Discrete Ziggurat: A Time-Memory Trade-off for Sampling from a Gaussian Distribution over the Integers

Johannes Buchmann, Daniel Cabarcas, Florian Göpfert,
Andreas Hülsing, Patrick Weiden

Technische Universität Darmstadt
Darmstadt, Germany

Selected Areas in Cryptography
Aug 16, 2013

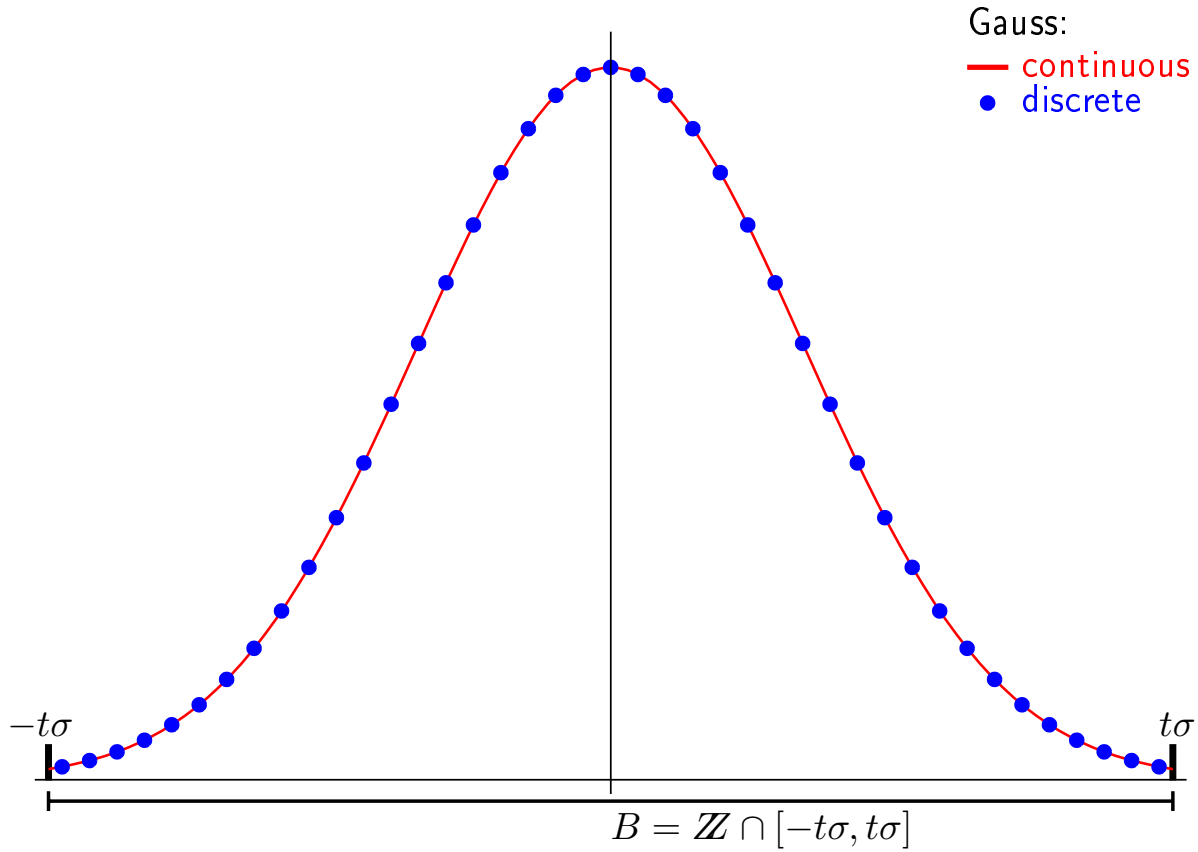
Outline

Motivation and Contribution

- Discrete Gaussians widely used in lattice-based crypto
  - E.g. signatures, encryption, (F)HE, multilinear maps
- Critical technical challenge: accurate and efficient sampling of discrete Gaussians
  - E.g. sampling $\approx 50\%$ of signing time [WHCB13]
- Existing methods: either large memory or very slow
  - E.g. Peikert's sampler about 12MB of storage [GD12]
  - No flexibility in choice of memory and speed
  - Memory requirement acceptable on PC, but not on smaller devices
- Our contribution: alternative sampler for discrete Gaussians offering a flexible trade-off between speed and memory

