Sperimentazione numerica di un metodo per l'approssimazione simultanea degli zeri di un polinomio (Calcolo Scientifico - proff. L. Gemignani e D. Bini)

Martino De Leo - 439650 - deleo@mail.dm.unipi.it

1 Introduzione

Ci si è basati su [5], in cui si descrive un metodo per l'approssimazione simultanea degli zeri di un polinomio e una condizione sufficiente (3.8) per la convergenza (localmente di ordine cubico), per impostare un esperimento numerico su calcolatore che verificasse la convergenza locale ed eventuali casi di convergenza globale del metodo iterativo.

L'esistenza di una condizione facilmente computabile come quella fornita in [5] fornisce un criterio sufficiente per sapere *a priori* che il metodo convergerà partendo con le approssimazioni iniziali $z_1^{(0)}, \ldots, z_n^{(0)}$, *senza* conoscere le radici esatte che si va approssimando.

1.1 Il metodo di Newton con la correzione di Weierstrass

Sia $p(z) = a_0 + a_1 z + a_2 z^2 + \cdots + a_{n-1} z^{n-1} + z^n \in \mathbb{C}[z]$ un polinomio a coefficienti complessi, che senza perdita di generalità[1] si suppone monico. Siano $\zeta_1, \ldots, \zeta_n$ le sue radici, supposte *semplici* ovvero tutte distinte: $\zeta_j \neq \zeta_i \quad (\forall j \neq i)$, cosicché si può anche scrivere

$$p(z) = \sum_{i=1}^{n} a_i z^i = \prod_{i=1}^{n} (z - \zeta_i)\,.$$

Seguendo la stessa notazione di [5], si indicano (per ogni $i = 1, \ldots, n$) con $z_i^{(m)}$ l'approssimazione di ζ_i calcolata alla m-esima iterazione del metodo (dunque $z_i^{(0)}$ sono le approssimazioni iniziali fornite in input); quando non serve indicare esplicitamente il passo, si indicherà più semplicemente con $\hat{z}_i$ l'approssimazione successiva ottenuta dagli z_i con una sola iterazione.

L'algoritmo in esame è pressoché identico al ben noto metodo di Newton, a parte per un fattore di correzione W_i che chiameremo *correzione di Weierstrass*:

$$\hat{z}_i = z_i - \frac{p(z_i)}{p'\left(z_i - \frac{1}{2} W_i\right)}, \qquad i = 1, \ldots, n \tag{1}$$

La correzione di Weierstrass è calcolata valutando sull'approssimazione corrente la funzione razionale $W_i(z)$, nel seguente modo:

$$W_i(z) = \frac{p(z)}{\prod_{j=1, j\neq i}^{n} (z - z_j)}, \qquad W_i := W_i(z_i).$$

In [3] si dimostra (Teorema 1) che se le approssimazioni iniziali $z_1^{(0)}, \ldots, z_n^{(0)}$ sono "abbastanza vicine" alle (corrispondenti) radici esatte allora il metodo è convergente, e l'ordine di convergenza è 3.

Il risultato principale esposto in [5] riguarda un criterio sugli $z_i^{(0)}$ sufficiente per la convergenza.

1.2 Analisi della convergenza

Un ruolo cruciale è rappresentato dalle quantità

$$w := \max_{i=1,\ldots,n} |W_i|, \qquad d := \min_{j\neq i} |z_i - z_j|.$$

Si noti che, come sopra, si preferisce non scrivere $w^{(m)}$ e $d^{(m)}$ per alleggerire la notazione, al prezzo di perdere il riferimento esplicito al passo dell'algoritmo; sempre seguendo la stessa analogia si intenderà con $\hat{w}, \hat{W}_i, \hat{d}$... i valori corrispondenti calcolati dopo aver eseguito una iterazione del metodo.

[1] per il nostro scopo di approssimarne gli zeri.

Si consideri la condizione

$$w < dc \tag{3.8}$$

dove $c := c_n$ è un numero reale dipendente dal grado del polinomio. Dalla definizione dei parametri si vede che questa condizione è verificabile in maniera effettiva (i.e. computazionalmente), conoscendo *soltanto* i dati $z_1, \ldots, z_n$ e p[2].

Nello specifico, in [5] si dimostra che se la condizione iniziale $w^{(0)} < d^{(0)}c$, con $c = \frac{1}{6n}$ dove $n = \deg p$ è verificata, allora il metodo è convergente. La dimostrazione poggia su due altri risultati, il primo dei quali riguarda la convergenza di un'intera classe di metodi cui quello in esame (1) appartiene:

Teorema 3.1. *Consideriamo la classe dei metodi iterativi che aggiornano ogni approssimazione con una* correzione C_i *che dipende dalle approssimazioni correnti:*

$$z_i^{(m+1)} = z_i^{(m)} - C_i\left(z_1^{(m)}, \ldots, z_n^{(m)}\right) \qquad (i = 1, \ldots, n \quad m \geq 0),$$

dove le correzioni sono della forma

$$C_i(z_1, \ldots, z_n) = \frac{p(z_i)}{f_i(z_1, \ldots, z_n)},$$

e le funzioni f_i verificano le seguenti condizioni per ogni $i = 1, \ldots, n$:

1. $f_i(\zeta_1, \ldots, \zeta_n) \neq 0$;
2. $f_i(z_1, \ldots, z_n) \neq 0 \quad (\forall z_i \in \Lambda(\zeta_i))$, *dove ogni $\Lambda(\zeta_i)$ è un opportuno intorno della radice ζ_i;*
3. $f_i \in C^0(\mathbb{C}^n, \mathbb{C})$.

Si consideri inoltre la funzione definita per $t \in]0, 1[$ da

$$g(t) = \begin{cases} 1 + 2t, & 0 < t \leq \frac{1}{2} \\ \frac{1}{1-t}, & \frac{1}{2} < t < 1 \end{cases}.$$

Siano $z_1^{(0)}, \ldots, z_n^{(0)}$ approssimazioni iniziali distinte *delle radici di p. Se esiste un numero reale $0 < \beta < 1$ tale che*

a) $|C_i^{(m+1)}| \leq \beta |C_i^{(m)}|$ per ogni $i = 1, \ldots, n$ e per ogni $m \in \mathbb{N}$, e

b) $|z_i^{(0)} - z_j^{(0)}| > g(\beta)\left(|C_i^{(0)}| + |C_j^{(0)}|\right)$ per ogni $i, j = 1, \ldots, n$, $i \neq j$,

allora il metodo iterativo è convergente.

