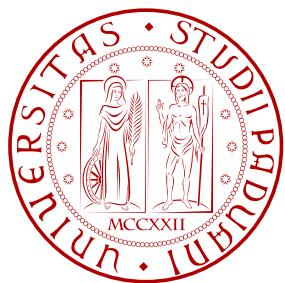


UNIVERSITÀ DEGLI STUDI DI PADOVA

Corso di Laurea Triennale in Matematica



Punti di Lebesgue sul disco

Andrea Pinto

Relatori:

- Prof. Francesco Rinaldi
- Prof. Alvise Sommariva

Anno Accademico 2011/12

Indice

| | |
|--|-----------|
| Introduzione | 2 |
| 1 Interpolazione polinomiale e costante di Lebesgue | 2 |
| 1.0.1 Punti di Lebesgue | 4 |
| 1.1 Richiami su alcuni nodi di interpolazione classici nell'intervallo | 5 |
| 2 Algoritmi | 7 |
| 2.1 Algoritmo Genetico | 7 |
| 2.1.1 Selezione | 7 |
| 2.1.2 Mutazione | 8 |
| 2.1.3 Crossover | 8 |
| 2.1.4 Altri parametri | 8 |
| 2.1.5 Alcune considerazioni | 9 |
| 2.2 Direct search (Pattern Search) | 9 |
| 2.2.1 Parametri | 10 |
| 2.3 Combinazione di GA con Patternsearch | 10 |
| 3 Funzione obiettivo | 11 |
| 4 Un primo approccio | 11 |
| 5 Secondo approccio: punti vincolati su cerchi concentrici | 13 |
| 5.1 La suddivisione in k gruppi | 13 |
| 5.2 La scelta dell'angolo | 16 |
| 5.3 Confronto tra alcune configurazioni di raggi | 17 |
| 5.4 Ottimizzando sia su raggi sia su angoli | 18 |
| 5.4.1 Confronto con la configurazione di raggi particolari . . | 20 |
| 5.4.2 Confronto con altri autori | 20 |
| 5.4.3 Confronto con la teoria | 23 |
| 6 Possibili sviluppi | 26 |
| A Alcuni dati | 28 |
| B Codici | 33 |
| Bibliografia | 45 |

Introduzione

La ricerca di punti adeguati per l'interpolazione polinomiale è uno dei grandi temi della teoria dell'approssimazione: il problema consiste nel ricercare le disposizioni dei punti che permettano l'interpolazione polinomiale col minor errore possibile. Nel caso unidimensionale i risultati a livello sia puramente teorico che numerico ci forniscono delle configurazioni decisamente buone. Nel caso multidimensionale il problema è più complesso e quindi numerosi autori continuano a interessarsene: esistono delle buone configurazioni per domini molto semplici quali cerchi, quadrati e triangoli. La bontà delle configurazioni trovate viene stimata attraverso la costante di Lebesgue. In tal senso, in questo lavoro ci siamo concentrati al caso del disco bidimensionale cercando configurazioni che minimizzassero tale costante: è stato possibile riuscire a trovare dei lievi miglioramenti ai già ottimi risultati presenti in letteratura a riguardo. Nelle prossime pagine, dopo aver fornito dei veloci richiami sull'interpolazione polinomiale e dopo aver definito in maniera rigorosa la costante di Lebesgue, illustreremo come sia stato possibile ottenere delle configurazioni di punti con una discreta costante di Lebesgue. Inoltre proporremo diversi confronti tra i dati che abbiamo raccolto nelle sperimentazioni numeriche sia con quelli di altri autori, sia con stime puramente teoriche.

1 Interpolazione polinomiale e costante di Lebesgue

Il nostro obiettivo è di fornire, per un certo valore n fissato, un "buon" insieme di punti $\{z_n\} = \{(x_i, y_i)\}_{i=1..N}$ nel cerchio unitario di \mathbb{R}^2 . La bontà di questi punti è espressa con una bassa costante di Lebesgue, che come vedremo sarà un buon indice di stabilità e della qualità dell'errore rispetto a quello della miglior approssimazione.

Indichiamo con $\mathbb{P}_n^d = \langle x_1^{n_1} x_2^{n_2} \dots x_d^{n_d}, 0 \leq \sum_{i=1}^d n_i \leq n \rangle$ lo spazio dei polinomi di grado totale n . Si può vedere che $N = \dim(\mathbb{P}_n^d) = \binom{n+d}{n}$. Sia Ω un dominio compatto di \mathbb{R}^d . Un insieme di punti $\{z_n\}$ si dice unisolvente su Ω per l'interpolazione polinomiale di grado n , se esiste, per ogni vettore $\gamma = \{\gamma_k\}_{k=1..N}$, un unico polinomio p_n in \mathbb{P}_n^d tale che

$$p_n(z_k) = \gamma_k \quad k = 1, \dots, N$$

Segue direttamente da questa definizione una condizione di necessità per l'unisolenza: la cardinalità N di tale insieme $\{z_n\}$ deve essere uguale alla

dimensione di \mathbb{P}_n .

Fissiamo nel dominio Ω una base $\langle p_i, i = 1, \dots, N \rangle$ di \mathbb{P}_n e denotiamo con $V = V(z_1, z_2, \dots, z_N)$ la matrice quadrata di Vandermonde le cui componenti sono

$$V_{ij} = \{p_j(x_i) : i, j = 1, \dots, N\}$$

Definiamo ora i polinomi di Lagrange come

$$L_k(z) = \frac{\det(V(z_1, \dots, z_{k-1}, z, z_{k+1}, \dots, z_N))}{\det(V(z_1, \dots, z_N))} \quad k = 1, \dots, N \quad (1)$$

è l'operatore di interpolazione su un insieme di punti \mathcal{P} come il funzionale $\mathcal{L}_n : (C(\Omega), \|\cdot\|_\infty) \rightarrow \mathbb{P}_n^d$

$$\mathcal{L}_n f(.) = \sum_{i=1}^N f(z_i) L_i(.)$$

Nel caso unidimensionale si può facilmente mostrare che

$$L_k(z) = \prod_{i=0, i \neq k}^N \frac{z - z_i}{z_k - z_i} = \frac{z - z_0}{z_k - z_0} \dots \frac{z - z_{k-1}}{z_k - z_{k-1}} \frac{z - z_{k+1}}{z_k - z_{k+1}} \dots \frac{z - z_N}{z_k - z_N}$$

Se $d = 1$, e se i punti su cui si interpola sono tutti distinti esiste ed è unico il polinomio p_n che interpola le coppie (z_k, γ_k) .

Più in generale, per d generico, da (1), si vede che se l'insieme di punti è unisolvente, si ha $L_k(x_s) = \delta_{k,s}$ per ogni k e per ogni s .

La costante di Lebesgue $\Lambda_n(\mathcal{P})$ di un insieme di punti \mathcal{P} è la norma dell'operatore \mathcal{L}_n :

$$\Lambda_n(\mathcal{P}) = \|\mathcal{L}_n\| = \frac{\|\mathcal{L}_n f\|_\infty}{\|f\|_\infty} = \max_{x \in \Omega} \sum_{i=1}^N |L_{z_i}| \quad (2)$$

La grandezza di $\Lambda_n(\mathcal{P})$ è in funzione dell'insieme dei punti di interpolazione \mathcal{P} , del dominio Ω e di \mathbb{P}_n^d ma non della base di quest'ultimo insieme poiché il determinante di una generica matrice non dipende dalla base scelta. Vediamo ora delle stime che ci fanno intuire l'importanza di una buona costante di Lebesgue.

Teorema.

Sia \mathbb{P}_n^d lo spazio dei polinomi di grado totale n su \mathbb{R}^d , sia Ω un insieme compatto di \mathbb{R}^d e sia $\mathcal{P} = \{\mathbf{x}_i\} \subset \Omega$ un insieme di punti unisolvente per l'interpolazione polinomiale di grado n . Sia inoltre $\mathcal{L}_n : (C(\Omega), \|\cdot\|_\infty) \rightarrow \mathbb{P}_n^d$ l'operatore di interpolazione polinomiale sui punti $\{\mathbf{x}_i\}$ e sia $\Lambda_n(\mathcal{P})$ la corrispondente costante di Lebesgue, allora

1. $\|\mathcal{L}_n f - f\|_\infty \leq (1 + \Lambda_n(\mathcal{P})) \|f - \phi_n^*\|_\infty$ ove $\phi_n^* \in \mathbb{P}_n^d$ è il polinomio di grado n di miglior approssimazione per f .
2. $\left\| \mathcal{L}_n f - \mathcal{L}_n \tilde{f} \right\|_\infty \leq \Lambda_n(\mathcal{P}) \max_k |f(x_k) - \tilde{f}(x_k)|$ ove \tilde{f} è la funzione perturbata di f .

Dimostrazione.

1. Notiamo che $\phi_n^* - \mathcal{L}_n \phi_n^* = 0$ poiché $\phi_n^* \in \mathbb{P}_n^d$. Dunque

$$f - \mathcal{L}_n f = f - \phi_n^* + \phi_n^* - \mathcal{L}_n f = f - \phi_n^* + \phi_n^* - \mathcal{L}_n \phi_n^* + \mathcal{L}_n \phi_n^* - \mathcal{L}_n f = f - \phi_n^* + \mathcal{L}_n(\phi_n^* - f)$$

Ma allora

$$\begin{aligned} \|f - \mathcal{L}_n f\|_\infty &\leq \|f - \phi_n^*\|_\infty + \|\mathcal{L}_n \phi_n^* - \mathcal{L}_n f\|_\infty \leq \\ &\leq \|f - \phi_n^*\|_\infty + \|\mathcal{L}_n\|_\infty \|f - \phi_n^*\|_\infty = (1 + \Lambda_n(\mathcal{P})) \|f - \phi_n^*\|_\infty. \end{aligned}$$

2. $\left| \mathcal{L}_n f - \mathcal{L}_n \tilde{f} \right| = \left| \sum_i^N f(x_i) L_i(x) - \sum_i^N \tilde{f}(x_i) L_i(x) \right| \leq \left| \sum_i^N ((f(x_i) - \tilde{f}(x_i)) L_i(x)) \right| \leq$

$$\leq \sum_i^N |f(x_i) - \tilde{f}(x_i)| |L_i(x)| \leq \max_i |f(x_i) - \tilde{f}(x_i)| \sum_i^N |L_i(x)|$$

Ma allora

$$\left\| \mathcal{L}_n f - \mathcal{L}_n \tilde{f} \right\|_\infty \leq \|\mathcal{L}_n\|_\infty \max_i |f(x_i) - \tilde{f}(x_i)|$$

□

1.0.1 Punti di Lebesgue

Fissati Ω e \mathbb{P}_n^d i punti che minimizzano la costante di Lebesgue sono detti *Punti di Lebesgue*. Tali punti non sono noti nemmeno nel caso univariato: in letteratura si ritiene che nell'intervallo limitato $[a,b]$ gli "expanded-Chebyshev"

$$x_j = \frac{a+b}{2} - \frac{(a-b)}{2} \frac{\cos((2j-1)\pi/(2n))}{\cos(\pi/(2n))} \quad j = 1, \dots, n$$

sono quasi-ottimi. Miglioramenti a questi punti sono stati trovati da [2]. Un'altra famiglia di punti molto importante è quella dei punti di Fekete che sono definiti come i punti che massimizzano il valore assoluto del determinante della matrice di Vandermonde V . È facile vedere che per tali punti vale

$$\Lambda_n = \max_{z \in \Omega} \sum_{k=1}^N |L_k(z)| \leq N = \binom{n+d}{n}.$$

Infatti applicando la Definizione 1 si vede che, per ogni k , $|L_k| \leq 1$ poiché i punti di Fekete massimizzano il determinante: quindi il denominatore di L_K è sempre maggiore del rispettivo numeratore. Da (2) deriva tale stima.

