Como o Google Ordena Páginas
MAP2110

Professor:	Nelson Mugayar Kuhl	NºUSP
Grupo:	Enzo Valentim Cappelozza	12556736
	Henrique Fujikawa Tokunaga	12675207
	Lucas Panfilo Donaire	12556552
	Marcos Martins Marchetti	11910868
	Ygor Peniche Maldonado	10271558

9 de julho de 2021

Conteúdo

1 Introdução

2 Resultados de álgebra Linear

2.1 Lema

Se M é uma matriz $n \times n$ tal que a soma dos elementos de cada coluna é igual a 1, então 1 é ***autovalor*** de M, o que é o mesmo que afirmar que a equação (2) tem solução. Uma tal solução é chamada de ***autovetor*** associado ao autovalor 1.

2.2 Teorema de Perron-Frobenius

Seja M uma matriz $n \times n$ com todos os elementos positivos ($M_{ij} > 0, \forall i, j \in \{1, ...n\}$)

3 O Algorítimo

4 Exercícios

4.1 Exercício 1

Seja M uma matriz $n \times n$ satisfazendo as hipóteses do *Teorema de Perron-Frobenius (1)*. Mostre que se y é um vetor com todas as entradas positivas e normalizado, então $z = My$ será um vetor com todas as entradas positivas e normalizado.

É notável que:

$$A = \begin{pmatrix} 0 & a_{12} & a_{13} & a_{14} & \dots & a_{1n} \\ a_{21} & 0 & a_{23} & a_{24} & \dots & a_{2n} \\ a_{31} & a_{32} & 0 & a_{34} & \dots & a_{3n} \\ a_{41} & a_{42} & a_{43} & 0 & \dots & a_{4n} \\ \vdots & \vdots & \vdots & \dots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & a_{n4} & \dots & 0 \end{pmatrix}$$

Pois para cada uma das equações de relação de menções em link, temos que uma página pode ser linkada a **no máximo** $n - 1$ páginas, visto que não tem sentido uma página fazer referência à si mesma. Portanto o elemento a_{ij} da matriz A, esse que é referente à si mesmo, deve ser sempre zero. Nota-se então que a diagonal principal é integralmente composta por elementos nulos e para qualquer i, j temos $a_{ij} = 0$ se $i = j$.

Note que cada coluna j da matriz A resulta na soma 1. Portanto, para cada coluna teremos:

$$c_j = \sum_{i=1}^{n} (a_{ij}) = 1$$

Portanto, para a matriz A o *Lema 1* nos garante a existência de uma solução.

No entanto, para garantirmos a possibilidade de (1), é necessário que se satisfaçam as hipóteses do Teorema. Introduz-se a *Matriz Perturbada* M.

$$M = (1-m)A + mS$$

Realizando as operações dos termos à direita do símbolo de igualdade, obtemos:

$$(1-m)A = \begin{pmatrix} 0 & (1-m)a_{12} & \dots & (1-m)a_{1n} \\ (1-m)a_{21} & 0 & \dots & (1-m)a_{2n} \\ (1-m)a_{31} & (1-m)a_{32} & \dots & (1-m)a_{3n} \\ \vdots & \vdots & \ddots & \vdots \\ (1-m)a_{n1} & (1-m)a_{n2} & \dots & 0 \end{pmatrix}$$

Note que $a_{ij} = 0$ sempre que $i = j$.

$$mS = \begin{pmatrix} m/n & m/n & m/n & m/n & \dots & m/n \\ m/n & m/n & m/n & m/n & \dots & m/n \\ m/n & m/n & m/n & m/n & \dots & m/n \\ m/n & m/n & m/n & m/n & \dots & m/n \\ \vdots & \vdots & \vdots & \dots & \ddots & \vdots \\ m/n & m/n & m/n & m/n & \dots & 0 \end{pmatrix}$$

Note que a soma de qualquer coluna da matriz mS é $\frac{m}{n} + \frac{m}{n} + \ldots + \frac{m}{n} = n\frac{m}{n} = m$

