Machine learning models for optical amplifiers

Course 34242
Lecture 3

Darko Zibar
dazi@dtu.dk

Agenda

- Linear and nonlinear regression models
- Neural-networks
- Learning of neural network parameters
- Model averaging
- Data-driven modelling of optical amplifier
- Inverse design of optical amplifier

Reading material

1. Christopher M. Bishop, Pattern Recognition and Machine Learning, Springer 2006
    - Introduction (pp. 1 - 12)
    - Chapter 3 (pp. 137 - 143)
    - Chapter 5 (pp. 225 - 246)
    - Chapter 14 (pp. 655 - 657)

http://dai.fmph.uniba.sk/courses/NN/haykin.neural-networks.3ed.2009.pdf

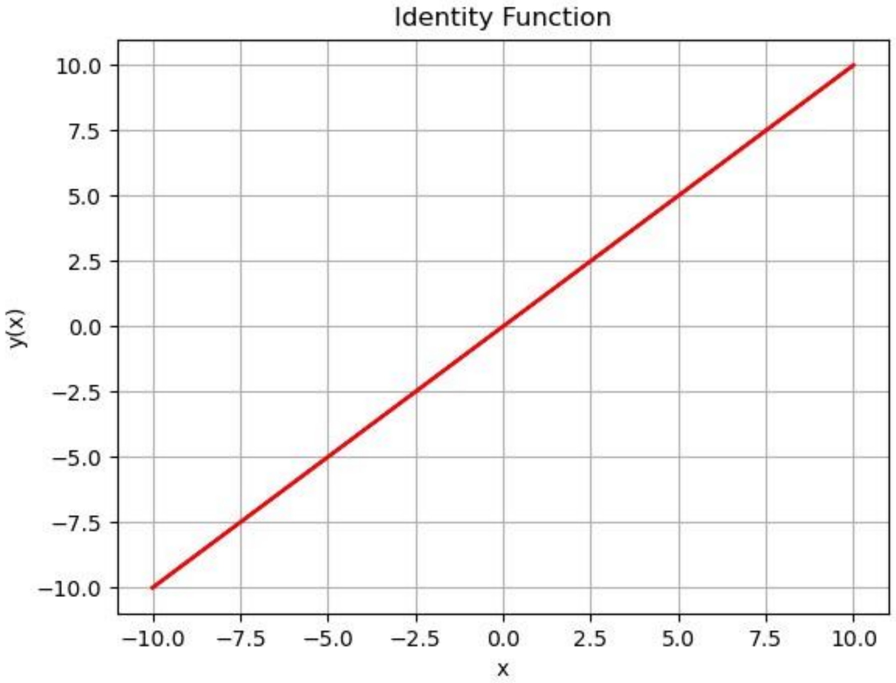
2. B. Widrow, “The No-Prop algorithm: A new learning algorithm for multilayer neural networks,” Neural Networks, 2013

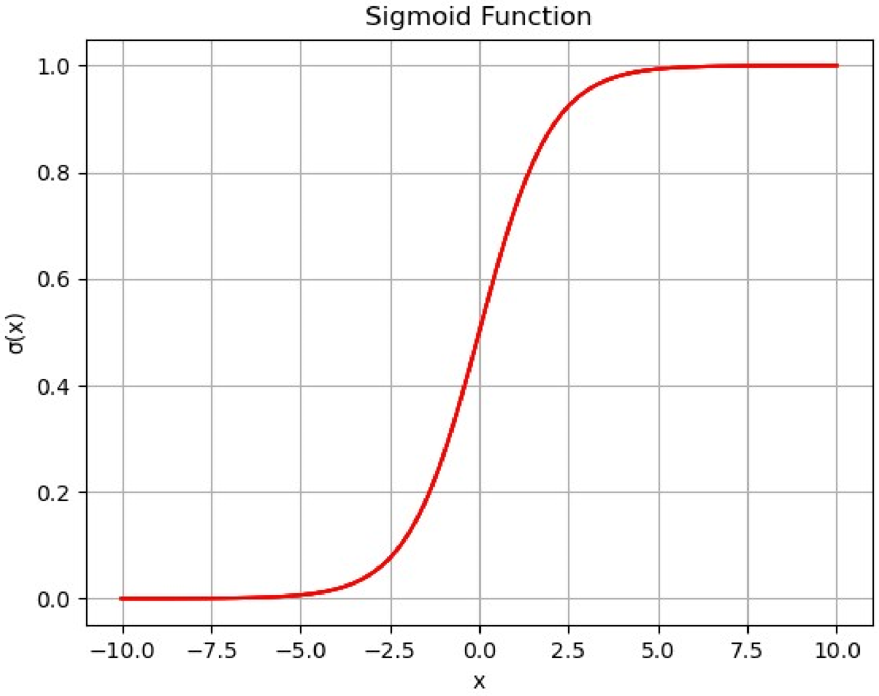
3. D. Zibar et a., “Inverse System Design Using Machine Learning: The Raman Amplifier Case,” Journal of Lightwave Technology, 2020

Linear models

Linear models for regression and classification: $\boxed{\hat{y}(\mathbf{x}, \mathbf{w}) = f\left(\sum_{j=1}^{M} w_j \phi_j(\mathbf{x}) \right)}$

For regression $f(\cdot)$ is identity function and for classification $f(\cdot)$ is sigmoid function





Linear model: Determining the weights in one step

Given the data-set: $\mathcal{D} = \{x_k, y_k\}_{k=1}^{K}$

Given the model: $\hat{y}_k(x_k, \mathbf{w}) = \sum_{j=0}^{3} w_j \phi_j(x_k)$ and assuming $y_k = \hat{y}_k(\mathbf{w}, x_k)$

We can create a linear set of equations in unknown: $\mathbf{W}$

$$\begin{bmatrix} y_1 \\ y_2 \\ \cdot \\ y_K \end{bmatrix} = \begin{bmatrix} 1 & \phi_1(x_1) & \phi_2(x_1) & \phi_3(x_1) \\ 1 & \phi_1(x_2) & \phi_2(x_2) & \phi_3(x_2) \\ \vdots & \vdots & \vdots & \vdots \\ 1 & \phi_1(x_K) & \phi_2(x_K) & \phi_3(x_K) \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ w_3 \end{bmatrix} \Longrightarrow \boxed{\mathbf{Y} = \mathbf{\Phi W}}$$

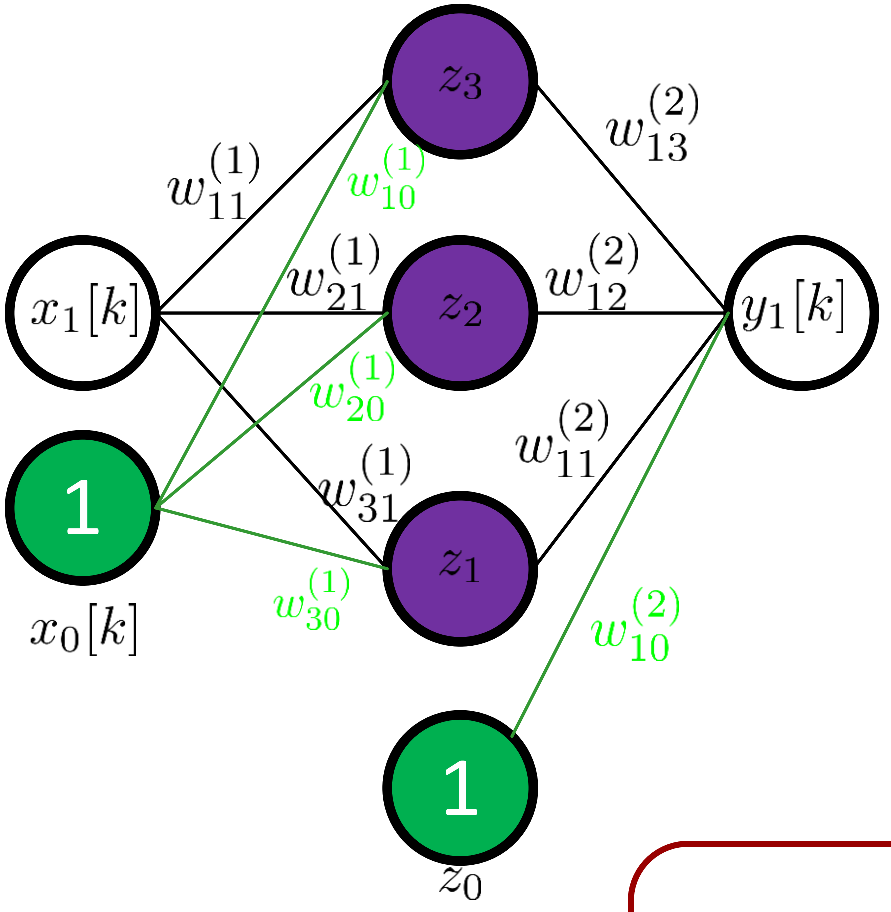
The weights are computed as Moore-Penrose pseudo inverse:

$$\boxed{\mathbf{W} = \left(\mathbf{\Phi}^T \mathbf{\Phi}\right)^{-1} \mathbf{\Phi}^T \mathbf{Y}}$$

Neural-Network

Training data-set:

$x_0[k]$	$x_1[k]$	$t_1[k]$
1	0.344	-1.23
1	0.241	0.67
1	7.100	5.78



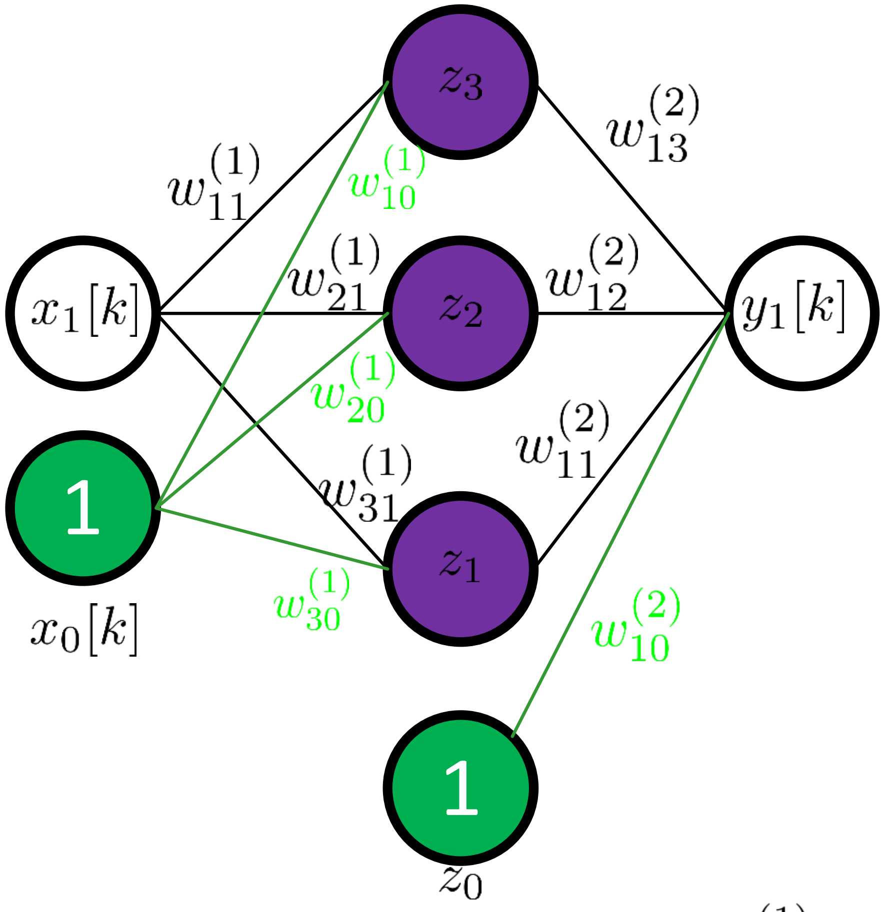
$$y_1[k] \approx t_1[k]$$

$$a_j = w_{j0}^{(1)} x_0[k] + w_{j1}^{(1)} x_1[k] = \begin{bmatrix} w_{0j} & w_{1j} \end{bmatrix} \begin{bmatrix} 1 \\ x_1[k] \end{bmatrix}$$

$$z_j = f(a_j)$$

$$y_1[k] = \sum_{j=0}^{3} w_{1j}^{(2)} z_j = [w_{10}^{(2)}, \ldots, w_{13}^{(2)}] \begin{bmatrix} 1 \\ z_1 \\ \vdots \\ z_3 \end{bmatrix}$$

