Blocked Iterative Sparse Linear System Solvers for Finite Fields*

Erich Kaltofen

June 7, 1996

Department of Mathematics, North Carolina State University
Raleigh, North Carolina 27695-8205; Internet: `kaltofen@eos.ncsu.edu`
URL: `http://www4.ncsu.edu/~kaltofen`

Extended Abstract

The problem of solving a large sparse or structured system of linear equations in the symbolic context, for instance when the coefficients lie in a finite field, has arisen in several applications. A famous example are the linear systems over $\mathbb{F}_2$, the field with 2 elements, that arise in sieve based integer factoring algorithms. For example, for the factorization of the RSA-130 challenge number several column dependencies of a $3\,504\,823 \times 3\,516\,502$ matrix with an average of 39.4 non-zero entries per column needed to be computed [10]. A second example is the Berlekamp polynomial factorization algorithm [6]. In that example, the matrix is not explicitly constructed, but instead a fast algorithm for performing the matrix times vector product is used. Further examples for such "black box matrices" arise in the power series solution of algebraic or differential equations by undetermined coefficients. The arising linear systems for the coefficients usually have a distinct structure that allows a fast coefficient matrix times vector product.

Here we shall assume that the coefficients are from a finite field, which is the case in the integer and polynomial factoring algorithms. In other symbolic applications computations modulo a large prime number are often used to discover the term structure of the solutions, as is done, for example, in the Macaulay system. Exact coefficient values can sometimes be recovered by Chinese remaindering or continued fraction approximation. Modular arithmetic also becomes necessary for very large inputs, where the rational coefficients would grow too large to be efficiently handled.

*This material is based on work supported in part by the National Science Foundation under Grant No. CCR-9319776.
Appears in Proc. Stratagem'96, INRIA Sophia Antipolis, pp. 91–95 (1996).

Algorithms for solving linear systems that use a fast coefficient matrix times vector product are sometimes referred to as "matrix free," as the coefficient matrix is never explicitly constructed. Numerical linear algebra has the Krylov, Lanczos, and conjugate gradient methods. As it turns out, these methods can be implemented over a finite field. The solutions cannot be approximated in this setting but instead the iterations must continue until certain subspaces are exhausted. With n we shall denote the dimensions of the coefficient matrix A. Let $b \in \mathbb{F}_q^n$ and consider the Krylov subspace spanned by the vectors

$$b, Ab, A^2b, \ldots, A^ib, \ldots \tag{1}$$

The approach due to Wiedemann [14] works in two rounds. First a linear dependency of the vectors (1) is found. This is done by applying a random linear map

$$a_0 = u^{\mathrm{T}}b, a_1 = u^{\mathrm{T}}Ab, \ldots, a_i = u^{\mathrm{T}}A^ib, \ldots$$

where u^{T} is the transpose of a random linear vector, and by computing a linear generator $c_0 + c_1\lambda + \cdots + c_m\lambda^m \in \mathbb{F}_q[\lambda]$ for the sequence $\{a_i\}$ of field elements. We then have

$$\forall\, j \geq 0\colon\ c_0a_j + c_1a_{j+1} + \cdots + c_ma_{j+m} = 0.$$

The generator can be found from no more than $2n$ initial sequence elements by the Berlekamp/Massey algorithm. It can proven for large q that the resulting generator with high probability remains one for the sequence of Krylov vectors (1). Thus

$$c_0b + c_1Ab + \cdots + c_mA^mb = 0$$

and if $c_0 \neq 0$, which is the case if the matrix is non-singular, one obtains the solution

$$A \cdot \frac{1}{c_0}(c_1b + c_2Ab + \cdots + c_mA^{m-1}b) = b$$

by a second round of matrix times vector products. Therefore, if one does not store all A^ib in the first round it can take as many as $3n$ iterations before the exact solution is constructed. Wiedemann and others [8] give several variants of this approach, including solution of singular systems. Most importantly, Wiedemann manages to analyze the probability of success for one of his variants even if the field is $\mathbb{F}_2$.

