Simplified high-speed high-distance list decoding for alternant codes

cr.yp.to/papers.html
#simplelist

D. J. Bernstein
University of Illinois at Chicago

Thanks to: Cisco
University Research Program

And thanks to: NIST
grant 60NANB10D263

Context: McEliece key size

Standard asymptotics:

For 2^b security, McEliece needs $(C_0 + o(1))b^2(\lg b)^2$-bit keys.
Here $C_0 \approx 0.7418860694$.

Standard asymptotics + sensible Grover (PQCrypto 2010 Bernstein "Grover vs. McEliece"):

For 2^b post-quantum security, McEliece needs $(4C_0 + o(1))b^2(\lg b)^2$-bit keys.
Same C_0 as before.

One definition of C_0:
$R = 1 - \exp(-2R)$ is satisfied
for a unique $R \approx 0.7968121300$;
then $C_0 = (\log 2)^2 / 4R(1 - R)$.

One definition of C_0:

$R = 1 - \exp(-2R)$ is satisfied for a unique $R \approx 0.7968121300$; then $C_0 = (\log 2)^2 / 4R(1 - R)$.

i.e. R with $0 < R < 1$ minimizes $C_0 = R/(1 - R)(\lg(1 - R))^2$.

One definition of C_0:

$R = 1 - \exp(-2R)$ is satisfied
for a unique $R \approx 0.7968121300$;
then $C_0 = (\log 2)^2 / 4R(1 - R)$.

i.e. R with $0 < R < 1$ minimizes
$C_0 = R/(1 - R)(\lg(1 - R))^2$.

Warning:
$o(1)$ does not mean 0.
It means something
that *converges* to 0 as $b \to \infty$.
Closer look: this $o(1)$ is positive,
so replacing $o(1)$ by 0
would *not* achieve 2^b security.

Where does this come from?

Where does this come from?

McEliece public key
(with Niederreiter compression)
is a $k \times (n-k)$ matrix over $\mathbf{F}_2$.
$R(1-R)n^2$ bits if $k = Rn$.

Where does this come from?

McEliece public key
(with Niederreiter compression)
is a $k \times (n-k)$ matrix over $\mathbf{F}_2$.
$R(1-R)n^2$ bits if $k = Rn$.

Best attacks known:
$c^{(1+o(1))n/\lg n}$ simple operations
where $c = 1/(1-R)^{1-R}$.

Where does this come from?

McEliece public key
(with Niederreiter compression)
is a $k \times (n-k)$ matrix over $\mathbf{F}_2$.
$R(1-R)n^2$ bits if $k = Rn$.

Best attacks known:
$c^{(1+o(1))n/\lg n}$ simple operations
where $c = 1/(1-R)^{1-R}$.

For $c^{(1+o(1))n/\lg n} \ge 2^b$
choose n as $(b \lg b)/(\lg c + o(1))$.
$R(1-R)b^2(\lg b)^2/(\lg c + o(1))^2 =$
$(C_0 + o(1))b^2(\lg b)^2$ key bits where
$C_0 = R/(1-R)(\lg(1-R))^2$.

$c^{(1+o(1))n/\lg n}$? $c = 1/(1-R)^{1-R}$?

$c^{(1+o(1))n/\lg n}$? $c = 1/(1-R)^{1-R}$?

The public key represents
a k-dimensional subspace of $\mathbf{F}_2^n$.
Secretly equivalent to
a classical binary Goppa code
efficiently correcting w errors.
Tradition: $n = 2^m$, $k = n - mw$;
so $w = (1 - R)n/\lg n$.

$c^{(1+o(1))n/\lg n}$? $c = 1/(1-R)^{1-R}$?

The public key represents
a k-dimensional subspace of $\mathbf{F}_2^n$.
Secretly equivalent to
a classical binary Goppa code
efficiently correcting w errors.
Tradition: $n = 2^m$, $k = n - mw$;
so $w = (1 - R)n/\lg n$.

