Uitwisseling van brongegevens in bron- (HDF5) en mat-format (Matlab)

Observaties over uitwisseling, interoperabiliteit en inzet van externe viewers

Versie : 0.3
Status : Concept
Datum : 26 september 2022
Afdeling : Water

Auteur : G.H.H. van Bergen, J. von Asmuth

provincie Gelderland

INHOUDSOPGAVE

provincie Gelderland

1 Introductie

1.1 Achtergrond

Dit document bevat observaties over het opslaan van bron-gegevens.

Op 1 januari 2018 is de Wet Basisregistratie Ondergrond in werking getreden. Met de Basisregistratie Ondergrond, hierna BRO, is er een landelijk informatiesysteem voor bodem- en ondergrondgegevens. De BRO maakt het mogelijk om digitale ruimtelijke geoinformatiemodellen in te zetten bij beleidsafwegingen. Het maakt (her)gebruik van gegevens mogelijk en zorgt ervoor dat gegevens volgens uniforme standaarden worden vastgelegd. De gegevens worden aangeleverd door verschillende bronhouders.

1.2 Doelstelling

Om het werken met bron-gegevens efficiënt te maken moeten deze gegevens in bestanden met een standaardformat worden opgeslagen zodat gegevens gemakkelijk worden kunnen uitgewisseld tussen gebruikers en software. Het doel van dit document is om de afwegingen te noteren die zijn gemaakt bij het kiezen van bestandsformats en andere gerelateerde zaken. Daarnaast evalueren we kort een aantal verschillende viewers, zowel web-based als offline, voor bestanden in het Bron/HDF5-format.

1.3 Leeswijzer

<nader uit te werken>

1.4 Versiehistorie

Datum	Versie	Omschrijving
xxx 2022	1.0	Definitief
xxx 2022	0.9	Eindconcept
xxx 2022	0.x	xxxxxx
Okt 2022	0.3	Derde concept, invulling van versie 0.2.
Sept 2022	0.2	Tweede concept, met verbrede opzet gericht op data-uitwisseling
Juli 2022	0.1	Eerste concept gericht op Bron-formatversies en opties

