Tartu Ülikool

Füüsika Instituut

Tartu Observatoorium

Sander Kõiv

Trigonomeetriliste summade kiire arvutamise meetodid astronoomiliste aegridade sagedusanalüüsis

Bakalaureusetöö

Juhendaja: Jaan Pelt

Tartu 2014

Sisukord

Sissejuhatus....3

Ülevaade....4

Algoritmi ajaline keerukus....4
Lomb-Scargle Periodogramm....5
Ebaühtlase ajasammuga kiire Fourier' teisendus....6
NFFT3....8
Teegi iseärasused....8
Teegis kasutatavate algoritmide ajaline keerukus....9
Metoodika kirjeldus....11

Periodogrammi teisendused....11
NFFT3 teegi testimine....12
Periodogrammi arvutamine....13
Erinevad meetodid....13
Täpsuse võrdlus....14
Ajaline võrdlus....14
Tulemused....16

NFFT3 katsed....16
Graafik1 Spektri reaalosa....16
Graafik2 Spektri imaginaarosa....17
Graafik3 Spektri mooduli ruut....17
Tabel1 Kiire ja aeglase teisenduse võrdlus....18
Graafik4 Kiire ja aeglase teisenduse võrdlus....18
Periodogrammi katsed....19
Tabel2 Periodogrammi arvutamise täpsuse võrdlus....19
Tabel3 Periodogrammi arvutamise ajaline võrdlus....19
Graafik5 Periodogrammi arvutamise ajaline võrdlus....20
Graafik6 LQ Hya periodogramm....20
Järeldused....21

NFFT3 teegi testimine....21
Periodogrammi arvutamine....21
Täpsuse võrdlus....21
Ajaline võrdlus....21
LQ Hya....22
Kokkuvõte....23

Viited....24

Summary....25

Lisad....26

Lisa1 Lähtekood....26

Sissejuhatus

Astronoomias valitseb olukord, kus on kogutud palju andmeid, mida keegi pole jõudnud töödelda ega analüüsida. Ideaalsel juhul oleks uuritav suurus antud pideva ajast sõltuva funktsioonina, kuid mistahes päris elus läbiviidavaid mõõtmisi ei saa teha pidevalt, vaid peab tegema teatud ajavahemiku tagant. Arvestades seda, et Maa pinnalt astronoomilisi mõõtmisi tehes pole tihti erinevatest asjaoludest sõltuvalt võimalik koguda andmeid kindla ajavahemiku tagant, on paljud kogutud andmetest sellised, kus tähte iseloomustav füüsikaline suurus on mõõdetud ebaühtlase ajalise sammuga. Sellest töös keskedutaksegi selliste andmete töötlemisele.

Üks meetod, mille abil ebaühtlase ajasammuga andmetes peituvaid sagedusi uuritakse on Lomb-Scargle periodogrammi arvutamine ja selle statistilise käitumise uurimine[1][2][3]. Antud töös keskendutakse Lomb-Scargle periodogrammi arvutamisele, kus lähteandmed on ebaühtlase sammuga, kuid arvutatava periodogrammi samm on ühtlane. Erilist tähelepanu pööratakse periodogrammi arvutamiseks kuluvale ajale.

Aegridate töötlemisel on levinud tööriistaks Fourier' teisendus ning antud töös kasutatakse selle arvutamiseks Lübecki Ülikooli, Osnabrücki Ülikooli ja Chemnitzi Tehnoloogia Ülikooli koostööl loodud teeki NFFT3[4]. Kõigepealt uuritakse ebaühtlase ajasammuga Fourier' teisenduse kiiret arvutamist olemasoleva teegi abil ning seejärel kasutatakse kiiret teisendust Lomb-Scargle periodogrammi arvutamiseks. Näidatakse ka seda, et kiire algoritmi kasutamisest tingitud täpsuskadude suurusjärk on võrreldav ujuvkoma arvude kasutamisest tuleneva arvutusveaga.

Arvutustäpsuse kontrollimiseks kasutatakse juhuslikult genereeritud arve ning illustreerimaks meetodi sobilikkust astronoomiliste aegridade sagedusanalüüsiks tehakse üks arvutus ka reaalseid mõõtmistulemusi kasutades. Mõõtmistulemusteks on Arizonas Fairborni Observatooriumis APT*(Automatic Photoelectric Telsecope)* abil 24 aasta jooksul kogutud andmed. Antud töös arvutatud tulemust võrreldakse eelnevalt avaldatud tulemusega[5].

Ülevaade

Antud töös uuritakse probleemi, kus suvalistel ajahetkedel t_n on mõõdetud reaalarvulist füüsikalist suurust X ja saadud väärtused: $f(t_n)=X_n$, kus $n\in\mathbb{N}$ ja $0\leqslant n<N_0$. Olgu kogu mõõtmiste periood: $T=t_{N_0-1}-t_0$. Antud töös eeldatakse, et mõõtmispunktide arv sõltub ligikaudu lineaarselt mõõtmisperioodi pikkusest $N_0\sim T$. Mõõtmistulemustele vastavat periodogrammi uuritakse diskreetsetel sageduse väärtustel $P_X(2\pi m)$, kus $m\in\mathbb{N}, 0\leqslant m<M_0$. Siinjuures m näitab mitu võnget toimub kogu mõõtmisperioodi T kestel.

Ühtlase sammuga diskreetne funktsioon on tähistatud muutujaga nurksulgudes, ebaühtlase sammuga diskreetne funktsioon muutujaga, millel on alaindeks ja pidev funktsioon muutujaga ümarsulgudes.

Selleks, et arvutada periodogrammi väärtusi parameetri $\omega\in\{2\pi m\}$ väärtustel, kus $m\in\mathbb{N}$ on vaja teada mõningaid summasi. Selles töös kasutatakse tähistust:

$$C_X=C_X[m]=\sum_n X_n\cos(2\pi m t_n)$$
$$S_X=S_X[m]=\sum_n X_n\sin(2\pi m t_n)$$
$$C_2=C_2[m]=\sum_n \cos(2\cdot 2\pi m t_n)$$
$$S_2=S_2[m]=\sum_n \sin(2\cdot 2\pi m t_n).$$

Algoritmi ajaline keerukus

Informaatikas kasutatakse algoritmide kiiruslike omaduste hindamiseks ajalise keerukuse mõistet. Sellest on täpsemalt kirjutanud Kiho[6]. Ajalist keerukust iseloomustab funktsioon $f_o(n_o)$, kus n_o on ülesande lähteandmete hulk ja funktsiooni väärtuseks on ülesande lahendamiseks kuluvate sammude arv. Olgu ühe sammu sooritamiseks kuluv aeg t_o, siis kogu ülesande lahendamiseks kulub aeg $t_o f_o(n_o)$. Lisaks ajalise keerukuse funktsioonile saab rääkida ajalise keerukuse asümptootilisest hinnangust, mida tähistatakse O - relatsiooniga. Definitsiooni kohaselt funktsioon $f_o(n_o)$ on asümptootilise hinnanguga $O(g_o(n_o))$, kui leiduvad sellised $c_o>0$ ja $N_o>0$, et $|f_o(n_o)|\leqslant c_o|g_o(n_o)|$ iga $n_0\geqslant N_o$ korral. Öeldes, et f_o on $O(g_o)$, tähendab see sisuliselt seda, et funktsiooni f_o kasv on tõkestatud funktsiooniga g_o ülevalt poolt. Lisaks O -relatsioonile kasutatakse ka Θ -relatsiooni. Definitsiooni kohaselt: f_o on $O(g_o)$ ja g_o on

$O(f_o)$, parajasti siis kui f_o on $\Theta(g_o)$. Täpsemalt tähendab see seda, et piisavalt suure funktsiooni argumendi korral $f/g<c_o$ ja $g/f<c_o'$. Sellest võib järeldada, et kui f_o on $\Theta(g_o)$, siis ka g_o on $\Theta(f_o)$. Öeldes, et f_o on $\Theta(g_o)$ tähendab see sisuliselt seda, et funktsiooni f_o kasv on nii ülevalt kui ka alt poolt tõkestatud funktsiooniga g_o.

Antud töö kontekstis olulised O ja Θ -relatsioonide kohta käivad omadused:

- Iga $k_o>0$ korral, $k_o f_o$ on $O(f_o)$.
- Kui f_o on $O(g_o)$ ja h_o on $O(g_o)$, siis (f_o+h_o) on $O(g_o)$.
- Kui f_o on $O(g_o)$ ja g_o on $O(h_o)$, siis f_o on $O(h_o)$.
- $\Theta(\log_{b_o}(n_o))$ on $\Theta(\log_{d_o}(n_o))$ iga $b_o, d_o>1$ korral.

Θ -relatsiooni abil saab hinnata algoritmi ajalise keerukuse klassi. Algoritmid, mille ajalise keerukuse funktsioon on kirjeldatav sama Θ -relatsiooni abil, kuuluvad samasse ajalise keerukuse klassi. Välja toodud relatsioonide omaduste põhjal võib järeldada, et ajalise keerukuse klassist rääkides võib konstandid ära jätta. Lisaks, kui algoritmi lahendamiseks on vaja sooritada üksteise järel $\Theta(n_o)$ ja $\Theta(n_o\log(n_o))$ ajalise keerukusega samme, siis algoritm kuulub $\Theta(n_o\log(n_o))$ ajalise keerukuse klassi.

