SZÁMÍTÓGÉPES SZIMULÁCIÓK

Harmonikus oszcillátor
jegyzőkönyv

Jegyzőkönyvet készítette:
Koroknai Botond (AT5M0G)

Jegyzőkönyv leadásának időpontja:
2023.12.11

1. A kód

Első teendőként elolvastam a feladatot, hogy milyen szempotok alapján kell majd vizsgálnom a Harmonikus oszcillátor viselkedését. Arra tekintettel, hogy az egyik részfeladatban a sima Euler algoritmust is használnom kell, a következő módosításokat végeztem elsősorban a kódon:

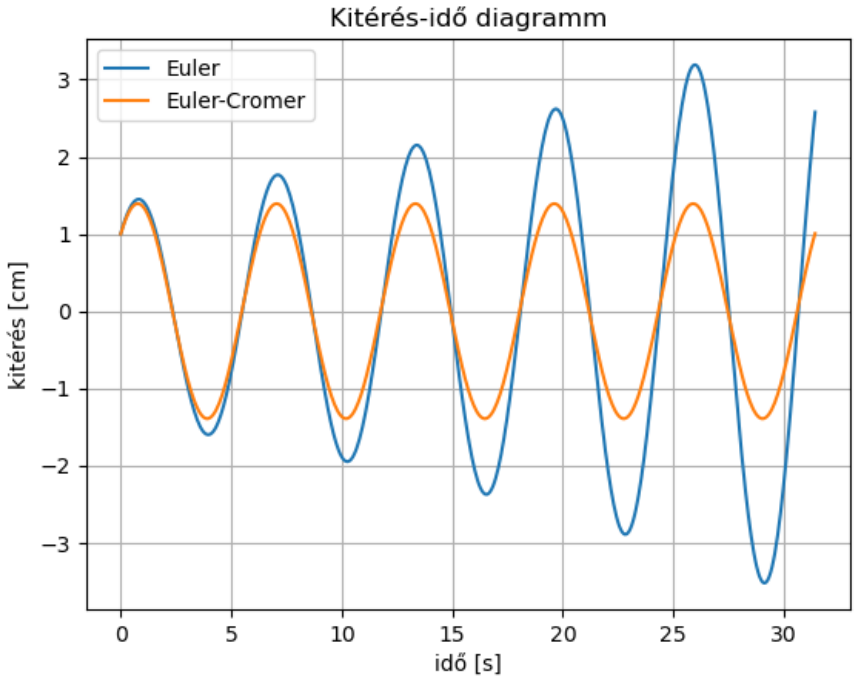
- Definiáltam egy új függvényt ami a sima Euler algorimtust valósítja meg, azaz először frissíti a helyzetet, és az új helyzetnek megfelelően a sebességet.
- Továbbá pár apróbb technikai változtatást is végeztem: 2 file nevet kér be inputként, és kettő .txt file lesz az output a két módszernek megfelelően.

2. A harmonikus oszcillátor kitérés-idő diagrammja

Az egyszerűség kedvéért az ω, $x(0)$, és $v(0)$ paramétereket mind 1-nek, míg a stepsPerPeriod változó értékét 100-nak választottam, ezzel viszonylag pontos, de nem túl időigényes szimulációkat hoztam létre. Az első feladat a kitérés-idő diagramm vizsgálata volt, melyet két féle képpen tettem meg:

2.1. Rövid intervallum

Ezen szimuláció során 5 periodus ideig futtattam az algoritmust.

