LATVIJAS UNIVERSITĀTE
FIZIKAS UN MATEMĀTIKAS FAKULTĀTE
FIZIKAS NODAĻA

FEROMAGNĒTISKAS VIENDOMĒNA DAĻIŅAS, AR PLAKNES ANIZOTROPIJU, DINAMIKA PRECESĒJOŠĀ MAGNĒTISKĀ LAUKĀ.

BAKALAURA DARBS

Autors: Oskars Sjomkāns
Studenta apliecības Nr.: os13011
Darba vadītājs: Dr. Phys. Jānis Cīmurs

RĪGA 2017

Anotācija

Šajā darbā tiek pētīta feromagnētiska materiāla viendomēna daļiņa ar plaknes anizotropiju, tās kustība un stabilie režīmi ārējā lauka ietekmē. Tiek apskatīti Stonera – Vohlfarta modeļa pielietojumi un no Stonera – Vohlfarta modeļa izvests diferenciālvienādojums, kas tiek skaitliki risināts, izmantojot Runges – Kutta 4. kārtas metodi. Secinājumi par modeļa pielietojamību feromagnētiskām daļiņām tiek izdarīti, analizējot atrisinājumu stabilitāti un aproksimējot fāzu diagrammas no iegūtajiem rezultātiem.

Atslēgvārdi: Anizotropija, Feromagnētisms, Paramagnētisms, Diamagnētisms, Simulācija, Skaitliski

Abstract

This thesis looks at a single-domain ferromagnetic particle with plane anisotropy, it`s motion and it`s stable regimes in the presence of an external field. Applications of the Stoner – Wohlfarth model and numerically found solutions using Runge – Kutta 4th order method of a differential equation derived from the Stoner – Wohlfarth model are examined. Conclusions about the applicability of the model for ferromagnetic particles are drawn by analysing the stability of the solutions and by approximating phase portraits from the results.

Keywords: Anisotropy, Ferromagnetism, Paramagnetism, Diamagnetism, Simulation, Numerically

Saturs

APZĪMĒJUMU SARAKSTS

χ - Materiāla magnētiskā susceptibilitāte.

μ_0 - Vakuuma magnētiskās caurlaidības koeficients.

μ - Konkrēta materiāla magnētiskās caurlaidības koeficients.

$\vec{M}$ - Magnētiskais moments

$\vec{e}$ - Magnētiskā dipola momenta virziena vienības vektors.

$\vec{n}$ - Magnētiskās anizotropijas ass virziena vienības vektors.

$\vec{H}$ - Magnētiskā lauka vektors.

$\vec{B}$ - Magnētiskā lauka indukcijas vektors

$\vec{h}$ - Magnētiska lauka virziena vienības vektors.

$\vec{\tau}$ - Mehāniskais spēka moments.

ξ - Viskozās berzes koeficients.

γ - Žiromagnētiskā attiecība.

H_a - Anizotropijas magnētiskais lauks

IEVADS

Šobrīd aktuāla tēma ir magnētiskās Janus daļiņas, kas sastāv no divām daļām, kur viena puse ir magnētiska, bet otra puse ir nemagnētiska (diamagnētiska). Homogēnā laukā tās veido zig-zag-veida ķēdītes [1].Precesējošā laukā tās var veidot dažādu izmēru caurulītes [2].

Šo formu veidošanās dod iespēju no Janus daļiņām veidot mikropeldētājus [3]. Ar magnētiskā lauka palīdzību ir iespējams dinamiski mainīt šīs daļiņu struktūras [4], tādā veidā ārēji regulējot procesus mikro un nano mērogā, piemēram, regulēt dažādu ķīmisko vai bioloģisko reakciju ātrumu, mainot materiāla porainību ar magnētiskā lauka palīdzību.

Šobrīd magnētiskās Janus daļiņas tiek modelētas pieņemot, ka magnētiskā daļa ir bez paliekošās magnetizācijas (ir paramagnētiska) [5] [6], kaut gan eksperimentāli novērojumi rāda, ka paliekošā magnetizācija Janus daļiņai ir novērojama [7]. Vienkāršākais modelis, kas apvieno, gan paliekošo magnetizāciju pie maziem laukiem, gan paramagnētiskās īpašības pie lieliem laukiem, ir anizotropas viendomēna daļiņas modelis. Šajā bakalaura darbā ir paredzēts pārbaudīt, kādu ietekmi uz kustību atstāj paliekošā magnetizācija, ja šai magnetizācijai ir plaknes anizotropija, kas vislabāk atbilst Janus daļiņas gadījumam.

Darba mērķis ir noskaidrot lietotā modeļa, kas faktiski ir paredzēts paramagnētisku daļiņu pētīšanai, pielietojamību feromagnētiskām daļiņām.

Darba uzdevumi:

1. Izveidot rīku, kas atrisina daļiņas kustību aprakstošo diferenciālvienādojumu pie fiksētām parametru vērtībām.
2. Konstruēt feromagnētiskās daļiņas ar plaknes anizotropiju fāzu diagrammu, attēlojot metstabilos, stabilos un nestabilos reģionus.
3. Salīdzināt feromagnētiskās daļiņas ar plaknes anizotropijas fāzu diagrammu pie lieliem ārējiem laukiem ar paramagnētiskas daļiņas fāzu diagrammu.

1. TEORĒTISKĀ DAĻA

1.1 FEROMAGNĒTISMS, PARAMAGNĒTISMS UN DIAMAGNĒTISMS

1.1.1 Magnētisms, parādības klasifikācija

Ir vispārzināms empīrisks fakts, ka visus materiālus var iedalīt trīs grupās pēc to mijiedarbības ar magnētisku lauku. Šīs trīs materiālu grupas ir diamagnētiķi, paramagnētiķi un feromagnētiķi.

Diamagnētiskiem materiāliem nepiemīt magnētiskas īpašības bez pielikta arējā lauka, bet uz ārējo laukudiamagnētiķi reaģē materiāla summārajam magnētiskā dipola momentam uz tilpuma vienību pieaugot pretēji pieliktā lauka virzienam, proporcionāli pieliktajam lauka intensitātei.

Paramagnētiskiem materiāliem arī nepiemīt magnētiskas īpašības, ja nav ārējā lauka, bet ja tāds ir, tad magnētiskais dipola moments uz tilpuma vienību materiālā palielinās pieliktā lauka virzienā.

Šīm divām parādībām mijiedarbību ar kādu ārējo magnētisko lauku vienkāršotā veidā var aprakstīt ar vienu konstanti – materiāla magnētisko uzņēmībuχ, kas raksturo kā magnētiskais dipola moments $\vec{M}$ uz tilpuma vienību ir saistīts ar magnētisko lauku.

$$\vec{M} = \chi\vec{H} \tag{1}$$

No šiem apsvērumiem, var secināt, ka diamagnētiskiem materiāliem uzņēmībai χ ir jābūt negatīvai, bet paramagnētiskiem materiāliem uzņēmībai ir jābūt pozitīvai. [1]

Izmantojot magnētiskās caurlaidības apzīmējumus, ir iespējams uzrakstīt sakarību starp magnētiskā lauka indukciju un magnētisko lauku vielā,izmantojot uzņēmību un vakuuma magnētisko caurlaidību μ_0:

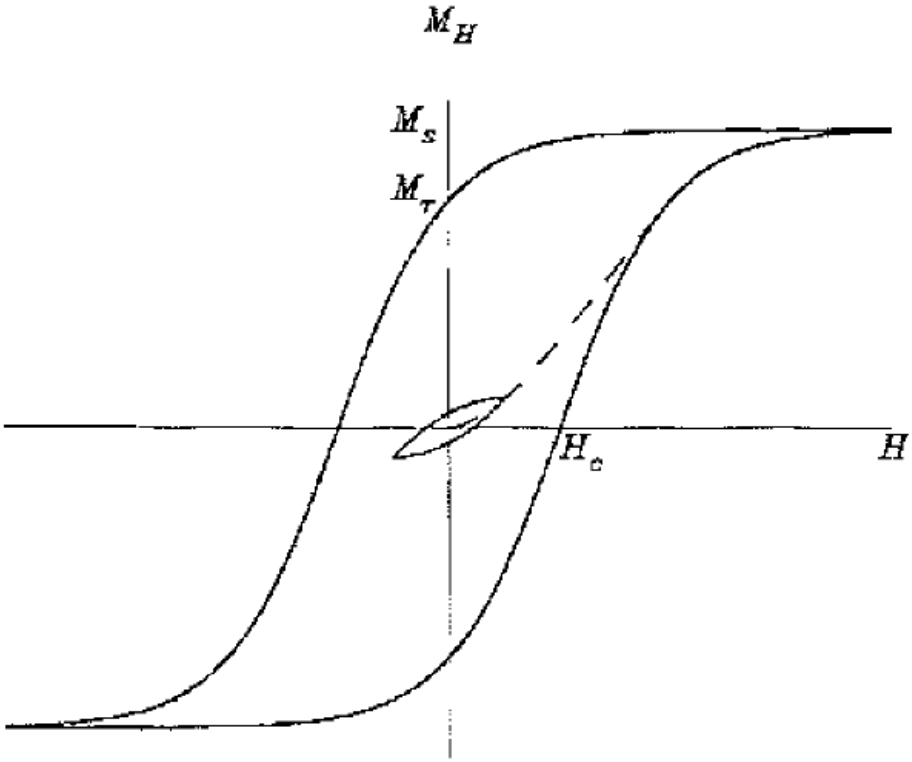
$$\vec{B} = \mu\vec{H} \tag{2}$$

Kur:

$$\mu = \mu_0(1 + \chi) \tag{3}$$

μ_0 - vakuuma magnētiskās caurlaidības koeficients, μ – Konkrēta materiāla magnētiskās caurlaidības koeficients. Šāds vienkāršots modelis gan neiekļauj reāli eksistējošus materiālus, kuriem piemītnenulles summārais magnētiskā moments uz tilpuma vienību t.i. *magnetizācija* arī pie neesoša (vai praktiski neesoša) ārējā magnētiskā lauka, kā arī faktu, ka šī magnetizācija nepakļaujas vienkāršai lineārai sakarībai atkarībā no ārējā lauka.

Feromagnētiskos materiālos koeficients μ ir atkarīgs gan no tam pieliktā lauka, gan arī materiālam piemītošās magnetizācijas un pieliktā lauka „vēstures". Viens šādas materiāla uzvedības piemērs ir magnētiskās histerēzes cilpa.



Attēls 1.1.1: Tipiska feromagnētiska materiāla magnētiskās histerēzes cilpa. Pie viena un tā paša pieliktā lauka ir iespējamas vairākas magnetizācijas vērtības. Grafikā attēlota arī magnētiskā remnance M_τ, koercivitāte H_c un mazā histerēzes cilpa. [2]

Darbā arī tiek lietoti jēdzieni kā piesātinājuma magnetizācija un spontānā magnetizācija. Piesātinājuma magnetizācija ir maksimālā magnetizācija, kas piemīt feromagnētiskam materiālam un nemainās, ja tiek palielināt ārējā magnētiskā lauka intensitāte.Spontānā magnetizācija, saukta arī par paliekošo magnetizāciju un remnanci, ir magnētiskais moments vielā, kas saglabājās pēc tam, kad lauks ir noņemts.

Feromagnētisma kontekstā jāpiemin,ka eksistē zināma magnetizācijas atkarība no temperatūras, kā arī eksperimentāls fakts, ka feromagnētiķi virs materiālam raksturīgās Kirī temperatūras T_c uzvedas kā paramagnētiķi. Tālāk iedziļināties tematā par atkarību no temperatūras darba kontekstā nav vērts, jo tiks pētīts teorētiskais modelis viena domēna daļiņas dinamikai temperatūrā, kas ir ievērojami zemāka par Kirī temperatūru, kā arī atkarība no temperatūras ir vairāku domēnu efekts, kā tiek aprakstīts nodaļā 1.2.

Šajā darbā paramagnētisma, superparamagnētisma un feromagnētisma parādības tiks apskatītas lielākoties kvaziklasiski, izmantojot klasiskās fizikas magnētisma modeļus, bet ieviešot spina jēdzienu.[11]

1.1.2 Domēnu teorija

Viens veids kā apskatīt feromagnētisma parādības ir izmantot domēnu teoriju. Domēns ir feromagnētiska materiāla reģions, kurā ir vienmērīga magnetizācija t.i. visi individuālie magnētiskie momenti ir sakārtoti vienā virzienā

Vispārīgā feromagnētisma teorijā ir jāpieņem, ka elektrona spins t.i. elektronam piemītošais kvantētais impulsa moments ir atbildīgs par paliekošo magnetizāciju. Turpmāk šajā nodaļā tiks apskatīta spinu mijiedarbība no kvantu mehānikas skatupunkta un izmantoti

arī klasiskie elektromagnētisma modeļi, lai izveidotu priekšstatu par magnētismu materiālos un to pielietotu, lai pētītu interesējošās feromagnētiskās daļiņas.

No kvantu mehānikas ir nepieciešams zināt dažus faktus par elektronu. Elektrona kvantētā leņķiskā momenta spina vērtība ir :

$$s = \pm\frac{1}{2} \tag{4}$$

$\pm$ Zīme šeit apzīmē to, ka spins var būt vērsts vai nu „uz augšu" vai „uz leju" relatīvi pret kādu izvēlētu koordinātu asi pret kuru spinu novēro.

Elektrons pārnes pēc moduļa:

$$M_e = \frac{q_e\hbar}{2m_e} \tag{5}$$

lielu magnētisko momentu, kas arī ir pazīstams ar nosaukumu Bora magnetons un šajā darbā turpmāk tiks apzīmēts arM_B.

Ir zināms, ka magnētiska dipola orientācijas enerģija ārējā laukā ir:

$$E = \vec{M} \cdot \vec{B} \tag{6}$$

Lai kādā materiālā aprakstītu magnētiskā lauka enerģiju elektrona spina dēļ, intuitīvā pieeja būtu enerģijas izteiksmē M vietā būtu ievietot Bora magnetonuunto sareizināt arspinu un summēt pa visiem nesapārotajiem valences elektroniem, ņemot vērā vērsumus. Situācija ar elektroniem orbitālēs gan nav tik vienkārša, jo elektrona spina orientācija ir atkarīga arī no apkārtējo spinu orientācijas, piemēram dzelzs atomā elektroniem ir spēcīga tendence sakārtoties ar spiniem „augšup", ja tam blakus esošam dzelzs atomam elektronu spini arī ir sakārtojušies „augšup". Tāpēc arī dzelzs, kā arī daži citi tā periodiskās elementu sistēmas „kaimiņi" kā kobalts un niķelis ir izteikti feromagnētiski. Tiem ir tendence sakārot spinus vienā virzienā. [12]

Ir novērojams eksperimentāls fakts, ka tuvu esoši elektroni ļoti spēcīgi cenšas noorientēt savus spinus pretēji viens otram ar šķietamu spēku. Šo šķietamo „apmaiņas spēku" var skaidrot ar Paulī izslēgšanas principu. Ar šo efektu konkurē tas, ka elektrona spiniem ir tendence sakārtoties vienā virzienā ar vidējo magnetizāciju, kāda piemīt tuvākajai tā apkārtnei. Reālā materiālā jāņem vērā arī, ka spinu orientācijā lomu spēlē arī temperatūra. Kā jau minēts nodaļā 1.1, feromagnētiski materiāli virs kādas kritiskās temperatūras kļūst paramagnētiski. Tagad to ir iespējams skaidrot ar to, ka sakārtotībai raksturīgo vidējo magnetizāciju elektrona apkārtnē spēj izjaukt termiski procesi.

Lai runātu par domēniem, ir jārunā par magnētiskā lauka enerģiju uz tilpuma vienību. Ja, piemēram, mēs pieņemtu pilnībā spontāni magnetizētu feromagnētiska materiāla kubu, tam piemistu liela magnētiskā lauka enerģija un ar to asociētais magnētiskais lauks. Enerģija un magnētiskais lauks, ko šis kubs radītu, būtu mazāki, ja viena puse kuba būtu magnetizēta pretēji otrai. Lai uz tilpuma vienību tiktu minimizēta magnētiskā lauka enerģija, spinu vektoriālajai summai uz tilpuma vienību būtu aptuveni jābūt nullei. Domēnu veidošanos nosaka apmaiņas mijiedarbība.[12]

Uzdevums atrast magnētiskās enerģijas saistību ar veidojošos domēnu izmēriem ir ārkārtīgi netriviāls un kvantitatīvs tā apraksts neatbilst bakalaura darba apjomam un sarežģītībai. Domēnu raksturīgie parametri kā izmērs un magnētiskā enerģija ir atrodami literatūras avotos.

