Black Box Linear Algebra

William J. Turner
North Carolina State University
www.math.ncsu.edu/~wjturner

Symbolic Computation

Symbols or exact arithmetic

Not floating point numbers (numerical analysis)

Randomized Algorithms

Monte Carlo: Always fast, probably correct

Las Vegas: Probably fast, always correct

Solving Linear Systems of Equations

Solve the linear system

$$\begin{bmatrix} 1 & 3 & 0 & 3 \\ 0 & 1 & 0 & 0 \\ 3 & 3 & -2 & 3 \\ 0 & -3 & 0 & -2 \end{bmatrix} x = \begin{bmatrix} 4 \\ 2 \\ 6 \\ -4 \end{bmatrix}$$

Gaussian Elimination

$$\begin{bmatrix} 1 & 3 & 0 & 3 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -2 & -6 \\ 0 & 0 & 0 & -2 \end{bmatrix} x = \begin{bmatrix} 4 \\ 2 \\ 6 \\ 2 \end{bmatrix} \quad \Longrightarrow \quad x = \begin{bmatrix} 1 \\ 2 \\ 0 \\ -1 \end{bmatrix}$$

Black Box Matrix Model

Only matrix-vector products allowed

Algorithms oblivious to how matrix-vector product is computed

Implementations may be efficient in time or space

Can compute Krylov sequence $\{A^i b\}_{i=0}^{\infty}$

Examples of Black Box Matrices

Example	storage	time
Arbitrary matrix	n^2	$O(n^2)$
Sparse matrix, η nonzero entries	$O(\eta)$	$O(\eta)$
Hilbert matrix $A_{i,j} = \frac{1}{i+j-1}$	$O(1)$	$O(n)$
Toeplitz matrix	$O(n)$	$O(n \log(n) \log\log(n))$

Linearly Generated Sequences

Let $\mathbb{V}$ be a vector space over field $\mathbb{F}$

A sequence

$$\{a_i\}_{i=1}^{\infty} \subset \mathbb{V}$$

is *linearly generated* if and only if there exist

$$c_0, \ldots, c_m \in \mathbb{F}, \quad c_m \neq 0$$

such that for all $j \geq 0$

$$c_0 a_j + c_1 a_{j+1} + \cdots + c_m a_{j+m} = 0$$

$$a_{j+m} = \frac{1}{c_m} \left(c_0 a_j + c_1 a_{j+1} + \cdots + c_{m-1} a_{j+m-1} \right)$$

The polynomial $f(\lambda) = c_0 + c_1\lambda + \cdots + c_m\lambda^m$ generates $\{a_i\}_{i=1}^{\infty}$

Any polynomial multiple of f also generates $\{a_i\}_{i=1}^{\infty}$

There exists a unique monic generator of minimal degree, the *minimum polynomial* of sequence

Example: Fibonacci Numbers

Let $\{a_i\}_{i=0}^{\infty} = \{0, 1, 1, 2, 3, 5, 8, 13, 21, 34, \ldots\}$

The minimum polynomial is $f = \lambda^2 - \lambda - 1$

$$a_{j+2} = a_{j+1} + a_j$$

Another generator is $(\lambda + 1)f = \lambda^3 - 2\lambda - 1$

$$a_{j+3} = 2a_{j+1} + a_j$$

And also $\lambda^k f = \lambda^{k+2} - \lambda^{k+1} - \lambda^k$

$$a_{j+k+2} = a_{j+k+1} + a_{j+k}$$

The characteristic polynomial $\det(\lambda I - A)$ generates the matrix sequence $\{A^i\}_{i=0}^{\infty}$ (Cayley-Hamilton Theorem)

The minimum polynomial f^A of the matrix sequence generates the Krylov sequence $\{A^i b\}_{i=0}^{\infty}$

$$A^j f^A(A) b = 0$$

Suppose $g = g_0 + g_1\lambda + \cdots + g_m\lambda^m \in \mathbb{F}[\lambda]$ linearly generates the Krylov sequence

$$g(A)b = g_0 b + g_1 Ab + \cdots + g_m A^m b = 0$$

If A is invertible and $g_0 \neq 0$,

$$x = A^{-1}b = -\frac{1}{g_0}(g_1 b + g_2 Ab + \cdots + g_m A^{m-1} b)$$

For any vector u, the minimum polynomial $f^{A,b}$ of the Krylov sequence $\{A^i b\}_{i=0}^{\infty}$ generates the bilinear projection of matrix powers $\{u^\mathsf{T} A^i b\}_{i=0}^{\infty}$

$$u^\mathsf{T} A^j f^{A,b}(A) b = 0$$

For u choosen randomly from finite $S^n \subset \mathbb{F}^n$, the minimum polynomial $f_u^{A,b}$ of the bilinear projection also generates the Krylov sequence

$$f_u^{A,b} = f^{A,b}$$

Berlekamp-Massey Algorithm

- From coding theory
- Interpolates elements of scalar sequence
- Similar to Newton iteration