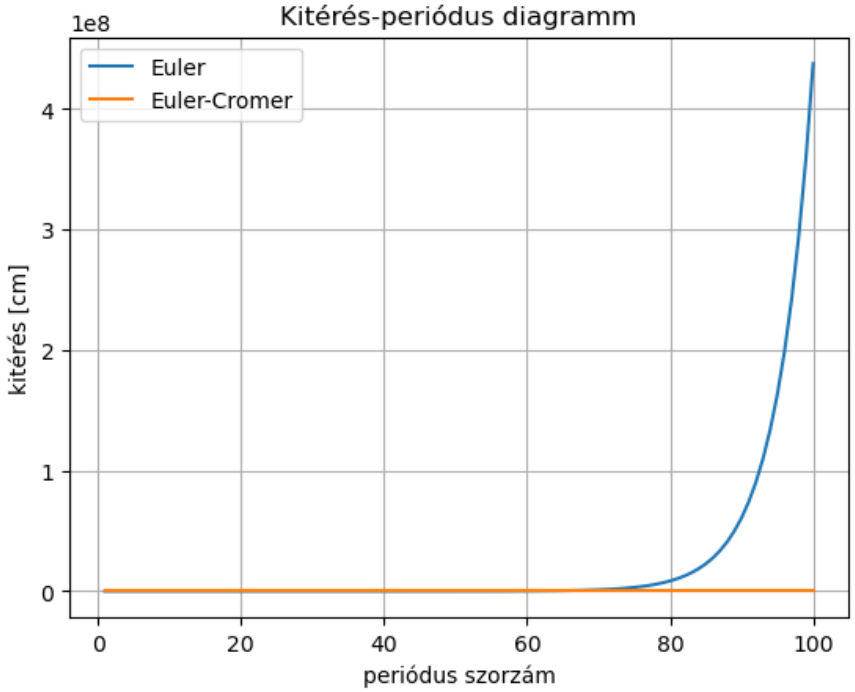


Az ábrához, valamint az eredményekhez két fontos megjezésem lenne:

- A kitérés skáláját cm-nek választottam, ugyanis a harmonikus oszcillátort leíró differenciálegyenlet, amely a szimulációban is szerepel, egy olyan lineáris közelítést használ, amely a Hooke-törvényen alapszik. Nagy kitérítés esetén azonban megjelennének bizonyos anharmonikus tagok is, melyekeket szint úgy figyelembe kellene vennünk a viselkedés modellezésekor
- A másik fontos megemlíteni való a két algoritmus stabilitásának külöbsége. Az Euler-Cromer algoritmus jobban megőrzi a fizikai rendszerek energetikai tulajdonságait a hosszú távú numerikus integrálás során, mert egy lépésben először a sebességet frissíti, majd ezt a frissített sebességet használva számítja ki a következő pozíciót, így jobb közelítést ad az időbeli fejlődésre. Ezzel szemben a sima Euler algortimus pont fordított sorrendben dolgozik, így nem veszi figyelembe, hogy a sebesség már megváltozott a lépés során. Mivel a pontosság érdekében sok lépésről beszélünk, jelen esetben 100/periódus, így ezek a kis hibák gyorsan összeadódnak és ez energia növekedéshez vagy csökkenéshez vezethet a rendszerben.

