Directed *st*-Connectivity with Few Paths Is in Quantum Logspace

Simon Apers ✉ ⌂ ◉
Université Paris Cité, CNRS, IRIF, Paris, France

Roman Edenhofer ✉ ⌂ ◉
Université Paris Cité, CNRS, IRIF, Paris, France

──── **Abstract** ────

We present a $\mathsf{BQSPACE}(O(\log n))$-procedure to count st-paths on directed graphs for which we are promised that there are at most polynomially many paths starting in s and polynomially many paths ending in t. For comparison, the best known classical upper bound in this case just to decide st-connectivity is $\mathsf{DSPACE}(O(\log^2 n/\log\log n))$. The result establishes a new relationship between BQL and unambiguity and fewness subclasses of NL. Further, we also show how to *recognize* directed graphs with at most polynomially many paths between any two nodes in $\mathsf{BQSPACE}(O(\log n))$. This yields the first natural candidate for a language separating BQL from L and BPL. Until now, all candidates potentially separating these classes were inherently promise problems.

1 Introduction and summary

Graph connectivity is a central problem in computational complexity theory, and it is of particular importance in the space-bounded setting. Given a graph G and two vertices s and t, the task is to decide whether there is a path from s to t. For undirected graphs the problem is denoted as USTCON. Aleliunas, Karp, Lipton, Lovász and Rackoff [1] showed that doing a random walk for a polynomial number of steps can solve it in *randomized logspace*, RL. After a long line of work, Reingold [21] was able to derandomize the result and showed that the problem is already contained in *deterministic logspace*, L, and is in fact complete for that class. In the directed graph setting the problem is denoted as STCON, and it is complete for *non-deterministic logspace*, NL. The best known deterministic algorithm for STCON in terms of space complexity is due to Savitch [23] and runs in space $O(\log^2 n)$.

COMPUTATIONAL
COMPLEXITY
CONFERENCE

We have the following well-known inclusions

$$\mathsf{L} \subseteq \mathsf{RL} \subseteq \mathsf{NL} \subseteq \mathsf{DET} \subseteq \mathsf{L}^2$$

where DET, introduced by Cook [7], is the class of languages that are NC^1 Turing reducible to the computation of the determinant of an integer matrix and L^2 is the class of languages decidable in deterministic space $O(\log^2 n)$. We refer to [4] for further details on reductions and basic complexity classes.

The most studied quantum space-bounded complexity class is *bounded error quantum logspace*, denoted BQL. This is the class of languages decided in $\mathsf{BQSPACE}(O(\log n))$, i.e., decided by a quantum Turing machine with error $1/3$ running in space $O(\log n)$ and time $2^{O(\log n)}$. The quantum Turing machine has read-only access to the input on a classical tape and can write on a uni-directional output tape which does not count towards the space complexity. The class BQL lies in between RL and DET. In fact, it was recently shown by Fefferman and Remscrim [10] that certain restricted versions of the standard DET-complete matrix problems are complete for prBQL, where prBQL is the promise version of BQL, i.e. the class of promise problems decided in $\mathsf{BQSPACE}(O(\log n))$. This extended the earlier work of Ta-Shma [24], who showed how to invert well-conditioned matrices in $\mathsf{BQSPACE}(O(\log n))$ building on the original idea of Harrow, Hassidim and Lloyd [15]. We restate two of Ta-Shma's main results:

1. (Compare [24, Theorem 5.2]) Given a matrix $M \in \mathbb{C}^{n \times n}$ with $\|M\|_2 \leq \mathrm{poly}(n)^1$, we can output all of its singular values and their respective multiplicities up to $1/\mathrm{poly}(n)$ additive accuracy in $\mathsf{BQSPACE}(O(\log n))$. In particular, we can determine $\dim(\ker(M))$ if all non-zero singular values have inverse polynomial distance from zero.

2. (Compare [24, Theorem 1.1]) Given two indices $s, t \in [n]$ and a matrix $M \in \mathbb{C}^{n \times n}$ which is *poly-conditioned*, by this we mean that its singular values satisfy

$$\mathrm{poly}(n) \geq \sigma_1(M) \geq ... \geq \sigma_n(M) \geq 1/\mathrm{poly}(n),$$

we can estimate $M^{-1}(s,t)$ up to $1/\mathrm{poly}(n)$ additive accuracy in $\mathsf{BQSPACE}(O(\log n))$.

The first result above directly implies a BQL-procedure for deciding USTCON (alternatively, this clearly follows from the containment $\mathsf{RL} \subseteq \mathsf{BQL}$). To see this, note that (i) USTCON can be reduced to counting the number of connected components, (ii) the dimension of the kernel of the random walk Laplacian $I - P$ is equal to the number of connected components, and (iii) for undirected graphs the spectral gap of $I - P$, that is its smallest non-zero eigenvalue, is inverse polynomially bounded from zero. Here I is the identity and P is the transition matrix of a random walk. This ties to the fact that a random walk on an undirected graph takes polynomial time to traverse the graph (see e.g. [19, Chapter 12]). Unfortunately, for directed graphs the situation is more complicated. Importantly, the smallest non-zero singular value of the random walk Laplacian can be inverse exponentially small, and similarly the time it takes a random walk to find connected nodes can be exponential. Hence, it is not obvious how Ta-Shma's results should be of any help in this setting.

---

1. Throughout this paper, $\| \cdot \|_2$ shall always denote the ℓ_2-norm of a vector or the spectral norm of a matrix.

Counting few paths

Somewhat surprisingly, we show that by analyzing a different matrix which we call the *counting Laplacian* $L = I - A$ we can use Ta-Shma's second result to solve instances of STCON in $\mathsf{BQSPACE}(O(\log n))$ that seem hard classically. Here A denotes the adjacency matrix of a graph. While the random walk Laplacian is spectrally well-behaved if a random walk is efficient, we find that the counting Laplacian is spectrally well-behaved if the underlying graph is acyclic and contains only few paths. We remark that the number of paths in a graph can be totally unrelated to the success probability of a random walk finding specific nodes. Even if there exist very few paths, it can happen that a random walk has an extreme bias to only pick paths we are not interested in. Consider for instance the following graph:
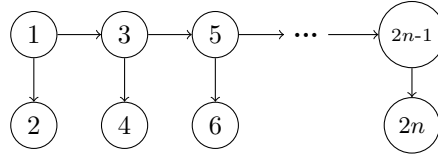
The number of paths between any two nodes is at most one. Nonetheless, a random walk starting at node 1 only has probability $1/2^{n-1}$ to reach node $2n$.

