

# Witch hunt!

demystifying lattice security

#### 4. Parameter Sets

Table 1. ML-DSA parameter sets

Parameters (see Sections 6.1 and 6.2 of this document)	ML-DSA-44	ML-DSA-65	ML-DSA-87
$q$ - modulus [see §6.1]	8380417	8380417	8380417
$\zeta$ - a 512th root of unity in $\mathbb{Z}_q$ [see §7.5]	1753	1753	1753
$d$ - # of dropped bits from $t$ [see §6.1]	13	13	13
$\tau$ - # of $\pm 1$ 's in polynomial $c$ [see §6.2]	39	49	60
$\lambda$ - collision strength of $\tilde{c}$ [see §6.2]	128	192	256
$\gamma_1$ - coefficient range of $y$ [see §6.2]	$2^{17}$	$2^{19}$	$2^{19}$
$\gamma_2$ - low-order rounding range [see §6.2]	$(q-1)/88$	$(q-1)/32$	$(q-1)/32$
$(k, \ell)$ - dimensions of $A$ [see §6.1]	(4,4)	(6,5)	(8,7)
$\eta$ - private key range [see §6.1]	2	4	2
$\beta = \tau \cdot \eta$ [see §6.2]	78	196	120
$\omega$ - max # of 1's in the hint $h$ [see §6.2]	80	55	75
Challenge entropy $\log_2(\frac{256}{\tau}) + \tau$ [see §6.2]	192	225	257
Repetitions (see explanation below)	4.25	5.1	3.85

Three parameter sets are defined:  
 public parameters  
 public keys  
 private keys  
 algorithms

are the dimensions of the matrix  $A$ .

These parameter sets were designed to meet certain security strength categories defined by NIST in its original Call for Proposals [26]. These security strength categories are explained further in SP 800-57, Part 1 [9].

Using this approach, security strength is not described by a single number, such as "128 bits of security." Instead, each ML-DSA parameter set is claimed to be at least as secure as a generic block cipher with a prescribed key size or a generic hash function with a prescribed output length. More precisely, it is claimed that the computational resources needed to break ML-DSA are greater than or equal to the computational resources needed to break the block cipher or hash function when these computational resources are estimated using any realistic model of computation. Different models of computation can be more or less realistic and, accordingly, lead to more or less accurate estimates of security strength. Some commonly studied models are discussed in [27].

Concretely, the parameter set ML-DSA-44 is claimed to be in security strength category 2, ML-DSA-65 is claimed to be in category 3, and ML-DSA-87 is claimed to be in category 5 [6]. For additional discussion of the security strength of MLWE-based cryptosystems, see [28].

The sizes of keys and signatures that correspond to each parameter set are given in Table 2. Certain optimizations are possible when storing ML-DSA public and private keys. If additional space is available, one can precompute and store  $\hat{A}$  to speed up signing and verifying. Alternatively, if one wants to reduce

#### 6 Faster tuple lattice sieving using spherical LSF

6

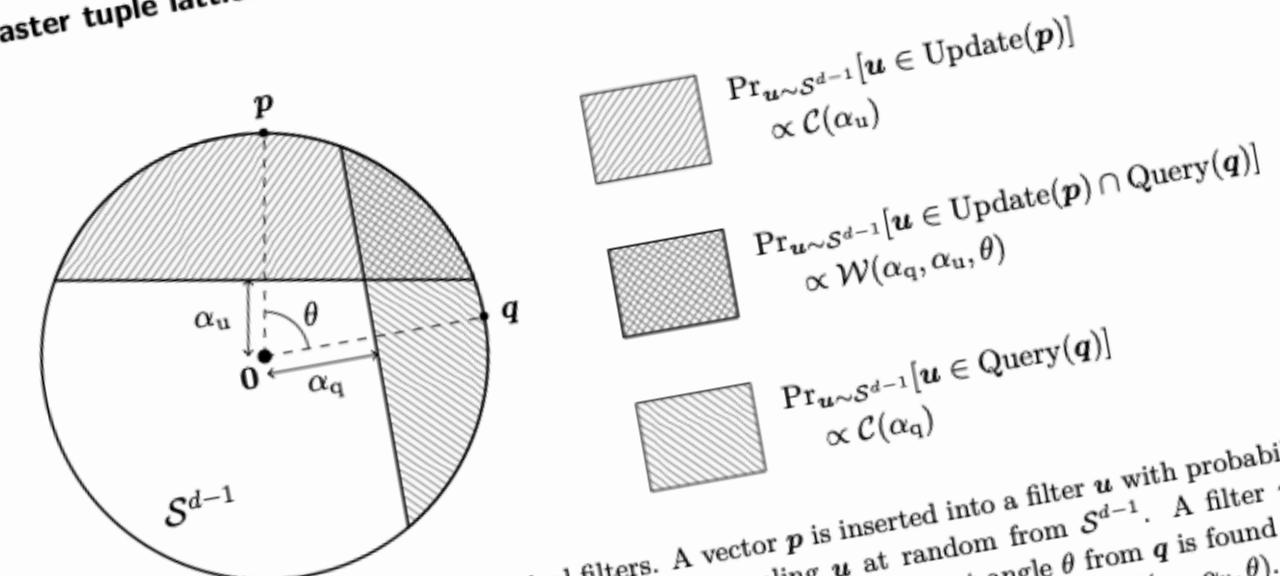


Figure 1 The geometry of spherical filters. A vector  $p$  is inserted into a filter  $u$  with probability proportional to  $C(\alpha_u)$ , over the randomness of sampling  $u$  at random from  $S^{d-1}$ . A filter  $u$  is queried for near neighbors for  $q$  with probability proportional to  $C(\alpha_q)$ . A vector  $p$  at angle  $\theta$  from  $q$  is found as a candidate nearest neighbor in one of the filters with probability proportional to  $W(\alpha_q, \alpha_u, \theta)$ .

To balance the query costs of (1) finding the  $f \cdot C(\alpha_u)$  relevant filters, and (2) going through  $f \cdot n \cdot C(\alpha_q) \cdot C(\alpha_u)$  false positives, we choose  $\alpha_u$  such that  $n \cdot C(\alpha_u) = 1$ , i.e. we set  $\alpha_u = \sqrt{1 - n^{-2/d}}$ . We further set  $\alpha_q = \beta \alpha_u$ , so that only  $\beta$  remains to be chosen, and  $\beta > 1$  leads to a better space complexity, and  $\beta < 1$  leads to a better query/update tradeoff: For  $\beta < 1$  we get the results from [9]. For  $\beta = 1$  we get the results from [9]. Complexity scales as  $n \cdot f \cdot C(\alpha_u) \sim n$ , which increasing  $\beta$  would only lead to a worse space complexity. To obtain the

# What kind of black magic is that ? !

```
Inp
Inpu
begin
    T^e ←
        query
        if a^(e, e)
            return
    while T^e
        T^e ← a^(e, e)
        query B^e X^e
        if a^(e, e)
            return (a^e, e)
        if T^e
            a^(e, e)
            ref
        else
            e ←
```

is [concrete] using spherical locality-sensitive filters with  $\alpha_u$  using spherical filters with update and query exponents  $\beta$  the  $\theta$ -near neighbor problem with update and query exponents  $\beta$  (5)

$$\rho_u = \log \left[ 1 - \left( 1 - n^{-2/d} \right) \frac{1 + \beta^2 - 2\beta \cos \theta}{\sin^2 \theta} \right] / \log(n^{-2/d}) - 1, \quad (6)$$

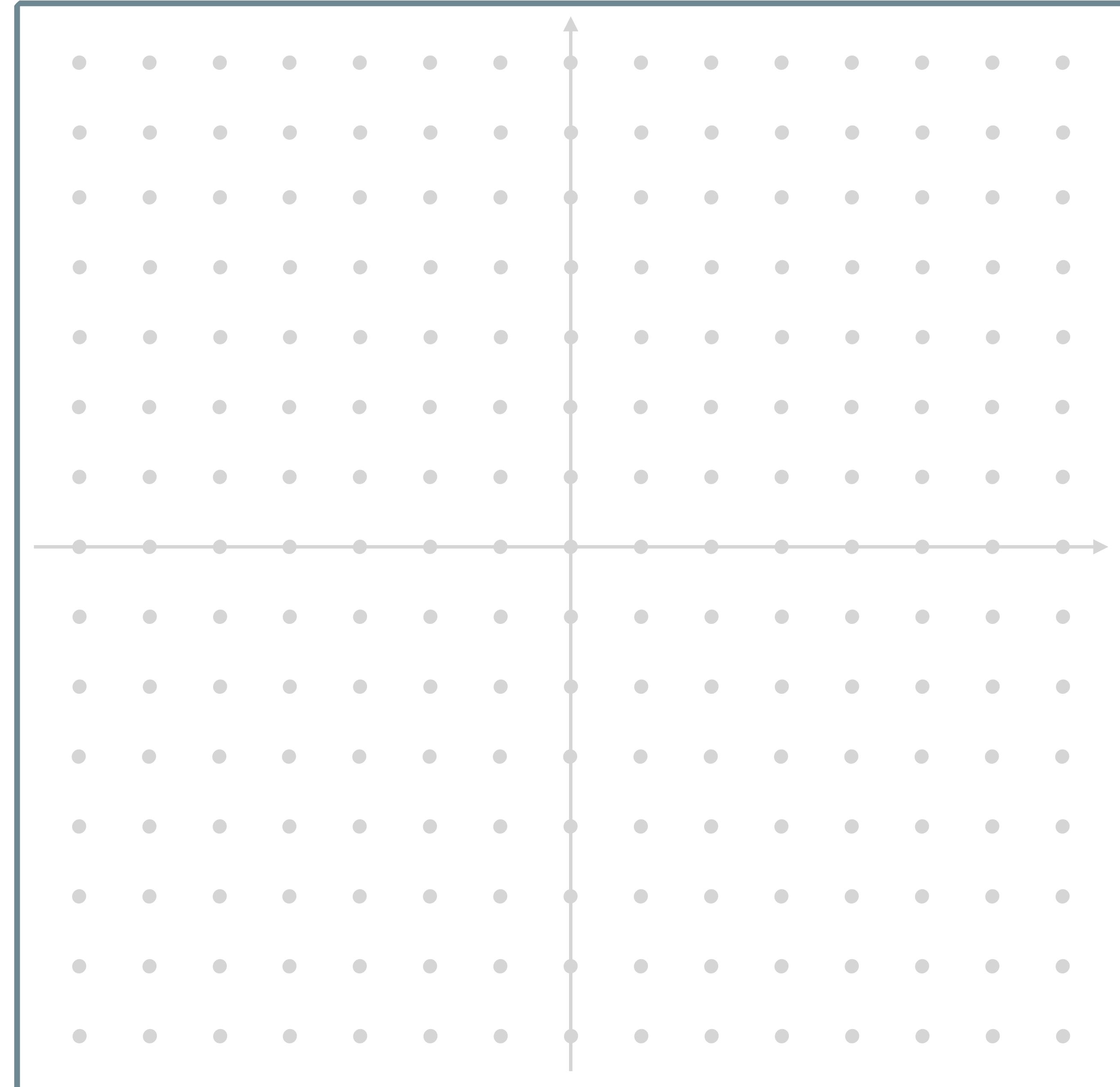
$$\rho_q = \log \left[ 1 - \left( 1 - n^{-2/d} \right) \frac{1 + \beta^2 - 2\beta \cos \theta}{\sin^2 \theta} \right] / \log(n^{-2/d}) - \log \left[ 1 - \left( 1 - n^{-2/d} \right) \beta^2 \right] / \log(n^{-2/d}). \quad (7)$$

