Poglavje 4

Interpolacija in aproksimacija funkcij

Na interpolacijo naletimo, kadar moramo vrednost funkcije, ki ima vrednosti znane le v posameznih točkah (pravimo jim *interpolacijske točke*), izračunati v kakšni točki, različni od interpolacijskih točk. Ker funkcije zunaj interpolacijskih točk ne poznamo, jo nadomestimo z *interpolacijsko funkcijo*. Da bi bila interpolacijska funkcija dober nadomestek za originalno funkcijo, (ki je lahko poznana analitično ali le v posameznih točkah) mora biti enostavno določljiva, lahko izračunljiva in njene vrednosti se morajo v predpisanih točkah ujemati z vrednostjo originalne funkcije. Navadno uporabljamo za aproksimacijo polinome, lahko pa tudi kakšne druge funkcije, na primer trigonometrične ali eksponentne funkcije.

Interpolacija je uporabna predvsem, kadar imamo funkcijo podano v obliki tabele, zanima pa nas vrednost funkcije v točki, ki leži med tabeliranimi vrednostmi. Nekoč je bil to zelo pomemben problem, danes pa so elektronski kalkulatorji in računalniki skoraj popolnoma izpodrinili uporabo tabeliranih funkcij. Kljub temu pa interpolacijo še vedno pogosto uporabljajo računalniki pri izračunu vrednosti funkcij, kot so trigonometrične, eksponentne, logaritemske in podobne.

V numerični matematiki pa je interpolacija pomembna tudi za numerično odvajanje in integriranje funkcij, kjer funkcije, tudi če so podane z analitično formulo, navadno nadomestimo z ustreznim interpolacijskim polinomom, ki ga laže integriramo ali odvajamo.

Tudi pri aproksimaciji skušamo dano funkcijo, od katere poznamo le vrednosti v nekaj točkah, nadomestiti z drugo, preprostejšo. Razlika je le v tem,

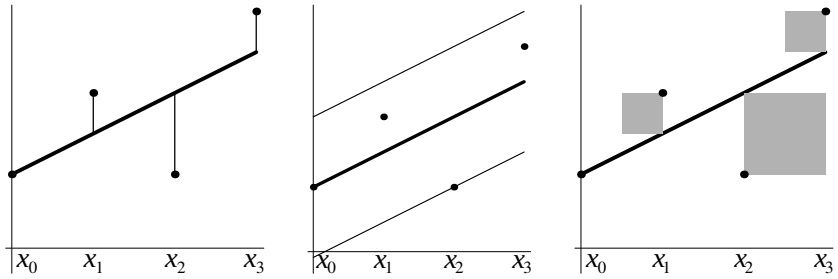
da imamo navadno pri aproksimaciji znane vrednosti v večih točkah, vendar te vrednosti običajno niso brez napak, zato ne zahtevamo, da se mora aproksimacijska funkcija točno ujemati z danimi vrednostmi. Pri tem se moramo vprašati, kako meriti napako take aproksimacije. Odgovor na to vprašanje je seveda odvisen od tega, kaj želimo z aproksimacijo doseči. Poglejmo si nekaj tipičnih možnosti:

1. Funkcijo f imamo podano le v posameznih točkah x_i, $i = 1, \ldots, n$, kjer je število točk n lahko razmeroma veliko. Te vrednosti so pogosto le približne (rezultat meritev ali približno izračunane vrednosti). Ker je točk veliko, funkcijske vrednosti pa le približne, bi z interpolacijo dobili le slab približek. Aproksimacijska napaka e_i v posamezni točki je razlika med dano vrednostjo f_i in vrednostjo aproksimacijske funkcije v tej točki $g(x_i)$:
$$e_i = f_i - g(x_i),$$
mero za celotno napako E pa lahko izberemo na več načinov:

   (a) Celotna napaka je vsota absolutnih vrednosti napak v posameznih točkah
   $$E_s = \sum_{i=1}^{n} |e_i|.$$
   (glej sliko 4.1 (a).)

   (b) Celotna napaka je enaka največji izmed absolutnih vrednosti napak v posameznih točkah
   $$E_e = \max_i |e_i|.$$
   Aproksimaciji, pri kateri izbiramo aproksimacijsko funkcijo tako, da je E_e najmanjša, pravimo *(diskretna) enakomerna* aproksimacija. (glej sliko 4.1 (b).)

   (c) Celotna napaka je kvadratni koren vsote kvadratov napak v posameznih točkah
   $$E_{LSQ} = \sqrt{\sum_{i=1}^{n} e_i^2}.$$
   Aproksimaciji, pri kateri izbiramo aproksimacijsko funkcijo tako, da je E_{LSQ} najmanjša, pravimo *aproksimacija z metodo najmanjših kvadratov.* (glej sliko 4.1 (c).)



Slika 4.1: Različni kriteriji za aproksimacijo funkcij — (a) vsota absolutnih vrednosti napak; (b) enakomerna aproksimacija; (c) aproksimacija z metodo najmanjših kvadratov

2. Funkcija f je zvezna na intervalu $[a, b]$. Aproksimacijska napaka $e(x)$ v posamezni točki je razlika med vrednostjo funkcije $f(x)$ in vrednostjo aproksimacijske funkcije $g(x)$:

$$e(x) = f(x) - g(x),$$

mero za celotno napako E pa lahko podobno kot prej izberemo na več načinov:

(a) Celotna napaka je integral absolutne vrednosti napake

$$E_s = \int_a^b |e(x)|\, dx.$$

(b) Celotna napaka je enaka maksimumu absolutne vrednosti napake

$$E_e = \max_{x\in[a,b]} |e(x)|.$$

Aproksimaciji, pri kateri izbiramo aproksimacijsko funkcijo tako, da je E_e najmanjša, pravimo *najboljša* ali *enakomerna* aproksimacija.

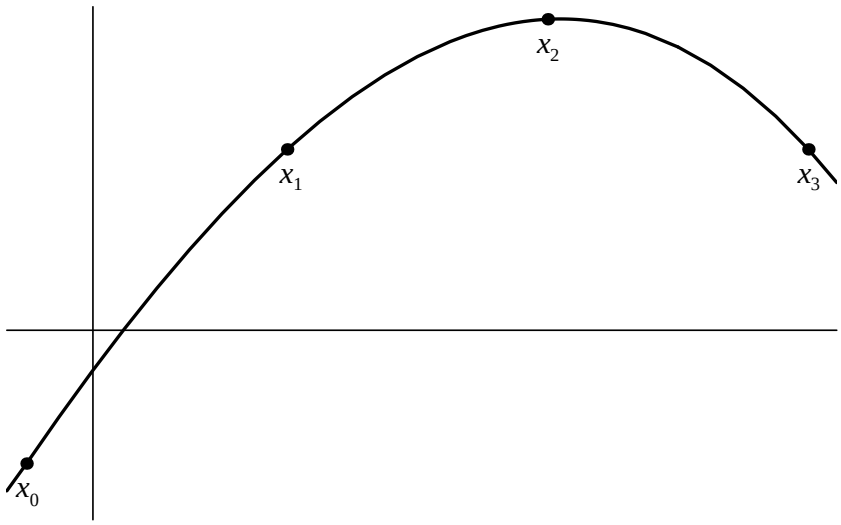
(c) Celotna napaka je kvadratni koren integrala kvadrata napake

$$E_{LSQ} = \sqrt{\int_a^b e^2(x)\, dx}.$$

Aproksimaciji, pri kateri izbiramo aproksimacijsko funkcijo tako, da je E_{LSQ} najmanjša, pravimo *(zvezna) aproksimacija z metodo najmanjših kvadratov*

V tem poglavju se bomo najprej ukvarjali z interpolacijo: formulirali bomo problem, s katerim se bomo ukvarjali v nadaljevanju in dokazali obstoj in enoličnost interpolacijskega polinoma. Posebej si bomo ogledali najprej Lagrangeovo, nato pa še Newtonovo obliko interpolacijskega polinoma, spotoma pa se bomo seznanili tudi z deljenimi diferencami ter s poenostavitvami interpolacijskih formul v primeru, ko so interpolacijske točke ekvidistantne. Spoznali bomo tudi, kakšna je napaka interpolacijskega polinoma. Na koncu pa si bomo ogledali še polinomsko aproksimacijo z metodo najmanjših kvadratov in aproksimacijo periodičnih funkcij z metodo najmanjših kvadratov.

4.1 Polinomska interpolacija



Slika 4.2: Interpolacijski polinom skozi 4 točke

Tipičen problem polinomske interpolacije lahko opišemo takole: Danih naj bo $n+1$ različnih točk $x_0, x_1, \ldots, x_n$ in funkcija f, definirana na intervalu, ki vsebuje te točke. Vrednost funkcije f v posamezni točki naj bo $f(x_i) = f_i$; $i = 0, \ldots, n$. Iščemo polinom $p(x)$ stopnje $\leq n$, da bo

$$p(x_i) = f_i; \quad i = 0, 1, \ldots, n. \tag{4.1}$$

Na sliki 4.2 je interpolacijski polinom, katerega graf poteka skozi štiri predpisane točke.

