Il problema del PageRank

Dario A. Bini,
appunti del corso di Calcolo Scientifico

9 ottobre 2019

Sommario

Descriviamo le idee generali e alcuni dettagli dei possibili modelli matematici che permettono di attribuire un valore di importanza alle pagine presenti sul Web indipendentemente dal valore del loro contenuto e in funzione unicamente delle interconnessioni tra le pagine. Modelli di questo tipo sono usati dai motori di ricerca in forme diverse con implementazioni e varianti che sono coperte da segreto industriale e non ci è dato di conoscere. Attualmente la ricerca sui modelli di *Page Rank* è molto attiva ed intensa.

1 Introduzione

Quando utilizziamo un motore di ricerca tipo Google per avere informazioni su un certo argomento ci viene fornita in risposta una lista numerosa di pagine, generalmente migliaia o centinaia di migliaia, che contengono le parole chiave che abbiamo richiesto. Queste pagine vengono ordinate in base alla loro importanza in modo che nei primi posti troviamo quelle che sono certamente più significative e in fondo alla lista si trovano quelle pagine che non hanno una grande rilevanza. In questo modo il motore di ricerca ci permette di evitare di passare in rassegna tutte le migliaia di pagine, impresa che sarebbe umanamente impossibile.

Ma come viene stabilito se una pagina è più importante di un'altra? Con quale criterio vengono ordinate le pagine senza dover entrare dentro il loro contenuto?

Nei motori di ricerca di molti anni fa l'importanza veniva calcolata in base al numero di volte con cui la parola cercata compariva nei documenti presenti nella pagina. Per cui in testa alla lista venivano messi i documenti che contenevano il numero più alto di occorrenze della parola cercata e in fondo alla lista i documenti che contenevano una volta sola la parola chiave. Questo criterio sembrava rispondere pienamente alle esigenze di allora Questo metodo si rivelò però inefficiente e vulnerabile. Ad esempio poteva capitare infatti che dopo l'uscita del film Titanic, digitando "Leonardo Di Caprio" in un motore di ricerca apparissero nella parte alta della lista dei siti "a luci rosse". Infatti i gestori di questi siti, conoscendo la modalità di ordinamento, avevano inserito dei file nelle loro pagine che contenevano il nome "Leonardo Di Caprio" milioni di volte

raggiungendo lo scopo di essere ben pubblicizzati. Sono stati Sergey Brin e Larry Page fondatori di Google, a rivoluzionare il modo di attribuire un rango alle pagine del Web indipendentemente dal loro contenuto. La loro idea si basa su un modello matematico particolare e utilizza la teoria di Perron-Frobenius delle matrici non negative. Questa teoria risale ai primi del 1900 quando il mondo di internet non veniva nemmeno immaginato dai più brillanti scrittori di fantascienza. Naturalmente sia Oskar Perron che Georg Frobenius, matematici tedeschi, quando hanno inventato il teorema che va sotto il loro nome non pensavano lontanamente alle applicazioni che esso avrebbe avuto in futuro. La consistenza del modello e l'esistenza e unicità della soluzione è infatti garantita dal teorema di Perron-Frobenius.

Il problema del calcolo della soluzione è un aspetto non trascurabile della questione. La soluzione infatti può essere vista come l'autovettore dominante di una matrice di N righe e di N colonne dove N è uguale al numero di pagine esistenti sul Web. Attualmente il valore di N è di circa 10 miliardi. Se usassimo i metodi standard per risolvere questo problema, pur usando i più veloci computer disponibili attualmente, dovremmo aspettare milioni di anni prima di conoscere la soluzione. Il metodo di calcolo dell'importanza delle pagine web che viene attualmente usato si basa su un adattamento del metodo delle potenze che viene chiamato algoritmo di PageRank.

In questo breve articolo presentiamo le idee generali e alcuni dettagli dei possibili modelli matematici che permettono di attribuire un valore di importanza alle pagine presenti sul Web indipendentemente dal valore del loro contenuto e in funzione unicamente delle interconnessioni tra le pagine.

Descriviamo poi il metodo delle potenze, adattato per sfruttare le specificità di questo problema e ne diamo una sua implementazione nel Octave.

I dettagli sulle effettive implementazioni di modelli di questo tipo, usati dai motori di ricerca per ordinare le pagine web, sono coperti da segreto industriale e non ci è dato di conoscere. Attualmente la ricerca sui modelli di PageRank è molto attiva ed intensa e presenta dei problemi computazionali particolarmente stimolanti visto le dimensioni elevate da trattare e visto l'interesse commerciale del problema.

Il metodo del PageRank è usato anche in altri contesti per valutare il prestigio delle riviste scientifiche, il prestigio di istituzioni di ricerca, la qualità dei lavori scientifici pubblicati valutati indipendentemente dal loro contenuto.

In questa nota si fa riferimento al libro [1] di Langville e Meyer per una trattazione più specifica e approfondita, e al recente articolo di rassegna [2] per applicazioni e generalizzazioni.

2 Il modello

Assumiamo di avere N pagine in rete e numeriamole con gli interi da 1 a N. Tanto per dare un ordine di grandezza, N è attualmente all'incirca 10 miliardi. Per descrivere il World-Wide Web è utile usare un grafo orientato in cui i nodi rappresentano le pagine presenti sul Web e gli archi orientati descrivono le

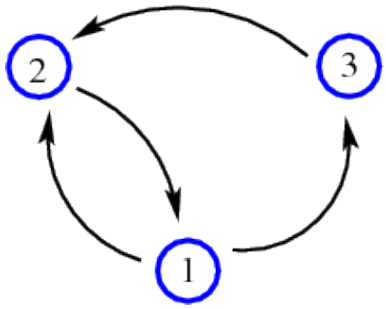

Figura 1: Grafo associato ad un Web costituito da 3 pagine

connessioni di tali pagine. Più precisamente un arco collega il nodo i col nodo j se la pagina i ha un *link* che punta alla pagina j.

Ad esempio se il nostro WWW fosse fatto da 3 pagine in cui la pagina 1 punta alla 2 e alla 3, la pagina 2 punta alla 1 e la 3 punta alla 2, allora il grafo sarebbe quello dato in figura 1

Un grafo orientato può essere univocamente descritto da una *matrice di adiacenza* $H = (h_{i,j})$ di dimensione $N \times N$ in cui $h_{i,j} = 1$ se c'è un arco orientato che collega il nodo i col nodo j (se la pagina i contiene un link alla pagina j); mentre $h_{i,j} = 0$ altrimenti.

La matrice di adiacenza associata al grafo di sopra è

$$H = \begin{bmatrix} 0 & 1 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix} \tag{1}$$

Ecco alcuni possibili criteri per definire l'importanza di una pagina:

1. una pagina è più importante se ha un numero maggiore di link ad altre pagine;

2. una pagina è importante se riceve un numero maggiore di link da altre pagine.

Si vede subito che il criterio 1 non è valido. Se così fosse, basterebbe riempire la propria pagina di un numero arbitrariamente grande di link ad altre pagine per renderla più importante.

Anche il secondo criterio, sebbene più sensato, non è immune da truffa. Non è infatti complicato costruire un numero arbitrario di pagine fittizie che contengono un link alla propria pagina per poterla rendere più importante. Inoltre in un modello sensato non dovrebbe dare troppa importanza essere puntati da tante pagine di livello trascurabile mentre sarebbe più rilevante essere puntati da (poche) pagine di importanza elevata.

Un criterio più corretto che cattura queste ultime considerazioni è il seguente:

una pagina i che punta altre pagine, ad esempio $j_1, j_2, \ldots, j_k$, distribuisce la sua importanza in parti uguali alle pagine $j_1, j_2, \ldots, j_k$, e quindi dà $1/k$ della sua importanza alle pagine che punta.

