Sublinear Estimation of a Single Element in Sparse Linear Systems

Nitin Shyamkumar, Siddhartha Banerjee, Peter Lofgren

***Abstract*— We present a bidirectional algorithm for estimating a single element in the solution of a linear system $Ax = b$, with sublinear average-case running time guarantees for sparse systems. Our work combines the von Neumann-Ulam scheme for solving linear systems with recent developments in bidirectional algorithms for estimating random-walk metrics. In particular, given a target additive-error threshold, we show how to combine a reverse local-variational technique with forward MCMC sampling such that the resulting algorithm is order-wise faster than each individual approach.**

I. Introduction

A. Related Work

Previous work has addressed the problem estimating a single component of a matrix equation through varying methods, including Monte Carlo methods. The Ulam-von Neumann algorithm shows that the inverse of B by defining $A = I - B$ and running random walks over the induced graph of A [**?**]. As long as the spectral norm of A is less than 1, the expectation of these random walks is exactly $(A)_{ij}$ where i is the start node and j is the end node. With these methods, one can solve the system $\mathbf{x} = G\mathbf{x} + \mathbf{z}$ provided the spectral norm $\rho(G) < 1$ since $\mathbf{x} = (I - G)^{-1}\mathbf{z}$, exactly the problem the Ulam-von Neumann algorithm solves. However, Monte Carlo style algorithms must have limited variance to be effective. [**?**] proved that there exist matrices G satisfying $\rho(G) < 1$ but with $||G||_\infty > 1$, and that convergence of the Ulam-von Neumann algorithm is not guaranteed on this class of matrices.

Our work is based on recently developments on bidirectional algorithms for estimating single-state transition probabilities in Markov chains. Such algorithms were first developed for *reversible Markov chains* using random-walk collision statistics; in particular, Kale et al. [**?**] proposed such a technique for estimating length-2ℓ random walk transition probabilities in a *regular undirected graph*. The main idea is that to test if a random walk goes from s to t in 2ℓ steps with probability $\geq \delta$, we can generate two independent random walks of length ℓ, starting from s and t respectively, and detect if they collide, i.e., terminate at the same intermediate node. The critical observation is that using $\sqrt{n}$ walks from s and t gives n potential collisions, which is sufficient to estimate probabilities on the order of $1/n$. This argument draws from older ideas on using the *birthday-paradox* in estimation problems [**?**]. A bidirectional algorithm for general graphs was first developed by Lofgren et al. [**?**] for PageRank estimation; the argument was subsequently simplified in [**?**], and extended to general Markov chains in [**?**]. Our work here further generalizes this line of work to computing single elements of powers of arbitrary matrices, with a particular application to solving for single elements in sparse linear systems.

N. Shyamkumar is with the Department of Computer Science, and S. Banerjee is with the School of Operations Research and Information Engineering, Cornell University, USA, Email: nhs56@cornell.edu, sbanerjee@cornell.edu. P. Lofgren is with the Department of Computer Science, Stanford University, USA, Email: plofgren@cs.stanford.edu.

II. Solving Linear Equations via Series Approximation

A. Basic Problem and Notation

Unless specified otherwise, we use boldface letters (e.g. $\mathbf{x}, \mathbf{y}$) to denote vectors in $\mathbb{R}^{n\times 1}$, and capital letters (e.g. A, Q) to denote matrices in $\mathbb{R}^{n\times n}$.

Given a linear system $\mathbf{y} = A\mathbf{x}$, where A is a positive definite matrix, our aim is to estimate $x[t]$ for some given index $t \in [n]$. This can be done by directly solving for $\mathbf{x}$; however, we are interested in settings where n is very large, and hence direct solution techniques may be impractical.

B. Approximation via the Truncated Neumann Series

One approach for approximating the solution to $\mathbf{y} = A\mathbf{x}$ is to expand it via the Neumann series and then compute the leading terms of the summation. In particular, if A is positive definite, we can find γ such that $G = I - \gamma A$ satisfies $\rho(G) < 1$. Then $\gamma\mathbf{y} = (I - (I - \gamma A)\mathbf{x}$.

Now let us examine this new system with $\mathbf{z} = \gamma\mathbf{y}$ and $G = I - \gamma A$. We obtain $\mathbf{z} = (I - G)\mathbf{x}$ and $\mathbf{x} = G\mathbf{x} + \mathbf{z}$. Since we ensure $\rho(G) < 1$, we can write $\mathbf{x}$ as a von Neumann series: $\mathbf{x} = \sum_{k=0}^{\infty} G^k\mathbf{z}$. Thus to find the t component of the solution vector $\mathbf{x}$, or $\mathbf{x}[t]$, we perform the operation $\langle \mathbf{x}, \mathbf{e}_t\rangle = \sum_{k=0}^{\infty} \langle G^k\mathbf{z}, \mathbf{e}_t\rangle$. Similar transformations have been used in [**?**], [**?**], [**?**].

Computing $\mathbf{x}[t]$ then amounts to computing $\mathbf{p}_{\mathbf{z}}^{\ell}[t] := \langle G^\ell\mathbf{z}, \mathbf{e}_t\rangle = \langle \mathbf{z}, (G^T)^\ell\mathbf{e}_t\rangle$ for any ℓ and taking their sum for some $\ell \in \{0, \ldots, \ell_{\max}\}$ where $\ell_{\max}$ is a finite term truncating the power series. Let $Q := G^T$; prior work by Banerjee and Lofgren [**?**] shows how to compute $\mathbf{p}_{\mathbf{z}}^{\ell}[t]$ for the special case where $\mathbf{z}$ is a probability vector and Q is a *stochastic matrix* (i.e., with all nonnegative entries, and each row summing to 1). We now extend this result for any $\mathbf{z}$ and a special class of matrices Q.

Note that the error from truncating the series to $\ell_{\max}$ can be determined a priori; using bounds for $\mathbf{z}$ and the condition that G has spectral norm less than 1, we have the following Lemma.

Lemma 1: Set $\ell_{\max} \geq \frac{1}{\ln \rho(G)} \ln\left(\frac{\Delta(1-\rho(G))}{||\mathbf{z}||}\right)$ to bound the series truncation error by Δ.