Information-set decoding
guesses k error-free positions.
Chance $\approx \binom{n-w}{k} / \binom{n}{k}$;
$(1 - R + o(1))^w$ since $w/n \to 0$.
More precise: 2009 Bernstein–
Lange–Peters–van Tilborg.

Smaller keys via list decoding

Proposal from PQCrypto 2008 Bernstein–Lange–Peters: reduce key size by "using list decoding to increase w."

List decoding efficiently corrects *more* than $(1 - R)n/\lg n$ errors in the same secret Goppa code. Larger $w \Rightarrow$ harder attacks $\Rightarrow$ smaller keys for 2^b security.

Smaller keys via list decoding

Proposal from PQCrypto 2008 Bernstein–Lange–Peters: reduce key size by "using list decoding to increase w."

List decoding efficiently corrects *more* than $(1 - R)n/\lg n$ errors in the same secret Goppa code. Larger $w \Rightarrow$ harder attacks $\Rightarrow$ smaller keys for 2^b security.

Literature also has many ideas for reducing key size by *changing* the code: see talks by Misoczki, Peters.

Fix distinct $a_1, \ldots, a_n \in \mathbf{F}_{2^m}$
and monic $g \in \mathbf{F}_{2^m}[x]$ with
$\deg g = t$ and $g(a_1) \cdots g(a_n) \neq 0$.

The Goppa code $\Gamma \subseteq \mathbf{F}_2^n$
is the set of $(c_1, \ldots, c_n)$ with
$\sum_i c_i/(x - a_i) = 0$ in $\mathbf{F}_{2^m}[x]/g$.
Typically $\#\Gamma = 2^{n-mt}$.

Define $P = (x - a_1) \cdots (x - a_n)$.
Can write any $(c_1, \ldots, c_n) \in \Gamma$ as

$$\left(\frac{f(a_1)g(a_1)}{P'(a_1)}, \ldots, \frac{f(a_n)g(a_n)}{P'(a_n)} \right)$$

for some $f \in \mathbf{F}_{2^m}[x]$
with $\deg f < n - t$.

Classic Reed–Solomon decoding:
For any $w \le \lfloor t/2 \rfloor$,
correct w errors
in $(f(a_1), \ldots, f(a_n))$
assuming $\deg f < n - t$.

1960 Peterson:
$n^{O(1)}$ arithmetic ops.

1968 Berlekamp: $O(n^2)$.
Modern view: Reduce
a 2-dimensional lattice basis.

1976 Justesen,
independently 1977 Sarwate:
$n(\lg n)^{2+o(1)}$. Modern view:
fast lattice-basis reduction.

Improvements (combinable!) in number of correctable errors:

1. 1998 Guruswami–Sudan: Increase w from $\lfloor t/2 \rfloor$ up to $n - \sqrt{n(n-t-1)} \approx t/2 + t^2/8n$.

Improvements (combinable!) in number of correctable errors:

1. 1998 Guruswami–Sudan: Increase w from $\lfloor t/2 \rfloor$ up to $n - \sqrt{n(n-t-1)} \approx t/2 + t^2/8n$.

2. 2000 Koetter–Vardy: Exploit: error vector $\in \{0,1\}^n$. Replaces n by $n' = n/2$: $n' - \sqrt{n'(n'-t-1)} \approx t/2 + t^2/4n$.

Improvements (combinable!) in number of correctable errors:

1. 1998 Guruswami–Sudan: Increase w from $\lfloor t/2 \rfloor$ up to $n - \sqrt{n(n-t-1)} \approx t/2 + t^2/8n$.

2. 2000 Koetter–Vardy: Exploit: error vector $\in \{0,1\}^n$. Replaces n by $n' = n/2$: $n' - \sqrt{n'(n'-t-1)} \approx t/2 + t^2/4n$.

3. 1970 Goppa?: $\Gamma_2(g) = \Gamma_2(g^2)$ if g is squarefree. Combine with, e.g., Peterson to reach $w = t$.