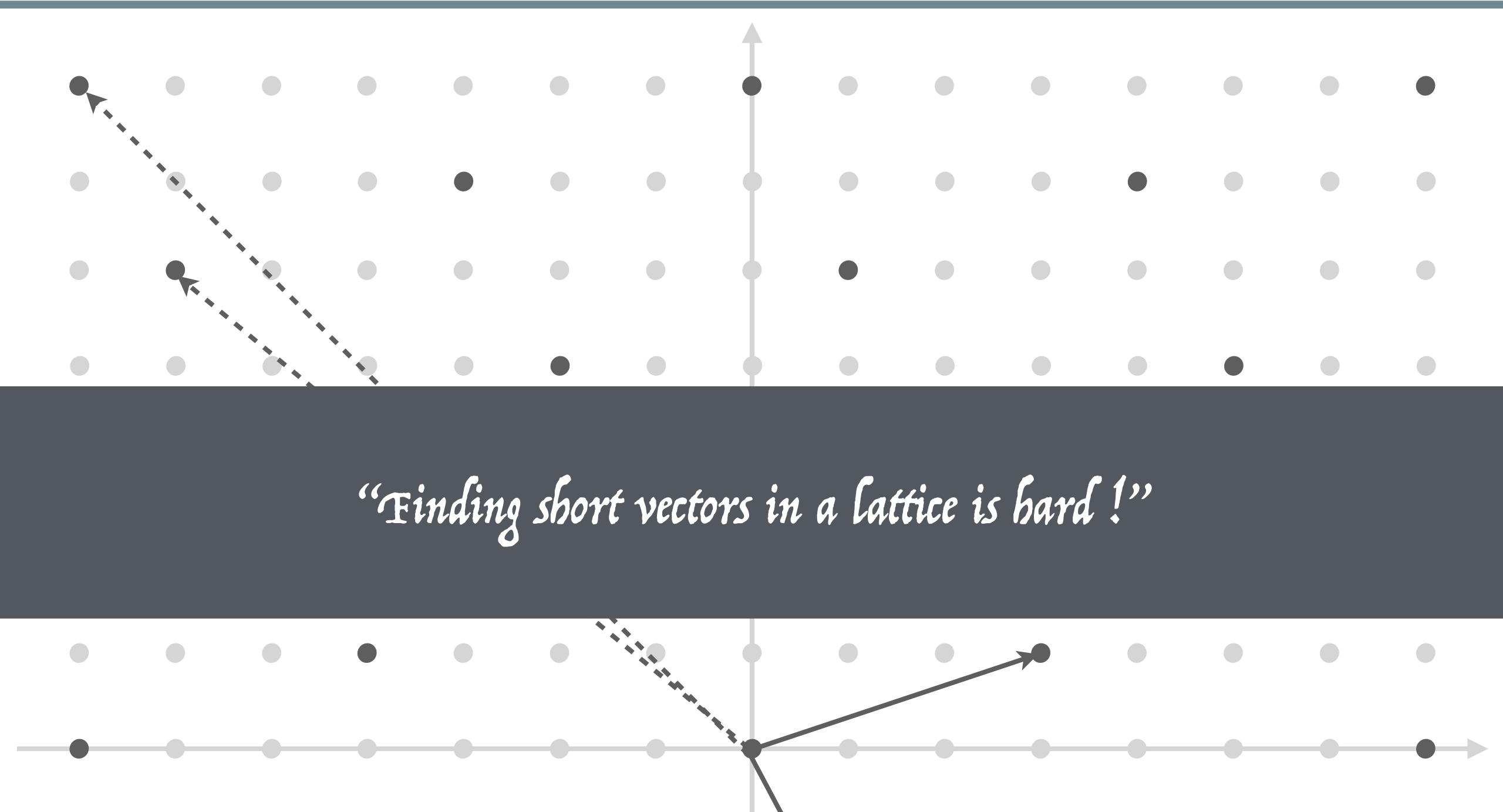
This data structure requires  $n^{1+\rho_u+o(1)}$  memory, can be initialized in time  $n^{1+\rho_u+o(1)}$ , allows for updates in time  $n^{\rho_u+o(1)}$ , and answers queries in time  $n^{\rho_u+o(1)}$ . In the limit of  $n^{1/d} = 1 + \varepsilon$  with  $\varepsilon \rightarrow 0$ , corresponding to random data sets in the sparse regime, a Taylor expansion around  $\varepsilon = 0$  of  $\rho_q$  and  $\rho_u$  yields:

$$\rho_u = \frac{(\beta - \cos \theta)^2}{\sin^2 \theta} + o(1), \quad \rho_q = \frac{(1 - \beta \cos \theta)^2}{\sin^2 \theta} + o(1).$$

This leads to the concise defining equation  $\sqrt{\rho_q} + \cos \theta \sqrt{\rho_u} = \sin \theta$  for the sparse regime, which is equivalent to [5]. Equation (1) after substituting  $\cos \theta = 1 - 1/c^2$ . This also clearly shows how to set  $\beta$  to minimize  $\rho_u$  or  $\rho_q$  in the sparse regime.

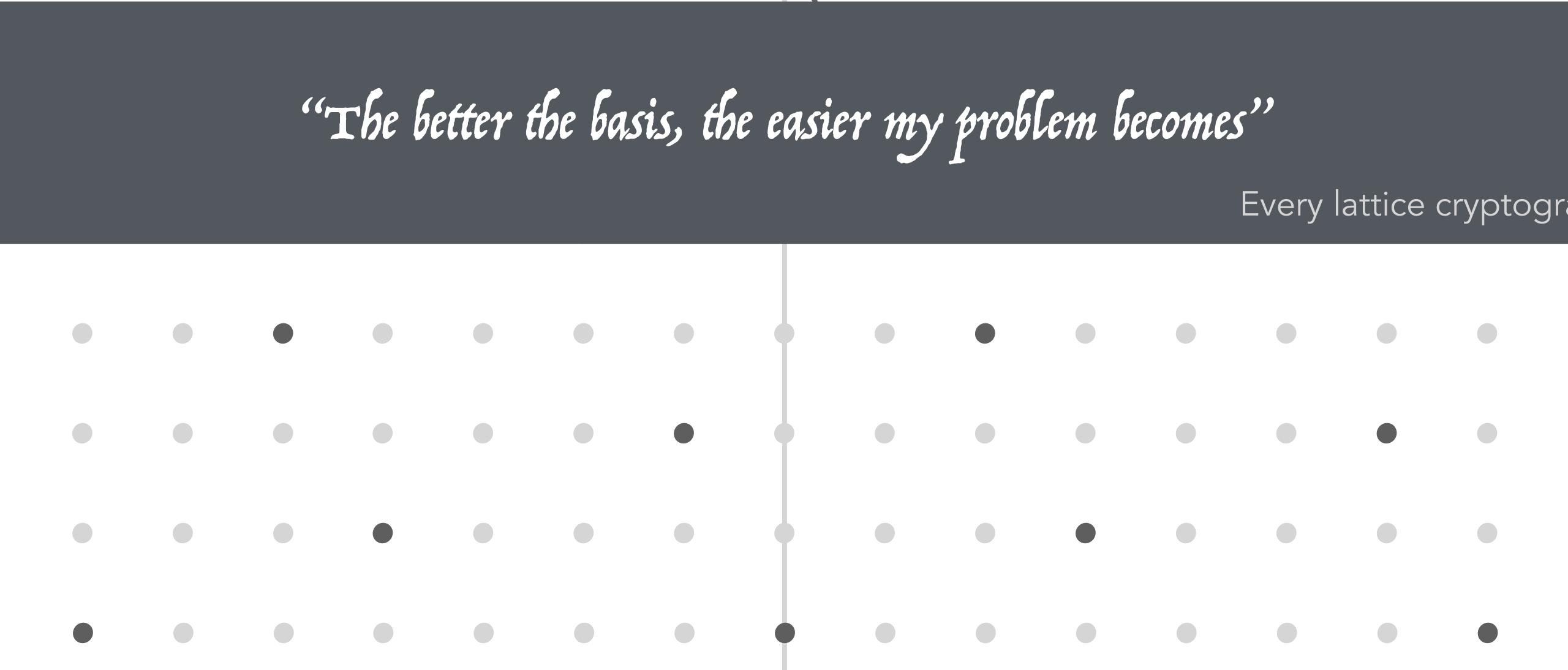
*What is a lattice ?*





*“Finding short vectors in a lattice is hard !”*

Ajtai '98

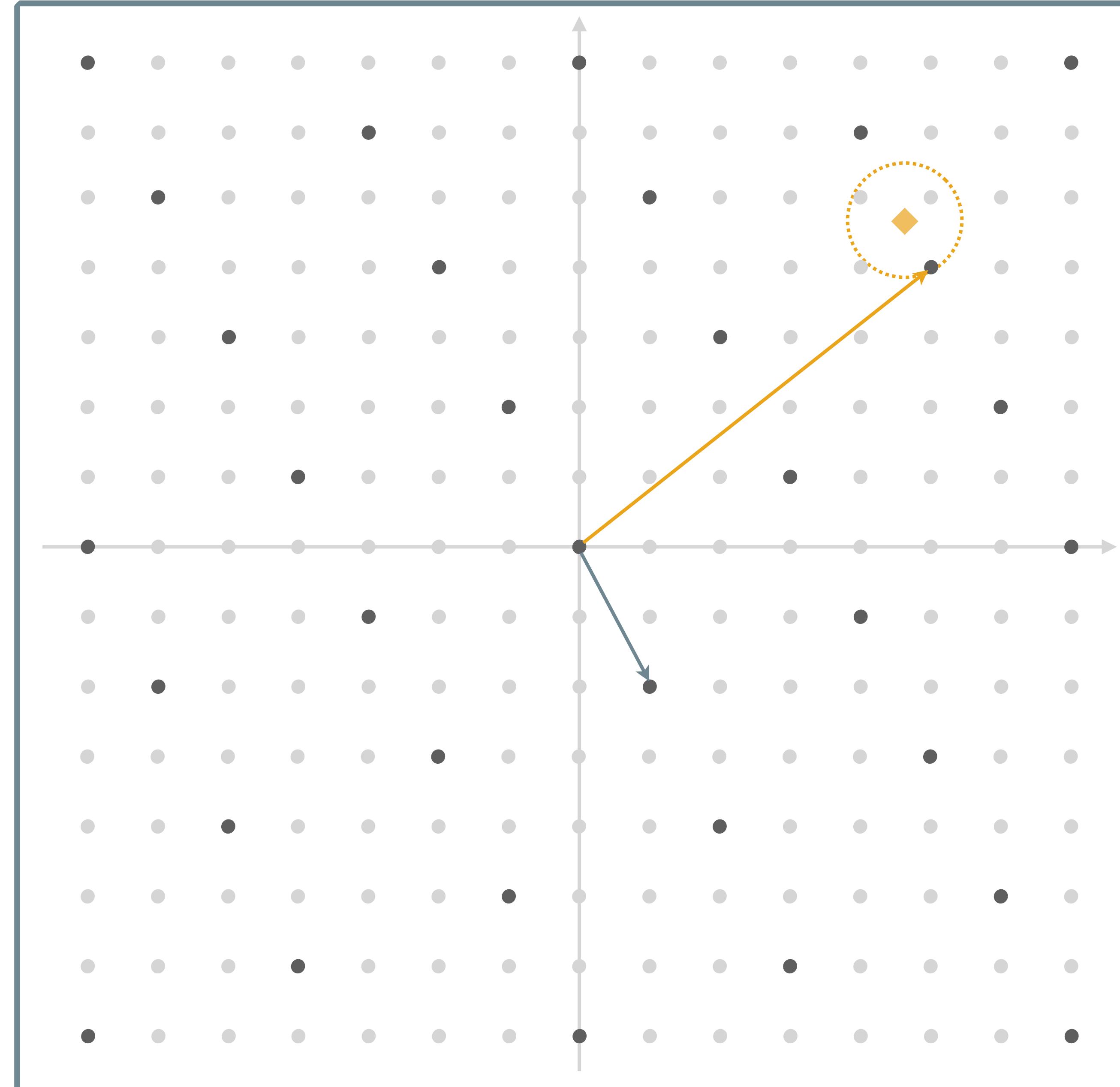


*“The better the basis, the easier my problem becomes”*

Every lattice cryptographer ever

*What are the main problems ?*

Shortest vector problem  
(SVP)



