Inverse System Design Using Machine Learning: The Raman Amplifier Case

Inverse System Design using Machine Learning: the Raman Amplifier Case

Darko Zibar, Ann M. Rosa Brusin, Uiara C. de Moura, Francesco Da Ros, Vittorio Curri and Andrea Carena

***Abstract*—A wide range of highly–relevant problems in programmable and integrated photonics, optical amplification and communication deal with inverse system design. Typically, a desired output (usually a gain profile, a noise profile, a transfer function or a similar continuous function) is given and the goal is to determine the corresponding set of input parameters (usually a set of input voltages, currents, powers and wavelengths). We present a novel method for inverse system design using machine learning and apply it to Raman amplifier design. Inverse system design for Raman amplifiers consists of selecting pump powers and wavelengths that would result in a targeted gain profile. This is a challenging task due to highly–complex interaction between pumps and Raman gain. Using the proposed framework, highly–accurate predictions of the pumping setup for arbitrary Raman gain profiles are demonstrated numerically in C and C+L–band, as well as experimentally in C band, for the first time. A low mean (0.46 and 0.35 dB) and standard deviation (0.20 and 0.17 dB) of the maximum error are obtained for numerical (C+L–band) and experimental (C–band) results, respectively, when employing 4 pumps and 100 km span length. The presented framework is general and can be applied to other inverse problems in optical communication and photonics in general.**

***Index Terms*—optical communication, optical amplification, machine learning, inverse system design, optimization**

I. Introduction

Determining a set of input parameters that would result in a targeted output function is a problem of general relevance to many areas in photonics ranging from optical components and communication systems design to network optimization [1]–[4]. Specific example applications are:

- in programmable photonics, a desired transfer function is given, and a set of corresponding control voltages for the optical phase shifters need to be selected [2], [3].
- for receiver testing in high–speed optical communication systems, distortion properties of a stress–test signal are given and the resulting signal–generation parameters need to be determined [5].
- in the case of photonic crystal fibres design, a specific dispersion profile is given and the corresponding fibre geometry needs to be found [6].

In this paper, we address a further application: the inverse system design for Raman amplifiers (RA)s. This problem is becoming increasingly important and the reason is that the next–generation of optical communication systems are envisioned to operate in O, E, S, C and L band [7]. Designing an optical amplification scheme that would cover all five bands is a challenging task as the requirements on the gain spectra profile and noise figure may be quite stringent. Compared to the well-established Erbium Doped Fibre Amplifiers (EDFA)s, RAs offer low noise properties due to distributed amplification, and gain availability across a broad range of wavelengths, when operated in multi-pump configurations. Moreover, RAs offer a great flexibility in the gain profile design making them an attractive solution for future ultra–wideband optical networks.

The main challenge with Raman amplifier design is on the selection of pump powers and wavelengths that would result in a specific gain profile. Ideally speaking, being able to realize an arbitrary gain profile may be useful in many situations especially when considering autonomous optical networks [4].

Designing a specific gain profile is in general a complex optimization procedure that requires the solution of a system of nonlinear differential equations describing the evolution of pumps and channels along the fiber. Several solutions to this problem have been reported in the literature [8]–[11] and references therein. So far, none of the solutions have been demonstrated for an arbitrary Raman gain profile design.

The general trend of the solutions presented in [8], [9]

Manuscript received xx, xx; revised xx, 200x.

D. Zibar, A. M. Rosa Brusin and U. C. de Moura have contributed equally to the presented work. This work is supported by the European Research Council through the ERC-CoG FRECOM project (grant agreement no. 771878) and by the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 754462.

D. Zibar, U. C. de Moura and F. Da Ros are with DTU Fotonik, Department of Photonics Engineering, Technical University of Denmark, Kgs. Lyngby, Denmark, (email: dazi@fotonik.dtu.dk)

A. M. Rosa Brusin, V. Curri and A. Carena are with Dipartimento di Elettronica e Telecomunicazioni, Politecnico di Torino, Torino, Italy

is to use genetic algorithms in combination with integration of propagation equations describing multi-pump and optical Wavelength Division Multiplexed (WDM) channels interaction. This approach is cumbersome, highly time consuming and in some cases the solution may not converge.

A recently proposed solution in [10] employs a neural network to learn the mapping between pump powers and wavelengths and the gain profile (forward model). The neural network is then used in combination with genetic algorithms to perform the optimization. This approach avoids solving the differential equation within the optimization routine. The authors in [10], demonstrate their optimization method for the design of flat gain profile only. Most importantly, the considered fiber–span length is 25 km, where in practical applications, the span length is typically in the vicinity of 100 km. Finally, the proposed method needs to be run every time a new gain profile is needed which induces non–negligible latency. Genetic algorithms are a powerful tool but they do rely on random optimization which has non-negligible convergence time especially if initial conditions are not properly set. Finally, the structure and the parameters of the genetic algorithms need to be set according to the problem. This can be very challenging every time a new Raman gain profile needs to be designed.

An interesting approach was also presented in [11], where a neural network is first used to learn the forward model and then the gradient descent is employed in combination with the forward model to only optimize pump powers. This approach has been demonstrated, in simulations, for flat gain designs and fiber–span length of up to 5 dB and 50 km, respectively. The gradient descent optimization is time–consuming and prone to local minima if the initial conditions are not in the vicinity of the solutions [12]. Therefore, an issue with the method presented in [11] is that a set of initial conditions close to the final solution are needed. This requires multi–dimensional exhaustive search which is highly complex and time consuming. Most importantly, the aforementioned search is not very practical every time pump allocations are needed for a new gain profile.

For the next generation of optical communication systems, fast routing, deployment and optimization of data traffic will be highly demanded [4], [13]. Network automatization at low-latency will be highly desired in the path toward autonomous and self-adaptive optical networks [4], [13]. Therefore, ultra-fast methods for selecting pump powers and wavelengths targeting a specific gain profile are essential.

In this paper, we present a novel, machine learning based approach, for inverse system design and apply it to the Raman amplifier design. A multi-layer neural network is employed to learn the mapping between the gain profile and pump powers and wavelengths. Once the model has been learned, pump powers and wavelengths can be predicted by a simple forward propagation through the multi–layer neural network. To address the problem of multi–layer neural network initialization, we employ model averaging. We run a number of parallel neural networks each initialized with a different seed, drawn from a Gaussian distribution. For the final result, we then take the average of all employed neural networks. The variance of the Gaussian distribution is obtained through cross-validation. This approach will make neural network performance less affected by the weight initialization.

The proposed solution in this paper offers a high-degree of flexibility compared to state-of-the-art methods where for each gain profile a complex optimization problem needs to be solved from scratch. The proposed approach has thus high-accuracy, low complexity and it is ultra-fast as the integration of propagation equation is completely avoided. This makes it highly attractive for application in network control-plane where almost real–time adjustments can be required and a full featured optimization process cannot take place.

To improve the accuracy of the aforementioned method, we propose a fine–adjustment of the pump powers and wavelengths. This is achieved by employing a gradient descent algorithm in combination with a second multi–layer neural network, that represents the forward mapping between the pump powers and wavelengths and the Raman gain profile.

The proof-of-principle results, we presented in [14], mainly focused on numerical simulations and C–band only. This paper includes significant extensions in terms of the method itself, as well as numerical (C+L–band) and experimental results (C–band).

Recently, a machine learning approach for the design of flat gain hybrid Raman–EDFA amplifier has been demonstrated for the C+L–band [15]. The proposed approach in [15] is able to learn the relationship between the amplifier output power, gain tilt and the pump powers. In contrary, the proposed framework, in this paper, learns the relationship between the Raman gain profile and pump powers and wavelengths, which is more general and challenging. We are also employing deep random projection method for learning the parameters of the multi–layer neural network, as well as neural network driven gradient descent for fine–adjustments. Indeed, our system resembles auto-encoders and it is significantly different in terms of learning algorithms and architecture compared to the work presented in [15]. In the end,

our framework is able to provide highly–accurate design of any arbitrary gain profile. Particularly, in this paper, we present for the first time numerical and experimental demonstration of such capabilities for Raman amplifiers.

The reminder of the paper is organized as follows. In section II, a framework for the machine learning based inverse system design is described. The general idea and the core algorithms are presented. In section III, the machine learning framework is demonstrated, using numerical simulations for C and C+L–band, for the design of arbitrary, flat and tilted Raman gain profiles. In section IV, experimental set–up used for the validation is presented. The numerical and experimental results are evaluated in terms of the maximum gain error and the root mean square error between the targeted and the predicted gain profiles. In section V, a brief discussion on the speed and complexity in relation to other methods is presented. The conclusions and the outlook of the proposed method are presented in section VI.

II. Machine learning framework

In the inverse system design, we are given a target output $\mathbf{Y}^{targ}$ and the objective is to determine the corresponding input $\mathbf{X}^{targ}$. The forward mapping is denoted by $\mathbf{Y} = f(\mathbf{X})$ and, to determine $\mathbf{X}^{targ}$, the inverse (backward) mapping function $f^{-1}(\cdot)$ needs to be determined. In many cases of interest, the forward mapping function $f(\cdot)$ is highly complex and described by a set of nonlinear differential or integral equations that must be solved numerically. Moreover, for some system, $f(\cdot)$, may even be unknown. Therefore, it is very challenging, and in some cases impossible, to obtain expression for the gradient and apply standard least-squares optimization techniques to find $\mathbf{X}^{targ}$ [16].

Throughout the entire paper both input and output parameters are considered in their sampled version: it is assumed that they are vectors, $\mathbf{X} = [x_1, ..., x_M]^T$ and $\mathbf{Y} = [y_1, ..., y_N]^T$, where T denotes transpose operator, and M and N are the length of the input and the output vectors, respectively.

We propose a machine learning based architecture, shown in Fig. 1, to address the inverse system design and determine $\mathbf{X}^{targ}$. A multi-layer neural network (ML–NN) is first employed to learn the inverse mapping $f^{-1}(\cdot)$. We call this ML-NN, backward network, and denote it as $NN_{bw}(\cdot)$. As the ML–NN is a universal function approximator, the proposed approach can theoretically learn the inverse mapping with relatively high–accuracy, provided that the forward mapping is unique [12]. The ML–NN relies on supervised learning and therefore, the training data–set: $\mathcal{D}_{bw}^{K\times(N+M)} = \{\mathbf{Y}_k^T, \mathbf{X}_k^T | k = 1, ..., K\}$, where K is the size of the data–set, needs to be generated. To generate the training data–set, we do not need to know the functional expression for the forward model, $f(\cdot)$, as we can excite the system with a set of inputs and record the desired outputs. This is exactly how a training data–set is gathered for the practical implementation of the proposed framework which is demonstrated in section IV.

However, it may be convenient to know the functional description of the forward model, $f(\cdot)$. This will allow for numerical simulations to be performed which can give an insight into a better understanding on the performance and the limitations of the framework. These insights can then be used for designing the experiments and the subsequent data collection. Finally, the numerical results may also be helpful in providing the interpretation of the experimental data.

Presenting $\mathbf{Y}_k$ and $\mathbf{X}_k$, from the training data–set, as the input and the output to the ML-NN, a set of weights $\mathbf{W}_{bw} = [\mathbf{W}_{bw}^{(1)}, ..., \mathbf{W}_{bw}^{(L_{bw})}]$ are determined, where L_{bw} is the number of the ML-NN layers (Fig. 1). The ML–NN is then said to be trained as it has learned the mapping between $\mathbf{Y}_k$ and $\mathbf{X}_k$. Even though ML–NN is a universal function approximators, the ML–NN will not provide an exact replica of mapping function $f(\cdot)$ or $f^{-1}(\cdot)$ but an approximation to it. In Appendix A, this is explained in more details.

