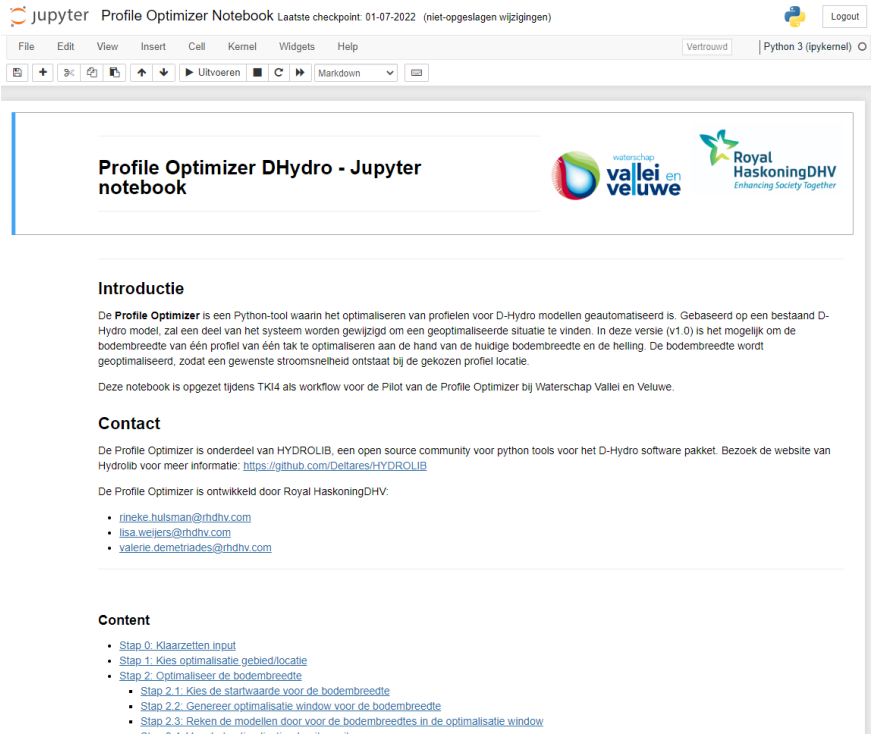
Hydrolib - Profile Optimizer

TKI4 – Voortgangsoverleg

BH6657
Project related
Lisa Weijers, Valerie Demetriades,
Rineke Hulsman
14 Juli 2022

Status Profile Optimizer

- Eerste versie afgerond
  - Beschikbaar op branch: *rhdhv_profile_optimizer_first_commit*
  - Pilot in uitvoering



Profile Optimizer: Wat doet het?

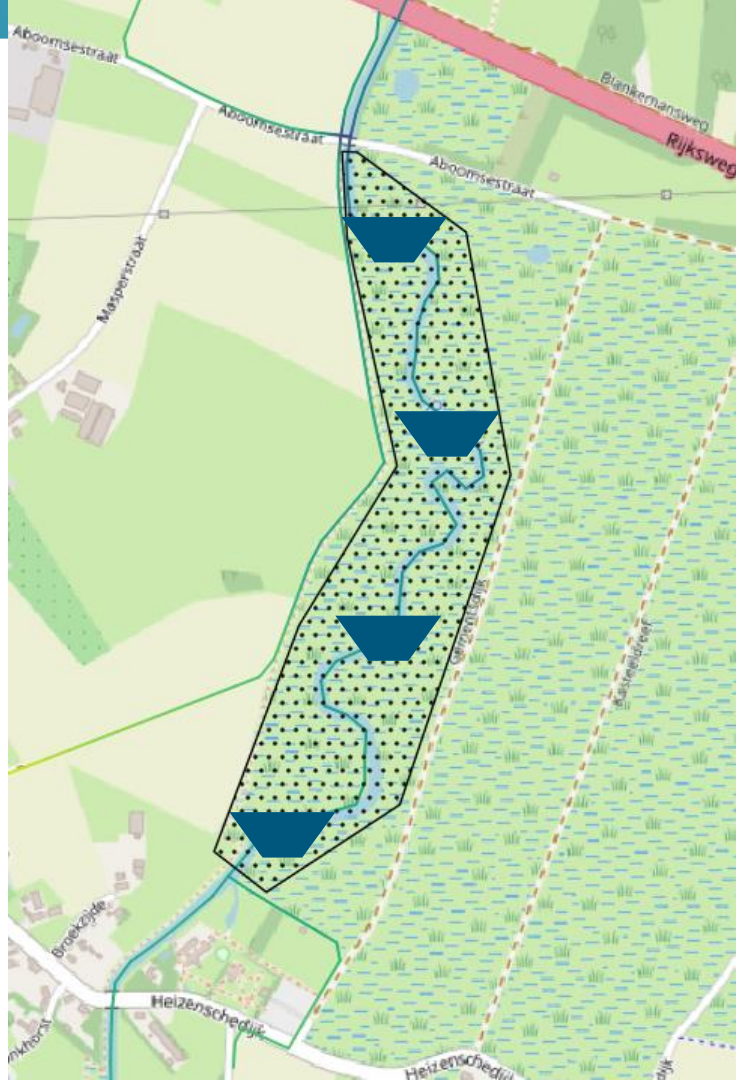
- Automatische modeloptimalisatie:
  - Bodembreedte aanpassen t.b.v. KRW/Beekhersel
  - → Doelwaarde: stroomsnelheid

