GPU Accelerated MIMO Adaptive Equalization

M. C. van Leeuwen

(Bachelor project - Final report)

***Abstract*—High capacity optical fiber transmission systems are heavily reliant on digital signal processing. With upcoming technologies to match the exponential growth in demand, computational complexity increases. An important component in spatial-division multiplexing (SDM) transmission systems is the multiple input, multiple output adaptive equalizer (MIMO) to unravel channel mixing. In this work we investigate optimization of the MIMO by means of parallel processing. To evaluate the performance of the implementation central processing unit (CPU) and graphical processing unit (GPU) implementations are developed and benchmarked. The results report an up to 150 times lower processing time in favor of the GPU.**

I. Introduction

As the Internet continues to grow, the demand for more bandwidth is increasing exponentially. This demand has led to more fiber optical cable deployment in order to provide high speed Internet connectivity around the world. However, the growth in demand coupled with the non-linear Shannon limit, ensures that the bandwidth of these cables is not sufficient for the world's future demand for high-capacity communication. [1] [2] [3] [4]

Multi-core fibers are seen as a good candidate for increasing the bandwidth of optical communication. These cables essentially consist of multiple single mode cores manufactured within the same fiber cladding. In essence, many fiber cores can be added, however the overall outer diameter of the fiber may prevent it from being cabled and manufactured due to mechanical sensitivity and losses that may result from bending large diameter fibers. In addition, accommodating multiple cores means that the spacing between cores reduces and this can lead to crosstalk and therefore requiring more signal processing. [5]

Another technique for increasing the bandwidth of optical fiber cables is by transmitting over multiple modes, a technique known as mode-Division Multiplexing (MDM). Multiple modes require bigger fiber cores than conventional single mode fibers to transfer the multiple modes of light. Typically the diameter of these few-mode or multi-mode fiber is between 13 and 50 microns, remaining within the standard cladding diameter. Different modes within the multi-mode fiber propagate at different speeds, which causes interference at longer distances. Therefore, multi mode optical fibers are currently only employed over small transmission distances such as within datacenters or within buildings. Furthermore these modes are used to carry the same data so it is not recognized as SDM [6]

Research teams around the world are currently investigating the optimal number of cores and the number of modes based on performance, manufacturability and cost. [5] [7] [8] [9] [6] However, when the number of cores and modes in a fiber increases, one aspect that increases is the required signal processing. Various research teams currently developing space division multiplexed transmission systems, typically rely on offline signal processing. Hence, speeding up the signal processing required will mean that experiments can be done much faster.

Most of the computational burden of equalizing these extra modes and cores comes from the MIMO. A single Adaptive equalizer poses a significant but manageable computational burden. However, this burden is rapidly grows as more modes and cores are added in the system. Nonetheless much of this extra load may be computed in parallel meaning that implementing an efficient algorithm on the GPU is expected to lead to dramatic speedups for the laboratory based transmission measurements. [10] [11]

In this work, the benefits of implementing these algorithms exploiting parallel computing on GPUs will be discussed. Furthermore a performance difference with respect to the conventional approach employing sequential implementations processed with CPUs is investigated. This work employs Python in order to support a more open-source, fast and efficient implementation of the algorithms for improved laboratory offline processed measurements employing GPUs.

Parallelism also plays a crucial role in the design of a real time implementation with either application specific integrated circuits (ASICs) or field-programmable gate arrays (FPGAs) since these chips can be designed to benefit from parallelism similar to the GPU. An experiment with a parallel implementation of a MIMO on a FPGAs has already proven the concept. [12].

Consequently, this work is discussed as follows; in Section II, the typical transmission impairments are outlined. Section III discusses the various implementations of adaptive equalizers and how these equalizers are structured to form a Multiple Input/ Multiple output adaptive equalizer (MIMO). A strategy for efficiently implementing the MIMO in a massively parallel configuration is presented in Section IV with the results discussed before the conclusion in Section V.

