Estimating the Difficulty of Breaking Lattice-Based Cryptography

with a focus on LWE

Martin R. Albrecht

2022: Royal Holloway, University of London, 2023: King's College London & SandboxAQ

The Plan

- Introduce estimating the cost of solving LWE-based schemes
  - strategies (primal and dual lattice attacks)
  - lattice reduction
  - finding short vectors
  - quantum considerations
- Introduce usage of "lattice estimator" along the way
  - highlight limitations[1]

```
https://github.com/malb/lattice-estimator/
```

[1] Most of what I know about running open-source projects, I learned from William Stein who responded with "Easy, implement it and send us a patch!" when asked "How do I do . . . in SageMath?"

Example

```
_ = LWE.estimate.rough(Kyber768)
```

```
usvp                 :: rop: ≈2^182.2, red: ≈2^182.2, δ: 1.002902, β: 624, d: 1427, tag: usvp
dual_hybrid          :: rop: ≈2^184.3, mem: ≈2^180.8, m: 705, β: 630, d: 1451, ▫: 1, ζ: 22, tag: dual_hybrid
```

```
_ = LWE.estimate(Kyber768)
```

```
arora-gb             :: rop: ≈2^inf, dreg: 94, mem: ≈2^513.9, t: 2, m: ≈2^inf, ...
bkw                  :: rop: ≈2^238.3, m: ≈2^225.5, mem: ≈2^226.5, b: 19, t1: 1, t2: 17, ℓ: 18, ...
usvp                 :: rop: ≈2^204.9, red: ≈2^204.9, δ: 1.002902, β: 624, d: 1427, tag: usvp
bdd                  :: rop: ≈2^201.0, red: ≈2^200.0, svp: ≈2^200.0, β: 606, η: 641, d: 1425, tag: bdd
bdd_hybrid           :: rop: ≈2^201.0, red: ≈2^200.0, svp: ≈2^200.0, β: 606, η: 641, ...
bdd_mitm_hybrid      :: rop: ≈2^356.7, red: ≈2^355.8, svp: ≈2^355.6, β: 623, η: 2, ζ: 189, ...
dual                 :: rop: ≈2^214.2, mem: ≈2^133.4, m: 723, β: 653, d: 1491, ▫: 1, tag: dual
dual_hybrid          :: rop: ≈2^206.4, mem: ≈2^201.9, m: 701, β: 625, d: 1437, ▫: 1, ζ: 32, tag: dual_hybrid
```

Computational Problems

Given $(\mathbf{A}, \mathbf{c})$, find $\mathbf{s}$ when

$$\begin{pmatrix} \\ \mathbf{c} \\ \\ \end{pmatrix} \equiv \begin{pmatrix} \leftarrow & n & \rightarrow \\ & \mathbf{A} & \end{pmatrix} \cdot \begin{pmatrix} \\ \mathbf{s} \\ \\ \end{pmatrix} + \begin{pmatrix} \\ \mathbf{e} \\ \\ \end{pmatrix} \bmod q$$

for $\mathbf{c} \in \mathbb{Z}_q^m$, $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$, and $\mathbf{s} \in \mathbb{Z}^n$ and $\mathbf{e} \in \mathbb{Z}^m$ having small entries.

Learning with Errors

Given $(\mathbf{A}, \mathbf{c})$, find $\mathbf{s}$ when

$$\begin{pmatrix} \\ \mathbf{c} \\ \\ \\ \end{pmatrix} \equiv \begin{pmatrix} \leftarrow & n & \rightarrow \\ & & \\ & \mathbf{A} & \\ & & \\ & & \end{pmatrix} \cdot \begin{pmatrix} \\ \mathbf{s} \\ \\ \end{pmatrix} + \begin{pmatrix} \\ \mathbf{e} \\ \\ \\ \end{pmatrix} \bmod q$$

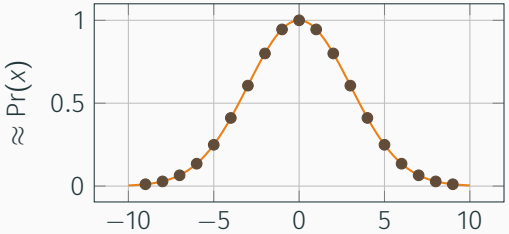
for $\mathbf{c} \in \mathbb{Z}_q^m$, $\mathbf{A} \in \mathbb{Z}_q^{m \times n}$, and $\mathbf{s} \in \mathbb{Z}^n$ and $\mathbf{e} \in \mathbb{Z}^m$ having small entries.

"Having Small Entries"

Discrete Gaussian the classic, annoying to sample from; ➡

Binomial $\sum_{0 \le i < \eta}(b_i - b_{\eta+i}) \mid b_j \hookleftarrow\$ \{0,1\}$

Small Uniform $\hookleftarrow\$ [a, \ldots, b] \mid a, b \in \mathbb{N}$




```
from estimator import *
ND.DiscreteGaussian(stddev=2), ND.CenteredBinomial(eta=8), ND.Uniform(-3, 3)
```

```
(D(σ=2.00), D(σ=2.00), D(σ=2.00))
```

- Literature assumes these all behave essentially the same under attacks
- No loss in security if secret **s** and error **e** have same distribution [ACPS09]

Estimator

```
Kyber768 = LWEParameters(
    n=3 * 256,
    q=3329,
    Xs=ND.CenteredBinomial(2),
    Xe=ND.CenteredBinomial(2),
    m=3 * 256,
    tag="Kyber 768",
)
```

Q: ”How do I do NTRU?”

A: ”Easy, implement it and send us a patch.”

We do not offer `NTRUParameters`, but you can hack it in using `LWEParameters`

Approaches

Unique SVP/BDD: Translation

We can reformulate $\mathbf{c} - \mathbf{A} \cdot \mathbf{s} \equiv \mathbf{e} \bmod q$ over the Integers as:

$$\begin{pmatrix} q\mathbf{I} & -\mathbf{A} \\ 0 & \mathbf{I} \end{pmatrix} \cdot \begin{pmatrix} * \\ \mathbf{s} \end{pmatrix} + \begin{pmatrix} \mathbf{c} \\ \mathbf{0} \end{pmatrix} = \begin{pmatrix} \mathbf{e} \\ \mathbf{s} \end{pmatrix}$$

Alternatively:

$$\mathbf{B} = \begin{pmatrix} q\mathbf{I} & -\mathbf{A} & \mathbf{c} \\ 0 & \mathbf{I} & 0 \\ 0 & 0 & 1 \end{pmatrix}, \qquad \mathbf{B} \cdot \begin{pmatrix} * \\ \mathbf{s} \\ 1 \end{pmatrix} = \begin{pmatrix} \mathbf{e} \\ \mathbf{s} \\ 1 \end{pmatrix}$$

In other words, there exists an integer-linear combination of the columns of $\mathbf{B}$ that produces a vector with “unusually” small entries $\rightarrow$ a unique shortest vector.

Unique SVP: Computational Problem

Unique Shortest Vector Problem for q-ary Lattices

Find a unique shortest vector amongst the integer combinations of the columns of:

$$\mathsf{B} = \begin{pmatrix} q\mathsf{I} & -\mathsf{A} & \mathsf{c} \\ 0 & \mathsf{I} & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

where $\mathsf{B} \in \mathbb{Z}^{d \times d}$.

Decision Variant

Decide if B has an unusually short vector.

Approx SVP/SIS: Translation

- Consider $\mathbf{c} \equiv \mathbf{A} \cdot \mathbf{s} + \mathbf{e} \bmod q$ with both $\mathbf{s}$ and $\mathbf{e}$ short or $\mathbf{c}$ uniform.
- Let $\mathbf{u}_i$ be short vectors such that $\mathbf{v}_i^T := \mathbf{u}_i^T \cdot \mathbf{A} \bmod q$ is also short.
- Compare:
  - $\mathbf{u}_i^T \cdot \mathbf{c} \equiv \mathbf{u}_i^T \cdot \mathbf{A} \cdot \mathbf{s} + \mathbf{u}_i^T \cdot \mathbf{e} \equiv \mathbf{v}_i^T \cdot \mathbf{s} + \mathbf{u}_i^T \cdot \mathbf{e}$ which is somewhat short
  - $\mathbf{u}_i^T \cdot \mathbf{c}$ which is uniform
- The shorter $(\mathbf{u}_i, \mathbf{v}_i)$ the fewer samples of $\mathbf{u}_i^T \cdot \mathbf{c}$ we need to consider
- Note

$$\begin{pmatrix} q\mathbf{I} & \mathbf{A}^T \\ 0 & \mathbf{I} \end{pmatrix} \cdot \begin{pmatrix} * \\ \mathbf{u}_i \end{pmatrix} = \begin{pmatrix} \mathbf{v}_i \\ \mathbf{u}_i \end{pmatrix}$$

Approx SVP: Computational Problem

Short Vectors Problem for q-ary Lattices

Find vectors $(\mathbf{u}_i, \mathbf{v}_i)$ of norm $\|(\mathbf{u}_i, \mathbf{v}_i)\| \leq \beta$ amongst the integer combinations of the columns of:

$$\mathsf{B} = \begin{pmatrix} q\mathsf{I} & \mathsf{A}^T \\ 0 & \mathsf{I} \end{pmatrix}$$

where $\mathsf{B} \in \mathbb{Z}^{d \times d}$.

Search Variant

Can extend this distinguishing attack to recover $\mathbf{s}$: guess a component and run the distinguisher

Lattice Reduction <3 Combinatorics: Hybrid Attacks

Both approaches can be augmented with a combinatorial step

- guess parts of the secret and run the lattice attack on a smaller dimensional lattice
- due to linearity costs are additive not multiplicative, i.e.

$$\approx T_{guess} + T_{lattice}$$

Estimator (Primal)

plain uSVP

```
LWE.primal_usvp(Kyber768)
```

```
rop: ≈2^204.9, red: ≈2^204.9, δ: 1.002902, β: 624, d: 1427, tag: usvp
```

plain BDD (minor parameter relaxation compared to uSVP)

```
LWE.primal_bdd(Kyber768)
```

```
rop: ≈2^201.0, red: ≈2^200.0, svp: ≈2^200.0, β: 606, η: 641, d: 1425, tag: bdd
```

BDD + combinatorics

```
LWE.primal_hybrid(Kyber768)
```

```
rop: ≈2^356.7, red: ≈2^355.8, svp: ≈2^355.6, β: 623, η: 2, ζ: 189, |S|: ≈2^367.1, d: 1278, ...
```

Estimator (Dual)

plain SIS

```
LWE.dual(Kyber768)
```

```
rop: ≈2^214.2, mem: ≈2^133.4, m: 723, β: 653, d: 1491, ▫: 1, tag: dual
```

SIS + combinatorics

```
LWE.dual_hybrid(Kyber768)
```

```
rop: ≈2^206.4, mem: ≈2^201.9, m: 701, β: 625, d: 1437, ▫: 1, ζ: 32, tag: dual_hybrid
```

Q: "How do I use the estimates from [MAT22]?"

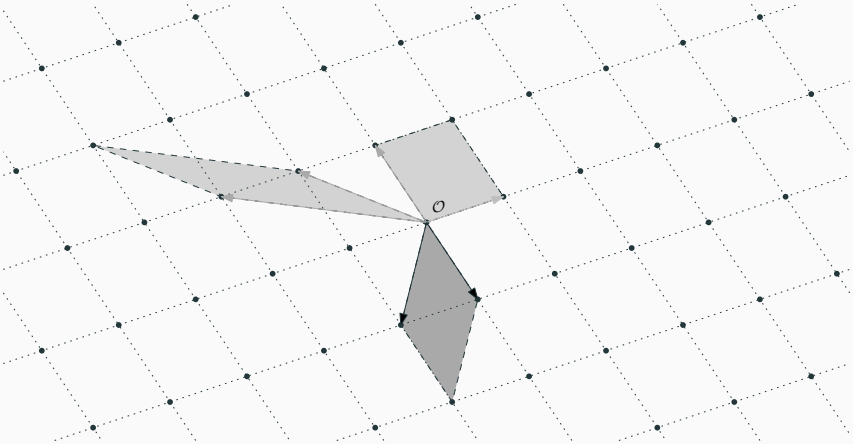
A: "Easy, I have implemented it [AS22] and just need to submit the patch."[2]

[2] Stuff gets implemented in the estimator when someone needs it for a project; ideally, attack authors would submit estimates for their attacks.

