IAIK

TU Graz

SCIENCE
PASSION
TECHNOLOGY

# Secure Classification as a Service

## Levelled Homomorphic, Post-Quantum Secure Machine Learning Inference based on the CKKS Encryption Scheme

Peter Waldert

Bachelor Thesis Presentation, 01.08.2022

## What do we want?

- Step 1: Encrypted Machine Learning (ML) as a service
- Step 2: Using Homomorphic Encryption
- Step 3: That is Post-Quantum secure
- Step 4: Which is somewhat fast
- Step 5: And accurate enough

And of course, an actual implementation.

## Privacy-Preserving Machine Learning (PPML)

- Machine Learning allows us to solve otherwise difficult problems.
- Development of new applications and solutions 'of numerical nature' in different fields
  - Example: Health Care with highly sensitive medical data.
  - Even more volatile results: disease indicators.
- $\Rightarrow$ Demand for privacy-preserving solutions in ML applications.

# Feedforward Neural Networks

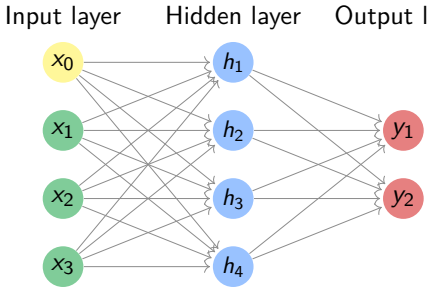
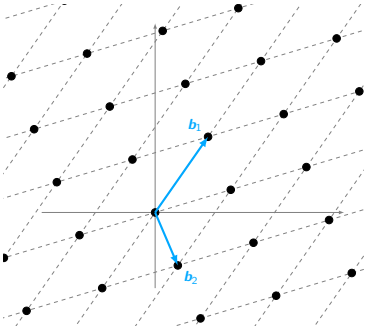



Figure: The simple neural network used in our demonstrator with $\boldsymbol{h} = \mathrm{relu}(M_1\boldsymbol{x} + \boldsymbol{b_1})$ and the output $\boldsymbol{y} = \mathrm{softmax}(M_2\boldsymbol{h} + \boldsymbol{b_2})$.

$\Rightarrow$ Need: Addition, Multiplication, Packing, Rotations. Trained in plain.

# Lattices




## Definition (Lattice)

A lattice $(\mathcal{L}, +, \cdot)$ is a vector field over the integers $(\mathbb{Z}, +, \cdot)$, given $n$ basis vectors $\boldsymbol{b_1}, \boldsymbol{b_2}, ..., \boldsymbol{b_n} \in \mathbb{R}^n$, with

$$\mathcal{L} := \left\{ \sum_{i=1}^{n} c_i \boldsymbol{b}_i \,\middle|\, c \in \mathbb{Z} \right\} \subseteq \mathbb{R}^n .$$

# The Learning With Errors (LWE) Problem

## Definition (LWE-Distribution $A_{\boldsymbol{s},\chi_{error}}$)

Given a prime $q \in \mathbb{N}$ and $n \in \mathbb{N}$, choose a secret $\boldsymbol{s} \in (\mathbb{Z}/q\mathbb{Z})^n$. Sampling from $A_{\boldsymbol{s},\chi_{error}}$:

- Sample a uniformly random vector $a \in (\mathbb{Z}/q\mathbb{Z})^n$.
- Sample a scalar 'error term' $\mu \in \mathbb{Z}/q\mathbb{Z}$ from $\chi_{error}$.
- Compute a noisy inner product $b = \boldsymbol{s} \cdot \boldsymbol{a} + \mu$.
- Output the pair $(\boldsymbol{a}, b) \in (\mathbb{Z}/q\mathbb{Z})^n \times (\mathbb{Z}/q\mathbb{Z})$.

Search-LWE-Problem: Given $m$ independent samples $(\boldsymbol{a}_i, b_i)_{0<i\leq m}$ from $A_{\boldsymbol{s},\chi_{error}}$, find $\boldsymbol{s}$.

Published by Regev in 2005 [6]. Lead to the FHE scheme by Gentry in 2009 [2].