Once the training of the ML–NN has been completed, the weights $\mathbf{W}_{bw}$ are fixed. If we are then given a target output $\mathbf{Y}^{targ}$, the corresponding input $\mathbf{X}^{targ}$ is computed by the backward neural network: $\mathbf{X}^{targ} = NN_{bw}(\mathbf{Y}^{targ})$. This approach offers ultra–fast and low–complexity computation of $\mathbf{X}^{targ}$ as the evaluation of the backward neural network only performs matrix multiplication on the inputs and the corresponding forward propagation though the network layers. To improve the accuracy of the inverse design achieved with the $NN_{bw}(\cdot)$, we propose a second step of fine–optimization by applying the gradient descent algorithm to fine–adjust $\mathbf{X}^{targ}$:

$$\mathbf{X}^{targ}(i+1) = \mathbf{X}^{targ}(i) - \eta \nabla e(\mathbf{X}_i^{targ}) \qquad (1)$$

where i is the iteration number and $\eta > 0$ is the learning rate. As $\mathbf{X}^{targ}$ should already be in vicinity of the final solution, the gradient decent does not need many iterations to converge. The error $e(\mathbf{X}_i^{targ})$ is defined as the mean square error (MSE) between the targeted output $\mathbf{Y}^{targ}$ and the output of the forward model $f(\mathbf{X}_i^{targ})$. However, this approach can be cumbersome and time–consuming as the forward model $f(\cdot)$ needs to be run at every iteration. Most importantly, in many cases it is not

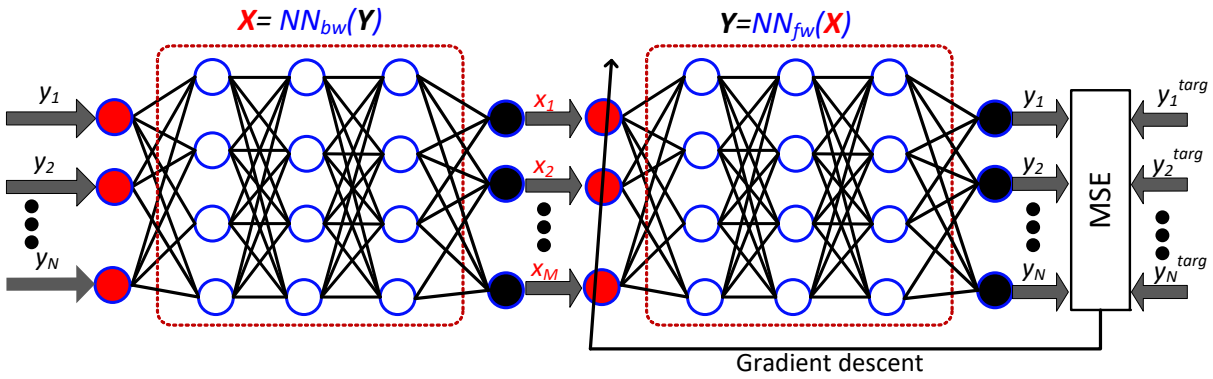

Fig. 1. Illustration of the machine learning based approach for the inverse system design.

possible or it is highly complex to compute the gradient if $f(\cdot)$ is directly used.

Instead of running the forward model $f(\cdot)$ within each iteration of the gradient descent, we propose to use a second ML–NN, denoted by $NN_{fw}(\cdot)$ trained to learn the forward mapping $f(\cdot)$ between the input $\mathbf{X}$ and the output $\mathbf{Y}$ [17], as shown in Fig. 1. This $NN_{fw}(\cdot)$ is completely independent from the $NN_{bw}(\cdot)$. The data–set for training it $\mathcal{D}_{bw}^{K\times(N+M)}$ can be reused by swapping the input with the output: $\mathcal{D}_{fw}^{K\times(M+N)} = \{\mathbf{X}_k^T, \mathbf{Y}_k^T | k = 1, ..., K\}$. The error $e(\mathbf{X}_i^{targ})$ is then redefined as the MSE between the targeted output $\mathbf{Y}^{targ}$ and the output of the forward multi–layer neural network $NN_{fw}(\cdot)$. This approach allows low-complexity and fast computation as the gradient can be calculated analytically, i.e. $NN_{fw}(\cdot)$ is fully described by weight matrices: $\mathbf{W}_{fw} = [\mathbf{W}_{fw}^{(1)}, ..., \mathbf{W}_{fw}^{(L_{fw})}]$, where L_{fw} is the number of the neural network layers.

Once we have computed $\mathbf{X}^{targ}$, we need to check if the results are satisfactory. Therefore, $f(\mathbf{X}^{targ})$ needs to be computed and compared to the desired output. The entire procedure for the proposed inverse system learning is summarized in Algorithm 1.

The core of the method shown in Fig. 1 and outlined in Algorithm 1 is the estimation of the weight matrices $\mathbf{W}_{bw} = [\mathbf{W}_{bw}^{(1)}, ..., \mathbf{W}_{bw}^{(L_{bw})}]$ and $\mathbf{W}_{fw} = [\mathbf{W}_{fw}^{(1)}, ..., \mathbf{W}_{fw}^{(L_{fw})}]$ associated with the backward and the forward neural networks, respectively. The ability to estimate the weight matrices and learn the mappings is highly dependent on the quality of the training data–set. If the training data–set does not contain enough information and the generated data severely violates Hadamard conditions, it is very hard to learn the mapping and thereby perform accurate predictions [12].

A procedure for training data–set generation is shown in Algorithm 2 where $\mathcal{U}$ denotes uniform distribution. It is important that the training data–set contains the outer and mid–points of each dimension of the input vector $\mathbf{X} = [x_1, ..., x_M]^T$ to obtain good generalization properties. This is illustrated in Fig. 2 by red circles. The reason is that during the training process, the algorithm for the ML-NN weight optimization is trying to find a set of weight matrices that minimize the mean square error of the training data and the ML–NN behaves as an interpolator. If the ML-NN is not trained around certain points, it will not be able to generalize (predict) well around those points once presented with the test data. As a general rule, the ML–NN is not good at extrapolating information outside the training range, so we must ensure that outer points are always present in the training data–set. Finally, we need to make a complete distinction between the training and test data–sets. The test data–set is generated by making a new run of Algorithm 2 without adding outer and mid–points.

Algorithm 1 Inverse system design

Run forward model K times $\mathbf{Y} = f(\mathbf{X})$ to generate data–set: $\mathcal{D}_{bw}^{K\times(N+M)} = \{\mathbf{Y}_k^T, \mathbf{X}_k^T | k = 1, ..., K\}$
Train NN_{bw} to learn $\mathbf{X} = f^{-1}(\mathbf{Y})$
Compute the input: $\mathbf{X}^{targ} = NN_{bw}(\mathbf{Y}^{targ})$
Train NN_{fw} to learn $\mathbf{Y} = f(\mathbf{X})$
Evaluate the error: $e = Err(\mathbf{Y}^{targ}, f(\mathbf{X}^{targ}))$
Err: max error or MSE, δ: error tolerance
if $e > \delta$ **then**
  Initialize gradient descent with $\mathbf{X}^{targ}$
  Define error: $E' = MSE(\mathbf{Y}^{targ} - NN_{fw}(\mathbf{X}^{targ}))$
  Run gradient descent: $\mathbf{X}_{(i+1)} = \mathbf{X}_i - \eta \nabla E'(\mathbf{X}_i)$
end if

Algorithm 2 Training data–set generation

Specify range of $\mathbf{X} = [x_1, ..., x_M]$
for k=1:K **do**
$x_1^k \sim \mathcal{U}[x_1^{min}; x_1^{max}]$
$\vdots$
$x_M^k \sim \mathcal{U}[x_M^{min}; x_M^{max}]$
Run the forward mapping $f(\cdot)$ to get $\mathbf{Y}$:
$\{\mathbf{Y}^k = f([x_1^k, ..., x_M^k])\}$
Data–set generation:
$\mathbf{D}_{bw}^k = \{y_1^k, ..., y_N^k, x_1^k, ..., x_M^k\}$
$\mathbf{D}_{fw}^k = \{x_1^k, ..., x_M^k, y_1^k, ..., y_N^k, \}$
end for
Add outer and mid–points, $\mathbf{C}$, to the data-set:
$\mathbf{D}_{bw} = [\mathbf{D}_{bw}^K; \mathbf{C}_{bw}]$
$\mathbf{D}_{fw} = [\mathbf{D}_{fw}^K; \mathbf{C}_{fw}]$
Randomly shuffle samples within $\mathbf{D}_{bw}$ and $\mathbf{D}_{fw}$

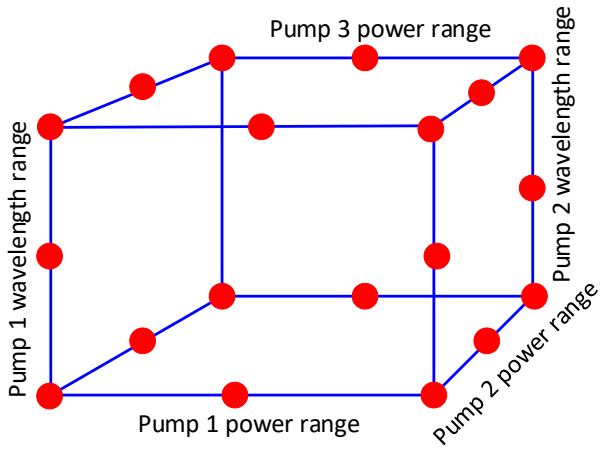

Fig. 2. Hypercube illustrating the outer and mid data points (red circles) that must be included in the training data–set.

Once a procedure for generating training data–set has been established, we can proceed with ML-NN weight matrices estimation. Some of the most popular algorithms for estimating the weight matrices are: gradient descent, Newton's method, conjugate gradient, quasi Newton's method, Levenberg-Marquadt, (LM), and random projection method, (RPS).[1] [12], [18]. Typically, an estimation algorithm that results in the best generalization properties should to be chosen. However, the task of choosing the most appropriate algorithm will be very much dependent on the problem we are trying to solve, as we cannot assume that one solution fits all. Also, a distinction needs to be made if the forward or inverse mapping is to be learned. Optimization algorithm that provides the best generalization properties for learning the forward mapping may be sub–optimal for learning the inverse mapping.

An approach to improve the generalization properties of the ML–NN employed for forward and inverse system learning is to employ model combination [12], [18]. In this paper, we employ model combination where we train P different ML–NNs and then make predictions using the average of the predictions made by each model – *model averaging*. We do not generate P data sets as the method of sampling and replacement is used to obtain P different data sets.

In general, training a neural networks is time consuming, but this is not an issue because it can performed offline as it is required only once in the installation phase as a part of calibration stage. Once the training has been completed, and the mapping has been learned, neural networks can then be operated in real–time to give ultra–fast predictions. In the case of optical communication, the fiber is a fairly static medium, and it will not be necessary to retrain the models. Moreover, for the training of $NN_{bw}(\cdot)$, we can employ the random projection method where the training is ultra–fast as only one matrix inversion needs to be executed.

Adding a fine–optimization phase, if needed, is more time consuming as it employs gradient descent. However, the initial conditions are an output form the $NN_{bw}(\cdot)$ and are very close to the final solution. In other words, the fine–optimization starts not so far away from the final solution and it usually converges after a small number of iterations. In any case, various methods from machine learning community such as gradient descent with momentum, or with Nesterov momentum may be employed to accelerate the fine–optimization [12]. This is a highly relevant topic for future research.

III. Raman amplifier design–numerical results

In this section, the machine learning framework presented in section II, is numerically investigated for the design of Raman amplifiers in C and C+L–band. In Fig. 3, the considered backward pumped Raman amplification setup is shown. The backward pumping is more practical approach for Raman amplification, compared to the forward and the bidirectional pumping, and thereby the most relevant to study at this stage. The proposed framework can also be applied to forward or mixed forward/backward pumping schemes. It is just a matter of creating the data–set in the condition of interest and, consequently, to properly dimension the neural networks.

[1]Random projection method is also known as extreme learning. In our view, random projection method is more descriptive of the method.

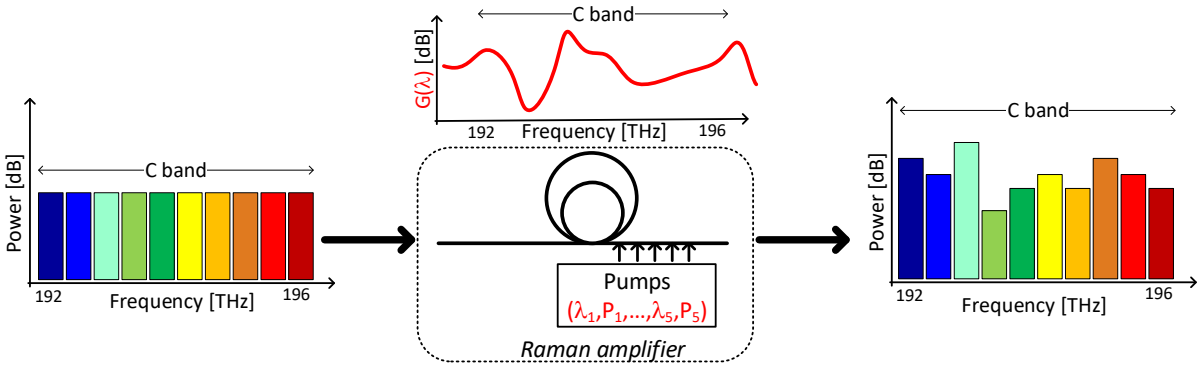

Fig. 3. A single span Raman amplifier employing backward pumping with up to five pumps.