Proof: Let ε be the error from truncating the power series to $\ell_{\max}$. Then by definition:

$$\begin{aligned}\varepsilon &= \left|\sum_{\ell=0}^{\infty}\left\langle \mathbf{z}, Q^{\ell}\mathbf{e}_t\right\rangle - \sum_{\ell=0}^{\ell_{\max}}\left\langle \mathbf{z}, Q^{\ell}\mathbf{e}_t\right\rangle\right| \\ &= \left|\sum_{\ell=\ell_{\max}+1}^{\infty}\left\langle \mathbf{z}, Q^{\ell}\mathbf{e}_t\right\rangle\right| \\ &= \left|\left\langle \mathbf{z}, Q^{\ell_{\max}+1}\sum_{\ell=0}^{\infty} Q^{\ell}\mathbf{e}_t\right\rangle\right|\end{aligned}$$

For any inner product of the form $\left\langle \mathbf{z}, Q^{\ell}\mathbf{e}_t\right\rangle$, by Cauchy - Schwarz we have $|\left\langle \mathbf{z}, Q^{\ell}\mathbf{e}_t\right\rangle| \leq ||\mathbf{z}|| \cdot ||Q^{\ell}\mathbf{e}_t|| \leq ||\mathbf{z}||\rho(Q)^{\ell}$. Hence we have:

$$\begin{aligned}\varepsilon &\leq ||\mathbf{z}||\rho(Q)^{\ell_{\max}+1}\left(\sum_{\ell=0}^{\infty}\rho(Q)^{\ell}\right) \\ &= ||\mathbf{z}||\rho(Q)^{\ell_{\max}+1}\left(\frac{1}{1-\rho(Q)}\right)\end{aligned}$$

Now suppose we want our error less than Δ, that is $\varepsilon \leq \Delta$. Then we can set the upper bound for ϵ to Δ and solve for $\ell_{\max}$:

$$\begin{aligned}||\mathbf{z}||\rho(Q)^{\ell_{\max}+1}\left(\frac{1}{1-\rho(Q)}\right) &= \Delta \\ (\ell_{\max}+1)\ln\rho(Q) &= \ln\left(\frac{\Delta(1-\rho(Q))}{||\mathbf{z}||}\right) \\ \ell_{\max} &= \frac{1}{\ln\rho(Q)}\ln\left(\frac{\Delta(1-\rho(Q))}{||\mathbf{z}||}\right) - 1\end{aligned}$$

Thus, if we take $\ell_{\max} \geq \frac{1}{\ln\rho(Q)}\ln\left(\frac{\Delta(1-\rho(Q))}{||\mathbf{z}||}\right)$, we will have an error of at most Δ when approximating the series. ■

Since we can a priori bound the error resulting from truncating the von Neumann series to $\ell_{\max}$, we will focus on the problem of estimating $\mathbf{p}^{\ell}_{\mathbf{z}}[t]$. It is then straightforward to develop overall error bounds by combining the results of the additive truncation error Δ and the relative error guarantees we prove for our bidirectional algorithm in Theorem 1.

III. A Bidirectional Algorithm for Computing Matrix Powers

We now describe our algorithm for computing an estimate $\hat{\mathbf{p}}^{\ell}_{\mathbf{z}}[t]$ for $\mathbf{p}^{\ell}_{\mathbf{z}}[t] = \left\langle \mathbf{z}, Q^{\ell}\mathbf{e}_t\right\rangle$. To do so, we first describe two existing algorithms – a forward MCMC technique based on the von Neumann-Ulam scheme [?], [?], and a variational method based on a natural local dynamic-programming update, proposed by Andersen et al. [?] for computing PageRank, and used by Lee et al. [?] in this setting. We present these along with statements of their correctness and running time – some of these results follow directly from previous work (as we note in the appropriate sections), and their proofs are included here mainly for the sake of completeness.

Our main contribution in this work is to show how the above two primitives can be combined into a bidirectional algorithm for estimating $\hat{\mathbf{p}}^{\ell}_{\mathbf{z}}[t]$ for any given matrix Q. Our algorithm follows the general structure proposed by Lofgren et al. [?], [?] for PageRank and Markov Chain transition probability estimation. It comprises of two distinct components: first we use the reverse local-DP primitive to estimate approximate values of $\left(Q^{\ell}e_t\right)[i]$ for all steps $\ell \in [\ell_{\max}]$ and $i \in [n]$. We then use MCMC samples to reduce the error in these estimates to get our desired accuracy.

We first introduce some notation which will help us better describe the algorithm. Drawing parallels to the case where Q is a stochastic matrix and $\mathbf{z}$ an element in the n-dimensional simplex (as in [?]), we define a (weighted) directed graph $\mathcal{G}_Q(\mathcal{V},\mathcal{E})$ with states $\mathcal{V} = [n]$, and edges $(i,j) \in \mathcal{E}$ if $Q_{ij} \neq 0$. Each edge $(i,j) \in \mathcal{E}$ has an associated weight $w_{ij} \in \mathbb{R}$, which we describe later. We refer to the label for a node $v \in V$ (i.e., a dimension $v \in [n]$) as a *dimension-index*, and the exponent of Q as the *step-index*. We also use $\mathbf{e}_v$ denote the indicator for index v (i.e., $\mathbf{e}_v[i] = \mathbb{1}_{i=v}$). Finally, we define Q^{+} to denote the matrix with $Q^{+}_{ij} = |Q_{ij}|$. Many of our bounds depend on the norm $||Q^{+}||_{\infty} = ||Q||_{\infty}$, i.e., the maximum (absolute) row-sum of Q; for ease of notation, we henceforth define $\beta = ||Q||_{\infty}$.

A. Computing Matrix Powers via Iterative Local-Update

One approach towards estimating $\mathbf{p}^{\ell}_{\mathbf{z}}[t]$ is via a standard power iteration for computing $Q^{\ell}e_t$. In settings where n is large enough such that a direct power iteration is infeasible, one can use a 'local' power iteration, which essentially corresponds to a natural dynamic programming update. Informally, the algorithm estimates $\mathbf{p}^{\ell}_{\mathbf{z}}[t]$ by starting off with a mass of 1 on dimension-index t, and then 'pushing' this mass in reverse along the edges of graph G_Q.