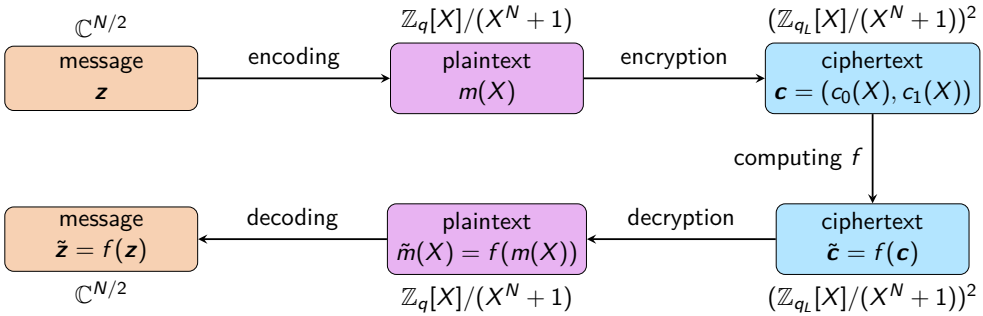
## What is CKKS?

- Levelled Homomorphic Encryption Scheme [1].

$$\forall m_1, m_2 : \mathcal{E}(m_1) + \mathcal{E}(m_2) = \mathcal{E}(m_1 + m_2) \text{ and } \mathcal{E}(m_1) \cdot \mathcal{E}(m_2) = \mathcal{E}(m_1 \cdot m_2)$$

- Enables Public-Key (Asymmetric) Cryptography.
- Approximative Floating-Point Arithmetic.
- Security based on Learning With Errors.
- Single Instruction Multiple Data (SIMD) Encoding.

# Overview of Cheon-Kim-Kim-Song (CKKS)



## Encryption and Decryption

Public key $\boldsymbol{p} = (b, a)$ with $b = -(as + \tilde{\mu})$, secret key $s$, probability distributions $\chi_{enc}$, $\chi_{error}$, plaintext (=message) $m \in R/qR$, ciphertext $\boldsymbol{c}$.

CKKS.

| | |
|---|---|
| Encrypt$(\boldsymbol{p}, m)$ | Let $(b, a) = \boldsymbol{p}$, $u \leftarrow \chi_{enc}$, $\mu_1, \mu_2 \leftarrow \chi_{error}$, then the ciphertext is $\boldsymbol{c} = u \cdot \boldsymbol{p} + (m + \mu_1, \mu_2) = (m + bu + \mu_1, au + \mu_2) \quad \rightarrow \boldsymbol{c}$ |
| Decrypt$(s, \boldsymbol{c})$ | Decrypt the ciphertext $\boldsymbol{c} = (c_0, c_1)$ as $m = [c_0 + c_1 s]_{q_L} \quad \rightarrow m$ |

Leaves the attacker with the Learning With Errors on Rings (RLWE) problem.

## Homomorphic Addition

$$\mathsf{CKKS.Add}(\boldsymbol{c}, \boldsymbol{c}') \quad \text{Output } \overline{\boldsymbol{c}} = \boldsymbol{c} + \boldsymbol{c}' = \begin{pmatrix} \delta(m + m') + b(u + u') + (\mu_1 + \mu_1') \\ a(u + u') + (\mu_2 + \mu_2') \end{pmatrix}^T$$

Indeed, the ciphertext $\overline{\boldsymbol{c}}$ correctly decrypts back to $\overline{m} := m + m'$:

$$\begin{aligned}
\mathsf{CKKS.Decrypt}(s, \overline{\boldsymbol{c}}) &= \lfloor \delta^{-1}[\overline{c_0} + \overline{c_1} s]_t \rceil \\
&= \lfloor \delta^{-1}[\delta\overline{m} + b\overline{u} + \overline{\mu_1} + (a\overline{u} + \overline{\mu_2})s]_t \rceil \\
&= \lfloor [(\delta^{-1}\delta)\overline{m} + \delta^{-1} b\overline{u} + \delta^{-1}\overline{\mu_1} + \delta^{-1} as\overline{u} + \delta^{-1}\overline{\mu_2} s]_t \rceil \\
&= \lfloor [\overline{m} - \cancel{\delta^{-1} as\overline{u}} - \delta^{-1}\tilde{\mu}\overline{u} + \delta^{-1}\overline{\mu_1} + \cancel{\delta^{-1} as\overline{u}} + \delta^{-1}\overline{\mu_2} s]_t \rceil \\
&= \lfloor [\overline{m} + \underbrace{\delta^{-1}(\overline{\mu_1} + \overline{\mu_2} s - \tilde{\mu}\overline{u})}_{:=\epsilon, ||\epsilon|| \ll 1}]_t \rceil \approx \lfloor [\overline{m}]_t \rceil = \lfloor \overline{m} \rceil \approx \overline{m}
\end{aligned}$$

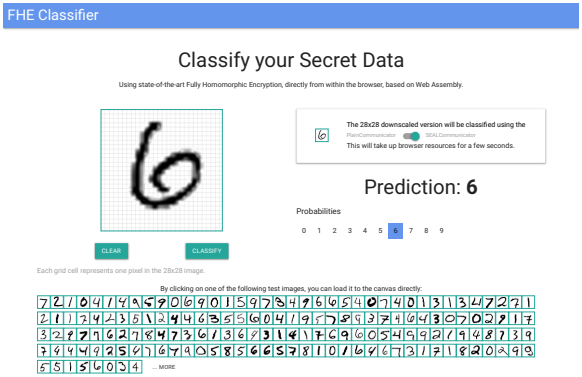
## What about Long-Term Security?

Quantum Computers affect Cryprography today:

- Problems believed to be NP-hard on classical computers can be computed in polynomial time using a quantum computer.
- No hardness proof of the RSA problem exists.
- SHOR's, GROVER's and other algorithms can 'break' many cryptographic schemes used today.
- Existence of a powerful quantum computer endangers the security of TLS, etc.

Our Webservice is (from the point of todays knowledge) still secure in the presence of a quantum computer.

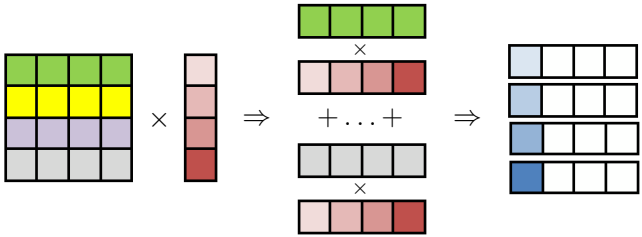
# Let's Check: Secure Handwritten Digit Classification as a Service - Demo




Scan the QR-Code:

Figure: https://secure-classification.peter.waldert.at/.

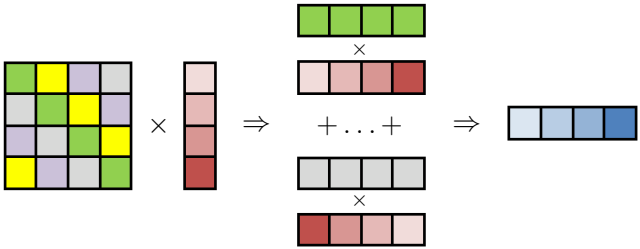
# Matrix Multiplication: The Naïve Method




$$\{M\boldsymbol{x}\}_i = \sum_{j=1}^{t} M_{ij} x_j \,.$$

Image adapted from [3].

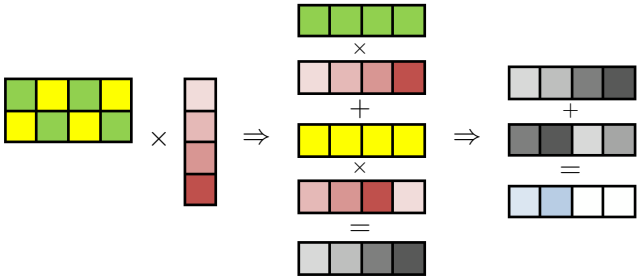
# Matrix Multiplication: The Diagonal Method




$$M\boldsymbol{x} = \sum_{j=0}^{t-1} \mathrm{diag}_j(M) \cdot \mathrm{rot}_j(\boldsymbol{x}) \,.$$

Image adapted from [3].