Lattice Reduction

Lattice Volume

The volume of a lattice is the volume of its fundamental parallelepiped.



Picture Credit: Joop van der Pol

Gaussian Heuristic

- The Gaussian heuristic predicts that the number $|\Lambda \cap \mathcal{B}|$ of lattice points inside a measurable body $\mathcal{B} \subset \mathbb{R}^d$ is approximately equal to $\mathrm{Vol}(\mathcal{B})/\mathrm{Vol}(\Lambda)$.
- Applied to Euclidean d-balls, this means that a shortest vector in a lattice has expected norm

$$\lambda_1(\Lambda) \approx \mathrm{GH}(d) \cdot \mathrm{Vol}(\Lambda)^{1/d} \approx \sqrt{\frac{d}{2\pi e}} \cdot \mathrm{Vol}(\Lambda)^{1/d}.$$

Unusually Shortest Vector

When $\lambda_1(\Lambda) \ll \sqrt{\frac{d}{2\pi e}} \cdot \mathrm{Vol}(\Lambda)^{1/d}$.

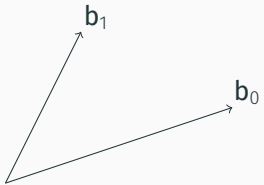
Length of Gram–Schmidt Vectors

It will be useful to consider the lengths of the Gram–Schmidt vectors.

The vector $\mathbf{b}_i^*$ is the orthogonal projection of $\mathbf{b}_i$ to the space spanned by the vectors $\mathbf{b}_0, \ldots, \mathbf{b}_{i-1}$.

Informally, this means taking out the contributions in the directions of previous vectors $\mathbf{b}_0, \ldots, \mathbf{b}_{i-1}$.

We have $\mathrm{Vol}(\Lambda) = \prod_{i=0}^{d-1} \|\mathbf{b}_i^*\|$.



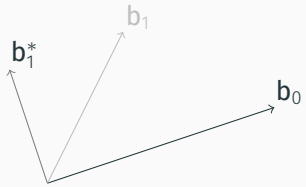
Length of Gram–Schmidt Vectors

It will be useful to consider the lengths of the Gram–Schmidt vectors.

The vector $\mathbf{b}_i^*$ is the orthogonal projection of $\mathbf{b}_i$ to the space spanned by the vectors $\mathbf{b}_0, \ldots, \mathbf{b}_{i-1}$.

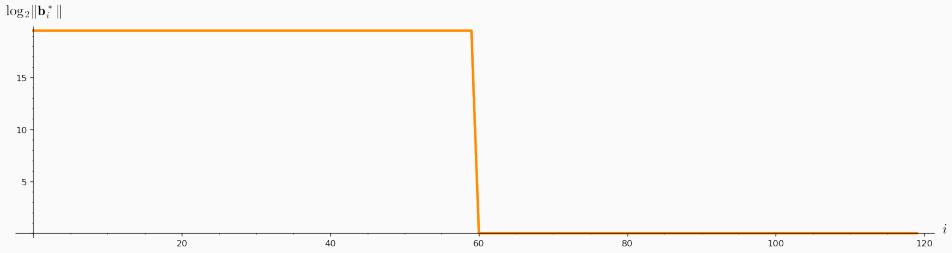
Informally, this means taking out the contributions in the directions of previous vectors $\mathbf{b}_0, \ldots, \mathbf{b}_{i-1}$.

We have $\mathrm{Vol}(\Lambda) = \prod_{i=0}^{d-1} \|\mathbf{b}_i^*\|$.



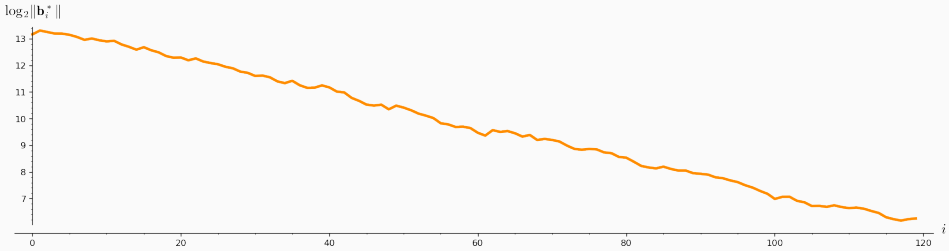
Example

```
A = IntegerMatrix.random(120, "qary", k=60, bits=20)[::-1]
M = GSO.Mat(A, update=True)
line([(i,log(r_, 2)/2) for i, r_ in enumerate(M.r())], **plot_kwds)
```

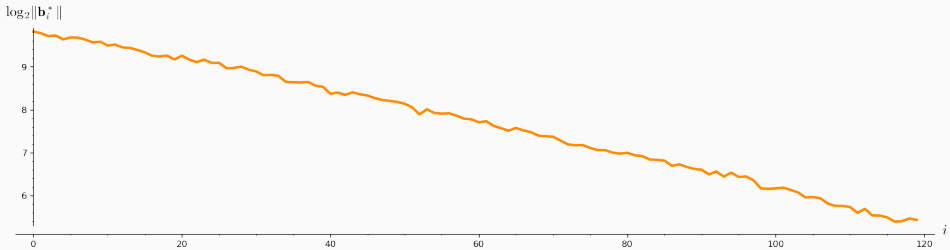


Example - LLL

```
A = LLL.reduction(A)
M = GSO.Mat(A, update=True)
line([(i,log(r_, 2)/2) for i, r_ in enumerate(M.r())], **plot_kwds)
```



GSA



Geometric Series Assumption: The shape after lattice reduction is a line with a flatter slope as lattice reduction gets stronger.[3]

[3] Claus-Peter Schnorr. Lattice Reduction by Random Sampling and Birthday Methods. In: *STACS 2003, 20th Annual Symposium on Theoretical Aspects of Computer Science, Berlin, Germany, February 27 - March 1, 2003, Proceedings*. Ed. by Helmut Alt and Michel Habib. Vol. 2607. Lecture Notes in Computer Science. Springer, 2003, pp. 145–156. DOI: 10.1007/3-540-36494-3_14. URL: http://dx.doi.org/10.1007/3-540-36494-3_14.

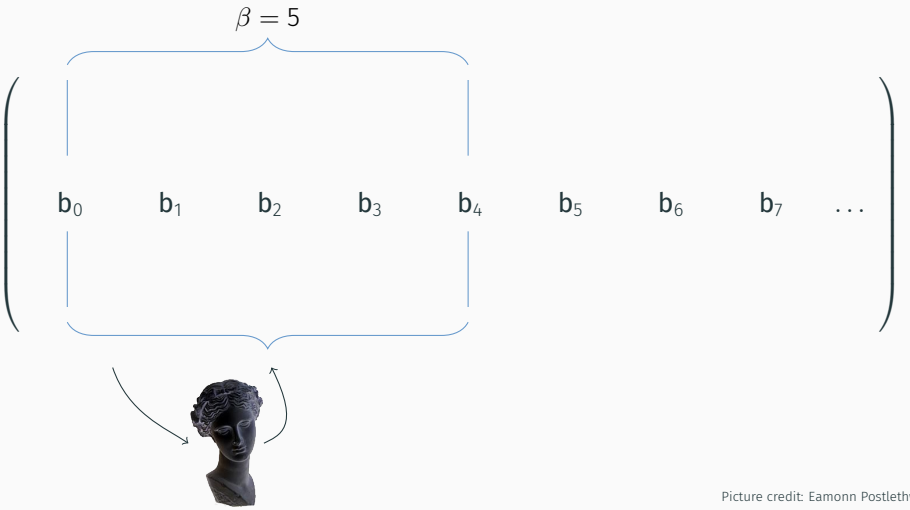
Strong Lattice Reduction: BKZ Algorithm (Block 0)

$$\beta = 5$$

$$\begin{pmatrix} | & | & | & | & | & | & | & | & \\ \mathbf{b}_0 & \mathbf{b}_1 & \mathbf{b}_2 & \mathbf{b}_3 & \mathbf{b}_4 & \mathbf{b}_5 & \mathbf{b}_6 & \mathbf{b}_7 & \dots \\ | & | & | & | & | & | & | & | & \end{pmatrix}$$

Picture credit: Eamonn Postlethwaite

Strong Lattice Reduction: BKZ Algorithm (Block 0)



Picture credit: Eamonn Postlethwaite

Strong Lattice Reduction: BKZ Algorithm (Block 0)

$$\beta = 5$$

$$\begin{pmatrix} \mathbf{b}_0 & \mathbf{b}_1 & \mathbf{b}_2 & \mathbf{b}_3 & \mathbf{b}_4 & \mathbf{b}_5 & \mathbf{b}_6 & \mathbf{b}_7 & \dots \end{pmatrix}$$

Picture credit: Eamonn Postlethwaite

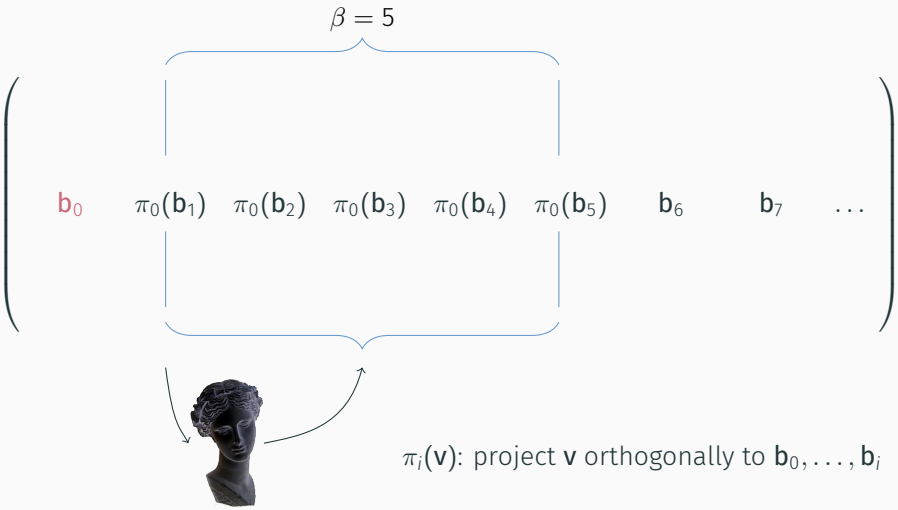
Strong Lattice Reduction: BKZ Algorithm (Block 1)

$$\beta = 5$$

$$\begin{pmatrix} \mathbf{b}_0 & \pi_0(\mathbf{b}_1) & \pi_0(\mathbf{b}_2) & \pi_0(\mathbf{b}_3) & \pi_0(\mathbf{b}_4) & \pi_0(\mathbf{b}_5) & \mathbf{b}_6 & \mathbf{b}_7 & \dots \end{pmatrix}$$

$\pi_i(\mathbf{v})$: project $\mathbf{v}$ orthogonally to $\mathbf{b}_0, \dots, \mathbf{b}_i$

Strong Lattice Reduction: BKZ Algorithm (Block 1)



$\pi_i(\mathbf{v})$: project $\mathbf{v}$ orthogonally to $\mathbf{b}_0, \ldots, \mathbf{b}_i$

