Universidad del País Vasco
Euskal Herriko Unibertsitatea
ZIENTZIA ETA TEKNOLOGIA FAKULTATEA
FACULTAD DE CIENCIA Y TECNOLOGÍA

Gradu Amaierako Lana
Fisika eta Ingeniaritza Elektronikoko Gradu Bikoitza (Ingeniaritza Elektronikoa)

Metodo sasiespektraletarako sarrera

Egilea:
Luis Arriola Carril
Zuzendaria:
Francisco de la Hoz Méndez Dk.

Leioa, 2022eko ekainaren 22a

Laburpena

Metodo espektralak konputazio zientifikoan erabiltzen diren zenbakizko metodo mota dira. Ekuazio diferentzialak ebazteko zenbakizko metodo anitz existitzen dira, eta kasu bakoitzerako hobe moldatzen den metodoa erabiliz bai konputazio denbora txikiagotu bai zehaztasun hobea lortu daitezke. Ingeniaritza elektronikoan eskaintzen den programazioaren ezagutza aberatsa da, eta zenbakizko metodoen arloan hazteko, metodo espektralen teoria eta haien nondik norakoak jakitea oso onuragarria izan daiteke.

Metodo espektralak deribatu partzialezko ekuazioak ebazteko matematika aplikatuan erabiltzen diren zenbakizko metodo indartsuak dira. Teoria konplexuago baten truke, metodo hauek zehaztasun altua eta memoria erabilera murriztua eskaintzen dute (onuragarriak diren baldintzak betetzen badira). Hortaz, aldagai kopuru edo dimentsio handiko sistemen ebazpen eraginkorra proposatzen dute, beste metodoek zehaztasun bera lortzeko behar duten baino denbora gutxiagoa erabiliz.

Lan hau metodo sasiespektraletan zentratuko da, ekuazio ez-linealetan aplikatuta. Datorrena ulertzeko oinarri teorikoa azalduko da lehenik, kasu praktikoa aztertzeko erabiliko diren metodologia eta algoritmoen funtsa azalduz. Landuko dugun kasu praktikoa Korteweg de Vriesen ekuazio ez-lineala izango da, gure lanean aipagarritasun berezia duen ekuazioa dela eta. Metodo ezberdinen bidez kasu orokor baten ebazpena burutuko da, haien zehaztasunak eta ebazpen-abiadurak alderatuz. Esperimentu honen bidez, metodo sasiespektralen onura teorikoak inguru praktiko batean frogan jarriko dira.

Gaien Aurkibidea

1. Sarrera

Lan honen helburua metodo espektralen oinarriak garatzea da, bereziki metodo sasiespektralenak. Metodo hauek era eraginkorrean ekuazio diferentzial arruntak edo deribatu partzialezkoak zehatz eta azkar ebazteko erabili daitezke. *Python* lengoaiaren bidez, lan honen simulazioak eta azalpenak bat egingo ditugu, metodoen gaitasunak era argian erakutsiz.

Metodo espektralak deribatu partzialezko ekuazioak ebazteko erabiltzen diren 3 metodo familietako bat dira. Hiru multzo hauek garatu ziren bata bestearen jarraian hurrengo hamarkadetan [1, Abst.]:

- 1950s: Diferentzia finituko metodoak. Metodo hauek gainezartzen diren ordena baxuko polinomio lokalak erabiliko dituzte emaitza hurbiltzeko.
- 1960s: Elementu finituko metodoak. Metodo hauek funtzio lokal leunak (jarraituak, deribagarriak eta errorik gabe) erabiliko dituzte emaitza hurbiltzeko.
- 1970s: Metodo espektralak. Metodo hauek funtzio global leunak erabiliko dituzte emaitza hurbiltzeko.

Metodo espektralak aztertuko ditugu. Garatu zen azken metodo familia da eta baldintza onuragarrietan beste metodoak baino askoz indartsuagoa da. Ebatzi nahi den emaitza leuna bada, metodo espektralek zehaztasun ikaragarria daukate, hamar bat digituko doitasuna lortuz non diferentzia edo elementu finituko metodoek bizpahiru digitu lortzen dute. Hau dena konputazio denbora parekoa erabiliz. Metodo espektralak fluxuak aztertzeko erabiliak dira, non turbulentziek edo inperfekzioek kate-efektuak piztu ditzakete. Are gehiago, aplikazio asko dituzte non zehaztasuna eskertzen den[1][1, Abst.].

Doitasun hau ez da beti beharrezkoa baina metodo espektralek beste onurak dakartzate. Beste metodo familiek baino aldagai gutxiago behar dute problemak-ebazteko. Abantaila hau dimentsio askotako problemetan nabarmentzen da, non ordenagailu-memoria aurreztu dezakegu. Gainera, algoritmo askok N aldagai kopuruaren menpe den denbora behar dute, eta N kopurua txikiagotzeak oso garrantzitsua izan daiteke, konputazio-denbora totala motzagoa izateko [1, Abst.][2, Chap. 1].

Hasteko, metodo espektralen ezaugarriak azalduko ditugu. Metodo hauen barruan, metodo sasiespektralak kokatuko ditugu eta beraien eraginkortasuna nabarmenduko dugu. Azalpenekin hasi baino lehen, lanaren beste edukiak adieraziko ditugu: bigarren atalean metodo espektralak azaldu eta gero, deribatu partzialezko ekuazioak ebazteko beharko ditugun beste tresnak aztertuko ditugu, Runge-Kutta familiako metodoak (3. atalean) eta aliasing fenomenoen kudeaketa (4. atalean). Azkenik, 5. atalean, *Python*en sortutako simulazioen bidez kasu praktiko bat aztertuko dugu, lehen azaldutako guztiaren eraginkortasuna alderatuz eta hobe moldatzen zaion metodoa eraikiz.

[1] Fluxu mekanika, mekanika kuantikoa, bibrazioak, uhin linealak eta ez-linealak eta analisi konplexua, besteak beste.

2. Metodo espektralak

Metodo espektralen ideia nagusia erantzuna N luzerako $\{\phi_k(x)\}$ oinarri baten bidez hurbiltzea da. Horrela, $\mathcal{L}u(x,t) = f(x,t)$ ekuazio diferentziala betetzen duen funtzioa ebazteko, $\{a_k(t)\}$ koefizienteak lortzeko metodoa aurkitu behar da.

$$u(x,t) \approx u_N(x,t) = \sum_{k=0}^{N-1} a_k(t)\phi_k(x) \tag{1}$$

Lanaren hurrengo azpiataletan, metodoak dituen bi ezezagun nagusiei emango diegu erantzuna. Lehenik, $\{\phi_k(x)\}$ oinarri gisa zer funtzio familia erabiliko dugun erabaki behar da. Ondoren, $\{a_k\}$ koefizienteak lortzeko zer estrategia erabiliko den erabakiko da [3][2, Chap. 1].

2.1. Oinarri trinkoa. Funtzioen serie garapena

Metodo espektralarekin hasteko, lehenik eta behin hurrengo ataletan erabiliko diren oinarriekin landu dugu. Oinarri bezala erabili ditugun funtzio familiak aztertu ditugu eta haien bidezko serie garapenak egin ditugu. Metodo espektraletan era eraginkorrean lantzeko, oinarri egoki batek hurrengo ezaugarriak izan behar ditu [3]:

1. Konbergentzia: Erantzunaren hurbilketak, $u(x,t) \approx u_N(x,t)$, azkar bat egin behar du $n \to \infty$ joan ahala (ebatzi nahi dugun funtzioa leuna bada behintzat).

2. Diferentziazioa: Erantzunaren $\{a_k(t)\}$ koefizienteak izanda, bere deribatuaren $\{a_k'(t)\}$ koefizienteak azkar eta erraz lortu izan behar dira, non:

$$\frac{\partial u_N}{\partial x} = \sum_{k=0}^{N-1} a_k(t)\frac{d}{dx}\phi_k(x) \to \sum_{k=0}^{N-1} a_k'(t)\phi_k(x)$$

3. Transformazioa: $\{u_n(t)\}$ eta $\{a_k(t)\}$ koefiziente bektoreen arteko konbertsioa azkarra izan behar da, hau da, erantzunaren koefizienteen eta oinarriaren hedapen koefizienteen arteko aldaketa.

$$\{u_n(t)\} \longleftrightarrow \{a_k(t)\}$$

Jakina, ebatzi nahi den sistema edo ekuazioaren arabera, ez dago beti hoberen jokatuko duen oinarririk. Eginkizunari hobe moldatzen den oinarria erabiltzeak dakartzan onuran dago abantailarik handiena. Problema periodikoetan, funtzio trigonometrikoak erabili ohi dira. Problema ez periodikoetan, ordea, Jacobi motako polinomio ortogonalak (Chebyshev, Legendre, Hermite...) erabiltzen dira [4]. Gure lanean, Chebysheven polinomioak aztertuko ditugu.

2.1.1. Oinarri esponentziala

Erabiliko den oinarria $\{\exp(ikx)\}$ izango da. Normalean oinarri infinitua dugu, non $k \in (-\infty, +\infty) \cap \mathbb{Z}$, baina metodo konputazionalekin ari garenez, N puntuz osatutako eta 0-n zentratutako oinarriaren lagina erabiliko dugu. Funtzio familia hau oinarri ortogonala da $x \in [0, 2\pi)$ inguruan, baina aldagai aldaketa baten bidez L luzerako segmentuan ortogonala den oinarria izan dezakegu, hau da, $x \to 2\pi x/L$ hartuz [3]. Hortaz, funtzio baten serie garapena (2) bidez idatziko dugu [5, Chap. 7].

$$f(x) = \sum_{k=-\infty}^{\infty} \hat{f}(k) e^{2\pi i k x/L} \approx \sum_{k=-N/2}^{N/2-1} \hat{f}(k) e^{2\pi i k x/L} \tag{2}$$

Egindako hurbilketa ona izateko, aztertutako funtzioa azkar zero bihurtu behar da. Ondorioz, k maiztasun altuko koefizienteek azkar zerora egingo dute. Horrela oinarri trinkoa erabili dezakegu, $[-N/2, N/2]$ kanpoko maiztasunen ekarpenak kenduz, zehaztasuna galdu gabe.

x aldagai librea ere diskretizatuko dugu, N puntu distantziakidetan $x_j = Lj/N, \quad 0 \leq j \leq N-1$ adierazpena jarraituko dutenak. Lortuko dugun funtzioa N luzerako bektorea izango da (3). Gainera eta gure oinarria diskretua denez, $\hat{f}(k)$ funtzio espektrala $a(k)$ koefizienteen bidez adierazi dezakegu[2], eta koefiziente hauek lortzeko adierazpena, hots (4), jadanik ezaguna dugu, Fourierren transformatu diskretua DFT() delako [5, Chap. 7]. Izan ere, oinarri esponentziala erabiltzeak dakartzan onurek Fourierren transformatuarekin dute erlazioa.

$$f(x_j) = \sum_{k=-N/2}^{N/2-1} \hat{f}(k) e^{2\pi i k x_j/L} = \sum_{k=-N/2}^{N/2-1} a_k e^{2\pi i j k/N} \tag{3}$$

$$a_k = \frac{1}{N} \sum_{j=-N/2}^{N/2-1} f(x_j) e^{-2\pi i j k/N} \tag{4}$$

Lanean zehar Fourierren transformatu diskretua kalkulatzeko Fourierren transformatu azkarra ('Fast Fourier Transform' `fft()`) metodo ezaguna erabiliko dugu [5, Sec. 7.3.2]. Algoritmo generala eta modernoaren aintzatespena 1965 urtean James Cooley eta John Tukey ikerlariek sortutakoari eman ohi da [6]. Metodo hau arinagoa da N zenbaki handiekin lan egitean eta dimentsio handiko problemak ebaztean mesedegarria izango da. *Python*en lan egiterakoan *NumPy* liburutegiko `fft()` metodoa erabiliko dugu, *numpy.fft.fft()*

[2]Fourierren maiztasun espazioa diskretua izanda, x espazio zuzeneko funtzioak periodikoak izango dira. Edozein funtzio (finitu) aztertu dezakegu baina interes gunetik kanpo funtzioak periodikotasuna azalduko du. Gibbsen fenomenoa azaldu daitekela kontuan izan behar da.

Eragiketa nagusiak garapen esponentzialaren bidez

Garapen esponentzialaren bidez aztertu nahi den funtzioaren gaineko eragiketak oso erraz egin daitezke, funtzio esponentzialen ezaugarriak direla medio. Ondoren, ekuazio diferentzialak ebazteko lagungarri izango diren eragiketak nola burutuko diren azalduko dugu. Funtsean, ekuazio diferentzialetan ebatzi nahi den $y(x)$ funtzioa $\{a_k\}$ koefiziente bektore baten bidez adieraziko dugu, eta eragiketa hauek koefizienteak nola aldatuko dituzte jakin behar dugu.

1. **Konstante baten bidezko biderketa**:

$$A \cdot f(x) = A \cdot \left(\sum_{k=-\infty}^{\infty} a_k e^{ikx} \right) = \sum_{k=-\infty}^{\infty} A \cdot a_k e^{ikx} \tag{5}$$

2. **Berreketa**:

$$\begin{aligned}(f(x))^2 &= \left(\sum_{k=-\infty}^{\infty} a_k e^{ikx} \right)^2 = \left(\sum_{k=-\infty}^{\infty} a_k e^{ikx} \right) \left(\sum_{l=-\infty}^{\infty} a_l e^{ilx} \right) \\ &= \sum_{k=-\infty}^{\infty} \sum_{l=-\infty}^{\infty} a_k a_l e^{i(k+l)x} = \sum_{k=-\infty}^{\infty} \left(\sum_{l=-\infty}^{\infty} a_{k-l} a_l \right) e^{ikx} \\ &= \sum_{k=-\infty}^{\infty} (a_k * a_k) e^{ikx} \end{aligned} \tag{6}$$

   Berreketaren koefiziente berriak lortzeko konboluzioa kalkulatu behar da, hau da, $a'_k = a_k * a_k$. Edozein ordenako berreketaren formula azaltzeko, notazio berria hartuko dugu, non $(f(x))^n$ funtzioaren koefizienteak $a_k^{[n]}$ diren. Koefiziente hauek lortzeko, prozedura errekurtsiboa behar dugu, hurrengo erlazioa erabiliz: $a_k^{[n]} = a_k^{[n-1]} * a_k$.

3. **Deribazioa**:

$$f^{(n)}(x) = \frac{d^n}{dx^n} \left(\sum_{k=-\infty}^{\infty} a_k e^{ikx} \right) = \sum_{k=-\infty}^{\infty} a_k \frac{d^n}{dx^n} e^{ikx} = \sum_{k=-\infty}^{\infty} (ik)^n a_k e^{ikx} \tag{7}$$

4. **Integrazioa**:

$$\begin{aligned}\int_0^x f(y)dy &= \int_0^x \left(\sum_{k=-\infty}^{\infty} a_k e^{iky} \right) dy = \sum_{k=-\infty}^{\infty} a_k \int_0^x e^{iky} dy \\ &= a_0 x + \sum_{\substack{k=-\infty \\ k \neq 0}}^{\infty} a_k \frac{e^{ikx} - 1}{ik} \end{aligned} \tag{8}$$

2.1.2. Chebyshev polinomioz osatutako oinarria

Batez ere funtzioak mugalde baldintza ez periodikoak dituenean eta Fourierren bidezko serie garapenean Gibbsen fenomenoa azaltzen denean, oso interesgarria da Chebysheven polinomioekin lan egitea [2, Sec. 2.11].

$$f(x) = \sum_{k=0}^{\infty} a_k T_k(x) = a_0 + \sum_{k=1}^{\infty} a_k T_k(x) \qquad \text{non} \quad T_k(x) = \cos(k \arccos(x)) \tag{9}$$

Chebysheven polinomioak $x \in [-1, 1]$ eremuan ditugu oinarri ortogonala, $1/\sqrt{1-x^2}$ pisuarekiko. Berriz ere aldagai aldaketa egin zitekeen oinarria $[-1, 1]$ kanpo nahi den eremuan erabiltzeko. Halere, polinomioen definizioari (9) begiratzeak [5, Chap. 2], $x = \cos(t)$ aldagai aldaketa egitea iradokitzen du, $f(x)$-ren ordez $f(\cos(t))$-rekin lan eginez. Honela, definizio trigonometrikotik polinomio idazkera lortu daiteke, $T_k(t)$ funtzioak k. ordenako $\cos(t)$-ren polinomioan deskonposatu daitekela erraz frogatuz.