- **Geparametriseerd profiel:**
  - Gebruiker kiest talud
  - Huidige versie resulteert in 1 optimaal profiel:
    - 1 Gekozen talud
    - 1 Geoptimaliseerde breedte

Workflow Profile Optimizer

- **Selecteer optimalisatie gebied**

- Maakt D-Hydro netwerk (net.nc)
- Crosssection location selecteren d.m.v. shapefile
- Alles binnen shapefile krijgt hetzelfde profiel
  - Huidige bodemhoogte behouden
  - 1 gekozen talud
  - 1 geoptimaliseerde bodembreedte



Workflow Profile Optimizer

- **Kies startwaarde voor optimalisatie van bodembreedte**

- Hulp-functie:
  - Schatting bodembreedte op basis van gewenste stroomsnelheid via Manning
  - Controle start-breedte met Q=V*A
    - Voldoende debiet bij gevonden B met gewenste V?
    - Indien nodig bijwerken en dubbel-check start-breedte

- Hulp-functie mag overgeslagen worden, kies dan zelf een start-breedte

In [7]:

```
from startvalue_for_optimization import get_b_from_manning_with_v, check_QVA

V_target = 0.5
Q = 0.9
d = 0.5
talud = 1
slope = 0.001
kmanning = 30

solved = get_b_from_manning_with_v(kmanning, slope, talud, d, V_target)
print(solved)

b = solved[0]

improved_b = check_QVA(Q, d, talud, b, slope, kmanning, allowed_variation=0.05)
```

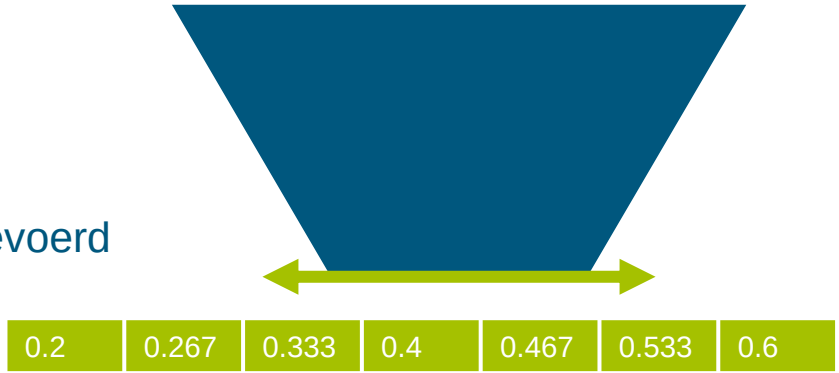
```
[2.48019353105879]
Adjustment 1: new width: 2.60, V: 0.5032, Q: 0.7809
Adjustment 2: new width: 2.73, V: 0.5062, Q: 0.8187
Adjustment 3: new width: 2.87, V: 0.5093, Q: 0.8584
```