In questo modello, se denotiamo con $d_i = \sum_{j=1}^{n} h_{i,j}$, supponendo, $d_i \neq 0$ per $i = 1, \ldots, N$, e se indichiamo con w_j l'importanza della pagina j, vale allora

$$w_j = \sum_{i=1}^{N} w_i \frac{h_{i,j}}{d_i}, \quad j = 1, \ldots, N.$$

Ad esempio nel caso dell'esempio (1) si ha

$$\begin{cases} w_1 = w_2 \\ w_2 = \frac{1}{2} w_1 + w_3 \\ w_3 = \frac{1}{2} w_1 \end{cases}$$

Si osserva che questo non è altro che un problema di autovalori e autovettori formulato nel seguente modo. Posto $e = (1, 1, \ldots, 1)^T$, $d = (d_i) = He$, e $D = \mathrm{diag}(d)$ si ha

$$w^T M = w^T, \quad M = D^{-1} H$$

dove $w^T = (w_1, \ldots, w_N)$.

2.1 Alcuni problemi

Elenchiamo alcuni problemi che si incontrano in questa formulazione.

1. Cosa succede se $d_i = 0$ per qualche i? Questo succede nei casi in cui ci sono pagine che non puntano a nulla. Il problema non è insolito, infatti ci possono essere pagine quali un file postscript che non hanno link a nulla. I nodi che hanno questa caratteristica sono chiamati *dangling nodes*.

2. Esiste sempre una soluzione?

3. La soluzione è unica (a meno di multipli scalari)?

4. La soluzione è positiva?

5. Come si può calcolare?

Si osserva che i dangling nodes sono individuati per il fatto che essi corrispondono alle righe di H con tutti gli elementi nulli. Per poter trattare il caso in cui esistano dei dangling nodes si introduce una leggera modifica al modello. Più precisamente si sostituisce la matrice iniziale di adiacenza H con una nuova matrice $\widehat{H}$ che coincide con H dappertutto eccetto che nelle righe tutte nulle in cui gli elementi di $\widehat{H}$ vengono posti tutti uguali a 1. Dal punto di vista modellistico è come assumere salomonicamente che un documento che nel modello originale non cita nessun altro documento nel web, nel nuovo modello

modificato va a citare tutti i documenti presenti. Quindi distribuisce $1/N$ della sua importanza uniformemente a tutti.

La matrice $\widehat{H}$ viene quindi scritta come

$$\widehat{H} = H + ue^T \tag{2}$$

dove u è il vettore con componente 1 in corrispondenza dei dangling nodes e con componente zero altrove.

In seguito denoteremo con M la matrice

$$M = \widehat{D}^{-1}\widehat{H}, \quad \widehat{D} = \mathrm{diag}(\widehat{d}), \quad \widehat{d} = \widehat{H}e. \tag{3}$$

Possiamo dare subito risposta affermativa alla domanda 2 osservando che $Me = e$ e quindi 1 è autovalore, quindi w è un qualsiasi autovettore sinistro corrispondente all'autovalore 1.

Per rispondere alle altre domande dobbiamo riportare alcuni risultati classici della teoria di Perron-Frobenius delle matrici non negative.

3 Alcuni risultati teorici

Riportiamo il teorema di Perron-Frobenius

Teorema 1 *Sia A una matrice $n \times n$ di elementi non negativi. Allora esiste un autovalore λ di A tale che $\lambda = \rho(A) \geq 0$. Esistono un autovettore destro x e sinistro y corrispondenti a λ con componenti non negative. Se inoltre A è irriducibile allora λ è semplice e gli autovettori x e y hanno componenti positive. Se infine A ha elementi positivi allora λ è l'unico autovalore di modulo massimo.*

Si osserva che in base al teorema di P-F ogni soluzione ha sempre componenti non negative come è giusto che sia. Però la sola condizione di nonnegativita non garantisce l'unicità della soluzione (a meno di multipli scalari). Mentre con la condizione di irriducibilità la soluzione è unica.

È facile costruire reti di pagine interconnesse che hanno una matrice di adiacenza riducibile. Quindi il modello così come è stato introdotto non è ancora adeguato.

Si osserva ancora che nel caso di matrici irriducibili e non negative possono esistere altri autovalori che hanno lo stesso modulo del raggio spettrale. Questo crea dei seri problemi dal punto di vista algoritmico come vedremo tra poco.

Esempio: La matrice

$$\begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

ha autovalori $1, \underline{i}, -1, -\underline{i}$ dove $\underline{i}$ è l'unità immaginaria tale che $\underline{i}^2 = -1$.

Matrici come quella dell'esempio precedente vengono dette cicliche. Il numero di autovalori distinti con modulo uguale a $\rho(A)$ è detto indice di ciclicità. Si può dimostrare che se la matrice di adiacenza H è irriducibile allora $M = D^{-1}H$ ha indice di ciclicità k se tutti i cammini chiusi nel grafo associato ad H hanno lunghezze che sono multipli interi di k.

Il teorema di P-F esclude però che esistano blocchi di Jordan, relativi al raggio spettrale, di dimensione maggiore di 1.

4 Modifica del modello

Per far fronte ai problemi discussi, il modello del PageRank descritto nella precedente sezione viene così modificato. La matrice M viene sostituita con la matrice

$$A = \gamma M + (1 - \gamma)ev^T, \quad 0 < \gamma < 1, \tag{4}$$

dove v è un arbitrario vettore a componenti positive tale che $v^T e = 1$, detto vettore di personalizzazione e γ è un parametro, di solito si sceglie $\gamma = 0.85$. In questo modo la matrice A ha elementi positivi. La soluzione quindi esiste, è unica (a meno di multipli) e $\rho(A)$ è l'unico autovalore di modulo 1.

Dal punto di vista modellistico è come se l'importanza di una pagina fosse ripartita in due parti: una frazione γ viene distribuita in base ai link come nel modello originale, la frazione complementare $1 - \gamma$ viene distribuita a tutte le altre pagine secondo un criterio dato dal vettore v. Se ad esempio $v = (1/n)e$ allora la distribuzione è fatta in modo uniforme a tutte le pagine del Web.

5 Aspetti computazionali

Poichè si conosce l'autovalore 1, il calcolo di un autovettore corrispondente comporterebbe la risoluzione di un sistema di equazioni lineari $N \times N$. Date le dimensioni del problema, è impensabile applicare metodi generali per la risoluzione di questo sistema. Tali metodi hanno un costo $O(N^3)$ e richiederebbero tempi di risoluzione superiori alla vita dell'universo anche usando i supercomputer più potenti al momento disponibili. Infatti, con $N = 10^{10}$ risulta $N^3 = 10^{30}$. Supponendo di usare uno dei computer più veloci esistenti, il *K-computer* della Fujitsu che ha una velocità di 8.16 petaflops (8.16×10^{15} operazioni) al secondo, occorrerebbero almeno 163 milioni di anni per il calcolo.

5.1 Il metodo delle potenze

Nel corso delle lezioni avete incontrato il *metodo delle potenze* per approssimare l'autovalore di modulo massimo e un autovettore corrispondente. Questo metodo richiede ad ogni passo di calcolare il prodotto matrice-vettore. Nel nostro caso questo prodotto ha un costo molto contenuto poichè il numero di elementi non

nulli presenti su ogni riga di A non corrispondente a un dangling node è molto basso rispetto a N. Infatti esso è il numero di link che partono dalla pagina corrispondente alla riga in questione, e generalmente le pagine web non hanno migliaia di link uscenti.

