

Gebruikershandleiding hatyan getijanalyse in Python



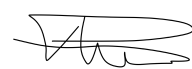


Gebruikershandleiding hatyan getijanalyse in Python

Opdrachtgever	Rijkswaterstaat Water, Verkeer en Leefomgeving
Contactpersoon	Martin Scholten, Jan Rolf Hendriks
Trefwoorden	Getij, getijanalyse, getijpredictie, waterstanden, extremen, hoogwaters, laagwaters, componentensplitsing, hatyan

Documentgegevens	
Versie	1.0
Datum	17-12-2020
Projectnummer	11205257-013
Document ID	11205257-013-DSC-0006
Pagina's	51
Classificatie	
Samenvatting	
Status	definitief

Auteur(s)	
	Jelmer Veenstra David Kerkhoven

Doc. Versie	Auteur	Controle	Akkoord	Publicatie
1.0	Jelmer Veenstra 	Jaco Stout 	Hans van Putten 	

Inhoud

1	De applicatie hatyan	4
1.1	Inleiding	4
1.2	Doelstelling en reikwijdte	4
1.3	Opzet van de applicatie	5
1.4	Beoogde gebruikers	5
1.5	Werkwijze	5
1.6	Programmatuur en systeemomgeving	6
1.7	Leeswijzer	6
2	Theoretische achtergrond	7
2.1	Werking getij – het evenwichtsgetij	7
2.1.1	Maansgetij	7
2.1.2	Zonsgetij	8
2.1.3	Declinatie – dagelijkse ongelijkheid	9
2.1.4	Spring- en doodtijcyclus – maandelijks ongelijkheid	9
2.1.5	Jaarlijkse ongelijkheid	11
2.2	Het astronomisch getij langs de Nederlandse Kust	12
2.2.1	Hoog- en Laagwaters	12
2.2.2	Agger	12
2.2.3	Propagatie langs de Nederlandse kust	13
2.2.4	Amfidromische punten	14
2.3	Harmonische analyse	15
2.4	Componentensplitsing	15
2.5	X-factor	16
3	Belangrijkste hatyan invoerbestanden	18
3.1	Bestand met tijdserie (equidistante dia-files)	18
3.2	Bestand met extremen (niet-equidistante dia-files)	19
3.3	Bestand met analyseresultaat: componentenbestand	19
3.4	Bestand met slotgemiddelden	20
4	Werkproces hatyan	21
4.1	Inleiding	21
4.2	Analyse	22
4.2.1	Inleiding	22
4.2.2	Invoer	22
4.2.3	Analyse meerdere periodes	22
4.2.3.1	4-jarige analyse	22
4.2.3.2	19-jarige analyse	23
4.2.3.3	Componentensplitsing	23
4.2.4	Samenvoegen van resultaten	23

4.2.5	Uitvoer	23
4.3	Predictie	24
4.3.1	Inleiding	24
4.3.2	Invoer	24
4.3.3	Uitvoer	24
4.4	Bepaling van extremen (hoog- en laagwaters)	24
4.4.1	Invoer	24
4.4.2	Uitvoer	24
5	Quick start guide	25
5.1	Klaarzetten van de invoerdata	25
5.2	Hatyan configuratiefile	25
5.2.1	Voorbeeld en uitleg configuratiefile	25
5.2.2	Meegeleverde configuratiefiles	26
5.3	Draaien van de applicatie	27
5.4	Output	27
5.5	Interpretatie van resultaten	28
5.5.1	Visuele inspectie van analyseresultaten	28
5.5.2	Visuele inspectie van predictieresultaten	29
5.5.3	Visuele inspectie van HWLW resultaten	32
6	Referenties	34
A	Hatyan componentensets	35
B	Hatyan componenten en kengetallen	36
C	Methode componentensplitsing	41
D	Ruimtelijke figuren	43
E	Builden en installatie hatyan RPM	46
E.1	Builden hatyan RPM	46
E.2	Installatie hatyan RPM	47
F	Automatisch gegenereerde documentatie vanuit docstrings	49

1 De applicatie hatyan

1.1 Inleiding

Het voorspellen van astronomisch getijden speelt een belangrijke rol in diverse werkprocessen en producten van Rijkswaterstaat en bij klanten van Rijkswaterstaat¹. Het gaat enerzijds om de getijtafels voor Nederland en diverse daaraan gerelateerde producten. Anderzijds speelt getijanalyse een belangrijke rol bij het maken van operationele waterstandsverwachtingen binnen de omgeving van het Water Management Centrum Nederland (WMCN); specifiek de onderdelen: Hydro Meteo Centra (HMC's) voor Kust en Benedenrivieren.

In de huidige operationele systemen van Rijkswaterstaat (de RWsOS-en) worden getijvoorspellingen gebruikt die afkomstig zijn uit DONAR (Data Opslag Natte Rijkswaterstaat). Deze getijvoorspellingen in DONAR worden al enkele decennia ieder jaar met de Fortran-HATYAN programmatuur berekend door Rijkswaterstaat. Jaarlijks, enkele maanden voor het begin van het nieuwe jaar worden de waterstandsreeksen van het astronomisch getij en hoog- en laagwaterextremen in DONAR geplaatst. Daarnaast levert Rijkswaterstaat de gegevens voor het getijtafelboekje ieder jaar aan SDU uitgevers, die het gele boekje voor de getijtafels van Nederland drukt (Figuur 1.1).

Naast dit interne gebruik bij Rijkswaterstaat worden de getijtafels en de diverse daaraan gerelateerde producten gebruikt in de operationele processen van waterbeheerders, beroepsscheepvaart, scheepvaartbegeleiding en bij recreatie (bijvoorbeeld surf- en duiksport, pleziervaart, etc.).



Figuur 1.1 Het gele boekje met de Getijtafels voor Nederland 2019.

1.2 Doelstelling en reikwijdte

Het doel achter de ontwikkeling van de hatyan Python-applicatie was om de oude Fortran-HATYAN code naar Python om te zetten en de hierbij opgedane kennis over de methodiek voor getijanalyse en getijpredictie reproduceerbaar vast te leggen.

De nieuwe hatyan applicatie zal gebruikt worden voor het jaarlijks genereren van de astronomische tijdreeksen voor de meetlocaties in de Nederlandse Getijdewateren en het genereren van de astronomische extremen van waterstanden, zoals voorheen ook Fortran-HATYAN dit deed. De applicatie kan eveneens ingezet worden om getijanalyses en getijpredicties te maken op basis van meetreeksen van drukdozen of waterstandssensoren of op basis van tijdreeksen van modelresultaten van hydrodynamische modellen. Hierbij kan, indien gewenst, gebruik gemaakt worden van componentensplitsing, een methodiek die het mogelijk maakt om astronomische tijdreeksen te genereren op basis van kortere meetreeksen dan normaal,

¹ Illustratief in dit kader is de RWS publicatie 'Het getij en wij', uitgegeven door SDU:
<http://publicaties.minienm.nl/documenten/het-getij-en-wij-honderd-jaar-getijtafels-voor-nederland>

bijvoorbeeld bij nieuwe meetlocaties of tijdelijke meetpunten. Bij deze methodiek worden naast de korte meetreeksen van de betreffende locatie, ook de amplitudeverhoudingen en faseverschillen van enkele reeds bekende getijcomponenten van naburige stations gebruikt. De applicatie beschikt tevens over de mogelijkheid om analyse- en predictieresultaten te vergelijken met validatieresultaten van het oudere Fortran-HATYAN².

1.3 Opzet van de applicatie

De hatyan applicatie is, ten behoeve van de onderlinge vergelijkbaarheid, reproduceerbaarheid en consistentie met voorgaande getijpredicties en analyses, zoveel mogelijk gebaseerd op de methodiek uit de voormalige Fortran-HATYAN applicatie (RWS-RIKZ, 1999; Van der Heijden et al, jaartal onbekend; Schureman, 1941).

Deze methodiek is in voorbereiding op de bouw van hatyan in detail beschreven in het Technisch Ontwerp van hatyan (Irazoqui Apecechea, 2018). De wijze waarop voor verschillende locaties het astronomisch getij wordt afgeleid is vastgelegd in een Excelfile (data_overzicht_20190528.xlsx, opgeleverd bij de applicatie). Deze recepten zijn gebaseerd op de jarenlange ervaring en kennis afkomstig van een medewerker bij RWS-CIV die al enkele decennia verantwoordelijk is voor de productie van getijvoorspellingen. De recepten zijn in het kader van de ontwikkeling van deze applicatie in kaart is gebracht en zoveel mogelijk opgenomen in de bij de applicatie meegeleverde configuratiefiles.

1.4 Beoogde gebruikers

De gebruikersgroep van deze applicatie is vooralsnog een zeer kleine groep mensen bij RWS. De gebruikershandleiding is zodanig opgezet dat deze door betrokken medewerkers bij RWS of Deltares kan worden gebruikt, maar ook door een marktpartij. De beoogde gebruikers die voor bestaande locaties een getijanalyse of -predictie willen uitvoeren, dienen over basiskennis te beschikken van hydrodynamica, getijden, harmonische analyses en metingen.

Indien nieuwe locaties dienen te worden toegevoegd voor harmonische analyse of nieuw verkregen inzichten omtrent meetperiodes dienen te worden doorgevoerd bij bestaande locaties, dan dienen deze door experts op het gebied van hydrodynamica, getijden, harmonische analyses en metingen te worden doorgevoerd in de configuratiefile(s).

1.5 Werkwijze

Het doorlopen van een getijdepredictie gebeurt door het eerst analyseren van meetdata en het vervolgens maken van een predictie met de resultaten van deze analyse. Ook kan gebruik gemaakt worden van de resultaten van een eerder analyse. In Figuur 4.1 is een stroomschema van het werkproces van de hatyan applicatie weergegeven. Bij zowel de analyse als de predictie wordt indien gewenst gecorrigeerd voor knoopfactoren en x-factoren. Bij de analyse kan eventueel componentensplitsing worden toegepast. Bij de predictie kunnen ook de extremen (hoog- en laagwaters) worden bepaald. Bij zowel de analyse, de predictie als de extremen is het mogelijk te vergelijken met validatiedata uit het voormalige Fortran-HATYAN applicatie. De validatiedata voor 2019 is tijdens de ontwikkeling van hatyan applicatie gebruikt ter controle van de reproduceerbaarheid van de resultaten.

De hatyan applicatie werkt door het uitvoeren van de configuratiefile. Daarin worden alle locaties van benodigde data gegeven en worden de verschillende functies binnen hatyan aangeroepen (inlezen van componentenfiles, inlezen van tijdreeksen, analyse, predictie, bepaling extremen, plotten). Bij het aanroepen van de functies kan de gebruiker zelf beslissen welke instellingen gebruikt worden door argumenten voor bijvoorbeeld knoopfactoren en x-factoren mee te geven. De gebruiker bepaalt vooraf welke stappen doorlopen zullen worden door het script op te stellen, en er kan een standaard Python for-loop gebruikt worden om dit voor meerdere stations te doen.

² Er staan in de applicatie validatiereeksen uit Fortran-HATYAN voor onder andere 2020.

De uitvoer van de applicatie bestaat onder andere uit bestanden en figuren met analyseresultaten en predictieresultaten. Ook is er een diagnostische file beschikbaar waarin alle naar het scherm geprinte tekst terecht komt. Tot slot worden er indien gewenst figuren opgeslagen die tijdens het werkproces zijn gemaakt. Tijdens uitvoering van het werkproces is het voor de gebruiker mogelijk om de gemaakte figuren interactief te bekijken (inclusief functionaliteit als het verschuiven in de tijd, inzoomen, etc) en deze bewerkte figuren vervolgens handmatig op te slaan, het is ook mogelijk om een standaardfiguur aan te passen binnen het script. Details over het werkproces van de applicaties zijn gedocumenteerd in Hoofdstuk 3.

1.6 Programmatuur en systeemomgeving

De hatyan applicatie is een Python v3.6 applicatie die in principe op ieder systeem gebruikt kan worden. Ook wordt er een RPM opgeleverd zodat de applicatie inclusief een Python virtuele omgeving en benodigde libraries bij Rijkswaterstaat geïnstalleerd kan worden (RedHat 6.8 bij RWS en CentOS 6.9 bij Deltares, zie ook Bijlage E). Deze Linux versie maakt gebruik van visualisaties op basis van X-forwarding.

De applicatie wordt bij RWS geïnstalleerd op de DONAR-omgeving. De applicatie wordt bij Deltares beheerd en onderhouden, hierbij wordt gebruik gemaakt van Subversion versiebeheer. Voor het testen en ontwikkelen beschikt Deltares over een Virtuele Machine met CentOS-omgeving. Deze laatste is ook nodig om de RPM mee te maken. In versiebeheer bevinden zich zowel de RPM als de individuele Python scripts en de Virtuele Machine.

1.7 Leeswijzer

Deze handleiding is opgesteld voor iedereen die met de hatyan applicatie werkt of dat wil gaan doen. Het richt zich voornamelijk op gebruikers met een zekere voorkennis van hydrodynamica, getijden en harmonische analyse methodieken. Verder is een basiskennis van Python en Linux command line praktisch.

Hoofdstuk 2 bevat een compacte en vereenvoudigde theoretische achtergrond van het evenwichtsgetij en van het astronomisch getij met daarbij een uitleg over de werking van het getij en de daarbij behorende definities, het concept harmonische analyse, componentensplitsing en de X-factor.

Hoofdstuk 3 bevat een beschrijving van de belangrijkste hatyan invoerbestanden.

Hoofdstuk 4 bevat een nadere functionele en technische beschrijving van het hatyan werkproces, de onderliggende workflows en de verschillende invoerfiles.

Hoofdstuk 5 bevat een quick start guide met een tutorial en een testbank. In dit hoofdstuk worden de verschillende outputs van de applicatie beschreven.

Bijlage F bevat de automatisch gegenereerde documentatie voor hatyan. Hier worden alle functies en bijbehorende argumenten beschreven.

2 Theoretische achtergrond

In dit hoofdstuk worden eerst de basisprincipes van het getij uitgelegd, zoals het maansgetij, zonsgetij, dagelijkse ongelijkheid, spring- en doortijcyclus, hoog- en laagwaters, agger en amfidromische punten. Daarna worden de basisprincipes van de harmonische analyse (de uitsplitsing in getijcomponenten) zoals geïmplementeerd in de hatyan applicatie, nader toegelicht.

2.1 Werking getij – het evenwichtsgetij

Met getij wordt bedoeld het wisselend stijgen en dalen van het zeewater- oppervlak ten opzichte van het land ten gevolge van de aantrekkingskrachten die zon en maan op de aarde uitoefenen. De maan is daarbij het belangrijkste getij genererende hemellichaam: de getijopwekkende kracht van de maan bedraagt circa tweemaal die van de zon. De overige planeten in het zonnestelsel staan te ver van de aarde vandaan om een merkbare invloed te hebben op het getij.

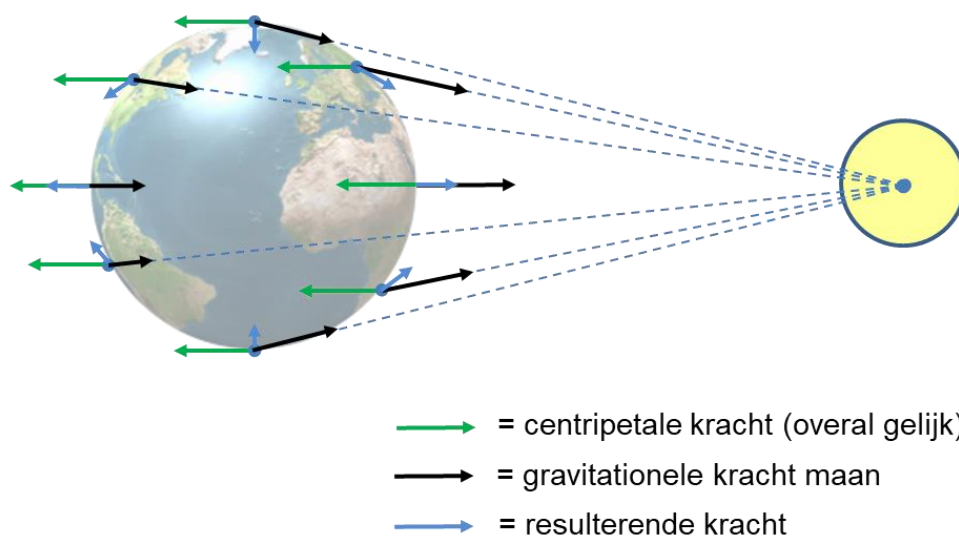
Naast deze astronomische invloeden wordt het getij tevens beïnvloed door niet-astronomische invloeden, zoals de ligging van kustlijnen en ijs, lokale veranderingen in waterdiepte, oceaانبodem topografie en meteorologische invloeden, welke bijdragen aan de spreiding van, het interval tussen hoog- en laag waters en aan de fasering/tijdstippen/vorm van het getij.

Voor begripsvorming omtrent de werking van het getij wordt meestal teruggevallen op de basisbeginselen van Newton uit de 17^e eeuw, waarbij - middels het “evenwichtsgetij”- uitgegaan wordt van een aardbol zonder continenten en een oceaan met uniforme diepte.

De term “getij” refereert specifiek naar het astronomische geïnduceerde deel van de verticale waterbeweging, zonder daarbij rekening te houden met de effecten van wind of golven. Met behulp van methodiek van harmonische analyse is het mogelijk om dit astronomische geïnduceerde deel van de verticale waterbeweging te filteren uit gemeten waterstandstijdreeksen en voor elke meetlocatie is een serie van getijcomponenten samen te stellen die het getij op die locatie beschrijft.

2.1.1 Maansgetij

Het maansgetij ontstaat door de onderlinge aantrekkingskracht van de aarde en de maan. De hemellichamen draaien om een gezamenlijk zwaartepunt, dit zorgt voor een constante centripetale kracht. Aangezien de aarde aanzienlijk zwaarder is dan de maan (de massa van de maan bedraagt ongeveer 1,2% van dat van de aarde), ligt het zwaartepunt van de draaiing van het koppel binnen de aardschil aan de kant van de maan. De maan draait als het ware om de aarde en maakt daarbij een elliptische baan. De aantrekkingskracht van de maan zorgt ervoor dat op de aarde (zonder continenten – uniforme waterlaag) het water naar de maan wordt getrokken. De aantrekkingskracht van de maan op de aarde is, aan de kant van de aarde waar de maan staat, het grootst. Aangezien de aarde een doorsnede heeft van ruim 6300 km, is de aantrekkingskracht van de maan aan de andere zijde van de aarde minder groot. De resulterende kracht van de uniforme centripetale kracht en de variërende aantrekkingskracht van de maan, zorgt voor twee getijbulten aan weerszijden van de aarde (zie Figuur 2.1). Doordat de aarde binnen de waterschil (getijbulten) om haar eigen as draait, ontstaan er hoogwater en laagwater.



Figuur 2.1 Getijopwekkende krachten op de aarde

De amplitude van het getij varieert in de loop van de maand doordat de baan van de maan rond de aarde geen cirkel is, maar een ellips waarbij de aarde niet in het centrum ligt. Daardoor varieert de afstand van de aarde tot de maan en is de getijopwekkende kracht maximaal 20 procent meer of minder dan gemiddeld. De tijd tussen twee opeenvolgende tijdstippen van maximale aantrekkingskracht (perigeum) of minimale aantrekkingskracht (apogeum) is 27,55 dagen (een anomalistische maand).

De aarde draait in ongeveer 24 uur om haar eigen as. Tijdens deze rotatie draait de maan ten opzichte van de aarde ook verder ($1/27.55$ deel van 24 uur, ongeveer 50 minuten). Vanaf een vast punt op de aarde duurt de rotatie van de maan om de aarde dus 24 uur en 50 minuten (een maansdag, of lunar day) (Pugh et al., 2014). De maan levert dus iets minder dan twee keer per dag een hoogwater en laagwater op de aarde op, resulterend uit de maanscomponenten (bijvoorbeeld M2). De maan levert door haar geringe afstand tot de aarde de grootste bijdrage aan het opwekken van het getij, maar dezelfde processen gelden voor het zonsgetij.

2.1.2 Zonsgetij

Het zonsgetij, dat ontstaat door de onderlinge aantrekkingskracht van de aarde en de zon wordt op vergelijkbare wijze als het maansgetij gevormd. De aarde draait in 365,26 dagen om de zon. De omlooprichting van de aarde om de zon is gelijk aan de richting waarin de aarde om haar eigen as draait. Ten opzichte van een ver punt in de ruimte draait de aarde in 23 uur en 56 minuten om haar as. Na deze omwenteling bevindt de aarde zich echter nog niet in dezelfde positie ten opzichte van de zon. De aarde draait nog gedurende 4 minuten door om in dezelfde positie ten opzichte van de zon te belanden. Na 24 uur is de aarde in de oorspronkelijke positie terug (een zonsdag, of solar day).

Het zonsgetij levert dus precies tweemaal per dag (24 uur) een hoog- en een laagwater voor de zonscomponenten (bijvoorbeeld S2). De getijopwekkende kracht van de zon bedraagt maar ongeveer de helft van de bijdrage van de maan, ook al heeft de zon een veel grotere massa dan de maan. Dit komt omdat de zon 389 keer zo ver weg van de aarde staat en de sterkte van de getijopwekkende kracht voor een groot gedeelte afhangt van de afstand tussen de hemellichamen. Het evenwichtsgetij bestaat uit de combinatie van de ellipsoïde van de maan en de zon.

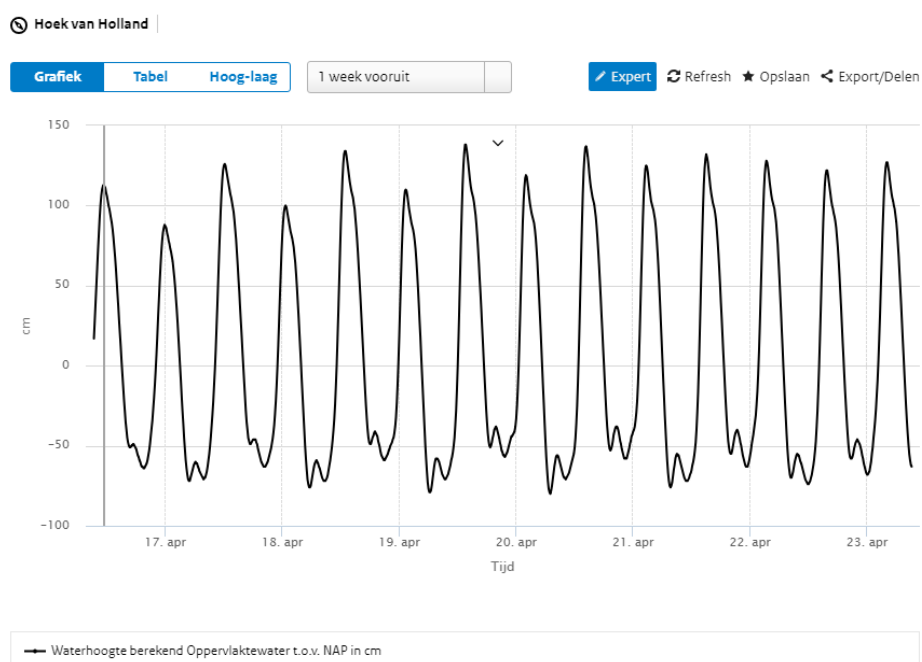
2.1.3

Declinatie – dagelijkse ongelijkheid

Doordat de draaiingsas van de aarde niet loodrecht staat op de vlakken waarin de aarde om de zon en de maan om de aarde bewegen, de zogenaamde declinatie, zijn twee opeenvolgende hoog- en laagwaters op dezelfde locatie niet aan elkaar gelijk. De twee getijbulten zijn door de declinatie ten opzichte van de evenaar verschoven; een van de bulten bevindt zich meer op het noordelijk halfrond en de tweede meer op het zuidelijk halfrond. Bij de dagelijkse draaiing van de aarde ontstaat in Nederland eerst een hogere getijslag bij passage van de noordelijker bult, en daarna een lagere getijslag. Deze dagelijkse ongelijkheid in hoog- en laagwaters is vooral merkbaar op gematigde geografische breedtegraad, waar Nederland ook op ligt. Doordat de twee hoog- en laagwaters per dag verschillen, zijn er in getijwaarnemingen ook enkeldagse componenten aanwezig.

