

1 Primo Esercizio

1.1 Primo obiettivo

Come primo obiettivo ci poniamo di studiare come varia il numero di iterazioni in funzione del parametro γ del PageRank.

In particolare si discuterà se questa variazione è legata alla densità di nonzeri.

Abbiamo fissato $n = 100000$ e 5 diversi valori della densità di nonzeri ($\frac{10}{n}, \frac{5}{n}, \frac{2}{n}, \frac{1}{n}, \frac{0.5}{n}$).

Per ciascun valore di densità abbiamo generato una matrice H random ed abbiamo lanciato la funzione di PageRank con valori di $\gamma = [0.85, 0.90, 0.95, 0.99, 0.999]$.

1.1.1 Script e function

Si riporta la funzione utilizzata per generare il vettore di PageRank.

```
% H: matrice di adiacenza
% v: vettore di personalizzazione
% gamma: parametro di combinazione convessa
% itmax: massimo numero di iterazione
% y: vettore di PageRank
% it: numero di iterazioni
function [y,it] = PageRank(H, v, gamma, itmax)
n = size(H,1);
usn = 1/n; e = ones(n,1);
d = H*e; d = d';
dang = d==0;
dh = d + dang*n; dh=1./dh;
x = rand(1,n); x=x/sum(x);
v = v/sum(v);
for it=1:itmax
y = x.*dh;
y = y*H + usn*sum(dang.*x);
y = y*gamma+(1-gamma)*v;
err = max(abs(x-y));
x = y;
%disp([it,err])
if err<1.e-13*max(x)
break
end
end
```

Per studiare il numero di iterazioni al variare di γ abbiamo realizzato uno script che utilizza la funzione del PageRank con valori di γ differenti e calcola il numero di iterazioni necessarie. Il valore di densità in questo caso è $\frac{0.5}{n}$. Tale script genera una matrice di adiacenza H in modo casuale e la passa alla funzione PageRank che viene chiamata con diversi valori della densità γ .

```
%PageRank1 punto 1
```

```
%vedere come il numero di iterazioni vari in funzione di gamma
%e capire se la variazione è legata alla densità di nonzeri
```

```
n=100000; %dimensione della matrice
dens=0.5/n; %densità di nonzeri
H = sprand(n,n,dens) ~ = 0; %matrice casuale sparsa nxn con densità dens di nonzeri
v=ones(1,n); %vettore di n componenti uguali a 1
```

```
[y,it1]=PageRank(H,v,0.85,1000);
[y,it2]=PageRank(H,v,0.90,1000);
[y,it3]=PageRank(H,v,0.95,1000);
[y,it4]=PageRank(H,v,0.99,1000);
[y,it5]=PageRank(H,v,0.999,1000);
```

1.1.2 Tabella

| | $\frac{10}{n}$ | $\frac{5}{n}$ | $\frac{2}{n}$ | $\frac{1}{n}$ | $\frac{0.5}{n}$ |
|-------|----------------|---------------|---------------|---------------|-----------------|
| 0.85 | 26 | 38 | 62 | 59 | 173 |
| 0.90 | 27 | 40 | 68 | 65 | 263 |
| 0.95 | 28 | 44 | 79 | 74 | 525 |
| 0.99 | 29 | 46 | 85 | 79 | 1000 |
| 0.999 | 29 | 46 | 87 | 84 | 1000 |

Tabella 1: In questa tabella si riportano i dati ottenuti col primo script, nella prima colonna sono riportati i valori di γ utilizzati mentre nella prima riga sono riportati i valori di densità utilizzati.

1.1.3 Commenti

Si noti che 1000 è il massimo di iterazioni consentite nel codice utilizzato. A meno di variazioni casuali, i dati della tabella confermano ciò che potevamo aspettarci dalla teoria, ovvero che il numero di iterazioni eseguite dal programma per calcolare il vettore di PageRank cresce proporzionalmente alla densità di nonzeri.

1.2 Secondo obiettivo

Il secondo obiettivo consiste nello studiare come il vettore di permutazione associato al PageRank sia legato alla scelta del parametro arbitrario γ .

Per far questo, tenuta fissa la dimensione n della matrice ($n = 100000$) calcoliamo il vettore di PageRank per due diversi valori di $\gamma = [0.85, 0.99]$ e ne calcoliamo le relative permutazioni, dunque, calcoliamo il vettore w che ha nella posizione i -esima l'indice tale che $\sigma_1(w_i) = \sigma_2(i)$.

