Random Sampling Revisited: Lattice Enumeration with Discrete Pruning

Yoshinori Aono
NICT

Phong Nguyễn
Inria — Inventors for the digital world
CNRS
東京大学 The University of Tokyo

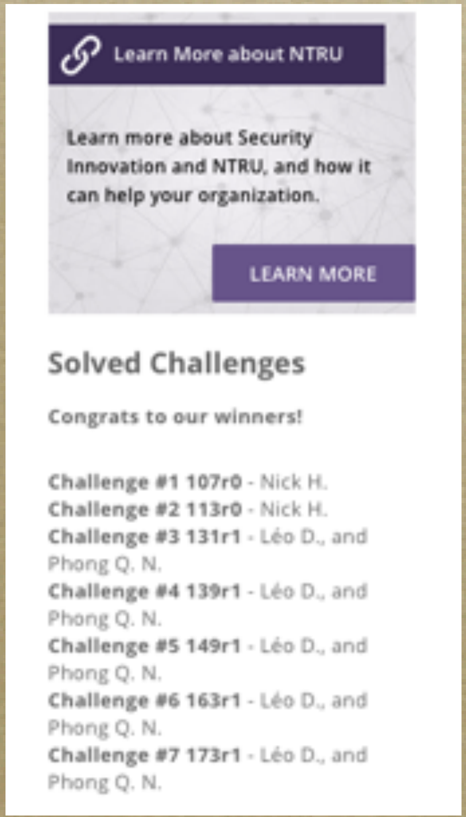
Summary

- Motivation
- Lattices, Enumeration and Pruning
- Enumeration with Discrete Pruning

Motivation

Context

- Needs: convincing security estimates for lattice-based cryptosystems.
- Sanity check: lattice challenges.



Learn More about NTRU

Learn more about Security Innovation and NTRU, and how it can help your organization.

LEARN MORE

Solved Challenges

Congrats to our winners!

Challenge #1 107r0 - Nick H.
Challenge #2 113r0 - Nick H.
Challenge #3 131r1 - Léo D., and Phong Q. N.
Challenge #4 139r1 - Léo D., and Phong Q. N.
Challenge #5 149r1 - Léo D., and Phong Q. N.
Challenge #6 163r1 - Léo D., and Phong Q. N.
Challenge #7 173r1 - Léo D., and Phong Q. N.

TU DARMSTADT LATTICE CHALLENGE

INTRODUCTION

Welcome to the lattice challenge.

Building upon a popular paper by Ajtai [1], we have constructed lattice
solution of SVP implies a solution of SVP in all lattices of a certain smaller
not mean that one can solve all instances simultaneously, but rather that
the worst case instances. We think these lattice bases are hard instances
test and compare modern lattice reduction algorithms.

We show how these lattice bases were constructed and prove the existence
each of the corresponding lattices in [2]. We challenge everyone to try
a short vector. There are two ways to enter the hall of fame:

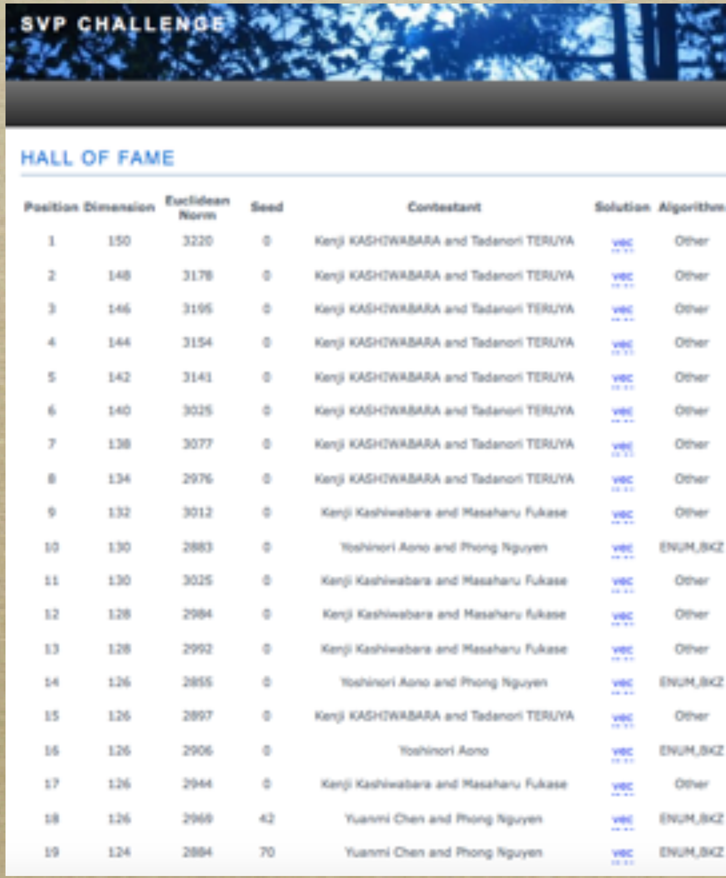
- Tackle a challenge dimension that nobody succeeded in before;
- Find an even shorter vector in one of the dimensions listed in the hall of

References

1. Ajtai: Generating Hard Instances of Lattice Problems, STOC 1996
2. Buchmann, Lindner, Rückert: Explicit Hard Instances of the Shortest
PQCrypto 2008

HALL OF FAME

Position	Dimension	Euclidean norm	Contestant
1	825	117.64	Yoshinori Aono Phong Nguyen
2	800	103.95	Yoshinori Aono Phong Nguyen
3	775	100.14	Yuanmi Chen Phong Nguyen
4	750	87.76	Yuanmi Chen Phong Nguyen
5	725	80.65	Yuanmi Chen Phong Nguyen



