Lattice Cryptography:

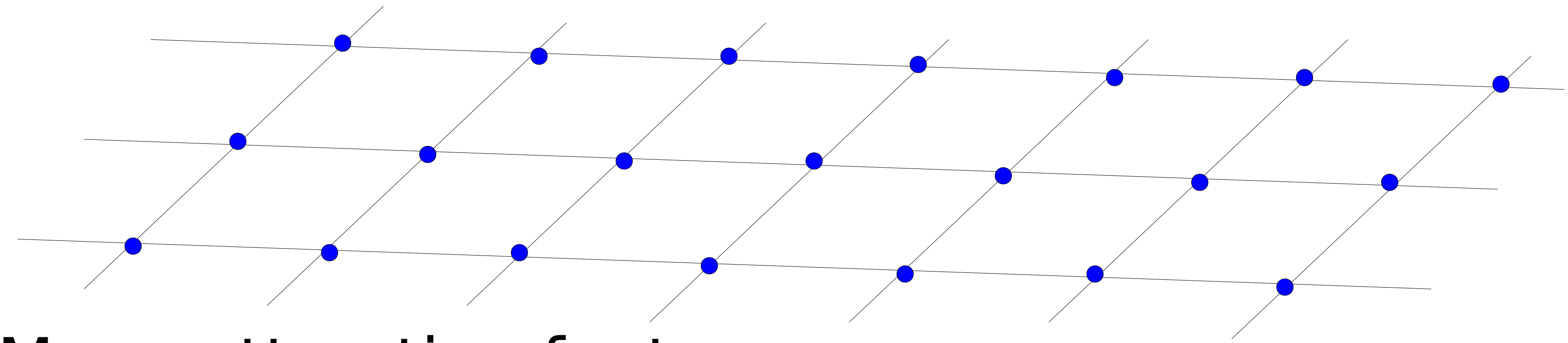
from Linear Functions
to Fully Homomorphic Encryption

Daniele Micciancio
(UC San Diego)