Questa relazione sovrastima di $\Lambda_n(\mathcal{P})$.

La costante di Lebesgue Λ_n è invariante per trasformazioni affini. Questo significa che conoscendo dei punti di interpolazione per l'intervallo $[-1,1]$, con una bassa costante di Lebesgue, li si conosce anche, per qualsiasi intervallo $[a,b]$, o nel caso bidimensionale noti i punti di Lebesgue di un dominio di riferimento Ω^* , si conoscono pure i punti di Lebesgue di domini Ω che sono trasformazioni affini di Ω^* .

1.1 Richiami su alcuni nodi di interpolazione classici nell'intervallo

Analizzeremo adesso alcune famiglie di punti classiche per l'interpolazione. Tali considerazioni verranno utilizzate nella sezione 5.3.

- **Chebyshev.** Questi punti nell'intervallo $[1-1]$ sono gli zeri dei polinomi di Chebyshev $T_n(x) = \cos(n \arccos(x))$. Esiste una formulazione ricorsiva di questi polinomi

$$\begin{cases} T_0(x) = 1 \\ T_1(x) = x \\ T_{n+1}(x) = 2xT_n(x) - T_{n-1} \end{cases}$$

Per il grado n , tali punti, in un intervallo $[a,b]$, risultano essere

$$x_i = \frac{(a+b)}{2} + \frac{(b-a)}{2} \cos\left(\frac{2i-1}{2n}\pi\right)$$

con $i=1\dots n$

- **Expanded Chebyshev** Come descritto in [3] sono una dilatazione dei punti di Chebyshev così che il primo e l'ultimo nodo siano estremi dell'intervallo. Sono dati dalla seguente formulazione:

$$x_i = \frac{a+b}{2} + \frac{b-a}{2} \frac{\cos(\frac{(2i-1)\pi}{2n})}{\cos(\frac{\pi}{2n})} \quad i = 1..n$$

Gli Expanded Chebyshev risultano avere una costante di Lebesgue leggermente più piccola rispetto a quella dei punti di Chebyshev, e risultano avere una costante di Lebesgue quasi ottimale per l'intervallo.

- **Chebyshev-Lobatto** Questi punti nell'intervallo [-1,1] sono gli zeri dei polinomi di Chebyshev di seconda specie, ovvero gli zeri di \mathcal{U}_n ove

$$\begin{cases} \mathcal{U}_0(x) = 1 \\ \mathcal{U}_1(x) = 2x \\ \mathcal{U}_{n+1}(x) = 2x\mathcal{U}_n(x) - \mathcal{U}_{n-1}(x) \end{cases}$$

Tali polinomi sono ortogonali con la funzione peso $\sqrt{1-x}$ nell'intervallo [-1,1]. Per il grado n , tali punti, in un intervallo [a,b], risultano essere:

$$x_i = \frac{a+b}{2} + \frac{b-a}{2} \cos\left(\frac{k-1}{n}\pi\right)$$

con $i=1..n$

- **Legendre-Lobatto** Questi punti nell'intervallo [-1,1] sono gli zeri dei polinomi di Legendre, ovvero gli zeri di \mathcal{P}_n con \mathcal{P}_n data dalla formulazione ricorsiva

$$\begin{cases} \mathcal{P}_0 = 1 \\ \mathcal{P}_1 = x \\ \mathcal{P}_n = \frac{2n-1}{n}x\mathcal{P}_{n-1} - \frac{n-1}{n}\mathcal{P}_{n-2} \end{cases}$$

con $i=0..n$. Tali punti sono molto utilizzati per le formule di quadratura utilizzando come pesi

$$A_i = \frac{2}{n(n+1)} \frac{1}{(\mathcal{P}_n(x_i))^2}$$

con $i = 0, \dots, n$

2 Algoritmi

Precedentemente in [1], per il procedimento di minimizzazione, sono stati usati algoritmi active-set o BGFS. A causa del determinante di Vandermonde che è nullo sulle configurazioni non unisolventi, la nostra funzione obiettivo non risulta essere particolarmente regolare ma con frequenti oscillazioni, estremanti locali, punti di discontinuità e singolarità. L'utilizzo di metodi gradiente quindi potrebbe non agevolare la ricerca di minimi. Inoltre con questi metodi è più probabile convergere ad un estremo locale, invece di minimizzare globalmente la costante di Lebesgue. In questo lavoro, invece, abbiamo seguito un approccio diverso: come già in parte [2], abbiamo usato algoritmi genetici di natura principalmente aleatoria o algoritmi di ricerca diretta.

2.1 Algoritmo Genetico

Il primo algoritmo è Genetic Algorithm (GA) presente nel pacchetto di Matlab "Global Optimization Toolbox". La diversità di GA è che invece di eseguire una ricerca diretta di configurazioni ottimali, ad esempio lungo la direzione gradiente, svolge una ricerca parallela, con un procedimento che ricorda l'evoluzione darwiniana. Vediamo GA in dettaglio. Fissato un certo intero NP, GA genera una popolazione in cui ogni individuo $x^{(i)}$ con $i=1,\dots,NP$, è rappresentato da un vettore (nel nostro caso costituito dalla posizione di nodi). La popolazione iniziale può essere generata in maniera casuale da GA oppure messa in input. Come nell'evoluzione naturale, la popolazione per passare alla generazione successiva, attraversa le fasi di selezione, mutazione e crossover.

2.1.1 Selezione

La selezione è la scelta dei genitori per la prossima generazione. Per prima cosa viene fatta una classifica delle persone, in base a quanto ottimizzano la funzione obiettivo. Da questa lista si scelgono poi le persone da usare in base a certi criteri. Noi ne abbiamo usati principalmente due: roulette e tournament. Roulette dà una percentuale a ogni persona: bassa se è in cima alla classifica, alta se in fondo. Poi in maniera casuale si sceglie chi selezionare. Tournament invece crea un torneo in cui partecipa tutta la popolazione: ci sono sfide tra un numero prefissato di persone e passa al turno successivo del torneo solo la migliore di ogni gruppo. Questo criterio permette con sicurezza che la persona migliore sia selezionata mentre quello Roulette permette maggiore aleatorietà anche a costo di perdere un ipotetico genitore migliore.

2.1.2 Mutazione

Alcune nuove persone subiscono una mutazione. Ci sono diversi modi per gestire questa mutazione. Noi abbiamo usato una distribuzione Gaussiana centrata nello zero. Uno dei vantaggi di questa impostazione è stata la possibilità di scegliere separatamente la deviazione standard della prima e delle altre generazioni.

2.1.3 Crossover

In questa fase viene gestita la creazione dei figli: avviene una combinazione tra i due genitori per dar vita ad una nuova persona. Ci sono diversi modi per gestire questa combinazione. Noi abbiamo usato principalmente Scattered, TwoPoint e Intermediate. Scattered per prima cosa crea un vettore binario casuale. Il figlio avrà la i -esima componente (o gene) uguale a quella del primo genitore, se il vettore casuale ha la i -esima componente uguale a 1, altrimenti avrà tale componente uguale a quella del secondo genitore. TwoPoint per prima cosa sceglie in maniera random due interi m e n tra 1 e il numero di variabili. L'algoritmo poi prende le componenti (o geni) con indice minore o uguale ad m e quelle con indice maggiore di n dal primo genitore mentre tutte le altre dal secondo. Intermediate crea invece i figli con una media pesata dei genitori. È controllata unicamente dal singolo parametro Ratio. Utilizzando una forma simbolica: $\text{figlio} = \text{genitore1} + \text{rand} * \text{Raz} * (\text{genitore2} - \text{genitore1})$ ove rand è un numero casuale tra 0 e 1. I primi due si sono rivelati forse più efficaci perché vanno a modificare la posizione dei punti in maniera meno drastica, agendo quasi componente per componente, mentre il terzo metodo crea una combinazione convessa dei genitori.

2.1.4 Altri parametri

Ora che abbiamo introdotto questi tre processi fondamentali illustreremo come viene creata la generazione successiva. Un numero prefissato di persone rimane uguale al genitore. Questo numero abbiamo preferito tenerlo basso (da 1 a 4) per rendere più aleatorio l'algoritmo. Delle restanti persone una parte viene gestita dal crossover, un'altra invece dalla mutazione. La percentuale di partizione tra queste due viene introdotta in input con un numero da 0 a 1. Essendo questo parametro particolarmente importante abbiamo provato molti numeri in tutto l'intervallo, concentrandoci maggiormente intorno allo 0.8 poiché ritenuata la configurazione migliore.

Il nostro problema di ottimizzazione è vincolato, e nulla impedisce al crossover o alla mutazione di creare persone fuori dai nuovi vincoli: per questo abbiamo dovuto inserire un numero elevato come penalità. Se un punto esce

dal disco automaticamente gli viene associato un alto valore in funzione obiettivo e quindi nella prossima generazione verrà scartato.

Un'impostazione, che abbiamo usato per aumentare l'aleatorietà, è di suddividere la popolazione in città e far agire l'algoritmo separatamente per ognuna di queste. Inoltre ogni 20 o 30 generazioni le varie città si scambiano una certa percentuale di persone.

Altra scelta che abbiamo fatto è stato legare GA con un altro algoritmo di ricerca: il Patternsearch. Una volta che GA termina i suoi processi pattern-search inizia. Ma vedremo nel dettaglio, nella prossima sezione, come agisce questo algoritmo.

Un aspetto da considerare è la dimensione della popolazione: in teoria anche questo parametro può influenzare la convergenza e la sua velocità, poiché una popolazione più grande fornisce una varietà nell'evoluzione maggiore e maggiori capacità di esplorazione in varie regioni. Per questo motivo in [2] si osserva che una popolazione di $10N$, ove N è il numero di parametri da parametrizzare, è sicuramente buono. Tuttavia con le sperimentazioni fatte sempre in [2] si osserva che già con 100 individui si hanno ottimi risultati con un costo computazionale molto più basso. Per questi motivi abbiamo deciso di utilizzare anche noi 100 in questo parametro. Altri parametri importanti sono il numero di generazioni totali e di stallo. Facile capire che questi parametri ci indicano la durata del nostro processo: un numero molto alto consentirà la convergenza. Abbiamo posto 500 per il primo e 100 per il secondo. A livello sperimentale abbiamo verificato la bontà di tali scelte.

2.1.5 Alcune considerazioni

Come abbiamo illustrato nei paragrafi precedenti, ad ogni iterazione (o generazione), viene selezionata una popolazione più adatta e l'algoritmo dovrebbe convergere al minimo globale. Sfortunatamente non esiste alcuna garanzia teorica di convergenza locale: il metodo è soggetto alla possibilità di stagnazione in un minimo locale. Analogamente, la velocità di convergenza non è stimabile a causa della natura aleatoria. Ricordiamo comunque con algoritmo simile è stato usato con successo già in [2] in un problema analogo di minimizzare la costante di Lebesgue.

2.2 Direct search (Pattern Search)

Direct search è un metodo per risolvere problemi di ottimizzazione che non richiede alcuna informazione circa il gradiente della funzione obiettivo. A differenza dei metodi di ottimizzazione tradizionali che utilizzano informazioni sul gradiente o più derivate per cercare un punto ottimale, un algoritmo

di ricerca diretta cerca in un insieme di punti intorno al punto attuale, cercandone uno cui il valore della funzione obiettivo è inferiore al valore nel punto corrente. È possibile utilizzare DirectSearch per risolvere i problemi per i quali la funzione obiettivo non è derivabile, o non continua.

