Complex Semidefinite Programming Revisited and the Assembly of Circular Genomes

Konstantin Makarychev[1] Alantha Newman[2]
[1]IBM T. J. Watson Research Center, Yorktown Heights, NY
[2]DIMACS, Rutgers University, New Brunswick, NJ
konstantin@us.ibm.com alantha@dimacs.rutgers.edu

Abstract: We consider the problem of arranging elements on a circle so as to approximately preserve specified pairwise distances. This problem is closely related to optimization problems found in genome assembly. The current methods for genome sequencing involve cutting the genome into many segments, sequencing each (short) segment, and then reassembling the segments to determine the original sequence. A useful paradigm has been using "mate pair" information, which, for a circular genome (e.g. bacterial genomes), generates information about the directed distance between non-adjacent pairs of segments in the final sequence.
Specifically, given a set of equations of the form $x_v - y_u \equiv d_{uv} \pmod{q}$, we study the objective of maximizing a linear payoff function that depends on how close the value $x_v - y_u \pmod{q}$ is to d_{uv}. We apply the rounding procedure used by Goemans and Williamson for "complex" semidefinite programs. Our main tool is a simple geometric lemma that allows us to easily compute the expected distance on a circle between two elements whose positions have been computed using this rounding procedure.

Keywords: semidefinite programming, genome assembly, circular arrangement, linear equations.

1 Introduction

We consider the problem of arranging elements on a circle subject to directed pairwise distance constraints. For example, consider the well-studied problem Linear Equations mod q. In this problem, we are given a set of equations of the form $x_v - x_u \equiv d_{uv} \pmod{q}$. The standard objective is to assign each element x_u an integral value in the range $[0, q)$ so as to satisfy the maximum number of constraints. Due to the circular symmetry, this problem can also be seen as arranging elements on a circle (i.e. on positions labeled zero through $q-1$ in the clockwise direction) so as to exactly satisfy the maximum number of specified directed pairwise distance constraints.

A natural relaxation of this problem is to try to preserve these distances *as much as possible.* In other words, if we have the equation $x_v - x_u \equiv 4 \pmod{16}$, we may prefer assignments to x_v and x_u such that $x_v - x_u$ equals three or five, rather than two or six. Given that x_u has some assignment, then the *target position* for x_v is the value of x_u plus d_{uv}. For instance, in the previous example, if x_u is assigned zero, then the target position for x_v would be four. We would like x_v to be as close to this target position as possible. Current algorithms for Linear Equations mod q (e.g. algorithms for Unique Games [2]) are aimed at satisfying equations exactly and say nothing about the quality of the solutions for the unsatisfied equations.

Although this problem does not appear to have been addressed from a theoretical perspective—beyond the standard formulation of Linear Equations mod q as explained above—the general problem of approximately preserving directed pairwise distances is closely related to optimization problems used for genome assembly, in particular to a problem known as Contig Scaffolding [9], upon which we now elaborate.

1.1 Genome assembly and Contig Scaffolding

Genome sequencing is an area of research into which tremendous amounts of time, money and computational resources are currently being invested. For our purposes, a genome can be viewed as two oppositely-oriented strings from a four letter alphabet, $\{A, C, G, T\}$. Each of the two strings is the complement of the other (A pairs with T, and C with G).

Moreover, each string (and each of its substrings) is directed; e.g. it can be viewed as $A \to A \to T \to C \to \ldots$. If we determine to which of the two strings a particular substring belongs, then we can determine the orientation of that substring. Genomes range in length from thousands to billions of letters, also known as *base pairs*. While some genomes, such as those of humans, are linear strings, a large class of genomes are circular. For example, the genomes of all bacteria are circular.

With the current technology for genome sequencing, an entire genome cannot be sequenced at once. Rather, comparatively short substrings are sequenced, and these short substrings must then be assembled to form the original genome. To obtain more information to enable this assembly, many copies of the original two strings of the genome are made. These copies are broken up randomly, the pieces sequenced and then reassembled based on the local overlap information gleaned from the many copies. This is a computationally intensive task, and the overlap information is sometimes insufficient to determine the sequence if, for example, there are repeated substrings in the original sequence.

An important innovation in genome assembly was to use so-called "mate pair" information [9]. Suppose we are able to sequence substrings of length ℓ. We consider a substring S of length $L >> \ell$ and sequence the two substrings of length ℓ that make up the two ends of S. Now we have two substrings whose relative distance and orientation in the original genome is known. This global information was crucial for sequencing and dealing with repeated substrings in the human genome [11].

The graph theoretic approach outlined by Huson et al. [9] is based on aggregated mate pair information: based on local overlap information, substrings are combined into longer substrings called *contigs*. If a mate pair (i.e. two substrings with known relative orientation and distance) belong to two different contigs, then we have information about the relative orientation and distance of these contigs. Of course, due to sequencing errors as well as repeated substrings, this information may be inconsistent, but there are methods for averaging this distance and orientation information. Ultimately, we obtain what Huson et al. refer to as the *contig-mate-pair-graph*. In this graph, contigs are represented by vertices, and some pairs of contigs have desired distances and relative orientations associated with them. The problem of Contig Scaffolding is to assign each contig an orientation (i.e. assign each contig to one of the two complementary strings in the genome) and a position so that the relative position of specified pairs of contigs that are assigned to the same string is approximately preserved. Specifically, suppose we have a pair of contigs u and v that are known to be at a distance L based on mate pair information, i.e. they are connected by a directed edge e of length L in the contig-mate-pair-graph. Note that both u and v have complementary contigs—call them $\bar{u}$ and $\bar{v}$, respectively—which should have the opposite orientation as u and v. Following the example in Section 3 of [9], either u and v have the same orientation, say clockwise (i.e. they are assigned to the string with clockwise direction), and $|pos(v) - pos(u) - L| \leqslant \sigma(e)$, or u and v are assigned to the string with counterclockwise direction and $|pos(u) - pos(v) - L| \leqslant \sigma(e)$. If u and v fulfill either of these situations, the pair or edge u, v is called "happy". The goal is to maximize the number (or weight) or the happy edges. In [9], $\sigma(e)$ denotes a function of the standard deviation of the distribution from which the length of the edge e was generated.

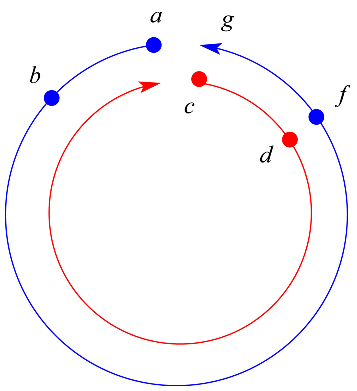

Figure 1: In this length L substring of a genome, the segments $a \to b$ and $c \to d$ are "mate pairs". Since the segment $f \to g$ is the complement of $d \leftarrow c$, we know both segments $a \to b$ and $f \to g$ after sequencing this mate pair. We also know that in the reassembled genome, if $a \to b$ and $f \to g$ are assigned to the clockwise string, then $f \to g$ should be approximately distance L ahead of $a \to b$ in the clockwise direction. If they are on the counterclockwise string, then $f \to g$ should be approximately distance L ahead of $a \to b$ in the counterclockwise direction.

In general, this problem has seen many greedy/local approaches [9]. As discussed in [13], most of these approaches iterate through the constraints on the pairs of contigs, attempting to optimally place and orient them. Constraints that are violated by the current position of the contigs are typically simply discarded. Recently, Dayarian et al. used a global approach based on simulated annealing to satisfy the distance constraints between pairs of contigs [3]. Another approach is based on Eulerian tours in specially constructed graphs [12].

1.2 Relaxed linear equations modulo q

We now discuss the precise formulation of the problem that we address in this paper and how it relates to the problem of Contig Scaffolding. Given an equation of the form $x_v - x_u \equiv d_{uv} (\bmod\ q)$, we define the following *payoff* function:

$$P_{\{u,v\}}(i,j) = 1 - \frac{2\ell}{q}, \tag{1}$$
$$\text{if } \ (j-i) \equiv d_{uv} \pm \ell \pmod q,$$
$$\text{where } \ \ell \in [0, q/2].$$

In words, $P_{\{u,v\}}(i,j)$ is the cost of simultaneously placing element x_u at position i and element x_v at position j. In each equation, if we assume that x_u is assigned some value, then there is a *target position* for x_v. In particular, if $x_u = 0$, then the target position for x_v is d_{uv}. There is a linear decrease in the contribution of an equation to the objective value the farther element x_v is from its target position with respect to the position of x_u.

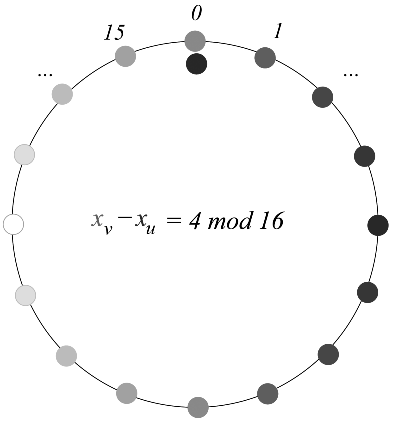

Figure 2: If x_u is in position 0, then the darkest red dot denotes the target position for x_v and the lighter the dots are, the less ideal the position is for x_v.

