2023 62nd IEEE Conference on Decision and Control (CDC)
December 13-15, 2023. Marina Bay Sands, Singapore

Homomorphically encrypted gradient descent algorithms for quadratic programming

André Bertolace, Konstantinos Gatsis, Kostas Margellos

***Abstract*— In this paper, we evaluate the different fully homomorphic encryption schemes, propose an implementation, and numerically analyze the applicability of gradient descent algorithms to solve quadratic programming in a homomorphic encryption setup. The limit on the multiplication depth of homomorphic encryption circuits is a major challenge for iterative procedures such as gradient descent algorithms. Our analysis not only quantifies these limitations on prototype examples, thus serving as a benchmark for future investigations, but also highlights additional trade-offs like the ones pertaining the choice of gradient descent or accelerated gradient descent methods, opening the road for the use of homomorphic encryption techniques in iterative procedures widely used in optimization based control. In addition, we argue that, among the available homomorphic encryption schemes, the one adopted in this work, namely CKKS, is the only suitable scheme for implementing gradient descent algorithms. The choice of the appropriate step size is crucial to the convergence of the procedure. The paper shows firsthand the feasibility of homomorphically encrypted gradient descent algorithms.**

I. Introduction

Homomorphic encryption (HE) is a ground-breaking mathematical method that enables the analysis or manipulation of encrypted data without revealing its content [14]. In doing so, HE permits the secure delegation of data processing to third-party cloud providers. Several encryption schemes, such as Paillier [26] or El Gamal [9] are partially HE schemes[1]. In 2009, Gentry [13] proposed the first fully HE scheme[1]. The computational overhead of the scheme was significant, but it showed that such schemes are indeed possible. Since then these approaches have been further developed. Currently, the state of the art schemes are BFV [3], [10], CKKS [7], BGV [4], and GSW [15]. The computational overhead remains large but it has been brought down to a level where these schemes can be implemented in practice.

In applications of control and decision-making, the benefits of delegating data processing without giving away access to the data are tremendous. The use of HE schemes applied in control theory is at its infancy, however, encrypted linear controllers have been implemented. Most results use partially HE schemes [1], [2], [6], [11], [12], [28], but approaches using fully HE schemes also exist [21], [23], [29]. In addition, [27] provides a detailed overview of the current status of research in the encrypted control for networked systems and [20] a comparison of different encrypted control approaches.

Yet, the implementation of algorithms in a HE setup is far from trivial. For instance, many HE schemes use random noise to guarantee the security of the encrypted data. This noise compounds at every arithmetic operation, resulting in a limited number of sequential arithmetic operations performed by an encryption circuit, in special, multiplication operations.

In this paper, we would like to understand the limits imposed by HE computation on challenging computation tasks beyond the controller implementation problems studied in the literature. Specifically, we consider the problem of solving quadratic programming (QP) problems. QP is commonly used in several control problems like those arising in state estimation under minimum square error and model predictive control. Numerically solving such a task often requires *iterative* methods (gradient descent) and the limit on the multiplication depth of HE circuits is a major challenge for iterative procedures. As a result, given the HE multiplication depth limits, we would like to determine the most appropriate iterative methods for QP. In our case, we adapt and implement gradient descent (GD) and accelerated gradient descent (AGD) algorithms to solve a QP in a HE manner. Our contributions are threefold:

1) We argue that among the available HE schemes CKKS is the only scheme suitable to handle GD and AGD iterations in a HE setup as it allows handling real-valued operations, an important feature, especially in the selection of an appropriate step-size that ensures the convergence of the underlying algorithm.
2) We implement our own HE matrix multiplication algorithm that is more efficient, in terms of multiplication depth, than other algorithms in the literature [18].
3) We demonstrate that in the HE setup, the condition number of the quadratic term matrix influences algorithm preference. HE-AGD is favored only for higher-condition-number matrices, while plain-text optimization typically favors AGD due to its superior convergence rate. This is because AGD incurs an extra multiplication step compared to GD within the fixed encryption circuit, which proves beneficial, particularly for matrices with lower condition numbers.

Other works in the literature, [1], also proposed to solve QPs in a secure/distributed manner. We differ from this work by using fully homomorphic encryption instead of partial homomorphic encryption schemes. Furthermore, [29], presented an encrypted model predictive control scheme for

The authors are with the Department of Engineering Science, University of Oxford, Oxford OX1 3PJ, U.K. E-mail: andre.bertolace@eng.ox.ac.uk, konstantinos.gatsis@eng.ox.ac.uk, kostas.margellos@eng.ox.ac.uk

Extended version: https://arxiv.org/abs/2309.01559

[1]Partially HE schemes enable the implementation of either addition or multiplication on encrypted data, but not both, whereas fully HE schemes enable the implementation of both addition and multiplication operations.

linear constrained systems, also using partial homomorphic encryption schemes and not solving the QP but instead using the corresponding piece-wise affine control law.

