

Exercise

You will implement a simplified version of Google's original PageRank algorithm on a [Wikipedia snapshot](#). Your aim is to rank all the pages by order of importance and implement a basic search function.

Extra credit: implement PageRank with a **damping factor**. You need to find out what it is yourself, and understand **why** a damping factor is necessary.



We shall make the following assumptions.

- Each page has the same probability of being the start page
- There is at most one link from one page to another
- On each page, each link has the same probability of being clicked.
- If there are no links, the next page can be any page with equal probability.

Main ideas

The main idea is to rank pages by the *probability* of landing on them.

What does this mean? Imagine you're browsing the web and click on a random link to go from one page to the next.

Denote by $X^{(k)}$ be our position after k clicks.

$$\underset{\text{initial page}}{X^{(0)}} \xrightarrow[\text{random link}]{\text{click}} X^{(1)} \xrightarrow[\text{random link}]{\text{click}} X^{(2)} \xrightarrow[\text{random link}]{\text{click}} \dots$$

$X^{(k)}$ → position (url) after k clicks
 $X^{(0)} \rightarrow X^{(1)} \rightarrow X^{(2)} \rightarrow \dots$

Aim P

Observations

- For small k , the position will heavily depend on the initial page $X^{(0)}$.
- The effect will subside as k grows larger.
- The page rank of a page will thus be defined via

$$\text{PageRank}(\text{page}) = \lim_{k \rightarrow +\infty} \mathbb{P}(X_k = \text{page})$$

$X^{(k)}$ = position after k clicks

Definition

$$\vec{p}^{(k)} = \begin{pmatrix} P(X^{(k)} = 1) \\ P(X^{(k)} = 2) \\ \vdots \\ P(X^{(k)} = n) \end{pmatrix}$$

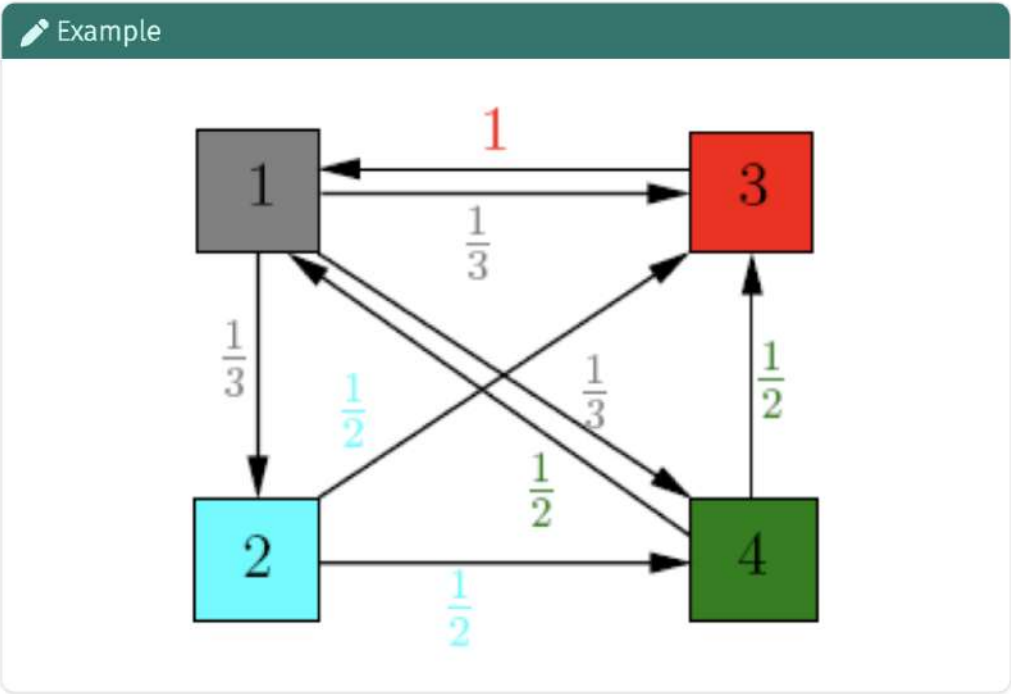
$$\vec{p}^{(k)} = \begin{pmatrix} P(X^{(k)} = 1) \\ P(X^{(k)} = 2) \\ \vdots \end{pmatrix} \rightarrow \text{probability of landing on 1 after } k \text{ clicks}$$