Strong Lattice Reduction: BKZ Algorithm (Block 1)

$$\beta = 5$$

$$\begin{pmatrix} \mathbf{b}_0 & \pi_0(\mathbf{b}_1) & \pi_0(\mathbf{b}_2) & \pi_0(\mathbf{b}_3) & \pi_0(\mathbf{b}_4) & \pi_0(\mathbf{b}_5) & \mathbf{b}_6 & \mathbf{b}_7 & \dots \end{pmatrix}$$

$\pi_i(\mathbf{v})$: project $\mathbf{v}$ orthogonally to $\mathbf{b}_0, \dots, \mathbf{b}_i$

BKZ Algorithm

Data: LLL-reduced lattice basis **B**
Data: block size β
repeat *until no more change*
  for $\kappa \leftarrow 0$ **to** $d - 1$ **do**
    LLL on local projected block $[\kappa, \ldots, \kappa + \beta - 1]$;
    v $\leftarrow$ find shortest vector in local projected block $[\kappa, \ldots, \kappa + \beta - 1]$;
    insert **v** into **B**;
  end

Quality

For SIS
$\|\mathbf{b}_0\| \approx \delta_\beta^{d-1} \cdot \mathrm{Vol}(\Lambda)^{1/d}$

For BDD
$\|\mathbf{b}_0\| \approx \delta_\beta^{2\cdot(d-\beta)} \cdot \lambda_1(\Lambda)$

β	2	5	24	50	100	200	500
δ_β	1.0219	1.0186	1.0142	1.0121	1.0096	1.0063	1.0034

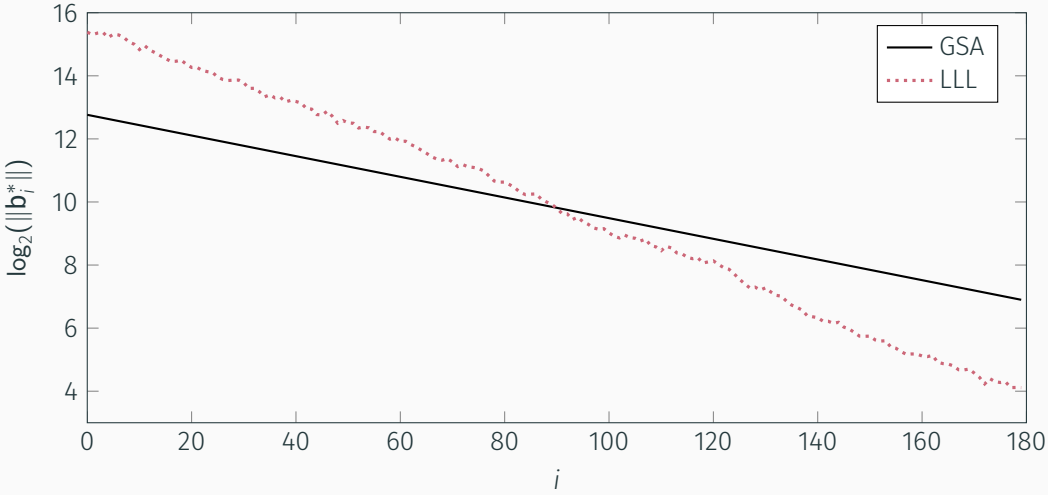
- We have **Root Hermite Factor** $\delta_\beta \approx \mathrm{GH}(\beta)^{1/(\beta-1)}$ for $\beta > 50$.

```
RC.delta(500)
```

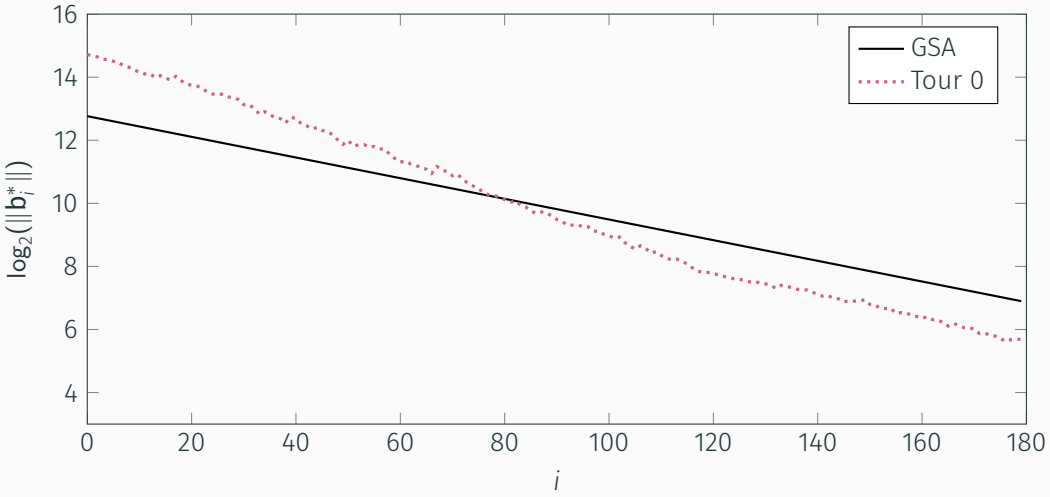
```
1.00340402678510
```

- The slope under the **Geometric Series Assumption** is $\alpha_\beta = \delta_\beta^{-2}$.

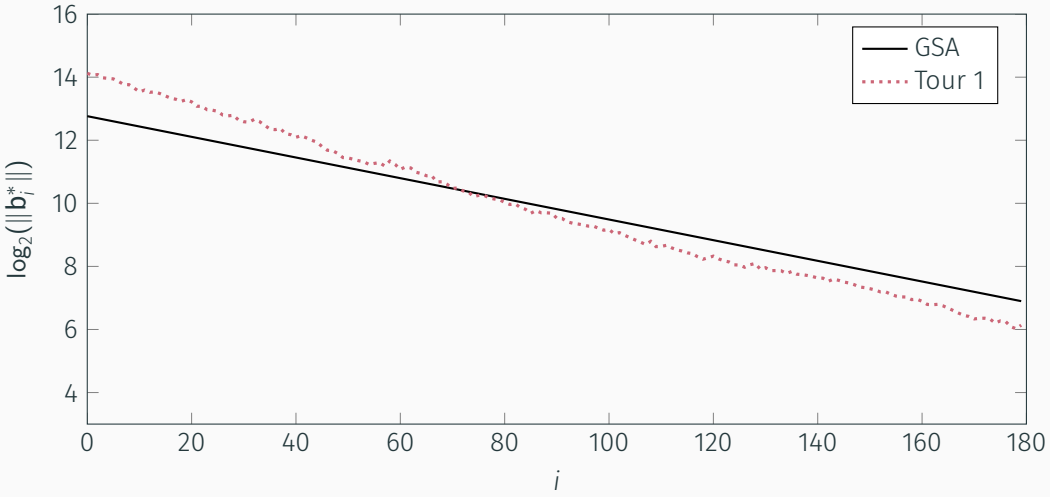
Behaviour in Practice: BKZ-60 in Dimension 180 I



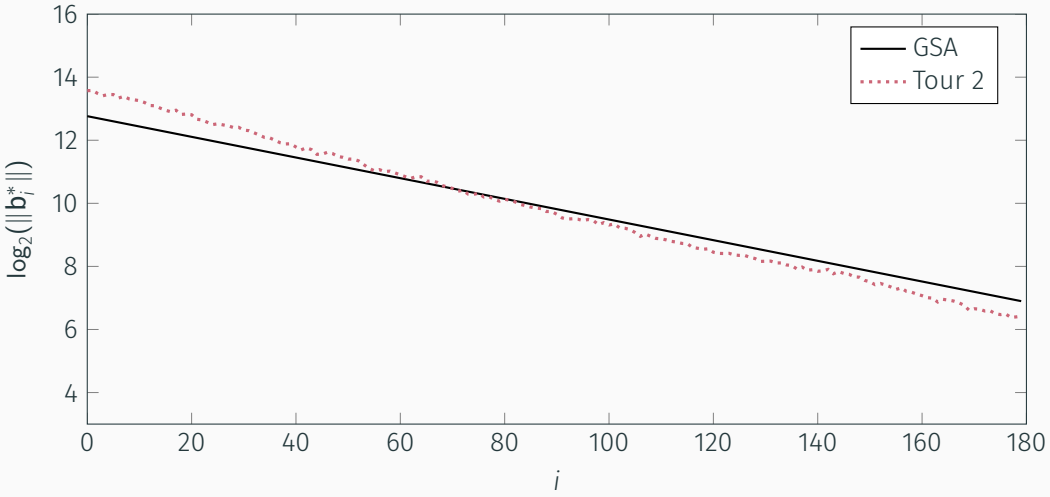
Behaviour in Practice: BKZ-60 in Dimension 180 II



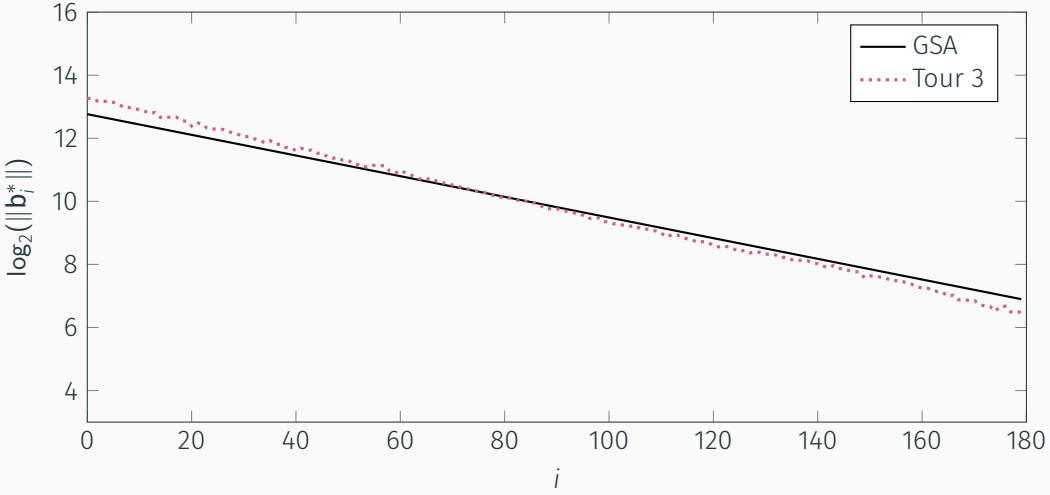
Behaviour in Practice: BKZ-60 in Dimension 180 III



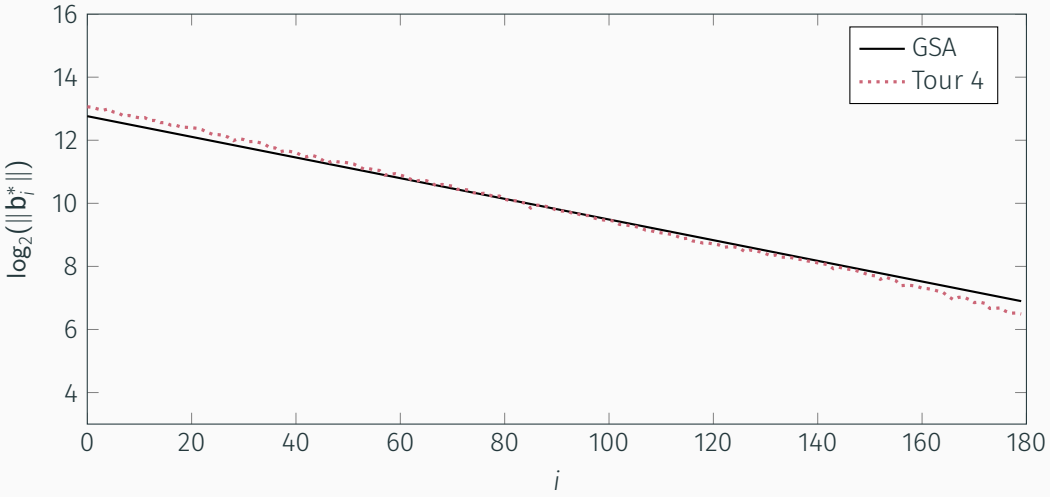
Behaviour in Practice: BKZ-60 in Dimension 180 iv