Discrete Gaussians and Samplers

- Discrete Gaussian distribution D_σ for parameter σ assigns $x \in \mathbb{Z}$ probability proportional to $\rho_\sigma(x) = \exp(-\frac{1}{2}x^2/\sigma^2)$
- Sufficient for cryptographic applications: bounded support $B := \mathbb{Z} \cap [-t\sigma, t\sigma]$ with tailcut $t > 0$ large enough [GPV08]

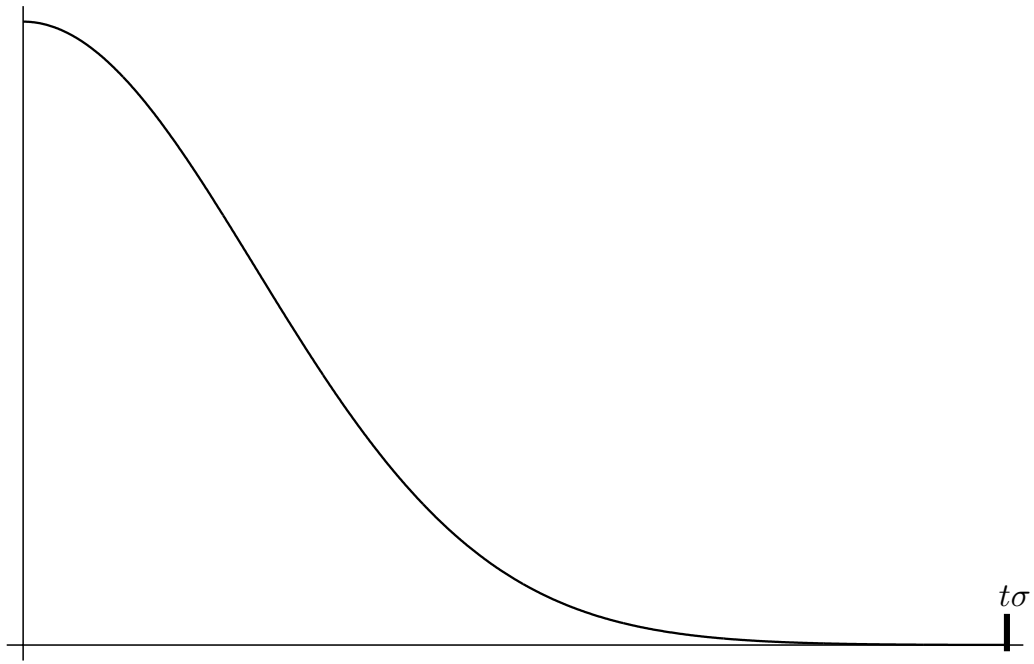


Discrete Gaussians and Samplers

- Rejection sampling (rejSam)
- Inverse cumulative distribution function (invCDF)
- Knuth-Yao (KY)
- Hybrid variants: rejection sampling with lookup-table, ...