Tomemos agora uma coluna qualquer da matriz M
Sabe-se que $a_{ij} = 0$ se $i = j$:

$$c_j^M = (1-m)\sum_{i=1}^{n}(a_{ij}) + n\frac{m}{n} = (1-m) + m = 1$$

Demonstração. Seja o y o vetor cuja forma é $y = \begin{pmatrix} \frac{y1}{||y||} & \frac{y2}{||y||} & \frac{y3}{||y||} \cdots \frac{yn}{||y||} \end{pmatrix}^T$
E cuja soma dos elementos é a unidade, pois y é normalizado.
Temos então que, para algum dos elementos do vetor z, por My :

$$z_i = \sum_{k=1}^{n} \left((1-m)a_{ik} + \frac{m}{n}\right) y_{k1}$$

Onde ainda teremos que $a_{ik} = 0$ caso $i = k$, e que resultará em:

$$z_i = \sum_{k=1}^{n} \left((1-m)a_{ik}\frac{y_k}{||y||}\right) + \sum_{k=1}^{n} \frac{m}{n}\left(\frac{y_k}{||y||}\right)$$

Portanto

$$z_i = (1-m)\sum_{k=1}^{n} a_{ik}\frac{y_k}{||y||} + \frac{m}{n} \qquad (1)$$

Agora, para a soma de todos os elementos do vetor z temos:

$$\sum_{i=1}^{n} z_i = \sum_{i=1}^{n} \left((1-m) \sum_{k=1}^{n} a_{ik} \frac{y_k}{||y||} \right) + n \left(\frac{m}{n} \right)$$

$$= (1-m) \sum_{i=1}^{n} \left(\sum_{k=1}^{n} a_{ik} \frac{y_k}{||y||} \right) + m$$

$$\stackrel{*}{=} \frac{(1-m)}{||y||} \sum_{i=1}^{n} (a_{i1}y_1 + a_{i2}y_2 + \ldots + a_{in}y_n) + m$$

E sabemos que, por construção:

$$\sum_{i=1}^{n} a_{ij} = 1$$

Aplicando a distributividade do somatório, obtemos:

$$\stackrel{*}{=} (1-m) \sum_{i=1}^{n} \left(\frac{y_i}{||y||} \right) + m$$

Sabemos também que o vetor y é normalizado, então

$$\sum_{i=1}^{n} \frac{y_i}{||y||} = 1$$

$$\stackrel{*}{=} 1 - m + m = 1$$

A soma de todos os elementos de z é a unidade. Ou seja, é um vetor normalizado. Note também que pela Equação (1) podemos afirmar que todas as entradas são positivas, pois $0 < m < 1$ e $n \in \mathbb{N}$.

$\square$

4.2 Exercício 2

Considere uma matriz M tal que $M_{n \times n}$ onde

$$M = (1-m)A + mS$$

com $0 < m < 1$, $A_{n \times n}$, e $S_{n \times n}$ com todas as $S_{ij} = 1/\text{n}$, tal que

$$S = \begin{pmatrix} S_{1,1} & \ldots & S_{1,n} \\ \vdots & \ddots & \vdots \\ S_{n,1} & \ldots & S_{n,n} \end{pmatrix}$$

Mostre que: Se y é um vetor com n entradas iguais a 1/n, então

$$My = (1-m)Ay + m \begin{pmatrix} 1/n \\ \vdots \\ 1/n \end{pmatrix}$$

Mostre também que se y for um vetor cujas entradas somam 1, então $M \times y$ também é calculado pela expressão acima.

Demonstração. Tomando n = 2. Temos $y_{2\times 1}$:

$$y = \begin{pmatrix} 1/2 \\ 1/2 \end{pmatrix}$$

Note que a soma de $Y_{1,1} + Y_{2,1} = \frac{1}{2} + \frac{1}{2} = 1$

Tomemos $y_{n\times 1}$

$$y = \begin{pmatrix} 1/n \\ \vdots \\ 1/n \end{pmatrix}$$

Note que $Y_{1,1} + ... + Y_{n,1} = \frac{1}{n} + ... + \frac{1}{n} = 1$

Temos que y é um vetor cujo somatório de todas as entradas equivale à 1.