November 2018

Lattice cryptography

- Lattices: regular sets of vectors in n-dim space

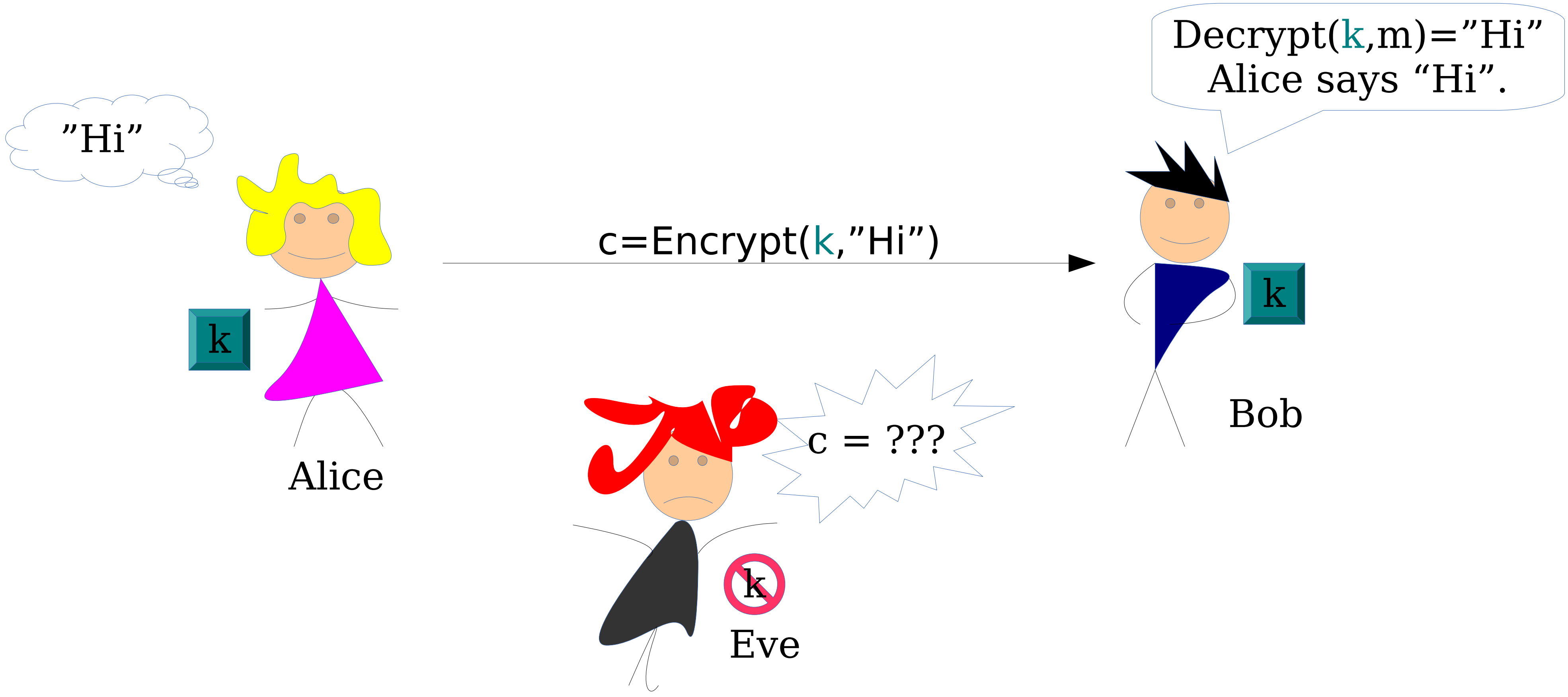


- Many attractive features:
  - Post-Quantum secure candidate
  - Simple, fast and easy to parallelize
  - Versatile (FHE and much more)

$$\begin{bmatrix}4\\1\\6\\2\\3\end{bmatrix} + \begin{bmatrix}8\\1\\7\\3\\3\end{bmatrix} = \begin{bmatrix}12\\2\\13\\5\\6\end{bmatrix}$$

Encryption

- Secure communication over insecure channel

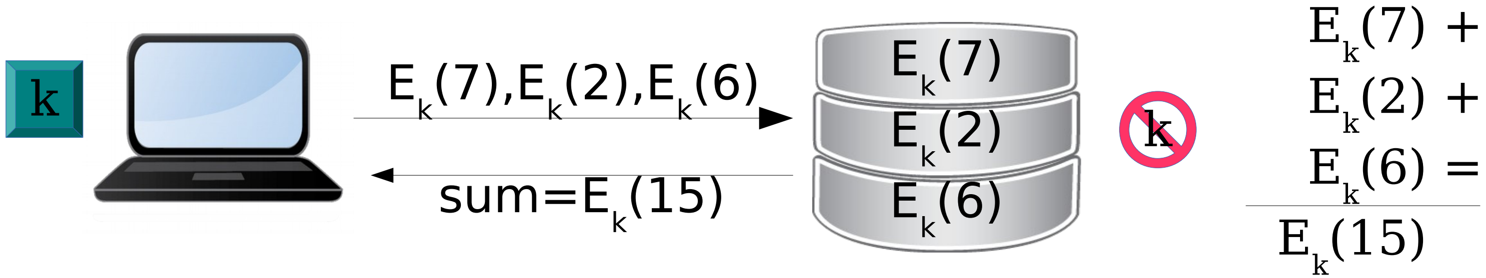


Homomorphic encryption

- Encryption function such that

  $E_k(a) + E_k(b) = E_k(a+b)$

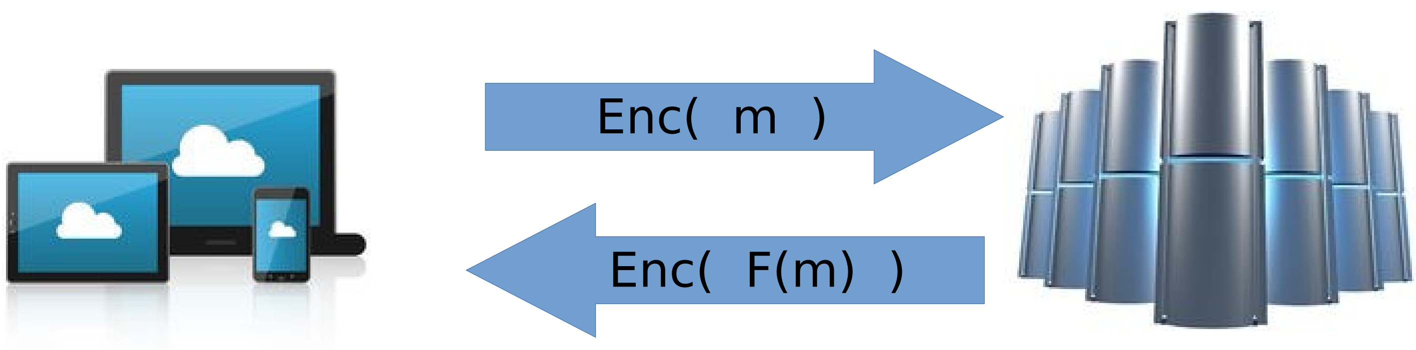
- (+) can be computed without knowing k!



