Short generators without quantum computers: the case of multiquadratics

Daniel J. Bernstein

University of Illinois at Chicago

31 July 2017

`https://multiquad.cr.yp.to`

Joint work with:
Jens Bauch & Henry de Valence & Tanja Lange & Christine van Vredendaal

Followup to my SIAM AG15 talk
"Computational algebraic number theory tackles lattice-based cryptography":
`https://cr.yp.to/talks.html#2015.08.06`

"*Lattice-based crypto is secure because lattice problems are hard.*"

— Everyone who works on lattice-based crypto

"*Lattice-based crypto is secure because lattice problems are hard.*"

— Everyone who works on lattice-based crypto

Really? How hard are they? Which problems are broken in time $<2^{100}$?
Which *cryptosystems* are broken in time $<2^{100}$?

"*Lattice-based crypto is secure because lattice problems are hard.*"

— Everyone who works on lattice-based crypto

Really? How hard are they? Which problems are broken in time $<2^{100}$? Which *cryptosystems* are broken in time $<2^{100}$?

2006 Silverman: "Lattices, SVP and CVP, have been intensively studied for more than 100 years, both as intrinsic mathematical problems and for applications in pure and applied mathematics, physics and cryptography."

"*Lattice-based crypto is secure because lattice problems are hard.*"

— Everyone who works on lattice-based crypto

Really? How hard are they? Which problems are broken in time $<2^{100}$? Which *cryptosystems* are broken in time $<2^{100}$?

2006 Silverman: "Lattices, SVP and CVP, have been intensively studied for more than 100 years, both as intrinsic mathematical problems and for applications in pure and applied mathematics, physics and cryptography."

2014 Peikert: "Because finding short vectors in high-dimensional lattices has been a notoriously hard algorithmic question for hundreds of years—even when one allows for the power of quantum algorithms (see, e.g., . . .)—we have solid and unique evidence that lattice-based cryptoschemes are secure."

"*Lattice-based crypto is secure because lattice problems are hard.*"

— Everyone who works on lattice-based crypto

Really? How hard are they? Which problems are broken in time $<2^{100}$? Which *cryptosystems* are broken in time $<2^{100}$?

2006 Silverman: "Lattices, SVP and CVP, have been intensively studied for more than 100 years, both as intrinsic mathematical problems and for applications in pure and applied mathematics, physics and cryptography."

2014 Peikert: "Because finding short vectors in high-dimensional lattices has been a notoriously hard algorithmic question for hundreds of years—even when one allows for the power of quantum algorithms (see, e.g., . . .)—we have solid and unique evidence that lattice-based cryptoschemes are secure."

Sounds like SVP is claimed to be a hard problem. How hard is it?

How secure is SVP?

Best SVP algorithms known at the end of the 20th century:
time $2^{\Theta(N \log N)}$ for almost all dimension-N lattices.

How secure is SVP?

Best SVP algorithms known at the end of the 20th century:
time $2^{\Theta(N \log N)}$ for almost all dimension-N lattices.

Best SVP algorithms known today: $2^{\Theta(N)}$, asymptotically much faster.
Some algorithms taking time $2^{(c+o(1))N}$, under plausible assumptions:

$c \approx 0.415$: 2008 Nguyen–Vidick.
$c \approx 0.415$: 2010 Micciancio–Voulgaris.

How secure is SVP?

Best SVP algorithms known at the end of the 20th century:
time $2^{\Theta(N \log N)}$ for almost all dimension-N lattices.

Best SVP algorithms known today: $2^{\Theta(N)}$, asymptotically much faster.
Some algorithms taking time $2^{(c+o(1))N}$, under plausible assumptions:

$c \approx 0.415$: 2008 Nguyen–Vidick.
$c \approx 0.415$: 2010 Micciancio–Voulgaris.
$c \approx 0.384$: 2011 Wang–Liu–Tian–Bi.

How secure is SVP?

Best SVP algorithms known at the end of the 20th century:
time $2^{\Theta(N \log N)}$ for almost all dimension-N lattices.

