Polynomial Modeling of Noise Figure in Erbium-Doped Fiber Amplifiers

fibers

MDPI

Article

Polynomial Modeling of Noise Figure in Erbium-Doped Fiber Amplifiers

Rocco D'Ingillo [1,*], Alberto Castronovo [1], Stefano Straullu [2] and Vittorio Curri [1]

[1] Politecnico di Torino, Department of Electronics and Telecommunications, 10129 Turin, Italy; s290238@studenti.polito.it (A.C.); vittorio.curri@polito.it (V.C.)
[2] LINKS Foundation, 10138 Turin, Italy; stefano.straullu@linksfoundation.com
* Correspondence: rocco.dingillo@polito.it

Highlights

What are the main findings?

- A polynomial regression model using Generalized Least Squares (GLS) is proposed for real-time noise figure estimation in Erbium-Doped Fiber Amplifiers (EDFAs), achieving accuracy within the measurement uncertainty of optical spectrum analyzers.
- The model outperforms traditional analytical methods and requires significantly less data than deep-learning approaches while maintaining high computational efficiency (inference time below 0.2 ms per evaluation).

What are the implications of these findings?

- The proposed approach enables fast and accurate EDFA noise figure prediction, making it suitable for real-time digital-twin applications in optical networks.
- This method provides a scalable and efficient alternative to complex machine-learning models, allowing for practical implementation in next-generation optical communication systems.

Abstract: Erbium-Doped Fiber Amplifiers (EDFAs) are fundamental to optical communication networks, providing signal amplification while introducing noise that affects system performance. Accurate noise figure estimation is critical for optimizing link budgets, monitoring optical Signal-to-Noise Ratio (OSNR), and enabling real-time network optimization. Traditional analytical models, while computationally efficient, often fail to capture device-specific variations, whereas machine-learning-based approaches require large training datasets and introduce high computational overhead. This paper proposes a polynomial regression model for real-time EDFA noise figure estimation, striking a balance between accuracy and computational efficiency. The model leverages Generalized Least Squares (GLS) regression to fit a multivariate polynomial function to measured EDFA noise figure data, ensuring robustness against measurement noise and dataset variations. The proposed method is benchmarked against experimental measurements from multiple EDFAs, achieving prediction errors that are within the measurement uncertainty of Optical Spectrum Analyzers (OSAs). Furthermore, the model demonstrates strong generalization across different EDFA architectures, outperforming analytical models while requiring significantly less data than deep-learning approaches. Computational efficiency is also analyzed, showing that inference time is below 0.2 ms per evaluation, making the model suitable for real-time digital-twin applications in optical networks. Future work will explore hybrid modeling approaches, integrating physics-based regression with Machine Learning (ML) to enhance performance in high-variance spectral regions. These results highlight the potential of lightweight polynomial regression models as an alternative to

Academic Editor: Paulo Caldas

Received: 7 February 2025
Revised: 6 March 2025
Accepted: 12 March 2025
Published: 14 March 2025

Citation: D'Ingillo, R.; Castronovo, A.; Straullu, S.; Curri, V. Polynomial Modeling of Noise Figure in Erbium-Doped Fiber Amplifiers. *Fibers* **2025**, *13*, 34. https://doi.org/10.3390/fib13030034

complex ML-based solutions, enabling scalable and efficient EDFA performance prediction for next-generation optical networks.

Keywords: EDFA; noise figure; polynomial model; machine learning; optical networks

1. Introduction

The increasing demand for applications such as virtual and augmented reality, expansive cloud services, and high-definition video streaming has led to a significant need for enhanced capacity in optical and wireless networks [1]. As these technologies evolve, they require higher bandwidth and lower latency, driving continuous improvements in network infrastructure. This poses a substantial challenge for 5G networks, as the existing backhaul infrastructure from previous generations is insufficient to meet the stringent requirements for very low latency and efficient dense traffic management [2]. Currently, optical systems are nearing their theoretical capacity limits due to advancements like probabilistic constellation shaping and forward error correction, which have maximized data transmission efficiency and pushed towards the Shannon limit. Future progress must focus on optimizing the physical layer and enhancing the control layer with more efficient resource allocation and management [3–5]. Emerging technologies such as Software-Defined Networking (SDN), Elastic Optical Networks (EONs), and self-driving optical networks are pushing performance further. SDN separates network control from hardware, enabling dynamic resource management and reconfiguration. EON allows flexible spectrum allocation based on traffic demands, improving bandwidth efficiency. This approach enables the automation and optimization of the network operations, enhancing overall efficiency and reliability [6].

Modeling Erbium-Doped Fiber Amplifiers (EDFAs) is essential for optimizing optical networks, as these components determine transmission bandwidth and significantly impact the Optical Signal-to-Noise Ratio (OSNR) [7]. Although EDFAs are well-established, parameters such as gain, Amplified Spontaneous Emission (ASE), and noise figure can vary significantly with signal attributes like input power and channel frequency, and these variations do not follow simple analytical formulas [8]. Therefore, developing accurate predictive models for EDFA parameters is a promising challenge. Several models for EDFA parameters exist in the literature [7,9,10], primarily focusing on gain estimation [11]. Explicit models are cost-effective as they rely on a priori knowledge of the devices but often lack high accuracy. Data-driven models, based on neural networks, offer greater accuracy [12–14] but are challenging to implement in real-world applications due to the large datasets required for training, which are often unavailable or expensive to obtain.

This paper proposes a polynomial model for the noise figure of EDFAs, emphasizing high-speed implementation to integrate seamlessly with comprehensive machine-learning EDFA models without significant computational delays. The model is designed for flexibility and portability, allowing users to adjust the training dataset size to balance accuracy and measurement costs.

2. State-of-the-Art

2.1. Overview of EDFA Devices

EDFAs are essential components in fiber-optic telecommunications, known for their high gain, low noise, and broad wavelength support. These devices amplify input signals in optical fibers using a core doped with erbium ions. The amplification process involves a

pump laser that excites the erbium ions, leading to the stimulated emission of additional photons [7]. EDFAs mainly operate in three modes:

1. **Automatic Gain Control (AGC):** Regulates the laser source current to achieve a target gain.
2. **Automatic Power Control (APC):** Similar to AGC but targets output power.
3. **Automatic Current Control (ACC):** Delivers a constant current to the laser source.

Commercial EDFAs allow setting the target gain (in dB) and target tilt, which indicates the slope of the peak power of active channels in the output power spectrum, usually evaluated as the difference between the power levels of the first and last channel within the bandwidth of interest (typically the optical C-band or L-band), also expressed in dB.

2.2. Noise Sources in Amplifiers

Various sources contribute to noise in optical amplifiers [15]:

1. **Signal–spontaneous (sig–sp) beating:** Results from the mixing of the coherent signal with incoherent ASE in the same polarization. It has a flat frequency distribution and is proportional to gain and input signal power.
2. **Spontaneous–spontaneous (sp–sp) beating:** Involves copolarized spectral components of ASE, independent of input signal power and gain, and is more pronounced around the central frequency.
3. **Multipath interference (MPI):** Converts phase or frequency noise in the input signal to intensity noise in the output signal. It has low bandwidth and increases with input power and gain.
4. **Shot noise:** Random fluctuations in the photo-generated current in the photodetector, resembling white noise, and proportional to the average photo-generated current.

For simplicity, we refer to the signal–spontaneous (sig–sp) beating contribution as *ASE noise*. The most significant noise in EDFAs is ASE, as both sig–sp and sp–sp beating depend on it, with sig–sp generally being dominant [16]. ASE occurs when excited erbium ions decay to the low-energy state before interacting with input signal photons, resulting in the spontaneous emission of photons with random phase and direction. Some of these photons align with the propagation axis, mixing with and amplifying the input signal, producing noise proportional to the amplifier's gain. Equations describing ASE noise are derived using carrier population and power formulas, with parameters obtained from complex experimental measurements. However, these parameters can change over time (e.g., due to aging), making one-time measurements unreliable long-term. Periodic measurements are required to maintain accuracy, which is expensive and impractical for in-use devices. Therefore, simpler models for ASE noise and other parameters are often preferred.

2.3. Noise Figure of EDFAs

EDFAs introduce unwanted optical power fluctuations to the amplified output signal, with ASE noise being the most significant contributor, as discussed earlier. This unwanted addition reduces the OSNR of the transmitted signal, making its measurement essential. To quantify this effect, a parameter known as the noise figure is used, similar to the approach in electronic amplifiers [17]. The noise figure NF is defined as the noise factor, F, expressed in dB units.

$$NF = 10\,log_{10}\,F \tag{1}$$

The noise factor, F, is the ratio of the Signal-to-Noise Ratio (SNR) at the input and output of the amplifier. It depends on the optical frequency, ν, and the baseband frequency, f, of the electrical output from the photodetector used in the receiver.

$$F(\nu, f) = \frac{SNR_{IN}}{SNR_{OUT}(\nu, f)} \tag{2}$$

In essence, the noise figure indicates the degradation of signal quality due to the amplifier's insertion into the optical line. Historically, from the 1980s to the early 2000s, it was believed that the noise figure of a high-gain, phase-insensitive linear amplifier had a lower bound of 3 dB [18], due to the Heisenberg uncertainty principle. Even though the noise figure of a phase-sensitive amplifier can be below 3 dB, it requires the transmission of a phase-conjugated signal and idler with a pump reference, and this remains a research topic without commercial deployment [19]. When discussing lumped amplifiers like EDFAs, however, the 3 dB noise figure limit is fundamental, and achieving this value is still considered a significant challenge. The best result obtained in the measurements used in this work is around 3.7 dB.

To evaluate the noise figure, we consider the setup shown in Figure 1. The receiver is assumed to be ideal, and the laser source is shot-noise limited, meaning it has a high SNR and that the primary contribution to source noise is shot noise.

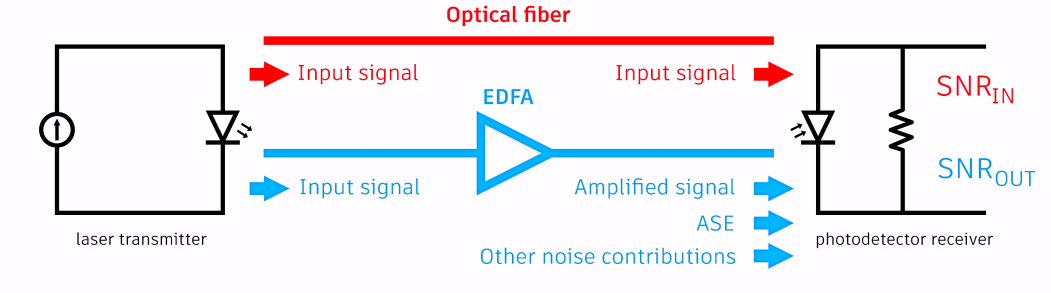


Figure 1. Input and output SNR measurement setups for noise figure evaluation.

The SNR for a shot-noise-limited source is given by:

$$SNR_{IN} = \frac{\eta P_{IN}}{2h\nu B_e} \tag{3}$$

where η is the quantum efficiency of the photodetector, h is Planck's constant expressed in J · s, and B_e is the noise bandwidth in Hz. For an ideal receiver, η equals 1.

The output SNR of the amplifier can be expressed as:

$$SNR_{OUT} = \frac{G^2 P_{IN}^2}{B_e[S_e(\nu, f) + \eta^{-1} S_{shot}]} \tag{4}$$

where the noise spectral density, S_e, expressed in $W^2\,Hz^{-1}$, accounts for all noise contributions except shot noise. The noise factor is then:

$$F(\nu, f) = \frac{S_{total}(\nu, f)}{2hfG^2 P_{IN}} \tag{5}$$

The spectral density of signal–spontaneous beating noise and shot noise can be written as:

$$S_{sig-sp} = 4\rho_{ASE} G P_{IN},\ S_{shot} = 2hfGP_{IN} \tag{6}$$

where ρ_{ASE} is the ASE power density. Other noise contributions can be neglected if the input channel power is above the effective input noise of the amplifier (typically around −50 dBm) [17].

The noise factor can now be rewritten as:

$$F = \frac{4\rho_{ASE}GP_{IN} + 2hfGP_{IN}}{2hfG^2P_{IN}} = \frac{2\rho_{ASE}}{hfG} + \frac{1}{G} \tag{7}$$

where the term $1/G$, corresponding to the shot noise, can usually be neglected. Finally, the expression can be rewritten by replacing ρ_{ASE} with the corresponding power, a metric more commonly obtained from spectrum analyzer measurements. This is achieved by multiplying the ASE noise power density by the corresponding noise bandwidth, B_0. The final expression for the noise factor is:

$$F = \frac{2\rho_{ASE}B_0}{hfGB_0} = \frac{P_{ASE_{AMP}}}{hfGB_0} \tag{8}$$

The term $P_{ASE_{AMP}}$ represents the ASE power introduced by the amplifier. It is calculated by measuring the ASE power at the output of the amplifier and subtracting the Source Spontaneous Emission (SSE) power multiplied by the gain of the amplifier. This procedure aims to exclude the contribution of pre-existing signal noise, which is also amplified. Figure 2 illustrates this process.

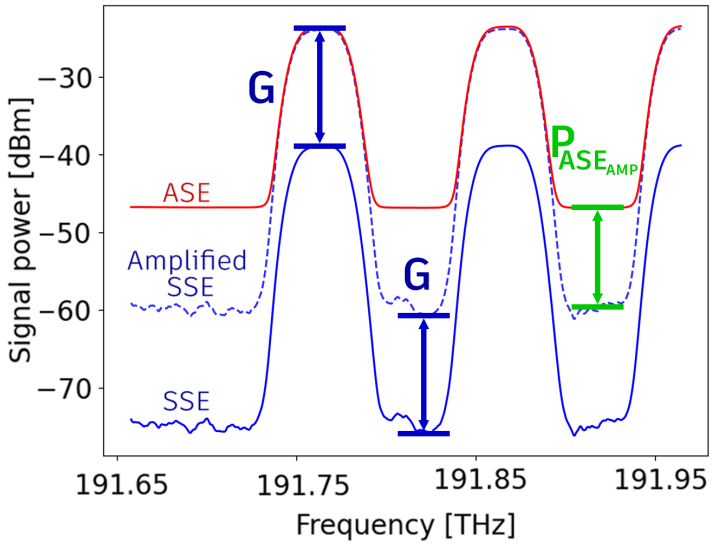


Figure 2. Illustration of $P_{ASE_{AMP}}$. The output spectrum (red) is shown alongside the input spectrum (blue) adjusted by the amplifier gain (blue dashed). $P_{ASE_{AMP}}$ represents the noise introduced by the EDFA to the amplified SSE level.