Lattice Cryptography: from simple encryption to FHE

- Encryption: used to protect data at rest or in transit

- Fully Homomorphic Encryption: supports arbitrary computations on encrypted data



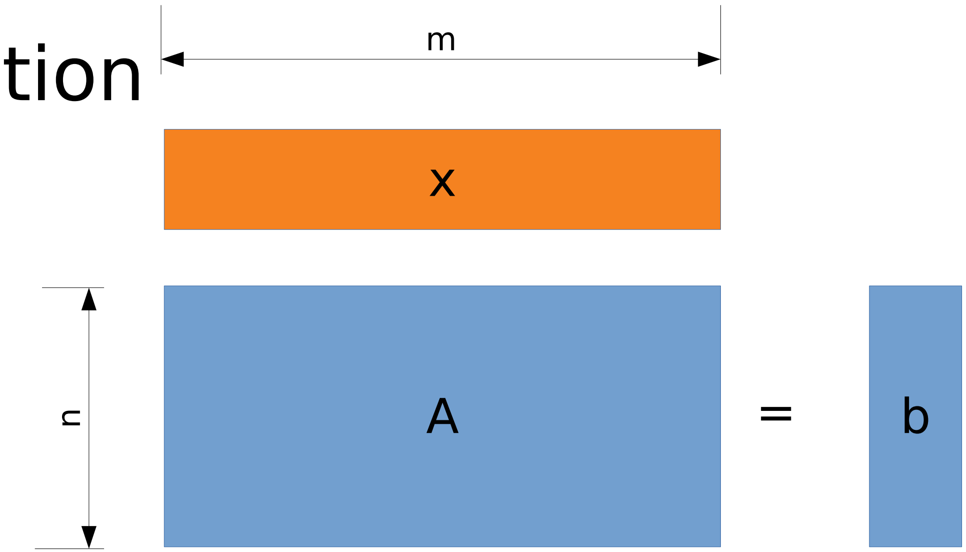
Talk Outline

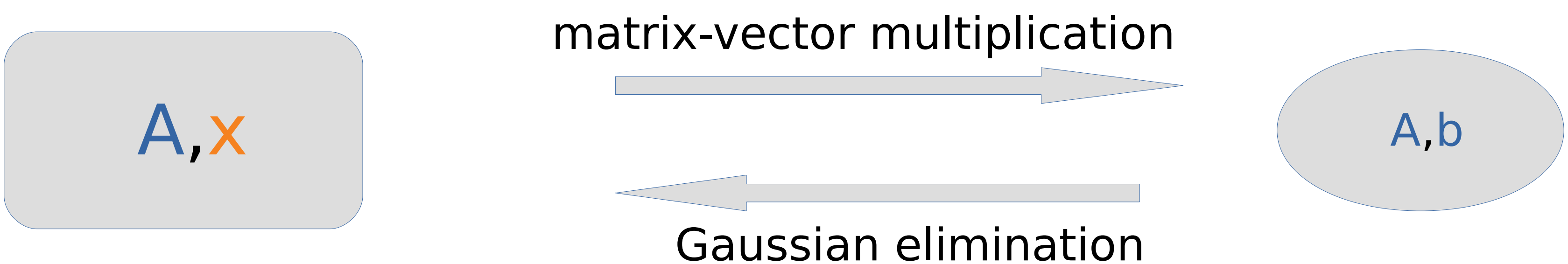
- Linear Functions: $x \to Ax$
- One-Way (hash) Functions
- Injective One-Way Functions
- Symmetric Encryption
- Public Key Encryption
- Linearly Homomorphic Encryption
- Fully Homomorphic Encryption!

