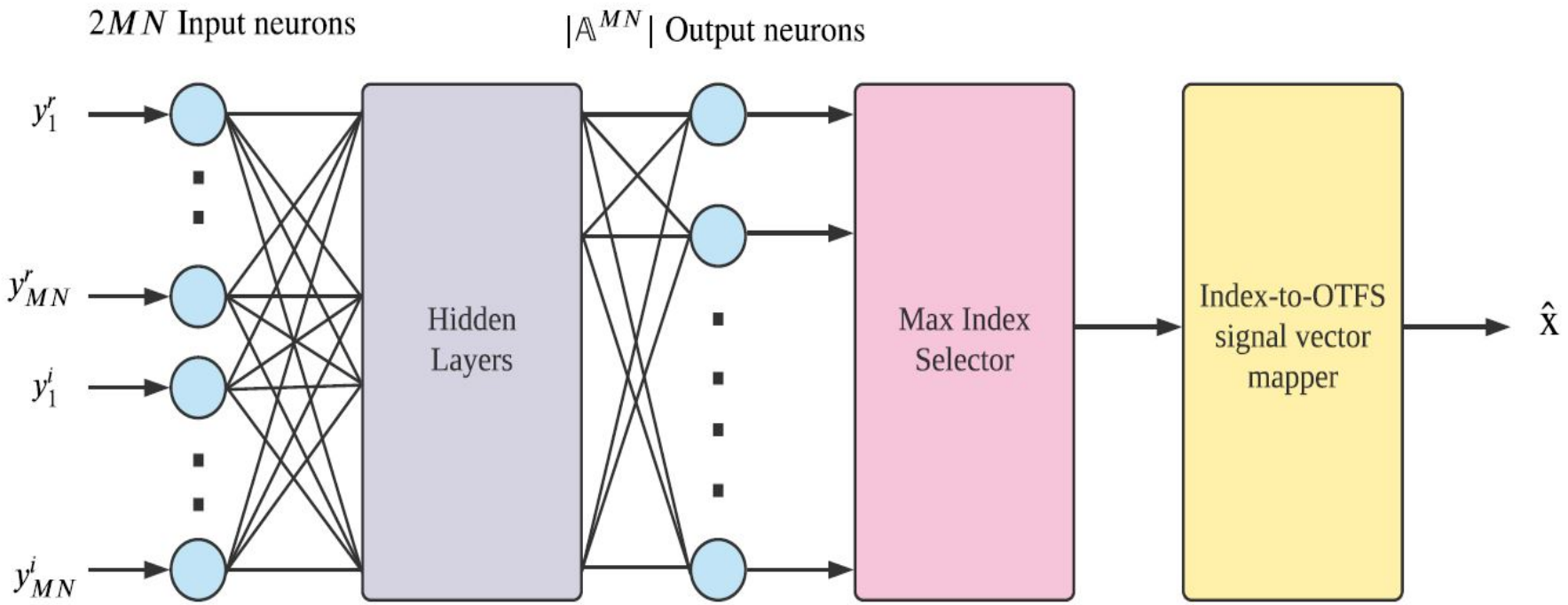
OTFS signal detection using DNNs

- System model: $\mathbf{y}=\mathbf{H}\mathbf{x}+\mathbf{v}$
- Detection problem: Given $\mathbf{y}$ and $\mathbf{H}$, obtain an estimate of $\mathbf{x}$
- Two detection approaches using DNNs
- Approach I (Full-DNN approach)
  - Use a single large DNN at the OTFS signal vector level
  - Input layer: $2MN$ neurons (real and imaginary parts of $\mathbf{y}$ vector)
  - Several large hidden layers
  - Output layer: $|\mathbb{A}^{MN}|$ neurons (one for each possible $\mathbf{x}$ vector)
  - High complexity (# o/p neurons increases exponentially in size of vector $\mathbf{x}$)

OTFS signal detection using DNNs

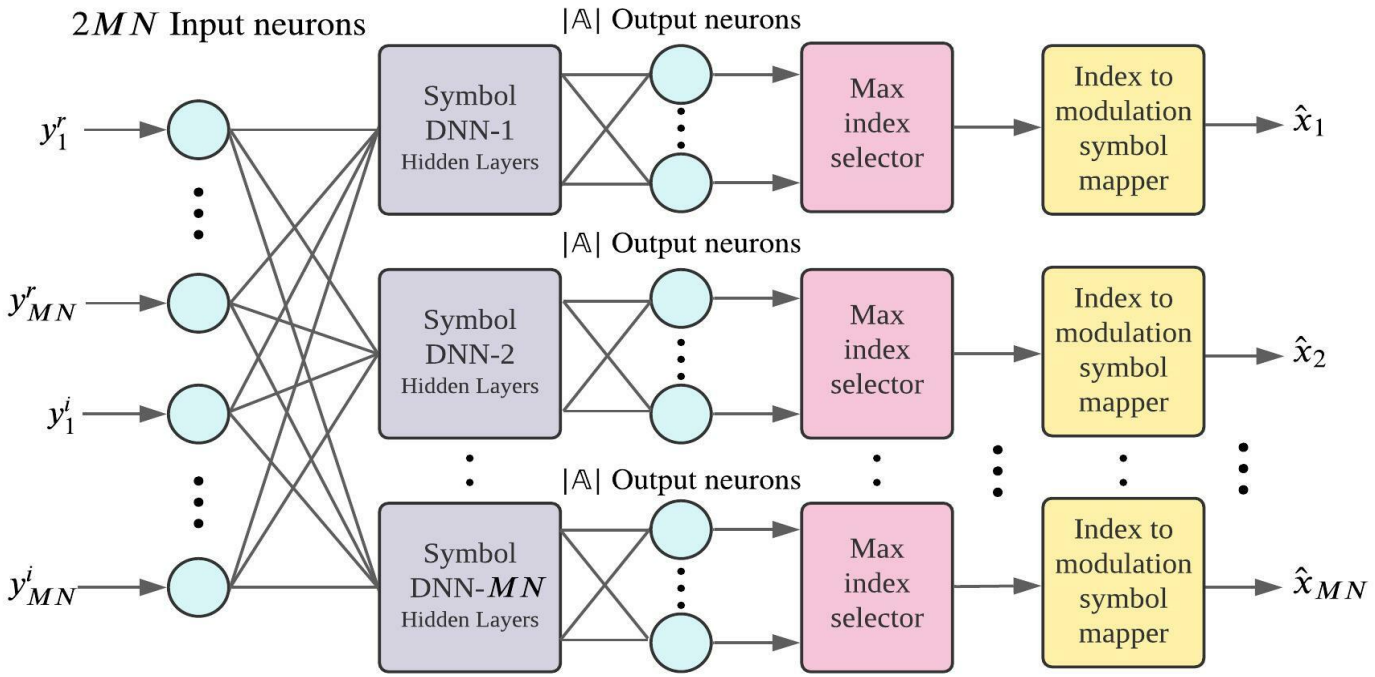
- System model: $\mathbf{y}=\mathbf{H}\mathbf{x}+\mathbf{v}$
- Detection problem: Given $\mathbf{y}$ and $\mathbf{H}$, obtain an estimate of $\mathbf{x}$
- Two detection approaches using DNNs
- Approach I (Full-DNN approach)
  - Use a single large DNN at the OTFS signal vector level
  - Input layer: $2MN$ neurons (real and imaginary parts of $\mathbf{y}$ vector)
  - Several large hidden layers
  - Output layer: $|\mathbb{A}^{MN}|$ neurons (one for each possible $\mathbf{x}$ vector)
  - High complexity (# o/p neurons increases exponentially in size of vector $\mathbf{x}$)
- Approach II (Symbol-DNN approach)
  - Use multiple small DNNs at the modulation symbol level
    - One small DNN for each coordinate in the $\mathbf{x}$ vector (# small DNNs: MN)
  - Common input layer: $2MN$ neurons (real amd imag. parts of $\mathbf{y}$ vector)
  - Relatively fewer/smaller hidden layers
  - Output layer of each DNN: $|\mathbb{A}|$ neurons (one for each possible symbol in $\mathbb{A}$)
  - Reduced complexity (total # o/p neurons: $MN|\mathbb{A}|$)

OTFS signal detection using DNNs (Approach I)



- Each neuron in output layer corresponds to one OTFS signal vector
- If ith signal vector ($i = 1, \cdots, |\mathbb{A}^{MN}|$) of the OTFS signal set is sent, then
  - ith o/p neuron is likely to result in high probability value
  - all other o/p neurons j, $j \neq i$, result in low probability values
- This architecture requires learning a large number of parameters

OTFS signal detection using DNN (Approach II)



- kth output neuron ($k = 1, \cdots, |\mathbb{A}|$) of lth Symbol-DNN ($l = 1, \cdots, MN$) gives the probability of kth symbol being transmitted from lth DD bin
- Requires learning much fewer number of parameters and hence scales well

[2] Ashwitha Naikoti Shamasundar and A. Chockalingam, "Low-complexity Delay-Doppler Symbol DNN for OTFS Signal Detection," *IEEE VTC2021-Spring*, Helsinki, April 2021.