To describe this REVERSE-LOCAL-UPDATE algorithm, we first define a REVERSE-PUSH operation corresponding to the DP iteration. This is directly adapted from the algorithm in Andersen [?] for the personal PageRank problem (and more generally for a stochastic matrix Q); however, it is straightforward to show that the invariant that holds for any matrix Q (in fact, it does not require Q to be full rank, as we have assumed).

The REVERSE-PUSH operation is a standard dynamic programming iteration which is used to compute an estimate $\hat{\mathbf{p}}^{\ell}_{\mathbf{z}}[t]$ by proceeding 'in reverse' from t. Essentially,REVERSE-PUSH is a local power-iteration for computing $Q^{\ell}\mathbf{e}_t$; instead of performing a full power-iteration, it adaptively exploits any sparsity in the computation. This operation was defined in the form given below in [?], and subsequently used as a primitive in [?], [?].

For each step-index $\ell \in \{0,1,\ldots,\ell_{\max}\}$, we store two vectors: the *estimate vector* $\mathbf{q}^{\ell}_t$ and the *residual vector* $\mathbf{r}^{\ell}_t$. We initialize all $\mathbf{r}^{\ell}_t, \mathbf{q}^{\ell}_t, \ell \in [\ell_{\max}]$ to 0, except for $\mathbf{r}^{0}_t$, which we set to $\mathbf{e}_t$. Now, given any dimension-index $v \in [n]$ and step-index $\ell \in [\ell_{\max}]$, the REVERSE-PUSH operation iteratively updates these vectors as follows:

Algorithm 1 REVERSE-PUSH(t, v, ℓ)

Inputs: Matrix Q, estimates $\mathbf{q}_t^\ell$, residuals $\mathbf{r}_t^\ell, \mathbf{r}_t^{\ell+1}$

1: **return** New estimates $\widetilde{\mathbf{q}}_t^\ell$ and residuals $\widetilde{\mathbf{r}}_t^\ell$ computed as:

$$\widetilde{\mathbf{q}}_t^\ell \leftarrow \mathbf{q}_t^\ell + \left\langle \mathbf{r}_t^\ell, \mathbf{e}_v \right\rangle \mathbf{e}_v$$
$$\widetilde{\mathbf{r}}_t^\ell \leftarrow \mathbf{r}_t^\ell - \left\langle \mathbf{r}_t^\ell, \mathbf{e}_v \right\rangle \mathbf{e}_v$$
$$\widetilde{\mathbf{r}}_t^{\ell+1} \leftarrow \mathbf{r}_t^{\ell+1} + \left\langle \mathbf{r}_t^\ell, \mathbf{e}_v \right\rangle (Q\mathbf{e}_v)$$

The REVERSE-PUSH iteration results in the following critical invariant for the estimate and residual vectors:

Lemma 2: Given the initialization described above, after any sequence of REVERSE-PUSH operations, and for any $\mathbf{z} \in \mathbb{R}^n$ and $\ell \geq 0$, the estimates $\{\mathbf{q}_t^k\}$ and residuals $\{\mathbf{r}_t^k\}$ satisfy the following invariant:

$$\mathbf{p}_\mathbf{z}^\ell[t] = \left\langle \mathbf{z}, \mathbf{q}_t^\ell \right\rangle + \sum_{k=0}^{\ell} \left\langle \mathbf{z}, Q^k \mathbf{r}_t^{\ell-k} \right\rangle = \left\langle \mathbf{z}, \mathbf{q}_t^\ell + \sum_{k=0}^{\ell} Q^k \mathbf{r}_t^{\ell-k} \right\rangle$$

The above invariant was first stated in [**?**] for the case of PageRank vectors. For the sake of completeness, we present a proof below for general matrices, adapted from [**?**]; an identical invariant is given in [**?**].

Proof: For our chosen initialization (i.e., $\mathbf{r}_t^0 = \mathbf{e}_t$, and all other estimate and residual vectors set to 0), the invariant simplifies to $\mathbf{p}_\mathbf{z}^\ell[t] = \left\langle \mathbf{z}, Q^\ell \mathbf{e}_t \right\rangle$ which is true by definition. Now, assuming the invariant holds at any stage with vectors $\{\mathbf{q}_t^\ell\}, \{\mathbf{r}_t^\ell\}_{\ell \in [\ell_{\max}]}$, and let $\{\widetilde{\mathbf{q}}_t^\ell\}, \{\widetilde{\mathbf{r}}_t^\ell\}_{\ell \in [\ell_{\max}]}$ be the new vectors after executing a REVERSE-PUSH(t, v, k) operation for any given $k \in [\ell_{\max}]$ and $\mathbf{z} \in \mathbb{R}^n$. We define:

$$\Delta_v^k = \left(\tilde{\mathbf{q}}_t^\ell + \sum_{i=0}^{\ell} (Q^i) \widetilde{\mathbf{r}}_t^{\ell-i} \right) - \left(\mathbf{q}_t^\ell + \sum_{i=0}^{\ell} (Q^i) \mathbf{r}_t^{\ell-i} \right)$$

Now to show that the invariant holds following REVERSE-PUSH(t, v, k), it suffices to show that Δ_v^k is zero for any $v \in V$ and $k \in [\ell_{\max}]$.