Workflow Profile Optimizer

- **Zoekruimte runs worden gemaakt en gedraaid**

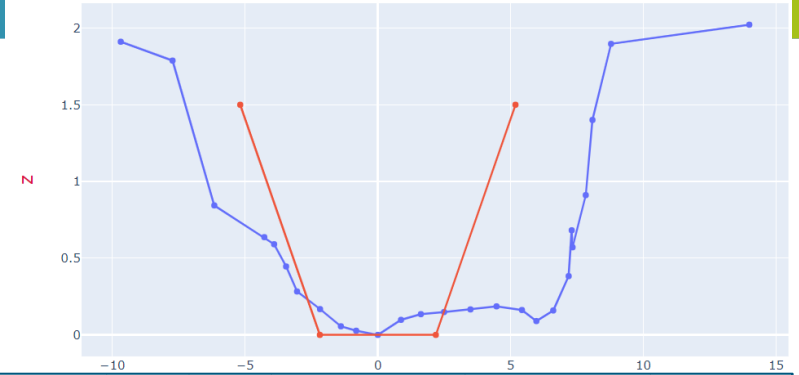
- Zoekruimte
  - Beginbreedte en x waardes erboven & eronder, met bandbreedte in %
    - Beginbreedte: 0.4 m
    - Aantal iteraties: 7
    - Bandbreedte: 50%

- Iteratie aanpassingen met Hydrolib-core uitgevoerd
- Iteraties doorrekenen met DIMR



Toepassing HYDROLIB-core

- Profielen aanpassen:
  - Crossdeffile omzetten naar dataframe
  - Slice (selecteer) gewenste profielen
  - Pas profielen aan:
    - ycoordinates & zcoordinates
    - yzcount
    - frictionpositions

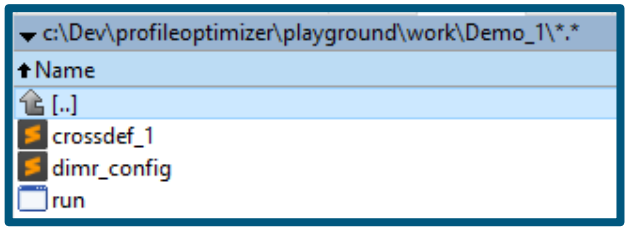


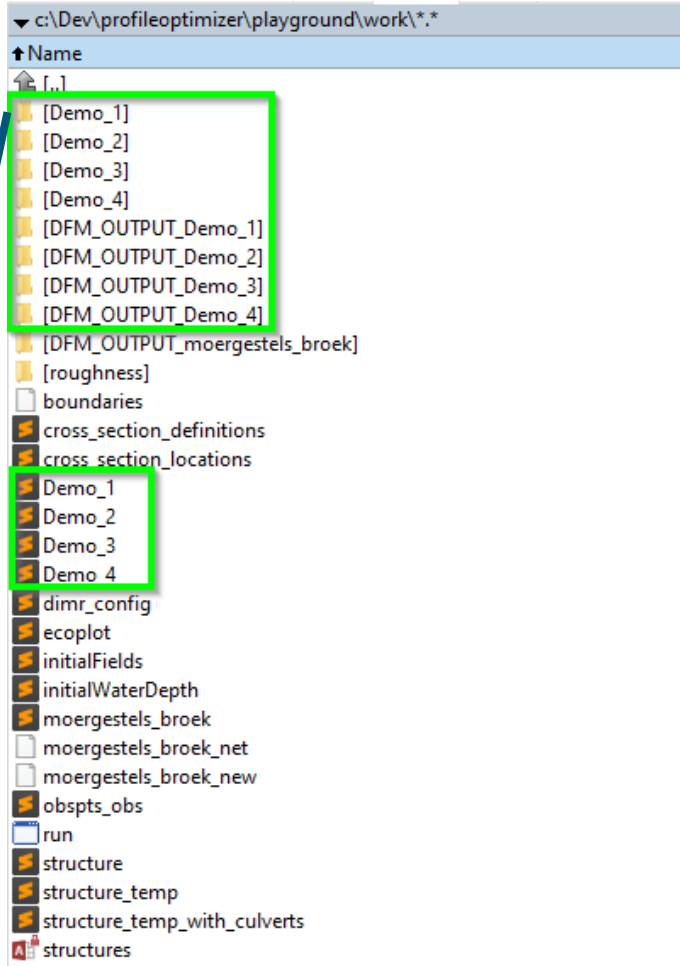
id	type	yzcount	ycoordinates	zcoordinates	frictionpositions
RS375-KDU4	circle				
prof_04082014-DP11	yz	4	[0, 4, 7, 11]	[9.769, 7.769, 7.769, 9.7	[0, 11]
prof_24102013-DP10	yz	27	[0.0, 1.855, 2.322, 4.25, 4	[11.392, 11.278, 11.287	[0.0, 23.172]
prof_24102013-DP11	yz	27	[0.0, 0.801, 2.22, 2.852, 3	[11.192, 10.98, 10.862,	[0.0, 21.657]
prof_24102013-DP12	yz	27	[0.0, 0.769, 2.171, 3.029,	[11.129, 10.962, 10.99,	[0.0, 22.831]
prof_24102013-DP13	yz	27	[0.0, 0.579, 2.17, 3.062, 3	[11.122, 10.943, 10.814	[0.0, 23.77]
prof_24102013-DP14	yz	26	[0.0, 1.951, 3.524, 5.407,	[11.006, 10.883, 9.939,	[0.0, 23.676]