For future work, we plan to investigate various pumping schemes.

The considered single span, shown in Fig. 3, can be a part of a multi-span system and also Raman amplification can take place in conjunction with EDFA. We analyze the gain $G(\lambda)$ over the entire C–band (4 THz band between 192 and 196 THz) as well as C+L–band (11 THz band between 185 THz and 196 THz). We consider pumping schemes with 2, 3, 4 and 5 pumps. The pump powers, $[P_1, ..., P_5]$, and wavelength ranges, $[\lambda_1, ..., \lambda_5]$, are specified in Table I.

The Raman amplifier is described by a set of non–linear ordinary differential equations [19]. Their expressions are shown in (2) and (3):

$$\frac{dP_{s,i}}{dz} = -\alpha_s P_{s,i} + C_R(\lambda_{s,i}, \lambda_{p,j})[P^+_{p,j} + P^-_{p,j}]P_{s,i} \quad (2)$$

$$\pm\frac{dP^{\pm}_{p,j}}{dz} = -\alpha_p P^{\pm}_{p,j} - \Big(\frac{\lambda_{s,i}}{\lambda_{p,j}}\Big) C_R(\lambda_{s,i}, \lambda_{p,j}) P_{s,i} P^{\pm}_{p,j} \quad (3)$$

where P_s is the average signal power, P_p^+ and P_p^- respectively the co- and counter-propagating pump powers (we only considered counter-propagating pumps), α_s and α_p respectively the signal and pumps attenuation coefficients measured in km^{-1}, λ_s and λ_p are respectively the signal and the pump wavelengths and C_R is the Raman gain efficiency, measured in (W$\cdot$ km)$^{-1}$, which depends on the Raman gain coefficient g_R and on the effective area A_{eff}. In these equations, i identifies the signal channel for $i = \{1, \ldots, n_{ch}\}$ being $n_{ch} = 40$ in the C–band and $n_{ch} = 110$ in C+L–band, whereas j identifies the pump, for $j = \{1, \ldots, n_{pumps}\}$ being $n_{pumps} = 2, \ldots, 5$.

The differential equation (2) and (3) govern the evolution of the power along the fiber at different wavelengths and can be used to evaluate gains, $\mathbf{Y} = [G(\lambda_1), ..., G(\lambda_N)]$, as a function of input parameters which are pump powers and wavelengths, i.e. $\mathbf{X} = [P_1, ..., P_M, \lambda^p_1, ..., \lambda^p_M]$ [2]. The objective is to determine a configuration of the pump powers and wavelengths, $\mathbf{X}^{targ} = [P_1^{targ}, ..., P_M^{targ}, \lambda_1^{targ}, ..., \lambda_M^{targ}]$, that would result in targeted gain profile $\mathbf{Y}^{targ} = [G^{targ}(\lambda_1), ..., G^{targ}(\lambda_N)]$, i.e. tilted gain profile, flat gain profile or an arbitrary gain profile. This is the general problem of gain profile design that derives from the fact that a Raman amplifier is very flexible. For instance, it can be used to compensate for EDFA gain ripples in hybrid amplifiers, wavelength dependent losses of Reconfigurable Optical Add Drop Multiplexers (ROADM)s or in general to fix any unwanted unbalance between channel power.

The training and test data–sets, $\mathcal{D} = \{(G_1^k, ..., G_N^k, P_1^k, ..., P_M^k, \lambda_1^k, ..., \lambda_M^k) | k = 1, ..., K\}$, are obtained by following the procedure specified in Algorithm 2 and using the parameters specified in Table I. A full Raman solver (RS) needs to be run to obtain the training and test data–sets. The RS has pump powers and wavelengths as inputs and the gain profile is obtained at the output. In this study, a comb of ideal Nyquist-WDM channels is selected as an input to the span. The WDM channels are packed at Nyquist limit, (channel spacing equal to symbol-rate), to fully load the entire C and C+L–band. This results in a flat optical spectrum: 125 and 343 channels at 32 Gbaud covering the C and C+L–band, respectively. Per-channel signal power was set to 0 dBm. This results in a total WDM signal power of 21 dBm (C–band) and 25 dBm

[2]To better complain with the physical meaning of input parameters, i.e. pump powers and wavelengths, with respect to Sec.II, we redefine the dimension of $\mathbf{X}$ as 2M.

TABLE I
POWER AND WAVELENGTH RANGES FOR RAMAN PUMPS.

Band	C			C+L
Config./Param.	2 pumps	3 pumps	5 pumps	4 pumps
P_1 [mW]	[0:400]	[0:300]	[0:160]	[0:200]
P_2 [mW]	[0:400]	[0:300]	[0:160]	[0:200]
P_3 [mW]		[0:300]	[0:160]	[0:200]
P_4 [mW]			[0:160]	[0:200]
P_5 [mW]			[0:160]	
λ_1 [nm]	[1414:1449]	[1414:1437.3]	[1414:1428]	[1414:1441.8]
λ_2 [nm]	[1449:1484]	[1437.3:1460.6]	[1428:1442]	[1441.8:1469.5]
λ_3 [nm]		[1460.6:1484]	[1442:1456]	[1469.5:1497.3]
λ_4 [nm]			[1456:1470]	[1497.3:1525]
λ_5 [nm]			[1470:1484]	

(C+L–band).

It should be noted that at the output of the Raman solver the gain is evaluated at 40 points for C–band and in 110 points for C+L–band. This is due to the resolution of the solver and corresponds to 100 GHz channel spacing.

Fiber parameters are listed in the following: span length $L_{span} = 100$ km, attenuation $\alpha_S = 0.2$ dB/km for optical data signals and $\alpha_P = 0.25$ dB/km for pumps, effective area $A_{eff} = 80$ μm^2, non-linear coefficient γ=1.26 1/W/km, chromatic dispersion $D = 16.7$ ps/nm/km, and Raman coefficient g_R =0.4125 1/W/km. Standard silica Raman efficiency profile has been assumed.

To determine the size of the training data–set, we have performed investigation by increasing the set size from 2000 to 9000 points. We found that for the training data-set of 4000 points, a good compromise in terms of computational time required for the training stage and the accuracy is achieved. Indeed, for increasing the training data-set size beyond 4000, the accuracy did not improve further. While, reducing it, allowed to speed up training time, the accuracy was lower. Since we have shown in [14] and in Appendix B, that the dependence of Raman gain design on the per-channel input power is almost negligible, the size of the training data-set will not depend on input power if the Raman amplifier does not reach saturation. Correspondingly, the test data–set has up to 5000 points.

A. Arbitrary gain design - C–band

First, we would like to investigate if the backward multi–layer neural network, $NN_{bw}(\cdot)$, is able to learn the inverse mapping, i.e. mapping between the Raman gain and pump powers and wavelengths. To evaluate the optimum topology and weight estimation algorithm, the method of early stopping is employed [12]. We found that the ML-NN topology that minimizes the test error for $NN_{bw}(\cdot)$ consists of 7 hidden layers with 800 hidden nodes each with hyperbolic tangent as the activation function. The model combination parameter P was set to 200. We tested different weight estimation algorithms and the best performance was achieved using the random projection method (RPM). This implies that all layers except the last one are initialized by random weights drawn from a Gaussian distribution with zero mean and variance σ^2_{RPM}. The variance σ^2_{RPM} is a hyper-parameter and is determined using cross–validation. The output of the network is then computed as the solution to the regularized least squares problem. This approach of estimating the ML-NN weights is ultra-fast as no iterations are needed as in the case of gradient based approaches.

For the fine–optimization, we employ $NN_{fw}(\cdot)$, that is trained with Levenberg–Marquardt learning algorithm. It consists of 2 hidden layers each with 10 hidden nodes and symmetric sigmoid as the activation function. By using a simpler topology structure and not considering model combination, it is possible to speed up the fine–optimization step. The parameters of employed ML–NNs, ($NN_{bw}(\cdot)$ and $NN_{fw}(\cdot)$) are shown in Table II.

In Fig. 4, we plot the probability density functions (pdf) for the root-mean-square-error (RMSE) and the maximum error ($Error_{MAX}$) for the cases of 2, 3 and 5 pumps. The results are shown for two cases: when applying only the $NN_{bw}(\cdot)$, legend "Before fine–optimization", and when adding a fine–optimization stage based on the $NN_{fw}(\cdot)$ in combination with a gradient descent algorithm, legend "After fine–optimization". Both RMSE and $Error_{MAX}$ are calculated considering the target gain profile against the one obtained with the RS with pump powers and wavelengths provided by the proposed framework, with or without fine–optimization. Relying on the RS for the final error evaluation, even if the application of the RS is computationally heavy to perform a validation

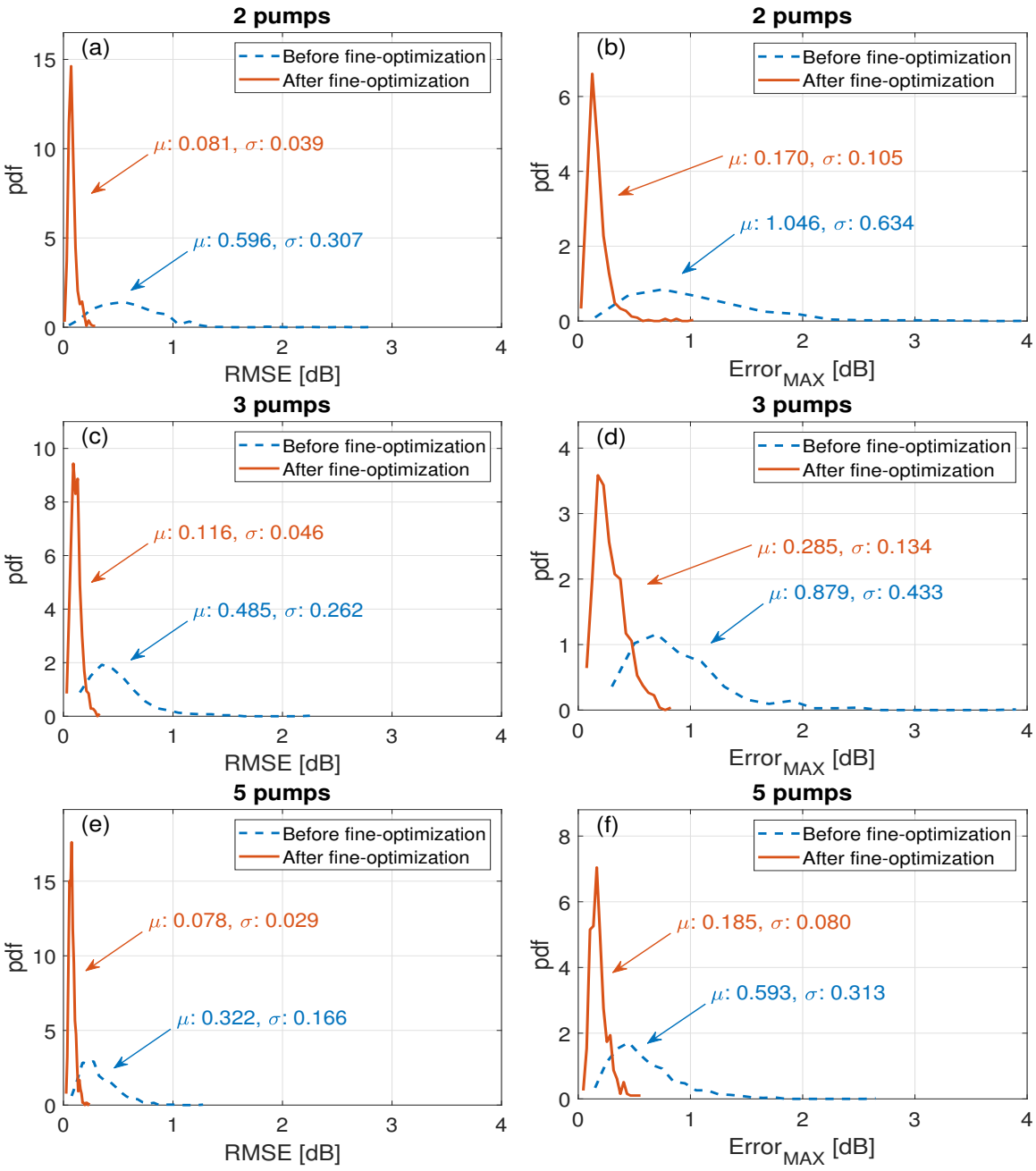

Fig. 4. Simulation: probability density function (pdf) of the RMSE (left column) and Error$_{MAX}$ (right column), with indication of the mean, μ, and the standard deviation, σ, values for 2, 3, and 5 pumps when using the proposed machine learning approach for obtaining the configuration of pump powers and wavelengths given arbitrary Raman gain profiles.

over a large data–set, is the correct approach: an error evaluation based on the $NN_{fw}(\cdot)$ is faster but it is affected by further potential errors that would mask the real RMSE and maximum error evaluation.