Algoritmi ajalise keerukuse kontekstis kasutakse selles töös terminit „kiire algoritm“ juhul kui algoritm on arvutatav $\Theta(n_o\log(n_o))$ ajalise keerukusega või kiiremini ja vastasel juhul kasutatakse terminit „aeglane algoritm“.

Lomb-Scargle Periodogramm

Selleks, et leida aegreana esitatud füüsikalise suuruse perioodilisi muutumisi on võetud kasutusele periodogrammi mõiste. Klassikaline periodogramm on Deemingu[7] poolt defineeritud kui funktsiooni Fourier' teisenduse mooduli normeeritud ruut:

$$P_X(\omega)=\frac{1}{N_0}|\boldsymbol{F}[f(t)]|^2=\frac{1}{N_0}[\sum_n X_n\cos(\omega t_n)]^2+\frac{1}{N_0}[\sum_n X_n\sin(\omega t_n)]^2.$$

Periodogramm seab igale sagedusele vastavusse väärtuse, mis määrab antud sagedusega komponendi amplituudi. Deemingu periodogrammi puuduseks on see, et selle väärtuste statistilist olulisust on raske hinnata juhul kui ajapunktid on ebaühtlaselt jaotunud.

Selleks, et paremini hinnata arvutatud väärtuste statistilist olulisust ebaühtlase ajalise jaotuse korral kasutatakse Lombi[1] ja Scargle[2] poolt kasutusele võetud modifitseeritud periodogrammi

definitsiooni: $P_X(\omega)=\frac{1}{2}\left\{\frac{[\sum_n X_n\cos(\omega t_n-\omega\tau)]^2}{\sum_n \cos^2(\omega t_n-\omega\tau)}+\frac{[\sum_n X_n\sin(\omega t_n-\omega\tau)]^2}{\sum_n \sin^2(\omega t_n-\omega\tau)}\right\}$, kusjuures τ on

defineeritud järgnevalt: $\tan(2\omega\tau)=\frac{\sum_n \sin(2\omega t_n)}{\sum_n \cos(2\omega t_n)}$.

Antud töös periodogrammist rääkides peetakse alati silmas Lomb-Scargle periodogrammi.

Ebaühtlase ajasammuga kiire Fourier' teisendus

Fourier' teisendusest rääkides on sobilik alustada Fourier' reast, mis on defineeritud perioodilisel funktsioonil. Fourier' rida seab lähtefunktsioonile vastavusse trigonomeetriliste funktsioonide summast koosneva lähendi. Trigonomeetrilisi funktsioone saab tänu Euleri valemile vaadelda ka kompleksete eksponentide summana. Fourier' rea üldine juhtum, kus perioodi pikkuseks on kogu muutumispiirkond, miinus lõpmatusest pluss lõpmatuseni, on tuntud Fourier' teisenduse ja ka Fourier' pöörde nime all, selles töös kasutatakse edaspidi terminit Fourier' teisendus.
Defineerime diskreetne Fourier' teisenduse järgnevalt:

$$F[k]=\boldsymbol{F}[f(t_n)]=\sum_{n=0}^{N-1} f(t_n)e^{i2\pi k t_n}.$$

$\boldsymbol{F}[\,]$ tähistab oma argumendiks oleva funktsiooni Fourier' teisendust.

Cooley ja Tukey poolt on kasutusele võetud FFT (*fast Fourier transform* – kiire Fourier' teisendus) algoritm[8], mille abil saab kiirelt sooritada diskreetseid Fourier' teisendusi, kui on olemas mõõtmisandmed konstantse ajavahemiku, $\Delta t=const$, tagant ja ka sageduse muutumispiirkonnas on väärtused arvutatud konstantse sammuga.
Nagu eelnevalt sai mainitud, siis antud töös keskendutakse olukordadele, kus lähteandmed on ebaühtlase ajalise sammuga. Seega tuleb sooritada ebaühtlase ajasammuga Fourier' teisendus, mille jaoks on olemas NFFT (*non-uniform FFT* – ebaühtlane FFT) algoritm, kus saadakse Fourier' teisendusega hakkama ka olukorras, kus mõõtmisandmed on aja teljel jaotunud ebaühtlaselt[9].
Ebaühtlase diskreetse Fourier' teisenduse kiireks läbiviimiseks on vaja sooritada järgmised sammud [10]:

1. Määrida mõõtmistulemustega kaalutud Dirac-i delta funktsioonide summa

$f(t)=\sum_{n=1}^{N} f(t_n)\delta(t-t_n)$ laiali mingisuguse tuumfunktsiooniga $k(t)$ sidumise abil, et saada

pidev ajas muutuv funktsioon: $g(t)=f(t)*k(t)$. Algoritmi realiseerides tuleb määrata parameeter q, mis näitab mitmes punktis arvutatakse $g(t)$ väärtust iga mõõdetud väärtuse korral.

2. Rakendada diskretiseerimisoperaatorit, et saada pidevast funktsioonist ühtlane, diskreetne funktsioon sammuga Δt : $g[t]=III_{\Delta t}(t)g(t)=\sum_{n=-\infty}^{\infty} g(n\Delta t)\delta(t-n\Delta t)$, kus

$III_p(x)=\sum_{n_p=-\infty}^{\infty} f(n_p p)\delta(x-n_p p)$. Kusjuures $\Delta t=\frac{T}{M_0}$.

3. Viia läbi Fourier' teisendus, et saada ajaruumis $g[t]$ muutuvast funktsioonist sagedusruumis muutuv funktsioon $G[s]=\boldsymbol{F}[g[t]]$. Selle teisenduse läbiviimiseks saab kasutada FFT algoritmi.

4. Jagada saadud tulemus läbi Fourier' teisendatud tuumfunktsiooniga $F[s]=G[s]/K[s]$, kus $K[s]=\boldsymbol{F}[k(s)]$.

Algoritmiliselt on kaht esimese sammu hästi kirjeldanud Duijdam ja Schonewille[11]. Nimelt luuakse kõigepealt diskreetse sammuga, Δt, võre $g[t_c]$ ja määratakse igas võrepunktis funktsiooni väärtuseks null. Seejärel käiakse läbi kõik lähteandmete paarid $(t_n, f(t_n))$, kus iga mõõtmispunkti t_n kohta leitakse temale q lähimat võrepunkti t_c. Igas niimoodi leitud võrepunktis leitakse mõõtmistulemuse ja tuumfunktsiooni sidum $(f(t_n)*k(t))[t_c]$ ja liidetakse see varasemale $g[t_c]\rightarrow g[t_c]+(f(t_n)*k(t))[t_c]$ väärtusele. Tuumfunktsiooni valiku juures on oluline see, et ta väärtused läheneksid piisavalt kiiresti nullile nii aja- kui ka sagedusruumis. Tuumfunktsiooniks sobivad muuhulgas näiteks Gaussi funktsioon või sinc funktsioon, millede käitumist on uurinud Pelt[10].

Kõik neli sammu saab kokku võtta valemiga $F[s]=\frac{\boldsymbol{F}[k(t)*f(t)]}{K[s]}$, mis on otsene järeldus Fourier' teisenduse kohta kehtivast sidumi teoreemist: $\boldsymbol{F}[f(t)*g(t)]=F(s)G(s)$. Lisaks on sammud illustreerivalt ära toodud juuresoleval joonisel, mis on võetud tööst[11].

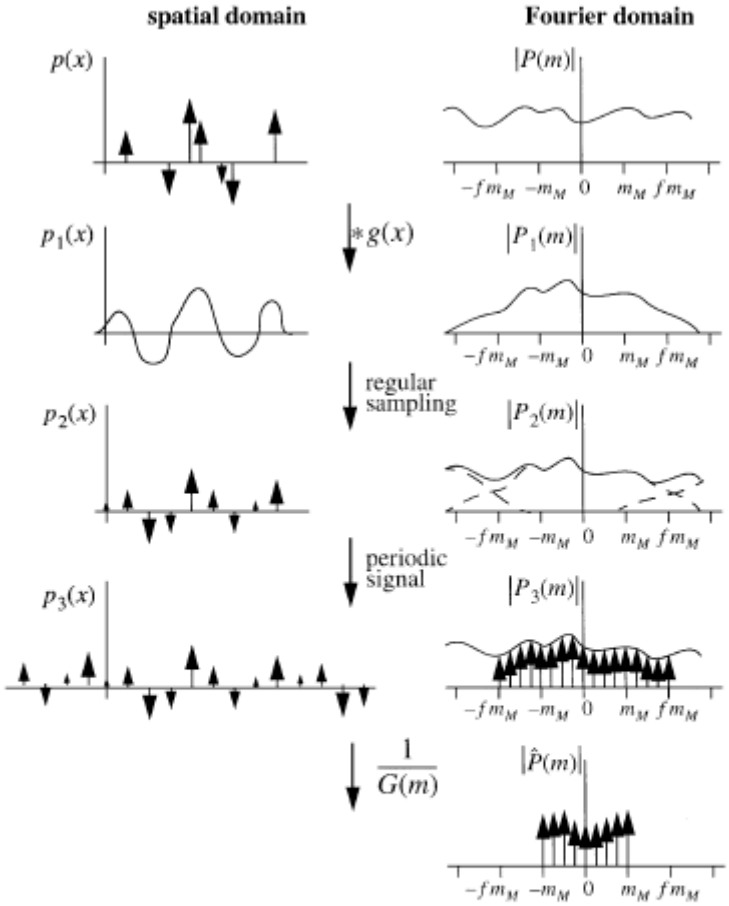


FIG. 1. Schematic overview of the NFFT algorithm. The interval $[-m_M, m_M)$, with $m_M = M/2$ is the region of interest. The interval $[-fm_M, fm_M)$ is computed by the FFT, since $p_1(x)$ is oversampled with a factor f, which is given the value $f = 2$ in this illustration. Aliasing caused by the sampling is illustrated with the dotted lines in the picture of $|P_2(m)|$. In the actual algorithm the sequence $p(x) \to p_2(x) \to p_3(x) \to P_3(m) \to \hat{P}(m)$ is followed.