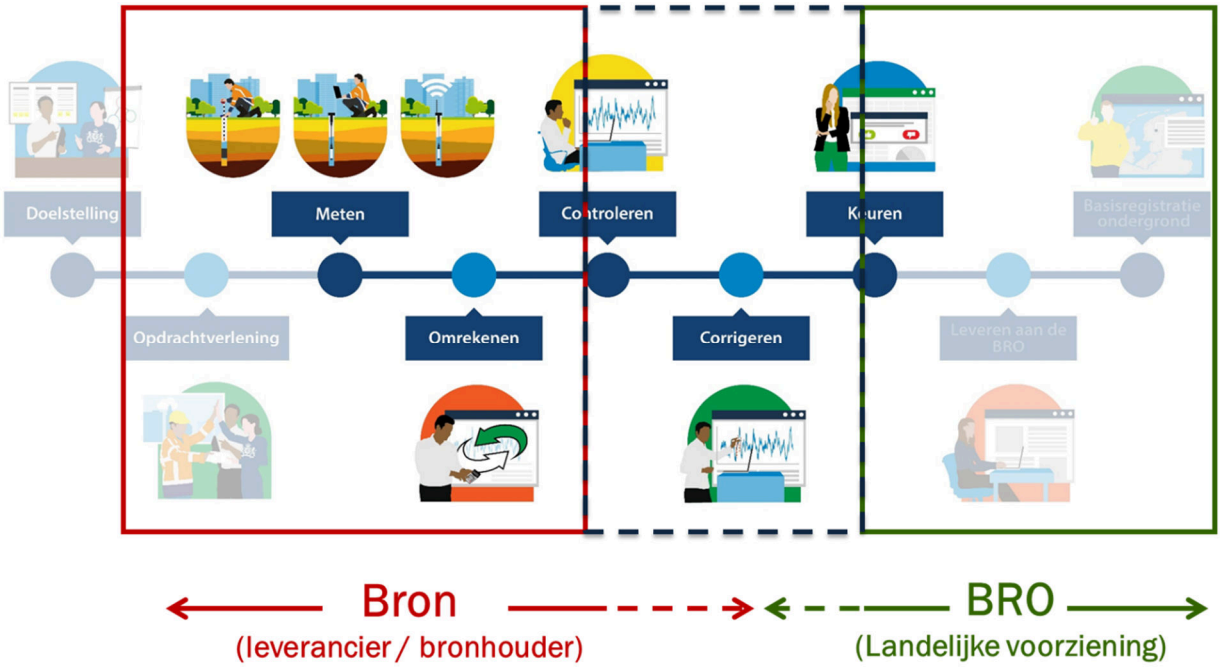
<nader uit te werken>

2 Bron- en BRO-gegevens: de principes

2.1 Inleiding

<nader uit te werken>

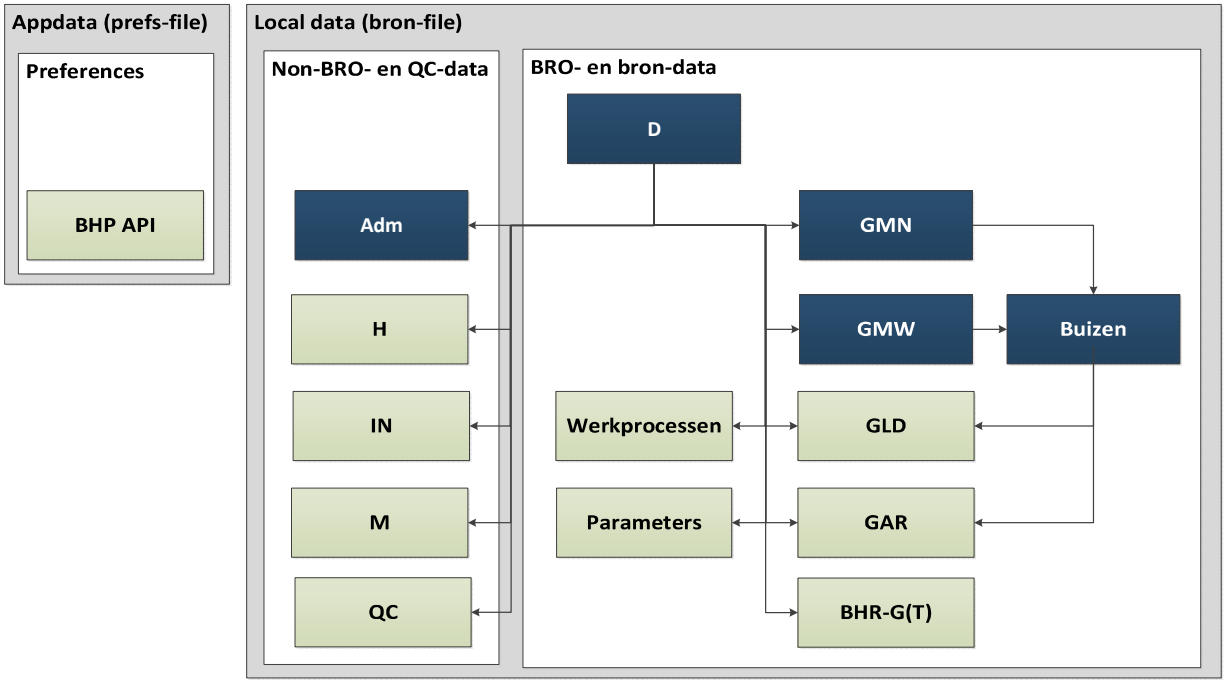
2.2 Bron-gegevens, hoe en waarom?