The pdfs are computed over a validation data–sets (5000 elements) with arbitrary shaped Raman gain profiles which were generated using Algorithm 2: this data–set is independent with respect to the one used to train backward and forward NNs. During the generation of the test data–sets, we prune the elements by selecting only

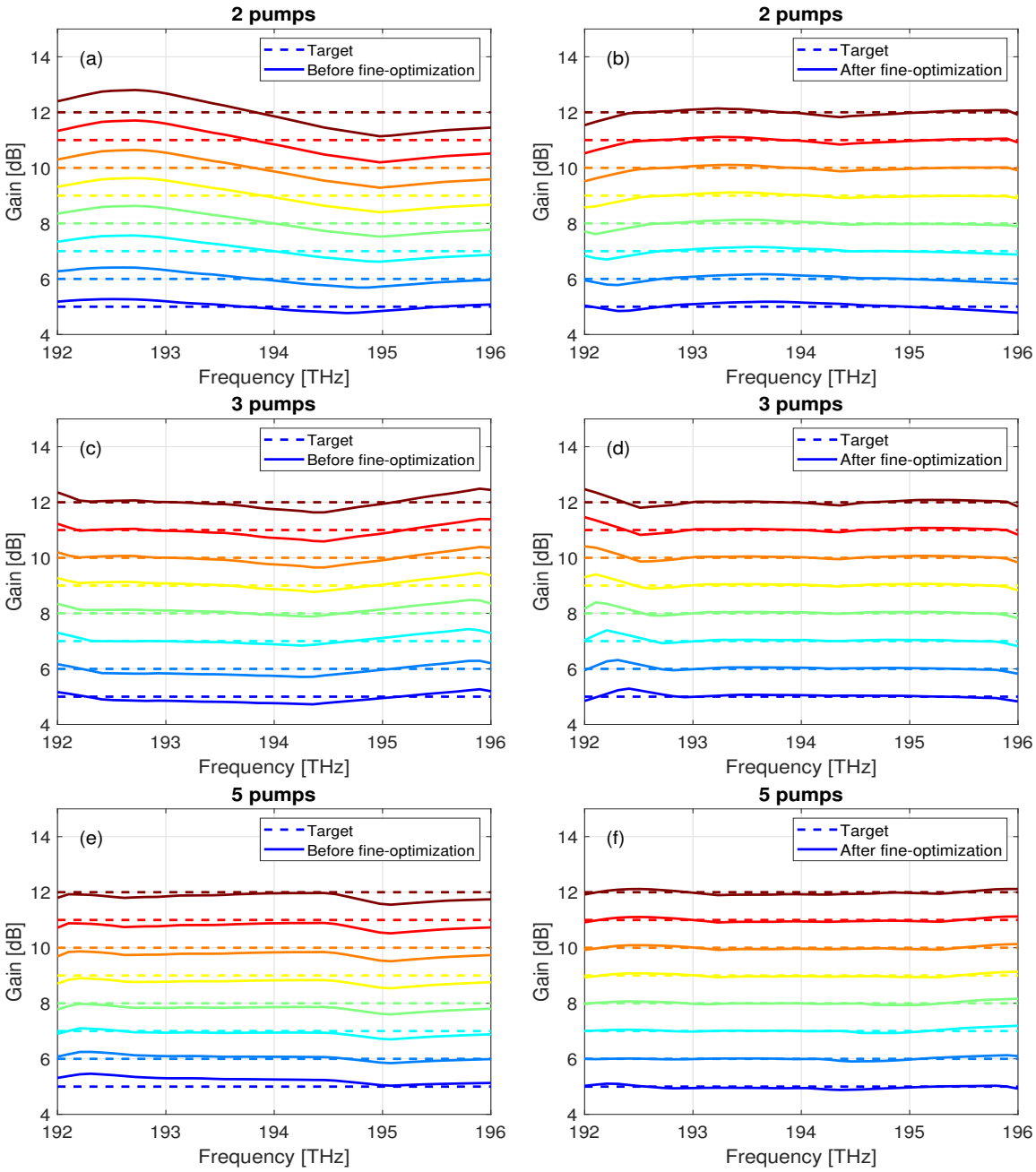

Fig. 5. Comparison between the predicted and the targeted gain profiles as a function of frequency for the flat gain design: (a)-(b) two pumps configuration, (c)-(d) three pumps configuration and (e)-(f) five pumps configuration.

the cases where both the minimum and the maximum values of Raman gain across the whole C–band are inside the range from 4 to 12 dB. This allows to analyze only Raman amplifier working in practical conditions, avoiding cases where the gain is too low, less than 4 dB, and cases outside the moderated pumping regime [20], which corresponds to 12 dB (i.e. 60% of 20 dB span loss), preventing gain saturation effects and loss of efficiency.

Fig. 4 indicates that for all the pumping schemes under analysis, when considering only results before fine–optimization, the mean $\pm$ standard deviation for RMSE

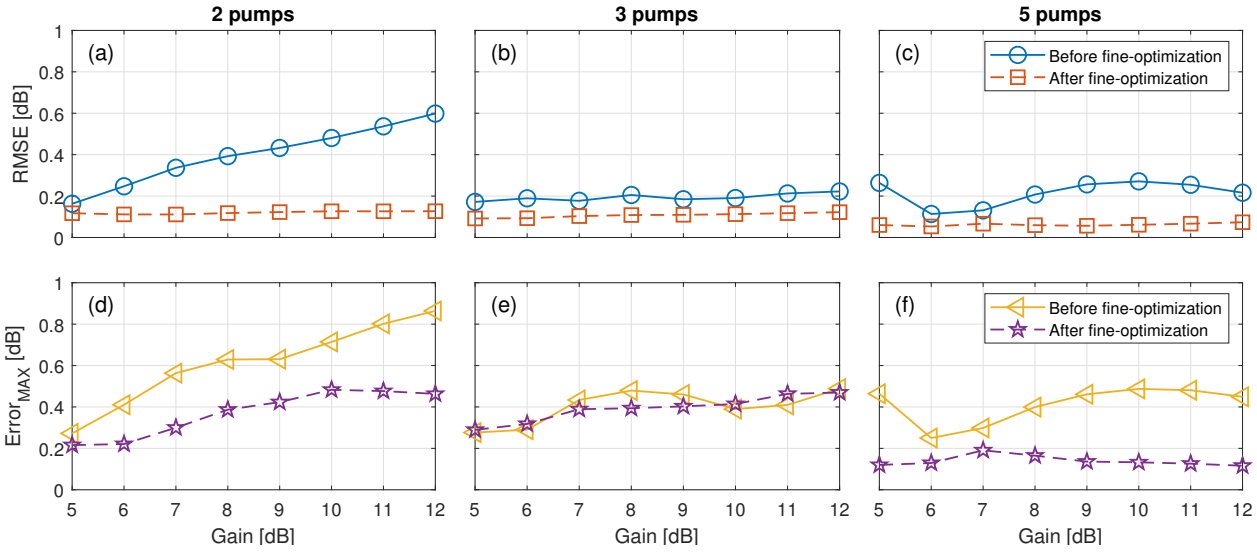

Fig. 6. Flat gain case: RMSE (top row) and Error_{MAX} (bottom row) as a function of gain for: (a) 2, (b) 3 and (c) 5 pumps.

TABLE II
ML-NN TOPOLOGIES AND PARAMETERS.

	$NN_{bw}(\cdot)$			$NN_{fw}(\cdot)$		
	C	C+L	Experiment	C	C+L	Experiment
Training algorithm	RPM	RPM	RPM	LM	LM	LM
# hidden layers	7	4	4	2	2	2
# hidden nodes	800	900	700	10	10	10
Activation function	tanh	logsig	tanh	tansig	logsig	tansig
Model combination parameter (P)	200	200	200	1	1	1

and $Error_{MAX}$ are kept bellow 0.596±0.307 dB and 1.046±0.634 dB, respectively. This is a clear indication that the $NN_{bw}(\cdot)$ itself can learn the inverse mapping with good accuracy. Depending on the application targeted, if the required accuracy is not very tight, the backward neural networks itself can deliver affordable results: especially in the case of 5 pumps both errors are reasonably small over a the whole validation data–set of 5000 cases.

In case the level of desired accuracy is higher, we can proceed employing the $NN_{fw}(\cdot)$, in combination with the gradient descent algorithm, to fine–adjust the values of the pump powers and wavelengths predicted using $NN_{bw}(\cdot)$. Fig. 4 illustrates a significant decrease in both the mean and in the standard deviation values of RMSE and $Error_{MAX}$. For the considered pumping schemes, the mean RMSE and mean $Error_{MAX}$ values do not exceed ∼0.15 dB and ∼0.3 dB, respectively. We can assert that the proposed method for designing Raman amplifiers is highly accurate as it can deliver the required gain profile over a wide band with negligible errors.

Moreover, these results also demonstrate the ability of $NN_{fw}(\cdot)$ to learn the forward mapping, which is essential to perform the fine–optimization in an ultra–fast and computationally efficient way, compared to invoking the full Raman solver at each gradient descent iteration.

B. Flat and tilted gain design - C–band

In this section, we analyze the ability of the multi-layer neural network, alone or in combination with the fine–optimization stage, to predict the configuration of pump powers and wavelengths that would result in a flat and tilted Raman gain profiles. The considered gain levels range from 5 to 12 dB and in the case of tilted gain we consider a slope of 0.25 dB/THz.

In Fig. 5(a),(c) and (e), the predicted and the target flat gain profiles are plotted as a function of the optical frequency when using only the $NN_{bw}(\cdot)$ to obtain the corresponding set of pump powers and wavelengths for 2, 3 and 5 pumps, respectively. Overall, Fig. 5 illustrate qualitatively that the $NN_{bw}(\cdot)$ is able to provide the pump power and wavelength allocations, resulting in a flat gain, with relatively good precision. Finally, it can be observed in Fig. 5 that for some limited number of

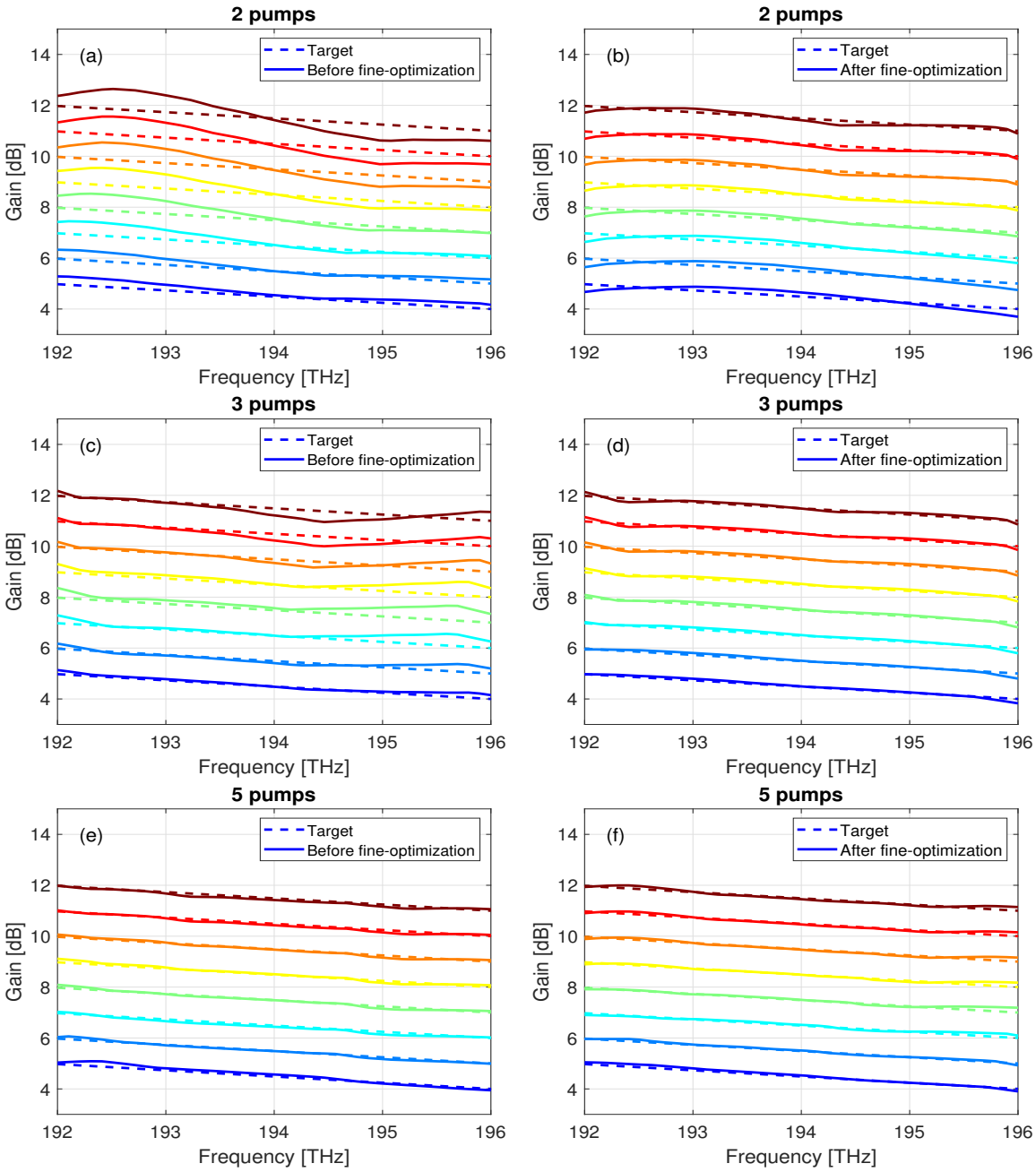

Fig. 7. Comparison between the predicted and the targeted gain profiles as a function of frequency for the tilted gain design: (a)-(b) two pumps configuration, (c)-(d) three pumps configuration and (e)-(f) five pumps configuration.

