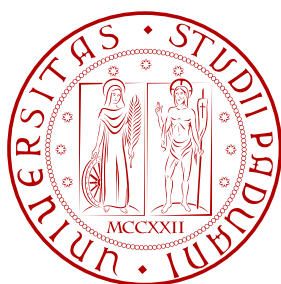


UNIVERSITÀ DEGLI STUDI DI PADOVA

---

Corso di Laurea Triennale in Matematica



# Costante di Lebesgue sul disco

Andrea Pinto

Relatori:

- Prof. Francesco Rinaldi
- Prof. Alvise Sommariva

Anno Accademico 2011/12

# Indice

<b>Introduzione</b>	<b>3</b>
<b>1 Interpolazione polinomiale e costante di Lebesgue</b>	<b>3</b>
1.1 La costante di Lebesgue . . . . .	4
1.1.1 Teorema fondamentale per l'errore di interpolazione . .	5
1.1.2 Punti di Lebesgue . . . . .	6
1.2 Richiami su alcuni nodi di interpolazione classici nell'intervallo	6
<b>2 Algoritmi</b>	<b>8</b>
2.1 Algoritmo Genetico . . . . .	8
2.1.1 Selezione . . . . .	8
2.1.2 Mutazione . . . . .	9
2.1.3 Crossover . . . . .	9
2.1.4 Altri parametri . . . . .	9
2.1.5 Alcune considerazioni . . . . .	10
2.2 Direct search (Pattern Search) . . . . .	11
2.2.1 Parametri . . . . .	11
2.3 Combinazione di GA con Patternsearch . . . . .	11
<b>3 Funzione obiettivo</b>	<b>12</b>
<b>4 Un primo approccio</b>	<b>13</b>
<b>5 Secondo approccio: punti vincolati su cerchi concentrici</b>	<b>14</b>
5.1 La suddivisione in k gruppi . . . . .	14
5.2 La scelta dell'angolo . . . . .	17
5.3 Confronto tra alcune configurazioni di raggi . . . . .	17
5.4 Ottimizzando sia su raggi sia su angoli . . . . .	19
5.4.1 Confronto con la configurazione di raggi particolari . .	21
5.4.2 Confronto con altri autori . . . . .	21
5.4.3 Confronto con la teoria . . . . .	23
<b>6 Possibili sviluppi</b>	<b>26</b>
<b>7 Conclusione?</b>	<b>28</b>
<b>A Alcuni dati</b>	<b>29</b>
<b>B Codici</b>	<b>34</b>



# Introduzione

L'interpolazione polinomiale è uno dei grandi temi della teoria dell'approssimazione: rispetto a quella a tratti offre diversi vantaggi e alcuni svantaggi. Da una parte, infatti, fornisce importanti informazioni sulle derivate delle funzioni che vogliamo interpolare, dall'altra invece, ha un costo computazionale più alto e una minor precisione, se i punti su cui si interpola non sono molto buoni. Esempio di questo secondo fatto è il fenomeno di Runge. La ricerca di punti adeguati per l'interpolazione polinomiale diventa, quindi, uno dei problemi più classici della teoria dell'approssimazione. 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. In questo lavoro ci siamo concentrati al caso del disco bidimensionale: è stato possibile riuscire a trovare dei miglioramenti ai già ottimi risultati presenti in letteratura a riguardo. Per prima cosa faremo dei richiami sull'interpolatore polinomiale e sulla sua norma, la costante di Lebesgue. In secondo luogo esporremo gli algoritmi usati, alcuni dei quali non propriamente tradizionali. A questo punto sposteremo la nostra attenzione sui vari tentativi che abbiamo fatto per trovare dei miglioramenti. In conclusione, illustreremo i dati che abbiamo raccolto nelle sperimentazioni numeriche confrontandole con quelle di altri autori.

## 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  $N = \dim(\mathbb{P}_n^d) = \binom{n+d}{d}$ . Sia  $\Omega$  un dominio compatto di  $\mathbb{R}^d$ . Un insieme di punti  $\{z_n\}$  si dice unisolvante su  $\Omega$  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'unisolvienza: la cardinalità  $N$  di tale insieme  $\{z_n\}$  deve essere uguale alla dimensione di  $\mathbb{P}_n$ .

Fissiamo nel dominio  $\Omega$  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

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

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

$$L_p f(\cdot) = \sum_{i=1}^M f(z_i) L_{z_i}(\cdot)$$

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  $(z_k, \gamma_k)$ .

Più in generale, per  $d$  generico, dalla definizione 1, si vede che se l'insieme di punti è unisolviente, si ha  $L_k(x_s) = \delta_{k,s}$  per ogni  $k, s$  e quindi il polinomio dei polinomi di Lagrange abbiamo che il polinomio interpolatore è

$$L_{p_n}(z) = \sum_{k=0}^n \gamma_k L_k(z)$$

## 1.1 La costante di Lebesgue

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

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

La grandezza di  $\Lambda$  è in funzione dei punti su cui si interpola, del dominio  $\Omega$  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.