II. Transmission Impairments

Optical fiber communication is disturbed by a mixture of both linear and Non-linear transmission impairments which the signal processing at the receiver side needs to compensate for. [13]

A. Chromatic Dispersion

As the signal propagates over an optical fiber, the spectral constituents spread with the shorter wavelength constituents of the signal propagating faster or slower than the longer wavelength spectral constituents of the signal, hence arriving at different times. This signal spread is mainly due to the material. This effect is typically known as chromatic dispersion, and if this effect is not compensated can cause inter-symbol interference (ISI) between symbols transmitted to close to each other. [14]

B. Mode Dispersion

To fully utilize the available bandwidth of the optical channel, the symbols are transmitted over both the perpendicular polarizations of light are modulated. However due to random perturbations (temperature, stress, vibrations etc) and the characteristics of the fiber core (step or graded index fiber) can cause the different polarizations to travel at different speeds through the fiber which leads to similar problems as with chromatic dispersion. This effect is the limiting factor for the bandwidth of single mode fiber optic communications since its unpredictability makes it hard to compensate. [15]

Another mode dispersion effect occurs when mode-division multiplexing is used. The different modes will be build out of different wavelengths, therefore these modes will also propagate through the optic fiber at different speeds. [15]

C. Crosstalk

Increasing the number of modes and the number of fiber cores used to transmit optical signals increases the efficiency and therefore the bandwidth of optical fiber communication. Due to perturbations and imperfections within the fiber and mode multiplexers the signals can start blending together. Therefore the several receivers will receive a mix of signals. [15]

III. Adaptive equalization

In Fig 1 a typical chain of digital signal processing is displayed. The focus of this paper will be on the MIMO equalizer, the other filters are discussed with more detail in [13]. Shortly, the analog to digital converters (ADCs) produce the in-phase (I) and quadrature (Q) components of the two polarizations (x and y) of the signal from the receivers. First a skew between these components is removed in the IQ-imbalances Compensation, then the chromatic dispersion is statically estimated and compensated accordingly. To compensate between clock errors between the transmitter and receivers timing recovery is applied. Furthermore phase and frequency offsets between the sender and receiver need to be compensated through their equalizers. Finally, the resulting symbols are mapped to their according bits in the demapping stage.

