Computational Analysis of a Fast Algorithm for High-order Sparse Linear Prediction

Jensen, Tobias Lindstrøm; Giacobello, Daniele; van Waterschoot, Toon; Christensen, Mads Græsbøll

Published in:
Signal Processing Conference (EUSIPCO), 2016 24th European

DOI (link to publication from Publisher):
10.1109/EUSIPCO.2016.7760413

Publication date:
2016

Document Version
Accepted author manuscript, peer reviewed version

COMPUTATIONAL ANALYSIS OF A FAST ALGORITHM FOR HIGH-ORDER SPARSE LINEAR PREDICTION

Tobias Lindstrøm Jensen[1]*, Daniele Giacobello*[2]*, Toon van Waterschoot*[3]*, Mads Græsbøll Christensen*[4]

[1]Signal and Information Processing, Aalborg Universitet, Denmark
[2]Codec Technologies R&D, DTS Inc., Calabasas, CA, USA
[3]Department of Electrical Engineering (ESAT-STADIUS/ETC), KU Leuven, Belgium
[4]Audio Analysis Lab., AD:MT, Aalborg Universitet, Denmark

tlj@es.aau.dk, giacobello@ieee.org, toon.vanwaterschoot@esat.kuleuven.be, mgc@create.aau.dk

ABSTRACT

Using a sparsity promoting convex penalty function on high-order linear prediction coefficients and residuals has shown to result in improved modeling of speech and other signals as this addresses the inherent limitations of standard linear prediction methods. However, this new formulation is computationally more demanding which may limit its use, in particular for embedded signal processing. This paper analyzes the algorithmic and computational aspects of the matrix structures associated with an alternating direction method of multipliers algorithm for solving the convex high-order sparse linear prediction problem. The paper also analyzes the inherent trade-off between accuracy and the objective measure of prediction gain and shows that a few iterations are sufficient to achieve similar results as computationally more expensive interior-point methods.

Index Terms— Sparse linear prediction, speech and audio processing, convex optimization, linear programming, embedded optimization.

1. INTRODUCTION

Sparse linear prediction (SLP) [1, 2] revisits the linear prediction (LP) framework [3, 4], one of the most successful tools for the analysis and coding of speech signals, in light of the developments that took place in the recent years in the field of convex optimization and sparse representations. While software packages like, e.g., CVX+SeDuMi [5, 6] and ℓ_1-MAGIC [7] allow reproducing the core of the research done by the authors, e.g., [8,9], in a few lines of code, serious efforts to make these algorithms run faster and, possibly, in a real-time platform is not an easy task and a currently a matter of research in signal processing [10, 11].

SLP has proved to be an interesting alternative to classic LP by allowing better statistical models and more meaningful signal representation finding its way in various applications ranging from time-frequency representation [12] and filter design [13] to more application specific tasks, like, e.g., speech dereverberation [14], video processing [15], geology [16], and radar imaging [17]. The compact representation that linear prediction provides reveals a model for the process, called the autoregressive (AR) model. This model is very useful conceptually by representing a sample $x[t]$ of the signal analyzed as a linear combination of previous samples and a prediction error [18]:

$$x[t] = \sum_{n=1}^{N} \alpha_n x[t-n] + r[t], \tag{1}$$

where $\alpha = [\alpha_n]_{n=1}^N$ are the prediction coefficients and $r[t]$ is the prediction error, also called prediction residual. The compact representation provided by the LP model has a clear impact on its employment in compression where the theorem of predictive quantization [19] states that the mean squared reproduction error in predictive encoding is equal to the mean squared quantization error when the residual signal is presented to the quantizer. This makes the quantization approach optimal when the coefficients of the predictor are obtained by minimizing the mean square error or 2-norm of the prediction error. The general formulation is:

$$\underset{\alpha}{\text{minimize}} \, \|x - X\alpha\|_p^p \tag{2}$$

where $\|\cdot\|_p$ is the p-norm defined as $\|x\|_p = (\sum_{m=1}^{M} |x_m|^p)^{\frac{1}{p}}$ for $p \geq 1$ and

$$[x|X] = \begin{bmatrix} x[T_1] & \cdots & x[T_1 - N] \\ \vdots & \ddots & \vdots \\ x[T_2] & \cdots & x[T_2 - N] \end{bmatrix}. \tag{3}$$

Assuming $x[t] = 0$ for $t < 1$ and $t > T$ for some T, the indexes T_1 and T_2 can be chosen in various ways which lead to different types of solutions with different properties [20]. In the above mentioned case, where $p = 2$, a closed-form solution can also be obtained $\alpha = (X^T X)^{-1} X^T x$.

