MAKING A CASE FOR ITERATIVE LINEAR EQUATION SOLVERS IN DSP EDUCATION

John Håkon Husøy

Department of Electrical and Computer Engineering, Stavanger University College,
Post Box 8002, N-4068 Stavanger, Norway. (email: John.H.Husoy@tn.his.no)

ABSTRACT

Iterative techniques for solving (large) sets of linear equations is an active research field in applied mathematics. Such techniques constitute an insignificant part, if any part at all, of the background of electrical and computer engineers. We argue that some exposure to such techniques is beneficial in the study of digital signal processing. While important in their own right, we show that the knowledge of such techniques facilitates a complementary understanding/interpretation of digital filters as equations solvers. Such connections between seemingly unrelated fields enhance the learning experience. Perhaps more important, we also show that elementary iterative equation solvers can be used as a vehicle for an introduction to adaptive filters at an elementary level.

1. INTRODUCTION

Iterative techniques for solving (large) sets of linear equations is an active research field in applied mathematics. Unfortunately iterative solvers for sets of linear equations plays no, or at best a very minor, role in the standard electrical and computer engineering curriculum. We believe that a short exposure to such techniques, possibly incorporated into a first or intermediate course on digital signal processing (DSP), can be used as a basis for providing a complementary understanding and/or interpretation of simple time domain digital filters. We give an example where it is shown that a simple recursive filter can be interpreted as an iterative solver of an underlying set of linear equations. Such exposure to connections between seemingly unrelated concepts enhance the learning experience. Subsequently, we show how iterative equation solvers are well suited as a tools facilitating a simple introduction to adaptive filtering.

We have organized the paper as follows: The next section gives a brief background on iterative algorithms for solving sets of linear equations. This is followed by an example showing the interpretation of a recursive filter as an equation solver. Finally, we present an alternative way of introducing simple adaptive filters.

This work was performed while on sabbatical leave with the School of Information Technology and Engineering, University of Ottawa, Ontario, Canada. Support from the Center for Information and Communication Technology (SIKT) at Stavanger University College and the Norwegian Research Council is gratefully acknowledged.

2. BACKGROUND ON ITERATIVE LINEAR EQUATION SOLVERS

The *Gauss-Seidel* iterative technique for solving sets of linear equations, $\mathbf{A}\underline{x} = \underline{b}$, with $\mathbf{A}$ assumed to have dimension $M \times M$, can be explained as follows [1]: The iterative scheme proceeds cyclically through the elements of the vector of unknowns, x_j, for $j = 1, 2 \ldots M$, finding one single x_j at time. In doing this, it is *assumed* that all the x_i, $i \neq j$, are known from previous computations. Given this, we can express x_j as

$$x_j = \frac{1}{a_{j,j}}[b_j - \sum_{i=1,(i \neq j)}^{M} a_{j,i}x_i], \quad (1)$$

where $a_{i,j}$ is element (i, j) of $\mathbf{A}$ and b_i is element i of $\underline{b}$. Cycling through all the indices $j = 1, 2 \ldots, M$ a sufficient number of times will, if matrix $\mathbf{A}$ satisfies certain conditions [1], produce a sufficiently accurate solution to $\mathbf{A}\underline{x} = \underline{b}$. In numerical linear algebra one cycle through $j = 1, 2 \ldots, M$ is referred to as one iteration. Thus, when computing x_j according to Equation 1, the x_i's with $i < j$ are available from previous steps of the current iteration, whereas the x_i's with $i > j$ are available from the *previous iteration*. Emphasizing this, and identifying the k'th iterate of x_i as $x_i^{(k)}$, we may write a Gauss-Seidel step as

$$x_j^{(k)} = \frac{1}{a_{j,j}}[b_j - \sum_{i=1}^{j-1} a_{j,i}x_i^{(k)} - \sum_{i=j+1}^{M} a_{j,i}x_i^{(k-1)}]. \quad (2)$$