It is important to note that our paper focuses on unconstrained quadratic problems due to inherent limitations in operations that are allowed to be performed by available HE schemes. However, our analysis not only quantifies these limitations on prototype examples thus serving as a benchmark for future investigations, but also highlights additional trade-offs like the ones pertaining the choice of GD or AGD methods, opening the road for the use of HE methods in iterative methods widely used in optimization based control.

II. Descent algorithms for unconstrained quadratic programming

QP has been a very successful tool for modeling many real-life problems. It is extensively used in applications that involve the variance minimization, such as in the formulation of portfolio optimization problems or in solving the ordinary least square (OLS) problem. In fact, many problems in physics on engineering can be formulated as some form of energy minimization problem, in which the energy can simply be formulated as a quadratic form, $\min_{x\in\mathbb{R}^n} \frac{1}{2}x^TQx + p^Tx$, where $Q \in \mathbb{R}^{n\times n}$ and $p \in \mathbb{R}^n$. Note that the QP is convex if $Q \succcurlyeq 0$.

This unconstrained QP has a closed form solution, it requires however the inversion of a matrix, a procedure that involves other operations than additions and multiplications, posing hence a challenge for its implementation in a HE setup. An alternative solution is to use gradient descent methods to solve the QP problem. This is a class of iterative algorithms that provide a simple way [5] to minimize a differentiable function f, $\min_{x\in\mathbb{R}^n} f(x)$. Starting at an initial estimate, it iteratively updates, $x_{t+1} = x_t - \eta\nabla f(x_t)$, where $\nabla f(x_t)$ denotes the gradient of f calculated at x_t and η the step-size, until reaching a desired tolerance in the solution. Particularly to the QP case, the gradient takes a linear form, $\nabla f(x) = Qx + p$.

Methods of this type have a convergence rate which is independent of the dimension n of the solution space. This feature makes them particularly attractive for optimization in very high dimensions [5]. The convergence is however deeply linked to the step-size η, be it too small, the algorithm may take too long to converge, be it too high, it may diverge.

Properties such as smoothness or strong convexity of the objective function f do play a relevant role in choosing η and a variant of the algorithm with faster convergence.

Definition 2.1: A continuous differentiable function f is β-smooth if the gradient ∇f is β-Lipschitz, i.e.,

$$\|\nabla f(x) - \nabla f(y)\| \leq \beta \|x - y\|, \ \forall x, y \in \mathbb{R}^n.$$

Definition 2.2: A function f is α-strongly convex, with $\alpha > 0$, if for any x, y it satisfies the following sub-gradient inequality, i.e.,

$$f(x) - f(y) \leq \nabla f(x)^T(x-y) - \frac{\alpha}{2}\|x-y\|^2, \ \forall x, y \in \mathbb{R}^n.$$

Given these definitions, an immediate consequence is that if f is twice differentiable, then f is α-strongly convex if the eigenvalues of the Hessian of f are larger than or equal to α. A quadratic f, as in our QP, is $\lambda_{\max}$-smooth and $\lambda_{\min}$-strongly convex, where $\lambda_{\max}, \lambda_{\min} > 0$ are, respectively, the maximum and minimum eigenvalues of the matrix Q. The ratio $\kappa = \frac{\lambda_{\max}}{\lambda_{\min}}$ is the condition number of the matrix Q and it plays an important role in the convergence of descent algorithms. As reference, for the β-smooth and α-strongly convex quadratic function, both Nesterov's accelerated gradient descent (AGD) and Gradient Descent (GD) methods converge exponentially fast [5], with convergence rate of order, $\mathcal{O}(1/\sqrt{\kappa})$ and $\mathcal{O}(1/\kappa)$ respectively.

III. Homomorphically encrypted arithmetic

A. Homomorphic encryption schemes

HE schemes have been developed using different approaches. BFV and BGV perform operations modulo integer whereas CKKS implements approximated fixed point arithmetics. The security of these schemes is based on the Ring Learning With Errors (RLWE) problem, a variant of the Learning With Errors problem (LWE), in which the goal is to distinguish random linear equations, which have been perturbed by a small amount of noise from uniform ones [25]. The HE cipher is defined by a pair E, D, of encryption-decryption algorithms respectively. E takes a public key pk along with a message m as inputs and outputs a cipher-text c, as $\mathsf{c} = E(pk, m)$. The decryption algorithm, D, takes a secret key sk along with the cipher-text c as inputs and outputs the message $m = D(sk, \mathsf{c})$. The algorithms are parameterized by a security parameter λ which plays a direct role in the derivation of the sk. In addition, these schemes exploit the structure of polynomial rings for its plain-text and cipher-text spaces, the cyclonomic polynomial, $R[\mathbb{Z}_q] = \mathbb{Z}_q[X]/(X^N+1)$. All schemes make use of random variables with values sampled from a discrete Gaussian distribution with a pre-defined variance and random variables sampled from a ternary distribution $\{-1, 0, 1\}$ [7].