NFFT3

NFFT3 on Lübecki Ülikooli, Osnabrücki Ülikooli ja Chemnitzi Tehnoloogia Ülikooli koostööl loodud teek, mis sisaldab funktsioone ebaühtlase kiire Fourier' teisenduse arvutamiseks[4]. NFFT3 sisaldab Fourier' teisenduse üldisusi mitme dimensionaalsel juhul ja ka teisendusi sagedusruumist ajaruumi, kuid selles töös piirdutakse ühedimensioanaalse juhuga, kus teisendus tehakse ajaruumist sagedusruumi.

Teegi iseärasused

Lugedes NFFT3 teegiga kaasas olevat juhendit tulevad välja mõningased iseärasused[12]. Esiteks on seal Fourier' teisendus defineeritud kui operatsioon, mis seab etteantud sagedusruumi

koefitsientidele vastavusse määratud „mõõtmiskohtadel“ saadud „tulemused“. Teise iseärasusena on NFFT3 teegis sagedused vahemikus $k \in \left[-\frac{M}{2}, \frac{M}{2}\right)$, kus $M \in 2\mathbb{N}$, kuid negatiivsed sagedused ei ole antud töö kontekstis olulised. Kolmandaks, muutuja t_n kohta tuleb ära märkida, et vastavalt NFFT3 teegi kitsendustele peab kehtima: $t_n \in [0,1)$ - selle saavutamiseks tuleb määrata lähteandmete ajaline vahemik ja teisendada kõik ajamomendid järgnevalt: $t_n \to \frac{t_n}{T}$. Antud töös defineeritud Fourier' teisendus on NFFT3 teegis defineeritud kui kaasteisendus (*adjoint transform*):

$F[k] = \sum_{n=0}^{N_0-1} X_n e^{i 2\pi k t_n}$. Kasutades Euleri valemit saab:

$F[k] = \sum_n X_n \cos(2\pi k t_n) + i \sum_n X_n \sin(2\pi k t_n)$. Selle tulemuse abil saab avaldada S_X ja C_X kaasteisenduse reaalosa ja imaginaarosa kaudu:

$$S_X[m] = \boldsymbol{Im}(F[m])$$
$$C_X[m] = \boldsymbol{Re}(F[m]).$$

Eelnevalt sai mainitud, et antud töös ei ole negatiivsed sagedused olulised. Teisalt on Fourier' teisenduse kohta teada seda, et kui lähteandmed on reaalsed ($\boldsymbol{Im}[f(t_n)] = 0$), mida nad antud töö eelduste põhjal ka on, siis teisenduse reaalosa on paarisfunktsioon ja imaginaarosa paaritu funktsioon. Järelikult negatiivsed sagedused ei kanna teisenduse kontekstis ka mittemingit lisainfot ja periodogrammi uurides võib piirduda vaid positiivsete või vaid negatiivsete sagedustega. NFFT3 teegis on sageduste vahemiku alumine raja, $-M/2$, kaasaarvatud, samas ülemine raja, $+M/2$, mitte. Olgu periodogrammis uuritavate sageduste arv $M_0 = M/2$, siis ühtlasi on suurim uuritav sagedus ka M_0 (korda perioodis $T = t_{N_0-1} - t_0$). Selleks, et periodogrammi uurivas algoritmis oleks info ka sageduse M_0 kohta on hõlpsam uurida negatiivseid sagedusi, ehk siis avaldada C_X ja S_X järgnevalt:

$$S_X[m] = -\boldsymbol{Im}(F[-m])$$
$$C_X[m] = \boldsymbol{Re}(F[-m]).$$

Teegis kasutatavate algoritmide ajaline keerukus

On ilmne, et üks võimalus Fourier' teisenduse leidmiseks on otsese summeerimise teel leida kõik $F[k]$ väärtused, sellisel juhul oleks vaja iga väärtuse leidmiseks liita kokku M eksponenti.

Peame elementaaroperatsiooni all silmas kahe eksponendi kokku liitmist, siis kokku on vaja teha $(M-1)*N_0$ elementaaroperatsiooni. Eeldades, et mõõtmisandmete arvu suurenedes suureneb ka meid huvitava sagedusspektri elementide arv: $M \sim N_0 \Leftrightarrow M = c\,N_0$, seega otsest summeerimist kasutav algoritm on ruutkeerukusega $\Theta(n_o^2)$.

Kasutades NFFT3 teeki saab arvutusi optimeerida, saades algoritmi keerukusega $\Theta(M\log(M)+\log(\epsilon)N_0)$, kus ϵ on soovitud täpsus[12]. Vaadeldes ajalist keerukust mingil konkreetsel täpsusel ϵ saab $\log(\epsilon)$ asendada konstandiga. Niimoodi saame väita, et fikseeritud täpsuse korral on tegu kiire algoritmiga, sest $\Theta(M\log(M)+\log(\epsilon)N_0)$ on sama ajalise keerukusega kui $\Theta(N_0\log(N_0)+c\,N_0)$, mis omakorda on sama ajalise keerukusega kui $\Theta(N_0\log(N_0))$. Seega NFFT3 teegi kasutamisel saame ajalise ruutkeerukusega algoritmist $\Theta(n_o\log(n_o))$ keerukusega algoritmi, mis omakorda tähendab ajalist võitu andmemahtude kasvamisel. Ajalise võidu suurust on eraldi hinnatud ka tulemuste osas.

Metoodika kirjeldus

Periodogrammi teisendused

Periodogrammile pealevaadates tundub, et seda ei ole võimalik kiire algoritmiga arvutada. Trigonomeetriliste teisenduste abil saab selle avaldise viia kujule, kus kõik liikmed on arvutatavad kiirelt. Järelikult kogu periodogrammi on võimalik arvutada algoritmiga, mis kuulub ajalise keerukuse klassi $\Theta(n_o \log(n_o))$. NFFT algoritmi kasutamiseks on vaja, et kõik summa märgi sees olevad siinused ja koosiinused oleksid esimeses astmes ja suurust τ sisaldavad liikmed oleks summa märgi ette toodud. Kasutades kahe nurga vahe valemeid ja trigonomeetrilise funktsiooni astet vähendavaid valemeid saab periodogrammi avaldada kujul:

$$P_X(\omega) = \frac{\left[\cos(\omega\tau)\sum_n X_n \cos(\omega t_n) + \sin(\omega\tau)\sum_n X_n \sin(\omega t_n)\right]^2}{N_0 + \cos(2\omega\tau)\sum_n X_n \cos(2\omega t_n) + \sin(2\omega\tau)\sum_n X_n \sin(2\omega t_n)} + \frac{\left[\cos(\omega\tau)\sum_n X_n \sin(\omega t_n) - \sin(\omega\tau)\sum_n X_n \cos(\omega t_n)\right]^2}{N_0 - \cos(2\omega\tau)\sum_n X_n \cos(2\omega t_n) - \sin(2\omega\tau)\sum_n X_n \sin(2\omega t_n)}.$$

Selleks, et edasine avaldis oleks lühem saab kasutusele võtta varem sisse toodud summade tähistused. Siinkohal on paras hetk meelde tuletada ka seda, et töös arvutatakse periodogrammi vaid teatud diskreetsetel väärtustel.

$$P_X(2\pi m) = \frac{\cos^2(2\pi m\tau) C_X[m]^2 + \sin(2\cdot 2\pi m\tau) S_X[m] C_X[m] + \sin^2(2\pi m\tau) S_X[m]^2}{N_0 + \cos(2\cdot 2\pi m\tau) C_2[m] + \sin(2\cdot 2\pi m\tau) S_2[m]} + \frac{\cos^2(2\pi m\tau) S_X[m]^2 - \sin(2\cdot 2\pi m\tau) S_X[m] C_X[m] + \sin^2(2\pi m\tau) C_X[m]^2}{N_0 - \cos(2\cdot 2\pi m\tau) C_2[m] - \sin(2\cdot 2\pi m\tau) S_2[m]}.$$

Trigonomeetria abil saab kõik suurust τ sisaldavad liikmed avaldada

$T_2 = T_2(2\pi m) = \tan(2\cdot 2\pi m\tau) = \frac{S_2[m]}{C_2[m]} \rightarrow \tau = \frac{\arctan(T_2)}{2\cdot 2\pi m}$ kaudu.

$\tau_1[m] = \cos^2(2\pi m\tau)$
$\tau_2[m] = \sin(2\cdot 2\pi m\tau)$
$\tau_3[m] = \sin^2(2\pi m\tau)$
$\tau_4[m] = \cos(2\cdot 2\pi m\tau)$

Seega lõplik avaldis periodogrammi arvutamiseks on:

$$P_X(2\pi m)=\frac{\tau_1 C_X^2+\tau_2 S_X C_X+\tau_3 S_X^2}{N_0+\tau_4 C_2+\tau_2 S_2}+\frac{\tau_1 S_X^2-\tau_2 S_X C_X+\tau_3 C_X^2}{N_0-\tau_4 C_2-\tau_2 S_2}.$$

Periodogrammi leidmiseks on vaja teha järgnevad sammud:

1. Leida $S_X[m]$ ja $C_X[m]$:

   NFFT teisenduse definitsiooni kohaselt on S_X lähtefunktsiooni Fourier' teisenduse imaginaarosa ja C_X reaalosa, seega on S_X ja C_X kiiresti leitavad NFFT algoritmi abil.

