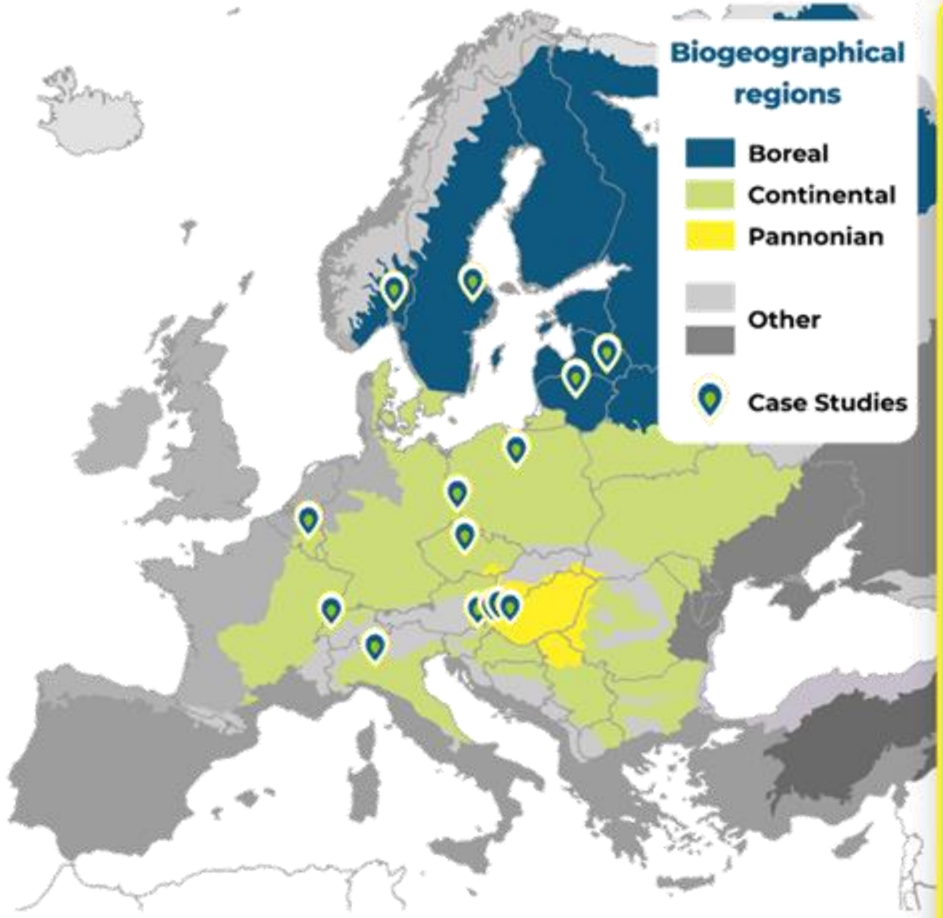
Introduction into OPTAIN & goals of the webinar

OPTAIN

Optimal strategies to retain and re-use water and nutrients in small agricultural catchments across different soil-climatic regions in Europe

PROJECT INFO



- **21** partners from 15 countries across Europe
- **14** partners will contribute with their own case study
- **7** million Euro budget
- **5** years duration 2020-2025

@H2020OPTAIN

@H2020_OPTAIN

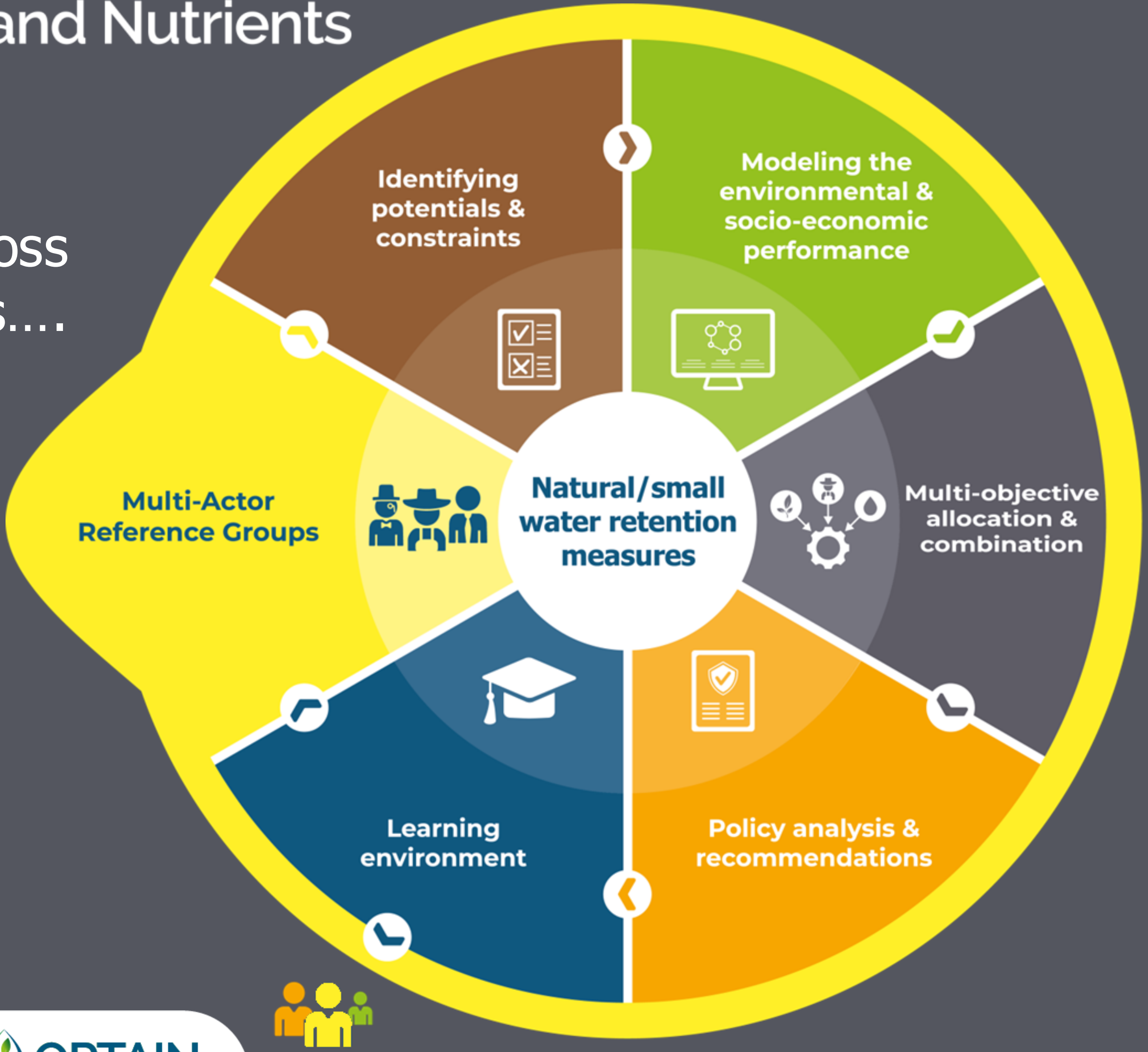
Coordinator
Prof. Dr. Martin Volk
Helmholtz Centre for Environmental Research – UFZ

WWW.OPTAIN.EU

H2020
GA 862756

OPTAIN Optimal Strategies to Retain Water and Nutrients

Fully harmonized approach across all 14 case studies….



Natural/Small Water Retention Measures (NSWRM)

Changing land cover ('permanent' greening)

- Riparian buffers
- Edge-of-field filter strips
- Hedges dividing large fields
- Grassland cover on erosive slopes
- Grassland cover in recharge areas
- Afforestation

Changing morphology & drainage

- Retention/detention ponds
- Constructed wetlands
- Controlled drainage
- Terracing
- Swales

Changing hydromorphology

- Floodplain restoration
- Channel restoration

Changing crop/soil management

- No-till agriculture
- Low-till agriculture
- Mulching
- Subsoiling
- Crop rotation
- Intercropping
- Cover crops
- Early sowing
- Drought-resistent plants

(● German case study)

→ Detailed documentation of existing examples (qcat.wocat.net)

→ Modeling environmental and economic performance

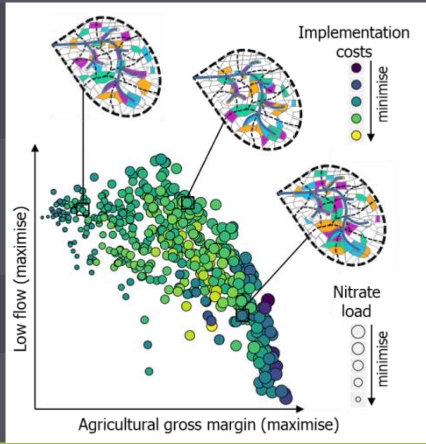
Environmental:
- Soil moisture, surface runoff, streamflow
- N, P, sediment (on-site losses, river loads)

Economic:
- Agricultural gross margin, grain units
- Implementation and maintenance costs

SWAP SWAT+ Surveys of famers/farm advisors

→ Multi-objective optimisation of measure allocation

Analyse trade-offs and identify solutions preferred by stakeholders



→ Policy recommendations

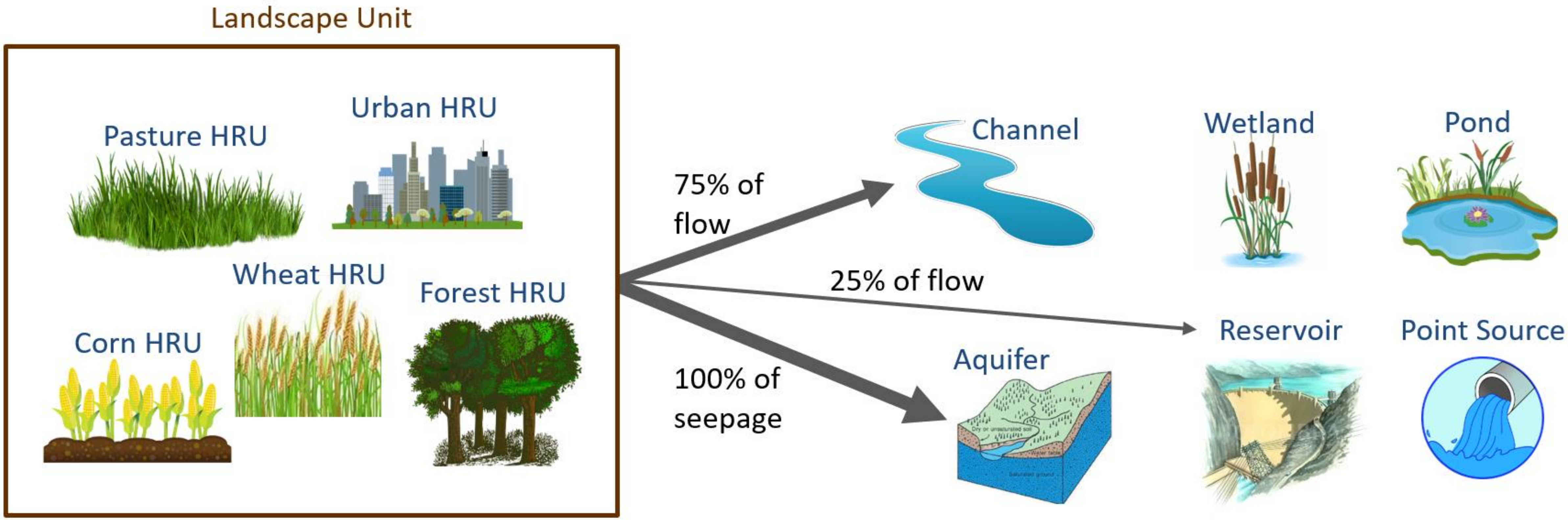
Goals of this Webinar:

- Get to know the modeling approach in OPTAIN
- Get aware of most relevant and innovative modeling tools and workflows
  - ➢ Why needed?
  - ➢ How to use?
  - ➢ Inspire you to use them!

Modeling challenge #1

Be sufficiently spatial with your processes!

Objects in SWAT+

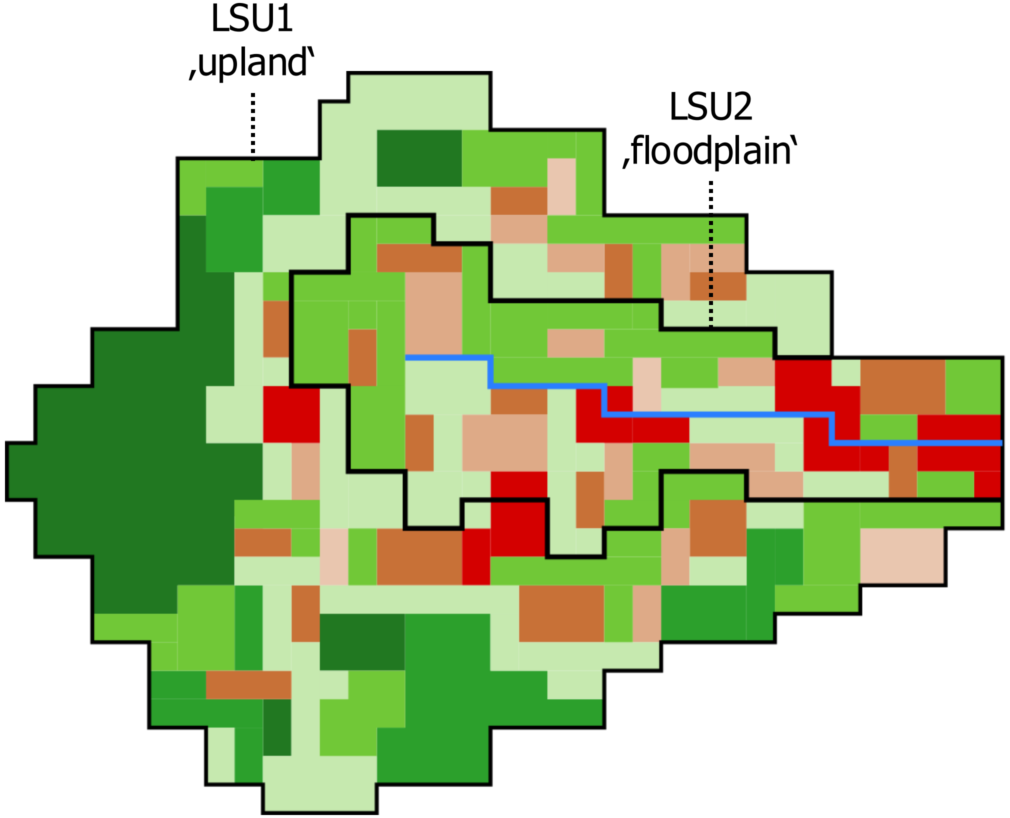


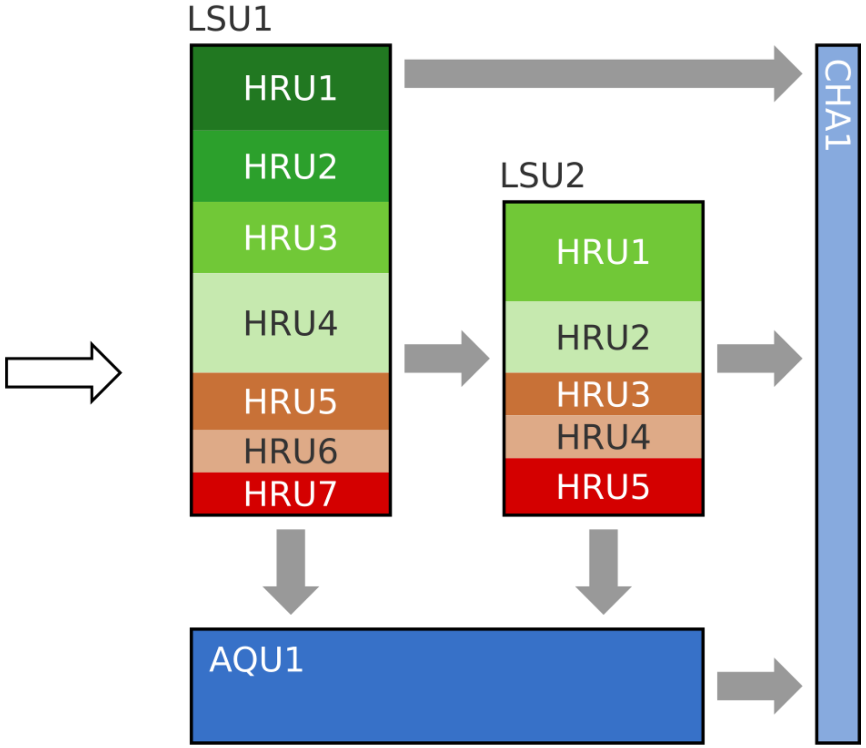
(source: Katrin Bieger)

- Units in a SWAT+ model setup with specific hydrologic functions
- Fluxes between objects are defined through their connectivity

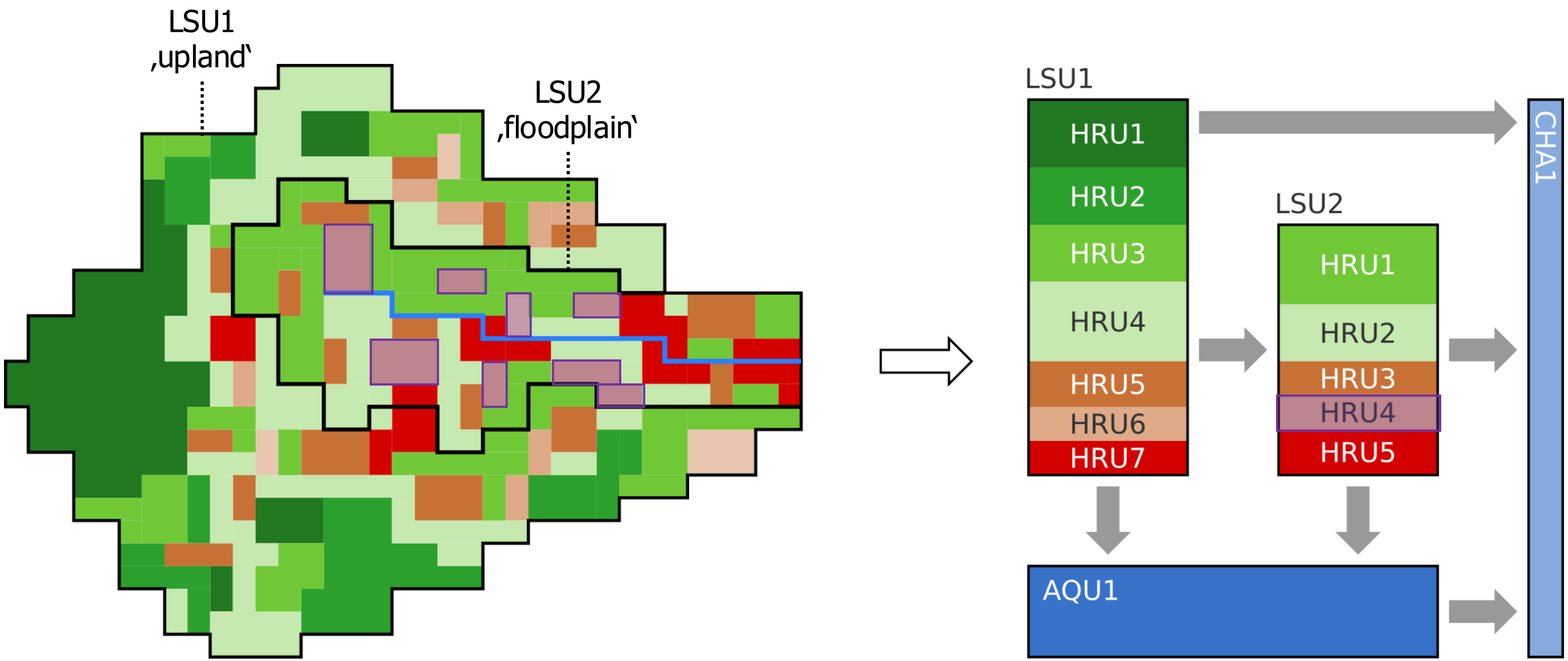
A conventional model setup - Default configuration of object connectivity with QSWAT+