Closest vector problem  
(CVP)

# Yes but...

oof

## A more cryptographic problem

$$(1 \quad 5) \begin{pmatrix} u \\ v \end{pmatrix} = 0 \pmod{7}$$

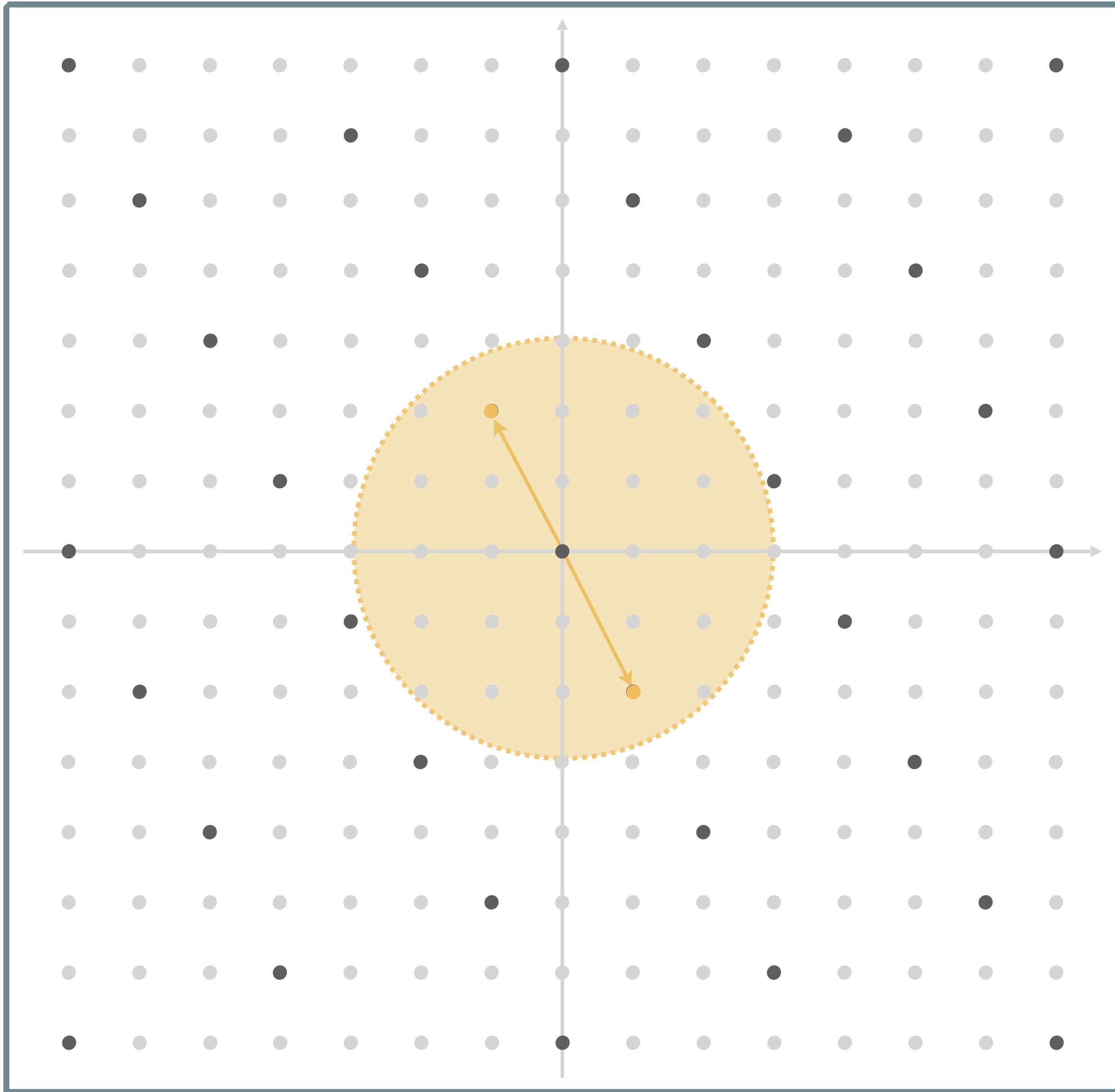
*small*  $(5 \quad -1) \quad (-5 \quad 1)$   
 $(2 \quad 1) \quad (-2 \quad -1)$

$$(10 \quad -2) \quad (-10 \quad 2)$$

The set of solutions is a lattice !

$$(24 \quad -2) \quad (-16 \quad 6)$$

$$(22 \quad -1) \quad (-22 \quad 12)$$



# Yes but...

oops ! (6)

od 11)

$$\begin{array}{c} \|x\| \leq \eta \\ A \cdot x = 0 \pmod{q} \\ b = A \cdot s + e \pmod{q} \\ \left\{ \begin{array}{c|cc} u & h \\ v & -1 \end{array} \right. = 0 \pmod{q} \end{array}$$

LWE

svp in

$$\left\{ \begin{array}{|c|} \hline x \\ \hline \end{array} \mid \begin{array}{|c|} \hline A \\ \hline \end{array} \cdot \begin{array}{|c|} \hline x \\ \hline \end{array} = \begin{array}{|c|} \hline 0 \\ \hline \end{array} \pmod{q} \right\}$$

svp in

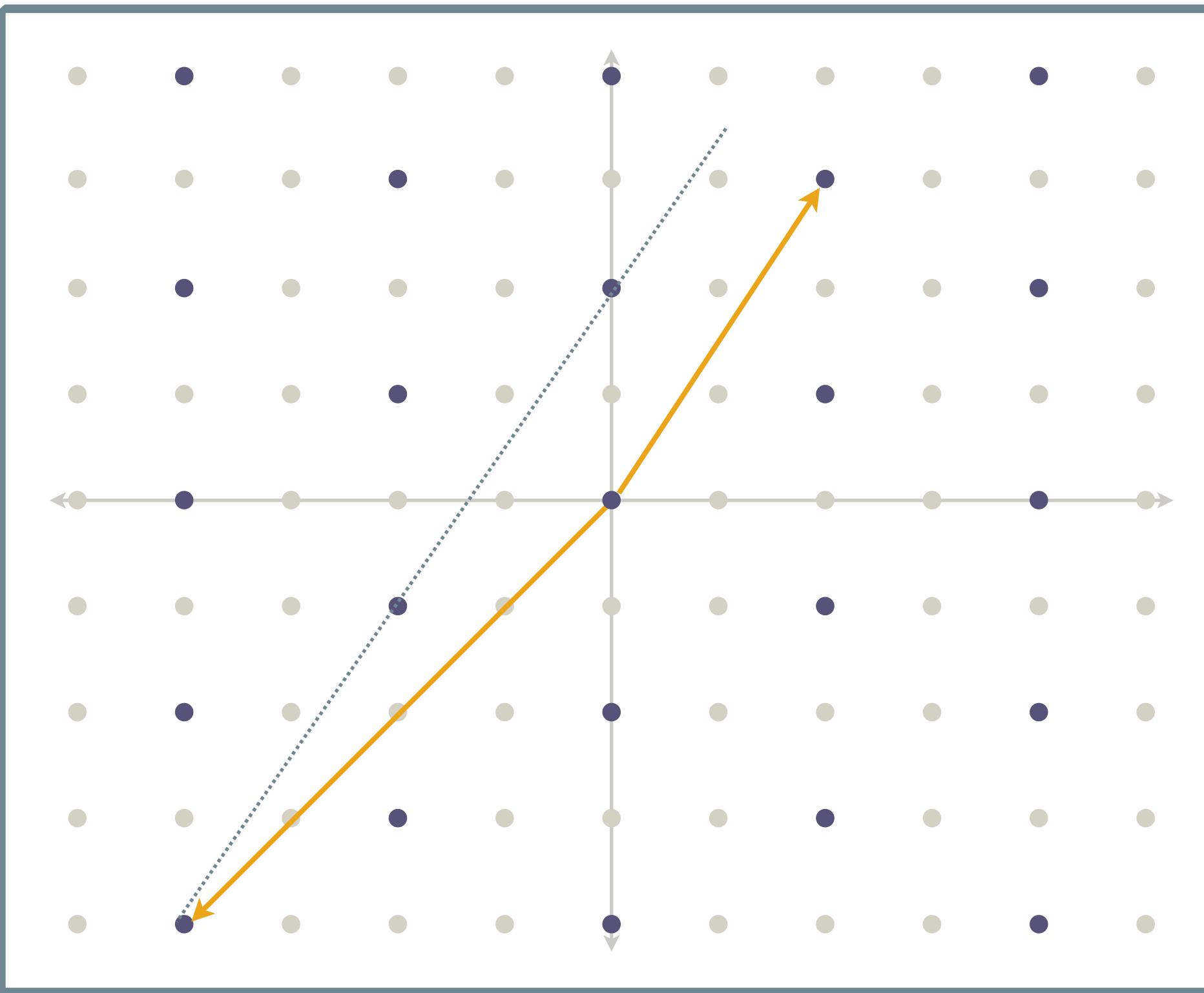
$$\left\{ \begin{array}{|c|} \hline u \\ \hline v \\ \hline \end{array} \mid \begin{array}{|c|c|} \hline u & v \\ \hline \end{array} \begin{array}{|c|} \hline h \\ \hline -1 \\ \hline \end{array} = \begin{array}{|c|} \hline 0 \\ \hline \end{array} \pmod{q} \right\}$$