2. Leida $S_2[m]$ ja $C_2[m]$:

   Pannes tähele, et kui S_X ja C_X avaldiste paremale poolele teha muutused: $X_n \to 1$ ja $m \to 2m$, siis saab avaldised S_2 ja C_2. Seega S_2 ja C_2 leidmiseks tuleb lähteandmetest võtta vaid ajamomendid ning asendada kõik mõõtmistulemused ühega. Selleks, et saada kaks korda suuremaid sagedusi tuleb määrata uuritavate sageduste arvu näitav parameeter kaks korda suurema väärtusega: $M_0 \to 2\mathrm{M}_0$. Selliste sammude abil on ka S_2 ja C_2 kiiresti leitavad NFFT abil.

3. Leida $\tau_{1\ldots4}[m]$

   Kõik τ -ga tähistatud kordajad on T_2 abil avaldatavad S_2 ja C_2 kaudu, seega kui S_2 ja C_2 on NFFT abil juba leitud, siis $\tau_{1\ldots4}$ on leitavad lineaarse ajalise keerukusega.

4. Leida $P_X(2\pi m)$

   Kui kõik paremal pool olevad väärtused on iga sageduse jaoks leitud, siis viimane samm on jällegi ajaliselt lineaarse keerukusega.

Avaldises olev N_0 on konstant, seega selle leidmiseks pole vaja vaeva näha.

Vaadeldes kogu periodogrammi leidmist on vaja teha üksteise järel kaks $\Theta(n_o \log(n_o))$ keerukusega sammu ja seejärel kaks lineaarse keerukusega sammu, seega kuulub esitatud periodogrammi leidmise algoritm $\Theta(n_o \log(n_o))$ ajalise keerukuse klassi.

NFFT3 teegi testimine

Enne kui asuda NFFT3 teeki kasutama periodogrammi arvutamiseks tuleb kasuks teha mõningad katsed kontrollimaks, et eelnevalt kirjeldatud iseärasused peava paika ja veendumaks, et saab ikka soovitud tulemused. Esiteks saab genereerida testandmeteks arvupaare, kus suvalistel kohtadel on mõõdetud kahe koosiinusfunktsiooni, millede sagedused on 1 ja 5 ühikut, summat. Kusjuures

sagedusega 5 võnkuva funktsiooni amplituud on suurem. Teise asjana saab võrrelda kiire ja aeglase algoritmi ajalist keerukust, selle jaoks saab genereerida hulga suvalisi andmeid ja kasutada NFFT3 teegi võimalust arvutada teisendust nii kiire meetodiga kui ka otsese summeerimise teel. Ajalise keerukuse kohta on esitatud tabel.

Periodogrammi arvutamine

Lomb-Scargle periodogrammi kasutatakse tema statistiliste omaduste tõttu, mida on uurinud mitmed autorid[1][2][3]. Antud töös uuritakse periodogrammi arvutamisel esinevaid vigu – nimelt esineb kahte liiki arvutusvigu:

1. Arvuti ujuvkoma arvudega kasutamisest tulenev viga.
2. Kiire algoritmi kasutamisel tehtud diskretiseerimisest tulenev viga.

Esimest liiki arvutusvigade vähendamiseks on NFFT3 teegis kasutusel topelt täpsusega ujuvkoma arvu tüüp *double*, kuid sellist viga ei ole võimalik kuidagi täielikult eemaldada. Teist liiki arvutusvigu on analüütiliselt kirjeldanud näiteks Duijdam ja Schonewille[11] ning neid on võimalik vähendada suurendades algoritmi parameetrit q, mis määrab tuumfunktsiooniga sidumi arvutamisel kasutatavate punktide arvu.

Erinevad meetodid

Antud töös arvutatakse periodogramm kolmel erineval viisil:

1. Arvutada periodogrammi väärtused tseselt, ilma mingeid teisendusi tegemata. Kõigepealt arvutada S_2 ja C_2 ning seejärel leida τ ja arvutada periodogramm vastavalt definitsioonvalemile.
2. Viia läbi trigonomeetrilised teisendused ja seejärel leida summad S_X, C_X, S_2 ja C_2 otsese summeerimise teel.
3. Viia läbi trigonomeetrilised teisendused ja seejärel leida summad S_X, C_X, S_2 ja C_2 kiiret algoritmi kasutades. Kiire algoritmi juures kasutatakse NFFT3 teegist tulevaid vaikimisi seadistusi. Parameetri q väärtuseks on 13 ja tuumfunktsioonina kasutatakse Kaiser-Besseli funktsiooni.

Ajaliselt on kaks esimest moodust ruutkeerukusega, samas kui kolmas on $\Theta(n_o \log(n_o))$ keerukusega. Lisaks on kahe esimese moodusega arvutatud periodogrammid matemaatiliselt identsed. Motivatsioon selleks, et arvutada kahte matemaatiliselt identset periodogrammi erinevatel

aeglastel meetoditel tuleb sellest, et nii saab võrrelda ujuvkoma arvude kasutamisest tingitud paratamatut arvutusviga kiire teisenduse diskretiseerimisest tuleneva arvutusveaga.

Täpsuse võrdlus

Selleks, et võrrelda erinevatel meetoditel arvutatud periodogrammi täpsust sai genereerida 16384 juhuslikku mõõtmistulemust juhuslikes ajapunktides. Veale hinnangu andmiseks saab võrrelda periodogrammi väärtuste erinevust maksimaalse amplituudiga:

$E_{a,b}=\frac{max[|P_{Xa}(2\pi m)-P_{Xb}(2\pi m)|]}{max[P_{Xa}(2\pi m)]}$, kus $a=\{1,2\}; b=\{2,3\}; a<b$ ja P_{Xl} tähistab meetodil l arvutatud periodogrammi[13]. Kümne katse vead ja vigade keskmine on välja toodud tulemuste osas.

Ajaline võrdlus

Ajaliselt võrdleme otsest summeerimist kiire algoritmiga. Teine arvutusmeetod sai kasutusele võetud vaid selleks, et võrrelda kiire aproksimatsiooni vigu paratamatute vigadega, seega ajalise võrdluse osas seda ei kasutata. Ajalise võrdlemise jaoks sai genereerida juhuslikkude arvude paaridest koosnevaid andmemassiive, alustades 1024 arvupaarist ja jätkates aeglase algoritmiga kuni kulub aega alla kahe minuti ja kiire algoritmiga kuni arvuti mälu otsa saab. Olgu lähteandmete massiivi suuruseks N_0 arvupaari ja arvutatavate periodogrammi punktide arv $M_0=N_0$. Enne tulemuste esitamist proovime analüütiliselt määrata aeglase ja kiire algoritmi ajalise keerukusklassi.

Esiteks aeglase algoritmi korral kõikide suuruste $\tau_{1...4}$ leidmiseks kulub:

$t_\tau=c_1N_0+c_2N_0\cdot 2N_0+c_3N_0\cdot 2N_0=c_1N_0+(c_2+c_3)N_0^2=c_1N_0+c_4N_0^2$, kus c_1 on jagamise ja arkustangensi jaoks kuluv aeg, c_2 on kahe siinuse kokkuliitmiseks kuluv aeg ja c_3 kahe koosinuse kokkuliitmiseks kuluv aeg. Kui kõik suurused $\tau_{1...4}$ on leitud, siis kõikide P_X leidmiseks kuluv aeg on $t_P=c_5N_0+c_6N_0^2$, kus c_5 on liitmisteks ja jagamisteks kuluv aeg ja c_6 nelja periodogrammis oleva summa kahe elemendi liitmiseks kuluv aeg. Kogu periodogrammi leidmiseks kuluv aeg on $t_{aeglane}=c_{10}+c_1N_0+c_4N_0^2+c_5N_0+c_6N_0^2+c_7N_0=c_8N_0+c_9N_0^2$, kus c_7 on lähteandmete failist lugemise ja tulemuse faili kirjutamiseks kuluv aeg ja c_{10} programmi töötamiseks kuluv aeg, mis ei sõltu andmete mahust.

Kiire algoritmi korral sooritatakse kaks Fourier' teisendust, kus S_X ja C_X leidmiseks tehtaval teisendusel on sageduspiirkonnas $M=2M_0=2N_0$ punkti ning S_2 ja C_2 leidmiseks tehtaval

teisendusel on sageduspiirkonnas $M = 4\mathrm{M}_0 = 4\mathrm{N}_0$ punkti.