We refer to the payoff function (1) as the Relaxed Linear Equations mod q problem (or Rel-Lin-Eq(q) for short). Also, we allow each equation to be weighted by a positive value, w_{uv}. We note that if we randomly assign each variable x_u a value in $[0, q)$, we would obtain a solution in which each equation contributes half to the objective value in expectation. Our main theoretical result is that given a set of equations in the form $x_v - x_u \equiv d_{uv} \pmod q$, we can efficiently assign each element x_u a value in $[0, q)$ such that for each equation, the payoff function, (1), is satisfied to within at least .854 times the payoff of that equation in an optimal solution.

1.3 Applying Rel-Lin-Eq(q) to Contig Scaffolding

If we had just one of the strings in a double-stranded circular genome, and we knew the relative distance of certain pairs of contigs, then the problem of Contig Scaffolding would be very similar to an approximate version of the Linear Equations mod q problem. However, since there are actually two oppositely oriented strings, applying Rel-Lin-Eq(q) to the Contig Scaffolding problem is not entirely straightforward. We now clarify the differences between the problems and discuss how our problem can be used as a tool to find good arrangements of the contigs in the Contig Scaffolding problem.

Rather than simply finding positions for the contigs, we must also determine their relative orientation, or to which of the two strings they belong. We discuss two ways to reduce the Contig Scaffolding problem to one very similar to the Rel-Lin-Eq(q) problem. The first approach is given by Dayarian et al [3]. In this approach, we find a maximum cut on the graph composed of edges that represent constraints for contigs that are oppositely oriented, since these contigs should be on different strings. Once we have found a partition of the contigs, we can order each set of contigs separately. For each set in the partition, we only need to consider constraints between pairs of contigs specifying that they should have the same orientation (a constraint between two oppositely oriented contigs can be replaced by one in which one of the contigs is substituted with its complement). We can then use our algorithm to find a circular arrangement for each set of contigs.

The second approach that we propose is to skip the step which uses a maximum cut to partition the contigs into sets, and try to arrange the contigs so that all pairwise directions are oriented clockwise. If we only consider pairwise constraints between contigs of the same orientation (i.e. between contigs believed to be on the same string), and if there were no errors in our data and we could find an arrangement that agreed with all of the pairwise distances, then the contigs would be partitioned into two disjoint strings. (In other words, the counterclockwise string would also be translated to the clockwise direction.) Of course, there may be noise in the data, and many of the pairwise distance constraints may be violated in the arrangement of the contigs that we find. However, our goal is to take advantage of techniques that try to globally satisfy the constraints. After finding such an arrangement, extracting actual feasible solutions from our arrangements will likely have to be conducted heuristically, e.g. throw away the most violated constraints.

Additionally, there are two more key issues that are not directly addressed when optimizing the payoff function in the Rel-Lin-Eq(q) problem. The first issue is that the payoff function (1) rewards each constraint to the extent that it is satisfied, whereas the stated goal in the Contig Scaffolding problem is to maximize the number (or weight) of equations within some fixed distance from the target position. We note that these two objectives are related. Suppose we have a solution for the objective function in (1) that has value $(1-\epsilon)|E|$, where $|E|$ is the number of constraints. Let δ, σ be values in $[0,1]$.

Lemma 1 *In a solution with value $(1-\epsilon)|E|$, at most $\delta|E|$ equations are more than distance σq from the target, where $\delta \leqslant \epsilon/(2\sigma)$.*

We note that the desired distances from the target, $\sigma(e)$, may actually be different for different constraints. A way to address this with our payoff function, (1), could be to weight the constraint corresponding to edge e more if $\sigma(e)$ is smaller.

The second issue is that in a feasible solution to the Contig Scaffolding problem, each position on the circle should be occupied by at most two contigs (one for the clockwise circle and one for the counterclockwise circle). We note that our solution to the problem posed in (1) does allow that more than one or two elements be assigned to a certain position. Since our solution consists of rounding a semidefinite programming (SDP) relaxation, we could add spreading constraints as in [5], so as to ensure that not too many elements are assigned to the same position. However, from a computational perspective, these constraints are extremely expensive to add, since each constraint is an inequality, which forces us to add a new vector to the SDP relaxation for each constraint. Thus, if an arrangement does map several contigs to the same position, then this issue will have to be addressed heuristically. A suggestion in [3] is that if they obtain a solution in which many contigs are assigned the same position, they go back and add more constraints to the contig graph based on local overlap information between pairs of these contigs. Moreover, we note that while our algorithm should generate useful global information about where the contigs should be placed relative to each other, as in many assembly algorithms, it would likely need work in the "finishing" stage to generate an actual feasible solution.

Finally, we mention that L—the distance between two ends of a mate pair—does not have to be the same length for each pair. In practice, there are usually a small set of possible values for L. Some algorithms for assembly based on mate pair information require that L is a small set of values. In contrast, in our algorithm, the desired directed distance between the contigs in a constraint pair (represented by d_{uv} in the respective equation) can take on any value between between zero and q, where q will represent the length of a string. This could be useful for future technologies if mate pair distances are generated from a broader range.

1.4 Complex semidefinite programming

Complex semidefinite programming (CSDP) was introduced by Goemans and Williamson to study the Max-3-Cut and Linear Equations mod 3 problems [8]. They represent each element with a complex vector $re^{-i\theta}$, which is equivalent to representing each element with an infinite set of real vectors, each with length r and corresponding to some angle $\theta \in [0, 2\pi)$. Throughout this paper, we refer to such a set of vectors as a two-dimensional disc. In their rounding algorithm, they choose a normally distributed random vector g and project the vector g onto each disc. The disc can be viewed as being partitioned into three equal sections, each of angle 120 degrees. Depending on which section of its disc the vector g projects onto determines if the element is assigned a 0, 1 or 2. Although the applications in the original paper are for problems with domains of size three, it is interesting to note that the actual positions are denoted by angles which are continuous in the range $[0, 2\pi)$. Thus, it seems likely that this technique may have applications for larger domain problems in which the goal is to place the elements on the circle so as to optimize a specified objective function. The main tool in [8] is that, if elements are represented by two-dimensional discs, and the rounding algorithm is as described above, they give a formula for the distribution of the angle between the positions of two elements, specifically, for the probability that the distance is less than a particular angle. Zhang and Huang gave a formula for the probability that two elements are an exact angle apart [15].

Despite the elegance of this approach, this technique does not appear to have been applied to other optimization problems. One barrier is that these two-dimensional discs have limited modeling power, i.e. while we can model Linear Equations mod 3 exactly with a complex semidefinite program, it is not clear how to model Linear Equations mod q for $q > 3$ with these two-dimensional discs. (By "model" a problem, we mean write an integer program for the problem for which a solution of value α corresponds to an actual integer solution to the problem with value α.)

Our main tool to address the problem presented in Section 1.2 is a simple geometric method for computing the *expected* angle between two elements if they are assigned positions on a circle using the algorithm from [8]. In contrast, as mentioned above, Goemans and Williamson give a formula for the probability that the angle after rounding is less than a certain angle. Theirs is clearly a stronger theorem, but our proof is quite simple. We believe that our proof might yield some geometric insight into CSDP, which could possibly promote what appears to be a useful but overlooked technique. We also note that although we do not know how to model the Rel-Lin-Eq(q) problem exactly using the CSDP methods from [8] (i.e. we cannot represent the elements with two-dimensional discs and obtain a relaxation of our problem in the usual sense), we nevertheless show how to use techniques from CSDP in our rounding methods.

1.5 Preliminaries and notation

The payoff function in (1) uses the following natural notion of distance between two points on a circle. Suppose a circle has q equally spaced points that are labeled clockwise from 0 through $q-1$ for some integer q. We note that the following definition of distance between two points on a circle is sometimes referred to as the *Lee distance.*

Definition 1 *Given two points on a circle with indices $x, y \in [0, q)$, let $\operatorname{dist}(x, y) = \min\{|x-y|,\ q-|x-y|\}$, that is, the length of the minor arc between x and y.*

In general, for a specified domain of size q, a *distance* on that domain will be a number between 0 and $q/2$ and a *normalized distance* will be a fraction between 0 and 1, i.e. the distance on domain q divided by $q/2$.

1.5.1 Notation

Let $E = \{x_v - x_u \equiv d_{uv} (\bmod\ q)\}$ be a given system of linear equations on the set of elements $V = \{x_u\}$. Let $n = |V|$. Let $[q]$ denote the integers in the range $[0, q)$.

1.6 Our results and organization

In Section 2 we present our main tool, which is a simple, geometric lemma to compute the expected angle between two elements in the rounding algorithm for CSDP given in [8]. In Section 3, we show how to apply this lemma to a relaxation of a quadratic program for the Rel-Lin-Eq(q) problem to obtain a .8046-approximation. This SDP relaxation contains only two vectors per element, and thus can be solved efficiently in practice for moderate values of $|E|$, e.g. $|E| \approx 100$. Next, in Section 3, we show that using an SDP relaxation that is standard in the CSP framework, we can obtain an improved approximation factor of .854, again using the same geometric tool from Section 2 to round the relaxation. Since an α-approximation algorithm for the Rel-Lin-Eq(q) problem implies an α-approximation for Max-Cut, we note that assuming the Unique Games conjecture, our approximation is within at least .0245 of optimal. Finally, we note that both algorithms yield an approximation guarantee of $(1-O(\sqrt{\epsilon}))|E|$ when the optimal solution has value at least $(1-\epsilon)|E|$. We also give an improved approximation guarantee for Linear Equations mod 4 in Section 5. All of our results hold for sets of constraints with non-negative weights.