Il problema è capire se il metodo delle potenze è applicabile sotto le nostre ipotesi.

Rivisitiamo il metodo delle potenze per il problema specifico del PageRank. Si osserva che poiché A ha elementi positivi allora per il teorema di P-F 1 è l'unico autovalore di modulo massimo ed è semplice. Quindi per la forma canonica di Jordan di A vale:

$$S^{-1}AS = J = \left[\begin{array}{c|c} 1 & 0 \\ \hline 0 & \widetilde{J} \end{array}\right]$$

con $\rho(\widetilde{J}) < 1$. Si osserva ancora che, poiché $Je_1 = e_1$, $e_1^T J = e_i^T$, dove e_1 è il primo vettore della base canonica, risulta $ASe_1 = Se_1$, $e_1^T S^{-1} A = e_1^T S^{-1}$. Cioè, $e = Se_1$ è autovettore destro di A mentre $w = e_1^T S^{-1}$ è autovettore sinistro di A. In particolare e è il vettore di tutti uni.

Ora si osserva che

$$J^k = \left[\begin{array}{c|c} 1 & 0 \\ \hline 0 & \widetilde{J}^k \end{array}\right].$$

Inoltre, poiché $\rho(\widetilde{J}) < 1$ risulta $\lim_k \widetilde{J}^k = 0$ quindi $\lim_k J^k = e_1 e_1^T$, Da cui

$$\lim_k (A)^k = Se_1 e_1^T S^{-1} = ew^T$$

Si osserva che la convergenza è tanto più veloce quanto più piccolo è $\rho(\widetilde{J})$., cioè il secondo autovalore di A più grande in modulo.

Per approssimare w non è necessario calcolare le potenze di A, ma è possibile applicare il metodo delle potenze nella seguente forma:

$$y_k^T = x_k^T A, \quad x_{k+1} = y_k/||y_k||, k = 0, 1, \ldots,$$

a partire da un vettore iniziale x_0 tale che $||x_0|| = 1$. Nel nostro caso conviene scegliere $||\cdot|| = ||\cdot||_1$. Infatti se $||x||_1 = 1$ anche $y = Ax$ è tale che $||y||_1 = 1$ e quindi non c'è bisogno di normalizzare ad ogni passo. Questo si dimostra facilmente essendo $||y||_1 = y^t e = x^t Ae = x^t e = 1$. Allora il metodo diventa

$$x_{k+1}^T = x_k^T A, \quad , k = 0, 1, \ldots,$$

cioè $x_k^T = x_0^T A^k$. Quindi la successione x_k converge all'autovettore $w > 0$ tale che $w^t e = 1$.

Si osserva che nelle ipotesi di positività di A questo limite non dipende dal vettore iniziale x_0 scelto.

5.2 Proprietà degli autovalori e velocità di convergenza

È evidente che tanto più piccolo è $\rho(\widetilde{J})$ e tanto più velocemente la successione $\widetilde{J}^k$ converge a 0 e quindi tanto più velocemente la successione x_k generata dal

metodo delle potenze converge al vettore PageRank. Ci chiediamo ora che ruolo ha il parametro γ nella piccolezza di $\rho(\widetilde{J})$. Per questo denotiamo con $\lambda_1 = 1, \lambda_2, \ldots, \lambda_n$ gli autovalori della matrice M in (3) e con $\mu_1 = 1, \mu_2, \ldots, \mu_n$ gli autovalori della matrice A in (4), dove abbiamo ordinato gli autovalori in modo che $|\lambda_i| \geq |\lambda_j|$, $|\mu_i| \geq |\mu_j|$ per $i < j$. Per il teorema di Perron-Frobenius si ha $1 = \mu_1 > |\mu_2|$ mentre $1 = \lambda_1 \geq |\lambda_2|$.

Osserviamo innanzitutto che in generale può accadere che $|\lambda_2| = 1$ e quindi, se applicassimo il metodo delle potenze direttamente alla matrice M non avremmo convergenza, oppure il limite potrebbe essere diverso a seconda della scelta del vettore iniziale. Ciò può accadere ad esempio se la matrice M è riducibile oppure se la matrice è ciclica.

Un esempio del primo caso è dato dalla matrice

$$M = \frac{1}{2}\begin{bmatrix} 1 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}$$

che ha due autovalori uguali a 1 e due autovalori nulli. Con questi valori la successione x_k converge (in un solo passo) a un vettore del tipo (a, a, b, b) con $a + b = 1/2$, dove a e b dipendono dalla scelta di x_0.

Un esempio del secondo caso è dato dalla matrice ciclica

$$M = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{bmatrix}$$

che ha tutti gli autovalori di modulo 1 e la successione M^k, formata da tutte matrici di permutazione, non ha limite. In questo caso la successione degli x_k ottenuta scegliendo $x_0^T = (a, b, c, d)$ è data da $x_1^T = (d, a, b, c)$, $x_2^T = (c, d, a, b)$, $x_3^T = (b, c, d, a)$, $x_4^T = (a, b, c, d)$ che si ripetono ciclicamente.

Per mettere in relazione i valori λ_i con i valori μ_i è utile richiamare un risultato classico noto come teorema di Brauer.

Teorema 2 (Brauer) *Sia B una matrice $n \times n$ di elementi complessi con autovalori $\lambda_1, \ldots, \lambda_n$, sia inoltre $v_1 \in \mathbb{C}^n$, $v \neq 0$ tale che $Bv = \lambda_1 v$. Sia $A = B + vu^T$ dove $u \in \mathbb{C}^n$. Allora la matrice A ha autovalori $\mu_1, \ldots, \mu_n$ tali che $\mu_1 = \lambda_1 + u^T v$, $\mu_i = \lambda_i$, $i = 2, \ldots, n$.*

La dimostrazione data da Brauer consiste nel verificare prima che $Av = Bv + vu^T v = (\lambda_1 + u^T v)v$. Come secondo passo si considera un autovalore $\lambda_i \neq \lambda_1$ e un suo autovettore sinistro w, cioè tale che $w^H B = \lambda_i w^H$. Si calcola $w^H A = w^H B + w^H u v^T$ e si ottiene $\lambda_i w^H$ essendo $w^H u = 0$. Quest'ultima proprietà deriva dal fatto che autovettori sinistri e autovettori destri di una matrice corrispondenti ad autovalori distinti sono ortogonali. A questo punto il teorema sarebbe dimostrato se la matrice B avesse tutti autovalori distinti.

Brauer completa la dimostrazione usando un ragionamento di continuità essendo una matrice con autovalori multipli il limite di una successione di matrici con autovalori distinti.

Una dimostrazione più semplice che non richiede il ragionamento per continuità, si ottiene portando B in forma normale di Schur $T = Q^H BQ$, dove $Q^H Q = I$ e T è triangolare superiore con gli autovalori $\lambda_1, \ldots, \lambda_n$ sulla diagonale. Si osserva che, avendo scelto v normalizzato in modo che $v^H v = 1$, vale $Qe_1 = v$, $e_1 = Q^H v$, dove e_1 è il primo versore della base canonica. Si ha allora

$$Q^H AQ = T + Q^H v u^T = T + e_1 u^T.$$

La correzione $e_1 u^T$ altera l'elemento diagonale di posto $(1,1)$ ma non altera gli altri elementi diagonali, per cui $\mu_i = \lambda_i$ per $i = 2, \ldots, n$.