Linear functions

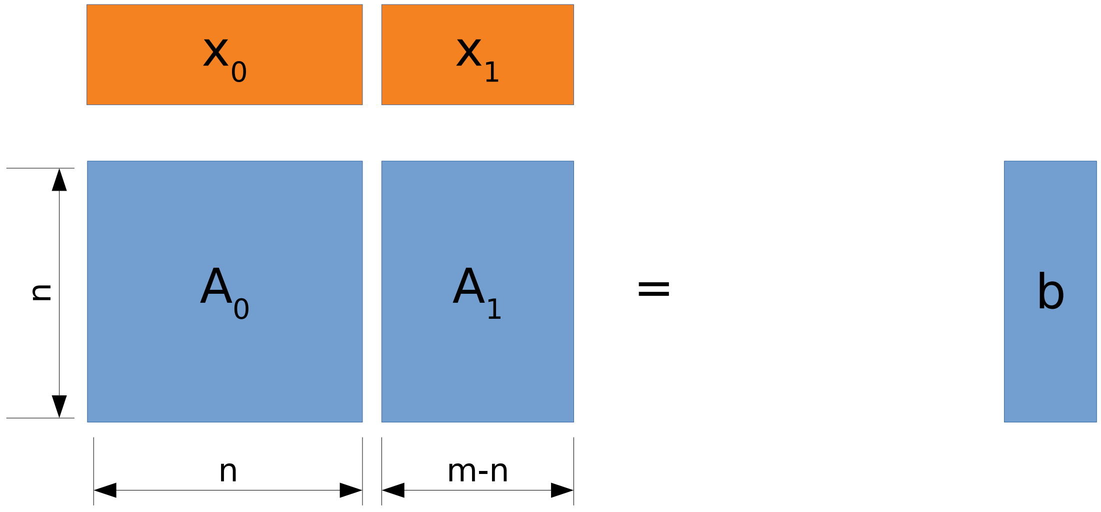
Matrix-Vector multiplication

- $A \in Z_q^{n \times m}$, $x \in Z_q^m$, $b \in Z_q^n$
- $f_A(x) = Ax$
- $f_A(x+y) = f_A(x)+f_A(y)$
- Easy to compute and invert