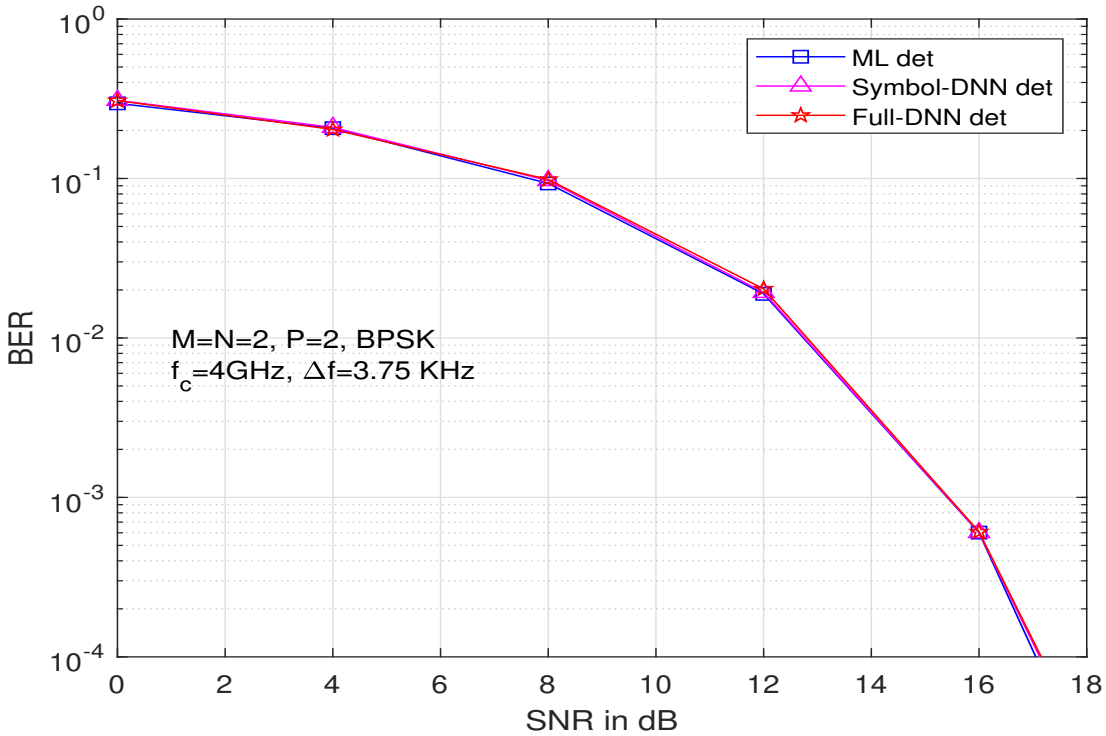
DNN detection performance (standard noise model)

- Standard noise model: i.i.d Gaussian noise
- OTFS system parameters
  - $f_c = 4$ GHz, $\Delta f = 3.75$ KHz, $M = N = 2$, $P = 2$, BPSK
- Channel: an instance of Rayleigh fading channel
- DNN parameters

Parameters	Symbol-DNN	Full-DNN
No. of input neurons	$2MN = 8$	$2MN = 8$
No. of output neurons	$\lvert\mathbb{A}\rvert = 2$	$2^{MN} = 16$
No. of hidden layers	1	1
Hidden layer activation	ReLU	ReLU
Output layer activation	Softmax	Softmax
Optimization	Adam	Adam
Loss function	Binary crossentropy	Categorical crossentropy
Training SNR	10 dB	10 dB
No. of training examples	30,000	30,000
No. of epochs	50	50

- Full-DNN: input $\rightarrow$ 8 $\rightarrow$ ReLU $\rightarrow$ 12 $\rightarrow$ ReLU $\rightarrow$ 16 $\rightarrow$ Softmax.
- Symbol-DNN: input $\rightarrow$ 8 $\rightarrow$ ReLU $\rightarrow$ 4 $\rightarrow$ ReLU $\rightarrow$ 2 $\rightarrow$ Softmax.

DNN detection performance (standard noise model)

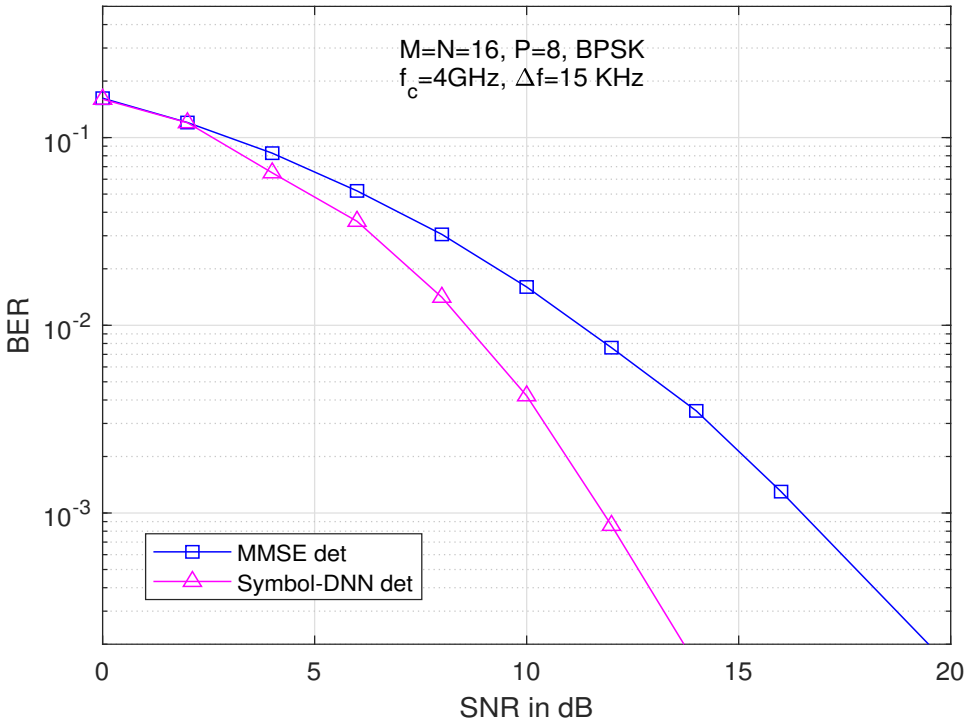


- Symbol-DNN achieves similar performance as that of Full-DNN
- Both DNNs achieve almost ML detection performance
- Complexity comparison (in no. of real operations):

Detector	ML det.	Symbol-DNN	Full-DNN
Complexity	1088	304	564
Trainable parameters	-	184	316

DNN detection performance (standard noise model)

- OTFS system parameters
  - $f_c = 4$ GHz, $\Delta f = 15$ KHz, $M = N = 16$, $P = 8$, BPSK

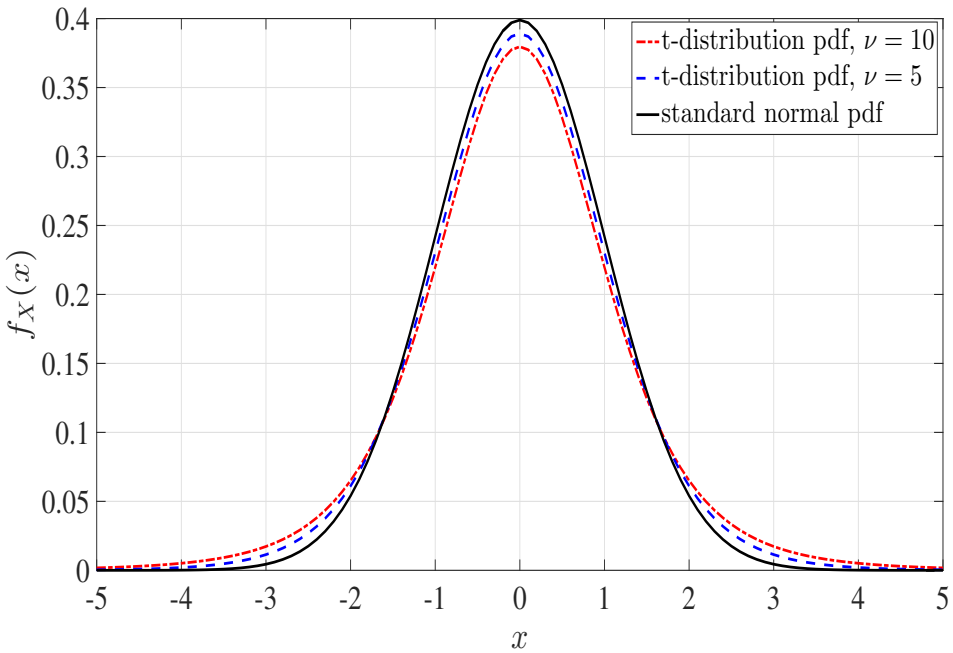


- Full-DNN: No. of output neurons $= 2^{256}$ (infeasible)
- No. of Symbol-DNNs: $MN = 256$
- Symbol-DNN: input $\rightarrow$ 512 $\rightarrow$ ReLU $\rightarrow$ 256 $\rightarrow$ ReLU $\rightarrow$ 2 $\rightarrow$ Softmax
- Trained for 20 epochs with 80,000 training samples at SNR=8 dB

Detector	Complexity	Trainable parameters
MMSE	83951616	-
Symbol-DNN	67305472	33751552