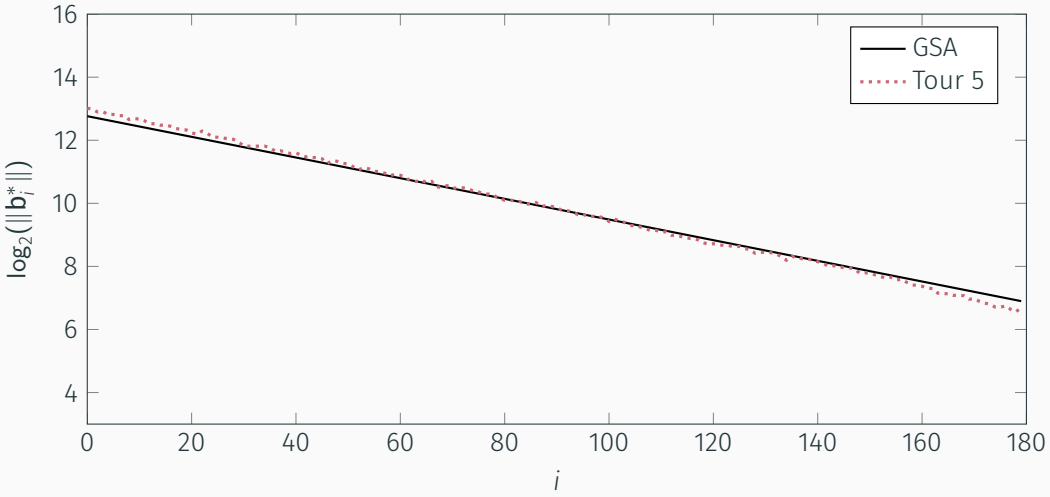


Behaviour in Practice: BKZ-60 in Dimension 180 v

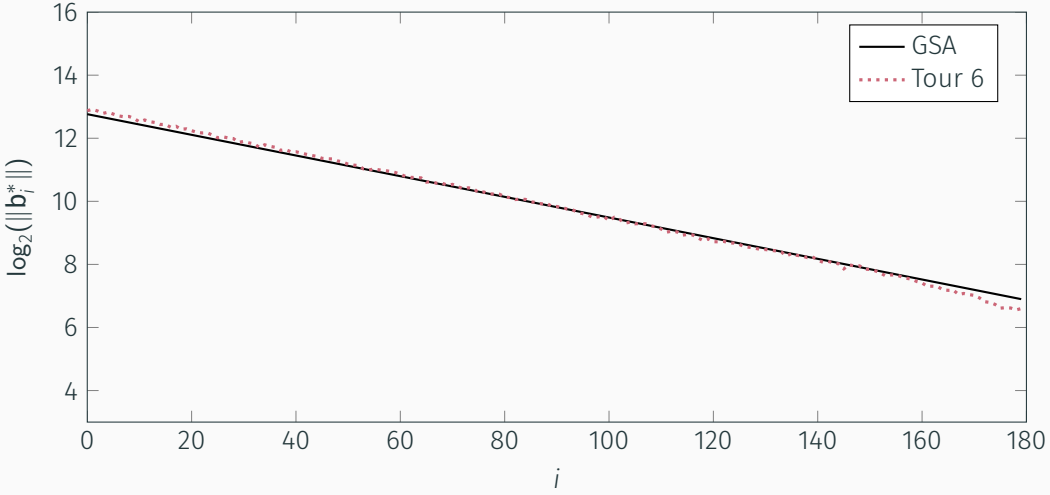


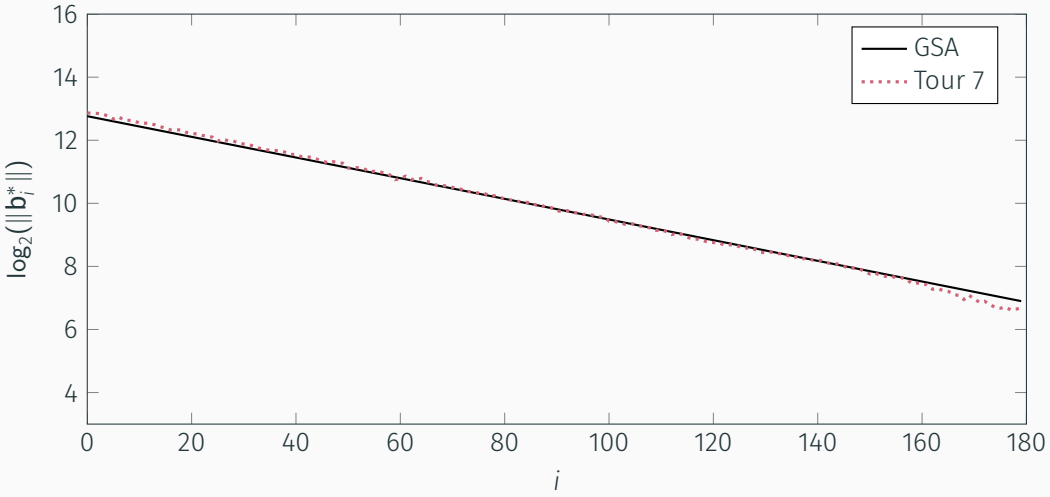
Behaviour in Practice: BKZ-60 in Dimension 180 vi



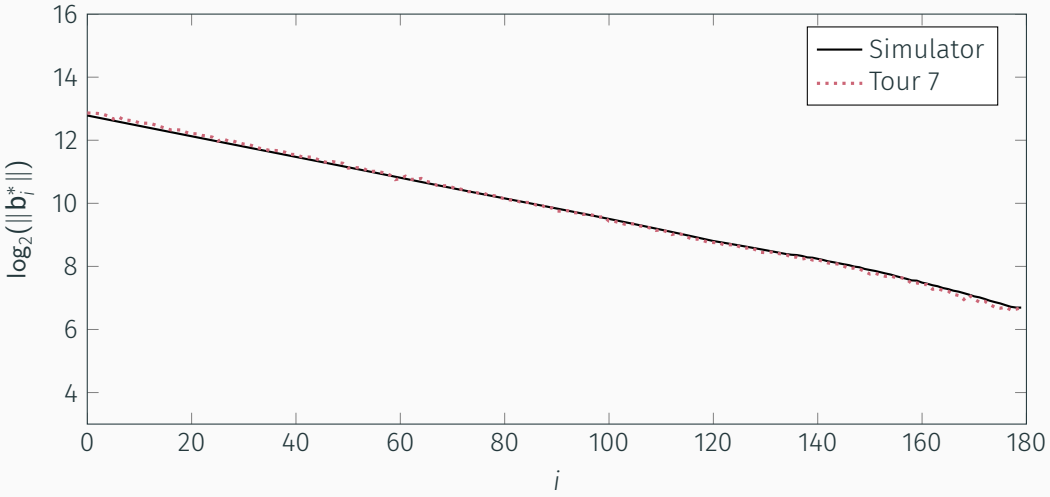

GSA
Tour 5
$\log_2(\|\mathbf{b}_i^*\|)$
i
16
14
12
10
8
6
4
0
20
40
60
80
100
120
140
160
180

Behaviour in Practice: BKZ-60 in Dimension 180 viii




16
14
12
10
8
6
4
0
20
40
60
80
100
120
140
160
180
$\log_2(\|\mathbf{b}_i^*\|)$
i
GSA
Tour 7

Behaviour in Practice: BKZ-60 in Dimension 180 x



Try it at Home

```
from fpylll import *
from fpylll.algorithms.bkz2 import BKZReduction as BKZ2
A = IntegerMatrix.random(180, "qary", k=90, bits=20)
bkz = BKZ2(A)
bkz(BKZ.EasyParam(block_size=60))
```

https://github.com/fplll/fplll C++ library

https://github.com/fplll/fpylll Python interface

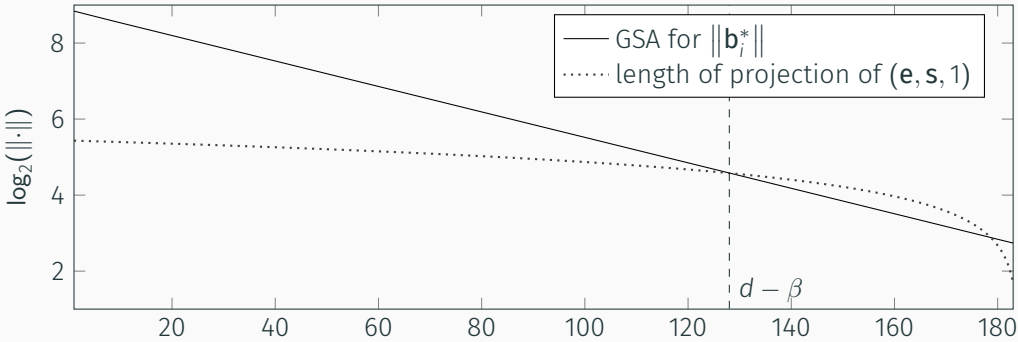
https://github.com/fplll/g6k Sieving (faster lattice reduction)

https://sagemath.org FPyLLL is in SageMath

https://sagecell.sagemath.org/ SageMath in your browser

https://cocalc.com/ SageMath worksheets in your browser

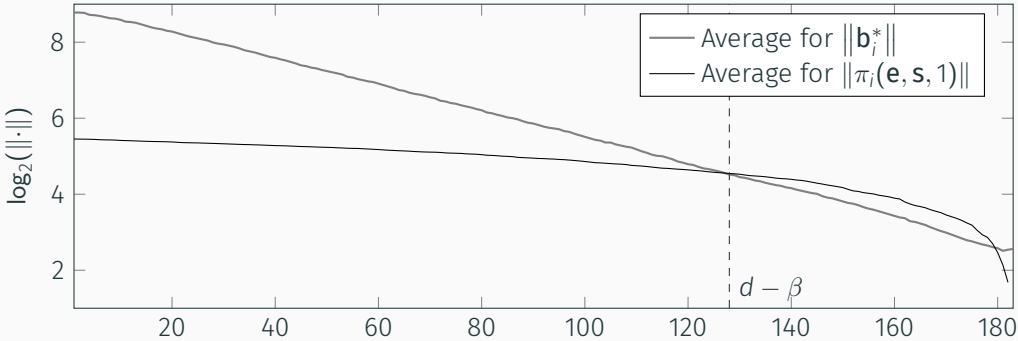
Success Condition for uSVP (Expectation)



$$\sqrt{\beta/d} \cdot \sigma \leq {\delta_\beta}^{2\beta-d-1} \cdot \mathrm{Vol}(\Lambda)^{1/d}$$

Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum Key Exchange - A New Hope. In: *USENIX Security 2016*. Ed. by Thorsten Holz and Stefan Savage. USENIX Association, Aug. 2016, pp. 327–343

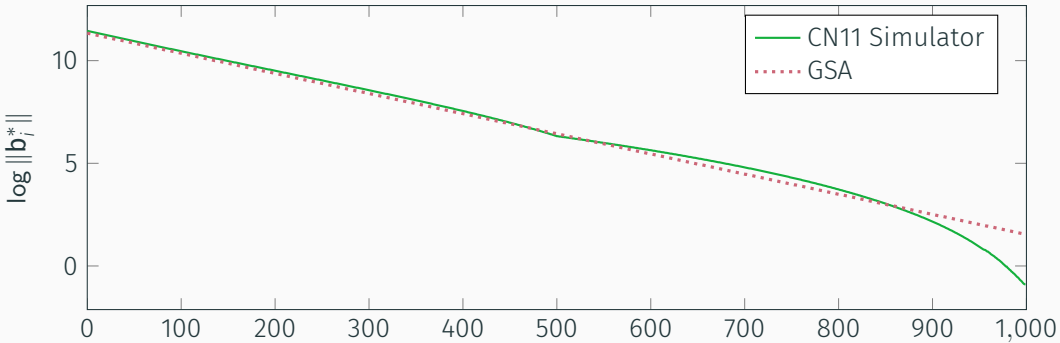
Success Condition for uSVP (Observed)