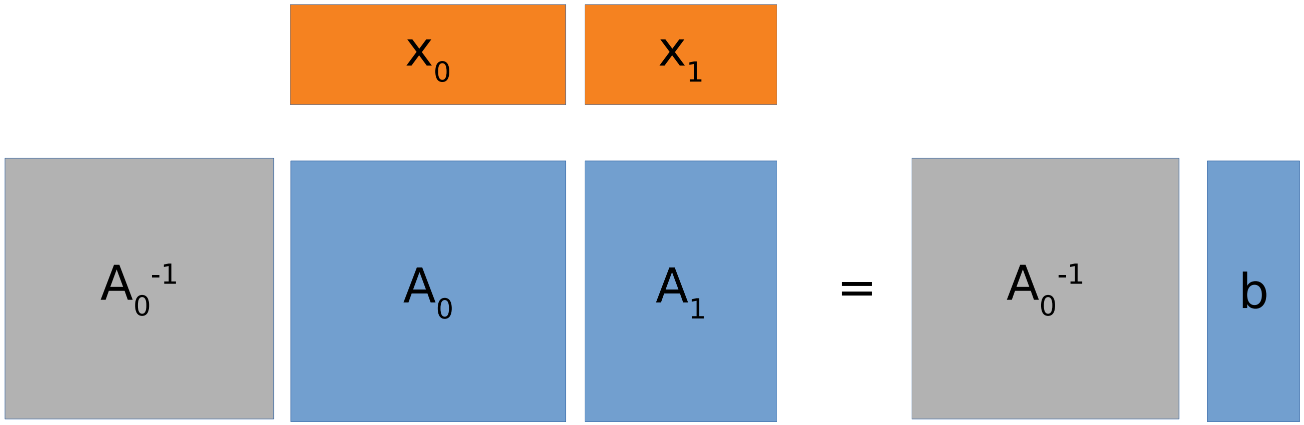


Hermite Normal Form



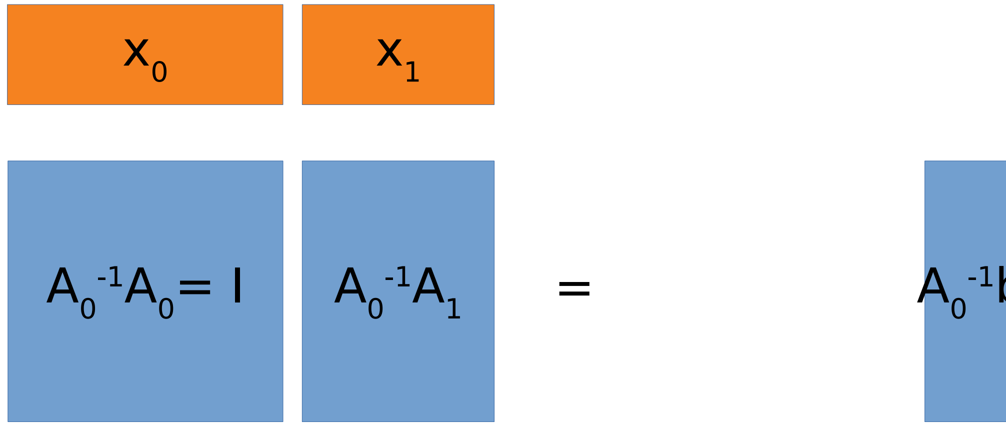
- $[A_0, A_1]\ x = b$

Hermite Normal Form



- $[A_0, A_1]\ x = b$
- $A_0^{-1}[A_0, A_1]\ x = A_0^{-1}b$

Hermite Normal Form

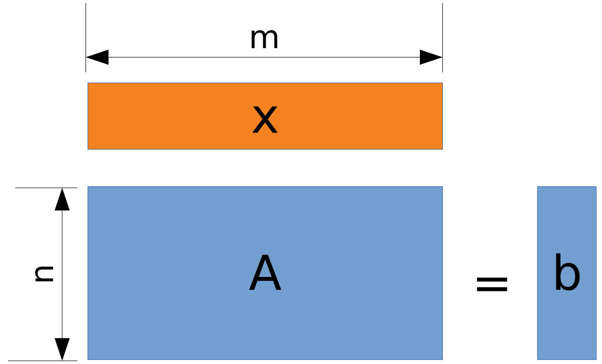


- $[A_0, A_1]\ x = b$
- $A_0^{-1}[A_0, A_1]\ x = A_0^{-1}b$
- $[\ I\ ,\ (A_0^{-1}A_1)]x = (A_0^{-1}b)$

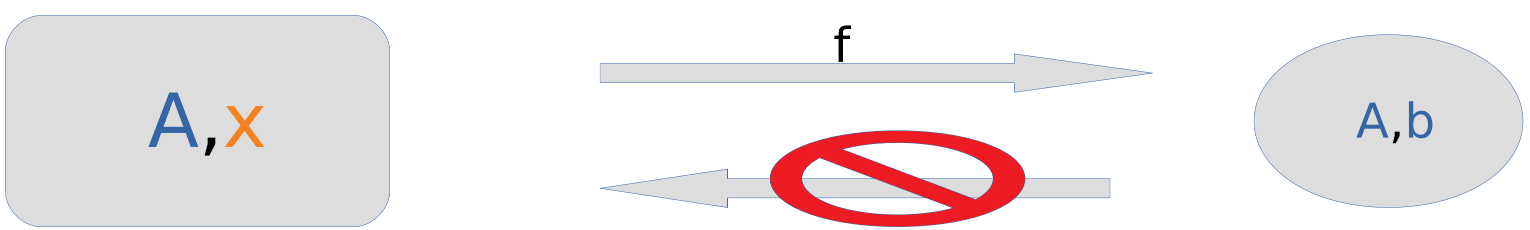
$$f_A(x_0, x_1) = x_0 + A'x_1$$

$$x_0 = b' - A'x_1$$

Short Integer Solution (SIS)



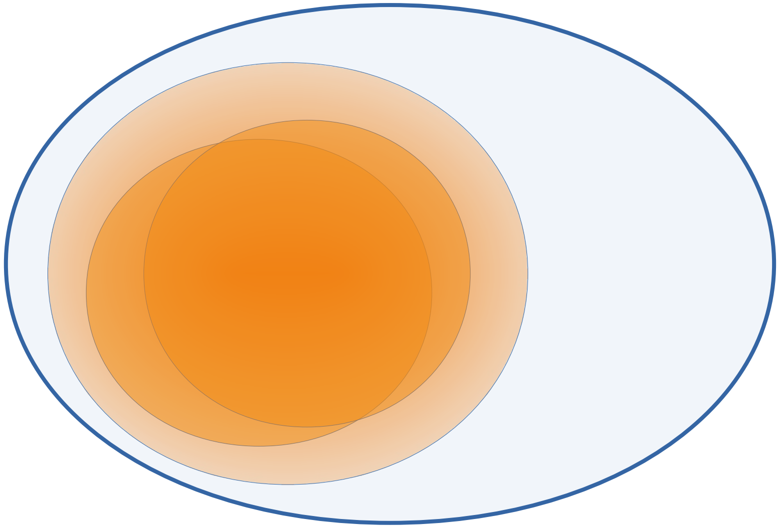
- Ajtai’s One-Way Function:
  - $f_A(x) = Ax \pmod q$
  - $A \in Z_q^{n \times m}$, $x \in$ **{1..β}**^{m}, $b \in Z_q^n$
- Short Integer Solution Problem:
  - Given [A,b] find a **<u>small</u>** x such that Ax=b



Ajtai’s SIS

- Linear function restricted to short input x

  (e.g., $\{0,1\}^m$ or $\{-3,\ldots,+3\}^m$)

