Pràctica 3: Interpolació polinòmica i integració numèrica

Arnau Mas

24 d'Abril 2018

Problema 1

L'objectiu d'aquest problema és interpolar la funció $f\colon \mathbb{R} \to \mathbb{R}$ donada per

$$f(x) = \frac{1}{1+25x^2}$$

a l'interval $[-1, 1]$ mitjançant el polinomi interpolador de Lagrange. Es faran servir dos conjunts de nodes diferents per a realitzar la interpolació. En primer lloc, n nodes equidistants dins de l'interval, és a dir, donats per

$$x_k = -1 + \frac{2k}{n}$$

per $k \in \{0, \cdots, n-1\}$. Els altres nodes que farem servir seran nodes de Chebyshev, definits com

$$x_k = \cos\left(\frac{2k+1}{n+1}\frac{\pi}{2}\right)$$

per $k \in \{0, \cdots, n-1\}$. Es proposa fer la interpolació fent servir 4, 8, 16, 32 i 64 nodes.

El programa `nodes.c` genera una llista amb n nodes equidistants o de Chebyshev. Aquesta llista serveix d'entrada per al programa `prob1.c`, que implementa el mètode de diferències dividides de Newton per a calcular els coeficients del polinomi interpolador de Lagrange per als nodes donats. A més avalua aquest polinomi aplicant la regla de Horner als 181 punts proposats i busca el màxim error comès. El fitxer `diferencies_dividides.c` conté la implementació de funcions auxiliars per aquests programes, com el càlcul de diferències dividides fent ús de l'expressió recursiva així com una implementació de la regla de Horner per avaluar un polinomi.

— ∗ —

Per a tenir una idea del màxim error que es comet en cada cas hem avaluat tant la funció com el polinomi en un nombre elevat de punts. En particular en els punts donats per $x_k = -0.989 + k \cdot 0.011$ per $k \in \{0, \cdots, 180\}$, que són 181 punts repartits de forma equidistant a l'interval on estem interpolant. Si denotem per L_n el polinomi de Lagrange

de grau n obtingut a partir dels nodes $\{(x_0, f(x_0), \cdots, (x_n, f(x_n))\}$ aleshores, per tot x de l'interval $[a, b]$ on estem interpolant existeix $\xi \in [a, b]$ tal que

$$f(x) - L_n(x) = \frac{f^{(n+1)}(\xi)}{(n+1)!}\omega(x)$$

on $\omega = (x - x_0) \cdots (x - x_n)$. En particular podem fitar l'error en la interpolació com

$$|f(x) - L_n(x)| \leqslant \frac{\max_{x\in[a,b]} \left|f^{(n+1)}(x)\right|}{(n+1)!} |\omega(x)| .$$

Observem que l'únic que depèn de la tria de nodes és el factor $\omega(x)$. Els nodes de Chebyshev apareixen quan intentem minimitzar $\omega(x)$. La interpolació que hem realitzat posa de manifest la importància de fer una bona tria de nodes: amb nodes equidistants, l'error màxim en $[-1, 1]$ no tendeix a zero quan $n \to \infty$. Interpolant amb nodes de Chebyshev això no passa. A la figura 1.1 hi ha representats el resultat d'interpolar f amb nodes equidistants i de Chebyshev. Quan el nombre de nodes és baix no s'aprecien grans diferències entre les dues tries, i en els dos casos el màxim error es comet a $x = 0$ —en cada gràfic hi ha representat en vermell el punt on es comet el màxim error—. Per a 16 i 32 nodes, però, fent la interpolació amb nodes equidistants, l'error als extrems de l'interval es dispara i arriba a ser superior a 600 per 32 nodes. En canvi, interpolant amb els nodes de Chebyshev el comportament és molt més estable i l'error màxim és manté sempre a $x = 0$.

