Matrici Sparse

Salvatore Filippone
salvatore.filippone@uniroma2.it

1. Sistemi Lineari
$$Ax = b$$
2. Minimi Quadrati
$$\min \|b - Ax\|_2$$
3. Autovalori/autovettori:
$$Ax = \lambda x$$

- Gran parte del calcolo scientifico si riconduce alla soluzione di questi problemi;
- Costruire dei metodi efficienti richiede la comprensione delle strutture matematiche coinvolte;
- Costruire delle implementazioni efficienti per grandi dimensioni presenta molti problemi e richiede la comprensione delle strutture dati e delle architetture di calcolo coinvolte.

Matrici Sparse

> *Una matrice si dice sparsa quando contiene una quantità di zeri tale che sia conveniente sfruttarli nella rappresentazione macchina*

Di solito i coefficienti non-nulli sono $O(n)$ invece che n^2.

- Sono onnipresenti nel calcolo scientifico;
- Sono di implementazione molto problematica;
- Non hanno una rappresentazione unica, e non sono supportate nativamente nella maggior parte dei linguaggi;
- Richiedono compromessi tra efficienza e usabilità.

Come si risolve

$$Ax = b$$

quando A è sparsa?

Solutori iterativi classici (stazionari)

Si cerca una soluzione di $Ax = b$ iterando:

$$x_{k+1} = Bx_k + f$$

Jacobi :

$$A = L + D + U \Rightarrow B = -D^{-1}(L + U)$$

Gauss-Seidel:

$$A = L + D + U \Rightarrow B = -(L + D)^{-1}U$$

Nota: richiede

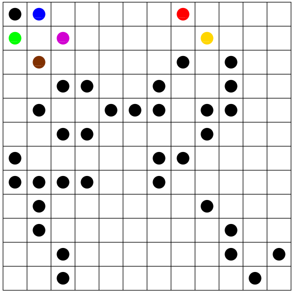
- Un prodotto matrice-vettore sparso;
- la soluzione di un sistema triangolare sparso;

Memorizzazione per COOrdinate :

M	Righe;
N	Colonne;
NZ	Non zeri;
IA(1:NZ)	Indici di riga;
JA(1:NZ)	Indici di colonna;
AS(1:NZ)	Coefficienti;

Nota: si ha normalmente per gli indici di riga $1 \leq IA(i) \leq M$, e analogamente per gli indici di colonna.

Memorizzazione COO



Elements Array

Col idx array	1	2	8	1	3	9	2	8	...
Row idx array	1	1	1	2	2	2	3	3	...

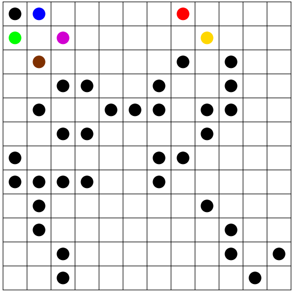
```
for(i=0; i<nz; i++){
  ir = ia[i];
  jc = ja[i];
  y[ir] = y[ir] + as[i]*x[jc];
}
```

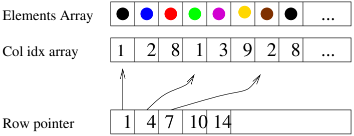
Costo 5 memory reads, 1 write e 2 flops per iterazione

Compressed Storage by Rows:

M	Righe;
N	Colonne;
IA(1:M+1)	Puntatori ad inizio riga;
JA(1:NZ)	Indici di colonna;
AS(1:NZ)	Coefficienti;

Strutture dati per matrici sparse






```c
for(i=0; i<m; i++){
  for(j=ia[i]; j<ia[i+1]; j++){
    y[i] = y[i] + as[j]*x[ja[j]];
  }
}
```

How many other storage schemes out there?