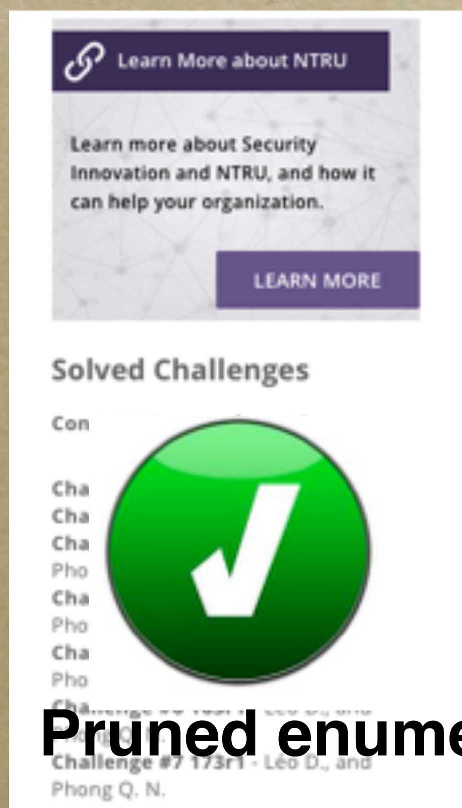
SVP CHALLENGE

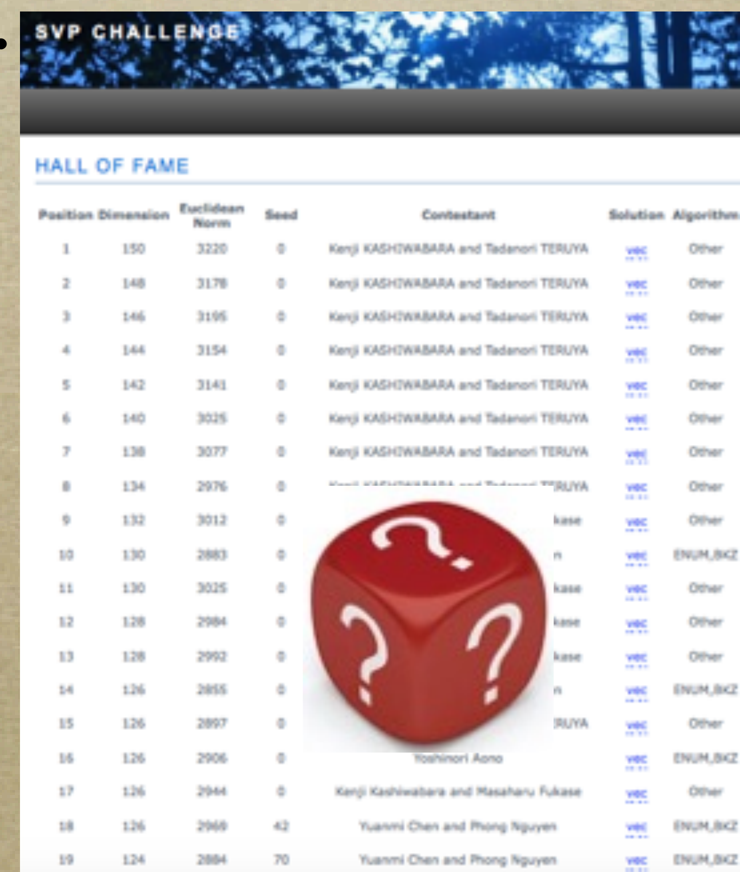
HALL OF FAME

Position	Dimension	Euclidean Norm	Seed	Contestant	Solution	Algorithm
1	150	3220	0	Kenji KASHIWABARA and Tadanori TERUYA	vec	Other
2	148	3178	0	Kenji KASHIWABARA and Tadanori TERUYA	vec	Other
3	146	3195	0	Kenji KASHIWABARA and Tadanori TERUYA	vec	Other
4	144	3154	0	Kenji KASHIWABARA and Tadanori TERUYA	vec	Other
5	142	3141	0	Kenji KASHIWABARA and Tadanori TERUYA	vec	Other
6	140	3025	0	Kenji KASHIWABARA and Tadanori TERUYA	vec	Other
7	138	3077	0	Kenji KASHIWABARA and Tadanori TERUYA	vec	Other
8	134	2976	0	Kenji KASHIWABARA and Tadanori TERUYA	vec	Other
9	132	3012	0	Kenji Kashiwabara and Masaharu Fukase	vec	Other
10	130	2883	0	Yoshinori Aono and Phong Nguyen	vec	ENUM,BKZ
11	130	3025	0	Kenji Kashiwabara and Masaharu Fukase	vec	Other
12	128	2984	0	Kenji Kashiwabara and Masaharu Fukase	vec	Other
13	128	2992	0	Kenji Kashiwabara and Masaharu Fukase	vec	Other
14	126	2855	0	Yoshinori Aono and Phong Nguyen	vec	ENUM,BKZ
15	126	2897	0	Kenji KASHIWABARA and Tadanori TERUYA	vec	Other
16	126	2906	0	Yoshinori Aono	vec	ENUM,BKZ
17	126	2944	0	Kenji Kashiwabara and Masaharu Fukase	vec	Other
18	126	2969	42	Yuanmi Chen and Phong Nguyen	vec	ENUM,BKZ
19	124	2884	70	Yuanmi Chen and Phong Nguyen	vec	ENUM,BKZ

Context

- Needs: convincing security estimates for lattice-based cryptosystems.
- Sanity check: lattice challenges.





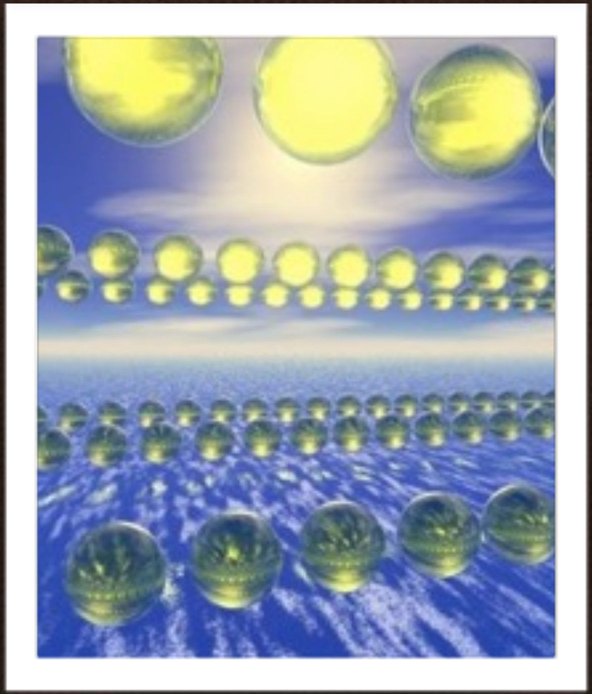
Pruned enumeration with BKZ

What Happened?