De declinatie varieert in de loop van een maand met de positie van de maan ten opzichte van de aarde. De maan staat de helft van de 27,21 dagen van deze variatieperiode (Pugh et al., 2014) boven het vlak en de helft van de tijd eronder, en tweemaal in het vlak. Daardoor verplaatst de positie van beide getijbulten zich tussen het noordelijk halfrond en het zuidelijk halfrond. Een voorbeeld van de resulterende variërende dagelijkse ongelijkheid is weergegeven in Figuur 2.2.

Waterhoogte berekend Oppervlaktewater t.o.v. NAP



Figuur 2.2 Dagelijkse ongelijkheid bij Hoek van Holland (bron: waterinfo.rws.nl)

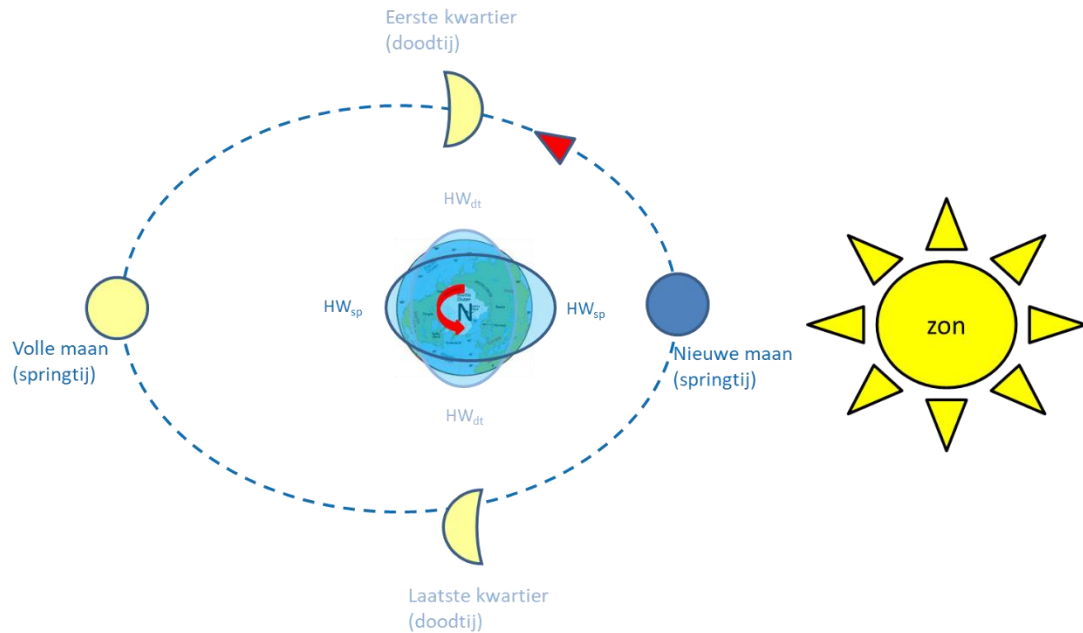
2.1.4

Spring- en doortijcyclus – maandelijks ongelijkheid

Tussen de zons- en maansgetijden treden interacties op. De grootste versterking, en daarmee het grootste verschil tussen hoog- en laagwater (springtij), treedt op als de zon, maan en aarde ongeveer op één lijn staan (ook wel syzygy genoemd, zie Figuur 2.3). Dit is het geval vlak na nieuwe maan en vlak na volle maan en er treedt dan springtij op.

Op het moment dat de zon en de maan onder een hoek van 90 graden ten opzichte van de aarde liggen, wordt van twee verschillende kanten aan het water aangetrokken met als gevolg dat het water veel minder stijgt dan gemiddeld (doortij). Dit is het geval vlak na het eerste en laatste (derde) kwartier.

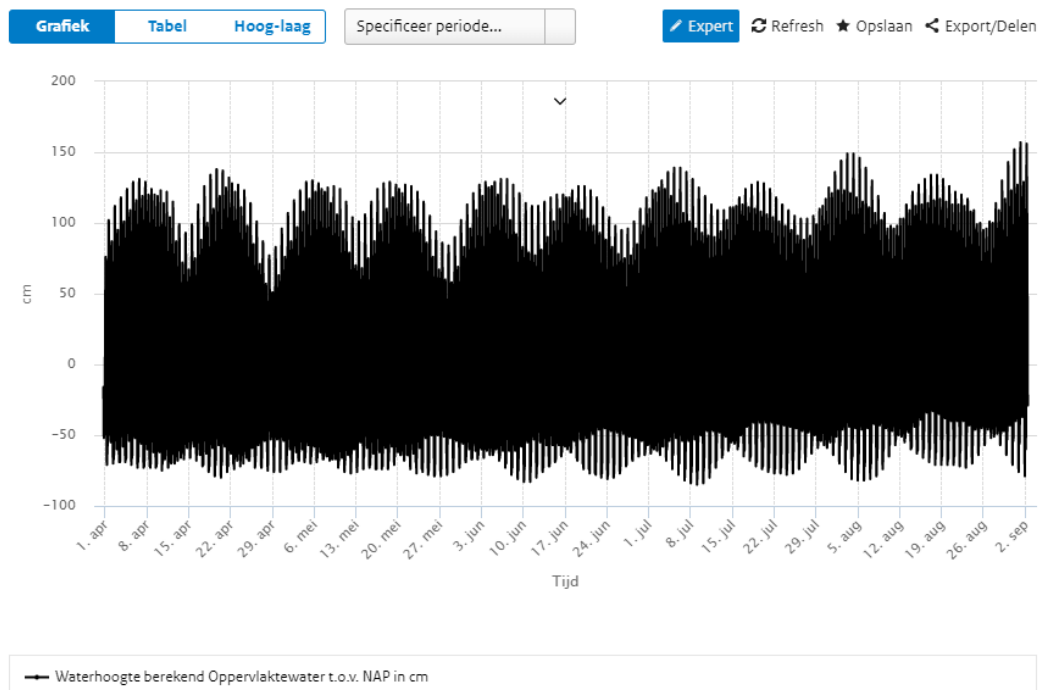
De periode van deze cyclus van nieuwe maan tot nieuw maan wordt een maanmaand genoemd (lunar month, 29,53 dagen) (Pugh, 2014). Springtij en doodtij treden dus beide ongeveer twee keer per kalendermaand op, zie ook Figuur 2.4.



Figuur 2.3 Ellipsvormige baan van de maan om de aarde en maanfases (uitlijning aarde, maan en zon) hebben invloed op de aantrekkingskracht

Waterhoogte berekend Oppervlaktewater t.o.v. NAP

Hoek van Holland

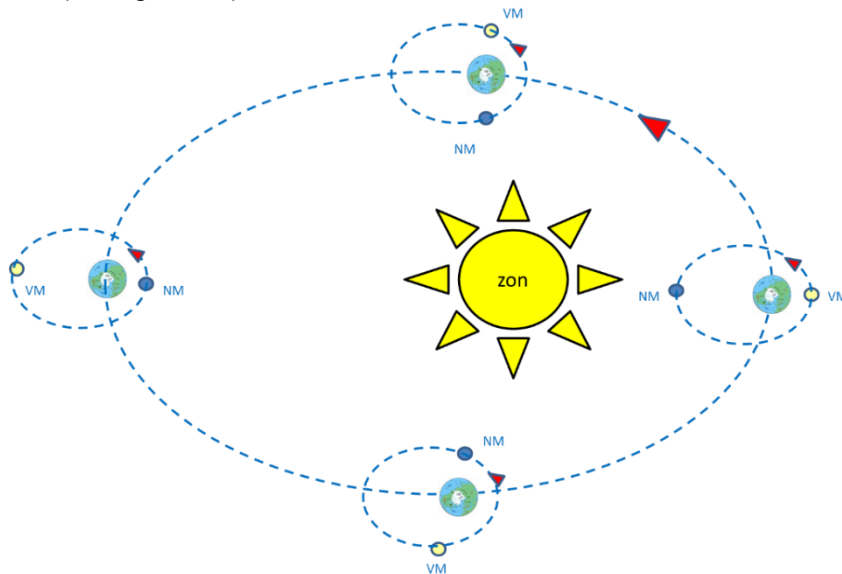


Figuur 2.4 Maandelijks ongelijkheid bij Hoek van Holland (bron: waterinfo.rws.nl)

2.1.5

Jaarlijkse ongelijkheid

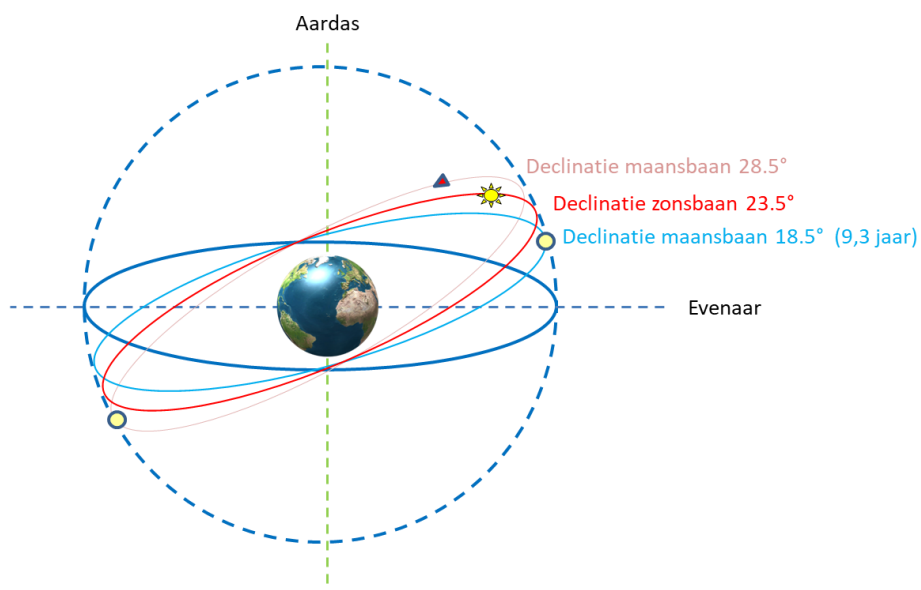
Naast dagelijkse en maandelijkse ongelijkheid van het getij, bestaat er ook een jaarlijkse ongelijkheid doordat er een afhankelijkheid is van de afstand van de aarde en de maan tot aan de zon (zie Figuur 2.5).



Figuur 2.5 Jaarlijkse ongelijkheid door de ellipsvormige baan van de aarde rondom de zon

Naast deze dagelijkse, maandelijkse en jaarlijkse ongelijkheden, bestaan er nog allerlei andere cycli die worden veroorzaakt door de afzonderlijke rotaties van aarde en maan om elkaar en als stelsel om de zon. De meeste relevante zijn:

- de 8,85 jarige cyclus die wordt veroorzaakt doordat de lange as van de ellips van de maansbaan in deze tijd rond de aarde roteert
- de 18,6 jarige cyclus. In deze tijd varieert de hoek die de maan maakt met het equatorvlak van de aarde tussen 18,5 graden en 28,5 graden.



Figuur 2.6 Figuur met uitleg van de 18,6 jarige cyclus ten gevolge van de variatie in de declinatiehoek van de maan.

2.2 Het astronomisch getij langs de Nederlandse Kust

Het evenwichtsgetij zoals omschreven in de vorige paragraaf, zou optreden als er geen continenten en ondiepe zeeën waren. De waterbeweging zoals die in de Noordzee kan worden waargenomen, is het resultaat van veranderingen die de opgewekte getijgolven op de Oceanen (met name de Atlantische Oceaan), in de Noordzee en in de Westerschelde hebben ondergaan. Eigenlijk ligt de oorsprong van de getijbeweging zoals wij dat langs de Nederlandse Kust waarnemen op het zuidelijke halfrond: De getijgolf die hier wordt opgewekt beweegt zich door de Atlantische Oceaan naar het noorden en onderweg wordt deze op verschillende manieren vervormd door de continenten en bodemvormen van de oceaan. Na twee etmalen arriveert de getijgolf in de Noordelijke IJzee. Ten gevolge van de draaiing van de aarde ondervindt deze getijgolf op het noordelijk halfrond een afwijking naar rechts (ten gevolge van de Corioliskracht). Verder is de Atlantische Oceaan zo groot, dat de periode van de getijgolf ongeveer gelijk is aan de eigen periode van de Oceaan. Hierdoor ontstaat er een staande golf in de breedterichting van de Atlantische Oceaan, die er voor zorgt dat de getijhoogten aan de rand van de Oceaan hoger zijn dan je op grond van de aantrekkingskracht zou verwachten. Dit is het begin van de getijgolf die vervolgens via Het Kanaal (bij Calais) en via het Noorden van Schotland en Engeland de Noordzee in loopt en langs de Nederlandse Kust weer naar het noorden gaat. Dit zorgt voor het getij zoals dat wordt waargenomen langs de Nederlandse Kust.

Naast deze astronomische invloeden wordt het getij tevens beïnvloedt door niet-astronomische invloeden, zoals luchtdruk, de ligging van kustlijnen en ijs, lokale veranderingen in waterdiepte, oceaanbodem topografie en meteorologische invloeden, welke bijdragen aan de spreiding van, het interval tussen hoog- en laag waters en aan de fasering/tijdstippen/vorm van het getij.

2.2.1 Hoog- en Laagwaters

De getijwaterstand volgt uit de som van alle getijdecomponenten, al dan niet beïnvloed door de bodem onderweg. Op het moment dat de som van alle getijcomponenten het grootst is ontstaat een hoogwater en op het moment dat de som van alle getijcomponenten het kleinst is een laagwater.

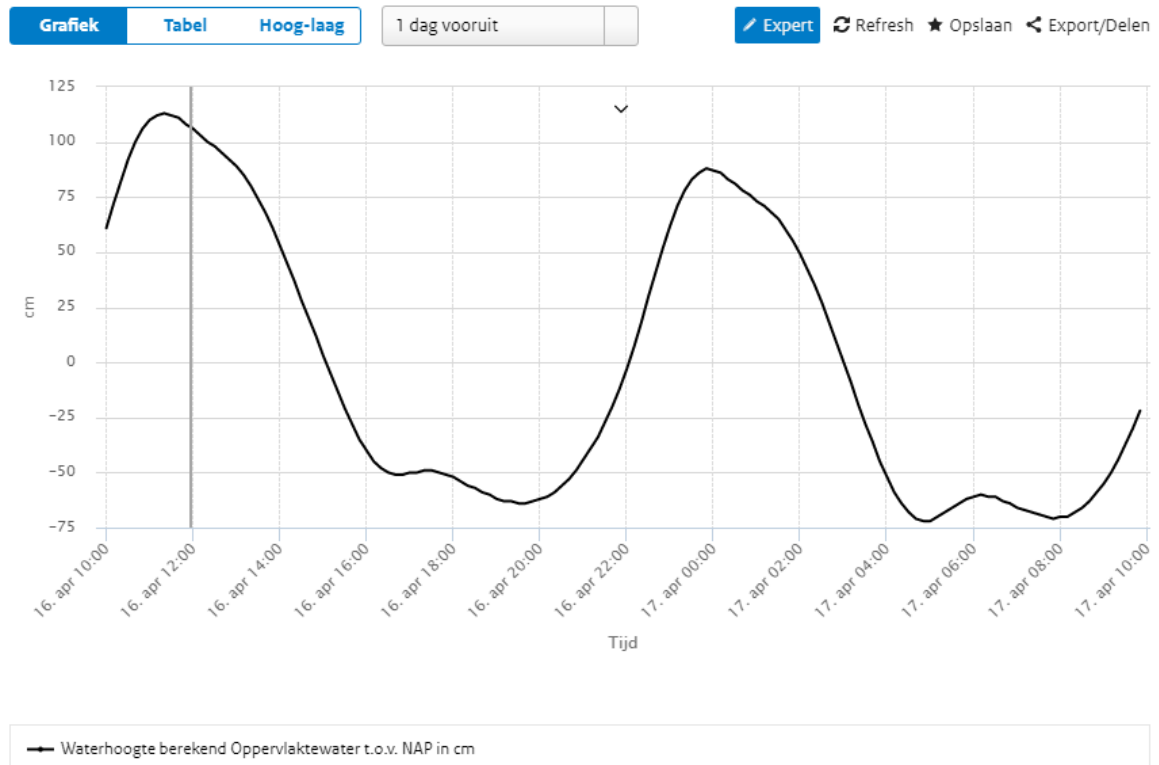
2.2.2 Agger

Bijzondere aandacht verdient de periode van laagwater in de gemiddelde getijkromme van bijvoorbeeld Hoek van Holland. Nadat het eerste laagwater is opgetreden, stijgt het waterniveau een klein beetje, waarna het vervolgens opnieuw daalt. Pas daarna stijgt het water snel en wordt het weer vloed. Eigenlijk treedt er dus een dubbel laagwater op, wat ook wel agger wordt genoemd. Dit verschijnsel treedt met enige regelmaat op, onder andere langs de Zuid-Hollandse kust en landinwaarts op de rivieren. De sterkte ervan verloopt van plaats tot plaats en met de intensiteit van het getij en de afvoer.

Een soortgelijk verschijnsel treedt ook op bij Den Helder, maar hier is sprake van een dubbel hoogwater, de vloedperiode wordt als het ware uitgerekt. De sterkte ervan verloopt, net als bij de agger, van plaats tot plaats en met de intensiteit van het getij.

Waterhoogte berekend Oppervlaktewater t.o.v. NAP

📍 Hoek van Holland



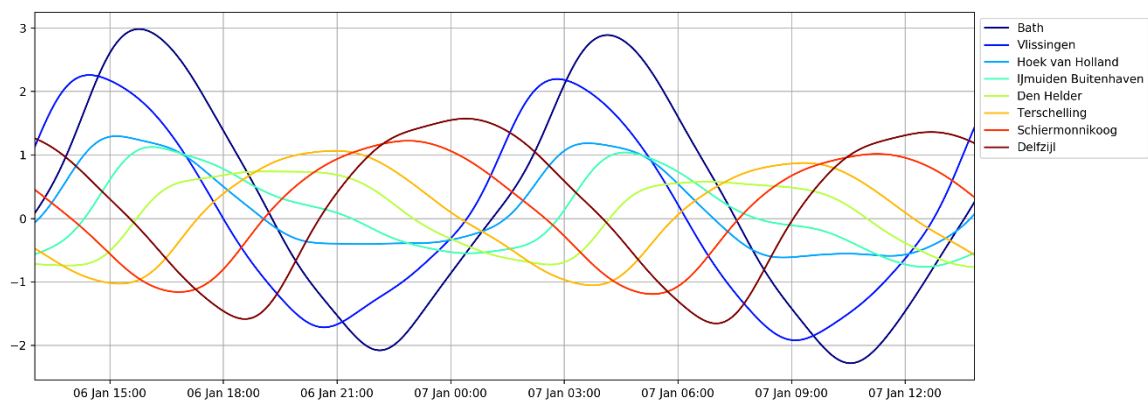
Figuur 2.7 Voorbeeld van agger bij het laagwater van Hoek van Holland (bron: waterinfo.rws.nl)

2.2.3 Propagatie langs de Nederlandse kust

De getijgolf op de Noordzee vindt zijn oorsprong in de Atlantische Oceaan op het Zuidelijk Halfrond. Deze getijgolf zal in de Noordzee deels worden doorgelaten en deels worden gereflecteerd. Het deel van de getijgolf dat doorgelaten wordt, verandert door de afnemende waterdiepte van richting op de overgang van oceaan naar zee. Daarnaast wordt door verondieping de voortplantingssnelheid van de golf kleiner en de golfhoogte groter. De getijslag in de diepere delen van de Noordzee is dan ook vrijwel overal groter dan die in de Atlantische Oceaan, waar de getijslag gemiddeld circa 1 meter is.

Als gevolg van enige bodemwrijving en kustvormen zal de getijgolf in de Noordzee ook een deel van zijn energie verliezen, waardoor de getijslag vervolgens weer wat afneemt. Ook de vormen van de estuaria van de Zuidwestelijke Delta, de Rijn Maasmonding en de Waddenzee beïnvloeden de vorm en fasering van de getijgolf. Omdat de golftop echter minder sterk wordt afgeremd dan het golfdal, krijgt het getij een asymmetrisch verloop

De getijkrommen van een aantal meetpunten langs de Nederlandse kust laten onderlinge verschillen zien tussen hoog- en laagwaterstanden (Figuur 2.8). De grootste tijverschillen in Nederland treden op bij Bath achterin de Westerschelde door stuwing van de getijgolf in het estuarium, deze is hier gemiddeld 4,80 meter tussen hoog- en laagwater. Vanaf Vlissingen neemt het tijverschil in noordelijke richting langs de kust duidelijk af. Bij Den Helder is het effect minimaal door de nabijheid van het amfidrome punt (zie 2.2.4), waarna het weer langzaam toeneemt.



Figuur 2.8 Verloop van het getij langs de Nederlandse kust, beginnend in het zuiden (Bath) en eindigend in het noorden (Delfzijl)

2.2.4

Amfidromische punten

Door veel waarnemingen te doen en deze informatie te combineren met de uitkomsten van numerieke modellen is het mogelijk op de Noordzee lijnen te trekken waar gelijktijdig hoogwater bestaat. Deze zogenaamde 'cotidal lines' kunnen voor zowel voor het maansgetij als voor het zonsgetij worden getekend. Ook kunnen ze worden gecombineerd (= zonsgetij + maansgetij).



Figuur 2.9 Lijnen van dubbeldaags maansgetij en amfidromische punten. De lijnen zelf geven de fase van het maansgetij in uren aan.

De lijnen van het overheersende dubbeldaagse maansgetij in bovenstaand figuur laten zien dat er op de Noordzee twee punten bestaan, waar de lijnen samenvallen en waar het getij omheen draait. Op deze knooppunten, ook wel amfidromische punten genoemd, komt vrijwel geen getijbeweging voor. De lijnen zelf geven de fase van het maansgetij in uren aan.

2.3 Harmonische analyse

Het getij kan worden weergegeven als de som van een aantal cosinussen, waarvan de perioden corresponderen met die van de bewegingen van de zon en de maan. Elke cosinus is daarbij een harmonische component, die wordt gekenmerkt door een golflengte en een amplitude. Aangezien we de bewegingen van de aarde, de maan en de zon precies weten, kunnen we de belangrijkste componenten (dus golflengten) van tevoren al precies definiëren. De amplituden en periodes/fasen worden vervolgens met behulp van de kleinste kwadratenmethode afgeleid uit waterstandsmetingen – de methode van harmonische analyse. Bij een harmonische analyse wordt het gemeten waterstandssignaal dus gesplitst in een aantal harmonische (cosinusvormige) componenten.