Writing this explicitly for all x_j's we have:

$$\begin{aligned}
x_1^{(k)} &= \frac{1}{a_{1,1}}[b_1 - \sum_{i=2}^{M} a_{1,i}x_i^{(k-1)}] \\
x_2^{(k)} &= \frac{1}{a_{2,2}}[b_2 - a_{2,1}x_1^{(k)} - \sum_{i=3}^{M} a_{2,i}x_i^{(k-1)}] \\
&\vdots \\
x_j^{(k)} &= \frac{1}{a_{j,j}}[b_j - \sum_{i=1}^{j-1} a_{j,i}x_i^{(k)} - \sum_{i=j+1}^{M} a_{j,i}x_i^{(k-1)}] \\
&\vdots \\
x_M^{(k)} &= \frac{1}{a_{M,M}}[b_M - \sum_{i=1}^{M-1} a_{M,i}x_i^{(k)}].
\end{aligned} \tag{3}$$

Collecting all the k'th iterates on the left hand side and the $k-1$'th iterates on the right hand side in appropriately defined vectors, defining

$$\mathbf{L} = \begin{bmatrix} a_{1,1} & 0 & \cdots & & 0 \\ a_{2,1} & a_{2,2} & 0 & \cdots & 0 \\ \vdots & & \ddots & \ddots & \\ \vdots & & \cdots & \ddots & 0 \\ a_{M,1} & a_{M,2} & \cdots & & a_{M,M} \end{bmatrix}, \tag{4}$$

and

$$\mathbf{U} = -\begin{bmatrix} 0 & a_{1,2} & \cdots & & a_{1,M} \\ 0 & 0 & a_{2,3} & \cdots & a_{2,M} \\ \vdots & & \vdots & \ddots & \\ 0 & \cdots & 0 & \cdots & a_{M-1,M} \\ 0 & 0 & \cdots & & 0 \end{bmatrix}, \tag{5}$$

a little reflection reveals that Equation 3 can be stated as

$$\mathbf{L}\underline{x}^{(k)} = \mathbf{U}\underline{x}^{(k-1)} + \underline{b}. \tag{6}$$

We note that $\mathbf{U}$ is strictly upper triangular, and that $\mathbf{L}$ can be written as $\mathbf{L} = \mathbf{D} + \tilde{\mathbf{L}}$ with $\mathbf{D}$ diagonal and $\tilde{\mathbf{L}}$ strictly lower triangular. In other words, we see that a *splitting* of $\mathbf{A}$ given by $\mathbf{A} = \mathbf{D} + \tilde{\mathbf{L}} - \mathbf{U}$ defines the matrices involved in the iteration of Equation 6. Defining $\mathbf{T} = (\mathbf{D} + \tilde{\mathbf{L}})^{-1}\mathbf{U}$ and $\underline{c} = (\mathbf{D} + \tilde{\mathbf{L}})^{-1}\underline{b}$ we can write the iterative scheme of Equation 6 as

$$\underline{x}^{(k)} = \mathbf{T}\underline{x}^{(k-1)} + \underline{c}. \tag{7}$$

This equation describes the class of *stationary* iterative linear equation solvers of which the Gauss-Seidel method is one example. Other selections of $\mathbf{T}$ and $\underline{c}$ based on different splittings of $\mathbf{A}$, give rise to other stationary iterative equation solvers. The simplest, and one of the older of these methods is the *Richardson iteration* [2] which is based on the splitting $\mathbf{A} = \frac{1}{\mu}\mathbf{I} - (\frac{1}{\mu}\mathbf{I} - \mathbf{A})$. In its simplest form the Richardson iteration is often stated as [3]:

$$\underline{x}^{(k)} = (\mathbf{I} - \mu\mathbf{A})\underline{x}^{(k-1)} + \mu\underline{b}. \tag{8}$$

A substantial amount of literature on iterative linear equation solvers, as is evidenced for example by the extensive list of references in [3], is available. Here we just quote an important result (page 104 of [1]): Assume that the set of linear equations giving rise to an iteration of the type given by Equation 7 has a unique solution. If the iteration converges, it converges to the solution of $\mathbf{A}\underline{x} = \underline{b}$. Such convergence is guaranteed if all the eigenvalues of $\mathbf{T}$ are less than unity.

