Molnár András

Adaptív Runge–Kutta módszerek

Szakdolgozat

Témavezető:

Fekete Imre

Alkalmazott Analízis és Számításmatematikai Tanszék

Eötvös Loránd Tudományegyetem, Természettudományi Kar
2019

Előszó

Ebben a dolgozatban az egylépéses numerikus módszerek egy híres családjáról, a Runge–Kutta módszerekről lesz szó. A dolgozat célja, hogy a változó lépésközű Runge–Kutta módszerek világának legfontosabb részeit az implementációs vonatkozások szem előtt tartása mellett bemutassa.

Az első fejezeten keresztül az Olvasó hirtelen a Runge–Kutta módszerek világában találhatja magát. A második fejezetben az implicit Runge–Kutta módszerek külön figyelmet igénylő implementációs kérdéseit tanulmányozhatja. A harmadik fejezetben módszereket ismerhet meg a megfelelő lépéshossz megválasztására. A negyedik fejezettel a mai napig leggyakrabban használt lépéshosszválasztó módszert tanulmányozva átkerülhet az adaptív módszerek világába. Az ötödik fejezetben a lépésközválasztás problematikájának tudományos vizsgálatához használható eszközökkel találkozhat, melyekkel felvértezve az utolsó fejezetben lépéshosszválasztó algoritmusok tervezésének lehet tanúja.

Szeretném megragadni az alkalmat, hogy köszönetet mondjak témavezetőmnek, Fekete Imrének a sok támogatásért, a téma felvetéséért, és nem utolsó sorban a dolgozat gondos átnézéséért. Továbbá szeretném megköszönni Gustaf Söderlind professzor úrnak az inspiráló előadásokat, melyeken a dolgozat utolsó része alapul.

Tartalomjegyzék

Tartalomjegyzék **3**

1. Bevezetés **5**
1.1. Néhány alapfogalom . . . 5
1.1.1. Differenciaegyenletek . . . 5
1.1.2. Z-transzformáció . . . 6
1.2. Kezdetiérték-problémák numerikus megoldása . . . 6
1.3. Runge–Kutta módszerek . . . 8
1.3.1. Explicit Runge–Kutta módszerek . . . 9
1.3.2. Implicit Runge–Kutta módszerek . . . 9
1.3.3. Stabilitás . . . 10

2. Az implicit Runge–Kutta módszerek implementálásáról **12**
2.1. Egyszerű iteráció . . . 12
2.2. Newton-iteráció . . . 13
2.3. Egy kétkomponensű iterációs algoritmus . . . 15
2.3.1. Alapgondolatok . . . 15
2.3.2. Váltás módosított Newton-iterációra . . . 15
2.3.3. Váltás egyszerű iterációra . . . 15
2.3.4. Összefoglalás . . . 16
2.4. Numerikus kísérletek . . . 16

3. (Kezdeti) lépésköz választás **19**
3.1. Motiváció . . . 19
3.2. Általános elvek, egyszerű megközelítések . . . 20
3.3. Hairer & Wanner . . . 20
3.4. Arévalo & Söderlind . . . 21

4. Klasszikus adaptív megközelítés **23**
4.1. Hibabecslés . . . 23
4.1.1. Richardson-extrapoláció . . . 23
4.1.2. Beágyazott módszerek . . . 24
4.2. A klasszikus lépésközfrissítő algoritmus . . . 26
4.2.1. Implementációs megfontolások . . . 27
4.3. A klasszikus adaptív algoritmus . . . 28
4.3.1. Numerikus kísérletek . . . 28

5. Az adaptív lépésközválasztás irányításelméleti eszközei **30**
5.1. Irányításelméleti bevezető . . . 30
5.1.1. Egy motiváló példa . . . 30
5.1.2. Vizsgálat z-transzformációval . . . 32
5.2. Digitális szűrők . . . 33
5.3. A lépésközválasztó rendszer . . . 34
5.3.1. Vizsgálat z-transzformációval . . . 34
5.4. A rendszer frekvencia-válasza . . . 36

6. Szabályozók adaptív lépésközválasztáshoz **38**
6.1. Az általános szabályozó struktúra . . . 38
6.1.1. Implementációs megjegyzések . . . 39
6.1.2. Néhány nevezetes speciális eset . . . 39
6.2. Az irányított rendszer rendjei . . . 41
6.3. Szabályozók az aszimptotikus hibamodellre . . . 43
6.3.1. Rendfeltételek . . . 43
6.3.2. A H211 szabályozó család . . . 44
6.4. Szabályozók a dinamikus hibamodellre . . . 48
6.4.1. A dinamikus hibamodell motivációja . . . 48
6.4.2. Egy elsőrendű dinamikus hibamodell . . . 48
6.4.3. Egy alkalmas PI szabályozó . . . 50

7. Konklúzió **53**

Irodalomjegyzék **55**
Cikkek . . . 55
Könyvek . . . 56

1. fejezet

Bevezetés

A bevezető fejezetben a dolgozat megértéséhez különösen fontos fogalmakat gyűjtöttük össze, a terjedelmi korlátokra tekintettel a teljesség igénye nélkül. A fejezet a Runge–Kutta módszerek tömör bevezetésével zárul.

1.1. Néhány alapfogalom

1.1.1. Differenciaegyenletek

A továbbiakban sokszor szükségünk lesz néhány alapvető tényre a differenciaegyenletek elméletéből, így ezeket most ismertetjük.

1.1.1. Definíció (Előretolás-operátor)**.** *Az*

$$E : \mathbb{C}^{\mathbb{N}} \to \mathbb{C}^{\mathbb{N}}; \quad \{x_n\}_{n=0}^{\infty} \mapsto \{x_{n+1}\}_{n=0}^{\infty}$$

leképezést előretolás-operátornak nevezzük.

1.1.2. Definíció (Lineáris differenciaegyenlet)**.** *Legyen P polinom, $f \in \mathbb{C}^{\mathbb{N}}$. Ekkor a*

$$P(E)x = f \tag{1.1}$$

alakú egyenletet lineáris differenciaegyenletnek nevezzük.

Amennyiben $f = 0$, akkor homogén egyenletről, ellenkező esetben pedig inhomogén egyenletről beszélhetünk.

1.1.3. Definíció (Karakterisztikus egyenlet)**.** *Az (1.1) egyenlet karakterisztikus egyenletén a*

$$P(\lambda) = 0 \tag{1.2}$$

egyenletet értjük.

Az elnevezést motiválja a következő két tétel.

1.1.4. Tétel (Homogén egyenlet megoldásai)**.** *A*

$$P(E)x = 0 \tag{1.3}$$

homogén egyenlet megoldásai

$$\{\lambda^k\}_{k=0}^{\infty}$$

alakú exponenciális függvények, ahol $P(\lambda) = 0$.

1.1.5. Tétel (Gyökkritérium)**.** *Amennyiben az (1.2) karakterisztikus egyenlet gyökei legfeljebb egység hosszúak, és az egységhosszú gyökök multiplicitása legfeljebb egy, akkor az (1.3) egyenlet megoldásai korlátosak maradnak.*

1.1.6. Megjegyzés. *A gyökkritérium teljesülése esetén az (1.1) egyenlet megoldásaiban az egyenlet jobboldala által meghatározott dinamika dominál.*

1.1.2. Z-transzformáció

A lineáris irányításelmélet alapvető eszköze a z-transzformáció.

1.1.7. Definíció (Z-transzformált)**.** *Egy adott $x \in \mathbb{C}^{\mathbb{N}}$ sorozat z-transzformáltján az*

$$\hat{x} = \hat{x}(z) = \sum_{n=0}^{\infty} x_k z^{-k}$$

kifejezést értjük.

A z-transzformáció egyik fő előnye az, hogy sorozatokon ható időbeni eltolást a frekvenciatérben egyszerű szorzással reprezentálja a következőképpen:

$$\widehat{Ex} = \widehat{Ex}(z) = \sum_{k=0}^{\infty} x_{k+1} z^{-k} = z \sum_{k=0}^{\infty} x_{k+1} z^{-k-1} + \text{konst} = z\hat{x}(z) + \text{konst} = z\hat{x} + \text{konst}.$$

Az irányításelméletben egy rendszer kezdeti állapota kevésbé érdekes, így fenti gondolatmenetben adódó konstans tagot elhanyagolhatjuk.

Megjegyezzük továbbá, hogy a z változót a szokásoknak megfelelően mi sem fogjuk kiírni amennyiben annak elhanyagolása nem okoz félreértést.

1.2. Kezdetiérték-problémák numerikus megoldása

1.2.1. Definíció (Kezdetiérték-probléma)**.** *Legyen $I \subseteq \mathbb{R}$ intervallum, $f : I \times \mathcal{X} \to \mathcal{X}$ adott függvény, $(t_0, x_0) \in I \times \mathcal{X}$ adott elemek. Ekkor az (f, t_0, x_0) hármas által meghatározott kezdetiérték-probléma a következő egyenletekből álló kifejezés:*

$$\begin{cases} \dot{x}(t) & = f(t, x(t)), \quad t \in I \\ x(t_0) & = x_0. \end{cases}$$

1.2.2. Definíció (Kezdetiérték-probléma megoldása)**.** *A kezdetiérték-probléma megoldásai alatt azon $x \colon I \to \mathcal{X}$ függvényeket értjük, amelyek kielégítik a fenti egyenleteket.*

Numerikus megoldó módszerekre elsősorban akkor van szükség, amikor az adott feladat analitikus megoldása nem ismert. Az alapgondolat az, hogy a megoldás értékét csak véges sok időpontban $\{t_0, t_1, \ldots, t_N\} \subseteq I$ közelítőleg keressük. A dolgozatban ha másként nem jelezzük, feltesszük hogy $t_0 = 0$, $I = [0, T]$, $\mathcal{X} = \mathbb{R}^d$ valamilyen $T > 0$, $d \in \mathbb{N}$ paraméterekkel. Továbbá azt is feltesszük, hogy a vizsgált problémák megoldása egyértelmű.

1.2.3. Definíció (Numerikus megoldás)**.** *A pontos megoldás numerikus közelítését numerikus megoldásnak hívjuk. Ennek értékére t_n időpontban az x_n jelölést fogjuk használni, amire tehát az*

$$x_n \approx x(t_n)$$

összefüggés teljesül.

1.2.4. Definíció (Globális hiba)**.** *A numerikus közelítések hibáját globális hibának hívjuk. Ennek a t_n időponthoz tartozó képlete*

$$e_n = x_n - x(t_n).$$

1.2.5. Definíció (Egylépéses numerikus módszer)**.** *Az egylépéses numerikus módszerek általános alakja*

$$x_{n+1} = \Phi_h(t_n, x_n, x_{n+1}).$$

Azt mondjuk, hogy a módszer **explicit**, ha Φ_h nem függ harmadik változójától. Ellenkező esetben **implicit** módszerről beszélhetünk.

1.2.6. Definíció (Lokális hiba)**.** *A numerikus közelítések lokális hibáját az*

$$l_n = \Phi_h(t_n, x(t_n), x(t_{n+1})) - x_{n+1}$$

formula definiálja. A lokális hiba lépéshosszal lenormált változatát, azaz az l_n/h kifejezést ***egységnyi lépéshosszra jutó lokális hibának*** *nevezzük.*

Amennyiben a t_n diszkrét időpontok egyenletesen helyezkednek el, azaz

$$h_n = t_{n+1} - t_n = h$$

teljesül minden n-re, akkor **állandó lépésközű**, egyébként **változó lépésközű**, vagy **adaptív** módszerről beszélhetünk.

1.2.7. Definíció (Konzisztencia rend)**.** *Amennyiben kellően sima f jobboldal mellett a lokális hibákra $h \to 0$ esetén az*

$$\|l_n\| = \varphi_n h^{p+1} + \mathcal{O}(h^{p+2})$$

összefüggés fennáll, akkor azt mondjuk, hogy a módszer p-edrendben konzisztens. Itt φ_n függ mind a numerikus módszertől, mind f különböző rendű deriváltjain keresztül a megoldandó feladattól.

1.2.8. Definíció (Konvergencia rend)**.** *Amennyiben kellően sima f jobboldal mellett a globális hibákra $h \to 0$ esetén az*

$$\|e_n\| = \mathcal{O}(h^p)$$

képlet teljesül, akkor azt mondjuk, hogy a numerikus módszer p-edrendben konvergens.

A lokális és a globális hibák -és ezáltal a konzisztencia és a konvergencia- fogalmának összekötéséhez szükségünk lesz egy további definícióra.

1.2.9. Definíció (Logaritmikus Lipschitz-konstans)**.** *Az f függvény logaritmikus Lipschitz-konstansát a következő képlet definiálja:*

$$M[f] = \sup_{u \neq v} \frac{\langle u - v, f(u) - f(v) \rangle}{\langle u - v, u - v \rangle}.$$

A differenciálegyenlőtlenségek elméletének felhasználásával megmutatható, hogy a globális hibákra teljesül a következő lemma [18].

1.2.10. Lemma (Globális hibabecslés)**.**

$$\|e_n\| \lessapprox \max_{m \leq n} \frac{\|l_m\|}{h} \cdot \frac{e^{M[f]t_n} - 1}{M[f]}$$

A lemma fontos következménye, hogy az egységnyi lépéshosszra jutó lokális hibák alacsonyan tartásával a globális hiba megfelelően kicsivé tehető. Amennyiben valamilyen módon sikerül elérnünk, hogy ezek a mennyiségek egy adott TOL értékhez közel legyenek, akkor a globális hiba ezzel a tolerancia értékkel arányos lesz, ahol az arányossági tényező csak a feladat jobboldalától és a diszkrét időpontok választásától függ. Ez a **tolerancia-arányosság** elve. Megjegyezzük, hogy olyan feladatok esetén, melyeknél a globális hibában a legutóbbi lokális hiba dominál, a megfelelő pontosság eléréséhez elég a lokális hibát tolerancia közelben tartani. Továbbá, hogy a két megközelítés egyszerre tárgyalható - hiszen a különbség csupán a felmerülő hatványkitevőkben rejlik.

1.3. Runge–Kutta módszerek

1.3.1. Definíció (Runge–Kutta módszer)**.** *Az (s-lépcsős) Runge–Kutta módszerek olyan egylépéses numerikus módszerek, melyek esetében a t_0 időpillanatban az x_0 értékből indulva tett h hosszú lépés képlete a következő:*

$$X_i = x_0 + h\sum_{j=1}^{s} a_{ij} f(t_0 + hc_j, X_j) \qquad i = 1, \dots, s \tag{1.4}$$

$$x_1 = x_0 + h\sum_{i=1}^{s} b_i f(t_0 + hc_i, X_i), \tag{1.5}$$

ahol az $a_{ij}, b_i, c_i \in \mathbb{R}$ értékek a módszer paraméterei.

1.3.2. Megjegyzés. *A fentiek egy kompakt, implementációkban is hasznos átfogalmazása a következő:*

$$X = \mathbb{1} \cdot x_0 + hA \cdot F(X) \tag{1.6}$$

$$x_1 = x_0 + hb \cdot F(X), \tag{1.7}$$

ahol

$$A \in \mathbb{R}^{s\times s}, \quad x_0 \in \mathbb{R}^{1\times N}, \qquad\qquad \mathbb{1} = [1, \dots, 1]^T \in \mathbb{R}^{s\times 1},$$
$$b \in \mathbb{R}^{1\times s}, \quad X \in \mathbb{R}^{s\times N},$$

továbbá

$$F\colon \mathbb{R}^{s\times N} \to \mathbb{R}^{s\times N}$$
$$F_i^J(X) = f^J(t_0 + hc_i, X_i),$$

ahol $i = 1, \dots, s$ és $J = 1, \dots, N$, valamint $(\cdot)$ a közönséges mátrixszorzást jelöli.

1.3.3. Definíció (Butcher-tabló)**.** *A Runge–Kutta módszert egyértelműen meghatározó A mátrixot és b kvadratúravektort egybefoglalóan az 1.1. táblázathoz hasonló formájú, úgynevezett Butcher-tablóban szokás ábrázolni.*

c_1	a_{11}	a_{12}	$\dots$	$a_{1,s-1}$	a_{1s}
c_2	a_{21}	a_{22}	$\dots$	$a_{2,s-1}$	a_{2s}
$\vdots$	$\vdots$				$\vdots$
c_s	a_{s1}	a_{s2}	$\dots$	$a_{s,s-1}$	a_{ss}
	b_1	b_2	$\dots$	b_{s-1}	b_s

1.1. táblázat. A Butcher-tabló

1.3.4. Megjegyzés. *A Runge–Kutta módszerek elméletében szokás a*

$$c_i = \sum_{j=1}^{s} a_{ij} \quad (i = 1, \ldots, s)$$

feltételezéssel élni. Ez biztosítja, hogy a módszer ugyanúgy viselkedjen az egyenlet eredeti, és az $y(t) = [z(t), x(t)]$ *változó bevezetésével*

$$\dot{y}(t) = \tilde{f}(y(t)) = [1, f(z(t), x(t))]$$
$$y(0) = [t_0, x_0]$$

módon autonómmá transzformált változatán.

1.3.1. Explicit Runge–Kutta módszerek

Az explicit Runge–Kutta módszereket Butcher-tablójuk alapján onnan lehet felismerni, hogy a benne szereplő A mátrix szigorú alsóháromszög mátrix. Ezen módszercsalád esetén az (1.4) egyenletrendszer pontos megoldása egyenletenként, iteratív módon meghatározható. Az explicit módszerekre példa az 1.2. táblázatban szereplő klasszikus RK3 és RK4 módszerek. Utóbbit szokás a Runge–Kutta módszerként is emlegetni.

$\frac{1}{2}$	$\frac{1}{2}$		
1	-1	2	
	$\frac{1}{6}$	$\frac{4}{6}$	$\frac{1}{6}$

$\frac{1}{2}$	$\frac{1}{2}$			
$\frac{1}{2}$		$\frac{1}{2}$		
1			1	
	$\frac{1}{6}$	$\frac{2}{6}$	$\frac{2}{6}$	$\frac{1}{6}$

1.2. táblázat. Az RK3 és az RK4 módszer

1.3.2. Implicit Runge–Kutta módszerek

Az implicit Runge–Kutta módszerek az explicit módszerekkel szemben tipikusan jobb stabilitási tulajdonságokkal rendelkeznek, sőt, adott lépcsőszám mellett magasabb rend érhető el velük. Hátrányuk viszont, hogy használatukhoz lényegesen nagyobb számítási kapacitás szükséges, ezért olyan feladatokra szokás implicit módszereket alkalmazni, amelyekre az explicit módszerek valamilyen szempontból nagyon rosszul működnek. Például ilyenek a merev feladatok, amelyekre egyáltalán ahhoz, hogy az explicit módszer konvergálni tudjon, nagyon kicsi lépésközökre van szükség.

A nagy számításigény és jó stabilitási tulajdonságok közötti arany középút megtalálása nem egyszerű. Erről árulkodik például Butcher [3] könyvében a témának szentelt fejezet címe: „Implementálható Implicit Runge–Kutta módszerek“. Megjegyezzük, hogy természetesen minden Runge–Kutta módszer implementálható, a szerző a választott címmel valószínűleg csak a fenti dichotómiára utal.

A kompromisszumos megoldást azon módszerek jelentik, amiket sokáig találóan szemi-implicit néven emlegettek [4]. Manapság a DIRK elnevezés (illetve ennek speciális esetei: SDIRK, ESDIRK, ...) az elterjedtebb [11]. Ezen módszerekben a lépésenként megoldandó (1.4) egyenletrendszer struktúrája lényegesen leegyszerűsödik, és így a megoldás előállítási költségére esetenként nagyságrendekkel jobb becslés adható [3]. Implicit módszerekre például szolgálnak az 1.3.

táblázatban látható implicit Euler és a LobattoIIIC módszerek.

1	1
	1

	$\frac{1}{2}$	$-\frac{1}{2}$
1	$\frac{1}{2}$	$\frac{1}{2}$
	$\frac{1}{2}$	$\frac{1}{2}$

1.3. táblázat. Az implicit Euler és a LobattoIIIC módszerek

1.3.3. Stabilitás

A Runge–Kutta módszerekről sokat megtudhatunk a legegyszerűbb nemtriviális egyenleten, a Dahlquist-féle tesztegyenleten való viselkedésükből.

1.3.5. Definíció (Dahlquist-féle tesztegyenlet)**.** *Adott $\lambda \in \mathbb{C}$ konstans esetén az $f(x) = \lambda x$ függvény által adott egyenletet Dahlquist-féle, vagy lineáris tesztegyenletnek nevezzük.*

A lineáris tesztegyenlet numerikus megoldása során a numerikus lépés az úgynevezett stabilitási függvénnyel reprezentálható.

