Digitális korszak a matematikai analízisben?

Peredy József professzor emeritus, BME

1) Mi is az a digitális korszak?

A digitális módszerek lényege az, hogy minden felhasználni kívánt információt (számot, szöveget, képet, hangot) különálló (más szóval diszkrét) számjegyek véges darabszámú sorozataivá alakítunk át, ebben a formában rögzítünk (pl. digitális fényképezőgéppel), tárolunk és dolgozunk fel (pl. digitális számítógépben), végül pedig adunk tovább (pl. digitális vezérlésű gépekbe, kijelzőkbe, vagy más számítógépekbe, az internet segítségével akár nagy távolságokra is). Mindez ma még aligha belátható lehetőségeket nyit az ismeretek, információk nagy tömegeinek az egyes ember és az emberiség jobb életét szolgáló feldolgozására és felhasználására, de nyilván számos új probléma is jelentkezik. Az egész kérdéskört sokan egy új korszak beköszöntének tekintik, lásd például **Eric Schmidt és Jared Cohen**: **„The New Digital Age”**című könyvét**.**

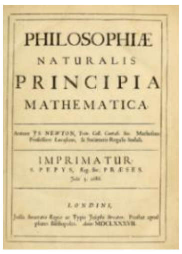
Az első pillanatra úgy tűnhet, hogy a matematika már kezdetétől fogva, azaz a számfogalom létrejötte, a számok megnevezése és különösen azok számjegyekkel való leírása óta már digitális. Ez azonban nincsen így, mivel az idő múlását, a tér kiterjedését viszont, ugyancsak kezdettől fogva, folytonosnak és végeláthatatlannak éreztük (érezzük?), s a matematika tárgykörébe ezek tanulmányozása is beleértődik. A digitális korszak újszerű hangsúlyai azonban az utóbbi időben kiváló matematikusok gondolkodásában is megjelennek:

- "A igazi megértés a diszkrét és a folytonos szemlélet valamiféle szintézisét igényli, az ennek megfelelő matematikai eszközök kidolgozása korunk jelentős intellektuális kihívása." (Lovász László professzortól szabadon idézve).
- "Korunk feladata, hogy a digitális számítógép centrikus matematikára való áttérést a lehetőség szerint zökkenőmentesen és hatékonyan valósítsuk meg" (Doron Zeilberger professzortól szabadon idézve).

Úgy tűnik, hogy ez a kétségtelenül alapvető jelentőségű, de eléggé elvontnak látszó kérdéskör egyszerű gyakorlati feladatokból kiindulva is jól megközelíthető.

2) Egy elemi példa, néhány tanulsággal.

Tekintsünk az egyenes vonalú egyenletes mozgást. Newton klasszikus művében (címlapját



lásd baloldalt) található „Newton 1. Törvénye”, s ez éppen ilyen mozgásra vonatkozik. Tegyük fel, hogy a mozgó test az $t = i_0 = 0$ időpontban az $x = j_0 = 0$ koordinátával jelölt helyzetből indul, és egyenes vonalú egyenletes mozgást végezve i_l időegység elteltével j_l hosszegységnyire távolodik el az induló helyzettől. Végezzünk mérést egy ilyen mozgásra vonatkozóan, más nem lévén, digitális műszerrel, de mondjuk csak igen egyszerű műszer áll rendelkezésre, amelyről (tízes rendszerben) mindössze kétjegyű számokat lehet leolvasni. Példaként tegyük fel, hogy a mérés eredményeként $i_l = 19$ és $j_l = 14$ adódott, s ki szeretnénk számítani, hogy a kezdőponttól milyen j_l hosszegységnyi távolságra volt a test az $i_a = 12$ időegységnek megfelelő időben.

A feladat szokásos megközelítése az, hogy az egyenletes mozgást az $x = m.t$ homogén elsőfokú folytonos függvénnyel írjuk le. Ebben az m együttható értéke

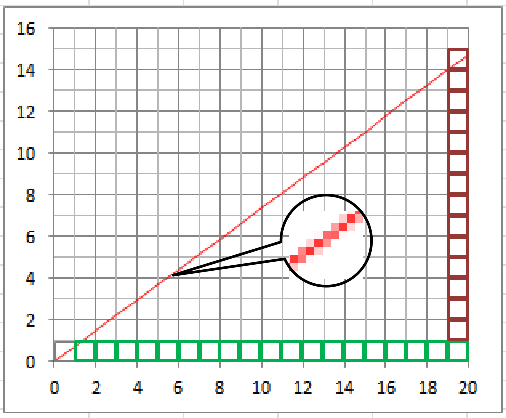
$$m = j_l / i_l = 14/19 = 0{,}736842105263158\ldots$$

valós szám. Az $x = m.t$ függvényt a síkbeli koordináta rendszerben egy egyenes-szakasszal szemléltetjük (lásd az alábbi ábrán a piros vonalat), az egyenletesen mozgó test i_a =12 időegységhez tatozó j_a helyzetét pedig a

$$j_a = m.i_a = 0{,}736842105263158*12= 8{,}84210526315789$$

formában határozzuk meg. Ehhez a számítási módhoz az az elgondolás tartozik, hogy az egyenes egy folytonos vonal, a számértékek pedig valós számok, azaz *végtelen* tizedes törtek, s a függvény minden szóba jövő valós t-hez megad egy valós x-et, azaz a folytonos egyenes egy (t, x) koordinátájú pontját.

Valójában persze az m-et és a j_a-t digitális számítógépünk 15 jegy pontossággal, vagyis számjegyek *véges* sorozatával írta ki, s az ábrán a kinagyítás jól mutatja azt is, hogy az „egyenes” raszter képernyőn való megjelenítése *véges* méretű és darabszámú, jól megkülönböztethető (azaz diszkrét) elemekből van összetéve. A szokásos eljárás tehát tulajdonképpen hibrid: a valós számok és a folytonos függvények körében gondolkodunk, a konkrét számításokat, megjelenítéseket pedig digitális eszközeinkkel csakis diszkrét jelleggel tudjuk végrehajtani.



Megfontolandó az is, hogy míg a kiinduló adatként szolgáló i_l és j_l mérési eredmények kétjegyűek, addig az ezek alapján (legalább is a szokásos, „felületes” módon) elvégzett gépi számítás a keresett j_a-ra 15 jegyű eredményt ad. Az, hogy e visszásság nekünk, a digitális számítógépek mai használóinak már szinte fel sem tűnik, jórészt annak tulajdonítható, hogy az elméletileg valósnak elgondolt számokat számítógépeinkben szinte kizárólagosan „lebegőpontos számokként” jelenítjük meg. Neumann János már kezdettől (1945) kerülte a lebegőpontos számok használatát, mivel ez „programozási szempontból kényelmes ugyan, de eltereli a figyelmet számításaink szabatos matematikai végiggondolásának a szükségességéről”, és jó fél évszázaddal későbben W. Kahan, a Berkeley egyetem professzora szerint a lebegőpontos aritmetika még mindig „tüske a számítógép-tudomány körme alatt” (1997). Kézenfekvő tehát az igény, kíséreljük meg a szóban forgó lineáris összefüggést tudatosan, a digitális számítógépeink alapelvének megfelelően *teljes egészében egységesen diszkrét formában*, tehát kizárólag *egész számokkal* dolgozva kezelni.

3) Van-e a digitális világban diszkrét egyenes?

Az egyenletes mozgás diszkrét leírására irányuló, bevezető jellegű kísérletben alkalmazzuk (a kiinduló adatokhoz igazodva) csak a legfeljebb kétjegyű egész számokat. Ekkor a remélt diszkrét integer egyenes a *(t, x)* koordinátájú valós pontok helyett a mondott egész számok felhasználásával képezhető $[i_k,\ j_k]$, ($k = 0, 1, \ldots\ l$) integer számpárok egy alkalmas véges sorozatából állhat, ezek lehetnek az $x = m.t$ valós függvény digitális társát jelentő „integer függvény" elemei. A folytonosság ideája azonban minden bizonnyal nem „véletlenül" alakult ki az emberi gondolkodás több ezer éves története során, viszont a diszkrét világban a maga elgondolt absztrakt tökéletességében nyilván nem értelmezhető. Alkossák tehát (mintegy „kompromisszumként") az integer egyenest „a folytonossághoz legközelebb álló", azaz szomszédos integer számpárok: vagyis ha $[i_k,\ j_k]$ és $[i_{k+1},\ j_{k+1}]$ ($k = 0, 1, \ldots\ l\text{-}1$) két egymást követő integer számpár, akkor legyen

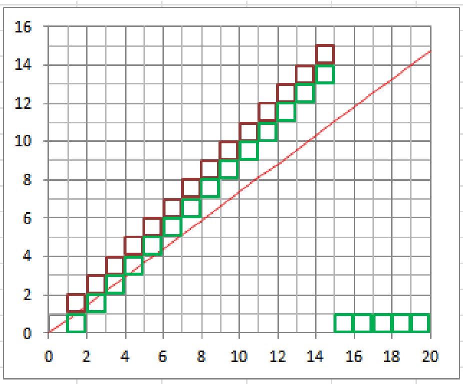
vagy $j_{k+1} = j_k$ és $i_{k+1} = i_k \pm 1$, vagy pedig $i_{k+1} = i_k$ és $j_{k+1} = j_k \pm 1$.

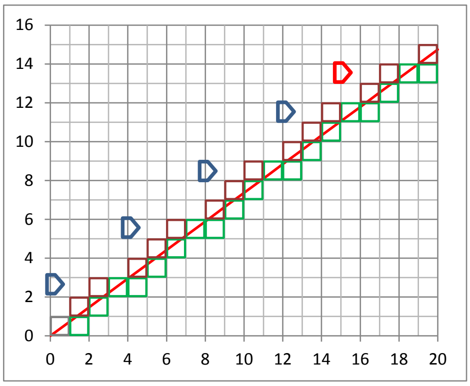
Ezt úgy is mondhatjuk, hogy az $[i_k,\ j_k]$ elemről a következő $[i_{k+1},\ j_{k+1}]$ elemre az első esetben egy egységnyi *i* lépéssel, a másodikban pedig egy egységnyi *j* lépéssel jutunk.

