26. Computing the Rank of a Webpage

Google PageRank
More Practice with 2D Array OPs
More Practice with numpy
Assume

```python
global random
from random import uniform as randu
from numpy import *
```

Let's write a function `randuM(m,n)` that returns an m-by-n array of random numbers, each chosen from the uniform distribution on [0,1].
A Function that Returns an $n$-by-$n$ Array of Random Numbers

def randuM(m,n):
    A = zeros((m,n))
    for i in range(m):
        for j in range(n):
            A[i,j] = randu(0,1)
    return A
A $n \times n$ probability array has the property that its entries are nonnegative and that the sum of the entries in each column is 1.

\[
\begin{array}{ccc}
.2 & .6 & .2 \\
.7 & .3 & .3 \\
.1 & .1 & .5 \\
\end{array}
\]
To generate a random probability array, generate a random matrix with nonnegative entries and then divide the numbers in each column by the sum of the numbers in that column.
def probM(n):
    A = randuM(n,n)
    for j in range(n):
        # Normalize column j
        s = 0;
        for i in range(n):
            s += A[i,j]
        for i in range(n):
            A[i,j] = A[i,j]/s
    return A
Here is a Network
A node

Think of a node as an island

Think of a node as a Web page

A Transition Probability
With prob \(0.1\), a person on island 1 will hop to island 2.
A Random Process

Suppose there are a 1000 people on each node.

At the sound of a whistle they hop to another node in accordance with the “outbound” probabilities.
At Node 0

0 1 2

0.1 0.2 0.3

1.0 1.1 0.2

200 700 100

0.7 0.6 0.3

0.5
At Node 1

0 → 1 → 2

Weights:
- 0 to 1: 0.7
- 0 to 2: 0.2
- 1 to 0: 0.6
- 1 to 2: 0.3
- 2 to 0: 0.1
- 2 to 1: 0.5

Numbers:
- Node 1: 300
- Node 2: 100
- Node 0: 600
At Node 2
The Population Distribution

<table>
<thead>
<tr>
<th></th>
<th>Before</th>
<th>After</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node 0</td>
<td>1000</td>
<td>1000</td>
</tr>
<tr>
<td>Node 1</td>
<td>1000</td>
<td>1300</td>
</tr>
<tr>
<td>Node 2</td>
<td>1000</td>
<td>700</td>
</tr>
<tr>
<td>Node</td>
<td>Before</td>
<td>After</td>
</tr>
<tr>
<td>------</td>
<td>--------</td>
<td>-------</td>
</tr>
<tr>
<td>0</td>
<td>1000</td>
<td>1120</td>
</tr>
<tr>
<td>1</td>
<td>1300</td>
<td>1300</td>
</tr>
<tr>
<td>2</td>
<td>700</td>
<td>580</td>
</tr>
</tbody>
</table>
**After 100 Iterations**

<table>
<thead>
<tr>
<th></th>
<th>Before</th>
<th>After</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node 0</td>
<td>1142.85</td>
<td>1142.85</td>
</tr>
<tr>
<td>Node 1</td>
<td>1357.14</td>
<td>1357.14</td>
</tr>
<tr>
<td>Node 2</td>
<td>500.00</td>
<td>500.00</td>
</tr>
</tbody>
</table>

Appears to reach a Steady State
### After 100 Iterations

<table>
<thead>
<tr>
<th>Node</th>
<th>Before</th>
<th>After</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node 0</td>
<td>1142.85</td>
<td>1142.85</td>
</tr>
<tr>
<td>Node 1</td>
<td>1357.14</td>
<td>1357.14</td>
</tr>
<tr>
<td>Node 2</td>
<td>500.00</td>
<td>500.00</td>
</tr>
</tbody>
</table>

In terms of popularity: Island 1 > Island 0 > Island 2
### After 100 Iterations

<table>
<thead>
<tr>
<th>Node 0</th>
<th>Before</th>
<th>After</th>
</tr>
</thead>
<tbody>
<tr>
<td>Node 0</td>
<td>1142.85</td>
<td>1142.85</td>
</tr>
<tr>
<td>Node 1</td>
<td>1357.14</td>
<td>1357.14</td>
</tr>
<tr>
<td>Node 2</td>
<td>500.00</td>
<td>500.00</td>
</tr>
</tbody>
</table>

\[ [ 1142.85, 1357.14, 500.0 ] \] is the "stationary array"
Computing the Stationary Array Involves a Probability Array
Computing the Stationary Array Involves a Probability Array.