HRU outputs are lumped within landscape units (LSU)



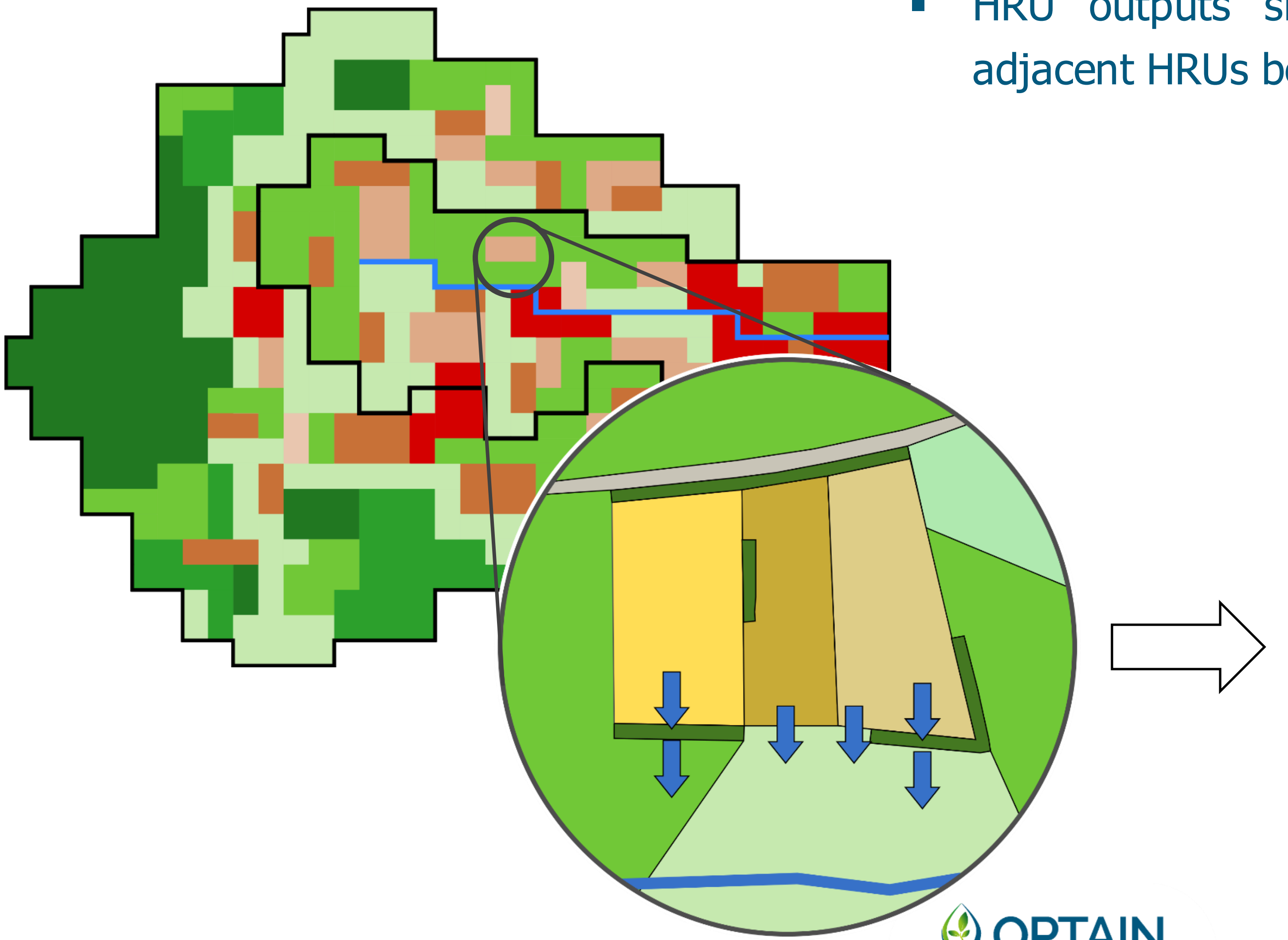


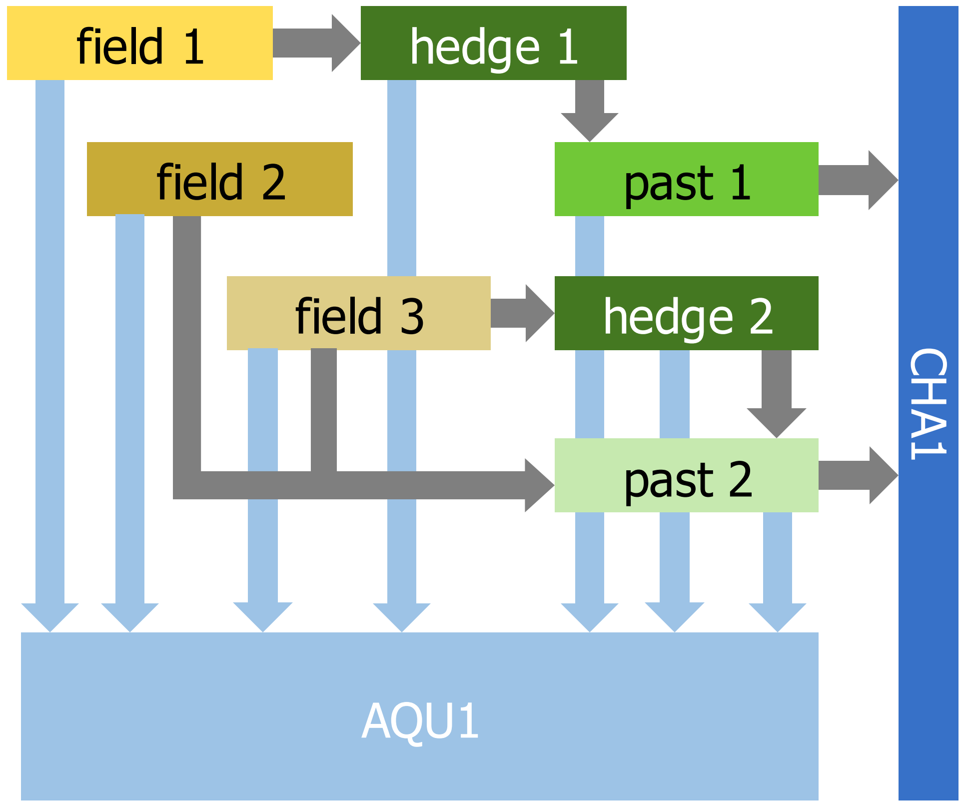
A conventional model setup – HRUs are usually fragmented units



OPTAIN requires a closer look to field-scale effects

- HRUs should be self-contained units
- HRU outputs should be routed through a cascade of adjacent HRUs before being fed into the channel/reservoir



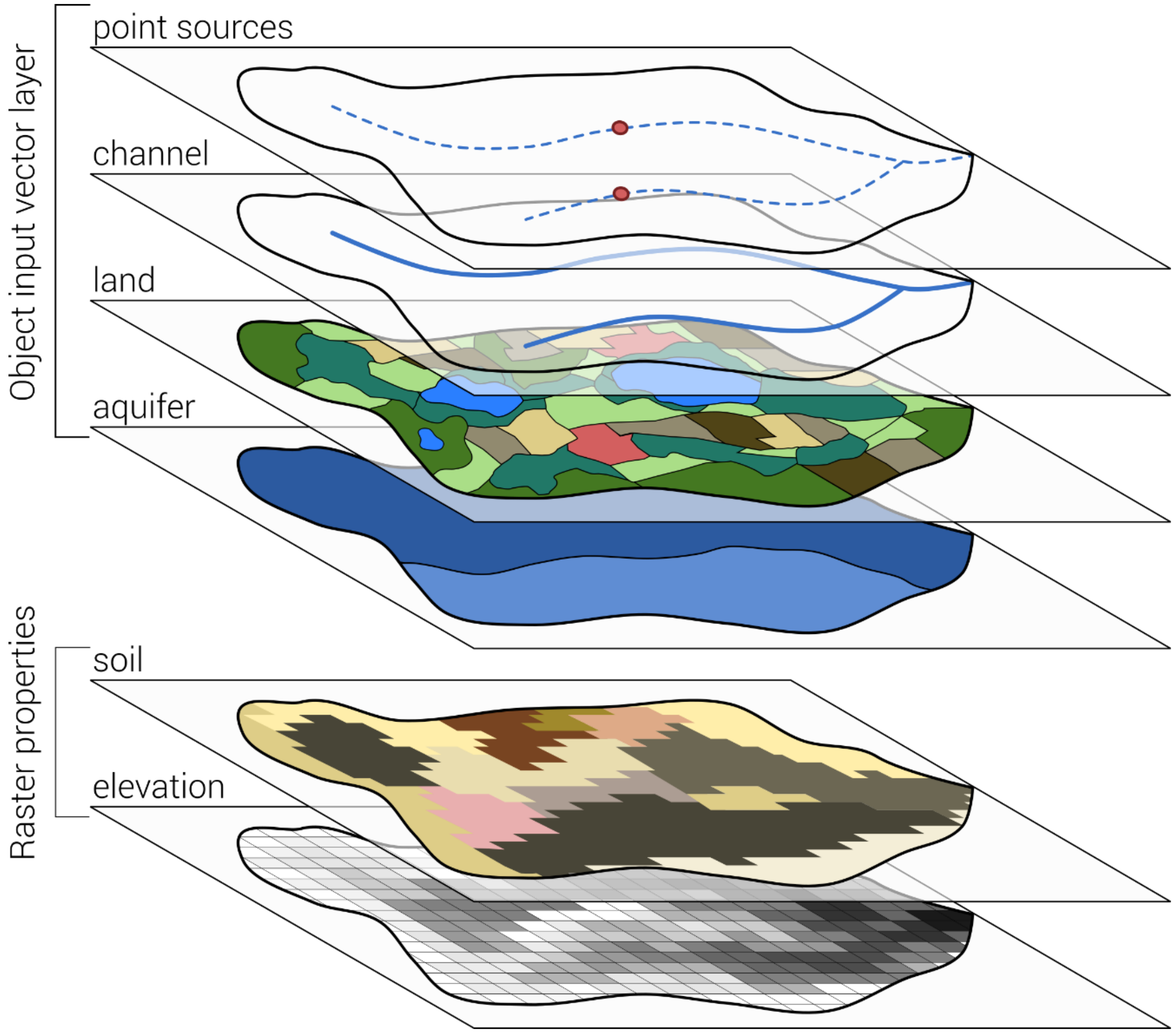


Our solution…

SWATbuildR

An object connectivity based SWAT+ model builder

SWATbuildR - Input data

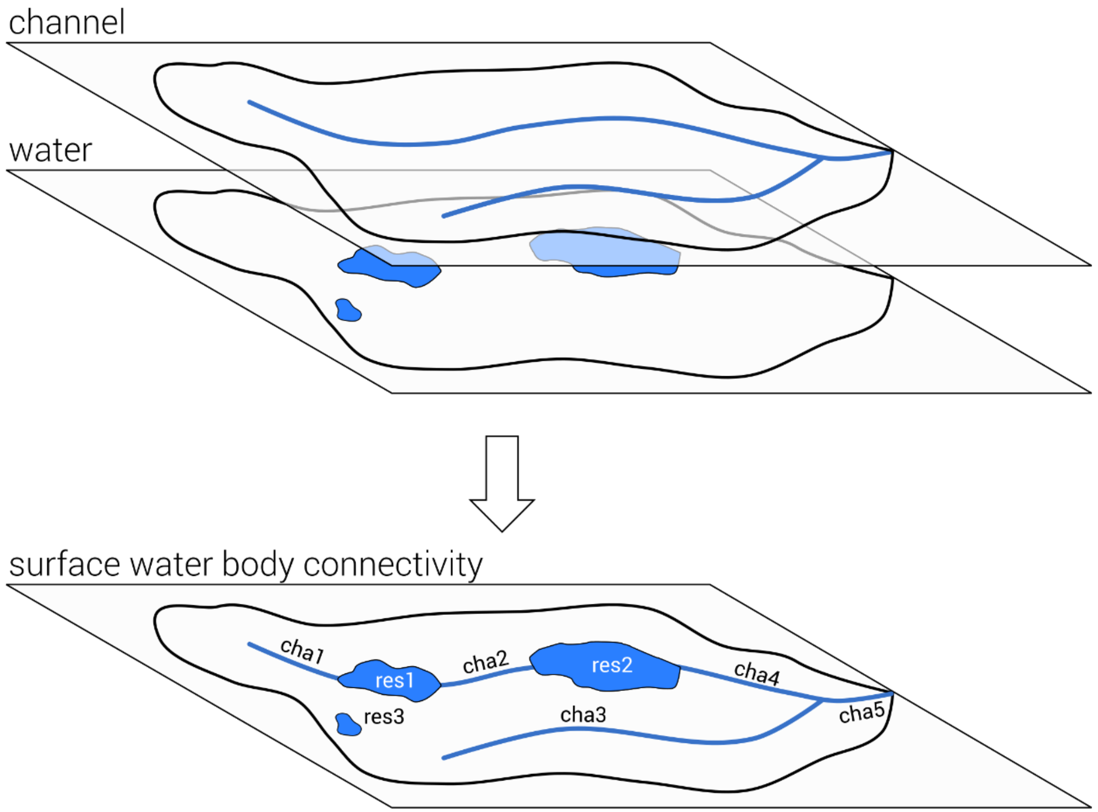


- Objects are organized in vector layers
- Each polygon in the input layers will be a unique object in the SWAT+ model setup
- Attribute tables require id column and a type column

- Soil layer with SWAT soil input table required

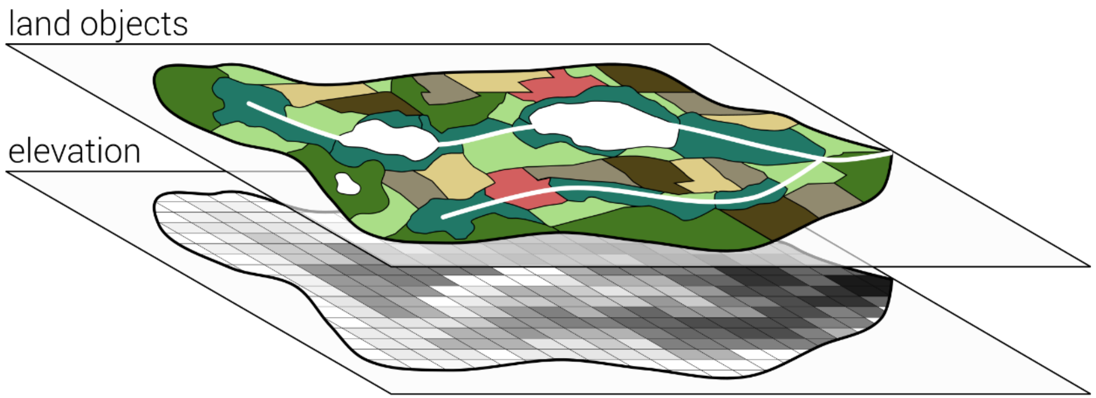
  → Dominant soil is assigned to land units
- DEM required to calculate the land object connectivity

SWATbuildR - Surface water object connectivity



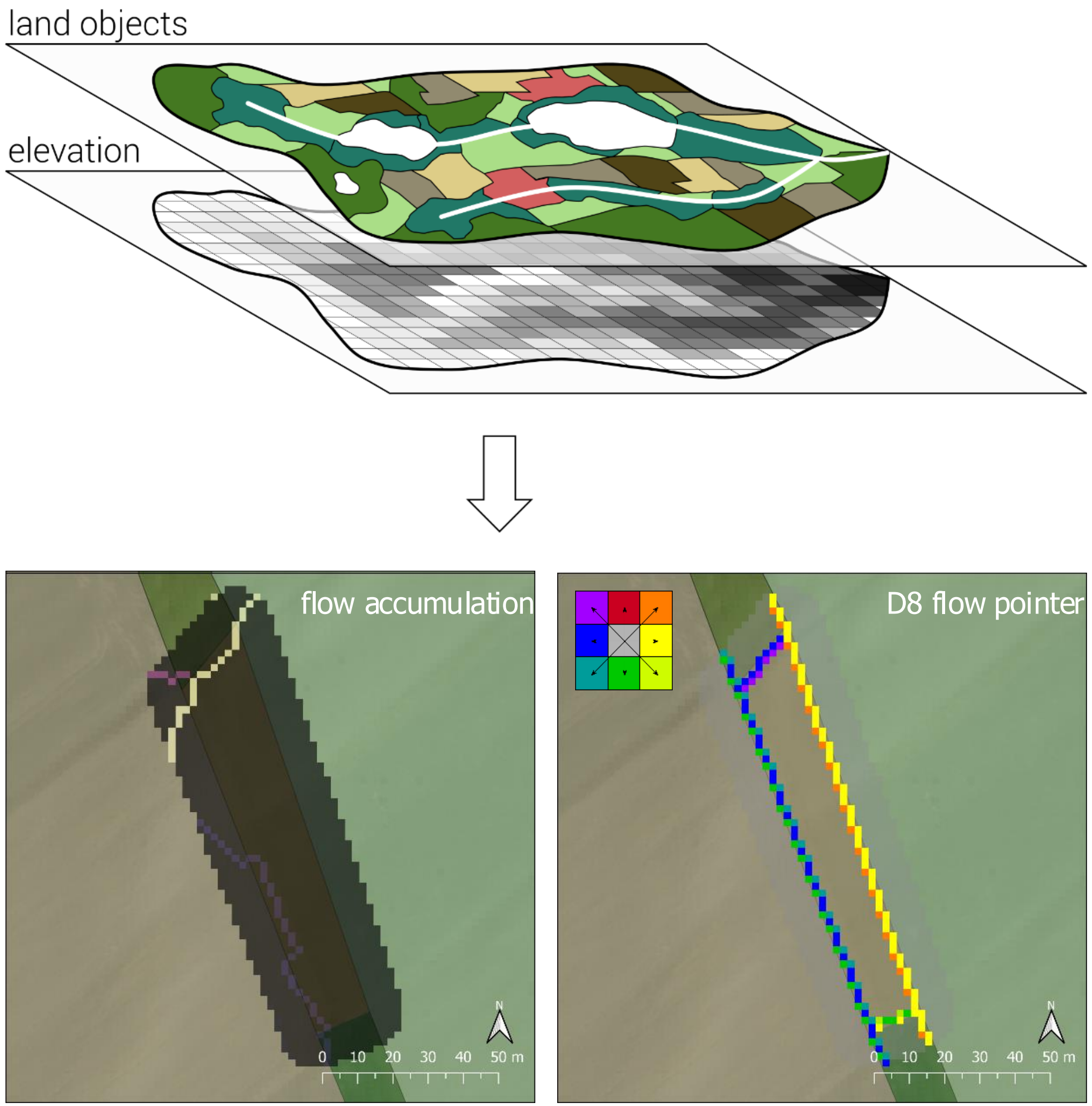
- Surface water body network is generated from channels and water objects in the land layer
- User defines outlet channel or reservoir id
- Connectivity network is back propagated from this object
- This approach allows branching rivers and multiple (and no) inlets in reservoirs