Az $[i_k,\ j_k]$ integer számpárokat ábráink folytonosnak elgondolt koordinátarendszerében a raszterpontok jelentik, mivel azonban a szigorúan vett absztrakt „pontok" magukban amúgy sem tehetők láthatóvá, használjuk ezért (mintegy szimbólumszerűen, a szokásos „nullkörök" vagy „keresztecskék" helyett) az integer függvény elemeinek jól látható megjelenítésre ezúttal azon egységnégyzeteket, amelyek bal alsó sarokpontja a szóban forgó integer számpár.

A példafeladatul választott egyenes vonalú egyenletes mozgás esetén a diszkrét egyenes az $[i_0,\ j_0] = [0,\ 0]$ kezdő integer számpárral indul, és 19 *i* lépés, valamint 14 *j* lépés megtétele után az $[i_{33},\ j_{33}] = [19,\ 14]$ integer számpárral fejeződik be. Az előző oldali ábrán az ennek megfelelő 34 integer párt a mondott egységnégyzetes jelöléssel tüntettük fel, a kezdő számpárt szürke, azon számpárokat, amelyekre *i* lépéssel lépünk zöld, amelyekre pedig *j* lépéssel, lila színnel.

Természetesen az előző oldalon mutatott elrendezés nem lehet a keresett integer egyenes, hiszen az utolsó időegységig nem jelez elmozdulást, a feladatban megkívánt teljes elmozdulást az utolsó időegységben jeleníti meg. Egyenletes mozgás esetén az *i* és *j* lépéseknek nyilván a lehető legegyenletesebben kell eloszlani. Esetünkben *i* lépésből van több, így azt kell megnéznünk, hogy egy *i*-re hány *j* lépés jut. Mivel 19-ből a 14-et egyszer lehet kivonni és még marad 5, a 14 *j* lépés mindegyikéhez legalább egy *i* lépés tartozik („rövid sor") , de 5 esetben kettő („hosszú sor"). Az 5 hosszú sort és a 9 rövid sort egymás között megint „egyenletesen" kell





kiosztani, azonban „teljesen egyenletesen” ezt sem lehet, mivel a 9-ből az 5-öt egyszer lehet kivonni, de még marad 4. Tehát 4 alkalommal *két* rövid sort követ egy hosszú (nagy csoport), 1 alkalommal pedig *egy* rövid és egy hosszú sor áll egymás után (kis csoport). Mivel a 4 nagy és az egy kis csoport éppen kiteszi a megkívánt 19 *i* és 14 *j* lépést, megtaláltuk a lehető legegyenletesebb elosztást, azaz a keresett integer egyenest. Hosszú függvények esetén persze az eljárást általában többször kellhet ismételni. A klasszikus görög matematikában ez a gondolatmenet (váltakozó kivonás megnevezéssel) fontos szerepet játszott a tört számok fogalmának kialakulásánál, és leírása talán az első írásban ránk maradt algoritmus a matematika története során. Most ez a több mint kétezer éves, klasszikus gondolatmenet egy megnyílni látszó új korszakkal („**The New Digital Age”)** kapcsolatban kerül felhasználásra.

A mintapéldában tulajdonképpen az volt a feladat, hogy az egyenletes mozgást végző test 12 időegység elteltéhez tartozó j_a helyzetét határozzuk meg. Integer egyenesünknek, mint az ábrából leolvasható, két olyan integer számpárja van, amely az i_a = 12 időegységhez tartozik, a *[12, 8]* és a *[12, 9]*. Ez nem mond ellent az előző szakaszban számított 12**14/19 = 8,84210526315789,* valós(nak elgondolt) értéknek, de azért igen csak más az információ-tartalma. Nevezetesen azt jelenti, hogy a 8 és 9 hosszegységnyi eltávolodások „táján” járt az adott időben a példa szerinti egyenletesen mozgó test, biztosan nem volt már a 7.-ben, és nem volt még a 10.-ben. Az integer egyenes esetén tehát az eredmény nincs pontosabbnak feltüntetve, mint a kiinduló adat, sőt felhívja a figyelmet arra, hogy a kitűzött feladat eredménye tulajdonképpen két szomszédos integer értékpár bármelyike lehet.

4) Egy másik lehetőség az integer egyenes meghatározására.

A diszkrét számítások alapeszközei az egész számok véges sorozatai, a digitális világban a függvények leírására és vizsgálatára is ezt használhatjuk. Az egyenletes mozgást leíró integer függvény esetén az idő múlását a „hétköznapi fizikában” eleve egyenletesnek tekintjük, a feladat szerint pedig az elmozdulás is egyenletes az időhöz képest. Az egész számok világában az egyenletes változást egyenlőközű sorozatok fejezhetik ki:

RX→*(X, 2X, … i.X, … m.X)* és RY→*(Y, 2Y, … j.Y, … n.Y)*, ahol is X és Y egész számok.

Esetünkben azt tudjuk, hogy a $i_l = 19$ időegység és az ez alatt megtett $j_l = 14$ hosszegységnyi út összetartoznak. Ez azzal fejezhető ki, hogy legyen $i_l.X = j_l.Y$, azaz – legegyszerűbb lehetőségként – legyen $X = 14$ és $Y = 19$. Ezen értékek választása mellett a két sorozat (megfelelő összerendezett formában) kijelöli a kívánt integer egyenes lépéssorrendjét. Az összerendezés részleteit az alábbi táblázat mutatja:

X = 14	**RX→**	0	14		28		42	56		70		84		98	112		126	
Y = 19	**RY→**	0		19		38			57		76		95			114		133
Összerendezve:		0	14	19	28	38	42	56	57	70	76	84	95	98	112	114	126	133
	R	-5	9	-10	4	-15	-1	13	-6	8	-11	3	-16	-2	12	-7	7	-12
	i	0	1	1	2	2	3	4	4	5	5	6	6	7	8	8	9	9
	j	0	0	1	1	2	2	2	3	3	4	4	5	5	5	6	6	7
	k	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16

X = 14	**RX→**	140		154	168		182		196		210	224		238		252	266	
Y = 19	**RY→**		152			171		190		209			228		247			266
Összerendezve:		140	152	154	168	171	182	190	196	209	210	224	228	238	247	252	266	266
	R	2	-17	-3	11	-8	6	-13	1	-18	-4	10	-9	5	-14	0	14	-5
	i	10	10	11	12	12	13	13	14	14	15	16	16	17	17	18	19	19
	j	7	8	8	8	9	9	10	10	11	11	11	12	12	13	13	13	14
	k	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33

Az összerendezés alapja az, hogy az integer egyenest olyan $[i_k, j_k]$ integer számpárok alkossák, amelyek esetében az egymást követő elemek koordinátáihoz tartozó RX_i ill. RY_j sorozatbeli értékek lehetőség szerint közel álljanak egymáshoz. Legyen az $[i_k, j_k]$ integer számpár esetén a hozzá tartozó RX_i ill. RY_j értékek különbsége $R_k=RX_i -RY_j$. Az összerendezést az R sorozat értékeinek segítségével valósítjuk meg: ha R_k pozitív, akkor az integer egyenes *(k+1)*-edik elemére *j* lépéssel, ha nem akkor pedig *i* lépéssel kell lépni. Így az R abszolút értéke sehol nem haladja meg a két pozitív X és Y egész szám közül a nagyobbikat. Az *i* lépés esetén az összerendezett sorozat következő elemét az RX részsorozatból választjuk, a *j* lépésnél pedig az RY-ból (és persze mindkét esetben a következő, még be nem válogatott elemet, amely az RX esetében X-szel, az RY esetében pedig Y-nal különbözik az előzőtől). Így az integer egyenest előállító algoritmus egyetlen összetett feltételes utasításban foglalható össze:

If R ≤ 0 Then R = R + X: i = i + 1 Else R = R – Y: j = j + 1.

Ezt a feltételes utasítást szükség szerint (esetünkbe 33-szor) ismételten végrehajtva (lásd az előző oldali táblázatot) megkapjuk a kívánt integer egyenest. Nem tettünk még említést arról, hogy a *k=0* kezdő oszlopban az R-nek is kell kezdőértéket kapni. Esetünkben az R = X - Y = -5 induló értéket alkalmaztuk, ezzel biztosítva, hogy az integer egyenes *origón átmenő* valós egyenes digitális társa lehessen. Az integer egyenesek számszerű előállításának a fentebb mutatott két lehetséges módja, azaz az összetett feltételes utasítás ismételt működtetése, és a „váltakozó kivonás" módszere, egyenértékű. A „váltakozó kivonás" másként is algoritmizálható, sőt ez az algoritmus jobban is hasonlít a szorzás/osztás megszokott műveletéhez, nem kell az integer függvény valamennyi közbenső elemét sorra előállítani ahhoz, hogy a kezdő elemtől a kívánt *i* vagy *j* koordinátájú elem(ek)ig eljussunk, és megjelenik a iránytangenst megadó „arányszám" digitális alakja is. A feltételes utasítás alapján való előállítás azonban egy ilyen rövid bemutató keretében könnyebben általánosítható, ezért a következőkben ezt fogjuk használni.

5) További elemi feladatok: a szabad esés és a harmonikus rezgés.

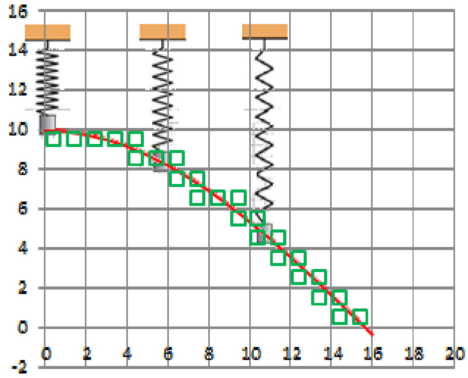
Az egyenes vonalú egyenletes mozgás mellett az egyenletesen gyorsuló mozgás (pl. a szabad esés) és a harmonikus rezgőmozgás is a klasszikus mechanika alapfeladatai közé tartozik. Ha az egyenletes mozgás integer függvényét előállító algoritmus magját alkotó

If R ≤ 0 Then R = R + X: i = i + 1 Else R = R – Y: j = j + 1

összetett feltételes utasítást megnézzük, azt látjuk, hogy változatlan Y mellett annál több *j* hosszlépés esik egy *i* időlépésre, minél nagyobb az X, azaz állandó Y mellett az X jellemzi a mozgás sebességét. Egyenletesen gyorsuló mozgás esetén minden időegységben ugyanannyival nő a sebesség, tehát minden *i* lépésben ugyanannyival kell növekednie az X-nek. Az egyenletesen gyorsuló mozgást leíró integer függvény előállításához tehát az algoritmusmagot ki kell egészíteni az X-nek egy azonos értékű megnövelésével minden *i* lépésben:

If R ≤ 0 Then **X = X + XX**: R = R + X: i = i + 1 Else R = R – Y: j = j + 1.