BFV: In the BFV scheme, [3], [10], the plain-text and cipher-text spaces are defined by two distinct rings, $R[\mathbb{Z}_t]$ and $R[\mathbb{Z}_q]$, where t and q are parameters of the plain-text and cipher-text coefficients, respectively.

BGV: The BGV scheme, [4], is similar to the BFV. The plain-text and cipher-text spaces are defined by two distinct rings, $R[\mathbb{Z}_t]$ and $R[\mathbb{Z}_q]$.

CKKS: The CKKS scheme, [7], is often quoted as being the most efficient method to perform approximate HE computations over real and complex numbers [19] and it can be considered as a noisy channel [24]. The scheme exploits the structure of integer polynomial rings for its plain-text and cipher-text spaces, $R[\mathbb{Z}_q]$ and $R[\mathbb{C}]$. Given a vector[2] $x \in \mathbb{C}^N$ and a canonical embedding transformation $\sigma : R[\mathbb{C}] \to \mathbb{C}^N$, one applies the inverse embedding transformation to get $\mu = \sigma^{-1}(x) \in R[\mathbb{C}]$, then scale μ by a factor $\Delta = 2^p$ and round to obtain the plain-text $m = \lfloor \Delta \cdot \mu \rceil \in R[\mathbb{Z}_q]$.

[2] The space size is actually $N/2$ because the roots of the cyclonomic polynomial lie on the unit circle and are pairwise complex conjugate.

B. Scheme choice

We claim that the most suitable choice for HE versions of the GD algorithm (similar considerations hold for the AGD one) is the CKKS scheme. The main reason for such claim is related to the selection of the step-size η. Note that η should be sufficiently small for GD to converge. This is summarized in the following proposition; it is a standard result but we present a proof below for completeness.

Proposition 3.1: Consider a QP with $Q \succcurlyeq 0$ with $Q = Q^\top$, and let $\lambda_{\max}$ denote the maximum eigenvalue of Q. The GD method converges for any $\eta < \frac{2}{\lambda_{\max}}$.

Proof: Given $f(x) = \frac{1}{2}x^TQx + p^Tx$, we have that $\nabla f(x) = Qx + p$ and the iterative GD procedure takes the form $x_{t+1} = (I - \eta Q)x_t - \eta p$. Let x^* be an unconstrained minimizer of f. As such, $\nabla f(x^*) = Qx^* + p = 0$, which in turn implies that $x^* = (I - \eta Q)x^* - \eta p$. We thus have that $x_{t+1} - x^* = (I - \eta Q)(x_t - x^*)$ and consequently $\|x_{t+1} - x^*\| \leq \|I - \eta Q\| \, \|(x_t - x^*)\| \leq \|I - \eta Q\|^{t+1} \|(x_0 - x^*)\|$. The latter implies that $\lim_{t\to\infty} \|x_{t+1} - x^*\| = 0$ if the maximum eigenvalue of $(I - \eta Q)$ is less than 1, which can be achieved if $\eta < \frac{2}{\lambda_{\max}}$. ■

A direct consequence of this fact is that if using BGV or BFV that require integer step-sizes, one can only ensure convergence for matrices with $\lambda_{\max} < 2$ that is the only choice that allows for an integer step-size η. The minimum then value of such step-size would be $\eta = 1$, which in turn may lead to an erratic numerical behaviour. Additionally, to be able to use BFV or BGV, one would need to limit the calculations to integer matrices $Q \in \mathbb{Z}^{n\times n}$, or manipulate $Q \in \mathbb{R}^{n\times n}$ to be made integer. Towards this direction, [20], [22], suggest the following manipulations:

- "Scaling-up" the real numbers by a factor, say 10^8, replicating a fixed point arithmetic, and proceed by calculating using the given integer numbers.
- Converting the matrix Q by finding an invertible matrix $T \in \mathbb{R}^{n\times n}$ such that $TQT^{-1} \in \mathbb{Z}^{n\times n}$.

The former is not a practical solution as the result of multiple multiplications will overflow and the output after the decryption will be incorrect [20], whereas the latter implies limiting ourselves to matrices Q in which every eigenvalue has an integer real and imaginary part [20], [22]. In summary, BGV and BFV are only suitable schemes for integer matrices or matrices that have integer eigenvalues, which for our setting would require $\lambda_{\max} < 2$. This would imply working only with identity matrices, $Q = I$, if we working with integer matrices which are symmetric and positive definite. As such, for the purpose of an iterative methodology like GD and AGD, CKKS is preferable.

