The randomizing FFT: an alternative to pivoting in Gaussian elimination

D. Stott Parker Brad Pierce
`stott@cs.ucla.edu` `pierce@cs.ucla.edu`
Computer Science Department
University of California
Los Angeles, CA 90024-1596

Abstract

Recently, randomizing input transformations have been developed for streamlining Gaussian elimination. Multiplying by special sparse random matrices, these transformations produce an equivalent linear system which, with probability 1, Gaussian elimination can solve without any augmentation by pivoting.

In this paper we study properties of the unitary *Randomizing Discrete Fourier Transform* (RDFT), which scales rows by random values from the complex unit circle and then applies the conventional DFT. When the problem size is a power of 2, the *Randomizing Fast Fourier Transform* (RFFT) algorithm can efficiently implement the RDFT, using existing FFT code, on the many parallel and vector machines whose architectures support FFT-like computations.

Keywords: Linear systems solution, Gaussian elimination, LU decomposition, pivoting, randomization, DFT, FFT, unitary transformations
AMS(MOS) subject classifications: 65F05, 65F40, 15A03, 15A52, 15A23, 15A15

1 Introduction

This paper presents a new alternative to pivoting in Gaussian elimination. Certain well-behaved randomizing transforms, specifically the unitary Randomizing Discrete Fourier Transform (RDFT), can be used presumptively to 'homogenize' a nonsingular linear system, yielding an equivalent system that, with probability 1, has no singular principal leading submatrices. When the problem size is a power of two, the RDFT can be implemented efficiently using the Randomizing Fast Fourier Transform (RFFT).

Earlier [4] we studied the randomization properties of 'random butterfly' transforms (RBTs). The RFFT requires fewer random values than the RBT and can reuse existing FFT code, but, unlike the RBT, uses complex arithmetic on real inputs.

1.1 The tail that wags the dog

High-performance computers are used heavily for solving large linear systems. These computers favor algorithms with little data movement, tight loops that run in cache, simple computations that can be performed with independent parallelism over a large set of values, and operations like summation that can be arranged in pipeline or with fan-in.

Many apparently suitable algorithms, however, that would seem to be easy to state and implement, instead get tangled up in the fine print of special cases. Gaussian elimination, for example, is typically interwoven with *pivoting*, a dynamic permutation of the rows and columns of the input matrix. Pivoting is a necessary evil that ensures (large) nonzero matrix entries on the diagonal, but that also interrupts the algorithm's data flow and introduces data movement that can slow high-performance computers.

A single pivot can take $O(n)$ steps on machines with limited interconnection schemes; in such a machine, the $O(n^2)$ data movements potentially required by pivoting, together with the synchronizations they entail, can consume more time than the $O(n^3)$ arithmetic operations required by Gaussian elimination. Significant overhead is also incurred on both MIMD and SIMD machines.

1.2 Randomization as a method to avoid degeneracy

The primary role of pivoting in Gaussian elimination is to cope with *degenerate* input matrices, *i.e.*, nonsingular matrices, such as $\begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$ or $\begin{pmatrix} 1 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{pmatrix}$, that have singular leading principal submatrices. On such matrices elimination will produce either a zero on the diagonal and then division by zero or, because of roundoff error, extreme cancellation on the diagonal and then division by inaccurate values. For nonsingular matrices this degeneracy is an artifact of the choice of vector basis, and under a random change of basis can be expected to evaporate.

Suppose we are given a linear system $A\mathbf{x} = \mathbf{b}$, where A is a nonsingular $n \times n$ matrix. Let V be a nonsingular $n \times n$ matrix whose entries are chosen randomly and independently. We should be able to solve the equivalent linear system $VA\mathbf{x} = V\mathbf{b}$ using pure Gaussian elimination *without any pivoting*.

Although this tactic may seem a natural one, upon reflection it should provoke some skepticism, because multiplication by arbitrary random matrices can be expected to magnify condition numbers and could potentially consume more time than Gaussian elimination itself. We shall, however, exhibit a *fast*, *unitary* transform that is sufficiently randomizing to guarantee (with probability 1) that application to a nonsingular matrix will yield a nondegenerate matrix.

2 Preliminaries

2.1 Randomness and zeroes of polynomials

Definition 1 *A* **random real index** θ *is a real-valued integrable function* $\theta : \mathcal{D} \to \mathbf{R}$ *with a strictly increasing (absolutely) continuous probability distribution on a real interval sample space* $\mathcal{D}$ *of nonzero measure.*

A 1-to-1 function of a random real index θ *is a* **random value** *if it is integrable with respect to the probability distribution of* θ*. A* **nonzero random value** *is a random value whose function never yields the value zero.*

All random values used in the RDFT are nonzero random values of the form $e^{i\theta}$.