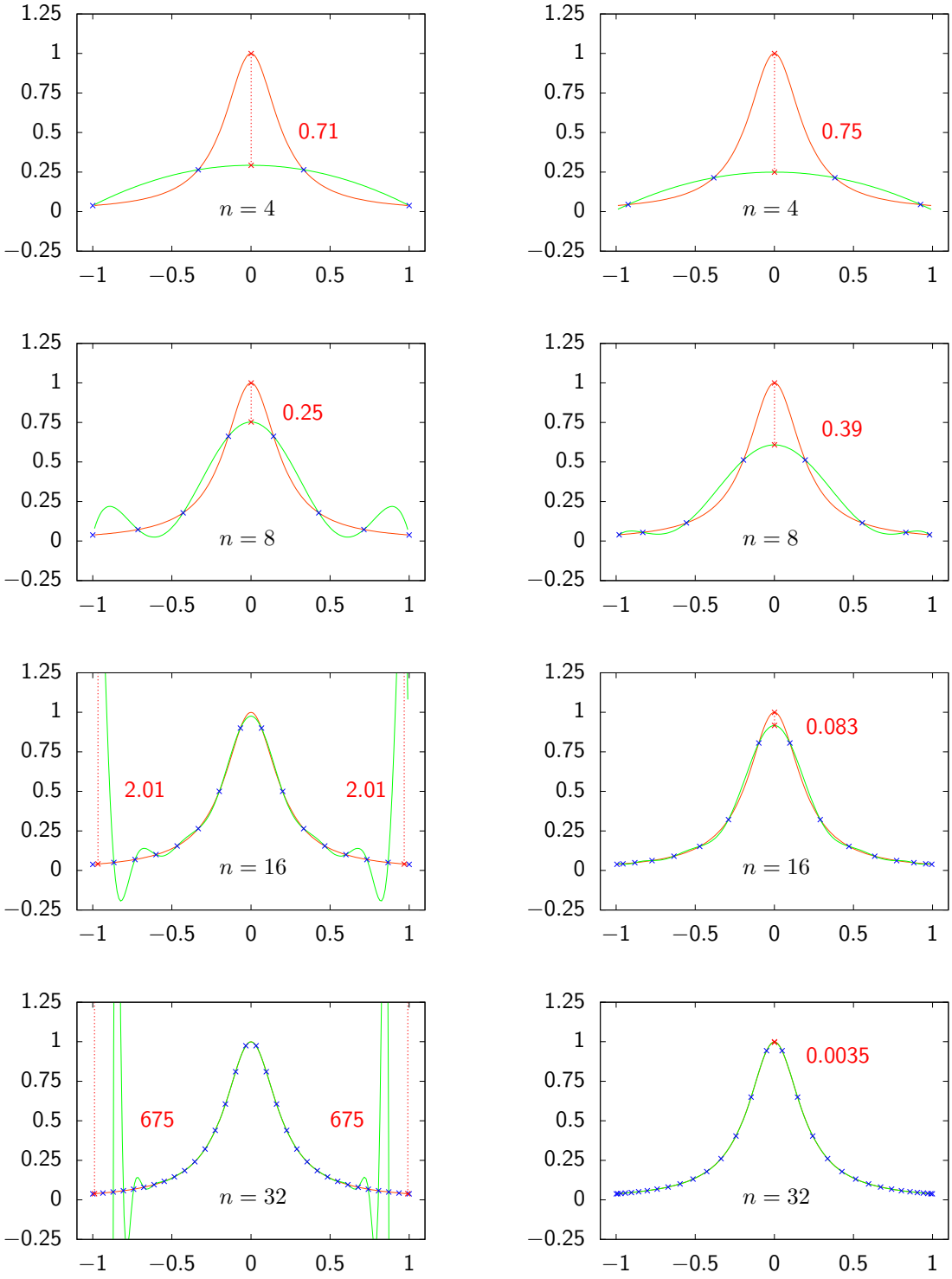

Figura 1.1: Resultat d'interpolar f fent servir n nodes equidistants (esquerra) i n nodes de Chebyshev (dreta). En vermell s'indica el màxim error comès.

Problema 2

Considerem la funció de Bessel de primera espècie d'ordre zero, $J_0(x)$. Volem estimar els valors de les arrels de J_0, és a dir, de les x^* tals que $J_0(x^*) = 0$. Una manera de fer-ho és interpolar la inversa de J_0 a l'interval $(1.9, 3)$. En aquest interval J_0 és localment invertible ja que és estrictament decreixent i derivable. Concretament construirem el polinomi interpolador que té per nodes $(J_0(x_n), x_n)$. D'aquesta manera, si p és el polinomi que obtenim, $p(0)$ és una aproximació de x^*.