The final expression for the noise figure is:

$$NF\,|_{dB} = P_{ASE_{AMP}}\,|_{dBm} - G\,|_{dB} - 10\,log_{10}\,(hfB_0) \tag{9}$$

It should be noted that the formula explicitly depends on the ASE noise power and therefore on the input signal power, the amplifier gain, and the channel frequency.

2.4. Comparison of Different EDFA Models

Estimating the parameters of network elements like EDFAs using specialized models is crucial for the optimal design, operation, and maintenance of optical communication systems. This estimation allows for accurate performance predictions of network elements under various conditions without the need to deploy the network initially, thereby greatly aiding in network design. Various models for EDFAs have been proposed in numerous articles [7,10]. Most early models focus on gain estimation, which is notoriously difficult to analyze due to two primary factors: static-state inter-channel gain variation and dynamic gain excursion [17]. Static-state variation arises from uneven gain across channels due

to the dependence of the fiber's absorption and gain spectra on the signal wavelength. Dynamic gain excursion refers to rapid gain fluctuations in response to changes in input power or channel configurations, which are common in optical networks [7].

Analytical models derive formulas for the theoretical estimation of EDFA parameters, such as gain and noise figure. By starting from fundamental physical equations, such as those governing the carrier populations in the energy levels of erbium ions, comprehensive analytical expressions for these parameters can be obtained. The main advantage of analytical models is that they depend solely on a priori knowledge of the equipment and a restricted amount of measurements, with values possibly provided by manufacturers, and do not require any deployment or input–output characterization. While this is the simplest method, it does come with several challenges. Firstly, these models are typically very complex and rely on several assumptions, such as the two-energy-level approximation for erbium ions [20], which can compromise their accuracy. Additionally, they do not consider dynamic effects, such as those previously mentioned for gain estimation, nor manufacturing defects. Another issue is the inconsistency between different devices. EDFAs can be constructed with various architectures and include different components in their schematics. Consequently, an analytical model based on the underlying physical equations must be adapted for each device architecture, making it impractical to apply the same model to different brands and devices. This issue is directly related to another challenge with commercial devices: manufacturers often keep the details of their device architectures confidential, making it difficult to develop accurate models for consumers. Some simpler analytical models for gain estimation are available, such as those based on a center of mass function [21], which enable evaluation of gain in AGC mode using a straightforward expression. These models can be easily adapted to multiple devices. However, their accuracy is limited because they overlook various internal effects of the devices. Specifically, they assume an ideally constant gain spectrum shape, which does not hold true for real devices, and they focus on the impact of the loaded channel configuration, disregarding gain variations due to channel power fluctuations. Finally, there is the issue of aging. The physical parameters used in these models change over time, necessitating continuous updates to maintain their reliability. This process can be very costly, as these parameters are determined through complex experimental measurements. While it is possible to incorporate time-dependent factors into the equations, doing so would further increase the already high complexity of the model.

Given the challenges associated with the analytical models of EDFA parameters, research has explored alternative solutions and technologies. The most promising alternatives are machine-learning models, especially those based on neural networks. These models offer two major advantages: accuracy and flexibility. There are several examples in the literature of models that consistently estimate gain with an RMSE of around 0.15 dB [7,22], and these models can be easily adapted or extended to many different devices and brands with little modification or training. The primary drawback of these models, and a significant barrier to their practical implementation, is their typically high requirement for large training datasets. Acquiring such datasets is often expensive or impractical. While smaller datasets can be utilized, they pose challenges in ensuring performance reliability, particularly in complex systems. One of the goals of the model proposed in this work is to offer a rapid and adaptable solution capable of being trained with datasets of varying sizes, aiming to strike a balance between achieving target accuracy and minimizing the costs associated with required measurements.

3. Materials and Methods

The EDFA model proposed in this work is specifically designed to deliver a precise estimation of the noise figure. However, it is not intended to function as an independent, stand-alone model. Instead, its purpose is to integrate within a broader digital-twin framework for network modeling while remaining entirely brand-agnostic. To fulfill these objectives, the model must adhere to the following key requirements:

1. **Accuracy:** The model should maintain a high level of precision, ensuring that errors remain comparable to the inherent measurement uncertainties present in the training datasets.
2. **Computational efficiency:** It must be capable of processing data swiftly to prevent performance bottlenecks, particularly when handling extensive training and test datasets within the complete network model.
3. **Brand neutrality:** The model should be easily adaptable across different manufacturers and device types with minimal modifications.
4. **Usability and portability:** While not a strict requirement, the model should ideally be intuitive to use and easily portable to different environments.

With respect to the model's input parameters, Equation (9) establishes that the noise figure is dependent on $P_{ASE_{AMP}}$, G, and f. The EDFAs considered in this study operate in AGC mode, allowing adjustments to both the target gain and tilt. Consequently, the channel gain, G, is primarily determined by these two parameters. Moreover, $P_{ASE_{AMP}}$ is influenced not only by gain but also by the input signal power. Based on these considerations, the model employs the following four input parameters:

1. **Input signal power**, expressed in dBm;
2. **Gain**, measured in dB;
3. **Tilt**, represented in dB;
4. **Channel frequency**, given in THz.

The central objective of this study is to attain an accuracy level where the maximum observed errors in the test datasets remain within or below the theoretical uncertainty of the noise figure, as analytically derived from the measurement data.

3.1. Polynomial Model Construction

After evaluating various possible approaches for implementing the model, a polynomial structure was selected. The model is formulated as a multivariate polynomial expression, which is efficiently computed using Horner's rule [23]. The polynomial coefficients are obtained through linear regression employing the Generalized Least Squares (GLS) algorithm [24]. Details regarding Horner's rule and the GLS algorithm application are presented in the following sections. While this model technically falls within the domain of machine learning, it distinguishes itself from conventional ML techniques and neural network-based models due to its simplicity, computational efficiency, and ease of deployment. The entire project was developed using the Python [25] (version 3.10) programming language, which was chosen for its robust open-source ecosystem, flexibility, and rapid development capabilities, as well as its extensive collection of well-documented libraries for data processing and visualization.

The modeling process begins with the evaluation of the noise figure from the signal power spectra measured before and after amplification by the EDFA. This procedure follows the methodology outlined in various OSA user manuals [26], which serve as a reference for the subsequent sections. Once the active channel frequencies are identified, a dataset is constructed. Each row in this dataset corresponds to a unique combination of four input

parameters—input power, target gain, target tilt, and channel central frequency—along with the associated noise figure value.

3.2. Test Setup

Figure 3 illustrates the setup used to obtain the measurement datasets.

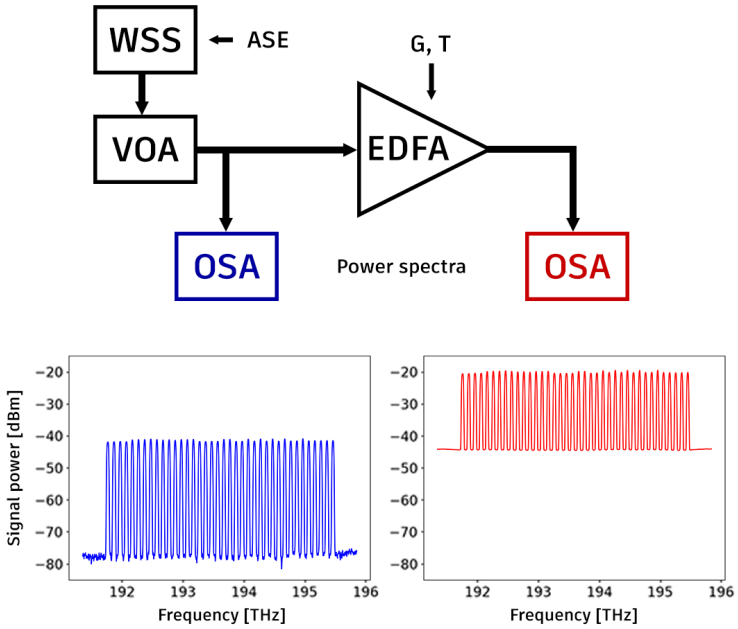


Figure 3. Diagram of the test setup for measuring input (blue) and output (red) power spectral densities.

This configuration enables the measurement of the input and output Power Spectral Densities (PSD) of the signal passing through the amplifier. The signal is fed to the amplifier using a Variable Optical Attenuator (VOA) to achieve the desired input power, while the target gain and tilt are set programmatically on the EDFA.

Devices from different vendors—and even different device types from the same vendor—require individual characterization and distinct models due to variations in internal EDFA architecture, which result in performance differences across devices. However, devices of the same type from the same vendor are treated as a single group, as the primary differences among them stem from manufacturing tolerances. Most of these devices support both high- and low-gain operation modes, which must be considered separately when developing models. Manufacturers typically optimize the noise figure independently for each mode, leading to uncorrelated performance characteristics.

The characterization process is influenced by the spectral load used during measurements. These measurements were conducted under an even spectral load, with the number of active channels varying between 38 and 48 depending on the device model. The input power profile followed a flat Wavelength Division Multiplexing (WDM) comb distribution. The channels were spaced at 100 GHz, each with a bandwidth of 32 GHz. All measurements were performed using an OSA with a Resolution Bandwidth (RBW) of 10 GHz.

To construct the datasets, it is essential to define appropriate step sizes for the input parameters. This strategy ensures a comprehensive understanding of their influence on the noise figure while maintaining dataset size within practical limits. Accordingly, the parameters are selected as follows:

1. **Total input power**: For the low-gain range, values span from −10 dBm to 6 dBm, while, for the high-gain range, they range from −18 dBm to 2 dBm, both with a step size of 2 dB.
2. **Gain**: Depending on the specific EDFA model and gain range, values are selected within the 10 dB to 35 dB range, with a step size of 1 dB.
3. **Tilt**: Defined within either −5 dB to 5 dB or −3 dB to 3 dB, depending on the EDFA model, with a step size of 1 dB.

Subsequent sections assess whether or not these initial step size choices are appropriate. This analysis will determine whether finer steps are necessary to maintain the desired accuracy or if larger steps could be sufficient, potentially reducing the required dataset size.

The datasets collected from OSA measurements are stored in MATLAB (.mat) files. Each file corresponds to a specific device and input power configuration and follows the structure outlined below:

1. **Gain_target:** A 1D array containing the target gain values set on the device, in dB.
2. **Gain_real:** A 1D array containing the actual gain values achieved by the device, in dB, serving as feedback relative to the target gain.
3. **Tilt_target:** A 1D array containing the target tilt values set on the device, in dB.
4. **Tilt_real:** A 2D array (one row per gain value) containing the actual tilt values achieved by the device, in dB, serving as feedback relative to the target tilt.
5. **spectrum_freq:** A 1D array listing all frequency points at which the OSA evaluated signal power, in THz.
6. **spectrum_TX_power:** A 1D array containing signal power measurements at the amplifier input across the entire frequency range, in dBm.
7. **spectrum_RX_power:** A 3D array (one row per target gain and tilt pair) containing signal power measurements at the amplifier output across the frequency range, in dBm.
8. **TOT_Power_IN:** A floating-point value representing the total input power of the amplifier, in dBm.
9. **TOT_Power_OUT:** A 2D array (one floating-point value per target gain and tilt pair) representing the total output power of the amplifier, in dBm.
10. **OSA_PARAMS:** A structured dataset containing additional parameters, including device vendor information, fiber connector type (flat or angled), RBW in GHz.

3.3. Optical Spectrum Processing

To adjust the offset of the power spectra, an intermediate step is necessary. OSA processing methods cause a discrepancy between the total power, indicated by *TOT_Power_IN* and *TOT_Power_OUT*, and the power evaluated from the corresponding power spectra. To reconcile these values, a normalization factor is needed. The total signal power, expressed in dBm, can be calculated from the power spectral density using the following equation:

$$P_{spectrum} = 10\log_{10}\left(\sum_{n=1}^{N} p_n|_{\mu W} \cdot \frac{\Delta f}{RBW}\right) \tag{10}$$

Here, p_n represents the power spectral density at each frequency point, f_n, RBW is the resolution bandwidth of the spectrum analyzer, and Δf is the frequency step, which is the spacing between each pair of frequency points. If the spacing is uniform, Δf is given by:

$$\Delta f|_{\mathrm{GHz}} = \frac{f_N - f_1}{N} \tag{11}$$

Once the total signal power is calculated from the spectrum, the offset can be determined using:

$$P_{offset}|_{\mathrm{dB}} = P_{tot}|_{\mathrm{dBm}} - P_{spectrum}|_{\mathrm{dBm}} \tag{12}$$

In this equation, P_{tot} is the signal power indicated in the measurements (stored in the parameters *TOT_Power_IN* and *TOT_Power_OUT*). Finally, the offset is applied to each element of the transmitted and received spectrum power arrays, namely *spectrum_TX_power* and *spectrum_RX_power* arrays. Note that there is a single offset for the input power spectrum, while for the output power there is a distinct offset for each unique pair of target gain and tilt.

The first step in processing the measurement data is to identify the central frequency of each active channel in the spectrum. This is done by using a filtered version of the power spectrum to minimize data fluctuations and eliminate erroneous samples. The filtering process involves clamping the spectrum values within a reasonable range to remove outliers and convolving the spectrum with a constant function. The convolution parameters are fine-tuned to ensure they work well across all devices. The central frequencies of the active channels are identified using the *find_peaks* function from *SciPy*, which locates the indices of all local maxima in the spectrum [27].

To address issues caused by slight variations in channel central frequencies, these frequencies are determined exclusively from the input power spectrum and then applied to the output power spectra. Although it is possible to use prior knowledge of channel central frequencies, which is typically available in test setups, this program is designed to be flexible. It aims to function with minimal dependence on external information. Consequently, the program does not require any input regarding the number of active channels or their central frequencies.