V načelu lahko koeficiente polinoma $p(x) = a_0 + xa_1 + \cdots + a_n x^n$ izračunamo iz sistema linearnih enačb

$$\begin{array}{ccccccccc} a_0 & + & x_0 a_1 & + & \cdots & x_0^n a_n & = & f_0 \\ a_0 & + & x_1 a_1 & + & \cdots & x_1^n a_n & = & f_1 \\ \vdots & & \vdots & & & \vdots & & \vdots \\ a_0 & + & x_n a_1 & + & \cdots & x_n^n a_n & = & f_n, \end{array} \tag{4.2}$$

z Gaussovo eliminacijo ali LU-razcepom, vendar se pokaže, da ima ta pristop nekaj bistvenih pomankljivosti in da obstajajo tudi boljše in ekonomičnejše metode:

- Reševanje sistema (4.2) zahteva $O(n^3)$ operacij. Kot bomo videli, obstajajo metode, ki zahtevajo le $O(n^2)$ operacij.
- Sistem (4.2) je lahko slabo pogojen, čemur se s primernejšo metodo lahko izognemo.
- Ko rešimo sistem (4.2), nimamo še nobene informacije o napaki interpolacijskega polinoma, medtem ko pri nekaterih drugih metodah dobimo obenem z interpolacijskim polinomom tudi oceno ali približek za njegovo napako.

Matrika sistema (4.2) ima posebno obliko, ki je poznana kot *Vandermondova matrika*. Njena determinanta je enaka produktu vseh možnih razlik

$$\det \begin{vmatrix} 1 & x_0 & x_0^2 & \cdots & x_0^n \\ 1 & x_1 & x_1^2 & \cdots & x_1^n \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^n \end{vmatrix} = \prod_{\substack{i,j=0 \\ i>j}}^{n} (x_i - x_j),$$

torej je nesingularna natanko tedaj, kadar so vsi x_i; $i = 0, 1, \ldots, n$ med seboj različni. To ugotovitev lahko zapišemo kot

Izrek 4.1.1. *Kadar so točke $x_0, x_1, \ldots, x_n$ vse med seboj različne, obstaja polinom p stopnje $\leq n$, ki zadošča pogojem* (4.1) *za vse vrednosti $f_0, f_1, \ldots, f_n$.*

Pokažimo še, da je interpolacijski polinom s pogoji (4.1) enolično določen. Spomnimo se znanega izreka iz algebre:

Izrek 4.1.2. *Če ima polinom stopnje n več kot n ničel, potem je identično enak 0.*

Denimo, da imamo dva polinoma stopnje ne več kot n, ki oba zadoščata pogojem (4.1). Njuna razlika, ki je tudi polinom stopnje ne več kot n, ima vrednost 0 v $n+1$ točki x_i; $i = 0, 1, \ldots, n$, torej je identično enaka 0 in oba interpolacijska polinoma sta nujno enaka. S tem smo pokazali, da je interpolacijski polinom, ki zadošča pogojem (4.1) en sam.

4.2 Lagrangeova interpolacijska formula

Da bi lahko konstruirali interpolacijski polinom, ki ustreza pogojem (4.1), v Lagrangeovi obliki, rešimo najprej naslednjo pomožno nalogo: za dane, med seboj različne točke $x_0, x_1, \ldots, x_n$ konstruirajmo $n+1$ polinomov l_i; $i = 0, 1, \ldots, n$ (pravimo jim *Lagrangeovi*[1] *polinomi*), katerih stopnja naj ne bo več kot n, da bo i-ti polinom zadoščal pogojem

$$l_i(x_j) = \begin{cases} 0 & \text{za} \quad i \neq j, \\ 1 & \text{za} \quad i = j. \end{cases} \tag{4.3}$$

Ker mora imeti polinom l_i ničle v n točkah x_j; $j \neq i$ in njegova stopnja ne sme biti večja od n, ga lahko zapišemo kot produkt

$$l_i(x) = C_i \prod_{\substack{j=0 \\ i \neq j}}^{n} (x - x_j), \quad i = 0, \ldots, n,$$

kjer konstanto C_i določimo tako, da bo $l_i(x_i) = 1$, kar nam da

$$C_i = \frac{1}{\prod_{\substack{j=0 \\ i \neq j}}^{n} (x_i - x_j)}, \quad i = 0, \ldots, n.$$

Tako vidimo, da polinome l_i, ki zadoščajo pogojem (4.3) lahko zapišemo kot

$$l_i(x) = \prod_{\substack{j=0 \\ i \neq j}}^{n} \frac{x - x_j}{x_i - x_j}, \quad i = 0, \ldots, n. \tag{4.4}$$

[1] Joseph-Louis Lagrange (1736 Torino – 1813 Paris), francoski matematik. Sodeloval je pri ustanovitvi dveh znamenituh francoskih visokih šol, *École Polytechnique* in *École Normale*. S teorijo interpolacije se je ukvarjal med svojim bivanjem v Berlinu (1766–1787), ko je bil direktor matematičnega oddelka Berlinske akademije.

S tem smo našli polinome, ki zadoščajo pogojem (4.3).

S pomočjo Lagrangeovih polinomov l_i lahko zapišemo polinom stopnje ne več kot n, ki zadošča pogojem (4.1) kot linearno kombinacijo Lagrangeovih polinomov

$$p(x) = \sum_{i=0}^{n} f_i l_i(x). \tag{4.5}$$

Lahko se prepričamo, da je ravno p tisti polinom, ki zadošča pogojem (4.1).

Če pišemo $\Pi(x) = (x-x_0)(x-x_1)\cdots(x-x_n)$, lahko Lagrangeovo formulo (4.5, 4.4) zapišemo preprosto kot

$$p(x) = \Pi(x) \sum_{i=0}^{n} \frac{f_i}{(x-x_i)\Pi'(x_i)}.$$

Primer 4.2.1. Izračunajmo približek za $\cos 0.15$, če imamo funkcijo cos tabelirano v tabeli 4.1 (prava vrednost $\cos 0.15 \approx 0.988771$).

x	$\cos x$
0.0	1.000000
0.1	0.995004
0.2	0.980066
0.3	0.955336

Tabela 4.1: Tabela vrednosti funkcije cos

1. Začnimo z linearno interpolacijo. Vzemimo dve sosednji točki $x_1 = 0.1$ in $x_2 = 0.2$ ter izračunajmo oba Lagrangeva polinoma

   $$l_1(x) = \frac{x-x_2}{x_1-x_2} \quad \text{in} \quad l_2(x) = \frac{x-x_1}{x_2-x_1},$$

   katerih vrednosti pri $x = 0.15$ sta

   $$l_1(0.15) = l_2(0.15) = 0.5,$$

   tako da je linearni približek za $\cos 0.15$ enak

   $$p_1(0.15) = \cos 0.1 \cdot l_1(0.15) + \cos 0.2 \cdot l_2(0.15) = 0.987535.$$

   Absolutna napaka linearnega približka je torej enaka

   $$e_1 = p_1(0.15) - \cos 0.15 \approx 0.001236.$$

2. Izračunajmo približek za $\cos 0.15$ s kvadratno interpolacijo skozi točke $x_1 = 0.1$, $x_2 = 0.2$ in $x_3 = 0.3$. Vrednosti ustreznih Lagrangeovih polinomov pri $x = 0.15$ so

$$l_1(0.15) = 0.375; \quad l_2(0.15) = 0.75; \quad l_3(0.15) = -0.125,$$

torej je kvadratni približek za $\cos 0.15$ enak

$$p_2(0.15) = \sum_{i=0}^{2} \cos x_i l_i(0.15) = 0.988759,$$

in njegova absolutna napaka

$$e_2 = p_2(0.15) - \cos 0.15 \approx -0.000012.$$

3. Izračunajmo še približek za $\cos 0.15$ s kubično interpolacijo skozi vse štiri točke $x_0 = 0$, $x_1 = 0.1$, $x_2 = 0.2$ in $x_3 = 0.3$ tabele 4.1. Najprej vrednosti Lagrangeovih polinomov pri $x = 0.15$

$$\begin{aligned} l_0(0.15) &= -0.0625; \quad l_1(0.15) = 0.5625, \\ l_2(0.15) &= 0.5625; \quad l_3(0.15) = -0.0625. \end{aligned}$$

Kubični približek za $\cos 0.15$ je torej

$$p_3(0.15) = \sum_{i=0}^{3} \cos x_i l_i(0.15) = 0.988768,$$

in njegova absolutna napaka

$$e_2 = p_3(0.15) - \cos 0.15 \approx -0.000003.$$

Iz primerjave rezultatov vidimo, da je napaka približka pada s stopnjo interpolacijskega polinoma. ■

Lagrangeova oblika interpolacijskega polinoma (4.5) je uporabna predvsem takrat, kadar vnaprej poznamo stopnjo interpolacijskega polinoma, vrednost interpolacijskega polinoma pa nas zanima le v kakšni posamezni točki.

Glavna pomanjkljivost Lagrangeove interpolacije je, da pri dodajanju novih interpolacijskih točk (da povečamo stopnjo interpolacijskega polinoma in s tem natančnost) ne moremo izkoristiti že izračunanih rezultatov, torej moramo vse Lagrangeove polinome (4.4) ponovno izračunati od začetka.

