Filter optimization for real time digital processing of radiofrequency signals: application to oscillator metrology

A. Hugeat*†, J. Bernard†, G. Goavec-Mérou*, P.-Y. Bourgeois*, J.-M. Friedt*
*FEMTO-ST, Time & Frequency department, Besançon, France
†FEMTO-ST, Computer Science department DISC, Besançon, France
Email: {pyb2,jmfriedt}@femto-st.fr

***Abstract*—Software Defined Radio (SDR) provides stability, flexibility and reconfigurability to radiofrequency signal processing. Applied to oscillator characterization in the context of ultrastable clocks, stringent filtering requirements are defined by spurious signal or noise rejection needs. Since real time radiofrequency processing must be performed in a Field Programmable Array to meet timing constraints, we investigate optimization strategies to design filters meeting rejection characteristics while limiting the hardware resources required and keeping timing constraints within the targeted measurement bandwidths.**

***Index Terms*—Software Defined Radio, Mixed-Integer Linear Programming, Finite Impulse Response filter**

I. Digital signal processing of ultrastable clock signals

Analog oscillator phase noise characteristics are classically performed by downconverting the radiofrequency signal using a saturated mixer to bring the radiofrequency signal to baseband, followed by a Fourier analysis of the beat signal to analyze phase fluctuations close to carrier. In a fully digital approach, the radiofrequency signal is digitized and numerically downconverted by multiplying the samples with a local numerically controlled oscillator (Fig. 1) [1].

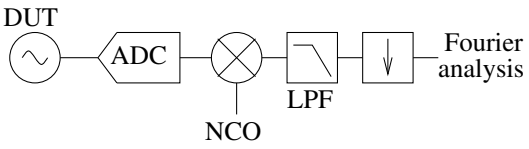

Fig. 1. Fully digital oscillator phase noise characterization: the Device Under Test (DUT) signal is sampled by the radiofrequency grade Analog to Digital Converter (ADC) and downconverted by mixing with a Numerically Controlled Oscillator (NCO). Unwanted signals and noise aliases are rejected by a Low Pass Filter (LPF) implemented as a cascade of Finite Impulse Response (FIR) filters. The signal is then decimated before a Fourier analysis displays the spectral characteristics of the phase fluctuations.

As with the analog mixer, the non-linear behavior of the downconverter introduces noise or spurious signal aliasing as well as the generation of the frequency sum signal in addition to the frequency difference. These unwanted spectral characteristics must be rejected before decimating the data stream for the phase noise spectral characterization [2]. The characteristics introduced between the downconverter and the decimation processing blocks are core characteristics of an oscillator characterization system, and must reject out-of-band signals below the targeted phase noise – typically in the sub -170 dBc/Hz for ultrastable oscillator we aim at characterizing. The filter blocks will use most resources of the Field Programmable Gate Array (FPGA) used to process the radiofrequency datastream: optimizing the performance of the filter while reducing the needed resources is hence tackled in a systematic approach using optimization techniques. Most significantly, we tackle the issue by attempting to cascade multiple Finite Impulse Response (FIR) filters with tunable number of coefficients and tunable number of bits representing the coefficients and the data being processed.

II. Finite impulse response filter

We select FIR filter for their unconditional stability and ease of design. A FIR filter is defined by a set of weights b_k applied to the inputs x_k through a convolution to generate the outputs y_k

$$y_n = \sum_{k=0}^{N} b_k x_{n-k}$$

As opposed to an implementation on a general purpose processor in which word size is defined by the processor architecture, implementing such a filter on an FPGA offer more degrees of freedom since not only the coefficient values and number of taps must be defined, but also the number of bits defining the coefficients and the sample size. For this reason, and because we consider pipeline processing (as opposed to First-In, First-Out FIFO memory batch processing) of radiofrequency signals, High Level Synthesis (HLS) languages [3] are not considered but the problem is tackled at the Very-high-speed-integrated-circuit Hardware Description Language (VHDL) level. Since latency is not an issue in a openloop phase noise characterization instrument, the large numbre of taps in the FIR, as opposed to the shorter Infinite Impulse Response (IIR) filter, is not considered as an issue as would be in a closed loop system.

The coefficients are classically expressed as floating point values. However, this binary number representation is not efficient for fast arithmetic computation by an FPGA. Instead, we select to quantify these floating point values into integer values. This quantization will result in some precision loss.