prof_24102013-DP8	yz	26	[0.0, 4.577, 6.377, 7.182,	[11.536, 11.366, 10.098	[0.0, 25.373]
prof_24102013-DP9	yz	26	[0.0, 2.719, 3.59, 3.942, 4	[11.431, 11.174, 10.747	[0.0, 23.205]
prof_29042015-DP1	yz	18	[0.0, 0.997, 2.119, 9.226,	[8.501, 8.872, 9.488, 9.7	[0.0, 42.513]

```
cross_def = pd.DataFrame([cs.__dict__ for cs in self.base_model.geometry.crossdeffile.definition])

to_change_def = cross_def[cross_def['id'].isin(prof_ids)]
bottom_levels = [min(zcoords) for zcoords in to_change_def['zcoordinates']]

yz = [self._trapezium_coordinates(bl, **trapezium_pars) for bl in bottom_levels]

cross_def.loc[to_change_def.index, 'ycoordinates'] = pd.Series([y for y, z in yz], index=to_change_def.index)
cross_def.loc[to_change_def.index, 'zcoordinates'] = pd.Series([z for y, z in yz], index=to_change_def.index)
cross_def.loc[to_change_def.index, 'frictionpositions'] = pd.Series([[0, y[-1]] for y, z in yz],
                                                                    index=to_change_def.index)
cross_def.loc[to_change_def.index, 'yzcount'] = pd.Series([len(y) for y, z in yz], index=to_change_def.index)

cross_def = cross_def.replace({np.nan: None})
crossdef_new = CrossDefModel(definition=cross_def.to_dict("records"))
```