IV. Homomorphically encrypted gradient descent algorithms

A. Algorithm description

For the HE version of gradient descent methods, let us start by defining the following arithmetic operators:

- $\boldsymbol{+}/\boldsymbol{-}$: the addition/subtraction of two cipher-texts;
- $\bullet$: the multiplication of two cipher-texts;
- $\odot$: the multiplication of a plain-text and a cipher-text;

Let us further assume that the user calculates $\lambda_{\min}$, $\lambda_{\max}$, and sends these as plain-text, i.e. not encrypted, values to the solver. Together with these constants, the user also sends the encrypted matrix and vector, $\mathtt{Q} = E(pk, Q)$ and $\mathtt{p} = E(pk, p)$ that determine the QP. The encrypted version of the descent algorithms will still proceed in an iterative fashion. The only difference is that one would be iterating over cipher-texts $\mathtt{c_t}$ instead of plain-text x_t. When iterating over cipher-texts, two steps deserve special attention, the stopping criteria $|\mathtt{c_{t+1}} \boldsymbol{-} \mathtt{c_t}| > \epsilon$ and the matrix multiplication procedure, referred to MMULT (Algorithm 4) and discussed in the sequel. The latter is relevant because of the exponential growth of the noise level with the multiplication depth. For the stopping rule, determining whether an encrypted value is larger than another encrypted value or even a plain-text without decrypting both values is directly not feasible, but complex approaches to implement comparisons have appeared in [17] for BFV and [8] for CKKS.

Given the challenge to implement the stopping rule in an HE setup, we propose that the HE version of the AGD algorithm is slightly modified:

- Instead of specifying the tolerance ϵ, the user fixes the number of iterations N.
- The user may hand in the initial estimate $\mathtt{x_0}$, although this is not necessary.

Algorithm 1 HE AGD for an unconstrained QP

1: **function** HEAGDQP($\mathtt{Q}, \mathtt{p}, d, \lambda_{\min}, \lambda_{\max}, \mathtt{x_0}, N$)
2: $\quad \kappa \leftarrow \frac{\lambda_{\max}}{\lambda_{\min}}$
3: $\quad \mathtt{x_-} \leftarrow \mathtt{x_0}$
4: $\quad \mathtt{y_-} \leftarrow \mathtt{x_0}$
5: $\quad \eta \leftarrow \frac{-1}{\lambda_{\max}}$
6: $\quad$ **for** $t = 0$ to $N-1$ **do**
7: $\quad\quad \mathtt{y_+} \leftarrow \mathtt{x_-} \boldsymbol{+} \text{MMULT}(\mathtt{Q}, \mathtt{x_-}, d, \eta) \boldsymbol{+} \eta \odot \mathtt{p}$
8: $\quad\quad \mathtt{x_+} \leftarrow \left(1 + \frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1}\right) \odot \mathtt{y_+} \boldsymbol{-} \frac{\sqrt{\kappa}-1}{\sqrt{\kappa}+1} \odot \mathtt{y_-}$
9: $\quad\quad$ RELINEARIZE($\mathtt{x_+}$)
10: $\quad\quad \mathtt{y_-} \leftarrow \mathtt{y_+}$
11: $\quad\quad \mathtt{x_-} \leftarrow \mathtt{x_+}$
12: $\quad$ **end for**
13: $\quad$ **return** $\mathtt{x_+}$
14: **end function**

The HE versions of the the AGD and GD still follow an iterative procedure. These take the form of (Algorithm 1) and (Algorithm 2) respectively and are very similar to the usual AGD and GD algorithms. The main difference is the use of HE arithmetic operators and the special MMULT matrix multiplication procedure.

The two algorithms differ solely in the presence of two $\odot$ and one $\boldsymbol{-}$ operations on line 8 of (Algorithm 1), absent in (Algorithm 2). These operations are integral to the accelerated gradient method, incorporating past information for step updates.

Algorithm 2 HE GD for an unconstrained QP

function HEGDQP($\mathtt{Q}, \mathtt{p}, d, \lambda_{\min}, \lambda_{\max}, \mathtt{x_0}, N$)
  $\kappa \leftarrow \frac{\lambda_{\max}}{\lambda_{\min}}$
  $\mathtt{x}_- \leftarrow \mathtt{x_0}$
  $\eta \leftarrow \frac{-2}{\lambda_{\min}+\lambda_{\max}}$
  for $t = 0$ to $N-1$ **do**
    $\mathtt{x}_+ \leftarrow \mathtt{x}_- \boldsymbol{+} \text{MMULT}(\mathtt{Q}, \mathtt{x}_-, d, \eta) \boldsymbol{+} \eta \odot \mathtt{p}$
    $\mathtt{x}_+ \leftarrow \mathtt{x}_-$
  end for
  return $\mathtt{x}_+$
end function

B. Matrix multiplication seen differently

To explore gradient descent methods, let's examine basic matrix multiplication. Halevi and Shoup [16] introduced Algorithm 3, which efficiently computes linear transformations on encrypted vectors. They leverage diagonal matrix encoding, simplifying matrix-vector multiplication through rotations and constant multiplications. In [18], the authors expand on this method and introduce the JKLS matrix multiplication scheme, enabling ciphered-matrix multiplication with just one ciphertext per matrix, using a row-ordering encoding $A \to a$. While convenient, the JKLS algorithm involves 2 $\odot$ operations.

