Adaptive and Iterative QoT Estimator Retraining for Launch Power Optimization

Ankush Mahajan(1)*, Kostas Christodoulopoulos(3), Ricardo Martínez(1), Raul Muñoz(1), Salvatore Spadaro(2)
(1) Centre Tecnològic de Telecomunicacions de Catalunya (CTTC/CERCA), (2) Polytechnic University of Catalonia (UPC), Barcelona, Spain,
(3)Nokia Bell Labs, Stuttgart, Germany
**ankush.mahajan@cttc.cat*

Abstract: We dynamically optimize the transponders launch powers with iterative closed control loops. We close the loop after an adaptable number of algorithm's intermediate calculations, to monitor and retrain the QoT estimator with the real world. © 2021 The Author(s)

1. Introduction

A physical layer model (PLM) or a Quality of Transmission (QoT) estimator is required for almost every optimization task in an optical network. The accuracy of the PLM affects how close to the real optimum we can reach. Typically, there are safety mechanisms in place: margins are used to cover the PLM uncertainties and the variability of the physical layer conditions [1], but also the network changes are typically performed during maintenance windows and checked before returning to operation. Monitoring and machine learning (ML) techniques have recently been proposed to account for the actual network conditions and improve the accuracy of the PLM [1], [2]. Nevertheless, the PLM accuracy becomes more critical when the PLM is used for dynamic optimization, since to justify dynamic optimization we should target higher network efficiency. The advantage is that the network operates and so we can use its feedback to achieve an efficient network optimization.

Optimization problems involving the optical physical layer are multidimensional and combinatorial, that is, a change in one variable affects several others [3]-[5]. For example, an establishment of a new connection or a change in a single transponder launch power results in the QoT variations of all the interfered connections [5]. The effect of a few reconfigurations is relatively easy to predict with the PLM. However, if the optimization algorithm at intermediate calculations has assumed several reconfigurations it can project the network into states where the PLM does not attain good accuracy. Since these are intermediate calculations, the network will be configured at the end, the deterioration of the accuracy of the PLM is not appreciated, which could mislead the calculations and result in poor optimization.

In this paper we study the dynamic launch power optimization problem, where we have a set of established connections and the aim is to optimize their transponders (TPs) launch powers while the network operates. In this context, several works have appeared aiming the minimization of intra and inter-channel non-linear impairments (NLI) by optimizing TPs launch powers [3], [5]-[6]. In particular, authors in [5], formulated the launch power optimization to maximize the sum or the worst case margin, using the Gaussian noise (GN) model [4] as PLM, as a convex optimization problem. However, all above works used a PLM with fixed parameters, but due to the interdependencies in the physical layer, the PLM needs either margins or parameter realignment when the algorithm uses it for intermediate calculations that project the network into substantially different states. Similar PLM accuracy issues arise in other multivariable dynamic optimization problems such as dynamic resource allocation, automatic network reconfiguration, defragmentation, virtual network reconfiguration etc. [7], [8]. To avoid PLM inaccuracies, [5] extended [6] and proposed to probe (change the launch power) and monitor to calculate the partial derivatives needed by the optimization algorithm's intermediate calculations. However, [6] assumes perfect NLI monitoring and interacts extensively with the network for probing and monitoring.

In light of the above herein, we propose to use *iterative* and *adaptive* closed control loops to dynamically optimize the launch powers. After an adaptable number of internal optimization algorithm calculations, we close the loop, configure the network and realign/retrain the PLM (via monitoring and ML). Leveraging these adaptive retraining cycles, the PLM represents the real system with enough accuracy to support the dynamic optimization at hand.

2. Methodology and Proposed Solution

In an SDN controlled network, the PLM would interact with algorithms to carry out the optimization tasks [7]. Many of these algorithms would solve multivariable dynamic optimization problems such as the optimization of TPs launch powers in the operating network. This problem is known to be convex and polynomial. The convex optimization algorithms, such as (sub)gradient methods, interior point, etc., are iterative; at each iteration they calculate partial derivatives for the objective and constraint functions [9]. If a PLM is used without closed form equations for partial derivatives, a way to calculate them is to use a finite differences method [10].