SWATbuildR - Land object connectivity



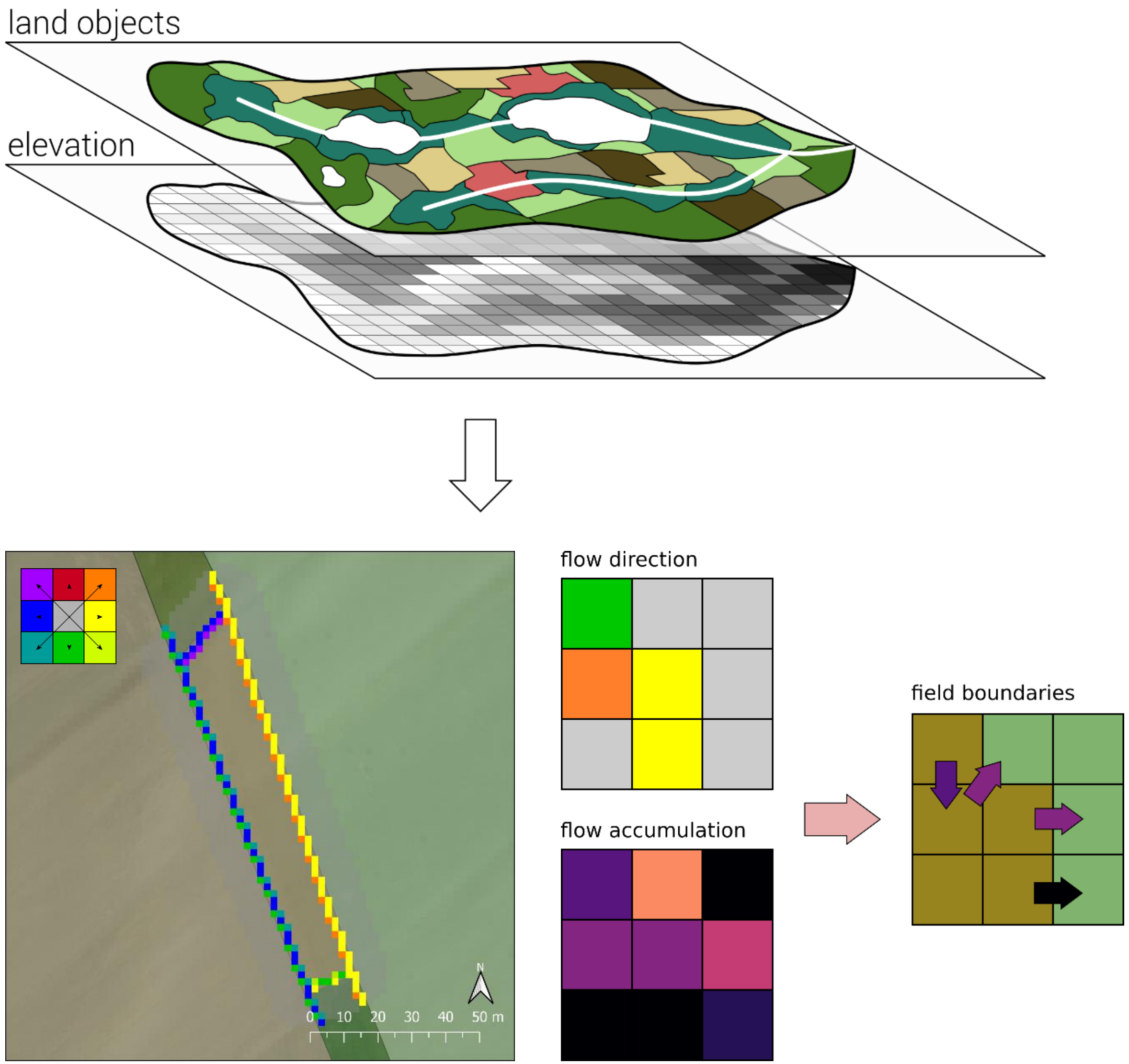
- Routine iterates over all land objects
- Water objects are excluded and burnt in (routed water will end up in water objects)

SWATbuildR - Land object connectivity



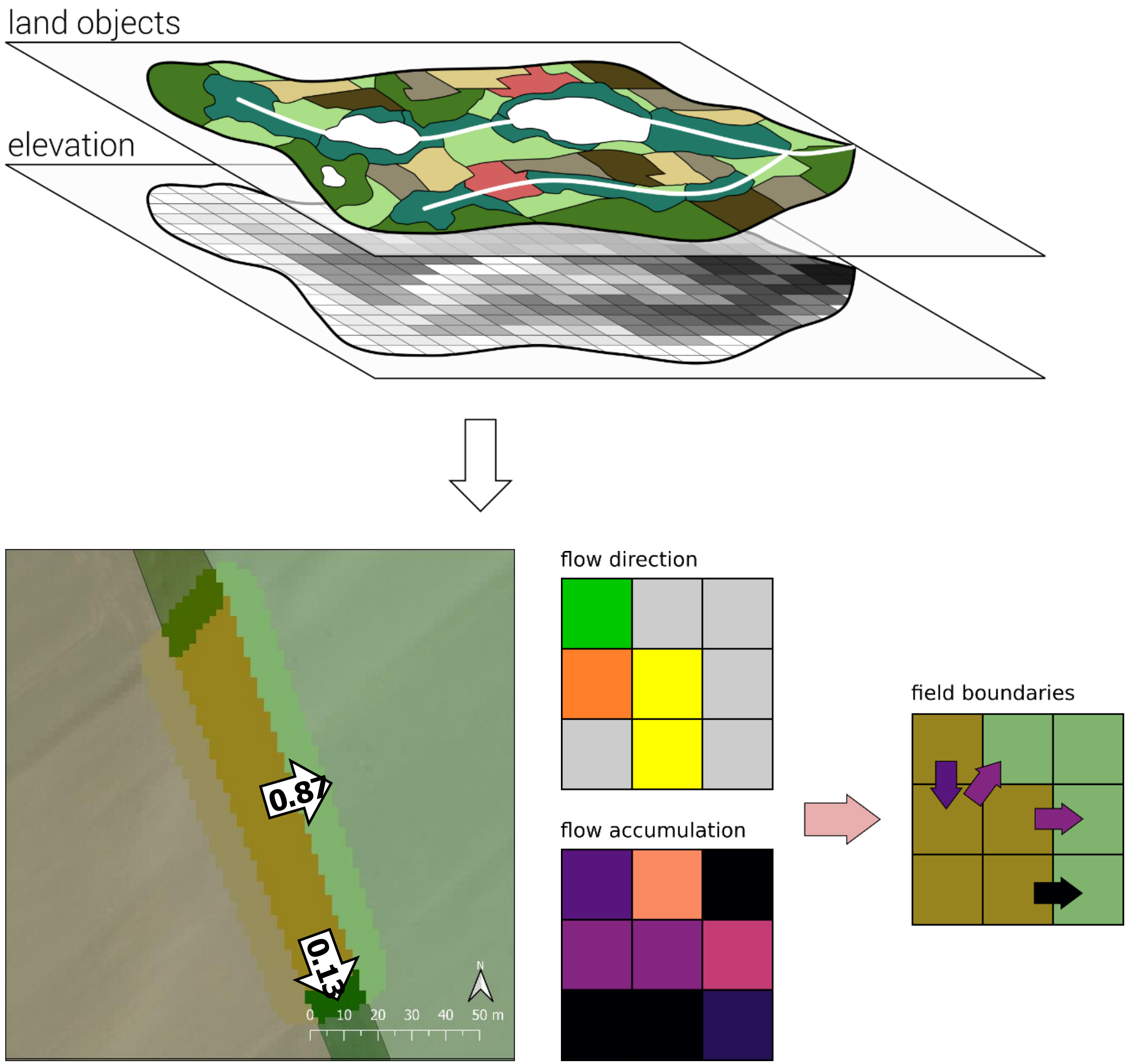
- Routine iterates over all land objects
- Water objects are excluded and burnt in (routed water will end up in water objects)
- For each object, flow accumulation and flow direction to neighbor objects is calculated

SWATbuildR - Land object connectivity



- Routine iterates over all land objects
- Water objects are excluded and burnt in (routed water will end up in water objects)
- For each object, flow accumulation and flow direction to neighbor objects is calculated
- Net sum of flow that crosses a boundary determines the flow fraction of sur and lat flow into neighbor object

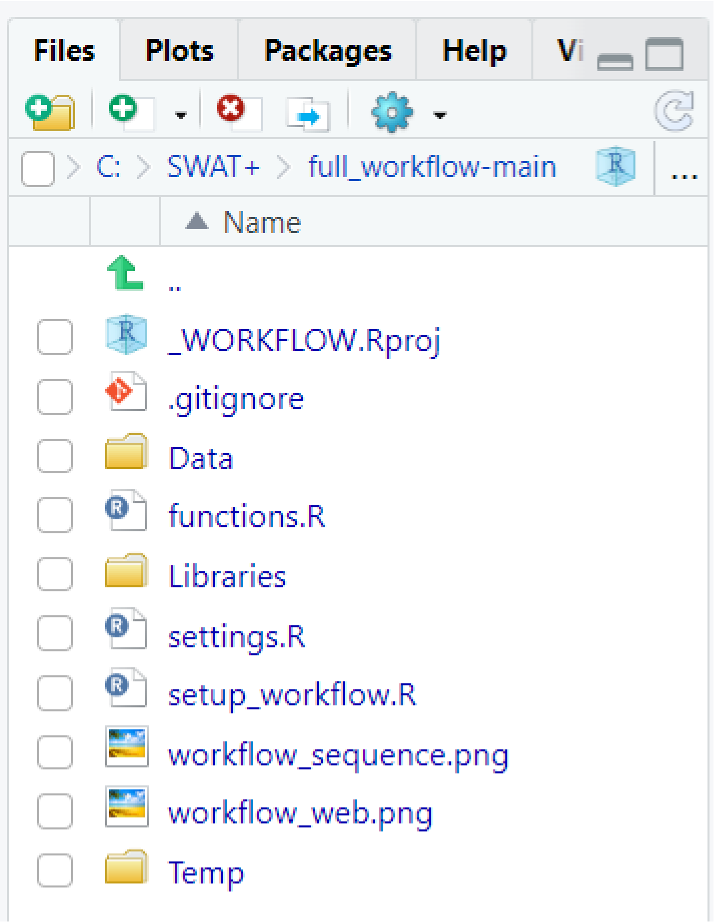
SWATbuildR - Land object connectivity



- Routine iterates over all land objects
- Water objects are excluded and burnt in (routed water will end up in water objects)
- For each object, flow accumulation and flow direction to neighbor objects is calculated
- Net sum of flow that crosses a boundary determines the flow fraction of sur and lat flow into neighbor object

SWATbuildR – How to use?

(1) Download full OPTAIN R-workflow (https://zenodo.org/records/12564534) or wait for stand-alone R-package on GitHub (will be published soon)



(2) Prepare inputs (DEM, soils, land use, channels, point sources)

⇒ see Data/for_buildr

(3) Open _WORKFLOW.Rproj in RStudio

(4) Open settings.R

⇒ in SWATbuildR section, adjust paths to your input data

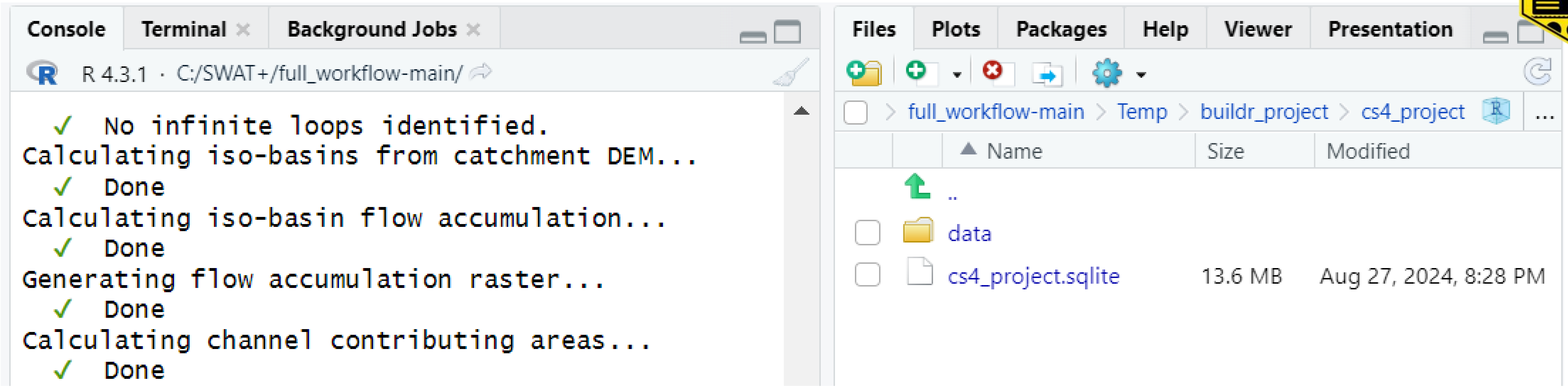
⇒ define channel or reservoir id at outlet and a few more things

(5) Open and run setup_workflow.R (until line 42)

⇒ In case of errors, follow advice of the error message

⇒ If the problem cannot be solved, post on GitHub or drop an email

SWATbuildR – Result



⇒ basic SWAT+ setup with connectivities defined for each land and water object, without resulting in infinite loops

⇒ further input data supply and parameterisation necessary

- sqlite database of your SWAT+ project

⇒ you can import to SWATEditor or just continue with the full workflow (setup_workflow.R)

- data folder including…

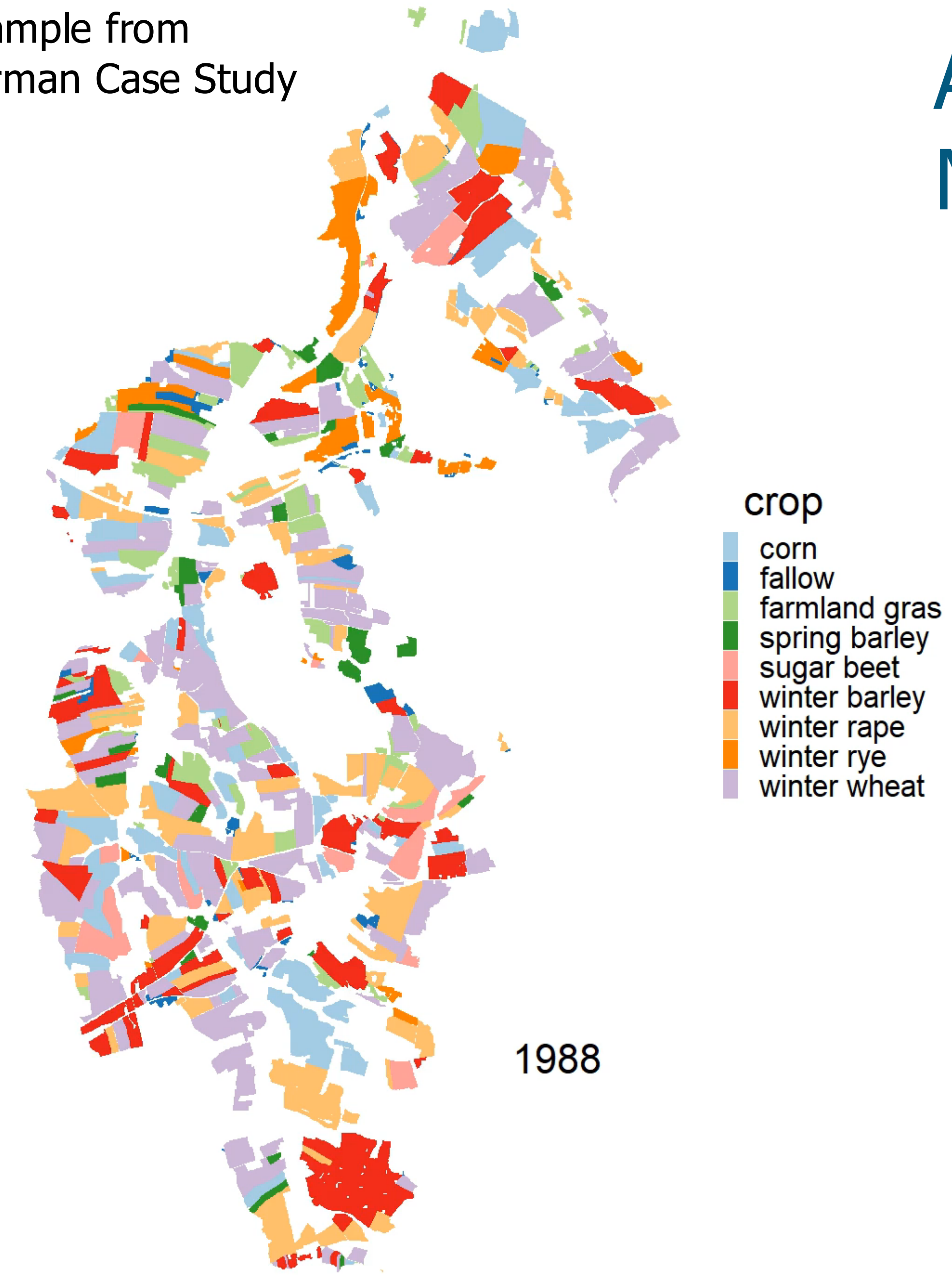
⇒ raster: soil.tif and lots of DEM processing results

⇒ vector: basin, channel, reservoir, hru, etc. shapefiles

Modeling challenge #3

Be sufficiently detailed with your crop management!