Best SVP algorithms known today: $2^{\Theta(N)}$, asymptotically much faster.
Some algorithms taking time $2^{(c+o(1))N}$, under plausible assumptions:

$c \approx 0.415$: 2008 Nguyen–Vidick.
$c \approx 0.415$: 2010 Micciancio–Voulgaris.
$c \approx 0.384$: 2011 Wang–Liu–Tian–Bi.
$c \approx 0.378$: 2013 Zhang–Pan–Hu.

How secure is SVP?

Best SVP algorithms known at the end of the 20th century:
time $2^{\Theta(N \log N)}$ for almost all dimension-N lattices.

Best SVP algorithms known today: $2^{\Theta(N)}$, asymptotically much faster.
Some algorithms taking time $2^{(c+o(1))N}$, under plausible assumptions:

$c \approx 0.415$: 2008 Nguyen–Vidick.
$c \approx 0.415$: 2010 Micciancio–Voulgaris.
$c \approx 0.384$: 2011 Wang–Liu–Tian–Bi.
$c \approx 0.378$: 2013 Zhang–Pan–Hu.
$c \approx 0.337$: 2014 Laarhoven.

How secure is SVP?

Best SVP algorithms known at the end of the 20th century:
time $2^{\Theta(N \log N)}$ for almost all dimension-N lattices.

Best SVP algorithms known today: $2^{\Theta(N)}$, asymptotically much faster.
Some algorithms taking time $2^{(c+o(1))N}$, under plausible assumptions:

$c \approx 0.415$: 2008 Nguyen–Vidick.
$c \approx 0.415$: 2010 Micciancio–Voulgaris.
$c \approx 0.384$: 2011 Wang–Liu–Tian–Bi.
$c \approx 0.378$: 2013 Zhang–Pan–Hu.
$c \approx 0.337$: 2014 Laarhoven.
$c \approx 0.298$: 2015 Laarhoven–de Weger.

How secure is SVP?

Best SVP algorithms known at the end of the 20th century:
time $2^{\Theta(N \log N)}$ for almost all dimension-N lattices.

Best SVP algorithms known today: $2^{\Theta(N)}$, asymptotically much faster.
Some algorithms taking time $2^{(c+o(1))N}$, under plausible assumptions:

$c \approx 0.415$: 2008 Nguyen–Vidick.
$c \approx 0.415$: 2010 Micciancio–Voulgaris.
$c \approx 0.384$: 2011 Wang–Liu–Tian–Bi.
$c \approx 0.378$: 2013 Zhang–Pan–Hu.
$c \approx 0.337$: 2014 Laarhoven.
$c \approx 0.298$: 2015 Laarhoven–de Weger.
$c \approx 0.292$: 2015 Becker–Ducas–Gama–Laarhoven.

How secure is SVP?

Best SVP algorithms known at the end of the 20th century:
time $2^{\Theta(N \log N)}$ for almost all dimension-N lattices.

Best SVP algorithms known today: $2^{\Theta(N)}$, asymptotically much faster.
Some algorithms taking time $2^{(c+o(1))N}$, under plausible assumptions:

$c \approx 0.415$: 2008 Nguyen–Vidick.
$c \approx 0.415$: 2010 Micciancio–Voulgaris.
$c \approx 0.384$: 2011 Wang–Liu–Tian–Bi.
$c \approx 0.378$: 2013 Zhang–Pan–Hu.
$c \approx 0.337$: 2014 Laarhoven.
$c \approx 0.298$: 2015 Laarhoven–de Weger.
$c \approx 0.292$: 2015 Becker–Ducas–Gama–Laarhoven.
$c \approx 0.268$ quantum algorithm: 2014 Laarhoven–Mosca–van de Pol.

How secure is SVP?

Best SVP algorithms known at the end of the 20th century:
time $2^{\Theta(N \log N)}$ for almost all dimension-N lattices.