Espazio zuzeneko diskretizazioa ez da lehen bezala egingo, t izango baita puntu distantziakidetan zatituko dugun aldagaia. Hortaz, N puntuko espazioa lortu nahi badugu, $x_i = \cos(\pi i/N)$; $0 \leq i \leq N-1$ adierazpenaren bitartez kalkulatuko dira puntuak. Izan ere, puntu hauek N. ordenako Chebysheven polinomioaren $T_{k=N}(x)$ erroak dira. Kosinuaren izaera kontuan hartuz, puntuak $x = 1, -1$ puntuen ingurunean metatuko dira. Hau mutur egonkorragoak izateko lagungarria da, Rungeren fenomenoak sortutako ezegonkortasunak saihestuz [2, Sec. 2.11]. N puntuko espazio hau erabiliz, soilik N Chebysheven polinomio independenteak lortu ditzakegu, hortaz serieen garapeneko batukaria nahitaez finitua izango da. Lehen bezala, guztiz diskretizatutako serie garapena eta koefizienteak lortzeko adierazpenak ere, (10) eta (11). Oraingoz, koefizienteak ez ditugu *fft()*-ren bidez kalkulatuko.

$$f(x_i) = \sum_{k=0}^{N-1} a_k \cos\frac{\pi k i}{N} \tag{10}$$

$$a_k = \frac{2}{N} \sum_{k=0}^{N-1} f\left(\cos\frac{\pi k}{N}\right) \cos\frac{\pi k i}{N} \tag{11}$$

2.1.3. $\mathbf{f(x) = \exp(\sin(x))}$ funtzioaren garapena

Deskribatutako metodoak praktikan jartzeko, $f(x) = \exp(\sin(x))$ funtzioaren serie garapena egiteko erabiliko ditugu. Funtzio hau jarraitua eta 2πko periodoa du. Lehenik, oinarri esponentzialaren bidez garatuko dugu, $x \in [0, 2\pi)$ eremuan. Eremu hau diskretizatzerakoan, N puntutan banatu dugu. Hortaz, N luzerako oinarri ortogonala hartu dezakegu, baina konputazio denbora arintzeko, ez dugu zertan oinarri guztia garapenenean erabili. Izan ere, k txikiko osagaiak izango dira ekarpen handienak izango dituztenak, hortaz $-N/2$tik $N/2$ra doazen oinarri guztia erabili behar, $-m$tik $+m$ra doazen $N' = 2m + 1$ osagaiak erabiltzeko

aukera dugu, doitasun handia lortuz. Atal honetan zenbat osagai erabili behar ditugu ikusiko dugu, eta batugaiaren osagai bakoitzak ekartzen duen hobekuntza ere bai.

f(x)-ren garapena $[0, 2\pi)$ eremuan

Lehenengo azterketan honetan, $N = 1024$ puntu erabiliko dira, eta bai oinarri esponentziala bai Chebysheven polinomioena erabiliko dira, bien emaitzak alderatzeko. m. indizeko batugairaino egingo da kalkulua, baina esponentzialaren kasuan koefizienteak negatiboak izan daitezke, beraz $k \in [-m, m]$ izango dugu ($N' = 2m + 1$ guztira), Chebysheven polinomioen kasuan $k \in [0, m]$ ($N' = m+1$) izango dugun bitartean. Bien arteko lehia ez da ariketa honen helburua, haien bilakaera nola den ikustea baizik. Gainera, Krasnyren iragazkia aplikatu da bi metodoen a_k koefizientetan, garapenaren doitasuna hobetzeko[3].

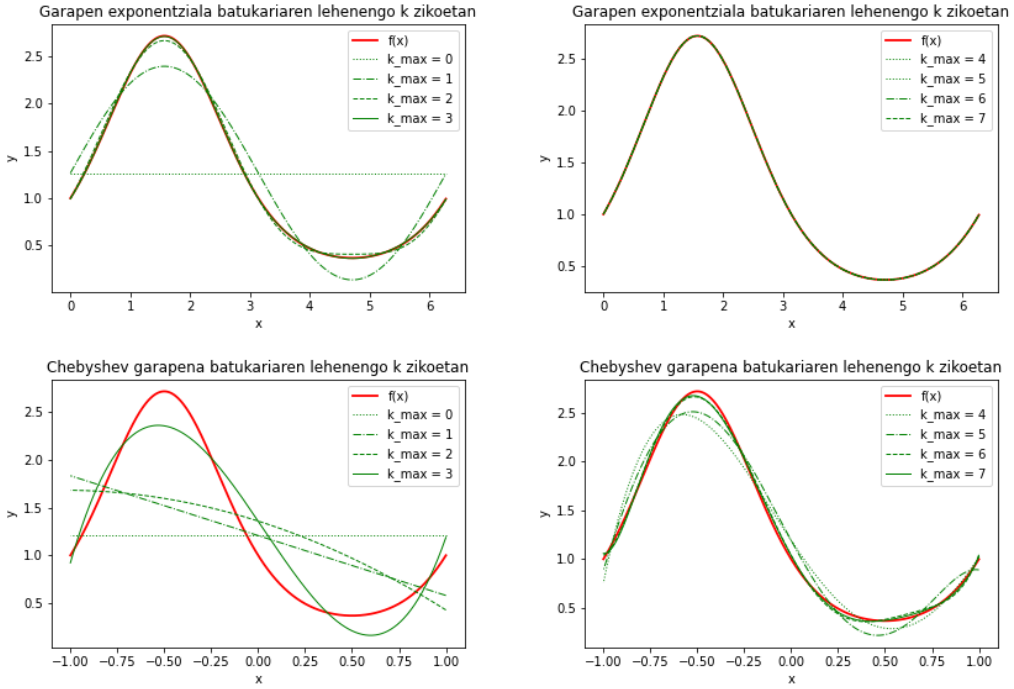
Aukeratutako $m = 8$ izango dugu, eta funtzioen grafikoak 1. irudian ikusi ditzakegu. Kasu honetan, $f(x)$ funtzioak sinu formaren antzekoa du eta oso aproposa da funtzio esponentzialen bidez garatzeko: osagai gutxiren bidez erantzun oso ona (zehatza) izan dugu. Chebysheven oinarriaren kasuan, erantzun ohikoagoa ikusi daiteke, non erantzuna ez den hain azkar lortzen baina gutxi gorabehera 10 bat osagairekin nahikoa izan daitekeela esan daiteke. Azkenik, bi metodoek erreferentziazko funtzioa irudikatzea lortu dutela aipatu behar da, oinarriaren zati txikia soilik erabiliz. Kasu zehatz honetan, $N' \approx 10, \dots 20$ luzerako oinarri mugatua erabiltzearen konputazio abantailak ikusi daitezke ($N = 1024$ osagai erabiltzearen aurrean), garapen zehatza lortu baita.

f(x)-ren garapena $[-1, 1)$ eremuan

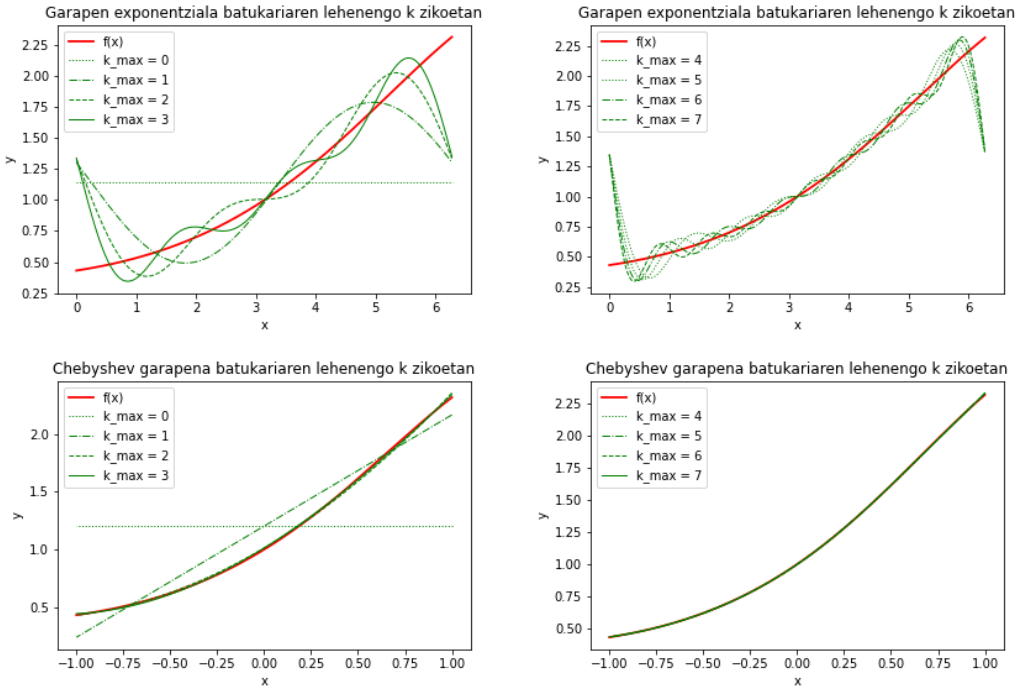
Oraingo kasuan, berriz, $N = 1024$ puntu erabiliko ditugu. Bi oinarriak erabiliko dira, baina mugalde baldintza ez-periodikoak kasu honen berezitasuna izango dira. Ariketa hau aurrekoaren era berean garatu dugu eta 2. irudian emaitzak ikusi daitezke. Aurreko azpiataletan azaldutako Gibbsen fenomenoaren efektua ikusi dezakegu garapen esponentzialean (funtzioa $x = 1$ puntuan jarraitua ez delako). Osagai kopurua handitu ahala, muturretako singulartasunak handituko dira, baina zehaztasun ona lortzea espero dugu, Gibbsen fenomenoaz aparte. Muturretatik urrun, erantzun ona espero dugu.

Chebysheven polinomioen kasuan, erantzun ona izan dugu eta funtzioak oinarri polinomiko baten bidez garapen hobea izan duela ikusi dezakegu. Garrantzitsua da Rungeren fenomenorik izan ez dugula aipatzea. Rungeren fenomenoa garapen polinomikoetan agertzen den muturretako ezegonkortasuna da, polinomioek $x = -1,\ 1$ puntuetatik gertu duten izaera oszilakorra dela eta.

[3]Krasnyren iragazkiak $(0, 2^{-52})$ tarteko zenbakien borobiltze errorea kentzen du. Horretarako, $eps = 2^{-52}$ baino txikiago diren zenbakiak zero bihurtzen ditu.



1. irudia: $f(x) = \exp(\sin(x))$ funtzioaren garapena $[0, 2\pi)$ eremuan, batukariaren k indizea k_{max}. balioraino kontsideratuz. Goian, oinarri esponentzialarekin egindako garapena, eta Chebysheven polinomioekin egindakoa, behean. Erreferentziazko funtzioa, gorriz.



2. irudia: $f(x) = \exp(\sin(x))$ funtzioaren garapena $[-1, 1]$ eremuan, batukariaren k indizea k_{max}. balioraino kontsideratuz. Goian, oinarri esponentzialarekin egindako garapena, eta Chebysheven polinomioekin egindakoa, behean. Erreferentziazko funtzioa, gorriz.

f(x)-ren garapenaren deribatuak

Hurrengo froga $\{a_k\}$ koefiziente-bektorea maneiatzea izango da. Oinarri esponentziala erabiliko dugu, 2.1. atalean deskribatutako eragiketen eraginkortasuna neurtuz. Bereziki, deribazio teknika aztertuko dugu. $f(x)$ funtzioaren garapena egingo dugu, eta Fourierren transformazio azkarra metodoaren bidez lortutako koefizienteak (7) ekuazioaren bidez manipulatuko ditugu. Serie garapena ondo joango dela espero dugu, lehen ikusi den bezala, hortaz deribatu guztien garapena egonkorra izango dela suposatuz, prozesuak lortuko duen doikuntza aztertuko dugu.

Funtzioaren lehenengo bost deribatuak kalkulatuko ditugu. Berriz ere $N = 1024$ puntuetan zatituko dugu x ardatza, 0-tik 2π-raino eremuan. Deribatu guztien serie garapenaren batukariaren lehenengo 20. indizeak kontuan izango ditugu ($k \in [-20, +20]$ indizeak, alegia), eta funtzio errealarekin izandako erroreen batura kalkulatuko ditugu, hurrengo adierazpena erabiliz:

$$\epsilon = \sqrt{\frac{1}{N}\sum_{i=0}^{N}\left|y_{exp}(x_i) - y_{teo}(x_i)\right|^2} \qquad (12)$$

1. taula: Serie garapenak izandako errore metatua

n: Ordena	ϵ: Errore metatua
0	$2.6 \cdot 10^{-16}$
1	$5.8 \cdot 10^{-16}$
2	$7.9 \cdot 10^{-15}$
3	$1.2 \cdot 10^{-13}$
4	$1.7 \cdot 10^{-12}$
5	$2.6 \cdot 10^{-11}$

Azaldutakoa egin ondoren, emaitza guztiak funtzio errealaren itxura lortu dute eta kalkulatutako errore metatuak 1. taulan bildu ditugu. Serie garapenak zehaztasun handia izan du kasu guztietan. Gutxi gorabehera, errorea ordena bat baino gehiago (≈15 aldiz) handitzen da, eta handipen hau nahiko konsistentea dirudi.

Ondorio gisa, Fourierren espazioan lan egiterakoan deribatu espazialak azkar eta zehaztasun handiarekin kalkulatu ditzakegu, $\{a_k\}$ koefizienteak $(ik)^n$kin biderkatuz.

2.2. Garapen koefizienteak. Metodo sasiespektrala

$\{a_n\}$ hedapen koefizienteak kalkulatzeko hiru teknika nagusi daude. Hirurek, hondar funtzioa minimizatzea dute helburu. Izan bedi $\mathcal{L}u(x,t) = f(x,t)$ ebatzi nahi dugun ekuazio diferentziala, non $\mathcal{L}$ bere eragilea den. Hondar funtzioak zenbakizko metodoak egindako hurbilketaren zehaztasuna kalkulatzen duen funtzioa da, eta hurrengo adierazpenaren bidez definitua dago [3]:

$$\mathcal{R}[u_0, u_1, ...u_{N-1}](x,t) = \big[\mathcal{L}u_N - f\big](x,t) \qquad (13)$$

Hondar funtzioa $\mathcal{R}[u_n](x,t)$ zero izango da erantzun zehatzaren aurrean, hortaz $\mathcal{R}$ funtzioa erantzun numerikoaren kalitatea neurtzeko erabilia da. Helburua, hondar funtzioa minimizatzea da, $[0, L]$ espazioaren zehar. Horretarako, $\mathcal{R}$ hondar funtzioaren ortogonalta-

suna ezarriko da, $\{\xi_k\}$ funtzio laguntzaile familiarekiko (14). Metodo espektral bakoitzak hau lortzeko funtzio laguntzaile ezberdina proposatzen du [3].

$$0 = (\xi_k, \mathcal{R}) = \int_0^L \xi_k(x) \cdot \mathcal{R}(x,t)dx \tag{14}$$

- Tau metodoa

  Tau metodoan, $\{a_k\}$ koefizienteak mugalde baldintzak bete ditzaten baldintzatuko dira. Ondoren, hondar funtzioa $\mathcal{R}[u_n]$ oinarriaren ahal den ϕ_k funtzio kopuru handienari ortogonal izan dezan, geratzen diren baldintzak jarriko dira, $\{a_k\}$ koefizienteak zehaztu arte. Ortogonaltasuna bermatzeko, $(\phi_k, \mathcal{R}) = 0$ ekuazioa bete behar da $x \in [0, L]$ espazio osoan. Hortaz, metodo honen funtzio laguntzaileak $\{\phi_k\}$ direla esaten da.

- Galerkin metodoa

  Galerkin metodoan, $\{\phi_k\}$ oinarria rekonbinatuko da, $\{\phi_k\} \to \{\tilde{\phi}_k\}$. Oinarri berria mugalde baldintzak betetzeko sortua da. Ondoren, hondar funtzioa $\mathcal{R}[u_n]$ oinarri berriaren ahal den $\tilde{\phi}_k$ funtzio kopuru handienari ortogonal izan dezan, $\{a_k\}$ koefizienteak zehaztuko dira. Metodo honen funtzio laguntzaileak $\{\tilde{\phi}_k\}$ izango dira.

- Kokapen metodoa

  Kokapen metodoan, Tau metodoan bezala, lehen $\{a_k\}$ koefizienteak mugalde baldintzak bete dezaten baldintzatuko dira. Ondoren, hondar funtzioa $\mathcal{R}[u_n]$ espazioaren $\{x_n\}$ puntuetan[4] zero izan dezan jarriko dira baldintzak. Sistema hau ebatziz, $\{a_k\}$ koefizienteak zehaztuko dira. Ez da oinarriarekiko ortogonaltasuna ezarri behar, soilik zero izan behar da ahal den $\{x_n\}$ puntuetan. Hortaz, metodo honen funtzio laguntzaileak $\{\delta(x - x_i)\}$ delta funtzioak direla esaten da. Metodo hau metodo sasiespektrala deitzen da [2, Chap. 1].

Lan honetan, metodo sasiespektrala erabiliko da. Ondorioz, $\mathcal{L}$ eta $\mathcal{R}$ eragileak diskretizatu ditzakegu, x_i puntuetan soilik ebaluatu behar ditugulako. Ondorioz, $\mathcal{L}u(x,t) = f(x,t)$ ekuazioa eta bere hondarra hurrengo eran geratuko dira:

$$R = Lu - f \implies \begin{pmatrix} a_{11} & a_{12} & \dots & a_{1N} \\ a_{21} & a_{22} & \dots & a_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N1} & a_{N2} & \dots & a_{NN} \end{pmatrix} \begin{pmatrix} u_1 \\ u_2 \\ \vdots \\ u_N \end{pmatrix} - \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_N \end{pmatrix} = 0 \tag{15}$$

Metodo sasiespektralen eraginkortasunak, ordea, Fourierren espazioan lan egitean distiratzen du, R eta L eragileen maiztasun espazioko adierazpena lortzea posible bada. Fourierren espazioan, $\{a_k\}$ koefizienteen ekuazio diferentzial arrunten sistema dugu. Gainera, `fft()` eta 2.1. sekzioan azaldutako eragiketak erabiliz, ekuazio diferentzialak asko sinplifikatuko dira.

[4] Ahal eta puntu gehienetan, mugalde baldintzak jada ezarri ditugulako.