3.4. SSE and ASE Evaluation

Source Spontaneous Emission (SSE) and Amplified Spontaneous Emission (ASE) are essential parameters for evaluating the noise figure, representing the noise floor in the input and output power spectra, respectively. Although named after spontaneous emission, these values also include other minor noise contributions, which are ignored for simplicity [28]. The process of determining the noise floor level is depicted in Figure 4. This approach involves identifying points between each pair of adjacent channels where the power value equals the noise floor. Then, the noise values at frequencies corresponding to the center frequencies of active channels are derived through linear interpolation. For the first and last channels, the first and last frequency points are used accordingly.

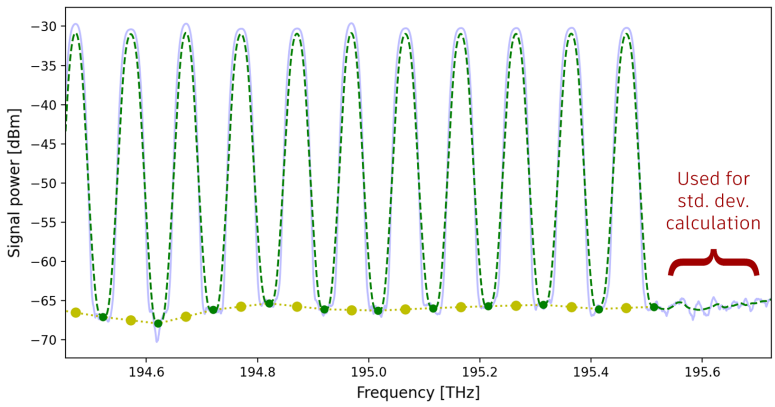


Figure 4. Evaluation of SSE: The power density spectrum is convolved to obtain a smooth curve (green line). The middle points between two adjacent channels (green dots) are used for piecewise linear interpolation to evaluate the noise at the channel frequencies (yellow dots).

In many datasets, the input power values show a noisy, wavy curve for the noise floor, which can be attributed to the sensitivity of the optical spectrum analyzer, corresponding to a noise floor of −65 dBm in the optical spectrum analyzer setting for these measurements. Therefore, an additional convolution step is performed for data cleaning. This step follows the trend of the curve while avoiding points that correspond to the local maxima and minima of the noise. SSE and ASE are used to compute the $P_{ASE_{AMP}}$ parameter shown in Equation (9). This value is the difference between the ASE noise and the amplified SSE noise and can be evaluated as:

$$P_{ASE_{AMP}}|_{\text{dBm}} = 10\log_{10}\left(10^{\frac{P_{ASE,\text{dBm}}}{10}} - 10^{\frac{P_{SSE,\text{dBm}}+G}{10}}\right) \tag{13}$$

This process is the mathematical formulation of the quantity shown in Figure 2.

3.5. Noise Figure Dataset

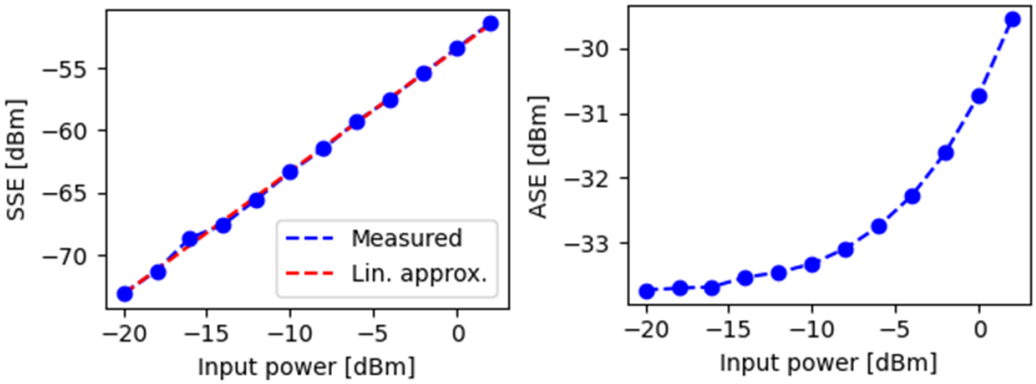
Once all necessary parameters are obtained, a dataset is created for each device. Additionally, two distinct datasets are generated for the low and high-gain ranges. This separation is crucial because the EDFAs examined in this study are designed to optimize the noise figure independently within each range, leading to different behaviors. Each dataset contains the evaluated noise figure of the device for every unique combination of total input power, target gain and tilt, and channel frequency. These datasets are intended for use in polynomial regression analysis. Each dataset consists of tens of thousands of entries, with a total storage size of a few megabytes each. To ensure both efficiency and readability, each dataset is saved in two formats: one optimized for efficient storage and I/O operations, and another in a human-readable format.

3.6. Polynomial Regression

In the earlier sections, the noise figure was assessed based on four input parameters. This analysis delves into how the noise figure changes with each parameter individually, providing a rationale for employing a polynomial structure. Subsequently, the suitable degree for each of the four polynomials will be determined. The coefficients will be calculated using the GLS algorithm. Each model is built by integrating datasets from various devices of the same brand, type, and gain range.

3.6.1. Noise Figure vs. Input Power

From the formula shown in Equation (9), it is clear that the noise figure is mainly influenced by three variables: $P_{ASE_{AMP}}$, G, and f. The input power primarily impacts the noise levels, including ASE and SSE values, while channel gain is less affected by this parameter. SSE represents the noise impacting the input signal before it is amplified. As shown in Figure 5a, the SSE value, measured for a commercial EDFA in dBm, exhibits a linear relationship with the input power. This linearity is expected, given that all measurements were performed using a test signal with a consistent SNR, which remains stable throughout the measurements. In contrast, ASE behaves differently. At higher input powers, it follows a linear trend similar to SSE. However, at lower input powers it levels off to a minimum floor. This behavior is due to additional noise contributions that are independent of the input power, as well as the RBW of the OSA used for the measurements. This characteristic pattern is illustrated in Figure 5b.

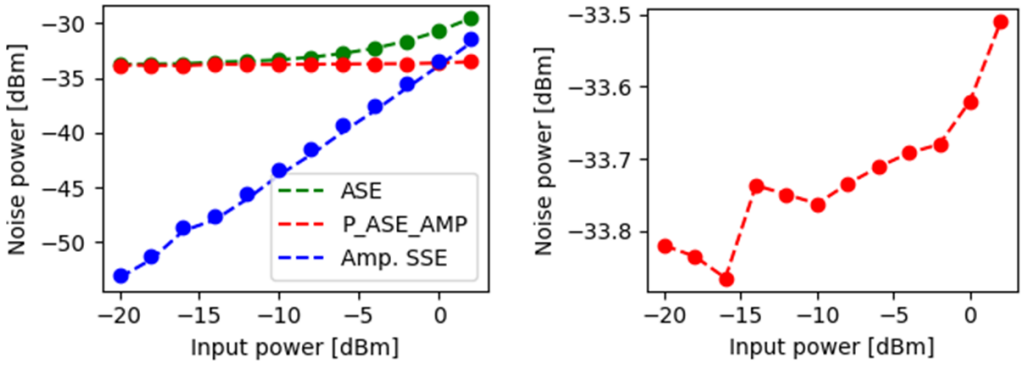


(**a**) (**b**)

Figure 5. Noise levels vs. input power for a commercial EDFA. (**a**) SSE; (**b**) ASE.

The behavior of the parameter $P_{ASE_{AMP}}$, which integrates the other two noise contributions, is depicted in Figure 6. According to Equation (13), at low input powers, the amplified SSE noise is roughly 12 dB lower than ASE, making $P_{ASE_{AMP}}$ nearly equal to the ASE level. As the input power increases, the gap between the two parameters narrows, causing the SSE contribution to become more significant in the formula. The curve of $P_{ASE_{AMP}}$ remains remarkably stable with respect to input power, exhibiting a maximum variation of around 0.5 dB across the entire input power range. Figure 7 shows the noise figure results as a function of input power for various values of target gain, tilt, and frequency.

It is clear that the noise figure tends to increase slightly with input power when a low target gain is set, but this trend reverses under high-gain settings. This shift can be attributed to the greater influence of SSE on $P_{ASE_{AMP}}$ at higher gain, resulting in a negative trend due to the steeper slope of SSE compared to ASE. Conversely, at low gain, $P_{ASE_{AMP}}$ closely follows the positive trend of ASE because the SSE contribution is minimal. A forth-degree polynomial is deemed appropriate for modeling the variations of the noise figure with respect to input power.



(**a**) (**b**)

Figure 6. $P_{ASE_{AMP}}$ vs. input power for a commercial EDFA. (**a**) $P_{ASE_{AMP}}$, ASE and amplified SSE; (**b**) $P_{ASE_{AMP}}$ only.

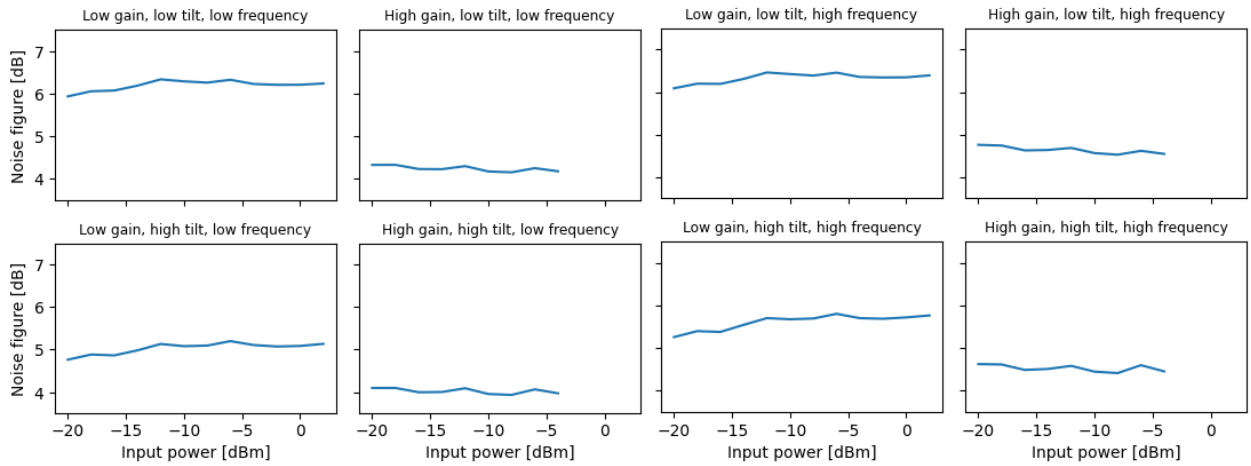
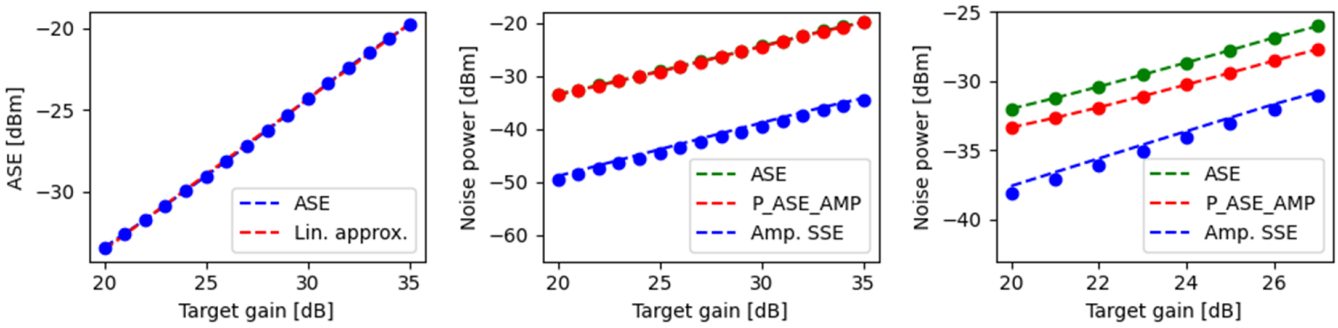


Figure 7. Noise figure vs. input power, at various input parameter configurations.

3.6.2. Noise Figure vs. Gain

The target gain is a crucial parameter of an EDFA. As shown in Equations (9) and (13), channel gain significantly affects the noise figure. It appears explicitly in the formulas and also influences the value of $P_{ASE_{AMP}}$. Specifically, SSE remains constant with respect to channel gain since it depends solely on input signal parameters and not on the amplifier's characteristics. In contrast, ASE noise exhibits a linear increase (in dBm units, which implies exponential growth) with respect to gain, as expected. This behavior is depicted in Figure 8a, which also includes a linear fit of the curve.

In Figure 8b,c it is possible to observe that both ASE and amplified SSE exhibit linear behaviors with respect to gain, albeit with similar slopes. Consequently, $P_{ASE_{AMP}}$ also demonstrates a linear behavior, with values lying between ASE and amplified SSE. At lower input power levels, the curve closely aligns with ASE, reflecting the significantly lower level of amplified SSE. As input power increases, the impact of $P_{ASE_{AMP}}$ becomes more pronounced.



(**a**) (**b**) (**c**)

Figure 8. (**a**) ASE vs. gain, with linear fitting; (**b**) $P_{ASE_{AMP}}$ vs. channel gain at low input power and (**c**) at high input power.

In all test scenarios, $P_{ASE_{AMP}}$ consistently rises with channel gain, with its linear approximation slope always below 1, nearing this limit as gain increases. As a result, the difference in $P_{ASE_{AMP}} - G$, as indicated in Equation (9), decreases as channel gain rises. Given that the third term in the equation remains constant with respect to gain (depending solely on frequency), we can deduce that the noise figure will exhibit a similar trend, showing lower values at higher target gains. These findings are depicted in Figure 9. The noise figure declines more sharply at lower gains because $P_{ASE_{AMP}}$ changes less with gain,

but the slope flattens as gain increases. A third-degree polynomial effectively models the tested curves, resulting in a very low Root Mean Square Error (RMSE).