1.3.6. Definíció (Stabilitási függvény)**.** *A λ paraméterű Dalqhuist-féle tesztegyenlet h lépésközű numerikus megoldása során a numerikus közelítésekre teljesülő*

$$x_{n+1} = R(\lambda h)x_n$$

*képletben az R függvényt a módszer **stabilitási függvényének** hívjuk.*

Megjegyezzük, hogy explicit módszerek esetén a stabilitási függvény polinom, implicit módszerek esetén pedig racionális törtfüggvény.
A módszerek stabilitási tartományát a stabilitási függvény segítségével a következőképpen definiálhatjuk.

1.3.7. Definíció (Stabilitási tartomány)**.** *Az R stabilitási függvényű Runge–Kutta módszer stabilitási tartománya*

$$S = \{z \in \mathbb{C} : |R(z)| \leq 1\}.$$

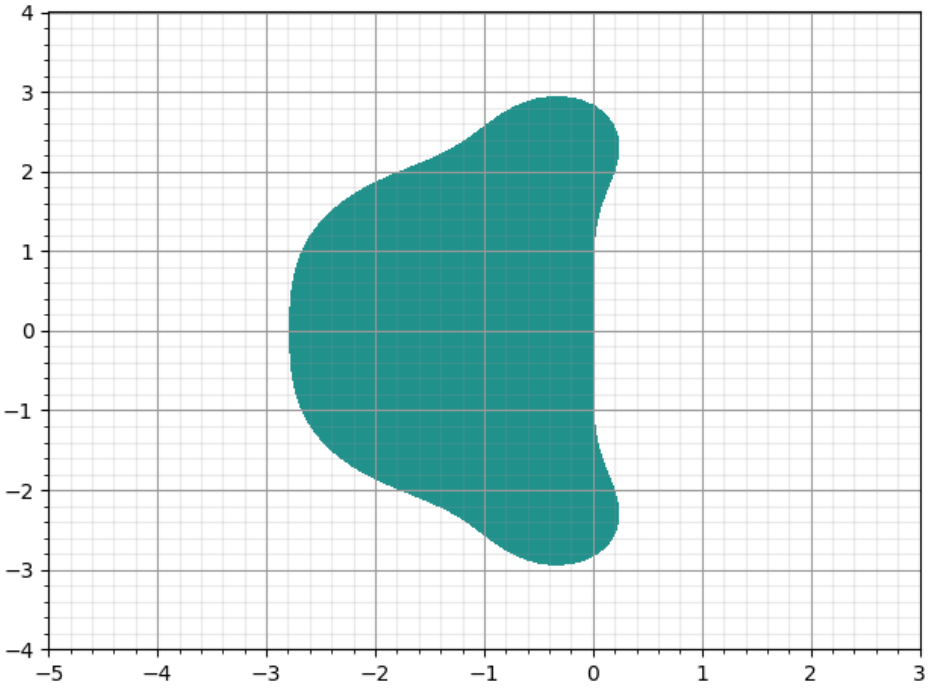
Megjegyezzük, hogy mint azt az 1.1. ábra is szemlétleti, az explicit módszerek stabilitási tartománya korlátos. Ennek oka abban keresendő, hogy stabilitási függvényük polinom.
A módszerekre különböző stabilitási tulajdonságok definiálhatók, ezek közül a dolgozatban az A stabilitás fogalmára lesz szükség.

1.3.8. Definíció (A-stabilitás)**.** *Az S stabilitási tartományú Runge–Kutta módszer A-stabil, amennyiben*

$$\mathbb{C}^- = \{z \in \mathbb{C} : \mathrm{Re}(z) \leq 0\} \subseteq S$$

teljesül.

Megjegyezzük, hogy az A-stabilitás szemléletes jelentése, hogy az ilyen tulajdonságú módszerek a lineáris és egyenletesen negatív monoton jobboldalú autonóm egyenletek esetén időben nem növő normájú numerikus megoldást generálnak. Azaz ilyen egyenletek esetén ha a folytonos megoldás normája monoton csökkenő, akkor a numerikus megoldásé is az lesz.



1.1. ábra. Az RK4 módszer stabilitási tartománya.

2. fejezet

Az implicit Runge–Kutta módszerek implementálásáról

Az implicit Runge–Kutta módszerek implementálásának fő nehézsége a lépésenként felmerülő egyenletrendszer megoldása. Erre a gyakorlatban közelítő módszereket szoktak alkalmazni. A fejezetben bemutatunk két hagyományosnak mondható módszert és egy kombinációs algoritmust, majd az utóbbit numerikus tesztelés alá vetjük.

Alternatív reprezentáció

Kezdjük a fejezetet egy érdekességgel. Az (1.4)-(1.5) egyenletek egy ekvivalens megfogalmazása a következő:

$$k_i = f(t_0 + hc_i, x_0 + h\sum_{j=1}^{s} a_{ij}k_j) \qquad i = 1, \ldots, s \tag{2.1}$$

$$x_1 = x_0 + h\sum_{i=1}^{s} b_i k_i. \tag{2.2}$$

Felmerülhet a kérdés, hogy implementációkban az (1.4) és az (2.1) alakok között van-e érdemi különbség, és ha igen, akkor közülük melyiket érdemes használni.
A numerikus teszteléshez az Octave nyelvet használva, a felmerülő egyenletrendszert az fsolve parancs segítségével megoldva azt tapasztaltuk, hogy a kétféle alak között sem futásidőben, sem pedig pontosságban nincs lényeges különbség.

2.1. Egyszerű iteráció

Az egyenletrendszer megoldásához a kezdetekkor egyszerű fixpont-iterációt használtak [2]. Azaz bevezetve a

$$\Phi\colon \mathbb{R}^{s\times N} \to \mathbb{R}^{s\times N}$$
$$\Phi(X) = \mathbb{1}\cdot x_0 + hA\cdot F(X)$$

függvényt, a közelítések sorozatát a

$$X^{[k+1]} := \Phi(X^{[k]})$$

formula adta, ahol $X^{[0]}$ megfelelően választott érték, például $\mathbb{1}\cdot x_0$.

Ahhoz, hogy az iteráció konvergáljon, kell találni egy normát, amiben Φ kontrakció. Kiindulva a

$$\Phi(X) - \Phi(Y) = hA \cdot (F(X) - F(Y))$$

képletből természetes a gondolat, hogy olyan normát használjunk, melyben F örökli f L második változós Lipschitz-konstansát. Vezessük be az

$$\|X\|_* = \max_{i=1}^{s} \|X_i\|$$

normát [3]. Könnyen látható, hogy eszerint a norma szerint Φ is Lipschitz-folytonos lesz. Valóban, ebben a normában fennáll, hogy

$$\begin{aligned} \|F(X) - F(Y)\|_* &= \max_{i=1}^{s} \|f(X_i) - f(Y_i)\| \\ &\leq \max_{i=1}^{s} L\|X_i - Y_i\| \\ &= L\|X - Y\|_*, \end{aligned}$$

továbbá Φ-re

$$\begin{aligned} \|\Phi(X) - \Phi(Y)\|_* &= \|hA \cdot (F(X) - F(Y))\|_* \\ &\leq h\|A\|_\infty L\|X - Y\|_* \end{aligned}$$

teljesül. Kaptuk tehát, hogy ha

$$\|A\|_\infty hL \leq 1$$

fennáll, akkor Φ a $\|\cdot\|_*$ normában kontrakció lesz, és így az iteráció a Banach-féle fixpont-tétel miatt konvergálni fog.
Látható tehát, hogy alkalmasan kicsi h lépésköz mellett az iteráció konvergálni fog, ráadásul ha L nem túl nagy, akkor ez a lépésközre sem jelent túlzott megszorítást.
A probléma ezzel a módszerrel az, hogy az olyan pontok környezetében, ahol f (lokális) Lipschitz-konstansa nagyon nagy, a konvergencia csak a lépésköz olyan mértékű megszorítása mellett biztosítható, hogy a gyakorlatban a módszer használhatatlan. Ilyen pontok létezése pedig például a merev feladatok osztályára jellemző, amikre a stabilitási tulajdonságaik miatt éppen a jelenleg ismertetett implicit módszereket szeretnénk alkalmazni. Elmondható tehát, hogy egy új módszer után kell nézni.

2.2. Newton-iteráció

Egy másik, a gyakorlatban bevált eljárás a Newton-iteráció. A szükséges formulák ismertetéséhez szükségünk lesz az (1.6)- (1.7) egyenletek alábbi átfogalmazására:

$$X = \mathbb{1} \otimes x_0 + (hA \otimes I_N)F(X) \tag{2.3}$$
$$x_1 = x_0 + h(b \otimes I_N)F(X), \tag{2.4}$$

ahol most

$$A \in \mathbb{R}^{s\times s}, \quad x_0 \in \mathbb{R}^{N\times 1}, \qquad \mathbb{1} = [1, \ldots, 1]^T \in \mathbb{R}^{s\times 1},$$
$$b \in \mathbb{R}^{1\times s}, \quad X \in \mathbb{R}^{sN\times 1}, \qquad I_N \in \mathbb{R}^{N\times N},$$

továbbá

$$F\colon \mathbb{R}^{sN} \to \mathbb{R}^{sN}$$
$$F_i^J(X) := F_{(N(i-1)+J)}(X) = f^J(t_0 + hc_i, X_i),$$

ahol $i = 1, \ldots, s$ és $J = 1, \ldots, N$; valamint $(\otimes)$ a Kronecker-szorzatot jelöli.
Bevezetve az alábbi

$$\Phi\colon \mathbb{R}^{sN} \to \mathbb{R}^{sN}$$
$$\Phi(X) = X - \mathbb{1} \otimes x_0 - (hA \otimes I_N)F(X)$$

függvényt, a Newton-iteráció k-adik lépése a következő alakot ölti:

$$\Phi'(X^{[k]})\Delta^{[k]} = \Phi(X^{[k]}) \tag{2.5}$$
$$X^{[k+1]} = X^{[k]} - \Delta^{[k]}, \tag{2.6}$$

ahol $\Phi'(X) \in \mathbb{R}^{sN \times sN}$ az alábbi blokkmátrix

$$I_s \otimes I_N - h \begin{pmatrix} a_{11}\partial_x f(t_0 + hc_1, X_1) & \ldots & a_{1s}\partial_x f(t_0 + hc_s, X_s) \\ \vdots & & \vdots \\ a_{s1}\partial_x f(t_0 + hc_1, X_1) & \ldots & a_{ss}\partial_x f(t_0 + hc_s, X_s) \end{pmatrix}. \tag{2.7}$$

Mivel a $\partial_x f$ függvény kiértékelése igen költséges, ezért implementációkban célszerű ezek számát minimalizálni. Erre egy lehetőséget nyújt a módosított Newton-iteráció.

Módosított Newton-iteráció

A módszer lényege, hogy a (2.5) egyenletet csak közelítőleg oldjuk meg. Megmutatható, hogy ha a közelítések hibája nem túl nagy, akkor a módszer konvergenciáját ezzel nem rontjuk el.
Mi ezt a módszert olyan formában fogjuk alkalmazni, hogy a pontos $\partial_x f((t_0 + hc_i, X_i))$ értékek helyett az alábbihoz hasonló

$$\partial_x f((t_0 + hc_i, X_i)) \approx \partial_x f(t_0, x_0)$$

közelítésekkel számolunk, így a (2.7) mátrix helyett a jelentősen egyszerűbb struktúrájú

$$M = I \otimes I_N - h(A \otimes \partial_x f(t_0, x_0))$$

mátrixszal dolgozhatunk, amivel az iterációnk az

$$M\Delta^{[k]} = \Phi(X^{[k]}) \tag{2.8}$$
$$X^{[k+1]} = X^{[k]} - \Delta^{[k]} \tag{2.9}$$

alakot ölti.

2.2.1. Megjegyzés. *Implementációkban az M mátrix helyett annak LU-felbontásával érdemes számolni, illetve azt érdemes tárolni.*

2.2.2. Megjegyzés. *Az iteráció kezdőpontjának ebben az esetben is természetes az $X^{[0]} = \mathbb{1} \otimes x_0$ választás, de elmondható, hogy speciális esetekben található alkalmasabb kezdőpont is [10].*

2.3. Egy kétkomponensű iterációs algoritmus

A [8], [12] gondolatait felhasználva ismertetünk egy kétkomponensű algoritmust állandó lépésközű implicit Runge–Kutta módszerek implementálására, ami a fenti gondolatokra és módszerekre épül.
Az algoritmus az integráció során a feladat lokális viselkedésének függvényében választani fog az egyszerű és a Newton-iteráció között. Az alapgondolatok a következők.

2.3.1. Alapgondolatok

- A feladat megoldását mindig egyszerű iterációval kezdjük. Ezzel például teljesen megspóroljuk a Jacobi-mátrix kiértékelését abban az esetben, ha a feladat Lipschitz-konstansa kellően kicsi a választott lépésközünkhöz mérten.
- Ha az egyszerű iteráció már nem konvergál elég gyorsan (például ha a feladat merevvé válik), az egyszerű iterációt le kell váltani módosított Newton-iterációra.
- A Newton-iteráció még módosított formában is igen költséges az egyszerű iterációhoz képest, ezért amikor lehet, célszerű egyszerű iterációt alkalmazni.
- A célból, hogy az algoritmus ne váltogasson a két módszer között szokás egy lehűlési periódust definiálni. Erre egy lehetséges megközelítés, hogy a módosított Newton-módszerre váltás után 10 iterációs lépésig nem váltunk vissza az egyszerű iterációra. Ez többek között azt is biztosítja, hogy az iterációs mátrix kiszámítására nem kerül feleslegesen sor.

2.3.2. Váltás módosított Newton-iterációra

A konvergencia sebességének méréséhez lényegében az iterációink kontraktivitását kell megbecsülnünk. Ehhez használhatjuk a szukcesszív különbségek normáinak hányadosait, illetve ezek maximumát

$$\alpha_k = \frac{\|\Delta^{[k]}\|}{\|\Delta^{[k-1]}\|}, \quad \alpha := \max_k \alpha_k.$$

A heurisztikánk a következő lesz: amennyiben $\alpha_k \leq \frac{1}{2}$, a konvergencia sebessége elég nagy. Ellenkező esetben, a konvergencia nem elég gyors, így a következők szerint fogunk cselekedni.

- Ha eddig egyszerű iterációval haladtunk, akkor át kell váltani módosított Newton-iterációra.
- Ha eddig módosított Newton-iterációt alkalmaztunk, akkor újra kell értékelni a Jacobi-mátrixot.

2.3.3. Váltás egyszerű iterációra

Felmerül a kérdés, hogy amennyiben módosított Newton-iterációt alkalmazunk, és a módszer a fenti becslések szerint jól konvergál, akkor vajon jól konvergálna-e elég gyorsan egyszerű iterációval is. Ennek megállapításához a feladat merevségét fogjuk megbecsülni. Vezessük be az alábbi, ún. merevségi mértéket [12]:

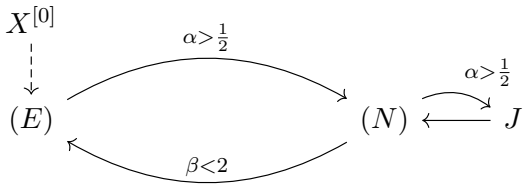
$$\beta = \max_k \frac{\|\Phi(X^{[k]})\|}{\|\Delta_k\|} = \max_k \frac{\|M\Delta_k\|}{\|\Delta_k\|}.$$

A heurisztikánk most az lesz, hogy amennyiben $\beta < 2$, akkor a feladat nem elég merev ahhoz, hogy indokolt legyen a Newton-módszer használata, így visszaváltunk egyszerű iterációra. A $\beta < 2$ választás indoklása az, hogy egyrészt teljesül a $\beta \leq \|M\|$ összefüggés, másrészt megmutatható,

hogy ha M elég jó közelítése az iteráció során használt mátrixnak, akkor a konvergenciából következik, hogy $\|M\| < 2$.

2.3.4. Összefoglalás

A heurisztikát a 2.1. ábra szemlélteti,



2.1. ábra. A kombinált iterációs algoritmus

ahol J a Jacobi-mátrix újraértékelésére, (E) és (N) pedig rendre az egyszerű, illetve a módosított Newton-iterációkra utal.

2.3.1. Megjegyzés. *Egy a fentihez hasonló algoritmus változó lépésköz esetén is használható, azonban ekkor az integrációs és az iterációs lépések irányítását egységesen kell kezelni, ami lényegesen elbonyolítja a problémát, és túlmutat ezen dolgozat keretein.*

2.4. Numerikus kísérletek

Az ismertetett algoritmust a lineáris tesztfeladaton, a Brusselator, és a Van der Pol problémák különböző paraméterezésein próbáltuk ki. Numerikus módszernek az implicit Euler és a Lobatto-IIIC módszereket választottuk (lásd 1.3. táblázat). Az iterációk megállási feltételére egységesen 10^{-7} nagyságú abszolút és relatív hibatoleranciákat szabtunk.
A numerikus kísérleteink azt mutatják, hogy az elv, miszerint a módosított Newton-iterációban fellépő derivált mátrixot

$$I_s \otimes I_N - hA \otimes \partial_x f(t_n, y_n)$$

alakban kellene közelíteni annak ellenére, hogy néhány problémán (pl. kis paraméterű tesztegyenlet) működőképes, merevebb feladatok esetén az iteráció konvergenciáját akadályozza.
A gond ezzel az elvvel ugyanis az, hogy amennyiben ezt alkalmazzuk, akkor egy adott integrációs lépésben az iterációnk kontraktivitását csupán a lépésköz méretének változtatásával tudjuk befolyásolni, így amennyiben az állandó lépésköz nincs egy bizonyos (feladatfüggő) határ alatt, akkor az integráció során lesz olyan pont, amikor a módosított Newton-iteráció elszáll.
Ezt az elvet tehát annyiban módosítottuk, hogy amennyiben az algoritmus szerint szükség van a Jacobi-mátrix újraértékelésére, akkor azt megtesszük az aktuális $X^{[k]}$ iterált értékének felhasználásával.
Az algoritmus ezzel a módosítással már lényegesen jobban működött. Sikerült olyan paraméterezésű feladatokra is használni, amikre a beépített fsolve egyáltalán nem konvergált.
Kérdéses azonban, hogy mennyit nyertünk a két fázisú algoritmus használatával. A kérdés megválaszolásához és az algoritmus további megismeréséhez vizsgáljuk az

$$f(0) = 0, \quad f'(x) = \begin{cases} \frac{1}{5} & x \le 1 \\ \frac{1}{5}(2-x) - 50(x-1) & 1 < x \le 2 \\ -50 & 2 < x \end{cases}$$

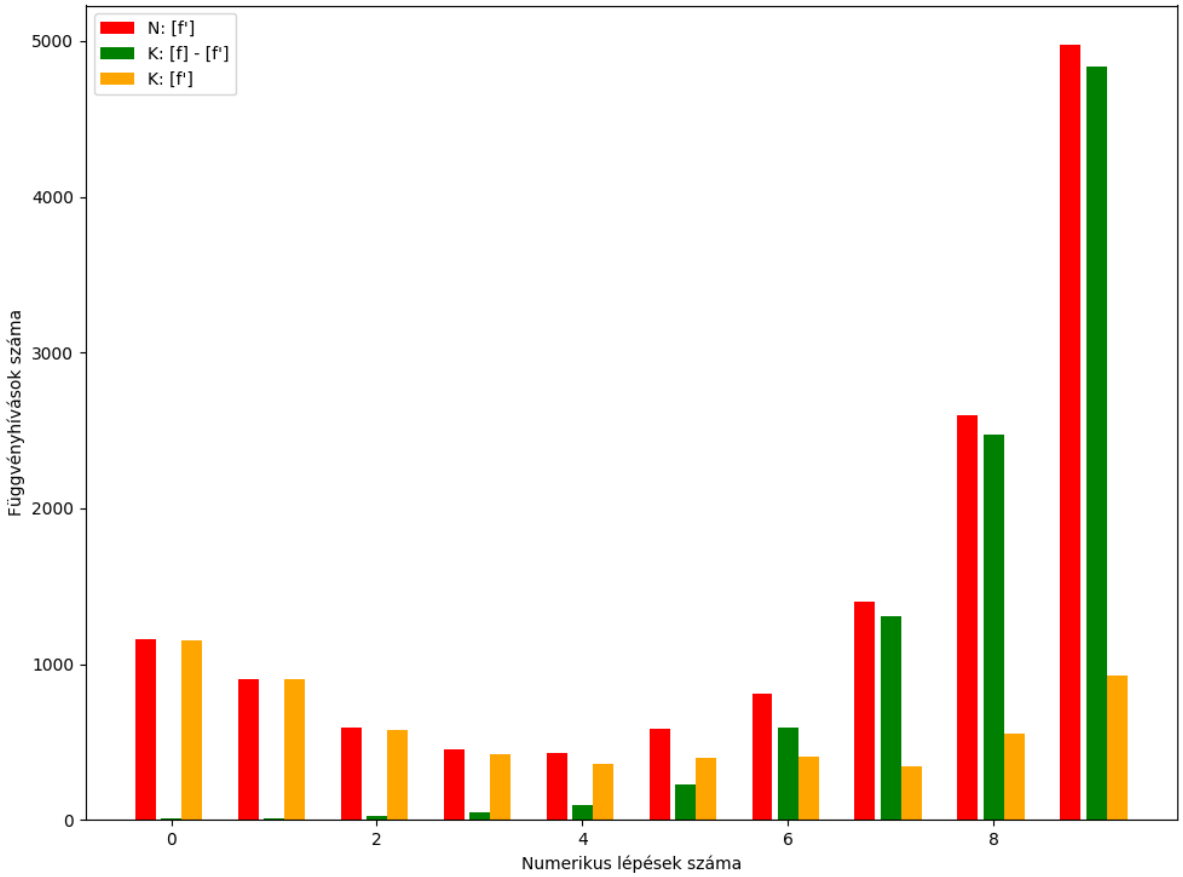
függvény segítségével definiált,

$$\begin{cases} \dot{x}(t) = f(x(t)), & t \in [0,5] \\ x(0) = \frac{1}{2} \end{cases}$$

problémát. A feladat választását az ihlette, hogy olyan problémát vizsgáljunk, ami a tér egyik részén merev, a másikon pedig nem.
Ezt a feladatot implicit Euler módszerrel oldottuk meg kétféleképpen. Az egyenletrendszer megoldására először közönséges Newton-iterációt, majd a kombinált iterációs módszert alkalmaztuk egyre csökkenő h lépésköz mellett; pontosabban, a numerikus integrációs lépések számát $5 \cdot 2^k$-nek választottuk, ahol $k = 0, \ldots, 10$. A kapott eredményeket, azaz a megtörtént függvényhívások ([f], $[\partial_x f]$) számát a 2.1. táblázat, illetve a 2.2. ábra mutatja.