A second approach is based on the Lanczos method. For a moment, suppose that the matrix A is real, symmetric, and positive definite. Then the bilinear map

$$\langle u, v\rangle_A = u^{\mathrm{T}} A v$$

constitutes an inner product. Lanczos proceeds by computing an orthogonal basis for (1) with respect to $\langle \cdot, \cdot \rangle_A$, namely

$$w_0 = b, w_1 = Aw_0 - \alpha_1 w_0, w_{i+2} = Aw_{i+1} - \alpha_{i+1} w_{i+1} - \beta_i w_i.$$

Here

$$\alpha_{i+1} w_{i+1} = \frac{\langle Aw_{i+1}, w_{i+1}\rangle_A}{\langle w_{i+1}, w_{i+1}\rangle_A} w_{i+1} \text{ and } \beta_i w_i = \frac{\langle Aw_{i+1}, w_i\rangle_A}{\langle w_i, w_i\rangle_A} w_i$$

are the projections with regard to $\langle \cdot, \cdot \rangle_A$ of Aw_{i+1} onto w_{i+1} and w_i, since $\langle Aw_{i+1}, w_j\rangle_A = \langle w_{i+1}, Aw_j\rangle_A = 0$ kills all the other projections onto w_j for $j < i$. If A is non-singular, then

$$A^{-1} b = \sum_{i=0}^{m-1} \frac{\langle A^{-1} b, w_i\rangle_A}{\langle w_i, w_i\rangle_A} w_i = \sum_{i=0}^{m-1} \frac{b^{\mathrm{T}} w_i}{\langle w_i, w_i\rangle_A} w_i.$$

Note that $A^{\mathrm{T}} = A$. The only matrix times vector products needed are $Aw_0, Aw_1, \ldots, Aw_{m-1}$, where $m \leq n$. One can perform the algorithm over a finite field as long as $\langle w_{i+1}, w_{i+1}\rangle_A \neq 0$. If A is not symmetric, LaMacchia and Odlyzko [9] suggest to use $\bar{A} = A^{\mathrm{T}} D A$ and $\bar{b} = A^{\mathrm{T}} D b$, where D is a random diagonal matrix. The matrix D helps in avoiding self-orthogonal w_{i+1}. The cost is then no more than $2n$ matrix times vector products, noting that one needs a black box for A^{T} as well. By the transposition principle (Tellegen's theorem) [13] a black box for A can be converted to a black box for A^{T} at no loss in time efficiency. We note that the overall form of the conjugate gradient method is very similar to the Lanczos algorithm. We know of no complete failure probability analysis for either method. For large finite fields a more complicated preconditioner is provable for the Lanczos method at least in the non-singular case [4].

When very large inputs are considered it becomes necessary to appeal to parallel techniques, that is reduce the number of outer loop iterations. This is accomplished by blocking the vector b. Coppersmith [2] describes such an approach for the Wiedemann algorithm. Choose the block vectors $\mathbf{u} \in \mathbb{F}_q^{n\times\beta}$ and $\mathbf{b} \in \mathbb{F}_q^{n\times\beta}$ and compute the matrix sequence

$$\mathbf{a}_i = \mathbf{u}^{\mathrm{T}} A^{i+1}\mathbf{b} \in \mathbb{F}_q^{\beta\times\beta}, \quad 0 \le i < \frac{2n}{\beta} + 2.$$

The linear generator then becomes a vector polynomial

$$c_0 + c_1\lambda + \cdots + c_d\lambda^d \in \mathbb{F}_q^{\beta}[\lambda],\ d \le \lceil n/\beta \rceil.$$

such that

$$\forall j \ge 0 : \sum_{i=0}^{d} \mathbf{a}_{j+i} c_i = \sum_{i=0}^{d} \mathbf{u}^{\mathrm{T}} A^{i+j}\mathbf{b} c_i = 0.$$

If one can drop the projection by $\mathbf{u}$, one again obtains a linear dependency of the vectors $A^i\mathbf{b}c_i$ leading to a solution. The generator polynomial can either be computed by a generalization of the Berlekamp/Massey algorithm [2] or by a block Toeplitz solver [5]. Parallelization is possible either by performing each $A \cdot (A^{i-1}\mathbf{b})$ in a parallel fashion, as is done over $\mathbb{F}_2$ where each column of $(A^{i-1}\mathbf{b})$ can be stored in a different bit position inside a computer word, or by computing the sequences of the ν-th columns of the $\mathbf{a}_i$ separately. The latter is especially useful when the matrix is given by a black box procedure that uses very little space, like in the polynomial factoring application.

A similar blocking approach is described for the Lanczos method by Coppersmith [1] and by Montgomery [12]. Again, a problem arises when $\mathbf{w}_{i+1}^{\mathrm{T}} A\mathbf{w}_{i+1}$ is a singular matrix, where $\mathbf{w}_{i+1} \in \mathbb{F}_q^{n\times\beta}$. A possibility described by Coppersmith is to maintain a new set of vectors that orthogonally spans a subspace of the Krylov space. Unfortunately, all known block methods must be considered heuristics over small finite fields, although they have been observed to work in practice on the matrices from integer factoring and polynomial factorization.