Viendomēna daļiņai atkarībā no tās magnetizācijas struktūras, ir iespējami dažādi izmēri pie kuriem ir enerģētiski visizdevīgāk būt viena domēna konfigurācijā.Vienkāršākais

gadījums ir sfēriska daļiņa. Rakstā izmantoto modeļu ietvaros, ir nepieciešams ieviest daļiņas rādiusa apakšējo robežu, lai pierādītu, ka aizvien sarežģītāku daļiņu konfigurāciju meklēšana, lai novērtētu to enerģijas, novedīs pie galīga daļiņas rādiusa. Izteikts CGS mērvienībās, šis rādiuss ir izsakāms kā:

$$R_{c0} = \frac{1.017\sqrt{C}}{M_s} \tag{7}$$

Kur C ir apmaiņas konstante, bet M_s ir piesātinājuma magnetizācija.Šis ir rādiuss, zem kura visizdevīgākā konfigurācija ir vienā domēnā, eksistē arī augšējā robeža bet eksistē *divas* šīs robežas. CGS sistēmā:

$$R_{c1} = \frac{2.2646\frac{\sqrt{C}}{\pi M_s^2}}{\sqrt{1 - \frac{1.4038|K_1|}{\pi M_s^2}}} \tag{8}$$

$$R_{c2} = \frac{9\sqrt{C(|K_1| + 8\pi\sigma M_s^2)}}{8(3\sigma - 2)M_s^2} \tag{9}$$

Kur K_1 ir pirmās kārtas magnetokristāliskāsanizotropijas konstante un σ ir parametrs, kura vērtība ir $\sigma = 0{,}785398$. Definīcijas ir divas, jo pie dažādām parametru kombinācijām viena var neatgriezt fizikāli interpretējamas vērtības. Dažreiz viena vai otra tiek pilnībā ignorēta. Variējot parametrus, par vadlīnijām ņemot zināmās konstantes reāliem materiāliem, iegūst, ka viendomēna daļiņu raksturīgie izmēri ir no 5 nm līdz 30 nm. Daļiņas izmēra atrašanās starp apakšējo un augšējo robežu gan nav garantija tam, ka daļiņa tiešām atradīsies šajā stāvoklī, jo stāvokļa eksistence nenozīmē to, ka tas tiks ieņemts. Sistēma var būt "iestrēgusi" augstākas enerģijas stāvoklī feromagnētiskās histerēzes dēļ.[13]

Efekts, kad eksistē kāds stāvoklis, kad enerģija ir „iestrēgusi" darbā netiek apskatīts. Šeit ar augšējo un apakšējo rādiusu ir domāti kritiskie rādiusi. Apakšējā robeža ir rādiuss zem kura pilnīgi noteikti daļiņa būs viendomēna.

1.1.3 Magnetokristāliskā anizotropija

Par magnetokristālisko anizotropiju sauc parādību, kuras dēļ magnētisku daļiņu var būt vieglāk magnetizēt dažos virzienos nekā citos, kur šeit ar „vieglumu" ir saprasts ārējais lauks, kas nepieciešams, lai sasniegtu piesātinājuma magnetizāciju. Virzienus, kuros ir visvieglāk ir magnetizēt daļiņu sauc par vieglajām asīm. Daļiņai var būt arī vairākas vieglās asis. Pretstats vieglajām asīm ir cietās asis, kuras ir tie virzieni, kuros daļiņu magnetizēt ir visgrūtāk.

Magnetokristāliskā anizotropija ir pamatojama ar spinu-orbītu mijiedarbību. Elektronu orbītas ir saistītas ar materiāla kristālisko struktūru un to mijiedarbība ar spiniem var padarīt magnetizācijas vērsumu gar kristalogrāfisko asi izdevīgāku. Lai gan magnetizācijas magnitūdu nosaka gandrīz tikai un vienīgi apmaiņas mijiedarbība, tās virzienu nosaka gandrīz tikai un vienīgi anizotropija, jo apmaiņas enerģija ir neatkarīga no telpas virziena, kurā feromagnētiskais materiāls ir magnetizēts (Heizenberga Hamiltoniānis ir izotropisks). [14] Tieši šī īpašība atšķir feromagnētiķus no paramagnētiķiem.

1.1.4 Formas anizotropija

Šajā darbā koncentrēšanās ir uz daļiņu, kurai ir plaknes anizotropija, kas nozīmē, ka daļiņas formai ir jābūt plakanam elipsoīdam. Ar daļiņas formu ir arī saistīts specifisks anizotropijas veids, ko sauc par formas anizotropiju

Tipiski lauks feromagnētiskos ķermeņos nav viendabīgs, bet eksistē teorēma, kura apgalvo, ka feromagnētiskā ķermenī lauks būs viendabīgs tad un tikai tad, ja šī ķermeņa virsma ir otrās kārtas. To parasti attiecina uz elipsoīdiem, jo visas citas otrās kārtas virsmas nevar aptvert galīga tilpuma ķermeni. [15]

Pēc būtības, formas anizotropijas izpausmes ne ar ko neatšķiras no magnetokristaliskās anizotropijas izpausmēm: Ir cietās un vieglās asis, kā arī ar anizotropiju saistīta enerģija, kurai magnetizācijas magnitūdā ir maza loma, bet spēcīga ietekme uz magnetizācijas virzienu.

Vispārīgi, anizotropijas enerģijai neeksistē vienkārša analītiska izteiksme, kas ir uzrakstāma kā funkcija no virziena, bet fizikālajai situācijai ir iespējams uzrakstīt lineāru tuvinājumu, uzrakstot enerģijas izvirzījumu rindā pēc lenķa pret kādu asi – anizotropijas asi. Ja daļiņai piemīt anizotropija (to ir iespējams paliekoši magnetizēt), tad to ievietojot magnētiskajā laukā,tā magnetizējas un tai parādās magnētiskais moments. Ja daļiņu ievieto magnētiskajā laukā, tai piemīt energija:

$$E = \vec{M} \cdot \vec{B} \tag{10}$$

Šeit var ieviest definīciju $\vec{M} = M\vec{e}$, kura būs nepieciešama vēlāk. Izvirzījumu rindā ir iespējams uzrakstīt ar virziena leņķu kosinusiem starp daļiņas anizotropijas asi un magnētisko momentu, iegūstot [16]:

$$E_i = -\frac{KV}{2}\cos^{2i}\theta \tag{11}$$

Izvirzījumā var atstāt vairākus locekļus, bet jebkurai praktiskai vai teorētiskai nepieciešamībai, pirmais loceklis ir pietiekoši labs tuvinājums. Vispārīgā gadījumā, formas anizotropija elipsoīda gadījumā ir aprakstāma ar izteiksmi[17]:

$$E_M = \frac{1}{2}V\left(N_x M_x^2 + N_y M_y^2 + N_z M_z^2\right) \tag{12}$$

V ir elipsoīda(plakanās daļiņas) tilpums $\frac{4}{3}\pi abc$, kur a, b, c ir elipsoīda pusasis, N ir demagnetizācijas konstantes un M piesātinājuma magnetizācija attiecīgajā virzienā. Demagnetizācijas konstantēm saspiestam elipsoīdam eksistē analītiskas izteiksmes. Tās ir aprēķināmas, izmantojot pusasu attiecības [18]:

$$N_x = \frac{4\pi}{\epsilon^2}\left[1 - \frac{\sqrt{1-\epsilon^2}}{\epsilon}\arcsin(\epsilon)\right] \quad (13)$$

$$N_y = N_z = \frac{4\pi - N_x}{2} \quad (14)$$

Kur ir spēkā izteiksme:

$$\epsilon = \sqrt{1 - \frac{a^2}{b^2}} \quad (15)$$

Vienkāršojot un apvienojot izteiksmes 13 un 14, lai uzrakstītu tās tādā pašā veidā kā vienādojumu priekš E_i , iegūst:

$$K_{formas} = -4\pi M_s^2 \quad (16)$$

Un izmantojot šo apzīmējumu, iegūst:

$$E_{anizotropijas} = \frac{4\pi M_s^2 V}{2}\cos^2\theta \quad (17)$$

Redzams, ka plakanai daļiņai anizotropijas konstante nomaina enerģijas izteiksmes zīmi. Šis ir svarīgs rezultāts, kam būs nozīme arī skaitliskās simulācijas enerģijas aprēķinā. Anizotropijas konstantes vērtības ir atrodamas gan skaitliski, gan analītiski, bet piesātinājuma magnetizācijas vērtības ir atkarīgas no materiāla, no kura sastāv daļiņa,un to analītiska aprēķināšana vai pat novērtēšana ir ārpus bakalaura darba sarežģītības robežām. Tās, tiešā veidā darba galvenajam mērķim – kvalitatīviem feromagnētiskas daļiņas dinamikas pētījumiem, nav nepieciešamas, jo ir izsakāmas ar citiem parametriem, kas tiek lietoti šajā darbā, tāpēc netiks rūpīgi apskatītas, bet tām ir vērtīgi zināt vismaz aptuveno vērtību, fizikālās situācijas priekšstatam. Tās var atrast literatūras avotos, tāpat kā dažus ar tām saistītos darbam nozīmīgos lielumus (skat. tabulu 1.1.4):[13]

	Lielums	Apzīmējums	Vērtība (vai aptuvenā vērtība)	Mērvienība
Dzelzs:				
	Piesātinājuma magnetizācija	Ms	1700	(emu)
	Apmaiņas konstante	C	0,000002	(erg/cm)
	Anizotropijas konstante	K	450000	(erg/cm^3)
	Kritiskais rādiuss	Rc	8.46 - 11.0	(nm)
Ni-Co sakausējums:				
	Piesātinājuma magnetizācija	Ms	638	(emu)
	Apmaiņas konstante	C	0,0000016	(erg/cm)
	Anizotropijas konstante	K	4000	(erg/cm^3)
	Kritiskais rādiuss	Rc	20.0 - 25.0	(nm)

Tabula 1.1.4: Sakarība starp nozīmīgiem parametriem dažiem reāliem feromagnētiskiem materiāliem.

1.1.5 Laika efekti

Magnētiskajai daļiņai ir iespējami vairāki izdevīgākie anizotropijas enerģijas vērsumi,vienā no kuriem vienmēr būs vērsts magnētiskais moments. Kā jau minēts nodaļā 1.1.1 , feromagnēti pie pietiekoši augstas temperatūras kļūst par paramagnētiem. Tas notiek, jo termiskā enerģija spēj „pārsviest" magnētisko momentu no viena minimuma uz citu un anizotropijas enerģija vairs lomu nespēlē.. Pieņemot, ka pie mazākas enerģijas starpības šie lēcieni notiek bieži un Izmantojot Bolcmaņa sadalījumu, ir iespējams atrast raksturīgo laiku τ_0, kurā, vidēji, notiek lēcieni starp enerģijas minimumiem:

$$\tau = \tau_0 \exp\left(\frac{\Delta E}{k_B T}\right) \tag{18}$$

ΔE ir atšķirīgs lielums dažādiem feromagnētiskiem materiāliem,jo tiem ir dažādas enerģijas. Tipiski feromagnētiskiem materiāliem anizotropijas konstante ir aptuveni ar kārtu $K = 10^5 \left(\frac{erg}{cm^3}\right)$ ar kuru ir saistīta kāda raksturīgā enerģija un kritiskais rādiuss:[19]

$$R_c \approx 11\ (nm).$$

Ar laika efektiem tiek saprasts tieši lēcienu biežums. Ja tas ir mazs, laika efekti tiek uzskatīti par vērā neņemamiem. Simulācijā tas nozīmē, ka minimumus spēj ietekmēt tikai ārējais lauks un enerģijas minimizācija nav nepieciešams ņemt vērā temperatūru.

1.2 DAĻIŅAS DINAMIKA MAGNĒTISKAJĀ LAUKĀ

1.2.1 Sakarība starp magnētisko enerģiju un mehānisko spēka momentu

Lai apskatītu daļiņas kustību, kas notiek mijiedarbības starp magnētisko lauku un anizotropijas dēļ, ir jāapskata kā saistās mehāniskais moments un enerģija rotācijas kustības gadījumā. Vispārīgā gadījumā, ir uzrakstāma šāda vispārzināma izteiksme darbam A, kur $\vec{F}$ ir spēka vektors un $d\vec{x}$ ir pārvietojuma diferenciālis:

$$dA = \vec{F} \cdot d\vec{x} \tag{19}$$

Pārvietojuma diferenciāli $d\vec{x}$ var uzrakstīt šādi, kā :

$$d\vec{x} = d\vec{\alpha} \times \vec{r} \tag{20}$$

Apvienojot divas iepriekšējās izteiksmes, iegūst:

$$dA = \vec{F} \cdot (d\vec{\alpha} \times \vec{r}) \tag{21}$$

Šis ir algebrā pazīstams kā trīskāršais jeb jauktais reizinājums, kuram viena no īpašībām ļauj pārkārtot reizinājuma locekļu reizināšanas secību pēc konkrēta likuma, nemainot vienādības patiesumu.

$$dA = \vec{r} \times \vec{F} \cdot d\vec{\alpha} \tag{22}$$

Kur $\vec{r} \times \vec{F} = \vec{\tau}$ ir spēka momenta definīcija. Tāpēc darba diferenciālis ir uzrakstāms šādi:

$$dA = \vec{\tau} \cdot d\vec{\alpha} \tag{23}$$

Ja nepieciešams izteikt padarīto darbu, var integrēt abas vienādojuma puses iegūstot:

$$A = \int_{\alpha_1}^{\alpha_2} \vec{\tau} \cdot d\vec{\alpha} \tag{24}$$

Bet svarīgāks rezultāts par šo ir tas, ka tagad ir iespējams nodefinēt spēka momentu caur enerģiju, kas magnētiskas viendomēna daļiņas gadījumā var būt atkarīga no virziena.

$$\tau = \frac{dE}{d\alpha} \tag{25}$$

Kad magnētiskais dipols tiek ievietots magnētiskajā laukā, uz to darbojas spēka moments, kas ir vērsts tā, lai dipolu novietotu pretēji lauka vērsumam. Jau ir zināma magnētiskās enerģijas izteiksme, kas izmantota nodaļā 1.1.2 **:**

$$E = -\vec{M} \cdot \vec{B} \tag{26}$$

Ir arī zināma iegūtā izteiksme spēka momenta sasaistei ar enerģiju, atkarībā no vērsuma.

$$\tau = \frac{dE}{d\alpha} \tag{27}$$

Ja starp $\vec{M}$ un $\vec{B}$ vektoriem ir kāds leņķis α, dipola enerģiju ir iespējams atvasināt pēc leņķa, iegūstot :

$$\frac{dE}{d\alpha} = \frac{d\left(-|\vec{M}||\vec{B}|cos\alpha\right)}{d\alpha} = \vec{\tau} = M \cdot B \cdot sin\alpha \tag{28}$$

Šī izteiksme atbilst augstāk izmantotajai sakarībai starp spēka momentu, magnētisko momentu un magnētisko lauku.

$$\vec{\tau} = \vec{M} \times \vec{B} \tag{29}$$

1.3 STONERA – VOHLFARTA MODELIS

Modelis, kas kvalitatīvi apraksta viendomēna daļiņas enerģiju, kurai piemīt anizotropija ir Stonera - Vohlfarta modelis. Lai Stonera – Vohlfarta modelis pareizi aprakstītu fizikālu situāciju, ir jāizdara pieņēmumi:[20]:

- Daļiņa ir pietiekoši liela, lai magnetizācijas laika efektus varētu neņemt vērā,
- Daļiņa ir pietiekoši maza, lai apmaiņas enerģija spinus noturētu vērstus vienā virzienā,
- Apmaiņas enerģija tiek pieņemta kā konstanta.

Stonera – Vohlfarta modeļa pamatā ir enerģijas izteiksme, kas apvieno anizotropijas enerģiju un magnētiskā $\vec{M}$ momenta enerģiju, tam mijiedarbojoties ar ārējo lauku. (skat. Attēlu 1.4.1)

$$E = E_{anizotropijas} + E_{lauka} \tag{30}$$

$$E = -\frac{K_{formas}V}{2}\cos^2\theta - \mu H cos\phi \tag{31}$$

Pieliktā ārējā laukā, daļiņas magnētiskais moments vienmēr centīsies noorientēties lauka virzienā, bet pašas daļiņas paliekošā magnetizācija (anizotropijas efekts) dos pienesumu magnētiskā momenta virzienam, kas situāciju sarežģī.

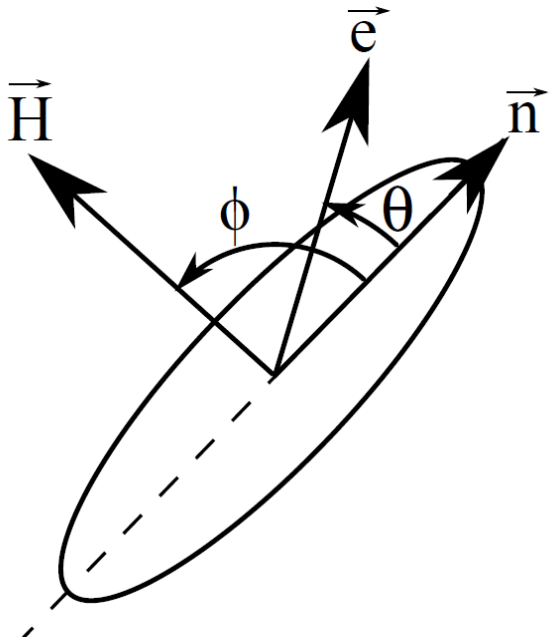
1.3.1 Stoksa tuvinājums

Darbā tiek izdarīts pieņēmums, ka pētāmā atrodas daļiņa vidē, kurā ir liela viskozitāte, tāpēc tiek pieņemts, ka berzes spēka moments nonāk līdzsvarā ar magnētiskā spēka momentu bezgalīgi ātri un inerces efekti ir verā neņemami, visa kustība norisinās ar konstantu ātrumu. Šāda situācija labi atbilstu reālam feromagnētisko daļiņu pielietojumam, kā aprakstīts ievadā.

1.4 MODELĒJAMĀ PROBLĒMA

Aprakstot daļiņas kustību precesējošā laukā, var apskatīt anizotropijas ass vērsumu atkarībā no laika. Izmantojot pieņēmumu, ka ārējais lauks nepārvieto daļiņas masas centru, neņemot vērā magnetostrikciju, termisko efektu ietekmi uz kustību, pieņemot, ka materiāla T_c nav pārsniegta un ņemot vērā Stoksa tuvinājumu, no Stonera – Vohlfarta modeļa ir iespējams konstruēt diferenciālvienādojumu:[21]

$$\frac{d\vec{n}}{dt} = \frac{\omega_a(\vec{e}\cdot\vec{n})^2}{\vec{n}\cdot\vec{h} + \frac{H_a}{H}\vec{e}\cdot\vec{n}}\left[\vec{h} - \vec{n}(\vec{n}\cdot\vec{h})\right] \tag{32}$$



Attēls 1.4.1: Magnētiskā lauka vektors, magnētiskā momenta un anizotropijas ass vērsuma vektori un leņķi starp tiem.