3. EXAMPLE 1: DIGITAL FILTERS AS ITERATIVE EQUATION SOLVERS

In this section we point out that a simple recursive digital filter exemplified by

$$y(n) = -a_1 y(n-1) - a_2 y(n-2) + x(n), \tag{9}$$

with a *periodic* input $x(n)$ can be interpreted as Gauss-Seidel solver for an underlying circulant linear equation system. The example can easily be generalized.

In matrix/vector terms, Equation 9, for $n, n-1$, and $n-2$, can be written as

$$\begin{bmatrix} 1 & a_1 & a_2 & 0 & 0 & 0 \\ 0 & 1 & a_1 & a_2 & 0 & 0 \\ 0 & 0 & 1 & a_1 & a_2 & 0 \end{bmatrix} \begin{bmatrix} y(n) \\ y(n-1) \\ y(n-2) \\ y(n-3) \\ y(n-4) \\ y(n-5) \end{bmatrix} = \begin{bmatrix} x(n) \\ x(n-1) \\ x(n-2) \end{bmatrix}.$$

Defining $\underline{y}(n-k) = [y(n-k), y(n-k-1), y(n-k-2)]^T$, where T denotes vector transpose, and $\underline{x}(n-k)$ similarly, we can write

$$\begin{bmatrix} 1 & a_1 & a_2 \\ 0 & 1 & a_1 \\ 0 & 0 & 1 \end{bmatrix} \underline{y}(n) = -\begin{bmatrix} 0 & 0 & 0 \\ a_2 & 0 & 0 \\ a_1 & a_2 & 0 \end{bmatrix} \underline{y}(n-3) + \underline{x}(n).$$

The immediate interpretation of this equation is as a block implementation of the filter of Equation 9. Assuming $x(n)$ is periodic with period 3 and viewing $\underline{y}(n)$ as a vector that is computed from a previous value of the same vector, $\underline{y}(n-3)$, it would make sense to substitute the used notation with $\underline{y}^{(k)}$ and $\underline{y}^{(k-1)}$, respectively. Using this we have

$$\begin{bmatrix} 1 & a_1 & a_2 \\ 0 & 1 & a_1 \\ 0 & 0 & 1 \end{bmatrix} \underline{y}^{(k)} = -\begin{bmatrix} 0 & 0 & 0 \\ a_2 & 0 & 0 \\ a_1 & a_2 & 0 \end{bmatrix} \underline{y}^{(k-1)} + \underline{x}.$$

Comparing this with Equation 6, and realizing that the matrices

$$\begin{bmatrix} 1 & a_1 & a_2 \\ 0 & 1 & a_1 \\ 0 & 0 & 1 \end{bmatrix} \text{and} \begin{bmatrix} 0 & 0 & 0 \\ a_2 & 0 & 0 \\ a_1 & a_2 & 0 \end{bmatrix} \tag{10}$$

constitute a splitting of the circulant matrix

$$\mathbf{A} = \begin{bmatrix} 1 & a_1 & a_2 \\ a_2 & 1 & a_1 \\ a_1 & a_2 & 1 \end{bmatrix}, \tag{11}$$

it should be evident that the recursive filter of Equation 9, when input $x(n)$ has the appropriate periodicity, can be interpreted as a stationary iterative procedure for solving $\mathbf{A}\underline{y} = \underline{x}$. Furthermore, comparing the structure of the present splitting of $\mathbf{A}$ with the splitting associated with the Gauss-Seidel technique, we realize, – perhaps after a little reflection, that the recursive filter of Equation 9 corresponds to a Gauss-Seidel iteration applied to $\mathbf{A}\underline{y} = \underline{x}$ where the equations are cycled through starting from the last equation and proceeding upwards towards the first equation, before the iteration again continues with the last equation.

Can this observation be useful for some practical purpose? Maybe not, but it is still an interesting connection that can be utilized as a teaching tool in reinforcing a concept by looking at the same thing from two quite different points of view.

4. EXAMPLE 2: ITERATIVE EQUATIONS SOLVERS AS ADAPTIVE FILTERS

Here we point out that the classical LMS algorithm for adaptive filtering can be derived directly from the Richardson iteration. We also present a connection with the more recently introduced *Fast Euclidian Direction Search* (FEDS) adaptive algorithm [4, 5, 6].