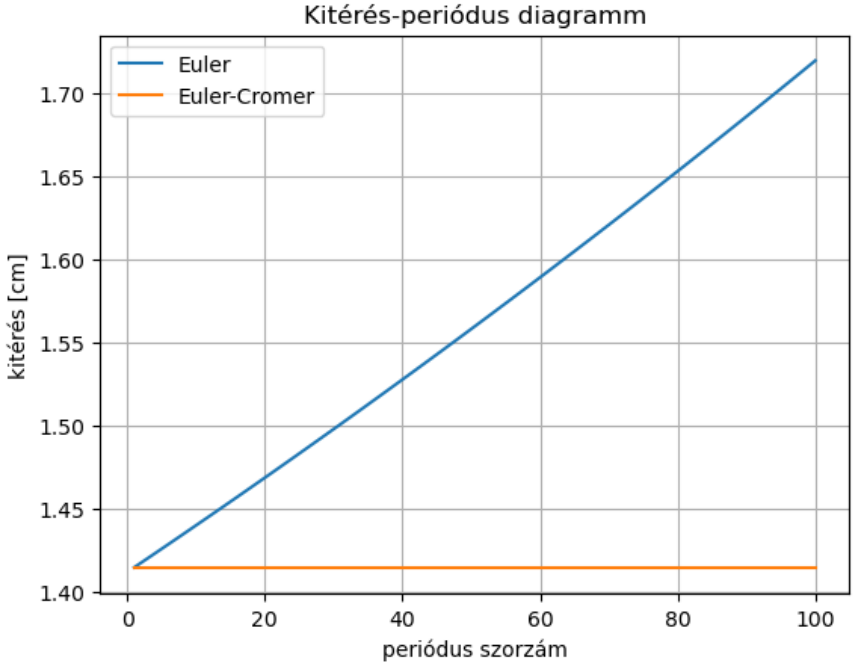
2.2. Hosszú intervallum

A második szimuláció során a paramétereket változtatlanul hagytam egydül a periódusok számát emeltem meg 100-ra. Az eredmények vizualizálása során egy új módszert alkalmaztam, ahelyett, hogy a hullámokat megint teljes egészükben ábrázoltam volna, a scipy.signal csomag segítségével kiszűrtem az amplitudóhoz tartozó értékéket és ezeket ábrázoltam a periódusok sorszámának (lényegében az idő) függvényében.



A tapasztalatom az, hogy a ”hosszú” intervallumú szimuláció nagyon szépen rávilágít az Euler módszer megbízhatatlanságára. Hiába tűnhet úgy, hogy az idő előrehaladtával lineárisan mindig csak egy kis hibát okoz, a valóság az, hogy ez az eltérés egy ponton exponenciálisan elszáll.

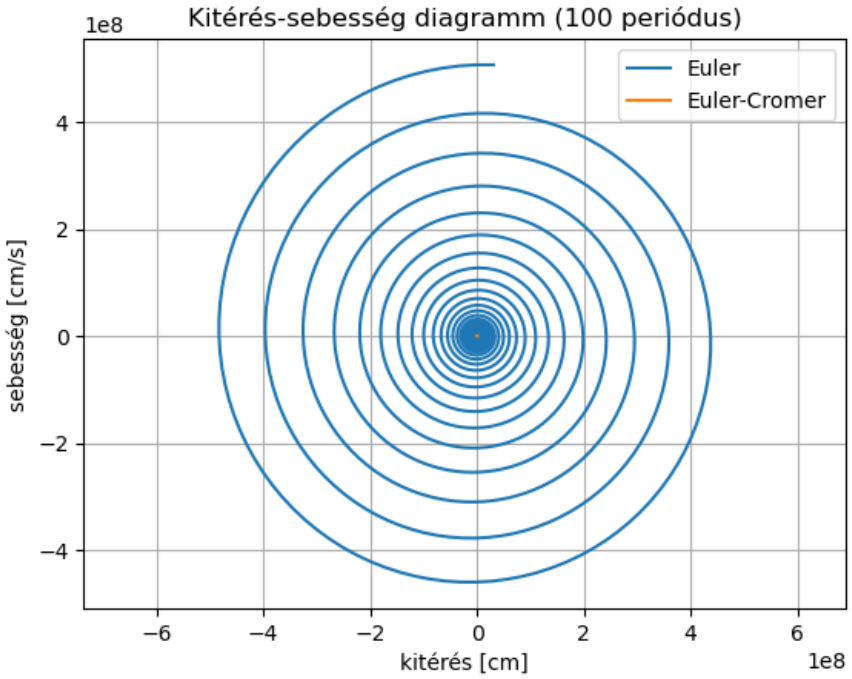
Kíváncsiságból kipróbáltam mi fog történni, ha még ”pontosabban” szimulálom a rendszert, ezért a stepsPer-Period paraméter értékét megnöveltem 10000-re, ezzel jelentősen csökkentettem a lépésközt.