In Matlab sono implementati tre algoritmi di DirectSearch: generalized pattern search (GPS) , the generating set search(GSS) , e the mesh adaptive search (MADS). Questi algoritmi sono tutti inglobati nella routine Pattern Search presente nel pacchetto "Global Optimization Toolbox". Ad ogni passo, questa routine cerca un insieme di punti, chiamato mesh, attorno al corrente punto, calcolato al passo precedente o al dato iniziale. La mesh è formata aggiungendo al punto corrente una combinazione lineare di vettori chiamati pattern. Se l'algoritmo trova un punto nella maglia che migliora la funzione obiettivo rispetto al punto corrente, il nuovo punto diventa il punto corrente al passaggio successivo dell'algoritmo. Si crea così una successione di punti che convergerà ad un minimo.

L'algoritmo GPS utilizza vettori direzionali fissi. L'algoritmo GSS è identico all'algoritmo GPS, tranne quando ci sono vincoli lineari, e quando il punto corrente è vicino a un confine vincolo lineare. L'algoritmo MADS utilizza una selezione casuale di vettori per definire la rete.

2.2.1 Parametri

Intuibile che come primo parametro dobbiamo scegliere quale usare tra gli algoritmi di cui sopra abbiamo discusso: noi ci siamo focalizzati sul primo. C'è la possibilità di decidere se verificare tutti i punti della mesh o solo alcuni: noi abbiamo optato per la prima soluzione anche se in questo modo abbiamo un costo computazionale più alto. Altra opzione è quella di far eseguire un altro algoritmo dopo che questo ha concluso: nel caso in cui routine venisse utilizzata non da sola abbiamo deciso di spegnere questa funzione.

2.3 Combinazione di GA con Patternsearch

Esistono risultati teorici che garantiscono che, combinando un algoritmo genetico con uno di Direct Search, la convergenza a un minimo globale è assicurata. Per questo motivo noi abbiamo scelto di combinare Ga con Patternsearch. Sfortunatamente non esistono simili risultati riguardo la velocità di convergenza.

3 Funzione obiettivo

Per quanto riguarda il calcolo della costante di Lebesgue di ogni insieme di punti $\mathcal{P} = \{z_i\}$ unisolvente su Ω , abbiamo ereditato le tecniche presenti in [1]. Data una mesh di riferimento sufficientemente fitta X , si valuta la costante di Lebegue Λ_n su questi punti (invece che su tutto il dominio) e si suppone

$$\Lambda_n(P) = \|\mathcal{L}_n\| = \max_{x \in \Omega} \lambda_n(x; P) \approx \max_{x \in X} \lambda_n(x, P)$$

dove λ_n è la funzione di Lebesgue:

$$\lambda_n(x, P) = \sum_{i=1}^N |L_i(x)|, \quad x \in X$$

Tale procedimento è lecito perché queste sono mesh debolmente ammissibili. In [6] si definisce *Mesh Debolmente Ammissibile* di Ω una successione di sottoinsiemi discreti $\mathcal{A}_n \subset \Omega$ tali che

$$\|p\|_\Omega \leq C(\mathcal{A}_n) \|p\|_{\mathcal{A}_n}, \forall p \in \mathbb{P}_n^d(\Omega)$$

con $\text{card}(\mathcal{A}_n) \geq N$ e $C(\mathcal{A}_n)$ abbia una crescita al massimo in modo polinomiale in n . Se inoltre $C(\mathcal{A}_n)$ è limitata si parla di *Mesh Ammissibile*. Utilizziamo quindi griglie che determinano la norma dei polinomi a meno di costanti. In questo caso, abbiamo usato \mathcal{A}_n aventi al massimo di $250^2 = 62500$ punti, come già in [1, 2].

In questo modo abbiamo che

$$\begin{aligned} |\Lambda_n(\mathcal{P}) - \tilde{\Lambda}_n(\mathcal{P})| &\leq \left| \max_{x \in \Omega} \sum_{i=1}^N |L_i(x)| - \max_{x \in \mathcal{A}_n} \sum_{i=1}^N |L_i(x)| \right| \leq \\ &\leq \left| \sum_{i=1}^N \|L_i\|_\Omega - \|L_i\|_{\mathcal{A}_n} \right| \leq \left| \sum_{i=1}^N (C(\mathcal{A}_n) - 1) \|L_i\|_{\mathcal{A}_n} \right| \leq \\ &\leq N |C(\mathcal{A}_n) - 1| \max_{i=1..N} \|L_i\|_{\mathcal{A}_n} \end{aligned}$$

ove con $\tilde{\Lambda}_n(\mathcal{P})$ abbiamo indicato la costante di Lebesgue ristretta all'insieme \mathcal{A}_n . L'uso di queste mesh, è una garanzia che la discretizzazione del dominio scelta è buona e, valutando la funzione di Lebesgue esclusivamente su questi nodi, si ottiene una buona approssimazione della costante di Lebesgue.

4 Un primo approccio

Di seguito, intendiamo calcolare punti unisolventi per l'interpolazione sul disco unitario

$$\Omega = \{(x, y) : x^2 + y^2 \leq 1\}$$

che abbiano costante di Lebesgue particolarmente piccola. Il proposito è determinare set di punti per cui questo valore sia inferiore a quello ottenuto nei lavori [1,2,7]. Per prima cosa abbiamo modificato i codici di [1] utilizzando prima GA e poi Pattern-Search al posto di fmincon. Dopo una prima fase di taratura dei suddetti algoritmi di ottimizzazione, nonostante la capacità di sorpassare possibili minimi locali ma non globali, abbiamo riottenuto le configurazioni di punti di [1] e quindi non abbiamo migliorato il valore minimo della costante di Lebesgue.

Nel caso di GA abbiamo riscontrato il problema, che impostando una popolazione iniziale casuale la convergenza era troppo lenta, impedendo di raggiungere risultati positivi in tempi ragionevoli. Per gradi bassi come 6 e 7 dopo venti ore di calcolo eravamo ancora ben lontani da una costante di Lebesgue paragonabile con quella trovata da [1] e [2]. Inoltre spesso abbiamo notato la difficoltà da parte dell'algoritmo di uscire da intorni di un minimo locale: di frequente infatti l'algoritmo rimaneva fermo nello stesso intorno anche per diverse ore. Problemi di stagnazione simili si sono presentati, impostando come popolazione iniziale, configurazioni di punti particolarmente "buone" come i quasi-punti di Lebesgue e di Fekete trovati da [1]. Pur variando molto i parametri dell'algoritmo, non siamo riusciti ad evitare questo stagnamento.

Con Pattern-Search, invece, abbiamo cercato, a livello locale, miglioramenti negli intorni dei punti "buoni" trovati da [1]. Anche in questo caso non abbiamo avuto fortuna: nell'intorno della nostra ricerca i punti iniziali risultavano essere quelli con la costante di Lebesgue più bassa.

Non abbiamo avuto migliori risultati combinando insieme GA con Pattern-search: partendendo da un punto iniziale casuale la convergenza si è rivelata troppo lenta, partendo da punti noti è stato impossibile evitare fenomeni di stagnamento.

Abbiamo quindi abbandonato questo approccio spostando la nostra attenzione a tentativi di ottimizzare la costante di Lebesgue per punti con configurazioni particolari.

5 Secondo approccio: punti vincolati su cerchi concentrici

Riprendendo un'idea presente in [7] siamo riusciti a trovare dei miglioramenti rispetto a [1, 2, 7] per la ricerca dei punti di Lebesgue. Nel caso di interpolazione con un polinomio di grado n in \mathbb{R}^2 determineremo insiemi unisolventi composti di $N = \binom{n+2}{n}$ punti che saranno opportunamente disposti in $\lfloor \frac{n}{2} \rfloor + 1$ cerchi. Dividiamo i cerchi in k gruppi:

$$v_1 + \dots + v_k = \lfloor \frac{n}{2} \rfloor + 1 \quad v_i \in \mathbb{N} \quad i = 1, \dots, k.$$

col j -esimo gruppo che contiene v_j cerchi di raggio rispettivamente di $r_1^{(j)}, \dots, r_{v_j}^{(j)}$. In ogni cerchio del j -esimo gruppo poniamo $2n_j + 1$ nodi equidistanti rispetto all'angolo. La scelta di n_j è data da:

$$\begin{aligned} n_1 &= n - v_1 + 1, \\ n - 2 &= n - 2v_1 - v_2 + 1, \\ &\vdots \\ n_k &= n - 2v_1 - \dots - 2v_{k-1} - v_k + 1. \end{aligned}$$

Si può verificare facilmente che

$$v_1 \cdot (2n_1 + 1) + \dots + v_k \cdot (2n_k + 1) = N + 1$$

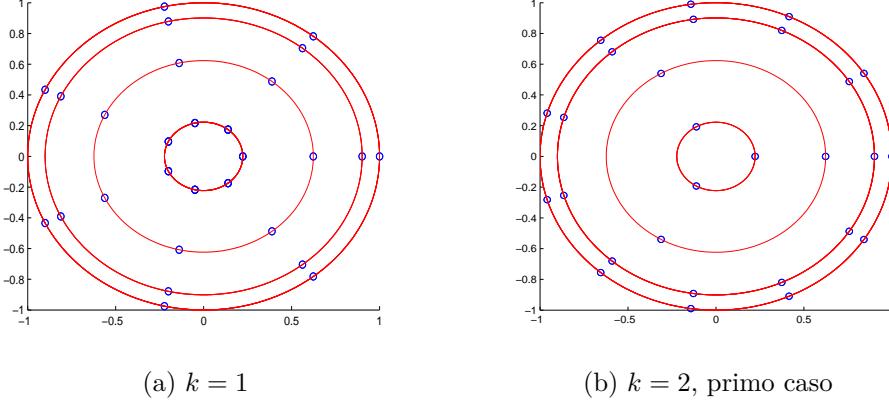
A questo punto rimangono da fare principalmente tre scelte: la lunghezza dei raggi, l'angolazione tra un fissato punto di interpolazione di ogni cerchio con un determinato semiasse e la suddivisione nei k gruppi.

5.1 La suddivisione in k gruppi

In [7], gli autori asseriscono che, al crescere del numero dei gruppi di cerchi, k , la costante di Lebesgue decresce. Vediamo nel dettaglio tale fenomeno per $n=6$. In questa situazione usiamo come configurazione per i raggi, i nodi positivi di Chebyshev-Lobatto nell'intervallo $[-1,1]$ di grado 8 o nel caso che ci sia solo un punto nel cerchio più interno, di grado 7. Sul k -simo cerchio, equidistribuiti rispetto all'angolo disponiamo $2n_k + 1$ punti, supponendo che il primo di questi punti abbia ordinata il cui valore è 0.