This model, however, fails to provide a general framework when the signal redundancies are present at different time intervals, like in the case of speech and audio, where a given segment has short-term and long-term redundancies that cannot be represented by a simple linear prediction model with a limited number of taps. Traditional approaches tend to represent short-term redundancies using traditional LP and represent long-term redundancies by applying a so-called long-term predictor (LTP) with a very limited number of taps clustered around the *pitch period* of the speech or audio signal [21]. Since the combination of these two filters is a high-order sparse predictor, a more effective way to model these types of signal was shown by increasing the order of the predictor and apply a sparsity criterion on its coefficients [22]:

$$\underset{\alpha}{\text{minimize}} \, \|x - X\alpha\|_p^p + \gamma\|\alpha\|_1, \tag{4}$$

The work of T. L. Jensen is supported by The Danish Council for Independent Research, Technology and Production Sciences, grant number 4005-00122. The work of T. van Waterschoot is supported by the KU Leuven Impulse Fund, grant number IMP/14/037.

where the 1-norm is the well known convex relaxation of the so-called 0-norm [23]. By applying this method it is shown to be possible to identify the true structure of the predictor, even for polyphonic audio signals [9]. In addition, by applying the 1-norm also on the residual, both modeling and coding advantages can be achieved [2]. Our SLP formulation then becomes:

$$\underset{\alpha}{\text{minimize}}\, f(\alpha) = \|x - X\alpha\|_1 + \gamma\|\alpha\|_1 \tag{5}$$

with a solution $\alpha^\star$. This problem does not have a closed-form solution, for which reason iterative methods have been developed.

Solving (5) using iterative methods is further complicated since neither one of the terms in the objective function $f(\alpha)$ are differentiable and hence fast proximal gradient methods cannot be employed [24, 25]. To solve more general convex optimization, a significant amount of work for real-time optimization has lately been focused on code generation using interior-point (IP) methods [10, 26, 27]. The most significant matrix structure these methods exploit is sparsity, i.e., for example at code-generation exploiting $[A\,0][x^T\,y^T]^T = Ax$ such that we avoid computing $0 \cdot y$. On the other hand many applications in signal processing, including the problem (5), involve around (dense) Toeplitz structured matrices, or in general, low displacement rank matrix structures.

For IP methods the main bulk of work is to solve a linear system of equations in each iteration where additional scaling matrices are introduced and in particular diagonal matrices for linear programming. Such diagonal matrices often destroys the possibility of faster direct methods for solving a linear system (see, e.g., the discussion in [28]). Specifically, IP methods for the SLP problem (5) would have cubic per iteration time complexity [29,30] and quadratic space complexity. In particular, the introduction of the diagonal scaling matrix for the SLP problem prohibits the exploration of Toeplitz structure since such matrices do not have a low displacement rank.

Fast (and superfast methods) for, e.g., solving linear Toeplitz systems are known with quadratic or linearithmic time complexity and linear space complexity, e.g., [31,32]. The last couple of years a new class of algorithms based on the alternating methods of multipliers (ADMM) have reemerged (see, e.g., the overview work [33,34]), with widespread use in signal processing [35–37]. ADMM algorithms require more iterations than IP methods but allow the use of fast(er) methods for Toeplitz systems.

This paper deals with understanding the trade-offs occurring in choosing a proper and fixed number of iterations for the ADMM algorithm and extends the analysis and algorithms presented in [30, 38]. We note that we will apply an ADMM algorithm in its straightforward form but several variants and extensions may be useful for solving the sparse linear prediction problem efficiently and may be considered for further investigations. On particular choice is a preconditioned ADMM (Chambolle-Pock) where the algorithm does not involve solving a linear system of equation [51].