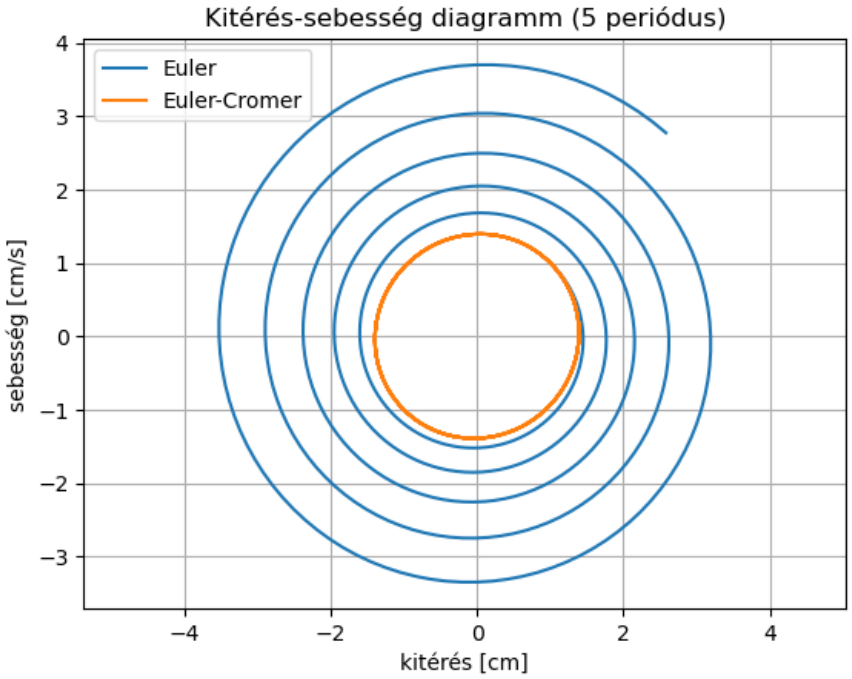
Mint láthatjuk a lépésköz drasztikus csökkenésével az integrálási hiba is kontrollálható. Még további növeléssel egyre pontosabb és pontosabb eredményeket érhetnénk el, de természetesen ez a számításhoz szükséges idő és a fájlok méretének növekedésével jár.

3. Kitérés-sebesség diagram

A kiértékelés során vettem a 100 periódus hosszú szimululáció tartozó időket és ábrázoltam a sebességet a kitérés függvényében.



Az átláthatóság érdekében a rövid intervallumú szimuláció ábráját is elkészítettem.

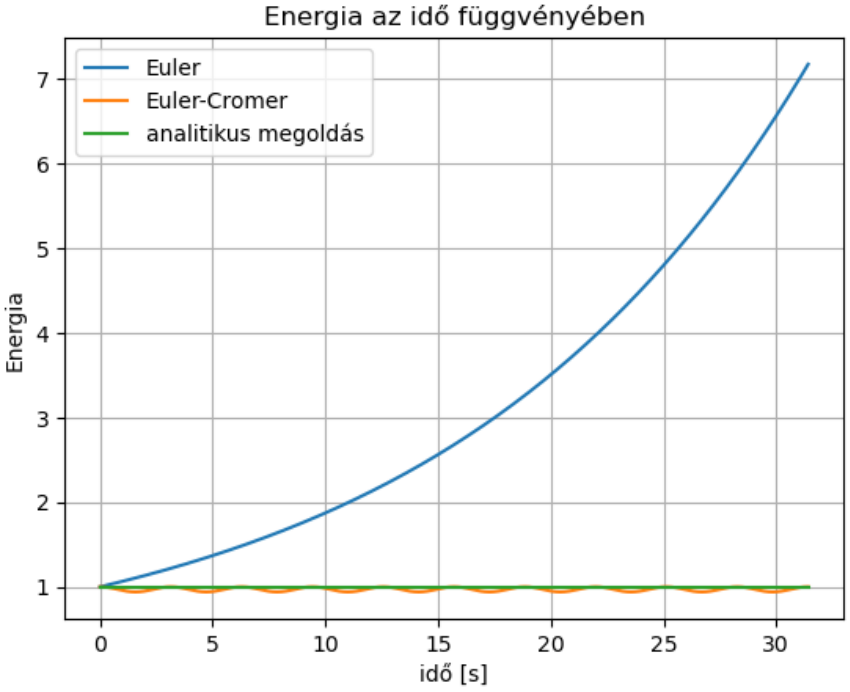


A harmonikus oszcillátor ideális, az energiamegmaradás törvényét követő fázistér-diagramjának, tipikusan ellipszisek vagy körök formájában kéne, hogy megjelenjen, melyek az energiaszinteket képviselik. Ezzel szemben az Euler módszerrel kapott eredmény fázistér-diagramja spirál alakú, ami azt mutatja, hogy a szimulált