The tradeoff between quantization resolution and number of coefficients when considering integer operations is not trivial. As an illustration of the issue related to the relation between number of fiter taps and quantization, Fig. 2 exhibits a 128-coefficient FIR bandpass filter designed using floating point numbers (blue). Upon quantization on 6 bit integers, 60 of the 128 coefficients in the beginning and end of the taps become null, making the large number of coefficients irrelevant and allowing to save processing resource by shrinking the filter length. This tradeoff aimed at minimizing resources to reach a given rejection level, or maximizing out of band rejection

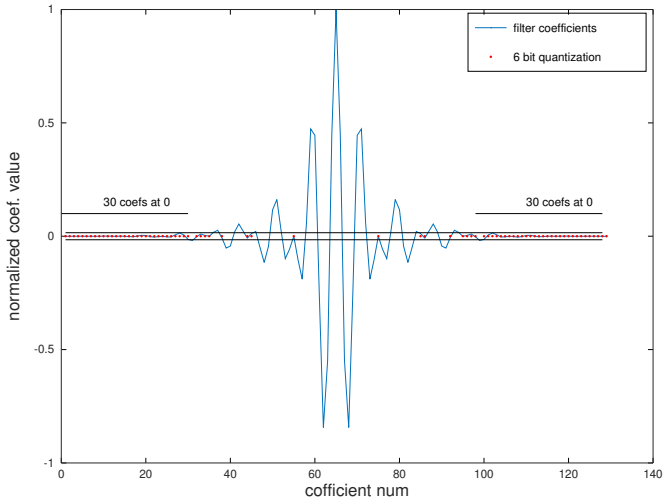

Fig. 2. Impact of the quantization resolution of the coefficients: the quantization is set to 6 bits – with the horizontal black lines indicating ± 1 least significant bit – setting the 30 first and 30 last coefficients out of the initial 128 band-pass filter coefficients to 0 (red dots).

for a given computational resource, will drive the investigation on cascading filters designed with varying tap resolution and tap length, as will be shown in the next section. Indeed, our development strategy closely follows the skeleton approach [4], [5], [6] in which basic blocks are defined and characterized before being assembled [7] in a complete processing chain. In our case, assembling the filter blocks is a simpler block combination process since we assume a single value to be processed and a single value to be generated at each clock cycle. The FIR filters will not be considered to decimate in the current implementation: the decimation is assumed to be located after the FIR cascade at the moment.

III. Filter optimization

A basic approach for implementing the FIR filter is to compute the transfer function of a monolithic filter: this single filter defines all coefficients with the same resolution (number of bits) and processes data represented with their own resolution. Meeting the filter shape requires a large number of coefficients, limited by resources of the FPGA since this filter must process data stream at the radiofrequency sampling rate after the mixer.

An optimization problem [8] aims at improving one or many performance criteria within a constrained resource environment. Amongst the tools developed to meet this aim, Mixed-Integer Linear Programming (MILP) provides the framework to formally define the stated problem and search for an optimal use of available resources [9], [10].

First we need to ensure that our problem is a real optimization problem. When designing a processing function in the FPGA, we aim at meeting some requirement such as the throughput, the computation time or the noise rejection noise. However, due to limited resources to design the process like BRAM (high performance RAM), DSP (Digital Signal Processor) or LUT (Look Up Table), a tradeoff must be generally searched between performance and available computational resources: optimizing some criteria within finite, limited resources indeed matches the definition of a classical optimization problem.

Specifically the degrees of freedom when addressing the problem of replacing the single monolithic FIR with a cascade of optimized filters are the number of coefficients N_i of each filter i, the number of bits C_i representing the coefficients and the number of bits D_i needed to represent the data x_k fed to each filter as provided by the acquisition or previous processing stage. Because each FIR in the chain is fed the output of the previous stage, the optimization of the complete processing chain within a constrained resource environment is not trivial. The resource occupation of a FIR filter is considered as $C_i \times N_i$ which aims at approximating the number of bits needed in a worst case condition to represent the output of the FIR. Indeed, the number of bits generated by the ith FIR is $(C_i + D_i) \times \log_2(N_i)$, but the $\log$ function is avoided for its incompatibility with a linear programming description, and the simple product is approximated as the number of gates needed to perform the calculation. Such an occupied area estimate assumes that the number of gates scales as the number of bits and the number of coefficients, but does not account for the detailed implementation of the hardware. Indeed, various FPGA implementations will provide different hardware functionalities, and we shall consider at the end of the design a synthesis step using vendor software to assess the validity of the solution found. As an example of the limitation linked to the lack of detailed hardware consideration, Block Random Access Memory (BRAM) used to store filter coefficients are not shared amongst filters, and multiplications are most efficiently implemented by using DSP blocks whose input word size is finite. DSPs are a scarce resource to be saved in a practical implementation. Keeping a high abstraction on the resource occupation is nevertheless selected in the following discussion in order to leave enough degrees of freedom in the problem to try and find original solutions: too many constraints in the initial statement of the problem leave little room for finding an optimal solution.