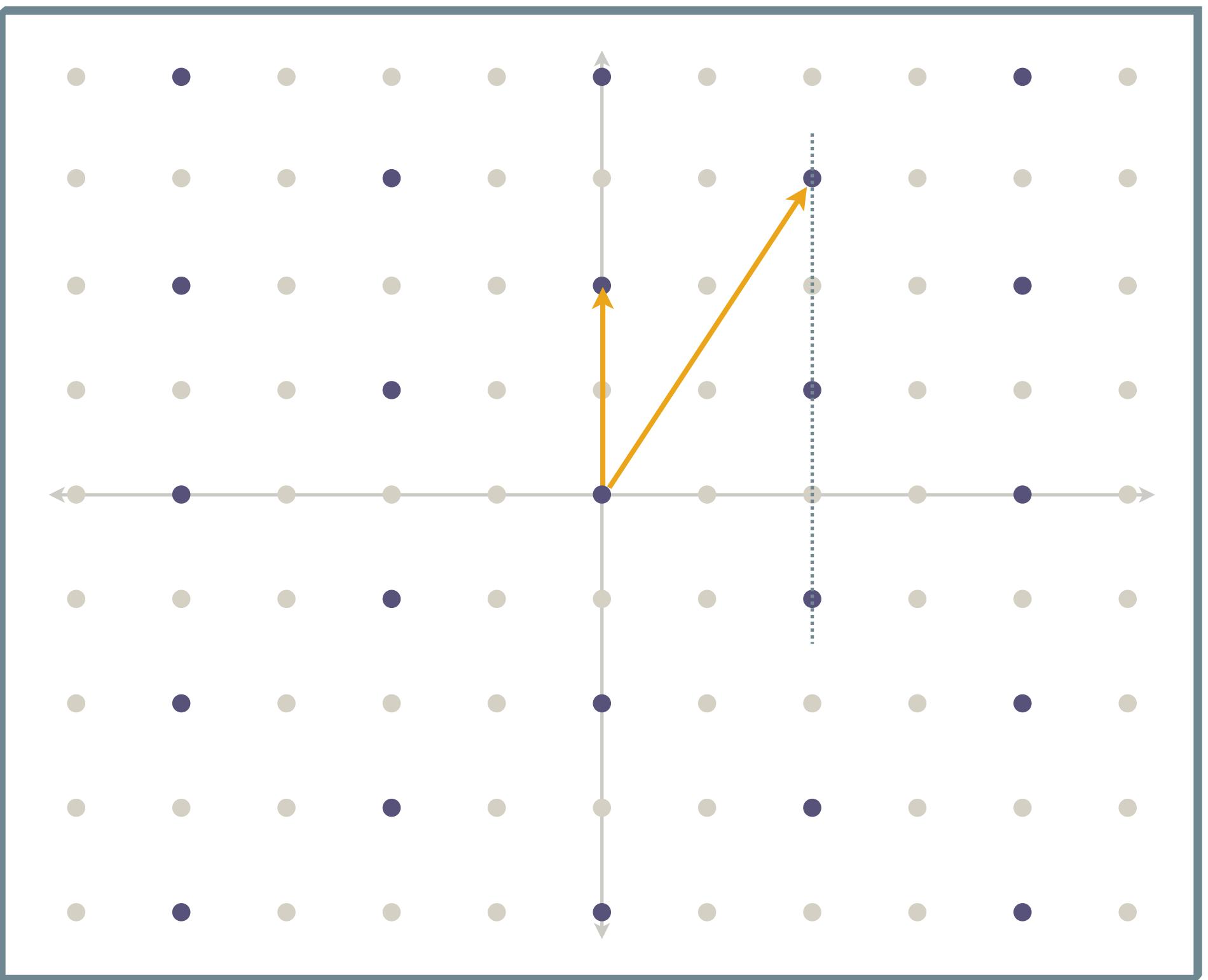
cvp in

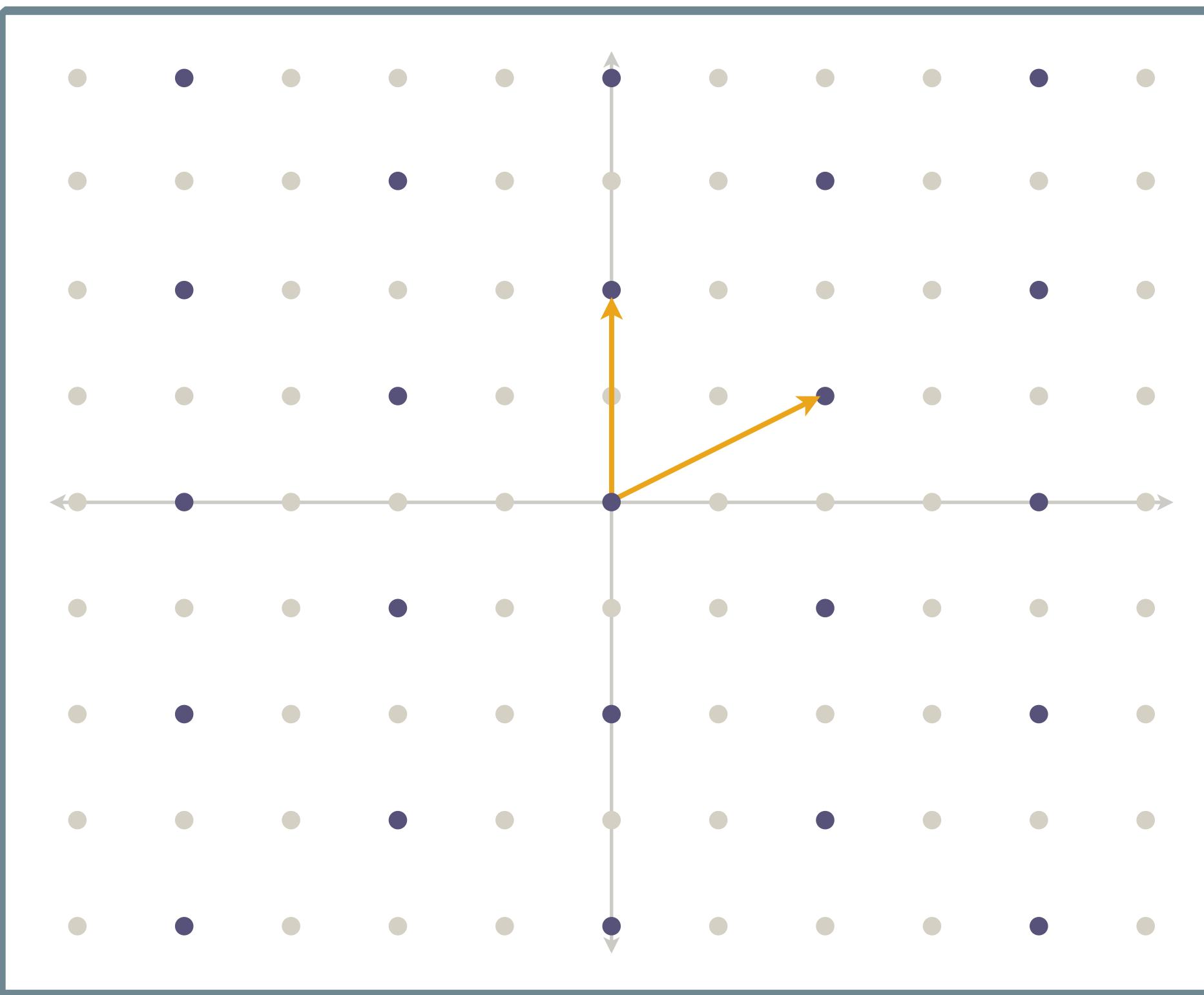
$$\left\{ \begin{array}{|c|} \hline A \\ \hline \end{array} \cdot \begin{array}{|c|} \hline u \\ \hline + \\ \hline v \\ \hline \end{array} \mid \begin{array}{|c|c|} \hline u & v \\ \hline \end{array} \in \mathbb{Z}^n \times q\mathbb{Z}^m \right\}$$