Per ulteriori dettagli vedasi il Teorema 1.2 in [1, p. 288] e, in una versione leggermente diversa, il Teorema 1 in [2, p. 315].

Si osservi che la condizione (a) dell'ipotesi implica che le correzioni siano strettamente decrescenti in modulo con l'avanzare delle iterazioni, mentre la (b) dice che il metodo non "confonde" le approssimazione dei diversi zeri di p nel senso che z_i, iterando, approssima lo zero ζ_i e non si "scambia di ruolo" con z_j.

In realtà, per trarre questa seconda conclusione bisognerebbe che la (b) valesse non soltanto per le approssimazioni di partenza ma per ogni iterazione $m \in \mathbb{N}$: vedremo che ciò è vero, considerando le $z_i^{(m)}$ come approssimazioni iniziali dell'istanza che calcola al primo passo le $z_i^{(m+1)}$. Si veda in proposito anche il Teorema 3.2.

L'altro risultato chiave afferma che una condizione della forma (3.8) assicura che certi *dischi di inclusione* contengono ognuno una radice esatta e l'approssimazione iniziale, e sono disgiunti. Insieme al risultato precedente ciò assicurerà la convergenza del metodo.

Si indichi con $D(z, r)$ il disco complesso di centro z e raggio r:

$$D(z, r) := \{y \in \mathbb{C} \text{ tali che } |y - z| \leq r\}.$$

Teorema 3.2. *Siano $z_1, \ldots, z_n \in \mathbb{C}$ tutti distinti tali che $w < dc$ per un certo $c < \frac{1}{2n}$.*
Allora i dischi

$$D_i := D\left(z_i, \frac{|W_i|}{1 - nc}\right), \quad i = 1, \ldots, n$$

sono mutuamente disgiunti e ognuno contiene una e una sola radice esatta di p:

$$i \neq j \Rightarrow D_i \cap D_j = \emptyset, \qquad \zeta_i \in D_i.$$

Questo risultato è riportato in [4, p. 4] (Teorema 1.1); si osservi però che la condizione (3.8) che stiamo considerando è più stretta dell'ipotesi del Teorema, perché richiede che $c < 1/6n$ anziché soltanto $c < 1/2n$.

[2] fornito ad esempio sotto forma di lista dei suoi coefficienti, più il suo grado.

Proseguendo, in [5] si dimostra poi (Lemma 3.1) che sotto la condizione (3.8) le correzioni di Weierstrass sono più che dimezzate in modulo ad ogni iterazione: $|\hat{W}_i| < 0.4|\hat{W}_i|$ e che la condizione (3.8) viene preservata: $\hat{w} < \hat{d}c$. Questa seconda affermazione in particolare permette di applicare induttivamente l'ipotesi al risultato di convergenza vero e proprio (Teorema 3.5), in cui si mostra come da $w < d/6n$ segue che entrambe le ipotesi del Teorema 1 sono verificate.

Nella conclusione dell'articolo si osserva che, nella pratica, la condizione (3.8) che garantisce la convergenza del metodo può verosimilmente essere rilassata, prendendo per c valori più grandi di $1/6n$.

2 Sperimentazione numerica

Ci si è basati su [5] per verificare l'ordine cubico di convergenza locale del metodo, la validità della condizione (3.8), ed eventuali casi di convergenza globale.

2.1 Convergenza locale

La suite di test si compone in questo caso di un generatore pseudocasuale di polinomi, della funzione che verifica la condizione (3.8) e della implementazione del metodo (1); infine una procedura per generare, a partire da p e dalle sue radici esatte ζ_i, delle approssimazioni iniziali $z_i^{(0)}$ che verifichino la (3.8). Quest'ultima procede generando per ogni radice esatta una direzione di perturbazione (un numero complesso di modulo unitario u_i) ed esegue una ricerca dicotomica sul modulo r_i della perturbazione[3] trovando il più alto r_i tale che la condizione (3.8) sia verificata per le approssimazioni iniziali $z_i^{(0)} = \zeta_i + r_i \cdot u_i$.

La sperimentazione ha confermato quanto ci si aspettava: un numero molto elevato di test è stato eseguito su polinomi di gradi diversi e con radici esatte di modulo variabile e, partendo da approssimazioni iniziali che verificano la (3.8), il metodo si è sempre arrestato entro i 3 o più raramente 4 passi.

Sia l'errore i-esimo alla m-esima iterazione $\epsilon_i = \epsilon_i^{(m)} = |z_i^{(m)} - \zeta_i|$. La tabella che segue mostra una evoluzione tipica di alcune approssimazioni per un polinomio di grado 70 con radici di modulo al più 15:

m	ϵ_1	ϵ_8	ϵ_{60}
0	$1.654919 \cdot 10^{-4}$	$1.014756 \cdot 10^{-4}$	$3.169143 \cdot 10^{-5}$
1	$1.891020 \cdot 10^{-11}$	$9.148140 \cdot 10^{-12}$	$6.645487 \cdot 10^{-13}$
2	$1.075261 \cdot 10^{-11}$	$4.440892 \cdot 10^{-16}$	$7.021667 \cdot 10^{-16}$

in cui la convergenza di ordine cubico è apprezzabile nella prima iterazione, mentre con la seconda si raggiunge l'ordine della precisione (doppia) di macchina; si noti anche che il verificarsi di (3.8) sulle approssimazione iniziali ha imposto in questo caso una distanza di queste dalle radici reali dell'ordine di 10^{-4}.