Propagation through the NN in one step



$$\mathbf{A}^{K\times 3} = \mathbf{X}^{K\times 2}\mathbf{W}^{(1)}$$

$$\Downarrow$$

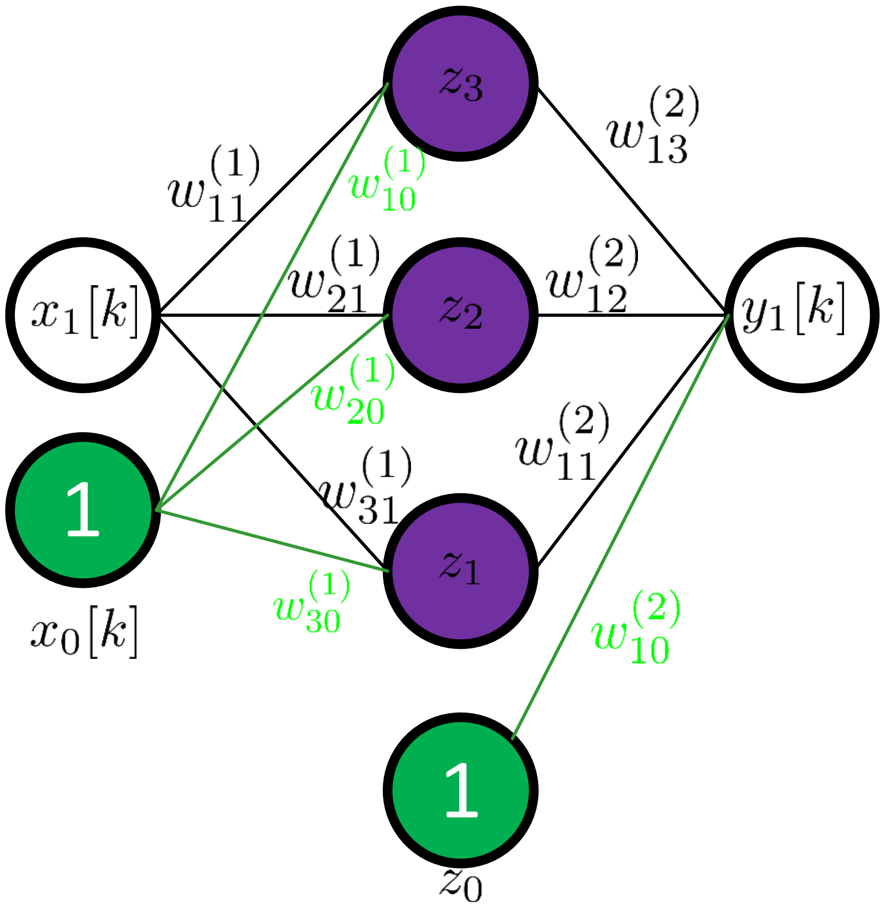
$$\mathbf{Z}^{K\times 4} = [f(\mathbf{A}); \mathbf{1}]$$

$$\Downarrow$$

$$\mathbf{Y}^{K\times 1} = \mathbf{Z}^{K\times 4}\mathbf{W}^{(2)}$$

$$\mathbf{W}^{(1)} = \begin{bmatrix} w_{11}^{(1)} & w_{21}^{(1)} & w_{31}^{(1)} \\ w_{10}^{(1)} & w_{20}^{(1)} & w_{30}^{(1)} \end{bmatrix} \qquad \mathbf{W}^{(2)} = \begin{bmatrix} w_{10}^{(2)} \\ w_{11}^{(2)} \\ w_{12}^{(2)} \\ w_{13}^{(2)} \end{bmatrix}$$

Neural network training using matrix inverse



Training:

$$\mathbf{W}^{(1)} \sim \mathcal{N}(0, \sigma)$$

$\Downarrow$

$$\mathbf{A}^{K\times 3} = \mathbf{X}_{train}^{K\times 2}\mathbf{W}^{(1)}$$

$\Downarrow$

$$\mathbf{Z}^{K\times 4} = [f(\mathbf{A}); \mathbf{1}]$$

$\Downarrow$

$$\mathbf{Y}_{train} = \mathbf{Z}^{K\times 4}\mathbf{W}^{(2)}$$

$\Downarrow$

$$\mathbf{W}^{(2)} = (\mathbf{Z}^T\mathbf{Z})^{-1}\mathbf{Z}^T\mathbf{Y}_{train}$$

Testing:

$$\mathbf{A}^{K\times 3} = \mathbf{X}_{test}^{K\times 2}\mathbf{W}^{(1)}$$

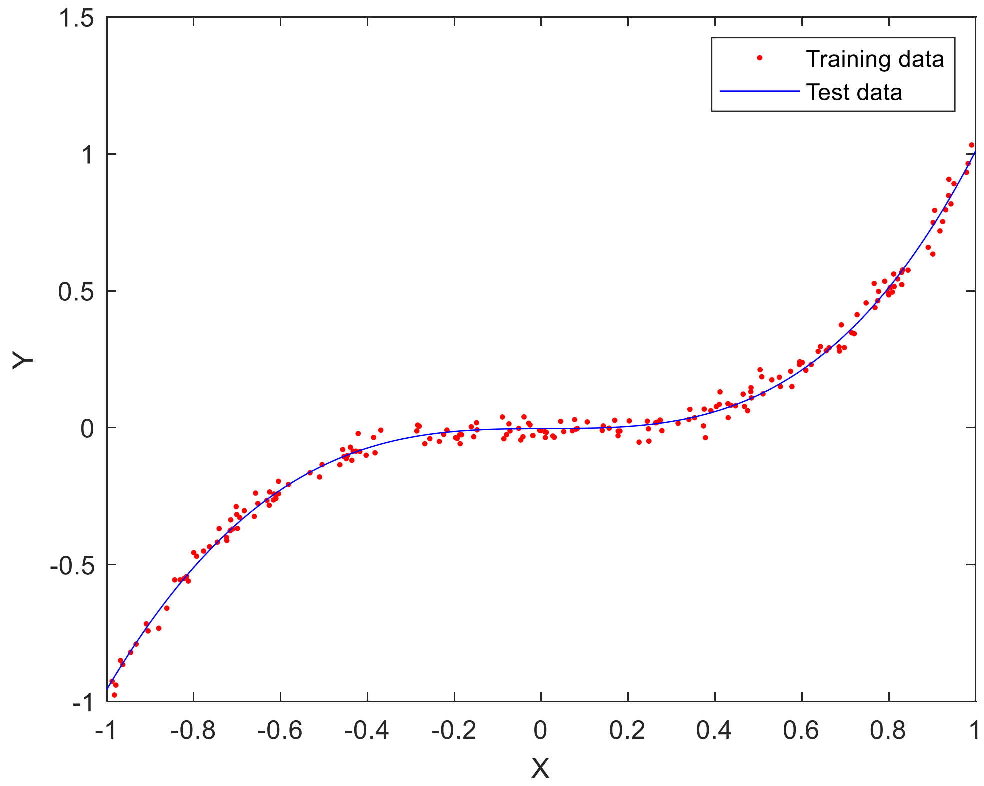
$\Downarrow$

$$\mathbf{Z}^{K\times 4} = [f(\mathbf{A}); \mathbf{1}]$$

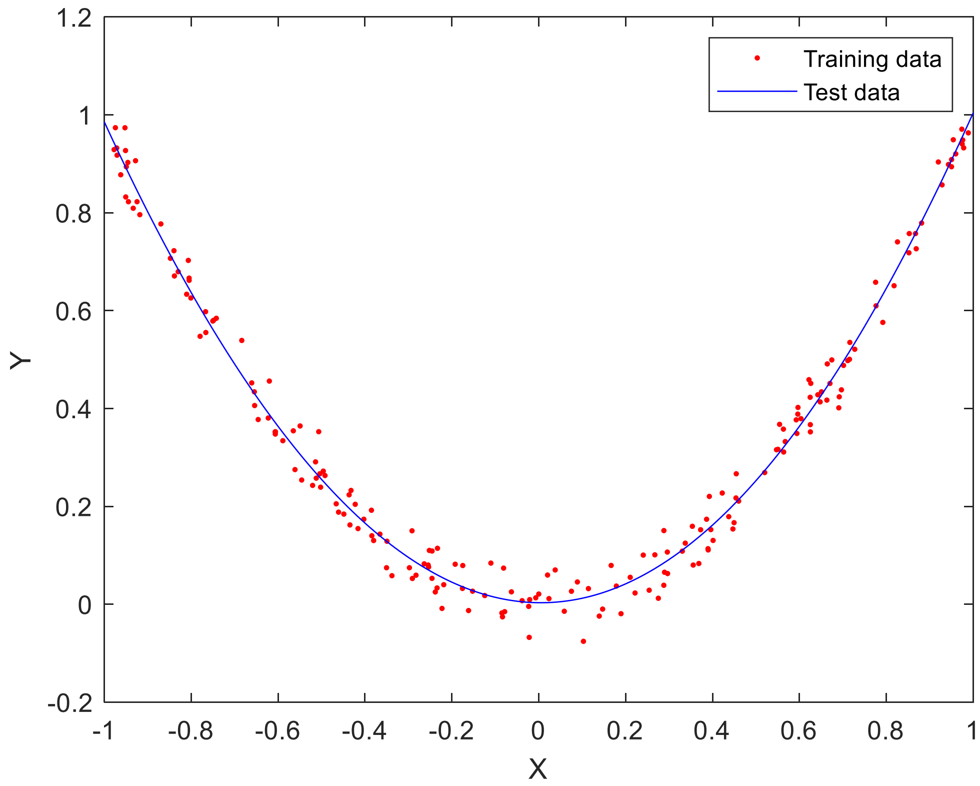
$\Downarrow$

$$\mathbf{Y}_{test}^{(K\times 1)} = \mathbf{Z}^{K\times 4}\mathbf{W}^{(2)}$$

Results

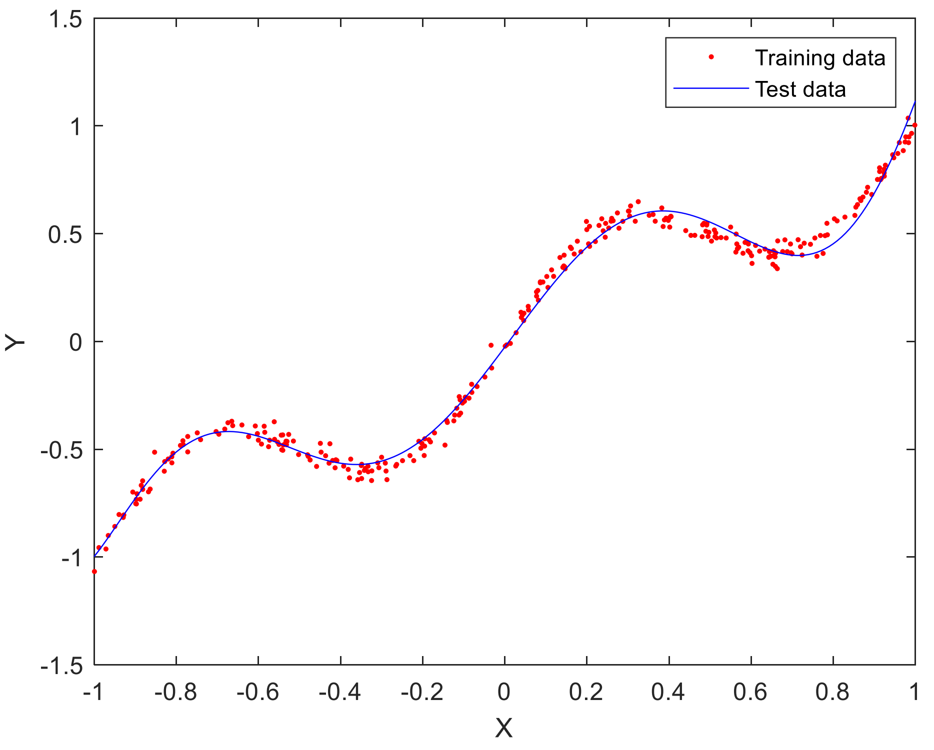


Training data
Test data
1.5
1
0.5
0
-0.5
-1
Y
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
X