Il teorema di Brauer è utile in tutti quei casi in cui vogliamo spostare un solo autovalore di una matrice, conoscendo la corrispondente autocoppia, mediante una correzione di rango 1. Una di queste applicazioni è infatti il modella del PageRank. Possiamo allora applicare il teorema di Brauer alla matrice A in (4) ottenendo immediatamente il seguente

Corollario 1 *Gli autovalori λ_i della matrice M e gli autovalori μ_i della matrice A sono tali che $\mu_1 = \lambda_1$, $\mu_i = \gamma\lambda_i$, $i = 2, \ldots, n$.*

Ne segue che più piccolo è il valore di $\theta \in (0,1)$ e più veloce è la convergenza del metodo delle potenze applicato alla matrice modificata A.

Un'altra osservazione interessante è che se $w^T A = w^T$ allora da (4) segue che

$$w^T(I - \gamma M) = (1 - \gamma)v^T \tag{5}$$

per cui il problema del calcolo di w può essere visto come il problema di risolvere un sistema lineare con matrice $I - \gamma M$. Si osserva che essendo $\rho(M) \leq 1$ e $\gamma \in (0,1)$, la matrice del sistema è invertibile.

5.3 Metodi iterativi per sistemi lineari

Il calcolo del vettore w di PageRank lo possiamo realizzare risolvendo il sistema lineare (5) con un qualsiasi metodo iterativo. Naturalmente escludiamo l'uso dei metodi diretti per le eccessive dimensioni del sistema. Possiamo ad esempio prendere in considerazione il metodo di Richardson e una modifica del metodo di Gauss-Seidel. Ricordiamo che il metodo di Richardson per risolvere un sistema del tipo $x^T B = f^T$ utilizza l'iterazione

$$x_{k+1}^T = x_k^T + \alpha(f^T - x_k^T B)$$

che combina ad ogni passo l'iterazione corrente x_k^T col residuo $f^T - x_k^T B$ usando un parametro α. La matrice di iterazione di questo metodo è chiaramente $P_\alpha = I - \alpha B$, per cui se $\rho(I - \alpha B) < 1$ il metodo è convergente. Nel nostro caso, abbiamo $B = (I - \gamma M)$ per cui $P_\alpha = (1 - \alpha)I + \alpha\gamma M$. Se $\alpha \leq 1$ allora la matrice P_α è non negativa, la somma lungo le righe vale $1 - \alpha + \alpha\gamma$ che coincide

col raggio spettrale. Il minimo di questa quantità si ha per $\alpha = 1$ e vale $\gamma < 1$. Scrivendo l'iterazione abbiamo

$$x_{k+1}^T = x_k^T + (1-\gamma)v^T - x_k^T(I - \gamma M) = \gamma x_k M + (1-\gamma)v^T.$$

Confrontando con (4) si scopre che abbiamo ottenuto nuovamente il metodo delle potenze.

Nel caso di α generale si avrebbe il metodo

$$x_{k+1}^T = x_k^T + \alpha(1-\gamma)v^T - \alpha x_k^T(I - \gamma M) = (1-\alpha)x_k^T + \alpha(\gamma x_k M + (1-\gamma)v^T).$$

Consideriamo ora una modifica del metodo di Gauss-Seidel. Per questo partizioniamo la matrice $I - \gamma M = I - \gamma(D^{-1}(H + ue^T)$, $D = \mathrm{diag}(d)$, $d = (H + ue^T)e$, come

$$I - \gamma M = -\gamma D^{-1}ue^T + (I - \gamma \mathrm{triu}(D^{-1}H)) - \gamma \mathrm{tril}(\gamma D^{-1}H, -1)$$

dove abbiamo usato le notazioni di Matlab per cui $\mathrm{triu}(Z)$ denota la parte triangolare superiore della matrice Z, mentre $\mathrm{tril}(Z, -1)$ denota la parte strettamente triangolare inferiore di Z. Introduciamo il metodo iterativo definito da

$$x_{k+1}^T = \left[(1-\gamma)v^T + \gamma x_k^T D^{-1}(ue^T + \mathrm{tril}(H, -1))\right] \mathrm{triu}(I - \gamma D^{-1}H)^{-1} \qquad (6)$$

la cui convergenza dipende dal valore di

$$\gamma\rho(D^{-1}(ue^T + \mathrm{tril}(H, -1)))\mathrm{triu}(I - \gamma D^{-1}H)^{-1}).$$

Non ci imbarchiamo nell'analisi del raggio spettrale di questa matrice, cercheremo di studiare computazionalmente questa iterazione.

6 Interpretazione probabilistica

È possibile dare una interpretazione probabilistica del modello del PageRank. Occorre per questo introdurre il concetto di catena di Markov. Lo facciamo in modo informale attraverso degli esempi.

Esempio. Random walk. Una particella si muove sulla retta reale con le seguenti regole:

- può occupare le posizioni di ascissa intera $1, 2, 3, \ldots, n$;
- ad ogni istante, se il posto che occupa è diverso da 1 e da n, può spostarsi a destra di un posto con probabilità d, può spostarsi a sinistra di un posto con probabilità s, può rimanere ferma nella posizione che occupa con probabilità $1 - s - d \geq 0$;
- se il posto che occupa è 1 allora può spostarsi solo a destra con probabilità d o rimanere ferma con probabilità $1 - d$;

- se il posto che occupa è n allora può spostarsi solo a sinistra con probabilità s o rimanere ferma con probabilità $1 - s$;

All'istante iniziale la particella si trova nella posizione i Qual è la probabilità che dopo k passi la particella si trovi nella posizione j?

È possibile dare risposta al quesito in modo semplice. Si definisce la matrice $P = (p_{i,j})$ in cui $p_{i,j}$ è la probabilità che la particella ad un generico istante si sposti dalla posizione i alla posizione j. Si assume che $s, d \neq 0$ in modo che la matrice è irriducibile. Vale quindi

$$P = \begin{bmatrix} 1-d & d & & & & 0 \\ s & 1-s-d & d & & & \\ & \ddots & \ddots & \ddots & & \\ & & s & 1-s-d & d & \\ 0 & & & s & 1-s & \end{bmatrix}$$

Si introduce il vettore $x(t) = (x_i(t))$ in cui $x_i(t)$ è la probabilità che al tempo t la particella si trovi in posizione i. Per le leggi elementari di composizione delle probabilità si deduce che

$$x(t+1)^T = x(t)^T P$$

Quindi, assegnato $x(0)$ è possibile calcolare $x(t)$ attraverso prodotti matrice-vettore. Si osservi che $x(k)^T = x(0)^T P^k$ e, poiché la matrice P è irriducibile per il teorema di Perron Frobenius esiste un autovettore sinistro positivo che corrisponde all'autovalore $\rho(T) = 1$. Inoltre se non esistono altri autovalori di modulo uguali al raggio spettrale, risulta $\lim x(k)^T = \pi^T$, $\pi^T = \pi^T P$.

Una matrice di probabilità, o stocastica, come è P e un insieme di n stati (particella che può occupare n posizioni diverse) con la proprietà che se $x(t)$ è il vettore di probabilità di stato al tempo $x(t)$ allora $x(t+1)^T = x(t)^T P$, costituiscono una *catena di Markov*.

Una situazione analoga si ha col metodo del PageRank. Infatti la matrice A può essere vista come una matrice di probabilità che descrive il comportamento casuale di un navigatore virtuale che ad ogni istante cambia pagina con le seguenti regole

- con probabilità γ decidere di scegliere un link presente nella pagina e la scelta viene fatta con probabilità uniforme;

- con probabilità $1 - \gamma$ sceglie una pagina non necessariamente presente tra i link della pagina corrente, scegliendo a caso tra tutti i link del web con probabilità data dal vettore di personalizzazione v.

La componente i-esima del valore limite π^T del vettore di PageRank descrive la probabilità che dopo un "tempo infinito" il navigatore virtuale si trovi nella pagina i. Risultano quindi più importanti le pagine che hanno maggior probabilità di essere visitate.