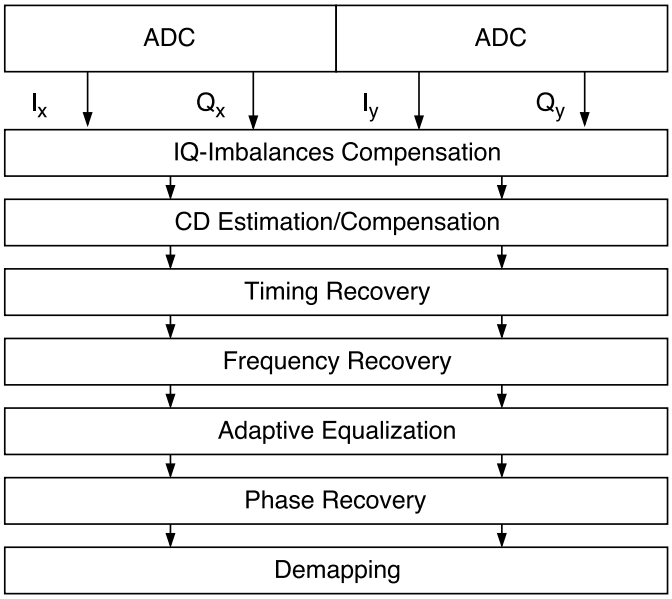


Fig. 1: Digital signal processing chain for a single Mode

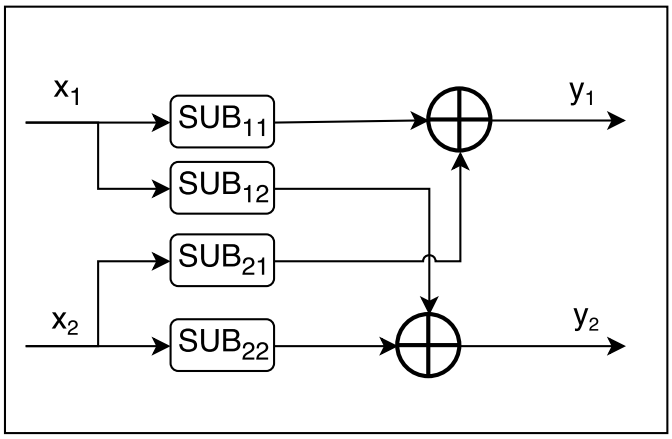


Fig. 2: Butterfly structure of the MIMO, where SUB denotes a single adaptive equalizer.

The main objective of the adaptive equalization is to effectively compensate the rapidly changing linear impairments of the transmission as outlined in section II. Such a filter can be designed using certain properties of the modulation format. For every sample or block of samples a certain deviation from this property is calculated and is used to change the filter for the next sample or block of samples.

A. MIMO

To compensate for the crosstalk between different modes and polarizations the equalizer has to be build out of several separate adaptive sub-equalizers between every input SDM channel and output SDM channel. This structure is known as the butterfly structure as depicted in Fig. 2.

The general idea behind MIMO is that the transmission impairments for all incoming samples can be estimated accurately using an $M \times M$ transfer matrix, where M represents

the total number of channels used to transmit through the optic fiber. So including the polarizations, modes and cores that are used to transmit signals.

Let's consider a transmitted signal with two modulated channels in the frequency domain as

$$T = \begin{pmatrix} E_x(\omega) \\ E_y(\omega) \end{pmatrix}.$$

where E_i is columnvector with the length of the input sequence N.

The impairments affecting the transmission can then be seen as $2 \times 2 \times N$ transfer matrix H

$$H(\omega) = \begin{pmatrix} H_{xx}(\omega) & H_{xy}(\omega) \\ H_{yx}(\omega) & H_{yy}(\omega) \end{pmatrix}.$$

where H_{ii} is a columnvector with the length of the input sequence N.

The received Signal R is then

$$R = HT \tag{1}$$

Therefore to obtain T the inverse of the matrix H should be estimated and multiplied with R

$$T = H^{-1}R \tag{2}$$

In a system where MDM is implemented the MIMO will need to compensate for more channels, and thus the matrix dimensions may grow. [16] However the crosstalk between channels with a bigger physical separation, such as is often the case in multi core fibers ,is often insignificant enough to leave uncompensated [17] .Therefore multiple independent smaller filters can often be used for multiple channels, which reduces the complexity of the MIMO. [5]

B. Filter updates

A common algorithm for updating the filter is the constant modulus algorithm. For a phase-shift keyed modulation format the constellation points are all position on a circle, therefore these points will share the same modulus. To calculate the error signal the difference in modulus can be calculated and used to update the filter. Such an algorithm can be extended to the multi-modulus to support QAM formats, where the constellations form multiple circles.

However for QAM formats the Least mean squares(LMS) algorithm is used more commonly. In the LMS algorithm compensation works similar to the CMA algorithm, except the error signal is created using the distance between the incoming signal to the constellation point it is eventually mapped to. To create such an error signal the incoming signal will have to be processed by the rest of the entire chain since the eventual decision needs to be known as displayed in Fig 1.

Another downside of a filter based on LMS is that it will struggle to initialize properly since most of the early decisions will be wrong. A training sequence is therefore required at the start of every frame to aid the correct initialization of the filter. [18]

C. Time domain Adaptive Equalizer

The disturbances acting on the transmitted signal are rapidly varying over time. Therefore the equalizer is most effective if after every sample an error signal is calculated and used to update its filter. Such an adaptive equalizers can be implemented in the time-domain though an adaptive FIR filter.