4.3 Newtonova interpolacijska formula

Že v 1. poglavju smo srečali polinom, zapisan v *Newtonovi obliki* (1.4). Kadar za parametre c_i; $i = 1, \ldots, n$ vzamemo interpolacijske točke x_i; $i = 0, 1, \ldots, n-1$, dobimo Newtonovo obliko interpolacijskega polinoma

$$\begin{aligned} p_n(x) &= a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \cdots + \\ &\quad a_n(x - x_0)(x - x_1)\cdots(x - x_{n-1}). \end{aligned} \tag{4.6}$$

Za vsako celo število k med 0 in n naj bo polinom $q_k(x)$ vsota prvih $k+1$ členov polinoma $p_n(x)$:

$$\begin{aligned} q_k(x) &= a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + \cdots \\ &+ a_k(x - x_0)(x - x_1)\cdots(x - x_{k-1}). \end{aligned}$$

Ker imajo vsi preostali členi skupen faktor $(x - x_0)\cdots(x - x_k)$, lahko interpolacijski polinom $p_n(x)$ zapišemo v obliki

$$p_n(x) = q_k(x) + (x - x_0)\cdots(x - x_k)r(x),$$

kjer je $r(x)$ nek polinom. Opazimo, da je člen $(x - x_0)\cdots(x - x_k)r(x)$ enak nič v interpolacijskih točkah $x_0, x_1, \ldots, x_k$, kar pomeni, da mora biti $q_k(x)$ že sam interpolacijeki polinom skozi te točke, ker pa je ta en sam, mora biti $p_k(x) = q_k(x)$.

To nam omogoča, da Newtonov interpolacijski polinom (4.6) konstruiramo po členih kot zaporedje Newtonovih interpolacijskih polinomov p_0, $p_1, \ldots, p_n$, kjer polinom p_i stopnje $\leq i$ poteka skozi interpolacijske točke $x_0, x_1, \ldots, x_i$. Zaporedje gradimo tako, da dobimo polinom p_i z dodajanjem naslednjega člena polinomu p_{i-1}:

$$p_i(x) = p_{i-1}(x) + a_i(x - x_0)\cdots(x - x_{i-1}).$$

Iz tega tudi vidimo, da je koeficient a_i vodilni koeficient i-tega polinoma $p_i(x)$, torej je njegova vrednost odvisna le od interpolacijskih točk $x_0, x_1, \ldots, x_{i-1}$ in funkcijskih vrednosti v teh točkah. Imenujemo ga *i-ta deljena diferenca v točkah* $x_0, x_1, \ldots, x_i$ in ga zapišemo kot

$$a_i = f[x_0, x_1, \ldots, x_i].$$

Tako lahko zapišemo Newtonov interpolacijski polinom kot

$$p_n(x) = \sum_{i=0}^{n} f[x_0, x_1, \ldots, x_i] \prod_{j=0}^{i-1} (x - x_j). \tag{4.7}$$

Ker je

$$p_0(x) = f_0,$$

je

$$f[x_0] = f_0.$$

Prav tako zaradi

$$p_1(x) = f[x_0] + f[x_0, x_1](x - x_0)$$

velja

$$f[x_0, x_1] = \frac{f_1 - f_0}{x_1 - x_0}.$$

Naj bo $p_i(x)$ interpolacijski polinom stopnje ne več kot i skozi točke x_j; $j = 0, 1, \ldots, i$ in naj bo $r_{k-1}(x)$ interpolacijski polinom stopnje ne več kot $k-1$ skozi točke x_j; $j = 1, \ldots, k$. Potem je

$$\frac{x - x_0}{x_k - x_0} r_{k-1}(x) + \frac{x_k - x}{x_k - x_0} p_{k-1}(x) \tag{4.8}$$

polinom stopnje $\leq k$, ki se v interpolacijskih točkah $x_0, x_1, \ldots, x_k$ ujema s predpisani vrednostmi, torej je zaradi enoličnosti enak interpolacijskemu polinomu $p_k(x)$. Zato je vodilni koeficient polinoma $p_k(x)$, ki smo ga pisali kot $f[x_0, x_1, \ldots, x_k]$, enak

$$\begin{aligned} &f[x_0, x_1, \ldots, x_k] = \\ &\quad \frac{\text{vodilni koeficient } r_{k-1}}{x_k - x_0} - \frac{\text{vodilni koeficient } p_{k-1}}{x_k - x_0} = \\ &\quad \frac{f[x_1, \ldots, x_k] - f[x_0, \ldots, x_{k-1}]}{x_k - x_0}. \end{aligned}$$

S tem smo dokazali tudi rekurzivno pravilo za računanje deljenih diferenc višjih redov

$$f[x_0, \ldots, x_k] = \frac{f[x_1, \ldots, x_k] - f[x_0, \ldots, x_{k-1}]}{x_k - x_0}. \tag{4.9}$$

x_i	f_i	$f[\cdot,\cdot]$	$f[\cdot,\cdot,\cdot]$	$f[\cdot,\cdot,\cdot,\cdot]$	$f[\cdot,\cdot,\cdot,\cdot,\cdot]$
x_0	f_0				
		$f[x_0,x_1]$			
x_1	f_1		$f[x_0,x_1,x_2]$		
		$f[x_1,x_2]$		$f[x_0,x_1,x_2,x_3]$	
x_2	f_2		$f[x_1,x_2,x_3]$		$f[x_0,x_1,x_2,x_3,x_4]$
		$f[x_2,x_3]$		$f[x_1,x_2,x_3,x_4]$	
x_3	f_3		$f[x_2,x_3,x_4]$		
		$f[x_3,x_4]$			
x_4	f_4				

Tabela 4.2: Tabela deljenih diferenc

Deljena diferenca k-tega reda je torej diferenčni kvocient deljenih diferenc reda $k-1$. S pomočjo pravila (4.9) lahko sestavimo *tabelo deljenih diferenc* (glej tabelo 4.2).

Ker so deljene diference ravno koeficienti Newtonovega interpolacijskega polinoma (4.7), zapišemo algoritem, ki nam izračuna tabelo deljenih diferenc in vrednosti interpolacijskega polinoma.

Algoritem 4.3.1 (Newtonov interpolacijski polinom)**.** Pri danih vektorjih neodvisnih spremenljivk x_i; $i = 1,\ldots,n+1$ in funkcijskih vrednosti f_i; $i = 1,\ldots,n+1$ naslednji algoritem izračuna vrednost Newtonovega interpolacijskega polinoma v točki t.

$a = \mathbf{zeros}(n+1, n+1)$
$a(:,1) = f$
$p = a(1,1)$
$u = 1$
for $i = 2 : n+1$
    $u = u * (t - x(i-1))$
    for $j = 2 : i$
        $a(i,j) = (a(i,j-1) - a(i-1,j-1))/(x(i) - x(i-j+1))$
    end
    $p = p + a(i,i) * u$
end

Preštejmo število operacij, ki so potrebne za izračun vrednosti interpola-

cijskega polinoma z algoritmom 4.3.1:

$$\sum_{i=2}^{n+1}(2+\sum_{j=2}^{i}1) = \frac{n^2+5n}{2}.$$

Primer 4.3.1. Izračunajmo približek za $\cos 0.15$ še v Newtonovi obliki. Funkcija cos je tabelirana v tabeli 4.1, njena iskana vrednost pa je enaka $\cos 0.15 \approx 0.988771$.

x	$\cos x$			
0.0	1.000000			
		-0.04996		
0.1	0.995004		-0.4971	
		-0.14938		0.025
0.2	0.980066		-0.4896	
		-0.24730		
0.3	0.955336			

Tabela 4.3: Tabela deljenih diferenc za funkcijo cos

1. Iz tabele deljenih diferenc (tabela 4.3) preberemo koeficiente linearnega interpolacijskega polinoma skozi točki x_0 in x_1:

$$p_1(t) = 1 - t \cdot 0.04996, \tag{4.10}$$

zato

$$p_1(0.15) = 1 - 0.15 \cdot 0.04996 = 0.992506.$$

Njegova napaka je

$$e_1 = p_1(0.15) - \cos 0.15 = 0.003735.$$

2. Ko dodamo točko x_2, moramo približku (4.10) dodati naslednji člen:

$$p_2(t) = p_1(t) - 0.4971(t-x_0)(t-x_1) \quad \text{in} \quad p_2(0.15) = 0.988778. \tag{4.11}$$

Napaka tega približka je

$$e_2 = p_2(0.15) - \cos 0.15 = 0.000007.$$

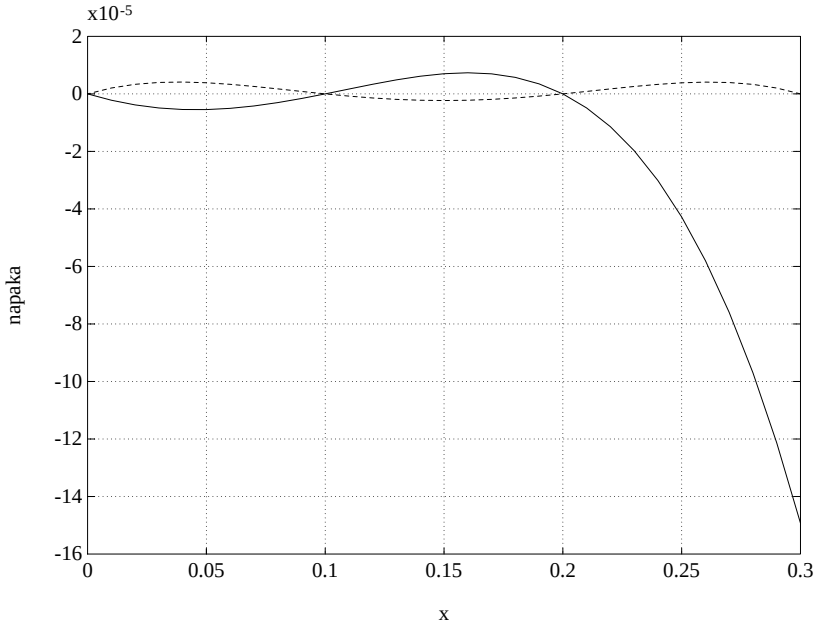
3. Za kubični približek moramo dodati še četrto točko x_3, kar pomeni, da približku (4.11) dodamo naslednji člen

$$p_3(t) = p_2(t) + 0.025(t - x_0)(t - x_1)(t - x_2) \tag{4.12}$$

$$p_3(0.15) = 0.988769.$$

Napaka kubičnega približka je

$$e_3 = p_3(0.15) - \cos 0.15 = -0.000002.$$



Slika 4.3: Napaka interpolacijskega polinoma p_2 (4.11) in p_3 (4.12)