We assume an operating network with N established connections. The Gaussian Noise (GN) model is a well-accepted PLM [4] and was used in this work [9]. The GN PLM takes as input several parameters and calculates the generalized SNR values of the connections. Let r denote the set of GN model fitted input parameters: i) fiber attenuation coefficient,

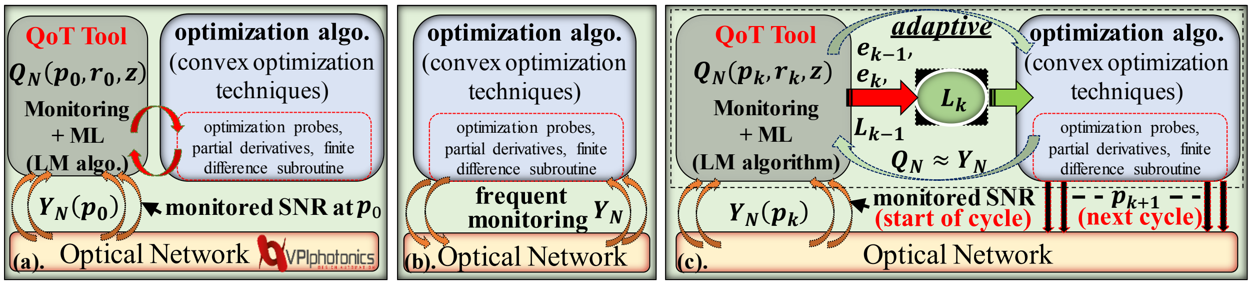
QoT Tool
$Q_N(p_0, r_0, z)$
Monitoring + ML (LM algo.)
optimization algo. (convex optimization techniques)
optimization probes, partial derivatives, finite difference subroutine
$Y_N(p_0)$
monitored SNR at p_0
(a). Optical Network
VPIphotonics
optimization algo. (convex optimization techniques)
optimization probes, partial derivatives, finite difference subroutine
frequent monitoring Y_N
(b). Optical Network
QoT Tool
$Q_N(p_k, r_k, z)$
Monitoring + ML (LM algorithm)
adaptive
e_{k-1}, e_k, L_{k-1}
L_k
$Q_N \approx Y_N$
optimization algo. (convex optimization techniques)
optimization probes, partial derivatives, finite difference subroutine
$Y_N(p_k)$
monitored SNR (start of cycle)
p_{k+1} (next cycle)
(c). Optical Network

Fig. 1. Optimization with (a) One-time trained PLM, (b) Monitoring probes, (c) Proposed PLM retraining adaptive and iterative control loop

ii) fiber non-linear coefficient, iii) fiber dispersion coefficients, iv) a wavelength dependent penalty term, implemented as a 4th order polynomial, to cover transponder loss mismatches, amplifiers ripple etc., and v) a bias. Also let $\boldsymbol{p} = [\boldsymbol{p_1}, \boldsymbol{p_1}, \ldots, \boldsymbol{p_N}]$ be the launch power vector of the *N* connections, which are the variables to be optimized, and let $\boldsymbol{z}$ represents the unchanged input parameters for our optimization, such as routes, used wavelengths, span lengths etc. We denote by $\boldsymbol{Q_N}(\boldsymbol{p}, \boldsymbol{r}, \boldsymbol{z})$ the calculated SNR vector for all the *N* connections. The PLM SNR calculation function is non-linear in its parameters $\boldsymbol{r}$ (and also $\boldsymbol{p}$). Finally, let $\boldsymbol{Y_N}(\boldsymbol{p})$ denote the monitored SNR vector. We assume that this is done at the coherent receivers. To align the PLM we identify the parameters $\boldsymbol{r}$ that minimize the squared error. To do this, we adopted ML based on Levenberg- Marquardt (LM) algorithm to solve the nonlinear least squares fitting problems [11]. Assuming launch powers $\boldsymbol{p_0}$, the LM algorithm would find $\boldsymbol{r_0} = argmin_r(\boldsymbol{Q_N}(\boldsymbol{p_0}, \boldsymbol{r}, \boldsymbol{z}) - \boldsymbol{Y_N}(\boldsymbol{p_0}))^2$.

When such PLM alignment is performed once before the optimization task, the PLM reflects with good accuracy the starting state of the network. Assuming that we start with launch powers $\boldsymbol{p_0}$, we align the PLM, denoted by $\boldsymbol{Q_N}(\boldsymbol{p_0}, \boldsymbol{r_0}, \boldsymbol{z})$, and use that for the intermediate algorithm's calculations, involving calculations for SNR values and partial derivates using the finite difference subroutine. Fig. 1(a) depicts the schematic for the traditional launch power *optimization with one-time trained PLM* approach. An alternative is to probe the real network, interface the algorithm and, in particular, the derivative identification subroutine with the network, bypassing the PLM. In this case, the derivative identification process configures the launch powers of the TPs via the SDN control plane, and monitors the outcomes (SNR values, $\boldsymbol{Y_N}(\boldsymbol{p})$) to calculate the derivatives. This is repeated several times at each algorithm's iteration. We refer to this as *optimization with monitoring probes* and show it in Fig. 1(b). Optimizing with monitoring probes is very accurate but slow as the algorithm continuously probes and monitors the network for derivative calculations. On the other hand, in the traditional one-time trained PLM approach, the derivative calculations with the PLM are fast. However, since the PLM was trained before the optimization, if the algorithm at its intermediate calculations projects the network into states (i.e. power levels) where the PLM has low accuracy, the optimization can be misled.