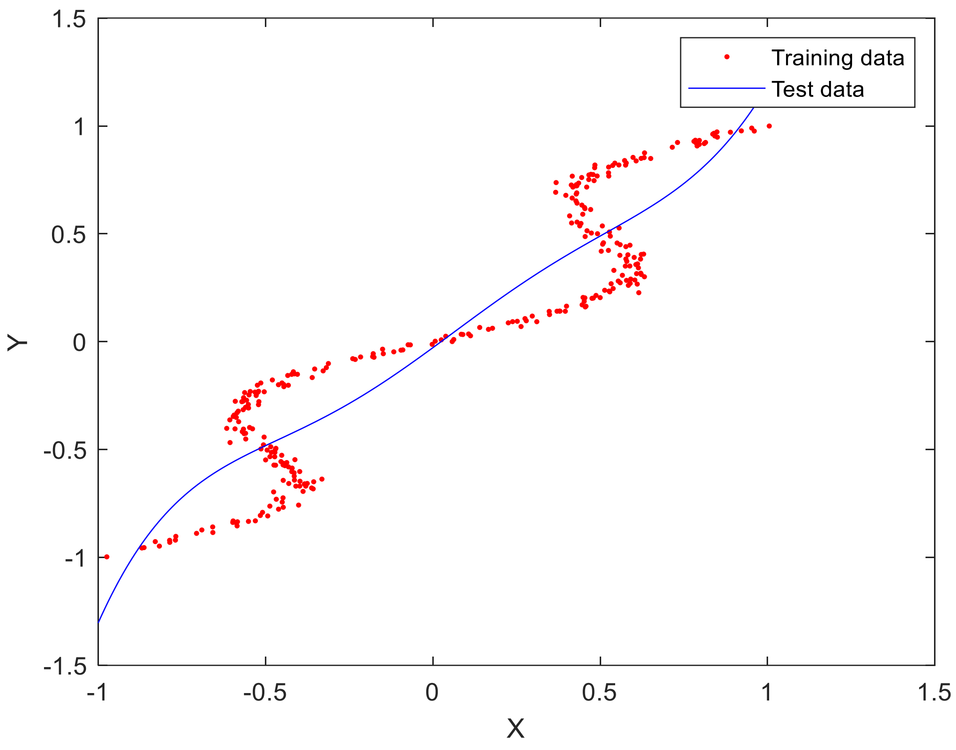


Training data
Test data
1.2
1
0.8
0.6
0.4
0.2
0
-0.2
Y
-1
-0.8
-0.6
-0.4
-0.2
0
0.2
0.4
0.6
0.8
1
X

Learning forward and inverse function

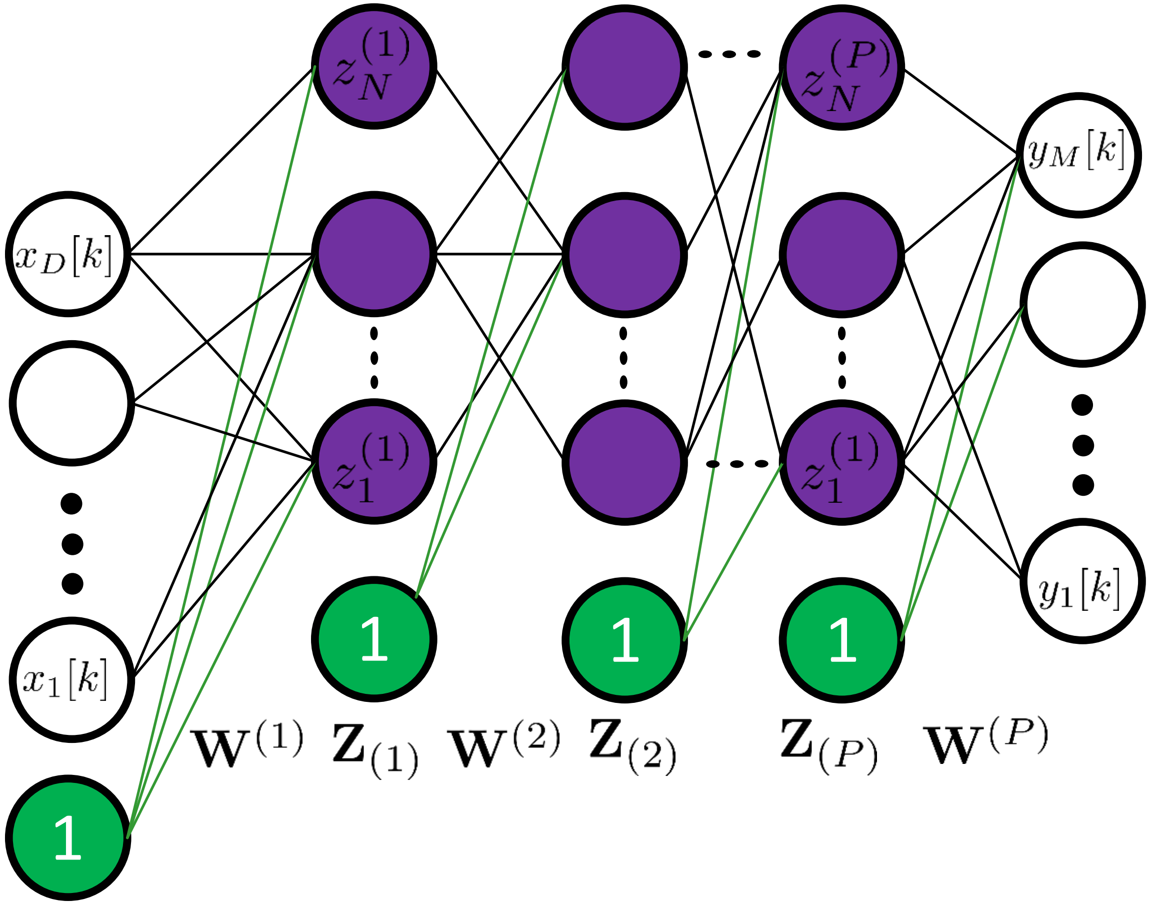


(a) Forward function



(b) Inverse function

Multi-layer network training



$$\mathbf{W}^{(1)} \sim \mathcal{N}(0, \sigma)$$

$$\mathbf{A}^{K\times D} = \mathbf{X}_{train}^{K\times D}\mathbf{W}^{(1)}$$

$$\mathbf{Z}_{(1)}^{K\times N+1} = [f(\mathbf{A}); \mathbf{1}]$$

$$\mathbf{W}^{(2)} \sim \mathcal{N}(0, \sigma)$$

$$\mathbf{A}^{K\times N} = \mathbf{Z}^{K\times N+1}\mathbf{W}^{(2)}$$

$$\Downarrow$$

$$\mathbf{W}^{(P)} = \left(\mathbf{Z}_{(P)}^T\mathbf{Z}_{(P)}\right)^{-1}\mathbf{Z}_{(P)}^T\mathbf{Y}_{train}$$

$$\mathcal{D} = \{x_1[k], x_2[k], ..., x_D[k] | y_1[k], y_2[k], ..., y_M[k]\}_{k=1}^{K}$$

Improving prediction performance

- We can realize different NN models (variability) by:
  - Bootstrapping the data-set
  - Varying activation functions
  - Varying weight initialization

- Improved performance can be obtained by combining multiple models

- Train *L* different models, prediction obtained by averaging made by each model (committees)

- Averaging leads to better prediction due to decreased variance

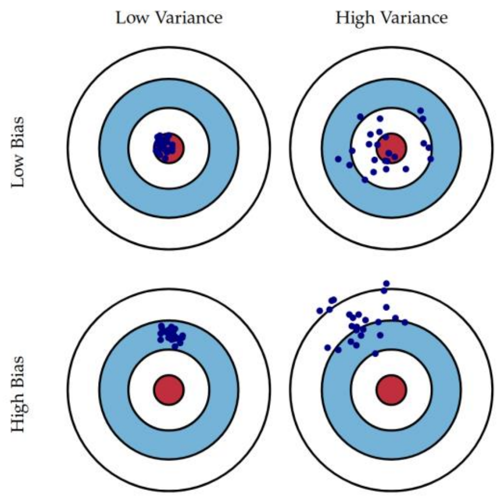


Fig. 1 Graphical illustration of bias and variance.

Committees

Consider regression problem of learning function: $h(\mathbf{x})$

Assume we can train *M* different models $y_m(\mathbf{x})$ where $m = 1, ..., M$

The *committee* prediction is given by: $y_{com} = \frac{1}{M}\sum_{m=1}^{M} y_m(\mathbf{x})$

The output of each model expressed as: $y_m = h(\mathbf{x}) + \epsilon_m(\mathbf{x})$

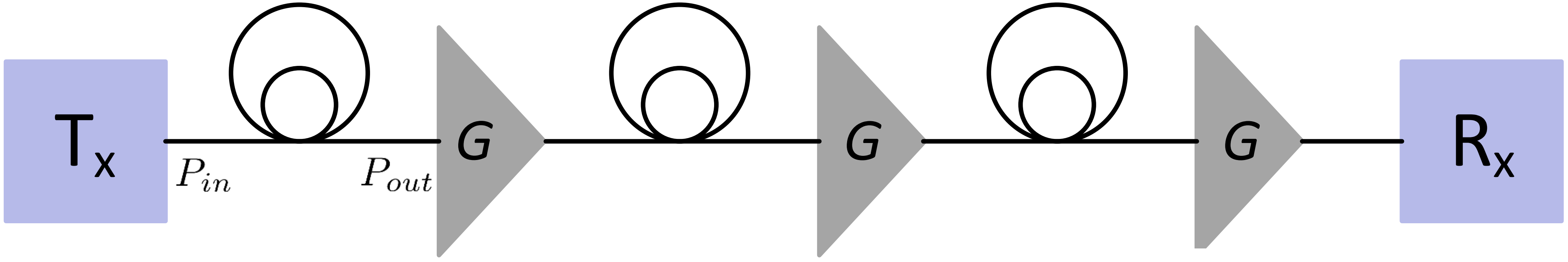
The average sum of squares of *individual model*: $\mathbb{E}_{\mathbf{x}}[(y_m(\mathbf{x}) - h(\mathbf{x}))^2] = \mathbb{E}_{\mathbf{x}}[\epsilon_m(\mathbf{x})^2]$

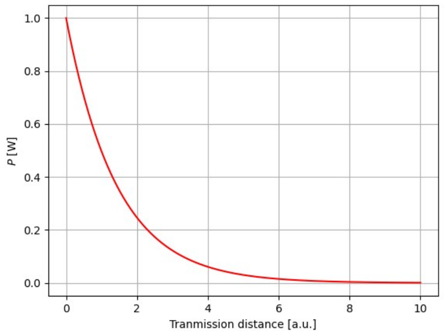
The average error made by the models acting individually: $E_{AV} = \frac{1}{M}\sum_{m=1}^{M} \mathbb{E}_{\mathbf{x}}[\epsilon_m(\mathbf{x})^2]$

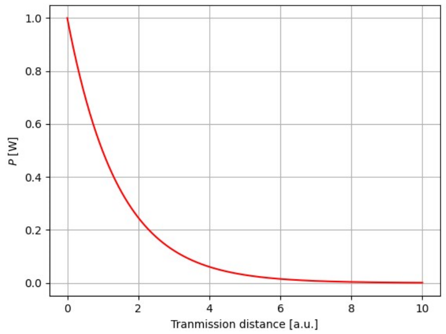
It can be shown that the expected error form the committee:

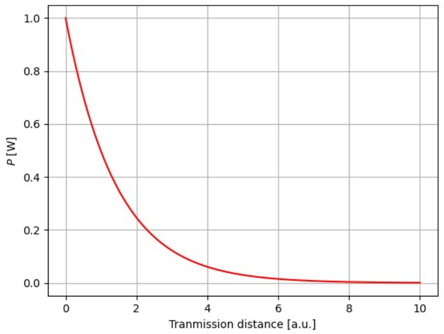
$$\boxed{E_{com} = \frac{1}{M} E_{AV}}$$