Tudi v tem primeru lahko ugotovimo, da napaka polinomske aproksimacije pada, ko raste število interpolacijskih točk. Kako se obnaša napaka interpolacijskih polinomov p_2 (4.11) in p_3 (4.12) na celem intervalu $[0, 0.3]$, pa si lahko ogledamo na sliki 4.3. ■

4.4 Interpolacija ekvidistantnih tabel

Lagrangeova in Newtonova oblika interpolacijskega polinoma sta uporabni za interpolacijo funkcij, ki so podane v poljubnih točkah, ki niso nujno enako oddaljene med seboj. Pogosto pa imamo opravka z interpolacijo, pri kateri so interpolacijske točke med seboj enako oddaljene. V tem primeru lahko računanje interpolacijskih polinomov nekoliko poenostavimo, kar si bomo ogledali v tem razdelku.

Neodvisna spremenljivka naj bo na intervalu $[a, b]$ podana v točkah

$$x_i = a + ih, \quad i = 0, 1, \ldots, N, \quad N = \frac{b-a}{h}.$$

Vpeljimo novo neodvisno spremenljivko

$$s = \frac{x-a}{h}; \quad \text{da je} \quad x = a + hs. \tag{4.13}$$

S tako zamenjavo postanejo interpolacijske točke $s = 0, 1, \ldots, N$. Posebej moramo poudariti, da je zamenjava spremenljivke (4.13) linearna, zato polinom stopnje n tudi v novi spremenljivki ostane polinom iste stopnje. Zaradi enostavnosti bomo funkcijske vrednosti v interpolacijskih točkah pisali kot $f_i = f(x_i) = f(a + ih)$.

Začnimo z Lagrangeovo obliko interpolacijskega polinoma. Brez težav vidimo, da lahko Lagrangeove polinome (4.4) zapišemo kot

$$l_i(s) = \prod_{\substack{j=0 \\ j \neq i}}^{n} \frac{s-j}{i-j},$$

interpolacijski polinom pa kot njihovo linearno kombinacijo

$$p(s) = \sum_{i=0}^{n} f_i l_i(s).$$

Le malo dalj se bomo pomudili pri Newtonovi obliki interpolacijskega polinoma. Pri ekvidistantnih interpolacijskih točkah bomo namesto tabele deljenih diferenc izračunali raje *tabelo direktnih diferenc*. V ta namen definirajmo *direktne diference*

$$\Delta^i f_j = \begin{cases} f_j; & i = 0 \\ \Delta(\Delta^{i-1} f_j) = \Delta^{i-1} f_{j+1} - \Delta^{i-1} f_j; & i > 0. \end{cases} \tag{4.14}$$

Lahko se je prepričati, da velja med deljenimi in direktnimi diferencami zveza

$$f[x_k, \ldots, x_{k+i}] = \frac{\Delta^i f_k}{i! h^i}. \tag{4.15}$$

Tako lahko interpolacijski polinom skozi točke $x_0, \ldots, x_n$ v Newtonovi obliki (4.7) zapišemo kot

$$p_n(x) = \sum_{i=0}^{n} \frac{\Delta^i f_0}{i! h^i} \prod_{j=0}^{i-1} (x - x_j).$$

Če upoštevamo, da je

$$x - x_k = a + sh - (a + kh) = (s - k)h,$$

lahko zapišemo

$$p_n(x) = p_n(a + sh) = \sum_{i=0}^{n} \Delta^i f_0 \prod_{j=0}^{i-1} \frac{s - j}{j + 1}. \tag{4.16}$$

Formulo (4.16) lahko še poenostavimo, če uporabimo *binomsko formulo*

$$\binom{y}{i} = \begin{cases} 1; & i = 0 \\ \displaystyle\prod_{j=0}^{i-1} \frac{y - j}{j + 1}; & i > 0. \end{cases} \tag{4.17}$$

Končno lahko zapišemo Newtonovo obliko interpolacijskega polinoma skozi ekvidistantne točke kot

$$p_n(a + sh) = \sum_{i=0}^{n} \Delta^i f_0 \binom{s}{i}. \tag{4.18}$$

4.5 Napaka polinomske interpolacije

Do sedaj smo gledali na vrednosti $f_0, \ldots, f_n$ kot poljubna števila. Za analizo napake interpolacijskega polinoma pa je smiselno privzeti, da obstaja taka funkcija f, definirana na intervalu, ki vsebuje vse interpolacijske točke, ki zadošča pogojem

$$f(x_i) = f_i; \quad i = 0, 1, \ldots, n$$

in je dovoljkrat odvedljiva za naše potrebe.

Naj bo p_n interpolacijski polinom skozi točke $x_0, \ldots, x_n$. Ker navadno računamo interpolacijski polinom v upanju, da se bo dobro ujemal s funkcijo f, se je smiselno vprašati, kakšna je napaka interpolacijskega polinoma

$$e(t) = p_n(t) - f(t). \tag{4.19}$$

Seveda je v tej enačbi smiselno izbrati vrednost spremenljivke t različno od točk $x_0, \ldots, x_n$, saj vemo, da napake v interpolacijskih točkah ni.

Naj bo $p_n(t)$ interpolacijski polinom skozi točke $x_0, \ldots, x_n$. Da bi našli izraz za napako interpolacijskega polinoma (4.19), naj bo $q_{n+1}(u)$ interpolacijski polinom skozi točke $x_0, \ldots, x_n$ in t, ki ga v Newtonovi obliki lahko zapišemo kot

$$q_{n+1}(u) = p_n(u) + f[x_0, \ldots, x_n, t]\omega(u), \tag{4.20}$$

kjer smo z $\omega(u)$ označili produkt

$$\omega(u) = (u - x_0) \cdot \cdots \cdot (u - x_n).$$

Ker q_{n+1} interpolira f tudi v točki t, je $q_{n+1}(t) = f(t)$, zato iz (4.20) dobimo

$$f(t) = p_n(t) + f[x_0, \ldots, x_n, t]\omega(t),$$

oziroma, če preuredimo

$$e(t) = p_n(t) - f(t) = -f[x_0, \ldots, x_n, t]\omega(t), \tag{4.21}$$

to pa je že izraz za napako, ki smo ga iskali. Njegova slabost je v tem, da z njim ne moremo dobiti uporabne ocene za napako, saj ne poznamo deljene diference $f[x_0, \ldots, x_n, t]$. Da bi dobili uporabnejšo oceno, si oglejmo funkcijo

$$\varphi(u) = f(u) - p_n(u) - f[x_0, \ldots, x_n, t]\omega(u). \tag{4.22}$$

Njena vrednost v interpolacijskih točkah x_i; $i = 0, \ldots, n$ je enaka 0:

$$\varphi(x_i) = f(x_i) - p_n(x_i) - f[x_0, \ldots, x_n, t]\omega(x_i) = 0,$$

pa tudi

$$\varphi(t) = f(t) - p_n(t) - f[x_0, \ldots, x_n, t]\omega(t) = 0.$$

Če je I najmanjši interval, ki vsebuje vse točke x_i; $i = 0, \ldots, n$ in točko t, ima φ na I najmanj $n+2$ ničli. Ker smo privzeli, da je funkcija f odvedljiva

tolikokrat, kot potrebujemo, p_n in ω pa sta polinoma, torej oba neskončnokrat odvedljiva, lahko sklepamo takole: po Rollejevem izreku ima odvod φ' ničlo vedno med dvema zaporednima ničlama funkcije φ, torej ima φ' na I vsaj $n+1$ ničlo. Podobno ima φ'' najmanj n ničel na I. Če tako nadaljujemo, vidimo, da mora imeti $n+1$-vi odvod $\varphi^{(n+1)}$ vsaj eno ničlo na I. Naj bo torej ξ ena izmed ničel funkcije $\varphi^{(n+1)}$ na intervalu I.

