Appunti di

Calcolo Scientifico

dalle lezioni di Lidia Aceto

Francesco Baldino

Primo semestre a.a. 20/21

v0.17.0

Indice

Disclaimer

- Questi appunti DOVREBBERO essere completi, ma non ne sono sicuro al 100%. Se volete avvisarmi di errori o di parti mancanti, potete avvisarmi alla mia mail di dipartimento: baldino@mail.dm.unipi.it
- Questi appunti seguono approssimativamente l'ordine delle lezioni del corso. Alcuni argomenti sono stati anticipati o posticipati di poco per rendere più organizzata la presentazione degli argomenti.
- Continuo ripetutamente a dimenticarmi che in LaTeXdevo essere in math mode per poter scrivere i simboli $<$ e $>$. Se vi capita di vedere dei ¡ o ¿, sappiate che ¡ $=<$ e ¿ $=>$. Per favore segnalatemeli che li correggo.

Prerequisiti

Per seguire questi appunti è fortemente consigliato l'aver già studiato il corso di Analisi Numerica e Geometria 1. Aver studiato Analisi 1 non guasta.

Contenuto

In questi appunti vengono riportate solo le lezioni tenute da Lidia Aceto. Il corso comprende anche delle lezioni di Dario Andrea Bini, ma il contenuto delle sue lezioni è già riportato in pieno (e con ogni probabilità meglio di come lo riporterei io) nelle dispense presenti nell'elearning di Calcolo Scientifico.

1 23/09/2020

1.1 Autovalori e autovettori

Ricordiamo velocemente definizioni e qualche proprietà di autovalori e autovettori.

Definizione (autovettore e autovalore)**.** Data una matrice $A \in \mathbb{C}^{n\times n}$ quadrata a valori complessi diciamo che $\lambda \in \mathbb{C}$ è un autovalore per A se esiste un vettore $x \in \mathbb{C}^n$ con $x \neq 0$ tale che $(A - \lambda I)x = 0$, dove I è la matrice identità e 0 è da intendere come il vettore nullo. In tal caso, x si dice autovettore di A per λ.

Il sistema $(A-\lambda I)x = 0$ è un sistema lineare omogeneo, che quindi ammette soluzione non nulla se e solo se la matrice $(A - \lambda I)$ non è invertibile, ovvero se e solo se $\det(A - \lambda I) = 0$.

L'equazione $\det(A - \lambda I) = 0$ è detta equazione caratteristica di A. Il polinomio $p(\lambda) = \det(A - \lambda I)$ è detto polinomio caratteristico di A.

Poiché stiamo lavorando in $\mathbb{C}$ che è algebricamente chiuso, il polinomio $p(\lambda)$ ha tutte le sue radici nel campo (radici che per come li abbiamo definiti, sono gli autovalori di A), quindi A ammette autovalori.

Osserviamo che poiché gli autovettori sono soluzioni non nulle di un sistema omogeneo, se x è autovettore allora per ogni $\alpha \neq 0$, anche αx è soluzione.

Come mostrato nelle dispense su elearning delle lezioni di Bini, saper calcolare gli autovalori di una matrice torna molto utile in alcuni problemi. Ricordandoci che gli autovalori sono gli zeri del polinomio caratteristico, vediamo ora alcuni metodi per calcolare gli autovalori di una matrice:

1. **Metodo di iterazione funzionale**
   Come visto in Analisi Numerica, esistono vari metodi iterativi per calcolare gli zeri di una funzione, come il metodo di Newton. Questi metodi risultano efficienti solo se è "facile" calcolare i valori della funzione in un punto e della derivata della funzione nello stesso punto.

2. **Esplicitare** $p(\lambda)$
   Conoscendo i coefficienti di $p(\lambda)$ (che per ora conosciamo solo implicitamente il risultato di $\det(A - \lambda I)$) è possibile, per $n \leq 4$, calcolare esplicitamente le radici in maniera analitica([1])

Il secondo metodo ha però dei pesanti svantaggi:

- il costo del calcolo dei coefficienti è molto elevato

[1](Non trattato in classe): Il motivo per cui ciò si possa fare solo per $n \leq 4$ è che non esiste una formula risolutiva generale per i polinomi di grado maggiore o uguale a 5. Vedasi il teorema di Abel-Ruffini.

- gli errori sui coefficienti di $p(\lambda)$ si ripercuotono pesantemente sul calcolo delle radici

Mostriamo quindi alcuni risultati ottenibili scegliendo di utilizzare il primo metodo.

1.2 Analisi del condizionamento

Cominciamo esibendo un teorema che fornisce un sottoinsieme di $\mathbb{C}$ che contiene sicuramente tutti gli autovalori di A, rendendo quindi possibile restringere la ricerca.

Teorema 1.1 (di Hirsch)**.** *Data una matrice $A \in \mathbb{C}^{n\times n}$ e $\|\cdot\|$ una norma matriciale indotta, l'insieme $\{z \in \mathbb{C} : |z| \leq \|A\|\}$ contiene tutti gli autovalori di A.*

Dimostrazione. Per definizione, λ è autovalore e $x \neq 0$ è autovettore se $(A - \lambda I)x = 0$, cioè se $Ax = \lambda x$.
Passando alle norme otteniamo $\|Ax\| = \|\lambda x\| = |\lambda|\|x\|$.
Poiché $x \neq 0$, anche $\|x\| \neq 0$ quindi posso dividere l'equazione appena ottenuta per $\|x\|$. Ottengo

$$|\lambda| = \frac{\|Ax\|}{\|x\|} \leq \sup_{x\neq 0} \frac{\|Ax\|}{\|x\|} = \|A\|$$

dove l'ultima uguaglianza è la definizione di norma matriciale indotta.
Allora ogni autovalore di A ha modulo minore o uguale alla norma di A, e sta quindi nell'insieme definito dal teorema □

Ora che abbiamo ristretto la ricerca, cerchiamo di capire come una perturbazione dei coefficienti di A si ripercuota sui suoi autovalori. A questo scopo ci viene in aiuto il seguente teorema.

Teorema 1.2 (di Bauer-Fike)**.** *Sia $\|\cdot\|$ una norma matriciale indotta assoluta, ovvero una norma matriciale indotta che per ogni $D \in \mathbb{C}^{n\times n}$ diagonale verifichi:*

$$\|D\| = \max_{i=1,\dots n} |d_{ii}|$$

(ad esempio una qualsiasi $\|\cdot\|_p$).
Sia $A \in \mathbb{C}^{n\times n}$ diagonalizzabile con $A = TDT^{-1}$, dove $T, D \in \mathbb{C}^{n\times n}$ con D diagonale e T invertibile.
Sia $\delta A \in \mathbb{C}^{n\times n}$ e ξ un autovalore di $(A + \delta A)$.
Allora $\exists \lambda$ autovalore di A tale che $|\lambda - \xi| \leq \mu(T)\|A\|$, dove ricordiamo $\mu(T) = \|T\|\|T^{-1}\|$ essere il numero di condizionamento di T.

Dimostrazione. Sia $(A + \delta A)y = \xi y$, con $y \neq 0$. Posso riscrivere l'uguaglianza come $(A - \xi I)y = -\delta A y$.
Analiziamo la matrice $(A - \xi I)$. I casi possibili sono due:

- $A - \xi I$ è singolare, cioè $\det(A - \xi I) = 0$. Allora ξ è autovalore di A e la tesi è banalmente verificata con $\lambda = \xi$.
- $A - \xi I$ è non singolare.
  Allora possiamo scrivere $y = -(A - \xi I)^{-1}\delta A y$ e passando alle norme otteniamo

$$\|y\| = \left\|-(A - \xi I)^{-1}\delta A y\right\| \leq \left\|-(A - \xi I)^{-1}\delta A\right\| \|y\|$$

  e dividendo per $\|y\|$ otteniamo

$$\begin{aligned}
1 \leq & \left\|(A - \xi I)^{-1}\delta A\right\| = \\
= & \left\|(TDT^{-1} - \xi I)^{-1}\delta A\right\| = \\
= & \left\|(TDT^{-1} - \xi TIT^{-1})^{-1}\delta A\right\| = \\
= & \left\|(T(D - \xi I)T^{-1})^{-1}\delta A\right\| = \\
= & \left\|T(D - \xi I)^{-1}T^{-1}\delta A\right\| \leq \\
\leq & \|T\| \left\|(D - \xi I)^{-1}\right\| \left\|T^{-1}\right\| \|\delta A\| = \\
= & \leq \underbrace{\|T\| \left\|T^{-1}\right\|}_{=\mu(T)} \left\|(D - \xi I)^{-1}\right\| \|\delta A\|
\end{aligned}$$

  e poiché vale

$$\left\|(D - \xi I)^{-1}\right\| = \frac{1}{\min\limits_{i=1,\ldots,n} |\lambda_i - \xi|}$$

  otteniamo che esiste almeno un λ_i per cui vale la disuguaglianza, e quindi la tesi

□

Quindi più piccolo è $\mu(T)$, più sarà ben condizionato il problema del calcolo degli autovalori. Ricordiamo però che poiché vale

$$1 = \|I\| = \left\|TT^{-1}\right\| \leq \|T\| \left\|T^{-1}\right\| = \mu(T)$$

il valore minimo di $\mu(T)$ è comunque 1.

Vediamo ora alcuni casi pratici e di come si comporta il condizionamento.

Definizione (matrice normale)**.** Diciamo che $A \in \mathbb{C}^{n \times n}$ è normale se $AA^H = A^H A$, dove $A^H = \bar{A}^T$

Vale che se A è normale, allora è anche diagonalizzabile tramite una matrice T unitaria. Poiché per le matrici unitarie vale, per la norma 2, $\mu(T) = 1$, il problema del calcolo degli autovalori per matrici normali è ben condizionato.

Consideriamo uinvece A non normale, e concentriamoci su un singolo autovalore di molteplicità algebrica uguale a 1.

Teorema 1.3. *Sia $A \in \mathbb{C}^{n\times n}$ non normale e λ un autovalore di molteplicità algebrica uguale a 1.*
Siano $x, y \in \mathbb{C}^n$ tali che $\|x\|_2 = \|y\|_2 = 1$ rispettivamente autovettori destri e sinistri di A per λ, ovvero tali che

$$Ax = \lambda x, \qquad y^H A = \lambda y^H$$

Allora per ogni $F \in \mathbb{C}^{n\times n}$ e per ogni $\varepsilon > 0$, definendo con $\lambda(\varepsilon)$ la funzione che associa ad ε l'autovalore di $A + \varepsilon F$ "corrispondente" a λ, vale che

$$\lambda(\varepsilon) - \lambda = \varepsilon \frac{y^H F x}{y^H x}$$

Dimostrazione. Sia $x(\varepsilon)$ la funzione che associa ad ε l'autovettore destro per $\lambda(\varepsilon)$ "corrispondente" a x. Vale

$$(A + \varepsilon F)x(\varepsilon) = \lambda(\varepsilon)x(\varepsilon) \tag{1}$$

Supponendo che $\lambda(\varepsilon)$ e $x(\varepsilon)$ siano funzioni sufficientemente regolari e supponendo ε vicino a 0 (che sono poi le condizioni del caso che davvero ci interessa), posso sviluppare con Taylor

- $\lambda(\varepsilon) = \lambda(0) + \varepsilon\lambda'(0) + O(\varepsilon^2)$
- $x(\varepsilon) = x(0) + \varepsilon x'(0) + O(\varepsilon^2)$

con $\lambda(0) = \lambda$, $x(0) = x$.
Facendo un'analisi al primo ordine di (1) e sostituendo i valori di $\lambda(\varepsilon)$ e $x(\varepsilon)$, otteniamo

$$\begin{aligned}
(A + \varepsilon F)(x + \varepsilon x'(0)) &\doteq (\lambda + \varepsilon\lambda'(0))(x + \varepsilon x'(0)) \\
Ax + \varepsilon Ax'(0) + \varepsilon Fx + \varepsilon^2 Fx'(0) &\doteq \lambda x + \varepsilon\lambda x'(0) + \varepsilon\lambda'(0)x + \varepsilon^2\lambda'(0)x'(0) \\
Ax + \varepsilon Ax'(0) + \varepsilon Fx &\doteq \lambda x + \varepsilon\lambda x'(0) + \varepsilon\lambda'(0)x \\
Ax + \varepsilon Ax'(0) + \varepsilon Fx &\doteq Ax + \varepsilon\lambda x'(0) + \varepsilon\lambda'(0)x \\
\varepsilon Ax'(0) + \varepsilon Fx &\doteq \varepsilon\lambda x'(0) + \varepsilon\lambda'(0)x \\
Ax'(0) + Fx &\doteq \lambda x'(0) + \lambda'(0)x \\
y^H Ax'(0) + y^H Fx &\doteq \lambda y^H x'(0) + \lambda'(0)y^H x \\
y^H Ax'(0) + y^H Fx &\doteq y^H Ax'(0) + \lambda'(0)y^H x \\
y^H Fx &\doteq \lambda'(0)y^H x \\
\varepsilon y^H Fx &\doteq \varepsilon\lambda'(0)y^H x
\end{aligned}$$

e poiché vale $\varepsilon\lambda'(0) = \lambda(\varepsilon) - \lambda$ otteniamo la tesi $\square$

Utilizzando questo risultato, possiamo stimare quanto varia $\lambda(\varepsilon)$. Vale

$$\begin{aligned}\left|\lambda(\varepsilon) - \lambda\right| &= \frac{\left|y^H(\varepsilon F)x\right|}{\left|y^H x\right|} \leq \\ &\leq \frac{\left\|y^H\right\|\|\varepsilon F\|\|x\|}{\left|y^H x\right|} = \\ &= \frac{1}{\left|y^H x\right|}\|\varepsilon F\|\end{aligned}$$

Osserviamo che questi conti possono essere ripetuti con A normale. Per A normale, gli autovalori destri e sinistri coincidono, quindi a meno di normalizzarli abbiamo $y^H x = 1$, e riotteniamo $\left|\lambda(\varepsilon) - \lambda\right| \leq \|\varepsilon F\|$ che è concorde al risultato del teorema di Bauer-Fike.

2 25/09/2020

2.1 Condizionamento in molteplicità maggiore di 1

Abbiamo visto nella lezione precedente come si comportano gli autovalori nel caso di molteplicità algebrica uguale a 1. Nel caso di molteplicità algebrica maggiore di 1, il problema di calcolare gli autovalori diventa molto complicato. Vediamo un esempio

Esempio. Consideriamo la seguente matrice, avente come unico autovalore 0:

$$A = \begin{pmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ & & & 0 \end{pmatrix}$$

Perturbiamo ora A con un $\varepsilon > 0$ sommando questo ε all'entrata in basso a sinistra di A, ottenendo

$$A + \varepsilon F = \begin{pmatrix} 0 & 1 & & \\ & \ddots & \ddots & \\ & & \ddots & 1 \\ \varepsilon & & & 0 \end{pmatrix}$$

Il polinomio caratteristico di questa nuova matrice è $p(t) = t^n - \varepsilon$, che ha n come radici le radici n-esime di ε, e possiamo considerare $\lambda(\varepsilon) = \sqrt[n]{\varepsilon}$.
Considerando quindi il caso $n = 16, \quad \varepsilon = 10^{-16}$, otteniamo che l'autovalore 0 finisce nell'autovalore $\lambda(\varepsilon) = \sqrt[16]{10^{-16}} = 10^{-1}$. Quindi anche una perturbazione molto piccola, quale 10^{-16} può provocare una perturbazione grossa, nel nostro caso 10^{-1}.

Quindi nel caso di molteplicità algebrica maggiore di 1, il problema può essere molto mal condizionato.

2.2 Metodi di determiazione di autovalori

Vediamo velocemente due tecniche utilizzabili per determinare gli autovalori di una matrice:

1. **Metodo a due fasi**
   Fase 1: si trasforma la matrice A in una matrice B simile ad A (in modo da conservare gli autovalori) per la quale siano noti metodi semplici per calcolare gli autovalori.
   Fase 2: si calcolano gli autovalori di B

2. **Completamenti Iterativi**
   Ripeto passi iterativi dove ogni volta risolvo un sistema lineare. Si usa di solito per matrici di taglia grande.

Concentriamoci per ora sul metodo a 2 fasi, e vediamo come poter ricondurre una matrice in forma di Hessemberg superiore.

Definizione (forma di Hessemberg superiore)**.** Una matrice $B \in \mathbb{C}^{n\times n}$ si dice in forma di Hessemberg superiore se vale che $i > j+1 \Rightarrow b_{ij} = 0$, ovvero una matrice di questa forma

$$B = \begin{pmatrix} b_{1,1} & b_{1,2} & \dots & \dots & b_{1,n} \\ b_{2,1} & \ddots & \ddots & & \vdots \\ 0 & \ddots & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & b_{n-1,n} \\ 0 & \dots & 0 & b_{n,n-1} & b_{n,n} \end{pmatrix}$$

Consideriamo il caso di A diagonalizzabile, con $A = SDS^{-1}$ e $B = T^{-1}AT$. Osserviamo che D contiene, sulla diagonale, gli autovalori di A, e S ha per colonne autovettori di A (si dimostra velocemente leggendo riga per riga l'equazione $AS = SD$). Scrivendo

$$B = T^{-1}AT = T^{-1}(SDS^{-1})T = (T^{-1}S)D(S^{-1}T) = (T^{-1}S)D(T^{-1}S)^{-1}$$

si ottiene che anche B è diagonalizzabile, ha gli stessi autovalori di A e la matrice $T^{-1}S$ ha per colonne autovettori di B.
Poiché B è diagonalizzabile possiamo applicare il teorema di Bauer-Fike, quindi il condizionamento degli autovalori dipende da $\mu(T^{-1}S)$. Osserviamo però che vale

$$\begin{aligned} \mu(T^{-1}S) &= \left\|T^{-1}S\right\|\left\|(T^{-1}S)^{-1}\right\| = \\ &= \left\|T^{-1}S\right\|\left\|S^{-1}T\right\| \leq \\ &\leq \|T\|\left\|S^{-1}\right\|\left\|T^{-1}\right\|\|S\| = \\ &= \mu(T)\mu(S) \end{aligned}$$

Quindi nel trasformare A in B dobbiamo cercare di scegliere una matrice T che renda minimo il numero di condizionamento di T, idealmente 1.
Cerchiamo ora un algoritmo $A \rightsquigarrow B$ con un procedimento iterativo con

$$A_{k+1} = T_k^{-1}A_kT_k$$

con $A_1 = A$, $A_m = B$ e varrà $T = T_1 \dots T_{m-1}$.
Esistono vari modi di scegliere tali matrici T_k, quali matrici di Gauss, di Householder o di Givens.

Scegliendo matrici di Householder o di Givens, poiché sono tutte matrici unitarie, otterremmo che il numero di condizionamento in norma 2 delle varie matrici T_k varrebbe $\mu_2(T_k) = \|T_k\|_2 \left\|T_k^{-1}\right\|_2 = 1$. Varrebbe quindi per T

$$\mu(T) = \mu(T_1 \dots T_{m-1}) \leq \mu(T_1) \dots \mu(T_{m-1}) = 1$$

quindi anche $\mu(T) = 1$ quindi otterremmo condizionamento ottimo.
Scegliendo invece matrici di Gauss otterremmo $\mu_\infty(T_k) = \|T_k\|_\infty \left\|T_k^{-1}\right\|_\infty \leq 4$, quindi $\mu_\infty(T) \leq 4^m$ che è accettabile solo per m piccolo.

La trasformazione $A \rightsquigarrow B$ ha il vantaggio di ridurre molto il costo del calcolo degli autovalori. Infatti, applicando ad esempio il metodo QR per il calcolo di autovalori e autovettori (che vedremo in seguito) su una generica matrice A il costo è $O(n^3)$, mentre su una matrice B del genere è di $O(n^2)$.

2.3 Metodo di Householder per A hermitiana

Definizione (matrice di Householder)**.** Si dice matrice di Householder una matrice H della forma $H = I - \sigma w w^H$, con $w \in \mathbb{C}^n$ e $\sigma = \frac{2}{\|w\|_2^2}$. Le matrici di Householder sono sempre elementari, unitarie ed hermitiane.

Il metodo di Householder prevede un algoritmo iterativo come definito prima scegliendo T_k di Householder.
Cominciamo considerando il caso di A hermitiana.
Cerchiamo una T_k di Householder tale il prodotto $T_k A_k$ annulli tutti gli elementi della $k-$esima colonna aventi indice di riga $i > k+1$. Descriviamo come costruire la matrice T_k per $k = 1$ per spiegare grossolanamente l'idea di base per poi esibire il generico passo iterativo $k-$esimo. Cominciamo con

$$A_1 = A = \left(\begin{array}{c|c} a_{11}^{(1)} & a_1^H \\ \hline a_1 & B^{(1)} \\ \vdots & \end{array}\right)$$

A è in questa forma perché per ora la stiamo supponendo hermitiana. L'apice $^{(1)}$ è utilizzato per ricordare che quel coefficiente o matrice apparteneva alla matrice della prima iterazione. Per costruire T_1 vogliamo prima trovare una matrice $P_1 \in \mathbb{C}^{(n-1)\times(n-1)}$ di Householder tale che $P_1 a_1 = \alpha_1 e_1$ per un qualche α_1, dove con e_1 intendiamo il vettore con tutti zeri tranne un uno in posizione 1.
P_1 deve essere della forma $P = I - \beta_1 v_1 v_1^H$. Pongo $v_1 = a_1 + \operatorname{sgn}(a_{21}^{(1)}) \|a_1\|_2 e_1$, e β_1 di conseguenza.
P_1 scelta in questo modo ha la proprietà che ci interessa, e svolgendo il prodotto otteniamo $\alpha_1 = -\operatorname{sgn}(a_{21}^{(1)}) \|a_1\|_2$.

Scegliamo allora T_1 come

$$T_1 = \left(\begin{array}{c|c} 1 & 0 \\ \hline 0 & P_1 \end{array}\right)$$

che osserviamo essere sempre di Householder, con $T = I - \beta_1 \begin{pmatrix} 0 \\ v_1 \end{pmatrix} (0|v_1^H)$. Svolgendo ora il prodotto $A_2 = T_1 A_1 T_1^{-1}$ otteniamo

$$A_2 = T_1 A_1 T_1^{-1} = \left(\begin{array}{c|c|c} a_{11}^{(1)} & \alpha_1^H & 0 \\ \hline \alpha_1 & a_{22}^{(2)} & a_2^H \\ \hline 0 & a_2 & B^{(2)} \end{array}\right)$$

Che è esattamente nella forma che volevamo per concludere il primo passo iterativo. Vediamo ora il generico passo k-esimo. Partiamo da una matrice A_k della seguente forma:

$$A_k = \left(\begin{array}{c|c|c} C_k & b_k & 0 \\ \hline b_k^H & a_{kk}^{(k)} & a_k^H \\ \hline 0 & a_k & B^{(k)} \end{array}\right)$$

dove C_k è una matrice tridiagonale di taglia $k-1$ e b_k ha le prime $k-2$ componenti nulle (ovvero ha al più l'ultima componente non nulla).
Come prima, scegliamo P_k tale che $P_k a_k = \alpha_k e_k \in \mathbb{C}^{n-k}$ e scegliamo T_k come

$$T_k = \left(\begin{array}{c|c} I_{n-k} & 0 \\ \hline 0 & P_k \end{array}\right)$$

Come prima, svolgendo il prodotto $A_{k+1} = T_k A_k T_k^{-1}$ otterremo

$$A_{k+1} = T_k A_k T_k^{-1} = \left(\begin{array}{c|c|c} C_k & b_k & 0 \\ \hline b_k^H & a_{kk}^{(k)} & \alpha_k e_k^H \\ \hline 0 & \alpha_k e_k & P_k B^{(k)} P_k^{-1} \end{array}\right)$$

che è nelle ipotesi di partenza del passo iterativo successivo.

2.4 Analisi dei costi

Osserviamo che, volendo svolgere questo algoritmo, non è strettamente necessario calcolare P_k e il prodotto $T_k A_k T_k^{-1}$, purché si disponga di un metodo per ottenere α_k e $B^{(k+1)}$ (poiché questi e A_k bastano per ricostruire A_{k+1}).
Per quanto riguarda α_k, abbiamo visto che vale $\alpha_k = -\text{sgn}(a^{(k)}_{k+1,k})\|a_k\|_2$.
Per $B^{(k+1)}$, invece, dobbiamo trovare un modo di calcolarla senza passare per P_k.
Cominciamo scrivendo la relazione $B^{(k+1)} = P_k B^{(k)} P_k^{-1}$, e poiché P_k è hermitiana unitaria, vale $P_k = P_k^{-1}$, quindi $B^{(k+1)} = P_k B^{(k)} P_k$.
Sviluppiamo:

$$\begin{aligned}
B^{(k+1)} &= P_k B^{(k)} P_k = \\
&= (I - \beta_k v_k v_k^H) B^{(k)} (I - \beta_k v_k v_k^H) = \\
&= (B^{(k)} - \beta_k v_k v_k^H B^{(k)})(I - \beta_k v_k v_k^H) = \\
&= B^{(k)} - \beta_k B^{(k)} v_k v_k^H - \beta_k v_k v_k^H B^{(k)} + \beta_k^2 v_k (v_k^H B^{(k)} v_k) v_k^H = \\
&= B^{(k)} - \beta_k B^{(k)} v_k v_k^H - \beta_k v_k v_k^H B^{(k)} + \beta_k v_k (v_k^H \beta_k B^{(k)} v_k) v_k^H = \\
&= B^{(k)} - \beta_k B^{(k)} v_k v_k^H - \beta_k v_k v_k^H B^{(k)} + \beta_k (v_k^H \beta_k B^{(k)} v_k) v_k v_k^H = \\
&= B^{(k)} - \beta_k B^{(k)} v_k v_k^H - \beta_k v_k v_k^H B^{(k)} + \\
&\quad + \frac{1}{2}\beta_k (v_k^H \beta_k B^{(k)} v_k) v_k v_k^H + \frac{1}{2}\beta_k (v_k^H \beta_k B^{(k)} v_k) v_k v_k^H = \\
&= B^{(k)} - (\beta_k B^{(k)} v_k - \frac{1}{2}\beta_k (v_k^H \beta_k B^{(k)} v_k) v_k) v_k^H - \\
&\quad - v_k (\beta_k v_k^H B^{(k)} - \frac{1}{2}\beta_k (v_k^H \beta_k B^{(k)} v_k) v_k^H)
\end{aligned}$$

Definendo ora $r_k = \beta_k B^{(k)} v_k$ e $\varphi_k = r_k - \frac{1}{2}(\beta_k (r_k^H v_k) v_k)$ possiamo esprimere $B^{(k+1)}$ come

$$B^{(k+1)} = P_k B^{(k)} P_k = B^{(k)} - \varphi_k v_k^H - v_k \varphi_k^H = B^{(k)} - \varphi_k v_k^H - (\varphi_k v_k^H)^H$$

Quindi per conoscere $B^{(k+1)}$ mi basta in realtà conoscere $B^{(k)}$ e $\varphi_k v_k^H$. Per poterli calcolare, dobbiamo prima conoscere r_k.
Per calcolare $B^{(k)} v_k$ servono $(n-k)^2$ operazioni (tra le quali contiamo solo le moltiplicazioni, e tralasciamo le altre operazioni meno costose), e anche per calcolare $\varphi_k v_k^H$ servono $(n-k)^2$ operazioni.
Quindi per passare da A^k a A_{k+1} (noti $B^{(k)}$, v_k e r_k) sono necessarie $2(n-k)^2$ operazioni. Poiché per l'algoritmo $A \rightsquigarrow B$ devo compiere il passo iterativo $n-2$ volte, il costo totale dell'algoritmo è di $\sum_{k=1}^{n-2} 2(n-k)^2 \sim \frac{2}{3}n^3$ operazioni.

2.5 Metodo di Householder per A generica

Il caso di A generica è analogo al caso di A hermitiana, ma poiché (analizzando ad esempio il passo $k = 1$) A è della forma

$$A_1 = A = \left(\begin{array}{c|c} a_{11}^{(1)} & \tilde{a}_1^H \\ \hline a_1 & B^{(1)} \\ & \end{array}\right)$$

con $\tilde{a}_1^H$ potenzialmente diverso da a_1^H (in quanto A non hermitiana), non ho garanzie di riuscire ad annullare sia a_1 che $\tilde{a}_1^H$ quando vado a moltiplicare per T_1 e T_1^{-1}. Invece di ottenere una matrice B tridiagonale, sarà possibile ottenere una matrice B in forma di Hessenberg superiore. Lo svolgimento del passo k-esimo, fatta eccezione per la forma finale della matrice, è identico al passo k-esimo nel caso di A hermitiana.

2.6 Analisi dei costi

Come prima, per ridurre il costo dell'algoritmo al minimo, basta trovare un modo furbo per calcolare A_{k+1} senza passare per il prodotto tra matrici. Posso passare da A_k a A_{k+1} come prima semplicemente conoscendo α_k, $B^{(k+1)}$ e questa volta anche $\tilde{a}_1^H P_k$. Ponendo r_k come prima e $s_k = \beta_k v_k^H B^{(k)}$ otteniamo

$$B^{(k+1)} = P_k B^{(k)} P_k = B^{(k)} - r_k v_k^H + \beta_k (s_k^H v_k) v_k v_k^H - v_k v_k^H$$

Per passare al passo successivo devo qundi calcolare r_k, s_k, $r_k v_k^H$, $s_k^H v_k$ e $\tilde{a}_1^H P_k$, per un totale di $6(n-k)^2$ operazioni ad ogni passo iterativo.
Il costo totale è quindi di $\sum_{k=1}^{n-2} 6(n-k)^2 \sim 2n^3$ operazioni.

3 29/09/2020

3.1 Calcolare gli autovalori di matrici tridiagonali

Abbiamo visto che partendo da una matrice A hermitiana è possibile arrivare facilmente, attraverso similitudine, ad una matrice B tridiagonale. Poiché la similitudine preserva la proprietà di essere hermitiana, la matrice B sarà di questa forma:

$$B = \begin{pmatrix} \alpha_1 & \bar{\beta_2} & & \\ \beta_2 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \bar{\beta_n} \\ & & \beta_n & \alpha_n \end{pmatrix}$$

con $\alpha_i \in \mathbb{R}$ e $\beta_i \in \mathbb{C}$. Chiamiamo ora $B_n = B$ e B_i il minore di testa di B di taglia $i \times i$. Sfrutteremo ora questa scomposizione di B per cercare di calcolarne gli autovalori.

Osservazione. Se anche uno solo dei β_i fosse uguale a 0, avremmo che la matrice B sarebbe riducibile e quindi il problema di determinare i suoi autovalori potrebbe essere semplificato al problema di determinare gli autovalori di due sottomatrici più piccole. Volendo analizzare il "caso base", supponiamo che tutti i β_i siano non nulli.

Sia $p_i(\lambda) = \det(B_i - \lambda I)$ il polinomio caratteristico di B_i. Trovare gli autovalori di B equivale a trovare gli zeri di $p_n(\lambda)$, e per farlo possiamo utilizzare metodi come il metodo di Newton. Vediamo ora come poter calcolare in modo efficiente $p_n(\lambda)$ e $p'_n(\lambda)$.

Cominciamo esprimendo una relazione tra $p_i(\lambda)$ e $p_n(\lambda)$. Vediamo in particolare il caso $n = 3$, che come conto è uguale al caso n generico ma è un po' più semplice di notazione. Sia

$$B_3 = \begin{pmatrix} \alpha_1 & \bar{\beta_2} & 0 \\ \beta_2 & \alpha_2 & \bar{\beta_3} \\ 0 & \beta_3 & \alpha_3 \end{pmatrix}$$

Allora

$$\begin{aligned} p_3(\lambda) = \det(B_3 - \lambda I) &= \det \begin{pmatrix} (\alpha_1 - \lambda) & \bar{\beta_2} & 0 \\ \beta_2 & (\alpha_2 - \lambda) & \bar{\beta_3} \\ 0 & \beta_3 & (\alpha_3 - \lambda) \end{pmatrix} = \\ &= (\alpha_3 - \lambda) \det \begin{pmatrix} (\alpha_1 - \lambda) & \bar{\beta_2} \\ \beta_2 & (\alpha_2 - \lambda) \end{pmatrix} - \beta_3 \det \begin{pmatrix} (\alpha_1 - \lambda) & 0 \\ \beta_2 & \bar{\beta_3} \end{pmatrix} = \\ &= (\alpha_3 - \lambda) \det(B_2 - \lambda I) - \beta_3 \bar{\beta_3} (\alpha_1 - \lambda) = \\ &= (\alpha_3 - \lambda) \det(B_2 - \lambda I) - |\beta_3|^2 \det(B_1 - \lambda) = \\ &= (\alpha_3 - \lambda) p_2(\lambda) - |\beta_3|^2 p_1(\lambda) \end{aligned}$$

Lo stesso identico conto (ovvero sviluppando con Laplace sull'ultima riga) può essere fatto per qualsiasi n generico e in realtà anche per qualsiasi polinomio p_i. Otteniamo quindi la seguente relazione di ricorrenza:

- $p_0(\lambda) = 1$
- $p_1(\lambda) = \alpha_1 - \lambda$
- $p_i(\lambda) = (\alpha_i - \lambda)p_{i-1}(\lambda) - |\beta_i|^2 p_{i-2}(\lambda)$

Osservazione. Sicuramente vale che $\alpha_i, |\beta_i|^2 \in R$. Inoltre, sappiamo che B_i ha solo autovalori reali (in quanto hermitiana). Ha senso quindi limitarci a cercare autovalori tra $\lambda \in \mathbb{R}$. In quanto polinomio a coefficienti reali calcolato in un valore reale, vale anche che $p_i(\lambda) \in \mathbb{R}$. Ci tornerà utile poi.

Conoscendo α_i, β_i, λ, $p_{i-1}(\lambda)$ e $p_{i-2}(\lambda)$ il costo per calcolare $p_i(\lambda)$ è di 5 operazioni (o 7 nel caso in cui β_i non sia reale, e che quindi $|\beta_i|^2$ non sia una singola moltiplicazione).
Poiché per calcolare $p_n(\lambda)$ devo calcolare tutti i polinomi precedenti, il costo per calcolare $p_n(\lambda)$ conoscendo solo la matrice B e λ è $\sim 5n$ (o $7n$ se i β_i non sono reali). Possiamo dire che calcolare è $p_n(\lambda)$ è un $O(n)$.

Calcoliamo ora $p'_n(\lambda)$. Sfruttando la relazione di ricorrenza trovata prima, deriviamo tutto rispetto a λ per trovare una relazione tra le derivate. Otteniamo:

- $p'_0(\lambda) = 0$
- $p'_1(\lambda) = -1$
- $p'_1(\lambda) = (\alpha_i - \lambda)p'_{i-1}(\lambda) - p_{i-1}(\lambda) - |\beta_i|^2 p'_{i-2}(\lambda)$

Possiamo supporre di conoscere già $(\alpha_i - \lambda), |\beta_i|^2$ e $p_i(\lambda)$ perché in una implementazione del metodo basterebbe salvarsi i valori utilizzati per calcolare $p_n(\lambda)$. Per calcolare $p'_i(\lambda)$ servono quindi 4 operazioni, e poiché per calcolare $p'_n(\lambda)$ devo calcolare tutti i polinomi precedenti, il costo totale è di $\sim 4n = O(n)$ operazioni.

In conclusione, per calcolare $\frac{p_n(\lambda)}{p'_n(\lambda)}$ il costo è di $\sim(7+4)n + 1 = O(n)$ operazioni.

3.2 Intervalli di autovalori di matrici tridiagonali

Abbiamo visto che calcolare $\frac{p_n(\lambda)}{p'_n(\lambda)}$ per poter utilizzare uno dei metodi che conosciamo per approssimare gli zeri di $p_n(\lambda)$ è un $O(n)$. Cerchiamo ora di determinare in che intervalli si trovano gli autovalori di B_n per sapere, approssimativamente, da dove partire nell'utilizzare uno dei metodi noti. Per semplicità ci ristringeremo al caso $B_n \in \mathbb{R}^{n\times n}$.

Per fare ciò, esprimiamo una relazione tra gli autovalori di B_n e B_{n-1} che tornerà utile. Cominciamo con un'osservazione.

Osservazione. B_n e B_{n-1} non possono avere autovalori in comune.