Definition 2 *If* r *is a random value, a property* $P(r)$ **holds with probability 1** *when* $P(r)$ *is true on all of* $\mathcal{D}$ *except possibly a subset of measure 0.*[1]

---

[1] To support implementation with floating point numbers, we need the requirement that the quantity n of random numbers used be 'small' relative to the number of floating point numbers available. If there are 2^w possible floating point numbers, $n \ll 2^w$ is necessary for the matrices to be considered random in the sense intended here. Furthermore, the phrase "with probability 1" must be understood as relative to floating point numbers. If floating point numbers have t-bit mantissa precision, then (ignoring exponents and normalization and distribution) the probability that two arbitrarily-chosen floating point numbers are equal is at least 2^{-t}, and (assuming a good pseudorandom number generator) the phrase must be replaced by "with probability $1 - O(2^{-t})$". Since these $O(2^{-t})$ terms are distracting and are not likely to be significant (most computers today support $t \geq 48$), we omit them from now on, but the phrase "with probability 1" should be treated with care.

Lemma 1 *Let $p(x) = ax + b$, where a, b are fixed complex values. Let r be a nonzero random value. With probability 1, if $p(r) = 0$, then $a = b = 0$.*

Proof
If $p(r) = 0$ for some nonzero r, then either $a = b = 0$ or $a \neq 0$, $b \neq 0$ and $r = -b/a$. Because r is a random value, $r \neq -b/a$ with probability 1. □

Theorem 1 *Let $p(x_1, \ldots, x_n)$ be a fixed multivariate polynomial in n variables that each appear with degree at most 1. Let $r_1, \ldots, r_n$ be independently chosen nonzero random values. With probability 1, if $p(r_1, \ldots, r_n) = 0$, then p is the zero polynomial.*

Proof
By induction on n. Lemma 1 establishes the base case, where $n = 1$. For $n > 1$ we can collect terms in x_n and write

$$p(x_1, \ldots, x_n) \quad = \quad a(x_1, \ldots, x_{n-1})\; x_n \;+\; b(x_1, \ldots, x_{n-1}),$$

where both a and b are polynomials in $n-1$ variables that each appear with degree at most 1. If $p(r_1, \ldots, r_n) = 0$, then by Lemma 1, with probability 1, $a(r_1, \ldots, r_{n-1}) = b(r_1, \ldots, r_{n-1}) = 0$, so by induction, with probability 1, a and b, hence also p, are the zero polynomial. □

2.2 Sequences and submatrices

We write **sequences** of integers as i_1, i_2, ..., i_k. The **length** of this sequence is k. Following convention, Greek letters (α, β, γ, ...) will denote sequences.

Definition 3 *S_k^n is the set of* **strictly increasing sequences** *of length k drawn from $\{1, \ldots, n\}$.*

For example, $1, 2, 5 \in S_3^5$, but $2, 2, 5 \notin S_3^5$, and $1, 2, 5 \in S_3^6$.

Definition 4 *The* **submatrix** *$A[\,\alpha\,|\,\beta\,]$ of A with $\alpha = i_1, \ldots, i_p$ and $\beta = j_1, \ldots, j_q$ denotes the matrix formed from the entries of the indicated rows and columns of A:*

$$A[\,\alpha\,|\,\beta\,] \quad = \quad \begin{pmatrix} a_{i_1,j_1} & a_{i_1,j_2} & \cdots & a_{i_1,j_q} \\ a_{i_2,j_1} & a_{i_2,j_2} & \cdots & a_{i_2,j_q} \\ \vdots & \vdots & \ddots & \vdots \\ a_{i_p,j_1} & a_{i_p,j_2} & \cdots & a_{i_p,j_q} \end{pmatrix}.$$

We follow the convention that matrix row and column indexing begins with 1. For example,

$$\begin{pmatrix} 2 & 3 & 5 \\ 7 & 11 & 13 \\ 17 & 19 & 23 \end{pmatrix} [\,1,2\,|\,2,3\,] \quad = \quad \begin{pmatrix} 3 & 5 \\ 11 & 13 \end{pmatrix}.$$