- The largest SVP records [KaTe,KaFu] use significant power (≈RSA-768) and a « secret » algorithm: partial description in [FuKa15].
- The main tool is an improved variant of Schnorr's Random Sampling [Sc03]: not well-understood.

Our Results

- Revisit Schnorr's Random Sampling [Sc03] and variants [BuLu06,FuKa15,DZW15].
  - Geometric description/generalization
  - First sound analysis: previously, gap between analyses and experiments.
  - Optimal parameters.
- Unify Random Sampling and an older algorithm: pruned enumeration [ScEu94,ScHo95,GNR10]

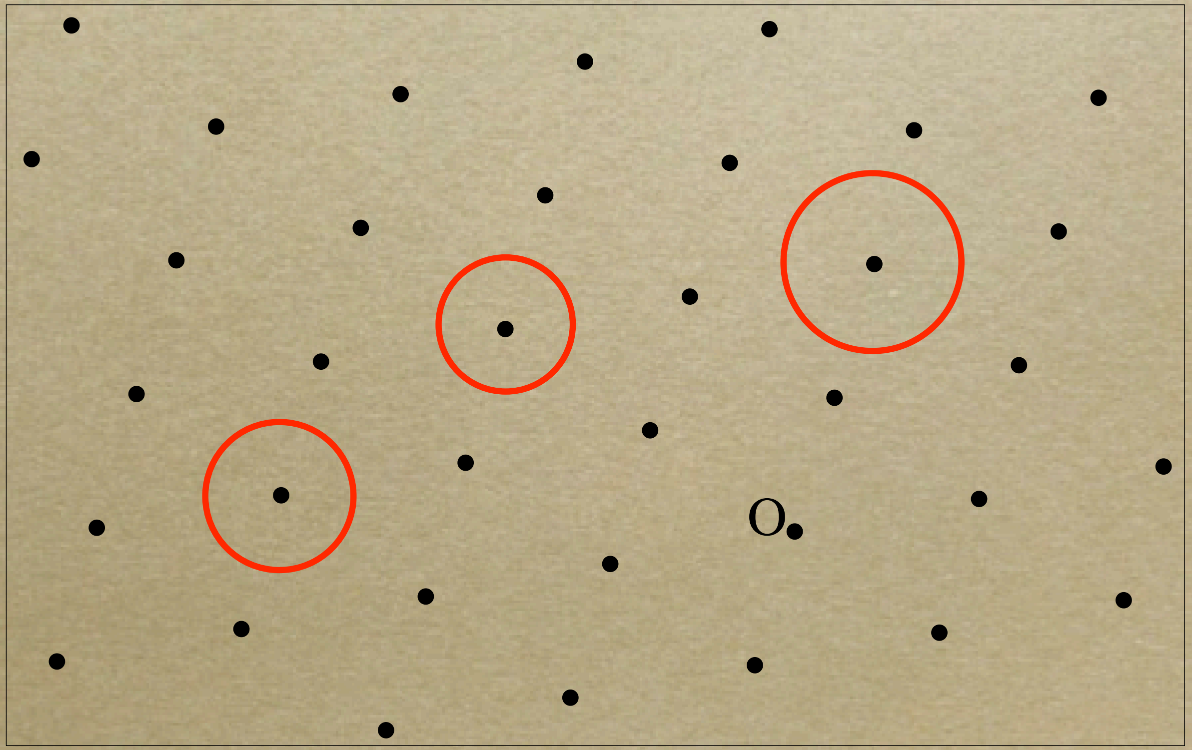


Background

What is a Lattice?

- A lattice is a discrete subgroup of $\mathbf{R}^n$, or the set $L(b_1,\ldots,b_d)$ of all linear combinations $\sum x_i b_i$ where $x_i \in \mathbf{Z}$, and the b_i's are linearly independent.

2	0	0	0	0
0	2	0	0	0
0	0	2	0	0
0	0	0	2	0
1	1	1	1	1



Hard Lattice Problems

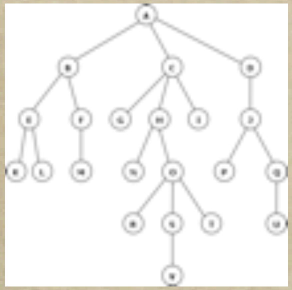
- Input: a lattice L and an n-dim ball C.
- Output: decide if $L \cap C$ is non-trivial, and find a point when applicable.
- Two settings

  - Approx: $L \cap C$ has many points. Ex: SIS and ISIS.

  - Unique: only one non-trivial point. Ex: BDD.

Enumeration

- The simplest method to solve hard lattice problems, going back to the 70s.
- Input: a lattice L and a small ball $S \subseteq \mathbf{R}^n$ s.t. $\#(L \cap S)$ is « small ».
- Output: All points in $L \cap S$.
- Drawback: running-time typically superexponential, much larger than $\#(L \cap S)$.



Enumeration Insight

- Key ideas:
  - Projections never increase norms: if $||v||\leq R$, then $||\pi(v)||\leq R$.
  - Using nice subspaces, π(lattice) is a lower-dim lattice.
- Enumeration is a depth-first search of a gigantic tree, whose running time depends on the quality of the basis.

Speeding Up Enumeration by Pruning

Speeding Up Enumeration

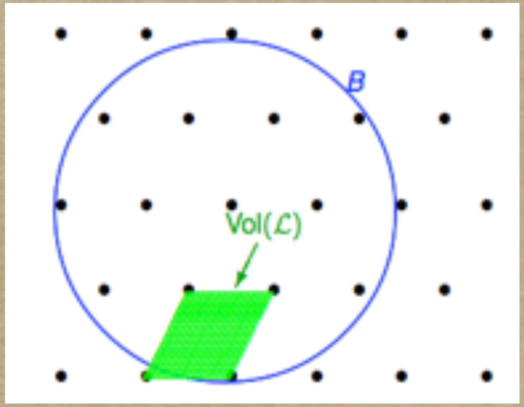
- Assume that we do not need all $L \cap S$:
  - Can we make enumeration faster if we only need to find one vector?

Enumeration with Pruning [ScEu94,ScHo95,GNR10]

- Input: a lattice L, a ball $S \subseteq \mathbf{R}^n$ and a pruning set $P \subseteq \mathbf{R}^n$.
- Output: All points in $L \cap S \cap P = (L \cap P) \cap S$.
- Pros: Enumerating $L \cap S \cap P$ can be much faster than $L \cap S$.
- Cons: Maybe $L \cap S \cap P \subseteq \{0\}$.