The purpose of any adaptive filter is to estimate some desired signal $d(n)$ by applying, to a related signal $x(n)$, a time varying filter whose coefficients at time n can be represented by a vector $\underline{h}(n) = [h_0(n), h_1(n), \ldots, h_{M-1}(n)]^T$. One of the quality metrics of an adaptive filter algorithm is the speed with which $\underline{h}(n)$ approaches the solution of the Wiener-Hopf equation in a stationary situation. The Wiener-Hopf equation is $\mathbf{R}\underline{h} = \underline{r}$, where $\mathbf{R}$ is the autocorrelation matrix of the input signal $\mathbf{R} = E\{\underline{x}(n)\underline{x}^T(n)\}$. $E\{\}$ is the expectation operator, and $\underline{x}(n) = [x(n), x(n-1), \ldots, x(n-M+1)]^T$ is the vector of random variables corresponding to signal $x(n)$ at various time instants[1]. $\underline{r}$ is the crosscorrelation vector defined by $\underline{r} = E\{\underline{x}(n)d(n)\}$. The key in traditional derivations of the LMS algorithm is the application of a steepest descent strategy to the objective function, $E\{(d(n) - \underline{h}^T(n)\underline{x}(n))^2\}$, and then using *instantaneous estimates* of $\mathbf{R}$ (i.e. $\underline{x}(n)\underline{x}^T(n)$) and $\underline{r}$ (i.e. $\underline{x}(n)d(n)$). Subsequently, in standard LMS presentations one goes back to the steepest descent algorithm formulated in terms of $\mathbf{R}$ and $\underline{r}$ to derive allowable ranges for the algorithm's step size parameter and its convergence behavior.

Suppose now that we knew the Richardson iteration, i.e. Equation 8, and formulated it for the Wiener-Hopf equation. Furthermore, suppose we wanted to devise an iterative solution for that equation using the instantaneous estimates $\underline{x}(n)\underline{x}^T(n)$ and $\underline{x}(n)d(n)$ for $\mathbf{R}$ and $\underline{r}$, respectively. Doing this immediately results in $\underline{h}^{(new)} = \underline{h}^{(old)} + \mu\underline{x}(n)\{\underline{d}(n) - \underline{x}^T(n)\underline{h}^{(old)}\}$ which is seen to be identical to the standard LMS algorithm. The standard results on step size range and convergence behavior can now be established as direct consequences of the properties of the Richardson iterative scheme in its simplest form. Such results predate the publication of the LMS algorithm by decades.

The key message here is: If exposure to the elementary theory of classical stationary iterative techniques can be taken for granted, the presentation of the LMS algorithm can be substantially simplified in that it can be established directly as a consequence of the use of *instantaneous estimates* and the *Richardson iterative scheme* for the Wiener-Hopf equation. Such a presentation of the LMS algorithm would also give students a better perspective in relating it to another important branch of science. As an anecdotal remark, we mention that whereas the LMS algorithm was in its infancy in the beginning 1960's, the Richardson iterative scheme of 1910, was rapidly losing favor in the numerical linear algebra community as a consequence of a 1962 statement attributed to Varga in a recent survey paper on iterative linear equation solvers [3]: *" Richardson's method has the disadvantage of being numerically unstable"*. As such one might say there is some truth in the snide remarks, mostly put forward with friendly intent, portraying DSP professionals as scavengers of the mathematicians' garbage can.

As a more modern example of the utility of iterative equation solvers for linear systems of equations, we mention the recently introduced *Fast Euclidian Direction Search* (FEDS) algorithm of Bose et. al. [4, 5, 6] which was originally developed within a classical adaptive filter theory framework. We now argue that, had the LMS algorithm been derived through the use of Richardson's iteration, the above mentioned FEDS algorithm almost suggests itself: Having an algorithm based on the Richardson iteration applied to the Wiener-Hopf equation using instantaneous estimates of the correlations quantities involved, the logical followup questions to ask would be: *1) What about applying a more sophisticated stationary iterative equation solver than the Richardson scheme?* and *2) What about using better estimates of $\mathbf{R}$ and $\underline{r}$ than what was done for LMS?* Self suggesting