2 A geometric tool

Suppose we have two two-dimensional discs, A and B, such that disc A is in the plane defined by two orthogonal vectors A_x and A_y and disc B is in the plane defined by two orthogonal vectors B_x and B_y. Let q be any (possibly huge) positive integer. For simplicity, we assume that q is a multiple of four. For each $i \in [q]$, we define the following vector:

$$A_i = \cos\left(\frac{2\pi \cdot i}{q}\right)A_y + \sin\left(\frac{2\pi \cdot i}{q}\right)A_x. \tag{2}$$

Then $\{A_i\}$ is the set of q vectors that comprise the two-dimensional disc A. In other words, $A_0 = A_y$ and $A_{q/4} = A_x$. We can define a set of vectors for the disc B similarly. We say that $A = \{A_i\} = (A_x, A_y)$. More generally, suppose we have the following properties relating the four vectors A_x, A_y, B_x and B_y:

(i) $A_x \cdot A_y = 0$ and $B_x \cdot B_y = 0$,

(ii) $|A_x| = |A_y| = |B_x| = |B_y|$,

(iii) $A_x \cdot B_y = A_y \cdot (-B_x)$,

(iv) $A_x \cdot B_x = A_y \cdot B_y$.

If properties (i) through (iv) hold, then we can show that an additional useful property holds.

Lemma 2 *Let $A = (A_x, A_y)$ and $B = (B_x, B_y)$ be discs that satisfy the above properties (i) through (iv). Then the angle between A_i and B_j equals the angle between A_{i+k} and B_{j+k} for all $i, j, k \in [q]$, where subscripts are computed modulo q.*

Proof: Let $\alpha = \frac{2\pi \cdot i}{q}$ and $\beta = \frac{2\pi \cdot j}{q}$. Let $\alpha_k = \frac{2\pi \cdot (i+k)}{q}$ and $\beta_k = \frac{2\pi \cdot (j+k)}{q}$, with subscripts computed modulo q. Then we have:

$$A_i \cdot B_j = \cos(\alpha - \beta)(A_y \cdot B_y) + \sin(\beta - \alpha)(A_y \cdot B_x).$$

We have used the facts that $A_x \cdot B_x = A_y \cdot B_y$ and that $A_y \cdot B_x = -A_x \cdot B_y$. Since it is the case that $\alpha - \beta = \alpha_k - \beta_k$ and $\beta - \alpha = \beta_k - \alpha_k$, we have shown that $A_i \cdot B_j = A_{i+k} \cdot B_{j+k}$. Additionally, if property (ii) holds, then it is straightforward to see that $|A_h| = |A_x|$ for all $h \in [q]$, and similarly for all vectors B_h. Thus, the lemma follows. □

Now suppose the two discs A and B represent two elements x_a and x_b that we want to place on a circle with following goal in mind: if element a is in position i, we want element b to be close to position $i + c_{ab}$. In other words, our "target" is to satisfy the equation $x_b - x_a \equiv c_{ab} \pmod{q}$. Is there a procedure that places x_a in a position i and places x_b in a position "close" to position $i + c_{ab}$ for all $i \in [q]$?

Input: A set of n elements, $\{x_a\}$, and a corresponding set of n two-dimensional discs, $\{A\}$ such that:
(a) Each disc A is defined by two specified orthogonal vectors, A_x and A_y in $\Re^{2n}$, and vector A_i is defined as in Equation (2).
(b) Each pair of discs obeys constraints (i)–(iv).
Output: An assigned position for each element in the range $[q]$.

Rounding Procedure:
1. Choose $g \in \mathcal{N}(0,1)^{2n}$.
2. For each disc, A, compute the set of values $\{A_i \cdot g\}$.
3. Let $pos(x_a) = i$ if $A_{i-1} \cdot g < 0$ and $A_i \cdot g > 0$.

We note that the above rounding procedure is equivalent to that presented by Goemans and Williamson for CSDP [8]. Since the disc A is two-dimensional, $pos(x_a)$ is uniquely defined. This is because the projection of g onto the disc A partitions the vectors in the disc into two sets—those that have positive dot products with g and those that have negative dot products—and each set contains consecutive indices. We now want to analyze the expected distance between $pos(x_a)$ and $pos(x_b)$ for two discs, A and B. Using the definition of $dist(x,y)$ from Definition 1, we have:

Lemma 3 *Let $A = (A_x, A_y)$ and $B = (B_x, B_y)$ be two discs that satisfy properties (i) through (iv). Suppose* $\arccos(A_0 \cdot B_0) = \theta$. *Then:* $\mathfrak{E}\Big[dist\big(pos(x_a), pos(x_b)\big)\Big] = \frac{\theta}{2\pi} \cdot q$.

Proof: Consider the pairs of vectors $\{A_k, B_k\}$ for integral $k \in [q]$. By Lemma 2, we have that for all k:

$$A_0 \cdot B_0 = A_k \cdot B_k.$$

Thus, for each pair of vectors, $\{A_k, B_k\}$, the angle between the vectors is also θ. By [7], the probability that this pair of vectors differs in sign is:

$$\Pr[sign(g \cdot A_0) \neq sign(g \cdot B_0)] \ = \ \frac{\theta}{\pi}.$$

This probability holds for all pairs, since all pairs have the same angle. Thus, in expectation, there are $\theta \cdot q/\pi$ pairs with different signs. Since a pair of dot products $g \cdot A_i$, $g \cdot B_i$ differ in sign exactly when the pair of dot products $g \cdot A_{i+q/2}$, $g \cdot B_{i+q/2}$ differs in sign (superscripts computed modulo q), then the expected distance between the position $pos(x_a)$ and $pos(x_b)$ is exactly $\theta \cdot q/(2\pi)$. □

More generally, if we assume without loss of generality that $pos(x_a) = 0$, then we can compute the expected distance of $pos(x_b)$ from its "target" position j.

Lemma 4 *Let $A = (A_x, A_y)$ and $B = (B_x, B_y)$ be two discs that satisfy properties (i) through (iv). Sup-*

pose $\arccos(A_0 \cdot B_j) = \theta_j$. *Then:* $E\Big[dist\big(pos(x_a) + j, pos(x_b)\big)\Big] = \frac{\theta_j}{2\pi} \cdot q$.

Proof. The proof is analogous to the proof of Lemma 3, except in this case we consider pairs of vectors $\{A_i, B_{i+j}\}$. □

3 Applying our techniques to Rel-Lin-Eq(q)

We now show how to apply the geometric tools from Section 2 to the problem of Rel-Lin-Eq(q). We consider two semidefinite programming relaxations. The first relaxation uses $2n$ vectors (two vectors per element), thus making the relaxation tractable for relatively large values of q (e.g. ≈ 100 on a desktop). We show that using this relaxation, we can obtain a .8046-approximation for the Rel-Lin-Eq(q) problem. In Section 4, we consider another relaxation, from which we can obtain an approximation guarantee of .854. The latter relaxation is in the standard CSP framework and therefore has q vectors per vertex, making it tractable only for very small values of q (e.g. ≈ 15).

In the following quadratic program, each element $x_u \in V$ corresponds to two vectors that associate this element with a two-dimensional plane defined by the vectors y_u and $y_u^\perp$. There are q elements, $\{y_u^i\}$ for $i \in [q]$, in the disc associated with element x_u.

$$y_u^i = \cos\left(\frac{2\pi \cdot i}{q}\right) y_u + \sin\left(\frac{2\pi \cdot i}{q}\right) y_u^\perp. \tag{3}$$

For the equation $x_v - x_u \equiv d_{uv} (\text{mod } q)$, we have the following objective function:

$$\frac{1 + y_u \cdot y_v^{d_{uv}}}{2} = \sum_{uv \in E} \frac{1 + \cos\left(\frac{2\pi \cdot d_{uv}}{q}\right)(y_u \cdot y_v) + \sin\left(\frac{2\pi \cdot d_{uv}}{q}\right)(y_u \cdot y_v^\perp)}{2}.$$

Thus, we can write the above objective function using only two vectors per element.

Quadratic Program (Q_1):

$$\max \sum_{uv \in E} \frac{1 + y_u \cdot y_v^{d_{uv}}}{2}$$

$$y_u \cdot y_u = 1, \ \forall x_u \in V, \tag{4}$$

$$y_u^\perp \cdot y_u^\perp = 1, \quad \forall x_u \in V, \tag{5}$$

$$y_u \cdot y_u^\perp = 0, \quad \forall x_u \in V, \tag{6}$$

$$y_u \cdot y_v = y_u^\perp \cdot y_v^\perp, \quad \forall x_u, x_v \in V, \tag{7}$$

$$y_u \cdot y_v^\perp = -y_u^\perp \cdot y_v, \quad \forall x_u, x_v \in V, \tag{8}$$

$$y_u, y_u^\perp \in \Re^2, \quad \forall x_u \in V. \tag{9}$$

A semidefinite relaxation can be obtained by replacing (9) with the constraint $y_u, y_u^\perp \in \Re^{2n}$, $\forall x_u \in V$. We refer to this relaxation as (Q_1'). Note that the objective value of this quadratic program—and thus that of the corresponding relaxation—might not be an upper bound on the value of an optimal solution. This is because on the interval $[\pi/2 \leqslant \phi \leqslant \pi]$, $\cos\phi$ is a lower bound to ϕ/π. Nevertheless, as we now show, we can derive an upper bound on the value of an optimal solution to the Rel-Lin-Eq(q) problem given an optimal solution to (Q_1'). Our main theorem of this section is that applying the rounding procedure from Section 2 to the two-dimensional discs obtained from (Q_1') results in the following guarantee:

Theorem 1 *Rounding the relaxation* (Q_1') *is a factor* .8046 *approximation for* Rel-Lin-Eq(q).