1. per $k=1$, abbiamo la scelta obbligata di $v_1 = 4$ e troviamo una configurazione con 7 nodi equispaziati in ciascuno dei 4 cerchi come in figura 2(a). In tal caso risulta una costante di Lebesgue di 4025,6;

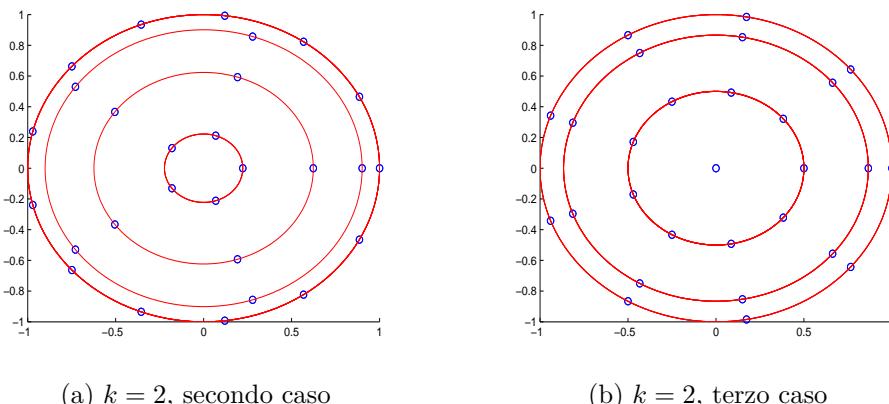
Figura 1:



2. per $k=2$ abbiamo diverse configurazioni possibili:

- se $v_1 = 2$ e $v_2 = 2$ troviamo una configurazione con 3 punti equispaziati nei due cerchi interni e 11 nei due esterni come in figura 2(b). In tal caso risulta una costante di Lebesgue di 67.2376;
- Prendendo invece $v_1 = 1$ e $v_2 = 3$ troviamo una configurazione con 13 punti equispaziati nel cerchio esterno e 5 nei tre interni come nella figura 3(a). In tal caso risulta una costante di Lebesgue di 98.1135;

Figura 2:



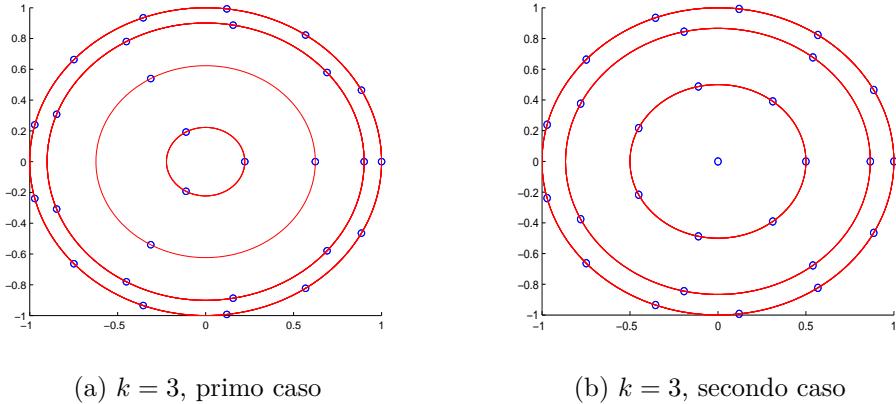
- Viceversa prendendo $v_1 = 3$ e $v_2 = 1$ otteniamo una configurazione con 9 punti equispaziati nei tre cerchi esterni e solo 1 in quello

interno come nella figura 3(b). In tal caso risulta una costante di Lebesgue di 546.47;

3. Anche con $k=3$ abbiamo diverse possibili configurazioni

- se $v_1 = 1, v_2 = 1$ e $v_3 = 2$ otteniamo una configurazione con 13 punti equispaziati nel cerchio esterno, 9 in quello intermedio e 3 nei due interni come in figura 4(a). In tal caso abbiamo una costante di Lebesgue di circa 13.52;

Figura 3:

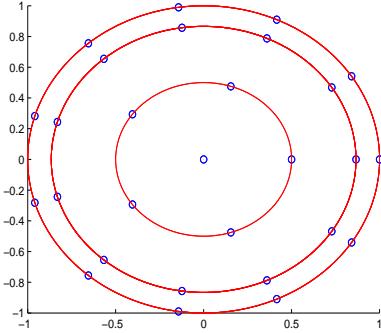


- se $v_1 = 1, v_2 = 2$ e $v_3 = 1$ troviamo 13 punti nel cerchio esterno, 7 punti nei due intermedi e 1 in quello interno come in figura 4(b).In tal caso abbiamo una costante di Lebesgue di circa 20.63;
- se $v_2 = 2, v_2 = 1$ e $v_3 = 1$ otteniamo una configurazione con 11 punti equispaziati nei due cerchi esterni, 5 in quello intermedio e 1 in quello interno come in figura 5(a). In tal caso abbiamo una costante di Lebesgue di circa 56.74;

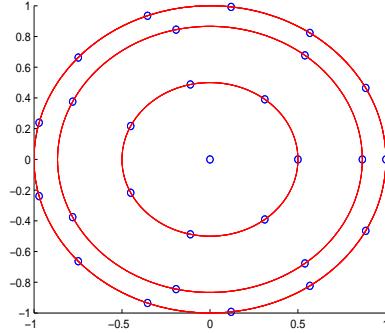
4. Per $k=4$ abbiamo un'unica configurazione possibile con $v_1 = 1, v_2 = 1, v_3 = 1$ e $v_4 = 1$ ovvero con 13 punti equispaziati nel cerchio esterno, 9 in quello successivo, 5 in quello intermedio e solo 1 in quello interno come in figura 5(b).In tal caso abbiamo una costante di Lebesgue di 5.48.

Abbiamo visto dunque che nel caso $n=6$ la costante di Lebesgue decresce al crescere del numero dei gruppi dei cerchi. Considerazioni simili possono essere fatte per tutti i gradi.

Figura 4:



(a) $k = 3$, terzo caso



(b) $k = 4$

5.2 La scelta dell'angolo

In precedenza abbiamo richiesto che il k -simo cerchio contenesse il punto $(r_k \cdot \cos(\theta), r_k \cdot \sin(\theta))$ con $\theta = 0$.

Supponiamo ora che sul k -simo disco siano piú in generale distribuiti m punti

$$x_j^{(k)} = (r_k \cdot \cos(\theta_k + s), r_k \cdot \sin(\theta_k)), \quad k = 1, \dots, m,$$

con θ_k arbitrario e $s = 2\pi/m$.

| n | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|--------|------|-------|-------|-------|-------|-------|-------|-------|-------|-------|
| EC | 1.67 | 1.99 | 2.69 | 3.32 | 3.86 | 4.42 | 5.20 | 5.80 | 6.77 | 7.85 |
| EC-opt | 1.67 | 1.99 | 2.62 | 3.30 | 3.84 | 4.35 | 5.17 | 5.78 | 8.70 | 7.80 |
| n | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| EC | 9.00 | 10.44 | 12.54 | 14.82 | 17.39 | 21.56 | 26.48 | 32.27 | 40.14 | 50.83 |
| EC-opt | 8.85 | 10.36 | 12.34 | 14.43 | 17.28 | 20.96 | 25.60 | 31.92 | 39.49 | 49.06 |

Tabella 1: Confronto tra le costanti di Lebesgue di set di punti aventi configurazioni con raggi dati dai nodi di Expanded-Chebyshev positivi di grado $n+1$ con angoli casuali con quelli ottenuti tramite ottimizzazione con GA

Risulta lecito domandarsi se una scelta opportuna di questi θ_k migliori decisamente la costante di Lebesgue rispetto alle altre.

Consideriamo le configurazioni con raggi con una lunghezza uguale alla distanza dall'origine ai nodi Expanded-Chebyshev positivi di grado $n+1$ e con

la divisione ottimale in gruppi trovata nel paragrafo precedente. Presentiamo nella tabella 5.2 un confronto tra le Costanti di Lebesgue ottenute con angoli casuali con quelle ottenute ottimizzando sull'angolo indicati rispettivamente con EC e EC-opt.

Ottimizzando sugli angoli θ_k la funzione obiettivo tramite GA, seguito da Pattern-Search, come si deduce dalla tabella, i miglioramenti sulla costante di Lebesgue sono molto limitati.

5.3 Confronto tra alcune configurazioni di raggi

Dando per accettata la configurazione ottimale trovata nella sezione 5.1, facciamo un confronto tra famiglie di punti i cui raggi corrispondano ai valori non negativi di classiche famiglie di punti in $[-1, 1]$.

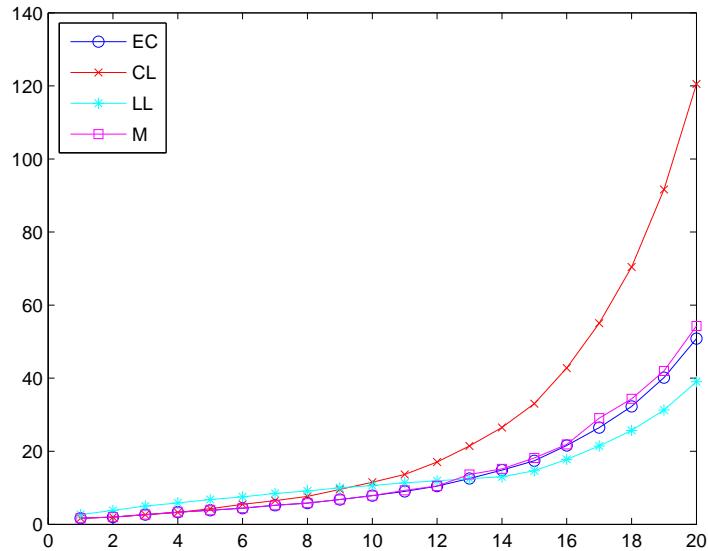
Riportiamo una tabella e un grafico con alcuni valori della costante di Lebesgue, ottenuti con la costruzione di cui abbiamo discusso sopra, utilizzando, per stabilire i raggi, le varie famiglie classiche di nodi oltre ai nodi equispaziati e ai punti di Lebesgue trovati da [2] per l'intervallo. Indicheremo i nodi Expandend Chebyshev con EC, i Chebyshev-Lobatto con CL, i Legendre-Lobatto con LL, i quasi punti di Lebesgue trovati da [2] per l'intervallo con M e i nodi equispaziati con Eq.

| n | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|-----|-------|-------|-------|-------|-------|-------|-------|-------|-------|--------|
| EC | 1.67 | 1.99 | 2.69 | 3.32 | 3.86 | 4.42 | 5.20 | 5.80 | 6.77 | 7.85 |
| CL | 1.67 | 1.99 | 2.63 | 3.24 | 4.29 | 5.48 | 6.49 | 7.63 | 9.56 | 11.52 |
| LL | 2.64 | 3.80 | 4.99 | 5.87 | 6.80 | 7.50 | 8.45 | 9.14 | 9.99 | 10.61 |
| M | 1.67 | 1.99 | 2.65 | 3.32 | 3.85 | 4.43 | 5.22 | 5.82 | 6.79 | 7.87 |
| Eq | 1.67 | 1.99 | 2.63 | 3.95 | 4.28 | 7.84 | 7.55 | 17.33 | 16.63 | 46.25 |
| n | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| EC | 9.00 | 10.44 | 12.54 | 14.82 | 17.39 | 21.56 | 26.48 | 32.27 | 40.14 | 50.83 |
| CL | 13.66 | 17.06 | 21.47 | 26.50 | 33.01 | 42.77 | 55.06 | 70.45 | 91.62 | 120.50 |
| LL | 11.33 | 11.91 | 12.52 | 12.99 | 14.69 | 17.76 | 21.45 | 25.69 | 31.30 | 39.02 |
| M | 9.29 | 10.49 | 13.63 | 15.16 | 18.17 | 21.89 | 29.08 | 34.37 | 41.99 | 54.28 |
| Eq | 42.07 | 139.3 | 118.6 | 445 | 360 | 1487 | 1145 | 5100 | 3749 | 17842 |

Tabella 2: Confronto tra le costanti di Lebesgue di set di punti aventi configurazioni con raggi dati da famiglie di nodi per gradi n compresi tra 1 e 20.

La tabella evidenzia come i punti equispaziati determinino come nel caso dell'intervallo un set di punti con un'alta costante di Lebesgue. Gli Expanded

Figura 5: Confronto tra le costanti di Lebesgue di set di punti aventi configurazioni con raggi dati da famiglie di nodi classiche.



Chebyshev e i punti di Lebesgue di [2] forniscono sets di punti con una costante simile: i primi risultano leggermente migliori per il nostro problema. In ogni caso, come si vede nel grafico i Expanded Chebyshev, i punti di [2] e i Legendre-Lobatto hanno un ottimo andamento col crescere di n .