COO	ALIGNED_COO [84] SCOO [25] BRO-COO [88] BCCOO and BCCOO+ [100]	DIA	DDD-NAIVE, DDD-SPLIT [106], CDS [38], HDI [9]
CSR	CSR optimization [12, 10, 44, 33, 81, 105, 93, 73, 3, 42], ICSR [101], CSR-Adaptive [39], ACSR [6], SIC [34], RgC-SR [77], ArgCSR [46], BIN-CSR and BIN-BCSR [94], CMRS [53], PCSR [29], BCSR optimization [15, 21, 99, 91]	Hybrid	HYB [12], Combination of CSR and ELL [69, 70], HEC [63], TILE-COMPOSITE [104], SHEC [35], Cock-tail [85], HDC [103], Combination of ELLPACK and DIA [68], Combination of BCOO and BCSR [71], BRO-HYB [88], BRO-HYBR, BRO-HYBR(S) [87]
ELLPACK	ELLPACK-R [89], Sliced ELLPACK [72], Warped ELL [68], SELL-C-σ [55], SELL-P [5], ELLR-T [90], Sliced ELLR-T [31], HLL [9], BELLPACK [21], BSELL-PACK [85], AdELL [66], CoAdELL [67], JAD optimization [62], ELLPACK-RP [16], BiELL and BiJAD [107], BRO-ELL [88], Enhanced JDS [19], pJDS [54], ELL-WARP [98], BTJAD [1], BRC [7]	Other	CSC optimization [47], BLSI [74], CR-SD [86]

Dato un grafo

$$G = (\mathcal{V}, \mathcal{E}),$$

si può associargli una matrice (sparsa); ad esempio

$$A = (a_{ij}), \quad e_{ij} \in \mathcal{E} \Rightarrow a_{ij} = 1,$$

oppure, se gli archi sono *pesati* w_{ij} allora

$$A = (a_{ij}), \quad e_{ij} \in \mathcal{E} \Rightarrow a_{ij} = w_{ij},$$

Teorema

Perron-Frobenius: Data una matrice stocastica (transizioni di stato di una catena di Markov)

$$e^T Q = e^T, \qquad Q \geq 0 \qquad e = (1, 1, 1 \dots 1)^T$$

se Q irriducibile, allora

1. $\lambda = 1$ *è l'autovalore dominante (tutti gli altri sono più piccoli in modulo);*
2. *Esiste ed è unico l'autovettore $r > 0$ per l'autovalore dominante $\lambda = 1$*
$$Qr = \lambda r = r;$$
3. *r fornisce la distribuzione di probabilità stazionaria della catena di Markov*

Teorema

Perron-Frobenius: Data una matrice stocastica (transizioni di stato di una catena di Markov)

$$e^T Q = e^T, \qquad Q \geq 0 \qquad e = (1, 1, 1 \dots 1)^T$$

se Q irriducibile, allora

1. $\lambda = 1$ *è l'autovalore dominante (tutti gli altri sono più piccoli in modulo);*
2. *Esiste ed è unico l'autovettore $r > 0$ per l'autovalore dominante $\lambda = 1$*

$$Qr = \lambda r = r;$$

3. *r fornisce la distribuzione di probabilità stazionaria della catena di Markov*

A quale applicazione stiamo pensando?

L'autovettore da $ 25.000.000.000

L'autovettore da $ 25.000.000.000

$$\text{Google} = \text{crawling} + \text{matching} + \text{PageRank}$$

$$\text{Google} = \text{crawling} + \text{matching} + \text{PageRank}$$

Criterio di rilevanza di una pagina WEB:

La rilevanza della pagina i è la media pesata della rilevanza di tutte le pagine j che rinviano a i

$$r_i = \sum_j \frac{r_j}{N_j}$$

L'autovettore da \$ 25.000.000.000

$$\text{Google} = \text{crawling} + \text{matching} + \text{PageRank}$$

Criterio di rilevanza di una pagina WEB:

La rilevanza della pagina i è la media pesata della rilevanza di tutte le pagine j che rinviano a i

$$r_i = \sum_j \frac{r_j}{N_j}$$

Algoritmo di PageRank:

- Crea una lista di tutte le pagine WEB e dei loro collegamenti (grafo di connettività di WWW);
- Assegna un peso a ciascun collegamento e costruisci la matrice Q;
- Trova l'autovettore r;

Il valore dell'autovettore r nella posizione della pagina i viene assunto come misura della autorevolezza della pagina stessa.