Example from
German Case Study



Agricultural management – Model setup requirements in OPTAIN

- A representative crop management needs to be defined for each individual field for long simulation periods (30 years + warm-up)
- Decision tables would blow up computation time (thus fixed operation dates need to be defined)
- Operation dates must account for rainfall (e.g. no fertiliser application on a rainy day)

⇒ A workflow is needed to automate the writing of long and consistent management operation schedules

One part of the solution…

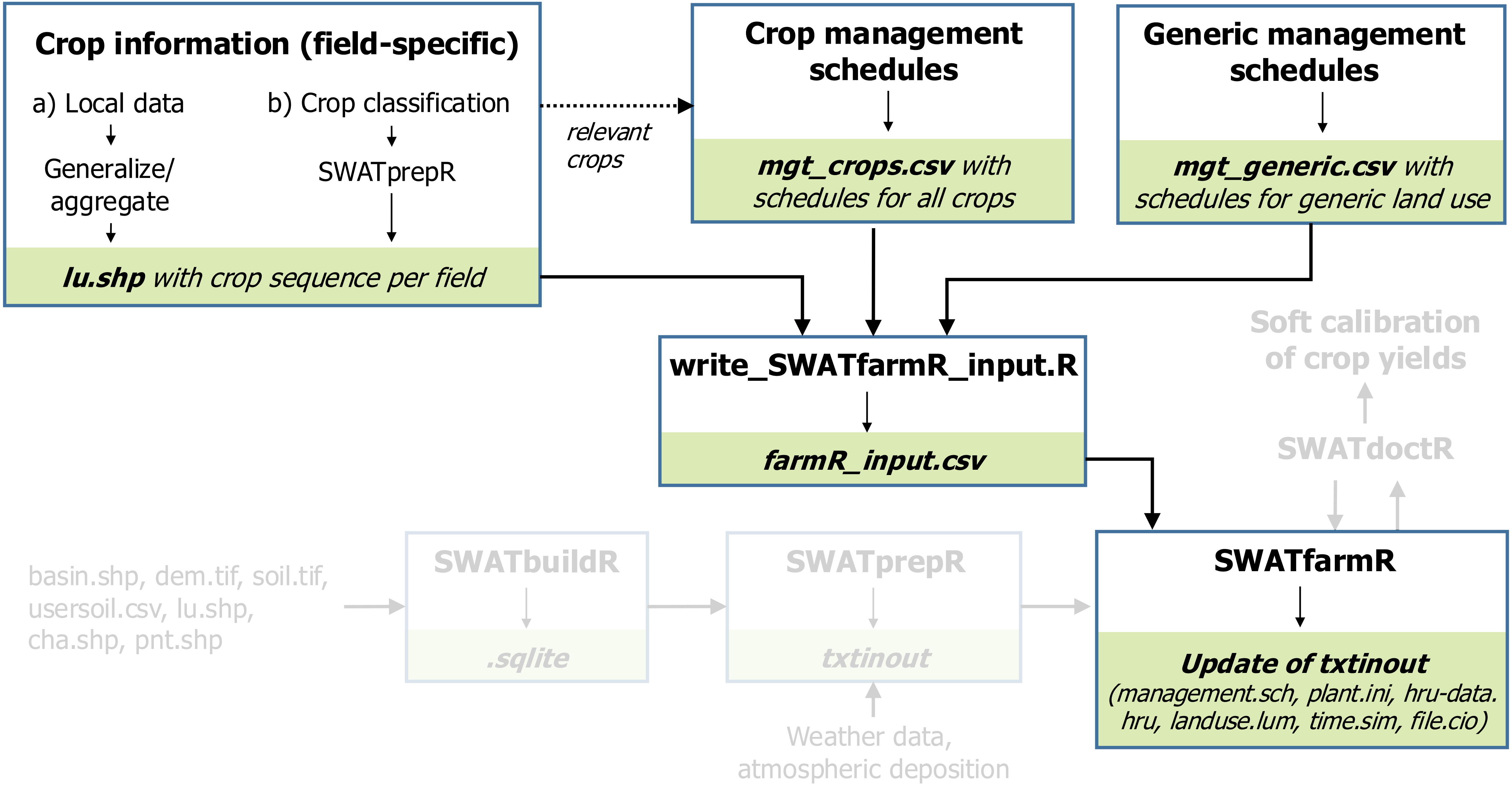
SWATfarmR

Simple rule based management operation scheduling

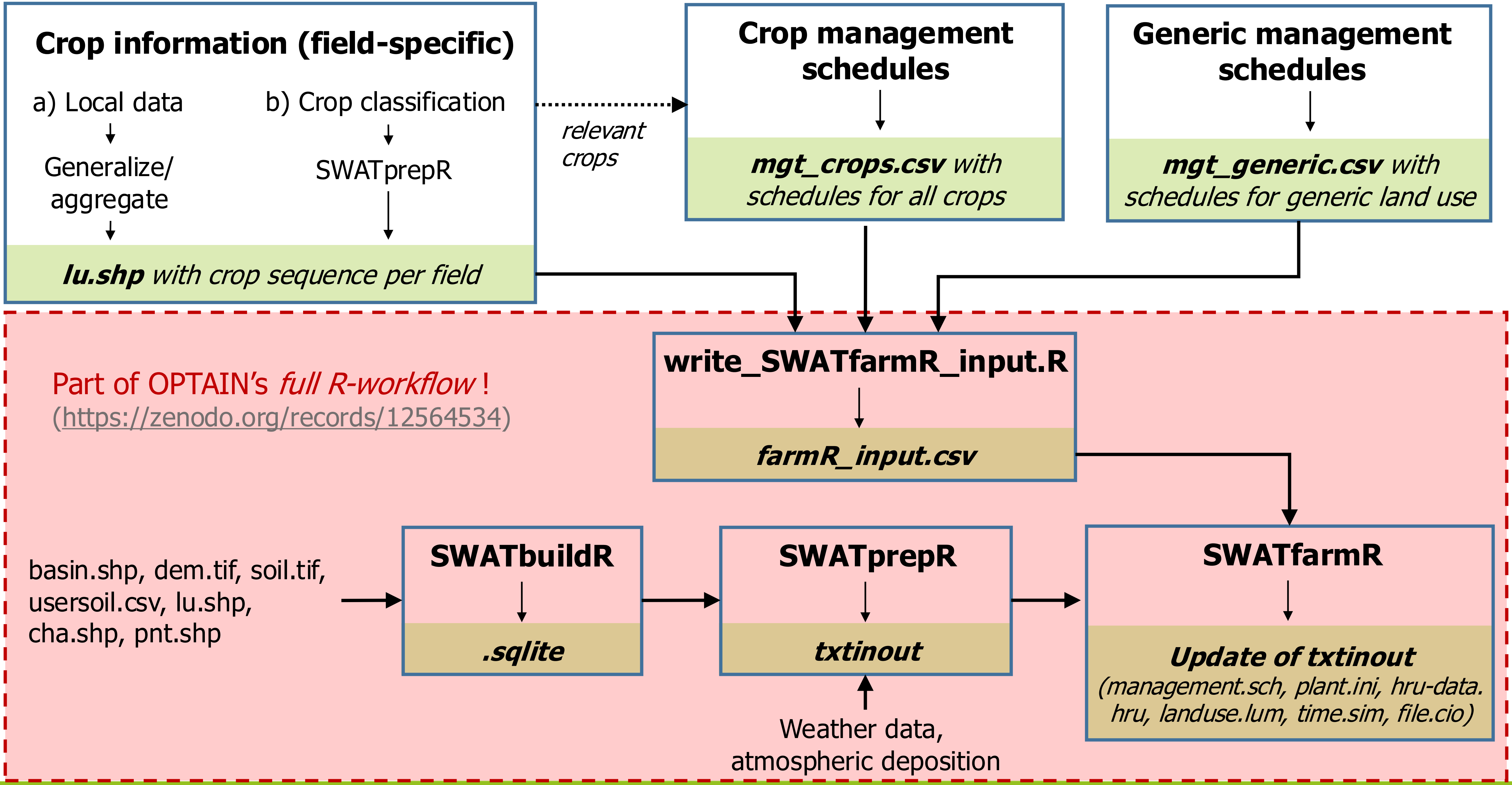
https://chrisschuerz.github.io/SWATfarmR/

‚The SWATfarmR is a pre-processing tool for the scheduling of farm management operations for SWAT+ and SWAT2012 projects. SWATfarmR develops management schedules for each HRU of a SWAT model setup based on user defined management tables. The user can define rules that control the timing of operations. These rules can include information on temporal constraints, an HRU's spatial properties, or any climatic or other external variable to control the scheduling of an operation.'

The full solution/workflow…



The full solution/workflow…



Preparing the input for the SWATfarmR

(1) You need a land use shapefile with a sequence of crops for each field and for each year of the simulation period

Crop sequence (observed):

Option A:
Local data (if available)

Option B:
OPTAIN crop classification

a) landuse.shp

extrapolate / repeat sequence

retrieve crop sequence

lu	y_1988	y_1989	...	y_2008	y_2009	y_2010	y_2011	y_2012	y_2013	y_2014	y_2015	y_2016	y_2017	y_2018	y_2019	y_2020
field_1	csil	wwht	...	wbar	wira	wwht	wbar	csil	wwht	wbar	wira	wwht	wbar	csil	wwht	wbar
field_2	wira	wwht	...	wira	akgs	akgs	wwht	wira	wwht	wira	akgs	akgs	wwht	wira	wwht	wira
...			...													

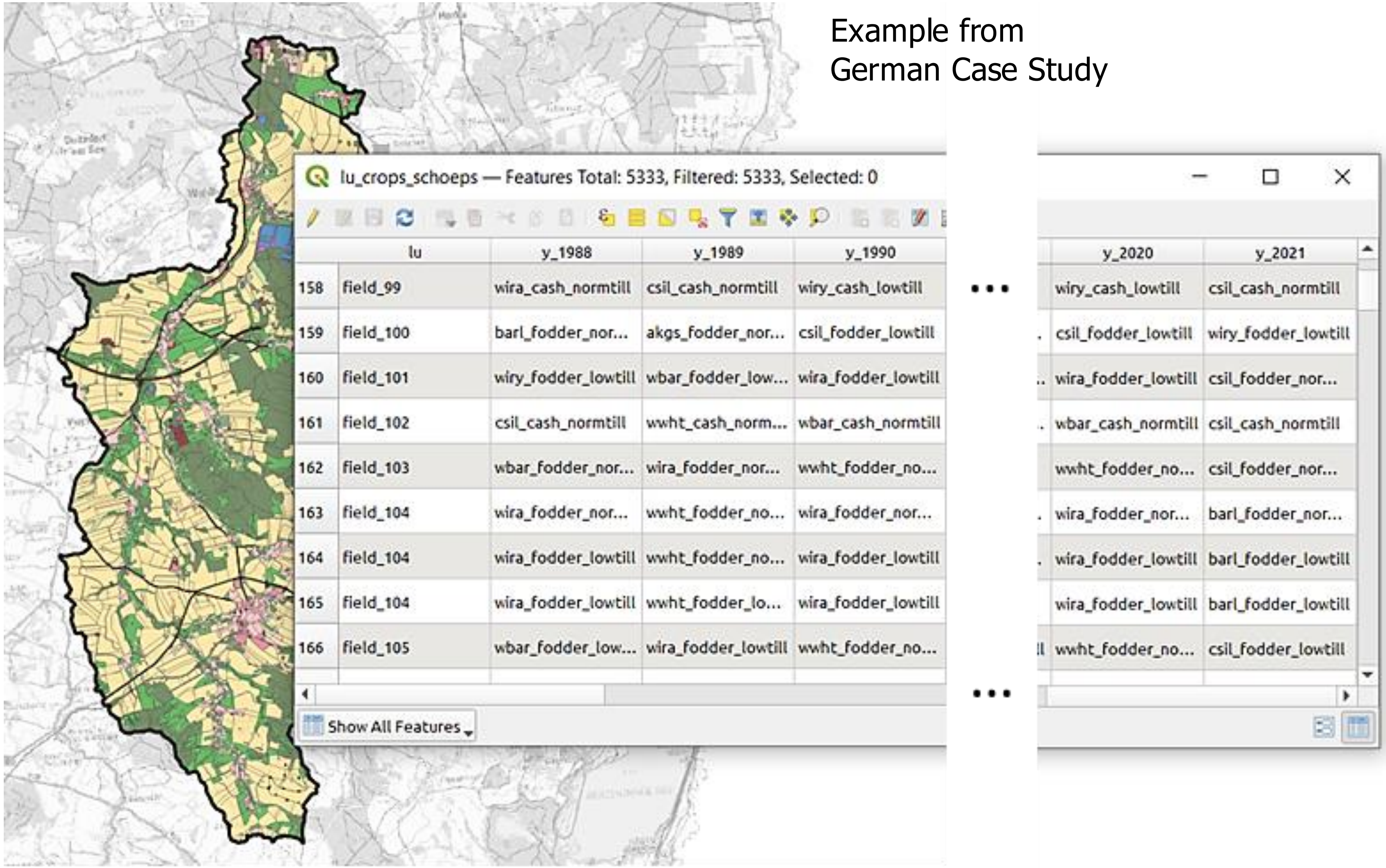
if required fill gaps
with plausible crops

For more details Schürz et al. (2022) https://doi.org/10.5281/zenodo.7463395

OPTAIN

Preparing the input for the SWATfarmR

Example from
German Case Study



lu_crops_schoeps — Features Total: 5333, Filtered: 5333, Selected: 0

	lu	y_1988	y_1989	y_1990	…	y_2020	y_2021
158	field_99	wira_cash_normtill	csil_cash_normtill	wiry_cash_lowtill		wiry_cash_lowtill	csil_cash_normtill
159	field_100	barl_fodder_nor...	akgs_fodder_nor...	csil_fodder_lowtill		csil_fodder_lowtill	wiry_fodder_lowtill
160	field_101	wiry_fodder_lowtill	wbar_fodder_low...	wira_fodder_lowtill		wira_fodder_lowtill	csil_fodder_nor...
161	field_102	csil_cash_normtill	wwht_cash_norm...	wbar_cash_normtill		wbar_cash_normtill	csil_cash_normtill
162	field_103	wbar_fodder_nor...	wira_fodder_nor...	wwht_fodder_no...		wwht_fodder_no...	csil_fodder_nor...
163	field_104	wira_fodder_nor...	wwht_fodder_no...	wira_fodder_nor...		wira_fodder_nor...	barl_fodder_nor...
164	field_104	wira_fodder_lowtill	wwht_fodder_no...	wira_fodder_lowtill		wira_fodder_lowtill	barl_fodder_lowtill
165	field_104	wira_fodder_lowtill	wwht_fodder_lo...	wira_fodder_lowtill		wira_fodder_lowtill	barl_fodder_lowtill
166	field_105	wbar_fodder_low...	wira_fodder_lowtill	wwht_fodder_no...		wwht_fodder_no...	csil_fodder_lowtill

Show All Features

Preparing the input for the SWATfarmR

(2) You need typical 1-year management schedules for single crops and generic land-use classes

b) crop_mgt.csv

	A	B	C	D	E	F	G	H	I
1	crop_mgt	mon_1	day_1	mon_2	day_2	operation	op_data1	op_data2	op_data3
2	wwht_cash_normtill	9	15	10	7	fertilizer	elem_p	broadcast	25.4
3	wwht_cash_normtill	9	16	10	8	tillage	cultiv25		
4	wwht_cash_normtill	9	24	10	9	tillage	harrow7		
5	wwht_cash_normtill	9	25	10	10	plant	wwht		
6	wwht_cash_normtill					skip			
7	wwht_cash_normtill	3	3	3	17	fertilizer	elem_n	broadcast	77.7
8	wwht_cash_normtill	4	23	5	7	fertilizer	elem_n	broadcast	50
9	wwht_cash_normtill	5	25	6	8	fertilizer	elem_n	broadcast	15
10	wwht_cash_normtill	7	25	8	17	harvest_only	wwht	grain	
11	wwht_cash_normtill	7	25	8	17	kill_only	wwht		
12	wwht_cash_normtill	7	26	8	19	tillage	fldcul10		
13	csil_fodder_lowtill	10	8	10	22	tillage	fldcul12		
14	csil_fodder_lowtill					skip			
15	csil_fodder_lowtill	4	14	4	28	fertilizer	beefg_fl	aerial_liquid	40000
16	csil_fodder_lowtill	4	14	4	28	fertilizer	elem_n	broadcast	15
17	csil_fodder_lowtill	4	14	4	28	fertilizer	elem_p	broadcast	15
18	csil_fodder_lowtill	4	15	4	29	tillage	harrow8		
19	csil_fodder_lowtill	4	17	5	1	plant	csil		
20	csil_fodder_lowtill	9	8	9	22	harvest_only	csil	silage	
21	csil_fodder_lowtill	9	8	9	22	kill_only	csil		
22	csil_fodder_lowtill	9	20	10	3	tillage	fldcul10		

Full schedule crop A (rows 2–12)

Full schedule crop B (rows 13–22)

...plus all other individual schedules for crops occuring in the sequences provided with the map (the order of crops does not matter)

c) generic_mgt.csv

	A	B	C	D	E	F	G	H	I
1	lulc_mgt	mon_1	day_1	mon_2	day_2	operation	op_data1	op_data2	op_data3
2	meadow_4cuts					initial_plant	fesc_comm	1,1000,0,0,1,1000	
3	meadow_4cuts	3	1	3	31	fertilizer	elem_n	broadcast	70
4	meadow_4cuts	3	1	3	31	fertilizer	elem_p	broadcast	25
5	meadow_4cuts	5	5	5	10	harvest_only	fesc	hay_cut_low	
6	meadow_4cuts	5	11	5	15	fertilizer	elem_n	broadcast	60
7	meadow_4cuts	6	12	6	23	harvest_only	fesc	hay_cut_low	
8	meadow_4cuts	6	25	6	30	fertilizer	elem_n	broadcast	40
9	meadow_4cuts	7	25	8	5	harvest_only	fesc	hay_cut_low	
10	meadow_4cuts	9	15	9	30	harvest_only	fesc	hay_cut_low	
11	meadow_4cuts	10	15	10	30	fertilizer	beefg_fl	aerial_liquid	25000
12	meadow_3cuts					initial_plant	fesc_comm	1,1000,0,0,1,1000	
13	meadow_3cuts	3	1	3	31	fertilizer	elem_n	broadcast	60
14	meadow_3cuts	3	1	3	31	fertilizer	elem_p	broadcast	25
15	meadow_3cuts	5	15	5	25	harvest_only	fesc	hay_cut_low	
16	meadow_3cuts	5	27	6	5	fertilizer	elem_n	broadcast	40
17	meadow_3cuts	7	25	8	5	harvest_only	fesc	hay_cut_low	
18	meadow_3cuts	8	7	8	12	fertilizer	elem_n	broadcast	40
19	meadow_3cuts	9	15	9	30	harvest_only	fesc	hay_cut_low	
20	meadow_3cuts	10	15	10	30	fertilizer	beefg_fl	aerial_liquid	15000
21	meadow_2cuts					initial_plant	fesc_comm	1,1000,0,0,1,1000	
22	meadow_2cuts	3	1	3	31	fertilizer	elem_n	broadcast	60
23	meadow_2cuts	3	1	3	31	fertilizer	elem_p	broadcast	25
24	meadow_2cuts	5	25	6	5	harvest_only	fesc	hay_cut_low	
25	meadow_2cuts	6	7	6	15	fertilizer	elem_n	broadcast	40
26	meadow_2cuts	8	10	8	25	harvest_only	fesc	hay_cut_low	
27	orcd					initial_plant	orcd_comm	2,20000,0,0,1,10000	
28	orcd	9	1	10	31	harvest_only	orcd	orchard	
29	frst					initial_plant	frst_comm	2,50000,0,0,1,10000	
30	wetl					initial_plant	wetl_comm	2,50000,0,0,1,10000	
31	rngb					initial_plant	rngb_comm	1,1000,0,0,1,1000	
32	rnge					initial_plant	rnge_comm	1,1000,0,0,1,1000	
33	bsvg					initial_plant	bsvg_comm	0.1,10,0,0,1,10	

For more details Schürz et al. (2022) https://doi.org/10.5281/zenodo.7463395

Preparing the input for the SWATfarmR

a) landuse.shp

b) crop_mgt.csv

c) generic_mgt.csv

Run write_SWATfarmR_input.R to (1) combine the schedules according to the sequences for each field

(2) solve date conflicts (overlaps) in the combined schedules

(3) write the combined schedules into SWATfarmR input format

write_SWATfarmR_input.R

Source on Save

```r

# Check for correct positioning of 'skip' line ----------------------------------
check_skip <- check_skip_position()

# Check for date conflicts within single crop schedules -----------------------
check_date_conflicts1()

# Build schedules for crop sequences ------------------------------------------
rota_schedules <- build_rotation_schedules()

# Check for date conflicts in combined (rotation) schedule --------------------
check_date_conflicts2()

# Solve minor date conflicts (where only a few days/weeks are overlapping)------
rota_schedules <- solve_date_conflicts()

## check again for date conflicts ----------------------------------------------
check_date_conflicts2()

## write the SWAT farmR input table --------------------------------------------
write_farmR_input()