Martin R. Albrecht, Florian Göpfert, Fernando Virdia, and Thomas Wunderer. Revisiting the Expected Cost of Solving uSVP and Applications to LWE. In: *ASIACRYPT 2017, Part I*. ed. by Tsuyoshi Takagi and Thomas Peyrin. Vol. 10624. LNCS. Springer, Heidelberg, Dec. 2017, pp. 297–322. DOI: 10.1007/978-3-319-70694-8_11

Eamonn W. Postlethwaite and Fernando Virdia. On the Success Probability of Solving Unique SVP via BKZ. In: *PKC 2021, Part I*. ed. by Juan Garay. Vol. 12710. LNCS. Springer, Heidelberg, May 2021, pp. 68–98. DOI: 10.1007/978-3-030-75245-3_4

The GSA is a Lie: Tail Shape



Yuanmi Chen and Phong Q. Nguyen. BKZ 2.0: Better Lattice Security Estimates. In: *ASIACRYPT 2011*. Ed. by Dong Hoon Lee and Xiaoyun Wang. Vol. 7073. LNCS. Springer, Heidelberg, Dec. 2011, pp. 1–20. DOI: 10.1007/978-3-642-25385-0_1

The GSA is a Lie: Tail Shape

```
from estimator import *
print(repr(LWE.primal_usvp(Kyber768, red_shape_model="GSA")))  # used in LWE.estimate.rough
print(repr(LWE.primal_usvp(Kyber768, red_shape_model="CN11"))) # used in LWE.estimate
```

```
rop: ≈2^204.9, red: ≈2^204.9, δ: 1.002902, β: 624, d: 1427, tag: usvp
rop: ≈2^209.9, red: ≈2^209.9, δ: 1.002842, β: 642, d: 1421, tag: usvp
```

Cost

- If τ is the number of tours we do, we run our oracle $\approx \tau \cdot d$ times
- So the cost is roughly $\tau \cdot d \cdot T_{SVP}$.
- We can reduce some of this cost

  Tail is cheaper than the head as we decrease the block sizes

  Progressive BKZ Run BKZ-β' with $\beta' < \beta$ before running BKZ-β

  Skipping blocks We may get away with "skipping" some blocks.
- `LWE.estimate.rough` assumes **one** call to the oracle ("Core-SVP")
- `LWE.estimate` assumes roughly $8 \cdot d$, i.e. $\tau = 8$

```
print(repr(LWE.primal_usvp(Kyber768, red_cost_model=RC.ADPS16))) # used in LWE.estimate.rough
print(repr(LWE.primal_usvp(Kyber768, red_cost_model=RC.MATZOV))) # used in LWE.estimate
```

```
rop: ≈2^182.2, red: ≈2^182.2, δ: 1.002902, β: 624, d: 1427, tag: usvp
rop: ≈2^204.9, red: ≈2^204.9, δ: 1.002902, β: 624, d: 1427, tag: usvp
```

Reading Estimator Output

```
LWE.primal_bdd(Kyber768, red_shape_model="CN11")
```

```
rop: ≈2^204.0, red: ≈2^203.1, svp: ≈2^202.8, β: 617, η: 651, d: 1457, tag: bdd
```

- **rop** elementary operations (”ring operations” for some reason)
- **red** elementary operations during lattice reduction
  - δ Root Hermite Factor
  - β BKZ block size
  - η dimension of final oracle call
  - d lattice dimension

Q: ”How do I . . .?”

SIS ”Easy, implement it and send us a patch.”[4]

Verify ”Easy, check the code and send us patches.”

[4] It is on the roadmap, though.

Solving SVP

Solving SVP

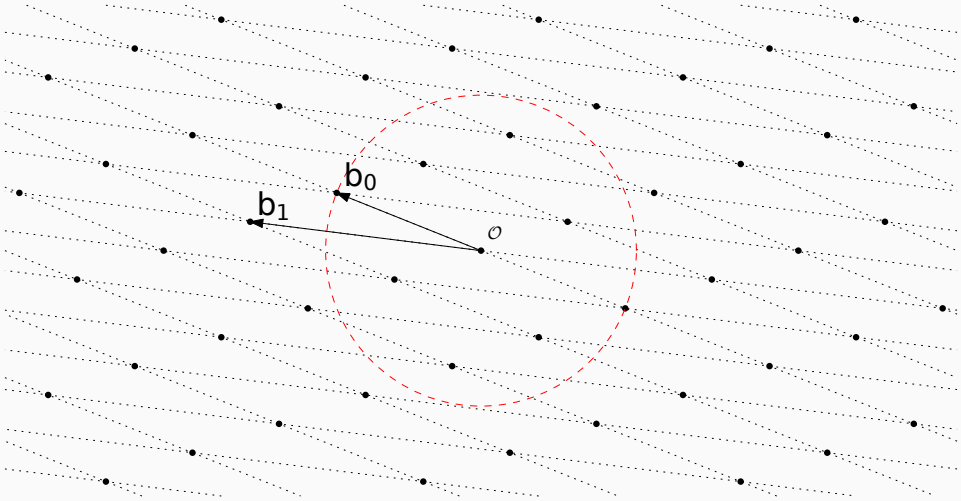
Enumeration

- Search through vectors smaller than a given bound: project down to 1-dim problem, lift to 2-dim problem ...
- Sensitive to the quality of the input basis
- **Time:** $2^{\Theta(\beta \log \beta)}$
- **Memory:** $\mathrm{poly}(\beta)$

Sieving

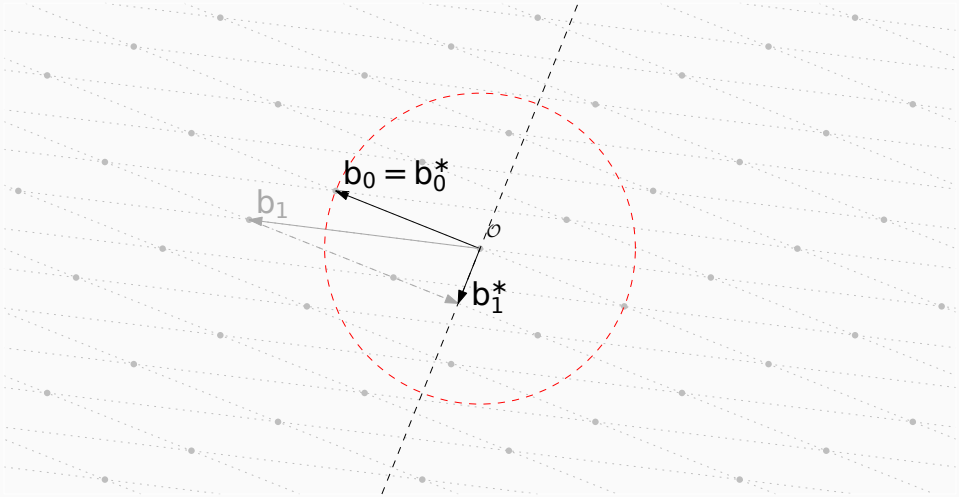
- Produce new, shorter vectors by considering sums and differences of existing vectors
- Fairly oblivious to the quality of the input basis
- **Time:** $2^{\Theta(\beta)}$
- **Memory:** $2^{\Theta(\beta)}$