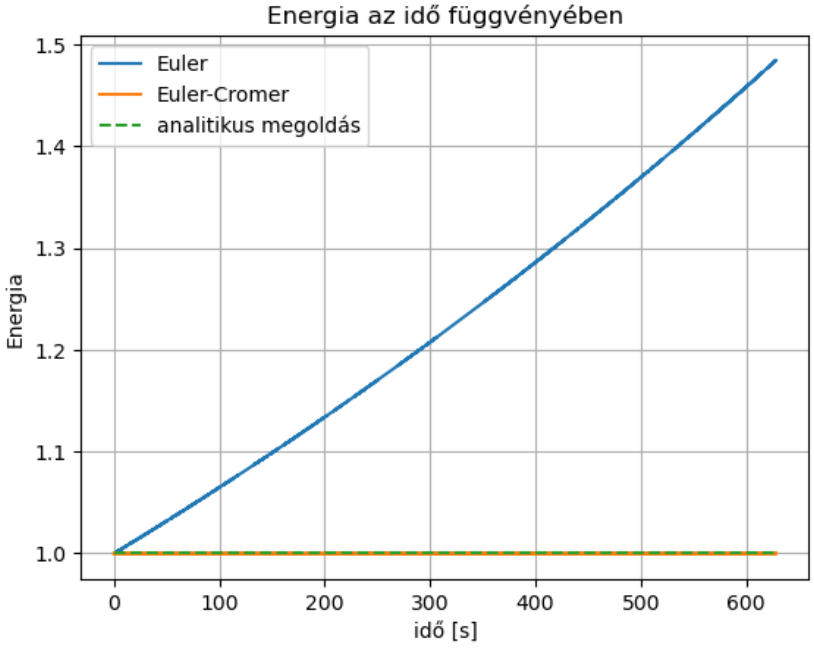
rendszer energiája az idő előrehaladtával nő, ezzel sérti az energiamegmaradást. Ezek alapján jól látszik, hogy az Euler módszer alkalmatlan a hosszú távú szimulációkra.

4. Energiamegmaradás

A harmadik részfeladat során az energia időfejlődését kell, hogy megvizsgáljuk. Az eddigi tapasztalataim alapján az energia megmaradás az sima Euler algoritmust használva nem fog teljesülni.



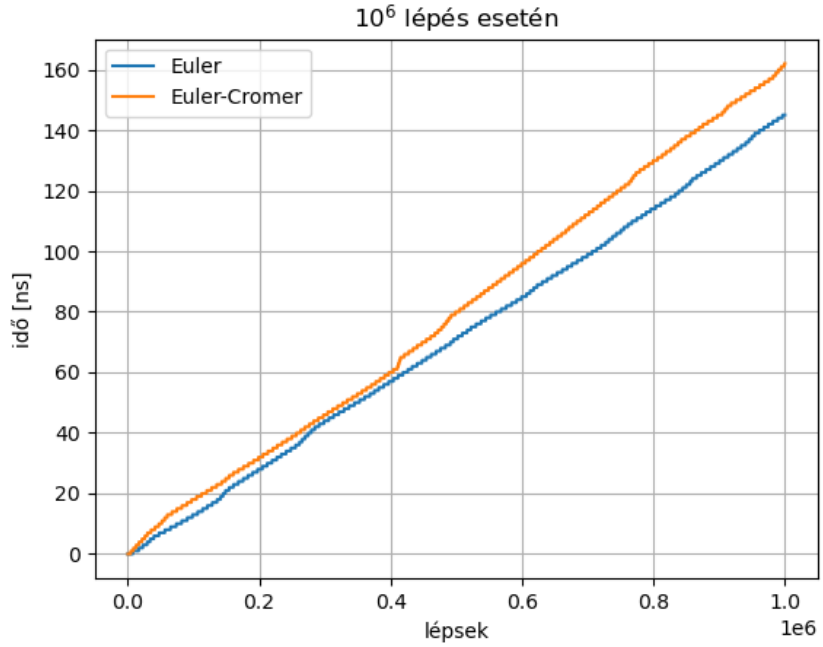
A korábbi sejtéseim beigazolódtak, már a rövid intervallumú szimuláció adatait használva is tökéletesen kirajzolódik, az energia exponenciális növekedése. Ami külön érdekes, hogy ha expliciten az energia értékét vizsgáljuk, akkor látszik, hogy az Euler-Cromer algoritmus sem tökéletes, hiába látszott a kitérés-idő, és kitérés-sebesség diagrammokon annak. Természetesen ez a minimális hiba itt is megszűntethető ha csökkentjük a lépésközt.