2. AN ADMM ALGORITHM FOR THE SLP PROBLEM

There are numerous approaches for formulating an ADMM algorithm for the problem (5), but, in this paper, we follow the overview work in [33, 34]. The problem (5) can be reformulated as the least absolute deviation problem

$$\text{minimize}\, \|Az - b\|_1, \tag{6}$$

with

$$A = \begin{bmatrix}\gamma I \\ -X\end{bmatrix},\; b = \begin{bmatrix}0 \\ -x\end{bmatrix},\; z = \alpha\,. \tag{7}$$

The ADMM algorithm in the scaled form for solving the least absolute deviation problem is

$$z^{(k+1)} = (A^TA)^{-1}A^T(y^{(k)} + b - u^{(k)}) \tag{8}$$

$$y^{(k+1)} = S_{1/\rho}\left(Az^{(k+1)} - b + u^{(k)}\right) \tag{9}$$

$$u^{(k+1)} = u^{(k)} + Az^{(k+1)} - y^{(k+1)} - b \tag{10}$$

where ρ is a scaling parameter and S is the soft-thresholding function given by

$$(S_t(v))_i = \text{sgn}(v_i)\max(|v_i| - t, 0), \tag{11}$$

and $(\cdot)_i$ denotes the ith element. The explicit ADMM form for the SLP problem can now be formulated. We have

$$(A^TA)^{-1}A^T = (X^TX + \gamma^2 I)^{-1}\begin{bmatrix}\gamma I & -X^T\end{bmatrix}, \tag{12}$$

and

$$Az^{(k)} - b = \begin{bmatrix}\gamma\alpha^{(k)} \\ -X\alpha^{(k)} + x\end{bmatrix} = \begin{bmatrix}\gamma\alpha^{(k)} \\ e^{(k)}\end{bmatrix}, \tag{13}$$

where $e^{(k)} = x - X\alpha^{(k)}$ is defined as the error signal at the kth iteration and the algorithm is then

$$\alpha^{(k+1)} = (X^TX + \gamma^2 I)^{-1}\begin{bmatrix}\gamma I & -X^T\end{bmatrix}(y^{(k)} + \begin{bmatrix}0 \\ -x\end{bmatrix} - u^{(k)}) \tag{14}$$

$$e^{(k+1)} = x - X\alpha^{(k+1)} \tag{15}$$

$$y^{(k+1)} = S_{1/\rho}\left(\begin{bmatrix}\gamma\alpha^{(k+1)} \\ e^{(k+1)}\end{bmatrix} + u^{(k)}\right) \tag{16}$$

$$u^{(k+1)} = u^{(k)} + \begin{bmatrix}\gamma\alpha^{(k+1)} \\ e^{(k+1)}\end{bmatrix} - y^{(k+1)}\,. \tag{17}$$

This algorithm has an interesting interpretation. In the first line we have the term

$$(X^TX + \gamma^2 I)^{-1}\begin{bmatrix}\gamma I & -X\end{bmatrix}\begin{bmatrix}0 \\ -x\end{bmatrix} = (X^TX + \gamma^2 I)^{-1}X^Tx \triangleq \alpha_{\gamma,2}\,. \tag{18}$$

Here $\alpha_{\gamma,2}$ is the "standard" regularized 2-norm linear prediction solution. With this definition we may reformulate the steps in the ADMM algorithm (14)–(17) to the more instructional form

$$\alpha^{(k+1)} = \alpha_{\gamma,2} + (X^TX + \gamma^2 I)^{-1}\begin{bmatrix}\gamma I & -X^T\end{bmatrix}(y^{(k)} - u^{(k)}) \tag{19}$$

$$e^{(k+1)} = x - X\alpha^{(k+1)} \tag{20}$$

$$y^{(k+1)} = S_{1/\rho}\left(\begin{bmatrix}\gamma\alpha^{(k+1)} \\ e^{(k+1)}\end{bmatrix} + u^{(k)}\right) \tag{21}$$

$$u^{(k+1)} = u^{(k)} + \begin{bmatrix}\gamma\alpha^{(k+1)} \\ e^{(k+1)}\end{bmatrix} - y^{(k+1)} \tag{22}$$

such that with $y^{(0)} = u^{(0)} = 0$, this ADMM algorithm for the SLP problem may be interpreted as an iterative sparsification of the regularized 2-norm linear prediction solution $\alpha_{\gamma,2}$.