5.4 Ottimizzando sia su raggi sia su angoli

Illustriamo ora i risultati ottenuti ottimizzando tramite l'algoritmo genetico GA e Pattern-Search, prima sui raggi e poi sugli angoli. In [7] infatti si afferma che non risulta fondamentale ottimizzare il valore delle variabili angolari: in parte abbiamo verificato questa affermazione in 5.2. Abbiamo però deciso di non trascurare completamente il valore delle variabili angolari ma semplicemente di considerarle in un secondo momento. In tal modo il tempo di esecuzione del nostro codice è risultato dell'ordine dei minuti per i gradi piu' bassi, mentre per quelli piu' alti e' stato comunque inferiore alle 10 ore. Durante tali processi abbiamo usato la divisione in gruppi ritenuta ottima nella sezione 5.1. Consci del fatto che l'ottimizzazione sugli angoli avrebbe inciso poco sul risultato finale, abbiamo utilizzato in questo caso un dato iniziale casuale. Riportiamo ora i risultati trovati da [1] e [2] confrontandoli con

quelli trovati da noi in questa sezione: nella tabella e nel grafico indichiamo con BSV quelli trovati da [1], con M quelli trovati da [2] e con CYIB-Opt quelli trovati da noi ottimizzando, seguendo alcune idee presenti in [7].

| Grado | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|----------|------|------|-------|-------|-------|-------|-------|-------|-------|-------|
| BSV | 1.67 | 1.99 | 2.47 | 2.97 | 3.50 | 4.30 | 5.08 | 5.43 | 6.73 | 7.62 |
| M | 1.67 | 1.99 | 2.47 | 2.97 | 3.49 | 4.10 | 5.00 | 5.38 | 6.54 | 7.14 |
| CYIB-Opt | 1.67 | 1.99 | 2.47 | 2.97 | 3.49 | 4.26 | 4.76 | 5.32 | 5.90 | 6.50 |
| Grado | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| BSV | 8.48 | 9.65 | 11.98 | 12.39 | 14.13 | 17.44 | 18.88 | 27.17 | 28.29 | 29.68 |
| M | 7.84 | 8.97 | 10.14 | 11.14 | 13.08 | 14.72 | 14.92 | 16.04 | 18.27 | 20.31 |
| CYIB-Opt | 7.20 | 7.93 | 8.73 | 9.64 | 10.65 | 11.82 | 13.13 | 14.63 | 16.46 | 18.45 |

Tabella 3: confronto tra le costanti di Lebesgue trovate da noi con alcune presenti in letteratura

Come si nota dalle tabelle sulle costanti di Lebesgue, per gradi bassi i nuovi set di punti non migliorano gli ottimi risultati precedentemente ottenuti in [1] e [2], mentre hanno costanti di Lebesgue significativamente più basse all'aumentare del grado.

5.4.1 Confronto con la configurazione di raggi particolari

Naturale ora domandarsi se la configurazione ottenuta ottimizzando sui raggi è simile a una di quelle ottenute usando le famiglie di nodi classiche per l'intervallo.

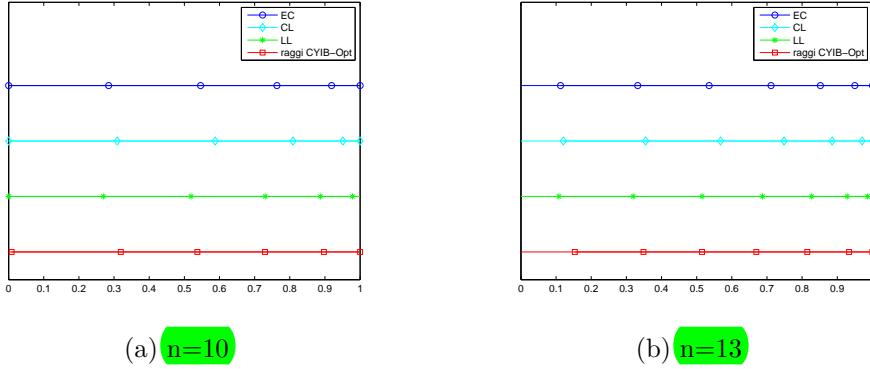


Figura 6: confronto tra raggi

Nei grafici illustriamo solo i casi $n=10$ e $n=13$, pur avendo verificato che ciò accade anche più in generale per $n \leq 20$. Vediamo che tutti i nodi riportati nel grafico tendono ad essere più fitti verso 1 e più radi verso 0. I raggi che abbiamo ottenuto tramite l'ottimizzazione pur prossimi alle altre classiche famiglie di punti univariati, non sembrano coincidere con una delle stesse.

5.4.2 Confronto con altri autori

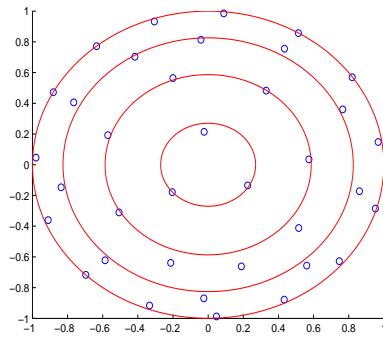
In questa sezione confrontiamo le configurazioni da noi ottenute con quelle presentate in [1]. Osserviamo che mentre in [1] i punti non hanno vincoli particolari, eccetto di stare nel disco unitario, quelli da noi introdotti, seguendo [7], richiedono di stare in cerchi concentrici e di avere una opportuna equispaziatura rispetto agli angoli.

Nelle figure 7, 8 e 9, confrontiamo le loro distribuzioni per i casi $n = 7$, $n = 13$ e $n = 20$. Denoteremo i punti descritti in [1] con BSV mentre quelli di questo lavoro con CYIB-Opt.

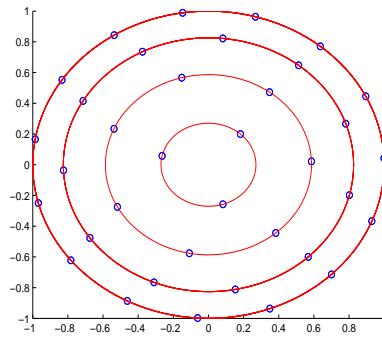
Le distribuzioni sono simili, con punti che in ?? sono approssimativamente su cerchi e equispaziati rispetto agli angoli, disposti in k poligoni regolari con numero di vertici uguale al numero dei nodi nel rispettivo cerchio della nostra configurazione.

Nella Figura 10, confrontiamo le Costanti di Lebesgue proposte in [1], con quelle in [2] (denotate con M) e ottenute dalla nostra ottimizzazione.

Figura 7: Grado $n = 7$. A sinistra: distribuzione dei punti ottenuti in [1], denotati con BSV. A destra: distribuzione dei punti ottenuti in questo lavoro, denotati con CYIB-Opt.

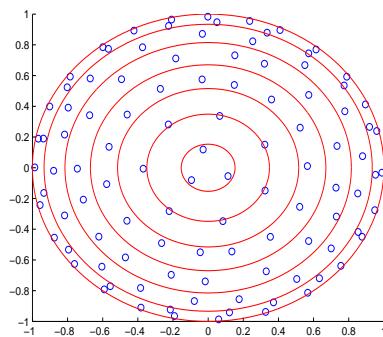


(a) BSV

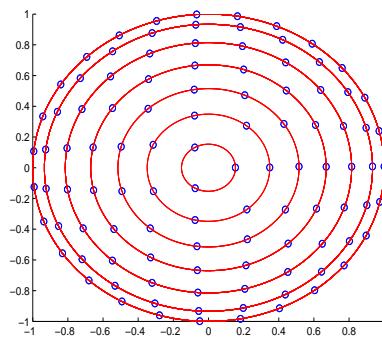


(b) CYIB-Opt

Figura 8: Grado $n = 13$. A sinistra: distribuzione dei punti ottenuti in [1], denotati con BSV. A destra: distribuzione dei punti ottenuti in questo lavoro, denotati con CYIB-Opt.



(a) BSV



(b) CYIB-Opt

Figura 9: Grado $n = 20$. A sinistra: distribuzione dei punti ottenuti in [1], denotati con BSV. A destra: distribuzione dei punti ottenuti in questo lavoro, denotati con CYIB-Opt.

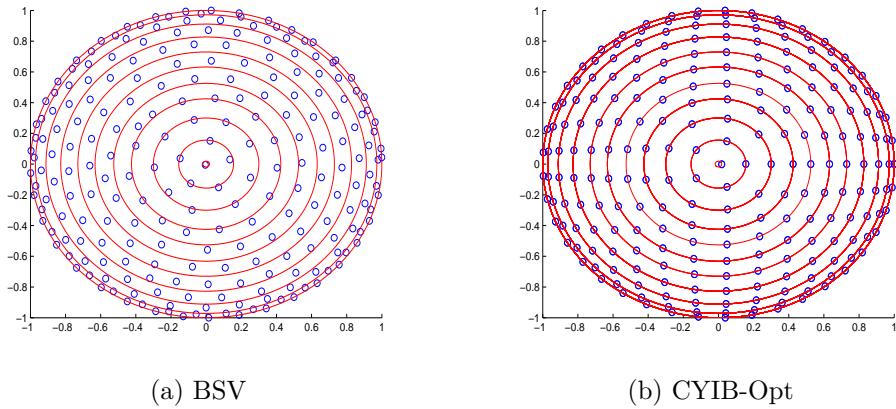
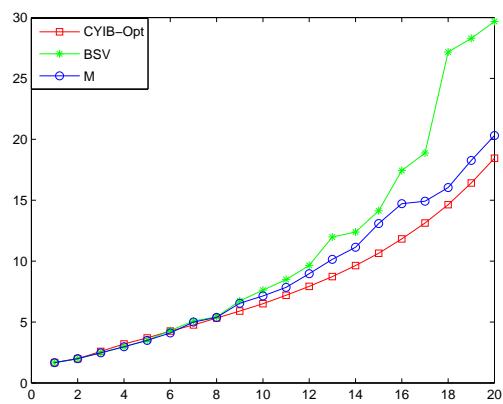


Figura 10: andamento delle costanti di Lebesgue di CYIB-Opt, di BSV e di M



Per completezza riportiamo anche delle tabelle (rispettivamente 4, 5) con i valori del determinante della matrice di Vandermonde V (rispetto la base di Logan-Shepp), del condizionamento di questa sia per i punti trovati in questa sezione sia per quelli trovati in [2] e [1]; inoltre riportiamo i dati trovati da [1] circa i punti di Fekete.

Come si vede dalle tabelle, i determinanti di V hanno valori prossimi a quelli dei punti di Fekete, ciò nonostante restano ben condizionate.

| determinante di Vandermonde | | | | | |
|-----------------------------|-------------|-------------|-------------|-------------|-------------|
| Grado | 1 | 2 | 3 | 4 | 5 |
| BSV | $4.95E-1$ | $2.12E0$ | $2.97E+1$ | $2.07E+3$ | $7.03E+5$ |
| M | $4.95E-1$ | $2.12E0$ | $2.97E+1$ | $2.11E+3$ | $7.11E+5$ |
| CYIB-Opt | $4.95E-1$ | $3.69E-1$ | $2.91E+1$ | $1.21E+3$ | $1.32E+6$ |
| BSV-F | $4.95E-01$ | $2.13E+00$ | $3.44E+01$ | $3.49E+03$ | $2.03E+06$ |
| Grado | 6 | 7 | 8 | 9 | 10 |
| BSV | $1.10E+10$ | $4.15E+13$ | $2.87E19$ | $1.74E+25$ | $2.28E+34$ |
| M | $1.03E+10$ | $4.14E+13$ | $2.96E+19$ | $2.29E+25$ | $1.52E+34$ |
| CYIB-Opt | $8.09E+9$ | $2.27E+14$ | $7.18E+18$ | $1.02E+26$ | $1.43E+34$ |
| BSV-F | $1.13E+10$ | $4.20E+14$ | $1.81E+20$ | $1.02E+27$ | $6.72E+34$ |
| Grado | 11 | 12 | 13 | 14 | 15 |
| BSV | $2.96E+38$ | $1.06E+48$ | $4.08E+58$ | $2.34E+76$ | $2.38E+89$ |
| M | $2.52E+43$ | $8.91E+52$ | $2.30E+60$ | $7.24E+75$ | $1.65E+89$ |
| CYIB-Opt | $5.76E+42$ | $3.82E+52$ | $2.99E+63$ | $6.70E+73$ | $1.90E+85$ |
| BSV-F | $4.88E+43$ | $6.51E+53$ | $7.11E+64$ | $3.04E+77$ | $1.57E+91$ |
| Grado | 16 | 17 | 18 | 19 | 20 |
| BSV | $3.51E+97$ | - | - | - | - |
| M | $4.36E+99$ | $8.42E+119$ | $1.76E+137$ | $3.93E+155$ | $7.96E+176$ |
| CYIB-Opt | $2.33E+103$ | $3.71E+116$ | $7.90E+135$ | $2.58E+153$ | $6.75E+173$ |
| BSV-F | $6.74e+105$ | $3.22E+122$ | $9.56E+139$ | $8.85E+158$ | $3.72E+179$ |

Tabella 4: Confronto tra i determinanti della matrice di Vandermonde per i set di punti BSV (cf. [1]), M (cf. [2]), i nuovi CYIB-Opt oltre i punti BSV-F (di [1] sui punti di Fekete).