Best SVP algorithms known today: $2^{\Theta(N)}$, asymptotically much faster.
Some algorithms taking time $2^{(c+o(1))N}$, under plausible assumptions:

$c \approx 0.415$: 2008 Nguyen–Vidick.
$c \approx 0.415$: 2010 Micciancio–Voulgaris.
$c \approx 0.384$: 2011 Wang–Liu–Tian–Bi.
$c \approx 0.378$: 2013 Zhang–Pan–Hu.
$c \approx 0.337$: 2014 Laarhoven.
$c \approx 0.298$: 2015 Laarhoven–de Weger.
$c \approx 0.292$: 2015 Becker–Ducas–Gama–Laarhoven.
$c \approx 0.268$ quantum algorithm: 2014 Laarhoven–Mosca–van de Pol.

Who thinks this is the end of the story?
Is $2^{(0.1+o(1))N}$ possible? $2^{\Theta(N/\log N)}$? $2^{N^{1/2+o(1)}}$?

How secure is approx SVP?

Public-key lattice-based crypto allows approximation factors.
How much does this damage security?

How secure is approx SVP?

Public-key lattice-based crypto allows approximation factors.
How much does this damage security?

2002 Micciancio–Goldwasser (emphasis added): "To date, the best known polynomial time (possibly randomized) approximation algorithms for SVP and CVP achieve worst-case (over the choice of the input) approximation factors $\gamma(n)$ that are **essentially exponential** in the rank n."

2007 Regev:

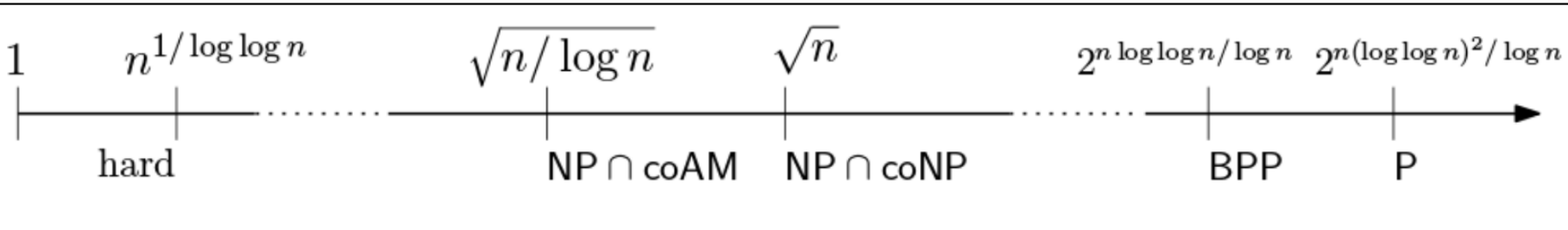


Figure 1: The complexity of lattice problems (some constants omitted)

2013 Micciancio: "Smooth trade-off between running time and approximation: $\gamma \approx 2^{O(n \log\log T/\log T)}$"

Quantum attacks against cyclotomic lattice problems

STOC 2014 Eisenträger–Hallgren–Kitaev–Song:
poly-time quantum algorithm for $K \mapsto \mathcal{O}_K^\times$.

K: number field.
$\mathcal{O}_K$: ring of algebraic integers in K.
$\mathcal{O}_K^\times$: group of units in $\mathcal{O}_K$.

Quantum attacks against cyclotomic lattice problems

STOC 2014 Eisenträger–Hallgren–Kitaev–Song:
poly-time quantum algorithm for $K \mapsto \mathcal{O}_K^\times$.

K: number field.
$\mathcal{O}_K$: ring of algebraic integers in K.
$\mathcal{O}_K^\times$: group of units in $\mathcal{O}_K$.

2015 (and SODA 2016) Biasse–Song,
also using an idea from 2014 Campbell–Groves–Shepherd:
poly-time quantum algorithm for $K, g\mathcal{O}_K \mapsto \zeta_m^j g$ for some j,
assuming cyclotomic $K = \mathbf{Q}(\zeta_m)$, small h_m^+, very short g.

Quantum attacks against cyclotomic lattice problems

STOC 2014 Eisenträger–Hallgren–Kitaev–Song:
poly-time quantum algorithm for $K \mapsto \mathcal{O}_K^\times$.