De algemene formule is:

$$h(t) = H_0 + \sum_{i=1}^N f_i H_i \cos\{\omega_i t + (v_0 + u)_i - g_i\}$$

met

$h(t)$	= waterstand op tijdstip t
t	= tijd (GMT)
H_0	= middenstandsvlak
N	= aantal componenten
f_i	= correctieterm amplitude voor 18,6-jarige cyclus
H_i	= amplitude van component i
ω_i	= hoeksnelheid van component i
$v_0 + u$	= fase evenwichtsgetij op $t = 0$, met correctieterm u voor de 18,6-jarige cyclus.
g_i	= verbeterd kappa-getal (fase) van component i

Het verbeterd kappa-getal brengt het faseverschil in rekening tussen het evenwichtsgetij en het werkelijke getij op de gewenste locatie en de bijbehorende tijdzone. Het evenwichtsgetij is het getij dat op zou treden wanneer de aarde volledig was bedekt door een diepe oceaan zonder continenten. De aanwezigheid van continenten en ondiepe zeeën en estuaria zorgt er echter voor dat het werkelijke getij qua amplitude en fase sterk afwijkt van het evenwichtsgetij.

H_i en g_i worden de getijconstanten genoemd. Deze worden berekend uit de waarnemingen door middel van een kleinste-kwadratenaanpassing.

Als de getijconstanten eenmaal bekend zijn, kan het getij worden voorspeld voor een willekeurig jaar, mits de morfologische veranderingen en menselijke ingrepen in de omgeving van het station waarvoor een voorspelling wordt gemaakt, beperkt zijn. Elke getijconstante heeft een internationaal overeengekomen afkorting: Zo wordt het zonsgetij gekarakteriseerd met de letter 'S' en het maansgetij met de letter 'M'. Het dubbeldaagse maans- en zonsgetij worden respectievelijk het M2- en S2-getij genoemd. Enkeldaagse fenomenen hebben het cijfer '1' achter de letter staan, dubbeldaagse '2'. Bijna overal op aarde is in de getijbeweging het aandeel van het dubbeldaags maans- en zonsgetij M2 en S2 dominant, maar vooral in gebieden met sterk vervormde getijbewegingen zijn in het getijsignaal hogere harmonische componenten te onderscheiden, en ook samengestelde componenten, de 'ondiepwatergetijden'. Zo levert bijvoorbeeld de interactie tussen M2 en N2 (een component dat verband houdt met de ellipsvormigheid van de maanbaan) het MN2-getij op. Ook is het aandeel van meerdaagse componenten vaak groot, zoals bijvoorbeeld de getijcomponent MS4.

2.4 Componentensplitsing

Bij korte reeksen is er niet genoeg data beschikbaar om alle gewenste componenten van elkaar te onderscheiden, doordat de frequenties te dicht op elkaar liggen. Er wordt in dit geval een analyse gedaan met een deel van de gewenste componenten, waarbij er een aantal worden gesplitst

zodat uiteindelijk amplitudes en fases voor alle gewenste componenten beschikbaar zijn. Componentensplitsing wordt veelal toegepast als er een maand aan data beschikbaar is, Bijlage A bevat de betreffende componentenset. Hierbij worden doorgaans de volgende componenten gesplitst om meer componenten te verkrijgen:

- K1 in K1 en P1
- N2 in N2 en NU2
- 2MN2 in 2MN2 en LABDA2
- S2 in S2, K2 en T2

Voor iedere set aan splitsingscomponenten wordt een verhouding opgegeven voor de amplitude en een verschil in de fase (zie Figuur D.4 in Bijlage D voor een voorbeeld van deze verhoudingen en verschillen). Kort samengevat wordt eerst een samengevoegde component uitgerekend, voor te stellen als een vector met amplitude en fase (bijvoorbeeld K1). Deze vector wordt vervolgens gesplitst in twee vectoren met ieder een amplitude en fase (bijvoorbeeld K1 en P1), welke zich verhouden met de opgegeven amplitudefactor en het opgegeven verschil in fase. De werking van componentensplitsing wordt toegelicht in Bijlage C.

2.5 X-factor

De applicatie hatyan is gebaseerd op de theorie, methodieken en principes uit Schureman (1941) en maakt dan ook gebruik van de theoretische knooppactoren³ zoals die zijn gegeven in deze publicatie. Gebleken is echter dat de theoretische knooppactoren niet goed voldoen voor de Nederlandse peilmeetstations als gevolg van niet-lineaire interacties in ondiepe randzeeën zoals de Noordzee. Omdat knooppactoren zowel bij de analyse als bij de voorspelling worden gebruikt, wordt de fout in de voorspelling versterkt.

In de ideale situatie geldt

$$Gem_{M2} = \frac{1}{n} \sum_{i=1}^n \frac{M2_i}{fM2_i}, \text{ met:}$$

Gem_{M2} = het gemiddelde (in feite het ware) M2-getij

$M2_i$ = het berekende M2-getij uit de waarnemingen van jaar i

$fM2_i$ = de knooppactor voor het M2-getij voor jaar i

n = het aantal geanalyseerde jaren

Uit analyses is echter gebleken dat de correctie met de theoretische knooppactoren voor het M2-getij een sinusoidale oplevert met een tegengestelde fase ten opzichte van het waargenomen M2-getij. Dat betekent dat er sprake is van overcorrectie (Lantsheer, 1985).

Het S2-getij laat ook een cyclus zien met een periode van 18,61 jaar, hoewel deze getijcomponent niet afhankelijk is van de knopencyclus. Om deze reden worden voor de huidige getijtafels waar nodig empirische (of praktische) knooppactoren gebruikt.

Deze praktische knooppactoren zijn afgeleid door het fitten (kleinste kwadratenmethode) van een sinusfunctie met een periode van 18,61 jaar door de amplitudes van de componenten na correctie met de theoretische knooppactoren.

Het verschil tussen de theoretische en de praktische knooppactoren wordt in rekening gebracht door de parameter X_{fac} volgens de formule:

³ Knooppactoren zijn kleine aanpassingen in de amplitudes en fases van harmonische componenten om te compenseren voor modulaties van een 18.6 jarige cyclus, de periode van de knooppactor cyclus.

$$fp_i = X_{fac} \{f_i - 1\} + 1, \text{ ofwel } X_{fac} = \frac{fp_i - 1}{f_i - 1}$$

Met:

fp_i = toe te passen praktische knoofactor voor jaar i

f_i = theoretische knoofactor voor jaar i

De default waarde voor X_{fac} is gelijk aan 1. In dat geval is de empirische knoofactor gelijk aan de theoretische. Voor niet-knoopafhankelijke componenten die toch een 18,61-jarige cyclus laten zien wordt een praktische knoofactor berekend volgens

$$fp_i = X_{fac} (fM2_i - 1) + 1$$

Waarin X_{fac} default de waarde 0 heeft en $fM2_i$ de ongewijzigde knoofactor van de M2-component is. Feitelijk is in hatyan deze mogelijkheid alleen beschikbaar voor S2. *Er zijn ook componenten waarvan de theoretische modulatie een andere periode van 18,61 jaar vertoond (bijvoorbeeld L2). Hierop is de methode met X_{fac} natuurlijk ook toepasbaar.*

Voor alle Nederlandse locaties worden voor de amplitudes de in de onderstaande tabel gegeven waardes voor de X_{fac} toegepast. De parameter X_{fac} is in hatyan ingebouwd.

Tabel 2.1 X_{fac} voor amplitudes voor de relevante componenten

component	X_{fac}
μ_2	0
N2	0
v2	0,8
M2	0,53
2MN2	0,2
S2 ⁴	-0,82
M4	0,7
MS4	0
M6	0,75
2MS6	0,2
M8	0,7
3MS8	0,6

⁴ Gerelateerd aan de theoretische knoofactor van het M2-getij (fM2)

3 Belangrijkste hatyan invoerbestanden

De belangrijkste invoerbestanden voor hatyan zijn⁵:

1. equidistante dia-files
2. non-equidistante dia-files
3. component-files
4. slotgemiddeldenbestand

Deze bestanden worden in de volgende paragrafen behandeld.

3.1 Bestand met tijdserie (equidistante dia-files)

Tijdseries worden door hatyan doorgaans ingelezen of weggeschreven als dia-bestanden (Figuur 3.1). Dit is een bestand met een equidistante tijdserie van een meting of predictie, inclusief de onderliggende metadata in de header van dit bestand.

```
[IDT;*DIF*;A;CENT;20180110]
[W3H]
WNS;1
PAR;WATHTEBRKD;;Yy
CFM;10
EHD;I;cm
HDH;NAP
ANI;RIKZITSDHG
BHI;RIKZITSDHG
BMI;NVT
OGI;RIKZMON_WAT
LOC;BATH
ANA;F012
BEM;NVT
BEW;NVT
VAT;NVT
TYP;TE
[TPS]
STA;20190101;0000;20191231;2350;0
[RKS]
TYD;20190101;0000;20191231;2350;10;min
[WRD]
210/0:197/0:184/0:171/0:157/0:143/0:129/0:115/0:101/0:86/0:71/0:56/0:41/0:25/0:10/0:-5/0:-21/0:-36/0:-51/0:-66/0:-81/0:
-95/0:-109/0:-122/0:-134/0:-145/0:-154/0:-162/0:-168/0:-173/0:-176/0:-177/0:-177/0:-175/0:-172/0:-168/0:-162/0:-155/0:
-146/0:-137/0:-126/0:-114/0:-102/0:-90/0:-77/0:-65/0:-54/0:-43/0:-31/0:-20/0:-9/0:3/0:16/0:29/0:44/0:60/0:76/0:93/0:
110/0:128/0:145/0:162/0:178/0:193/0:206/0:217/0:226/0:233/0:237/0:238/0:236/0:231/0:223/0:214/0:203/0:191/0:178/0:164/0:
151/0:137/0:124/0:110/0:96/0:82/0:68/0:53/0:38/0:23/0:7/0:-8/0:-23/0:-39/0:-54/0:-68/0:-83/0:-98/0:-112/0:-126/0:-140/0:
-152/0:-164/0:-174/0:-183/0:-191/0:-198/0:-203/0:-207/0:-209/0:-210/0:-210/0:-208/0:-204/0:-199/0:-191/0:-182/0:-170/0:
-158/0:-145/0:-131/0:-117/0:-104/0:-90/0:-77/0:-65/0:-52/0:-39/0:-25/0:-11/0:5/0:21/0:39/0:58/0:77/0:97/0:117/0:136/0:
156/0:174/0:192/0:208/0:222/0:234/0:243/0:249/0:253/0:254/0:252/0:247/0:240/0:231/0:221/0:210/0:198/0:185/0:171/0:158/0:
```

Figuur 3.1 Dia-bestand met een equidistante tijdreeks. Het bestandsformaat waarin tijdseries van metingen en predicties in DONAR worden opgeslagen.

Bij het inlezen van een equidistante dia-file wordt door hatyan gecontroleerd op minimale content van de header, dit zijn de regels 'PAR', 'LOC', 'HDH' en 'TYD'. Verder zijn er twee toegestane parameternamen, namelijk 'WATHTE' en 'WATHTEBRKD' (voor metingen versus predicties).

Er wordt afhankelijk van de instelling van `SETTINGS.stations_strict` in de configuratiefile gecontroleerd of de stationsnaam in de header van de dia-file overeenkomt met het station dat op dat moment geanalyseerd wordt. Ook wordt er in sommige gevallen gecontroleerd of het verticale referentieniveau overeen komt met die is gedefinieerd in de verwachting, zoals bijvoorbeeld voorafgaand aan het samenvoegen van analyseresultaten.

Mocht een aangeleverde tijdreeks met meetdata hiaten bevatten, dan zal hatyan deze waarden verwijderen voorafgaand aan de analyse. Enkele vuistregels hierbij:

- Het is aan de gebruiker te bepalen hoeveel hiaten nog kunnen resulteren in een realistisch resultaat van de getijanalyse.

⁵ Het is ook mogelijk met tijdseries in NOOS-format of matfiles te werken. Deze functionaliteit is nodig geweest in het begin van het ontwikkelproces voor validatie tijdens het ontwikkelen, maar is niet gedocumenteerd.

- Het is in ieder geval duidelijk dat het niet verstandig is een dataset van 19 jaar te gebruiken voor een analyse, waarvan de eerste helft van de dataset waardes bevat voor iedere 60 minuten en de tweede helft waardes voor iedere 10 minuten.
- Wat wel goed werkt is het los analyseren van twee niet aansluitende datasets en deze vervolgens samen te voegen. Er zijn stations waar een analyse wordt gedaan voor 2010 en 2012, welke vervolgens vectorieel worden gemiddeld.

3.2 Bestand met extremen (niet-equidistante dia-files)

Extremen (hoog- en laagwaters) worden opgeslagen in DONAR-format in niet-equidistante dia-files.

```
[IDT;*DIF*;A;CENT;20190213]
[W3H]
MUX;GETETBRKD2;Getijextreem berekend
GBD;WESTSIDE;Westerschelde
LOC;VLISSGN;Vlissingen;P;RD;3048000;38522000
ANA;F012;Waterhoogte astronomisch mbv harmonische analyse
[MUX]
MXP;2;WATHTBRKD;Waterhoogte berekend;J
MXC;2;10;Oppervlaktewater
MXE;2;I;cm
MXH;2;NAP;T.o.v. Normaal Amsterdams Peil
[TYP]
TVL;1;1;hoogwater
TVL;1;2;laagwater
TVL;1;3;laagwater 1
TVL;1;4;topagger
TVL;1;5;laagwater 2
[RKS]
TYD;20190101;0405;20191231;2335
PLT;NVT;-999999999;3048000;38522000
SYS;CENT
[TPS]
STA;20190101;0405;20191231;2335;0
[WRD]
20190101;0405;2/0;-133:
20190101;1015;1/0;173:
20190101;1656;2/0;-168:
20190101;2255;1/0;189:
20190102;0524;2/0;-141:
20190102;1118;1/0;183:
20190102;1756;2/0;-172:
20190102;2355;1/0;198:
20190103;0616;2/0;-155:
```

Figuur 3.2 Dia-bestand met een niet-equidistante tijdreeks. Het bestandsformaat waarin tijdseries van metingen en predicties in DONAR worden opgeslagen.

Bij het inlezen van een niet-equidistante diafile wordt door hatyan gecontroleerd op minimale content van de header, dit zijn de regels 'MUX', 'LOC' en 'MXH;2'. Verder is er één toegestane parameternaam, namelijk 'GETETBRKD2' (voor predictie van extremen).

Er wordt, net als bij equidistante dia-files, gecontroleerd of het station en het verticale referentieniveau uit de header overeenkomt met de verwachting, indien dit zo is ingesteld in de configuratiefile.

3.3 Bestand met analyseresultaat: componentenbestand

Een analyseresultaat (amplitudes en fases per component) wordt door hatyan na uitvoering van de analyse opgeslagen in een componentenbestand. Dit resultaat kan vervolgens voor een latere predictie met hatyan weer worden ingelezen.

Dit componentenbestand is een bestandsformaat afkomstig uit Fortran-HATYAN waarin per component de amplitude en fase zijn opgeslagen: Dit formaat is overgenomen in hatyan, zodat ook oudere componenten bestanden uit Fortran-HATYAN toch kunnen worden aangeboden aan hatyan ten behoeve van een predictie.

Verder staat in dit bestand de analyseperiode, een middenstand (H0), het aantal componenten en of het een samengevoegd resultaat betreft van meerdere periodes. Figuur 3.3 en Figuur 3.4 geven twee voorbeelden van componentenbestanden weer, die van een analyse van een enkel jaar en die van een samengevoegde analyse.

```
* Analyse over 200901010000 t/m 201212312300
* geen componenten afgeleid met andere periode
* Met empirische knooppfactorcorrecties (xfac=1)
* Station VLISSGN
STAT VLISSGN WATHT E      NAP      cm      1
PERD 20090101 0000 20121231 2300 60
CODE 3
MIDD 0.264
NCOM 94
COMP 1 0.041069 9.050 198.42 SA
COMP 6 1.015896 3.463 21.78 SM
COMP 17 13.398661 3.055 128.63 Q1
COMP 20 13.943036 10.341 191.97 O1
COMP 24 14.492052 1.526 116.90 MLC
```

Figuur 3.3 Bestand met analyseresultaten van een enkel jaar (2010, PERD). Rechts is dit vectorieel gemiddeld met drie ander jaren (header en PERD) en samengevoegd met de analyse voor SA en SM uit een 19-jarige analyse (eerste twee COMP-regels: SA en SM). Tot slot is de middenstand H0 vervangen door een slotgemiddelde (MIDD).

```
* Analyse over 200901010000 t/m 201212312300
* niet-primaire const_group (['SA', 'SM']) over 197601010000 t/m 199412312300 afgeleid
* Met empirische knooppfactorcorrecties (xfac=1)
* Station VLISSGN
STAT VLISSGN WATHT E      NAP      cm      1
PERD 20090101 0000 20121231 2300 60
CODE 3
MIDD 1.000
NCOM 94
COMP 1 0.041069 7.406 216.05 SA
COMP 6 1.015896 3.900 33.20 SM
COMP 17 13.398661 3.055 128.63 Q1
COMP 20 13.943036 10.341 191.97 O1
COMP 24 14.492052 1.526 116.90 MLC
```

Figuur 3.4 Bestand met analyseresultaten, vectorieel gemiddeld over vier jaren (header en PERD) en samengevoegd met de analyse voor SA en SM uit een 19-jarige analyse (eerste twee COMP-regels: SA en SM). Tot slot is de middenstand H0 vervangen door een slotgemiddelde (MIDD).

3.4 Bestand met slotgemiddelden

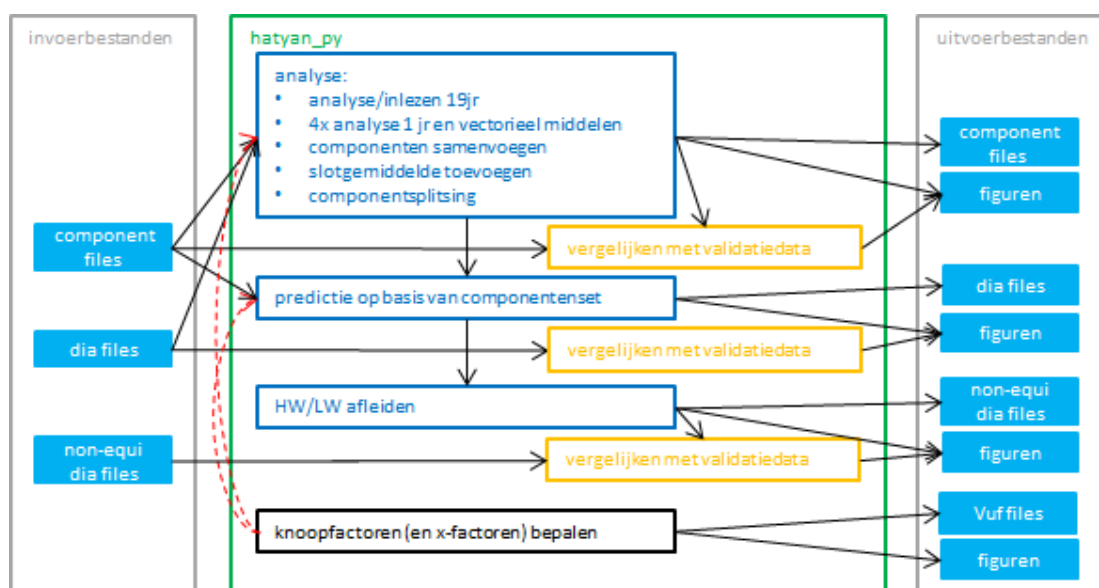
Uit een getijanalyse volgt ook een gemiddelde waterstand. Omdat deze waarde slechts representatief is voor de (beperkte) geanalyseerde periode, wordt deze vaak vervangen door het slotgemiddelde voordat er een predictie wordt gemaakt. Dit slotgemiddelde is een gemiddelde waterstand gebaseerd op een trendlijn afgeleid over een periode van tien jaar (RWS-RIKZ, 1991).

In het bestand met slotgemiddelden staat per station het slotgemiddelde dat wordt gebruikt voor de predictie. Het bij hatyan meegeleverde bestand ('station_middenstanden.txt') bevat de slotgemiddelden die zijn toegepast voor de predictie van 2019, maar dit kan vervangen worden door een willekeurige lijst met stationsnamen en waarden.

4 Werkproces hatyan

4.1 Inleiding

Bij het analyseren en voorspellen van het getij zijn er een aantal stappen in het werkproces te onderscheiden. Dit hoofdstuk bevat een nadere functionele en technische beschrijving van het hatyan werkproces en de mogelijke invoer- en uitvoerbestanden. Figuur 4.1 geeft een overzicht van deze werkprocessen. Het door hatyan gevolgde werkproces is direct afhankelijk van de door de gebruiker gebruikte functies en bijbehorende instellingen/argumenten in de configuratiefile. In de volgende paragrafen wordt per stap van het werkproces (analyse, predictie, extremen) in meer detail ingegaan op de verschillende functionele en technische aspecten van elke stap.



Figuur 4.1 Werkproces hatyan. Invoerbestanden (lichtblauwe blokken links) als tijdreeksen en analysesresultaten worden ingelezen en afhankelijk van de invoer in de configuratiefile worden de verschillende interne stappen doorlopen (alles binnen het groene blok), die vervolgens bijbehorende uitvoer (lichtblauwe blokken rechts) genereren. Het bepalen van knoopfactoren en x-factoren is een intern proces en speelt een rol bij zowel de analyse als de predictie.

4.2 Analyse

4.2.1 Inleiding

In deze workflow wordt een harmonische analyse uitgevoerd op basis van door de gebruiker aangeboden tijdseries van bijvoorbeeld gemeten of berekende waterstanden. De standaard analyse bij Rijkswaterstaat omvat de analyse van 94 getijcomponenten op basis van gemeten waterstanden, waarbij een 19-jarige periode wordt geanalyseerd voor de componenten SA en SM en een 4-jarige periode voor de overige 92 componenten. Bij de analyse over de periode van vier jaar worden de resultaten per jaar geanalyseerd en vervolgens vectorieel gemiddeld. Vervolgens worden de analyseresultaten van de 4-jarige en de 19-jarige analyse samengevoegd, eventueel het slotgemiddelde toegevoegd en eventueel componentensplitsing toegepast. De knooppactoren worden alleen in het midden van de analyseperiodes berekend en voor de gehele periode gebruikt. Het resultaat van de analyse is per meetlocatie een bestand met doorgaans 94 getijcomponenten, welke kan worden gebruikt als invoer voor de getij predictie.

4.2.2 Invoer

Voor het uitvoeren van een analyse worden dia-files en/of componentenfiles als invoer gebruikt (zie links in Figuur 4.1). Dit kunnen bijvoorbeeld dia-bestanden uit DONAR zijn met daarin de tijdseries van metingen, maar kan ook de componentenfile zijn met SA en SM component zijn uit een voorgaande analyse van 19 jaar. De bestanden worden in principe door de gebruiker verzameld en klaargezet, maar er zijn ook een aantal bestanden meegeleverd met hatyan. Naast dia-files, kunnen dit ook bestanden in NOOS-format zijn, afkomstig uit Matroos.