We now have three cases: (i) if $\ell < k$, then the REVERSE-PUSH(t, v, k) operation does not affect the residual or estimate vectors $\{\mathbf{q}_t^i, \mathbf{r}_t^i\}_{i<k}$, and hence $\Delta_v^k = 0$; (ii) If $\ell = k$, we have:

$$\begin{aligned}\Delta_v^k &= \left(\tilde{\mathbf{q}}_t^k + \tilde{\mathbf{r}}_t^k\right) - \left(\mathbf{q}_t^k + \mathbf{r}_t^k\right) \\ &= \mathbf{q}_t^k + \left\langle \mathbf{r}_t^k, \mathbf{e}_v \right\rangle \mathbf{e}_v + \mathbf{r}_t^k - \left\langle \mathbf{r}_t^k, \mathbf{e}_v \right\rangle \mathbf{e}_v - \mathbf{q}_t^k - \mathbf{r}_t^k = 0\end{aligned}$$

(iii) Finally, when $\ell > k$, we have:

$$\begin{aligned}\Delta_v^k &= Q^{\ell-k} \left(\tilde{\mathbf{r}}_t^k - \mathbf{r}_t^k\right) + Q^{\ell-k-1} \left(\tilde{\mathbf{r}}_t^{k+1} - \mathbf{r}_t^{k+1}\right) \\ &= -\left\langle \mathbf{r}_t^k, \mathbf{e}_v \right\rangle Q^{\ell-k} \mathbf{e}_v + \left\langle \mathbf{r}_t^k, \mathbf{e}_v \right\rangle Q^{\ell-k-1} (Q\mathbf{e}_v) = 0\end{aligned}$$

Hence we have shown that the invariant is preserved for any sequence of reverse push operations. ■

The above invariant gives a natural iterative algorithm for computing $\mathbf{p}_\mathbf{z}^\ell[t]$, by performing repeated REVERSE-PUSH operations and controlling the residual vectors $\mathbf{r}_t^k$, and using $\hat{\mathbf{p}}_\mathbf{z}^\ell[t] = \langle \mathbf{z}, \mathbf{q}_t^\ell \rangle$ as the estimate. Depending on the norm we choose to control, we can get a bound for the error via Hölder's inequality. In particular, controlling the infinity norm (i.e., the maximum absolute value of the residual vectors) to be less than some chosen $\delta_r > 0$ gives us a bound: $|\mathbf{p}_\mathbf{z}^\ell[t] - \langle \mathbf{z}, \mathbf{q}_t^\ell \rangle| \leq ||x||_1 \delta_r \beta^\ell$ [1].

Algorithm 2 REVERSE-LOCAL-UPDATE$(t, Q, \ell_{\max}, \delta_r)$

Inputs: Matrix Q, maximum step-index $\ell_{\max}$, target residual threshold δ_r

1: Initialize all residual $\mathbf{r}_t^\ell$ and estimate vectors $\mathbf{q}_t^\ell, \ell \in [\ell_{\max}]$ to 0; set $\mathbf{r}_t^0 = \mathbf{e}_t$
2: **for** $\ell \in \{0, 1, 2, ...\ell_{\max}\}$ **do**
3:   **while** $\exists v$ such that $\left|\mathbf{r}_t^\ell[v]\right| > \delta_r$ **do**
4:     REVERSE-PUSH(t, v, ℓ)
5:   **end while**
6: **end for**
7: **return** $\{\mathbf{q}_t^\ell\}, \{\mathbf{r}_t^\ell\}_{\ell \in [\ell_{\max}]}$

Finally, we want to bound the running time of REVERSE-LOCAL-UPDATE$(t, Q, \ell_{\max}, \delta_r)$. It is easy to see that in the worst case, the running time can be as much as the $\ell_{\max}$-hop in-neighborhood of t in Q. However, for a *uniform random* choice of t, we can obtain a more informative bound. Recall we define $\beta = ||Q||_\infty$. Now we have the following:

Lemma 3: For any $Q \in \mathbb{R}^{n \times n}$ and uniform random dimension-index $t \in [n]$, the expected running time of REVERSE-LOCAL-UPDATE$(t, Q, \ell_{\max}, \delta_r)$ is

$$O\left(\frac{nnz(Q)}{n\delta_r}(\ell_{\max}+1)\beta^{\ell_{\max}}\right)$$

In particular, note that if $||Q||_\infty \leq 1$ and $\ell_{\max} = O(1)$, then the average running time is $O\left(\frac{nnz(Q)}{n\delta_r}\right)$.

Proof: Let $T(t)$ be the running time of REVERSE-LOCAL-UPDATE$(t, Q, \ell_{\max}, \delta_r)$. Recall we define Q^+ as the matrix with $Q_{ij}^+ = |Q_{ij}|$; let $\hat{T}(t)$ be the running time of REVERSE-LOCAL-UPDATE$(t, Q^+, \ell_{\max}, \delta_r)$. Then we have that for every matrix Q and every t, we have $\hat{T}(t) > T(t)$ – this follows from the fact that any cancellation between positive and negative residuals in REVERSE-LOCAL-UPDATE$(t, Q^+, \ell_{\max}, \delta_r)$ can only decrease the number of iterations. Also, note that under Q^+, since all residuals are positive, we have that for any $\ell \leq \ell_{\max}, v \in [n]$, the residuals satisfy $\mathbf{r}_t^\ell[v] \leq \left(\mathbf{e}_v^T Q^\ell\right)[t]$.

Now let $d_i := \sum_j \mathbb{1}_{\{Q_{ij} \neq 0\}}$, i.e., the support of i^{th} row in Q, and $\mathbf{r}_t^\ell$ denote the residuals under REVERSE-LOCAL-UPDATE$(t, Q^+, \ell_{\max}, \delta_r)$. Recall we define $\beta = ||Q||_\infty$.From Algorithm 2, we have $\hat{T}(t) = \sum_{\ell=0}^{\ell_{\max}} \sum_{v \in [n]} \mathbb{1}_{\mathbf{r}_t^\ell[v] > \delta_r}$. Thus, the expected running time

[1]This approach was used in [**?**], [**?**], [**?**]; an alternative is to control $||r_t^k||_2$ giving error bounds in terms of $||x||_2$, which was suggested in [**?**].