```

d) farmR_input.csv

For more details Schürz et al. (2022) https://doi.org/10.5281/zenodo.7463395

Preparing the input for the SWATfarmR

d) farmR_input.csv

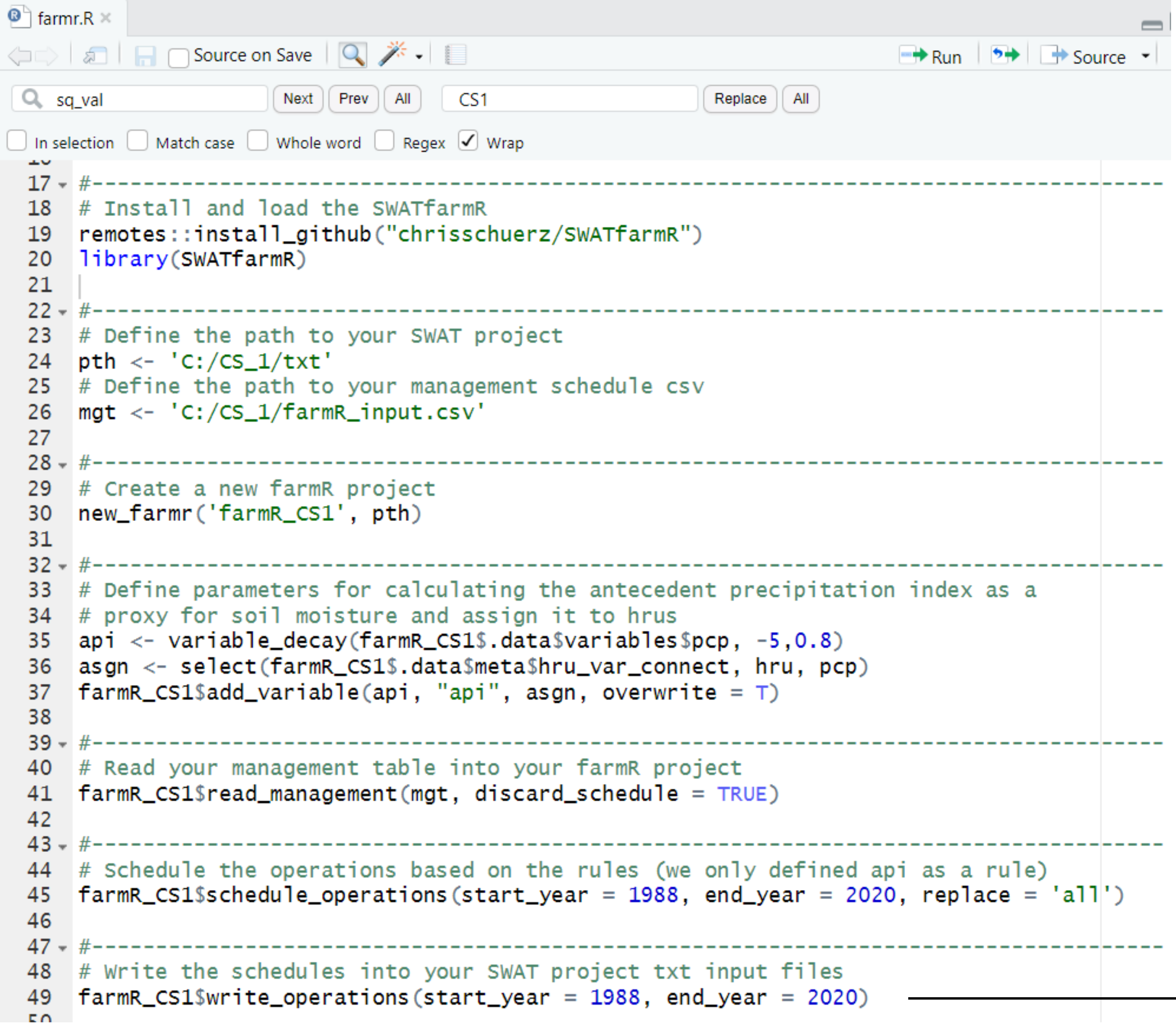
	A	B	C	D	E	F	G	H	I
1	land_use	management	weight	filter_attribute	condition_schedule	operation	op_data1	op_data2	op_data3
2	field_1_lum					initial_plant	wbar	1,1000,0,0,1,1000	
3	field_1_lum				(md >= 0215) * (md <= 0305) * (1 - w_log(pcp, 0, 7)) * (1 - w_log(api, 5, 20)) * (year == (year(prev_op) + 1))	fertilizer	beefg_fl	aerial_liquid	21000
4	field_1_lum				(md >= 0325) * (md <= 0410) * (1 - w_log(pcp, 0, 7)) * (1 - w_log(api, 5, 20)) * (year == (year(prev_op)))	fertilizer	elem_n	broadcast	68.145
5	field_1_lum				(md >= 0701) * (md <= 0720) * (1 - w_log(pcp, 0, 7)) * (1 - w_log(api, 5, 20)) * (year == (year(prev_op)))	harvest_only	wbar	grain	
6	field_1_lum					kill_only	wbar		
7	field_1_lum				(md >= 0710) * (md <= 0731) * (1 - w_log(pcp, 0, 7)) * (1 - w_log(api, 5, 20)) * (year == (year(prev_op)))	tillage	fldcul15		
8	field_1_lum				(md >= 0805) * (md <= 0905) * (1 - w_log(pcp, 0, 7)) * (1 - w_log(api, 5, 20)) * (year == (year(prev_op)))	tillage	harrow5		
9	field_1_lum				(md >= 0806) * (md <= 0906) * (1 - w_log(pcp, 0, 7)) * (1 - w_log(api, 5, 20)) * (year == (year(prev_op)))	plant	radi		
10	field_1_lum				(md >= 0314) * (md <= 0328) * (1 - w_log(pcp, 0, 7)) * (1 - w_log(api, 5, 20)) * (year == (year(prev_op) + 1))	harvest_only	radi	grass_mulch	
11	field_1_lum					kill_only	radi		
12	field_1_lum				(md >= 0408) * (md <= 0422) * (1 - w_log(pcp, 0, 7)) * (1 - w_log(api, 5, 20)) * (year == (year(prev_op)))	tillage	fldcul12		
13	field_1_lum				(md >= 0414) * (md <= 0428) * (1 - w_log(pcp, 0, 7)) * (1 - w_log(api, 5, 20)) * (year == (year(prev_op)))	fertilizer	beefg_fl	aerial_liquid	40000
14	field_1_lum					fertilizer	elem_n	broadcast	15
15	field_1_lum					fertilizer	elem_p	broadcast	15
16	field_1_lum				(md >= 0415) * (md <= 0429) * (1 - w_log(pcp, 0, 7)) * (1 - w_log(api, 5, 20)) * (year == (year(prev_op)))	tillage	harrow8		
17	field_1_lum				(md >= 0417) * (md <= 0501) * (1 - w_log(pcp, 0, 7)) * (1 - w_log(api, 5, 20)) * (year == (year(prev_op)))	plant	csil		
18	field_1_lum				(md >= 0908) * (md <= 0922) * (1 - w_log(pcp, 0, 7)) * (1 - w_log(api, 5, 20)) * (year == (year(prev_op)))	harvest_only	csil	silage	
19	field_1_lum					kill_only	csil		
20	field_1_lum				(md >= 0920) * (md <= 1003) * (1 - w_log(pcp, 0, 7)) * (1 - w_log(api, 5, 20)) * (year == (year(prev_op)))	tillage	fldcul10		
21	field_1_lum				(md >= 0916) * (md <= 1008) * (1 - w_log(pcp, 0, 7)) * (1 - w_log(api, 5, 20)) * (year == (year(prev_op)))	tillage	fldcul12		
22	field_1_lum				(md >= 0924) * (md <= 1009) * (1 - w_log(pcp, 0, 7)) * (1 - w_log(api, 5, 20)) * (year == (year(prev_op)))	tillage	harrow7		
23	field_1_lum				(md >= 0925) * (md <= 1010) * (1 - w_log(pcp, 0, 7)) * (1 - w_log(api, 5, 20)) * (year == (year(prev_op)))	plant	wwht		
24	field_1_lum				(md >= 0303) * (md <= 0317) * (1 - w_log(pcp, 0, 7)) * (1 - w_log(api, 5, 20)) * (year == (year(prev_op) + 1))	fertilizer	beefg_fl	aerial_liquid	27500
25	field_1_lum				(md >= 0423) * (md <= 0507) * (1 - w_log(pcp, 0, 7)) * (1 - w_log(api, 5, 20)) * (year == (year(prev_op)))	fertilizer	elem_n	broadcast	88.99

Run SWATfarmR

For more details Schürz et al. (2022) https://doi.org/10.5281/zenodo.7463395

Run SWATfarmR

d) farmR_input.csv → Run SWATfarmR to schedule exact dates (so far we have time windows) and write SWAT+ files




```r
#------------------------------------------------------------------------------
# Install and load the SWATfarmR
remotes::install_github("chrisschuerz/SWATfarmR")
library(SWATfarmR)

#------------------------------------------------------------------------------
# Define the path to your SWAT project
pth <- 'C:/CS_1/txt'
# Define the path to your management schedule csv
mgt <- 'C:/CS_1/farmR_input.csv'

#------------------------------------------------------------------------------
# Create a new farmR project
new_farmr('farmR_CS1', pth)

#------------------------------------------------------------------------------
# Define parameters for calculating the antecedent precipitation index as a
# proxy for soil moisture and assign it to hrus
api <- variable_decay(farmR_CS1$.data$variables$pcp, -5,0.8)
asgn <- select(farmR_CS1$.data$meta$hru_var_connect, hru, pcp)
farmR_CS1$add_variable(api, "api", asgn, overwrite = T)

#------------------------------------------------------------------------------
# Read your management table into your farmR project
farmR_CS1$read_management(mgt, discard_schedule = TRUE)

#------------------------------------------------------------------------------
# Schedule the operations based on the rules (we only defined api as a rule)
farmR_CS1$schedule_operations(start_year = 1988, end_year = 2020, replace = 'all')

#------------------------------------------------------------------------------
# Write the schedules into your SWAT project txt input files
farmR_CS1$write_operations(start_year = 1988, end_year = 2020)
```

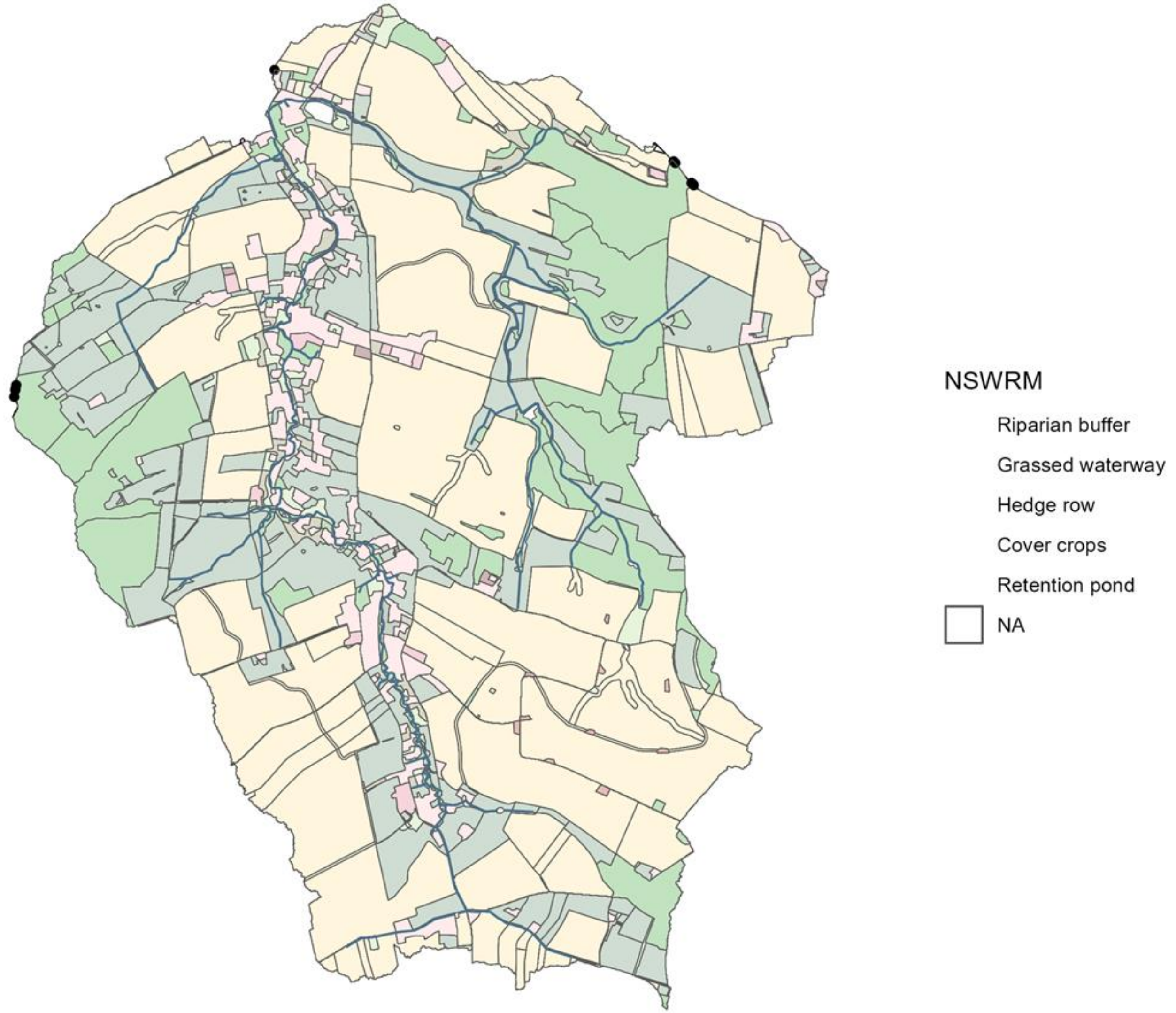
In your SWAT txt folder it rewrites:

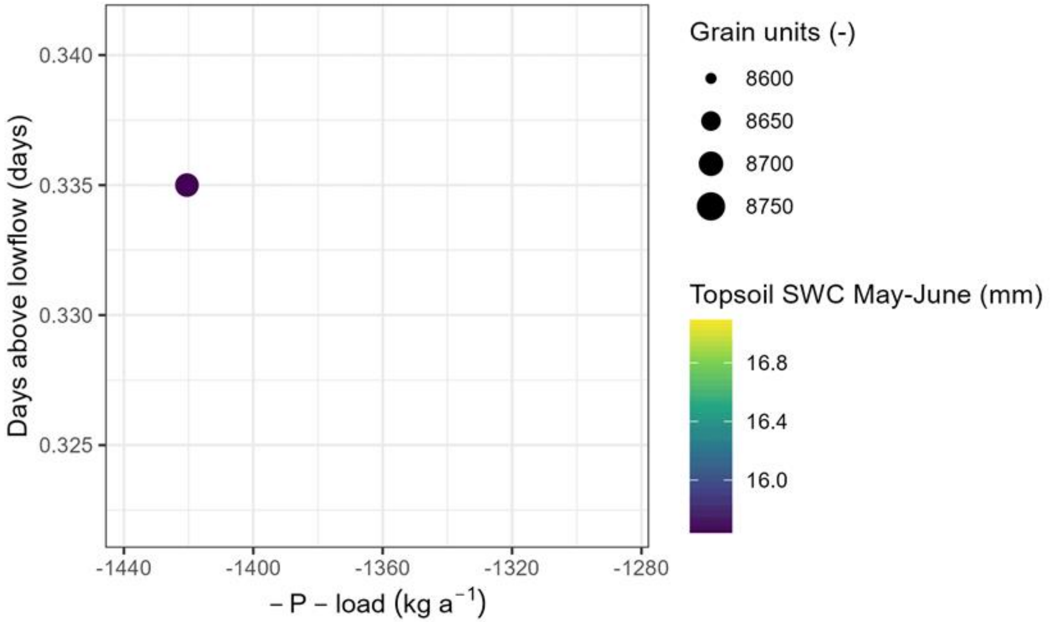
- management.sch
- hru-data.hru
- landuse.lum
- file.cio
- time.sim

Modeling challenge #5

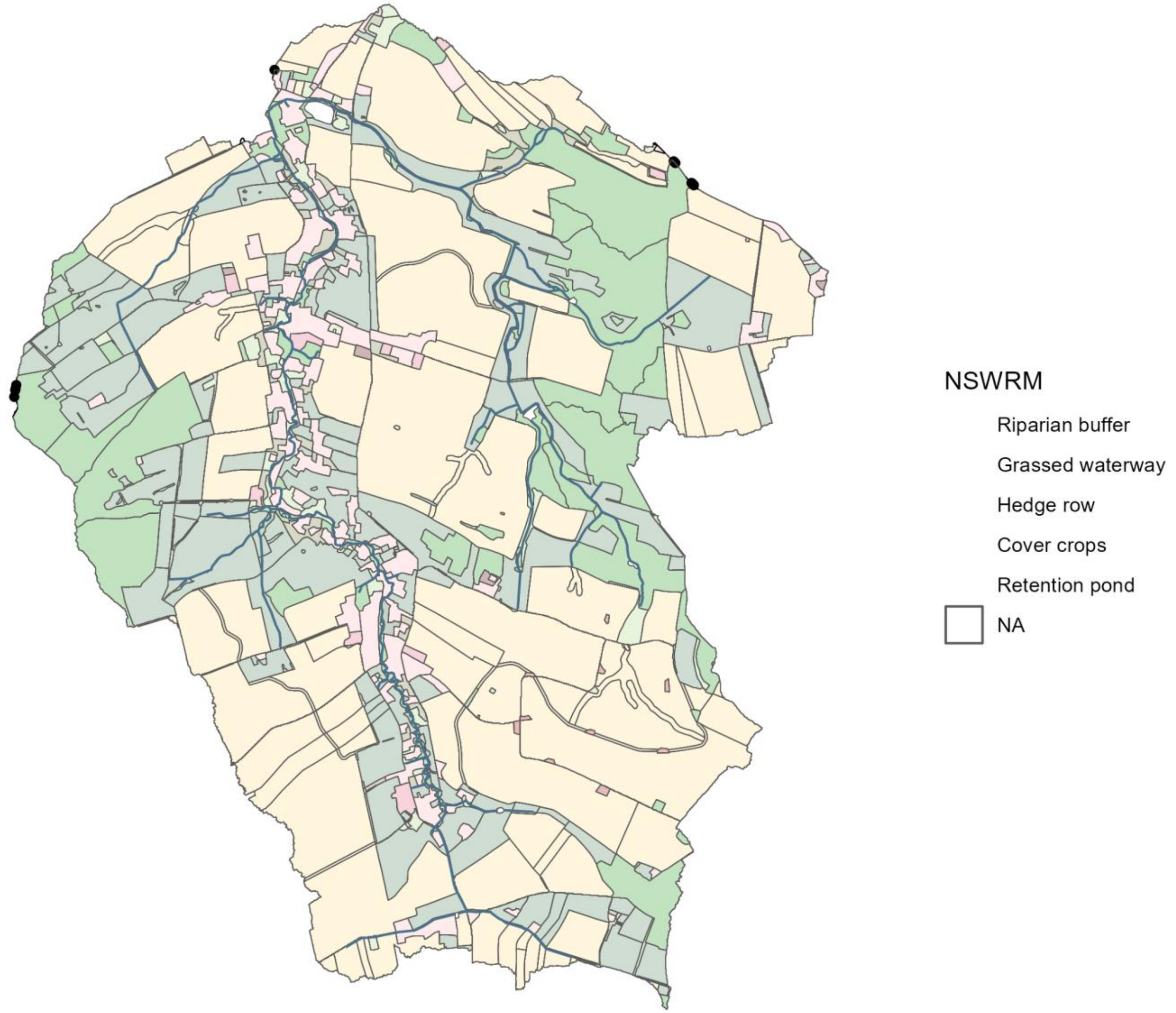
Be sufficiently spatial with your retention measures!