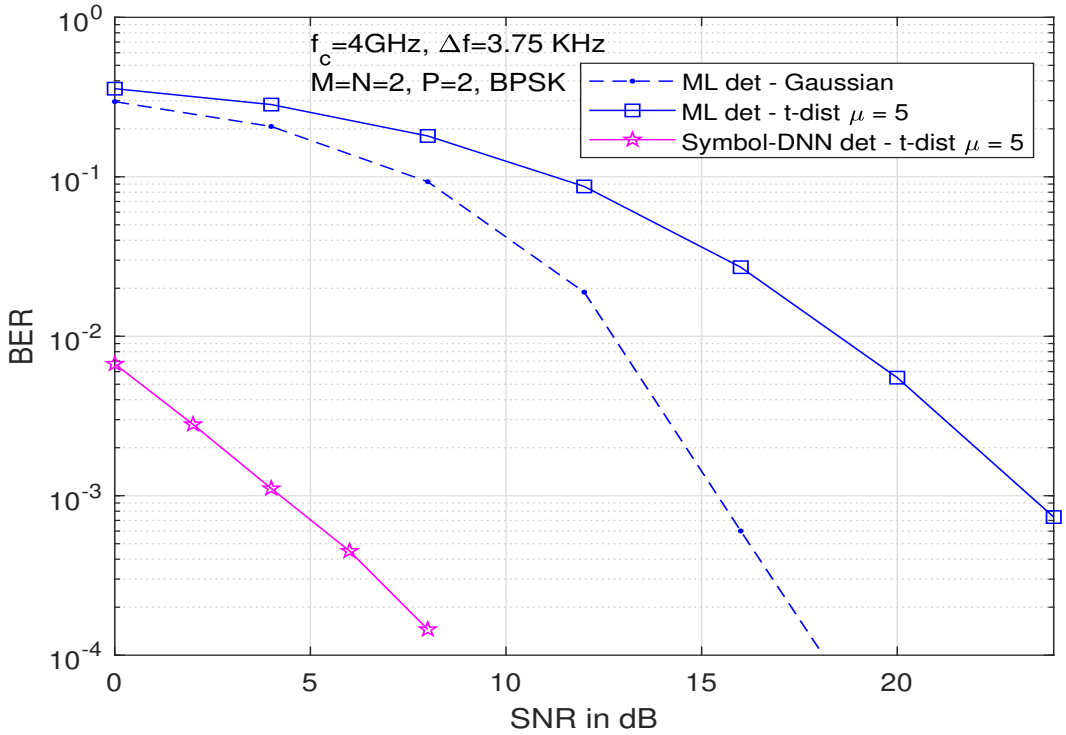
Model mismatch in noise

- Deviations from standard noise model
  - Non-Gaussian noise
  - Correlated noise
- t-distributed noise
  - t-distribution looks very similar to Gaussian
  - parameterized by ν
  - deviates more from Gaussian for smaller values of ν



DNN detection performance (noise model mismatch)

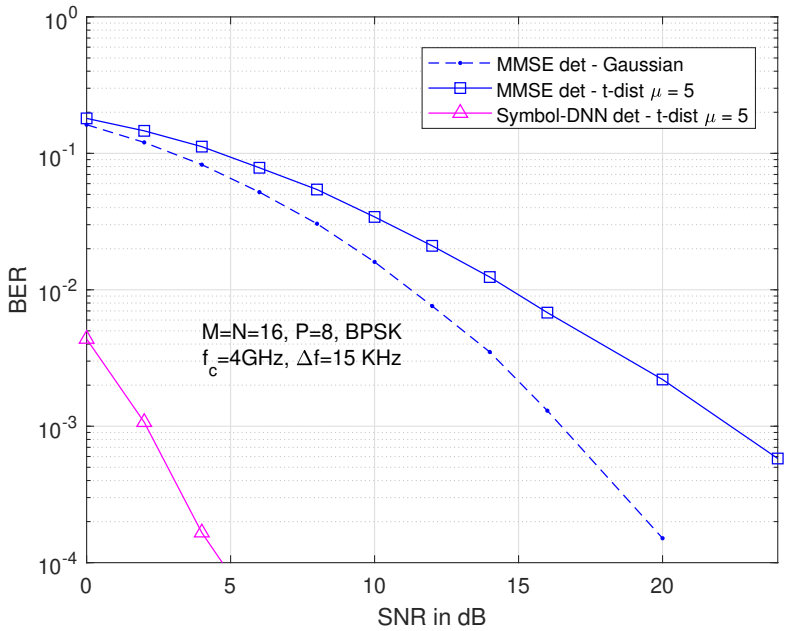
- OTFS system parameters
  - $f_c = 4$ GHz, $\Delta f = 3.75$ KHz, $M = N = 2$, $P = 2$, BPSK
- Performance with t-distributed noise



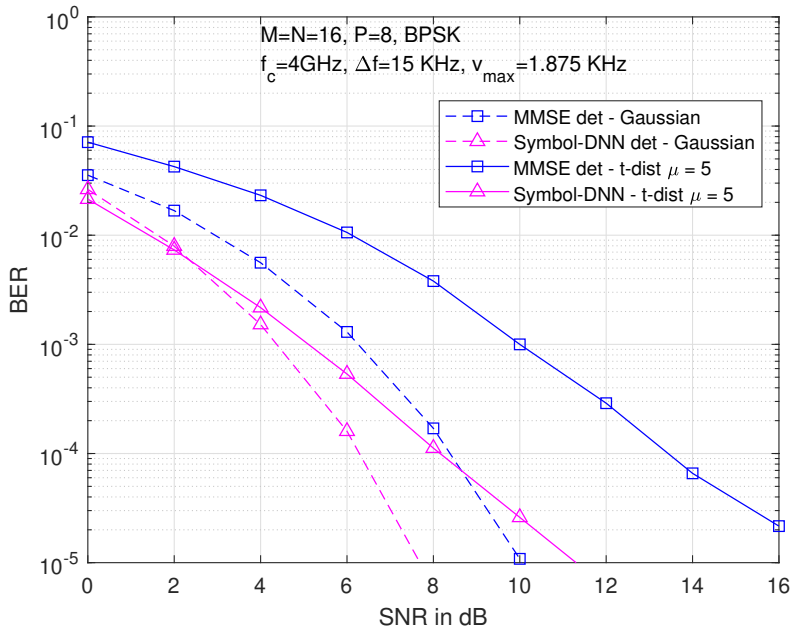
- ML detection is not optimal in non-Gaussian noise
- DNN detector learns the noise model and performs better than ML detector

DNN detection performance (noise model mismatch)

- OTFS system parameters
  - $f_c = 4$ GHz, $\Delta f = 15$ KHz, $M = N = 16$, $P = 8$, BPSK
- Performance with t-distributed noise



(a) Static channel



(b) Max. Doppler=1.875 KHz

Path, i	1	2	3	4	5	6	7	8
τ_i (μs)	0	4.16	8.32	12.48	16.64	20.8	24.96	29.12
ν_i (Hz)	0	0	938.5	938.5	938.5	1875	1875	1875

- DNN detector performs better in non-Gaussian noise

Correlated noise in MIMO-OTFS

- Noise correlation across multiple receive antennas due to insufficient spacing[1,2]
- Model that characterizes this correlation is receiver hardware dependent
- DNNs can learn the underlying noise model specific to the receiver hardware
- An example noise correlation matrix $\mathbf{N}_c$: $\mathbf{n}_c = \mathbf{N}_c\mathbf{n}$

$$\mathbf{N}_c = \begin{bmatrix} 1 & \rho & \rho^2 & \cdots & \rho^{n_r-1} \\ \rho & 1 & \rho & \cdots & \rho^{n_r-2} \\ & & & \ddots & \\ \rho^{n_r-1} & \rho^{n_r-2} & \cdots & & 1 \end{bmatrix}$$

and ρ is the correlation coefficient such that $0 \leq \rho \leq 1$

- Modified ML detector for the case of correlated noise

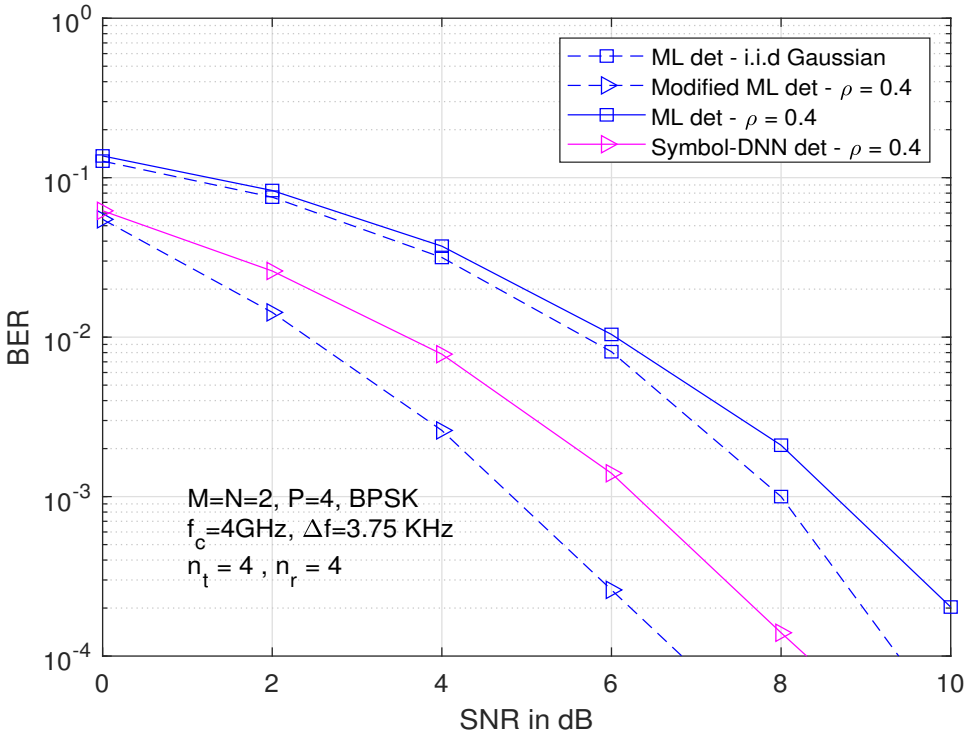
$$\hat{\mathbf{x}} = \underset{\mathbf{x}\in\mathbb{A}^{MN}}{\arg\min}\ (\mathbf{y} - \mathbf{H}\mathbf{x})^H \Sigma^{-1} (\mathbf{y} - \mathbf{H}\mathbf{x})$$

[1] S. Krusevac, P. Rapajic, and R. A. Kennedy, "Channel capacity estimation for MIMO systems with correlated noise," *Proc. IEEE GLOBECOM'05*, pp. 2812-2816, Dec. 2005.

[2] C. P. Domizioli, B. L. Hughes, K. G. Gard, and G. Lazzi, "Receive diversity revisited: correlation, coupling and noise," *Proc. IEEE GLOBECOM'2007*, pp. 3601-3606, Dec. 2007.

