# MIMOPack

## *A HPC Library for MIMO Communication Systems*

Carla Ramiro Sánchez[1], Antonio M. Vidal Maciá[1] and Alberto Gonzalez Salvador[2]
[1]*Dpto. De Sistemas Informáticos y Computación (DSIC), Universitat Politècnica de València, Valencia, Spain*
[2]*Instituto de Telecomunicaciones y Aplicaciones Multimedia (iTEAM), Universitat Politècnica de València, Valencia, Spain*

## 1 RESEARCH PROBLEM

Nowadays, several communication standards are emerging and evolving searching higher transmission rates, reliability and coverage. This expansion is primarily driven by the continued increase in consumption of mobile multimedia services due to the emergence of new handheld devices such as smart phones and tablets.

One of the most significant techniques employed to meet these demands is the use of multiple transmit and receive antennas, known as Multiple-Input Multiple-Output (MIMO) systems (Paulraj et al., 2004). The use of this technology allows to increase the transmission rate and the quality of the transmission through the use of multiple antennas at the transmitter and receiver side.

MIMO technologies have become an essential key in several wireless and broadband standards such as Wireless Local Area Network (WLAN), Wordlwide interoperability for Microwave Acces (WiMAX), Long Term Evolution (LTE) and Next Generation Handheld (DVB-NGH), for the reception of digital terrestrial television (TDT) for handheld devices. These technologies will be incorporated also in future standards, therefore is expected in the coming years a great deal of research in this field.

Clearly, the study of MIMO systems is critical in the current investigation, however the problems that arise from this technology are very complex. The detector is present in the receiver side and is the responsible for recover the transmmitted signals (which are affected by the channel fluctuations) with the maximum reliability. This step becomes in many cases the most complex stage in the communication.

Another important factor that affects the performance of a MIMO system is the number of transmit and receive antennas, because as the system grows the communication data processing becomes more complicated. Although the number of antennas currently allowed in the standards is not large, it is expected that in the near future more than 100 transmit antennas will be used (Rusek, 2013), (Lamare, 2013). All the above reasons motivate the search for high-throughput versatile receiver implementations capable to be reconfigured and scalable with the system parameters.

The High Performance Computing (HPC) systems, and specifically, modern hardware architectures as multicore and manycores (e.g Graphic Processing Units (GPU)) are playing a key role in the development of efficient and low-complexity algorithms for MIMO transmissions. Proof of this is that the number of scientific contributions and research projects related to its use has been increased in the last years (Wu, 2011), (Nylanden, 2010), (Falcao, 2009). Also, some high performance libraries have been implemented as tools for researchers or companies involved in the development of future communication standards, for example: IT++ library based on the use of some optimized libraries for multicore processors (IT++, 2013) or the Communications System Toolbox of Matlab (MathWorks, 2014) which use GPU computing. However, there is not a library able to run on a heterogeneous platform using all the available resources whenever possible.

In view of the high computational requirements in MIMO research and the shortage of tools able to satisfy them, we have made a special effort to develop a library to ease the development of adaptable parallel applications in accordance with the different architectures of the executing platform. The library, called MIMOPack, aims to implement efficiently, using parallel computing, a set of functions to perform some of the critical stages in MIMO communication systems. This library can be run over the last generation of machine architectures (e.g GPUs and multicore), or even simultaneously, since it is designed to use on heterogeneous machines to exploit the whole computational capacity thus reducing the response time of the most complex problems.

MIMOPack, may allow industrial and academic researchers to include more complex algorithms in



Sánchez C., Maciá A. and Salvador A..
MIMOPack - A HPC Library for MIMO Communication Systems.

their simulations and obtain its results faster. Moreover, it can be run in a wide range of architectures even many of them callable from MATLAB increasing the portability of developed codes between different computing environments.

# 2 OUTLINE OF OBJECTIVES

The use of MIMO technology has had enormous repercussions in today's telecommunications systems and certainly it will do in the near future. The benefits offered are achieved at the expense of an increase in the material costs to deploy multiple antennas at both the transmitter and the receiver, and also at the expense of additional complexity at the receiver end of the MIMO system. For that reason, signal detection has been the subject of intense study during the last decade and the search for high throughput practical implementations also scalable with the system size remains essential today.

The grand challenge is to develop fast algorithms to optimize the design and the validation process of new MIMO schemes and technologies. Then, this work aims to contribute to meeting this goal. This section describes some particular motivations leading towards the high performance library design.