Consider a set of vectors $\{y_u, y_u^\perp\}$ obtained from an optimal solution to (Q_1'). Let $\theta_{uv} = \arccos(y_u^0 \cdot y_v^{d_{uv}})$ and $\theta_{uv} \in [0, \pi]$ (i.e. corresponding to the equation $x_v - x_u \equiv d_{uv} (\text{mod } q)$). Define γ as follows:

$$\gamma = \sum_{uv \in E} \frac{1}{|E|} \cos(\theta_{uv}). \tag{10}$$

We can assume that $\gamma \in [0, 1]$. Consider some arbitrarily large integer N and define $\theta_i = \frac{2\pi i}{N}$. Define $\theta_{min}(\gamma)$ to be:

$$\theta_{min}(\gamma) = \min_{\{a_i\}} \sum_{i=0}^{N/2} a_i \theta_i \tag{11}$$

$$\text{subject to:} \quad 0 \leqslant a_i \ \leqslant \ 1, \tag{12}$$

$$\sum_{i=1}^{N/2} a_i = 1, \tag{13}$$

$$\sum_{i=0}^{N/2} a_i \cos(\theta_i) = \gamma. \tag{14}$$

Define $\theta_{max}(\gamma)$ analogously.

Lemma 5 *Given an optimal solution for* (Q_1') *where* $\gamma = \sum_{uv \in E} \cos(\theta_{uv})/|E|$, *we can upper bound the value of an optimal solution:* $OPT \leqslant (1 - \theta_{min}(\gamma))|E|$.

Proof. Suppose there were a solution with value more than $(1-\theta_{min}(\gamma))|E|$. Then, there would be a solution to (Q_1') with value $(1+\gamma')|E|/2$ such that $\gamma' > \gamma$. □

Lemma 6 *For* $\gamma \in [\cos(\theta_{aw}), 1)$*,* $\theta_{min} = \frac{(1-\gamma)}{1-\cos(\theta_{aw})}$.

Proof: Consider $\gamma \in [\cos(\theta_{aw}), 1)$. Begin with any set of values $\{a_i\}$ that fulfill constraints (12), (13) and (14) above. Since $\cos\phi$ is concave in the range $[0 \leqslant \phi \leqslant \pi/2]$ and convex on the range $[\pi/2 \leqslant \phi \leqslant \pi]$, we can replace the values of a_i with non-negative values a_i' such that $a_i' = 0$ for $0 < i < N/4$ and there is only one value of $j \in [N/4, N/2]$ such that $a_j' \neq 0$. For these values a_i', the following statements hold:

$$\begin{aligned}\gamma &= \sum_{i=1}^{N/2} a_i \cos(\theta_i)\\ &= 1 - a_j' + a_j' \cos(\theta_j),\\ \sum_{i=1}^{N/2} a_i\theta_i &\geqslant a_j'\theta_j.\end{aligned}$$

In other words, we have $\gamma = 1 - x + x\cos\theta$ for some $x > 0$ and some value of θ. We want to determine the value of $\theta > 0$ that minimizes the value $x \cdot \theta$. We have:

$$g(\theta) = x \cdot \theta \;=\; \frac{(1-\gamma)\cdot\theta}{1-\cos(\theta)} \tag{15}$$

$$g'(\theta) = \frac{(1-\gamma)(1-\cos\theta-\theta\sin\theta)}{(1-\cos\theta)^2} \tag{16}$$

Note that $\theta_{aw} \approx 133.56°$ is the value that minimizes (15) and it follows that θ_{aw} is also the value for which (16) equals zero. Thus, for $\gamma \in [\cos(\theta_{aw}), 1)$, we have:

$$\theta_{min}(\gamma) = \frac{(1-\gamma)\theta_{aw}}{1-\cos(\theta_{aw})} \tag{17}$$

□

Lemma 7 *For* $\gamma \in (-1, \cos(\pi - \theta_{aw})]$*,* $\theta_{max} = \frac{(1-\gamma)\theta_{aw}}{1-\cos(\theta_{aw})}$*. For* $\gamma \in [\cos(\pi-\theta_{aw}), 1]$*,* $\theta_{max} = \arccos\gamma$.

Proof: First, we consider $\gamma \in (-1, \cos(\pi-\theta_{aw})]$. Begin with any set of values $\{a_i\}$ that fulfill constraints (12), (13) and (14). Since $\cos\phi$ is concave in the range $[0 \leqslant \phi \leqslant \pi/2]$ and convex on the range $[\pi/2 \leqslant \phi \leqslant \pi]$, we can replace the values of a_i with non-negative values a_i' such that $a_i' = 0$ for $N/4 < i < N/2$ and there is only one value of $j \in [0, N/4]$ such that $a_j' \neq 0$. For these values a_i', the following statements hold:

$$\begin{aligned}\gamma &= \sum_{i=1}^{N/2} a_i \cos(\theta_i)\\ &= -1(1 - a_j') + a_j' \cos(\theta_j),\\ \sum_{i=1}^{N/2} a_i\theta_i &\leqslant a_j'\theta_j.\end{aligned}$$

In other words, we have $\gamma = x - 1 + x\cos\theta$ for some $x > 0$ and some value of θ. We want to determine the value of θ that maximizes the value $x \cdot \theta$. We have:

$$\begin{aligned}g(\theta) &= x\cdot\theta + (1-x)\pi\\ &= \frac{(1+\gamma)\theta + (\cos\theta - \gamma)\pi}{1+\cos\theta}.\end{aligned}$$

We want to find the value of θ that minimizes $g(\theta)$. Taking the derivative, we have:

$$g'(\theta) = \frac{\pi(1+\gamma)(\cos\theta + 1 + (\theta-\pi)\sin\theta)}{(1+\cos\theta)^2}.$$

We are looking for $\theta \in [0, \pi/2]$. Thus, $g'(\theta)$ is zero when the following holds:

$$\begin{aligned}0 &= \cos\theta + 1 + (\theta-\pi)\sin\theta\\ &= 1 - \cos(\pi-\theta) - (\pi-\theta)\sin(\pi-\theta).\end{aligned}$$

Since this is the same expression as (16), we see that it is satisfied when $\pi - \theta = \theta_{aw}$, which implies that $\theta = \pi - \theta_{aw}$. Thus, for $\gamma \in (-1, \cos(\pi-\theta_{aw})]$, we have:

$$\begin{aligned}\theta_{max}(\gamma) &= \frac{(1+\gamma)(\pi-\theta_{aw}) + (\cos(\pi-\theta_{aw}) - \gamma)\pi}{1+\cos(\pi-\theta_{aw})}\\ &= \frac{\pi - (1+\gamma)\theta_{aw} - \pi\cos\theta_{aw}}{1-\cos(\theta_{aw})}.\end{aligned}$$

Now we consider $\gamma \in (\cos(\pi-\theta_{aw}), 1]$. Let $\theta_\gamma = \arccos\gamma$. On the interval $\theta \in [0 \leqslant \theta \leqslant \theta_\gamma]$, the function $g(\theta)$ is an increasing function of θ. So the function is maximized when $\theta = \theta_\gamma$. Thus, for $\gamma \in (\cos(\pi-\theta_{aw}), 1]$, we have $\theta_{max} = \arccos\gamma$. □

Now we can prove Theorem 1:

Proof of Theorem 1: Given a solution to (Q_1'), where $\gamma = \sum_{uv\in E}\cos(\theta_{uv})/|E|$, the approximation ratio achieved by the rounding procedure is:

$$\frac{\pi - \min\{\theta_{max}(\gamma), \pi/2\}}{\pi - \theta_{min}(\gamma)}.$$

since the numerator is a lower bound on what our rounding procedure yields, and the denominator is an upper bound on the value of an optimal solution.

In the interval $\gamma \in [-1, \frac{\pi(1-\cos(\theta_{aw}))}{2\theta_{aw}} - 1]$, the approximation ratio is at least:

$$\frac{\frac{\pi}{2}(1-\cos(\theta_{aw}))}{\pi(1-\cos(\theta_{aw})) - (1-\gamma)\theta_{aw}} \geqslant .8046.$$

In the interval $\gamma \in [\frac{\pi(1-\cos(\theta_{aw}))}{2\theta_{aw}} - 1, \cos(\pi - \theta_{aw})]$, the approximation ratio is at least:

$$\frac{(1+\gamma)\theta_{aw}}{\pi(1-\cos(\theta_{aw})) - (1-\gamma)\theta_{aw}} \geqslant .8046.$$

Finally, in the interval $\gamma \in [\cos(\pi - \theta_{aw}), 1]$, the approximation ratio is at least:

$$\frac{\pi - \arccos\gamma}{\pi - \frac{(1-\gamma)\theta_{aw}}{1-\cos(\theta_{aw})}} \geqslant .8593.$$

□

We conclude this section by analyzing the asymptotic guarantee of our algorithm when the optimal value of an instance of Rel-Lin-Eq(q) is $(1-\epsilon)|E|$ for small ϵ.