cases, deviations of the predicted gain profiles at low frequencies are observable. The deviations are always much less than 1 dB, so we can consider them almost irrelevant. Furthermore, it is difficult to draw some general conclusions on the occurrence of the gain deviations. This is because the Raman gain is a nonlinear, (and non–trivial), combination of 2, 3 and 5 separate Raman gain profiles, depending on the number of pumps used and on their separation. What we can observe in Fig. 5 is that for the increasing number of pumps, the difference between the targeted and predicted flat gain profiles decreases and the combined Raman gain becomes more smooth.

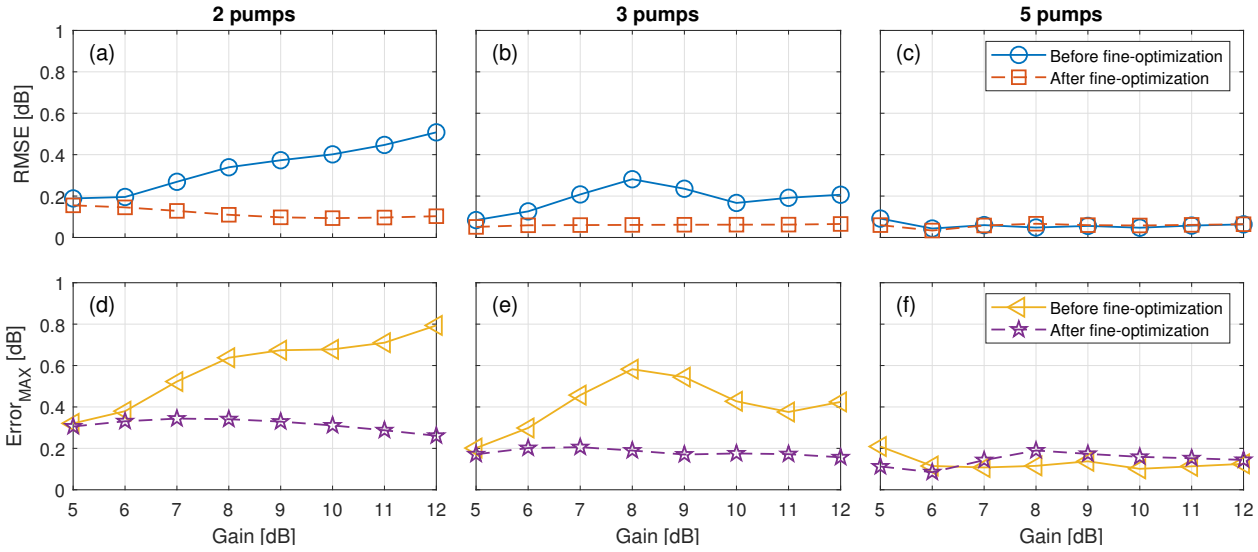

Fig. 8. Tilted gain case: RMSE and Error_{MAX} as a function of gain for: (a) 2 pumps, (b) 3 pumps, (c) 5 pumps.

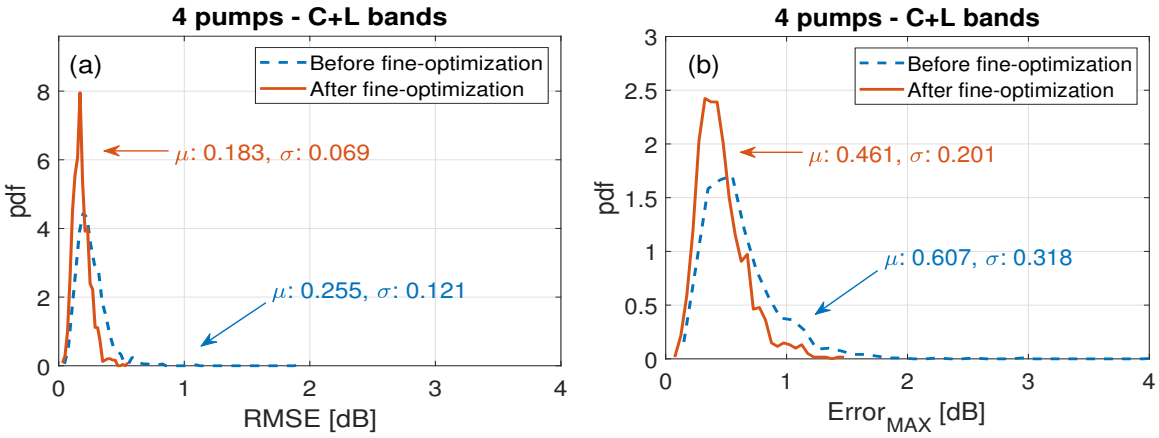

Fig. 9. C+L–band - probability density function (pdf) of the (a) RMSE and (b) $Error_{MAX}$, with indication of the mean, μ, and the standard deviation, σ, values for 4 pumps when using the proposed machine learning approach for obtaining the configuration of pump powers and wavelengths given arbitrary Raman gain profiles. The maximum values of $Error_{MAX}$ is 4.05 dB.

Next, to quantify the accuracy, in Fig. 6 RMSE and Error_{MAX} are plotted as a function of the targeted gain. The highest errors in terms of RMSE and Error_{MAX} occur both for two pump configuration and are ∼0.6 dB and ∼0.8 dB, respectively. The RMSE values for the configuration with three and five pumps are much lower, of the order of ∼0.2 dB. Likewise, the performance in terms of Error_{MAX} does not exceed 0.5 dB for 3 and 5 pumps.

Next, the fine–adjustment of the pump powers and wavelengths is employed and the results are shown in Fig. 5(b),(d) and (f) and Fig. 6. A comparison between Fig. 5(a),(c) and (e) and 5(b),(d) and (f), indicates that a significantly lower difference between the predicted and the targeted flat gain profiles is achievable after applying the fine–adjustments. This is also indicated in Fig. 6 in terms of the RMSE and the Error_{MAX}. The RMSE values for 2, 3 and 5 pumps are approximately the same, namely around 0.2 dB, always lower than before fine–optimization. In particular, the case of two pumps is strongly improved and it reaches same error levels as the other cases. This is an encouraging result and it also has a practical implication for Raman amplifier design showing that only two pumps may be needed to obtain the flat gain profile for the entire C–band. Moreover, the Error_{MAX} for two pumps drops down to a maximum

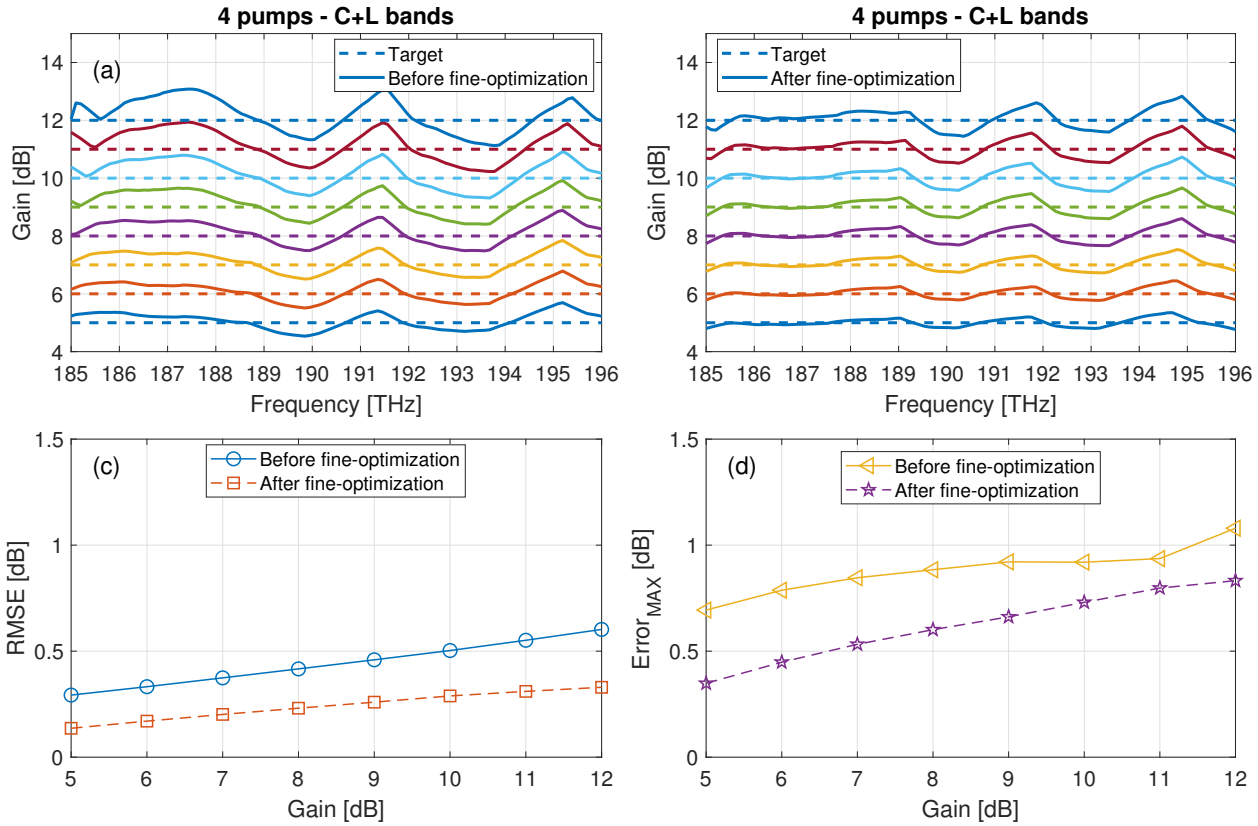

Fig. 10. C+L–band - flat gain case with 4 pumps: comparison between the predicted and the targeted gain profiles as a function of frequency (a) Before fine–optimization and (b) after fine–optimization. (c) RMSE, (d) and Error_{MAX} as a function of gain.

value of ∼0.4 dB which can be an acceptable level. For 3 pumps configuration, the fine–optimization does not bring any significant benefits. Although, the Error_{MAX} improves significantly and stays below 0.2 dB for 5 pumps configuration.

In Fig. 7(a),(c) and (e), the predicted and the target tilted gain profiles are plotted as a function of the optical frequency when using only the $NN_{bw}(\cdot)$ to obtain the corresponding set of pump powers and wavelengths for 2, 3 and 5 pumps, respectively. We consider 1 dB of tilt along the C–band. Fig. 8 shows the corresponding RMSE and Error_{MAX} plotted as a function of the target gain at the lowest frequency.

Fig. 7(a),(c) and (e) illustrate a good agreement between the predicted and targeted gains, especially for 3 and 5 pumps. As for the case of flat gain, the RMSE values become very similar, for 2,3 and 5 pumps configurations, after employing NN_{fw} in combination with the gradient descent algorithm. Likewise, Error_{MAX} are also almost within the same range after fine–adjustment of the pump powers and wavelengths. Finally, it should be noticed that RMSE and Error_{MAX} are very similar before and after fine–adjustments for 5 pumps.