The use of HPC systems brings big benefits, but it will also poses big challenges. These systems introduce asymmetries and heterogeneities that complicate the development of efficient algorithms. In the recent years, a large variety of machine architectures have appeared, in view of this situation researchers of the scientific community are obliged to write the codes in different programming languages and consider many details of the architecture to use efficiently the whole target system. Therefore, this high performance computing library is essential to facilitate the implementation of scientific codes on a widespread range of architectures.

Furthermore, there are several important companies implicated in the development of new communication standards. In the standardization process different entities are involved such as: administrations, network operators, manufacturers, users, research bodies, universities, consultancy companies, partnerships and others (ETSI, 2014). The objective of a standard is to provide a set of rules, guidelines or characteristics ensuring the interoperability between systems developed by different manufacturers.

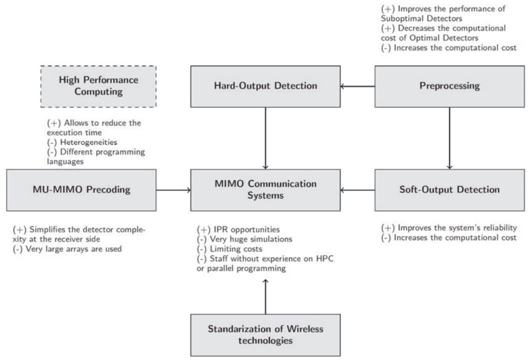


Figure 1: Nature and scope of the thesis.

These manufacturers work to propose their own technologies with the aim that it will be approved as standard. This will allow to enforce their intellectual property rights over its competitors, who will must obtain the corresponding license to incorporate the technology adopted as standard into their products. In many cases, simulation is the only way to get these proposals but these simulations often involve a high computational burden, since they try to simulate the transmission of large amount of bits in order to obtain results close to than would be obtained in a real transmission. Normally these simulations require weeks or even months to be completed. Thereby, *MIMOPack* may allow the launch of large simulations, opening the door for industrial researchers to analyze its technologies faster than its conventional simulation and obtain more patent opportunities than their potential competitors. Taking into account the above presented motivations, the main scope of this thesis is the following (see Figure 1):

- To develop an efficient library of functions able to perform some of the most important and complex stages in a MIMO communication as preprocessing, precoding, detection and decoding.
- To contribute with high-throughput implementations of functions using parallel processing and to evaluate them in terms of speedup, bit error rate and throughput and compare them with other existing implementations.
- To facilitate to the programmer the implementation of codes on a wide range of architectures, incrementing the portability of codes between different computing platforms by using common interfaces for all the considered environments. This approach simplifies the use of the library, regardless of the machine where it will be executed.

- To develop a set of functions highly configurable able to be executed with different simulation parameters and different computational resources.

# 3 STATE OF THE ART

Programming applications over HPC systems allows to reduce the execution time of complex problems, but it use leads serious programming difficulties. The programming challenge involves the developers to know different programming languages and the characteristics of the architecture in depth. In this sense the high performance libraries become in valuable tools for specialists of a particular field, since it facilitate the development of scientific codes.

Some software companies have already released to the market various libraries, these libraries not only facilitate the preparation of code but also exploit the huge computing capabilities of new architectures to accelerate and optimize these implementations. There are several libraries with extensive backgrounds and acceptance in the scientific community, most of them in numerical linear algebra. For example the Linear Algebra PACKage (LAPACK) designed to run efficiently on shared-memory vector and parallel processors (Anderson, 1999). There are also some versions of them, which provide implementations of basic mathematical functions on GPUs as for example CULA (CULA, 2014) or MAGMA (Tomov, 2011).

In the field of communication systems applications, few tools or HPC libraries are available. Nevertheless, we can find two remarkable libraries:

- Communications System Toolbox provides algorithms for designing, simulating, and analyzing communications systems. These capabilities are provided as MATLAB functions, MATLAB System objects, and Simulink blocks. The system toolbox enables source coding, channel coding, interleaving, modulation, equalization, synchronization, and channel modeling. Also allows analyze bit error rates, generate eye and constellation diagrams, and visualize channel characteristics. Although this software is excellent and widely used by the scientific community, nowadays just a small set of functions are prepared to use parallel computing with GPUs.
- IT++ Library: is a C++ library of mathematical, signal processing and communication classes and functions. The kernel of the library consists of generic vector and matrix classes, and a set of accompanying routines. IT++ makes an extensive use of existing open-source or commercial libraries for increased functionality, speed and accuracy (e.g BLAS, LAPACK, FFTW, ATLAS, ACML, etc). However, this library is oriented to its exclusive use on multicore machines; it does not have support to use GPUs.

# 4 METHODOLOGY

The methodology to be followed in this research work is summarized in the following tree phases.

