# Improved Inapproximability of Lattice and Coding Problems with Preprocessing

Oded Regev *

**Abstract**

We show that the closest vector problem with preprocessing (CVPP) is NP-hard to approximate to within $\sqrt{3}-\epsilon$ for any $\epsilon > 0$. In addition, we show that the nearest codeword problem with preprocessing (NCPP) is NP-hard to approximate to within $3-\epsilon$. These results improve the results of Feige and Micciancio in [10]. We also present the first inapproximability result for the relatively nearest codeword problem with preprocessing (RNCP). Finally, we describe an $n$-approximation algorithm to CVPP.

## 1 Introduction

An (integer) lattice is the set of all integral linear combinations of a set $\mathbf{B}$ of $n$ linearly independent vectors in $\mathbb{Z}^n$. The set $\mathbf{B}$ is known as a basis of the lattice. The Closest Vector Problem (CVP) is: given a basis $\mathbf{B}$ of a lattice in $\mathbb{Z}^n$ and a target vector $\mathbf{v} \in \mathbb{Z}^n$ find the closest lattice point to $\mathbf{v}$. Equivalently, find a vector $\mathbf{r} \in \mathbb{Z}^n$ such that $\|\mathbf{Br} - \mathbf{v}\|$ is minimized. The norm is often taken to be the Euclidean norm but in this paper we will consider any $l_p$ norm. The CVP is one of the main lattice problems and it has many applications in computer science, especially in coding theory and cryptography (see [14]). The best inapproximability result for CVP is due to Dinur et al. [7] where it is shown that approximating CVP to within almost polynomial factors is NP-hard. The best probabilistic polynomial time approximation algorithm due to Ajtai et al. [1] obtains a $2^{O(n \log\log n/\log n)}$-approximation factor and it uses the deterministic polynomial time $2^{O(n(\log\log n)^2/\log n)}$-approximation algorithm by Schnorr [16].

Our model is motivated by applications of lattices is coding theory and cryptography. There, the encoding or encryption function is usually based on a lattice and the received message is represented by the vector $\mathbf{v}$. Finding the closest vector corresponds to decoding or decrypting the received message. In most cases however, the lattice is fixed and only the received message $\mathbf{v}$ changes among different inputs. This brings us to the following natural question: is the hardness of the CVP caused by the requirement that the algorithm solves the CVP for *any* lattice? Perhaps there exists a good approximation algorithm that first applies preprocessing to the lattice (which is not necessarily computable in polynomial time) and then answers queries of the form $\mathbf{v}$ efficiently. Interestingly, a positive answer is implied by [12]: by using a representation that combines several bases of the so-called Korkin-Zolotarev form, one can obtain a good approximation ratio of $O(n^{1.5})$ to the CVP. Although this representation is not known to be efficiently computable, once we have it we can approximate CVP to within polynomial factors. In this paper we will be interested in

*Institute for Advanced Study, Princeton, NJ. E-Mail: odedr@ias.edu. Research supported by NSF grant CCR-9987845.

showing that there are lattices for which no matter what preprocessing is performed, no polynomial time algorithm can approximate the CVP to within a certain constant factor. This problem is known as the preprocessing variant of CVP, or CVPP. A formal definition is given in Section 2. For further motivation the reader is referred to [10, 14].

The CVP has an analogous problem in codes to which we refer as the Nearest Codeword Problem (NCP). Here, we are given a generating matrix $\mathbf{C} \in \mathbb{F}^{n \times k}$ and a target vector $\mathbf{v} \in \mathbb{F}^n$. We are asked to find the codeword $\mathbf{Cr}$ closest in Hamming distance to the target $\mathbf{v}$. The NCP is also known to be hard to approximate. Namely, Arora et al. [2] show that approximating NCP to within $2^{\log^{(1-\epsilon)} n}$ for any $\epsilon > 0$ is hard under the assumption that $NP \nsubseteq QP$. We also define NCPP as the preprocessing variant of NCP.

Bruck and Naor [6] show that the NCPP problem is NP-hard to solve exactly. This was later extended by Micciancio [13] to the CVPP. However, both results apply only to the exact version of the problems and as noted in [13], it is not clear how to extend them to hardness of approximation. The first inapproximability result is due to Feige and Micciancio [10]. There, it is shown that NCPP over any field $GF(q)$ is NP-hard to approximate within constant factors less than $5/3$ and that as a result CVPP is NP-hard to approximate within constant factors less than $\sqrt{5/3}$. Their result is shown using a reduction from MAX3LIN - the problem of maximizing the number of satisfied equations in a system of linear equations over $GF(q)$ where each equation contains at most 3 variables. This problem was shown to be hard to approximate by Håstad in [11]. Since MAX3LIN becomes easier as $q$ decreases, their result is first shown for large $q$'s and then it is extended to any $q$ by using the technique of concatenating codes.

**Our results.** Our first result is that CVPP is NP-hard to approximate within any constant factor less than $\sqrt{3}$, or, for the $l_p$ norm, within any factor less than $\sqrt[p]{3}$. Actually, we begin by proving that NCPP is NP-hard to approximate within any constant factor less than 3 and then we use a standard reduction from NCP to CVP. Like most other hardness results, our hardness result is shown for the decision version of the problem where we are only asked to approximate the distance of a vector from the lattice and not to actually find a close vector. Our techniques differ from that of [10] in two main respects. First, we directly use the PCP obtained by applying the parallel repetition theorem to the 3SAT5 problem (which are certain regular SAT instances). As mentioned above, the result of [10] is based on a reduction from MAX3LIN. Secondly, our technique directly applies to any field $GF(q)$ and thus we do not need the method of concatenating codes used in [10]. We believe that the factor 3 represents a barrier for current techniques.

As mentioned above, the previous best upper bound for CVPP is $O(n^{1.5})$ and is implied by [12]. Our second result is an improved upper bound of $n$ for the decision problem of CVPP. Interestingly, we use non-constructive bounds on the distance of the closest vector. Therefore, it is not obvious how to extend our result to the search problem, i.e., to find close vectors. Nevertheless, the result is comparable to the hardness result since both were shown for the decision problem.