The taps of this filter are updated using both the input and the output of the signal through

$$h_i(n+1) = h_i(n) + \mu\epsilon(n)x_o^*(n)u_i(n). \tag{3}$$

where

- h_i is a tap-weight.
- μ is the step size parameter.
- x_o^* is the conjugate of the output sample of the output-channel connected to the equalizer.
- ϵ is the error.
- u_i is the sample currently at the tap.

The step size μ will affect how fast the filter will respond to the error ϵ. A lower μ commonly leads to a lower final error at a cost of slower convergence. The step size is commonly between 1×10^{-3} and 1×10^{-5} depending on the requirements. Earlier research has proven that a variable μ may lead to up to a 50% increase in convergence speed for a similar final error in a time domain implementation. A variable μ can be implemented through a lookup table where smaller error ranges are linked to lower μ steps.

[19] [15] [20] [13].

D. Block-Adaptive Frequency Domain Equalizer

A high number of taps such as required for MDM systems causes a huge computational burden for a time domain implementation such as the equalizer proposed previously. [21] The equalizer can however be implemented more efficiently in the frequency domain since the forward and inverse Fourier transforms required for such a compensation scale better with the number of taps.

Another important optimization is implementing the equalizer on a block by block basis using the overlap save method instead of a sample by sample basis. The channel changes slowly enough such that the filter does not need to update after every sample but can happen after a group of samples called a block. For this equalizer an overlap of 50% since it is effective and simple.

To prepare for the loop the blocks are prepared and transformed to the frequency domain using

$$X_i = FFT(x_i(kN_B - N_B \cdots kN_B + N_B - 1). \tag{4}$$

Where

- N_B is the chosen length of the block.
- x_i are the input samples of channel i.

The appended blocks can subsequently be compensated with filter weights H and are transformed back into the time domain. The last block from appended blocks is put on top of the output samples. As can be observed in fig. 3 the output of the other sub-equalizers is added to form

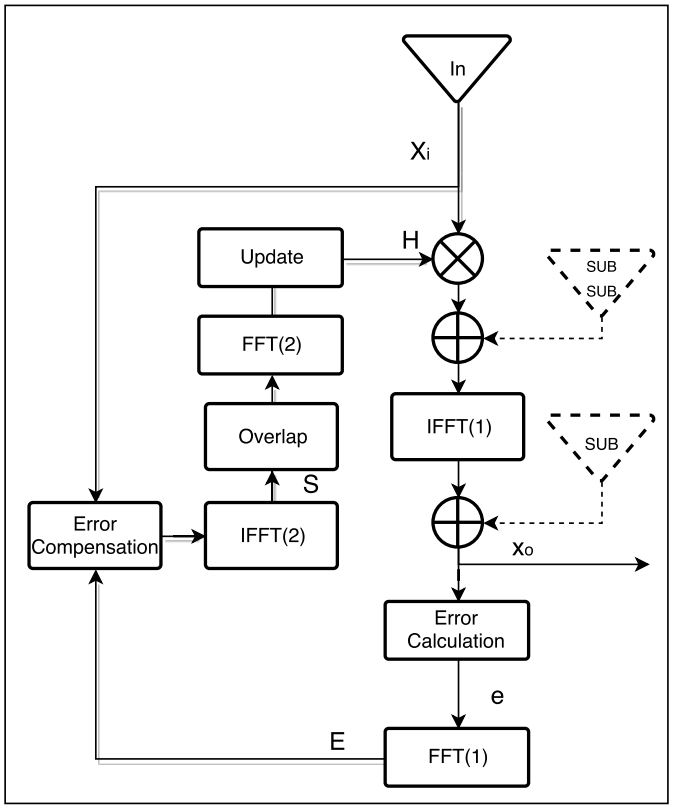


Fig. 3: Overview of the Frequency domain implementation. SUBSUB here represents the output of a subsub-equalizers processing the same input samples, such as the case in oversampled or real-valued system. Sub represents the input of the sub-equalizers connecting the other inputs to this particular output.