### 1.1.1 Teorema fondamentale per l'errore di interpolazione

**Teorema.**

Sia  $X$  uno spazio di funzioni su  $\Omega \subseteq \mathbb{R}^d$  ove  $\Omega$  è un insieme compatto, sia  $L_p: (C(\Omega), \|\cdot\|_\infty) \rightarrow \mathbb{P}_n^d$  interpolatore su  $\{x_i\} \subset \Omega$  unisolvante e sia  $\Lambda_n$  la corrispondente costante di Lebesgue, allora

1.  $\|Lf - f\|_\infty \leq (1 + \Lambda_n) \|f - \phi_n^*\|_\infty$  ove  $\phi_n^* \in \mathbb{P}_n^d$  è il polinomio di miglior approssimazione per  $f$ .
2.  $\|Lf - L\bar{f}\|_\infty \leq \Lambda_n \max_k |f(x_k) - \bar{f}(x_k)|$  ove  $\bar{f}$  è la funzioni perturbata di  $f$ .

*Dimostrazione.*

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

$$f - Lf = f - \phi_n^* + \phi_n^* - Lf = f - \phi_n^* + \phi_n^* - L\phi_n^* + L\phi_n^* - Lf = f - \phi_n^* + L(\phi_n^* - f)$$

Ma allora

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

2.  $|Lf - L\bar{f}| = |\sum_i f(x_i)L_i(x) - \sum_i \bar{f}(x_i)L_i(x)| \leq |\sum_i (f(x_i) - \bar{f}(x_i))L_i(x)| \leq$   
 $\leq \sum_i |f(x_i) - \bar{f}(x_i)| |L_i(x)| \leq \max_i |f(x_i) - \bar{f}(x_i)| \sum_i |L_i(x)|$

Ma allora

$$\|Lf - L\bar{f}\|_\infty \leq \|L\|_\infty \max_i |f(x_i) - \bar{f}(x_i)|.$$

□

### 1.1.2 Punti di Lebesgue

Fissati  $\Omega$  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"

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

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. Dalla **definizione 2** deriva tale stima.

Questa relazione **ci fornisce una sovrastima per i punti di Lebesgue.**

La costante di Lebesgue  $\|L\|$  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 conoscendoli per un triangolo particolare, o un quadrato o un cerchio, li si conosce per qualsiasi triangolo, qualsiasi rettangolo o qualsiasi ellisse.

## 1.2 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.

- **Extended Chebyshev.** Questi punti nell'intervallo  $[1,-1]$  sono gli zeri dei polinomi di Chebyshev, ovvero gli zeri di  $\mathcal{T}_n$  con  $\mathcal{T}_n(x) = \cos(n \arccos(x))$ . Esiste una formulazione ricorsiva di questi polinomi

$$\mathcal{T}_0(x) = 1$$

$$T_1(x) = x$$

$$T_{n+1}(x) = 2xT_n(x) - T_{n-1}(x)$$

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 spiegato in [3] uno sviluppo dei nodi Extended Chebyshev sono gli Expanded Chebyshev: infatti questi ultimi sono una dilatazione dei primi così che il primo e l'ultimo nodo siano sugli estremi dell'intervallo. Sono dati dalla seguente formulazione:

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

Gli Expanded Chebyshev risultano avere una costante di Lebesgue leggermente più piccola rispetto a quella degli Extended. Tali punti sono considerati in letteratura quasi punti di Lebesgue 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  $U_n$  ove  $U_n$  è dato dalla formula ricorsiva

$$U_0(x) = 1$$

$$U_1(x) = 2x$$

$$U_{n+1}(x) = 2xU_n(x) - U_{n-1}(x)$$

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  $i=1 \dots n$ .

- **Legendre-Lobatto** Questi punti nell'intervallo  $[-1, 1]$  sono gli zeri dei polinomi di Legendre, ovvero gli zeri di  $P_n$  con  $P_n$  data dalla formulazione ricorsiva

$$P_0 = 1$$

$$P_1 = x$$

$$P_n = \frac{2n-1}{n} x P_{n-1} - \frac{n-1}{n} P_{n-2}$$

Tali punti sono molto utilizzati per le formule di quadratura.



## 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:  $\text{figlio} = \text{genitore1} + \text{rand} * \text{Razio} * (\text{genitore2} - \text{genitore1})$  ove  $\text{rand}$  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 ritenuta 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 patternsearch 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 Pattern-

search. 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  $P$  unisolvente su  $\Omega$ , abbiamo ereditato le tecniche presenti in [1]. Data una mesh di riferimento sufficientemente fitta  $X$ , si valuta la costante di Lebesgue  $\lambda_n$  su questi punti (invece che su tutto il dominio) e si suppone

$$\Lambda(P) = \|L_P\| = \max_{x \in \Omega} \lambda_n(x; P) \approx \max_{x \in X} \lambda_n(x, P)$$

dove la funzione di Lebesgue è

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

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

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

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 **mesh** al massimo di  $250^2 = 62500$  punti, come già in [1, 2].