In effetti si vede dalla forma di (3.8) che tale condizione è tanto più restrittiva quanto più sono piccoli c e d, cioè quanto più alto è il grado del polinomio e quanto più piccola è la distanza minima tra le approssimazioni iniziali $z_i^{(0)}$, rispettivamente. Ci si aspetta dunque che per un polinomio di grado più basso, con radici (scelte in maniera pseudocasuale) di modulo maggiore, la (3.8) sia verificata anche scegliendo approssimazioni iniziali più lontane dalle radici reali: si confronti con la precedente la prossima tabella, che mostra l'esecuzione del metodo su un polinomio di grado 8 con radici di modulo 1000 al massimo:

m	ϵ_1	ϵ_2	ϵ_7
0	1.389739	1.389739	$4.278648 \cdot 10^{-1}$
1	$7.600935 \cdot 10^{-5}$	$6.709213 \cdot 10^{-5}$	$2.454438 \cdot 10^{-5}$
2	$4.582863 \cdot 10^{-13}$	$4.547474 \cdot 10^{-13}$	$2.842171 \cdot 10^{-14}$

l'ordine di convergenza è sempre 3, ma le approssimazioni iniziali stavolta sono state scelte molto più distanti dalle radici reali (perturbazione dell'ordine di 1 anziché dell'ordine di 10^{-4}).

Uno dei test della suite lancia il metodo (1) su un numero specificato dall'utente di polinomi di vario grado e con limiti diversi sul modulo delle radici, scegliendo approssimazioni iniziali che verifichino la condizione (3.8); registrando l'evoluzione delle norme degli errori $\eta^{(m)} = \sqrt{\sum_{i=1}^{n} {\epsilon_i^{(m)}}^2}$ si è confermato l'ordine cubico di convergenza locale del metodo di Newton-Weierstrass. Risultati-tipo di questo genere di test sono illustrati nelle figure 1 e 3; si è posto in ascissa la norma dell'errore m-esimo $\eta^{(m)}$ e in ordinata $\eta^{(m+1)}$. In verde è riportato per confronto il grafico di $y = x^3$.

[3] due procedure assolvono a questo compito, una delle quali fornisce generalmente risultati migliori. Vedasi la sottosezione 3.2.3 in proposito.

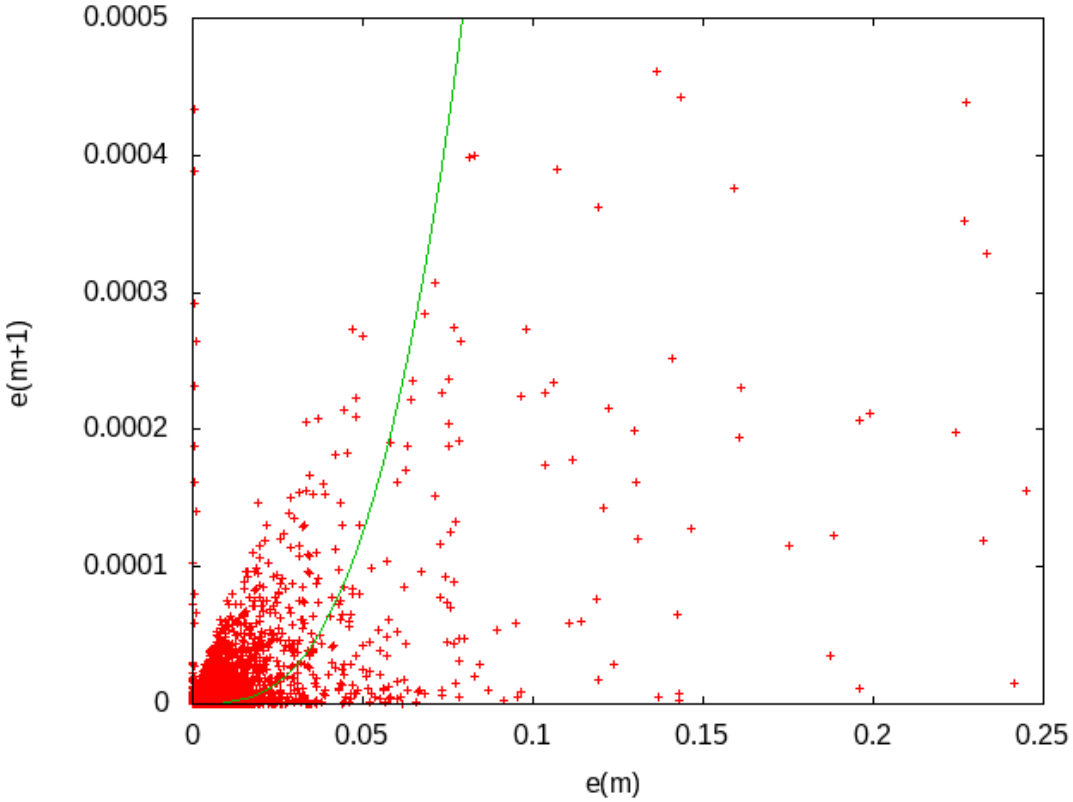

Figura 1: Test su 444000 polinomi di grado compreso tra 4 e 150, con radici esatte di modulo al più 200. Il metodo non ha mai superato le 4 iterazioni e mediamente ne sono occorse 3. Quindi in questo grafico sono presenti circa $3 \cdot 444000 = 1,332 \cdot 10^6$ punti e si può allora considerare risibile il numero di punti che non sta nel sovragrafico di $y = x^3$; vedasi in proposito anche la figura 2.