Theorem 2 *If an optimal solution to an instance of* Rel-Lin-Eq*(q) has value $(1-\epsilon)|E|$, then our algorithm finds a solution with value at least $\big(1 - O(\sqrt{\epsilon})\big)|E|$.*

Proof: Suppose ϵ is very close to zero. Consider the ratio in (18). We have:

$$1 - \frac{(1-\gamma)\theta_{GW}}{\pi(1-\cos(\theta_{aw}))} \geqslant 1 - \epsilon.$$

This implies that: $\gamma \geqslant 1 - pi\epsilon/\beta$ for some constant $\beta > 0$. The value obtained by the algorithm is $1 - \theta_\gamma/\pi$. Since we have $\gamma = \cos(\theta_\gamma) \geqslant 1 - \pi\epsilon/\beta$, it follows that $\cos^2(\theta_\gamma) \geqslant 1 - 2\pi\epsilon/\beta$. Thus, for small values of θ_γ, we have $\theta_\gamma \leqslant \sqrt{2\pi\epsilon/\beta}$. So the value obtained by our algorithm is at least $1 - O(\sqrt{\epsilon})$. □

4 An improved approximation ratio for Rel-Lin-Eq(q)

We now consider another relaxation for which we can also use the rounding procedure described in Section 2. This standard relaxation is the one recently analyzed by Raghavendra for CSP problems [14]. Each element is represented by an orthogonal constellation rather than a two-dimensional disc. The first step in our approach is therefore to find a solution for the relaxation of the following quadratic program. Then, the first step in our rounding procedure is to create a two-dimensional disc for each element using the vectors we obtain from this solution. We then apply the rounding procedure from Section 2 to obtain positions for each element in the range $[0, 2\pi)$. Thus, despite the fact, that unlike in the output of relaxation (Q_1'), we do not directly obtain two-dimensional discs from the SDP, we can still use our geometric tool from Section 2. We therefore demonstrate a general connection between the standard relaxation for CSPs and the CSDP framework. Our analysis yields a .854-approximation, which is better than the approximation guarantee given in Section 3.

Quadratic Program (Q_2):

$$\max \sum_{uv\in E} \left(\sum_{i,j\in[q]} P_{\{u,v\}}(i,j) u_i \cdot v_j \right)$$

$$u_i \cdot v_j \geqslant 0,\ u_i \cdot u_j = 0 \quad \forall x_u, x_v \in V,\ i,j \in [q], \tag{18}$$

$$\sum_{i\in[q]} |u_i|^2 = 1 \quad \forall x_u \in V, \tag{19}$$

$$|\sum_{i\in[q]} u_i - \sum_{j\in[q]} u_j|^2 = 0 \quad \forall x_u, x_v \in V, \tag{20}$$

$$u_i \in \{0,1\} \quad \forall x_u \in V,\ i \in [q]. \tag{21}$$

We consider the semidefinite relaxation in which constraint (21) is replaced by the constraint $u_i \in \Re^{qn}$ for all $x_u \in V, i \in [q]$. We refer to this relaxation as (Q_2'). In the relaxation (Q_2'), each element $x_u \in V$ corresponds to q orthogonal *assignment* vectors, $\{u_0, u_1, \ldots u_{q-1}\}$. In an integral solution, for each element $x_u \in V$, only one vector u_i (for a single value of i) is allowed to be a unit vector. The index of this vector corresponds to the position to which this element x_u is assigned in this solution. The relaxation (Q_2') is (exactly) the same as SDP(II) from [14] (although there are also other ways to write it) and has been used to obtain approximation algorithms for Max-Dicut [6], Linear Equations mod q [1] and Unique Games [2,10]. We note that we are using the payoff function given in (1). However, if, for example, we wanted to more accurately model the Contig-Scaffold problem, we could modify the payoff function so that for a particular equation corresponding to edge e, all positions more than $\sigma(e)$ away from the target position have payoff zero. It would be more difficult to analyze what our rounding procedure gives with this payoff function.

If the size of the domain, q, is a fixed integer, then for any payoff function, Raghavendra gives an algorithm that has an optimal approximation guarantee assuming the Unique Games Conjecture [14]. Moreover, he shows that an integrality gap of α for the problem corresponding to this payoff function implies a Unique-Games hardness factor of α, and shows a rounding scheme whose approximation ratio is arbitrarily close to the integrality gap. However, there is a shortcoming to these results in terms of efficiency: both the rounding algorithm and the computation of the approximation ratio require time that is exponential in both the domain size and the inverse of the accuracy parameter, making them impractical to compute for a given payoff function. Moreover, note that for our problem, q need not be a fixed integer. Thus, in the case that q is not a fixed integer, Raghavendra's algorithm does not guarantee an optimal algorithm even assuming the Unique Games Conjecture.

Note that our particular payoff function (1) is *shift invariant*. In other words, the payoff function only depends on the relative positions of i and j. Because of this circular symmetry, for each solution with a certain value, there are actually q solutions with the same value. We can therefore augment the relaxation (Q'_2) with the following constraints:

$$\boxed{\begin{aligned} (*) \quad & |u_i|^2 = 1/q, \quad \forall x_u \in V,\ i \in [q], \\ & u_i \cdot v_j = u_{i+k} \cdot v_{j+k}, \\ & \qquad\qquad \forall x_u, x_v \in V,\ i, j, k \in [q]. \end{aligned}}$$

We refer to the relaxation (Q'_2) augmented with the constraints $(*)$ as (Q^*_2). Note that the constraints $(*)$ are also valid (for the same reasons) in the standard Linear Equations mod q problem. We will also use the following lemma that holds for a feasible solution to the relaxation (Q^*_2).

Lemma 8 *For every $u, v \in V$, it is the case that $\sum_{j=0}^{q} u_0 \cdot v_j = u_0 \cdot u_0$.*

Proof: Note that for each $u \in V$, the sum of elements equals the same unit vector, call this vector x. Since the sum of the lengths is 1 and the vectors are pairwise orthogonal, it must be the case that $u_0 \cdot u_0 = u_0 \cdot x$. In other words, we have:

$$\sum_{j=0}^{q} u_0 \cdot v_j = u_0 \cdot \sum_{j=0}^{q} v_j = u_0 \cdot x = u_0 \cdot u_0.$$

The lemma follows. □

4.1 Geometric rounding of (Q^*_2)

Let $\{u_i\}$ be a feasible solution to (Q^*_2) corresponding to a given instance of Rel-Lin-Eq(q). The main idea behind our improved algorithm is to create the following two-dimensional disc for each element $x_u \in V$.

$$U_x = \sum_{i=0}^{q-1} \sin\left(\frac{2\pi \cdot i}{q}\right) u_i \tag{22}$$

$$U_y = \sum_{i=0}^{q-1} \cos\left(\frac{2\pi \cdot i}{q}\right) u_i. \tag{23}$$

We can show that the following holds for the vectors $\{U_x, U_y\}$.

Lemma 9 *The vectors $\{U_x, U_y\}$ for $x_u \in V$ satisfy properties (i) through (iv).*

(Since the proof of Lemma 9 is somewhat long, it can be found in Appendix B.) We can therefore apply the rounding procedure from Section 2.

In particular, we can apply Lemma 4 to find positions for the elements in V. If we choose $g \in \mathcal{N}(0,1)^m$ and we wish to compute the expected distance between $pos(U, g)$ and $pos(V, g) + h$, then we must determine the angle between U_0 and V_h, in other words we need to compute the angle between U_0 and V_h.

Lemma 10

$$\frac{U_0 \cdot V_h}{|U_0| \cdot |V_h|} = q \sum_{k=0}^{q-1} \cos(\theta_k - \theta_h)\ u_0 \cdot v_k.$$

Proof:

$$\begin{aligned} U_0 \cdot V_h = \sum_{i=0}^{q-1} \cos\left(\frac{2\pi \cdot i}{q}\right) u_i \cdot \Bigg(\left(\cos\left(\frac{2\pi \cdot h}{q}\right)\right) V_y \\ + \left(\sin\left(\frac{2\pi \cdot h}{q}\right)\right) V_x \Bigg) \end{aligned}$$

$$= \sum_{i=0}^{q-1} \cos(\theta_i) u_i \cdot \left(\cos(\theta_h) \sum_{j=0}^{q-1} \cos(\theta_j) v_j + \sin(\theta_h) \sum_{j=0}^{q-1} \sin(\theta_j) v_j \right)$$

$$= \sum_{i=0}^{q-1} \cos(\theta_i) u_i \cdot \left(\sum_{j=0}^{q-1} \cos(\theta_j - \theta_h) v_j \right)$$

$$= \sum_{i=0}^{q-1} \cos(\theta_i) u_i \cdot \left(\sum_{j=i}^{i-1} \cos(\theta_j - \theta_h) v_j \right) \tag{24}$$

$$= \sum_{i=0}^{q-1} \cos(\theta_i) u_0 \cdot \left(\sum_{j=i}^{i-1} \cos(\theta_j - \theta_h) v_{j-i} \right) \tag{25}$$

$$= \sum_{i=0}^{q-1} \cos(\theta_i) \cdot \left(\sum_{j=i}^{i-1} \cos(\theta_j - \theta_h) u_0 \cdot v_{j-i} \right)$$

$$= \sum_{i=0}^{q-1} \cos(\theta_i) \cdot \left(\sum_{j=i}^{i-1} \cos(\theta_i + (\theta_j - \theta_h - \theta_i)) u_0 \cdot v_{j-i} \right)$$

$$= \sum_{i=0}^{q-1} \cos(\theta_i) \cdot \left(\sum_{k=0}^{q-1} \cos(\theta_i + (\theta_k - \theta_h)) u_0 \cdot v_k \right) \tag{26}$$

$$= \sum_{i=0}^{q-1} \cos^2(\theta_i) \cdot \left(\sum_{k=0}^{q-1} \cos(\theta_k - \theta_h) u_0 \cdot v_k \right) \tag{27}$$

$$= \frac{q}{2} \cdot \left(\sum_{k=0}^{q-1} \cos(\theta_k - \theta_h) u_0 \cdot v_k \right). \tag{28}$$

To obtain (24), note that the indices of v_j are computed modulo p. Line (25) follows from the constraints of (Q_2^*). Substituting k for $j-i$ and θ_k for $\theta_j - \theta_i$, we obtain line (26). Using the identity for $\cos(a+b) = \cos a \cos b - \sin a \sin b$ and Lemma 15 (found in Appendix A), we obtain (27). Lastly, applying Lemma 14 (also found in Appendix A), we obtain the final equality.