K: number field.
$\mathcal{O}_K$: ring of algebraic integers in K.
$\mathcal{O}_K^\times$: group of units in $\mathcal{O}_K$.

2015 (and SODA 2016) Biasse–Song,
also using an idea from 2014 Campbell–Groves–Shepherd:
poly-time quantum algorithm for $K, g\mathcal{O}_K \mapsto \zeta_m^j g$ for some j,
assuming cyclotomic $K = \mathbf{Q}(\zeta_m)$, small h_m^+, very short g.

This recovers secret keys in, e.g.,
STOC 2009 Gentry homomorphic-encryption system using cyclotomics,
Eurocrypt 2013 Garg–Gentry–Halevi multilinear-map system, etc.

Is the attack idea limited to very short generators?

More lattice problems of interest:

$I \mapsto$ shortest nonzero vector in I. ("Exact Ideal-SVP".)

$I \mapsto$ close to shortest nonzero vector in I. ("Approximate Ideal-SVP".)

Attack is against principal I with a *very short generator*.

Is the attack idea limited to very short generators?

More lattice problems of interest:
$I \mapsto$ shortest nonzero vector in I. ("Exact Ideal-SVP".)
$I \mapsto$ close to shortest nonzero vector in I. ("Approximate Ideal-SVP".)

Attack is against principal I with a *very short generator*.

2015 Peikert says technique is "useless" for more general principal ideals. ("We simply hadn't realized that the added guarantee of a short generator would transform the technique from useless to devastatingly effective.")

Is the attack idea limited to very short generators?

More lattice problems of interest:
$I \mapsto$ shortest nonzero vector in I. ("Exact Ideal-SVP".)
$I \mapsto$ close to shortest nonzero vector in I. ("Approximate Ideal-SVP".)

Attack is against principal I with a *very short generator*.

2015 Peikert says technique is "useless" for more general principal ideals. ("We simply hadn't realized that the added guarantee of a short generator would transform the technique from useless to devastatingly effective.")

Counterargument: attack is poly time against **arbitrary principal ideals** for approx factor $2^{N^{1/2+o(1)}}$ in degree-N cyclotomics, assuming small h^+. See, e.g., 2016 Cramer–Ducas–Peikert–Regev.

Is the attack idea limited to principal ideals?

2015 Peikert:
"Although cyclotomics have a lot of structure, nobody has yet found a way to exploit it in attacking Ideal-SVP/BDD . . . For commonly used rings, principal ideals are an extremely small fraction of all ideals. . . . The weakness here is not so much due to the structure of cyclotomics, but rather to the extra structure of principal ideals that have short generators."

Is the attack idea limited to principal ideals?

2015 Peikert:
"Although cyclotomics have a lot of structure, nobody has yet found a way to exploit it in attacking Ideal-SVP/BDD ... For commonly used rings, principal ideals are an extremely small fraction of all ideals. ... The weakness here is not so much due to the structure of cyclotomics, but rather to the extra structure of principal ideals that have short generators."

Counterargument, 2016 Cramer–Ducas–Wesolowski:
fast Ideal-SVP attack for approx factor $2^{N^{1/2+o(1)}}$ in degree-N cyclotomics, under plausible assumptions about class-group generators etc.
Starts from Biasse–Song, uses more features of cyclotomic fields.

This shreds the standard approx-Ideal-SVP tradeoff picture.

Non-cyclotomic lattice-based cryptography

Cyclotomics are scary. Let's explore alternatives:

- Eliminate the ideal structure.
  e.g., use LWE instead of Ring-LWE.
  But this limits the security achievable for key size K.

Non-cyclotomic lattice-based cryptography

Cyclotomics are scary. Let's explore alternatives:

- Eliminate the ideal structure.
  e.g., use LWE instead of Ring-LWE.
  But this limits the security achievable for key size K.
- 2016 Bernstein–Chuengsatiansup–Lange–van Vredendaal "NTRU Prime" (preliminary announcement 2014.02, *before* these attacks): as in discrete-log crypto, eliminate unnecessary ring morphisms.
  Use prime degree, large Galois group: e.g., $x^p - x - 1$.