Enumeration I – Pick a radius



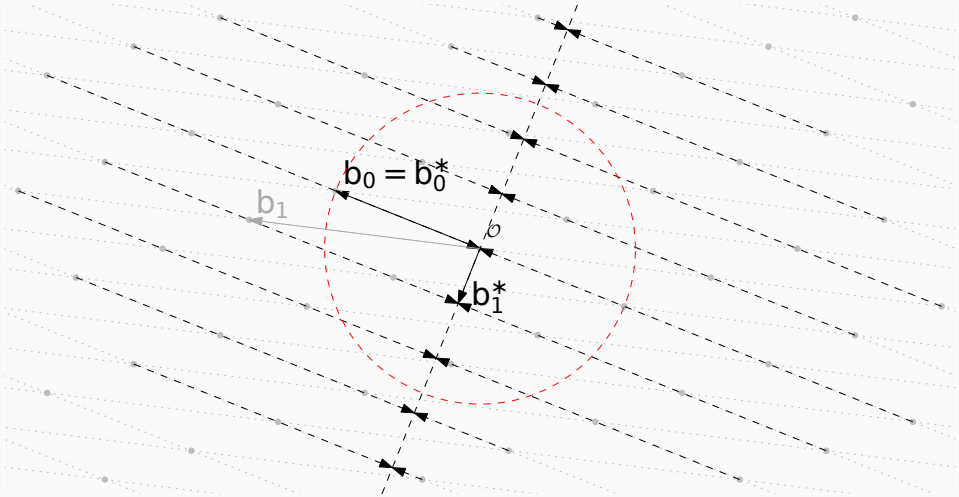
Picture credit: Joop van de Pol

Enumeration II – Project basis



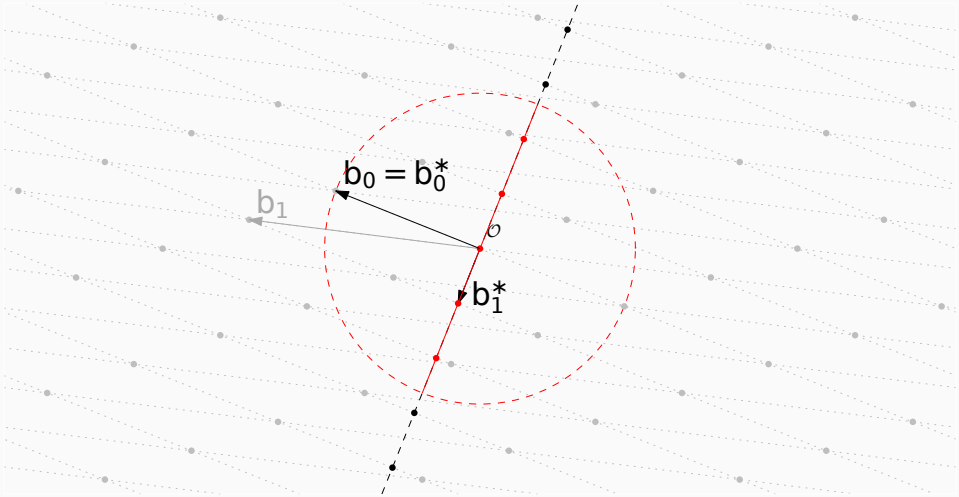
Picture credit: Joop van de Pol

Enumeration III – Project lattice



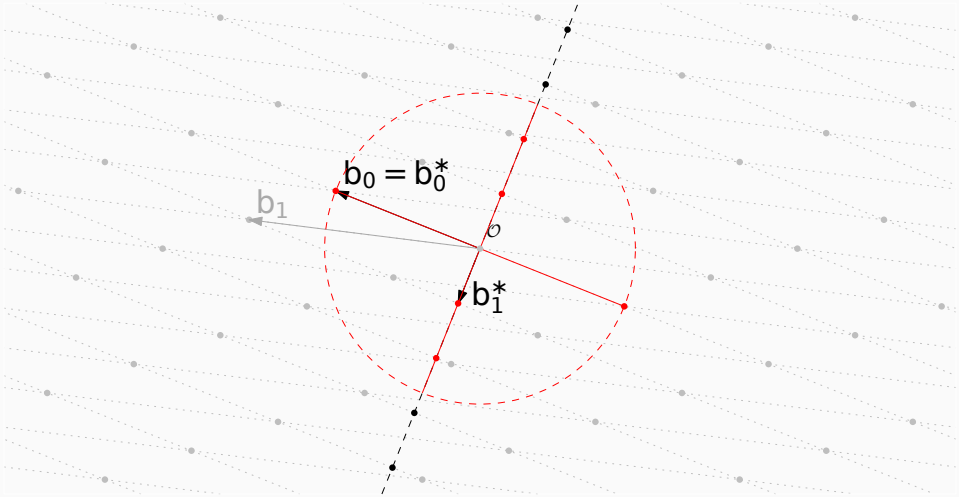
Picture credit: Joop van de Pol

Enumeration IV – Enumerate projections



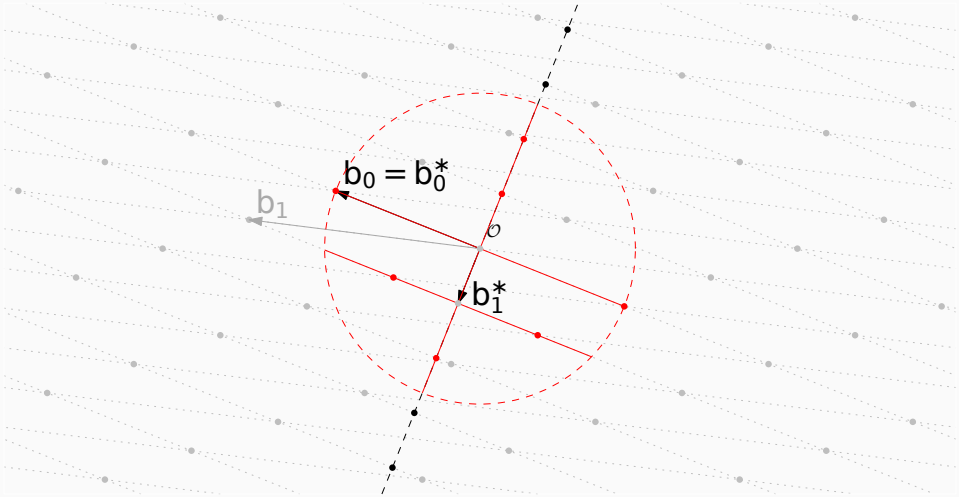
Picture credit: Joop van de Pol

Enumeration V – For each lift and enumerate



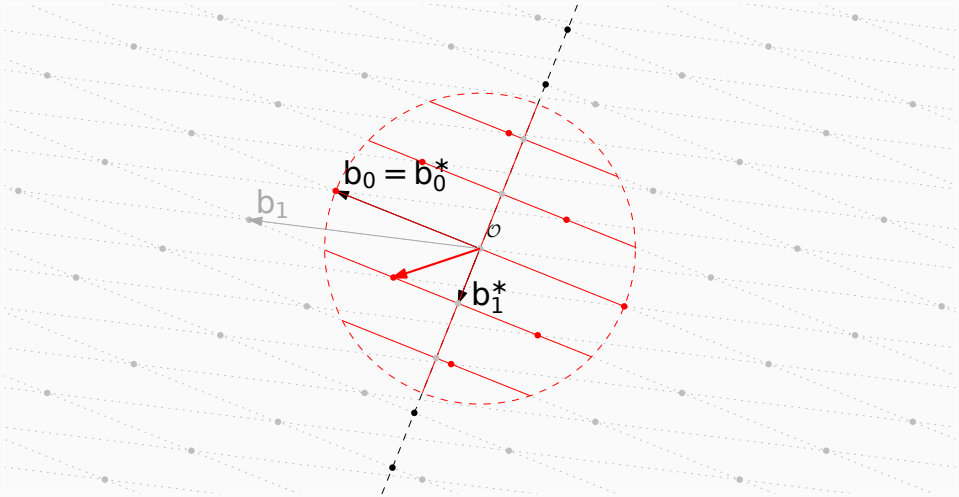
Picture credit: Joop van de Pol

Enumeration V – For each lift and enumerate



Picture credit: Joop van de Pol

Enumeration VI – Keep shortest

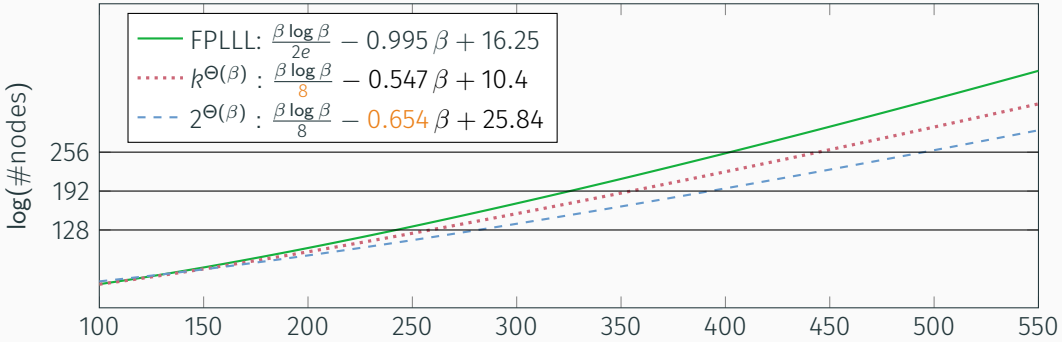


Picture credit: Joop van de Pol

Fast Enumeration

- Do not exhaust the search space, but focus on a fraction with exponentially small probability of success, repeat exponentially often: speed-up $2^{\Theta(\beta)}$
- Preprocess the basis with BKZ-β' for some $\beta' \leq \beta$ before enumerating.

Practical Performance (Simulation)

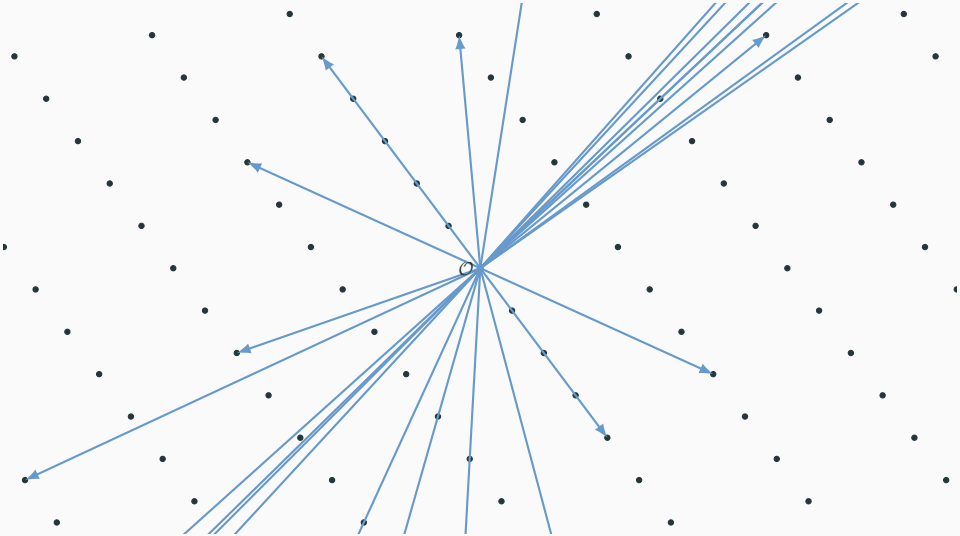



```
beta, d = 500, 1000
RC.CheNgu12(beta, d).log(2), RC.ABFKSW20(beta, d).log(2), RC.ABLR21(beta, d).log(2)
```

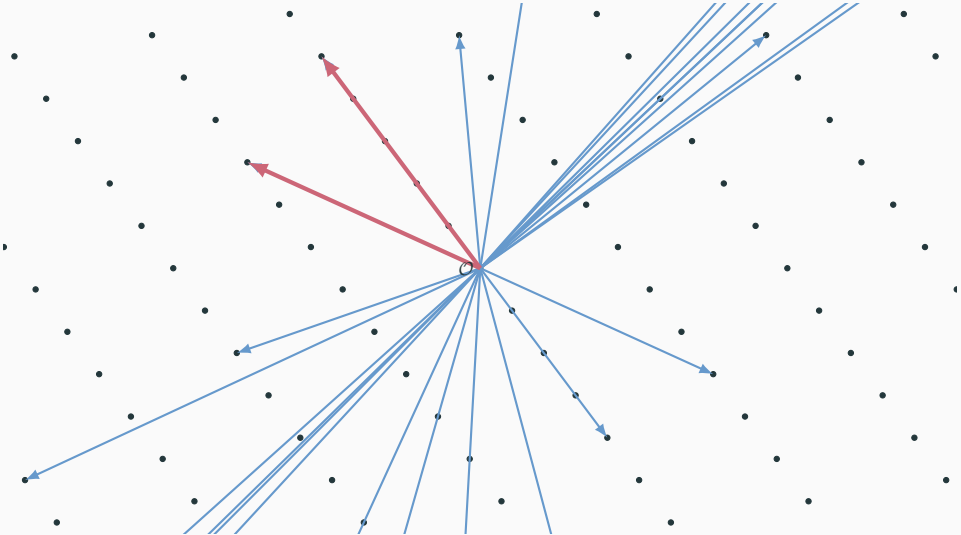
```
(365.668328064860, 316.227302076042, 278.167302076042)
```

[CN11; ABFKSW20; ABLR21]