7 Implementazione

Vogliamo ora progettare del software che realizzi il metodo delle potenze nel caso del problema del PageRank. Utilizziamo per questo il linguaggio Octave che permette di memorizzare matrici in forma sparsa.

Un modo conveniente di memorizzare matrici $n \times n$ che hanno un numero di elementi non nulli molto più piccolo di n è quello di registrare per ciascun elemento i suoi indici di riga e di colonna assieme al suo valore. In questo modo l'ingombro di una matrice sparsa $n \times n$ con m elementi non nulli è di soli 2 vettori di interi lunghi m con le coordinate di ciascun punto e un vettore di m componenti con i valori degli elementi non nulli. Di fatto Octave usa questa modalità di codifica con la variante del *compressed column format* che permette una maggiore efficienza. Si rimanda al manuale di Octave in rete per maggiori informazioni su questo tipo di formato.

In questo modo il costo del prodotto $y = x^T H$ tra un vettore riga x^T per la matrice di adiacenza sparsa è molto basso perché viene svolto secondo questo schema dove `i` e `j` sono i vettori con le coordinate degli elementi non nulli della matrice di adiacenza

```
for k=1:m
   y(j(k))=y(j(k))+x(i(k));
end
```

In ogni caso non dobbiamo implementare questa operazione perché viene automaticamente svolta quando si invoca il comando

```
w=v*H;
```

dove `v` è il vettore riga e `H` la matrice di adiacenza.

In Octave esiste il comando

```
sprand(m,n,d)
```

che genera una matrice casuale sparsa $m \times n$ di elementi con valori casuali compresi tra 0 e 1 in cui il numero di elementi diversi da 0 è circa $d * n^2$. Una matrice di adiacenza casuale $n \times n$ con densità di non zeri pari a d si può quindi generare col comando

```
H=sprand(n,n,d)~=0;
```

Se vogliamo avere un'idea di come sono distribuiti gli elementi non nulli basta scrivere `spy(H)` e si ottiene una descrizione grafica come in figura 2

Il passo fondamentale dell'algoritmo è il calcolo di $y^T = x^T A$ dato x. Tenendo presente le (2), (3), (4), si ottiene la formula

$$y^T = \gamma(x^T \widehat{D}^{-1} H + \frac{1}{n}(\sum_{i \in Dang} x_i)e^T) + (1-\gamma)(x^T e)v^T$$

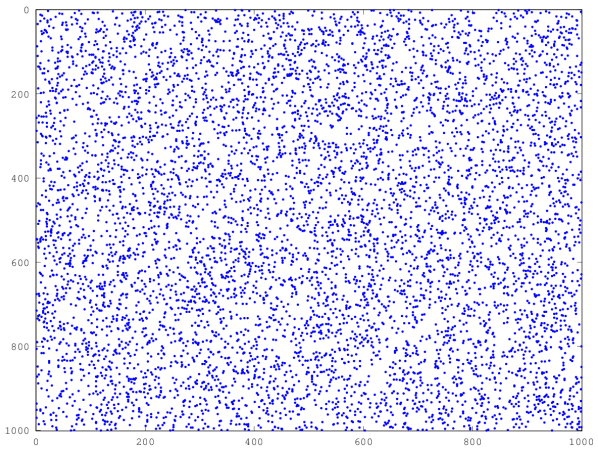

Figura 2: Rappresentazione grafica di una matrice sparsa. I puntolini rappresentano elementi non nulli.

dove H è la matrice di adiacenza, e, posto $d = He$, si denota con $\widehat{D}$ la matrice diagonale che ha i valori d_i^{-1} sulla diagonale se $d_i \neq 0$, e n altrimenti (questo valore è ininfluente poiché va a moltiplicare le righe nulle della matrice di adiacenza H). Inoltre $Dang$ è l'insieme di indici i che corrispondono ai nodi dangling, cioè tali che $d_i = 0$.

Il programma che implementa il metodo è riportato nel Listing 1.

Per calcolare la permutazione p dato il vettore di PageRank y basta scrivere `[v,p]=sort(y)`. Se `nomi` è la variabile vettoriale di caratteri che contiene i nomi dei siti che vogliamo ordinare, allora `nomi_ordinati=nomi(p)` è il vettore con i nomi ordinati in ordine di importanza

Proviamo a vedere come salgono i tempi di calcolo aumentando le dimensioni n della matrice di adiacenza H avendo un numero di elementi non nulli dato all'incirca da $10n$.

Svolgendo l'esperimento con un laptop Dell con processore i5 e arrestando l'iterazione quando la componente di massimo modulo della differenza di due iterate consecutive è minore di `1.e-13` si hanno iterazioni e tempi mostrati nella tabella1.

Si vede chiaramente come con un semplice laptop sia possibile trattare dimensioni importanti in pochi secondi. È interessante vedere come variano i tempi prendendo una matrice di adiacenza H ancora più sparsa, ad esempio con un fattore $d = 1/n$ in modo che ci sia alta probabilità di avere numerosi dangling nodes.

Possiamo vedere ora come l'ordinamento del PageRank sia influenzato dal parametro γ. Nel listing 2 è riportato lo script che svolge la sperimentazione. Lo script calcola due valori diversi del vettore di PageRank, uno per $\gamma = 0.85$, l'altro per $\gamma = 0.99$. Calcola le due permutazioni $\sigma_{0.85}(i)$ e $\sigma_{0.99}(i)$ così ottenute.

Listing 1: Programma PageRank

```
function y = PageRank(H, v, gamma, itmax)
% function [y, verr] = PageRank(H, v, gamma, itmax)
% Calcola il vettore y del PageRank applicato alla matrice di adiacenza H
% con vettore di personalizzazione v e parametro gamma
% applicando al piu' itmax iterazioni del metodo delle potenze
% In output il vettore verr contiene il massimo errore ad ogni passo
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 = norm(x-y,inf);
    x = y;
    verr(it) = err;
    if debug
       fprintf('it=%d, err=%d\n',it, err);
    end
    if err<1.e-13*max(y)
        break
    end
end
fprintf('numero di iterazioni: %d \n', it)
if it == itmax
    fprintf('raggiunto il numero massimo di iterazioni \n')
end
```

n	iterazioni	secondi
10.000	18	0.03
100.000	17	0.27
1.000.000	15	3.9
10.000.000	14	40.1

Tabella 1: Numero di iterazioni e tempi impiegati dalla function PageRank per ottenere un residuo inferiore a `1.e-13`

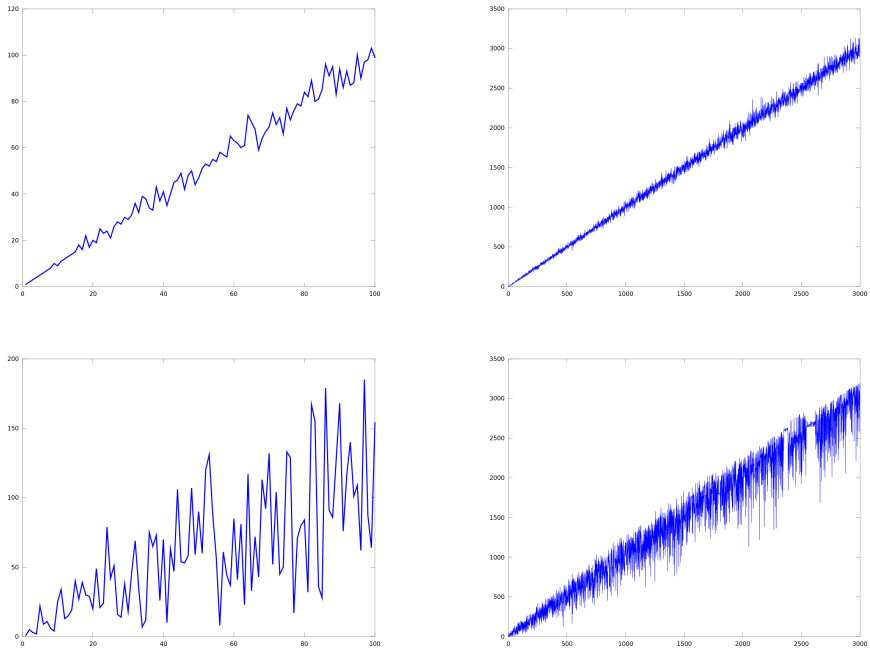

Figura 3: Scostamento degli ordinamenti ottenuti con i valori di $\gamma = 0.85$ e $\gamma = 0.99$. In alto si riporta il caso di una matrice H con alta densità di uni. In basso il caso di una matrice H con bassa densità di uni.