In the above we considered the preprocessing variant of NCP as a way to find whether the hardness of NCP is a result of the algorithm required to work with *any* linear code. Another reason for the hardness of NCP was raised in the literature ([8]). It might be hard to find the closest word to $\mathbf{v}$ because $\mathbf{v}$ is very *far* from any codeword. In other words, we are trying to recover from too many errors. The Relatively Near Codeword with parameter $\rho > 0$ ($RNC^{(\rho)}$) is defined as follows. Given a generating matrix $\mathbf{C} \in \mathbb{F}^{n \times k}$, a target vector $\mathbf{v} \in \mathbb{F}^n$ and a value $t \in \mathbb{Z}^+$ such that $t$ is less than $\rho$ times the minimum distance in the code generated by $\mathbf{C}$, find a codeword within distance $t$ from $\mathbf{v}$. The algorithm is expected to work only when such a codeword exists. The problem becomes easier as $\rho$ decreases and many classical error correcting algorithms work in polynomial time when $\rho \leq \frac{1}{2}$. For $\rho = \frac{1}{2}$ we are guaranteed to have exactly one codeword within distance $t$ from $\mathbf{v}$. In [8] it is shown that $RNC^{(\rho)}$ for any $\rho > \frac{1}{2}$ is hard to approximate to within any constant factor. In this paper we consider $RNCP^{(\rho)}$, the preprocessing variant of the problem. The problem was mentioned in [10] but the authors were not able to provide any inapproximability result. The $RNCP^{(\rho)}$ is interesting because it asks for an algorithm that decodes a *specific* linear code for a *small* amount of error, hence combining the two reasons mentioned above. In this paper we provide the first inapproximability result for $RNCP^{(\rho)}$ by showing that for any $\rho > \frac{1}{2}$ approximating to within any constant factor less than $3 - \frac{1}{\rho}$ is hard.

## 2 Preliminaries

Vectors are denoted by bold lower case letter and matrices by bold upper case letters. The set $\{1, \ldots, n\}$ is denoted by $[n]$. Throughout the paper, $\mathbb{F}$ denotes the field $GF(q)$ for an arbitrary but fixed prime power $q$. We use the notation $\mathcal{C}_{\mathbf{C}}$ for the code generated by the matrix $\mathbf{C}$. The Hamming distance between two words $\mathbf{v}, \mathbf{w}$ is denoted by $\Delta(\mathbf{v}, \mathbf{w})$ and the distance between a word $\mathbf{v}$ and the code $\mathcal{C}_{\mathbf{C}}$ is denoted by $\Delta(\mathcal{C}_{\mathbf{C}}, \mathbf{v})$. The minimum distance of the code $\mathcal{C}_{\mathbf{C}}$, denoted by $\Delta(\mathcal{C}_{\mathbf{C}})$, is defined as $\min_{\mathbf{x} \neq 0}\{\text{weight}(\mathbf{Cx})\}$.

Our hardness of approximability results will be shown through the corresponding gap problems:

**Definition 2.1 (Closest Vector Problem)** *For $\gamma \geq 1$, an instance of* GapCVP$_\gamma$ *is a triple $(\mathbf{B}, \mathbf{v}, t)$, $\mathbf{B} \in \mathbb{Z}^{n \times k}$, $\mathbf{v} \in \mathbb{Z}^n$ and $t \in \mathbb{R}$. It is a* Yes *instance if there exists $\mathbf{r} \in \mathbb{Z}^k$ such that $\|\mathbf{Br} - \mathbf{v}\| \leq t$. It is a* No *instance if for any $\mathbf{r} \in \mathbb{Z}^k$, $\|\mathbf{Br} - \mathbf{v}\| > \gamma \cdot t$.*

**Definition 2.2 (Nearest Codeword Problem)** *For $\gamma \geq 1$, an instance of* GapNCP$_\gamma$ *is a triple $(\mathbf{C}, \mathbf{v}, t)$, $\mathbf{C} \in \mathbb{F}^{n \times k}$, $\mathbf{v} \in \mathbb{F}^n$ and $t \in \mathbb{Z}^+$. It is a* Yes *instance if $\Delta(\mathcal{C}_{\mathbf{C}}, \mathbf{v}) \leq t$. It is a* No *instance if $\Delta(\mathcal{C}_{\mathbf{C}}, \mathbf{v}) > \gamma \cdot t$.*

**Definition 2.3 (Relatively Near Codeword)** *For $\rho > 0$ and $\gamma \geq 1$, an instance of* GapRNC$_\gamma^{(\rho)}$ *is a triple $(\mathbf{C}, \mathbf{v}, t)$, $\mathbf{C} \in \mathbb{F}^{n \times k}$, $\mathbf{v} \in \mathbb{F}^n$ and $t \in \mathbb{Z}^+$ such that $t < \rho \cdot \Delta(\mathcal{C}_{\mathbf{C}})$. It is a* Yes *instance if $\Delta(\mathcal{C}_{\mathbf{C}}, \mathbf{v}) \leq t$. It is a* No *instance if $\Delta(\mathcal{C}_{\mathbf{C}}, \mathbf{v}) > \gamma \cdot t$.*

Notice that the problem GapRNC$_\gamma^{(\infty)}$ is equivalent to GapNCP$_\gamma$. We define the preprocessing variant of CVP as follows:

**Definition 2.4 (Closest Vector Problem with Preprocessing)** *For $\gamma \geq 1$, we say that a function $P$ solves* GapCVPP$_\gamma$ *if the following is satisfied. The input to the function $P$ is a uniform description of a sequence of lattices of increasing dimension $\{\mathbf{B}_i\}_{i \geq 1}$, i.e., a polynomial time algorithm that on input $1^i$ outputs the lattice $\mathbf{B}_i$. The function $P$ should output a polynomial time algorithm that on input $(1^i, \mathbf{v}, t)$ solves the* GapCVP$_\gamma$ *instance $(\mathbf{B}_i, \mathbf{v}, t)$. There are no computational requirements from $P$.*

The preprocessing variants of the coding problems (GapNCPP$_\gamma$ and GapRNCP$_\gamma$) are defined similarly. We note that a non-uniform definition of a preprocessing problem exists (see [10]). Our results apply to the non-uniform case as well with the only difference being that the assumption $P \neq NP$ is replaced by $NP \not\subseteq P/poly$.