[1] As is common, we do not use any notation to distinguish the cases when $\underline{x}(n)$ is to be interpreted as a random vector and when it is to be interpreted as a vector of signal samples.

answers would be: Use the Gauss-Seidel iterative scheme and use $\mathbf{X}^T(n)\mathbf{X}(n)$ and $\mathbf{X}^T(n)\underline{d}(n)$, where $\mathbf{X}(n)$ and $\underline{d}(n)$ are given as

$$\mathbf{X}(n) = \begin{bmatrix} x(n) & x(n-1) & \cdots & x(n-M+1) \\ x(n-1) & x(n-2) & \cdots & x(n-M) \\ \vdots & & \cdots & \vdots \\ x(0) & x(-1) & \cdots & x(-M+1) \end{bmatrix},$$

and $\underline{d}(n) = [d(n), d(n-1), \ldots, d(0)]^T$. In a practical adaptive scheme, we would emphasize the recent past at the expense of the more distant past by including only the L most recent rows/elements of $\mathbf{X}(n)$ and $\underline{d}(n)$ or by exponentially weighting the rows/elements of the mentioned matrix/vector. Applying one Gauss-Seidel step according to Equation 2 to

$$\mathbf{X}^T(n)\mathbf{X}(n)\underline{h} = \mathbf{X}^T(n)\underline{d}(n) \tag{12}$$

at time n, i.e. updating only one element of $\underline{h}$ at time n and going through the elements of $\underline{h}$, one for each time instant in a cyclical fashion, the FEDS algorithm is immediately established. This observation was also made by Bose, see his forthcoming book [7]. Using *block exponentially weighted* versions of the $\mathbf{X}(n)$ matrix and $\underline{d}(n)$ vector[2] an algorithm with multiplicative complexity given by $4M$, – M being the number of filter coefficients, and convergence properties far superior to the LMS algorithm results. In some situations this algorithm is also claimed to be competitive with Recursive Least Squares (RLS) algorithms. Again, presented as an application of the theory of stationary iterative linear equation solvers, this algorithm can be presented in a very elementary fashion.

5. SUMMARY AND CONCLUSIONS

In this paper we have presented two ”non-standard” connections between important digital signal processing concepts and iterative techniques for solving sets of linear equations. This sheds new light to previously established results that we believe can be used effectively in the teaching of digital signal processing.

6. REFERENCES

[1] Y. Saad, *Iterative Methods for Sparse Linear Systems*, PWS Publishing, 1996.

[2] L. F. Richardson, “The approximate arithmetical solution by finite differences of physical problems involving differential equations with an application to the stresses to a masonry dam,” *Philos. Trans. Roy. Soc. London*, vol. Ser A 210, 1910.

[3] Y. Saad and H. A. van der Horst, “Iterative solution of linear systems in the 20-th century,” *Journal of Computational and Applied Mathematics*, vol. 123, no. 1-2, pp. 1–33, November 2000.

[4] G. F. Xu, T. Bose, and J. Schroeder, “The Euclidian direction search algorithm for adaptive filtering,” in *Proc. ISCAS*, Orlando, Fl., USA, May 1999, pp. 146 –149 (Vol. III).

[5] G. F. Xu and T. Bose, “Analysis of the Euclidian direction set adaptive algorithm,” in *Proc. ICASSP*, Seattle, Washington, USA, May 1998, pp. 1689–1692.

[6] Mei-Quin Chen, T. Bose, and Guo Fang Xu, “A direction set based algorithm for adaptive filtering,” *IEEE Trans. Signal Processing*, vol. 47, no. 2, pp. 535–539, Feb. 1999.

[7] Tamal Bose, *Digital Signal and Image Processing*, John Wiley, 2003.

[2] The first M rows/elements of $\mathbf{X}(n)$ and $\underline{d}(n)$ are non-weighted, the next M rows/elements are multiplied by weighting factor $\lambda^{1/2}$, the following M rows/elements are multiplied by $\lambda^{2/2}$ etc.