Tomemos

- o vetor $y_{n\times 1}$

$$y = \begin{pmatrix} 1/n \\ \vdots \\ 1/n \end{pmatrix}$$

- Matriz ligação $A_{n\times n}$
- Matriz perturbada $M_{n\times n}$
- Matriz $S_{n\times n}$

Multiplicando a igualdade $M = (1-m)A + mS$ por y temos

$$My = (1-m)Ay + mSy$$

realizando o produto matricial Sy temos uma matriz resultado

$$R = Sy = \begin{pmatrix} R_{1,1} \\ \vdots \\ R_{n,1} \end{pmatrix}$$

tal que cada elemento $R_{i,1} = \frac{1}{n} \times \frac{1}{n} + \ldots + \frac{1}{n} \times \frac{1}{n} = n \times \frac{1^2}{n^2} = \frac{1}{n}$

Ou seja,

$$mSy = mR = m \begin{pmatrix} 1/n \\ \vdots \\ 1/n \end{pmatrix}$$

Lembremos que, *por construção*, o somatório das colunas de A equivale a 1, ou seja:

$$\sum_{i=1}^{n} a_{ij} = 1$$

Então, realizando o produto Ay com $A_{n\times n}$ e $y_{n\times 1}$ Obtemos uma Matriz Resultado* R^*:

$$Ay = R^* = \begin{pmatrix} 0 & a_{1,2} & \dots & a_{1,n} \\ a_{2,1} & 0 & \dots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n,1} & a_{n,2} & \dots & a_{n,n} \end{pmatrix} \times \begin{pmatrix} 1/n \\ \vdots \\ 1/n \end{pmatrix}$$

Onde, a diagonal principal tem elementos nulos tais que $a_{1,1} = a_{n,n} = 0$

Então

$$R^* = \begin{pmatrix} ((a_{1,1} + ... + a_{1,n})/n) \\ ((a_{2,1} + ... + a_{2,n})/n) \\ \vdots \\ ((a_{n,1} + ... + a_{n,n})/n) \end{pmatrix}$$

E a somatória de todos os elementos de R^* é equivalente à 1, ou seja:

$$\sum_{i=1}^{n} R^*_{i,n} = 1$$

Portanto,

$$My = (1-m)Ay + mSy$$

$$My = (1-m)R^* + mR$$

$$My = (1-m) \begin{pmatrix} ((a_{1,1} + ... + a_{1,n})/n) \\ ((a_{2,1} + ... + a_{2,n})/n) \\ \vdots \\ ((a_{n,1} + ... + a_{n,n})/n) \end{pmatrix} + m \begin{pmatrix} 1/n \\ \vdots \\ 1/n \end{pmatrix}$$

$\square$

4.3 Exercício 3

Escreva um programa que, dada uma matriz de ligação A $n \times n$, calcula o vetor dos pesos normalizado conforme o algoritmo descrito na seção 3 e ordena as "páginas"de acordo com ele. Use $m = 0.15$ e a condição inicial

$$x^{(0)} = (1/n, 1/n, ..., 1/n)^t$$

Calcule as iterações até que o erro dado por (4) seja menor do que 10^{-5} (para isso, você deve calcular a constante c, que deve ser impressa também). Teste o seu programa com o exemplo relacionado à figura presente no anexo.

Para o cálculo do vetor dos pesos normalizados, teremos que sua norma é dada por:

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

A partir dela, o algoritmo para o cálculo do vetor de pesos será obtido por meio dos seguintes passos:

- Escolhemos um vetor $x^{(0)}$ normalizado ($\|x^{(0)}\| = 1$) e todo positivo ($x_i^{(0)} > 0, 1 \leq i \leq n$)
- Para $k = 1, 2, 3, ...$ calculamos

$$x^{(k)} = Mx^{(k-1)}$$

Este é denominado Método das Potências. Caso a matriz satisfaça as hipóteses do Teorema 1, podemos mostrar que $x^{(k)}$ converge para o autovetor normalizado associado ao autovalor 1 com taxa de:

$$c = max_{1 \leq j \leq n} |1 - 2\ min_{1 \leq i \leq n}\ M_{ij}|$$

Assim, a estimativa de erro a *posteriori* será:

$$\|x - x^{(k)}\| \leq \frac{c}{1-c} \|x^{(k)} - x^{(k-1)}\|$$

Sendo x o autovetor normalizado.