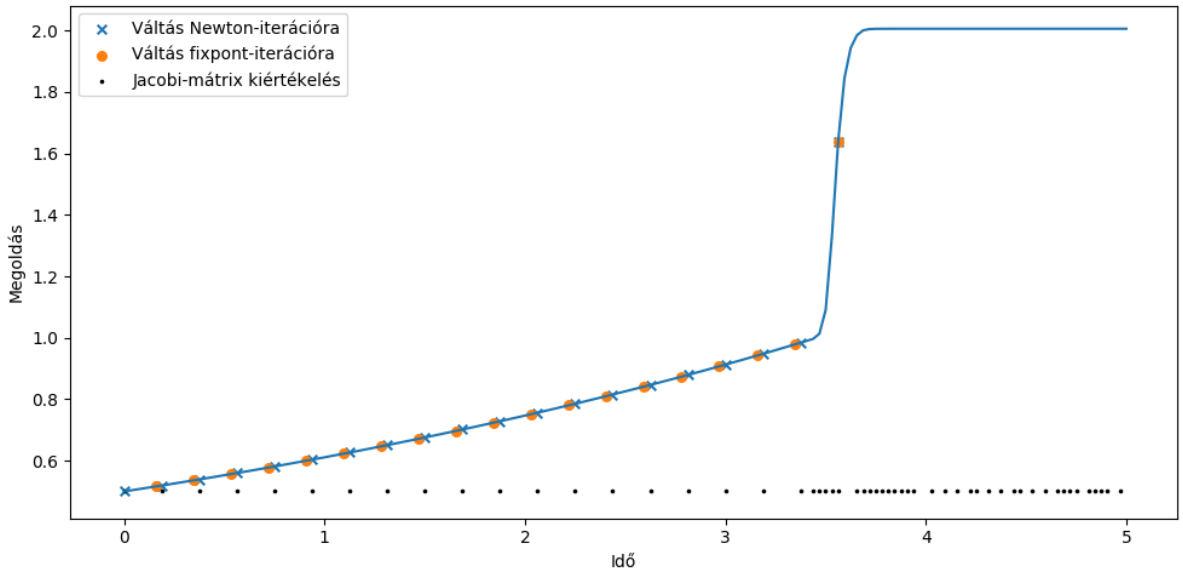
	Num. lépések	**5**	**10**	**20**	**40**	**80**	**160**	**320**	**640**	**1280**	**2560**
N	$[\partial_x f]$	1157	905	589	456	432	587	809	1398	2598	4980
K	$[\partial_x f]$	1156	902	576	419	361	400	408	343	557	926
	$[f] - [\partial_x f]$	8	11	22	51	95	231	589	1309	2475	4833

2.1. táblázat. Függvényhívások számának összehasonlítása



2.2. ábra. Függvényhívások számának összehasonlítása. A vízszintes tengely mentén az $5 \cdot 2^k$ alakú numerikus lépésszámot, a függőleges tengely mentén pedig az ehhez tartozó függvényhívások száma látható.

A feladat 160 lépéssel történő megoldását a 2.3. ábrán külön kiemeltük. Az ábra az algoritmus tipikus viselkedését jól szemlélteti, ugyanis néhány speciális feladattól eltekintve az algoritmus főleg módosított Newton-iterációs lépéseket tesz. Amennyiben a feladat engedi, akkor esetenként pár lépés erejéig átvált fixpont-iterációra, amit jellemzően a Jacobi-mátrix újraértékelése követ. Az ábra azt is mutatja, hogy mint azt elvárnánk, a mátrix újraértékelésére a hirtelen változások környékén van főleg szükség.



2.3. ábra. Váltás az iterátorok között. Az ábra a feladat 160 lépéssel történő megoldását mutatja. A pontokat, ahol váltás történt Newton-iterációra x-szel, míg azokat, ahol egyszerű iterációra váltott az algoritmus körrel jelöltük. A Jacobi-mátrix újraértékelésének sűrűségét az ábra alján látható fekete pontok jelzik.

3. fejezet

(Kezdeti) lépésköz választás

Egy kezdetiérték-probléma numerikus megoldása során célunk az, hogy a probléma analitikus megoldását diszkrét pontokban egy általunk választott hibahatáron, tolerancián belül megközelítsük.
Ha a módszerünk konvergens, akkor tetszőlegesen kicsi tolerancia esetén létezik olyan h lépésköz, amivel a módszer által adott numerikus megoldás hibája az adott tolerancia alatt lesz.
Ebben a fejezetben, ami végül átvezet majd a változó lépésköz világába, arról lesz szó, hogy a gyakorlatban ezt a lépésközt hogyan érdemes algoritmikusan megválasztani.

3.1. Motiváció

Azt, hogy ez a feladat korántsem triviális, alátámasztják az alábbi gondolatok.

- Ha a lépésköz nagyon kicsi, akkor bár a megoldás kellően pontos lesz, annak elkészítése nagyon sokáig tarthat.
- Ha a lépésközt nem választjuk elég kicsire, akkor esetenként előfordul, hogy a módszerünk használhatatlan megoldást produkál. Például merev feladatok esetén ha explicit módszereket használunk, akkor az iteráció során kilépünk a módszer stabilitási tartományából és a megoldás elszáll. Explicit módszerekre a legnagyobb megengedett lépésköz egy közelítése a referencia időskála [22].
- A megfelelő lépésköz természetesen módszerenként, feladatonként, és kezdeti értékenként is változik.

A kezdeti lépésköz megválasztásának fontossága állandó lépésköz esetén nem igényel további magyarázatot. Ami talán meglepőbb, az az, hogy változó lépésközű módszereknél is egy kritikus feladatról van szó, hiszen természetes az a gondolat, hogyha rendelkezésünkre áll egy lépésköz állító módszer (**szabályozó**), akkor ha esetleg h_0 választását el is rontjuk, ez a módszer majd korrigálja a hibánkat. Ezzel a gondolattal a fő probléma az, hogy az első lépés utáni hibát a szabályozó már nem tudja befolyásolni, és az így keletkezett hiba már végig a megoldóval marad. Ehhez hasonló, illetve stabilitási megfontolásokból szokás inkább finomabb kezdeti lépésközt választani, azonban ekkor legalábbis az első néhány lépésben feleslegesen sok erőforrást fogunk a számításba fektetni, illetve ha az a célunk, hogy a globális hiba egyenletes legyen, akkor nem választhatunk túl kicsi kezdeti lépésközt, hiszen ekkor az integrációs tartomány első szakaszában aránytalanul pontos lesz a megoldás.

3.2. Általános elvek, egyszerű megközelítések

Természetesen a legegyszerűbb megoldás az, ha a felhasználót orákulumnak tekintjük és megkérjük, hogy adjon nekünk egy elég jó lépésközt. Persze ez a módszer minden feladatra realisztikusan nem fog működni. Arról, hogy a felhasználót érdemes-e egyáltalán megkérdezni, a szakirodalomban nincs megegyezés [25],[7]. A legjobb kompromisszumnak az tűnik, hogy a felhasználót megkérdezzük, de elvárjuk tőle, hogy csak akkor adjon választ, amennyiben tudja, hogy az adott kezdeti lépésköz jól fog működni az adott feladatra. Amennyiben nem tud válaszolni, akkor pedig az alábbi automatikus heurisztikák egyikére hagyatkozunk.
Talán a legegyszerűbb automatizált megoldás az, ha az elvégzendő lépések N számát tekintjük rögzítettnek, és a lépésközt ezzel fordítottan arányosnak választjuk. Ez a megközelítés a feladatról lényegében semmilyen információt nem használ ki. A másik véglet a pontos megoldás lokális ismerete alapján történő döntéshozás, de ez nyilván nem realisztikus.
A kettő között helyezkedik el az a bevett szokás, hogy a megoldás t_0-beli alacsonyabb rendű Taylor-polinomjának különböző módszerekkel történő közelítéséből először számítanak egy (általában durva) hibabecslést, majd abból kezdeti lépésközt.
A hibabecsléshez használt kulcsgondolat, hogy amennyiben a megoldás m-edfokú Taylor-polinomja

$$x(t_0 + h) \approx x(t_0) + h\dot{x}(t_0) + \frac{h^2}{2}\ddot{x}(t_0) + \ldots + \frac{h^m}{m!}x^{(m)}(t_0),$$

a hozzátartozó hibatag pedig

$$E_m = \frac{h^{m+1}}{(m+1)!}x^{m+1}(t_0),$$

akkor $m \leq p$ esetén tetszőleges legalább p-edrendű egylépéses módszerre a lokális hiba normája közelíthető az

$$\|E_m\|^{\frac{p+1}{m+1}}$$

kifejezéssel [15].
A közelítésekhez jellemzően használt másodrendű Taylor-polinom első két együtthatója olcsón, általában további számítások nélkül adja magát, hiszen x_0 értékén túl szinte minden numerikus módszer felhasználja az $\dot{x}(t_0) = f(t_0, x_0)$ értéket is.
A második derivált becslése problematikusabb, ehhez az adott feladat valamivel mélyebb ismeretére van szükség. Az erre irányuló számos különböző eljárásban közös vonás, hogy felhasználják a megoldás egy t_0-hoz közeli, t_1 pontbeli értékét, vagy legalábbis annak egy közelítését. Ezt a közelítést természetesen alacsony számításigénnyel, és ezzel együtt relatíve precízen kell tudni elvégezni, így ehhez általában egy explicit módszerrel szokás egy nagyon finom lépést tenni.
Most következzen két, a szakirodalomban elfogadott, a gyakorlatban bevált módszer ismertetése, melyek a fent taglalt elvek mentén különböző heurisztikákat alkalmaznak.

3.3. Hairer & Wanner

A szerzők a [7] cikk által inspirálva, az ott szereplő módszernél egyszerűbb, de állandó lépésközű megoldókhoz is használható algoritmust írnak le. Bevezetnek abszolút, illetve relatív hibatolerancia vektorokat, amik segítségével az egyes koordináták hibáira külön-külön szabható korlát.
A deriváltak nagyságának méréséhez az euklideszi norma helyett bevezetnek egy súlyozott négyzetes közép normát:

$$\|z\| = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{z_i}{w_i}\right)^2},$$

ahol a súlyokat az abszolút illetve relatív hibatoleranciákból keverik ki a következő módon:

$$w_i = \mathrm{ATOL}_i + \mathrm{RTOL}_i \, |x_0|_i.$$

A javasolt algoritmus tehát a következő.

1. Első lépésben a fentiek felhasználásával kiszámítják az első két derivált hosszát:

$$d_0 = \|x_0\|,$$
$$d_1 = \|f(t_0, x_0)\|.$$

2. A feladat további megismeréséhez a

$$h_1 = \begin{cases} 10^{-6}, & \text{ha } \min(d_0, d_1) < 10^{-5} \\ 10^{-2}\frac{d_0}{d_1}, & \text{egyébként} \end{cases}$$

   lépésközt választják, és ezzel tesznek egy explicit Euler lépést,

$$x_1 = x_0 + h_1 f(t_0, x_0).$$

3. A kapott közelítés segítségével a

$$d_2 = \frac{\|f(t_0 + h_1, x_1) - f(t_0, x_0)\|}{h_1}$$

   formulából számolnak egy becslést a második derivált nagyságára.

4. Az első két derivált hossza közül a nagyobbikkal dolgozva a következő módon definiálnak egy h_2 lépéshosszt:

$$h_2^{p+1} = \begin{cases} \max(10^{-6}, h_1 \cdot 10^{-3})^{p+1}, & \text{ha } \max(d_1, d_2) \leq 10^{-15} \\ 10^{-2} \max(d_1, d_2)^{-1}, & \text{egyébként.} \end{cases}$$

5. Végezetül a nagyságrendek függvényében a kezdeti lépéshosszt a

$$h_0 = \min(100 \cdot h_1, h_2)$$

   képlettel választják.

3.4. Arévalo & Söderlind

A szerzők a 2017-es cikkükben [1] egy kétkomponensű κ arányossági tényezőt javasolnak, amivel a kezdeti lépésköz

$$h_0 = \min\left(\kappa \cdot h \, \mathrm{TOL}^{\frac{1}{k}}, 10^{-3} H\right),$$

ahol κ a κ_p precízióért, és κ_s stabilitásért felelős tényezők számtani közepe:

$$\kappa = \frac{\kappa_s + \kappa_p}{2},$$

h egy finom, feladatspecifikus lépésköz, H az integrációs út hossza, amivel arányosan maximalizálták a választható lépésközök hosszát, végül $k \in \{p, p+1\}$ attól függően, hogy a lokális hibát vagy annak az egységnyi lépéshosszra jutó változatát szeretnénk irányítani.
A κ és h értékek számításához szükség van a feladat lokális feltérképezésére, erre a következő háromlépéses eljárást javasolják.

1. Lássuk el az x_0 kezdeti értéket egy apró véletlenszerű ξ perturbációval, majd ebből készítsük el f lokális Lipschitz-konstansának egy durva alsó becslését:

$$L_0 = \frac{\|f(t_0, x_0 + \xi) - f(t_0, x_0)\|}{\|\xi\|}.$$

2. L_0 segítségével definiáljuk úgy a h lépésközt, hogy azzal a következő lépésben használt explicit módszer nagy valószínűséggel stabil legyen. Ehhez szükséges, hogy a lépésközre $h \cdot L[f] \lessapprox 1$ teljesüljön, ahol $L[f]$ az f Lipschitz-konstansa [20]. Figyelembe véve, hogy L_0 egy alsó becslés, egy lehetséges választás például a következő:

$$h = \frac{1}{10L_0}.$$

3. Az új, másodrendben közelítő $\tilde{x}_0$ értéket két explicit Euler lépés egymásutánjával a következő módon számolják:

$$\begin{aligned} x_1 &= x_0 + hf(t_0, x_0), \\ \tilde{x}_0 &= x_1 - hf(t_0 + h, x_1). \end{aligned}$$

A pontosságért felelős tényezőre a bevezetőben leírt elv alapján innen adódik a

$$\kappa_p = \frac{1}{\sqrt{\|x_0 - \tilde{x}_0\|}}$$

képlet. A stabilitásért felelős tényező tartalmaz egy javított becslést a Lipschitz-konstansra, illetve egy becslést az M logaritmikus normára:

$$\kappa_s = \frac{1}{h(L + M/2)}, \quad \text{ahol} \quad L = \frac{\|f(t_0, \tilde{x}_0) - f(t_0, x_0)\|}{\|\tilde{x}_0 - x_0\|}, \quad M = \frac{(\tilde{x}_0 - x_0) \cdot (f(t_0, \tilde{x}_0) - f(t_0, x_0))}{\|\tilde{x}_0 - x_0\|^2}.$$

4. fejezet

Klasszikus adaptív megközelítés

A fejezet célja, hogy ismertesse azt, a szakirodalomban sokáig egyeduralkodónak számító eljárást, amivel még a mai napig sok adaptív kód működik. Az eljárás megismeréséhez először szükségünk lesz olyan módszerekre, amikkel a lokális hibát becsülhetjük, majd pedig olyanokra, amikkel ezen hibabecslések alapján módosítani tudjuk az iterációban használt lépésközt. A fejezet felépítése a [9] könyv II.4. részét követi.

4.1. Hibabecslés

Ebben a részben két, a gyakorlatban sokat használt hibabecslő eljárásról lesz szó, nevezetesen a Richardson-extrapolációról, és a beágyazott Runge–Kutta módszerekről.

4.1.1. Richardson-extrapoláció

A Richardson-extrapoláció a számításmatematika egy hasznos módszere, mellyel sok numerikus módszer esetén alkalmunk nyílik a módszer rendjét eggyel növelni. Ebben a részben megmutatjuk, hogy ez a Runge–Kutta módszerek esetén is megtehető, illetve azt is, hogy mindeközben hogyan nyerhetünk egy hibabelcsést is.
A módszerről általánosságban részletesebben például a [13] cikkben olvashatunk, de az alapötlet az alábbiakból is kiolvasható. Tegyük fel, hogy adott egy kezdetiérték-problémánk

$$x(t_0) = x_0$$

kezdeti feltétellel, amit p-edrendű Runge–Kutta módszerrel oldunk meg, továbbá legyen h egy adott lépéshossz. Tegyünk egy $2h$ hosszú lépést, kapva az $\tilde{x}_2$ értéket. Ennek a közelítésnek a hibája

$$\tilde{e}_2 = x(t_0 + 2h) - \tilde{x}_2 = C(x_0) \cdot (2h)^{p+1} + \mathcal{O}(h^{p+2}). \tag{4.1}$$

Térjünk vissza a kezdeti értékünkhöz és lépjünk most két h hosszú lépést. Ekkor az első közelítés hibája a korábbihoz hasonlóan

$$e_1 = x(t_0 + h) - x_1 = C(x_0) \cdot h^{p+1} + \mathcal{O}(h^{p+2}).$$

A második lépés hibája két komponensből tehető össze. Az egyik a második lépés lokális hibája,

$$C(x_1) \cdot h^{p+1} + \mathcal{O}(h^{p+2}) = (C(x_0) + \mathcal{O}(h)) \cdot h^{p+1} + \mathcal{O}(h^{p+2}),$$

ahol a második egyenlőség az elemi differenciák simasága miatt teljesül. A másik pedig az első lépés hibájának továbbterjedéséből adódik, értéke pedig Taylor-sorba fejtéssel láthatóan

$$(I + \mathcal{O}(h)) \cdot e_1.$$

Összességében így a hiba a következő képlettel írható le:

$$\begin{aligned} e_2 &= x(t_0+2h) - x_2 = (I + \mathcal{O}(h))\, C(x_0)\cdot h^{p+1} + (C(x_0)+\mathcal{O}(h))\cdot h^{p+1} + \mathcal{O}(h^{p+2}) \\ &= 2C(x_0)h^{p+1} + \mathcal{O}(h^{p+2}). \end{aligned} \tag{4.2}$$

A fenti (4.1) és (4.2) egyenlőségeket egymásból kivonva, az $\mathcal{O}(h^{p+2})$ tagot elhanyagolva a $C(x_0)\cdot h^{p+1}$ tag kifejezhető, és ezáltal a következő tétel adódik.

4.1.1. Tétel. *Legyen adott egy p-edrendű Runge–Kutta módszer. Jelölje rendre x_2, illetve $\tilde{x}_2$ azt a pontot, amit a módszerrel kétszer h, illetve egyszer $2h$ hosszút lépve kapunk. Ekkor fennáll a következő*

$$x(t_0+2h) - x_2 = \frac{x_2 - \tilde{x}_2}{2^p - 1} + \mathcal{O}(h^{p+2})$$

hibabecslés, továbbá

$$\hat{x}_2 = x_2 + \frac{x_2 - \tilde{x}_2}{2^p - 1}$$

egy $p+1$-edrendű közelítése az $x(t_0+2h)$ értéknek.