Given a directed graph $G = (V, E)$ with nodes $i, j \in V$ let us denote by $N(i, j)$ the number of paths from i to j. By a path from i to j we mean a sequence of edges $(e_1, ..., e_k)$ which joins a sequence of nodes $(v_0, ..., v_k)$ such that $v_0 = i$, $v_k = j$ and $e_i = (v_{i-1}, v_i)$ for all $i \in [k]$.[2] As our first result, and as a primer for the rest of the paper, we show that we can count the number of *st*-paths on directed graphs for which there are at most polynomially many paths between any two nodes in $\mathsf{BQSPACE}(O(\log n))$. In particular, this allows to decide STCON on such graphs.

▶ **Theorem 1.** *Fix a polynomial $p : \mathbb{N} \to \mathbb{N}$. Let G be a directed graph with $|V(G)| = n$ nodes such that*

▬ $\forall i, j \in V(G) : N(i, j) \leq p(n)$.

There is an algorithm running in $\mathsf{BQSPACE}(O(\log n))$ that, given access to the adjacency matrix A of G and $s, t \in V(G)$, returns the number of paths from s to t.

Proof. First of all, observe that the finite path bound between any two nodes implies that the graph is in particular acyclic. Second, note that $A^k(i, j)$ is equal to the number of paths of length k from node i to node j. Both in mind, it follows that $A^n = 0$, i.e. A is nilpotent. As a consequence we obtain that the inverse of the counting Laplacian exists and is equal to

$$L^{-1} = (I - A)^{-1} = I + A + A^2 + ... + A^{n-1}.$$

Clearly, its entries simply count the total number of paths from i to j, i.e. $L^{-1}(i, j) = N(i, j)$. Crucially, we find that the polynomial bound on the number of paths ensures that the matrix is poly-conditioned. Observe that the max-norm of a matrix, i.e. its maximum entry in absolute value, can be used to bound its spectral norm. More precisely, we have that $\|M\|_2 \leq n \cdot \|M\|_{\max}$ for any matrix M. Using that $\|L^{-1}\|_{\max} = \max_{i,j\in[n]} |L^{-1}(i,j)| \leq p(n)$ and $\|L\|_{\max} = 1$, we thus find bounds for the largest and smallest singular value of L,

---

[2] Some authors call this a *walk* and disallow a path to visit any vertex more than once. We follow [3, 17] in our definition. By convention we also count the empty sequence as a path from any node to itself such that $N(i, i) \geq 1$ for all $i \in V$.

$$n \geq \sigma_1(L) \quad \text{and} \quad \sigma_n(L) = \frac{1}{\|L^{-1}\|_2} \geq \frac{1}{n \cdot \|L^{-1}\|_{\max}} \geq \frac{1}{n \cdot p(n)}.$$

Hence, the counting Laplacian is poly-conditioned and we can apply Ta-Shma's matrix inversion algorithm, the second restated result above, to approximate entry $L^{-1}(s,t) = N(s,t)$ up to additive error $1/3$ in $\mathsf{BQSPACE}(O(\log n))$. Rounding to the closest integer gives the number of paths from s to t. ◀

The proof of Theorem 1 uses a reduction of summing matrix powers to matrix inversion. The idea for this is well-known, and appeared in this context in early work by Cook [7], and more recently by Fefferman and Remscrim [10]. However, they did not study the implications to the space complexity of STCON.

In Section 4 we push the algorithmic idea of Theorem 1 further by doing a more fine-grained analysis of the counting Laplacian's singular values and singular vectors. We show that we can already count *st*-paths in $\mathsf{BQSPACE}(O(\log n))$ on directed graphs for which only the number of paths starting from s and the number of paths ending in t are polynomially bounded. We state the formal result here.

▶ **Theorem 2.** *Fix a polynomial $p : \mathbb{N} \to \mathbb{N}$. Let G be a directed graph with $|V(G)| = n$ nodes such that*
■ *$\forall j \in V(G) : N(s,j) \leq p(n)$ and $N(j,t) \leq p(n)$.*
There is an algorithm running in $\mathsf{BQSPACE}(O(\log n))$ that, given access to the adjacency matrix A of G and $s,t \in V(G)$, returns the number of paths from s to t.

Classically (deterministically or randomly), the best known space bound just to decide STCON on graphs promised to satisfy a polynomial bound on the number of paths between any two nodes as in Theorem 1 or only on the number of paths starting from s and on the number of paths ending in t as in Theorem 2 is $\mathsf{DSPACE}(O(\log^2(n)/\log\log(n)))$ [3, 12]. Alternatively, as noticed in [6, 18] we can also solve such STCON instances simultaneously in deterministic polynomial time and space $O(\log^2 n)$. We elaborate on these bounds in the next section.

Unambiguity, Fewness and a language in BQL (maybe) not in L

Previous works studied restrictions on the path count in connection to "unambiguity" and "fewness" of configuration graphs of space-bounded Turing machines. These notions were introduced to interpolate between determinism and non-determinism, and gives rise to complexity classes between L and NL. We formally introduce these classes in Section 2. As a direct consequence of Theorem 1 we obtain:

▶ **Corollary 3.** $\mathsf{StrongFewL} \subseteq \mathsf{BQL}$

where $\mathsf{StrongFewL}$ denotes the class of languages which are decidable by an NL-Turing machine whose configuration graphs for all inputs are *strongly-few*, that is graphs for which there are at most polynomially many paths between any two nodes. By some preprocessing in the algorithm of Theorem 1 we can actually check in $\mathsf{BQSPACE}(O(\log n))$ whether a given graph is strongly-few.[3] We obtain the subsequent language containment (notably, the containment of this language in $\mathsf{StrongFewL}$ is not known).

---

[3] Unfortunately, we see no way to do the same for the weaker promise of Theorem 2.

▶ **Theorem 4.** *The language*

$$\text{STCON}_{\text{sf}} = \{\langle G, s, t, 1^k \rangle \mid \forall i, j \in V(G) : N(i,j) \leq k \text{ and } N(s,t) \geq 1\}$$

is contained in BQL.

This seems to be the first concrete example of a *language* in BQL not known to lie in L or BPL. As far as we are aware, before this work only *promise problems* were known that lie in prBQL but which are potentially not contained in prBPL [24, 9, 10, 11].

