Enumeration: Finding Short Lattice Vectors by Exhaustive Search

Phong Nguyễn

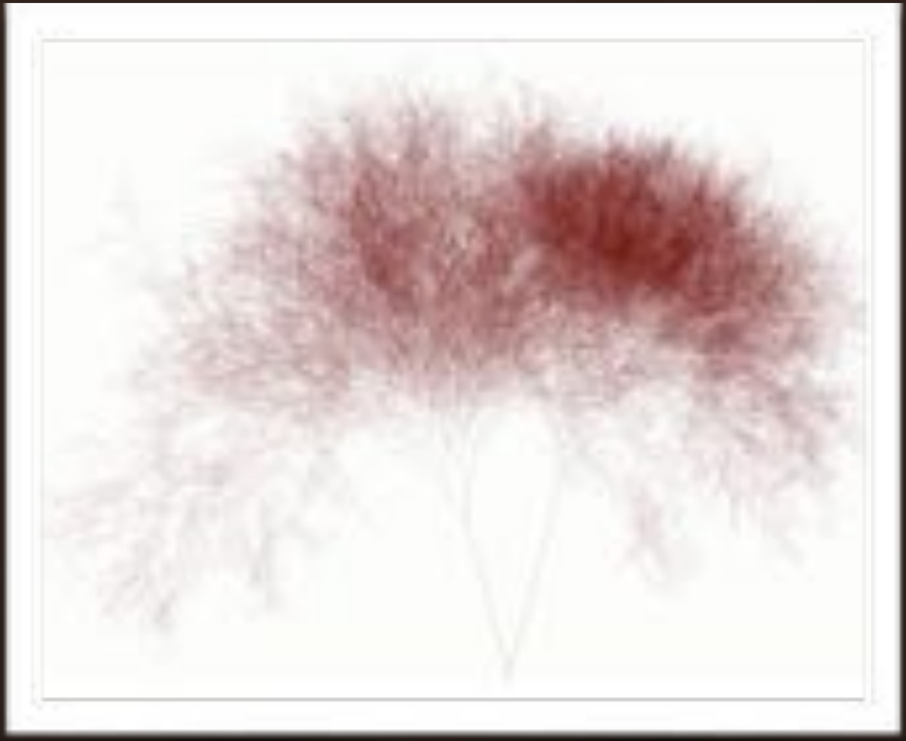
References

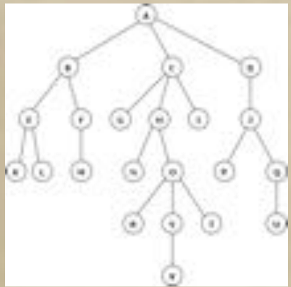
- Joint work with:
  - Nicolas Gama and Oded Regev, published at EUROCRYPT 2010: « Lattice Enumeration with Extreme Pruning ».
  - Yoshinori Aono, published at EUROCRYPT 2017: « Random Sampling Revisited: Lattice Enumeration with Discrete Pruning ».
  - Aono, Seito and Shikata, published at CRYPTO 2018: « Lower Bounds on Lattice Enumeration with Extreme Pruning ».
  - Aono and Shen, published at ASIACRYPT 2018: « Quantum Lattice Enumeration and Tweaking Discrete Pruning »

Summary

- Enumeration
- Enumeration with Pruning
  - Cylinder Pruning
  - Discrete Pruning

Solving SVP by Enumeration



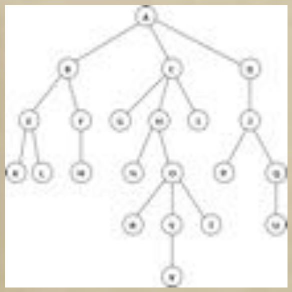


Enumeration

- 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 $\#(L \cap S)$.

Basis and Filtration

- If $(b_1,\ldots,b_d)$ is a basis of L:
  - $L_i := L(b_1,\ldots,b_i)$ is a sublattice of L for $1 \leq i \leq d$
  - $(L_1,\ldots,L_d)$ is a flag of L.
  - If $i \leq j$, the quotient L_j/L_i is a lattice of rank $j-i$ s.t. $\mathrm{vol}(L_j/L_i)=\mathrm{vol}(L_j)/\mathrm{vol}(L_i)$



Enumeration Insight

- Key ideas:
  - Projections never increase norms: if $\|v\| \le R$, then $\|v \bmod L_i\| \le R$.
  - L/L_j is a lower-rank lattice, whose short vectors can be lifted into short vectors of L/L_i if $i<j$.

Enumeration

- A) Reduce a basis.
- B) Exhaustive search all vectors ≤ R by enumerating all short vectors in L/L_{d-1}, then L/L_{d-2} ... until L
- Usually, B) is much more expensive than A).
- If the basis is LLL-reduced, B) costs $2^{O(d^2)}$.
- [Kannan1983] showed that A) and B) can be done in $2^{O(d \ln d)}$ poly-time operations.