A sebesség változását, tehát a gyorsulást XX-szel jelöltük. Ez a jelölésválasztás az integer függvényeket előállító jelölésrendszer alapszabályát tükrözi: az utolsó betű azt adja meg, hogy *i* lépés esetén van működésben, mégpedig hozzá adódik (természetesen előjelhelyesen) az utolsó betű elhagyásával nyerhető azonosítójú változóhoz.



Ugyanilyen magától értetődően kapjuk a harmonikus rezgőmozgást leíró integer függvényt generáló algoritmus összetett feltételes utasítását is. Ha ugyanis a szabad esést egy lineárisan rugalmas felfüggesztés akadályozza, harmonikus rezgőmozgás áll elő. A szabad esésnél az XX egyenletes gyorsulást az állandó súlyerő hozza létre. Rugalmas felfüggesztés esetén viszont a rugóerő ez ellen hat, mégpedig lineáris rugalmasság esetén, oly módon, hogy minden egyes azonos mértékű elmozduláshoz, azaz minden egyes *j* lépéshez, az ellenerő szintén azonos mértékű növekedése tartozik, ez pedig Newton 2. törvénye szerint ugyancsak *j* lépésenként azonos mértékű változást eredményez a gyorsulásban is. A harmonikus rezgőmozgáshoz tartozó algoritmus esetén tehát az XX gyorsulás minden *j* lépésben ugyanannyival csökken, amit azzal valósíthatunk meg, hogy egy XXY azonosítójú változónak zérusnál kisebb kezdeti értéket adunk:

If R ≤ 0 Then X = X + XX: R = R + X: i = i + 1 Else **XX = XX + XXY**: R = R – Y: j = j - 1.

A választott XXY azonosító természetesen megfelel az integer függvényeket előállító algoritmusok jelölési alapszabályának. (Megjegyzendő, hogy a *j* koordináta változása értelemszerűen negatív, mivel az ábrán a *j* koordináták felfelé növekednek, viszont a súlyerő nyilván lefelé működik, és így lefelé való elmozdulást eredményez.)

6) Trigonometrikus függvények számítása egész számokkal?

A fentebbi ábrán látható integer függvény az

$[i_0, j_0] = [0, 9]$, X=9, Y=99, XX=9, XXY=-1 és R=-89

kezdőértékekhez tartozik. Ezekből kiindulva a harmonikus rezgést leíró integer függvény kiszámításhoz elég pár perc, egy ceruza és egy kis darab papír, érdemes kipróbálni. A következő táblázat ezt a számítást mutatja:

R	-89	-71	-44	-8	37	-62	-9	52	-47	21	-78	-4	76	-23	62	-37	52	-47	45	-54	40	-59	36	-63	32	-67
RX	9	27	54	90	135	135	188	249	249	317	317	391	471	471	556	556	645	645	737	737	831	831	926	926	1021	1021
RY	98	98	98	98	98	197	197	197	296	296	395	395	395	494	494	593	593	692	692	791	791	890	890	989	989	1088
X	9	18	27	36	45	45	53	61	61	68	68	74	80	80	85	85	89	89	92	92	94	94	95	95	95	95
Y	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99	99
XX	9	9	9	9	9	8	8	8	7	7	6	6	6	5	5	4	4	3	3	2	2	1	1	0	0	-1
XXY	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1	-1
i	0	1	2	3	4	4	5	6	6	7	7	8	9	9	10	10	11	11	12	12	13	13	14	14	15	15
j	9	9	9	9	9	8	8	8	7	7	6	6	6	5	5	4	4	3	3	2	2	1	1	0	0	-1

Nem tévedünk, ha – visszaemlékezve a harmonikus rezgésekre vonatkozó korábbi tanulmányainkra – arra gondolunk, hogy a kétjegyű egész számok körében végzett néhány összeadással/kivonással tulajdonképpen egy olyan integer függvényhez jutottunk, amely a *folytonos* koszinusz függvény egy szakaszával állítható párhuzamba. Az előző oldali ábrán a piros „*folytonos*” vonal be is jelöli ezt a függvényszakaszt. (Az idézőjelek *egyszeri* használata a folytonosság *két* említésénél nem elírás: *elgondolni* lehet folytonos koszinusz függvényt, de *az anyagi világban a maga absztrakt tökéletességében megjeleníteni* nem.) Ezen igencsak egyszerű számítás tanulságaként akár azt a megjegyzést is tehetnénk, hogy a digitális korszakban a második általános elvégzése után, amikor már „százig jól megy az összeadás/kivonás”, el lehet kezdeni a trigonometrikus függvényekkel ismerkedni…☺

7) Az integer lépték.

Az előző oldali ábrán látható integer koszinusz persze eléggé „durva”, a legegyszerűbb gyakorlati alkalmazások is lényegesen nagyobb pontosságot igényelnének, a bejelölt piros görbével pedig az a „baj”, hogy az ábra tengelyeire beírt számok 10-szeresei a folytonos *y=cos(x)* függvény jól ismert *y* értékeinek. A dolgok nyitja ismét az, hogy az ábra „hibrid”, a beírt kották az *integer* függvényhez tartoznak, a piros vonal pedig a *valós matematikai analízis* koszinuszgörbéjére utal. Míg a digitális számítások alapja mindig és kizárólag az egész számok véges sorozata, addig a folytonosnak elgondolt világban különböző mérték-egységeket használunk, és a számokban különböző helyeken (kettedes/tizedes) vesszőt is alkalmazhatunk. Ilyen körülmények között az egyértelmű kapcsolatot úgy biztosíthatjuk, ha megadunk egy integer léptéket, amely megmutatja, hogy az integer világ mindig ugyanazon $\underline{\underline{1}}$ egységének az éppen vele párhuzamba állítani kívánt „folytonos” rendszerben mi felel meg. Az előző ábrán a „digitális” egységnek egy tized „valós” egység felel meg, azaz az integer lépték $\underline{\underline{1}} \rightarrow 0{,}1$. A fentebb elvégzett kis számítás tehát a diszkrét koszinuszt egy tizedes pontossággal határozta meg. Az előző mondat azonban szó szerint nyilván csak a diszkrét és a folytonos megközelítések közötti párhuzamosság (szintézis?, átmenet?) *viszonylatában* értelmezhető, az integer függvényekkel való számítások a maguk diszkrét digitális világában természetükből adódóan mindig „abszolút” pontosak.

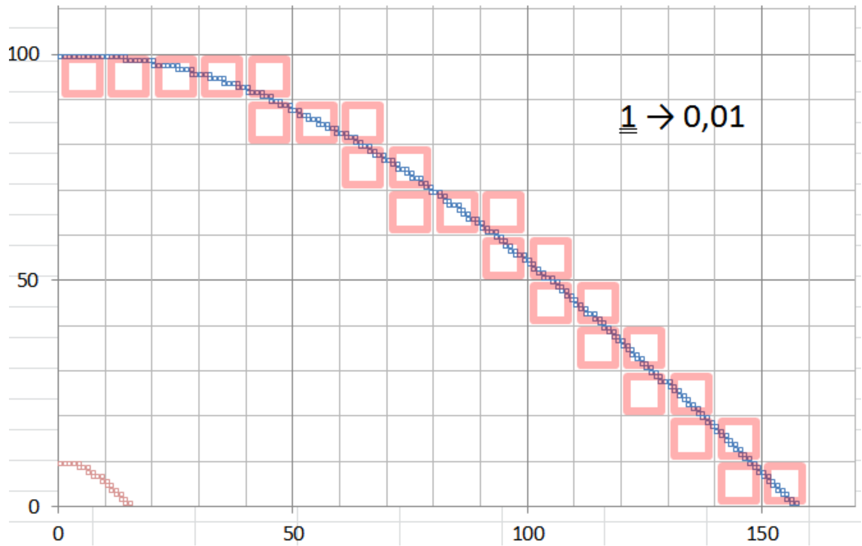
8) Különböző hosszúságú összeillő integer függvények.

A bemutatott integer számítási módszer a kezdőértékek nagyságára és az ismétlések számára nézve semmiféle megkötést nem tartalmaz (azon túlmenően, hogy véges egész számok). Tetszőleges pontosság elérhető, „csak” többször kell a szóban forgó összetett feltételes utasítást ismételni, és nagyobb egész számokat kell használni. Ekkor persze a „papír és ceruza” már nem elég, de egy kommersz laptop és elemi programozási lehetőséggel rendelkező táblázatkezelő rendszer már alkalmas fokozott pontossági igények kielégítésére. Ehhez természetesen több elemet tartalmazó, hosszabb, részletesebb, úgy is mondhatjuk, finomabb integer függvények kellenek.

A következő oldali ábra a szóban forgó koszinusz görbeszakasz eddig vizsgált k_l=25 elemű, rövid (egy tizedes pontosságúnak értelmezhető) változatát egy („két tizedes pontosságú”) $k_{l'}$=*257* elemű, XXY=-1: XX=99: YY =0: X=0: Y= 9900: R=-9899 kezdőértékű finomabbal együtt mutatja. Az ábrán az „egy tizedes” integer koszinusz tulajdonképpen kétszer szerepel, egyszer a bal alsó sarokban a maga $\underline{\underline{1}} \rightarrow 0{,}1$ saját integer léptékével, másodszor pedig tízszeresre nagyítva, s ezzel a részletesebb integer függvény $\underline{1} \rightarrow 0{,}01$ léptékéhez igazítva. Mindkét függvényt ugyanaz az

If R $\leq$ 0 Then X = X + XX: R = R + X: i = i + 1 Else XX = XX + XXY: R = R – Y: j = j - 1

összetett feltételes utasítás generálta, különböző nagyságú/nagyságrendű kezdeti értékekkel és különböző ismétlési szám mellett.