over a uniform random choice of $t \in [n]$ is given by

$$\begin{aligned}
\frac{1}{n}\sum_t \hat{T}(t) &= \frac{1}{n}\sum_{t\in[n]}\sum_{\ell=0}^{\ell_{\max}}\sum_{v\in[n]} \mathbb{1}_{\{\mathbf{r}_t^\ell > \delta_r\}} d_v \\
&= \frac{1}{n}\sum_{\ell=0}^{\ell_{\max}}\sum_{v\in[n]}\sum_{t\in[n]} \mathbb{1}_{\{\mathbf{r}_t^\ell > \delta_r\}} d_v \\
&\leq \frac{1}{n}\sum_{\ell=0}^{\ell_{\max}}\sum_{v\in[n]} \mathbb{1}_{\{(\mathbf{e}_w^T(Q^+)^\ell)[t] > \delta_r\}} d_v \\
&= \frac{1}{n}\sum_{\ell=0}^{\ell_{\max}}\sum_{v\in[n]} \frac{||\mathbf{e}_w^T(Q^+)^\ell||_1}{\delta_r} d_v \\
&\leq \frac{1}{n}\sum_{\ell=0}^{\ell_{\max}}\sum_{v\in[n]} \frac{||Q^+||_\infty^\ell}{\delta_r} d_v \\
&\leq (\ell_{\max}+1)\,\beta^{\ell_{\max}}\frac{nnz(Q)}{n\delta_r}
\end{aligned}$$

■

B. Computing Matrix Powers via MCMC Sampling

In the previous section, we computer $\mathbf{p}_\mathbf{z}^\ell[t]$ by working backwards from t. Note that our final algorithm is independent of $\mathbf{z}$. We now present an alternate technique which is based on a forward MCMC sampling technique called the von Neumann-Ulam scheme. Note that in this case, the algorithm starts from $\mathbf{z}$, and computes $\mathbf{p}_\mathbf{z}^\ell[t]$ for all $t \in [n]$.

More generally, given any vectors $\mathbf{a}$ and $\mathbf{b}$ and matrix Q, the von Neumann-Ulam scheme can be used for computing $\langle \mathbf{a}, Q^k\mathbf{b}\rangle$. To understand the algorithm, note that we can expand $\langle \mathbf{a}, Q^k\mathbf{b}\rangle$ as the sum $\sum_{(v_0,\dots,v_k)\in V^k}\left(\prod_{j\in[k]} Q_{v_{j-1}v_j}\right)\mathbf{a}[v_0]\mathbf{b}[v_k]$. Now we can interpret this sum as an expectation over a k-step random walk $W = (V_0, V_1, \dots, V_k)$ on $\mathcal{G}$, specified as follows:

- V_0, the starting node for the random walk, is sampled from $\sigma_\mathbf{a} = \{|\mathbf{a}[i]|/||\mathbf{a}||_1\}_{i\in[n]}$, and has an associated weight $w_{V_0} = sign(\mathbf{a}[V_0])||\mathbf{a}||_1$.
- The transition probability matrix for the walk is given by $P = \{|Q_{ij}|/||Q_i||_1\}$ (where $||Q_i||_1$ is the 1-norm of the i^{th} row of Q).
- Each edge $(i,j) \in E$ has associated weight $w_{ij} = sgn(Q_{ij})||Q_i||_1$.
- The 'score' for a walk W is the product of weights of traversed edges, i.e.,

$$S_t^k(W) = w_{V_0}\prod_{i=0}^{k-1} w_{V_iV_{i+1}}\mathbf{b}[V_k]$$

This procedure is summarized in Algorithm 3.

Recall we define $\beta = \max\{1, ||Q||_\infty\}$. Now we have the following Lemma.

Lemma 4: MCMC-SAMPLER$(Q, k, \mathbf{a}, \mathbf{b})$ returns a walk-score $S_t^k(W)$ which satisfies:

1) $\mathbb{E}_{W\sim\sigma_\mathbf{a}(P)^k}\left[S_t^k(W)\right] = \langle \mathbf{a}, Q^k\mathbf{b}\rangle$
2) $S_t^k(W) \in [\,-\beta^k||\mathbf{a}||_1||\mathbf{b}||_\infty,\ \beta^k||\mathbf{a}||_1||\mathbf{b}||_\infty\,]$

Algorithm 3 MCMC-SAMPLER$(Q, k, \mathbf{a}, \mathbf{b})$

Inputs: Matrix Q, exponent k, vectors $\mathbf{a}$ and $\mathbf{b}$

Construct transition matrix $P = \{|Q_{ij}|/||Q_i||_1\}$ and starting measure $\sigma_a = \{|\mathbf{a}[i]|/||\mathbf{a}||_1\}_{i\in V}$

Define node weights $w_{V_0} = sign(\mathbf{a}[V_0])||\mathbf{a}||_1$ and edge weights $w_{V^iV^j} = sign(Q_{V^iV^j})||Q_i||_1$

Construct source distribution $\sigma_\mathbf{a}$ with $\sigma_\mathbf{a}[i] = \frac{\mathbf{z}[i]}{||\mathbf{z}||_1}$

Sample $V^0 \sim \sigma_a$, and generate a random walk $W = \{V^0, V^1, \dots, V^k\}$ of length k on $\mathcal{G}$ using transition probability P.

return Walk-score $S_t^k(W) = w_{V_0}\prod_{i=0}^{k-1} w_{V_iV_{i+1}}\mathbf{b}[V_k]$

Proof: We can expand $\mathbb{E}_{W\sim\sigma_\mathbf{a}(P)^k}\left[S_t^k(W)\right]$ to obtain:

$$\begin{aligned}
&\mathbb{E}_{W\sim\sigma_\mathbf{a}(P)^k}\left[S_t^k(W)\right] \\
&= \sum_{(V^0,\dots V^k)\in[n]^{k+1}}\left(\frac{sign(\mathbf{a}[V_0])||\mathbf{a}||_1|\mathbf{a}[V^0]|}{||\mathbf{a}||_1}\right. \\
&\times\left.\left(\prod_{j\in[0,k-1]}\frac{sign(Q_{V^jV^{j+1}}||Q_{V^j}||_1)|Q_{V^jV^{j+1}}|}{||Q_{V^j}||_1}\right)\mathbf{b}[V^k]\right) \\
&= \sum_{(V^0,\dots V^k)\in[n]^{k+1}}\left(\mathbf{a}[V_0]\left(\prod_{j\in[0,k-1]} Q_{V^jV^{j+1}}\right)\mathbf{b}[V^k]\right)
\end{aligned}$$

The final expression is exactly the definition of $\langle \mathbf{a}, Q^k\mathbf{b}\rangle$; hence $\mathbb{E}_{W\sim\sigma_\mathbf{a}(P)^k}\left[S_t^k(W)\right] = \langle \mathbf{a}, Q^k\mathbf{b}\rangle$.