Figuur 1: Stappen in het werkproces, vanaf het inwinnen van de originele data of metingen in het veld tot aan het leveren van de gekeurde, definitieve grondwaterstandgegevens

2.3 Het bron-datamodel

Volgens het QC-protocol (Platform meetnetbeheerders, 2018) dienen voor opslag en uitwisseling van bron-gegevens open, eenduidig en expliciet gedefinieerde dataformats, datadefinities en datamodellen gebruikt te worden. Een eis of richtlijn is verder dat daarbij de gegevens in originele, onbewerkte en ongefilterde toestand opgeslagen dienen te worden, met daarnaast de uitgevoerde omrekeningen, controles, correcties en keurmerken.



Figuur 2: Schematische weergave van de globale opzet en inhoud van het zogeheten bron-datamodel dat MenyanthesOS en de QC-Wizard gebruiken voor dataopslag en uitwisseling.

<nader uit te werken>

3 Opslag en uitwisseling van Bron-bestanden

3.1 Inleiding

Volgens het QC-protocol dienen voor opslag en uitwisseling van gegevens open, eenduidig en expliciet gedefinieerde dataformats, datadefinities en datamodellen gebruikt te worden. Een eis of richtlijn is verder dat daarbij de gegevens in originele, onbewerkte en ongefilterde toestand opgeslagen dienen te worden, met daarnaast de uitgevoerde omrekeningen, controles, correcties en keurmerken.
Het zogeheten bron-format is het format dat MenyanthesOS en de QC-Wizard zelf gebruiken voor dataopslag en uitwisseling, de structuur is dat het naadloos aansluit bij de datamodellen van de BRO zelf aan de ene kant, terwijl het aan de andere kant ook ruimte laat voor opslag en uitwisseling van de relevante bron- en QC-gegevens. Dergelijke aanvullende gegevens worden daarbij netjes afgebakend en 'gemapt' op de BRO-gegevens. Het spreadsheet- en tabelformat format is bovendien eenvoudig overdraagbaar en vertaalbaar in verschillende varianten en platforms. De globale opzet en inhoud van het zogeheten bron-format zijn weergegeven in Figuur 2.

3.2 HDF5

HDF5 is een open hiërarchisch binair bestandsformaat: een specificatie hoe gegevens in een bestand kunnen worden opgeslagen, de HDF5 standaard wordt onderhouden door The HDF Group (https://www.hdfgroup.org). Naast het actief ontwikkelen van de standaard[1] ontwikkelt The HDF5 Group ook een library genaamd 'HDF5' die de standaard implementeert. met HDF5 bestanden kan worden gewerkt. De library 'HDF5' ondersteunt de programmeertalen C, C++, Fortran en Java en is beschikbaar onder een permissive vrije software licentie (BSD-3 clause). The HDF Group ondersteunt ook nog HDF4, een oudere standaard, daar gaan wij verder niet op in.

De structuur van HDF5-bestanden is heel vergelijkbaar met een (Unix) directorystructuur: er zijn Groups (analoog aan mappen) en Datasets (bestanden), allemaal onder het top-level `/` geplaatst. Groups kunnen zelf niet direct gegevens bevatten, maar kunnen wel andere Groups en Datasets bevatten.

Om dit concreet te maken kijken naar het voorbeeld van een HDF5-bestand met een Dataset genaamd `/group/subgroup/data5` bevindt. Dit pad betekent dat de naam van de Dataset `data5` is, zich in de Group `subgroup`, die op zijn beurt weer in de Group `group` bevindt, die zich in het top-level `/` bevindt.

Naast gegevens, die alleen in Datasets zitten, zijn er ook Attributes. Attributes hebben een naam en een waarde, en kunnen als metadata bij een Group of Dataset worden opgeslagen voor context. Als motivatie voor Attributes, neem aan dat we een Dataset hebben waarin een 1-dimensionale array van 32 bit floating point getallen is opgeslagen. Zonder verdere informatie is het lastig om te weten wat deze getallen betekenen. Misschien staat in een ander bestand de betekenis van deze gegevens, maar dat bestand kan kwijtraken of uit de pas gaan lopen. Met Attributes kan extra informatie, bijvoorbeeld `Temperatuureenheid`=`K`, `SensorID`=12345 en `Tijdeenheid`=`h`, direct bij de Dataset worden opgeslagen. Er is in principe geen beperking in grootte van Attributes, maar echte gegevens in Attributes opslaan zou erg inefficiënt zijn omdat veel functionaliteit om efficiënt te lezen en schrijven alleen voor Datasets is geïmplementeerd en niet voor Attributes.