Jelölje $[i_k, j_k]$ a kevésbé részletes, $[I_K, J_K]$ a részletesebb integer függvény egy-egy általános elemét, és legyen m a részletezettség növelésének mértéke, esetünkben $m = 10$. Tekintsük most az

$$m.\ i_k, <= I_K < m.\ (i_k,+1) \text{ és } m.\ j_k, <= J_K < m.\ (j_k,+1)$$

egyenlőtlenségeket. Ha ezek az egyenlőtlenségek fennállnak, akkor azt mondhatjuk, hogy a kevésbé részletes integer függvény $[i_k, j_k]$ eleme tartalmazza a részletesebb függvény $[I_K, J_K]$ elemét, ha pedig a részletesebb függvény minden elemét tartalmazza a kevésbé részletes függvény valamelyik eleme, akkor a rövidebb függvény tartalmazza, „lefedi" az egész részletesebb, hosszabb függvényt, a hosszabb pedig részletezi, finomítja, "pontosítja" a rövidebbet. Ha ehhez még hozzáértjük azt is, hogy a két függvény ugyanazon típusba (pontosabban lásd majd a 9. részben) tartozik, a két függvényt különböző hosszúságú, részletességű összeillő integer függvényeknek tekintjük. A fenti ábra egy ilyen függvénypárt mutat.

Az integer függvények számítása tehát tetszőleges (de persze mindig véges) részletezettséggel végrehajtható. Megjegyzendő, hogy a *véges* részletezettség nem a „diszkrét elgondolás gyengéje", akármilyen más gondolati háttérrel végezzünk is *digitális számítógépen* számítást, mindenképpen „csak" ennyi valósítható meg. Tekintsük például a π/2 „digitális számításból" nyerhető értékeit. Az előző ábráról leolvasható, hogy a π/2 értéke az $\underline{1} \rightarrow 0{,}1$ integer lépték mellett 1,5, az $\underline{1} \rightarrow 0{,}01$ integer lépték mellett pedig 1,57. A számítás, (ugyanazokkal a kommersz számítógépi és szoftver lehetőségekkel) elvégezhető pl. $\underline{1} \rightarrow 0{,}000001$ integer lépték esetén is, ekkor a „digitálisan számított" π/2 = 1,570796-ra adódik. Ehhez 8 byte hosszú integer változókat kellett alkalmazni, a szóban forgó feltételes utasítást pedig mintegy két és fél milliószor kellett ismételni, ami a használt, 2,53 Ghz-es processzorú laptopon 0,2 sec gépidőt igényelt. A számítógépen természetesen lekérhető a hagyományos módon, a valós matematikai analízis gondolati háttere alapján (ill. ennek numerikus közelítésével) számított π/2 érték is: π/2 = 1,5707963267949, ami szintén véges, bár minden bizonnyal nagyobb pontosságú. Pontosságának megítélésére a valós függvények numerikus matematikai analízise szabatos hibahatárokat tud adni, de valamely adott véges bitszámú lebegőpontos processzor és adott felhasználói programrendszer esetében nem könnyű az éppen érvényes hibahatárnak utánanézni. A jelen esetben azonban elmondhatjuk, hogy az első hat tizedes jegy „biztosan jó”, hiszen megegyezik a fentebbi, „digitálisan számított” értékkel ☺.

9) Kapcsolat az integer és a valós függvények között.

Az 5. szakaszban az integer függvények körében leírtuk a harmonikus rezgés fizikai jelenségét, az ezt követő három szakaszban pedig ennek a megfogalmazásnak az alapján végezhető számításokat mutattunk be. Ennek során emlékeztettünk arra, hogy a harmonikus rezgés a valós matematikai analízisnek is egyik alapfeladata, s ott az $y'' + C.y = 0$ differenciálegyenlet egy megfelelő megoldása, esetünkben a valós koszinusz függvény adja a szóban forgó fizikai jelenség matematikai leírását. Ha a bemutatott integer függvényes digitális megoldás és valós koszinusz függvényes megoldás egyazon fizikai jelenségnek ugyan logikailag egymástól független, de érvényes leírása, a kettő között meg kell mutatkoznia valami kapcsolatnak.

Az egyenes vonalú egyenletes mozgás tárgyalásánál bevezettük az integer függvények elemeit alkotó $[i_k, j_k]$ integer számpárok szemléletes ábrázolására az egységnégyzet szimbólumokkal való megjelenítést. Ha most a diszkrét integer függvények és a folytonos valós függvények valamiféle szabatos kapcsolatáról kívánunk beszélni, akkor tekintsük az integer-párokat jelölő egységnégyzet szimbólumokat egyszersmind a valós sík (x, y) pontjai megfelelő tartományainak. (Az, hogy a tartomány nyílt vagy zárt voltát nem említjük, tudatosan történik, ennek a megkülönböztetésnek tárgyunk szempontjából nincs értelme.) Ha egy valós függvénynek nincs olyan pontja, amely ne tartozna egy adott integer függvény valamely egységnégyzetéhez és az adott integer függvény minden egységnégyzetébe ténylegesen esik is pontja, akkor azt mondjuk, hogy az illető valós és integer függvények összetartozóak, egymás megfelelői. (Persze az integer függvények mindig végesek, a valós függvények értelmezési tartománya pedig sokszor nem az, ilyenkor a valós függvények véges *szakaszai* értendők.) Az előzőekben tárgyalt (lineáris, ill. koszinusz) diszkrét és folytonos függvények ebben az értelemben egymás megfelelői. Általánosságban elmondhatjuk, hogy egy integer függvénynek van számos (vagy inkább megszámlálhatatlanul sok) valós megfelelője, egy valós függvénynek pedig van egy (adott léptékű) integer megfelelője. (Az utóbbi állításhoz a pontosság kedvéért hozzá lehet tenni, hogy *legfeljebb* egy, hiszen pl. egy mindenhol folytonos, de sehol sem deriválható valós függvény integer társáról nincs értelme beszélni.)

Az integer függvények és a valós függvények viszonyának érzékeltetésére célszerű lehet még a 8. szakaszban említett különböző hosszúságú összeillő integer függvényekre is visszagondolni. Egy integer függvénynek lehet részletezettje. Ez maga is integer függvény, tehát tovább részletezhető, és nincs semmi, ami a részletezettség (persze mindig véges) mértékét (a hardver adottságokon túl) elvileg korlátozná. Hogy ez a meggondolás átvezet-e a valós függvények világába, egy olyan matematikafilozófiai kérdés, amelynek további vizsgálata nem tárgya a jelen rövid ismertetőnek.

10) Az integer függvények rendszere

A bemutatott három elemi példa integer függvényekkel való leírásához, mint láttuk, három rokon szerkezetű, de nem teljesen azonos összetett feltételes utasítást használtunk, az elsőnél 4, a másodiknál 5, a harmadiknál pedig 6 összeadás/kivonás szerepel az integer függvény ujabb és újabb elemeinek a megahatátozásához szükges aritmetikai műveletként:

If R ≤ 0 Then R = R + X: i = i + 1 Else R = R – Y: j = j + 1, ***~~{~~X, Y~~}~~***,

If R ≤ 0 Then X = X + XX: R = R + X: i = i + 1 Else R = R – Y: j = j + 1, ***~~{~~XX, Y~~}~~***,

If R ≤ 0 Then X = X + XX: R = R + X: i = i + 1 Else XX = XX + XXY: R = R – Y: j = j + 1, ***~~{~~XXY, Y~~}~~***.

Említettük azt is, hogy az X és Y betűjelekből álló azonosítók olyan kialakításúak, hogy maga az azonosító kifejezi a vele azonosított változó szerepét az algoritmusban. Ha ugyanis az utolsó betűjel X, akkor az *i* lépésben, ha Y akkor a *j* lépésben aktív, s az aktivitás abból áll, hogy hozzáadódik az utolsó jegy elhagyásával nyerhető, eggyel kevesebb betűjelből álló azonosítójú változóhoz, az egybetűs X és Y változó pedig az R „vezérlő változóhoz" adódik hozzá, ill. abból vonódik le. Az e szabályosságnak eleget tévő, integer függvényt generáló algoritmusokat integer függvényalgoritmusoknak nevezhetjük. Egy integer függvény-algoritmus teljes szerkezete meghatározható abból, ha megadjuk azokat a *zérustól különböző kezdőértékű* azonosítókat, amelyek *csak hozzáadandóként* szerepelnek, összeadás eredményének befogadójaként pedig nem, tehát előfordulnak értékadó utasításban, de csak a jobb oldalon. Ezeket az azonosítókat megkeresve és, mondjuk, áthúzott kapcsos zárójelpárba *~~{~~ ~~}~~* írva, kapjuk az illető algoritmussal generálható integer függvények típusjelét. A felsorolt három esetre vonatkozóan e típusjeleket fentebb piros színnel kiemelve be is írtuk az összetett feltételes utasítások mellé.

Egy integer függvényt egyértelműen megadhatunk a típusjelével, és a típusjelben foglalt regiszterek kezdőértékeivel. A „regiszterek" megnevezés alatt a nagybetűkkel jelölt változókat értjük. Így az *~~{~~XXY, Y~~}~~* típusjel esetén az R „vezérlő" regiszternek, a típusjelben megjelenő XXY és Y, továbbá az „ezekben bennük foglalt" XX és X „munka" regisztereknek, tehát összesen öt regiszter változónak és persze az *i* és *j* koordinátáknak kell kezdőértéket kapniuk.