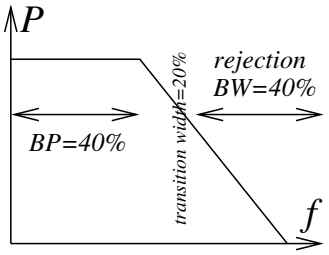

Fig. 3. Shape of the filter transmitted power P as a function of frequency: the bandpass BP is considered to occupy the initial 40% of the Nyquist frequency range, the stopband the last 40%, allowing 20% transition width.

Following these considerations, the model is expressed as:

$$\begin{cases} \mathcal{R}_i &= \mathcal{F}(N_i, C_i) \\ \mathcal{A}_i &= N_i \times C_i \\ \Delta_i &= \Delta_{i-1} + \mathcal{P}_i \end{cases} \quad (1)$$

To explain the system 1, $\mathcal{R}_i$ represents the stopband rejection dependence with N_i and C_i, $\mathcal{A}_i$ is a theoretical area occupation of the processing block on the FPGA as discussed earlier, and Δ_i is the total rejection for the current stage i. Since the function $\mathcal{F}$ cannot be explictly expressed, we run simulations to determine the rejection depending on N_i and C_i. However, selecting the right filter requires a clear definition of the

rejection criterion. Selecting an incorrect criterion will lead the linear program solver to produce a solution which might not meet the user requirements. Hence, amongst various criteria including the mean or median value of the FIR response in the stopband as will be illustrated lated (section III-B), we have designed a criterion aimed at avoiding ripples in the passband and considering the maximum of the FIR spectral response in the stopband (Fig. 3). The bandpass criterion is defined as the sum of the absolute values of the spectral response in the bandpass, reminiscent of a standard deviation of the spectral response: this criterion must be minimized to avoid ripples in the passband. The stopband transfer function maximum must also be minimized in order to improve the filter rejection capability. Weighing these two criteria allows designing the linear program to be solved.