Algorithm 3 Halevi-Shoup LINTRANS algorithm

function LINTRANS($\mathtt{c}, U$)
  $n \leftarrow \dim(U)$
  $\mathtt{cU} \leftarrow \mathtt{c} \odot u_0$
  for $l = 1$ to $n-1$ **do**
    $\mathtt{cU} \leftarrow \mathtt{cU} \boldsymbol{+} \text{ROT}(\mathtt{c}, l) \odot u_l$
  end for
  RELINEARIZE($\mathtt{cU}$)
  return $\mathtt{cU}$
end function

We propose a modified version of a matrix multiplication algorithm with 1 less $\odot$ multiplication step. Just with this reduction of 1 $\odot$ operation in the matrix the multiplication algorithm we are able to perform 9 and 6 iterations on GD and AGD respectively, as opposed to 6 and 4 iterations if we were using the JKKS multiplication scheme.

$$a_k = V_k \odot a \ , b_k = W_k \odot b \ , \ \ k = 0, \ldots, d-1$$
$$ab = \sum_{k=0}^{d-1} a_k \bullet b_k$$

with:

$$V_k(d \cdot i + j, l) = \begin{cases} 1 & \text{if } l = d \cdot i + [i + j + k]_d \\ 0 & \text{otherwise} \end{cases}$$

$$W_k(d \cdot i + j, l) = \begin{cases} 1, & \text{if } l = d \cdot [i + j + k]_d + j \\ 0, & \text{otherwise} \end{cases}$$

where $[\cdot]_d$ is a shortcut for $\cdot$ modulo d. The matrices V_k and W_k are permuting the row-encoded matrices A and B respectively such that the matrix multiplication algorithm as we know can be implemented with element-wise multiplication and additions.

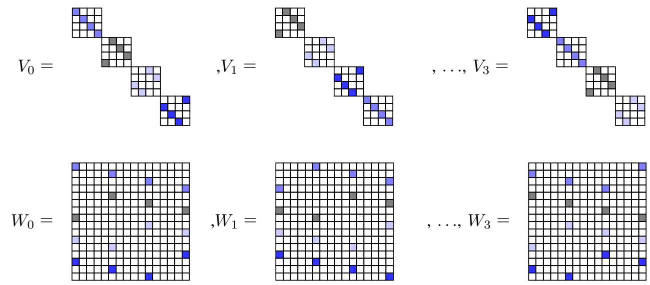

Fig. 1: Matrix multiplication - V_k and W_k examples for $d = 4$

Algorithm 4 HE Matrix Multiplication (MMULT)

function MMULT($\mathtt{A}, \mathtt{B}, d, a$)
  $\mathtt{AB} \leftarrow$ CIPHERTEXT()
  for $k = 0$ to $d-1$ **do**
    $\mathtt{A_k} \leftarrow \text{LINTRANS}(\mathtt{A_0}, V_k(a))$
    $\mathtt{B_k} \leftarrow \text{LINTRANS}(\mathtt{B_0}, W_k(1))$
    $\mathtt{AB_k} \leftarrow \mathtt{A_k} \bullet \mathtt{B_k}$
    RELINEARIZE($\mathtt{AB_k}$)
    $\mathtt{AB} \leftarrow \mathtt{AB} \boldsymbol{+} \mathtt{AB_k}$
  end for
  return $\mathtt{AB}$
end function

C. Extension to other QP problems

Extension to other QP problems is feasible. For instance, linear equality constraints could be handled by converting the problem to an unconstrained QP, or by solving primal-dual methods. These approaches sound completely viable but are subject to the multiplication depth limitations on the HE circuit. In other words, extra arithmetic operations can take place, but at the cost of reducing the number of maximum iterations. Linear inequality constraints are not directly supported, but an approach would be to decrypt and re-encrypt at each iteration (not actually a practical solution).

V. NUMERICAL ANALYSIS

The HE resource requirements are directly proportional to the capacity of the encryption circuit. The larger the circuit's capacity, the larger the computing memory and computational power required at each arithmetic operation. Given our computing resources and the parameters of the Microsoft SEAL [30], the largest circuit we can implement, in the CKKS scheme, has a multiplication depth of 18. At each iteration, AGD and GD have a multiplication depth of 3 and 2, resulting in a cap of 6 and 9 steps for AGD and GD respectively. Even though we could not implement longer iterations due to limitations on our computational resources, we believe that the results would apply in that case too.

We start by running both algorithms with initial condition $x_0 \neq x^*$ for the same matrix Q and decrypt the outcomes at every iteration. Figure 2 shows that at each iteration the solution gets closer to the optimal x^*.