# Matrix Multiplication: The Hybrid Method



$$M\boldsymbol{x} = (y_i)_{i \in \mathbb{Z}/s\mathbb{Z}} \text{ with } \boldsymbol{y} = \sum_{k=1}^{t/s} \mathrm{rot}_{ks}\left( \sum_{j=1}^{s} \mathrm{diag}_j(M) \cdot \mathrm{rot}_j(\boldsymbol{x}) \right).$$

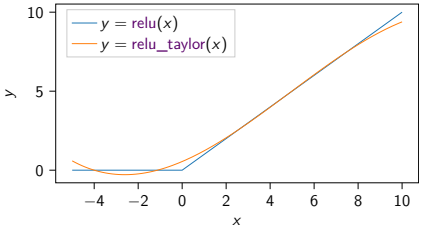
Image adapted from [3].

# Polynomial Evaluation

- In between the dense layers, we need to evaluate the $\text{relu}(x) := max(x, 0)$ function.
  $\Rightarrow$ Approximate it by a series expansion...

$$\text{relu\_taylor}(x) = -0.006137x^3 + 0.090189x^2 + 0.59579x + 0.54738\,.$$

- The softmax activation at the end can be done by the client.



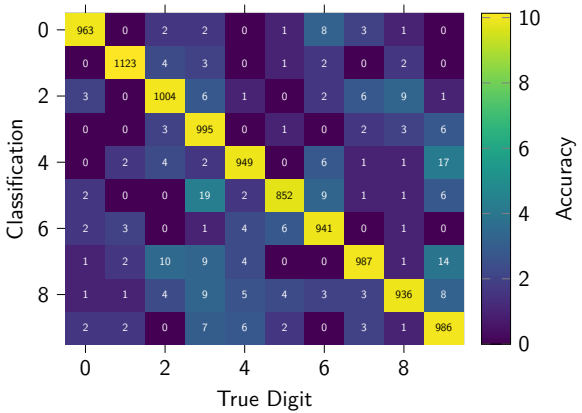
# Runtime Benchmarks & Communication Overhead

Table: Performance benchmarks and communication overhead of the classification procedure on an Intel® i7-5600U CPU, including the encoding and decoding steps.

| Mode | SecLevel | $B_1$ | $B_2$ | $N$ | MatMul | $T$ / s | $M$ / MiB | $\Delta$ / 1 |
|---|---|---|---|---|---|---|---|---|
| Release | tc128 | 34 | 25 | 8192 | Diagonal | 8.39 | 132.72 | 0.0364 |
| | | | | | Hybrid | 1.35 | 132.72 | 0.0362 |
| | | | | | BSGS | 1.66 | 132.72 | 0.1433 |
| | tc128 | 60 | 40 | 16384 | Diagonal | 17.24 | 286.51 | 0.0363 |
| | | | | | Hybrid | 3.05 | 286.51 | 0.0364 |
| | | | | | BSGS | 3.66 | 286.51 | 0.1399 |
| | tc256 | 60 | 40 | 32768 | Diagonal | 35.24 | 615.16 | 0.0363 |
| | | | | | Hybrid | 5.99 | 615.16 | 0.0364 |
| | | | | | BSGS | 7.34 | 615.16 | 0.1399 |

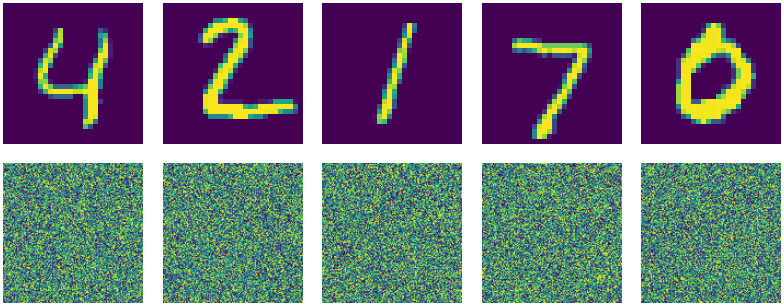
In Plain: 784 byte requests, taking 50 µs; Encrypted: 132 MiB requests, taking 1.4 s.