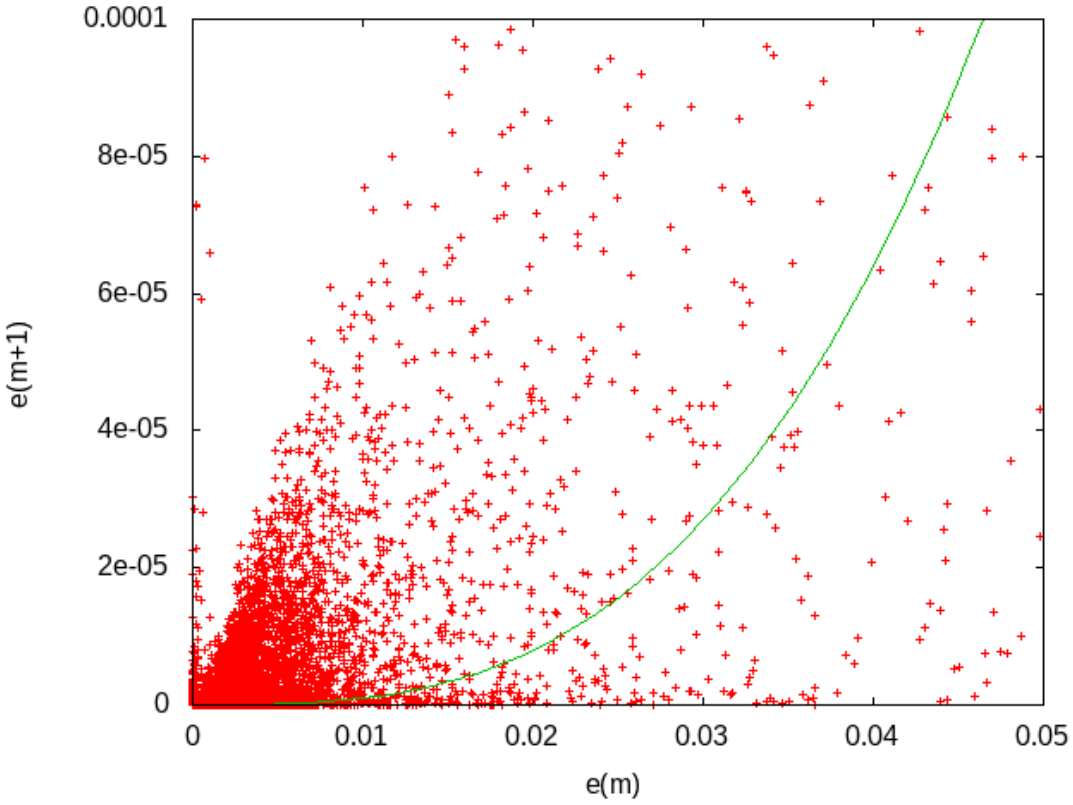

Figura 2: Dal grafico in figura 1: ingrandimento della regione in cui cade la maggior parte dei punti.

2.2 Convergenza globale

Sappiamo dalla teoria che il calcolo di una radice di un polinomio è tanto peggio condizionato quanto più essa "si avvicina" ad essere uno zero multiplo, cioè quanto più è vicina a un'altra radice. Questa caratteristica è catturata anche dalla condizione (3.8), che impone di scegliere approssimazioni iniziali molto vicine agli zeri mal condizionati del polinomio.

2.2.1 Polinomi ciclotomici

Si è testata la convergenza del metodo (1) su polinomi che hanno radici molto ben separate, ovvero i *polinomi ciclotomici* definiti per ogni $n \in \mathbb{N}$ da

$$\psi_n(z) = \prod_{\substack{k=1 \\ \gcd(k,n)=1}}^{n} \left(z - \zeta_{(n)}^k\right)$$

dove $\zeta_{(n)}$ è una radice primitiva n-esima dell'unità, come ad esempio $e^{2\pi i/n}$. Le radici di ψ_n giacciono sulla circonferenza unitaria S^1, e le approssimazioni iniziali sono state prese riscalando questa circonferenza, ossia prendendo per ogni k coprimo con n come approssimazione iniziale k-esima $z_k = \alpha \cdot \zeta_{(n)}^i = \alpha e^{2k\pi i/n}$, con i moduli delle perturbazioni α progressivamente maggiori.

Si è verificato che il metodo è convergente anche per approssimazioni iniziali molto lontane dalle radici esatte, dell'ordine di 10^3; si veda come esempio il grafico in figura 4 che riporta in ascissa i rapporti w/dc e in ordinata i passi necessari per la convergenza.

Il modo in cui il numero di passi impiegati cresce all'aumentare del modulo della perturbazione dipende dall'indice n del polinomio ciclotomico e non dal suo grado; le curve caratteristiche per ogni indice sembrano avere tutte lo stesso andamento generale ma diversi fattori di scala. Come esempio si veda il grafico in figura 5 in cui il colore dei punti dipende dall'indice di ψ_n.

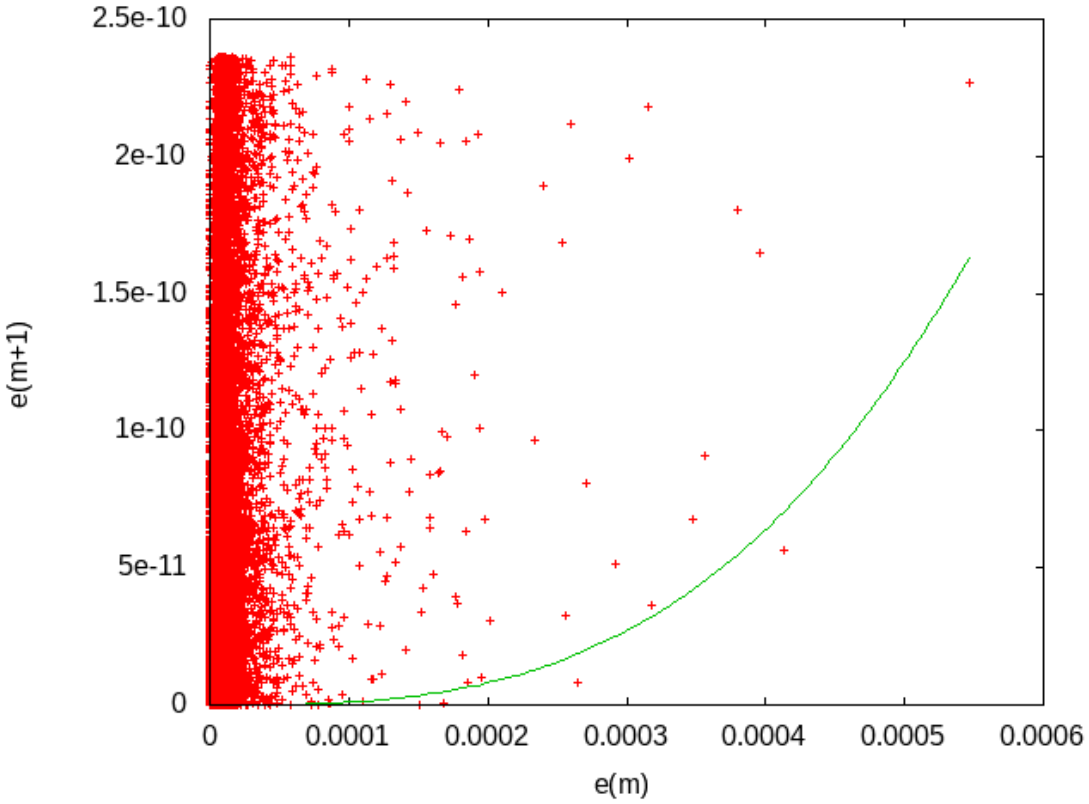

Figura 3: Test condotto su 144000 polinomi di grado compreso tra 4 e 50, con radici di modulo massimo 20. Il numero massimo di iterazioni è stato 4. In questo grafico si mostra soltanto il 90% "migliore" dei punti e cioè quelli corrispondenti a un passo d'iterazione del metodo che più ha ridotto la norma dell'errore. Se ne conclude che la condizione (3.8) sulle approssimazioni iniziali assicura nella gran parte dei casi una convergenza estremamente rapida.

