Simple, Fast and Constant-Time Gaussian Sampling over the Integers for Falcon

Thomas Prest - Thomas Ricosset - Mélissa Rossi

NIST PQC Workshop

2019/08/23

Falcon

(P-A Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Prest, T. Ricosset, G. Seiler, W. Whyte, Z. Zhang)

Falcon

(P-A Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Prest, T. Ricosset, G. Seiler, W. Whyte, Z. Zhang)

Based on the GPV framework

Gentry, Peikert and Vaikuntanathan STOC 2008

Falcon

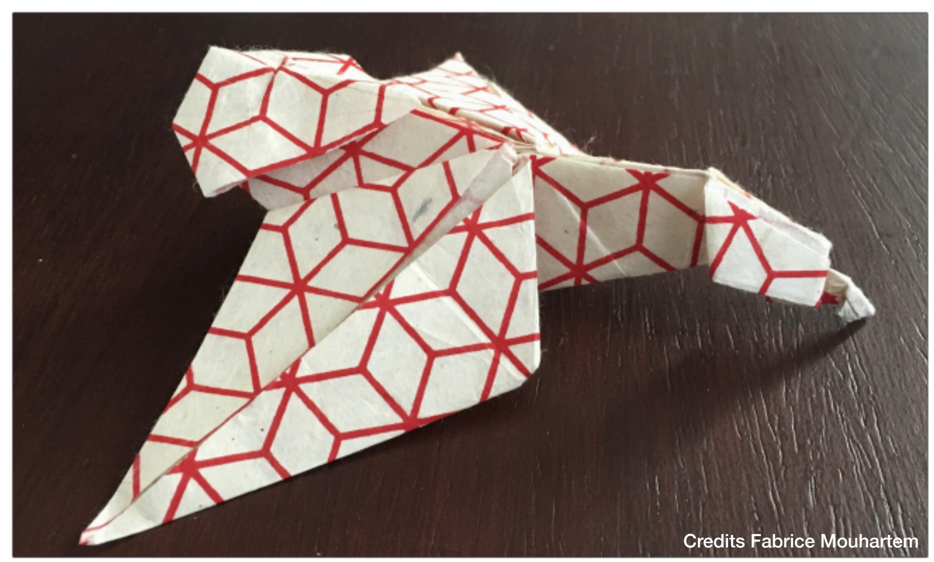
(P-A Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Prest, T. Ricosset, G. Seiler, W. Whyte, Z. Zhang)

Based on the GPV framework

Gentry, Peikert and Vaikuntanathan STOC 2008

Relying on NTRU lattices

Hoffstein et al. ANTS 1998, CT-RSA 2003



Falcon

(P-A Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Prest, T. Ricosset, G. Seiler, W. Whyte, Z. Zhang)

Based on the GPV framework

Gentry, Peikert and Vaikuntanathan STOC 2008

Relying on NTRU lattices

Hoffstein et al. ANTS 1998, CT-RSA 2003

Using Fast Fourier Orthogonalization

Ducas-Prest, ISSAC 2016

Falcon

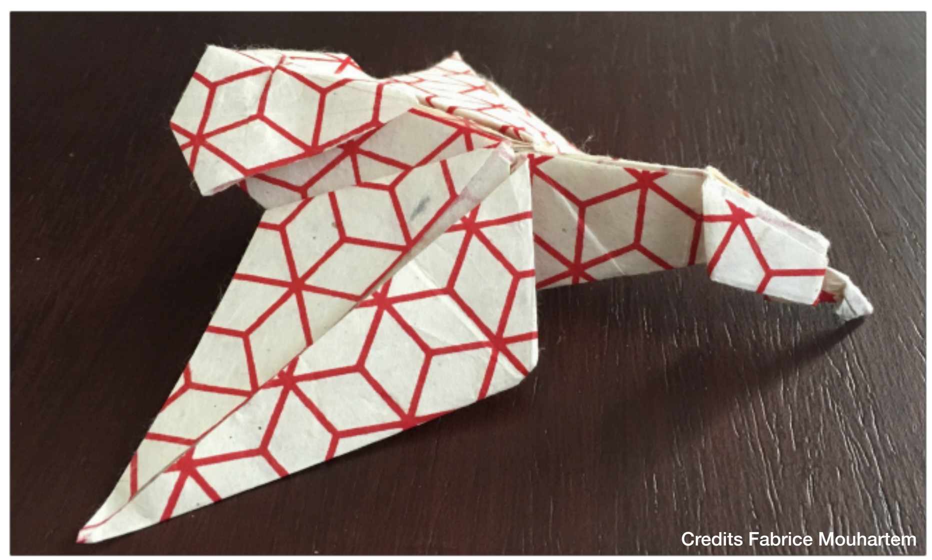
(P-A Fouque, J. Hoffstein, P. Kirchner, V. Lyubashevsky, T. Pornin, T. Prest, T. Ricosset, G. Seiler, W. Whyte, Z. Zhang)

Based on the GPV framework

Gentry, Peikert and Vaikuntanathan STOC 2008

Relying on NTRU lattices

Hoffstein et al. ANTS 1998, CT-RSA 2003



Using Fast Fourier Orthogonalization

Ducas-Prest, ISSAC 2016

Compact signatures

|s| + |pk| minimized

Falcon in a nutshell

$$\mathscr{R} = \frac{\mathbb{Z}_q[x]}{x^n + 1}$$

KeyGen()

- Generate matrices $\mathbf{A}, \mathbf{B}$ with coefficients in $\mathscr{R}$

  such that $\begin{cases} \mathbf{BA} = \mathbf{0} \\ \mathbf{B} \text{ has small coefficients} \end{cases}$
- $pk \leftarrow \mathbf{A}$
- $sk \leftarrow \mathbf{B}$

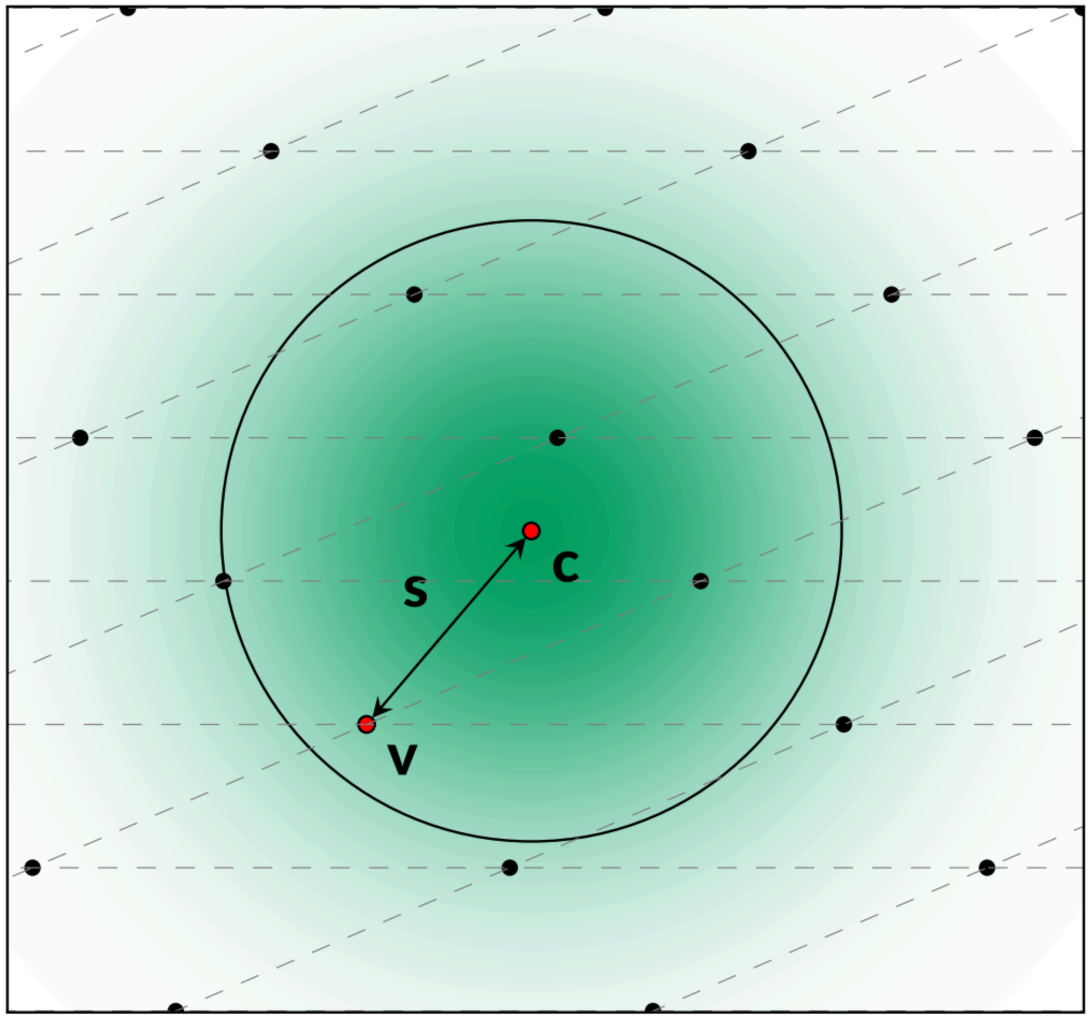
Sign(m,sk)

- Compute $\mathbf{c}$ such that $\mathbf{cA} = H(m)$
- $\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$
- $\mathbf{s} \leftarrow \mathbf{c} - \mathbf{v}$

Verify(m,pk,s)

Accept iff:

$$\begin{cases} \mathbf{s} \text{ is short} \\ \mathbf{sA} = H(m) \end{cases}$$



Round 1 Falcon

Advantages

- ☑ Compact
- ☑ Fast
- ☑ GPV framework proved secure in the ROM and QROM (Boneh et al. ASIACRYPT 2011)

Round 1 Falcon

Limitations

- ☐ Non Trivial to understand and implement
- ☐ Floating point arithmetic
- ☐ Side channel resistance not very studied

Round 1 Falcon

Limitations

?

- [ ] Non Trivial to understand and implement
- [ ] Floating point arithmetic
- [ ] Side channel resistance not very studied

This work

- [x] Integer arithmetic
- [x] Theoretically studied constant time
- [x] Implementations

What is not constant time and not portable in Falcon?