Input: Scalar sequence $\{a_i\}_{i=0}^{\infty} \subset \mathbb{F}$ with generator g with $\deg(g) \leq m$
Output: Minimum polynomial f of sequence

1: $f \leftarrow 1$ {Initial guess}
2: **for** $r = 0$ to $2m - 1$ **do** {f generates $a_0, \ldots, a_{r-1}$}
3: $f = c_0 + c_1\lambda + \cdots + c_d\lambda^d$
4: $\Delta \leftarrow c_0 a_{r-d} + c_1 a_{r-d+1} + \cdots + c_d a_r$
5: **if** $\Delta \neq 0$ **then** {f does not generate $a_0, \ldots, a_r$}
6: update f to generate $a_0, \ldots, a_r$
7: **end if**
8: **end for**

Example: Fibonacci Numbers – continued

Let $\{a_i\}_{i=0}^{\infty} = \{0, 1, 1, 2, 3, 5, 8, 13, 21, 34, \ldots\}$

r	Δ	recursion
0	0	$a_j = 0$
1	1	$a_{j+2} = a_j$
2	1	$a_{j+2} = a_{j+1} + a_j$
3	0	$a_{j+2} = a_{j+1} + a_j$
4	0	$a_{j+2} = a_{j+1} + a_j$
5	0	$a_{j+2} = a_{j+1} + a_j$
6	0	$a_{j+2} = a_{j+1} + a_j$
7	0	$a_{j+2} = a_{j+1} + a_j$

Wiedemann's Algorithm

Input: nonsingular $A \in \mathbb{F}^{n \times n}$ and $b \in \mathbb{F}^n$

Output: $x \in \mathbb{F}^n$ such that $Ax = b$

1: compute $\{A^i b\}_{i=0}^{2n-1}$

2: $u \leftarrow$ random vector in S^n where $S \subset \mathbb{F}$

3: compute $\{u^{\mathsf{T}} A^i b\}_{i=0}^{2n-1}$

4: use Berlekamp-Massey to compute $f^{A,b} = c_0 + c_1\lambda + \cdots + c_m\lambda^m$

5: $x \leftarrow -\frac{1}{c_0}(c_1 b + c_2 Ab + \cdots c_m A^{m-1} b)$

$\det(\lambda I - A)$ generates $\{u^{\mathsf{T}} A^i b\}_{i=0}^{\infty} \Longrightarrow$ only need $\{u^{\mathsf{T}} A^i b\}_{i=0}^{2n-1}$

Back to original problem

Solve the (nonsingular) linear system ($n = 4$)

$$\begin{bmatrix} 1 & 3 & 0 & 3 \\ 0 & 1 & 0 & 0 \\ 3 & 3 & -2 & 3 \\ 0 & -3 & 0 & -2 \end{bmatrix} x = \begin{bmatrix} 4 \\ 2 \\ 6 \\ -4 \end{bmatrix}$$

1: compute Krylov sequence $\{A^i b\}_{i=0}^{2n-1}$

$$\left\{ \begin{bmatrix} 4 \\ 2 \\ 6 \\ -4 \end{bmatrix}, \begin{bmatrix} -2 \\ 2 \\ -6 \\ 2 \end{bmatrix}, \begin{bmatrix} 10 \\ 2 \\ 18 \\ -10 \end{bmatrix}, \begin{bmatrix} -14 \\ 2 \\ -30 \\ 14 \end{bmatrix}, \begin{bmatrix} 34 \\ 2 \\ 66 \\ -34 \end{bmatrix}, \begin{bmatrix} -62 \\ 2 \\ -126 \\ 62 \end{bmatrix}, \begin{bmatrix} 130 \\ 2 \\ 258 \\ -130 \end{bmatrix}, \begin{bmatrix} -254 \\ 2 \\ -510 \\ 254 \end{bmatrix} \right\}$$