# 3 Inapproximability of Preprocessing Problems

In this section we prove that GapNCPP$_{3-\epsilon}$ is NP-hard for any $\epsilon > 0$. The GapCVPP hardness result follows using a well known reduction. Our proof is based on a reduction from an NP-hard problem (described in Section 3.1) to GapNCP. The reduction has the property that it produces GapNCP$_{3-\epsilon}$ instances $(\mathbf{C}, \mathbf{v}, t)$ where the matrix $\mathbf{C}$ depends only on the size of the input to the reduction. In other words, there exists a universal sequence of codes $\{\mathbf{C}_i\}_{i\geq 1}$ such that the reduction always outputs GapNCP$_{3-\epsilon}$ instances of the form $(\mathbf{C}_i, \mathbf{v}, t)$ where $i$ is the size of the input to the reduction. Now assume that there exists a function $P$ as in Definition 2.4. Then, given the sequence $\{\mathbf{C}_i\}_{i\geq 1}$, the function outputs a polynomial time algorithm that solves the instances given by the reduction and hence solves the original NP-hard problem. Therefore, such a reduction establishes the NP-hardness of GapNCPP$_{3-\epsilon}$.

## 3.1 The PCP

The PCP constructed in this section is obtained by applying the parallel repetition lemma [15] to the 3SAT5 problem [9]. Although the construction is fairly standard by now, we describe it here with special emphasis on the differences among instances of the problem of the same size. The results of this section are summarized in Lemma 3.1.

In a 3SAT5 problem of size $n$ we are given a set $X$ of $\frac{5}{3}n$ clauses over a fixed set of $n$ variables $Y = (y_1, \ldots, y_n)$. Each variable appears in exactly 5 clauses and each clause is the OR of exactly 3 literals. An assignment to the 3SAT5 is a function $A$ from $Y$ to the set $R_Y = \{T, F\}$ and from $X$ to a set $R_X$ of size 7 representing the 7 satisfying assignments of a clause. We define the set $\Phi$ of tests as follows. For each $x \in X$ we construct three tests $\varphi_{x,y}$ - one for each $y$ that appears in $x$. The test $\varphi_{x,y}$ is supposed to check that the assignment given to $x$ agrees with the assignment given to $y$. Formally, it is a function from $R_X$ to $R_Y$ that for each assignment to $x$ gives either true or false based on the restriction to $y$ of the assignment. A test $\varphi_{x,y}$ is *satisfied* by an assignment $A$ if $\varphi_{x,y}(A(x)) = A(y)$. As shown in [9] (based on [3, 4]), it is NP-hard to distinguish between the case where there exists an assignment that satisfies all the tests and the case where no assignment satisfies more than $1-\epsilon$ fraction of the tests for some universal constant $\epsilon$. Notice that an instance of the problem of size $n$ can be specified by the set $X$ of clauses; the set of variables is assumed to be fixed and the set of tests is implied by the set $X$.

Now we apply the parallel repetition lemma [15] with a constant parameter $u$. The resulting problem is the following. We are given a set $X$ of size $(\frac{5}{3}n)^u$ containing $u$-tuples of clauses and a set $Y$ of size $n^u$ containing $u$-tuples of variables. An assignment $A$ is a function from $X$ to a set $R_X$ of size $7^u$ and from $Y$ to a set $R_Y$ of size $2^u$. The set $R_X$ represents the possible assignments to a $u$-tuple of clauses and similarly for $R_Y$. The set of tests $\Phi$ contains $3^u$ tests for each $x \in X$ and $5^u$ tests for each $y \in Y$. Namely, $\Phi$ contains the test $\varphi_{x,y}$ if for every $i \in [u]$ the $i$'th entry of $y$ appears in the $i$'th entry of $x$. As before, the test $\varphi_{x,y} : R_X \to R_Y$ checks the agreement of the assignment to $x$ with the assignment to $y$ by mapping each assignment to $x$ to the corresponding assignment to $y$. A test is satisfied by an assignment $A$ if $\varphi_{x,y}(A(x)) = A(y)$. It is NP-hard to distinguish between the case where there exists an assignment that satisfies all the tests and the case where no assignment satisfies more than $2^{-\Omega(u)}$ fraction of the tests. Notice that we can still specify an instance of the problem of size parameter $n$ by the set $X$ of clauses; the set $Y$ is fixed and the set of tests is implied. Let $\hat{X} = \hat{X}_{n,u}$ be the set of all $u$-tuples of clauses over $n$ variables. Thus, we can say that an instance of the problem of size $n$ is specified by a subset $X \subseteq \hat{X}$.

We summarize the above in the following lemma:

**Lemma 3.1** *For any fixed $\epsilon > 0$ the following problem is NP-hard. Let $R_X, R_Y$ be two sets and $d_X, d_Y$ two integers depending only on $\epsilon$. For any size parameter $n$ there exist two sets of variables*[1] *$Y, \hat{X}$ and a set of tests $\hat{\Phi}$, i.e., functions from $R_X$ to $R_Y$ indexed by variables $x \in \hat{X}$ and $y \in Y$. Each variable $x \in \hat{X}$ has $d_X$ tests in $\hat{\Phi}$ associated with it. For $i \in [d_X]$ we denote by $z_i(x)$ the i'th variable in $Y$ with which $x$ has a test for some ordering of the variables. Then, an instance of the problem of size parameter $n$ is specified by a subset $X \subseteq \hat{X}$. Let $\Phi \subseteq \hat{\Phi}$ be the set of tests associated with $X$. Each variable $y \in Y$ has $d_Y$ tests in $\Phi$ associated to it. An assignment $A$ is a function from $X$ to $R_X$ and from $Y$ to $R_Y$. A test $\varphi_{x,y} \in \Phi$ is satisfied by $A$ if $\varphi_{x,y}(A(x)) = A(y)$. The problem is to distinguish between the case where there exists an assignment that satisfies all the tests $\Phi$ and the case where no assignment satisfies more than $\epsilon$ of the tests.*

## 3.2 The Code

In this section we show the reduction to GapNCPP$_{3-\epsilon}$. Recall that our construction is over the field $\mathbb{F} = GF(q)$ for an arbitrary but fixed prime power $q$.