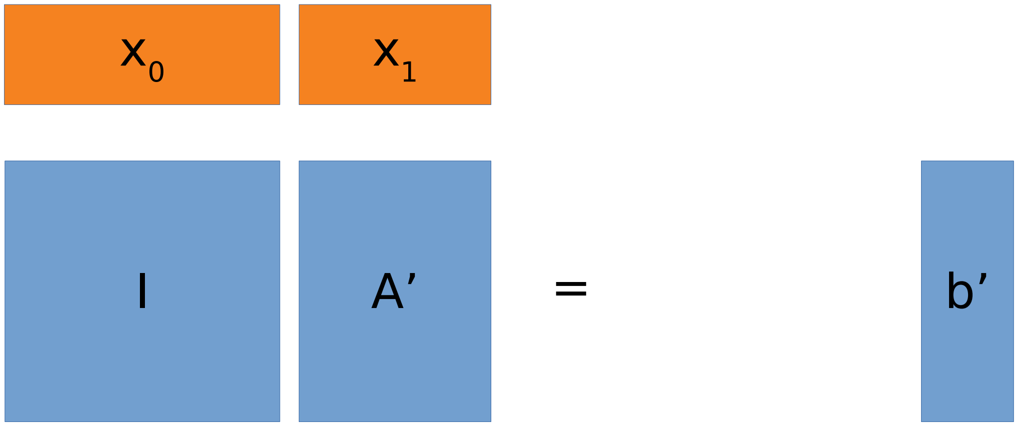


- $\{0,1\}^m$ not closed under (+)
  - Non-linear restriction
  - breaks Gaussian Elimination
  - makes function hard to invert
- $\{0,1\}^m$ approximately closed under (+) and (-)
  - $\{0,1\}^m \pm \{0,1\}^m \subset \{-2,\ldots,+2\}^m$
  - Limited homomoprhic property: still very useful

One-way Hash Functions

- SIS function f_A: $x \rightarrow b$ where $x \in \{1..\beta\}^m$, $b \in Z_q^n$
- [Ajtai 1998] inverting f_A is as hard as worst case lattice problems when
  - $m(\log \beta) > n(\log q)$
  - $|x| > |b|$
- Function f_A: compresses the input
  - surjective (w.h.p.)
  - not injective
- Applications: hashing, digital signatures

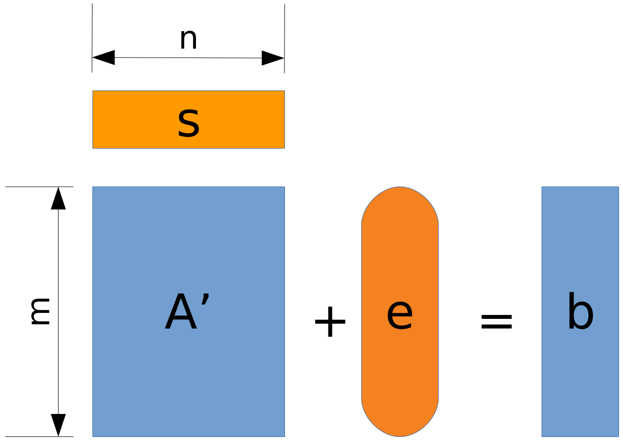
Hermite Normal Form



- $[A_0, A_1]\, x = b$
- $A_0^{-1}[A_0, A_1]\, x = A_0^{-1} b$
- $[\, I\, ,\ (A_0^{-1} A_1)]\, x = (A_0^{-1} b)$

$$f_A(x_0, x_1) = x_0 + A' x_1$$

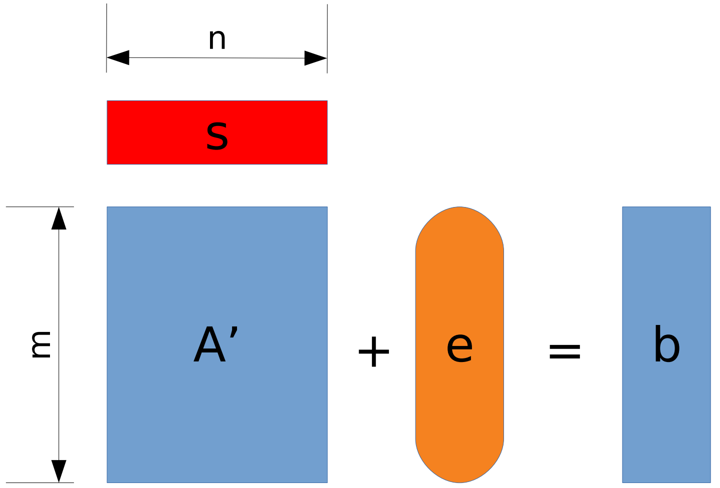
Learning With Errors (LWE)