Dimostrazione. Sia ξ un autovalore di B_n. Vale quindi che $p_n(\xi) = 0$. Supponiamo per assurdo che valga anche $p_{n-1}(\xi) = 0$, ovvero che ξ sia anche un autovalore di B_{n-1}. Allora, per come è definita la relazione di ricorrenza, dovrebbe valere anche $p_{n-2}(\xi) = 0$.
Poiché $p_{n-1}(\xi) = p_{n-2}(\xi) = 0$, sempre per la relazione di ricorrenza dovrebbe valere $p_{n-3}(\xi) = 0$.
Iterando, si ottiene che tutti i polinomi si annullano in ξ. Vale però che $p_0(\xi) = 1 \neq 0$, e abbiamo un assurdo. □

Raffiniamo la relazione tra gli autovalori di B_n e quelli di B_{n-1}. Poiché $B_n \in \mathbb{R}^{n\times n}$, B_n è di questa forma:

$$B_n = \begin{pmatrix} \alpha_1 & \beta_2 & & \\ \beta_2 & \alpha_2 & \ddots & \\ & \ddots & \ddots & \beta_n \\ & & \beta_n & \alpha_n \end{pmatrix} = \left(\begin{array}{c|c} B_{n-1} & v_n \\ \hline v_n^T & \alpha_n \end{array}\right), \quad v_n := \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \beta_n \end{pmatrix}$$

Anche B_{n-1} è simmetrica, quindi per teorema spettrale reale è ortogonalmente diagonalizzabile. Sia Q_{n-1} una matrice ortogonale tale che:

$$B_{n-1} = Q_{n-1}^T D_{n-1} Q_{n-1}$$

con D_{n-1} diagonale. Definiamo ora $\hat{Q}$ come

$$\hat{Q} = \left(\begin{array}{c|c} Q_{n-1} & 0 \\ \hline 0 & 1 \end{array}\right)$$

che è comunque ortogonale. Chiamiamo $F_n = \hat{Q} B_n \hat{Q}^T$ e sviluppiamo il prodotto.

$$\begin{aligned} F_n = \hat{Q} B_n \hat{Q}^T = \\ = \left(\begin{array}{c|c} Q_{n-1} B_{n-1} Q_{n-1}^T & Q_{n-1} v \\ \hline v^T Q_{n-1}^T & \alpha_n \end{array}\right) = \\ = \left(\begin{array}{c|c} D_{n-1} & w \\ \hline w^T & \alpha_n \end{array}\right), \quad w := Q_{n-1} \begin{pmatrix} 0 \\ \vdots \\ 0 \\ \beta_n \end{pmatrix} \end{aligned}$$

Nota. F_n ha il minore di testa di taglia $(n-1)\times(n-1)$ diagonale, e non ci sono restrizioni sulle entrate rimanenti. Una matrice di questa forma si chiama matrice a freccia.

Poiché F_n è stata ottenuta per similitudine, $\det(F_n - \lambda I) = \det(B_n - \lambda I) = p_n(\lambda)$.
Enunciamo ora un teorema che ci servirà per continuare l'analisi degli autovalori di B_n.

Proposizione 3.1 (Complemento di Schur)**.** *Sia M una matrice a blocchi della forma*

$$M = \begin{pmatrix} A & B \\ C & D \end{pmatrix}$$

con A e D quadrate, A non singolare. Allora, ponendo $\Gamma := D - CA^{-1}B$ vale la scomposizione

$$M = \begin{pmatrix} I & 0 \\ CA^{-1} & I \end{pmatrix} \begin{pmatrix} A & 0 \\ 0 & \Gamma \end{pmatrix} \begin{pmatrix} I & A^{-1}B \\ 0 & I \end{pmatrix}$$

e Γ è detto complemento di Schur di A nella matrice M. In particolare vale

$$\det(M) = \det(A)\det(\Gamma) = \det(A)\det(D - CA^{-1}B)$$

Dimostrazione. La dimostrazione consiste semplicemente nello svolgere la moltiplicazione. Per il determinante, il risultato deriva da Binet □

Consideriamo ora la matrice $F_n - \lambda I$, ovvero

$$F_n - \lambda I = \left(\begin{array}{c|c} D_{n-1} - \lambda I & w \\ \hline w^T & (\alpha_n - \lambda) \end{array}\right) =$$

$$= \left(\begin{array}{ccc|c} (\lambda_1^{(n-1)} - \lambda) & & & \\ & \ddots & & w \\ & & (\lambda_{n-1}^{(n-1)} - \lambda) & \\ \hline & w^T & & (\alpha_n - \lambda) \end{array}\right) =$$

$$=: \left(\begin{array}{c|c} E_{n-1} & w \\ \hline w^T & (\alpha_n - \lambda) \end{array}\right)$$

Dove $\lambda_1^{(n-1)}, \ldots, \lambda_{n-1}^{(n-1)}$ sono gli autovalori di B_{n-1}.
Poiché sappiamo che B_n e B_{n-1} non hanno autovalori in comune, ha senso restringere la ricerca ai λ tali che per ogni i, $\lambda \neq \lambda_i^{(n-1)}$.

In queste ipotesi, E_{n-1} è sempre invertibile e siamo quindi sempre nelle ipotesi per poter applicare il complemento di Schur. Possiamo allora dire che

$$\begin{aligned} p_n(\lambda) &= \det(F_n - \lambda I) = \\ &= \det(E_{n-1}) \det((\alpha_n - \lambda) - w^T E_{n-1}^{-1} w) = \\ &= \left(\prod_{j=1}^{n-1} (\lambda_j^{(n-1)} - \lambda) \right) \left((\alpha_n - \lambda) - \sum_{i=1}^{n-1} \frac{w_i^2}{\lambda_i^{(n-1)} - \lambda} \right) \end{aligned}$$

Sia ora ξ un autovalore di B_n, ovvero tale che $p_n(\xi) = 0$. Poiché sappiamo che necessariamente deve valere $\xi \neq \lambda_j^{(n-1)}$ per ogni j, allora

$$p_n(\xi) = \underbrace{\left(\prod_{j=1}^{n-1} (\lambda_j^{(n-1)} - \xi) \right)}_{\neq 0} \left((\alpha_n - \xi) - \sum_{i=1}^{n-1} \frac{w_i^2}{\lambda_i^{(n-1)} - \xi} \right) = 0$$

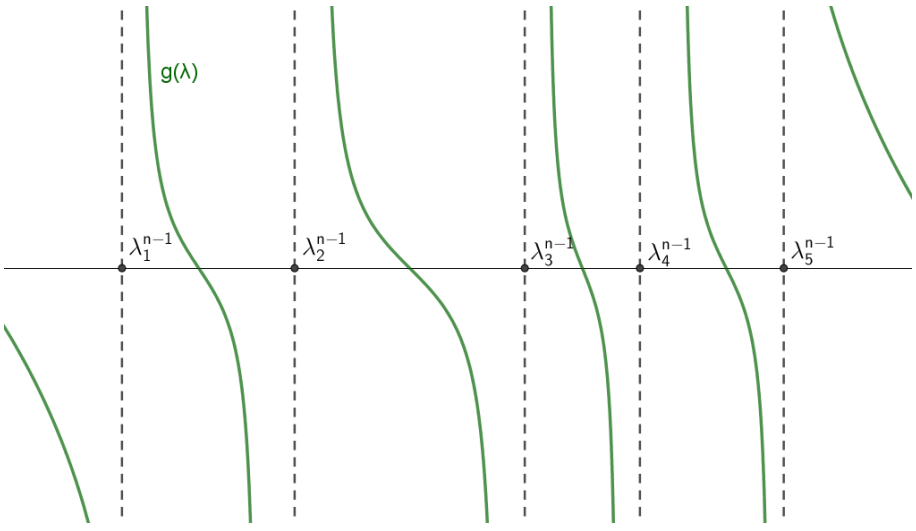
Deve quindi valere

$$0 = (\alpha_n - \xi) - \sum_{i=1}^{n-1} \frac{w_i^2}{\lambda_i^{(n-1)} - \xi} =: g(\xi)$$

Quindi gli zeri di $p_n(\lambda)$ coincidono con gli zeri di $g(\lambda)$, con $\lambda \neq \lambda_i^{(n-1)}$. Studiamo quindi $g(\xi)$. Derivando si ossserva che

$$g'(\lambda) = -\left(1 + \sum_{i=1}^{n-1} \frac{w_i^2}{(\lambda_i^{(n-1)} - \lambda)^2} \right) < 0 \quad \forall \lambda \neq \lambda_i^{(n-1)}$$

quindi $g(\lambda)$ è sempre decrescente. Inoltre, per $\lambda = \lambda_i^{(n-1)}$, $g(\lambda)$ ha degli asintoti verticali. Possiamo quindi stimare il grafico di $g(\lambda)$ nel seguente modo:



Quindi gli zeri di $g(\lambda)$ (ovvero gli zeri di $p_n(\lambda)$) sono "interlacciati" dagli zeri di $p_{n-1}(\lambda)$. Possiamo esprimere questa proprietà con la seguente catena di disuguaglianze

$$\lambda_1^{(n)} < \lambda_1^{(n-1)} < \lambda_2^{(n)} < \lambda_2^{(n-1)} < \cdots < \lambda_{n-1}^{(n)} < \lambda_{n-1}^{(n-1)} < \lambda_n^{(n)}$$

dove con $\lambda_1^{(n)}, \ldots, \lambda_n^{(n)}$ intendiamo gli zeri di $p_n(\lambda)$, ovvero gli autovalori di B_n. Osserviamo che dalla dimostrazione appena fatta segue anche che tutti gli zeri di $p_n(\lambda)$ hanno molteplicità 1, fatto che utilizzeremo fra non molto.

4 30/09/2020

4.1 Quantificare gli zeri di $p_n(\lambda)$ in $[a, b)$

Mostreremo ora un risultato che permetterà di quantificare gli zeri di $p_n(\lambda)$ (definito come nella lezione precedente) in un generico intervallo $[a, b)$.

Definizione (successione di Sturm)**.** Una successione di Sturm è una successione di polinomi $p_0(\lambda), \dots, p_m(\lambda)$ tali che

1. $p_0(\lambda)$ ha segno costante,
2. $p_i(\lambda) = 0 \Rightarrow p_{i-1}(\lambda)p_{i+1}(\lambda) < 0$
3. $p_n(\lambda) = 0 \Rightarrow p'_n(\lambda)p_{n-1}(\lambda) < 0$

Teorema 4.1. *La successione dei polinomi $\{p_i(\lambda)\}_{i=0}^n$ definita come nella lezione precedente è una successione di Sturm.*

Dimostrazione.

1. Poiché $p_0(\lambda) = 1$ è costante, a maggior ragione il segno di $p_0(\lambda)$ è costante.
2. Sia λ tale che $p_i(\lambda) = 0$. Per la relazione di ricorrenza so che

$$p_{i+1}(\lambda) = (\alpha_{i+1} - \lambda)p_i(\lambda) - |\beta_{i+1}|^2 p_{i-1}(\lambda)$$

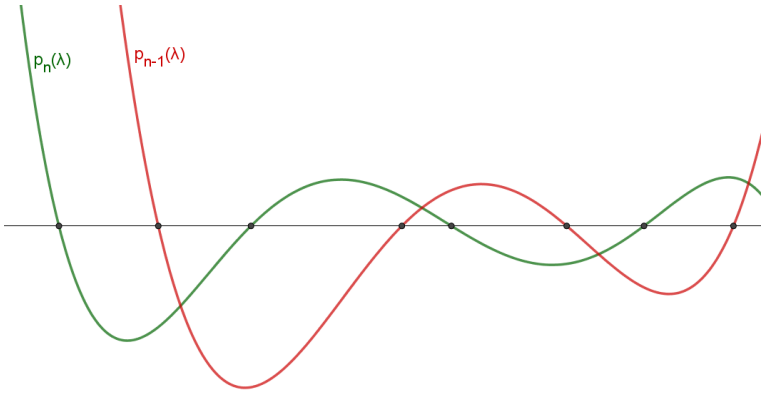
   Considero quindi il prodotto $p_{i-1}(\lambda)p_{i+1}(\lambda)$ e sostituisco

$$\begin{aligned} p_{i-1}(\lambda)p_{i+1}(\lambda) &= p_{i-1}(\lambda)[(\alpha_{i+1} - \lambda)\underbrace{p_i(\lambda)}_{=0} - |\beta_{i+1}|^2 p_{i-1}(\lambda)] = \\ &= -p_{i-1}(\lambda)|\beta_{i+1}|^2 p_{i-1}(\lambda) = \\ &= -(|\beta_{i+1}|\, p_{i-1}(\lambda))^2 \end{aligned}$$

   e poiché per quanto visto in precedenza sappiamo che, nel caso a cui ci siamo ristretti, $p_i(\lambda) \in \mathbb{R}$ per ogni i e per ogni λ tra quelli considerati, il prodotto che abbiamo sviluppato è uguale all'opposto di un quadrato di un numero reale, ed è quindi sempre negativo (non può valere uguale a zero perché altrimenti vorrebbe dire o che $\beta_{i+1} = 0$, ma li avevamo supposti tutti diversi da zero per irriducibilità, o $p_{i-1}(\lambda) = 0$, ma p_{i-1} non può avere radici in comune con p_i). Quindi $p_i(\lambda) = 0 \Rightarrow p_{i-1}(\lambda)p_{i+1}(\lambda) < 0$.
3. Sviluppando la relazione di ricorrenza si ottiene che il termine di grado massimo di $p_i(\lambda)$ ha sempre coefficiente $(-1)^i$. In particolare, per ogni $i = 1, ..., n$, vale

$$\lim_{\lambda \to -\infty} p_i(\lambda) = +\infty$$

   In particolare è vero per $i = n$ e $i = n - 1$. Sapendo inoltre che gli zeri di $p_n(\lambda)$ e di $p_{n-1}(\lambda)$ si interlacciano, possiamo stimare il grafico di $p_n(\lambda)$ e $p_{n-1}(\lambda)$ nel seguente modo:



Osserviamo innanzitutto che, dato λ tale che $p_n(\lambda) = 0$, sia $p'_n(\lambda)$ sia $p_{n-1}(\lambda)$ sono diversi da zero. Il primo perché, per quanto mostrato, $p_n(\lambda)$ non ha radici multiple. Il secondo perché p_n e p_{n-1} non hanno radici in comune.
Si dimostra facilmente, e si vede ancora più facilmente graficamente che, dato λ tale che $p_n(\lambda) = 0$, se $p'_n(\lambda) > 0$ allora $p_{n-1}(\lambda) < 0$, e viceversa se $p'_n(\lambda) < 0$ allora $p_{n-1}(\lambda) > 0$. In generale hanno segno discorde, quindi vale che: $p_n(\lambda) = 0 \Rightarrow p'_n(\lambda)p_{n-1}(\lambda) < 0$

□

Questa dimostrazione segue da fatti che avevamo dimostrato per il caso particolare $B_n \in \mathbb{R}^{n\times n}$, ma tutto ciò vale anche per $B_n \in \mathbb{C}^{n\times n}$.

Fissiamo ora un λ^* e consideriamo la successione $p_0(\lambda^*), \ldots, p_n(\lambda^*)$. A questa successione associamo una successione di segni, definita nel seguente modo:

- se $p_i(\lambda^*) \neq 0$, l'i-esimo termine della successione dei segni è uguale al segno di $p_i(\lambda^*)$.
- se $p_i(\lambda^*) = 0$, l'i-esimo termine della successione dei segni è uguale all'$(i-1)$-esimo termine della successione dei segni.

(Osserviamo che questa definizione è ben posta poiché il primo termine della successione dei segni è ben definito ed è uguale a $+$, in quanto $p_0(\lambda^*) = 1$).
Dato un λ^*, definiamo $w(\lambda^*)$ come il numero di cambi di segni nella successione appena definita.

Teorema 4.2. *Se $\{p_i(\lambda)\}_{i=0}^n$ è una successione di Sturm, il numero $w(b)-w(a)$ è uguale al numero di zeri di $p_n(\lambda)$ nell'intervallo $[a, b)$*

Dimostrazione. Facciamo variare λ con continuità tra a e b. Sicuramente il numero di cambi di segni non può cambiare a meno che λ non sia una radice di uno dei $p_i(\lambda)$. Infatti, se tutti i $p_i(\lambda)$ sono diversi da zero, allora per permanenza

del segno esiste tutto un intorno di λ in cui tutti i polinomi mantengono lo stesso segno, quindi λ non poteva essere un punto in cui cambiava il numero di cambi di segno.
Sia allora λ^* tale che $\exists i$ tale che $p_i(\lambda^*) = 0$. Consideriamo i due casi:

- se $i \neq n$, allora per proprietà 2 delle successioni di Sturm vale che

  $$p_{i-1}(\lambda^*)p_{i+1}(\lambda^*) < 0$$

  Per permanenza del segno esiste un intervallo $I = [\lambda^* - h, \lambda^* + h]$ tale che per ogni $\lambda \in I$ vale $p_{i-1}(\lambda)p_{i+1}(\lambda) < 0$. Allora in tutto questo intervallo, $p_i(\lambda)$ deve avere segno concorde con uno dei due tra $p_{i-1}(\lambda)$ e $p_{i+1}(\lambda)$, e discorde con l'altro. Quindi in generale su tutto l'intervallo I, nella sottosuccessione $p_{i-1}(\lambda), p_i(\lambda), p_{i+1}(\lambda)$ c'è sempre esattamente un cambio di segno, quindi non cambia il numero di cambi di segno. Allora Il numero di cambi di segno nella successione globale non cambia.

- se $i = n$, allora per proprietà 3 delle successioni di Sturm vale che

  $$p'_n(\lambda^*)p_{n-1}(\lambda^*) < 0$$

  Per permanenza del segno e per proprietà della derivata, esiste h sufficientemente piccolo tale che valgano:

  1. $p_{n-1}(\lambda)$ ha segno discorde da $p'_n(\lambda)$ in $[\lambda^* - h, \lambda^*)$
  2. $p_{n-1}(\lambda)$ ha segno costante in $[\lambda^* - h, \lambda^* + h]$.
  3. $p_n(\lambda)$ ha segno discorde da $p'_n(\lambda)$ in $[\lambda^* - h, \lambda^*)$
  4. $p_n(\lambda)$ ha segno concorde con $p'_n(\lambda)$ in $(\lambda^*, \lambda^* + h]$

  Da questi quattro fatti si ricava che in $[\lambda^* - h, \lambda^*)$ $p_n(\lambda)$ e $p_{n-1}(\lambda)$ hanno segno concorde, mentre in $(\lambda^*, \lambda^* + h]$ hanno segno discorde. Quindi vuol dire che in λ^* il valore $w(\lambda^*)$ "incrementa" di 1 (o più precisamente, λ^* è un punto di discontinuità a salto per $w(\lambda)$ e la lunghezza del salto è $+1$).

Riassumendo, facendo variare λ con continuità tra a e b, il valore $w(\lambda)$ non cambia per λ tale che $p_n(\lambda) \neq 0$ e aumenta di uno ogni volta che $p_n(\lambda) = 0$. Allora $w(b) - w(a)$ è uguale al numero di zeri di $p_n(\lambda)$ in $[a, b)$. □

(Il motivo per cui l'intervallo considerato è $[a, b)$ invece di $[a, b]$ è che se b fosse uno zero di $p_n(\lambda)$, comunque il valore di $w(\lambda)$ non incrementerebbe poiché il segno associato a $p_n(b) = 0$ sarebbe il segno di $p_{n-1}(b)$, che non va quindi ad incrementare di 1 il numero di cambi di segni, cosa che invece accade negli intorni destri di b).

5 02/10/2020

Abbiamo visto nelle lezioni precedenti come calcolare autovalori di una matrice hermitiana trasformandola in una matrice B tridiagonale. Nel caso però in cui la matrice di partenza non sia hermitiana non disponiamo di un algoritmo per trasformarla in una matrice tridiagonale, e possiamo solo ridurla in forma di Hessenber superiore. Vediamo quindi ora un metodo efficiente per calcolare $p(\lambda)$ e $p'(\lambda)$ di una generica matrice A in forma di Hessenberg superiore.

5.1 Metodo di Heyman

Sia $A \in \mathbb{C}^{n\times n}$ in forma di Hessenberg superiore. Supponiamo anche che sia irriducibile (ovvero che tutti gli $a_{i+1,i}$ siano diversi da zero), poiché se non lo fosse potremmo ridurla e ricondurci a due sottoproblemi più semplici.

Il Metodo di Heyman comincia fissando λ e cercando dei valori $x = (x_1, \dots, x_{n-1}, 1)^T$ e γ (tutti in funzione di λ) tali che valga:

$$(A-\lambda I)x := \begin{pmatrix} (a_{1,1}-\lambda) & \dots & \dots & a_{1,n} \\ a_{2,1} & \ddots & & \vdots \\ & \ddots & \ddots & \vdots \\ & & a_{n,n-1} & (a_{n,n}-\lambda) \end{pmatrix} \begin{pmatrix} x_1 \\ \vdots \\ x_{n-1} \\ 1 \end{pmatrix} = \begin{pmatrix} \gamma \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Partendo da questo sistema, poiché tutti gli $a_{i+1,i}$ sono diversi da zero possiamo applicare una sostituzione all'indietro dall'ultima alla seconda riga per trovare i valori di $x_1, \dots, x_{n-1}$, e una volta trovati questi possiamo utilizzare la prima riga per imporre il valore di γ. Il costo computazionale per trovare tutti gli x_i e γ è di $O(n^2)$.
Utilizziamo ora queste soluzioni per calcolare $p(\lambda)$ e $p'(\lambda)$.
Possiamo supporre $p(\lambda) = \det(A-\lambda I) \neq 0$ poiché se lo fosse disporremmo già di un autovalore e non ci sarebbe bisogno di utilizzare il metodo.
Poiché $\det(A-\lambda I) \neq 0$ possiamo utilizzare il metodo di Cramer sul sistema appena costruito. In particolare, utilizzandolo per calcolare il valore di x_n, otteniamo

$$x_n = \frac{\det \left(\begin{array}{ccc|c} (a_{1,1}-\lambda) & \dots & a_{1,n-1} & \gamma \\ a_{2,1} & \ddots & \vdots & 0 \\ & \ddots & (a_{n-1,n-1}-\lambda) & \vdots \\ & & a_{n,n-1} & 0 \end{array}\right)}{\det(A-\lambda I)}$$

Sviluppando il primo determinante (ad sull'ultima colonna con Laplace, andando ad ottenere una matrice triangolare il cui determinante è prodotto degli elementi sulla diagonale) otteniamo

$$x_n = \frac{(-1)^{n+1}\gamma\, a_{2,1} \dots a_{n,n-1}}{\det(A-\lambda I)}$$

Poiché però sappiamo che $x_n := 1$, deve valere che

$$p(\lambda) = \det(A - \lambda I) = (-1)^{n+1}\gamma\, a_{2,1} \dots a_{n,n-1}$$

quindi per calcolare $p(\lambda)$ basta calcolare γ (che richiede $O(n^2)$) e fare n moltiplicazioni, quindi in totale sempre $O(n^2)$. Ricordiamo che γ è in realtà $\gamma(\lambda)$, un valore in funzione di λ.
Per quanto riguarda $p'(\lambda)$, sappiamo per l'equazione appena ottenuta che

$$p'(\lambda) = (-1)^{n+1}\gamma'(\lambda) a_{2,1} \dots a_{n,n-1}$$

Dobbiamo quindi trovare un modo per calcolare $\gamma'(\lambda) =: \gamma'$. Torniamo al sistema iniziale, esplicitando le dipendenze da λ, e deriviamo rispetto a λ (mettendoci in ipotesi di regolarità):

$$\begin{aligned}
(A - \lambda I)x(\lambda) &= \gamma(\lambda)e_1 \\
-x(\lambda) + (A - \lambda I)x'(\lambda) &= \gamma'(\lambda)e_1 \\
(A - \lambda I)x'(\lambda) &= \gamma'(\lambda)e_1 + x(\lambda) \\
(A - \lambda I)\begin{pmatrix} x_1' \\ x_2' \\ \vdots \\ x_{n-1}' \\ 0 \end{pmatrix} &= \begin{pmatrix} \gamma' + x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ 1 \end{pmatrix}
\end{aligned}$$

In modo analogo a quanto fatto con il primo sistema, è possibile utilizzare la sostituzione all'indietro per trovare tutti gli x_i' e soprattutto γ' in $O(n^2)$. Quindi, tanto quanto per $p(\lambda)$, è possibile calcolare $p'(\lambda)$ in $O(n^2)$ e, noti entrambi, è possibile ottenere approssimazioni degli autovalori tramite metodi come il metodo di Newton.

5.2 Metodo Divide et Impera

Questo metodo, introdotto da J. M. M. Cuppen nel 1981, fornisce un metodo ricorsivo, basato sul divide et impera, per calcolare autovalori e autovettori di una qualsiasi matrice tridiagonale simmetrica.

Sia T una matrice tridiagonale simmetrica, e siano a_i, b_i i coefficienti:

$$T = \begin{pmatrix} a_1 & b_1 & & \\ b_1 & a_2 & \ddots & \\ & \ddots & \ddots & b_{n-1} \\ & & b_{n-1} & a_n \end{pmatrix}$$

La parte "divide" del "divide et impera" comincia scomponendo T nel seguente modo

$$
T = \left(\begin{array}{ccccccccc}
a_1 & b_1 & & & & & & & \\
b_1 & a_2 & \ddots & & & & & & \\
 & \ddots & \ddots & b_{m-1} & & & & & \\
 & & b_{m-1} & a_m & b_m & & & & \\
 & & & b_m & a_{m+1} & b_{m+1} & & & \\
 & & & & b_{m+1} & a_{m+2} & \ddots & & \\
 & & & & & \ddots & \ddots & b_{n-1} & \\
 & & & & & & b_{n-1} & a_n &
\end{array}\right) =
$$

$$
= \left(\begin{array}{cccc|cccc}
a_1 & b_1 & & & & & & \\
b_1 & a_2 & \ddots & & & & & \\
 & \ddots & \ddots & b_{m-1} & & & & \\
 & & b_{m-1} & a_m & b_m & & & \\
\hline
 & & & b_m & a_{m+1} & b_{m+1} & & \\
 & & & & b_{m+1} & a_{m+2} & \ddots & \\
 & & & & & \ddots & \ddots & b_{n-1} \\
 & & & & & & b_{n-1} & a_n
\end{array}\right) =
$$

$$
= \left(\begin{array}{c|c}
\hat{T}_1 & 0 \\
\hline
0 & \hat{T}_2
\end{array}\right) + \left(\begin{array}{cc|cc}
 & & 0 & \\
 & & & b_m \\
\hline
 & b_m & & \\
 & & 0 &
\end{array}\right)
$$

e sottraendo e sommando b_m a T in posizione (m,m) e $(m-1,m-1)$ otteniamo la scomposizione

$$T = \left(\begin{array}{c|c} T_1 & 0 \\ \hline 0 & T_2 \end{array}\right) + \left(\begin{array}{c|c} & \\ \hline & \begin{matrix} b_m & b_m \\ b_m & b_m \end{matrix} \end{array}\right) =$$

$$= \left(\begin{array}{c|c} T_1 & 0 \\ \hline 0 & T_2 \end{array}\right) + b_m \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 1 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} (0 \dots 0\,1\,1\,0 \dots 0) =:$$

$$=: \begin{pmatrix} T_1 & 0 \\ 0 & T_2 \end{pmatrix} + b_m v v^T$$

dove $v := e_m + e_{m+1}$
T_1 e T_2 sono simmetriche quindi ortogonalmente diagonalizzabili. Supponiamo, per ricorrenza, di conoscere le matrici ortogonali Q_1, Q_2 e le matrici diagonali Λ_1, Λ_2 tali che

- $T_1 = Q_1 \Lambda_1 Q_1^T$
- $T_2 = Q_2 \Lambda_2 Q_2^T$

Per ragionamenti già fatti in precedenza, Λ_i ha sulla diagonale gli autovalori di T_i e Q_i ha per colonne autovettori di T_i (conoscere queste matrici è quindi equivalente a conoscere autovalori e autovettori di T_i, che è quello che andremo a fare adesso per T).
Prendendo u il vettore tale che

$$v = \begin{pmatrix} Q_1 & 0 \\ 0 & Q_2 \end{pmatrix} u$$

possiamo riscrivere la scomposizione di T come

$$T =: \begin{pmatrix} T_1 & \\ & T_2 \end{pmatrix} + b_m v v^T =$$

$$= \begin{pmatrix} Q_1 & \\ & Q_2 \end{pmatrix} \left[\begin{pmatrix} \Lambda_1 & \\ & \Lambda_2 \end{pmatrix} + b_m u u^T \right] \begin{pmatrix} Q_1^T & \\ & Q_2^T \end{pmatrix} =:$$

$$=: \begin{pmatrix} Q_1 & \\ & Q_2 \end{pmatrix} [D + b_m u u^T] \begin{pmatrix} Q_1^T & \\ & Q_2^T \end{pmatrix}$$

con $D := \begin{pmatrix} \Lambda_1 & \\ & \Lambda_2 \end{pmatrix}$. Osserviamo che poiché è stata ottenuta per similitudine, gli autovalori di $D + b_m uu^T$([2]) sono gli stessi di T. Calcoliamo ora il polinomio caratteristico di $D + b_m uu^T$.
Poiché $D + b_m uu^T$ dista da D di una correzione di rango 1, è ragionevole supporre che gli autovalori siano diversi, e quindi è possibile restringere la ricerca ai λ tali che $\det(D - \lambda I) \neq 0$, cioè che $D - \lambda I$ sia invertibile (quindi detti d_i gli elementi sulla diagonale di D, stiamo considerando $\lambda \neq d_i$).

$$\begin{aligned} p(\lambda) &= \det(D + b_m uu^T - \lambda I) = \\ &= \det(D - \lambda I + b_m uu^T) = \\ &= \det((D - \lambda I)(I + b_m (D - \lambda I)^{-1} uu^T)) = \\ &= \det(D - \lambda I) \det(I + b_m (D - \lambda I)^{-1} uu^T) \end{aligned}$$

Per le assunzioni appena fatte, $p(\lambda)$ si annulla se e solo se $\det(I + b_m(D - \lambda I)^{-1} uu^T)$ si annulla, quindi il problema equivale a cercare gli autovalori di $I + b_m(D - \lambda I)^{-1} uu^T$.
Ponendo $w := b_m(D - \lambda I)^{-1} u$ ci accorgiamo che $I + b_m(D - \lambda I)^{-1} uu^T = I + wu^T$, ovvero la matrice che stiamo considerando è uguale all'identità più una correzione di rango 1. In questo caso si può dimostrare([3]) che il determinante di tale matrice è:

$$\det(I + wu^T) = 1 + u^T w$$

Segue allora che

$$\begin{aligned} \det(I + b_m(D - \lambda I)^{-1} uu^T) &= \det(I + wu^T) = \\ &= 1 + u^T w = \\ &= 1 + b_m u^T (D - \lambda I)^{-1} u = \\ &= 1 + b_m \sum_{i=1}^{n} \frac{u_i^2}{d_i - \lambda} =: \\ &=: f(\lambda) \end{aligned}$$

che è ben definito ricordandoci che $\lambda \neq d_i$.
Per quanto visto, vale $p(\lambda) = 0 \iff f(\lambda) = 0$. Studiamo quindi $f(\lambda)$.
Derivando otteniamo

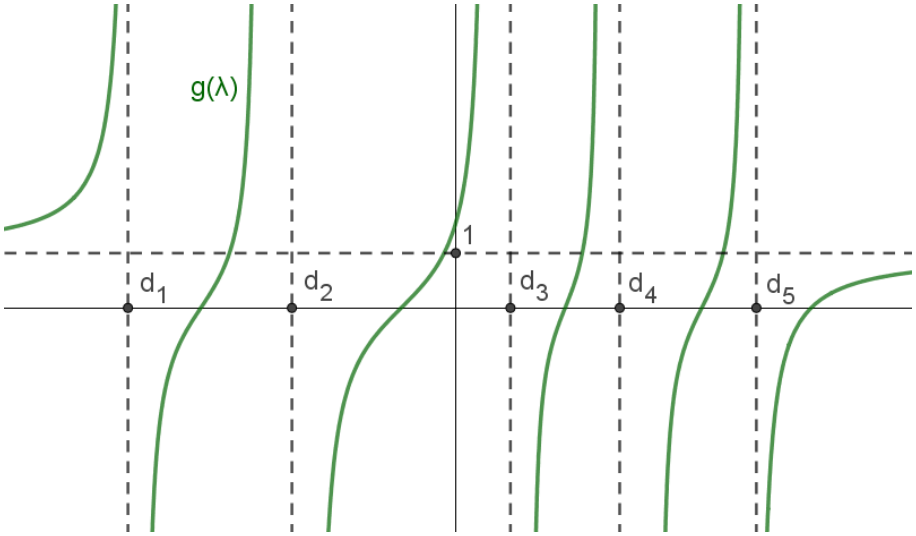
$$f'(\lambda) = b_m \sum_{i=1}^{n} \frac{u_i^2}{(d_i - \lambda)^2}$$

Quindi il segno di $f'(\lambda)$ dipende esclusivamente da b_m. Consideriamo il caso $b_m > 0$ (il caso $b_m < 0$ è del tutto analogo).

Poiché $b_m > 0$, $f(\lambda)$ è sempre crescente. Inoltre, sappiamo che in tutti i d_i

[2] : "$b_m uuuuuu^T$"
[3] La dimostrazione non è stata fatta in classe. Per completezza, comunque, la riporto come appendice in fondo

ha asintoti verticali e che $lim_{\lambda \to \pm\infty} f(\lambda) = 1$. Allora possiamo approssimare il grafico di $f(\lambda)$ nel seguente modo (supponendo senza perdita di generalità che $d_1 \leq \cdots \leq d_n$):



Quindi gli autovalori di T_1 e T_2 "separano" gli autovalori di T.
Per calcolare approssimazioni degli autovalori di T possiamo quindi limitarci a calcolare $f(\lambda)$ e $f'(\lambda)$.
Il numero di operazioni da dover svolgere per calcolare $f(\lambda)$ è $3n$[operazioni per il singolo addendo]$+(n-1)$[somme della sommatoria]$+1+1 = 4n-1 = O(n)$.
Allo stesso modo anche per $f'(\lambda)$ è $O(n)$. Quindi il costo computazionale per trovare tutti gli autovalori è $O(n^2)$ (poiché sono n autovalori).

Per concludere rimane da calcolare gli autovettori di T. Consideriamo la seguente proposizione:

Proposizione 5.1. *Se α è autovalore di $D + b_m uu^T$, allora $(D-\alpha I)^{-1}u$ è un autovettore di $D + b_m uu^T$ relativo all'autovalore α.*

Dimostrazione. Dobbiamo dimostrare che vale $(D + b_m uu^T)(D - \alpha I)^{-1}u =$

$\alpha(D-\alpha I)^{-1}u$. Consideriamo allora $(D+b_m uu^T)(D-\alpha I)^{-1}u$ e sviluppiamo

$$\begin{aligned}
(D+b_m uu^T)(D-\alpha I)^{-1}u &= (D-\alpha I+\alpha I+b_m uu^T)(D-\alpha I)^{-1}u = \\
&= ((D-\alpha I)+\alpha I+b_m uu^T)(D-\alpha I)^{-1}u = \\
&= u+\alpha(D-\alpha I)^{-1}u+b_m uu^T(D-\alpha I)^{-1}u = \\
&= u+\alpha(D-\alpha I)^{-1}u+b_m u(u^T(D-\alpha I)^{-1}u) = \\
&= u+\alpha(D-\alpha I)^{-1}u+u[b_m u^T(D-\alpha I)^{-1}u] = \\
&= u+\alpha(D-\alpha I)^{-1}u+u[\underbrace{f(\alpha)}_{=0}-1] = \\
&= u+\alpha(D-\alpha I)^{-1}u-u = \\
&= \alpha(D-\alpha I)^{-1}u
\end{aligned}$$

dove $f(\alpha)=0$ perché avevamo dimostrato che gli autovalori di $D+b_m uu^T$ coincidono con gli zeri di f. □

Allora per trovare gli autovettori di $D+b_m uu^T$ basta calcolare $(D-\alpha I)^{-1}u$ e poiché $D-\alpha I$ è diagonale, il costo computazionale è di $O(n)$ per autovettore. In totale, il costo computazionale è di $O(n^2)$ per tutti gli autovettori.
Una volta trovati gli autovettori di $D+b_m uu^T$, per trovare gli autovettori di T basta moltiplicarli per la matrice $\begin{pmatrix} Q_1 & \\ & Q_2 \end{pmatrix}$. Detta quindi Q' la matrice di autovettori di $D+b_m uu^T$, il conto da svolgere è $\begin{pmatrix} Q_1 & \\ & Q_2 \end{pmatrix} Q'$. In generale il prodotto tra matrici richiede $O(n^3)$, ma possiamo sfruttare una caratteristica di Q' per diminuire il costo computazionale.