5.4.3 Confronto con la teoria

In [9] si ricorda che la stima teorica della costante di Lebesgue per i punti di Lebesgue nel disco: $\Lambda_n = O(\sqrt{n+1})$. Vogliamo ora confrontare la costante di Lebesgue dei punti trovati con questa puramente teorica.

| Numero di condizionamento della matrice di Vandermonde. | | | | | |
|---|----------|----------|----------|----------|----------|
| Grado | 1 | 2 | 3 | 4 | 5 |
| BSV | 2.98E0 | 5.33E0 | 1.08E+1 | 1.32E+1 | 1.55E+1 |
| M | 2.98E0 | 5.13E0 | 1.08E+1 | 1.49E+1 | 2.12E+1 |
| CYIB-Opt | 2.98E0 | 5.43E0 | 1.05E+1 | 1.43E+1 | 1.93E+1 |
| BSV-F | 2.99E+00 | 5.33E+00 | 1.06E+01 | 1.39E+01 | 1.65E+01 |
| Grado | 6 | 7 | 8 | 9 | 10 |
| BSV | 2.84E+1 | 2.55E+1 | 3.58E+1 | 5.34E+1 | 6.37E+1 |
| M | 2.84E+1 | 2.83E+1 | 3.53E+1 | 5.52E+1 | 6.95E+1 |
| CYIB-Opt | 2.77E+1 | 3.35E+1 | 4.33E+1 | 4.48E+1 | 6.33E+1 |
| BSV-F | 2.65E+01 | 3.28E+01 | 4.33E+01 | 4.80E+01 | 6.09E+01 |
| Grado | 11 | 12 | 13 | 14 | 15 |
| BSV | 7.85E+1 | 1.06E+1 | 9.55E+1 | 1.18E+2 | 1.52E+2 |
| M | 7.82E+1 | 1.07E+1 | 1.24E+2 | 1.49E+2 | 1.59E+2 |
| CYIB-Opt | 7.12E+1 | 8.06E+1 | 1.16E+2 | 3.66E+3 | 1.73E+4 |
| BSV-F | 6.88E+01 | 7.40E+01 | 1.09E+02 | 1.12E+02 | 1.25E+02 |
| Grado | 16 | 17 | 18 | 19 | 20 |
| BSV | 1.43E+2 | - | - | - | - |
| M | 2.12E+2 | 2.12E+2 | 2.66E+2 | 2.96E+2 | 3.44E+2 |
| CYIB-Opt | 1.65E+2 | 1.97E+2 | 2.39E+2 | 1.34E+3 | 4.05E+2 |
| BSV-F | 2.32E+02 | 2.42E+02 | 2.22E+02 | 3.60E+02 | 2.44E+02 |

Tabella 5: Confronto tra i condizionamenti della Matrice di Vandermonde per i set di punti BSV (cf. [1]), di M (cf. [2]) e i nuovi CYIB-Opt, oltre i punti BSV-F (di [1] sui punti di Fekete).

Nel grafico 12(a) abbiamo rappresentato l'andamento di $\Lambda_n = \sqrt{n+1}$ e le costanti di Lebesgue $\tilde{\Lambda}_n$ trovate nella sezione 5.4 (indicate con CIYT). Si vede subito che, pur a meno di una costante moltiplicativa, l'andamento non sembra essere lo stesso, specialmente per gradi prossimi a 20.

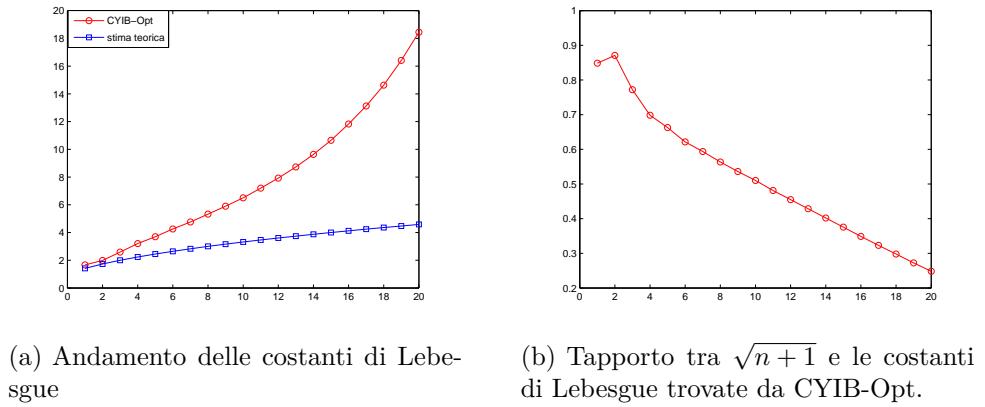
Se l'andamento fosse asintoticamente uguale, il rapporto

$$\frac{\sqrt{n+1}}{\tilde{\Lambda}_n}$$

tenderebbe al crescere di n ad una costante.

Proponiamo la tabella 5.4.3 e il grafico 12(b) circa tali considerazioni.

Figura 11: Confronto tra la costante di Lebesgue dei punti CYIB-Opt con l'andamento asintotico dei punti di Lebesgue.



| grado | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
|------------------------------------|------|------|------|------|-------|-------|-------|-------|-------|-------|
| $\sqrt{n+1}$ | 1.41 | 1.73 | 2.00 | 2.24 | 2.45 | 2.65 | 2.83 | 3.00 | 3.16 | 3.32 |
| $\tilde{\Lambda}_n$ | 1.67 | 1.99 | 2.59 | 3.20 | 3.70 | 4.26 | 4.76 | 5.32 | 5.90 | 6.50 |
| $\sqrt{n+1}/\tilde{\Lambda}_n = C$ | 0.85 | 0.87 | 0.77 | 0.70 | 0.66 | 0.62 | 0.59 | 0.56 | 0.54 | 0.51 |
| grado | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| $\sqrt{n+1}$ | 3.46 | 3.61 | 3.74 | 3.87 | 4.00 | 4.12 | 4.24 | 4.36 | 4.47 | 4.58 |
| $\tilde{\Lambda}_n$ | 7.20 | 7.93 | 8.73 | 9.64 | 10.65 | 11.82 | 13.13 | 14.63 | 16.42 | 18.45 |
| $\sqrt{n+1}/\tilde{\Lambda}_n = C$ | 0.48 | 0.45 | 0.43 | 0.40 | 0.38 | 0.35 | 0.32 | 0.30 | 0.27 | 0.25 |

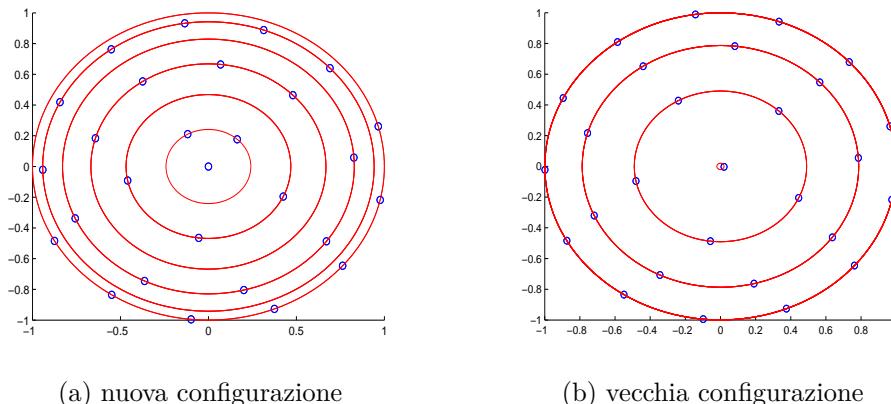
Tabella 6: Confronto tra le costanti di Lebesgue di CYIB-Opt con $\sqrt{n+1}$.

6 Possibili sviluppi

Ci poniamo il problema di trovare a partire da CYIB-Opt, configurazioni unisolventi che abbiano migliori costanti di Lebesgue.

Di conseguenza, alleggeriamo i vincoli, cercando configurazioni vicine a quelle di CYIB-Opt con migliori costanti di Lebesgue. Usiamo come punto di partenza la configurazione trovata nella sezione precedente. Come abbiamo visto i nodi sono disposti in cerchi concentrici. Ora consideriamo un determinato cerchio di raggio r che contiene $2n_k + 1$ nodi. Disponiamo i primi n_k nodi in un cerchio di raggio r_{k_1} e gli altri $n_k + 1$ nodi in un cerchio di raggio r_{k_2} senza modificarne l'angolo: nella figura 12 è presentato il confronto per $n = 6$ tra la vecchia configurazione e quella nuova data con raggi r_{k_i} uguali ai nodi di Chebyshev di grado 13. Ovviamente ponendo sia r_{k_1} , sia r_{k_2} uguali

Figura 12: Confronto oer n=6 tra vecchia e nuova configurazione



a r_k per ogni k , tale configurazione risulterà uguale alla vecchia.

Agendo in questo modo abbiamo visto che abbiamo disposto tutti i nodi in $n+1$ cerchi. A questo punto ottimizziamo tramite `ga` o `patternsearch` identificando come incognite i raggi degli $n+1$ cerchi. Utilizzando come dato iniziale raggi tali da avere la configurazione trovata nella sezione 5.4, siamo certi di non peggiorare il risultato. Proponiamo ora i risultati trovati con la tabella 7 e dei grafici circa tali nuove configurazioni.

| Grado | 6 | 7 | 8 | 9 | 10 | 11 | 12 | |
|-----------|-------|-------|--------|--------|--------|--------|--------|--------|
| CYIB-opt | 4.258 | 4.764 | 5.324 | 5.900 | 6.500 | 7.200 | 7.927 | |
| CYIB-opt2 | 4.188 | 4.753 | 5.284 | 5.871 | 6.490 | 7.183 | 7.899 | |
| Grado | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 |
| CYIB-opt | 8.729 | 9.636 | 10.650 | 11.824 | 13.130 | 14.632 | 16.415 | 18.448 |
| CYIB-opt2 | 8.723 | 9.623 | 10.624 | 11.761 | 13.079 | 14.604 | 16.286 | 18.139 |

Tabella 7: confronto tra le costanti di Lebesgue trovate in questa sezione (indicate con CYIB-opt2) con quelle trovate nella sezione 5.4 (indicate con CYIB-opt)

La Tabella 7 mostra che con questa tecnica abbiamo ottenuto piccoli miglioramenti, seppur ancora modesti. D'altra parte indica che i punti CYIB-opt non sono ottimali (cosa nota perché si sa teoricamente che la distribuzione di nodi sul cerchio di CYIB non e' ottimale).