**Definition 3.2** *An instance of* GapMLD$_\gamma$ *(Maximum Likelihood Decoding) is given by a triple $(\mathbf{C}, \mathbf{v}, t)$, $\mathbf{C} \in \mathbb{F}^{n\times k}$. It is a* Yes *instance if there exists a solution to $\mathbf{Cr} = \mathbf{v}$ whose weight is at most $t$. In a* No *instance the weight of any solution to $\mathbf{Cr} = \mathbf{v}$ is more than $\gamma t$. The preprocessing variant (*GapMLDP$_\gamma$*) is defined similarly.*

The GapMLD problem is a standard formulation and it is essentially equivalent to the GapNCP: a GapMLD instance $(\mathbf{C}, \mathbf{v}, t)$ can be translated into the GapNCP instance $(\mathbf{C}^*, \mathbf{z}, t)$ where $\mathbf{C}^*$ is the orthogonal complement to $\mathbf{C}$ and $\mathbf{z}$ is any vector that satisfies $\mathbf{Cz} = \mathbf{v}$. Any solution $\mathbf{r}$ to the GapNCP can be mapped to a solution to the GapMLD given by the vector $\mathbf{z} - \mathbf{C}^*\mathbf{r}$ since $\mathbf{C}(\mathbf{z} - \mathbf{C}^*\mathbf{r}) = \mathbf{Cz} = \mathbf{v}$. Hence, it is enough to describe a reduction from the PCP of the previous section to GapMLD$_{3-\epsilon}$.

For any given size parameter let $d_Y, d_X, R_X, R_Y, Y, \hat{X}, \hat{\Phi}$ be as described in Lemma 3.1. Let $S$ denote the set $[d_Y]^{d_X}$. For a given PCP instance $X \subseteq \hat{X}$ we define a function $s^* : X \to S$ such that for each variable $y \in Y$ and for each $j \in [d_Y]$ there exists a unique pair $(x, i) \in X \times [d_X]$ such that both $z_i(x) = y$ and $s^*_i(x) = j$. Intuitively, for each $y \in Y$ we can label the $d_Y$ tests that contain $y$ with $1, \ldots, d_Y$ and then define $s^*(x)$ as the $d_X$ labels that the tests that contain $x$ got. More formally, such an $s^*$ exists and can be efficiently computed, say by the following process: for every $y \in Y$ let $\alpha(y)$ be a value initially set to 0. For each value $x \in X$ in an arbitrary order we increment $\alpha(z_i(x))$ by one for each $i \in [d_X]$ and define $s^*(x)$ as $(\alpha(z_1(x)), \ldots, \alpha(z_{d_X}(x)))$. This process ends when $\alpha(y)$ is $d_Y$ for all $y \in Y$.

We describe the set of equations $\mathbf{Cr} = \mathbf{v}$ given by the GapMLDP$_{3-\epsilon}$ instance $(\mathbf{C}, \mathbf{v}, t)$. Let $T$ denote the set $\hat{X} \times [d_X] \times S \times R_X$. The vector $\mathbf{r}$ is $|T|$-dimensional and we index its coordinates by $r_{(x,i,s,a)}$ for $(x, i, s, a) \in T$.

$$\forall x \in \hat{X}, i_1, i_2 \in [d_X], s \in S, a \in R_X \quad r_{(x,i_1,s,a)} = r_{(x,i_2,s,a)} \tag{1}$$

$$\forall y \in Y, j_1, j_2 \in [d_Y], b \in R_Y, \quad \sum_{\substack{(x,i,s,a) \in T \mid \\ (z_i(x), s_i) = (y, j_1) \\ \varphi_{x,y}(a) = b}} r_{(x,i,s,a)} = \sum_{\substack{(x,i,s,a) \in T \mid \\ (z_i(x), s_i) = (y, j_2) \\ \varphi_{x,y}(a) = b}} r_{(x,i,s,a)} \tag{2}$$

[1] From now on, we will refer to elements of $\hat{X}$ and $Y$ simply as variables. The fact that they are $u$-tuples will not be used.

$$\forall x \in X, i \in [d_X] \quad \sum_{a \in R_X} r_{(x,i,s^*(x),a)} = 1 \tag{3}$$

$$\forall x \in X, i \in [d_X], s \neq s^*(x) \quad \sum_{a \in R_X} r_{(x,i,s,a)} = 0 \tag{4}$$

$$\forall x \in \hat{X} \setminus X, i \in [d_X], s \in S \quad \sum_{a \in R_X} r_{(x,i,s,a)} = 0 \tag{5}$$

This set of equations can be used to show the hardness of GapMLDP because it has the property that the coefficients of $\mathbf{r}$ (and hence the matrix $\mathbf{C}$) are independent of the instance $X$; only the target vector $\mathbf{v}$ depends on $X$. To emphasize this point, we now describe the matrix $\mathbf{C}$ and the target vector $\mathbf{v}$ corresponding to the set of equations above. The matrix $\mathbf{C}$ has $|T|$ columns indexed by quadruples $(x, i, s, a) \in T$. For each $x \in \hat{X}$, $i_1, i_2 \in [d_X]$, $i_1 \neq i_2$, $s \in S$ and $a \in R_X$ there is a row denoted by $\alpha(x, i_1, i_2, s, a)$. In addition, for each $y \in Y$, $j_1, j_2 \in [d_Y]$, $j_1 \neq j_2$ and $b \in R_Y$, there is a row denoted by $\beta(y, j_1, j_2, b)$. Finally, there is a row denoted by $\gamma(x, i, s)$ for each $x \in \hat{X}$, $i \in [d_X]$ and $s \in S$. The column $(x, i, s, a)$ is 1 in $\alpha(x, i, i', s, a)$ and -1 in $\alpha(x, i', i, s, a)$ for each $i' \in [d_X]$, $i' \neq i$. It is also 1 in $\beta(z_i(x), s_i, j', \varphi_{x,z_i(x)}(a))$ and -1 in $\beta(z_i(x), j', s_i, \varphi_{x,z_i(x)}(a))$ for each $j' \in [d_Y]$, $j' \neq s_i$. In addition, it is 1 in the coordinate $\gamma(x, i, s)$. It is 0 in all other coordinates. This completes the description of the matrix $\mathbf{C}$. Notice that for any given size parameter it is a fixed matrix. Given an instance $X \subseteq \hat{X}$ we define the function $s^*$ as above and the target vector $\mathbf{v}$ as 1 in $\gamma(x, i, s^*(x))$ for each $(x, i) \in X \times [d_X]$ and 0 in all other coordinates.