Definizione (Matrici Cauchy-like)**.** Una matrice C dove tutti gli elementi sono della forma $c_{i,j}=\frac{r_i s_j}{x_i-y_j}$ si chiama matrice Cauchy-like.

Q' è una matrice Cauchy-like. Infatti, la generica colonna di Q' è della forma

$$(D-\alpha I)^{-1}u = \begin{pmatrix} \frac{u_1}{d_1-\alpha} \\ \vdots \\ \frac{u_n}{d_n-\alpha} \end{pmatrix}$$

Esistono algoritmi per il prodotto tra una matrice Cauchy-like e una generica matrice il cui costo computazionale è $O(n^2\log(n))$, quindi nel nostro caso possiamo ridurre molto il costo computazionale (invece di avere $O(n^3)$).

6 07/10/2020

6.1 Metodo fattorizzazione QR

E' possibile sfruttare la fattorizzazione QR di una matrice per calcolarne autovalori e autovettori.
Prima di cominciare con il metodo QR ricordiamo che data una generica matrice A la fattorizzazione QR esiste sempre, ma non è unica (a differenza, per esempio, della fattorizzazione LU che non esiste sempre ma se esiste è unica).

Definizione. Una matrice diagonale S con

$$S = \begin{pmatrix} \theta_1 & & \\ & \ddots & \\ & & \theta_n \end{pmatrix}$$

con $|\theta_i| = 1$ per ogni i si dice matrice di fase.

Come detto poco fa, la fattorizzazione QR non è unica. Infatti, sia $A = QR$ e S è una matrice di fase. Vale

$$A = \underbrace{QS}_{Q'}\underbrace{S^H R}_{R'}$$

e $A = Q'R'$ è ancora una valida fattorizzazione QR di A. Se A è invertibile, però, si può dimostrare che la fattorizzazione QR è unica a meno di matrice di fase.
Dette infatti $A = QR = Q'R'$, poiché A è invertibile esistono le inverse di Q, R, Q' e R' e in particolare da $QR = Q'R'$ si ricava $Q'^H Q = R'R^{-1}$. Allora $R'R^{-1}$ è una matrice triangolare superiore unitaria, ed è facile verificare che le matrici triangolari unitarie sono diagonali, e quindi di fase.

Il metodo si basa sull'iterazione di uno stesso passaggio, partendo da una matrice A e producendo matrici A_k così definite

$$\begin{cases} A_1 = A \\ A_{k+1} = R_k Q_k \quad \text{con } A_k = Q_k R_k \text{ una fattorizzazione QR di } A \end{cases}$$

Osservazione. Tutte le matrici A_i, A_j sono simili.

Per dimostrarlo basta in realtà dimostrare che per ogni k, A_k e A_{k+1} sono simili. Infatti vale

$$\begin{aligned} A_{k+1} &= R_k Q_k = \\ &= (Q_k^H Q_k) R_k Q_k = \\ &= Q_k^H (Q_k R_k) Q_k = \\ &= Q_k^H A_k Q_k \end{aligned}$$

Osservazione. Poiché stiamo lavorando con trasformazioni di tipo unitario, per quanto visto alle prime lezioni il problema è ben condizionato.

Mostreremo che, sotto opportune ipotesi, la successione delle matrici A_k converge ad una matrice triangolare. Poiché le A_k sono tutte simili, hanno in particolare gli stessi autovalori di A, ma nel caso di una matrice triangolare essi sono molto più facili da calcolare.

6.2 Convergenza del metodo QR

Teorema 6.1. *Sia $A \in \mathbb{C}^{n\times n}$ una matrice complessa i cui autovalori hanno tutti moduli distiniti, con*

$$|\lambda_1| > \ldots |\lambda_n| > 0$$

Sia X la matrice tale che $A = XDX^{-1}$ con D diagonale (A è diagonalizzabile perché ha n autovalori distinti) e supponiamo che X^{-1} ammetta fattorizzazione $X = LU$ di tipo LU.
Allora esiste una successione di matrici $\{S_k\}_{k\in\mathbb{N}}$ tale che valgano

$$\lim_{k\to+\infty} S_k^H R_k S_{k-1} = \lim_{k\to+\infty} S_{k-1}^H A_k S_{k-1} = T$$
$$\lim_{k\to+\infty} S_{k-1}^H Q_k S_k = I$$

con T una matrice triangolare superiore.

Osservazione. Se A è hermitiana, anche T sarà hermitiana e quindi diagonale.

Osservazione. Questo teorema è enunciato in forma più debole (cioè con ipotesi più forti) per semplificare la dimostrazione, che comunque non risulta corta. Lo stesso risultato si può ottenere con ipotesi più deboli.

Dimostrazione. Definiamo le matrici $H_k := Q_1 Q_2 \ldots Q_k$ e $U_k := R_k \ldots R_2 R_1$. Osserviamo che poiché il prodotto di unitarie è unitario e il prodotto di triangolari superiori è tirangolare superiore, H_k e U_k sono rispettivamente unitaria e triangolare superiore.
Cominciamo mostrando che vale $A^k = H_k U_k$ (non si confonda A^k con A_k. Stiamo parlando del prodotto di A con se stessa k volte). Dimostriamolo per induzione

- $k = 1$. Allora $A^1 = A = A_1 = Q_1 R_1 = H_1 U_1$
- Supponiamo che la tesi sia vera per k, dimostriamo che vale per $k+1$. Osserviamo che poiché vale $A_k = Q_k R_k$ e $A_{k+1} = R_k Q_k$, moltiplicando la seconda equazione a sinistra per Q_k otteniamo

$$Q_k A_{k+1} = Q_k (R_k Q_k) = (Q_k R_k) Q_k = A_k Q_k$$

In particolare vale

$$\begin{aligned} H_k A_{k+1} &= Q_1 \dots Q_k A_{k+1} = \\ &= Q_1 \dots Q_{k-1} A_k Q_k = \\ &= Q_1 \dots Q_{k-2} A_{k-1} Q_{k-1} Q_k = \\ &= \dots = \\ &= A_1 Q_1 \dots Q_k = \\ &= A H_k \end{aligned}$$

Consideriamo ora il prodotto $H_{k+1}U_{k+1}$ e mostriamo che vale $A^{k+1} = H_{k+1}U_{k+1}$

$$\begin{aligned} H_{k+1}U_{k+1} &= Q_1 \dots Q_k Q_{k+1} R_{k+1} R_k \dots R_1 = \\ &= Q_1 \dots Q_k A_{k+1} R_k \dots R_1 = \\ &= H_k A_{k+1} U_k = \\ &= A H_k U_k = \\ &= A A^k = \\ &= A^{k+1} \end{aligned}$$

Per ipotesi sappiamo che $A = XDX^{-1}$ con $X^{-1} = LU$. Con un semplice conto si verifica che $A^k = XD^kX^{-1}$, e sostituendo a X^{-1} la sua fattorizzazione otteniamo $A^k = XD^kLU$. Inserendo tra L e U la matrice identità $I = D^{-k}D^k$ otteniamo

$$A^k = X(D^kLD^{-k})D^kU$$

dove la matrice D^kLD^{-k} è triangolare superiore della seguente forma

$$(D^kLD^{-k})_{i,j} = \begin{cases} L_{i,j}\left(\frac{\lambda_i}{\lambda_j}\right)^k & \text{se } i > j \\ 1 & \text{se } i = j \\ 0 & \text{se } i < j \end{cases}$$

In particolare, poiché $i > j$ implica $|\lambda_i| < |\lambda_j|$, per $k \to +\infty$ vale $\left(\frac{\lambda_i}{\lambda_j}\right)^k \to 0$. Possiamo quindi scrivere $D^kLD^{-k} = I + E_k$ dove E_k è una certa matrice tale che $\lim_{k\to+\infty} E_k = 0$
Scriviamo quindi $A^k = X(I + E_k)D^kU$
Sia ora $X =$ una fattorizzazione QR di X. Sostituiamo e otteniamo

$$\begin{aligned} A^k &= X(I + E_k)D^kU = \\ &= QR(I + E_k)D^kU = \\ &= QR(I + E_k)R^{-1}RD^kU = \\ &= Q(I + RE_kR^{-1})RD^kU \end{aligned}$$

Sia ora $I+RE_kR^{-1} = P_kT_k$ una fattorizzazione QR di $I+RE_kR^{-1}$. Sostituiamo e otteniamo

$$\begin{aligned} A^k &= Q(I + RE_kR^{-1})RD^kU = \\ &= Q(P_kT_k)RD^kU = \\ &= (QP_k)(T_kRD^kU) \end{aligned}$$

che è un'altra valida fattorizzazione QR di A^k, poiché la prima matrice è prodotto di unitarie e la seconda è prodotto di triangolari superiori e diagonali. Allora per quanto visto, poiché A è invertibile (che segue da $|\lambda_1| > \ldots |\lambda_n| > 0$), la fattorizzazione QR è unica a meno di matrice di fase, ovvero esiste una matrice $\hat{S}_k$ tale che

$$H_k = QP_k\hat{S}_k^H \tag{2}$$

$$U_k = \hat{S}_kT_kRD^kU \tag{3}$$

Otteniamo ora due risultati che ci serviranno poi.

- Possiamo scrivere $Q_k = (H_{k-1})^{-1}H_k$. Allora per la (2), vale

  $$Q_k = \hat{S}_{k-1}P_{k-1}^H \underbrace{Q^HQ}_{=I} P_k\hat{S}_k^H$$

  da cui si ottiene

  $$\hat{S}_{k-1}^HQ_k\hat{S}_k = P_{k-1}^HP_k \tag{4}$$

- Possiamo scrivere $R_k = U_k(U_{k-1})^{-1}$. Allora per la (3), vale

  $$\begin{aligned} R_k &= (\hat{S}_kT_kR\underbrace{D^kU}_{=I})\underbrace{(U^{-1}D^{1-k}}_{=D}R^{-1}T_{k-1}^{-1}\hat{S}_{k-1}^H) = \\ &= \hat{S}_kT_kRDR^{-1}T_{k-1}^{-1}\hat{S}_{k-1}^H \end{aligned}$$

  da cui si ottiene

  $$\hat{S}_k^HR_k\hat{S}_{k-1} = T_kRDR^{-1}T_{k-1}^{-1} \tag{5}$$

Torniamo ora alla matrice $I + RE_kR^{-1} = P_kT_k$. Poiché sappiamo che

$$\lim_{k\to+\infty} P_kT_k = \lim_{k\to+\infty} I + RE_kR^{-1} = I$$

si può dimostrare([4]) che allora esiste una successione di matrici di fase $\{\check{S}_k\}$ tale che

$$\lim_{k\to+\infty} P_k\check{S}_k = \lim_{k\to+\infty} \check{S}_k^HT_k = I$$

[4] A lezione l'abbiamo dato praticamente per scontato. La dimostrazione è riportata come appendice.

Definiamo allora $S_k := \hat{S}_k \check{S}_k$.
Per la (4) vale $S_{k-1}^H Q_k S_k = \check{S}_{k-1}^H P_{k-1}^H P_k \check{S}_k$ e quindi

$$\lim_{k\to+\infty} S_{k-1}^H Q_k S_k = \lim_{k\to+\infty} \underbrace{\check{S}_{k-1}^H P_{k-1}^H}_{\to I} \underbrace{P_k \check{S}_k}_{\to I} = I$$

Per la (5) vale $S_k^H R_k S_{k-1} = \check{S}_k^H T_k R D R^{-1} T_{k-1}^{-1} \check{S}_{k-1}$ e quindi

$$\lim_{k\to+\infty} S_k^H R_k S_{k-1} = \lim_{k\to+\infty} \underbrace{\check{S}_k^H T_k}_{\to I} R D R^{-1} \underbrace{T_{k-1}^{-1} \check{S}_{k-1}}_{\to I} = R D R^{-1} =: T$$

da cui segue

$$\begin{aligned}
\lim_{k\to+\infty} S_k^H A_k S_{k-1} &= \lim_{k\to+\infty} S_k^H Q_k R_k S_{k-1} = \\
&= \lim_{k\to+\infty} S_k^H Q_k (S_k S_k^H) R_k S_{k-1} = \\
&= \lim_{k\to+\infty} \underbrace{S_k^H Q_k S_k}_{\to I} \underbrace{S_k^H) R_k S_{k-1}}_{\to T} = \\
&= T
\end{aligned}$$

che conclude la dimostrazione. □

6.3 Costo computazionale del metodo QR

Per come abbiamo definito l'algoritmo, partendo da una generica matrice A, ad ogni passo bisogna prima compiere una fattorizzazione QR e poi svolgere il prodotto $R_k Q_k$. Il costo di ogni passo è quindi un $O(n^3)$.

Adottando però il metodo a due fasi, ovvero portando prima A ad una forma B più comoda, è possibile abbassare il costo dell'algoritmo.
Ad esempio, con i metodi visti nelle lezioni precedenti è possibile portare $A \rightsquigarrow B$ in forma di Hessenberg (o addirittura tridiagonale se A è hermitiana). Il costo di questa trasformazione è $O(n^3)$, per cui a priori non sembrerebbe un grande vantaggio. La trasformazione però è vantaggiosa, poiché essa è un'operazione che va compiuta una volta sola e non ad ogni iterazione dell'algoritmo (la cui quantità dipende dalla precisione con cui vogliamo gli autovalori). La complessità di ogni passo iterativo partendo dalla matrice B diminuisce molto. Infatti:

- Se B è in forma di Hessenberg (ovvero, se A non è hermitiana) il costo di ogni iterazione è di $O(n^2)$.
- Se B è tridiagonale (ovvero, se A è hermitiana) il costo di ogni iterazione è di solo $O(n)$.

Il problema di questo algoritmo è che se esistono autovalori di modulo molto vicino, la convergenza può essere molto lenta. In questo caso si possono adottare tecniche di traslazione dello spettro (che danno all'algoritmo il nome di "QR con shift) che vedremo la prossima volta.

7 09/10/2020

7.1 Condizioni di arresto per il metodo QR

Abbiamo visto che con il metodo QR è possibile ottenere una successione di matrici $\{A_k\}$ convergenti ad una matrice triangolare con le quali è possibile approssimare gli autovalori della matrice A iniziale (trasformata almeno in forma di Hessenberg per efficienza). Dopo aver visto il metodo e il costo computazionale, l'unica domanda a cui dobbiamo rispondere è specificare la condizione di arresto per questo metodo iterativo.
In generale, si fissa un $\varepsilon > 0$ piccolo a priori, e si continua ad iterare finché non vale che per un certo indice p con $p \in \{1, ..., n-1\}$ l'elemento $a^{(k)}_{p+1,p}$ diventa "sufficientemente piccolo" in modulo. Con "sufficientemente piccolo" intendiamo che valga la seguente relazione:

$$\left|a^{(k)}_{p+1,p}\right| \leq \varepsilon \left(\left|a^{(k)}_{p,p}\right| + \left|a^{(k)}_{p+1,p+1}\right|\right)$$

Nonappena vale questa condizione, possiamo scomporre la matrice A_k in blocchi nel seguente modo

$$A_k = \left(\begin{array}{c|c} B_k & D_k \\ \hline E_k & C_k \end{array}\right)$$

dove $B_k \in \mathbb{C}^{p\times p}$, $C_k \in \mathbb{C}^{(n-p)\times(n-p)}$ e E_k è una matrice con elemento in alto a destra pari a $a^{(k)}_{p+1,p}$ e tutti gli altri elementi nulli.
Poiché $a^{(k)}_{p+1,p}$ è sufficientemente piccolo, E_k è sufficientemente vicina alla matrice nulla per poter approssimare gli autovalori di A_k con quelli di B_k e C_k e possiamo quindi ricondurci a due problemi più semplici.

7.2 Metodo QR con shift

Abbiamo visto che una condizione necessaria per poter utilzzare il metodo QR è che gli autovalori della matrice considerata siano tali che

$$|\lambda_1| > \ldots |\lambda_n| > 0$$

e in questo caso la velocità di convergenza è tanto maggiore quanto minore è il rapporto $\frac{|\lambda_i|}{|\lambda_j|}$ al variare di i e j. In generale, poiché vale la relazione tra gli autovalori scritta sopra, la velocità di convergenza è dettata da $\max_{1\leq i<n} \frac{|\lambda_i|}{|\lambda_{i+1}|}$.
Come comportarsi nel caso in cui questo max sia molto vicino a 1? In questo caso possiamo utilizzare strategie di traslazione dello spettro della matrice per accelerare la velocità di convergenza. Il metodo QR passante per traslazione dello spettro prende il nome di metodo QR con shift.

Sia μ un valore che approssima un certo autovalore λ (vedremo tra poco tecniche per scegliere μ). Consideriamo il metodo QR applicato alla matrice $A - \mu I$,

che genererà delle certe successioni di matrici $\{Q_k\}$, $\{R_k\}$. Osserviamo che le stesse matrici $\{Q_k\}$, $\{R_k\}$ possono essere ottenute con la seguente relazione di ricorrenza:

$$\begin{cases} A_1 = A \\ A_k - \mu I = Q_k R_k \\ A_{k+1} = R_k Q_k + \mu I \end{cases}$$

Mostriamo, come abbiamo fatto per il metodo QR semplice, che anche le A_k ottenute con il metodo QR con shift sono tra loro simili (e che quindi preservano gli autovalori):

$$\begin{aligned} Q_k A_{k+1} &= Q_k(R_k Q_k + \mu I) = \\ &= Q_k R_k Q_k + \mu Q_k = \\ &= (A_k - \mu I)Q_k + \mu Q_k = \\ &= A_k Q_k - \mu Q_k + \mu Q_k = \\ &= A_k Q_k \\ &\Downarrow \\ A_{k+1} &= Q_k^{-1} A_k Q_k \end{aligned}$$

Vediamo ora come scegliere μ. Partendo dalla matrice $A - \mu I$, gli autovalori che andremo ad ottenere (e che determineranno la velocità di convergenza) non saranno più i λ_i di A, ma i $\lambda_i - \mu$.
Poiché la velocità di convergenza è dettata da $\max_{1 \leq i < n} \frac{|\lambda_i - \mu|}{|\lambda_{i+1} - \mu|}$ può convenire scegliere $\mu \approx \lambda_n$. Poiché però il valore di λ_n non è noto a priori, dobbiamo trovare una strategia per approssimare λ_n (in modo anche grossolano) per poter poi approssimare tutti gli autovalori efficientemente con QR con shift. Un modo per farlo è applicare il metodo QR senza shift per un numero q fissato di volte, prendere $\mu = a_{n,n}^{(q)}$ (che sarà un'approssimazione di λ_n) e applicare poi il metodo QR con shift con μ.

7.3 Caso $|\lambda_{n-1}| = |\lambda_n|$ e doppio shift implicito

Abbiamo visto come risolvere il problema di convergenza lenta nel caso valga la relazione tra i moduli degli autovalori con massimo tra i rapporti molto vicino ad 1. In realtà è possibile velocizzare la convergenza anche nel caso particolare di $|\lambda_{n-1}| = |\lambda_n|$. In questo caso, si procede prendendo μ_k diverso ad ogni passaggio. La strategia per scegliere μ_k è la seguente: consideriamo la matrice $A_{n-1}^{(k)}$ definita come

$$A_{n-1}^{(k)} = \begin{pmatrix} a_{n-1,n-1}^{(k)} & a_{n-1,n}^{(k)} \\ a_{n,n-1}^{(k)} & a_{n,n}^{(k)} \end{pmatrix}$$

e prendiamo μ_k uguale all'autovalore di $A_{n-1}^{(k)}$ più vicino a $a_{n,n}^{(k)}$.
Possiamo fare ciò perché calcolare esplicitamente gli autovalori di una matrice 2×2 è facile, ma corriamo il rischio di introdurre nel metodo valori complessi.

Non c'è garanzia, infatti, che gli autovalori di una matrice 2×2 siano in generale reali anche se stiamo lavorando con matrici reali, e andare ad introdurre elementi complessi significa aumentare il costo computazionale (in generale le operazioni tra complessi richiedono maggiore costo computazionale delle stesse operazioni tra reali, banalmente perché i complessi vengono visti come coppie di reali e quindi una stessa operazione deve essere compiuta almeno due volte).

E' possibile evitare di passare per elementi complessi (partendo da A reale) applicando la tecnica del doppio shift implicito, ovvero passare direttamente da A_k a A_{k+2} usando $\mu_k = \alpha$ e $\mu_{k+1} = \beta$ dove α e β sono i due autovalori di $A_{n-1}^{(k)}$. Cominciamo mostrando che con $A_k \rightsquigarrow A_{k+2}$ si ottiene una matrice senza elementi complessi.

Consideriamo le relazioni che definiscono A_k, A_{k+1}, A_{k+2}.

$$\begin{cases} A_k - \alpha I = Q_k R_k \\ A_{k+1} = R_k Q_k + \alpha I \\ A_{k+1} - \beta I = Q_{k+1} R_{k+1} \\ A_{k+2} = R_{k+2} Q_{k+2} + \beta I \end{cases}$$

Osservazione. α e β possono essere complessi, ma in quanto radici di uno stesso polinomio reale di grado due, sicuramente $(\alpha + \beta)$ e $\alpha\beta$ sono reali (in quanto coefficienti di questo polinomio)

Poniamo $Z = Q_k Q_{k+1}$ e $S = R_{k+1} R_k$, rispettivamente matrici unitaria e triangolare superiore (in quanto prodotto di unitarie e prodotto di triangolari superiori). Considero il prodotto $M := ZS$

$$\begin{aligned} ZS &= Q_k Q_{k+1} R_{k+1} R_k = \\ &= Q_k (A_{k+1} - \beta I) R_k = \\ &= Q_k (R_k Q_k + \alpha I - \beta I) R_k = \\ &= Q_k (R_k Q_k + \alpha I - \beta I) R_k = \\ &= Q_k (R_k Q_k R_k + \alpha R_k - \beta R_k) = \\ &= Q_k R_k (Q_k R_k + \alpha I - \beta I) = \\ &= Q_k R_k (A_k - \alpha I + \alpha I - \beta I) = \\ &= Q_k R_k (A_k - \beta I) = \\ &= (A_k - \alpha I)(A_k - \beta I) = \\ &= A_k^2 - (\alpha + \beta) A_k + \alpha\beta I = M \end{aligned}$$

Quindi se A_k è reale, anche M è reale. Consideriamo ora il prodotto ZA_{k+2}

$$\begin{aligned} ZA_{k+2} &= Q_kQ_{k+1}(R_{k+1}Q_{k+1} + \beta I) = \\ &= Q_kQ_{k+1}R_{k+1}Q_{k+1} + \beta Q_kQ_{k+1} = \\ &= Q_k(A_{k+1} - \beta I)Q_{k+1} + \beta Q_kQ_{k+1} = \\ &= Q_kA_{k+1}Q_{k+1} = \\ &= Q_k(R_kQ_k + \alpha I)Q_{k+1} = \\ &= Q_kR_kQ_kQ_{k+1} + \alpha Q_kQ_{k+1} = \\ &= (A_k - \alpha I)Q_kQ_{k+1} + \alpha Q_kQ_{k+1} = \\ &= A_kQ_kQ_{k+1} = \\ &= A_kZ \\ &\Downarrow \\ A_{k+2} &= Z^HA_kZ \end{aligned}$$

Quindi poiché M è reale e ZS è una fattorizzazione QR di M, anche Z è reale, quindi anche A_{k+2} lo è. Inoltre, per passare da A_k ad A_{k+2} basta conoscere una fattorizzazione QR di M. Lo svantaggio è che ad ogni iterazione abbiamo un costo di $O(n^3)$ aggiuntivo per fattorizzare M. Vediamo ora un metodo per abbattere questo costo da $O(n^3)$ ad $O(n^2)$.

Teorema 7.1 (del Q implicito)**.** *Sia $A = QHQ^T = VGV^T$ dove H e G sono in forma di Hessenberg superiore e irriducibili, Q e V sono ortogonali. Siano $q_1, \dots, q_n$ le colonne di Q e $v_1, \dots, v_n$ le colonne di V.*
Se $q_1 = v_1$, allora per ogni $i = 2, \dots, n$ vale $q_i = \pm v_i$

Dimostrazione. Consideriamo la matrice $W = V^TQ = (w_1, \dots, w_n)$. Poiché $q_1 = v_1$ e Q ortogonale, vale che $w_1 = v_1^Tq_1 = q_1^Tq_1 = e_1$. Considero il prodotto GW, e ottengo

$$\begin{aligned} GW &= GV^TQ = \\ &= V^TAQ = \\ &= V^TQH = \\ &= WH \end{aligned}$$

e leggendo questa equazione sull'i-esima riga otteniamo

$$\begin{aligned} Gw_i &= (GW)_i = \\ &= (WH)_i = \\ &= \sum_{j=1}^{i+1} h_{j,i}w_j = \end{aligned}$$

da cui segue che

$$h_{i+1,i}w_{i+1} = Gw_i - \sum_{j=1}^{i} h_{j,i}w_j$$

Per $i = 1$ otteniamo

$$h_{2,1}w_2 = \underbrace{Gw_1}_{=Ge_1} - \underbrace{h_{1,1}w_1}_{=h_{1,1}e_1}$$

Poiché G è in forma di Hessenberg superiore, Gw_1 ha al più le prime due entrate non nulle e le restanti nulle. Allora, anche $h_{2,1}w_2$ (e quindi w_2) ha al più le prime due entrate non nulle e le restanti nulle.
Per induzione su i (il passo induttivo è analogo al passo base appena mostrato) si dimostra che W è triangolare superiore. Poiché però $W = V^TQ$ è anche ortogonale, W deve essere per forza diagonale. Poiché W è diagonale ortogonale, $w_i = \pm e_i$ e quindi $q_i = \pm v_i$. □

Con questo risultato è possibile sviluppare un algoritmo che ci permetta di costruire, al costo di $O(n^2)$, una matrice Z tale che $M = ZS$ e $A_{k+2} = Z^H A_k Z$.

Per fare ciò, cominciamo calcolando la prima colonna di M. Consideriamo il caso $k = 1$ per semplicità dei conti (dove però questa restrizione non è una perdita di generalità: la prima colonna di M resta comunque più semplice da calcolare rispetto ad M e la sua fattorizzazione QR, e questo è quello che importa).
Vale

$$Me_1 = (A_1^2 - (\alpha+\beta)A_1 + \alpha\beta I)e_1 = \begin{pmatrix} a_{1,1}^2 + a_{1,2}a_{2,1} - (\alpha+\beta)a_{1,1} + \alpha\beta \\ a_{2,1}(a_{1,1} + a_{2,2} - (\alpha+\beta)) \\ a_{2,1}a_{3,2} \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

Calcolata Me_1, individuiamo una matrice P_0 di Householder tale che $P_0(Me_1) = \gamma e_1$. Utiliziamo poi il seguente algoritmo per costruire matrici $P_1, \dots, P_{n-2}$ tali che $Z' := P_0 \dots P_{n-2}$ abbia la prima colonna uguale a quella di P_0 e che $(Z')^T A_1 Z^T$ sia in forma di Hessenberg superiore

Definizione (Algoritmo di bulge-chasing)**.** Consideriamo la matrice P_0. Poiché A_1 è in forma di Hessenberg superiore, non è restrittivo supporre che P_0 sia della forma

$$P_0 = \left(\begin{array}{c|c} \tilde{P}_0 & \\ \hline & I_{n-3} \end{array}\right)$$

Consideriamo il prodotto $P_0^T A_1 P_0$. Vale

$$P_0^T A = \left(\begin{array}{c|c} \tilde{P}_0^T & \\ \hline & I_{n-3} \end{array}\right) \left(\begin{array}{c|c} (A_1)_{1,1} & (A_1)_{1,2} \\ \hline (A_1)_{2,1} & (A_1)_{2,2} \end{array}\right) = \left(\begin{array}{ccc|c} * & * & * & \\ * & * & * & \\ + & * & * & \\ 0 & 0 & * & * \\ 0 & 0 & 0 & \\ & \vdots & & \\ \hline 0 & 0 & 0 & \end{array}\right)$$

$$P_0^T A P_0 = \left(\begin{array}{ccc|c} * & * & * & \\ * & * & * & \\ + & * & * & \\ 0 & 0 & * & * \\ 0 & 0 & 0 & \\ & \vdots & & \\ \hline 0 & 0 & 0 & \end{array}\right) \left(\begin{array}{c|c} \tilde{P}_0 & \\ \hline & I_{n-3} \end{array}\right) = \left(\begin{array}{ccc|c} * & * & * & \\ * & * & * & \\ + & * & * & \\ + & + & * & * \\ 0 & 0 & 0 & \\ & \vdots & & \\ \hline 0 & 0 & 0 & \end{array}\right)$$

dove con $*$ e $+$ indichiamo elementi non necessariamente nulli. L'unica differenza tra gli elementi indicati con $*$ e gli elementi indicati con $+$ è che gli elementi indicati con $+$ sono "in più" rispetto alla forma di Hessenberg. Questi elementi formano una "sporgenza" (bulge) che andremo adesso a spostare fino a farla scomparire, ed ottenere una matrice in forma di Hessenberg superiore.
Scegliamo ora P_1 in modo che coniugando la matrice appena ottenuta per P_1 si vadano ad annullare gli elementi in posizione $(3,1)$ e $(4,1)$ (entrambi nella "sporgenza"). P_1 sarà della forma

$$P_1 = \left(\begin{array}{c|c|c} 1 & & \\ \hline & \tilde{P}_1 & \\ \hline & & I_{n-4} \end{array}\right)$$

Svolgendo il prodotto $P_1^T (P_0^T A_1 P_0) P_1$ otteniamo, con ragionamento analogo a quello fatto in precedenza:

$$P_1^T (P_0^T A_1 P_0) P_1 = \left(\begin{array}{ccc|c} * & * & * & \\ * & * & * & \\ 0 & * & * & \\ 0 & + & * & * \\ 0 & + & + & \\ 0 & 0 & 0 & \\ & \vdots & & \\ \hline 0 & 0 & 0 & \end{array}\right)$$

Ovvero otteniamo che la "sporgenza" si è spostata di uno verso destra.
Iterando, prendendo le P_i uguali a

$$P_i = \left(\begin{array}{c|c|c} I_i & & \\ \hline & \tilde{P}_i & \\ \hline & & I_{n-3-i} \end{array}\right)$$

otteniamo, arrivando fino a P_{n-3}, la seguente matrice

$$P_{n-3}^T \dots P_0^T A_1 P_0 \dots P_{n-3} = \begin{pmatrix} * & \dots & \dots & \dots & * \\ * & \ddots & & & \vdots \\ & \ddots & \ddots & & \vdots \\ & & * & \ddots & \vdots \\ & & + & * & * \end{pmatrix}$$

e scegliamo a questo punto P_{n-2} tale che annulli l'elemento in posizione $(n, n-2)$.
La matrice $P_{n-2}^T \dots P_0^T A_1 P_0 \dots P_{n-2}$ è quindi in forma di Hessenberg superiore. Inoltre, poiché tutte le P_i, $i = 1, \dots n$ hanno come prima colonna e_1, la prima colonna di $Z' := P_0 P_1 \dots P_{n-2}$ è uguale alla prima colonna di P_0.

(Grazie ad Ivan Bioli per la seguente parte di appunti, non davvero spiegata a lezione)
Mostriamo ora che la prima colonna di Z' coincide, a meno di scalare, con la prima colonna di Z. Poiché $M = ZS$ e S è triangolare superiore, la prima colonna di Z è proporzionale alla prima colonna di M, cioè Me_1. Abbiamo scelto P_0 tale che $P_0(M_e 1) = \gamma e_1$. Poiché P_0 è di Householder vale $P_0 = P_0^{-1}$, da cui otteniamo $P_0 e_1 = \frac{1}{\gamma} M e_1$. Allora la prima colonna di Z è proporzionale alla prima colonna di P_0, che coincide con la prima colonna di Z'.

A questo punto, a meno di scalare, per il teorema del Q implicito sappiamo che esiste D tale che $DZ = Z'$, con

$$D = \begin{pmatrix} 1 & & & \\ & \pm 1 & & \\ & & \ddots & \\ & & & \pm 1 \end{pmatrix}$$

Chiamiamo ora $A_3' = Z^T A_1 Z$. Vale quindi $A_3' = D A_3 D$. In particolare segue che gli elementi diagonali di A_3' e di A_3 coincidono. Poiché per il metodo QR siamo interessati solo agli elementi diagonali, possiamo procedere ripartendo da A_3' "come se fosse" A_3.

8 14/10/2020

8.1 Metodo delle potenze

Il metodo delle potenze è un metodo iterativo che data una matrice $A \in \mathbb{C}^{n\times n}$ generica, permette di approssimare l'autovalore di modulo massimo e un relativo autovettore. Il metodo funziona in generale, ma la dimostrazione è complicata. Dimostreremo quindi la convergenza del metodo nelle ipotesi aggiuntive di A diagonalizzabile con autovettori $\lambda_1, \dots, \lambda_n$ tali che

$$|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_n|$$

e chiamiamo $x_1, \dots, x_n$ dei relativi autovettori (che formano una base di $\mathbb{C}^n$).

Scegliamo $t_0 \in \mathbb{C}^n \setminus \{0\}$. Poiché $\{x_1, \dots, x_n\}$ è una base di $\mathbb{C}^n$, esistono (e sono unici) $\alpha_1, \dots, \alpha_n$ tali che

$$t_0 = \sum_{i=1}^{n} \alpha_i x_i$$

Possiamo supporre inoltre che t_0 sia tale che $\alpha_1 \neq 0$. In generale non è lecito supporlo ($t_0 = x_2$ è un controesempio, e in generale non conoscendo la base non c'è garanzia che la prima coordinata sia diversa da 0) ma è anche vero che prendendo t_0 a caso, α_1 è diverso da 0 con probabilità 1, quindi operativamente il metodo funziona. Consideriamo ora la successione di vettori $\{y_k\}$ definita da

$$\begin{cases} y_0 = t_0 \\ y_k = Ay_{k-1} \end{cases}$$

Questa relazione di ricorrenza ci fornisce una formula generica per y_k, ovvero

$$\begin{aligned} y_k &= Ay_{k-1} = \\ &= A^2 y_{k-2} = \\ &= A^k y_0 = \\ &= A^k t_0 = \\ &= \sum_{i=1}^{n} \alpha_i A^k x_i = \\ &= \sum_{i=1}^{n} \alpha_i \lambda_i^k x_i = \\ &= \lambda_1^k \left(\alpha_1 x_1 + \sum_{i=2}^{n} \alpha_i \left(\frac{\lambda_i}{\lambda_1} \right)^k x_i \right) \end{aligned}$$

Usando la notazione $(v)_j$ per intendere la j-esima componente del vettore v, possiamo scrivere, per ogni $j = 1, \ldots, n$,

$$\frac{(y_{k+1})_j}{(y_k)_j} = \frac{\lambda_1^{k+1}\left(\alpha_1(x_1)_j + \sum_{i=2}^n \alpha_i \left(\frac{\lambda_i}{\lambda_1}\right)^{k+1}(x_i)_j\right)}{\lambda_1^k\left(\alpha_1(x_1)_j + \sum_{i=2}^n \alpha_i \left(\frac{\lambda_i}{\lambda_1}\right)^{k}(x_i)_j\right)} =$$

$$= \lambda_1 \frac{\alpha_1(x_1)_j + \sum_{i=2}^n \alpha_i \left(\frac{\lambda_i}{\lambda_1}\right)^{k+1}(x_i)_j}{\alpha_1(x_1)_j + \sum_{i=2}^n \alpha_i \left(\frac{\lambda_i}{\lambda_1}\right)^{k}(x_i)_j}$$

e poiché per ipotesi $\left|\frac{\lambda_i}{\lambda_1}\right| < 1$, il limite di $\left(\frac{\lambda_i}{\lambda_1}\right)^k$ per k che tende a $+\infty$ è 0, e quindi vale

$$\lim_{k\to+\infty} \frac{(y_{k+1})_j}{(y_k)_j} = \lambda_1$$

Inoltre, osservando la formula generica per y_k trovata prima, vale

$$\lim_{k\to+\infty} \frac{y_k}{\lambda_1^k} = \alpha_1 x_1$$

che è possibile calcolare con l'approssimazione di λ_1 ottenuta dal limite precedente.
Quindi, partendo da t_0 generico, è possibile calcolare approssimazioni di λ_1 e di un suo autovalore al costo di $O(n^2)$ per iterazione (ovvero il costo della moltiplicazione matrice-vettore Ay_{k-1}, che può essere abbattuto nel caso in cui A avesse una struttura particolare, come tridiagonale o sparsa).