- HNF variant of f_A:
  - $f_{[I,A']}(x_0,x_1)=x_0 + A'x_1$
  - $f_{[I,A']}(e, s)= A's + e$
- Regev 2005:
  - f_A is one-way, assuming quantum hardness of lattice problems
  - $\sqrt{n} < \beta \ll q = poly(n)$, $m=poly(n) > n$
  - $|x| = |(s,e)| \approx (n+m)(\log \beta) \approx m(\log \beta)$
  - $|b| = m(\log q) \gg |x|$
  - **injective** one-way function
  - applications: private key encryption and much more

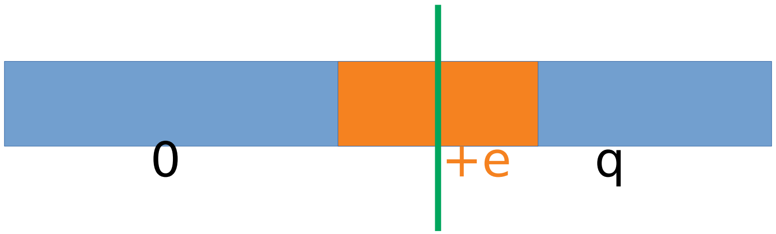
Encrypting with LWE

- Idea: Use [A,b=As+e] as a one-time pad
- Private key encryption scheme:
  - secret key: $s \in Z_q^n$,
  - message: $m \in Z^m$
  - encryption randomness: [A,e]
  - E(s, m; [A,e]) = [A,As+e+m]
- [Blum,Furst,Kearns,Lipton 1993]
  - Learning Parity with Noise (LPN): q=2
  - If f_A is one-way, then b=As+e is pseudo-random
- Regev LWE: q → poly(n)

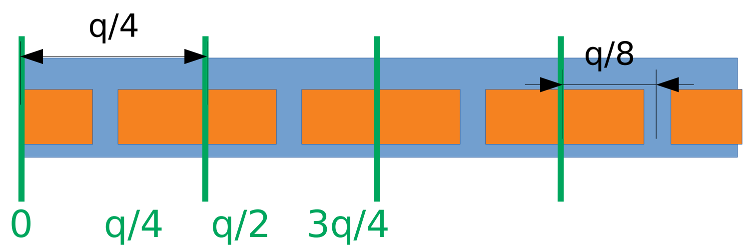


Noisy Decryption

- E(s,m;[A,e]) = [A,b] where b = As+e+m
- Decryption:
  - D(s,[A,b]) = b - As = m+e mod q



  - Low order bits of m are corrupted by e
- Fix: scale m, and round:



Still, a linear function!

- $[A_1, A_1 s + e_1 + m_1] + [A_2, A_2 s + e_2 + m_2]$

  $= [(A_1 + A_2), (A_1 + A_2)s + (e_1 + e_2) + (m_1 + m_2)]$

$E(m;\beta)$: encryption of m with error $|e| < \beta$

- $E(m_1;\beta_1) + E(m_2;\beta_2) \subset E(m_1 + m_2;\beta_1 + \beta_2)$

Decryption is also linear

- $D_s(A,b) = b - As = m+e$
- Linear in the ciphertext (A,b)
- Linear in the secret key $s' = (-s,1)$
  - $D_{s'}(A,b) = [A,b]s' = m+e$
  - $D_{cs'}(A,b) = [A,b](cs') = cm+ce$
- Simplifying assumption: $A = a \in Z$
  - This is just for notational simplicity

Operations on Ciphertexts

- Add: $E(m_1;\beta_1)+E(m_2;\beta_2) \subset E(m_1+m_2;\beta_1+\beta_2)$
- Neg: $-E(m;\beta) = E(-m;\beta)$
- Mul: $c*E(m;\beta) = E(c*m;\ c*\beta)$
- Const: $[O,m] \in E(m;0)$

Weak linear homomorphic properties

  - can perform a limited number of additions and multiplications by small constants
  - decryption is linear in the secret key $s'=(-s,1)$

Public Key Encryption

- Public Key:

  $[a_1,b_1] = E_s(0), \ldots, [a_n,b_n]=E_s(0)$

- Encrypt(m): $(\Sigma_i\ r_i * [a_i,b_i]) + (0,m)$
  - $E_s(0)+\ldots+E_s(0)+E_s(m;0) = E_s(m)$
- Decrypt normally using secret key
- [Regev'05] LWE Public Key Encryption
- [Rothblum'11]: any weakly linear homomorphic encryption implies public key encryption