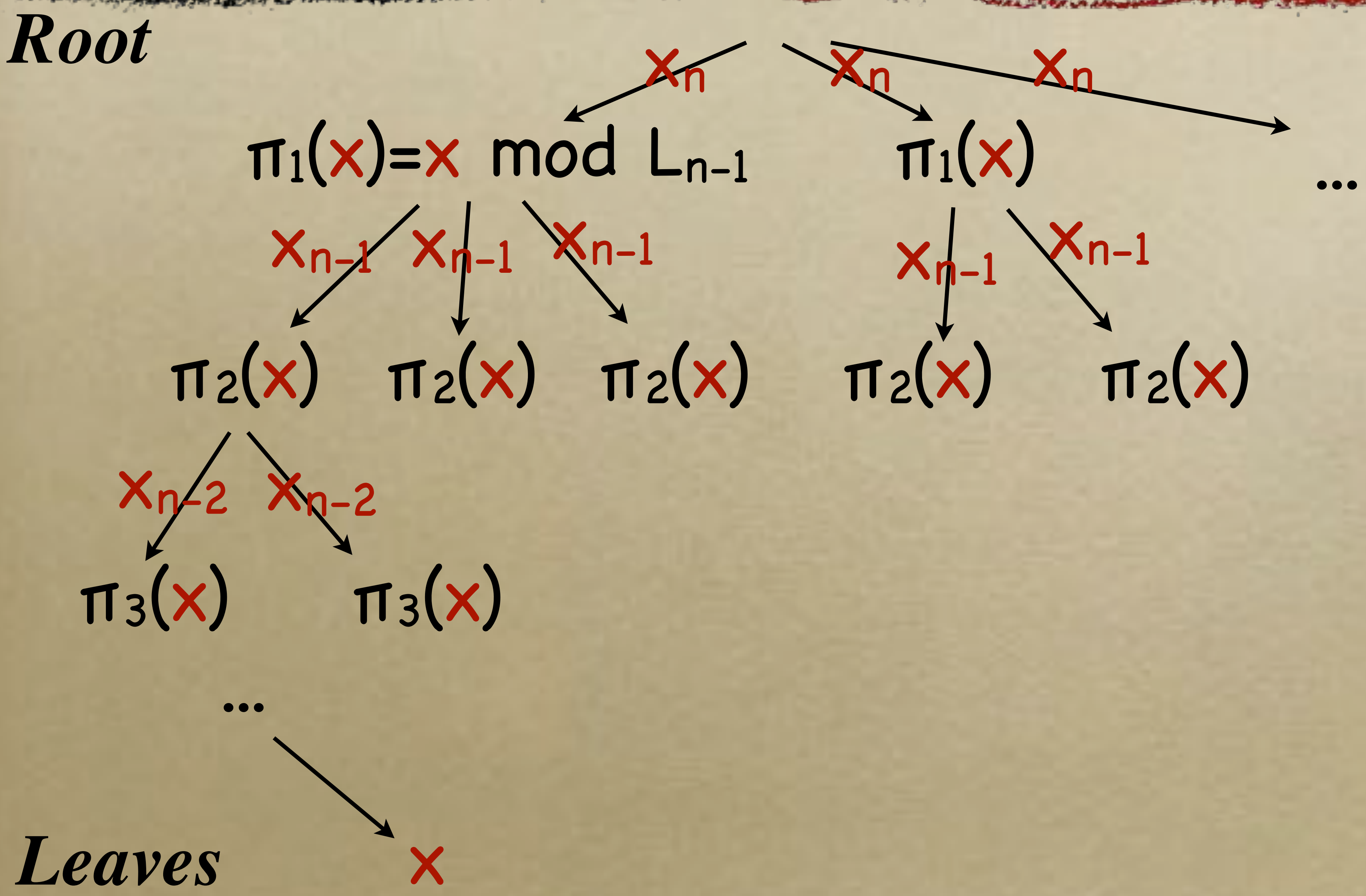
More precisely…

- Consider a lower-triangular matrix:

x_1	$b_{1,1}$				
x_2	$b_{2,1}$	$b_{2,2}$			
x_3	$b_{3,1}$	$b_{3,2}$	$b_{3,3}$		
x_4	$b_{4,1}$	$b_{4,2}$	$b_{4,3}$	$b_{4,4}$	
x_5	$b_{5,1}$	$b_{5,2}$	$b_{5,3}$	$b_{5,4}$	$b_{5,5}$

- If norm ≤ R, then
  - $(x_5 b_{5,5})^2 \leq R^2$
  - $(x_4 b_{4,4} + x_5 b_{5,4})^2 + (x_5 b_{5,5})^2 \leq R^2$
  - ...
  - So enumerate x_5, then x_4, etc.