A Alcuni dati

Riportiamo ora i raggi e gli angoli trovati nella sezione 5.4: da questi facilmente si possono recuperare i singoli punti.

| n=1 | |
|--------------------|---------------------|
| raggi | angoli |
| 1.0000000000000000 | -0.5360658176023683 |

| n=2 | |
|--------------------|---------------------|
| raggi | angoli |
| 0.0501412131798916 | 0.5153562929508685 |
| 1.0000000000000000 | -0.5360653836472800 |

| n=3 | |
|--------------------|--------------------|
| raggi | angoli |
| 0.4679820687982619 | 0.6121850120941608 |
| 0.9922533482313156 | 0.7619954732948395 |

| n=4 | |
|--------------------|--------------------|
| raggi | angoli |
| 0.0597749650478364 | 0.8496021924106213 |
| 0.6673011466621492 | 0.0720945171727454 |
| 0.9741323143243790 | 0.0029377280900289 |

| n=5 | |
|--------------------|--------------------|
| raggi | angoli |
| 0.3331279963027359 | 0.8346023697260052 |
| 0.7482138764188286 | 0.6703890566245899 |
| 0.9935782036455215 | 0.6279469971218775 |

| n=6 | |
|--------------------|---------------------|
| raggi | angoli |
| 0.0204461514949799 | -0.1565667484475872 |
| 0.4902358284444396 | 0.8250282120957410 |
| 0.7865588559125651 | 0.7685327244649982 |
| 0.9992146641016007 | 0.7473053362691920 |

| n=7 | |
|--------------------|--------------------|
| raggi | angoli |
| 0.2701259148120880 | 0.8325245107173920 |
| 0.5864102960109711 | 0.9354058859586716 |
| 0.8258213682651520 | 0.3292158101320267 |
| 0.9996177076578140 | 0.4618112205743790 |

| n=8 | |
|--------------------|--------------------|
| raggi | angoli |
| 0.0073638911008835 | 2.6760407668590545 |
| 0.3783044869422912 | 0.3751779296875000 |
| 0.6390220703125000 | 0.8143196695327759 |
| 0.8572000000000000 | 0.5116545382499695 |
| 1.0000000000000000 | 1.0861805424690247 |

| n=9 | |
|--------------------|---------------------|
| raggi | angoli |
| 0.2281401437282562 | 0.0219051558017731 |
| 0.4770650120735169 | -0.0020005364418030 |
| 0.6841000000000000 | 0.0131000000000000 |
| 0.8769034719705582 | 0.0170064671039581 |
| 0.9966717937946320 | 0.0147901950359344 |

| n=10 | |
|--------------------|--------------------|
| raggi | angoli |
| 0.0084579621315002 | 0.1506734061717987 |
| 0.3192393871545792 | 0.0009898641586304 |
| 0.5371251961708069 | 0.0163923590898514 |
| 0.7296000000000000 | 0.0101779296875000 |
| 0.8971426769256592 | 0.0086548826694489 |
| 1.0000000000000000 | 0.0087738237380981 |

| n=11 | |
|--------------------|--------------------|
| raggi | angoli |
| 0.1618626478672028 | 0.0922528561115265 |
| 0.4029310242176056 | 0.0203948492765427 |
| 0.5862985115766526 | 0.0087529587268829 |
| 0.7637140666961670 | 0.0134395029783249 |
| 0.9113000000000000 | 0.0102000000000000 |
| 0.9999000000000000 | 0.0077000000000000 |

| n=12 | |
|--------------------|---------------------|
| raggi | angoli |
| 0.0134378658771515 | -0.8066954908370971 |
| 0.2585437647104263 | -0.0033923553586006 |
| 0.4687437332153320 | 0.0158562098979950 |
| 0.6298000000000000 | 0.0109114589929581 |
| 0.7917879151582717 | 0.0089289431333542 |
| 0.9231000000000000 | 0.0116005364418030 |
| 0.999999850988388 | 0.0132779296875000 |

| n=13 | |
|--------------------|--------------------|
| raggi | angoli |
| 0.1532199110031128 | 0.0075092447996140 |
| 0.3487236098766327 | 0.0046431737422943 |
| 0.5153000000000000 | 0.0152833520174026 |
| 0.6699001473188401 | 0.0102781714916229 |
| 0.8147649077653885 | 0.0087838654279709 |
| 0.9337370276927948 | 0.0090777542591095 |
| 0.9999879896640778 | 0.0086549107789993 |

| n=14 | |
|--------------------|---------------------|
| raggi | angoli |
| 0.0146180314064026 | -0.0339168361663818 |
| 0.2272806511878967 | 1.0319699355125427 |
| 0.4164489120483398 | 1.0177344057559967 |
| 0.5565876286506652 | 1.0087047964811324 |
| 0.7038138844966888 | 1.0077924769401549 |
| 0.8342079223394394 | 1.0093376274585724 |
| 0.9423193498373031 | 1.0104341155290604 |
| 1.0000000000000000 | 1.0113154924631118 |

| n=15 | |
|--------------------|--------------------|
| raggi | angoli |
| 0.1190022724151611 | 0.0035097047209740 |
| 0.2997729266166687 | 0.0093276866912842 |
| 0.4583072734594345 | 0.0093807195901871 |
| 0.5963000000000001 | 0.0159242011785507 |
| 0.7292202208757400 | 0.0125567138195038 |
| 0.8518155436038971 | 0.0063822097063065 |
| 0.9487023692846298 | 0.0020596561193466 |
| 1.0000000000000000 | 0.0012639452934265 |

| n=16 | |
|--------------------|--------------------|
| raggi | angoli |
| 0.0116734451293945 | 0.8913270408630372 |
| 0.2038372586488724 | 0.0493280541419983 |
| 0.3634565512657166 | 0.0184412667274475 |
| 0.5043260319948196 | 0.0104779296875000 |
| 0.6289943351984024 | 0.0096051114320755 |
| 0.7551768913507462 | 0.0083206035614014 |
| 0.8672801625013351 | 0.0069510415554047 |
| 0.9541399666070938 | 0.0062580931901932 |
| 0.9999985545873642 | 0.0061262216329575 |

| n=17 | |
|--------------------|--------------------|
| raggi | angoli |
| 0.1112140186071396 | 0.0072829410390437 |
| 0.2738825385570526 | 0.0013713313152313 |
| 0.4095260025262832 | 0.0015158725507736 |
| 0.5393581856489181 | 0.0010482209087372 |
| 0.6573200420618057 | 0.0007965859474182 |
| 0.7735519586563110 | 0.0003916458013535 |
| 0.8769343918800354 | 0.0000829594209790 |
| 0.9565776648521424 | 0.0001873411238194 |
| 0.9971023841857910 | 0.0007979187633514 |

| n=18 | |
|--------------------|---------------------|
| raggi | angoli |
| 0.0117608038187027 | -0.2516008030176163 |
| 0.1806577165842056 | -0.0020822134315968 |
| 0.3392438639402390 | 0.0032626153469086 |
| 0.4541558593750000 | 0.0045464333534241 |
| 0.5753969168186188 | 0.0063139569520950 |
| 0.6845059669017792 | 0.0042043236970901 |
| 0.7958881119251251 | 0.0034288144350052 |
| 0.8913935227394104 | 0.0031170035719872 |
| 0.9635165497779846 | 0.0021234432458878 |
| 0.9999994784593582 | 0.0000690479159355 |

| n=19 | |
|--------------------|---------------------|
| raggi | angoli |
| 0.0975661797761917 | -0.2516008030176163 |
| 0.2375609057903290 | -0.0020822134315968 |
| 0.3763999305725098 | 0.0032626153469086 |
| 0.4949106096267700 | 0.0045464333534241 |
| 0.6023279809951783 | 0.0063139569520950 |
| 0.7096699409723282 | 0.0042043236970901 |
| 0.8109961853027344 | 0.0034288144350052 |
| 0.9020362064361572 | 0.0031170035719872 |
| 0.9678093081474304 | 0.0021234432458878 |
| 1.0000000000000000 | 0.0000690479159355 |

| n=20 | |
|--------------------|---------------------|
| raggi | angoli |
| 0.0186744861125946 | -0.0749280889987946 |
| 0.1560149214744568 | 0.0040022382020950 |
| 0.2999794526576996 | 0.0005867928266525 |
| 0.4247275044918060 | 0.0001686215400696 |
| 0.5255104277610779 | 0.0001198649406433 |
| 0.6320530430555343 | 0.0000456720590591 |
| 0.7298337890625000 | 0.0000000000000000 |
| 0.8272896484375000 | 0.0000312328338623 |
| 0.9113364103078843 | 0.0000000000000000 |
| 0.9716422878026962 | 0.0001283288002014 |
| 0.9999992996454239 | 0.0005152076482773 |

B Codici

Forniamo in questa sezione alcuni dei codici utilizzati in questo lavoro.

```

1 %-----
2 %/
3 %
4 %demo optimization
5 %/
6 %

```

```

7  clear all;

9  degv=[15]; % deg: WAM DEGREE.
10   extraction_method=1; % extraction_method: 1: QR 2:LU.

11  family_type=2; % 1:Fekete
12      % 2:Costante Lebesgue

15  starting_point=0; % 0: punti di Lebesgue di Briani
16      % 1: punti di Fekete di Briani

17  mildness_vector=8;

19  ottimizzazione=0; % 1: CYIBopt.
20      % 2: CYIBopt2.

23  do_plots=0;

25  %-----
26  % ..... HERE THE MAIN PROGRAM BEGINS .....
27  %-----

29  s='set_disk_leb'; % PUT THE NAME OF YOUR FUNCTION HERE
30  diaryname=cat(2,'diary_',date,'_',s,'.m');

33  diary on; diary(diaryname);

35  fprintf('\n\t');
36  uu=cat(2,'function_pts=',s,'(deg)');
37  fprintf(uu);

39  for kk=1:length(degv)
40
41      deg=degv(kk);

43

45  %-----
46  % INITIAL SET.
47  %-----
48

49

```

```

        switch starting_point
51         case 0
52             pts_start=set_disk_leb(deg);
53         case 1
54             pts_wam=disk_wam(deg);
55             V_wam= koornwinder_typeII(deg,pts_wam,-0.5);
56             [pts_start,w_disk,ind,C,P]=computing_fekete_points(... 
57                 pts_wam,V_wam,2,[],extraction_method);
58         end
59
60 %-----
61 % OPTIMIZATION.
62 %-----
63
64     pts_in=pts_start;
65     cputiming1=cputime;
66     for ii=1:length(mildness_vector)
67         mildness=mildness_vector(ii);
68         if ottimizzazione==0
69             pts_out=pts_in;
70             value=stima(pts_out,mildness,deg,0);
71         elseif ottimizzazione== 1
72             [pts_out,value]=CYIBopt(deg,... 
73                 pts_in,family_type,mildness);
74                 disegno(pts_out,kk);
75         else
76             [pts_out,value]=CYIBopt2(deg,... 
77                 pts_in,family_type,mildness);
78         end
79         pts_in=pts_out;
80     end
81     cputiming2=cputime;

82 %-----
83 % STATS.
84 %-----
85 timing=cputiming2-cputiming1;
86 [lebs,dets,conds]=do_stats(deg,pts_out,timing);

87
88 %-----
89 % PLOTS SENZA CERCHI
90 %-----