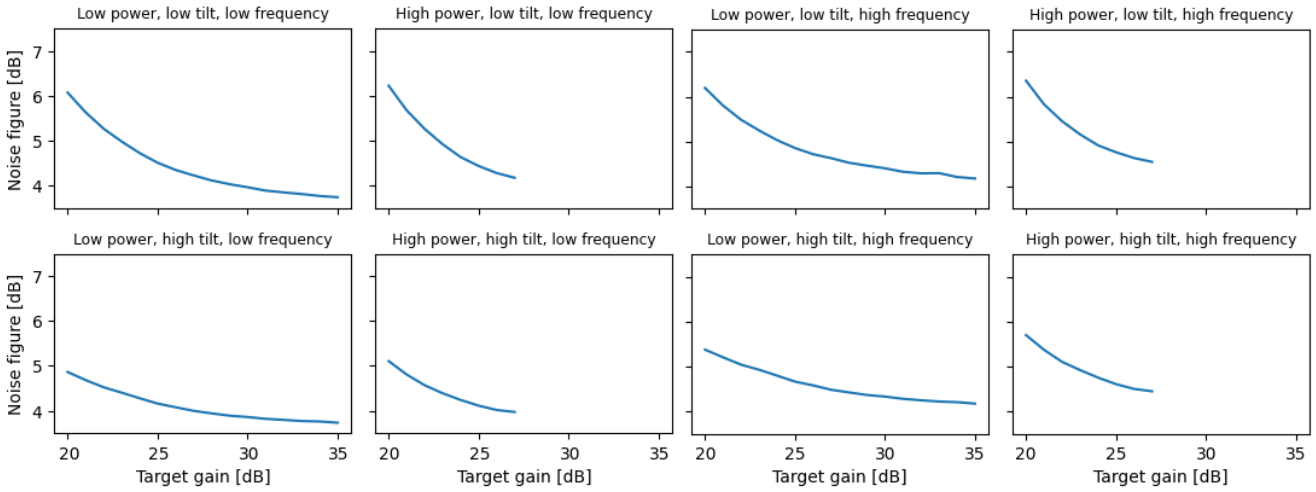


Figure 9. Noise Figure vs. channel gain, at various input parameter configurations.

3.6.3. Noise Figure vs. Tilt

In EDFAs, the gain *tilt* refers to the slope of the gain curve relative to frequency. This parameter can be adjusted to address wavelength-dependent gain variations, which cause different power levels among channels traveling through the fiber at various frequencies. Factors contributing to this include the amplifier's limited gain bandwidth and potential gain saturation. Tilt control enables the shaping of the amplifier's gain profile, helping to mitigate these unwanted effects and achieve more uniform amplification across the frequency spectrum. The gain in an EDFA can be modeled as follows [11]:

$$G(f) = G + \frac{T}{B}(f - f_0) + g(f) \tag{14}$$

where G is the target gain, T is the target tilt, B is the amplifier bandwidth, f_0 is the center frequency, and $g(f)$ represents the gain ripple.

Tilt primarily affects channel gain, and its influence varies with frequency. As a result, different outcomes can be expected for low and high channel indices. As shown in Figure 10, the noise figure generally decreases as tilt increases. This effect is more pronounced at lower target gains and diminishes at higher gains. In the previous discussion, it was noted that higher gain typically results in a lower noise figure. For lower channel frequencies, gain increases with increasing tilt, whereas, for higher channel frequencies, the opposite occurs. Typically, the target gain value is achieved with tilt values close to 0 or slightly negative. This trend also applies to $P_{ASE_{AMP}}$. The variation in channel gain due to tilt remains consistent across different input powers, target gains, and channel frequencies. However, $P_{ASE_{AMP}}$ shows a more negative slope with respect to tilt when the target gain is low. This results in minimal variation for low frequencies, where the positive slope is flattened, and a more pronounced negative variation for high frequencies, where the slope is already negative. Therefore, the difference with gain is more distinct at low gains, while the contributions tend to balance each other more at higher gains. For this reason, a third-degree polynomial was found to be an appropriate fitting function for the noise figure's dependency on target tilt.

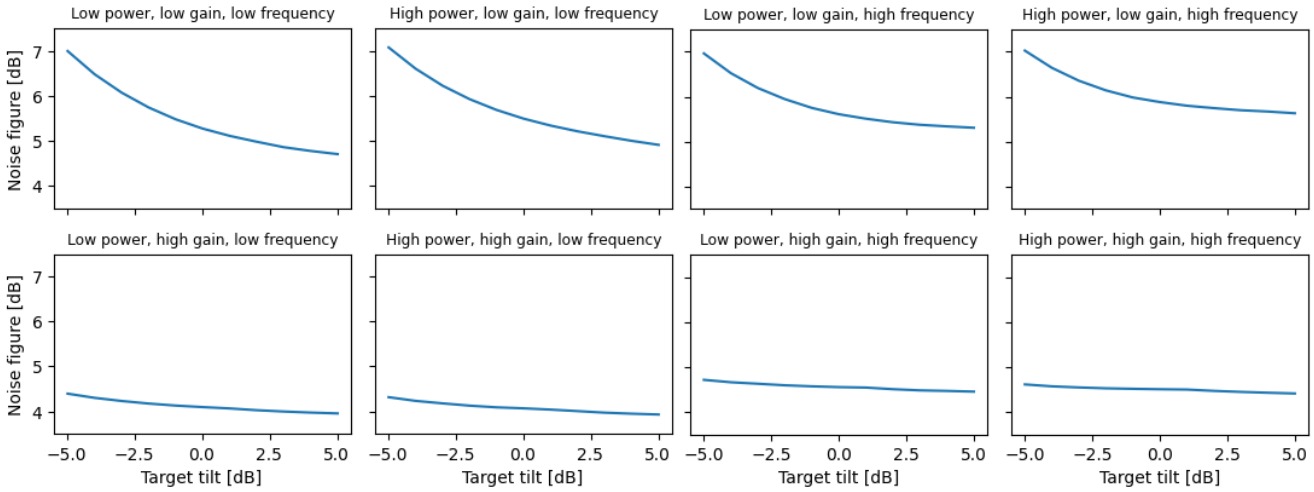


Figure 10. Noise figure vs. target gain tilt, at various input parameter configurations.

3.6.4. Noise Figure vs. Frequency

Finally, the impact of channel frequency on the noise figure is evaluated. As indicated in Equation (9), frequency explicitly affects the noise figure. Additionally, from Equation (14), it is clear that frequency influences channel gain. Unlike other cases, the term $10\log_{10}(hfB_0)$, which was previously a constant, now represents an explicit dependency on frequency. However, this term shows nearly negligible variations relative to frequency. To assess the maximum excursion of this term within the frequency range of interest, we can use a differential approximation:

$$\Delta NF = 10\,log_{10}\,(hf_{MAX}\,B_0) - 10\,log_{10}\,(hf_{MIN}\,B_0) \tag{15}$$

$$\Delta NF = \Delta f \cdot \frac{\delta}{\delta f}(10\,log_{10}\,(hf_{MID}\,B_0)),\ f_{MID} = \frac{f_{MIN} + f_{MAX}}{2} \tag{16}$$

$$\Delta NF = (f_{MAX} - f_{MIN})\frac{10}{f_{MID}\,ln(10)} \tag{17}$$

By substituting values corresponding to the C-band for f_{MIN} and f_{MAX}, the resultant change is approximately 0.08 dB. Given that the noise figure typically exceeds 4 dB, this corresponds to a relative contribution of, at most, 2%. Therefore, this term alone does not significantly influence the noise figure's trend as a function of frequency. As discussed earlier, when the target tilt is positive, channel gain decreases with increasing channel frequency, and the reverse is true for negative tilt. ASE noise exhibits a similar behavior, forming a plateau in the middle for positive tilts and varying more at the band edges. SSE tends to be noisier at lower power levels, likely due to the instrument's sensitivity. $P_{ASE_{AMP}}$ follows a pattern similar to channel gain, meaning the resulting noise figure will be influenced by the specific slopes and variations of each parameter.

The results are summarized in Figure 11. It is immediately apparent that the noise figure's variation with frequency shows significantly greater variance compared to other parameters, making it more challenging to fit into a polynomial curve. This behavior is detailed in Figure 12. One advantage is that, unlike other models, overfitting is not a concern because the C-band remains constant, so the model does not need to predict accurately outside the tested frequency range. Therefore, increasing the polynomial degree to improve accuracy is a viable approach. The only consideration is that raising the polynomial's degree requires keeping the degrees of other polynomials low to maintain a manageable dataset size and ensure short training times for our models. An eighth-degree polynomial

was found to strike the best balance between training speed, numerical precision, and model accuracy.

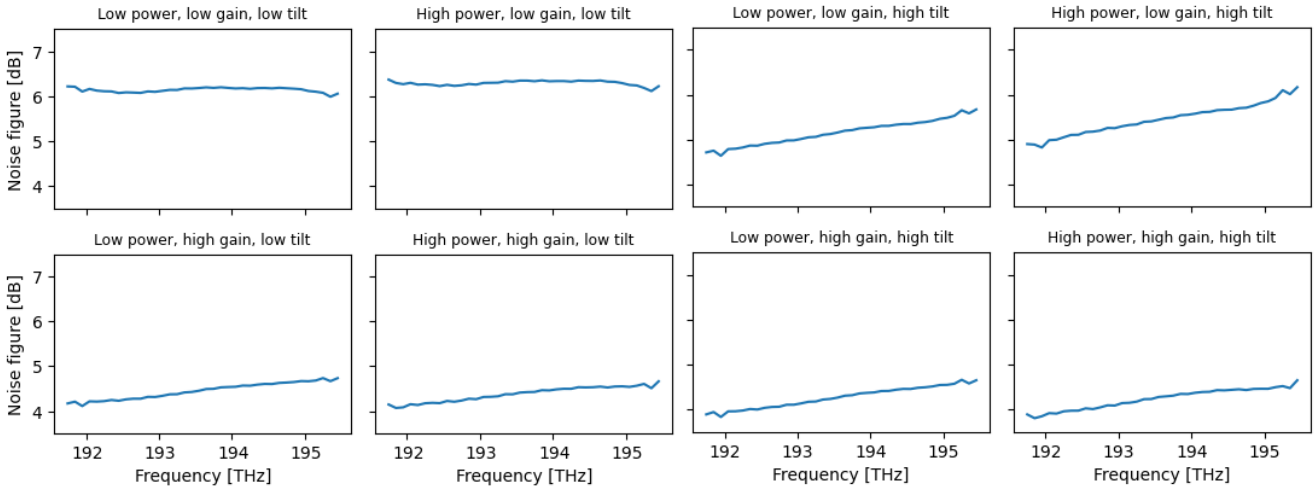


Figure 11. Noise figure vs. channel frequency, at various input parameter configurations.

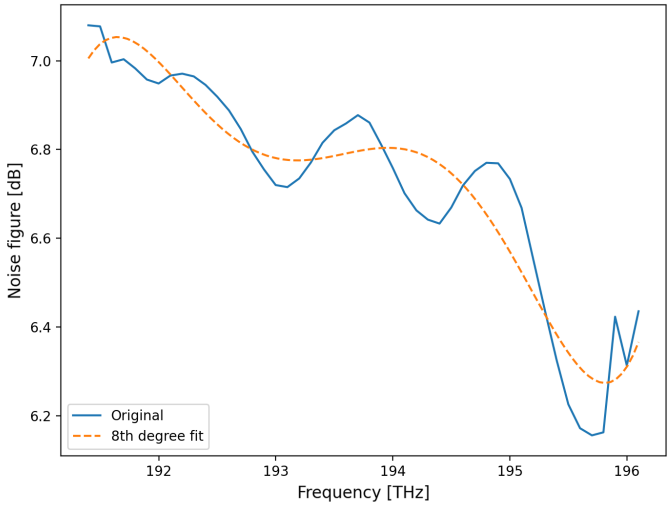


Figure 12. An example of the significant variance in noise figure versus frequency under a worst-case scenario for a commercial EDFA.

3.6.5. Generalized Least Squares Regression

The initial phase of the analysis involved generating multiple datasets for each device within each gain range. Each dataset includes the noise figure values for every unique combination of input power, target gain, tilt, and channel index. The current goal is to develop a mathematical model that characterizes each type of EDFA under examination. The polynomial model can be mathematically expressed as follows:

$$NF(P, G, T, f) = \sum_{a=0}^{4} \sum_{b=0}^{3} \sum_{c=0}^{3} \sum_{d=0}^{8} m_{abcd} P^a G^b T^c f^d \quad (18)$$

where m_{abcd} represents the coefficient of each monomial. The goal is to determine the polynomial coefficients using the Generalized Least Squares (GLS) method, an algorithm for estimating the parameters of a linear regression model. To perform polynomial regression, a new dataset is required. This dataset will also include a column for the noise figure and an additional column for each monomial. The GLS method requires a training set to construct our model and a test set to evaluate its performance. For each device within each

gain range, a model is developed by combining multiple datasets from different devices of the same brand and model to improve accuracy.

Two types of analysis will be performed. Initially, the models will be created using a 70% training and a 30% test split to evaluate their performance. Subsequently, the models will be trained on the entire datasets, producing the final models to be used in conjunction with other models on different datasets.

3.7. Notes on Code and Algorithms

Although the entire model creation process was carried out using Python programs and libraries, the results are represented by coefficients for a multivariate polynomial, stored in column-oriented *.parquet* files. This means that the noise figure evaluation can be performed using any programming language or tool capable of reading these files and performing basic mathematical operations. This ensures excellent portability for the model, offering maximum flexibility in creating an interface. A Python interface was chosen for the model due to the language's widespread use and its seamless integration with powerful libraries such as *Pandas* and *NumPy* [29]. This interface is implemented as a Python class, providing attributes and methods to enhance ease of use and allowing integration into more comprehensive EDFA models. The models are named using the pattern **X_Y_Z**, where X represents the device brand and specific model, Y denotes the gain range, and Z indicates the number of active channels in the training datasets of the model. This number is crucial for two reasons. First, for more accurate results, it is ideal to select a model trained with a spectral load similar to the one being estimated, to mitigate potential effects from different channel configurations. Second, the input parameter that the model accepts is the total power of all active channels. Therefore, when estimating the noise figure for a spectrum with a different number of channels, this difference needs to be considered. It is important to remember that the input power parameter refers to the total power of all active channels. Therefore, a conversion is necessary if the data being used correspond to a different number of channels:

$$P_{IN_{ADJ}} = P_{IN} - 10\log_{10}(N_{current}) + 10\log_{10}(N_{model}) \tag{19}$$

where $N_{current}$ and N_{model} represent the numbers of active channels in the configuration to be estimated and in the model training datasets, respectively.