Questo metodo, però, ha un problema intrinseco. Nel caso in cui λ_1 sia un autovalore con modulo diverso da 1, la dimensione delle componenti di y_k cresce o diminuisce esponenzialmente, e si può arrivare facilmente a condizioni di underflow o overflow che vanificano il conto. Possiamo quindi costruire una successione diversa, che di fatto va a normalizzare di volta in volta i vettori ottenuti, per gestire questo problema. Scelta una norma $\|\cdot\|$, definiamo questa successione come

$$\begin{cases} u_k = At_{k-1} \\ t_k = \frac{1}{\beta_k} u_k \qquad \text{con } \beta_k \text{ tale che } \|t_k\| = 1 \end{cases}$$

Scriviamo ora delle formule generali per t_k e u_k. Vale

$$\begin{aligned} t_k &= \frac{1}{\beta_k} u_k = \\ &= \frac{1}{\beta_k} A t_{k-1} = \\ &= \frac{1}{\beta_k} A \frac{1}{\beta_{k-1}} A t_{k-2} = \\ &= \frac{1}{\underbrace{\prod_{i=1}^{k} \beta_i}_{=:\gamma_k}} A^k t_0 = \\ &= \frac{1}{\gamma_k} A^k t_0 \end{aligned}$$

$$\begin{aligned} u_k &= A t_k = \\ &= \frac{1}{\gamma_k} A^{k+1} t_0 \end{aligned}$$

Ricordandoci che $A^k t_0 = y_k$, possiamo utilizzare la formula generale per y_k ricavata in precedenza per calcolare, per ogni $j = 1, \dots, n$,

$$\begin{aligned} \frac{(u_{k+1})_j}{(t_k)_j} &= \frac{\frac{1}{\gamma_k}\lambda_1^{k+1}\left(\alpha_1 (x_1)_j + \sum_{i=2}^{n} \alpha_i \left(\frac{\lambda_i}{\lambda_1}\right)^{k+1} (x_i)_j\right)}{\frac{1}{\gamma_k}\lambda_1^{k}\left(\alpha_1 (x_1)_j + \sum_{i=2}^{n} \alpha_i \left(\frac{\lambda_i}{\lambda_1}\right)^{k} (x_i)_j\right)} = \\ &= \lambda_1 \frac{\alpha_1 (x_1)_j + \sum_{i=2}^{n} \alpha_i \left(\frac{\lambda_i}{\lambda_1}\right)^{k+1} (x_i)_j}{\alpha_1 (x_1)_j + \sum_{i=2}^{n} \alpha_i \left(\frac{\lambda_i}{\lambda_1}\right)^{k} (x_i)_j} \end{aligned}$$

e quindi anche in questo caso, andando al limite, vale

$$\lim_{k \to +\infty} \frac{(u_{k+1})_j}{(t_k)_j} = \lambda_1$$

8.2 Caso norma infinito

Vediamo ora il metodo delle potenze scegliendo come norma la $\|\cdot\|_\infty$.
Per iniziare scegliamo t_0 tale che $\|t_0\|_\infty = 1$. Detto m l'indice $\left|(u_k)_m\right| = \|u_k\|_\infty$, scegliamo ora $\beta_k = (u_k)_m$, che verifica la richiesta $\|t_k\|_\infty = 1$.
Prendendo il conto svolto in precedenza per il rapporto tra $(u_{k+1})_j$ e $(t_k)_j$, leggendolo sulla componente m e ricordandoci che t_k è normalizzato (quindi

$(t_k)_m = 1$), otteniamo([5])

$$(u_{k+1})_m = \lambda_1 \left(1 + O\left(\left(\frac{\lambda_2}{\lambda_1}\right)^k\right)\right), \quad \text{per } k \to +\infty$$

e quindi anche

$$\lim_{k\to+\infty} \beta_k = \lambda_1$$

Osservazione. A priori m non è una costante ma un valore $m(k)$ che dipende da k. Vale però che([6]) per k che tende a $+\infty$ il valore di $m(k)$ è definitivamente costante, e possiamo prendere m uguale a quella costante.

(La dimostrazione del fatto che t_k tenda a x_1 normalizzato che sto per riportare è diversa da quella vista a lezione. Francamente la dimostrazione vista a lezione mi sembra falsa, quindi riporto quella del libro Metodi Numerici di Algebra Lineare)
Riprendiamo temporaneamente un limite ottenuto studiando y_k, ovvero

$$\lim_{k\to+\infty} \frac{y_k}{\lambda_1^k} = \alpha_1 x_1$$

Osserviamo che questo limite implica, per ogni $j = 1, \ldots, n$,

$$\lim_{k\to+\infty} \frac{(y_k)_j}{\lambda_1^k} = \alpha_1 (x_1)_j$$

e combinando questi due limiti otteniamo

$$\lim_{k\to+\infty} \frac{y_k}{(y_k)_j} = \frac{x_1}{(x_1)_j}$$

Torniamo a t_k. Osserviamo che $t_k = \frac{1}{\gamma_k} A^k t_0 = \frac{1}{\gamma_k} y_k$. Da questa uguaglianza si ricava che, per ogni $j = 1, \ldots, n$,

$$\frac{t_k}{(t_k)_j} = \frac{\frac{1}{\gamma_k} y_k}{(\frac{1}{\gamma_k} y_k)_j} = \frac{\frac{1}{\gamma_k} y_k}{\frac{1}{\gamma_k} (y_k)_j} = \frac{y_k}{(y_k)_j}$$

In particolare, passando al limite per k che tende a $+\infty$, e scegliendo $j = m$, otteniamo

$$\lim_{k\to+\infty} t_k = \lim_{k\to+\infty} \frac{y_k}{(y_k)_m} = \frac{x_1}{(x_1)_m}$$

dove la prima uguaglianza vale perché $(t_k)_m = 1$ per come abbiamo definito la successione.

[5] Si verifica utilizzando lo sviluppo di Taylor e facendo i conti non fatti a lezione e che non riporterò.

[6] Non è stato visto il perché a lezione, ma credo sia non troppo difficile da far vedere. E' possibile che prima o poi provi a fare una dimostrazione da aggiungere a questi appunti per completezza. Dovrebbe valere anche che l'm che si ottiene è anche l'indice della componente che realizza il modulo massimo di x_1

Quindi con la successione degli t_k e u_k è possibile quindi calcolare approssimazioni di λ_1 e di un autovettore normalizzato relativo a λ_1. Come criterio di arresto per decidere quando fermarsi con le iterazioni e accontentarsi della approssimazione ottenuta possiamo fissare un $\varepsilon > 0$ a priori e usare uno dei due seguenti criteri:

- $|\beta_{k+1} - \beta_k| < \varepsilon$
- $\left|\frac{\beta_{k+1}-\beta_k}{\beta_k}\right| < \varepsilon$

9 16/10/2020

9.1 Caso norma 2

Consideriamo ora il metodo delle potenze scegliendo come norma la $\|\cdot\|_2$.
Per iniziare, scegliamo t_0 tale che $\|t_0\|_2 = 1$. Per continuare la successione, prendiamo per ogni k $\beta_k = \|u_k\|_2$. Questa scelta è particolarmente conveniente nel caso di A normale, poiché garantisce una velocità di convergenza a λ_1 maggiore rispetto ai risultati ottenuti precedentemente. Infatti, chiamando $\sigma_k = t_k^H u_{k+1}$, e ricordandoci che t_k ha norma 2 uguale a 1 quindi $t_k^H t_k = 1$, otteniamo

$$\begin{aligned}
\sigma_k &= t_k^H u_{k+1} = \\
&= t_k^H A t_k = \\
&= \frac{t_k^H A t_k}{t_k^H t_k} = \\
&= \frac{\left(\frac{1}{\gamma_k} A^k t_0\right)^H A \left(\frac{1}{\gamma_k} A^k t_0\right)}{\left(\frac{1}{\gamma_k} A^k t_0\right)^H \left(\frac{1}{\gamma_k} A^k t_0\right)} = \\
&= \frac{\left(A^k t_0\right)^H \left(A^{k+1} t_0\right)}{\left(A^k t_0\right)^H \left(A^k t_0\right)} = \\
&= \lambda_1 \frac{|\alpha_1|^2 + \sum_{i=2}^n |\alpha_i|^2 \left|\frac{\lambda_i}{\lambda_1}\right|^{2k} \left(\frac{\lambda_i}{\lambda_1}\right)}{|\alpha_1|^2 + \sum_{i=2}^n |\alpha_i|^2 \left|\frac{\lambda_i}{\lambda_1}\right|^{2k}} = \\
&= \lambda_1 \left(1 + O\left(\left|\frac{\lambda_2}{\lambda_1}\right|^{2k}\right)\right)
\end{aligned}$$

Quindi abbiamo una convergenza con un errore di $\left|\frac{\lambda_2}{\lambda_1}\right|^{2k}$ invece di $\left(\frac{\lambda_2}{\lambda_1}\right)^k$.

Come criterio di arresto, fissata una tolleranza ε a priori, possiamo usare

$$\|u_{k+1} - \sigma_k t_k\|_2 < \varepsilon$$

Infatti, vale

$$\begin{aligned}
\|u_{k+1} - \sigma_k t_k\|_2 &= \|A t_k - \sigma_k t_k\|_2 = \\
&= \left\|(A - \sigma_k I) t_k\right\|_2 = \\
&= \frac{\left\|(A - \sigma_k I) t_k\right\|_2}{\|t_k\|_2}
\end{aligned}$$

e poiché per il teorema 6.2 del libro Metodi Numerici per l'Algebra Lineare vale

$$|\lambda_1 - \sigma_k| \leq \frac{\left\|(A - \sigma_k I) t_k\right\|_2}{\|t_k\|_2}$$

otteniamo che se vale $\|u_{k+1} - \sigma_k t_k\|_2 < \varepsilon$ allora vale anche $|\lambda_1 - \sigma_k| < \varepsilon$.

9.2 Metodo delle potenze inverse (variante di Wielandt)

Sia $A \in \mathbb{C}^{n\times n}$ matrice invertibile e diagonalizzabile e, detti $\lambda_1, \dots, \lambda_n$ i suoi autovalori, sia tale che

$$|\lambda_1| \geq \cdots \geq |\lambda_{n-1}| > |\lambda_n|$$

ovvero sia A tale che λ_n è l'unico autovalore di modulo minimo.
Vale che gli autovalori di A^{-1} sono gli inversi degli autovalori di A, e in particolare vale

$$\left|\frac{1}{\lambda_n}\right| > \left|\frac{1}{\lambda_{n-1}}\right| \geq \cdots \geq \left|\frac{1}{\lambda_1}\right|$$

Siamo quindi nelle ipotesi per poter applicare il metodo delle potenze con A^{-1}, calcolando quindi l'autovalore di modulo minimo di A.

Andiamo ora ad analizzare operativamente il metodo delle potenze con A^{-1}. La successione dei t_k, u_k è definita da

$$\begin{cases} u_k = A^{-1} t_{k-1} \\ t_k = \frac{1}{\beta_k} u_k \qquad \text{con } \beta_k \text{ tale che } \|t_k\| = 1 \end{cases}$$

Data A, però, calcolare A^{-1} può essere un'operazione molto costosa computazionalmente. Operativamente, quindi, invece di calcolare A^{-1} e poi tutti i prodotti $u_k = A^{-1}t_{k-1}$ può convenire risolvere il sistema lineare $Au_k = t_{k-1}$ ad ogni passo. In questo caso, poiché i sistemi che andremo a risolvere passo per passo avranno sempre gli stessi coefficienti, possiamo calcolare una volta all'inizio una fattorizzazione LU di A (se esiste, o una QR altrimenti) per velocizzare la risoluzione dei sistemi da risolvere ad ogni step.

Avendo un metodo per calcolare l'autovalore di modulo minimo di una matrice, è possibile ora calcolare qualsiasi autovalore di questa matrice, tramite uno shift. Il procedimento è il seguente.
Per ogni valore μ, è facile verificare che $\mathrm{Sp}(A - \mu I) = \mathrm{Sp}(A) - \mu$, ovvero che gli autovalori della matrice $(A - \mu I)$ corrispondono agli autovalori di A meno lo shift μ. Supponiamo ora che μ sia l'approssimazione grossolana di un certo autovalore λ_j di A (possiamo supporre che $\mu \neq \lambda_j$ poiché se lo fosse, non avremmo bisogno di raffinare l'approssimazione che già abbiamo). Con ciò intendiamo che λ_j è l'autovalore più vicino di tutti a μ, ovvero che

$$0 < |\lambda_j - \mu| < |\lambda_i - \mu|, \quad \forall i \neq j$$

Questo equivale a dire che $(\lambda_j - \mu)$ è l'autovalore di modulo minimo della matrice $(A - \mu I)$. Con il metodo delle potenze inverse possiamo quindi calcolare con precisione a piacere l'autovalore $(\lambda_j - \mu)$ di $(A - \mu I)$ per ottenere indirettamente il valore dell'autovalore λ_j di A.

9.3 Problema generalizzato di autovalori

Data una matrice A, abbiamo definito come suo autovalore un qualsiasi valore λ tale che la matrice $(A - \lambda I)$ fosse singolare. Generalizziamo ora questa definizione.

Definizione (Matrix pencil)**.** Siano $A, B \in \mathbb{C}^{n \times n}$. $A - \lambda B$ (con λ variabile, non un certo valore fissato) viene detto matrix pencil, o semplicemente pencil. Inoltre, detto $p(\lambda) = \det(A - \lambda B)$ il polinomio caratteristico del pencil $A - \lambda B$:

- se esiste λ tale che $p(\lambda) \neq 0$ (ovvero se $p(\lambda)$ non è la funzione identicamente nulla) il pencil si dice regolare,
- altrimenti, il pencil si dice singolare.

Definizione (Autovalore generalizzato)**.** I valori λ per cui esiste un vettore $x \neq 0$ tale che $Ax = \lambda Bx$ si dicono autovalori generalizzati (o semplicemente autovalori) del pencil $A - \lambda B$. In generale, gli autovalori generalizzati del pencil $A - \lambda B$ sono l'unione di

- le radici di $p(\lambda)$ con la rispettiva molteplicità,
- ∞ con molteplicità $n - \deg(p)$ (presente solo se $\deg(p) < n$).

Esempio. Consideriamo il seguente pencil:

$$A - \lambda B = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix} - \lambda \begin{pmatrix} 2 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$

Allora

$$\begin{aligned} p(\lambda) &= \det(A - \lambda B) = \\ &= (1 - 2\lambda)(1 - 0\lambda)(0 - 1\lambda) = \\ &= \lambda(2\lambda - 1) \end{aligned}$$

Quindi gli autovalori del pencil $A - \lambda B$ sono $\{\frac{1}{2}, 0, \infty\}$.

Esistono delle relazioni tra gli autovettori del pencil $A - \lambda B$ e le matrici A, B:

- Se B è non singolare, l'equazione può essere trasformata in $B^{-1}Ax = \lambda x$ quindi gli autovalori generalizzati del pencil $A - \lambda B$ sono gli autovalori della matrice $B^{-1}A$.
- Se A è non singolare, l'equazione può essere trasformata in $A^{-1}Bx = \frac{1}{\lambda}x$ quindi gli autovalori generalizzati del pencil $A - \lambda B$ sono i reciproci degli autovalori della matrice $A^{-1}B$, dove il reciproco di 0 è definito come ∞.

Il pencil $A - \lambda B$ ha n autovalori (considerate le molteplicità) se e soltanto se la matrice B ha rango n. In caso contrario, l'insieme $\lambda(A, B)$ degli autovalori di $A - \lambda B$ può essere vuoto, finito o infinito. Ad esempio

$$A = \begin{pmatrix} 1 & 2 \\ 0 & 2 \end{pmatrix}, \quad B = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \Rightarrow \lambda(A, B) = \{1\}$$

$$A = \begin{pmatrix} 1 & 2 \\ 0 & 2 \end{pmatrix}, \quad B = \begin{pmatrix} 1 & 0 \\ 1 & 0 \end{pmatrix} \Rightarrow \lambda(A, B) = \varnothing$$

$$A = \begin{pmatrix} 1 & 2 \\ 0 & 0 \end{pmatrix}, \quad B = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \Rightarrow \lambda(A, B) = \mathbb{C}$$

Teorema 9.1 (Forma di Schur generalizzata)**.** *Siano $A, B \in \mathbb{C}^{n \times n}$ tali che $A - \lambda B$ sia un pencil regolare.*
Allora è possibile determinare delle matrici Q_L e Q_R unitarie tali che $Q_L A Q_R = T_A$ e $Q_L B Q_R = T_B$, dove $T_A, T_B \in \mathbb{C}^{n \times n}$ sono matrici triangolari superiori. Vale inoltre che gli autovalori di $A - \lambda B$ sono i rapporti $\frac{(T_A)_{i,i}}{(T_B)_{i,i}}$ al variare di $i = 1, ..., n$.

Dimostrazione. Sia λ' un autovalore del pencil $A - \lambda B$ e x un relativo autovettore, tale che $\|x\|_2 = 1$. Poiché vale $Ax = \lambda' Bx$, i vettori Ax e Bx sono proporzionali, ed esiste quindi un vettore y con $\|y\|_2 = 1$ tale che Ax e Bx siano multipli di y. Andiamo ora a costruire iterativamente delle matrici X e Y unitarie tali che $Y^H A X$ e $Y^H B X$ siano triangolari.
Scegliamo la prima colonna di X uguale ad x (l'autovettore rispetto a λ') e scegliamo la prima colonna di Y uguale a y (il vettore "divisore comune" di Ax e Bx).
Poiché $Ax = \gamma y$ per qualche γ e poiché Y è unitaria, $Y^H A x = \gamma e_1$. Allo stesso modo, $Y^H B x = \eta e_1$ per qualche η. Allora, scrivendo $X = (x|\tilde{X})$ e $Y = (y|\tilde{Y})$, vale

$$Y^H A X = (Y^H A x | Y^H A \tilde{X}) =: \left(\begin{array}{c|c} \gamma & \tilde{a} \\ \hline 0 & A_{2,2} \\ \vdots & \end{array}\right)$$

$$Y^H B X = (Y^H B x | Y^H B \tilde{X}) =: \left(\begin{array}{c|c} \eta & \tilde{b} \\ \hline 0 & B_{2,2} \\ \vdots & \end{array}\right)$$

Iterando il procedimento considerando il pencil $A_{2,2} - \lambda B_{2,2}$ si riesce a produrre completamente le matrici X e Y, e a questo punto basta prendere $Q_L = Y^H$ e $Q_R = X$.

Per concludere la dimostrazione basta osservare che, poiché le matrici Q_L e Q_R sono unitarie e quindi invertibili, i due pencil $A - \lambda B$ e $Q_L A Q_R - \lambda Q_L B Q_R$

hanno gli stessi autovalori. Allora gli autovalori di $A - \lambda B$ sono le radici di $p(\lambda) = \det(Q_L A Q_R - \lambda Q_L B Q_R) = \det(T_A - \lambda T_B)$. Una facile verifica dimostra che se $(T_B)_{i,i} \neq 0$ allora $\lambda = \frac{(T_A)_{i,i}}{(T_B)_{i,i}}$ è una radice di $p(\lambda)$. Invece, la differenza tra n e il grado di $p(\lambda)$ è uguale al numero di elementi diagonali nulli di T_B. Quindi la molteplicità di ∞ come autovalore di $A - \lambda B$ è uguale al numero di elementi diagonali nulli di T_B e quindi anche in questo caso l'autovalore ∞ di $A - \lambda B$ è uguale a $\frac{(T_A)_{i,i}}{(T_B)_{i,i}}$ (per come abbiamo definito il reciproco([7]) di 0). □

9.4 Algoritmo QZ per la risoluzione di autovalori generalizzati

Vediamo ora l'algoritmo QZ, un algoritmo per trovare gli autovalori generalizzati del pencil $A - \lambda B$ a partire da A, B matrici generiche.
L'algoritmo punta a costruire delle matrici Q e Z unitarie tali che QAZ e QBZ siano triangolari. Per fare ciò, l'algoritmo si scompone in tre fasi

1. Trasformazione di A da generica a forma di Hessenberg e (contemporaneamente) di B da generica a triangolare superiore
2. Trasformazione di A da Hessenberg superiore a triangolare, preservando la triangolarità di B
3. Calcolo degli autovalori

Non analizzeremo l'intero algoritmo ma solo la seconda fase([8]).

Supponiamo quindi di avere A e B rispettivamente in forma di Hessenberg superiore e in forma triangolare superiore, e cerchiamo di trasformare A in forma triangolare superiore preservando la triangolarità di B. Per spiegare la generica iterazione di questa fase, "facciamo finta" che B sia non singolare e di poter quindi disporre di una matrice $C = AB^{-1}$. L'idea è quella di utilizzarla per esibire delle matrici Q e Z, per poi mostrare che è possibile ottenere tali matrici senza passare per C.

Sia quindi $C = AB^{-1}$ e applichiamo il metodo QR con shift σ alla matrice C.
Il primo passo è calcolare una fattorizzazione $Q^T R = (C - \sigma I)$. La fattorizzazione consiste quindi nel trovare una matrice Q tale che $Q(C - \sigma I) = R$ sia una matrice triangolare superiore.
Il secondo passo produce una matrice $C' = RQ^T + \sigma I$. Sostituendo, ricaviamo

[7] Non è stato chiarito a lezione ma credo che anche $\frac{0}{0}$ sia definito, nel nostro caso, come ∞, perché altrimenti non è detto che torni questo risultato del teorema

[8] Per i più interessati, l'intero algoritmo non è presente né sul Demmel, né su Metodi Numerici per l'Algebra Lineare. Potete trovarlo al paper originale del 1973: `https://www.jstor.org/stable/2156353`

che

$$\begin{aligned} C' &= RQ^T + \sigma I = \\ &= Q(C - \sigma I)Q^T + \sigma I = \\ &= QCQ^T \end{aligned}$$

da cui otteniamo che anche C' è in forma di Hessenberg superiore (C lo era poiché prodotto di una Hessenberg superiore e una triangolare superiore).
Scegliendo Z una qualsiasi matrice unitaria e ponendo $A' = QAZ$, $B' = QBZ$ otteniamo che $A'(B')^{-1} = QAZZ^TB^{-1}Q^T = QAB^{-1}Q^T = QCQ^T = C'$. Possiamo scegliere Z unitaria tale che $B' = (QB)Z$ resti triangolare superiore. Allora $A' = C'B'$ sarà in forma di Hessenberg superiore, in quanto prodotto di una Hessenberg superiore e una triangolare superiore.

Mostriamo ora che per ottenere Q e Z non è necessario passare per C.

Per determinare Q, osserviamo che dalla fattorizzazione QR di C otteniamo

$$R = Q(C - \sigma I) = Q(AB^{-1} - \sigma I) \quad \iff \quad Q(A - \sigma B) = RB =: S$$

e vale che S è triangolare superiore un quanto prodotto di triangolari superiori. Possiamo quindi scegliere Q come una matrice unitaria tale che il prodotto $Q(A - \sigma B)$ sia triangolare superiore (che è ottenibile come prodotto $Q_{n-1} \dots Q_1 =: Q$ di matrici di Householder).

Per determinare Z esprimiamola come prodotto di matrici $Z_1 \dots Z_{n-1} =: Z$. Per capire come scegliere le Z_i, definiamo $B^{(1)} = B$, $B^{(i)} = Q_i B^{(i-1)} Z_i$. A questo punto, per ogni i possiamo scegliere Z_i tale che $Q_i B^{(i-1)} Z_i$ sia triangolare superiore. Allora vale che $B' = QBZ$ è triangolare superiore per costruzione.
Non è ovvio il motivo per cui questa scelta di Z produca una $A' = QAZ$ in forma di Hessenberg superiore. Dall'uguaglianza $Q(A - \sigma B) = S$ ricaviamo che

$$QAZ = SZ + \sigma QBZ = SZ + \sigma B'$$

Per come abbiamo costruito Z e per come è definita S, SZ è in forma di Hessenberg superiore([9]), quindi $A' = QAZ$ è in forma di Hessenberg superiore in quanto somma di una Hessenberg superiore e di una triangolare superiore.

Possiamo quindi applicare questa singola iterazione più volte, con degli shift $\sigma_1, \dots, \sigma_v, \dots$ e ottenendo delle matrici A_v, B_v tali che

$$A_v = Q_v A_{v-1} Z_v, \quad B_v = Q_v B_{v-1} Z_v$$

Nel caso in cui tutte le B_v fossero invertibili, avremmo anche la successione delle $C_v = A_v(B_v)^{-1}$ e le varie iterazioni sarebbero semplicemente delle iterazioni di metodo QR con shift che partono da C e producono la successione di

[9] Non capisco particolarmente il perché di questo fatto ma a lezione non è stato argomentato e anche nel paper originale dove viene presentato l'algoritmo questa implicazione non è particolarmente spiegata

C_v. Quindi per come funziona il metodo QR, le C_v tendono ad una matrice triangolare superiore. Allora poiché $A_v = C_v B_v$, con C_v che tende a triangolare superiore e B_v triangolare superiore, anche A_v tende a triangolare superiore. In realtà questo risultato vale anche anche nel caso di B e B_v non necessariamente invertibili([10]).

9.5 Pencil definiti

Definizione (Pencil definito positivo)**.** Un pencil $A - \lambda B$ si dice deifinito positivo se vale che $A = A^T, B = B^T$ e B è definita positiva.

Osserviamo che dato un pencil $A - \lambda B$ definito positivo e una matrice X, il pencil $X^T AX - \lambda X^T BX$ è anch'esso definito positivo.

Teorema 9.2. *Sia $A - \lambda B$ un pencil definito positivo. Allora esiste X non singolare tale che*

$$X^T AX = diag(\alpha_1, \dots, \alpha_n) \quad X^T BX = diag(\beta_1, \dots, \beta_n)$$

e vale che gli autovalori di $A - \lambda B$, che sono $\frac{\alpha_i}{\beta_i}$, sono tutti reali e finiti

Dimostrazione. Sia $B = LL^T$ la fattorizzazione di Cholesky([11]), con L triangolare inferiore e non singolare.
Denotiamo con L^{-T} la trasposta dell'inversa di L (che è uguale all'inversa della trasposta). Allora $B^{-1}A$ è simile a $L^{-1}AL^{-T} =: H$, infatti

$$L^T(B^{-1}A)L^{-T} = L^T L^{-T} L^{-1} AL^{-T} = H$$

Poiché anche H è simmetrica (infatti $H^T = (L^{-1}AL^{-T})^T = L^{-1}A^T L^{-T} = H$), esiste una fattorizzazione $H = Q\Lambda Q^T$ con Q unitaria e Λ diagonale.
Allora, prendendo $X = L^{-T}Q$, otteniamo che

$$X^T AX = Q^T \underbrace{L^{-1}AL^{-T}}_{=H} Q = \Lambda$$

$$X^T BX = Q^T L^{-1} BL^{-T} Q = Q^T L^{-1}(LL^T)L^{-T}Q = I$$

□

[10] Come per la nota precedente, non capisco benissimo il perché, ma non è stato spiegato esplicitamente a lezione e non è spiegato esplicitamente neanche nel paper originale

[11] Il nome di questa fattorizzazione non è stato detto in classe. Lo riporto solo per spiegare perché possiamo assumere che tale fattorizzazione esista

10 21/10/2020

10.1 Problema dei minimi quadrati

Concluso l'argomento degli autovalori, cominciamo una nuova parte del programma.
Consideriamo il sistema lineare $Ax = b$, con $A \in \mathbb{C}^{m\times n}, m \geq n$ e $x \in \mathbb{C}^n, b \in \mathbb{C}^m$. In generale sappiamo che questo sistema ammette soluzione se e solo se b appartiene all'immagine di A, cioè se e solo se

$$b \in S(A) := \{y \in \mathbb{C}^m \,|\, \exists x \in \mathbb{C}^n \text{ t.c. } y = Ax\}$$

In generale, però, non è detto che la soluzione esatta esista. Cerchiamo quindi di risolvere un problema più generale: trovare x tale che, per una certa norma $\|\cdot\|$ data,

$$\|Ax - b\| = \min_{y\in\mathbb{C}^n} \|Ay - b\| =: \gamma$$

Dato un qualsiasi x, il vettore $b - Ax$ sarà detto "residuo" e sarà indicato con $r := b - Ax$.
Nel caso in cui la norma considerata sia la norma 2, il problema viene detto problema dei minimi quadrati.

10.2 Sistema delle equazioni normali

Per cercare delle strategia risolutive per il problema dei minimi quadrati, consideriamo il sottospazio ortogonale all'immagine di A, definito come

$$S(A)^\perp := \{z \in \mathbb{C}^m \,|\, \forall y \in S(A),\, z^H y = 0\}$$

Per quanto sappiamo dal corso di Geometria 1, possiamo scomporre $\mathbb{C}^m$ in somma diretta di $S(A)$ e $S(A)^\perp$. In particolare, sappiamo che dato il vettore b, esistono e sono unici $b_1, b_2 \in C^m$ tali che

$$b = b_1 + b_2, \quad b_1 \in S(A), \quad b_2 \in S(A)^\perp$$

In questo caso, per come è stato definito il residuo, possiamo scrivere $r = b_1 - Ax + b_2$, e definiamo $y := b_1 - Ax \in S(A)$. Da questa riscrittura, analizzando la norma 2 di r (che è ciò che stiamo cercando di minimizzare), otteniamo

$$\begin{aligned}\|r\|_2^2 &= (y + b_2)^H (y + b_2) = \\ &= y^H y + \underbrace{b_2^H y}_{=0} + \underbrace{y^H b_2}_{=0} + b_2^H b_2 = \\ &= \|y\|_2^2 + \|b_2\|_2^2\end{aligned}$$

dove $b_2^H y = y^H b_2 = 0$ poiché $b_2 \in S(A)^\perp$ e $y \in S(A)$.
Poiché il termine $\|b_2\|_2^2$ è costante e non dipende da x, r ha norma minima se e solo se $\|y\|_2 = 0$, ovvero se e solo se $y = 0$ (per proprietà delle norme), ovvero

se e solo se $Ax = b_1$ per come abbiamo definito y.
Per come abbiamo definito r, $y = 0$ se e solo se $r = b_2 \in S(A)^{\perp}$, cioè $A^H r = 0$.
Ma per come abbiamo definito r, questo equivale a $A^H r = A^H(b - Ax) = 0$.
Quindi x è soluzione al problema dei minimi quadrati se e solo se $A^H Ax = A^H b$.
Questo nuovo sistema si chiama sistema delle equazioni normali.

10.3 Metodi risolutivi per i minimi quadrati

Vediamo ora alcuni metodi risolutivi per il problema dei minimi quadrati.
Supponiamo che A abbia rango massimo (ovvero n). In questo caso la matrice $A^H A$ è hermitiana e definita positiva, quindi esiste la fattorizzazione di Cholesky $A^H A = LL^H$ con L triangolare inferiore con diagonale positiva.
A questo punto possiamo risolvere il sistema $A^H Ax = A^H b$ con due sistemi "a cascata", ovvero risolvendo prima $Ly = A^H b$ e poi $L^H x = y$, che sono due sistemi facili da risolvere poiché L e L^H sono matrici triangolari.
Questo metodo risolutivo però può portare a difficoltà numeriche. In particolare:

- La matrice $A^H A$ potrebbe essere mal condizionata, in un senso che spiegheremo meglio in futuro
- La matrice $A^H A$ potrebbe non risultare definita positiva per problemi di limitatezza di precisione di macchina. Ad esempio, sia *alpha* un valore reale tale che $\alpha^2 < u \leq \alpha < 1$, dove u indica la precisione di macchina. Consideriamo allora la matrice

  $$A = \begin{pmatrix} 1 & 1 \\ \alpha & 0 \\ 0 & \alpha \end{pmatrix}$$

  Allora andando a calcolare $A^H A$ (che coincide con $A^T A$ perché in questo esempio stiamo lavorando con una matrice reale) otteniamo

  $$A^T A = \begin{pmatrix} 1 & \alpha & 0 \\ 1 & 0 & \alpha \end{pmatrix} \begin{pmatrix} 1 & 1 \\ \alpha & 0 \\ 0 & \alpha \end{pmatrix} = \begin{pmatrix} 1+\alpha^2 & 1 \\ 1 & 1+\alpha^2 \end{pmatrix}$$

  Poiché però $\alpha^2 < u$, a livello computazionale abbiamo che

  $$A^T A = \begin{pmatrix} 1 & 1 \\ 1 & 1 \end{pmatrix}$$

  che non è una matrice definita positiva.

Abbiamo bisogno quindi di altri metodi che non cadano in problemi computazionali.

E' possibile utilizzare la fattorizzazione QR per risolvere il problema dei minimi quadrati. Questo metodo prende il nome di metodo QR per la risoluzione

dei minimi quadrati.
Sia $A = QR$ una fattorizzazione QR di A, con $Q \in \mathbb{C}^{m\times m}$ unitaria e $R \in \mathbb{C}^{m\times n}$ della forma

$$R = \begin{pmatrix} R_1 \\ \hline 0 \end{pmatrix}$$

con $R_1 \in \mathbb{C}^{n\times n}$ triangolare superiore. Sempre nell'ipotesi in cui A abbia rango massimo vale che R_1 è non singolare. Consideriamo ora la norma di r che stiamo cercando di minimizzare.

$$\begin{aligned} \|Ax - b\|_2 &= \|QRx - b\|_2 = \\ &= \left\|Q(Rx - Q^H b)\right\|_2 = \\ &= \left\|Rx - Q^H b\right\|_2 \end{aligned}$$

dove l'ultimo passaggio è giustificato dal fatto che le matrici unitarie preservano la norma 2. Definiamo ora $c := Q^H b$ e scomponiamo il vettore c in $c^t = (c_1|c_2)^T$ dove c_1 è il vettore composto dalle prime n componenti di c, e c_2 sono le rimanenti $(m-n)$ componenti. A questo punto possiamo scomporre $Rx - c$ come

$$Rx - c = \begin{pmatrix} R_1 x - c_1 \\ -c_2 \end{pmatrix}$$

e vale quindi, riprendendo lo sviluppo fatto prima,

$$\begin{aligned} \|Ax - b\|_2^2 &= \|Rx - c\|_2^2 = \\ &= \|R_1 x - c_1\|_2^2 + \|-c_2\|_2^2 \end{aligned}$$

Minimizzare questa norma vuol dire quindi cercare x tale che $R_1 x = c_1$, che ha soluzione ed è unica poiché R_1 è non singolare.
Per trovare la soluzione al problema dei minimi quadrati, quindi, dobbiamo prima calcolare una fattorizzazione QR di A, calcolare il vettore $c = Q^H b$ e risolvere il sistema $R_1 x = c_1$ dove R_1 è immediatamente ottenibile da R e c_1 è immediatamente ottenibile da c.

11 23/10/2020

11.1 Decomposizione ai valori singolari (SVD)

Vediamo ora un altro metodo di risoluzione del problema dei minimi quadrati, applicabile anche nel caso di A non di rango massimo.
Dobbiamo prima introdurre la definizione di decomposizione ai valori singolari (SVD, per Singular Value Decomposition in inglese).

Teorema 11.1 (Decomposizione ai valori singolari, senza dimostrazione)**.** *Sia $A \in \mathbb{C}^{m\times n}$.*
Allora esistono matrici

- $U \in \mathbb{C}^{m\times m}, V \in \mathbb{C}^{n\times n}$ *unitarie,*
- $\Sigma = (\sigma_{i,j}) \in \mathbb{R}^{m\times n}$ *con* $\sigma_{i,j} = 0$ *se* $i \neq j$ *e* $\sigma_{1,1} \geq \cdots \geq \sigma_{p,p} \geq 0$ *dove* $p = min\{m, n\}$

tali che $A = U\Sigma V^H$. Questa decomposizione viene detta decomposizione ai valori singolari di A.

