Implementing in a Stream

- S has a concise description using $O(\log n)$-wise independent hash functions

- We need a way to define the matrix $\boldsymbol{E}$ that preserves the properties we want

- We cannot use independent $\mathbf{e}_1, \ldots, \mathbf{e}_n$ -- $\Omega(n)$ space is required

- **Idea:** Use a pseudorandom string to generate $\mathbf{e}_1, \ldots, \mathbf{e}_n$

- Andoni uses the PRG of Nisan and Zuckerman

- Slow to retrieve an arbitrary $\mathbf{e}_i$

31

Implementing in a Stream

- $\boldsymbol{S}$ has a concise description using $O(\log n)$-wise independent hash functions
- We need a way to define the matrix $\boldsymbol{E}$ that preserves the properties we want
    - We cannot use independent $\mathbf{e}_1, \ldots, \mathbf{e}_n$ -- $\Omega(n)$ space is required
    - **Idea:** Use a pseudorandom string to generate $\mathbf{e}_1, \ldots, \mathbf{e}_n$
    - Andoni uses the PRG of Nisan and Zuckerman
        - Slow to retrieve an arbitrary $\mathbf{e}_i$

Preserving Properties via Nisan's PRG