3. PDE-en ebazpena. Runge-Kutta metodoa

Metodo espektralak deribatu partzialezko ekuazioak (Partial Differential Equation, PDE, ingelesez) ebazteko nola erabili daitezkeen aztertuko dugu orain. Aztertuko ditugun ekuazioak 1-2 dimentsio espazialak eta denbora dimentsio bat izango dituzte. Normalean, denbora menpeko PDE-ak numerikoki ebazteko hurrengo eskema jarraitzen da: zati espaziala aztertzeko diferentziazio espektrala erabiliko da, baina denbora garapena diferentzia finituko metodoen bidez egingo da [1, Chap. 1]. Guk Runge-Kutta metodoa erabiliko dugu.

Diferentzia finituko metodoen erabilerak zehaztasun espektrala galtzen du, baina denbora-pauso txikiak egitea ez da oso garestia: konputazio denbora linealki aldatzen dute baina erabili beharreko memoria ez da handitzen [1, Chap. 10]. Zehaztasun onargarria izateko, ordena handiko metodoak erabiltzen dira.

Runge-Kutta 4 metodoa (RK4)

Runge-Kutta metodoak ekuazio diferentzial arrunten hasierako baloreen problema numerikoki integratzeko erabiltzen diren diferentzia finituko metodoak dira. Hauen helburua Eulerren ordena altuagoen zehaztasun berdina lortzea da, deribatu analitikorik egin gabe. Izan bedi hurrengo hasierako baloreen problema,

$$y'(t) = F(t,\ y(t))\ ; \quad y(t_a) = y_a\ ; \quad t \in [t_a,\ t_b] \tag{16}$$

Denbora-pausoak h luzerakoak badira, y funtzioaren garapen diskretua (17) ekuazioaren bidez kalkulatuko da:

$$\begin{aligned} k_0 &= hF(t_i,\ y_i) & k_1 &= hF(t_i + \frac{h}{2},\ y_i + \frac{k_0}{2}) \\ k_2 &= hF(t_i + \frac{h}{2},\ y_i + \frac{k_1}{2}) & k_3 &= hF(t_i + h,\ y_i + k_2) \end{aligned}$$

$$y_{i+1} = y_i + (k_0 + 2k_1 + 2k_2 + k_3)/6 \tag{17}$$

Metodo hau laugarren ordenakoa da, hots, puntu bakoitzaren trunkatze errorea $O(h^5)$ ordena du, eta errore metatu totala, $O(h^4)$ ordena. Eulerren metodoaren aurrean daukan abantaila, bere egonkortasuna eta konbergentzia dira [5, Sec. 12.7].

3.1. Dimentsio bakarreko lehen ordenako ekuazioa: $\mathbf{u_t + c(x)u_x = 0}$

PDE-ak ebazteko erabiliko dugun metodoa ondo ulertzeko, hainbat adibidetan frogan jarriko dugu. Simulatuko dugun lehenengo funtzioa dimentsio bakarreko funtzioa da (diskretizazio espaziala + diskretizazio denborala), Trefethenen liburuak aztertzen duena [1]. Aztertuko dugun ekuazioa eta bere baldintzak hurrengoak izango dira:

$$\frac{\partial u}{\partial t} + c(x)\frac{\partial u}{\partial x} = 0 \ ; \quad c(x) = \frac{1}{5} + \sin^2(x-1) \ ; \quad \begin{cases} x \in [0, 2\pi) \ ; \ t \in [t_a, t_b) \\ \texttt{MB:}\ u(0,t) = u(2\pi, t) \\ \texttt{HB:}\ u(x, t_a) = u_0 \end{cases} \tag{18}$$

non MB mugalde baldintzak diren eta HB hasierako edo hastapen baldintzak diren. Runge-Kutta metodoaren aldetik, honen funtzionamendua jada azaldua dugu eta landu beharreko F funtzioa $F(t,u) = -c(x) \cdot u_x$ izango da. Funtzio honek u_x deribatua du, eta hau ebazteko metodo espektralak erabiliko ditugu.

x aldagaiaren espazioa N puntu distantziakidetan banatuko dugu, hortaz $h = 2\pi/N$ izango dugu. Bestalde, $x \in [0, 2\pi)$ lan egingo dugun eremua izango da, Fourierren oinarria erabili nahi dugulako.

Diferentziazio matrizearen erabilera

Hasteko, u_x diferentziazioa era matrizialean egingo dugu, prozesuaren azalpen grafikoa ahalbidetzen duelako. Demagun diferentzia finituko metodoak erabili nahi ditugula: x_i puntuaren deribatua kalkulatzeko, Eulerren formula erabili behar dugu (Taylorren lehen ordenako serie garapenatik lortua).

$$u_x(t, x_i) \approx \frac{u(t, x_{i+1}) - u(t, x_{i-1})}{2h}$$

Hau bigarren ordenako Eulerren formula *zentratua* izango da[5]. Ikusi daitekeenez, aldameneko $i-1$ eta $i+1$ elementuak kontuan hartzen ditu. Beraz, $\{u(t, x_i) \equiv u_i\}_i$ funtzioaren bektore errepresentazioarekin lan egin nahi badugu, diferentziazioa matrize baten bidez kalkulatu dezakegu. Bai *zentratua*, bai *aurreranzko* edo *atzeranzko* formulazioa erabilita, matrize tridiagonala izango dugu. Ordena handiagoko formula erabiliko bagenu, bestalde, matrizearen diagonal gehiago beteko genituzke. Esate baterako, laugarren ordenako[6] *zentratua* formula $\{i-2, i-1, i+1, i+2\}$ elementuak erabiliko ditu, bost diagonal zentralak erabiliz.

Metodo espektralak erabiltzean, aztertu beharreko funtzioaren deribatua kalkulatzeko, espazioaren puntu guztien informazioa erabiliko dugu. Hortaz, matrizearen i errenkadak x_i puntuaren alboko pisuen koefizienteen informazioa izango du, eta hauen balioa hurrengoa da:

$$d_{ij} = \frac{1}{h}\frac{1}{i-j} \ , \quad d_{ii} = 0$$

Ikusi daitekeenez, ez da egongo zeroz osatuta egongo den diagonalik (nagusia izan ezik), eta diferentziazio matrize dentsoa izango dugu. Gainera, matrize zirkulantea dugu, hau da, honen informazioa $(i-j)$-ren arabera dugulako: diagonal bakoitzak balio bera du diagonal

[5] Taylorren serie garapenaren lehenengo gaia hartuta bigarren ordenako zehaztasuna lortzen da [5, Chap. 3].
[6] Berebat.

guztiaren zehar, $A_{i,j} = A_{i+1,j+1}$. Adierazpen hau erabiliz sortutako diferentziazio matrizea infinitua da, x_i puntu kopuru infinitua kontuan hartzen duelako.

Metodo konputazionaletan, ordea, lagin finituarekin egingo dugu lan, eta aurreko formula egokitu behar da $N \times N$ tamainako matrizea nahi badugu. Hau eginez, (19) matrizea lortzen da. Kotangenteen bidezko adierazpena azaldu dugunaren baliokidea da, eta frogapen teorikoa Trefethenen liburuan eskuragarri dago. [1, Chap. 1].

$$D_N = \begin{pmatrix} 0 & \frac{1}{2}\cot\frac{1h}{2} & \dots & -\frac{1}{2}\cot\frac{(N-1)h}{2} \\ -\frac{1}{2}\cot\frac{1h}{2} & 0 & \dots & \frac{1}{2}\cot\frac{(N-2)h}{2} \\ \dots & \dots & \dots & \dots \\ \frac{1}{2}\cot\frac{(N-1)h}{2} & -\frac{1}{2}\cot\frac{(N-2)h}{2} & \dots & 0 \end{pmatrix} \tag{19}$$

Fast Fourier Transform-aren hobekuntza

Oinarri esponentzialarekin lan egiten ari garelako, maiztasunen espazioko aritmetikaren onurak izan nahi ditugu. Praktikan eta posible den heinean, FFT funtzioak erabiliko dira diferentziazio matrizeen ordez. Izan ere, FFTa aplikagarria da (19) matrizea zirkularra delako [1, Chap. 1]. FFTaren erabilerak hurrengo onurak dakartza:

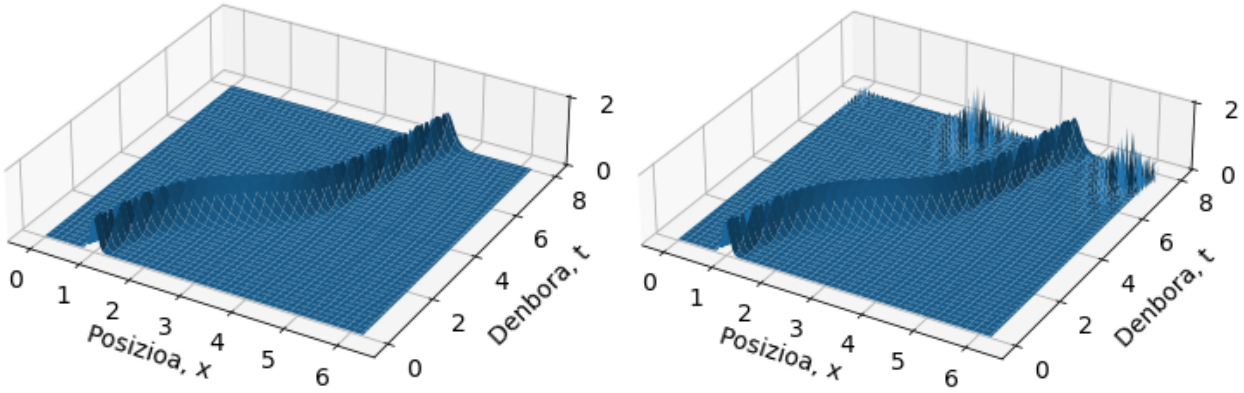
- FFTaren konputazio denbora oso ondo eskalatzen du, $O(N \ln_2 N)$ ordenakoa da; diferentziazio matrizearena $O(N^2)$ den bitartean.
- Fourierren espazioko aritmetika sinplifikatua: (5), (6), (7), (8). Alde batetik, azkarragoa da (*Adibidez, bigarren deribatua kalkulatzeko, (ik)^2 egitea tribiala da, baina $(D_N)^2$ lortzeko $O(N^3)$ ordena duen matrizeen arteko biderkadura egin behar da*). Bestetik, aritmetika sinpleagoak maiz errore txikiagoak dakartza.

Fourierren transformatu azkarra erabiliz, u_x deribatua lortzeko hurrengo eragiketa egin daiteke: $u_x = \texttt{ifft}(ik \cdot \texttt{fft}(u))$. Are gehiago, zuzenean ekuazio diferentziala Fourierren espazioan ebatzi dezakegu, (20) eskema jarraituz, non $\hat{y} = \texttt{fft}(y)$. Horrela, deribatu espazialak linealizatu ditzakegu.

$$y_t(x,t) = F(t,\ y(x,t),\ y_x(x,t)) \quad \rightarrow \quad \hat{y}_t(k,t) = \hat{F}(t,\ \hat{y}(k,t),\ ik\hat{y}(k,t)) \tag{20}$$

*Python*en bidez, zenbakizko ebazpena egin dugu, bai diferentziazio matrizearen bidez, bai `fft()` funtzioa erabiliz. Algoritmoa hasieratzeko hurrengo hasierako baldintza erabili dugu: $u(t_0, x) = \exp\{-100(x-1)^2\}$. Diskretizazioari dagokienez, x espazioa $N = 128$ puntuetan banatu dugu, eta denbora-gehikuntza $\Delta t = h/4$ izatea erabaki dugu, non $h = 2\pi/N$ espaziotartea da. Ezaugarri hauekin, bi algoritmoek emaitza bera eman dute, eta egonkorrak izan dira $t \in [0, 8]$ tartean.

Diferentzia finituen metodoetan, erantzun ezegonkorra lortu dezakegu Δt txikiegia ez bada. 3. irudian, lehen aipatutako simulazioaren emaitza ikusi dezakegu, $\Delta t = h/4$ denbora-gehikuntza egonkorrarekin kalkulatua. Segidan, hau handitu dugu irteera ezegonkorra lortu arte. Gure esperimentuak $\Delta t > h/1.16$ denborarekin ezegonkortasunak agertzen direla ikusi du.



(a) Denbora-gehikuntza: $\Delta t = h/4$.

(b) Denbora-gehikuntza: $\Delta t = h/1.16$.

3. irudia: $u_t + c(x) \cdot u_x = 0$ ekuazioaren ebazpena. x eremuaren diskretizazioa $N = 128$ puntuetan. Ezkerreko irudian, $\Delta t = h/4$ da eta simulazioa egonkorra da. Eskuinekoan, berriz, $\Delta t = h/1.16$ denborakoa da eta ezegonkortasunak agertzen dira simulazioaren bukaeran. Ezegonkortasunak azkar handitzen dira. Denbora-gehikuntza handiagoa izanez gero, simulazioa guztiz ezegonkorra da. Simulazioa egiteko, `fft()`-aren diferentziazio teknika erabili da.

Bestalde, bi diferentziazio teknika azaldu ditugu (`fft()`-a eta D_N). Lehen aipatu dugun bezala, bien arteko prozesaketa denbora oso ezberdina da, `fft()`-ak $O(N \ln N)$-rekin handitzen den denbora-tartea behar du, D_N diferentziazio matrizeak $O(N^2)$-rekin handitzen den bitartean. Ekuazioaren dimentsionaltasuna gora egin ahala, gainera, N punturen ordez N^d puntuekin lan egin beharko dugu. Efektu hau esperimentalki ikusi egin dugu gure simulazioetan. $\Delta t = h/4$ erabiliz, N puntu kopurua aldatuz joan gara, prozesaketa denbora[7] neurtu dugun bitartean. Emaitzak 2. taulan ikusi ditzakegu, eta espero genuen bezala, `fft()`-ren erabilerak prozesaketa-denbora asko murriztu du. N puntu txikiekin aldea txikia badirudi ere, dimentsio gehiagotan lan egin ezkero, D_N-ren geldotasuna jasanezina izan liteke.

2. taula: Exekuzio denbora, `fft()`-aren eta diferentziazio matrizearen erabileran, N puntuko x espazioan lan eginda.

N	16	128	256	512	1024
$t(s)$, `fft`	0.016	0.11	0.31	0.84	2.41
$t(s)$, D_N	0.016	0.36	1.17	12.9	108.7

[7]Exekuzio-denbora denbora garapen kalkuluena da soilik, problemaren inizializazioa eta grafikoen sorkuntza ez dira kontuan izan. Lanaren zehar agertuko diren exekuzio denbora guztiak baldintza berdinetan erabilitako ordenagailu berean neurtu dira, irakurleak alderatu ditzan.

3.2. Bigarren ordenako ekuazioa. 1D uhin ekuazioa: $\mathbf{u_{tt} = c^2 u_{xx}}$

Hurrengo simulazioetan, dimentsio gehiagoko kasuak aztertu ditugu. Hasteko, dimentsio bakarreko uhin ekuazioa ikusiko dugu. Uhin ekuazioa bigarren ordenako PDE-a da, eta bere definizioa, hurrengoa: $u_{tt} = c^2 u_{xx}$, non $u(t,x)$-ren soluzioa aurkitu nahi dugun. Uhin ekuazioa ingeniaritza elektronikoan sakonki aztertu dugun ekuazioa da, eta hainbat sailean ezinbestekoa zaigu. Gainera, bere uhinen itxura eta portaera ezagunak ditugu. Simulazioetan erraz ikusiko dugu ea erantzun zehatza lortu dugun.