It is observed in Fig. 8(f) that fine–optimization will result in higher error for gains beyond 7 dB. The reason may be that the fine–optimization relies on $NN_{fw}(\cdot)$ which is an approximation to Raman equations. The model mismatch between $NN_{fw}(\cdot)$ and Raman equations, can potentially introduce some errors in the optimization process. This may happen when the solution obtained with the $NN_{bw}(\cdot)$ has already very low errors or has reached theoretically minimum obtainable error. In that case, the fine–optimization can no longer reduce the error but slightly increase it (overfitting).

C. Arbitrary gain design - C+L–band

Here, we investigate the performance of proposed framework to provide pump powers and wavelengths allocations for the design of an arbitrary Raman gain profile for the C+L–band. We consider four pumps with the power and wavelength ranges specified in Table I. The topology and the parameters for the inverse system modelling framework are specified in Table II.

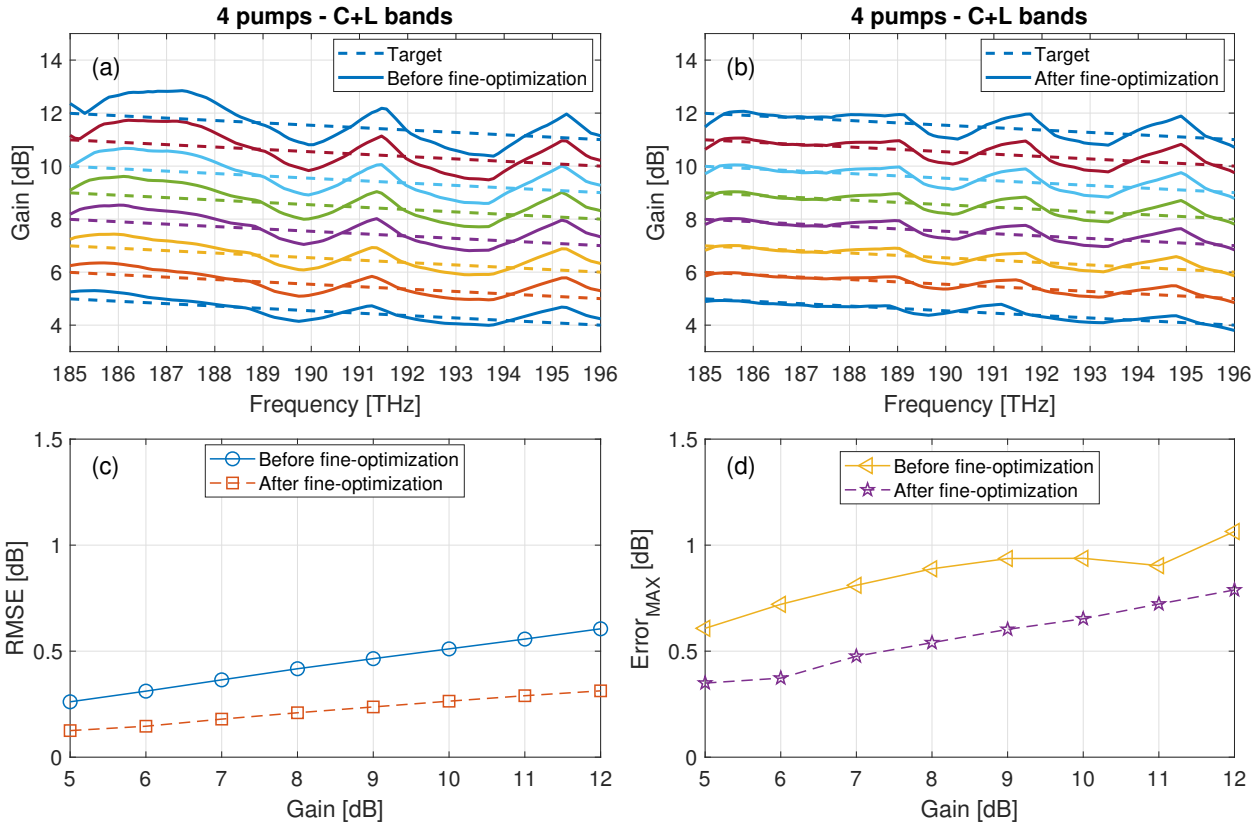

Fig. 11. C+L–band - tilted gain case with 4 pumps: comparison between the predicted and the targeted gain profiles as a function of frequency (a) Before fine–optimization and (b) after fine–optimization. (c) RMSE, (d) and Error_{MAX} as a function of gain.

Fig. 9(a) and (b) show the probability density functions before and after the fine–optimization for the RMSE and Error_{MAX}, respectively. Both RMSE and Error_{MAX} are calculated over the validation data–set with arbitrary shaped Raman gain profiles generated using Algorithm 2. Moreover, we also prune the elements on the validation data–set by selecting only the cases where both the minimum and the maximum values of Raman gain across the whole C+L–band are inside the range from 4 to 12 dB for reasons already explained in Section III-A.

Fig. 9 illustrates that the $NN_{bw}(\cdot)$ can learn the inverse mapping for C+L–band with high accuracy. The corresponding mean and standard deviation of the RMSE are 0.25 dB and 0.12 dB, respectively. For the Error_{max}, the corresponding mean and standard deviation are 0.61 dB and 0.32 dB, respectively. These values are similar to the values obtained for the C–band.

Moreover, after applying the fine–optimization, we observe a decrease in both the mean and standard deviation values for the RMSE and $Error_{MAX}$ to 0.18±0.07 dB and 0.461±0.20 dB, respectively. This clearly demonstrates the ability of the $NN_{fw}(\cdot)$ to learn the forward mapping for the C+L–band and to be employed within the gradient descent.

D. Flat and tilted gain design - C+L–band

As a special case, we investigate the performance of the proposed framework to provide pump power and wavelength allocations for the design of flat and tilted Raman gain profiles for the C+L–band. The gain levels range from 4 to 12 dB. For the tilted gains, a slop of 0.1 dB/THz is considered to provide 1 dB of tilt along the entire bandwidth. The topology and the parameters for the inverse system design framework are specified in Table II.

In Fig. 10(a), the predicted and the target flat gain profiles are plotted as a function of the optical frequency when using only the $NN_{bw}(\cdot)$ to obtain the corresponding set of pump powers and wavelengths. Fig. 10(a) demonstrates that the $NN_{bw}(\cdot)$ is able to provide pump power and wavelength allocations, resulting in a relatively flat gain profiles. The performances regarding

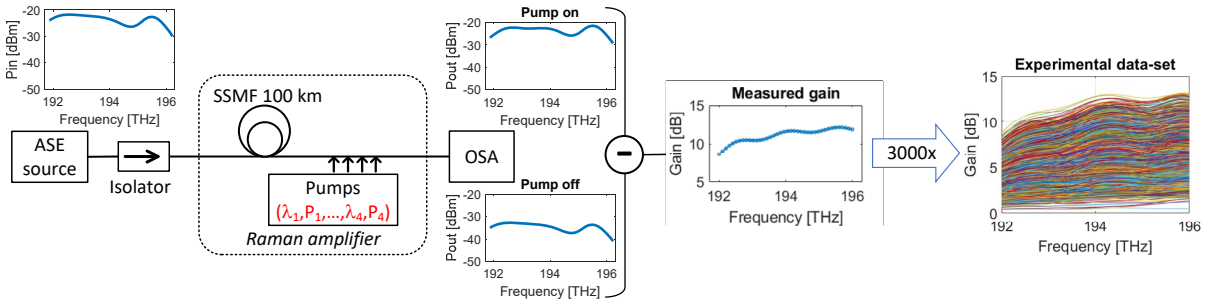

Fig. 12. Experimental setup to capture the data–set for the neural networks training and further validations. Spectra were measured at a resolution of 0.1 nm.

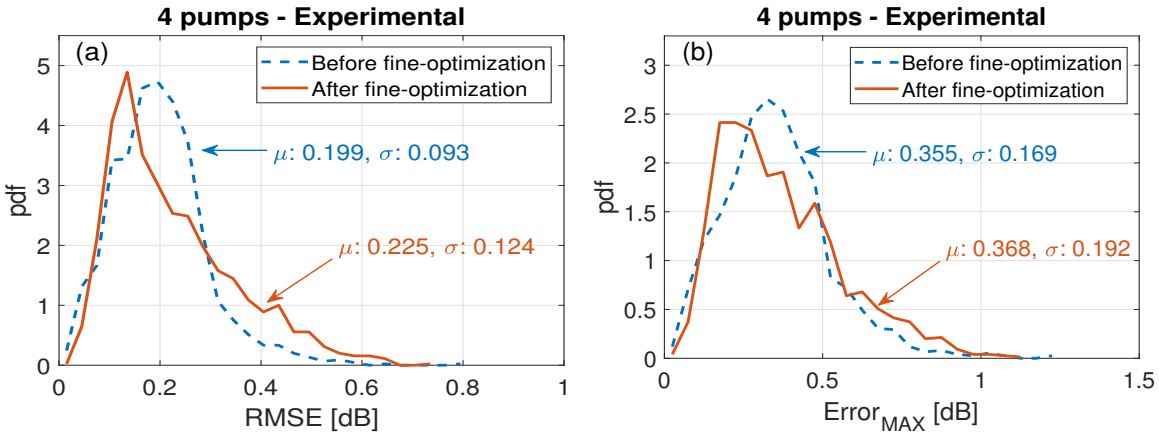

Fig. 13. Experimental validation: probability density function (pdf) of the (a) RMSE and (b) $Error_{MAX}$, with indication of the mean, μ, and the standard deviation, σ, values for 4 pumps when using the proposed machine learning approach for obtaining the configuration of pump powers and wavelengths given arbitrary Raman gain profiles.

RMSE and Error_{MAX} are shown in Fig. 10(c) and (d), with values lower than 0.6 dB and 1.1 dB, respectively.

Fig. 10(b) shows the predicted flat gain profiles after applying the fine–optimization of the pump powers and wavelengths. The results present a better match between the target and predicted gain profiles, especially for low frequencies, when compared to Fig. 10(a). The maximum values for RMSE and Error_{MAX} are kept bellow 0.3 dB and 0.8 dB, as shown in Fig. 10(c) and (d), respectively.

The results for the tilted gains, before and after the fine–optimization, are shown in Fig. 11(a) and (b), respectively. The RMSE and Error_{MAX} curves shown in Fig. 11(c) and (d) indicate that a highly accurate pump power and wavelength allocation is feasible using the proposed framework. The corresponding RMSE and $ERROR_{MAX}$ value are similar to the ones obtain for flat gain profile. The RMSE and $ERROR_{MAX}$ curves present values lower than 0.6 and 1.1 dB, respectively, before the fine–optimization, and lower than 0.4 and 0.8 dB after the fine–optimization.

TABLE III
POWER RANGES AND WAVELENGTHS FOR THE 4 RAMAN PUMPS ON THE EXPERIMENTAL SETUP.