Multiplication by any constant

- $E'[m] = E[m], E[2m], E[4m], \ldots, E[2^{\log(q)} m]$
- Multiplication by $c \in Z_q$:
  - Write $c = \Sigma_i\, c_i\, 2^i$, where $c_i \in \{0,1\}$
  - Compute $\Sigma_i\, c_i\, E[2^i\, m] = E[\Sigma_i\, c_i\, 2^i m] = E[cm]$
- $cE'[m] = E[cm]$
- We can also compute $E'[cm]$:

  $c{*}E'[m], (2c){*}E'[m], \ldots, (2^{\log q} c){*}E'[m]$

Homomorphic Decryption

- Idea:
  - Encryption $E(m) = (a, as+e+m)$ is linearly homomorphic
  - Decryption $D(a,b) = b - as = m+e$ is linear in $s' = (-s, 1)$
  - We can decrypt homomorphically using an encryption of s'
- Details
  - Given: $E(m)=(a,b)$ and $E'(s')=(E'(-s),E'(1))$
  - Compute $E(m)*E'(s') = a*E'(-s)+b*E'(1)=E(m)$
- More interesting:
  - Given $E(m)$ and $E'(cs')$
  - Compute $E(m)*E'(cs') = E(cm)$

Homomorphic “decrypt and multiply”

- $E''(c) = E'(cs') = E'($“$E(m) \rightarrow c*m$”$)$
- $E''(c) = \{E(\alpha_i c)\}_i$ for some $\alpha_i(s)$
- Homomorphic Properties:
  - $E''(m_1) + E''(m_2) = E''(m_1+m_2)$
  - $E''(m_1)*E''(m_2)$

    $=\{E(\alpha_i m_1)*E''(m_2)\}_i$

    $=\{E(\alpha_i m_1*m_2)\}$

    $= E''(m_1*m_2)$

GSW Encryption

- [Gentry,Sahai,Waters'13]
  - FHE based on "approximate eigenvectors"
  - Essentially equivalent to E''(m)
- [Alperin-Sheriff,Peikert'14]
  - Use E'' to implement homomoprhic decrypt.
  - $E_s(m;\beta)$ @ $E_s''(s) = E_s(m;\beta')$
  - $\beta' \ll \beta$: Fully Homomorphic Encryption via bootstrapping [Gentry 2009]

Many other FHE variants

- [Brakerski,Gentry,Vaikuntanathan’12]
- [Brakerski’12 / Fan,Vercauteren’12]
- HELib [Halevi,Shoup’13]
- FHEW,TFHE,HEAAN,…
- All based on similar building blocks and techniques
- Complexity of bootstrapping still main efficiency bottleneck

FHEW / TFHE

- [Ducas, M. 2015] FHEW
  - Multiplication via addition:
  - $m_1, m_2 \in \{0,1\} \subset \{0,1,2,3\}$
  - $m_1 + m_2 \in \{0,1,2\}$: $2 \leftrightarrow m_1 = m_2 = 1$
  - $(m_1 + m_2)/2 = m_1 * m_2$
  - Allows fast bootstrapping (<1 sec)
- [Chillotti,Gama,Georgieva,Izabachene'16]
  - TFHE: improved bootstrapping (<0.1 sec)
- [M., Sorrell'18] Amortized FHEW bootstrapping

Approximate FHE

- HEAAN [Cheon,Kim,Kim,Song’16]
  - HE for Arithmetic on Approximate Numbers
  - Many real world applications deal with approximate (floating point) data
  - D(a,b)=m+e is ok
  - no need to scale m, results in much better performance in many applications
  - Allows to use numerical techniques

Combining different schemes

- Chimera [Boura,Gama,Georgieva’18]
  - uses linearity of decryption to convert between different FHE
  - allows combined use of B/FV, TFHE, HEAAN
- [Choudhury,Loftus,Orsini,Patra,Smart’13]
  - similar idea used to bridge FHE and Multi Party Computation (MPC) protocols

Open Problems

- In practice, bootstrapping still slow
  - active area of research and implementation
  - can bootstrapping be avoided completely?
- Main theoretical problem
  - E_s''(m) = {E_s(α(s)*m)} is circular secure! (E_s can securely encrypt linear functions of s, under standard LWE assumption.)
  - FHE also requires circular security of E_s''(s) to reduce error.
  - Can security of E_s''(s) be proved based on standard LWE?

Thank You!

Questions?