In questo modo abbiamo che

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

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

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 intorno 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 intorno 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: partendo 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  $\mathbb{R}^2$  abbiamo già detto di aver bisogno di  $N = \binom{n+2}{n}$  punti: vincoliamo questi in  $\lfloor \frac{n}{2} \rfloor + 1$  cerchi. La scelta di tale valore deriva dalle considerazioni fatte da [7, 8]. Abbiamo verificato sperimentalmente almeno parzialmente questa scelta. 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  $2v_j + 1$  punti equidistanti su cui interpoleremo. La scelta di  $n_j$  è data da:

$$n_1 = n - v_1 + 1,$$

$$n - 2 = n - 2v_1 - v_2 + 1,$$

:

$$n_k = n - 2v_1 - \dots - 2v_{k-1} - v_k + 1.$$

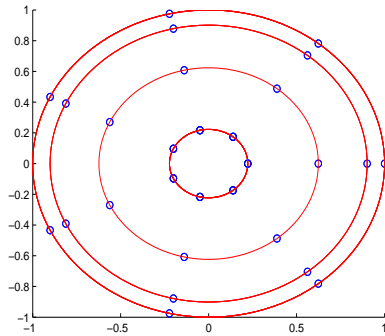
Facilmente si può verificare che

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

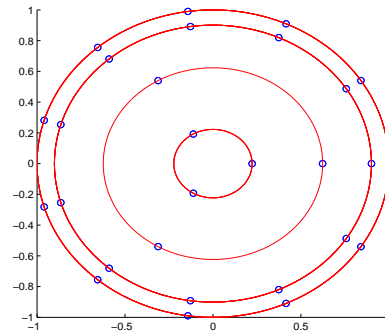
A questo punto rimangono da fare principalmente tre scelte: la lunghezza dei raggi, l'angolazione tra un fissato punto di interpolazione di ogni cerchio con un determinato semiasse e la suddivisione nei  $k$  gruppi.

### 5.1 La suddivisione in $k$ gruppi

Come si vede in [7], 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: maggiori dettagli su questa scelta verranno forniti nel paragrafo 5.3. Fissiamo gli angoli tutti uguali a zero: in tal modo in ogni cerchio ci sarà un punto in un certo semiasse.



(a)  $k=1$

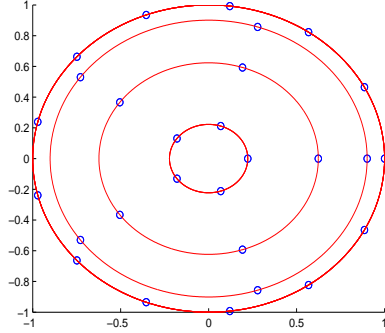


(b)  $k=2$ , primo caso

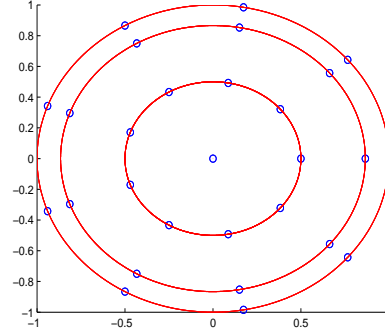
Figura 1:

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 1(a).  
In tal caso risulta una costante di Lebesgue di 4025,6;
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 1(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 2(a).  
In tal caso risulta una costante di Lebesgue di 98.1135;
  - 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 2(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 3(a).  
In tal caso abbiamo una costante di Lebesgue di circa 13.52;



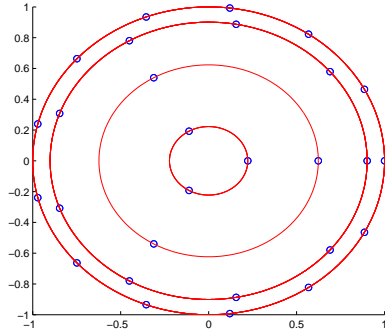


(a)  $k=2$ , secondo caso

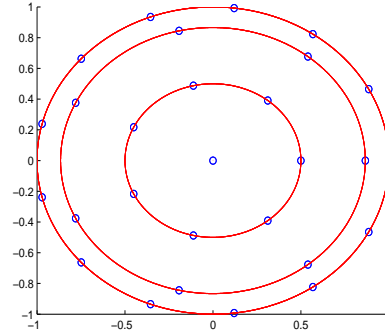


(b)  $k=2$ , terzo caso

Figura 2:



(a)  $k=3$ , primo caso



(b)  $k=3$ , secondo caso

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 3(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 4(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 4(b). In tal caso abbiamo una costante di Lebesgue di 5.48.