L'autovettore da $ 25.000.000.000

Costruzione della matrice:

$$Q_{ij} = \begin{cases} 1/N_j & \text{se la pagina J manda alla I} \\ 0 & \text{altrimenti} \end{cases}$$

Costruzione della matrice:

$$Q_{ij} = \begin{cases} 1/N_j & \text{se la pagina J manda alla I} \\ 0 & \text{altrimenti} \end{cases}$$

Q va corretta con il "teletrasporto" che modella il fatto che se da una pagina non ci sono link uscenti, allora saltiamo (arbitrariamente) ad un'altra pagina a caso (e questo fra l'altro preserva la irriducibilità della matrice, ossia la connettività del grafo associato).

L’autovettore da \$ 25.000.000.000

Quindi: al cuore di Google c’è il calcolo di un autovettore di dimensione gigantesca;

L'autovettore da $ 25.000.000.000

Quindi: al cuore di Google c'è il calcolo di un autovettore di dimensione gigantesca; si calcola con il *metodo delle potenze*

Algorithm 2: Metodo delle potenze

$r^{(0)} \leftarrow r_0$;
for $k = 1, \ldots$ **do**
  $q^{(k)} \leftarrow Ar^{(k-1)}$;
  $r^{(k)} \leftarrow q^{(k)}/\|q^{(k)}\|$

- L'autovalore è noto a priori e vale 1;
- Richiede alcune decine di iterazioni, anche diversi giorni di calcolo!
- Se il vettore r_0 è normalizzato, allora si può risparmiare il passo di normalizzazione nel caso in cui $A = Q$ è stocastica.

Il metodo delle potenze

Se diagonalizzabile, la matrice Q si può decomporre in una somma di componenti ortogonali

$$Q = \sum_i \lambda_i Q_i$$

da cui si ottiene che moltiplicando ripetutamente per Q

$$Q^k v = \sum_i \lambda_i^k Q_i v = \sum_i \lambda_i^k v_i$$

In questa presentazione abbiamo assunto per semplicità che la matrice sia diagonalizzabile, ma il metodo vale in ogni caso.
Se $|\lambda_i| < 1, i > 1$ allora v converge a $v_1 = r$.

L'autovettore da $ 25.000.000.000

Considerando il teletrasporto, stiamo effettivamente calcolando:

$$A = \alpha Q + (1-\alpha)\frac{1}{n}ee^T$$

con (tipicamente) $\alpha = 0.85$ (che migliora la velocità di convergenza).
Inoltre, non ci interessa (e sarebbe impossibile) costruire esplicitamente ee^T; si può invece procedere con

$$y = Az = \alpha Qz + (1-\alpha)ee^T z = aQz + \beta e$$

con

$$\beta = (1-\alpha)e^T z$$

Infine, possiamo aggiungere un vettore di "teletrasporto" personalizzato (e pagato profumatamente)

$$Q + \frac{1}{n}ed^T$$

che si gestisce tranquillamente nello stesso modo

L'autovettore da $ 25.000.000.000

Da dove viene α?

L'autovettore da \$ 25.000.000.000

Da dove viene α?

Q è stocastica

Quindi $\lambda_1 = 1$, e se irriducibile allora $|\lambda_i| < 1$, $i > 1$, e quindi le componenti successive si smorzano facendo sopravvivere il primo autovettore v_1

Teorema

Se gli autovalori di Q sono $\{1, \lambda_2, \lambda_3, \ldots, \lambda_n\}$, allora gli autovalori di $A = \alpha Q + (1-\alpha)\frac{1}{n}ee^T$ sono

$$\{1, \alpha\lambda_2, \alpha\lambda_3, \ldots, \alpha\lambda_n\}$$

E quindi il coefficiente $\alpha = 0.85$ controlla il secondo autovalore di Q, che nel caso peggiore vale $\lambda_2 = 1$, e quindi governa la convergenza (con 60 iterazioni il contributo del secondo autovalore si riduce a 10^{-4}).

L'autovettore da \$ 25.000.000.000

P.S.: \$ 25.000.000.000 era il valore della offerta pubblica iniziale (IPO) delle azioni di Google