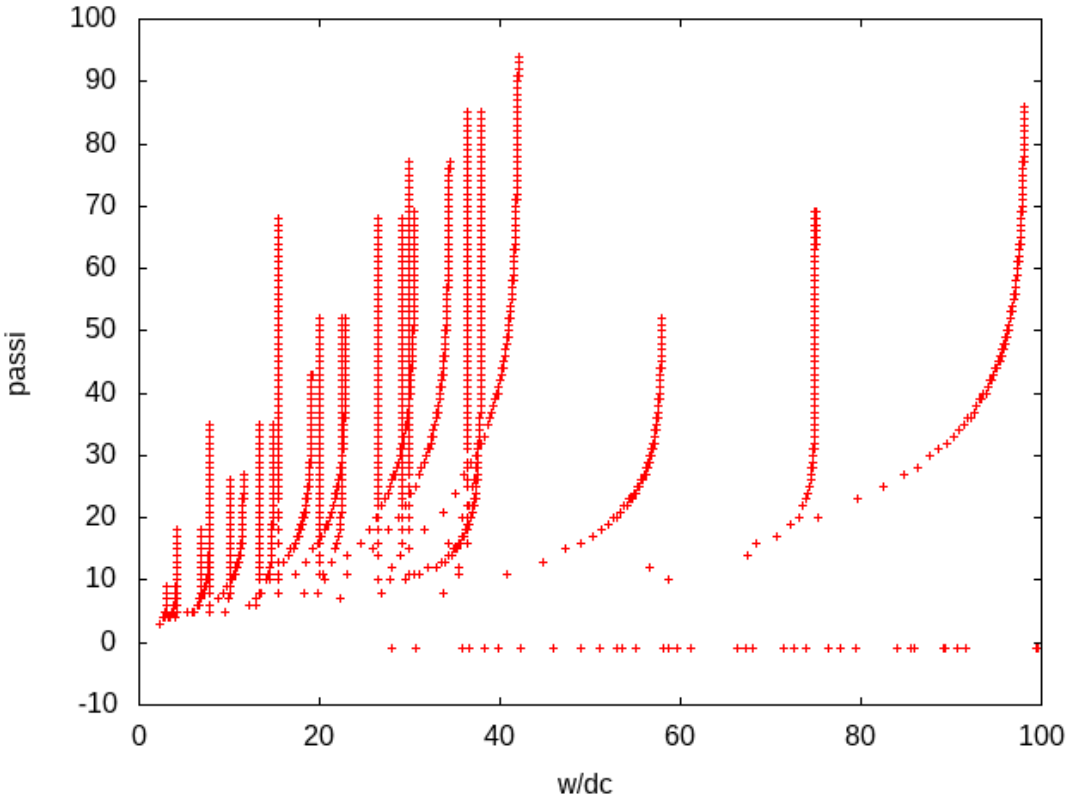

Figura 4: Risultato di 39138 test condotti su ψ_n con n tra 7 e 180. Per ogni n l'approssimazione iniziale si è allontanata di 1 da S^1, fino a una perturbazione di modulo 1000. I punti che giacciono sulla retta $y = -1$ corrispondono a istanze del metodo che divergono: come si può osservare dalla figura 5, esse corrispondono a gradi (indici) alti, per cui perturbazioni molto grandi in modulo determinano un overflow nel calcolo della correzione di Weierstrass.

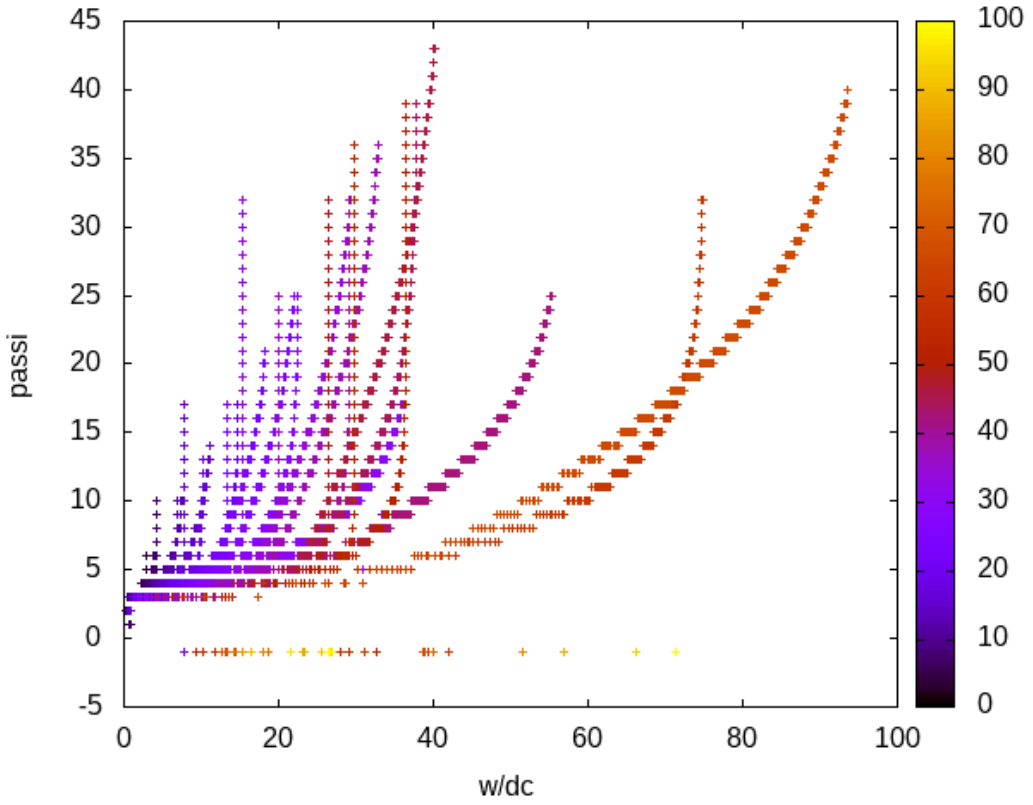

Figura 5: Risultato di 39058 test condotti su ψ_n con n tra 7 e 100. Il modulo massimo raggiunto per la perturbazione iniziale è 20.