Toepassing HYDROLIB-core

- Vervang crosssection definitions in FM Model
- Exporteer nieuwe iteratie MDU
  - Houd algemene bestanden ongewijzigd
    structures, crosssection locations, boundaries, etc.
  - Nieuwe MDU, crosssection definitions
- Maak een DIMR voor dit model

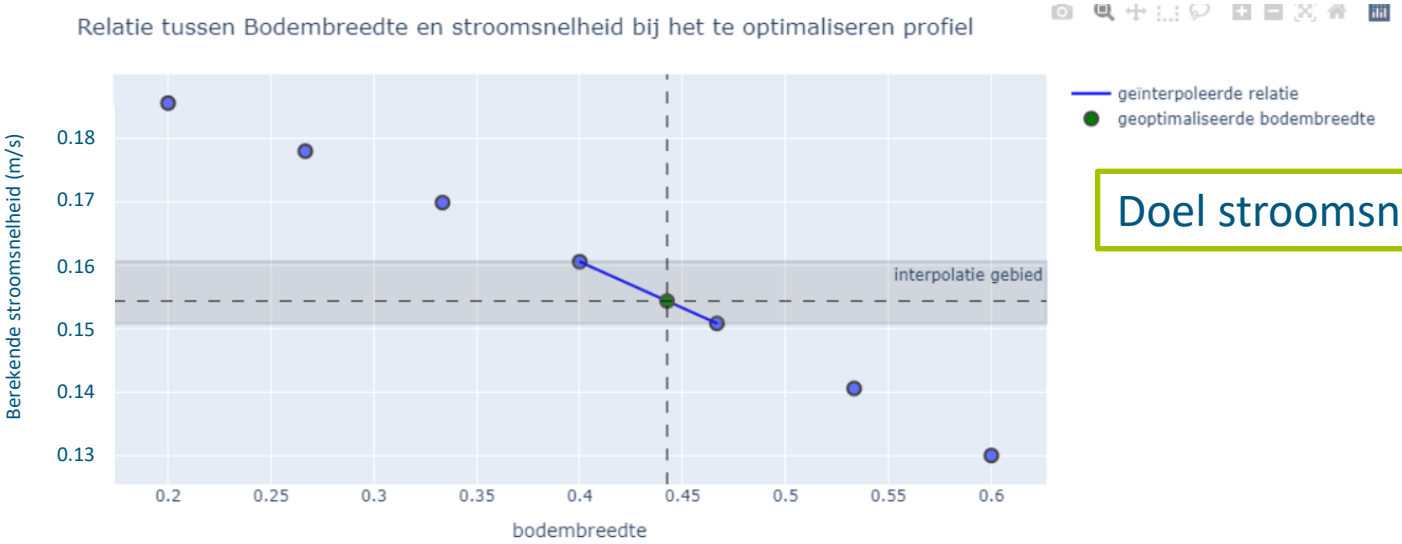




Workflow Profile Optimizer

- **Optimalisatie run wordt gemaakt en gedraaid**

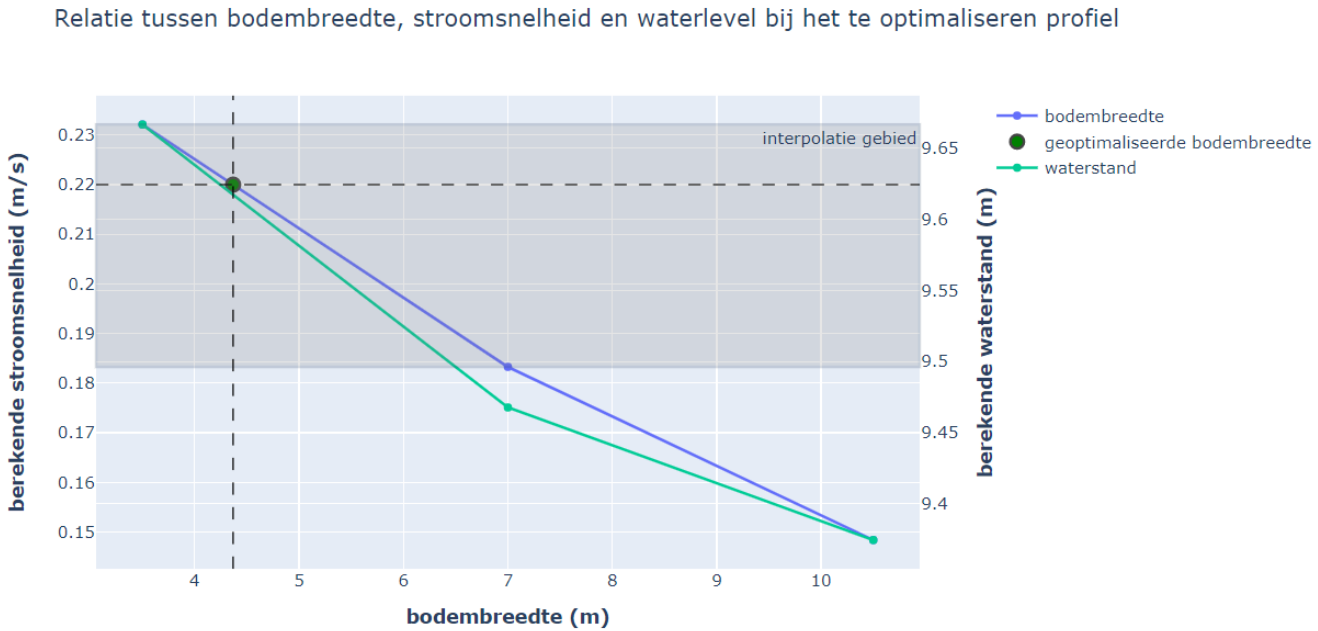
- Stroomsnelheid in toetsruimte wordt uitgelezen (op 1 gekozen locatie (x, y))
  - Map.nc lezer → GIS bestand met resultaat laatste tijdstap
- Tussen welke waardes van de zoekruimte ligt de gewenste stroomsnelheid?
  - Hiertussen interpoleren
  - = optimale bodembreedte