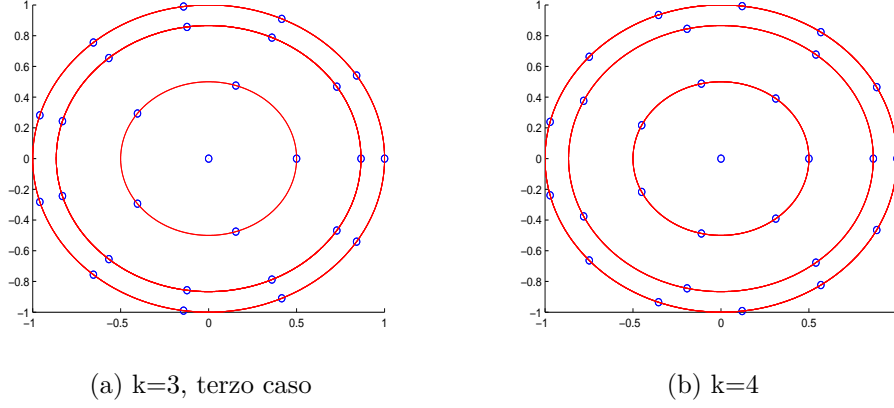


Figura 4:

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.

## 5.2 La scelta dell'angolo

Vediamo ora quanto la scelta degli angoli influisce sulla costante di Lebesgue dei punti di interpolazione strutturati come sopra. Studiamo il caso di  $n=6$ , con raggi con una lunghezza uguale alla distanza dall'origine ai nodi di Chebyshev-Lobatto positivi di grado 8 o 7 a seconda dei casi (questa scelta verrà spiegata in dettaglio nella sezione 5.3) e con la divisione ottimale in gruppi trovata nel paragrafo precedente. In una configurazione di angoli, sicuramente non ottimale, tale che ogni cerchio abbia un punto in un determinato semiasse, abbiamo una costante di Lebesgue di circa 5.48. Ottimizzando invece la nostra funzione obiettivo tramite la funzione GA seguita da patensearch, con solo i 3 angoli come variabili, otteniamo una costante di Lebesgue di circa 5.35. Concludiamo che la scelta del raggio influisce limitatamente sulla costante di Lebesgue nel caso di configurazioni con raggi buoni.

## 5.3 Confronto tra alcune configurazioni di raggi

Dando per accettata la configurazione ottimale trovata nella sezione 5.1, facciamo un confronto tra raggi particolari. Utilizziamo infatti, raggi di lunghezza uguale a quella del segmento tra l'origine e i nodi positivi di famiglie classiche (come quelle viste nel paragrafo 1.2).

Limitiamoci, momentaneamente, al caso  $n=6$ . In questa situazione, la configurazione di punti giace su 4 cerchi centrati nell'origine e quindi dobbiamo determinare i relativi 4 raggi. Per determinare i 4 raggi c'è necessità di 4 nodi nell'intervallo  $[0,1]$ : 4 nodi significa, poiché utilizziamo solo i nodi positivi, utilizzare i nodi di grado 8 o 7 delle famiglie viste nella sezione 1.2. Infatti, tali famiglie per l'intervallo  $[-1,1]$ , col grado 8 e 7, hanno esattamente 4 nodi nell'intervallo  $[0,1]$ . Il grado 7 sperimentalmente si rivela migliore. Si può vedere facilmente che, in tale occasione, il primo raggio sarà uguale a 0. Tale fatto, in questa situazione, non ci genera disagi poiché nel cerchio interno c'è un solo punto. Nel caso invece di  $n=7$ , poiché la configurazione dei punti giace su 4 cerchi centrati nell'origine dobbiamo determinare ancora 4 raggi. Come sopra utilizziamo i nodi di grado 8 o 7 delle famiglie classiche. In questo caso, però, nel cerchio interno ci sono 3 punti di interpolazione: la scelta di utilizzare il grado 7, che fornirebbe il numero giusto di raggi, si rivela poco saggia poiché con questa scelta ci sarebbe un raggio nullo, e i tre punti del cerchio più interno dovrebbero coincidere nell'origine, creando così una singolarità nella costante di Lebesgue. La scelta di utilizzare il grado 8 si rivela più consona.

Se  $n$  è pari, con la configurazione trovata nei paragrafi precedenti, ci sarà un unico punto nel cerchio più interno: in questa situazione risulterà essere consigliabile utilizzare un set di nodi relativi con grado pari a  $n+1$ . Se invece  $n$  è dispari, è necessario evitare di avere un raggio nullo: il grado dei nodi risulterà essere in ogni caso  $n+1$ . Considerazioni simili valgono per le famiglie di nodi di Chebyshev e di Legendre-Lobatto.

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 Expanded 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.

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 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  $n$ .