We now summarize briefly what is known classically. A language similar to STCON_{sf}, namely

$$\text{STCON}_{\text{ru}} = \{\langle G, s, t \rangle \mid \forall j \in V(G) : N(s,j) \leq 1 \text{ and } N(s,t) = 1\}$$

has been studied before. It was first introduced by Lange [18] who showed that it is complete for ReachUL, which is the class of languages decidable by an NL-Turing machine whose configuration graphs for all inputs are *reach-unambiguous*, meaning that there is a unique computation path from the start configuration to any reachable configuration.[4] Further, it was noticed in [18, 6] that ReachUL is contained in SC^2, which is the class of languages decidable simultenously in deterministic polynomial time and space $O(\log^2 n)$. Additionally, Allender and Lange [3] found that STCON on graphs promised to be reach-unambiguous is solvable in deterministic space $O(\log^2(n)/\log\log(n))$ implying ReachUL $\subseteq$ DSPACE($O(\log^2(n)/\log\log(n))$). In particular, these results put STCON_{ru} in SC^2 and in DSPACE($O(\log^2(n)/\log\log(n))$). Also, more recently Garvin, Stolee, Tewari and Vinodchandran [12] proved that ReachUL is equal to ReachFewL, where ReachFewL is the class of languages decidable by an NL-Turing machine whose configuration graphs for all inputs are *reach-few*, i.e. they have at most polynomially many computation paths from its start configuration to any reachable configuration. We thus have

$$\text{StrongFewL} \subseteq \text{ReachFewL} = \text{ReachUL} \subseteq \text{SC}^2, \text{ DSPACE}(O(\log^2(n)/\log\log(n))).$$

Notably, while [12] implies a procedure to solve STCON on graphs promised to be reach-few in DSPACE($O(\log^2(n)/\log\log(n))$) (in particular, this includes the graphs from Theorems 1 and 2), this does not allow to verify whether a general graph satisfies this promise. Indeed, none of the upper bounds for STCON_{ru} directly carry over to STCON_{sf}, for which the best classical space bound to date seems to be $O(\log^2 n)$.

Conclusion and open questions

Summarizing, we show that in quantum logspace we can count *st*-paths on graphs for which even solving *st*-connectivity is not known to be possible in classical (deterministic or randomized) logspace. Further, we obtain the first language in BQL not known to lie in L or BPL. Our work also yields a number of open questions.

An obvious first question is whether STCON_{sf} really separates BQL and BPL. A first step towards answering this might be tackling the related question, whether we can carry over some of the known upper bounds for STCON_{ru} to STCON_{sf}. That is, is STCON_{sf} also contained in SC^2 or in DSPACE($O(\log^2(n)/\log\log(n))$)? Note that the known prBQL-complete problems are not known to satisfy these upper bounds [24, 9, 10, 11].

---

[4] Note that the ReachUL-hardness of STCON_{ru} is trivial but the completeness is not. This is because the uniqueness in the definition of the complexity class is used as a restriction on the machine, while it is used as an acceptance criterion in the definition of the language.

Further, we wonder whether it is possible to improve the $O(\log^2(n)/\log\log(n))$ space bound by Allender and Lange. In fact, Allender recently asked this in an article [2] in which he reflects on some open problems he encountered throughout his career. Allender suspects that for the restricted case of *strongly-unambiguous* graphs, also called *mangroves*, for which the number of paths between any two nodes is bounded by one, there should exist an algorithm deciding STCON running in deterministic space $O(\log n)$. He even offers a \$1000 reward for any improvement of their space bound, already for this restricted case. A dequantization of our results would thus yield some good pocket money.[5]

Another natural question is whether in $\mathsf{BQSPACE}(O(\log n))$ we can solve STCON on a larger class of graphs such as reach-few ones, where only the number of paths from s is polynomially bounded. This relaxes our promise in Theorem 2. Unfortunately, our current approach of using an effective pseudoinverse seems to require our stronger promise. More generally, we feel that a better understanding of the singular values and vectors of directed graph matrices will yield further insights into the utility of quantum space-bounded computation for solving problems on directed graphs.

We further believe it would be interesting to investigate whether there is a quantum algorithm running simultaneously in polynomial time and truly sublinear space $O(n^{1-\varepsilon})$ for some $\varepsilon > 0$ deciding STCON on general graphs. Note that the best known classical algorithm for deciding STCON on general graphs and running in polynomial time needs almost linear space $\Omega(n/2^{\sqrt{\log n}})$ as described in [5].

Finally, the link between poly-conditionedness and bounds on the path count raises the question of whether some variation of STCON could be proven complete for BQL.

2   Space-bounded computation

In Section 2.1 we introduce the Turing machine model of space-bounded computation and define the most important appearing complexity classes. We mainly follow the definitions from Ta-Shma [24] and refer to [10, Section 2.2] for a discussion on the equivalence of the quantum Turing machine model and the quantum circuit model. In Section 2.2 we mention some complete problems of promise versions of the defined complexity classes.

2.1   Turing machines

A *deterministic* space-bounded Turing machine (DTM) acts according to a transition function δ on three semi-infinite tapes: A read-only tape where the input is stored, a read-and-write work tape and a uni-directional write-only tape for the output. The TMs computation time is defined as the number of transition steps it performs on an input, and its computation space is the number of used cells on the work tape, i.e. we do not count the number of cells on the input or output tape towards its computation space. DTMs with space-bound $s(n)$ for inputs of length n give rise to $\mathsf{DSPACE}(s(n))$. We define L as the class of languages decided in $\mathsf{DSPACE}(O(\log n))$ and L^2 as the class of languages decided in $\mathsf{DSPACE}(O(\log^2 n))$.

A *non-deterministic* Turing machine (NTM) is similar to a DTM except that it has two transition functions δ_0 and δ_1. At each step in time the machine non-deterministically chooses to apply either one of the two. It is said to accept an input if there is a sequence of

---

[5] Even if the dequantization were randomized, i.e. in $\mathsf{BPSPACE}(O(\log n))$, it would still imply a $\mathsf{DSPACE}(O(\log^{3/2} n))$ bound by the inclusion $\mathsf{BPSPACE}(O(\log n)) \subseteq \mathsf{DSPACE}(O(\log^{3/2} n))$ due to Saks and Zhou [22] which has recently been slightly improved on by Hoza [16].

these choices so that it reaches an accepting configuration and it is said to reject the input if there is no such sequence of choices. We obtain $\mathsf{NSPACE}(s(n))$. Further, NL is the class of languages decided in $\mathsf{NSPACE}(O(\log n))$.

A *probabilistic* space-bounded Turing machine (PTM) is again similar to a DTM but with the additional ability to toss random coins. This can be conveniently formulated by a fourth tape that is uni-directional, read-only and initialized with uniformly random bits at the start of the computation. It does not count towards the space. A language is said to be decided in $\mathsf{BPSPACE}_{a,b}(s(n))$ if there is a PTM running in space $s(n)$ and time[6] $2^{O(s(n))}$ deciding it with completeness error $a \in [0,1]$ and soundness error $b \in [0,1]$, that is every input in the language is accepted with probability at least a and every input not in the language is accepted with probability at most b. Also, we write $\mathsf{BPSPACE}(s(n))$ for $\mathsf{BPSPACE}_{\frac{1}{3},\frac{2}{3}}(s(n))$ and $\mathsf{RSPACE}(s(n))$ for $\mathsf{BPSPACE}_{\frac{1}{2},0}(s(n))$. Further, BPL is the class of languages decided in $\mathsf{BPSPACE}(O(\log n))$ and RL is the class of languages decided in $\mathsf{RSPACE}(O(\log n))$.

