Algorithms for sparse and black box matrices over finite fields

Erich Kaltofen
North Carolina State University
www.kaltofen.net

Factorization of an integer N
(continued fraction, quadratic sieves, number field sieves)

Compute a solution to the congruence equation

$$X^2 \equiv Y^2 \pmod{N}$$

via r relations on b basis primes

$$X_1^2 \cdot X_2^2 \cdots X_r^2 \equiv (p_1^{e_1})^2 \cdot (p_2^{e_2})^2 \cdots (p_b^{e_b})^2 \pmod{N}$$

Then N divides $(X+Y)(X-Y)$, hence

$$\mathrm{GCD}(X+Y, N) \text{ divides } N$$

Relation computation

Step 1: Compute $s > r$ relations on b basis primes

$$\forall 1 \le i \le s \colon Y_i^2 \equiv p_1^{c_{i,1}} \cdot p_2^{c_{i,2}} \cdots p_b^{c_{i,b}} \pmod{N}$$

Step 2: select r relations $X_1 = Y_{i_1}, \ldots, X_r = Y_{i_r}$ such that

$$\forall 1 \le j \le b \colon c_{i_1,j} + c_{i_2,j} + \cdots + c_{i_r,j} \equiv 0 \pmod{2}$$

One must compute non-zero solutions to the
sparse homogeneous linear system modulo 2

$$\begin{bmatrix} x_1 & \ldots & x_s \end{bmatrix} \begin{bmatrix} c_{1,1} \bmod 2 & \ldots & c_{1,b} \bmod 2 \\ c_{2,1} \bmod 2 & \ldots & c_{2,b} \bmod 2 \\ \vdots & & \vdots \\ c_{2,1} \bmod 2 & \ldots & c_{2,b} \bmod 2 \end{bmatrix} \equiv \begin{bmatrix} 0 & \ldots & 0 \end{bmatrix} \pmod{2}$$

LDDMLtR's RSA-120 matrix modulo 2

Row nr.	Columns with non-zero entries
1	0 1 481 1355 3b42 5cf6 c461 eda1 f0e7 15d19 199e0 2c317 33a5
2	0 1 9b4 f26 3214 7f99 a146 bc7e 10087 175c5 1953a 320b5 394
⋮	⋮
245 811	0 1 2 3 4 6 8 9 b c d f 10 12 13 14 16 17 18 19 1d 1e 1f 20 25 2
	. . . 3624a 36473 36905 37727 395

There are $10 - 217$ non-zero entries/column, with 252 222 columns and 11 037 745 non-zero entries total; in the above format the matrix occupies 48 Mbytes of disc space.

challenge-rsa-honor-roll@rsa.com

RSA-155

Factors:

1026395928297411057720541965739916759007165678080380668033419335217907113

*

1066034883801684548209272203600128786792079585759892915222706082371930628

Date: August 22, 1999

Method: the General Number Field Sieve,
with a polynomial selection method of Brian Murphy
and Peter L. Montgomery,
with lattice sieving (71%) and with line sieving (29%),
and with Peter L. Montgomery's blocked Lanczos and
square root algorithms;

Time: * Polynomial selection:
The polynomial selection took approximately 100 MIPS years,
equivalent to 0.40 CPU years on a 250 MHz processor.
...
* Sieving: 35.7 CPU-years in total,
...
124 722 179 relations were collected by eleven different sites
...
* Filtering the data and building the matrix took about a month

* Matrix: 224 hours on one CPU of the Cray-C916 at SARA, Amsterda
  the matrix had 6 699 191 rows and 6 711 336 columns,
  and weight 417 132 631 (62.27 nonzeros per row);
  calendar time: ten days
* Square root: Four jobs assigned one dependency each were run
  in parallel on separate 300 MHz R12000 processors
  within a 24-processor SGI Origin 2000 at CWI.
  One job found the factorisation after 39.4 CPU-ho

...

* The total calendar time for factoring RSA-155 was 5.2 months
  (March 17 - August 22)
  (excluding polynomial generation time)
  We could reduce this to one month sieving time and
  one month processing time if we had more sievers and
  had more experience with matrix-generation strategies.