Enumeration Tree



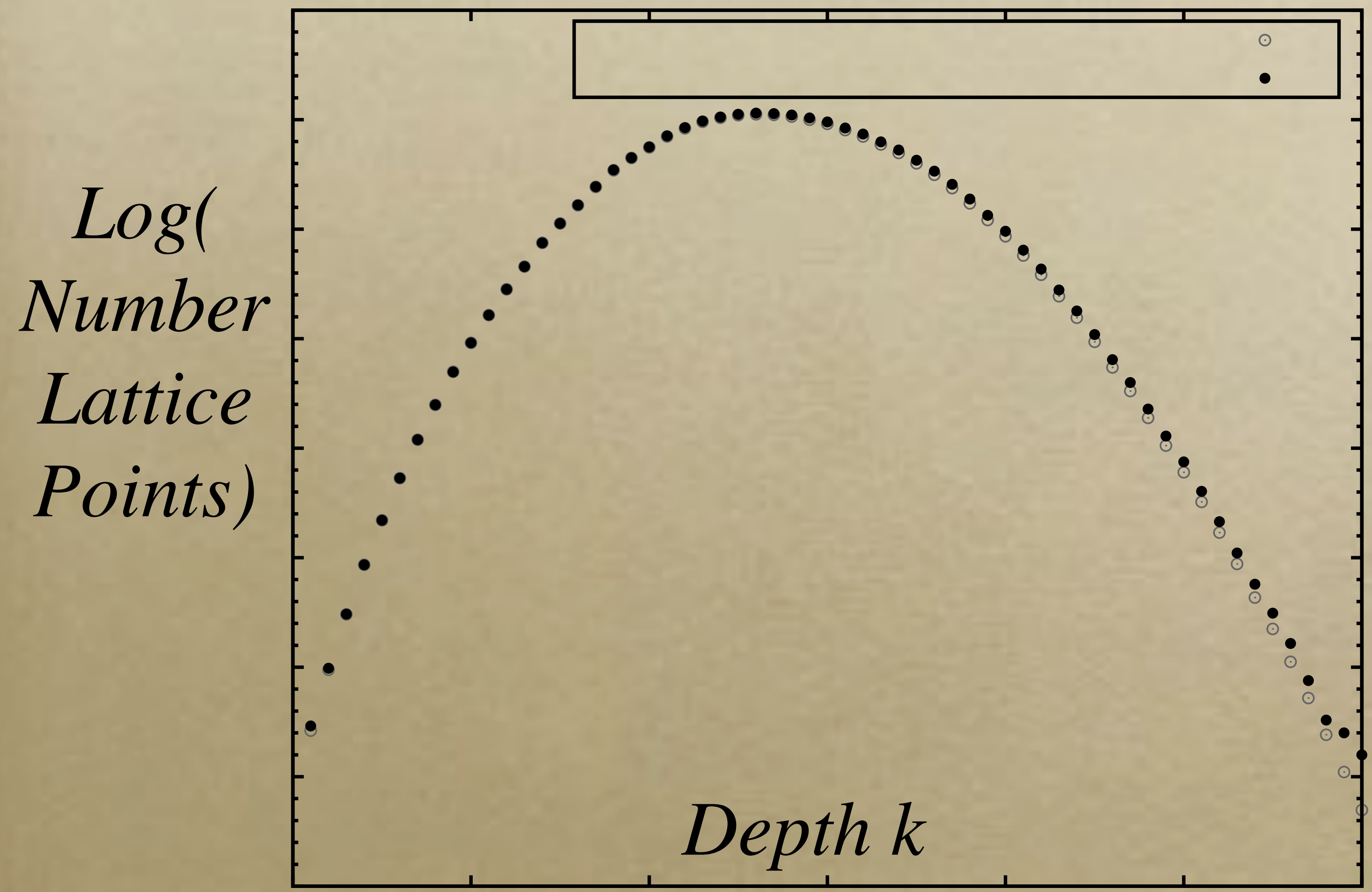
Enumeration tree

- Depth k contains all projected lattice points $\|\pi_k(y)\|$ ($y \in L$) of norm $\leq R$.
- The leaves are all $y \in L$ of norm $\leq R$.
- Enumeration searches the whole tree to compute all leaves, compare their norm to output a shortest vector $x \in L$.