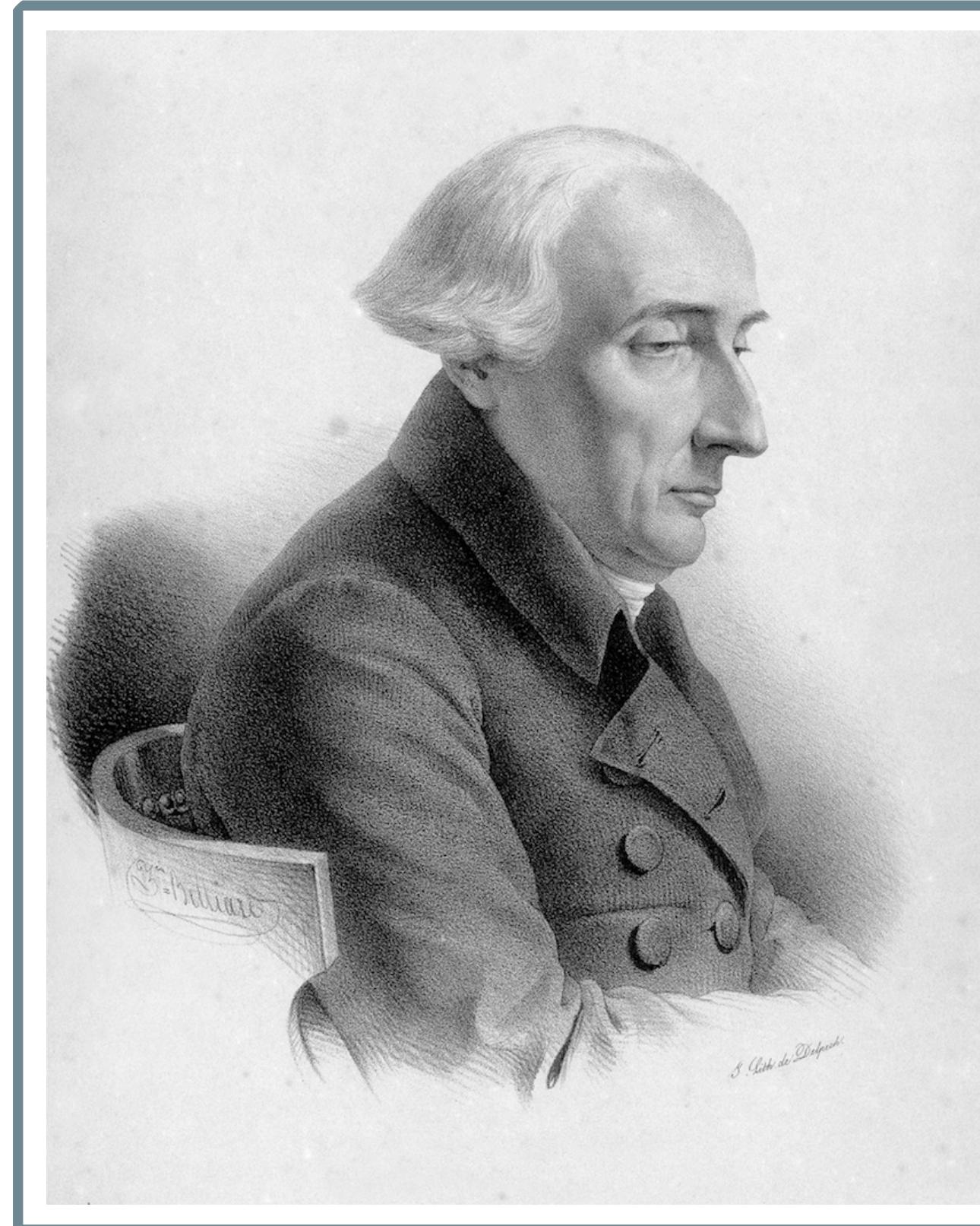
LWE

*What is lattice reduction ?*

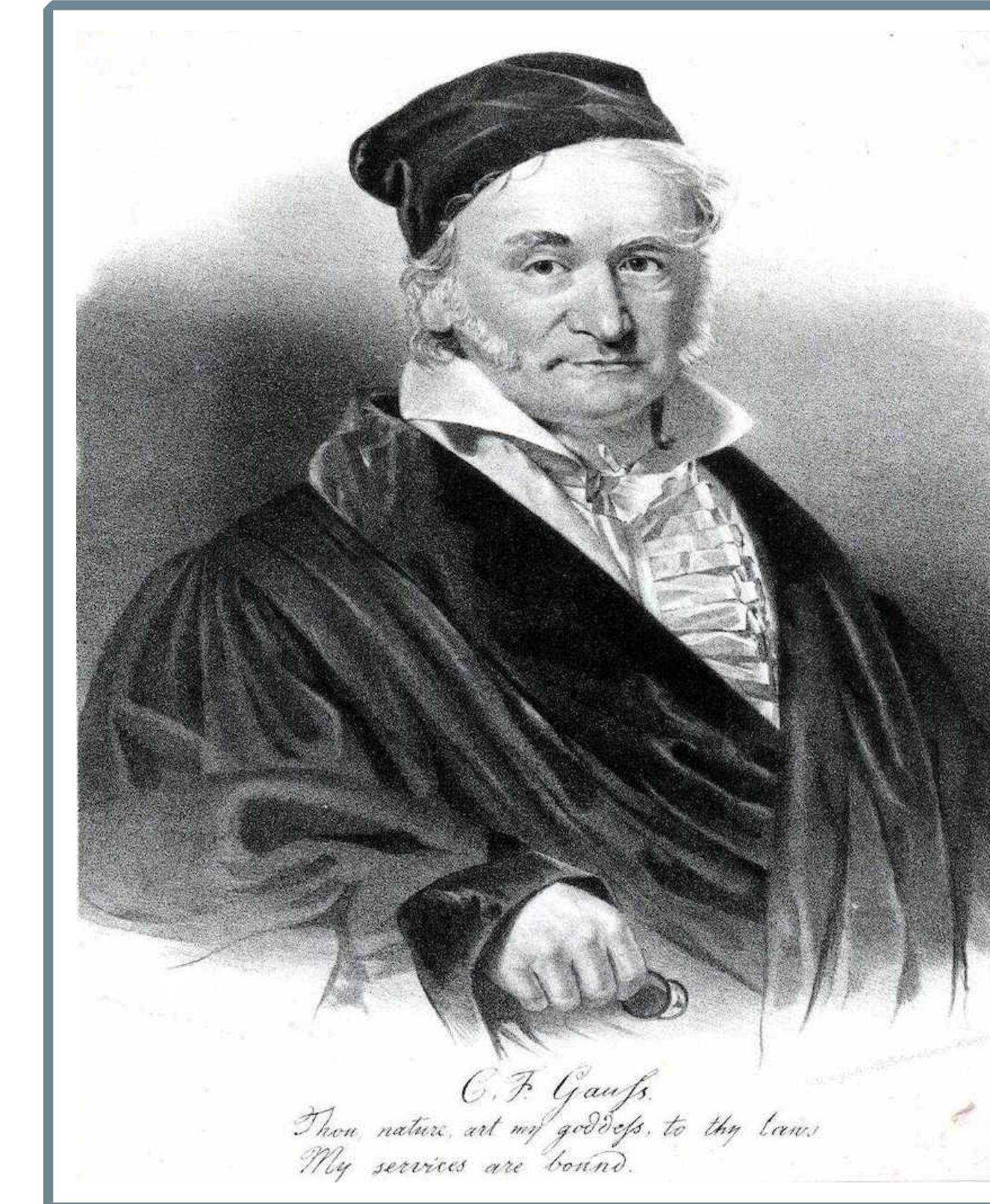






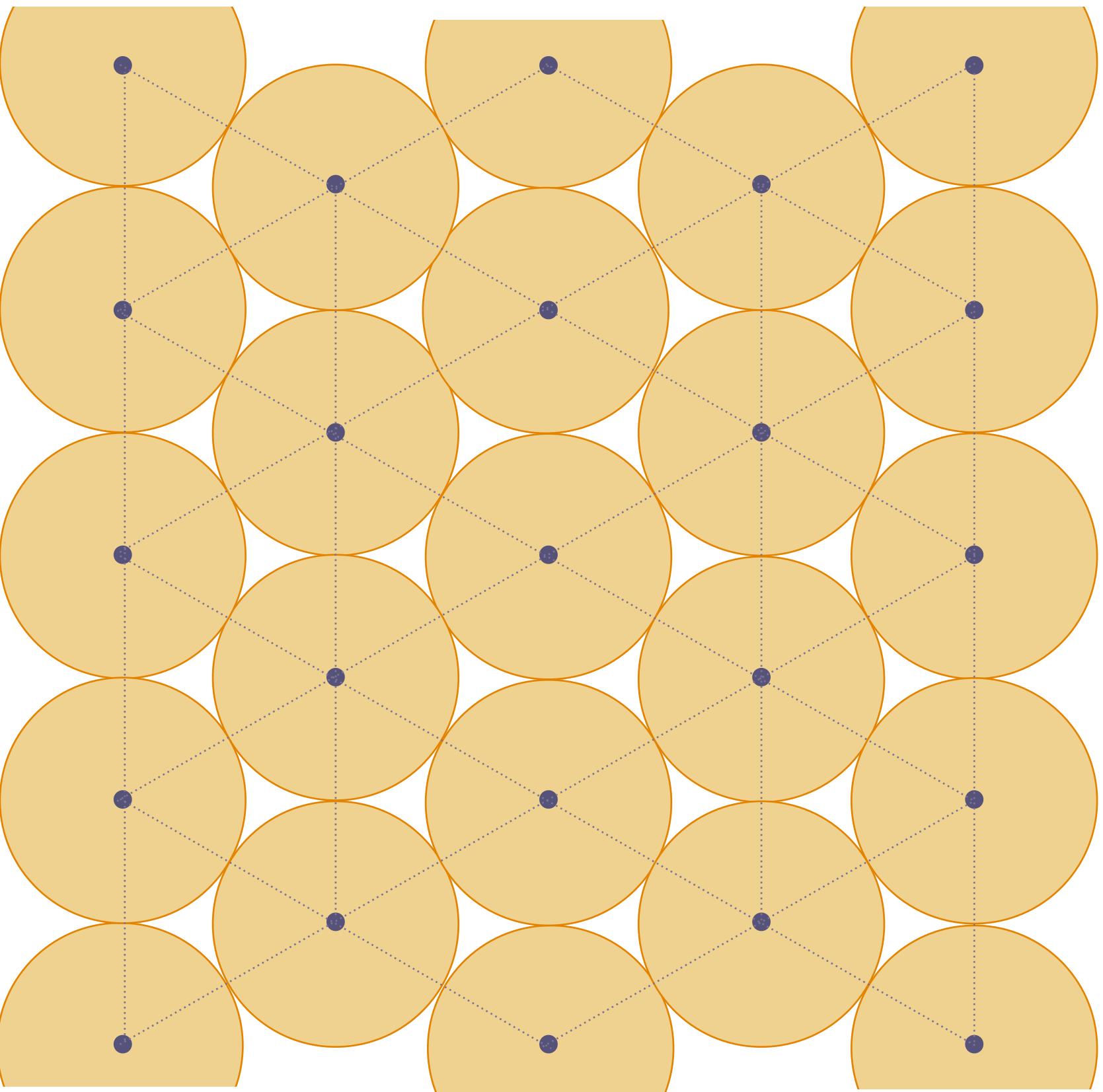


Joseph L. Lagrange



Carl F. Gauss

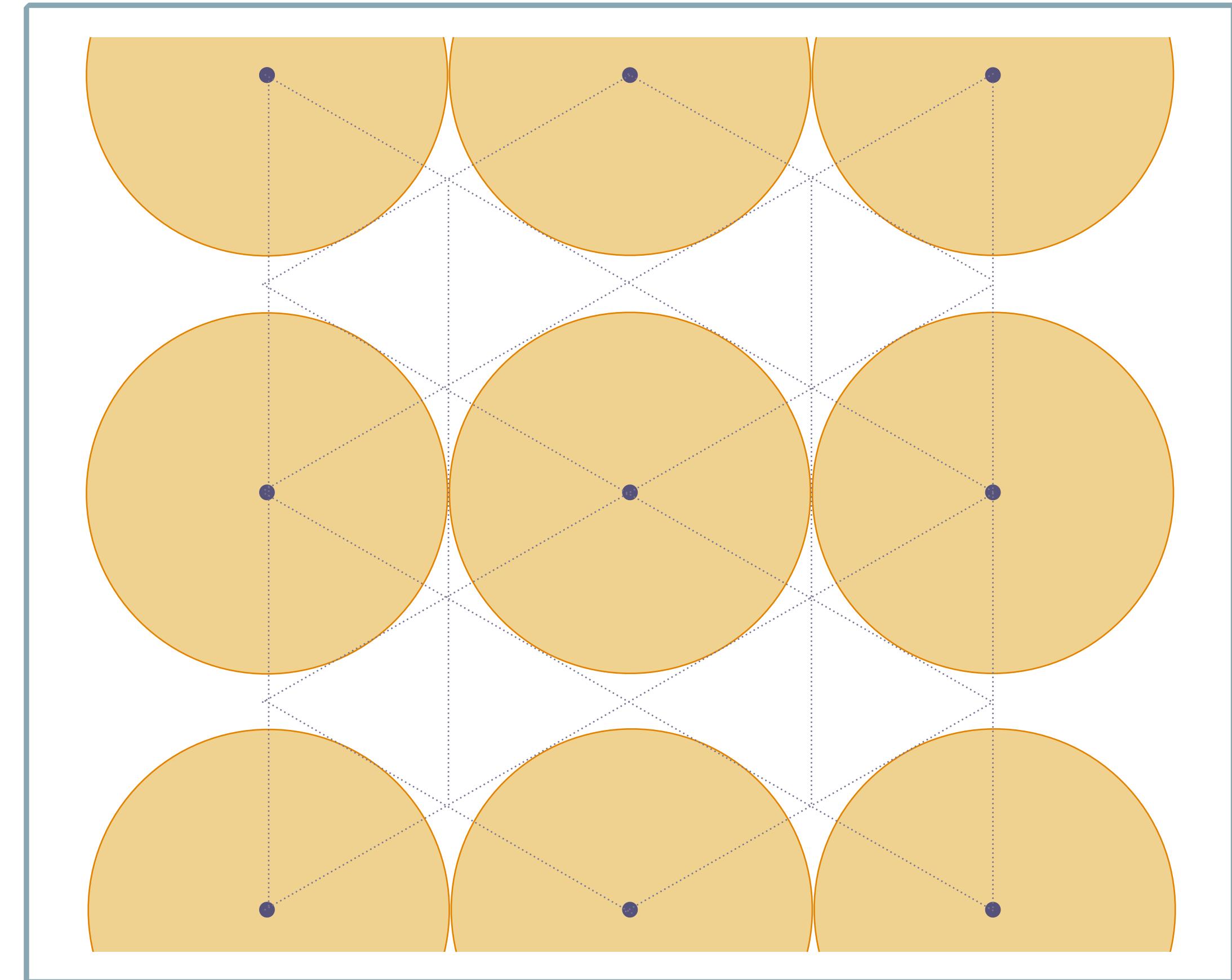
THE HEXAGONAL LATTICE IN DIMENSION 2



*Minkowski theorem*

$$\text{shortest vector} \leq \sqrt{\frac{4}{3}} \frac{1}{\text{density}}$$

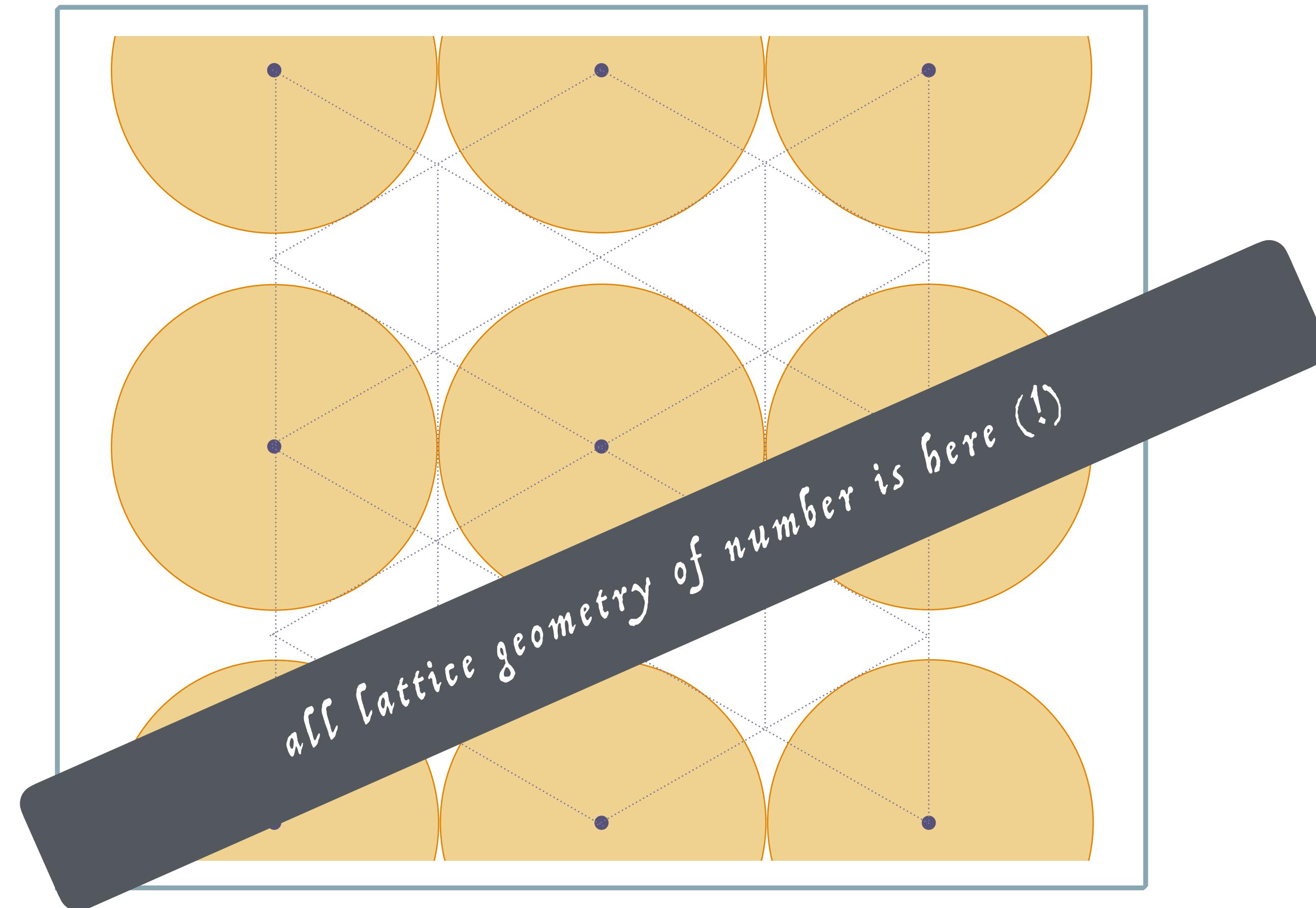
THE (SQUARE, SCALED) LATTICE IN DIMENSION 2



*Minkowski theorem*

$$\text{shortest vector} \leq \sqrt{\frac{4}{3}} \frac{1}{\text{density}}$$

THE (SQUARE, SCALED) LATTICE IN DIMENSION 2



*Minkowski theorem*

$$\text{shortest vector} \leq \sqrt{\frac{4}{3}} \frac{1}{\text{density}}$$

### Minkowski theorem

$$\text{shortest vector} \lesssim \sqrt{d} \frac{1}{\text{density}}$$

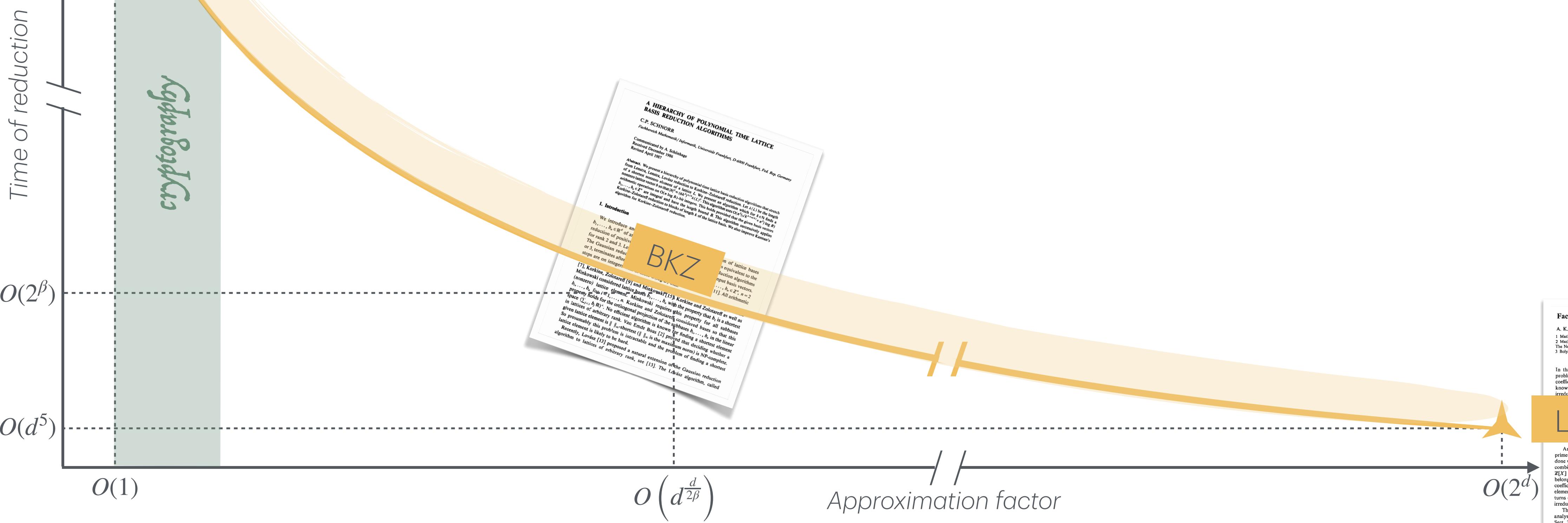
Shortest vectors exist but...

are terribly hard to find...

we can only approximate

find the shortest vector costs exponential time

exponentially bad approximation costs polynomial time



# Polynomials with Rational Coefficients

we present a polynomial-time algorithm to find a non-zero polynomial  $f \in \mathbb{Q}[X]$  in one variable such that the decomposition of  $f$  into irreducible factors in  $\mathbb{Z}[X]$  is equivalent to factoring primitive polynomials in  $\mathbb{Z}[X]$ . Here we call  $f \in \mathbb{Z}[X]$  primitive if

In practice, cf. [8]. Its run length  $|f|$  is. Here  $f \in \mathbb{Z}[X]$  is of  $f$ , and

$$\sum_i a_i X^i = \left( \sum_i a_i^2 \right)^{1/2}$$

all coefficients  $a_i$ .

<sup>i</sup> of the algorithm is as follows. First we find a  $p$ -adic irreducible factor  $h$  of  $f_i$ , to a certain  $p$ -adic precision, by using a modified version of Hensel's lemma. Next we look for the irreducible factors of  $h$ . The condition that  $h_0$  is divisible by  $h$ , the condition that  $h_0$  is divisible by a certain lattice, and the condition that  $h_0$  divides  $h_0'$  are relatively small. It follows that we may search for  $h_0$  by using a basis of a lattice, and this is done by means of a basis of  $\mathbb{Z}_p$ . This enables us to determine  $h_0$ . The algorithm for  $f$  has been found.

reduction algorithm that we employ is new, and it. It improves the algorithm given in a previous paper. At the end of Sect. 1, we briefly mention a problem related to the Diophantine approximation.

between factors of  $f$  and reduced bases of  $\mathcal{A}$ . The theory presented here extends a result which should be remarked that the latter result, while it has sufficed for our purpose.