2.2.2 Polinomi casuali

Si è voluto verificare quanto la proprietà di convergenza globale dipendesse dalle buone proprietà di separazione z_i; nel caso dei polinomi ciclotomici tale separazione è garantita dalla disposizione delle radici primitive n-esime dell'unità ben distanziate sopra S^1, e dal fatto che le approssimazioni iniziali sono state scelte allontanandosi radialmente.

Nel caso generale, cioè con polinomi generati in maniera pseudocasuale, si è tentato di riprodurre questo effetto scegliendo le condizioni iniziali allontanandosi dalle radici esatte in modo da preservarne le proprietà di separazione.

Un metodo immediato per fare ciò consiste nello scegliere l'approssimazione $z_i^{(0)}$ lungo la direzione che allontana ζ_i dal baricentro di tutte le altre $\frac{1}{n-1}\sum_{j\neq i}\zeta_j$, aumentando progressivamente il modulo dell'errore iniziale. Questo sistema non è infallibile, ed anzi a volte peggiora la separazione delle radici anziché migliorarla: si pensi ad esempio a un polinomio di grado 3 con tre radici distinte che giacciono su una stessa retta.

Nonostante ciò il metodo spesso converge anche molto dopo che l'errore iniziale non verifica più la condizione (3.8), ma la perdita di separazione fa sì che le approssimazioni si "scambino" nel senso che una z_i può convergere verso una ζ_j con $j \neq i$ (cfr. con la (b) della tesi del Teorema 3.1).

Si confrontino le figure 6 e 7 per verificare che preservando la separazione il metodo converge più spesso rispetto alla scelta di approssimazioni iniziali lungo direzioni casuali; quando la condizione (3.8) non è verificata (per $x > 1$ nei due grafici) è comunque presente qualche istanza divergente (sulla retta $y = -1$).

3 Implementazione

Si è implementata dapprima una suite di test piuttosto semplice in Python, per poi passare a una più robusta e completa in C.

3.1 Python

Python è un linguaggio di scripting interpretato, dunque intrinsecamente più lento nell'esecuzione rispetto al C; inoltre senza appoggiarsi a librerie apposite per il calcolo in precisione arbitraria è piuttosto instabile numericamente.

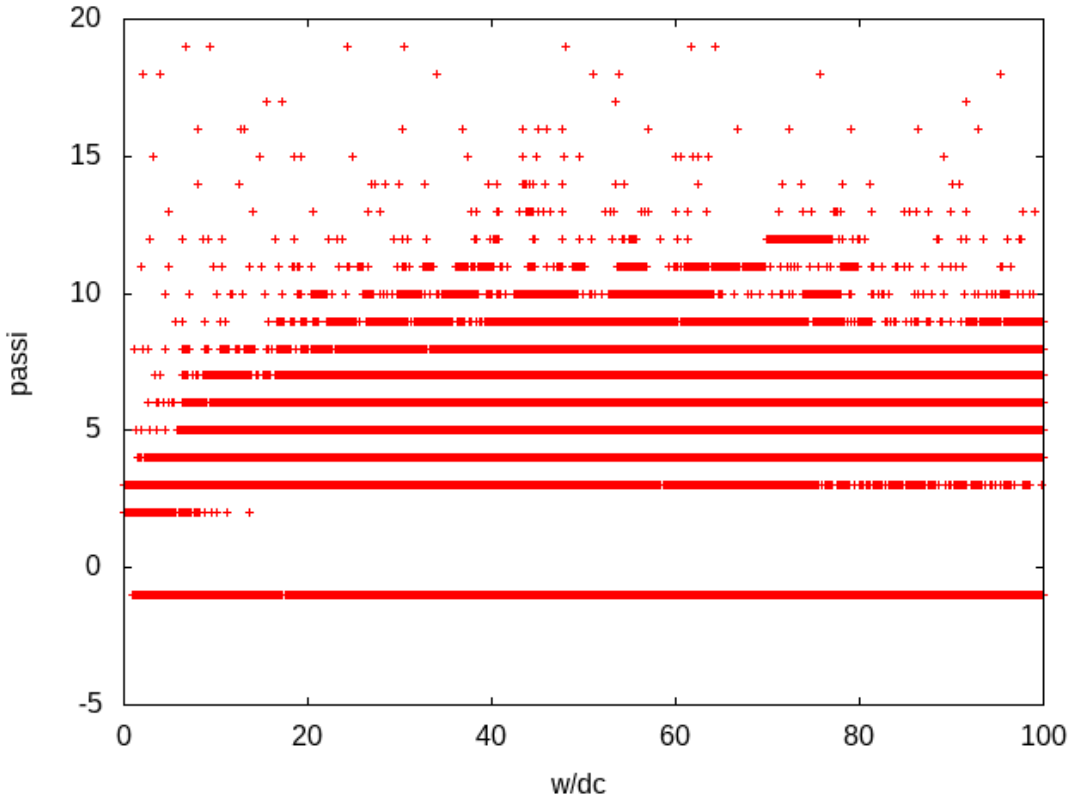

Figura 6: 202556 test eseguiti su 9600 polinomi di grado compreso tra 4 e 100, generati casualmente con radici di modulo inferiore a 100. Le approssimazioni iniziali sono state scelte progressivamente (massima distanza 3.992) più lontane dalle corrispondenti radici reali, ma tentando di preservarne la separazione. In 193030 casi si è avuta la convergenza; le istanze su cui il metodo diverge giacciono sulla retta $y = -1$ e come prescrive la condizione (3.8) non ve ne sono per $x < 1$.