3.8. Polynomial Evaluation: Horner's Rule

As previously mentioned, speed is a critical factor for this model, as it is intended to be integrated into more complex EDFA or network models. Therefore, optimizing the computation speed of polynomial evaluation is essential. To achieve this, we employ Horner's method, an algorithm designed to minimize the number of operations required for polynomial evaluation by computing it in a nested form. This method can be generalized for any polynomial [23].

$$\begin{aligned} p(x) = a_n x^n + a_{n-1}x^{n-1} + \ldots + a_1 x + a_0 \\ = a_0 + x(a_1 + x(a_2 + \ldots + x(a_{n-1} + a_n x)\ldots)) \end{aligned} \tag{20}$$

In this format, an n_{th} degree polynomial can be evaluated with just n multiplications and n additions. While any polynomial can be evaluated using this algorithm, and a general function for this computation could be easily created, Python loops are known to be relatively slow, which is common for interpreted languages. To further speed up the computation, the entire polynomial expression is unrolled into Horner's form and directly implemented within the interface library.

4. Results and Discussion

This section provides a summary of the results concerning the absolute and relative errors observed when estimating the noise figure using polynomial models, compared to values derived from the datasets. It also details the computation times for noise figure calculations from the datasets and the creation of the model itself. First, we consider the uncertainty associated with OSA measurements to set realistic expectations for the model's accuracy. Then, we evaluate the performance of each model in two scenarios: initially using a 70%/30% split of the datasets, and subsequently with complete datasets for broader application. We further analyze the relationship between dataset size and accuracy. This assessment aims to determine whether or not further improvements are possible, if accuracy has reached its peak, or if it is feasible to reduce the number or size of datasets while maintaining similar accuracy levels. Next, we evaluate performance metrics such as model speed and memory usage. Finally, we conduct a test where this model is integrated into a deep-learning framework designed to estimate the power spectral density at the output of an EDFA, based on configuration parameters and input PSD.

4.1. Uncertainty of OSA Measurements

The measurements provided by the OSA come with a certain degree of uncertainty, which sets the lower limit for model accuracy. Striving for a model that achieves a maximum error lower than that of the measurements is impractical, as it would mean accurately estimating an inherently inaccurate value. Instead, it is wise to estimate this uncertainty and aim for similar accuracy levels with the models.

One contribution to the uncertainty is the sensitivity of the total input and output power measurements. OSAs represent those values with one decimal place, so the absolute uncertainty due to quantization of P_{IN} and P_{OUT} is:

$$\delta P = \frac{0.1\,\text{dBm}}{2} = 0.05\,\text{dBm} \tag{21}$$

This variation in both P_{IN} and P_{OUT} results in a noise figure error of up to 0.1 dB, representing the absolute minimum uncertainty, assuming that all other quantities are obtained with ideal infinite accuracy. To establish a benchmark for the experimental error in measurements, four distinct datasets from identical devices are analyzed. The commercial devices under test are *EDFA-35* from Cisco in the low-gain range [30]. Measurements are compared for identical values of total input power, gain, tilt, and channel index, and the maximum difference among them is evaluated. Figure 13 shows the cumulative distribution of measurement errors. In this graph, the y-axis represents the percentage of dataset measurements with errors within the value indicated on the x-axis. A curve that shifts more to the left indicates better performance, as it means a larger portion of the dataset has errors lower than those shown on the x-axis.

From this figure, it is clear that 90% of the measurements for Cisco EDFA-35 devices working in the low-gain range condition have errors within 0.149 dBm, while an absolute error of 0.247 dBm ensures that 99% of the measurement errors are safely included. These two error values will be referred to as the 90%-threshold and the 99%-threshold in later sections. Therefore, it is reasonable to select 0.25 dBm as a plausible lower limit for model accuracy. However, this value may vary for different devices due to its experimental nature. This worst-case threshold for 99% of the measurements highlights the superior accuracy of our proposed polynomial approach with respect to other models in the literature. Specifically, we refer to [31], where a Gradient Boosting Regression (GBR) approach was used for THz applications, achieving a Mean Absolute Error (MAE) of 0.3428 in fitting the predicted noise figure values.

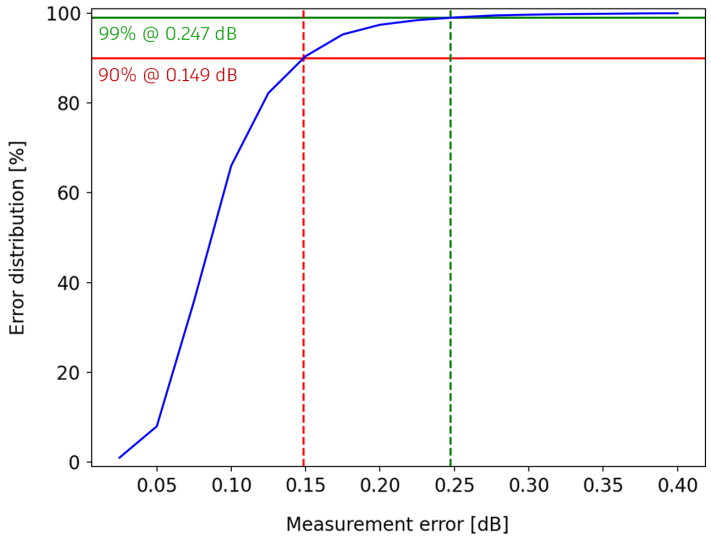


Figure 13. Cumulative distribution of measurement errors for Cisco EDFA-35 devices operating in the low-gain range (blue) and 90%-threshold (red), 99%-threshold (green) error values evaluation.

4.2. 70–30 Split Ratio Models

These initial models are developed by combining all datasets from various devices of the same brand, model, and gain range. The combined datasets are then split into a training set, used for model construction, and a test set, used to evaluate result accuracy. Six different models are created:

1. Cisco EDFA-17 [32], low-gain range, 1 dataset;
2. Cisco EDFA-35 [30], high-gain range, 2 datasets;
3. Cisco EDFA-35, low-gain range, 4 datasets;
4. Cisco L-band EDFA [33], low-gain range, 6 datasets;
5. Juniper EDFA [34], high-gain range, 4 datasets;
6. Juniper EDFA, low-gain range, 4 datasets.

The training and test sets are divided using a 70%/30% ratio, following common practice in machine-learning applications [35]. The results for the cumulative distributions of the absolute and relative errors are shown in Figure 14.

The EDFA-17 model demonstrates superior performance since it relies on data from a single device, thereby avoiding potential errors introduced by manufacturing variations and resulting in higher overall measurement correlation. If additional datasets were available for this model, its performance might be negatively impacted, but unfortunately, none were available. For absolute errors, the 90%-threshold ranges between 0.05 and 0.28 dB, depending on the model, while the 99%-threshold ranges between 0.10 and 0.46 dB. For relative errors, the 90%-threshold ranges between 1.19 and 4.83%, depending on the model, while the 99%-threshold ranges between 2.08 and 7.19%. The maximum values for absolute and relative error are 0.81 dB and 10.56%, both originating from the Juniper devices in the high-gain range.

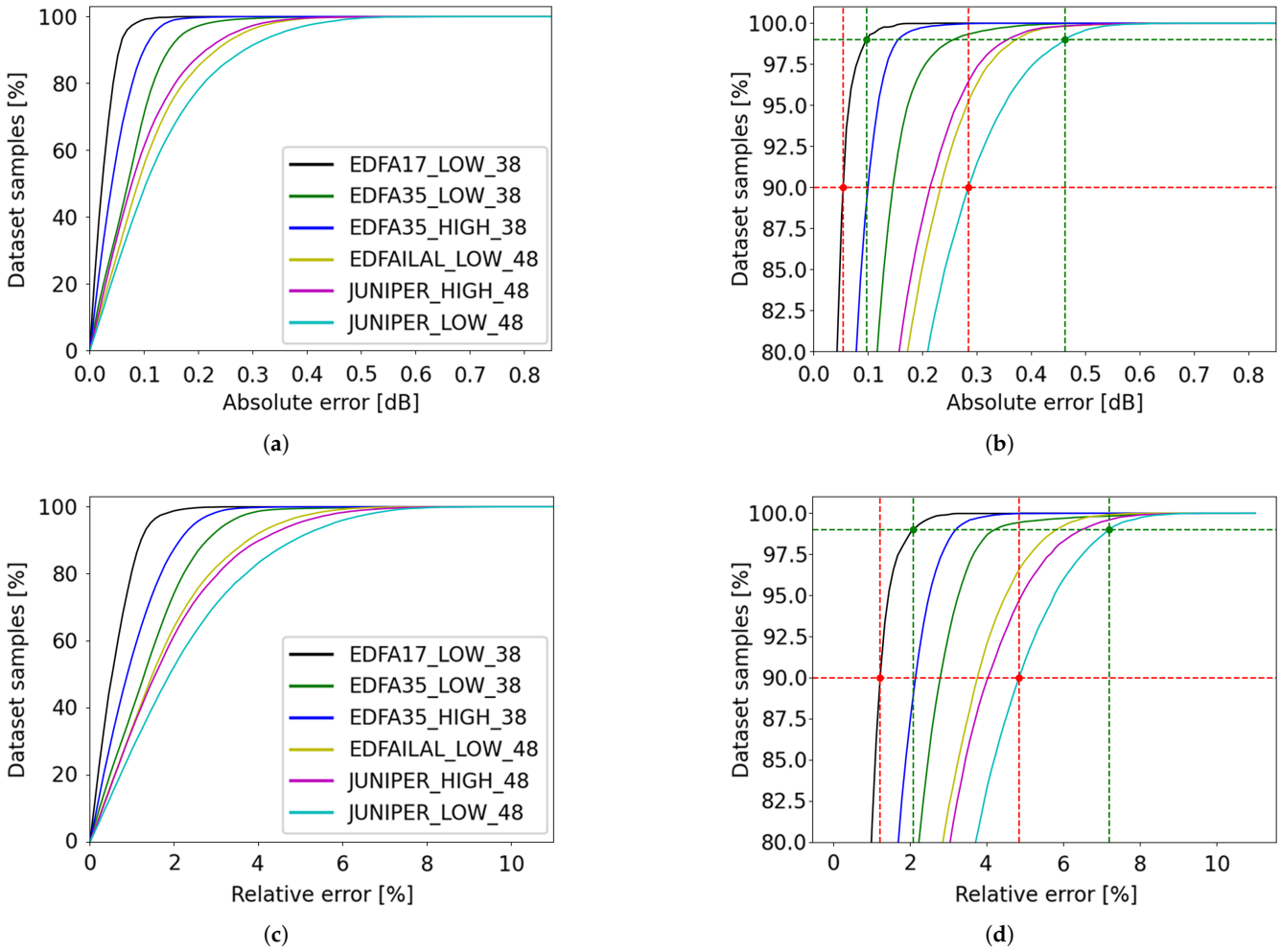


Figure 14. Cumulative distribution of absolute and relative errors of 70–30 split models. (**a**) Absolute errors; (**b**) Absolute errors, detail; (**c**) Relative errors; (**d**) Relative errors, detail. In (**b**,**d**) green and red dashed lines represent 99% and 90%-threshold values, respectively, with green and red bullets highlighting minimum and maximum values obtained through these curves.

4.3. Full Dataset Models

The same models are then created using the entire datasets for training, rather than a random 70% of their entries. The test set is composed of the average of all datasets for each device. This approach aims to identify errors specific to the model functions, such as the polynomial fits, and to evaluate whether a polynomial fit is appropriate for this model or if there is potential for improvement in the regression function.

The models created are the same as in the previous case in terms of brand, gain range, and number of datasets. The results are shown in Figure 15. For absolute errors, the 90%-threshold varies between 0.05 and 0.11 dB, depending on the model, while the 99%-threshold ranges from 0.09 to 0.23 dB. In terms of relative errors, the 90%-threshold spans from 1.01 to 1.74%, while the 99%-threshold falls within 1.79 to 3.27%, depending on the model. The highest observed absolute and relative errors are 0.42 dB and 7.06%, respectively, both recorded for Juniper devices operating in the low-gain range. Notably, despite being trained on a single dataset, the EDFA-17 exhibits higher relative errors than the EDFA-35 in the low-gain range. The error contribution from the model is approximately on par with the measurement uncertainties within the dataset, as illustrated in Figure 13. This indicates that the polynomial form used for the fitting function introduces an error level

comparable to the inherent measurement uncertainty. Therefore, modeling the noise figure as a polynomial function of the four input parameters is a valid and effective approach.