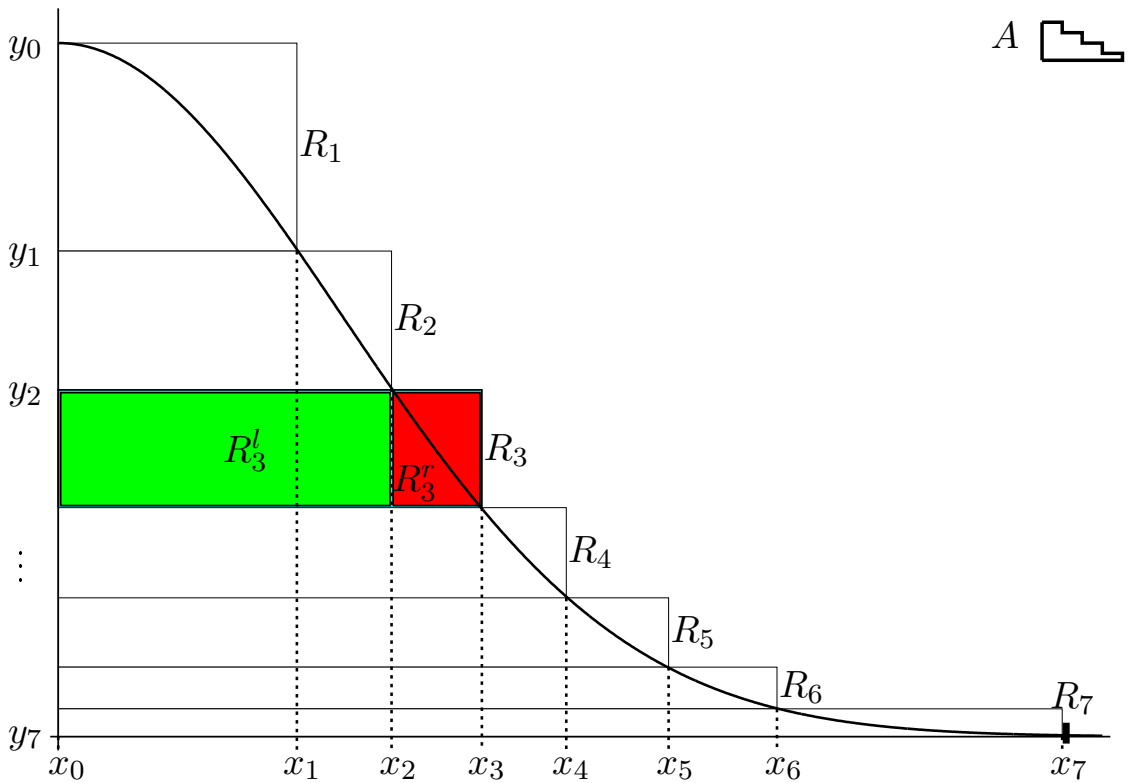
The Ziggurat Algorithm

- Belongs to class of rejection sampling algorithms
- Introduced by Marsaglia and Tsang for sampling from a continuous Gaussian distribution [MT00]
- Observation:
  - Symmetry: sample $x \in [0, t\sigma]$ acc. to PDF
  - Sample sign $s \in \{-1, 1\}$ and return sx
  - Attention: case $x = 0$



The Ziggurat Algorithm

- Sampling $x \in [0, t\sigma]$: Intuition
  - Given: partition of area into rectangles of equal size
  - Choose rectangle $R_i = R_i^l \cup R_i^r$ randomly
  - Sampling in rectangle R_i:
    - Sample $x \in [0, x_i]$ randomly
    - If $x \in R_i^l$: accept x
    - Else sample in R_i^r using rejection sampling (restart)



The Ziggurat Algorithm

- Ziggurat = efficient "instantiation" of rejection sampling in enclosing area A (instead of in $[0, t\sigma] \times [0, 1]$)
- Rectangles of equal size: ensures equality of probabilities
- Storage: (x_i, y_i) for R_i where $i = 1, \ldots, \#\text{rectangles}$
- Expensive part: sampling in R_i^r
- Trade-off:
  - Controlled by #rectangles
  - More rectangles: R_i^l comparatively bigger than R_i^r

  $\rightarrow$ acceptance of x without computing $\rho_\sigma(x)$ with higher probability

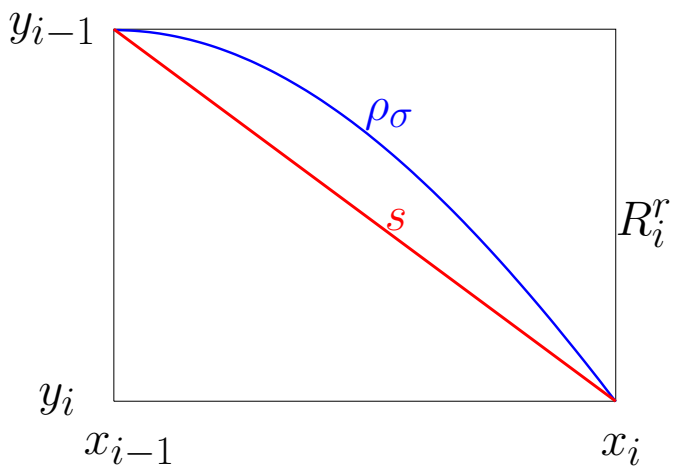
  $\rightarrow$ less rejections of $x \rightarrow$ less 'restarts'
  - But: more memory needed