$$x_o(k) = x_o(k) + Last\ N_B\ elements\ of\ IFFT(X_i \circ H). \quad (5)$$

This output block will then be used to calculate an error signal e. In the case of CMA

$$e = (j_n - x_o[k] \circ x_o^*[k]) \circ x_o[k]). \quad (6)$$

where j_n is a vector with all ones. This error signal then is appended to a block of zeros and transformed to the frequency domain.

$$E = FFT(O_n; e) \quad (7)$$

where O_n is a block of zeros and ; denotes an append operation. Next a gradient constrain has to be applied Next the error is compensated and the gradient vector s is determined through

$$s = First\ N_B\ elements\ of\ IFFT(E \circ X_i(k)^*) \quad (8)$$

where $(*)$ denotes the complex conjugate. H is then updated with

$$H = H + \mu FFT(s; O_n) \quad (9)$$

[13]

The gradient constraint step from eq. 8 serves to avoid distortions due to the cyclic nature of the FFT. Research has shown however that omitting this step may be possible in some scenarios and can lead to less complexity in the MIMO. [22]

E. Oversampling and Real valued MIMO

The algorithm discussed previously can be extended to support an oversampled signal. In such a configuration equalizers are separated into sub-sub-equalizers for every sample per symbol. Each of these equalizers uses its assigned input samples and the output of the equalizer to update its filter through (6),(7),(8) and (9). To calculate the output of a sub equalizer for a symbol the frequency domain output of the two sub-sub-equalizers are summed. This is visualized using the SUB SUB input in Fig. 3.

A similar extension can be made to implement a real valued MIMO. During the transmission of the signal a gain/phase mismatch may occur causing an imbalance between the Quadrature and the In-phase components of the signal. This impairment can be compensated statically as discussed in [13]. A real valued MIMO however may also be effective at equalizing the IQ-imbalance. Similar to a MIMO with oversampling support, a real valued MIMO employs two sub-equalizers to compensate the Quadrature and In-Phase independently.

IV. Massively Parallel MIMO - (Extension)

An adaptive equalizer works by sequentially processing samples or blocks of samples, a CPU is generally faster at processing sequentially than a GPU. Therefore a single equalizer will rarely perform better on a GPU than a CPU.

However for MDM-systems require multiple adaptive equalizers combined in a butterfly structure to effectively equalize crosstalk. Generally experiments are done ranging from 2 to 12 channels which rapidly increases the computational burden of the equalizer since the MIMO's complexity scales quadratically.

Furthermore block adaptive equalization has been proven to be effective for optical channels which increases the potential for a massively parallel implementation.

Summarizing, in MIMO equalization multiple dimensions of parallelism can be identified in the block length, the input modes, the output modes and the number of samples per symbol. This means that the number of cores the system could use within the loop N_{core} is

$$N_{core} = (Ch)^2 N_B 2 N_S \quad (10)$$

where

- Ch is the number of SDM channels.
- N_B is the length of the block.
- N_s is the amount of samples per symbol.

A. The implementation

Considering all these properties of MIMO equalization makes it an attractive candidate for a massively GPU acceleration. In the massively parallel implementation will benefit

from all the dimensions of parallelism as mentioned earlier. The main strategy behind the implementation will be to do as much as possible in a parallel setup rather than a sequential one.

Employing parallel signal processing is made easy thanks to the work of many developers to bring GPU programming to Python through several open source libraries [23]. These libraries work by generating, compiling and optimizing code for the GPU at runtime. The implementation for this paper is build using the Numba and Pyculib libraries. Using the Pyculib library multiple transforms can be done at the same time in parallel. This is essential since the Fourier transform is key to the frequency domain implementation of the MIMO.