Address: (of contact person)
Email: Herman.te.Riele@cwi.nl

Factorization of polynomial f over finite field $\mathbb{F}_p$ (Berlekamp 1967 algorithm)

Note that since $a^p \equiv a \pmod{p}$ for all $a \in \mathbb{F}_p$ we have

$$x^p - x \equiv x \cdot (x-1) \cdot (x-2) \cdots (x-p+1) \pmod{p}$$

Compute a polynomial solution to the congruence equation

$$w(x)^p \equiv w(x) \pmod{f(x)}$$

Then f divides $w \cdot (w-1) \cdot (w-2) \cdots (w-p+1)$, hence

$\mathrm{GCD}(w(x)-a, f(x))$ divides $f(x)$ for some $a \in \mathbb{F}_p$

Solving $w^p \equiv w \pmod{f}$ by linear algebra

For $w(x) \in \mathbb{F}_p[x]$, $\deg(w) < n = \deg(f)$:

$$w(x)^p = w(x^p) \equiv w(x) \pmod{f(x)}$$

$$\Updownarrow$$

$$\overrightarrow{w(x^p) \bmod f(x)} = \underbrace{[w_0 \ldots w_{n-1}]}_{\vec{w}} \cdot \underbrace{\begin{bmatrix} \vdots \\ \overrightarrow{x^{ip} \bmod f(x)} \\ \vdots \end{bmatrix}_{0 \le i < n}}_{\text{Q}} = \vec{w}$$

(Petr's 1937 matrix)

Black box matrix concept

$A \in \mathbb{F}^{n \times n}$ singular
$\mathbb{F}$ an arbitrary, e.g., finite field

Perform linear algebra operations, e.g., $A^{-1}b$ [Wiedemann 86] with

$O(n)$	black box calls and
$n^2(\log n)^{O(1)}$	arithmetic operations in $\mathbb{F}$ and
$O(n)$	intermediate storage for field elements

Black box model is useful for dense, structured matrices

$$\begin{bmatrix} 1 & \cdots & & \cdots & \frac{1}{n} \\ & & \vdots & & \\ & & \frac{1}{i+j-1} & & \\ & & \vdots & & \\ \frac{1}{n} & \cdots & & \cdots & \frac{1}{2n-1} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}$$

(Hilbert matrix)

Savings is in space, not time: $O(1)$ vs. $O(n^2)$.

Idea for Wiedemann's algorithm

$A \in \mathbb{F}^{n\times n}$, $\mathbb{F}$ a (possibly finite) field

$\phi^A(\lambda) = c'_0 + \cdots + c'_m \lambda^m \in \mathbb{F}[\lambda]$ **minimum polynomial** of A

$\forall\, u, v \in \mathbb{F}^n\colon\ \forall\, j \geq 0\colon$

$$u^{Tr} A^j \phi^A(A) v = 0$$

$$\Updownarrow$$

$$c'_0 \cdot \underbrace{u^{Tr} A^j v}_{a_j} + c'_1 \cdot \underbrace{u^{Tr} A^{j+1} v}_{a_{j+1}} + \cdots + c'_m \cdot \underbrace{u^{Tr} A^{j+m} v}_{a_{j+m}} = 0$$

$$\Updownarrow$$

$\{a_0, a_1, a_2, \ldots\}$ is generated by a linear recursion

Theorem [Wiedemann 1986]: *For **random*** $u, v \in \mathbb{F}^n$, *a linear generator for* $\{a_0, a_1, a_2, \ldots\}$ *is one for* $\{I, A, A^2, \ldots\}$.

$$\forall\, j \geq 0:\;\; c_0 a_j + c_1 a_{j+1} + \cdots + c_d a_{j+d} = 0$$

$$\Big\Downarrow \;\text{(with high probability)}$$

$$c_0 A^j v + c_1 A^{j+1} v + \cdots + c_d A^{j+d} v = \mathbf{0}$$

$$\Big\Downarrow \;\text{(with high probability)}$$

$$c_0 A^j + c_1 A^{j+1} + \cdots + c_d A^{j+d} = \mathbf{0}$$

that is, with high probability $\phi^A(\lambda)$ divides $c_0 + c_1\lambda + \cdots + c_d\lambda^d$

Algorithm homogeneous Wiedemann

Input: $A \in \mathbb{F}^{n \times n}$ singular
Output: $w \neq \mathbf{0}$ such that $Aw = \mathbf{0}$

Step W1: Pick random $u, v \in \mathbb{F}^n$; $\quad b \leftarrow Av$;
for $i \leftarrow 0$ **to** $2n-1$ **do** $a_i \leftarrow u^{Tr} A^i b$.
(Requires $2n$ black box calls.)