4.1.2. Beágyazott módszerek

A beágyazott módszerek alapötlete az, hogy olyan Runge–Kutta módszerek párjaival dolgozunk, amelyekre strukturális hasonlóságukból kifolyólag igaz, hogy az első módszerrel történő lépés közben felmerülő értékek felhasználásával a második módszerrel történő lépés költsége alacsony. Tipikusan olyan módszereket szokás használni, amelyek páronként közös a_{ij}, de különböző $b_i, \hat{b}_i$ együtthatókkal rendelkeznek. Az ilyen párokat a 4.1. táblázathoz hasonló kiegészített Butcher-tablóval szokás jelölni.

c_1	a_{11}	a_{12}	$\dots$	$a_{1,s-1}$	a_{1s}
c_2	a_{21}	a_{22}	$\dots$	$a_{2,s-1}$	a_{2s}
$\vdots$	$\vdots$				$\vdots$
c_s	a_{s1}	a_{s2}	$\dots$	$a_{s,s-1}$	a_{ss}
	b_1	b_2	$\dots$	b_{s-1}	b_s
	$\hat{b}_1$	$\hat{b}_2$	$\dots$	$\hat{b}_{s-1}$	$\hat{b}_s$

4.1. táblázat. Beágyazott Runge–Kutta módszer

A párokat úgy szokás választani, hogy az adódó

$$x_1 = x_0 + h\sum_{i=1}^{s} b_i k_i \quad p\text{-edrendű}$$

illetve

$$\hat{x}_1 = x_0 + h\sum_{i=1}^{s} \hat{b}_i k_i \quad \hat{p}\text{-odrendű}$$

közelítések rendjére fennálljon a $\hat{p} = p \pm 1$ összefüggés. Világos, hogy ekkor ezekre fennáll, hogy

$$\begin{aligned} x(t_0+h) - x_1 &= C\cdot h^{p+1} + \mathcal{O}(h^{p+2}) \\ x(t_0+h) - \hat{x}_1 &= \hat{C}\cdot h^{\hat{p}+1} + \mathcal{O}(h^{\hat{p}+2}), \end{aligned}$$

ezért amennyiben az algoritmus során az alacsonyabb rendű közelítéssel lépünk tovább, a keletkező hiba egy aszimptotikusan korrekt becslése az

$$x_1 - \hat{x}_1 = C \cdot h^{p+1} - \hat{C} \cdot h^{\hat{p}+1} + \mathcal{O}(h^{\min(\hat{p}+2,p+2)})$$

kifejezés.
Valóban, sajnos ez a módszer nem sokat mond a magasabb rendű közelítés hibájáról. Hiába természetes hát az a gondolat, hogyha már rendelkezésünkre áll egy pontosabb közelítés, akkor használjuk azt a továbblépéshez, a fenti okokból kifolyólag ez esetben elveszítjük az aszimptotikusan korrekt hibabecslésünket, ezért ezt vannak akik nem javasolják. A dolog érdekessége, hogy ennek ellenére mások tapasztalatai alapján [5], [14] az alkalmazásokban mégis a magasabb rendű közelítéssel érdemes tovább dolgozni. Ezt a megközelítést lokális extrapolációnak hívják.
Most következzen két híres beágyazott módszer.

Runge–Kutta–Fehlberg

A 4.2. tablójú klasszikus Runge–Kutta-Fehlberg [6] módszerben a lejjebb elhelyezkedő b súlyokkal negyedrendű, míg a felette lévő b súlyokkal ötödrendű közelítést kapunk. A szerzők az együtthatókat úgy választották, hogy minimalizálják a negyedrendű közelítés hibáját.

$\frac{1}{4}$	$\frac{1}{4}$					
$\frac{3}{8}$	$\frac{3}{32}$	$\frac{9}{32}$				
$\frac{12}{13}$	$\frac{1932}{2197}$	$-\frac{7200}{2197}$	$\frac{7296}{2197}$			
1	$\frac{439}{216}$	-8	$\frac{3680}{513}$	$-\frac{845}{4104}$		
$\frac{1}{2}$	$-\frac{8}{27}$	2	$-\frac{3544}{2565}$	$\frac{1859}{4104}$	$-\frac{11}{40}$	
	$\frac{16}{135}$		$\frac{6656}{12825}$	$\frac{28561}{56430}$	$-\frac{9}{50}$	$\frac{2}{55}$
	$\frac{25}{216}$		$\frac{1408}{2565}$	$\frac{2197}{4104}$	$-\frac{1}{5}$	

4.2. táblázat. A Runge–Kutta-Fehlberg módszer

Dormand–Prince

A 4.3. tablójú legendás Dormand-Prince [5] módszer szintén egy negyedrendű és ötödrendű módszerekből álló pár, ahol a pontosabb közelítést a felső b súlyok használatával kapjuk.
A fenti módszerrel ellentétben az együtthatókat ez esetben úgy választották, hogy lokális extrapoláció esetén minimalizálják a hibát. Láthatóan ez a módszer FSAL (First Same As Last) tulajdonságú, azaz az A mátrix utolsó sora megegyezik az egyik kvadratúravektorral, így lépésenként a 7 lépcső ellenére is csak 6 függvényevaluációra van szükség. Érdemes megyjegyezni,

hogy a gyakorlatban ezt a módszert használja a MATLAB ode45 megoldóprogramja.

$\frac{1}{5}$	$\frac{1}{5}$						
$\frac{3}{10}$	$\frac{3}{40}$	$\frac{9}{40}$					
$\frac{4}{5}$	$\frac{44}{45}$	$-\frac{56}{15}$	$\frac{32}{9}$				
$\frac{8}{9}$	$\frac{19372}{6561}$	$-\frac{25360}{2187}$	$\frac{64448}{6561}$	$-\frac{212}{729}$			
1	$\frac{9017}{3168}$	$-\frac{355}{33}$	$\frac{46732}{5247}$	$\frac{49}{176}$	$-\frac{5103}{18656}$		
1	$\frac{35}{384}$		$\frac{500}{1113}$	$\frac{125}{192}$	$-\frac{2187}{6784}$	$\frac{11}{84}$	
	$\frac{35}{384}$		$\frac{500}{1113}$	$\frac{125}{192}$	$-\frac{2187}{6784}$	$\frac{11}{84}$	
	$\frac{5179}{57600}$		$\frac{7571}{16695}$	$\frac{393}{640}$	$-\frac{92097}{339200}$	$\frac{187}{2100}$	$\frac{1}{40}$

4.3. táblázat. A Dormand-Prince módszer

4.2. A klasszikus lépésközfrissítő algoritmus

Az első igen elterjedt lépésközválasztó módszer [9], amit még a mai napig számtalan alkalmazásban használnak, egy egyszerű multiplikatív eljárás, aminek alapötletét a következő gondolatok tartalmazzák.
Egyszerűség kedvéért most egy abszolút hibahatárral (TOL) fogunk dolgozni, azaz azt követeljük meg, hogy a lokális hiba hossza valamilyen normában (például euklideszi) TOL alatt maradjon,

$$\|l\| = C \cdot h^{p+1} + \mathcal{O}(h^{p+2}) \leq \text{TOL}. \tag{4.3}$$

Tegyük fel, hogy a feladat integrálása során a legutóbbi lépésünk hossza h_{n-1} volt, ekkor tehát keressük egy h_n hosszt, amivel a fenti egyenlőtlenség teljesül.
Természetesen ha találunk egy ilyen h_n értéket, akkor minden nála kisebb pozitív hossz is megfelelő lesz. Számításigényi megfontolásokból célszerű a lehető legnagyobb ilyen h_n értéket választani, hiszen ha nekünk csak TOL pontosságra van szükségünk, akkor felesleges ennél pontosabban számolni, illetve így elérhetjük azt is, hogy a lokális hibák hossza egyenletesen TOL körül helyezkedjen el.
Tehát keressük a h_n értéket úgy, hogy

$$\text{TOL} \approx C \cdot h_n^{p+1}$$

teljesüljön. Felhasználva a fentihez hasonló

$$\|l_n\| \approx C \cdot h_{n-1}^{p+1}$$

hibabecslést, adódik a

$$\rho_{n-1} = \frac{h_n}{h_{n-1}} = \left(\frac{\text{TOL}}{\|l_n\|}\right)^{\frac{1}{p+1}}$$

tényező, aminek segítségével elérkeztünk a multiplikatív formulához:

$$h_n = \rho_{n-1} h_{n-1}.$$

Amennyiben szeretnénk -akár koordinátánként különböző- abszolút és relatív korlátokat szabni, azaz például ha szeretnénk, hogy az x_0 pontból indulva x_1 közelítéssel élve a hibára az

$$|l_i| \leq w_i = \mathrm{ATOL}_i + \max(|(x_0)_i|, |(x_1)_i|) \cdot \mathrm{RTOL}_i$$

becslés teljesüljön, ahol az ATOL, RTOL vektorok tartalmazzák az általunk előírt hibahatárokat, akkor arra lehetőségünk nyílik például úgy, hogy a fenti gondolatmenetben az alábbi w_i súlyokkal súlyozott négyzetes közép normát használjuk:

$$\|x\| = \sqrt{\frac{1}{n}\sum_{i=1}^{n}\left(\frac{x_i}{w_i}\right)^2}.$$

Ezzel a normával a megfelelő hibabecslések az

$$\|l_n\| \approx C \cdot h_{n-1}^{p+1}, \quad 1 \approx C \cdot h_n^{p+1}$$

alakot öltik, és az új lépést adó formula a

$$h_n = \rho_{n-1} \cdot h_{n-1} = \left(\frac{1}{\|l_n\|}\right)^{\frac{1}{p+1}} \cdot h_{n-1}.$$

4.2.1. Implementációs megfontolások

A klasszikus megközelítés hátránya, hogy az alkalmazásokban csak sok biztonsági kapcsolóval együtt használható. Erre példa az alábbi heurisztika szükségessége. Induljunk ki a fenti formulából, azaz legyen

$$\rho_{n,0} = \left(\frac{1}{\|l_n\|}\right)^{\frac{1}{p+1}}.$$

Mivel a hibák számítására nem pontos formulákat, hanem csak becsléseket alkalmaztunk, vezessünk be egy m biztonsági tényezőt. Általában $m \in \{0.8, 0.9, (0.25)^{\frac{1}{p+1}}, (0.38)^{\frac{1}{p+1}}\}$ választásokkal szokás élni. Ezzel legyen

$$\rho_{n,1} = m \cdot \rho_{n,0},$$

azt biztosítván, hogy nagy valószínűséggel ne kapjunk túl nagy hibát eredményező lépésközt. Továbbá, biztosítsuk, hogy h ne változzon túl gyorsan, azaz szabjunk ezen tényezőnek egy alsó $\rho_{\min}$, és egy felső $\rho_{\max}$ korlátot (általában $1.5 \leq \rho_{\max} \leq 5$),

$$\rho_{n,2} = \max(\rho_{\min}, \rho_{n,1}), \quad \rho_{n,3} = \min(\rho_{\max}, \rho_{n,2}),$$

amivel elérkeztünk a heurisztikánk végéhez, azaz frissíthetjük a lépésközünket a

$$h_{n+1} = \rho_{n,3} \cdot h_n,$$

képlettel, ami kibontva a

$$h_{n+1} = \min\left(\rho_{\max}, \max\left(\rho_{\min}, m \cdot \left(\frac{1}{\|l_n\|}\right)^{\frac{1}{p+1}}\right)\right) \cdot h_n$$

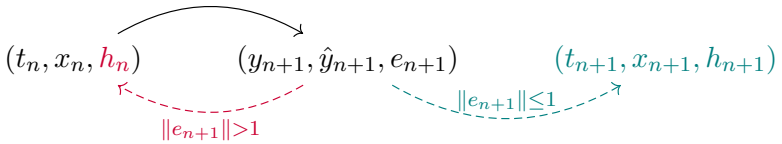
alakot ölti.

4.3. A klasszikus adaptív algoritmus

Az eddigiek alkalmazásaként kapjuk a következő módszert.

1. Vegyünk egy alkalmasan megválasztott h_0 kezdeti lépéshosszt.
2. Amikor a (t_n, x_n) állapotban vagyunk:
   (a) Egy hibabecslő módszerrel hozzuk létre az $x(t_n+h_n)$ két $y_{n+1}, \hat{y}_{n+1}$ közelítését, illetve az $\tilde{e}_{n+1} = y_{n+1} - \hat{y}_{n+1}$ hibavektort.
   (b) Amennyiben $\|e_{n+1}\| \leq 1$, akkor fogadjuk el az egyik közelítésünket, azaz legyenek
   $$x_{n+1} := y_{n+1}, \quad t_{n+1} := t_n + h_n, \quad h_{n+1} := h_n,$$
   és folytassuk a módszert a 2 ponttól.
   (c) Egyébként ismételjük meg az utóbbi három pontot h_n helyett a $\rho_n h_n$ lépésközzel.

Az eljárást a 4.1. ábra szemlélteti.



4.1. ábra. A klasszikus adaptív algoritmus

4.3.1. Numerikus kísérletek

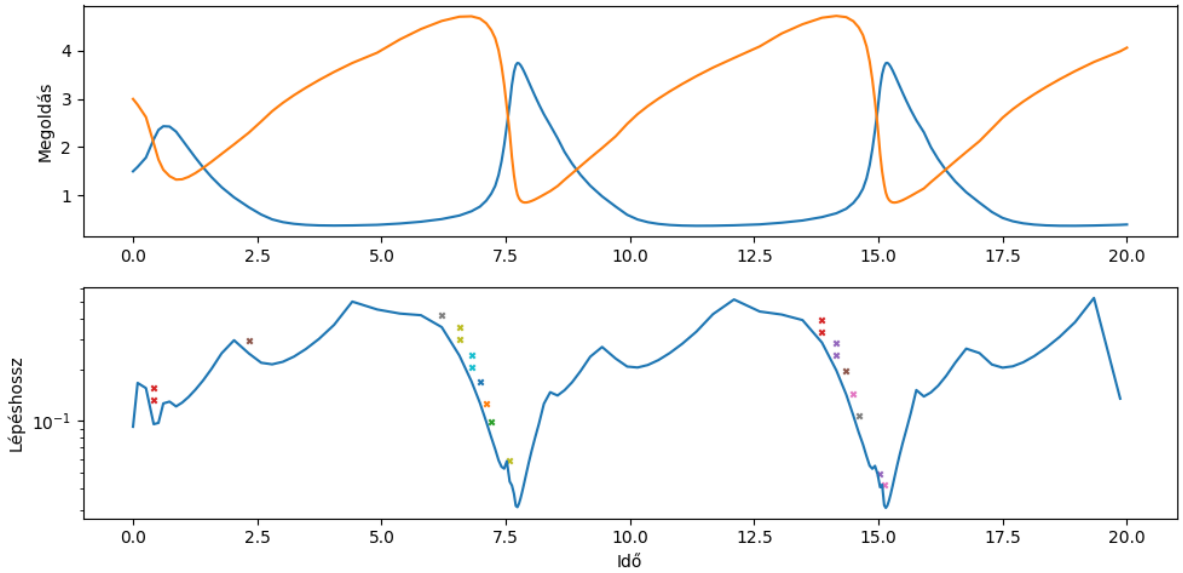
Az ismertetett algoritmust a Dormand–Prince módszerrel alkalmaztuk két problémára. A toleranciákat mindkét esetben 10^{-6}-nak választottuk.
Először a Brusselator problémán alapuló

$$\begin{aligned} \dot{x} &= 1 + x^2 y - 4x, & x(0) &= 1.5, \\ \dot{y} &= 3x - x^2 y, & y(0) &= 3 \end{aligned}$$

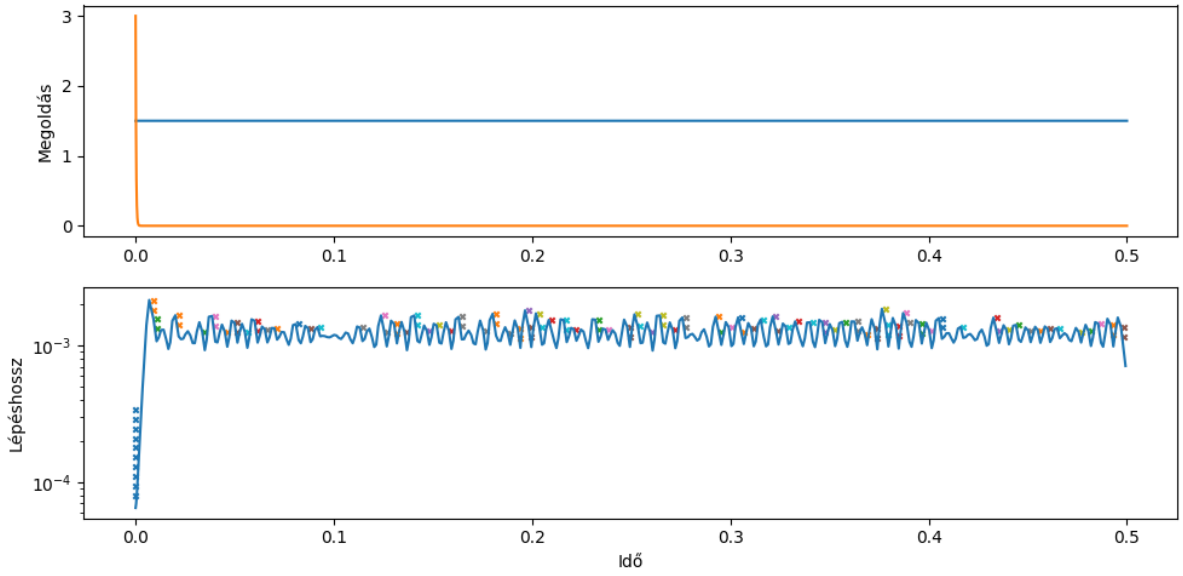
kezdetiérték-problémát oldottuk meg a $[0, 20]$ intervallumon. A futás során 111 elfogadott és 21 elutasított lépésköz lett kipróbálva. Az eredményeket a 4.2. ábra mutatja.
Másodszor pedig a Van der Pol problémán alapuló

$$\begin{aligned} \dot{x} &= y, & x(0) &= 1.5, \\ \dot{y} &= \mu(1 - x^2)y - x \quad , & y(0) &= 3 \end{aligned} \tag{4.4}$$

kezdetiérték-problémát $\mu = 3000$ paraméter választása mellett oldottuk meg a $[0, \frac{1}{2}]$ intervallumon. A futás során 428 elfogadott és 120 elutasított lépésköz lett kipróbálva. Az eredményeket a 4.3. ábra mutatja. Érdemes megjegyezni, hogy ezekkel a paraméterezésekkel élve ez a probléma egy lényegesen merevebb feladatnak felel meg. Az ábrán szembetűnő a lépéshosszok oszcillációja. Az utolsó fejezetben kiderül, hogy ez az oszcilláció megfelelő szabályozó választásával kiküszöbölhető, lásd a 6.5. ábrát.



4.2. ábra. A Brusselator probléma megoldása adaptív DORPRI módszerrel. A felső ábrán látható a megoldás két komponense az idő függvényében. Az alsó ábrán logaritmikus skálán az adott időpillanatban elfogadott lépésközök hossza látható, az értékek törtvonallal összekötve. Az adott időpillanatban visszautasított lépésközöket x-ek jelölik.



4.3. ábra. A Van der Pol probléma megoldása adaptív DORPRI módszerrel. A felső ábrán látható a megoldás két komponense az idő függvényében. Az alsó ábrán logaritmikus skálán az adott időpillanatban elfogadott lépésközök hossza látható, az értékek törtvonallal összekötve. Az adott időpillanatban visszautasított lépésközöket x-ek jelölik. Vesd össze a 6.5. ábrával.

5. fejezet

Az adaptív lépésközválasztás irányításelméleti eszközei

Ebben a fejezetben bemutatunk néhány olyan eszközt, melyek segítségével az adaptív lépésközválasztó módszerek matematikailag precízen vizsgálhatóak. Ezzel párhuzamosan a bemutatott eszközök segítségével megvizsgáljuk az előző fejezetben ismertetett klasszikus lépésközválasztó módszert.
A fejezet célja, hogy a [18] előadások nyomán a felhasznált elméleteket a releváns alkalmazások megértéséhez elegendő mélységig ismertesse. Az érdeklődő Olvasónak mélyebb iránytáselméleti bevezetőt például a [24] könyv szolgáltat.