Infine traccia il grafico del vettore $w = (w_i)$ dove w_i è l'indice per cui $\sigma_{0.99}(w_i) = \sigma_{0.85}(i)$. Se le due permutazioni coincidessero allora si avrebbe $w_i = i$. Il discostamento di i da w_i fornisce lo spostamento di rango che la pagina i-esima nell'ordinamento di $\sigma_{0.85}$ ha subìto con l'ordinamento $\sigma_{0.99}$. Dai grafici della figura 3 si vede come questo spostamento sia abbastanza contenuto per le pagine più importanti, mentre cresce al diminuire del valore del PageRank. Inoltre lo spostamento è molto più contenuto per matrici di adiacenza con densità di uni maggiore.

Un'altra sperimentazione consiste nel 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, molti link è mediamente più importante delle altre pagine.

Per calcolare i vettori che hanno componente i-esima il numero di link che entrano e che escono dalla pagina i è sufficiente fare le somme lungo le righe e lungo le colonne di H. Per normalizzarli è sufficiente dividerli per la loro somma. Basta allora scrivere

```
outlink = H*e;
```

Listing 2: Script di sperimentazione

```
% Si studia come il page rank dipende da gamma e dalla sparsita'
% Fissato un valore 'alto' di densita' di uni in H, ad esempio 70/n, si
% calcola il pagerank con due valori diversi di gamma ad es. 0.85 e 0.99
% per vedere come cambia l'ordinamento
% Si ripetere l'esperimento con un valore 'piccolo' di densita' di uni,
    ad es. 1/n
% come vettore di personalizzazione v si usa il vettore uniforme

% genero H
n = 10000;
d = 70/n;
H = sprand(n,n,d);
v = ones(n,1)'/n; % vettore di personalizzazione
itmax=100;
gamma = 0.85;
u = PageRank(H, v, gamma, itmax);
[su1,iu1] = sort(u, 'descend');
gamma = 0.99;
u = PageRank(H, v, gamma, itmax);
[su2,iu2] = sort(u, 'descend');
for i=1:n;
  idx(i) = find(iu2==iu1(i));
end
disp("indici delle coordinate delle prime 20 componenti del vettore
    pagerank ottenuto con gamma=0.85 nel vettore PageRank ottenuto con
    gamma=0.99 con v=(1,...,1)")
for i=1:20;
  disp([i,idx(i)]);
end
disp( 'Traccio il grafico');
plot(idx(1:100),'LineWidth',3)
```

Listing 3: Altro script di sperimentazione

```
% script di elaborazioni -2
% Si studia come il page rank sia legato al numero di link in ingresso o
    in uscita
% come vettore di personalizzazione v si usa il vettore uniforme
% genero H
n = 10000;
d = 70/n;
H = sprand(n,n,d);
v = ones(n,1)'/n; % vettore di personalizzazione
itmax=300;
gamma = 0.85;
u = PageRank(H, v, gamma, itmax);
e = ones(n,1);
outlink = H*e;
outlink = outlink/sum(outlink);
inlink = e'*H;
inlink = inlink/sum(inlink);
x=[1:n];
uin = sort(inlink./u, 'descend');
uout = sort(outlink'./u,'descend');
plot(x, uin,'b','LineWidth',3, x,uout,'r','LineWidth',3)
```

```
outlink = outlink/sum(outlink);
inlink = e'*H;
inlink=inlink/sum(inlink);
```

I rapporti `yout = outlink'./y` e `yin = inlink./y` ci danno un'idea di quanto i link in uscita e in entrata contribuiscano al PageRank. Più componenti di questi rapporti sono vicini a 1 e maggiore è la relazione che c'è tra queste quantità e il PageRank.

Per vedere graficamente i risultati ordiniamo in modo non crescente questi rapporti e poi tracciamo il grafico della n-upla così ottenuta.

Il programma che svolge questa analisi è riportato nel listing 3.

Nella figura 4 si riportano i grafici ottenuti nel caso di $n = 100000$ con densità pari a $70/n$ e $2/n$ uni in H. In rosso il grafico dei rapporti tra il Page Rank e il numero normalizzato di link uscenti, in blu l'analogo grafico per i link entranti. Si nota che il grafico blu è più piatto del grafico rosso ed è disposto non lontano dalla retta di equazione $y = 1$. Invece il grafico rosso sembra discostarsi maggiormente dal valore 1. Questo andamento è confermato dalla media artmetica di questi rapporti che nel caso di bassa densità dei nonzeri è di 1.6937 per i link uscenti e di, 1.0322 per i link entranti. Si noti come quest'ultimo valore sia più vicino ad 1. La varianza è 3.9870 per i link uscenti e 0.56705 per i link entranti. La varianza viene calcolata col comando `var(uin)` e `var(uout)`.

Nel caso di matrice con densità $d = 70/n$ si hanno media e varianza indicati

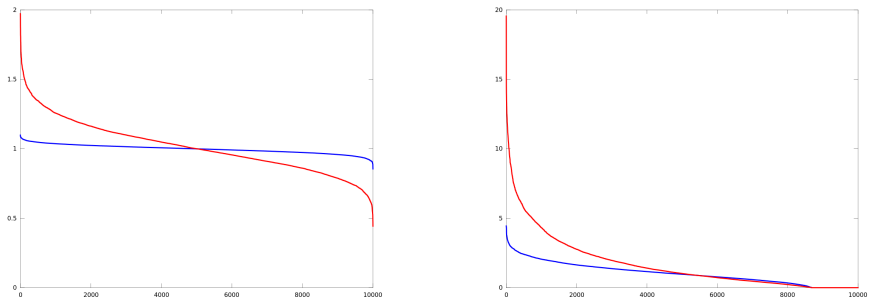

Figura 4: Rapporti tra numero di link rispettivamente uscenti (rosso) ed entranti (blu), normalizzati, col PageRank nel caso di densità di non xeri in H pari a $d = 70/n$ (figura di sinistra) e $d = 2/n$ (figura di destra).

in tabella

	inlink	outlink
media	0.99794	1.0140
varianza	9.2878e − 4	0.044262

Si può notare come la varianza sia molto più piccola nel caso dei link entranti. Questo significa che i link entranti hanno un maggior ruolo nel determinare il vettore PageRank.

Un'altra prova che possiamo fare è modificare H in modo che tutta una riga, ad esempio la prima, sia fatta di 1. Calcolare il vettore `y` e verificare che la prima componente di `y` non è quella massima. Poi possiamo provare a fare la stessa cosa con la prima colonna e scoprire che la prima componente di `y` è quella massima.

Cosa succede se le prime due colonne di H sono formate da uni? Cosa cambia al variare di γ?