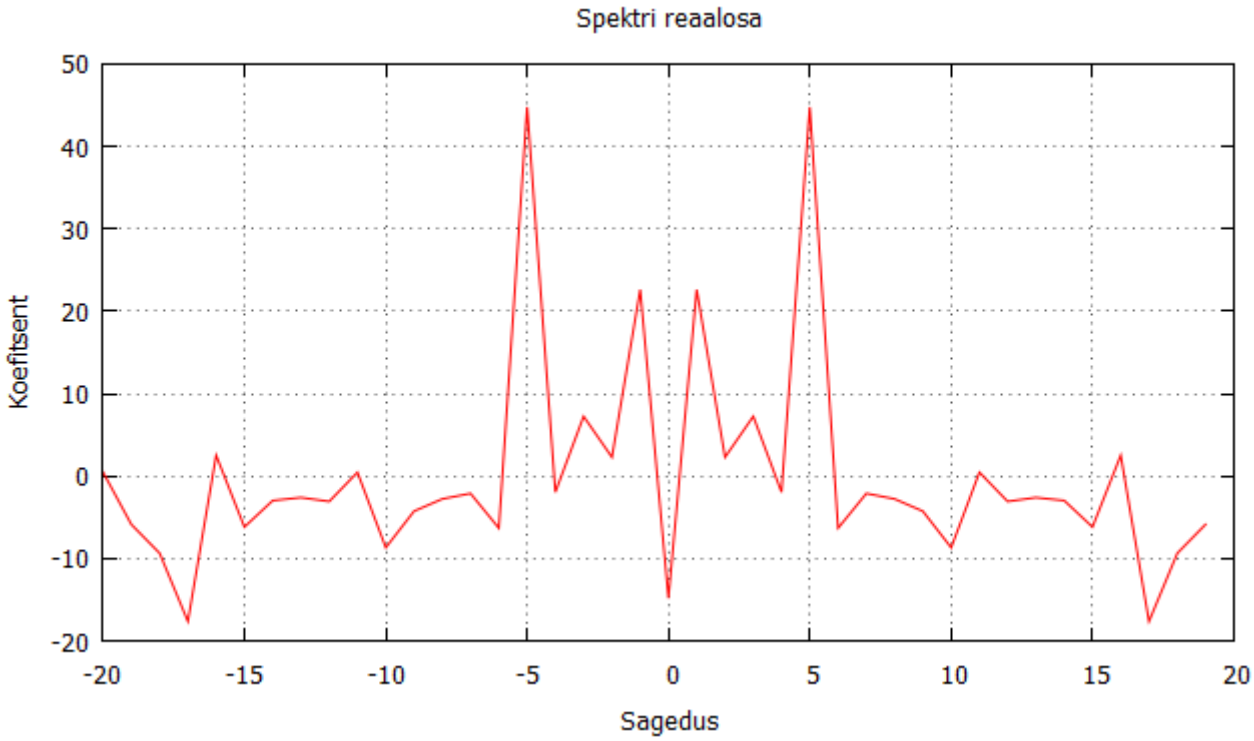
Kogu periodogrammi leidmiseks kuluv aeg kiirel meetodil on avaldatav seega järgnevalt: $t_{kiire} = c_4 + c_1 N_0 + c_2 2\mathrm{N}_0 \log(2\mathrm{N}_0) + c_2 4\mathrm{N}_0 \log(4\mathrm{N}_0) = c_4 + c_3 N_0 + c_2 6 N_0 \log(N_0)$. Siinkohal on võimalik $c_2 6$ asendada uue kordajaga c_5, kuid enne selle tegemist tuleks ära märkida, et Fourier' teisenduse paarsusest tulenevatele omadustele tuginedes on võimalik sooritada kaks teisendust samaaegselt[14]. Tehes seda, jääks ära S_X ja C_X leidmise ajakulu ja uus konstant c_5 oleks kolmandiku võrra väiksem. $t_{kiire} = c_4 + c_3 N_0 + c_5 N_0 \log(N_0)$.

Tulemused

NFFT3 katsed

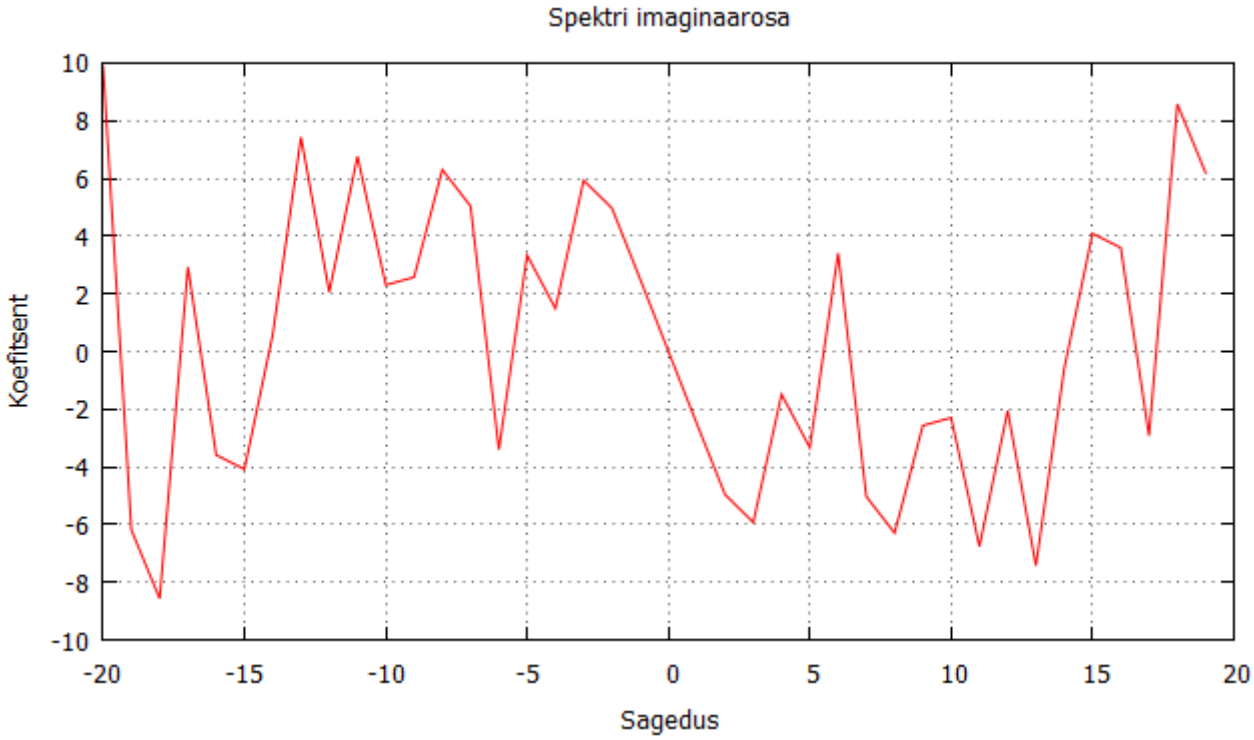
NFFT3 teek saab kiire teisendusega hakkama veel juhul kui lähteandmete massiivi suurus on seitse miljonit arvupaari, kaheksa miljoni suuruse andmemassiivi puhul jäi sülearvuti 4GB RAM mälust väheks. Otsese summeerimise puhul on seatud piir, et kui teisenduse tegemiseks läheb üle kahe minuti, et siis enam suurema andmemassiiviga teisendust ei tehta.

Graafik1 Spektri reaalosa



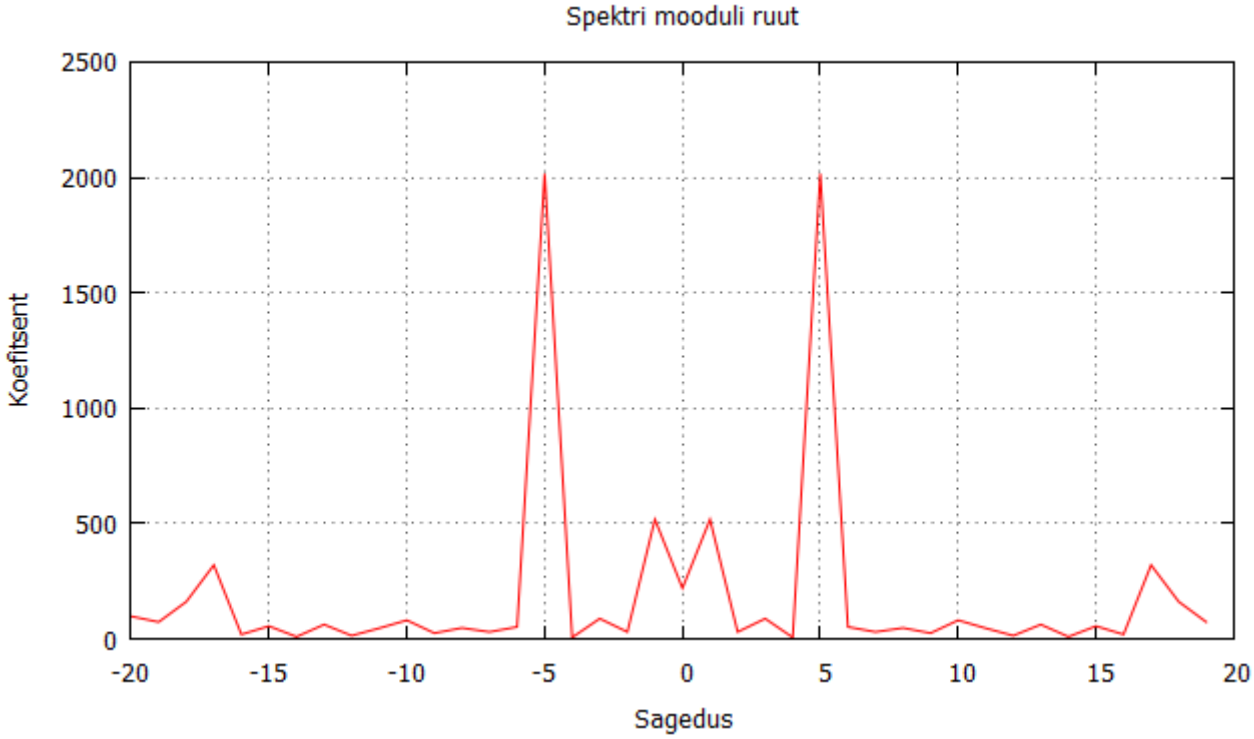
Jooniselt on näha, et spektri reaalosa on paarisfunktsioon.