Grado	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

Grado	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 1: Confronto tra le costanti di Lebesgue di set di punti aventi configurazioni con raggi dati da famiglie di nodi classiche

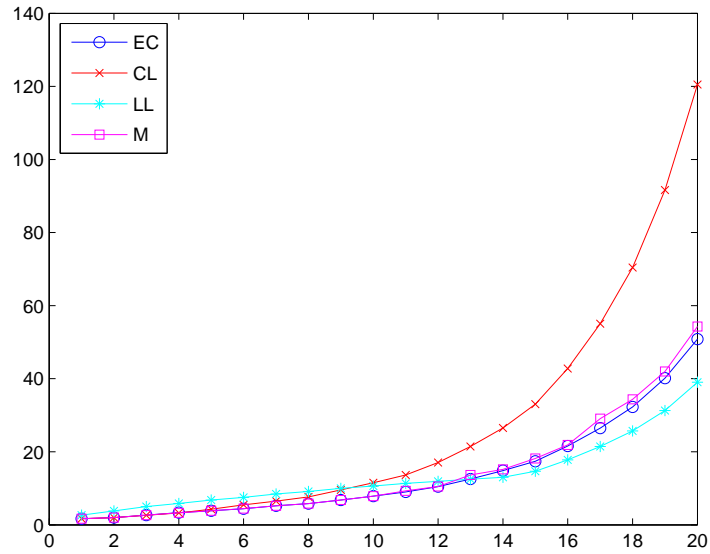


Figura 5: Confronto tra le costanti di Lebesgue di set di punti aventi configurazioni con raggi dati da famiglie di nodi classiche

## 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 2: 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 piu' 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.

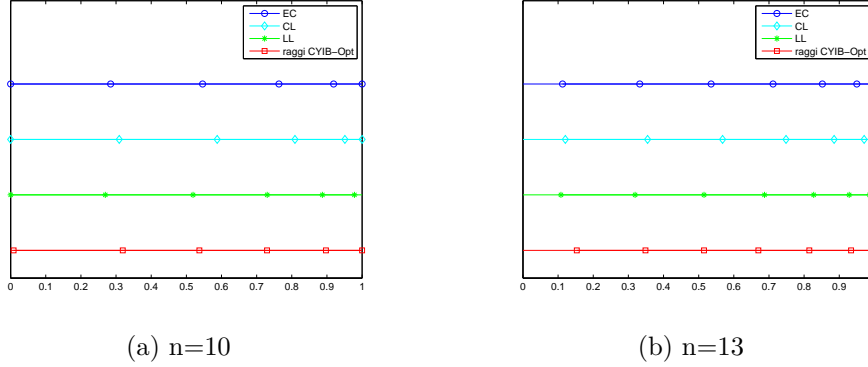


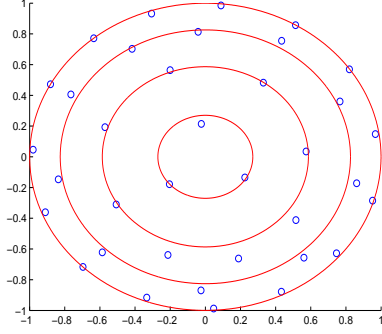
Figura 6: confronto tra raggi

Nei grafici illustriamo solo i casi  $n=10$  e  $n=13$ , pur avendo verificato che cio' accade anche piu' 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.

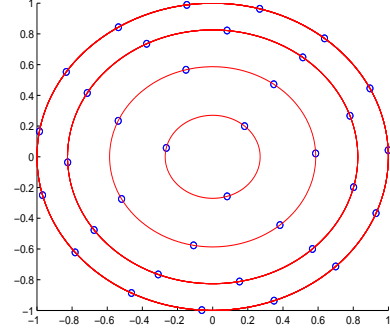
### 5.4.2 Confronto con altri autori

Vogliamo ora invece fare un confronto tra le nostre configurazioni e quelle trovate da [1]: i nostri punti sono vincolati in cerchi concentrici al contrario di quelli di [1].

Abbiamo riportato in grafico i casi di  $n=7$ ,  $n=13$  e  $n=20$ : per ogni caso in un grafico ci sono i punti trovati da [1] con le circonferenze aventi raggio trovato ottimizzando come abbiamo descritto sopra, nell'altro i punti da noi trovati in questa sezione. Possiamo vedere come i punti, pur non essendo nelle circonferenze, non risultano essere così lontani da esse. Interessante notare come i punti di [1] non lontani da una certa circonferenza risultino essere circa equispaziati in essa ed essere nello stesso numero rispetto ai punti trovati da noi. Infatti già nel lavoro di [1] ci si rende conto che i punti sono disposti in  $k$  poligoni regolari di grado  $4k+1$  per i gradi pari e  $4k-1$  per i gradi dispari.