La tria de nodes s'ha fet de tres maneres diferents. En primer lloc s'ha interpolat a partir dels nodes amb valors de $J_0(x)$ positius i més propers al canvi de signe. Similarment s'ha interpolat a partir dels nodes amb $J_0(x)$ negatiu i més proper al canvi de signe. I per últim s'ha interpolat prenent un nombre igual de nodes amb $J_0(x)$ positiu i negatiu al voltant del canvi de signe.

El programa `prob2.c` calcula el polinomi interpolador de Lagrange amb el mètode de les diferències dividides de Newton a partir d'un conjunt de nodes donats. Seguidament l'avalua a $x = 0$ fent servir la regla de Horner. La interpolació s'ha fet de grau 1, 3 i 5 –és a dir , amb 2, 4, i 6 nodes— seguint cada un dels tres mètodes detallats anteriorment. Els resultats es mostren a les taules 2.1, 2.2 i 2.3, on també es mostra el resultat d'avaluar J_0 a l'aproximació de l'arrel per tenir una idea de com de bones són cada aproximació —evidentment, com més proper a 0 és $J_0(x^*)$, millor és l'aproximació—.

Taula 2.1: Aproximació de l'arrel de J_0 interpolant a partir de nodes positius propers al canvi de signe.

Grau	x^*	$J_0(x^*)$
1	2.404 728 613 882 804	$5.032\,915\,224\,793\,92 \times 10^{-5}$
3	2.404 822 718 113 948	$1.474\,162\,667\,958\,62 \times 10^{-6}$
5	2.404 825 294 785 460	$1.364\,892\,383\,294\,46 \times 10^{-7}$

Taula 2.2: Aproximació de l'arrel de J_0 interpolant a partir de nodes negatius propers al canvi de signe.

Grau	x^*	$J_0(x^*)$
1	2.400 077 241 947 102	$2.467\,503\,813\,402\,49 \times 10^{-3}$
3	2.404 149 375 353 531	$3.510\,877\,051\,945\,27 \times 10^{-4}$
5	2.404 216 734 868 258	$3.161\,088\,435\,468\,27 \times 10^{-4}$

Taula 2.3: Aproximació de l'arrel de J_0 interpolant a partir de nodes simètrics al voltant del canvi de signe.

Grau	x^*	$J_0(x^*)$
1	2.404 927 513 002 775	$-5.292\,872\,039\,523\,10 \times 10^{-5}$
3	2.404 824 021 911 155	$7.972\,989\,952\,971\,00 \times 10^{-8}$
5	2.404 825 653 043 717	$-4.949\,964\,568\,641\,48 \times 10^{-8}$

El millor resultat l'obtenim amb la interpolació de grau 5 a partir de valors simètrics al voltant del canvi de signe de J_0. En general les millors són les interpolacions amb els

valors positius i amb els simètrics, i la interpolació amb valors negatius és prou dolenta en comparació.

De fet ja podiem esperar des del principi que la millor interpolació fos la simètrica, ja que és l'única que conté l'arrel a l'envolupant convexa dels nodes. A més, en general si tenim els nodes d'interpolació continguts en un interval $[a, b]$ i volem avaluar el polinomi interpolador $p(x)$ en un punt fora d'aquest interval l'error obtingut pot ser molt gran.

Problema 3

El nostre objectiu és obtenir un valor aproximat de la integral

$$I = \int_0^1 \frac{dx}{1+x^2} = \arctan(1) - \arctan(0) = \frac{1}{4}\pi \approx 0.785\,398\,163\,397\,448 \tag{3.1}$$

pel mètode dels trapezis i pel mètode de Simpson dividint l'interval $[0, 1]$ en quatre parts iguals.

El programa `prob3.c` calcula aquestes aproximacions així com una fita de l'error que es comet amb cadascuna. Fent servir el mètode dels trapezis hem obtingut $I \approx 0.782\,794\,117\,647\,059$, que comparat amb el valor exacte fins a 15 decimals de l'equació (3.1) ens dóna un error aproximat de $2.604\,045\,750\,389 \times 10^{-3}$. Amb el mètode de Simpson obtenim $I \approx 0.785\,392\,156\,862\,745$, que comparat amb l'equació (3.1) té un error aproximat de $6.006\,534\,703 \times 10^{-6}$.

Per tant veiem que amb la regla de Simpson hem obtingut una millor aproximació.

Problema 4

Volem obtenir un valor aproximat de la integral

$$I = \int_1^5 \frac{e^x}{x}\, dx$$

pel mètode dels trapezis dividint l'interval $[1, 5]$ en 4, 8, 16, 32 i 64 parts iguals.