At this point it is worth mentioning that following (18), we are solving the least-squares problem

$$\underset{\alpha}{\text{minimize}}\left\|\begin{bmatrix}\gamma I \\ -X\end{bmatrix}\alpha - \left(y^{(k)} + \begin{bmatrix}0 \\ -x\end{bmatrix} - u^{(k)}\right)\right\|_2, \tag{23}$$

in each iteration. This should be compared with IP methods for the least absolute deviation problems, where we need to solve a number of *reweighted* least-squares problems, see e.g. [39, §11.8.2]. However, the reweighting destroys the Toeplitz structure and we need to resort to more general linear solvers like Cholesky factorization with time-complexity $\mathcal{O}(N^3)$, as in [30].

2.1. Solving a symmetric positive Toeplitz linear system

From the ADMM algorithm outlined in (14)–(17) it is clear that special attention should be paid to solving the linear system

$$(X^T X+\gamma^2 I)\alpha^{(k+1)} = \begin{bmatrix}\gamma I & -X^T\end{bmatrix}(y^{(k)}+\begin{bmatrix}0\\-x\end{bmatrix}-u^{(k)}), \quad (24)$$

in order to obtain an efficient algorithm. Firstly, the behavior of $y^{(k)} - u^{(k)}$ from iteration to iteration is currently unclear, so we consider this as a general right-hand side problem

$$(X^T X + \gamma^2 I)\alpha^{(k+1)} = v^{(k)}\,. \quad (25)$$

If X is formed using the autocorrelation method with $T_1 = 1$ and $T_2 = T + N$ then

$$X^T X + \gamma^2 I = \{t_{|i-j|}\}_{i,j=1}^{N} \quad (26)$$

is a symmetric, positive definite Toeplitz matrix with $t_0 = r_0 + \gamma^2$ and $t_i = r_i, i = 1, \ldots, N-1$ where $R = X^T X = \{r_{|i-j|}\}_{i,j=1}^{N}$ is the Toeplitz autocorrelation matrix.

A standard algorithm for solving symmetric positive definite linear systems is the Levinson algorithm [40] with time-complexity $\mathcal{O}(N^2)$ [41]. Another approach is to use the Gohberg-Semencul representation of the inverse of a Toeplitz matrix and use the algorithm presented in [42]. We denote this as the GS approach. The advantage of the GS approach is that we initially run the Szegö recursion with time-complexity $\mathcal{O}(N^2)$ and then the system (25) can be solved with time-complexity $\mathcal{O}(N \log N)$ via a number of FFTs/IFFTs for each iteration[1] $k = 1, \ldots, K$ [42]. The time-complexity of the initialization may be further reduced using superfast solvers in time $\mathcal{O}(N \log^2 N)$ but the constant is larger such that this approach would require more operations unless $N \geq 256$, see [31, 32]. Typical applications of SLP have dimensions $N \leq 256$, and the initial Szegö recursion in the GS approach is a much simpler algorithm than the initial step in superfast algorithms and do not exploit this possibility further in this paper.

2.2. Accuracy requirement

While in the previous section we have discussed the need for *solving* the problem (5), most iterative methods will only solve (5) *approximately* so an important question is how accurate should the solution be for the task given. First of all, the LP speech model only approximates the human speech production system and the model is not noise free [43]. Moreover, the convex optimization framework uses the 1-norm because of its feasibility but more effective solutions for our needs might be possible by, e.g., reducing the mismatch between 1-norm and the 0-norm. In other words, we are not interested in a solution of (5) but we are seeking a good enough $\hat{\alpha} \approx \alpha^\star$ that captures the essence of our endeavor. For example, using (5) for speech coding purposes might require different accuracy than using it for modeling the speech glottal flow [44]. So, since we have used several approximations to formulate (5) it is likely that the performance as a function of the accuracy $f(\alpha^{(k)}) - f(\alpha^\star) \leq \epsilon$ shows a saturation effect as a function of k (like in [11]). Specifically, for a certain ϵ or $k = K$ we may see no or little improvement in our performance metrics.

[1] Precisely 3 FFTs and 3 IFFTs per iteration and 2 FFTs in connection to the initial Szegö recursion.

3. NUMERICAL SIMULATIONS

We will now proceed to investigate the prediction gain performance as a function of the number of iterations of the presented ADMM algorithm and the associated computational cost assessed via timings.