I valori $\sigma_{i,i}$ (alle volte indicati con σ_i per semplificare la notazione) sono detti valori singolari di A. Le colonne di $U = (u_1, \ldots, u_m)$ sono dette vettori singolari sinistri e le colonne di $V = (v_1, \ldots, v_n)$ sono dette vettori singolari destri di A.
Vediamo ora alcune proprietà delle SVD, nel seguente teorema

Teorema 11.2. *Sia $A \in \mathbb{C}^{m\times n}$ e sia $A = U\Sigma V^H$ una sua SVD, con Σ tale che*

$$\sigma_1 \geq \cdots \geq \sigma_k > \sigma_{k+1} = \cdots = \sigma_p = 0, \quad p = \min\{m, n\}$$

Ovvero sia A con esattamente k valori singolari non nulli.
Allora valgono le seguenti proprietà:

1. *Dette $U_k = (u_1, \ldots, u_k) \in \mathbb{C}^{m\times k}$ e $V_k = (v_1, \ldots, v_k) \in \mathbb{C}^{n\times k}$ le restrizioni alle prime k colonne di U, V e detta $\Sigma_k = diag(\sigma_1, \ldots, \sigma_k) \in \mathbb{R}^{k\times k}$, vale*

$$A = U_k\Sigma_k V_k^H = \sum_{i=1}^{k} \sigma_i u_i v_i^H$$

2. *Il nucleo di A è generato dai vettori $v_{k+1}, \ldots, v_n$*
3. *L'immagine di A è generata dai vettori $u_1, \ldots, u_k$*
4. *σ_i^2 sono autovalori di A^HA. Inoltre, se $m < n$, gli autovalori rimanenti sono nulli. In generale, poiché per quanto visto ad Analisi Numerica sappiamo che $\|A\|_2 = \sqrt{\rho(A^HA)}$, vale $\|A\|_2 = \sigma_1$*

Dimostrazione. Consideriamo il caso $p = n \leq m$ (la dimostrazione nel caso $n > m$ è analoga e basta lavorare con A^H invece di A).

1. Decomponiamo le matrici U, Σ e V come

$$\Sigma = \left(\begin{array}{c|c} \Sigma_k & 0 \\ \hline 0 & 0 \end{array}\right)$$
$$U := \left(U_k \mid U'_{m-k}\right)$$
$$V := \left(V_k \mid V'_{n-k}\right)$$

Allora possiamo riscrivere il prodotto $A = U\Sigma V^H$ come

$$\begin{aligned} A = U\Sigma V^H = \\ &= \left(U_k \mid U'_{m-k}\right) \left(\begin{array}{c|c} \Sigma_k & 0 \\ \hline 0 & 0 \end{array}\right) \left(\begin{array}{c} V_k^H \\ \hline (V'_{n-k})^H \end{array}\right) = \\ &= \left(U_k\Sigma_k \mid 0\right) \left(\begin{array}{c} V_k^H \\ \hline (V'_{n-k})^H \end{array}\right) = \\ &= U_k\Sigma_k V_k^H \end{aligned}$$

2. Il nucleo di A è definito come $\ker(A) := \{x \in \mathbb{C}^n | Ax = 0\}$. Allora

$$\begin{aligned} Ax &= 0 \\ &\Updownarrow \\ U\Sigma V^H x &= 0 \\ &\Updownarrow \\ \Sigma V^H x &= 0 \\ &\Updownarrow \\ \left(\begin{array}{c|c} \Sigma_k & 0 \\ \hline 0 & 0 \end{array}\right) \left(\begin{array}{c} V_k^H \\ \hline (V'_{n-k})^H \end{array}\right) x &= 0 \\ &\Updownarrow \\ \left(\begin{array}{c} \Sigma_k V_k^H \\ \hline 0 \end{array}\right) x &= 0 \\ &\Updownarrow \\ \Sigma_k V_k^H x &= 0 \\ &\Updownarrow \\ V_k^H x &= 0 \end{aligned}$$

dove abbiamo utilizzato che U e Σ_k fossero invertibili per eliminarle dalle equazioni.
Abbiamo quindi che $\ker(A) = \{x \in \mathbb{C}^n | V_k^H x = 0\}$ che per definizione è $\text{Span}(v_1, \ldots, v_k)^\perp$.

Poiché $V = (V_k|V'_{n-k})$ è unitaria, allora $\text{Span}(v_1, \ldots, v_k)^\perp = \text{Span}(v_{k+1}, \ldots, v_n)$. Quindi il nucleo di A coincide con il generato dei vettori $v_{k+1}, \ldots, v_n$.

3. L'immagine di A è definita come $\{y \in \mathbb{C}^m | \exists x \in \mathbb{C}^n, Ax = y\}$.
   Sia y nell'immagine, ovvero sia $y = Ax$ per un certo x.
   Per quanto visto prima, sappiamo che $y = U_k \Sigma_k V_k^H x$.
   Sicuramente, per ogni x, $y \in \text{Span}(u_1, \ldots, u_k)$, quindi $\text{Span}(u_1, \ldots, u_k) \subset \{y \in \mathbb{C}^m | \exists x \in \mathbb{C}^n, Ax = y\}$.
   Poiché Σ_k è invertibile e V_k^H ha rango massimo, anche il prodotto $\Sigma_k V_k^H$ ha rango massimo. In particolare, l'immagine di $\Sigma_k V_k^H$ è tutto $\mathbb{C}^k$. Quindi per ogni $z \in \mathbb{C}^k$ esiste x tale che $z = \Sigma_k V_k^H x$.
   Per ogni $y \in \text{Span}(u_1, \ldots, u_k)$, per definizione, esiste $z \in \mathbb{C}^k$ tale che $y = U_k z$. Allora per ogni $y \in \text{Span}(u_1, \ldots, u_k)$ esiste x tale che $y = Ax$, quindi l'immagine di A coincide con il generato delle prime k colonne di U.

4. Sappiamo che $A = U\Sigma V^H$. Allora

   $$A^H A = V \Sigma^H U^H U \Sigma V^H = V \Sigma^H \Sigma V^H$$

   Vale poi che $\Sigma^H \Sigma = \text{diag}(\sigma_1^2, \ldots, \sigma_n^2) \in \mathbb{R}^{n \times n}$. Quindi $V \Sigma^H \Sigma V^H$ è una decomposizione spettrale di $A^H A$, per cui lo spettro di $A^H A$ è $\{\sigma_1^2, \ldots, \sigma_n^2\}$ e poiché sono ordinati in ordine non crescente, $\rho(A^H A) = \sigma_1^2$, da cui la tesi.

□

Ora che abbiamo definito le SVD e alcune proprietà, vediamo un algoritmo per produrre tali decomposizioni. Consideriamo il seguente algoritmo, diviso in due passi, che prende in input una matrice $A \in \mathbb{C}^{m \times n}$ con $m \geq n$.

- Si considera il prodotto $A^H A$ e si calcola una decomposizione spettrale $A^H A = QDQ^H$ con D diagonale contenente autovalori di $A^H A$ e Q le cui colonne sono autovettori di $A^H A$.

- Si considera $C = AQ \in \mathbb{C}^{m \times m}$ e si usa la tecnica del massimo pivot totale (vista ad Analisi Numerica) per calcolare una fattorizzazione QR della forma

  $$C\Pi = UR$$

  con Π matrice di permutazione, U unitaria e $R^T = (R_1|0)$ con R_1 quadrata triangolare superiore con elementi sulla diagonale reali, non negativi e ordinati in ordine non crescente.
  In questo caso vale $A = CQ^H = C\Pi\Pi^T Q^H = UR\Pi^T Q^H$. Allora

  $$\begin{aligned} A^H A = Q\Pi R^H U^H U R \Pi^T Q^H = \\ = Q\Pi R_1^H R_1 \Pi^T Q^H \end{aligned}$$

cioè $R_1^H R_1 = \Pi^T D \Pi$. Allora $R_1^H R_1$ è diagonale, e deve quindi necessariamente valere che R_1 stessa è diagonale([12]). Allora R è nella stessa forma di Σ in una SVD, e poiché se Q è unitaria anche $\Pi^T Q^H$ è unitaria, la decomposizione $A = U \underbrace{R}_{\Sigma} \underbrace{\Pi^T Q^H}_{V^H}$ è una decomposizione ai valori singolari di A

Disponiamo quindi adesso di un algoritmo per calcolare SVD di A. Il problema di questo algoritmo è che, per come l'abbiamo esposto, può essere molto costoso computazionalmente. Vediamo ora un modo per calcolare D e Q senza conoscere esplicitamente $A^H A$.
L'idea di base è cercare due matrici P e H unitarie tali che

$$PAH = \left(\begin{array}{c} B \\ \hline 0 \end{array}\right)$$

con $B \in \mathbb{R}^{n \times n}$ bidiagonale superiore.
In questo caso, infatti, varrebbe $A^H A = H B^T B H^H$ con $B^T B$ tridiagonale simmetrica, e conoscere una decomposizione spettrale di $B^T B$ (molto più facile da calcolare, ad esempio col metodo QR per gli autovalori) corrisponde a conoscere una decomposizione spettrale di $A^H A$.

Vediamo quindi un algoritmo per trovare le matrici P e H. Siano

- $A^{(1)} := A$
- $P^{(1)}$ matrice di Householder tale che

$$A^{(2)} := P^{(1)} A^{(1)} = \left(\begin{array}{c|c} \alpha & c^H \\ \hline 0 & B^{(2)} \\ & \end{array}\right), \quad c \in \mathbb{C}^{n-1}$$

- $K^{(1)} \in \mathbb{C}^{(n-1)\times(n-1)}$ tale che

$$K^{(1)} c = \begin{pmatrix} * \\ 0 \\ \vdots \\ 0 \end{pmatrix}$$

- $H^{(1)} \in \mathbb{C}^{n \times n}$ definita come

$$H^{(1)} := \left(\begin{array}{c|c} 1 & \\ \hline & K^{(1)} \\ & \end{array}\right)$$

[12] La dimostrazione non è stata vista in classe. Per completezza, la riporto nell'appendice.

In questo caso vale che

$$A^{(3)} := A^{(2)}H^{(1)} = \left(\begin{array}{c|cccc} * & * & 0 & \dots & 0 \\ \hline 0 & & & & \\ \vdots & & * & & \\ 0 & & & & \end{array}\right)$$

Iterando e andando a lavorare sulle sottomatrici delle $A^{(i)}$ riusciamo a ridurre A in forma bidiagonale, e le matrici P e H che stavamo cercando sono $P := P^{(n-1)} \dots P^{(1)}$ e $H := H^{(1)} \dots H^{(n-2)}$.
Il costo computazionale dell'algoritmo per calcolare la SVD di A, per come lo abbiamo espresso adesso, è di $2mn^2 - \frac{2}{3}n^3$. In un articolo del 1983, però, è stato proposto un metodo più efficiente che calcola la fattorizzazione QR di B^TB senza bisogno di conoscere esplicitamente la matrice B^TB.

11.2 SVD per il problema dei minimi quadrati

Vediamo ora come è possibile utilizzare la SVD per risolvere il problema dei minimi quadrati.

Teorema 11.3. *Sia $A \in \mathbb{C}^{m\times n}$ di rango k con $m \geq n \geq k$ e sia $A = U\Sigma V^H$ una sua SVD.*
Allora il problema dei minimi quadrati di $Ax = b$ ha come soluzione

$$x^* = \sum_{i=1}^{k} \frac{u_i^H b}{\sigma_i} v_i$$

e vale che

$$\gamma^2 := \left(\min_{y\in\mathbb{C}^n} \|Ay - b\|\right)^2 = \sum_{i=k+1}^{n} \left|u_i^H b\right|^2$$

Dimostrazione. Consideriamo $\|Ax - b\|_2$. Poiché U è una matrice unitaria, allora vale

$$\begin{aligned} \|Ax - b\|_2 &= \left\|U^H(Ax - b)\right\|_2 = \\ &= \left\|U^H(AVV^Hx - b)\right\|_2 = \\ &= \left\|\Sigma V^H x - U^H b\right\|_2 = \end{aligned}$$

e detto $y = \Sigma V^H x$, vale

$$\begin{aligned} \|Ax - b\|_2 &= \left\|\Sigma V^H x - U^H b\right\|_2 = \\ &= \sum_{i=1}^{n} \left|\sigma_i(y)_i - u_i^H b\right|^2 + \sum_{i=n+1}^{m} \left|u_i^H b\right|^2 = \\ &= \sum_{i=1}^{k} \left|\sigma_i(y)_i - u_i^H b\right|^2 + \sum_{i=k+1}^{m} \left|u_i^H b\right|^2 = \end{aligned}$$

dove l'ultimo passaggio segue dal fatto che $\sigma_{k+1} = \cdots = \sigma_n = 0$, per quanto visto sui valori singolari e sull'immagine di A.
Minimizzare $\|Ax - b\|_2$ vuol dire quindi minimizzare la prima sommatoria, e tale minimo si ha nel caso in cui tutti gli addendi della prima sommatoria siano nulli. Quindi si ha norma minima per y^* definito come

$$(y^*)_i := \begin{cases} \frac{u_i^H b}{\sigma_i} & \text{per } i = 1, \dots, k \\ 0 & \text{altrimenti} \end{cases}$$

La soluzione ottima per il problema dei minimi quadrati si ha quindi per x^* tale che $y^* = V^H x^*$.
Allora

$$x^* = Vy = \sum_{i=1}^{n} (y^*)_i v_i = \sum_{i=1}^{k} \frac{u_i^H b}{\sigma_i} v_i$$

Inoltre

$$\gamma^2 := \left(\|Ax^* - b\|\right)^2 = \sum_{i=k+1}^{n} \left| u_i^H b \right|^2$$

□

11.3 Pseudo inversa di Moore-Penrose e numero di condizionamento

Vediamo brevemente un nuovo metodo per risolvere il problema dei minimi quadrati.

Definizione (Pseudo inversa di Moore-Penrose)**.** Data $A \in \mathbb{C}^{m \times n}$ di rango k e $A = U\Sigma V^H$, viene detta pseudo inversa di Moore-Penrose di A la matrice $A^+ \in \mathbb{C}^{n \times m}$ definita come $A^+ := V\Sigma^+ U^H$, dove $\Sigma^+ \in \mathbb{R}^{n \times m}$ con

$$\sigma^+_{i,j} \begin{cases} 0 & \text{se } i \neq j \\ \frac{1}{\sigma_i} & \text{se } i = j = 1, \dots, k \\ 0 & \text{se } i = j = k+1, \dots, p \end{cases}$$

Osservazione. se A è quadrata e non singolare, allora $A^+ = A^{-1}$

Dato un generico problema dei minimi quadrati $Ax = b$, vale che la soluzione ottima si ha per $x* = A^+ b$ (segue dal teorema 11.3, non è stato visto a lezione come).
Utilizzando le pseudo inverse di Moore-Penrose è possibile estendere il concetto di numero di condizionamento (visto ad Analisi Numerica) anche a matrici rettangolari (e non invertibili).

Definizione (Numero di condizionamento)**.** Data una matrice $A \in \mathbb{C}^{m\times n}$ di rango k non necessariamente massimo. Si definisce il numero di condizionamento di A come

$$\mu(A) = \|A\| \left\|A^+\right\|$$

con $\|\cdot\|$ una norma matriciale.

Per quanto visto, $\|A\|_2 = \sigma_1$. Poiché la definizione di A^+ consiste in una sua SVD sappiamo che i suoi valori singolari sono $\frac{1}{\sigma_i}$, quindi $\left\|A^+\right\|_2 = \frac{1}{\sigma_k}$ che è il massimo dei suoi valori singolari. In particolare, otteniamo che $\mu(A) = \frac{\sigma_1}{\sigma_k}$.
Sappiamo che gli autovalori di A^HA, e quindi i valori singolari(fra poco vedremo il perché), sono σ_i^2. I valori singolari non nulli di $(A^HA)^+$ sono quindi $\frac{1}{\sigma_i^2}$. In particolare, otteniamo che $\mu(A^HA) = \frac{\sigma_1^2}{\sigma_k^2} = (\mu(A))^2$. Un'importante conseguenza di ciò è che, poiché il condizionamento aumenta quadraticamente, anche se A è ben condizionata non è detto che A^HA lo sia.

Facciamo ora delle considerazioni sugli autovalori e valori singolari di A^HA e di AA^H.
Cominciamo osservando che, data $A \in \mathbb{C}^{n\times n}$ normale con $A = U\mathrm{diag}(\lambda_1, \dots \lambda_n)U^H$ (decomposizione che esiste per il teorema 2.28 del libro Metodi Numerici per l'Algebra Lineare), possiamo scomporre $D = |D|\,\hat{D}$ con $|D|$ e $\hat{D}$ diagonali dove, elemento per elementi di D, la prima contiene il modulo dell'elemento e la seconda la rotazione, immaginando di scomporre tutti i λ_j come $\lambda_j = \left|\lambda_j\right| e^{i\theta_j}$.
Fatta questa scomposizione, vale

$$A = U\underbrace{|D|}_{\Sigma}\underbrace{\hat{D}U^H}_{V^H}$$

che è una SVD. Quindi i valori singolari coincidono con i moduli degli autovalori.

Torniamo ora a $A \in \mathbb{C}^{m\times n}$ $(m \geq n)$ e consideriamo $A^HA \in \mathbb{C}^{n\times n}$ e $AA^H \in \mathbb{C}^{m\times m}$.
Sia $A = U\Sigma V^H$. Allora $A^HA = V\Sigma^T\Sigma V^H$ e $AA^H = U\Sigma\Sigma^TU^H$. Vale che

$$\Sigma^T\Sigma = \begin{pmatrix} \sigma_1^2 & & \\ & \ddots & \\ & & \sigma_n^2 \end{pmatrix} \in \mathbb{R}^{n\times n}$$

$$\Sigma\Sigma^T = \begin{pmatrix} \sigma_1^2 & & & & & \\ & \ddots & & & & \\ & & \sigma_n^2 & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{pmatrix} \in \mathbb{R}^{m\times m}$$

Allora A^HA e AA^H hanno come autovalori i valori singolari, ma nel secondo caso ci sono anche autovalori nulli.

12 28/10/2020

12.1 Numero di condizionamento per matrici non normali

Abbiamo visto la relazione tra gli autovalori e i valori singolari di A nel caso di $A \in \mathbb{C}^{n\times n}$ normale. Consideriamo ora il caso generico. Vale il seguente teorema

Teorema 12.1. *Data $A \in \mathbb{C}^{n\times n}$ generica, e detti $\sigma_1 \geq \cdots \geq \sigma_n$ i suoi valori singolari, per ogni λ autovalore di A vale*

$$\sigma_n \leq |\lambda| \leq \sigma_1$$

Dimostrazione. Sia λ autovalore di A. Sappiamo quindi che esiste $x \neq 0$ tale che $Ax = \lambda x$. Allora vale

$$x^H(A^HA)x = (x^HA^H)(Ax) = (\bar{\lambda}x^H)(\lambda x) = |\lambda|^2\|x\|_2^2 \tag{6}$$

Tenendo a mente questo risultato, consideriamo la decomposizione ai valori singolari di $A = U\Sigma V^H$. Da questa decomposizione otteniamo $A^HA = V\Sigma^2V^H$. In particolare, andando a sostituire nel risultato appena ottenuto, vale

$$\begin{aligned}
|\lambda|^2\|x\|_2^2 &= x^H(A^HA)x = \\
&= x^H(V\Sigma^2V^H)x = \\
&= (x^HV)\Sigma^2(V^Hx) = \\
&=: y^H\Sigma^2y = \\
&= \sum_{i=1}^{n}\sigma_i^2|y_i|^2
\end{aligned}$$

con $y := V^Hx$. Poiché i valori singolari sono ordinati decrescente, valgono le seguenti stime

$$\begin{aligned}
|\lambda|^2\|x\|_2^2 &= \sum_{i=1}^{n}\sigma_i^2|y_i|^2 \leq \\
&\leq \sigma_1^2\sum_{i=1}^{n}|y_i|^2 = \\
&= \sigma_1^2\|y\|_2^2 \\
|\lambda|^2\|x\|_2^2 &= \sum_{i=1}^{n}\sigma_i^2|y_i|^2 \geq \\
&\geq \sigma_n^2\sum_{i=1}^{n}|y_i|^2 = \\
&= \sigma_n^2\|y\|_2^2
\end{aligned}$$

Inoltre, poiché y è ottenuto da x tramite una trasformazione unitaria, vale $\|x\|_2 = \|y\|_2$. Con le stime di prima otteniamo ora

$$\sigma_n^2 \|x\|_2^2 \leq |\lambda|^2 \|x\|_2^2 \leq \sigma_1^2 \|x\|_2^2$$

Poiché $x \neq 0$ possiamo dividere per $\|x\|_2^2$, e facendo la radice quadrata si ottiene la tesi. □

Quindi, tornando allo studio del numero di condizionamento, vale sempre in generale $\mu_2(A) = \frac{\sigma_1}{\sigma_n} \geq \frac{|\lambda_1|}{|\lambda_n|}$, e se A è normale vale l'uguale. Osserviamo però che anche se il rapporto tra modulo massimo e minimo degli autovalori di una matrice ci permetta di fornire una stima del numero di condizionamento di essa, nulla ci garantisce che questa stima sia buona. E' possibile che A sia mal condizionata anche se $\frac{|\lambda_1|}{|\lambda_n|}$ è piccolo.

12.2 Problema di diminuzione del rango

Vediamo ora come la SVD permetta di risolvere il seguente problema di minimo: data una matrice $A \in \mathbb{C}^{m \times n}$ di rango k e dato $r < k$, vogliamo trovare la matrice B di rango r più "vicina" ad A, ovvero tale che minimizzi la norma 2 della matrice $(A - B)$.

Teorema 12.2. *Sia $A \in \mathbb{C}^{m \times n}$ di rango k e sia $A = U\Sigma V^H$ con*

$$\sigma_1 \geq \cdots \geq \sigma_k > \sigma_{k+1} = \cdots = \sigma_p = 0, \qquad \text{con } p = \min\{m, n\}$$

Sia $r < k$ un intero positivo.
Indichiamo con $A_r := \sum_{i=1}^{r} \sigma_i u_i v_i^H$ e $S := \{B \in \mathbb{C}^{m \times n} \text{ t.c. } rnk(B) = r\}$.
Allora vale

$$\min_{B \in S} \|A - B\|_2 = \|A - A_r\|_2 = \sigma_{r+1}$$

Dimostrazione. Definiamo la seguente matrice

$$\Sigma_r := \begin{pmatrix} \sigma_1 & & & & & \\ & \ddots & & & & \\ & & \sigma_r & & & \\ & & & 0 & & \\ & & & & \ddots & \\ & & & & & 0 \end{pmatrix} \in \mathbb{R}^{m \times n}$$

Si verifica facilmente che $A_r = U\Sigma_r V^H$, da cui si ricava anche che $\text{rnk}(A_r) = r$. Poiché U e V sono unitarie e poiché $\|M\|_2 = \sqrt{\rho(M^H M)}$, si ha che

$$\|A - A_r\|_2 = \left\| U^H (A - A_r) V \right\|_2 = \|\Sigma - \Sigma_r\|_2 = \sigma_{r+1}$$

Mostriamo ora che $A_r \in S$ realizza la norma minima tra le matrici in S.
Sia $B \in S$ e indichiamo il suo nucleo con $N(B)$. Vale $\dim N(B) = n - r$. Consideriamo ora T il sottospazio vettoriale di $\mathbb{C}^n$ generato dai vettori $v_1, \ldots, v_r, v_{r+1}$ di dimensione $r+1$. Poiché vale $\dim N(B) + \dim T > n$, deve esistere $z \neq 0$ tale che $z \in N(B) \cap T$. Senza perdita di generalità possiamo prendere z tale che $\|z\|_2 = 1$.
Poiché $z \in N(B)$, ovvero $Bz = 0$, vale

$$\begin{aligned}\|Az\|_2 &= \|Az - Bz\|_2 = \\ &= \big\|(A-B)z\big\|_2 \leq \\ &\leq \|A-B\|_2 \|z\|_2 = \\ &= \|A-B\|_2\end{aligned}$$

Quindi, poiché $\|A-B\|_2 \geq \|Az\|_2$, per concludere ci basta mostrare che $\|Az\|_2 \geq \sigma_{r+1}$.
Poiché $z \in T$, vale $z = \sum_{j=1}^{r+1} \alpha_j v_j$. Allora $v_i^H z = 0$ per $i > r+1$, quindi possiamo scrivere

$$\begin{aligned}Az &= \left(\sum_{i=1}^{k} \sigma_i u_i v_i^H \right) z = \\ &= \sum_{i=1}^{k} \sigma_i u_i (v_i^H z) = \\ &= \sum_{i=1}^{r+1} \sigma_i u_i (v_i^H z)\end{aligned}$$

ricordando che $r+1 \leq k$. Possiamo allora scrivere

$$
\begin{aligned}
\|Az\|_2^2 &= \left(\sum_{i=1}^{r+1}\sigma_i u_i v_i^H z\right)^H \left(\sum_{i=1}^{r+1}\sigma_i u_i v_i^H z\right) = \\
&= \left(\sum_{i=1}^{r+1}\sigma_i z^H v_i u_i^H\right) \left(\sum_{i=1}^{r+1}\sigma_i u_i v_i^H z\right) = \\
&= \sum_{i=1}^{r+1}\sum_{j=1}^{r+1}\sigma_i\sigma_j (z^H v_i u_i^H)(u_j v_j^H z) = \\
&= \sum_{i=1}^{r+1}\sum_{j=1}^{r+1}\sigma_i\sigma_j z^H v_i (u_i^H u_j) v_j^H z = \\
&= \sum_{i=1}^{r+1}\sigma_i^2 z^H v_i v_i^H z = \\
&= \sum_{i=1}^{r+1}\sigma_i^2 (v_i^H z)^H (v_i^H z) = \\
&= \sum_{i=1}^{r+1}\sigma_i^2 \left|v_i^H z\right|^2 \geq \\
&\geq \sigma_{r+1}^2 \sum_{i=1}^{r+1}\left|v_i^H z\right|^2 = \\
&= \sigma_{r+1}^2 \sum_{i=1}^{r+1}\left|v_i^H \sum_{j=1}^{r+1}\alpha_j v_j\right|^2 = \\
&= \sigma_{r+1}^2 \sum_{i=1}^{r+1}\left|\sum_{j=1}^{r+1}\alpha_j v_i^H v_j\right|^2 = \\
&= \sigma_{r+1}^2 \sum_{i=1}^{r+1}|\alpha_i|^2 = \\
&= \sigma_{r+1}^2 \|z\|_2^2 = \\
&= \sigma_{r+1}^2
\end{aligned}
$$

da cui la tesi. □

12.3 Decomposizione ai valori singolari troncata (TSVD)

Supponiamo di voler risolvere il problema $Ax = b$ dove A è una matrice con condizionamento molto alto (ad esempio $\mu(A) = 10^{12}$ con precisione di macchina $u = 10^{-16}$). Per ovviare al numero di condizionamento possiamo aumentare la precisione di macchina, ma questo comporta anche un aumento di tempo e

spazio necessari per svolgere i conti. Proviamo allora a trovare un altro metodo. Data $A = U\Sigma V^H$ sappiamo che vale $\mu_2(A) = \frac{\sigma_1}{\sigma_n}$, e nel nostro caso σ_n è molto piccolo rispetto a σ_1. Consideriamo ora Σ_{n-1} definita come nella sezione precedente prendendo $r = n-1$, e consideriamo $\hat{A} = U\Sigma_{n-1}V^H$. Varrà $\mu_2(\hat{A}) = \frac{\sigma_1}{\sigma_{n-1}}$ e nel caso σ_{n-1} fosse considerevolmente più grande di σ_n, il numero di condizionamento di $\hat{A}$ (ottenuta come matrice più vicina ad A di rango minore di uno di quello di A) sarebbe considerevolmente più piccolo di quello di A.
Se invece σ_{n-1} non fosse significativamente più grande di σ_n è possibile utilizzare un certo valore di soglia (o di filtro) s e andando ad "annullare" (come abbiamo appena "annullato" σ_n in questo caso) tutti i valori singolari minori del valore di soglia. In questo caso varrebbe $\mu_2(\hat{A}) \leq \frac{\sigma_1}{s}$. Questo metodo risulta quindi efficiente solo se c'è un "salto" netto tra la soglia e i valori singolari non piccoli. In generale, però, non è detto che la soluzione di $\hat{A}x = b$ sia vicina alla soluzione del problema originale.

La decomposizione ai valori singolari troncata al valore di soglia s può essere usata per calcolare più accuratamente il rango di una matrice, correggendo errori che possono sorgere data la precisione di macchina finita. Teoricamente parlando, infatti, con una SVD di una matrice è possibile sapere quanti valori singolari sono non nulli, e questo numero corrisponde al rango della matrice. Computazionalmente, però, è possibile che alcuni valori che dovrebbero essere nulli risultino solo molto piccoli, andando ad aumentare il rango calcolato della matrice di partenza. Vediamo un esempio.
Consideriamo la matrice di rango 2

$$A = \begin{pmatrix} 1 & 2 & 3 \\ \frac{1}{2} & \frac{1}{3} & \frac{5}{6} \\ -1 & 4 & 3 \end{pmatrix}$$

E' possibile utilizzare il comando di MatLab svd(A) per ottenere i valori singolari di A. Quello che otterremmo, però, sarebbe

$$\text{svd}(A) = \begin{pmatrix} 6.1420 \\ 1.8252 \\ 2.4094 \cdot 10^{-16} \end{pmatrix}$$

ovvero tre valori singolari non nulli invece di due. Per risolvere questo problema, MatLab calcola il rango di una matrice con una TSVD utilizzando come valore di soglia $s = \max\{m, n\} \cdot \sigma_1 \cdot eps$, dove eps è la precisione di macchina. Ad oggi questo è il metodo più affidabile per il calcolo del rango di una matrice.

13 30/10/2020

13.1 Compressione di immagini tramite SVD

Vediamo ora come utilizzare le SVD per comprimere un'immagine, ovvero ottenere un'immagine sufficientemente simile all'originale ma che richieda meno informazioni da salvare.

Definiamo un'immagine come un insieme di pixel, ovvero una matrice $m \times n$. Se l'immagine è in bianco e nero possiamo salvare ogni pixel come un float tra 0 e 1, e occupa quindi mn spazio. Se è a colori abbiamo bisogno di $3mn$ spazio([13]).
Abbiamo visto che se abbiamo una SVD di $A = U\Sigma V^H$, è possibile ottenere $A \approx A_k := \sum_{i=i}^{k} \sigma_i u_i v_i^H$ con $\|A - A_k\|_2 = \sigma_{k+1}$.
Se vogliamo memorizzare A_k invece di A, basta memorizzare $u_1, \dots, u_k$ vettori di lunghezza m e $\sigma_1 v_1, \dots, \sigma_k v_k$ vettori di lunghezza n per poterla ricostruire. In questo caso lo spazio necessario è $mk + nk = (m+n)k$ invece di mn.
Poiché k può essere preso sufficientemente più piccolo di m e di n, questo procedimento diminuisce la quantità di informazione da dover memorizzare.
In generale, però, non esiste un metodo per scegliere un k ottimale, ovvero per ridurre al minimo l'informazione da memorizzare pur lasciando riconoscibile l'immagine originale.

13.2 Approssimazione polinomiale in norma 2 tramite SVD

Sia f una funzione reale e siano $\{x_i\}_{i=1}^m$ un insieme di punti a due a due distinti nel dominio di f. Supponiamo di conoscere i valori $\{f(x_i)\}_{i=1}^m$. Fissato n, vogliamo trovare un polinomio $p_{n-1}(x) = \alpha_0 + \alpha_1 x + \cdots + \alpha_{n-1} x^{n-1}$ di grado minore o uguale a $n-1$ tale che

$$\sum_{i=1}^{m} (p_{n-1}(x_i) - f(x_i))^2$$

sia minimo. Chiamiamo tale polinomio il polinomio di approssimazione ai minimi quadrati di f.

Riformuliamo il problema. Definiamo

$$A = \begin{pmatrix} x_1^0 & \dots & x_1^{n-1} \\ \vdots & & \vdots \\ x_m^0 & \dots & x_m^{n-1} \end{pmatrix}, \quad y = \begin{pmatrix} \alpha_0 \\ \vdots \\ \alpha_{n-1} \end{pmatrix}, \quad b = \begin{pmatrix} f(x_1) \\ \vdots \\ f(x_m) \end{pmatrix}$$

A questo punto vogliamo determinare y che minimizzi $\|Ay - b\|_2$. Abbiamo quindi trasformato il problema di approssimazione polinomiale ad un problema

[13] Vedasi la scomposizione RGB dei colori nei computer

dei minimi quadrati.

Osservazione. Per quanto visto ad Analisi Numerica, poiché gli x_i sono a due a due distinti, la matrice A (che è di Vandermonde) ha rango massimo, quindi la soluzione al problema dei minimi quadrati esiste ed è unica.
Inoltre, nel caso in cui $m = n$, il problema diventa il problema di interpolazione, già visto ad Analisi Numerica.

13.3 Algoritmo di Lanczos

Alcuni dei metodi per risolvere il problema dei minimi quadrati possono diventare inutilizzabili nel caso in cui la matrice considerata sia sparsa. Per di più, in alcuni casi è possibile sfruttare la struttura di sparsità della matrice per rendere il procedimento utilizzato molto più efficiente.

Data la matrice $A \in \mathbb{C}^{m\times n}$ consideriamo la matrice B così costruita

$$B := \left(\begin{array}{c|c} 0 & A \\ \hline A^H & 0 \end{array}\right) \in \mathbb{C}^{(m+n)\times(m+n)}$$

B è una matrice hermitiana. Inoltre, data $A = U\Sigma V^H$ una SVD di A e scomposte le matrici U e Σ in

$$U = \left(U_1 \mid U_2\right), \quad U_1 \in \mathbb{C}^{m\times n}, U_2 \in \mathbb{C}^{m\times(m-n)}$$
$$\Sigma = \left(\frac{\Sigma_1}{0}\right), \quad \Sigma_1 \in \mathbb{R}^{n\times n}$$

possiamo definire la seguente matrice unitaria

$$Z := \frac{1}{\sqrt{2}} \left(\begin{array}{c|c|c} U_1 & U_1 & \sqrt{2}U_2 \\ \hline V & -V & 0 \end{array}\right)$$

che ci permette di riscrivere B nel seguente modo

$$B = Z \left(\begin{array}{c|c|c} \Sigma_1 & 0 & 0 \\ \hline 0 & -\Sigma_1 & 0 \\ \hline 0 & 0 & 0 \end{array}\right) Z^H$$

Da questa scomposizione segue che gli autovalori di B sono $\sigma_i, -\sigma_i$ per $i = 1, \ldots, n$ più l'autovalore 0 con una certa molteplicità, e che le colonne di Z sono un insieme ortonormale di autovettori di B.
Nel caso particolare di $m = n$ si ha che $U = U_1$ e che

$$Z = \frac{1}{\sqrt{2}} \left(\begin{array}{c|c} U & U \\ \hline V & -V \end{array}\right)$$

Da cui si ottiene che gli autovettori di B saranno tutti della forma $z = \frac{1}{\sqrt{2}}\begin{pmatrix} u \\ v \end{pmatrix}$ con u, v entrambi normalizzati per norma 2.
Se invece $m > n$, B ha anche degli autovettori della forma $z = \begin{pmatrix} u \\ 0 \end{pmatrix}$ associati all'autovalore 0.

Introduciamo ora un algoritmo che tornerà utile al nostro scopo. L'algoritmo di Lanczos permette di trovare esplicitamente, data una matrice $A \in \mathbb{C}^{n\times n}$ hermitiana, una matrice $Q = (q_1, \dots, q_n)$ unitaria tale che

$$Q^H A Q := T = \begin{pmatrix} \alpha_1 & \beta_1 & & \\ \beta_1 & \ddots & \ddots & \\ & \ddots & \ddots & \beta_{n-1} \\ & & \beta_{n-1} & \alpha_n \end{pmatrix}$$

partendo dal vettore q_1. Vediamo ora come procedere per ottenere gli altri vettori $q_2, \dots, q_n$.
Si parte rigirando l'equazione $Q^H AQ = T$ per ottenere $AQ = QT$. Da questa otteniamo delle relazioni ricorsive tra i vettori q_i. Leggendo l'equazione colonna per colonna si ottiene

$$\begin{cases} Aq_1 = Q \begin{pmatrix} \alpha_1 \\ \beta_1 \\ 0 \\ \vdots \\ 0 \end{pmatrix} = \alpha_1 q_1 + \beta_1 q_2 \\ Aq_i = \beta_{i-1} q_{i-1} + \alpha_i q_i + \beta_{i-1} q_{i+1} \quad \text{per } i = 2, \dots, n-1 \\ Aq_n = \beta_{n-1} q_{n-1} + \alpha_n q_n \end{cases}$$

Prendendo la prima equazione otteniamo che, poiché le colonne di Q devono essere ortogonali, deve valere

$$q_1^H A q_1 = \alpha_1 \underbrace{q_1^H q_1}_{=1} + \beta_1 \underbrace{q_1^H q_2}_{=0} = \alpha_1$$

e anche

$$q_2 = \frac{1}{\beta_1}(A - \alpha_1 I) q_1$$

dove necessariamente $\beta_1 = \left\|(A - \alpha_1 I) q_1\right\|_2$ perché per avere Q ortonormale le sue colonne devono essere normalizzate per norma 2.
E' quindi possibile determinare α_1, β_1 e q_2 conoscendo solo A e q_1.
Analogamente, prendendo la seconda equazione otteniamo che

$$\begin{aligned} \alpha_i &= q_i^H A q_i \\ q_{i+1} &= \frac{1}{\beta_i}\left[(A - \alpha_i I) q_i - \beta_{i-1} q_{i-1}\right] \end{aligned}$$

Quindi iterativamente riusciamo a costruire tutti gli α_i, β_i e q_i partendo solo da A e da q_i. Inoltre, per costruzione vale che per ogni i, $\alpha_i, \beta_i \in \mathbb{R}$ e che $\beta_i \geq 0$.