# Chaos everywhere: The Confusion Matrix




Plain Accuracy: 97.6 %, Encrypted Accuracy: 97.3 %.

# Ciphertext Visualisations



## What did we do?

- Designed a neural network that solves our problem.
  - Homomorphic encryption enables private inference.
  - CKKS is inherently post-quantum secure.
- Implemented a server in C++ with a frontend interface.
- Optimized using different matrix multiplication techniques and proved that they work.
- Compared results.
- Future Work: More Complicated Networks (CNNs, etc.).

Questions?

# Glossary I

| | | |
|---|---|---|
| BSGS | Babystep-Giantstep | 34 |
| CKKS | Cheon-Kim-Kim-Song | 8 |
| FHE | Fully Homomorphic Encryption | 6 |
| LWE | Learning With Errors | 6 |
| ML | Machine Learning | 2 |
| MNIST | Modified National Institute of Standards and Technology | 33 |
| NP | Non-deterministic Polynomial time | 11 |
| PPML | Privacy-Preserving Machine Learning | 3 |
| RLWE | Learning With Errors on Rings | 9 |
| RSA | Rivest-Shamir-Adleman | 37 |
| SIMD | Single Instruction Multiple Data | 7 |
| TLS | Transport Layer Security | 11 |

# Bibliography I

[1] Jung Hee Cheon, Andrey Kim, Miran Kim and Yongsoo Song. **Homomorphic Encryption for Arithmetic of Approximate Numbers**. ASIACRYPT. 2017.

[2] Craig Gentry. **Fully homomorphic encryption using ideal lattices**. STOC '09. 2009.

[3] Daniel Huynh. **Cryptotree: fast and accurate predictions on encrypted structured data**. (2020). DOI: 10.48550/ARXIV.2006.08299. URL: https://arxiv.org/abs/2006.08299.

[4] Yann LeCun and Corinna Cortes. **The MNIST database of handwritten digits**. 1998. URL: http://yann.lecun.com/exdb/mnist/.

[5] Vadim Lyubashevsky, Chris Peikert and Oded Regev. **A Toolkit for Ring-LWE Cryptography**. IACR Cryptol. ePrint Arch. 2013.

[6] Oded Regev. **On lattices, learning with errors, random linear codes, and cryptography**. STOC '05. 2005.

[7] Ronald L Rivest, Adi Shamir and Leonard M Adleman. **Cryptographic communications system and method**. US Patent 4,405,829. Sept. 1983.

# Bibliography II

[8] Peter W. Shor. **Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer**. *SIAM Journal on Computing* 26.5 (Oct. 1997), pp. 1484–1509. DOI: 10.1137/s0097539795293172.

## Some more Details...

Additional Material omitted in main talk.

- Learning With Errors on Rings
- Encoding and Decoding transformations
- The BabyStep-Giantstep method
- Proof of Diagonal, Hybrid method
- Shor's Algorithm

# Polynomial Rings

## Definition (Cyclotomic Polynomial)

Given the $n^{\text{th}}$ roots of unity $\{\xi_k\}$, define $\Phi_n \in \mathbb{Z}[X]$ as

$$\Phi_n(x) := \prod_{\substack{k=1 \\ \xi_k \text{primitive}}}^{n} (x - \xi_k)\,.$$

It is unique for each given $n \in \mathbb{N}$.