Non-cyclotomic lattice-based cryptography

Cyclotomics are scary. Let's explore alternatives:

- Eliminate the ideal structure.
  e.g., use LWE instead of Ring-LWE.
  But this limits the security achievable for key size K.
- 2016 Bernstein–Chuengsatiansup–Lange–van Vredendaal "NTRU Prime" (preliminary announcement 2014.02, *before* these attacks): as in discrete-log crypto, eliminate unnecessary ring morphisms. Use prime degree, large Galois group: e.g., $x^p - x - 1$.
- This talk: Switch from cyclotomics to other Galois number fields. Another popular example in algebraic-number-theory textbooks: multiquadratics; e.g., $\mathbf{Q}(\sqrt{2}, \sqrt{3}, \sqrt{5}, \sqrt{7}, \sqrt{11}, \sqrt{13}, \sqrt{17}, \sqrt{19}, \sqrt{23})$.

A reasonable multiquadratic cryptosystem

Case study of a lattice-based cryptosystem
that was already defined in detail for arbitrary number fields:
2010 Smart–Vercauteren, optimized version of 2009 Gentry.

Parameter: $R = \mathbf{Z}[\alpha]$ for an algebraic integer α.
Secret key: very short $g \in R$.
Public key: gR.

A reasonable multiquadratic cryptosystem

Case study of a lattice-based cryptosystem
that was already defined in detail for arbitrary number fields:
2010 Smart–Vercauteren, optimized version of 2009 Gentry.

Parameter: $R = \mathbf{Z}[\alpha]$ for an algebraic integer α.
Secret key: very short $g \in R$.
Public key: gR.

To handle multiquadratics better,
we generalized beyond $\mathbf{Z}[\alpha]$; fixed a keygen speed problem;
used twisted Hadamard transforms as replacement for FFTs;
adapted 2011 Gentry–Halevi cyclotomic speedups to multiquadratics.

Like Smart–Vercauteren, we took $N \in \lambda^{2+o(1)}$ for target security 2^λ.
Checked security against standard lattice attacks:
nothing better than exponential time.

Our main multiquadratic results

See `https://multiquad.cr.yp.to` for paper and software:

- **Huge parameter range for this cryptosystem is now broken.**
- Applicability: Attack is non-quantum. Works on your PC.
- Verifiability: We implemented the attack. It works as predicted.
- Fits the pattern motivating the "NTRU prime" recommendations: subfields and automorphisms tend to damage security.
- Speed: Quasipoly for $\mathbb{Q}(\sqrt{d_1}, \ldots, \sqrt{d_n})$ if $d_1, \ldots, d_n$ are quasipoly. (Quasipoly as function of 2^n: i.e., logs are bounded by $n^{O(1)}$.)
- Analysis assuming standard heuristics:
  - Attack always finds unit group $\mathcal{O}_L^\times$.
  - Attack always finds some generator of input ideal.
  - Attack always finds short generator if $d_1, \ldots, d_n > 2^{1.03n}$.
- Experiments find short generator even for much smaller d's. Some failures for tiny d's—presumably should do better BDD.

Conventional techniques to find generators

Want to find g given gR.

Lattice-basis-reduction algorithms find fairly short nonzero vectors $\alpha \in gR$. Could α be as small as g? Extremely rare in high dimensions.

Conventional techniques to find generators

Want to find g given gR.

Lattice-basis-reduction algorithms find fairly short nonzero vectors $\alpha \in gR$.
Could α be as small as g? Extremely rare in high dimensions.

Much more common: can factor $(\alpha/g)R$ into prime ideals.
Multiply and divide these equations sensibly.

Example from SIAM AG15 talk:
If $\alpha_1 R = gR \cdot P^2 \cdot Q^2$ and $\alpha_2 R = gR \cdot P \cdot Q^3$ and $\alpha_3 R = gR \cdot P \cdot Q^2$
then $P = \alpha_1 \alpha_3^{-1} R$ and $Q = \alpha_2 \alpha_3^{-1} R$ and $gR = \alpha_1^{-1} \alpha_2^{-2} \alpha_3^4 R$.