A *quantum* space-bounded Turing machine (QTM) is a DTM with a fourth tape for quantum operations instead of a random coins tape. The transition function δ is still classically described. The tape cells of the quantum tape are qubits and initialized in state $|0\rangle$ at the start of the computation. There are two tape heads moving in both directions on the quantum tape. At each step during the computation, the machine can apply a gate from some universal gate set, say $\{\mathrm{HAD}, \mathrm{T}, \mathrm{CNOT}\}$, or perform a measurement to a projection in the standard basis to the qubits below its tape heads. The measurement outcomes are communicated to the machine and can therefore control later steps of the computation. In particular, the machine can reset qubits to their initial state. The used cells on the quantum tape count towards the computation space. As before, we say a language is decided in $\mathsf{BQSPACE}_{a,b}(s(n))$ if there is a QTM running in space $s(n)$ and time $2^{O(s(n))}$ deciding it with completeness error a and soundness error b. Also, we mean $\mathsf{BQSPACE}_{\frac{2}{3},\frac{1}{3}}(s(n))$ by $\mathsf{BQSPACE}(s(n))$ if not mentioned otherwise. Finally, BQL is the class of languages decided in $\mathsf{BQSPACE}(O(\log n))$. The particular choice of the universal gate set does not affect the resulting complexity class due to the space-efficient version of the Solovay-Kitaev theorem of van Melkebeek and Watson [20]. The same applies to disallowing intermediate measurements, thanks to the space-efficient "deferred measurement principle" proven by Fefferman and Remscrim [10] building on the earlier work of Fefferman and Lin [9] (see also [14, 13] for an alternative time- and space-efficient version of this principle which applies to a special type of intermediate measurements that cannot control later computations). Also, the chosen success probability of 2/3 can be amplified by sequentially repeating computations.

2.2   Complete (promise) problems

In this work we only consider language classes. In particular, we present the first candidate to distinguish BQL and BPL. Note that candidates for distinguishing the promise versions of these classes, prBQL and prBPL, are well-known. As mentioned in the introduction, Fefferman and Remscrim [10] showed that well-conditioned promise versions of all the standard DET-complete matrix problems are complete for prBQL. Let us highlight that matrix powering is one of these problems. Restricted to stochastic matrices it is easily seen to be complete for prBPL, and restricted to matrices for which the largest singular values of its powers grow at most polynomially, it is complete for prBQL. Indeed, this seems to indicate that powering

---

[6] The time bound does not follow from the space-bound and is equivalent to demanding that the TM absolutely halts for all possible assignments of the random coins tape.

the adjacency matrix of a graph, to which our approach in Theorem 1 essentially boils down to, rather than powering the corresponding random walk matrix, truly exploits a quantum advantage. A similar distinction of prBQL and prBPL is expected from the approximation of the spectral gap of matrices. Doron, Sarid and Ta-Shma [8] showed that a promise decision version of this problem is prBPL-complete for stochastic matrices while it is prBQL-complete for general Hermitian matrices. More recently, Le Gall, Liu and Wang [11] presented another group of prBQL-complete problems based on state testing.

3   Fewness language in BQL

In Section 3.1 we introduce the notions of unambiguity and fewness in space-bounded computation. In Section 3.2 we prove Theorem 4 presenting a language in BQL not known to lie in L or BPL.

3.1   Unambiguity and fewness

The computation of a Turing machine can be viewed as a directed graph on configurations, and certain restrictions on the Turing machine translate to natural restrictions on the corresponding configuration graph. The notions of "unambiguity" and "fewness" of a Turing machine relate to the following graph-theoretic notions.

▶ **Definition 5.** *Let $G = (V, E)$ be a directed graph and let k be an integer. Then G is called*
- *k-unambiguous with respect to nodes $s, t \in V$ if $N(s, t) \leq k$,*
- *k-reach-unambiguous with respect to node $s \in V$ if for all $j \in V$, $N(s, j) \leq k$,*
- *k-strongly unambiguous if for all $i, j \in V$, $N(i, j) \leq k$.*

In the case of $k = 1$, we simply say G is unambiguous, reach-unambiguous or strongly unambiguous, respectively. Furthermore, a family of directed graphs $\{G_x\}_{x \in X}$ is called few-unambiguous, reach-few *or* strongly-few *if there exists a polynomial $p : \mathbb{N} \to \mathbb{N}$ such that each of the graphs G_x from the family with $|V(G_x)| = n$ nodes is $p(n)$-unambiguous, $p(n)$-reach-unambiguous or $p(n)$-strongly unambiguous, respectively.*

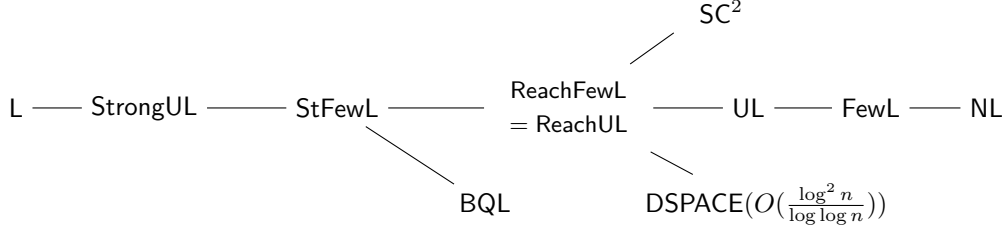Consider the following examples of the above definition due to Lange [18]:
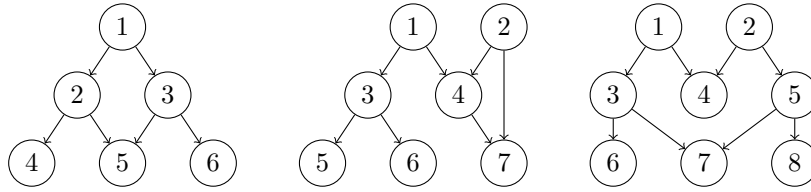
The left graph is unambiguous with respect to nodes 1 and 6 but not reach-unambiguous with respect to node 1, the middle one is reach-unambiguous with respect to node 1 but not strongly unambiguous and the right one is strongly unambiguous.