« Constant time »

The execution time does not depend on the **private key** $\mathbf{B}$

➡ Not necessarily constant

Sign(m,sk)

- Compute $\mathbf{c}$ such that $\mathbf{cA} = H(m)$
- $\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$
- $\mathbf{s} \leftarrow \mathbf{c} - \mathbf{v}$

What is not constant time and not portable in Falcon?

« Constant time »

The execution time does not depend on the **private key** $\mathbf{B}$

➡ Not necessarily constant

Sign(m,sk)

- Compute $\mathbf{c}$ such that $\mathbf{cA} = H(m)$
- $\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$
- $\mathbf{s} \leftarrow \mathbf{c} - \mathbf{v}$

What is not constant time and not portable in Falcon?

« Constant time »

The execution time does not depend on the **private key** $\mathbf{B}$

➡ Not necessarily constant

Sign(m,sk)

- Compute $\mathbf{c}$ such that $\mathbf{cA} = H(m)$
- $\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$
- $\mathbf{s} \leftarrow \mathbf{c} - \mathbf{v}$

ffsampling

What is not constant time and not portable in Falcon?

« Constant time »

The execution time does not depend on the **private key** $\mathbf{B}$

➡ Not necessarily constant

Sign(m,sk)

- Compute $\mathbf{c}$ such that $\mathbf{cA} = H(m)$
- $\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$
- $\mathbf{s} \leftarrow \mathbf{c} - \mathbf{v}$

ffsampling

Gaussian Sampling over $\mathbb{Z}$

What is not constant time and not portable in Falcon?

« Constant time »

The execution time does not depend on the **private key** $\mathbf{B}$

➡ Not necessarily constant

Sign(m,sk)

- Compute $\mathbf{c}$ such that $\mathbf{cA} = H(m)$
- $\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$
- $\mathbf{s} \leftarrow \mathbf{c} - \mathbf{v}$

→ ffsampling → Gaussian Sampling over $\mathbb{Z}$

Except Gaussian sampling, other operations do not use conditional branching

What is not constant time and not portable in Falcon?

« Constant time »

The execution time does not depend on the **private key** $\mathbf{B}$

➡ Not necessarily constant

Sign(m,sk)

- Compute $\mathbf{c}$ such that $\mathbf{cA} = H(m)$
- $\mathbf{v} \leftarrow$ a vector in $\Lambda(\mathbf{B})$ close to $\mathbf{c}$
- $\mathbf{s} \leftarrow \mathbf{c} - \mathbf{v}$

→ ffsampling → Gaussian Sampling over $\mathbb{Z}$

Except Gaussian sampling, other operations do not use conditional branching

Assumption

$+, -, \times, /$ Constant time on integers

Constant time Gaussian sampling

Some literature on Gaussian Samplers:

Sinha Roy, Vercauteren and
Verbauwhede SAC 2013
Hulsing, Lange and Smeets PKC 2018
Micciancio and Walter CRYPTO 2017
Karmakar et al. DAC IEEE 2019

Constant time Gaussian sampling

Some literature on Gaussian Samplers:

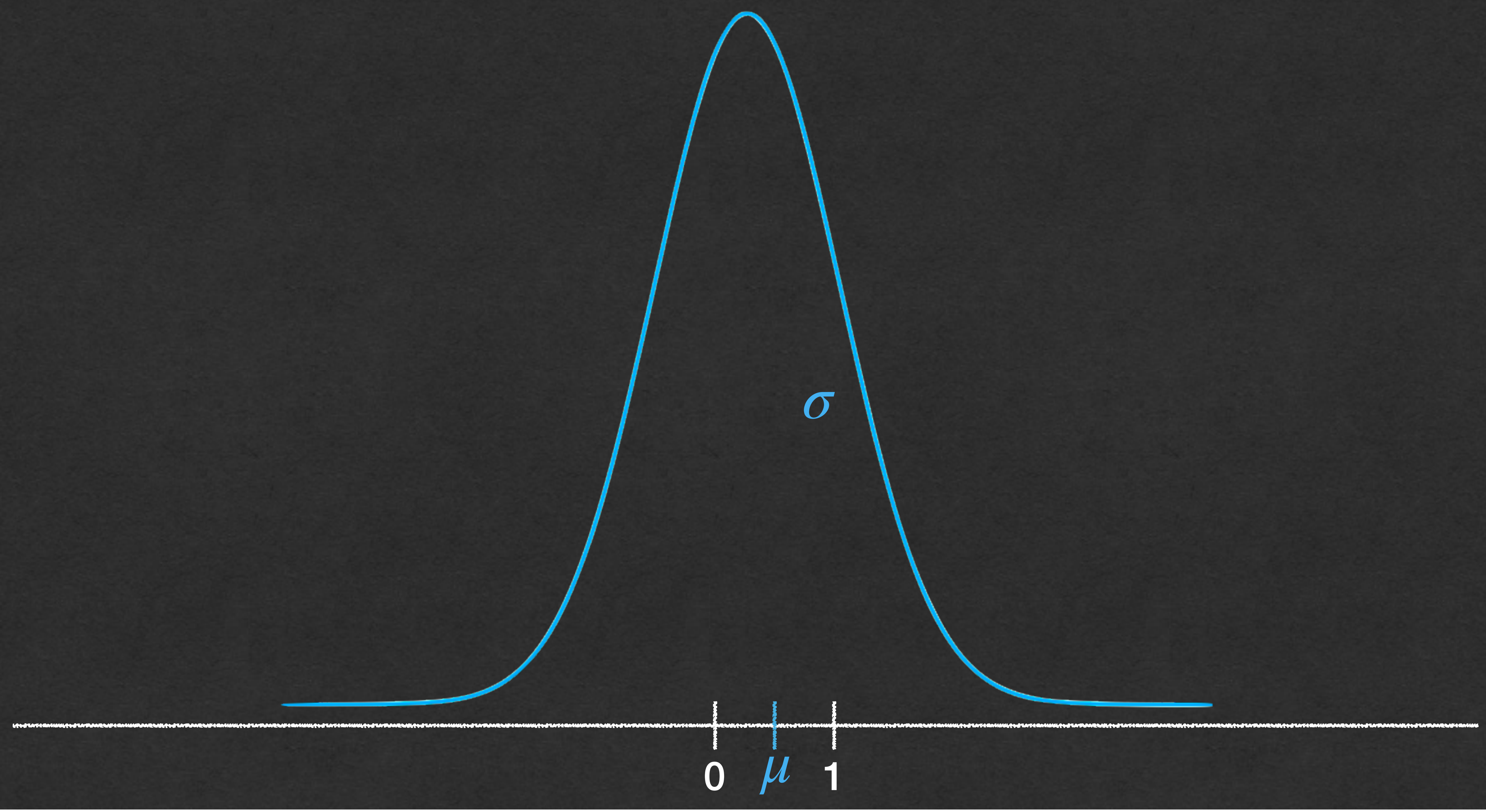
Sinha Roy, Vercauteren and
Verbauwhede SAC 2013
Hulsing, Lange and Smeets PKC 2018
Micciancio and Walter CRYPTO 2017
Karmakar et al. DAC IEEE 2019

