**Problem 8.** There are many alternative algorithms for computing pagerank vectors. In this problem, we will investigate an algorithm that I call the *exponentially accelerated power method*, although it does not have a commonly accepted name. This is a divide and conquer algorithm that can achieve the same accuracy $\epsilon$ as the power method with only a logarithmic number of iterations.

The estimated pagerank vector is given by

$$\mathbf{y}^{(K)} = \mathbf{x}^{(0)}\mathbf{Q}_K, \tag{6}$$

where

$$\mathbf{Q}_k = \begin{cases} \bar{\bar{\mathbf{P}}} & \text{if } k = 0 \\ \mathbf{Q}_{k-1}\mathbf{Q}_{k-1} & \text{otherwise} \end{cases}. \tag{7}$$

In the standard power method, the matrix $\bar{\bar{\mathbf{P}}}$ is not stored explicitly, but is calculated from the $\mathbf{P}$ matrix. In this problem, you can assume for simplicity that the $\bar{\bar{\mathbf{P}}}$ matrix is stored explicitly as a dense matrix, and that $\mathbf{Q}_k$ is also stored as a dense matrix.

1. Show that $\mathbf{y}^{(K)} = \mathbf{x}^{(2^K)}$. This equivalence is why the algorithm is "exponentially accelerated."

   HINT: Use induction to show that $Q_K = \bar{\bar{\mathbf{P}}}^{2^K}$. The result follows by combining this fact with (6) and Equation (5.1) in the paper.

base case $K=0$: $\underset{\uparrow \text{ by } (7)}{Q_0 = \bar{\bar{P}}} = \bar{\bar{P}}^{2^0}$ ✓

inductive step: Assume $Q_{k-1} = \bar{\bar{P}}^{2^{k-1}}$

$$Q_k = Q_{k-1}Q_{k-1} \quad \text{by } (7)$$
$$= \bar{\bar{P}}^{2^{k-1}}\bar{\bar{P}}^{2^{k-1}}$$
$$= \bar{\bar{P}}^{2^k} \checkmark$$

2. What is the runtime of calculating $\mathbf{Q}_k$ given $\mathbf{Q}_{k-1}$?

$O(n^3)$

3. What is the runtime of computing $\mathbf{y}^{(K)}$ in terms of $K$?

$O(k n^3)$

4. As with the standard power method, we do not know the total number of iterations of the exponential power method in advance. Instead, we iterate until

$$\|\mathbf{y}^{(K)} - \mathbf{y}^{(K-1)}\|_2 \leq \epsilon, \tag{8}$$

where $\epsilon$ is a predetermined small constant value. Bounding the number of iterations $K$ required to satisfy (8) is quite a bit more technical than in the previous problem. You do not have to compute a bound on $K$ yourself, and may instead assume that

$$K = O\left( \log \frac{\log \epsilon}{\log \alpha} \right) \tag{9}$$

satisfies (8). Notice that this number of iterations is logrithmic compared to the number of iterations in the standard power method, and this is where the name exponentially accelerated comes from..

What is the runtime of computing $\mathbf{y}^{(K)}$ in terms of $\epsilon$?

$$O\left( n^3 \log \frac{\log \varepsilon}{\log \alpha} \right)$$

5. Under what conditions is the exponentially accelerated power method faster than the standard power method?

when $n$ is small, and $\varepsilon$ is very small

6. Under what conditions is it slower?

when $n$ is large and $\varepsilon$ is large

or

when P is sparse

7. What bad thing would happen if $\mathbf{P}$ was stored as a sparse matrix and $\bar{\mathbf{P}}$ was calculated from $\mathbf{P}$ as in the standard power method?

You cannot multiply two sparse matrices in pytorch, so the exponentially accelerated method cannot be implemented