Optical transmission system









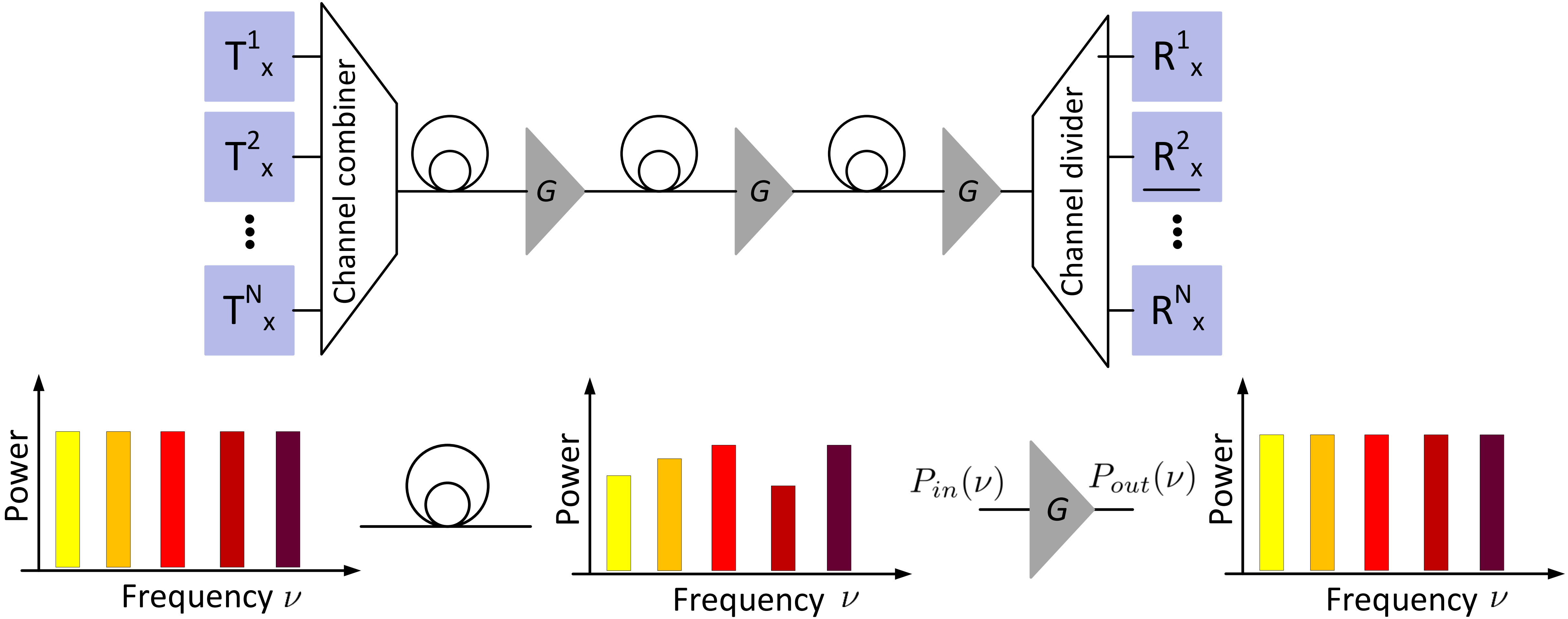
$$P_{out}(z) = P_{in}e^{-\alpha z}$$

$$P_{in} = GP_{out}(L)$$

α : fiber attenuation

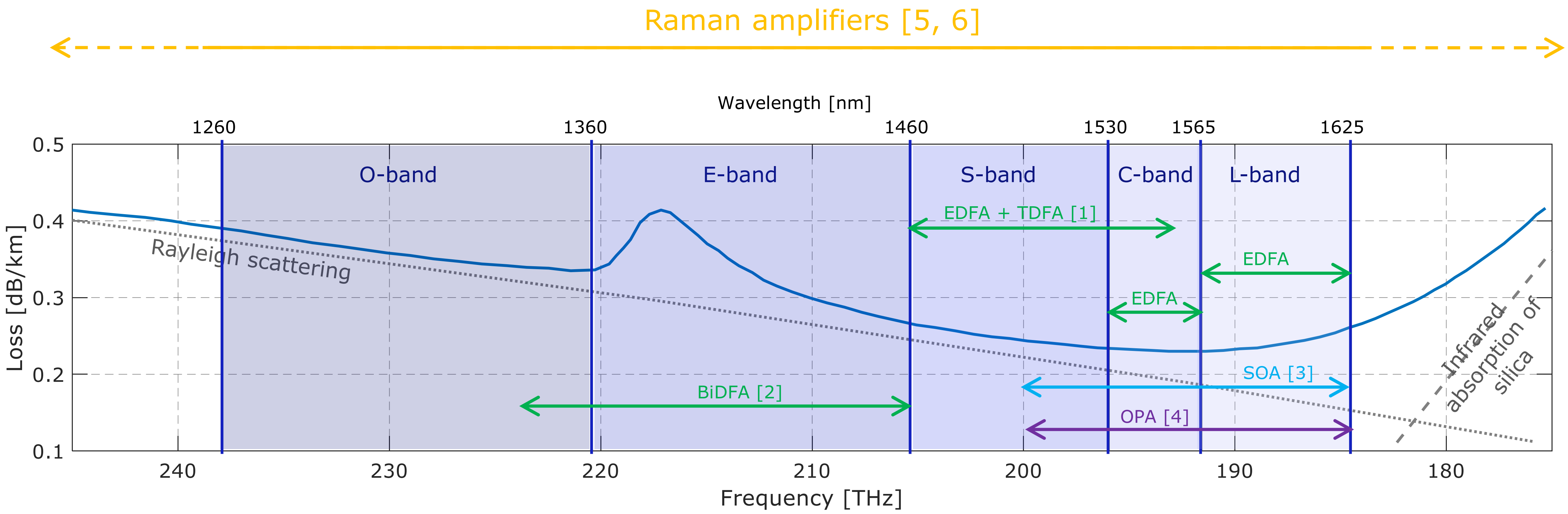
z : variable (distance)

Multi-channel transmission system



$$\boxed{G(\nu) = \frac{P_{out}(\nu)}{P_{in}(\nu)}}$$

Ultra-wideband optical amplification



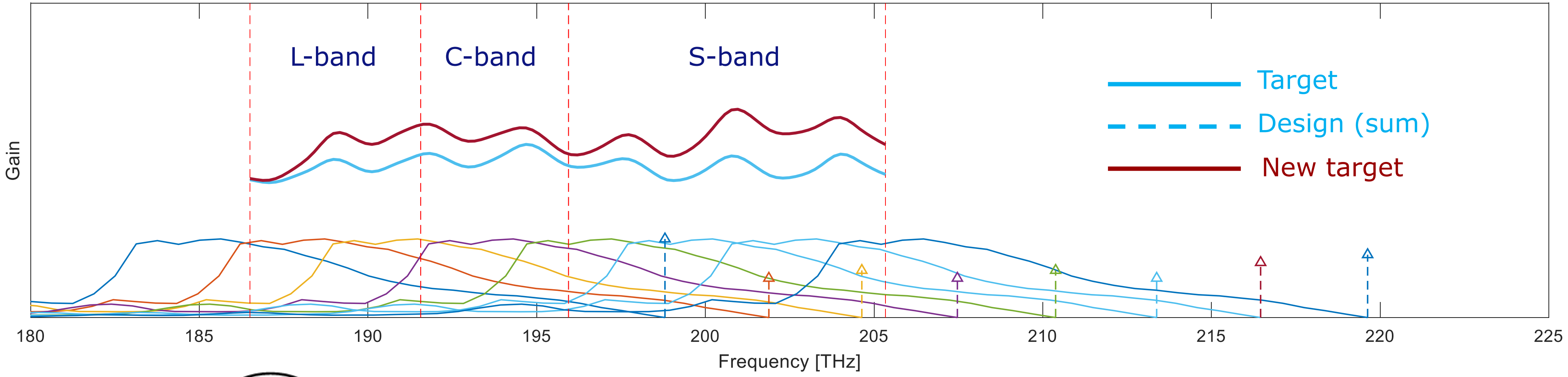
xDFA: doped fiber amplifier

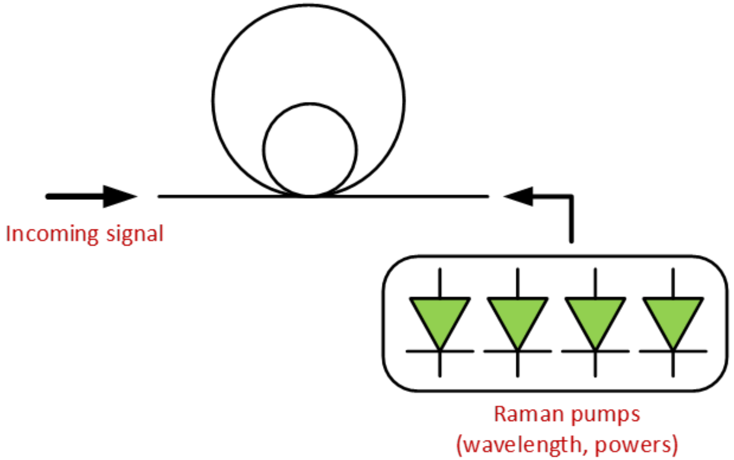
SOA: semiconductor optical amplifier

OPA: optical parametric amplifier

[1] T. Sakamoto, JLT, vol. 24, no. 6, 2006
[2] Y. Wang, OFC 2020, Th4B.1
[3] J. Renaudier, ECOC, 2018
[4] T. Kobayashi, OFC 2020, Th4C.7
[5] J. Chen, IEEE Photonics Journal, vol. 10, 2018
[6] M. A. Iqbal, OFC 2020, W3E.4

Arbitrary gain Raman amplifiers





$$\frac{dP_s}{dz} = -\alpha_s P_s + C_R(\lambda_s, \lambda_p)\left[P_p^+ + P_p^-\right] P_s \qquad (1)$$

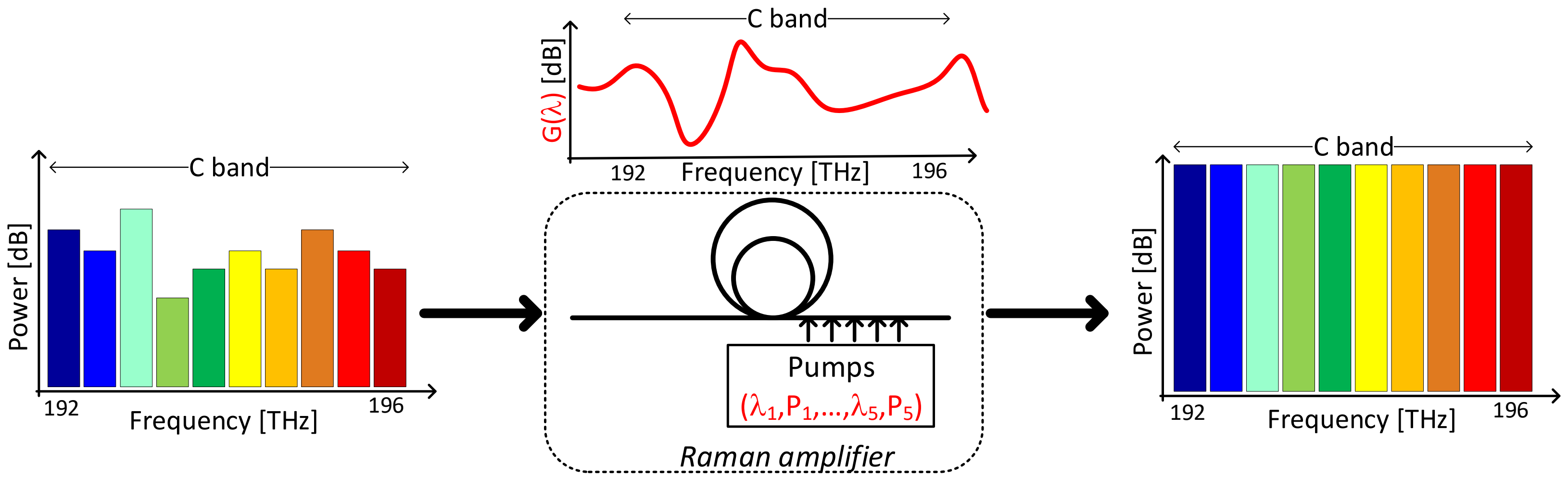
$$\pm\frac{dP_p^\pm}{dz} = -\alpha_p P_p^\pm - \left(\frac{\lambda_s}{\lambda_p}\right) C_R(\lambda_s, \lambda_p)\, P_s\, P_p^\pm \qquad (2)$$

$$\pm\frac{dP_A^\pm}{dz} = -\alpha_A P_A^\pm + C_R(\lambda_A, \lambda_p) P_p P_A^\pm + C_R(\lambda_A, \lambda_p)[1 + \eta(T)] h\nu_A B_{ref} P_p \qquad (3)$$