Next, in order to bound the score $S_t^k(W)$, recall that

$$S_t^k(W) = w_{V_0}\prod_{i=0}^{k-1} w_{V_iV_{i+1}}\mathbf{b}[V_k]$$

We defined $w_{V_0} = sign(\mathbf{a}[V_0])||\mathbf{a}||_1$, so trivially $|w_{V_0}| \leq ||\mathbf{a}||_1$. Additionally, $w_{V_iV_{i+1}} = sign(Q_{V^iV^j})||Q_i||_1$ and by definition $||Q_i||_1 \leq ||Q||_\infty$ and $|\mathbf{b}[V^k]| \leq ||\mathbf{b}||_\infty$. Hence, since we are multiplying k edge scores, we finally obtain $|S_t^k(W)| \leq ||Q||_\infty^k||\mathbf{a}||_1||\mathbf{b}||_\infty$ as stated in the lemma. ■

C. A Bidirectional Algorithm for Computing Matrix Powers

Finally, we can present our bidirectional estimator for $\mathbf{p}_\mathbf{z}^\ell[t] = \langle \mathbf{z}, Q^\ell e_t\rangle$. Our algorithm combines the REVERSE-LOCAL-UPDATE and MCMC-SAMPLER algorithms, following the technique established by Banerjee and Lofgren [?].

Intuitively, combining the two algorithms allows us to perform sampling more effectively by reducing the variance in the walk scores. To achieve this, given a desired error threshold δ, we first run REVERSE-LOCAL-UPDATE(t, Q, ℓ, δ_r) for an appropriately chosen $\delta_r \gg \delta$ (cf. Theorem ??). At this point, from Lemma 3, we know that we have $\mathbf{p}_\mathbf{z}^\ell[t] = \langle \mathbf{z}, \mathbf{q}_t^\ell\rangle + \sum_{k=0}^{\ell}\langle \mathbf{z}, Q^k\mathbf{r}_t^{\ell-k}\rangle$, with $\mathbf{r}_t^k[v] \leq \delta_r$ for all $v \in [n], k \leq \ell$. Now, instead of ignoring the residual terms (as we suggest in III-A; this is the approach used in [?], [?]), we can further reduce our error by using MCMC-SAMPLER$(Q, k, \mathbf{z}, \mathbf{r}_t^k)$ to estimate the residual $\langle \mathbf{z}, \mathbf{r}_t^k\rangle$ for all $k \leq \ell$. Note however that this is better than directly executing MCMC-SAMPLER$(Q, \ell, \mathbf{z}, e_t)$, as the residuals have much smaller magnitude than 1, and hence the walk-scores have lower variance.

Algorithm 4 BIDIR-MATRIX-POWER$(Q, \mathbf{z}, t, \ell_{\max})$

Inputs: Matrix Q, vector $\mathbf{z}$, target node-index t.

```
Compute estimate and residual vectors via {r_t^ℓ}, {q_t^ℓ} = REVERSE-LOCAL-UPDATE(t, Q, ℓ_max, δ_r)
for l ∈ {1, 2, ..., ℓ_max} do
    for i ∈ [n_f] do
        k ~ Unif[0, ℓ]
        S_{i,t}^ℓ = MCMC-SAMPLER (Q, k, z, ℓ · r_t^{ℓ-k})
    end for
    p_z^ℓ[t] = ⟨z, q_t^ℓ⟩ + (1/n_f) Σ_{i=0}^{n_f} S_{i,t}^ℓ
end for
return Σ_{ℓ=0}^{ℓ_max} p_z^ℓ[t]
```

Lemma 5: BIDIR-MATRIX-POWER computes an unbiased estimator of $\mathbf{p}_{\mathbf{z}}^{\ell}[t]$

$$\mathbf{p}_{\mathbf{z}}^{\ell}[t] = \left\langle \mathbf{z}, Q^{l}\mathbf{e}_t \right\rangle = \left\langle \mathbf{z}, \mathbf{q}_t^{\ell} \right\rangle + \sum_{k=0}^{\ell} \left\langle \mathbf{z}, Q^{k}\mathbf{r}_t^{\ell-k} \right\rangle$$
$$= \mathbb{E}\left[\hat{\mathbf{p}}_{\mathbf{z}}^{\ell}[t]\right]$$

Proof: By definition of the estimator $\hat{\mathbf{p}}_{\mathbf{z}}^{\ell}[t]$ and linearity of the expectation operator:

$$\mathbb{E}\left[\hat{\mathbf{p}}_{\mathbf{z}}^{\ell}[t]\right] = \left\langle \mathbf{z}, \mathbf{q}_t^{\ell} \right\rangle + \frac{1}{n_f}\sum_{i=1}^{n_f} \mathbb{E}_{k\sim Unif[0,l]}\left[S_t^k\right]$$
$$= \left\langle \mathbf{z}, \mathbf{q}_t^{\ell} \right\rangle + \mathbb{E}_{k\sim Unif[0,\ell]}\left[S_t^k\right]$$

From Lemma 2, we know $\mathbb{E}\left[S_t^k\right] = \left\langle \mathbf{a}, Q^k\mathbf{b} \right\rangle$ when we call MCMC-SAMPLER$(Q, k, \mathbf{a}, \mathbf{b})$, so in the linear system estimator, we obtain:

$$\mathbb{E}\left[\hat{\mathbf{p}}_{\mathbf{z}}^{\ell}[t]\right] = \left\langle \mathbf{z}, \mathbf{q}_t^{\ell} \right\rangle + \mathbb{E}_{k\sim Unif[0,\ell]}\left[\left\langle \mathbf{z}, Q^k(\ell \cdot \mathbf{r}_t^{\ell-k}) \right\rangle\right]$$
$$= \left\langle \mathbf{z}, \mathbf{q}_t^{\ell} \right\rangle + \frac{1}{\ell}\sum_{k=0}^{\ell} \left\langle \mathbf{z}, Q^k(l \cdot \mathbf{r}_t^{\ell-k}) \right\rangle$$
$$= \left\langle \mathbf{z}, \mathbf{q}_t^{\ell} \right\rangle + \sum_{k=0}^{l} \left\langle \mathbf{z}, Q^k\mathbf{r}_t^{\ell-k} \right\rangle$$