The lemma follows from the fact that the length of $|U_i| = \frac{1}{\sqrt{2}}$ for all $i \in [q]$ and all $x_u \in V$. This proof can be found in Appendix B. □

Applying Lemma 10, we obtain the following theorem. Let $\theta_i = \frac{2\pi \cdot i}{q}$.

Theorem 3 *Given a set of vectors $\{u_i\}$ that forms a feasible solution to (Q_2^*) corresponding to a set of elements V, we can find a set of positions, $\{pos(x_u)\}$, for all $x_u \in V$ such that for all $x_u, x_v \in V$ and $h \in [q]$:*

$$\{E\}\Big[dist\Big(pos(x_u) + h, pos(x_v)\Big)\Big]$$
$$=$$
$$\arccos\left(q \sum_{i=0}^{q-1} \cos(\theta_i - \theta_h) u_0 \cdot v_i \right) \cdot \frac{q}{2\pi}$$

Theorem 3 follows from Lemma 10.

4.2 Analysis

We now consider the payoff function (1) and show that rounding (Q_2^*) using Theorem 3 gives a good approximation. In particular, let ALG represent the value returned by our algorithm, and let SDP represent the objective value of the formulation (Q_2^*) with the payoff given in (1). We wish to determine the worst case ratio of ALG/SDP. Let $a_i = (u_0 \cdot v_i)p$. By Lemma 8, we have $\sum_{i=0}^{q-1} a_i = 1$. Given a set of values $\{a_0, \ldots a_{q-1}\}$ such that $\sum_{i=0}^{q-1} a_i = 1$, we can assume without loss of generality that $\sum_{i=0}^{q/2} a_i = 1$, i.e. we let $a_i = (u_0 \cdot v_i + u_0 \cdot v_{q/2+i})p$. This follows from the fact that both of the two functions below are symmetric.

$$ALG = 1 - \frac{\arccos\left(\sum_{i=0}^{q/2} \cos(\frac{2\pi \cdot i}{q}) a_i \right)}{\pi},$$

$$SDP = \sum_{i=0}^{q/2} (1 - 2\frac{i}{q}) a_i.$$

Let $\theta_i = \frac{2\pi \cdot i}{q}$. Then we have:

$$ALG = 1 - \frac{\arccos\left(\sum_{i=0}^{q/2} \cos(\theta_i) a_i \right)}{\pi},$$

$$SDP = \sum_{i=0}^{q/2} (1 - \frac{\theta_i}{\pi}) a_i \;=\; 1 - \sum_{i=0}^{q/2} \frac{\theta_i}{\pi}\, a_i.$$

$$\frac{ALG}{SDP} = \frac{\pi - \arccos\left(\sum_{i=0}^{q/2} \cos(\theta_i) a_i \right)}{\pi - \sum_{i=0}^{q/2} \theta_i \cdot a_i} \tag{29}$$

Theorem 4 $\frac{ALG}{SDP} \geqslant .854$.

Proof: For any value of θ from $0 \leqslant \theta \leqslant \pi$, we consider an arbitrary set $\{a_i\}$ such that $\sum_{i=0}^{q/2} \theta_i a_i = \theta$. We first show that there exists some set $\{a_i'\}$ where $a_i' = 0$ for all $i : 0 < i < \frac{q}{4}$ and $a_i' = a_i$ for all $i : \frac{q}{4} < i \leqslant \frac{q}{2}$ such that:

$$\sum_{i=0}^{q/2} \theta_i \cdot a_i \;=\; \sum_{i=0}^{q/2} \theta_i \cdot a_i'$$

and

$$\arccos\left(\sum_{i=0}^{q/2} \cos(\theta_i) \cdot a_i \right)$$
$$\leqslant \tag{30}$$
$$\arccos\left(\sum_{i=0}^{q/2} \cos(\theta_i) \cdot a_i' \right)$$

and both $\sum_{i=0}^{q/2} a_i = 1$ and $\sum_{i=0}^{q/2} a_i' = 1$. In other words, the ratio in (29) is no greater using the set $\{a_i'\}$ in place of the set $\{a_i\}$. The inequality in line

(30) follows from the fact that the function $\cos(\phi)$ is concave on the range $[0 \leqslant \phi \leqslant \pi/2]$. Next, we will show that there is some set $\{a''_i\}$ such that $a''_0 = a'_0$ and $a''_i \neq 0$ for only one value of $i \neq 0$. Let us refer to this index as j. This follows from the fact that the function ϕ is convex on range $[\pi/2 \leqslant \phi \leqslant \pi]$. Since $\sum_{i=0}^{q/2} a''_i = 1$, we have $a''_0 + a''_j = 1$. In other words, we have:

$$\arccos\left(\sum_{i=0}^{q/2} \cos(\theta_i) \cdot a'_i\right) \leqslant \arccos\left(\sum_{i=0}^{q/2} \cos(\theta_i) \cdot a''_i\right) = \arccos\left(a''_0 + \cos(\theta_j) \cdot a''_j\right) \tag{31}$$

We will now show that there exists some θ_x and some value x between 0 and 1 such that $\theta = \theta_x \cdot x$ and:

$$\frac{\pi - \arccos(1 - a''_j + \cos(\theta_j) \cdot a''_j)}{\pi - \theta} \geqslant \frac{\pi - \arccos(1 - x + \cos(\theta_x) \cdot x)}{\pi - \theta} \tag{32}$$

Let $f(\theta_x) = \cos(\theta_x) \cdot x - x = \cos(\theta/x) \cdot x - x$. Note that the righthand side of Equation (32) is minimized when $f(\theta_x)$ is minimized. Substituting $x = \theta/\theta_x$.

$$\begin{aligned} f(\theta_x) &= \cos(\theta_x) \cdot \frac{\theta}{\theta_x} - \frac{\theta}{\theta_x} \\ &= \frac{-\theta(1 - \cos(\theta_x))}{\theta_x}. \end{aligned}$$

Thus $f(\theta_x)$ is minimized when $(1 - \cos(\theta_x))/\theta_x$ is maximized. Note that by Lemma 3.5 in [7], we have:

$$\frac{\theta_x}{1 - \cos(\theta_x)} \geqslant \frac{\pi}{2}(.87856\ldots).$$

Thus, for any fixed value of $\theta = \theta_x \cdot x$, the function $f(\theta_x)$ is minimized for a value of θ_x that we will refer to as θ_{GW}. (However, note that since $x \leqslant 1$, this is only true for $\theta \leqslant \theta_{GW}$. We will deal with θ such that $\theta_{GW} \leqslant \theta < \pi$ separately.) We recall that $x = \theta/\theta_{GW}$. Thus, we have the following function of x.

$$h(x) = \frac{\pi - \arccos(1 - x + \cos(\theta_{GW}) \cdot x)}{\pi - \theta_{GW} \cdot x}$$

Figure 3 shows the function $h(x)$. We conclude that the function is always at least .854 achieves this value for $\theta \approx 37$ degrees.

For values of θ in the range $[\theta_{GW}, \pi)$, we note that $\theta_x \geqslant \theta$. We note that the function $(1 - \cos\theta_x)/\theta_x$ is a decreasing function of θ_x in the range $\theta_{GW} \leqslant \theta < \pi$. Thus, the function is maximized for $\theta_x = \theta$ in this range, and it is straightforward to observe that (32) is always at least 1 for values of θ in this interval. □

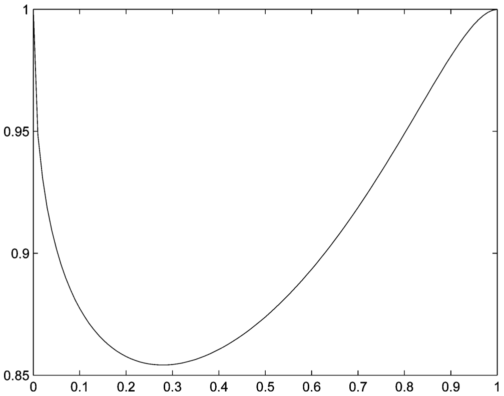

Figure 3: Graph of the function $h(x)$ on the interval $x \in [0, 1]$.