1.2.1 Script

Si riporta il secondo script usato nella sperimentazione, tale script genera una matrice di adiacenza H , la passa alla funzione PageRank con 2 valori di γ diversi dopodiché calcola le permutazioni associate e traccia il grafico del vettore w tale che $w(i)$ è l'indice i per cui $\sigma_1(w_i) = \sigma_2(i)$.

```
%PageRank1 punto 2
%vedere come la permutazione associata al PageRank vari in funzione di gamma

n=100000; %dimensione della matrice
dens=0.5/n; %densità di nonzeri
H = sprand(n,n,dens) ~ = 0; %matrice sparsa nxn di densita' dens di non zeri
v=ones(1,n); %vettore di n componenti uguali a 1
[y1,it1]=PageRank(H,v,0.85,1000);
[y2,it2]=PageRank(H,v,0.99,1000);
[z1,sigma1]=sort(y1,'descend'); %calcola sigma1
[z2,sigma2]=sort(y2,'descend'); %calcola sigma2
for i=1:n
    w(i) = find(sigma1==sigma2(i)); %trova gli indici di non zeri
end
base=[1:n];
plot(base(1:n),w(1:n)) %traccia il grafico
```

1.2.2 Grafici

Riportiamo di seguito i grafici ottenuti dalle prime 1000 componenti di w (figura 1), poi dalle prime 10000 (figura 2) e infine il grafico ottenuto dall'intero vettore (figura 3) w :

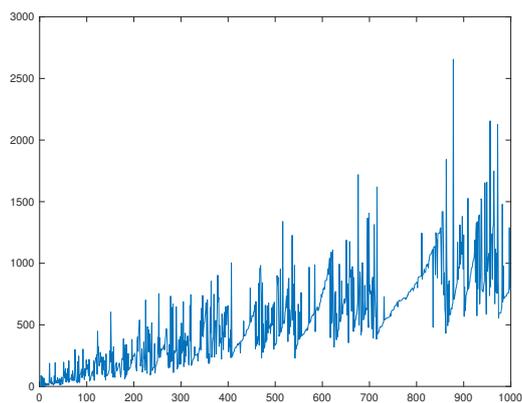


Figura 1: Prime 1000 componenti del vettore w

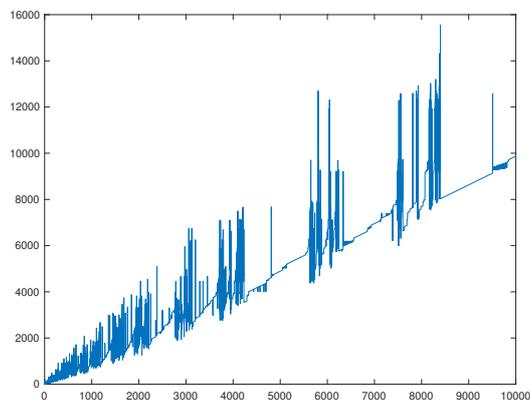


Figura 2: Prime 10000 componenti del vettore w

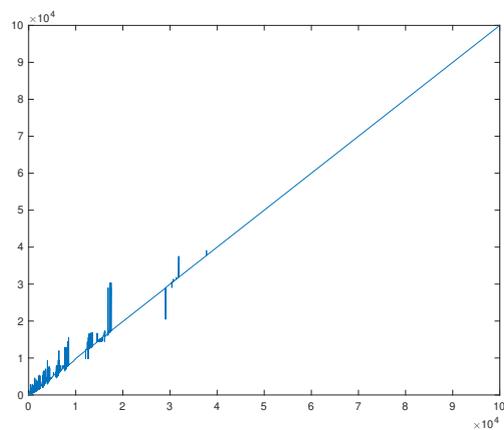


Figura 3: Tutte le componenti del vettore w

1.2.3 Commenti

Per eseguire la sperimentazione si è usato un valore di n pari a 100000.

Da tali grafici si evince che più il grafico risulta "vicino" alla bisettrice (cioè alla retta $w_i = i$) più i vettori di permutazione che abbiamo calcolato sono simili, quindi le due permutazioni coincidono maggiormente sulle componenti iniziali e finali di w mentre si diversificano per le componenti intermedie. Questo implica che i vettori di PageRank così calcolati sono abbastanza d'accordo su quali siano le pagine più importanti e quali, invece, siano le meno rilevanti mentre sono abbastanza in disaccordo su come ordinare le pagine di rilevanza intermedia.

2 Secondo Esercizio

In questo esercizio si vuole testare su una rete reale l'algoritmo di PageRank precedentemente implementato. In particolare testeremo l'efficienza e "l'affidabilità" di questi algoritmi sulla rete Berkeley-Stanford.

Di seguito riportiamo una rappresentazione grafica della rete in questione.