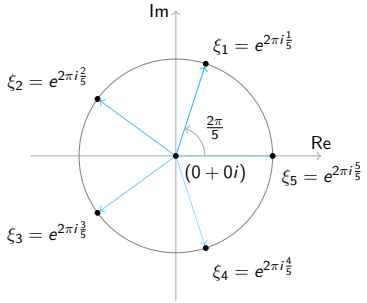



Figure: The $5^{\text{th}}$ roots of unity

## Some Notation

- $\mathbb{Z}[X] := \{p : \mathbb{C} \mapsto \mathbb{C}, p(x) = \sum_{k=0}^{\infty} a_k x^k, a_k \in \mathbb{Z}\ \forall k \geq 0\}$
  - Complex-valued Polynomials with integer coefficients.
- $\mathbb{Z}_q[X] := (\mathbb{Z}/q\mathbb{Z})[X] = \mathbb{Z}[X]/q\mathbb{Z}[X]$
- $\mathbb{Z}_q[X]/\Phi_M(X) \underbrace{= \mathbb{Z}_q[X]/(X^N + 1)}_{\text{for } M=2N, \text{ powers of 2}}$ using the $M^{\text{th}}$ cyclotomic polynomial $\Phi_M(X)$.
  - Its elements are polynomials of degree $(N-1)$[1] with integer coefficients mod $q$.

[1] For general $M$, degree $\varphi(M) - 1$

# The RLWE Problem

## Definition (RLWE-Distribution $B_{s,\chi_{error}}$)

Given a quotient ring $(R/qR, +, \cdot)$, choose a secret $s \in R/qR$. Sampling from the RLWE distribution $B_{s,\chi_{error}}$:

- Uniformly randomly draw an element $a \in R/qR$
- Sample 'noise' $\mu \in R/qR$ from $\chi_{error}$.
- Set $b = s \cdot a + \mu$, with $\cdot$ denoting the ring multiplication operation.
- Output the pair $(a, b) \in R/qR \times R/qR$.

Proven equivalent to LWE.

Use Search-RLWE to construct a cryptosystem... Idea: Attacker needs to solve LWE given the public key to recover the secret $s$.

## Encoding and Decoding

CKKS.

Encode($\boldsymbol{z}$) For a given input vector $\boldsymbol{z}$, output
$m = (\underline{\sigma}^{-1} \circ \underline{\rho_\delta}^{-1} \circ \underline{\pi}^{-1})(\boldsymbol{z}) = \underline{\sigma}^{-1}(\lfloor \delta \cdot \underline{\pi}^{-1}(\boldsymbol{z}) \rceil_{\underline{\sigma}(R)}) \quad \rightarrow m$

Decode($m$) Decode plaintext $m$ as $\boldsymbol{z} = (\underline{\pi} \circ \underline{\rho_\delta} \circ \underline{\sigma})(m) = (\underline{\pi} \circ \underline{\sigma})(\delta^{-1} m) \quad \rightarrow \boldsymbol{z}$

- Three transformations: $\underline{\sigma}^{-1}$, $\underline{\rho_\delta}^{-1}$ and $\underline{\pi}^{-1}$.
- Key idea: Homomorphic property, they preserve additivity and multiplicativity.
- Allows for homomorphic SIMD operations.

## Definition (Canonical Embedding $\underline{\sigma}$)

For a real-valued polynomial $p \in \mathcal{S}$, define the canonical embedding of $\mathcal{S}$ in $\mathbb{C}^N$ as a mapping $\underline{\sigma} : \mathcal{S} \mapsto \mathbb{C}^N$ with

$$\underline{\sigma}(p) := \left(p(e^{-2\pi ij/N})\right)_{j \in \mathbb{Z}_d^*}$$

with $\mathbb{Z}_d^* := \{x \in \mathbb{Z}/d\mathbb{Z} \,|\, \gcd(x, d) = 1\}$ the set of all integers smaller than $d$ that do not share a factor $> 1$ with $d$. The image of $\underline{\sigma}$ given a set of inputs $R$ shall be denoted as $\underline{\sigma}(R) \subseteq \mathbb{C}^N$. Let the inverse of $\underline{\sigma}$ be denoted by $\underline{\sigma}^{-1} : \mathbb{C}^N \mapsto \mathcal{S}$.