3.2.1 Matlabondersteuning

In Matlab zijn er twee manieren om met HDF5 te werken: de high-level functies en de low-level functies. De high-level functies zijn makkelijk in het gebruik, maar vrij beperkt in functionaliteit. Omdat deze functies in de oudere Matlab-versie die wordt gebruikt voor MenyanthesOS niet geen Datasets ondersteunen die strings bevatten, hebben we dat zelf geïmplementeerd.

[1] Op het moment van schrijven is de meest recente stabiele versie HDF5-1.12.0 van 1 juli 2021.

De low-level functies zijn een lichte wrapper om de C library, zo licht dat de handleiding van Matlab veel naar de HDF5 documentatie verwijst, omdat de API bijna hetzelfde is.

3.2.2 Pythonondersteuning

In Python is h5py de meest gangbare HDF5 wrapper library, h5py ondersteunt de meeste functionaliteit op een high-level manier, maar behoudt de mogelijkheid om de details in te stellen.

3.3 Bron-format: versies en opties

Er zijn meerdere, incompatibele versies van het bron-format: BronV1 en BronV2. Deze zijn allemaal op HDF5 gebaseerd en volgen de hiërarchie van de bron-datamodellen maar verschillen in de details waarop de BRO-datastructuren naar de datastructuren van HDF5 worden vertaald. Deze verschillen zitten op een dusdanig detailniveau dat eindgebruikers hier niets van merken, behalve met betrekking tot bestandsgrootte en lees- en schrijfsnelheid (zie Hfst 4).

3.3.1 BronV1

Het BronV1 format is identiek aan het format van Matlab MAT files v7.3+. Dit format is gebaseerd op HDF5 en deze bestanden kunnen met HDF5-software worden gelezen. Als deze bestanden echter custom Matlab Datatypes, zoals Tables, bevatten, dan worden de gegevens als onnavolgbare binaire blobs opgeslagen en kan enkele andere software dan Matlab de gegevens op een bruikbare manier inlezen. Omdat de BRO-objecten die we willen opslaan inderdaad Tables bevatten is dit een probleem voor gegevensuitwisseling met software die niet in Matlab geschreven is.

3.3.2 BronV2

Omdat het gewenst is als BRO-objecten ook met software in andere programmeertalen dan Matlab uitgewisseld kunnen worden, is BronV2 ontwikkeld. Dit format is ook op HDF5 gebaseerd en specifiek bedoeld om BRO-objecten op te slaan en uit te lezen, maar heeft een open specificatie (Van Bergen, 2022). BronV2 implementaties voor Python en Matlab om BRO-objecten op te slaan en in te lezen zijn op aanvraag beschikbaar.[2] De Matlab versie is geschreven op basis van de high-level HDF5 API, de Python library met behulp van h5py.

3.3.3 Mat Legacy

Het Mat format van vóór versie 7.3, dat wij kortweg “Mat Legacy” format zullen noemen, is het opslagformat dat Matlab gebruikte vóór het op HDF5 gebaseerde 7.3 format overschakelde, maar dat nog altijd door Matlab ondersteund. Net als bij BronV1 bestanden kan dit format door andere software dan Matlab gelezen worden als er geen custom Matlab Datatypes (tabellen) in staan. In python gaat dit met de functie `scipy.io.loadmat`.