Šeit ω_a ir anizotropijas frekvence, H_a ir anizotropijas lauks, $\vec{e}$ ir magnētiskais moments.

Pirmā šī diferenciālvienādojuma daļa, vienkāršības labad, tiks apzīmēta:

$$\frac{\omega_a(\vec{e}\cdot\vec{n})^2}{\vec{n}\cdot\vec{h}+\frac{H_a}{H}\vec{e}\cdot\vec{n}} = f\left(\phi,\frac{H_a}{H}\right) \tag{33}$$

Tas, kāpēc ir izvelēts tieši šāds apzīmējums, tiks paskaidrots pie skaitliskās implementācijas nodaļā 2.2. Konstantes ω_a un H_a saistās ar Stonera – Vohlfarta modeļa parametriem[22]:

$$\omega_a = \frac{KV}{\xi} \tag{34}$$

$$H_a = \frac{KV}{M} \tag{35}$$

Šeit šķidruma viskozitāte, kurā daļiņa tiek modelēta, nosaka to, ka daļiņas pagriešanās ātrums būs apgriezti proporcionāls viskozās berzes koeficientam.

Darbam centrālā problēma, kas jāatrisina, ir magnētiskā momenta vērsuma vektora $\vec{e}$ atkarības no laika atrašana. Lai gan diferenciālvienādojums apraksta anizotropijas ass vērsuma vektoru izmaiņu laikā, anizotropijas ass pati par sevi nemijiedarbojas ar magnētisko lauku, to dara magnētiskais moments. Anizotropijas ass ir izvēlēta ass, kas ir piesaistīta daļiņas novietojumam, faktiski aprakstot daļiņas dinamiku.

Modeļa galvenā ideja ir tāda, ka daļiņas vērsuma izmaiņa laikā ir atkarīga no pieliktā lauka un daļiņai piemītošā magnētiskā momenta. Magnētiskais moments, savukārt, diskrēti izmaina virzienu atkarībā no pieliktā lauka anizotropijas efektu dēļ un tad cenšas noorientēties magnētiskā lauka virzienā.

Plaknes anizotropija paredz, ka ir jāmeklē anizotropijas enerģijas ekstrēmi, no kuriem vismazākajai enerģijas vērtībai atbilstošajā virzienā būs pavērsts magnētiskā lauka momenta vērsuma vektors $\vec{e}$. Lai atvieglotu aprēķinus, $\vec{e}$ komponentes simulācijā atsevišķi netiek

aprēķinātas, jo aprēķinos pietiek tikai ar leņķi θ starp anizotropijas asi un magnētisko momentu. Svarīgi ir atzīmēt, ka vektori $\vec{n}$, $\vec{h}$ un $\vec{e}$ vienmēr atradīsies vienā plaknē.

No Stonera – Vohlfarta modeļa:

$$E = -\frac{KV}{2}\cos^2\theta - mHcos(\phi - \theta) \tag{36}$$

Meklējot enerģijas ekstrēmus, enerģijas atvasinājumu $\frac{\partial E}{\partial \theta}$ pielīdzinot nullei un Izmantojot substitūciju $x = cos\theta$ ir iespējams konstruēt algebrisku 4. kārtas vienādojumu, kura saknes ir enerģijas ekstrēmi [23]:

$$\left(\frac{H_a}{H}x + cos\phi\right)^2 (1 - x^2) = x^2 \sin^2\phi \tag{37}$$

Šī vienādojuma saknes arī ir iespējamie magnētiskā momenta leņķi ar anizotropijas asi, ko būs jāņem vērā aprēķinos.

2. MODELĒJAMĀS PROBLĒMAS ATRISINĀŠANA

2.1 Modeļa stabilitāte

Darbam centrālais diferenciālvienādojums:

$$\frac{d\vec{n}}{dt} = f\left(\phi, \frac{H_a}{H}\right)\left[\vec{h} - \vec{n}(\vec{n}\cdot\vec{h})\right] \tag{38}$$

Ir nelineārs, jo to nav iespējams uzrakstīt formā $\vec{n}' = L\vec{n}$, kur L ir lineārs atvasināšanas operators. Nelineārus diferenciālvienādojumus analītiski ir iespējams apskatīt tikai dažos specifiskos gadījumos un vispārīga atrisinājuma atrašana ir, ja ne neiespējams, tad vismaz ārkārtīgi sarežģīts process. Par spīti tam, ir iespējams iegūt informāciju par vienādojumā aprakstītās fizikālās situācijas uzvedību, vienādojumu faktiski neatrisinot. To var darīt, pētot diferenciālvienādojuma stabilitāti.

Lai pētītu vienādojuma stabilitāti, tipiski tiek meklēti stacionārie punkti un nosacījumi pie kuriem tie parādās. Lai iegūtu informāciju par stabilitāti, fāzu telpā ir jāatrod reģioni, kuros vienādojums ir stabils, nestabils un metastabils. Stabilitātes analīze tiks veikta skaitliski, izvēloties līdzsvara nosacījumus vektoram $\vec{n}$, tālāk vektoru perturbējot un apskatot perturbācijas attīstību laikā. Perturbācija, atkarībā no parametriem, laikā var augt, var dilt un var nemainīties. Turpmāk ar „stabilu" tiks saprasta un apskatīta viena iespējamā situācija, kuru apraksta diferenciālvienādojums, kas arī tiks implementēta skaitliski – ar lauku sinhrons režīms.

Sinhrona kustības režīma gadījumā, kad anizotropijas ass precesē līdzi laukam ar tādu pašu periodu kā laukam, izpildās vienādojums:

$$\frac{d\vec{n}}{dt} = \overrightarrow{\omega_h} \times \vec{n} \tag{39}$$

Kam ir tiešs analogs mehānikā:

$$\dot{\vec{r}} = \vec{\omega} \times \vec{r} \tag{40}$$

Tāpēc tas netiks izvests. Jāatzīmē, ka $\overrightarrow{\omega_H}$ nav vienības vektors, savukārt visi pārējie vektori diferenciālvienādojumā ir vienības vektori. Tālāk jāatrod jau minētie vektora $\vec{n}$ līdzsvara nosacījumi. Mēģinājumi to darīt analītiski nevainagojās ar panākumiem, tāpēc tika izmēģinātas vairākas skaitliskas pieejas, lai atrastu sākuma nosacījumus vektoram $\vec{n}$, kas tālāk tiek apskatītas nodaļās 2.1 un 2.2.

2.2 Atrisinājuma implementācija

Lai iegūtu kvantitatīvus secinājumus par feromagnētiskas daļiņas dinamiku precesējošā ārējā laukā, jāatrisina no Stonera – Vohlfarta modeļa iegūtais diferenciālvienādojums:

Izvēlētā ir atrisināšana tuvināti, izmantojot skaitliskās metodes un brīvpieejas programmēšanas valodu *Python*. Darba gaitā tika izmēģinātas vairākas skaitliskās metodes un paņēmieni, lai risinātu diferenciālvienādojumu

Tiks izmantoti jauni apzīmējumi. Vienības vektoru $\vec{e}, \vec{n}, \vec{h}$ skalārie reizinājumi $\vec{n} \cdot \vec{h}$ un $\vec{e} \cdot \vec{n}$ tiks apzīmēti ar vektoru savstarpējo leņķu kosinusiem (skat. Attēlu 1.4.1):

$$\vec{e} \cdot \vec{n} = cos\theta \tag{41}$$

$$\vec{n} \cdot \vec{h} = cos\phi \tag{42}$$

Lai atrisinātu no Stonera – Vohlfarta modeļa iegūto diferenciālvienādojumu (38), anizotropijas ass vērsuma vektora $\vec{n}$ atvasinājums tiek sadalīts trīs vienādojumos, pa komponentēm:

$$\frac{dn_x}{dt} = f(\vec{e}, \vec{n})[h_x - n_x cos\phi] \tag{43}$$

$$\frac{dn_y}{dt} = f(\vec{e}, \vec{n})\left[h_y - n_y cos\phi\right] \tag{44}$$

$$\frac{dn_z}{dt} = f(\vec{e}, \vec{n})[h_z - n_z cos\phi] \tag{45}$$

Tālāk tiks apskatītas programmā lietotās funkcijas un problēmas ar kurām darba autors saskārās atrisinājuma implementācijas gaitā. Vairākās vietās nācās ieviest *print* komandas, kas rīka darbības laikā atgrieza kādu *if...else* testu rezultātus, vektora moduļus, parametru vērtības pie kurām iegūti rezultāti u.t.m.l.

2.2.1 Funkcija *hlauks* (α_H, t)

Magnētiskā lauka vērsuma vektora $\vec{h}$ komponentes nav nepieciešams pārrēķināt katrā iterācijā, tās ir neatkarīgas no diferenciālvienādojuma un programmā, zinot w_a, simulācijas paredzēto laika posmu, kā arī izvēloties lauka sākuma nosacījumus, tās ir iespējams aprēķināt visai simulācijai uzreiz. Izdevīgāk ir nevis glabāt izveidot un glabāt atmiņā veselu masīvu, bet saukt funkciju, kad nepieciešams, jo, piemēram, Runges - Kutta 4. Kārtas metodē ir nepieciešams aprēķināt magnētisko lauka vērtību starp simulācijas laika soļiem un tam masīvs nav praktisks. Funkcijā ievadot lauka precesijas leņķi „alfa_h" , izvēloties precesijas sākuma leņķi.

2.2.2 Funkcija Fi ($\vec{n}, \vec{h}$)

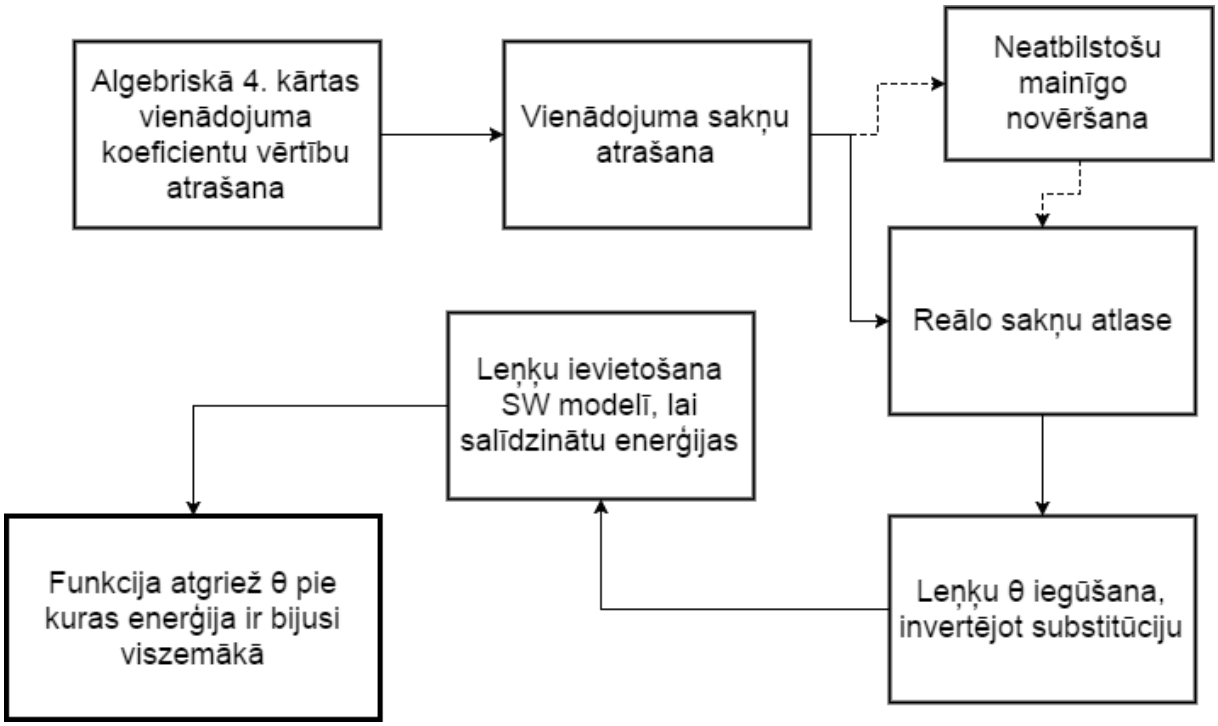
Pirmā problēma, kas jāatrisina, ir leņķa ϕ iegūšana, izmantojot vienības vektorus $\vec{n}$ un $\vec{h}$. To ir salīdzinoši vienkārši izdarīt, jo valodā *NumPy* ir iebūvēta divu vektoru skalārās reizināšanas opcija. Operācija ir vienkārša:

$$\phi = \arccos(\vec{n} \cdot \vec{h})$$

Šeit nācās saskarties ar retiem gadījumiem, kad, reizinot divus vienības vektorus, rezultāts pārsniedza vienīnieku,kas notika mašīnprecizitātes kļūdas jeb tā saucamās „floating point" kļūdas dēļ, tāpēc, automatizējot masveida aprēķinus, izmantojot Cos^{-1} funkciju, bija nepieciešams noapaļot reizinājumu līdz četriem zīmīgajiem cipariem.

2.2.3 Funkcija Teta ($\phi, H, \frac{\omega_H}{\omega_a}, \alpha_H$)

Leņķi teta atrod izmantojot enerģijas ekstrēmu atrašanu un salīdzināšanu kā aprakstīts nodaļā 1.4. No vienādojuma (37) atrastajām saknēm $x_1, x_2 x_3$ un $x_4$tiek atmestas tās, kurām ir nenulles imaginārā daļa, atlikušās tiek ievietotas atpakaļ substitūcijā, tad no saknēm iegūtie leņķi tiek ievietoti Stonera – Vohlfarta vienādojumā. Leņķa θ vērtība, pie kuras enerģija ir viszemākā, tiek izmantota tālāk, iterācijā. Arī šeit tika sastapta problēma ar mašīnprecizitāti kas izpaudās tā, ka dažas reizes parādījās kļūda, jo sakne kurai tika izpildīta Cos^{-1} bija pavisam mazliet lielāka par viens. Arī šeit tas tika izlabots, izmantojot noapaļošanu.



Attēls 2.2.3: Leņķa θ aprēķināšanas shēma (funkcija Teta)

2.2.4 Eilera metode

Pirmā, kas tika lietota, bija Eilera metode. Pēc vairākiem mēģinājumiem implementēt metodi ar mērķi iegūt aptuvenu kvalitatīvu izpratni par atrisinājumu, šī metode bija jāatmet, jo atrisinājums bija nestabils un fizikāli nepareizs visiem izmēģinātajiem laika soļiem, pie kuriem saprātīgā laikā bija iespējams iegūt rezultātus, kā arī atrisinājums kvalitatīvi nemainījās, samazinot laika soli 1000 reizes. Tas ir skaidrojams ar to, ka Eilera metode pie vektora rotācijas palielina vienības vektora garumu.

2.2.5 Runges – Kutta 4. kārtas metode (RK4)

Otrā metode, kas tikai izmēģināta, bija Runges – Kutta 4. kārtas metode (turpmāk darbā vienkārši "RK4"), izmantojot *Python* valodā pieejamās bibliotēkas *SciPy* rīku *odeint*. Šeit bija sastopamas ievērojamas grūtības panākt, lai *odeint* rīks risināšanas laikā atļauj izmainīt ar algoritmisku metodi aprēķināmā parametra θ vērtību katrā iterācijā, kuras nebija iespējams atrisināt. Tika pieņemts lēmums pašam implementēt RK4 metodi, lai izveidotu programmu, kas varētu atrisināt apskatāmo diferenciālvienādojumu.