Ekuazio diferentzial hau ebazteko, Runge-Kutta algoritmoak $u'' = F(t,u,u')$ bigarren ordenako sistema ebatzi behar du. Ordezkapen erraz baten bidez bi dimentsioko lehen ordenako sistema batean transformatu daiteke. Hitz gutxitan, $u_1 = u$ eta $u_2 = u'$ aldagaiak aztertuko ditugu, RK algoritmoa era bektorialean kontsideratuz. Matematikoki:

$$\mathbf{u}' = \mathbf{F}(t,\mathbf{u}) \quad non \quad \mathbf{u} = \begin{pmatrix} u_1 \\ u_2 \end{pmatrix} \quad , \quad \mathbf{F}(t,\mathbf{u}) = \begin{pmatrix} u_2 \\ F(t,u_1,u_2) \end{pmatrix} \tag{21}$$

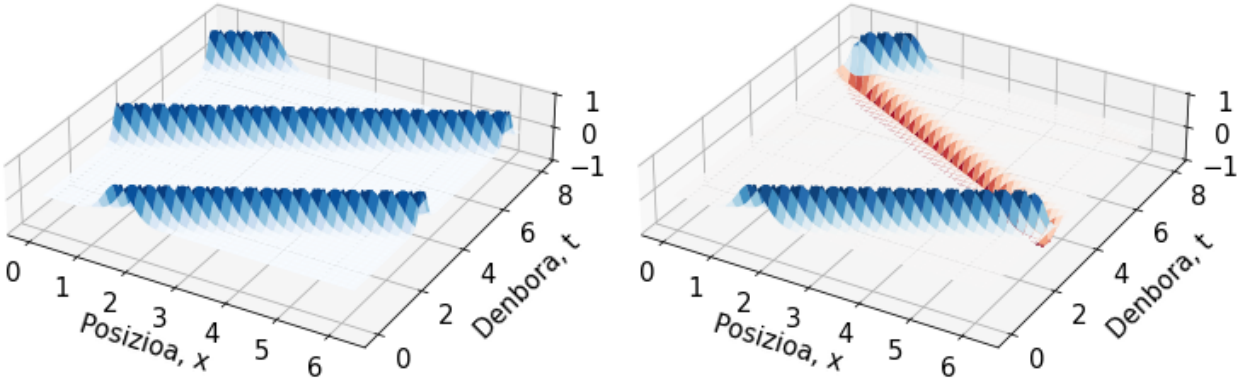
Aldagai aldaketa honen bidez, aurreko azpiatalaren antzeko prozedura jarraitu dezakegu uhinaren ekuazioa simulatzeko. Oraingoan, ordea, funtzioaren eta deribatuaren hasierako baldintzak finkatu behar ditugu. Uhin ekuazioaren kasuan, edozein $u(t,x) = u(x \pm ct)$ funtziok beteko du ekuazio diferentziala. Hortaz, hasierako baldintzak sortzea tribiala izango da, edozein formatako eta c abiadurako $\phi(x)$ funtzioa simulatu nahi badugu:

$$u(0,x) = \phi(x) \ , \qquad u_t(0,x) = \phi_t(x \pm ct)|_{t=0}$$

Mugalde baldintza ez-periodikoak

Aldatu dezakegun beste ezaugarria espazioaren mugalde baldintzak dira. Aurreko ereduan mugalde baldintza periodikoak erabili ditugu, hau da, $u(t,0) = u(t,2\pi)$; baina MB ez-periodikoak aukeratu ditzakegu. Izan ere, uhin ekuazioa aproposa da MB-taz hitz-egiteko, honen funtzioen portaera ezaguna dugulako. Dirichleten MB ezarriko ditugu. Definizioz, $u(t,x)|_\Omega = 0$, non Ω espazioaren mugalde osoa den. Gure kasuan, $u(t,0) = u(t,2\pi) = 0$ inposatuko dugu. Gure sistemaren eta esangura fisikoa, mutur finkodun 2π luzerako soka izango da.

Aztertuko dugu uhina $u(0,x) = \exp\{-a \cdot (x-b)^2\}$ izango da, non $a = 2\pi$, $b = 0.5\pi$ aukeratu ditugun. Uhinaren abiadura $c = 1.5$ da. Simulazioaren lehenengo zortzi segundoak kalkulatuko dira, eta bai mugalde baldintza periodikoen, bai Dirichleten mugalde baldintzen ondorioak ikusiko ditugu. Emaitzak 4. irudian ikusi ditzakegu, eta espero genituen portaerak lortu ditugu.



(a) Mugalde baldintza periodikoak.

(b) Dirichleten mugalde baldintzak.

4. irudia: Uhin ekuazioaren ebazpena. x eremuaren diskretizazioa $N = 128$ puntuetan. Ezkerreko irudian, mugalde baldintza periodikoak erabili dira, eta uhinak ez du paretaren kontra jotzen. Eskuinekoan, berriz, Dirichleten mugalde baldintzak inposatu dira eta uhinaren momentua espazioaren mugaldetan aldatzen da.

Chebysheven polinomioen oinarria

Mugalde baldintzataz hitz egitea matematikoki zuzena izan bada ere, espazioaren mugetatik gertu Fourierren oinarria ezegonkorra dela gogoratu behar dugu (Gibbsen fenomenoa). Oinarri hau erabiltzerakoan, espazioaren mugekin elkarrekintzarik egongo ez dela ziurtatu behar dugu. Soka batean zehar hedatzen diren uhinak aztertzeko, hobe da beste oinarria erabiltzea. Gure lanean, Chebysheven polinomioz osatutako oinarria erabili nahi dugu.

Espazioaren diskretizazioa[8] eta diferentziazio funtzioa moldatzen baditugu, simulazioa egiteko prest izango gara. Chebysheven metodoak bere diferentziazio matrize propioa du, eta hurrengo definizioen bidez eraiki daiteke[9] [1, Chap. 6]:

$$
\begin{aligned}
(D_N)_{00} &= \frac{2N^2+1}{6}, \quad (D_N)_{NN} = -\frac{2N^2+1}{6} \\
(D_N)_{ii} &= \frac{-x_i}{2(1-x_i^2)}, \quad i = 1, \ldots N-1 \\
(D_N)_{ij} &= \frac{c_i}{c_j}\frac{(-1)^{i+j}}{(x_i - x_j)}, \quad c_i = 1 + \delta_{i,0} + \delta_{i,N}, \quad i \neq j, \quad i,j = 1, \ldots N-1
\end{aligned}
\tag{22}
$$

Zoritxarrez, Chebysheven diferentziazio matrizea ez da zirkulantea, eta honen ondorioz

[8] Gogoratu Chebysheven oinarrian x_i puntuak distantziakideak ez direla

[9] D_N Chebysheven diferentziazio matrizea $N+1 \times N+1$ tamainako matrizea da. Literaturan maiz agertzen den hitzarmena da, Chebysheven polinomioek $[-1, 1]$ eremu itxian lan egiten dutelako. Lanaren zehar, espazioen diskretizazioa biren multiploa diren N zenbaki bikoitiekin egingo dugu lan. Aipatuko ez bada ere, D_{127} edo D_{1023} erako Chebysheven matrizeekin egingo dugu lan.

ezin dugu `fft()` algoritmoa *era zuzenean* erabili. Egin nahi dugun simulazioan $(D_N)^2$ eraiki behar dugu, bigarren ordenako deribatu espaziala dugulako, eta honek $O(N^3)$ prozesaketa-denbora behar du. Eragiketen ordena aldatuz, $O(N^2)$ denbora kostea lortu dezakegu, baina 2. taulan `fft()`-aren arintasuna jada demostratu dugu.

Lan honen hedaduraren kanpo geratzen bada ere, Chebysheven diferentziazioa `fft()` algoritmoaren bidez kalkulatzea posible da. Gai honi buruzko informazioa Trefethenen [1, Chap. 6] edo Boyden [2, Chap. 10] (eduki sakonagoa nahi izatekotan) liburuetan aurkitu daiteke. Lan honetan, Praveen Chandrashekar[10] ikerlariaren *ChebPy.py* eta *ChebfftPy.py* moduluak erabili izan dira, Chebysheven diferentziazioa egiten duten funtzioak erabiltzeko.

Arazo hau konponduta, aurreko sistema bera simulatu dugu Chebysheven polinomioekin. `fft()` algoritmoa erabiltzen duen metodoa aukeratu dugu. Gogoa izan behar da x espazioa $[-1, 1]$ eskualdera mugatua dagoela. Hasierako baldintzak moldatu ditugu, Fourierren metodoaren L luzera generikoko sokaren simulazio bera kalkulatu dezan.

Oinarriaren zehaztasuna aztertu baino lehen, Runge-Kutta metodoaren egonkortasun limiteak aztertuko ditugu, aurreko azpiatalean egin den moduan. RK4 metodoa egonkorra izateko onartzen duen denbora-gehikuntza limitea bilatu dugu, bai Fourierren bai Chebysheven oinarrientzako, eta hurrengo emaitzak lortu ditugu:

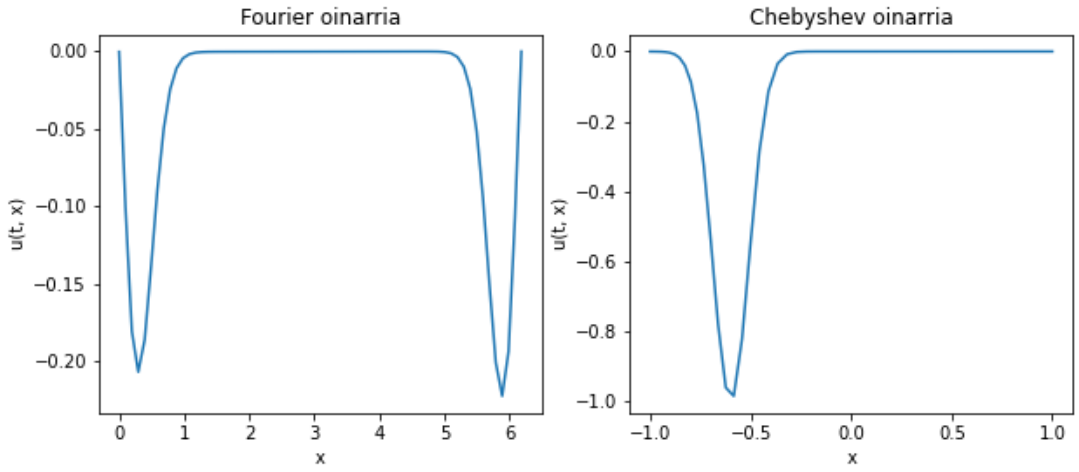
- Fourierren oinarria. Denbora-gehikuntzaren gehienezko tamaina egonkorra $\Delta t = h/2.17$ dugu. Lehenengo 8 segundoak simulatzeko behar izan diren iterazioak: 177. Iterazio bakoitzeko konputazio-denbora: $0.26ms$.
- Chebysheven oinarria. Denbora-gehikuntzaren gehienezko tamaina egonkorra $\Delta t = h/6.06$ dugu. Lehenengo 8 segundoak simulatzeko behar izan diren iterazioak: 1552. Iterazio bakoitzeko konputazio-denbora: $1.90ms$.

Chebysheven oinarriaren erabileraren konputazio kostuaren bi ondorio nagusi atera daitezke: (i) Iterazioak motelagoak izango dira, `fft()`-ren erabilera hain zuzena ez delako, eta (ii) egonkortasun murriztuagoa du, eta iterazio-kopurua handituko da.

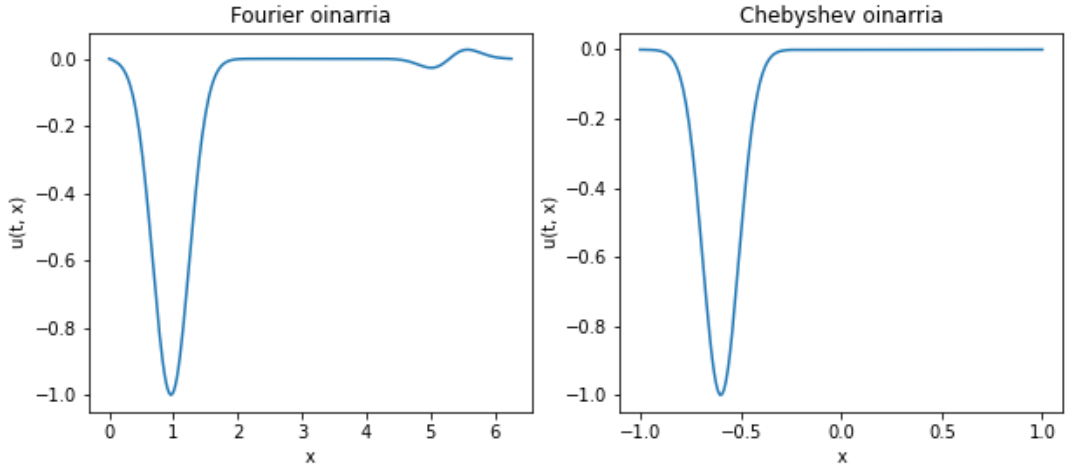
Chebysheven oinarriaren onura, bestalde, espazioaren mugekin talkak (edo edozein elkarrekintza) izateko aukera da. Esate baterako, Dirichleten MB erabili ordez, Neumannen mugalde-baldintzak erabili ditzakegu. Baldintza hauekin $u'(t, -1) = u'(t, 1) = 0$ ezarriko dugu, sistemak materia galerarik izan ez dezan. Uhin ekuazioaren kasuan, ur gainazaleko uhinen portaera ereduztatu dezakegu.

Neumannen MB daukaten sistemak ereduztatu ditugu, Fourierren eta Chebysheven oinarriekin. Helburua Chebysheven oinarriaren zehaztasuna hobea dela frogatzea da. Simulazioa denbora gehiagoren zehar egin dugu, uhinak talka gehiago izateko. Bukaerako uhinaren formak 5. irudian ikusi ditzakegu, N luzera ezberdinetako simulatua. Erraz ikusi daiteke Fourierren oinarriaren hutsa, N puntu kopurua handitzen badugu txikiagotuz badoa ere. Chebysheven oinarriaren simulazioak, ordea, ez dute forma galerarik jasan, N txikiak barne.

[10] Bere Google Scholar webgunea: https://scholar.google.co.in/citations?user=9Mb6H7EAAAAJ



(a) $N = 64$ puntutako simulazioa.



(b) $N = 256$ puntutako simulazioa.

5. irudia: Neumannen mugalde baldintzako uhin ekuazioaren ebazpena. Grafikoek uhinaren egoera irudikatzen dute, $t = 40s$ eta gutxi gorabehera 10 errebote ondoren. Ezkerreko grafikoetan, Fourierren oinarria erabili da, eta zehaztasun galera du. Eskuinekoetan, berriz, Chebysheven oinarria erabili da.

3.3. Bi dimentsioko ekuazioa. 2D uhin ekuazioa: $\mathbf{u_{tt} = c^2(u_{xx} + u_{yy})}$

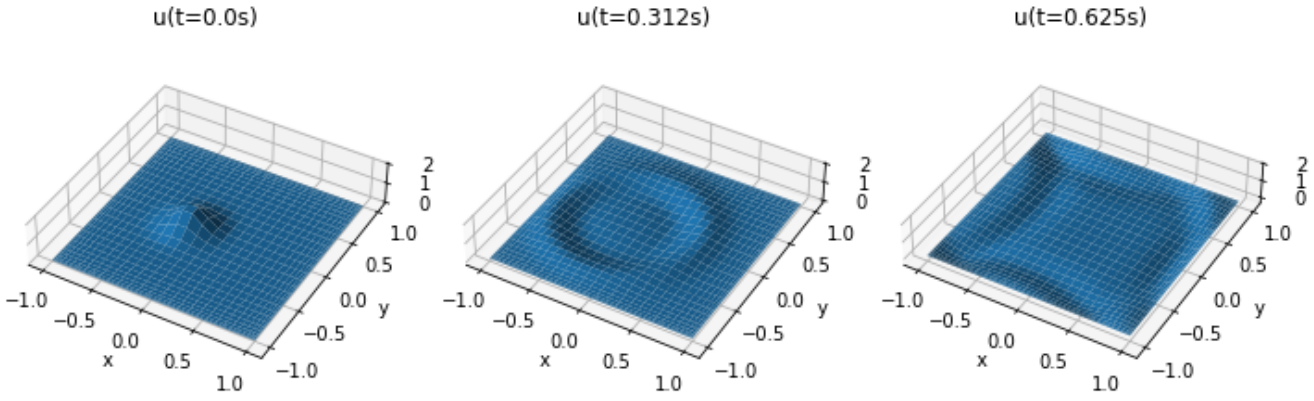
Azkenik, uhin ekuazioaren dimentsionaltasuna handituko dugu, bi dimentsioko sistema aztertuz. Prozedura bera jarraitu behar da. Orain, $\mathbf{u} = (u_1, u_2)$ eta $\mathbf{F} = (F_1, F_2)$ bektoreak bi elementu izango dituzte ere, bere osagaiak u_1, u_2, F_1, F_2 dimentsionaltasuna handituko dutenak izango direlako: lehen bektoreak ziren eta oraingo kasuan, ordea, matrizeak.

Diferentziazio mekanismoak matrizeak kudeatzeko kapazak izatea ahalbidetu behar dugu, aldaketa nagusi bezala. Gainera, zer dimentsioan (ardatzean) diferentziatu nahi dugun agindu behar da. Hortaz aparte, simulazioa 'erraz' prestatu daiteke. Fourierren eta Chebysheven diferentziazio metodoak programatuko dira, bien prozedurak ondo inplementatu ditzakegula ikusteko.

Emaitzak erakusteko ordea, Chebysheven oinarria erabiliko dugu, Neumannen mugalde-baldintza nuluekin. Orain arte N puntu kopuru handia erabili dugu, konputazio limiteak frogatzeko eta metodo ezberdinak alderatzeko, baina $N = 32$ puntuekin (ardatz bakoitzeko) emaitza zehatza lortu dezakegu. Izan ere, metodo espektralek puntu gutxiekin emaitza oso onak atera dezakete. Aukeratutako hasierako egoera hurrengoa izango da:

$$u(t = 0, x, y) = \exp\{-30(x + 0.2)^2\} \cdot \exp\{-30y^2\} \ , \quad u_t(t = 0, x, y) = 0$$

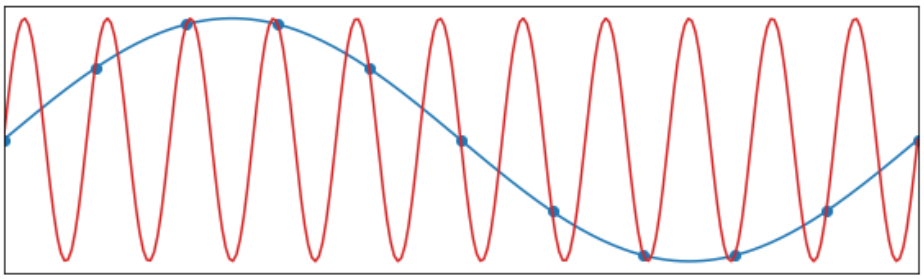
Simulazioak bi oinarriekin ondo atera dira, eta berriz ere Fourierren oinarriak Chebyshevena baino denbora-gehikuntza txikiagoak onartu ditu. Chebysheven oinarriarekin lortutako emaitzak 6. irudian ikusi daitezke.