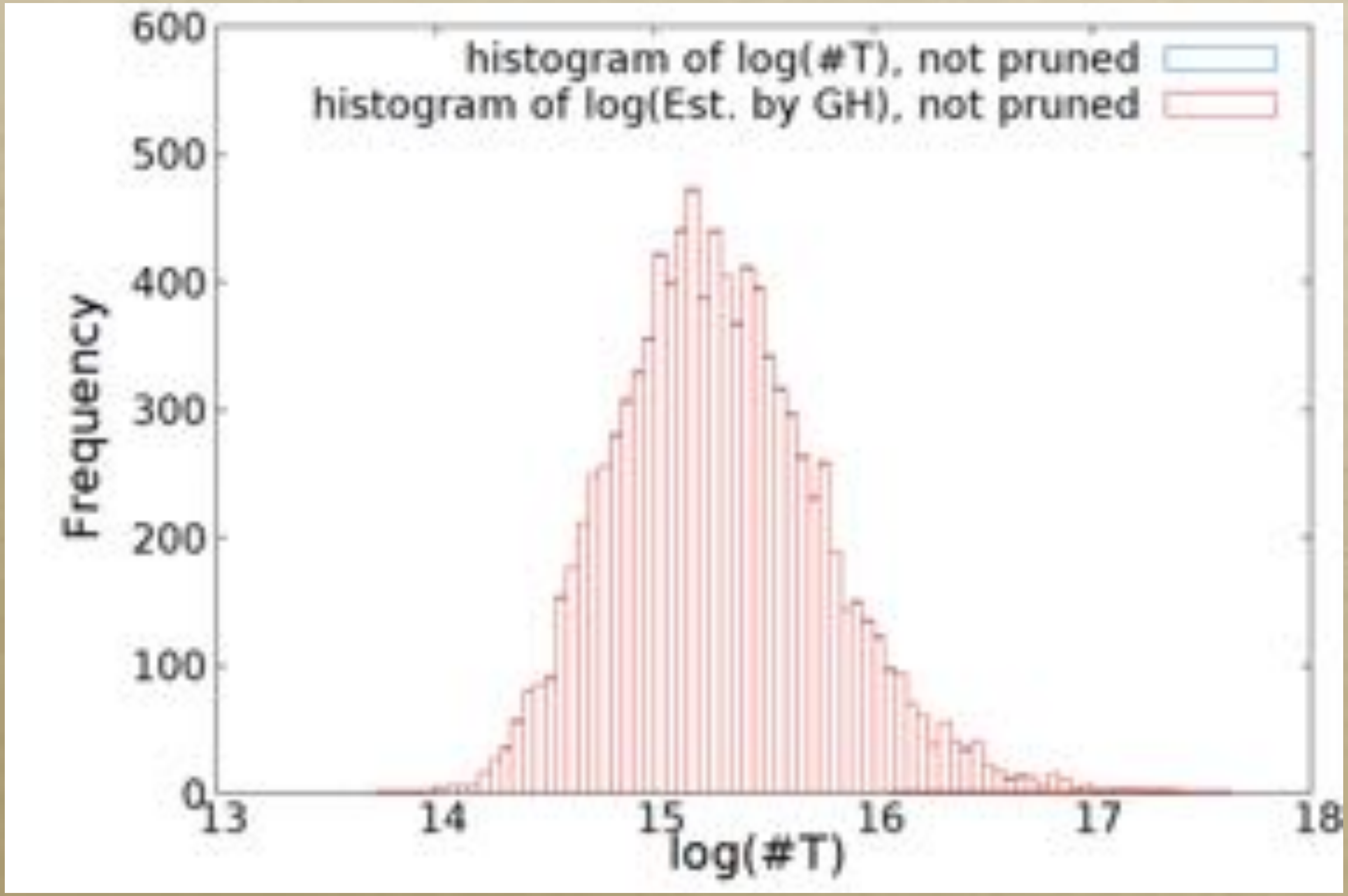
Complexity of Enumeration

- The complexity of enumeration is, up to a polynomial factor, the number of lattice points in all projected lattices inside the ball of radius R.
- This number can be upper bounded, but experimental numbers are close to the Gaussian heuristic $\sum_{1\leq k\leq d} v_k(R)/\mathrm{vol}(\pi_k(L))$, where $v_k(R)$ is the volume of the k-dim ball of radius R.

Accuracy of Gaussian Heuristic



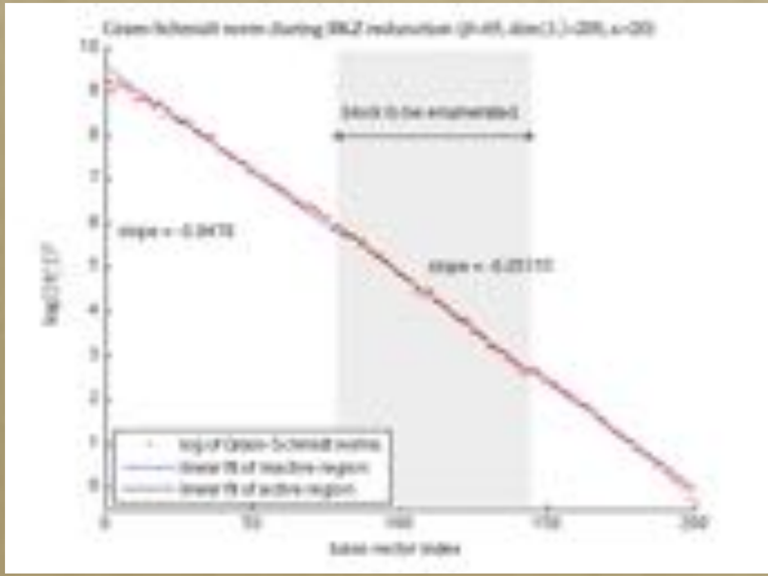
Accuracy of Gaussian Heuristic

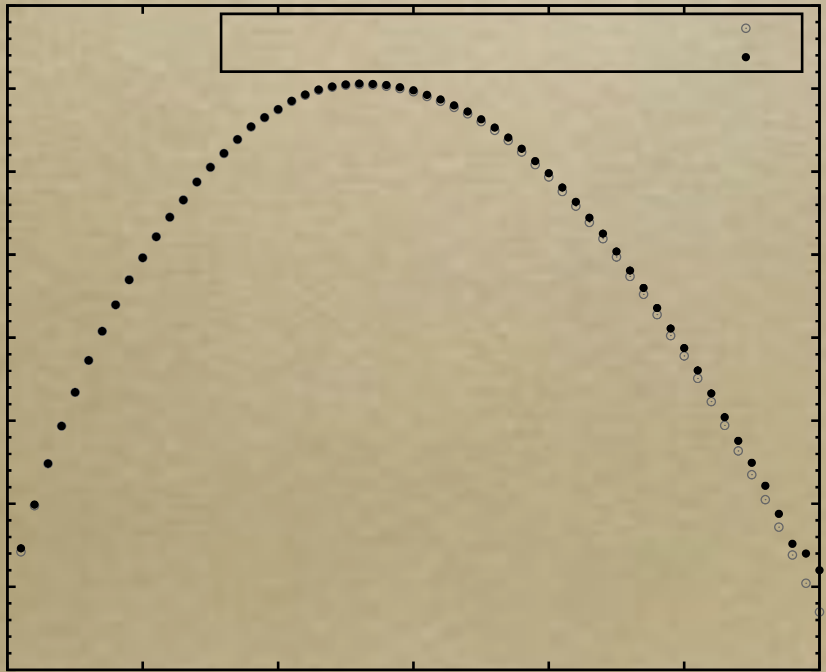


Distribution of Log(Number of nodes)

Shape

- For typical reduced bases, the Gram-Schmidt norms decrease geometrically in practice: most of the nodes are in middle depths k≈n/2. Their number is super-exponential.





Optimizing the Basis

- The basis should be chosen to minimize $\Sigma_{1 \le k \le d} v_k(R)/\mathrm{vol}(\pi_k(L))$ especially for $k \approx n/2$, i.e. to minimize $\mathrm{vol}(b_1,\ldots,b_{n/2})$.

Take Away

- Enumeration is based on one key idea
  - Filtration to decrease the lattice rank
- Once parameters are fixed, it is possible to reasonably estimate the running time
- Enumeration can be significantly sped up in practice using pruning, which slices a ball in a randomized manner.

Speeding Up Enumeration by Pruning

Speeding Up Enumeration

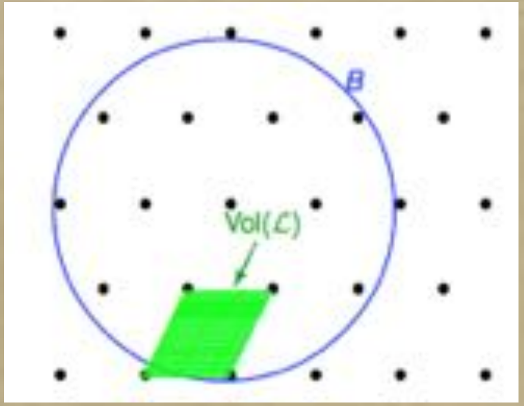
- Assume that we do not need all L∩S:
  - Can we make enumeration faster if we only need to find one vector?