**Lemma 3.3 (Completeness)** *If there exists an assignment $A$ that satisfies all the tests in $\Phi$ then there exists a solution $\mathbf{r}$ to $\mathbf{Cr} = \mathbf{v}$ whose weight is $|\Phi|$.*

*Proof:* Given an assignment $A$ let $\mathbf{r}$ be the vector which is 1 in the coordinates $(x, i, s^*(x), A(x))$ for $x \in X$ and $i \in [d_X]$ and 0 elsewhere. The weight of the vector is $d_X \cdot |X| = |\Phi|$. We prove $\mathbf{Cr} = \mathbf{v}$ by showing that Equations 1-5 hold. Equation 1 holds since $r_{(x,i,s,a)}$ is independent of $i$. Equation 3 holds because for $x \in X$ and any $i$ the value of $r_{(x,i,s^*(x),a)}$ is 1 if $a = A(x)$ and 0 otherwise. Equations 4 and 5 hold because $r_{(x,i,s,a)}$ is 0 whenever $x \notin X$ or $s \neq s^*(x)$.

Consider the expression that appears in Equation 2. Since $r_{(x,i,s,a)}$ is nonzero only if $x \in X$ and $s = s^*(x)$, the expression equals:

$$\sum_{\substack{x \in X, i \in [d_X] \mid (z_i(x), s_i^*(x)) = (y,j) \\ a \in R_X \mid \varphi_{x,y}(a) = b}} r_{(x,i,s^*(x),a)}.$$

Also, by the choice of $s^*$ there is a unique pair $(x, i) \in X \times [d_X]$ such that $(z_i(x), s_i^*(x)) = (y, j)$. Let $(x', i')$ denote this pair. Then the expression above equals

$$\sum_{a \in R_X \mid \varphi_{x',y}(a) = b} r_{(x',i',s^*(x'),a)}$$

which is 1 if $\varphi_{x',y}(A(x')) = b$ and 0 otherwise. Since $A$ is a satisfying assignment $\varphi_{x',y}(A(x')) = A(y)$ and therefore the expression above is 1 if $b = A(y)$ and 0 otherwise. In particular, it is independent of $j$, as required by Equation 2. ∎

**Lemma 3.4 (Soundness)** *For any $\epsilon > 0$, if there exists a solution $\mathbf{r}$ to $\mathbf{Cr} = \mathbf{v}$ whose weight is less than $(3 - \epsilon)|\Phi|$ then there exists an assignment that satisfies at least a $\frac{1}{6}\epsilon$ fraction of the tests.*

*Proof*: Let $\mathbf{r}$ be a vector of weight less than $(3-\epsilon)|\Phi|$ such that $\mathbf{Cr} = \mathbf{v}$. Consider the following assignment. To each $x \in X$ we assign a value $a \in R_X$ chosen uniformly at random from the $a$'s for which $r_{(x,i,s^*(x),a)} \neq 0$ where $i \in [d_X]$ is arbitrary. According to Equation 1 this set is independent of the choice of $i$ and according to Equation 3 this set is not empty. To a variable $y \in Y$ we uniformly choose at random a value $b \in R_Y$ such that

$$\sum_{\substack{(x,i,s,a) \in T \mid (z_i(x), s_i) = (y,j) \\ \varphi_{x,y}(a) = b}} r_{(x,i,s,a)} \neq 0$$

where $j \in [d_Y]$ is arbitrary. Notice that according to Equation 2 this set is independent of the choice of $j$. This set is also not empty because by summing over all $b \in R_Y$ we get

$$\sum_{b \in R_Y} \sum_{\substack{(x,i,s,a) \in T \mid (z_i(x), s_i) = (y,j) \\ \varphi_{x,y}(a) = b}} r_{(x,i,s,a)} = \sum_{(x,i,s,a) \in T \mid (z_i(x),s_i)=(y,j)} r_{(x,i,s,a)} =$$
$$\sum_{x \in \hat{X}, i \in [d_X], s \in S \mid (z_i(x),s_i)=(y,j)} \sum_{a \in R_X} r_{(x,i,s,a)} = 1$$

where the last equality is by Equations 3, 4 and 5. We claim that the expected number of satisfied tests is at least $\frac{1}{6}\epsilon|\Phi|$ and therefore there exists an assignment that satisfies at least a $\frac{1}{6}\epsilon$ fraction of the tests.

Associate each quadruple $(x,i,s,a) \in T$ for which $r_{(x,i,s,a)}$ is nonzero with the pair $(z_i(x), s_i) \in Y \times [d_Y]$. Since the weight of $\mathbf{r}$ is at most $(3-\epsilon)|\Phi|$, there exist at most $(1-\frac{1}{3}\epsilon)|\Phi|$ pairs with which we associate at least 3 quadruples. Therefore, there exist $\frac{1}{3}\epsilon|\Phi|$ pairs with which we associate 2 quadruples or less. Let $(y',j')$ be such a pair and let $(x',i')$ be the unique pair in $X \times [d_X]$ such that $(z_{i'}(x'), s^*_{i'}(x')) = (y',j')$ (it is unique by the choice of $s^*$). Quadruples associated with $(y',j')$ can be of the form $(x',i',s^*(x'),a)$ for some $a \in R_X$. We call such quadruples 'good'. According to Equation 3 there must be at least one good quadruple associated with $(y',j')$. Other quadruples associated with $(y',j')$ are called 'bad'. They can either be of the form $(x,i,s,a)$ for $x \in X$ and $s \neq s^*(x)$ or of the form $(x,i,s,a)$ for $x \in \hat{X} \setminus X$. According to Equations 4 and 5, if a bad quadruple $(x,i,s,a)$ is associated with $(y',j')$ then there must exist another bad quadruple of the form $(x,i,s,a')$ for some other $a' \neq a$ that is also associated with $(y',j')$ (because $r_{(x,i,s,a)} \neq 0$). Since we already have at least one good quadruple associated with $(y',j')$ and the total number is at most 2, it must be the case that no bad quadruples are associated with $(y',j')$.