3.1. Prediction Gain

Similarly to what was done in [38], we processed only the vowel and semivowel phones [45] from the TIMIT database (sampled at 16 kHz), belonging to 3696 sentences from 462 speakers. We chose the ones of duration of at least 640 samples (40 ms) for a total of about 40,000 voiced speech frames. We extend the analysis in [38] by investigating the prediction gain from the ADMM solution with a different number of iterations and compare with the IP solution obtained through the CVX+SeDuMi interface and solver. In both formulation of the SLP problem (5), we chose $\gamma = 0.12$ obtained through a modified version of the L-curve [46] by using all except 50 frames picked randomly that will be used as a test set. This value provided a good trade-off between the sparsity of the residual and sparsity of the predictor. We chose $N = 250$ by which we were able to cover the pitch lag range $T_\mathrm{p} \in [34, 231]$ as done in commercial speech codecs like the wideband version of the Adaptive Multi-Rate codec (AMR-WB [47]). We tested the ADMM algorithm for a different number of iterations $k \in \{5, 10, 15, 20, 25, 30\}$, the results for the 50 frames part of the test set are shown in Figure 1.

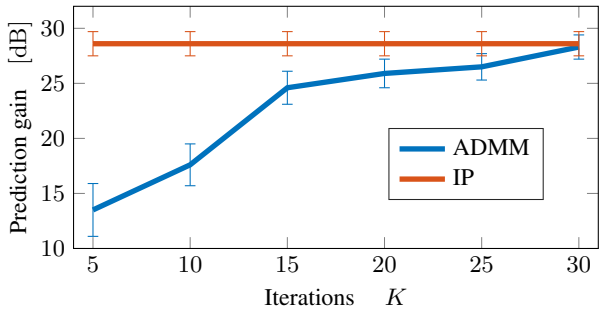

Fig. 1. Average prediction gains for a fixed number of iterations for the ADMM solution. A 95% confidence interval is shown. The IP solutions as returned by CVX+SeDuMi is independent of the fixed number of iterations K but shown for the ease of comparison. Notice the saturation effect.

We can see that already above 20 iterations, we achieved prediction gains with the ADMM in the same order as the one obtained with the IP. At 30 iterations, the mean value of the IP solution fell within the 95% confidence interval of the ADMM solution, proving that the two algorithms exhibit statistically the same performance. Notice the saturation effect. In Figure 2, we show an example of the behavior of the different methods for the vowel /a/ extracted from our experimental dataset. The result of the solution of 2-norm unconstrained LP was shown for comparison. While theoretically achieving the highest prediction gain, it can be seen that the solution

was quite similar to the ones obtained with SLP with a significant lower number of non-zero coefficients. In particular, the cardinality of the solutions obtained through ADMM and IP were truncated to the 21 largest values to compare to traditional two step LP and LTP while the 2-norm unconstrained LP retains all 250 nonzero coefficients [38].

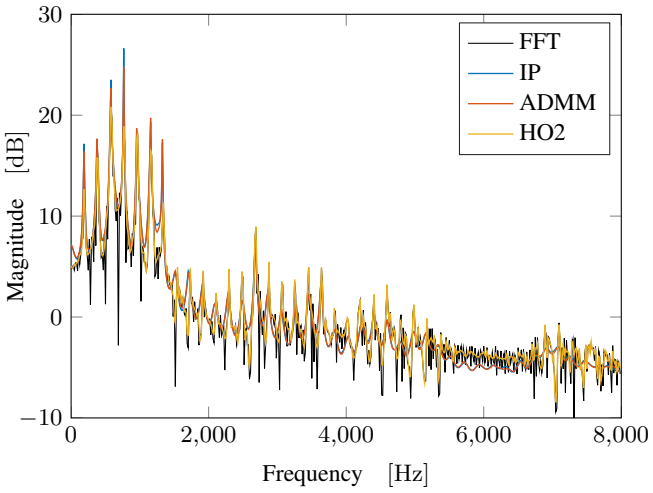

Fig. 2. Magnitude of the frequency response of the different methods proposed. A 320 samples segment of the voiced speech (vowel /a/ uttered by a female speaker) is used for the analysis. The ADMM solution is obtained with 30 iterations, CVX+SeDuMi is used to obtain the IP solution, and HO2 is the classical LP solution with no constraint. The FFT of the segment is shown for comparison.

