GeoPython 2018
May 07-09

QGIS Processing Framework: Automating Tasks with Python

Prof. Stefan Keller
Kang Zi Jing

Geometa Lab
HSR Hochschule für Technik
Rapperswil

HSR
HOCHSCHULE FÜR TECHNIK
RAPPERSWIL

FHO Fachhochschule Ostschweiz

Processing Framework: Automating Tasks with Python

Processing Framework - What is it?

- When you Google Processing...
  - Programming language
  - Programming software (IDE)
  - Processing.js (JS port)
  - Big data processing frameworks
  - Software Architecture Framework
- So which one are we actually talking about?
  - Processing Framework - a plugin for QGIS

What is QGIS?

“free and open-source cross-platform desktop geographic information system (GIS) application that supports viewing, editing, and analysis of geospatial data” -Wikipedia

- FOSS4G (Free and Open Source Software For Geospatial)
- GNU GPL (General Public License)
- A software that can display, compose, edit, export and analyze geospatial data and maps
- Supports raster and vector layers
- Supports multiple data sources:
  - File formats like GeoPackage, ESRI Shapefiles, DXF
  - Web Services like WMS and WFS
  - Databases like PostGIS, etc

QGIS and Processing Timeline

2002	2004	2007	2012
Conception of QGIS	**Launch of SEXTANTE**	**QGIS 0.9.0**	**Processing Plugin**
Development and written with Qt toolkit and C++, by Gary Sherman	Processing started as SEXTANTE, a library of geospatial analysis algorithms	QGIS 0.9.0 introduced the Python API to allow developers to add new functionalities using Python	SEXTANTE became a QGIS core plugin and renamed to Processing

Early QGIS pre-Processing

- Lacked comprehensive framework for spatial analysis
- Geographic Resources Analysis Support System (GRASS GIS) plugin had many redundancies and was cumbersome and errorsome
- In general, the geoprocessing tools in core QGIS were:
  - Not homogenous and inconsistent
  - No code modularity and reusability
  - Isolation of tools

Processing Framework - Proper Introduction

- Built-in Python plugin
- Connected to QGIS with Python API (PyQt)
- Contains Toolbox binaries for R, Orfeo (Raster), SAGA GIS, GRASS GIS
- “Middleman” between these algorithms and tools and QGIS client, making them easy and convenient to use

Processing Framework - 4 Main Goals

- Efficiency
  - Efficient integration of libraries and binaries to make them easy and convenient to use
- Modularity
  - Promote consistency and reduce duplication of commonly used code blocks
- Flexibility
  - Implemented algorithms can be reused in any other graphical tools included, without additional work or conversion
- Automatic GUI generation
  - Processing helps generate GUI based on algorithm description so developers don’t have to

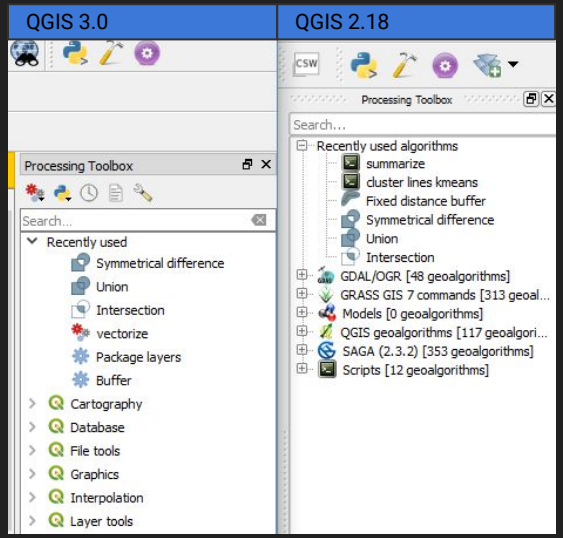
Processing Framework - Architecture

Processing - How an Algorithm Works

1. Plugin loads
2. *ProcessingPlugin* instance initializes *Processing* class
3. *ProcessingConfig* and *ProcessingLog* initialize
4. *AlgorithmProviders* loads
5. Each *AlgorithmProvider* contains list of *GeoAlgorithms*
6. These contain logic and specifications for the algorithms to run (param and output types)
7. Files that contain such specifications and logic of backend binaries can be found in the subfolders of *algs*

Processing Plugin Features

- Recently revamped!
- Toolbox (main element of GUI)
  - Execute a single algorithm once or a batch process of it
- Graphic Modeler
  - GUI where several algorithms can be combined into a workflow
  - Creates essentially a single process with several subprocesses
- History Manager
  - Easily reproduce previously used algorithms and features
- Batch Processing Interface
  - Execute batch processes
  - Automate execution of single algorithm with multiple datasets
- Scripts
  - Write user defined standalone scripts
  - These scripts can be implemented to run via Processing Framework



Advantages of Processing

- Workflow automation (obviously)
- Documentation
- Integration of algorithms
- Can be called from command line and within Pythonic scripts
- Automatic GUI generation

Limitations of Processing

- Inflexible inputs and outputs
  - Originally not possible, but possible as of 2.14
- No room for interactivity
  - Doesn’t accept user inputs
- Reduced performance when conversions of outputs
  - Results parsed as inputs in a chain algorithm still requires conversion