Analyzing Pruned Enumeration [GNR10] Framework

- Enumerating L∩S∩P is deterministic, but:
  - The set P is randomized: it depends on a (random) reduced basis.
  - The success probability is Pr(L∩S∩P ⊄ {0}).
- #(L∩S∩P) « should be » ≈vol(S∩P)/covol(L) (Gaussian heuristic).

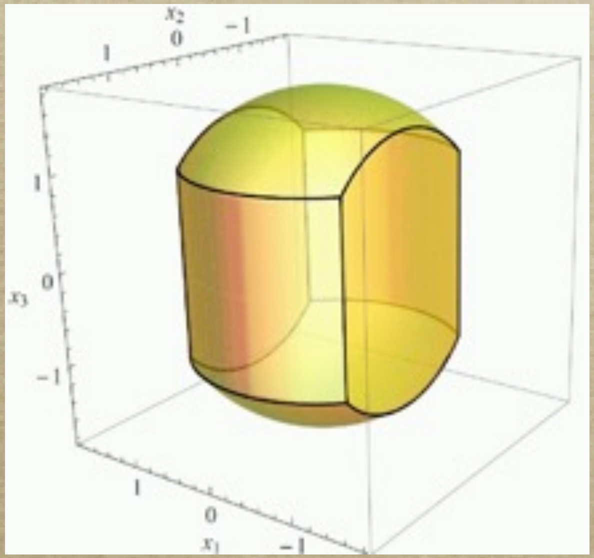


Extreme Pruning [GNR10]

- Repeat until success
  - Generate P by reducing a "random" basis.
  - Enumerate($L \cap S \cap P$)
- Can be much faster than enumeration, even if $\Pr(L \cap S \cap P \not\subseteq \{0\})$ is tiny.

Two Kinds of Pruning

- Cylinder Pruning ([GNR10] generalizing [ScEu94,ScHo95]): P is a cylinder intersection.



- Discrete Pruning (today): P is a union of cells, in practice a union of many boxes.

Enumeration with Discrete Pruning

Insight

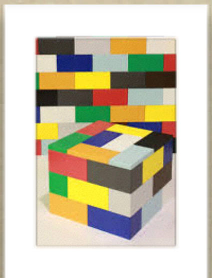
- Previous analyses of Random Sampling studied the distribution of certain lattice points (based on encodings): tricky!
- New point of view: it's actually about partitioning the n-dim space.
  - Description
  - Analysis

Lattice Partitions

- Any partition of $\mathbf{R}^n = \cup_{t \in T} C(t)$ into countably many cells s.t.:
  - cells are disjoint: $C(i) \cap C(j) = \varnothing$
  - each cell can be « opened » : it contains one and only one lattice point, which can be found efficiently. Given a tag $t \in T$, one can compute $L \cap C(t)$.

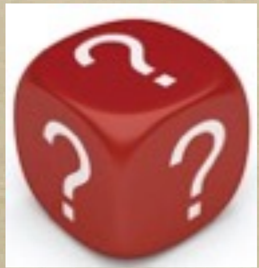
Intuitively

- $\mathrm{Enum}(L \cap C(t))$
  $\simeq$ Egg opening



Lattice Enumeration with Discrete Pruning

- Repeat until success
  - Select $P = \cup_{t \in U} C(t)$ for some finite $U \subseteq T$.
  - Enumerate$(L \cap S \cap P)$ by enumerating all $C(t) \cap L$ where $t \in U$.
- Each iteration takes $\#U$ poly-time operations and succeeds with $\Pr(L \cap S \cap P \not\subseteq \{0\})$.
  - We need to calculate $\mathrm{vol}(S \cap P) = \Sigma_{t \in U} \mathrm{vol}(S \cap C(t))$.
  - Time(Enum$(L \cap P)$) « linear » in $\#(L \cap P)$.