Lemma 11 *If the ratio from* (29) *is considered for a domain of $q = 4$, then the ratio is $\alpha_{GW} > .878$.*

Proof: In the case of $q = 4$, the only non-zero values of a_i correspond to the angles 0, $\pi/2$ and π. Thus, we want to compute the minimum value of the following expression. Let $x = a_0$ and let $y = a_{q/2}$. Note that $\theta = y \cdot \pi$ and that $x + y < 1$.

$$\begin{aligned} \frac{\pi - \arccos(x - y)}{\pi - \theta} &= \frac{\pi - \arccos(x - y)}{\pi - y \cdot \pi - (1 - x - y) \cdot \pi/2} \\ &= \frac{1 - \arccos(x - y)/\pi}{1 - y - (1 - x - y)/2} \\ &= \frac{1 - \arccos(x - y)/\pi}{(1 + x - y)/2} \end{aligned}$$

Let $z = x - y$. Then the above expression is:

$$\frac{1 - \arccos(z)/\pi}{(1 + z)/2}$$

which is has a minimum value of $\alpha_{GW} > .878$. □

Suppose the optima value of an instance of Rel-Lin-Eq(q) is at least $(1 - \epsilon)|E|$. It is not surprising that in this case we can obtain a $(1 - O(\sqrt{\epsilon}))|E|$, since we could obtain the same guarantee for the rounding of (Q_1'), which does not appear to be a stronger relaxation than (Q_2^*).

Lemma 12 *If an optimal solution to an instance of* Rel-Lin-Eq*(q) has value* $(1-\epsilon)|E|$*, then our algorithm finds a solution with value at least* $\big(1 - O(\sqrt{\epsilon})\big)|E|$.

Proof: Given that we fix the angle θ, i.e. the SDP value is fixed, what is the minimum value returned by the algorithm? For $\theta \leqslant \theta_{GW}$, the worst case is:

$$\frac{\pi - \arccos\left(1 - x + \cos\left(\theta_{GW}\right) \cdot x\right)}{\pi - \theta}$$

The value of $\cos(\theta_{GW})$ is at most $-.689$. If $\epsilon = \frac{x \cdot \theta_{GW}}{\pi}$, so $x = \epsilon \cdot \pi/\theta_{GW}$.

$$\frac{1 - \arccos\left(1 - \epsilon(1.689)/\theta_{GW}\right)}{1-\epsilon}$$
$$=$$
$$\frac{1 - \arccos\left(1 - \epsilon \cdot \beta\right)/\pi}{1-\epsilon}$$

where β is a constant that is at least 2.276. This last quantity is $1 - O(\sqrt{\epsilon})$ for small ϵ. □

This is the same asymptotic behavior as the Max Cut algorithm of Goemans-Williamson [7].

5 Linear equations mod 4

We can apply our techniques to obtain an improved approximation factor for the linear equations mod 4 problem. Note that for linear equations mod 3, there is an algorithm with an approximation guarantee of at least .793 [8]. However, such a strong result is not known for the problem of linear equations mod 4. Although previous results indicate that one can do better than random for linear equations mod q (i.e. there is an approximation algorithm with a guarantee strictly better than $1/p$, [1,4]), the best known explicitly computed approximation ratio for $p = 4$ is $1/4 + 1/5120$ [4]. We show here that we can do much better than this. Before doing so, we state the following useful lemma.

Lemma 13 *Given an instance of linear equations* mod q*, the optimal value for the relaxed objective is at least as large as the optimal value for the standard objective.*

Proof: This follows since any solution for the standard linear equations mod q contributes 1 for each satisfied assignment and at least 0 for every unsatisfied assignment. □

theorem 5 *Linear equations* mod 4 *can be approximated to within a factor of* $\alpha_{GW}/2 \approx .439$.

Proof: Consider a solution to the relaxed version of linear equations mod 4. For each equation, we either satisfy it (contributing 1 to the objective function) or we do not satisfy it (contributing .5 or 0 to the objective function). For each element, x_i, we keep the assignment of x_i unchanged with probability .5. With probability .5, we make the assignment $x_i := (x_i + 1)\text{mod}4$. If the equation was satisfied, then we have probability half that it is still satisfied (i.e. both variables remain unchanged or both change). If the contribution was .5, there is a .25 chance that the equation is satisfied.

Thus, we are obtaining a solution with value $\alpha_{GW}/2$ of the optimal for the relaxed version, which by Lemma 13 is at least as large as optimal. Thus, we can conclude that we can satisfy at least $\alpha_{GW}/2$ as many equations as an optimal solution. □

6 Future directions

We plan to run experiments on actual or simulated contig data to see how our methods can be applied in practice. For example, we can consider constraints based on pairs of contigs in an actual arrangement that has had some noise added to it. We note that even if actual contig-mate-pair-graphs are much larger than what our algorithm can handle, we would like to know if constructing a scaffold from a small sample of the contigs gives any helpful information in determining the final arrangement of contigs.

Finally, we remark that we can possibly use our techniques for assembly on a line, simply by constraining the elements to lie on, say, a half circle. Given that the current methods are more naturally tailored to a circle, this would likely be more computationally expensive (more constraints), but is a direction for future investigation.

Acknowledgments

AN would like to thank Nir Ailon, Khaled Elbassioni and Kurt Mehlhorn for valuable suggestions and for comments on an earlier version of this paper. She would also like to thank Kevin Chen, Adel Dayarian and Alexander Schliep for helpful discussions related to genome assembly. This work was done in part while AN was a member of the Algorithms and Complexity Group at the Max-Planck-Institut für Informatik in Saarbrücken, Germany.

References

[1] G. Andersson, L. Engebretsen, and J. Hastad. A new way to use semide?nite programming with ap-plications to linear equations mod p. *Journal of Algorithms*, 39:162-204, 2001.

[2] M. Charikar, K. Makarychev, and Y. Makarychev. Near-optimal algorithms for unique games. In *Pro-ceedings of the 38th Annual Symposium on the Theory of Computing (STOC)*, Seattle, 2006.

[3] A. Dayarian, T. P. Michael, and A. M. Sengupta. SOPRA: Scaffolding algorithm for paired reads via statistical optimization. *BMC Bioinformatics*, 11:345, 2010.

[4] L. Engebretsen and V. Guruswami.Is constraint satisfaction over two variables always easy? *Random Structures and Algorithms,* 25(2):150-178, 2004.

[5] G. Even, J. Naor, S. Rao, and B. Schieber. Divide-and-conquer approximation algorithms via spreading metrics. *Journal of the ACM*, 47(4):585-616, 2000.

[6] U. Feige and M. X. Goemans.Approximating the value of two prover proof systems with applications to MAX-2-SAT and MAX DICUT. In *Proceedings of the Third Israel Symposium on Theory of Computing and Systems*, pages 182-189, 1995.

[7] M.X. Goemans and D. P. Williamson. Improved approximation algorithms for maximum cut and satisfiability problems using semide?nite programming. *Journal of the ACM*,42:1115-1145, 1995.

[8] M.X. Goemans and D. P. Williamson. Approximation algorithms for MAX-3-CUT and other problems via complex semidefinite program-ming. *Journal of Computer and System Sciences*, 68:442-470, 2004.

[9] D. H. Huson, K. Reinert, and E. W. Myers.The greedy path-merging algorithm for contig scaffolding. *Journal of the ACM*, 49(5):603-615, 2002.

[10] S. Khot. On the power of unique 2-prover 1-round games. In *Proceedings of the 34th Annual Symposium on the Theory of Computing (STOC)*, pages 767-775, Montreal, 2002.

[11] E. W. Myers, G. G. Sutton, H. O. Smith, M. D. Adams, and J. C. Venter. On the sequencing and assembly of the human genome. *Proceedings of the National Academy of Sciences*, 99(7):4145-4146, 2002.

[12] P. A. Pevzner and H. Tang. Fragment assembly with double-barreled data. In *ISMB (Supplement of Bioinformatics)*, pages 225-233, 2001.

[13] M. Pop.Genome assembly reborn: recent computational challenges. *Briefings in Bioinformatics*, 10(4):354-366, 2009.

[14] P. Raghavendra. Optimal algorithms and inapprox-imability results for every CSP? In *Proceedings of the 40th ACM Symposium on the Theory of Com-puting (STOC)*, pages 244-254, 2008.

[15] S. Zhang and Y. Huang.Complex quadratic optimization and semidefinite programming. *SIAM Journal on Optimization*, 16(3):871-890, 2006.

A Some basics

We give some definitions and basic lemmas that we use several times. Let $\theta_i = \frac{2\pi \cdot i}{q}$. Let q be an integer representing the size of the domain.