RK4 ir metode, kas izmanto augstāku kārtu atvasinājumu vidējošanu, lai pat pie relatīvi rupjiem laika soļiem iegūtu analītiskajiem atrisinājumiem diezgan tuvus rezultātus. Vispārīgā gadījumā, ja ir dots atvasinājums:

$$y' = f(y,t)$$

Tad katru nākamo funkcijas vērtību var aprēķināt pēc shēmas:

$$y_{n+1} = y_n + \frac{\Delta t}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

Kur Δt ir laika solis.

$$t_{n+1} = t_n + \Delta t$$

Metodē lietotie koeficientik_1, k_2, k_3, k_4 ir RK4 metodes koeficienti, kas katrā iterācijā tiek secīgi pārrēķināti, izmantojot jau aprēķinātos koeficientus un zināmo atvasinājuma funkciju.

$$k_1 = f(t_n, y_n)$$

$$k_2 = f\left(t_n + \frac{\Delta t}{2}\ , y_n + \frac{\Delta t}{2}k_1\right)$$

$$k_3 = f\left(t_n + \frac{\Delta t}{2}\ , y_n + \frac{\Delta t}{2}k_2\right)$$

$$k_4 = f(t_n + \Delta \mathrm{t}\ , y_n + \Delta t k_3)$$

2.2.6 Funkcija $k(\vec{n}, \vec{h}, \phi, \theta)$

Nākamā problēma, kas jāapskata, ir RK4 koeficientu atrašana. Vienīgā tiešā atkarība no laika ir magnētiskā lauka vektorā un tātad arī leņķī ϕ. Šeit indekss i apzīmē x, y un z koordinātas, katrai koordinātai rīkā tiek aprēķināts savs koeficientu komplekts:

$$\mathrm{k_{1i}} = \mathrm{f(t_n, y_n)} = \ \omega_\mathrm{a}\frac{\cos^2\theta}{\cos\phi + \frac{\mathrm{H_a}}{\mathrm{H}}\cos\theta}[\mathrm{h_i} - \mathrm{n_i}\cos\phi]$$

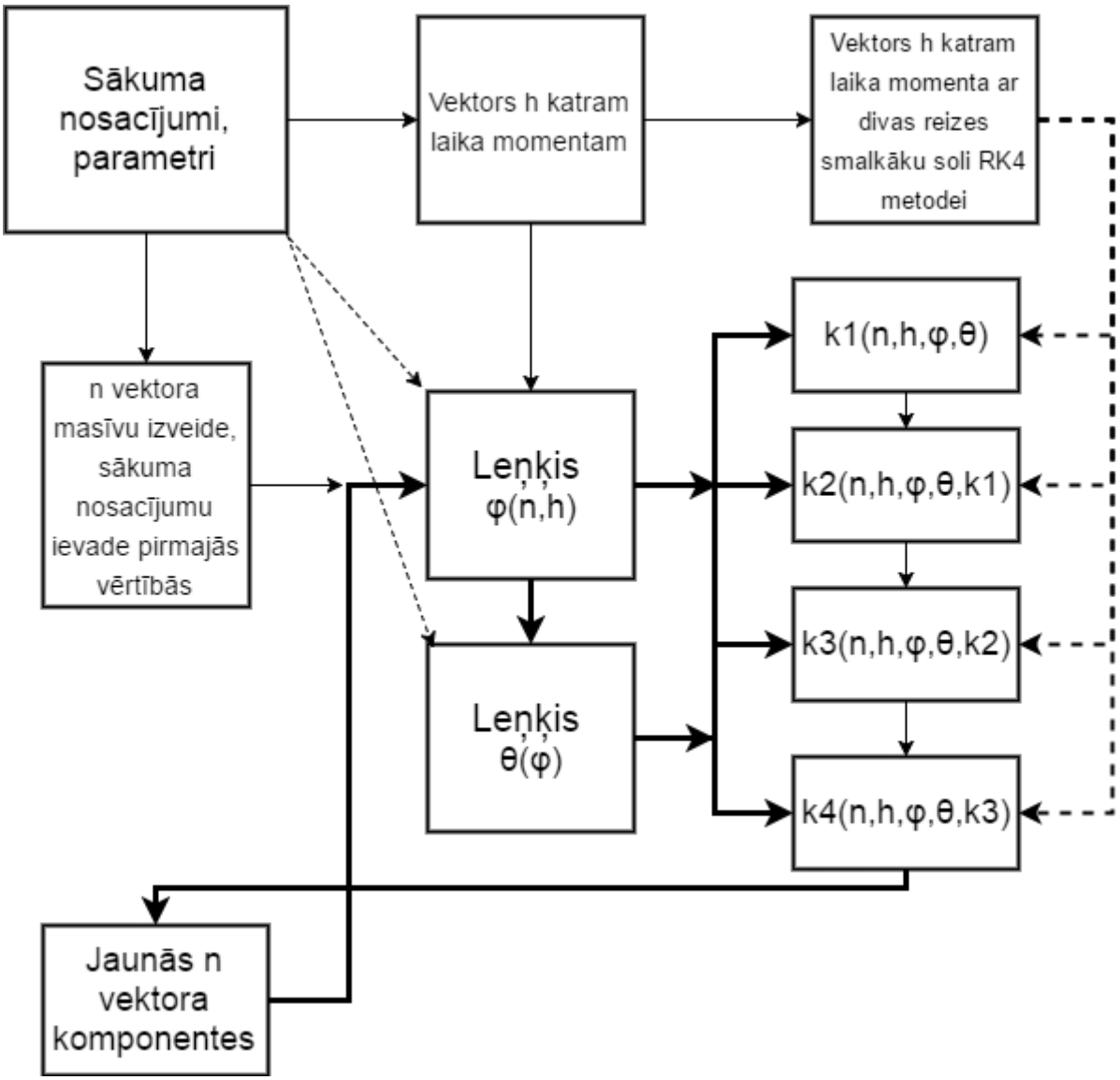
$$\mathrm{k_{2i}} = \omega_\mathrm{a}\frac{\cos^2(\theta(\mathrm{n} + \mathrm{k_1}\Delta\mathrm{t}/2\ ))}{\cos[\phi(\mathrm{t} + \Delta\mathrm{t}/2)] + \frac{\mathrm{H_a}}{\mathrm{H}}\cos\theta(\mathrm{n} + \mathrm{k_1}\Delta\mathrm{t}/2)}[\mathrm{h_i}(\mathrm{t} + \Delta\mathrm{t}/2) - \mathrm{n_i}(\mathrm{t} + \Delta\mathrm{t}/2)\cos[\phi(\mathrm{t} + \Delta\mathrm{t}/2)]]$$

$$\mathrm{k_{3i}} = \omega_\mathrm{a}\frac{\cos^2(\theta(\mathrm{n} + \mathrm{k_2}\Delta\mathrm{t}/2))}{\cos[\phi(\mathrm{t} + \Delta\mathrm{t}/2)] + \frac{\mathrm{H_a}}{\mathrm{H}}\cos\theta(\mathrm{n} + \mathrm{k_2}\Delta\mathrm{t}/2)}[\mathrm{h_i}(\mathrm{t} + \Delta\mathrm{t}/2) - \mathrm{n_i}(\mathrm{t} + \Delta\mathrm{t}/2)\cos[\phi(\mathrm{t} + \Delta\mathrm{t}/2)]]$$

$$\mathrm{k_{4i}} = \omega_\mathrm{a}\frac{\cos^2\theta(\mathrm{n} + \mathrm{k_3}\Delta\mathrm{t})}{\cos[\phi(\mathrm{t} + \Delta\mathrm{t})] + \frac{\mathrm{H_a}}{\mathrm{H}}\cos\theta(\mathrm{n} + \Delta\mathrm{t}\mathrm{k_3})}[\mathrm{h_i}(\mathrm{t} + \Delta\mathrm{t}) - \ \mathrm{n_i}(\mathrm{t} + \Delta\mathrm{t})\cos[\phi(\mathrm{t} + \Delta\mathrm{t})]]$$

Izmantojot attēlā 2.2.6 aprakstīto aprēķina shēmu un atvasinājuma izteiksmes un izvēloties gadījuma rakstura sākuma nosacījumus, ir iespējams iegūt sākotnējos rezultātus, lai

iegūtu priekštatu par sistēmas uzvedību pie dažām parametru kombinācijām, kas ir raksturojami kā pieņemami.



Attēls 2.2.6: Vispārīga diferenciālvienādojuma risināšanas shēma. Raustītās līnijas nozīmē netiešu atkarību, parametri tiek lietoti, bet dinamiski nemainās, biezās līnijas apzīmē ciklu.

3. SKAITLISKĀ STABILITĀTES ANALĪZE

Līdz šim uzrakstītā programma spēj dot priekšstatu tikai par atsevišķām parametru kombinācijā, bet nedod ieskatu vispārīgajā sistēmas uzvedībā. Kā jau minēts nodaļā 2.1, tiek pētīti tikai kustības režīmi, kuri atrodas sinhronas precesijas režīmā.

3.1 Bisekciju metode sākuma nosacījumu meklēšanai

Izmantojot substitūciju:

$$\overrightarrow{\omega_h} \times \vec{n} = \vec{P} \tag{46}$$

Ir iespējams risināmo diferenciālvienādojumu uzrakstīt komponentēs:

$$P_x = f\left(\phi, \frac{H_a}{H}\right)[h_x - n_x cos\phi\ ] \tag{47}$$

$$P_y = f\left(\phi, \frac{H_a}{H}\right)[h_y - n_y cos\phi] \tag{48}$$

$$P_z = f\left(\phi, \frac{H_a}{H}\right)[h_z - n_z cos\phi] \tag{49}$$

Pārnesot P_x, P_y un P_z uz labo pusi, saskaitot visus trīs vienādojumus un nomainot zīmes, ir iespējams iegūt vienādojumu formā $F(\vec{n}) = 0$, ko tālāk ir iespējams minimizēt ar bisekciju metodi vai ar citādiem paņēmieniem.

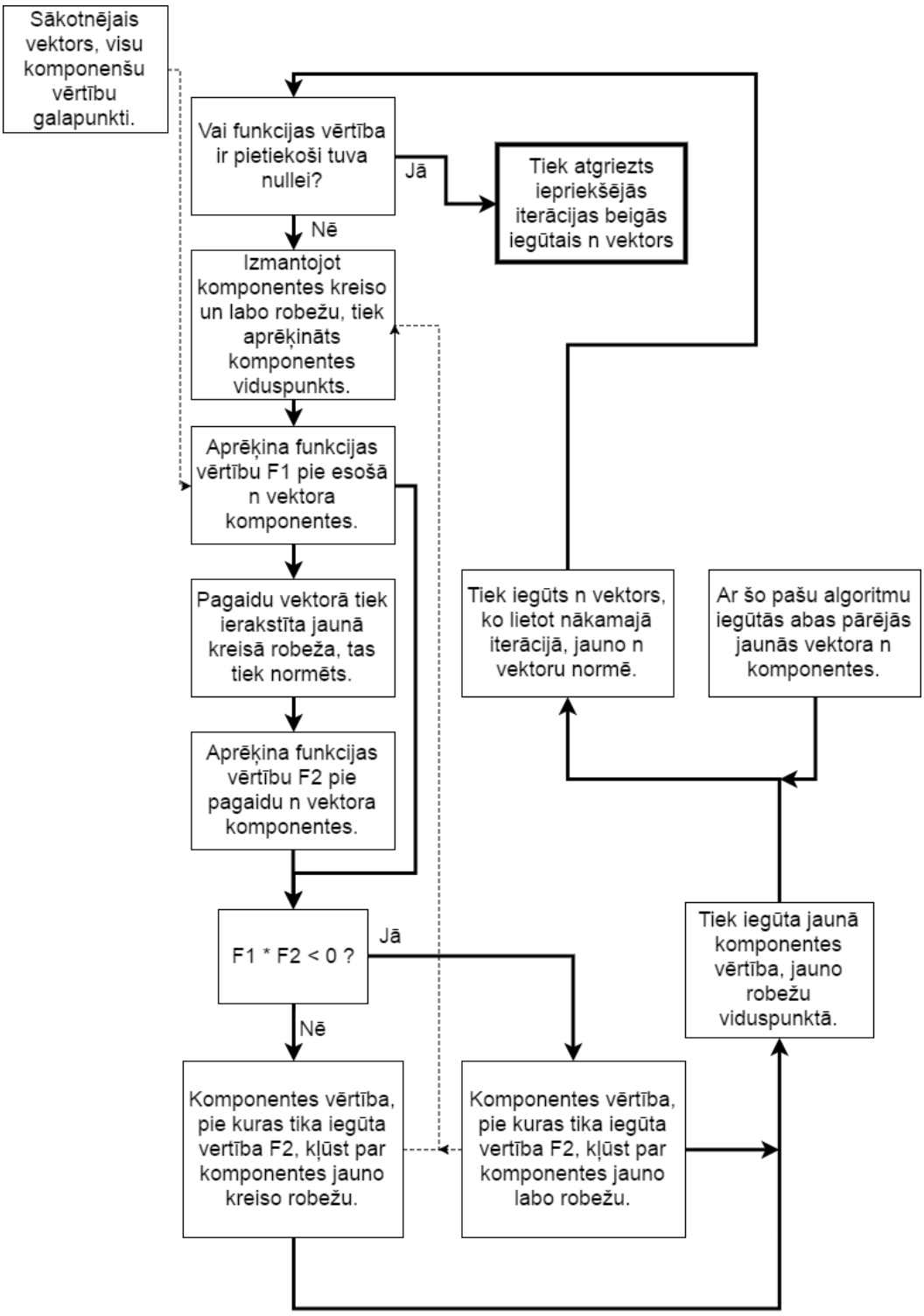
$$F(\vec{n}) = P_x + P_y + P_z - f\left(\phi, \frac{H_a}{H}\right)\left([h_x - n_x cos\phi\ ] + \left[h_y - n_y cos\phi\right] + [h_z - n_z cos\phi]\right) \tag{50}$$

Šādi ir iespējams tuvoties vektora $\vec{n}$ nepieciešamām komponenšu vērtībām no visām pusēm. Vienādojums tiek minimizēts iteratīvi. Tiek izveidota funkcija, kas pie dota $\vec{n}$ aprēķina $F(\vec{n})$. Ir nepieciešams izveidot arī sākotnējo vienības vektoru $\vec{n}$ no kura sākt iterēt uz patieso vērtību, jo funkcijā ir nepieciešams aprēķināt $\vec{P}$ un to var izdarīt tikai ar $\vec{n}$, kura modulis ir viens.

Šādai pieejai bija vairāki trūkumi. Galvenais trūkums: mainot vienu vektora komponenti, mainās arī vektora skalārais reizinājums. Citiem vārdiem, bisekcijā izvēlētā kādas komponentes izmaiņa izmaina arī atrisinājumu. Tikai izveidots atsevišķs koda fragments ar kuru tika testēti ar bisekciju iegūtie vektora nosacījumi. Rezultāti nebija apmierinoši,neatkarīgi no tā vai pārbaudes gaitā tika mainīts vai nemainīts skalārais reizinājums, kā arī neatkarīgi no tā, kad vektori tika normēti un kā tikai izvēlēts sākotnējais n vektors. Visos izmēģinātajos aprēķina variantos iegūtais vektors $\vec{n}$ nebija atrisinājums vienādojumam:

$$\overrightarrow{\omega_h} \times \vec{n} = f\left(\phi, \frac{H_a}{H}\right)\left[\vec{h} - \vec{n}(\vec{n} \cdot \vec{h})\right] \tag{51}$$

Bija nepieciešams izvēlēties citu pieeju kā meklēt simulācijas sākuma nosacījumus.



Attēls 3.1: Viens no izmēģinātajiem bisekciju metodes pieliuetojumiem simulācijas sākuma nosacījumu atrašanai

3.2 Alternatīvas sākuma nosacījumu atrašanai

Bisekciju metodes trūkumu dēļ, tikai izmantoti VISI pieejamie *SciPy* nelineāro vienādojumu risināšanas rīki. [24]

Katra nākamā metode tika izmēģināta, ja iepriekšējā nekonverģēja uz risinājumu. Koda fragments, kas par to ir atbildīgs, secīgi izmēģina šādas metodes, lai atrastu sakni funkcijai $F(\vec{n}) = 0$:

- Ņūtona – Krilova metode, izmantojot Krilova aproksimāciju,
- Extended Anderson mixing metode,
- Broyden`s first Jacobian approximation metode,
- Broyden`s second Jacobian approximation metode,
- Diagonal Jacobian approximation metode,
- Scalar Jacobian approximation metode
- Diagonal Broyden Jacobian approximation metode

Visām uzskaitītajām metodēm bija nepieciešams minējums, no kura sākt iterēt, lai meklētu vienādojuma $F(\vec{n}) = 0$ atrisinājumu. Kā sākuma minējums tiek izmantots magnētiskā lauka vienības vektors $\vec{h}$ simulācijas sākuma momentā, kas ir diezgan labs minējums, jo $\vec{n}$ būtu jābūt vērstam aptuveni tādā pašā virzienā. Ja neviena metode nekonverģē, tiek izmantots for cikls, lai atkārtotu mēģinājumus 3 reizes ar jaunu minējumu, magnētiskā lauka vektoru nobīdot par aptuveni trešdaļu no rotācijas perioda katru reizi, lai atrisinājumam būtu iespējams tuvoties no dažādām pusēm. Nav izslēgts arī, ka vienādojumam eksistē vairāk kā viens atrisinājums pie kādām parametru kombinācijām, bet tas nesagādā problēmas, jo, ja vektors, kaut tuvināti, apmierina vienādojumu, tas der kā sākuma nosacījums. Ja ir vairāki atrisinājumi, tad der jebkurš.

3.3 Bisekciju „Vidējās vērtības" metode fāzu diagrammas izveidei

Lai veidotu fāzu diagrammas, bija jāatrod veids kā paātrināti atrast metastabilo punktu virsmu, kuru nepieciešams atrast stabilitātes pētījumiem. Apmierinošs virsmas grafiks visā fāzu telpā būtu jābāzē kā minimums uz 30^3 elementu režģi. Šāds aprēķins, pat ja perturbētā un neperturbētā simulācija aizņemtu tikai pussekundi, aizņem četras stundas. To ir iespējams optimizēt ar sava veida bisekcijas algoritmu, kas aprakstīts 3.3.2.