5.1. Irányításelméleti bevezető

Az irányításelmélet központi problémája egy olyan irányító jel konstruálása, aminek segítségével egy adott dinamikai rendszer egy számunkra kívánatos állapotba hozható.
Szemléletes példa egy tenyéren álló pálca apró kézmozdulatokkal való egyensúlyban tartása. Ennek a feladatnak az absztrahált változata az invertált inga balanszírozásának problémája.
Az irányítani kívánt dinamikai rendszer lehet folytonos, vagy diszkrét idejű. A folytonos (analóg) rendszerekből jelfeldolgozási módszerekkel (például mintavételezéssel) készíthetünk diszkrét (digitális) rendszereket, amiket már számítógépek segítségével irányíthatunk. Alkalmazásokban ezért digitális rendszerekkel, és azok irányításával találkozhatunk gyakrabban, így ebben a dolgozatban is erről az esetről lesz szó. Megjegyezzük, hogy az analóg eset hasonló eszközökkel vizsgálható.
Amennyiben az irányító módszerünk jól működik, a fellépő perturbációk gyorsan korrigálásra kerülnek, így végig az egyensúlyi pont kicsi környezetében maradunk, ezért a legtöbb felmerülő feladat esetén nem vétünk nagy hibát azzal, ha az eredeti rendszer helyett annak linearizáltját vizsgáljuk. A továbbiakban tehát feltesszük, hogy az irányítandó rendszereink lineárisak. Végezetül megjegyezzük, hogy mivel céljainkhoz elegendő a skalár értékű rendszerek irányítása, ezért ebben a dolgozatban csak ilyenekről lesz szó.

5.1.1. Egy motiváló példa

Az irányításelmélet alapfogalmainak bevezetéséhez tekintsünk egy egyszerű instabil diszkrét dinamikai rendszert:

$$x_{n+1} = 2x_n.$$

A rendszer megoldásai $x_0 \cdot 2^n$ alakúak, és világos, hogy a 0 instabil egyensúlyi pontja. Felmerül a kérdés, hogy feltéve, hogy az n-edik időpillanatig ismerjük a rendszer x **állapotát** egy y

kimeneti jel formájában, tudunk-e úgy választani egy u **bemeneti jel**et, hogy azzal a rendszert megperturbálva az adódó

$$x_{n+1} = 2x_n + u_n$$

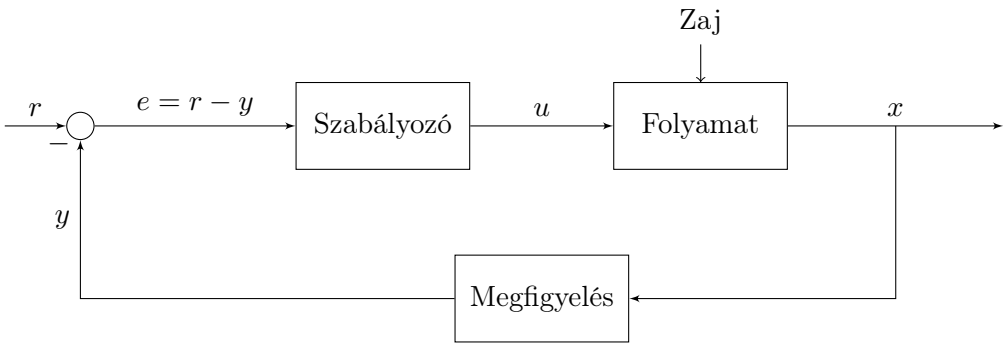
egyenlet már egy stabil rendszert írjon le, azaz 0 közeléből indulva a rendszer állapota ne távolodjon el túl messzire.
Az y kimeneti jelet figyelve célunk tehát az, hogy olyan u jelet konstruáljunk, amivel a rendszer kimenete egy adott r **referenciaérték** körül lesz, azaz minimalizáljuk az

$$e = r - y$$

hibát.
Irányításelméleti értelemben adott tehát egy **folyamat**, ami az u bemeneti jelből gyárt egy y kimeneti jelet -azaz egy $u \mapsto y$ leképezés, nekünk pedig készítenünk kell egy **szabályozót**, ami a tapasztalt hibából elkészíti az alkalmas perturbációt - azaz egy $e \mapsto u$ leképezést.
Az egyszerűség kedvéért tegyük fel most, hogy $y = x$, azaz a rendszer állapotát pontosan megfigyelhetjük. Továbbá, hogy a szabályozó is az

$$u = K \cdot e$$

alakot ölti, ahol K egy konstans. Mivel lineáris rendszerekkel szeretnénk dolgozni, ezért a szabályozót is természetes lineárisnak választani, ezek között pedig a legegyszerűbb típus a fenti.



5.1. ábra. Az irányított rendszer

Kapjuk tehát az 5.1. ábrán látható,

$$\begin{aligned} x_{n+1} &= 2x_n + u_n \\ y &= x \\ e &= r - y \\ u &= Ke \end{aligned}$$

egyenletekkel leírható rendszert. A változók eliminálásával innen adódik a

$$x_{n+1} = 2x_n + K(r_n - x_n) = (2 - K)x_n + Kr_n = (2 - K)x_n$$

zártkörű rendszer, ahol a harmadik egyenlőség esetünkben az $r = 0$ választás miatt teljesül. Láthatjuk, hogy az új rendszer pontosan akkor lesz aszimptotikusan stabil, ha $1 < K < 3$. Megemlítjük, hogy a szabályozó ilyen módú megválasztása a **negatív visszacsatolás** egy példája volt, mivel az irányító jelben a bemeneti jel ellentétes előjellel szerepel.

5.1.2. Vizsgálat z-transzformációval

Az irányításelmélet egyik kulcsfontosságú eszköze a z-transzformáció. Erejét szemléltetendő, segítségével vizsgáljuk meg újra a korábbi,

$$\begin{aligned} x_{n+1} &= 2x_n + u_n \\ y &= x \end{aligned}$$

példánkat. Alkalmazzunk z-transzformációt az egyenletek mindkét oldalára, így adódik a

$$\begin{aligned} (z-2)\hat{x} &= \hat{u} \\ \hat{y} &= \hat{x} \end{aligned}$$

egyenletrendszer. Az $\hat{y}$ változóra rendezve kapjuk, hogy

$$\hat{y} = \frac{1}{z-2} \cdot \hat{u} = G(z) \cdot \hat{u}.$$

Ezzel megkaptuk a folyamat G **átviteli függvény**ét.
Hasonlóképpen elkészíthető a többi fogalom transzformált megfelelője is, például a szabályozó működése az

$$\hat{u} = C(z) \cdot (\hat{r} - \hat{y})$$

egyenlettel írható le, ahol C a szabályozó átviteli függvénye. Ez egy racionális törtfüggvény, példánkban $C(z) = K$.
Összegezve tehát azt kaptuk, hogy

$$\begin{aligned} \hat{y} &= G(z)\hat{u} \\ \hat{u} &= C(z)(\hat{r} - \hat{y}). \end{aligned}$$

Innen az ismeretlenek baloldalra rendezésével, majd a felmerülő

$$\begin{aligned} G(z)\hat{u} - \hat{y} &= 0 \\ \hat{u} + C(z)\hat{y} &= C(z)\hat{r} \end{aligned}$$

lineáris rendszer megoldásával kapjuk a **zárt rendszer** két átviteli függvényét

$$\begin{aligned} \hat{r} \mapsto \hat{u} &= \frac{C(z)}{1 + C(z)G(z)}\hat{r}, \\ \hat{r} \mapsto \hat{y} &= \frac{C(z)G(z)}{1 + C(z)G(z)}\hat{r}. \end{aligned}$$

Világos, hogy mindkét leképzés racionális törtfüggvény, illetve az is, hogy ezen függvények pólusai fogják meghatározni a zárt rendszer stabilitását.
Példánkhoz visszatérve az átviteli függvény

$$\frac{C(z)G(z)}{1 + C(z)G(z)} = \frac{K\frac{1}{z-2}}{1 + K\frac{1}{z-2}} = \frac{K}{z - (2-K)},$$

és a rendszer stabilitásához szükséges, hogy a pólusa, $(2-K)$ az egységkörön belül legyen, azaz újfent megkaptuk, hogy a stabilitáshoz $1 \leq K \leq 3$ szükséges.

5.2. Digitális szűrők

A lineáris differenciaegyenletek, és a lineáris irányításelmélet rokonterülete a lineáris **digitális szűrő**k világa. Ezen elmélet keretei között is

$$Q(E)x = P(E)f$$

alakú egyenleteket vizsgálnak, ahol f a bemeneti jel, x pedig a kimeneti jel szerepét játssza, P és Q polinomok, E pedig az előretolás operátor.
A szűrő viselkedését a megfelelő átviteli függvényen keresztül szokás vizsgálni. Ezt szokás szerint z-transzformációval kaphatjuk az eredeti egyenletünkből, ebben az esetben az

$$\hat{x} = \frac{P(z)}{Q(z)}\hat{f} = F(z)\hat{f}$$

alakú egyenletről van szó. Kiemelt jelentősége van a **frekvencia-válasz** nevű

$$\omega \mapsto |F(e^{i\omega})|$$

leképezésnek, melynek vizsgálatáról később részletesebben is szó lesz.
Világos, hogy Q ebből a nézőpontból is a stabilitásért felel, azaz amennyiben teljesül rá a gyök-kritérium, akkor a megoldás homogén egyenlethez tartozó komponensei gyorsan jelentéktelenné válnak, és x-ben az f által meghatározott komponensek fognak dominálni.

HF, LP szűrők

A mi alkalmazásunkban a bemeneti jel alacsony frekvenciás komponenseit tekintjük lényegesnek, a magas frekvenciás komponenseket pedig alapvetően zajként fogjuk fel, igyekszünk azoktól megszabadulni. Az ilyen szűrőket hívják az angol high-frequency és low-pass kifejezések után **HF**, vagy **LP** szűrőknek.
Míg a stabilitásért a Q felelős, a szűrést a P polinom megválasztásával végezhetjük el, mint ahogy azt a következő példa is szemlélteti.
Tegyük fel, hogy f tartalmaz egy $(-1)^n$ jellegű oszcillációt, az egyszerűség kedvéért legyen

$$x = P(E)f, \quad \text{ahol} \quad f_n = 100 + (-1)^n.$$

Amennyiben a jel alacsony frekvenciás részeit tekintjük lényegesnek, akkor a

$$P(z) = z + 1$$

választással élve a zajt a szűrő sikeresen eltávolítja, és kapjuk a

$$x = 200$$

konstans kimeneti jelet. Ha célunk az, hogy az alacsony frekvenciás jeleket (mint például a konstans jel) a szűrő engedje át lényegében érintetlenül, akkor pedig a

$$P(z) = \frac{z+1}{2}$$

átlagoló operátor választással jó úton haladunk, hiszen ezzel a kimeneti jel $x = 100$ lesz. A szűrő tehát a zajt teljes mértékben eltávolította és a jel lényegi részét is meghagyta.
Észrevehetjük, hogy ennek a két feltételnek, miszerint a szűrő engedje át a konstans jelet és teljes mértékben blokkolja a π frekvenciájú oszcillációt, minden olyan P polinom megfelelő lenne, amire

$$P(-1) = 0 \quad \text{és} \quad P(1) = 1$$

teljesül. A legegyszerűbb struktúrájú, ezeket a kritériumokat kielégítő szűrők példája a k-szorosan átlagoló operátorok családja, amiknek a

$$P_k(z) = \left(\frac{z+1}{2}\right)^n$$

polinomok felelnek meg. A hatványkitevő növelésével elérhetjük, hogy a π körüli frekvenciájú jelek szintén lényegében teljesen blokkolódjanak, azonban ennek ára van, hiszen k növelésével a 0 körüli frekvenciájú jelek is egyre inkább tompulnak.

5.3. A lépésközválasztó rendszer

A fenti eszközökkel felvértezve a lépésközválasztás problematikája új megvilágításba kerül. Ezt a korábbi fejezetben ismertetett klasszikus lépésközválasztási módszer vizsgálatán keresztül mutatjuk be.
Emlékeztetőül, az alapgondolat ott az volt, hogy a lokális hibát (annak alkalmas normáját) szeretnénk egy tolerancia értékhez közel tartani.
A célból, hogy a lépésközválasztó folyamatot könnyen modellezhessük lineáris differenciaegyenletek segítségével, a felmerülő mennyiségek helyett azoknak logaritmusával fogunk dolgozni.
Irányításelméleti nyelvezettel élve tehát a toleranciánkból referenciaérték, a lokális hibabecslésből kimeneti jel lesz.

A folyamat

A lokális hibabecslést előállító folyamat lesz a folyamatunk, ami most lépéshosszból lokális hibabecslést készít. A fejezetben csak az aszimptotikus hibamodellel foglalkozunk, azaz feltesszük, hogy a lokális hibabecslés alakja

$$r_n = \varphi_n h_n^k.$$

Logaritmust véve adódik a

$$\log r_n = k \cdot \log h_n + \log \varphi_n \tag{5.1}$$

alak, ahol φ_n felfogható úgy is, mint egy külső zaj.

A szabályozó

A kör a szabályozó megválasztásával zárul, ami most a lokális hibabecslés és a tolerancia érték közti különbségből legyártja a megfelelő lépésközt. A klasszikus lépésközválasztó módszer, azaz a

$$h_{n+1} = \left(\frac{\mathrm{TOL}}{r_n}\right)^{1/k} h_n$$

formula esetén logaritmust véve adódik a megfelelő

$$\log h_{n+1} = \log h_n + \frac{1}{k}\left(\log \mathrm{TOL} - \log r_n\right) \tag{5.2}$$

lineáris differenciaegyenlet.

5.3.1. Vizsgálat z-transzformációval

A rendszert vizsgálhatjuk a z-transzformáció segítségével is. Az egyszerűség kedvéért a $\{\log r_n\}_{n=0}^{\infty}$ sorozat transzformáltját $\log \hat{r}$ módon fogjuk jelölni.

A folyamat

Könnyen meggondolható, hogy az (5.1) folyamat transzformáltja

$$\log \hat{r} = k \log \hat{h} + \log \hat{\varphi} = G(z) \log \hat{h} + \log \hat{\varphi}.$$

Megjegyezzük, hogy más hibamodellek esetén G alakja bonyolultabb lehet.

A szabályozó

Hasonlóan, az (5.2) egyenletet mindkét oldalát transzformálva kapjuk a

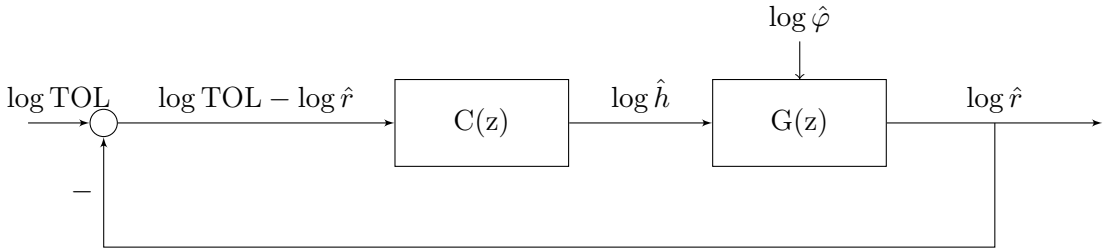
$$(z-1) \log \hat{h} = \frac{1}{k} (\log \mathrm{TOL} - \log \hat{r}) \tag{5.3}$$

egyenletet, amiből $\log \hat{h}$ kifejezésével láthatóvá válik a szabályozó transzformáltja:

$$\log \hat{h} = \frac{1}{k} \frac{1}{z-1} \cdot (\log \mathrm{TOL} - \log \hat{r}) = C(z) \cdot (\log \mathrm{TOL} - \log \hat{r}) \,. \tag{5.4}$$

A zárt rendszer

A zárt rendszert az 5.2. ábra szemlélteti.



5.2. ábra. A lépéshosszválasztó rendszer

A z-transzformáció ereje abban mutatkozik meg, hogy a rendszer viselkedése a kapott átviteli függvényeken keresztül egyszerűbben vizsgálható. Ez a könnyebbség a zárt rendszer átviteli függvényeinek esetében a legszembetűnőbb.
Ugyanis a z-transzformáció lehetővé teszi, hogy a (5.3), (5.4) egyenleteket együttesen kezeljük az alábbi,

$$\begin{aligned} \log \hat{h} + C(z) \log \hat{r} &= C(z) \log \mathrm{TOL} \\ -G(z) \log \hat{h} + \log \hat{r} &= \log \hat{\varphi} \end{aligned}$$

lineáris egyenletrendszer formájában.
Ennek előnye, hogy a számunkra érdekes mennyiségeket ebből könnyedén kifejezhetjük a külső hatások függvényként:

$$\begin{aligned} \log \hat{h} &= -\frac{C(z)}{1 + C(z)G(z)} \log \hat{\varphi} + \frac{C(z)}{1 + C(z)G(z)} \log \mathrm{TOL} \\ &= H_{\varphi}(z) \log \hat{\varphi} + H_{\mathrm{TOL}}(z) \log \mathrm{TOL}, \\ \log \hat{r} &= \frac{1}{1 + C(z)G(z)} \log \hat{\varphi} + \frac{C(z)G(z)}{1 + C(z)G(z)} \log \mathrm{TOL} \\ &= R_{\varphi}(z) \log \hat{\varphi} + R_{\mathrm{TOL}}(z) \log \mathrm{TOL}. \end{aligned}$$

Az adódó leképezések tovább egyszerűsödnek amennyiben a TOL toleranciát egynek választjuk. Szerencsénkre ezzel az általánosságot nem veszítjük el, ugyanis tegyük fel, hogy egy adott kezdetiérték-problémát oldunk meg egy adott adott numerikus integráló módszerrel. Továbbá tegyük fel, hogy a rendszer éppen a (t, x) állapotban van. Amennyiben rendelkezésünkre áll egy lépésközválasztó módszer, aminek segítségével létrehozható egy olyan $h(t)$ lépésköz, amivel a lokális hiba körülbelül 1, akkor az aszimptotikus hibastruktúrából adódóan

$$1 \approx \varphi h_1^k$$

teljesül. Ebből a két oldal TOL konstanssal való szorzásával kapjuk, hogy

$$\mathrm{TOL} \approx \varphi \cdot \left(h_1 \mathrm{TOL}^{1/k}\right)^k,$$

azaz a (t, x) állapotban $h_{\mathrm{TOL}} = \mathrm{TOL}^{1/k} h_1$ lépésközzel lépve a lokális hiba a kívánt tolerancia körül lesz. Tehát a tolerancia megváltoztatása a lépésközök átskálázódását eredményezi, így a folyamat megértéséhez elég a $\mathrm{TOL} = 1$ esetet vizsgálni.

5.4. A rendszer frekvencia-válasza

A modellünkben φ reprezentálja a külső (differenciálegyenlet numerikus megoldási folyamatából) származó jelet. Amennyiben a zárt rendszerünk stabil, a differenciaegyenlet homogén megoldásai gyorsan lecsengenek, és így a φ-hez tartozó partikuláris megoldások fognak dominálni.
Ebben a részben tehát a H_φ, R_φ leképezéseket fogjuk alaposabban megvizsgálni, azaz arról lesz szó, hogy miként modellezhető a megoldás során fellépő hiba hatása a választott lépésközre, illetve a lokális hibabecslésre.
Az aszimptotikus hibamodellt feltéve H_φ helyett szokás a kH_φ skálázott átviteli függvényt vizsgálni, ezért mi is így fogunk tenni. Ebben az esetben az átviteli függvényeink a következő alakot öltik:

$$kH_\varphi(z) = -\frac{kC(z)}{1 + kC(z)},$$
$$R_\varphi(z) = \frac{1}{1 + kC(z)}.$$

A további vizsgálódáshoz feltesszük, hogy φ korlátos és különböző frekvenciák lineáris kombinációjaként áll elő. Mivel a rendszerünk lineáris, ezért viselkedését frekvenciánként elemezhetjük, hiszen a bemeneti jel komponenseinek frekvenciáit a rendszer nem tudja összekeverni, csupán a komponensek amplitúdóján és fázisán tud változtatni.
Precízebben, legyen $\omega \in [0, \pi]$ egy adott frekvencia. Ekkor az ω frekvenciájú jelnek az

$$\{e^{i\omega n}\}_{n=0}^{\infty}$$

sorozat felel meg. Mivel az átviteli függvényeink racionális törtfüggvények, ezért előállnak

$$H_\varphi = \frac{P}{Q}$$

alakban, ahol P és Q polinomok és $\deg P < \deg Q$. Így adódik a következő z-transzformált egyenlet, illetve az annak megfelelő lineáris differenciaegyenlet:

$$Q(z) \log \hat{h} = P(z) \log \hat{\varphi},$$
$$Q(E) \log h_n = P(E) \log \varphi_n,$$

ahol E az előretolás operátor. Behelyettesítve a $\log \varphi_n = e^{i\omega n}$ jelet némi számolással adódik, hogy

$$\log h_n = \frac{P(e^{i\omega})}{Q(e^{i\omega})} e^{i\omega n} = H_\varphi(e^{i\omega}) e^{i\omega n}.$$

A keletkező $\log h$ jel tehát csak fázisban és amplitúdóban tér el a bemeneti jeltől. A számunkra lényeges ezek közül az adott frekvenciájú jel amplitúdójának megváltozása lesz, amit az úgynevezett **attenuáció** formájában szokás számszerűsíteni:

$$A(\omega) = |H_\varphi(e^{i\omega})|.$$

A frekvenciához attenuációt rendelő $\omega \mapsto A(\omega)$ leképezés grafikonját **frekvencia-válasznak** nevezik.
Vezessük be továbbá a zárt rendszer két legfontosabb átviteli függvényének frekvencia-válaszát, azaz a skálázott lépéshossz frekvencia-válaszát

$$20 \cdot \log_{10} |kH_\varphi(e^{i\omega})|,$$

és a hiba frekvencia-válaszát

$$20 \cdot \log_{10} |R_\varphi(e^{i\omega})|,$$

ahol a skálázást azért végeztük el, hogy az attenuációkat a szokásos mértékegységükben, decibelben adhassuk meg.