The Ziggurat Algorithm: Discretization

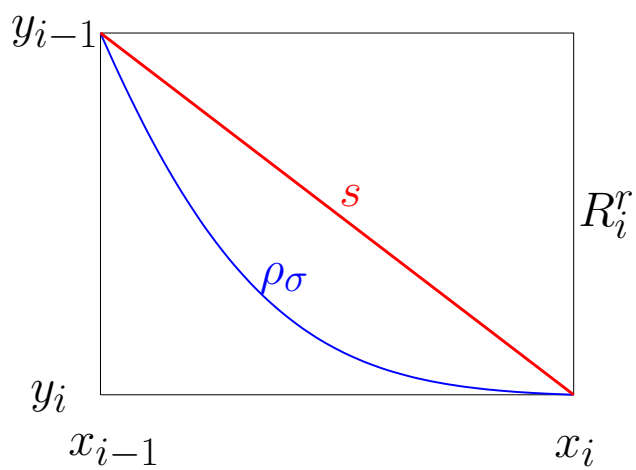
Procedure: same as continuous

Adaptation to discrete case:

- Notion of 'size'
- Pre-computation of rectangles
- Implementation issues:
  - Fix point precision
  - Discretizing the height
- Improvement of sampling in R_i^r: straight line approach



The concave-down case



The concave-up case

Quality of our Sampler and Parameter Choice

Theorem

The statistical distance between the discrete Gaussian distribution D_σ and the distribution $\overline{D}_\sigma$ output by our algorithm is bounded by

$$\Delta(D_\sigma, \overline{D}_\sigma) < te^{(1-t^2)/2} + \frac{|B_0^+|}{\overline{\rho}_\sigma(B^+) + \frac{1}{2}}(2^{-\omega+1} + 2^{-n}).$$

Proof idea: Hybrid argument using intermediary distributions

Quality of our Sampler and Parameter Choice

- Parameters: Gaussian parameter σ, tailcut t, fix point precision n, height precision ω
- Goal: negligible statistical distance, e.g.

$$\underbrace{te^{(1-t^2)/2}}_{l} + \underbrace{\frac{|B_0^+|}{\overline{\rho}_\sigma(B^+) + \frac{1}{2}}(2^{-\omega+1} + 2^{-n})}_{r} < 2^{-100}$$