Ker je p_n polinom stopnje n, je njegov $n+1$-vi odvod enak 0, ω pa je polinom stopnje $n+1$ z vodilnim členom 1, torej $\omega(u) = u^{n+1} + \cdots$, zato je

$$\omega^{(n+1)}(\xi) = (n+1)!$$

Če to vstavimo v enačbo (4.22), dobimo

$$0 = \varphi^{(n+1)}(\xi) = f^{(n+1)}(\xi) - f[x_0, \ldots, x_n, t](n+1)!,$$

oziroma, po preureditvi

$$f[x_0, \ldots, x_n, t] = \frac{f^{(n+1)}(\xi)}{(n+1)!}. \tag{4.23}$$

Ko to vstavimo v enačbo (4.21), dobimo naslednji rezultat:

Izrek 4.5.1. *Naj bo funkcija f vsaj $n+1$-krat zvezno odvedljiva in naj bo p_n interpolacijski polinom stopnje $\leq n$ skozi točke x_i; $i = 0, \ldots, n$. Potem je*

$$f(t) - p_n(t) = \frac{f^{(n+1)}(\xi)}{(n+1)!}(t - x_0) \cdots (t - x_n), \tag{4.24}$$

kjer ξ leži v najmanjšem intervalu, ki vsebuje točke x_i; $i = 0, \ldots, n$ in t.

QED

Za trenutek se ustavimo še ob enačbi (4.23), iz katere razberemo, da je n-ta deljena diferenca sorazmerna n-temu odvodu v neki točki na intervalu, ki vsebuje vse točke, ki smo jih uporabili za računanje deljene diference. Če točke x_i; $i = 0, \ldots, n$ izberemo dovolj blizu točke t, lahko iz n-te deljene diference dobimo dober približek za n-ti odvod funkcije v točki t.

Kako torej lahko ocenimo napako interpolacije? Oglejmo si to na primeru:

Primer 4.5.1. Naj bo $p_1(t)$ linearni polinom, ki interpolira funkcijo f v točkah x_0 in x_1 (naj bo $x_0 < x_1$). Funkcija f naj ima omejen drugi odvod, kar pomeni, da obstaja število M, da je

$$|f''(t)| < M$$

na nekem intervalu, na katerem nas interpolacija zanima, in ki vsebuje točki x_0 in x_1. Ocena napake interpolacije je odvisna od tega, ali točka t leži na intervalu $[x_0, x_1]$ ali zunaj njega.

1. Kadar je $t \in [x_0, x_1]$, pravimo, da gre za *interpolacijo v ožjem smislu*. Iz ocene (4.24) dobimo

$$|f(t) - p_1(t)| = \frac{|f''(\xi)|}{2}|(t - x_0)(t - x_1)| \leq \frac{M}{2}|(t - x_0)(t - x_1)|.$$

   Ker funkcija $|(t - x_0)(t - x_1)|$ doseže v točki $(x_0 + x_1)/2$ maksimalno vrednost $(x_1 - x_0)^2/4$, lahko napako interpolacije v tem primeru enakomerno omejimo z

$$|f(t) - p_1(t)| \leq \frac{M}{8}(x_1 - x_0)^2 \quad \text{za vsak} \quad x \in [x_0, x_1]. \tag{4.25}$$

   Vzemimo, da želimo iz tabeliranih vrednosti izračunati vrednost funkcije sinus z natančnostjo pod 10^{-4}. Kakšen je lahko največ *korak tabele*, da bo zadoščala že linearna interpolacija?

   Ker je absolutna vrednost drugege odvoda $|\sin'' t| = |\sin t| \leq 1$, je ocena (4.25) v tem primeru

$$|\sin t - p_1(t)| \leq \frac{h^2}{8},$$

   kar je manj od 10^{-4}, kadar je

$$h \leq \sqrt{8} \cdot 10^{-2} \approx 0.0283.$$

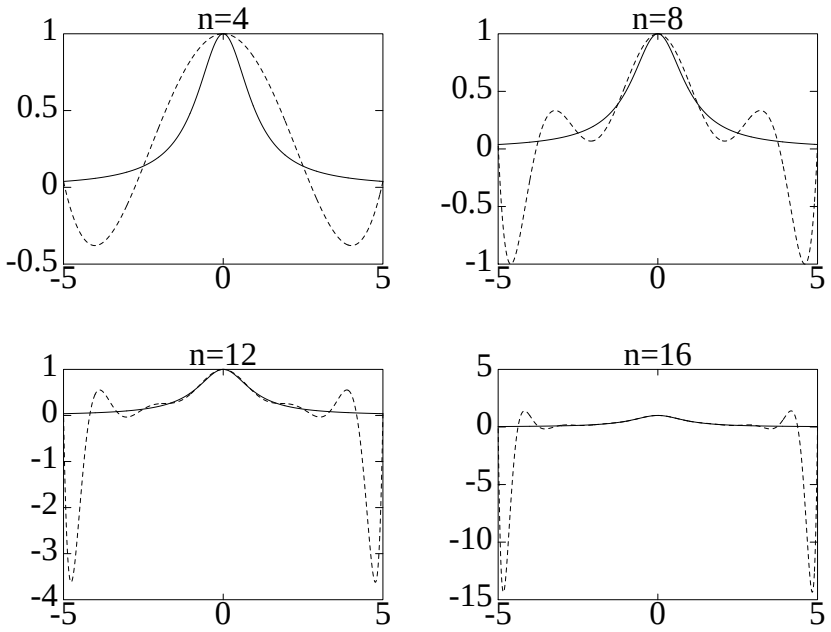
   Da bi za navedeno natančnost zadoščala linearna interpolacija, mora biti korak tabele manjši kot 0.0283.

2. Kadar pa $t \notin [x_0, x_1]$, pravimo, da gre za *ekstrapolacijo*. Ker vrednost izraza $|(t-x_0)(t-x_1)|$ hitro in neomejeno narašča, ko se t oddaljuje od intervala $[x_0, x_1]$, je ekstrapolacija velikokrat zelo tvegana in potrebna je dodatna pazljivost pri interpretaciji rezultatov.

■

V prejšnjem primeru smo na primeru sinusne funkcije videli, da morajo biti vrednosti funkcije, ki jo interpoliramo, tabelirane dovolj na gosto, da za predpisano natančnost zadostuje linearna interpolacija na majnih intervalih. Druga možnost pa je, da uporabimo na celem intervalu, ki nas zanima, isti interpolacijski polinom visoke stopnje. Na naslednjem primeru bomo videli, da je to navadno slabša rešitev.

Primer 4.5.2. Funkcijo



Slika 4.4: Interpolacija v ekvidistantnih točkah

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

interpolirajmo na intervalu $[-5, 5]$ zaporedoma z interpolacijskimi polinomi skozi 4,8,12 in 16 točk, ki naj bodo enakomerno razporejene po intervalu. Iz

slike 4.4 vidimo, kako napaka na robu intervala narašča z naraščanjem števila interpolacijskih točk. ■

4.6 Metoda najmanjših kvadratov

Funkcija f naj bo podana s tabelo vrednosti v n med seboj različnih točkah $f_i = f(x_i)$, $i = 1, \ldots, n$. Iščemo tak polinom p_k stopnje ne večje od $k < n$, za katerega ima izraz

$$E_{LSQ} = \sqrt{\sum_{i=1}^{n} (p_k(x_i) - f_i)^2}$$

najmanjšo vrednost. Ker je E_{LSQ} odvisen od koeficientov polinoma p_k

$$E_{LSQ}(a_0, \ldots, a_k) = \sqrt{\sum_{i=1}^{n} (a_0 + a_1 x_i + \cdots + a_k x_i^k - f_i)^2}, \tag{4.26}$$

imamo pred seboj problem iskanja prostega ekstrema funkcije več spremenljivk. Potreben pogoj za nastop ekstrema je, da so vsi parcialni odvodi $\partial E^2_{LSQ}/\partial a_i$ enaki 0, od koder dobimo sistem linearnih enačb (imenujemo ga *normalni* sistem enačb)

$$\begin{array}{lcllcl}
a_0 n & + & a_1 \sum_{i=1}^n x_i & + \cdots + & a_k \sum_{i=1}^n x_i^k & = \sum_{i=1}^n f_i \\
a_0 \sum_{i=1}^n x_i & + & a_1 \sum_{i=1}^n x_i^2 & + \cdots + & a_k \sum_{i=1}^n x_i^{k+1} & = \sum_{i=1}^n f_i x_i \\
\vdots & & \vdots & & \vdots & \vdots \\
a_0 \sum_{i=1}^n x_i^k & + & a_1 \sum_{i=1}^n x_i^{k+1} & + \cdots + & a_k \sum_{i=1}^n x_i^{2k} & = \sum_{i=1}^n f_i x_i^k,
\end{array} \tag{4.27}$$

ki ga načeloma lahko rešimo z Gaussovo metodo (poglavje 2), vendar je pri velikem številu parametrov (polinom visoke stopnje) ta sistem lahko zelo slabo pogojen.