DNN detection performance (in noise model mismatch)

- MIMO-OTFS system parameters
  - 4×4 MIMO, $f_c = 4$ GHz, $\Delta f = 3.75$ KHz, $M = N = 2$, $P = 4$, BPSK
- Performance in correlated noise



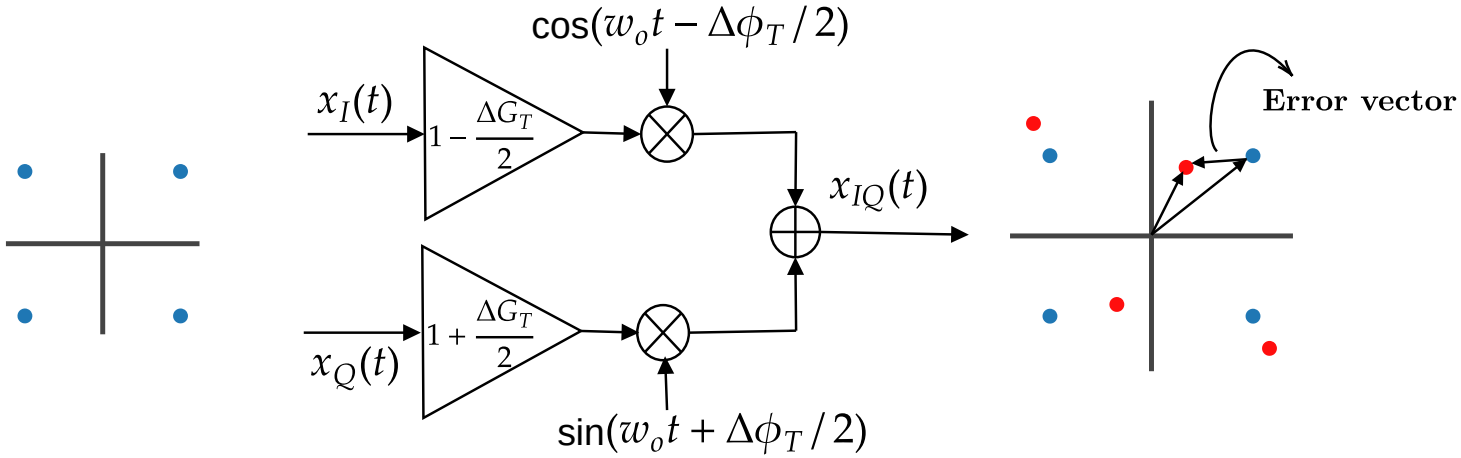
- Symbol-DNN: input $\rightarrow$ 32 $\rightarrow$ ReLU $\rightarrow$ 8 $\rightarrow$ ReLU $\rightarrow$ 16 $\rightarrow$ ReLU $\rightarrow$ 2 $\rightarrow$ Softmax
- Trained for 100 epochs with 80000 training examples at SNR $= 4$ dB

- DNN detection performs better in correlated noise

IQI compensation using DNNs

IQ imbalance model - Tx IQI

- Transmitter IQI



- Transmit signal $\mathbf{x}_{IQ}$ in the presence of Tx IQI can be modelled as

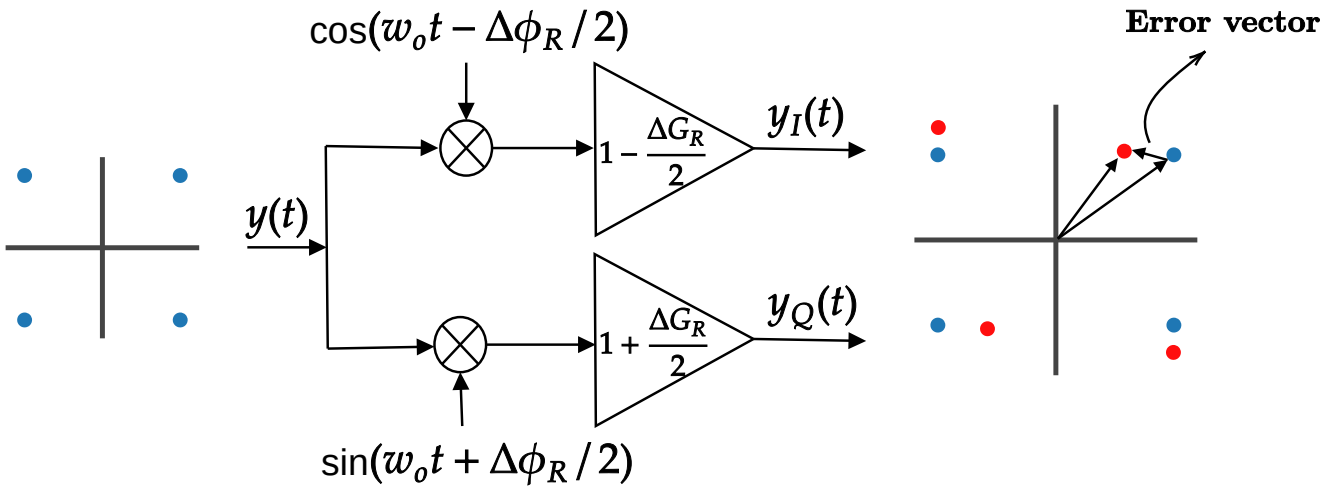
$$\mathbf{x}_{IQ} = \alpha_T \mathbf{x} + \underbrace{\beta_T \mathbf{x}^*}_{\text{image}}, \qquad \mathbf{y} = \mathbf{H}\mathbf{x}_{IQ} + \mathbf{v}$$

$$\alpha_T = \frac{1}{2}\left[\cos\left(\frac{\Delta\phi_T}{2}\right) + j\frac{\Delta G_T}{2}\sin\left(\frac{\Delta\phi_T}{2}\right)\right], \quad \beta_T = \frac{1}{2}\left[-\frac{\Delta G_T}{2}\cos\left(\frac{\Delta\phi_T}{2}\right) - j\sin\left(\frac{\Delta\phi_T}{2}\right)\right]$$

- Tx IQI generates an image signal (causes self interference in zero-IF receivers)
- Image suppression ratio (ISR) $= \frac{|\beta_T|^2}{|\alpha_T|^2} \approx \frac{\Delta G_T^2}{4} + \frac{\Delta\phi_T^2}{4}$
- ISR $=$ EVM^2; $\quad$ SNR$_{\text{IQI}} = \frac{1}{\text{ISR}}$; $\quad$ SNR degradation due to Tx IQI $\propto$ ISR

IQ imbalance model - Rx IQI

- Receiver IQI



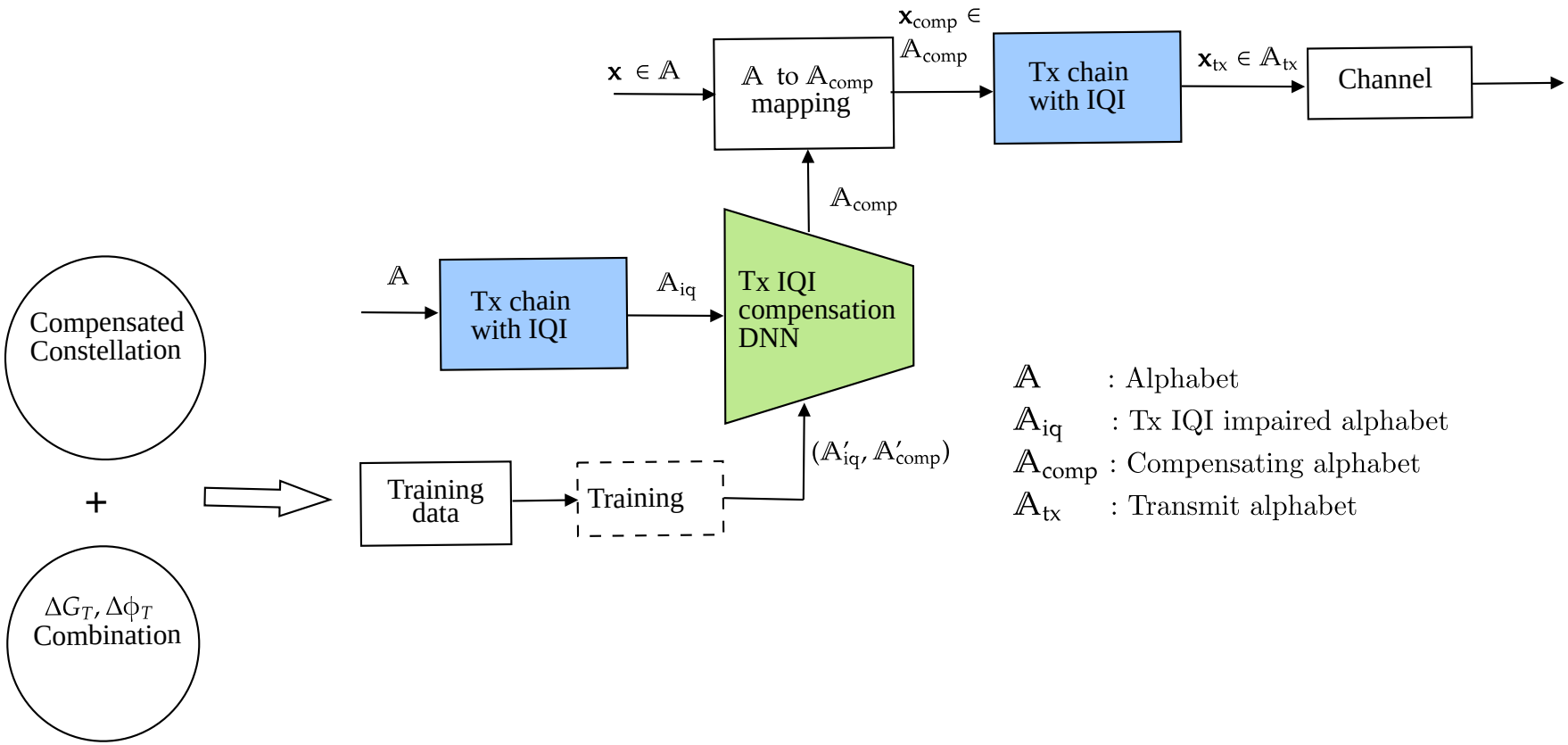
- Received signal $\mathbf{y}_{IQ}$ in the presence of Rx IQI can be modelled as

$$\mathbf{y}_{IQ} = \alpha_R \mathbf{y} + \beta_R \mathbf{y}^*$$

$$\alpha_R = \frac{1}{2}\left[\cos\left(\frac{\Delta\phi_R}{2}\right) + j\frac{\Delta G_R}{2}\sin\left(\frac{\Delta\phi_R}{2}\right)\right], \quad \beta_R = \frac{1}{2}\left[-\frac{\Delta G_R}{2}\cos\left(\frac{\Delta\phi_R}{2}\right) - j\sin\left(\frac{\Delta\phi_R}{2}\right)\right]$$