B. Optimization

The Fourier transform of the overlapping blocks required at the start of the algorithm are done all at once rather than sequentially and the results are cached. Doing so will reduce the computation time of the algorithm since one less Fourier transform is required per cycle.

Furthermore the algorithm performs more efficiently for base 2 blocklengths since this speeds up the Fourier transforms. [24]

C. Results

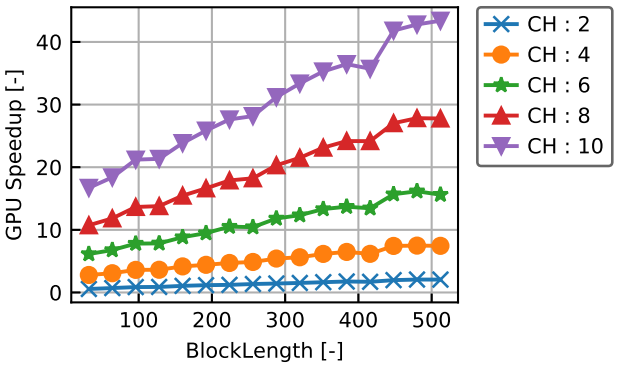
The unit used to evaluate the performance is the time it takes per symbol. The implementations are tested on a Nvidia GTX 780 GPU, and a Intel Xeon-E5-1607 CPU In Fig. 4 the achieved speedup of the GPU implementation is depicted relative to variations in oversampling, the number of SDM channels and blocklengths. The observation can be made that the GPU MIMO can decrease the time significantly when a larger number of SDM channels is involved. It should be noted however that the CPU uses only one core, so in a solution where multi-threading is implemented, the CPU may outperform the GPU in certain scenarios. As expected, the speedup increases quadratically with the number of SDM channels and linearly with the amount of oversampling.

The Performance characteristics of the MIMO implemented on the GPU are displayed in more detail in Fig 5. From this plot can be concluded that the custom kernels developed in Python for this implementation perform rather poorly compared to the Fourier transform.

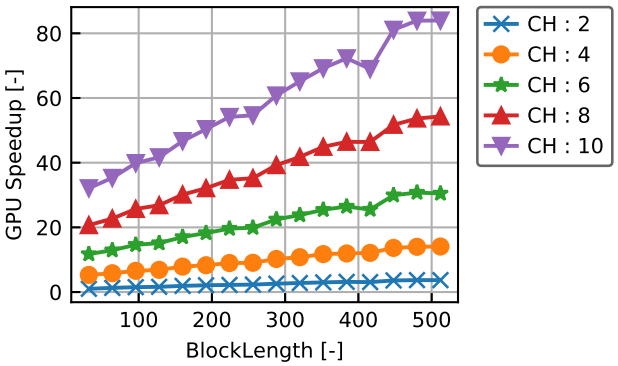
V. Conclusion

This paper discussed the most prevalent impairments for optical fiber communication. These impairments are dynamic and rapidly changing, thus to compensate these impairments a group of adaptive equalizers (MIMO) is required. Supporting multiple inputs and outputs, and a high number of filter taps as required by SDM/MDM systems increases the computational burden on the MIMO. To handle the increased computational burden a massively parallel block adaptive frequency domain implementation of the MIMO is proposed.

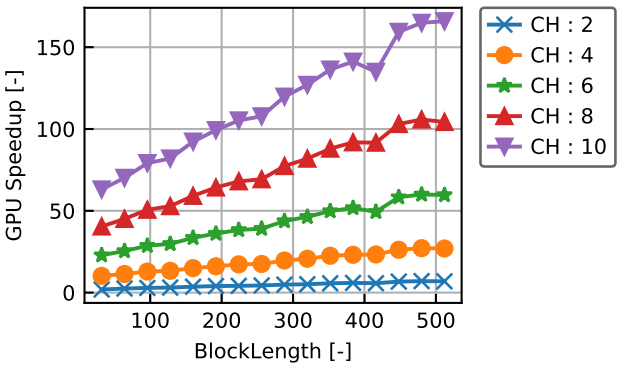
For the proposed implementation, the amount of input/output channels does not affect the computation time, therefore



(a)



(b)



(c)

Fig. 4: The speedup achieved with the GPU implementation relative to the single core CPU implementation for a) symbol spaced sampling, b) twofold oversampling, c) fourfold oversampling. The speedup has been calculated from minimum computation times for the GPU and CPU.