Eddig három különböző típusú integer függvényről volt szó. Az integer függvények fogalma és az integer függvényeket előállító függvényalgoritmus több irányban és igen széles körben általánosítható. A fentebb tárgyalt esetekben a regiszterek azonosítóiban csak két betűjel szerepelt, az X és az Y. Ezzel síkbeli feladatokat tudtunk tárgyalni. Térbeli függvények esetén meg kell engedni a Z betűjelet, időbeli változás vizsgálatához még (mondjuk) a T-t is. Eddig legfeljebb három betűjelből állt egy azonosító, de szükség szerint több (persze véges számú) betű is használható. Az egy azonosítóhoz használt betűjelek számát „rendűségnek” is mondhatjuk. Alább az integer függvényeket előállító algoritmus „2 dimenziós harmadrendű” alakja, továbbá egy ezen alapuló *egyetlen egyszerű programmal* kidolgozott mintapéldák sorozata látható. Az algoritmus leírása még annyiban is speciális, hogy csak monoton nem csökkenő függvényszakaszokra vonatkozik (az *i* és a *j* minden lépésben növekszik). Azt, hogy e tekintetben is lehetséges az általánosítás, a mintapéldák is mutatják. (Ezen általánosítás alapelve az, hogy az összeadó/kivonó műveletek előjeleit az elsőrendű regiszterek előjeleihez kell igazítani, de az általánosítások részletes tárgyalása meghaladja a jelen rövid bemutató kereteit.)

If R ≤ 0 Then

XX = XX + XXX: XY = XY + XYX: YX = YX + YXX: YY= YY+YYX

X = X + XX: Y = Y + YX

R = R + X

i = i + 1 (i step)

Else

XX = X + XXY: XY = XY + XYY: YX = YX +YXY: YY = YY +YYY

X = X + XY: Y = Y + YY

R = R – Y

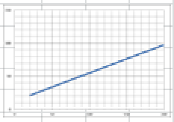
j = j + 1 (j step)

End If.

ALGEBRAI FÜGGVÉNYEK (és inverzeik)

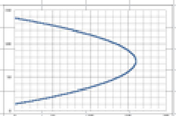
EGYENES {X, Y}

i=	20	j=	20	R=	0	k=	2000
X	3	Y	7				
XX	0	XY	0	YX	0	YY	0
XXX	0	XXY	0	XYX	0	XYY	0
YXX	0	YXY	0	YYX	0	YYY	0



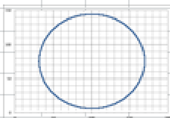
Vizszintes tengelyű PARABOLA {X, YY}

i=	0	j=	10	R=	400	k=	2000
X	140	Y	700				
XX	0	XY	0	YX	0	YY	-11
XXX	0	XXY	0	XYX	0	XYY	0
YXX	0	YXY	0	YYX	0	YYY	0



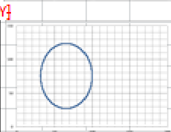
KÖR {XX, YY}, ha |XX|=|YY|, de XX ≠YY.

i=	30	j=	75	R=	-35	k=	2000
X	70	Y	0				
XX	-1	XY	0	YX	0	YY	1
XXX	0	XXY	0	XYX	0	XYY	0
YXX	0	YXY	0	YYX	0	YYY	0



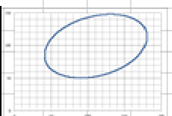
ELLIPSZIS koordináta irányú főtengelyekkel {XX, YY}

i=	30	j=	75	R=	100	k=	2000
X	280	Y	0				
XX	-8	XY	0	YX	0	YY	4
XXX	0	XXY	0	XYX	0	XYY	0
YXX	0	YXY	0	YYX	0	YYY	0



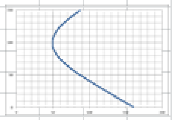
Általános helyzetű ELLIPSZIS {XX, XY=YX, YY}

i=	40	j=	80	R=	100	k=	1000
X	-750	Y	120				
XX	12	XY	-5	YX	5	YY	-25
XXX	0	XXY	0	XYX	0	XYY	0
YXX	0	YXY	0	YYX	0	YYY	0



HIPERBOLA {XX, YY}, ha sgn(XX) = sgn(YY)

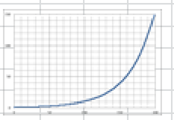
i=	160	j=	0	R=	0	(k=	2000
X	600	Y	-990				
XX	3	XY	0	YX	0	YY	10
XXX	0	XXY	0	XYX	0	XYY	0
YXX	0	YXY	0	YYX	0	YYY	0



TRANSZCENDENS FÜGGVÉNYEK (és inverzeik)

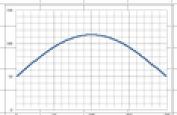
EXPONENCIÁLIS {X, XY}

i=	0	j=	0	R=	-1	k=	2000
X	1	Y	501				
XX	0	XY	14	YX	0	YY	0
XXX	0	XXY	0	XYX	0	XYY	0
YXX	0	YXY	0	YYX	0	YYY	0



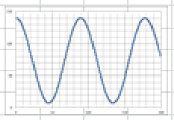
TRIGONOMETRIKUS {X, XXY} (SZINUSZ)

i=	0	j=	50	R=	1800	1	2000
X	3960	Y	3960				
XX	0	XY	0	YX	0	YY	0
XXX	0	XXY	-1	XYX	0	XYY	0
YXX	0	YXY	0	YYX	0	YYY	0



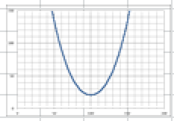
TRIGONOMETRIKUS {X, XXY} (KOSZINUSZ)

i=	0	j=	140	R=	500	1	2000
X	0	Y	600				
XX	-200	XY	0	YX	0	YY	0
XXX	0	XXY	-3	XYX	0	XYY	0
YXX	0	YXY	0	YYX	0	YYY	0



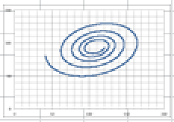
KÖTÉLGÖRBE, KOSZINUSZ hiperbolikusz {X, XXY}

i=	46	j=	148	R=	-1000	k=	2000
X	-8500	Y	1500				
XX	245	XY	0	YX	0	YY	0
XXX	0	XXY	1	XYX	0	XYY	0
YXX	0	YXY	0	YYX	0	YYY	0



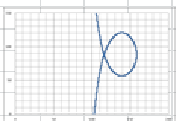
"SPIRÁLIS" {XX, XY, YX, YY}

i=	40	j=	80	R=	100	k=	1000
X	-750	Y	120				
XX	12	XY	-5	YX	3	YY	-25
XXX	0	XXY	0	XYX	0	XYY	0
YXX	0	YXY	0	YYX	0	YYY	0



"HUROK" {XX, YYX }

i=	102	j=	0	R=	-111	k=	2000
X	11020	Y	1000				
XX	-300	XY	0	YX	0	YY	0
XXX	0	XXY	0	XYX	0	XYY	0
YXX	0	YXY	0	YYX	2	YYY	0



Az integer függvényekkel foglalkozva úgy tűnik, hogy a bemutatott típusjelek az integer függvények véges sokaságának egy jól áttekinthető és teljes rendszerezését adják, ami a digitális megközelítés igen nagy pozitívuma. Ilyen rendszerezés a valós folytonos függvények körében nem ismert, ezek összessége azonban nyilván minőségileg gazdagabb is, mint az integer függvényeké. A töréspont, csúcspont esetei csak „hirtelen" változás formájában jelennek meg az integer függvényeknél, s ugyanígy a valós függvények körében értelmezhető sajátértékekkel kapcsolatos stabilitásvesztés, rezonancia stb. absztrakt formái is. Ez komoly hiányosságnak tűnhet, de például sok műszaki feladatnál amúgy is vizsgáljuk a posztkritikus állapotot is, és a megfelelő kritikus értékeket átlépve a „baj", ha nem is (mérnöki szemmel amúgy is nehezen értelmezhető) „egyetlen pillanat alatt bekövetkező teljes katasztrófaként", de „kedvezőtlen hirtelen mennyiségi változásként" az integer függvények használata estén is megmutatkozik (hasonlóan a kísérletileg megfigyelhető valóságos jelenségekhez).

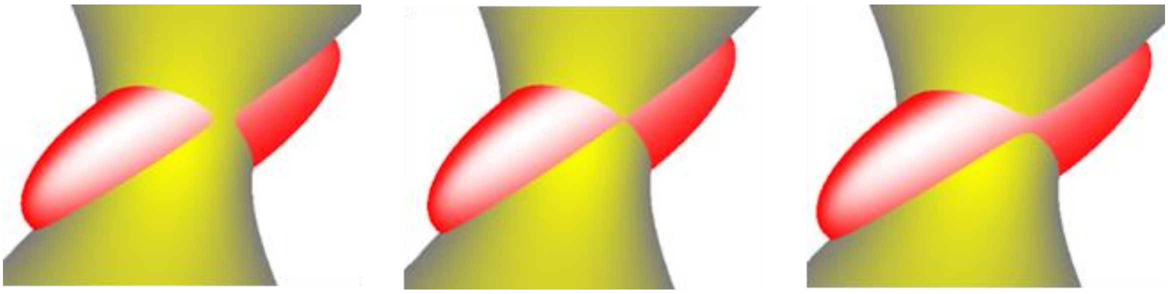
Az integer függvények és a folytonos függvények rendszerei között sok más elvi eltérés is feltűnhet. Az integer függvényeknél a két változó, az *i* és a *j* szerepe teljesen szimmetrikus, nincs független és függő változó, csak két azonos szerepű változó közötti összefüggés van. Ha egy adott i_a –hoz tartozó j_a –t kívánjuk meghatározni, a kezdőponttól kell elindítani a generáló algoritmust és az $i = i_a$ –nál kell leállítani. Pontosan ugyanígy megy a számítás akkor is, ha *j* az adott, csak a leállítás feltétele nem az *i*–re, hanem a *j*–re vonatkozik. Ezt a valós függvényeknél használatos terminológiával úgy is mondhatjuk, hogy minden integer függvény egyben önmagának az inverze is. Így az ***{XX, Y}*** nem csak az $y=a.x^2+b.x+c$ valós függvénynek, hanem ugyanúgy az $x = \frac{-b \pm \sqrt{b^2-4a(c-y)}}{2a}$ inverz valós függvénynek is integer megfelelője.