- ISR $= \frac{|\beta_R|^2}{|\alpha_R|^2} \approx \frac{\Delta G_R^2}{4} + \frac{\Delta\phi_R^2}{4}$
- Image signal causes SNR degradation

Tx IQI compensation: DNN-1



- A fully-connected DNN with $2|\mathbb{A}|$ input neurons and $2|\mathbb{A}|$ output neurons
- Training data is obtained using the compensation model, given by

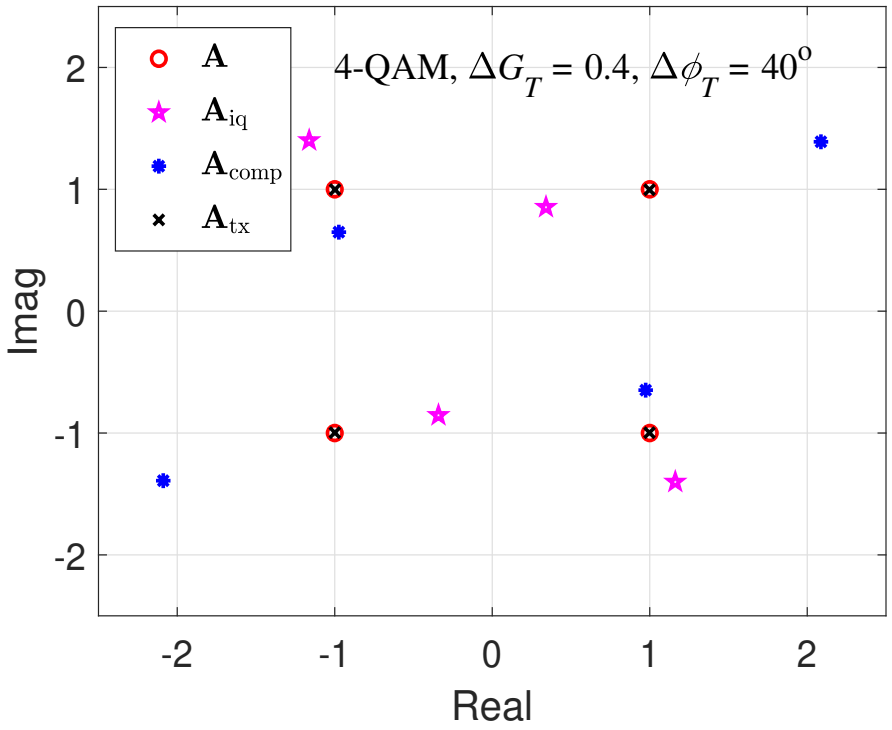
$$\begin{bmatrix} \mathbb{A}'_{\text{comp}} \\ \mathbb{A}'^{*}_{\text{comp}} \end{bmatrix} = \begin{bmatrix} \alpha_T & \beta_T \\ \beta_T^* & \alpha_T^* \end{bmatrix}^{-1} \begin{bmatrix} \mathbb{A} \\ \mathbb{A}^* \end{bmatrix}.$$

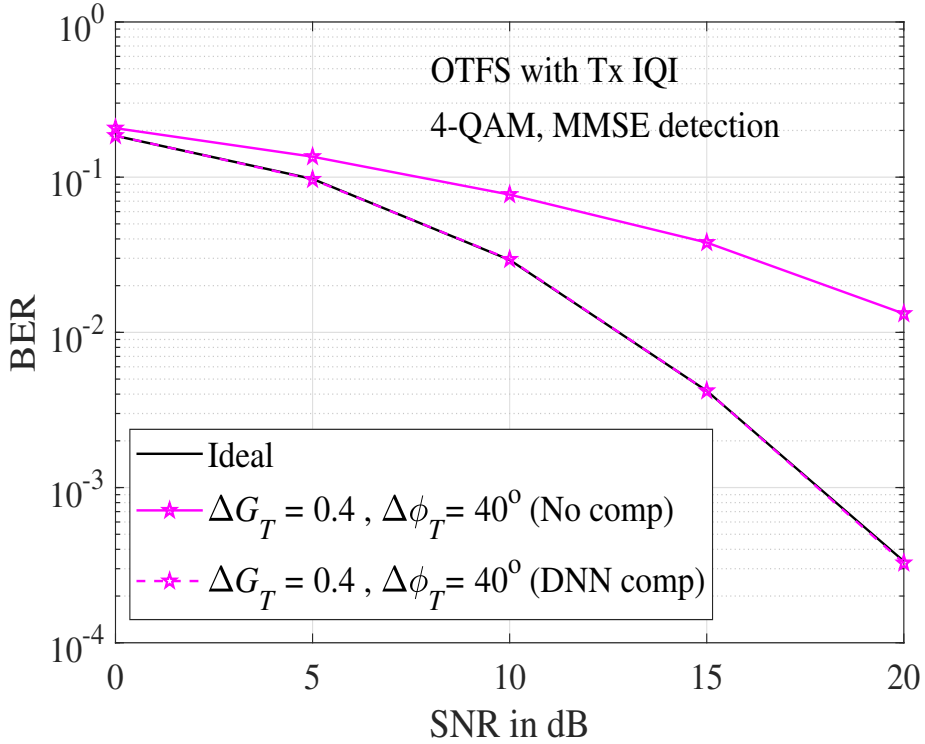
Paramaters of Tx IQI compensation DNN-1

- Parameters of Tx IQI compensation DNN-1

Parameters	BPSK-DNN	4QAM-DNN	16QAM-DNN
No. of input neurons	$2\|\mathbb{A}\| = 4$	$2\|\mathbb{A}\| = 8$	$2\|\mathbb{A}\| = 32$
No. of output neurons	$2\|\mathbb{A}\| = 4$	$2\|\mathbb{A}\| = 8$	$2\|\mathbb{A}\| = 32$
No. of hidden layers	4	3	3
Hidden layer activation	Tanh	Tanh	Tanh
Output layer activation	Linear	Linear	Linear
Optimization	Adam	Adam	Adam
Loss function	MSE	MSE	MSE
No. of training examples	1000	1000	1000
No. of epochs	5000	5000	5000
Batch size	5	5	5