These notions of unambiguity and fewness naturally give rise to six complexity classes between L and NL. We follow the original paper of Buntrock, Jenner, Lange and Rossmanith [6] and define *strongly-unambiguous logspace*, StrongUL, *strongly-few logspace*, StrongFewL, *reach-unambiguous logspace*, ReachUL, *reach-few logspace*, ReachFewL, *unambiguous logspace*, UL, and *few logspace*, FewL as the classes of languages that are decidable by an NL-Turing machine M with unique accepting configuration whose family of configuration graphs for inputs $x \in \Sigma^*$, $\{G_{M,x}\}_{x \in \Sigma^*}$, satisfies the corresponding unambiguity or fewness restriction with respect to its starting and its accepting configuration.

Recall that Theorem 1 implies the inclusion:

▶ **Corollary 3.** StrongFewL $\subseteq$ BQL

Putting this together with the trivial containments and the previously mentioned results in [6, 18, 3, 12], we obtain the following inclusion diagram to read from left to right:

$$\begin{array}{c} \text{SC}^2 \\ \diagup \end{array}$$

L —— StrongUL —— StFewL ——— ReachFewL = ReachUL ——— UL —— FewL —— NL

BQL $\qquad\qquad$ DSPACE$(O(\frac{\log^2 n}{\log\log n}))$

3.2 Proof of Theorem 4

Observe that Theorem 1 only shows that we can decide *st*-connectivity on directed graphs that are promised to be strongly-few. We now show that we can also check whether this promise holds in BQSPACE$(O(\log n))$, i.e. we obtain a language containment. We restate the result mentioned in the introduction:

▶ **Theorem 4.** *The language*

$$\text{STCON}_{\text{sf}} = \{\langle G, s, t, 1^k\rangle \mid \forall i,j \in V(G) : N(i,j) \leq k \text{ and } N(s,t) \geq 1\}$$

is contained in BQL.

The idea for proving the above theorem is to use Ta-Shma's spectrum approximation procedure to estimate the smallest singular value of the counting Laplacian L. In case the graph is acyclic and the smallest singular value is below some threshold, this gives us a lower bound for $\|L^{-1}\|_{\max}$, the maximum number of paths, so that we can correctly reject graphs with too many paths. In case it is higher than the threshold, we know that L is poly-conditioned and we can proceed as in Theorem 1 to compute the number of paths between any pair of nodes exactly. However, in order for this approach to work, it is crucial that the graph is acyclic. Otherwise, the smallest singular value of L need not be related to the number of paths at all. We ensure this by first mapping the input graph to a layered graph that is guaranteed to be acyclic, similar as in [12]. We give the following definition.

▶ **Definition 6.** *Let $G = (V, E)$ be a directed graph with $|V| = n$ vertices. We define the layered graph* $\text{lay}(G)$ *on the vertex set $V' := V \times \{0, 1, ..., n\}$ with two types of edges:*
1. *For all edges $(i,j) \in E$ and all $l \leq n - 2$ add an edge from (i,l) to $(j, l+1)$ in $\text{lay}(G)$,*
2. *for all $i \in V$ and all $l \leq n - 1$ add an edge from (i,l) to (i,n) in $\text{lay}(G)$.*

It is easy to see that the paths in the first n layers of $\text{lay}(G)$ directly correspond to paths of length less than n in G. The last layer in $\text{lay}(G)$ just serves as a catch basin for all paths of different lengths. Let us quickly remark that counting paths in $\text{lay}(G)$ also allows us to detect cycles in G. To see this, note that there is a cycle in G through a node i if and only if $N((i,0),(i,n)) \geq 2$ in $\text{lay}(G)$. With this in mind, let us now prove the above theorem.

Proof of Theorem 4. Let $\langle G, s, t, 1^k\rangle$ be a given input graph instance with $|V(G)| = n$ nodes. Without loss of generality assume that $k \leq p(n)$ for some fixed polynomial $p : \mathbb{N} \to \mathbb{N}$. We first construct $\text{lay}(G)$ which is acyclic. Note that this is possible in AC^0. We consider

the counting Laplacian L of $\operatorname{lay}(G)$ and run Ta-Shma's spectrum approximation algorithm (compare [24, Theorem 5.2]) to approximate its singular values with error $\varepsilon = 1/6$ and accuracy $\delta = 1/(2n \cdot k)$. If we obtain a singular value smaller than δ, then we have with probability at least $5/6$,

$$2\delta = \frac{1}{n \cdot k} > \sigma_n(L) = \frac{1}{\|L^{-1}\|_2} \geq \frac{1}{n \cdot \|L^{-1}\|_{\max}} = \frac{1}{n \cdot \max_{i,j \in V(\operatorname{lay}(G))} N(i,j)}$$

which implies $\max_{i,j \in V(G)} N(i,j) \geq \max_{i,j \in V(\operatorname{lay}(G))} N(i,j) > k$. In this case we reject the input. Otherwise, if we do not obtain a singular value smaller than δ, then we know with the same probability that the counting Laplacian of $\operatorname{lay}(G)$ is poly-conditioned. Thus, we can proceed as in Theorem 1 and run Ta-Shma's matrix inversion algorithm to determine all entries of L^{-1} with total error $\varepsilon' = 1/6$ and accuracy $1/3$. For all $i \in V(G)$ we check in this way whether $L^{-1}((i,0),(i,n)) \geq 2$. If this is the case for some $i \in V(G)$, then there is a cycle in G containing i, i.e. $N(i,i) = \infty$ in G, and we reject the input. Otherwise, we further check whether all entries of L^{-1} are upper bounded by k and if $L^{-1}((s,0),(t,n)) \geq 1$. The former implies $N(i,j) \leq k$ for all $i,j \in V(G)$, and the latter implies $N(s,t) \geq 1$ in G. If both conditions are satisfied, we accept the input. Otherwise, we reject it. The total error probability of the algorithm is no higher than $\varepsilon + \varepsilon' = 1/3$. ◄

4 Counting few paths in quantum logspace

In this section we show how to push the algorithmic idea behind Theorem 1 to obtain Theorem 2, which shows how to count *st*-paths on graphs with a polynomial bound on the number of paths leaving s, and on the number of paths arriving in t, in $\mathsf{BQSPACE}(O(\log n))$. For this, we need the notion of an effective pseudoinverse, which we introduce in Section 4.1. The proof of Theorem 2 is in Section 4.2.

4.1 Effective pseudoinverse

A close look at Ta-Shma's matrix inversion algorithm shows that it can be easily altered to also handle ill-conditioned matrices as input and only invert them on their well-conditioned part. In order to appropriately state this observation we make the following definition.