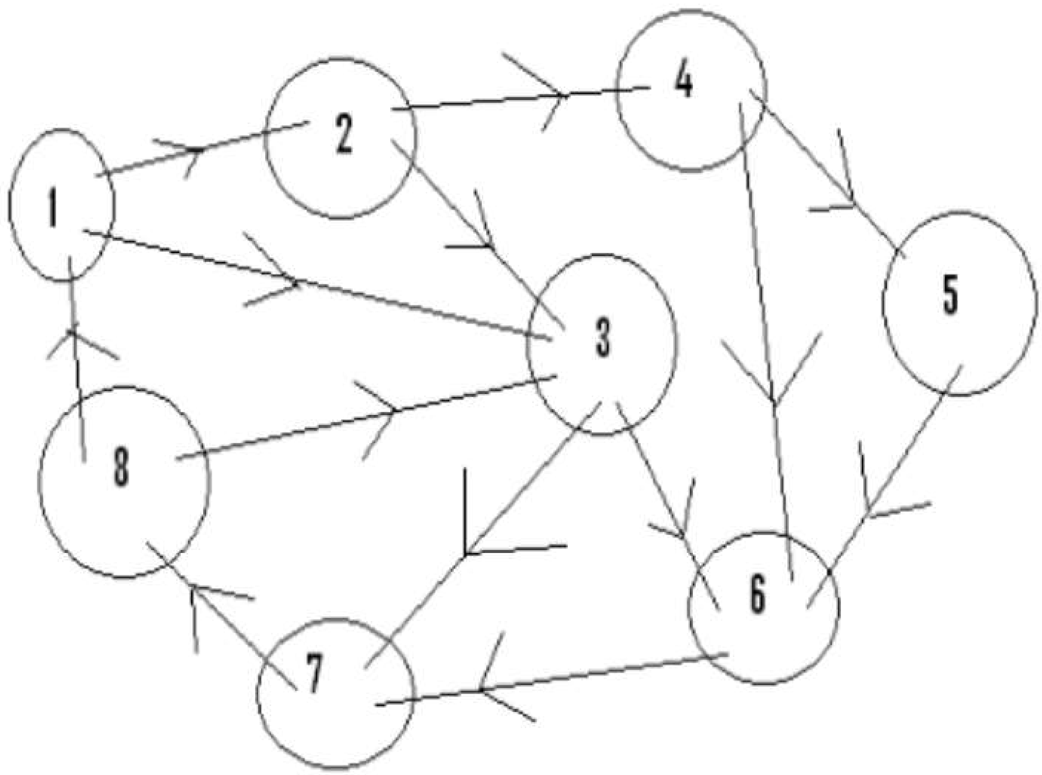
A partir disso, apresentamos o programa feito, bem como os resultados obtidos, na seção de Anexos das página seguintes.

5 Anexos

In [1]:

```
import numpy as np
```

Exercício 3



In [ ]:

```

```

Pesos de acordo com a Representação:

$$\begin{cases} x_1 = x_8/2 \\ x_2 = x_1/2 \\ x_3 = x_1/2 + x_2/2 + x_8/2 \\ x_4 = x_2/2 \\ x_5 = x_4/2 \\ x_6 = x_3/2 + x_4/2 + x_5/1 \\ x_7 = x_3/2 + x_6/1 \\ x_8 = x_7/1 \end{cases}$$

In [ ]:

```

```

In [ ]:

```

```

Criando a matriz A, conforme a representação anterior:

In [2]:

```
A = np.array([
    [.0, .0, .0, .0, .0, .0, .0, .5],
    [.5, .0, .0, .0, .0, .0, .0, .0],
    [.5, .5, .0, .0, .0, .0, .0, .5],
    [.0, .5, .0, .0, .0, .0, .0, .0],
    [.0, .0, .0, .5, .0, .0, .0, .0],
    [.0, .0, .5, .5, 1., .0, .0, .0],
    [.0, .0, .5, .0, .0, 1., .0, .0],
    [.0, .0, .0, .0, .0, .0, 1., .0]
])
```

In [3]:

```
A
```

Out[3]:

```
array([[0. , 0. , 0. , 0. , 0. , 0. , 0. , 0.5],
       [0.5, 0. , 0. , 0. , 0. , 0. , 0. , 0. ],
       [0.5, 0.5, 0. , 0. , 0. , 0. , 0. , 0.5],
       [0. , 0.5, 0. , 0. , 0. , 0. , 0. , 0. ],
       [0. , 0. , 0. , 0.5, 0. , 0. , 0. , 0. ],
       [0. , 0. , 0.5, 0.5, 1. , 0. , 0. , 0. ],
       [0. , 0. , 0.5, 0. , 0. , 1. , 0. , 0. ],
       [0. , 0. , 0. , 0. , 0. , 0. , 1. , 0. ]])
```