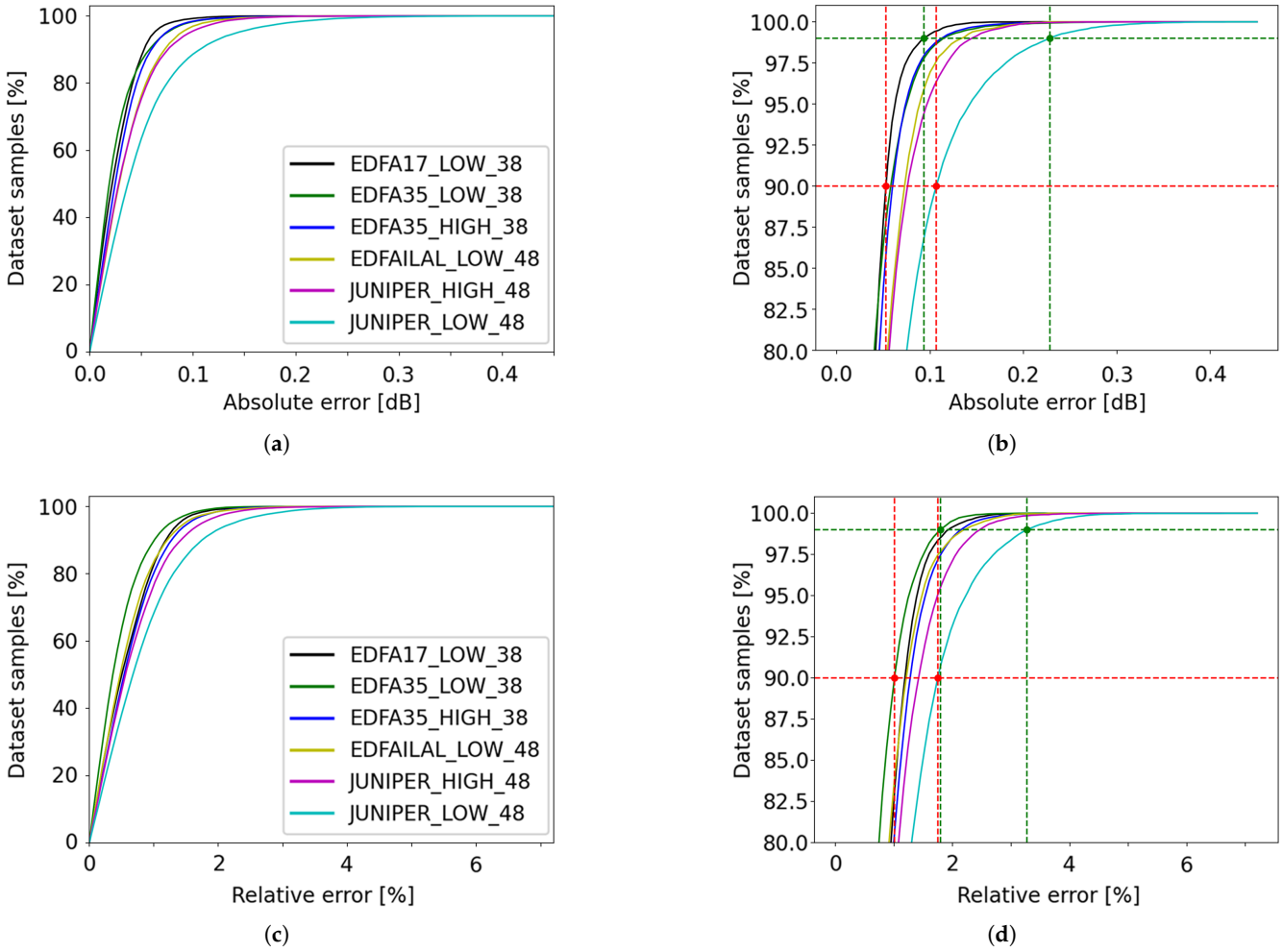


Figure 15. Cumulative distribution of absolute and relative errors of full dataset models. (**a**) Absolute errors; (**b**) Absolute errors, detail; (**c**) Relative errors; (**d**) Relative errors, detail. In (**b**,**d**) green and red dashed lines represent 99% and 90%-threshold values, respectively, with green and red bullets highlighting minimum and maximum values obtained through these curves.

The results obtained for the prediction of the noise figure are comparable to those achieved using machine-learning approaches for EDFA noise figure prediction. Specifically, from the literature we can observe that an Artificial Neural Network (ANN) approach yields a Mean Absolute Error (MAE) of 0.1 dB [36], while a deep neural network (DNN) approach achieves a MAE of 0.2 dB [37]. Both approaches, however, exhibit the common limitations of machine-learning techniques, such as the need for large datasets (4000 points and 45,000 points, respectively) and long training times (5000 epochs for each model). Additionally, the results are also comparable to those obtained using a Gradient Boosting Regression (GBR) approach, which produces an MAE of 0.3428 dB [31]. These comparisons highlight the effectiveness of the proposed method in providing competitive noise figure predictions while addressing typical challenges faced by machine-learning models.

4.4. Correlation of Errors

An important aspect of model error analysis is examining the potential correlation between input configurations and the magnitude of absolute or relative errors. This is particularly relevant because, if the largest errors are concentrated in specific input patterns, the model's overall accuracy could be overestimated. A detailed understanding of error distribution allows for the implementation of varying error tolerances across amplifier models based on their respective operating ranges. To address this, the mean absolute error of the noise figure is analyzed as a function of each input parameter, considering the cumulative distribution of absolute errors discussed in previous sections. By plotting absolute error against each parameter, distinct trends and behaviors across different device brands can be identified. A notable trend observed across nearly all devices is the increase in error near the frequency band edges. As illustrated in Figure 16, restricting the analysis to central channels results in lower overall errors. This effect may stem from overfitting, as polynomial models typically exhibit reduced accuracy at the boundaries of their fitting intervals.

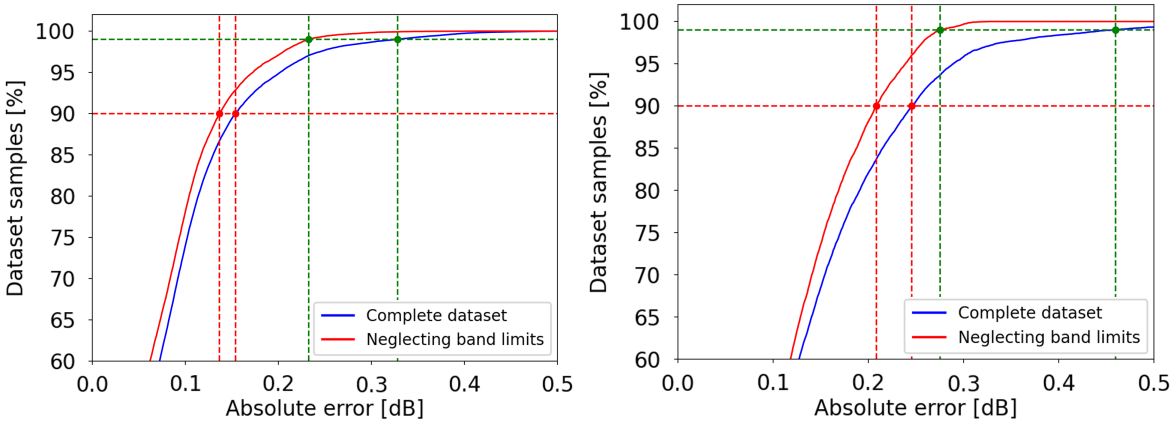


Figure 16. Cumulative distribution of absolute errors and their correlation with the frequency range. (**a**) EDFA-35, low-gain range; (**b**) L-band EDFA, low-gain range. Green and red dashed lines represent 99% and 90%-threshold values, respectively, with green and red bullets highlighting minimum and maximum values obtained through these curves.

The total input power seems to have a negligible impact on the error distribution, with varying trends observed across datasets of the same device type. However, Cisco EDFA-17, EDFA-35, and Juniper devices consistently exhibit lower errors at higher target gain values, as shown in Figure 17. Juniper devices also exhibit a reduction in errors as the target tilt increases, as depicted in Figure 18. In summary, the noise figure model demonstrates varying degrees of error across different input data intervals. These variations can be quite substantial, as shown in Figure 16b, where the 99%-threshold error decreases from 0.46 to 0.28 dB, and in Figure 17c, where it drops from 0.43 to 0.27 dB. However, these intervals are highly device-dependent, preventing the establishment of a universal guideline for narrowing the input parameter range to minimize errors, except for the consistent influence of channel frequency, which remains a common trend across all devices.

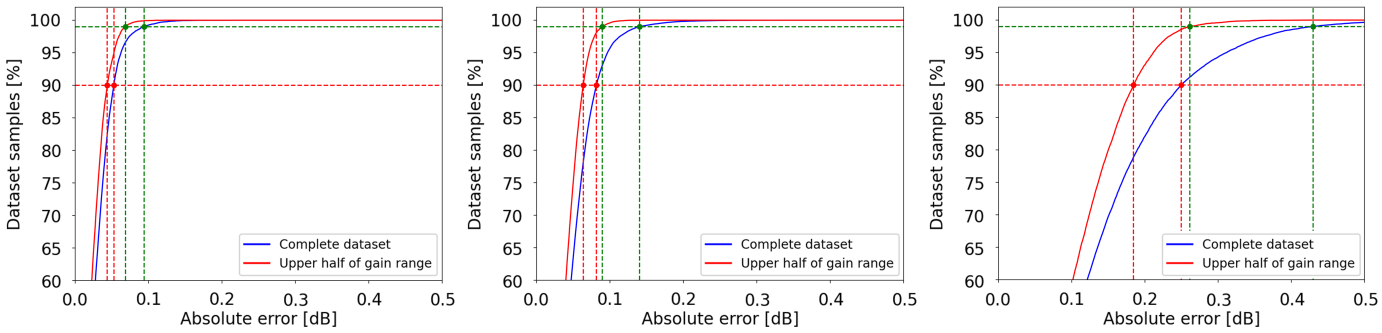


Figure 17. Cumulative distribution of absolute errors and their correlation with gain and tilt range: (**a**) EDFA-17, low-gain range; (**b**) EDFA-35, high-gain range; (**c**) Juniper, low-gain range. Green and red dashed lines represent 99% and 90%-threshold values, respectively, with green and red bullets highlighting minimum and maximum values obtained through these curves.

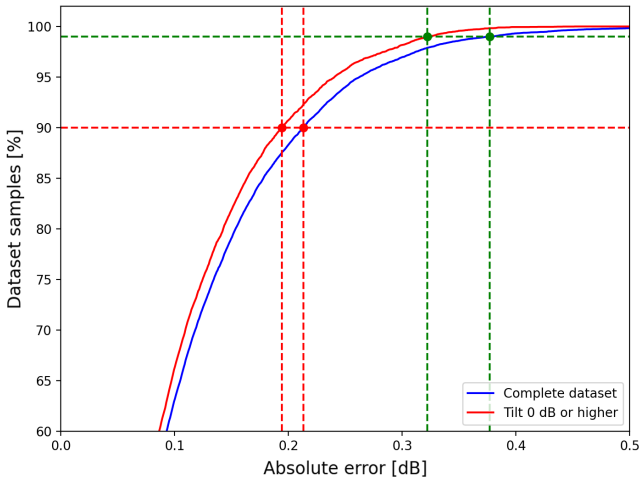


Figure 18. Correlation between tilt range and error reduction in JUNIPER devices. Green and red dashed lines represent 99% and 90%-threshold values, respectively, with green and red bullets highlighting minimum and maximum values obtained through these curves.

4.5. Code Execution Performance

All experiments are carried out on a system featuring an Intel Core i7-1250U processor. Initially, we evaluate the time required to compute the noise figure from Matlab datasets and store the results in *.parquet* files. Then, we analyze the time taken to generate the device models.

4.5.1. Converting Matlab Files to Noise Figure Datasets

The first segment of the code follows these steps:

1. Load each *.mat* EDFA dataset, which contains the input and output spectral power densities of the amplifier as a function of total input power, target gain, and tilt.
2. Execute all the processing steps outlined in Section 3.
3. Store the results in *.parquet* files to enhance speed and reduce storage requirements.

The execution time of the program is summarized in Table 1. Each row, representing a unique combination of total input power, gain, tilt, and channel index within a dataset, is processed in approximately 0.60 ms. This corresponds to a processing rate of approximately 1667 rows per second or 100,000 rows per minute. While this provides a general estimate of computation time based on dataset size, it is important to note that the actual

performance is machine-dependent and should be experimentally verified for different hardware configurations.

Table 1. Computation time to compute the noise figure for each EDFA dataset.

Device	Gain	Dataset No.	Time [s]	Row Count	Time per Row [ms]
EDFA-17	LOW	1	13.22	21,318	0.62
EDFA-35	HIGH	1	32.95	56,848	0.58
EDFA-35	HIGH	2	31.99	56,848	0.56
EDFA-35	LOW	1	25.79	45,144	0.57
EDFA-35	LOW	2	27.77	46,816	0.59
EDFA-35	LOW	3	25.77	45,144	0.57
EDFA-35	LOW	4	34.54	58,608	0.59
EDFA-ILA-L	LOW	1	12.08	21,360	0.57
EDFA-ILA-L	LOW	2	12.52	21,360	0.59
EDFA-ILA-L	LOW	3	9.03	14,976	0.60
EDFA-ILA-L	LOW	4	9.16	14,976	0.61
EDFA-ILA-L	LOW	5	8.83	14,976	0.59
EDFA-ILA-L	LOW	6	8.66	14,976	0.58
JUNIPER	HIGH	1	10.79	18,816	0.57
JUNIPER	HIGH	2	11.44	18,816	0.61
JUNIPER	HIGH	3	12.15	18,816	0.65
JUNIPER	HIGH	4	11.06	18,480	0.60
JUNIPER	LOW	1	16.15	29,232	0.55
JUNIPER	LOW	2	16.70	29,232	0.57
JUNIPER	LOW	3	17.23	29,232	0.59
JUNIPER	LOW	4	16.90	29,232	0.58

4.5.2. Transforming Noise Figure Datasets into Models

This section assesses the time required to train the polynomial model. The tested program performs the following steps:

1. Combine all datasets containing the noise figure calculations for each EDFA device across the various gain ranges.
2. Construct the training data by calculating the additional dataset columns, as outlined in the GLS regression section.
3. Run the GLS algorithm to determine the polynomial coefficients.
4. Save the coefficients in *.parquet* files.

The execution times for the program across the six different models are presented in Table 2. The computation time remains dependent on the size of the training datasets. The program requires approximately 0.18 ms per dataset row to calculate the model coefficients. When combined with the previous step, it takes around 7.8 s to generate a model from a single `.mat` dataset containing 10^4 combinations of total input power, target gain, tilt, and active channel central frequency. This time increases to 1 min 18 s for 10^5 combinations, and 13 min for 10^6 sets.

Table 2. Computation time for training the noise figure polynomial models.

Device	Gain	Time [s]	Total Row Count	Time per Row [ms]
EDFA-17	LOW	2.70	21,318	0.13
EDFA-35	HIGH	15.04	113,696	0.13
EDFA-35	LOW	35.08	195,712	0.18
EDFA-ILA-L	LOW	16.58	102,624	0.16
JUNIPER	HIGH	12.05	74,928	0.16
JUNIPER	LOW	17.83	116,928	0.15

4.6. Impact of Training Dataset Size on Accuracy

The measurement of power spectral densities required for training this model can be highly time-consuming, primarily due to the extensive number of OSA measurements needed, which directly depends on the number of active channels. As an example, the EDFA-17 dataset in the low-gain range was characterized using the following input parameters:

1. **Total input power**: Ranging from −10 dBm to 6 dBm, with a step size of 2 dBm (9 values);
2. **Gain**: Ranging from 14 dB to 20 dB, with a step size of 1 dB (7 values);
3. **Tilt**: Ranging from −5 dB to 5 dB, with a step size of 1 dB (11 values).