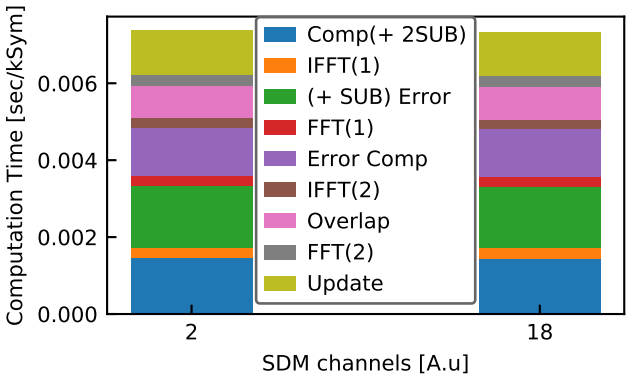


Fig. 5: GPU computation speed (2x oversampled) MIMO for the stages as depicted in Fig. 3. 2SUB here denotes input from the SUBSUB-equalizers.

the proposed implementation is very effective at mitigating the increased computational burden of MDM and SDM systems. Therefore, a speedup of up to 150 times relative to the CPU can be achieved by the GPU, however the benchmarks reveal that larger speedups of the GPU may be possible.

The fast Fourier transform (FFT) in the current implementation is rapid but its limitations bottleneck the implementation. The FFT cannot operate from inside a customized GPU function, therefore after every FFT time is consumed by reorganizing the computation units of the GPU. This increased overhead causes systems that are not well suited for a parallel implementation such as a single mode system to run inefficiently on the GPU implementation.

In the future development of a GPU implementation a better integrated Fourier transform may significantly increase the effectiveness of the GPU implementation of the MIMO, furthermore the findings in this paper may be linked to possible real time-implementations using ASICs or FPGAs, since these chips can be developed to handle the increased computational expensiveness similar to the GPU.

References

[1] "The Zettabyte Era: Trends and Analysis." https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/vni-hyperconnectivity-wp.html

[2] Essiambre, R.-J. R.-J. *et al.*"Capacity Limits of Optical Fiber Networks," *Journal of Lightwave Technology*, vol. 28, no. 4, pp. 662–701, 2010.

[3] Essiambre, R. J. & Tkach, R. W."Capacity trends and limits of optical communication networks," *Proceedings of the IEEE*, vol. 100, no. 5, pp. 1035–1055, 2012.

[4] Winzer, P. J. & Neilson, D. T."From Scaling Disparities to Integrated Parallelism: A Decathlon for a Decade," *Journal of Lightwave Technology*, vol. 35, no. 5, pp. 1099–1115, 2017.

[5] Saitoh, K."Multicore Fiber Technology," *Optical Fiber Communication Conference*, vol. 34, no. 1, p. Th4C.1, 2015. https://www.osapublishing.org/abstract.cfm?uri=OFC-2015-Th4C.1

[6] Ryf, R. *et al.*"Mode-multiplexed transmission over conventional graded-index multimode fibers," *Optics Express*, vol. 23, no. 1, p. 235, 2015. https://www.osapublishing.org/abstract.cfm?URI=oe-23-1-235

[7] ——"Space-division multiplexing over 10 km of three-mode fiber using coherent 6 6 MIMO processing," *Optical Fiber Communication Conference/National Fiber Optic Engineers Conference 2011*, p. PDPB10, 2011. https://www.osapublishing.org/abstract.cfm?uri=NFOEC-2011-PDPB10

[8] Weerdenburg, J. V. *et al.*"138 Tbit / s Mode- and Wavelength Multiplexed Transmission over 6-Mode Graded-Index Fiber," vol. 14, no. 8, pp. 1–6, 2015.