4.2.3 Analyse meerdere periodes

Bij de analyse worden binnen hatyan meerdere periodes gedefinieerd. De standaard analyse van Rijkswaterstaat omvat de analyse van een 4-jarige en een 19-jarige periode. Indien deze data niet beschikbaar is, worden voor een of beide datasets kortere periodes gebruikt. Als er zeer weinig data beschikbaar is, bijvoorbeeld 1 maand, wordt componentensplitsing toegepast.

Merk op: In de huidige analyses zoals die tot op heden met Fortran-HATYAN zijn uitgevoerd worden alle gebruikte tijdseries uitgedund tot een interval van 60 minuten. Dit betekent dat er bij analyse van een dataset met een interval van 10 minuten, vijf van de zes datapunten worden genegeerd. Veelal konden in het verleden de 19 jarige periodes alleen met een interval van 60 minuten geanalyseerd worden, omdat pas vanaf 1987/1988 de data met een interval van 10 minuten wordt ingewonnen en opgeslagen. Inmiddels is het voor de analyse van zowel de periode van 19 jaar als de periode van 4 jaar mogelijk om alle tien minuten waardes te gebruiken en niet meer uit te dunnen.

4.2.3.1 4-jarige analyse

De meeste getijcomponenten (doorgaans alle 94 uit de componentenset voor 1 jaar in Bijlage A, behalve de laagfrequente componenten SA en SM) worden afgeleid uit een harmonische analyse van een tijdserie van metingen van bij voorkeur een periode van 4 jaar. De jaren worden per kalenderjaar geanalyseerd, waarbij de knooppactoren (f en u) steeds voor het midden van iedere afzonderlijke periode wordt bepaald. De amplitudes en fases uit de analyseresultaten van de afzonderlijke jaren worden vervolgens vectorieel gemiddeld en zo samengevoegd tot een analyse van 4 jaar aan data. De keuze voor een periode van vier jaar is bij voorkeur een relatief recent beschikbare periode.

Merk op: Voor de predicties van 2016 tot aan 2019 is voor veel meetlocaties gebruik gemaakt van meetdata van de periode van 2009 tot en met 2012. Omdat inmiddels recentere periodes beschikbaar zijn, lijkt actualisatie van de periode toepasselijk.

Merk op: De knooppactoren, samenhangend met een component met een periode van 18,61 jaar (die niet expliciet in de analyse wordt meegenomen vanwege de lengte van de daarvoor

benodigde reeks), worden bepaald voor het midden van het betreffende jaar en worden verder constant verondersteld gedurende dat jaar. Een mogelijke verbetering is het berekenen van de knooppfactor per tijdstap in plaats van in het midden van een analyseperiode en vier jaar ineens analyseren.

4.2.3.2 19-jarige analyse

Omdat de componenten SA en SM langjarige variaties vertonen, waaronder een van 18,6 jaar, is het niet goed mogelijk om deze componenten via een harmonische analyse uit een tijdserie van slechts een of meerdere jaren te verkrijgen. Voor de analyse van deze componenten wordt daarom bij voorkeur een periode van 19 jaar aaneengesloten data gebruikt. De analyse wordt in dit geval niet steeds per jaar uitgevoerd, omdat de langjarige variaties van SA en SM dan niet gevat zouden worden. In plaats daarvan wordt de analyse in een keer over de gehele periode uitgevoerd, waarbij de knooppfactoren (f en u) in het midden van de periode van 19 jaar worden bepaald.

De door RWS toegepaste periode van 19 jaar is doorgaans een oudere periode, voor veel stations is dit de periode van 1976 tot en met 1994. Tijdreeksen voor een aaneengesloten periode van 19 jaar zijn echter niet voor alle stations beschikbaar en dus wordt in de praktijk soms ook een kortere periode gebruikt of worden de componenten SA en SM overgenomen van een nabijgelegen meetlocatie. Het komt ook voor dat een er in het verleden wel een analyse op 19 jaar aan data is uitgevoerd, maar dat de (exacte) dataset niet meer beschikbaar is. Dit kan komen doordat de data niet meer beschikbaar is of de handmatige wijzigingen, die zijn gedaan voorafgaand aan de analyse destijds, niet zijn opgeslagen. In veel gevallen worden de resultaten voor SA en SM daarom overgenomen uit deze eerdere analyses.

Merk op: Het is van belang dat er binnen een analyseperiode geen veranderingen nabij de meetlocatie hebben plaatsgevonden of maatregelen zijn uitgevoerd die invloed hebben op het getij in de buurt van deze meetlocatie: De aanleg van de tweede Maasvlakte heeft bijvoorbeeld effect op het getij bij Hoek van Holland en andere nabijgelegen stations.

Merk op: ook voor de 19-jarige periode wordt de knooppfactor standaard in het midden van de periode bepaald.

Merk op: het is niet mogelijk is om deze 19-jaars methode toe te passen voor een meer reguliere getijanalyse (alles behalve SA en SM), omdat de knooppfactoren daarvoor teveel variëren. Deze methode is alleen geschikt voor het schatten van SA en SM.

4.2.3.3 Componentensplitsing

Als er weinig data beschikbaar is, kan componentensplitsing worden toegepast als de gebruiker dit nodig acht. Zie hiervoor Hoofdstuk 2.4. Er wordt dan bijvoorbeeld gebruik gemaakt van een gereduceerde lijst van componenten geschikt voor bijvoorbeeld een maand aan data (zie Bijlage A). De werking van componentensplitsing wordt toegelicht in Bijlage C.

4.2.4 Samenvoegen van resultaten

De analyseresultaten worden per station samengevoegd. Behalve het vectorieel middelen van de resultaten van de vier afzonderlijke jaren, worden ook de resultaten van SA en SM uit de 19-jarige analyse samengevoegd met de andere componenten uit de 4-jarige analyse. Tot slot wordt de berekende gemiddelde waterstand vervangen door het slotgemiddelde. Figuur 3.4 geeft twee voorbeelden van uitvoerbestanden met analyseresultaten weer – de zogenaamde componentenfiles .

4.2.5 Uitvoer

Zoals rechts in Figuur 4.1 gespecificeerd, resulteert de workflow analyse in componentenfiles (zoals in Figuur 3.4) en figuren (zoals Figuur 5.2 en Figuur 5.3). Als in de configuratiefile ook

validatiedata is gespecificeerd, staat er in de resulterende figuren ook een vergelijking van de analyse met deze validatiedata. De componentenfiles bestaan per meetlocatie uit een tabel met per component de fase en amplitude.

4.3 Predictie

4.3.1 Inleiding

In deze workflow wordt een predictie uitgevoerd op basis van een reeds bestaande componentenfile of op basis van een componentenfile afkomstig uit een net uitgevoerde analyse. De standaard predictie bij Rijkswaterstaat omvat een predictie op basis van 94 getijcomponenten (zie ook Bijlage A). De predictie wordt in het algemeen een jaar voorruit gemaakt, dus in mei 2020 wordt de predictie voor 2021 gemaakt. Het resultaat van de predictie is een DIA-file per meetlocatie met een tijdreeks met een 10 minuten tijdsinterval. Deze tijdreeks kan in DONAR worden geplaatst voor verdere verspreiding door RWS.

4.3.2 Invoer

Als invoer voor een predictie worden de resultaten van een analyse gebruikt die vlak ervoor is gedaan en als variabele wordt doorgegeven, of er worden analysebestanden ingelezen van een eerdere analyse (van hatyan of Fortran-HATYAN). In ieder geval is de invoer altijd een verzameling van amplitudes en fases per component.

4.3.3 Uitvoer

Zoals rechts in Figuur 4.1 met het werkproces van hatyan gespecificeerd, resulteert een predictie in een bestand met een predictietijdreeks (dia-file, Figuur 3.2) en eventueel figuren (zoals Figuur 5.4, Figuur 5.5 en Figuur 5.6) indien dit gespecificeerd is in de configuratie file. Als in de configuratiefile ook validatiedata is gespecificeerd, staat er in de resulterende figuren ook een vergelijking van de analyse met deze validatiedata.

4.4 Bepaling van extremen (hoog- en laagwaters)

4.4.1 Invoer

Als invoer voor de bepaling van Hoogwaters en Laagwaters uit een tijdreeks wordt een tijdreeks aangeleverd (bijvoorbeeld een equidistante dia-file of een zojuist gemaakte predictie). Eventueel wordt ook validatiedata gebruikt in de vorm van een non-equidistante dia-file met extremen.

4.4.2 Uitvoer

Zoals rechts in Figuur 4.1 met het werkproces van hatyan is gespecificeerd, resulteert een bepaling van extremen in een dia-bestand met een extreme (hoog-/laagwaters) (dia-file, Figuur 3.1) en figuren (zoals Figuur 5.8) . Als in de configuratiefile ook validatiedata is gespecificeerd, staat er in de resulterende figuren ook een vergelijking van de analyse met deze validatiedata.

5 Quick start guide

Dit hoofdstuk dient voor een gebruiker als quick start guide voor het gebruik van de hatyan Python applicatie. Waar Hoofdstuk 3 het werkproces en de onderliggende workflows tot in functioneel en technisch detail beschrijft (inclusief details over verschillende commando's en invoerfiles) beschrijft dit hoofdstuk juist de “hands-on” interactie van de gebruiker met hatyan en de interpretatie en beoordeling van de resultaten die daaruit voortkomen. De invulling van dit hoofdstuk is daarmee vooral praktisch: Aan de hand van een tutorial i.c.m. met de meegeleverde configuratiebestanden en een aantal tests, maakt de gebruiker kennis met de applicatie, de onderliggende mogelijkheden en de data. Voor de Tutorial zijn verschillende meegeleverde configuratiebestanden beschikbaar.

5.1 Klaarzetten van de invoerdata

Voorafgaand aan het gebruik van hatyan-applicatie, dient door de gebruiker alle benodigde data (tijdseries van metingen en/of componenten files) verzameld en klaargezet te worden (blok links in Figuur 4.1). Hoofdstuk 3 geeft een overzicht van deze datatypen.

5.2 Hatyan configuratiefile

Na het klaarzetten van de data, kan de gebruiker er voor kiezen om zelf een configuratiefile te schrijven, of een reeds bestaande configuratiefile aan te passen op de gewenste workflow. Omdat de configuratiefile een standaard Python script is, kunnen behalve alle hatyan functies, ook alle standaard Python functies gebruikt worden.

5.2.1 Voorbeeld en uitleg configuratiefile

In Figuur 5.1 is een simpel voorbeeld van een configuratie weergegeven. Dit voorbeeld bevat de volgende stappen:

- Regel 1 tot 2: importeren van Python libraries
- Regel 3 tot 4: importeren van hatyan Python libraries
- Regel 9 tot 19: definiëren van variabelen om later te gebruiken, in dit geval enkele hatyan instellingen. In regel 15 en 16 wordt de stationsnaam (`current_station`) gebruikt om de paden naar de juiste files te genereren.
- Regel 22: inlezen van een equidistante diafile (tijdreeks met meetdata)
- Regel 23: analyseren van de tijdreeks, resulterend in de amplitude en fase van de componenten gedefinieerd in de variabele `const_list`
- Regel 24: inlezen van een componentenset
- Regel 27: samenvoegen van de twee componentensets
- Regel 30: maken van een predictie van de samengevoegde componentenset
- Regel 31: een figuur maken van de gemaakte predictie

```

1 import os
2 import datetime as dt
3 from hatyan import timeseries as Timeseries
4 from hatyan import components as Components
5 from hatyan.analysis_prediction import get_components_from_ts, prediction
6 from hatyan.hatyan_core import get_const_list_hatyan
7
8 #SETTINGS
9 nodalfactors = 1
10 xfac=1
11 const_list = get_const_list_hatyan(94)
12 selected_stations = ['VLISGN']
13
14 for current_station in selected_stations:
15     file_data_comp0 = [os.path.join('C:\\data', 'tests', 'data', '%s_obs%i.txt'%(current_station, file_id)) for file_id in [1,2,3,4]]
16     file_data_comp1 = os.path.join('C:\\data', 'tests', 'data', '%s_ana.txt'%(current_station))
17
18     times_ext_pred = [dt.datetime(2019,1,1),dt.datetime(2020,1,1)]
19     times_step_pred = 10
20
21     #component groups
22     ts_measurements = Timeseries.readts_dia(filename=file_data_comp0, station=current_station)
23     comp_frommeas_avg = get_components_from_ts(ts=ts_measurements, const_list=const_list, nodalfactors=nodalfactors, xfac=xfac)
24     comp_fromfile = Components.read_components(filename=file_data_comp1)
25
26     #merge component groups (SA/SM from 19Y, rest from 4Y)
27     COMP_merged = Components.merge_componentgroups(comp_main=comp_frommeas_avg, comp_sec=comp_fromfile, comp_sec_list=['SA', 'SM'])
28
29     #prediction and validation
30     ts_prediction = prediction(comp=COMP_merged, nodalfactors=nodalfactors, xfac=xfac, times_ext=times_ext_pred, timestep_min=10)
31     fig, (ax1,ax2) = Timeseries.plot_timeseries(ts=ts_prediction)

```

Figuur 5.1 Simpele configuratiefile voor hatyan

De in dit voorbeeld gebruikte hatyan functies zijn slechts een kleine selectie. In Bijlage F is een volledig overzicht gegeven van alle scripts, de (beschrijving van) functies, argumenten en uitvoer.

5.2.2 Meegeleverde configuratiefiles

Er zijn vier ‘master configfiles’ gedefinieerd, welke de analyse en predictie van alle stations kunnen doorlopen. Binnen de scripts is gedefinieerd voor welke stations een uitzondering geldt zoals het niet gebruiken van de x-factor, het gebruiken van een andere componentenset of het niet analyseren per jaar. De te analyseren jaren worden automatisch afgeleid uit de toegeleverde meetdata. Deze master configfiles bevatten dus in wezen ook een overzicht van de te gebruiken instellingen voor alle stations. De vier master configfiles onderscheiden zich van elkaar door de hoeveelheid analyse er wordt gedaan:

- predictie_2019_frommergedcomp_all.py
 - o inlezen analyseresultatenbestand
 - o predictie maken
- predictie_2019_19Ycomp4Ydia_all.py
 - o analyse van de beschikbare periode aan data, eventueel per jaar en eventueel met componentensplitsing
 - o combineren met SA+SM uit analyseresultatenbestand
 - o predictie maken
- predictie_2019_allfromdia_all.py
 - o analyse van de beschikbare periode aan data, eventueel per jaar
 - o analyse van 19 jaar aan data voor de componenten SA+SM
 - o combineren van de analyseresultaten
 - o predictie maken

De wijze waarop voor verschillende locaties het astronomisch getij wordt afgeleid (“de recepten per locatie”) is in meer detail vastgelegd in het Excelbestand meegeleverd met hatyan (‘data_overzicht_20190528.xlsx’). In dit document staat per station gedocumenteerd welke meetdata (periode) is gebruikt voor analyse en welke instellingen zijn gebruikt voor analyse en predictie voor 2019. Het document is een vastlegging van vele jaren praktische kennis en ervaring bij Rijkswaterstaat omtrent de uitvoering van getijanalyses en –predicties voor Nederland.

Ook zijn er direct onder de map configfiles nog een aantal configuratiefiles meegeleverd waarmee door de gebruiker extra uitvoer gegenereerd kan worden, bijvoorbeeld:

- `spatialsummary.py`: een file waarmee ruimtelijke figuren kunnen worden gemaakt van bijvoorbeeld amplitudes en fases langs de Noordzeekust, afkomstig uit componentenfiles.
- `export_vuf_data.py`: exporteren van textfiles en figuren van vuf-waardes.

Merk op: Bij gebruik van hatyan via de RPM staan de configuratiefiles in de map /opt/hatyan/tests/configfiles/, deze kunnen gekopieerd worden en vervolgens aangepast.

5.3 Draaien van de applicatie

Nadat deze twee stappen zijn doorlopen en nadat er een connectie is gelegd met het systeem waarop hatyan staat geïnstalleerd, kan het werkproces met het volgende commando worden gestart:

```
hatyan [pad_naar_configfile]
```

Voorbeeld van een commando:

```
hatyan  
/opt/hatyan/tests/configfiles/predictie_2019_19Ycomp4Ydia_VLISSGN_test_interactive.py
```

Hierbij wordt de applicatie hatyan aangeroepen, met als invoer het pad en bestandsnaam van de uit te voeren configuratiefile.

Merk op: Omdat de configuratiefile een normaal Python script is, kan hatyan ook worden gebruikt door de configfile met Python te runnen. Het is dus ook mogelijk om zonder installatie van de RPM een getijanalyse te doen met hatyan, mits het pad naar de hatyan scripts toegevoegd wordt. De voorwaarde hiervoor is wel dat er een Python installatie met de benodigde libraries geïnstalleerd is, dit is in feite waar de RPM direct in voorziet.

5.4 Output

De outputbestanden worden altijd weggeschreven in een automatisch door de applicatie nieuw gecreëerde map in de current directory. Deze map heeft naamgeving met een timestamp en de naam van de configuratiefile, bijvoorbeeld:

```
./20190528_1000_predictie_2019_19Ycomp4Ydia_VLISSGN_test_interactive /
```

In deze outputmap staan vervolgens:

1. De resultaten van de run, o.a.: dia-files, componentenbestanden, figuren, etc.,
2. een kopie van het gebruikte configuratiebestand voor die run,
3. een bestand met diagnostische informatie, een kopie van alle output die naar de terminal is geschreven gedurende de hatyan run (file DIAGNOSTICS.txt). In deze logfile staan bijvoorbeeld de gebruikte instellingen, de gebruikte invoerfiles, de stappen die worden doorlopen, het aantal missende waardes in een tijdreeks en eventuele errors die het proces voortijdig af hebben gebroken. De inhoud van de logfile is dus voor iedere run van hatyan uniek.
4. Bij het gebruiken van de functies `init_RWS()` en `exit_RWS()` (zie Bijlage F voor de beschrijving hiervan) staat er tijdens de run een file genaamd `'__NOT_FINISHED__'`, welke wordt verwijderd na een succesvolle afronding van hatyan.

Merk op: Het is mogelijk interactieve plots/figuren te genereren gedurende de run vanaf het Linux-systeem waarop hatyan draait. Deze plots worden dan via X11⁶ doorgegeven naar de omgeving

⁶ X11 = Protocol onder de X Window-server, waarmee niet enkel in een tekstuele omgeving gewerkt kan worden, maar ook gebruikgemaakt kan worden van grafische elementen op de Linux omgeving zoals grafische vensters, gebruik muiscursor, etc.

waar de gebruiker op werkt. Hierdoor heeft de gebruiker de vrijheid om per locatie op specifieke periodes in te zoomen. Ook kunnen er zo handmatig extra figuren worden opgeslagen. Het programma hatyan pauzeert zolang er interactieve plots open staan en gaat pas weer verder als al deze plots zijn afgesloten.

5.5 Interpretatie van resultaten

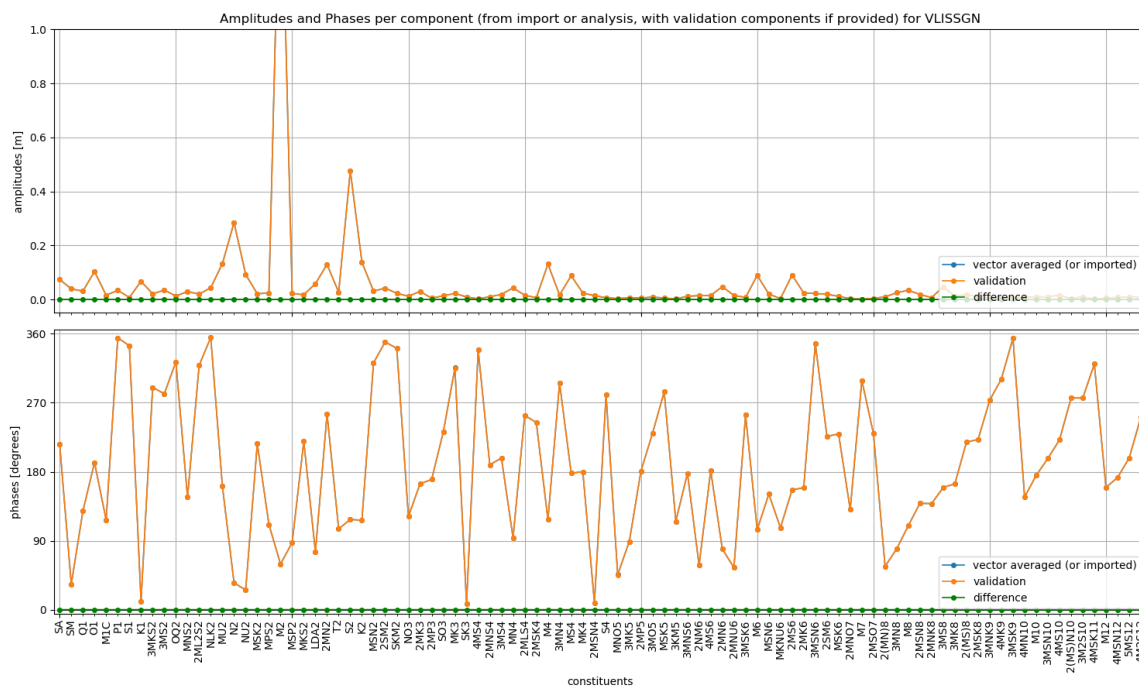
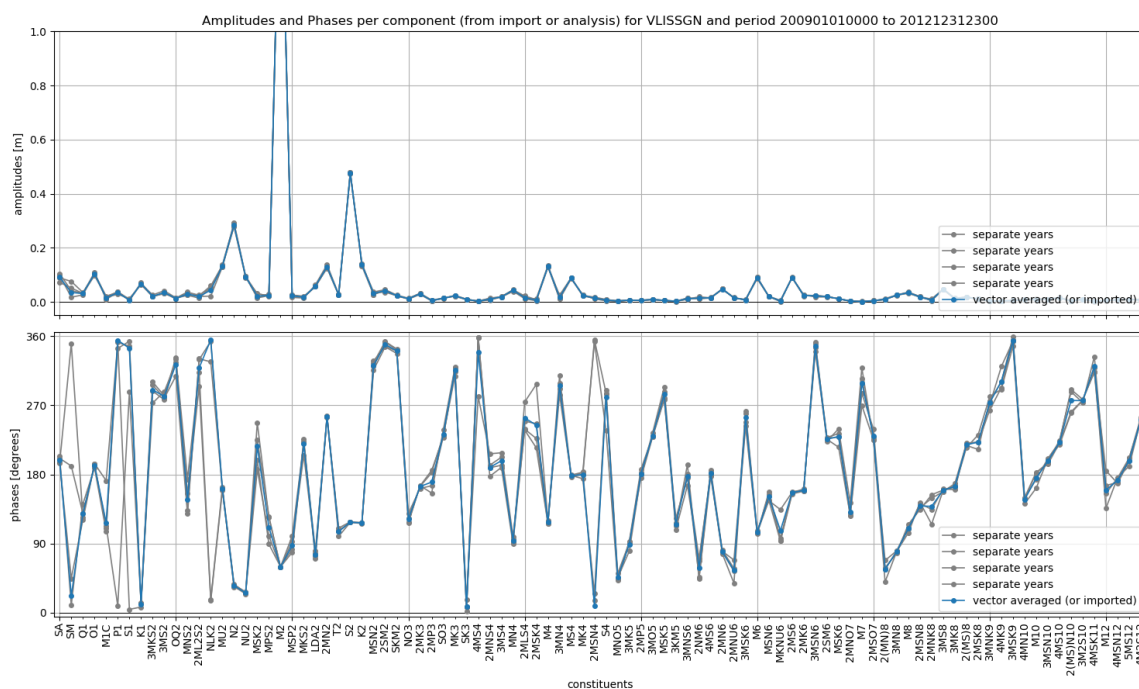
Het doorlopen van het volledige werkproces, bestaande uit de analyse van een periode van 4 jaar voor 92 componenten, analyse voor een periode van SA en SM, samenvoegen resultaten en predictie, resulteert in een aantal figuren waaruit conclusies kunnen worden afgeleid over de kwaliteit van de analyse en de predictie. Hieronder volgen een aantal handvatten voor de gebruiker op grond waarvan deze kan beoordelen of het resultaat van de harmonische analyse en predictie correct tot stand is gekomen of dat er per ongeluk in de onderliggende data of invoer in de configuratiefile onvolkomenheden zijn gekomen.