Step W2: Compute a linear recurrence generator for $\{a_i\}$,
$c_\ell \lambda^\ell + c_{\ell+1}\lambda^{\ell+1} + \cdots + c_d \lambda^d, \quad \ell \geq 0, d \leq n, c_\ell \neq 0.$

Step W3: $\widehat{w} \leftarrow c_\ell v + c_{\ell+1} Av + \cdots + c_d A^{d-\ell} v$;
(With high probability $\widehat{w} \neq 0$ and $A^{\ell+1}\widehat{w} = 0$.)
Compute first k with $A^k \widehat{w} = 0$; **return** $w \leftarrow A^{k-1}\widehat{w}$.
(Requires $\leq n$ black box calls.)

Step W2 detail

Coefficients $c_0, \ldots, c_n$ can be found by computing a non-trivial solution to the Toeplitz system

$$\begin{bmatrix} a_n & a_{n-1} & \cdots & & a_1 & a_0 \\ a_{n+1} & a_n & & & a_2 & a_1 \\ \vdots & a_{n+1} & \ddots & & \vdots & a_2 \\ & \vdots & & & & \vdots \\ a_{2n-2} & & & \ddots & a_{n-1} & \\ a_{2n-1} & a_{2n-2} & \cdots & & a_n & a_{n-1} \end{bmatrix} \cdot \begin{bmatrix} c_n \\ c_{n-1} \\ c_{n-2} \\ \vdots \\ \\ c_0 \end{bmatrix} = \mathbf{0}$$

or by the Berlekamp/Massey algorithm.

Cost: $O(n(\log n)^2 \operatorname{loglog} n)$ arithmetic ops.

Flurry of recent results

Lambert [96], Teitelbaum [98], Eberly & Kaltofen [97]	relationship of Wiedemann and Lanczos approach
Villard [97]	analysis of **block** Wiedemann algorithm
Giesbrecht [97] and Mulders & Storjohann [99]	computation of **diophantine** solutions
Giesbrecht, Lobo & Saunders [98]	certificates for inconsistency
Chen, Eberly, Kaltofen, Saunders, Villard & Turner [2K]	butterfly network, sparse and diagonal preconditioners
Villard [2K] & Storjohann [01]	characteristic polynomial
Kaltofen & Villard [2K]	fast algorithm for determinant of a **dense** integer matrix

LinSolve0: *Given blackbox* A, *compute* $w \neq 0$ *such that* $Aw = 0$.

NonSingular$\leq$LinSolve0: For $Ax = b$ solve $\left[A \mid b\right] w = 0$ and compute $x = \frac{1}{w_{n+1}} \begin{bmatrix} w_1 \\ \dots \\ w_n \end{bmatrix}$.

Harder (?) problem
LinSolve1: *Given blackbox* A *(possibly singular) and* b, *compute* x *such that* $Ax = b$.

Random sampling in the nullspace is equivalent to LinSolve1: select a random vector y and solve $Ax = b$ for $b = Ay$.

LinSolve1 via preconditioning

Suppose the minpoly of A is $1 \cdot \lambda + \cdots + c_m \lambda^m$
(the canonical form of A has "no nil-potent blocks.")

$$\begin{aligned} \text{If} \quad & Ax = b \text{ is consistent then } b = A \cdot y \\ \text{hence} \quad & 1 \cdot b + \cdots + c_m A^{m-1} b = 1 \cdot Ay + \cdots + c_m A^m y = 0 \\ \text{so} \quad & b = A \cdot (\underbrace{-c_2 b - \cdots - c_m A^{m-2} b}_{x}). \end{aligned}$$

In [Chen *et al.* 2000] it is shown that Wiedemann's random sparse matrix multipliers give $\widetilde{A}$ the above property:

$\widetilde{A} = LAR$ where L, R are certain sparse 0-1 matrices

Note: L, R have $O(n(\log n)^2)$ non-zero entries.