This work: a simple alternative
dedicated to Falcon

The sampling distribution

$1.31 = \sigma_{min} \leq \sigma \leq \sigma_0 = 1.82$

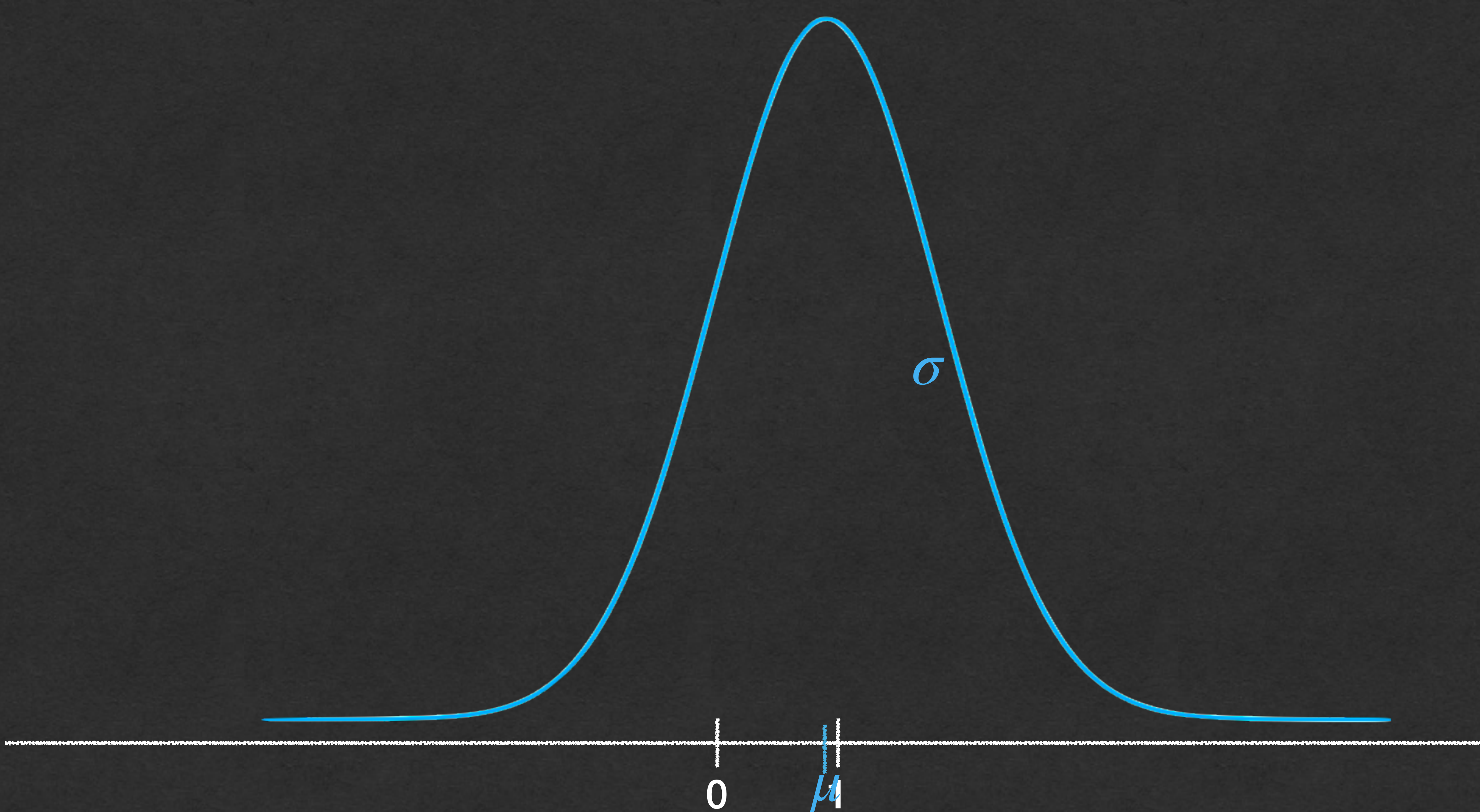
$\mu \in [0,1)$



The sampling distribution

$1.31 = \sigma_{min} \leq \sigma \leq \sigma_0 = 1.82$

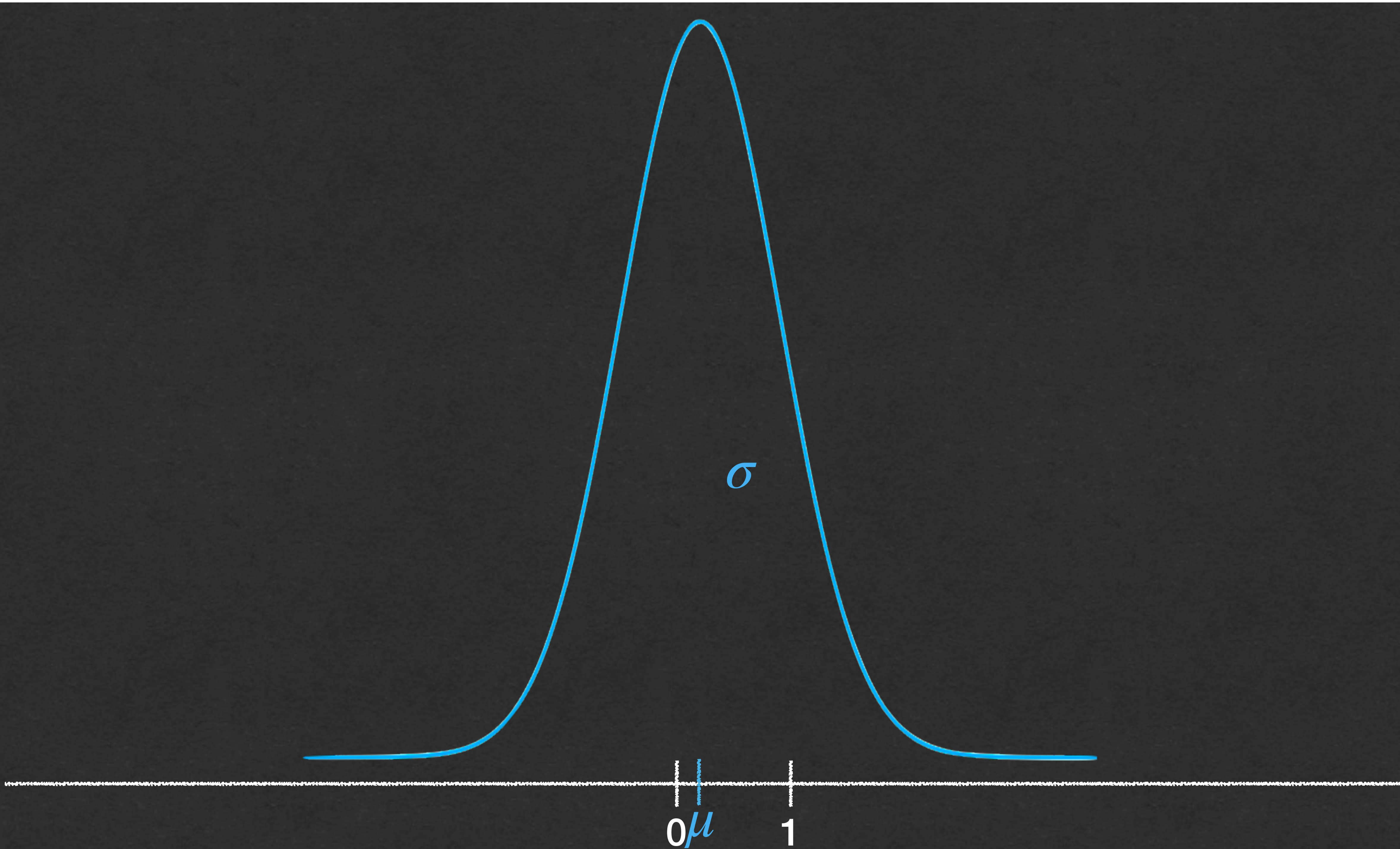
$\mu \in [0,1)$



The sampling distribution

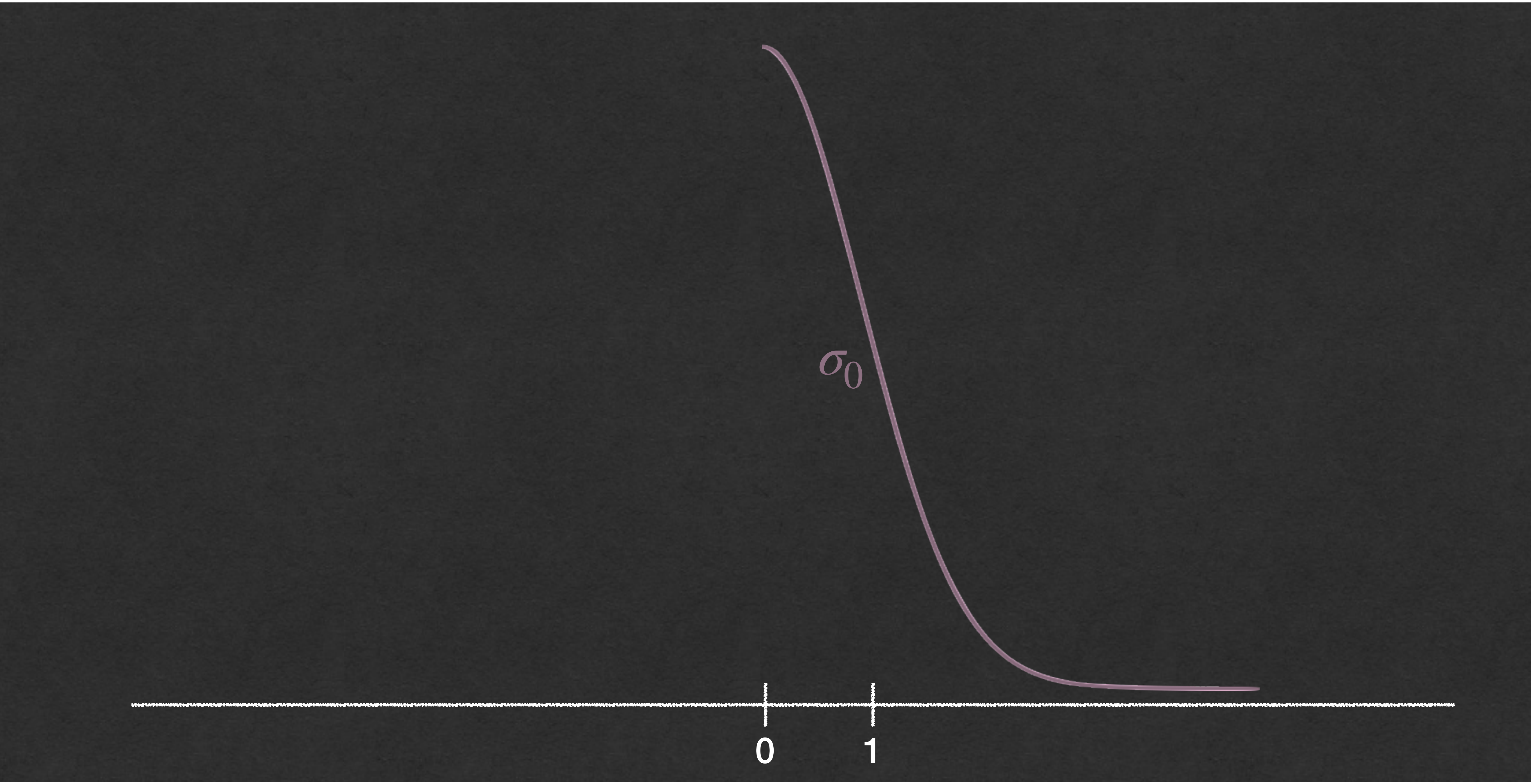
$$1.31 = \sigma_{min} \leq \sigma \leq \sigma_0 = 1.82$$