To overcome this, we propose to use an adaptive and iterative closed control loop process to solve the dynamic multivariable launch power optimization problem. At certain (adaptable) number of intermediate iterations of the algorithm we close the loop, configure the network and monitor to retrain the PLM (with ML) to follow the projected network states, as shown in Fig. 1(c). In this way, the PLM becomes a Digital Twin (DT): a parametric model that works in evolving network conditions and has a method to adapt to follow these conditions. Let us be on *k*-1 PLM retraining cycle, have configured the power vector $\boldsymbol{p_{k-1}}$ and monitored SNR values $\boldsymbol{Y_N}(\boldsymbol{p_{k-1}})$. These are used to train the PLM $\boldsymbol{Q_N}(\boldsymbol{p_{k-1}}, \boldsymbol{r_{k-1}}, \boldsymbol{z})$. Then in this *k*-1 PLM retraining cycle, the algorithm performs intermediate iterations. At each iteration it calculates new powers using that PLM without configuring them to the network. After L_{k-1}iterations (we discuss how to find this for the next cycle), it finds the powers $\boldsymbol{p_k}$ and configures them to the network. We then monitor and obtain $\boldsymbol{Y_N}(\boldsymbol{p_k})$. We calculate the PLM alignment error $\boldsymbol{e_k}$=$\|\boldsymbol{Y_N}(\boldsymbol{p_k}) - \boldsymbol{Q_N}(\boldsymbol{p_k}, \boldsymbol{r_{k-1}}, \boldsymbol{z})\|^2$, using the PLM trained at the start of the previous *k*-1 cycle. We then start cycle *k*, use $\boldsymbol{Y_N}(\boldsymbol{p_k})$ to train the PLM, which is in turn used to perform L_k optimization intermediate iterations. In our previous work [9] the number of iterations was fixed for all cycles, so $L_{k-1} = L_k = L_{fixed}$, for all *k*. In this paper we devised an adaptive scheme. We adapt the number of iterations L_k at cycle *k* using a function *f* of the PLM alignment errors of previous *k-1* and current *k* cycle: L_k=*f* (e_k, e_{k-1}, L_{k-1}). The idea is to track the evolution of the alignment of the PLM and reduce/increase the realignment period accordingly. In particular, in the following results we implemented a simple adaptive function *f* where we increased/decreased *L* by 5 if we observed a decrease/increase of the alignment error.

3. Results & Discussion

To quantify the benefits of the devised PLM/QoT estimator retraining based launch power optimization process, we performed simulations, using VPI [12] as the real world/ground truth and the GN model as the PLM. Note that VPI is more detailed and complex and closer to a real system than GN. This choice was made to capture the mismatch between the real network and the PLM used in the optimization process, and is considerably more realistic than most prior studies [2], [5], where the same PLM was used for both the real network and proposed solution. We simulated a 4 nodes topology (assumed to be a part of a bigger network) with 15 reused wavelengths and 21 connections in VPI, see

Fig. 2(a). We assumed different attenuation (0.19-0.21dB/km), dispersion (16.5-17ps/(nm·km)) and fiber non-linearity coefficients (1.3-1.38$W^{-1}\cdot km^{-1}$) for the 3 links, introducing uncertainties which need to be modeled in ML using per link parameters ***r***. We used EDFAs with gain ripple 0.4dB peak-to-peak, and 32GBaud 16-QAM mod. format per connection. We implemented in MATLAB the GN model and the launch power optimization algorithm to maximize: (obj#1) the sum of SNR margins, or (obj#2) the lowest margin, where margin is the difference between the real SNR (measured in the network/VPI) and the SNR threshold [9]. Since the optimization problem is convex [5], we implemented in MATLAB an interior-point algorithm to solve it, which was run with optimality tolerance 10^{-6}.

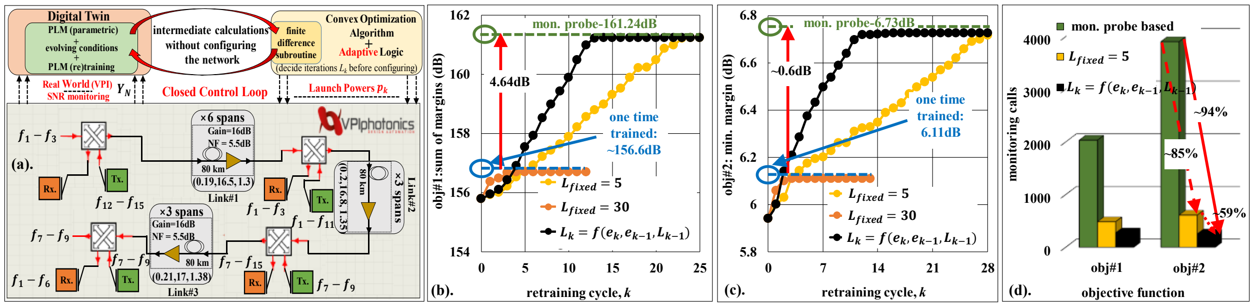

Fig. 2. (a) VPI setup, (b) Obj#1, (c) Obj#2 values as a function of proposed adaptive, iterative PLM retraining based launch optimization approach for EDFAs with min-max peak to peak gain ripple values of ±0.1 to ±0.3dBs, and (d) Corresponding reduction in the number of monitoring calls