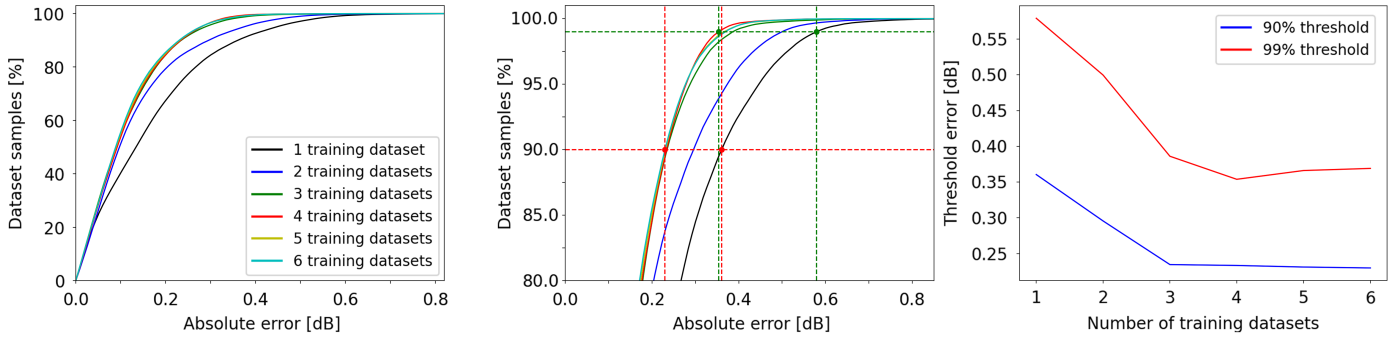
In this scenario, 9 input power spectra and 693 output power spectra are required. When the channel density is high and the spacing between channels is narrow, a lower resolution bandwidth is necessary to accurately capture the noise level between adjacent channels. However, since measurement time is inversely proportional to the square of the resolution bandwidth, this requirement can significantly increase the time required for data acquisition. This underscores the importance of carefully balancing model accuracy with dataset size and acquisition time. As previously mentioned, the model is designed to be adaptable, allowing for trade-offs between precision and the duration of data collection.

To assess performance, two types of analyses are conducted. The first approach trains the model using the complete dataset of one or more devices while evaluating accuracy on the remaining datasets. This represents a worst-case scenario, where the correlation between devices is minimal. The second analysis aggregates all datasets and applies a random train/test split, analyzing accuracy as a function of the split ratio. For these evaluations, the Cisco L-band EDFA datasets will be utilized, as they provide six available datasets.

4.6.1. Accuracy in Function of Number of Datasets

The first analysis explores the model's accuracy as a function of the number of measurement datasets utilized for training. The test results are evaluated by concatenating all device datasets. The outcomes, presented as the cumulative distribution of absolute error, are displayed in Figure 19. Figure 19a,b demonstrate that models trained on one or two datasets show noticeably worse performance compared to the other cases, which exhibit similar curves across the entire error range. The variations are substantial, with the 90%-threshold varying from 0.22 dB to 0.36 dB, and the 99%-threshold ranging from 0.35 dB to 0.58 dB. Figure 19c shows these error thresholds as a function of the number of datasets. It indicates that accuracy improves consistently up to three datasets, after which it stabilizes with only minor gains as the number increases to six. Based on these results, it can be concluded that incorporating data from multiple devices during training is essential

to account for errors introduced by the manufacturing process. However, the number of devices included can remain relatively low.

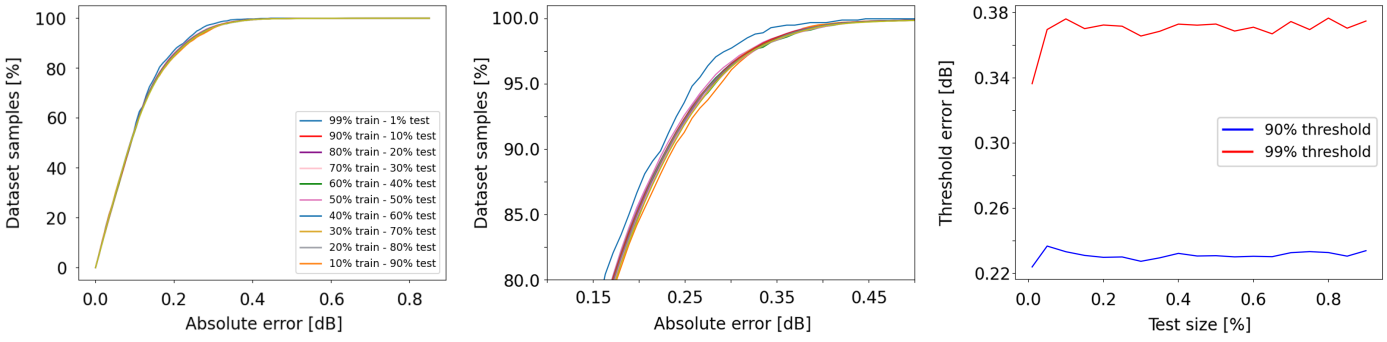


(**a**) (**b**) (**c**)

Figure 19. Cumulative distribution of the absolute error for the L-band EDFA device, as a function of the number of training datasets. (**a**) Overview of absolute errors; (**b**) Zoomed-in view of the *y*-axis region between 80% and 100%; (**c**) 90% and 99% error thresholds versus the number of datasets.

4.6.2. Accuracy in Function of the Train/Test Split Ratio

This second analysis focuses on assessing the impact of the train/test split ratio on model accuracy using datasets from different EDFAs sourced from multiple vendors. The goal is to evaluate how the size of the measurement dataset influences the model's performance. Rather than using datasets from separate devices, the models are built by merging the six available measurement datasets and varying the proportion of data allocated to training and testing. The results, shown as the cumulative distribution of absolute error, are presented in Figure 20. The results in the figure refer to the Cisco L-band EDFA device, which is used as a reference for this analysis. Figure 20a and, in detail, Figure 20b, demonstrate that model accuracy remains stable across different train/test split ratios. The curves largely overlap, indicating minimal performance variation, except for the 1% test size case (in blue), which shows a slight improvement in the zoomed-in region. Figure 20c, which illustrates the 90%- and 99%-thresholds as a function of the test size, further supports the observation that the absolute error remains consistent across different splits.



(**a**) (**b**) (**c**)

Figure 20. Cumulative distribution of the absolute error for the L-band EDFA device, as a function of the train/test split ratio. (**a**) Overview of absolute errors; (**b**) Zoomed-in view of the *y*-axis region between 80% and 100%; (**c**) 90% and 99% error thresholds versus test size.

Based on these findings, along with the results from the previous section, we conclude that the most effective strategy for achieving good model performance while minimizing the time required for EDFA power density spectrum measurements is to use data from multiple devices, even if the datasets are relatively small. For instance, increasing the step

size of total input power, gain, and tilt can help reduce the number of required power spectrum measurements. However, reducing the number of devices tested would not be advisable, as data diversity across different devices contributes significantly to model accuracy and generalization.

4.7. Computational Efficiency and Memory Usage of the Model

This section presents an analysis of the technical performance of the polynomial model. Two key metrics are considered: computational efficiency, which is measured by the execution time of the noise figure evaluation functions, and memory usage, which includes both the storage size of the model files and Python scripts in local memory, as well as the amount of RAM required for the library to perform computations. These metrics were assessed for the Python implementation of the model evaluation class, though they may vary depending on the specific solution used.

4.7.1. Computational Efficiency

A primary objective of the model is to ensure high time efficiency. As previously mentioned, this model is designed for integration into larger, more comprehensive EDFA models, rather than for standalone use. A fast model is less likely to become a performance bottleneck when incorporated into such systems. The time analysis is conducted using the `timeit` module, part of the standard Python 3 library.

The first test involves the instantiation of the class and the initialization of the model:

```
e = EDFA_NF()
e.init_model("EDFA17_LOW_38")
```

The first line creates an instance of the class and calls the constructor, which initializes the internal attributes to their default values. The `init_model` function performs the following operations:

1. checks if the `models` folder exists in the working directory; if not, an exception is raised;
2. attempts to retrieve the model file, using the name provided as a parameter, from the `models` folder, raising an exception if the file is not found;
3. uses the `Pandas` library to open the model's `.parquet` file and saves the model name and polynomial coefficients in two internal attributes.

The next function to be tested is the single-value noise figure calculation method:

```
nf = e.estimate_NF(p, g, t, f)
```

where `p`, `g`, `t`, and `f` are four floating-point variables corresponding to the input parameters. This function performs the following operations:

1. checks if the `init_model` function has been called and raises an exception if it has not;
2. calls the private `__warnings` method, which ensures that the input parameters are within the amplifier's operating limits, clamping the values if the corresponding flag is set; otherwise, it raises a warning;
3. runs the polynomial evaluation function.

It is important to note that this calculation involves some overhead due to the input parameter checks, which may affect execution speed. Lastly, the performance of the function responsible for computing the noise figure across multiple values simultaneously is evaluated:

```
nf = e.estimate_NF_array(p, g, t, f)
```

Here, `p`, `g`, `t`, and `f` are NumPy `ndarray` objects, all of the same length. The calculation is performed element-wise, ensuring that all four input arrays are processed simultaneously while maintaining their respective dimensions. This function extends the operations of the single-value noise figure computation to an entire array, enabling efficient batch processing. The execution times, measured in ms, µs, and ns, are reported in Table 3. To facilitate a direct comparison with the single-value evaluation case, the execution times for array processing are normalized by the array length.

Table 3. Execution time of the model evaluation Python class, categorized by function.

Class Method(s)	Exec. Time [ms]	Exec. Time [µs]	Exec. Time [ns]
Initialization	40.76	40,759.31	40,759,306.60
NF evaluation (single)	0.05	47.60	47,600.11
NF evaluation (array)	<0.01	0.61	613.81

It is immediately evident that the `init_model` method is significantly slower than the evaluation methods, primarily due to its file-reading operations. As a result, it is more efficient to use a single EDFA model for all computations and switch models only when necessary to estimate the noise figure for different devices. In scenarios requiring access to multiple models simultaneously, modifications to the library can be considered, such as storing several models within the class instance. This change would speed up operations but increase the memory footprint of the class instance. The `init_model` method takes approximately the same amount of time as 856 calls to the `estimate_NF` function, or a single call to the `estimate_NF_array` function with an array of roughly 6.6×10^4 elements.

It is also noteworthy that the `estimate_NF_array` method is approximately 80 times faster than the `estimate_NF` method when evaluating per-element computation time with a large array length (in this case, the length was about 2×10^4). On the other hand, the `estimate_NF` method is faster for single-element computations. Table 4 explores the threshold, determining the number of elements beyond which `estimate_NF_array` becomes faster than repeated calls to `estimate_NF`. Based on these results, we can conclude that the `estimate_NF` method is faster for array lengths up to approximately 33.

Table 4. Execution time of the array estimation function for the noise figure, as a function of array length.

Array Length	Exec. Time per Sample [µs]
1	1479.72
2	746.99
5	298.97
10	150.46
20	76.76
33	47.39

4.7.2. Memory Usage

This section evaluates the model's performance in terms of disk space and RAM consumption. The total size of the two Python scripts is approximately 36 kB. Each model file has a size of about 10 kB. These files contain 720 64-bit floating-point coefficients, each associated with a four-character string key. Given that each key and coefficient occupy 4 and 8 bytes, respectively, the storage size for the information in each model file can be

calculated as $(4 + 8) \times 720 = 8640$ bytes, or roughly 8.43 kB. The discrepancy between this calculated size and the actual file size is due to the overhead from the *.parquet* file structure. Thus, the model can be considered very lightweight for typical computing systems. To assess the RAM usage of the EDFA class, the `sys.getsizeof` method from the standard Python 3 library was employed. This method was applied to both the class instance and its internal attributes, yielding a result of 25.72 kB. A more detailed analysis using a memory profiler could provide further insights into memory usage during noise figure calculations. However, such an analysis would include other memory contributors, such as linked libraries, which makes it difficult to isolate the exact memory usage of the `EDFA_NF` class instance.

4.8. Utilization of the Model in a Deep-Learning EDFA Framework

The performance of the developed model is assessed by integrating it into a more complex EDFA model, as outlined in earlier sections. This integration is essential because the model was not intended for standalone use, but rather as a component within a broader system. The deep-learning EDFA model, previously trained for varying spectral load configurations in the input [8], accepts the signal power spectral PSD and EDFA configuration settings as inputs and generates the amplified output PSD. A critical factor for the model's success is its ability to accurately estimate the amplifier's noise figure. In previous usages of the model, this noise figure was determined using a heuristic, fixed-value method, which was independent of the input settings and configuration. This involved manually iterating through training and testing loops to find an optimal value. The tests were conducted using seven datasets from Juniper Networks devices, chosen specifically for their representation of worst-case scenarios. This selection ensures a rigorous test of the model's performance.

The results, comparing the fixed noise figure approach with the estimation model, are summarized in Tables 5–8. From Tables 5 and 6, it is clear that the maximum and mean absolute errors in noise figure predictions are significantly reduced when using the estimation model. This improvement is observed across most datasets, though a few datasets show only minor degradation in results. The R^2 score, which estimates the proportion of the noise figure's variation explainable by the input parameters, is provided in Table 7. The table shows minimal difference between the fixed-value method and the estimation model, suggesting that both approaches explain the variation in noise figure similarly. Finally, Table 8 presents the RMSE results, reinforcing the conclusion that the estimation model reduces errors in most cases. These results further support the model's ability to enhance prediction accuracy and overall performance. Given that these tests represent worst-case scenarios, the model is expected to perform even better with different devices.

Table 5. Performance of deep-learning EDFA model, maximum absolute error [dB].