The (0,1) entry is the probability of hopping from island 1 to island 0.
Transition Probability Array

$P[i,j]$ is the probability of hopping from node $j$ to node $i$
Formula for Updating the Distribution Array

\[
\mathbf{w} = \begin{bmatrix}
0.2 & 0.6 & 0.2 \\
0.7 & 0.3 & 0.3 \\
0.1 & 0.1 & 0.5
\end{bmatrix}
\]

\[
\begin{align*}
\mathbf{w}[0] &= 0.2 \cdot \mathbf{v}[0] + 0.6 \cdot \mathbf{v}[1] + 0.2 \cdot \mathbf{v}[2] \\
\mathbf{w}[1] &= 0.7 \cdot \mathbf{v}[0] + 0.3 \cdot \mathbf{v}[1] + 0.3 \cdot \mathbf{v}[2] \\
\mathbf{w}[2] &= 0.1 \cdot \mathbf{v}[0] + 0.1 \cdot \mathbf{v}[1] + 0.5 \cdot \mathbf{v}[2]
\end{align*}
\]

\(\mathbf{V}\) is the old distribution array, 
\(\mathbf{w}\) is the updated distribution array
Formula for Updating the Distribution Vector

$$\begin{bmatrix}
.2 & .6 & .2 \\
.7 & .3 & .3 \\
.1 & .1 & .5 \\
\end{bmatrix}$$

$$\mathbf{w}[0] = P[0,0] \times \mathbf{v}[0] + P[0,1] \times \mathbf{v}[1] + P[0,2] \times \mathbf{v}[2]$$

$$\mathbf{w}[1] = P[1,0] \times \mathbf{v}[0] + P[1,1] \times \mathbf{v}[1] + P[1,2] \times \mathbf{v}[2]$$

$$\mathbf{w}[2] = P[2,0] \times \mathbf{v}[0] + P[2,1] \times \mathbf{v}[1] + P[2,2] \times \mathbf{v}[2]$$

**V is the old distribution vector, w is the updated distribution vector**
A Function that Computes the Update

def Update(P, v):
    n = len(x)
    w = zeros((n,1))
    for i in range(n):
        for j in range(n):
            w[i] += P[i,j] * v[j]
    return w
Back to PageRank
Background

Index all the pages on the Web from 0 to N-1. (N is around 50 billion.)

The PageRank algorithm orders these pages from “most important” to “least important”.

It does this by analyzing links, not content.
Key Ideas

The Transition Probability Array

A Very Special Random Walk

The Connectivity Array
A Random Walk on the Web

Repeat:
You are on a webpage.
There are $m$ outlinks.
Choose one at random.
Click on the link.
The Connectivity Array

$G[i,j]$ is 1 if there is a link on page $j$ to page $i$.

<p>| | | | | | | | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
</tbody>
</table>
The Probability Array

\[ P: \]

\[
\begin{array}{cccccccc}
0 & a & 0 & 0 & b & 0 & c & 0 \\
a & 0 & 0 & 0 & 0 & 0 & c & 1 \\
a & 0 & 0 & 0 & b & 0 & 0 & 0 \\
a & 0 & a & b & 0 & a & 0 & 0 \\
0 & 0 & 0 & b & 0 & 0 & c & 0 \\
0 & a & a & 0 & 0 & a & 0 & 0 \\
a & 0 & 0 & 0 & 0 & 0 & c & 0 \\
0 & 0 & a & 0 & 0 & a & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & a & 0 \\
0 & 0 & 0 & 0 & a & 0 & 0 & 0 \\
\end{array}
\]

\[
\begin{align*}
a &= \frac{1}{3} \\
b &= \frac{1}{2} \\
c &= \frac{1}{4}
\end{align*}
\]
PageRank From the Stationary Array

Stationary Array

<table>
<thead>
<tr>
<th>PageRank</th>
<th>Webpage 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5723</td>
<td>3</td>
</tr>
<tr>
<td>0.8206</td>
<td>1</td>
</tr>
<tr>
<td>0.7876</td>
<td>2</td>
</tr>
<tr>
<td>0.2609</td>
<td>5</td>
</tr>
<tr>
<td>0.2064</td>
<td>7</td>
</tr>
<tr>
<td>0.8911</td>
<td>0</td>
</tr>
<tr>
<td>0.2429</td>
<td>6</td>
</tr>
<tr>
<td>0.4100</td>
<td>4</td>
</tr>
</tbody>
</table>

Webpage 5 Has pageRank 0