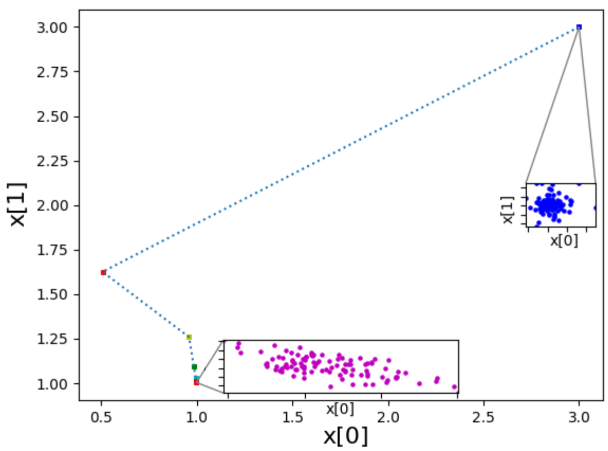

Fig. 2: Decrypted HE-AGD steps for 100 repetitions with a 2-by-2 matrix with $\kappa = 2$, optimal value at $x^* = (1, 1)$ initial condition $x_0 = (3, 3)$. A similar behavior is observed for HE-GD.

As discussed in Section II AGD exhibits a superior convergence rate, hence it allows meeting a given convergence (in terms of optimality) tolerance with fewer iterations. However, in case of encryption, the computational limits imposed by the allowable depth of the encryption circuit introduces a trade-off, as AGD involves more arithmetic operations compared to GD (see Section IV-A). As such, it might be computationally impossible to perform the number of iterations needed by AGD to meet a given tolerance. We investigate this trade-off numerically, and show that the preferred method depends on the condition number κ of the quadratic matrix Q.

To analyze this trade-off numerically we generate QP instances with condition number κ ranging from 1.5 to 50. To collect numerical statistics on the effect of κ, for each κ we generate 100 sets of randomly generated symmetric positive-definite matrices Q of dimension 2, 4 and 8[3], and associated random vectors p. We solve each QP instance via the HE-GD and HE-AGD methods with an initial condition x_0 such that $\|x_0 - x^*\|_2 = 1$. Our goal is to investigate which algorithm achieves better tolerance values (in terms distance to the optimal value) at the last iteration allowed by the encryption's circuit depth. The latter is iteration 6 for AGD and iteration 9 for GD. Figure 3 illustrates the distribution of the tolerance $f(x) - f(x^*)$ (optimality gap of the returned solution x from the optimal cost $f(x^*)$ for AGD (top) and GD (bottom) with matrices of different dimensions (x axis) and different values of the condition number κ (color code).

d \ κ	1.5	2	3	5	10	20	50
2	$3e^{-9}$	$4e^{-9}$	$3e^{-7}$	$5e^{-5}$	$7e^{-3}$	$2e^{-3}$	$5e^{-3}$
4	$1e^{-8}$	$1e^{-8}$	$8e^{-8}$	$1e^{-5}$	$2e^{-4}$	$8e^{-4}$	$2e^{-3}$
8	$6e^{-8}$	$4e^{-8}$	$7e^{-8}$	$5e^{-6}$	$6e^{-5}$	$2e^{-4}$	$9e^{-4}$

TABLE I: Comparison of AGD (6th iteration) against GD (9th iteration) for matrices Q of different sizes and conditional values κ. The numbers represent the median tolerance level $f(x) - f(x^*)$ out of the 100 repetitions.

Table I highlights the important observations stemming from Figure 3. In particular, GD profits from the extra iterations and achieves better tolerance (getting closer to the optimal) values when $\kappa \leq 5$ (upper table). Yet, AGD outperforms GD in cases where $\kappa > 5$ (lower table) although with worse tolerance values (that is, further form the optimal).

The code running the numerical examples presented here (https://github.com/f2cf2e10/agd-he) used our own wapper of the Microsoft SEAL [30] C++ library (https://github.com/f2cf2e10/pSEAL). We used an Intel Xeon E5-1620 with 24GB of RAM machine running Debian 11.

VI. Conclusion

In this paper, we examine gradient and accelerated gradient descent algorithms for solving QP problems in a homomorphic encryption (HE) context. We assess various encryption schemes (BFV, BGV, CKKS) and advocate for CKKS due to its flexibility in selecting step sizes. In our implementation, AGD requires an additional multiplication operation per step compared to GD, limiting its multiplication depth with the same security parameters and channel capacity. We demonstrate that the condition number of the quadratic term's matrix significantly influences algorithm preference: AGD excels with higher condition numbers for faster convergence, while GD outperforms with lower condition numbers due to additional iterations. Furthermore, we propose an efficient HE matrix multiplication algorithm with reduced multiplication depth. Our findings are empirically validated, but challenges persist in HE iterative numerical procedures, especially for constrained problems. Future research should prioritize optimizing multiplication depth and exploring alternative matrix encoding approaches.