▶ **Definition 7** (Effective pseudoinverse). *Let M be an $n \times n$ matrix with singular value decomposition $M = \sum_{j=1}^{n} \sigma_j |u_j\rangle \langle v_j|$. For $\zeta > 0$ we define the ζ-effective pseudoinverse of M as the matrix*

$$M_\zeta^+ := \sum_{\sigma_j \geq \zeta} \sigma_j^{-1} |v_j\rangle \langle u_j| .$$

Note that in the definition we essentially drop the largest terms of the actual inverse $M^{-1} = \sum_{j=1}^{n} \sigma_j^{-1} |v_j\rangle \langle u_j|$. While this produces significant error to approximate M^{-1} as a whole, we find that it can still give a good approximation for some relevant entries. We now state the refined version of Ta-Shma's matrix inversion algorithm which allows us to compute effective pseudoinverses of general matrices.

▶ **Theorem 8.** *Fix $\varepsilon(n), \zeta(n), \delta(n) > 0$ and $Z(n) \geq 1$. Let M be an $n \times n$ matrix such that $Z \geq \sigma_1(M) \geq ... \geq \sigma_n(M)$. There is an algorithm running in $\mathsf{BQSPACE}(O(\log \frac{nZ}{\varepsilon\delta}))$ that given M and two indices $s, t \in [n]$ outputs with probability $1 - \varepsilon$ an ε-additive approximation of the entry $M_{\widetilde{\zeta}}^+(s,t)$, where $\widetilde{\zeta}$ is a random value that is fixed at the beginning of the computation and is δ-close to ζ.*

While it is a simple modification of [24], for completeness we provide a proof of this theorem in Section A. The fact that we cannot control the exact threshold ζ of which singular values should be ignored during the effective pseudoinversion is a consequence of the inaccuracy of quantum phase estimation. This inaccuracy is limited by using Ta-Shma's consistent phase estimation procedure.

4.2 Proof of Theorem 2

We now have the necessary tools to prove our final result which we recall here:

▶ **Theorem 2.** *Fix a polynomial $p : \mathbb{N} \to \mathbb{N}$. Let G be a directed graph with $|V(G)| = n$ nodes such that*

- $\forall j \in V(G) : N(s, j) \leq p(n)$ and $N(j, t) \leq p(n)$.

There is an algorithm running in $\mathsf{BQSPACE}(O(\log n))$ that, given access to the adjacency matrix A of G and $s, t \in V(G)$, returns the number of paths from s to t.

The idea for the proof is to approximate the effective pseudoinverse entry of the counting Laplacian $L_{\tilde{\zeta}}^+(s, t)$ for some small enough $\tilde{\zeta} = 1/\text{poly}(n)$ instead of the actual inverse entry $L^{-1}(s, t)$. While ignoring the smallest singular values during the effective pseudoinversion normally leaves out the largest terms, we find that our path bounds imply low overlap of $\langle s | v_j \rangle$ and $\langle u_j | t \rangle$ for small singular values σ_j such that entry $L_{\tilde{\zeta}}^+(s, t)$ is close to $L^{-1}(s, t)$.

Proof. Without loss of generality we assume the graph to be acyclic. Otherwise, we proceed as in Section 3.2 and first map it to $\text{lay}(G)$ which is guaranteed to be acyclic. Now consider the counting Laplacian and its singular value decomposition $L = I - A = \sum_{j=1}^n \sigma_j |u_j\rangle \langle v_j|$. Its inverse is $L^{-1} = \sum_{j=1}^n \sigma_j^{-1} |v_j\rangle \langle u_j|$. Further, consider the vectors $(L^{-1})^T |s\rangle$ and $L^{-1} |t\rangle$. They contain as entries the number of paths starting in s and the number of paths ending in t, respectively. Hence, we find for their squared ℓ_2-norms:

$$\left\| (L^{-1})^T |s\rangle \right\|_2^2 = \left\| \sum_{j=1}^n \sigma_j^{-1} |u_j\rangle \langle v_j|s\rangle \right\|_2^2 = \sum_{j=1}^n \sigma_j^{-2} |\langle v_j|s\rangle|^2 \leq n \cdot p(n)^2 \qquad \text{and similarly}$$

$$\left\| L^{-1} |t\rangle \right\|_2^2 = \left\| \sum_{j=1}^n \sigma_j^{-1} |v_j\rangle \langle u_j|t\rangle \right\|_2^2 = \sum_{j=1}^n \sigma_j^{-2} |\langle u_j|t\rangle|^2 \leq n \cdot p(n)^2$$

where the last equalities of both lines follow because the $\{|u_j\rangle\}_{j\in[n]}$ and $\{|v_j\rangle\}_{j\in[n]}$ are orthonormal bases. As a consequence we obtain for each $j \in [n]$

$$|\langle s|v_j\rangle| \leq \sigma_j \sqrt{n} \cdot p(n) \quad \text{and} \quad |\langle u_j|t\rangle| \leq \sigma_j \sqrt{n} \cdot p(n).$$

Combining the two we find as a bound for the j-th term of the singular value decomposition of L^{-1},

$$\sigma_j^{-1} |\langle s|v_j\rangle \langle u_j|t\rangle| \leq \sigma_j n \cdot p(n)^2.$$

This allows us to estimate the error in computing an effective pseudoinverse entry $L_{\tilde{\zeta}}^+(s, t) = \sum_{\sigma_j \geq \tilde{\zeta}} \sigma_j^{-1} \langle s|v_j\rangle \langle u_j|t\rangle$ instead of the actual inverse entry $L^{-1}(s, t) = N(s, t)$. In fact, for $\tilde{\zeta} \leq \frac{1}{5n^2 \cdot p(n)^2}$ we have

$$\left| L^{-1}(s, t) - L_{\tilde{\zeta}}^+(s, t) \right| \leq \sum_{\sigma_j < \tilde{\zeta}} \sigma_j^{-1} |\langle s|v_j\rangle \langle u_j|t\rangle| < \tilde{\zeta} n^2 \cdot p(n)^2 \leq 1/5.$$

Choosing $Z = n$, $\varepsilon = 1/5$ and $\delta = \zeta = \frac{1}{10n^2 \cdot p(n)^2}$ in Theorem 8 ensures $\widetilde{\zeta} \leq \frac{1}{5n^2 \cdot p(n)^2}$ and yields an additive $2/5$ approximation of $L^{-1}(s,t)$ within the desired space complexity. Rounding to the closest integer gives the number of paths from s to t. ◀

A natural open question is whether the simultaneous polynomial bound on (i) the number of paths starting from s and on (ii) the number of paths ending in t is really necessary for a $\mathsf{BQSPACE}(O(\log n))$ procedure. Unfortunately, at least with our approach above, this seems to be the case. Note that (i) implies low overlap of $|s\rangle$ with the left singular vectors $|v_j\rangle$ of L^{-1} and (ii) implies low overlap of $|t\rangle$ with the right singular vectors $|u_j\rangle$ of L^{-1}. It turns out that if only one of the two is small, then the contribution of $\sigma_j^{-1} \langle s|v_j\rangle \langle u_j|t\rangle$ to $L^{-1}(s,t)$ can still be significant for very small σ_j but will be ignored in the pseudoinversion.