3 The RDFT

3.1 The Discrete Fourier Transform

The **Discrete Fourier Transform (DFT)** is the family of unitary $n \times n$ matrices $F_n = (f_{ij})$ defined by

$$f_{ij} \quad = \quad \frac{1}{\sqrt{n}}\,(\,{\omega_n}^{(i-1)(j-1)}\,),$$

where ω_n is a primitive nth root of unity, typically $exp(-\frac{2\pi}{n}\mathrm{i})$. When n is a power of 2, the **Fast Fourier Transform (FFT)** algorithm results from factoring F_n into recursive products of butterfly transformations and permutations. Specifically:

$$F_{2n} \quad = \quad B_{2n} \;\; (F_n \oplus F_n) \;\; {\sigma_{2n}}^{-1} \quad = \quad B_{2n} \;\; (I_2 \otimes F_n) \;\; {\sigma_{2n}}^{-1},$$

where I_n is the $n \times n$ identity, B_{2n} is the butterfly

$$B_{2n} \quad = \quad \frac{1}{\sqrt{2}} \begin{pmatrix} I_n & D_n \\ I_n & -D_n \end{pmatrix}, \quad \text{where} \quad D_n \quad = \quad \mathrm{diag}(\; 1,\, \omega_{2n},\, {\omega_{2n}}^2,\, \ldots,\, {\omega_{2n}}^{n-1} \;),$$

and ${\sigma_{2n}}^{-1}$ is the 'perfect unshuffle' permutation [3, 5]. Van Loan [5] reviews how the FFT can be implemented efficiently on high-performance computers.

3.2 The Randomizing DFT

Definition 5 *A* **random matrix** *is a matrix whose complex-valued entries are functions of independent random values sharing a common distribution.*

For example, a diagonal matrix whose diagonal entries are complex exponentials $e^{i\theta}$ dependent on a random index θ with continuous distribution on the interval $[0, 2\pi)$ is a random matrix that is *always* unitary.

Definition 6 *A* **Randomizing Discrete Fourier Transform (RDFT)** *is any matrix of the form $F_n R$, where F_n is a DFT matrix and R is an $n \times n$ unitary random diagonal matrix.*

- All the diagonal entries of R are of the form $e^{i\theta}$.
- As R is a unitary random diagonal matrix, so is its Hermitian adjoint (conjugate transpose) R^*.
- The RDFT is a unitary transformation.

Definition 7 *A* **Randomizing Fast Fourier Transform (RFFT)** *algorithm is any implementation of the RDFT that incorporates the FFT algorithm.*

3.3 Randomization properties of the RDFT

Gaussian elimination is a sequence of transformations to an $n \times n$ matrix $A = (a_{ij})$, reducing it to upper-diagonal form in $n - 1$ steps. With the initial assignment $a_{ij}^{(1)} = a_{ij}$, it can be equationally defined for $1 \le k \le n - 1$ as

$$a_{ij}^{(k+1)} = \begin{cases} 0 & i \ge k+1,\, j = k \\ a_{ij}^{(k)} - \dfrac{a_{ik}^{(k)} a_{kj}^{(k)}}{a_{kk}^{(k)}} & i \ge k+1,\, j \ge k+1 \\ a_{ij}^{(k)} & \text{otherwise.} \end{cases}$$

Gaussian elimination works only when, for all $1 \le k \le n - 1$, the $a_{kk}^{(k)}$ are nonzero. It is well-known [2, p.96] that this is equivalent to the following:

> An $n \times n$ matrix A is **Gauss eliminable** iff all its leading principal submatrices are nonsingular.

Lemma 2 *Let F_n be a DFT matrix of size n. Then for all $1 \le k \le n$ and all sequences $\alpha \in S_k^n$, the submatrix $F_n[\,1,\ldots,k \,|\, \alpha\,]$ is nonsingular.*

Proof
Let $\alpha = \alpha_1,\ldots,\alpha_k$. Then $F_n[\,1,\ldots,k \,|\, \alpha\,]$ is the $k \times k$ matrix

$$\frac{1}{\sqrt{n}} \begin{pmatrix} {\omega_n}^{(\alpha_1-1)\,0} & {\omega_n}^{(\alpha_2-1)\,0} & \cdots & {\omega_n}^{(\alpha_k-1)\,0} \\ {\omega_n}^{(\alpha_1-1)\,1} & {\omega_n}^{(\alpha_2-1)\,1} & \cdots & {\omega_n}^{(\alpha_k-1)\,1} \\ \vdots & \vdots & \ddots & \vdots \\ {\omega_n}^{(\alpha_1-1)(k-1)} & {\omega_n}^{(\alpha_2-1)(k-1)} & \cdots & {\omega_n}^{(\alpha_k-1)(k-1)} \end{pmatrix}.$$