Graafik2 Spektri imaginaarosa



Jooniselt on näha, et spektri imaginaarosa on paaritu funktsioon.

Graafik3 Spektri mooduli ruut

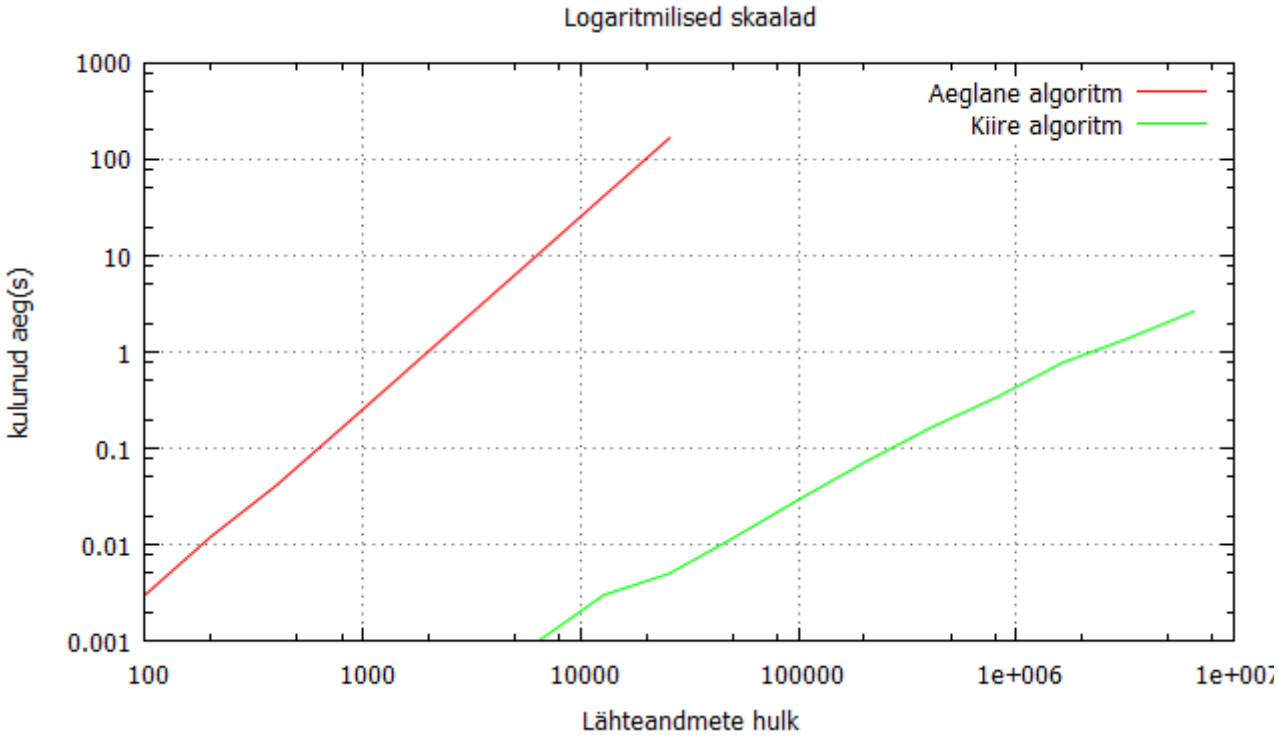


Joonisel on näha selged piigid sagedustel 1 ja 5.

Tabel1 Kiire ja aeglase teisenduse võrdlus

Andmete hulk	Aeg kiire algoritmiga (s)	Aeg aeglase algoritmiga (s)
200		0,012
400		0,041
800		0,162
1600		0,648
3200	0,001	2,593
6400	0,001	10,349
12800	0,003	41,586
25600	0,005	166,456
51200	0,012	-
102400	0,030	-
204800	0,073	-
409600	0,165	-
819200	0,341	-
1638400	0,775	-
3276800	1,384	-
6553600	2,634	-

Graafik4 Kiire ja aeglase teisenduse võrdlus



Periodogrammi katsed

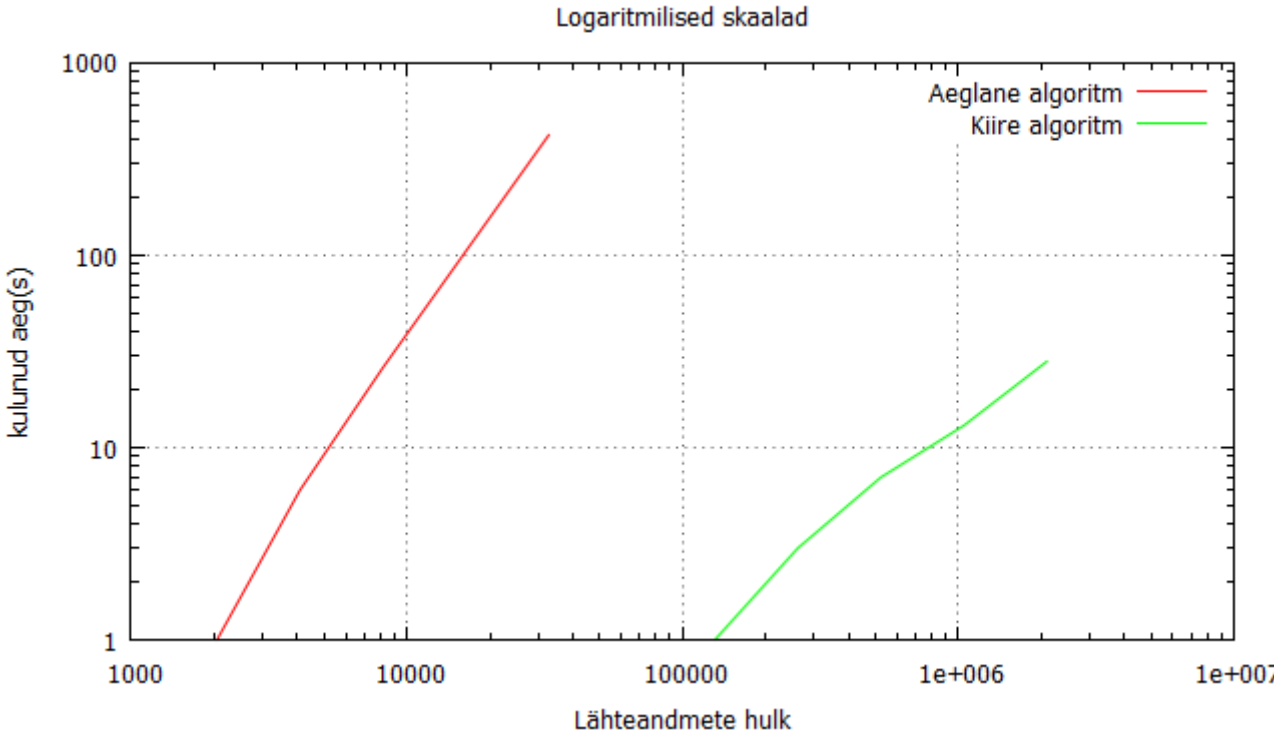
Tabel2 Periodogrammi arvutamise täpsuse võrdlus

	$E_{1,2}$	$E_{1,3}$	$E_{2,3}$
Katse1	$8{,}35 \cdot 10^{-12}$	$8{,}79 \cdot 10^{-12}$	$8{,}74 \cdot 10^{-12}$
Katse2	$6{,}85 \cdot 10^{-12}$	$1{,}36 \cdot 10^{-11}$	$1{,}34 \cdot 10^{-11}$
Katse3	$6{,}89 \cdot 10^{-12}$	$1{,}58 \cdot 10^{-11}$	$1{,}42 \cdot 10^{-11}$
Katse4	$8{,}64 \cdot 10^{-12}$	$1{,}99 \cdot 10^{-11}$	$1{,}98 \cdot 10^{-11}$
Katse5	$5{,}81 \cdot 10^{-12}$	$7{,}69 \cdot 10^{-12}$	$7{,}67 \cdot 10^{-12}$
Katse6	$1{,}74 \cdot 10^{-11}$	$1{,}83 \cdot 10^{-11}$	$1{,}36 \cdot 10^{-11}$
Katse7	$1{,}58 \cdot 10^{-11}$	$2{,}65 \cdot 10^{-11}$	$2{,}67 \cdot 10^{-11}$
Katse8	$1{,}60 \cdot 10^{-11}$	$1{,}83 \cdot 10^{-11}$	$3{,}42 \cdot 10^{-11}$
Katse9	$7{,}89 \cdot 10^{-12}$	$2{,}74 \cdot 10^{-11}$	$2{,}76 \cdot 10^{-11}$
Katse10	$7{,}78 \cdot 10^{-12}$	$1{,}34 \cdot 10^{-11}$	$1{,}29 \cdot 10^{-11}$
Keskmine	$\mathbf{1{,}01 \cdot 10^{-11}}$	$\mathbf{1{,}70 \cdot 10^{-11}}$	$\mathbf{1{,}79 \cdot 10^{-11}}$