## 4.1 Conceptual Phase

This phase includes from conception of the research problem to the concreteness of objectives.

- Find information about the current MIMO systems and techniques used in different modules existent in the communication process (Channel coding and decoding, preprocessing, precoding, hard/soft detection).
- Bibliographic review on efficient algorithms and high performance libraries for MIMO communication systems.
- Enumerate the main objectives and detect the main motivations leading towards the library implementation.

## 4.2 Methodological Phase

This phase includes the algorithms design and its implementation on high performance hardware.

- Study of the performance of the algorithms implemented sequentially to establish which parts of the code are the computationally most expensive.
- Design of algorithms: The parallelization opportunities of each algorithm will be discussed, and also the possibility of its implementation with heterogeneous multicore + heterogeneous multi-GPU programming.
- Implement the algorithms designed in the previous stage using OpenMP (for multicore systems) and CUDA for GPU devices. Moreover, will be explored the possibility of using numerical linear algebra libraries and

optimized as LAPACK, BLAS, etc in some parts of the code.

### 4.3 Empirical Phase

This phase involves validation, analysis and interpretation of the results obtained with our implementations.

- Analysis of different parameters to obtain the best performance in the design of each module. For example: number of threads, type of distribution of tasks (static or dynamic), and thread block size.
- Validation stage: Unit testing for each function implemented above. These tests allow us to ensure that our code work properly.
- Analyze the performance based on speedup, Bit Error Rate and throughput.
- Perform the same computational analysis on several systems, checking if the library is portable and has a good performance on any type of hardware platform. For example, different types of machines GPUs, other numbers of processors, etc.
- Optimization performance and efficiency of various library functions. The optimization is carried out either if the speedup was not expected, or when necessary to optimize the code for certain fixed parameters specified in a particular standard.

## 5 EXPECTED OUTCOME

As a result of the research work described in previous sections, a C library will be able for simulation of MIMO communication systems. The library will include tools and functions to perform some of the critical stages in the communication process, which can be executed over different types of architectures to ease the development of adaptable parallel applications in accordance with the different architectures of the executing platform:

- Sequential processor
- Multi-core processor
- GPU and Multi-GPU
- Heterogeneous (multicore and GPUs)

For each simulation the library shall allow to configure the following MIMO systems parameters:

- Number of transmitter/receiver antennas
- Signal to Noise Ratio (dB)
- Modulation
- Number of signal vectors to be transmitted
- Variation of the channel

Each function will be executed with the configuration selected by the user:

- Number of CPU threads
- Number of GPUs
- Quantity of workload for GPUs and CPU
- Use or not MKL library whenever possible

## 6 STAGE OF THE RESEARCH

The library is composed by several modules. In Figure 1 we can see the basic simulation chain through the library headers, as we can see the library allows the user to assign the type and the number of resources to use during the execution and get the simulation results to calculate some statistics (e.g simulation time, Bit Error Rate, Symbol Error Rate or throughput).

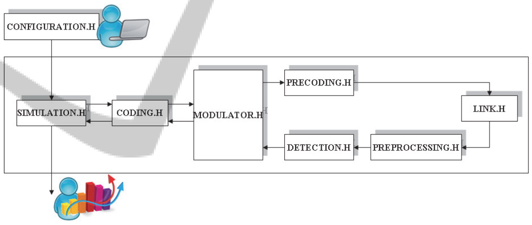


Figure 2: Simulation chain through the MIMOPack library modules.

### 6.1 Software Package Functions

The library is continuously growing; the updated release collects a set of functions to compute the most important and complex stages in a MIMO communication which are described in this section.

#### 6.1.1 Hard-output Detectors

If nearly optimal data detection is desired, the detector becomes often the most computationally expensive algorithm within a MIMO receiver. The detector is responsible of processing the received mixture of signals affected by the channel in order to recover the transmitted data with the accuracy required by the considered application. This issue motivates the search for high-throughput MIMO hard-output detectors capable to be reconfigured and scalable with the system parameters. The hard-output detectors currently available in the library are listed below:

- MLE: MLE-Exhaustive (Agrell, 2002).
- SESD: Schnorr Euchner Sphere Decoder (Schnorr, 1994).
- ZF-SIC: Zero Forcing SIC (Berenguer, 2003).
- HFSD: Hard Fixed Sphere Decoder (Barbero, 2008).
- K-BEST: K-Best Decoder (Guo, 2006).
- ASD: Automatic Sphere Decoder (Su, 2005).

Also, some strategies such as channel matrix preprocessing techniques can be used in order to decrease the computational cost of data detection and are also implemented in *MIMOPack*.