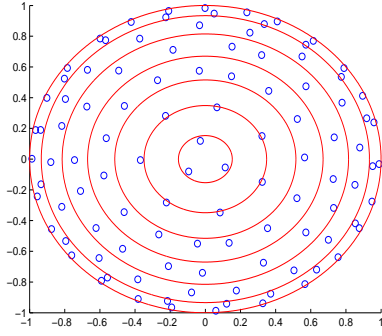


(a) BSV

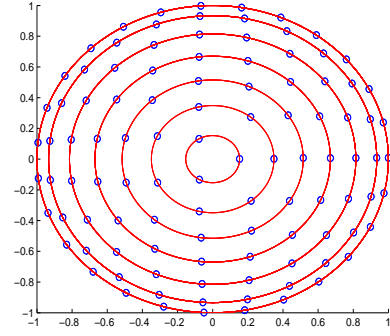


(b) CYIB-Opt

Figura 7:  $n=7$

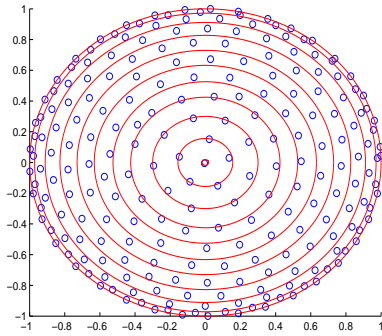


(a) BSV

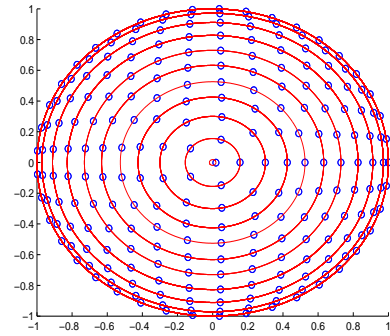


(b) CYIB-Opt

Figura 8:  $n=13$



(a) BSV



(b) CYIB-Opt

Figura 9:  $n=20$

A questo punto esponiamo un grafico raffigurante le costanti di Lebesgue per i punti presentati da [1], [2] e da noi in questa sezione.

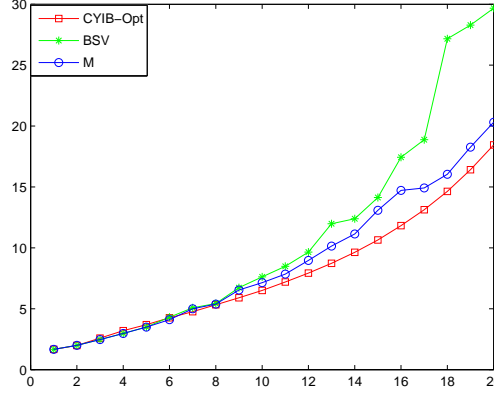


Figura 10: andamento delle costanti di Lebesgue di CYIB-Opt, di BSV e di M

Riportiamo ora per completezza anche delle tabelle (rispettivamente 3 e 4) con i valori del determinante della matrice di Vendermonde e del condizionamento di questa sia per i punti trovati in questa sezione sia per quelli trovati in [1] e [2].

### 5.4.3 Confronto con la teoria

In [7] si dá una 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 da noi con questa puramente teorica.

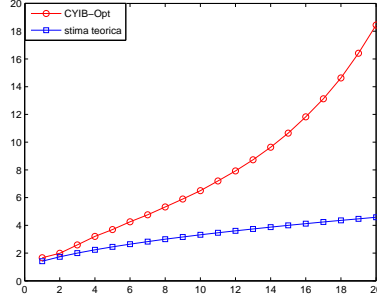
Nel grafico 11(a) abbiamo posto rappresentato l'andamento di  $\Lambda_n = \sqrt{n+1}$  e le costanti di Lebesgue trovate nella sezione 5.4 (indicate con CIYT): questo solo per mostrare come l'andamento delle nostre costanti sia palesemente diverso.

Come già esplicitato, le costanti di Lebesgue teoriche sono stimate con  $\Lambda_n = C\sqrt{n+1}$  ove  $C$  é una costante. A livello sperimentale é facile convincersi di come le nostre costanti non hanno questo andamento: se lo avessero, e non lo hanno, avremmo che

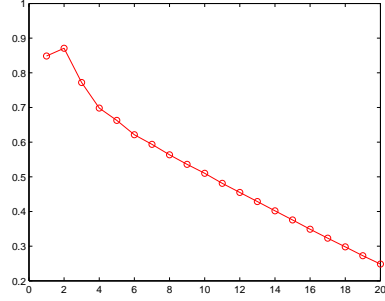
$$\frac{\sqrt{n+1}}{\tilde{\Lambda}_n} = C$$

ove con  $\tilde{\Lambda}_n$  abbiamo indicato le costanti di Lebesgue trovate nella sezione 5.4. Proponiamo la tabella 5.4.3 e il grafico 11(b) circa tali considerazioni.