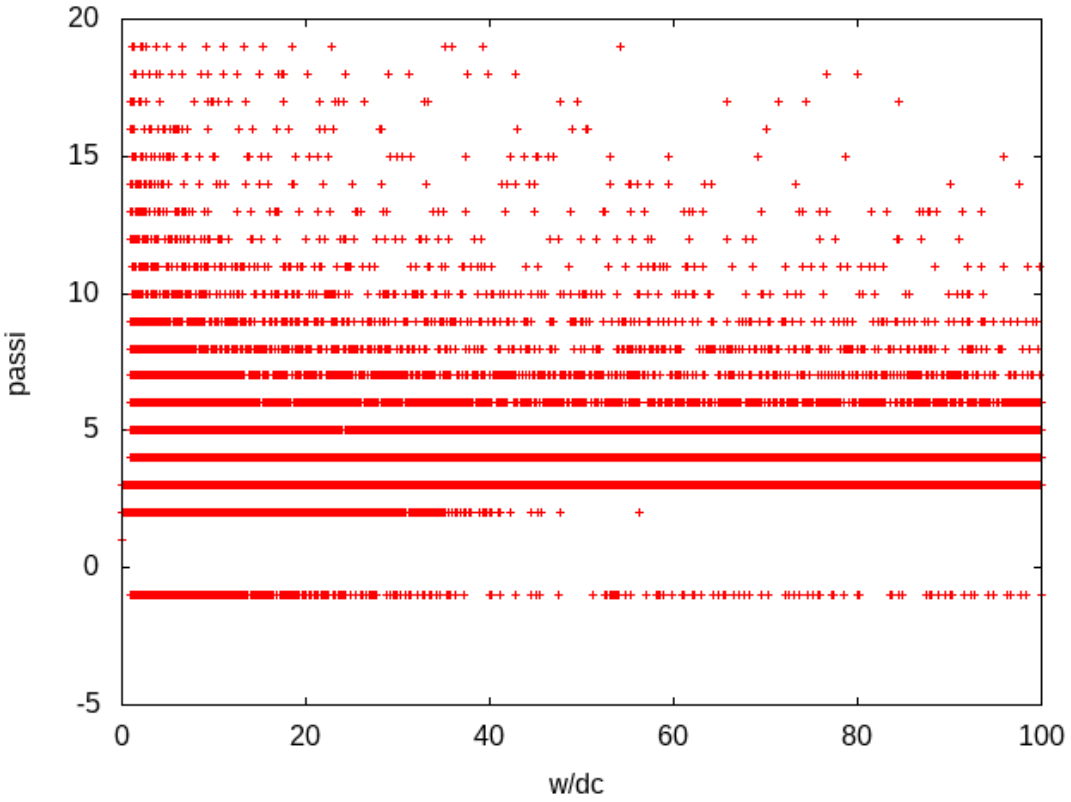

Figura 7: 1362909 test su polinomi generati casualmente con grado tra 4 e 100 e radici di modulo massimo 100. In questo caso le perturbazioni iniziali degli zeri esatti sono state scelte casualmente. In più della metà delle istanze di convergenza si è verificato almeno uno "scambio" delle approssimazioni. Si può notare, rispetto al caso esaminato in figura 6, una concentrazione molto maggiore di casi divergenti anche poco al di sopra della soglia $w/dc = 1$.

Si dovrebbe quindi considerare l'implementazione in Python dell'esperimento soltanto come una "guida" per capire il funzionamento delle corrispondenti procedure in C, il cui codice è spesso più intricato e contenente delle "sottigliezze" proprie del C volte a migliorarne l'efficienza e la robustezza.

Il codice è commentato per spiegare cosa calcola (e come) ogni funzione. Una volta avviato l'interprete interattivo di Python dando il comando `python` dal terminale (nella directory contenente i file sorgenti .py), si deve caricare il modulo principale nw.py con `import nw`; a questo punto è possibile chiamare i metodi definiti nel modulo con la sintassi `valori_di_ritorno = modulo.metodo(argomenti)`, come mostrato nel codice 1 (linea 5).

Codice 1: Esempio di utilizzo del modulo `nw`

```
>>> import nw
>>> p1= [ 4-4j, -9+5j, 10-1j, -6, 1] # p1 dato come lista dei suoi coefficienti
>>> eps= 0.0001
>>> z0= [ -1j + eps, 1 + eps, 4 + eps, (1+1j) + eps ] # radici perturbate
>>> esito= nw.verify_convergence_condition( p1, z0, 4 )
>>> esito
True
```

Si può esplorare l'aiuto contestuale sia per l'intero modulo con `help(nw)`, sia per un singolo metodo ivi definito; ad esempio nel codice 2 si mostra l'output di `help(nw.weierstrass_corr)`, che ne spiega la sintassi e il funzionamento.

Codice 2: Aiuto contestuale per il metodo `weierstrass_corr`

```
>>> help(nw.weierstrass_corr)

Help on function weierstrass_corr in module nw:

weierstrass_corr(p, zetas, n, i=None)
    Calculates the ith Weierstrass correction for polynomial p
    given the vector zetas=[z_1, ..., z_n] of current approximations.
    If the index of required correction (i.e. (i-1) -> W_i) isn't fed,
    then returns a vector containing all the n W_i's
```

3.2 C

Il codice C si compone di un file sorgente `newton-weierstrass.c` che fornisce essenzialmente un'interfaccia di tipo menu contestuale ai metodi veri e propri per la sperimentazione che si trovano nell'header `newton-weierstrass.h`. Le funzionalità ausiliarie per salvare i dati in file di testo ed elaborarli per ricavarne dei grafici si trovano in `analisi.h` e `txt2dat.py`.

Anche nel codice C i commenti dovrebbero spiegare abbastanza bene cosa calcolano le varie procedure e in che modo; in particolare all'inizio dell'header si definiscono alcune costanti come le soglie per i criteri di arresto e i flag di debug.

Per compilare il programma si usi il comando `gcc -lm -Wall -o eseguibile newton-weierstrass.c`; per eseguirlo `./eseguibile`.

3.2.1 Criterio d'arresto

La procedura `NW_metodo` lancia il metodo iterativo prendendo per argomenti il polinomio, le approssimazioni iniziali e un puntatore alle approssimazioni finali in cui scrivere il risultato calcolato.