Zapišimo algoritem, s katerim lahko izračunamo aproksimacijski polinom z metodo najmanjših kvadratov:

Algoritem 4.6.1 (Metoda najmanjših kvadratov)**.** Naj bo funkcija f podana v obliki tabele

$$\begin{array}{rl}
x & : x_1, \quad x_2, \quad \ldots, \quad x_n \\
f(x) & : f_1, \quad f_2, \quad \ldots, \quad f_n.
\end{array}$$

Naslednji algoritem izračuna koeficiente polinoma p_k stopnje $< k$, ki aproksimira funkcijo f z metodo najmanjših kvadratov:

for $i = 1 : 2 * k + 1$
    $y(i) = \mathbf{sum}(x.\hat{}(i-1))$
end
for $i = 1 : k + 1$
    for $j = 1 : k + 1$
        $A(i,j) = y(i+j-1)$ % matrika normalnega sistema
    end
        $b(i) = \mathbf{sum}(f. * x.\hat{}(i-1))$ % desne strani
end
$p = A\backslash b$ % rešitve normalnega sistema

Primer 4.6.1. Poišči linearno funkcijo, ki v smislu najmanjših kvadratov najbolje aproksimira funkcijo f, dano s tabelo

x_i	1.0	2.0	3.0	4.0	5.0	6.0	7.0	8.0	9.0	10.0
f_i	5.5	7.0	12.5	13.0	17.0	19.5	24.5	26.0	27.5	32.5

Najprej izračunamo koeficiente sistema linearnih enačb

$$\begin{aligned} a_{11} &= n &&= 10 \\ a_{12} = a_{21} &= \textstyle\sum_{i=1}^{10} x_i &&= 55 \\ a_{22} &= \textstyle\sum_{i=1}^{10} x_i^2 &&= 385, \end{aligned}$$

in desne strani

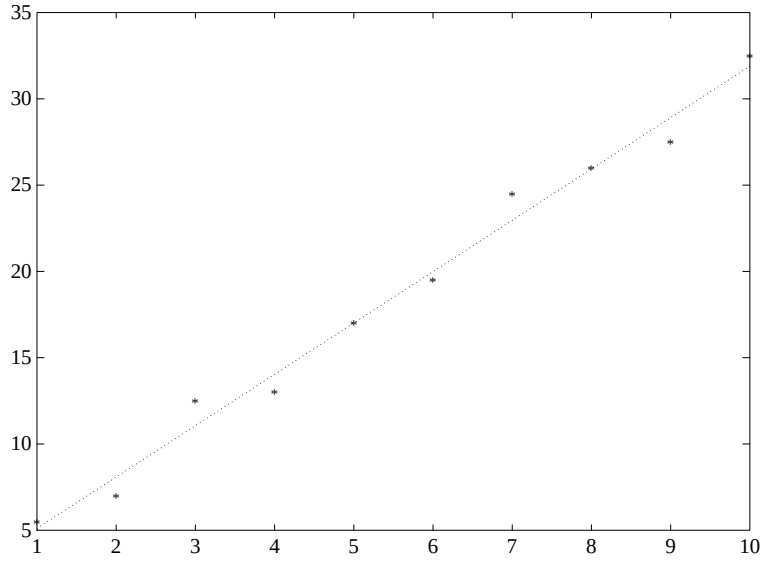
$$\begin{aligned} b_1 &= \textstyle\sum_{i=1}^{10} f_i &&= 185 \\ b_2 &= \textstyle\sum_{i=1}^{10} x_i f_i &&= 1263. \end{aligned}$$

Normalni sistem enačb

$$\begin{aligned} 10.0a + \quad 55.0b &= 185.0 \\ 55.0a + \quad 385.0b &= 1263.0 \end{aligned}$$

ima rešitev (na tri mesta)

$$a = 2.13 \quad \text{in} \quad b = 2.98,$$



Slika 4.5: Linearna aproksimacija

tako, da je linearna aproksimacija po metodi najmanjših kvadratov enaka

$$p(x) = 2.98x + 2.98.$$

Napaka linearne aproksimacije je v tem primeru $E_{LSQ} \approx 3.07$. (glej sliko 4.5.) ■

Podobno kot med polinomi, ki so linearne kombinacije potenc x^i; $i = 0, \ldots, k$, lahko poiščemo aproksimacijsko funkcijo med linearnimi kombinacijami poljubnih linearno nedvisnih funkcij $g_i(x)$; $i = 0, \ldots, k$. V tem primeru je matrika normalnega sistema enačb enaka

$$\begin{bmatrix} \sum_{i=1}^n g_0^2(x_i) & \sum_{i=1}^n g_0(x_i)g_1(x_i) & \cdots & \sum_{i=1}^n g_0(x_i)g_k(x_i) \\ \sum_{i=1}^n g_1(x_i)g_0(x_i) & \sum_{i=1}^k g_1^2(x_i) & \cdots & \sum_{i=1}^n g_1(x_i)g_k(x_i) \\ \vdots & \vdots & \vdots & \\ \sum_{i=1}^n g_k(x_i)g_0(x_i) & \sum_{i=1}^n g_k(x_i)g_1(x_i) & \cdots & \sum_{i=1}^n g_k^2(x_i) \end{bmatrix}, \quad (4.28)$$

desne strani pa

$$\left[\sum_{i=1}^{n} f_i g_0(x_i), \sum_{i=1}^{n} f_i g_1(x_i), \dots \sum_{i=1}^{n} f_i g_k(x_i)\right]^T. \tag{4.29}$$

Sistem linearnih enačb je posebno enostaven, kadar je njegova matrika diagonalna:

Definicija 4.6.1. *Zaporedje funkcij* $(g_j(x))$; $j = 0, \dots$ *je* ortogonalno na sistemu točk x_i; $i = 1, \dots, n$, *če je*

$$\sum_{i=1}^{n} g_j(x_i) g_k(x_i) \begin{cases} = 0 & \text{kadar} \quad j \neq k \\ \neq 0 & \text{kadar} \quad j = k, \end{cases} \tag{4.30}$$

V primeru, ko iščemo aproksimacijsko funkcijo $g(x)$ kot linearno kombinacijo funkcij $g_j(x)$; $j = 0, \dots, k$, ki so ortogonalne na sistemu točk x_i; $i = 1, \dots, n$, je

$$g(x) = \sum_{j=0}^{k} c_j g_j(x), \tag{4.31}$$

kjer so koeficienti enaki

$$c_j = \frac{\sum_{i=1}^{n} f_i g_j(x_i)}{\sum_{i=1}^{n} g_j^2(x_i)}. \tag{4.32}$$

4.7 Ortogonalni polinomi nad sistemom točk

V (4.26) smo aproksimacijski polinom zapisali kot linearno kombinacijo potenc x^j in smo dobili normalni sistem enačb v obliki (4.27). Če pa aproksimacijski polinom zapišemo kot linearno kombinacijo polinomov $t_j(x)$; $j = 0, \dots, k$ stopnje j, ki so ortogonalni na sistemu med seboj različnih točk x_i; $i = 1, \dots, n$, dobimo aproksimacijsko funkcijo

$$p(x) = \sum_{j=0}^{k} c_j t_j(x); \quad c_j = \frac{\sum_{i=1}^{n} f_i t_j(x_i)}{\sum_{i=1}^{n} t_j^2(x_i)}. \tag{4.33}$$

Ortogonalni polinomi na sistemu točk imajo veliko zanimivih lastnosti, tukaj bomo potrebovali le dve. Obe sta podobni lastnostim "klasičnih" ortogonalnih polinomov.

Izrek 4.7.1. *Vsak polinom t_m iz ortogonalnega zaporedja polinomov je ortogonalen na vse polinome nižje stopnje*

$$\sum_{i=1}^{n} t_m(x_i)p_j(x_i) = 0 \quad za \quad j < m.$$

Dokaz: Ker lahko poljuben polinom $p_m(x)$ zapišemo kot linearno kombinacijo (4.33) polinomov iz zaporedja ortogonalnih polinomov, je

$$\begin{aligned}\sum_{i=1}^{n} t_m(x_i)p_j(x_i) &= \sum_{i=1}^{n} t_m(x_i)\left(\sum_{k=0}^{j} d_k t_k(x_i)\right) \\ &= \sum_{k=0}^{j} d_k \sum_{i=1}^{n} t_m(x_i)t_k(x_i) = 0.\end{aligned}$$

QED

Druga lastnost ortogonalnih polinomov pa nam bo pomagala, da bomo zaporedje ortogonalnih polinomov lahko enostavno konstruirali. Tako spotoma pokažemo tudi, da za vsako množico med seboj različnih točk obstaja zaporedje ortogonalnih polinomov. Pogoj (4.30) sicer tega zaporedja še ne določa enolično, saj lahko vsakega izmed polinomov pomnožimo s poljubno konstanto. Če pa zahtevamo poleg tega še, naj imajo vsi polinomi vodilni koeficient enak 1 (tako imenovani *moničnі polinomi*), pa je zaporedje ortogonalnih polinomov na množici točk enolično določeno.

Izrek 4.7.2. *Zaporedje moničnih ortogonalnih polinomov zadošča* tričlenski rekurzivni relaciji

$$t_{i+1}(x) = (x - \alpha_{i+1})t_i(x) - \beta_i t_{i-1}(x), \tag{4.34}$$

kjer so koeficienti α_i in β_i določeni z

$$\alpha_i = \frac{\sum_{j=1}^{n} x_j t_{i-1}^2(x_j)}{\sum_{j=1}^{n} t_{i-1}^2(x_j)} \tag{4.35}$$

$$\beta_0 = 0$$

$$\beta_i = \frac{\sum_{j=1}^{n} t_i^2(x_j)}{\sum_{j=1}^{n} t_{i-1}^2(x_j)}, \tag{4.36}$$

prvi člen zaporedja pa je $t_0(x) = 1$.