This is the core of the conventional methods of
computing units (find *two* generators of same ideal; divide);
computing class group (obstruction to all ideals having generators);
computing discrete logarithms by NFS; factoring by NFS; etc.

Some ways to exploit subfields and automorphisms

1. Well known: In many geometric cases, such as elliptic curves, use automorphisms for much faster class-group ("point-counting") methods.

Some ways to exploit subfields and automorphisms

1. Well known: In many geometric cases, such as elliptic curves, use automorphisms for much faster class-group ("point-counting") methods.

2. Factorization of $\alpha \Rightarrow$ factorization of $\sigma(\alpha)$
for each automorphism σ of R.

Special case: obtain "cyclotomic units" from $\alpha = 1 - \zeta_N$.
Analogous: "multiquadratic units" from units of quadratic subfields.

Some ways to exploit subfields and automorphisms

1. Well known: In many geometric cases, such as elliptic curves, use automorphisms for much faster class-group ("point-counting") methods.

2. Factorization of $\alpha \Rightarrow$ factorization of $\sigma(\alpha)$
for each automorphism σ of R.

Special case: obtain "cyclotomic units" from $\alpha = 1 - \zeta_N$.
Analogous: "multiquadratic units" from units of quadratic subfields.

3. How to obtain *all* units of a multiquadratic field?
(Can't find all units of "hard" cyclotomics without solving this!)

1966 Wada: Use subfield relation $u^2 = N_\sigma(u)N_\tau(u)/\sigma(N_{\sigma\tau}(u))$.
First solve same problem recursively for the $\sigma, \tau, \sigma\tau$ subfields;
then try square roots of exponential number of products of generators.

Better: find squares using quadratic chars, as in 1991 Adleman NFS.

Some ways to exploit subfields and automorphisms

1. Well known: In many geometric cases, such as elliptic curves, use automorphisms for much faster class-group ("point-counting") methods.

2. Factorization of $\alpha \Rightarrow$ factorization of $\sigma(\alpha)$
for each automorphism σ of R.

Special case: obtain "cyclotomic units" from $\alpha = 1 - \zeta_N$.
Analogous: "multiquadratic units" from units of quadratic subfields.

3. How to obtain *all* units of a multiquadratic field?
(Can't find all units of "hard" cyclotomics without solving this!)