Si itera finché il numero di passi non supera un limite di sicurezza (fissato dalla costante `MAX_ITERATIONS`), oppure finché non è verificato un criterio d'arresto.

Tale criterio ferma l'iterazione del metodo quando tutte le approssimazioni calcolate sono "sufficientemente buone" in un senso che dipende dalle condizioni iniziali:

- Se le condizioni iniziali verificano la condizione (3.8) allora una radice approssimata è considerata "buona" se è sufficientemente vicina alla corrispondente radice esatta[4], oppure se l'errore relativo ha smesso di diminuire cubicamente[5] (nel qual caso l'ulteriore iterazione del metodo non migliora l'approssimazione).
- Se invece le condizioni iniziali non verificano la (3.8), ciò che accade ad esempio quando si vuole testare la convergenza globale del metodo, una radice approssimata viene considerata soddisfacente solo quando è vicina alla corrispondente radice esatta, secondo lo stesso criterio del caso precedente.

3.2.2 Passo del metodo iterativo

La funzione che esegue una singola iterazione del metodo di Newton-Weierstrass (`NW_step`) utilizza direttamente la formula (1); il calcolo di ogni $\hat{z}_i$ richiede $9n$ operazioni aritmetiche[6] dunque il singolo passo ha costo $9n^2 = O\left(n^2\right)$.

La formula fornita in [5, p. 419]

$$\hat{z}_i = z_i - \frac{W_i}{1 + H_i} \cdot \prod_{j \neq i} \left(1 + \frac{\frac{1}{2} W_i}{z_i - \frac{1}{2} W_i - z_j}\right) \tag{2.5}$$

e usata nei risultati di convergenza, è di scarsa utilità pratica perché più onerosa in termini di operazioni richieste, e non più stabile della (1).

Il calcolo di W_i comporta una divisione per $\prod_{j=1, j\neq i}^{n} (z_i - z_j)$, che può molto grande in modulo se le approssimazioni z_i sono grandi in modulo e ben distanziate oppure molto piccolo se gli z_i sono piccoli in modulo oppure vicini tra loro. Questo inconveniente si amplifica all'aumentare del grado di p, dato che aumenta il numero dei fattori.

Si è dunque inserito un controllo nel calcolo del passo iterativo che smette di approssimare la radice i-esima nel caso in cui il calcolo di W_i abbia determinato un underflow od overflow (linee 5-7 del codice 3).

Codice 3: routine per la singola iterazione del metodo

```
void NW_step ( Polinomio_t p, Radici_t appross , Radici_t *nuove_appross ) {
    Polinomio_t dp= U_derivata( p );
    int i;
    for( i= 0; i<p.Grado; i++ ) {
        if ( !isfinite(NW_correzione( p, appross, i )) ) {
            (*nuove_appross)[i]= appross[i];
        }
        else {
            (*nuove_appross)[i]= appross[i] - ( U_valuta( p, appross[i] ) / \
                U_valuta( dp, appross[i] - 0.5*NW_correzione( p, appross, i ) ) );
        }
    }
}
```

3.2.3 Ricerca delle approssimazioni iniziali

Dato il polinomio p con le sue radici esatte, la routine `NW_trova_appross_furbo` cerca delle approssimazioni iniziali il più lontano possibile dalle corrispondenti radici esatte ma che al contempo verifichino la condizione (3.8). Il motivo del nome è dovuto a una leggera modifica della più ingenua `NW_trova_appross_iniz` che permette in genere di ottenere distanze iniziali $|z_i^{(0)} - \zeta_i|$ maggiori.

Entrambe le routine generano delle direzioni u_i, $|u_i| = 1$ e compiono una ricerca dicotomica sul modulo m da scegliere per propagare l'errore lungo queste direzioni. Nella versione ingenua tale modulo è lo stesso per tutte le radici, mentre la versione "furba" si fa una combinazione baricentrica di m basandosi sulle distanze relative delle z_i, in modo da allontanarle il più possibile dalle corrispondenti ζ_i mantenendo però verificata la (3.8).

Nello specifico si calcolano le mutue distanze delle radici esatte

$$d_i = \min_{j \neq i} |\zeta_i - \zeta_j| \qquad D := \sum_{i=1}^{n} d_i$$

[4] cioè se $|z_i - \zeta_i| <$ `EPSILON`.

[5] cioè se $\epsilon_i^{(m)} / \epsilon_i^{(m+1)} >$ `CA_EPSILON`

[6] che si potrebbe abbassare facilmente a $7n$ riutilizzando la valutazione $p(z_i)$ calcolata per W_i.

e si utilizzano i coefficienti $\alpha_i := d_i/D$ per moltiplicare il modulo m della perturbazione (su si sta compiendo la bisezione):

$$z_i = \zeta_i + \alpha_i \cdot m \cdot u_i.$$

Notiamo che siccome le direzioni u_i sono scelte a caso, non sempre il risultato di questa procedura è apprezzabilmente migliore di quello prodotto da `NW_trova_appross_iniz`, anche se nella grandissima parte dei test l'uso di questa accortezza ha effettivamente migliorato i risultati.

Riferimenti bibliografici

[1] Miodrag S. Petković e Đorđe Herceg. «Point estimation of simultaneous methods for solving polynomial equations: a survey». In: *Journal of Computational and Applied Mathematics* 136 (1-2 2001).

[2] Miodrag S. Petković, Đorđe Herceg e Snežana Ilić. «Safe convergence of simultaneous methods for polynomial zeros». In: *Numerical Algorithms* 17 (3-4 1998).

[3] Miodrag S. Petković, Đorđe Herceg e I. Petković. «On a simultaneous method of Newton-Weierstrass'type for finding all zeros of a polynomial». In: *Applied Mathematics and Computation* 215 (2009), pp. 2456–2463.

[4] Miodrag S. Petković, Ljiljana D. Petković e Lidija Z. Rančić. «Point estimation of simultaneous methods for solving polynomial equations: a survey (II)». In: *Journal of Computational and Applied Mathematics* 205 (1 2007).

[5] Lidija Z. Rančić. «Point Estimation of Cubically Convergent Root Finding Method of Weierstrass'type». In: *Ser. Math. Inform.* 28.4 (2013), pp. 417–428.