Lemma 14 *If $q \geqslant 3$, then:*

$$\frac{1}{q}\sum_{i=0}^{q-1}\cos^2\theta_i = \frac{1}{q}\sum_{j=0}^{q-1}\sin^2\theta_i = \frac{1}{2}.$$

Proof: If we assume that the two sums are equal, then since the two sums sum to one, it follows that each sum equals one half. Thus, it remains to prove that the two sums are equal. We have:

$$\sum_{i=0}^{q-1}\sin^2\theta_i = \sum_{i=0}^{q-1}\frac{1-\cos(2\theta_i)}{2}$$
$$\sum_{i=0}^{q-1}\cos^2\theta_i = \sum_{i=0}^{q-1}\frac{1+\cos(2\theta_i)}{2}.$$

Thus, if we can show that:

$$\sum_{i=0}^{q-1}\cos(2\theta_i) = 0,$$

then we are done. To do this we can use complex numbers. Let $x = \frac{4\pi}{q}$. We have:

$$\begin{aligned}\sum_{j=0}^{q-1}\cos(2\theta_j) &= \mathrm{Re}\left(\sum_{j=0}^{q-1} e^{\mathbf{i}(xj)}\right)\\ &= \mathrm{Re}\left(\frac{1-e^{\mathbf{i}(4\pi)}}{1-e^{\mathbf{i}\frac{4\pi}{q}}}\right)\\ &= \mathrm{Re}\left(\frac{1-\cos 4\pi - \mathbf{i}\sin 4\pi}{1-\cos\frac{4\pi}{q} - \mathbf{i}\sin\frac{4\pi}{q}}\right)\end{aligned}$$

We can multiply both the numerator and denominator by the complex conjugate of the denominator:

$$\begin{gathered}\mathrm{Re}\left(\left(\frac{1-\cos 4\pi - \mathbf{i}\sin 4\pi}{1-\cos x - \mathbf{i}\sin x}\right)\left(\frac{1-\cos x + \mathbf{i}\sin x}{1-\cos x + \mathbf{i}\sin x}\right)\right)\\ =\\ \mathrm{Re}\left(\frac{0\cdot(1-\cos x + \mathbf{i}\sin x)}{(1-\cos x)^2+\sin^2 x}\right)\\ =\\ \mathrm{Re}\left(\frac{0\cdot(1-\cos x + \mathbf{i}\sin x)}{2-2\cos x}\right) \qquad (33)\\ = 0\end{gathered}$$

Note that the numerator is always 0, regardless of q. However, the denominator is also 0 when $q = 1, 2$. Thus, the lemma holds only for $q \geqslant 3$. □

Lemma 15 *If $q \geqslant 3$, then:*

$$\sum_{i=0}^{q-1}\cos\theta_i\ \sin\theta_i = 0.$$

Proof: Since we have that $\cos\theta_i \cdot \sin\theta_i = 2\sin 2\theta_i$, the resulting sum obtained by this substitution is just the imaginary part of the expression (33). When $q \geqslant 3$, the imaginary part of this expression is always 0. □

Fact 1

$$\begin{aligned}\sum_{j=0}^{q-1}\cos(\theta_j)v_j &= \sum_{j=0}^{q-1}\cos(\theta_i + (\theta_j-\theta_i))v_j\\ &= \sum_{j=0}^{q-1}(\cos(\theta_i)\cos(\theta_j-\theta_i) -\\ &\qquad \sin(\theta_i)\sin(\theta_j-\theta_i))v_j\end{aligned}$$

Fact 2

$$\begin{aligned}\sum_{j=0}^{q-1}\sin(\theta_j)v_j &= \sum_{j=0}^{q-1}\sin(\theta_i + (\theta_j-\theta_i))v_j\\ &= \sum_{j=0}^{q-1}(\sin(\theta_i)\cos(\theta_j-\theta_i) +\\ &\qquad \cos(\theta_i)\sin(\theta_j-\theta_i))v_j\end{aligned}$$

B Proof of Lemma 9

Given a set of vectors $\{u_i\}$ for $u \in V$ and $i \in [p]$ that form a feasible solution to the relaxation (Q_2'), we show that the corresponding vectors $\{U_x, U_y\}$ satisfy the following properties. We assume that $q \geqslant 3$. Recall that $\theta_i = \frac{2\pi\cdot i}{q}$.

(i) $U_x \cdot U_y = 0$, for all $x_u \in V$,

(ii) $|U_x| = |U_y| = |V_x| = |V_y|$, for all $x_u, x_v \in V$,

(iii) $U_x \cdot V_y = U_y \cdot (-V_x)$, for all $x_u, x_v \in V$,

(iv) $U_x \cdot V_x = U_y \cdot V_y$, for all $x_u, x_v \in V$.

(i)

$$\begin{gathered}U_x \cdot U_y\\ =\\ \left(\sum_{i=0}^{q-1}\sin\left(\frac{2\pi\cdot i}{q}\right)u_i\right)\cdot\left(\sum_{j=1}^{q-1}\cos\left(\frac{2\pi\cdot j}{q}\right)u_j\right)\end{gathered}$$

Since $u_i \cdot u_j \neq 0$ if and only if $i = j$, we have:

$$\begin{aligned}U_x \cdot U_y &= \left(\sum_{i=0}^{q-1}\sin\theta_i\ \cos\theta_i\ (u_i\cdot u_i)\right)\\ &= \frac{1}{q}\left(\sum_{i=0}^{q-1}\sin\theta_i\ \cos\theta_i\right)\\ &= 0\end{aligned}$$

The last equality follows from Lemma 15.

(ii)

$$\begin{aligned}U_x \cdot U_x &= \sum_{i=0}^{q-1}\sin\left(\frac{2\pi\cdot i}{q}\right)u_i \cdot \sum_{i=0}^{q-1}\sin\left(\frac{2\pi\cdot i}{q}\right)u_i\\ &= \sum_{i=0}^{q-1}\sin^2(\theta_i)u_i\cdot u_i\\ &= \frac{1}{q}\sum_{i=0}^{q-1}\sin^2(\theta_i)\\ &= \frac{1}{2}\end{aligned}$$

The last inequality follow from Lemma 14. Thus we have $|U_x| = \frac{1}{\sqrt{2}}$. Note that we also use Lemma 14 to show that $|U_y| = \frac{1}{\sqrt{2}}$. Thus, property (ii) holds for all $u \in V$.

(iii)

$$U_x \cdot V_y = \sum_{i=0}^{q-1}\sin\left(\frac{2\pi\cdot i}{q}\right)u_i \cdot \sum_{j=0}^{q-1}\cos\left(\frac{2\pi\cdot j}{q}\right)v_j$$

By Fact 1, we have:

$$\begin{aligned}U_x \cdot V_y &= -\sum_{i=0}^{q-1} \sin^2 \theta_i u_i \sum_{j=0}^{q-1} \sin(\theta_j - \theta_i) v_j \\ &= -\sum_{j=0}^{q-1} \sin^2 \theta_i \sum_{k=0}^{q-1} \sin(\theta_k) u_0 \cdot v_k\end{aligned}$$

By definition, we have:

$$U_y \cdot (-V_x) = \sum_{i=0}^{q-1} \cos(\frac{2\pi \cdot i}{q}) u_i \cdot \left(-\sum_{j=0}^{q-1} \sin(\frac{2\pi \cdot j}{q}) v_j\right)$$

By Fact 2, we have:

$$\begin{aligned}U_y \cdot (-V_x) &= -\sum_{i=0}^{q-1} \cos^2 \theta_i u_i \sum_{j=0}^{q-1} \sin(\theta_j - \theta_i) v_j \\ &= -\sum_{j=0}^{q-1} \cos^2 \theta_i \sum_{k=0}^{q-1} \sin(\theta_k) u_0 \cdot v_k.\end{aligned}$$

Thus, by Lemma 14, we see that property (iii) holds.

(iv)

$$\begin{aligned}U_x \cdot V_x &= U_y \cdot V_y \\ &= \sum_{i=0}^{q-1} \cos(\frac{2\pi i}{q})\ u_0 \cdot v_i.\end{aligned}$$

$$U_x \cdot V_x = \left(\sum_{i=0}^{q-1} \sin(\theta_i) u_i\right) \cdot \left(\sum_{j=0}^{q-1} \sin(\theta_j) v_j\right)$$

We recall that for $j \geqslant i$, we have $u_i \cdot v_j = u_0 \cdot v_{(j-i)}$. By Fact 2, we have:

$$\begin{aligned}U_x \cdot V_x = &\sum_{i=0}^{q-1} \sum_{j=i}^{q-1+i} (\sin^2(\theta_i) \sin(\theta_j - \theta_i) - \\ &\cos(\theta_i) \sin(\theta_i) \sin(\theta_j - \theta_i)) u_0 \cdot v_{(j-i)} \\ = &\sum_{i=0}^{q-1} \sin^2 \theta_i \sum_{k=0}^{q-1} (\cos(\theta_k))\ u_0 \cdot v_k \\ = &\frac{q}{2} \sum_{k=0}^{q-1} (\cos(\theta_k))\ u_0 \cdot v_k\end{aligned}$$

$$U_y \cdot V_y = (\sum_{i=0}^{q-1} \cos(\theta_i) u_i) \cdot (\sum_{j=0}^{q-1} \cos(\theta_j) v_j)$$

We recall that for $j \geqslant i$, we have $u_i \cdot v_j = u_0 \cdot v_{(j-i)}$. By Fact 1, we have:

$$\begin{aligned}U_y \cdot V_y = &\sum_{i=0}^{q-1} \sum_{j=i}^{q-1+i} (\cos^2(\theta_i) \cos(\theta_j - \theta_i) - \\ &\cos(\theta_i) \sin(\theta_i) \sin(\theta_j - \theta_i)) u_0 \cdot v_{(j-i)} \\ = &\sum_{i=0}^{q-1} \cos^2 \theta_i \sum_{k=0}^{q-1} (\cos(\theta_k))\ u_0 \cdot v_k \\ = &\frac{q}{2} \sum_{k=0}^{q-1} (\cos(\theta_k))\ u_0 \cdot v_k.\end{aligned}$$

Thus, property (iv) holds. □