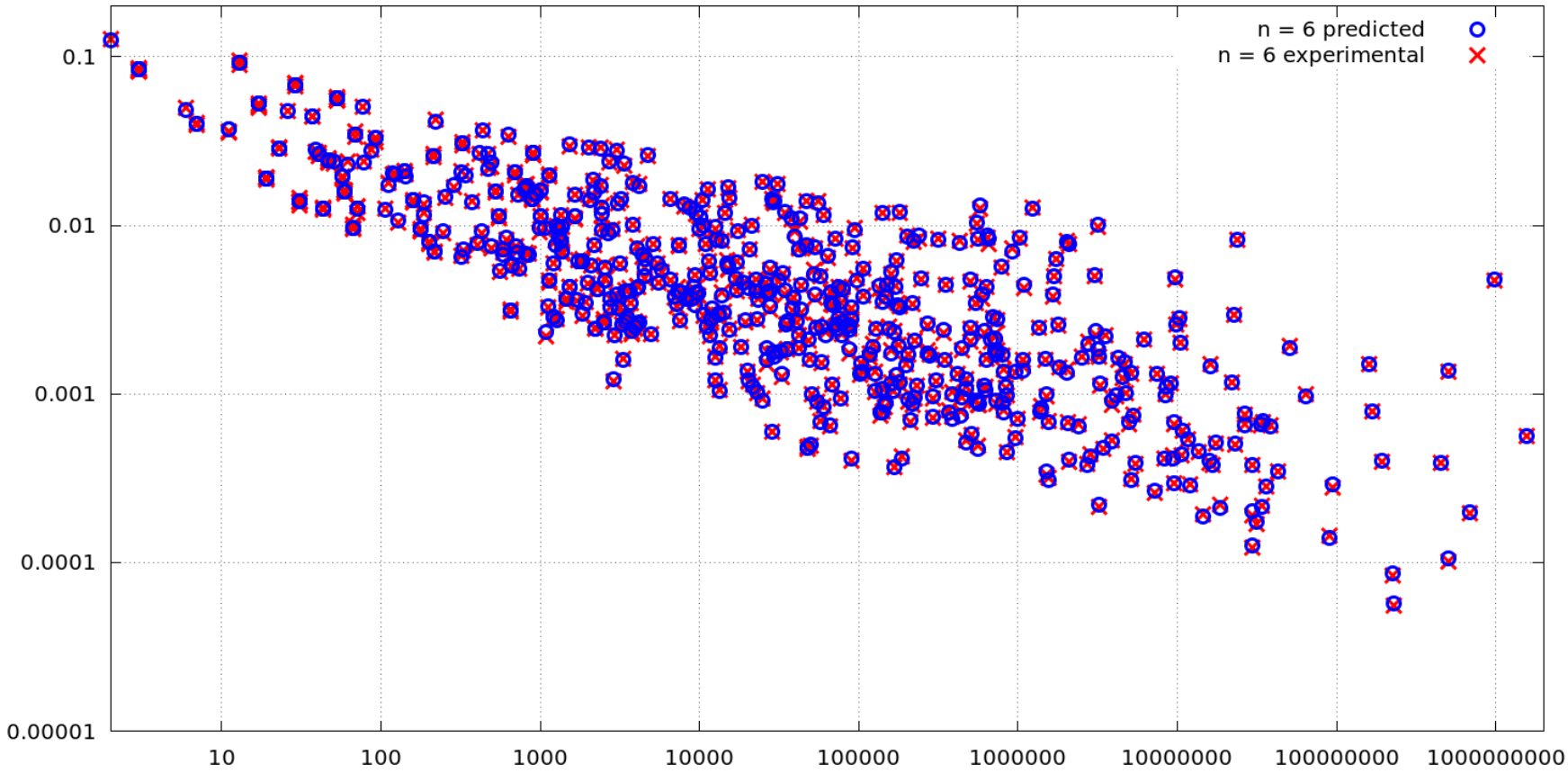
1966 Wada: Use subfield relation $u^2 = N_\sigma(u)N_\tau(u)/\sigma(N_{\sigma\tau}(u))$.
First solve same problem recursively for the $\sigma, \tau, \sigma\tau$ subfields;
then try square roots of exponential number of products of generators.

Better: find squares using quadratic chars, as in 1991 Adleman NFS.