References

1   Romas Aleliunas, Richard M. Karp, Richard J. Lipton, Laszlo Lovasz, and Charles Rackoff. Random walks, universal traversal sequences, and the complexity of maze problems. In *20th Annual Symposium on Foundations of Computer Science*, pages 218–223, 1979. `doi:10.1109/SFCS.1979.34`.

2   Eric Allender. Guest column: Parting thoughts and parting shots (read on for details on how to win valuable prizes! *SIGACT News*, 54(1):63–81, March 2023. `doi:10.1145/3586165.3586175`.

3   Eric Allender and klaus-joern Lange. $\mathsf{RUSPACE}(\log n) \subseteq \mathsf{DSPACE}(\log^2 n/\log\log n)$. *Theory of Computing Systems*, 31, October 1998. `doi:10.1007/s002240000102`.

4   Sanjeev Arora and Boaz Barak. *Computational Complexity: A Modern Approach*. Cambridge University Press, USA, 1st edition, 2009.

5   Greg Barnes, Jonathan F. Buss, Walter L. Ruzzo, and Baruch Schieber. A sublinear space, polynomial time algorithm for directed s-t connectivity. *SIAM Journal on Computing*, 27(5):1273–1282, 1998. `doi:10.1137/S0097539793283151`.

6   Gerhard Buntrock, Birgit Jenner, Klaus-Jörn Lange, and Peter Rossmanith. Unambiguity and fewness for logarithmic space. In *Proceedings of the 8th International Symposium on Fundamentals of Computation Theory*, FCT '91, pages 168–179, Berlin, Heidelberg, 1991. Springer-Verlag. `doi:10.1007/3-540-54458-5_61`.

7   Stephen A. Cook. A taxonomy of problems with fast parallel algorithms. *Information and Control*, 64(1):2–22, 1985. International Conference on Foundations of Computation Theory. `doi:10.1016/S0019-9958(85)80041-3`.

8   Dean Doron, Amir Sarid, and Amnon Ta-Shma. On approximating the eigenvalues of stochastic matrices in probabilistic logspace. *Comput. Complex.*, 26(2):393–420, June 2017. `doi:10.1007/s00037-016-0150-y`.

9   Bill Fefferman and Cedric Yen-Yu Lin. A Complete Characterization of Unitary Quantum Space. In Anna R. Karlin, editor, *9th Innovations in Theoretical Computer Science Conference (ITCS 2018)*, volume 94 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 4:1–4:21, Dagstuhl, Germany, 2018. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. `doi:10.4230/LIPIcs.ITCS.2018.4`.

10 Bill Fefferman and Zachary Remscrim. Eliminating intermediate measurements in space-bounded quantum computation. In *Proceedings of the 53rd Annual ACM SIGACT Symposium on Theory of Computing*, STOC '21, pages 1343–1356. ACM, June 2021. `doi:10.1145/3406325.3451051`.

11 François Le Gall, Yupan Liu, and Qisheng Wang. Space-bounded quantum state testing via space-efficient quantum singular value transformation, 2024. `doi:10.48550/arXiv.2308.05079`.

12 Brady Garvin, Derrick Stolee, Raghunath Tewari, and N. Vinodchandran. Reachfewl = reachul. *Electronic Colloquium on Computational Complexity (ECCC)*, 18:60, August 2011. `doi:10.1007/s00037-012-0050-8`.

13   Uma Girish and Ran Raz. Eliminating intermediate measurements using pseudorandom generators, 2021. `arXiv:2106.11877`.

14   Uma Girish, Ran Raz, and Wei Zhan. Quantum Logspace Algorithm for Powering Matrices with Bounded Norm. In Nikhil Bansal, Emanuela Merelli, and James Worrell, editors, *48th International Colloquium on Automata, Languages, and Programming (ICALP 2021)*, volume 198 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 73:1–73:20, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. `doi:10.4230/LIPIcs.ICALP.2021.73`.

15   Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Physical Review Letters*, 103(15), October 2009. `doi:10.1103/physrevlett.103.150502`.

16   William M. Hoza. Better Pseudodistributions and Derandomization for Space-Bounded Computation. In Mary Wootters and Laura Sanità, editors, *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques (APPROX/RANDOM 2021)*, volume 207 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 28:1–28:23, Dagstuhl, Germany, 2021. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. `doi:10.4230/LIPIcs.APPROX/RANDOM.2021.28`.

17   Sampath Kannan, Sanjeev Khanna, and Sudeepa Roy. Stcon in directed unique-path graphs. In *FSTTCS*, pages 256–267, 2008. `doi:10.4230/LIPIcs.FSTTCS.2008.1758`.

18   Klaus-Jörn Lange. An unambiguous class possessing a complete set. In *Proceedings of the 14th Annual Symposium on Theoretical Aspects of Computer Science*, STACS '97, pages 339–350, Berlin, Heidelberg, 1997. Springer-Verlag. `doi:10.1007/BFB0023471`.

19   D.A. Levin, Y. Peres, and E.L. Wilmer. *Markov Chains and Mixing Times*. American Mathematical Soc., 2008. URL: `http://pages.uoregon.edu/dlevin/MARKOV/`.

20   Dieter van Melkebeek and Thomas Watson. Time-space efficient simulations of quantum computations. *Theory of Computing*, 8(1):1–51, 2012. `doi:10.4086/toc.2012.v008a001`.

21   Omer Reingold. Undirected connectivity in log-space. *J. ACM*, 55(4), September 2008. `doi:10.1145/1391289.1391291`.

22   Michael Saks and Shiyu Zhou. BPHSPACE(s) ⊆ DSPACE($s^{3/2}$). *Journal of Computer and System Sciences*, 58(2):376–403, 1999. `doi:10.1006/jcss.1998.1616`.

23   Walter J. Savitch. Relationships between nondeterministic and deterministic tape complexities. *Journal of Computer and System Sciences*, 4(2):177–192, 1970. `doi:10.1016/S0022-0000(70)80006-X`.

24   Amnon Ta-Shma. Inverting well conditioned matrices in quantum logspace. In *Proceedings of the Forty-Fifth Annual ACM Symposium on Theory of Computing*, STOC '13, pages 881–890, New York, NY, USA, 2013. Association for Computing Machinery. `doi:10.1145/2488608.2488720`.