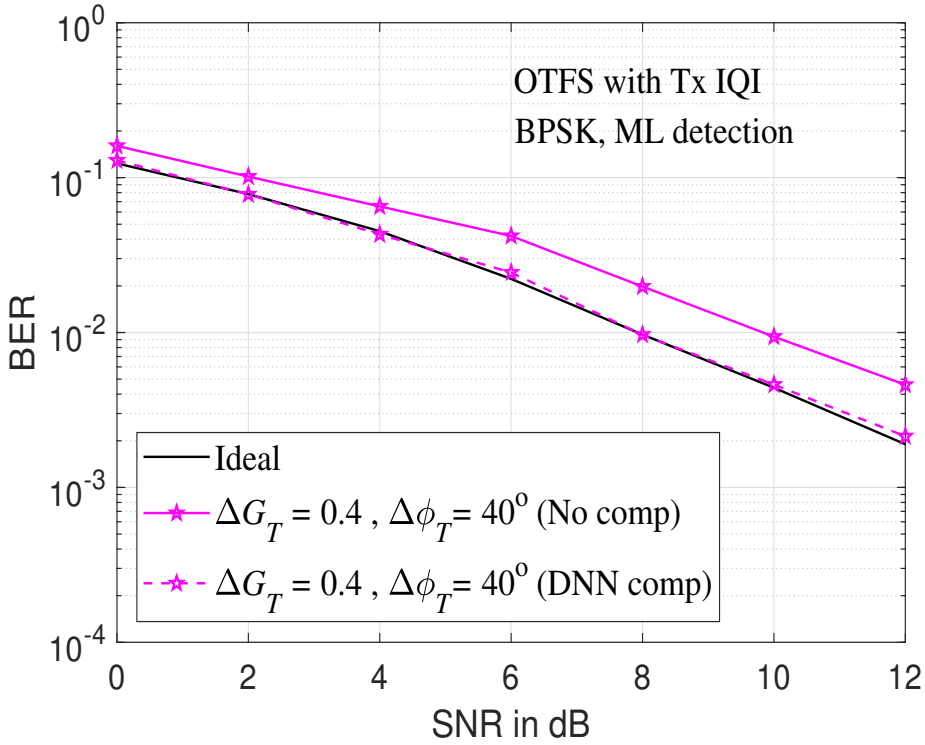
Tx IQI compensation using DNN-1: 4-QAM



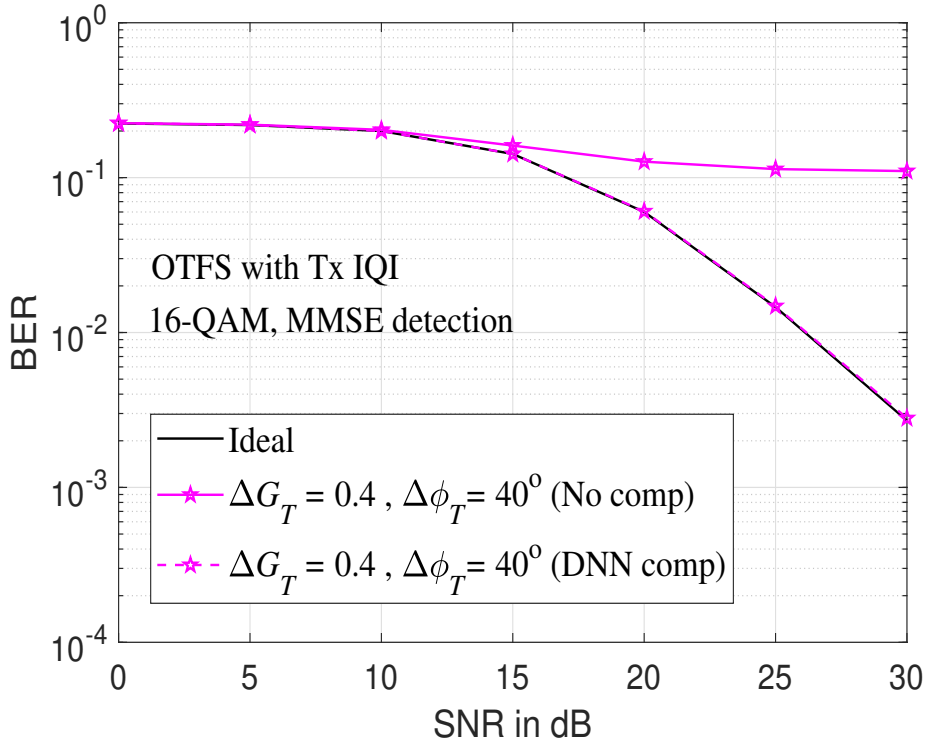


- **DNN-1 for 4-QAM**
  - *Input* $\rightarrow$ *8* $\rightarrow$ *Tanh* $\rightarrow$ *64* $\rightarrow$ *Tanh* $\rightarrow$ *32* $\rightarrow$ *Tanh* $\rightarrow$ *16* $\rightarrow$ *Tanh* $\rightarrow$ *8* $\rightarrow$ *Linear*
- DNN effectively compensates the Tx IQI

Tx IQI compensation using DNN-1: BPSK, 16-QAM



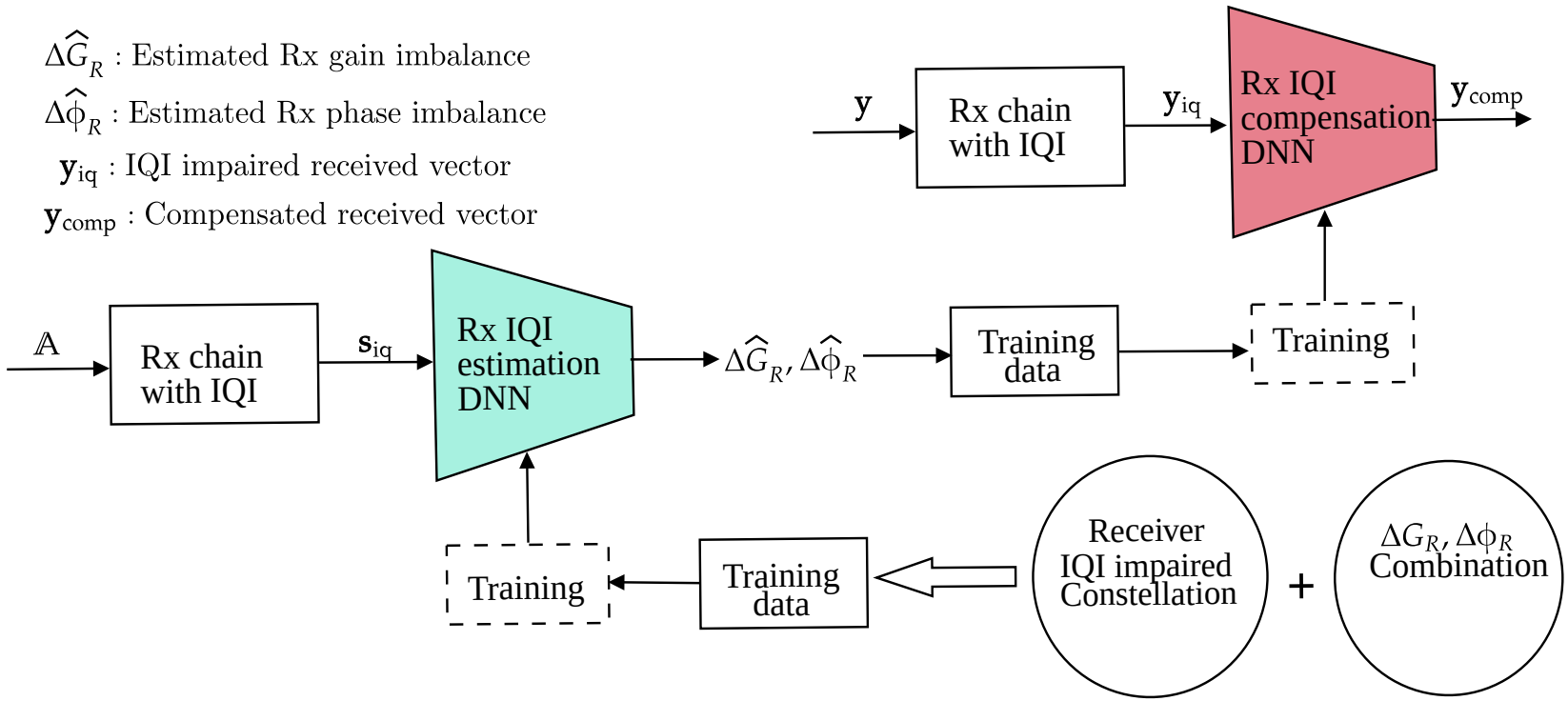
((c)) BPSK



((d)) 16-QAM

- **DNN-1 for BPSK**
  - *Input* $\rightarrow$ *4* $\rightarrow$ *Tanh* $\rightarrow$ *64* $\rightarrow$ *Tanh* $\rightarrow$ *32* $\rightarrow$ *Tanh* $\rightarrow$ *16* $\rightarrow$ *Tanh* $\rightarrow$ *8* $\rightarrow$ *Tanh* $\rightarrow$ *4* $\rightarrow$ *Linear*
- **DNN-1 for 16-QAM**
  - *Input* $\rightarrow$ *32* $\rightarrow$ *Tanh* $\rightarrow$ *256* $\rightarrow$ *Tanh* $\rightarrow$ *128* $\rightarrow$ *Tanh* $\rightarrow$ *64* $\rightarrow$ *Tanh* $\rightarrow$ *32* $\rightarrow$ *Linear*

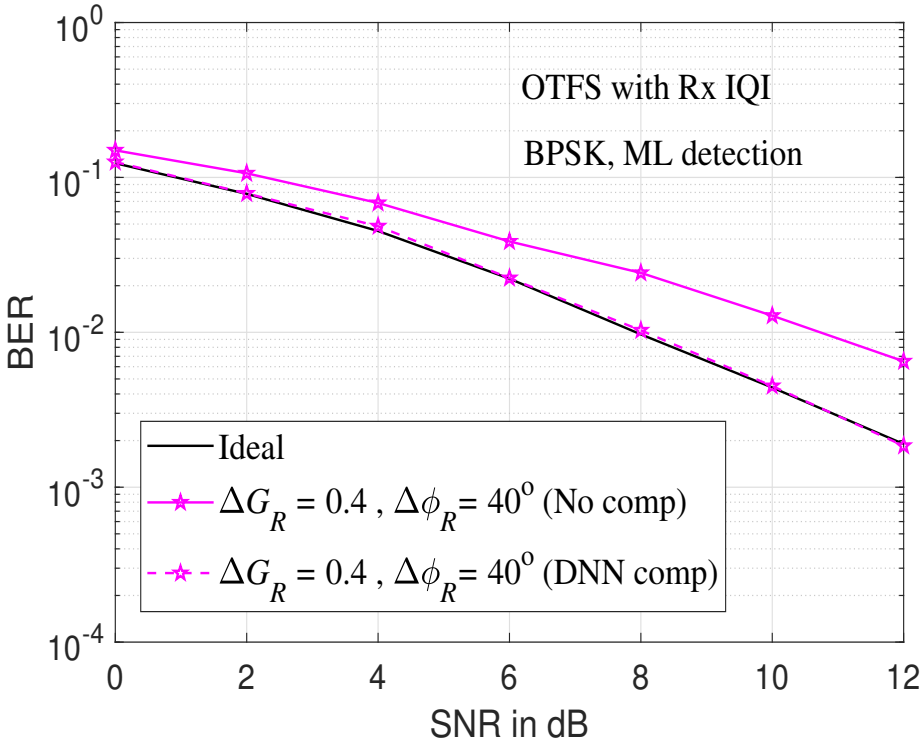
Rx IQI Estimation & compensation: DNN-2, DNN-3