	Pump 1	Pump 2	Pump 3	Pump 4
P [mW]	[0:145]	[0:158.5]	[0:180]	[0:152.5]
λ [nm]	1454.4	1444.8	1434.4	1423.4

IV. Experimental validation

In this section, a proof-of-principle experimental validation of the proposed machine learning based inverse system design framework is presented. The experimental set-up is shown in Fig. 12. The Raman amplifier module under consideration consists of a 100-km long standard single mode fibre (SSMF) span and four backpropagating

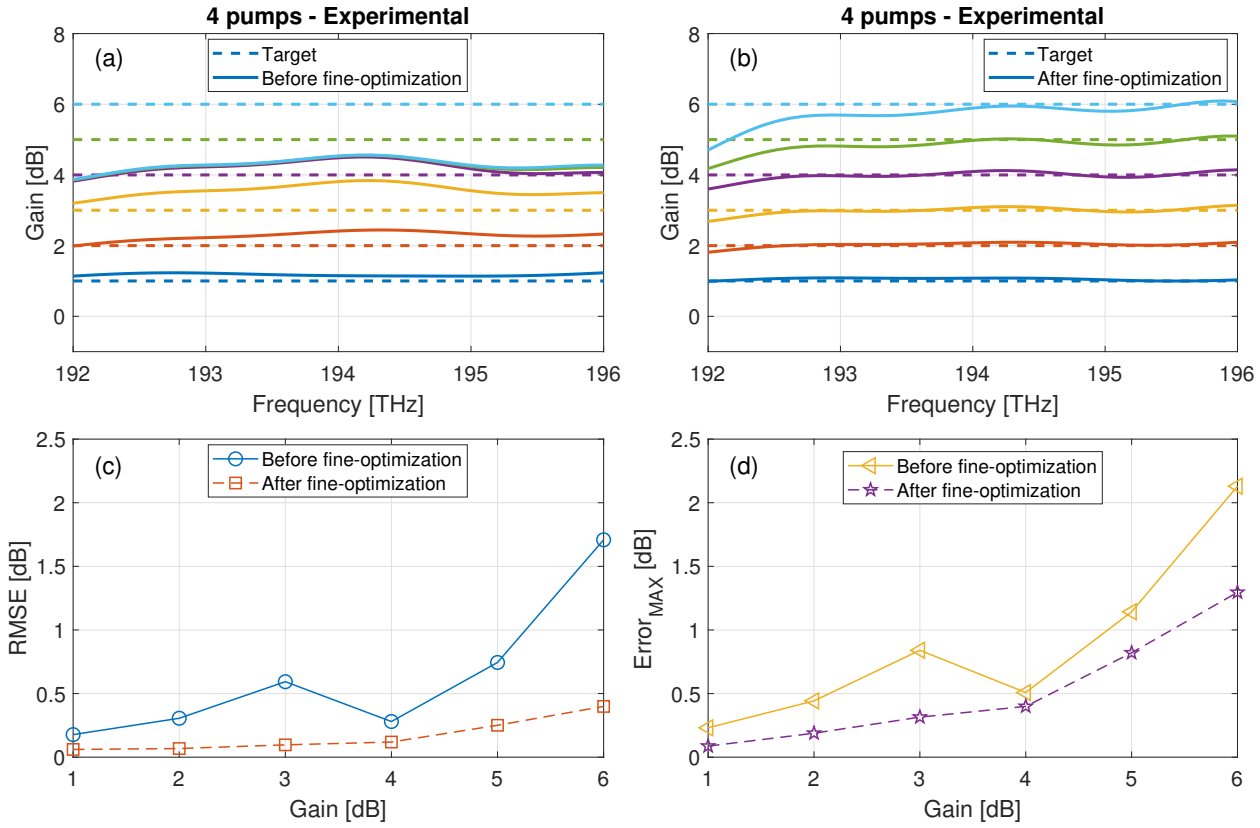

Fig. 14. Experimental validation for the flat gain case with 4 pumps: comparison between the predicted and the targeted gain profiles as a function of frequency (a) Before fine–optimization and (b) after fine–optimization. (c) RMSE, (d) and Error_{MAX} as a function of gain.

pumps. The parameters for the Raman pumps are shown in Table III. The only adjustable degrees of freedom are the pump powers. We therefore investigate the performance of the proposed framework to provide accurate pump power allocations resulting in an arbitrary, flat and tilted gain profiles for the entire C–band.

The experimental set-up, the parameters of the SSMF are the same as for the simulation. The input signal occupies the entire C–band (192-196 THz) and is generated by an amplified spontaneous emission (ASE) source. Compared to the simulations, here, we use an arbitrary input signal power profile. The total input signal power to the span is +2.6 dBm.

The pump powers are controlled by the voltages which are specified using a MATLAB script that follows the procedure described in Algorithm 2. An optical spectrum analyzer (OSA) is used to capture optical power spectra at a 0.1 nm resolution. The Raman gain profiles are obtained by the difference in the recorded optical power spectra when the pump powers are on and off. This is illustrated in Fig. 12. The Raman gain is then discretized to obtain 40 C–band channels. We collect 3000 data points and split it in half for the training and the validation. The 3000 gain profiles are shown on the right side of Fig.12, with gains varying from almost 0 to 13 dB for some channels. We found out that the training data–set of 1500 is enough to learn the mapping. Compared to simulations, where training data–set contained 4000 examples, the wavelengths are fixed, and the dimensionality of the input space to the ML-NN is thereby reduced. Due to the fixed wavelength allocation of the pumps, there is a tendency to provide higher gains for higher frequencies.

A. Arbitrary gain design

Fig.13 shows the pdf curves for the RMSE and the $Error_{MAX}$ computed over the validation data–set. The error is computed in the following way: Using the training data, the mapping between the Raman gain profiles and pump powers is learned using the proposed machine learning framework. Then, given the arbitrary

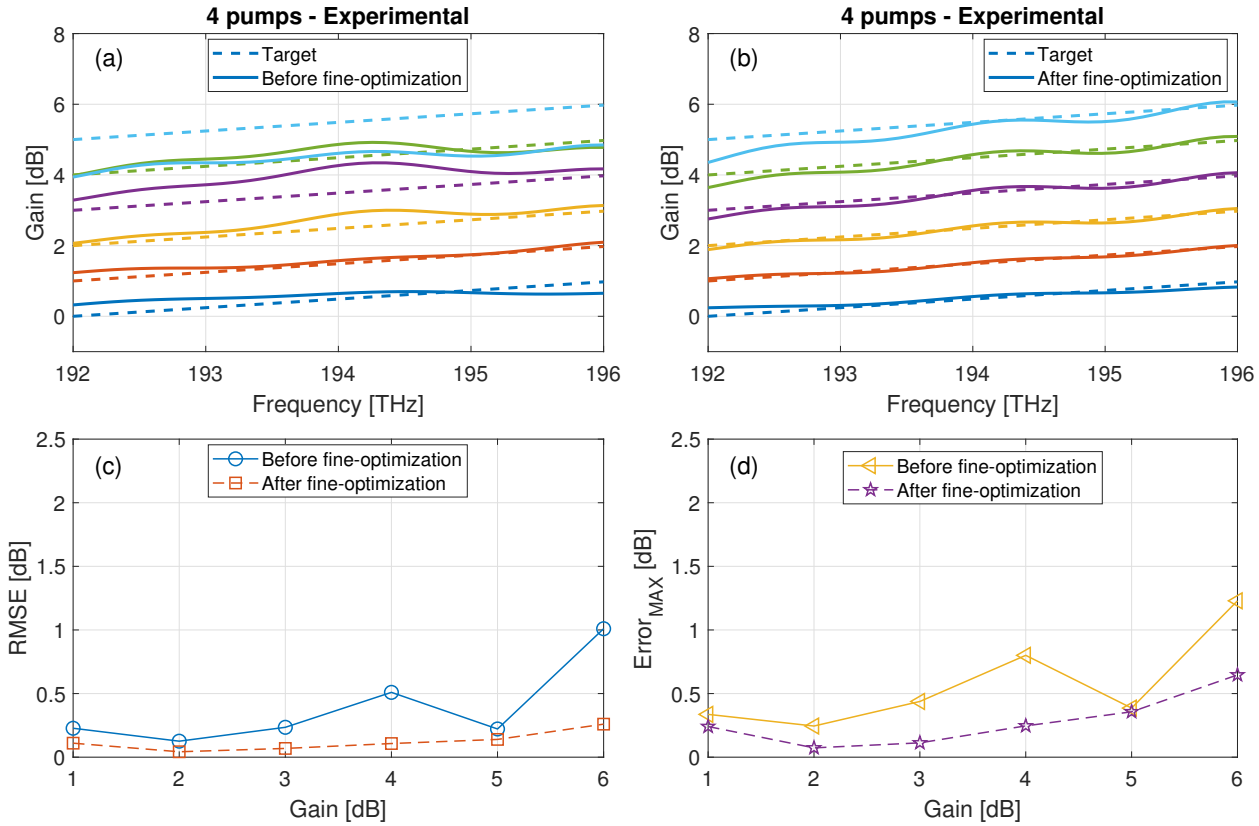

Fig. 15. Experimental validation for the tilted gain case with 4 pumps: comparison between the predicted and the targeted gain profiles as a function of frequency (a) Before fine–optimization and (b) after fine–optimization. (c) RMSE, (d) and Error_{MAX} as a function of gain.

gains, from the validation set, we compute the corresponding pump powers using the learned mapping. This is all performed offline. Next, using the obtained pump powers, we run the experiment and measure the corresponding Raman gain profiles and compare them to the ones in the validation set.

Fig.13 illustrates that highly accurate pump power allocation is feasible. When just employing $NN_{bw}(\cdot)$ to learn the mapping, the corresponding mean RMSE and $Error_{max}$ are 0.20 and 0.36 dB, with the standard deviation of 0.09 and 0.17 dB, respectively. It is also observed in Fig.13 that no improvements are obtained after performing fine–optimization. Indeed, slightly higher mean and standard deviations are observed. This may have to do with the fact that $NN_{bw}(\cdot)$ already provides theoretically minimum achievable error. A similar trend is observed for the simulations results in Fig. 8(f) as explained earlier.

B. Flat and tilted gain design

Next, we investigate if the learned mapping, between the Raman gain profiles and pump powers, can be used to predict pump powers that will result in a flat and tilted gain profiles. We would also like to stress that given the four pump powers with ranges specified in Table III and fixed wavelength, we are not sure if it is physically possible to achieve completely flat gain across the entire C–band.

In Fig. 14(a)-(b), the predicted and the target (flat) gain profiles are shown before and after the fine–optimization. It is observed that when only employing NN_{bw}, the deviation between the predicted and target gain increases for gains beyond 4 dB. However, after applying fine–optimization the predicted flat gain profiles are achievable. We should also keep in mind that the employed Raman module for the experiment, provides lower gains for lower frequencies. This may be the reason why the largest deviations from the targeted flat gain are observed at lower frequencies. Fig. 14(c)-(d) shows the RMSE and $Error_{max}$ as a function of gain. It is observed that

relatively low RMSE and $Error_{max}$ below 0.5 dB and 1.4 dB, respectively, are obtainable.

The results for the design of the tilted gain are shown in Fig. 15. The results indicate that accurate tilted gain design is feasible using the proposed machine learning framework. Fig. 15(c-d) shows that compared to the flat gain design Fig. 14(c-d), lower maximum RMSE and $Error_{max}$ are achievable before and after fine–optimization.

V. Discussions on speed, complexity and spectrum used for training

Regarding the complexity and speed: the approach presented in our manuscript relies on supervised learning (training stage) which can be performed as part of the calibration stage or during the deployment stage. During the training stage, the mapping between the Raman gain and pump powers and wavelengths are learned. This implies that the weight matrices of the multi–layer neural network, $NN_{bw}(\cdot)$, are learned and the hyper–parameters are determined. Once this has been completed, the response of the $NN_{bw}(\cdot)$ is almost immediate for an arbitrary gain design as it corresponds to matrix multiplication. This is not the case for the approach that relies on genetic algorithms, [10], where for any new gain profile a new optimization in terms of pump powers and wavelengths needs to be performed. As already mentioned in the introduction, this has certain drawbacks.

Regarding the complexity added by the fine–optimization, it is shown that, in many cases, the first ML–NN, $NN_{bw}(\cdot)$, will provide accurate enough results such the fine–optimization is not needed. In case a user needs a more accurate solution, the fine-optimization process based on the second ML-NN, $NN_{fw}(\cdot)$, will add some extra complexity as the pump powers and wavelengths are fine adjusted using gradient descent. However, $NN_{bw}(\cdot)$ provides a good set of initial pump powers and wavelengths and therefore the fine–optimization may converge after a small number of iterations. The convergence time will also depend on the required accuracy of the predicted gain. Also, there are many modification to gradient descent that can be employed to accelerate the learning and obtain faster convergence time [12]. This is left for the future work, as the main objective of the submitted manuscript is to demonstrate the novel idea and also trigger new research directions.

Finally, we can affirm that a direct and quantitative comparison with method presented in [10] and [11] in term of complexity and speed can not easily be conducted. Computational complexity for example in [10] depends on the number of iterations in the genetic algorithm, that can vary based on the the algorithm set–up and targeted accuracy. Finally, considering just the speed of each approach depends on even larger set of conditions, like specific numerical algorithms used for the implementation, the hardware employed (GPU or CPU) and more precisely on their combination.

To demonstrate the main principles behind the proposed framework, we needed to consider a simplified case where the input power spectrum is flat. Considering the non–flat input spectrum, the set of per-channel power levels to be swept for the generation of the training data-set would significantly increase. This would translate into an increased input dimensionality $NN_{bw}(\cdot)$.

However, our assumption is not far from practical cases: usually a well designed link is run with almost flat power in every span. In our example of applications, we show how to design a Raman amplifier delivering a flat gain. And in case of second amplification stage based on EDFA, usually EDFAs are coupled with a gain flattening filter (GFF) to equalize the output spectrum.