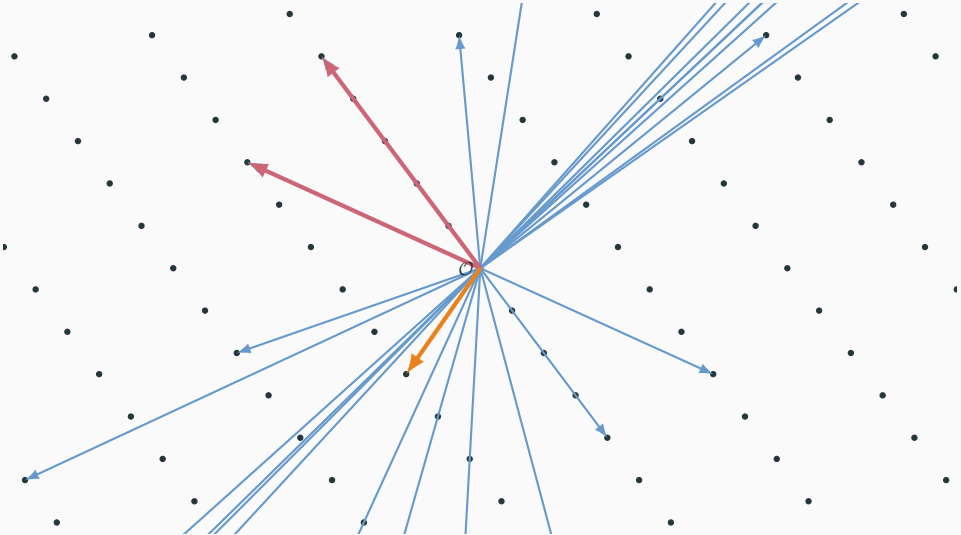
Sieving: Key Idea I



Sieving: Key Idea II



Sieving: Key Idea III



Sieving: Basic (Gauss) Sieve Complexity

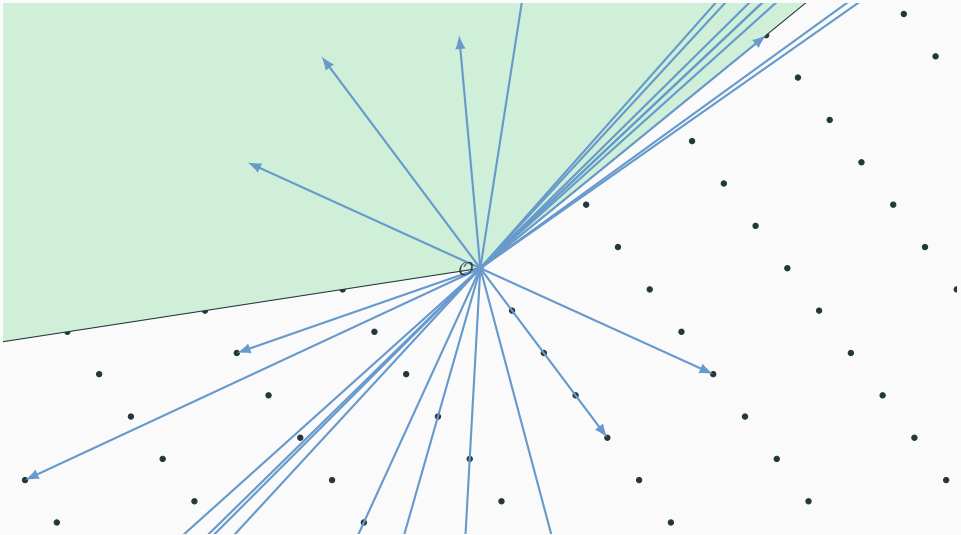
- Assume all vectors have (roughly) the same length
- Normalise to unit sphere $\mathcal{S}^{d-1} := \{\mathbf{x} \in \mathbb{R}^d | \|\mathbf{x}\| = 1\}$
- We have $\|\mathbf{v} - \mathbf{w}\| \leq 1$ iff $\langle \mathbf{v}, \mathbf{w} \rangle \geq 1/2 = \cos(\pi/3)$
- The probability that two random $\mathbf{v}, \mathbf{w} \in \mathcal{S}^{d-1}$ satisfy $\langle \mathbf{v}, \mathbf{w} \rangle \geq 1/2$ is

$$= \mathrm{poly}(d) \cdot \left(\frac{4}{3}\right)^{d/2} \approx 2^{0.2075\, d + o(d)}$$

- Need $\mathrm{poly}(d) \cdot \left(\frac{4}{3}\right)^{d/2}$ vectors, comparing all pairs costs $\mathrm{poly}(d) \cdot \left(\frac{4}{3}\right)^{d} \approx 2^{0.4150\, d + o(d)}$.

Daniele Micciancio and Panagiotis Voulgaris. Faster Exponential Time Algorithms for the Shortest Vector Problem. In: *21st SODA*. ed. by Moses Charika. ACM-SIAM, Jan. 2010, pp. 1468–1480. DOI: 10.1137/1.9781611973075.119

Sieving: Buckets I



Sieving: Buckets II

If $\mathbf{v}$, $\mathbf{c}$ are somewhat close and $\mathbf{w}$, $\mathbf{c}$ are somewhat close then perhaps $\mathbf{w}$, $\mathbf{v}$ are close?

Strategy

- Sort vectors into somewhat loose buckets,
- Do quadratic pairwise comparison only within each bucket.

BGJ Split search space into buckets. **Cost**: $2^{0.311\,\beta+o(\beta)}$.[5]

BDGL Use codes to decide which bucket to consider. **Cost**: $2^{0.292\,\beta+o(\beta)}$.[6]

[5] Anja Becker, Nicolas Gama, and Antoine Joux. Speeding-up lattice sieving without increasing the memory, using sub-quadratic nearest neighbor search. Cryptology ePrint Archive, Report 2015/522. https://eprint.iacr.org/2015/522. 2015.

[6] Anja Becker, Léo Ducas, Nicolas Gama, and Thijs Laarhoven. New directions in nearest neighbor searching with applications to lattice sieving. In: *27th SODA*. ed. by Robert Krauthgamer. ACM-SIAM, Jan. 2016, pp. 10–24. DOI: 10.1137/1.9781611974331.ch2.

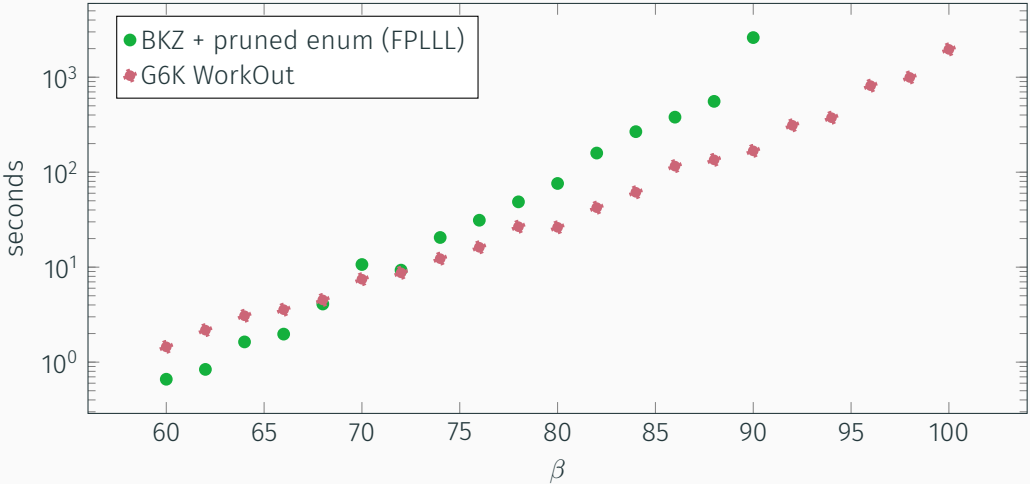
Sieving: G6K

G6K[7] is a Python/C++ framework for experimenting with sieving algorithms (inside and outside BKZ)

- Does not take the "oracle" view but considers sieves as stateful machines.
- Implements several sieve algorithms
  - Gauss and NV
  - Triple Sieve
  - BGJ1 (BGJ with one level of filtration)
  - BDGL (with one and two block respectively)
- Applies recent tricks and adds new tricks for improving performance of sieving

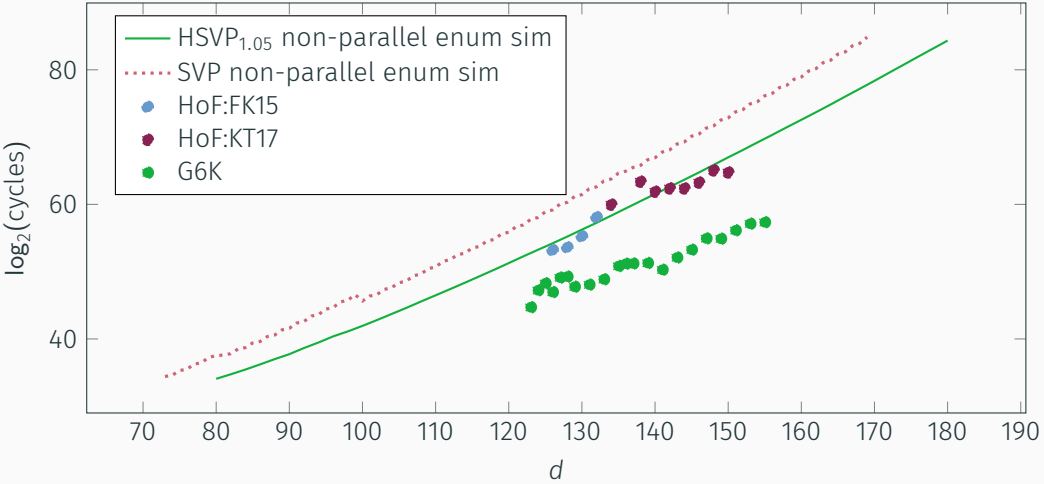
[7] Martin R. Albrecht, Léo Ducas, Gottfried Herold, Elena Kirshanova, Eamonn W. Postlethwaite, and Marc Stevens. The General Sieve Kernel and New Records in Lattice Reduction. In: *EUROCRYPT 2019, Part II*. ed. by Yuval Ishai and Vincent Rijmen. Vol. 11477. LNCS. Springer, Heidelberg, May 2019, pp. 717–746. DOI: 10.1007/978-3-030-17656-3_25.

Sieving: SVP



Average time in seconds for solving exact SVP

Darmstadt $HSVP_{1.05}$ Challenges



GPU Sieving

- Stream database of vectors to GPU
- Run low precision inner products there

dim	TD4F	D4F	MSD	Norm	Norm/GH	FLOP	Walltime	Mem GiB
158	31	29	129	3303	1.04329	262.1	9h 16m	89
162	31	31	131	3341	1.04220	263.2	18h 32m	156
176	34	33	143	3487	1.04412	267.5	12d 11h	806
178	34	32	146	3447	1.02725	268.6	22d 18h	1060
180	34	30	150	3509	1.04003	269.9	51d 14h	1443

Léo Ducas, Marc Stevens, and Wessel P. J. van Woerden. Advanced Lattice Sieving on GPUs, with Tensor Cores. In: *EUROCRYPT 2021, Part II*. ed. by Anne Canteaut and François-Xavier Standaert. Vol. 12697. LNCS. Springer, Heidelberg, Oct. 2021, pp. 249–279. DOI: 10.1007/978-3-030-77886-6_9

Try it at Home

```
from fpylll import IntegerMatrix, GSO, LLL
from fpylll.tools.bkz_stats import dummy_tracer
from g6k import Siever
from g6k.algorithms.bkz import pump_n_jump_bkz_tour

A = LLL.reduction(IntegerMatrix.random(180, "qary", k=90, bits=20))
g6k = Siever(A)

for b in range(20, 60+1, 10):
    pump_n_jump_bkz_tour(g6k, dummy_tracer, b, pump_params={"down_sieve": True})
```

https://github.com/fplll/g6k C++ kernel + Python frontend

https://github.com/WvanWoerden/G6K-GPU-Tensor G6K fork adding GPU support

Costing Sieves

"The main difference is the cost of the random product code decoding algorithm."

MATZOV. Report on the Security of LWE: Improved Dual Lattice Attack. Available at https://doi.org/10.5281/zenodo.6412487. Apr. 2022. DOI: 10.5281/zenodo.6412487. URL: https://doi.org/10.5281/zenodo.6412487

"Concretely, we conclude on an overhead factor of about on the number of gates in the RAM model compared to the idealized model for dimensions around after an appropriate re-parametrization."

Léo Ducas. Estimating the Hidden Overheads in the BDGL Lattice Sieving Algorithm. Cryptology ePrint Archive, Report 2022/922. https://eprint.iacr.org/2022/922. 2022

"Core-SVP" [ADPS16]: $2^{0.292\,\beta \pm 0}$ v [Sch+20; AGPS20] v [MAT22]

```
RC.ADPS16(500, 1000).log(2), RC.Kyber(500, 1000).log(2), RC.MATZOV(500, 1000).log(2)
```

```
(146.000000000000, 176.547704482770, 169.704298365530)
```

Quantum Stuff

Quantum Estimates

Sieving Given some vector $\mathbf{w}$ and a list of vectors L, apply Grover's algorithm to find $\{\mathbf{v} \in L \text{ s.t. } \|\mathbf{v} \pm \mathbf{w}\| \leq \|\mathbf{w}\|\}$.[8]

Enumeration Apply Montanaro's quantum backtracking algorithm for quadratic speed-up.[9]

[8] Thijs Laarhoven. Search problems in cryptography: From fingerprinting to lattice sieving. PhD thesis. Eindhoven University of Technology, 2015.

[9] Yoshinori Aono, Phong Q. Nguyen, and Yixin Shen. Quantum Lattice Enumeration and Tweaking Discrete Pruning. Cryptology ePrint Archive, Report 2018/546. `https://eprint.iacr.org/2018/546`. 2018.