### 6.1.2 Soft-Output Detectors

Error control coding ensures the desired quality of service for a given data rate and is necessary to improve reliability of MIMO systems. Therefore, a good combination of detection MIMO schemes and coding schemes has drawn attention in recent years. The most promising coding schemes are Bit-Interleaved Coded Modulation (BICM) (Caire, 1998). At the transmitter the information bits are encoding using an error-correction code. The soft demodulator provides the reliability information in form of real valued log-likehood ratios (LLR). These values are used by the channel decoder to make final decisions on the transmitted coded bits. Nevertheless, these sophisticated techniques produce a significant increase in the computational cost and require large computational power. The following Soft-Output detectors have been implemented:

- Soft Fixed Sphere Decoder (Barbero, 2008).
- Fully Parallel Fixed-Complexity Detector
- Max-Log Detector (Müller, 2002).
- ML Optimum Detector (Müller, 2002).

### 6.1.3 Multiuser MIMO Precoding

In multi-user (MU) MIMO transmissions, specifically, in the downlink scenario, a base station equipped with multiple antennas transmits information to several independent users. The detection process becomes more complicated due to the absence of cooperation between them. In order to simplify the detector complexity at the receiver side, several precoding techniques were devised by various authors and have been included in the library (Windpassinger, 2004), (Yao, 2002):

- Zero-Forcing Precoding
- Tomlinson-Harashima.
- Lattice Reduction Aided Tomlison-Harashima
- Enhanced Lattice Reduction Aided

## 6.2 Preliminary Performance Results

In order to asses the performance of the library, we have evaluated the execution time of a set of Hard-Output detectors. The tests are executed on a platform with one Nvidia Tesla K20Xm GPU with 14 SM, each SM including 192 cores. The core frequency is 0.73 GHz. The GPU has 5GB of GDDR5 global memory and 48KB of shared memory per block. The installed CUDA toolkit is 5.5. The Nvidia card is mounted on a PC with two Intel Xeon CPU E5-2697 at 2.70 GHz with 12 cores and hyperthreading activated.

We consider as simple simulation example of a MIMO system with 6 transmit and receive antennas, 16-QAM symbol alphabet and a constant channel during the entire simulation. The speedup is defined as the ratio between the computational time resulting of executing the simulation of 50000 signals on the sequential CPU (with one OpenMP thread) and the time to execute the same simulation on a multicore and GPU system.

By linking the code with MIMOPack library, we can easily reduce the execution time of our simulations significantly. This execution time can be decreased even more using the heterogeneous mode (i.e multicore and GPU simultaneously).

Table 1: Runtime of Hard-Output MIMOPack detectors with different library configurations.

| Detector | | Runtime (sec) | | |
|---|---|---|---|---|
| | | Sequential | 48 Threads | GPU |
| MLE | | $1.48 \cdot 10^5$ | $7.97 \cdot 10^3$ | $5.69 \cdot 10^3$ |
| SESD | | 89.08 | 3.88 | 12.80 |
| L = 1 | HFSD | 0.45 | 0.13 | 0.34 |
| | 2-BEST | 0.48 | 0.07 | 0.50 |
| | 4-BEST | 0.82 | 0.14 | 0.74 |
| | 16-BEST | 3.31 | 0.26 | 1.64 |
| L = 3 | HFSD | 64.70 | 3.24 | 3.71 |
| | 2-BEST | 15.98 | 0.81 | 3.86 |
| | 4-BEST | 16.49 | 0.84 | 4.15 |
| | 16-BEST | 20.45 | 0.89 | 5.02 |
| | 64-BEST | 37.07 | 1.51 | 7.01 |

As we can see in Table 1, we have a good speedup for the parallel versions for all kind of detectors. MLE GPU version exhibits a good performance even better than that obtained with multicore version. However, due to the low complexity and the non parallel pattern of other detectors especially of the suboptimal ones (ZF-SIC and HFSD) methods, the OpenMP version obtains better performance than CUDA version.

This gain gradually disappears when the complexity of the detector increases, for example with the number of levels to be fully expanded (L) or increasing the number of survivors (K) to be computed in each level in the K-BEST detector.

The variety of detectors with mixed complexities and performances allows to cover multiple use cases with different channel conditions and scenarios such as massive MIMO. Moreover, parallel implementations allow the execution of large simulations over different architectures thus exploiting the capacity of the modern machines.