## Definition (Discretisation to an element of $\underline{\sigma}(R)$)

Using one of several round-off algorithms (cf. [5]), given an element of $\mathbb{H}$, define a rounding operation $\underline{\rho}^{-1} : \mathbb{H} \mapsto \underline{\sigma}(R)$ that maps an $\boldsymbol{h} \in \mathbb{H}$ to its closest element in $\underline{\sigma}(R) \subset \mathbb{H}$, also denoted as

$$\underline{\rho}^{-1}(\boldsymbol{h}) := \lfloor \boldsymbol{h} \rceil_{\underline{\sigma}(R)} .$$

Further let $\underline{\rho_\delta}^{-1}(\boldsymbol{h}) = \lfloor \delta \cdot \boldsymbol{h} \rceil_{\underline{\sigma}(R)}$ denote the same rounding operation but with prior scaling by a scalar factor $\delta$. Note that $\underline{\rho}$ is given directly as the identity operation because all elements of its domain are already elements of its image. Similarly, $\underline{\rho_\delta}(\boldsymbol{y}) = \delta^{-1} \cdot \boldsymbol{y}$.

## Definition (Natural Projection $\underline{\pi}$)

Let $T$ be a multiplicative subgroup of $\mathbb{Z}_d^*$ with $\mathbb{Z}_d^*/T = \{\pm 1\} = \{1T, -1T\}$, then the natural projection $\underline{\pi} : \mathbb{H} \mapsto \mathbb{C}^{N/2}$ is defined as

$$\underline{\pi}\left((z_j)_{j\in\mathbb{Z}_M^*}\right) := (z_j)_{j\in T}$$

Let its inverse be denoted by $\underline{\pi}^{-1} : \mathbb{C}^{N/2} \mapsto \mathbb{H}$ and consequently defined as

$$\underline{\pi}^{-1}\left((z_j)_{j\in T}\right) := \left(\nu(z_j)\right)_{j\in\mathbb{Z}_M^*} \text{ with } \nu(z_j) = \begin{cases} z_j & \text{if } j \in T \\ \overline{z_j} & \text{otherwise} \end{cases}$$

## Goal: Classify MNIST Images of Handwritten Digits

- Two major types of ML: Supervised and Unsupervised Learning
- Popular dataset: Modified National Institute of Standards and Technology (MNIST). Encode as vector of 784 entries.

Figure: Sample images of the MNIST database of handwritten digits [4]. The dataset contains 70,000 images of $28 \times 28$ greyscale pixels valued from 0 to 255 as well as associated labels (as required for supervised learning).

## Theorem (Babystep-Giantstep Method)

*Given a matrix $M \in \mathbb{R}^{t \times t}$ and a vector $\boldsymbol{x} \in \mathbb{R}^t$, with $t = t_1 \cdot t_2$ split into two Babystep-Giantstep (BSGS) parameters $t_1, t_2 \in \mathbb{N}$ and*

$$diag'_p(M) = rot_{-\lfloor p/t_1 \rfloor \cdot t_1}(diag_p(M)),$$

*one can express a matrix-vector multiplication as follows:*

$$M\boldsymbol{x} = \sum_{k=0}^{t_2-1} rot_{(kt_1)}\left( \sum_{j=0}^{t_1-1} diag'_{(kt_1+j)}(M) \cdot rot_j(\boldsymbol{x}) \right)$$

*where $\cdot$ denotes an element-wise multiplication of two vectors.*

## Proof (Diagonal Method).

For all indices $i \in \mathbb{Z}/t\mathbb{Z}$,

$$\left\{ \sum_{j=0}^{t-1} \mathrm{diag}_j(M) \cdot \mathrm{rot}_j(\boldsymbol{x}) \right\}_i = \sum_{j=0}^{t-1} M_{i,(i+j)} x_{i+j} \overset{[k=i+j]}{=} \sum_{k=i}^{t+i-1} M_{ik} x_k = \sum_{k=0}^{t-1} M_{ik} x_k = \{M\boldsymbol{x}\}_i \,.$$

□

## Proof (Hybrid Method).

For all indices $i \in \mathbb{Z}/s\mathbb{Z}$,

$$\{\boldsymbol{y}\}_i = \left\{ \sum_{k=1}^{t/s} \mathrm{rot}_{ks}\left( \sum_{j=1}^{s} \mathrm{diag}_j(M) \cdot \mathrm{rot}_j(\boldsymbol{x}) \right) \right\}_i = \sum_{k=1}^{t/s} \sum_{j=1}^{s} M_{i,(i+j)+ks} x_{(i+j)+ks} \,,$$

substituting $l = i + j + ks$ and condensing the nested sums into one single summation expression since $\sum_{k=1}^{t/s} \sum_{j=1}^{s} f(j + ks) = \sum_{l=1}^{t} f(l)$, we obtain

$$y_i = \sum_{l=1+i}^{t+i} M_{il} x_l = \sum_{l=1}^{t} M_{il} x_l = \{M\boldsymbol{x}\}_i \,.$$