Osservazione. E' possibile che in un'iterazione si ottenga $\beta_i = 0$. In questo caso l'algoritmo procede prendendo q_{i+1} uguale ad un qualsiasi vettore nell'ortogonale del generato di $q_1, \ldots, q_i$.

Osservazione. Partendo da una matrice A hermitiana generica l'algoritmo costa $O(n^3)$. Tuttavia, prendendo ad esempio A una matrice a banda $2p+1$, il costo dell'algoritmo è solo $(2p+6)n^2$. Questo algoritmo è in genere indicato per le matrici di grossa taglia e sparse.

Vediamo ora un teorema che mostra come una scelta opportuna di q_1 permetta di minimizzare sia la complessità in tempo che quella in spazio dell'algoritmo di Lanczos applicato alla matrice di partenza $B = \left(\begin{array}{c|c} 0 & A \\ \hline A^H & 0 \end{array}\right)$ nel caso in cui A sia reale.

Teorema 13.1 (di Golub e Kahan)**.** *Sia $A \in \mathbb{R}^{m \times n}$ con $m \geq n$ e B definita come sopra.*
Allora scegliendo q_1 in una delle seguenti forme

- $q_1 = \begin{pmatrix} u \\ 0 \end{pmatrix} \in \mathbb{R}^{m+n}$ *con* $u \in \mathbb{R}^m$ *tc* $\|u\|_2 = 1$, *oppure*
- $q_1 = \begin{pmatrix} 0 \\ v \end{pmatrix} \in \mathbb{R}^{m+n}$ *con* $v \in \mathbb{R}^n$ *tc* $\|v\|_2 = 1$

l'algoritmo di Lanczos genera una matrice tridiagonale dove tutti gli α_i sono nulli.
Inoltre, i vettori q_i generati assumono in maniera alternata le due forme possibili mostrate per q_1.

Dimostrazione. Consideriamo il caso $q_1 = \begin{pmatrix} u_1 \\ 0 \end{pmatrix}$. La dimostrazione per l'altra scelta di q_1 è del tutto analoga.
Svolgendo l'algoritmo di Lanczos otteniamo

$$\begin{aligned} \alpha_1 &= q_1^H B q_1 = \\ &= \begin{pmatrix} u_1^H & 0 \end{pmatrix} \left(\begin{array}{c|c} 0 & A \\ \hline A^H & 0 \end{array}\right) \begin{pmatrix} u_1 \\ 0 \end{pmatrix} = \\ &= \begin{pmatrix} 0 & u_1^H A \end{pmatrix} \begin{pmatrix} u_1 \\ 0 \end{pmatrix} = \\ &= 0 \end{aligned}$$

e per q_2 otteniamo

$$\begin{aligned} q_2 &= \frac{1}{\beta_1} B q_1 = \\ &= \frac{1}{\beta_1} \left(\begin{array}{c|c} 0 & A \\ \hline A^H & 0 \end{array}\right) \begin{pmatrix} u_1 \\ 0 \end{pmatrix} = \\ &= \frac{1}{\beta_1} \begin{pmatrix} 0 \\ A^H u_1 \end{pmatrix} =: \\ &=: \begin{pmatrix} 0 \\ v_2 \end{pmatrix} \end{aligned}$$

e poiché β_1 è preso in modo da normalizzare q_2, vale che $\|v_2\|_2 = 1$, quindi q_2 è in una delle forme possibili per q_1, e precisamente nell'altra rispetto a q_1.
Analogamente ad α_1 si ottiene che $\alpha_2 = 0$. Per calcolare q_3 otteniamo

$$\begin{aligned} q_3 &= \frac{1}{\beta_2} \left[B q_2 - \beta_1 q_1\right] = \\ &= \frac{1}{\beta_2} \left[\left(\begin{array}{c|c} 0 & A \\ \hline A^H & 0 \end{array}\right) \begin{pmatrix} 0 \\ v_2 \end{pmatrix} - \beta_1 \begin{pmatrix} u_1 \\ 0 \end{pmatrix} \right] = \\ &= \frac{1}{\beta_2} \left[\begin{pmatrix} A v_2 \\ 0 \end{pmatrix} - \beta_1 \begin{pmatrix} u_1 \\ 0 \end{pmatrix} \right] =: \\ &=: \begin{pmatrix} u_3 \\ 0 \end{pmatrix} \end{aligned}$$

e analogamente a prima abbiamo $\|u_3\|_2 = 1$. Iterando il processo per il generico vettore q_i (che è facile da immaginare ma leggermente complicato da scrivere perché bisognerebbe distinguere in due casi in base a che forma assumeva q_{i-1}) si ottiene la tesi. □

13.4 Metodi di rilassamento

Introduciamo ora un nuovo argomento. Sia $A \in \mathbb{C}^{n\times n}, b \in \mathbb{C}^n$ e cerchiamo la soluzione x al problema

$$Ax = b$$

Sebbene esistano metodi per calcolare soluzioni esatte a questo problema, è possibile applicare metodi iterativi per ottenere approssimazioni della soluzione vicine a piacere alla soluzione esatta.
Per esempio, data una matrice $A \in \mathbb{C}^{n\times n}$ con $\det(A) \neq 0$ possiamo scomporre la matrice $A = M - N$, dove richiediamo che $\det(M) \neq 0$, per poter trasformare il problema originale in

$$x = M^{-1}Nx + M^{-1}b$$

e dette $P := M^{-1}N$ e $q := M^{-1}b$, il problema diventa un problema di punto fisso per la funzione $Px + q$, che può essere risolto iterativamente con

$$x^{(k)} = Px^{(k-1)} + q$$

Questo tipo di approccio è già stato trattato nel corso di Analisi Numerica, in particolare per quanto riguarda il metodo di Gauss-Seidel e il metodo di Jacobi, che rinfreschiamo ora brevemente.
Scomponiamo $A = D - B - C$ dove

$$D := (d_{i,j}),\ \text{con}\ d_{i,j} = \begin{cases} a_{i,j} & \text{se } i = j \\ 0 & \text{altrimenti} \end{cases}$$

$$B := (b_{i,j}),\ \text{con}\ b_{i,j} = \begin{cases} -a_{i,j} & \text{se } i > j \\ 0 & \text{altrimenti} \end{cases}$$

$$C := (c_{i,j}),\ \text{con}\ c_{i,j} = \begin{cases} -a_{i,j} & \text{se } i < j \\ 0 & \text{altrimenti} \end{cases}$$

Questa scomposizione permette due tipi di raggruppamenti:

- $M = D, N = B + C$, che è la premessa del metodo di Jacobi, e che genera una matrice $P_J = D^{-1}(B + C)$
- $M = D - B, N = C$, che è la premessa del metodo di Gauss-Seidel, e che genera una matrice $P_{GS} = (D - B)^{-1}C$

Ricordiamo ora brevemente un teorema visto ad Analisi Numerica (non ne riporteremo quindi la dimostrazione):

Teorema 13.2. *Se A è una matrice tridiagonale, allora $\rho(P_{GS}) = \rho(P_J)^2$*

Poiché (sempre per quanto visto ad Analisi Numerica) la velocità di convergenza del metodo iterativo dipende dal raggio spettrale della matrice P, nel caso in cui almeno una delle due tra Jacobi e Gauss-Seidel converga (ovvero almeno una delle due, e quindi entrambe, abbiano raggio spettrale minore di 1), conviene scegliere il metodo di Gauss-Seidel poiché con raggio spettrale più piccolo.
Poiché abbiamo appena visto che la velocità di convergenza può dipendere dalla scomposizione di A che si usa, la domanda che viene spontaneo da chiedersi è se esistano metodi che migliorino la velocità di convergenza rispetto a quella di Gauss-Seidel. Introduciamo quindi i metodi di rilassamento.

Prendiamo $\omega \in \mathbb{C} \setminus \{0\}$ e riscriviamo il problema originale come

$$\omega Ax = \omega b$$

e consideriamo la seguente scomposizione

$$M = D - \omega B, \quad N = (1 - \omega)D + \omega C$$

dove D, B e C sono definite come sopra.

Definizione. Nel caso di ω reale, se $\omega < 1$ si parla di metodo di sottorilassamento, mentre se $\omega > 1$ si parla di metodo di sovrarilassamento. Quest'ultimo è noto anche come SOR (dall'inglese, successive over-relaxation).

Osservazione. Per $\omega = 1$ si riottiene Gauss-Seidel.

Il metodo iterativo generato da questa scomposizione è il seguente

$$x^{(k)} = \underbrace{(D - \omega B)^{-1} \left[(1-\omega)D + \omega C\right]}_{=:H(w)} x^{(k-1)} + \omega (D - \omega B)^{-1} b$$

Operativamente, otteniamo che

$$\begin{aligned} x^{(k)} &= H(w)x^{(k-1)} + \omega(D - \omega B)^{-1} b \\ &\Downarrow \\ Dx^{(k)} &= (1-\omega)Dx^{(k-1)} + \omega B x^{(k)} + \omega C x^{(k-1)} + \omega b \\ &\Downarrow \\ x^{(k)} &= (1-\omega)x^{(k-1)} + \omega D^{-1}\left[Bx^{(k)} + Cx^{(k-1)} + b\right] \end{aligned}$$

da cui, coordinata per coordinata,

$$x_i^{(k)} = (1-\omega)x_i^{(k-1)} + \frac{\omega}{a_{i,i}} \left[b_i - \sum_{j=1}^{i-1} a_{i,j} x_j^{(k)} - \sum_{j=i+1}^{n} a_{i,j} x_j^{(k-1)} \right]$$

14 04/11/2020

14.1 Velocità di convergenza del metodo di rilassamento

Abbiamo visto nella scorsa lezione la definizione di metodo di rilassamento, ovvero un metodo iterativo convergente alla soluzione del problema $Ax = b$. Abbiamo visto che la velocità di convergenza dipende dal raggio spettrale della matrice $H(\omega)$, che dipende da ω. Vediamo ora che relazioni ci sono tra $\rho(H(\omega))$ e ω, e in che condizioni possiamo avere velocità di convergenza ottimale.

Teorema 14.1 (di Kahan)**.** $\rho(H(\omega)) \geq |1 - \omega|$

Dimostrazione. Il determinante di una matrice corrisponde al prodotto dei suoi autovalori. In particolare

$$\big|\det(H(\omega))\big| = \prod_{\lambda \in \mathrm{Sp}(H(\omega))} |\lambda| \leq \rho(H(\omega))^n$$

Vale anche, per Binet,

$$\det(H(\omega)) = (\det(D - \omega B))^{-1} \det((1-\omega)D + \omega C)$$

Poiché $(D - \omega B)$ è triangolare inferiore, il suo determinante coincide con il prodotto degli elementi diagonali, ovvero

$$\det(D - \omega B) = \det(D)$$

In modo analogo, e sfruttando la multilinearità del determinante, si ottiene

$$\det((1-\omega)D + \omega C) = \det((1-\omega)D) = (1-\omega)^n \det(D)$$

Sostituendo i due risultati appena ottenuti nella formula per il determinante di $H(\omega)$ otteniamo

$$\det(H(\omega)) = (\det(D))^{-1}(1-\omega)^n \det(D) = (1-\omega)^n$$

Allora si ha

$$\begin{aligned} |1-\omega|^n &= \big|(1-\omega)^n\big| = \\ &= \big|\det(H(\omega))\big| = \\ &= \prod_{\lambda \in \mathrm{Sp}(H(\omega))} |\lambda| \leq \\ &\leq \rho(H(\omega))^n \end{aligned}$$

da cui si ottiene la tesi. □

Osservazione. Poiché per la convergenza è necessario che $\rho(H(\omega)) < 1$, abbiamo come condizione necessaria per la convergenza $|1 - \omega| < 1$. In particolare, nel caso di ω reale, deve valere $0 < \omega < 2$.

Vediamo ora come, nel caso di A definita positiva e ω reale, è possibile garantire la convergenza del metodo di rilassamento.

Teorema 14.2 (di Ostrowski-Reich)**.** *Sia A definita positiva (ovvero hermitiana e $\forall x \in \mathbb{C}^n \setminus \{0\}, x^H Ax > 0$). Se $0 < \omega < 2$ allora il metodo di rilassamento è convergente.*

Dimostrazione. Vogliamo mostrare che $\rho(H(\omega)) < 1$.
Poiché $A = A^H$, vale $C = B^H$. Riscriviamo allora $H(\omega)$.

$$\begin{aligned} H(\omega) &= (D - \omega B)^{-1} \left[(1-\omega)D + \omega B^H\right] = \\ &= (D - \omega B)^{-1} \left[D - \omega D + \omega B^H\right] = \\ &= (D - \omega B)^{-1} \left[D - \omega B - \omega A\right] = \\ &= I - \omega (D - \omega B)^{-1} A =: \\ &=: I - F \end{aligned}$$

Consideriamo ora la matrice $(A - H(\omega)^H AH(\omega))$ e facciamo vedere che è anch'essa definita positiva. Riscriviamola nel seguente modo

$$\begin{aligned} A - H(\omega)^H AH(\omega) &= A - (I-F)^H A(I-F) = \\ &= A - A + AF + F^H A - F^H AF = \\ &= F^H (F^{-H} A + AF^{-1} - A)F = \\ &= F^H \left(\frac{1}{\omega}(D - \omega B^H)A^{-1}A + A\frac{1}{\omega}A^{-1}(D - \omega B) - A\right) F = \\ &= F^H \left(\frac{2}{\omega} D \underbrace{- B^H - B - A}_{-D}\right) F = \\ &= \left(\frac{2}{\omega} - 1\right) F^H D F \end{aligned}$$

Osserviamo ora che, poiché $0 < \omega < 2$, $\frac{2}{\omega} - 1 > 0$. Inoltre, D è diagonale e positiva ($d_{i,i} = e_i^H A e_i > 0$) quindi in particolare definita positiva, e la proprietà di essere definiti positivi è invariante per congruenza. Quindi anche $F^H DF$ moltiplicata per una costante positiva è definita positiva. In particolare, $A - H(\omega)^H AH(\omega)$ è definita positiva.
Sfruttando questo fatto, mostriamo che $\rho(H(\omega)) < 1$. Sia λ un autovalore di $H(\omega)$ e x un relativo autovettore. Poiché $A - H(\omega)^H AH(\omega)$ è definita positiva deve valere

$$x^H (A - H(\omega)^H AH(\omega))x > 0$$

Sviluppiamo e otteniamo

$$\begin{aligned} 0 < x^H(A - H(\omega)^H A H(\omega))x = \\ = x^H A x - x^H H(\omega)^H A H(\omega) x = \\ = x^H A x - (H(\omega)x)^H A (H(\omega)x) = \\ = x^H A x - |\lambda|^2 x^H A x = \\ = (1 - |\lambda|^2) \underbrace{x^H A x}_{>0} \end{aligned}$$

da cui segue che necessariamente $1 - |\lambda|^2 > 0$ e quindi la tesi $\square$

Nel caso in cui A sia tridiagonale è possibile migliorare il risultato. In particolare è possibile trovare esplicitamente ω che massimizza la velocità di convergenza, e si dimostra che è unico, ovvero:

$$\exists! \omega_0 \text{ t.c. } \rho(H(\omega_0)) = \min_{0<\omega<2} \rho(H(\omega))$$

e

$$\omega_0 = \frac{2}{1 + \sqrt{1 - \rho(P_J)^2}}$$

Per gli interessati, questo fatto è dimostrato nella dimostrazione del teorema 5.28 del libro Metodi Numerici per l'Algebra Lineare.

Il miglioramento dato da questa scelta di ω può essere significativo. Vediamo un esempio pratico:

$$A = \begin{pmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{pmatrix} \in \mathbb{R}^{6\times 6}$$

In questo caso vale

$$\begin{aligned} \rho(P_J) &\approx 0.9009688 \\ \rho(P_{GS}) &\approx 0.81 \\ \omega_0 &\approx 1.394812 \\ \rho(H(\omega_0)) &\approx 0.3949117 \end{aligned}$$

$$\rho(H(\omega_0)) \approx \rho(P_J)^9 \approx \rho(P_{GS})^{4.5}$$

In questo caso, quindi, il metodo ottenuto con questa scelta di ω è 9 volte più veloce del metodo di Jacobi e 4.5 volte più veloce del metodo di Gauss-Seidel.

14.2 Equazione di Poisson monodimensionale

Cerchiamo di risolvere il seguente problema differenziale

$$-\frac{d^2}{dx^2}v(x) = f(x), \quad 0 < x < 1$$

Dove v è incognita e f è nota. Un'equazione del genere si chiama equazione di Poisson monodimensionale.
Aggiungiamo nelle ipotesi delle condizioni agli estremi, ovvero $v(0) = v(1) = 0$ (dette condizioni di Dirichlet).

Un metodo per risolvere questo problema comincia prendendo $N + 2$ punti equispaziati nell'intervallo $[0, 1]$ (la scelta di $N+2$ è data dal fatto che 2 di questi punti saranno 0 e 1, e i valori di v in questi punti sono già noti. Così facendo lavoreremo con esattamente N valori incogniti). Ponendo $h = \frac{1}{N+1}$ possiamo definire questi punti come $x_i := ih$, $i = 0, \ldots, N + 1$. Poniamo ora $v_i := v(x_i)$ e $f_i := f(x_i)$.
Scriviamo ora due sviluppi di Taylor al quarto ordine di v centrato in x_i.

$$v(x_i - h) = v(x_i) - hv'(x_i) + \frac{h^2}{2}v^{(2)}(x_i) - \frac{h^3}{3!}v^{(3)}(x_i) + O(h^4)$$
$$v(x_i + h) = v(x_i) + hv'(x_i) + \frac{h^2}{2}v^{(2)}(x_i) + \frac{h^3}{3!}v^{(3)}(x_i) + O(h^4)$$

Sommando membro a membro otteniamo

$$v^{(2)}(x_i) = \frac{1}{h^2}(v(x_i - h) - 2v(x_i) + v(x_i + h)) - \underbrace{O(h^2)}_{\tau_i}$$

che secondo la notazione introdotta prima e sfruttando $v^{(2)}(x) = f(x)$, diventa

$$v_{i-1} - 2v_i + v_{i+1} = h^2 f_i + h^2\tau_i, \qquad i = 1, \ldots, N$$

Poiché abbiamo aggiunto le condizioni $v_0 = v_{N+1} = 0$ possiamo riscrivere l'insieme di queste equazioni come un sistema lineare a N incognite e N equazioni:

$$\underbrace{\begin{pmatrix} 2 & -1 & & \\ -1 & \ddots & \ddots & \\ & \ddots & \ddots & -1 \\ & & -1 & 2 \end{pmatrix}}_{=:T_N} \underbrace{\begin{pmatrix} v_1 \\ \vdots \\ v_N \end{pmatrix}}_{=:v} = h^2 \underbrace{\begin{pmatrix} f_1 \\ \vdots \\ f_N \end{pmatrix}}_{=:f} + h^2 \underbrace{\begin{pmatrix} \tau_1 \\ \vdots \\ \tau_N \end{pmatrix}}_{=:\tau}$$

Poiché però tutte le componenti di τ sono degli $O(h^2)$, l'apporto dato da $h^2\tau$ è un $O(h^4)$. E' lecito quindi considerare il problema approssimato

$$T_N\hat{v} = h^2 f$$

15 06/11/2020

15.1 Considerazioni sull'equazione di Poisson monodimensionale

Nella scorsa lezione abbiamo visto come trasformare il problema dato dall'equazione di Poisson in un sistema lineare la cui matrice dei coefficienti è T_N. Facciamo ora delle considerazioni su questo sistema lineare.

Cerchiamo di determinare quanto bene o mal condizionata sia T_N. E' possibile dimostrare che gli autovalori di T_N sono

$$\lambda_j = 2\left(1 - \cos\frac{j\pi}{N+1}\right), \qquad j = 1, \dots N$$

e che gli autovettori relativi sono dei vettori z_j normalizzati la cui componente k esima è data da

$$z_j(k) = \sqrt{\frac{2}{N+1}} \sin\frac{jk\pi}{N+1}$$

In particolare possiamo ottenere una decomposizione spettrale di T_N con

$$T_N = Z\Lambda Z^T$$

dove $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_N)$ e $Z = (z_1, \dots, z_N)$. Ora che conosciamo gli autovalori di T_N possiamo fare delle stime su $\mu(T_n)$.
Con un semplice studio degli autovalori si ottiene che l'autovalore di modulo massimo sarà λ_N e quello di modulo minimo sarà λ_1. In particolare, per N sufficientemente grande abbiamo che

$$\begin{aligned}
\lambda_N &= 2\left(1 - \cos\frac{N}{N+1}\pi\right) \approx 4 \\
\lambda_1 &= 2\left(1 - \cos\frac{\pi}{N+1}\right) \approx \\
&\approx 2\left(1 - \left(1 - \frac{\pi^2}{2(N+1)^2}\right)\right) = \\
&= \frac{\pi^2}{(N+1)^2}
\end{aligned}$$

Otteniamo quindi che, per N sufficientemente grande, abbiamo $\mu(T_N) \approx \frac{4(N+1)^2}{\pi^2}$.

Dato il problema $T_N v = h^2 f + h^2\tau$, avevamo ipotizzato di semplificarlo al problema $T_N\hat{v} = h^2 f$ poiché le componenti di τ sono degli $O(h^2)$. Resta da verificare però quanto lecita sia questa semplificazione, ovvero quanto vicine siano le soluzioni v e $\hat{v}$. Proviamo quindi a stimare $\|v - \hat{v}\|_2$.

Sicuramente vale $T_N(v - \hat{v}) = h^2\tau$. Allora

$$\begin{aligned} v - \hat{v} &= h^2 T_N^{-1}\tau \\ \|v - \hat{v}\|_2 &= \left\|h^2 T_N^{-1}\tau\right\|_2 \leq \\ &\leq h^2 \left\|T_N^{-1}\right\|_2 \|\tau\|_2 \approx \\ &\approx h^2 \frac{(N+1)^2}{\pi^2}\|\tau\|_2 = \\ &= \frac{1}{(N+1)^2}\frac{(N+1)^2}{\pi^2}\|\tau\|_2 = \\ &= O(\|\tau\|_2) = \\ &= O\left(h^2\left\|v^{(4)}\right\|_\infty\right) \end{aligned}$$

Quindi supponendo che v sia sufficientemente regolare (ovvero che $\left\|v^{(4)}\right\|_\infty$ sia limitata), l'approssimazione di v a $\hat{v}$ è lecita.

Prima di passare all'equazione di Poisson in dimensione 2, esplicitiamo una forte relazione tra il sistema lineare appena descritto e un altro problema differenziale, che ci aiuterà per algoritmi futuri([14]).
E' possibile riscrivere il sistema lineare che stiamo considerando come

$$h^{-2}T_N v = f$$

Vediamo come gli autovalori e autovettori della matrice $h^{-2}T_N$ sono strettamente legati alle soluzioni del seguente problema differenziale:

$$\begin{cases} -\frac{d^2}{dx^2}\hat{z}_i(x) = \hat{\lambda}_i \hat{z}_i(x) \\ \hat{z}_i(0) = \hat{z}_i(1) = 0 \end{cases}$$

dove $\hat{z}_i$ e λ_i sono entrambe incognite([15]).
Con un po' di esperienza delle equazioni differenziali si ottiene che le soluzioni della prima equazione sono le $\hat{z}_i$ della forma

$$\hat{z}_i(x) = \alpha \sin(x\sqrt{\hat{\lambda}_i}) + \beta\cos(x\sqrt{\hat{\lambda}_i})$$

Dalla condizione $\hat{z}_i(0) = 0$ si ricava che necessariamente deve valere $\beta = 0$.
Dalla condizione $\hat{z}_i(1) = 0$ si ricava che $\alpha\sin(\sqrt{\hat{\lambda}_i}) = 0$. Da questa condizione si ottiene o che $\alpha = 0$, che implicherebbe la soluzione banale di $\hat{z}_i(x) = 0$ per ogni

[14] In classe non abbiamo davvero detto perché abbiamo esplicitato questa correlazione. Il Demmel la giustifica con "This correspondence will be the motivation for the design and analysis of later algorithms."

[15] Non è stato detto in classe ma il Demmel lo dice: una funzione $\hat{z}_i$ che rispetti la prima equazione viene detta autofunzione dell'operatore di derivata seconda

x, oppure che $\sin(\sqrt{\hat{\lambda}_i}) = 0$. Da quest'ultima condizione ricaviamo che $\sqrt{\hat{\lambda}_i}$ deve essere un multiplo qualsiasi di π e senza perdita di generalità possiamo scegliere $\sqrt{\hat{\lambda}_i} = i\pi$.
Da ciò ricaviamo le soluzioni

$$\hat{\lambda}_i = i^2\pi^2$$
$$\hat{z}_i(x) = \alpha \sin(i\pi x)$$

Queste soluzioni sono strettamente correlate agli autovalori e autovettori di $h^{-2}T_N$ per N grande. Infatti, gli autovalori di $h^{-2}T_N$ sono $h^{-2}\lambda_i$ e

$$h^{-2}\lambda_i = (N+1)^2 2\left(1 - \cos\frac{i\pi}{N+1}\right) \approx (N+1)^2 2\frac{i^2\pi^2}{2(N+1)^2} = i^2\pi^2 = \hat{\lambda}_i$$

Analogamente, prendendo $\alpha = \sqrt{\frac{2}{N+1}}$ (e lo prendiamo così solo perché normalizza il risultato che stiamo per ottenere) abbiamo che

$$z_i(k) = \hat{z}_i(x_k)$$

15.2 Equazione di Poisson bidimensionale

Cerchiamo di risolvere il seguente problema differenziale

$$-\frac{\partial^2}{\partial x^2}v(x,y) - \frac{\partial^2}{\partial y^2}v(x,y) = f(x,y), \qquad (x,y) \in \Omega := (0,1)\times(0,1)$$

che prende il nome di equazione di Poisson bidimensionale.
Analogamente a quanto fatto in precedenza, aggiungiamo nelle ipotesi delle condizioni sugli estremi, ovvero che $v(x,y) = 0$ se $(x,y) \in \partial\Omega$, dove con $\partial\Omega$ indichiamo la frontiera di Ω.
Anche in questo caso dividiamo il quadrato $[0,1]\times[0,1]$ in un reticolo di $(N+2)\times(N+2)$ punti (in modo da avere esattamente $N \times N$ punti interni il cui valore di v sia effettivamente incognito, visto che il valore ai bordi è noto). Fissiamo N, poniamo $h := \frac{1}{N+1}$ e definiamo $x_i = y_i := ih$, $i = 0,\dots,N+1$. Abbiamo quindi diviso Ω in $N \times N$ punti della forma (x_i, y_j), $i,j = 1,\dots,N$. Poniamo $v_{i,j} := v(x_i, y_j)$ e $f_{i,j} := f(x_i, y_j)$.

Possiamo usare lo studio fatto nel caso monodimensionale per studiare le derivate parziali in x e in y. Otteniamo

$$-\left.\frac{\partial^2}{\partial x^2}v(x,y)\right|_{\substack{x=x_i\\y=y_j}} = \frac{2v_{i,j} - v_{i-1,j} - v_{i+1,j}}{h^2} + \tau_{i,j}$$
$$-\left.\frac{\partial^2}{\partial y^2}v(x,y)\right|_{\substack{x=x_i\\y=y_j}} = \frac{2v_{i,j} - v_{i,j-1} - v_{i,j+1}}{h^2} + \tau_{i,j}$$

Combinando le due si ottiene

$$f_{i,j} = \left(-\frac{\partial^2}{\partial x^2} v(x,y) - \frac{\partial^2}{\partial y^2} v(x,y) \right) \Bigg|_{\substack{x=x_i \\ y=y_j}} =$$
$$= \frac{4v_{i,j} - v_{i-1,j} - v_{i+1,j} - v_{i,j-1} - v_{i,j+1}}{h^2} + \tau_{i,j}$$

Questa formula viene anche chiamata formula a 5 punti, poiché per stimare la derivata in un punto vengono usati il punto stesso e i punti sopra, sotto, a destra e a sinistra (5 in totale). Sfruttando le condizioni $v_{0,j} = v_{N+1,j} = v_{i,0} = v_{i,N+1} = 0$ otteniamo N^2 equazioni lineari in N^2 incognite.
Per esplicitare queste equazioni in un sistema della forma $Av = f$ dobbiamo determinare un ordine con cui mettere i $v_{i,j}$ nel vettore delle incognite. Il modo che utilizzeremo per farlo è più facilmente mostrato che spiegato. Rappresentiamo l'insieme dei punti $v_{i,j}$ come una matrice e definiamo v_t tale che

$$\begin{vmatrix} v_{1,1} & v_{2,1} & \dots & v_{N,1} \\ v_{1,2} & v_{2,2} & & \vdots \\ \vdots & & \ddots & \vdots \\ v_{1,N} & \dots & \dots & v_{N,N} \end{vmatrix} = \begin{vmatrix} v_1 & v_{N+1} & \dots & v_{N(N-1)+1} \\ v_2 & v_{N+2} & & \vdots \\ \vdots & & \ddots & \vdots \\ v_N & \dots & \dots & v_{N^2} \end{vmatrix}$$

Per gli amanti delle formule, $v_{i,j} =: v_{N(i-1)+j}$. Numeriamo analogamente $f_{i,j} =: f_{N(i-1)+j}$.
Si può dimostrare (e si può provare a mano un caso con N piccolo per convincersi del perché funziona così) che ponendo $T_{N\times N}$ come la seguente matrice tridiagonale a blocchi

$$T_{N\times N} := \left(\begin{array}{c|c|c|c} T_N + 2I_N & -I_N & & \\ \hline -I_N & \ddots & \ddots & \\ \hline & \ddots & \ddots & -I_N \\ \hline & & -I_N & T_N + 2I_N \end{array}\right)$$

dove T_N è definita come nella lezione precedente e I_N è l'identità di taglia N, allora vale

$$T_{N\times N}\hat{v} = h^2 f$$

Facciamo ora qualche considerazione su $T_{N\times N}$.

Definizione (Prodotto di Kronecker)**.** Date due matrici A, B rispettivamente di taglie $m \times n$ e $p \times q$ qualsiasi, definiamo il loro prodotto di Kronecker, e indichiamo con $A \otimes B$, la matrice

$$A \otimes B = \left(\begin{array}{c|c|c} a_{1,1}B & \dots & a_{1,n}B \\ \hline \vdots & \ddots & \vdots \\ \hline a_{m,1}B & \dots & a_{m,n}B \end{array}\right)$$

Con questa definizione, basta scomporre $T_{N\times N}$ in

$$T_{N\times N} = \left(\begin{array}{c|c|c|c} T_N & & & \\ \hline & \ddots & & \\ \hline & & \ddots & \\ \hline & & & T_N \end{array}\right) + \left(\begin{array}{c|c|c|c} 2I_N & -I_N & & \\ \hline -I_N & \ddots & \ddots & \\ \hline & \ddots & \ddots & -I_N \\ \hline & & -I_N & 2I_N \end{array}\right)$$

per notare che vale

$$T_{N\times N} = I \otimes T_N + T_N \otimes I$$

Sappiamo, però, che $T_N = Z\Lambda Z^T$. Sfruttiamo ora due proprietà del prodotto di Kronecker per ottenere una decomposizione spettrale di $T_{N\times N}$.

- $(A \otimes B)^T = A^T \otimes B^T$
- $(A \otimes B)(C \otimes D) = (AC) \otimes (BD)$ se i prodotti AC e BD sono ben definiti

Sostituendo a T_N la sua decomposizione spettrale e utilizzando le proprietà otteniamo

$$T_{N\times N} = (Z \otimes Z)(I_N \otimes \Lambda + \Lambda \otimes I_N)(Z \otimes Z)^T$$

dove è facile verificare che la matrice $(I_N \otimes \Lambda + \Lambda \otimes I_N)$ sia diagonale.
Allora gli autovalori di $T_{N\times N}$ sono tutti della forma $\lambda_{i,j} := \lambda_i + \lambda_j$. Gli autovettori relativi sono $z_{i,j} := z_i \otimes z_j$.

15.3 Metodi iterativi per Poisson bidimensionale

Abbiamo visto come sia possibile trasformare l'equazione di Poisson bidimensionale in un problema di risoluzione di un sistema lineare, che sappiamo risolvere tramite metodi iterativi. Vediamo ora in pratica come si applicano tre dei sistemi iterativi che conosciamo su questo specifico problema. Analizzeremo il metodo di Jacobi, il metodo di Gauss-Seidel e il metodo di rilassamento SOR.

Sappiamo che il metodo di Jacobi si basa su un'iterazione della forma

$$x^{(m+1)} = D^{-1}(B + C)x^{(m)} + D^{-1}b$$

Algoritmicamente, un'iterazione di Jacobi diventa:

for int $i = 1$ to n **do**
  $x_i^{(m+1)} = \frac{1}{a_{i,i}} \left(b_i - \sum_{j\neq i} a_{i,j} x_j^{(m)}\right)$
end for

Nel nostro caso, riutilizzando la notazione $v_{i,j}$ invece della v_t poiché più comoda per scrivere il codice, l'm-esima iterazione diventa:

for int $i = 1$ to N **do**
  for int $j = 1$ to N **do**

$v_{i,j}^{(m+1)} = \frac{1}{4}(v_{i-1,j}^{(m)} + v_{i+1,j}^{(m)} + v_{i,j-1}^{(m)} + v_{i,j+1}^{(m)} + h^2 f_{i,j})$

end for

end for

Nel metodo di Jacobi, quindi, all'm-esima iterazione tutti i $v_{i,j}^{(m+1)}$ possono essere calcolati in ordine indipendente l'uno dall'altro. Ciò non vale per Gauss-Seidel e per il metodo di rilassamento SOR.

Il metodo di Gauss-Seidel si basa su un'iterazione della forma

$$x^{(m+1)} = (D - B)^{-1}Cx^{(m)} + (D - B)^{-1}b$$

che algoritmicamente diventa, rimaneggiando un po' l'equazione

for int $i = 1$ to n **do**

$x_i^{(m+1)} = \frac{1}{a_{i,i}} \left(b_i - \sum_{j<i} a_{i,j} x_j^{(m+1)} - \sum_{j>i} a_{i,j} x_j^{(m)} \right)$

end for

In questo caso quindi le componenti del vettore $x^{(m)}$ non possono essere calcolate in ordine indipendente ma vanno necessariamente calcolate dalla prima all'ultima.
Analogamente, l'iterazione di SOR

$$x^{(m+1)} = (1 - \omega)x^{(m)} + \omega D^{-1}(Bx^{(m+1)} + Cx^{(m)} + b)$$

induce il seguente algoritmo

for int $i = 1$ to n **do**

$x_i^{(m+1)} = (1 - \omega)x_i^{(m)} + \frac{\omega}{a_{i,i}} \left(b_i - \sum_{j<i} a_{i,j} x_j^{(m+1)} - \sum_{j>i} a_{i,j} x_j^{(m)} \right)$

end for

quindi anche in questo caso l'ordine in cui vengono calcolate le componenti di $x^{(m)}$ è obbligato. In generale però, poter calcolare più cose in ordine indipendente è una proprietà comoda da avere poiché permette di svolgere conti in parallelo. Vediamo quindi come è possibile modificare raggruppare le componenti (i, j) del vettore v per poter garantire una parziale indipendenza di ordine.