5.5.1 Visuele inspectie van analyseresultaten

Tijdens het uitvoeren van de analyse genereert de applicatie een aantal plots waarmee de gebruiker per meetlocatie een visuele inspectie kan doen van de analyse resultaten. Eventuele afwijkingen worden daarmee al in een vroeg stadium zichtbaar.

Figuur 5.2 geeft voor de locatie Vlissingen voor de periode 2009-2012 de analyseresultaten van amplitude en fase weer van vier afzonderlijke jaren (grijs), samen met het vectoriele gemiddelde ervan (blauw). Het figuur voldoet aan de verwachting als de resultaten van de vier afzonderlijke jaren redelijk dicht bij elkaar liggen en niet te sterk afwijken van het vectorieel gemiddelde over vier jaar. Dit volgt uit het beeld dat getijcomponenten vaak maar geleidelijk veranderen over de tijd en het hier vier aaneengesloten jaren betreft. Grote afwijkingen tussen de individuele jaren zouden kunnen duiden op eventuele meetfouten of veranderingen in fysisch systeem. Dit kan voor de gebruiker aanleiding zijn om nog eens nader naar de ruwe meetdata te kijken, na te gaan of er mogelijke sensorwisselingen zijn geweest of na te gaan of er in de nabijheid van de meetlocatie gebiedswijzigingen hebben plaatsgevonden.

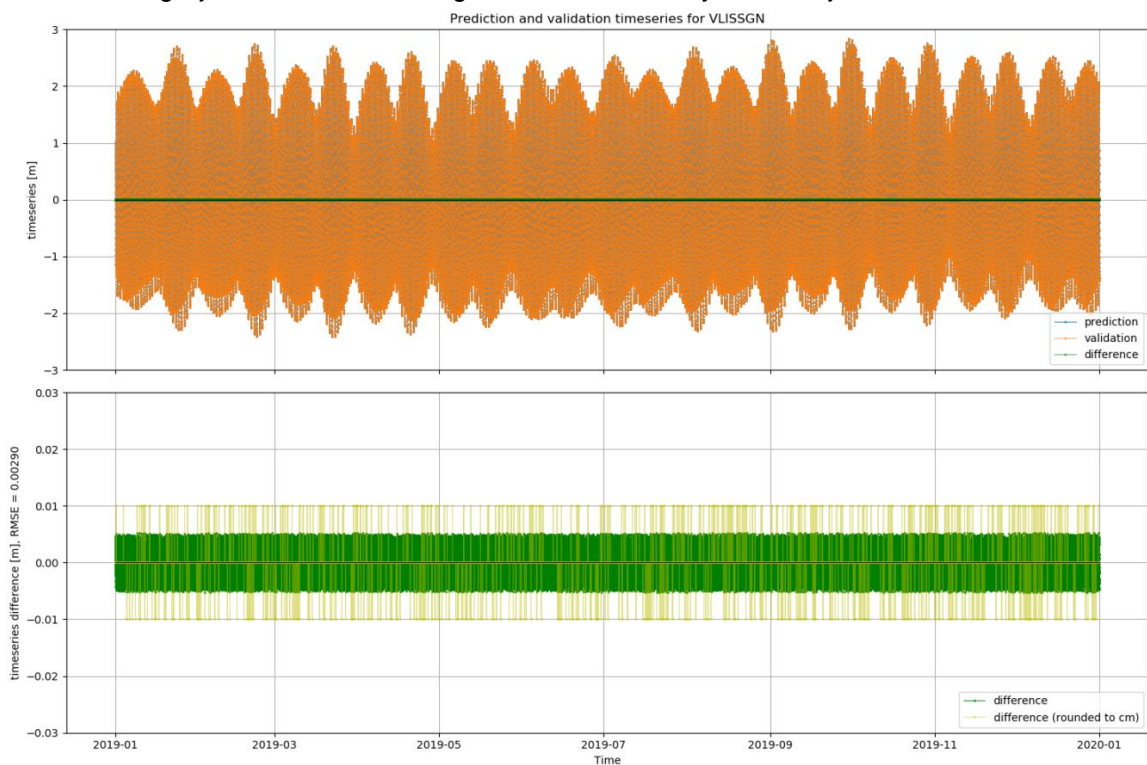
Na het samenvoegen van de verschillende analyseresultaten wordt in Figuur 5.3 het vectoriele gemiddelde resultaat (blauw) vergeleken met (eventueel beschikbare) validatiedata van een eerdere analyse gedaan met Fortran-HATYAN (oranje). Het resultaat van de analyse is gelijk aan de eerdere analyse met Fortran-HATYAN als de blauwe en de oranje lijn van zowel de amplitudes als de fases op elkaar liggen en het verschil (groen) voor alle componenten (vrijwel) gelijk is aan 0.



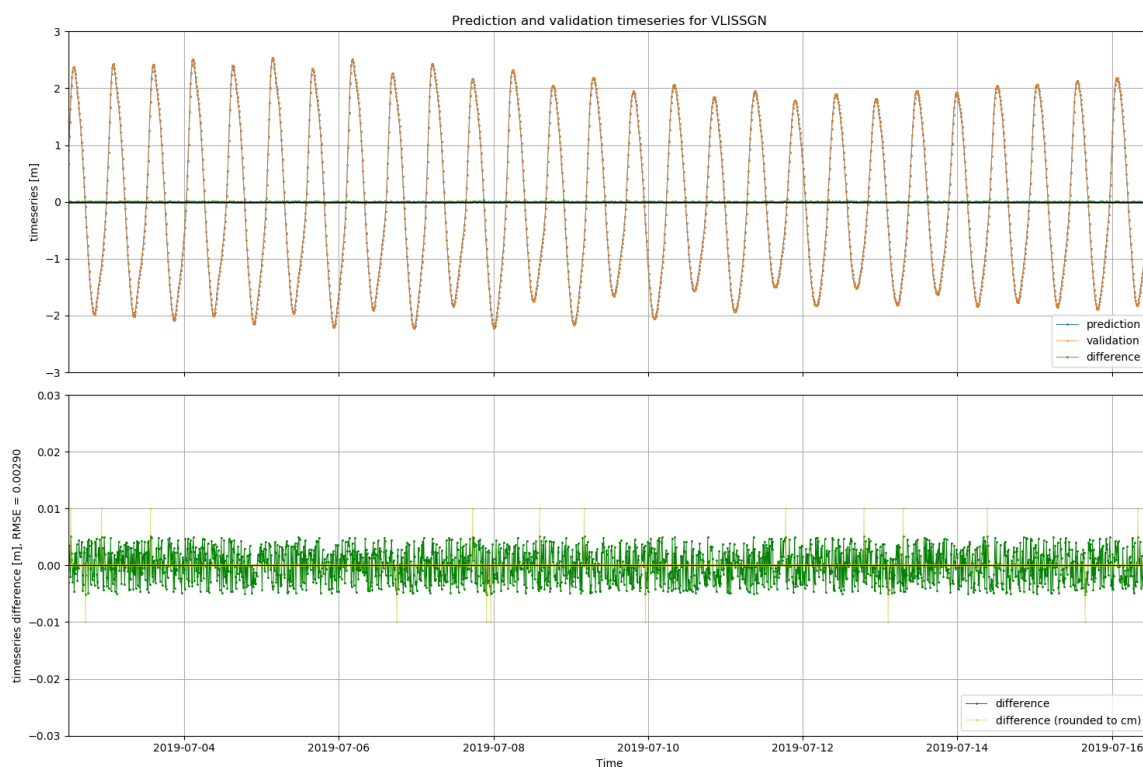
weergegeven. Doordat in de Fortran-HATYAN validatiedataset alle resultaten zijn afgerond op centimeters, is het exact reproduceren van deze predictie van niet mogelijk. Het verschil is daarom niet 0 maar varieert tussen -0.5 en +0.5 centimeter. De figuur is correct als het verschil binnen deze bandbreedte blijft en er hierin geen periodiciteit aanwezig is. Is er wel periodiciteit zichtbaar, dan kan dit duiden op het niet (goed) meenemen van een specifieke component.

Bij het inzoomen op een periode van bijvoorbeeld twee weken (Figuur 5.5), is ook zichtbaar dat de tijdseries (bijna) helemaal op elkaar liggen. Ook is te zien dat er in het verschil tussen beide predicties maar enkele uitschieters zijn die afwijken van de bandbreedte van $\pm 0.5\text{cm}$, door de verschiltijdreeks afgerond naar gehele centimeters te raadplegen (geel).

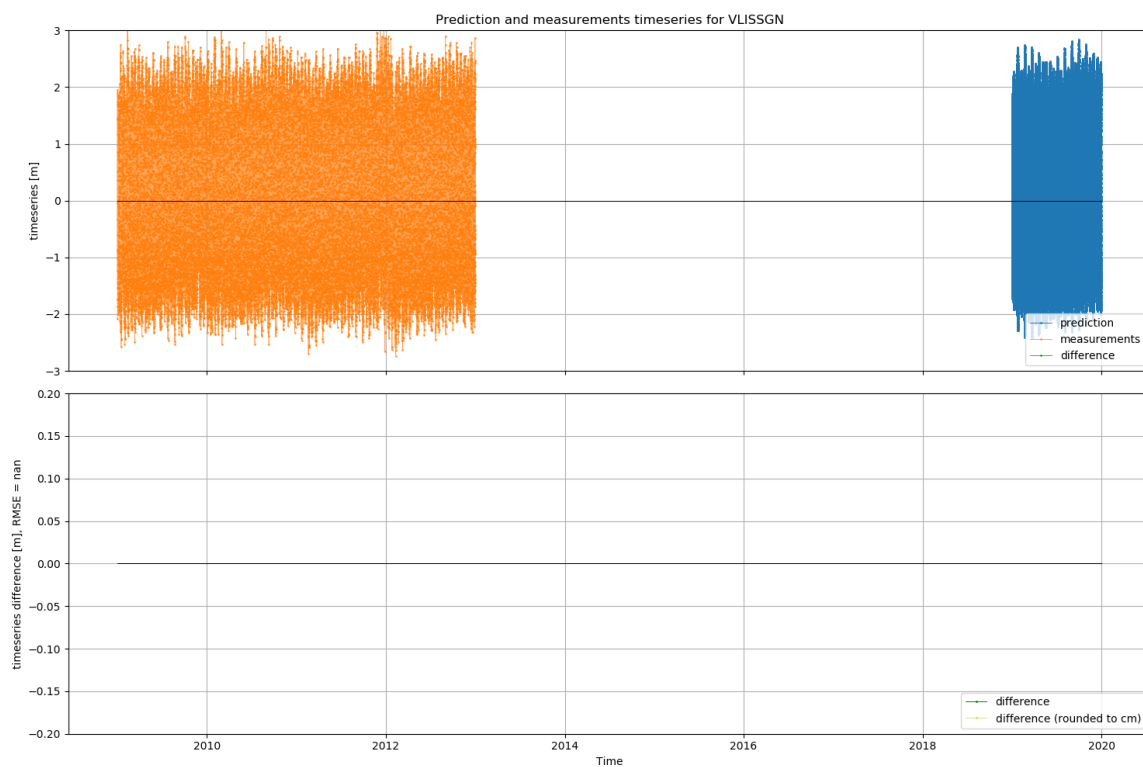
Bij het maken van een predictie wordt ook een vergelijking gemaakt met de meetdata, indien die is gebruikt voor de analyse in hetzelfde werkproces (Figuur 5.6). De opzet is gelijk aan de vergelijking tussen de predictie en validatiedata. Echter, doordat er meetdata gebruikt wordt van een andere periode dan waarvoor de predictie is gedaan, is er geen verschilberekening mogelijk. Dit is wel mogelijk als de periode van de predictie overlapt met die van de geanalyseerde metingen. Daarnaast is het mogelijk om binnen een werkproces predicties voor beide periodes te genereren door dit te definiëren in de configuratiefile. Het beoordelen van het verschil tussen de predictie en de meetdata is aan de gebruiker en verschilt vaak sterk per station. In ditzelfde figuur is het ook mogelijk de meetdata die is gebruikt voor de analyse te bekijken.



Figuur 5.4 Predictie gemaakt met analyseresultaten (blauw), validatiedata (oranje) en het verschil hiertussen (groen, afgerond op centimeters in geel)



Figuur 5.5 Predictie gemaakt met analyseresultaten (blauw), validatiedata (oranje) en het verschil hiertussen (groen, afgerond op centimeters in geel). Ingezoomd op een periode van twee weken.



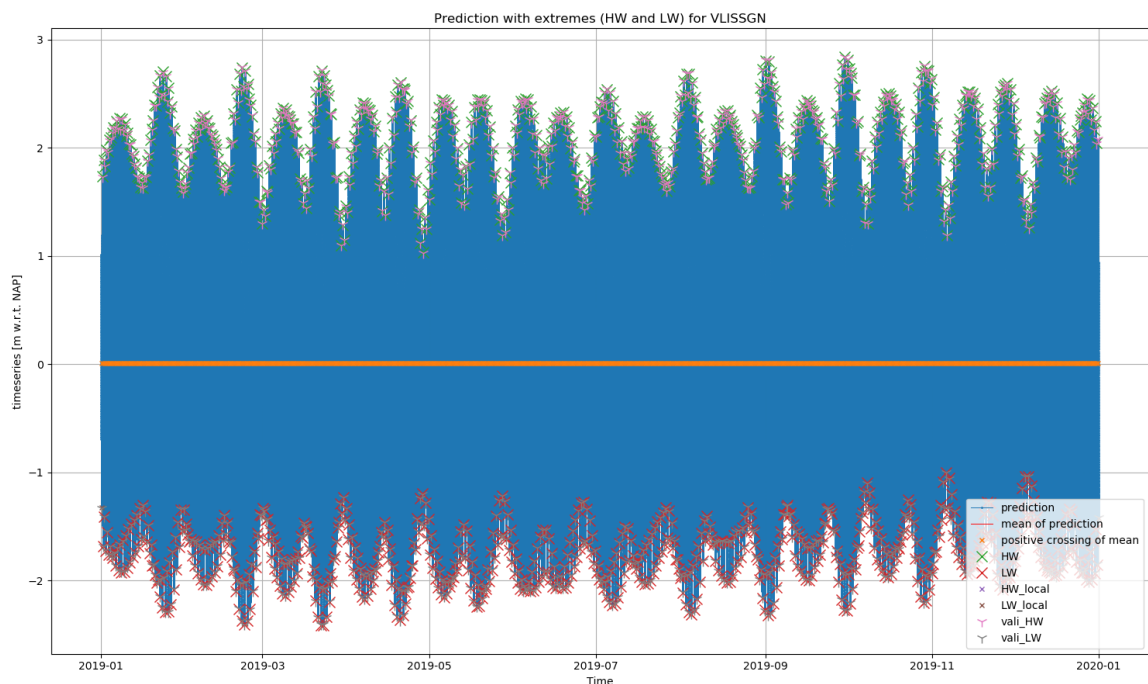
Figuur 5.6 Predictie gemaakt met analyseresultaten (blauw), meetdata gebruikt in de analyse (oranje) en het (hier ontbrekende) verschil hiertussen (groen, afgerond op centimeters in geel)

5.5.3

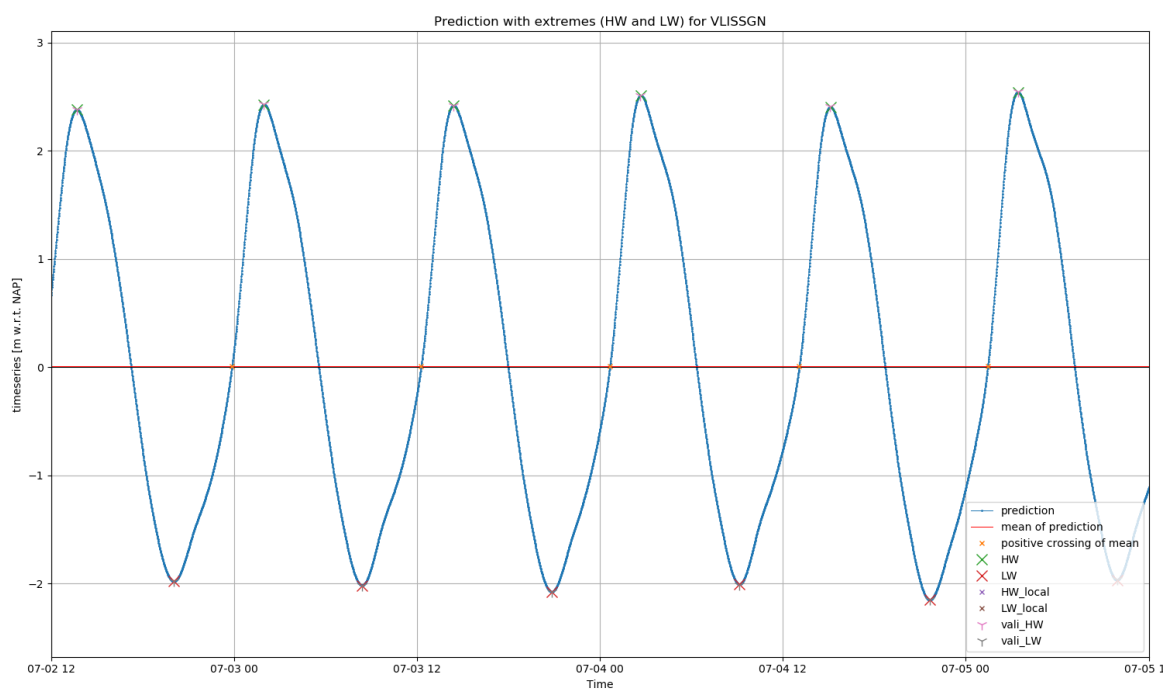
Visuele inspectie van HWLW resultaten

In Figuur 5.7 en Figuur 5.8 is de predictie samen met de berekende voor hoog- en laagwaters weergegeven. Ook zijn de hoog- en laagwaters uit de validatiedataset weergegeven.

Merk op: de oorspronkelijke uitvoerfiles met extremen (gemaakt met het oude Fortran-HATYAN en harbek) bevatten in geval van aggers een verschillende code voor hoogwater, eerste laagwater, agger en tweede laagwater. In hatyan in Python wordt gebruik gemaakt van primaire hoog- en laagwaters voor de extremen, en secundaire hoog- en laagwaters voor de overige waarden. De laatste worden op het moment niet uitgevoerd. Als er geen aggers zijn of worden uitgevoerd, zijn de gebruikte HWLW-codes in de uitvoer gelijk. Als er wel aggers zijn, is de uitvoer op dit moment nog afwijkend.



Figuur 5.7 Predictie gemaakt met analyseresultaten (blauw), inclusief berekende extremen (groene en rode kruizen) en extremen uit validatiedata (roze en grijze driepunten).



Figuur 5.8 Predictie gemaakt met analyseresultaten (blauw), inclusief berekende extremen (groene en rode kruizen) en extremen uit validatiedata (roze en grijze driepunten), ingezoomd op een periode van drie dagen.

6 Referenties

Van der Heijden, A., Roosenschoon, F.G. (jaartal onbekend), Systeem-documentatie HATYAN* (00, 15, 20, 30, 35, 40, 45, 50 en 55)

Irazoqui Apecechea, M. (2018), HATYAN technical design, Deltares rapport 11202223-000-ZKS-0002, 27 Februari 2018

Pugh D., Woodworth P.L. (April 2014). Sea Level Science: Understanding Tides, Surges, Tsunamis and Mean Sea Level changes. Tweede druk, ISBN 9781107028197

RWS-RIKZ (1991). Gemiddelde getijkromme 1991.0, ISBN 90-369-0453-6, Den Haag 1991

RWS-RIKZ (1999), Gebruikershandleiding HATYAN, 1 december 1999

Schureman P, Manual of harmonic analysis and prediction of tides, U.S. Coast and Geodetic Survey, Special Publication No. 98, Washington 1941.

Veenstra J., Kerkhoven D. (2020), Functioneel Ontwerp hatyan getijanalyse in Python, Deltares rapport 11205257-013-DSC-0002 versie 0.4, 5 juni 2020

A Hatyan componentensets

Binnen hatyan zijn enkele vooraf gedefinieerde componentensets beschikbaar. H0 is geen getijdecomponent maar het gemiddelde van de tijdreeks.