$$\mu \in [0,1)$$



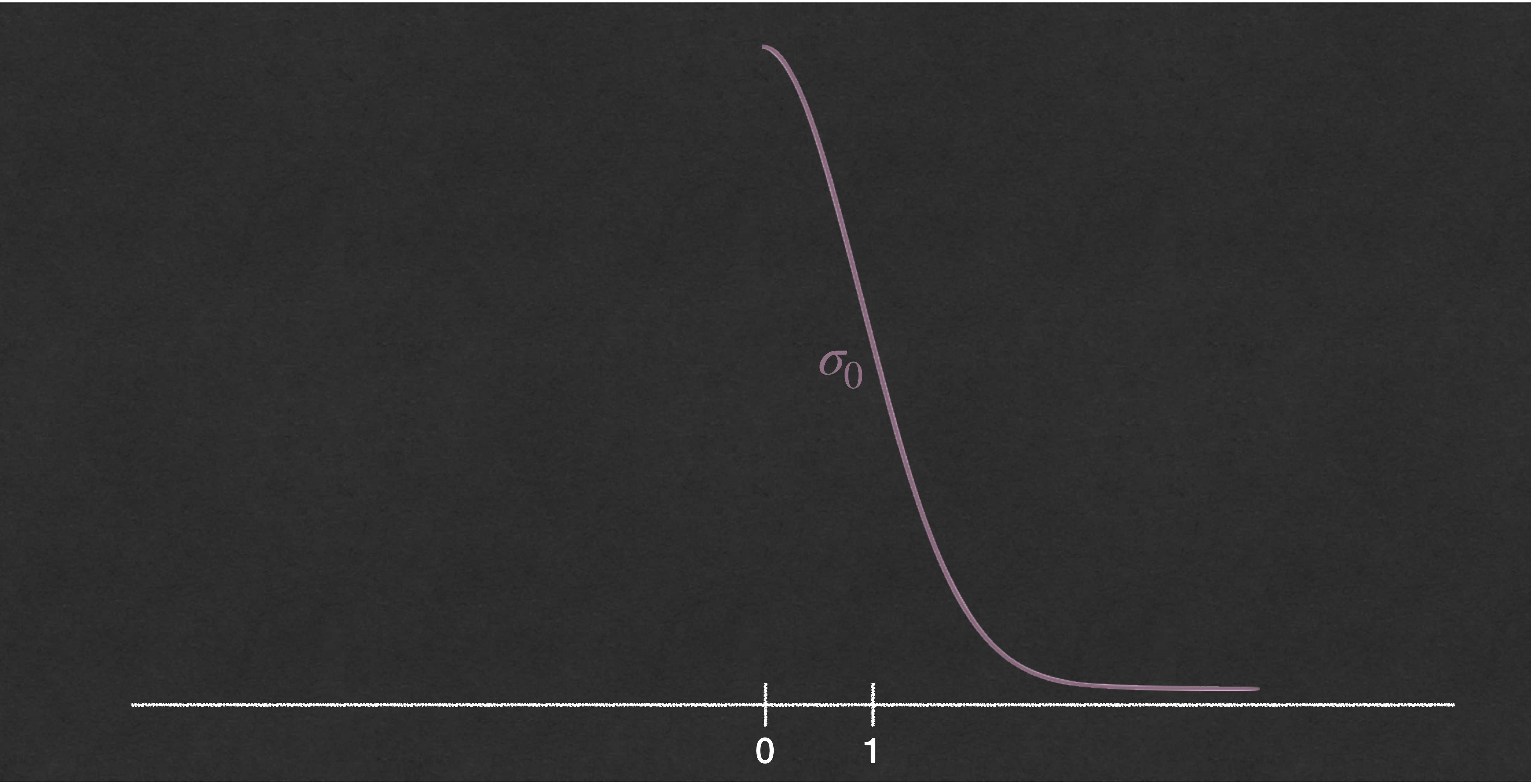
The technique

1 Draw an element z_0 from a centered half Gaussian of standard deviation σ_0



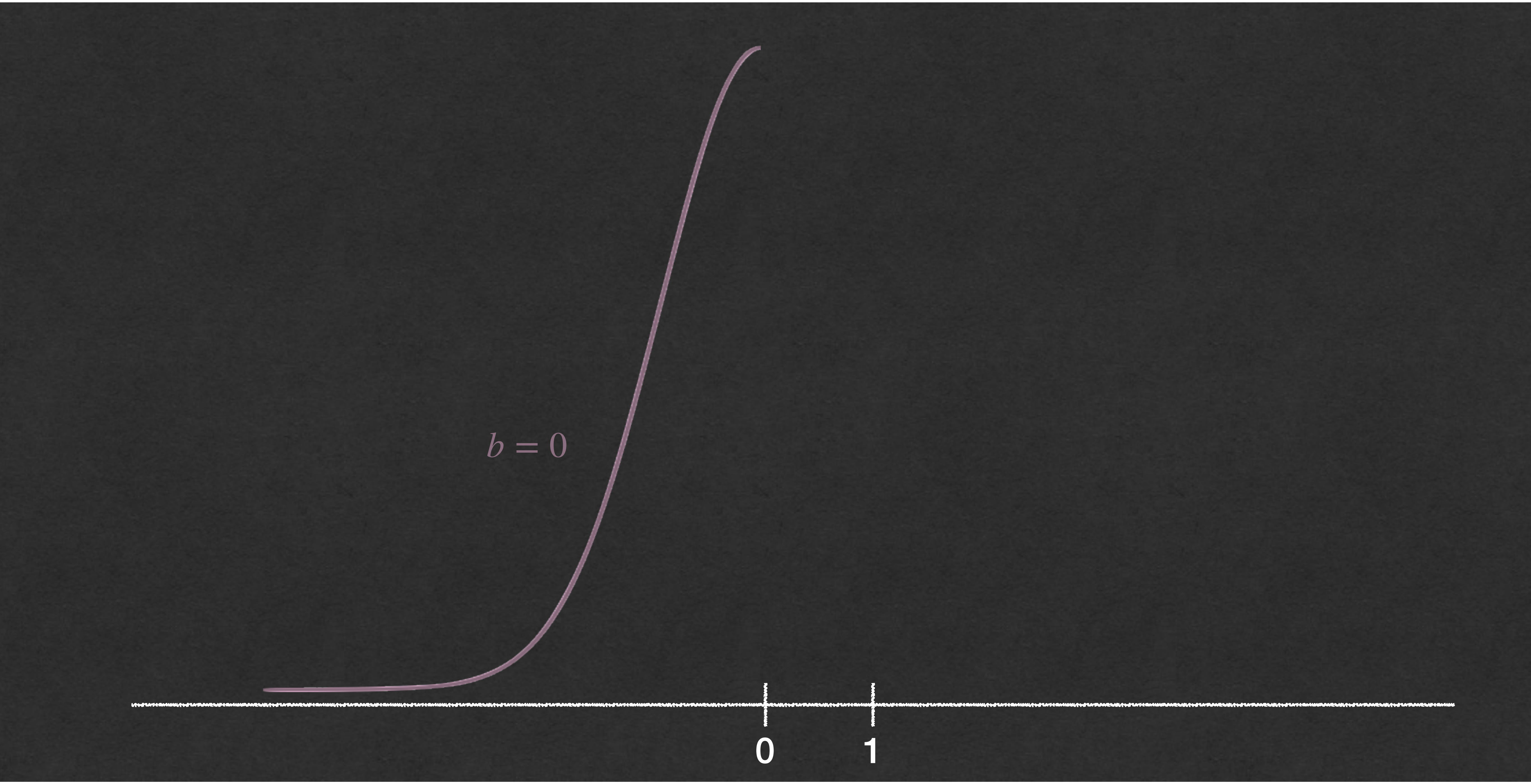
The technique

2 Draw b uniformly at random in {0,1} and compute $z \leftarrow (2b-1) \cdot z_0 + b$



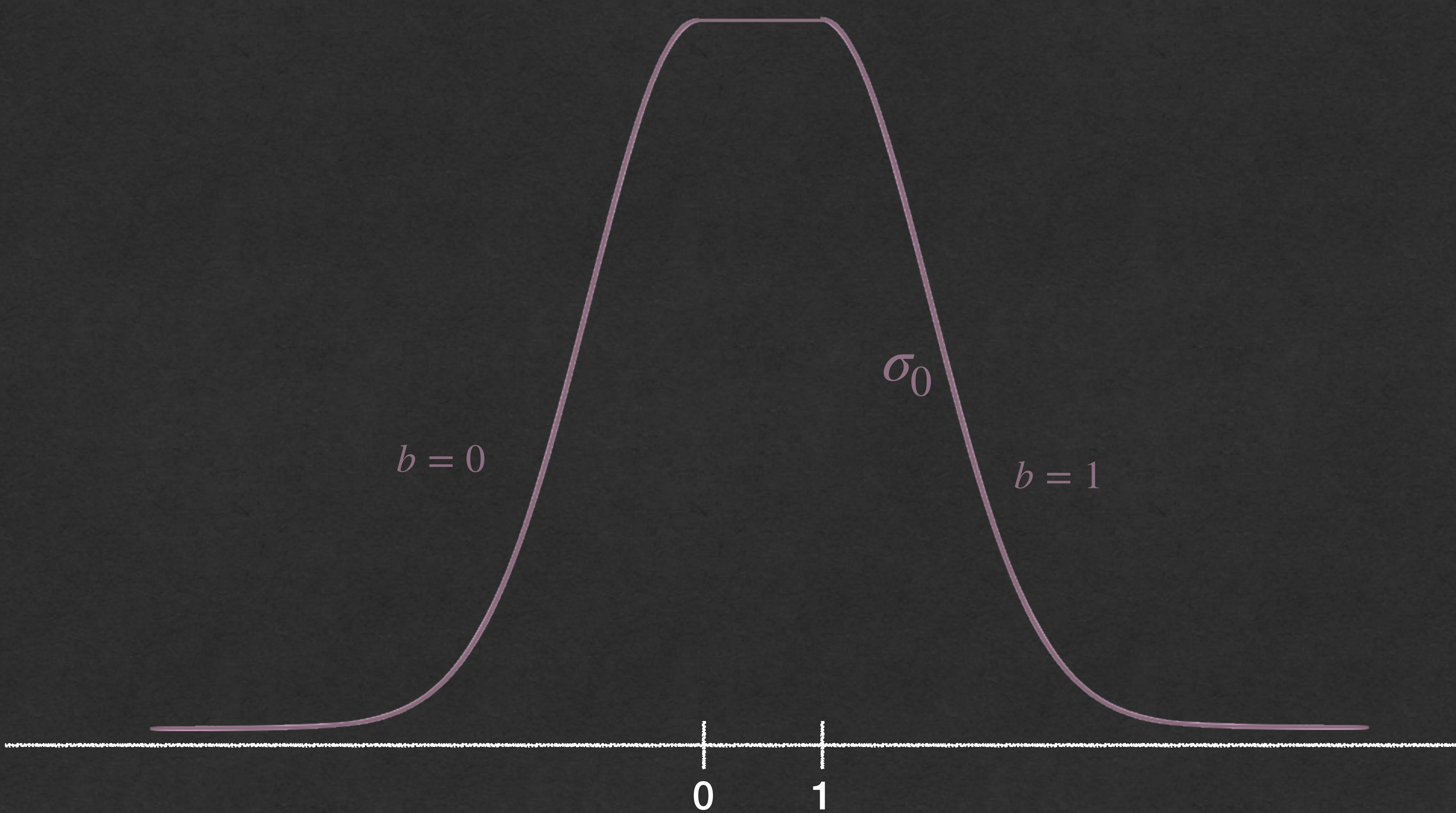
The technique

2 Draw b uniformly at random in {0,1} and compute $z \leftarrow (2b-1) \cdot z_0 + b$



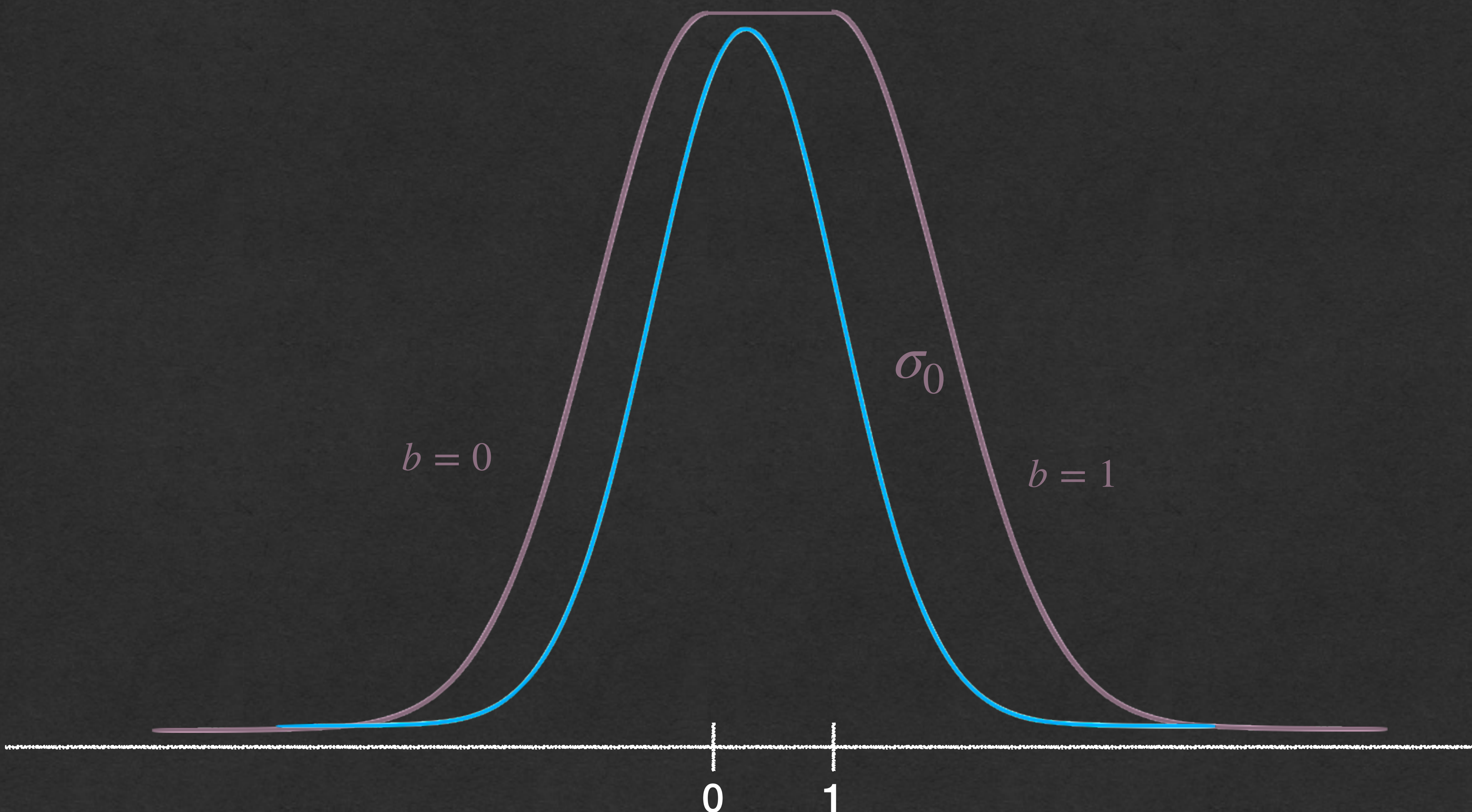
The technique

2 Draw b uniformly at random in {0,1} and compute $z \leftarrow (2b-1) \cdot z_0 + b$



The technique

3 Rejection Sampling (Lyubashevsky EC 2012) Accept with probability $P_{\text{accept}} \propto \dfrac{D_{\sigma,\mu}(z)}{G_{\mathbb{Z},\sigma_0}(z)}$



Falcon Gaussian sampler

Algorithm SampleZ(σ, μ)

Require: $\mu \in [0,1), \sigma \leq \sigma_0$

Ensure: $z \sim D_{\mathbb{Z},\sigma,\mu}$

1. $z_0 \leftarrow$ Basesampler()
2. $b \leftarrow \{0,1\}$ uniformly
3. $z \leftarrow (2b-1) \cdot z_0 + b$
4. $x \leftarrow -\dfrac{(z-\mu)^2}{2\sigma^2} + \dfrac{z_0^2}{2\sigma_0^2}$
5. Accept with probability $\exp(x)$
   Restart to 1. otherwise

Falcon Gaussian sampler

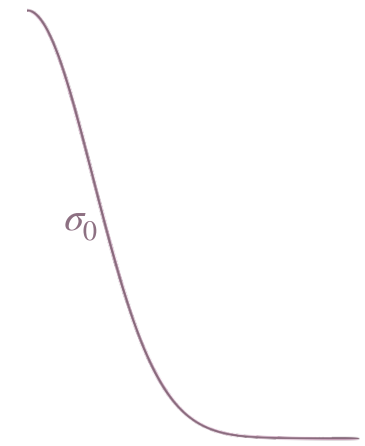
Algorithm SampleZ(σ, μ)

Require: $\mu \in [0,1), \sigma \leq \sigma_0$

Ensure: $z \sim D_{\mathbb{Z},\sigma,\mu}$

1. $z_0 \leftarrow$ Basesampler()