Pieņēmuma pamatā ir apgalvojums, ka fāzu telpā starp stabilu un nestabilu punktu eksistē vismaz viens metastabils punkts. Tāpēc, lai pētītu sistēmas uzvedību pie vāja, vidēja un stipra ārējā lauka, tika izvēlētas trīs parametra H_a/H vērtības intervālā $[0.1\,,2]$, 20 parametra ω_H/ω_a vērtības intervālā $[0.1\,,2]$ un 10 parametra α_H vērtības intervālā $[0.1\,,1.57]$. Abu simulāciju aprēķini tika ievietotas funkcijā, kas aprēķina masīvus abām simulācijām katram laika momentam, tad salīdzina to starpības moduli, izmantojot funkciju probe(x,ModPerturb)

3.3.1 Funkcija *probe (x,ModPerturb)*

Vadoties pēc darba vadītāja ieteikuma, kā stabilitātes kritērijs, tika izvēlēts nosacījums:

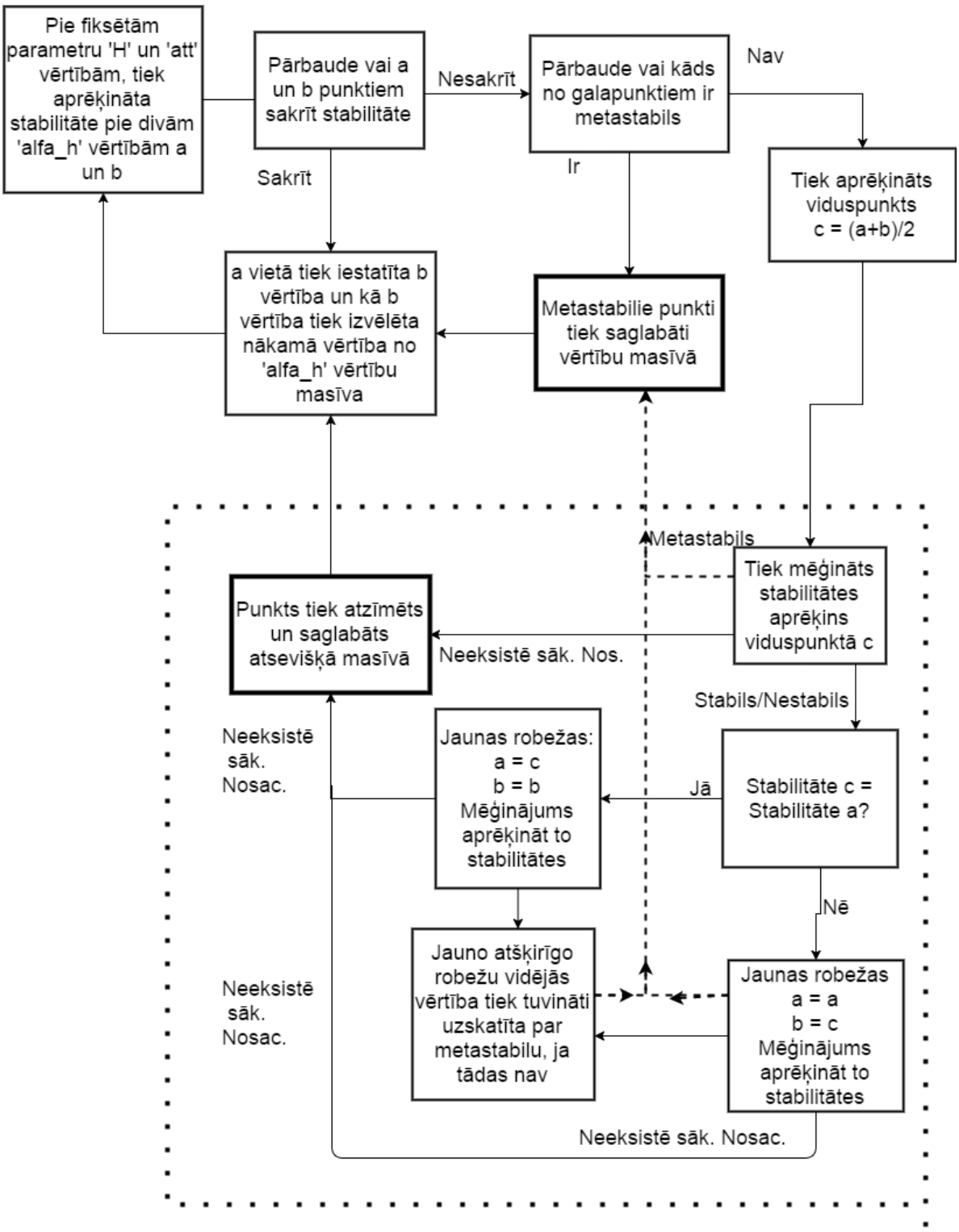
- Ja abu simulāciju priekšpēdējā laika momenta starpības modulis ir vismaz 1.05 reizes lielāks, nekā sākotnējā perturbācijas vektora modulis, tad fāzu telpā punkts tiek atzīmēts kā nestabils
- Ja abu simulāciju priekšpēdējā laika momenta starpības modulis ir mazāks kā $0.95 \cdot ModPerturb$ (sākotnējā perturbācijas vektora modulis), tad fāzu telpā punkts tiek atzīmēts kā stabils
- Izņēmuma gadījumā, kad starpības vektora modulis atrodas robežās starp 0.95 un 1.05 reizinājuma ar sākotnējā perturbācijas vektora moduli, punkts tiek atzīmēts kā nestabils.

3.3.2 „Vidējās vērtības" metodes pielietojums

Izveidotā funkcija, kas no parametriem H_a/H , ω_a/ω_Hun $\alpha_H$aprēķina sākuma nosacījumus un paralēli palaiž perturbētu un neperturbētu simulāciju, atgriež to vai šī parametru kombinācija dod stabilu vai nestabilu rezultātu. Lai iegūtu precīzu virsmu, ir nepieciešams apskatīt fāzu diagrammas smalkākā režģī kā paredzēts. Lai iegūtu vispārīgu skatu uz stabilitātes reģioniem, katra „rupjā" režģa punkta stabilitāte tiek saglabāta masīvā. Ja diviem punktiem $a = \alpha_H$ un $b = \alpha_H$` sakrīt stabilitātes rupjākajā režģī, tad tālāk šis reģions apskatīts netiek. Ja tiek atrasts intervāls, kur divas stabilitātes nesakrīt, tiek izpētīta stabilitāte tā viduspunktā $c = (a + b/2)$. Tālāk tiek salīdzināta stabilitāte starp punktiem a un c, un b un c. Atšķirīgo stabilitāšu punktiem tad atkal tiek pētīta stabilitāte to viduspunktā. Bisekcija tiek pārtraukta šādos gadījumos:

- Viduspunkti ir izmantoti, lai pārstatītu robežas trīs reizes, viduspunkts tiek tuvināti uzskatīts par labu esam un ir saglabāts metastabilo punktu masīvā
- Kādā punktā nav iespējams aprēķināt sākuma nosacījumus, punkts tiek atzīmēts un saglabāts atsevišķā masīvā
- Punkts ir metastabils, tas tiek saglabāts metastabilo punktu masīvā

Pārbaudes pārtraukuma gadījumiem notiek katrā iterācijā. Katrā aprēķinā, neatkarīgi no stabilitātes, punkta stabilitāte tiek saglabāta tajā pašā masīvā,kurā „rupjie" režģa punkti, kas palīdz labāk ilustrēt stabilos un nestabilos reģionus īpaši metstabilo punktu tuvumā.



Attēls 3.3.2: Aptuvena „Bisekcijas" shēma. Punktotais reģions var variēt, ja izvēlas izmantot for ciklu, lai metastabilos punktus atliktu precīzāk.

4. REZULTĀTI

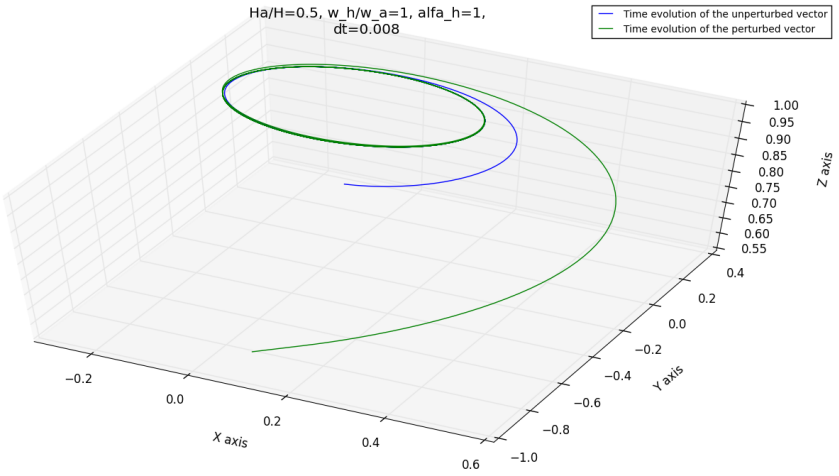
Vadoties pēc darba vadītāja ieteikuma, diferenciālvienādojuma stabilitāte tika pētīta skaitliski un tika mēģināts iegūt un apskatīt pēc iespējas vairāk punktu. Šāda pieeja ļāva atrast arī kustības režīmus pie sākuma nosacījumiem kā aprakstīts nodaļā 2.1 , kas nebija gluži sinhronās kustības režīmi. Rezultātos tiek apskatītas tikai simulācijas, kam tika veiksmīgi atrasti sākuma nosacījumi no vienādojuma $\overrightarrow{\omega_h} \times \vec{n} = \frac{d\vec{n}}{dt}$, izmantojot nelineāro vienādojumu risināšanas rīkus.

Par spīti visām izmēģinātajām metodēm, fāzu telpā tomēr eksistē punkti pie kuriem sākuma nosacījumus atrast neizdodas. Tas varētu būt skaidrojams ar to, ka pie šādām parametru kombinācijām sinhrons kustības režīms nav iespējams, tāpēc arī mēģinot tuvoties atrisinājumam no dažādām pusēm, vienādojumam neeksistē atrisinājums, kurš apmierinoši izpildītu nosacījumu $F(\vec{n}) = 0$ un to iteratīvi nevar atrast.

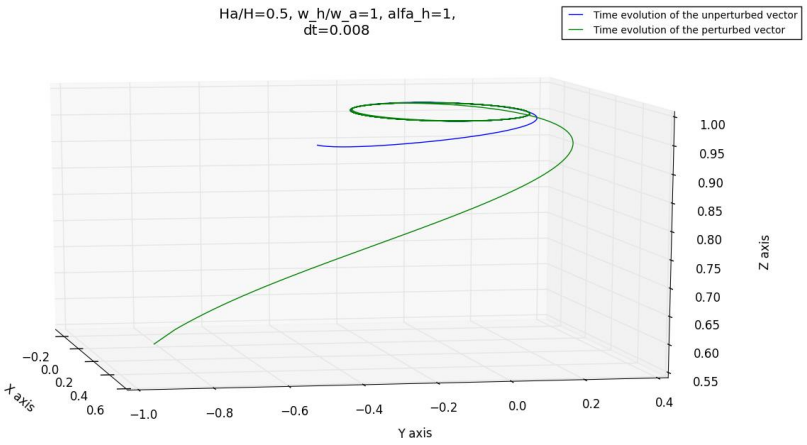
4.1 Simulācijas ar perturbāciju pie atsevišķu parametru kombinācijām

Sekojošajos grafikos ir attēlotas perturbētā un neperturbētā vektora galapunkti trīs dimensiju DOKS koordinātās:

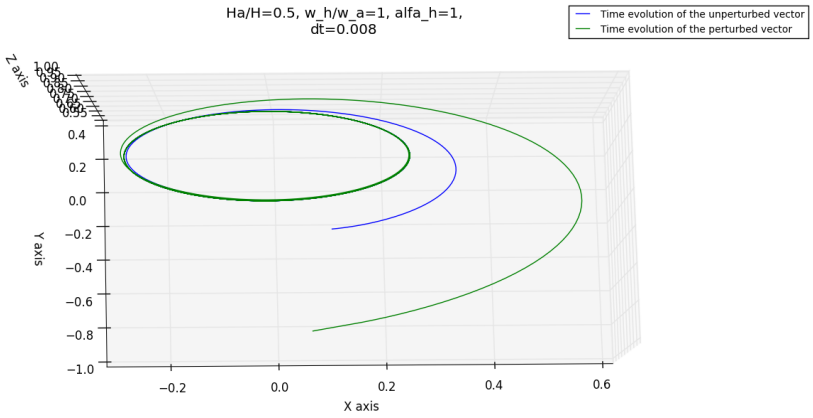
4.1.1 Tipiska stabila punkta piemērs



Attēls 4.1.1.1: Perturbētā un neperturbētā vektora galapunkta attīstība trīs dimensiju DOKS koordinātās pie dotām programmas parametru vērtībām atkarībā no laika.

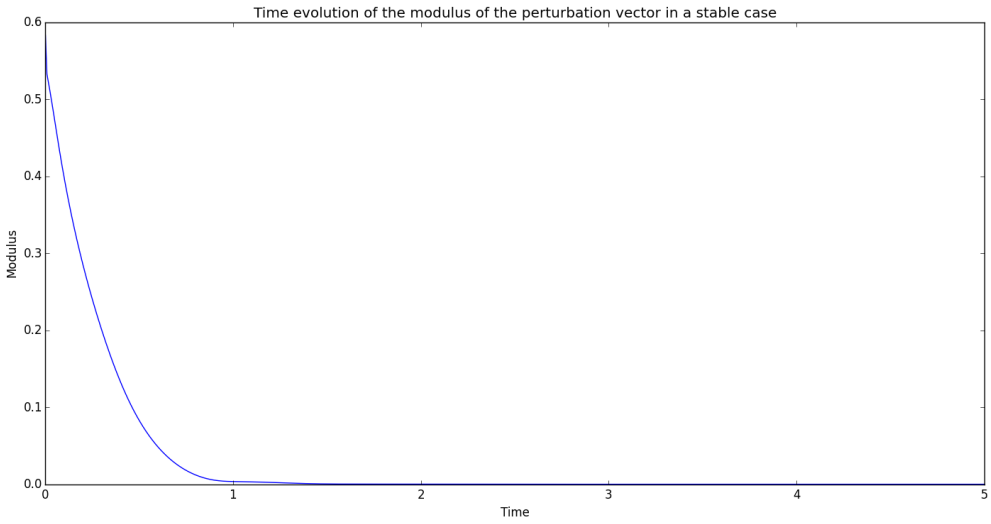


Attēls 4.1.1.2: Perturbētā un neperturbētā vektora galapunkta attīstība trīs dimensiju DOKS koordinātās pie dotām programmas parametru vērtībām atkarībā no laika.



Attēls 4.1.1.3: Perturbētā un neperturbētā vektora galapunkta attīstība trīs dimensiju DOKS koordinātās pie dotām programmas parametru vērtībām atkarībā no laika.

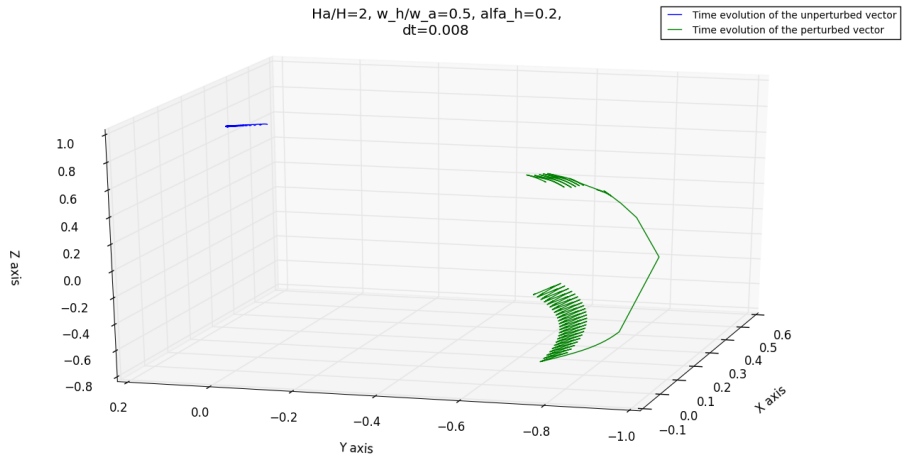
Visi punkti, kas tika pārbaudīti vizuāli (~15), kurus programma kvalificēja kā stabilus, vizuāli arī tādi bija. Kā paredzams, neperturbētais vektors ieiet sinhronā kustības režīmā un perturbētais vektors tam seko un to starpība dilst.



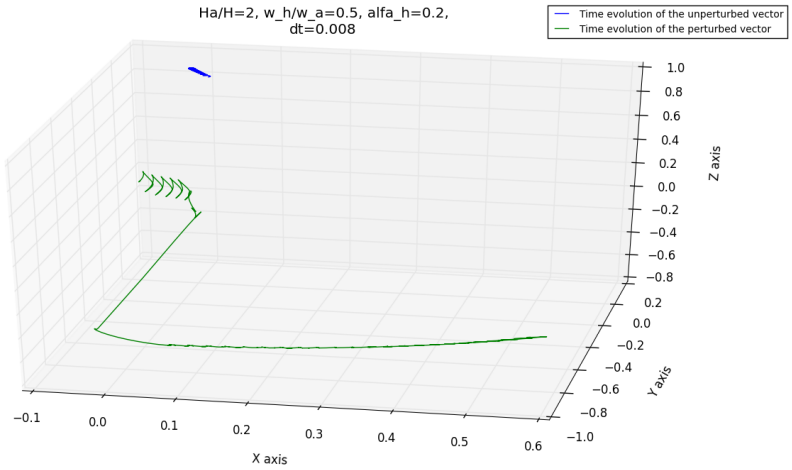
Attēls 4.1.1.4: Tipiska stabila punkta gadījuma simulācijas perturbētā un neperturbētā vektora starpības modulis atkarībā no laika

4.1.2 Nestabila punkta piemērs

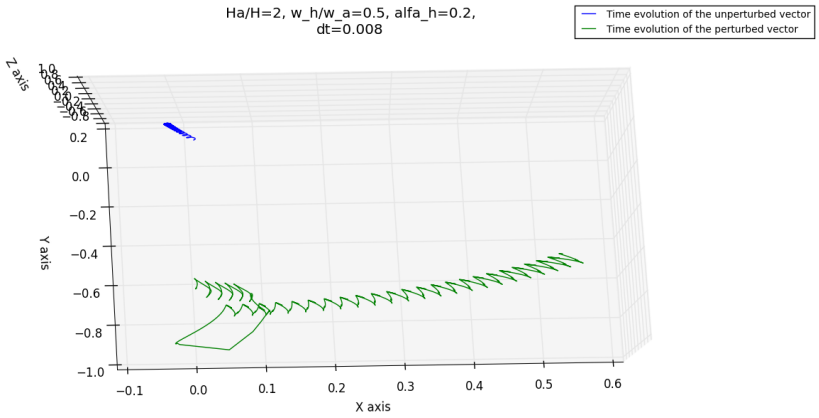
Pretēji stabiliem, nestabilo simulāciju kustības režīmi bija ārkārtīgi dažādi, visbiežāk no vizuāli pārbaudītajiem (~10) ir sastopami lecieni vai kustība, kura atgādina haotisku.