- Estimation DNN-2: $2|\mathbb{A}|$ input neurons and 2 output neurons
- Compensation DNN-3: $2MN$ input neurons and $2MN$ output neurons
- IQI impaired vector at the receiver

$$\mathbf{y}_{IQ} = \alpha_R \mathbf{y} + \beta_R \mathbf{y}^*.$$

Rx IQI compensation: BPSK



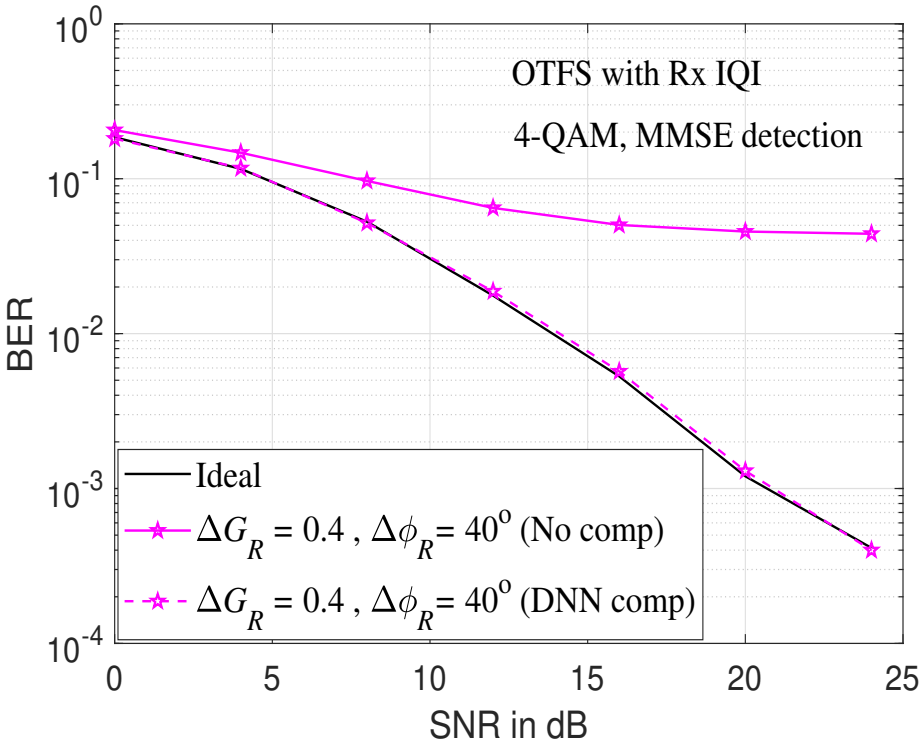
Parameters	DNN-2	DNN-3
No. of input neurons	$2\lvert\mathbb{A}\rvert$	$2MN = 32$
No. of output neurons	2	$2MN = 32$
No. of hidden layers	1	1
Hidden layer activation	Tanh	Tanh
Output layer activation	Linear	Linear
Optimization	Adam	Adam
Loss function	MSE	MSE
No. of training examples	1000	50000
No. of epochs	500	500
Batch size	5	50

Table: Parameters of DNN-2 and DNN-3.

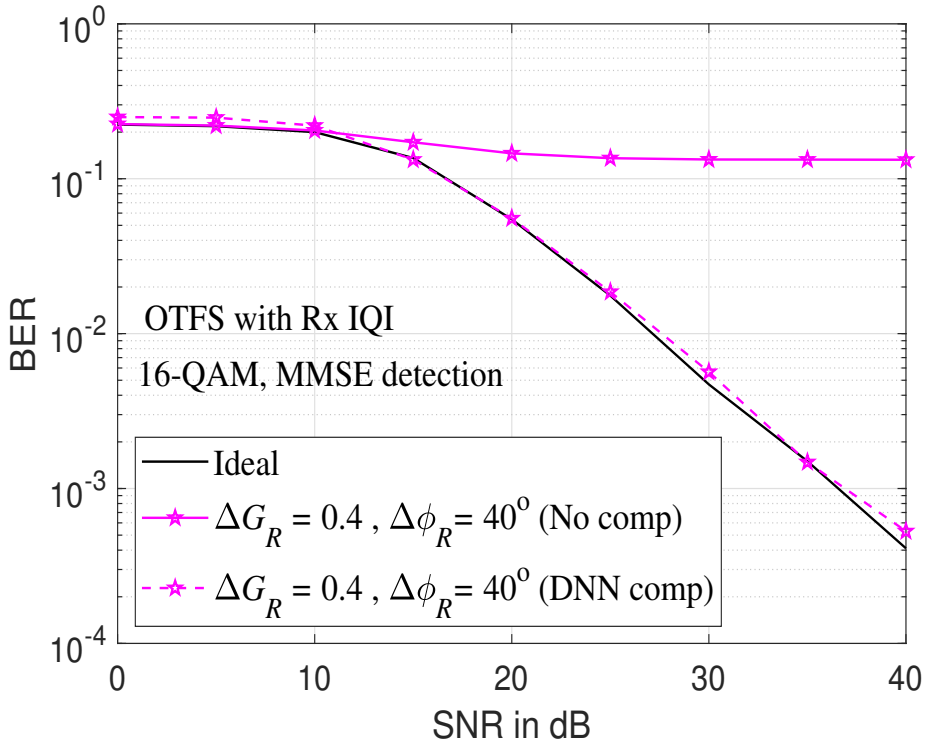
- **Estimation DNN-2:**
  - *Input* $\rightarrow$ *4* $\rightarrow$ *Tanh* $\rightarrow$ *8* $\rightarrow$ *Tanh* $\rightarrow$ *2* $\rightarrow$ *Linear*
- **Compensation DNN-3:**
  - *Input* $\rightarrow$ *32* $\rightarrow$ *Tanh* $\rightarrow$ *64* $\rightarrow$ *Tanh* $\rightarrow$ *32* $\rightarrow$ *Linear*

Rx IQI compensation: 4-QAM, 16-QAM

- Performance of Rx IQI estimation and compensation DNNs for 4-QAM and 16-QAM

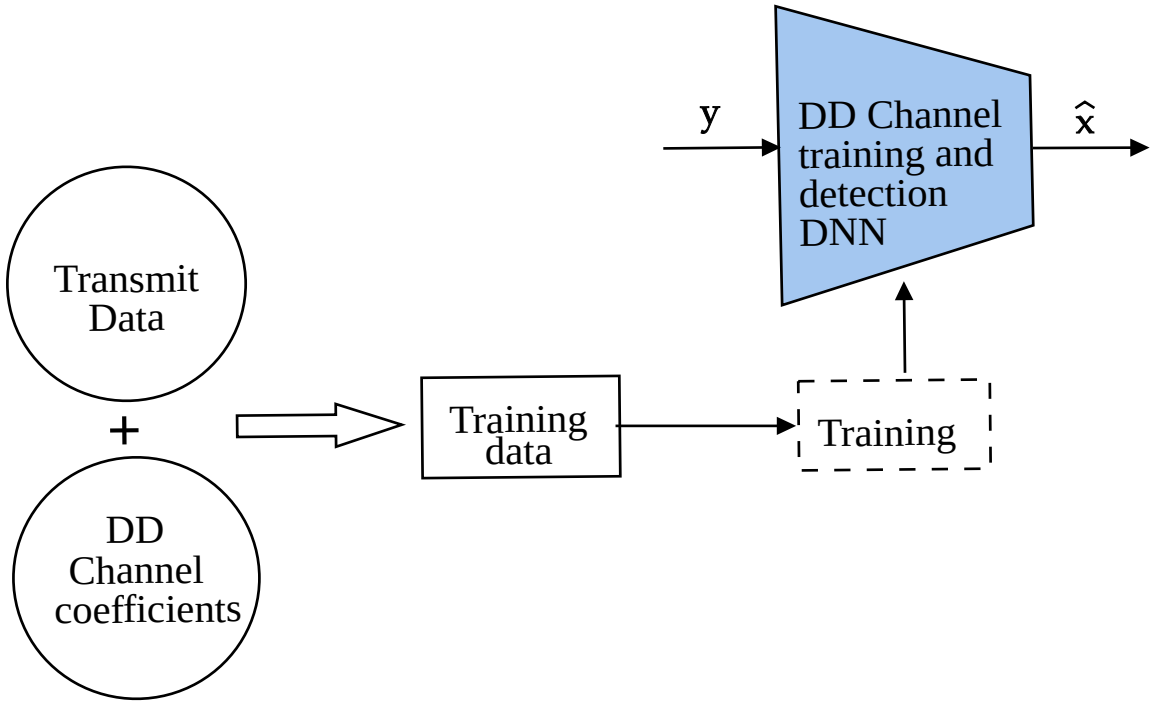


((e)) 4-QAM



((f)) 16-QAM

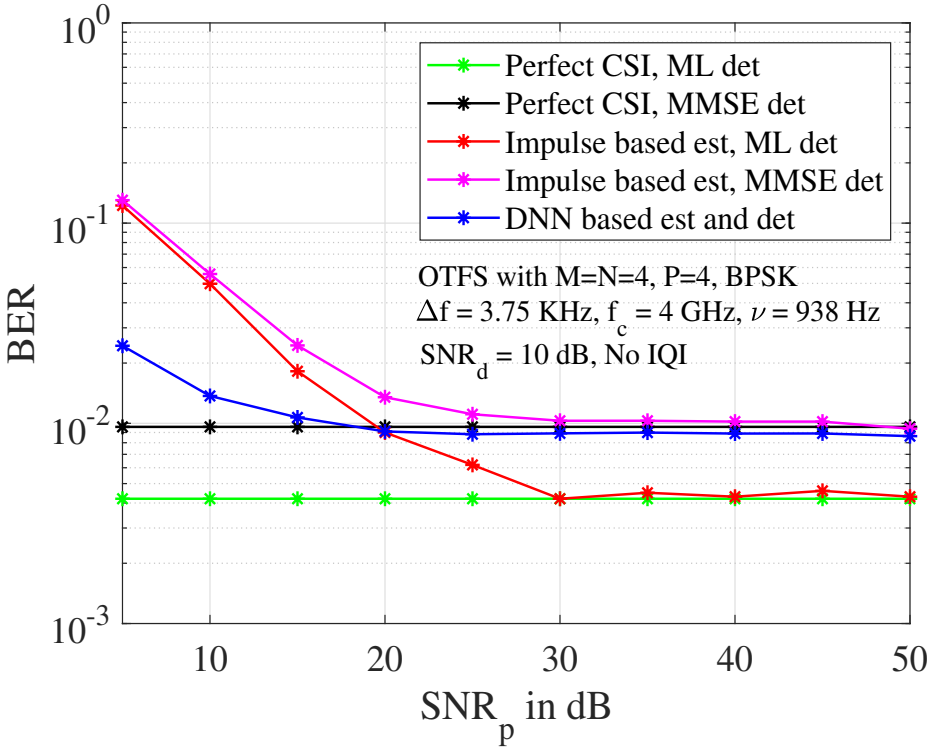
Channel training and detection: DNN-4



- $4MN$ input neurons and MN output neurons
- Input to the DNN consists of two OTFS frames - pilot frame and data frame
- These two frames are vectorized to form the input to the DNN as $\mathbf{y} = [\mathbf{y}_p; \mathbf{y}_d]$