Alle beschikbare componenten binnen hatyan ('all'), H0 en 195 componenten:

[H0, SA, SM, Q1, O1, M1C, P1, S1, K1, 3MKS2, 3MS2, OQ2, MNS2, 2ML2S2, NLK2, MU2, N2, NU2, MSK2, MPS2, M2, MSP2, MKS2, LABDA2, 2MN2, T2, S2, K2, MSN2, 2SM2, SKM2, NO3, 2MK3, 2MP3, SO3, MK3, SK3, 4MS4, 2MNS4, 3MS4, MN4, 2MLS4, 2MSK4, M4, 3MN4, MS4, MK4, 2MSN4, S4, MNO5, 3MK5, 2MP5, 3MO5, MSK5, 3KM5, 3MNS6, 2NM6, 4MS6, 2MN6, 2MNU6, 3MSK6, M6, MSN6, MKNU6, 2MS6, 2MK6, 3MSN6, 2SM6, MSK6, 2MNO7, M7, 2MSO7, 2(MN)8, 3MN8, M8, 2MSN8, 2MNK8, 3MS8, 3MK8, 2(MS)8, 2MSK8, 3MNK9, 4MK9, 3MSK9, 4MN10, M10, 3MSN10, 4MS10, 2(MS)N10, 3M2S10, 4MSK11, M12, 4MSN12, 5MS12, 4M2S12]

Componentenset voor een jaar ('year'), H0 en 94 componenten:

[H0, SA, SM, Q1, O1, M1C, P1, K1, 3MKS2, 3MS2, OQ2, MNS2, 2ML2S2, NLK2, MU2, N2, NU2, M2, LABDA2, 2MN2, S2, K2, MSN2, 2SM2, SKM2, NO3, 2MK3, 2MP3, SO3, MK3, SK3, 4MS4, 2MNS4, 3MS4, MN4, 2MLS4, 2MSK4, M4, 3MN4, MS4, MK4, 2MSN4, S4, MNO5, 3MK5, 2MP5, 3MO5, MSK5, 3KM5, 3MNS6, 2NM6, 4MS6, 2MN6, 2MNU6, 3MSK6, M6, MSN6, MKNU6, 2MS6, 2MK6, 3MSN6, 2SM6, MSK6, 2MNO7, M7, 2MSO7, 2(MN)8, 3MN8, M8, 2MSN8, 2MNK8, 3MS8, 3MK8, 2(MS)8, 2MSK8, 3MNK9, 4MK9, 3MSK9, 4MN10, M10, 3MSN10, 4MS10, 2(MS)N10, 3M2S10, 4MSK11, M12, 4MSN12, 5MS12, 4M2S12]

Componentenset voor een half jaar ('halfyear'), H0 en 88 componenten:

[H0,SA,SM,Q1,O1,M1C,P1,K1,3MKS2,3MS2,OQ2,MNS2,2ML2S2,NLK2,MU2,N2,NU2,M2,LABDA2,2MN2,S2,K2,MSN2,2SM2,SKM2,NO3,2MK3,2MP3,SO3,MK3,SK3,4MS4,2MNS4,3MS4,MN4,2MLS4,2MSK4,M4,3MN4,MS4,MK4,2MSN4,S4,MNO5,3MK5,2MP5,3MO5,MSK5,3KM5,3MNS6,2NM6,4MS6,2MN6,2MNU6,3MSK6,M6,MSN6,MKNU6,2MS6,2MK6,3MSN6,2SM6,MSK6,2MNO7,M7,2MSO7,2(MN)8,3MN8,M8,2MSN8,2MNK8,3MS8,3MK8,2(MS)8,2MSK8,3MNK9,4MK9,3MSK9,4MN10,M10,3MSN10,4MS10,2(MS)N10,3M2S10,4MSK11,M12,4MSN12,5MS12,4M2S12]

Componentenset voor een maand ('month'), H0 en 21 componenten:

[H0, Q1, O1, K1, 3MS2, MNS2, MU2, N2, M2, 2MN2, S2, 2SM2, 3MS4, MN4, M4, MS4, 2MN6, M6, 2MS6, M8, 3MS8, 4MS10]

Componentenset voor een spring- doortijdcyclus ('springneap'), H0 en 14 componenten:

[H0, O1, K1, MU2, M2, S2, 2SM2, 3MS4, M4, MS4, M6, 2MS6, M8, 3MS8, 4MS10]

Componentenset voor een dag ('day'), H0 en 10 componenten:

[H0, M1, M2, M3, M4, M5, M6, M7, M8, M10, M12]

Componentenset voor een enkele getijcyclus ('tidalcycle'), H0 en 6 componenten:

[H0, M2, M4, M6, M8, M10, M12]

B Hatyan componenten en kengetallen

Alle binnen hatyan beschikbare componenten en hun kengetallen zijn weergegeven in Tabel B.1. H0 is het gemiddelde van de geanalyseerde tijdreeks, dit is dus in feite geen component omdat het niet wordt meegenomen in de harmonische analyse en de predictie, maar apart wordt berekend respectievelijk toegevoegd.

Tabel B.1 hatyan componenten en kengetallen

Component	Frequency [# per hour]	Angular velocity [rad/hr]	Period [hr]	Angular frequency [deg/hr]
H0	0.000000	0.0000	inf	0.0000
SA	0.000114	0.0007	8765.8128	0.0411
SSA	0.000228	0.0014	4382.9064	0.0821
MSM	0.001310	0.0082	763.4865	0.4715
MM	0.001512	0.0095	661.3092	0.5444
SM	0.002822	0.0177	354.3671	1.0159
MSF	0.002822	0.0177	354.3671	1.0159
MF	0.003050	0.0192	327.8590	1.0980
SNU2	0.004132	0.0260	242.0303	1.4874
SN	0.004334	0.0272	230.7292	1.5603
MFM	0.004562	0.0287	219.1904	1.6424
2SM	0.005644	0.0355	177.1835	2.0318
2SMN	0.007156	0.0450	139.7425	2.5762
NJ1	0.035706	0.2243	28.0062	12.8543
2Q1	0.035706	0.2243	28.0062	12.8543
NUJ1	0.035909	0.2256	27.8484	12.9271
SIGMA1	0.035909	0.2256	27.8484	12.9271
Q1	0.037219	0.2339	26.8684	13.3987
NUK1	0.037421	0.2351	26.7231	13.4715
RO1	0.037421	0.2351	26.7231	13.4715
O1	0.038731	0.2434	25.8193	13.9430
TAU1	0.038959	0.2448	25.6681	14.0252
MP1	0.038959	0.2448	25.6681	14.0252
M1B	0.040243	0.2529	24.8492	14.4874
M1C	0.040256	0.2529	24.8412	14.4921
M1D	0.040256	0.2529	24.8412	14.4921
M1A	0.040269	0.2530	24.8332	14.4967
M1	0.040269	0.2530	24.8332	14.4967
NO1	0.040269	0.2530	24.8332	14.4967
CHI1	0.040471	0.2543	24.7091	14.5695
LP1	0.040471	0.2543	24.7091	14.5695
TK1	0.041439	0.2604	24.1321	14.9179
PI1	0.041439	0.2604	24.1321	14.9179
P1	0.041553	0.2611	24.0659	14.9589
S1	0.041667	0.2618	24.0000	15.0000

K1	0.041781	0.2625	23.9345	15.0411
PSI1	0.041895	0.2632	23.8693	15.0821
RP1	0.041895	0.2632	23.8693	15.0821
FI1	0.042009	0.2639	23.8045	15.1232
KP1	0.042009	0.2639	23.8045	15.1232
THETA1	0.043091	0.2707	23.2070	15.5126
LABDAO1	0.043091	0.2707	23.2070	15.5126
J1	0.043293	0.2720	23.0985	15.5854
2PO1	0.044375	0.2788	22.5355	15.9748
SO1	0.044603	0.2802	22.4202	16.0570
OO1	0.044831	0.2817	22.3061	16.1391
KQ1	0.046343	0.2912	21.5782	16.6835
3MKS2	0.074639	0.4690	13.3978	26.8702
3MS2	0.074868	0.4704	13.3569	26.9523
OQ2	0.075949	0.4772	13.1667	27.3417
MNK2	0.075949	0.4772	13.1667	27.3417
MNS2	0.076177	0.4786	13.1273	27.4238
2ML2S2	0.076380	0.4799	13.0925	27.4967
2MS2K2	0.077233	0.4853	12.9478	27.8039
NLK2	0.077461	0.4867	12.9097	27.8861
2N2	0.077487	0.4869	12.9054	27.8954
MU2	0.077689	0.4881	12.8718	27.9682
2MS2	0.077689	0.4881	12.8718	27.9682
SNK2	0.078771	0.4949	12.6950	28.3576
N2	0.078999	0.4964	12.6583	28.4397
NU2	0.079202	0.4976	12.6260	28.5126
2KN2S2	0.079456	0.4992	12.5857	28.6040
OP2	0.080283	0.5044	12.4559	28.9020
MSK2	0.080283	0.5044	12.4559	28.9020
MPS2	0.080397	0.5052	12.4382	28.9430
M2	0.080511	0.5059	12.4206	28.9841
MSP2	0.080625	0.5066	12.4030	29.0252
MKS2	0.080740	0.5073	12.3855	29.0662
M2(KS)2	0.080968	0.5087	12.3506	29.1484
2SN(MK)2	0.081593	0.5127	12.2559	29.3735
LABDA2	0.081821	0.5141	12.2218	29.4556
L2A	0.082024	0.5154	12.1916	29.5285
L2	0.082024	0.5154	12.1916	29.5285
2MN2	0.082024	0.5154	12.1916	29.5285
L2B	0.082049	0.5155	12.1878	29.5378
2SK2	0.083105	0.5222	12.0329	29.9179
T2	0.083219	0.5229	12.0164	29.9589
S2	0.083333	0.5236	12.0000	30.0000
R2	0.083447	0.5243	11.9836	30.0411
K2	0.083561	0.5250	11.9672	30.0821
MSN2	0.084845	0.5331	11.7861	30.5444

ETA2	0.085074	0.5345	11.7545	30.6265
KJ2	0.085074	0.5345	11.7545	30.6265
MKN2	0.085074	0.5345	11.7545	30.6265
2KM(SN)2	0.085302	0.5360	11.7231	30.7086
2SM2	0.086155	0.5413	11.6070	31.0159
SKM2	0.086383	0.5428	11.5763	31.0980
2SNU2	0.087465	0.5496	11.4331	31.4874
3(SM)N2	0.087465	0.5496	11.4331	31.4874
2SN2	0.087667	0.5508	11.4067	31.5603
SKN2	0.087896	0.5523	11.3771	31.6424
MQ3	0.117730	0.7397	8.4940	42.3828
NO3	0.117730	0.7397	8.4940	42.3828
MO3	0.119242	0.7492	8.3863	42.9271
2MK3	0.119242	0.7492	8.3863	42.9271
2MP3	0.119470	0.7507	8.3703	43.0093
M3	0.120767	0.7588	8.2804	43.4762
SO3	0.122064	0.7670	8.1924	43.9430
MK3	0.122292	0.7684	8.1771	44.0252
2MQ3	0.123804	0.7779	8.0773	44.5695
SP3	0.124886	0.7847	8.0073	44.9589
SK3	0.125114	0.7861	7.9927	45.0411
K3	0.125342	0.7875	7.9782	45.1232
2SO3	0.127936	0.8038	7.8164	46.0570
4MS4	0.155379	0.9763	6.4359	55.9364
2MNS4	0.156689	0.9845	6.3821	56.4079
3MK4	0.157973	0.9926	6.3302	56.8702
MNLK4	0.157973	0.9926	6.3302	56.8702
3MS4	0.158201	0.9940	6.3211	56.9523
MSNK4	0.159282	1.0008	6.2782	57.3417
MN4	0.159511	1.0022	6.2692	57.4238
2MLS4	0.159713	1.0035	6.2612	57.4967
2MSK4	0.160795	1.0103	6.2191	57.8861
M4	0.161023	1.0117	6.2103	57.9682
2MKS4	0.161251	1.0132	6.2015	58.0503
SN4	0.162333	1.0200	6.1602	58.4397
3MN4	0.162535	1.0212	6.1525	58.5126
2SMK4	0.163617	1.0280	6.1119	58.9020
MS4	0.163845	1.0295	6.1033	58.9841
MK4	0.164073	1.0309	6.0949	59.0662
2SNM4	0.165155	1.0377	6.0549	59.4556
SL4	0.165357	1.0390	6.0475	59.5285
2MSN4	0.165357	1.0390	6.0475	59.5285
S4	0.166667	1.0472	6.0000	60.0000
SK4	0.166895	1.0486	5.9918	60.0821
2SMN4	0.168179	1.0567	5.9461	60.5444
3SM4	0.169489	1.0649	5.9001	61.0159

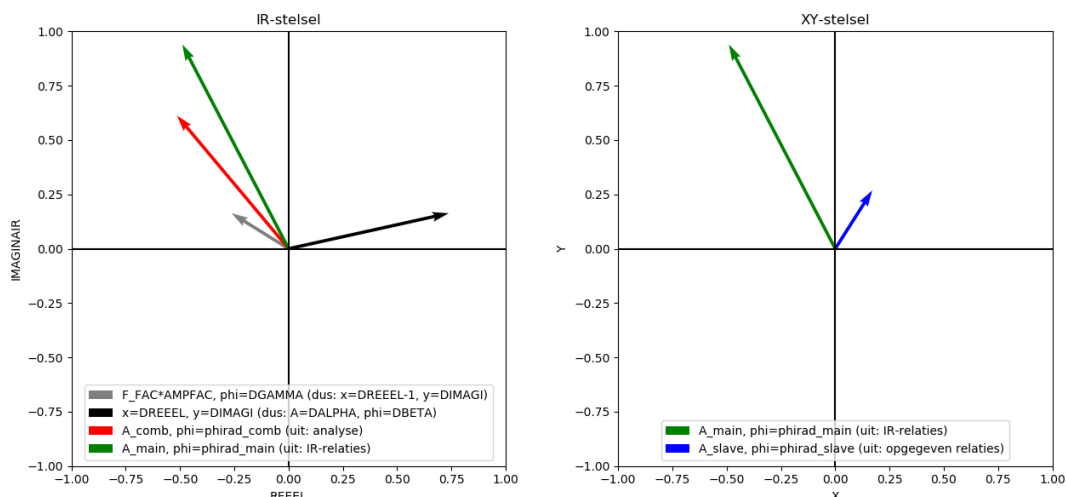
2SKM4	0.169717	1.0664	5.8922	61.0980
MNO5	0.198241	1.2456	5.0444	71.3669
3MK5	0.199753	1.2551	5.0062	71.9112
3MP5	0.199982	1.2565	5.0005	71.9934
M5	0.201291	1.2648	4.9679	72.4649
MNK5	0.201291	1.2648	4.9679	72.4649
2MP5	0.202575	1.2728	4.9364	72.9271
3MO5	0.202804	1.2743	4.9309	73.0093
MSK5	0.205625	1.2920	4.8632	74.0252
3KM5	0.205854	1.2934	4.8578	74.1073
2(MN)S6	0.235688	1.4809	4.2429	84.8477
3MNS6	0.237200	1.4904	4.2158	85.3920
2NM6	0.238510	1.4986	4.1927	85.8636
4MS6	0.238712	1.4999	4.1891	85.9364
2MSNK6	0.239794	1.5067	4.1702	86.3258
2MN6	0.240022	1.5081	4.1663	86.4079
2MNU6	0.240224	1.5094	4.1628	86.4808
3MSK6	0.241306	1.5162	4.1441	86.8702
M6	0.241534	1.5176	4.1402	86.9523
MSN6	0.242844	1.5258	4.1179	87.4238
4MN6	0.243046	1.5271	4.1144	87.4967
MNK6	0.243072	1.5273	4.1140	87.5060
MKNU6	0.243275	1.5285	4.1106	87.5788
2(MS)K6	0.244128	1.5339	4.0962	87.8861
2MS6	0.244356	1.5353	4.0924	87.9682
2MK6	0.244584	1.5368	4.0886	88.0503
2SN6	0.245666	1.5436	4.0706	88.4397
3MSN6	0.245868	1.5448	4.0672	88.5126
MKL6	0.246096	1.5463	4.0634	88.5947
2SM6	0.247178	1.5531	4.0457	88.9841
MSK6	0.247406	1.5545	4.0419	89.0662
S6	0.250000	1.5708	4.0000	90.0000
2MNO7	0.278753	1.7515	3.5874	100.3510
2NMK7	0.280291	1.7611	3.5677	100.9046
M7	0.281803	1.7706	3.5486	101.4490
2MSO7	0.283087	1.7787	3.5325	101.9112
MSKO7	0.286137	1.7979	3.4948	103.0093
2(MN)8	0.319021	2.0045	3.1346	114.8477
3MN8	0.320533	2.0140	3.1198	115.3920
3MNKS8	0.320762	2.0154	3.1176	115.4742
M8	0.322046	2.0235	3.1052	115.9364
2MSN8	0.323355	2.0317	3.0926	116.4079
2MNK8	0.323584	2.0331	3.0904	116.4901
3MS8	0.324868	2.0412	3.0782	116.9523
3MK8	0.325096	2.0426	3.0760	117.0344
2SNM8	0.326177	2.0494	3.0658	117.4238

MSNK8	0.326405	2.0509	3.0637	117.5060
2(MS)8	0.327689	2.0589	3.0517	117.9682
2MSK8	0.327918	2.0604	3.0495	118.0503
3SM8	0.330511	2.0767	3.0256	118.9841
2SMK8	0.330740	2.0781	3.0235	119.0662
S8	0.333333	2.0944	3.0000	120.0000
2(MN)K9	0.360802	2.2670	2.7716	129.8887
3MNK9	0.362314	2.2765	2.7600	130.4331
4MK9	0.363826	2.2860	2.7486	130.9775
3MSK9	0.366648	2.3037	2.7274	131.9934
4MN10	0.401045	2.5198	2.4935	144.3761
M10	0.402557	2.5293	2.4841	144.9205
3MSN10	0.403867	2.5376	2.4761	145.3920
4MS10	0.405379	2.5471	2.4668	145.9364
2(MS)N10	0.406689	2.5553	2.4589	146.4079
2MNSK10	0.406917	2.5567	2.4575	146.4901
3M2S10	0.408201	2.5648	2.4498	146.9523
4MSK11	0.447160	2.8096	2.2363	160.9775
M12	0.483068	3.0352	2.0701	173.9046
4MSN12	0.484378	3.0434	2.0645	174.3761
5MS12	0.485890	3.0529	2.0581	174.9205
3MNKS12	0.487428	3.0626	2.0516	175.4742
4M2S12	0.488712	3.0707	2.0462	175.9364

C Methode componentensplitsing

Invoer:

- A_comb = 0.8
- phirad_comb = deg2rad(130)
- AMPFAC = 0.3
- RADINCR = deg2rad(-60)



Figuur C.1 Componentensplitsing

Bij het splitsen van componenten worden de volgende stappen doorlopen:

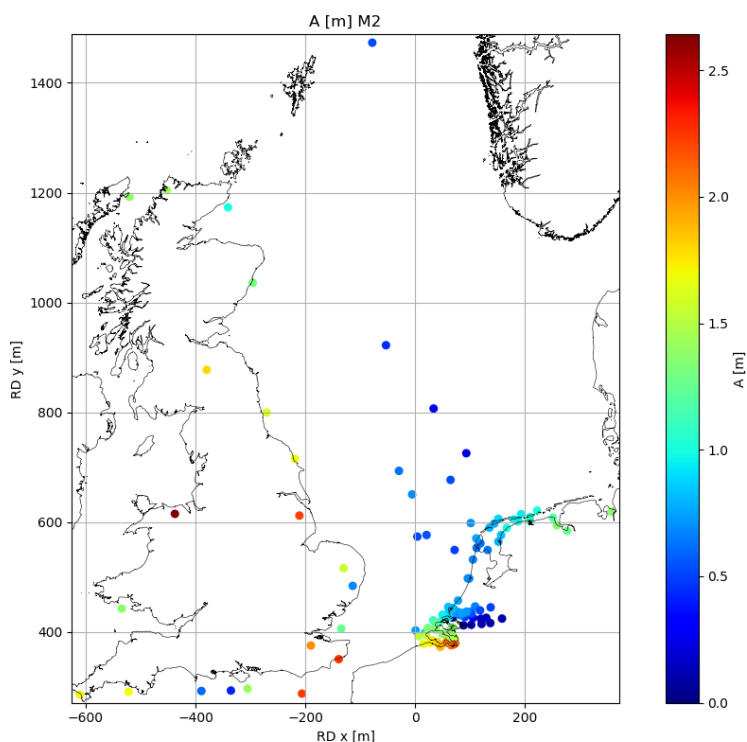
1. Afleiden van de relaties die gelden tussen vectoren opgebouwd in de reële en imaginaire ruimte. Hierin volgt de grijze vector uit relaties tussen $v/u/f$ (v en knooppactoren) en de opgegeven relaties voor amplitude en hoek tussen het eerste en tweede component. De grijze vector:
 - A: $F_FAC \cdot AMPFAC = (f_slave/f_main) \cdot AMPFAC$
 - \emptyset : $DGAMMA = RADINCR - 0 - (v_slave + u_slave) + (v_main + u_main)$
 - X: $X_grijs = F_FAC \cdot AMPFAC \cdot \cos(DGAMMA)$
 - Y: $Y_grijs = F_FAC \cdot AMPFAC \cdot \sin(DGAMMA)$
2. Er wordt 1 opgeteld bij de x-component van deze grijze vector, om tot de zwarte vector te komen. Van de zwarte vector wordt in het IR-stelsel geprojecteerd en met goniometrie worden de amplitude en hoek bepaald:
 - X: $DREEEL = 1 + F_FAC \cdot AMPFAC \cdot \cos(DGAMMA)$ (is plus 1 ten opzichte van grijze vector)
 - Y: $DIMAGI = F_FAC \cdot AMPFAC \cdot \sin(DGAMMA)$ (overgenomen uit grijze vector)
 - A: $DALPHA = \sqrt{DREEEL^2 + DIMAGI^2}$
 - \emptyset : $DBETA = \arctan2(DIMAGI, DREEEL)$
3. De ongesplitste (gecombineerde) component uit de analyse vormt de rode vector:
 - A: A_comb
 - \emptyset : phirad_comb
4. Met behulp van relaties die gelden in het IR-stelsel wordt de eerste component afgeleid, de groene vector:
 - A: $A_main = A_comb / DALPHA$
 - \emptyset : $phirad_main = (phirad_comb - DBETA) \% (2 \cdot \pi)$
5. Met behulp van de opgegeven relaties tussen de eerste en tweede component wordt de tweede component (de blauwe vector) afgeleid:
 - A: $A_slave = A_main \cdot AMPFAC$
 - \emptyset : $phirad_slave = (phirad_main + RADINCR) \% (2 \cdot \pi)$

Deze methode is geldig voor het enkelvoudig splitsen van componenten (bijvoorbeeld $K1^*$ in $K1$ en $P1$, of $N2^*$ in $N2$ en $NU2$), een splitsing in een eerste en tweede component. Bij het dubbelvoudig splitsen van componenten (bijvoorbeeld $S2^*$ in $S2$, $K2$ en $T2$, dus het splitsen in een eerste, tweede en derde component) worden de stappen 1 tot en met 4 nogmaals doorlopen voor het derde component. Om $DGAMMA2$ te berekenen wordt dan in plaats van 0, $DBETA$ afgetrokken.

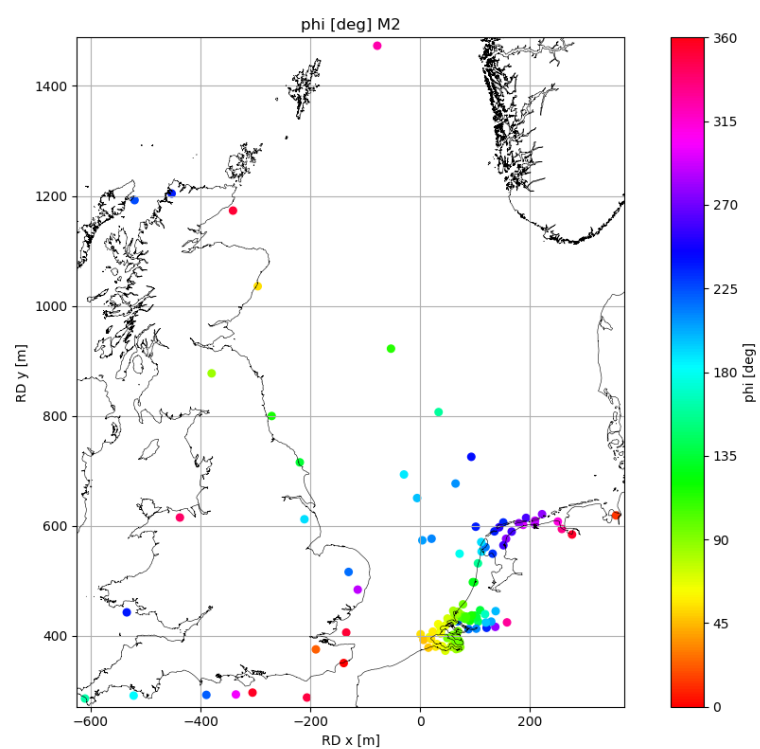
D Ruimtelijke figuren

Op dit moment is het mogelijk om alle informatie uit componentenfile ruimtelijk te visualiseren. Dit zijn de amplitude en fase per component (voor M2 in Figuur D.1 en Figuur D.2), de middenstand H0 (Figuur D.3), evenals het verschil in de amplitude van verschillende componenten om bijvoorbeeld een factor voor componentensplitsing af te leiden (factor P1 en K1 in Figuur D.4). Dit geeft een beeld van de ruimtelijke variatie van deze waarden.

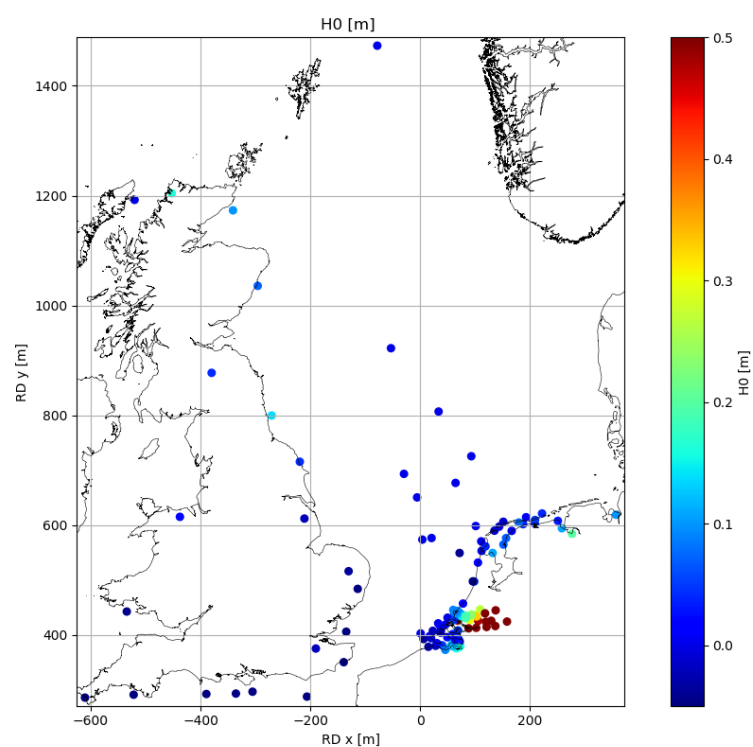
Deze waarden kunnen worden afgeleid voor een door de gebruiker aangeleverde verzameling componentenfiles. Het kan hierbij dus voorkomen dat de analyses voor verschillende stations gedaan zijn met zeer verschillende analyseperiodes of instellingen als knoofactoren. Het is van belang om hiervan bewust te zijn.



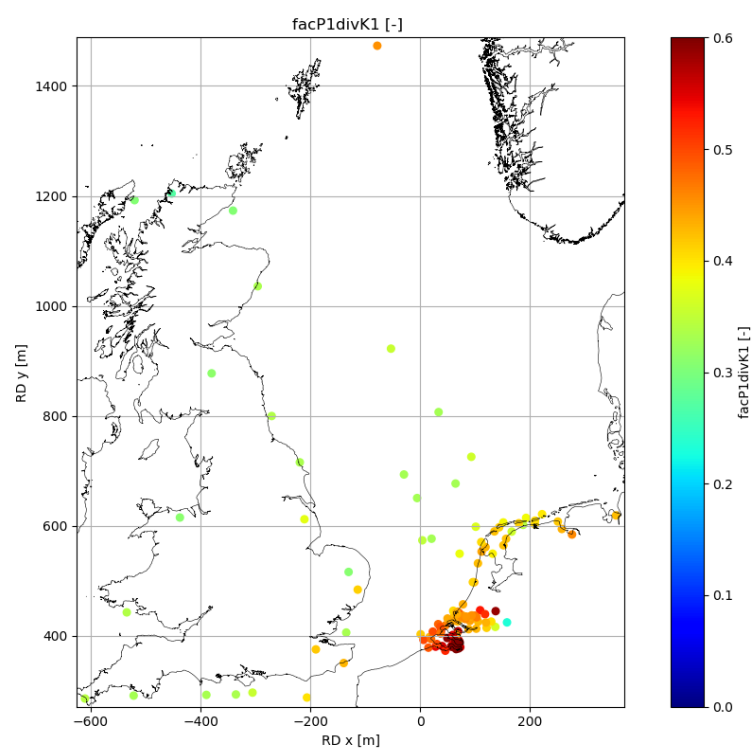
Figuur D.1 ruimtelijke variatie van de amplitude van M2



Figuur D.2 ruimtelijke variatie van de fase van M2



Figuur D.3 ruimtelijke variatie van amplitude H0 (gemiddelde/middenstand)



Figuur D.4 ruimtelijke variatie van de factor tussen de amplitudes van de componenten P1 en K1

E Builden en installatie hatyan RPM

Deze bijlage bevat informatie voor het inpakken van de broncode in een RPM voor het gebruik op CentOS, het systeem dat op de getijserver bij RWS geïnstalleerd staat. Voor het gebruik van hatyan is deze werkwijze echter niet essentieel, de algemene installatieprocedure voor hatyan staat beschreven in Bijlage F.

E.1 Builden hatyan RPM

Om de hatyan RPM te maken is een vergelijkbaar Linux systeem nodig als het systeem waarop het geïnstalleerd wordt. Bij Rijkswaterstaat is een RedHat 6.8 aanwezig. Bij Deltares is een CentOS 6.9 besturingssysteem geïnstalleerd op een virtuele machine, deze VM is verder zo kaal mogelijk gehouden, om zeker te weten dat alle benodigde software bewust wordt geïnstalleerd tijdens het installatieproces en dus ook als dependency opgegeven wordt.

E.1.1 Opzetten Virtuele Machine

Allereerst moet er een Virtuele Machine (VM) opgezet worden met CentOS:

- installeer VirtualBox van <https://www.virtualbox.org/>
- Openen VM door het vbox-bestand op de lib_tide repository te openen:
[https://repos.deltares.nl/repos/lib_tide/trunk/src/hatyan_python/RWS/VM/CentOS 6 hatyan/CentOS 6 hatyan.vbox](https://repos.deltares.nl/repos/lib_tide/trunk/src/hatyan_python/RWS/VM/CentOS%206%20hatyan/CentOS%206%20hatyan.vbox)
- De instellingen hiervan zouden al correct moeten zijn, maar moeten bijgewerkt worden indien de lokale lib_tide checkout op een andere locatie staat dan bovenstaand (kopje 'Shared folders'):
 - o Folder path: e.g. C:\DATA\hatyan (de lokale locatie van de map https://repos.deltares.nl/repos/lib_tide/trunk/src/hatyan_python)
 - o folder name: hatyan
 - o auto mount: yes
 - o er is nu een link naar deze lokale map in de VM: /media/sf_hatyan_python
- Start VM en wacht tot de voortgangsbalk voltooid is, vanaf dat moment is de VM benaderbaar en hoeft er in dit scherm niets meer gedaan te worden. Het kan nog iets langer duren voordat de netwerkverbinding open is en de VM echt benaderbaar is, maar er volgt hiervan geen visuele feedback.
- Benader de VM met het programma MobaXterm (dit X11 forwarding possibility, can be downloaded from <https://mobaxterm.mobatek.net/download.html>):
 - o New SSH session
 - o Remote host: 192.168.56.101 (check in VM screen by hovering over network icon, IP-address of host-only adapter)
 - o Specify username: veenstra
 - o Port: 22
 - o password: veenstrapw

E.1.2 RPM builden

Eventueel kan voorafgaand aan het builden van de RPM het versienummer opgehoogd worden en de html en pdf documentatie bijgewerkt (zie Bijlage F). Vervolgens kan de RPM met hatyan worden gebuild. De RPM bevat het Python virtual environment ten behoeve van hatyan, samen met de scripts, data en configuratiefiles. Hierbij wordt een sh file aangeroepen met enkele commando's. Kortweg wordt hiermee:

- Python geïnstalleerd (rh-python36)
- een 'relocatable virtual environment' gecreëerd met alle benodigde Python libraries
- de huidige versie van hatyan geïnstalleerd in dit environment, dit deel kan later bijgewerkt worden (Hoofdstuk E.2.5).
- Een commando 'hatyan' ingesteld dat op Linux het volgende doet:

- Instellen van Qt-omgevingsvariabelen
 - Activeren van virtuele Python environment
 - Creëren van uitvoerfolder en diagnostische file
 - Runnen van opgegeven hatyan configuratiefile
 - Al het bovenstaande ingepakt in een RPM-file
- De bovenstaande handelingen kan de gebruiker starten door het volgende script uit te voeren
`/media/sf_hatyan_python/RWS/hatyan_python_commands.sh`

E.2 Installatie hatyan RPM

Het getijde analyse- en predictieprogramma hatyan wordt als RPM geleverd om bij RWS op een RedHat 6.9 Linux systeem geïnstalleerd te kunnen worden.

E.2.1 Dependencies hatyan installatie

Voor deze RPM moeten op het CentOS/Redhat 6 systeem de volgende packages zijn of worden geïnstalleerd, opgegeven in de dependencies van de RPM. Hiervan zijn veel nodig voor het mogelijk maken van (interactieve) plots:

- rh-python36-python >= 3.6.3
- rh-python36-python-libs >= 3.6.3
- glibc >= 2.12 (2.12 is de hoogst beschikbare versie op CentOS/RedHat 6)
- coreutils expect
- stix-fonts fontconfig freetype
- libstdc++ jasper
- libXcursor libXrender xorg-x11-xauth
- mesa-libGL mesa-libEGL libXi

Op het moment wordt gebruik gemaakt van de QT5 backend voor de interactieve plots, het is mogelijk dat in de toekomst wordt overgestapt naar een GTK-gebaseerde backend waardoor de dependencies van de RPM's kunnen veranderen.

E.2.2 Inhoud RPM

De RPM bevat een virtuele Python omgeving met onder andere de volgende Python libraries, welke via pip zijn geïnstalleerd in de RPM en daardoor zonder internet kunnen worden uitgerold bij RWS:

- matplotlib==3.2.1 (genereren en opslaan van figuren)
- pandas==1.0.3 (versnellen van acties op tijdreeksen)
- numpy==1.18.4 en scipy==1.4.1 (aanmaken en handelingen uitvoeren op arrays)
- PyQt5==5.7.1 en sip==4.19.8 (voor interactieve plots met QT5. Deze versies zijn bewust niet de meest recente, omdat nieuwere versies minimaal glibc 2.14 vereisen, dit is niet beschikbaar voor CentOS/RedHat 6)
- pyproj==1.9.6 (coördinaattransformatie)
- netCDF4==1.5.3 (inlezen en wegschrijven van netCDF files)
- hatyan

Het installeren van de RPM zal de volgende mappen/bestanden aanmaken op het systeem:

- `/opt/hatyan_python/env/` (map met python environment)
- `/usr/bin/hatyan` (de file die het runnen van hatyan mogelijk maakt)

E.2.3 Installatie van hatyan

Het installeren van de RPM's op RedHat 6 kan met yum, de dependencies worden dan automatisch geïnstalleerd uit de Red Hat Software Collection. Voor CentOS 6 moet er eerst een vergelijkbare repository worden gekoppeld om de juiste Python versie te verkrijgen (bijvoorbeeld `centos-release-scl` op CentOS6), dit is niet nodig op RedHat:

```
sudo yum -y install centos-release-scl-rh
```

De RPM kan geïnstalleerd worden met het volgende commando, verwijzend naar de (eventueel lokale) locatie van de RPM:

```
sudo yum -y install /media/sf_hatyan_python/RWS/hatyan_python-2.2.22.x86_64.rpm
```

Het updaten met een nieuwe RPM kan met hetzelfde commando, yum werkt dan de installatie bij. Het verwijderen van hatyan op RedHat 6 kan met yum:

```
sudo yum remove hatyan
```

E.2.4 Testen van hatyan installatie

Het succesvol doorlopen van het installatieproces toont aan dat alle gevraagde linux bibliotheken die door hatyan gevraagd worden beschikbaar zijn. Dit kan snel getest worden met het commando:

```
hatyan
```

Er verschijnt dan de foutmelding dat er geen configuratiefile is opgegeven, samen met de huidige versie van hatyan.

Functioneel kan hatyan getest worden door de applicatie te laten draaien met een configuratiefile die een uitgebreid werkproces doorloopt (inlezen van files, wegschrijven van files en weergeven van interactieve plots). Als er een link is gelegd met de lokale computer (host), kan iedere configuratiefile worden aangeroepen die hier staat, bijvoorbeeld:

```
hatyan  
/media/sf_hatyan_python/tests/configfiles/predictie_2019_19Ycomp4Ydia_VLI  
SSGN_test_interactive.py
```

Dit zou moeten resulteren in vijf interactieve figuren die het scherm. Als de waarschuwing 'RuntimeError: Invalid DISPLAY variable' verschijnt, start dan de MobaXterm connectie met het Linux systeem opnieuw en probeer het nog een keer.

De volgende waarschuwing kan altijd worden genegeerd: 'QXcbConnection: XCB error: 145 (Unknown), sequence: 171, resource id: 0, major code: 139 (Unknown), minor code: 20'. Om deze melding te voorkomen kan bij MobaXterm bij Settings > Configuration > X11 de extension RANDR uitgeschakeld worden.

Zie Hoofdstuk 5.5 voor het inhoudelijk beoordelen van de geproduceerde testresultaten.

E.2.5 Updaten van hatyan source code

Vanaf hatyan_python-2.2.22 is het mogelijk om updates van de hatyan source code binnen de bestaande installatie die via RPM is uitgevoerd. De benodigde omgeving wordt intact gelaten en slechts de library hatyan wordt bijgewerkt. Op dit moment is het mogelijk om dit te doen door middel van zipfile met de source code, bijvoorbeeld:

```
sudo /opt/hatyan_python/env/bin/python -m pip install  
/media/sf_hatyan_python/RWS/SIG/hatyan-20201028.zip
```

In de toekomst wordt hatyan eventueel ontsloten via PyPI of github, dan zijn andere commando's van toepassing. Het blijft ook mogelijk hatyan door middel van een RPM te updaten.

F Automatisch gegenereerde documentatie vanuit docstrings

De documentatie van de functies van hatyan voor Python is opgeslagen in de broncode zelf. Van deze 'docstrings' wordt vervolgens met enkele commando's de documentatie in html en pdf formaat gegenereerd, waarvoor de benodigde stappen in ditzelfde document zijn opgeslagen onder de kop 'Information for developers'. Op deze manier kan de documentatie centraal beheerd worden (in de scripts) en is deze altijd bijgewerkt bij oplevering van een nieuwe versie van hatyan. Deze gebruikershandleiding hoeft dan niet per definitie bij iedere oplevering aangepast te worden. De volgende pagina's bevatten deze documentatie ten tijde van oplevering van deze handleiding. Nog niet alle functies zijn gedocumenteerd, dit is te herkennen aan de tekst in hoofdletters 'TYPE' en 'DESCRIPTION'. Enkele functies zijn niet gedocumenteerd, dit is dan ook duidelijk aangegeven.

hatyan-2.2.23 documentation, automatically generated
from script comments and function docstrings

Contents

Module <code>hatyan</code>	1
Installation	1
Example Usage	2
Information For Developers	3
Sub-modules	4
Module <code>hatyan.analysis_prediction</code>	6
Functions	6
Function <code>vectoravg</code>	6
Function <code>get_components_from_ts</code>	7
Function <code>analysis</code>	8
Function <code>split_components</code>	8
Function <code>prediction</code>	8
Module <code>hatyan.components</code>	10
Functions	10
Function <code>plot_components</code>	10
Function <code>write_components</code>	11
Function <code>merge_componentgroups</code>	11
Function <code>read_components</code>	11
Module <code>hatyan.hatyan_core</code>	13
Functions	13
Function <code>get_const_list_hatyan</code>	13
Function <code>get_doodson_eqvals</code>	14
Function <code>get_hatyan_constants</code>	14
Parameters	14
Function <code>get_allconstituents_withdoodson</code>	15
Parameters	15
Function <code>get_hatyan_freqs</code>	15
Function <code>get_hatyan_v</code>	15
Function <code>get_hatyan_u</code>	16
Function <code>get_hatyan_f</code>	16
Module <code>hatyan.timeseries</code>	17
Functions	17
Function <code>calc_HWLW</code>	17
Function <code>number_HWLW</code>	18

Parameters	18
Function <code>plot_timeseries</code>	18
Function <code>write_tsnetcdf</code>	19
Function <code>write_tsdia</code>	19
Function <code>write_tsdia_HWLW</code>	20
Function <code>crop_timeseries</code>	20
Function <code>resample_timeseries</code>	21
Function <code>check_ts</code>	21
Function <code>get_diablocks</code>	21
Function <code>getcheck_diameta</code>	22
Function <code>readts_dia_HWLW</code>	22
Function <code>readts_dia</code>	23
Function <code>readts_noos</code>	23
Module <code>hatyan.wrapper_RWS</code>	24
Functions	24
Function <code>init_RWS</code>	24
Function <code>exit_RWS</code>	25
Function <code>get_outputfoldername</code>	25

Module hatyan

hatyan is a Python program for tidal analysis and prediction, based on the FORTRAN version. Copyright (C) 2019-2020 Rijkswaterstaat. Maintained by Deltares, contact: Jelmer Veenstra (jelmer.veenstra@deltares.nl). Source code available at: https://repos.deltares.nl/repos/lib_tide/trunk/src/hatyan_python/hatyan

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 3 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, see <https://www.gnu.org/licenses/>.

Installation

Install Python (Anaconda):

- download Anaconda 64 bit Python 3.7 from <https://www.anaconda.com/distribution/#download-section> (miniconda should also be sufficient, but this is not yet tested)
- install it with the recommended settings, but check ‘add Anaconda3 to my PATH environment variable’ if you want to use conda from the windows command prompt instead of anaconda prompt

Install hatyan OPTION 1: no hatyan installation, use existing checkout (this example is only possible on the Deltares network):

- add to the top of your script `sys.path.append(r'n:\Deltabox\Bulletin\veenstra\hatyan_py`

Install hatyan OPTION 2: Create a separate python environment `hatyan_env` and install hatyan from a zipfile with the source code and `setup.py`:

- open command line window
- `conda env create -f environment.yml` (sometimes you need to press enter if it hangs extremely long)
- `conda info --envs` (should show `hatyan_env` virtual environment in the list)
- `conda activate hatyan_env`
- `python -m pip install path_to_zipfile.zip`

- `conda deactivate`
- to remove `hatyan_env` when necessary: `conda remove -n hatyan_env --all`
- to update hatyan, install a new zip file

Install hatyan OPTION 3: install from github or even PyPI (not yet possible, this is just an example):

- open command window (or anaconda prompt)
- `conda create --name hatyan_env -c conda-forge python=3.7 git spyder -y`
- `conda activate hatyan_env`
- `python -m pip install git+https://github.com/openearth/hatyan.git`
(this command installs hatyan and all required packages)
- to remove venv when necessary: `conda remove -n hatyan_env --all`
- to update hatyan: `python -m pip install --upgrade git+https://github.com/openearth/`

Install hatyan OPTION 4 (for CentOS only): get and install RPM (see hatyan user manual for details)

- installing the RPM results in a hatyan command in linux, calling a Python virtual environment and setting some variables
- from hatyan-2.2.22 it is possible to update the source code in this installation with a zipfile (similar to OPTION 2).

Launch Spyder after installation (in case of option 2 or 3):

- open 'Spyder(hatyan_env)' via your windows start menu (not 'Spyder' or 'Spyder(Anaconda3)', since hatyan was installed in `hatyan_env`)
- if launching Spyder gives a Qt related error: remove the system/user environment variable 'qt_plugin_path' set by an old Delft3D4 installation procedure
- to get figures in separate windows: go to Tools > Preferences > IPython console > Graphics > change graphics backend to 'Automatic' and restart Spyder (or the kernel).
- copy the code from [Example usage](#) to your own scripts to get started

Example Usage

```
import os, sys
sys.path.append(r'n:\Deltabox\Bulletin\veenstra\hatyan_python')
import datetime as dt
```

```
from hatyan import timeseries as Timeseries
from hatyan import components as Components
from hatyan.analysis_prediction import get_components_from_ts, prediction
from hatyan.hatyan_core import get_const_list_hatyan
```

```
#defining a list of the components to be analysed (can also be 'half_year' and other)
const_list = get_const_list_hatyan('year')
```

```
#reading and editing time series, results in a pandas dataframe with columns 'times
```

```

file_data_meas = os.path.join(r'n:\Deltabox\Bulletin\veenstra\VLISSGN_waterlevel_2010
times_ext = [dt.datetime(2012,1,1),dt.datetime(2013,1,1)]
timestep_min = 10
ts_meas = Timeseries.readts_noos(filename=file_data_meas)
ts_meas = Timeseries.resample_timeseries(ts_meas, timestep_min=timestep_min)
ts_meas = Timeseries.crop_timeseries(ts=ts_meas, times_ext=times_ext)

#tidal analysis and plotting of results. commented: saving figure
comp_frommeas = get_components_from_ts(ts=ts_meas, const_list=const_list, nodalfactor
fig,(ax1,ax2) = Components.plot_components(comp=comp_frommeas)
#fig.savefig('components_%s_4Y.png'%(current_station))

#tidal prediction and plotting of results. commented: saving figure and writing to
ts_prediction = prediction(comp=comp_frommeas, nodalfactors=True, xfac=True, fu_allti
fig, (ax1,ax2) = Timeseries.plot_timeseries(ts=ts_prediction, ts_validation=ts_meas)
ax1.legend(['prediction','measurement','difference','mean of prediction'])
ax2.set_ylim(-0.5,0.5)
ax2.lines.pop(1)
#fig.savefig('prediction_%im_%s_measurements'%(timestep_pred, current_station))
#Timeseries.write_tsnetcdf(ts=ts_prediction, ts_ext=ts_ext_prediction, station=curr

#calculation of HWLW and plotting of results. commented: saving figure
ts_ext_prediction = Timeseries.calc_HWLW(ts=ts_prediction)
fig, (ax1,ax2) = Timeseries.plot_timeseries(ts=ts_prediction, ts_ext=ts_ext_prediction)
#fig.savefig('prediction_%im_%s_HWLW'%(timestep_pred, current_station))

```

Information For Developers

Set up local repository checkout:

- Checkout part of the lib_tide repository in the local folder: https://repos.deltares.nl/repos/lib_tide/trunk/src/hatyan_python

Install Python (Anaconda):

- download Anaconda 64 bit Python 3.7 from <https://www.anaconda.com/distribution/#download-section> (miniconda should also be sufficient, but this is not yet tested)
- install it with the recommended settings, but check ‘add Anaconda3 to my PATH environment variable’ if you want to use conda from the windows command prompt instead of anaconda prompt

Create a separate python environment hatyan_env and install hatyan as developer:

- open command line and navigate to hatyan local folder, e.g. C:\DATA\hatyan_python
- `conda env create -f environment.yml` (sometimes you need to press enter if it hangs extremely long)
- `conda info --envs` (should show hatyan_env virtual environment in the list)
- `conda activate hatyan_env`

- `python -m pip install -e . -r requirements_dev.txt` (pip developer mode, also install all packages in `requirements_dev.txt` containing CentOS tested libraries, linked via `setup.py`)
- `conda deactivate`
- to remove `hatyan_env` when necessary: `conda remove -n hatyan_env --all`

Increasing version number:

- open command line and navigate to hatyan local folder, e.g. `C:\DATA\hatyan_python`
- `conda activate hatyan_env`
- `bumpversion major` or `bumpversion minor` or `bumpversion patch`
- The hatyan version number in these files gets updated: `setup.py`, `hatyan/init.py`, `RWS/hatyan_commands.sh`, `RWS/hatyan-latest.spec`

Running the testbank:

- open command line and navigate to hatyan local folder, e.g. `C:\DATA\hatyan_python`
- `conda activate hatyan_env`
- `pytest -v --tb=short`
- `pytest -v --tb=short -m acceptance`
- `pytest -v --tb=short -m systemtest`
- `pytest -v --tb=short -m slow`
- `pytest -v --tb=short -m "not slow"` (exclude 'slow' testbank scripts for all stations)

Generating html documentation:

- open command line and navigate to hatyan local folder, e.g. `C:\DATA\hatyan_python`
- `conda activate hatyan_env`
- `pdoc --html hatyan -o doc --force --config sort_identifiers=False`

Generating pdf documentation:

- install miktex for windows
- update miktex packages from the miktex console
- open command line and navigate to hatyan local folder, e.g. `C:\DATA\hatyan_python`
- `conda activate hatyan_env`
- `FOR /F "tokens=*" %g IN ('python -c "import hatyan; print(hatyan.__version__)"')`
do `(SET hatyan_version=%g)`
- `pdoc --pdf hatyan --config sort_identifiers=False > doc/hatyan_functions.md`
- `pandoc --metadata=title:"hatyan-%hatyan_version% documentation, automatically generated from script comments and function docstrings"`
`--toc --toc-depth=4 --variable fontsize=12pt --from=markdown+abbreviations`
`--pdf-engine=xelatex --top-level-division=chapter --output=doc/hatyan_functions.`
`doc/hatyan_functions.md`
- The last two steps might take a while, since it also installs necessary latex packages via miktex

Sub-modules

- [hatyan.analysis_prediction](#)

- [hatyan.components](#)
- [hatyan.hatyan_core](#)
- [hatyan.timeseries](#)
- [hatyan.wrapper_RWS](#)

Module `hatyan.analysis_prediction`

`analysis_prediction.py` contains hatyan definitions related to tidal analysis and prediction.

hatyan is a Python program for tidal analysis and prediction, based on the FORTRAN version. Copyright (C) 2019-2020 Rijkswaterstaat. Maintained by Deltares, contact: Jelmer Veenstra (jelmer.veenstra@deltares.nl). Source code available at: https://repos.deltares.nl/repos/lib_tide/trunk/src/hatyan_python/hatyan

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 3 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, see <https://www.gnu.org/licenses/>.

Functions

Function `vectoravg`

```
def vectoravg(  
    A_all,  
    phi_deg_all  
)
```

calculates the vector average of A and phi per constituent, it vector averages over values resulting from multiple periods. A regular average is calculated for the amplitude of H0 (middenstand)

Parameters

`A_i_all` : TYPE DESCRIPTION.
`phi_i_deg_all` : TYPE DESCRIPTION.

Returns

`A_i_mean` : TYPE DESCRIPTION.
`phi_i_deg_mean` : TYPE DESCRIPTION.

Function get_components_from_ts

```
def get_components_from_ts(
    ts,
    const_list,
    nodalfactors=True,
    xfac=False,
    fu_alltimes=True,
    CS_comps=None,
    analysis_peryear=False,
    analysis_permonth=False,
    return_allyears=False
)
```

Wrapper around the analysis() function, it optionally processes a timeseries per year and vector averages the results afterwards, passes the rest of the arguments on to analysis function

Parameters

ts : **pandas.DataFrame** The DataFrame contains columns 'times' and 'values', it contains the timeseries to be analysed, as obtained from e.g. readts_*.

const_list : **list, pandas.Series or str** list or pandas.Series: contains the tidal constituent names for which to analyse the provided timeseries ts. str: a predefined name of a component set for hatyan_core.get_const_list_hatyan()

nodalfactors : **bool/int, optional** Whether or not to apply nodal factors. The default is True.

xfac : **bool/int, optional** Whether or not to apply x-factors. The default is False.

fu_alltimes : **bool/int, optional** determines whether to calculate nodal factors in middle of analysis period (default) or on every timestep. The default is True.

analysis_peryear : **bool/int, optional** DESCRIPTION. The default is False.

analysis_permonth : **bool/int, optional** caution, it tries to analyse each month, but skips it if it fails. analysis_peryear argument has priority. The default is False.

return_allyears : **bool/int, optional** DESCRIPTION. The default is False.

CS_comps : **pandas.DataFrame, optional** contains the main/slave component lists for components splitting, as well as the amplitude factor and the increase in degrees. The default is None.

Raises

Exception DESCRIPTION.

Returns

COMP_mean_pd : **pandas.DataFrame** The DataFrame contains the component data with component names as index, and columns 'A' and 'phi_deg'.

COMP_all_pd : **pandas.DataFrame, optional** The same as COMP_mean_pd, but with all years added with MultiIndex

Function analysis

```
def analysis(
    ts,
    const_list,
    nodalfactors=True,
    xfac=False,
    fu_alltimes=True,
    CS_comps=None
)
```

harmonic analysis with matrix transformations (least squares fit), optionally with component splitting for details about arguments and return variables, see `get_components_from_ts()` definition

Function split_components

```
def split_components(
    comp,
    CS_comps,
    dood_date_meas,
    xfac=False
)
```

component splitting function for details about arguments and return variables, see `get_components_from_ts()` definition

Function prediction

```
def prediction(
    comp,
    times_pred_all=None,
    times_ext=None,
    timestep_min=None,
    nodalfactors=True,
    xfac=False,
    fu_alltimes=True
)
```

generates a tidal prediction from a set of components A and phi values

Parameters

comp : **pandas.DataFrame** The DataFrame contains the component data with component names as index, and columns 'A' and 'phi_deg'.

times_pred_all : **pandas.DatetimeIndex, optional** Prediction timeseries. The default is None.

times_ext : **list of datetime.datetime, optional** Prediction time extents (list of start time and stop time). The default is None.

timestep_min : **int, optional** Prediction timestep in minutes. The default is None.

nodalfactors : **bool/int, optional** Whether or not to apply nodal factors. The default is True.
xfac : **bool/int, optional** Whether or not to apply x-factors. The default is False.
fu_alltimes : **bool/int, optional** determines whether to calculate nodal factors in middle of the prediction period (default) or on every timestep. The default is True.

Raises

Exception DESCRIPTION.

Returns

ts_prediction_pd : **pandas.DataFrame** The DataFrame contains columns ‘times’ and ‘values’, it contains the prediction times and values.

Module `hatyan.components`

`components.py` contains all the definitions related to `hatyan` components.

`hatyan` is a Python program for tidal analysis and prediction, based on the FORTRAN version. Copyright (C) 2019-2020 Rijkswaterstaat. Maintained by Deltares, contact: Jelmer Veenstra (jelmer.veenstra@deltares.nl). Source code available at: https://repos.deltares.nl/repos/lib_tide/trunk/src/hatyan_python/hatyan

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 3 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, see <https://www.gnu.org/licenses/>.

Functions

Function `plot_components`

```
def plot_components(  
    comp,  
    comp_allyears=None,  
    comp_validation=None  
)
```

Create a plot with the provided analysis results

Parameters

`comp` : TYPE DESCRIPTION.

`comp_allyears` : TYPE, optional DESCRIPTION. The default is None.

`comp_validation` : TYPE, optional DESCRIPTION. The default is None.

Returns

`fig` : `matplotlib.figure.Figure` The generated figure handle, with which the figure can be adapted and saved.

axs : (tuple of) `matplotlib.axes._subplots.AxesSubplot` The generated axis handle, with which the figure can be adapted.

Function `write_components`

```
def write_components(  
    comp,  
    filename,  
    metadata=None  
)
```

Writes the provided analysis results to a file

Parameters

comp : TYPE DESCRIPTION.

filename : TYPE DESCRIPTION.

metadata : TYPE, optional DESCRIPTION. The default is None.

Returns None.

Function `merge_componentgroups`

```
def merge_componentgroups(  
    comp_main,  
    comp_sec,  
    comp_sec_list=['SA', 'SM']  
)
```

Merges the provided component groups into one

Parameters

comp_main : TYPE DESCRIPTION.

comp_sec : TYPE DESCRIPTION.

comp_sec_list : TYPE, optional DESCRIPTION. The default is ['SA','SM'].

Raises

Exception DESCRIPTION.

Returns

COMP_merged : TYPE DESCRIPTION.

Function `read_components`

```
def read_components(  
    filename,  
    get_metadata=False  
)
```

Reads analysis results from a file.

Parameters

filename : **TYPE** DESCRIPTION.

get_metadata : **TYPE**, **optional** DESCRIPTION. The default is False.

Raises

Exception DESCRIPTION.

Returns

TYPE DESCRIPTION.

Module `hatyan.hatyan_core`

`hatyan_core.py` contains definitions with data for the hatyan constituents.

hatyan is a Python program for tidal analysis and prediction, based on the FORTRAN version. Copyright (C) 2019-2020 Rijkswaterstaat. Maintained by Deltares, contact: Jelmer Veenstra (jelmer.veenstra@deltares.nl). Source code available at: https://repos.deltares.nl/repos/lib_tide/trunk/src/hatyan_python/hatyan

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 3 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, see <https://www.gnu.org/licenses/>.

Functions

Function `get_const_list_hatyan`

```
def get_const_list_hatyan(  
    listtype  
)
```

Definition of several hatyan components lists. H0 was added to each list, this is not a tidal component but the mean of the timeseries. H0 is not considered in the harmonic analysis and prediction, but calculated resp. added afterwards automatically.

Parameters

listtype : str The type of the components list to be retrieved, options: - 'all': all 195 hatyan components - 'year': default list of 94 hatyan components - 'halfyear': list of 88 components to be used when analyzing approximately half a year of data - 'month': list of 21 components to be used when analyzing one month of data. If desired, the K1 component can be splitted in P1/K1, N2 in N2/Nu2, S2 in T2/S2/K2 and 2MN2 in Labda2/2MN2. - 'springneap': list of 14 components to be used when analyzing one spring neap period (approximately 15 days) of data - 'day': list of 10 components to be used when analyzing one day - 'tidalcycle': list of

6 components to be used when analyzing one tidal cycle (approximately 12 hours and 25 minutes)

Raises

Exception DESCRIPTION.

Returns

const_list_hatyan : list of str A list of component names.

Function get_doodson_eqvals

```
def get_doodson_eqvals(
    dood_date,
    mode=None
)
```

Berekent de doodson waardes T, S, H, P, N en P1 voor het opgegeven tijdstip.

Parameters

dood_date : **datetime.datetime** or **pandas.DateTimeIndex** Date(s) on which the doodson values should be calculated.

mode : **str**, **optional** Calculate doodson values with Schureman (hatyan default) or Sea Level Science by Pugh. The default is False.

Returns

dood_T_rad : **TYPE** Bij hatyan 180+, maar in docs niet. In hatyan fortran code is +/-90 in v ook vaak omgedraaid in vergelijking met documentation.

dood_S_rad : **TYPE** Middelbare lengte van de maan. Mean longitude of moon

dood_H_rad : **TYPE** Middelbare lengte van de zon. mean longitude of sun

dood_P_rad : **TYPE** Middelbare lengte van het perieum van de maan. Longitude of lunar perigee

dood_N_rad : **TYPE** Afstand van de klimmende knoop van de maan tot het lentepunt. Longitude of moons node

dood_P1_rad : **TYPE** Middelbare lengte van het perieum van de zon. Longitude of solar perigee

Function get_hatyan_constants

```
def get_hatyan_constants(
    dood_date
)
```

Parameters

dood_date : **TYPE** DESCRIPTION.

Returns

DOMEGA : TYPE DESCRIPTION.
DIKL : TYPE DESCRIPTION.
DC1681 : TYPE DESCRIPTION.
DC5023 : TYPE DESCRIPTION.
DC0365 : TYPE DESCRIPTION.

Function get_allconstituents_withdoodson

```
def get_allconstituents_withdoodson(  
    dood_date,  
    mode=None  
)
```

Parameters

dood_date : TYPE, optional DESCRIPTION. The default is None.

Returns

DOMEGA : TYPE DESCRIPTION.

Function get_hatyan_freqs

```
def get_hatyan_freqs(  
    const_list,  
    sort_onfreq=True  
)
```

Returns the frequencies of the requested list of constituents. Source: beromg.f

Parameters

const_list : list or pandas.Series contains the tidal constituent names.

Raises

Exception DESCRIPTION.

Returns

t_const_freq : TYPE DESCRIPTION.

Function get_hatyan_v

```
def get_hatyan_v(  
    const_list,  
    dood_date  
)
```

Returns the v-values of the requested list of constituents for the requested date(s)

Parameters

const_list : list or **pandas.Series** contains the tidal constituent names.
dood_date : TYPE DESCRIPTION.

Returns

v_0i_rad : TYPE DESCRIPTION.

Function get_hatyan_u

```
def get_hatyan_u(  
    const_list,  
    dood_date  
)
```

Returns the u-values of the requested list of constituents for the requested date(s)

Parameters

const_list : list or **pandas.Series** contains the tidal constituent names.
dood_date : TYPE DESCRIPTION.

Returns

u_i_rad_HAT : TYPE DESCRIPTION.

Function get_hatyan_f

```
def get_hatyan_f(  
    const_list,  
    dood_date,  
    xfac  
)
```

Returns the f-values of the requested list of constituents for the requested date(s)

Parameters

const_list : list or **pandas.Series** contains the tidal constituent names.
dood_date : TYPE DESCRIPTION.

Returns

f_i_HAT : TYPE DESCRIPTION.

Module `hatyan.timeseries`

`timeseries.py` contains all definitions related to `hatyan timeseries`.

`hatyan` is a Python program for tidal analysis and prediction, based on the FORTRAN version. Copyright (C) 2019-2020 Rijkswaterstaat. Maintained by Deltares, contact: Jelmer Veenstra (jelmer.veenstra@deltares.nl). Source code available at: https://repos.deltares.nl/repos/lib_tide/trunk/src/hatyan_python/hatyan

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 3 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, see <https://www.gnu.org/licenses/>.

Functions

Function `calc_HWLW`

```
def calc_HWLW(  
    ts,  
    calc_HWLWlocal=False  
)
```

Calculates extremes (high and low waters) for the provided timeseries

Parameters

ts : `pandas.DataFrame` The DataFrame contains columns 'times' and 'values', it contains the timeseries.

calc_HWLWlocal : `bool, optional` Whether to also calculate local extremes like first/second high waters and 'aggers'. The default is False, in which case only extremes of each tidal period are calculated.

Returns

data_pd_HWLW : `pandas.DataFrame` The DataFrame contains columns 'times', 'values' and 'HWLW_code', it contains the times, values and codes of the timeseries that

are extremes. 1: high water. 2: low water. 11: local high water (e.g. an ‘agger’). 22: local low water.

Function `number_HWLW`

```
def number_HWLW(
    ts_ext
)
```

Parameters

ts_ext : TYPE DESCRIPTION.

margin_hr : TYPE, optional DESCRIPTION. The default is 2.5.

Raises

Exception DESCRIPTION.

Returns

ts_ext_nos : TYPE DESCRIPTION.

Function `plot_timeseries`

```
def plot_timeseries(
    ts,
    ts_validation=None,
    ts_ext=None,
    ts_ext_validation=None
)
```

Creates a plot with the provided timeseries

Parameters

ts : **pandas.DataFrame** The DataFrame contains columns ‘times’ and ‘values’, it contains the timeseries.

ts_validation : **pandas.DataFrame, optional** The DataFrame contains columns ‘times’ and ‘values’, it contains the timeseries. The default is None.

ts_ext : **pandas.DataFrame, optional** The DataFrame contains columns ‘times’, ‘values’ and ‘HWLW_code’, it contains the times, values and codes of the timeseries that are extremes. The default is None.

ts_ext_validation : **pandas.DataFrame, optional** The DataFrame contains columns ‘times’, ‘values’ and ‘HWLW_code’, it contains the times, values and codes of the timeseries that are extremes. The default is None.

Returns

fig : **matplotlib.figure.Figure** The generated figure handle, with which the figure can be adapted and saved.

axs : (tuple of) `matplotlib.axes._subplots.AxesSubplot` The generated axis handle, with which the figure can be adapted.

Function `write_tsnetcdf`

```
def write_tsnetcdf(
    ts,
    station,
    vertref,
    filename,
    ts_ext=None,
    tzone_hr=1
)
```

Writes the timeseries to a netCDF file

Parameters

ts : `pandas.DataFrame` The DataFrame contains columns 'times' and 'values', it contains the timeseries.

station : `str` DESCRIPTION.

vertref : `str` DESCRIPTION.

filename : `str` The filename of the netCDF file that will be written.

ts_ext : `pandas.DataFrame`, **optional** The DataFrame contains columns 'times', 'values' and 'HWLW_code', it contains the times, values and codes of the timeseries that are extremes. The default is None.

tzone_hr : `int`, **optional** The timezone (GMT+tzone_hr) that applies to the data. The default is 1 (MET).

Returns None.

Function `write_tsdia`

```
def write_tsdia(
    ts,
    station,
    vertref,
    filename
)
```

Writes the timeseries to an equidistant dia file

Parameters

ts : `pandas.DataFrame` The DataFrame contains columns 'times' and 'values', it contains the timeseries.

station : `TYPE` DESCRIPTION.

vertref : `TYPE` DESCRIPTION.

filename : `TYPE` DESCRIPTION.

Raises

Exception DESCRIPTION.

Returns None.

Function `write_tsdia_HWLW`

```
def write_tsdia_HWLW(  
    ts_ext,  
    station,  
    vertref,  
    filename  
)
```

writes the extremes timeseries to a non-equidistant dia file

Parameters

ts_ext : **pandas.DataFrame** The DataFrame contains columns 'times', 'values' and 'HWLW_code', it contains the times, values and codes of the timeseries that are extremes.

station : **TYPE** DESCRIPTION.

vertref : **TYPE** DESCRIPTION.

filename : **TYPE** DESCRIPTION.

Raises

Exception DESCRIPTION.

Returns None.

Function `crop_timeseries`

```
def crop_timeseries(  
    ts,  
    times_ext  
)
```

Crops the provided timeseries

Parameters

ts : **pandas.DataFrame** The DataFrame contains columns 'times' and 'values', it contains the timeseries.

times_ext : **TYPE** DESCRIPTION.

Raises

Exception DESCRIPTION.

Returns

ts_pd_out : TYPE DESCRIPTION.

Function `resample_timeseries`

```
def resample_timeseries(  
    ts,  
    timestep_min  
)
```

resamples the provided timeseries, only overlapping timesteps are selected, so no interpolation

Parameters

ts : **pandas.DataFrame** The DataFrame contains columns 'times' and 'values', it contains the timeseries to be resampled.

timestep_min : **int** the amount of minutes with which to resample the timeseries.

Returns

data_pd_resample : **pandas.DataFrame** with columns 'times' and 'values' the resampled timeseries.

Function `check_ts`

```
def check_ts(  
    ts  
)
```

prints several statistics of the provided timeseries

Parameters

ts : **pandas.DataFrame** The DataFrame contains columns 'times' and 'values', it contains the timeseries to be checked.

Returns

print_statement : **str** For printing as a substring of another string.

Function `get_diablocks`

```
def get_diablocks(  
    filename,  
    station  
)
```

Gets information about the data blocks present in a dia file

Parameters

filename : TYPE DESCRIPTION.
station : TYPE DESCRIPTION.

Raises

Exception DESCRIPTION.

Returns

block_starts : TYPE DESCRIPTION.
data_starts : TYPE DESCRIPTION.
data_ends : TYPE DESCRIPTION.
block_stations : TYPE DESCRIPTION.
block_id : TYPE DESCRIPTION.

Function getcheck_diameta

```
def getcheck_diameta(  
    data_meta,  
    station,  
    diatype=None  
)
```

Gets and checks the metadata of a dia file

Parameters

data_meta : TYPE DESCRIPTION.
station : TYPE DESCRIPTION.
diatype : TYPE, optional DESCRIPTION. The default is None.

Raises

Exception DESCRIPTION.

Returns

file_station : TYPE DESCRIPTION.
file_vertref : TYPE DESCRIPTION.
times_fromfile : TYPE DESCRIPTION.
file_station_coord : TYPE DESCRIPTION.

Function readts_dia_HWLW

```
def readts_dia_HWLW(  
    filename,  
    station  
)
```

Reads a non-equidistant dia file

Parameters

filename : TYPE DESCRIPTION.
station : TYPE DESCRIPTION.

Returns

data_pd_HWLW : TYPE DESCRIPTION.

Function readts_dia

```
def readts_dia(  
    filename,  
    station  
)
```

Reads an equidistant dia file, or a list of equidistant dia files

Parameters

filename : TYPE DESCRIPTION.
station : TYPE DESCRIPTION.

Raises

Exception DESCRIPTION.

Returns

data_pd : TYPE DESCRIPTION.

Function readts_noos

```
def readts_noos(  
    filename,  
    datetime_format='%Y%m%d%H%M',  
    na_values=None  
)
```

Reads a noos file

Parameters

filename : TYPE DESCRIPTION.
datetime_format : TYPE, optional DESCRIPTION. The default is '%Y%m%d%H%M'.
na_values : TYPE, optional DESCRIPTION. The default is None.

Returns

data_pd : TYPE DESCRIPTION.

Module `hatyan.wrapper_RWS`

`wrapper_RWS.py` contains wrapper functions around the `hatyan` process for RWS related calculations.

`hatyan` is a Python program for tidal analysis and prediction, based on the FORTRAN version. Copyright (C) 2019-2020 Rijkswaterstaat. Maintained by Deltares, contact: Jelmer Veenstra (jelmer.veenstra@deltares.nl). Source code available at: https://repos.deltares.nl/repos/lib_tide/trunk/src/hatyan_python/hatyan

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 3 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, see <https://www.gnu.org/licenses/>.

Functions

Function `init_RWS`

```
def init_RWS(
    file_config,
    argvlist=[None],
    interactive_plots=True,
    silent=False
)
```

Initializes the `hatyan` process for RWS related calculations. Besides the return variables, it prints a header for the print output (shows up in the `hatyan` diagnostics file)

Parameters

`file_config` : TYPE DESCRIPTION.

`argvlist` : TYPE DESCRIPTION.

`interactive_plots` : **bool/int**, **optional** sets the correct matplotlib backend so plots are (not) displayed on both RedHat and windows. The default is True.

Raises

Exception DESCRIPTION.

Returns

dir_output : path/str the output directory for the hatyan process, the current directory is set to this folder.

timer_start : datetime.datetime provides a start time with which exit_RWS calculates the total time of the process.

Function exit_RWS

```
def exit_RWS(  
    timer_start  
)
```

Provides a footer to the print output (shows up in the hatyan diagnostics file)

Parameters

timer_start : TYPE The start time of the hatyan process, which is used to calculate the total time of the process.

Returns None.

Function get_outputfoldername

```
def get_outputfoldername(  
    file_config  
)
```

Creates an output folder based on the start time of the filename of the configfile and the current time.

Parameters

file_config : str or path path to the configuration file.

Raises

Exception DESCRIPTION.

Returns

dir_output : str or path path to the output directory.

Deltares is een onafhankelijk kennisinstituut voor toegepast onderzoek op het gebied van water en ondergrond. Wereldwijd werken we aan slimme oplossingen voor mens, milieu en maatschappij.

Deltares

www.deltares.nl