(a) andamento delle costanti di Lebesgue



(b) rapporto tra  $\sqrt{n+1}$  e le costanti di Lebesgue trovate da CYIB-Opt

Figura 11:

determinante di Vandermonde					
Grado	1	2	3	4	5
BSV	$4.95 \cdot 10^{-1}$	$2.12 \cdot 10^0$	$2.97 \cdot 10^1$	$2.07 \cdot 10^3$	$7.03 \cdot 10^5$
M	$4.95 \cdot 10^{-1}$	$2.12 \cdot 10^0$	$2.97 \cdot 10^1$	$2.11 \cdot 10^3$	$7.11 \cdot 10^5$
CYIB-Opt	$4.95 \cdot 10^{-1}$	$3.69 \cdot 10^{-1}$	$2.91 \cdot 10^1$	$1.21 \cdot 10^3$	$1.32 \cdot 10^6$
Grado	6	7	8	9	10
BSV	$1.10 \cdot 10^{10}$	$4.15 \cdot 10^{13}$	$2.87 \cdot 10^{19}$	$1.74 \cdot 10^{25}$	$2.28 \cdot 10^{34}$
M	$1.03 \cdot 10^{10}$	$4.14 \cdot 10^{13}$	$2.96 \cdot 10^{19}$	$2.29 \cdot 10^{25}$	$1.52 \cdot 10^{34}$
CYIB-Opt	$8.09 \cdot 10^9$	$2.27 \cdot 10^{14}$	$7.18 \cdot 10^{18}$	$1.02 \cdot 10^{26}$	$1.43 \cdot 10^{34}$
Grado	11	12	13	14	15
BSV	$2.96 \cdot 10^{38}$	$1.06 \cdot 10^{48}$	$4.08 \cdot 10^{58}$	$2.34 \cdot 10^{76}$	$2.38 \cdot 10^{89}$
M	$2.52 \cdot 10^{43}$	$8.91 \cdot 10^{52}$	$2.30 \cdot 10^{60}$	$7.24 \cdot 10^{75}$	$1.65 \cdot 10^{89}$
CYIB-Opt	$5.76 \cdot 10^{42}$	$3.82 \cdot 10^{52}$	$2.99 \cdot 10^{63}$	$6.70 \cdot 10^{73}$	$1.90 \cdot 10^{85}$
Grado	16	17	18	19	20
BSV	$3.51 \cdot 10^{97}$	-	-	-	-
M	$4.36 \cdot 10^{99}$	$8.42 \cdot 10^{119}$	$1.76 \cdot 10^{137}$	$3.93 \cdot 10^{155}$	$7.96 \cdot 10^{176}$
CYIB-Opt	$2.33 \cdot 10^{103}$	$3.71 \cdot 10^{116}$	$7.90 \cdot 10^{135}$	$2.58 \cdot 10^{153}$	$6.75 \cdot 10^{173}$

Tabella 3: Confronto tra i determinanti della Matrice di Vandermonde per i set di punti trovati da noi e alcuni presenti in letteratura

Condizionamenti					
Grado	1	2	3	4	5
BSV	$2.98 \cdot 10^0$	$5.33 \cdot 10^0$	$1.08 \cdot 10^1$	$1.32 \cdot 10^1$	$1.55 \cdot 10^1$
M	$2.98 \cdot 10^0$	$5.13 \cdot 10^0$	$1.08 \cdot 10^1$	$1.49 \cdot 10^1$	$2.12 \cdot 10^1$
CYIB-Opt	$2.98 \cdot 10^0$	$5.43 \cdot 10^0$	$1.05 \cdot 10^1$	$1.43 \cdot 10^1$	$1.93 \cdot 10^1$
Grado	6	7	8	9	10
BSV	$2.84 \cdot 10^1$	$2.55 \cdot 10^1$	$3.58 \cdot 10^1$	$5.34 \cdot 10^1$	$6.37 \cdot 10^1$
M	$2.84 \cdot 10^1$	$2.83 \cdot 10^1$	$3.53 \cdot 10^1$	$5.52 \cdot 10^1$	$6.95 \cdot 10^1$
CYIB-Opt	$2.77 \cdot 10^1$	$3.35 \cdot 10^1$	$4.33 \cdot 10^1$	$4.48 \cdot 10^1$	$6.33 \cdot 10^1$
Grado	11	12	13	14	15
BSV	$7.85 \cdot 10^1$	$1.06 \cdot 10^1$	$9.55 \cdot 10^1$	$1.18 \cdot 10^2$	$1.52 \cdot 10^2$
M	$7.82 \cdot 10^1$	$1.07 \cdot 10^1$	$1.24 \cdot 10^2$	$1.49 \cdot 10^2$	$1.59 \cdot 10^2$
CYIB-Opt	$7.12 \cdot 10^1$	$8.06 \cdot 10^1$	$1.16 \cdot 10^2$	$3.66 \cdot 10^3$	$1.73 \cdot 10^4$
Grado	16	17	18	19	20
BSV	$1.43 \cdot 10^2$	-	-	-	-
M	$2.12 \cdot 10^2$	$2.12 \cdot 10^2$	$2.66 \cdot 10^2$	$2.96 \cdot 10^2$	$3.44 \cdot 10^2$
CYIB-Opt	$1.65 \cdot 10^2$	$1.97 \cdot 10^2$	$2.39 \cdot 10^2$	$1.34 \cdot 10^3$	$4.05 \cdot 10^2$