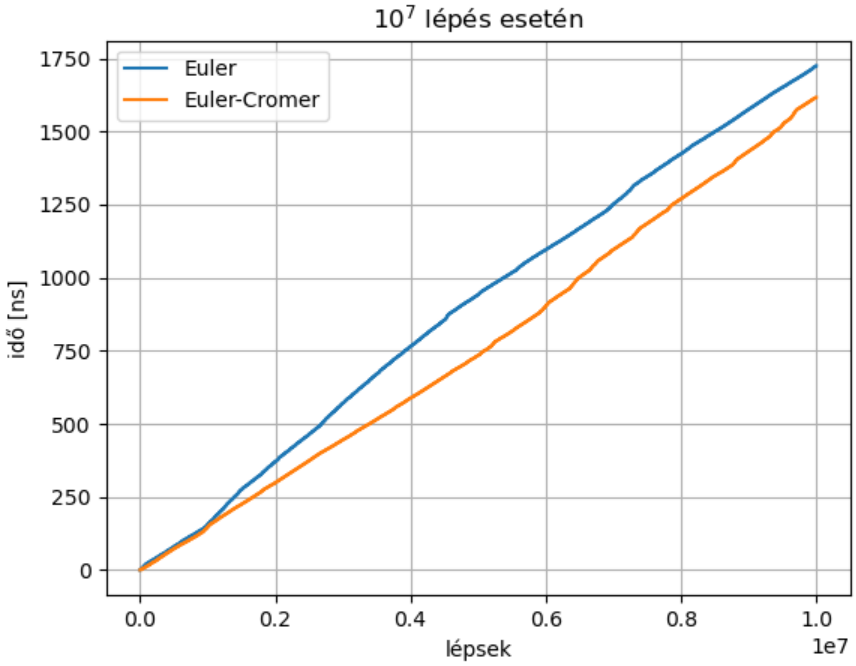
5. Futási idő elemzése

5.1. Kód módosítása

Annak érdekében, hogy ne kelljen újra és újra lefuttatnom az algoritmusokat különböző lépésszámmal a következőt találtam ki. Definiáltam két új függvényt *Eulerperformance* és *EulerCromerperformance* néven, melyek pontosan ugyan úgy működnek mint a korábbi megfelelőik egy dolgot leszámítva, a lépések után nem a hely és sebesség értékeket jegyzik fel hanem az algoritmus futásától kezdve addig a pillanatig eltelt időt miliszekundummban. A teszteléshez először 1 millió lépést választottam.



A grafikon szerint a sima Euler módszer a gyorsabb, de viszont pontatlanabb. Kíváncsiságból elvégeztem még egy tesztet és megemeltem a lépések számát 10 millióra.



Ebben az esetben pont az ellenkezőket kaptam. A második méréshez tartozó fileokat nem csatoltam a Koplex felületen, mert egyenként 140 Mb körül voltak. Összeségében azt állapíthatjuk meg, hogy a futási idő nagyjából lineárisan növekszik a lépések számával, csak a meredekségben vannak különbség a módszerek között.