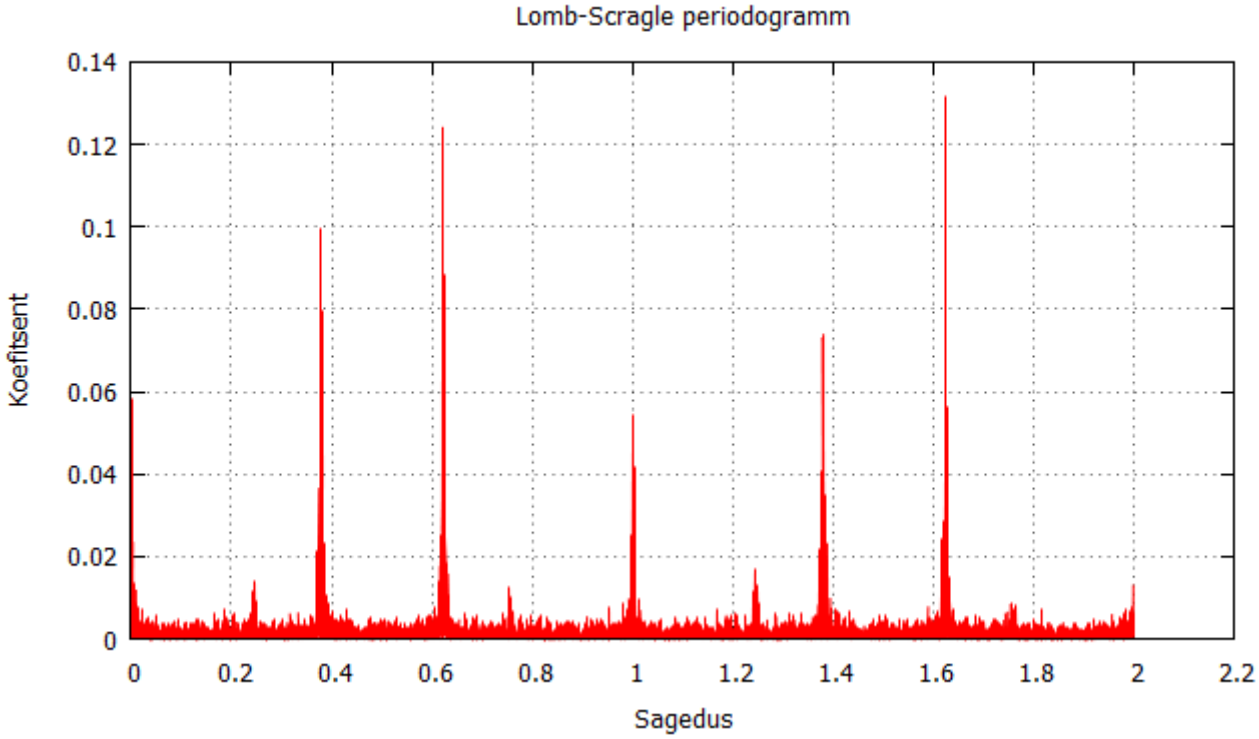
Tabel3 Periodogrammi arvutamise ajaline võrdlus

Andmete hulk	Aeg kiire algoritmiga (s)	Aeg aeglase algoritmiga (s)
1024	0,015	0,436
2048	0,031	1,653
4096	0,062	6,630
8192	0,109	26,488
16384	0,202	105,705
32768	0,421	422,511
65536	0,842	-
131072	1,716	-
262144	3,416	-
524288	7,269	-
1048576	13,946	-
2097152	28,688	-

Graafik5 Periodogrammi arvutamise ajaline võrdlus



Graafik6 LQ Hya periodogramm



Järeldused

NFFT3 teegi testimine

Graafikutes 1 kuni 3 on näha, et teegi kohta tehtud eeldused peavad paika: reaalosa on paarisfunktsioon ja imaginaarosa paaritu ning sagedustel 1 ja 5 on selgelt eristuvad piigid, kusjuures 5 juures asetsev piik on ka kõrgem.
Võrreldes tabelis 1 kiire algoritmi kaht viimast rida siis eeldaks, et viimase ja eelviimase rea suhe on natuke suurem kui 2, kuid antud mõõtmiste puhul tuli nende suhe natuke väiksem kui 2. Põhjus on kasutatud aja mõõtmise metoodikas – nimelt tabelis esitatud aeg on arvuti süsteemsete aegade vahe enne ja pärast teisendust. Kogu arvutamise aja on operatsiooni süsteem hõivatud ka mitmete muude protsessidega ning mõõtes väikseid ajavahemikke tulevad niimoodi sisse märkimisväärsed vead.
Samas, kui võrrelda tabelis 1 kahte viimast aeglase algoritmi rida on näha, et andmemahtude kahekordistumisel algoritmi lahendamiseks kuluv aeg tõepoolest peaaegu neljakordistub.
Tabelis 1 toodud info põhjal on võimalik järeldada, et NFFT3 teek võimaldab arvutada varasemalt ajalise ruutkeerukusega algoritmi tunduvalt kiiremini.

Periodogrammi arvutamine

Täpsuse võrdlus

Tabelist 2 on näha, et kiire algoritmi kasutusele võtmisega sisse tulev viga on samas suurusjärgus ujuvkoma arvude kasutamisest tingitud veaga. Saab väita, et antud meetodil tekkivad arvutusvead on piisavalt väiksed, et meetodit võiks laialdaselt kasutada astronoomiliste aegridade sagedusanalüüsis.

Ajaline võrdlus

Tabelist on näha, et aeglane algoritm on ruutkeerukusega. Selle põhjal on võimalik hinnata aeglase algoritmiga kuluvat aega 2097152 arvupaariga andmemassiivi töötlemiseks:

$$\left(\frac{2097152}{32768}\right)^2 * 422{,}511\ s = 1730605{,}056\ s = 20{,}03\ päeva\,.$$

Periodogrammi arvutades on kiire algoritmiga kuluvad ajad mõnevõrra suuremad kui need oli teeki

testides, võrreldes kiire algoritmi viimaseid ridu saab nende suhteks natuke kahest suurema arvu nagu kiire algoritmi puhul võiski eeldada.

LQ Hya

LQ Hya on noor magnetiliselt aktiivne üksiktäht. Tähe vanuseks on hinnatud 35 miljonit kuni 55 miljonit aastat. Lisaks sellele on LQ Hya plekiline, mis omakorda põhjustab koos pöörlemisega kõikumisi tähe heleduses.[5]

Näitlikustamaks NFFT3 teegi sobivust astronoomiliste aegridade jaoks sai kasutatud andmeid tähe LQ Hya jaoks. Varasemalt on tähe perioodi uurides tuvastatud, et pöörlemise periood muutub ajas, kuid asetseb umbes 1,6 päeva lähedal[5]. Tähe andmetega sai arvutatud periodogramm maksimaalse sageduseni 2 võnget päevas.Graafikus 4 välja toodud periodogrammi joonisel on näha piik sagedusega 0,62 korda päevas, mis vastab 1,61 päevasele perioodile. Lisaks sellele sai ka LQ Hya kohta käivaid andmeid arvutada aeglase algoritmiga ning viga tuli: $E_{1,3} = 1{,}07 \cdot 10^{-12}$.

Kokkuvõte

Trigonomeetriliste summade kiire arvutamise meetodid astronoomiliste aegridade sagedusanalüüsis

Sander Kõiv

Kokkuvõte

Antud töös näidati erinevates kontekstides ebaühtlase ajasammuga Fourier' teisenduse kiiret arvutamist, tuues välja NFFT3 teegi iseärasused. Viidi varasemalt tuttav Lomb-Scargle periodogramm kujule, mis on kiire algoritmi abil arvutatav. Lisaks näidati seda, et kiire algoritmi kasutamisest tingitud arvutusvigade suurusjärk on võrreldav ujuvkoma arvude kasutamisest tingitud ebatäpsusega, mistõttu sobib antud meetod kasutamiseks astronoomiliste aegridade sagedusanalüüsis. Sellise sobilikkuse näitlikustamiseks arvutati kiire algoritmi abil varasemalt uuritud tähe, LQ Hya, periodogramm ning näidati, et tulemus langeb kokku eelnevalt teada olevaga.

Viited

1. N. R. Lomb, "Least-squares frequency analysis of unequally spaced data," Astrophysics and Space Science 39, 447-462 (1976).
2. J. D. Scargle, "Studies in astronomical time series analysis. II. Statistical aspects of spectral analysis of unevenly spaced data," The Astrophysical Journal 263, 835-853 (1982).
3. C. Koen, "Significance testing of periodogram ordinates," The Astrophysical Journal 348, 700-702 (1990).
4. J. Keiner, S. Kunis, D. Potts, NFFT - http://www-user.tu-chemnitz.de/~potts/nfft/.
5. J. Lehtinen, L. Jetsu, T. Hackman, P. Kajatkari, G. W. Henry, "Spot activity of LQ Hya from photometry between 1988 and 2011," Astronomy & Astrophysics 524, A38 (2012).
6. J. Kiho, Algoritmid ja andmestruktuurid (Tartu Ülikooli Kirjastus, Tartu, 2003).
7. T. J. Deeming, "Fourier Analysis with Uneually-Spaced data," Astrophysics and Space Science 36, 137-158 (1975).
8. J. W. Cooley, J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," Mathematics of Computation 19, 297-301 (1965).
9. A. Dutt, V. Rokhlin, "Fast Fourier transforms for nonequispaced data," Society for Industrial and Applied Mathematics 007, 1368-1393 (1993).
10. J. Pelt, "Fast computation of trigonometric sums with applications to frequency analysis of astronomical data," kogumikus Astronomical Time Series, D. Maoz, A. Sternberg, E. Leibowitz, eds. (Springer Netherlands, 1997), lk 179-182.
11. A. J. W. Duijdam, M. A. Schonewille, "Nonuniform fast Fourier transform," Geophysics 64, 539-551 (1999).
12. J. Keiner, S. Kunis, D. Potts, "Using NFFT 3 a software library for various nonequispacedfast Fourier transforms," (2009) http://www-user.tu-chemnitz.de/~potts/paper/nfft3.pdf.
13. A. Dutt, V. Rokhlin, "Fast Fourier transform for nonequispaced data," Society for Industrial and Applied Mathematics 007, 1368-1393 (1993).
14. J. Shima, "Computing the FFT of two real signals using a single FFT," (2000) http://www.hyperdynelabs.com/dspdude/papers/COMPUTING%20THE%20FFT%20OF%20TWO%20REAL%20SIGNALS%20USING%20A%20SINGLE%20FFT.pdf.