4 Read-Write performance Benchmarks

Om de performance van de verschillende bestandsformaten te vergelijken hebben we een grotere dataset (1193 GMWs uit Gelderland) in de drie bestandsformaten Mat, BronV1 en BronV2 opgeslagen en ingelezen en daarvan de bestandsgrootte en de tijd die het duurde om te lezen en schrijven opgeslagen. Deze benchmarks zijn gedaan op een laptop met Windows 10, op een SSD en met Matlab R2014a.

	MatLegacy	**BronV1**	**BronV2**
Lezen (s)	**1,4**	9,7	341
Schrijven (s)	**1,3**	22,9	1490 (≈ 25 minuten)
Bestandsgrootte (MB)	**2,9**	144	386
Cross-platform	×	×	✓

[2] Email het verzoek naar g.van.bergen@gelderland.nl

Het ouderwetse Mat format is de duidelijke winnaar zowel in opslagruimte, als in lees- en schrijfsnelheid, BronV2 is veruit het langzaamst en grootst.

Een aanpak om dit te verhelpen is optimalisatie van de BronV2 library: volgens de Matlab-profiler gaat bij het schrijven bijna 90% van de tijd zitten in het telkens openen van het bestand, dit komt doordat de BronV2 library de high-level functies gebruikt, en deze functies openen en sluiten het bestand steeds zelf. Met de low-level functies is het wel mogelijk om het bestand één keer te openen, en de file handle rond te sturen. BronV2 herschrijven met gebruik van de low-level functies kost wel veel extra tijd en het is onduidelijk hoezeer dit nodig is, maar kan wel de tijd met een factor 10 verminderen. Dit vereist geen veranderingen aan de python code, die trouwens al met file handles werkt en inderdaad veel sneller is dan de Matlab-code. Dit heeft geen invloed op het bestandsformaat, en dus de bestandsgrootte.

Een grote verbetering van performance door het BronV2 format zelf te verbeteren lijkt niet realistisch omdat er in de BronV1 en MatLegacy formats vele uren denkwerk van specialisten is gestoken, die niet voor handen zijn voor BronV2.

Vanwege het grote verschil in bestandsgrootte en snelheid raden we aan om bron-gegevens wanneer ze zeker alleen met Matlab software hoeven worden uitgewisseld, op te slaan in het Mat-format, en anders in het BronV2-format.

5 Gebruik van HDF5-viewers

5.1 Inleiding

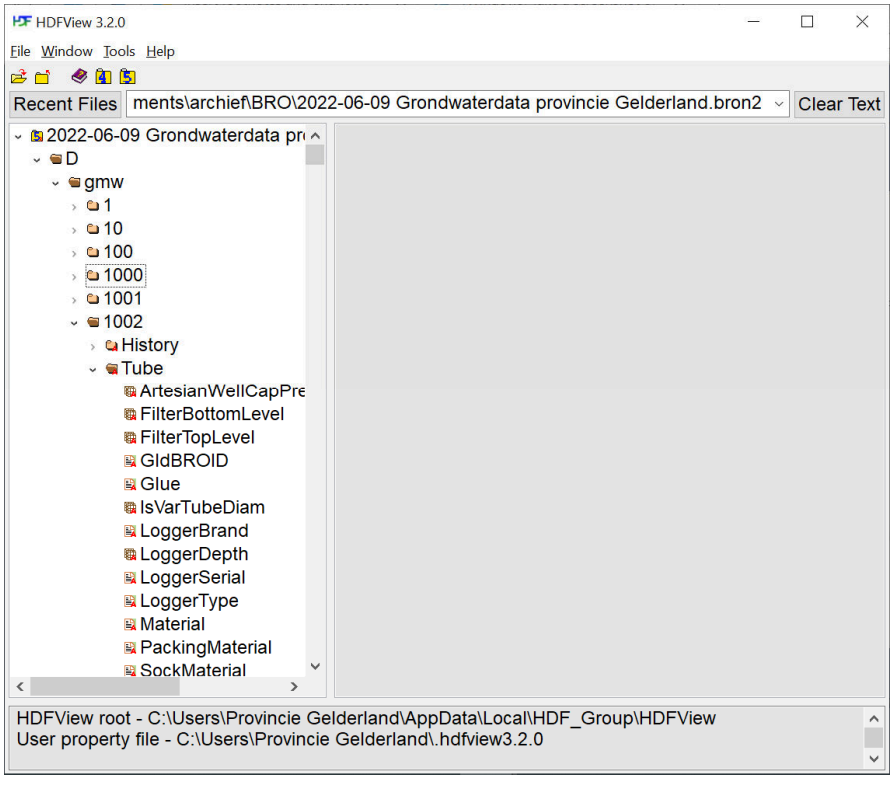
HDF5 is zelf een binair format en dus niet rechtstreeks te bekijken of wijzigen, maar er bestaan verschillende programma's met ondersteuning voor het bekijken en wijzingen van HDF5-bestanden. Omdat BronV2 bovenop het HDF5-format is gespecificeerd kunnen alle HDF5-viewers de BronV2 bestanden in principe openen, maar hebben ze geen ondersteuning voor de BronV2 structuur. Tabellen kunnen bijvoorbeeld niet in hun geheel worden bekeken, maar alleen de individuele kolommen die als aparte HDF5 Datasets zijn opgeslagen. Omdat de elementen van een tabelrij elk in een andere HDF5 dataset staan, is het met al deze viewers ook niet mogelijk om een tabelrij in zijn geheel weer te geven.