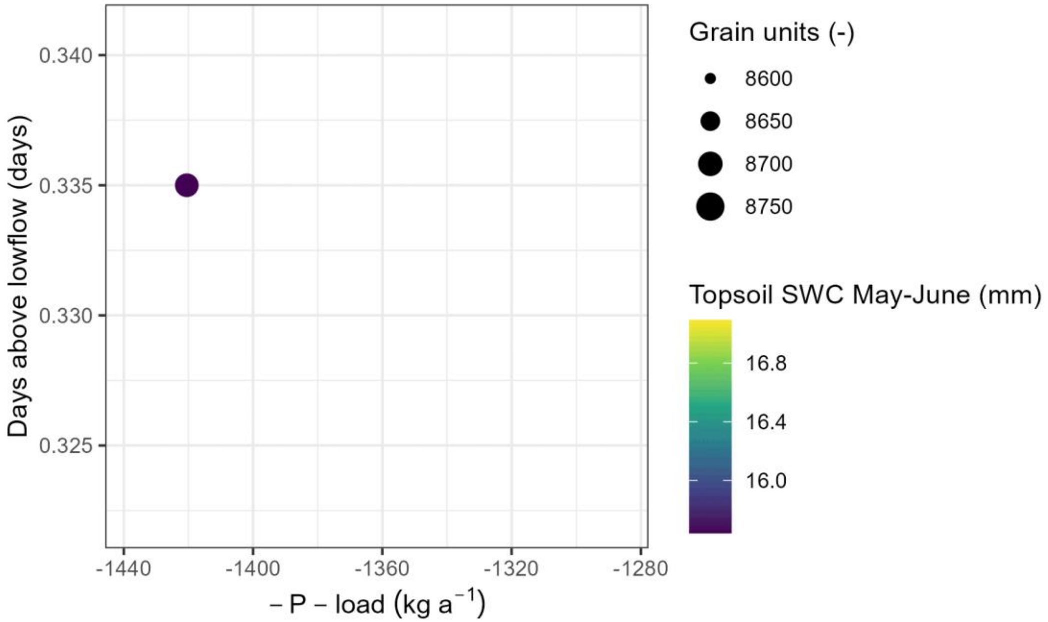
What is needed…





What is needed…





The solution…

SWATmeasR

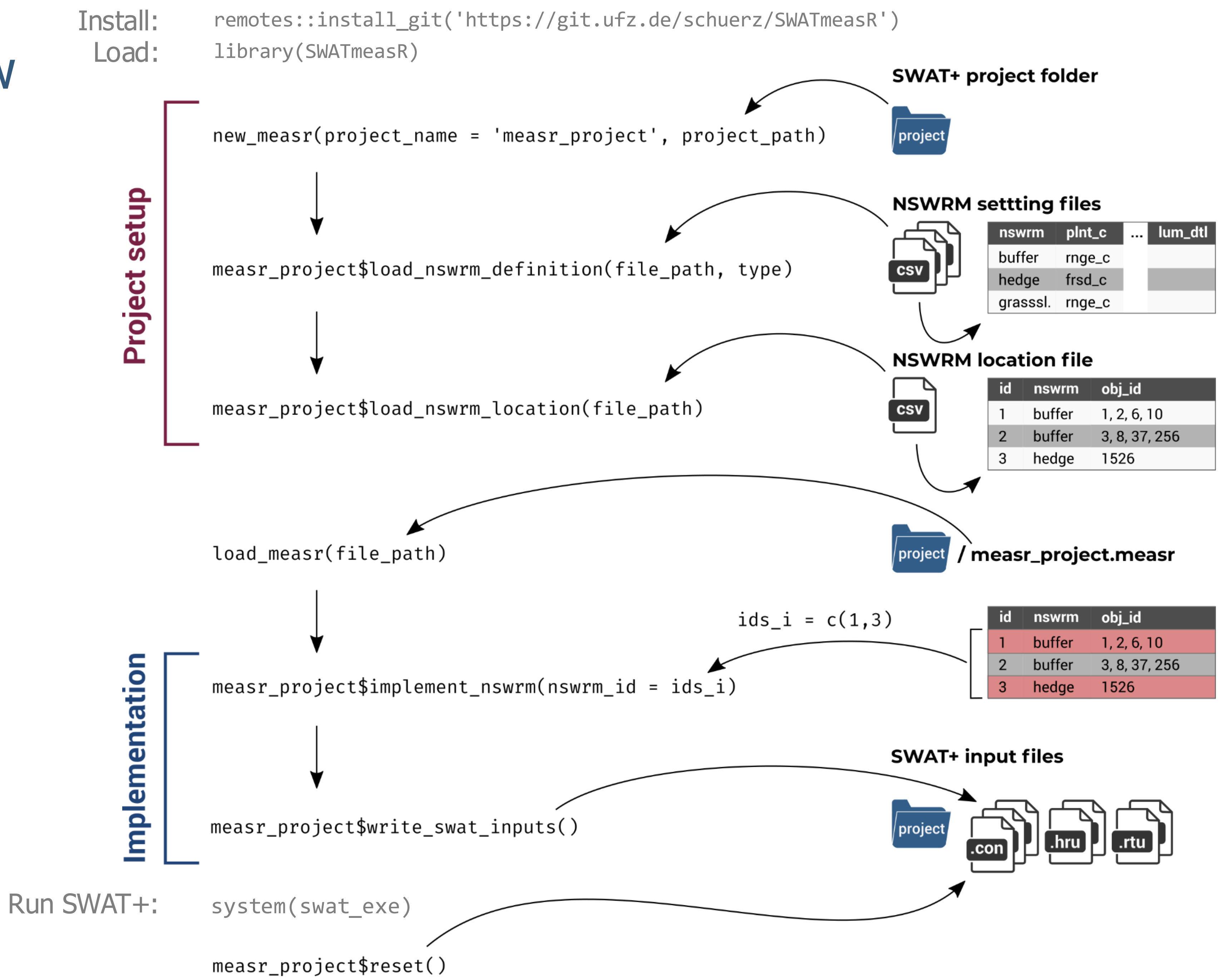
A flexible tool to implement BMPs in SWAT+ model setups

See also Christoph’s presentation at the SWAT conference:
https://swat.tamu.edu/conferences/2024-france/agenda/#gallery-19

SWATmeasR workflow

(1) Setup a .measr project

(2) Then use it to implement NSWRMs

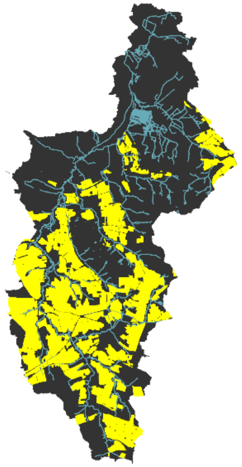


Project setup: `measr_project$load_nswrm_definition(file_path, type)`

- Each measure which could potentially be implemented must be defined
- The following measure types can be defined:
  - `'management'` : All management scheduling related measures, e.g. conservation farming, cover crops...
  - `'land_use'` : All land use change type measures, e.g. buffer strips, grassed waterways, hedge rows...
  - `'wetland'` : Wetland water storage is added to a land object
  - `'pond'`/`'constr_wetland'` : Retention and detention ponds. Land object is replaced by a reservoir.

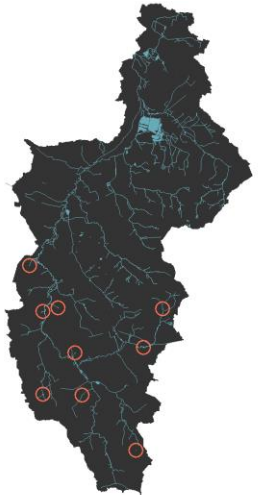
NSWRM allocation in CS1

Low tillage combined with cover crops



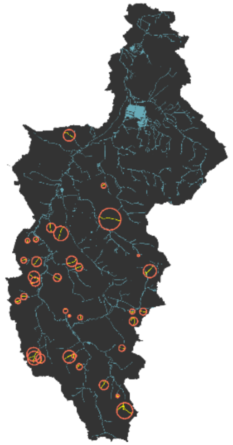
What: Lower tillage depth (max. 12 cm), no autumn furrow, instead winter cover crop before corn, sugar beet or spring barley.
Where: Fields with high potential erosion risk (on average > 15 t/ha,a) according to LfULG risk map.

Detention ponds



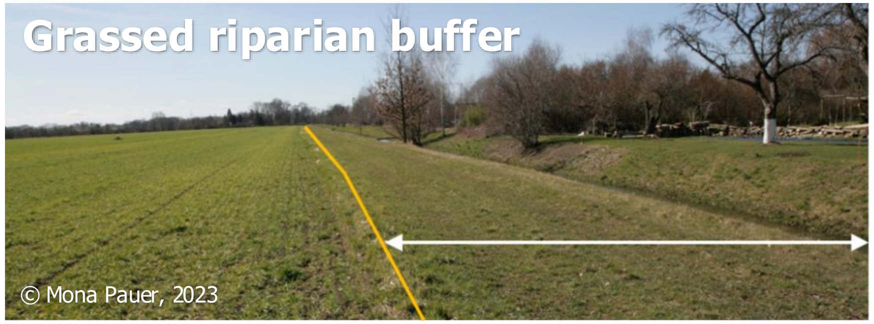
What: Depending on site, principal volume of 700-1700 m³ and emergency volume of 1300-3200 m³; drawdown from emergency to principal spillway within 2 days.
Where: At the end of erosive slopes with close connection to streams and a minimum drainage area of 50 ha.

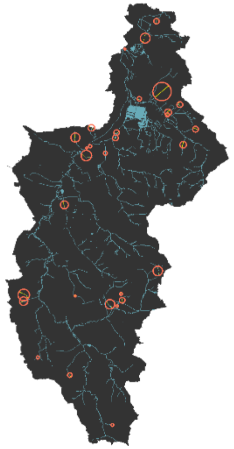
Grassed waterways



What: Permanent grass strip of 30 m width.
Where: Erosive slopes in close distance/connection to river network.

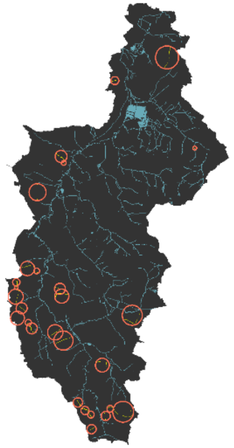
Grassed riparian buffer





What: Permanent grass strip of 12 m width.
Where: Fields without existing 'green' buffer along streams.

Hedgerows



What: Deciduous forest strip of 15 m width; pruning every 5 years.
Where: In fields with exceptionally low density of semi-natural habitats (SNH) in the surrounding area, if possible along contour lines and connecting existing SNH.

Project setup: `measr_project$load_nswrm_definition(file_path, type)`

`type = 'land_use'`

- Definition csv table for land use change type NSWRMs:

nswrm	plnt_com	mgt	cn2	cons_prac	ov_mann	tile	lum_dtl
buffer	rnge_test_com	rnge_mgt	pasth	greening	densegrass	null	
hedge	frsd_com	hedge_mgt	wood_g	greening	forest_light		
grassslope	rnge_test_com	rnge_mgt	pasth	greening	densegrass		
contr_drn						mw24_1000	control_drainage

blank lines preserve the value of the changed land use

'null' will add null in the landuse.lum e.g. remove tile if tile was in place

Project setup: `measr_project$load_nswrm_definition(file_path, type)`

`type = 'pond' / 'constr_wetland'`

- Definition csv table for ponds and constructed wetlands:

hru_id	cha_to_id	cha_from_id	area_ps	vol_ps	area_es	vol_es	k	evap_co	shp_co1	shp_co2	rel	sed	nut
153	475		0.035	0.071	0.044	0.132					drawdown_days2	sedres1	nutres1
201	565		0.059	0.119	0.0742	0.223					drawdown_days2	sedres1	nutres1
234	846		0.071	0.141	0.088	0.264					drawdown_days2	sedres1	nutres1
270	941										drawdown_days2	sedres1	nutres1
997	613		0.07	0.145	0.085	0.271					drawdown_days2	sedres1	nutres1
1634	929										drawdown_days2	sedres1	nutres1
2104	395	604, 605									drawdown_days2	sedres1	nutres1
5087	934	931, 932									drawdown_days2	sedres1	nutres1
5096	524										drawdown_days2	sedres1	nutres1

cha_from_id is optional

Empty fields default values will be assigned

```
area_ps  = 0.8 * area_hru
vol_ps   = 2.0 * area_ps
area_es  = 1.0 * area_hru
vol_es   = 3.0 * area_es
k        = 1.0
evap_co  = 0.6
shp_co1  = 0
shp_co2  = 0
rel      = 'drawdown_days'
sed      = 'sedres1'
nut      = 'nutres1'
```

Project setup: `measr_project$load_nswrm_definition(file_path, type)`

id	name	nswrm	obj_id
1	buffer_1	buffer	479
2	buffer_2	buffer	710
3	buffer_3	buffer	468
4	buffer_4	buffer	337
5	buffer_5	buffer	206, 287, 856
6	buffer_6	buffer	140, 142
7	grassslope_1	grassslope	201, 203
8	grassslope_2	grassslope	209
9	grassslope_3	grassslope	607, 608, 610, 613
10	hedge_1	hedge	5278, 5284
11	hedge_2	hedge	126
12	hedge_3	hedge	122, 123, 124
13	hedge_4	hedge	74, 75, 76, 83
14	lowtillcc_1	lowtillcc	1, 2, 5, 9, 10, 11
15	lowtillcc_2	lowtillcc	17, 18
16	lowtillcc_3	lowtillcc	19, 20, 21, 22, 23
17	lowtillcc_4	lowtillcc	25
18	lowtillcc_5	lowtillcc	49, 50, 51, 52
19	pond_1	pond	997
20	pond_2	pond	5087

- **id** will be the ID to implement a measure.
- **name** is user definable for each measure location.
- **nswrm** refers to the entries in the definition files for the respective NSWRM types (e.g. buffer, hedge, and grassslope were defined in the land_use definition file).
- **obj_id** are single or multiple values of HRUs which are affected by the implementation of an NSWRM.

Highly recommended reading:

Strauch & Schürz (2024)

https://zenodo.org/records/11473793

Optimal Strategies to Retain Water and Nutrients

D5.1: Common optimisation protocol

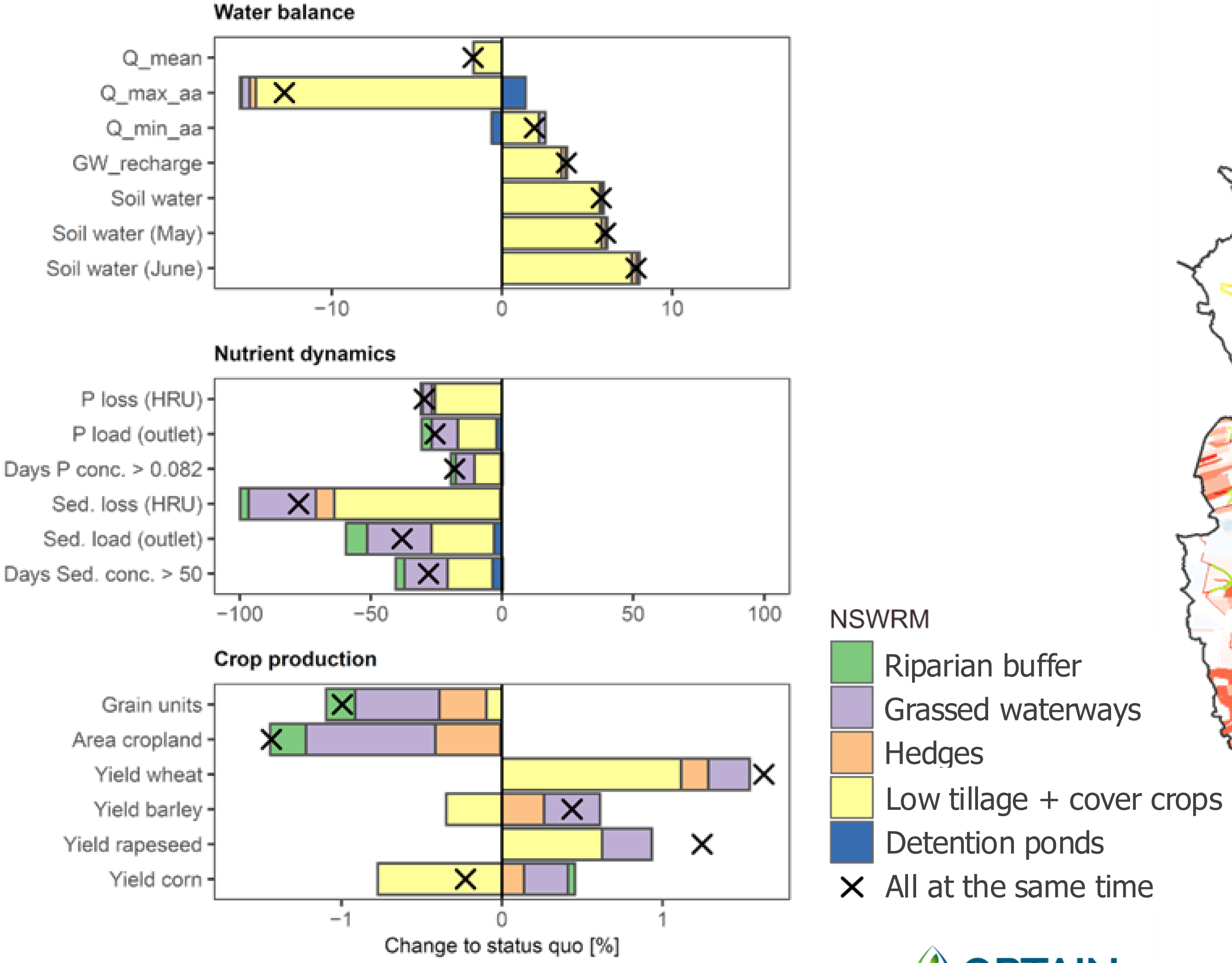
Authors:
Michael Strauch (UFZ), Christoph Schürz (UFZ)

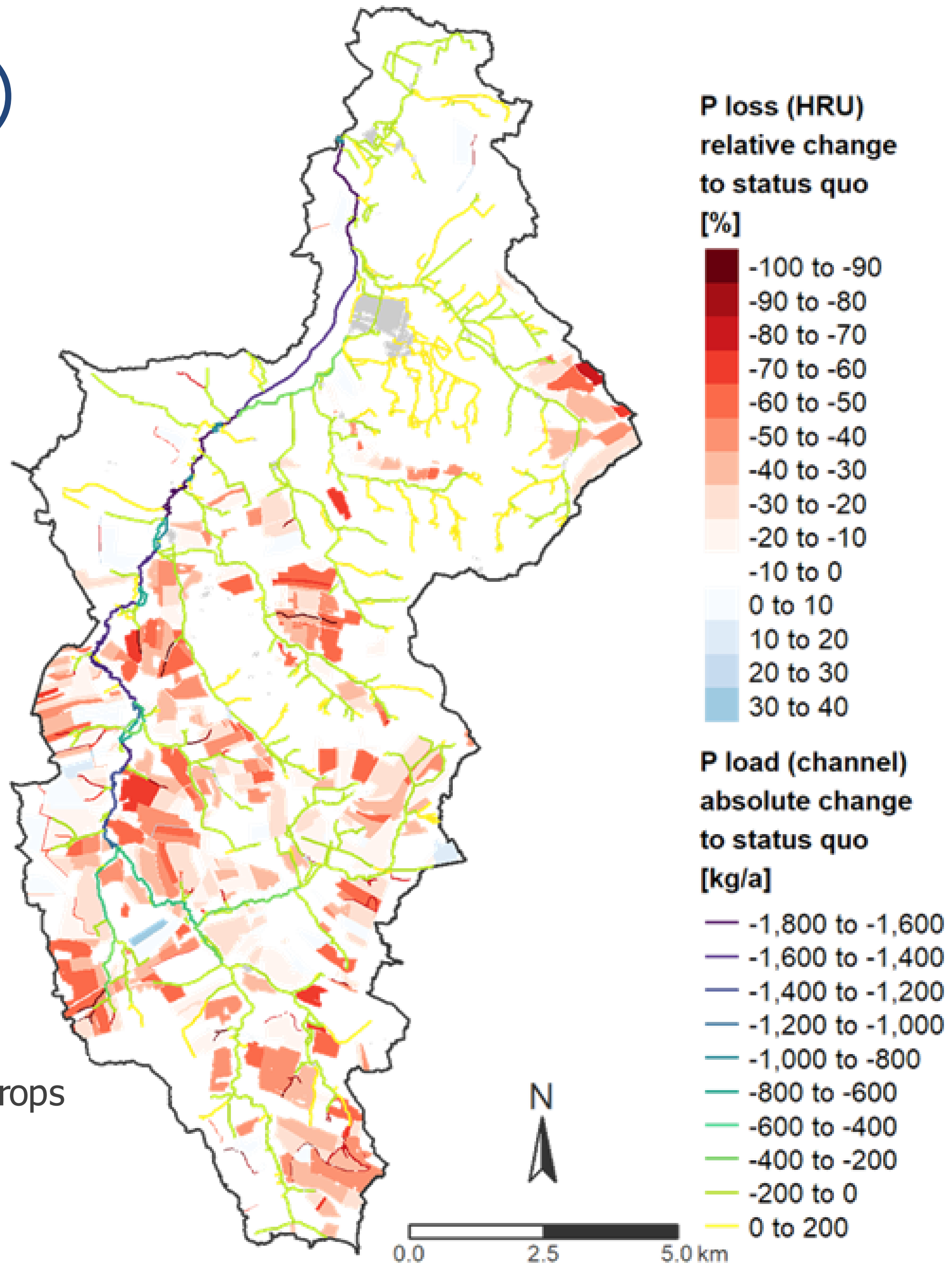
Delivery Date: 31. May 2024

This project has received funding from the European Union's Horizon 2020 research and innovation programme under Grant agreement No. 862756.