4. Similarly find generators starting with generators of norms.

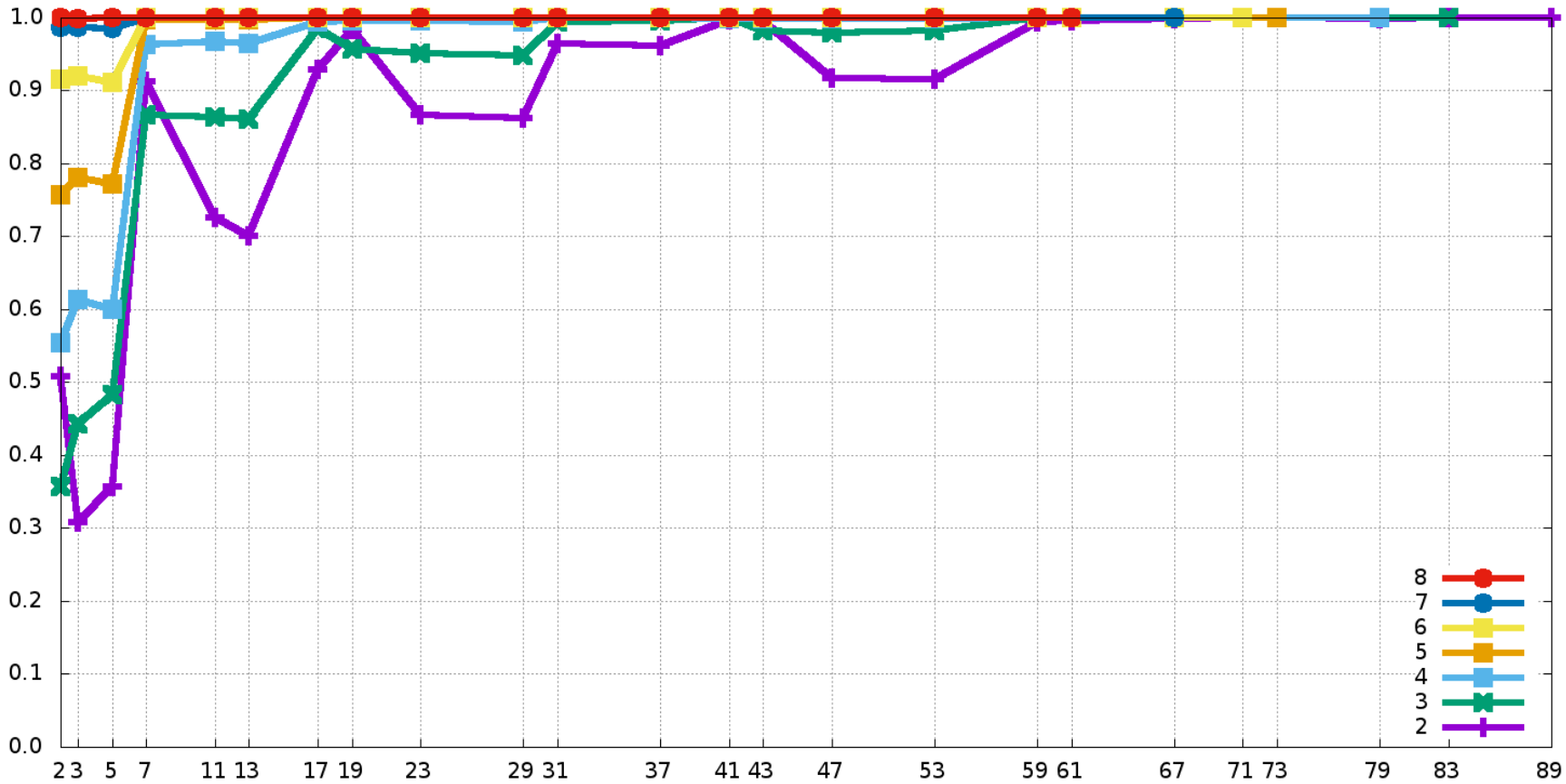
Coefficients for MQ lattice

Vertical axis: Average absolute coefficients of $\operatorname{Log} g$ on MQ basis.
Horizontal axis: $1.11/(2^{n/2}\log(u_D))$.



Success for MQ lattice

Vertical axis: Success probability of simple rounding (in the MQ lattice).
Horizontal axis: d_1, using n consecutive primes for $(d_1, \ldots, d_n)$.



Time (in seconds) to find full lattice and generator

2^n	Sage tower units	Sage absolute units	new units	new units2	new gen	new gen2
8	0.05	0.03	0.90	0.91	0.07	0.07
16	0.48	0.24	2.33	2.39	0.20	0.19
32	6.75	4.73	6.61	7.36	0.56	0.51
64	>700000	>700000	23.30	37.51	1.51	1.51
128			93.02	1560.49	4.95	7.29
256			463.91	31469.23	27.95	100.65

Table: Observed time to compute (once) the units of $\mathbb{Q}(\sqrt{d_1}, \ldots, \sqrt{d_n})$; and to find a generator for the public key in the cryptosystem.

Success at finding short generator of ideal

n	3	4	5	6	7	8
$p_{\mathrm{suc}}(L_1)$	0.122	0.137	0.132	0.036	0.001	0.000
$p_{\mathrm{suc}}(L_n)$	0.203	0.490	0.648	0.936	0.631	0.423
$p_{\mathrm{suc}}(L_{n^2})$	0.784	0.981	1.000	1.000	1.000	1.000

Table: Observed attack success probabilities for various multiquadratic fields.