2: choose a random vector

$$u = \begin{bmatrix} 2 \\ 1 \\ 1 \\ 2 \end{bmatrix}$$

3: compute first $2n$ entries of scalar sequence $\{u^{\mathsf{T}} A^i b\}_{i=0}^{2n-1}$

$$\{8, -4, 20, -28, 68, -124, 260, -508\}$$

4: use Berlekamp-Massey to compute $f^{A,b} = c_0 + c_1\lambda + \cdots + c_m\lambda^m$

$$f^{A,b} = f_u^{A,b} = \lambda^2 + \lambda - 2$$

5: solution

$$x = \frac{1}{2}(Ab + b) = \frac{1}{2}\left(\begin{bmatrix} -2 \\ 2 \\ -6 \\ 2 \end{bmatrix} + \begin{bmatrix} 4 \\ 2 \\ 6 \\ -4 \end{bmatrix}\right) = \begin{bmatrix} 1 \\ 2 \\ 0 \\ -1 \end{bmatrix}$$

Computing Matrix Determinant

Given $A \in \mathbb{F}^{n \times n}$

For u and v chosen randomly from finite $S^n \subset \mathbb{F}^n$,

$$f_u^{A,v} = f^{A,v} = f^A$$

If $f^A = \det(\lambda I - A)$,

$$\det(A) = (-1)^n f_u^{A,v}(0)$$

Want to precondition A so that $f^{\tilde{A}} = \det(\lambda I - \tilde{A})$

Precondition by pre- or post-multiplication $\tilde{A} = BA$ or $\tilde{A} = AB$

Need efficient matrix-vector product

Theorem (Turner, 2001). *Let $\mathbb{F}$ be a field, $A \in \mathbb{F}^{n\times n}$ be nonsingular, and S be a finite subset of $\mathbb{F}$. If*

$$U = \begin{bmatrix} 1 & s_1 & & \\ & \ddots & \ddots & \\ & & 1 & s_{n-1} \\ & & & 1 \end{bmatrix}$$

where $s_1, \ldots, s_{n-1}$ are chosen uniformly and independently from S, then AU is nonsingular and

$$f^{AU} = \det(\lambda I - AU)$$

with probability at least

$$1 - \frac{n(n-1)}{2\,|S|}$$

Can also use AU^{T}, UA, and $U^{\mathsf{T}}A$

Computing Matrix Rank

Lemma (Kaltofen & Saunders 1991). *Let $A \in \mathbb{F}^{n \times n}$ have leading principal minors nonzero up to A_r where r is the (unknown) rank of A, and suppose that $r < n$. Let $S \subset \mathbb{F}$ and let $D = \mathsf{diag}(d_1, \ldots, d_n)$, d_i chosen uniformly from S. Then,*