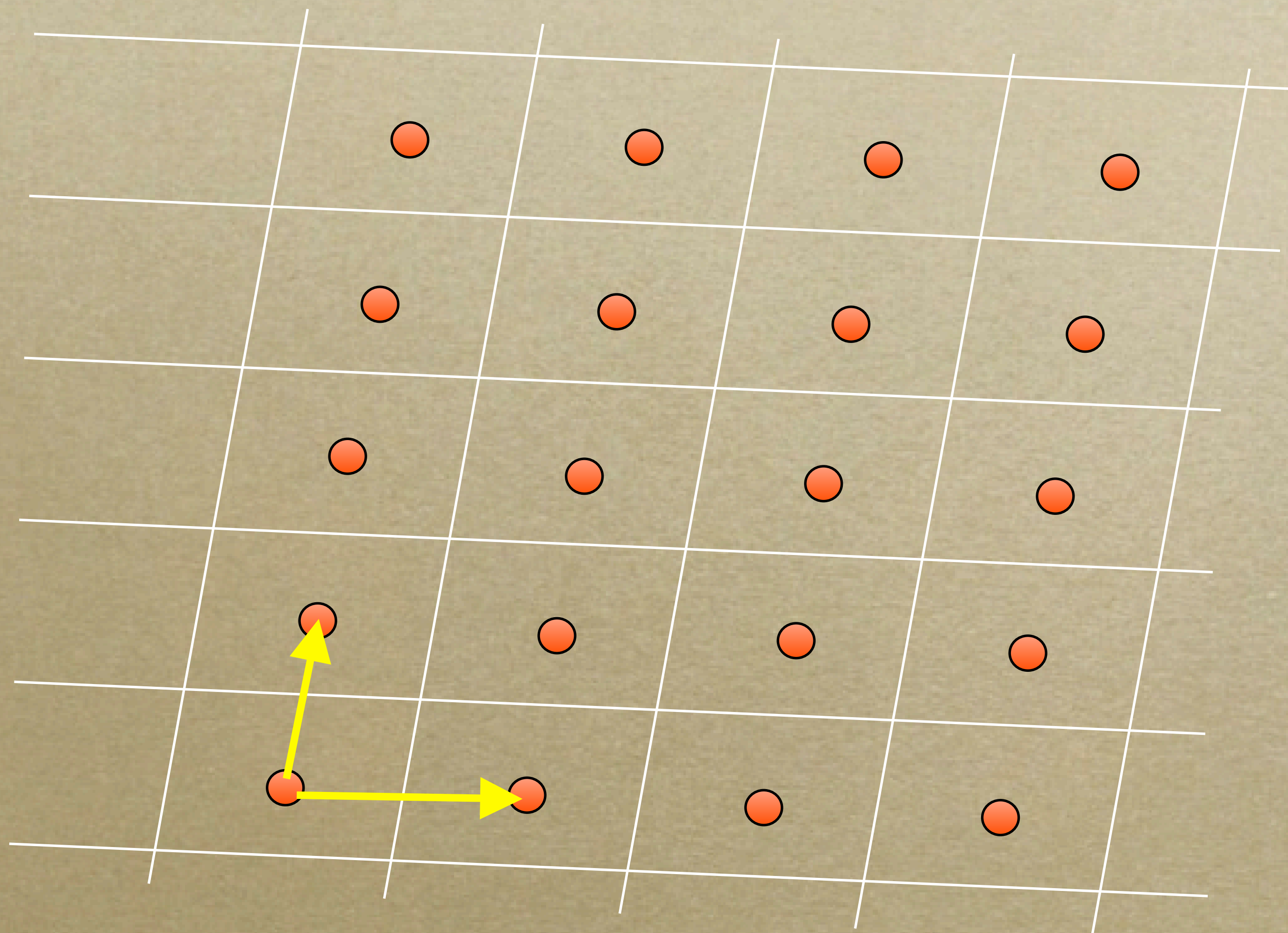
Issues

- Which lattice partition?
- How to compute $vol(S \cap C(t))$?
  To deduce $vol(S \cap P) = \sum_{t \in U} vol(S \cap C(t))$
- How to select the set U of tags?
  We'd like the ones maximizing $vol(S \cap C(t))$: different from [Sc03,FK15].

A) Which Lattice Partitions?

- Lattice partitions from fundamental domains: $T=\mathbf{Z}^n$.
- Lattice partitions using boxes
  - Babai's partition, implicit in [DZW15]: $T=\mathbf{Z}^n$.
  - The natural partition, implicit in [FK15]: $T=\mathbf{N}^n$.

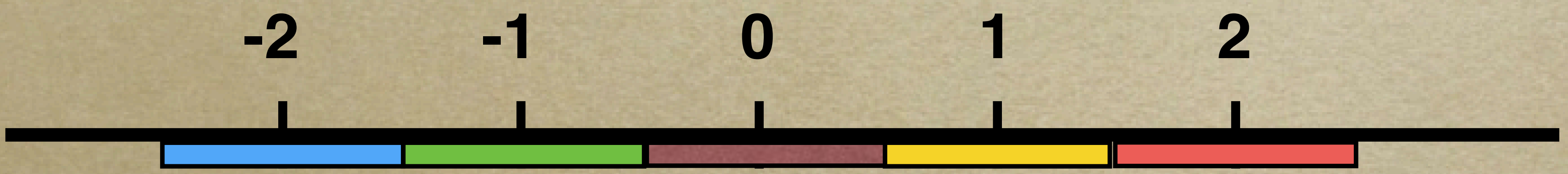
Trivial Lattice Partitions



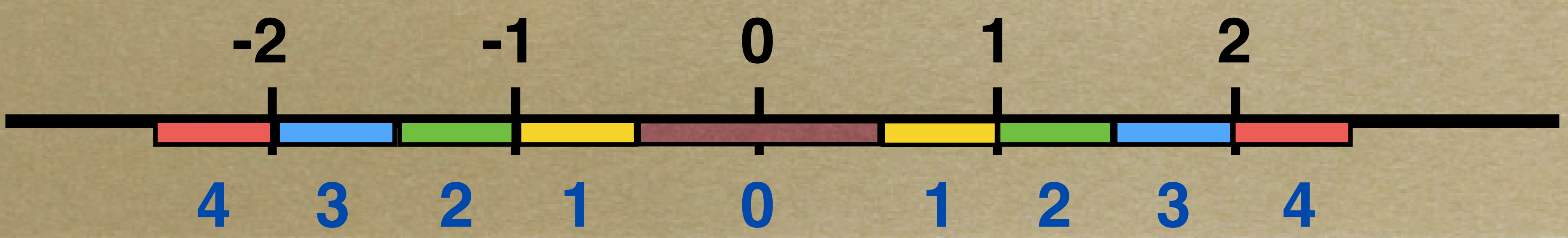
- $T=\mathbf{Z}^n$. Cell opening: matrix/vector product.