3.2. Timing

We empirically assessed the computational complexity of the proposed ADMM algorithm by timing a C++ implementation using the FFTW3 library [48] and Intel Math Kernel Library (MKL) [49] for BLAS level 1 routines and element-wise operations $(a \leftarrow b \odot c)$[2]. The simulations were executed on an Intel(R) Core(TM) i7-5600U CPU 2.6 GHz with Ubuntu Linux kernel 3.19.0-43-generic, MKL 11.3 Beta. The algorithms implemented in C++ were compiled using gcc-4.8 and the `-Os -march=native` optimization option. The timings were obtained by executing the algorithms 10^i times using the smallest $i \in \{0, 1, 2, \cdots\}$ such that the total execution time was $\tau_0 \geq 0.2\,\mathrm{s}$. For this i, three repetitions of 10^i executions were then run and timed, producing the three repetition times τ_1, τ_2 and τ_3. The execution time of the algorithm is then estimated as $t = \min(\tau_1, \tau_2, \tau_3)/10^i$ (same as Pythons timeit module [50]). The time t_k was calculated for $k \in \{5, 10, 15, 20, 25, 30\}$ and the results almost followed a straight line so we use linear regression and report the coefficients of determination C^2.

Using the Levinson algorithm for solving the linear system we obtain

$$t_k \approx 7 \cdot 10^{-6} + 159 \cdot 10^{-6} k \quad [\mathrm{s}], \quad C^2 = 0.999\,. \tag{27}$$

Using the GS algorithm we obtain

$$t_k \approx 62 \cdot 10^{-6} + 55 \cdot 10^{-6} k \quad [\mathrm{s}], \quad C^2 = 0.999\,. \tag{28}$$

[2]C++ and Matlab implementations available at `http://kom.aau.dk/~tlj/`

From this is clear that the Levinson approach has a smaller initialization cost but higher per iteration cost compared to the GS approach. For these implementations, the GS approach was a faster choice than Levinson for $k > 1$. For comparison, a hand-tailored IP method [30] gave

$$t_k \approx 1206 \cdot 10^{-6} + 2033 \cdot 10^{-6} k \quad [\mathrm{s}], \quad C^2 = 0.999\,. \tag{29}$$

As argued previously, one IP method iteration is much more expensive. But more importantly, it is possible to run $k \approx 36$ iterations of the ADMM algorithm using the GS approach for just a single IP method iteration (including the Szegö recursion in the initialization of the GS approach). As mentioned previously in this section, $k = K = 30$ iterations is enough to obtain (in statistical terms) the same performance.

4. CONCLUSIONS

A fast algorithm for the sparse linear prediction problem was presented. The method is based on an alternating direction method of multipliers that has the important characteristic that the matrix structures which also occur in classical least-squares linear prediction is preserved. Experimental analysis clearly shows the reduced cost per iteration of the methods used versus common interior-point methods while achieving the same prediction gain.

5. REFERENCES

[1] D. Giacobello, M. G. Christensen, J. Dahl, S. H. Jensen, and M. Moonen, "Sparse linear predictors for speech processing," in *Proc. INTERSPEECH*, pp. 1353–1356, 2008.

[2] D. Giacobello, M. G. Christensen, M. N. Murthi, S. H. Jensen, and M. Moonen, "Sparse linear prediction and its applications to speech processing," *IEEE Trans. Audio, Speech, Language Process.*, vol. 20, no. 5, pp. 1644–1657, 2012.

[3] J. Makhoul, "Linear prediction: A tutorial review," *Proc. IEEE*, vol. 63, no. 4, pp. 561–580, 1975.

[4] T. Bäckström, *Linear predictive modeling of speech: constraints and line spectrum pair decomposition*, Helsinki University of Technology, 2004.

[5] M. Grant and S. Boyd, "CVX: Matlab software for disciplined convex programming, v. 1.21," `http://cvxr.com/cvx/`, 2011.

[6] J. F. Sturm, "Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones," *Optim. Methods Software*, vol. 11-12, pp. 625–653, 1999.

[7] E. Candes and J. Romberg, "ℓ_1-MAGIC: Recovery of Sparse Signals," `http://users.ece.gatech.edu/justin/l1magic/`, 2005.