Gain in Raman amplifiers nonlinear function of pump powers and frequencies:

$$G(\nu) = f\left(P_p^{(1)}, \dots, P_p^{(N)}, \nu_p^{(1)}, \dots, \nu_p^{(N)}\right)$$

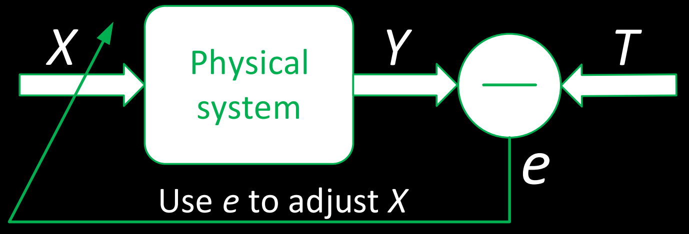
Arbitrary gain profile amplifiers



Proper adjustment of pump powers and frequencies is needed to obtain the specific gain

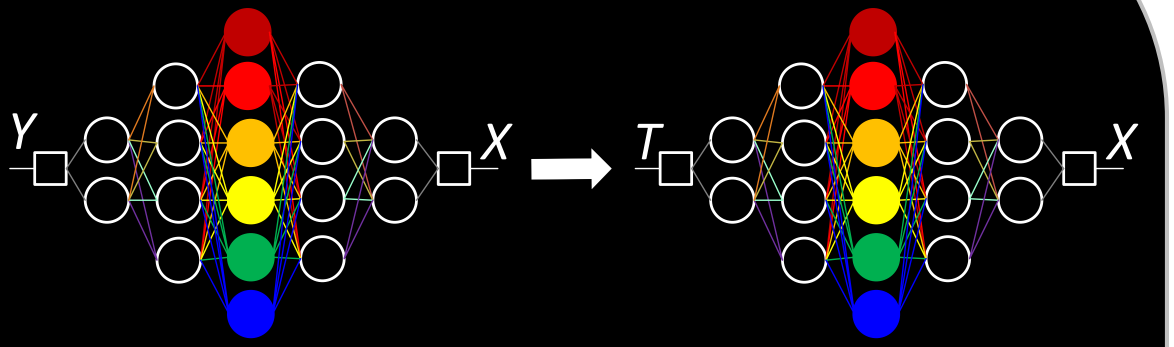
Inverse system learning

#1 Problem statement:

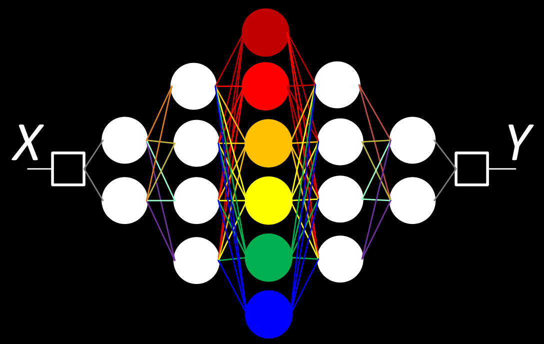


A physical system describing relation between input *X* and output *Y* is given. The objective is to determine input *X* that would result in a targeted output *T*.

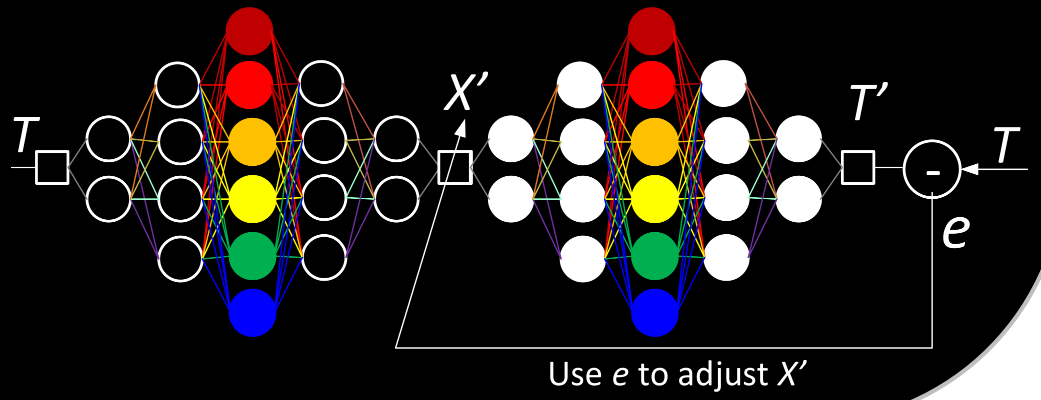
#2 Train neural network to learn *inverse* mapping (from *X* to *Y*):



#3 Train neural network to learn *forward* mapping (from *X* to *Y*):



#4 Perform *final* optimization:



[1] D. Zibar et al., ”Inverse system design using machine learning: the Raman amplifier case,” Journal of Lightwave technology, 2019

Building the model from the data

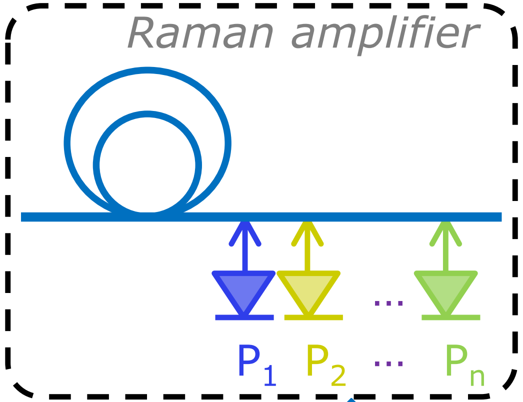
Given N pumps generate M gain profiles

$\lambda_1 \sim U[\lambda_{1,min};\ \lambda_{1,max}]$ nm
$\lambda_2 \sim U[\lambda_{2,min};\ \lambda_{2,max}]$ nm
…
$\lambda_N \sim U[\lambda_{n,min};\ \lambda_{n,max}]$ nm

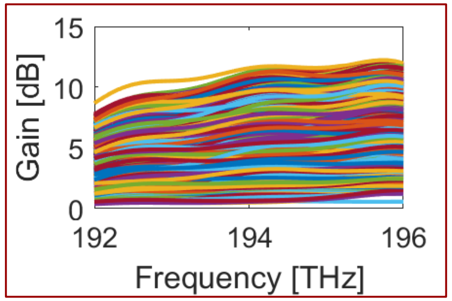
$P_1 \sim U[P_{1,min};\ P_{1,max}]$ W
$P_2 \sim U[P_{2,min};\ P_{2,max}]$ W
…
$P_N \sim U[P_{n,min};\ P_{n,max}]$ W

Numerically

Experimentally



M gain profiles each with K points



Data-set $\Rightarrow$ $\mathcal{D} = \{(\lambda_1^i, \lambda_2^i, \dots, \lambda_N^i, P_1^i, P_2^i, \dots, P_N^i, G_1^i, G_2^i, \dots, G_K^i) | i = 1, \dots, M\}$

Training

Validation

Gather input/output data:

$$\mathcal{D} = \{x(k), y(k)\}_{k=1}^{K}$$
$$\mathcal{D} = \{\mathbf{X}, \mathbf{Y}\}$$

Split the data

Training data:

$$\mathcal{D} = \{\mathbf{X}^{train}, \mathbf{Y}^{train}\}$$

Test data:

$$\mathcal{D} = \{\mathbf{X}^{test}, \mathbf{Y}^{test}\}$$

Select the model:

neural network, polynomial, etc

Learn the model parameters:

$$\mathbf{W} = [w_0, w_1, ..., w_M]$$

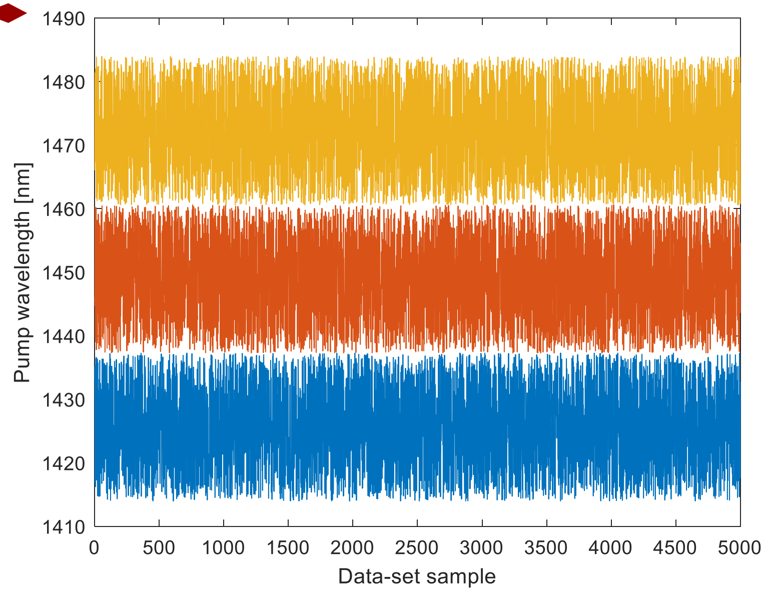
Evaluate the model:

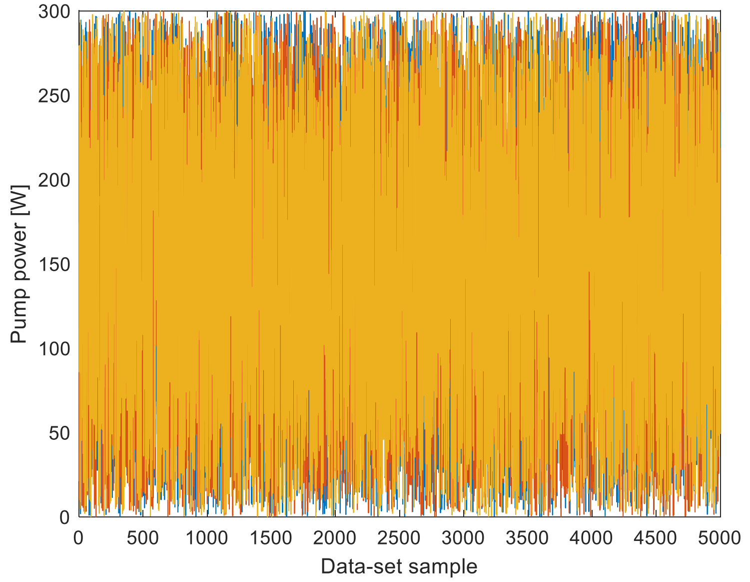
$$\mathbf{Y}^{pred.} = \mathbf{\Phi}(\mathbf{X}^{\mathbf{test}})\mathbf{W}$$