Doel stroomsnelheid: 0.155 m/s

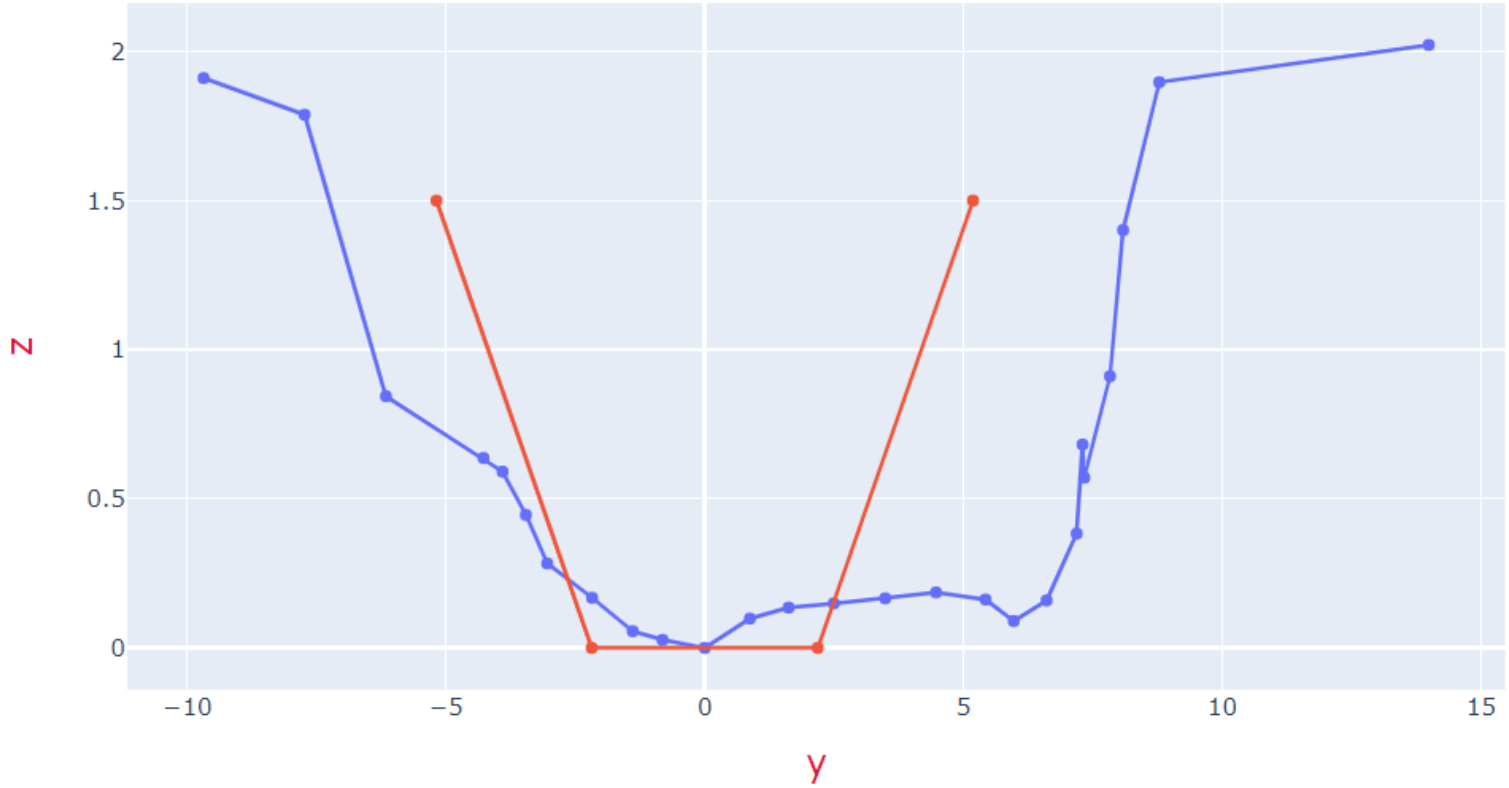
Berekende waterdiepte

- Extra grafiek: Waterdiepte
  - Geen constraint, alleen informatief.



Royal HaskoningDHV

Resulterend profiel:



Workflow Profile Optimizer

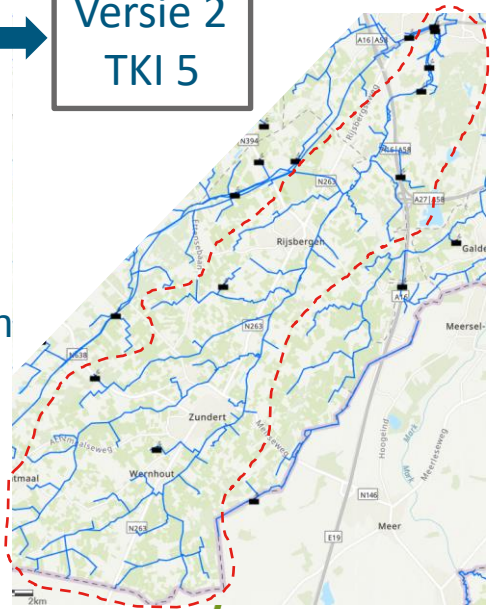
- Startpunt: D-Hydro FM model (Model met RR, RTC, 2D nog niet getest)
  Stationair
  YZ-profielen