References

[1] Andreea B. Alexandru, Konstantinos Gatsis, Yasser Shoukry, Sanjit A. Seshia, Paulo Tabuada, and George J. Pappas. Cloud-based quadratic optimization with partially homomorphic encryption. *IEEE Transactions on Automatic Control*, 66(5):2357–2364, 2021.

[2] Andreea B. Alexandru, Anastasios Tsiamis, and George J. Pappas. Encrypted distributed lasso for sparse data predictive control, 2021.

[3] Zvika Brakerski. Fully homomorphic encryption without modulus switching from classical gapsvp. In Reihaneh Safavi-Naini and Ran Canetti, editors, *Advances in Cryptology – CRYPTO 2012*, pages 868–886, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[4] Zvika Brakerski, Craig Gentry, and Vinod Vaikuntanathan. (leveled) fully homomorphic encryption without bootstrapping. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference*, ITCS '12, page 309–325, New York, NY, USA, 2012. Association for Computing Machinery.

[3]Higher dimensions are also feasible, and this is independent of the multiplication depth limits.

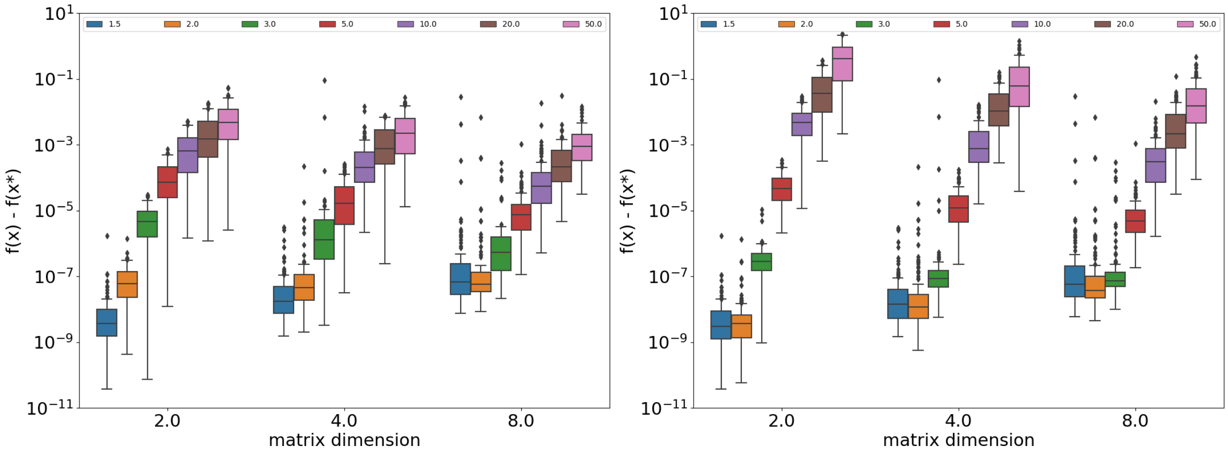

Fig. 3: Box plot of tolerance $f(x) - f(x^*)$ for AGD (left) and GD (right) with matrices of different dimensions (x axis) and different κ (colored bars).

[5] Sébastien Bubeck. Convex optimization: Algorithms and complexity. *Found. Trends Mach. Learn.*, 8(3–4):231–357, November 2015.

[6] Jung Hee Cheon, Kyoohyung Han, Hyuntae Kim, Junsoo Kim, and Hyungbo Shim. Need for controllers having integer coefficients in homomorphically encrypted dynamic system. In *2018 IEEE Conference on Decision and Control (CDC)*, pages 5020–5025, 2018.

[7] Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. Homomorphic encryption for arithmetic of approximate numbers. In Tsuyoshi Takagi and Thomas Peyrin, editors, *Advances in Cryptology – ASIACRYPT 2017*, pages 409–437, Cham, 2017. Springer International Publishing.

[8] Jung Hee Cheon, Dongwoo Kim, Duhyeong Kim, Hun Hee Lee, and Keewoo Lee. Numerical method for comparison on homomorphically encrypted numbers. Cryptology ePrint Archive, Paper 2019/417, 2019. `https://eprint.iacr.org/2019/417`.

[9] T. Elgamal. A public key cryptosystem and a signature scheme based on discrete logarithms. *IEEE Transactions on Information Theory*, 31(4):469–472, 1985.

[10] Junfeng Fan and Frederik Vercauteren. Somewhat practical fully homomorphic encryption. Cryptology ePrint Archive, Report 2012/144, 2012. `https://eprint.iacr.org/2012/144`.