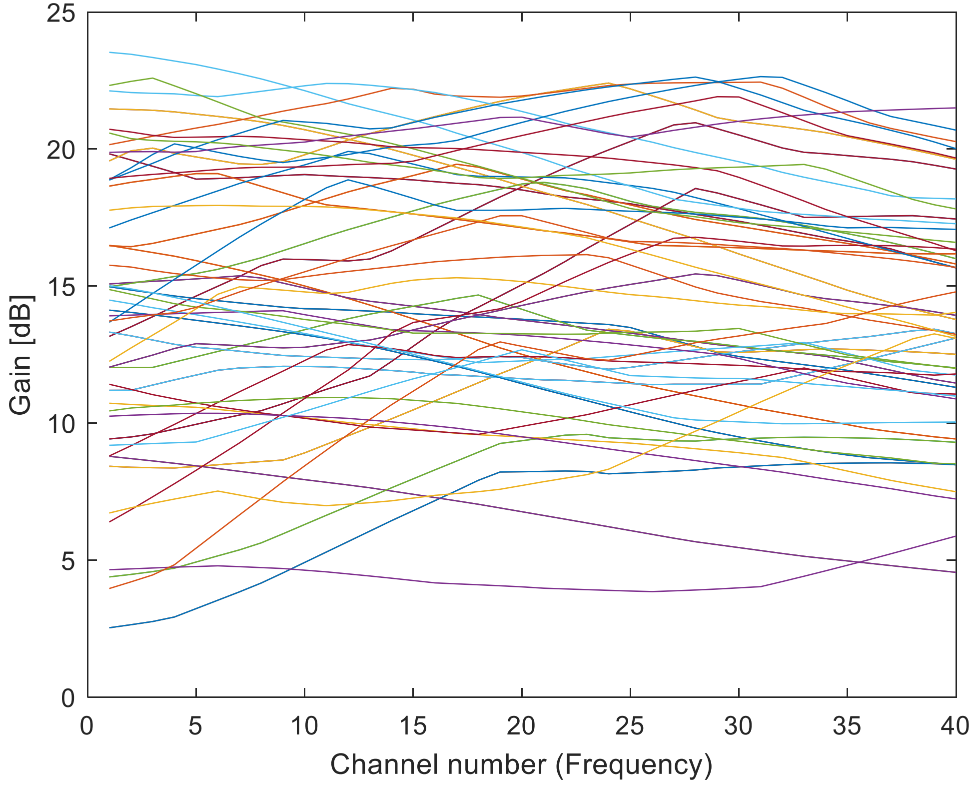
Evaluate the error:

$$e = \sum(\mathbf{Y}^{test} - \mathbf{Y}^{pred})^2$$

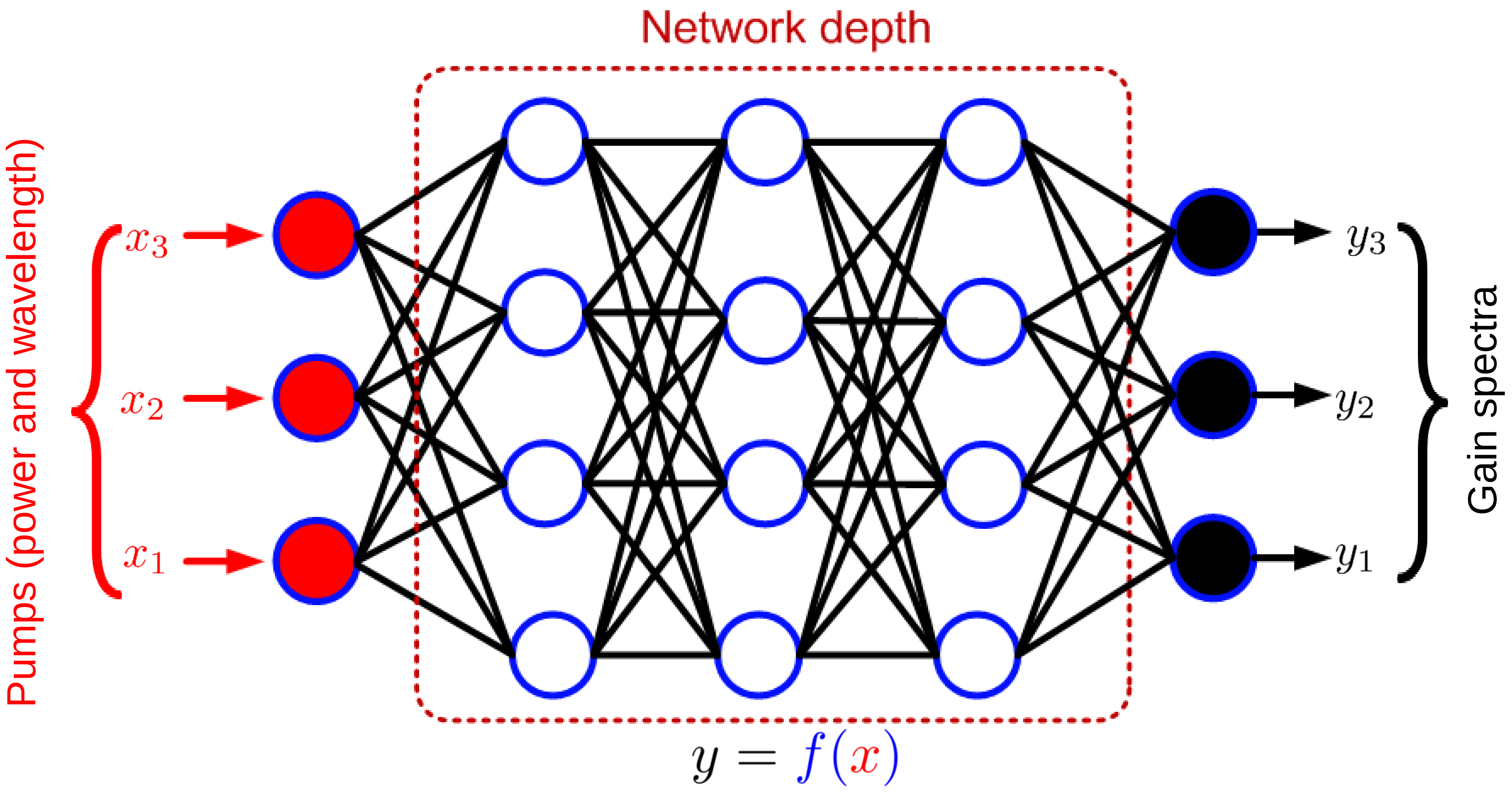
Data-set generation





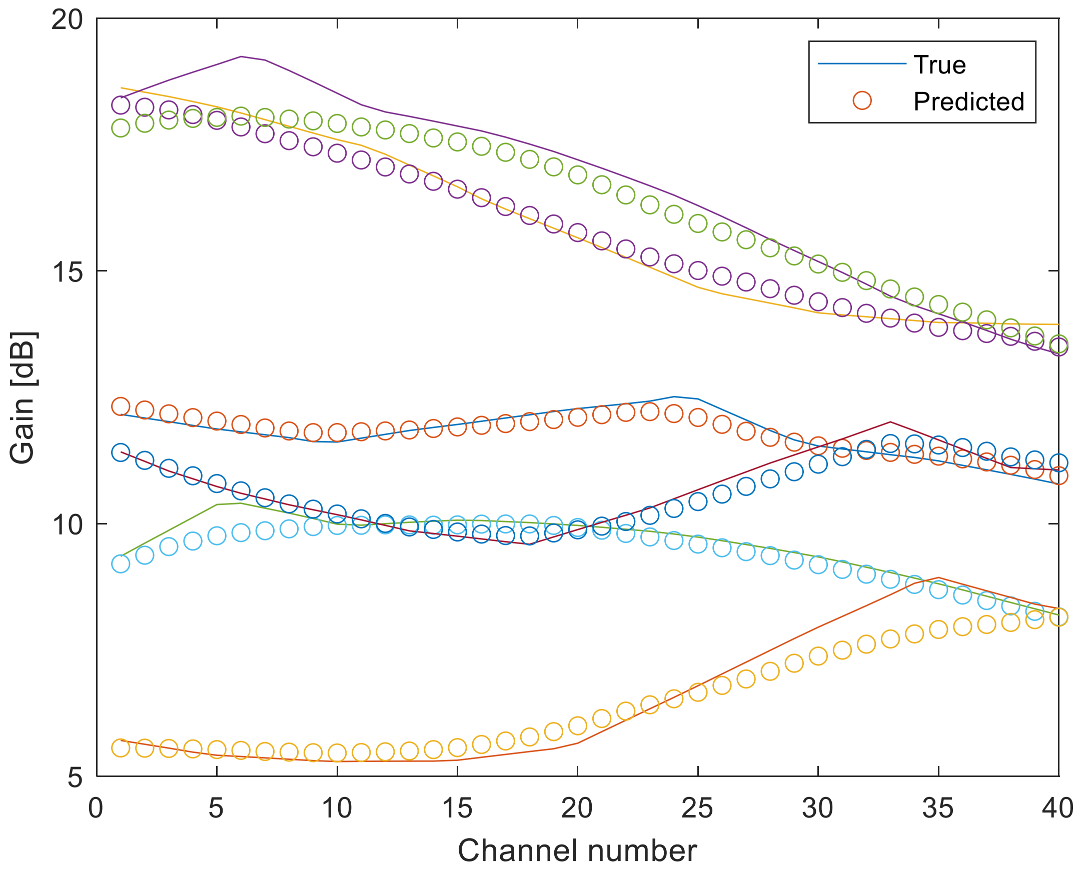


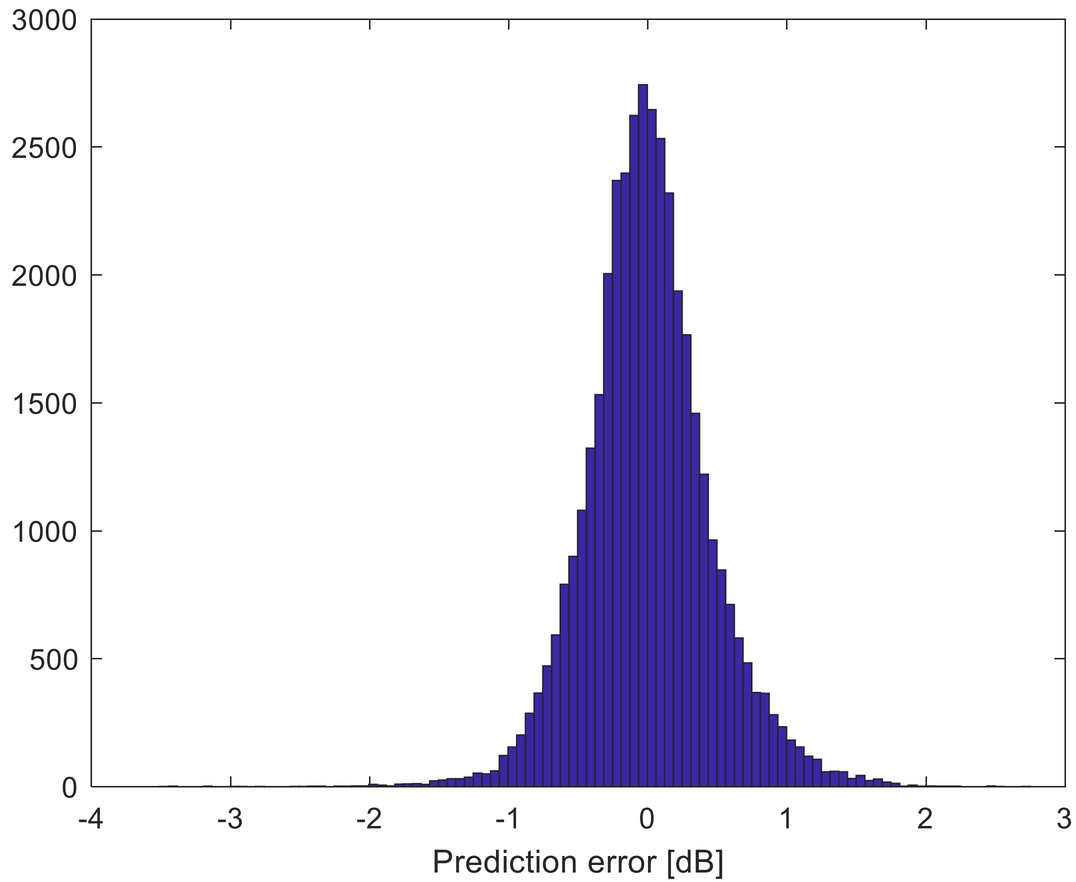
Approximating Raman amplifier with NN



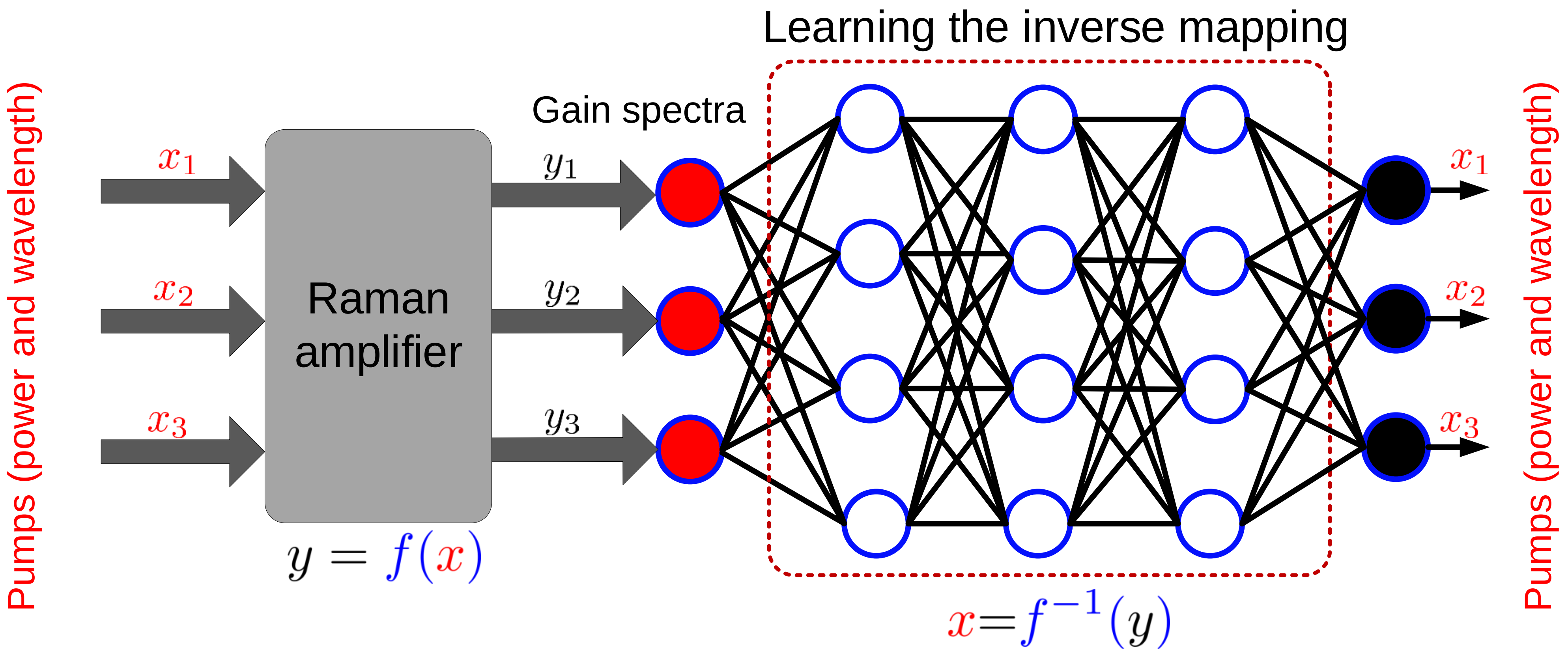
Neural network learns forward mapping, $f(\cdot)$, using training data and perform predictions for *new input* data: $y_{new} = f(x_{new})$

Machine learning model of Raman amplifier (Forward model)



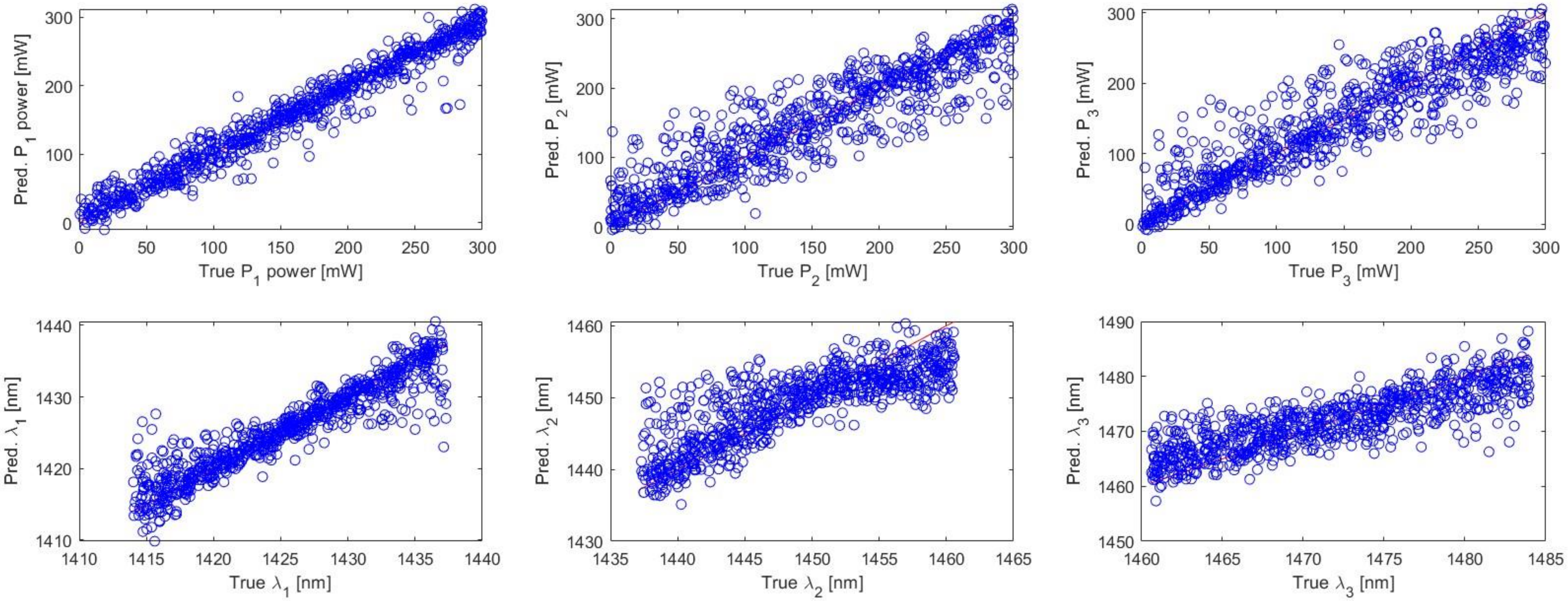


Learning inverse mapping

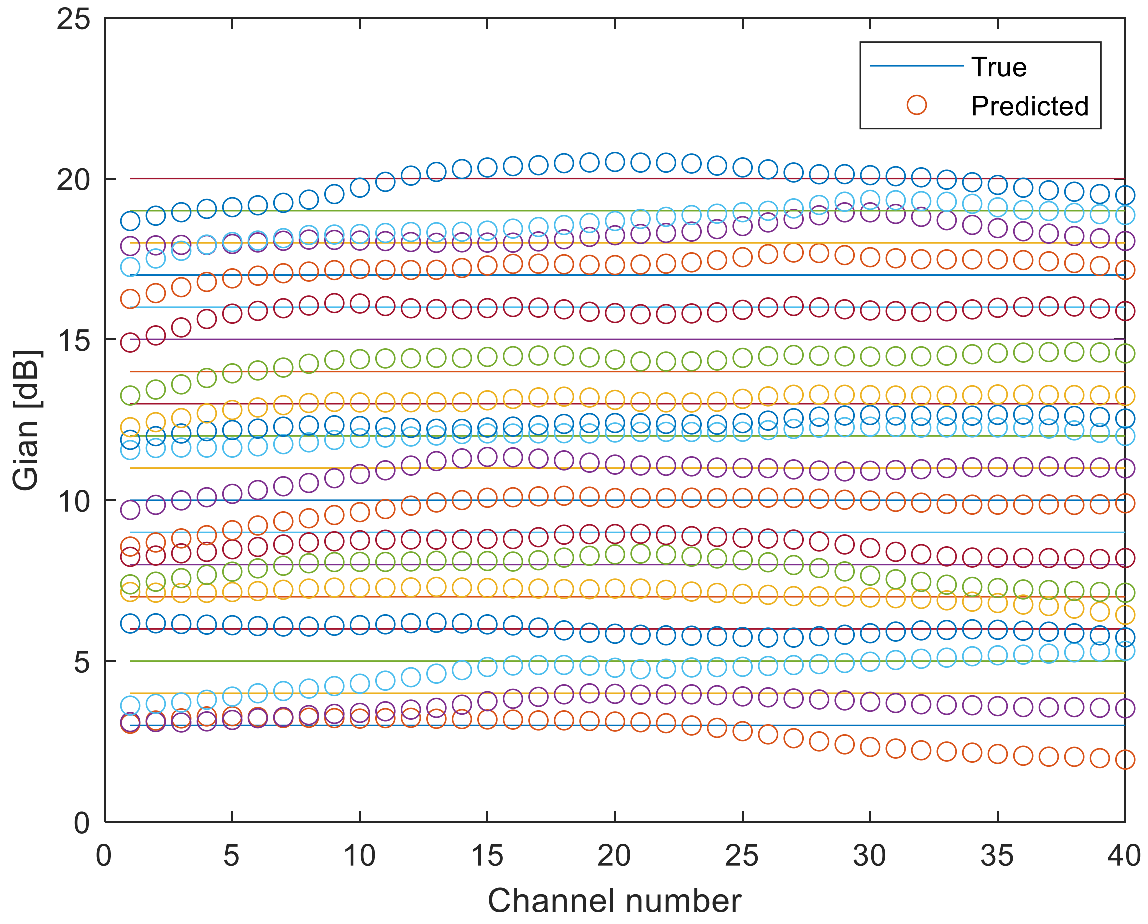


Learning the inverse mapping allows for designing arbitrary gain profile

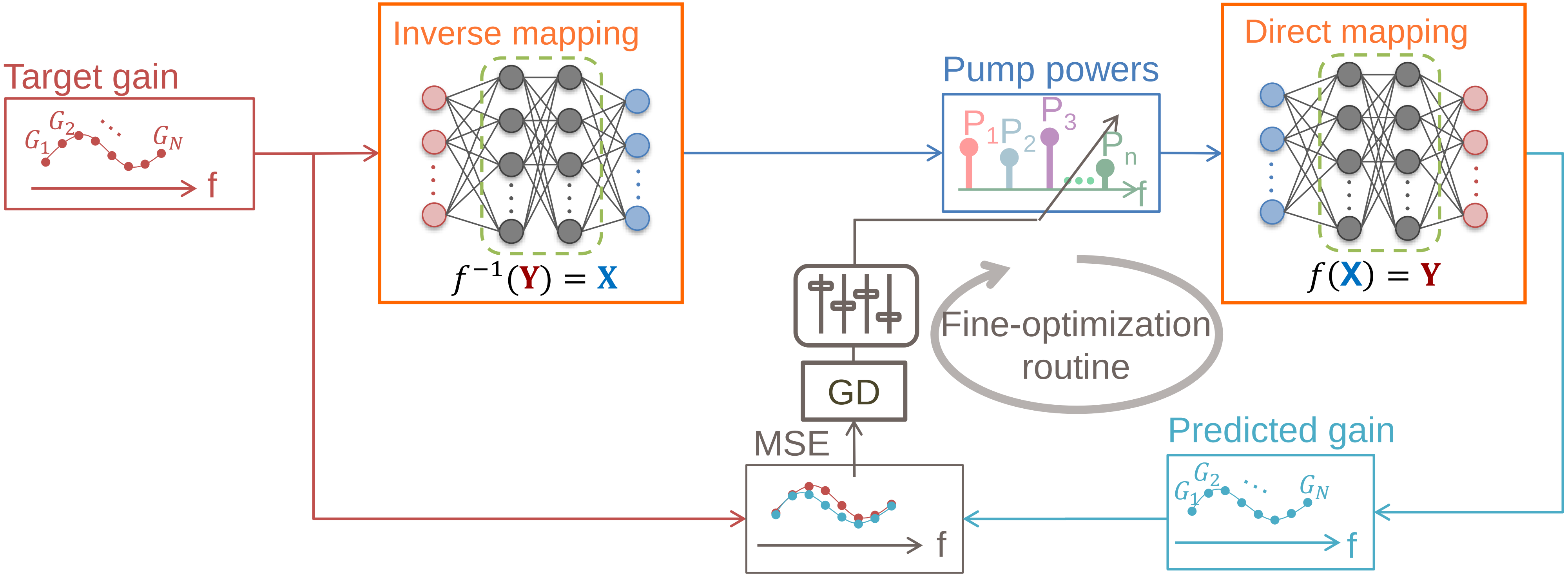
Inverse system learning (backward model)