Fortunately, we have a complete analysis of a variant of the block Wiedemann algorithm for large fields which reveals several interesting properties [5]. For one, it appears that blocking not only allows for parallelization, but it improves the probability that the algorithms succeed in finding a solution. The analysis in [5] is based on the observation that if the random entries of $\mathbf{u}$ and $\mathbf{b}$ are picked in a special way, the block algorithm simulates the

Wiedemann algorithm. Thus, the needed conditions are even more likely to occur when the entries are chosen at random. Furthermore, the Corollary to Theorem 7 in [5] shows that one may compute a solution to a linear system sequentially with only $(1+\epsilon)n$ matrix times vector products and an additional $O(n^{2+o(1)})$ arithmetic operation while using $O(n)$ intermediate storage. This result is accomplished by using blocks of size β^2 for $\mathbf{u}$. Note that in the unsymmetric case both the Lanczos and the conjugate gradient methods appear to require at the worst $2n$ matrix times vector products. These improvements of the Wiedemann approach carries over to the parallel case.

The block Lanczos method over $\mathbb{F}_2$ has been implemented on an IBM 3090 computer by Coppersmith [1] and on a CRAY-C90 computer by Montgomery [12]. The block Wiedemann method has been implemented over $\mathbb{F}_2$ by Coppersmith on an IBM RS-6000 [2] and by Lobo on an IBM SP-2 [7]. Furthermore, over $\mathbb{F}_p$ the block Wiedemann method has been used on a network of workstations to solve linear systems and factor polynomials modulo a prime number [6, 11, 3]. The most pressing open problems are to supply a probabilistic analysis for the block approach when the field is small, and to explore reduction of matrix times vector products. In the current state of knowledge, the Wiedemann approach seems to have the edge.

Acknowledgement: I thank Wayne Eberly for the discussions we have had on the Lanczos approach.

References

[1] D. Coppersmith. Solving linear systems over GF(2): block Lanczos algorithm. *Lin. Algebra Applic.*, 192:33–60, 1993.

[2] D. Coppersmith. Solving homogeneous linear equations over GF(2) via block Wiedemann algorithm. *Math. Comput.*, 62(205):333–350, 1994.

[3] A. Diaz, M. Hitz, E. Kaltofen, A. Lobo, and T. Valente. Process scheduling in DSC and the large sparse linear systems challenge. *J. Symbolic Comput.*, 19(1–3):269–282, 1995.

[4] W. Eberly and E. Kaltofen. Work in progress. To be published.

[5] E. Kaltofen. Analysis of Coppersmith's block Wiedemann algorithm for the parallel solution of sparse linear systems. *Math. Comput.*, 64(210):777–806, 1995.

[6] E. Kaltofen and A. Lobo. Factoring high-degree polynomials by the black box Berlekamp algorithm. In J. von zur Gathen and M. Giesbrecht, editors, *Proc. Internat. Symp. Symbolic Algebraic Comput. ISSAC '94*, pages 90–98, New York, N. Y., 1994. ACM Press.

[7] E. Kaltofen and A. Lobo. Distributed matrix-free solution of large sparse linear systems over finite fields. In A. M. Tentner, editor, *High Performance Computing 1996*, pages 244–247, San Diego, CA, 1996. Society for Computer Simulation, Simulation Councils, Inc.

[8] E. Kaltofen and B. D. Saunders. *On Wiedemann's method of solving sparse linear systems*, volume 539 of *Springer Lect. Notes Comput. Sci.*, pages 29–38. Springer Verlag, Heidelberg, Germany, 1991.

[9] B. A. LaMacchia and A. M. Odlyzko. *Solving large sparse linear systems over finite fields*, volume 537 of *Lect. Notes Comput. Sci.*, pages 109–133. Springer Verlag, Heidelberg, Germany, 1991.

[10] A. K. Lenstra. Factorization of RSA-130 using the number field sieve, April 1996. Message posted on newsgroup `sci.crypt.research`.

[11] A. Lobo. *Matrix-free linear system solving and applications to symbolic computation.* PhD thesis, Rensselaer Polytechnic Institute, Troy, New York, December 1995.

[12] P. L. Montgomery. *A block Lanczos algorithm for finding dependencies over GF(2)*, volume 921 of *Springer Lecture Notes Comput. Sci.*, pages 106–120. Springer Verlag, Heidelberg, Germany, 1995.

[13] P. Penfield Jr., R. Spencer, and S. Duinker. *Tellegen's Theorem and Electrical Networks.* M.I.T. Press, Cambridge, MA, 1970.

[14] D. Wiedemann. Solving sparse linear equations over finite fields. *IEEE Trans. Inf. Theory*, IT-32:54–62, 1986.