Channel training and detection: DNN-4

- Performance of channel training and detection DNN-4

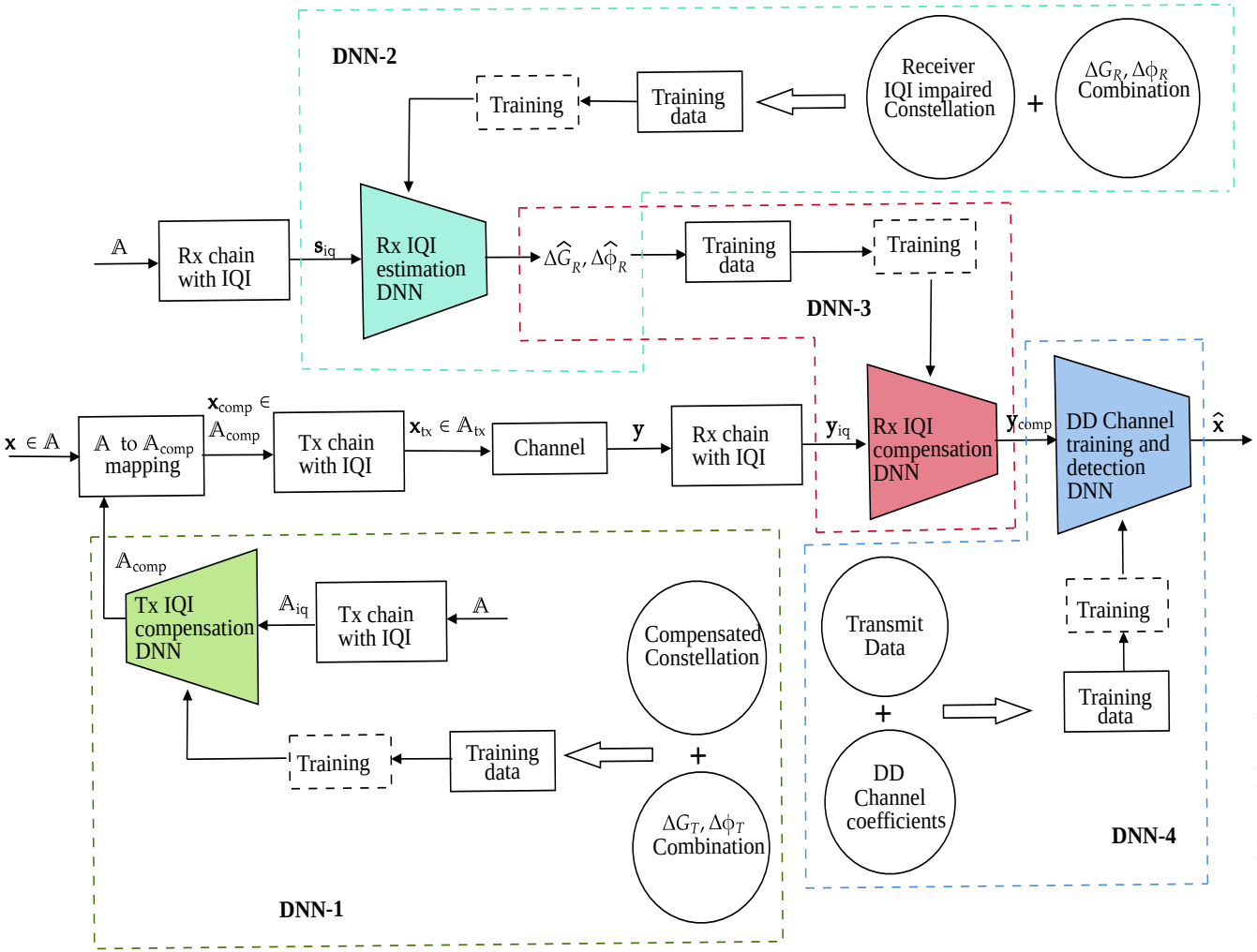


Parameters	DNN-4
No. of input neurons	$4MN = 64$
No. of output neurons	$MN = 16$
No. of hidden layers	2
Hidden layer activation	ReLU
Output layer activation	Sigmoid
Training data SNR	10 dB
Training pilot SNR	10 dB
Optimization	Adam
Loss function	MSE
No. of training examples	200000
No. of epochs	500
Batch size	500

Table: Parameters of DNN-4.

- **DNN-4:**
  - *Input* $\rightarrow$ *64* $\rightarrow$ *ReLU* $\rightarrow$ *256* $\rightarrow$ *ReLU* $\rightarrow$ *64* $\rightarrow$ *ReLU* $\rightarrow$ *16* $\rightarrow$ *Sigmoid.*

DNN-based OTFS transceiver



DNN-1: Tx IQI Compensation DNN

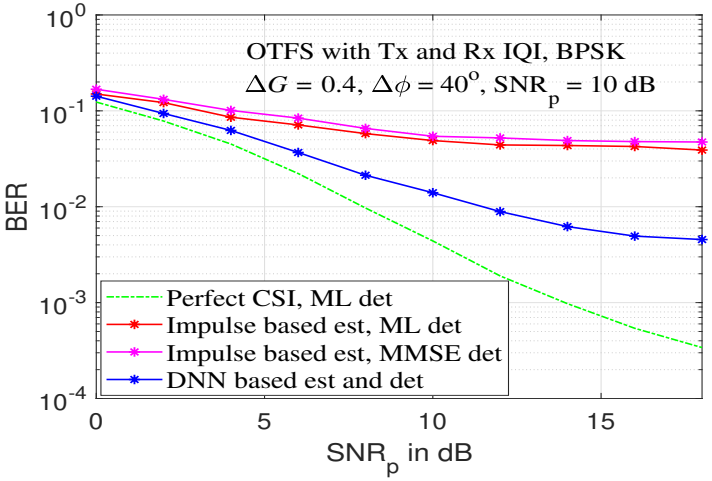
DNN-2: Rx IQI Estimation DNN

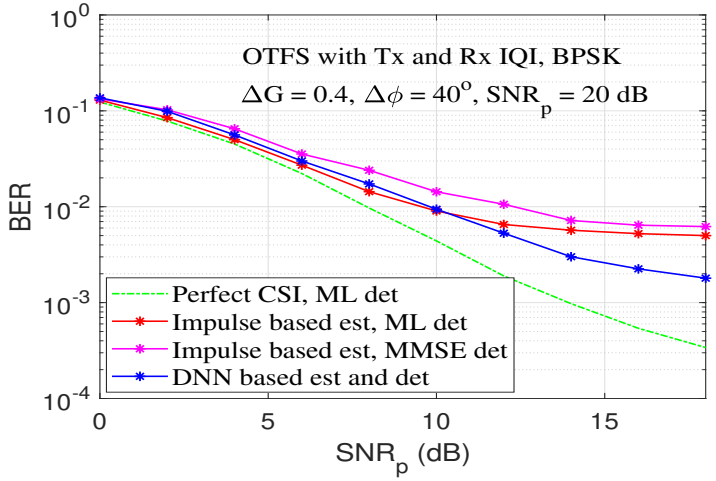
DNN-3: Rx IQI Compensation DNN

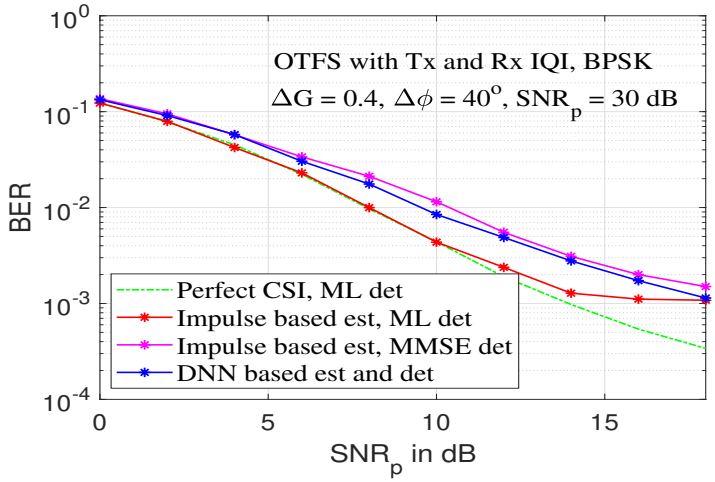
DNN-4: DD Channel Training and Detection

Combined performance of DNN-based OTFS transceiver

- Combined performance of the DNN-based transceiver at different pilot SNRs







- DNN-based transceiver makes a big difference in low pilot SNR regime

Concluding remarks

- OTFS is a promising new modulation waveform for 6G and beyond
- DNN approach is a promising approach for the design of practical OTFS transceivers
- DNN approach can score over conventional approaches when there are model mismatches
- Deviations from the standard models can open learning opportunities, leading to better solutions/performance
- Good tool for system design/optimization in dynamic environments
- Need to pay closer attention to training and complexity view points
- Potential for more research/investigations

Thank you