6. irudia: Bi dimentsioko uhin baten denbora garapena, kutxa karratu baten barruan eta Neumannen mugalde-baldintza nuluak ezarrita. Chebysheven oinarria erabili da, $N = 32$ puntuekin. Ezkerretik eskuinera, une ezberdinetan uhinaren egoeraren grafikoak ditugu.

4. PDE-en ebazpena. Aliasing eta de-aliasing

Metodo espektralak ondo erabiltzeko, aliasing erroreak saihestu behar dira. Aliasing erroreak oinarri finitua dugulako ateratzen dira, gure Fourierren maiztasun espazioko oinarria finitua delako. Hau da, soilik $k \in [-N/2, N/2]$ maiztasunak kontuan izango dira. Irudikatu nahi den funtzioak osagai altuagoak baditu, hauek trunkatuko ditugu, eta aliasing errorea agertuko da. (2) ekuazioan egindako hurbilketa oinarri finitua hartzerakoan onartzen dugun errorea da [3]. Aliasingaren eredu ohikoa 7. irudian azalduta dago.



7. irudia: Aliasingaren eredu sinplea dugu. $\sin(11x)$ funtzio gorritik eta $\sin(x)$ funtzio urdinetik lagin diskretu bera atera daiteke, lagin maiztasun egokia aukeratuz gero. Kasu honetan, lagin-maiztasuna $h = 2\pi/10$ da. Hortaz, Fourierren espazioan ondo prozesatu daitekeen maiztasun maximoa $k_{max} = 10$ da. Lagin maiztasun handiago batekin aldiberekotasun hau lortzea ezinezkoa da.

PDE-ak ebazterakoan izango dugun aliasing iturri nagusia funtzioen biderketa izango da. (6) ekuazioan azaldu genuen bezala, bi $y_1(k)$ eta $y_2(k)$ funtzioen maiztasun sorten biderketa egiteko, konboluzioa egin dezakegu. Egia esanda, pare bat gauza kontuan izan behar dira biderketa hauek egiteko, eta lanaren sekzio honetan aurkeztuko dugu nola egingo diren. Hasteko, konboluzioaren adierazpen matematikoa sinplea bada ere, honen kalkulua $O(N^2)$ denbora behar du. Hortaz, bi biderkarien alderantzizko Fourierren transformatua kalkulatzea azkarragoa izango da, espazio zuzenean biderketa eginez. Hurrengo eskema jarraituko da:

$$y_1(k),\ y_2(k) \quad \xrightarrow{\texttt{ifft()}} \quad w(x) = y_1(x) \cdot y_2(x) \quad \xrightarrow{\texttt{fft()}} \quad w(k)$$

Fourierren transformatu azkarra $O(N \ln N)$ denbora behar du, eta biderketa normala, $O(N)$. Guztira, prozesu berriak hobe eskalatuko du N zenbaki handiekin. Baina, lehen esan dugun bezala, biderketan aliasing errorea atera daiteke. Hau ulertzeko, (6) adierazpena kasu finituan nola aldatu den ikusi behar dugu:

$$w(x) = y_1(x) \cdot y_2(x) = \left(\sum_{k=-N/2}^{N/2} a_k e^{ikx} \right) \left(\sum_{l=-N/2}^{N/2} b_l e^{ikx} \right) = \sum_{k=-N}^{N} c_k e^{ikx} \tag{23}$$

Biderketak gure oinarriaren kanpo diren maiztasunak eman ditzake, aliasing errorea sortuz. Arazo hau konpontzeko hainbat metodo existitzen dira. Lan honetan 2/3 eta 3/2 legeei

buruz hitz egingo dugu [3].

Trunkaketa. 2/3 legea.

Lehen metodo honek inplementazio zuzena du, baina informazio galera dakar. Izan bitez $\{a_k\}$ eta $\{b_k\}$ biderkatu nahi diren bi koefiziente segida diskretuak, non $k \in [-N/2,\ N/2]$. Biderkadura eta gero maiztasun altuegiak ez izateko, $\{a_k\}$ eta $\{b_k\}$-tik maiztasun muturreko maiztasunak kenduko ditugu, hau da, $\{a_{k'}\}$ eta $\{b_{k'}\}$ non $k' \in [-m,\ m]$ den. Informazio minimoa galtzeko, $m = 2/3 \cdot N/2$ izan behar dela frogatu daiteke. Horrela, $[-m,\ m]$ maiztasunek ez dute aliasingik izango.

Zeroz betetzea. 3/2 legea.

Bigarren metodoak informazio galerarik ez du. Printzipio basikoa oinarria luzatzea da. Kasu honetan, maiztasunak $k' \in [-m,\ m]$ eremuan kontsideratuko ditugu, non $m = 3/2 \cdot N/2$ izango da kasu honetan. Metodo hau erabiltzeko, biderkatu nahi diren funtzioak zerorekin osatu behar dira, -$3N/4$tik -$N/2$ra eta $N/2$tik -$3N/4$ra. Ondoren, $3N/2$ luzerako `fft()` eta `ifft()` metodoak inplementatu daitezke, bi funtzio luzatuak espazio zuzenean biderkatzeko [3].

Fourierren espaziora bueltatzean, $2N + 1$ luzerako oinarria berreskuratu dezakegu, maiztasun txikiak hartuz. `fft()`-aren normalizazio konstantea aldatuko dela kontuan izan behar da. Metodo hau *Python*en funtzio batean inplementatuko dugu, lanaren zehar esfortzurik gabe erabiltzeko.

4.1. Eredua: PDE ez lineala

3/2 legearen inplementazioa (24) adierazpen matematikoa duen ekuazio ez-linealean egingo dugu. Ikusi daitekeenez, biderketa bat egin behar da, hortaz aliasingak zer eragin duen ikusi dezakegu.

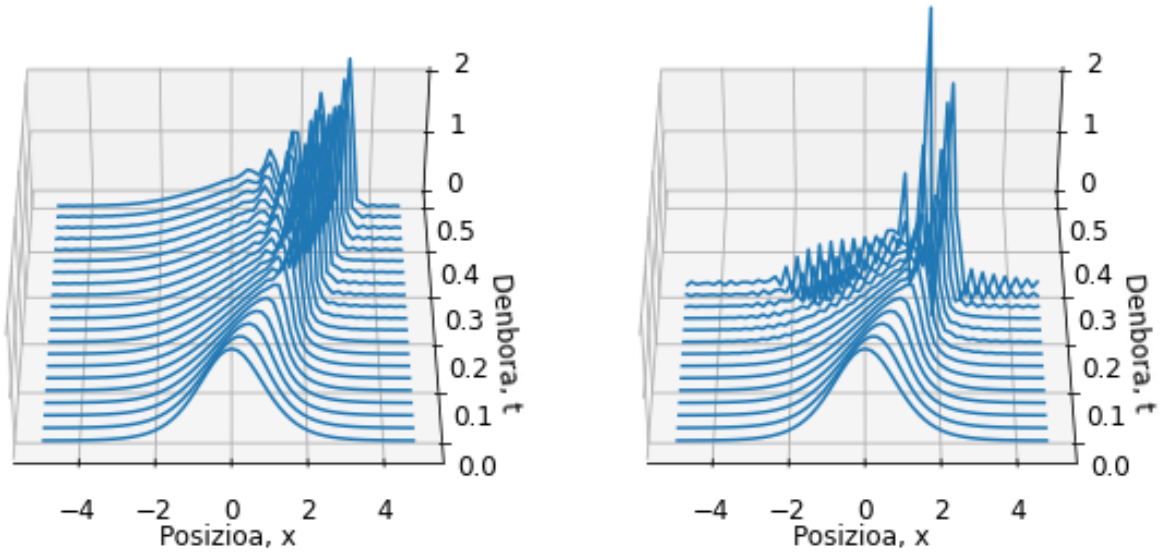
$$u_t + 6u \cdot u_x = 0 \tag{24}$$

Ekuazio hau Korteweg de Vries (KdV) ekuazioaren kasu berezia da, non dispertsio terminorik ez dagoen. Hortaz, energia metaketa izango dugu, emaitza singularrak lortu arte. Ekuazio hau aukeratu dugu hurrengo atalean KdV ekuazioa aztertu dugulako, eta jada bere soluzio analitikoak dakizkigu. Ondorioz, gure emaitzak soluzio teorikoarekin alderatzea posible da. Burutuko dugun simulazioan, hasierako baldintza (25) *solitoia* izango dugu. Hurrengo atalean solitoi hauen frogapena egingo dugu.

$$u(x, t = 0) = -\frac{1}{2}c \cdot \operatorname{sech}^2 \frac{1}{2}(\sqrt{c}x) \tag{25}$$

Printzipioz, uhinaren energia metatuko da oszilazio bortitzak emanez. Hortaz, de-aliasing teknika ezberdinen erabileraren efektua ikusteko aukera aproposa dugu.

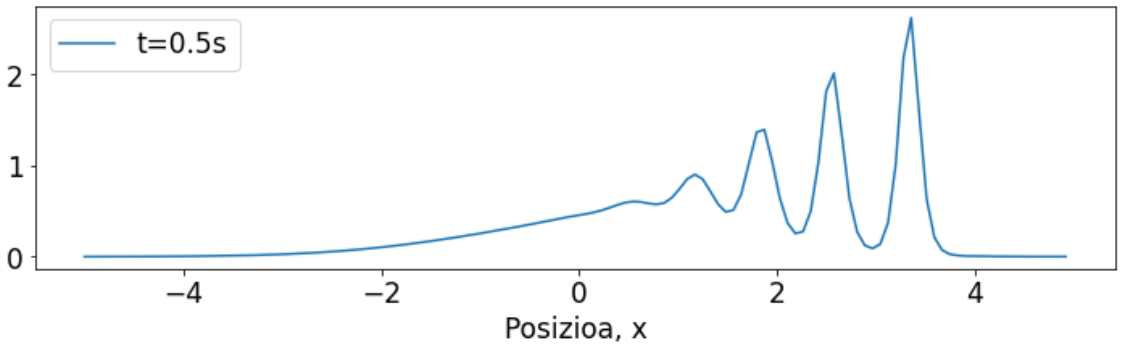
$N = 128$ puntuko simulazioa egin dugu, eta oszilazio oso bortitzak jasaten ditu $t = 0.25s$-tik aurrera. 3/2 betetze teknika erabili dugu eta $t \in [0, 0.4s]$ artean simulazio ona izan dugu. De-aliasing teknikarik ez badugu erabiltzen, simulazioa azkar ezegonkortzen da. Bi soluzioen arteko aldeak 8. irudian ikusi dezakegu (9. irudian betetze teknika erabiliz lortu dugun soluzioa hobe ikusi dezakegu).



(a) Zeroz betetzearen (3/2) teknika.

(b) De-aliasing teknikarik erabili gabe.

8. irudia: KdV ekuazioaren soluzio oszilakorraren zehetasuna. Oszilazioak $t \sim 0.25s$ unean hasten dira. (a) Simulazioa ondo atera da eta oszilazioak ondo bildu dira. (b) Oszilazioak hastean, azkar ezegonkortu da simulazioa, `NaN` balioak emanez.



9. irudia: KdV ekuazioaren soluzioa, $t = 0.5s$ unean, RK4 algoritmoa eta 3/2 de-aliasing teknika erabilita. *Datuak:* $N = 128$, $\Delta t = 10^{-3}$

5. Korteweg de Vries ekuazioa

Hemendik aurrera, Korteweg de Vries (KdV) ekuazioarekin egingo dugu lan. Ekuazio hau sakontasun gutxiko uraren portaera simulatzeko erabilia da. Bere adierazpen matematikoa (26) ekuazioan ikusi dezakegu. Konstanteen aukera ohikoa $\alpha = 1$ eta $\beta = 6$ da, eta simulazioetan zehar konstante hauek ere erabiliko ditugu.

$$u_t + \alpha u_{xxx} + \beta u \cdot u_x = 0 \tag{26}$$

Ekuazio hau aukeratu dugu ez-linealtasuna eta difusioa nahasten duen ekuazioa delako. Gainera, ekuazio hau soluzio analitikoak dituen ekuazio ez-lineala da, hau da, PDE-aren ebazpen zehatza posiblea da[11]. Hortaz, ekuazio oso garrantzitsua da fisika-matematikoaren arloan. Ekuazioaren soluzioa *solitoia* da, ikerkuntza eta garapenaren arloan arreta asko jasan duen uhin mota [7]. Adibidez:

- Matematika aplikatua: ur-uhinak, zuntz optikoak...
- Materialen fisika: ferro-elektrizitatea (Van der Waals materialak: MoS_2 eta grafenoa)
- Partikulen fisika: Oinarrizko partikulen ereduak, nukleoien ereduak (Skyrmeren eredua).
- Breathers. Bi solitoien egoera lotua [8].

Solitoiak

Solitoia (uhin bakartiak, *solitary waves* ingelesez) bere forma mantentzen duen uhin hedakorra da. Haien definizio formala emateko, Drazin eta Johnsonen liburura jo dugu [7]. Solitoia hurrengo ezaugarriak betetzen duen EDPen soluzioa da; (i) lokalizatua, (ii) bere forma denboran zehar mantentzen du eta (iii) beste solitoiekin izandako elkarrekintzek ez dute bere forma aldatzen. KdV ekuazioaren soluzioek hiru baldintza hauek betetzen dute, hortaz solitoiak dira.

Demagun $u(x,t) = f(x - ct)$ formako duen eta KdV ekuazioa betetzen duen uhina nahi dugula, hau da, c abiaduraz mugitzen den uhina. (26) ekuazioan ordezkatzen badugu, $\alpha = 1$ eta $\beta = 6$ izanda:

$$-cf' + 6ff' + f''' = 0$$

non $()'$ deribazioa $x - ct$ aldagaiarekiko den. Behin integratuz:

$$-cf + 3f^2 + f'' = A$$

[11] Atzeranzko sakabanaketa transformazioa deituriko metodoaren bidez ebatzitako ekuazioa da (backwards scattering transform, ingelesez) [7].

non A integrazio konstantea den. f' bi aldeetan biderkatuz eta berriz integratuz:

$$-\frac{1}{2}cf^2 + f^3 + \frac{1}{2}(f')^2 = Af + B$$

non B beste integrazio konstantea den. Orain mugalde baldintzak aplikatuko ditugu. Uhin bakartia espazioan lokalizatua dago, beraz $x \to \pm\infty$ puntuetan, funtzioa eta bere deribatuak zero izan behar dira, hau da, $f|_{\pm\infty} = f'|_{\pm\infty} = f''|_{\pm\infty} = 0$. Horretarako, $A = B = 0$ izan behar dira:

$$(f')^2 = cf^2 - 2f^3$$

Lortutako ekuazioa Ricatti motakoa da, zehazki ebatzi daitekeena, hurrengo integralaren bidez:

$$\int \frac{df}{f\sqrt{c-2f}} = \pm(x - ct)$$

Integrala ebazteko, $f = \frac{c}{2}\text{sech}^2\theta$ ordezkapena erabiliko dugu.

$$\frac{df}{f\sqrt{c-2f}} = \frac{-c\text{sech}^2\theta\tanh\theta d\theta}{c/2\text{sech}^2\theta\sqrt{c}\sqrt{1-\text{sech}^2\theta}} = -\frac{2}{\sqrt{c}}\frac{c\text{sech}^2\theta\tanh\theta d\theta}{c\text{sech}^2\theta\tanh\theta} = -\frac{2}{\sqrt{c}}d\theta$$

Ondorioz,

$$-\frac{2}{\sqrt{c}}\int d\theta = -\frac{2\theta}{\sqrt{c}} = (x - ct) \implies f = \frac{c}{2}\text{sech}^2\theta = \frac{c}{2}\text{sech}^2\frac{\sqrt{c}}{2}(x - ct)$$

Lortutako emaitzak solitoien hedapena adierazten du. Hauek c abiadurarekin higitzen diren uhinak dira, eta (27) adierazpenaren bidez definituko ditugu. Ikusi daitekeenez, solitoien abiadura, altuera eta zabalera c parametroaren menpe daude: Azkar mugitzen diten uhinak altuak eta argalak izango dira.

$$u(x,t) = f(x - ct) = \frac{c}{2}\text{sech}^2\frac{\sqrt{c}}{2}(x - ct - x_0) \tag{27}$$

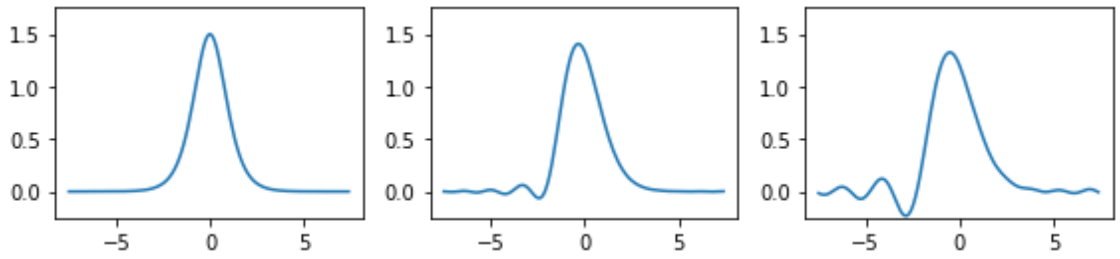
Solitoia difusio eta elkarretaratze indarren arteko oreka da. Haien portaera hobe ulertzeko, oreka apurtzen denean zer gertatuko den ikusiko dugu.