$\rightarrow$ Find smallest integer t s.t. $l < 2^{-101}$: $t = 13$

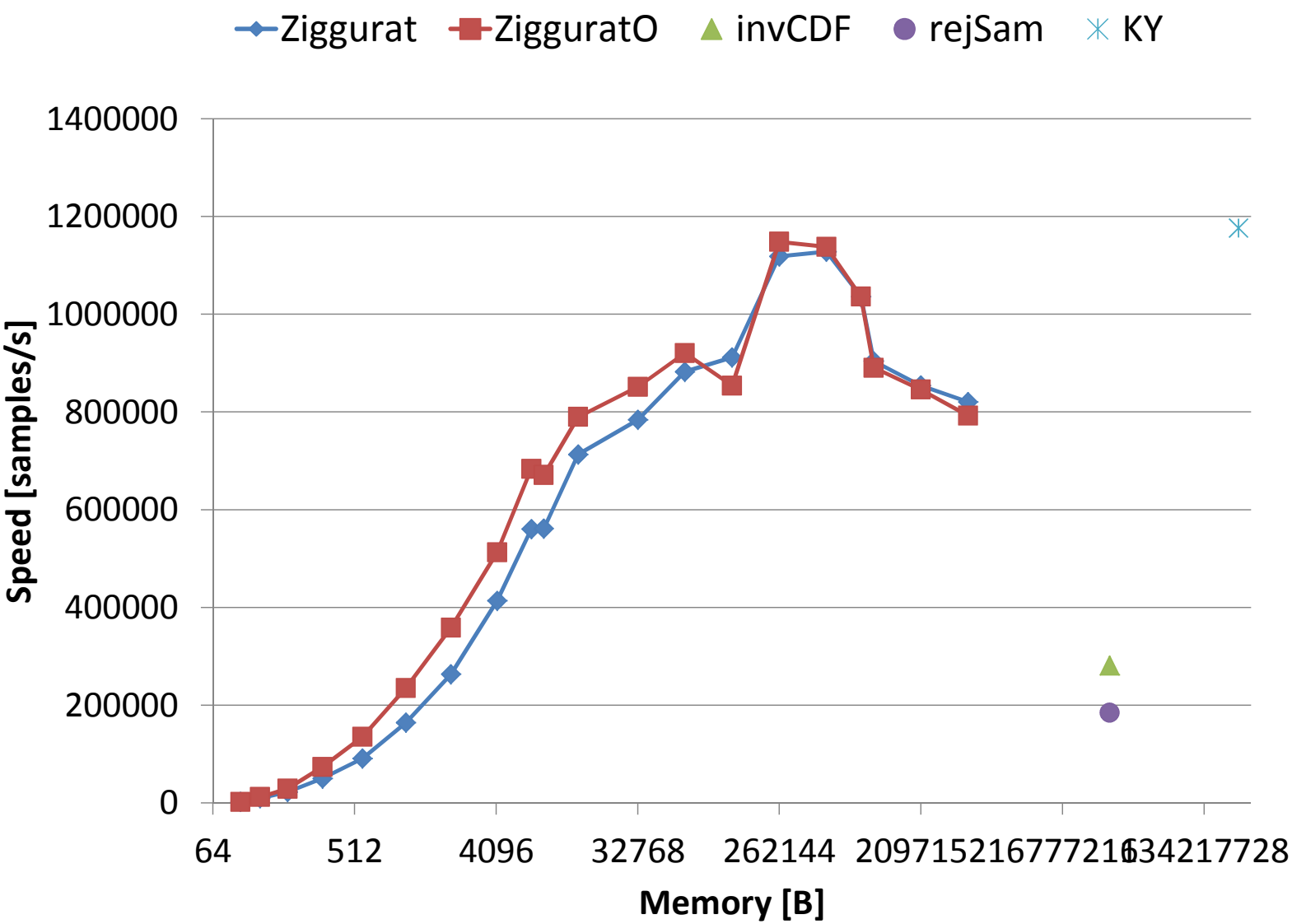
$\rightarrow$ Choose $\omega = n + 1$ reduces complexity of r

$\rightarrow$ Find n such that $r < 2^{-101}$: $n = 106$

Experiments and Results

- C++ implementation using Number Theory Library (NTL, [Sho])
- Parameters: $n = 106$ ($\omega = 107$), $t = 13$, different σ's
- $\sigma = 32$ maintains worst-to-average-case reduction [Reg05], $\sigma = 1.6 \cdot 10^5$ according to [GD12]
- Algorithms: Ziggurat, ZigguratO, invCDF*, rejSam*, KY (* = lookup-table)
- Each algorithm queried to output 10^6 samples
- Measured running time using `clock_gettime` with clock `CLOCK_PROCESS_CPUTIME_ID` (excluded pre-/post-comps.)
- Computed memory consumption using #fixed variables in regard to their type