Recall from Lemma 1, that for any matrix Q and after any sequence of reverse push operations, $\mathbf{p}_{\mathbf{z}}^{\ell}[t] = \left\langle \mathbf{z}, Q^{\ell}\mathbf{e}_t \right\rangle$ obeys the invariant:

$$\mathbf{p}_{\mathbf{z}}^{\ell}[t] = \left\langle \mathbf{z}, \mathbf{q}_t^{\ell} \right\rangle + \sum_{k=0}^{\ell} \left\langle \mathbf{z}, Q^k\mathbf{r}_t^{\ell-k} \right\rangle$$

Hence $\mathbb{E}\left[\hat{\mathbf{p}}_{\mathbf{z}}^{\ell}[t]\right] = \mathbf{p}_{\mathbf{z}}^{\ell}[t]$. ■

Theorem 1: Theorem: LINEAR-SYSTEM-ESTIMATOR estimates $\langle \mathbf{x}, \mathbf{e}_t \rangle$ with relative accuracy ϵ and with probability $1 - p_{fail}$ in running time

$$\mathcal{O}\left(\left(\frac{\ell_{\max}^5 nnz(Q)\left(\ln\frac{\ell_{\max}}{p_{fail}}\right)^2}{n}\right)^{1/3}\left(\frac{\beta^{\ell_{\max}}|\mathbf{z}|_1}{\epsilon\delta}\right)^{4/3}\right)$$

Proof: We will prove this statement by first considering the single estimator $\hat{\mathbf{p}}_{\mathbf{z}}^{\ell}[t]$ for some ℓ since $\langle \mathbf{x}_t, \mathbf{e}_t \rangle = \sum_{l=0}^{\infty} \mathbf{p}_{\mathbf{z}}^{\ell}[t]$. ■

We will prove this statement by first considering the single estimator $\hat{\mathbf{p}}_{\mathbf{z}}^{\ell}[t]$ for some ℓ since $\langle \mathbf{x}_t, \mathbf{e}_t \rangle = \sum_{l=0}^{\infty} \mathbf{p}_{\mathbf{z}}^{\ell}[t]$.

Consider the estimator S_t^{ℓ}. We have already shown that $\mathbb{E}\left[S_{t,i}^{\ell}\right] = \mathbf{p}_{\mathbf{z}}^{\ell}[t] - \left\langle \sigma, \mathbf{q}_t^{\ell} \right\rangle$ and computed the work to achieve relative error ϵ for these estimators. Now observe:

$$\mathbb{P}\left[|\hat{\mathbf{p}}_{\mathbf{z}}^{\ell}[t] - \mathbf{p}_{\mathbf{z}}^{\ell}[t]| \geq \epsilon\mathbf{p}_{\mathbf{z}}^{\ell}[t]\right]$$
$$\leq \mathbb{P}\left[|X - \mathbb{E}[X]| \geq \epsilon\mathbb{E}[X]\right] \leq p_{fail}$$

By Lemma 5, the work done by MCMC-SAMPLER to achieve relative accuracy ϵ with probability $1 - p_{fail}$ is

$$\mathcal{O}\left(\frac{\ell_{\max}^2|\mathbf{z}|_1^2\delta_r^2\beta^{2\ell_{\max}}}{\epsilon^2\delta^2}\ln\left(\frac{\ell_{\max}}{p_{fail}}\right)\right)$$

By Lemma 6, we get that the forward and reverse work running times are equal asymptotically if we set

$$\delta_r = \sqrt[3]{\frac{nnz(Q)\epsilon^2\delta^2}{n\ell_{\max}|\mathbf{z}|_1^2\beta^{2\ell_{\max}}\ln(\ell_{\max}/p_{fail})}}$$

Substituting in this value for the forward walk work and we obtain:

$$\mathcal{O}\left(\left(\frac{\ell_{\max}^5 nnz(Q)\left(\ln\frac{\ell_{\max}}{p_{fail}}\right)^2}{n}\right)^{1/3}\left(\frac{\beta^{\ell_{\max}}|\mathbf{z}|_1}{\epsilon\delta}\right)^{4/3}\right)$$

which is in fact the total running time.

Lemma 6: The work done by MCMC-SAMPLER with averaged score variables $\frac{1}{n_f}\sum_{i=0}^{n_f} S_{i,t}$ returns an estimator $\widehat{\mathbb{E}[S_{i,t}^l]}$ with relative accuracy ϵ with probability $1 - p_{fail}$ in running time

$$\mathcal{O}\left(\frac{\ell_{\max}^2|\mathbf{z}|_1^2\delta_r^2\beta^{2\ell_{\max}}}{\epsilon^2\delta^2}\ln\left(\frac{\ell_{\max}}{p_{fail}}\right)\right)$$

provided that q

Proof: Let $X_i = S_{t,i}^{\ell}$ and $X = \sum_{i=1}^{n} X_i$. Then $X_i \in [-\ell_{\max}|\mathbf{z}|_1\delta_r\beta^{\ell_{\max}}, \ell_{\max}|\mathbf{z}|_1\delta_r\beta^{\ell_{\max}}]$ by Lemma 3. Since $\mathbb{E}[X_i] \geq \delta$, we let $c = \ell_{\max}|\mathbf{z}|_1\delta_r\beta^{\ell_{\max}}$, $a = \delta$ and by Lemma 6

$$n_f \geq \frac{2\ell_{\max}^2|\mathbf{z}|_1^2\delta_r^2\beta^{2\ell_{\max}}}{\epsilon^2\delta^2}\ln\left(\frac{\ell_{\max}}{p_{fail}}\right)$$

random walks to ensure that $\mathbb{P}\left[|\widehat{\mathbb{E}[S_{i,t}^l]} - \mathbb{E}[S_{i,t}^l]| \geq \epsilon\mathbb{E}[S_{i,t}^l]\right] \leq p_{fail}$ holds for $\ell_{\max}$ sets of $\hat{\mathbf{p}}_{\mathbf{z}}^{\ell}[t]$. Hence, the total work in the forward estimate to ensure $\mathbb{P}\left[|X - \mathbb{E}[X]]| \geq \epsilon\mathbb{E}[X]\right] \leq p_{fail}$ for $\ell_{\max}$ sets of estimators X is

$$\mathcal{O}\left(\frac{\ell_{\max}^2\delta_r^2|\mathbf{z}|_1^2\beta^{2\ell_{\max}}}{\epsilon^2\delta^2}\ln\left(\frac{\ell_{\max}}{p_{fail}}\right)\right)$$