A klasszikus lépésközválasztó szabályozó példája

Vizsgáljuk meg, hogy mit árulnak el ezek a leképezések a klasszikus lépésközválasztó szabályozóról. A skálázott lépéshossz frekvencia-válasza az

$$\omega \mapsto 20 \log_{10} \left|kH_\varphi(e^{i\omega})\right| = 20 \log_{10} \left|-\frac{1}{e^{i\omega}}\right| = 0$$

decibel értéket adja, tehát ez a szabályozó semmilyen frekvenciájú bemeneti jel amplitúdóját sem változtatja meg, például a magas frekvenciájú zajt sem tompítja.
A hiba frekvencia-válasza ezzel szemben a következőképpen alakul:

$$\omega \mapsto 20 \log_{10} \left|R_\varphi(e^{i\omega})\right| = 20 \log_{10} \left|\frac{e^{i\omega} - 1}{e^{i\omega}}\right| \approx 20 \log_{10} |\omega|,$$

ahol az utolsó közelítés alkalmasan kicsi ω értékekre ésszerű.
Észrevehetjük, hogy a φ-ből származó konstans komponenst, azaz az $\omega = 0$ frekvenciájút a szabályozó teljes mértékben eltünteti. Így az integrációs lépések során a lokális hibabecslésekben nem lesz egy közös konstans komponens, amiből azt láthatjuk, hogy a szabályozó a lépéshosszok nagyságrendjét megfelelően beállítja. Továbbá elmondható, hogy a 0 körüli viselkedésből egyfajta rend olvasható le. A rendekről bővebben a következő fejezetben lesz szó.

6. fejezet

Szabályozók adaptív lépésközválasztáshoz

Ebben a fejezetben a [17],[19], [21] cikkek és a [18] előadások nyomán a lépésközválasztás irányításelméleti megközelítésében használt szabályozókról lesz szó. Bemutatjuk az ilyen szabályozók általános struktúráját, majd röviden ismertetjük a szabályozók és az irányított rendszerek osztályozására alkalmas rendeket. Ezt követően szó lesz az aszimptotikus és a dinamikus hibamodell esetéről is, és ezzel párhuzamosan példákat mutatunk arról, hogy a különböző hibamodelleket feltételezve milyen szabályozók készíthetők.

6.1. Az általános szabályozó struktúra

A fejezet kiindulópontja a klasszikus adaptív megközelítés alapgondolata, a multiplikatív lépésközválasztás. Ez az elv azt mondja ki, hogy az új, $(n+1)$-edik lépéshosszt az alábbi,

$$h_{n+1} = \rho_n h_n$$

alakban keressük. Ez az alak a lépésközválasztó módszer általánosságát lényegében nem befolyásolja, azonban magában hordozza azt az ötletet, miszerint az új lépésköz meghatározásánál a legutóbbi lépésköznek jelentős szerepe van.
Logaritmust véve, majd z-transzformálva kapjuk a

$$(z-1)\log\hat{h} = \log\hat{\rho}$$

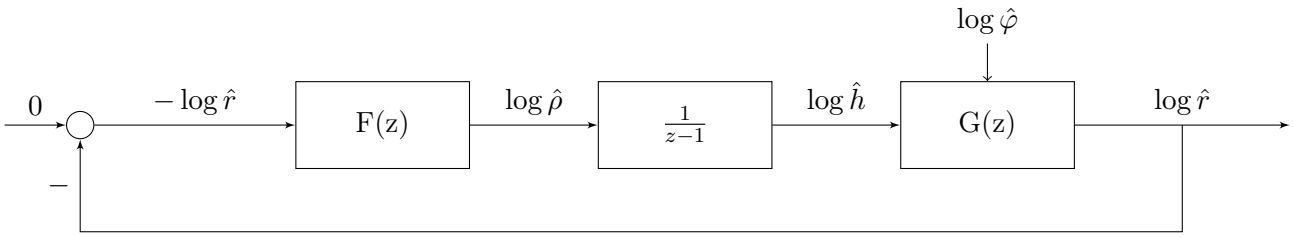
egyenletet, azaz irányításelméleti megközelítésünkben ez a feltevés nem jelent mást, mint azt, hogy a szabályozó átviteli függvénye a

$$\log\hat{h} = -C(z)\cdot\log\hat{r} = -\frac{1}{z-1}F(z)\cdot\log\hat{r} = \frac{1}{z-1}\cdot\log\hat{\rho}$$

alakban faktorizálható, ahol

$$F(z) = \frac{P(z)}{Q(z)}$$

egy digitális szűrő. A szabályozó faktorizációja után a lépésközválasztó rendszer a 6.1. ábrán látható alakot ölti.



6.1. ábra. **A lépéshosszválasztó rendszer digitális szűrővel**

A megközelítés eredménye, hogy az alkalmas szabályozó megtalálásának feladatát visszavezeti egy megfelelő digitális szűrő megtalálására.
Ezekkel a feltevésekkel, amennyiben a polinomok

$$P(z) = \sum_{j=0}^{p} \beta_{p-j} z^j, \qquad\qquad Q(z) = \sum_{j=0}^{p} \alpha_{p-j} z^j$$

alakúak (ahol feltesszük, hogy $\alpha_0 = 1$), akkor az általunk vizsgált legáltalánosabb lépéshossz frissítő képlet a következő alakot ölti:

$$\left(\frac{h_n}{h_{n-1}}\right)^{\alpha_1} \left(\frac{h_{n-1}}{h_{n-2}}\right)^{\alpha_2} \cdots \left(\frac{h_{n-(p-1)}}{h_{n-p}}\right)^{\alpha_p} h_{n+1} = \left(\frac{\mathrm{TOL}}{r_n}\right)^{\frac{\beta_0}{k}} \left(\frac{\mathrm{TOL}}{r_{n-1}}\right)^{\frac{\beta_1}{k}} \cdots \left(\frac{\mathrm{TOL}}{r_{n-p}}\right)^{\frac{\beta_p}{k}} \cdot h_n. \tag{6.1}$$

Ez az alak paraméterezéstől függően magában foglalja többek között az autoregresszív (AR), mozgóátlag (MA), ARMA, véges válaszidejű (FIR), I, PI, PID, ... struktúrájú folyamatokat. A megfelelő paraméterezés megtalálásának egy módja tehát ezen struktúrák egyikének kiválasztása, majd a paraméterek további tuningolása a célból, hogy az eredmény egy stabil, és jó frekvenciaválaszú szabályozó legyen.

6.1.1. Implementációs megjegyzések

Skálázott irányítási hiba

A skálázott irányítási hibák

$$c_n = \left(\frac{\mathrm{TOL}}{r_n}\right)^{\frac{1}{k}}$$

bevezetésével a (6.1) formula az egyszerűbb,

$$\begin{aligned} \rho_n &= c_n^{\beta_0} c_{n-1}^{\beta_1} \cdots c_{n-p}^{\beta_p} \cdot \rho_{n-1}^{-\alpha_1} \rho_{n-2}^{-\alpha_2} \cdots \rho_{n-p}^{-\alpha_p} \\ h_{n+1} &= \rho_n h_n \end{aligned}$$

alakot ölti. Ennek előnye, hogy implementációkban is könnyen használható formája van, hiszen alkalmazásához az irányítatlan iterációhoz képest elég számontartani a ρ_n lépéshossz hányadosokat és a c_n skálázott irányítási hibákat.

6.1.2. Néhány nevezetes speciális eset

Most következzen a (6.1) formulájú általános struktúra szűrésért felelős jobboldalának néhány nevezetes speciális esete.

I-szabályozás

Az I-szabályozás struktúrája a β_0 paraméter bevezetésével a klasszikus megközelítés apró módosításával adódik:

$$h_{n+1} = \left(\frac{\mathrm{TOL}}{r_n}\right)^{\frac{\beta_0}{k}} h_n.$$

Az I az integráció szóra utal, ami alatt most a diszkrét mérték szerinti integrálást kell érteni, ugyanis logaritmust véve az $(n+1)$-edik lépéshosszra a

$$\log h_{n+1} = \log h_0 + \sum_{j=0}^{n} \frac{\beta_0}{k} \left(\log \mathrm{TOL} - \log r_j\right)$$

képlet adódik. A $\frac{\beta_0}{k}$ érték az integrációs erősítési tényező (integral gain).

PI-szabályozás

A PI-szabályozást az I-szabályozásból úgy kapjuk, hogy külön figyelmet fordítunk a legutóbbi irányítási hibára. Struktúráját a következő formula írja le:

$$h_{n+1} = \left(\frac{\mathrm{TOL}}{r_n}\right)^{\frac{\beta_0}{k}} \left(\frac{\mathrm{TOL}}{r_{n-1}}\right)^{\frac{\beta_1}{k}} h_n.$$

A P az angol proportionality szóra utal, mely magyarul arányosságot jelent. Az elnevezést az motiválja, hogy az I-szabályozást leíró formulához hozzáveszünk egy, a legutóbbi hibával arányos tagot. Ha ez az arányossági tényező γ, akkor a megfelelő formula a

$$\log h_{n+1} = \log h_0 + \sum_{j=0}^{n} \frac{\beta}{k} \left(\log \mathrm{TOL} - \log r_j\right) + \frac{\gamma}{k} \left(\log \mathrm{TOL} - \log r_n\right),$$

ahol β az I-szabályozásnál látott integrációs arányossági tényező. Ezen paraméterezés mellett gyors számolással adódik a fent ígért struktúra. Valóban,

$$\begin{aligned}\log h_{n+1} - \log h_n &= \frac{\beta}{k} \left(\log \mathrm{TOL} - \log r_n\right) + \frac{\gamma}{k} \left(\log \mathrm{TOL} - \log r_n\right) - \frac{\gamma}{k} \left(\log \mathrm{TOL} - \log r_{n-1}\right) \\ &= \frac{\beta+\gamma}{k} \left(\log \mathrm{TOL} - \log r_n\right) - \frac{\gamma}{k} \left(\log \mathrm{TOL} - \log r_{n-1}\right),\end{aligned}$$

így tehát megkaptuk a

$$h_{n+1} = \left(\frac{\mathrm{TOL}}{r_n}\right)^{\frac{\beta+\gamma}{k}} \left(\frac{\mathrm{TOL}}{r_{n-1}}\right)^{-\frac{\gamma}{k}} h_n$$

lépésközfrissítő képletet.

PID-szabályozás

A PID-szabályozást a PI-szabályozásból úgy kapjuk, hogy külön figyelmet fordítunk a hibák trendjének alakulására. Struktúráját a következő formula írja le:

$$h_{n+1} = \left(\frac{\mathrm{TOL}}{r_n}\right)^{\frac{\beta_0}{k}} \left(\frac{\mathrm{TOL}}{r_{n-1}}\right)^{\frac{\beta_1}{k}} \left(\frac{\mathrm{TOL}}{r_{n-2}}\right)^{\frac{\beta_2}{k}} h_n.$$

A D az angol derivative szóra utal. Az elnevezést az motiválja, hogy a PI-szabályozást leíró formulához hozzáveszünk egy, a hiba trendjének legutóbbi értékével arányos tagot. Ha ez az arányossági tényező δ, akkor a megfelelő formula a

$$\log h_{n+1} = \log h_0 + \sum_{j=0}^{n} \frac{\beta}{k} (\log \mathrm{TOL} - \log r_j) + \frac{\gamma}{k} (\log \mathrm{TOL} - \log r_n) + \frac{\delta}{k} (\log r_{n-1} - \log r_n),$$

ahol β, γ a PI-szabályozás esetén látott konstansok. Ezen paraméterezés mellett a korábbihoz teljesen hasonló számolással a lépésköz frissítésére a következő formula adódik:

$$h_{n+1} = \left(\frac{\mathrm{TOL}}{r_n}\right)^{\frac{\beta+\gamma+\delta}{k}} \left(\frac{\mathrm{TOL}}{r_{n-1}}\right)^{-\frac{\gamma+2\delta}{k}} \left(\frac{\mathrm{TOL}}{r_{n-2}}\right)^{\frac{\delta}{k}} h_n.$$

6.2. Az irányított rendszer rendjei

Dinamikai rend

A dinamikai rend nem más, mint a szabályozónak megfelelő differenciaegyenlet rendje, azaz az áviteli függvényekkel kifejezve,

$$p_D := \deg((z-1)Q(z)).$$

Mivel p_D növelésével a szabályozó a múlt egyre hosszabb szeletét használja fel az új lépéshossz meghatározására, ezért a gyakorlatban $p_D \leq 3$ rendű szabályozókat szokás alkalmazni.

Szűrési rend

A szűrési rend alatt azt értjük, hogy a (-1) hányszoros gyöke a P polinomnak, ezért

$$p_F := \inf_{n\in\mathbb{N}} \left\{ \lim_{z\to -1} \frac{P(z)}{(z+1)^n} \neq 0 \right\}.$$

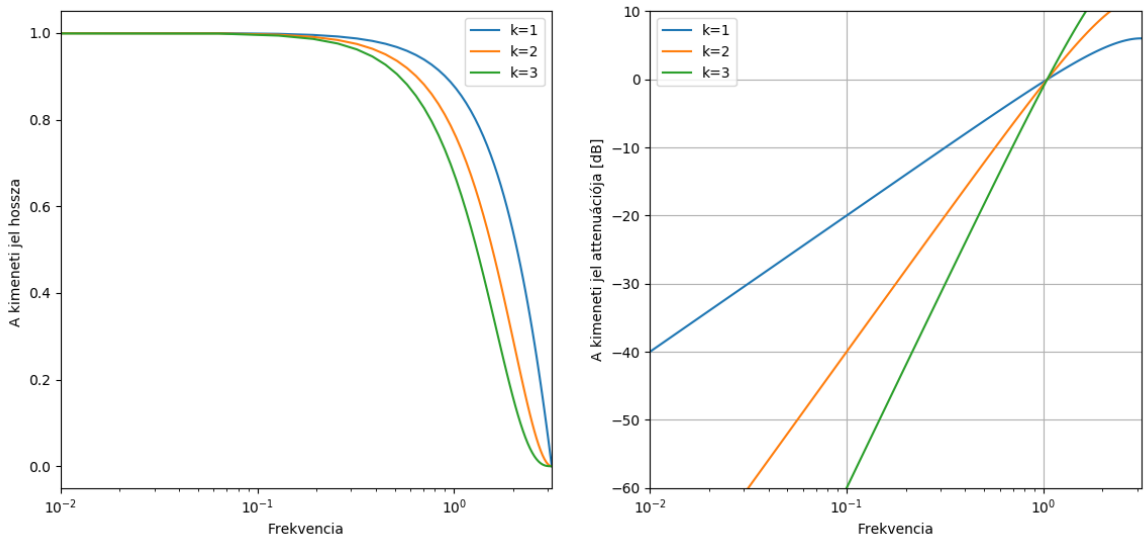
A szűrő feladata, hogy a magas frekvenciájú zaj elnyomása közben az alacsony frekvenciás jelet átengedje. Amennyiben $1 \leq p_F$, akkor a szűrő a legmagasabb, azaz a π frekvenciájú komponenst teljesen eltávolítja, továbbá a π-hez közeli frekvenciákat jelentősen elnyomja. Elmondható, hogy p_F növelésével a szűrő a magas frekvenciákat jobban kiszűri, cserébe viszont az alacsony frekvenciákat is egyre inkább elnyomja. A jelenséget a $2^{-k}(z+1)^k$ átviteli függvényű átlagoló operátor frekvencia-válaszán keresztül az 6.2. ábra szemlélteti.

Adaptivitási rend

Az adaptivitási rend alatt azt értjük, hogy az 1 hányszoros gyöke az R_φ átviteli függvénynek, azaz

$$p_A := \inf_{n\in\mathbb{N}} \left\{ \lim_{z\to 1} \frac{R_\varphi(z)}{(z-1)^n} \neq 0 \right\}.$$

Az adaptivitási rend azt írja le, hogy a rendszer milyen gyorsan képes a hibajel lényegi, alacsony frekvenciás részét eliminálni, azaz, hogy a rendszer milyen sebességgel képes adaptálni a lépéshosszokat a feladat numerikus megoldásából származó hibajel lényegi, alacsony frekvenciájú részének struktúrájához. Ennek eléréséhez azt követeljük meg, hogy a rendszer a 0 frekvenciájú (konstans) jelet teljesen eliminálja, az alacsony frekvenciájú jeleket pedig jelentősen elnyomja.
Érdemes megjegyezni, hogy $p_A = 1, 2$ és 3 esetekben a rendszer rendre a $\log r$ jel konstans, lineáris és kvadratikus komponenseire képes adaptálódni. Ebből következik, hogy amennyiben olyan rendszert szeretnénk tervezni, ami képes a megfelelő nagyságrendű lépéshossz megtalálására, akkor $1 \leq p_A$ konzisztencia jellegű feltételt biztosítanunk kell.



6.2. ábra. A baloldalon a $2^{-k}(z+1)^k$ átviteli függvényű átlagoló operátor, a jobboldalon pedig a $(z-1)^k$ átviteli függvényű differencia operátor frekvencia-válasza látható. A jobboldali ábráról leolvasható a $k = 1, 2, 3$ adaptivitási rendre jellegzetes $k \cdot 20\frac{\text{dB}}{\text{dec}}$ meredekség, azaz a nagyságrendenkénti $k \cdot 20\text{dB}$ változás.

Integrációs rend

Az integrációs rend alatt azt értjük, hogy az 1 hányadrendű pólusa a szabályozó átviteli függvényének:

$$p_C := \inf_{n\in\mathbb{N}} \left\{ \lim_{z\to 1} |C(z)(z-1)^n| \neq \infty \right\}.$$

Ez a gyakorlatban egyenlő az adaptivitási renddel, hiszen az átviteli függvényekre teljesül a

$$C(z) = -\frac{H_\varphi(z)}{R_\varphi(z)}$$

összefüggés.

Hiba szűrési rend

Amennyiben szeretnénk, hogy a hibajelből a magas frekvenciájú zaj is eltávolításra kerüljön, akkor erre a célra is használhatunk LP szűrőket. A p_F szűrési rendhez analóg módon definiálhatjuk a hiba szűrési rendet is:

$$p_R := \inf_{n\in\mathbb{N}} \left\{ \lim_{z\to -1} \frac{R_\varphi(z)}{(z+1)^n} \neq 0 \right\}.$$

A rendek és a frekvencia-válasz kapcsolata

Érdemes megemlíteni, hogy a különböző rendek sok esetben az adott leképezés frekvencia-válaszának diagramjáról leolvashatók. Például a hiba átviteli függvényét az $R_\varphi(z) = (z-1)^k$ alakú differencia operátorral modellezve a 6.2. ábrát kapjuk. Az ábráról az adaptivitási rend az $\omega \ll 3$ értékekhez tartozó attenuációk alkotta egyenesek meredekségéből leolvasható. Látható, hogy amennyiben az $\omega \ll 3$ esetben a frekvencia egy nagyságrenddel nő, akkor a hiba válasz a

$k = 1, 2$ és 3 hatványkitevők esetén rendre $20, 40$, és 60 decibel lesz. Irányításelméletben az így kapott meredekségekre a $\frac{\text{dB}}{\text{dec}}$ mértékegységet szokás használni.

6.3. Szabályozók az aszimptotikus hibamodellre

Az aszimptotikus hibamodell alapfeltevése, hogy a lokális hiba jól közelíthető az

$$r_{n+1} \approx \varphi_n h^k$$

formulával. Ebben az esetben az irányítandó folyamat átviteli függvénye az egyszerű,

$$G(z) = k$$

alakot ölti, ezáltal a zárt rendszer átviteli függvényeire a következő képletek adódnak. A skálázott lépéshossz átviteli függvénye

$$-kH_\varphi(z) = \frac{kP(z)}{(z-1)Q(z) + kP(z)},$$

míg a hiba átviteli függvénye

$$R_\varphi(z) = \frac{(z-1)Q(z)}{(z-1)Q(z) + kP(z)}$$

alakú lesz.