Diophantine solutions by Giesbrecht: Find several rational solutions.

$$A(\tfrac{1}{2}x^{[1]}) = b, \quad x^{[1]} \in \mathbb{Z}^n$$
$$A(\tfrac{1}{3}x^{[2]}) = b, \quad x^{[2]} \in \mathbb{Z}^n$$

$$\gcd(2,3) = 1 = 2 \cdot 2 - 1 \cdot 3$$
$$A(2x^{[1]} - x^{[2]}) = 4b - 3b = b$$

Hensel lifting [Moenck and Carter 1979, Dixon 1982]:

1: For $j = 0, 1, \ldots, k$ and a prime p Do

Compute $\bar{x}^{[j]} = x^{[0]} + px^{[1]} + \cdots + p^j x^{[j]} \equiv x \pmod{p^{j+1}}$

1.a. $b^{[j]} = \dfrac{b - A\bar{x}^{[j-1]}}{p^j} = \dfrac{b^{[j-1]} - Ax^{[j-1]}}{p}$

1.b. Solve $Ax^{[j]} \equiv b^{[j]} \pmod{p}$ reusing the minpoly of $A \bmod p$

2: Recover denominators of x_i by continued fractions of $\bar{x}_i^{[k]}/p^k$.

Original idea for Lanczos's algorithm

Assumption: $A \in \mathbb{F}^{n\times n}$ is non-singular and symmetric. Then

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

is a pseudo-inner product, i.e., $\langle v, v\rangle_A \neq 0$ for $v \neq 0$ is not guaranteed unless A is positive definite.

Orthogonalize w.r.t. $\langle \cdot, \cdot \rangle_A$ the Krylov space $b,\ Ab,\ A^2b, \ldots$

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

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

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

Lambert's 1996 interpretation

Lanczos = Wiedemann with projections $u = v = b$
+ evaluating polynomials at A, b
as they are updated in Berlekamp/Massey.

Since Berlekamp/Massey = Euclid [Dornstetter] we may state Lanczos implicitly performs Euclid [Gutknecht].

If polynomial remainders drop in degree, one needs to perform "lookahead." (May not be space efficient.)

If the degree of the minpoly is low, Lanczos "terminates early."

Analysis for Wiedemann can be applied to Lanczos.

Probabilistic analysis by Eberly and Kaltofen 1996

Precondition (nonsymmetric) A and b as

$$\widetilde{A} \leftarrow G_1 \cdot A^{Tr} \cdot G_2 \cdot A \cdot G_1 \quad \text{where} \quad \begin{array}{l} G_1 \text{ is random diagonal} \\ G_2 \text{ is random diagonal} \end{array}$$

$$\widetilde{b} \leftarrow G_1 \cdot A^{Tr} \cdot G_2 \cdot b + \widetilde{A} \cdot y \quad \text{where} \quad y \text{ is a random vector}$$

Then for $\widetilde{A}$, $\widetilde{b}$ and all $w_{i+1} \neq 0$ we have $\langle w_{i+1}, w_{i+1} \rangle_{\widetilde{A}} \neq 0$ and a solution is found with probability $\geq 1 - \frac{11n^2 - n}{2|\mathbb{F}|}$

Computational cost: n black box and n transpose black box calls.

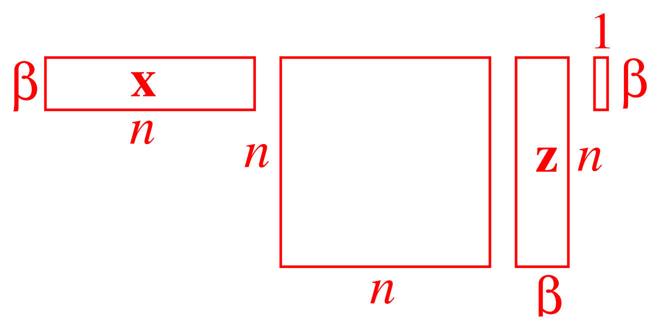
Coppersmith's 1992 blocking

Use of the block vectors $\mathbf{x} \in \mathbb{F}^{n\times\beta}$ in place of u
$\mathbf{z} \in \mathbb{F}^{n\times\beta}$ in place of v

$$\mathbf{a}_i = \mathbf{x}^{Tr} A^{i+1} \mathbf{z} \in \mathbb{F}^{\beta\times\beta}, \quad 0 \le i < 2n/\beta + 2.$$

Find a **vector** polynomial $c_\ell\lambda^\ell + \cdots + c_d\lambda^d \in \mathbb{F}^\beta[\lambda], d = \lceil n/\beta \rceil$:

$$\forall\, j \ge 0\colon \ \sum_{i=\ell}^{d} \mathbf{a}_{j+i} c_i = \sum_{i=\ell}^{d} \mathbf{x}^{Tr} A^{i+j}\, A\mathbf{z} c_i = \mathbf{0} \in \mathbb{F}^{\beta\times\beta}$$