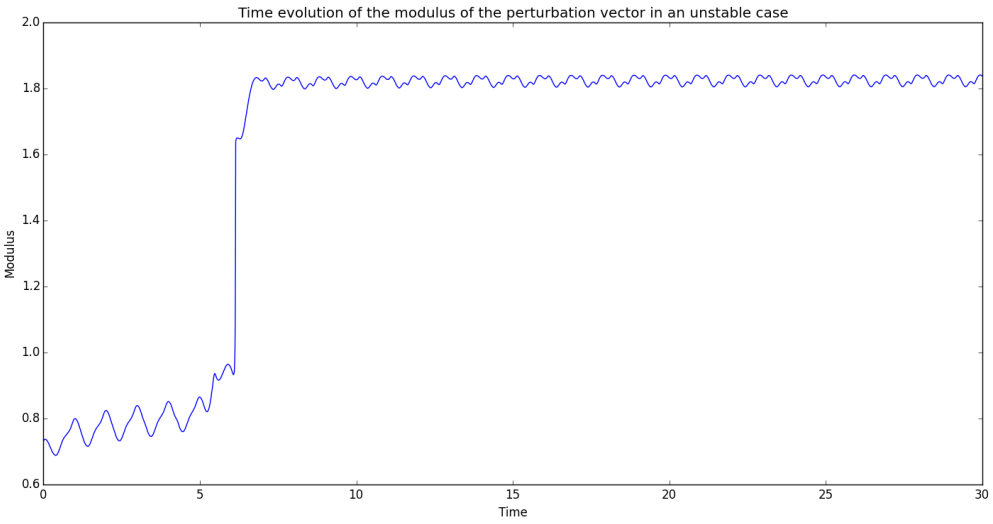
Attēls 4.1.2.1:Perturbētā un neperturbētā vektora galapunkta attīstība trīs dimensiju DOKS koordinātās pie dotām programmas parametru vērtībām atkarībā no laika.



Attēls 4.1.2.2: Perturbētā un neperturbētā vektora galapunkta attīstība trīs dimensiju DOKS koordinātās pie dotām programmas parametru vērtībām atkarībā no laika.



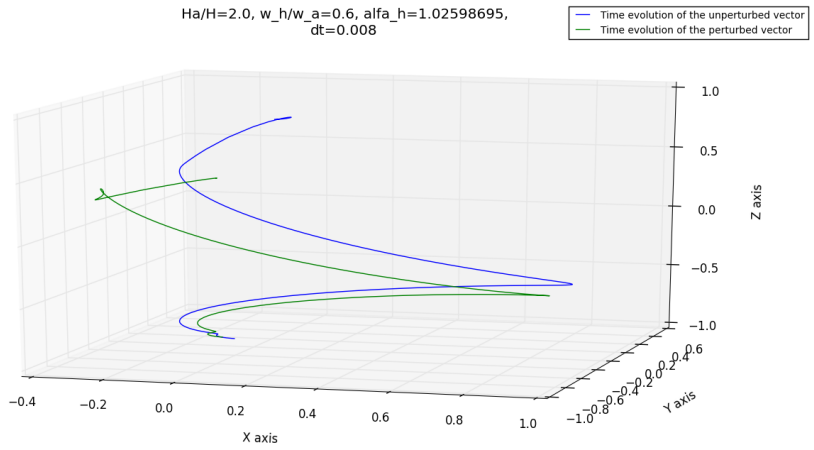
Attēls 4.1.2.3: Perturbētā un neperturbētā vektora galapunkta attīstība trīs dimensiju DOKS koordinātās pie dotām programmas parametru vērtībām atkarībā no laika.



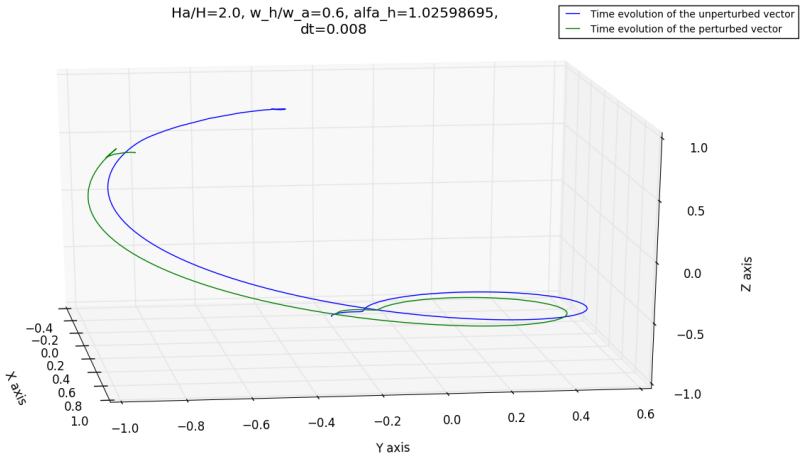
Attēls 4.1.2.4: Viena tipa nestabila punkta gadījuma simulācijas perturbētā un neperturbētāvektora starpības modulis atkarībā no laika.

4.1.3 Metastabila punkta piemērs

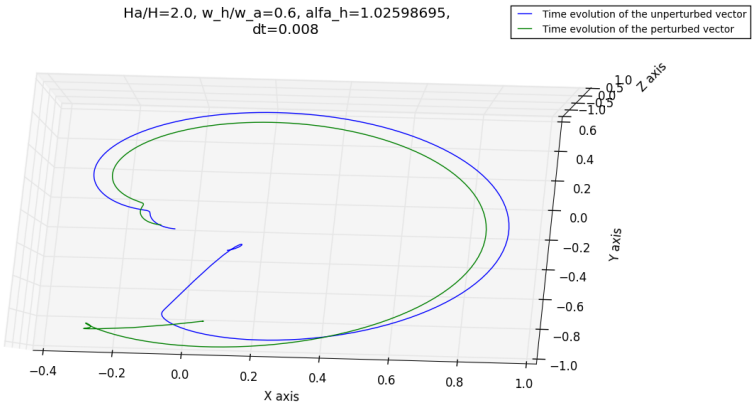
Tāpat kā nestabiliem punktiem, arī metastabiliem tika novērotas krasas atšķirības kustības režīmos. Neperturbētā vektora neatrašanās sinhronā režīmā rada pamatu domāt, ka iespējams vektora sākuma nosacījumi netiek atrasti pareizi, kā arī rada nepieciešamību izmantot citu pieeju, lai noteiktu punkta stabilitāti.



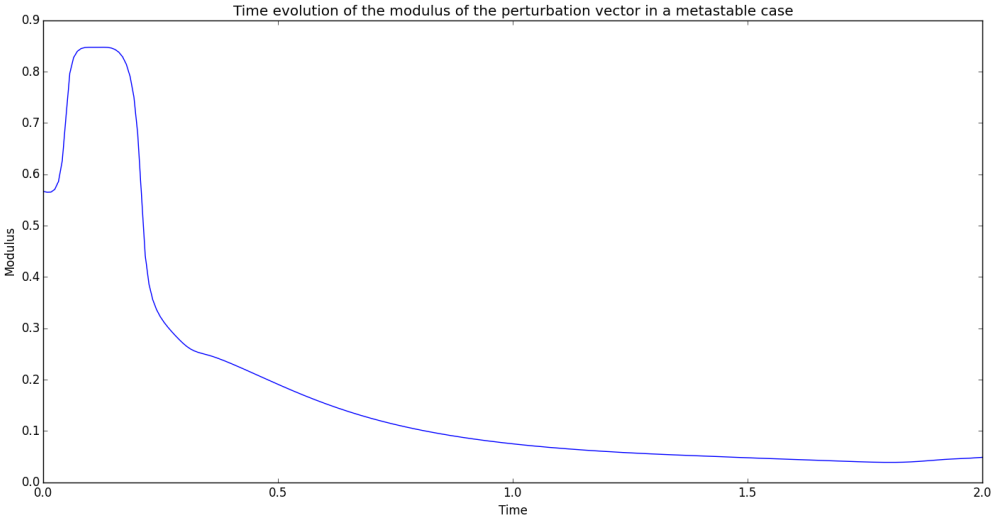
Attēls 4.1.3.1: Perturbētā un neperturbētā vektora galapunkta attīstība trīs dimensiju DOKS koordinātās pie dotām programmas parametru vērtībām atkarībā no laika.



Attēls 4.1.3.2: Perturbētā un neperturbētā vektora galapunkta attīstība trīs dimensiju DOKS koordinātās pie dotām programmas parametru vērtībām atkarībā no laika.



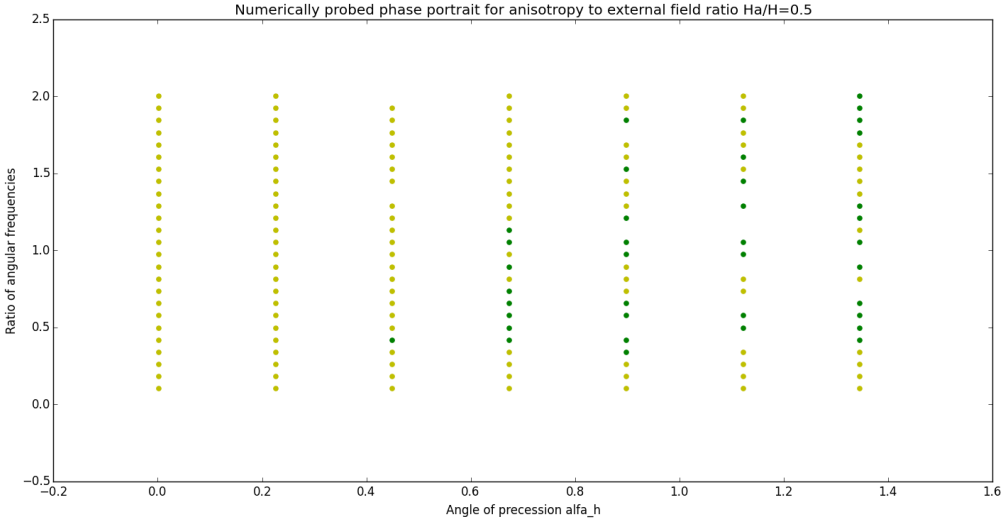
Attēls 4.1.3.3: Perturbētā un neperturbētā vektora galapunkta attīstība trīs dimensiju DOKS koordinātās pie dotām programmas parametru vērtībām atkarībā no laika.



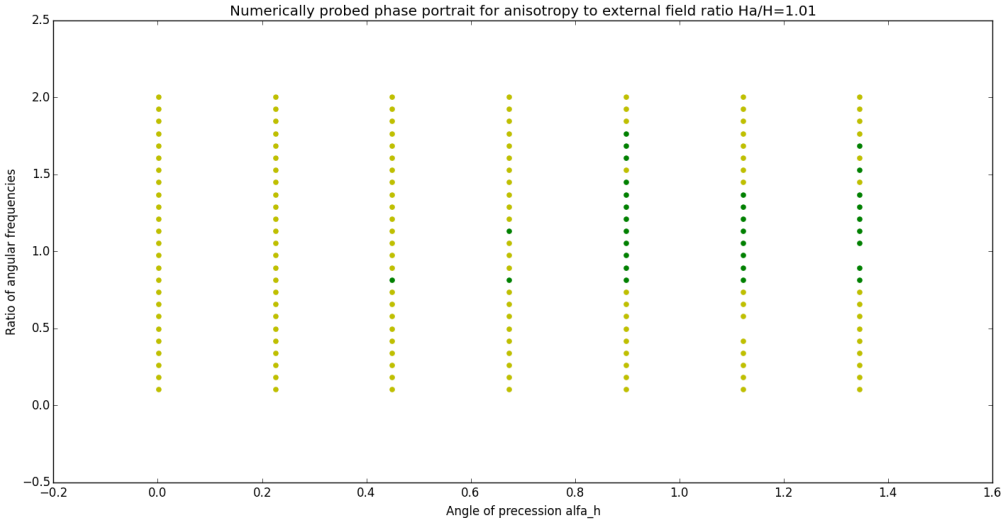
Attēls 4.1.3.4: Viena tipa metastabila punkta gadījuma simulācijas perturbētā un neperturbētāvektora starpības modulis atkarībā no laika

4.2 Skaitliski atrastās fāzu diagrammas

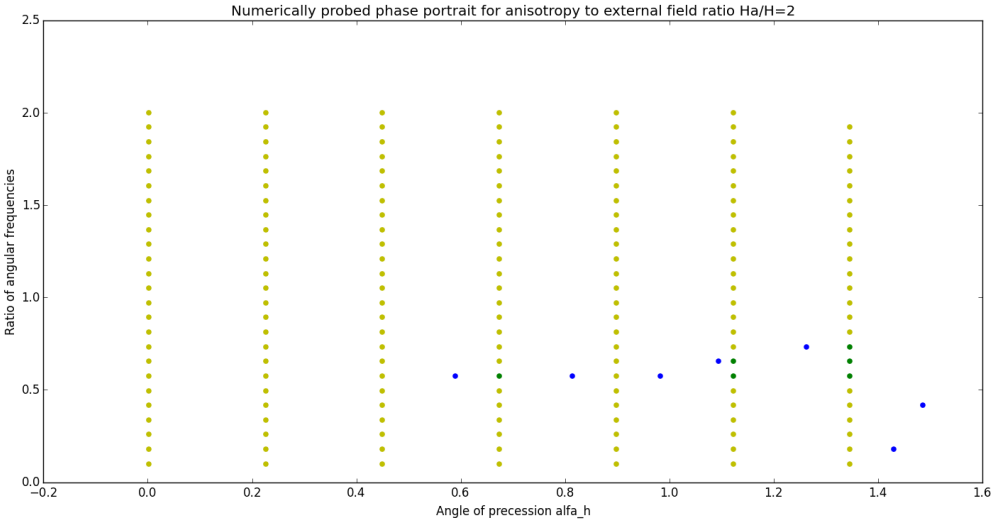
Izmantojot aprakstīto bisekcijas algoritmu, tika meklēti stabilie un nestabilie punkti fāzu telpā pie trim dažādām H_a/H vērtībām. Ar zaļu krāsu tiek apzīmēti stabili punkti, ar dzeltenu nestabili un ar zilu metastabili punkti.



Attēls 4.2.1: Skaitliski pētītā fāzu diagramma pie $\frac{H_a}{H} = 0{,}5$



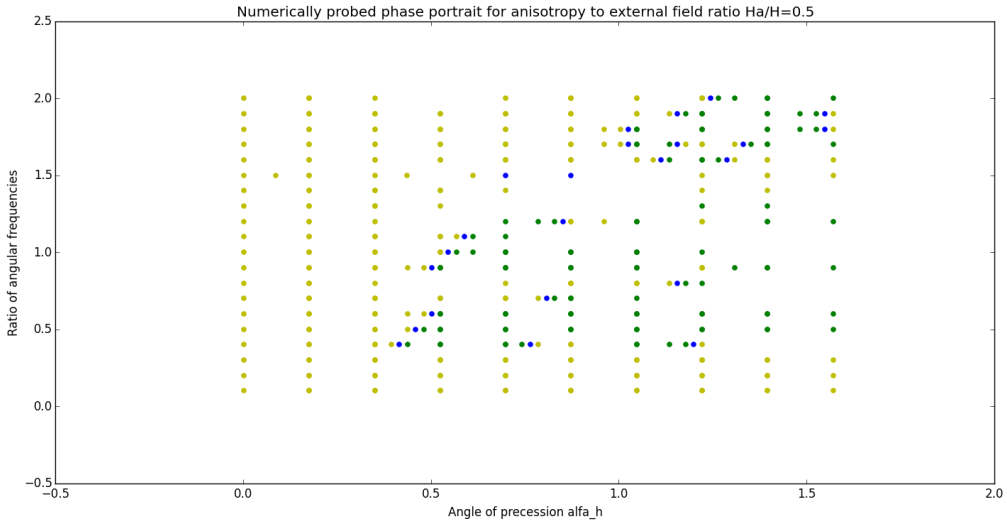
Attēls 4.2.1: Skaitliski pētītā fāzu diagramma pie $\frac{H_a}{H} = 1.01$



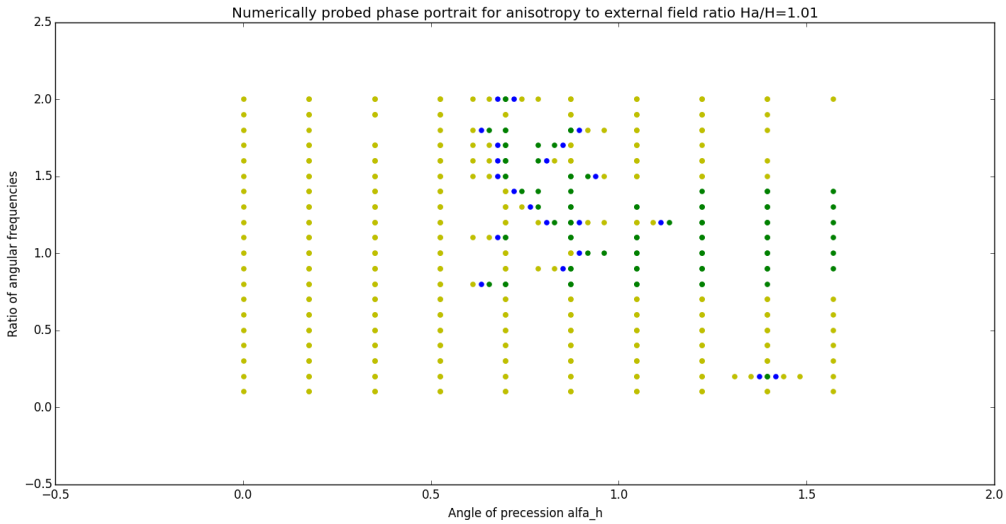
Attēls 4.2.1: Skaitliski pētītā fāzu diagramma pie $\frac{H_a}{H} = 2$

Kā var secināt, rezultātiem nav acīmredzamas interpretācijas, jo virsmu nav iespējams acīmredzami iezīmēt. Vizuāli, ir līdzība ar literatūrā atrodamu attēlu [25] , bet iegūtie rezultāti ir pārāk neprecīzi, lai varētu salīdzināt kvantitatīvi.