We map the pair $(y',j')$ to the unique test in $\varphi' \in \Phi$ that is associated with $x'$ and $y'$ where $x'$ is defined above. Notice that this is a one-to-one and onto mapping from $Y \times [d_Y]$ to $\Phi$. Thus it is enough to show that the assignment $A$ satisfies $\varphi'$ with probability at least $\frac{1}{2}$. The assignment given to $x'$ is an $a \in R_X$ for which $r_{(x',i',s^*(x'),a)} \neq 0$. Since there are at most two good vectors associated with $(y',j')$ there are at most two such values. Denote them by $a_1, a_2 \in R_X$ where possibly $a_1 = a_2$. The assignment given to $y'$ is chosen from the set of $b \in R_Y$ for which

$$\sum_{\substack{(x,i,s,a) \in T \mid (z_i(x), s_i) = (y',j') \\ \varphi_{x,y'}(a) = b}} r_{(x,i,s,a)} \neq 0.$$

Since no bad vectors are associated with $(y',j')$, this is equivalent to

$$\sum_{a \in R_X \mid \varphi_{x',y'}(a)=b} r_{(x',i',s^*(x'),a)} \neq 0.$$

The proof is completed by noting that the above expression is non-zero if and only if $b = \varphi_{x',y'}(a_1)$ or $b = \varphi_{x',y'}(a_2)$. ∎

**Theorem 3.5** *For any $\epsilon > 0$,* $\textsc{GapMLDP}_{3-\epsilon}$ *is NP-hard.*

*Proof:* The proof follows from Lemmas 3.3 and 3.4 and the PCP of Lemma 3.1 with parameter $\frac{1}{6}\epsilon$. ∎

The $\textsc{GapNCPP}$ hardness result follows from the equivalence between $\textsc{GapMLD}$ and $\textsc{GapNCP}$ described in the beginning of this section.

**Theorem 3.6** *For any $\epsilon > 0$,* $\textsc{GapNCPP}_{3-\epsilon}$ *is NP-hard.*

Using a well known and simple reduction from $\textsc{GapNCPP}$ over $GF(2)$ to $\textsc{GapCVPP}$ (see, e.g., Theorem 6 in [10]) we obtain the following:

**Theorem 3.7** *For any $p \geq 1, \gamma < \sqrt[p]{3}$,* $\textsc{GapCVPP}_\gamma$ *in the $l_p$ norm is NP-hard.*

# 4 Inapproximability of the Relatively Near Codeword with Preprocessing

Theorem 16 in [8] shows a reduction from $\textsc{GapNCP}$ to $\textsc{GapRNC}$. We would like to prove a similar reduction for the preprocessing variants of the problems. Actually, as already noted in [10], one can apply Theorem 16 of [8] directly to the preprocessing variants of the problems. This, however, requires hardness of $\textsc{GapNCPP}_\gamma$ for $\gamma > 2$ which was not previously known. So, by using Theorem 3.6, one can directly obtain that $\textsc{GapRNCP}_\gamma^{(\rho)}$ for any $\rho > 1$ and $\gamma < 1.5$ is hard unless $NP = RP$. Based on the results of [8], we present an improved reduction that allows us to show hardness of $\textsc{GapRNCP}_\gamma^{(\rho)}$ for any $\rho > \frac{1}{2}$ and $\gamma < 3 - \frac{1}{\rho}$.

First, we quote the following lemma. We denote by $\mathcal{B}(\mathbf{v}, r)$ the Hamming ball of radius $r$ around $\mathbf{v}$.

**Lemma 4.1 (Lemma 15, [8])** *For any $\epsilon \in (0, 1)$ there exists a probabilistic polynomial time algorithm that on input $k, s$ outputs in $poly(k, s)$-time integers $l, m, r$, matrices $\mathbf{A} \in \mathbb{F}^{l \times m}$ and $\mathbf{T} \in \mathbb{F}^{k \times l}$ and a vector $\mathbf{w} \in \mathbb{F}^l$ such that*

- *The weight of $\mathbf{v}$ is $r$,*
- *The minimum distance in $\mathcal{C}_\mathbf{A}$ is greater than $2(1-\epsilon)r$,*
- $\mathbf{T}(\mathcal{B}(\mathbf{w}, r) \cap \mathcal{C}_A) = \mathbb{F}^k$ *with probability at least $1 - 2^{-\Omega(s)}$.*

**Claim 4.2** *The output of the algorithm in Lemma 4.1 satisfies that there is no codeword in $\mathcal{C}_\mathbf{A}$ whose distance to $\mathbf{v}$ is less than $(1-2\epsilon)r$.*

*Proof:* The claim follows since the weight of $\mathbf{v}$ is $r$ and the minimum distance in $\mathcal{C}_\mathbf{A}$ is more than $(2-2\epsilon)r$. ∎

Our proof follows the lines of Theorem 16 in [8] with the following two differences. Our construction uses a more careful choice of parameters $a, b$ and in addition, the analysis of $\textsc{No}$ instances is improved by using Claim 4.2. For completeness, we describe the modified proof:

**Theorem 4.3** *For any $\rho > \frac{1}{2}$ and arbitrarily small $\epsilon > 0$ there exists a polynomial RUR-reduction from* $\text{GAPNCP}_\gamma$ *to* $\text{GAPRNC}^{\rho}_{\gamma'}$ *where $\gamma' = (1 - \frac{1}{2\rho})\gamma + \frac{1}{2\rho} - \epsilon$. The same reduction is also a reduction from* $\text{GAPNCPP}_\gamma$ *to* $\text{GAPRNCP}^{\rho}_{\gamma'}$*.*

*Proof:* Let $(\mathbf{C}, \mathbf{v}, t)$ be an instance of $\text{GAPNCP}_\gamma$ where $\mathbf{C} \in \mathbb{F}^{n \times k}$. We construct an instance $(\mathbf{C}', \mathbf{v}', t')$ of $\text{GAPRNC}^{(\rho)}_{\gamma'}$ as follows. We apply the algorithm of Lemma 4.1 with a parameter $\epsilon' > 0$ to be specified later. Let $a, b$ be two integers such that $\frac{r}{t}(2\rho(1 - 2\epsilon') - 1) \leq \frac{b}{a} \leq \frac{r}{t}(2\rho(1 - \epsilon') - 1)$. Notice that we can choose $a$ and $b$ to be polynomial in $r$ and $t$ and hence polynomial in the size of the input. We define $\mathbf{C}'$ by placing $a$ copies of $\mathbf{A}$ and $b$ copies of $\mathbf{CTA}$. Similarly, $\mathbf{v}'$ contains $a$ copies of $\mathbf{w}$ and $b$ copies of $\mathbf{v}$:

$$\mathbf{C}' = \begin{bmatrix} \mathbf{A} \\ \vdots \\ \mathbf{A} \\ \mathbf{CTA} \\ \vdots \\ \mathbf{CTA} \end{bmatrix} \qquad \mathbf{v}' = \begin{bmatrix} \mathbf{w} \\ \vdots \\ \mathbf{w} \\ \mathbf{v} \\ \vdots \\ \mathbf{v} \end{bmatrix}$$