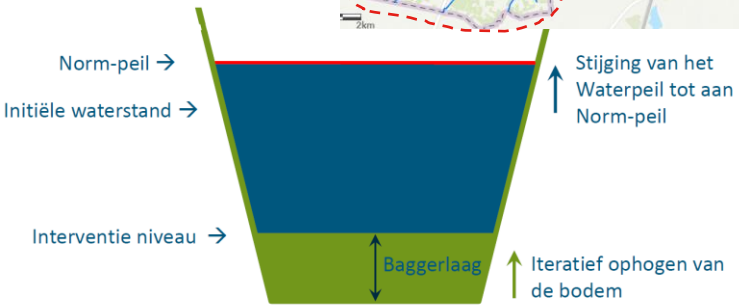
Beperkingen & Wensen

- Beperkingen:
  - Maar 1 profiel mogelijk
  - Stroomsnelheid wordt op 1 locatie getoetst
  - Bodemhoogte ongewijzigd
- Als ideale bodembreedte buiten zoekruimte ligt faalt optimalisatie
  - Advies: handmatige start-breedte en voldoende bandbreedte kiezen

- Wensen:
  - Groter optimalisatie gebied, verschillende profielen op vertakkingen of op intervallen
  - Bodemhoogte verondiepen
  - Mogelijkheid verschillende afvoersituaties

Versie 1 TKI 4 → Versie 2 TKI 5

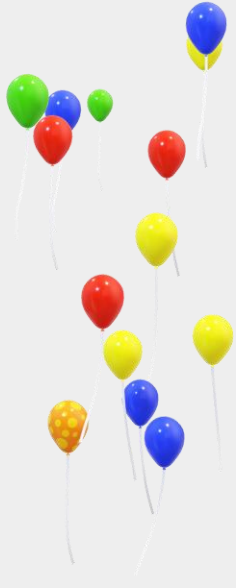




Pilot – Waterschap Veluwe

- Downloaden GitHub
- Openen Jupyter Notebook -> Firefox

Profile Optimizer DHydro - Jupyter notebook



Introductie

De **Profile Optimizer** is een Python-tool waarin het optimaliseren van profielen voor D-Hydro modellen geautomatiseerd is. Gebaseerd op een bestaand D-Hydro model, zal een deel van het systeem worden gewijzigd om een geoptimaliseerde situatie te vinden. In deze versie (v1.0) is het mogelijk om de bodembreedte van één profiel van één tak te optimaliseren aan de hand van de huidige bodembreedte en de helling. De bodembreedte wordt geoptimaliseerd, zodat een gewenste stroomsnelheid ontstaat bij de gekozen profiel locatie.

Deze notebook is opgezet tijdens TKI4 als workflow voor de Pilot van de Profile Optimizer bij Waterschap Vallei en Veluwe.

Contact

De Profile Optimizer is onderdeel van HYDROLIB, een open source community voor python tools voor het D-Hydro software pakket. Bezoek de website van Hydrolib voor meer informatie: https://github.com/Deltares/HYDROLIB

De Profile Optimizer is ontwikkeld door Royal HaskoningDHV:

- rineke.hulsman@rhdhv.com
- lisa.weijers@rhdhv.com
- valerie.demetriades@rhdhv.com

Content

Suggesties ter verbetering (Waterschap V&V)

- De modellen worden via de DIMR gestart en er wordt 1 berekening tegelijk uitgevoerd. Tegenwoordig hebben de meeste computers meerde processoren, handig om die te benutten.
- Eenmaal de notebook volledig doorlopen, dan loopt ie er tegenaan dat de work-folder al bestaat en informatie niet kan worden overschreven. Ik heb dit nu opgelost door een work2 aan te maken.
- Gegevens van de hydrologische kenmerken uit het model halen, i.p.v. zelf invoeren?
- De checkpointlocatie X en Y die moet worden opgegeven, een ID opgeven?
- Selectie van één (of meerdere) reaches die je wil optimaliseren i.p.v. een shapefile