Quantum Sieving

- A quantum sieve needs list of $2^{0.2075\beta}$ vectors before pairwise search with Grover
- Fast sieves use that the search is structured, Grover does unstructured search
  - Quantum Gauss Sieve

$$2^{(0.2075+\frac{1}{2}0.2075)\,\beta+o(\beta)} = 2^{0.311\,\beta+o(\beta)} \text{ time}, \qquad 2^{0.2075\,\beta+o(\beta)} \text{ memory}$$

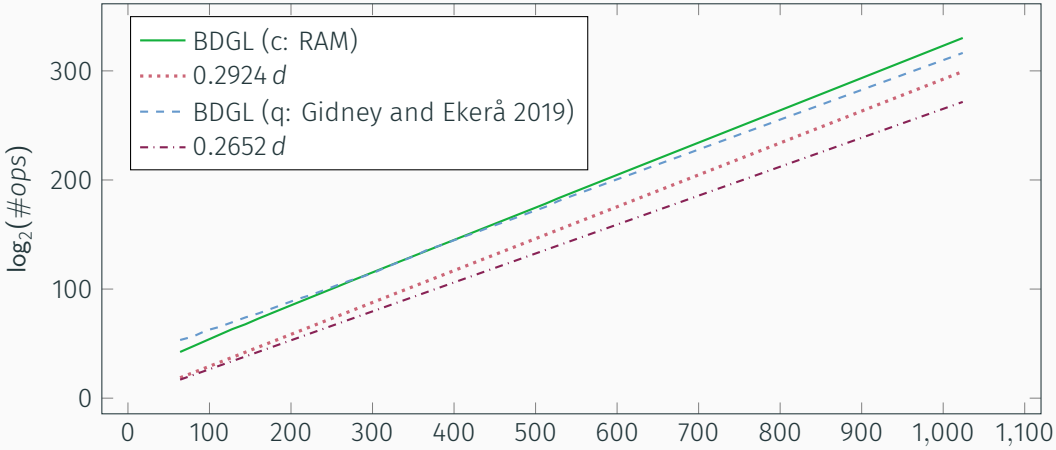
  - Classical BGJ Sieve[10]

$$2^{0.311\,\beta+o(\beta)} \text{ time}, \qquad 2^{0.2075\,\beta+o(\beta)} \text{ memory}$$

- Asymptotically fastest sieves have small lists and thus less Grover speed-up potential

[10] Anja Becker, Nicolas Gama, and Antoine Joux. Speeding-up lattice sieving without increasing the memory, using sub-quadratic nearest neighbor search. Cryptology ePrint Archive, Report 2015/522. https://eprint.iacr.org/2015/522. 2015.

Implementing Quantum Algorithms for SVP: Sieving (Underestimates)



Martin R. Albrecht, Vlad Gheorghiu, Eamonn W. Postlethwaite, and John M. Schanck. Estimating Quantum Speedups for Lattice Sieves. In: *ASIACRYPT 2020, Part II*. ed. by Shiho Moriai and Huaxiong Wang. Vol. 12492. LNCS. Springer, Heidelberg, Dec. 2020, pp. 583–613. DOI: 10.1007/978-3-030-64834-3_20

Quantum estimates do not seem to matter

- The resistance of post-quantum algorithms to quantum computers is routinely, e.g. in the NIST PQC Standardization Process, compared to that of the AES family of block ciphers.
- The state of the art is that AES-λ resists classical attacks of cost $\approx 2^{\lambda}$ and quantum attacks of cost $\approx 2^{\lambda/2}$, the latter being due to Grover's algorithm.[11]
- Parameters for post-quantum schemes are chosen such that they resist known classical attacks of cost $\approx 2^{\lambda}$ and known quantum attacks of cost $\approx 2^{\lambda/2}$ and any algorithm with complexity $\gg 2^{\lambda/2}$ will not affect the claimed security level.

[11] See [JNRV20] for more detailed cost estimates

Other Approaches

BKW combinatorial technique, relatively efficient for small secrets

Arora-Ge use Gröbner bases, asymptotically efficient , but large constants in the exponent

Rule of Thumb

Don't need to worry about these unless secret is unusually small (e.g. ternary) and/or sparse.

Thank You

From 2023 I will work for
King's College London and for SandboxAQ.

KCL will recruit academic staff, (maybe postdocs) and PhD students (not limited to post-quantum)

SandboxAQ will recruit staff, postdocs, PhD students and interns

References I

[ABFKSW20] Martin R. Albrecht, Shi Bai, Pierre-Alain Fouque, Paul Kirchner, Damien Stehlé, and Weiqiang Wen. Faster Enumeration-Based Lattice Reduction: Root Hermite Factor $k^{1/(2k)}$ Time $k^{k/8+o(k)}$. In: *CRYPTO 2020, Part II*. Ed. by Daniele Micciancio and Thomas Ristenpart. Vol. 12171. LNCS. Springer, Heidelberg, Aug. 2020, pp. 186–212. DOI: 10.1007/978-3-030-56880-1_7.

[ABLR21] Martin R. Albrecht, Shi Bai, Jianwei Li, and Joe Rowell. Lattice Reduction with Approximate Enumeration Oracles - Practical Algorithms and Concrete Performance. In: *CRYPTO 2021, Part II*. Ed. by Tal Malkin and Chris Peikert. Vol. 12826. LNCS. Virtual Event: Springer, Heidelberg, Aug. 2021, pp. 732–759. DOI: 10.1007/978-3-030-84245-1_25.

[ACPS09] Benny Applebaum, David Cash, Chris Peikert, and Amit Sahai. Fast Cryptographic Primitives and Circular-Secure Encryption Based on Hard Learning Problems. In: *CRYPTO 2009*. Ed. by Shai Halevi. Vol. 5677. LNCS. Springer, Heidelberg, Aug. 2009, pp. 595–618. DOI: 10.1007/978-3-642-03356-8_35.

[ADHKPS19] Martin R. Albrecht, Léo Ducas, Gottfried Herold, Elena Kirshanova, Eamonn W. Postlethwaite, and Marc Stevens. The General Sieve Kernel and New Records in Lattice Reduction. In: *EUROCRYPT 2019, Part II*. Ed. by Yuval Ishai and Vincent Rijmen. Vol. 11477. LNCS. Springer, Heidelberg, May 2019, pp. 717–746. DOI: 10.1007/978-3-030-17656-3_25.

[ADPS16] Erdem Alkim, Léo Ducas, Thomas Pöppelmann, and Peter Schwabe. Post-quantum Key Exchange - A New Hope. In: *USENIX Security 2016*. Ed. by Thorsten Holz and Stefan Savage. USENIX Association, Aug. 2016, pp. 327–343.

References II

[AGPS20] Martin R. Albrecht, Vlad Gheorghiu, Eamonn W. Postlethwaite, and John M. Schanck. Estimating Quantum Speedups for Lattice Sieves. In: *ASIACRYPT 2020, Part II*. Ed. by Shiho Moriai and Huaxiong Wang. Vol. 12492. LNCS. Springer, Heidelberg, Dec. 2020, pp. 583–613. DOI: 10.1007/978-3-030-64834-3_20.

[AGVW17] Martin R. Albrecht, Florian Göpfert, Fernando Virdia, and Thomas Wunderer. Revisiting the Expected Cost of Solving uSVP and Applications to LWE. In: *ASIACRYPT 2017, Part I*. Ed. by Tsuyoshi Takagi and Thomas Peyrin. Vol. 10624. LNCS. Springer, Heidelberg, Dec. 2017, pp. 297–322. DOI: 10.1007/978-3-319-70694-8_11.

[ANS18] Yoshinori Aono, Phong Q. Nguyen, and Yixin Shen. Quantum Lattice Enumeration and Tweaking Discrete Pruning. Cryptology ePrint Archive, Report 2018/546. https://eprint.iacr.org/2018/546. 2018.

[AS22] Martin R. Albrecht and Yixin Shen. Quantum Augmented Dual Attack. Cryptology ePrint Archive, Report 2022/656. https://eprint.iacr.org/2022/656. 2022.

[BDGL16] Anja Becker, Léo Ducas, Nicolas Gama, and Thijs Laarhoven. New directions in nearest neighbor searching with applications to lattice sieving. In: *27th SODA*. Ed. by Robert Krauthgamer. ACM-SIAM, Jan. 2016, pp. 10–24. DOI: 10.1137/1.9781611974331.ch2.

[BGJ15] Anja Becker, Nicolas Gama, and Antoine Joux. Speeding-up lattice sieving without increasing the memory, using sub-quadratic nearest neighbor search. Cryptology ePrint Archive, Report 2015/522. https://eprint.iacr.org/2015/522. 2015.

References III

[CN11] Yuanmi Chen and Phong Q. Nguyen. BKZ 2.0: Better Lattice Security Estimates. In: *ASIACRYPT 2011*. Ed. by Dong Hoon Lee and Xiaoyun Wang. Vol. 7073. LNCS. Springer, Heidelberg, Dec. 2011, pp. 1–20. DOI: 10.1007/978-3-642-25385-0_1.

[DSW21] Léo Ducas, Marc Stevens, and Wessel P. J. van Woerden. Advanced Lattice Sieving on GPUs, with Tensor Cores. In: *EUROCRYPT 2021, Part II*. Ed. by Anne Canteaut and François-Xavier Standaert. Vol. 12697. LNCS. Springer, Heidelberg, Oct. 2021, pp. 249–279. DOI: 10.1007/978-3-030-77886-6_9.

[Duc22] Léo Ducas. Estimating the Hidden Overheads in the BDGL Lattice Sieving Algorithm. Cryptology ePrint Archive, Report 2022/922. https://eprint.iacr.org/2022/922. 2022.

[JNRV20] Samuel Jaques, Michael Naehrig, Martin Roetteler, and Fernando Virdia. Implementing Grover Oracles for Quantum Key Search on AES and LowMC. In: *EUROCRYPT 2020, Part II*. Ed. by Anne Canteaut and Yuval Ishai. Vol. 12106. LNCS. Springer, Heidelberg, May 2020, pp. 280–310. DOI: 10.1007/978-3-030-45724-2_10.

[Laa15] Thijs Laarhoven. Search problems in cryptography: From fingerprinting to lattice sieving. PhD thesis. Eindhoven University of Technology, 2015.

[MAT22] MATZOV. Report on the Security of LWE: Improved Dual Lattice Attack. Available at https://doi.org/10.5281/zenodo.6412487. Apr. 2022. DOI: 10.5281/zenodo.6412487. URL: https://doi.org/10.5281/zenodo.6412487.

References iv

[MV10] Daniele Micciancio and Panagiotis Voulgaris. Faster Exponential Time Algorithms for the Shortest Vector Problem. In: *21st SODA*. Ed. by Moses Charika. ACM-SIAM, Jan. 2010, pp. 1468–1480. DOI: 10.1137/1.9781611973075.119.

[PV21] Eamonn W. Postlethwaite and Fernando Virdia. On the Success Probability of Solving Unique SVP via BKZ. In: *PKC 2021, Part I*. Ed. by Juan Garay. Vol. 12710. LNCS. Springer, Heidelberg, May 2021, pp. 68–98. DOI: 10.1007/978-3-030-75245-3_4.

[Sch03] Claus-Peter Schnorr. Lattice Reduction by Random Sampling and Birthday Methods. In: *STACS 2003, 20th Annual Symposium on Theoretical Aspects of Computer Science, Berlin, Germany, February 27 - March 1, 2003, Proceedings*. Ed. by Helmut Alt and Michel Habib. Vol. 2607. Lecture Notes in Computer Science. Springer, 2003, pp. 145–156. DOI: 10.1007/3-540-36494-3_14. URL: http://dx.doi.org/10.1007/3-540-36494-3_14.

[Sch+20] Peter Schwabe et al. CRYSTALS-KYBER. Tech. rep. available at https://csrc.nist.gov/projects/post-quantum-cryptography/round-3-submissions. National Institute of Standards and Technology, 2020.