Finally, we define $t'$ as $ar + bt$. The minimum distance of the code $\mathbf{C}'$ is

$$\Delta(\mathcal{C}_{\mathbf{C}'}) \geq a\Delta(\mathcal{C}_A) > a \cdot 2(1 - \epsilon')r$$

Now $t'$ satisfies

$$t' = ar + bt = ar(1 + \frac{b}{a} \cdot \frac{t}{r}) \leq ar(1 + (2\rho(1 - \epsilon') - 1)) = \rho \cdot a \cdot 2(1 - \epsilon')r < \rho\Delta(\mathcal{C}_{\mathbf{C}'})$$

and therefore $(\mathbf{C}', \mathbf{v}', t')$ is indeed a valid $\text{GAPRNC}^{(\rho)}$ instance.

In case $(\mathbf{C}, \mathbf{v}, t)$ is a YES instance, there exists an $\mathbf{x}$ such that $\Delta(\mathbf{Cx}, \mathbf{v}) \leq t$. Let $\mathbf{z}$ be such that $\Delta(\mathbf{Az}, \mathbf{w}) \leq r$ and $\mathbf{TAz} = \mathbf{x}$. Such a $\mathbf{z}$ exists with probability exponentially close to 1 according to the second property in Lemma 4.1. Hence,

$$\Delta(\mathbf{C}'\mathbf{z}, \mathbf{v}') = a \cdot \Delta(\mathbf{Az}, \mathbf{w}) + b \cdot \Delta(\mathbf{CTAz}, \mathbf{v}) \leq a \cdot r + b \cdot t = t'.$$

Assume that $(\mathbf{C}, \mathbf{v}, t)$ is a NO instance, i.e., the distance of $\mathbf{v}$ from $\mathcal{C}_{\mathbf{C}}$ is greater than $\gamma t$. For all $\mathbf{x} \in \mathbb{F}^m$ we have

$$\Delta(\mathbf{C}'\mathbf{x}, \mathbf{v}') \geq a \cdot (1 - 2\epsilon')r + b \cdot \Delta(\mathbf{CTAx}, \mathbf{v}) > a \cdot (1 - 2\epsilon')r + b \cdot \gamma t$$

where the first inequality is due to Claim 4.2. We claim that this number is at least $\gamma' t' = ((1 - \frac{1}{2\rho})\gamma + \frac{1}{2\rho} - \epsilon)t'$. Indeed,

$$\frac{a \cdot (1 - 2\epsilon')r + b \cdot \gamma t}{t'} = \frac{1 - 2\epsilon' + \frac{b}{a}\frac{t}{r}\gamma}{1 + \frac{b}{a}\frac{t}{r}} \geq \frac{1 - 2\epsilon' + (2\rho(1 - 2\epsilon') - 1)\gamma}{1 + (2\rho(1 - \epsilon') - 1)}$$

which can be made arbitrarily close to $(1 - \frac{1}{2\rho})\gamma + \frac{1}{2\rho}$ by choosing a small enough $\epsilon'$.

Notice that the reduction also applies to the preprocessing variants of the problems because the RNC code depends only on the NCP code. ■

**Corollary 4.4** *For any $\rho > \frac{1}{2}$,* $\text{GAPRNCP}^{(\rho)}_\gamma$ *with $\gamma < 3 - \frac{1}{\rho}$ is not in RP unless NP = RP.*

## 5 Upper Bound

In this section we describe a solution to GAPCVPP$_n$ in the Euclidean norm. For $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$, let $d(\mathbf{x}, \mathbf{y})$ denote the Euclidean distance between the two vectors. Also, let $d(\mathbf{x}, A)$ denote the $\min_{\mathbf{y} \in A} d(\mathbf{x}, \mathbf{y})$. The dual of a lattice $L$ is a lattice $L^*$ with the same linear span such that for every $\mathbf{u} \in L$ and $\mathbf{v} \in L^*$, the inner product $\langle \mathbf{u}, \mathbf{v} \rangle$ is integer. We will use the following transference theorem due to Banaszczyk:

**Lemma 5.1 ([5], Theorem 2.2)** *For any lattice $L$ and any $\mathbf{x} \in \mathbb{R}^n$, $d(\mathbf{x}, L) \le \frac{1}{2\lambda_1(L^*)} n$ where $\lambda_1(L^*)$ denotes the length of the shortest nonzero vector in the dual lattice $L^*$.*

Given a lattice $L$ let $L_1 = L$ and let $\mathbf{v}_1$ be the shortest vector of $L_1^*$. Also, let $\mathbf{u}_1$ be any vector in $L$ such that $\langle \mathbf{u}_1, \mathbf{v}_1 \rangle = 1$ and $L_2$ be the $n-1$ dimensional lattice given by $L_1 \cap \{\mathbf{v}_1\}^\dagger$. Notice that $L_1$ is a union of translations of $L_2$ by integer multiples of $\mathbf{u}_1$. Moreover, the orthogonal distance between the linear span of two adjacent translations is $1/\|\mathbf{v}_1\|$. In general, we define $\mathbf{v}_i$ for $i \le n$ as the shortest vector of $L_i^*$ and $\mathbf{u}_i \in L_i$ such that $\langle \mathbf{u}_i, \mathbf{v}_i \rangle = 1$. We let $L_{i+1}$ be $L_i \cap \{\mathbf{v}_i\}^\dagger$. The preprocessing function produces the vectors $\mathbf{v}_1, \ldots, \mathbf{v}_n$ and $\mathbf{u}_1, \ldots, \mathbf{u}_n$ (although it is easy to compute $\mathbf{u}_i$ given $\mathbf{v}_i$). We now describe a procedure $C(i, \mathbf{w})$ where $\mathbf{w}$ is in the linear span of $L_i$ that returns a number which is between $d(\mathbf{w}, L_i)$ and $n \cdot d(\mathbf{w}, L_i)$. Clearly, $C(1, \mathbf{w})$ solves GAPCVPP$_n$, as required. If $i = n$ then we can find $d(\mathbf{w}, L_n)$ exactly because this is a one dimensional lattice. Otherwise, fix $\mathbf{w}$ in the linear span of $L_i$. As mentioned above, $L_i$ is a union of translations of $L_{i+1}$ by multiples of $\mathbf{u}_i$ with an orthogonal distance of $1/\|\mathbf{v}_i\|$ between two adjacent translations. Let $k$ be the integer closest to $\langle \mathbf{w}, \mathbf{v}_i \rangle$. Then $\mathbf{w}' := \mathbf{w} - (\langle \mathbf{w}, \mathbf{v}_i \rangle - k)\mathbf{v}_i / \|\mathbf{v}_i\|^2$ is the orthogonal projection of $w$ on the closest translation of the linear span of $L_{i+1}$. The procedure returns the value $\min(\frac{1}{2\|\mathbf{v}_i\|} n, \sqrt{(d(\mathbf{w}, \mathbf{w}'))^2 + (C(i+1, \mathbf{w}' - k\mathbf{u}_i))^2})$.