5.2 ViTables

ViTables is een HDF5 viewer geschreven in python, bovenop de PyTables library. Deze hebben we al snel afgewezen omdat PyTables, en dus ook ViTables, geen Variable Length Strings ondersteunt. Omdat in het BronV2-format elke string een VLS is, kan ViTables veel gegevens niet kan laten zien. Reken er niet op dat ondersteuning van VLS binnenkort komt, de feature request is ondertussen al meer dan 11 jaar oud.

5.3 HDFView

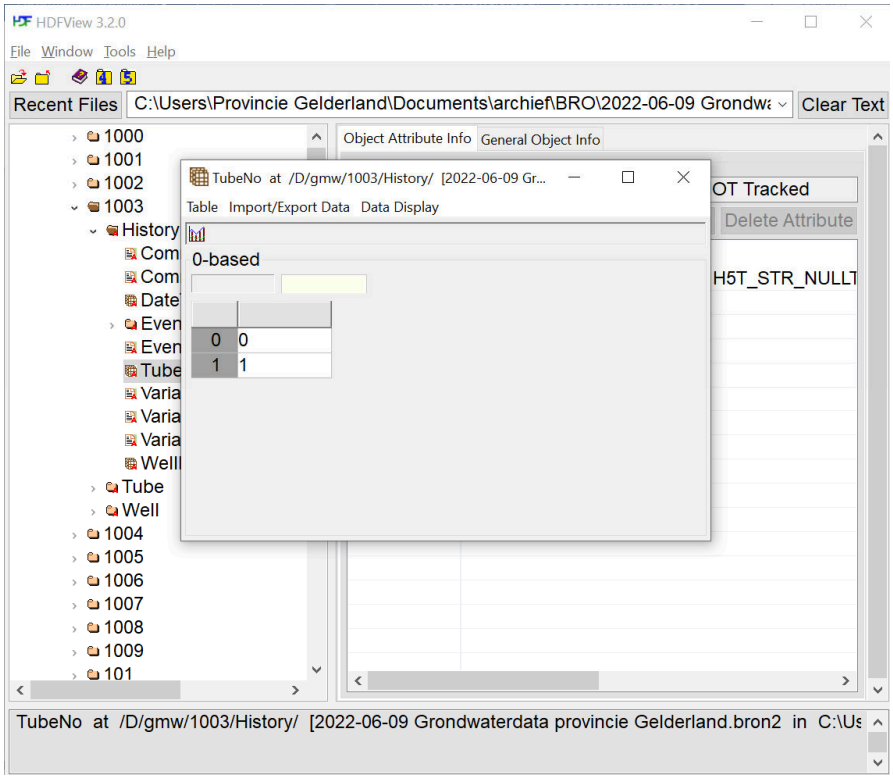
HDFView,[3] door de HDFGroup, is de "officiële" HDF5-viewer en via het webadres https://www.hdfgroup.org/downloads/hdfview/ te krijgen (gratis registratie vereist). HDFView ondersteunt Variable Length Strings wél, en kan gegevens en groepen wijzigen, toevoegen en verwijderen, al moet dat bij Bron-bestanden voorzichtig gebeuren om inconsistenties in het format te voorkomen.
Met HDFView zijn BronV2-bestanden prima te bekijken, zie en **Error! Reference source not found.**.
HDFView kan ook uitstekend overweg met grotere BronV2-bestanden, we hebben het getest met een bestand van meer dan 380MB.