2. $b \leftarrow \{0,1\}$ uniformly
3. $z \leftarrow (2b-1) \cdot z_0 + b$
4. $x \leftarrow -\dfrac{(z-\mu)^2}{2\sigma^2} + \dfrac{z_0^2}{2\sigma_0^2}$
5. Accept with probability $\exp(x)$
   Restart to 1. otherwise

1.



Falcon Gaussian sampler

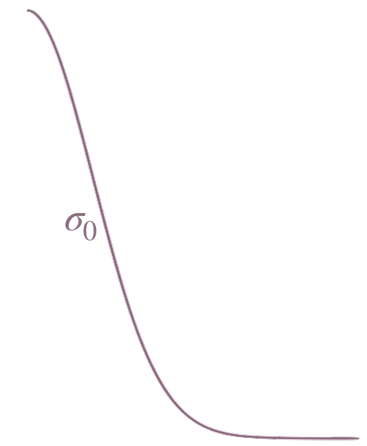
Algorithm SampleZ(σ, μ)

Require: $\mu \in [0,1), \sigma \le \sigma_0$

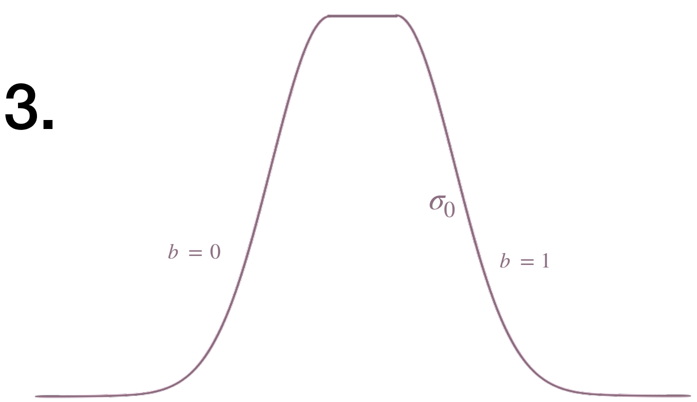
Ensure: $z \sim D_{\mathbb{Z},\sigma,\mu}$

1. $z_0 \leftarrow$ Basesampler()
2. $b \leftarrow \{0,1\}$ uniformly
3. $z \leftarrow (2b-1)\cdot z_0 + b$
4. $x \leftarrow -\dfrac{(z-\mu)^2}{2\sigma^2} + \dfrac{z_0^2}{2\sigma_0^2}$
5. Accept with probability $\exp(x)$
   Restart to 1. otherwise

1.



3.



Falcon Gaussian sampler

Algorithm SampleZ(σ, μ)

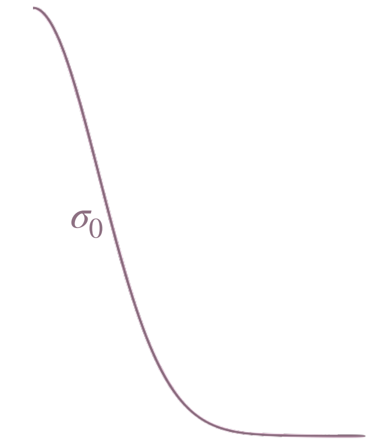
Require: $\mu \in [0,1), \sigma \leq \sigma_0$

Ensure: $z \sim D_{\mathbb{Z},\sigma,\mu}$

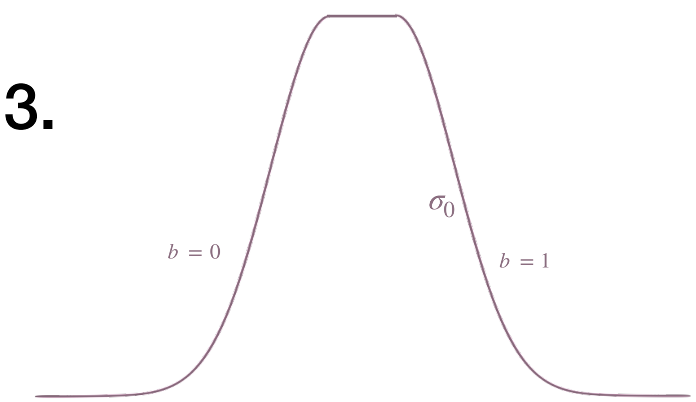
1. $z_0 \leftarrow$ Basesampler()
2. $b \leftarrow \{0,1\}$ uniformly
3. $z \leftarrow (2b-1)\cdot z_0 + b$
4. $x \leftarrow -\dfrac{(z-\mu)^2}{2\sigma^2} + \dfrac{z_0^2}{2\sigma_0^2}$
5. Accept with probability $\exp(x)$
   Restart to 1. otherwise

$$P_{\text{accept}} = \frac{\exp\left(-\frac{(z-\mu)^2}{2\sigma^2}\right)}{\exp\left(-\frac{z_0^2}{2\sigma_0^2}\right)}$$

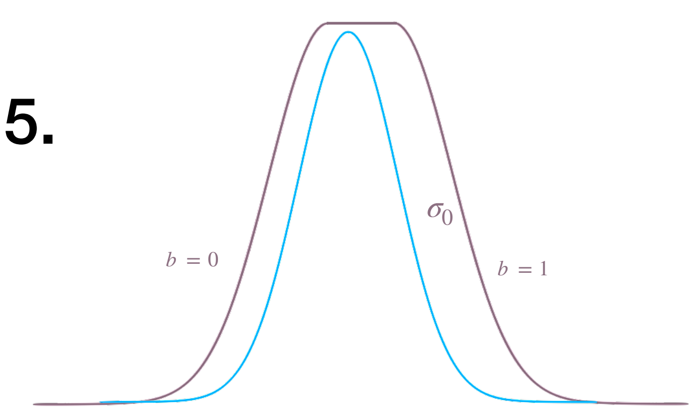
1.