We prove the correctness of the procedure by induction. It is obviously correct for $i = n$. Assume that the procedure is correct for $i+1, \ldots, n$. First we show that $C(i, \mathbf{w}) \le n \cdot d(\mathbf{w}, L_i)$. The closest point in $L_i$ to $\mathbf{w}$ can be on $L_{i+1} + k \cdot \mathbf{u}_i$ in which case,

$$d(\mathbf{w}, L_i) = \sqrt{(d(\mathbf{w}, \mathbf{w}'))^2 + (d(\mathbf{w}', L_{i+1} + k \cdot \mathbf{u}_i)^2} = \sqrt{(d(\mathbf{w}, \mathbf{w}'))^2 + (d(\mathbf{w}' - k \cdot \mathbf{u}_i, L_{i+1})^2}$$

$$\ge \sqrt{(d(\mathbf{w}, \mathbf{w}'))^2 + (\frac{1}{n} C(i+1, \mathbf{w}' - k \cdot \mathbf{u}_i))^2} \ge \frac{1}{n} C(i, \mathbf{w}).$$

Otherwise, the closest point to $\mathbf{w}$ is on a different translation of $L_{i+1}$ and

$$d(\mathbf{w}, L_i) \ge \frac{1}{2\|v_i\|} \ge \frac{1}{n} C(i, \mathbf{w}).$$

It remains to show that $C(i, \mathbf{w}) \ge d(\mathbf{w}, L_i)$. According to Lemma 5.1,

$$d(\mathbf{w}, L_i) \le \frac{1}{2\|\mathbf{v}_i\|} \cdot n.$$

Also,

$$d(\mathbf{w}, L_i) \le \sqrt{(d(\mathbf{w}, \mathbf{w}'))^2 + (d(\mathbf{w}', L_{i+1} + k \cdot \mathbf{u}_i)^2} \le \sqrt{(d(\mathbf{w}, \mathbf{w}'))^2 + (C(i+1, \mathbf{w}' - k\mathbf{u}_i))^2}$$

which completes the proof.

## 6 Acknowledgements

I would like to thank Daniele Micciancio for presenting me the problem during a visit to UCSD, for helpful discussions and for many comments on an earlier version of this paper.

## References

[1] M. Ajtai, R. Kumar, and D. Sivakumar. A sieve algorithm for the shortest lattice vector problem. In *Proc. 33rd ACM Symp. on Theory of Computing*, pages 601–610, 2001.

[2] S. Arora, L. Babai, J. Stern, and Z. Sweedyk. The hardness of approximate optima in lattices, codes, and systems of linear equations. *J. Comput. System Sci.*, 54(2, part 2):317–331, 1997. 34th Annual Symposium on Foundations of Computer Science (Palo Alto, CA, 1993).

[3] S. Arora, C. Lund, R. Motwani, M. Sudan, and M. Szegedy. Proof verification and the hardness of approximation problems. *J. ACM*, 45(3):501–555, 1998.

[4] S. Arora and S. Safra. Probabilistic checking of proofs: a new characterization of NP. *J. ACM*, 45(1):70–122, 1998.

[5] W. Banaszczyk. New bounds in some transference theorems in the geometry of numbers. *Mathematische Annalen*, 296(4):625–635, 1993.

[6] J. Bruck and M. Naor. The hardness of decoding linear codes with preprocessing. *IEEE Trans. Inform. Theory*, 36(2):381–385, 1990.

[7] I. Dinur, G. Kindler, and S. Safra. Approximating-CVP to within almost-polynomial factors is NP-hard. In *Proc. 39th IEEE Symp. on Foundations of Computer Science*, 1998.

[8] I. Dumer, D. Micciancio, and M. Sudan. Hardness of approximating the minimum distance of a linear code. In *Proc. 40th IEEE Symp. on Foundations of Computer Science*, pages 475–484, 1999.

[9] U. Feige. A threshold of ln n for approximating set cover. *JACM: Journal of the ACM*, 45, 1998.

[10] U. Feige and D. Micciancio. The inapproximability of lattice and coding problems with preprocessing. In *Computational Complexity*, pages 44–52, 2002.

[11] J. Håstad. Some optimal inapproximability results. In *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, pages 1–10, May 1997.

[12] J. C. Lagarias, H. W. Lenstra, Jr., and C.-P. Schnorr. Korkin-Zolotarev bases and successive minima of a lattice and its reciprocal lattice. *Combinatorica*, 10(4):333–348, 1990.

[13] D. Micciancio. The hardness of the closest vector problem with preprocessing. *IEEE Transactions on Information Theory*, 47(3):1212–1215, 2001.

[14] D. Micciancio and S. Goldwasser. *Complexity of Lattice Problems: a cryptographic perspective*, volume 671 of *The Kluwer International Series in Engineering and Computer Science*. Kluwer Academic Publishers, Boston, Massachusetts, March 2002.

[15] R. Raz. A parallel repetition theorem. *SIAM Journal on Computing*, 27(3):763–803, June 1998.

[16] C.-P. Schnorr. A hierarchy of polynomial time lattice basis reduction algorithms. *Theoretical Computer Science*, 53(2-3):201–224, 1987.