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 $L \cap S \cap P$ is deterministic, but:
  - The set P is randomized: it depends on a (random) reduced basis.
  - The success probability is $\Pr(L \cap S \cap P \not\subseteq \{0\})$.
- $\#(L \cap S \cap P)$ « should be » $\approx \mathrm{vol}(S \cap P)/\mathrm{covol}(L)$ (Gaussian heuristic).

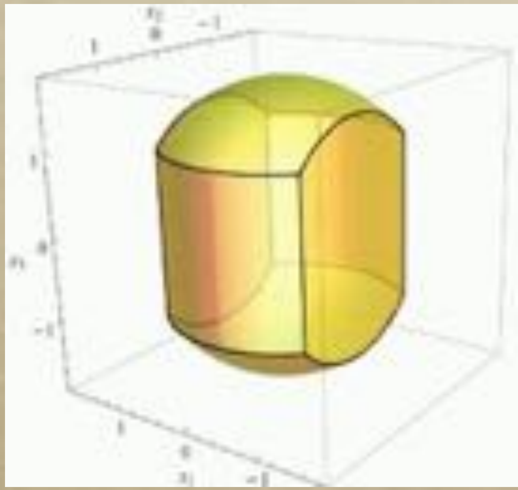


Extreme Pruning [GNR10]

- 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.

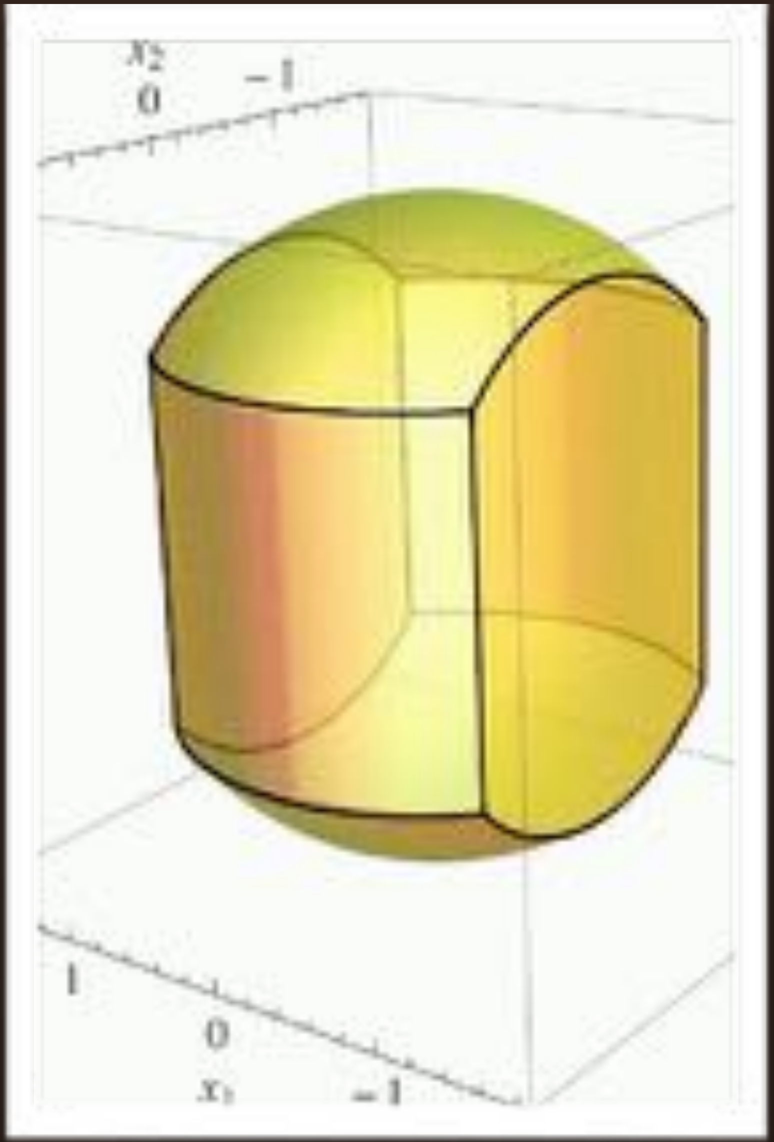


- Discrete Pruning ([AoN17] generalizing [Sc03,FuKa15]): P is a union of boxes.

Take Away

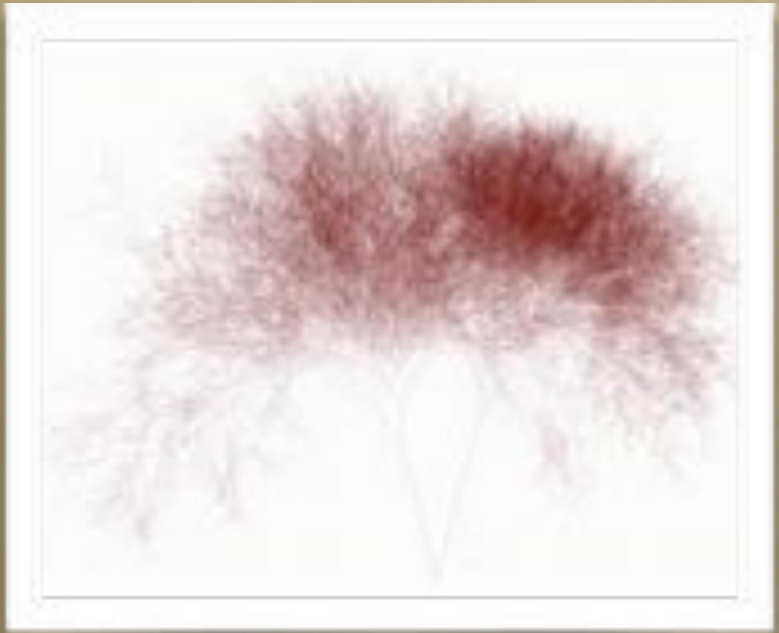
- Pruned enumeration is based on one more key idea
  - Slicing the ball in a randomized manner
- Once all parameters are fixed, it is possible to reasonably estimate the running time. But difficult to optimize.