Inverse system design for flat gain profiles



The machine learning framework



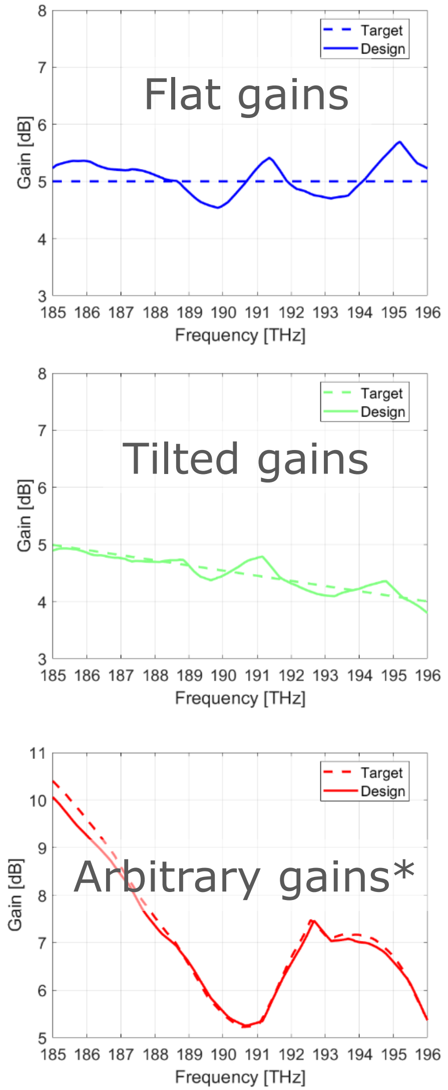
MSE: mean squared error

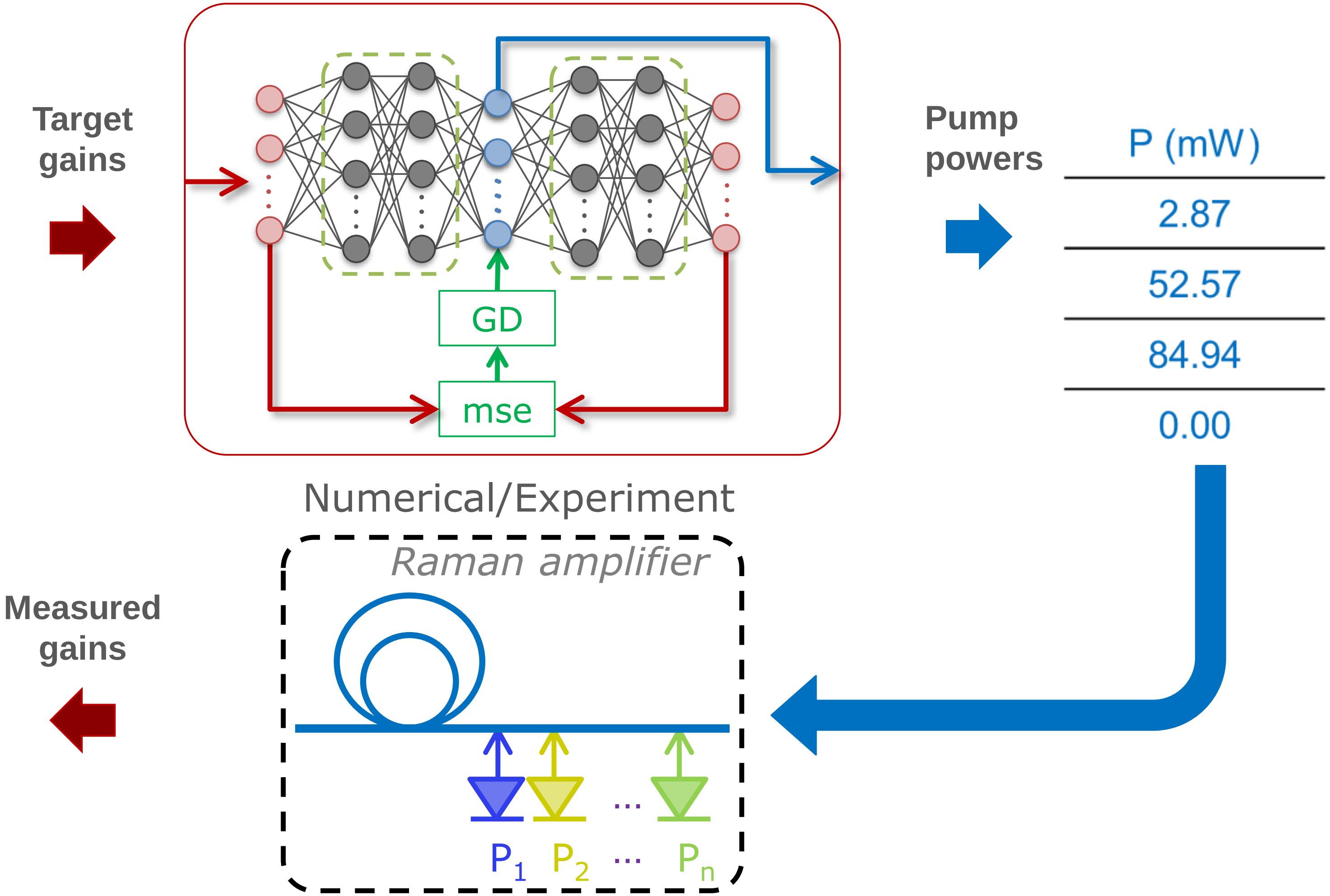
GD: gradient descent

D. Zibar, J. Lightwave Technol. **38**(4), 736–753 (2019)

U. de Moura, J. Lightwave Technol. **39**(4),1162–1170 (2021)

Experimental validation of the learned model

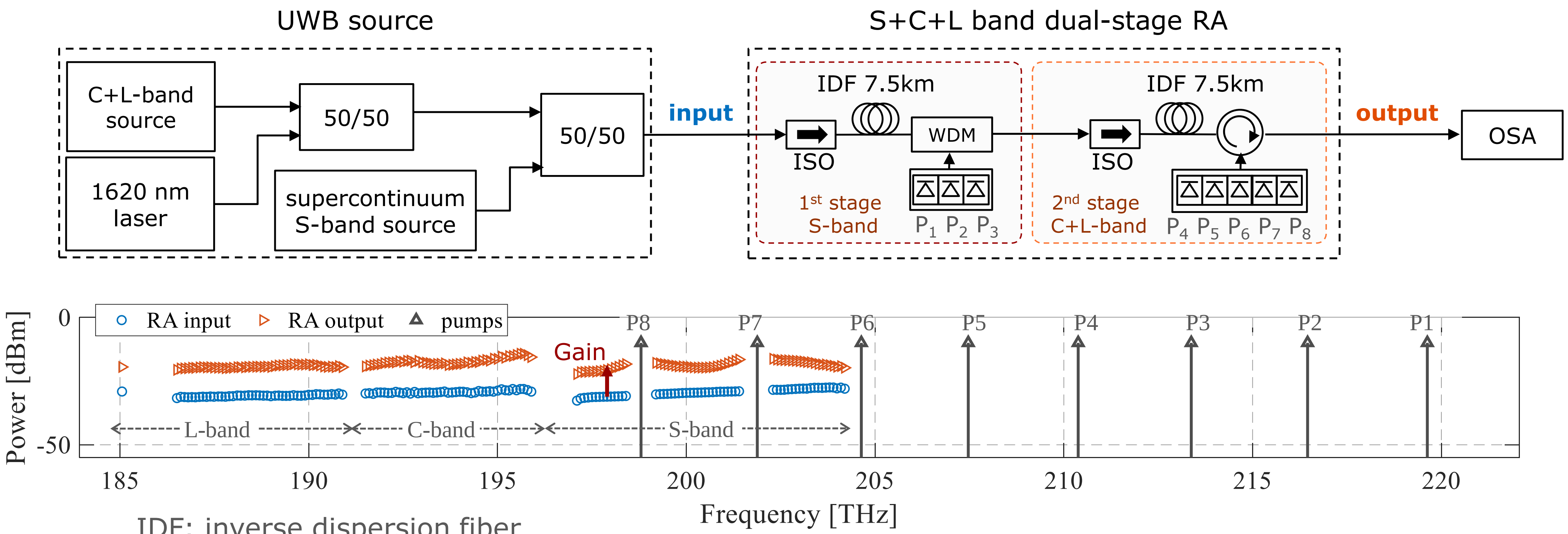




* Part from the acquired data not used on training

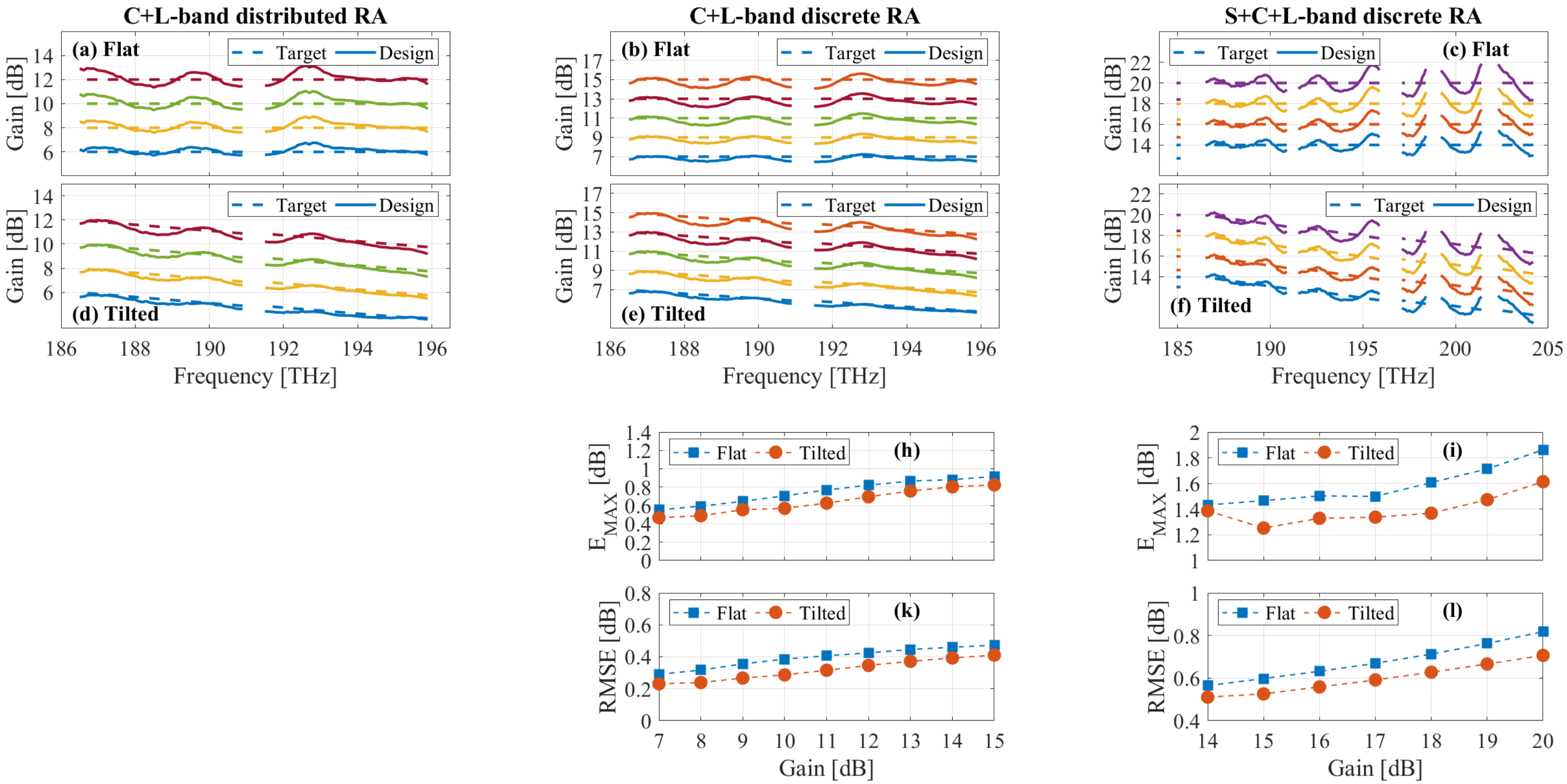
Experimental results for S+C+L band

S+C+L-band discrete RA



IDF: inverse dispersion fiber
RA: Raman amplifier
UWB: ultra-wideband

Experimental results for S+C+L band



In this lecture we have learned….

- How to train neural network
- The importance of optical amplifier in fiber-optic communication
- How to build and test forward model of optical amplifier
- How to build inverse model of optical amplifier
- How to combine forward and inverse models
- Practical application of machine learning for design of ultra-wideband optical amplifier