Box Partitions in Dimension 1

- Babai's partition: T=**Z**



- The natural partition: T=**N**

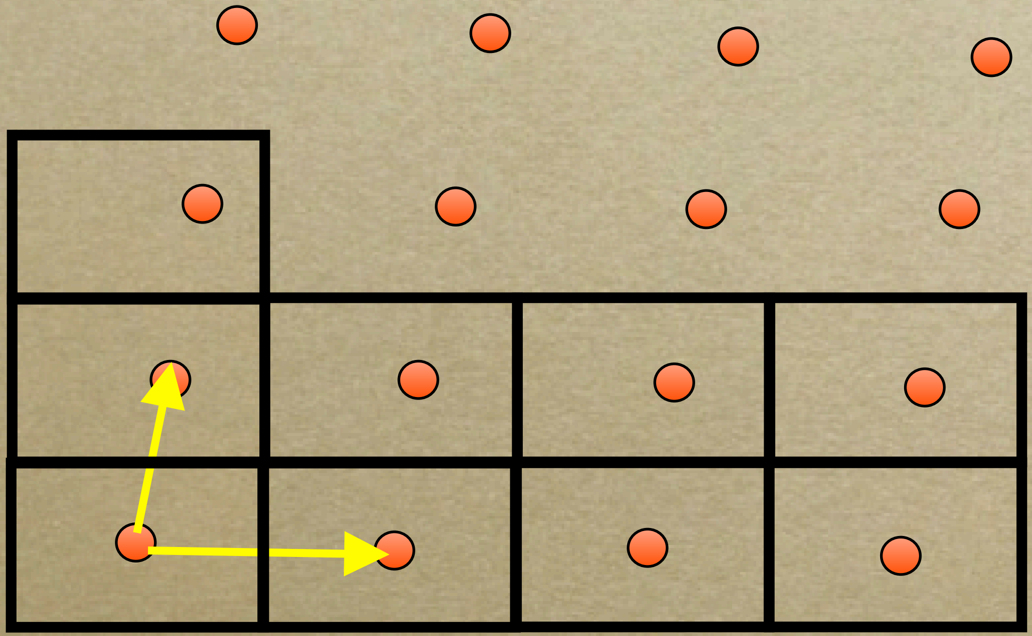


Dimension n

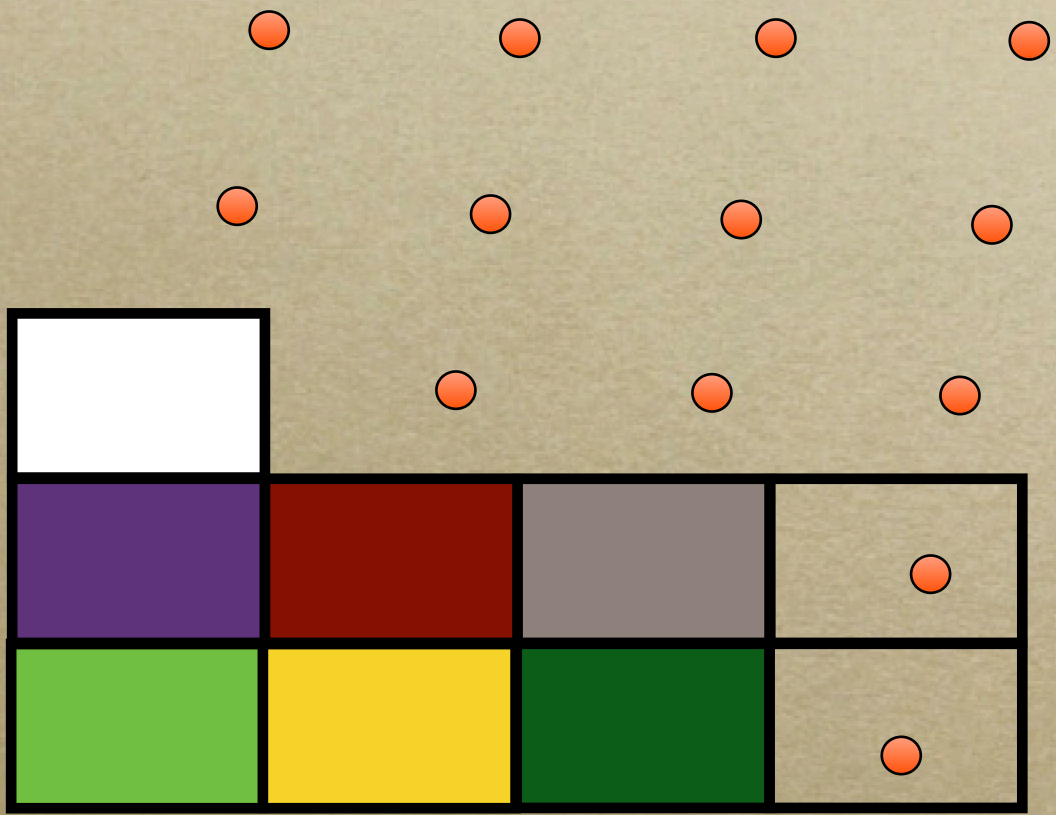
- We can generalize with projections.
  - Let $b_1,\ldots,b_n \in \mathbf{R}^m$.
  - Its Gram-Schmidt Orthogonalization is $b^*_1,\ldots,b^*_n \in \mathbf{R}^m$:
    - $b^*_1 = b_1$
    - b^*_i = component of b_i orthogonal to $b_1,\ldots,b_{i-1}$.


Babai's partition

- $T = \mathbf{Z}^n$ and $C(t) = tB^* + \{\sum_i x_i b^*_i$ s.t. $-1/2 \le x_i < 1/2\}$.
- Cell opening: Babai's algorithm [Ba86].



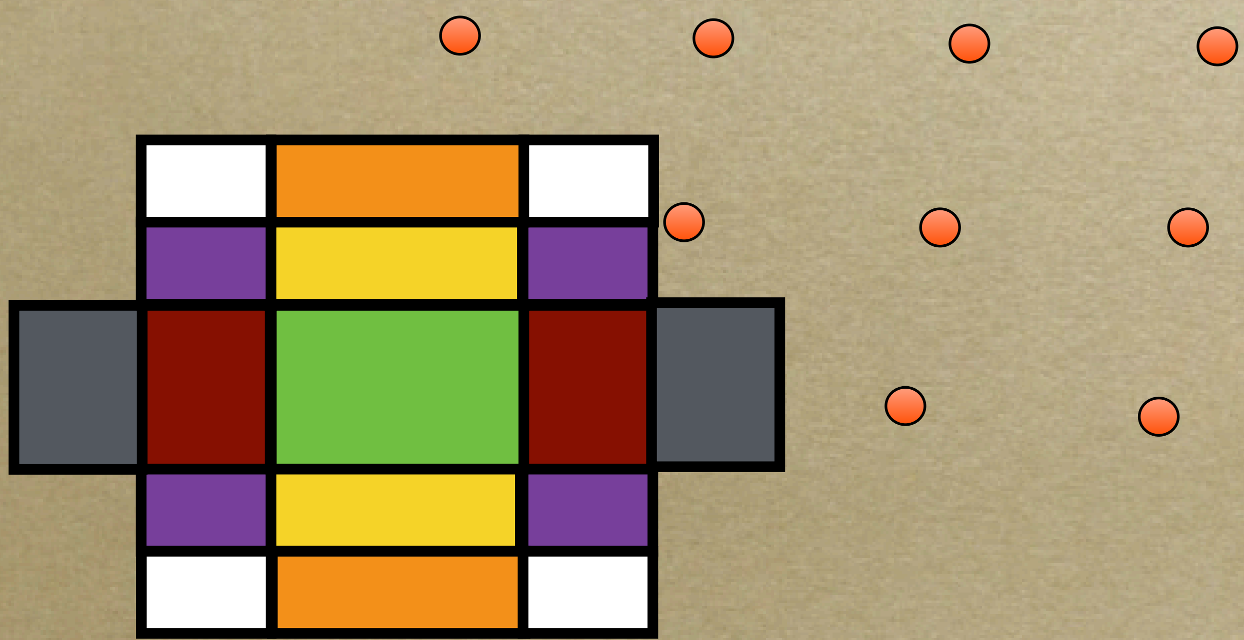
Babai's partition



The « Natural » Partition

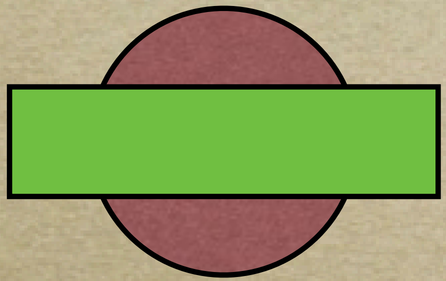
- $T=\mathbf{N}^n$ and $C((t_1,\ldots,t_n))$ is $\{\sum_i x_i b^*_i$ s.t. $-(t_j+1)/2<x_j\leq -t_j/2$ or $t_j/2<x_j\leq(t_j+1)/2\}$