When optimizing with the one-time trained PLM we monitor the initial SNR values in VPI, pass them to MATLAB, use them to train the PLM, which is then used for all algorithm intermediate iterations. The final powers are passed to VPI where we find the resulted real margin. When optimizing with monitoring probes, the derivatives calculation subroutine probes, gives new launch powers, to VPI, which calculates the SNR values and passes them back. This process is performed several times until the algorithm calculates the partial derivatives and the next launch powers, and is repeated for each algorithm iteration. In our proposed approach, VPI takes the launch powers from the algorithm and calculates the SNR values. These are passed back to MATLAB, used to find the alignment error and the next period L_k, and train the PLM. This PLM is then used for the next L_k algorithm intermediate iterations.

Fig. 2(b) shows the optimization objective values (evaluated in VPI) as a function of the retraining cycle (k) of our proposed scheme. Compared to one-time trained PLM approach, we noticed ~4.6dB improvement in obj#1 (Fig. 2(b)) and ~0.6dB in obj#2 (Fig. 2(c)). Fig. 2(b) and 2(c) shows that with fixed retraining period L_{fixed}= 5, the solution reaches the optimum, however it took more iterations. With the adaptive loop (L_k=$f(e_k, e_{k-1}, L_{k-1})$) we obtained ~54-60% reduction in retraining cycles without compromising the performance. In general, the monitoring probes approach achieves the optimum since it continually tracks the network state. However, the extensive interactions with the network are prohibitive. In Fig. 2(d) we see that optimizing with monitoring probes required from ~1500 to ~4000 monitoring calls, for obj#1 and obj#2 with rippled EDFA gain profiles. This makes the overall approach very slow. Our approach substantially reduces the calls. With respect to [9] where we used a fixed period (L_{fixed}) and links without uncertainties, the devised adaptive period method reduced the calls by ~59%. Ultimately, we reduce by ~90 to 94% the monitoring calls with respect to optimizing with monitoring probes, which will reflect to a similar reduction in the total optimization time.

4. Conclusion

We proposed to use an adaptive and iterative closed control loop process to solve the multivariable dynamic TPs launch power optimization problem. We showed an improvement of ~4.6dB and ~0.6dB in sum of margins and lowest margin, respectively, over optimization with a one-time trained PLM. We achieved near to optimum solutions as found by optimizing and continuously probing and monitoring the network, but with ~94% fewer monitoring probe calls.

Acknowledgements: This work is a part of H2020-MSCA, ONFIRE project supported by EU, grant agreement No. 765275.

5. References

[1] Y. Pointurier, “Design of low-margin optical networks,” JOCN 9(1), A9-A17 (2017).
[2] I. Sartzetakis, et al., “Accurate Quality of Transmission Estimation with Machine Learning,” JOCN 11(3), 140-150 (2019).
[3] D. J. Ives et al., “Transmitter optimized optical networks,” OFC/NFOEC (2013).
[4] P. Poggiolini et al., “The GN-Model of Fiber Non-Linear Propagation and its Applications,” JLT 32(4), 694-721 (2014).
[5] I. Roberts et al., “Convex channel power optimization in nonlinear WDM systems using GN model,” JLT 34(13), 3212–3222 (2016).
[6] I. Roberts et al., “Measurement-Based Optimization of Channel Powers with Non-Gaussian NLI Noise,” JLT 36(13), 2746-2756 (2018).
[7] D. Rafique et al., “Machine learning for network automation: overview, architecture, and applications,” JOCN 10(10), D126-D143 (2018).
[8] K. D. R. Assis et al., “Network virtualization over elastic optical networks with different protection schemes,” JOCN 8(4), 272-281 (2016).
[9] A. Mahajan et al., “QoT estimator retraining for dynamic optimization in optical networks,” JOCN (2021)
[10] M. S. Gockenbach, "Computing Derivatives by Finite Differences," Notes MA5630, MTU, Houghton, Michigan (2013)
[11] H. P. Gavin, "The LM Algo. for Non-Linear Least Squares Curve-Fitting Problems," Notes CE281, Duke university, North Carolina (2020)
[12] https://www.vpiphotonics.com/index.php