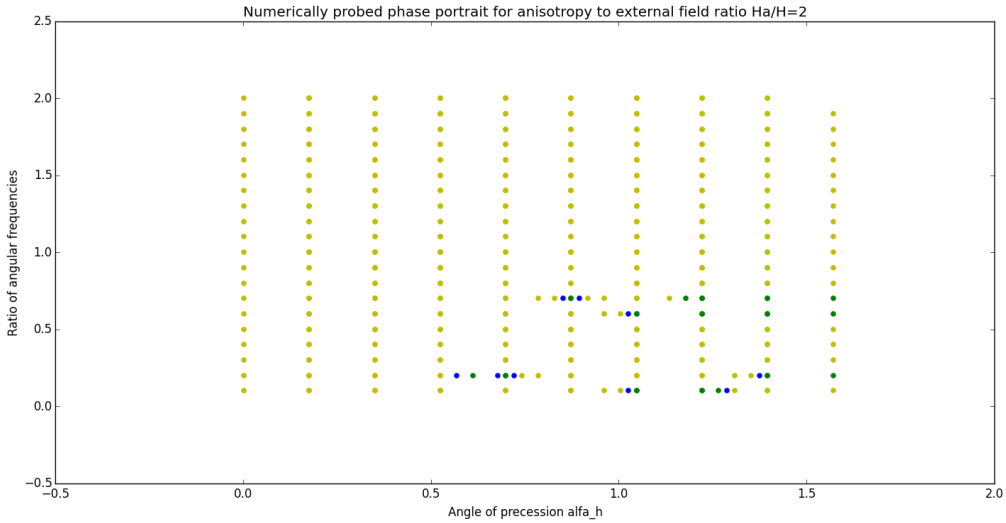
Nemainot nevienu parametru vai funkciju, šī pati simulācija tika palaista izmantojot citus „rupjā" režģa izmērus.



Attēls 4.2.1: Skaitliski pētītā fāzu diagramma pie $\frac{H_a}{H} = 0{,}5$



Attēls 4.2.1: Skaitliski pētītā fāzu diagramma pie $\frac{H_a}{H} = 1{,}01$



Attēls 4.2.1: Skaitliski pētītā fāzu diagramma pie $\frac{H_a}{H} = 2$

Ir novērojama mazliet kvalitatīvi savādāk uzvedība, ko visticamāk skaidro nepilnīgais stabilitātes analīzes mehānisms. Caurumi fāzu telpā ir apgabali, kuros nav bijis iespējams izrēķināt sākuma nosacījums, tāpēc tie vienkārši ir izlaisti.

5. SECINĀJUMI

1. Izdevās izveidot rīku, kas atrisināja daļiņas kustību aprakstošo diferenciālvienādojumu pētāmajā fāzu telpas reģionā pie fiksētām fāzu telpas parametru vērtībām.
    a. Izdevās novērot stabilus, metastabilus un nestabilus kustības režīmus, kuru uzvedība bija fizikāli interpretējama un neuzrādīja acīmredzami nepareizus rezultātus.
2. Neizdevās iegūt viennozīmīgas fāzu diagrammas pie dažādām H_a/H parametra vērtībām. Līdzšinējie rezultāti rāda, ka sistēmas stabilitāte var būt daudz sarežģītāka kā sākotnēji sagaidīts un tās pētīšanai būtu jāpielieto citas metodes. Stabilitātes pētīšanai iespējams būtu jāizmanto citi stabilitātes kritēriji kā tie, kas tika pielietoti.
3. Autors uzskata, ka nav izslēdzama arī haotiska uzvedība, jo eksistē gan jūtība uz sākuma stāvokli, gan periodiskas orbītas. No fāzu diagrammām ir iegūti secinājumi:
    a. Kvalitatīvi izdevās secināt, ka, pieaugot ārējā magnētiskā lauka stiprumam, sašaurinās anizotropijas un ārējā lauka frekvenču attiecību reģions pie kura ir novērojams stabils stāvoklis.
    b. Praktiski visi precesijas leņķi α_H, kas ir mazāk par aptuveni 0.4 radiāniem ir nestabili, kas plakanai viendomēna daļiņai ar plaknes anizotropiju ir fizikāli pieņemams rezultāts,
4. Eksistē punkti, kuros nav iespējams aprēķināt sinhrona režīma nosacījumu vektoram $\vec{n}$, kas liecina vai nu par alternatīvas pieejas nepieciešamību sākuma nosacījumu aprēķinam, vai arī to, ka fāzu telpā eksistē reģioni, kuros sinhrons režīms nav iespējams.
5. Skaitliski meklētās fāzu diagrammas pie lielām parametra H_a/H vērtībām nesakrīt ar sagaidāmajām literatūrā atrastajām paramagnētiskas daļiņas fāzu diagrammām, kas norāda, ka nepieciešama dziļāka temata izpēte.
6. Darbs ir pieredzes bāze ar veiksmīgām un neveiksmīgām pieejām, idejām, risinājumiem un implementācijām, kas var būt vērtīgas turpmākajiem tēmas pētījumiem.

6. PERSPEKTĪVAS

1. Lai varētu izpētīt fāzu telpas reģionus, kuros ir liels metastabilo punktu blīvums, izveidotais rīks būtu jāoptimizē uz veiktspēju un laikietilpību.
2. Lai varētu iegūt pārliecinošus rezultātus, būtu jāatrod veids kā aprēķināt vektora $\vec{n}$ sākuma nosacījumus pie jebkuras parametru kombinācijas vai arī jāpierāda, ka eksistē parametru kombinācijas pie kurām sinhrons režīms nav iespējams. Tas būtu nepieciešams, lai varētu iegūt informāciju par punktiem, kurus līdz šim nav vispār izdevies apskatīt.

7. LITERATŪRAS SARAKSTS

1. Bin Ren, Assembly Behavior of Iron Oxide-Capped Janus Particles in a Magnetic Field (doi.org/10.1021/la203969f)
2. J. Yan, Rotating crystals of magnetic Janus colloids (doi.org/10.1039/C4SM01962H)
3. Baohu Dai, Programmable artificial phototactic microswimmers (doi.org/10.1038/nnano.2016.187)
4. S. Yoshida, Dynamic structural transformation of self-assembled Janus hydrogel microparticles under periodically-changed magnetic field(doi.org/10.1109/TRANSDUCERS.2015.7181161)
5. Yujin Seong, Magnetic interaction of Janus particles suspended in a viscous fluid(doi.org/10.1103/PhysRevE.93.022607)
6. Hui Eun Kim, Numerical investigation of the dynamics of Janus magnetic particles in a rotating magnetic field(doi.org/10.1007/s13367-017-0003-5)
7. Jing Yan, Linking synchronization to self-assembly using magnetic Janus colloids (doi.org/10.1038/nature11619)
8. Jānis Cīmurs,Three-Dimensional Dynamics of a Particle With a Finite Energy of Magnetic Anisotropy in a Rotating Magnetic Field (doi.org/10.1103/PhysRevE.88.062315)
9. Amikam Aharoni, Introduction to the Theory of Ferromagnetsm, 2nd ed., Oxford University Press
10. Amikam Aharoni, introduction to the Theory of Ferromagnetism,2nd ed., Oxford University Press, fig 1.1
11. Amikam Aharoni, introduction to the Theory of Ferromagnetism, 2nd ed., Oxford University Press
12. Feynman Lectures on Physics Vol. II, sec 37-1
13. Ramesh B. Kamble, Domain Size correlated magnetic properties and electrical impedance of size independant nicklel ferrite nanoparticles (http://dx.doi.org/10.1063/1.4906101)
14. Amikam Aharoni, introduction to the Theory of Ferromagnetism , 2nd ed., Oxford University Press, 5.1, p85
15. Amikam Aharoni, introduction to the Theory of Ferromagnetism , 2nd ed., Oxford University Press, 6.1.3, p114
16. J. Cīmurs, Viendomēna un superparamagnētiskas daļiņas dinamika mainīgos magnētiskajos laukos, sec 3.2 eq 3.1
17. Amikam Aharoni, introduction to the Theory of Ferromagnetism, 2nd ed., Oxford University Press, 6.1.3, eq 6.1.22
18. J. Cīmurs, Viendomēna un superparamagnētiskas daļiņas dinamika mainīgos magnētiskajos laukos, sec 1.3.2 eq 1.9, 1.10
19. J. Cīmurs, Viendomēna un superparamagnētiskas daļiņas dinamika mainīgos magnētiskajos laukos, sec 1.4
20. Amikam Aharoni, introduction to the Theory of Ferromagnetism, 2nd ed.,Oxford University Press, sec 5.4
21. J. Cīmurs, Viendomēna un superparamagnētiskas daļiņas dinamika mainīgos magnētiskajos laukos, sec 3.2 eq 3.1

22. J. Cīmurs, Viendomēna un superparamagnētiskas daļiņas dinamika mainīgos magnētiskajos laukos, sec 2.2
23. J. Cīmurs, Viendomēna un superparamagnētiskas daļiņas dinamika mainīgos magnētiskajos laukos, sec 1.5.2 eq 1.30
24. SciPy.org Nonlinear solvers (https://docs.scipy.org/doc/scipy/reference/optimize.nonlin.html)
25. J. Cīmurs, Viendomēna un superparamagnētiskas daļiņas dinamika mainīgos magnētiskajos laukos, sec 3.5, fig 3.2

8. PIELIKUMI

8.1 Programma valodā Python ar kuru tika rēķinātas inviduālās simulācijas pie fiksētām parametru vērtībām:

```
1. import numpy as np
2. import matplotlib as mpl
3. from mpl_toolkits.mplot3d import Axes3D
4. import sys, matplotlib.pylab as plt
5. import time
6. from scipy.optimize import newton_krylov, anderson, broyden1, broyden2, exc
   itingmixing, linearmixing, diagbroyden
7.
8.
9. def modulis(vect):
10.
11.     return np.sqrt(vect[0]**2+vect[1]**2+vect[2]**2)
12.
13. #Laika solis
14. dt = 0.008
15. Pi = 3.14159265
16. #Magnētiskā lauka pilns precesijas periods ir Tau:
17. Tau = 1
18. #omega_h definīcija, izmantojot lauka precesijas periodu.
19. w_h = 2*Pi/Tau
20. endlaiks = 5
21. laiksoli = int(endlaiks/dt)
22. laikskala = np.linspace(0,endlaiks,laiksoli)
23. angle_h = 0
24. #visas 'range' vērtības ir tas, cik smalku režģi veidot virsmai.
25. omegarange = 10
26. Hrange = 10
27. alfarange = 6
28. omega_arr = np.linspace(0.1,2,omegarange)
29. #
30. #Masīvi, kuros glabājas 'range' skaits vienību, vienmērīgi izlīdzinātas nep
   ieciešamajos vērtību intervālos.
31. att_arr = 1/(omega_arr)
32. HHa_arr = np.linspace(0.1,2,Hrange)
33. Harr = 1/HHa_arr
34. alfa_harr = np.linspace(0.1*Pi/180,90*Pi/180,alfarange)
35. #Magnētiskā lauka
36. w = np.array([0,0,1])
37. total = omegarange*Hrange*alfarange
38.
39. save = np.array([ [0,0,0,0],[0,0,0,0] ])
40.
41. def Fi(arr, hArr):
42.     modarr = np.sqrt(arr[0]**2 + arr[1]**2 + arr[2]**2)
43.     modhArr = np.sqrt(hArr[0]**2 + hArr[1]**2 + hArr[2]**2)
44.
45.     return np.arccos( np.round_(np.dot(arr/modarr,hArr/modhArr),4) )
46.
47. def Teta(fi):
48.     #4. kārtas vienādojuma koeficienti izteikti ar zināmiem lielumiem.
49.     J = -H
50.     K = np.cos(fi)
51.     L = (np.sin(fi))**2
52.     p = np.array([-J**2 , -2*J*K , J**2-1 , 2*J*K , K**2])
```

```
    #Sakņu atrašana
    saknes = np.roots(p)
    rsaknes = np.round_(saknes[np.isreal(saknes)].astype(float),4)
    #Inversā substitūcija
    lenki = np.arccos(rsaknes)
    #Ievietošana enerģijas vienādojumā, lai salīdzinātu
    energijas = -0.5 * H * (np.cos(lenki))**2 - np.cos(fi-lenki)
    return lenki[np.argmin(energijas)]

def hlauks(alfa_h,moment):
    hx = np.cos(angle_h + w_h * moment) * np.sin(alfa_h)
    hy = np.sin(angle_h + w_h * moment) * np.sin(alfa_h)
    hz = np.cos(alfa_h)
    return np.array([hx,hy,hz])/np.sqrt(hx**2+hy**2+hz**2)

def probe(x,pertmod):

    if (x > 1.05*pertmod):
        koef = 1
        print("UNSTABLE")
    else:
        if(( 0.95*pertmod <= x <= 1.05*pertmod ) == True):
            koef = 0
            print("METASTABLE")
        else:
            koef =-1
            print("STABLE")
    return koef


def k(n, h):
    FiCos = np.cos(Fi(n, h))
    tetaCos = np.cos(Teta(FiCos))
    w_a = w_h/att
    return (w_a*tetaCos**2/(FiCos + H*tetaCos))*(h-n*FiCos)

ninit = np.array([0,0,1])
#
def Funk(n):
    h = hlauks(alfa_h,0)
    p = np.cross(w,n)

    return p-( k(n,h)/w_h )


def AdVictoriam(H,att,alfa_h):

        laiks = 0
        nd = np.zeros([laiksoli,3,])
        ndprim = np.zeros([laiksoli,3,])
        print("     ")
        Failed = False
        for find in range(0,3):
            ninit = hlauks(alfa_h,find*Tau*0.9/3)
            try:
                nvec = newton_krylov(Funk,ninit)
                print("Attempting Newton - Krylov Method")
            except:
                try:
                    nvec = anderson(Funk,ninit)
```

```
                    print("Newton -
    Krylov method did not converge, attempting to find initial conditions usin
    g Anderson mixing method,")
                except:
                    try:
                        nvec = broyden1(Funk,ninit)
                        print("Attempting Broyden`s first Jacobian approx
    imation,")
                    except:
                        try:
                            nvec = broyden2(Funk,ninit)
                            print("Attempting Broyden`s second Jacobian a
    pproximation,")
                        except:
                            try:
                                nvec = excitingmixing(Funk,ninit)
                                print("Attempting to use tuned diagonal J
    acobian approximation,")
                            except:
                                try:
                                    nvec = linearmixing(Funk,ninit)
                                    print("Attempting to use a scalar Jac
    obian approxmation,")
                                except:
                                    try:
                                        nvec = diagbroyden(Funk,ninit)
                                        print("Attempting Broyden Jacobia
    n approximation")
                                    except:
                                        Failed = True
            if (Failed==False):
                break
        if (Failed == True):
            print("ALL ATTEMPTED METHODS FAILED, POINT FLAGGED AS A NON -
     CONVERGENT FOR INITIAL CONDITIONS")
            return np.array([H,att,alfa_h,2])
        print("NVEC:")
        print(nvec)
        print(np.sqrt(nvec[0]**2+nvec[1]**2+nvec[2]**2))

        nvec = nvec/modulis(nvec)
        pert = np.cross(-1*w,hlauks(alfa_h,0))/100
        ndnew = nvec
        nd[0] = ndnew/np.sqrt( ndnew[0]**2+ndnew[1]**2+ndnew[2]**2 )
        ndnewprim = nvec + pert/np.sqrt(pert[0]**2+pert[1]**2+pert[2]**2)

        ndprim[0] = ndnewprim/np.sqrt(ndnewprim[0]**2+ndnewprim[1]**2+ndn
    ewprim[2]**2)

    #  G   A   L   V   E   N   A   I   S       C   I   K   L   S (NEPERTUR
    BĒTS) :
        nulltime = time.time()
        for i in range(0,laiksoli - 1):
            k1 = k(nd[i], hlauks(alfa_h,laiks))
            k2 = k(nd[i] + k1/2, hlauks(alfa_h,laiks+dt/2))
            k3 = k(nd[i] + k2/2, hlauks(alfa_h,laiks+dt/2))
            k4 = k(nd[i] + k3, hlauks(alfa_h,laiks+dt))

            nd[i+1] = nd[i] + dt*(k1 + 2*(k2 + k3) + k4)/6
            nd[i+1] = nd[i+1]/modulis(nd[i+1])
            laiks = laiks + dt
```

```
163.      #  S  E  K  U  N  D  Ā  R  A  I  S     C  I  K  L  S (PERTRUBĒTS) :
164.          laiks = 0
165.          for i in range(0,laiksoli - 1):
166.              k1 = k(ndprim[i], hlauks(alfa_h,laiks))
167.              k2 = k(ndprim[i] + k1/2, hlauks(alfa_h,laiks+dt/2))
168.              k3 = k(ndprim[i] + k2/2, hlauks(alfa_h,laiks+dt/2))
169.              k4 = k(ndprim[i] + k3, hlauks(alfa_h,laiks+dt))
170.
171.              ndprim[i+1] = ndprim[i] + dt*(k1 + 2*(k2 + k3) + k4)/6
172.              ndprim[i+1] = ndprim[i+1]/modulis(ndprim[i+1])
173.              laiks = laiks + dt
174.          runtime = time.time()
175.          #Starpība starp abiem vektoriem
176.          dev = ndprim - nd
177.          mod_dev = np.sqrt(dev[:,0]**2+dev[:,1]**2+dev[:,2]**2)
178.          probe(mod_dev[laiksoli-1],modulis(pert))
179.          #Informatīvi, lai skatītos vai viss nav salūzis:
180.          print("nd[i]:")
181.          print(nd[i])
182.          print("mod nd[i]:")
183.          print(np.sqrt(nd[i][0]**2+nd[i][1]**2+nd[i][2]**2))
184.          print("ndprim:")
185.          print(ndprim[i])
186.          print("mod ndprim[i]:")
187.          print(np.sqrt(ndprim[i][0]**2+ndprim[i][1]**2+ndprim[i][2]**2))
188.          print("pavaditais laiks:")
189.          print(runtime-nulltime)
190.
191.          return nd,ndprim
192.      #Šeit jāizvēlās trīs parametru vērtības
193.      H,att,alfa_h = 2.        ,  0.6       ,  0.38
194.      strh = str(H)
195.      stra = str(att)
196.      strah = str(alfa_h)
197.      rezultati = AdVictoriam(H,att,alfa_h)
198.      a = rezultati[0]
199.      b = rezultati[1]
200.      # 3 D   P L O T
201.      #
202.      mpl.rcParams['legend.fontsize'] = 10
203.      fig = plt.figure()
204.      fulltitle = str("Ha/H=" + strh + ", w_h/w_a=" +  stra + ", alfa_h=" +
      strah + ", \ndt="+ str(dt))
205.      ax = fig.gca(projection='3d')
206.      plt.title(fulltitle)
207.      ax.plot(a[:,0], a[:,1], a[:,2], label='Time evolution of the unpertur
     bed vector')
208.      ax.plot(b[:,0], b[:,1], b[:,2], label='Time evolution of the perturbe
     d vector')
209.      ax.legend()
210.
211.      ax.set_xlabel('\nX axis')
212.      ax.set_ylabel('\nY axis')
213.      ax.set_zlabel('\nZ axis')
214.      plt.show()
```