Ezek után magától értetődő, hogy például a valós ötödfokú polinomok integer változatait jelentő ***{XXXXX, Y}*** típusú integer függvények gyökeit ugyanolyan egyszerű módon és ugyanolyan értelemben garantált pontossággal számíthatjuk ki, mint bármely *j*-hez tartozó *i* értéket. Ez talán (kissé felületesen gondolkodva) akár meglepőnek is tűnhet, hiszen emlékezhetünk olyasmire, hogy az „ötödfokú polinomok gyökei általánosságban nem számíthatók ki". Természetesen az 1824-ben közzétett Abel-Ruffini tétel, amiből következik, hogy *a négy alapművelet és az n-edik gyökvonás segítségével* nem lehet *általános megoldó képletet* adni az ötödfokú egyenletek gyökeinek számítására, a „digitális korszakban" is érvényben marad. Ehhez azonban, a maga korában magától értetendően, még az is hozzá volt értendő, hogy valós számokkal végzett műveletekről van szó, s a gyökök maguk is valós számok, nem pedig olyan egész számok, amelyek valós számok intervallumaival állíthatók párhuzamba. Az integer függvények rendszere tehát nem jelent ellentmondást, inkább kedvező, hogy a véges integer függvények széles (mondhatni teljes) körét egységes módon kezeli. Ennek megfelelően az integer függvények körében nem beszélünk elemi függvényekről sem, így az a kérdés is értelmezhetetlen, hogy valamely függvény előállítható-e „zárt alakban", s persze közelítő eljárások (pl. sorfejtés) alkalmazása sem jön szóba.

Érdemes még megemlíteni, hogy valamely adott generáló algoritmus *egy bizonyos* integer függvényt generál, de ugyanazt az integer függvényt *több különböző* generáló algoritmussal is elő lehet állítani, pl. egy ***{X, Y}*** egyenesnél valamely adott X_0, Y_0 kezdőértékek és azok tetszőleges egész számmal való $c.X_0$, $c.Y_0$ felszorzottjai ugyanazt az integer függvényt generálják, de sok más, kevésbé triviális eset is létezik. Az ilyen „egyenértékű” generáló algoritmusok tanulmányozása az integer függvények rendszerének mind teljesebb kibontakoztatása során igencsak hasznos lehet.

11) Hogyan működnek az integer függvények?

A valós matematikai analízis alapfeladatai közé tartozik a függvények érintőinek, görbületi viszonyainak vizsgálata, jellegzetes pontjainak (szélsőérték, inflexió) megkeresése. E feladatok megoldása közül több is szinte magától adódik az integer függvények körében. Az X és az Y munkaregiszterek mindenkori értéke megadja a diszkrét görbe helyi normálisát, az XX és az YY alapján pedig, azokat a normálissal egybevetve, a helyi görbületi sugár is kiadódik. A szélsőértékre az X vagy az Y előjelváltása jellemző, az inflexiós pontoknál pedig az XX, illetve az YY tekintetében következik be előjelváltás. Görbült felületek szemléletes, árnyalt ábrázolásánál például igen kényelmes, hogy a folytonosak elgondolt felületet tartalmazó szomszédos „egységkockák”, voxelek egész számhármasainak az integer függvényalgoritmus 3D-s változatával való generálásakor az X, Y és Z elsőrendű munkaregiszterek egyszersmind a helyes árnyalás alapjául szolgáló felületi normálisokat is megadják:



Az integer függvényeket generálásánál szereplő regiszterek tehát gyakran használt, fontos geometriai jellemzőkkel hozhatók összefüggésbe.

A jelen rövid ismertető első szakaszaiban fizikai jelenségekhez (nevezetesen az egyenes vonalú egyenletes mozgáshoz, a szabad eséshez és a harmonikus rezgéshez) kapcsolódva vezettük be az integer függvények fogalmát. Ez egyben azt is mutatja, hogy az integer függvényeket generáló összetett feltételes utasítás változói, regiszterei fizikai jelenségek leírásánál is jól felhasználhatók, példáinkban X a sebességet, XX a gyorsulást, XXY a rugalmas ellenerőnek a gyorsulást módosító hatását fejezte ki.

Mindezeket a feladatokat a valós matematikai analízis keretében a differenciálhányadosok illetve a derivált függvények alkalmazásával kezeljük. Ezekkel szoros párhuzamba állítható diszkrét struktúrák a digitális szemlélet keretében is kialakíthatók, de lehet, hogy ezek tárgyalása nem is okvetlenül tartozik egy ilyen rövid bemutatóhoz, hiszen az integer függvények rendszerének hatékonysága talán már az eddigiekből is érzékelhető. A következő négy szakaszban (a még kitartó olvasókra gondolva) mégis érintjük ezt a problémakört, mivel ebből a diszkrét és a folytonos megközelítések viszonyára vonatkozóan alapvető elvi következtetések adódnak.

12) Integer függvények különbségi mezője.

A differenciálhányados, azaz a differenciák hányadosának határértéke két okból is kívül esik a diszkrét digitális világ gondolatkörén. Egyrészt már a differenciák *hányadosa* is csak a 3. szakaszban mutatott váltakozó kivonogatás, a klasszikus görög „anthyphairesis" formájában kezelhető az egész számok körében, a *határérték* fogalma pedig eleve értelmezhetetlen. Az angol nyelvű szakirodalomban valamiféle „digitization of calculus"-nak vagy „discrete calculus"-nak nevezett rendszer létrehozására irányuló törekvéseknél egyszerűen véges differenciákkal próbálnak dolgozni, eléggé szegényes eredménnyel. Az integer függvényeknek a szóban forgó, és a gyakorlatban hatékonynak mutatkozó diszkrét rendszerében az integer függvények különbségi mezője s az erre illeszkedő diszkrét deriváltak töltenek be a valós függvények differenciálhányadosával, derivált függvényével szoros párhuzamba állítható szerepet.

A különbségi mező tárgyalásához tekintsük az integer függvény azon elemeit, amelyekre i lépéssel lépünk s nevezzük ezeket „i-karakterisztikus" elemeknek. (Természetesen a két változó teljes szimmetriája miatt a j karakterisztikus elemekkel is ugyanúgy dolgozhatunk.) Az integer függvények monoton szakaszain minden szóba jövő i oszlopban van egy i karakterisztikus elem, legyen j ennek másik koordinátája. Egy i oszlopban természetesen több elem is lehet, különböző j koordinátákkal, azonban a jelen részben csak a karakterisztikus elemekkel foglalkozunk, így ennek állandó hangsúlyozását a könnyedebb fogalmazás és jelölés érdekében mellőzzük. Tekintsünk két ilyen karakterisztikus elemet, $[i_k, j_k]$-t és $[i_{k'}, j_{k'}]$-t, és képezzük ezek „különbségi számpárját" $[(i_{k'} - i_k), (j_{k'} - j_k)]$. Állítsuk elő az adott véges integer függvény esetében értelmezhető valamennyi ilyen ($k'>k$) számpárt. Ezek az adott integer függvény teljes különbségi mezőjét alkotják.

A különbségi mezőt alkotó különbségi számpárokat osszuk különbségi osztályokba, nevezetesen az $[(i_{k'} - i_k), (j_{k'} - j_k)]$ különbségi számpár tartozzék a d differencia osztályba, ha $(i_{k'} - i_k) = d$, és jelöljük az ehhez tartozó j koordináták különbségét $(j_{k'} - j_k) = {}_dj_k$ –val. Az adott véges integer függvény esetén értelmezhető ${}_dj_k$ karakterisztikus aritmetikai különbségek összessége helyről-helyre, egyértelműen és maradéktalanul leírja a j és i lépéseknek az egymáshoz viszonyított relatív gyakoriságát, azaz az integer függvény változásának menetét, s ennyiben szerepe a valós függvények deriváltjáéval rokonítható. A derivált azonban maga is valós függvény, míg a ${}_dj_k$–k különálló egész számok, s annak érdekében, hogy megfelelő integer függvényeket tudjuk hozzájuk illeszteni, még egy további meggondolás szükséges.

13) Finomítható különbségek.

A különböző részletezettségű összeillő integer függvények tárgyalásánál (8. rész) szó volt egy $[i_k, j_k]$ általános elemű kevésbé részletes, és egy $[I_K, J_K]$ általános elemű részletesebb integer függvényről. Ezek összeillő elemeinek j ill. J koordinátáira vonatkozóan fennáll az $m.\, j_k <= J_K < m.(j_k+1)$ egyenlőtlenség, ahol is m a részletezettség mértéke. Képezzük ezek ${}_dj_k$ ill. ${}_DJ_K$ karakterisztikus aritmetikai különbségeit. (Természetesen a d és D differencia osztályok is összeillők kell, hogy legyenek, most azonban csak a j ill. J koordinátákat vizsgáljuk.) A kevésbé részletes függvény esetén csak a ${}_dj_k = (j_{k'} - j_k)$ aritmetikai különbségről beszélhetünk. A finomabb esetben azonban az $m.\, j_k <= J_k < m.(j_k + 1)$ egyenlőtlenségnek megfelelő bármelyik J_k, és az $m.\, j_{k'} <= J_{k'} < m.(j_{k'}+1)$ egyenlőtlenségnek megfelelő bármelyik $J_{k'}$ szerepelhet a ${}_DJ_K = (J_{K'} - J_K)$ különbség képzésében. Így a finomított esetben akkor kapjuk a legkisebb különbséget, ha a kisebbítendő lehetséges legkisebb értékéből a kivonandó lehetséges legnagyobb értékét vonjuk ki, a legnagyobb pedig az alsó és felső határok felcserélésével adódik:

$$m.\, j_{k'} - m.(j_k + 1) = m.\, j_{k'} - m.\, j_k - m \;<=\; {}_DJ_K <= m.\,(j_{k'} - j_k + 1) - 1.$$

A kevésbé finom integer függvény esetén képezhető ${}_dj_K = (j_{k'} - j_k)$ aritmetikai különbség viszont csak az

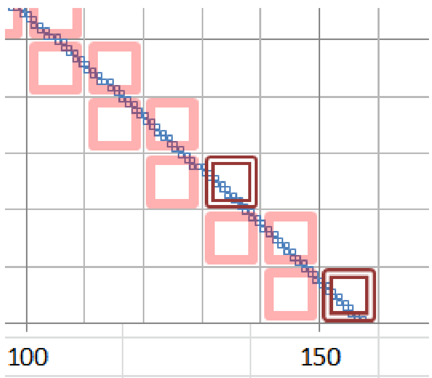
$$m.\,(j_{k'} - j_k) <= {}_DJ_K <= m.\,(j_{k'} - j_k + 1) - 1$$

egyenlőtlenségnek megfelelő ${}_DJ_K$ számértékeket tartalmazza. Az egyszerű aritmetikai különbségképzés esetén tehát nem valósul meg, hogy a kevésbé finom és a finomabb összetartozó elemek aritmetikai különbségei is ugyanabban az értelemben összetartozók maradjanak. Ezért az integer függvények vizsgálatához bevezetjük a ${}_\Delta j_K$ „finomítható különbségek” fogalmát:

$${}_\Delta j_K = j_{K'} \underset{\sim}{-} j_k = [\,(j_{k'} - j_k - 1),\, (j_{k'} - j_k)\,].$$

A finomítható különbség tehát két egymást követő egész szám: az aritmetikai különbséget megelőző egész szám, és maga az aritmetikai különbség.