6.3.1. Rendfeltételek

Mint említettük, a gyakorlati tapasztalatok alapján a lépéshossz választáshoz nem érdemes túl hosszan a múltba tekinteni, ezért a gyakorlatban használt szabályozók dinamikai rendje általában legfeljebb 3. Ekkor a lépésközfrisstő képlet a

$$h_{n+1} = \left(\frac{\text{TOL}}{r_n}\right)^{\frac{\beta_0}{k}} \left(\frac{\text{TOL}}{r_{n-1}}\right)^{\frac{\beta_1}{k}} \left(\frac{\text{TOL}}{r_{n-2}}\right)^{\frac{\beta_2}{k}} \left(\frac{h_n}{h_{n-1}}\right)^{-\alpha_1} \left(\frac{h_{n-1}}{h_{n-2}}\right)^{-\alpha_2} h_n \tag{6.2}$$

alakot ölti.
Az így kapott szabályozó család tagjai osztályozhatók a szabályozók rendjei szerint. A különböző rendek eléréséhez szükséges az α_i, β_i paraméterekre vonatkozó összefüggéseket hívjuk rendfeltételeknek. A rendfeltételek nem csak a szabályozótól, hanem az irányítandó folyamat dinamikájától, azaz a hibamodelltől is függnek. Következzenek most az aszimptotikus hibamodellt feltételezve az erre a szabályozó családra vonatkozó rendfeltételek.

Adaptivitási rendfeltételek

Az adaptivitási rend azt mondja meg, hogy az 1 hányszoros gyöke $R_\varphi(z)$-nek. Tehát konstrukciónkban $p_A \geq 1$. A rend növeléséhez a $Q(z)$ polinomot $(z-1)$-es tényezőkkel kell szoroznunk, így kapjuk a következő feltételeket:

$$\begin{aligned} p_A = 2 \quad &\Leftrightarrow \quad \alpha_1 + \alpha_2 = -1, \\ p_A = 3 \quad &\Leftrightarrow \quad \alpha_1 = -2,\ \alpha_2 = -1. \end{aligned}$$

Lépéshossz szűrési rendfeltételek

A p_F szűrési rend azt mondja meg, hogy a (-1) hányszoros gyöke P-nek. Növelésével a magas, π körüli frekvenciás zaj kiszűrhető a lépéshosszok alkotta jelből. Növeléséhez a P polinomot kell $(z+1)$-es tényezőkkel szorozni, így adódnak a következő feltételek:

$$\begin{aligned} p_F = 1 \qquad &\Leftrightarrow \qquad \beta_0 - \beta_1 + \beta_2 = 0, \\ p_F = 2 \qquad &\Leftrightarrow \qquad \beta_0 = \beta_1/2 = \beta_2. \end{aligned}$$

Hiba szűrési rendfeltételek

A p_R a hiba magas frekvenciás komponenseinek szűrési rendje. Azt mondja meg, hogy a (-1) hányszoros gyöke R_φ-nek. Növelésével a magas, π körüli frekvenciás zaj kiszűrhető a hibabecslések alkotta jelből. Növeléséhez az R_φ gyökei közül kell néhányat (-1)-be helyezni. Az adódó feltételek tehát a következők:

$$\begin{aligned} p_R = 1 \qquad &\Leftrightarrow \qquad \alpha_1 - \alpha_2 = 1, \\ p_R = 2 \qquad &\Leftrightarrow \qquad \alpha_1 = 2,\ \alpha_2 = 1. \end{aligned}$$

6.3.2. A H211 szabályozó család

A digitális szűrő alapú szabályozók legfontosabb tulajdonságait a (p_D, p_A, p_F) hármassal szokás összefoglalni. Egy gyakorlatban jól használható típus a $(2,1,1)$ rendekkel rendelkező, vagyis a H211 szabályozók családja.
Mivel a dinamikai rend 2, az adaptivitási rendre $p_A \leq 2$, míg a szűrési rendre $p_F \leq 1$ teljesül. A rendfeltételekbe behelyettesítve adódik a

$$p_F = 1 \qquad \Leftrightarrow \qquad \beta_0 = \beta_1$$

összefüggés, így a lépéshosszfrissítő képlet a

$$\begin{aligned} \rho_n &= c_n^{\beta_0} c_{n-1}^{\beta_0} \rho_{n-1}^{-\alpha_1} \\ h_{n+1} &= \rho_n h_n \end{aligned}$$

alakot ölti.

A H211 család analízise

A szabályozó átviteli függvénye

$$C(z) = \frac{\beta_0}{k(z-1)} \frac{z+1}{z+\alpha_1},$$

így a szűrőt alkotó polinomokra a

$$\begin{aligned} P(z) &= \frac{\beta_0}{k}(z+1), \\ Q(z) &= z + \alpha_1 \end{aligned}$$

képlet adódik, amiből rövid számolás és egyszerűsítés után következik hogy a zárt rendszer átviteli függvényei

$$\begin{aligned} -kH_\varphi(z) &= \frac{\beta_0(z+1)}{z^2 + (\alpha_1 + \beta_0 - 1)z + \beta_0 - \alpha_1}, \\ R_\varphi(z) &= \frac{(z-1)(z+\alpha_1)}{z^2 + (\alpha_1 + \beta_0 - 1)z + \beta_0 - \alpha_1} \end{aligned}$$

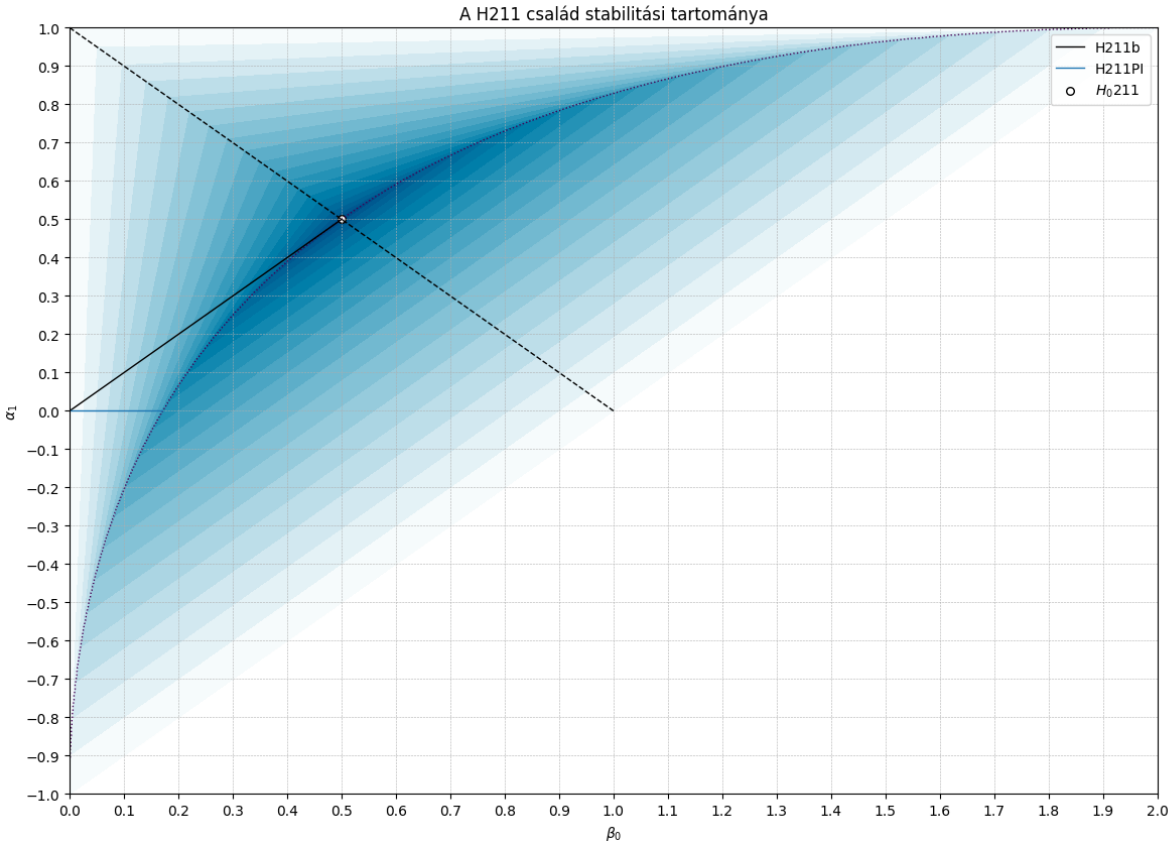
alakúak.

Paraméter választás

A megfelelő paraméterezés megtalálása során a legfontosabb szem előtt tartandó kritérium az, hogy a zárt rendszer stabil legyen. Ezt a zárt rendszer átviteli függvényeinek pólusainak elhelyezésével (pole placement) tudjuk biztosítani, azaz a

$$z^2 + (\alpha_1 + \beta_0 - 1)z + \beta_0 - \alpha_1 \tag{6.3}$$

polinom gyökeinek helyét kell megfelelően beállítani a két szabad paraméter β_0, α_1 megválasztásával. A problémát a 6.3. ábra szemlélteti. Célunk olyan paraméterezést találni, amellyel egyrészt a pólusok pozitív valósrészűek, sőt ideális esetben valósak is, hogy a szabályozó ne adjon oszcillációkat a rendszerhez. Másrészt a pólusok minél közelebb lesznek az origóhoz, hogy a keletkező rendszer elegendően stabil legyen.



6.3. ábra. A H211 család stabilitási tartománya. A színezés intenzitása a domináns gyök hosszának szintvonalait ábrázolja. A szintvonalakat a $\{0, 0.05, \ldots, 0.95, 1\}$ értékekre ábrázoltuk és fehér színnel jelöltük azt, amikor a domináns gyök hossza egynél nagyobb. A valós gyökök a pontozott görbe feletti paramétereknek felelnek meg. A domináns gyök valósrésze pozitív, amennyiben a szaggatott vonal alatti paramétereket választunk. Az ábrán fekete vonal jelöli a H211b paramétereinek megfelelő szakaszt; a család speciális esete a $b = 2$ paraméterű H_0211 szabályozó, ami egy fehér körrel van jelölve. Kék vonal ábrázolja a H211PI család paraméterterét. Látható, hogy az $\alpha_1 = 0$ kikötés után β_0 értékeire viszonylag kevés lehetőség marad, ezek között a legstabilabb valós gyökökkel és pozitív valósrészű domináns gyökkel rendelkező eset a $\beta_0 \approx \frac{1}{6}$ paraméterhez tartozik.

H_0211

A legstabilabb dinamikát akkor kapjuk, ha a (6.3) polinomnak kétszeres gyöke a 0. Ez a

$$\beta_0 = \alpha_1 = \frac{1}{2}$$

paraméterezés esetén áll fenn. Az így kapott szűrő véges válaszidejű (FIR).

H211b

Egy további érdekes típus adódik a $b > 0$ paraméter bevezetésével. Válasszuk konstansainkat a következő módon:

$$\beta_0 = \alpha_1 = \frac{1}{b},$$

ekkor kapjuk a H211b családot. A családhoz tartozó átviteli függvények a

$$-kH_\varphi(z) = \frac{(z+1)/b}{z^2 - (1-2/b)z},$$
$$R_\varphi(z) = \frac{(z-1)(z+1/b)}{z^2 - (1-2/b)z}$$

alakot öltik, azaz a pólusok a

$$z = 0, \qquad z = 1 - 2/b$$

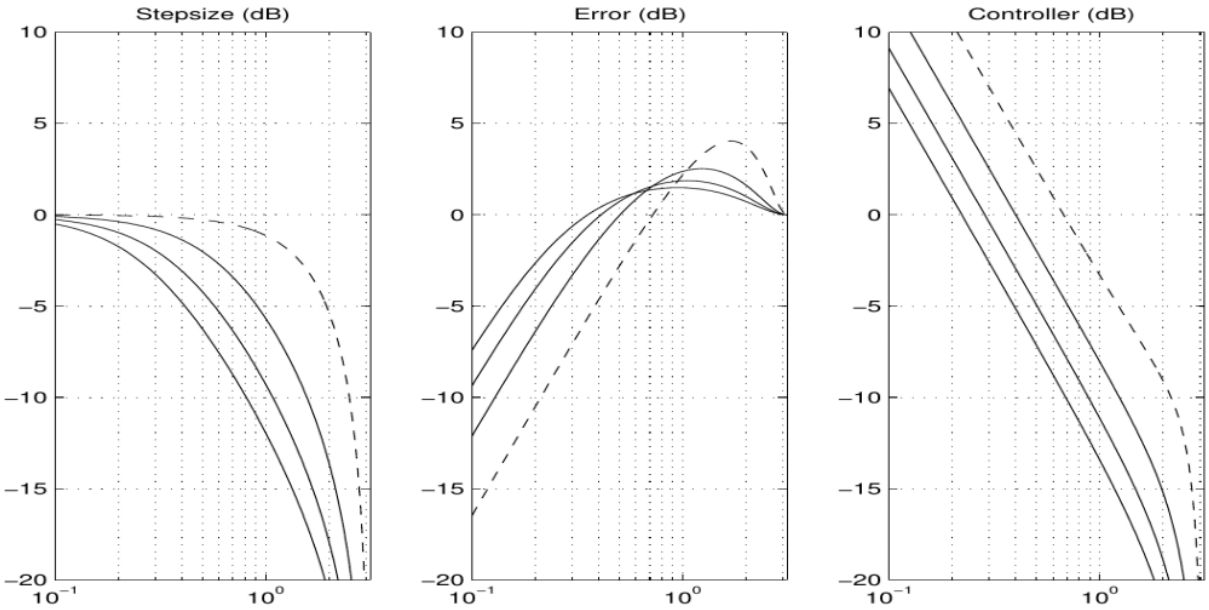
helyeket foglalják el, így két valós pólusról van szó. A paraméter függvényében a következők mondhatók.

- Ha $b < 2$, akkor a mozgatható pólus negatív lesz, ez oszcillációkat ad a rendszerhez, ezért ezt érdemes elkerülni.
- Ha $b = 2$, akkor visszakapjuk a H_0211 szabályozót.
- Ha $b > 2$, akkor elmondható, hogy b fokozatos növelésével a szabályozó egyre simább lépéshosszokat generál, viszont ezzel párhuzamosan a válaszideje is nő.

Az utóbbi jelenséget szemlélteti a 6.4. ábra.
Az ábrán látható, hogy b növelésével a lépéshosszok alkotta jel magas frekvenciás komponensei intenzívebb szűrésen mennek át, de cserébe a rendszer lassabban adaptálódik a feladathoz megfelelő lépéshossz nagyságrendjéhez, azaz az alacsony frekvenciás (közel konstans) komponensek kevésbé intenzíven lesznek szűrve a hibák alkotta jelből.
Összességében tehát a lépéshosszfrissítő képlet ebben az esetben a

$$\rho_n = c_n^{1/b} c_{n-1}^{1/b} \rho_{n-1}^{-1/b}$$
$$h_{n+1} = \rho_n h_n$$

alakot ölti, ahol a b paraméterre nézve a gyakorlatban a $2 \leq b \leq 8$ választás jó eredményeket hoz. Elmondható, hogy ezek közül a $b = 4$ használata ajánlott.



6.4. ábra. A H211b szabályozó frekvencia-válasza [19]. A $b = 2, 4, 6, 8$ paraméterek esete látható, a $b = 2$ paraméternek a szaggatott vonal felel meg. A baloldali ábrán a skálázott lépéshossz frekvencia-választ, középen a hiba frekvencia válaszát, a jobb oldalon pedig a szabályozó frekvencia-válaszát láthatjuk. Elmondható, hogy b növelésével a szabályozó erősítése csökken, és mind a lépéshosszok jeléből, mind pedig a hibajelből a magasabb frekvenciák egyre jobban kiszűrődnek, azaz a keletkező görbék egyre simábbak lesznek. Ezzel párhuzamosan a hiba frekvencia-válaszából látható, hogy a hibajelből az alacsony frekvenciás komponensek egyre lassabban szűrődnek ki, azaz a rendszer egyre lassabban képes adaptálni a lépéshosszokat a megfelelő nagyságrendre. Az arany középutat a gyakorlatban a $b = 4$ választás jelenti.

H211PI

A család további érdekes tagjai az $\alpha_1 = 0$ választással adódó, PI struktúrával rendelkező,

$$\rho_n = c_n^{\beta_0} c_{n-1}^{\beta_0}$$
$$h_{n+1} = \rho_n h_n$$

képlettel definiált szabályozók. Az általános H211 esetre vonatkozó képletből behelyettesítéssel adódnak a zárt rendszer átviteli függvényeire vonatkozó formulák:

$$-kH_\varphi(z) = \frac{\beta_0(z+1)}{z^2 + (\beta_0 - 1)z + \beta_0},$$
$$R_\varphi(z) = \frac{z(z-1)}{z^2 + (\beta_0 - 1)z + \beta_0}.$$

A pólusok tehát csak a β_0 paramétertől függnek. Ezt az oszcillációk elkerülése végett úgy érdemes megválasztanunk, hogy pozitív valósrészű, kicsi abszolútértékű pólusokat kapjunk; amennyiben lehet, akkor valósakat. A 6.3. ábrát megvizsgálva láthatjuk, hogy a $\beta_0 = \frac{1}{6}$ választás egyszerű alakú és közel optimális értéket ad, így a H211PI szabályozó alatt a

$$\rho_n = c_n^{1/6} c_{n-1}^{1/6}$$
$$h_{n+1} = \rho_n h_n$$

képlettel definiált szabályozót szokás érteni.

6.4. Szabályozók a dinamikus hibamodellre

6.4.1. A dinamikus hibamodell motivációja

Ismeretes, hogy amennyiben egy numerikus módszer p-edrendben konzisztens, akkor a lokális hibák

$$C \cdot h^{p+1} + \mathcal{O}(h^{p+2})$$

alakúak. Ha feltesszük, hogy h az adott feladattól és numerikus módszertől függő korlát alatt van, azokhoz viszonyítva elég kicsi, akkor a fenti formula második tagjának elhanyagolásával minimális hibát vétünk. Ezen feltevésen alapul az előző részben ismertetett aszimptotikus hibamodell.
A gyakorlatban változó lépésközű módszerek esetén a TOL tolerancia alacsonyra állításával a legtöbb feladat esetén elérhető, hogy a szabályozók a lépéshosszokat olyan alacsonyan tartsák, hogy az aszimptotikus hibamodell jól leírja a lokális hibákat.
Azokban az esetekben, amikor a lépéshosszok ilyen mértékű restrikciója nem kívánatos, például ha a felhasználónak egy kevésbé precíz megoldás is megfelelő, vagy ha a feladat struktúrájából adódóan az adott módszerrel csak alacsonyabb rendű konvergencia érhető el realisztikus lépéshosszok választása mellett, akkor az egyszerű aszimptotikus hibamodell már nem írja le jól a lokális hibák viselkedését, ezért ehelyett ún. dinamikus hibamodelleket használhatunk. A megfelelő dinamikus hibamodell feladattípusonként és megoldó módszerenként különböző lehet.

Merev feladatok

A dinamikus hibamodellek szükségességének egy tipikus példája a merev feladatok explicit módszerekkel való megoldása.
Merev feladatok esetén az aszimptotikus hibamodellre épülő hibabecslések a hibát jelentősen alulbecsülik, ezért jellemzően a szabályozó a lépéshosszakat fokozatosan növeli, így idővel a számítás elhagyja a stabilitási tartományt. Ekkor a lokális hibák ugrásszerűen megnőnek, ezáltal a lokális hibabecslések jelentősen alulbecsült változata is eléri, sőt jellemzően túllépi a tolerancia küszöböt, így a szabályozó elkezdi csökkenteni a lépéshosszokat. Azonban mire a stabilitási tartományba visszaérve a felgyülemlett hibák kellőképpen lecsillapodnak a lépéshossz már jellemzően igen lecsökkent, így a kapott hibabecslések arra ösztönzik a szabályozót, hogy a lépéshosszokat újra növelni kezdje, így a folyamat kezdődik előről és egyfajta oszcilláció alakul ki a stabilitási tartomány határa felett.
A következőkben arról lesz szó, hogy hogyan lehet egy dinamikus hibamodell megfelelő megválasztásával elérni, hogy a kialakult oszcilláció minél finomabb legyen, illetve, hogy hogyan lehet olyan szabályozót tervezni, ami képes a számítási folyamatot lényegében a stabilitási tartomány határán balanszírozni.