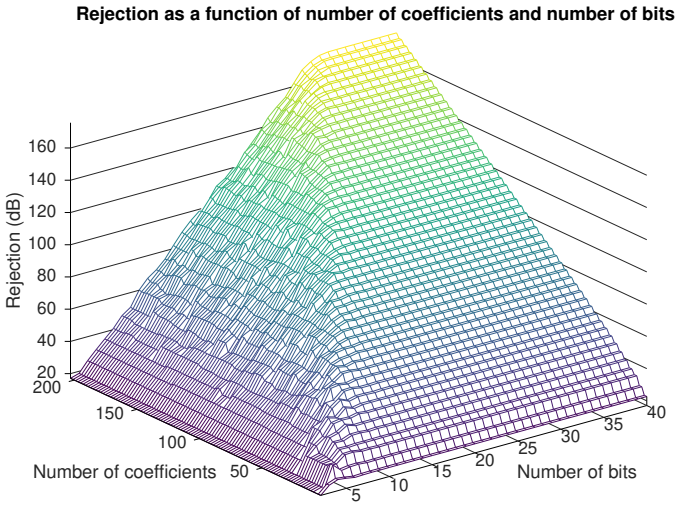

Fig. 4. Rejection as a function of number of coefficients and number of bits

The objective function maximizes the noise rejection ($\max(\Delta_{i_{\max}})$) while keeping resource occupation below a user-defined threshold, or as will be discussed here, aims at minimizing the area needed to reach a given rejection ($\min(S_q)$ in the forthcoming discussion, Eqs. 2 and 3). The MILP solver is allowed to choose the number of successive filters, within an upper bound. The last problem is to model the noise rejection. Since filter noise rejection capability is not modeled with linear equations, a look-up-table is generated for multiple filter configurations in which the C_i, D_i and N_i parameters are varied: for each one of these conditions, the low-pass filter rejection is stored as computed by the frequency response of the digital filter (Fig. 4). Various rejection criteria have been investigated, including mean value of the stopband response, median value of the stopband response, or as finally selected, maximum value in the stopband. An intuitive analysis of the chart of Fig. 4 hints at an optimum set of tap length and number of bit for representing the coefficients along the line of the pyramidal shaped rejection capability function.

Linear program formalism for solving the problem is well documented: an objective function is defined which is linearly dependent on the parameters to be optimized. Constraints are expressed as linear equations and solved using one of the available solvers, in our case GLPK[11]. With the notations used in the description of system 1, we have defined the linear problem as:

a) Variables:

$x_{i,j} \in \{0,1\}$ i is a given filter
j is the stage
If $x_{i,j}$ is equal to 1, the filter is selected

b) Constants:

$\mathcal{F} = \{F_1 ... F_p\}$ All possible filters
p is the number of different filters
$\mathcal{S}_{\max}$ Total space available inside the FPGA

c) Constraints:

$$1 \leq i \leq p$$
$$1 \leq j \leq q \; q \text{ is the max of filter stage}$$
$$\forall j, \sum_i x_{i,j} = 1 \text{ At most one filter by stage}$$
$$\mathcal{S}_0 = 0 \text{ initial occupation}$$
$$\forall j, \mathcal{S}_j = \mathcal{S}_{j-1} + \sum_i (x_{i,j} \times \mathcal{A}_i) \quad (2)$$
$$\mathcal{S}_j \leq \mathcal{S}_{\max}$$
$$\mathcal{N}_0 = 0 \text{ initial rejection}$$
$$\forall j, \mathcal{N}_j = \mathcal{N}_{j-1} + \sum_i (x_{i,j} \times \mathcal{R}_i) \quad (3)$$
$$\mathcal{N}_q \geqslant 160 \text{ an user defined bound}$$
(e.g. 160 dB here)

d) Goal:

$$\min \mathcal{S}_q$$

The constraint 2 means the occupation for the current stage j depends on the previous occupation and the occupation of current selected filter (it is possible that no filter is selected for this stage). And the second one 3 means the same thing but for the rejection, the rejection depends the previous rejection plus the rejection of selected filter.

A. Low bandpass ripple and maximum rejection criteria

The MILP solver provides a solution to the problem by selecting a series of small FIR with increasing number of bits representing data and coefficients as well as an increasing number of coefficients, instead of a single monolithic filter.

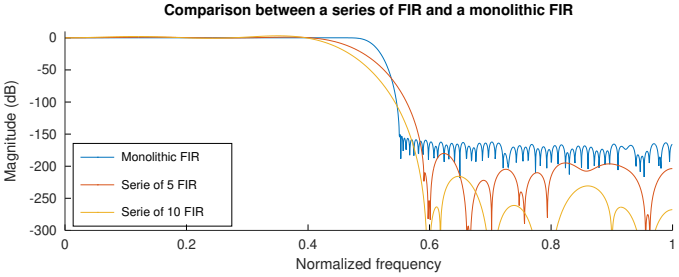

Fig. 5. Comparison of the rejection capability between a series of FIR and a monolithic FIR with a cutoff frequency set at half the Nyquist frequency.

Fig. 5 exhibits the performance comparison between one solution and a monolithic FIR when selecting a cutoff frequency of half the Nyquist frequency: a series of 5 FIR and a series of 10 FIR with the same space usage are provided as selected by the MILP solver. The FIR cascade provides improved rejection than the monolithic FIR at the expense of a lower cutoff frequency which remains to be tuned or compensated for.