Lehen, KdV ekuazioaren gai ez-lineala kenduko dugu, $\beta = 0$ eginez. Geratzen zaigun ekuazioa $u_t + u_{xxx} = 0$ da, eta bere izaera difusiboa izango da. Hortaz, uhina zabaldu eta bere energia sakabanatuko da. Ikusi 10. irudia.

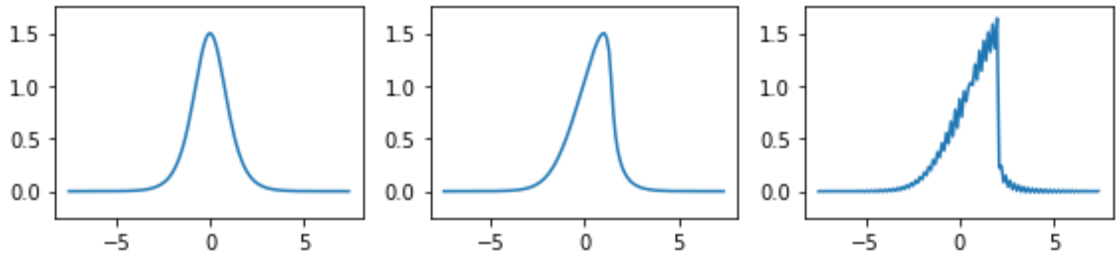
Orain, beste gaia kenduko dugu, hau da gai dispertsiboa, $\alpha = 0$ eginez. Geratzen zaigun ekuazioa $u_t + 6u \cdot u_x = 0$ da, eta bere izaera metakorra izango da. Hortaz, uhina bildu eta bere energia metatuko da, singulartasunak agertu arte. Ikusi 11. irudia.

Solitoien arteko talkak

Metodo numerikoen abantaila objektuen arteko elkarrekintzak erraz simulatu dituela da. Behin KdV ekuazioa eta haien solitoiak ondo inplementatu ditugunean, haien arteko talka



10. irudia: Solitoi baten dispertsioa, bere forma mantendu ezin duen ingurunean.



11. irudia: Solitoi baten metaketa, bere forma mantendu ezin duen ingurunean.

nola garatuko den ikusi dezakegu. Solitoien ezaugarriak direla medio, haien forma ez da talkaren ostean hautsi behar.

Bi solitoi simulatu ditugu, $c = 1$ eta $c = 3$ abiadurak dituztenak, eta haien arteko talka $x = 0$ puntuan eta $t = 5s$ unean gerta dezan simulazioa prestatu dugu. Emaitzak 12. irudian bildu ditugu.

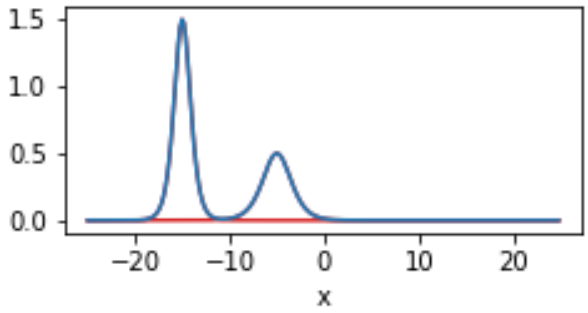
5.1. RK4 metodoa Fourierren espazioan

KdV ekuazioa simulatzeko, orain arte azaldu dugun teoria guztia praktikan jarriko dugu. Runge-Kutta 4. ordenako metodoa erabiliko dugu. Deribatu espazialak sinplifikatzeko, Fourierren espazioan egingo dugu lan, non $\partial_x \to ik$ bihurtuko da. Hau eginez, (26) hurrengo eran adieraziko dugu:

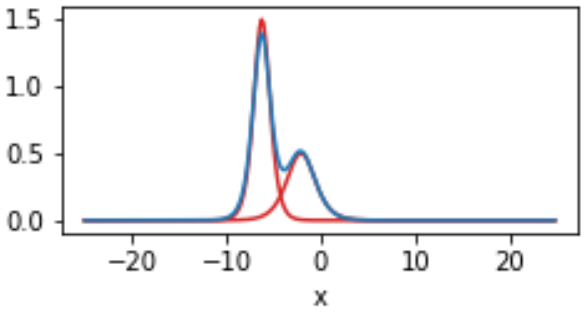
$$\hat{u}_t = -(ik)^3\hat{u} - 6(\hat{u} * ik\hat{u}) \tag{28}$$

Ikusi daitekeenez, ekuazioan zati lineala $(ik)^3\hat{u}$ eta zati ez-lineala $6(\hat{u} * ik\hat{u}) = 6ik(\hat{u} * \hat{u})$ dugu. Zati ez lineala aztertzeko, de-aliasing teknikak erabili ditugu (konboluzioa egin ordez).

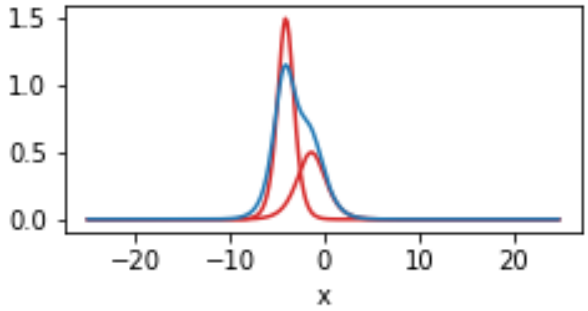
*Python*en bidez, RK4 zenbakizko metodoa aplikatu dugu. Solitoien simulaketa guztiak $L = 50$ luzerako tartean egin ditugu (hau da, $x \in [-25, 25]$). Denborari dagokionez, $t = 10s$ luzerako simulazioa egin ditugu. Atal honetan zehar bi kasu aztertuko dira: Solitoi bakar baten hedapena eta bi solitoien arteko talka. Ikusi 13. irudia.



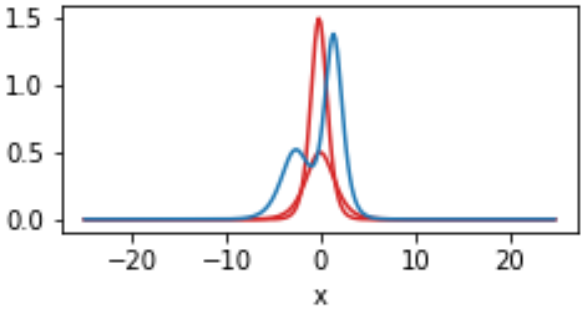
(a) $t = 0s$



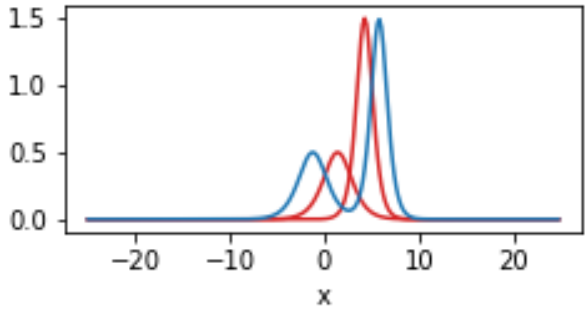
(b) $t = 2.93s$



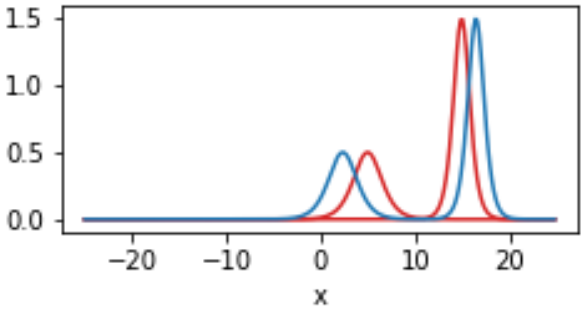
(c) $t = 3.65s$



(d) $t = 4.95s$

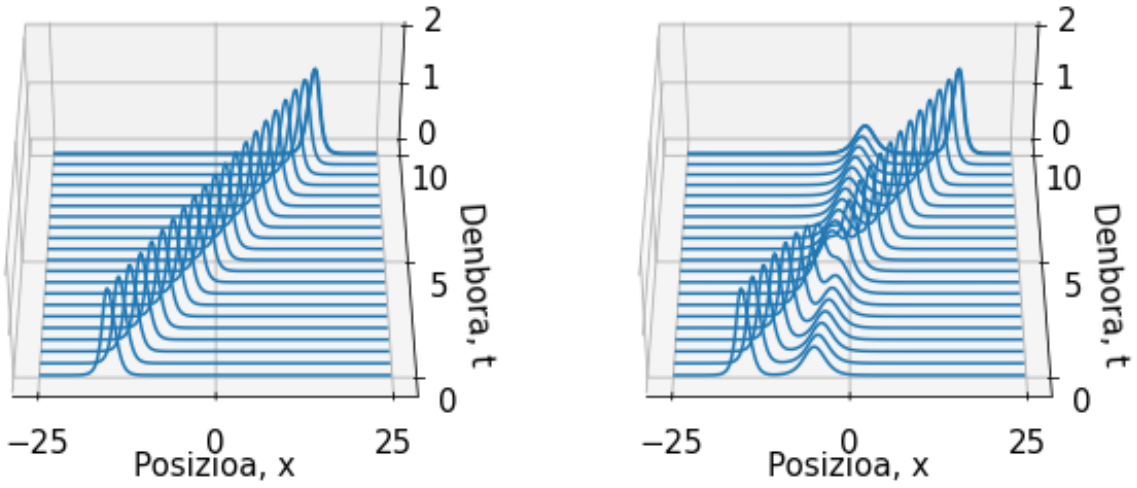


(e) $t = 6.43s$



(f) $t = 10s$

12. irudia: KdV bi solitoien arteko talkaren eboluzioa. Bi solitoiek $c = 1$ eta $c = 3$ abiadurak dituzte, eta bata bestearen gainetik pasa da. Gorriz, solitoien adierazpen teorikoa, eta urdinez, bien arteko talkaren simulazioa. Ikusi daitekeenez, solitoien forma ez da aldatzen, baina haien posizioak talka baino lehen zituztenak ez dira.



(a) Soliton bakarra, $c = 3$ abiadurarekin.

(b) Bi solitoien talka, $c = 1, 3$ abiadurekin.

13. irudia: KdV solitoien simulazioak.

Egonkortasuna. Gehienezko denbora-gehikuntza

Dakigunez, egonkortasun limite txikia izateak konputazio denbora asko handitu dezake. Hasteko, RK4-ren egonkortasuna aztertuko dugu, denbora-gehikuntza limitea zein den jakiteko. Froga hau N puntu ezberdinetarako egingo dugu, 3. taularen emaitzak lortuz. N puntu kopurua 2^nren potentziak izango dira beti, `fft` algoritmoa eraginkorra izateko.

3. taula: RK4 metodoak onartzen duen denbora-jauzi maximoa (N puntu kopurua aldatuz), simulazioa egonkorra izan dezan.

Puntu kopurua: N	Gehienezko denbora-gehikuntza: $\Delta t_{lim}(s)$	Konputazio denbora: $t(s)$
32	0.15	0.016
64	0.04	0.078
128	0.005	0.64
256	0.0006	6.8
512	0.00008	62

Emaitzei begira, jauzi limitea oso azkar txikiagotzen dela ikusi dezakegu. Izan ere, hirugarren ordenako deribatua dugu ekuazio diferentzialean, eta izaera oszilakorra du. Limite hauek ez dira egokiak. Askotan, denbora-gehikuntza handiak egiteagatik ere ez da zertan doitasuna galdu.

Zehaztasuna

Hurrengo zeregina zehaztasuna aztertzea da. Hurrengo ezaugarrien bidez erabakiko da zein zehatz den algoritmoa:

- Alde batetik, simulazioaren errorea zein txiki egin dezakegun ikusi nahi dugu.
- Beste aldetik, zehaztasun hori lortzeko zer diskretizazio parametroak (espazialak eta denboralak) behar dugun balioztatuko da.

Simulazioaren errorea jakiteko, (27) solitoien adierazpen teorikoa eta lortuko ditugun emaitzak alderatuko ditugu. Hasteko, N puntu kopuru optimoa zein den jakin nahi dugu. Horretarako, solitoi baten simulazioa prestatu dugu, eta $N = 32$-tik 512-ra errepikatu dugu simulazioa. Errorea kalkulatzeko (12) erabili dugu. Emaitzak 4. taulan bildu ditugu. Ondorio bezala, $N = 256$ aukera optimoa dela ikusi dezakegu.

4. taula: Simulazioaren errorea, N puntu kopuru ezberdinetarako. Denbora-gehikuntza: $\Delta t = 7.5 \cdot 10^{-5}$

Puntu kopurua: N	Errorea: ϵ
32	$2.88 \cdot 10^{-2}$
64	$1.85 \cdot 10^{-3}$
128	$2.07 \cdot 10^{-6}$
256	$8.96 \cdot 10^{-9}$
512	$9.55 \cdot 10^{-9}$

Ondoren, denbora gehikuntza zein txiki egin behar dugun jakin nahi dugu. Aztertzen ari garen RK4 metodoaren kasuan, ordea, bigarren zeregin hau ezin dugu egin. Izan ere, gehikuntzak mugapen handiak jartzen dizkigu, eta denbora-gehikuntza limiteekin jada zehaztasun ona lortzen dugu.

Haatik, hurrengo orrialdeetan metodo ezberdinak proposatuko ditugu, arazo hau konpontzen dutenak.

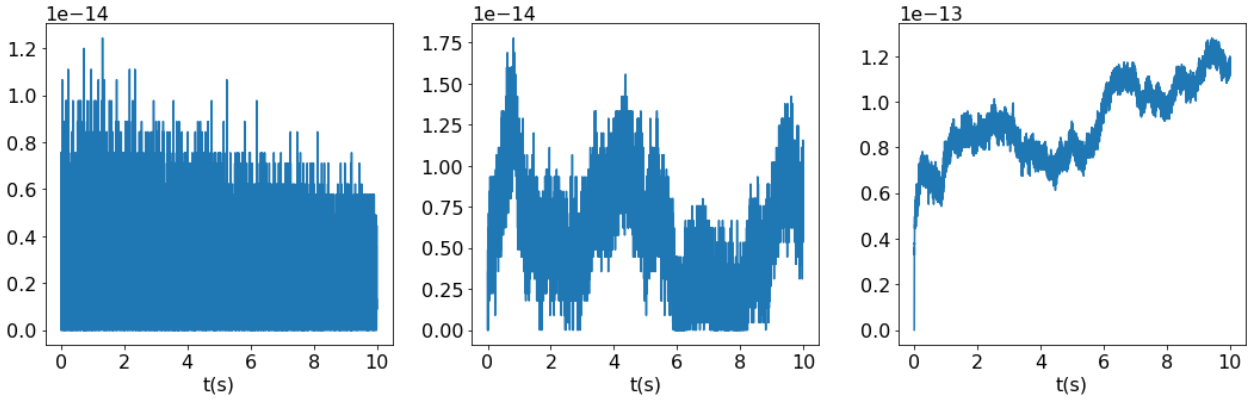
Kontserbazio legeak

Korteweg de Vries-en ekuazioak, izan ere, kontserbatzen diren magnitude infinituak ditu [7]. Guk, ordea, lehenengo hiru magnitudeetan jarriko dugu arreta:

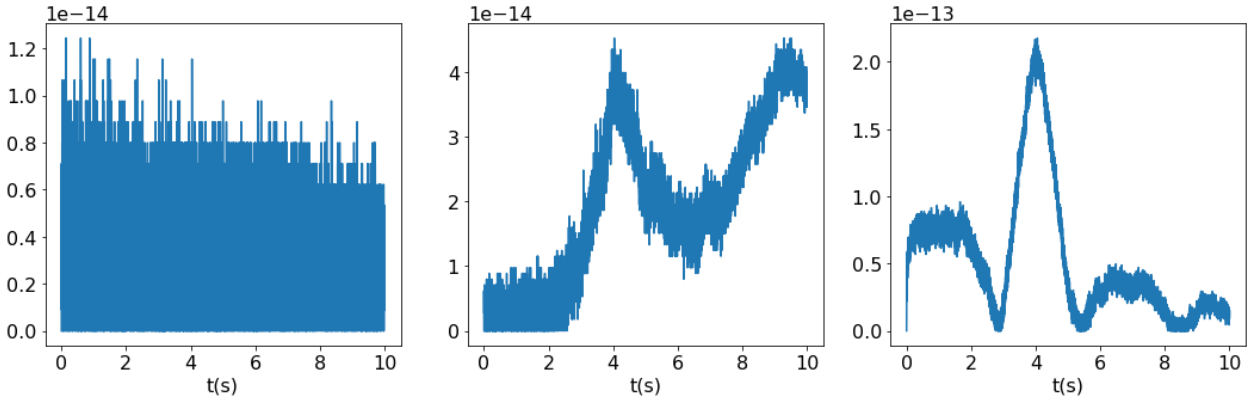
1. Masa: $m = \int_{-\infty}^{\infty} u dx$
2. Momentua: $p = \int_{-\infty}^{\infty} u^2 dx$
3. Energia: $E = \int_{-\infty}^{\infty} \left[2u^3 + (u_x)^2\right] dx$

Integralak egiteko Newton-Cotesen $n = 1$ legea erabili dugu (hots, trapezioaren legea). Lehen aipatutako bi kasuak aztertu ditugu: solitoi baten hedapena eta bi solitoien arteko talka. Magnitude kontserbatuen grafikoak 14. eta 15. irudietan bildu ditugu.