Definition (PageRank vector)

$$\vec{p}^{(\infty)} = \lim_{k \rightarrow +\infty} \vec{p}^{(k)}$$

Definition (Probability transition matrix)

$$T_{ij} = \text{probability of from } j \text{ to } i$$
$$= P(X^{(k+1)} = i \mid X^{(k)} = j)$$



$T_{i \leftarrow j}$ = probability of going from j to i

Handwritten matrix T and annotations:

$$T = \begin{pmatrix} 0 & 0 & 1 & 1/2 \\ 1/3 & 0 & 0 & 0 \\ 1/3 & 1/2 & 0 & 1/2 \\ 1/3 & 1/2 & 0 & 0 \end{pmatrix}$$

Annotations:

- T_{12} points to the top-right element (1).
- $3 \rightarrow 1$ points to the top-right element (1).
- $1 \rightarrow 2$ points to the second row, first column element ($1/3$).
- A bracket under the first column is labeled "1st column".
- Below the bracket is "start from page 1".

Proposition

$\vec{p}^{(k+1)}$
probabilities after $k+1$ clicks

=

stochastic matrix
 T

$\vec{p}^{(k)}$
probabilities after k clicks

$$\vec{p}^{(0)} = \begin{pmatrix} 1/N \\ 1/N \\ 1/N \\ \vdots \end{pmatrix}$$

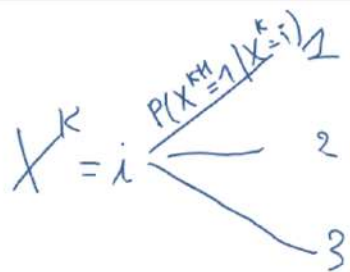
$$\vec{p}^{(k+1)} = T \vec{p}^{(k)}$$

Proposition

$$\vec{p}^{(k+1)} = T \vec{p}^{(k)}$$

$$\begin{aligned} p_j^{(k+1)} &= P(X^{k+1} = j) = \sum_{i=1}^N \underbrace{P(X^{k+1} = j | X^{(k)} = i)}_{T_{ji}} \underbrace{P(X^{(k)} = i)}_{p_i^{(k)}} \\ &= (T \vec{p}^{(k)})_j \end{aligned}$$

```
1 from numpy import matrix
2 T = matrix([
3     [0, 0, 1, 1/2],
4     [1/3, 0, 0, 0],
5     [1/3, 1/2, 0, 1/2],
6     [1/3, 1/2, 0, 0],
7 ])
8 p = matrix([[1/4], [1/4], [1/4], [1/4]])
9 for i in range(100):
10     p = T*p
11 p
```



$$P(X^{k+1} = j) = \sum P(\dots)$$

Our aim is to calculate $\vec{p}^{(\infty)} = \lim_{k \rightarrow +\infty} \vec{p}^{(k)}$ via the iteration

$$\vec{p}^{(k+1)} = T\vec{p}^{(k)}$$

In other words, we'll approximate $\vec{p}^{(\infty)} \approx \vec{p}^{(k)}$ for some large k .

? Question

How do we choose k ?

Note that taking the limit on both sides, we get

$$\vec{p}^{(\infty)} = T\vec{p}^{(\infty)} \iff T\vec{p}^{(\infty)} - \vec{p}^{(\infty)} = 0.$$

A good stopping criterion is thus

$$\frac{\|T\vec{p}^{(k)} - \vec{p}^{(k)}\|_1}{\|\vec{p}^{(k)}\|_1} \leq \epsilon$$

where $\|v\|_1 = \sum_{i=1}^n |v_i|$.

- 1. Read the graph from the CSV files
- 2. Construct the transition matrix T . The dataset is huge, have a look at *sparse* matrices.

- 3. Calculate $\vec{p}^{(k)}$ for k sufficiently large via the iteration

$$\vec{p}^{(k+1)} = T\vec{p}^{(k)}$$

- 4. Deduce an approximation of the PageRank vector

$$\vec{p}^{(\infty)} \approx \vec{p}^{(k)}$$

for k sufficiently large.

$T = \begin{pmatrix} 0 & a & 0 & b \\ 0 & 0 & 0 & 0 \\ \vdots & & & \end{pmatrix}$ ← sparse matrix

↕

$[(1,2,a), (1,4,b), \dots]$ → scipy sparse matrix