We agree that in realistic cases spectral flatness cannot perfectly be controlled, but ripples/tilts on span per span basis can be very limited, of the order of 0.5 dB. Under this condition of small perturbation of the flatness we can further add two comments. First is that such ripples/tilts are distributed over the spectrum in random manner, so span after span they do not accumulate but usually there is a statistical averaging toward the flat condition. Second, if the ripple/tilt is not very large as we commented above, the Raman amplification will not suffer so much this variation, and gains predicted by $NN_{bw}(\cdot)$ will still have limited errors. Moreover, the $NN_{bw}(\cdot)$ could even be used to design a Raman amplifier able to cancel the ripple/tilt delivering a flat output spectrum. This is the reason why we inserted the tilted gain design case in our paper. Finally, if the error resulting from the non–flat spectrum is too large, fine–optimization will act on it and reduce it.

In the end, even if the main motivation of having selected a flat input spectrum as simplified assumption to ease the analysis of the proposed framework, we can consider it not very far from practical condition of realistic optical links. However, for the future work, plan to investigate the performance of the proposed framework when the input spectrum is not flat.

VI. Conclusion and future work

A low–complexity and low–latency framework, based on machine learning, has been demonstrated for the inverse system design. The framework consists of two

stages: 1) a multi-layer neural network employed to learn the inverse mapping and 2) a multi-layer neural network, representing the forward mapping, in combination with the gradient descent algorithm for fine–adjustments. The first stage can also be used in combination with various optimization algorithms to provide a qualified set of initial conditions.

The framework has been demonstrated, using numerical simulations, to provide highly-accurate pump power and wavelength configurations for the design of C and C+L–band Raman amplifiers with arbitrary, flat and tilted gain profiles. Moreover, the corresponding experimental verification has been presented for the design of C–band Raman amplifiers. The experimental results demonstrate that highly–accurate pump power allocations for arbitrary, flat and tilted gains are feasible using the proposed framework.

To reduce the training time, for the future work, it would be useful to investigate the minimum size of the training data–set required to obtain a certain error. As a final remark, experimental validation extended to the C+L–band would be desirable and will be considered in future publications.

VII. Acknowledgments

This work is supported by the European Research Council through the ERC-CoG FRECOM project (grant agreement no. 771878) and European Union Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 754462. We would also like to thank Md Asif Iqbal, from Aston University, whose detailed comments and insights, on this topic, have significantly improved the paper.

Appendix

A. Learning mappings using neural networks

The objective of the neural network training is to learn the underlying generator of the training data–set given a set of input and output variables $\mathbf{x} = [x_1, ..., x_N]$ and $\mathbf{y} = [y_1, ..., y_N]$, respectively. The data generation process is described as $\mathbf{y} = f(\mathbf{x}) + \epsilon$, where $f(\cdot)$ is a mapping function and ϵ is an additive noise term that has Gaussian distribution. It is crucial to consider the noise term ϵ as all measurements are noisy or to express it differently, there will always be an uncertainty associated with measurements.

Considering the finite size of the training data–set, N, the neural network can only learn the underlying data generation within the set $(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^{N\times 1}$. Most importantly, we must limit the complexity of the network by introducing a regularization term to achieve a good balance between bias and the variance of the networks output. So, already at this point we can state that the output of the neural network, denoted by $NN(\mathbf{x}; \mathbf{w})$, is an approximation to the output of the "true" system $\mathbf{y}$. $\mathbf{w}$ are the weights describing the neural network. Therefore, there will always be present a finite error between the output of the neural network and $\mathbf{y}$.

Now, let's go one step further. The neural network training is performed by solving the sum–of–squares error which is defined as:

$$E(\mathbf{w}) = \frac{1}{2}\sum_{k=1}^{N}[NN(x_k; \mathbf{w}) - y_k]^2 \tag{4}$$

We would like the network output to be as close as possible to y_k, and the minimization of the error is obtained by following functional differentiation:

$$\frac{dE}{dNN(x_k; \mathbf{w})} = 0 \tag{5}$$

The solution to this minimization problem is [18]:

$$NN(x_q; \mathbf{w}^*) = E[y_k|\mathbf{x}] = \int y_k p(y_k|\mathbf{x}) dy_k \tag{6}$$

where $p(y_k|\mathbf{x})$ is the conditional probability density and $\mathbf{w}^*$ represent the optimal weights, $E[\cdot]$ represents average and $E[y_k|\mathbf{x}]$ is the conditional average. This means that given the optimal neural network weights, $\mathbf{w}^*$, the output of the neural network corresponds to the conditional average of the output data conditioned on the input vector. In conclusion, we do not get the exact replica of $f(\cdot)$ or $f^{-1}(\cdot)$.

B. Dependence on input signal power levels

In this section, the dependence of the predicted Raman gain profile on per–channel input signal power levels is investigated. We have already shown in [14] that no significant gain prediction errors occur when changing the per-channel inputs signal power levels from -3 dBm to 3 dBm. This verifies that if the Raman amplifier is not operated in the saturated regime, the gain is almost independent of the input signal power levels. Typically, any practical application of Raman amplifier usually avoids the operation in saturation regime.

To further investigate the dependence on the input signal power levels and demonstrate that the performance of the proposed framework is independent over even a broader range of input signal power levels, a new set of simulations have been carried out.

The employed framework is trained with 0 dBm per-channel input signal power levels and only $NN_{bw}(\cdot)$

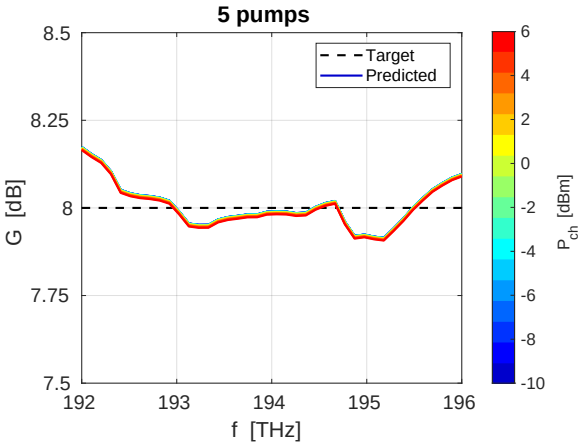

Fig. 16. Gain profile for increasing per-channel input signal power levels from -10 dBm to 6 dBm.

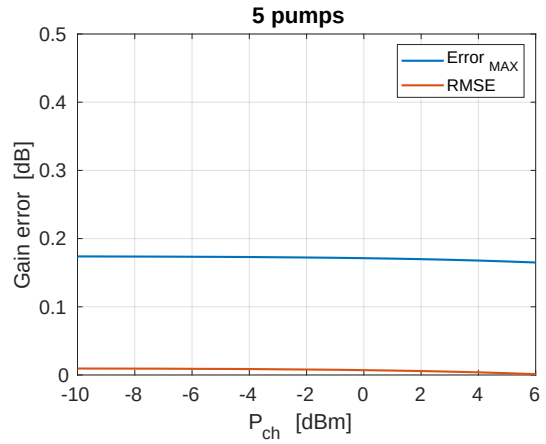

Fig. 17. Gain profile prediction errors for increasing per-channel input signal power levels from -10 dBm to 6 dBm.

is employed. Five Raman pumps are considered. The machine learning framework is employed to predict pump powers and wavelengths needed to deliver a flat gain of 8 dB.

Next, we use the predicted pump powers and wavelengths, and run the Raman solver to obtain the predicted Raman profile gains. We also vary per-channel inputs signal power levels. We still consider flat input signal power spectrum. In Fig. 16, the predicted gain is plotted as a function of frequency when varying per-channel input signal power levels from -10 dBm to 6 dBm (blue to red lines). It is observed that the predicted Raman gain profile is not affected by changing the input signal power levels. Next, in Fig. 17, the maximum error and RMSE are plotted as a function of per-channel input signal power levels. The errors have a negligible dependence on input power. Moreover, we would like to stress that here we have only used the $NN_{bw}(\cdot)$ for predicting the gain, without employing the fine–optimization proposed in the paper, as it was not needed.

REFERENCES

[1] S. Molesky, Z. Lin, A. Y. Piggott, W. Jin, J. Vuckovic, and A. W. Rodriguez, "Outlook for inverse design in nanophotonics," Nature Photonics, vol. 12, no. November, 2018.

[2] D. Pérez, I. Gasulla, L. Crudgington, D. J. Thomson, A. Z. Khokhar, K. Li, W. Cao, G. Z. Mashanovich, and J. Capmany, "Multipurpose silicon photonics signal processor core," Nature Communications, vol. 8, no. 1, p. 636, 2017.

[3] L. Zhuang, C. G. H. Roeloffzen, M. Hoekman, K.-J. Boller, and A. J. Lowery, "Programmable photonic signal processor chip for radiofrequency applications," Optica, vol. 2, no. 10, p. 854, 2015.

[4] L. Velasco, "Machine Learning for Network Automation : Overview , Architecture , and Applications [Invited Tutorial]," vol. 10, no. 10, pp. 126–143, 2018.

[5] S. Echeverri-Chacón, J. J. Mohr, J. J. V. Olmos, P. Dawe, B. V. Pedersen, T. Franck, and S. B. Christensen, "Transmitter and dispersion eye closure quaternary (TDECQ) and its sensitivity to impairments in pam4 waveforms," J. Lightwave Technol., vol. 37, no. 3, pp. 852–860, Feb 2019.

[6] J.-H. Lu, D.-P. Cai, Y.-L. Tsai, C.-C. Chen, C.-E. Lin, and T.-J. Yen, "Genetic algorithms optimization of photonic crystal fibers for half diffraction angle reduction of output beam," Opt. Express, vol. 22, no. 19, pp. 22 590–22 597, Sep 2014.

[7] A. Napoli, N. Costa, J. K. Fischer, J. ao Pedro, S. Abrate, N. Calabretta, W. Forysiak, E. Pincemin, J. P.-P. Gimenez, C. Matrakidis, G. Roelkens, and V. Curri, "Towards multiband optical systems," in Proceedings of Signal Processing in Photonic Communications (SPPCom) Conference, 2018, p. NeTu3E.1.

[8] B. Neto, A. L. J. Teixeira, N. Wada, and P. S. Andre, "Efficient use of hybrid Genetic Algorithms in the gain optimization of distributed Raman amplifiers." Optics express, vol. 15, no. 26, pp. 17 520–17 528, 2007.

[9] G. C. Ferreira, S. P. Cani, M. J. Pontes, and M. E. Segatto, "Optimization of distributed Raman amplifiers using a hybrid genetic algorithm with geometric compensation technique," IEEE Photonics Journal, vol. 3, no. 3, pp. 390–399, 2011.

[10] J. Chen and H. Jiang, "Optimal Design of Gain-Flattened Raman Fiber Amplifiers Using a Hybrid Approach Combining Randomized Neural Networks and Differential Evolution Algorithm," IEEE Photonics Journal, vol. 10, no. 2, pp. 1–15, 2018.

[11] J. Zhou, J. Chen, X. Li, G. Wu, Y. Wang, and W. Jiang, "Robust, compact, and flexible neural model for a fiber Raman amplifier," Journal of Lightwave Technology, vol. 24, no. 6, pp. 2362–2367, 2006.

[12] I. Goodfellow, Y. Bengio, and A. Courville, Deep Learning. MIT Press, 2016, http://www.deeplearningbook.org.

[13] Y. Pointurier, "Design of low-margin optical networks," J. Opt. Commun. Netw., vol. 9, no. 1, pp. A9–A17, Jan 2017.

[14] D. Zibar, A. Ferrari, V. Curri, and A. Carena, "Machine learning-based raman amplifier design," in Optical Fiber Communication Conference (OFC) 2019. Optical Society of America, 2019, p. M1J.1. [Online]. Available: http://www.osapublishing.org/abstract.cfm?URI=OFC-2019-M1J.1

[15] M. Ionescu, "Machine learning techniques in optical communication," in Proceedings of International Conference on Transparent Optical Networks (ICTON), p. We.B7.3, 2019.

[16] S. Boyd and L. Vandenberghe, Introduction to Applied Linear Algebra: Vectors, Matrices, and Least Squares, 2017.

[17] A. M. Rosa Brusin, V. Curri, D. Zibar, and A. Carena, "An ultra-fast method for gain and noise prediction of raman amplifiers," in European Conference on Optical Communication, ECOC, 2019, p. Th.1.C.3.
[18] C. M. Bishop, Pattern recognition and machine learning. Springer, 2006.
[19] J. Bromage, "Raman Amplification for Fiber Communications Systems," Journal of Lightwave Technology, vol. 22, no. 1, pp. 79–93, January 2004.
[20] V. Curri and A. Carena, "Merit of Raman Pumping in Uniform and Uncompensated Links Supporting NyWDM Transmission," Journal of Lightwave Technology, vol. 34, no. 2, pp. 554–565, Jan. 2016.