A finomítható különbségek mibenlétét és jelentőségét jól megvilágítja, ha visszatekintünk a



8. szakaszban tárgyalt különböző részletességű összeillő integer függvénypárra. A baloldali ábra példaként ezen összeillő függvénypár egy szakaszát mutatja. A kevésbé részletes integer függvény két i karakterisztikus eleme $[i_k=13,\, j_k=2]$ és a $[i_{k'}=15,\, j_{k'}=0]$, kettős keretezéssel ki van emelve. A részletesebb integer függvény ezek által lefedett elemei közül a két-két szélső: $[\,130,\ 27\,]$ és $[137, 20]$, illetve $[\,150,\ 7\,]$ és $[157,\ 0]$. A két kevésbé részletes integer függvény esetén a karakterisztikus aritmetikai különbség

$${}_{d=2}j_k = (j_{k'} - j_k) = (0 - 2) = -2.$$

A finomabb esetben viszont a megfelelő ${}_DJ_K$ karakterisztikus aritmetikai különbségek

$$(0 - 27) = -27 \;\le\; {}_DJ_K \;\le\; (7 - 20) = -13$$

közé esnek. A ${}_{d}j_k = -2$ aritmetikai különbség finomítottjai (lévén a finomítás mértéke $m=10$) *20, -21, … -29*, míg a ${}_{\Delta}j_K = [(-1), (-2)]$ finomítható különbség finomítottjai *-10, -11, … -29.* A szokásos aritmetikai különbség tehát nem fedi le a finomabb különbségek teljes skáláját, a finomítható viszont igen.

A finomítható különbség képzésének megfelelő kivonási szabály természetesen jól ismert a valós számok intervallum-aritmetikájában, mint a különbség számításának az egységnyi intervallumoknál érvényes speciális esete. A jelen szakaszban mondottak csak annak megmutatására szolgálnak, hogy az érvényesség nincs a valós számok intervallumaihoz kötve. Különböző véges finomságú integer függvények lehetőségének megengedése elegendő e kivonási szabály érvényességéhez, a folytonosság, s a kontinuumnyi számosságú valós számok feltételezés nem szükséges. Így e szabály az integer függvények vizsgálata során a folytonos matematikai analízistől való teljes logikai függetlenség sérelme nélkül használható és használandó.

14) Az integer függvények diszkrét deriváltjai.

A 12. szakaszban tárgyaltuk az integer függvények karakterisztikus ${}_{d}j_k$ aritmetikai különbségeit, ezek összességét, a különbségi mezőt, és ennek különbségi osztályokra való tagolását. Említettük azt is, hogy a különbségi osztályokra osztott különbségi mező teljes képet ad az integer függvény *i* és *j* lépéseinek egymáshoz viszonyított gyakoriságáról, azaz a függvény változásának mikéntjéről a teljes integer függvény mentén, és így a valós függvények deriváltjának szerepéhez hasonló feladatot tölthet be.

Használjuk most a ${}_{d}j_k$ karakterisztikus aritmetikai különbségek ${}_{\Delta}j_k$ finomítható változatát, a $[({}_{d}j_k - 1), {}_{d}j_k]$ két egymást követő egész számot. Mindkét egész számmal és az integer függvény megfelelő $[i_k, j_k]$ *i* karakterisztikus elemének akár az egyik, akár másik koordinátájával képezhetünk egy-egy integer számpárt:

$[i_k, ({}_{d}j_k - 1)]$ és $[i_k, {}_{d}j_k]$, illetve $[({}_{d}j_k - 1), j_k]$ és $[{}_{d}j_k, j_k]$.

Az első esetben azt mondhatjuk, hogy a *i* karakterisztikus finomítható különbségek az eredeti függvény *i* változójához vannak rendezve, a másik esetben pedig azt, hogy a *j*-hez. Mivel az integer függvények rendszerében a két változó szerepe teljesen szimmetrikus, ugyanígy beszélhetünk a *i* karakterisztikus finomítható különbségek *i* vagy *j* változóhoz való rendezéséről.

Azt az integer függvényt, amely az adott eredeti (persze véges) integer függvény Δ differencia osztályában értelemmel bíró minden *i* karakterisztikus eleméhez tartalmazza saját elemeként az $[i_k, ({}_{d}j_k - 1)]$ és $[i_k, {}_{d}j_k]$ integer számpárok közül legalább az egyiket, az adott differenciaosztályú osztályderiváltnak tekintjük. Az osztályderiváltak összessége integer függvények formájában fejezi ki mindazt az információt, amely a szóban forgó integer függvény menetére, változásának mikéntjére vonatkozóan a kivonás művelete segítségével nyerhető, és ebben az értelemben a folytonos függvények derivált függvényével állítható párhuzamba. Mivel mind az *i* mind a *j* finomítható karakterisztikus különbségek esetén két-két féle rendezési lehetőség van, az integer osztályderiváltakat négy megjelenési alakban vizsgálhatjuk, így van

D differencia osztályú *i* karakterisztikus *i*-hez rendezett (röviden *i/i*),

D differencia osztályú *i* karakterisztikus *j*-hez rendezett (röviden *i/j*),

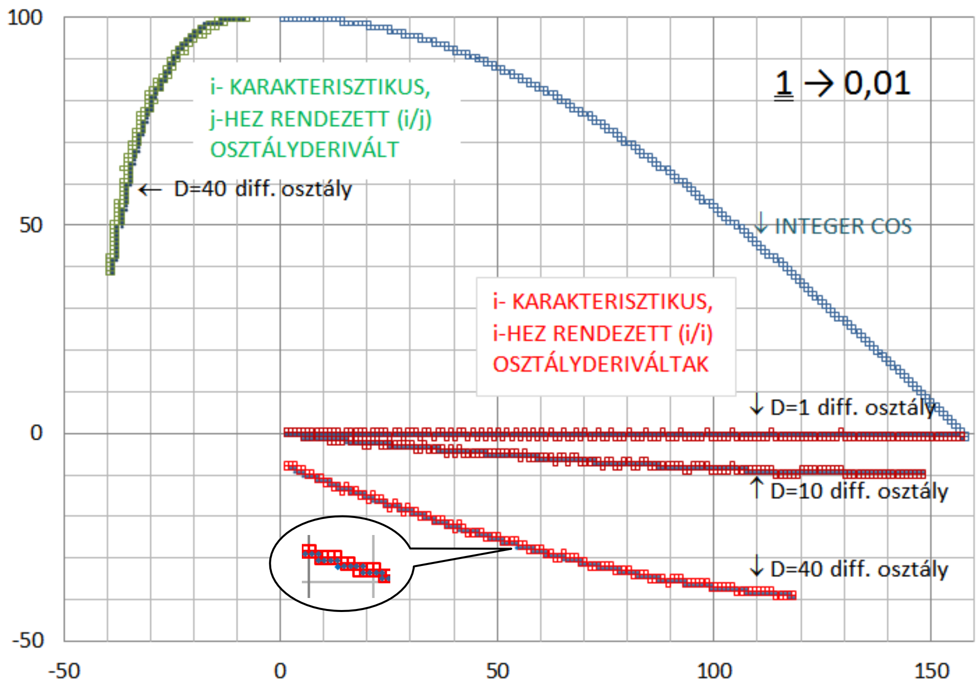
D differencia osztályú *j* karakterisztikus *i*-hez rendezett (röviden *j/i*) és

D differencia osztályú *j* karakterisztikus *j*-hez rendezett (röviden *j/j*)

osztályderivált. Ezek osztályonként külön-külön integer függvények, ha valamely alak valamennyi osztályban azonos típusú, akkor beszélhetünk általában a szóban forgó integer függvény adott alakú diszkrét deriváltjának típusáról.

15) Az integer koszinusz vizsgálata deriváltjai segítségével.

Az integer függvények deriváltjaik segítségével való vizsgálatának bemutatására a tekintsük az ismerős példát (lásd 8. szakasz). A következő ábra a vizsgálni kívánt integer koszinusz függvény *i* karakterisztikus finomítható különbségeinek és az ezekre illeszkedő osztályderiváltjainak néhány példáját mutatja. A karakterisztikus különbségeket piros (*i/i*) ill. zöld (*i/j*) üres egységnégyzetek, az osztályderivált integer függvények elemeit pedig ezekbe illő sötétkék tömör rombuszok jelölik:



Tekintsük először is az *(i/i)* osztályderiváltakat. A D=1 differenciaosztályban a finomítható karakterisztikus különbségek vagy *[0,-1]* , vagy pedig *[-1,-2]* értékűek, a *j=-1* állandó integer függvény tehát első differencia osztályú osztályderivált. Ez a koszinusz függvény menetére vonatkozóan mindössze annyi információt ad, hogy a D=1 (azaz szomszédos) *i*-karakterisztikus különbségek között nincs sem 0-nál nagyobb, sem -1-nél kisebb. Az persze „ránézésre" látható, hogy a $_{d}j_k = 0$ értékek inkább a kisebb i_k-k esetén, a $_{d}j_k = -1$ értékek viszont inkább a nagyobb i_k-k esetén fordulnak gyakrabban elő, de éppen ennek számszerű kifejezésére alkalmasak a magasabb differencia osztályok. Már a D=10 differenciaosztály is valamivel több információt mutat a meredekség alakulására vonatkozóan, a D=40 differenciaosztály osztályderiváltja pedig már határozottan emlékeztet az *y=-sin(x)* valós függvénynek, azaz az *y=cos(x)* deriváltjának, egy szakaszára.