Tabella 4: Confronto tra i condizionamenti della Matrice di Vandermonde per i set di punti trovati da noi e alcuni presenti in letteratura

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
$\frac{\sqrt{n+1}}{\tilde{\Lambda}_n} = C$	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
$\frac{\sqrt{n+1}}{\tilde{\Lambda}_n} = C$	0.48	0.45	0.43	0.40	0.38	0.35	0.32	0.30	0.27	0.25

Tabella 5: confronto tra le costanti di Lebesgue di CYIB-Opt con  $\sqrt{n+1}$

## 6 Possibili sviluppi

Una possibile via di sviluppo di questo lavoro é di uscire dai vincoli che abbiamo imposto: pensiamo infatti che la configurazione presentata da noi é ben lontana dalla migliore possibile. Abbiamo in parte esplorato questo aspetto riprendendo un'idea presente in [7]. 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  $n$  nodi in un cerchio di raggio  $r - \varepsilon$  e gli altri  $n+1$  nodi in uno di raggio  $r + \varepsilon$  senza modificarne l'angolo: un esempio nella figura 12. Ovviamente se  $\varepsilon = 0$  ab-

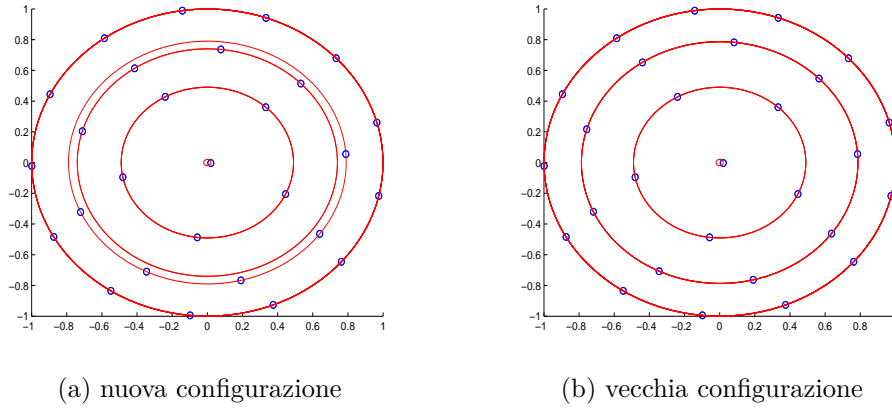


Figura 12: Esempio di come dividere i punti di un un cerchio in due cerchi

biamo la vecchia configurazione.

Agendo in questo modo in tutti i cerchi con piú di un nodo, disponiamo tutti i nodi in  $n+1$  cerchi. A questo punto ottimizziamo tramite GA e 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 6 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 6: 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)

Una differenza é presente anche se spesso troppo piccola perché tali risultati possano essere ritenuti un miglioramento rispetto ai precedenti. In tal senso pensiamo che applicando questa idea più volte fino ad ottenere un cerchio per ogni nodo potrebbe fornire un vantaggio considerevole. Oltre a questa indagine piu' accurata un'ulteriore sviluppo potrebbe essere una diversa scelta dei nodi da porre nei nuovi cerchi.

## 7 Conclusione?

## 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
0.9999999850988388	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

## Riferimenti bibliografici

- [1] Matteo Briani, Alvis Sommariva, Marco Vianello: *Computing Fekete and Lebesgue points: simplex, square, disk*(2010), Department of Mathematics, University of Padova, Italy.  
<http://www.math.unipd.it/marcov/pdf/leb.pdf>
- [2] Luca Mezzalana: *Calcolo di punti quasi ottimali per l'interpolazione sull'intervallo, il quadrato e il disco*(2011), Department of Mathematics, University of Padova, Italy.
- [3] Carl de Boor: *A Practical Guide to Splines*, Springer
- [4] <http://www.mathworks.it/it/help/gads/ga.html>
- [5] <http://www.mathworks.it/it/help/gads/patternsearch.html>
- [6] L. Bos, S. De Marchi, A. Sommariva and M. Vianello: *Computing multivariate Fekete and Leja points by numerical linear algebra*(2010)  
<http://www.math.unipd.it/marcov/pdf/leja.pdf>
- [7] Annie Cuyt, Irem Yaman, Bayram Ali Ibrahimoglu, B. Benouahmane: *Radial orthogonality and Lebesgue constants on the disk*, Numerical Algorithms(2012), volume 61, pagine 291-313.
- [8] Xu, Y.: *Polynomial interpolation on the unit sphere and on the unit ball*, Advances in Computational Mathematics (2004), volume 20, pagine 247-260. Department of Mathematics, University of Oregon, Eugene, USA