Case	EDFA 1	EDFA 2	EDFA 3	EDFA 4	EDFA 5	EDFA 6	EDFA 7
Fixed NF	1.3914	1.1150	0.3941	0.4304	0.4176	0.5063	0.5245
Model NF	1.3099	1.1103	0.3730	0.4461	0.4067	0.4881	0.5351
	−5.86%	−0.42%	−5.35%	+3.65%	−2.61%	−3.59%	+2.02%

Table 6. Performance of deep-learning EDFA model, mean absolute error [dB].

Case	EDFA 1	EDFA 2	EDFA 3	EDFA 4	EDFA 5	EDFA 6	EDFA 7
Fixed NF	0.1269	0.0849	0.0583	0.0590	0.0566	0.0587	0.0771
Model NF	0.1093	0.0816	0.0543	0.0596	0.0555	0.0625	0.0772
	−13.87%	−3.89%	−6.86%	+1.02%	−1.94%	+6.47%	+0.13%

Table 7. Performance of deep-learning EDFA model, adjusted R^2 score [%].

Case	EDFA 1	EDFA 2	EDFA 3	EDFA 4	EDFA 5	EDFA 6	EDFA 7
Fixed NF	92.2129	95.3613	95.9068	97.7005	91.6659	94.8887	93.1631
Model NF	89.3877	95.2287	96.0442	97.5909	91.9185	94.2216	93.5089
	−2.82%	+0.09%	+0.14%	−0.11%	+0.25%	−0.67%	+0.35%

Table 8. Performance of deep-learning EDFA model, RMSE [dB].

Case	EDFA 1	EDFA 2	EDFA 3	EDFA 4	EDFA 5	EDFA 6	EDFA 7
Fixed NF	0.1908	0.1413	0.0810	0.0813	0.0782	0.0828	0.1093
Model NF	0.1715	0.1390	0.0773	0.0828	0.0758	0.0885	0.1081
	−10.11%	−1.63%	−4.57%	+1.84%	−3.07%	+6.88%	−1.10%

5. Conclusions

This paper presented the development and evaluation of a polynomial model for estimating the noise figure of Erbium-Doped Fiber Amplifiers (EDFAs). The model leverages machine-learning techniques to determine polynomial coefficients, achieving high accuracy and computational efficiency. The polynomial model demonstrated high accuracy in estimating the noise figure, with errors comparable to the inherent uncertainty in the measurement datasets. The 90%-threshold for absolute errors ranged from 0.05 to 0.28 dB, and the 99%-threshold ranged from 0.09 to 0.46 dB. These results indicate that the model provides reliable noise figure estimations, comparable to the accuracy of the measurement instruments. The analysis revealed that incorporating data from multiple devices is essential to account for manufacturing variations. Accuracy improved consistently up to three datasets, after which it stabilized with only minor gains as the number increased. This finding suggests that using measurements from multiple devices, even with smaller datasets, is a more effective strategy for achieving good model performance.

The model's computational efficiency was evaluated by measuring the execution time of the noise figure evaluation functions. The `estimate_NF_array` method was found to be approximately 80 times faster than the `estimate_NF` method when evaluating per-element computation time with a large array length. This efficiency makes the model suitable for integration into larger, more comprehensive EDFA models without becoming a performance bottleneck. The memory usage of the model was also found to be minimal, with the total size of the Python scripts and model files being approximately 46 kB and the RAM usage of the EDFA class instance being 25.72 kB. The integration of the polynomial model into a deep-learning framework for EDFA performance prediction demonstrated its practical applicability. The results showed that the estimation model significantly reduced the maximum and mean absolute errors in noise figure predictions compared to the fixed-value method. This improvement highlights the model's ability to enhance prediction accuracy and overall performance in real-world applications.

While the polynomial model has shown promising results, there are some limitations to consider. The model's accuracy is dependent on the quality and size of the training datasets, and further improvements could be achieved by exploring additional parameters or improved polynomial fitting techniques. Future research could also investigate the model's performance with different types of EDFAs and under varying operating conditions to enhance its generalizability and robustness.

In conclusion, the polynomial model developed in this work provides a reliable, efficient, and adaptable solution for estimating the noise figure of EDFAs. Its integration into a deep-learning framework for EDFA performance prediction further validates its practical applicability, including its validation in challenging conditions for prediction, such as in cases with variable spectra. Future research should focus on addressing the identified limitations and exploring additional enhancements to improve the model's accuracy and applicability across a broader range of devices and operating conditions.

Author Contributions: Conceptualization, methodology, software, validation, formal analysis, and investigation, R.D., A.C., S.S. and V.C.; resources and data curation, S.S.; writing—original draft preparation, R.D., A.C. and V.C.; writing—review and editing, R.D. and A.C.; visualization, R.D.; supervision and project administration: R.D. and V.C.; funding acquisition, V.C. All authors have read and agreed to the published version of the manuscript.

Funding: This research has been supported by the EU Horizon Europe research and innovation program, ALLEGRO Project, GA No. 101092766. .

Data Availability Statement: The raw data supporting the conclusions of this article will be made available by the authors on request..

Conflicts of Interest: The authors declare no conflicts of interest..

References

1. Winzer, P.J.; Neilson, D.T.; Chraplyvy, A.R. Fiber-optic transmission and networking: The previous 20 and the next 20 years. *Opt. Express* **2018**, *26*, 24190–24239. [CrossRef]
2. Zhu, S.; Gutterman, C.; Montiel, A.D.; Yu, J.; Ruffini, M.; Zussman, G.; Kilper, D. Hybrid Machine Learning EDFA Model. In Proceedings of the Optical Fiber Communication Conference (OFC) 2020, San Diego, CA, USA, 8–12 March 2020; p. T4B.4. [CrossRef]
3. Roberts, K.; Zhuge, Q.; Monga, I.; Gareau, S.; Laperle, C. Beyond 100 Gb/s: Capacity, Flexibility, and Network Optimization. *J. Opt. Commun. Netw.* **2017**, *9*, C12–C24. [CrossRef]
4. Gerstel, O.; Jinno, M.; Lord, A.; Yoo, S.B. Elastic optical networking: A new dawn for the optical layer? *IEEE Commun. Mag.* **2012**, *50*, s12–s20. [CrossRef]
5. Roberts, I.; Kahn, J.M.; Boertjes, D. Convex Channel Power Optimization in Nonlinear WDM Systems Using Gaussian Noise Model. *J. Light. Technol.* **2016**, *34*, 3212–3222. [CrossRef]
6. Curri, V.; Cantono, M.; Gaudino, R. Elastic All-Optical Networks: A New Paradigm Enabled by the Physical Layer. How to Optimize Network Performances? *J. Light. Technol.* **2017**, *35*, 1211–1221. [CrossRef]
7. Yichen, L.; Xiaomin, L.; Lei, L.; Yihao, Z.; Meng, C.; Lilin, Y.; Weisheng, H.; Qunbi, Z. Modeling EDFA Gain: Approaches and Challenges. *Photonics* **2021**, *8*, 417. [CrossRef]
8. D'Ingillo, R.; D'Amico, A.; Ambrosone, R.; Virgillito, E.; Gatto, V.; Straullu, S.; Aquilino, F.; Curri, V. Deep Learning Gain and Tilt Adaptive Digital Twin Modeling of Optical Line Systems for Accurate OSNR Predictions. In Proceedings of the 28th International Conference on Optical Network Design and Modelling, Madrid, Spain, 6–9 May 2024.
9. Burgemeier, J.; Cords, A.; Marz, R.; Schaffer, C.; Stummer, B. A black box model of EDFA's operating in WDM systems. *J. Light. Technol.* **1998**, *16*, 1271–1275. [CrossRef]
10. You, Y.; Jiang, Z.; Janz, C. Machine Learning-Based EDFA Gain Model. In Proceedings of the 2018 European Conference on Optical Communication (ECOC), Rome, Italy, 23–27 September 2018; pp. 1–3. [CrossRef]
11. Borraccini, G.; Gatto, V.; D'Amico, A.; Straullu, S.; Aquilino, F.; Nespola, A.; Piciaccia, S.; Tanzi, A.; Galimbert, G.; Curri, V. Gain Profile Characterization and Modeling for Dual-Stage EDFA Abstraction and Control. *IEEE Photonics Technol. Lett.* **2023**, *36*, 107–110. [CrossRef]

12. Liu, Y.; Liu, X.; Zhang, Y.; Cai, M.; Fu, M.; Zhong, X.; Yi, L.; Hu, W.; Zhuge, Q. Building a digital twin of EDFA: A grey-box modeling approach. *arXiv* **2023**, arXiv:2307.06862.
13. D'Amico, A.; Straullu, S.; Nespola, A.; Khan, I.; London, E.; Virgillito, E.; Piciaccia, S.; Tanzi, A.; Galimberti, G.; Curri, V. Using machine learning in an open optical line system controller. *J. Opt. Commun. Netw.* **2020**, *12*, C1–C11. [CrossRef]
14. D'Amico, A.; Straullu, S.; Borraccini, G.; London, E.; Bottacchi, S.; Piciaccia, S.; Tanzi, A.; Nespola, A.; Galimberti, G.; Swail, S.; et al. Enhancing lightpath QoT computation with machine learning in partially disaggregated optical networks. *IEEE Open J. Commun. Soc.* **2021**, *2*, 564–574. [CrossRef]
15. Baney, D.M.; Gallion, P.; Tucker, R.S. Theory and Measurement Techniques for the Noise Figure of Optical Amplifiers. *Opt. Fiber Technol.* **2000**, *6*, 122–154. [CrossRef]
16. Premaratne, M.; Agrawal, G.P. *Light Propagation in Gain Media: Optical Amplifiers*; Cambridge University Press: Cambridge, UK, 2011.
17. Setiawan Putra, A.W.; Yamada, M.; Tsuda, H.; Ambran, S. Theoretical Analysis of Noise in Erbium Doped Fiber Amplifier. *IEEE J. Quantum Electron.* **2017**, *53*, 5100108. [CrossRef]
18. Caves, C.M. Quantum limits on noise in linear amplifiers. *Phys. Rev. D* **1982**, *26*, 1817–1839. [CrossRef]
19. Vasilyev, M. Distributed phase-sensitive amplification. *Opt. Express* **2005**, *13*, 7563–7571. [CrossRef]
20. Sun, Y.; Zyskind, J.; Srivastava, A. Average inversion level, modeling, and physics of erbium-doped fiber amplifiers. *IEEE J. Sel. Top. Quantum Electron.* **1997**, *3*, 991–1007. [CrossRef]
21. Ishii, K.; Kurumida, J.; Namiki, S. Experimental Investigation of Gain Offset Behavior of Feedforward-Controlled WDM AGC EDFA Under Various Dynamic Wavelength Allocations. *IEEE Photonics J.* **2016**, *8*, 7901713. [CrossRef]
22. Pointurier, Y. Machine learning techniques for quality of transmission estimation in optical networks. *J. Opt. Commun. Netw.* **2021**, *13*, B60–B71. [CrossRef]
23. Dorn, W.S. Generalizations of Horner's rule for polynomial evaluation. *IBM J. Res. Dev.* **1962**, *6*, 239–245. [CrossRef]
24. Menke, W. Review of the generalized least squares method. *Surv. Geophys.* **2015**, *36*, 1–25. [CrossRef]
25. Van Rossum, G. Python Programming Language. USENIX Annual Technical Conference. 2007. Available online: https://www.python.org (accessed on 1 January 2023)
26. Agilent Technologies. *EDFA Testing with the Interpolation Technique*; Agilent Technologies: Santa Clara, CA, USA, 2000.
27. Virtanen, P.; Gommers, R.; Oliphant, T.E.; Haberland, M.; Reddy, T.; Cournapeau, D.; Burovski, E.; Peterson, P.; Weckesser, W.; Bright, J.; et al. SciPy 1.0: Fundamental algorithms for scientific computing in Python. *Nat. Methods* **2020**, *17*, 261–272. [CrossRef]
28. Ghazisaeidi, A. A Theory of Nonlinear Interactions Between Signal and Amplified Spontaneous Emission Noise in Coherent Wavelength Division Multiplexed Systems. *J. Light. Technol.* **2017**, *35*, 5150–5175. [CrossRef]
29. McKinney, W. *Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython*; O'Reilly Media, Inc.: Sebastopol, CA, USA, 2012.
30. Cisco Systems. *A Switchable-Gain Optical Amplifier Functioning in the Flexible Spectrum of C-Band for the Cisco NCS 2000 Platform Data Sheet*; Cisco Systems: Tokyo, Japan, 2018.
31. Sadik, S.A. Noise Figure Estimation of EDFA Based on Gradient Boosting Regression Approach for THz Applications. In Proceedings of the 2022 Workshop on Microwave Theory and Techniques in Wireless Communications (MTTW), Riga, Latvia, 5–7 October 2022; pp. 86–89. [CrossRef]
32. Cisco Systems. *Enhanced C-Band 96-Channel EDFA Amplifiers for the Cisco ONS 15454 MSTP Data Sheet*; Cisco Systems: Tokyo, Japan, 2016.
33. Cisco Systems. *L-Band Optical Amplifier Portfolio for the Cisco ONS 15454*; Cisco Systems: Tokyo, Japan, 2016.
34. Juniper Networks. *Optical Inline Amplifier Description*; Juniper Networks: Sunnyvale, CA, USA, 2023.
35. Kahloot, K.M.; Ekler, P. Algorithmic Splitting: A Method for Dataset Preparation. *IEEE Access* **2021**, *9*, 125229–125237. [CrossRef]
36. Bastos-Filho, C.J.; Barboza, E.d.A.; Martins-Filho, J.F.; de Moura, U.C.; de Oliveira, J.R. Mapping EDFA noise figure and gain flatness over the power mask using neural networks. *J. Microwaves, Optoelectron. Electromagn. Appl. (JMOe)* **2013**, *12*, 128–139.
37. Bastos-Filho, C.J.; Barboza, E.d.A.; Martins-Filho, J.F. Estimating the spectral gain and the noise figure of EDFA using artificial neural networks. In Proceedings of the 2017 19th International Conference on Transparent Optical Networks (ICTON), Girona, Spain, 2–6 July 2017; pp. 1–4. [CrossRef]