6.4.2. Egy elsőrendű dinamikus hibamodell

A dinamikus hibamodell felállításához a lineáris differenciaegyenletek és a lineáris irányításelmélet eszközeit fogjuk használni.
Az ilyen keretek között alkotható legegyszerűbb dinamikus hibamodell egy, a $\log r$ változóban elsőrendű differenciaegyenlet lesz, melynek jobboldala függ a megoldandó differenciálegyenlettől és a megoldó módszertől is.
A merev feladatok modellezésére a Dahlqhuist-féle tesztegyenletet használjuk negatív λ paraméterrel.
A numerikus megoldómódszerünk egy explicit Runge–Kutta módszerek alkotta beágyazott pár lesz. A módszerek stabilitási függvénye legyen P és Q. Ekkor a numerikus megoldásokra a

$z_n = h_n\lambda$ jelölés mellett teljesül, hogy

$$x_{n+1} = P(z_n)x_n,$$
$$\hat{x}_{n+1} = Q(z_n)x_n.$$

Így a lokális hibabecslés az

$$r_n = x_{n+1} - \hat{x}_{n+1} = (P-Q)(z_n)x_n = E(z_n)x_n$$

alakot ölti. Innen $\log r$-ben elsőrendű lineáris differenciaegyenlet a következőképpen kapható:

$$\begin{aligned} r_n &= E(z_n)x_n = E(z_n)P(z_{n-1})x_{n-1} \\ &= \frac{E(z_n)}{E(z_{n-1})}P(z_{n-1})r_{n-1}. \end{aligned} \tag{6.4}$$

Célunk most tehát az, hogy a lépéshosszokat úgy válasszuk meg, hogy a számítás a numerikus módszerünk stabilitási tartományának határa körül maradjon. Pontosabban a stabilitási tartomány határán lévő $z_* = h_*\lambda$ pont közelében szeretnénk maradni, mely megoldása a

$$|P(z)| = 1, \quad \mathbb{C} \ni z : \arg z = \arg \lambda$$

egyenletnek. Ezért keressük a h_n lépéshosszokat a

$$h_n = h_*(1+\varepsilon_n)$$

alakban. A modell elkészítéséhez a (6.4) egyenletet linearizálni, majd delinearizálni fogjuk. Először tehát az egyenlet tényezőit z_* körül linearizálva adódnak az

$$\begin{aligned} E(z_*(1+\varepsilon_n)) &\approx E(z_*) + E'(z_*)z_*\varepsilon_n \\ &= \left(1 + \frac{E'(z_*)}{E(z_*)}z_*\varepsilon_n\right)E(z_*) \\ &= (1 + c_E\varepsilon_n)\,E(z_*), \end{aligned}$$

alakú formulák, ahol

$$c_E = \frac{E'(z_*)}{E(z_*)}z_*.$$

Felhasználásukkal kapjuk az

$$\begin{aligned} r_n &= \frac{E(z_n)}{E(z_{n-1})}P(z_{n-1})r_{n-1} \\ &\approx \frac{1+c_E\varepsilon_n}{1+c_E\varepsilon_{n-1}}(1+c_P\varepsilon_{n-1})P(z_*)r_{n-1} \end{aligned}$$

közelítést. Innen a megfelelő tényezőket az

$$1 + c_E\varepsilon_n \approx (1+\varepsilon_n)^{c_E}$$

alakú közelítésekkel lecserélve kapjuk végül a $\log r$ változóban lineáris differenciaegyenletet:

$$\begin{aligned} r_n &\approx \frac{1+c_E\varepsilon_n}{1+c_E\varepsilon_{n-1}}(1+c_P\varepsilon_{n-1})P(z_*)r_{n-1} \\ &\approx (1+\varepsilon_n)^{c_E}(1+\varepsilon_{n-1})^{-c_E}(1+\varepsilon_{n-1})^{c_P}P(z_*)r_{n-1} \\ &= (1+\varepsilon_n)^{c_E}(1+\varepsilon_{n-1})^{c_P-c_E}P(z_*)r_{n-1}. \end{aligned} \tag{6.5}$$

Az egyszerűség kedvéért az explicit módszereknek az 1.2. táblázatban megadott RK4 és RK3 módszereket választva az 1.1. ábra alapján $z_* \approx -2.785$ teljesül. Továbbá a megfelelő polinomok

$$\begin{aligned} P(z) &= 1 + z + \frac{z^2}{2} + \frac{z^3}{6} + \frac{z^4}{24}, \\ Q(z) &= 1 + z + \frac{z^2}{2} + \frac{z^3}{6}, \\ E(z) &= \frac{z^4}{24} \end{aligned}$$

alakúak lesznek, így $P(z_*) = 1$ is fennáll. A kitevőben szereplő konstansokra a

$$c_E = \frac{E'(z_*)}{E(z_*)} z_* = \frac{z_*^3/6}{z_*^4/24} z_* = 4$$

és a

$$c_P = \frac{P'(z_*)}{P(z_*)} z_* = \frac{P(z_*) - E(z_*)}{P(z_*)} z_* = \frac{1 - E(z_*)}{1} z_* \approx 4.2$$

becslés számolható, amikkel a (6.5) hibamodell az

$$\begin{aligned} r_n &\approx (1 + \varepsilon_n)^4 (1 + \varepsilon_{n-1})^{\frac{1}{5}} r_{n-1} \\ &= \left(\frac{h_n}{h_*}\right)^4 \left(\frac{h_{n-1}}{h_*}\right)^{\frac{1}{5}} r_{n-1} \end{aligned}$$

alakot ölti. Logaritmust véve innen a differenciaegyenlet

$$\log r_n - \log r_{n-1} = 4 \log h_n + \frac{1}{5} \log h_{n-1} - 4.2 \log h_*$$

alakú lesz. Látható, hogy a képletben az aszimptotikus hibamodellhez képest a differenciálegyenlet numerikus megoldásának folyamata φ helyett a h_* maximális stabil lépéshossz formájában jelenik meg.
A formulát a z-transzformáltak nyelvén is kifejezhetjük

$$(z - 1) \log \hat{r} = \left(4z + \frac{1}{5}\right) \log \hat{h} - 4.2 \log \hat{h}_*,$$

innen a folyamat átviteli függvénye könnyen leolvasható:

$$G(z) = \frac{4z + \frac{1}{5}}{z - 1}.$$

A hibamodell alapján azt mondhatjuk, hogy bár a lokális hibák csillapítás nélkül adódnak össze, növekedésük csupán lineáris. Ez pedig megfelelő szabályozók választása esetén kezelhető. Sőt, alkalmas szabályozókkal a h_* körüli oszcilláció minimalizálható és ezáltal sima lépéshossz görbét kaphatunk.

6.4.3. Egy alkalmas PI szabályozó

Induljunk ki a következő PI struktúrájú, azaz a

$$\begin{aligned} \rho_n &= \left(\frac{\mathrm{TOL}}{r_n}\right)^{\beta_0/k} \left(\frac{\mathrm{TOL}}{r_{n-1}}\right)^{\beta_1/k} \\ h_{n+1} &= \rho_n h_n \end{aligned}$$

képlettel leírható szabályozóból, melynek átviteli függvénye

$$C(z) = -\frac{1}{k}\frac{\beta_0 z + \beta_1}{z(z-1)}.$$

Ha célunk az, hogy olyan szabályozót készítsünk, amely képes az aszimptotikus és a fent ismertetett dinamikus, azaz a

$$G_A(z) = k, \qquad\qquad G_D(z) = \frac{4z + \frac{1}{5}}{z-1}$$

átviteli függvényű hibamodellek bármelyikét feltételezve jól működni, akkor úgy kell megválasztanunk a β_0, β_1 paramétereket, hogy a megfelelő zárt rendszerek mindegyike stabil és jó frekvencia-válaszú legyen.

PI3333

A beágyazott explicit Runge–Kutta módszerek irányítására a gyakorlatban sok esetben igen jól használható szabályozó a PI3333, mely a

$$\beta_0 = \frac{2}{3}, \qquad \beta_1 = -\frac{1}{3}$$

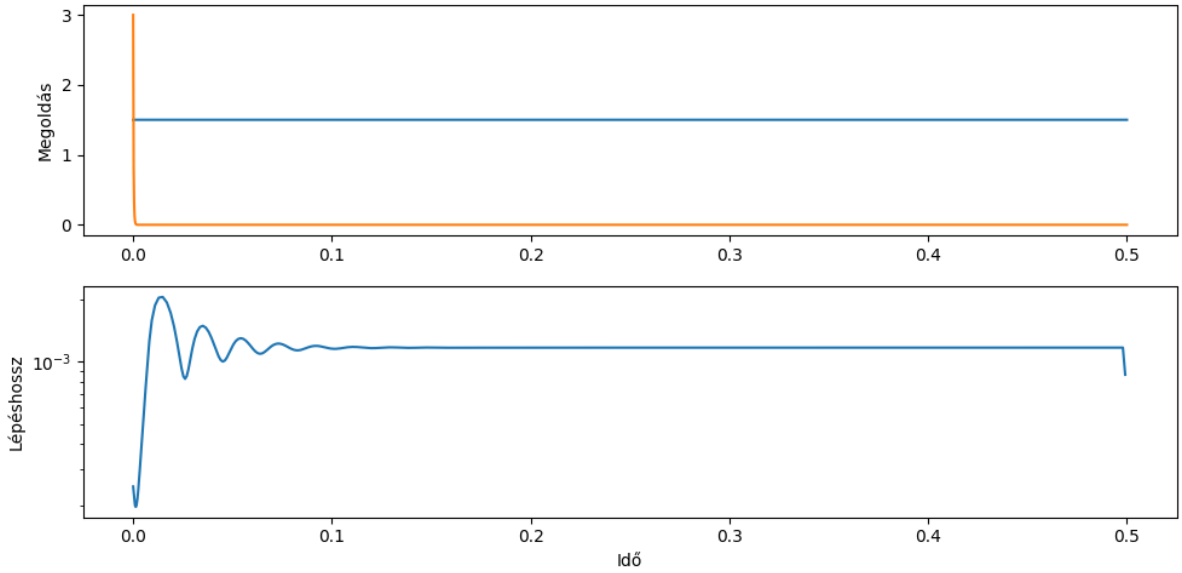
paraméterezés választásnak, és ezáltal a

$$h_{n+1} = \left(\frac{\mathrm{TOL}}{r_n}\right)^{2/3k} \left(\frac{\mathrm{TOL}}{r_{n-1}}\right)^{-1/3k} h_n$$

lépésközfrissítő képletnek felel meg. Érdemes megjegyezni, hogy a legutóbbi lokális hibabecslés egy nagyobb, pozitív súlyt kap, míg az azt megelőző egy kisebb, negatív súllyal van figyelembe véve. Ez a konstrukció akadályozza a lépéshosszok túl hirtelen növekedését és csökkenését, hiszen például ha a $\left(\frac{\mathrm{TOL}}{r_n}\right)$ és a $\left(\frac{\mathrm{TOL}}{r_{n-1}}\right)$ kifejezések mindegyike egynél kisebb, azaz az elmúlt két lépéshossz túl nagy volt, akkor a formulában az első tényező egy erősebb súllyal a lépéshossz csökkentését szorgalmazza, míg a második tényező egy gyengébb súllyal a lépéshossz növeléséért dolgozik.

Numerikus kísérlet

A PI3333 szabályozó tesztelésére a (4.4) problémát felhasználó kísérletet megismételtük úgy, hogy a klasszikus megközelítés helyett a PI3333 szabályozót használtuk. A kísérlet során azt tapasztaltuk, hogy a lépéshosszok oszcillációja lényegében eltűnt. Az eredményeket az 6.5. ábra mutatja.

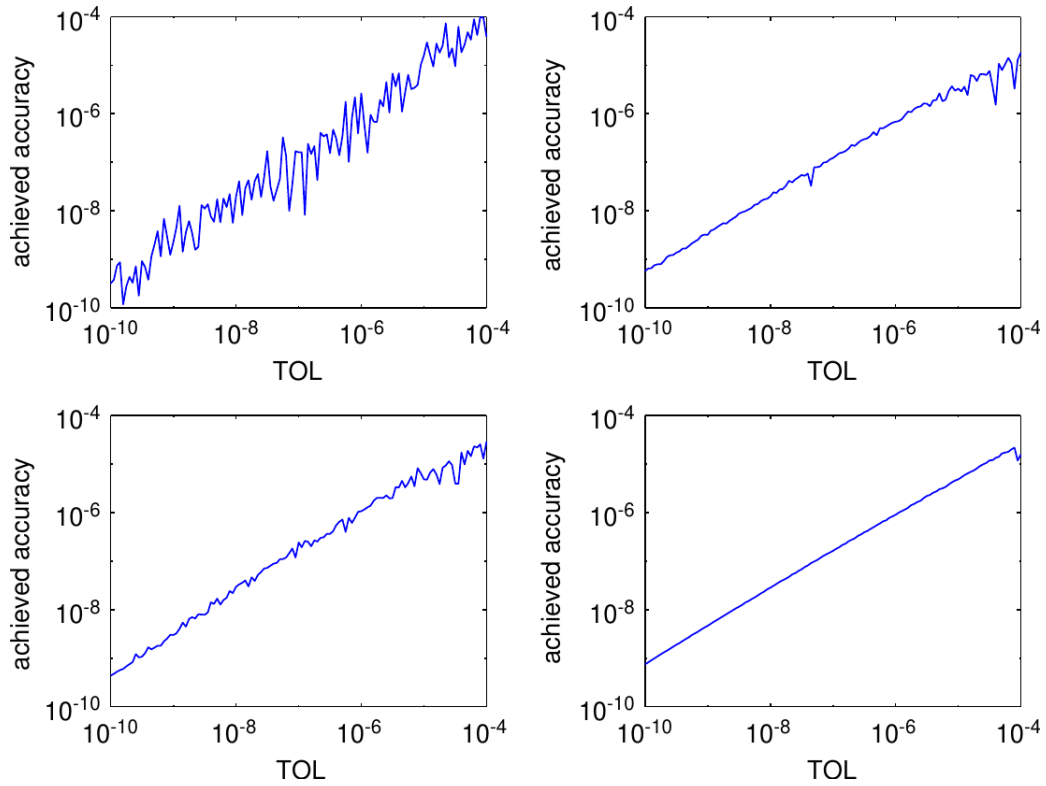


6.5. ábra. A Van der Pol probléma megoldása a PI3333 szabályozóval irányított adaptív DORPRI módszerrel. A felső ábrán látható a megoldás két komponense az idő függvényében. Az alsó ábrán logaritmikus skálán az adott időpillanatban elfogadott lépésközök hossza látható, az értékek törtvonallal összekötve. Vesd össze a 4.3. ábrával.

7. fejezet

Konklúzió

Az állandó lépésközű Runge–Kutta módszerek algoritmusának - számítási és implementációs komplexitási szempontból is - minimális módosításával a módszerek adaptívvá tehetők. Az adaptivitás legszembetűnőbb előnye, hogy a felhasználó pontossági követelményeit közvetlenül a TOL tolerancián keresztül adhatja meg ellentétben az állandó lépésközű módszerek esetével, ahol erre csak közvetetten, a lépéshossz megadásán keresztül van lehetőség. Az adaptív módszerek teljesítménybeli fölénye az olyan feladatok esetén nyilvánul meg igazán, melyeknél az adott pontosságú numerikus megoldás kiszámításához a megoldás mentén helyenként különböző nagyságrendű lépéshosszok használata elegendő. Tipikusan ilyenek a merev feladatok.



7.1. ábra. A [23] cikk kísérlete, melyben az implicit DASSL módszert (bal oldalon) a beépített és (jobb oldalon) a H211b szabályozóval irányítva alkalmazták az igen merev Chemakzo problémára. A felső sorban a Newton-iteráció leállási feltétele lazább, míg az alsó sorban lényegesen szigorúbb volt. A vízszintes tengely mentén a tolerancia, míg a függőleges tengely mentén az elért pontosság látható. Az ábra magáért beszél.

A merev feladatok esetén a tranziens rész kiintegrálásához finomabb lépéshosszokra van szükség, utána viszont nagyobb lépéshosszok használata is hasonló pontossághoz vezet, hiszen a feladat merevsége miatt a feltorlódott hibák nagymértékben csillapodnak. Heurisztikákkal és biztonsági kapcsolókkal terhelten már a klasszikus adaptív megközelítés is rendelkezik ezekkel az előnyökkel, azonban irányításelméleti és jelfeldolgozási eszközök felhasználásával tudományosan megalapozott keretek között tervezhetünk olyan, akár feladattípusra és megoldómódszerre szabott lépésközválasztó eljárásokat, melyek további jó tulajdonságokkal rendelkeznek. Ezek közül a legfontosabb talán az, hogy a megfelelő szabályozó választásával egyfajta számítási stabilitás formájában elérhető, hogy a bemeneti toleranciától lényegében folytonosan függjön a numerikus megoldás hibája. Ezt a jelenséget a 7.1. ábra szemlélteti.

Irodalomjegyzék

Cikkek

[1] Carmen Arévalo és Gustaf Söderlind. “Grid-independent construction of multistep methods”. *Journal of Computational Mathematics* 35.5 (2017), 672–692. old.

[2] John C. Butcher. “Implicit Runge–Kutta processes”. *Mathematics of Computation* 18.85 (1964), 50–64. old.

[4] Jeff R. Cash. “Diagonally implicit Runge–Kutta formulae with error estimates”. *IMA Journal of Applied Mathematics* 24.3 (1979), 293–301. old.

[5] John R. Dormand és Peter J. Prince. “A family of embedded Runge–Kutta formulae”. *Journal of computational and applied mathematics* 6.1 (1980), 19–26. old.

[6] Erwin Fehlberg. “Klassische Runge–Kutta-formeln vierter und niedrigerer ordnung mit schrittweiten-kontrolle und ihre anwendung auf waermeleitungsprobleme”. *Computing* 6.1-2 (1970), 61–71. old.

[7] Ian Gladwell, Lawrence F. Shampine és R. W. Brankin. “Automatic selection of the initial step size for an ODE solver”. *Journal of Computational and Applied Mathematics* 18.2 (1987), 175–192. old.

[11] Christopher A. Kennedy és Mark H. Carpenter. “Diagonally implicit Runge–Kutta methods for ordinary differential equations. a review”. (2016).

[12] Syvert P. Nørsett és Per G. Thomsen. “Switching between modified Newton and fix-point iteration for implicit ODE-solvers”. *BIT Numerical Mathematics* 26.3 (1986), 339–348. old.

[13] Lewis Fry Richardson és J. Arthur Gaunt. “VIII. The deferred approach to the limit”. *Philosophical Transactions of the Royal Society of London. Series A, containing papers of a mathematical or physical character* 226.636-646 (1927), 299–361. old.

[14] Lawrence F. Shampine. “Some practical Runge–Kutta formulas”. *Mathematics of Computation* 46.173 (1986), 135–150. old.

[15] Lawrence F. Shampine és M. K. Gordon. “Some numerical experiments with DIFSUB”. *ACM Signum Newsletter* 7.3 (1972), 24–26. old.

[16] Leonid Markovich Skvortsov. “Diagonally implicit Runge–Kutta methods for stiff problems”. *Computational mathematics and mathematical physics* 46.12 (2006), 2110–2123. old.

[17] Gustaf Söderlind. “Automatic control and adaptive time-stepping”. *Numerical Algorithms* 31.1-4 (2002), 281–310. old.

[18] Gustaf Söderlind. “Control theory, Digital filters, and Signal Processing in Adaptive Time-Stepping”. *Lecture Notes* (2019).

[19] Gustaf Söderlind. “Digital filters in adaptive time-stepping”. *ACM Transactions on Mathematical Software (TOMS)* 29.1 (2003), 1–26. old.

[20] Gustaf Söderlind. “The logarithmic norm. History and modern theory”. *BIT Numerical Mathematics* 46.3 (2006), 631–652. old.

[21] Gustaf Söderlind. “Time-step selection algorithms: Adaptivity, control, and signal processing”. *Applied numerical mathematics* 56.3-4 (2006), 488–502. old.

[22] Gustaf Söderlind, Laurent Jay és Manuel Calvo. “Stiffness 1952–2012: Sixty years in search of a definition”. *BIT Numerical Mathematics* 55.2 (2015), 531–558. old.

[23] Gustaf Söderlind és Lina Wang. “Adaptive time-stepping and computational stability”. *Journal of Computational and Applied Mathematics* 185.2 (2006), 225–243. old.

[25] Herman A. Watts. “Starting step size for an ODE solver”. *Journal of Computational and Applied Mathematics* 9 (1983. jún.).

Könyvek

[3] John C. Butcher és Nicolette Goodwin. *Numerical methods for ordinary differential equations*. 2. köt. Wiley Online Library, 2008.

[8] Kjell Gustafsson. *Control of error and convergence in ODE solvers*. Licentiate thesis, Lund, 1992.

[9] Ernst Hairer, Syvert P. Nørsett és Gerhard Wanner. *Solving Ordinary Differential Equations I: Nonstiff Problems*. Springer Series in Computational Mathematics. Springer Berlin Heidelberg, 2008.

[10] Ernst Hairer, Syvert P. Nørsett és Gerhard Wanner. *Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems*. Lecture Notes in Economic and Mathematical Systems. Springer, 1993.

[24] Eduardo D. Sontag. *Mathematical control theory: deterministic finite dimensional systems*. 6. köt. Springer Science & Business Media, 2013.