Experiments and Results



Different samplers for $\sigma = 1.6 \cdot 10^5$

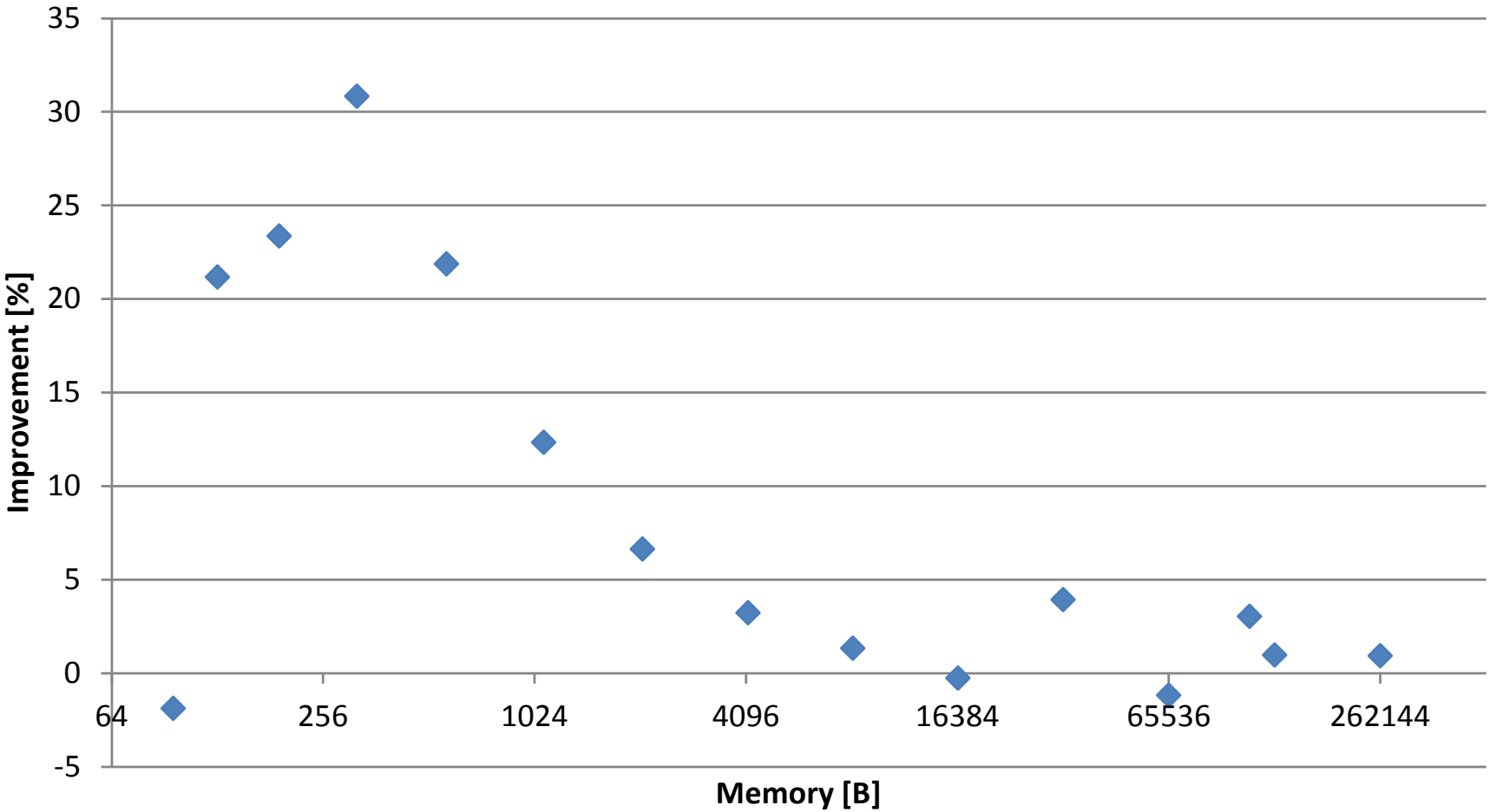
Experiments and Results

Some numbers...

- $\sigma = 32$:
  - rejSam factor 4.2 slower than invCDF, without lookup-table factor 558 slower
  - Ziggurat factor 1.91 slower than invCDF, 2.19 faster than rejSam
  - KY factor 3.53 faster than invCDF, but doubled memory
- $\sigma = 1.6 \cdot 10^5$:
  - invCDF factor 4 slower than Ziggurat, factor 64 more memory
  - rejSam about factor 6 slower than Ziggurat
  - KY only better than Ziggurat by 4%, but 424 times more memory

Experiments and Results

Improvement rate of ZigguratO to Ziggurat



Conclusion: Take-Home-Message

Discrete Ziggurat

=

Alternative **sampler** for
discrete Gaussians offering a
flexible **trade-off** between
speed and memory

Further details...

Source code on homepage:
`https://www.cdc.informatik.tu-darmstadt.de/~pschmidt/implementations/ziggurat/ziggurat-src.zip`

Version of paper with proofs on eprint:
`https://eprint.iacr.org/2013/510.pdf`

Thanks!