El programa `prob4.c` calcula aquestes aproximacions i una estimació de l'error comés. Els resultats obtinguts per a cada n es mostren a la taula 4.1.

Taula 4.1: Resultat i estimació de l'error obtingut per a cada n.

n	Aproximació de I	Estimació de l'error
4	40.239 701 356 634 455	10
8	38.782 928 156 314 796	2.5
16	38.413 711 363 539 406	0.625
32	38.321 069 162 332 130	0.156 25
64	38.297 886 904 128 802	0.039 062 5

L'estimació de l'error l'hem calculat segons la fórmula:

$$\left| \frac{(b-a)F}{12} h^2 \right|,$$

on b i a són els extrems de l'interval —en el nostre cas $b = 5$ i $a = 1$—, F és una fita superior en l'interval de la segona derivada de la funció que volem integrar, i $h = (b-a)/n$ amb n el nombre de parts en les que dividim l'interval. Tenim, per tot $x \geqslant 1$

$$\frac{d^2}{dx^2}\left(\frac{e^x}{x}\right) = \frac{e^x(x^3 - 2x^2 + 2x)}{x^4} \leqslant \frac{e^x}{x},$$

per tant, com que e^x/x és creixent a l'interval $[1, 5]$ podem fitar la segona derivada per $F = e^5/5 \leqslant 30$. De manera que per a cada n ens queda que l'error està fitat per $160n^{-2}$. Observem que per a un major nombre de divisions del interval esperem millorar l'aproximació de I.

Problema 5

Volem calcular amb un error menor que 10^{-2} el valor de la integral

$$I = \int_1^2 \log(x)\, dx$$

utilitzant la regla composta de Simpson.

Una fita de l'error amb aquest mètode ve donada per

$$\varepsilon = \left| \frac{(b-a)F}{180} h^4 \right|$$

on b, a són els extrems del interval, F és una fita superior del valor absolut de la quarta derivada de la funció que volem integrar, i $h = (b-a)/n$ on n és el nombre de divisions de l'interval. En el nostre cas $a = 1$, $b = 2$, $|f^{(4)}(x)| = |-6/x^4| \leqslant 6$ per a $x \in [1, 2]$, de manera que volem

$$\left| \frac{6}{180} \frac{1}{n^4} \right| \leqslant 10^{-2}$$

Per tant necessitarem fer com a mínim $n = 2$ divisions de l'interval $[1, 2]$ per obtenir la precisió demanada.

El programa `prob5.c` calcula aquesta integral amb diversos valors de n. Els resultats es mostren a la taula 5.1.

Taula 5.1: Resultats per a diversos n parells

n	Aproximació de I
2	0.385 834 602 165 434
4	0.386 259 562 814 567
6	0.386 287 163 278 802
8	0.386 292 043 466 313
10	0.386 293 403 804 806
12	0.386 293 897 301 413
14	0.386 294 110 052 020
16	0.386 294 213 675 793
18	0.386 294 268 953 807
20	0.386 294 300 594 357

Observem que amb $n = 2$ iteracions ja hem obtingut dues xifres decimals correctes, per tant la nostra estimació de l'error ha sigut adequada. Si volem garantir un error menor que 10^{-4} necessitem $n \geqslant \left\lceil \sqrt[4]{10^4/30} \right\rceil = 19$ i com que n ha de ser parell $n = 20$.

Problema 6

En aquest problema se'ns demana de calcular una aproximació de la distància recorreguda per un automòbil a partir d'un conjunt finit de valors de la seva velocitat a certs instants de temps. En concret volem aproximar la integral

$$L = \int_0^{84} v(t)\, dt,$$

tenint només valors de $v(t)$ cada 6 s. Per tant estem dividint l'interval $[0, 84]$ en $n = 14$ parts iguals. Per a trobar L utilitzarem la regla de Simpson composta, ja que en general dóna un millor resultat que la regla dels trapezis composta.

El programa `prob6.c` calcula aquesta aproximació. El resultat que obtenim és $L = 2909.400\,000\,000\,000\,091$ m, per tant la pista mesura aproximadament 2.909 km.