Come può la scelta del vettore di personalizzazione v influenzare il PageRank? Si può verificare che le pagine corrispondenti alle componenti più grandi di v hanno un PageRank maggiore?

7.1 Risoluzione mediante sistema lineare

Proviamo ora a implementare un risolutore basato sull'applicazione di un metodo iterativo al sistema lineare (5). Utilizziamo l'iterazione definita in (6). Denotiamo $L = \gamma\text{tril}(H, -1)$, $U = \text{triu}(I - \gamma D^{-1}H)$ dove $D = \text{diag}(d)$, $d = (H + ue^T)e$. Allora possiamo riscrivere la (6) come

$$x_{k+1}^T = (1-\gamma)v^T + (y^T L + \gamma(y^T u)e)U^{-1}, \quad y^T = x_k^T D$$

Per risparmiare operazioni calcoliamo una volta per tutte il vettore $(1-\gamma)v^T$ e le matrici L ed U. La function che si ottiene è riportata nel listato 4.

Listing 4: Function per il calcolo del vettore PageRank basato su una variante del metodo di Gauss-Seidel

```
function [y, verr] = PageRankGS(H, v, gamma, itmax)
% function y = PageRankGS(H, v, gamma, itmax)
% Calcola il vettore y del PageRank applicato alla matrice di adiacenza H
% con vettore di personalizzazione v e parametro gamma applicando
% al piu' itmax iterazioni di una variante del metodo di Gauss-Seidel
% In output il vettore verr contiene il massimo errore ad ogni passo
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);
L = gamma*tril(H,-1);
U = speye(n)-diag(sparse(gamma*dh))*triu(H);
vg = (1-gamma)*v;

for it=1:itmax
    y = x.*dh;
    y = y*L + gamma*sum(dang.*y) + vg;
    y = y/U;
    err = norm(x-y,inf);
    if debug
       fprintf('it=%d, err=%d\n',it, err);
    end
    x = y; verr(it) = err;
    if err<1.e-13*max(y)
        break
    end
end
fprintf('numero di iterazioni: %d \n', it)
if it == itmax
    fprintf('raggiunto il numero massimo di iterazioni \n')
end
```

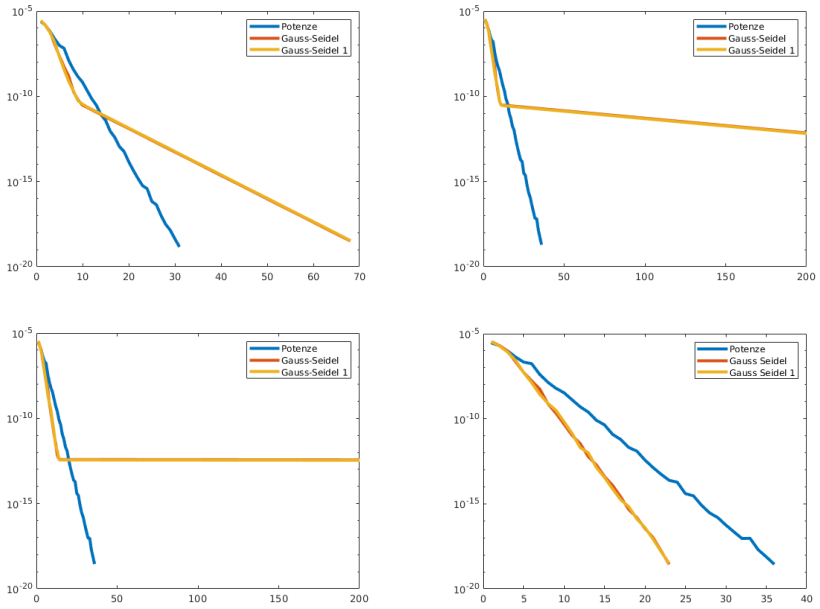

Figura 5: Massimo errore per passo dei metodi delle potenze, di Gauss-Seidel e di Gauss-Seidel con lo scambio di L e U per la matrice di Stanford con i valori $\gamma = 0.85, 0.99, 0.9999, 1$.

Una variante di questo metodo consiste nello scambiare i ruoli delle matrici U ed L questo si ottiene sostituendo la definizione di `L` e di `U` nella function scambiando "tril" e "triu" e sostituendo `(H,-1)` con `(H,1)`.

Proviamo a confrontare l'algoritmo basato sul metodo delle potenze con quello basato sulla variante dell'iterazione del metodo di Gauss-Seidel. Generiamo una matrice col comando `sprand` scegliendo $n = 1000000$, indice di sparsità pari a $7/n$ e scegliamo $\gamma = 0.85, 0.99, 0.9999, 1$. La figura 5 riporta i grafici del massimo errore per iterazione del metodo delle potenze, di Gauss-Seidel, e del metodo ottenuto scambiando i ruoli di L e U.

Si può notare come in questo caso ci siano minime differenze tra le due versioni del metodo di Gauss-Seidel. Inoltre i metodi basati sull'iterazione di Gauss-Seidel quando $\gamma < 1$, dopo un primo tratto in cui l'errore sembra diminuire più rapidamente rispetto al metodo delle potenze, presentano un andamento rettilineo dell'errore con una inclinazione che si riduce all'aumentare di γ. In questi casi il metodo delle potenze fornisce una convergenza più rapida. Nel caso $\gamma = 1$ entrambi i metodi convergono (per una matrice generata casualmente la probabilità che il secondo autovalore sia uguale a 1 è molto piccola) ma il metodo delle potenze risulta più lento rispetto agli altri due.

La situazione per matrice del web è sensibilmente diversa come si può vedere

dalla sperimentazione fatta nella prossima sezione.

7.2 Un caso reale

Finora abbiamo generato matrici di adiacenza in modo casuale. Queste matrici non catturano le caratteristiche tipiche del Web in cui si possono incontrare dei grappoli di pagine maggiormente interconnessi rispetto alle altre pagine e con connessioni meno frequenti tra loro. Si trovano sul web matrici di adiacenza reali ottenute analizzando parti del web. Ad esempio all'indirizzo `https://snap.stanford.edu/data/` si trova un ricco campionario di matrici di adiacenza provenienti da sottoreti del web.

Abbiamo scaricato il file `web-BerkStan.txt` che contiene la matrice dei link delle università di Stanford e di Berkeley che ha dimensione $n = 685230$ ed un numero di link pari a 7600595. Abbiamo anche scaricato la matrice più "leggera" `web-Stanford.txt` del web della sola università di Stanford nell'anno 2002 di dimensione $n = 281903$ e con un numero di link pari a 2312497

Col comando `G = load('web-BerkStan.txt');` si carica questa matrice che è formata da due colonne: la prima colonna contiene l'indice di riga di ciascun 1 mentre la seconda colonna contiene l'indice di colonna di ciascun 1. In testa al file ci sono alcuni commenti quali la provenienza, il numero dei nodi e degli archi. Le righe sono commentate col carattere "`#`" che viene riconosciuto da Octave ma non da Matlab. Per Matlab il carattere di commento è "`%`". Occorre quindi sostituire nel file i caratteri "`#`" con "`%`".

Fatto questo, con i seguenti comandi possiamo costruire da questa matrice `G` la matrice di adiacenza `H`

```
n = 685230;
H = sparse(G(:,1), G(:,2), ones(size(G,1),1), n, n);
```

In figura 6 si riporta la struttura della sottomatrice principale di testa di H di dimensione 10000 ottenuta col comando `spy(H(1:10000,1:10000))`.

In figura 7 si riporta l'analoga struttura per la matrice di Stanford dove si è selezionata la sottomatrice principale di testa di dimensione 20000.