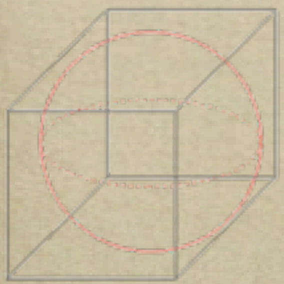
- Cell opening: variant of Babai's algorithm.



B) Intersection Volumes

- To estimate the success probability, we need to approximate vol($S \cap C(t)$) for many t's where:
  - S is a ball
  - C(t) is a box, or a union of symmetric boxes.





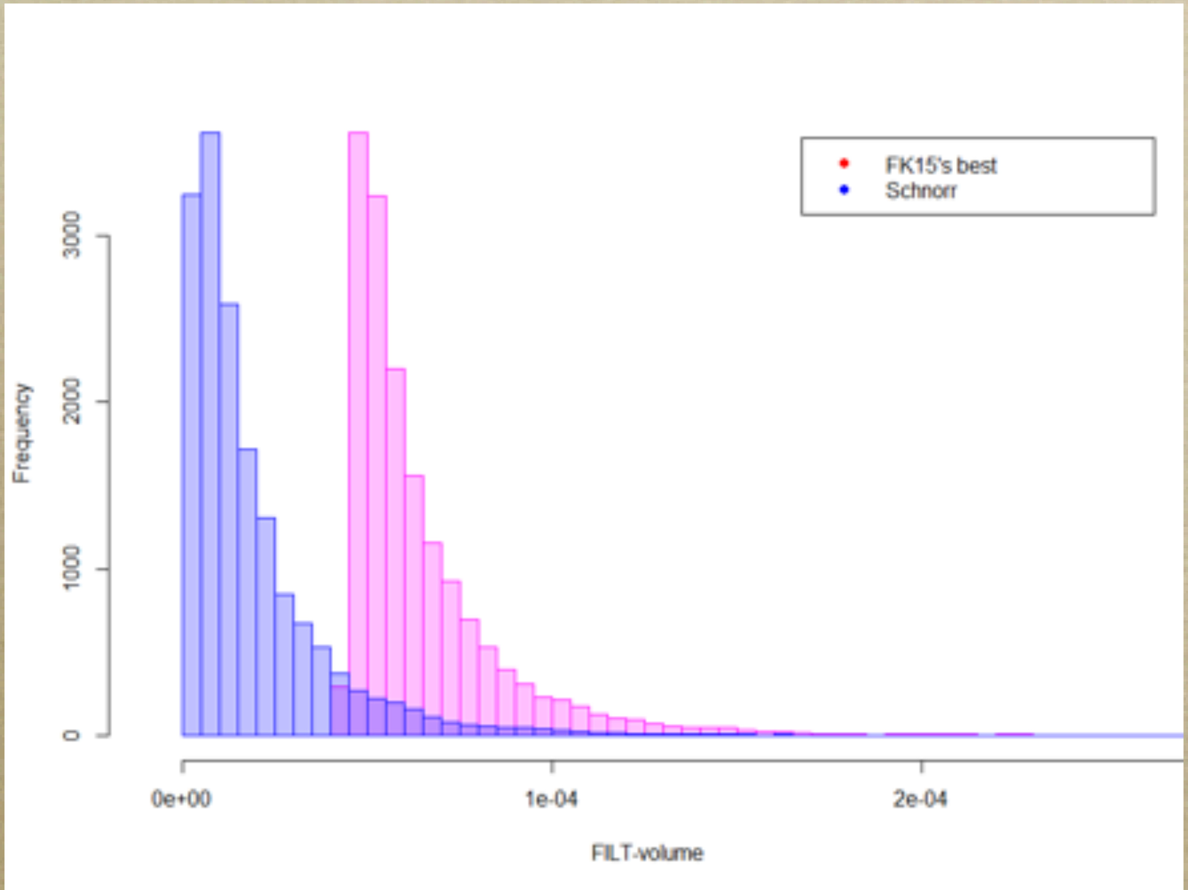
Ball-Box Intersections

- Let S=unit-ball and $H = \Pi_i [\alpha_i, \beta_i]$ be a box.

  Compute vol(S∩H).

- We give:
  - Asymptotic formula for balanced boxes using the Central Limit Theorem.
  - Two infinite-series formulas by generalizing [CoTi1997] (Fourier analysis).
  - Practical method using [Hosono81]'s Fast Inverse Laplace Transform.

Application: [Schnorr03] vs [FK15]