□

## The Rivest-Shamir-Adleman (RSA) Scheme

From the integers $\mathbb{Z}$, define the quotient ring $(\mathbb{Z}/q\mathbb{Z}, +, \cdot)$ for some modulus $q \in \mathbb{N}$.

With unpadded RSA [7], $\mathcal{E} : \mathbb{Z}/q\mathbb{Z} \mapsto \mathbb{Z}/q\mathbb{Z}$

$$\mathcal{E}(m) := m^r \mod q \quad r, q \in \mathbb{N}$$

applied to the messages $m_1, m_2 \in \mathbb{Z}/q\mathbb{Z}$ respectively, the following holds:

$$\begin{aligned} \mathcal{E}(m_1) \cdot \mathcal{E}(m_2) &\equiv (m_1)^r (m_2)^r \mod q \\ &\equiv (m_1 m_2)^r \mod q \\ &\equiv \mathcal{E}(m_1 \cdot m_2) \mod q \end{aligned}$$

$\Rightarrow$ A Group Homomorphism!

## SHOR's Algorithm I

Peter SHOR's algorithm was published in 1994 [8] and will be outlined here shortly as it is a core element to security considerations of modern cryptosystems. The core structure of the algorithm is

1. guessing some $g \in \mathbb{N}$ that we hope shares a factor with a large $N = p \cdot q$ $(p, q, N \in \mathbb{N})$,
2. improving that guess $g$ by a quantum subroutine and
3. applying EUCLID's algorithm to find $p$ and $q$ the factors of $N$.

## Shor's Algorithm II

The core factorisation idea is the following, not specific to quantum computation: We know that for a pair $g, N \in \mathbb{N}$, we can always find some $r \in \mathbb{N}$ such that

$$g^r = mN + 1,\ m \in \mathbb{N},$$

we are looking for a $g^r$ that is exactly one more than a multiple of $N$. Rearranging,

$$g^r - 1 = mN \iff (g^{\frac{r}{2}} + 1)(g^{\frac{r}{2}} - 1) = mN$$

we have found two factors $g^{\frac{r}{2}} + 1$ and $g^{\frac{r}{2}} - 1$ (for even $r$) that share a common factor with $N$ and apply Euclid's algorithm to get $p$ and $q$.

## Shor's Algorithm III

Thereby, we instruct the quantum computer to raise our guess $g$ by all possible powers $\in \mathbb{N}$ up to some boundary in order to obtain

$$|1, g^1\rangle + |2, g^2\rangle + |3, g^3\rangle, ...$$

which we then take modulo $N$, resulting in a superposition of remainders

$$|1, [g^1]_N\rangle + |2, [g^2]_N\rangle + |3, [g^3]_N\rangle + ... .$$

Here is where Shor's key idea came in: The remainders in the above superposition expose repetitions at a period of exactly $r$ (which, by our definition fulfils $g^r \equiv 1 \mod N$)

$$g^x \equiv g^{x+r} \equiv g^{x+2r} \equiv ... \equiv g^{x+ar} \mod N$$

## Shor's Algorithm IV

the remainders are periodic with frequency $\frac{1}{r}$.

The above can be quickly derived from $g^r = mN + 1$, therefore

$$g^{x+r} = g^x g^r = (\tilde{m}N + [g^x]_N)(mN + 1) = (m\tilde{m}N + [g^x]_N m + \tilde{m})N + [g^x]_N$$

is indeed congruent to $g^x \mod N$.

From the output of

$$\mathrm{QFT}(|1, [g^1]_N\rangle + |2, [g^2]_N\rangle + |3, [g^3]_N\rangle + ...)$$

we obtain the dominant frequency $\frac{1}{r}$ yielding us our desired improved guess [8].