# 7 CONCLUSIONS

This thesis is focused in the development of a high performance library for MIMO communications systems which aims to provide a set of routines needed to perform the most complex stages in the current wireless communications. The proposed library has three important features: portable, efficiently and user friendly. Results obtained with the efficient hard-output detectors presented in this paper demonstrate that MIMOPack library may become in a very useful tool for companies involved in the development of new wireless and broadband standards which need obtain results and statistics of its proposals quickly and also for other researchers making easier the implementation of scientific codes.

# ACKNOWLEDGEMENTS

This work has been supported by SP20120646 project of Universitat Politècnica de València, by ISIC/2012/006 and PROMETEO FASE II 2014/003 projects of Generalitat Valenciana; and has been supported by European Union ERDF and Spanish Government through TEC2012-38142-C04-01.

# REFERENCES

Paulraj A.J., Gore D.A., Nabar R.U., Bölcskei H., 2004. *An overview of MIMO communications-a key to gigabit wireless*. Proceedings of the IEEE, 92(2):198-218.

Rusek F., Persson D., Lau B., Larsson E., Marzetta T., Edfors O., Tufvesso F., 2013. *Scaling Up MIMO: Opportunities and Challenges with Very Large Arrays.* IEEE Signal Processing Magazine, 30(1):40-60.

Lamare R.C., 2013. *Massive MIMO Systems: Signal Processing Challenges and Research Trends*. URSI Radio Science Bulletin.

Wu M., Sun Y., Gupta S., Cavallaro J., 2011. *Implementation of a high throughput soft MIMO detector on GPU*. Journal of Signal Processing Systems, 64(2):123-136.

Nylanden T., Janhunen J., Silven O., Juntti M., 2010. *A GPU implementation for two MIMO-OFDM detectors*. International Conference on Embedded Computer Systems.

Falcao G., Silva V., Sousa L., 2009. *How GPUs can outperform ASICs for Fast LDPC decoding*. International Conference of Supercomputing.

IT++., 2014. *IT++ User's guide*. http://itpp.sourceforge .net /4.3.1/users_guide.html.

MathWorks., 2014. *Communication System Toolbox. User's guide Version 6.5*. http://jp.mathworks.com/ help/pdf_doc/comm/comm.pdf.

ETSI, 2014. *European telecommunications standards institute Members*, http://www.etsi.org/index.php /membership.

Anderson E., Bai Z., Bishof C., Demmel J., Dongarra J., 1999. *Lapack User Guide. Third Edition*. http://www.netlib.org/lapack/lug/

CULA., 2014. *CULA Programmer's Guide*. http://www.culatools.com/

Tomov S., Nath R., Du P., Dongarra J., 2011 *MAGMA Users' Guide.* http://icl.cs.utk.edu/projectsfiles/ magma/doxygen/

Agrell E., Eriksson T., Vardy A., Zeger K., 2002. *Closest point search in lattices*. IEEE Transactions on Information Theory, 48(8):2201-2214.

Schnorr C., Euchner M., 1994. *Lattice basis reduction: Improved practical algorithms and solving subset sum problems*. Mathematical Programming, 66(2):181-191.

Berenguer I., Wang X., 2003. *Space-Time coding and signal processing for MIMO communications*. Journal of Computer Science and Technology, 18(6):689-702.

Barbero L.G., Thompson J.S., 2008. *Fixing the complexity of the sphere decoder for MIMO detection*, IEEE Transactions on Wireless Communications, 7(6): 2131-2142.

Guo Z., Nilsson P., 2006. *Algorithm and implementation of the K-Best Sphere Decoding for MIMO Detection.* IEEE Journal on Selected Areas in Communications, 24(3):491-503., March 2006.

Su K., 2005. *Efficient Maximum Likelihood detection for communication over MIMO channels*. University of Cambridge, Technical Report.

Caire G., Taricco G., Biglieri E., 1998. *Bit-interleaved coded modulation.* IEEE Transactions on Information Theory, 44(3):927-946.

Barbero L.G., Ratnarajah T., Cowan C., 2008. *A low-*

*complexity soft-MIMO detector based on the fixed-complexity sphere decoder*. IEEE International Conference on Acoustics, Speech and Signal Processing. Las Vegas, Nevada (USA).

Müller-Weinfurtner S.H., 2002. *Coding approaches for multiple antenna transmission in fast fading and OFDM*. IEEE Transactions on Signal Processing, 50: 2442-2450.

Windpassinger C., Fischer R., Vencel T., H., Huber, 2004. *Precoding in multiantenna and multiuser communications*. IEEE Transactions on Communications, 3(4):2057-2060.

Yao H., Wornell G., 2002. *Lattice-reduction-aided detectors for MIMO communication systems*. IEEE Global Communications Conference, Taipei, Taiwan.