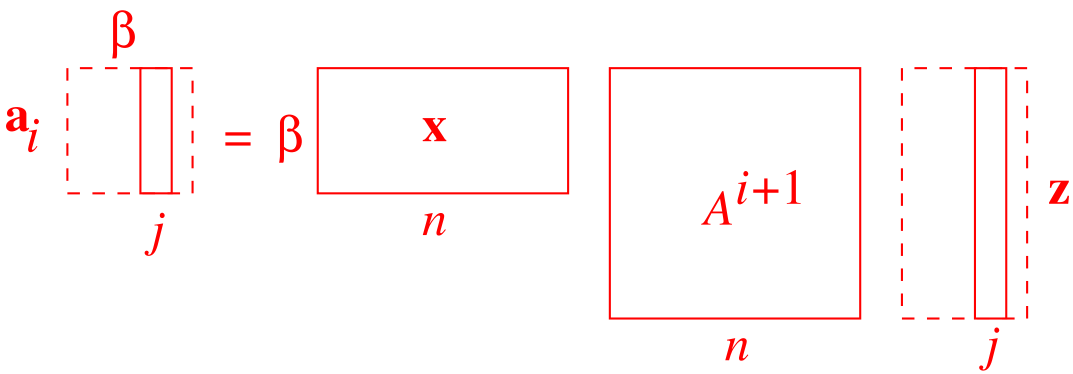


Then, analogously to before, with high probability

$$\widehat{w} = \sum_{i=\ell}^{d} A^{i-\ell}\, \mathbf{z} c_i \neq \mathbf{0}, \quad A^{\ell+1}\widehat{w} = \sum_{i=\ell}^{d} A^i A\mathbf{z} c_i = \mathbf{0} \in \mathbb{F}^n$$

Advantages of blocking

1. Parallel coarse- and fine-grain implementation



The j^{th} processor computes the j^{th} column of the sequence of (small) matrices.

2. Faster sequential running time:
   multiple solutions [Coppersmith; Montgomery 1994];
   $1 + \epsilon$ matrix times vector ops [Kaltofen 1995];
   determinant [Kaltofen & Villard 2000];
   charpoly of sparse matrix [Villard & Storjohann 2001]

3. Better probability of success [Villard 1997]

Analysis of blocking

All vector polynomials that generate $\{\mathbf{a}_i\}$ form a **module** over $\mathbb{F}[\lambda]$.

β vectors of minimal degree determinant from a $\mathbb{F}[\lambda]$-basis

A matrix canonical (Popov, Hermite) version of the basis defines a unique minimal polynomial $\mathbf{c}_0 + \mathbf{c}_1\lambda + \cdots + \mathbf{c}_d\lambda^d \in \mathbb{F}^{\beta\times\beta}[\lambda]$

From $(I - \lambda B)^{-1} = I + B\lambda + B^2\lambda^2 + \cdots$

$$\mathbf{x}^{Tr}(I - \lambda B)^{-1}\mathbf{y}(\mathbf{c}_d + \cdots + \mathbf{c}_0\lambda^d) = R(\lambda) \in \mathbb{F}[\lambda]^{\beta\times\beta}$$

we obtain a matrix Padé approximation (“realization”)

$$\mathbf{x}^{Tr}(I - \lambda B)^{-1}\mathbf{y} = \sum_i \mathbf{a}_i\lambda^i = R(\lambda)(\mathbf{c}_d + \cdots + \mathbf{c}_0\lambda^d)^{-1}$$

Computation of the matrix linear generator

Explicitly in Popov form by block Berlekamp/Massey algorithm [Coppersmith 1994, Thomé 2001] or implicitly in a block Lanczos version

Explicitly by a power Hermite-Padé approximation
[Beckermann & Labahn 1994]

By a block Toeplitz solver [Kaltofen 1995]

Implementations

By Coppersmith, Kaltofen & Lobo, Montgomery, Dumas, Brent,...

The LinBox group [Canada: UWO, Calgary; France: ENS Lyon, IMAG Grenoble; USA: Delaware, NCSU, Washington Coll. MD]: A generic C++ library for black box linear algebra, including integer problems

Open Problems

Large fields: Compute the characteristic polynomial
Certify the minimal polynomial
$\textsc{LinSolve}1 \le \textsc{LinSolve}0$

Small fields: Compute the determinant, rank of a sparse/blackbox matrix without $O(\log n)$ slowdown