Dokaz: Najprej iz (4.34) izračunamo (z upoštevanjem (4.35))

$$t_1(x) = x - \frac{\sum_{j=1}^{n} x_j}{n}. \tag{4.37}$$

Polinom t_1 je ortogonalen na $t_0 = 1$, saj je

$$\sum_{i=1}^{n} \left(x_i - \frac{\sum_{j=1}^{n} x_j}{n} \right) = 0.$$

Denimo, da smo že konstruirali polinome t_k, $k = 0, \ldots, i$, ki sestavljajo zaporedje ortogonalnih polinomov na naši množici točk, torej zanje veljajo pogoji (4.30). Relacijo (4.34) pomnožimo s $t_i(x)$, vstavimo za x po vrsti vse točke x_j in dobljene enačbe seštejemo. Ko upoštevamo, da je t_i ortogonalen na t_{i-1}, in da mora biti tudi t_{i+1} ortogonalen na t_i, dobimo za α_{i+1}

$$\alpha_{i+1} = \frac{\sum_{j=1}^{n} x_j t_i^2(x_j)}{\sum_{j=1}^{n} t_i^2(x_j)},$$

kar dokazuje veljavnost enačbe (4.35).

Če na podoben način relacijo (4.34) pomnožimo s $t_{i-1}(x)$, za x vstavimo po vrsti vse točke x_j in seštejemo dobljene enačbe, nato pa upoštevamo, da je t_i ortogonalen na t_{i-1} in do mora biti tudi t_{i+1} ortogonalen na t_{i-1}, dobimo

$$\beta_i = \frac{\sum_{j=1}^{n} x_j t_i(x_j) t_{i-1}(x_j)}{\sum_{j=1}^{n} t_{i-1}^2(x_j)}.$$

Števec tega izraza lahko poenostavimo, če relacijo

$$t_i(x) = (x - \alpha_i) t_{i-1}(x) - \beta_{i-1} t_{i-2}(x)$$

pomnožimo s $t_i(x)$, vstavimo za x zaporedoma vse x_j in dobljene izraze seštejemo. Tako dobimo, da je

$$\sum_{j=1}^{n} x_j t_i(x_j) t_{i-1}(x_j) = \sum_{j=1}^{n} t_i^2(x_j),$$

kar pomeni, da lahko koeficiente β_i izračunamo po formuli (4.36).

Tako lahko s tričlensko rekurzivno relacijo (4.34) izračunamo naslednji polinom iz ortogonalnega zaporedja, če poznamo vsaj dva prejšnja polinoma

iz zaporedja. Ker poznamo tudi prva dva polinoma ($t_0(x) = 1$ in $t_1(x)$ iz enačbe (4.37)), lahko izračunamo po indukciji poljuben člen zaporedja ortogonalnih polinomov. QED

Poglejmo si algoritem za izračun koeficientov ortogonalnih polinomov:

Algoritem 4.7.1 (Ortogonalni polinomi)**.** Vektor $(x_i,\ i = 1, \ldots, n)$ naj vsebuje med seboj različne točke. Naslednji algoritem izračuna koeficiente tričlenske rekurzivne relacije (4.34), ki jih potrebujemo, da izračunamo polinome ortogonalnega zaporedja do stopnje k, in vrednosti ortogonalnih polinomov v izbranih točkah:

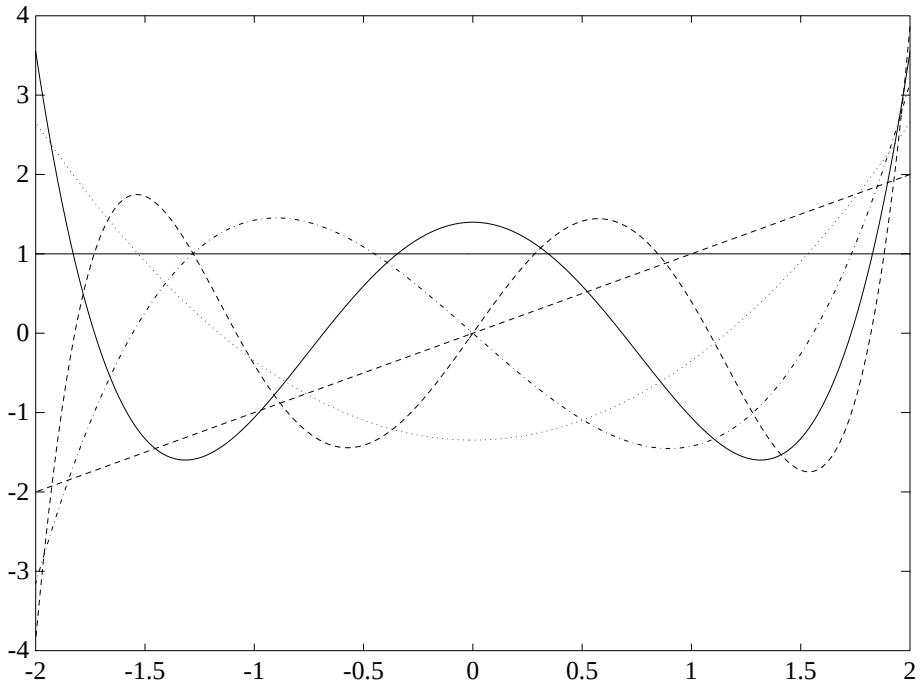
$[m, n] = \mathbf{size}(x)$
$t = \mathbf{zeros}(k, n)$
$a = \mathbf{zeros}(k, 1)$
$b = \mathbf{zeros}(k, 1)$
$t(1, :) = \mathbf{ones}(1, n)$
$t(2, :) = x - \mathbf{sum}(x)/n$
for $i = 2 : k$
    $a(i + 1) = (x * (t(i, :). * t(i, :))')/(t(i, :) * t(i, :)')'$
    $b(i) = t(i, :) * t(i, :)'/(t(i - 1, :) * t(i - 1, :)')$
    $t(i + 1, :) = (x(:) - a(i + 1)). * t(i, :) - b(i) * t(i - 1, :)$
end

Primer 4.7.1. Izračunajmo prvih 8 polinomov, ortogonalnih na množici točk $x_i = a + ih$; $i = 0, \ldots, 20$, kjer je $a = -2$ in $h = 0.2$. Te točke so enakomerno razporejene na intervalu $[-2, 2]$ z razmakom h.

Izračunani koeficienti α_i in β_i rekurzivne formule (4.34) so v tabeli 4.4, grafi prvih petih polinomov pa so na sliki 4.6. ■

Računanje aproksimacijskih polinomov, izraženih s pomočjo ortogonalnih polinomov ima prednosti pred izražavo s potencami:

- Normalni sistem, zapisan z ortogonalnimi polinomi ima diagonalno obliko in je torej enostavno rešljiv, medtem ko je normalni sistem, izražen s potencami, pogosto slabo pogojen.



Slika 4.6: Zaporedje ortogonalnih polinomov

- Kadar z natančnostjo izračunanega aproksimacijskega polinoma stopnje k nismo zadovoljni, mu lahko enostavno dodamo naslednji člen $d_{k+1}t_{k+1}(x)$.

4.8 Aproksimacija periodičnih funkcij

Pri aproksimaciji funkcij, ki opisujejo periodične pojave, je najbolje uporabljati *trigonometrične polinome*

$$p(x) = a_0 + \sum_{i=1}^{m}(a_i \cos ix + b_i \sin ix). \tag{4.38}$$

Vzemimo, da je f periodična funkcija s periodo 2π. Kadar ima funkcija $g(x)$, ki jo aproksimiramo, periodo $\tau \neq 2\pi$, lahko s spremembo neodvisne spremenljivke dosežemo, da ima funkcija $f(x) = g(\frac{\tau x}{2\pi})$ periodo 2π.

i	α_i	β_i
1		1.3467
2	$0.13398 \cdot 10^{-15}$	1.0772
3	$-0.21107 \cdot 10^{-15}$	1.0387
4	$0.30531 \cdot 10^{-15}$	1.0257
5	$-0.08703 \cdot 10^{-15}$	1.0196
6	$-0.06727 \cdot 10^{-15}$	1.0162
7	$-0.09497 \cdot 10^{-15}$	1.0140
8	$-0.16334 \cdot 10^{-15}$	1.0124
9	$-0.01955 \cdot 10^{-15}$	

Tabela 4.4: Koeficienti α in β rekurzivne formule za izračun zaporedja ortogonalnih polinomov na množici točk $x_i = -2 + 0.2i$; $i = 0, \ldots, 20$

Obravnavali bomo le primer, ko imamo $2n$ aproksimacijskih točk

$$x_j = \frac{j\pi}{n}, \quad j = -n, -n+1, \ldots, n-1 \tag{4.39}$$

razporejenih ekvidistantno na intervalu $[-\pi, \pi]$.