[9] van Weerdenburg, J. *et al.*"10 Spatial mode transmission using low differential mode delay 6-LP fiber using all-fiber photonic lanterns," *Optics Express*, vol. 23, no. 19, p. 24759, 2015. https://www.osapublishing.org/abstract.cfm?URI=oe-23-19-24759

[10] Mohanty, S. P."P-1-2 GPU-CPU Multi-Core For Real-Time Signal Processing," 2009.

[11] Wu, M. *et al.*"Implementation of a high throughput soft MIMO detector on GPU," *Journal of Signal Processing Systems*, vol. 64, no. 1, pp. 123–136, 2011.

[12] Randel, S. *et al.*"First real-time coherent MIMO-DSP for six coupled mode transmission," *2015 IEEE Photonics Conference, IPC 2015*, pp. 1–2, 2015.

[13] Faruk, M. S. & Savory, S. J."Digital Signal Processing for Coherent Transceivers Employing Multilevel Formats," *Journal of Lightwave Technology*, vol. 35, no. 5, pp. 1125–1141, 2017.

[14] Savory, S. J. *et al.*"Electronic compensation of chromatic dispersion using a digital coherent receiver." *Optics express*, vol. 15, no. 5, pp. 2120–2126, 2007.

[15] Sleiffer, V.*Towards petabit per second optical long-haul transmission links using space-division multiplexing technology*, 2014. http://repository.tue.nl/774392

[16] Berdagué, S. & Facq, P."Mode division multiplexing in optical fibers." *Applied optics*, vol. 21, no. 11, pp. 1950–1955, 1982.

[17] Fini, J. M. *et al.*"Crosstalk in multi-core optical fibres," *Ecoc 2011*, vol. 1, p. Mo.1.LeCervin.4., 2011. papers2://publication/uuid/5689D843-F8DF-4527-994E-C7444655D2FC

[18] Fan, Y. *et al.*"The comparison of CMA and LMS equalization algorithms in optical coherent receivers," *2010 6th International Conference on Wireless Communications, Networking and Mobile Computing, WiCOM 2010*, no. 2009, 2010.

[19] van Uden, R. G. H. *et al.*"MIMO equalization with adaptive step size for few-mode fiber transmission systems," *Optics Express*, vol. 22, no. 1, p. 119, 2014. https://www.osapublishing.org/oe/abstract.cfm?uri=oe-22-1-119

[20] Kuschnerov, M. *et al.*"Data-aided versus blind single-carrier coherent receivers," *IEEE Photonics Journal*, vol. 2, no. 3, pp. 387–403, 2010.

[21] van Uden, R.*MIMO Digital Signal Processing for Optical Spatial Division Multiplexed Transmission Systems*, 2014, no. september.

[22] Winzer, P. *et al.*"Complexity Analysis of Adaptive Frequency-Domain Equalization for MIMO-SDM Transmission," *39th European Conference and Exhibition on Optical Communication (ECOC 2013)*, pp. 801–803, 2013. http://digital-library.theiet.org/content/conferences/10.1049/cp.2013.1540

[23] Klöckner, A. *et al.*"PyCUDA and PyOpenCL: A scripting-based approach to GPU run-time code generation," *Parallel Computing*, vol. 38, no. 3, pp. 157–174, 2012.

[24] Greengard, L. & Lee, J.-Y."Accelerating the Nonuniform Fast Fourier Transform," *SIAM Review*, vol. 46, no. 3, pp. 443–454, 2004. http://epubs.siam.org/doi/10.1137/S003614450343200X