[11] Farhad Farokhi, Iman Shames, and Nathan Batterham. Secure and private cloud-based control using semi-homomorphic encryption. *IFAC-PapersOnLine*, 49(22):163–168, 2016. 6th IFAC Workshop on Distributed Estimation and Control in Networked Systems NECSYS 2016.

[12] Farhad Farokhi, Iman Shames, and Nathan Batterham. Secure and private control using semi-homomorphic encryption. *Control Engineering Practice*, 67:13–20, 2017.

[13] Craig Gentry. Fully homomorphic encryption using ideal lattices. In *Proceedings of the Forty-First Annual ACM Symposium on Theory of Computing*, STOC '09, page 169–178, New York, NY, USA, 2009. Association for Computing Machinery.

[14] Craig Gentry. Computing arbitrary functions of encrypted data. *Commun. ACM*, 53(3):97–105, March 2010.

[15] Craig Gentry, Amit Sahai, and Brent Waters. Homomorphic encryption from learning with errors: Conceptually-simpler, asymptotically-faster, attribute-based. In Ran Canetti and Juan A. Garay, editors, *Advances in Cryptology – CRYPTO 2013*, pages 75–92, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.

[16] Shai Halevi and Victor Shoup. Faster homomorphic linear transformations in helib. Cryptology ePrint Archive, Report 2018/244, 2018. `https://ia.cr/2018/244`.

[17] Ilia Iliashenko and Vincent Zucca. Faster homomorphic comparison operations for bgv and bfv. Cryptology ePrint Archive, Paper 2021/315, 2021. `https://eprint.iacr.org/2021/315`.

[18] Xiaoqian Jiang, Miran Kim, Kristin Lauter, and Yongsoo Song. Secure outsourced matrix computation and application to neural networks. In *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, CCS '18, page 1209–1222, New York, NY, USA, 2018. Association for Computing Machinery.

[19] Andrey Kim, Antonis Papadimitriou, and Yuriy Polyakov. Approximate homomorphic encryption with reduced approximation error. Cryptology ePrint Archive, Report 2020/1118, 2020. `https://ia.cr/2020/1118`.

[20] Junsoo Kim, Dongwoo Kim, Yongsoo Song, Hyungbo Shim, Henrik Sandberg, and Karl H. Johansson. Comparison of encrypted control approaches and tutorial on dynamic systems using learning with errors-based homomorphic encryption. *Annual Reviews in Control*, 54:200–218, 2022.

[21] Junsoo Kim, Chanhwa Lee, Hyungbo Shim, Jung Hee Cheon, Andrey Kim, Miran Kim, and Yongsoo Song. Encrypting controller using fully homomorphic encryption for security of cyber-physical systems. *IFAC-PapersOnLine*, 49(22):175–180, 2016. 6th IFAC Workshop on Distributed Estimation and Control in Networked Systems NECSYS 2016.

[22] Junsoo Kim, Hyungbo Shim, and Kyoohyung Han. Dynamic controller that operates over homomorphically encrypted data for infinite time horizon. *IEEE Transactions on Automatic Control*, 68(2):660–672, 2023.

[23] Shane Kosieradzki, Yingxin Qiu, Kiminao Kogiso, and Jun Ueda. Rewrite rules for automated depth reduction of encrypted control expressions with somewhat homomorphic encryption. In *2022 IEEE/ASME International Conference on Advanced Intelligent Mechatronics (AIM)*, pages 804–809, 2022.

[24] Yongwoo Lee, Joonwoo Lee, Young-Sik Kim, HyungChul Kang, and Jong-Seon No. High-precision and low-complexity approximate homomorphic encryption by error variance minimization. Cryptology ePrint Archive, Report 2020/1549, 2020. `https://ia.cr/2020/1549`.

[25] Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. Cryptology ePrint Archive, Paper 2012/230, 2012. `https://eprint.iacr.org/2012/230`.

[26] Pascal Paillier. Public-key cryptosystems based on composite degree residuosity classes. In Jacques Stern, editor, *Advances in Cryptology — EUROCRYPT '99*, pages 223–238, Berlin, Heidelberg, 1999. Springer Berlin Heidelberg.

[27] Moritz Schulze Darup, Andreea B. Alexandru, Daniel E. Quevedo, and George J. Pappas. Encrypted control for networked systems: An illustrative introduction and current challenges. *IEEE Control Systems Magazine*, 41(3):58–78, 2021.

[28] Moritz Schulze Darup, Adrian Redder, and Daniel E. Quevedo. Encrypted cooperative control based on structured feedback. *IEEE Control Systems Letters*, 3(1):37–42, 2019.

[29] Moritz Schulze Darup, Adrian Redder, Iman Shames, Farhad Farokhi, and Daniel Quevedo. Towards encrypted mpc for linear constrained systems. *IEEE Control Systems Letters*, 2(2):195–200, 2018.

[30] Microsoft SEAL (release 3.7). `https://github.com/Microsoft/SEAL`, September 2021. Microsoft Research, Redmond, WA.