The resource occupation when synthesizing such FIR on a Xilinx FPGA is summarized as Tab. I. We have considered a set of resources representative of the hardware platform we work on, Avnet's Zedboard featuring a Xilinx XC7Z020-CLG484-1 Zynq System on Chip (SoC). The results reported in Tab. I emphasize that implementing the monolithic single FIR is impossible due to the insufficient hardware resources (exhausted LUT resources), while the FIR cascading 5 or 10 filters fit in the available resources. However, in all cases the DSP resources are fully used: while the design can be synthesized using Xilinx proprietary Vivado 2016.2 software, implementing the design fails due to the excessive resource usage preventing routing the signals on the FPGA. Such results emphasize on the one hand the improvement prospect of the optimization procedure by finding non-trivial solutions matching resource constraints, but on the other hand also illustrates the limitation of a model with an abstraction layer that does not account for the detailed architecture of the hardware.

TABLE I
RESOURCE OCCUPATION ON A XILINX ZYNQ-7000 SERIES FPGA WHEN SYNTHESIZING THE FIR CASCADE IDENTIFIED AS OPTIMAL BY THE MILP SOLVER WITHIN A FINITE RESOURCE CRITERION. THE LAST LINE REFERS TO AVAILABLE RESOURCES ON A ZYNQ-7020 AS FOUND ON THE ZEDBOARD.

FIR	BlockRAM	LookUpTables	DSP	rejection (dB)
1 (monolithic)	1	76183	220	-162
5	5	18597	220	-160
10	8	24729	220	-161
Zynq 7020	**420**	**53200**	**220**	

B. Alternate criteria

Fig. 5 provides FIR solutions matching well the targeted transfer function, namely low ripple in the bandpass defined as the first 40% of the frequency range and maximum rejection of 160 dB in the last 40% stopband. We illustrate now, for demonstrating the need to properly select the optimization criterion, two cases of poor filter shapes obtained by selecting the mean value and median value of the rejection, with no consideration for the ripples in the bandpass. The results of the optimizations, in these cases, are shown in Figs. 6 and 7.

In the case of the mean value criterion (Fig. 6), the solution is not acceptable since the notch at the end of the transition band compensates for some unacceptable rise in the rejection close to the Nyquist frequency. Applying such a filter might yield excessive high frequency spurious components to be aliased at low frequency when decimating the signal. Similarly, the lack of criterion on the bandpass shape induces a shape with poor flatness and and slowly decaying transfer function starting to attenuate spectral components well before the transition band starts. Such issues are partly aleviated by replacing a mean rejection value with a median rejection value (Fig. 7) but solutions remain unacceptable for the reasons

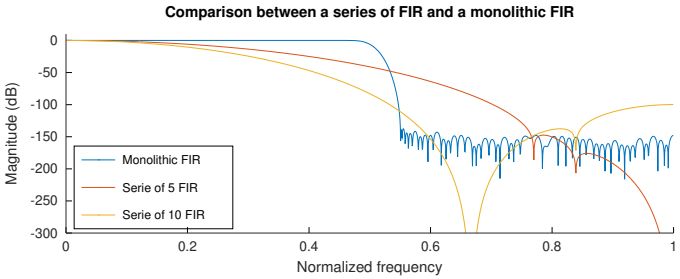

Fig. 6. Comparison of the rejection capability between a series of FIR and a monolithic FIR with a cutoff frequency set at half the Nyquist frequency.

stated previously and much poorer than those found with the maximum rejection criterion selected earlier (Fig. 5).

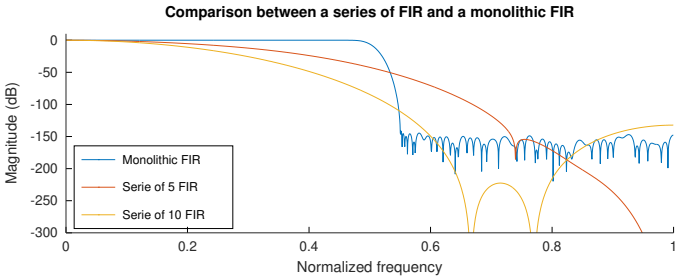

Fig. 7. Comparison of the rejection capability between a series of FIR and a monolithic FIR with a cutoff frequency set at half the Nyquist frequency.

IV. FILTER COEFFICIENT SELECTION

The coefficients of a single monolithic filter are computed as the impulse response of the filter transfer function, and practically approximated by a multitude of methods including least square optimization (Matlab's `firls` function), Hamming or Kaiser windowing (Matlab's `fir1` function).