A   Effective pseudoinversion in quantum logspace

The algorithm for the effective pseudoinversion is essentially identical to Ta-Shma's matrix inversion procedure except for a small change in the rotation step to appropriately treat ill-conditioned matrices. For completeness we describe it here but refer to [24] for the detailed complexity and error analysis.

Proof sketch of Theorem 8. We make the following two simplifying assumptions:

- Without loss of generality we assume that $Z = 1$. Otherwise, we simply choose $Z = n \cdot \|M\|_{\max}$ as an upper bound for $\sigma_1(M)$ and rescale the matrix with factor $1/Z$. Approximating the entries of a $\left(\frac{\tilde{\zeta}}{Z}\right)$-effective pseudoinverse of this rescaled matrix up to accuracy $\varepsilon \cdot Z$ gives an ε approximation of the entries of a $\tilde{\zeta}$-effective pseudoinverse of M within the same space complexity.

- Furthermore, we assume the matrix M to be Hermitian. Otherwise we use the well-known reduction to the Hermitian case

$$H = H(M) := \begin{bmatrix} 0 & M^\dagger \\ M & 0 \end{bmatrix} \qquad \text{with inverse} \qquad H^{-1} = \begin{bmatrix} 0 & M^{-1} \\ (M^\dagger)^{-1} & 0 \end{bmatrix}.$$

It is easily verified that the eigenpairs of H are given by $\{\pm\sigma_j, \frac{1}{\sqrt{2}}(|0\rangle |v_j\rangle \pm |1\rangle |u_j\rangle)\}_{j\in[n]}$, where σ_j and $|v_j\rangle, |u_j\rangle$ denote the singular values and vectors of M, respectively. We then find $M_\zeta^\pm(s,t) = H_\zeta^\pm(s, t+n)$.

Assuming the above, let the spectral decomposition of the matrix be given by $M = H = \sum_{j=1}^{n} \lambda_j |h_j\rangle \langle h_j|$ and let $|t\rangle = \sum_{j=1}^{n} \beta_j |h_j\rangle$. The algorithm works on four registers: An input register I, an estimation register E, a shift register S and an ancillary register A of dimension at least three.

1. We start by preparing the initial state

$$|t\rangle_I |0\rangle_E |0\rangle_S |\text{initial}\rangle_A = \sum_{j=1}^{n} \beta_j |h_j\rangle_I |0\rangle_E |0\rangle_S |\text{initial}\rangle_A.$$

2. We then apply the consistent phase estimation procedure (compare [24, Section 5.2]) to the input, estimation and shift register and obtain a state close to

$$\sum_{j=1}^{n} \beta_j |h_j\rangle_I |0\rangle_E |s(j)\rangle_S$$

where $s(j)$ is the *j-th section number*, that is a fixed classical value depending on h_j only, from which we can recover a δ-approximation $\widetilde{\lambda}_j = \widetilde{\lambda}(s(j))$ of the eigenvalue λ_j.

3. We next approximately apply a unitary map acting on the shift and ancillary register partially described by

$$|s\rangle_S |\text{initial}\rangle_A \mapsto \begin{cases} |s\rangle_S \left(\frac{\zeta}{\widetilde{\lambda}(s)} |\text{well}\rangle_A + \sqrt{1 - \left(\frac{\zeta}{\widetilde{\lambda}(s)}\right)^2} |\text{nothing}\rangle_A \right) & \text{if } \left|\widetilde{\lambda}(s)\right| \geq \zeta, \\ |s\rangle_S |\text{ill}\rangle_A & \text{if } \left|\widetilde{\lambda}(s)\right| < \zeta. \end{cases}$$

4. We reverse the consistent phase estimation and are left with a state close to

$$\sum_{\left|\widetilde{\lambda}_j\right| \geq \zeta}^{n} \beta_j |h_j\rangle_I \left(\frac{\zeta}{\widetilde{\lambda}_j} |\text{well}\rangle_A + \sqrt{1 - \left(\frac{\zeta}{\widetilde{\lambda}_j}\right)^2} |\text{nothing}\rangle_A \right) + \sum_{\left|\widetilde{\lambda}_j\right| < \zeta}^{n} \beta_j |h_j\rangle_I |\text{ill}\rangle_A.$$

Note that the approximations of the eigenvalues are monotone in the sense that $\lambda_i \leq \lambda_j$ implies $\widetilde{\lambda}_i \leq \widetilde{\lambda}_j$. From this we get the existence of $\widetilde{\zeta}_+ > 0$ and $\widetilde{\zeta}_- < 0$, both in absolute value δ-close to ζ, such that $\{j \in [n] : |\widetilde{\lambda}_j| \geq \zeta\} = \{j \in [n] : \lambda_j \geq \widetilde{\zeta}_+ \text{ or } \lambda_j \leq \widetilde{\zeta}_-\}$. Choosing and combining symmetric shifts for the positive and negative eigenvalues during the consistent phase estimation allows to assume $|\widetilde{\zeta}_+| = |\widetilde{\zeta}_-|$. Denoting this value by $\widetilde{\zeta}$ we find $\{j \in [n] : |\widetilde{\lambda}_j| \geq \zeta\} = \{j \in [n] : |\lambda_j| \geq \widetilde{\zeta}\}$.

5. Finally, we measure the ancillary register. If the measurement outcome is $|\text{well}\rangle_A$, this leaves us with a state close to the normalized desired one $\frac{1}{\left\| M_\zeta^\pm |t\rangle \right\|_2} M_\zeta^\pm |t\rangle$. In fact, estimating the success-probability of this measurement outcome gives a good approximation of

$$\zeta^2 \sum_{|\lambda_j| \geq \widetilde{\zeta}} \beta_j^2 \lambda_j^{-2} = \zeta^2 \left\| M_{\widetilde{\zeta}}^+ |t\rangle \right\|_2^2$$

from which we recover the norm $\left\| M_{\widetilde{\zeta}}^+ |t\rangle \right\|_2$.

Repeating the steps above sufficiently many times lets us create multiple copies of the final state close to $\frac{1}{\left\| M_{\widetilde{\zeta}}^+ |t\rangle \right\|_2} M_{\widetilde{\zeta}}^+ |t\rangle$. We then use Ta-Shma's space-efficient tomography procedure [24, Theorem 6.1] to approximately learn the state, and in particular entry $\frac{1}{\left\| M_{\widetilde{\zeta}}^+ |t\rangle \right\|_2} \langle s| M_{\widetilde{\zeta}}^+ |t\rangle$.

◀