Preso l'insieme delle componenti $v_{i,j}$, immaginiamo di dividere il quadrato che esse descrivono in una scacchiera rossa e nera e dividiamo l'insieme originale delle componenti in due sottoinsiemi: le componenti rosse e le componenti nere (questo ordinamento prende il nome di BR-ordering, o ordinamento rosso-nero). Osserviamo ora un'importante proprietà: ad ogni passo, il valore aggiornato di ogni componente, indipendentemente dal colore, dipende esclusivamente dal valore dei vicini, che sono necessariamente del colore opposto. In particolare, preso uno dei due sottoinsiemi, i valori aggiornati di ciascuna delle componenti di questo sottoinsieme non dipende dalle altre componenti dello stesso sottoinsieme. Quindi i valori delle componenti rosse possono essere calcolati in ordine

indipendente tra le componenti rosse e i valori delle componenti nere possono essere calcolati in ordine indipendente tra le componenti nere. Possiamo scrivere quindi un nuovo algoritmo composto da due cicli for consecutivi, dove ogni ciclo for può potenzialmente svolgere tutte le sue iterazioni in parallelo. Per Gauss-Seidel, l'algoritmo è:

for $(i,j) \in \{\text{componenti rosse}\}$ **do**
$v_{i,j}^{(m+1)} = \frac{1}{4}(v_{i-1,j}^{(m)} + v_{i+1,j}^{(m)} + v_{i,j-1}^{(m)} + v_{i,j+1}^{(m)} + h^2 f_{i,j})$
end for

for $(i,j) \in \{\text{componenti nere}\}$ **do**
$v_{i,j}^{(m+1)} = \frac{1}{4}(v_{i-1,j}^{(m+1)} + v_{i+1,j}^{(m+1)} + v_{i,j-1}^{(m+1)} + v_{i,j+1}^{(m+1)} + h^2 f_{i,j})$
end for

Per il metodo di rilassamento SOR, invece, abbiamo:

for $(i,j) \in \{\text{componenti rosse}\}$ **do**
$v_{i,j}^{(m+1)} = (1-\omega)v_{i,j}^{(m)} + \frac{\omega}{4}(v_{i-1,j}^{(m)} + v_{i+1,j}^{(m)} + v_{i,j-1}^{(m)} + v_{i,j+1}^{(m)} + h^2 f_{i,j})$
end for

for $(i,j) \in \{\text{componenti nere}\}$ **do**
$v_{i,j}^{(m+1)} = (1-\omega)v_{i,j}^{(m)} + \frac{\omega}{4}(v_{i-1,j}^{(m+1)} + v_{i+1,j}^{(m+1)} + v_{i,j-1}^{(m+1)} + v_{i,j+1}^{(m+1)} + h^2 f_{i,j})$
end for

Facciamo ora delle considerazioni sulla velocità di convergenza dei tre metodi analizzati.
Per il metodo di Jacobi, scomponiamo la matrice $T_{N\times N}$ in

$$T_{N\times N} = 4I_N - (4I_N - T_{N\times N})$$

Allora la matrice di iterazione ottenuta per il metodo di Jacobi è

$$P_J = (4I_N)^{-1}(4I_N - T_{N\times N}) = I - \frac{1}{4}T_{N\times N}$$

In particolare, gli autovalori della matrice P_J sono tutti della forma $1 - \frac{\lambda_{i,j}}{4}$ dove

$$\lambda_{i,j} = \lambda_i + \lambda_j = 4 - 2\left(\cos\frac{i\pi}{N+1} + \cos\frac{j\pi}{N+1}\right)$$

Allora, per N grande, vale

$$\begin{aligned}\rho(P_J) &= \max_{i,j}\left|\frac{1}{2}\left(\cos\frac{i\pi}{N+1} + \cos\frac{j\pi}{N+1}\right)\right| = \\ &= \cos\frac{\pi}{N+1} \approx \\ &\approx 1 - \frac{\pi^2}{2(N+1)^2}\end{aligned}$$

Osserviamo che quindi, per $N \to \infty$, $\rho(P_J) \to 1$.

Per confrontare la velocità di convergenza dei tre metodi, fissiamo e^{-1} come soglia di precisione arbitraria e vediamo quante iterazioni sono necessarie per ciascun metodo per ridurre l'errore al di sotto di e^{-1}. Un modo per stimare il numero di iterazioni necessarie per ridurre l'errore al di sotto di una certa soglia è prendere l'm più piccolo per cui il valore $\rho(P)^m$ è sotto la soglia scelta, dove P è la matrice di iterazione del metodo considerato.
Per Jacobi abbiamo che

$$\begin{aligned}
\rho(P_J)^m &\approx e^{-1} \\
&\Downarrow \\
\left(1 - \frac{\pi^2}{2(N+1)^2}\right)^m &\approx e^{-1} \\
&\Downarrow \\
m &\approx \frac{-1}{\ln 1 - \frac{\pi^2}{2(N+1)^2}} \approx \\
&\approx \frac{-1}{-\frac{\pi^2}{2(N+1)^2}} = \\
&= \frac{2(N+1)^2}{\pi^2} = \\
&= O(N^2) = \\
&= O(n)
\end{aligned}$$

dove $n = N^2$ corrisponde al numero di incognite. In particolare, supponendo che per ogni iterazione aggiornare una componente costi $O(1)$, poiché ogni iterazione deve aggiornare n componenti e poiché il numeri di iterazioni per portare l'errore vicino a e^{-1} è $O(n)$, il costo totale per portare tutto il vettore ad un errore sotto la soglia è di $O(n^2)$.

Per Gauss-Seidel sappiamo che $\rho(P_{GS}) = \rho(P_J)^2$. In particolare il numero di iterazioni necessario per portare l'errore sotto la soglia di e^{-1} con Gauss-Seidel è la metà del numero di iterazioni necessarie con Jacobi. Quindi in questo caso, sfruttando i conti fatti prima

$$m \approx \frac{(N+1)^2}{\pi^2}$$

Anche in questo caso quindi, nonostante sia comunque più conveniente di Jacobi, il numero di iterazioni da svolgere per portare l'errore sotto la soglia è $O(n)$, quindi il costo totale è $O(n^2)$.

Per il metodo di rilassamento SOR, invece, il costo totale è migliore.

La scelta di ω ottimale si ha per

$$\omega_0 = \frac{2}{1 + \sin \frac{\pi}{N+1}}$$

Per questa scelta otteniamo([16])

$$\rho(P_{SORopt}) = \frac{\left(\cos \frac{\pi}{N+1}\right)^2}{\left(1 + \sin \frac{\pi}{N+1}\right)} \approx$$
$$\approx 1 - \frac{2\pi}{N+1}$$

In particolare otteniamo che il numero di iterazioni da svolgere per portare l'errore sotto la soglia è $O(N) = O(\sqrt{n})$. Quindi abbiamo che il costo totale dell'algoritmo è $O(n^{\frac{3}{2}})$

[16] Il motivo per cui la scelta ottimale di ω sia ω_0 definito come sopra e che il raggio spettrale di P_{SORopt} sia uguale alla frazione riportata è spiegato nella parte finale del capitolo 6.5.5 dell Demmel. La dimostrazione non è stata fatta in classe ed è un po' troppo lunga per riportarla

16 11/11/2020

16.1 Metodi del gradiente

Introduciamo ora un'altra famiglia di metodi iterativi per la risoluzione di $Ax = b$ dove A è una matrice reale definita positiva.

Sia $A \in \mathbb{R}^{n\times n}, b \in \mathbb{R}^n$. Per calcolare iterativamente vettore $x = A^{-1}b$ senza conoscere esplicitamente A^{-1} (che esiste perché definita positiva), definiamo il seguente funzionale

$$\Phi(x) = \frac{1}{2}x^T Ax - b^T x$$

Derivando questo funzionale otteniamo

$$\nabla\Phi(x) = \begin{pmatrix} \frac{\partial\Phi(x)}{\partial x_1} & \dots & \frac{\partial\Phi(x)}{\partial x_n} \end{pmatrix}^T = Ax - b$$

In particolare otteniamo che $Ax = b \iff \nabla\Phi(x) = 0$, quindi il problema originale può essere tradotto in un problema di ricerca di punto stazionario per $\Phi(x)$. In particolare, poiché la matrice è definita positiva, il punto stazionario è un punto di minimo globale([17]).

Descriviamo ora un metodo dipendente da una scelta che genererà una famiglia di metodi iterativi, le cui possibili implementazioni discuteremo poi. Per comodità poniamo $r(x) := b - Ax$ il vettore residuo. Osserviamo che $r(x) = -\nabla\Phi(x)$.
Sappiamo che $Ax^* = b \iff \Phi(x^*) = \min_{x\in\mathbb{R}^n}\Phi(x)$. Dato un vettore x_k al passo k-esimo, vediamo come ottenere un vettore x_{k+1} per ottenere una successione di vettori convergenti a x^*.
Scegliamo p_k una direzione di decrescita per Φ in x_k. Un vettore p_k è una direzione di decrescita per Φ in x_k se vale

$$p_k^T\nabla\Phi(x_k) < 0$$

Chiaramente questo vettore non è unico. Dalla scelta di questo vettore dipenderà il tipo di metodo che otterremo. Scelto p_k, poniamo $x_{k+1} = x_k + \alpha_k p_k$ dove α_k è scelto in modo tale che

$$\Phi(x_k + \alpha_k p_k) = \min_{\alpha\in\mathbb{R}}\Phi(x_k + \alpha p_k)$$

Sostanzialmente, per x_{k+1} prendiamo il punto di minimo di Φ nella retta di direzione p_k passante per x_k.
Per calcolare esplicitamente α_k, essendo un punto di minimo, deriviamo $\Phi(x_k + \alpha p_k)$ rispetto a α. Otteniamo

$$\frac{\partial}{\partial\alpha}\Phi(x_k + \alpha p_k) = (x_k + \alpha p_k)^T Ap_k - b^T p_k$$

[17] Non dimostrato in classe, metto la (breve) dimostrazione nelle appendici

Per ottenere α_k dobbiamo imporre che la derivata in α_k si annulli, da cui otteniamo

$$\alpha_k = \frac{(b - Ax_k)^T p_k}{p_k^T A p_k} = \frac{r_k^T p_k}{p_k^T A p_k}$$

dove abbiamo posto per semplicità di notazione $r_k := r(x_k)$. Ricordandoci che $r(x) = -\nabla\Phi(x)$, che $p_k^T \nabla\Phi(x_k) < 0$ e che $p_k^T A p_k > 0$ (poiché A è definita positiva) otteniamo

$$\alpha_k = \frac{r_k^T p_k}{p_k^T A p_k} > 0$$

La successione degli x_k così definita converge a x^*.

Osservazione.

$$\begin{aligned} r_{k+1} &= b - Ax_{k+1} = \\ &= b - A(x_k + \alpha_k p_k) = \\ &= b - Ax_k + A\alpha_k p_k = \\ &= r_k - \alpha_k A p_k \end{aligned}$$

da cui otteniamo

$$\begin{aligned} r_{k+1}^T p_k &= (r_k - \alpha_k A p_k)^T p_k = \\ &= r_k^T p_k - \alpha_k A p_k^T p_k = \\ &= r_k^T p_k - r_k^T p_k = \\ &= 0 \end{aligned}$$

Quindi ad ogni passo il residuo è ortogonale alla direzione del passo precedente.

16.2 Steepest Descent

In base a come scegliamo p_k otteniamo dei metodi iterativi diversi. Un metodo classico, detto "steepest descent", si ottiene prendendo $p_k = r_k$([18]). In questo caso otteniamo $p_{k+1}^T p_k = 0$, ovvero ogni direzione è ortogonale alla direzione precedente.

Per studiare la velocità di convergenza di questo metodo, consideriamo $e_k := x^* - x_k$ l'errore al passo k-esimo. Detti $\lambda_{\max}$ e $\lambda_{\min}$ gli autovalori di modulo massimo e minimo di A, mostriamo che vale

$$e_{k+1}^T A e_{k+1} \leq \left(\frac{\lambda_{\max} - \lambda_{\min}}{\lambda_{\max} + \lambda_{\min}}\right)^2 e_k^T A e_k$$

[18] $p_k = r_k$ è una scelta valida poiché verifica la richiesta su p_k, ovvero $p_k^T \nabla\Phi(x_k) = r_k^T \nabla\Phi(x_k) = (-\nabla\Phi(x_k))^T \nabla\Phi(x_k) = -((\nabla\Phi(x_k))^T \nabla\Phi(x_k)) < 0$

Infatti, cominciamo osservando che

$$\begin{aligned} e_{k+1} &= x^* - x_{k+1} = \\ &= x^* - (x_k + \alpha_k r_k) = \\ &= e_k - \alpha_k r_k = \\ &= e_k - \frac{r_k^T r_k}{r_k^T A r_k} r_k \end{aligned}$$

Da ciò otteniamo che

$$\begin{aligned} e_{k+1}^T A e_{k+1} &= \left(e_k - \frac{r_k^T r_k}{r_k^T A r_k} r_k \right)^T A \left(e_k - \frac{r_k^T r_k}{r_k^T A r_k} r_k \right) = \\ &= e_k^T A e_k - \frac{r_k^T r_k}{r_k^T A r_k} r_k^T A e_k - e_k^T A \frac{r_k^T r_k}{r_k^T A r_k} r_k + \frac{r_k^T r_k}{r_k^T A r_k} r_k^T A \frac{r_k^T r_k}{r_k^T A r_k} r_k \end{aligned}$$

Osserviamo ora che

$$A e_k = A(x^* - x_k) = A x^* - A x_k = b - A x_k = r_k$$

quindi possiamo sostituire e ottenere

$$\begin{aligned} e_{k+1}^T A e_{k+1} &= e_k^T A e_k - \frac{r_k^T r_k}{r_k^T A r_k} (r_k^T A e_k) - (e_k^T A) \frac{r_k^T r_k}{r_k^T A r_k} r_k + \left(\frac{r_k^T r_k}{r_k^T A r_k} \right)^2 r_k^T A r_k = \\ &= e_k^T A e_k - \frac{(r_k^T r_k)^2}{r_k^T A r_k} - \frac{(r_k^T r_k)^2}{r_k^T A r_k} + \frac{(r_k^T r_k)^2}{r_k^T A r_k} = \\ &= e_k^T A e_k - \frac{(r_k^T r_k)^2}{r_k^T A r_k} = \\ &= e_k^T A e_k \left(1 - \frac{(r_k^T r_k)^2}{(r_k^T A r_k)(e_k^T A e_k)} \right) = \\ &= e_k^T A e_k \left(1 - \frac{(r_k^T r_k)^2}{(r_k^T A r_k)(r_k^T A^{-1} r_k)} \right) \end{aligned}$$

dove l'ultima uguaglianza si ottiene osservando che

$$e_k^T A e_k = r_k^T A^{-1} A A^{-1} r_k = r_k^T A^{-1} r_k$$

Possiamo ora utilizzare la disuguaglianza di Kantorovich:

$$\forall x \in \mathbb{R}^n, \qquad \frac{(x^H x)^2}{(x^H A x)(x^H A^{-1} x)} \geq 4 \frac{\lambda_{\max} \lambda_{\min}}{(\lambda_{\max} + \lambda_{\min})^2}$$

dove $\lambda_{\max}, \lambda_{\min}$ sono gli autovalori di modulo massimo di A.
Sostituendo, otteniamo

$$\begin{aligned} e_{k+1}^T A e_{k+1} &= e_k^T A e_k \left(1 - \frac{(r_k^T r_k)^2}{(r_k^T A r_k)(r_k^T A^{-1} r_k)}\right) \leq \\ &\leq e_k^T A e_k \left(1 - 4\frac{\lambda_{\max}\lambda_{\min}}{(\lambda_{\max}+\lambda_{\min})^2}\right) = \\ &= e_k^T A e_k \left(\frac{\lambda_{\max}-\lambda_{\min}}{\lambda_{\max}+\lambda_{\min}}\right)^2 \end{aligned}$$

Per quantificare meglio questa velocità di convergenza, introduciamo la seguente norma vettoriale

$$\|x\|_A = \sqrt{x^T A x}$$

Ricordandoci che $\mu(A)_2 = \frac{\lambda_{\max}}{\lambda_{\min}}$ ricaviamo dalla precedente disuguaglianza il seguente risultato:

$$\|e_{k+1}\|_A \leq \left(\frac{\mu_2(A)-1}{\mu_2(A)+1}\right)\|e_k\|_A \leq \left(\frac{\mu_2(A)-1}{\mu_2(A)+1}\right)^{k+1}\|e_0\|_A$$

Quindi ad ogni iterazione la precisione aumenta di un fattore dipendente da $\mu_2(A)$. In particolare, se $\mu_2(A) \approx 1$ il metodo converge molto velocemente, mentre se $\mu_2(A) >> 1$ il metodo rischia di essere molto lento.
Nella prossima lezione vedremo un metodo che cerca di migliorare questo problema di convergenza lenta.

17 13/11/2020

17.1 Metodo del gradiente coniugato

Abbiamo visto il metodo di steepest descent come soluzione del problema $Ax = b$. Questo metodo era la specializzazione di un metodo del gradiente più generale, che dipendeva dalla scelta ad ogni iterazione della direzione di decrescita p_k.

Vediamo ora un altro modo per scegliere questa direzione, che genererà il metodo del gradiente coniugato. Prendiamo

$$p_k = \begin{cases} r_0 & \text{se } k = 0 \\ r_k + \beta_k p_{k-1} & \text{altrimenti} \end{cases}$$

dove β_k è preso in modo tale che p_k sia A-coniugato di p_{k-1}, ovvero che $p_k^T A p_{k-1} = 0$. Sostituendo la definizione di p_k nella richiesta appena fatta otteniamo

$$\beta_k = -\frac{r_k^T A p_{k-1}}{p_{k-1}^T A p_{k-1}}$$

Verifichiamo ora che un tale p_k sia effettivamente una direzione di decrescita. Utilizzando l'ortogonalità tra il residuo e la direzione del passo precedente (che vale ad ogni iterazione e che abbiamo dimostrato nella scorsa lezione), otteniamo

$$\begin{aligned} p_k^T \nabla \Phi(x_k) &= -p_k^T r_k = \\ &= -(r_k + \beta_k p_{k-1})^T r_k = \\ &= -r_k^T r_k - \beta_k \underbrace{p_{k-1}^T r_k}_{=0} = \\ &= -\|r_k\|_2^2 < \\ &< 0 \end{aligned}$$

Osserviamo che non può valere $\|r_k\|_2 = 0$ poiché implicherebbe $x_k = x^*$ e non ci sarebbe quindi bisogno di compiere un'altra iterazione. Da quest'ultima catena di equazioni si ricava anche $p_k^T r_k = r_k^T r_k$. In particolare, sostituendo nell'espressione generale per il valore di α_k ricavata nella scorsa lezione, otteniamo

$$\alpha_k = \frac{r_k^T r_k}{p_k^T A p_k}$$

Mostriamo ora che residui successivi sono ortogonali, che ci servirà per trovare

una formula più semplice per β_k.

$$\begin{aligned} r_k^T r_{k-1} &= r_k^T (p_{k-1} - \beta_{k-1} p_{k-2}) = \\ &= \underbrace{r_k^T p_{k-1}}_{=0} - \beta_{k-1} r_k^T p_{k-2} = \\ &= -\beta_{k-1} (r_{k-1} - \alpha_{k-1} A p_{k-1})^T p_{k-2} = \\ &= -\beta_{k-1} \underbrace{r_{k-1}^T p_{k-2}}_{=0} + \beta_{k-1} \alpha_{k-1} \underbrace{p_{k-1}^T A p_{k-2}}_{=0} = \\ &= 0 \end{aligned}$$

Scriviamo ora in modo più semplice β_k. Per farlo, scriviamo due sviluppi di $p_k^T r_{k-1}$:

$$\begin{aligned} p_k^T r_{k-1} &= (r_k + \beta_k p_{k-1})^T r_{k-1} = \\ &= \underbrace{r_k^T r_{k-1}}_{=0} + \beta_k p_{k-1}^T r_{k-1} = \\ &= \beta_k p_{k-1}^T r_{k-1} = \\ &= \beta_k r_{k-1}^T r_{k-1} \end{aligned}$$

$$\begin{aligned} p_k^T r_{k-1} &= p_k^T (r_k + \alpha_{k-1} A p_{k-1}) = \\ &= p_k^T r_k + \alpha_{k-1} \underbrace{p_k^T A p_{k-1}}_{=0} = \\ &= p_k^T r_k = \\ &= r_k^T r_k \end{aligned}$$

Da ciò si ricava facilmente che

$$\beta_k = \frac{r_k^T r_k}{r_{k-1}^T r_{k-1}}$$

17.2 Velocità di convergenza del metodo del gradiente coniugato

Facciamo ora delle considerazioni sul numero di iterazioni necessarie per ottenere una soluzione. In particolare, consideriamo il seguente teorema

Teorema 17.1. *Supponiamo di avere un h tale che tutte le iterazioni del metodo del gradiente, dalla prima alla h-esima, hanno residuo non nullo, ovvero sia h tale che $\forall 0 \leq k \leq h$, $r_k \neq 0$.*
Allora, per ogni $0 \leq j, k \leq h$ con $j \neq k$, vale

$$\begin{aligned} r_k^T r_j &= 0 \\ p_k^T A p_j &= 0 \end{aligned}$$

ovvero tutti i residui, dal primo all'h-esimo, sono a due a due ortogonali, e tutte le direzioni di decrescita, dalla prima all'h-esima, sono a due a due A-coniugate.

Osservazione. In particolare ciò ci garantisce che, lavorando in modo puramente teorico e non considerando le limitazioni di macchina, in al più n passi si ottiene la soluzione esatta x^*. Se così non fosse, infatti, avremmo i primi n residui non nulli e avremmo, per il teorema, $n+1$ vettori a due a due ortogonali in $\mathbb{R}^n$, che è assurdo.

Dimostrazione. Dimostriamolo per induzione su h.
Sia $h = 1$. Allora dobbiamo solo verificare che

$$r_1^T r_0 = 0$$
$$p_1^T A p_0 = 0$$

ma queste sono vere per costruzione.
Supponiamo ora che la tesi sia vera per h e dimostriamo che vale per $h+1$. Se le ipotesi valgono per $h+1$ in particolare valgono anche per h, quindi per ipotesi induttiva possiamo applicare il teorema sui primi h passi iterativi. Sappiamo quindi che

$$r_k^T r_j = 0$$
$$p_k^T A p_j = 0$$

per $0 \leq j, k \leq h$ con $j \neq k$. Per completare il passo induttivo ci rimane solo da dimostrare che

$$r_{h+1}^T r_j = 0$$
$$p_{h+1}^T A p_j = 0$$

per $0 \leq j \leq h$. Inoltre, come visto nel passo base, il caso $j = h$ è vero per costruzione. Dobbiamo quindi dimostrare le equazioni per $0 \leq j \leq h-1$.
Utilizzando ripetutamente l'ipotesi induttiva otteniamo che([19])

$$\begin{aligned} r_{h+1}^T r_j &= (r_h - \alpha_h A p_h)^T r_j = \\ &= \underbrace{r_h^T r_j}_{=0} - \alpha_h p_h^T A r_j = \\ &= -\alpha_h p_h^T A (p_j - \beta_j p_{j-1}) = \\ &= -\alpha_h \underbrace{p_h^T A p_j}_{=0} - \beta_j \alpha_h \underbrace{p_h^T A p_{j-1}}_{=0} \end{aligned}$$

[19] Questi conti, anche se non formalmente corretti, funzionano anche per $j = 0$ anche se compare il vettore p_{j-1}. Infatti, per come è definito p_0, avremmo $r_0 = p_0$ senza alcuna componente p_{-1} (che non esisterebbe)

Per quanto riguarda $p_{h+1}^T A p_j$, riscriviamo prima $Ap_j = \frac{1}{\alpha_j}(r_j - r_{j+1})$. A questo punto sviluppiamo

$$\begin{aligned} p_{h+1}^T A p_j &= (r_{h+1} + \beta_{h+1} p_h)^T A p_j = \\ &= r_{h+1}^T A p_j + \beta_{h+1} \underbrace{p_h^T A p_j}_{=0} = \\ &= \frac{1}{\alpha_j} r_{h+1}^T (r_j - r_{j+1}) = \\ &= \frac{1}{\alpha_j} \underbrace{r_{h+1}^T r_j}_{=0} - \frac{1}{\alpha_j} r_{h+1}^T r_{j+1} = \\ &= -\frac{1}{\alpha_j} r_{h+1}^T r_{j+1} \end{aligned}$$

Per $0 \leq j \leq h-2$ quel prodotto è nullo per quanto dimostrato prima, e per $j = h-1$ quel prodotto diventa $r_{h+1}^T r_h$ che è nullo per costruzione. □

Abbiamo quindi un teorema che dimostra che in al più n passi si ottiene la soluzione al problema, in assenza di limitazioni di macchina. Ciò non è vero, però, nel caso in cui si implementi questo algoritmo con precisione finita. Per questo motivo questo algoritmo viene considerato un algoritmo iterativo per approssimare x^*, e non come un algoritmo finito per ottenere la soluzione esatta x^*.
Come criterio di arresto usiamo $\|r_k\|_2 < \varepsilon \|b\|_2$.

Implementiamo ora in pseudocodice l'algoritmo appena visto:

$k = 0,\ x_0 = 0,\ r_0 = b,\ p_0 = r_0,\ v_0 = r_0^T r_0,\ tol = \varepsilon v_0$
while $v_k \geq tol$ **do**
  $w = Ap_k$
  $\alpha_k = v_k / p_k^T w$
  $x_{k+1} = x_k + \alpha_k p_k$
  $r_{k+1} = r_k - \alpha_k w$
  $v_{k+1} = r_{k+1}^T r_{k+1}$
  $\beta_{k+1} = v_{k+1} / v_k$
  $p_{k+1} = r_{k+1} + \beta_{k+1} p_k$
  $k = k + 1$
end while

L'operazione più costosa del ciclo while è una moltiplicazione matrice-vettore, che in generale è un $O(n^2)$. Poiché per ottenere la soluzione esatta sono necessarie n iterazioni, l'algoritmo in generale è un $O(n^3)$. Nel caso in cui A sia sparsa e sia nota la struttura di sparsità, però, il costo del prodotto matrice-vettore diminuisce notevolmente. Inoltre, spesso il numero di iterazioni necessarie per

ridurre la norma del residuo sotto la tolleranza considerata è molto più piccolo di n, quindi non sono necessarie tutte e n le iterazioni.

Si può dimostrare che, in modo analogo a quanto visto per lo steepest descent, è possibile stimare la norma dell'errore al passo k-esimo con la potenza di un parametro dipendente da $\mu_2(A)$. In particolare, detta $\|x\|_A = \sqrt{x^T Ax}$, vale

$$\|e_k\|_A \leq \left(\frac{\sqrt{\mu_2(A)} - 1}{\sqrt{\mu_2(A)} + 1} \right)^k \|e_0\|_A$$

In questo caso, quindi, se $\mu_2(A) \approx 1$ il numero di passi necessari sarà significativamente minore di n. Se invece $\mu_2(A) >> 1$, il numero di passi necessari sarà molto vicino ad n.

17.3 Tecniche di precondizionamento

Data un problema $Ax = b$, esistono delle tecniche di trasformazione del problema atte a migliorare il condizionamento della matrice dei coefficienti del problema considerato, lasciando invariate le soluzioni.

L'idea di base delle tecniche di precondizionamento è quella di trovare due matrici, C_1 e C_2 invertibili e di forma "semplice"([20]) tali che la matrice $B := C_1^{-1} A C_2^{-1}$ sia meglio condizionata di A. A questo punto, invece di risolvere il sistema $Ax = b$ è possibile risolvere

$$\underbrace{C_1^{-1} A C_2^{-1}}_{=:B} \underbrace{C_2 x}_{=:y} = \underbrace{C_1^{-1} b}_{=:c}$$

le cui soluzioni sono le stesse del sistema originale.
Se $C_2 = I$, si parla di precondizionamento a sinistra. Se $C_1 = I$, si parla di precondizionamento a destra.
La matrice $M := C_1 C_2$ si chiama precondizionatore.

17.4 Metodo del gradiente coniugato con precondizionamento

E' possibile applicare tecniche di precondizionamento per migliorare l'efficienza del metodo del gradiente coniugato. Per il gradiente coniugato si tende a considerare $C_2 = C_1^T$ (e al posto di parlare di C_1 e C_2 si parla di C e C^T). In questo caso è possibile descrivere completamente il metodo di precondizionamento solo con la matrice M, necessariamente reale e definita positiva, da cui è possibile ricavare C tramite la fattorizzazione di Cholesky.

Alcuni metodi usano come precondizionatore la matrice $M := \text{diag}(a_{1,1}, \ldots, a_{n,n})$.

[20] Ad esempio diagonali, triangolari, sparse con struttura di sparsità nota, a banda...

Un'altra scelta di possibile precondizionamento, nel caso di A sparsa con struttura di sparsità nota, si basa sulla fattorizzazione di Cholesky di A. Data la fattorizzazione $A = LL^T$, si considera la matrice $C = (c_{i,j})$ così definita

$$c_{i,j} = \begin{cases} l_{i,j} & \text{se } a_{i,j} \neq 0 \\ 0 & \text{altrimenti} \end{cases}$$

Supponiamo ora di aver trasformato il problema originale $Ax = b$ in $By = c$ tramite matrici C, C^T, e vediamo come modificare l'algoritmo scritto precedentemente.
Definiamo il "nuovo" vettore residuo

$$\begin{aligned} s_k := c - By_k = & \\ & = C^{-1}b - C^{-1}AC^{-T}x_k = \\ & = C^{-1}r_k \end{aligned}$$

A questo punto vale

$$s_k^T s_k = r_k^T C^{-T} C^{-1} r_k = r_k M^{-1} r_k$$

Per calcolare $s_k^T s_k$ dovremmo quindi calcolare l'inversa della matrice M per poi moltiplicarla per un vettore. In generale, in casi come questo è più conveniente risolvere il sistema lineare $Mz_k = r_k$ e utilizzare direttamente $z_k = M^{-1}r_k$ invece di calcolare esplicitamente M^{-1} per poi moltiplicarla per r_k([21]).
A questo punto riscriviamo $s_k^T s_k = r_k^T z_k$ dove z_k è la soluzione del sistema lineare $Mz_k = r_k$.
L'algoritmo diventa:

$k = 0,\ x_0 = 0,\ r_0 = b,\ z_0$ t.c. $Mz_0 = r_0$
$p_0 = r_0,\ v_0 = r_0^T r_0,\ tol = \varepsilon v_0$
while $v_k \geq tol$ **do**
  $w = Ap_k$
  $\alpha_k = v_k / p_k^T w$
  $x_{k+1} = x_k + \alpha_k p_k$
  $r_{k+1} = r_k - \alpha_k w$
  calcolo z_{k+1} t.c. $Mz_{k+1} = r_{k+1}$
  $v_{k+1} = r_{k+1}^T z_{k+1}$
  $\beta_{k+1} = v_{k+1} / v_k$
  $p_{k+1} = C^{-1} z_{k+1} + \beta_{k+1} p_k$
  $k = k + 1$

[21] Può sembrare controintuitivo cercare di risolvere molteplici sistemi lineari $Mz_k = r_k$ per evitare di dover risolvere esplicitamente il sistema $Ax = b$. In realtà, data la "semplicità" (vedi nota precedente) della forma di C, risolvere $Mz_k = r_k$ è considerevolmente più semplice di $Ax = b$

end while

Osservazione. L'operazione più costosa della singola iterazione è la risoluzione del sistema lineare $Mz_{k+1} = r_{k+1}$, che è più costosa del prodotto matrice-vettore del metodo del gradiente coniugato semplice. In generale, quindi, la singola iterazione del metodo del gradiente coniugato con precondizionamento è più costosa della singola iterazione del metodo del gradiente coniugato semplice. L'abbassamento del numero di condizionamento, però, permette di diminuire il numero di iterazioni necessarie, risultando quindi più efficiente.

17.5 Gradiente coniugato per problemi rettangolari

E' possibile adattare il metodo del gradiente coniugato per la risoluzione di problemi $Ax = b$ dove A non è una matrice quadrata, ovvero per risolvere il problema dei minimi quadrati.

Data $A \in \mathbb{R}^{m\times n}$ con $m \geq n$ di rango massimo, risolvere $Ax = b$ significa trovare x che minimizzi $\|Ax - b\|_2$. Riscriviamo quindi questa norma per un generico vettore y

$$\begin{aligned}\|Ay - b\|_2^2 &= (Ay - b)^T(Ay - b) = \\ &= y^T A^T Ay - b^T Ay - y^T A^T b + b^T b = \\ &= y^T A^T Ay - 2b^T Ay + b^T b = \\ &= 2(\frac{1}{2} y^T A^T Ay - b^T Ay) + b^T b\end{aligned}$$

Posta $\Phi(y) := \frac{1}{2} y^T A^T Ay - b^T Ay$, minimizzare $\Phi(y)$ equivale a minimizzare $\|Ay - b\|_2$. Possiamo quindi applicare il metodo del gradiente coniugato con Φ come appena definita.

Abbiamo

$$\nabla\Phi(y) = -A^T(b - Ay) = -A^T r(y)$$

da cui

$$-\nabla\Phi(x_k) = A^T r_k =: s_k$$

Costruiamo ora $x_{k+1} = x_k + \alpha_k p_k$, dove p_k direzione di decrescita, in modo che sia il minimo di Φ sulla retta di direzione p_k passante per x_k. Procedendo in modo analogo a quanto visto precedentemente, otteniamo

$$\alpha_k = \frac{p_k^T s_k}{\|Ap_k\|_2^2}$$

Vale quindi $r_{k+1} = r_k - \alpha_k A p_k$. Per quanto riguarda la direzione di decrescita al passo k-esimo, scegliamo

$$p_k = \begin{cases} s_0 & \text{se } k = 0 \\ s_k + \beta_k p_{k-1} & \text{altrimenti} \end{cases}$$

con β_k tale che $p_k^T(A^TA)p_{k-1} = 0$. Analogamente a quanto fatto per il metodo nel caso quadrato, otteniamo

$$\beta_k = -\frac{s_k^T A^T A p_{k-1}}{p_{k-1}^T A^T A p_{k-1}}$$

E' possibile trovare delle formule più semplici per α_k e β_k.
Per α_k, cominciamo osservando che

$$\begin{aligned} p_{k-1}^T s_k &= p_{k-1}^T A^T r_k = \\ &= p_{k-1}^T A^T (r_{k-1} - \alpha_{k-1} A p_{k-1}) = \\ &= p_{k-1}^T A^T r_{k-1} - \alpha_{k-1} p_{k-1}^T A^T A p_{k-1} = \\ &= p_{k-1}^T s_{k-1} - \alpha_{k-1} \|A p_{k-1}\|_2^2 = \\ &= 0 \end{aligned}$$

dove l'ultima uguaglianza segue dalla definizione di α_{k-1}. Segue che

$$\begin{aligned} p_k^T s_k &= (s_k + \beta_k p_{k-1})^T s_k = \\ &= s_k^T s_k + \beta_k \underbrace{p_{k-1}^T s_k}_{=0} = \\ &= s_k^T s_k = \\ &= \|s_k\|_2^2 \end{aligned}$$

Allora possiamo sostituire nella definizione di α_k e ottenere

$$\alpha_k = \frac{p_k^T s_k}{\|A p_k\|_2^2} = \frac{\|s_k\|_2^2}{\|A p_k\|_2^2}$$

Per β_k, utilizzando $s_k^T s_{k-1} = 0$([22]), partiamo dalla formula che già abbiamo:

$$\begin{aligned}
\beta_k &= -\frac{s_k^T A^T A p_{k-1}}{p_{k-1}^T A^T A p_{k-1}} = \\
&= -s_k^T A^T A p_{k-1} \frac{1}{\|A p_{k-1}\|_2^2} = \\
&= s_k^T A^T A p_{k-1} \frac{-\alpha_{k-1}}{\|s_{k-1}\|_2^2} = \\
&= \frac{s_k^T A^T (-\alpha_{k-1} A p_{k-1})}{\|s_{k-1}\|_2^2} = \\
&= \frac{s_k^T A^T (r_k - r_{k-1})}{\|s_{k-1}\|_2^2} = \\
&= \frac{s_k^T A^T r_k}{\|s_{k-1}\|_2^2} - \frac{s_k^T A^T r_{k-1}}{\|s_{k-1}\|_2^2} = \\
&= \frac{s_k^T s_k}{\|s_{k-1}\|_2^2} - \frac{s_k^T s_{k-1}}{\|s_{k-1}\|_2^2} = \\
&= \frac{s_k^T s_k}{\|s_{k-1}\|_2^2} = \\
&= \frac{\|s_k\|_2^2}{\|s_{k-1}\|_2^2}
\end{aligned}$$

[22] Da dimostrare come esercizio

18 20/11/2020

18.1 Metodi di Krylov

(Non c'è stata lezione il 18/11/2020)
Vediamo ora un'ultima famiglia di metodi iterativi per la risoluzione di sistemi lineari $Ax = b$.
Troviamo un modo per riscrivere il sistema lineare in modo da renderne più facile il calcolo della soluzione cercata. Partendo da $y_1 := b$, definiamo $y_i := Ay_{i-1}$ per $i = 2, ..., n$, e definiamo la matrice $K = \left(y_1 \mid \dots \mid y_{n-1} \mid y_n\right)$ che ha per colonne i vettori appena definiti. Osserviamo che per come abbiamo definito K e i vettori y_i, vale

$$\begin{aligned} AK &= A\left(y_1 \mid \dots \mid y_{n-1} \mid y_n\right) = \\ &= \left(Ay_1 \mid \dots \mid Ay_{n-1} \mid Ay_n\right) = \\ &= \left(y_2 \mid \dots \mid y_n \mid A^n y_1\right) \end{aligned}$$

Ricordandoci che $y_i = Ke_i$ (dove con e_i intendiamo l'i-esima colonna dell'identità), e definendo un nuovo vettore $c := K^{-1}A^n y_1$, possiamo riscrivere

$$\begin{aligned} AK &= \left(y_2 \mid \dots \mid y_n \mid A^n y_1\right) = \\ &= \left(Ke_2 \mid \dots \mid Ke_n \mid -Kc\right) = \\ &= K\left(e_2 \mid \dots \mid e_n \mid -c\right) =: \\ &=: KC \end{aligned}$$

In particolare([23])

$$K^{-1}AK = C = \begin{pmatrix} 0 & & & -c_1 \\ 1 & \ddots & & \vdots \\ & \ddots & 0 & \vdots \\ & & 1 & -c_n \end{pmatrix}$$

Tornando al sistema $Ax = b$, poiché vale $Ke_1 = b$, possiamo riscrivere il sistema come

$$\underbrace{K^{-1}AK}_{C}\underbrace{K^{-1}x}_{\tilde{x}} = K^{-1}b = e_1$$

Risolvere questo nuovo sistema significa risolvere

$$\begin{aligned} -c_1\tilde{x}_n &= 1 \\ \tilde{x}_1 - c_2\tilde{x}_n &= 0 \\ &\vdots \\ \tilde{x}_{n-1} - c_{n-1}\tilde{x}_n &= 0 \end{aligned}$$

[23] Una matrice C di questa forma si chiama "companion matrix" o matrice compagna

facilmente risolvibili. Per ottenere la soluzione del sistema originale basta poi calcolare semplicemente $x = K\tilde{x}$.