8.2 Programma valodā Python ar kuru tika rēķinātas fāzu diagrammas

```
1. import numpy as np
2. import sys, matplotlib.pylab as plt
3. import time
4. from scipy.optimize import newton_krylov, anderson, broyden1, broyden2, exc
   itingmixing, linearmixing, diagbroyden
5.
6.
7.
8. def modulis(vect):
9.
10.     return np.sqrt(vect[0]**2+vect[1]**2+vect[2]**2)
11.
12.
13. dt = 0.005
14. Pi = 3.14159265
15. Tau = 1
16. w_h = 2*Pi/Tau
17. endlaiks = 4*Tau
18. laiksoli = int(endlaiks/dt)
19. laikskala = np.linspace(0,endlaiks,laiksoli)
20. angle_h = 0
21. attrange = 20
22. Hrange = 3
23. alfarange = 10
24. #
25. att_arr = np.linspace(0.1,2,attrange)
26. Harr = np.linspace(0.1,2,Hrange)
27. alfa_harr = np.linspace(0.1*Pi/180,90*Pi/180,alfarange)
28. w = np.array([0,0,1])
29. total = attrange*Hrange*alfarange
30. save = np.array([ [0,0,0,0],[0,0,0,0] ])
31. failed = np.array([ [0,0,0,0],[0,0,0,0] ])
32.
33. def Fi(arr, hArr):
34.     modarr = np.sqrt(arr[0]**2 + arr[1]**2 + arr[2]**2)
35.     modhArr = np.sqrt(hArr[0]**2 + hArr[1]**2 + hArr[2]**2)
36.
37.     return np.arccos( np.round_(np.dot(arr/modarr,hArr/modhArr),4) )
38. # ##########################################################################
   #####
39. def Teta(fi):
40.     #4. kārtas vienādojuma koeficienti izteikti ar zināmiem lielumiem.
41.     J = -H
42.     K = np.cos(fi)
43.     L = (np.sin(fi))**2
44.     p = np.array([-J**2 , -2*J*K , J**2-1 , 2*J*K , K**2])
45.     #Sakņu atrašana
46.     saknes = np.roots(p)
47.     rsaknes = np.round_(saknes[np.isreal(saknes)].astype(float),4)
48.     #Inversā substitūcija
49.     lenki = np.arccos(rsaknes)
50.     energijas = -0.5 * H * (np.cos(lenki))**2 + np.cos(fi-lenki)
51.     return lenki[np.argmin(energijas)]
52.
53. def hlauks(alfa_h,moment):
54.     hx = np.cos(angle_h + w_h * moment) * np.sin(alfa_h)
```

```
    hy = np.sin(angle_h + w_h * moment) * np.sin(alfa_h)
    hz = np.cos(alfa_h)
    return np.array([hx,hy,hz])/np.sqrt(hx**2+hy**2+hz**2)

def probe(x,pertmod):

    if (x > 1.05*pertmod):
        koef = 1
        print("UNSTABLE")
    else:
        if(( 0.95*pertmod <= x <= 1.05*pertmod ) == True):
            koef = 0
            print("METASTABLE")
        else:
            koef =-1
            print("STABLE")
    return koef


def k(n, h):
    FiCos = np.cos(Fi(n, h))
    tetaCos = np.cos(Teta(FiCos))
    w_a = w_h/att
    return (w_a*tetaCos**2/(FiCos + H*tetaCos))*(h-n*FiCos)
ninit = hlauks(alfa_harr[0],0)
#
def Funk(n):
    h = hlauks(alfa_h,0)
    p = np.cross(w,n)

    return p-( k(n,h)/w_h )


global H
global att
global alfa_h

def AdVictoriam(H,att,alfa_h):
    print(" ")
    print(" ")
    print("POINT  [ H_a / H  :  w_h / w_a  :  prec.angle (rad) ]  : ")
    print(H,att,alfa_h)
    laiks = 0
    nd = np.zeros([laiksoli,3,])
    ndprim = np.zeros([laiksoli,3,])
          Failed = False
          for find in range(0,3):
              ninit = hlauks(alfa_h,find*Tau*0.9/3)
              try:
                  nvec = newton_krylov(Funk,ninit)
                  print("Attempting Newton - Krylov Method")
              except:
                  try:
                      nvec = anderson(Funk,ninit)
                      print("Newton -
    Krylov method did not converge, attempting to find initial conditions usin
   g Anderson mixing method,")
                  except:
                      try:
                          nvec = broyden1(Funk,ninit)
```

```
                            print("Attempting Broyden`s first Jacobian approx
imation,")
                        except:
                            try:
                                nvec = broyden2(Funk,ninit)
                                print("Attempting Broyden`s second Jacobian a
pproximation,")
                            except:
                                try:
                                    nvec = excitingmixing(Funk,ninit)
                                    print("Attempting to use tuned diagonal J
acobian approximation,")
                                except:
                                    try:
                                        nvec = linearmixing(Funk,ninit)
                                        print("Attempting to use a scalar Jac
obian approxmation,")
                                    except:
                                        try:
                                            nvec = diagbroyden(Funk,ninit)
                                            print("Attempting Broyden Jacobia
n approximation")
                                        except:
                                            Failed = True
                if (Failed==False):
                    break
            if (Failed == True):
                print("ALL ATTEMPTED METHODS FAILED, POINT FLAGGED AS A NON -
 CONVERGENT FOR INITIAL CONDITIONS")
                return np.array([H,att,alfa_h,2])
            print("NVEC:")
            print(nvec)
            print(np.sqrt(nvec[0]**2+nvec[1]**2+nvec[2]**2))

            pert = np.cross(-1*w,hlauks(alfa_h,laiks))/50
            ndnew = nvec
            nd[0] = ndnew/np.sqrt( ndnew[0]**2+ndnew[1]**2+ndnew[2]**2 )
            ndnewprim = nvec + pert/np.sqrt(pert[0]**2+pert[1]**2+pert[2]**2)

            ndprim[0] = ndnewprim/np.sqrt(ndnewprim[0]**2+ndnewprim[1]**2+ndn
ewprim[2]**2)

        # G   A   L   V   E   N   A   I   S       C   I   K   L   S (NEPERTUR
BĒTS) :
            nulltime = time.time()
            for i in range(0,laiksoli - 1):
                k1 = k(nd[i], hlauks(alfa_h,laiks))
                k2 = k(nd[i] + k1/2, hlauks(alfa_h,laiks+dt/2))
                k3 = k(nd[i] + k2/2, hlauks(alfa_h,laiks+dt/2))
                k4 = k(nd[i] + k3, hlauks(alfa_h,laiks+dt))

                nd[i+1] = nd[i] + dt*(k1 + 2*(k2 + k3) + k4)/6
                nd[i+1] = nd[i+1]/modulis(nd[i+1])
                laiks = laiks + dt
        #  S  E  K  U  N  D  Ā  R  A  I  S    C  I  K  L  S (PERTRUBĒTS) :
            laiks = 0
            for i in range(0,laiksoli - 1):
                k1 = k(ndprim[i], hlauks(alfa_h,laiks))
                k2 = k(ndprim[i] + k1/2, hlauks(alfa_h,laiks+dt/2))
                k3 = k(ndprim[i] + k2/2, hlauks(alfa_h,laiks+dt/2))
                k4 = k(ndprim[i] + k3, hlauks(alfa_h,laiks+dt))
```

```
165.
166.                 ndprim[i+1] = ndprim[i] + dt*(k1 + 2*(k2 + k3) + k4)/6
167.                 ndprim[i+1] = ndprim[i+1]/modulis(ndprim[i+1])
168.                 laiks = laiks + dt
169.             runtime = time.time()
170.             dev = ndprim - nd
171.             print("nd[i]:")
172.             print(nd[i])
173.             print("mod nd[i]:")
174.             print(np.sqrt(nd[i][0]**2+nd[i][1]**2+nd[i][2]**2))
175.             print("ndprim:")
176.             print(ndprim[i])
177.             print("mod ndprim[i]:")
178.             print(np.sqrt(ndprim[i][0]**2+ndprim[i][1]**2+ndprim[i][2]**2))
179.             mod_dev = np.sqrt(dev[:,0]**2+dev[:,1]**2+dev[:,2]**2)
180.             print("pavaditais laiks:")
181.             print(runtime-nulltime)
182.             stability = probe(mod_dev[laiksoli - 1],modulis(pert))
183.             return np.array([H,att,alfa_h,stability])
184.
185.         Harr = np.array([0.5,1.01,2])
186.         skaits = 0
187.         total = len(Harr)*attrange*alfarange
188.         gridnum = 0
189.         grid = 2*np.zeros([total,4])
190.         for H in Harr:
191.             for att in att_arr:
192.                 for j in range(1,alfarange):
193.                     alfa_h = alfa_harr[j-1]
194.                     a = alfa_h
195.                     fa = AdVictoriam(H,att,alfa_h)[3]
196.                     alfa_h = alfa_harr[j]
197.                     b = alfa_h
198.                     fb = AdVictoriam(H,att,alfa_h)[3]
199.                     skaits = skaits+1
200.                     print("PROGRESS:")
201.                     print(skaits, " / ",total)
202.                     approx=(a+b)/2
203.                     if fb==0:
204.                         approx = b
205.                         a = (alfa_harr[j]-alfa_harr[j-1])/2
206.                         b = (alfa_harr[j]+alfa_harr[j-1])/2
207.                         save = np.concatenate(( save, np.array([[H, att, appr
    ox, np.absolute((a-b)/2)]]) ), axis=0 )
208.                     elif fb==2:
209.                         failed = np.concatenate(( failed, np.array([[H, att,
    approx, 999]]) ), axis=0 )
210.                         continue
211.                     if fa==0:
212.                         approx = a
213.                         a = (alfa_harr[j]-alfa_harr[j-1])/2
214.                         b = (alfa_harr[j]+alfa_harr[j-1])/2
215.                         save = np.concatenate(( save, np.array([[H, att, appr
    ox, np.absolute((a-b)/2)]]) ), axis=0 )
216.                     elif fa==2:
217.                         failed = np.concatenate(( failed, np.array([[H, att,a
    , 999]]) ), axis=0 )
218.                         continue
219.                     gridwrite = np.array([[H,att,a,fa]])
220.                     grid = np.concatenate((grid,gridwrite),axis=0)
221.                     gridwrite = np.array([[H,att,b,fb]])
```

```
                    grid = np.concatenate((grid,gridwrite),axis=0)
                    if (fb != fa):
                        c = (a+b)/2
                        alfa_h = c
                        fc = AdVictoriam(H,att,alfa_h)[3]
                        if (fc == 0):
                            a = a + 0.5*a
                            b = b + 0.5*b
                            approx = c
                            save = np.concatenate(( save, np.array([[H, att, approx, np.absolute((a-b)/2)]]) ), axis=0 )
                        elif fc==2:
                            failed = np.concatenate(( failed, np.array([[H, att,c, 999]]) ), axis=0 )
                            continue
                        gridwrite = np.array([[H,att,c,fc]])
                        grid = np.concatenate((grid,gridwrite),axis=0)
                        if(fa != fc):
                            d = (a+c)/2
                            alfa_h = d
                            fd = AdVictoriam(H,att,alfa_h)[3]
                            if (fd==0):
                                a = a + 0.5*a
                                c = c + 0.5*c
                                approx = d
                                save = np.concatenate(( save, np.array([[H, att, approx, np.absolute((a-b)/2)]]) ), axis=0 )
                            elif fd==2:
                                failed = np.concatenate(( failed, np.array([[H, att, d, 999]]) ), axis=0 )
                                continue
                            gridwrite = np.array([[H,att,d,fd]])
                            grid = np.concatenate((grid,gridwrite),axis=0)
                            if (fa != fd):
                                a = a
                                b = d
                                approx = (a+d)/2
                                save = np.concatenate(( save, np.array([[H, att, approx, np.absolute((a-b)/2)]]) ), axis=0 )

                            elif(fd != fc):
                                a = d
                                b = c
                                approx = (d+c)/2
                                save = np.concatenate(( save, np.array([[H, att, approx, np.absolute((a-b)/2)]]) ), axis=0 )
                            #Absolūtais izņēmuma gadījums. Punkts, kam bija jābūt starp a un b vnk pazudis.
                            else:
                                a = (a+d)/2
                                b = (d+c)/2
                                approx = d
                                print("K A U T _ K A S _ S A L U Z A _ P I E :")
                                print("H_a/H \ att \ alfa_h approx \ delta alfa_h :")
                                print(H,att,approx,np.absolute((a-b)/2))
                                save = np.concatenate(( save, np.array([[H, att, approx, np.absolute((a-b)/2)]]) ), axis=0 )
                                continue
                    else:
```

```
                            e = (b+c)/2
                            alfa_h = e
                            fe = AdVictoriam(H,att,alfa_h)[3]
                            if (fe == 0):
                                c = c + 0.5*c
                                b = b + 0.5*b
                                save = np.concatenate(( save, np.array([[H, a
tt, approx, np.absolute((a-b)/2)]]) ), axis=0 )
                            elif fe==2:
                                failed = np.concatenate(( failed, np.array([[
H, att,e,999]]) ), axis=0 )
                                continue
                            gridwrite = np.array([[H,att,e,fe]])
                            grid = np.concatenate((grid,gridwrite),axis=0)
                            if(fc != fe):
                                a = c
                                b = e
                                approx = (c+e)/2
                                save = np.concatenate(( save, np.array([[H, a
tt, approx, np.absolute((a-b)/2)]]) ), axis=0 )
                            elif( fe != fb ):
                                a = e
                                b = b
                                approx = (e+b)/2
                                save = np.concatenate(( save, np.array([[H, a
tt, approx, np.absolute((a-b)/2)]]) ), axis=0 )
      #Absolūtais izņēmuma gadījums. Punkts, kam bija jābūt starp a un b vn
k pazudis.
                            else:
                                a = (c+e)/2
                                b = (e+b)/2
                                approx = e
                                print("K A U T _ K A S _ S A L U Z A _ P I E:
")
                                print( "H_a/H \ att \ alfa_h approx \ delta a
lfa_h :" )
                                print(H,att,approx,np.absolute((a-b)/2))
                                save = np.concatenate(( save, np.array([[H, a
tt, approx, np.absolute((a-b)/2)]]) ), axis=0 )
                                continue
                    else:
                        print("  ")
                        print( " PABEIGTA CILPA PIE H_a/H \ att \ alfa_h appr
ox \ delta alfa_h :" )
                        print(H, att, alfa_harr[j], np.absolute((a-b)/2))
                        print("J, J-1")
                        print(j,j-1)

```

Bakalaura darbs „Feromagnētiskas viendomēna daļiņas, ar plaknes anizotropiju, dinamika precesējošā magnētiskā laukā” izstrādāts LU Fizikas un matemātikas fakultātē.

Ar savu parakstu apliecinu, ka pētījums veikts patstāvīgi, izmantoti tikai tajā norādītie informācijas avoti un iesniegtā darba elektroniskā kopija atbilst izdrukai.

Autors Oskars Sjomkāns

Rekomendēju darbu aizstāvēšanai

Vadītājs Dr.Phys. Jānis Cīmurs 26.06.2017

Recenzents: Dr.Phys Imants Kaldre

Darbs iesniegts Fizikas nodaļā __.06.2017.

Dekāna pilnvarotā persona: vecākā metodiķe Dzintra Holsta

Darbs aizstāvēts bakalaura gala pārbaudījuma komisijas sēdē

____ 06.2017. prot. Nr._____

Komisijas sekretārs: