Öldur og sog í höfnum

Úrvinnsla sjávarborðsmælinga

Valdís Guðmundsdóttir
Sigurður Sigurðarson
Maí 2020

Útgáfa	Dagsetning	Endurskoðun	Útgefið af	Útgefið til
Útgáfa A	2020.05.27		VG	Vegagerðin
Drög B	2020.05.18		VG	Vegagerðin
Drög A	2019.11.01		VG	Vegagerðin

Upplýsingar um skýrslu			
Verkkaupi:			
Verkefni:	Öldur og sog í höfnum - Úrvinnsla sjávarborðsmælinga		
Verkefnisnúmer.:			
Aðgengi:	☒ Opið	☐ Dreifing háð samþykki verkkaupa	☐ Lokað
Höfundar:	Valdís Guðmundsdóttir og Sigurður Sigurðarson		
Tilvísun:			
Forsíðumynd	Landeyjahöfn í apríl 2015, ljósmyndari Guðmundur Alfreðsson.		

Helstu niðurstöður

Við sjávarborðsmælingar í höfnum eru yfirleitt notaðir þrýstimælir sem komið er fyrir í stálröri og það fest við hafnarkant. Í þrýstimælinum er þrýstinemi sem skilar þrýstimælingu á einnar sekúndu fresti, þ.e. með tíðninni 1 Hz. Í sjávarborðsmælingum er verið að leitast eftir meðalsjávarborði yfir ákveðinn tíma, t.d. 1 mínútu eða 10 mínútur og því tekið meðaltal af öllum mælingum nemans yfir þann tíma. Þannig eru síaðar frá bæði öldu- og sogahreyfingar á mælistað í höfninni. Í þessu verkefni er hins vegar verið að nýta fyrirliggjandi mælingar til að vinna úr þeim upplýsingar um öldur og sog. Slíkar upplýsingar eru mikilvægar til að meta viðleguskilyrði skipa í höfnum og nýtast m.a. við líkanagerð tengdum breytingum á höfnum, hvort sem um er að ræða reiknilíkön eða vatnslíkön.

Með því að greina tímaraðir þrýstimælinga og með tíðnigreiningu þeirra má reikna út kennistærðir öldu eins og kenniöldu og sveiflutíma.

Gögn úr þrýstinemum eru unnin með skriftum skrifuðum í Python 3.6. Skrifturnar innihalda annars vegar aðgerðir til að sinna mismunandi útreikningum og eftirvinnslu gagnanna og hins vegar lykkju sem kallar á aðgerðirnar þegar gögn berast úr þrýstinemunum, einu sinni á klukkustund.

Kennistærðirnar eru reiknaðar með tíðnigreiningu. Gögnin eru miðjuð, þjálguð og leiðrétt er fyrir dempun öldu vegna dýpis nema. Ölduróf sem úr þessu fæst er skipt niður í mismunandi ölduhluta: sog, undiröldu og vindöldu og kennistærðir allra ölduhluta reiknaðar.

Útreikningarnir taka aðeins nokkrar sekúndur og eru niðurstöðurnar skráðar í gagnagrunn og birtar á vefnum vedur.mogt.is samstundis og þær verða til.

Með þessum hætti eru rauntímaupplýsingar um viðleguskilyrði skipa í höfnum aðgengilegar og uppfærðar á klukkustundar fresti.

Skýrsla þessi er unnin fyrir styrk frá Rannsóknasjóði Vegagerðarinnar. Höfundur skýrslunnar ber ábyrgð á innihaldi hennar. Niðurstöður hennar ber ekki að túlka sem yfirlýsta stefnu Vegagerðarinnar.

Efnisyfirlit

1 Inngangur

Meginmarkmið þessa verkefnis er að bæta eftirlit með viðleguskilyrðum skipa í höfnum. Þessi skilyrði ráðast af ólíkum þáttum, einkum vindhraða, úthafsöldu, sjávarföllum og legu skjólgarða og kanta. Þar sem þessi skilyrði eru misbreytileg þarf nokkuð náið eftirlit með öldufari.

Í þónokkrum höfnum hefur verið komið fyrir þrýstinemum sem sinna eftirliti með sjávarhæð. Í þessari skýrslu er því lýst hvernig vinna má úr gögnunum sem berast úr þrýstinemunum. Þrýstimælingarnar má nota til að annars vegar aðskilja frumþætti öldunnar, þ.e. sog, vindöldu og undiröldu og hins vegar til þess að reikna út kennistærðir þessara ölduhluta, svo sem kenniöldu, meðal sveiflutíma og sveiflutíma orkutopps (peak period). Sveiflur á yfirborði sjávar í höfnum sem eru lengri en haföldur kallast sog. Á meðan öldur hafa sveiflutíma frá u.þ.b. 3 sekúndum upp í um 25 sekúndur þá eru sogin með sveiflutíma frá um 25 sekúndur upp í nokkra mínútur. Sogin eru sveiflur á eigintíðni hafna og hólfa innan þeirra. Lengd soga fer eftir stærð hafnanna eða hafnarhólfanna. Orkan sem kemur sogum af stað á uppruna sinn í löngum sveiflum á haffletinum sem talin eru tengjast ölduhópum, þ.e. fyrirbærinu „sjaldan er ein báran stök“.

Það má reikna þessar stærðir beint út frá tímaröðum en með því að varpa tímagögnunum á tíðniás má beita leiðréttingum, svo sem leiðréttingu fyrir dempun öldu með dýpi, en þær auka nákvæmni útreikninganna. Að auki má aðgreina fleiri þætti öldufars en þegar unnið er með tímaraðir.

Gögnin eru unnin með skriftum skrifuðum í Python 3.6 sem finna má í viðauka. Skrifturnar innihalda annars vegar aðgerðir til að sinna mismunandi útreikningum og úrvinnslu gagnanna og hins vegar lykkju sem kallar á aðgerðirnar þegar gögn berast úr þrýstinemunum. Skrifaðar voru aðgerðir til útreikninga kennistærða út frá tímaröðum og tíðnigreiningu og niðurstöður skoðaðar fyrir báðar nálganir.

Í kafla 0 er því lýst hvernig unnið er úr tímaröðum úr þrýstinema en í kafla 0 er tíðnigreiningu lýst.

1.1 Mælingar með þrýstinema

Þrýstinemarnir eru frá framleiðandanum Keller. Þeir eru kvarðaðir hjá framleiðanda til að gefa straum sem vísun í hæð vatnssúlu, þannig að 4 – 20 mA straumur svarar til 0 -10 m vatnssúlu. Þar sem mælingarnar eru í sjó en ekki ferskvatni eru þær leiðréttar um 2,5%, þar sem sjórinn er saltur og þar af leiðandi eðlisþyngri. Þessi leiðrétting er byggð á meðalseltu sjávar við Ísland, en hana má líka mæla á uppsetningarstað. Til þess að tryggja að loftþrýstingur inni í mælikerfi nemans sé sá sami og á landi er notuð núllpunktsmæling á þrýsting í gegnum hola slöngu sem áföst er mælinum með opið á þurru.

Straumurinn er skráður jafnóðum og umreiknaður í metra út frá kvörðuninni á nemunum. Svið straums er 16 mA og svið nema 10 m. Nemarnir eru staðsettir um það bil 1 m undir meðal stórstraumsfjöru þannig að aldrei fjari undan þeim. Þannig er dýpi á þrýstinemann breytilegt, allt frá því að vera innan við einn meter, yfir í að vera um og yfir 6 m þar sem mesti munur er á flóði og fjöru.

Breytingar á hæð vatnssúlunnar stafa í fyrsta lagi af sjávarföllum, í öðru lagi af löngum sveiflum í höfnum, sogi, og í þriðja lagi af öldum, allt frá vindölum upp í lengri úthafsöldur.

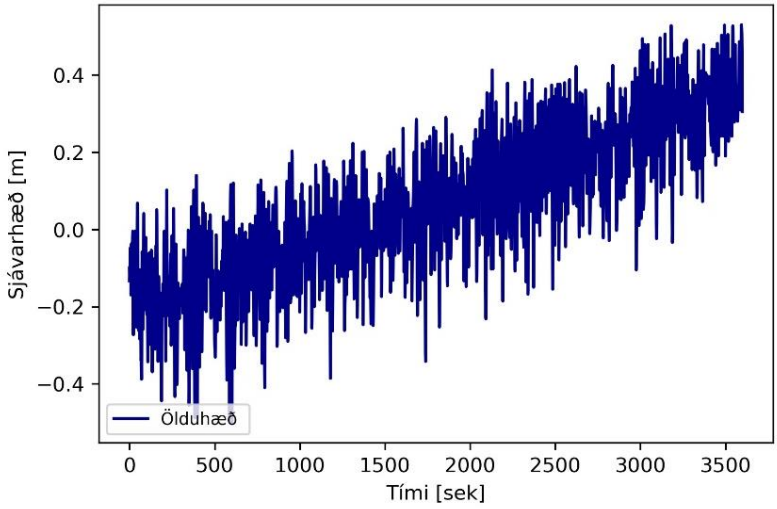
Þrýstingsbreytingar með stuttan sveiflutíma dempast með dýpi. Sjávarföllin, sogin og lengri öldurnar ná ódempuð niður á nokkurra metar dýpi, en styttri vindöldur dempast. Til að lágmarka þessa dempun er þrýstinema komið fyrir eins ofarlega og mögulegt er, án þess þó að hann komi upp úr á allra lægstu sjávarstöðum. Leiðrétt er fyrir þessa dempun í úrvinnslu gagnanna sjá kafla 3.3.

Þrýstineminn skráir þrýsting á sekúndu fresti og er klukkustundar löng tímaröð send út frá nemanum á klukkustundar fresti á miðlægan þjón þar sem gögnin eru skráð í gagnagrunn. Unnið er úr gögnunum á þjóninum og niðurstöður skráðar í gagnagrunn og birtar á vef M og T (vedur.mogt.is).

2 Úrvinnsla tímaraða úr þrýstinema

Lengd tímaraðar þessarar greiningar er ein klukkustund, 3600 mælingar á sekúndu fresti og hentar þessi lengd tímaraðar vel til þess að lýsa öldufari hverju sinni. Gögnin úr þrýstinemunum eru í einingunum metrum yfir núllstöðu (z=0).

Hér fyrir neðan má sá dæmi um klukkustundarlanga tímaröð. Mælipunktar eru 3600. Hægur stígandi er í gögnunum en það eru áhrif sjávarfalla.

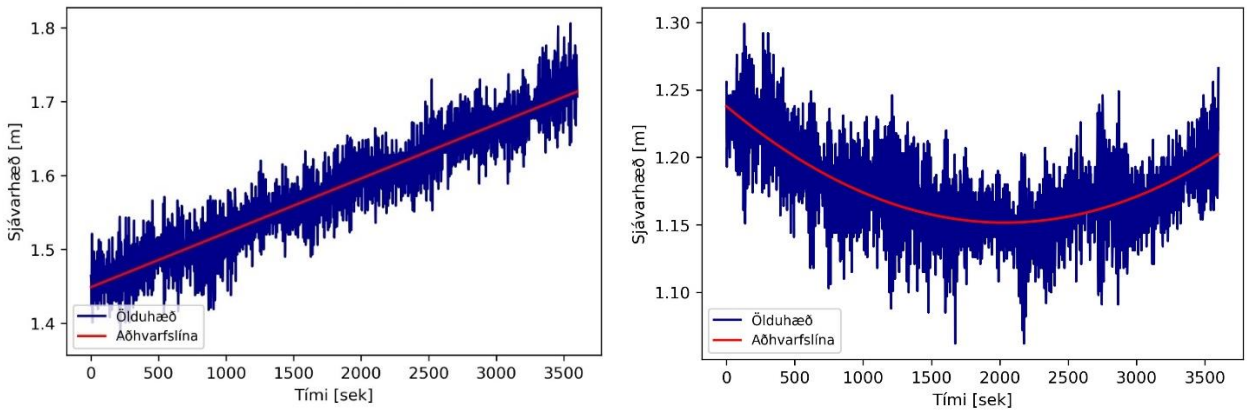


Mynd 1. Dæmi um klukkustundarlanga tímaröð gagna úr þrýstinema.

2.1 Áhrif flóðs og fjöru

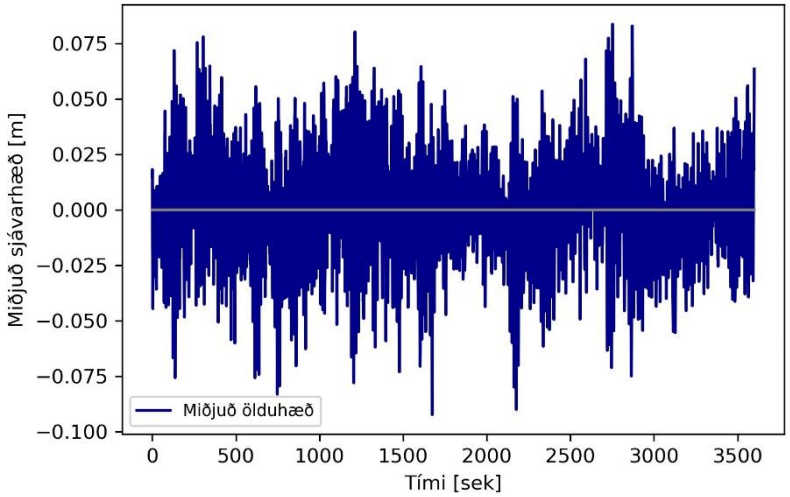
Áhrifa flóðs og fjöru gætir í gögnunum og sést til dæmis á Mynd 2 sem heildarhækkun sjávarhæðar á tímanum 3600 sekúndur (vinstri mynd). Til þess að fjarlægja sjávarfallasveifluna úr gögnunum er aðhvarfslína gagnanna fundin, sem er ýmist bein lína eða annars stigs fall.

Til þess að ákvarða sjálfvirkt hvort hentar frekar, lína eða annars stigs fall, þá gerir úrvinnslan hvortveggja og velur svo þann kostinn sem líkir betur eftir hegðun gagnanna. Þetta er gert með því að bera saman uppsafnaða summu afgangsliða fyrsta og annars stigs fallanna og velja þá minni.



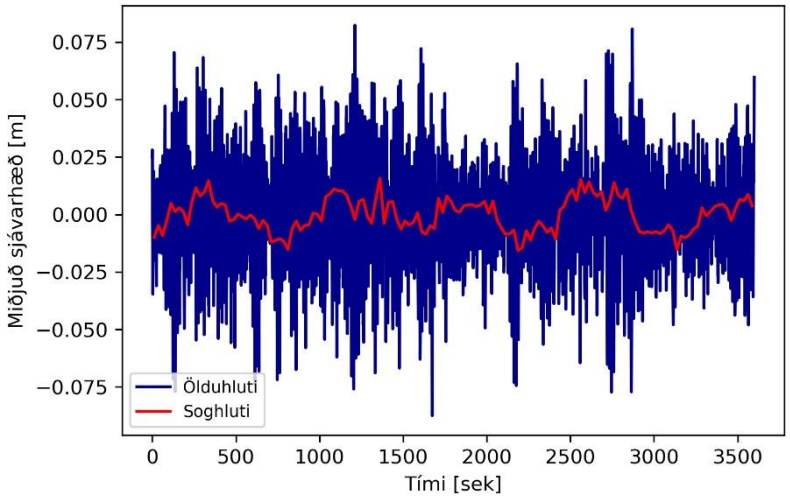
Mynd 2. Dæmi um hvernig fyrsta og annars stigs föll eru notuð til að nálga áhrif sjávarfalla í gögnunum.

Til þess að fjarlægja áhrif sjávarfalla úr gögnunum er aðhvarfslínan er dregin frá gögnunum. Með þessu eru gögnin miðjuð í kringum núllgildi.



2.2 Soghluti og ölduhluti einangraðir

Með því að þjálga eða slétta tímaraðir öldugagna má einangra soghluta þeirra. Þetta er gert þannig að rófinu er skipt upp í glugga, 25 mælipunktar hver og í hverjum glugga er reiknað meðaltal. Tímaupplausn hrörnar því 25falt. Þannig sléttist tímaröðin og samsvarar nú soghluta öldunnar. Með því að draga soghluta öldunnar frá heildargögnunum fæst ölduhlutinn, sjá Mynd 3.



Mynd 3. Ölduhluti og soghluti tímaraða öldugagna einangraðir.

2.3 Kennialda, meðalsveiflutími, mesta ölduhæð, hæsta og lægsta gildi ölduhluta og soghluta

Meðal þeirra kennistærða sem reikna má út úr tímarunu öldugagna eru kennialda og meðal sveiflutími. Kennialda (H_s, enska: *significant wave height*) er reiknuð með því að margfalda staðalfrávik tímaraðar með fjórum. Þannig má reikna hvort heldur kenniöldu soga eða ölduhluta.

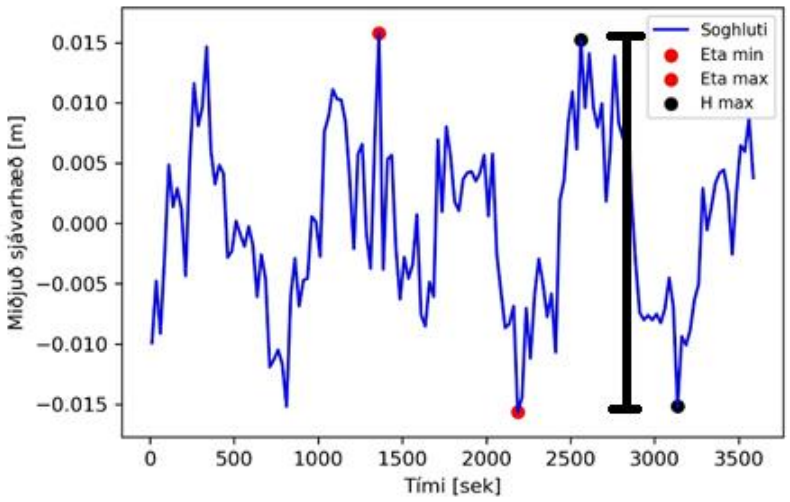
$$H_s = 4 * \sigma$$

Þar sem σ er staðalfrávik miðjaðra gagna (tímaraðarinnar).

Meðallotu öldu eða soga (T_z, enska: *mean wave period*) má reikna með því að deila lengd merkis (hér 3600 sek) með fjölda alda í merkinu. Fjöldi alda er fundinn með talningu þess hve oft hið miðjaða öldumerki sker x-ás. Hægt er að telja hversu oft merkið sker ásinn annars vegar þegar merkið fer minnkandi og hins vegar þegar það fer hækkandi (enska: *zero up crossin, zero down crossing*). Niðurstöður eru sambærilegar hvort heldur er notað. Með því að deila tímanum sem merkið varir með fjölda skurðpunkta fæst meðalsveiflutími merkisins, hún hefur eininguna sekúndur.

$$T_p = Lengd\ merkis / Fjöldi\ skurðpunkta$$

Aðrar stærðir sem greina má úr miðjuðum tímaröðum öldugagna eru: Mesta ölduhæð (H_{max}) sem fundin er með því að leita að mesta mun á aðliggjandi hápunkti og lágpunkti. Hæsta gildi (η_{min}) er fundið með því að leita að hæsta gildi miðjaðrar tímaraðar. Lægsta gildi (η_{max}) er fundið með því að leita að lægsta gildi miðjaðrar tímaraðar. Í úrvinnslunni sem hér er lýst eru þessar stærðir aðeins reiknaðar fyrir soghluta, sjá Mynd 4. (Kamphuis, 2000)



Mynd 4. Á myndinni má sjá miðjaðan soghluta öldu (blá lína). Rauðir punktar tákna hámark og lágmark soghluta og milli svörtu punktanna tveggja liggur hámarksölduhæð, merkt með svartri stiku.

3 Öldurófsgreining

Öldurófsgreining er það kallað þegar unnið er með öldugögn í tíðnirúmi. Ef tímaröðum úr þrýstimælunum er varpað á tíðniás fæst tíðniróf gagnanna. Rófinu má skipta niður í hluta, s.s. vindöldu, undiröldu og sog og í kjölfarið má reikna út kennistærðir þeirra. Þegar gögnin eru komin á tíðniás er hægt að leiðrétta fyrir dempunaráhrif sem hafa mest áhrif á hærri öldutíðnir og sem aukast með dýpi nemans. Öldurófsgreining býður einnig upp á það umfram tímaraðagreiningu að aðgreina öldurófið í fleiri hluta en tímagreining gerir. Vegna alls þessa eru kennistærðir fengnar úr öldurófsgreiningu teknar framyfir sömu kennistærðir fengnar úr tímaraðagreiningu.

Hér að neðan er því líst hvernig ölduróf er fengið úr tímaröðum, hvernig gögnin eru unnin og hvernig kennistærðir vindöldu, undiröldu og sogs eru reiknaðar.

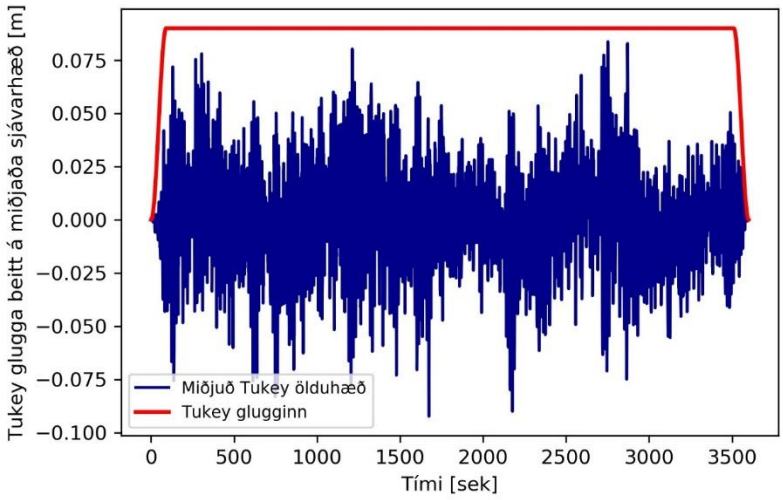
3.1 Gögnin síuð með Tukey glugga

Áður en tímagögnunum er varpar á tíðnirófi er Tukey glugga beitt á öldumælingarnar og deyfir hann fyrstu og síðustu 5% merkisins.

Sjávaröldur hafa ölduróf sem er samfellt og í mældu merki af lengd *D* hafa ekki allar öldur tíðni sem samsvarar einni yfirsveiflu í bilinu *D*. Ef bylgjulengd hefur ekki tíðni sem samsvarar yfirsveiflu *D* getur orka bylgjunnar lekið til næstu yfirsveiflu eða enn fjarlægari yfirsveifla. Gluggaföll eins og Tukey glugginn eru notuð til þess að minnka slíkan leka. Þá er öldumerkið margfaldað með falli sem minnkar ölduhæð við byrjun og enda mælingarinnar. Tukey gluggi hefur þann kost að hafa minnst áhrif á upprunalegt öldumerki og er því beitt hér. Tukey glugganum má lýsa þannig að hann er hefur gildið 1 alls staðar nema við byrjun og enda merkisins en þar hefur hann form kósínusbylgju.

$$f(k) = \begin{cases} \frac{1}{2}\left[1 + cos\left(\frac{2\pi}{r}\frac{(k-1)}{(N-1)} - \pi\right)\right] & k < \frac{r}{2}(N-1) + 1 \\ 1 & \frac{r}{2}(N-1) + 1 < k < N - \frac{r}{2}(N-1) \\ \frac{1}{2}\left[1 + cos\left(\frac{2\pi}{r} - \frac{2\pi}{r}\frac{(k-1)}{(N-1)} - \pi\right)\right] & k > N - \frac{r}{2}(N-1) \end{cases}$$

Þar sem *k* er vísir á hvern gagnapunkt, *r* er hlutfall dempunar (hér 5%, 2,5% á hvorum enda) og *N* er lengd tímaraðar.
Glugginn margfaldar tvíveldisfrávik merkisins með stuðlinum $G = 1 - 5 * r/8$. Það þarf að gæta þess að deila lokarófinu (þjálgað róf) með *G*. (Casas-Prat, 2008)



Mynd 5. Tukey glugginn, rauð lína á myndinni, sem beitt er á öldumælingarnar deyfir fyrstu og síðustu 2,5% merkisins. Gildi miðhluta gluggans er 1 en til að geta sýnt hann með merkinu á þessari mynd hefur það verið minnkað niður í 0,09.

3.2 Ölduróf

Ölduróf er fengið með því að að beita Fourier vörpun á tímaraðir. Fourier vörpun varpar tímamerki á tíðniás.

Öldumerkið (η_j) er ekki samfellt heldur samanstendur það af tímaröð þar sem mæligildum er safnað N sinnum á Δt fresti (hér 3600 sinnum á 1 sekúndu fresti). Þar sem tímaöðin er ekki samfellt fall verður tíðnirrúmið sem henni er varpað í ekki samfellt heldur. Lægsta tíðnin sem skilgreina má $f_{min} = 1/t_R = 1/3600 = 2.77 * 10^{-4}$. Hámarksupplausn öldurófs næst með því að nota þessa lágmarkstíðni sem skreflengd á tíðniás. Tímaskrefið, Δt , sem er 1 sekúnda hér, ákvarðar svo efri mörk tíðnirófsins, kölluð Nyquist tíðnin, skilgreind með þessum hætti:

$$f_{Nyq} = 1/(2 * \Delta t)$$

Nyquist tíðni öldugagnanna er 0,5 Hz.

Fourier-vörpunin skiptir falli upp í hina mismunandi tíðniþætti sína og skilar útslagi og fasa fyrir hvern tíðniþátt. Fyrir stakstæð gögn eins og tímaraðir er Fourier vörpun

$$\eta_j = \sum_{n=-\frac{N}{2}+1}^{\frac{N}{2}} F_n e^{i[2\pi(f_n t_j)]}$$

Þar sem η_j táknar tímaröðina, N táknar lengd tímaraðar, n er vísir, F_n er tíðnifallið á tvinntöluformi, f_n er tíðni og $t_j = j * \Delta t$. Útkoman úr Fourier vörpun er því á tvinntöluformi og rófið er samhverft um núll (sjá **Error! Reference source not found.**). Þ ar sem F_n er tvinntala gildir að $F_n = F_{-n}{}^*$ og því má rita jöfnuna hér að ofan sem summu frá n=0 til N-1. Einnig gildir um tvinntöluföll eins og F_n að $F_n = |F_n| * e^{-i\theta_n}$, en fyrri liðurinn er útslagsþáttur og síðari er fasaþátturinn:

$$\eta_j = \sum_{n=0}^{N-1} |F_n| e^{i[2\pi(f_n t_j) - \theta_n]}$$

Öldur hafa hins vegar bara jákvæða tíðni og eru rauntölur þannig að hvorki þvertölu né neikvæði hluti rófsins eru notaðir.

$$\eta_j = \sum_{n=0}^{N-1} |F_n| \cos(2\pi f t - \theta_n) = \sum_{n=0}^{N/2} |A_n| \cos(2\pi f t - \theta_n)$$

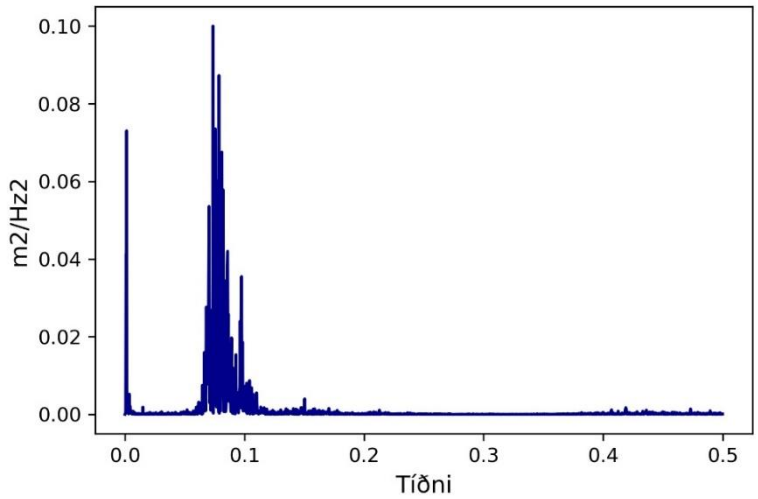
En hér gildir að $2 * |F_n| = |A_n|$ þar sem $|A_n|$ er styrkróf öldunnar. Fasi öldugagna er slembibreyta og því er það styrkrófið sem notað er. Út frá því má reikna öldurófið $S(f_n)$ (enska: *power spectral density eða wave variance spectral density*):

$$S(f_n) = \frac{1}{2f_{min}} |A_n|^2$$

Tvífrávik öldurófs er fundið með:

$$\sigma^2 = \sum_{n=0}^{N/2} \frac{1}{2} |A_n|^2$$

Mikilvægar kennistærðir öldurófs má svo reikna út frá öldurófinu og staðalfráviki þess, en nánar verður fjallað um það í kafla 3.5.(Kamphuis, 2000)



Mynd 6. Dæmi um ölduróf. Aðeins jákvæði hluti gagnanna er skoðaður, frá núlli og upp að Nyquist tíðninni sem hér er 0,5 Hz.

3.3 Leiðrétting fyrir deyfingu merkis með sjávarstöðu

Því styttri sem bylgjulengd öldu á yfirborði sjávar er (því hærri sem tíðni hennar er) því meira dempast merkið frá henni með dýpi. Þar sem öldugögnunum er safnað í klukkustundarlöngum tímaröðum og sjávarstaða er talsvert breytileg á slíkum tímaskala þarf að reikna út hver dempunin er fyrir hverja tímaröð sem send er úr nemanum og leiðrétta hverja tímaröð.

Leiðréttingarfallið, K_p, er skilgreint á eftirfarandi hátt (sjá Mynd 7):

$$K_p = \cosh\big(k * (z + d)\big)/\cosh k * d$$

Þar sem:

- z er dýpi niður á nemann
- d er raundýpi, heildardýpi frá núll-sjávarstöðu niður á sjávarbotn
- k er öldustuðull (enska: wave number) = 2π/L
- L er öldulengd

Reikniaðgerðin fyrir þessa leiðréttingu notast við upplýsingar um dýpi á nema og raundýpi úr inntakskrá sem geymir þessar upplýsingar fyrir mismunandi hafnir.

Til þess að finna Kp þarf að byrja á því að áætla k, öldustuðul. k er fall af öldulengd, $k = 2\pi\ /L$ og öldulengd er skilgreind:$L = C * T$ þar sem C táknar útbreiðsluhraða öldu (m/s) og T er sveiflutími öldu (s).

Stuðullinn g er þyngdarhröðun. Af því leiðir að öldustuðull er fall af tíðni öldu.

$$k = \frac{2\pi}{CT}$$

Sveiflutíminn, T, samsvarar 1/tíðni öldunnar. Tíðnin eru þá einfaldlega gildin á x-ás öldurófsins. Útbreiðsluhraðann, C, má finna út frá nálguninni:

$$\frac{C^2}{gd} = [y_0 + (1 + 0{,}6522y_0 + 0{,}4622{y_0}^2 + 0{,}0864{y_0}^4 + 0{,}0675{y_0}^4)^{-1}]^{-1}$$

Þar sem:

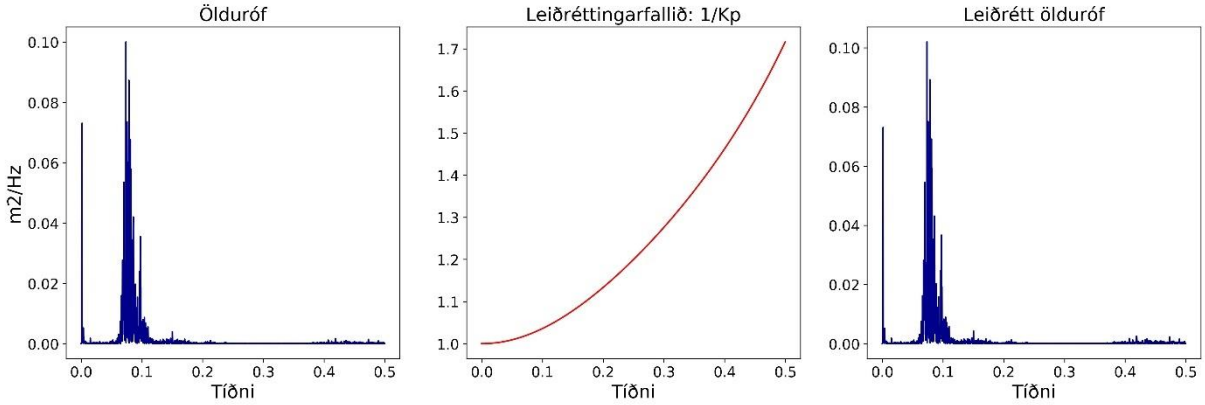
$$y_0 = \frac{2\pi d}{L_0} = \frac{(2\pi)^2 d}{gT^2}$$

Þar sem L_0 er skilgreint:

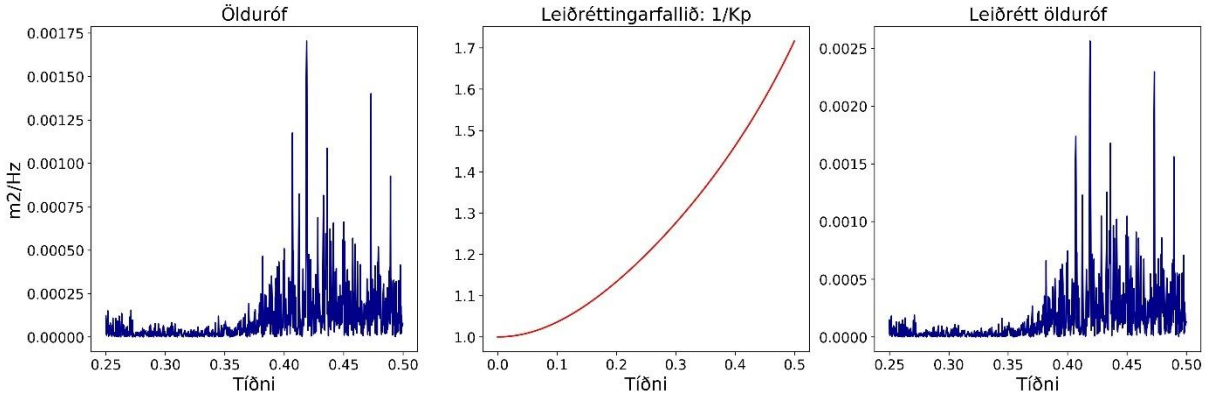
$$L_0 = \frac{gT^2}{2\pi}$$

Þegar k hefur verið fundið fyrir hverja tíðni má reikna K_p, en þar sem það veltur bæði á dýpi á nema og raundýpi við nema. Þessar stærðir þarf að leiðrétta fyrir sjávarstöðu á hverjum tíma. Þar sem raundýpi er áætlað fá lægstu sjávarstöðu og niður á botn þarf að bæta við það sjávarhæð á þeim tíma sem reiknað er fyrir. Þetta er gert með því að taka meðaltal af sjávarhæðargögnunum úr prýstinemanum fyrir við komandi höfn á viðkomandi tíma og leggja það meðaltal við raundýpið og við dýpi niður á nemann.

Leiðréttingin fyrir dempun öldu sem fall af sjávarstöðu og tíðin öldu er nú gerð með því að margfalda öldurófið með $1/K_p$, sem er 1 fyrir lægstu tíðnirnar en fer hækkandi fyrir hækkandi tíðnir, sjá Mynd 7. Þetta er gert fyrir öll gildi af $1/K_p$upp að 5 en það er hámarksleiðréttingin. Fyrir gildi af $1/K_p$ hærri en 5 er einfaldlega margfaldað með 5. (Kamphuis, 2000)



Mynd 7. Ölduróf, vinstri mynd. Leiðréttingarfallið 1/Kp, miðmynd. Leiðrétt ölduróf, hægri mynd.



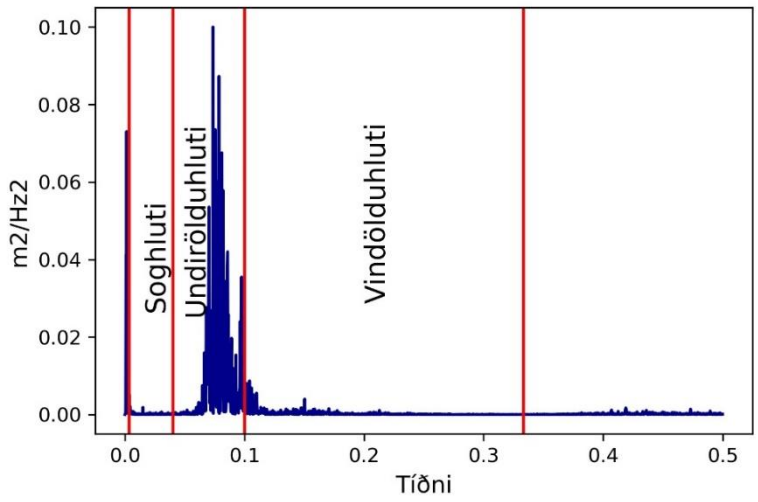
Mynd 8. Þysjað að hærri tíðnum öldurófs þar sem leiðrétting vegna dempunar hefur meiri áhrif. Frá vinstri til hægri: Ölduróf, leiðréttingarfallið 1/Kp, leiðrétt ölduróf.

3.4 Undirhlutar öldurófs: Sog, undiralda og vindalda

Öldurófinu má skipta upp eftir tíðni (bylgjulengd) ölduhluta:

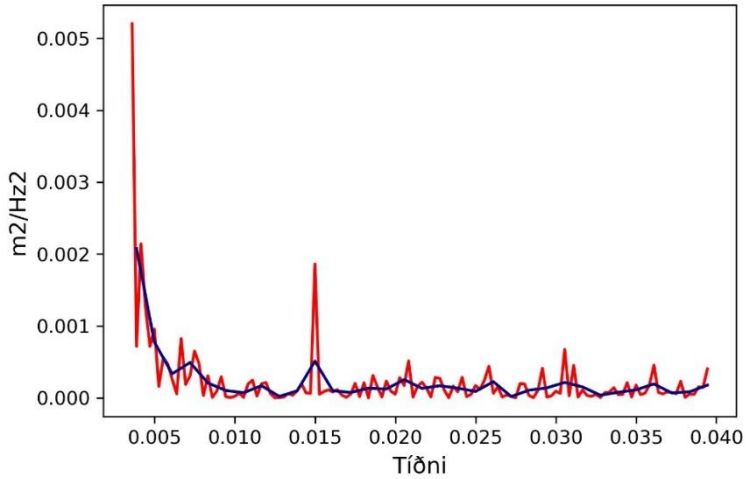
- Soghluti er frá 0,0033 Hz til 0,04 Hz.
- Undiralda er frá 0,04 Hz til 0,1 Hz
- Vindalda er frá 0,1 Hz til 0,33 Hz

Sá hluti rófsins sem liggur utan þessara marka (<0,0033 og >0,33) er ekki skoðaður frekar.

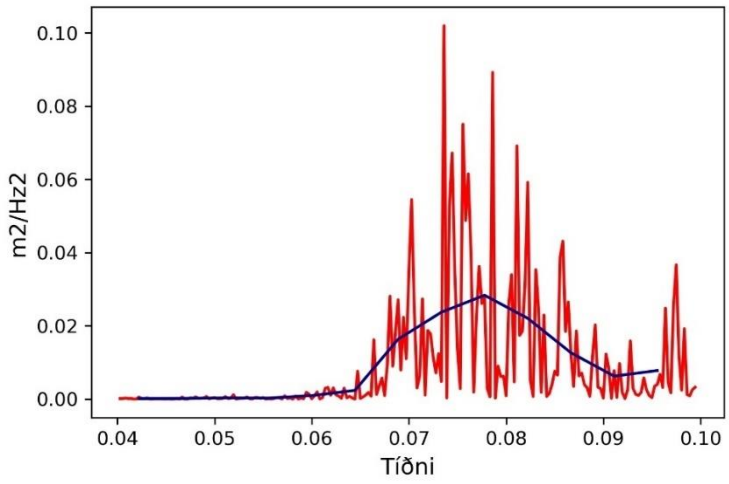


Mynd 9. Rauðu línurnar á myndinni sýna mörk sog-, undiröldu- og vindölduhluta öldurófs.

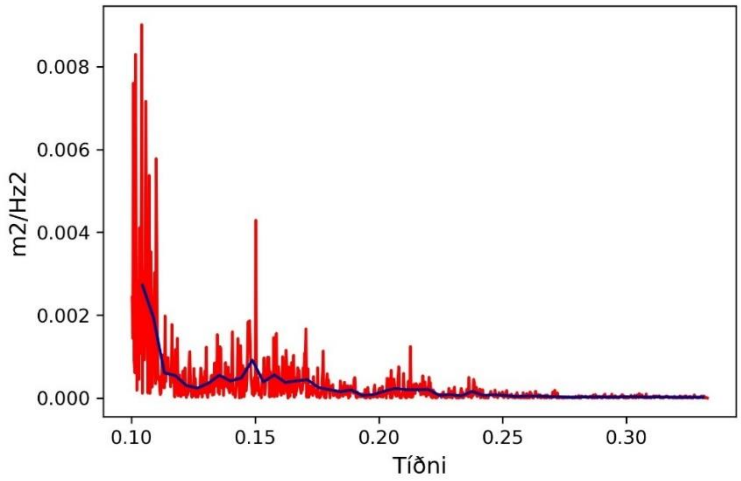
Undirhlutar öldurófsins eru þjálgaðir (enska: smoothed) mismikið. Þjálgunin er framkvæmd með hlaupandi meðaltalsglugga. Hann virkar þannig að fyrir hvert gildi er glugganum stillt upp með gildið í miðjum glugga og meðaltal allra gilda í glugganum er notað sem nýtt miðjugildi. Soghluti er þjálgaður með 0.0022 Hz víðum glugga (4 punkta víðum) á meðan vindöldu og undirölduhlutar eru þjálgaðir með 0.0089 Hz víðum glugga (16 punkta víðum). Stærð þjálgunarglugga hefur töluverð áhrif á hve há hæstu gildi rófsins verða en óveruleg áhrif á heildi rófsins (flatarmál undir rófinu).



Mynd 10. Soghluti öldurófs óþjálgaður (rauð lína) og þjálgaður með 4 punkta breiðum glugga (blá lína).



Mynd 11. Undirölduhluti öldurófs óþjálgaður (rauð lína) og þjálgaður með 16 punkta breiðum glugga (blá lína).



Mynd 12. Vindölduhluti öldurófs óþjálgaður (rauð lína) og þjálgaður með 16 punkta breiðum glugga (blá lína).

3.5 Kennialda, meðalsveiflutími og sveiflutími orkutopps.

Við útreikninga kennistærða öldurófsins er notast við vægi rófs, m_h, (enska: moment) sem er reiknað á eftirfarandi hátt:

$$m_h = \int_{f=0}^{f=\infty} f^h S(f) df$$

Núllta vægi, m_0, samsvarar flatarmálinu undir rófinu:

$$m_0 = \int_{f=0}^{f=\infty} S(f) df = \sigma_f^2$$

Kennialda er fundin út frá flatarmálinu undir öldurófinu:

$$H_s = 4 * \sigma_f$$

Þar sem

$$\sigma_f = \sqrt{m_0}$$

Meðalsveiflutími, Tz, er fundin út frá flatarmáli undir öldurófi sem og öðru vægi rófsins.

$$T_z = \sqrt{\frac{m_0}{m_2}}$$

Sveiflutími orkutopps, Tp, er fundin út frá því tíðnigildi þar sem hæsta gildi þjálgaðs öldurófs liggur (f_p).

$$T_p = \frac{1}{f_p}$$

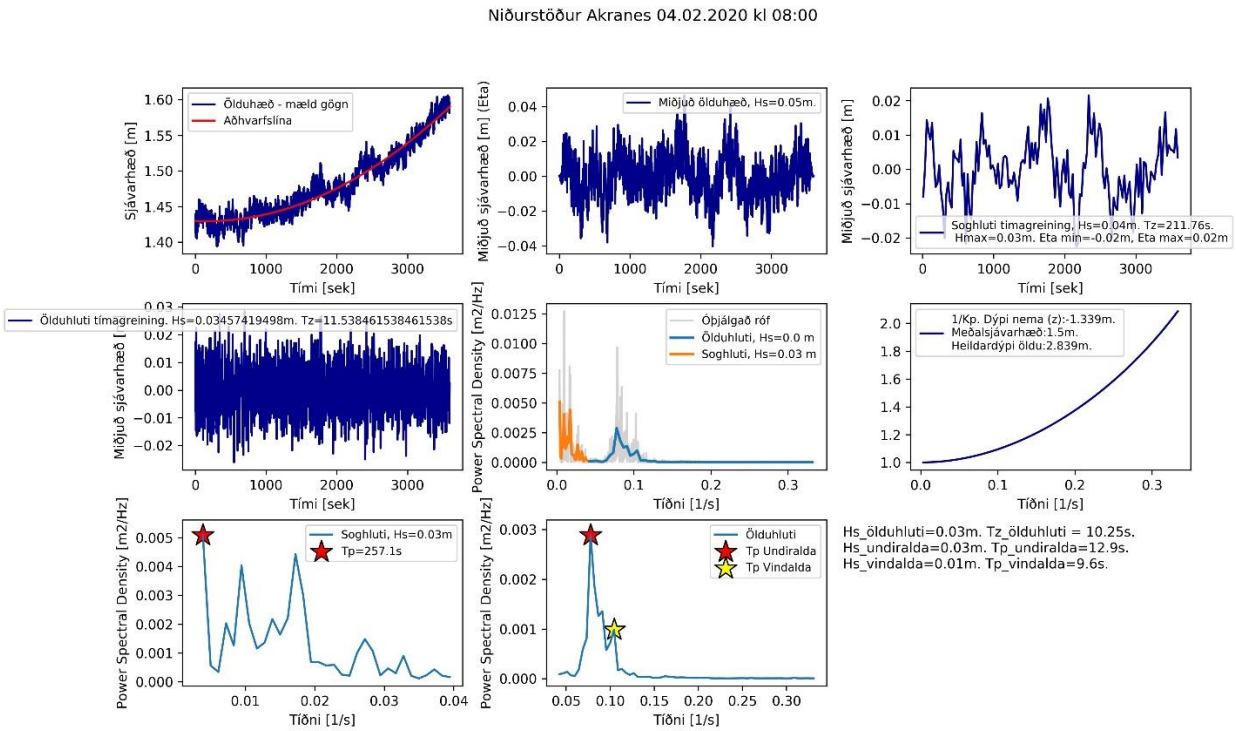
4 Úrvinnsla gagna, frágangur og birting niðurstaðna

4.1 Uppbygging úrvinnsluskriftu

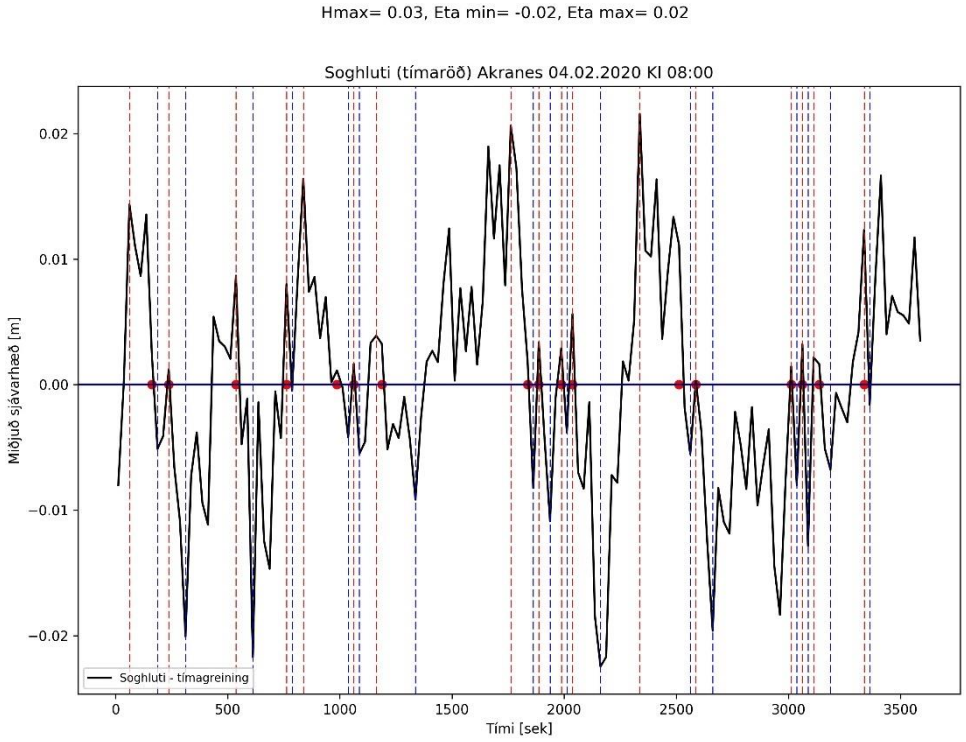
Úrvinnsluskriftan er þannig upp byggð að einu sinni á klukkustund, þegar gögn berast úr þrýstinemum á miðlægan þjón, er hún látin keyra. Úrvinnslulykkjan byrjar á því að kalla á skrá sem inniheldur inntaksstærðir sem eru breytilegar milli hafna eða sem breyta má til að aðlaga úrvinnslu að aðstæðum í hverri höfn. Þessar stærðir eru:

- Byrjun vindöldu á rófi
- Byrjun undiröldu á rófi
- Byrjun soghluta á rófi
- Endir soghluta á rófi
- Vídd þjálgunarglugga til notkunar á soghluta tíðnirófs
- Vídd þjálgunarglugga til notkunar á ölduhluta tíðnirófs.
- Dýpi á nema (í m undir stórstraumsfjöru)
- Raundýpi við nema (m)

Úrvinnslulykkjan les inn allar upplýsingar um hafnirnar sem finna má í skránni en reiknar úr gögnum fyrir eina höfn í einu. Það tekur innan við sekúndu að reikna úr gögnum fyrir eina höfn.
Lykkjan leitar að gögnum fyrir hverja höfn. Finni hún ekki gögn fyrir ákveðna höfn skilar hún niðurstöðum sem gefa skýrt til greina að ekki hafi fundist ein gögn, þar sem öllum niðurstöðustærðum er gefið gildið -6999. Gagnagrunnur MogT skráir í kjölfarið gildið NAN fyrir allar stærðir úr höfninni þar sem gögn ekki fundust.
Lykkjan gerir útreikninga á miðjuðum tímaröðum og skilar meðalsveiflutíma og kenniöldu fyrir sog og ölduhluta tímaraða. Einnig skilar hún hæsta og lægsta gildi soghluta sem og mestu ölduhæð soghluta.
Næst er tíðnigreining sem skilar kenniöldu, meðalsveiflu og mestu sveiflu vindöldu, undiröldu og sogs. Að lokum skilar úrvinnslulykkjan tveimur niðurstöðumyndum sé þess óskað, sjá Mynd 14Mynd 15.



Mynd 13. Dæmi um niðurstöðumynd þar sem birtar eru kennistærðir ölduhluta fengnar úr tíma- og tíðnigreiningu.



Mynd 14. Dæmi um mynd sem sýnir talningu skurðpunkta soghluta við x-ás en hún er notuð við útreikninga meðalsveiflutíma sogs.

4.2 Frágangur og birting niðurstaðna

Allar stærðir reiknaðar af úrvinnsluskriftunni eru sendar til geymslu í gagnagrunn MogT.

Nokkar lykilstærðir fengnar úr tíðnigreiningu öldugagna eru birtar á vef MogT (vedur.mogt.is), þær eru eftirfarandi:

- kennialda sogs, vindöldu og undirölduhluta
- sveiflutími sogs, vindöldu og undirölduhluta

Þessar stærðir eru mikilvægar þegar kemur að því að meta viðleguskilyrði í höfnum.

Gögn berast frá þrýstinemunum einu sinni á klukkustund og er reiknað út úr þeim inná heila tímanum og niðurstöður birtar jafnóðum og þær verða til á vef MogT.

Akranes öldumælingar

Dagsetning	Timi	Sog Hs (m)	Sog Tp (s)	Alda Hs (m)	Undiralda Hs (m)	Undiralda Tp (s)	Vindalda Hs (m)	Vindalda Tp (s)
17-05-2020	22:00	0.04	138	0.19	0.07	11.5	0.17	3.1
17-05-2020	21:00	0.05	257	0.19	0.07	12.2	0.17	3.1
17-05-2020	20:00	0.05	72	0.19	0.07	11.5	0.18	3.1
17-05-2020	19:00	0.05	257	0.20	0.07	11.5	0.18	3.1
17-05-2020	18:00	0.05	257	0.20	0.07	11.5	0.18	3.1
17-05-2020	17:00	0.02	95	0.20	0.07	12.2	0.19	3.1

Mynd 15. Dæmi um framsetningu niðurstaðna á vef MogT.

5 Heimildir

Casas-Prat, Mercè. (2008). Overview of ocean wave statistics.

Kamphuis, J.W. (2000). Advanced Series on Ocean Engineering - Volume 16: Introduction to Coastal Engineering and Management. World Scientific Publishing Co. Pte. Ltd.

6 Viðauki - Úrvinnslukóðar

6.1 Aðgerðir

```
from matplotlib import pyplot as plt
import numpy as np
import scipy
import pandas as pd
import math
from math import log10, floor
import os.path

def read_tidedatafile(datapath, datafile):
    if os.path.isfile(datapath+datafile)==True:
        input_datafile=datapath+datafile
        # Les gögnin inn
        tide_data_dict = {}
        tide_data_df= pd.read_csv(input_datafile, sep=',', names=['time','height'])
        tide_data_df=tide_data_df.dropna()
        tide_data_dict['height']= np.array(tide_data_df['height'])
        # Tíminn er bara 0 svo ég bý til array of 3600 sekúndum
        if len(tide_data_dict['height'])==3600:
            tide_data_dict['time'] = np.arange(len(tide_data_dict['height']), dtype =int)
        else:
            tide_data_dict='nan'
    else:
        tide_data_dict='nan'
    return(tide_data_dict)

def normalise_tidedata(tide_data_dict):
    # Miðja gögnin: Finn aðhvarfslínu gagnanna og dreg hana frá þeim
    z1=np.polyfit(tide_data_dict['time'], tide_data_dict['height'],1,  full = True)
    z2=np.polyfit(tide_data_dict['time'], tide_data_dict['height'],2,  full = True)
    first_order_function = np.zeros(len(tide_data_dict['time']))
    second_order_function = np.zeros(len(tide_data_dict['time']))
    residuals_1st_order = z1[1]
    residuals_2nd_order = z2[1]
    for t in tide_data_dict['time']:
        first_order_function[t] = z1[0][0]*t + z1[0][1]
        second_order_function[t] = z2[0][0]*(t**2)+z2[0][1]*t+z2[0][2]
    if residuals_1st_order < residuals_2nd_order:
        correction_function = first_order_function
        correction = '1st order fit' #1. stigs, línulega aðhvarfslínu'
    elif residuals_1st_order > residuals_2nd_order:
        correction_function = second_order_function
        correction = '2nd order fit' #. stigs fjölliðu'
    else: print('There is a problem with the first and second order fit to the data')
    tide_data_dict['correction_function'] = correction_function
    tide_data_dict['eta'] = tide_data_dict['height']-correction_function
    return(tide_data_dict)

def tukey_window_eta(tide_data_dict, r):
```

```
    N=len(tide_data_dict['eta'])
    signal_Tukey = np.zeros(N)
    #r=0.05 # proportion of tapering
    lower_end_list=[]
    upper_end_list=[]
    middle_list=[]
    for k in np.arange(N):
        if k < ((((N-1)*r/2))-1):
            lower_end_list.append(k)
            signal_Tukey[k]=0.5*(1.0+np.cos((2.0*np.pi/r)*((k-1.0)/(N-1.0))-
np.pi))*tide_data_dict['eta'][k]
        elif k > (N-(r/2)*(N-1)):
            upper_end_list.append(k)
            signal_Tukey[k]=0.5*(1.0+np.cos((2.0*np.pi/r)-(2.0*np.pi/r*((k-1.0)/(N-1.0)))-
np.pi))*tide_data_dict['eta'][k]
        else:
            middle_list.append(k)
            signal_Tukey[k]=tide_data_dict['eta'][k]
    return(signal_Tukey)

def smooth_xy(window_width, x, y ):
    half_window_width=int(window_width/2)
    y_smoothed = []
    x_smoothed = []
    for inst in np.arange(half_window_width, y.size, window_width):
        y_smoothed.append(np.mean(y[inst-
(half_window_width):inst+(half_window_width)]))
        x_smoothed.append(x[inst])
    return(np.array(x_smoothed), np.array(y_smoothed))

# Hs : Kennialda / Significant wave height: Hs = 4*sigma
def Hs_timagreining(merki):
    sigma_t = np.std(merki)
    return(4*sigma_t)

# Tz: Mean period
def Tz_timagreining(tide_data_dict, smoothing=0):
    zc = zero_down_crossings(tide_data_dict)
    return((len(tide_data_dict['time']))/(zc))

def zero_down_crossings(tide_data_dict):
    zero_down_counter = 0.0
    for i in np.arange(len(tide_data_dict['eta'])-1):
        if (tide_data_dict['eta'][i] > 0) and (tide_data_dict['eta'][i+1] < 0):
            zero_down_counter=zero_down_counter+1
    return(zero_down_counter)

def zero_up_crossings(tide_data_dict):
    zero_up_counter = 0.0
    for i in np.arange(len(tide_data_dict['eta'])-1):
        if (tide_data_dict['eta'][i] < 0) and (tide_data_dict['eta'][i+1] > 0):
            zero_up_counter=zero_up_counter+1
    return(zero_up_counter)
```

```
def soghluti_timagreining(tide_data_dict):
    return(smooth_xy(25, tide_data_dict['time'],tide_data_dict['eta'] ))

def olduhluti_timagreining(tide_data_dict, sog_timarod):
    #olduhluti_timarod={}
    alda= np.zeros(len(tide_data_dict['height']))
    for i in np.arange(0,len(sog_timarod['time'])):
        therange = (np.arange(sog_timarod['time'][i]-12, sog_timarod['time'][i]+13))
        alda[therange] = tide_data_dict['eta'][therange]-sog_timarod['eta'][i]
    return(np.arange(len(tide_data_dict['height'])), alda)

def Sog_timagreining_Hmax_eta_min_max(sog_timarod):
    # Aðferð: Finna min og max í hverju bili milli zero down crossings.
    zero_down_counter = 0
    current_zdc_point_index = 0
    previous_zdc_point_index = 0
    closest_min_x = []
    closest_min_y = []
    closest_max_x = []
    closest_max_y = []
    for i in np.arange(len(sog_timarod['eta'])-1):
        if (sog_timarod['eta'][i] > 0) and (sog_timarod['eta'][i+1] < 0):
            zero_down_counter=zero_down_counter+1
            current_zdc_point_index = i
            plt.scatter(sog_timarod['time'][i],0, c='red')
            localmax =
np.max(sog_timarod['eta'][previous_zdc_point_index:current_zdc_point_index+1])
            closest_max_y.append(localmax)
            if previous_zdc_point_index > 0:
                localmin =
np.min(sog_timarod['eta'][previous_zdc_point_index:current_zdc_point_index+1])
                closest_min_y.append(localmin)
            previous_zdc_point_index=current_zdc_point_index+1

    lastmin =
np.min(sog_timarod['eta'][current_zdc_point_index+1:len(sog_timarod['eta'])])
    closest_min_y.append(lastmin)
    Hmax_array=np.array(closest_max_y)-np.array(closest_min_y)
    Hmax, Eta_min, Eta_max= np.max(Hmax_array), np.min(np.array(closest_min_y)),
np.max(np.array(closest_max_y))
    return(Hmax, Eta_min, Eta_max)

def PSD_rofgreining(r, tide_data_dict):
    tR = max(tide_data_dict['time']+1) # length of wave record in sec
    N = len(tide_data_dict['eta_Tukey']) # number of points in wave record
    delta_t = np.float(tide_data_dict['time'][1]-tide_data_dict['time'][0])
    fmin=np.float(1/(N*delta_t))
    fNyq = 1/(2*delta_t)
    fN = N/(2*tR)
    # Bý til array af tíðnum, alls 3600, með 1/3600 Hz millibili
    f_array = np.fft.fftfreq(N, delta_t)
    # Reikna fourier transform af Tukey gluggaða merkinu
```

```
    fft_complex = np.fft.fft(tide_data_dict['eta_Tukey'])
    fft_pos=fft_complex[0:int(N/2)]
    ## bls 66-67 í Kamphuis
    A=(2*np.sqrt(fft_pos*fft_pos.conj()))/N # Amplitude spectrum
    A = A.real
    PSD = (A*A)/(2*fmin)  # Wave Variance (Power) Spectral Density or Wave
Spectrum
    # Leiðrétti spektrið fyrir þá dempun sem varð vegna Tukey gluggans (G=1-(5r/8))
    G=1-((5.0*r)/8.0)
    PSD= PSD/G
    return(f_array[0:int(N/2)], PSD)

def inv_Kp(d, z, tide_data_dict):
    mean_sea_height = np.mean(tide_data_dict['height'])
    d=d+mean_sea_height
    z=-1*(mean_sea_height-z)
    g = 9.8
    Kp_inv = []
    x_vals=tide_data_dict['f_array']
    for x in x_vals:  #tíðnir
        y0 = (np.pi**2)*(d/g)*(x**2)
        fy0 = (y0+((1+ 0.6522*y0+0.4622*(y0**2)+0.0864*y0**4+0.0675*y0**5)**-
1))**-1  #=C2/gd
        c = np.sqrt(g*d*fy0)
        k = (2*np.pi/c)*x
        kp=np.cosh(k*(d+z))/np.cosh(k*d)
        if 1/kp < 5.0:
            Kp_inv.append(1/kp)
        else:
            Kp_inv.append(5.0)
    return(np.array(Kp_inv))

def rofhluti(tide_data_dict, smoothing_window_width, start_sec, end_sec):
    freq_smoothed, ps_smoothed =
smooth_xy(smoothing_window_width,tide_data_dict['f_array'],
tide_data_dict['PSD_leidrett_Kp'] )
    index_max=np.max(np.where(freq_smoothed<(1/start_sec)))
    index_min=np.min(np.where(freq_smoothed>(1/end_sec)))
    rofhluti_dict={}
    rofhluti_dict['ps_smoothed'] = ps_smoothed[index_min:index_max+1]
    rofhluti_dict['f_array_smoothed'] = freq_smoothed[index_min:index_max+1]
    return(rofhluti_dict)

def Hs_rofgreining(rofhluti_dict):
    sigma= np.sqrt(np.trapz(rofhluti_dict['ps_smoothed'],
rofhluti_dict['f_array_smoothed'] ))
    return(sigma*4)

def Tp_punktur(freq_smoothed, ps_smoothed):
    gradients=np.diff(ps_smoothed)
    sorted_indices = np.argsort(np.abs(gradients))
    upper99pc_index_list=[] #skoða bara punkta sem eru í topp 99% af stærð gilda
    for upper99pc_index in sorted_indices:
```

```
        if ps_smoothed[upper99pc_index]> 0.01*np.max(ps_smoothed):
            if (ps_smoothed[upper99pc_index]>ps_smoothed[upper99pc_index-
1])&(ps_smoothed[upper99pc_index]>ps_smoothed[upper99pc_index+1]):
                upper99pc_index_list.append(upper99pc_index)
    max_index_ascend = np.argsort(ps_smoothed[upper99pc_index_list])
    if len(upper99pc_index_list)>0:
        TP_punktur =[0,0]
        TP_punktur =[freq_smoothed[upper99pc_index_list][max_index_ascend[-
1]],ps_smoothed[upper99pc_index_list][max_index_ascend[-1]]]
    else:
        TP_punktur = float('nan')
    return(TP_punktur)

def Tp_gildi(freq_smoothed, ps_smoothed):
    gradients=np.diff(ps_smoothed)
    sorted_indices = np.argsort(np.abs(gradients))
    upper99pc_index_list=[] #skoða bara punkta sem eru í topp 99% af stærð gilda
    for upper99pc_index in sorted_indices:
        if ps_smoothed[upper99pc_index]> 0.01*np.max(ps_smoothed):
            if (ps_smoothed[upper99pc_index]>ps_smoothed[upper99pc_index-
1])&(ps_smoothed[upper99pc_index]>ps_smoothed[upper99pc_index+1]):
                upper99pc_index_list.append(upper99pc_index)
    max_index_ascend = np.argsort(ps_smoothed[upper99pc_index_list])
    if len(upper99pc_index_list)>0:
        TP_gildi = 1/freq_smoothed[upper99pc_index_list][max_index_ascend[-1]]
    else:
        TP_gildi = float('nan')
    return(TP_gildi)

def Tz_rofgreining_olduhluti(rofhluti_dict):
    f2S = (rofhluti_dict['f_array_smoothed']**2)*rofhluti_dict['ps_smoothed']
    olduhluti_m2= np.trapz(f2S, rofhluti_dict['f_array_smoothed'])
    olduhluti_m0= np.trapz(rofhluti_dict['ps_smoothed'],
rofhluti_dict['f_array_smoothed'])
    Tz = np.sqrt(olduhluti_m0/olduhluti_m2)
    return(Tz)

def nidurstodumynd(hofn, timi, hafnir, tide_data_dict,results_dict, sog_timarod,
alda_timarod, sog_tidnigreining, alda_tidnigreining, undiralda_tidnigreining,
vindalda_tidnigreining, imagepath, z):
    plt.close()
    fig, axs = plt.subplots(3, 3)
    fig.set_size_inches(11.69,8.27)
    fig.subplots_adjust(left=0.08, right=0.98, wspace=0.3, hspace = 0.3 )
    fig.suptitle(u'Niðurstöður {} {}.{}.{} kl {}:00'.format(hofn, str(timi)[4:6], str(timi)[6:8],
str(timi)[0:4], str(timi)[8:10]), fontsize='large')

    ax = axs[0, 0]
    ax.plot(tide_data_dict['time'],tide_data_dict['height'], color='darkblue',
label=u'Ölduhæð - mæld gögn')
    ax.plot(tide_data_dict['time'],tide_data_dict['correction_function'], color='red',
label=u'Aðhvarfslína')
    ax.set_xlabel(u'Tími [sek]')
```

```
    ax.set_ylabel(r'Sjávarhæð [m]')
    ax.legend(loc='best', fontsize='small')
    ax = axs[0, 1]
    xmin = 0
    xmax=len(tide_data_dict['time'])
    ax.plot(tide_data_dict['time'][xmin:xmax],tide_data_dict['eta_Tukey'][xmin:xmax],
color='darkblue', label=u'Miðjuð ölduhæð,
Hs='+str(np.round(results_dict['Hs_timagogn'][0], 2))+'m.')
    xlim=ax.axes.get_xlim()
    ax.set_xlabel(u'Tími [sek]')
    ax.set_ylabel(r'Miðjuð sjávarhæð [m] (Eta)')
    ax.legend(loc='best', fontsize='small')

    ax = axs[0, 2]
    ax.plot(sog_timarod['time'], sog_timarod['eta'], color='darkblue', label='Soghluti
timagreining, Hs='+str(np.round(results_dict['Hs_sog_timagreining'][0], 2))+'m.
Tz='+str(np.round(results_dict['Tz_sog_timagreining'][0],2))+'s. \n
Hmax='+str(np.round(results_dict['Hmax_sog_timagreining'][0],2))+'m. Eta
min='+str(np.round(results_dict['Eta_min_sog_timagreining'][0],2))+'m, Eta
max='+str(np.round(results_dict['Eta_max_sog_timagreining'][0],2))+'m')
    ax.set_xlabel(u'Tími [sek]')
    ax.set_ylabel(r'Miðjuð sjávarhæð [m]')
    ax.legend(loc='best', fontsize='small')

    ax = axs[1, 0]
    ax.plot(alda_timarod['time'], alda_timarod['eta'], color='darkblue', label=u'Ölduhluti
tímagreining. Hs='+str(alda_timarod['Hs'])+'m. Tz='+str(alda_timarod['Tz'])+'s')
    ax.set_xlabel(u'Tími [sek]')
    ax.set_ylabel(r'Miðjuð sjávarhæð [m]')
    ax.legend(loc='best', fontsize='small')

    minfreq = 1/np.max(np.array(hafnir[['vindalda_byrjun_sek']]))
    maxfreq = 1/np.min(np.array(hafnir[['sog_endar_sek']]))
    index_min = np.max(np.where(tide_data_dict['f_array']<maxfreq))
    index_max = np.min(np.where(tide_data_dict['f_array']>minfreq))

    ax = axs[1, 1]
    ax.plot(tide_data_dict['f_array'][index_min:index_max],
tide_data_dict['PSD_leidrett_Kp'][index_min:index_max], color='lightgray',
label=u'Óþjálgað róf')
    ax.plot(alda_tidnigreining['f_array_smoothed'], alda_tidnigreining['ps_smoothed'],
lw=2, label='Ölduhluti, Hs='+str(np.round(alda_tidnigreining['Hs'],1))+' m')
    ax.plot(sog_tidnigreining['f_array_smoothed'], sog_tidnigreining['ps_smoothed'],
lw=2, label='Soghluti, Hs='+str(np.round(results_dict['Hs_sog_tidnigreining'][0],2))+'
m')
    ax.set_xlabel(u'Tíðni [1/s]')
    ax.set_ylabel(r'Power Spectral Density [m2/Hz]')
    ax.legend(loc='upper right', fontsize='small')

    ax = axs[1, 2]
    mean_sea_height=np.round(np.mean(tide_data_dict['height']), 1)

ax.plot(tide_data_dict['f_array'][index_min:index_max],tide_data_dict['inv_Kp'][index_
```

```
min:index_max], color='darkblue', label=u'1/Kp. Dýpi nema (z):{}m.
\nMeðalsjávarhæð:{}m. \nHeildardýpi öldu:{}m.'.format(str(z), str(mean_sea_height),
str(mean_sea_height-z)) )
    ax.set_xlabel(u'Tíðni [1/s]')
    ax.legend(loc='best', fontsize='small')

    ax = axs[2, 0]
    ax.plot(sog_tidnigreining['f_array_smoothed'], sog_tidnigreining['ps_smoothed'],
label='Soghluti, Hs='+str(np.round(results_dict['Hs_sog_tidnigreining'][0],2))+'m')
    if not math.isnan(results_dict['Tp_gildi_sog_tidnigreining'][0]):
        ax.scatter(sog_tidnigreining['Tp_punktur'][0], sog_tidnigreining['Tp_punktur'][1],
marker='*', s=300, edgecolor = 'black',color='red', label='Tp='+
str(np.round(results_dict['Tp_gildi_sog_tidnigreining'][0],1))+'s')
    ax.set_xlabel(u'Tíðni [1/s]')
    ax.set_ylabel(r'Power Spectral Density [m2/Hz]')
    ax.legend(loc='upper right', fontsize='small')

    ax = axs[2, 2].remove()  # don't display empty ax

    ax = axs[2, 1]
    ax.plot(alda_tidnigreining['f_array_smoothed'], alda_tidnigreining['ps_smoothed'],
label=u'Ölduhluti')
    if not math.isnan(results_dict['Tp_gildi_undiralda_tidnigreining'][0]):
        ax.scatter(undiralda_tidnigreining['Tp_punktur'][0],
undiralda_tidnigreining['Tp_punktur'][1], marker='*', s=300, edgecolor =
'black',color='red', label='Tp Undiralda')
    if not math.isnan(results_dict['Tp_gildi_vindalda_tidnigreining'][0]):
        ax.scatter(vindalda_tidnigreining['Tp_punktur'][0],
vindalda_tidnigreining['Tp_punktur'][1], marker='*', s=300, edgecolor =
'black',color='yellow', label='Tp Vindalda')
    ax.set_xlabel(u'Tíðni [1/s]')
    ax.set_ylabel(r'Power Spectral Density [m2/Hz]')
    ax.legend(loc='upper right', fontsize='small')
    mytxt= 'Hs_ölduhluti='+str(np.round(alda_tidnigreining['Hs'], 2))+'m. Tz_ölduhluti =
'+str(np.round(alda_tidnigreining['Tz'],2))+'s.
\nHs_undiralda='+str(np.round(results_dict['Hs_undiralda_tidnigreining'][0],2))+'m.
Tp_undiralda='+str(np.round(results_dict['Tp_gildi_undiralda_tidnigreining'][0],1))+ 's.
\nHs_vindalda='+str(np.round(results_dict['Hs_vindalda_tidnigreining'][0], 2))+'m.
Tp_vindalda='+str(np.round(results_dict['Tp_gildi_vindalda_tidnigreining'][0],1))+ 's.'

ax.text(1.1*max(alda_tidnigreining['f_array_smoothed']),0.77*max(alda_tidnigreining['
ps_smoothed']), mytxt)

    fig.savefig(imagepath+u'Niðurstöður_{}_{}.jpg'.format(hofn, str(timi)), dpi=300,
facecolor='w', edgecolor='w',
                orientation='portrait', papertype=None, format=None,
                transparent=False, bbox_inches='tight', pad_inches=0.1)
    plt.clf()

def Hmax_eta_min_max_mynd(hofn, timi, sog_timarod, imagepath):
    # Aðferð: Finnur min og max í hverju bili milli zero down crossings.
    fig, ax = plt.subplots()
    fig.set_size_inches(11.69,8.27)
```

```
    xmin = 0
    xmax=3600
    timeaxis = np.zeros(xmax)
    ax.set_title(u'Soghluti (tímaröð) '+hofn+'
'+str(timi)[4:6]+'.'+str(timi)[6:8]+'.'+str(timi)[0:4]+' Kl '+str(timi)[8:10]+':00')
    ax.plot(sog_timarod['time'],sog_timarod['eta'], color='black', label=u'Soghluti -
tímagreining')
    zero_down_counter = 0
    current_zdc_point_index = 0
    previous_zdc_point_index = 0
    closest_min_x = []
    closest_min_y = []
    closest_max_x = []
    closest_max_y = []
    for i in np.arange(len(sog_timarod['eta'])-1):
        if (sog_timarod['eta'][i] > 0) and (sog_timarod['eta'][i+1] < 0):
            zero_down_counter=zero_down_counter+1
            current_zdc_point_index = i
            plt.scatter(sog_timarod['time'][i],0, c='red')
            localmax =
np.max(sog_timarod['eta'][previous_zdc_point_index:current_zdc_point_index+1])
            closest_max_y.append(localmax)
            local_maxpoint =
np.where(sog_timarod['eta']==np.max(sog_timarod['eta'][previous_zdc_point_index:c
urrent_zdc_point_index+1]))
            local_maxpoint= np.asscalar(local_maxpoint[0])
            closest_max_x.append(local_maxpoint)
            plt.axvline(sog_timarod['time'][local_maxpoint], c='red', lw=0.7, ls='--')
            if previous_zdc_point_index > 0:
                localmin =
np.min(sog_timarod['eta'][previous_zdc_point_index:current_zdc_point_index+1])
                closest_min_y.append(localmin)
                local_minpoint =
np.where(sog_timarod['eta']==np.min(sog_timarod['eta'][previous_zdc_point_index:cu
rrent_zdc_point_index+1]))
                local_minpoint= np.asscalar(local_minpoint[0])
                closest_min_x.append(local_minpoint)
                plt.axvline(sog_timarod['time'][local_minpoint], c='blue', lw=0.7, ls='--')
            previous_zdc_point_index=current_zdc_point_index+1

    lastmin =
np.min(sog_timarod['eta'][current_zdc_point_index+1:len(sog_timarod['eta'])])
    closest_min_y.append(lastmin)
    last_minpoint =
np.where(sog_timarod['eta']==np.min(sog_timarod['eta'][current_zdc_point_index+1:l
en(sog_timarod['eta'])]))
    last_minpoint= np.asscalar(last_minpoint[0])
    closest_min_x.append(last_minpoint)
    plt.axvline(sog_timarod['time'][last_minpoint], c='blue', lw=0.7, ls='--')
    Hmax_array=np.array(closest_max_y)-np.array(closest_min_y)
    Hmax, Eta_min, Eta_max= np.max(Hmax_array), np.min(np.array(closest_min_y)),
np.max(np.array(closest_max_y))
    #print('Hmax, Eta_min, Eta_max=',Hmax, Eta_min, Eta_max )
```

```
    plt.axhline(0 , c='darkblue')
    xlim=ax.axes.get_xlim()
    plt.xlabel(u'Tími [sek]')
    plt.ylabel(r'Miðjuð sjávarhæð [m]')
    ax.legend(loc='lower left', fontsize='small')
    fig.suptitle(u'Hmax= {}, Eta min= {}, Eta max= {}'.format(str(np.round(Hmax, 2)),
str(np.round(Eta_min, 2)), str(np.round(Eta_max,2)), fontsize='large'))
    fig.savefig(imagepath+'{}_sog_timarod_local_max_og_min_{}_kl{}.jpg'.format(hofn,
str(timi)[0:8],  str(timi)[8:10]), dpi=300, facecolor='w', edgecolor='w',
                orientation='portrait', papertype=None, format=None,
                transparent=False, bbox_inches='tight', pad_inches=0.1)
    plt.clf()
```

6.2 Úrvinnslulykkja

```
import os
import os.path
import datetime
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# Staðsetning gagna:
os.chdir('C:/Users/vg/oldumaelingar_Pnema')
cwd = os.getcwd()
import Olduhaed_hafnir_functions_mogt_v2 as fns

# Niðurstöðumynd eða skrá ?
##########################################################################
mynd = 'TRUE'
Hmax_eta_mynd = 'TRUE'
imagepath = cwd+'\\images\\'
##########################################################################
# INPUT/OUTPUT: Vísa á hvert myndir eiga að fara og hvar gögn er að finna
hafnir_inntaksskra= cwd+'\\Hafnir_inntaksskra.csv'
hafnir = pd.read_csv(hafnir_inntaksskra, header=0)
hafnir.index=hafnir['Hofn']

results = {}
for hofn in hafnir.index:
    now = datetime.datetime. now()
    timestring= now.strftime("%Y%m%d%H")
    datapath = cwd+'\\data\\{}\\'.format(hofn)
    datafile = '{}_Sj1_Alda_{}.dat'.format(hofn, timestring)
    tidedata=0
    tidedata = fns.read_tidedatafile(datapath, datafile)
    if tidedata=='nan':
        print(hofn+' '+timestring+u': skráin finnst ekki eða eitthvað er að gögnum')
        results['Dags']=[timestring]
        results['Eta_max_sog_timagreining']=-6999
        results['Eta_min_sog_timagreining']=-6999
```

```
        results['Hmax_sog_timagreining']=-6999
        results['Hofn']=[hofn]
        results['Hs_sog_tidnigreining']=-6999
        results['Hs_sog_timagreining']=-6999
        results['Hs_timagogn']=-6999
        results['Hs_undiralda_tidnigreining']=-6999
        results['Hs_vindalda_tidnigreining']=-6999
        results['Tp_gildi_sog_tidnigreining']=-6999
        results['Tp_gildi_undiralda_tidnigreining']=-6999
        results['Tp_gildi_vindalda_tidnigreining']=-6999
        results['Tz_sog_tidnigreining']=-6999
        results['Tz_sog_timagreining']=-6999
        results['Tz_timagogn']=-6999

##########################################################################
##################
    if not tidedata=='nan':
        ### Miðja gögnin
        tidedata= fns.normalise_tidedata(tidedata)
        ### Nota tapering glugga af gerðinni: Tukey
        r_percent=0.05
        tidedata['eta_Tukey']= fns.tukey_window_eta(tidedata, r_percent)
        # Safna niðurstöðum í pandas dataframe sem heitir results
        results['Hofn']=[hofn]
        results['Dags']=[np.int(timestring)]

##########################################################################
##################
        ########## Greining í tíma / Time domain analysis
####################

##########################################################################
##################

        # Hs : Kennialda / Significant wave height: Hs = 4*sigma
        results['Hs_timagogn']=[ fns.Hs_timagreining(tidedata['eta'])]
        # Mean period - time based
        results['Tz_timagogn']=[fns.Tz_timagreining(tidedata)]

        # Soghluti merkis fenginn með þjálgun merkis með 25 sek glugga  ###
        sog_timagreining={}
        sog_timagreining['time'], sog_timagreining['eta']=
fns.soghluti_timagreining(tidedata)
        # Ölduhluti merkis (heildarmerki-soghluti merkis)###################
        alda_timagreining={}
        alda_timagreining['time'], alda_timagreining['eta'] =
fns.olduhluti_timagreining(tidedata, sog_timagreining)
        # Hs : Kennialda
        alda_timagreining['Hs']=4*np.std(alda_timagreining['eta'])
        # Mean period - time based
        alda_timagreining['Tz']=fns.Tz_timagreining(alda_timagreining)
        # Hs : Kennialda sogmerkis
        results['Hs_sog_timagreining']= [fns.Hs_timagreining(sog_timagreining['eta'])]
```

```
        # Mean period - time based
        tz=25*len(sog_timagreining['time'])/fns.zero_down_crossings(sog_timagreining)
        results['Tz_sog_timagreining']= [tz]

        hmax, etamin, etamax =
fns.Sog_timagreining_Hmax_eta_min_max(sog_timagreining)
        results['Hmax_sog_timagreining']= [hmax]
        results['Eta_min_sog_timagreining']= [etamin]
        results['Eta_max_sog_timagreining']= [etamax]


##########################################################
##############################
        ########## Greining í tíðnirúminu / Frequency domain analysis

##########################################################
##############################
        tidedata['f_array'], tidedata['PSD'] = fns.PSD_rofgreining(r_percent, tidedata)

        # Leiðrétti spektrið fyrir dempun vegna sjávarstöðu: Pressure response factor: Kp
        tidedata['inv_Kp']= fns.inv_Kp(hafnir.loc[hofn, 'raundypi_m'], hafnir.loc[hofn,
'dypi_nema_m'], tidedata)
        tidedata['PSD_leidrett_Kp'] = tidedata['PSD']*tidedata['inv_Kp']

        # SOG
        sog_tidnigreining = fns.rofhluti(tidedata, hafnir.loc[hofn, 'thjalgun_sog_gluggi'],
hafnir.loc[hofn, 'sog_byrjun_sek'], hafnir.loc[hofn, 'sog_endar_sek'])
        results['Hs_sog_tidnigreining'] = [fns.Hs_rofgreining(sog_tidnigreining)]
        sog_tidnigreining['Tp_punktur'] =
fns.Tp_punktur(sog_tidnigreining['f_array_smoothed'],
sog_tidnigreining['ps_smoothed'])
        results['Tp_gildi_sog_tidnigreining'] =
[fns.Tp_gildi(sog_tidnigreining['f_array_smoothed'], sog_tidnigreining['ps_smoothed'])]
        results['Tz_sog_tidnigreining'] = [fns.Tz_rofgreining_olduhluti(sog_tidnigreining)]

        # ÖLDUHLUTI
        alda_tidnigreining = fns.rofhluti(tidedata, hafnir.loc[hofn, 'thjalgun_alda_gluggi'],
hafnir.loc[hofn, 'vindalda_byrjun_sek'], hafnir.loc[hofn, 'sog_byrjun_sek'])
        alda_tidnigreining['Hs'] = fns.Hs_rofgreining(alda_tidnigreining)
        alda_tidnigreining['Tz'] = fns.Tz_rofgreining_olduhluti(alda_tidnigreining)

        # UNDIRALDA
        undiralda_tidnigreining = fns.rofhluti(tidedata, hafnir.loc[hofn,
'thjalgun_alda_gluggi'], hafnir.loc[hofn, 'undiralda_byrjun_sek'], hafnir.loc[hofn,
'sog_byrjun_sek'])
        results['Hs_undiralda_tidnigreining'] =
[fns.Hs_rofgreining(undiralda_tidnigreining)]
        undiralda_tidnigreining['Tp_punktur'] =
fns.Tp_punktur(undiralda_tidnigreining['f_array_smoothed'],
undiralda_tidnigreining['ps_smoothed'])
        results['Tp_gildi_undiralda_tidnigreining'] =
[fns.Tp_gildi(undiralda_tidnigreining['f_array_smoothed'],
undiralda_tidnigreining['ps_smoothed'])]
```

```
        # VINDALDA
        vindalda_tidnigreining = fns.rofhluti(tidedata, hafnir.loc[hofn,
'thjalgun_alda_gluggi'], hafnir.loc[hofn, 'vindalda_byrjun_sek'], hafnir.loc[hofn,
'undiralda_byrjun_sek'])
        results['Hs_vindalda_tidnigreining'] = [fns.Hs_rofgreining(vindalda_tidnigreining)]
        vindalda_tidnigreining['Tp_punktur'] =
fns.Tp_punktur(vindalda_tidnigreining['f_array_smoothed'],
vindalda_tidnigreining['ps_smoothed'])
        results['Tp_gildi_vindalda_tidnigreining'] =
[fns.Tp_gildi(vindalda_tidnigreining['f_array_smoothed'],
vindalda_tidnigreining['ps_smoothed'])]

        if mynd == 'TRUE':
            nidurstmynd =  fns.nidurstodumynd(hofn, timestring, hafnir, tidedata, results,
sog_timagreining, alda_timagreining, sog_tidnigreining, alda_tidnigreining,
undiralda_tidnigreining, vindalda_tidnigreining, imagepath, hafnir.loc[hofn,
'dypi_nema_m'])
            nidurstmynd = 0
            plt.close('all')
        if Hmax_eta_mynd == 'TRUE':
            nidurstmynd= fns.Hmax_eta_min_max_mynd(hofn, timestring,
sog_timagreining, imagepath)
            nidurstmynd = 0
            plt.close('all')
```