Figuur 3: Screenshot van HDFView waarin een BronV2 bestand is geopend.

[3] Op die website staan de downloads 'HDFView' en 'HDFViewApp'. Die downloads zijn hetzelfde programma, maar met het verschil dat de 'HDFView' een installer bevat om HDFView te installeren en dat 'HDFViewApp' een map met een executable die direct uitgevoerd kan worden zonder systeembrede installatie.

Bestanden in het BronV1-format met tabellen zijn HDF5 bestanden en in principe met HDFView te openen, maar ondoorgrondelijk en met grotere BronV1 bestanden wordt HDFView vaak zo traag dat het lijkt te zijn vastgelopen.



Figuur 4: Screenshot van HDFView met hetzelfde BronV2 bestand als Figuur 3, waar de waarden van een tabelkolom (bestaande uit twee elementen) zijn weergegeven.

5.4 Online Viewers

Fileproinfo.com (https://fileproinfo.com/free-viewer/hdf): Deze online viewer hebben we op verschillende momenten geprobeerd te gebruiken, maar na uploaden kregen we steeds de foutmelding “Oops! All servers are extremely busy, please try again later.” Daarom raden we deze niet aan.

H5web/h5wasm (https://h5web.panosc.eu/h5wasm): Een open source online viewer die eventueel zelf kan worden gehost (Github). Deze viewer kan prima overweg met een BronV2 bestand van meer dan 300MB en heeft de mogelijkheid om data visualiseren.

6 Gebruik in Python

6.1 Inleiding

<Waarom hier? Dit hoort toch in de brohdf5-documentatie thuis?>

7 Samenvatting en conclusies

7.1 Bron-format versies en opties

Zonder tabellen zijn alle formats uitwisselbaar met andere programmeertalen. In Python kunnen Mat-files worden geopend met scipy.io.loadmat en BronV1 en BronV2 files met de h5py library.
Mét tabellen kunnen alleen BRO-objecten uit BronV2-bestanden worden ingelezen. De implementatie van BronV2 is beperkt tot het opslaan van BRO-objecten, al zijn die beperkingen makkelijk te verhelpen.
De performance van BronV2 is beduidend slechter dan die van de oude formats, zowel in bestandsgrootte als lees- en schrijfsnelheid. Daarom genieten BronV1 of MatLegacy de voorkeur als gegevens alleen met andere Matlab-software hoeft worden uitgewisseld. Als de gegevens moeten worden uitgewisseld met software niet in Matlab geschreven is, is BronV2 de enige manier om BRO-objecten uit te wisselen.

7.2 HDF5-Viewers

Van de offline HDF5-viewers waar we naar hebben gekeken ondersteunt HDFView de meeste features, en is makkelijk te installeren, de gebrek aan ondersteuning van Variable Length Strings in ViTables maakt dit minder geschikt. Van de online HDF5-Viewers is H5wasm een uitstekende keuze.

Bijlage 1

8 Literatuur

Van Bergen, Giel. 2022. *BronV2 Specificatie*. Arnhem : sn, 2022.