```

```

93     if do_plots == 1
94         degrees=linspace(0,2*pi,1000); degrees=degrees';
95         pts_bnd=[cos(degrees) sin(degrees)];
96         plot(pts_bnd(:,1),pts_bnd(:,2),'b-',...
97             pts_out(:,1),pts_out(:,2),'r+');
98     end
99
100    end
101
102
103
104
105
106
107
108
109 %-----
110 %
111 %
112 %CYIBopt
113 %
114 %
115
116
117 function [pts_out,value]=CYIBopt(deg,...
118     pts_in,family_type,mildness)
119
120
121 % DATA BEFORE STARTING THE PROCESS.
122 dim_in=size(pts_in,1);
123 x_in=[pts_in(:,1); pts_in(:,2)];
124
125 % OPTIONS.
126 [degree_leb,maxfun,maxtol,tolx]=option_settings(...%
127     deg,mildness);
128 switch family_type
129     case 2
130         pts_leb=disk_wam(degree_leb);
131         Vleb= koornwinder_typeII(deg,pts_leb,-0.5);
132     otherwise
133         Vleb=[];
134 end

```

```

%-----
137 %Creazione configurazione
%-----
139 numerocerchi=fix(deg/2)+1;
gruppi=ones(numerocerchi,1);
141 nnodi(1)=deg-gruppi(1)+1;
lun=size(gruppi,1);
143 nnodi=zeros(lun,1);
for i=0:lun-1
145 nnodi(lun-i)=deg+1-gruppi(i+1)-2*norm(gruppi(1:i),1);
end
147 nnodi=2*nnodi+ones(lun,1);
ragv=ones(numerocerchi,1)/numerocerchi;
149
for kk=1:numerocerchi
151 ragv(kk)=kk*ragv(kk);
end
153
if rem(deg,2)==0;
155 rcheb=expanded_chebyshev(2*numerocerchi-1);
lun=length(rcheb);
157 m=(lun-1)/2+1;
ragv=rcheb(m:lun);
159 else
rcheb=expanded_chebyshev(2*numerocerchi);
161 lun=length(rcheb);
m=lun/2+1;
ragv=rcheb(m:lun);
end
165 theta_out=zeros(numerocerchi);
ragv=ragv';
167 x_inv=cerchio(theta_out,ragv,gruppi,nnodi);

169 %-----
% OPTIMIZATION PROCESS.
171 %-----
173 A=[]; B=[]; Aeq=[]; Beq=[]; LB=zeros(numerocerchi,1);
UB=2*pi*ones(numerocerchi,1);
A2=eye(numerocerchi);
175 for i=1:nummerocerchi-1
A2(i,i+1)=-1;
177 end

```

```

179 time1=cputime;
    options=gaoptimset('PopulationType','Doublevector',...
181     'PopulationSize',[20 20 20],...
        'Generations',100,'CrossoverFcn',...
183     @crossovertwopoint,'CrossoverFraction',0.8,
        'FitnessScalingFcn',...
185     @fitscalingrank,'SelectionFcn',@selectionroulette,'MutationFcn',...
        @mutationadaptfeasible,'EliteCount',2,'Display','off',...
187     'StallGenLimit',100,....
        'TolFun',maxfun,'TolCon',tolx,'InitialPopulation',ragv', ...
189     'MigrationDirection','both','MigrationInterval',20,
        'HybridFcn',@patternsearch);

191 [ragv,vout] = ga(@(ragv1) opt_target(theta_out,ragv1, ...
193     gruppi,nnodi,deg,family_type,Vleb),...
        numerocerchi,[],[],Aeq,Beq,LB,UB/(2*pi),[],options);
195 options=gaoptimset('PopulationType','Doublevector',...
        'PopulationSize',[20 20 20],...
        'Generations',100,'CrossoverFcn',...
        @crossovertwopoint,'CrossoverFraction',...
199     0.8,'FitnessScalingFcn',...
        @fitscalingrank,'SelectionFcn',...
        @selectionroulette,'MutationFcn',...
        @mutationadaptfeasible,'EliteCount',2,'Display','off',...
203     'StallGenLimit',100,....
        'TolFun',maxfun,'TolCon',tolx,'InitialPopulation',theta_out, ...
205     'MigrationDirection','both','MigrationInterval',20,
        'HybridFcn',@patternsearch);
207 [theta_out,vout] = ga(@(theta_out1) opt_target(theta_out1,ragv, ...
        gruppi,nnodi,deg,family_type,Vleb),...
        numerocerchi,A,B,Aeq,Beq,[],[],[],options);

211 time2=cputime;
213
    fprintf('\n\traggi\n');
215 fprintf('\t%3.16f,',ragv);
    fprintf('\n\tangoli\n');
217 fprintf('\t%3.16f,',theta_out);
    fprintf('\n\tgruppi\n');
219 fprintf('\t%3.0f,',gruppi);
    fprintf('\n\tnnodi\n');
221 fprintf('\t%3.0f,',nnodi);

```

```

x_out=cerchio(theta_out,ragv,gruppi,nnodi);
223
fprintf ('\n\n\tCostante di Lebesgue\t%6.16f\n',vout);
225 pts_out=x_out;
%-----
227 % SEE IF THERE ARE IMPROVEMENTS.
%-----
229
switch family_type
231
case 1 % FEKETE.
233     value_out= 1000001;

235
case 2 % LEBESGUE.
237     pts_leb=disk_wam(250);
238     V_leb= koornwinder_typeII(deg,pts_leb,-0.5);
239
240     V_pts_in=koornwinder_typeII(deg,pts_in,-0.5);
241     value_in=norm(V_pts_in'\V_leb',1);

243 %         pts_out=[x_out(1:dim_in,1) x_out(dim_in+1:end,1)];
244     V_pts_out = koornwinder_typeII(deg,pts_out,-0.5);
245     value_out=norm(V_pts_out'\V_leb',1);

247     if abs(value_in) < abs(value_out)
248         x_out=x_in; value=value_in;
249         fprintf('fail\n\n\n')
250     else
251         value=value_out;
252     end
253
otherwise
255     value_in= opt_target(x_in,deg,family_type,Vleb);
256     value_out= opt_target(x_out,deg,family_type,Vleb);
257     if abs(value_in) < abs(value_out)
258         x_out=x_in;
259         value=value_in;
260     else
261         value=value_out;
262     end
263
end

```

```

265
267
269
271
%-----
273 % ADDITIONAL ROUTINES.
%-----
275
function [lebdeg,maxfun,maxtol,tolx]=option_settings(...
277     deg,mildness)

279 switch mildness
    case 0
        if deg < 7
            lebdeg=70;
        else
            if deg < 12
                lebdeg=50;
            else
                lebdeg=30;
            end
        end
        maxfun=50; maxtol=10^(-12); tolx=maxtol;
    case 1

        if deg < 7
            lebdeg=80;
        else
            if deg < 12
                lebdeg=60;
            else
                lebdeg=40;
            end
        end
        maxfun=50; maxtol=10^(-12); tolx=maxtol;
    case 2
        if deg < 7
            lebdeg=100;
        else

```

```

          if deg < 12
309           lebdeg=70;
            else
311             lebdeg=50;
            end
313         end
            maxfun=50; maxtol=10^(-12); tolx=maxtol;
315     case 3
            if deg < 7
317              lebdeg=135;
            else
319              if deg < 12
321                lebdeg=85;
            else
323              lebdeg=60;
            end
325         end
            maxfun=50; maxtol=10^(-12); tolx=maxtol;
327     case 4
            if deg < 7
329              lebdeg=170;
            else
331              if deg < 12
333                lebdeg=120;
            else
335              lebdeg=80;
            end
337         end
            maxfun=50; maxtol=10^(-14); tolx=maxtol;
339     case 5
341       if deg < 7
343         lebdeg=250;
            else
345             if deg < 12
347               lebdeg=150;
            else
349               lebdeg=100;
            end
        end
            maxfun=25; maxtol=10^(-14); tolx=10^(-60);

```

```

351      case 6
352          if deg < 7
353              lebdeg=270;
354          else
355              if deg < 12
356                  lebdeg=170;
357              else
358                  lebdeg=120;
359              end
360          end
361          maxfun=25; maxtol=10^(-14); tolx=maxtol;

362      case 7
363
364          if deg < 7
365              lebdeg=300;
366          else
367              if deg < 12
368                  lebdeg=200;
369              else
370                  lebdeg=140;
371              end
372          end
373          maxfun=25; maxtol=10^(-14); tolx=10^(-60);

374      case 8
375
376          if deg < 7
377              lebdeg=325;
378          else
379              if deg < 12
380                  lebdeg=250;
381              else
382                  lebdeg=170;
383              end
384          end
385          maxfun=25; maxtol=10^(-14); tolx=10^(-60);

386      otherwise
387
388          if deg < 7
389              lebdeg=350;
390          else

```

```

        if deg < 12
395            lebdeg=250;
        else
397            lebdeg=170;
        end
399    end
        maxfun=50; maxtol=10^(-400); tolx=10^(-60);
401
    end
403
405
407 function value=opt_target(theta,r,gruppi,numeronodi,deg,family_type,Vleb)
409
411 %-----
412 % REASSEMBLING OPTIMIZATION VECTOR INTO N x 2
413 % MATRIX.
414 %-----
415 pts=cerchio(theta,r,gruppi,numeronodi);
417 X=pts(:,1);
418 Y=pts(:,2);
419
420 %-----
421 % OPTIMIZATION TARGET FUNCTION..
422 %-----
423 s = (X.^2 + Y.^2 - 1 > eps); sum_s=sum(s);
425
if sum_s > 0
427     % fprintf('\n \t %% WARNING: ITERATION OUT OF DOMAIN.');
429
switch family_type
431
case -2
433     value=10^30; return;
435
case -1
436     value=10^30; return;

```

```

437
438     case 0
439         value=10^30; return;
440
441     case 1 % VANDERMONDE DETERMINANT: FEKETE POINTS.
442         value=10^(-30); return;
443
444     case 2 % APPROXIMATE LEBESGUE CONSTANT: APPROXIMATE LEBESGUE POINTS
445         value=10^30; return;
446
447     end
448 end
449
450 V = koornwinder_typeII(deg,pts,-0.5);
451
452 switch family_type
453
454     case -2
455         s=-family_type;
456         value=riesz_target(pts,s);
457
458     case -1
459         s=-family_type;
460         value=riesz_target(pts,s);
461
462     case 0
463         s=family_type;
464         value=riesz_target(pts,s);
465
466     case 1 % VANDERMONDE DETERMINANT: FEKETE POINTS.
467         detV=det(V);
468         value=-abs(detV);
469         %fprintf('\n \t %% %2.20e',value);
470
471     case 2 % APPROXIMATE LEBESGUE CONSTANT: APPROXIMATE LEBESGUE POINTS.
472         value=norm(V'\Vleb',1);
473
474 end

```

Riferimenti bibliografici

- [1] M. Briani, A. Sommariva, M. Vianello: *Computing Lebesgue and Fekete points in Matlab*, J. Comput. Appl. Math. 236 (2012), 2477-2486.
- [2] L Mezzalira: *Calcolo di punti quasi ottimali per l'interpolazione sull'intervallo, il quadrato e il disco*, Tesi di Laurea (2011), Dipartimento di Matematica, Universitá di Padova.
- [3] C. de Boor, *A Practical Guide to Splines*, Springer-Verlag, 1978.
- [4] Mathworks, ga routine, <http://www.mathworks.it/it/help/gads/ga.html>.
- [5] Mathworks, patternsearch routine, <http://www.mathworks.it/it/help/gads/patternsear>
- [6] L. Bos, S. De Marchi, A Sommariva e M.Vianello, *Computing multivariate Fekete and Leja points by numerical linear algebra*, SIAM J. Numer. Anal., 48 (2010), 1984-1999.
- [7] A. Cuyt, I. Yaman, B.A. Ibrahimoglu e B. Benouahmane, *Radial orthogonality and Lebesgue constants on the disk*, Numer. Algo., 61 (2012), 291-313.
- [8] Y. Xu, *Polynomial interpolation on the unit sphere and on the unit ball*, Adv. in Comp. Math., 20 (2004), 247-260.
- [9] Sundermann,B.: *On projection constants of polynomial space on the unit ball in several variables*, Math. Z. ,188 (1984), 111-117.