Il metodo per come è stato definito adesso, però, comporta due grossi problemi:

- Calcolare il vettore c comporta, di fatto, risolvere il sistema lineare $Kc = A^n y_1$, che non ha nessun motivo di essere più facile da risolvere rispetto al problema iniziale (anzi, nel caso di A sparsa, K è molto probabilmente non sparsa, quindi questo nuovo sistema lineare potrebbe essere perfino più difficile del problema iniziale)
- La matrice K potrebbe essere molto mal condizionata. Infatti, le colonne di K sono ottenute iterativamente moltiplicando la colonna attuale per una matrice fissata per ottenere la colonna successiva. Per quanto visto nel metodo delle potenze, sappiamo che per N grande queste colonne convergono all'autovalore relativo all'autovettore di modulo massimo, ovvero diventano sempre più parallele, cosa che comporta numero di condizionamento alto([24]).

Per ovviare a questi problemi la famiglia di metodi che stiamo per descrivere, che prendono il nome metodi di Krylov, utilizzano al posto di K una matrice Q ortogonale costruita in modo che per ogni $k = 1, ..., n$, le prime k colonne della matrice Q siano una base dello spazio generato dalle prime k colonne di K.

Definizione. Lo spazio generato dalle prime k colonne di K sarà della forma $\text{Span}(y_1, Ay_1, \ldots, A^{k-1}y_1)$. Uno spazio di questa forma si chiama spazio di $Krylov$ e si indica con $\mathcal{K}_k(A, y_1)$

Una tale matrice Q risolve i problemi dell'implementazione precedente. Infatti, per risolvere un sistema lineare che abbia lei come coefficienti basta usare la sua inversa che corrisponde con la sua trasposta. Inoltre, è sicuramente ben condizionata perché per Q ortogonale vale $\mu_2(Q) = 1$.

La matrice Q che considereremo sarà quella ottenibile dalla fattorizzazione QR di $K = QR$. Vediamo ora come poter calcolare Q senza dover calcolare esplicitamente K per poi doverne trovare una fattorizzazione QR. Cominciamo osservando che

$$\begin{gathered} C = K^{-1}AKR^{-1}Q^TAQR \\ \Downarrow \\ Q^TAQ = RCR^{-1} \end{gathered}$$

In particolare poiché R, R^{-1} sono triangolari superiori e C è in forma di Hessenberg superiore, anche Q^TAQ è in forma di Hessenberg superiore. Definiamo $H := RCR^{-1} = Q^TAQ$

[24] Non so davvero perché comporti un numero di condizionamento alto. Se qualcuno sa spiegarmelo lo aggiungo

Osservazione. Osserviamo che se A è simmetrica, anche H sarebbe simmetrica. In particolare, poiché sappiamo già che è in forma di Hessenberg, H è tridiagonale.
Tra non molto faremo qualche approfondimento del caso A simmetrica (o simmetrica definita positiva), e in questo caso chiameremo la matrice $T := H$ per ricordarci che è tridiagonale e non solo in forma di Hessenberg.

Chiamiamo le colonne di $Q = \begin{pmatrix} q_1 & | & \dots & | & q_n \end{pmatrix}$ e vediamo come poterle ricavare ricorsivamente questi vettori senza dover passare per K. Vale

$$Q^T AQ = H \iff AQ = QH$$

Leggendo l'ultima equazione sulla j-esima colonna, sfruttando la forma di H, otteniamo:

$$Aq_j = \sum_{i=1}^{j+1} q_i h_{i,j}$$

Allora, per $1 \leq m \leq j$, vale

$$q_m^T Aq_j = \sum_{i=1}^{j+1} q_m^T q_i h_{i,j} = h_{m,j}$$

poiché essendo le colonne di Q ortogonali tra loro e normalizzate, il prodotto $q_m^T q_i$ corrisponde al delta di Kronecker. Abbiamo quindi una strategia per calcolare gli $h_{i,j}$. Per calcolare i vettori, invece, possiamo calcolari iterativamente notando che, per trovare il $j+1$-esimo vettore, vale

$$h_{j+1,j} q_{j+1} = Aq_j - \sum_{i=1}^{j} q_i h_{i,j}$$

Notiamo che anche se non abbiamo modo di calcolare $h_{j+1,j}$ senza avere già il vettore q_{j+1}, questo non costituisce un problema per trovare il vettore q_{j+1} utilizzando la formula appena trovata. Infatti sappiamo che q_{j+1} è normalizzato, quindi anche se lo troviamo a meno di scalare basta normalizzare il vettore ottenuto per ottenere q_{j+1}.
L'algoritmo per calcolare tutti i vettori è:

$q_1 = b/\|b\|_2$
for $j = 1$ to k **do**
  $z = Aq_j$
  for $i = 1$ to j **do**
    $h_{i,j} = q_i^T z$
    $z = z - h_{i,j} q_i$
  end for
  $h_{j+1,j} = \|z\|_2$
  $q_{j+1} = z/h_{j+1,j}$

end for

L'algoritmo utilizzato per calcolare Q si chiama algoritmo di Arnoldi. I vettori q_j vengono spesso chiamati vettori di Arnoldi.
Osserviamo che nel caso di A simmetrica e quindi T tridiagonale, l'algoritmo diventa l'algoritmo di Lanczos già visto in passato, che è quindi un caso particolare dell'algoritmo di Arnoldi.

Svolgendo n passi dell'algoritmo di Arnoldi otteniamo la matrice Q per poter risolvere agevolmente il sistema e trovare la soluzione cercata. Tuttavia, svolgere n passi è molto costoso. sarebbe comodo se, anche svolgendo $k < n$ passi, si riuscisse ad ottenere una soluzione approssimata sufficientemente vicina alla soluzione cercata. Facciamo ora delle considerazioni su che approssimazioni di x possiamo ottenere fermandoci al passo k-esimo dell'algoritmo di Arnoldi, che chiameremo x_k.
Fermandoci al passo k-esimo, partizioniamo la matrice Q in $Q = \left(Q_k \mid Q_n\right)$ dove $Q_k = \left(q_1 \mid \dots \mid q_k\right)$ e $Q_k = \left(q_{k+1} \mid \dots \mid q_n\right)$.

Osservazione. Ricordiamo che a questo passo, tutte le colonne di Q_k e q_{k+1} sono note, mentre tutte le colonne di Q_n tranne la prima non sono note, e sono quindi da considerare non utilizzabili per ottenere l'approssimazione x_k.

Con questa partizione, riscriviamo H:

$$H = Q^T AQ = \left(\begin{array}{c|c} Q_k^T AQ_k & Q_k^T AQ_n \\ \hline Q_n^T AQ_k & Q_n^T AQ_n \end{array}\right) =: \left(\begin{array}{c|c} H_k & H_{n,k} \\ \hline H_{k,n} & H_n \end{array}\right)$$

Poiché sappiamo che H è in forma di Hessenberg superiore, sicuramente anche H_k è in forma di Hessenberg superiore. $H_{k,n}$, invece, è quasi completamente nulla eccezion fatta per l'elemento in alto a destra non necessariamente nullo (per intenderci, l'elemento $h_{k+1,k}$ di H).
Notiamo che H_k e $H_{k,n}$ sono ricavabili con le informazioni che abbiamo, mentre $H_{n,k}$ e H_n no.

Osservazione. Se A è simmetrica anche T lo è, per cui anche $T_{n,k}$ è ricavabile (poiché basta trasporre $T_{k,n}$ che è ricavabile).

Dopo k passi, le colonne di Q_k formano una base dello spazio $\mathcal{K}_k(A, b)$. H_k (o T_k) viene detta la proiezione di A sullo spazio di Krylov. I metodi di Krylov cercano approssimazioni di x nello spazio $\mathcal{K}_k(A, b)$. In base a come viene scelta x_k si ottengono diversi metodi. Vediamo ora un teorema che mostra una possibile scelta al passo k-esimo di x_k.

Teorema 18.1. *Sia A simmetrica e sia $T_k = Q_k^T AQ_k$ ottenuta fermandosi al passo k-esimo dell'algoritmo di Arnoldi. Data una soluzione candidata $x_k \in \mathcal{K}_k(A, b)$, definiamo $r_k = b - Ax_k$ il residuo.*
Se T_k è non singolare, allora scegliendo $x_k = Q_k T_k^{-1} e_1 \|b\|_2$ si ha $Q_k^T r_k = 0$.
Se A è definita positiva, allora T_k è necessariamente non singolare e, oltre al

risultato precedente, si ha anche che questa scelta di x_k minimizza $\|r_k\|_{A^{-1}}$. Inoltre, sempre se A è definita positiva, vale $r_k = \pm\|r_k\|_2\, q_{k+1}$, ovvero r_k è parallelo a q_{k+1} (che ricordiamo essere normalizzato per norma 2).

Dimostrazione. Per quanto riguarda il primo risultato del teorema, se supponiamo T_k non singolare e scegliamo x_k come sopra, vale

$$\begin{aligned} Q_k^T r_k &= Q_k^T(b - Ax_k) = \\ &= \underbrace{Q_k^T b}_{=e_1\|b\|_2} - Q_k^T A x_k = \\ &= e_1\|b\|_2 - Q_k^T A(Q_k T_k^{-1} e_1\|b\|_2) = \\ &= e_1\|b\|_2 - T_k T_k^{-1} e_1\|b\|_2) = \\ &= 0 \end{aligned}$$

Consideriamo ora il caso di A definita positiva. In tal caso, anche T_k è definita positiva e in particolare non singolare.
Verifichiamo ora che tale scelta di x_k minimizza $\|r_k\|_{A^{-1}}$. Sia $\hat{x}_k \in \mathcal{K}_k(A,b)$ un altro possibile candidato come soluzione che minimizza $\|\hat{r}_k\|_{A^{-1}}$. Poiché sia x_k che $\hat{x}_k$ stanno nello span delle colonne di Q, possiamo esprimere $\hat{x}_k$ come $\hat{x}_k = x_k + Qz$ per un certo vettore z. Detto $\hat{r}_k := b - A\hat{x}_k$, basta mostrare che $\|\hat{r}_k\|_{A^{-1}}$ è minimo per $z = 0$. Infatti, osservando che

$$\hat{r}_k = b - A(x_k + Q_k z) = r_k - AQ_k z$$

otteniamo

$$\begin{aligned} \|\hat{r}_k\|_{A^{-1}}^2 &= \hat{r}_k^T A^{-1}\hat{r}_k = \\ &= (r_k - AQ_k z)^T A^{-1}(r_k - AQ_k z) = \\ &= r_k^T A^{-1} r_k - (AQ_k z)^T A^{-1} r_k - r_k^T A^{-1} AQ_k z + (AQ_k z)^T A^{-1} AQ_k z = \\ &= \|r_k\|_{A^{-1}} - 2(AQ_k z)^T A^{-1} r_k + \|AQ_k z\|_{A^{-1}} = \\ &= \|r_k\|_{A^{-1}} - 2z^T \underbrace{Q_k^T r_k}_{=0} + \|AQ_k z\|_{A^{-1}} = \\ &= \|r_k\|_{A^{-1}} + \|AQ_k z\|_{A^{-1}} \end{aligned}$$

e questo viene minimizzato se e solo se $z = 0$ (infatti viene minimizzato per $AQ_k z = 0$, con A invertibile e Q_k di rango massimo).
Per concludere, osserviamo che se $x_k \in \mathcal{K}_k(A,b)$ allora $r_k = b - Ax_k \in \mathcal{K}_{k+1}(A,b)$. Quindi r_k è combinazione lineare di $q_1, \ldots, q_{k+1}$. Poiché però $Q_k^T r_k = 0$, r_k è anche ortogonale a $q_1, \ldots, q_k$. In particolare, otteniamo che r_k appartiene allo span di q_{k+1}, e gli è quindi parallelo (e poiché q_{k+1} è normalizzato per norma 2, vale $r_k = \pm\|r_k\|_2\, q_{k+1}$). □

18.2 Gradiente coniugato come metodo di Krylov

E' possibile esprimere il metodo del gradiente coniugato come un caso particolare dei metodi di Krylov. Ricordiamo che il metodo del gradiente coniugato è un

metodo iterativo che parte da un sistema $Ax = b$ con A simmetrica definita positiva e che è completamente identificato dalle seguenti relazioni ricorsive:

$$x_k = \begin{cases} 0 & \text{se } k = 0 \\ x_{k-1} + \alpha_{k-1} p_{k-1} & \text{altrimenti} \end{cases}$$

$$p_k = \begin{cases} r_0 & \text{se } k = 0 \\ r_k + \beta_k p_{k-1} & \text{altrimenti} \end{cases}$$

$$\alpha_k = \frac{r_k^T r_k}{p_k^T A p_k}$$

$$\beta_k = \frac{r_k^T r_k}{r_{k-1}^T r_{k-1}}$$

Mostriamo ora che utilizzando il metodo di Krylov che sceglie, al passo k-esimo dell'algoritmo di Arnoldi, $x_k = Q_k T_k^{-1} e_1 \|b\|_2$ (come nel teorema precedente), il metodo generato è equivalente al metodo del gradiente coniugato. Per farlo, mostriamo che tra questi x_k esistono le stesse relazioni ricorsive degli x_k del metodo del gradiente coniugato. Per evitare confusione, d'ora in poi parleremo solo del metodo di Krylov: x_k si riferirà solo all'x_k del metodo di Krylov e non a quella del metodo del gradiente coniugato. Torneremo a parlare di quest'ultimo metodo alla fine quando mostreremo l'equivalenza.

Data A simmetrica definita positiva, sappiamo che anche T_k definita dal metodo di Krylov è definita positiva. In particolare, modificando leggermente la decomposizione di Cholesky, possiamo scomporla in

$$T_k = L_k D_k L_k^T$$

dove L_k è triangolare inferiore con elementi diagonali uguali a 1, e D_k è diagonale (con elementi diagonali positivi). Possiamo allora scrivere([25])

$$\begin{aligned} x_k &= Q_k T_k^{-1} e_1 \|b\|_2 = \\ &= Q_k (L_k D_k L_k^T)^{-1} e_1 \|b\|_2 = \\ &= \underbrace{Q_k L_k^{-T}}_{=:\tilde{P}_k} \underbrace{D_k^{-1} L_k^{-1} e_1 \|b\|_2}_{=:y_k} \end{aligned}$$

Chiamiamo $\tilde{p}_1, \ldots, \tilde{p}_k$ le colonne di $\tilde{P}_k = \left(\tilde{p}_1 \mid \ldots \mid \tilde{p}_k\right)$.

Proposizione 18.1. *Le colonne di $\tilde{P}_k$ sono A-coniugate.*

[25] La notazione è un po' infelice. Non si confondano gli y_k definiti adesso con gli $y_1, \ldots, y_n$ definiti nell'introduzione ai metodi di Krylov. Sono degli y_k diversi. D'ora in poi quando scriveremo y_k intenderemo quelli appena definiti.

Dimostrazione. Dimostrare la proposizione equivale a dimostrare che la matrice $\tilde{P}_k^T A \tilde{P}_k$ è diagonale. Allora osserviamo che

$$\begin{aligned}\tilde{P}_k^T A \tilde{P}_k &= (Q_k L_k^{-T})^T A (Q_k L_k^{-T}) = \\ &= L_k^{-1} Q_k^T A Q_k^T L_k^{-T} = \\ &= L_k^{-1} T_k L_k^{-T} = \\ &= D_k\end{aligned}$$

□

Cerchiamo ora di esplicitare una relazione ricorsiva per x_k per poterla poi confrontare con quella del metodo del gradiente coniugato. Per farlo, esprimiamo una relazione ricorsiva per y_k e $\tilde{P}_k$ con il vettore e la matrice ottenuti al passo $(k-1)$-esimo. Cominciamo osservando che possiamo scomporre T_k come

$$\begin{aligned}T_k &= L_k D_k L_k^T = \\ &= \begin{pmatrix} 1 & & & \\ l_1 & \ddots & & \\ & \ddots & \ddots & \\ & & l_{k-1} & 1 \end{pmatrix} \begin{pmatrix} d_1 & & \\ & \ddots & \\ & & d_k \end{pmatrix} \begin{pmatrix} 1 & l_1 & & \\ & \ddots & \ddots & \\ & & \ddots & l_{k-1} \\ & & & 1 \end{pmatrix} = \\ &= \left(\begin{array}{ccc|c} & L_{k-1} & & 0 \\ \hline l_{k-1}\hat{e}_{k-1}^T & & & 1 \end{array}\right) \left(\begin{array}{ccc|c} & D_{k-1} & & 0 \\ \hline 0 & & & d_k \end{array}\right) \left(\begin{array}{ccc|c} & L_{k-1} & & l_{k-1}\hat{e}_{k-1} \\ \hline 0 & & & 1 \end{array}\right)\end{aligned}$$

dove con $\hat{e}_{k-1}$ intendiamo la $(k-1)$-esima colonna della matrice identità $(k-1)\times(k-1)$ per distinguerlo dal vettore e_{k-1} con cui intendiamo la $(k-1)$-esima colonna della matrice identità $k \times k$. Riutilizzeremo questa notazione tra poco per esprimere colonne della matrice identità $(k-1)\times(k-1)$.
Osserviamo, ricordandoci che T_k è il k-esimo minore di testa della matrice T fissata, che le matrici L_{k-1} e D_{k-1} sono effettivamente le matrici "L" e "D" ottenute nel passo $(k-1)$-esimo.
In particolare vale

$$D_k^{-1} = \left(\begin{array}{ccc|c} & D_{k-1}^{-1} & & 0 \\ \hline 0 & & & d_k^{-1} \end{array}\right), \qquad L_k^{-1} = \left(\begin{array}{ccc|c} & L_{k-1}^{-1} & & 0 \\ \hline * & & & 1 \end{array}\right)$$

Possiamo quindi scrivere, per y_k, la seguente relazione con y_{k-1}:

$$
\begin{aligned}
y_k &= D_k^{-1} L_k^{-1} e_1 \|b\|_2 = \\
&= \left(\begin{array}{c|c} D_{k-1}^{-1} & 0 \\ \hline 0 & d_k^{-1} \end{array}\right) \left(\begin{array}{c|c} L_{k-1}^{-1} & 0 \\ \hline * & 1 \end{array}\right) e_1 \|b\|_2 = \\
&= \left(\begin{array}{c|c} D_{k-1}^{-1} L_{k-1}^{-1} & 0 \\ \hline d_k^{-1} * & d_k^{-1} \end{array}\right) \left(\begin{array}{c} \hat{e}_1 \\ \hline 0 \end{array}\right) \|b\|_2 = \\
&= \left(\begin{array}{c} D_{k-1}^{-1} L_{k-1}^{-1} \hat{e}_1 \|b\|_2 \\ \hline * \end{array}\right) =: \\
&=: \left(\begin{array}{c} y_{k-1} \\ \hline \eta_k \end{array}\right)
\end{aligned}
$$

Per quanto riguarda $\tilde{P}_k$ rispetto a $\tilde{P}_{k-1}$, invece, vale:

$$
\begin{aligned}
\tilde{P}_k &= Q_k L_k^{-T} = \\
&= \left(\begin{array}{c|c} Q_{k-1} & q_k \end{array}\right) \left(\begin{array}{c|c} L_{k-1}^T & * \\ \hline 0 & 1 \end{array}\right) = \\
&= \left(\begin{array}{c|c} Q_{k-1} L_{k-1}^T & \tilde{p}_k \end{array}\right) = \\
&= \left(\begin{array}{c|c} \tilde{P}_{k-1} & \tilde{p}_k \end{array}\right)
\end{aligned}
$$

Note queste equazioni di ricorrenza per y_k e per $\tilde{P}_k$, possiamo scrivere per x_k la seguente relazione

$$
\begin{aligned}
x_k &= \tilde{P}_k y_k = \\
&= \left(\begin{array}{c|c} \tilde{P}_{k-1} & \tilde{p}_k \end{array}\right) \left(\begin{array}{c} y_{k-1} \\ \hline \eta_k \end{array}\right) = \\
&= \tilde{P}_{k-1} y_{k-1} + \tilde{p}_k \eta_k = \\
&= x_{k-1} + \tilde{p}_k \eta_k
\end{aligned}
$$

Possiamo trovare una relazione ricorsiva anche per $\tilde{p}_k$. Infatti, riscrivendo $\tilde{P}_k = Q_k L_k^{-T}$ come $Q_k = \tilde{P}_k L_k^T$ e leggendo questa equazione sulla k-esima colonna, otteniamo

$$\begin{aligned}
q_k &= \tilde{P}_k L_k^T e_k = \\
&= \left(\begin{array}{c|c} \tilde{P}_{k-1} & \tilde{p}_k \end{array} \right) \left(\begin{array}{c|c} L_{k-1} & l_{k-1}\hat{e}_{k-1} \\ \hline 0 & 1 \end{array} \right) e_k = \\
&= l_{k-1}\tilde{P}_{k-1}\hat{e}_{k-1} + \tilde{p}_k = \\
&= l_{k-1}\tilde{p}_{k-1} + \tilde{p}_k
\end{aligned}$$

da cui ricaviamo

$$\tilde{p}_k = q_k - l_{k-1}\tilde{p}_{k-1}$$

Per il teorema 18.1 sappiamo che r_{k-1} è parallelo a q_k, che è normalizzato per norma 2. In particolare possiamo riscrivere q_k come $q_k = \frac{1}{\|r_{k-1}\|_2} r_{k-1}$.
Sappiamo che $x_k = x_{k-1} + \eta_k \tilde{p}_k$. Moltiplicando a sinistra questa equazione per A da entrambi i lati e ricordandoci che $r_k = b - Ax_k$, otteniamo

$$r_k = r_{k-1} - \eta_k A\tilde{p}_k$$

Ponendo $p_{k-1} := \|r_{k-1}\|_2 \tilde{p}_k$ e $\alpha_k := \frac{\eta_k}{\|r_{k-1}\|_2}$, possiamo riscrivere $x_k = x_{k-1} + \eta_k\tilde{p}_k$ come

$$x_k = x_{k-1} + \alpha_k p_k$$

Per quanto riguarda p_{k-1}, vale

$$\begin{aligned}
p_{k-1} &= \|r_{k-1}\|_2 \tilde{p}_k = \\
&= \|r_{k-1}\|_2 (q_k - l_{k-1}\tilde{p}_{k-1}) = \\
&= \|r_{k-1}\|_2 \left(\frac{1}{\|r_{k-1}\|_2} r_{k-1} - l_{k-1}\frac{1}{\|r_{k-2}\|_2} p_{k-2}\right) = \\
&= r_{k-1} - l_{k-1}\frac{\|r_{k-1}\|_2}{\|r_{k-2}\|_2} p_{k-2} =: \\
&=: r_{k-1} + \beta_{k-1} p_{k-2}
\end{aligned}$$

dove definiamo $\beta_k := -l_k \frac{\|r_k\|_2}{\|r_{k-1}\|_2}$

Ora che abbiamo definito x_k, p_k, α_k e β_k, verifichiamo che corrispondono agli stessi che avremmo ottenuto con il metodo del gradiente coniugato.
Abbiamo appena dimostrato che

$$\begin{aligned}
x_k &= x_{k-1} + \alpha_k p_k \\
p_k &= r_k + \beta_k p_{k-1}
\end{aligned}$$

Ad essere più precisi, più che averlo appena dimostrato abbiamo scelto α_k e β_k in modo da avere almeno queste due relazioni ricorsive gratuitamente. Verifichiamo ora che α_k e β_k stessi rispettano le loro relazioni ricorsive. Cominciamo con α_k. Sappiamo che $r_k = r_{k-1} - \eta_k A\tilde{p}_k$. Moltiplicando a sinistra questa equazione per r_{k-1}^T da entrambi i lati, otteniamo

$$\begin{aligned} r_{k-1}^T r_k =& \|r_{k-1}\|_2^2 - \eta_k r_{k-1}^T A\tilde{p}_k = \\ =& \|r_{k-1}\|_2^2 - \frac{\eta_k}{\|r_{k-1}\|_2} r_{k-1}^T A p_{k-1} \end{aligned}$$

e poiché $r_{k-1}^T r_k = 0$ (perché r_{k-1}, r_ksono paralleli rispettivamente a q_k e q_{k+1} che sono tra loro ortogonali) otteniamo

$$\alpha_{k-1} = \frac{\eta_k}{\|r_{k-1}\|_2} = \frac{\|r_{k-1}\|_2^2}{r_{k-1}^T A p_{k-1}}$$

che è esattamente come avevamo definito α_{k-1} nel metodo del gradiente coniugato.
Con un po' di conti, non svolti a lezione e lasciati per esercizio, si verifica che anche la definizione di β_k appena data corrisponde alla definizione di β_k del metodo del gradiente coniugato.
Quindi, i due metodi sono equivalenti.

18.3 Sui metodi di Krylov

La scelta di x_k individua diversi metodi nella famiglia dei metodi di Krylov. Ne elenchiamo ora alcuni:

- Scegliendo x_k che minimizzi $\|r_k\|_2$ otteniamo il metodo MINRES. Esiste anche una versione più generale, chiamata GMRES, applicabile nel caso di A simmetrica ma non necessariamente definita positiva.
- Scegliendo x_k in modo da prendere r_k ortogonale allo spazio $\mathcal{K}_k(A, r_0)$, si genera il metodo di Arnoldi, indicato anche con FOM.
- Scegliendo x_k che minimizzi $\|r_k\|_{A^{-1}}$, come abbiamo appena mostrato, si ottiene il metodo del gradiente coniugato

19 Appendici

19.1 02/10/2020 - $\det(I + wu^T) = 1 + u^T w$

Dimostrazione. Dimostriamolo per induzione su n, dove n è la taglia delle matrici considerate. Il passo base per $n = 1$ è vero in modo banale. Supponiamo che sia vero per $n-1$ e dimostriamolo per n.
Sia $w = (w_1, \dots, w_n)^T$, $u = (u_1, \dots, u_n)^T$ e definiamo i vettori $\tilde{w} = (w_2, \dots, w_n)^T$ e $\tilde{u} = (u_2, \dots, u_n)^T$. Vale

$$\det(I_n + wu^T) = \det \left(\begin{array}{c|c} (1 + w_1 u_1) & w_1 \tilde{u}^T \\ \hline u_1 \tilde{w} & (I_{n-1} + \tilde{w}\tilde{u}^T) \end{array} \right) =$$

$$\begin{aligned} &= (1 + w_1 u_1) \det(I_{n-1} + \tilde{w}\tilde{u}^T) - w_1 \tilde{u}^T u_1 \tilde{w} = \\ &= (1 + w_1 u_1)(1 + \tilde{u}^T \tilde{w}) - w_1 u_1 \tilde{u}^T \tilde{w} = \\ &= 1 + w_1 u_1 + \tilde{u}^T \tilde{w} + w_1 u_1 \tilde{u}^T \tilde{w} - w_1 u_1 \tilde{u}^T \tilde{w} = \\ &= 1 + w_1 u_1 + \tilde{u}^T \tilde{w} = \\ &= 1 + u^T w \end{aligned}$$

□

19.2 07/10/2020 - Esistenza di $\{\check{S}_k\}$

Il seguente fatto viene riportato con una traccia per la dimostrazione copiata dal libro Metodi Numerici per l'Algebra Lineare([26]) ed è riportata qua solo per completezza degli appunti, per non dover studiare da 7 fonti diverse.

Teorema 19.1. *Sia $\{A_k\}$ una successione di matrici tali che*

$$\lim_{k \to +\infty} A_k = I$$

e sia $A_k = Q_k R_k$ una fattorizzazione QR della matrice A_k.
Allora esistono matrici S_k diagonali e unitarie (cioè matrici di fase) tali che

$$\lim_{k \to +\infty} Q_k S_k = \lim_{k \to +\infty} S_k Q_k = I, \quad \lim_{k \to +\infty} S_k^H R_k = \lim_{k \to +\infty} R_k S_k^H = I$$

Dimostrazione (Traccia). vale

$$\lim_{k \to +\infty} (Q_k R_k - I) = 0$$

e quindi, poiché gli elementi di Q_k hanno modulo limitato, si ha

$$\lim_{k \to +\infty} (R_k - Q_k^H) = \lim_{k \to +\infty} Q_k^H (Q_k R_k - I) = 0$$

[26] `http://people.cs.dm.unipi.it/bini/Metodi_Numerici_per_l'Algebra_Lineare.pdf` pagine 412-413, esercizio 6.30

Poiché R_k è triangolare superiore, segue che

$$\lim_{k\to+\infty} \bar{q}_{ji}^{(k)} = 0, \quad \text{per } i > j$$

Da cui, poiché Q_k è unitaria, si ottiene

$$\lim_{k\to+\infty} \bar{q}_{ji}^{(k)} = 0, \quad \text{per } i < j$$

Quindi $q_{ii}^{(k)} \neq 0$ da un certo indice k in poi, e posto

$$S_k = \begin{pmatrix} \theta_1^{(k)} & & & \\ & \theta_2^{(k)} & & \\ & & \ddots & \\ & & & \theta_2^{(k)} \end{pmatrix}, \quad \theta_i^{(k)} := \frac{\bar{q}_{ii}^{(k)}}{\left|q_{ii}^{(k)}\right|}$$

si dimostri che

$$\lim_{k\to+\infty} S_k Q_k^H = \lim_{k\to+\infty} Q_k^H S_k = I$$

□

19.3 23/10/2020 - R_1 è diagonale

Proposizione 19.1. *Sia R_1 triangolare superiore tale che gli elementi diagonali sono non negativi e ordinati in modo non crescente, di cui almeno uno non nullo. Se $R_1^H R_1$ è diagonale, allora anche R_1 è diagonale.*

Osservazione. La richiesta di avere almeno un elemento diagonale non nullo non è restrittiva poiché, applicando la tecnica del massimo pivot totale, l'unico caso in cui R_1 abbia almeno gli elementi diagonali tutti nulli è il caso in cui la matrice A di partenza è la matrice nulla. Possiamo supporre che A sia diversa dalla matrice nulla poiché se lo fosse ci sarebbe ben poco da analizzare.

Dimostrazione. Si procede per induzione su n taglia della matrice. Il caso $n = 1$ è banale, poiché tutte le matrici di taglia 1 sono diagonali.
Supponiamo che la tesi sia vera per $n-1$ e dimostriamola per n.
Scomponiamo R_1 nel seguente modo

$$R_1 = \left(\begin{array}{c|c} T_1 & v \\ \hline & r \end{array}\right)$$

Considero allora il prodotto $R_1^H R_1 = \Lambda$ diagonale. Vale

$$R_1^H R_1 = \left(\begin{array}{c|c} T_1^H & \\ \hline v^H & r^H \end{array}\right) \left(\begin{array}{c|c} T_1 & v \\ \hline & r \end{array}\right) = \left(\begin{array}{c|c} T_1^H T_1 & T_1^H v \\ \hline v^H T_1 & v^H v + r^H r \end{array}\right) = \Lambda$$

Allora anche $T_1^H T_1$ è diagonale e, per ipotesi induttiva, T_1 deve essere diagonale. Dall'ultima uguaglianza ricaviamo anche che $T_1^H v = 0$. Poiché T_1 è diagonale, questa cosa può accadere se e solo se almeno uno dei due è nullo. Poiché per ipotesi almeno uno degli elementi diagonali di R_1 è non nullo e poiché sono tutti ordinati in ordine non crescente, sicuramente almeno il primo elemento diagonale è non nullo, quindi T_1 non è la matrice nulla. Ne segue che $v = 0$.
Per come abbiamo scritto R_1, poiché $v = 0$ e poiché T_1 è diagonale per ipotesi induttiva, anche R_1 è diagonale. □

19.4 11/11/2020 - Il punto stazionario di $\Phi(x)$ è un minimo globale

Proposizione 19.2. *Sia $A \in \mathbb{R}^{n\times n}$ definita positiva e $b \in \mathbb{R}^n$. Sia $x^* = A^{-1}b$.*

$$\Phi(x) = \frac{1}{2}x^T Ax - b^T x$$

Allora x^* non solo è un punto stazionario di Φ (già dimostrato nella lezione 11/11/2020) ma è un punto di minimo globale.

Dimostrazione. Mostriamo che per ogni $x \neq x^*, \Phi(x^*) < \Phi(x)$.
Sia $x \neq x^*$. Indichiamo con $h := x - x^* \neq 0$ la differenza tra x e x^*. Possiamo riscrivere $x = x^* + h$.
Vale

$$\begin{aligned}
\Phi(x) &= \frac{1}{2}x^T Ax - b^T x = \\
&= \frac{1}{2}(x^* + h)^T A(x^* + h) - b^T(x^* + h) = \\
&= \frac{1}{2}(x^*)^T Ax^* + \frac{1}{2}h^T Ax^* + \frac{1}{2}(x^*)^T Ah + \frac{1}{2}h^T Ah - b^T x^* - b^T h = \\
&= \Phi(x^*) + \frac{1}{2}h^T Ax^* + \frac{1}{2}(x^*)^T Ah + \frac{1}{2}h^T Ah - b^T h = \\
&= \Phi(x^*) + \frac{1}{2}(x^*)^T Ah + \frac{1}{2}(x^*)^T Ah + \frac{1}{2}h^T Ah - b^T h = \\
&= \Phi(x^*) + (x^*)^T Ah + \frac{1}{2}h^T Ah - b^T h
\end{aligned}$$

Poiché $Ax^* = b$ possiamo riscrivere $b^T = (x^*)^T A^T = (x^*)^T A$ utilizzando la simmetria di A. Sostituendo otteniamo

$$\begin{aligned}
\Phi(x) &= \Phi(x^*) + (x^*)^T Ah + \frac{1}{2}h^T Ah - b^T h = \\
&= \Phi(x^*) + (x^*)^T Ah + \frac{1}{2}h^T Ah - (x^*)^T Ah = \\
&= \Phi(x^*) + \frac{1}{2}h^T Ah
\end{aligned}$$

Inoltre, poiché A è definita positiva, $\frac{1}{2}h^T Ah > 0$, quindi

$$\Phi(x) = \Phi(x^*) + \frac{1}{2}h^T Ah > \Phi(x^*)$$

□