Az *(i/j)* osztályderiváltak közül csak az információban gazdag D=40 differenciaosztályt tünteti fel az ábra. Ha ebben egy ellipszisívet sejtünk, nem tévedünk, hiszen ha a valós koszinusz függvény esetében kiszámítjuk a $\Delta y = cos(x + \Delta x) - cos(x)$ értéket a szögek összegére vonatkozó ismert képlet alapján, akkor rögzített Δx esetén a Δy és a *cos(x)* összefüggésére egy valós másodrendű algebrai görbét nyerünk. Ez egyrészt megnyugtathat a felől, hogy a valós függvények és az integer függvények egymástól logikailag független két világa között ellentmondás e tekintetben sem jelentkezik, másrészt azonban azt is mutatja, hogy a talán „furcsa" *(i/j)* osztályderiváltaknak a digitális szemléletben való megjelenése nem is felesleges. A diszkrét derivált négy megjelenési alakja az integer függvények analízise során olyan tulajdonságokra is rávilágíthat, amelyekkel párhuzamos tulajdonságok természetesen a megfelelő valós függvényeknél is megvannak, a valós matematikai analízis eszköztárával is kifejezhetők, de a szokásos, „tankönyvi" anyag kevésbé hívja fel rájuk a figyelmet.

16) Hol helyezhető el az integer függvények rendszere a szak-irodalomban?

A világhálón hozzáférhető szakirodalomban újabban számos helyen igény mutatkozik a függvényeknek és a függvények vizsgálatának, analízisének a diszkrét matematika keretében való tárgyalására:

- vezető egyetemek rendszeresen hirdetnek ilyen jellegű tantárgyakat,
- megjelent több ezzel (is) foglalkozó könyv,
- érdekes, hogy az MIT-n működő „népszerű matematika" megnevezésű ismeretterjesztő beszélgetés-sorozatban is volt már ennek a témának szentelt összejövetel, s ennek során az „egész számokból álló függvény" fogalma is felmerült, a valós függvény „kuzinjaként" aposztrofálták.

Az integer függvények rendszere tehát napjainkban jelentős érdeklődést kiváltó területet érint. Az alapkövetelmények között gyakran hangsúlyozzák a diszkrét analízis teljes logikai függetlenségét a „hagyományos" valós analízistől, (illetve a tényleges számszerű eredményeket nyújtani képes közelítésétől, a numerikus analízistől), amint ezt a jelen bemutató is tette az integer függvények rendszere vonatkozásában.

A síkbeli analízis különböző megjelenéseinél szinte egyöntetű a diszkrét tárgyalás kiindulása:

- a (valós) függvény diszkrét változata az $a_1, a_2, \dots a_k, \dots a_n$ sorozat,
- a derivált diszkrét változata a $\Delta a_1, \Delta a_2, \dots \Delta a_k, \dots \Delta a_{n-1}$ sorozat, ahol is $\Delta a_k = a_{k+1} - a_k$.

(Idézve a University of Pennsylvania neten hozzáférhető analízis jegyzete alapján.)

Ehhez képest az integer függvények bemutatott rendszere több tekintetben is alapvető eltérést mutat:

- Az integer függvény - mint láttuk - *[i_k, j_k]*, ($k = 0, 1, \ldots l$) szomszédos integer számpárok véges sorozata. Ebben a két változó, az i_k, és a j_k teljesen azonos szerepet játszik. Természetesen, ha függvényről beszélünk, kell lenni (legalább) két változónak, amelyek összefüggését vizsgájuk. Két változó az a_1, a_2, … a_k, … a_n sorozatnál is megjelenik, nevezetesen az egyik maga az a_k, értéke, ami bármely egész szám lehet, a másik pedig a k index, ami minden esetben szigorúan kötötten, az egymás utáni 1, 2, 3, … értékeket veszi fel. A sorozatoknál tehát a két változó szerepe nem szimmetrikus, ezt a kritikát abban a formában is olvastam, hogy ha sorozatot használunk függvény leírására, akkor ez egyik változó óhatatlanul „hamupipőke szerepbe” kényszerül.
- Még fontosabbak azok az eltérések, amelyek a különbségek vizsgálata tekintetében mutatkoznak. Először is a integer függvények rendszere, mint a 12. szakaszban tárgyaltuk, a teljes különbségi mezővel dolgozik, míg az irodalomban használt $\Delta a_k = a_{k+1} - a_k$ különbségek ennek csak az első, azaz a $D = 1$ differenciaosztályának felelnek meg. Azt, hogy a $D = 1$ differenciaosztály önmagában a függvény változásának a menetéről milyen kevés információt nyújt, a 15. szakasz ábrája, illetve a hozzá fűzött magyarázat mutatja. Viszont (mint a 12. szakaszban említettük) a differenciaosztályokra osztott teljes különbségi mező maradéktalanul leírja a j és i lépéseknek az egymáshoz viszonyított relatív gyakoriságát, azaz az integer függvény változásának menetét.
- Alapvető fontosságú eltérés még az is, hogy a „szokásos”, „aritmetikai” különbség helyett az integer függvények vizsgálatánál mindig „finomítható” különbséget használunk (lásd 13. szakasz), azaz két egymást követő egész számot, nevezetesen az aritmetikai különbséget megelőző egész számot, és magát az aritmetikai különbséget együttesen tekintünk különbségnek az ellentmondásmentes finomíthatóság érdekében. A diszkrét deriváltaknak, amelyek integer függvények formájában fejezik ki a változás menetét, elegendő e két egész szám egyikére illeszkedni, s ez teszi lehetővé, hogy a változás menetének törvényszerűségét jól szemléltető integer deriváltak jelenjenek meg. Tekintsünk vissza még egyszer a 12 szakasz ábrájának osztályderiváltjaira: láthatjuk, hogy a törvényszerűséget jól kifejező integer deriváltak a finomítható különbségeknek hol a kisebbik, hol a nagyobbik, hol pedig mindkét integerére illeszkednek.

Az integer függvények rendszere és a hozzáférhető szakirodalom fenti vázlatos egybevetésének lezárásaként megemlíthető, hogy az integer függvények generálására bemutatott, kizárólag az összeadás/kivonás számítási műveleteit használó, világos rendszert alkotó integer függvényalgoritmusok szerkezetének előzményeit csak a jelen ismertető szerzőjének korábbi publikációiban lehetett találni. Ezeknek a gyökerei részben a fél évszázaddal ezelőtti, kézzel tekerhető mechanikus számológépekkel végzett számítások tapasztalataiban (pl. gyökvonási „trükkjeiben”), részben pedig a számítógépes grafika kezdeti „scan conversion” algoritmusainak az általánosításában kereshetők.

17) Mi lehet az integer függvények bemutatott rendszerének szerepe a továbbiakban?

- Az integer függvények rendszerének kialakulásához vezető vizsgálatok a számítógépes grafika területéről indultak ki. A hasznosítás első lépésében a meglévő, kipróbált program-rendszereket forgalomképes formába lehetne hozni, akár például szabadon formált görbék/felületeket tervezését támogató CAD program-rendszerekként, de akár okos telefon AP-ok is szóba jöhetnek. Ezekben megmutatkoznak az újszerű elvi alapok (pl. nincs trianguláció a görbült felületek kezelésénél, közelítő számítás a metszésgörbéknél, pontosan hozzáilleszthető a számítás a hardver adottságokhoz, optimálisan kihasználva azokat), így terjesztésük egyben az újszerű elvi alapokra is felhívná a széles körű felhasználói tábor figyelmét.

- Mint láttuk, az integer függvények rendszere a biztonságos, garantált pontosságú számításokat is újszerű módon támogatja. Lehetne például egy garantált pontosságú műveleteket megvalósító Excel Add-In-t készíteni, és/vagy ki lehetne dolgozni a Microsoft Developer Network „intsafe” programjainak új, hatékony változatait. Mindehhez is vannak már bizonyos, a jelen rövid összeállításban nem kifejtett kutatási eredmények.

- Ha a téma iránt érdeklődés nyilvánul meg, szóba jöhet az integer függvények rendszerét közvetlenül kiszolgáló hardver megoldások kidolgozása és bevezetése. Az alapalgoritmus hatékony futtatásához már régebben közlésre került egy parallel processzálási lehetőség elvi vázlata, és a biztonságos, garantált pontosságú számítások „intsafe” műveletei is (a lebegőpontosokhoz hasonlóan) elvileg beépíthetők a processzorok mikro-architektúrájába.

- Elvi jelentőségű lehetne, ha az integer függvények rendszerének felhasználásával, továbbfejlesztésével, elterjedésével egyre csökkenne a lebegőpontos aritmetika szerepe a számítógépes gyakorlatban, korrigálva „egy kétes irányba tett lépést” (Neumann J.) a számítógép-tudomány történetében, ezzel kihúzva „a tüskét a számítógép-tudomány körme alól” (W. Kahan).

- És végezetül, persze a legnagyobb általános jelentősége a jelen ismertető címében is feltett kérdésnek van: létrejön-e (akár az integer függvények rendszerét is figyelembe véve, akár más módon) egy, a hagyományos matematikai analízis évszázadok alatt kialakult csodálatos gondolati épületével összemérhető diszciplína az egész számok digitális világában? És ez bekerülhet-e az Erdős Pál által feltételezett, az „igazi” matematikai megfogalmazásokat gyűjtő KÖNYVbe …? Talán igen, hiszen már a jelen szerény ismertetőben is felmerül a két- és többdimenziós integer függvények átfogó, egységes szempontú rendszerezésének lehetősége, amely szinte a helyiértékes számábrázolás megjelenésére emlékeztet a sok esetlegességet tartalmazó korábbi, pl. alfabetikus számírási módszerekkel szemben.