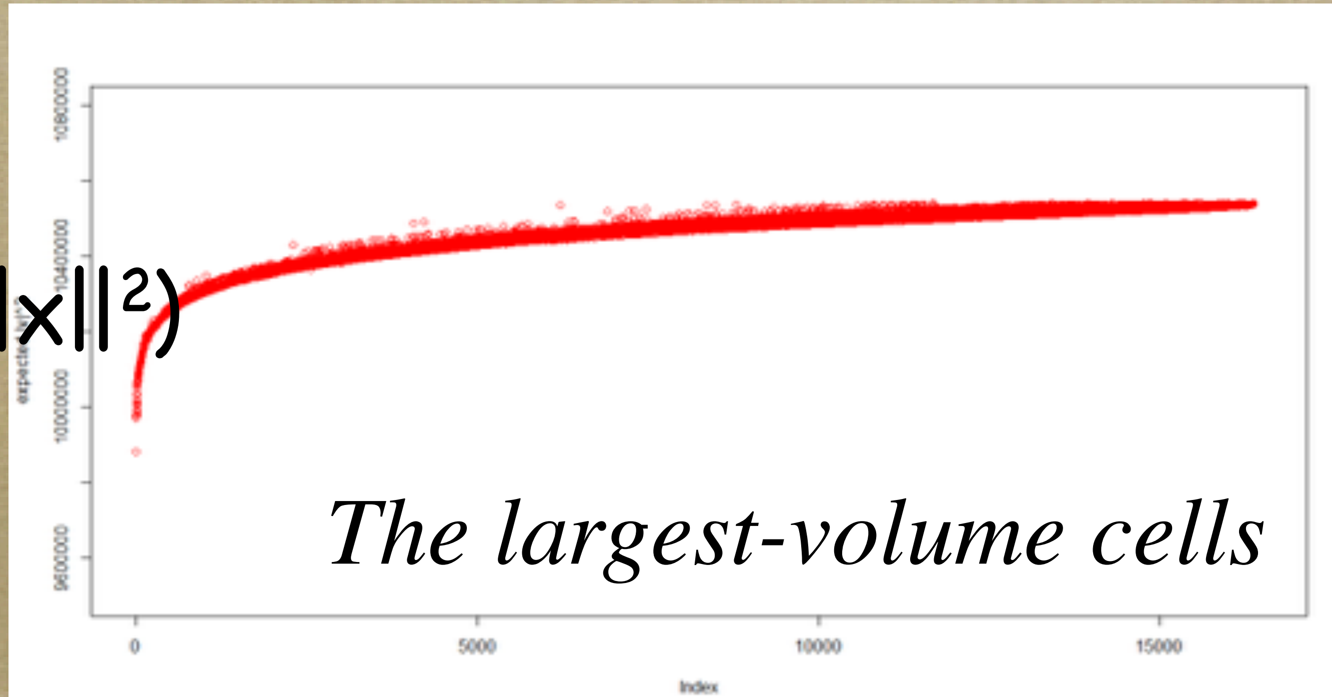


Distribution of vol(S∩C(i)):
[FK15] cells have larger intersection volume.

C) Which Cells?

- The computation of vol(S∩C(t)) is too « slow » to find the cells with largest vol(S∩C(t)).
- But it is easy to find the cells C(t) minimizing $E_{x \in C(t)}(||x||^2)$: orthogonal enumeration. Almost the same cells!

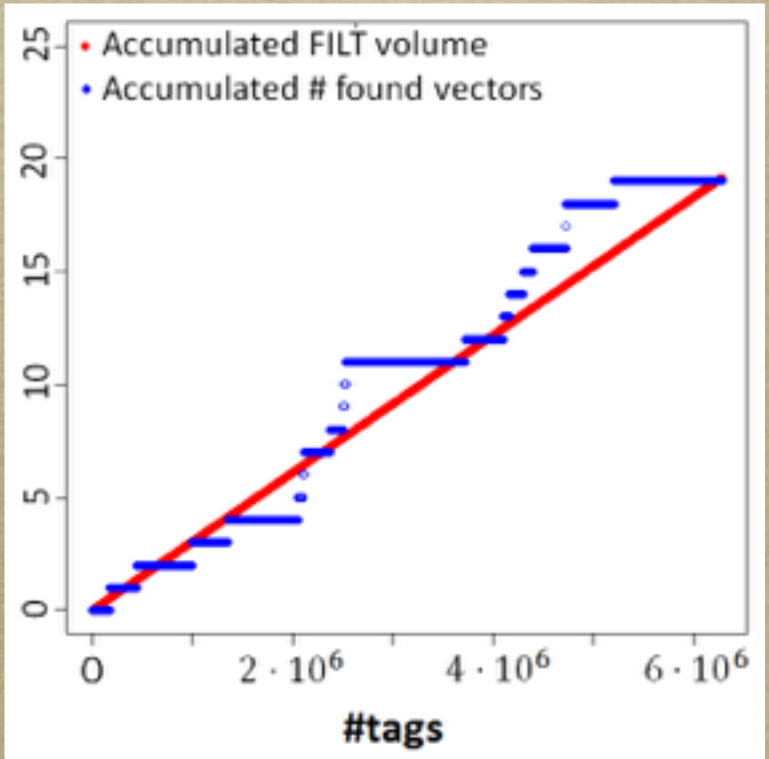
$E_{x \in C(t)}(||x||^2)$



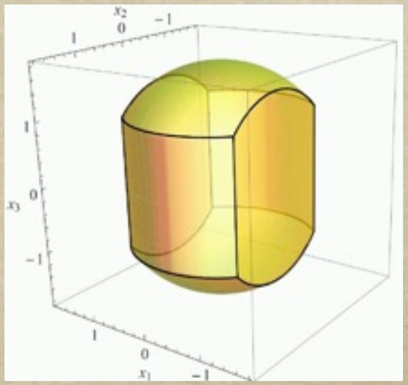
The largest-volume cells

Success probability by Statistical Inference

- The computation of vol(S∩C(t)) is too « slow » to approximate $\Sigma_{t\in U}$vol(S∩C(t)).
- So we ``select" a few thousands cells and… extrapolate!
  - Errors ≤ 1% in practice.
- Sound success probabilities for discrete pruning.



Discrete Pruning vs Cylinder Pruning



- Discrete pruning is faster when:
  - Small number of tags
  - High dimension
  - Weakly-reduced bases
- Benefits
  - Easy to parallelize
  - Easy generation of parameters

Optimizing the Basis

- The basis should try to maximize vol($S \cap C(t)$), which may be the same as minimizing $E_{x \in C(t)}(\|x\|^2)$. This suggests to minimize $\sum_j \|b_j^*\|^2$.
- The best bases for discrete pruning may not be the best bases for cylinder pruning.

Conclusion

Conclusion

- We unify Schnorr's algorithms [ScEu94] and [Sc03]: view random sampling as some pruned enumeration, and [GNR10]-analyze it under only the Gaussian heuristic.
  - Boxes instead of cylinder intersections.

Conclusion

- New tools
  - Computing volumes of ball/box intersections
  - Approximating a sum of many volumes
  - « Optimal » parameters for discrete pruning

Open Problems

- Asymptotically, what is the best form of pruning?
- Adapt blockwise reduction to discrete pruning
- What is the best reduction algorithm for discrete pruning?

Thank you for your attention...

Any question(s)?

http://eprint.iacr.org/2017/155