■

Lemma 7: Set

$$\delta_r = \sqrt[3]{\frac{nnz(Q)(\ell_{\max}+1)\epsilon^2\delta^2}{n\ell_{\max}^2|\mathbf{z}|_1^2\beta^{2\ell_{\max}-1}\ln(\ell_{\max}/p_{fail})}}$$

to balance the reverse push and forward walk work.

Proof: From Lemma 2, the work from the reverse push operation is $O\left(\frac{nnz(Q)}{n\delta_r}(\ell_{\max}+1)\beta\right)$. To get the optimal running time asymptotically, we set the forward work and reverse work equal to each other and solve for δ_r:

$$\frac{nnz(Q)}{n\delta_r}(\ell_{\max}+1)\beta = \frac{\ell_{\max}^2\delta_r^2|\mathbf{z}|_1^2\beta^{2\ell_{\max}}}{\epsilon^2\delta^2}\ln\left(\frac{\ell_{\max}}{p_{fail}}\right)$$

Thus we obtain

$$\delta_r = \sqrt[3]{\frac{nnz(Q)(\ell_{\max}+1)\epsilon^2\delta^2}{\ell_{\max}^2 n|\mathbf{z}|_1^2\beta^{2\ell_{\max}-1}\ln(\ell_{\max}/p_{fail})}}$$

For $\ell_{\max}$ sufficiently large, we can replace $\ell_{\max}+1$ with just $\ell_{\max}$ and $\beta^{2\ell_{\max}-1}$ with $\beta^{2\ell_{\max}}$. ■

Lemma 8: Set $\ell_{\max} = \frac{1}{\ln\rho(G)}\ln\left(\frac{\delta(1-\rho(G))}{||\mathbf{z}||}\right)$ to satisfy additive error threshold of δ.

Proof: Note that $\ell_{\max}$ controls error by truncating the power series $\sum_{l=0}^{\infty}\langle\mathbf{z},Q^l\mathbf{e}_t\rangle$. Suppose we have this error as $\Delta = \sum_{l=0}^{\infty}\langle\mathbf{z},Q^l\mathbf{e}_t\rangle - \sum_{l=0}^{\ell_{\max}}\langle\mathbf{z},Q^l\mathbf{e}_t\rangle = \langle\mathbf{z},Q^{\ell_{\max}}\sum_{l=1}^{\infty}Q^{\ell}\mathbf{e}_t\rangle$. Then $\Delta(\ell_{\max}) \leq ||\mathbf{z}||\frac{\rho(G)^{\ell_{\max}}}{1-\rho(G)}$. If we want additive error δ (that is, $\delta \leq \Delta(\ell_{\max})$), provided $\delta \leq ||\mathbf{z}||$, we have $\ell_{\max} \geq \frac{1}{\ln\rho(G)}\ln\left(\frac{\delta(1-\rho(G))}{||\mathbf{z}||}\right)$. Recall that $\rho(G) < 1$, so $\ln\rho(G) < 0$ and the suggested value for $\ell_{\max}$ increases as δ shrinks. ■

Lemma 9 (Hoeffding's Inequality): Let $\{X_i\}$ be independent random variable s.t. for all i, $X_i \in [-c,c]$ a.s., and $|\mathbb{E}[X_i]| \geq a$. Then $X = \sum_{i=1}^n X_i$ satisfies $\mathbb{P}[|X-\mathbb{E}[X]| \geq \epsilon\mathbb{E}[X]] \leq p_{fail}$ provided that

$$n \geq \frac{4c^2}{\epsilon^2a^2}\ln\left(\frac{2}{p_{fail}}\right)$$

Proof: Let $X = \sum_{i=1}^n X_i$. Since $E[X] = nE[X_i]$, $E[X] \geq n\epsilon a$. Let $t = \epsilon a$.

$$\begin{aligned}\mathbb{P}\left[|X-\mathbb{E}[X]| \geq \epsilon\mathbb{E}[X]\right] &\leq \mathbb{P}\left[|X-\mathbb{E}[X]| \geq nt\right] \\ &= \mathbb{P}\left[\left|\frac{1}{n}X - \frac{1}{n}\mathbb{E}[X]\right| \geq t\right]\end{aligned}$$

Applying Hoeffding's inequality to the rightmost term above, we obtain

$$\mathbb{P}\left[|X-\mathbb{E}[X]| \geq \epsilon\mathbb{E}[X]\right] \leq 2\exp\left(-\frac{2n^2t^2}{\sum_{i=1}^n(b_i-a_i)^2}\right)$$

Hence substituting the values for t and $b_i = c$, $a_i = -c$ gives us:

$$\mathbb{P}\left[|X-\mathbb{E}[X]| \geq \epsilon\mathbb{E}[X]\right] \leq 2\exp\left(-\frac{2n^2\epsilon^2a^2}{n(4c^2)}\right)$$

Now set this upperbound to p_{fail} to obtain the relation:

$$\frac{n\epsilon^2a^2}{2c^2} = \ln\left(\frac{2}{p_{fail}}\right)$$

Thus, for $n \geq \frac{2c^2}{\epsilon^2a^2}\ln\left(\frac{2}{p_{fail}}\right)$, we are guaranteed that $\mathbb{P}\left[|X-\mathbb{E}[X]| \geq \epsilon\mathbb{E}[X]\right] \leq p_{fail}$.

Note that if we require the failure conditions to hold for $\ell_{\max}$ sets of $X's$, then we perform a simple union bound which adds a $\ell_{\max}$ in the $\frac{2}{p_{fail}}$ term. ■