[8] D. Giacobello, M. G. Christensen, M. N. Murthi, S. H. Jensen, and M. Moonen, "Speech coding based on sparse linear prediction," in *Proc. EUSIPCO*, pp. 2524–2528, 2009.

[9] D. Giacobello, T. van Waterschoot, M. G. Christensen, S. H. Jensen, and M. Moonen, "High-order sparse linear predictors for audio processing," in *Proc. EUSIPCO*, pp. 234–238, 2010.

[10] J. Mattingley and S. Boyd, "Real-time convex optimization in signal processing," *IEEE Signal Process. Magazine*, vol. 27, no. 3, pp. 50–61, 2010.

[11] B. Defraene, T. van Waterschoot, H. J. Ferreau, M. Diehl, and M. Moonen, "Real-time perception-based clipping of audio signals using convex optimization," *IEEE Trans. Audio, Speech, Language Process.*, vol. 20, no. 10, pp. 2657–2671, 2012.

[12] D. Angelosante, "Sparse regressions for joint segmentation and linear prediction," in *Proc. ICASSP*, pp. 335–339, 2014.

[13] D. Wei, C.K. Sestok, and A.V. Oppenheim, "Sparse filter design under a quadratic constraint: Low-complexity algorithms," *IEEE Trans. Signal Process.*, vol. 61, no. 4, pp. 857–870, 2013.

[14] A. Jukic, T. van Waterschoot, T. Gerkmann, and S. Doclo, "Multi-channel linear prediction-based speech dereverberation with sparse priors," *IEEE/ACM Trans. Audio, Speech, Language Process.*, vol. 23, no. 9, pp. 1509–1520, 2015.

[15] J. Koloda, J. Østergaard, S.H. Jensen, V. Sanchez, and A.M. Peinado, "Sequential error concealment for video/images by sparse linear prediction," *IEEE Trans. Multimedia*, vol. 15, no. 4, pp. 957–969, 2013.

[16] N. Bochud, A.M. Gomez, G. Rus, and A.M. Peinado, "Sparse signal model for ultrasonic nondestructive evaluation of cfrp composite plates," in *Proc. ICASSP*, pp. 2844–2847, 2013.

[17] I. Erer, K. Sarikaya, and H. Bozkurt, "Enhanced radar imaging via sparsity regularized 2d linear prediction," in *Proc. EUSIPCO*, pp. 1751–1755, 2014.

[18] P. P. Vaidyanathan, "The theory of linear prediction," *Synthesis lectures on signal processing*, Morgan & Claypool, 2007.

[19] A. Gersho and R. M. Gray, *Vector quantization and signal compression*, Kluwer Academic Press, 1991.

[20] P. Stoica and R. L. Moses, *Spectral analysis of signals*, Pearson/Prentice Hall, 2005.

[21] Q. Yasheng and P. Kabal, "Pseudo-three-tap pitch prediction filters," in *Proc. ICASSP*, vol. 2, pp. 523–526, 1993.

[22] D. Giacobello, M. G. Christensen, J. Dahl, S. H. Jensen, and M. Moonen, "Joint estimation of short-term and long-term predictors in speech coders," in *Proc. ICASSP*, pp. 4109–4112, 2009.

[23] D. L. Donoho, "For most large underdetermined systems of linear equations the minimal 1-norm solution is also the sparsest solution," *Comm. Pure Appl. Math.*, vol. 59, no. 6, pp. 797–829, 2006.

[24] Yu. Nesterov, "Gradient methods for minimizing composite objective function," Université Catholique de Louvain, Center for Operations Research and Econometrics (CORE), CORE Discussion Papers, no. 2007076, 2007.

[25] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imag. Sci.*, vol. 2, pp. 183–202, 2009.

[26] A. Domahidi, E. Chu, and S. Boyd, "ECOS: An SOCP solver for embedded systems," in *Proc. EUSIPCO*, pp. 3071–3076, 2013.

[27] D. Dueri, J. Zhang, and B. Açikmeşe, "Automated custom code generation for embedded, real-time second order cone programming," in *Proc. Int. Fed. Autom. Control*, pp. 1605–1612, 2015.

[28] D. O'Connor and L. Vandenberghe, "Primal-dual decomposition by operator splitting and applications to image deblurring," *SIAM J. Imag. Sci.*, vol. 7, pp. 1724–1754, 2014.