Cylinder Pruning



Cylinder Pruning

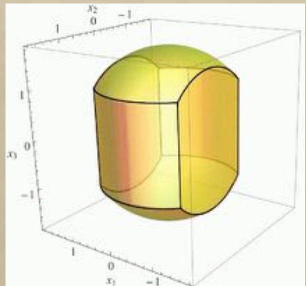
- [ScEu94,ScHo95], revisited in [GNR10].
- Idea: random projections are shorter.
- We can prune the gigantic tree.



Pruned enumeration cuts off many branches, by bounding projections.

Intuition

- Enumeration says: If $\|x\| \leq R$, then $\|\pi_k(x)\| \leq R$ for all $1 \leq k \leq n$
- But if x is random in the ball of radius R, its projection are shorter.
- For instance, we would expect $\|\pi_{n/2}(x)\| \approx R/\sqrt{2}$.



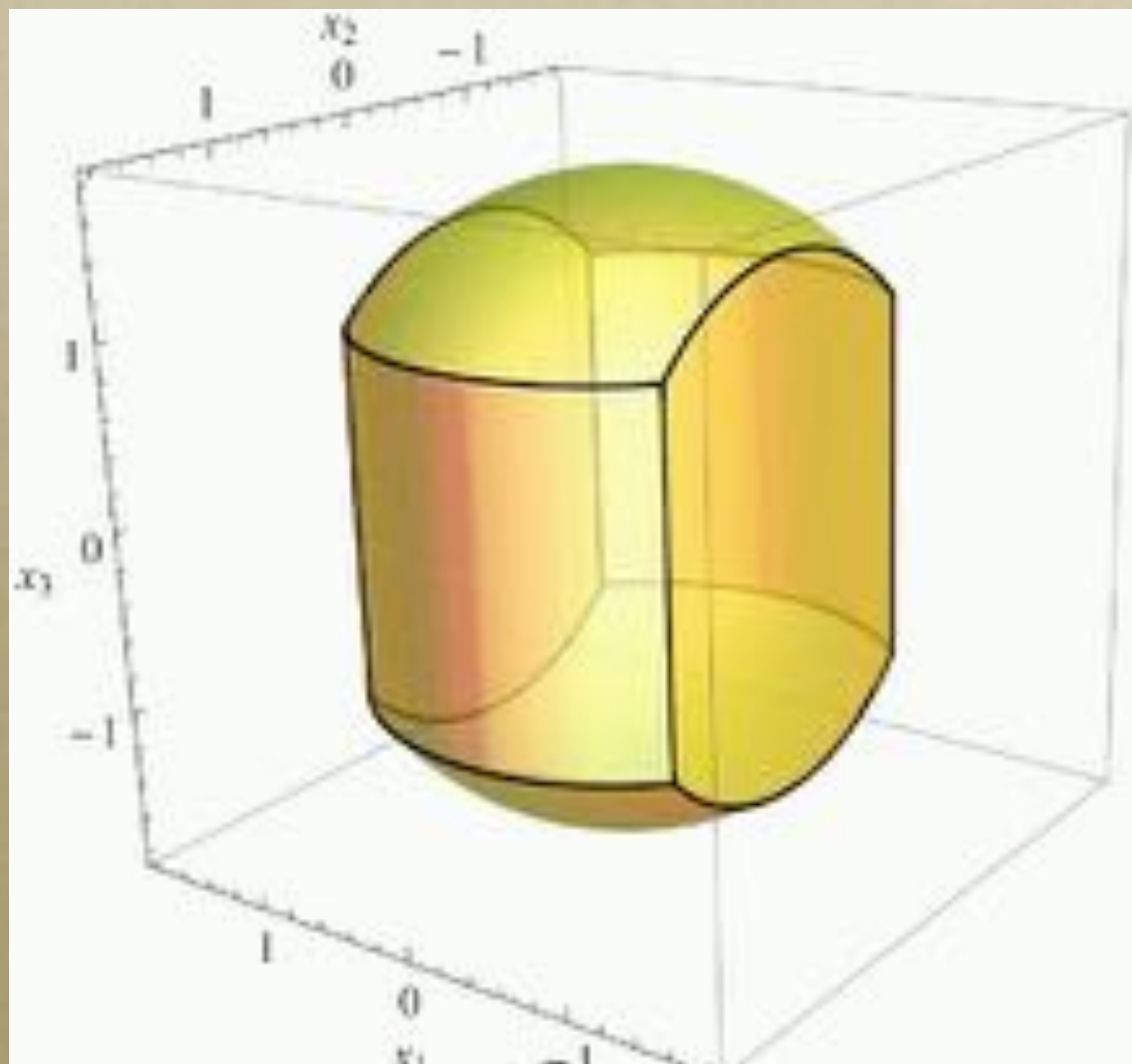
Cylinder Pruning

- Replace each inequality $||\pi_k(x)|| \leq R$ by $||\pi_k(x)|| \leq R_k\, R$ for each index k in $\{1,\ldots,n\}$, where $0 < R_k \leq 1$.
- The enumeration tree is pruned with $P = \{x \in \mathbf{R}^n \text{ s.t. } ||\pi_k(x)|| \leq R_k\, R \text{ for } 1 \leq k \leq n\}$. Again, one searches the tree to find all leaves.
- The algorithm is faster because there are less nodes.