Grafikoen y ardatzek argi usten dute erroreak infinitesimalak direla eta beraz magnitude hauek konstante direla: 10^{-14} eta 10^{-13} doitasuna dugu. Bi kasuetan erroreen ordenak berdinak dira. Izan ere, simulazioa zehatza izan da solitoien arteko talkan zehar (erroreak handitu arren, gero egonkortu dira).



14. irudia: Magnitude kontserbatuen erroreen denbora garapena $|m(0)-m(t)|$, $|p(0)-p(t)|$ eta $|E(0)-E(t)|$, hurrenez hurren. Solitoi baten simulazioa. Erroreak infinitesimalak dira.



15. irudia: Magnitude kontserbatuen erroreen denbora garapena $|m(0)-m(t)|$, $|p(0)-p(t)|$ eta $|E(0)-E(t)|$, hurrenez hurren. Bi solitoien arteko talkaren simulazioa. Erroreak infinitesimalak dira.

5.2. IFRK4 metodoa (Faktore integratzailea)

Simulazioa hobetzeko, Runge-Kutta metodoa faktore integratzailearekin aplikatuko dugu (hau da, IFRK4, *Integrating Factor* RK4 ingelesez). Metodo honek ekuazio diferentzialaren zati lineala zehazki ebatziko du, aldagai aldaketa baten bidez. Zati lineala zehazki ebazteak bi abantaila nagusi dakartza: (i) simulazioaren zehaztasuna handituko du eta (ii) RK4 algoritmoa egonkortuko du [9]. Runge-Kutta metodoa egonkortzen bada, Δt denbora-gehikuntza handiagoak erabiltzeko aukera emango du, simulazio-denbora txikiagotuz.

Faktore integratzailea aplikatzeko, gure PDE-aren zati lineala eta ez-lineala banandu behar ditugu. Azalpena sinpletzeko, zati lineala c konstantea izango dugu. Beraz, ekuazioa:

$$u_t = c \cdot u + F(u,t) \tag{29}$$

Ekuazioaren zati lineala soilik kontuan hartuz, $u_t - c \cdot u = 0$, ikusi daiteke $\mu(x,t) = \exp(ct)$ motako soluzioak dituela. Hau izango da, beraz, gure faktore integratzailea. Faktore integratzailearen alderantzizkoa (29) ekuazioaren bi zatitan jarriz:

$$\begin{aligned} e^{-ct}(u_t - c \cdot u) &= e^{-ct}F(u,t) \\ e^{-ct}u_t - c \cdot e^{-ct}u &= e^{-ct}F(u,t) \\ (e^{-ct}u)_t &= e^{-ct}F(u,t) \\ v_t &= F(e^{ct}v, t) \end{aligned} \tag{30}$$

$v = \exp(-ct) \cdot u$ aldagai aldaketa eginez, ekuazioaren zati lineala kentzen zaigu. (30) ekuazio diferentzial ez-lineal purua da, eta metodo espektralen inplementazio errazagoa espero dugu. Ekuazio hau RK4 algoritmoaren bitartez simulatu daiteke, eta nahi dugun emaitza lortzeko $u = \exp(+ct) \cdot v$ aldaketa desegin dezakegu.

Badakigunez, KdV ekuazioa ez da lineala, baina Fourierren maiztasunen espazioan lan egitean, zehazki ebatzi dezakegun zati lineala lortzen dugu (ikusi (28) adierazpena). Izango dugun faktore integratzailea (31) da. k koefiziente konstantedun bektorea da, beraz integrazio faktore metodoa zuzen aplikatu dezakegu, N ekuazio diferentzial kontsideratuz (*Python*en bidez sistema era bektorialean ebatziko dugu).

$$\mu(x,t) = \exp\{-(ik)^3 t\} \tag{31}$$

Egonkortasuna eta zehaztasuna

Egonkortasuna da konpondu nahi dugun arazo handiena, RK4 metodoak egonkortasunaren oso eskalatze txarra baitu. Hortaz, konputazio denbora azkar bihurtzen da arazo. Lehen egindako esperimentu bera egin dugu, faktore integratzailedun RK4-ren egonkortasuna aztertuz. Froga hau N puntu ezberdinetarako egin da, 5. taularen emaitzak lortuz.

Emaitzei begira, Δt limitea asko hobetu da. Alde batetik, limite malguagoak ditugu. Bestalde, limitea ez da hain azkar txikiagotzen N zenbakiarekin. Hortaz, zehaztasun handiko N puntu kopuru handiekin lan egin dezakegu, konputazio denbora hain luzeak izan gabe.

Ondoren, zehaztasun azterketa egin dugu, simulazioen emaitzak datu teorikoekin alderatuz. Lehenik, N puntu kopurua handitu dugu simulazioaren errorearen balioa ezin hobetu arte(Δt konstante batekin). Gero, N kopuru hori konstante utziz, denbora-gehikuntza optimoa bilatu dugu. Emaitzak lanaren hurrengo orrialdetan azalduko dira, beste algoritmoekin alderatuta. Dena den, esperimentu hauetatik hurrengo ondorioak atera ditugu:

5. taula: IFRK4 metodoak onartzen duen denbora-gehikuntza maximoa (N puntu kopurua aldatuz), simulazioa egonkorra izan dezan.

Puntu kopurua: N	Gehienezko denbora-gehikuntza: $\Delta t_{lim}(s)$	Konputazio denbora: $t(s)$
32	0.5	< 0.001
64	0.15	0.03
128	0.065	0.0625
256	0.025	0.172
512	0.012	0.469
1024	0.0055	1.89
2048	0.0025	6.11
4096	0.00125	23.5
8192	0.0006	198

1. Zehaztasuna ez da hain ona, RK4 metodoa estandarrarekin alderatuz.
2. Beharrezko N puntu-kopurua oso baxua da. $N = 128$ puntutik aurrera ez da hobekuntzarik lortzen.
3. Haatik, denbora gehikuntza asko txikiagotu behar da zehaztasun onena lortzeko. $\Delta t = 10^{-5}$ da aukera egokiena.

Dena dela, kontsideratzen ari garen erroreak oso txikiak dira, eta aplikazio askotarako denbora gehikuntza handia erabili daiteke. Izan ere, metodo espektralek N eta Δt txikiekin emaitza onak lortzen dituzte. Faktore integratzailearen bidez, egonkortasunaren arazoa konpondu da, eta N eta Δt txikiak erabiltzeko aukera dugu, simulazio zehatz eta azkarrak lortuz.

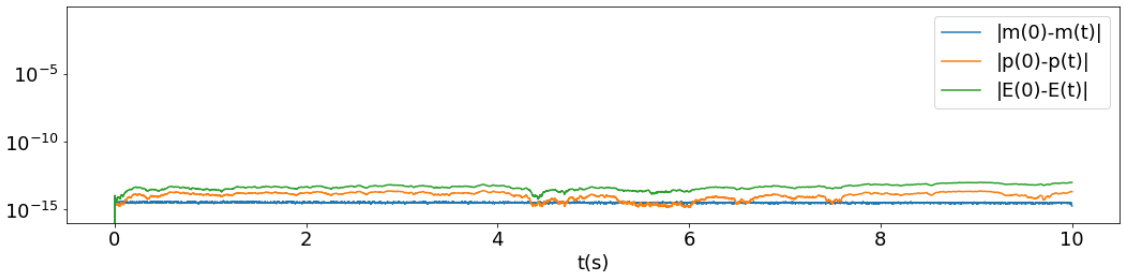
Kontserbazio legeak

Azkenik, masa, momentu eta energia zein ondo kontserbatzen diren ikusiko dugu.

Magnitudeak konstante mantentzen dira, baina bai zarata bai oszilazioak dituzte. Lehen, erroreak eskala linealean adierazi ditugu, 14. eta 15. irudietan. Ezaugarri hauek ez dira gure interesekoak, eta horregatik, hemendik aurrera erroreak iragazki batetik pasatuko ditugu, eta irudiek eskala logaritmikoa izango dute. Honela, errorearen ordena eta bere garapen orokorra errazago aztertuko dugu.

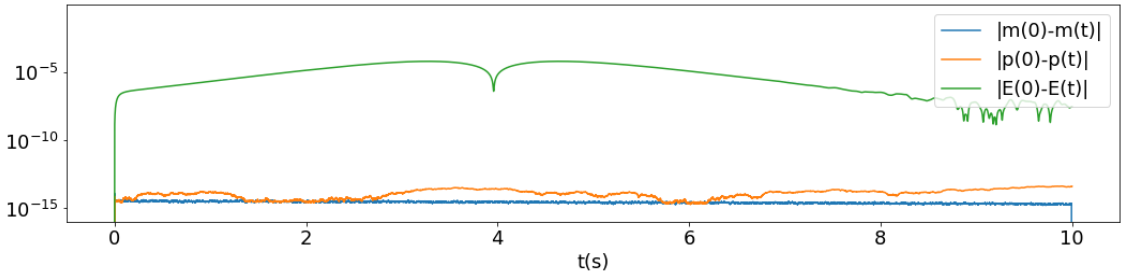
Bi solitoien arteko talka aztertu dugu.

- RK4 metodoarekin lehen lortutako datuak 16. irudian jarri ditugu, era logaritmikoan eta iragazkia pasata.
- Faktore integratzailedun metodo berriaren emaitzak 17. irudian bildu dugu. Simulazioaren zehaztasuna baxuagoa denez, magnitude kontserbatuen erroreak RK4-koak



16. irudia: Magnitude kontserbatuen erroreen denbora garapena $|m(0)-m(t)|$, $|p(0)-p(t)|$ eta $|E(0)-E(t)|$, hurrenez hurren. RK4 metodoa erabili da. Bi solitoien arteko talkaren simulazioa. *Datuak:* $N = 256$, $\Delta t = 10^{-4}$. *(Exekuzio-denbora: 51s.)*

baino handiagoak izatea espero dugu. Haatik, magnitudeak konstanteak dira eta erroreak, txikiak (energiarena izan ezik). Datu hauek lortzeko, simulazioaren diskretizazioa egiteko lehen aipatu ditugun parametro optimoak baino txikiagoak hartu ditugu, emaitzak hobetuz.



17. irudia: Magnitude kontserbatuen erroreen denbora garapena $|m(0)-m(t)|$, $|p(0)-p(t)|$ eta $|E(0)-E(t)|$, hurrenez hurren. Bi solitoien arteko talkaren simulazioa. Faktore integratzailedun RK4 metodoa erabili da. *Datuak:* $N = 512$, $\Delta t = 10^{-4}$. *(Exekuzio-denbora: 61s.)*

5.3. ETDRK4 metodoa (Denbora diferentziazio esponentziala)

Inplementatuko dugun azken metodoa ETDRK4 (ingelesetik, *Exponential Time Differencing* RK4) izango da. Metodo hau oszilazio bortitzak dituzten sistemak ebazteko pentsatuta dago, malda handiko funtzioak zehaztasun bikainarekin ebatziz. Metodo honek ekuazioaren zati lineala zehazki ebazten du ere [10]. Hortaz, faktore integratzailearekin lortutako emaitzekin alderatuko dugu, ea azkarragoa edo zehatzagoa den. Lehen bezala, gure ekuazio diferentzialak zati lineala eta ez-lineala izango ditu:

$$u_t = c \cdot u + F(u, t)$$

ETD metodoaren $\mu = \exp(-ct)$ faktore integratzailea biderkatuko dugu goiko ekuazioaren bi zatitan, lehen egin izan dugun bezala. Orain, ordea, lortu den ekuazioa integratuko dugu, denbora pauso generiko batean zehar (hau da, $t = t_n$ eta $t = t_{n+1} = t_n + h$ denboren artean). Lortuko den formula hurrengoa da:

$$u(t_{n+1}) = u(t_n)e^{ch} + e^{ch}\int_0^h e^{-c\tau}F(u(t_n+\tau), t_n+\tau)d\tau \tag{32}$$

Formula hau zehatza da eta zenbakizko metodoak integralaren hurbilketa nola egiten den arabera aterako dira [10]. Guk erabili nahi ditugun metodoak Runge-Kutta familiakoak dira, hasierako baldintzen inizialiazioa erraza delako eta metodo zehatzak eta egonkorrak direkalo. Bai ETDRK2 bai ETDRK4 inplementatu dira, baina formulak eta emaitzak bakarrik ETDRK4-koak azalduko dira, errore ordena txikiagoak dituelako eta RK4-ekin alderatu ditzakegulako. Cox eta Matthewsen lanatik lortu dugun adierazpena, ETDRK4 kasurako, hurrengoa da:

$$\begin{aligned}
a_n &= u_n \cdot e^{ch/2} + (e^{ch/2}-1)F(u_n, t_n)/c \\
b_n &= u_n \cdot e^{ch/2} + (e^{ch/2}-1)F(a_n, t_n + h/2)/c \\
c_n &= a_n \cdot e^{ch/2} + (e^{ch/2}-1)(2F(b_n, t_n + h/2) - F(u_n, t_n))/c \\
u_{n+1} &= u_n \cdot e^{ch} + \{F(u_n, t_n)[-4 - hc + e^{ch}(4 - 3hc + h^2c^2)] \\
&\quad + 2(F(a_n, t_n + h/2) + F(b_n, t_n + h/2))[2 + hc + e^{ch}(-2 + hc)] \\
&\quad + F(c_n, t_n + h)[-4 - 3hc - h^2c^2 + e^{ch}(4 - hc)]\}/(h^2c^3)
\end{aligned} \tag{33}$$

Egileek (Cox eta Matthewsek) metodo honek $O(h^5)$ trunkatze errorea duela frogatu dute. Ondorioz, $O(h^4)$ trunkatze errore metatua du, laugarren ordenako metodoa izanez (RK4-ren baliokidea).

RK Butcher taulak

(33) nondik datorren ulertzeko, RK metodoen eskema orokorra azalduko dugu. Azalpen hau Bhatt eta Khaliq egileen lanan oinarritu dugu [11]. Runge-Kutta metodoak s-fase dituzten adierazpenen bidez sortuak dira. Adierazpen hauek hurrengo itxura dute:

$$u_{n+1} = u_n + h\sum_{i=1}^{s} b_i F(k_i, t_n + c_i h) \tag{34}$$

$$k_i = u_n + h\sum_{j=1}^{i-1} a_{ij} F(k_j, t_n + c_i h), \quad i = 1, 2, ...s \tag{35}$$

non koefiziente guztiak hobe antolatzeko, Butcher taulak erabiltzen dira [12]. Runge-Kutta metodo orokor batek 6. taularen eskema jarraitzen du. (33) adierazpenean aurkeztu dugun ETDRK4 metodoaren Butcher eskema 7. taulan dugu.

Denbora esponentzial diferentziazio metodoa garatzeko, faktore integratzailearekin erlazioa duten $\phi_\mu(hc)$ funtzio laguntzaileak erabili dira. Haien adierazpenak:

6. taula: RK metodo orokor baten Butcher taula.

0					
c_2	a_{21}				
c_3	a_{31}	a_{32}			
$\vdots$	$\vdots$		$\ddots$		
c_s	a_{s1}	a_{s2}	$\cdots$	$a_{s,s-1}$	
	b_1	b_2	$\cdots$	b_{s-1}	b_s

7. taula: ETDRK4 metodoaren Butcher taula.

	0				
	$\frac{1}{2}$	$\frac{1}{2}\phi_1$			
	$\frac{1}{2}$	0	$\frac{1}{2}\phi_1$		
1	$\frac{1}{2}$	$\frac{1}{2}\phi_1$	0	0	
	$\frac{1}{2}$	$-\frac{1}{2}\phi_1$	0	ϕ_1	
		$4\phi_3 - 3\phi_2 + \phi_1$	$-4\phi_3 + 2\phi_2$	$-4\phi_3 + 2\phi_2$	$4\phi_3 - \phi_2$

$$\phi_0(hc) = e^{-hc}, \quad \phi_\mu(hc) = (-hc)^{-\mu}\left(e^{-hc} - \sum_{j=0}^{\mu-1}\frac{(-hc)^j}{j!}\right), \ \mu = 1,2,3 \tag{36}$$

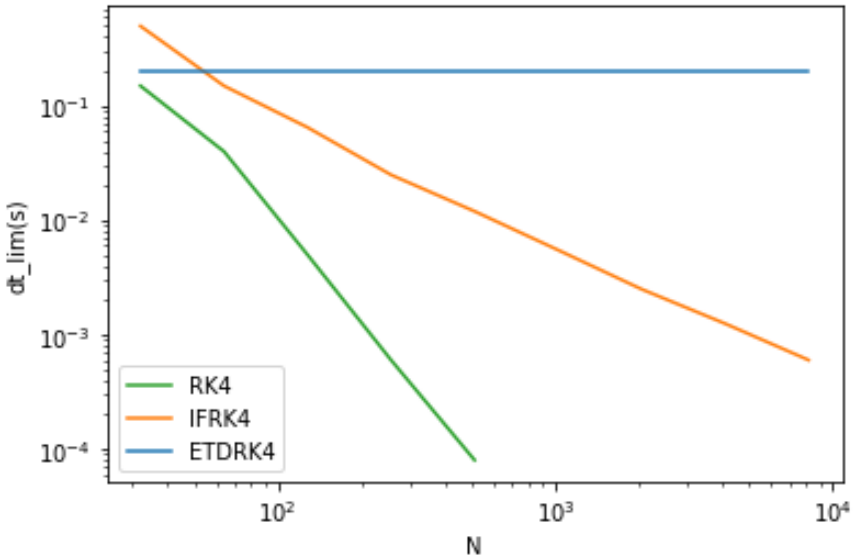
7. irudiaren eskematik Runge-Kutta metodoa sortu dezakegu, (34) eta (35) adierazpenen bidez. ETDRK4-ren eskemak azpi-fase bat du. Bhatt eta Khaliq egileek (37) ekuazioan adierazten dute metodo hau. Definizio hau sinplifikatuz, (36) ordezkatuz, Cox eta Matthewsek proposatutako ETDRK4-ren (33) adierazpenaren baliokidea dela ikusi daiteke.

$$\begin{aligned}
a_n &= \phi_0(hc/2)u_n + h/2 \cdot \phi_1(hc/2)F(u_n, t_n) \\
b_n &= \phi_0(hc/2)u_n + h/2 \cdot \phi_1(hc/2)F(a_n, t_n + h/2) \\
c_n &= \phi_0(hc/2)a_n + h/2 \cdot \phi_1(hc/2)[2F(b_n, t_n + h/2) - F(u_n, t_n)] \\
u_{n+1} &= \phi_0(hc)u_n + h\phi_1(hc)F(u_n, t_n) \\
&\quad + h\phi_2(hc)[-3F(u_n, t_n) + 2F(a_n, t_n + h/2) + 2F(b_n, t_n + h/2) - F(c_n, t_n + h)] \\
&\quad + 4h\phi_3(hc)[F(u_n, t_n) - F(a_n, t_n + h/2) - F(b_n, t_n + h/2) + F(c_n, t_n + h)]
\end{aligned} \tag{37}$$

Egonkortasuna eta zehaztasuna

Metodo honek zehaztasun oso ona du. Izan ere, $\Delta t = 0.2$ *denbora-gehikuntzak limite konstantea du, non N puntu-kopurua zein den ez du inporta.* Orain arte denbora gehikuntza N puntu kopuruarekin handitzen zen, konputazio denbora handituz. Ezaugarri honek abantaila oso inportanteak dakartza, zehaztasun handiko edo dimentsio altuko problemak ebaztekotan.