3.



5.



Constant time Falcon Gaussian sampler

Algorithm SampleZ(σ, μ)

Require: $\mu \in [0,1), \sigma \leq \sigma_0$

Ensure: $z \sim D_{\mathbb{Z},\sigma,\mu}$

1. $z_0 \leftarrow$ Basesampler()
2. $b \leftarrow \{0,1\}$ uniformly
3. $z \leftarrow (2b-1) \cdot z_0 + b$
4. $x \leftarrow -\dfrac{(z-\mu)^2}{2\sigma^2} + \dfrac{z_0^2}{2\sigma_0^2}$
5. Accept with probability $\exp(x)$
   Restart to 1. otherwise

If all the distributions and computations are perfect (Basesampler(), uniform and exp()),
SampleZ$(\mu, \sigma) = D_{\mathbb{Z},\sigma,\mu}$

Constant time Falcon Gaussian sampler

Require: $\mu \in [0,1), \sigma \leq \sigma_0$

Ensure: $z \sim D_{\mathbb{Z},\sigma,\mu}$

1. $z_0 \leftarrow$ Basesampler()
2. $b \leftarrow \{0,1\}$ uniformly
3. $z \leftarrow (2b-1)\cdot z_0 + b$
4. $x \leftarrow -\dfrac{(z-\mu)^2}{2\sigma^2} + \dfrac{z_0^2}{2\sigma_0^2}$
5. Accept with probability $\exp(x)$
   Restart to 1. otherwise

Constant time and portability modifications

1) Basesampler with a table
2) Polynomial approximation for exp
3) Make the number of iterations independent from the secret

If all the distributions and computations are perfect (Basesampler(), uniform and exp()),
SampleZ$(\mu, \sigma) = D_{\mathbb{Z},\sigma,\mu}$

Constant time Falcon Gaussian sampler

Require: $\mu \in [0,1), \sigma \leq \sigma_0$

Ensure: $z \sim D_{\mathbb{Z},\sigma,\mu}$

1. $z_0 \leftarrow$ Basesampler()
2. $b \leftarrow \{0,1\}$ uniformly
3. $z \leftarrow (2b-1) \cdot z_0 + b$
4. $x \leftarrow -\dfrac{(z-\mu)^2}{2\sigma^2} + \dfrac{z_0^2}{2\sigma_0^2}$
5. Accept with probability $\exp(x)$
   Restart to 1. otherwise

Constant time and portability modifications

1) Basesampler with a table
2) Polynomial approximation for exp
3) Make the number of iterations independent from the secret

If all the distributions and computations are perfect (Basesampler(), uniform and exp()),

$$\text{SampleZ}(\mu,\sigma) \overset{?}{=} D_{\mathbb{Z},\sigma,\mu}$$

Rényi divergence result

SampleZ$(\mu, \sigma) = D_{\mathbb{Z},\sigma,\mu}$ Yes as long as the number of queries is bounded

Rényi divergence result

SampleZ$(\mu, \sigma) = D_{\mathbb{Z},\sigma,\mu}$ Yes as long as the number of queries is bounded

Security loss theorem

For at most 2^{64} signature queries,

if BaseSampler is « close » to $D_{\mathbb{Z}^+,\sigma_0}$

and

$\exp()$ replaced by a polynomial P that is also « close » to $\exp()$ on $[0, \ln(2)]$

$\Longrightarrow$ The security is preserved:

One cannot notice the changes with the output distribution

Rényi divergence result

SampleZ$(\mu, \sigma) = D_{\mathbb{Z},\sigma,\mu}$ Yes as long as the number of queries is bounded

Security loss theorem

For at most 2^{64} signature queries,

$$R_a\left(\text{BaseSampler()}, D_{\mathbb{Z}^+,\sigma_0}\right) \leq 1 + 2^{-80}$$

and $\exp()$ replaced by a polynomial P such that

$$\forall x \in [0, \ln(2)] \quad \left|\frac{P(x) - \exp(x)}{\exp(x)}\right| \leq 2^{-44}$$

$\Longrightarrow$ at most 2 bits of security are lost.

Rényi divergence result

SampleZ$(\mu, \sigma) = D_{\mathbb{Z},\sigma,\mu}$ Yes as long as the number of queries is bounded

Security loss theorem

For at most 2^{64} signature queries,

$$R_a\left(\text{BaseSampler()}, D_{\mathbb{Z}^+,\sigma_0}\right) \leq 1 + 2^{-80}$$

and $\exp()$ replaced by a polynomial P such that

$$\forall x \in [0, \ln(2)] \quad \left|\frac{P(x) - \exp(x)}{\exp(x)}\right| \leq 2^{-44}$$

$\Longrightarrow$ at most 2 bits of security are lost.

See paper for the proof.

- Application of Bai et al. ASIACRYPT 2015, Prest ASIACRYPT 2017
- Parameterized by the number of queries to the sampler

The constant time sampler

- ☐ Basesampler with a table
- ☐ Polynomial approximation for exp
- ☐ Make the number of iterations independent from the secret

1) Sampling with a table

BaseSampler() close to $D_{\mathbb{Z}^+,\sigma_0}$

Cumulative Distribution Table (CDT) with w elements of θ bits

CDT sampling can be done in constant time if the algorithm reads the entire table each time and carry out each comparison

1) Sampling with a table

BaseSampler() close to $D_{\mathbb{Z}^+,\sigma_0}$

We provide a script that generates w and the CDT table for a given target precision $\epsilon = 2^{-80}$ and θ

CDT sampling can be done in constant time if the algorithm reads the entire table each time and carry out each comparison

1) Sampling with a table

BaseSampler() close to $D_{\mathbb{Z}^+,\sigma_0}$

We provide a script that generates w and the CDT table for a given target precision $\epsilon = 2^{-80}$ and θ

Algorithm Renyification$(\sigma, \epsilon, \theta)$

Require: $\sigma, \epsilon \leq 0, \theta$

Ensure: w, the CDT table

1. $w \leftarrow$ Smallest tailcut such that $R_a\left(D_{[w],\sigma_0}, D_{\mathbb{Z}^+,\sigma_0}\right) \leq 1+\epsilon$
2. Compute the table values with a « clever » rounding
   1. For $z \geq 1$, $CDT(z) \leftarrow 2^{-\theta}\left\lfloor 2^{\theta} \cdot D_{[w],\sigma_0}(z)\right\rfloor$
   2. $CDT(0) \leftarrow 1 - \sum_{z\geq 1} CDT(z)$
3. Recompute Rényi divergence and return the new precision, w and CDT

I) CDT Sampling

$$R_\infty\left(\textbf{BaseSampler()}, D_{\mathbb{Z}^+,\sigma_0}\right) \leq 1 + 2^{-80}$$

For $\sigma_0 = 1.8205$, our script gave

$w = 19$ elements

$\theta = 72$ bits

$\epsilon = 80$

$\mathtt{CDT}(0) = 2^{-72} \times 1697680241746640300030$

$\mathtt{CDT}(1) = 2^{-72} \times 1459943456642912959616$

$\mathtt{CDT}(2) = 2^{-72} \times 928488355018011056515$

$\mathtt{CDT}(3) = 2^{-72} \times 436693944817054414619$

$\mathtt{CDT}(4) = 2^{-72} \times 151893140790369201013$

$\mathtt{CDT}(5) = 2^{-72} \times 39071441848292237840$

$\mathtt{CDT}(6) = 2^{-72} \times 7432604049020375675$

$\mathtt{CDT}(7) = 2^{-72} \times 1045641569992574730$

$\mathtt{CDT}(8) = 2^{-72} \times 108788995549429682$

$\mathtt{CDT}(9) = 2^{-72} \times 8370422445201343$

$\mathtt{CDT}(10) = 2^{-72} \times 476288472308334$

$\mathtt{CDT}(11) = 2^{-72} \times 20042553305308$

$\mathtt{CDT}(12) = 2^{-72} \times 623729532807$

$\mathtt{CDT}(13) = 2^{-72} \times 4354889437$

$\mathtt{CDT}(14) = 2^{-72} \times 244322621$

$\mathtt{CDT}(15) = 2^{-72} \times 3075302$

$\mathtt{CDT}(16) = 2^{-72} \times 28626$

$\mathtt{CDT}(17) = 2^{-72} \times 197$

$\mathtt{CDT}(18) = 2^{-72} \times 1$

The constant time sampler

- [x] Basesampler with a table
- [ ] Polynomial approximation for exp
- [ ] Make the number of iterations independent from the secret

2) Polynomial approximation

Find P such that $$\left|\frac{P(x) - \exp(x)}{\exp(x)}\right| \leq 2^{-44} \quad \forall x \in [0, \ln(2)]$$

Polynomial approximation tools

- Floating points option: FACCT by Zhao, Steinfeld and Sakzad 2018/1234

2) Polynomial approximation

Find P such that $$\left|\frac{P(x)-\exp(x)}{\exp(x)}\right| \leq 2^{-44} \quad \forall x \in [0, \ln(2)]$$

Polynomial approximation tools

- Floating points option: FACCT by Zhao, Steinfeld and Sakzad 2018/1234
- Integer option: GALACTICS by Barthe et al. 2019/511

$\Longrightarrow$ 32-bit coefficients
degree 10

2) Polynomial approximation

Find P such that $\left| \frac{P(x) - \exp(x)}{\exp(x)} \right| \leq 2^{-44} \quad \forall x \in [0, \ln(2)]$

Polynomial approximation tools

- Floating points option: FACCT by Zhao, Steinfeld and Sakzad 2018/1234
- Integer option: GALACTICS by Barthe et al. 2019/511