Thus the submatrix is a constant multiple of a transposed Vandermonde matrix (x_j^{i-1}), where $x_j = {\omega_n}^{(\alpha_j-1)}$. Because ω_n is a primitive nth root of unity, the x_j are distinct, hence the Vandermonde matrix must be nonsingular [2, p.178ff]. □

Theorem 2 *The RDFT yields Gauss eliminable matrices,* i.e.*: with probability 1, if A is an $n \times n$ nonsingular matrix, then Gaussian elimination can operate without any pivoting on $\widetilde{A} = (F_n R)A$ and on $\widehat{A} = A(F_n R)^*$.*

Proof
We may consider only the case of $\widetilde{A}$ without loss of generality, since both nonsingularity and Gauss eliminability are invariant under adjoint.

It is a consequence of the *Binet-Cauchy theorem* [1, §39] that, for any $n \times n$ matrices X and Y, for any $1 \le k \le n$, and for any sequences $\alpha, \beta \in S_k^n$,

$$\det\,(XY)[\,\alpha \,|\, \beta\,] \;=\; \sum_{\gamma \in S_k^n} \det X[\,\alpha \,|\, \gamma\,] \;\; \det Y[\,\gamma \,|\, \beta\,].$$

In particular,

$$\begin{aligned} \det \widetilde{A}[\,1,\ldots,k \,|\, 1,\ldots,k\,] &= \sum_{\alpha \in S_k^n} \det\,(F_n R)[\,1,\ldots,k \,|\, \alpha\,] \;\; \det A[\,\alpha \,|\, 1,\ldots,k\,] \\ &= \sum_{\alpha \in S_k^n} \det A[\,\alpha \,|\, 1,\ldots,k\,] \;\; \det F_n[\,1,\ldots,k \,|\, \alpha\,] \;\; \left(\prod_{j \in \alpha} r_j \right). \end{aligned}$$

Thus *det* $\widetilde{A}[\,1,\ldots,k \,|\, 1,\ldots,k\,]$ is a polynomial of degree no more than one in the random values of R. Corresponding to each index α is a monomial summand constructed from a coefficient *det* $A[\,\alpha \,|\, 1,\ldots,k\,]$, a scalar factor *det* $F_n[\,1,\ldots,k \,|\, \alpha\,]$ that is guaranteed nonzero by Lemma 2, and a product $\left(\prod_{j\in\alpha} r_j\right)$ of a distinct combination of random values.

If $\widetilde{A}$ were not Gauss eliminable, *i.e.*, if there were some k such that *det* $\widetilde{A}[\,1,\ldots,k \,|\, 1,\ldots,k\,] = 0$, then Theorem 1 would imply that, with probability 1, *det* $A[\,\alpha \,|\, 1,\ldots,k\,] = 0$ for all $\alpha \in S_k^n$, hence that the first k columns of the nonsingular matrix A were linearly dependent. □

Note that the proof of Theorem 2 would go through with only minor changes if, rather than using n independent random values in the definition of the RDFT, we were to repeatedly square just one, say

$$R \;=\; \operatorname{diag}\,(r, r^2, r^4, \ldots, r^{2^{n-1}}).$$

4 Concluding remarks

The Randomizing Discrete Fourier Transform (RDFT) scales the rows of a linear system by random values from the complex unit circle, then transforms the result with the DFT. We have proved that the randomization properties of the RDFT suffice to obviate pivoting in Gaussian elimination. The RDFT is a unitary transformation, hence should not degrade the numerical stability of well-conditioned problems. When the problem size is a power of 2, the RDFT can be implemented efficiently with a Randomizing Fast Fourier Transform (RFFT) algorithm, reusing existing code, on the many parallel and vector machines whose architectures support FFT-like computations.

References

[1] A.C. Aitken. *Determinants and Matrices.* University mathematical texts. Interscience Publishers, New York, 9th edition, 1956.

[2] G. H. Golub and C. F. Van Loan. *Matrix Computations: second edition.* Johns Hopkins University Press, Baltimore, Maryland, 1989.

[3] D.S. Parker. Notes on shuffle/exchange-type switching networks. *IEEE Trans. Comput.*, C-29(3):213–222, March 1980.

[4] D.S. Parker. Random butterfly transformations with applications in computational linear algebra. Technical Report CSD-950023, UCLA Computer Science Department, 1995.

[5] C. Van Loan. *Computational Frameworks for the Fast Fourier Transform*, volume 10 of *Frontiers in applied mathematics.* SIAM, Philadelphia, 1992.