relationship:  $J \ll \beta \leq d$ . Since the inserting area of each Pump is at most the value of dimension for free  $d4f(\beta)$  (Eq.(1)) according to entire block size  $\beta$ . To ensure the output lattice basis of each Pump is almost HKZ-reduced lattice basis, one needs  $J \leq d4f(\beta)$ . Eq.(1) shows the dimension for free value used in the implementation of G6K([3],[10]). In other words, to ensure the output lattice basis of each Pump is almost HKZ-reduced lattice basis, under the dimension for free value setting in G6K,  $J \leq 0.076\beta \ll \beta \leq d$  when  $\beta$  is bigger enough.

$$d4f(\beta) = \begin{cases} 0, & \beta < 40 \\ \lfloor \frac{\beta-40}{2} \rfloor, & 40 \leq \beta \leq 75 \\ \lfloor 11.5 + 0.075\beta \rfloor, & \beta > 75. \end{cases} \quad (1)$$

We use  $\mathbf{J}_{i,j}$  to represent all-ones matrix where every entry is equal to 1 with  $i$  rows and  $j$  columns,  $\mathbf{0}_{i,j}$  represent  $i \times j$  zero matrix, and  $\mathbf{I}_n$  represent  $n$ -dimensional identity matrix. It is easy to get that:

$$\mathbf{A}^{(1+J)} \cdot \mathbf{A}^{(1)} = \begin{pmatrix} \frac{1}{\beta} \mathbf{J}_{J,\beta} & \mathbf{0}_{J,J} & \mathbf{0}_{J,d-\beta-J} \\ \frac{\beta-J}{\beta^2} \mathbf{J}_{\beta,\beta} & \frac{1}{\beta} \mathbf{J}_{\beta,J} & \mathbf{0}_{\beta,d-\beta-J} \\ \mathbf{0}_{d-\beta-J,\beta} & \mathbf{0}_{d-\beta-J,J} \mathbf{I}_{d-\beta-J,d-\beta-J} & \mathbf{0}_{J,d-\beta-J} \end{pmatrix},$$

$$\mathbf{A}^{(1+2J)} \cdot \mathbf{A}^{(1+J)} \cdot \mathbf{A}^{(1)} = \begin{pmatrix} \frac{1}{\beta} \mathbf{J}_{J,\beta} & \mathbf{0}_{J,J} & \mathbf{0}_{J,J} \\ \frac{\beta-J}{\beta^2} \mathbf{J}_{\beta,\beta} & \frac{1}{\beta} \mathbf{J}_{\beta,J} & \mathbf{0}_{J,J} \\ \frac{(\beta-J)^2}{\beta^3} \mathbf{J}_{\beta,\beta} & \frac{\beta-J}{\beta^2} \mathbf{J}_{\beta,J} & \mathbf{0}_{J,J} \\ \mathbf{0}_{d-\beta-2J,\beta} & \mathbf{0}_{d-\beta-2J,J} \mathbf{I}_{d-\beta-2J,d-\beta-2J} & \mathbf{0}_{\beta,d-\beta-2J} \end{pmatrix}$$

$$\text{We can set } \mathbf{A}^{(1+(k-1)J)} \dots \dots \mathbf{A}^{(1+2J)} \cdot \mathbf{A}^{(1+J)} \cdot \mathbf{A}^{(1)} = \begin{pmatrix} \frac{1}{\beta} \mathbf{J}_{J,\beta} & \mathbf{0}_{J,J} & \dots & \mathbf{0}_{J,d-(k-1)J} \\ \frac{\beta-J}{\beta^2} \mathbf{J}_{\beta,\beta} & \frac{1}{\beta} \mathbf{J}_{\beta,J} & \dots & \mathbf{0}_{J,d-(k-1)J} \\ \frac{(\beta-J)^2}{\beta^3} \mathbf{J}_{\beta,\beta} & \frac{\beta-J}{\beta^2} \mathbf{J}_{\beta,J} & \dots & \mathbf{0}_{J,d-(k-1)J} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{1}{\beta} \mathbf{J}_{J,\beta} & \mathbf{0}_{J,J} & \dots & \mathbf{0}_{J,d-(k-1)J} \end{pmatrix} \cdot \mathbf{A}^{(1+(k-1)J)} \dots \dots \mathbf{A}^{(1+2J)} \cdot \mathbf{A}^{(1+J)} \cdot \mathbf{A}^{(1)} =$$