$\Longrightarrow$ 32-bit coefficients
degree 10

Depending on the architecture, several tradeoffs

2) Polynomial approximation

Find P such that $$\left|\frac{P(x)-\exp(x)}{\exp(x)}\right| \leq 2^{-44} \quad \forall x \in [0, \ln(2)]$$

Polynomial approximation tools

- Floating points option: FACCT by Zhao, Steinfeld and Sakzad 2018/1234
- Integer option: GALACTICS by Barthe et al. 2019/511

$\Longrightarrow$ 32-bit coefficients
degree 10

Depending on the architecture, several tradeoffs

Degree

2) Polynomial approximation

Find P such that $$\left|\frac{P(x) - \exp(x)}{\exp(x)}\right| \leq 2^{-44} \quad \forall x \in [0, \ln(2)]$$

Polynomial approximation tools

- Floating points option: FACCT by Zhao, Steinfeld and Sakzad 2018/1234
- Integer option: GALACTICS by Barthe et al. 2019/511

$\Longrightarrow$ 32-bit coefficients
degree 10

Depending on the architecture, several tradeoffs

Degree Size

2) Polynomial approximation

Find P such that $\left| \dfrac{P(x) - \exp(x)}{\exp(x)} \right| \leq 2^{-44} \quad \forall x \in [0, \ln(2)]$

Polynomial approximation tools

- Floating points option: FACCT by Zhao, Steinfeld and Sakzad 2018/1234
- Integer option: GALACTICS by Barthe et al. 2019/511

$\Longrightarrow$ 32-bit coefficients
degree 10

Depending on the architecture, several tradeoffs

Degree | Size | Depth

The constant time sampler

- ☑ Basesampler with a table
- ☑ Polynomial approximation for exp
- ☐ Make the number of iterations independent from the secret

3) Number of iterations of the while loop

?

Zhao, Steinfeld and Sakzad (2018/1234)
Karmakar et al (2019/267)

- Could the number of iterations leak the secret?

3) Number of iterations of the while loop

? Zhao, Steinfeld and Sakzad (2018/1234)
Karmakar et al (2019/267)

- Could the number of iterations leak the secret?

The number of iterations follows a geometric distribution of average $\dfrac{2 \cdot \rho_{\sigma_0}(\mathbb{Z}^+)}{\rho_{\sigma,\mu}(\mathbb{Z})}$

3) Number of iterations of the while loop

?

Zhao, Steinfeld and Sakzad (2018/1234)
Karmakar et al (2019/267)

- Could the number of iterations leak the secret?

The average number of iterations is $\dfrac{2 \cdot \rho_{\sigma_0}(\mathbb{Z}^+)}{\frac{\sigma_{min}}{\sigma}\rho_{\sigma,\mu}(\mathbb{Z})}$

The acceptance probability P_{accept} is scaled by a factor $\dfrac{\sigma_{min}}{\sigma} \leq \dfrac{\sigma_{min}}{\sigma_{max}} \approx 0.73$

3) Number of iterations of the while loop

? Zhao, Steinfeld and Sakzad (2018/1234)
Karmakar et al (2019/267)

- Could the number of iterations leak the secret?

The average number of iterations is $\dfrac{2 \cdot \rho_{\sigma_0}(\mathbb{Z}^+)}{\frac{\sigma_{min}}{\sigma}\rho_{\sigma,\mu}(\mathbb{Z})}$

The acceptance probability P_{accept} is scaled by a factor $\dfrac{\sigma_{min}}{\sigma} \leq \dfrac{\sigma_{min}}{\sigma_{max}} \approx 0.73$

Indeed, with a Poisson summation (under a Rényi divergence argument),

$$\rho_{\sigma,\mu}(\mathbb{Z}) \approx \sigma\sqrt{2\pi}$$

So, $$\frac{2 \cdot \rho_{\sigma_0}(\mathbb{Z}^+)}{\frac{\sigma_{min}}{\sigma}\rho_{\sigma,\mu}(\mathbb{Z})} \approx \frac{2 \cdot \rho_{\sigma_0}(\mathbb{Z}^+)}{\frac{\sigma_{min}}{\sigma}\sigma\sqrt{2\pi}} = \frac{2 \cdot \rho_{\sigma_0}(\mathbb{Z}^+)}{\sigma_{min}\sqrt{2\pi}}$$

3) Number of iterations of the while loop

? Zhao, Steinfeld and Sakzad (2018/1234)
Karmakar et al (2019/267)

- Could the number of iterations leak the secret?

The average number of iterations is $\dfrac{2 \cdot \rho_{\sigma_0}(\mathbb{Z}^+)}{\frac{\sigma_{min}}{\sigma}\rho_{\sigma,\mu}(\mathbb{Z})}$

The acceptance probability P_{accept} is scaled by a factor $\dfrac{\sigma_{min}}{\sigma} \leq \dfrac{\sigma_{min}}{\sigma_{max}} \approx 0.73$

Indeed, with a Poisson summation (under a Rényi divergence argument),

$$\rho_{\sigma,\mu}(\mathbb{Z}) \approx \sigma\sqrt{2\pi}$$

So, $\dfrac{2 \cdot \rho_{\sigma_0}(\mathbb{Z}^+)}{\frac{\sigma_{min}}{\sigma}\rho_{\sigma,\mu}(\mathbb{Z})} \approx \dfrac{2 \cdot \rho_{\sigma_0}(\mathbb{Z}^+)}{\frac{\sigma_{min}}{\sigma}\sigma\sqrt{2\pi}} = \dfrac{2 \cdot \rho_{\sigma_0}(\mathbb{Z}^+)}{\sigma_{min}\sqrt{2\pi}}$

- ✓ Independent from μ
- ✓ Independent from σ
- ✓ Independent from z

3) Number of iterations of the while loop

?

Zhao, Steinfeld and Sakzad (2018/1234)
Karmakar et al (2019/267)

- Could the number of iterations leak the secret?

The average number of iterations is $\dfrac{2 \cdot \rho_{\sigma_0}(\mathbb{Z}^+)}{\frac{\sigma_{min}}{\sigma}\rho_{\sigma,\mu}(\mathbb{Z})}$

Tweak for Falcon's sampler

The acceptance probability P_{accept} is scaled by a factor $\dfrac{\sigma_{min}}{\sigma} \leq \dfrac{\sigma_{min}}{\sigma_{max}} \approx 0.73$

Indeed, with a Poisson summation (under a Rényi divergence argument),

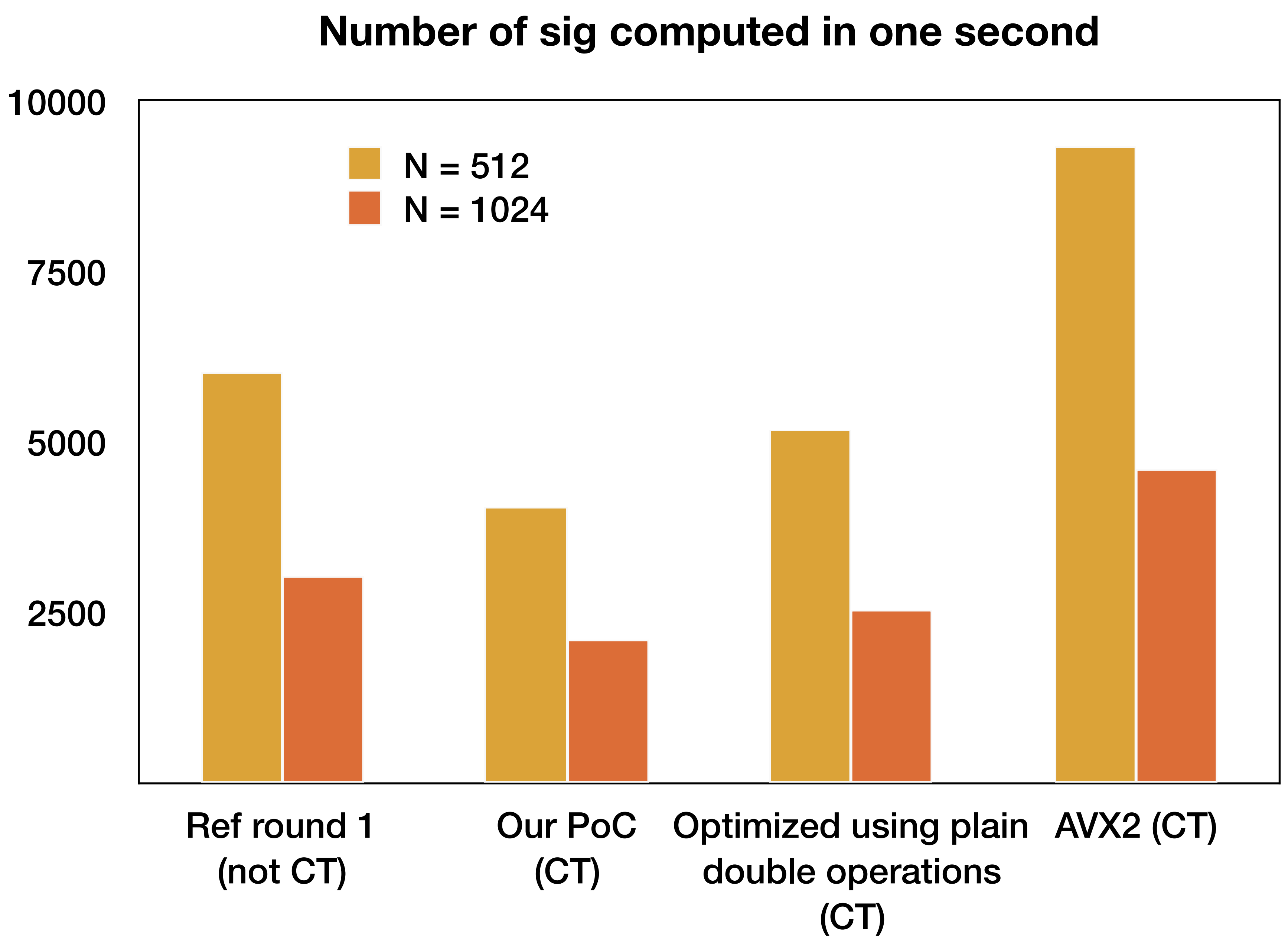
$$\rho_{\sigma,\mu}(\mathbb{Z}) \approx \sigma\sqrt{2\pi}$$