Problema 7

Una altra manera d'interpolar funcions que és més estable quan el nombre de nodes creix és a partir de splines. En lloc de construir un polinomi de grau molt alt es construeixen molts polinomis de grau baix —els splines solen ser cúbics— imposant condicions de regularitat als nodes. Concretament hem fet servir splines cúbics amb condicions d'extrem naturals, és a dir, imposant

$$s_0''(x_0) = s_n''(x_n) = 0,$$

on denotarem per s_i l'spline i-èssim. Podem escriure

$$s_i = a_{i0} + a_{i1}(x - x_i) + a_{i2}(x - x)^2 + a_{i3}(x - x_i)^3.$$

Per tant hem de calcular els coeficients a_{ij}. La manera estàndard de procedir és considerar els anomenats moments, $M_i := s_i''(x_i)$, que són la solució d'un sistema lineal que només involucra els nodes d'interpolació $\{(x_i, f(x_i)\}$. A partir dels moments podem reconstruir la resta de coeficients com

$$\begin{aligned} a_{i0} &= f(x_i) \\ a_{i1} &= \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} - \frac{2M_{i+1} + M_i}{6}(x_{i+1} - x_i) \\ a_{i2} &= \frac{M_i}{2} \\ a_{i3} &= \frac{M_{i+1} - M_i}{6(x_{i+1} - x_i)} \end{aligned} \tag{7.1}$$

— ∗ —

Per aproximar f definida com al problema 1 amb quatre splines cúbics primer hem resolt el sistema que determina els moments fent servir 4 nodes de Chebyshev mitjançant SageMath i hem obtingut

$$\begin{aligned} M_0 &= M_3 = 0 \\ M_1 &= M_2 = -0.557\,114. \end{aligned}$$

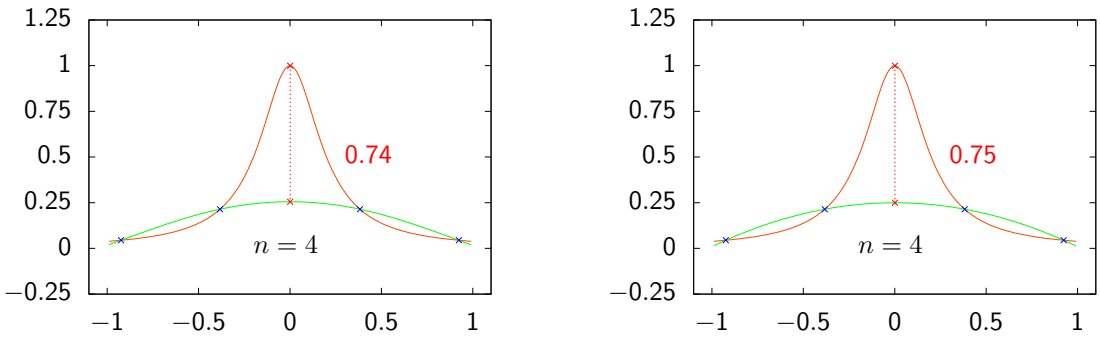

Figura 7.1: Resultat d'interpolar f fent servir interpolació per splines cúbics amb quatre nodes (esquerra) i 4 nodes de Chebyshev (dreta). En vermell s'indica el màxim error comès.

El programa `prob7.c` llegeix la llista de nodes i de moments i implementa els càlculs de l'equació (7.1). Fet això, avalua els splines als 181 punts proposats al problema 1 i busca l'error màxim comès a la interpolació. El resultat, així com la mateixa interpolació fent servir diferències dividides amb quatre nodes de Chebyshev es mostra a la figura 7.1.

Tal i com podem apreciar, la diferència entre ambdós mètodes és petita. En part és perquè 4 és un nombre més aviat petit de nodes, a partir dels quals es pot extreure poca informació sobre la funció a interpolar. En particular, en aquest cas no estem donant pràcticament cap tipus d'informació sobre el comportament de f al voltant del 0. Per a realitzar la interpolació amb un nombre arbitrari d'splines caldria un programa que resolvés el sistema que determina els moments, per al qual encara no disposem d'eines. Així doncs no hem pogut fer la interpolació amb splines per a nombres elevats de nodes. Tot i així és raonable concloure que fer servir diferències dividides amb els nodes de Chebyshev és molt més eficient que no pas fer servir splines, ja que no involucra la resolució d'un sistema i només requereix de $n + 1$ coeficients, en contra dels $4(n - 1)$ coeficients necessaris pels splines.