Improvements (combinable!)
in number of correctable errors:

1. 1998 Guruswami–Sudan:
Increase w from $\lfloor t/2 \rfloor$ up to
$n - \sqrt{n(n-t-1)} \approx t/2 + t^2/8n$.

2. 2000 Koetter–Vardy:
Exploit: error vector $\in \{0,1\}^n$.
Replaces n by $n' = n/2$:
$n' - \sqrt{n'(n'-t-1)} \approx t/2 + t^2/4n$.

3. 1970 Goppa?: $\Gamma_2(g) = \Gamma_2(g^2)$
if g is squarefree. Combine with,
e.g., Peterson to reach $w = t$.

4. Guess a few error positions.

Recall asymptotic analysis:
Each extra error
makes attacks more difficult
by a factor $1/(1-R+o(1))$.
Have $1/(1-R) \approx 4.92+o(1)$.

Combining all known
poly-time improvements:
$\approx t^2/n \approx (1-R)^2 n/(\lg n)^2$
extra errors.

Multiplies security level b
by $\approx 1+(1-R)/\lg b$.

For same security, divides key size
by $\approx (1+(1-R)/\lg b)^2$.
$1+o(1)$ but still noticeable.

Streamlining list decoding

"Multiplicity 2" example of GS:

Input vector $(v_1, \ldots, v_n)$.

Find small nonzero $Q \in \mathbf{F}_{2^m}[x, y]$
having multiplicity ≥ 2
at each (a_i, v_i): i.e.,
$Q \in \langle x - a_i, y - v_i \rangle^2 =$
$\langle (x-a_i)^2, (x-a_i)(y-v_i), (y-v_i)^2 \rangle$.

Find all $f \in \mathbf{F}_{2^m}[x]$ with
$Q(f) = 0$ and $\deg f < n - t$.
Notation: $Q(f)$ is Q mod $y - f$.

Check whether $(v_1, \ldots, v_n)$
is close to $(f(a_1), \ldots, f(a_n))$.

"List size 3" definition of "small"

if $\frac{1}{4}(n-1), \frac{1}{2}(n-t-1) \in \mathbf{Z}$:

$Q = Q_0 + Q_1 y + Q_2 y^2 + Q_3 y^3$
for some $Q_0, Q_1, Q_2, Q_3 \in \mathbf{F}_{2^m}[x]$;
$\deg Q_0 \le \frac{3}{4}(n-1) + \frac{3}{2}(n-t-1)$;
$\deg Q_1 \le \frac{3}{4}(n-1) + \frac{1}{2}(n-t-1)$;
$\deg Q_2 \le \frac{3}{4}(n-1) - \frac{1}{2}(n-t-1)$;
$\deg Q_3 \le \frac{3}{4}(n-1) - \frac{3}{2}(n-t-1)$.

"List size 3" definition of "small"

if $\frac{1}{4}(n-1), \frac{1}{2}(n-t-1) \in \mathbf{Z}$:

$Q = Q_0 + Q_1 y + Q_2 y^2 + Q_3 y^3$

for some $Q_0, Q_1, Q_2, Q_3 \in \mathbf{F}_{2^m}[x]$;

$\deg Q_0 \le \frac{3}{4}(n-1) + \frac{3}{2}(n-t-1)$;

$\deg Q_1 \le \frac{3}{4}(n-1) + \frac{1}{2}(n-t-1)$;

$\deg Q_2 \le \frac{3}{4}(n-1) - \frac{1}{2}(n-t-1)$;

$\deg Q_3 \le \frac{3}{4}(n-1) - \frac{3}{2}(n-t-1)$.

$\deg Q(f) \le \frac{3}{4}(n-1) + \frac{3}{2}(n-t-1)$

but $Q(f)$ is divisible by D^2

where $D = \prod_{i:f(a_i)=v_i}(x - a_i)$.

Must have $Q(f) = 0$ if

$\deg D > \frac{3}{8}(n-1) + \frac{3}{4}(n-t-1)$.