Criando a matriz pertubada (Perron-Frobenius)

Valor do m, fixado no trabalho em 0.15

In [4]:

```
m_ = 0.15
```

Matriz S (n x n), com todas as entradas iguais a 1 / n:

In [5]:

```
x = np.ones((8, 8), dtype=float)
S = x * 1 / 8
S
```

Out[5]:

```
array([[0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125],
       [0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125],
       [0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125],
       [0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125],
       [0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125],
       [0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125],
       [0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125],
       [0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125]])
```

Cáculo da matriz M:

In [6]:

```
M = ((1 - m_) * A) + (m_ * S)
```

In [7]:

```
for row in M:
    print(row)
```

```
[0.01875 0.01875 0.01875 0.01875 0.01875 0.01875 0.01875 0.44375]
[0.44375 0.01875 0.01875 0.01875 0.01875 0.01875 0.01875 0.01875]
[0.44375 0.44375 0.01875 0.01875 0.01875 0.01875 0.01875 0.44375]
[0.01875 0.44375 0.01875 0.01875 0.01875 0.01875 0.01875 0.01875]
[0.01875 0.01875 0.01875 0.44375 0.01875 0.01875 0.01875 0.01875]
[0.01875 0.01875 0.44375 0.44375 0.86875 0.01875 0.01875 0.01875]
[0.01875 0.01875 0.44375 0.01875 0.01875 0.86875 0.01875 0.01875]
[0.01875 0.01875 0.01875 0.01875 0.01875 0.01875 0.86875 0.01875]
```

In [ ]:

Verificando se a soma das colunas da matriz M é igual a 1:

In [8]:

```
np.sum(M, axis=0)
```

Out[8]:

```
array([1., 1., 1., 1., 1., 1., 1., 1.])
```

In [ ]:

In [ ]:

Obtendo:

c = max 1≤j≤n | 1 − 2 min 1≤i≤n Mij |

In [9]:

```
c = np.max(1 - 2 * np.amin(M, axis=0))
```

In [10]:

```
c
```

Out[10]:

```
0.9625
```

Erro a posteriori:

|| kx − x^(k) || ≤ (c / (1 − c) * || x^k − x^(k-1) ||)

In [11]:

```
def calculate_error(x_k_new, x_k_old):
    result = c / (1 - c) * np.sum(np.abs(x_k_new - x_k_old))

    return result
```

Realizando os cáculos para obtenção do resultado.

A função nos traz a matriz, o erro e a quantidade de iterações necessárias para a obtenção do resultado.

In [12]:

```
def calculate(X_, matriz_a, epslon=10e-5):

    cont = 1

    error = np.inf

    x_k_new = X_
    while True:
        # print(f'erro: {error}')

        x_k_old = x_k_new
        x_k_new = np.dot(matriz_a, x_k_old)

        error = calculate_error(x_k_new, x_k_old)

        if error < epslon:
            break

        cont += 1

    return cont, error, x_k_new
```

In [ ]:

Conforme o enunciado, utilizaremos o $X^{(0)} = 1/n$ para iniciar os cálculos.

In [13]:

```
X_0 = np.array([
    1/8,
    1/8,
    1/8,
    1/8,
    1/8,
    1/8,
    1/8,
    1/8
])
```

In [14]:

```
contador, error, matriz = calculate(X_0, M)
```

In [15]:

```
print(f'Foram realizadas {contador} iterações e o erro final foi de {error}')
```

```
Foram realizadas 24 iterações e o erro final foi de 6.41347701652746e-05
```

In [16]:

```
print(f'A matriz resultante foi: \n')
print('{')
for row in matriz:
    print(row)
print('}')
```

```
A matriz resultante foi: 

{
0.10564241611753487
0.06364823391396042
0.17759124170116974
0.045800591669674415
0.038215085398377376
0.14617428821693
0.2184747527797174
0.20445339020263512
}
```

In [17]:

```
print(f'As páginas exibidas por ordem de importância são: \n{matriz.argsort()[::-1] + 1}')
```

```
As páginas exibidas por ordem de importância são: 
[7 8 3 6 1 2 4 5]
```

6 Referências Bibliográficas

- Anton & Rorres, Álgebra Linear com Aplicações, 10ª ed, Bookman, 2012
- Material do Professor Nelson - *Projeto 5 - Como o Google Ordena Páginas*