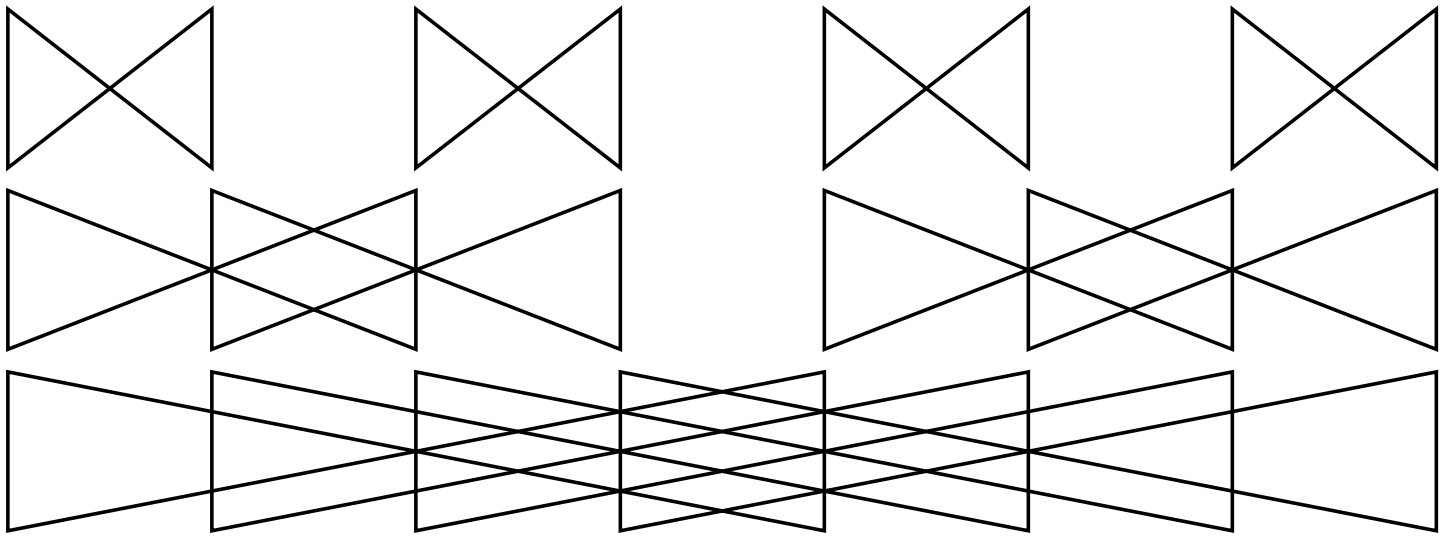
$$r = \mathsf{deg}(f^{AD}) - 1$$

with probability at least

$$1 - \frac{n(n-1)}{2\,|S|}$$

Want to precondition A so that leading principal minors are nonzero up to rank

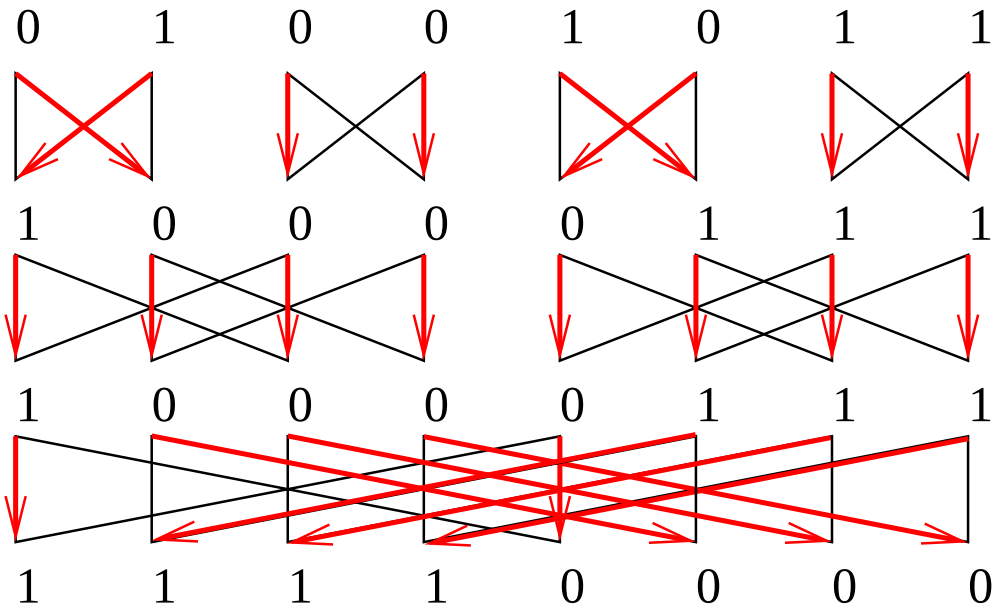
In our paper (Chen *et al.*, LAA 2002) we use preconditioner based on butterfly network:



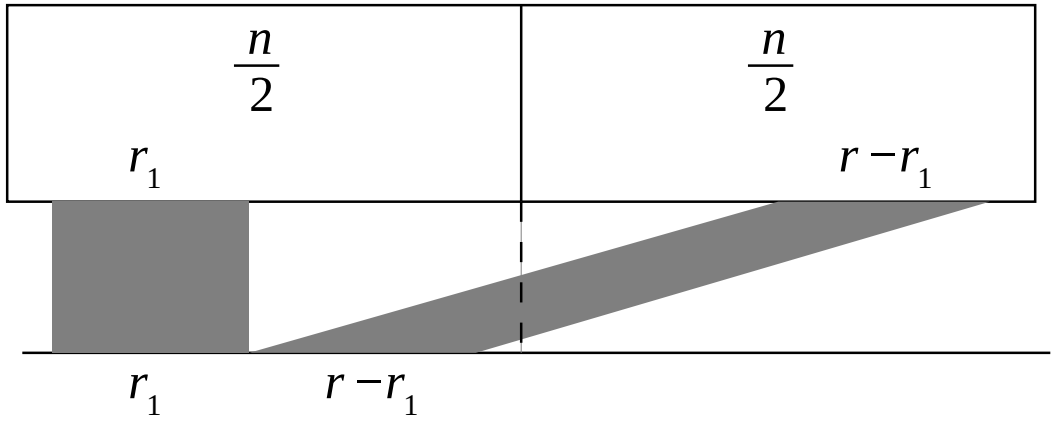
Lemma. *Let $n = 2^l$. The l-dimensional butterfly network can switch any r indices*