*that being said... too slow to run, too hard to predict*

```

1: 
2: ds := f + jump
3: B=LLL(B);
4: for i in {1, ..., d+2f - β} do
5:   if 1 ≤ i ≤ f+1 then
6:     if 1 ≤ i ≤ jump then
7:       f' := 1, β - f + jump · i - 1, jump · i
8:       ds := f' + jump
9:   end if
10: end for
11: B=LLL(B);
12: return B
  
```

```

13: 
14: B=LLL(B)
15: 
16: end for
17: B'=Pump(B, d - β + f + 1, β, J)
18: return B
  
```

One can obtain an (almost) HKZ reduced basis, in the Pump-down stage, which has actually already in the implementation of G6K-GPU[10]. After Pump-down stage the output projected basis an HKZ reduction. More detail about the reduction in the description of Pump in Section 4.1 of

*oops ! (ter)*

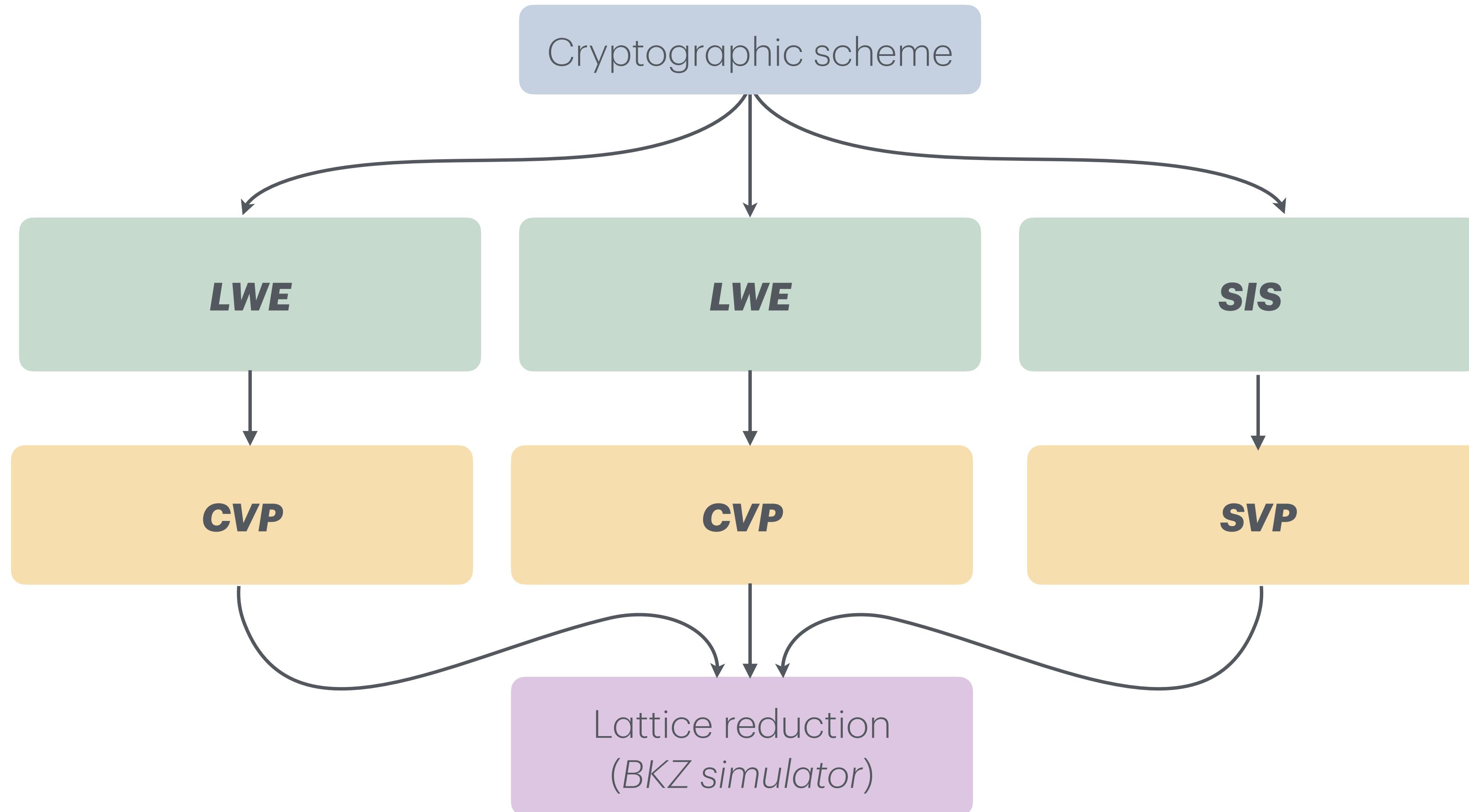
$$\mathbf{A} = \begin{pmatrix} \frac{1}{\beta} \mathbf{J}_{J,\beta} & \dots & \frac{\beta-J}{\beta^2} \mathbf{J}_{\beta,\beta} & \mathbf{0}_{J,J} & \dots & \mathbf{0}_{J,d-\beta-k,J} \\ \frac{\beta-J}{\beta^2} \mathbf{J}_{\beta,\beta} & \dots & \frac{1}{\beta} \mathbf{J}_{\beta,J} & \mathbf{0}_{J,J} & \dots & \mathbf{0}_{\beta,d-\beta-k,J} \\ \frac{(\beta-J)^2}{\beta^3} \mathbf{J}_{\beta,\beta} & \dots & \frac{\beta-J}{\beta^2} \mathbf{J}_{\beta,J} & \mathbf{0}_{J,J} & \dots & \mathbf{0}_{\beta,d-\beta-k,J} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{(\beta-J)^{k-1}}{\beta^k} \mathbf{J}_{\beta,\beta} & \dots & \frac{1}{\beta} \mathbf{J}_{\beta,J} & \mathbf{0}_{J,J} & \dots & \mathbf{0}_{J,J} \\ \frac{(\beta-J)^k}{\beta^{k+1}} \mathbf{J}_{\beta,\beta} & \dots & \frac{\beta-J}{\beta^k} \mathbf{J}_{\beta,J} & \mathbf{0}_{J,J} & \dots & \mathbf{0}_{J,J} \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ \frac{(\beta-J)^{k-1}}{\beta^k} \mathbf{J}_{\beta,\beta} & \dots & \frac{1}{\beta} \mathbf{J}_{\beta,J} & \mathbf{0}_{J,J} & \dots & \mathbf{0}_{J,J} \\ \frac{(\beta-J)^k}{\beta^{k+1}} \mathbf{J}_{\beta,\beta} & \dots & \frac{\beta-J}{\beta^k} \mathbf{J}_{\beta,J} & \mathbf{0}_{J,J} & \dots & \mathbf{0}_{J,J} \end{pmatrix} \quad (10)$$

$$\text{Finally, we have: } \mathbf{A}^{(1+kJ)} \cdot \mathbf{A}^{(1+(k-1)J)} \dots \dots \mathbf{A}^{(1+2J)} \cdot \mathbf{A}^{(1+J)} \cdot \mathbf{A}^{(1)} =$$