Corrects $\frac{1}{2}t + \frac{1}{4}t - \frac{1}{8}n + \frac{9}{8}$ errors.

Have $3n + 1$ coeffs of Q.

$Q \in \langle x - a_i, y - v_i \rangle^2$
is 3 linear equations on coeffs:
e.g., $Q \in \langle x, y \rangle^2$
says coeffs of $1, x, y$ are 0.

Total $3n$ linear equations.

Linear algebra now
finds a small $Q \neq 0$.

Standard root-finding methods
find all f with $Q(f) = 0$;
use, e.g., 1969 Zassenhaus.

Eliminating localization:

Start with 0-error interpolation:
$R \in \mathbf{F}_{2^m}[x]$ has $R(a_i) = v_i$.
Compute $P = (x - a_1) \cdots (x - a_n)$.

$Q \in \langle x - a_i, y - v_i \rangle^2$ for all i iff
$Q_0 + Q_1 R + Q_2 R^2 + Q_3 R^3 \in \langle P^2 \rangle$
and $Q_1 + 2Q_2 R + 3Q_3 R^2 \in \langle P \rangle$.

Thus have a basis for dual
of $\mathbf{F}_{2^m}[x]$-lattice of $\{Q\}$.
Find small Q by basis reduction.

Eliminating localization:

Start with 0-error interpolation:
$R \in \mathbf{F}_{2^m}[x]$ has $R(a_i) = v_i$.
Compute $P = (x - a_1) \cdots (x - a_n)$.

$Q \in \langle x - a_i, y - v_i \rangle^2$ for all i iff
$Q_0 + Q_1 R + Q_2 R^2 + Q_3 R^3 \in \langle P^2 \rangle$
and $Q_1 + 2Q_2 R + 3Q_3 R^2 \in \langle P \rangle$.

Thus have a basis for dual
of $\mathbf{F}_{2^m}[x]$-lattice of $\{Q\}$.
Find small Q by basis reduction.

This algorithm is a
special case of 1996 Coppersmith,
later understood to supersede GS.

Eliminating the dual:

Simply write down a basis
$P^2, (y - R)P, (y - R)^2, y(y - R)^2$
for the same lattice.
Find Q by basis reduction.

Eliminating the dual:

Simply write down a basis
$P^2, (y - R)P, (y - R)^2, y(y - R)^2$
for the same lattice.
Find Q by basis reduction.

Special case of 1997
Howgrave-Graham improvement
to 1996 Coppersmith.

Very fast basis computation.
Fast basis reduction: use
2003 Giorgi–Jeannerod–Villard.

Cost $\ell^{<3.5} n (\lg n)^{O(1)}$
for general list size ℓ.

2006 Lee–O'Sullivan
rediscovered this construction
of a basis for this lattice,
but then found small Q
by Buchberger reduction
(finding a Gröbner basis).

Howgrave-Graham is better!
Buchberger reduction
is more general than
lattice-basis reduction
but is slower.

Lehmer, Knuth, et al.:
start reducing lattice basis
by reducing rounded basis.

What this paper does

Koetter and Vardy change
1998 GS lattice
(= 1996 Coppersmith lattice)
to correct more errors.

Seems outside scope of HG.
Can KV avoid localization?
Can KV avoid dualization?

What this paper does

Koetter and Vardy change
1998 GS lattice
(= 1996 Coppersmith lattice)
to correct more errors.

Seems outside scope of HG.
Can KV avoid localization?
Can KV avoid dualization?

Yes. 2011 Bernstein `simplelist`:
Streamlined construction of
basis for KV lattice, analogous
to 1997 HG construction of
basis for Coppersmith lattice.

More speedups

1975 Patterson:
Speed up Berlekamp for $\Gamma_2(g^2)$.

2007 Wu:
rational list decoding.
Same w as GS,
but much smaller multiplicity.

2008 Bernstein `goppalist`:
rational + HG + Patterson.

2011 Bernstein `jetlist`:
rational + HG + KV.
Same w as KV,
but much smaller multiplicity.