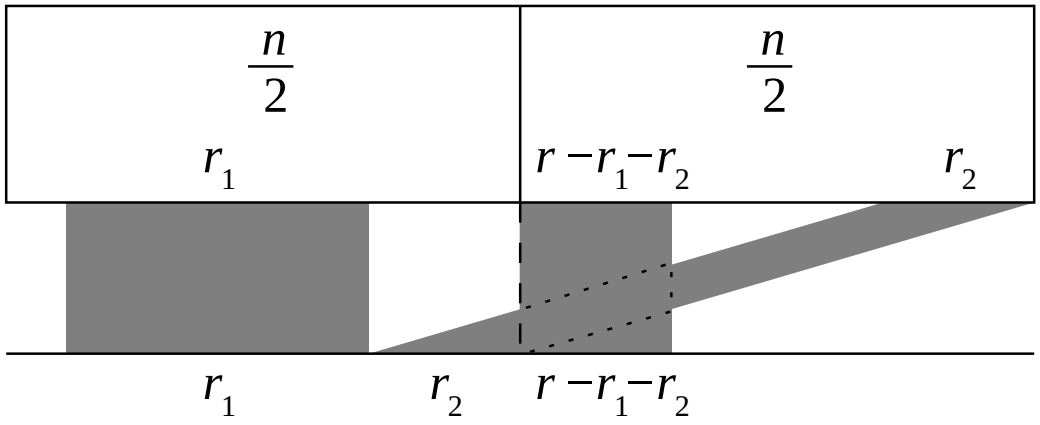
$$1 \leq i_1 < \cdots < i_r \leq n$$

into any desired contiguous block of indices; wrap around outside, for our purposes, shall preserve contiguity. Furthermore, the network contains a total of $\frac{1}{2}\, n \log_2(n)$ switches.



Idea behind proof:





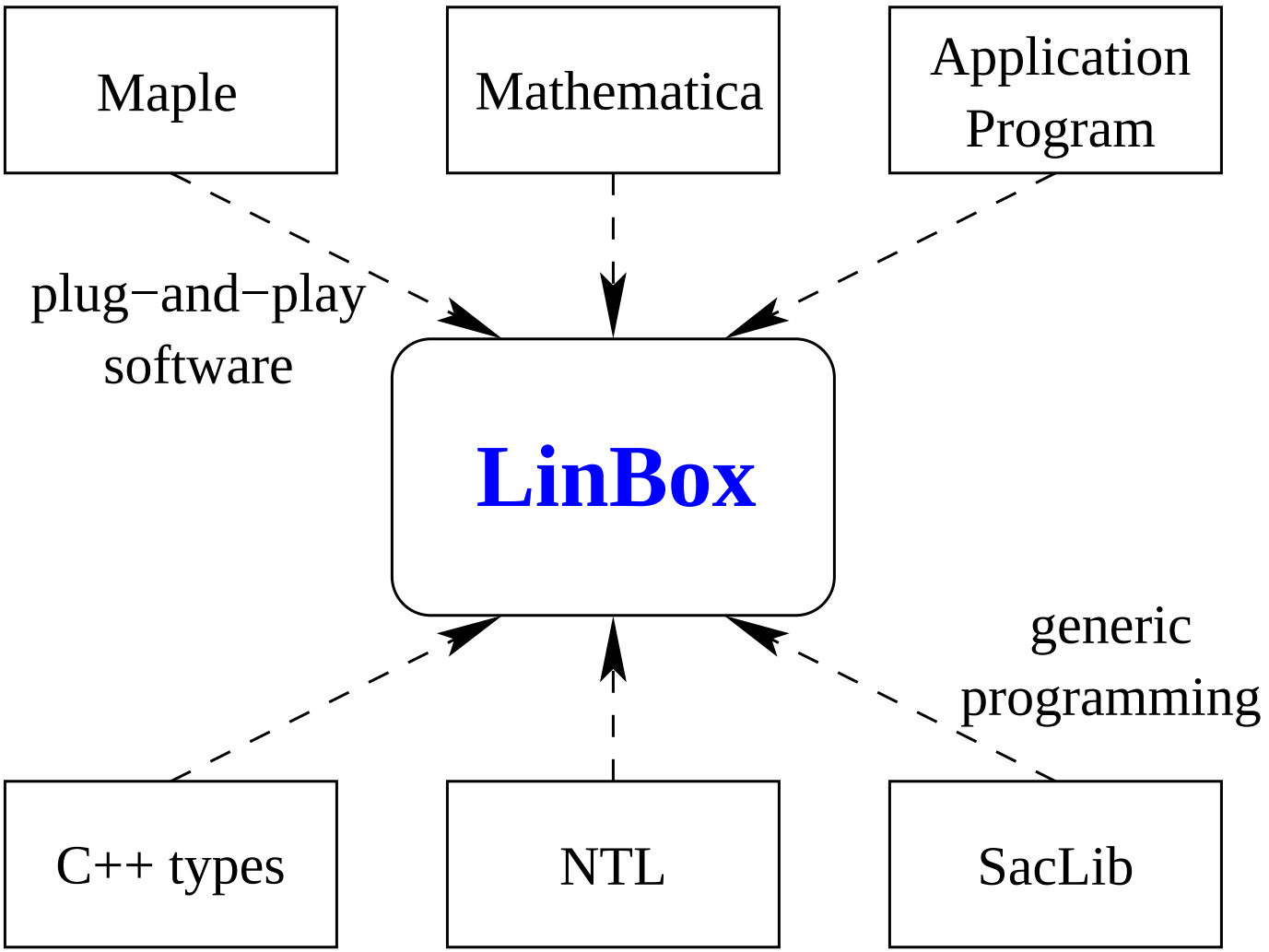
Project LinBox

Approximately 20 researchers from Canada, France, and U.S.

Algorithms and software for symbolic linear algebra, particularly black box matrix methods

Generic or reusable programming through C++ templates

“Middle-ware”


Maple
Mathematica
Application Program
plug−and−play software
LinBox
generic programming
C++ types
NTL
SacLib

Field Design

Goal: Pluggable type for elements in abstract field

Difficulties:

- Common object interface
- Parameterized and unparameterized fields
- Using C++ built-in types, like `double` and `long`

Solution: (Dumas *et al.*, 2002)

- Two types: field and element
- No distinction between parameterized and unparameterized fields
- Field object contains methods
- Template wrapper for unparameterized fields

Field Example

Unparametric field from C++ type:

```
unparam_field<double> F;
unparam_field<double>::element a, b, c;
F.init(a,0);     // a = 0
F.init(b,23);    // b = 23
F.init(c,-5);    // c = -5
F.add(a,b,c);    // a = b + c = 18
```

Parametric field (integers modulo 13):

```
param_modular F(13);
param_modular::element a, b, c;
F.init(a,0);     // a = 0 mod 13 = 0
F.init(b,23);    // b = 23 mod 13 = 10
F.init(c,-5);    // c = -5 mod 13 = 8
F.add(a,b,c);    // a = b + c mod 13 = 5
```

Conclusion

Black box linear algebra is

Theoretically: efficient

Practically: implementable through generic programming

References

E. R. BERLEKAMP (1968). *Algebraic Coding Theory*. McGraw-Hill, New York.

L. CHEN, W. EBERLY, E. KALTOFEN, B. D. SAUNDERS, W. J. TURNER, & G. VILLARD (2002). Efficient Matrix Preconditioners for Black Box Linear Algebra. *Linear Algebra and Applications*, **343-344**: 119–146. Special issue on *Infinite Systems of Linear Equations Finitely Specified*, edited by P. Dewilde, V. Olshevsky and A. H. Sayed.

J.-G. DUMAS, T. GAUTIER, M. GIESBRECHT, P. GIORGI, B. HOVINEN, E. KALTOFEN, B. D. SAUNDERS, W. J. TURNER, & G. VILLARD (2002). LinBox: A Generic Library for Exact Linear Algebra. Paper submitted, 10 pages.

E. KALTOFEN & B. D. SAUNDERS (1991). On Wiedemann's Method of Solving Sparse Linear Systems. In H. F. MATTSON, T. MORA, & T. R. N. RAO (eds.), *Proc. AAECC-9*, vol. 539 of *Lect. Notes Comput. Sci.*, 29–38. Springer Verlag, Heidelberg, Germany.

J. L. MASSEY (1969). Shift-Register Synthesis and BCH Decoding. *IEEE Trans. Inf. Theory*, **IT-15**: 122–127.

W. J. TURNER (2001). A Note on Determinantal Divisors and Matrix Preconditioners. To be submitted.

D. H. WIEDEMANN (1986). Solving Sparse Linear Equations Over Finite Fields. *IEEE Trans. Inf. Theory*, **IT-32**: 54–62.