Pri aproksimaciji s trigonometričnim polinomom izkoristimo dejstvo, da tvorijo funkcije

$$1, \cos x, \sin x, \cos 2x, \sin 2x, \ldots, \cos mx, \sin mx \tag{4.40}$$

zaporedje ortogonalnih funkcij na množici točk (4.39), saj veljajo relacije

$$\begin{aligned}
\sum_{j=-n}^{n-1} \sin mx_j \cos kx_j &= 0, \\
\sum_{j=-n}^{n-1} \cos mx_j \cos kx_j &= \begin{cases} 0 & m \neq k \\ n & m = k \neq 0 \\ 2n & m = k = 0, \end{cases} \qquad (4.41) \\
\sum_{j=-n}^{n-1} \sin mx_j \sin kx_j &= \begin{cases} n & m = k \neq 0 \\ 0 & \text{sicer} \end{cases} \qquad (4.42)
\end{aligned}$$

za vse $m, k = 0, 1, \ldots, n$. O njihovi veljavnosti se lahko prepričamo, če

zapišemo vsoto geometrijskega zaporedja

$$\sum_{j=-n}^{n-1} e^{ikx_j} = \sum_{j=-n}^{n-1} e^{ik\pi j/n} = e^{-ik\pi}\frac{e^{2ik\pi}-1}{e^{ik\pi/n}-1},$$

ki je enaka 0 za $k = 1, \ldots, 2n-1$ in enaka $2n$ za $k = 0$. Od tod dobimo

$$\sum_{j=-n}^{n-1} \cos kx_j = \sum_{j=-n}^{n-1} \sin kx_j = 0, \quad k = 1, \ldots, 2n-1,$$

relacije (4.41) pa lahko izpeljemo s pomočjo adicijskih izrekov za sinus in cosinus.

Koeficienti trigonometričnega polinoma (4.38), ki najbolje aproksimira dano funkcijo f po metodi najmanjših kvadratov so torej enaki

$$a_0 = \frac{1}{2n}\sum_{j=-n}^{n-1} f(x_j) \tag{4.43}$$

$$a_k = \frac{1}{n}\sum_{j=-n}^{n-1} f(x_j)\cos kx_j \tag{4.44}$$

$$b_k = \frac{1}{n}\sum_{j=-n}^{n-1} f(x_j)\sin kx_j \tag{4.45}$$

Formule (4.43–4.45) so podobne formulam za koeficiente Fourierove vrste. V poglavju o numerični integraciji bomo spoznali, da so (4.43–4.45) približki za Fourierove koeficiente, izračunani s trapeznim pravilom.

4.9 Povzetek

Interpolacija je postopek, pri katerem dano funkcijo nadomestimo z neko drugo, ki se s prvotno ujema v nekaj izbranih točkah. Nekoč so interpolacijo uporabljali predvsem za računanje vmesnih vrednosti tabeliranih funkcij (logaritemske tabele). Danes to nalogo opravljajo elektronski kalkulatorji in računalniki, pri tem pa si še vedno marsikdaj pomagajo z interpolacijo.

Drugo področje, kjer uporabljamo interpolacijo, in ki ga bomo srečali v naslednjih poglavjih, pa so formule za numerično odvajanje in integriranje.

V tem poglavju smo spoznali dva osnovna načina za računanje interpolacijskih polinomov, Lagrangeovo in Newtonovo interpolacijsko formulo. Lagrangeova formula je nekoliko preprostejša za uporabo, kadar želimo izračunati vrednosti interpolacijskega polinoma z vnaprej predpisano stopnjo. Kadar pa stopnje interpolacijskega polinoma ne poznamo vnaprej, je primernejša Newtonova oblika.

Poseben problem je zagotavljanje natančnosti interpolacijskega polinoma. Iz ocene (4.24) vidimo, da je napaka manjša, če so točke bliže skupaj ali pa če je stopnja interpolacije dovolj velika in pri tem višji odvodi ostajajo omejeni. Ker je v praksi le redko mogoče oceniti vrednost višjih odvodov, je varneje uporabljati interpolacijske polinome nizkih stopenj, zato pa morajo biti interpolacijske točke gostejše. Prav tako se moramo zavedati, da je napaka navadno najmanjša sredi intervala, na katerem ležijo interpolacijske točke in da narašča proti robu intervala. Kadar pa računamo vrednosti interpolacijske funkcije zunaj intervala, na katerem ležijo interpolacijske točke (ekstrapolacija), pa je napaka lahko neomejeno velika.

Večkrat (posebej, kadar imamo veliko število podatkov, ki so lahko nenatančni) je pomembno, da znamo dano funkcijo aproksimirati z neko drugo (enostavnejšo, laže izračunljivo) funkcijo. Podrobneje smo si ogledali aproksimacijo s polinomi po metodi najmanjših kvadratov. Ugotovili smo, da lahko aproksimacijski polinom, v primerjavi z običajno izražavo v obliki linearne kombinacije potenc, preprosteje in učinkoviteje izračunamo v obliki linearne kombinacije polinomov, ki so ortogonalni na množici aproksimacijskih točk.

Spoznali smo tudi, kako lahko izračunamo aproksimacijo periodične funkcije v obliki trigonometrijskega polinoma. Ker so trigonometrične funkcije ortogonalne na ekvidistantni mreži točk, lahko njegove koeficiente preprosto izračunamo.

4.10 Problemi

1. Pokaži, da rešitev normalnega sistema enačb (4.28) z desnimi stranmi (4.29) določa koeficiente c_i funkcije

$$F(x) = \sum_{i=0}^{k} c_i g_i(x),$$

ki dano funkcijo f aproksimira po metodi najmanjših kvadratov,

Navodilo: Poišči ekstrem funkcije

$$E(c_0, \ldots, c_k) = \sqrt{\sum_{j=1}^{n} (F(x_j) - f(x_j))^2}.$$

2. Funkcije $g_i(x)$; $i = 0, \ldots, k$ naj bodo ortogonalne na sistemu točk x_j; $j = 1, \ldots, n$. Prepričaj se, da funkcija (4.31) s koeficienti (4.32) res aproksimira dano funkcijo f v smislu najmanjših kvadratov.

3. V tabeli

x	0	1	2	3	4
$f(x)$	6	4	20	108	370

je tabelirana vrednost nekega polinoma 4. stopnje.

   (a) Zapiši algoritem za izračun vrednosti polinoma v poljubni točki, ki je ni v tabeli.

   (b) Sestavi tabelo deljenih diferenc.

   (c) Izračunaj vrednost $f(5)$ in $f(-1)$.

   (d) Zapiši polinom v Newtonovi in v normalni obliki.

4. Funkcijo $e^{-x} \sin x$ želimo tabelirati na intervalu $[0, 2]$, da bomo tabelirane vrednosti uporabili pri računanju vmesnih vrednosti z interpolacijo.

   (a) Kolikšen naj bo korak tabele, da bo napaka kubične interpolacije pod 10^{-5}?

(b) Napiši algoritem, ki iz tabeliranih vrednosti izračuna kubični interpolacijski polinom.

(c) Kolikšna mora biti stopnja interpolacijskega polinoma, da bo napaka pod 10^{-6}, če je korak tabele 0.02?

5. Iz tabele za vrednosti funkcije sinus dobimo naslednje vrednosti:

x	$\sin x$
2.4	0.6755
2.6	0.5156
2.8	0.3350
3.0	0.1411
3.2	−0.0584
3.4	−0.2555

(a) Sestavi tabelo deljenih diferenc.

(b) Izračunaj približni vrednosti za $\sin 3.1$ in $\sin e$ s pomočjo kubičnega interpolacijskega polinoma.

(c) Oceni napako obeh aproksimacij s pomočjo formule (4.24) in jo primerjaj z dejansko napako.

(d) Kako bi s pomočjo kubičnega interpolacijskega polinoma ugotovil, za kateri x je $\sin x = 0.3$?

6. Funkcija $\sin x$ je tabelirana na intervalu $[0, 1.6]$.

(a) Kolikšna je maksimalna napaka kvadratne interpolacije, če je korak tabele enak $h = 0.01$?

(b) Kolikšen naj bo korak tabele, da bo napaka kvadratne interpolacije pod 10^{-4}?

(c) Napiši algoritem, ki iz tabeliranih vrednosti izračuna kubični interpolacijski polinom.

(d) Kolikšna mora biti stopnja interpolacijskega polinoma, da bo napaka pod 10^{-6}, če je korak tabele 0.02?

7. Iz tabele naravnih logaritmov dobimo naslednje vrednosti

x	$\log x$
1.0	0.0000
1.5	0.4055
2.0	0.6931
3.0	1.0986
3.5	1.2528
4.0	1.3863

(a) Sestavi tabelo deljenih diferenc.

(b) Izračunaj približni vrednosti za $\log 2.5$ in $\log 1.75$ s pomočjo kubične interpolacije.

(c) Oceni napako približkov iz prejšnje točke in primerjaj s točnimi vrednostmi.

8. Pri merjenju hitrosti izstrelka smo dobili naslednje rezultate:

$t[s]$	$h[m]$
1.0	197.2
1.2	393.3
1.4	519.7
1.6	568.0
1.8	528.2
2.0	391.2

(a) Napiši algoritem, ki dane podatke nadomesti s polinomom stopnje n po metodi najmanjših kvadratov.

(b) Po metodi najmanjših kvadratov izračunaj kvadratno aproksimacijo.

(c) Izračunaj največjo višino izstrelka? Kdaj jo doseže? Kdaj bo izstrelek padel na tla ($h = 0$)?

9. Funkcijo f, dano v obliki tabele

x	0.00	0.25	0.50	0.75	1.00
$f(x)$	0.5000	0.7143	1.0000	1.4000	2.0000

želimo aproksimirati s funkcijo oblike $p(x) = a + be^x$.

(a) Zapiši normalni sistem enačb za neznana koeficienta a in b.

(b) Zapiši algoritem, ki izračuna a in b.

(c) Izračunaj a in b.