Enumeration with cylinder pruning

- The complexity is, again up to a polynomial factor, a number of lattice points in projected lattices, but instead of balls, we have to consider new sets, whose volume might be harder to compute.

Balls Replaced by Cylinder Intersections



More Precisely

- The k-dim ball of radius R is replaced by: $\{(y_1,\ldots,y_k) \in \mathbf{R}^k$ s.t. for all $1 \leq i \leq k$, $y_1^2+\ldots+y_i^2 \leq R_i^2 \times R^2\}$.
- Its volume is $V_k(R)$ times the probability P_k that for $(y_1,\ldots,y_k)$ chosen uniformly at random from the unit ball, $y_1^2+\ldots+y_i^2 \leq R_i^2$ for all $1 \leq i \leq k$.

In other words

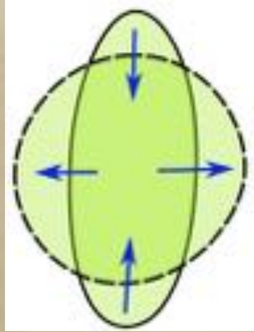
- The heuristic complexity of enumeration $\Sigma_{1\leq k\leq d}\, v_k(R)/\mathrm{vol}(\pi_{d-k+1}(L))$ is reduced to $\Sigma_{1\leq k\leq d}\, v_k(R)P_k/\mathrm{vol}(\pi_{d-k+1}(L))$.
- At depth k, the number of nodes is decreases by the multiplicative factor P_k.

Technical Problem [GNR10]

- To analyze and select good parameters for cylinder pruning, we need to estimate the volume of:
  - $C(R_1,\ldots,R_n)=\{(y_1,\ldots,y_n)\in\mathbf{R}^n$ s.t. for all $1\leq k\leq n$, $y_1^2+\ldots+y_k^2 \leq R_k^2\}$.
  - This can be done efficiently thanks to the Dirichlet distribution and well-chosen polytopes.

[ANSS18]
Limits to Cylinder Pruning

- Th: If $C(R_1,\ldots,R_n)$ achieves a success probability $\geq \alpha$, one can compute $\alpha_1,\ldots,\alpha_n > 0$ s.t. for all k, $R_k \geq \alpha_k$ and $\mathrm{vol}(C(R_1,\ldots,R_k)) \geq V_k(\alpha_k)$.
- This is based on isoperimetric inequalities.



Isoperimetric Inequalities

- Th: In $\mathbf{R}^n$, among all Borel sets of given measure, the n-dim ball has the least surface.
- Variant: Let A be a Borel set, and B the n-dim centered ball s.t. vol(A)=vol(B).

  Let $X \in \mathbf{R}^n$ with Gaussian distribution (or any radial pdf which decays monotonically). Then $\Pr(X \in A) \leq \Pr(X \in B)$.

Discrete Pruning

Lattice Partitions

- Any partition of $\mathbf{R}^n = \cup_{t \in T} C(t)$ into countably many cells s.t.:
  - cells are disjoint: $C(i) \cap C(j) = \emptyset$
  - 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

- Enum(L∩C(t)) ≃ Egg opening

Lattice Enumeration with Discrete Pruning [AoN17]

- 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 $vol(S \cap P) = \Sigma_{t \in U}\, vol(S \cap C(t))$
- How to select the set U of tags?
  We'd like the ones maximizing $vol(S \cap C(t))$.

Trivial Lattice Partitions

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

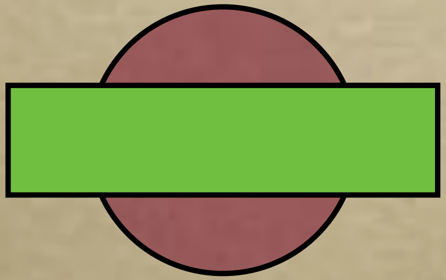
The « Natural » Partition [FuKa15]

- $T=\mathbf{N}^n$ 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

- This discrete pruning is very easy to implement.
- But there is one technical issue: to estimate the success probability, we need to approximate vol(S∩C(t)) for many t's where:



  - S is a ball
  - C(t) is a box, or a union of symmetric boxes.