Table of Contents

Example results for single scenarios (CS1)



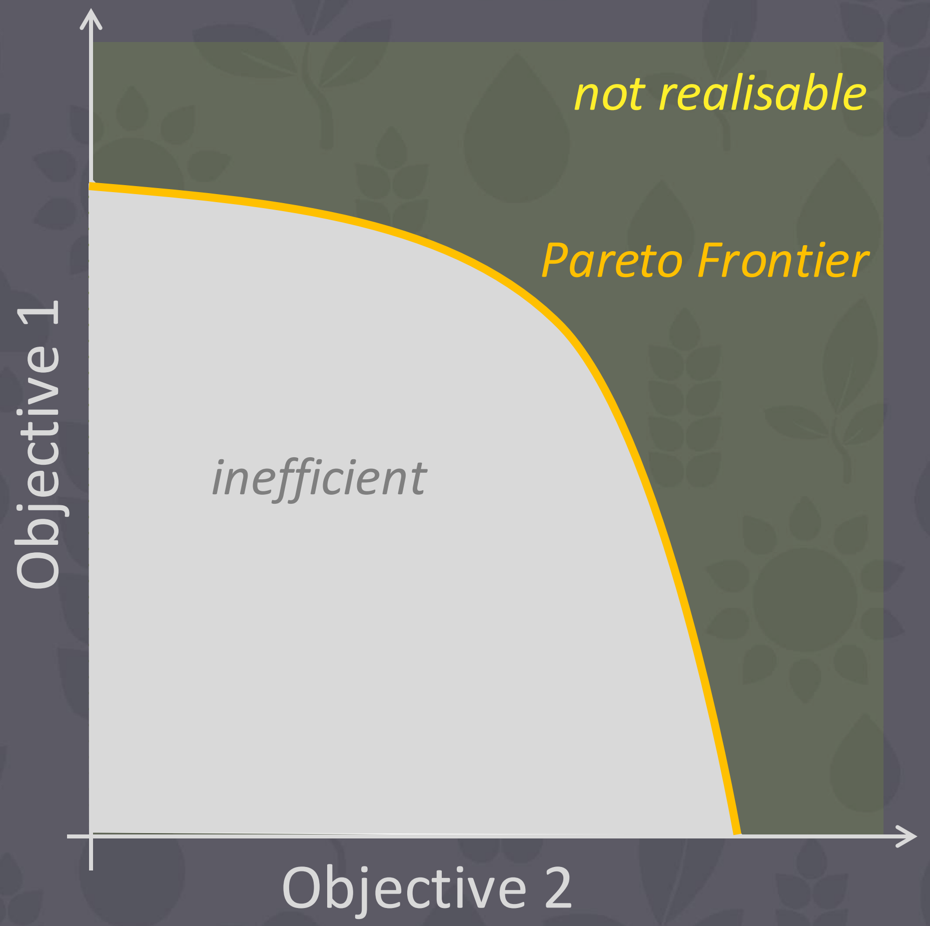


Modeling challenge #6

Optimize the allocation of NSWRMs for multiple objectives!

Pareto optimality

Pareto optimality or Pareto efficiency is a situation where no action or allocation is available that makes one objective better off without making another worse off



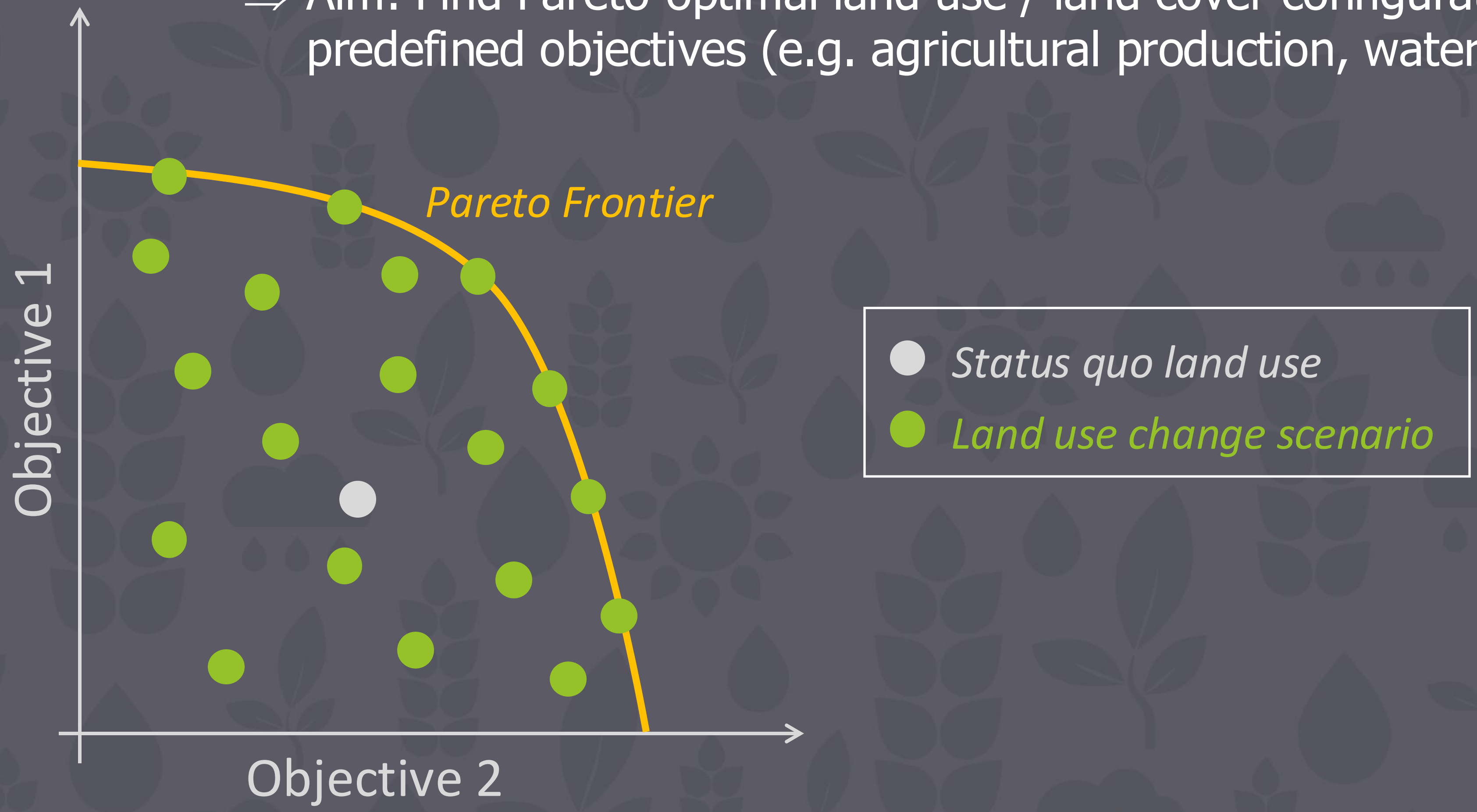
- Concept goes back to Vilfredo Pareto (1848–1923)
- Widely used in economics

⇒ Aim: Find optimal solutions along Pareto Frontier!

Pareto optimality

Concept applied to landscape research…

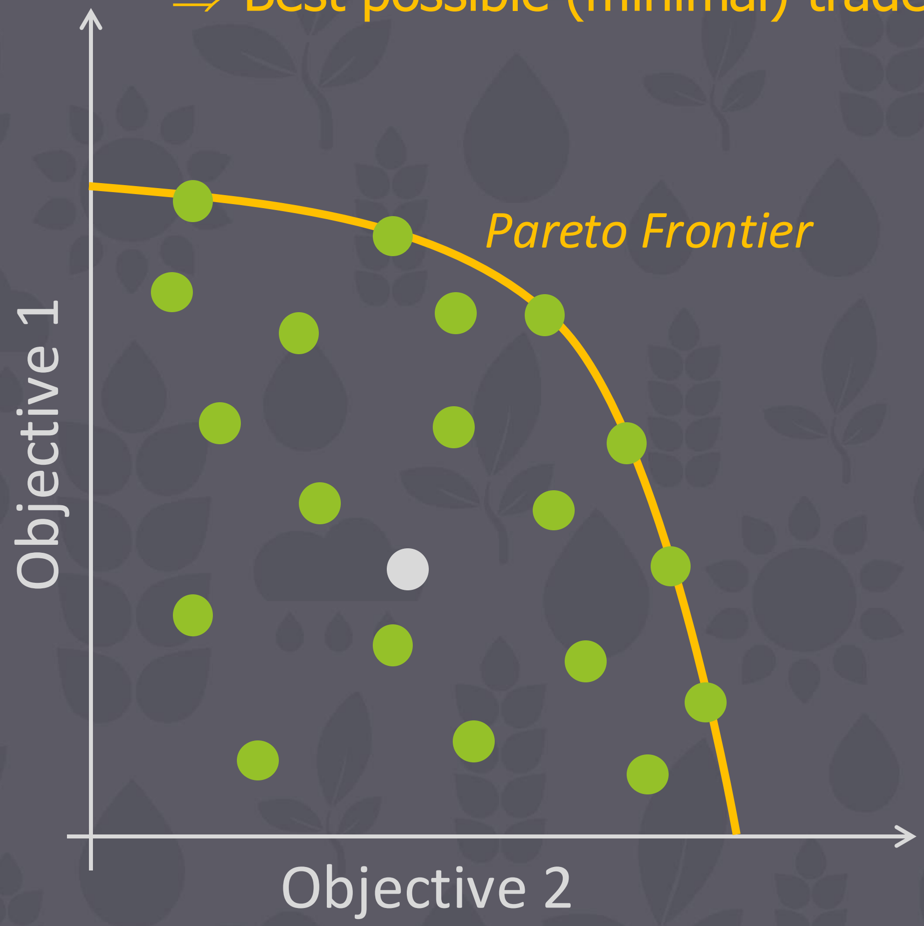
⇒ Aim: Find Pareto-optimal land use / land cover configurations in a landscape for predefined objectives (e.g. agricultural production, water quality, biodiversity, …)



Pareto optimality – Why is it useful for decision making?

❖ Pareto-optimal solutions indicate the potential of a landscape to fulfill different objectives

⇒ Best possible (minimal) trade-offs among conflicting objectives become visible



❖ Stakeholders from different sectors usually have different preferences for different objectives

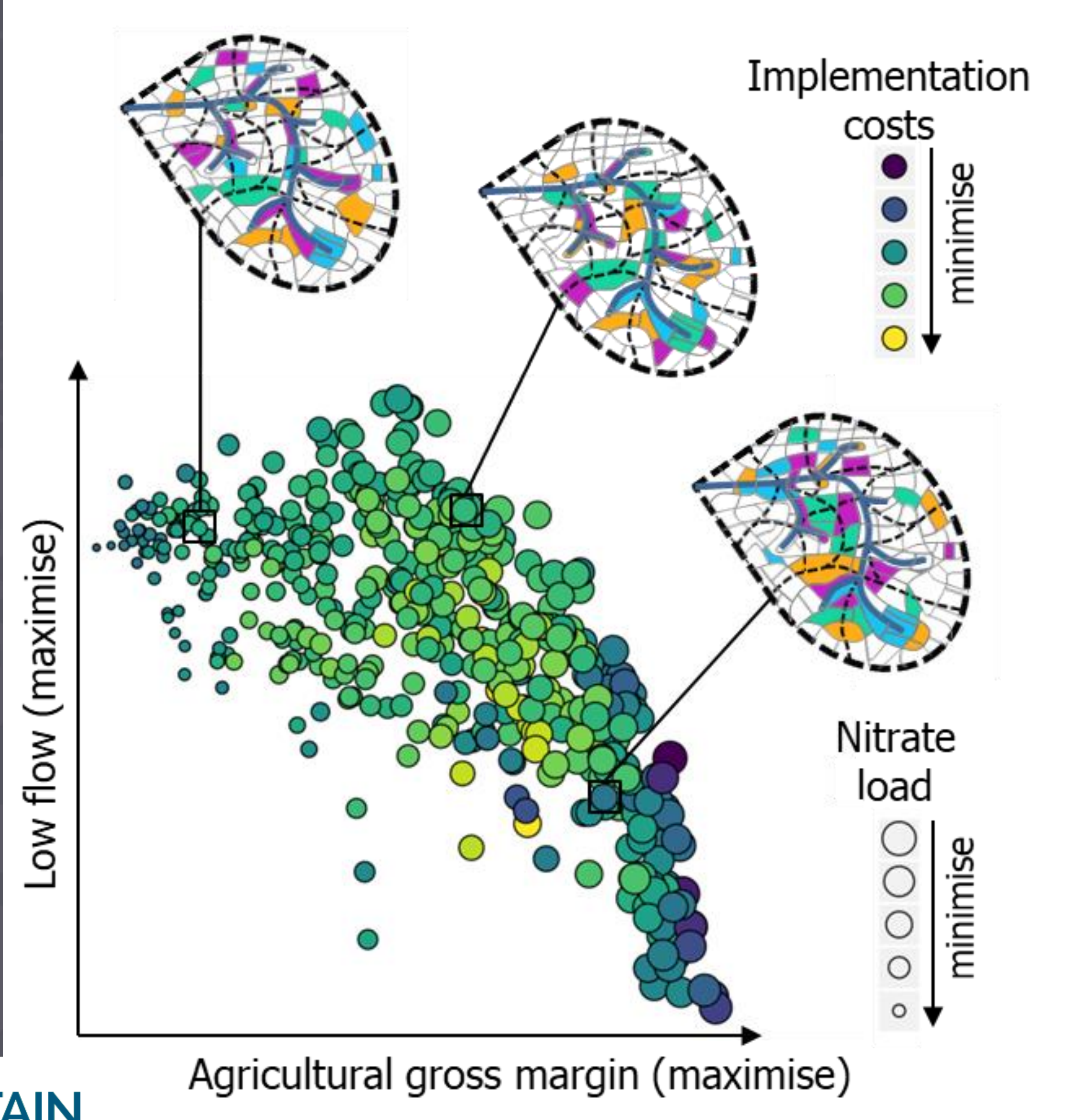
⇒ Stakeholders can discuss and try to find the best possible compromise solution

⇒ Results can support planning processes across sectors

Pareto optimality

The challenge of visualizing multi-dimensionality …

- With >2 objectives the Pareto frontier is not a line anymore but an n-dimensional cloud
- The more dimensions, the more difficult is the search for and visualization of Pareto-optimal solutions

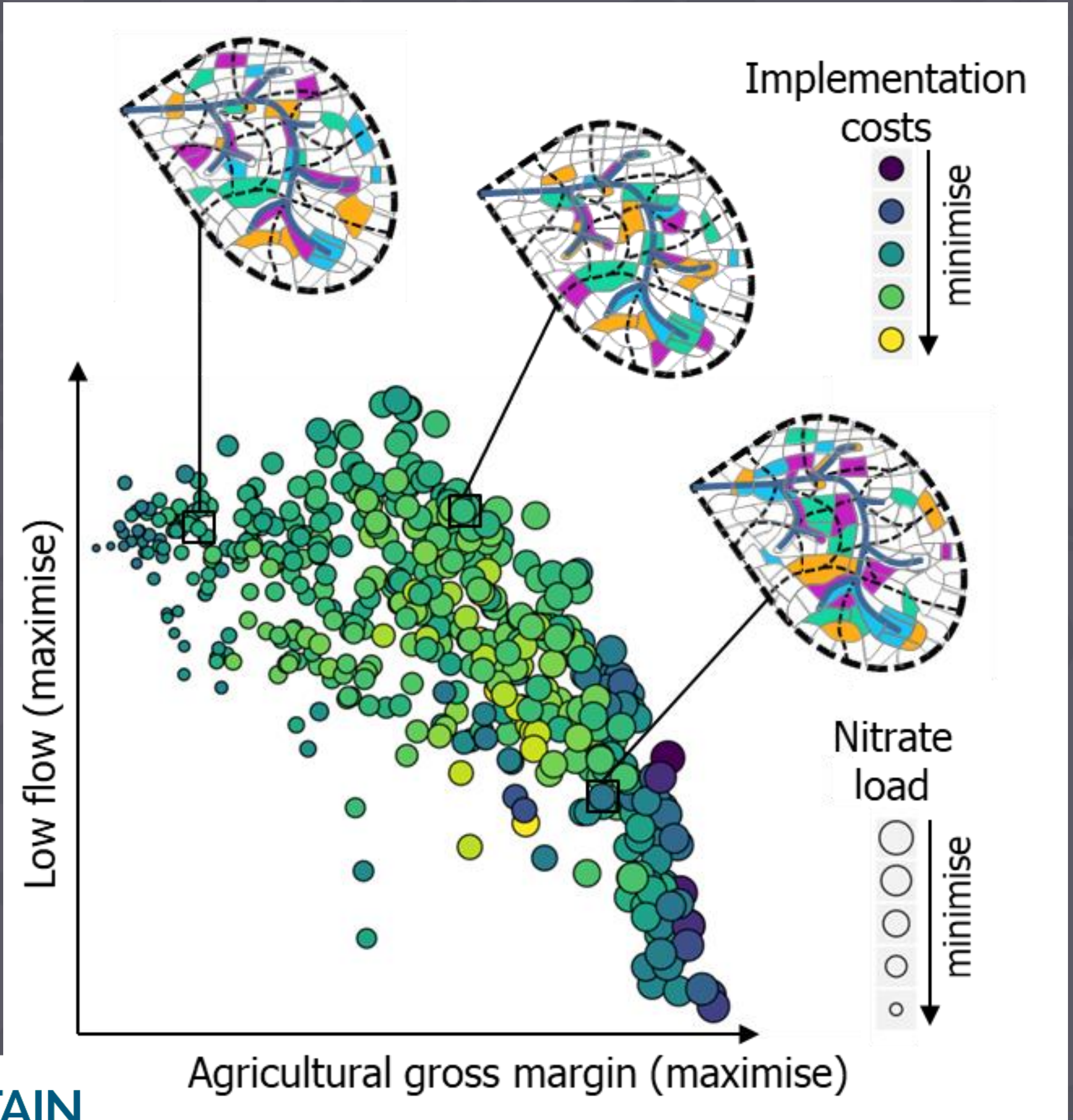


Key research questions in WP5

- Where to implement which measure(s) within a case study to best possible fulfill various environmental and socio-economic objectives?
  => identify Pareto-optimal solutions!