Emaitza honen inplikazioak ulertzeko, hiru metodoen konparaketa 18. irudian egin dugu (RK4, IFRK4 eta ETDRK4).



18. irudia: RK metodo bakoitzak onartzen duen denbora-gehikuntza limitea, N puntu-kopuru ezberdinetarako.

ETDRK4-ren zehaztasunari begira, IFRK4 metodoak baino zehaztasun hobea du. Zoritxarrez, programaren zehaztasuna oszilakorra bihurtuko da, N txiki eta Δt handiko parametroekin lan egitean. Honek zehaztasun handia lortzea saihesten du.

ETDRK4-B metodo alternatiboa

Aurkitu dugun egonkortasun arazoak konpontzeko, S. Krogstadek proposatutako ETDRK4-B metodoa erabiliko dugu [13]. Metodo honek Cox eta Matthewsek proposatutakoa baino zehaztasun eta egonkortasun ezaugarri hobeak ditu.

Metodo honen Butcher eskema 8. taulan bildu dugu. Izan ere, ETDRK4-ren eskema antzekoa da, non a_n, b_n eta c_n koefiziente ezberdinak hartu dira. Honen definizioan $\phi_2(hc)$ formula erabiliz, laugarren ordenako metodoa lortu da azpi-faserik erabili gabe.

8. taula: ETDRK4-B metodoaren Butcher taula.

0				
$\frac{1}{2}$	$\frac{1}{2}\phi_1$			
$\frac{1}{2}$	$\frac{1}{2}\phi_1 - \phi_2$	ϕ_2		
1	$\phi_1 - 2\phi_2$	0	$2\phi_2$	
	$4\phi_3 - 3\phi_2 + \phi_1$	$-4\phi_3 + 2\phi_2$	$-4\phi_3 + 2\phi_2$	$4\phi_3 - \phi_2$

Konstanteen definizio berria:

$$\begin{aligned} a_n =& \phi_0(hc/2)u_n + h/2 \cdot \phi_1(hc/2)F(u_n, t_n) \\ b_n =& \phi_0(hc/2)u_n + h/2 \cdot \phi_1(hc/2)F(u_n, t_n) + h\phi_2(hc/2)[F(a_n, t_n + h/2) - F(u_n, t_n)] \\ c_n =& \phi_0(hc)u_n + h\phi_1(hc)F(u_n, t_n) + 2h\phi_2(hc)[F(b_n, t_n + h/2) - F(u_n, t_n)] \end{aligned}$$

Metodo hau *Python*en inplementatuta, egonkortasun eta zehaztasun testak egin ditugu. Hasteko, ERTDRK4 metodoak zituen egonkortasunaren oszilazioak ERTDRK4-B metodoak konpondu ditu. Beste ezaugarriei begira, metodo honek denbora esponentzial diferentziazio RK4 metodoa arruntaren oso antzekoa da.

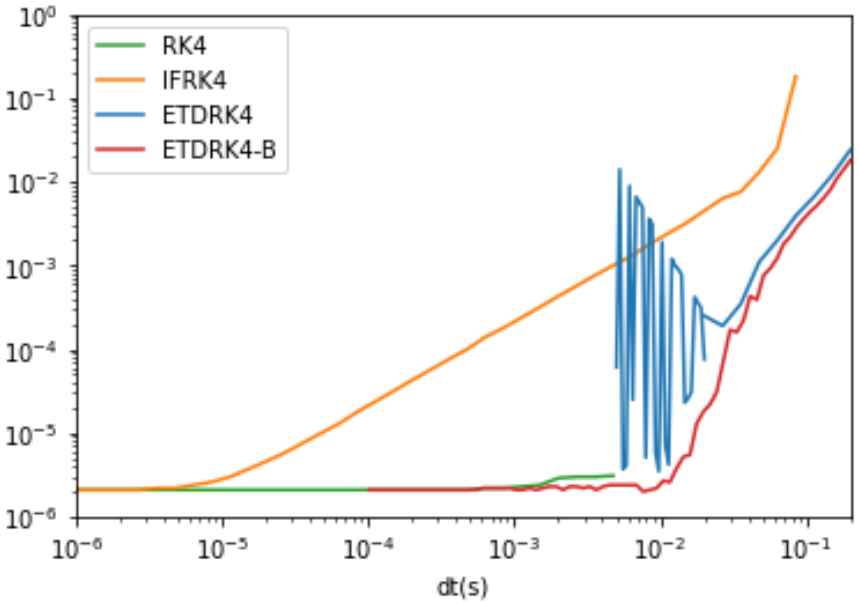
Orain arte inplementatu ditugun laugarren ordenako metodoen zehaztasunak alderatu ditugu. Denbora-gehikuntza parametroaren aurkako zer jokaera duten ikusteko, ikerketa estentsiboa egin dugu, metodo bakoitzaren Δt optimoa zein den ikus dezagun. Emaitzak 19. eta 20. irudietan bildu ditugu, N altu eta baxuko kasuak aztertu baititugu. $N = 128$ puntu gutxiko kasuan, RK4, IFRK4, ETDRK4 eta ETDRK4-B metodoen zehaztasunaren izaera argi ikusi dezakegu.

- RK4 oso zehatza da baina Δt limite txarra du. N puntu gutxiekin, ordea, Δt limitea ez da hain txikia.
- IFRK4 metodoak denbora gehikuntza minimoaren arazoa konpontzen du, baina errore oso altuak ditu. Metodoaren zehaztasun guztia erabiltzeko, Δt oso txikia hartu behar da.
- ETDRK4 metodoak zehaztasun hobea aurkezten du, baina Δt batetik aurrera ezegonkortzen da. N puntu gutxiekin lan egiteko ez da egokia.
- ETDRK4-B metodoak ETDRK4-ren ezegonkortasunak saihesten ditu. Oro har, IFRK4 baino metodo eraginkorragoa da.

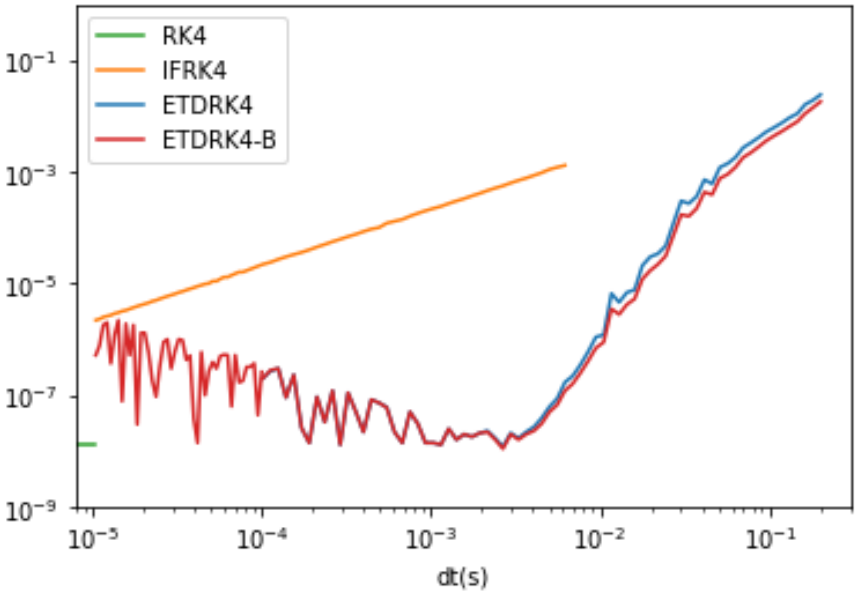
19. irudian ezin dira zehaztasun altuko portaerak alderatu. Izan ere, $N = 128$ puntuekin diskretizazioa ez da ona eta 10^{-6}ren araberako errorea sortzen du. Orain, $N = 1024$ puntuko beste muturreko kasua aztertu dezakegu. Ondorio esanguratsuenak hurrengoak dira:

- RK4-ren denbora gehikuntza limitea txartu da. N altuekin, metodo hau erabiltzeak ez du merezi.
- ETDRK4-ren ezegonkortasunak joan dira, eta ETDRK4-B metodoaren portaera berdina du. Bi metodo hauen trunkatze-errorea azkar nabarmentzen da, $\Delta t = 0.003$ izango da haien denbora gehikuntza optimoa.

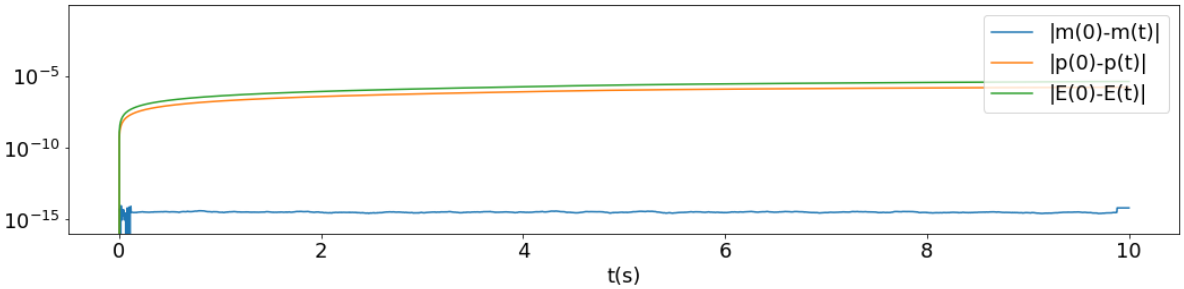
Azkenik, masa, momentu eta energia kontserbazioa aztertu dugu. Aurreko kasuetako baldintza berekin simulazioa egin dugu. 21. irudian bildu ditugu emaitzak. Alde batetik, erroreak handiagoak dira, RK4 eta IFRK4 kasuekin alderatuz. Beste aldetik, ordea, bi solitoien arteko talkak ez du perturbaziorik sortu, ez energian ez inon.



19. irudia: RK metodoei loturiko errorea, Δt denbora-gehikuntzaren funtzioan. $N = 128$ espazioaren diskretizazio baxua erabili da. Erroreak ez dira 10^{-6}tik jaisten diskretizazioarekin erlazioa duten erroreengatik.



20. irudia: RK metodoei loturiko errorea, Δt denbora-gehikuntzaren funtzioan. $N = 1024$ espazioaren diskretizazio altua erabili da.



21. irudia: Magnitude kontserbatuen erroreen denbora garapena $|m(0)-m(t)|$, $|p(0)-p(t)|$ eta $|E(0)-E(t)|$, hurrenez hurren. ETDRK4-B metodoa erabili da. *Datuak:* $N = 256$, $\Delta t = 3 \cdot 10^{-3}$. *(Exekuzio-denbora: 2.7s.)*

6. Ondorioak

Lan honetan, metodo espektralen bidezko PDE-ak ebazpenerako behar diren tresna guztiak azaldu eta era eraginkorrean inplementatu ditugu. 3. atalean ikusi dugun bezala, deribatu partzialezko ekuazioak ebazteko erarik ohikoena espaziozko diskretizazioak metodo espektralekin egitea eta denborazko diskretizazioa diferentzia finituekin egitea da. Zenbakizko ereduan hipotesi hau frogatu dugu, non KdV ekuazioaren magnitude konstanteen kontserbazioa doitasun altuarekin lortu dugun.

Gainera, 4. atalean ikusitako de-aliasing teknikak erabilita, ekuazio ez-linealak ebazterakoan ez dugu arazorik izan. Adibidez, bi solitoien arteko talka ezin da simulatu alias fenomenoa kontuan hartu gabe.

Korteweg de Vries ekuazioa dispertsioa eta ez-linealtasuna nahasten dituen ekuazioa da. Runge-Kutta metodoaren bidezko ebazpenak zehaztasun altua du, baina denboraren gehikuntzarekiko egonkortasuna txarra dela ikusi dugu. Fourierren maiztasun espazioan lan eginda, KdV ekuazioaren gai dispertsiboa adierazpen lineala du. Hortaz, faktore integratzailedun metodoak erabili ditzakegu, hau da, IFRK4 eta ETDRK4 metodoak. Bigarren metodoaren bidez, zehaztasunaren eta denbora gehikuntzarekiko egonkortasunaren arteko konpromiso ona lortu dugu. Azkenik eta metodoaren ezegonkortasunak konpontzeko, proposatutako ETDRK4-B metodo baliokidea erabili dugu, emaitzak are gehiago hobetuz.

Oro har, landutakoan zehar metodo espektralen (eta zenbakizko metodoen) ezagutza hobetu dugu, eta edozein ekuazio diferentzial arrunt edo deribatu partzialezko simulazioa eraikitzeko beharrezkoak zaizkigun tresnak ulertu ditugu.

Erreferentziak

[1] L. Trefethen, *Spectral Methods in MATLAB*. Berlin, Heidelberg: SIAM, 2000.

[2] J. P. Boyd, *Chebyshev and Fourier Spectral Methods*. Mineola, New York 11501: Dover Publications; Second Edition, Revised (5 juin 2013), 2000.

[3] D. Dutykh, "A brief introduction to pseudo-spectral methods: application to diffusion problems," *arXiv:1606.05432, Cornell University*, pp. 167–209, 2019. [Online]. Available: http://arxiv.org/abs/1606.05432

[4] B.-Y. Guo, J. Shen, and L.-L. Wang, "Optimal spectral-galerkin methods using generalized jacobi polynomials," *Journal of Scientific Computing*, vol. 27, no. 1-3, pp. 305–322, 2006. [Online]. Available: https://doi.org/10.1007/s10915-005-9055-7

[5] P. O. J. Scherer, *Computational Physics*. Berlin, Heidelberg: Springer, 2010.

[6] J. W. Cooley and J. W. Tukey, "An algorithm for the machine calculation of complex fourier series," *Mathematics of Computation*, vol. 19, no. 90, pp. 297–301, 1965. [Online]. Available: https://doi.org/10.1090/s0025-5718-1965-0178586-1

[7] P. G. Drazin and R. S. Johnson, *Solitons: An Introduction*, 2nd ed. Cambridge University Press, 1989.

[8] T. I. Belova and A. E. Kudryavtsev, "Solitons and their interactions in classical field theory," *Physics-Uspekhi*, vol. 40, no. 4, pp. 359–386, 1997. [Online]. Available: http://stacks.iop.org/1063-7869/40/i=4/a=R02?key=crossref.184aaf501869f28a53c690b6dec250be

[9] S. Gottlieb, Z. J. Grant, and L. Isherwood, "Strong stability preserving integrating factor runge-kutta methods," 2017. [Online]. Available: https://arxiv.org/abs/1708.02595

[10] S. Cox and P. Matthews, "Exponential Time Differencing for Stiff Systems," *Journal of Computational Physics*, vol. 176, no. 2, pp. 430–455, 2002. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0021999102969950

[11] H. Bhatt and A. Khaliq, "Fourth-order compact schemes for the numerical simulation of coupled Burgers' equation," *Computer Physics Communications*, vol. 200, pp. 117–138, 2016. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S001046551500418X

[12] J. C. Butcher, "Coefficients for the study of Runge-Kutta integration processes," *Journal of the Australian Mathematical Society*, vol. 3, no. 2, pp. 185–201, 1963. [Online]. Available: https://www.cambridge.org/core/product/identifier/S1446788700027932/type/journal_article

[13] S. Krogstad, "Generalized integrating factor methods for stiff PDEs," *Journal of Computational Physics*, vol. 203, no. 1, pp. 72–88, 2005. [Online]. Available: https://linkinghub.elsevier.com/retrieve/pii/S0021999104003183