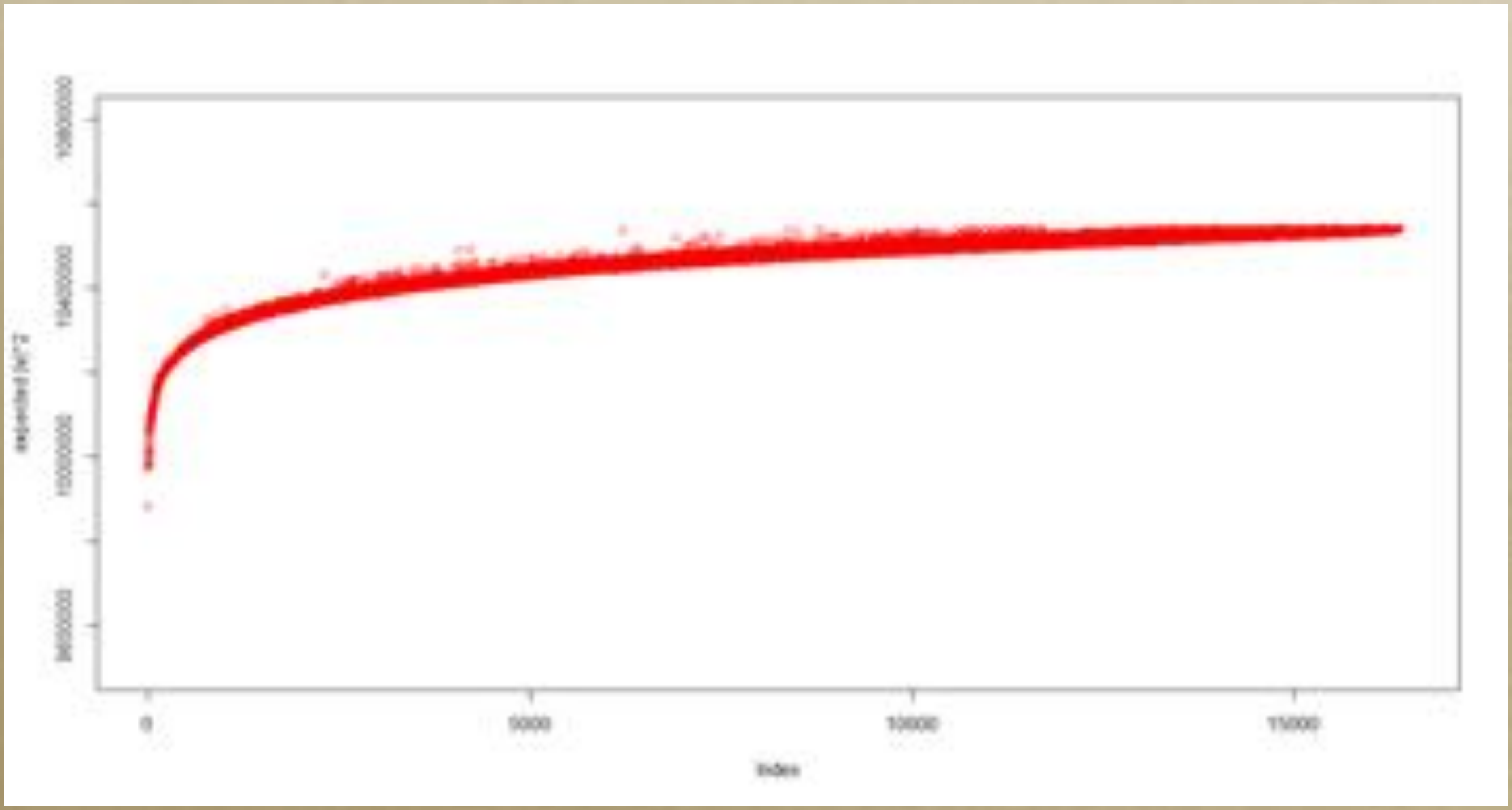
Intersection of a Ball with a Box

- Let B=unit-ball and $H=\Pi_i [a_i,b_i]$ be a box. Compute $vol(B \cap H)$.
- There are exact formulas as infinite series, based on Fourier transforms and Fourier series.
- In practice, the Fast Inverse Laplace Transform takes less than 1s in dim 100.

Heuristics For Selecting Cells

- The exact computation of $vol(S \cap H)$ is « slow ».
- Heuristic: the M cells maximizing $vol(S \cap C((t_1, \ldots, t_n)))$ are the M cells minimizing $E_{x \in H}(\|x\|^2)$.
- It suffices to find the M minimal values of $f(t_1,\ldots,t_n)=\Sigma_j(3t_j^2+3t_j+1)\|b_j^*\|^2/12$ over $\mathbf{N}^n$. This can be done in time essentially M poly-time operations [ANS18].

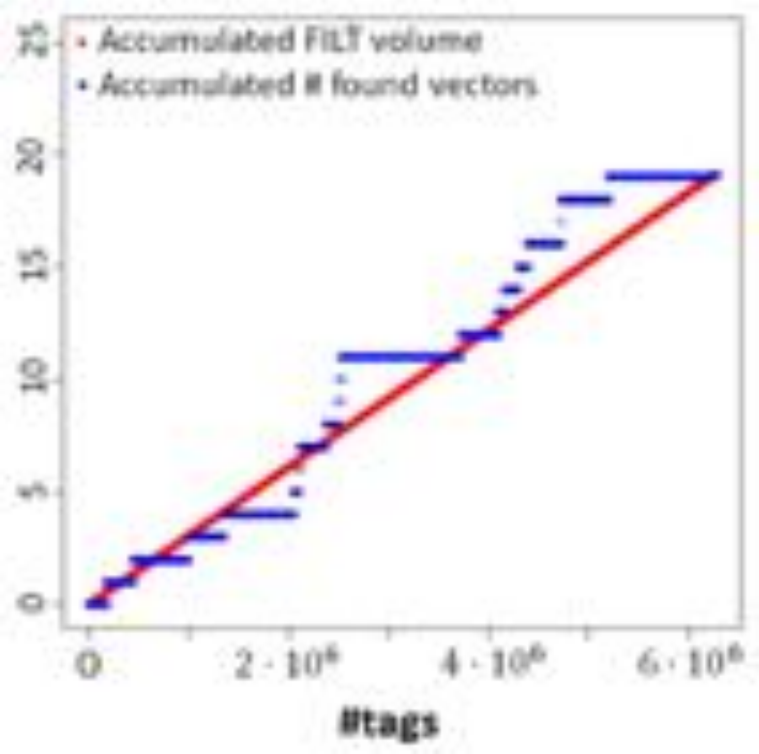
Correlation Between Expectation and Volume



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.



Conclusion on Enumeration

- Enumeration is very useful in practice to find extremely short vectors. It can also be used to approximate with small factors.
- But it requires pruning, whose main technical issue is approximating volumes of certain bodies: cylinder intersections or box-ball intersections.