Possiamo quindi fare sperimentazione. Guardiamo innanzitutto quante iterazioni sono richieste al variare di γ

γ	it	sec
0.75	94	4.0
0.80	121	5.2
0.85	168	7.4
0.90	262	11.0
0.95	549	23
0.99	2866	120.4

Dalla tabella si vede che il numero di iterazioni cresce rapidamente quando γ si avvicina ad 1.

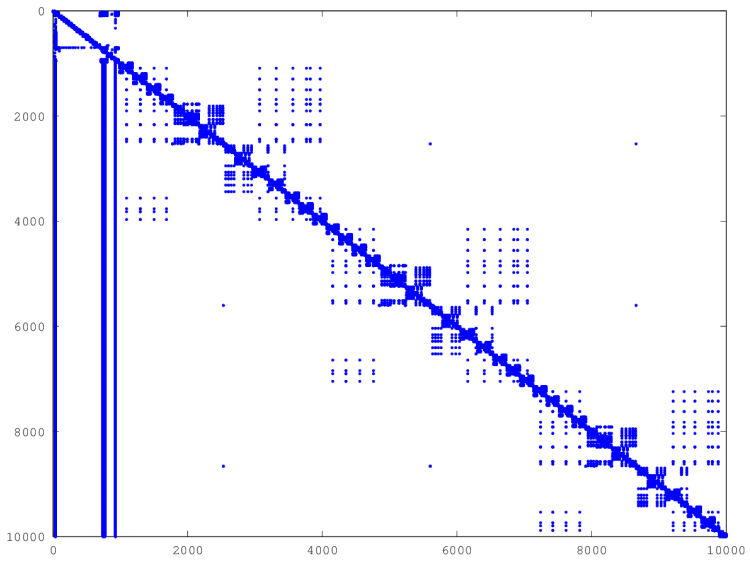

Figura 6: sottomatrice principale di testa di ordine 10000 della matrice H di Berkeley e Stanford

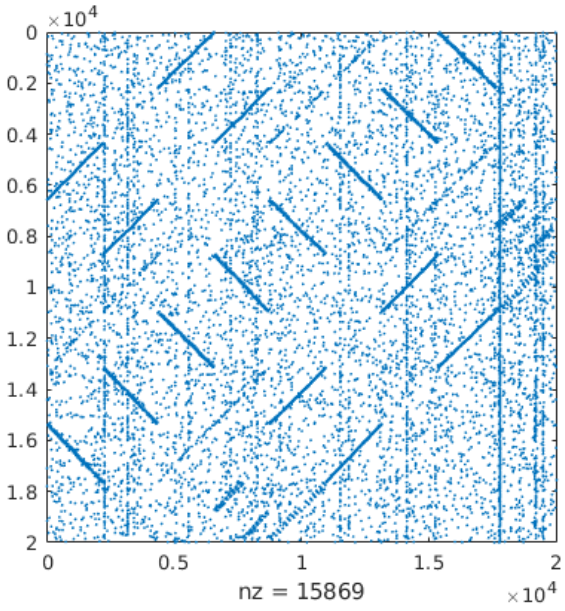

Figura 7: sottomatrice principale di testa di ordine 20000 della matrice H di Stanford

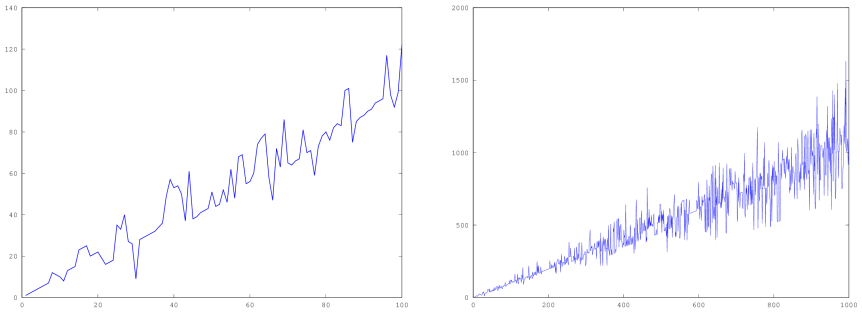

Figura 8: Grafico del vettore idx ottenuto con $\gamma = 0.85$ e $\gamma = 0.90$: prime 100 componenti a sinistra, prime 1000 componenti a destra

Possiamo similmente analizzare il rapporto tra il PageRank e il numero di link in ingresso e in uscita. Facendo il calcolo si ottiene questa tabella

	inlink	outlink
media	1.1476	4.0204
varianza	5.7747	48.741

Si nota la grande differenza con dei dati sintetici generati in modo casuale. Per questi dati reali la varianza è abbastanza più grande del caso di dati generati casualmente anche se la varianza relativa ai link in ingresso rimane sostanzialmente inferiore a quella relativa ai link in uscita.

Possiamo anche in questo caso valutare la dipendenza del vettore di PageRank al variare di γ. Riportiamo in figura 8 il grafico di idx nel caso in cui si sono scelti i valori di $\gamma = 0.85$ e $\gamma = 0.90$.

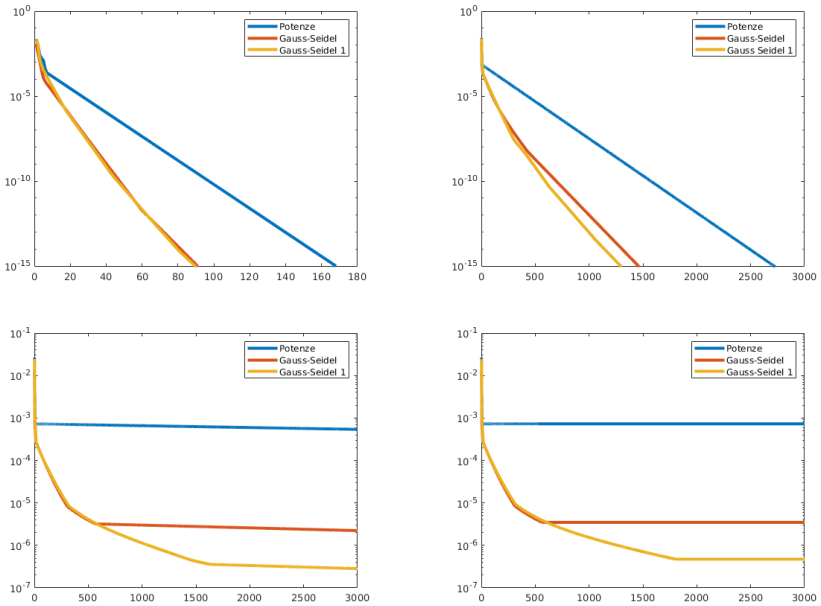

Figura 9: Massimo errore per passo dei metodi delle potenze, di Gauss-Seidel e di Gauss-Seidel con lo scambio di L e U per la matrice di Stanford con i valori $\gamma = 0.85, 0.99, 0.9999, 1$.

Confrontiamo sulla matrice di Stanford le iterazioni dei metodi delle potemze e di Gauss-Seidel nella versione originale e in quella dove i ruoli di U e L sono scambiati. La figura 9 riporta i valori del massimo errore in norma infinito a ciascuna iterazione.

Nell'andamento del grafico in scala logaritmica si nota una decrescita lineare a tratti. Sapreste dare una spiegazione di questo fatto?

Bibliografia

[1] Amy N. Langville, Carl D. Meyer, *Google's PageRank and Beyond: The Science of Search Engine Rankings*, Princeton University Press, Princeton, NJ, 2006.

[2] David F. Gleich, PageRank Beyond the Web, SIAM Review, 57,3, pp. 321–363, 2015.