[29] G. Alipoor and M. H. Savoji, "Wide-band speech coding based on bandwidth extension and sparse linear prediction," in *Int. Conf. Telecom. Signal Process. (TSP)*, pp. 454–459, 2012.

[30] T. L. Jensen, D. Giacobello, M. G. Christensen, S. H. Jensen, and M. Moonen, "Real-time implementations of sparse linear prediction for speech processing," in *Proc. ICASSP*, pp. 8184–8188, 2013.

[31] R. R. Bitmead and B. D.O. Anderson, "Asymptotically fast solution of Toeplitz and related systems of linear equations," *Linear Algebra Appl.*, vol. 34, pp. 103–116, 1980.

[32] G. S. Ammar and W. B. Gragg, "Superfast solution of real positive definite Toeplitz systems," *SIAM J. Matrix Analysis. Appl.*, vol. 9, no. 1, pp. 61–76, 1988.

[33] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundation and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[34] D. P. Bertsekas, *Convex Optimization Algorithms*, Athena Scientific, 2015.

[35] W. Yin, S. Osher, D. Goldfarb, and J. Darbon, "Bregman iterative algorithms for ℓ_1-minimization with application to compressed sensing," *SIAM J. Imag. Sci.*, vol. 1, no. 1, pp. 143–168, 2008.

[36] M. V. Afonso, J. M. Bioucas-Dias, and M. A. T. Figueiredo, "Fast image recovery using variable splitting and constrained optimization," *IEEE Trans. Image Process.*, vol. 19, no. 9, pp. 2345–2356, 2010.

[37] J. Yang and Y. Zhang, "Alternating direction algorithms for ℓ_1-problems in compressive sensing," *SIAM J. Sci. Comput.*, vol. 33, no. 1, pp. 250–278, 2011.

[38] T. L. Jensen, D. Giacobello, T. van Waterschoot, and M. G. Christensen, "Fast algorithms for high-order sparse linear prediction with applications to speech processing," *Speech Comm.*, vol. 76, pp. 143–156, 2016.

[39] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.

[40] N. Levinson, "The Weiner RMS error criterion in filter design and prediction," *J. Math. Phys.*, vol. 25, pp. 261–278, 1947.

[41] G. H. Golub and C. F. van Loan, *Matrix Computations*, The Johns Hopkins University Press, 2013.

[42] J. R. Jain, "An efficient algorithm for a large Toeplitz set of linear equations," *IEEE Trans. Acoustic, Speech, Signal Proc.*, vol. 27, no. 6, pp. 612–615, 1979.

[43] J. H. L. Hansen, J. G. Proakis, and J. R. Deller Jr, *Discrete-time processing of speech signals*, Prentice-Hall, 1987.

[44] V. Khanagha and K. Daoudi, "Efficient GCI detection for efficient sparse linear prediction," in *Advances in Nonlinear Speech Processing*, Lecture Notes in Computer Science, vol. 7911, pp. 17–24, 2013.

[45] A. K. Halberstadt and J. R. Glass, "Heterogeneous acoustic measurements for phonetic classification," in *EUROSPEECH*, 1997.

[46] P. C. Hansen and D. P. O'Leary, "The use of the L-curve in the regularization of discrete ill-posed problems," *SIAM J. Sci. Comput.*, vol. 14, no. 6, pp. 1487–1503, 1993.

[47] B. Bessette, R. Salami, R. Lefebvre, M. Jelinek, J. Rotola-Pukkila, J. Vainio, H. Mikkola, and K. Jarvinen, "The adaptive multirate wideband speech codec (AMR-WB)," *IEEE Trans. Speech Audio Process.*, vol. 10, no. 8, pp. 620–636, 2002.

[48] M. Frigo and S. G. Johnson, "The design and implementation of FFTW3," in *Proc. IEEE*, vol. 93, pp. 216–231, 2005.

[49] Intel, *Intel(R) Math Kernel Library*, `https://software.intel.com/en-us/intel-mkl`.

[50] G. van Rossum and the Python development team, *The Python Library Reference*, Python Software Foundation, v. 2.7.11, `https://docs.python.org/2/index.html`, 2016.

[51] A. Chambolle and T. Pock, "A first-order primal-dual algorithm for convex problems with applications to imaging," *J. Math. Imag. Vis.*, vol. 40, no. 1, pp. 120–145, 2011.