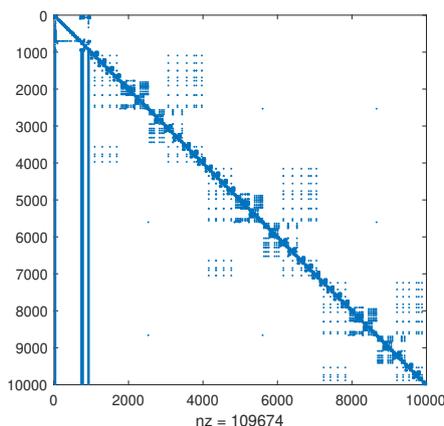


Figura 4: Matrice di adiacenza della rete di Berkeley-Stanford

2.1 Primo obiettivo

Per la matrice della rete Berkeley-Stanford valutiamo quante iterazioni e quanto tempo di CPU occorrono per i valori di γ pari a [0.85, 0.90, 0.95, 0.99, 0.999]

2.1.1 Script e function

```
%PageRank2 punto 1
```

```
G= load('web-BerkStan.txt'); %carico la matrice della rete Berkeley-Stanford
```

```
%costruiamo da questa matrice G la matrice di adiacenza H
```

```
n=685230;
```

```
H = sparse(G(:,1), G(:,2), ones(size(G,1),1),n,n);
```

```
spy(H(1:10000,1:10000)) %riportiamo la figura della struttura della sottomatrice principale di testa d
```

```
v=ones(1,n);
```

```
%calcoliamo iterazioni e tempi
```

```
tic
```

```
[y,it1]=PageRank(H,v,0.85,1000);
```

```
t1=toc;
```

```
tic
```

```
[y,it2]=PageRank(H,v,0.90,1000);
```

```
t2=toc;
```

```
tic
```

```
[y,it3]=PageRank(H,v,0.95,1000);
```

```
t3=toc;
```

```
tic
```

```
[y,it4]=PageRank(H,v,0.99,1000);
t4=toc;
tic
[y,it5]=PageRank(H,v,0.999,1000);
t5=toc;
```

2.1.2 Tabella

| γ | Iterazioni | Tempo(s) |
|----------|------------|----------|
| 0.85 | 168 | 3.2942 |
| 0.90 | 262 | 5.1717 |
| 0.95 | 549 | 10.5990 |
| 0.99 | 1000 | 19.2814 |
| 0.999 | 1000 | 19.5869 |

Tabella 2: In questa tabella si riportano i dati ottenuti con il primo script dell'esercizio PageRank2

2.1.3 Commenti

Come notiamo il numero di iterazioni e dunque i tempi di calcolo crescono al crescere di γ per il motivo discusso nell'esercizio Pagerank1.

2.2 Secondo obiettivo

Si vuole ora mettere in relazione l'importanza attribuita ad una pagina dal PageRank con il numero di link che puntano a questa pagina, per stabilire se questi possono o meno essere presi come stima realistica dell'importanza della pagina stessa.

2.2.1 Script e function

```
%PageRank 2 punto 2
%Mettere in relazione il valore del PageRank di ciascuna pagina col numero di link che entrano,
%o che escono, da quella pagina per vedere se è vero che una pagina che riceve, o che contiene,
%multi link è mediamente più importante delle altre pagine.

G= load('web-BerkStan.txt');
n=685230;
H = sparse(G(:,1), G(:,2), ones(size(G,1),1),n,n);
v=ones(1,n); %vettore di personalizzazione
y = PageRank(H, v, 0.85, 1000); %vettore y di PageRank
e = ones(n,1);
yout = H*e; %calcolare il vettore yout che ha per componente i-esima il numero di link che escono dall.
yout = yout/sum(yout); %normalizzo
yin = e'*H; %calcolare il vettore yin che ha per componente i-esima il numero di link che entrano nell.
yin = yin/sum(yin); %normalizzo
base=[1:n];
%calcolare i rapporti componente a componente tra il vettore y di PageRank e i vettori rispettivamente
%ordino i vettori ottenuti
yin = sort(yin./y, 'descend');
yout = sort(yout'./y,'descend');
plot(base, yin,'r', base, yout, 'b'); %traccio il grafico
```

2.2.2 Grafici

.