Summary

Fast computation of trigonometric sums in astronomical time series ferquency analysis

Sander Kõiv

Summary

In current paper nonuniform timestep fast Fourier' transform was calculated in different contexts and special characteristics of NFFT3 library were outlined. Well-known Lomb-Scargle periodogram was given a form, which makes it calculable by fast algorithm. In addition to that it was shown that loss of precision caused by the use of fast algorithm is about the size of floating point operations precision – this makes used method suitable for astronomical data analysis. To futher illustrate the suitability of this method, the periodogram of an example star, LQ Hya, was calculated and the result was compared to what was previously known about LQ Hya.

Lisad

Lisa1 Lähtekood

```
/*
 * Copyright (c) 2002, 2012 Jens Keiner, Stefan Kunis, Daniel Potts
 *
 * This program is free software; you can redistribute it and/or modify it under
 * the terms of the GNU General Public License as published by the Free Software
 * Foundation; either version 2 of the License, or (at your option) any later
 * version.
 *
 * This program is distributed in the hope that it will be useful, but WITHOUT
 * ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS
 * FOR A PARTICULAR PURPOSE.  See the GNU General Public License for more
 * details.
 *
 * You should have received a copy of the GNU General Public License along with
 * this program; if not, write to the Free Software Foundation, Inc., 51
 * Franklin Street, Fifth Floor, Boston, MA 02110-1301, USA.
 */

#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include <time.h>
#ifndef HAVE_COMPLEX_H
#define HAVE_COMPLEX_H 1
#endif
#include <complex.h>

#include "nfft3util.h"
#include "nfft3.h"

typedef struct INFO
{
        int lines;
        double minPos;
        double maxPos;
} INFO;

typedef struct FRE
{
        double XC;
        double XS;
        double C2;
        double S2;
        double T2;
        double Z1;
        double Z2;
        double Z3;
        double Z4;
        double Z5;
        double P;
        double tau;
```

```
} FRE;

FRE *f;

int main (int argc, char* argv[])
{
        if (argv[1] && argv[2])
        {
                char* format = "%lf%lf";
                int periodCount;
                double T;
                double minPeriod;

                INFO *info;
                INFO inf;
                info = &inf;

                fillInfo (argv[1], format, info);

                minPeriod = atof(argv[2]);

                T = info->maxPos - info->minPos;
                periodCount = ceil(T/minPeriod);

                printf("lines: %d\n", info->lines);
                printf("periodCount: %d\n", periodCount);

                f = malloc (periodCount * sizeof(FRE));
                fillXSums(info, periodCount, argv[1]);
                fillSums(info, periodCount, argv[1]);

                int i;
                calcPeriodogram(periodCount, info->lines);

                char* outFileName = "Output.txt";
                FILE *outFile;
                outFile = fopen(outFileName, "w+");
                for (i = 0; i < periodCount; i++)
                {
                        double dou = (double)(i+1);
                        if (argv[2])
                        {
                                fprintf(outFile, "%.20lf\t%.20lf\n", dou/T,
f[i].P);
                        } else
                        {
                                fprintf(outFile, "%d\t%.20lf\n", i, f[i].P);
                        }
                }

                printf("Output in \"Output.txt\"");

                fclose(outFile);

                free(f);

        } else
        {
                printf("Give file name as first argument and minimum period as
second.");
        }
```

```
}

//Get line count, that matches format and min and max values.
int fillInfo(char* fileName, char* format, INFO* info)
{
        int lines = 0;
        char line[100];
        double samplePos, sampleVal, minPos, maxPos;
        FILE *f = fopen(fileName,"r+");

        int bool = 0;
        while (fgets(line, 100, f) != NULL)
        {
                if (sscanf(line,"%lf%lf", &samplePos, &sampleVal) == 2)
                {
                        lines += 1;
                        if ( !bool || samplePos < minPos)
                        {
                                minPos = samplePos;
                        }
                        if ( !bool || samplePos > maxPos)
                        {
                                maxPos = samplePos;
                        }
                        bool++;
                }
        }
        fclose(f);

        info->lines = lines;
        info->maxPos = maxPos;
        info->minPos = minPos;

        return 1;
}

int fillXSums(INFO* info, int count, char* tempFileName)
{
        nfft_plan p;
        int M = 2 * count;
        nfft_init_1d(&p, M, info->lines);

        FILE *tempFile;
        tempFile = fopen(tempFileName, "r+");
        double samplePos, sampleVal, fullPos;
        double T = info->maxPos - info->minPos;
        char line[100];

        int i;
        for (i = 0; i < info->lines; i++)
        {
                fgets(line, 100, tempFile);

                sscanf(line,"%lf%lf", &samplePos, &sampleVal);
                fullPos = (samplePos - info->minPos);
                double integral;
                double mappedPos = modf(fullPos / T, &integral);
                mappedPos -= 0.5;

                p.x[i] = mappedPos;
```

```
                p.f[i] = sampleVal;
        }

        fclose(tempFile);

        /** precompute psi, the entries of the matrix B */
        if(p.nfft_flags & PRE_ONE_PSI)
        {
                nfft_precompute_one_psi(&p);
        }

        //fast Fourier transform
        nfft_adjoint(&p);

        for (i = 0; i < count; i++)
        {
                f[i].XC = creal(p.f_hat[count - 1 - i]);
                f[i].XS = -cimag(p.f_hat[count - 1 - i]);
        }

    nfft_finalize(&p);

        return 1;
}


int fillSums(INFO* info, int count, char* fileName)
{
        nfft_plan p;
        int M = 2 * count;
        nfft_init_1d(&p,2*M, info->lines);

        FILE *file;
        file = fopen(fileName, "r+");
        double samplePos, sampleVal,fullPos;
        double T = info->maxPos - info->minPos;
        char line[100];

        int i;
        for (i = 0; i < info->lines; i++)
        {
                fgets(line, 100, file);

                sscanf(line,"%lf%lf", &samplePos, &sampleVal);
                fullPos = (samplePos - info->minPos);
                double integral;
                double mappedPos = modf(fullPos / T, &integral);
                mappedPos -= 0.5;

                p.x[i] = mappedPos;
                p.f[i] = 1;
        }

        fclose(file);

        /** precompute psi, the entries of the matrix B */
        if(p.nfft_flags & PRE_ONE_PSI)
        {
                nfft_precompute_one_psi(&p);
        }
```

```
        //fast
        nfft_adjoint(&p);

        //slow
        //nfft_adjoint_direct(&p);

        for (i = 0; i < count; i++)
        {
                f[i].C2 = creal(p.f_hat[2 * (count - 1 - i)]);
                f[i].S2 = -cimag(p.f_hat[2 * (count - 1 - i)]);
        }

    nfft_finalize(&p);

        return 1;
}

int calcPeriodogram(int count, int N_0)
{
        int oomega;
        double L1,N1,L2,N2;
        double tau, Z1, Z2, Z3, Z4, Z5;
        for (oomega = 1; oomega <= count; oomega++)
        {
                int i = oomega - 1;
                tau = (atan(f[i].S2/f[i].C2))/(2 * oomega);
                Z1 = cos(oomega * tau) * cos(oomega * tau);
                Z2 = 2 * sin(oomega * tau) * cos(oomega * tau);
                Z3 = sin(oomega * tau) * sin(oomega * tau);
                Z4 = cos(2 * oomega * tau);
                Z5 = sin(2 * oomega * tau);
                L1 = Z1 * f[i].XC * f[i].XC + Z2 * f[i].XC * f[i].XS + Z3 *
f[i].XS * f[i].XS;
                N1 = N_0 + Z4 * f[i].C2 + Z5 * f[i].S2;
                L2 = Z1 * f[i].XS * f[i].XS - Z2 * f[i].XC * f[i].XS + Z3 *
f[i].XC * f[i].XC;
                N2 = N_0 - Z4 * f[i].C2 - Z5 * f[i].S2;
                f[i].P = L1/N1 + L2/N2;
                f[i].tau = tau;
        }
        return 1;
}
```

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina, Sander Kõiv,

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose
Trigonomeetriliste summade kiire arvutamise meetodid astronoomiliste aegridade sagedusanalüüsis

mille juhendaja on Jaan Pelt,

1.1. reprodutseerimiseks säilitamise ja üldsusele kättesaadavaks tegemise eesmärgil, sealhulgas digitaalarhiivi DSpace-is lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
1.2. üldsusele kättesaadavaks tegemiseks Tartu Ülikooli veebikeskkonna kaudu, sealhulgas digitaalarhiivi DSpace´i kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.

2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.

3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tartus/Tallinnas/Narvas/Pärnus/Viljandis, **23.05.2014**