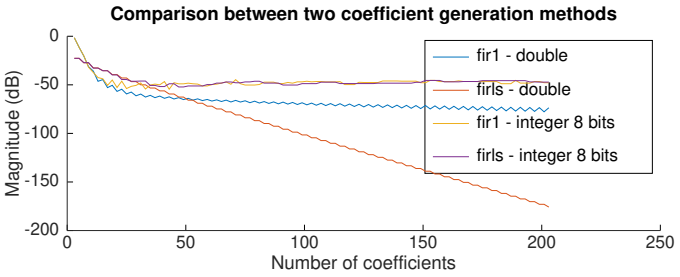

Fig. 8. Evolution of the rejection capability of least-square optimized filters and Hamming FIR filters as a function of the number of coefficients, for floating point numbers and 8-bit encoded integers.

Cascading filters opens a new optimization opportunity by selecting various coefficient sets depending on the number of coefficients. Fig. 8 illustrates that for a number of coefficients ranging from 8 to 47, `fir1` provides a better rejection than `firls`: since the linear solver increases the number of coefficients along the processing chain, the type of selected filter also changes depending on the number of coefficients and evolves along the processing chain.

V. CONCLUSION

We address the optimization problem of designing a low-pass filter chain in a Field Programmable Gate Array for improved noise rejection within constrained resource occupation, as needed for real time processing of radiofrequency

signal when characterizing spectral phase noise characteristics of stable oscillators. The flexibility of the digital approach makes the result best suited for closing the loop and using the measurement output in a feedback loop for controlling clocks, e.g. in a quartz-stabilized high performance clock whose long term behavior is controlled by non-piezoelectric resonator (sapphire resonator, microwave or optical atomic transition).

ACKNOWLEDGEMENT

This work is supported by the ANR Programme d'Investissement d'Avenir in progress at the Time and Frequency Departments of the FEMTO-ST Institute (Oscillator IMP, First-TF and Refimeve+), and by Région de Franche-Comté. The authors would like to thank E. Rubiola, F. Vernotte, and G. Cabodevila for support and fruitful discussions.

REFERENCES

[1] J. A. Sherman and R. Jördens, "Oscillator metrology with software defined radio," *Review of Scientific Instruments*, vol. 87, no. 5, p. 054711, 2016.

[2] C. Andrich, A. Ihlow, J. Bauer, N. Beuster, and G. Del Galdo, "High-precision measurement of sine and pulse reference signals using software-defined radio," *IEEE Transactions on Instrumentation and Measurement*, vol. 67, no. 5, pp. 1132–1141, May 2018.

[3] S. J. Kasbah, I. W. Damaj, and R. A. Haraty, "Multigrid solvers in reconfigurable hardware," *Journal of Computational and Applied Mathematics*, vol. 213, no. 1, pp. 79–94, 2008.

[4] D. Crookes, K. Alotaibi, A. Bouridane, P. Donachy, and A. Benkrid, "An environment for generating FPGA architectures for image algebra-based algorithms," in *Image Processing, 1998. ICIP 98. Proceedings. 1998 International Conference on*. IEEE, 1998, pp. 990–994.

[5] D. Crookes, K. Benkrid, A. Bouridane, K. Alotaibi, and A. Benkrid, "Design and implementation of a high level programming environment for FPGA-based image processing," *IEE Proceedings-Vision, Image and Signal Processing*, vol. 147, no. 4, pp. 377–384, 2000.

[6] K. Benkrid, D. Crookes, and A. Benkrid, "Towards a general framework for FPGA based image processing using hardware skeletons," *Parallel Computing*, vol. 28, no. 7, pp. 1141–1154, 2002.

[7] K. Benkrid, S. Belkacemi, and A. Benkrid, "Hide: A hardware intelligent description environment," *Microprocessors and Microsystems*, vol. 30, no. 6, pp. 283–300, 2006.

[8] J. Y. Leung, *Handbook of scheduling: algorithms, models, and performance analysis*. CRC Press, 2004.

[9] Y. J. Yu and Y. C. Lim, "Design of linear phase FIR filters in subexpression space using mixed integer linear programming," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 54, no. 10, pp. 2330–2338, 2007.

[10] D. Kodek, "Design of optimal finite wordlength FIR digital filters using integer programming techniques," *IEEE Transactions on Acoustics, Speech, and Signal Processing*, vol. 28, no. 3, pp. 304–308, 1980.

[11] "https://www.gnu.org/software/glpk/," availble online, accessed May 2018.