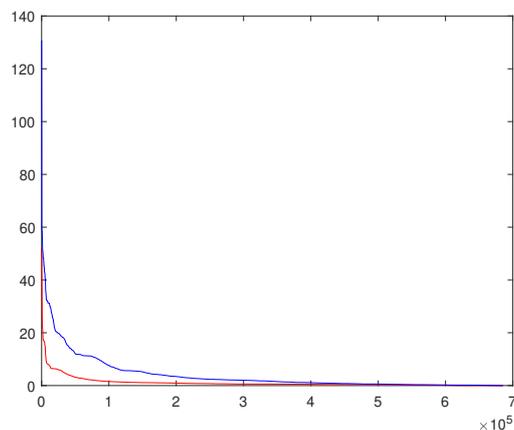


Figura 5: In rosso si riporta il grafico del rapporto link entranti/importanza PageRank. In blu si riporta il grafico del rapporto link uscenti/importanza PageRank

2.2.3 Commenti

Il numero di link che puntano ad una pagina può essere assunto come una stima indicativa, per quanto non precisa, dell'importanza della pagina stessa; questo è in accordo con l'idea intuitiva con cui è stata introdotta la teoria del calcolo della centralità delle reti. Invece il numero dei link in 'uscita' è una stima per nulla affidabile e assolutamente non indicativa dell'importanza di una pagina.

2.3 Terzo obiettivo

Infine, vogliamo ripetere il calcolo fatto nel paragrafo 1.2 per questa specifica rete focalizzandoci sui primi 20 elementi del vettore w sopra descritto.

2.3.1 Script e function

```
%PageRank2 punto 3
%Mettere in relazione la permutazione ottenuta con
%gamma= 0.85 con la permutazione ottenuta ponendo
%gamma = 0.99, come si è fatto nell'esercizio1punto1.
%Valutare come cambiano le prime 20 posizioni

G= load('web-BerkStan.txt');
n=685230;
H = sparse(G(:,1), G(:,2), ones(size(G,1),1),n,n);
dens=0.5/n;
v=ones(1,n); %vettore di n componenti uguali a 1
[y1,it1]=PageRank(H,v,0.85,1000);
[y2,it2]=PageRank(H,v,0.99,1000);
[z1,sigma1]=sort(y1,'descend'); %calcola sigma1
[z2,sigma2]=sort(y2,'descend'); %calcola sigma2
for i=1:20
    w(i) = find(sigma1==sigma2(i)); %trova gli indici di non zeri
end
base=[1:20];

plot(base(1:20),w(1:20),'o') %traccia il grafico

%per analizzare meglio l'idea dell'allineamento con la bisettrice, %eseguire il seguente comando
%in sostituzione al plot precedente.
%plot(base(1:20),w(1:20), 'o'), axis([0 20 0 20])
```

2.3.2 Grafici

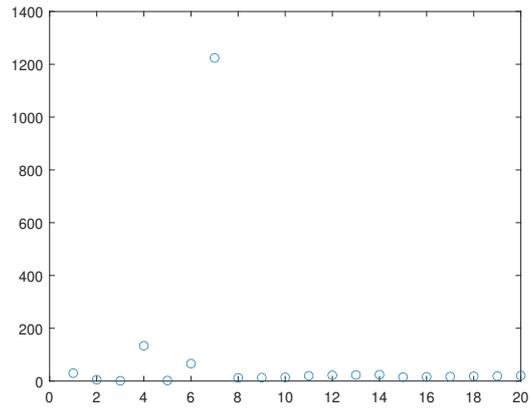


Figura 6: Grafico integrale dei primi 20 elementi

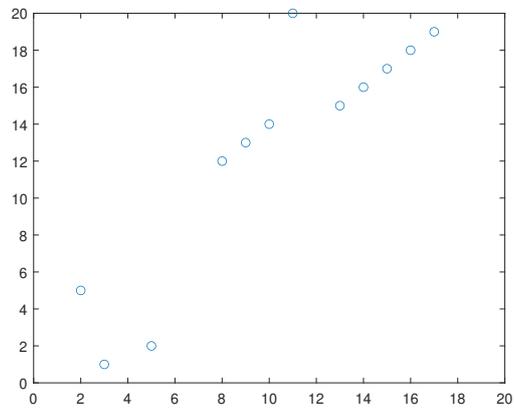


Figura 7: Grafico con asse y limitato a $[0, 20]$

2.3.3 Commenti

I risultati ottenuti confermano, anche in questo caso 'reale', quanto visto prima, ovvero, a meno di eccezioni dovute a fattori casuali, le prime 20 posizioni del PageRank variano poco per i due valori di γ considerati (in particolare il grafico riportato si allinea secondo le aspettative alla bisettrice). Per motivi di dimensioni si è riportato lo stesso grafico in due scale differenti in modo da analizzare meglio l'idea dell'allineamento con la bisettrice.

note: Ho lavorato con la mia collega Chiara Di Sano