- How to post-process, i.e. analyse, visualise, filter and navigate through the set of Pareto-optimal solutions?

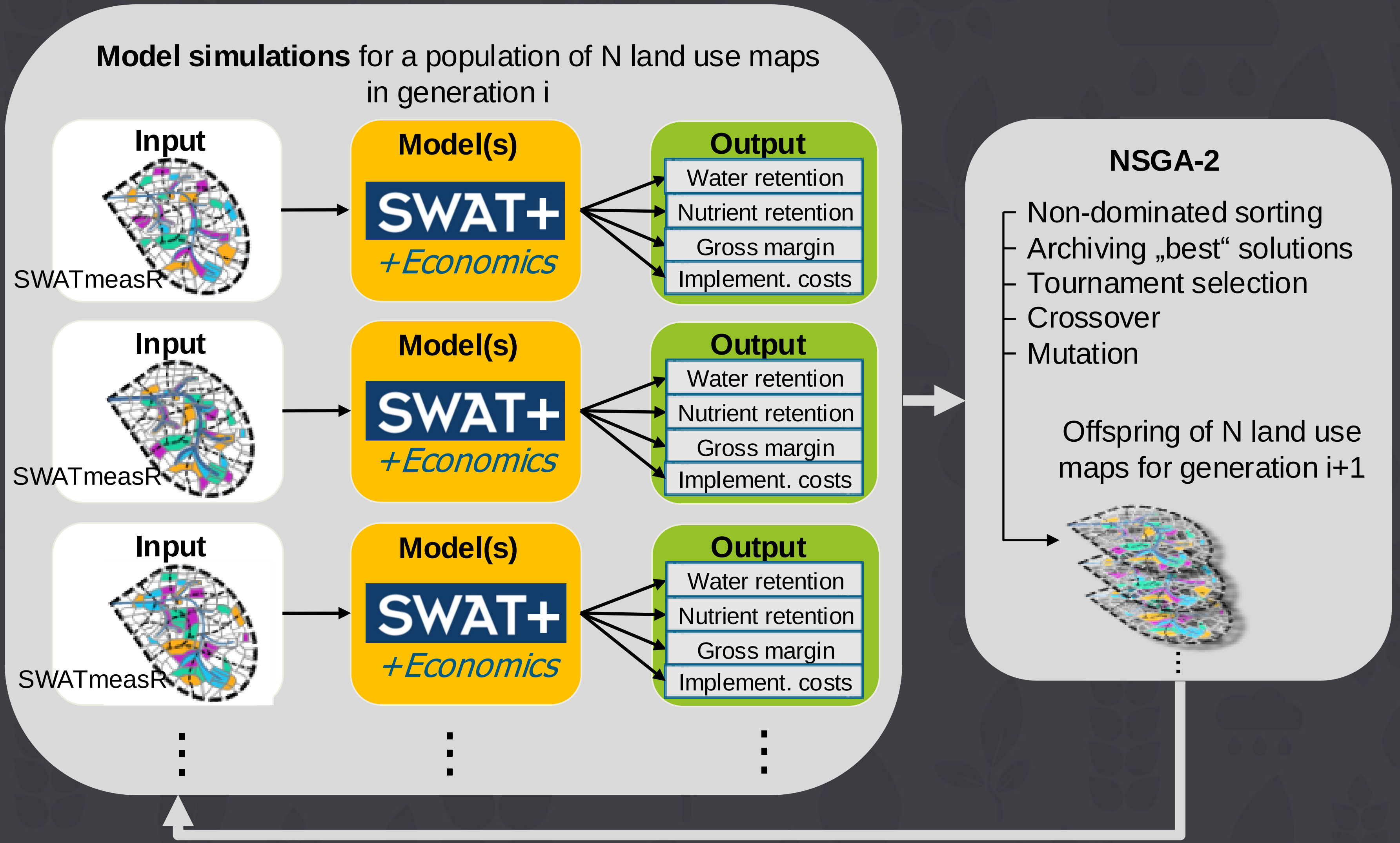
- Which solutions are preferred by stakeholders?



CoMOLA – The tool linking the things to optimize land use allocation

Constrained
Multi-objective
Optimization of
Land use
Allocation

Strauch et al. (2019)
https://doi.org/10.1016/j.envsoft.2019.05.003



OPTAIN

CoMOLA

Encoding a land use map as genome

p = pond
g = grassed waterway
h = hedge
b = riparian buffer
t = minimum tillage

Genome: String of numbers representing a state (1 = scenario off, 2 = scenario on). Each position in the string refers to a specific measure at a certain location (= NSWRM_id for the SWATmeasR)

All measures inactive (= status quo)

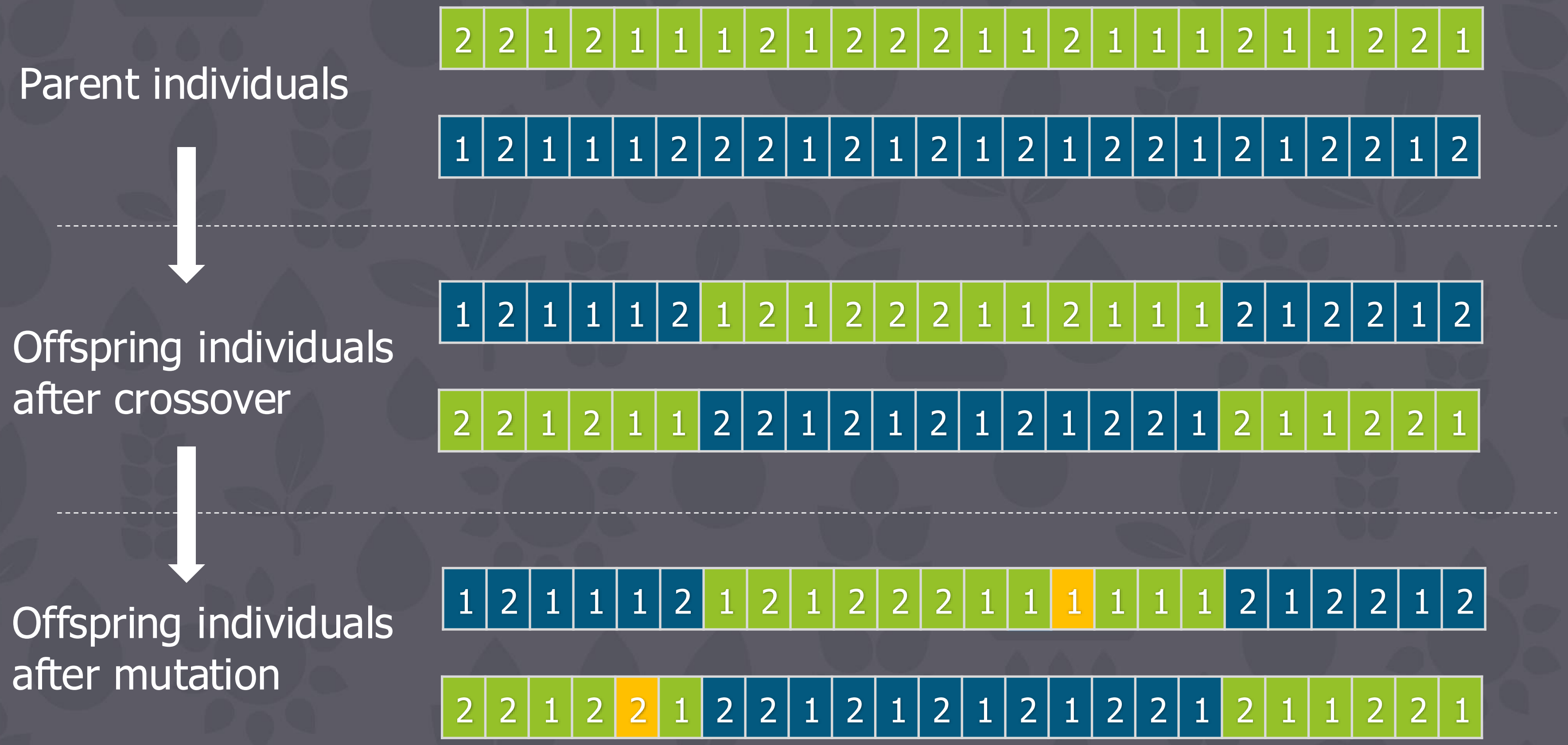
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
p1	p2	p3	g1	g2	g3	h1	h2	h3	b1	b2	b3	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11	t12

All measures active (= maximum implementation)

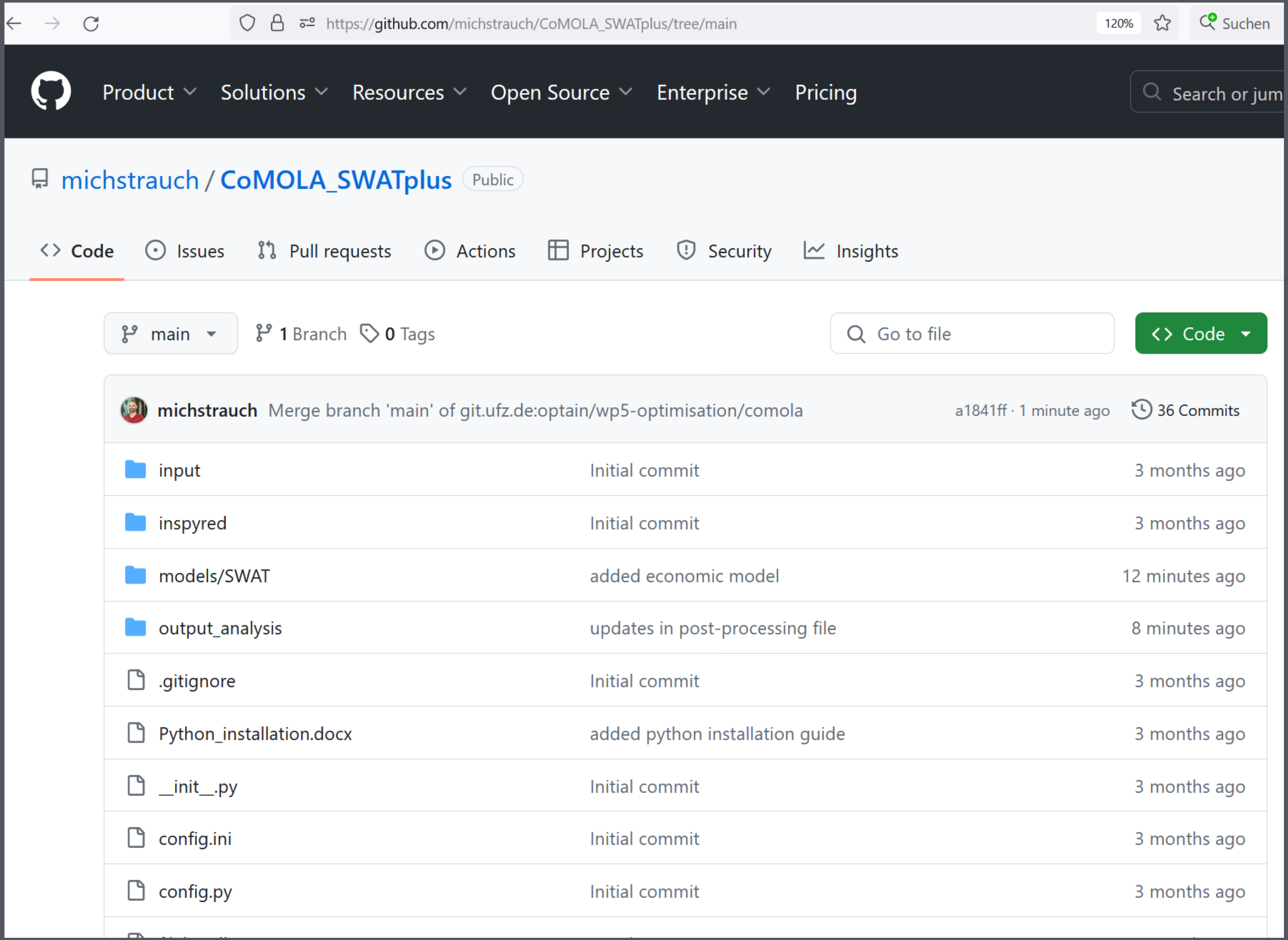
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24
2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
p1	p2	p3	g1	g2	g3	h1	h2	h3	b1	b2	b3	t1	t2	t3	t4	t5	t6	t7	t8	t9	t10	t11	t12

CoMOLA

Crossover and mutation



CoMOLA_SWATplus on GitHub



https://github.com/michstrauch/CoMOLA_SWATplus/comola

CoMOLA file structure for SWAT+ applications

- CoMOLA
  - inspyred (library for optimization algorithm)
  - input (generated automatically)
  - models
    - SWAT
      - txt
        - files to run SWAT+*
          - for ~10 years (e.g. calibration period)
          - fully parameterized!
          - calibrated (calibration.cal)!
          - with proper print settings (only what‘s needed for calculating the indicators)!
        - measr_project.measr*
          - only one project!
          - plausible/tested settings for each NSWRM!
      - economic_model
        - CS_input_data.xlsx*
          - adjust economic parameters
        - CS_input_data.R
      - calc_opt_indis.R (functions for environmental performance indicators)
      - calc_spi_indis.R (functions for economic performance indicators)
      - SWAT.R*
        - adjust for your optimization objectives!
    - …other models (if required)
  - output
  - output_analysis (tools to post-process/visualize optimization results, will be enhanced soon)
  - filehandler.py, maphandler.py, optiAlgorithm.py, config.py, __init__.py (optimization code)
  - config.ini*
    - adjust R.exe path, python.exe path, pop_size (~100), max_generations (100-200)
  - init.R, run_comola.bat
    - start optimization (including initial checks)

Highly recommended reading:

Strauch & Schürz (2024)

https://zenodo.org/records/11473793

OPTAIN

Optimal Strategies to Retain Water and Nutrients

D5.1: Common optimisation protocol

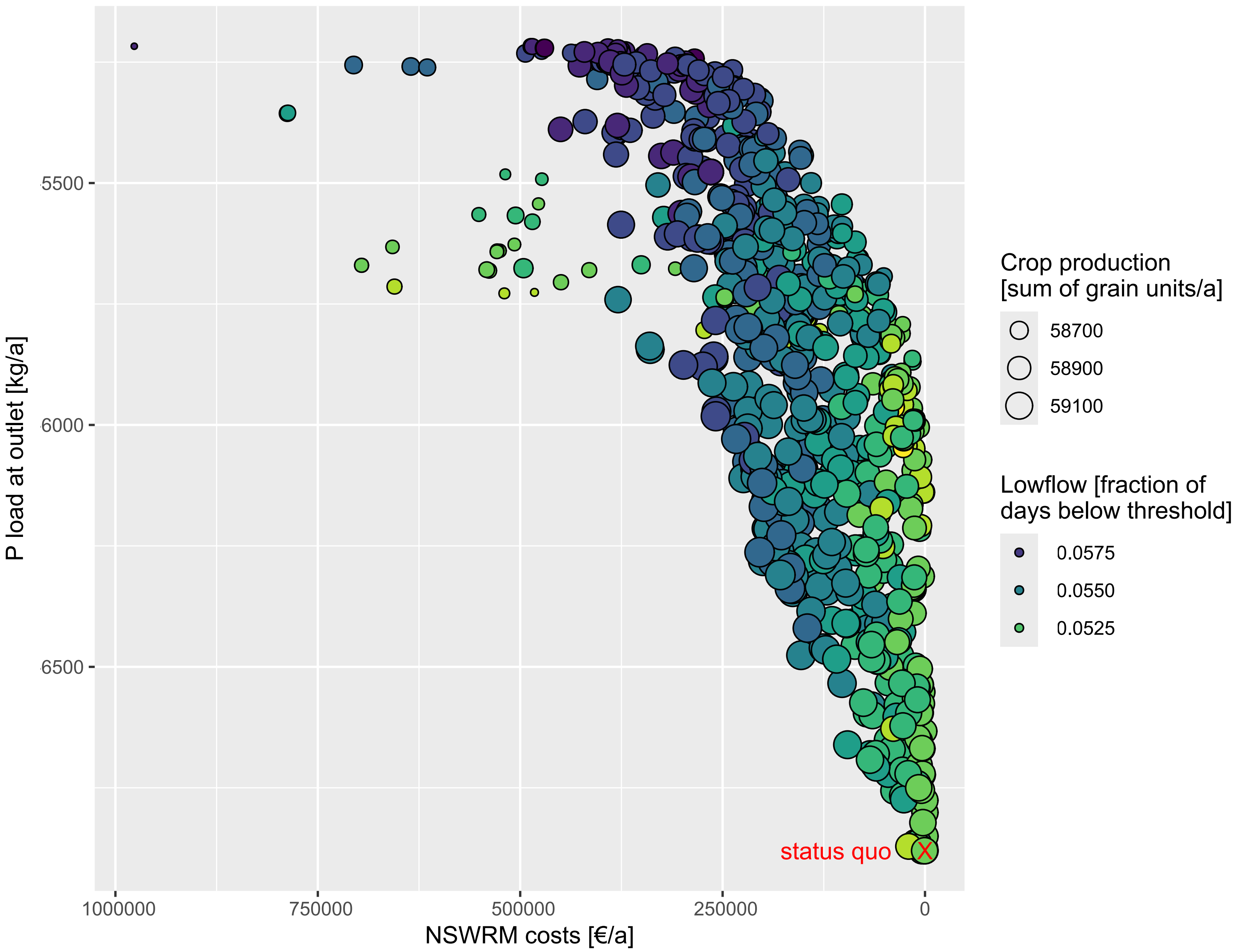
Authors:
Michael Strauch (UFZ), Christoph Schürz (UFZ)

Delivery Date: 31. May 2024

This project has received funding from the European Union's Horizon 2020 research and innovation programme under Grant agreement No. 862756.

Table of Contents

Very first test runs (CS1)



OPTAIN