So, $$\frac{2 \cdot \rho_{\sigma_0}(\mathbb{Z}^+)}{\frac{\sigma_{min}}{\sigma}\rho_{\sigma,\mu}(\mathbb{Z})} \approx \frac{2 \cdot \rho_{\sigma_0}(\mathbb{Z}^+)}{\frac{\sigma_{min}}{\sigma}\sigma\sqrt{2\pi}} = \frac{2 \cdot \rho_{\sigma_0}(\mathbb{Z}^+)}{\sigma_{min}\sqrt{2\pi}}$$

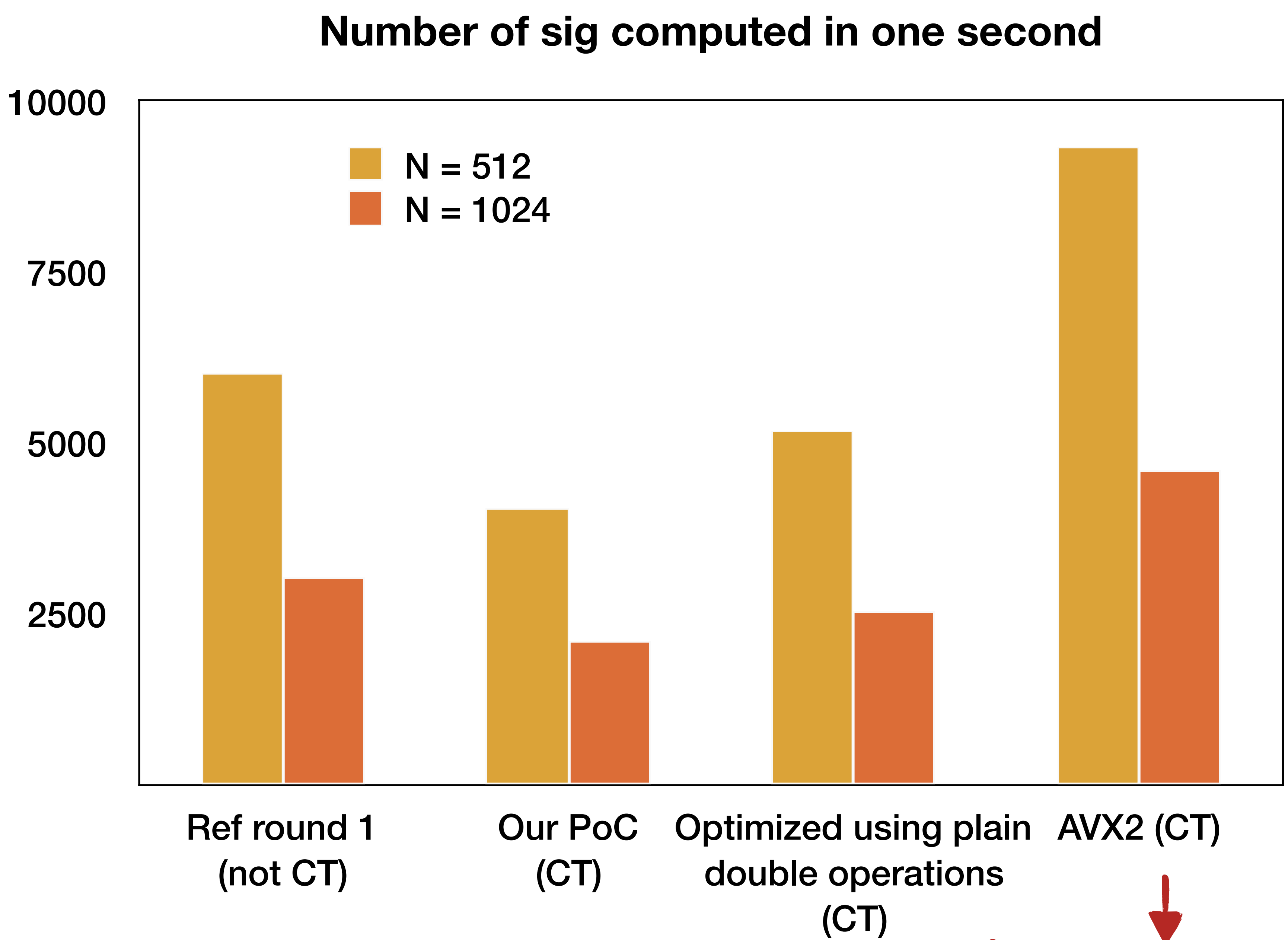
- ✓ Independent from μ
- ✓ Independent from σ
- ✓ Independent from z

The whole algorithm
is constant time

Implementations



Implementations

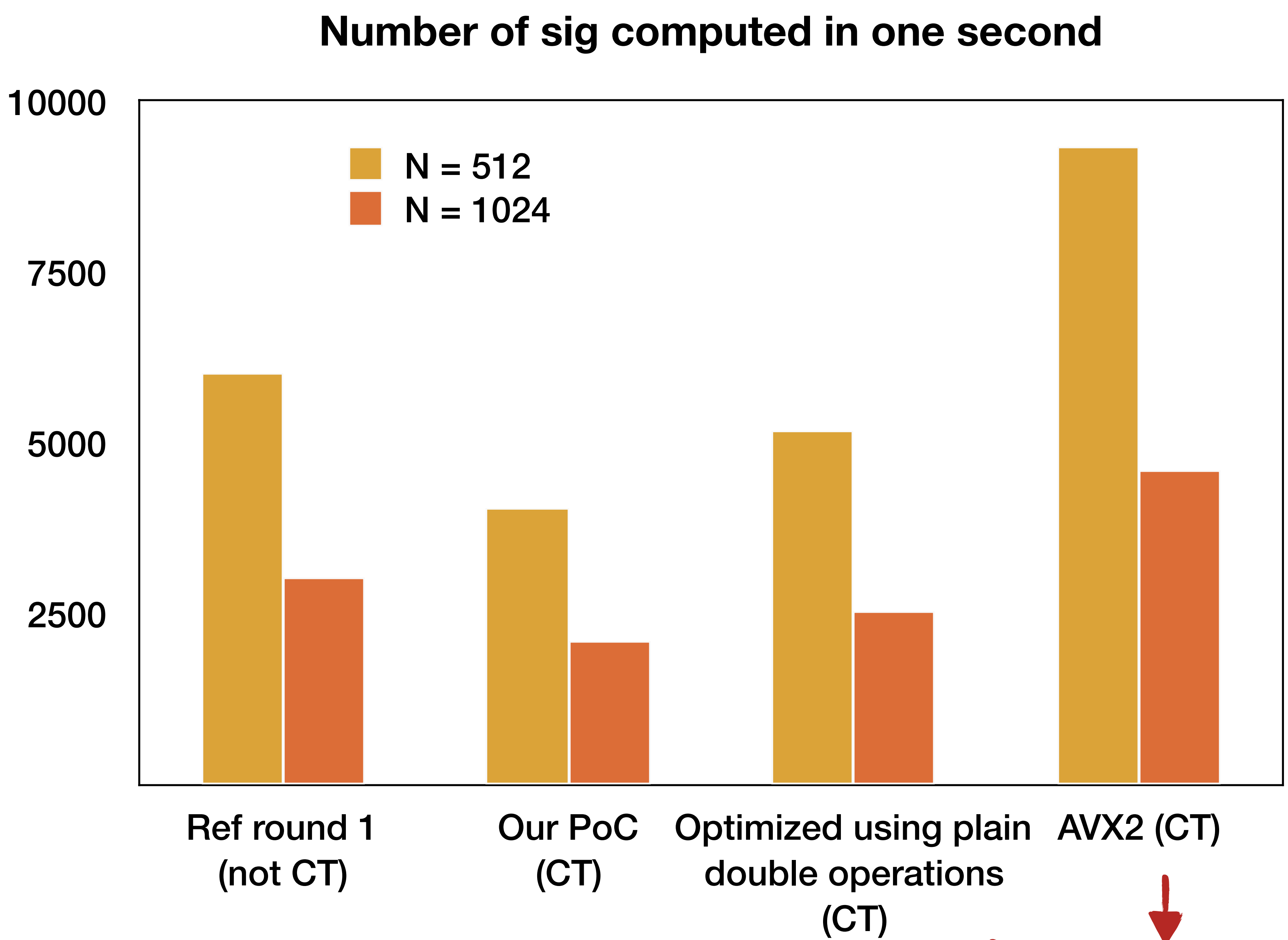


Very recent implementations done by Thomas Pornin

See https://github.com/PQClean/PQClean/

And https://falcon-sign.info/falcon-impl-20190802.pdf

Implementations



The performance loss for constant time and portability is acceptable

Very recent implementations done by Thomas Pornin

See https://github.com/PQClean/PQClean/

And https://falcon-sign.info/falcon-impl-20190802.pdf

Implementations

Constant time and integers help Cortex M4 implementations

Falcon-512 (168 MHz)	Dynamic signatures (in milliseconds)	Memory (in bytes of extra RAM, not counting the key)
First M4 implementation (Oder et al. PQCRYPTO 2019)	479	50508
Recent Constant time and integers (Thomas Pornin) https://github.com/mupq/pqm4	243	36864

Conclusion

- ☑ Compact
- ☑ Fast
- ☑ GPV framework proved secure

Conclusion

- ☑ Compact
- ☑ Fast
- ☑ GPV framework proved secure

- ☑ Constant time and still fast
- ☑ Integer arithmetic and still fast
- ☑ Solid implementations available
  (Thanks to Thomas Pornin)

Conclusion

- ☑ Compact
- ☑ Fast
- ☑ GPV framework proved secure

- ☑ Constant time and still fast
- ☑ Integer arithmetic and still fast
- ☑ Solid implementations available (Thanks to Thomas Pornin)

❑ Currently studied: masking protection

Conclusion

- ☑ Compact
- ☑ Fast
- ☑ GPV framework proved secure

- ☑ Constant time and still fast
- ☑ Integer arithmetic and still fast
- ☑ Solid implementations available (Thanks to Thomas Pornin)

☐ Currently studied: masking protection

Paper available at:
https://csrc.nist.gov/CSRC/media/Events/Second-PQC-Standardization-Conference/documents/accepted-papers/rossi-simple-fast-constant.pdf