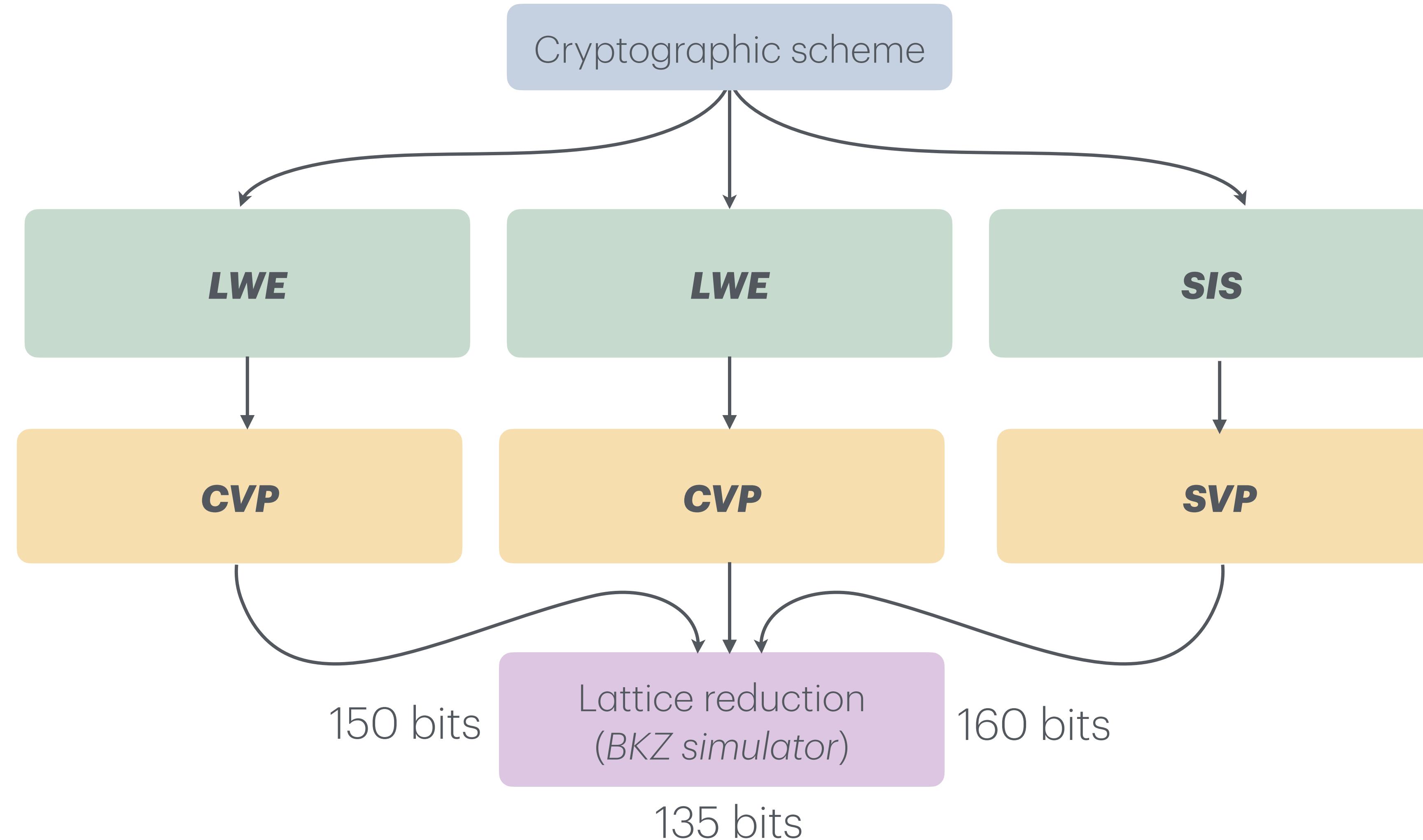


*How do we set parameters ?*

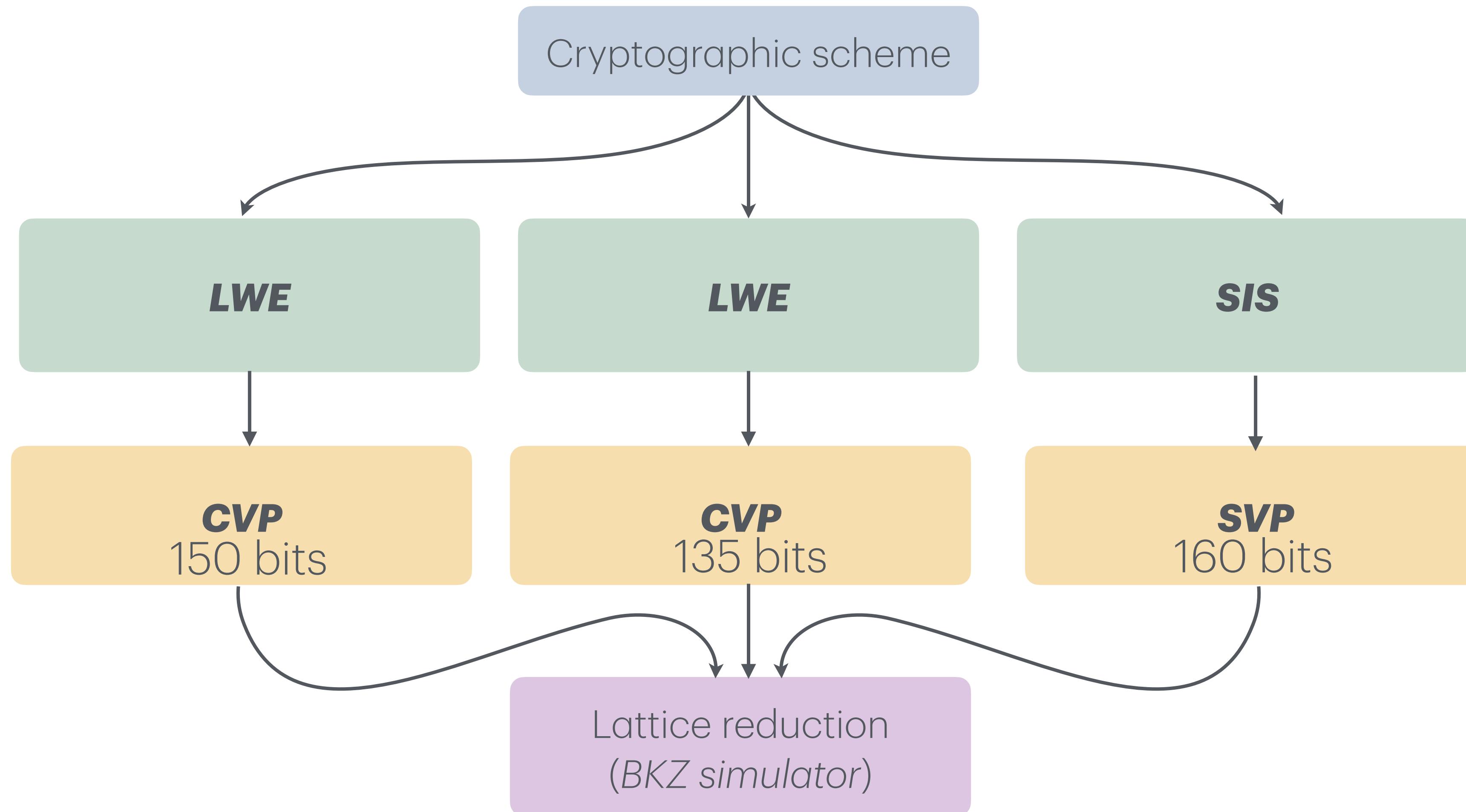
# Flowchart time!



# Flowchart time!

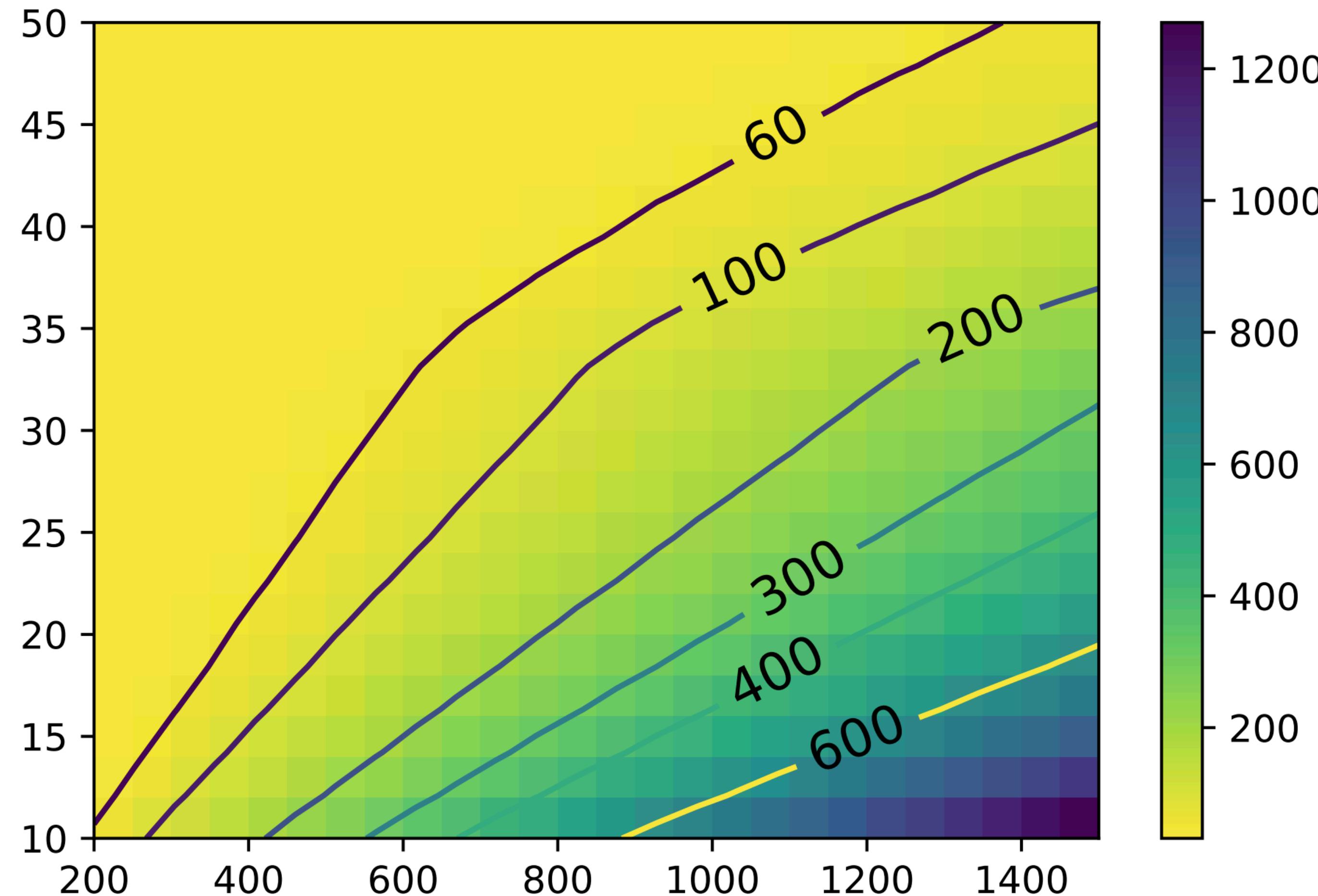


# Flowchart time!





LWE HARDNESS DENSITY/DIMENSION



Parameter	Explanation
$\mathcal{R}_q$	Polynomial ring $\mathcal{R}_q = \mathbb{Z}[X]/(q, X^n + 1)$
$(k, \ell)$	Dimension of public matrix $\mathbf{A} \in \mathcal{R}_q^{k \times \ell}$
$(\mathcal{D}_{\mathbf{t}}, \sigma_{\mathbf{t}})$	Gaussian distribution with width $\sigma_{\mathbf{t}}$ used for the verification key $\mathbf{t}$
$(\mathcal{D}_{\mathbf{w}}, \sigma_{\mathbf{w}})$	Gaussian distribution with width $\sigma_{\mathbf{w}}$ used for the commitment $\mathbf{w}$
$\nu_{\mathbf{t}}$	Amount of bit dropping performed on verification key
$\nu_{\mathbf{w}}$	Amount of bit dropping performed on (aggregated) commitment
$(q_{\nu_{\mathbf{t}}}, q_{\nu_{\mathbf{w}}})$	Rounded moduli satisfying $(q_{\nu_{\mathbf{t}}}, q_{\nu_{\mathbf{w}}}) := (\lfloor q/2^{\nu_{\mathbf{t}}} \rfloor, \lfloor q/2^{\nu_{\mathbf{w}}} \rfloor) = (\lfloor q/2^{\nu_{\mathbf{t}}} \rfloor, \lfloor q/2^{\nu_{\mathbf{w}}} \rfloor)$
$\mathbb{T} \subset \mathcal{R}_q$	Set of signed monomials (see Section 3)
$\text{rep}$	An integer s.t. $ \mathbb{T} ^{\text{rep}-1} \geq 2^\lambda$
$(\mathcal{C} \subset \mathcal{R}_q, W)$	Challenge set $\{c \in \mathcal{R}_q \mid \ c\ _\infty = 1 \wedge \ c\ _1 = W\}$ s.t. $ \mathcal{C}  \geq 2^\lambda$
$B$	Two-norm bound on the signature

PARAMETERS OF 2 ROUND THRACCOON , CRYPTO 2024

*oops ! (quad)*

	$q$	$u_t$	$u_w$	$d \cdot \text{rep}$	$v_t$	$v_w$	$n$	$\ell$	$k$	$\omega$
MLWE	↔	↔	=	↔	=	=	↔	↔	=	=
SelfTargetMSIS	↔	=	↔	↔	↔	↔	↔	↔	↔	↔
$\epsilon_{\text{TAIL}}$	=	↔	↔	↔	=	=	↔	↔	↔	↔
$R_\alpha^{\epsilon_{\text{TAIL}}}$	=	↔	↔	=	=	=	↔	↔	↔	↔
Size of $\text{vk}$	↔	↔	=	=	↔	=	↔	↔	↔	=
Size of $\text{sig}$	↔	↔	↔	↔	=	↔	↔	↔	↔	=

PARAMETERS OF THRACCOON , EUCROCRYPT 2024

size

v.

security

	$q$	$u_t$	$u_w$	$d \cdot \text{rep}$	$\nu_t$	$\nu_w$	$n$	$\ell$	$k$	$\omega$
MLWE										
SelfTargetMSIS										
$\epsilon_{\text{TAIL}}$										
$R_\alpha^{\epsilon_{\text{TAIL}}}$										
Size of $\text{vk}$										
Size of $\text{sig}$										

PARAMETERS OF THRACCOON , EUCROCRYPT 2024

# Pipeline time!