Credits and References

- Processing: A Python Framework for the Seamless Integration of Geoprocessing Tools in QGIS by Anita Graser
  - In-depth development history on Processing Framework
- Processing GitHub repository by Victor Olaya (developer of Processing)
- QGIS 2.18 Documentation
  - Contains a lot of resources and documentations
  - Links to tutorials and textbooks like the PyQGIS Cookbook, QGIS Developers Guide
- QGIS Tutorials by Ujaval Gandhi
  - Helpful step by step tutorials on many aspects of QGIS
- Vast amount of resources, forums and an active and helpful community online
- Special thanks to helpful developers like Anita Graser and other users on GIS Stack Exchange for answering my questions
- And of course, the wonderful people at Geometa Lab, HSR

So… where to now?

Workflow Introduction - Autobahn Construction

- On the GitHub repository is a workflow that we would be attempting to automate today
  - GitHub repository:
- Based on Task 6 of the course *Introduction to GIS and Digital Cartography* by Claas Leiner, University of Kassel, 2010
  - Adapted by Prof. Keller in 2017 for teaching Vector Analysis
  - Translated from German into English, updated for QGIS 3.0, and adapted for this workshop by me
  - Translated workflow in English can be found in the GitHub repository, along with adapted problem sets and tasks for this workshop

Autobahn Construction -
Introduction

- Geospatial analysis
  - On the effects of building a highway on nearby habitat types
- Working with buffers, intersections and other Geoprocessing tools
- Automation of workflow with Processing Framework
  - Graphic Modeler
  - Scripting with Python Console
  - Using Processing algorithms
  - Creating scripts

Workflow Datasets

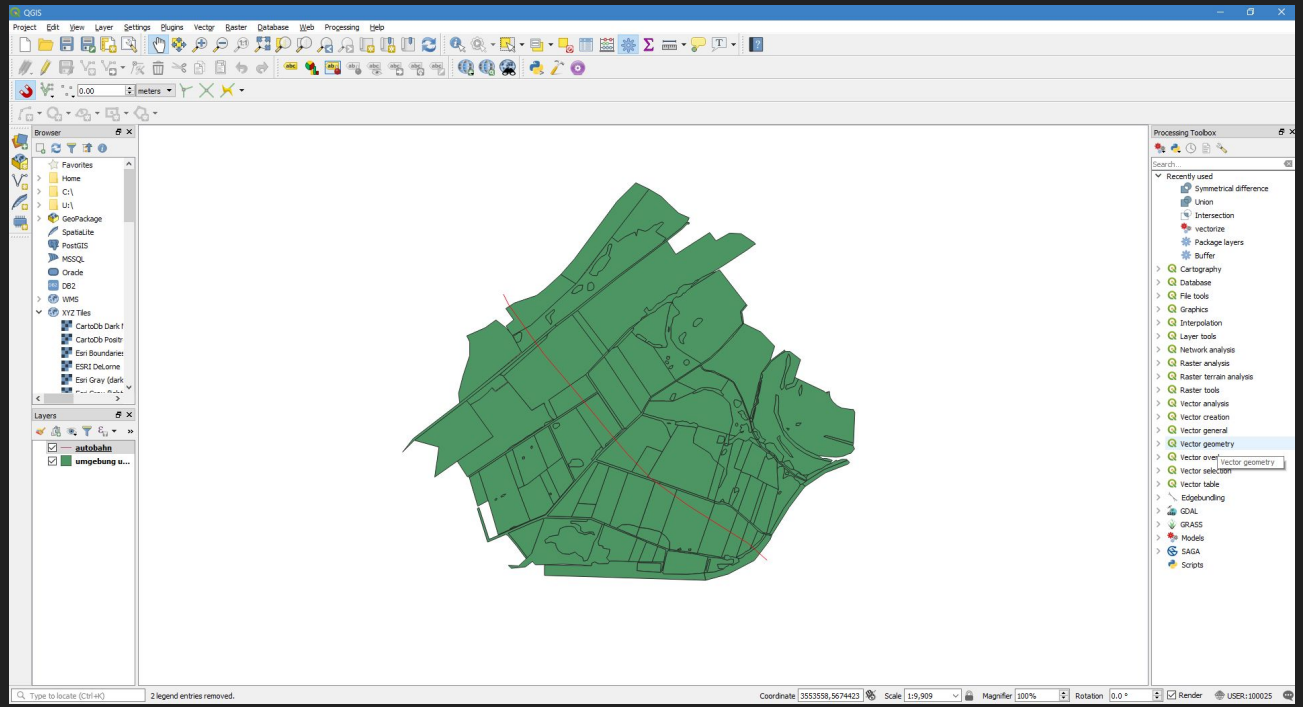
1. Umgebung.gpkg (= environment)
    - Environment polygonal vector layer
    - Attribute table that shows attributes like the type, usage, etc of habitats
2. Autobahn.gpkg (= expressway)
    - Line vector layer that shows the proposed area of autobahn to be constructed
    - Does not include the physical space the actual autobahn will take

Workflow Steps

1. Load the GeoPackage layers as vector layers on to QGIS
2. Represent the physical space of the proposed Autobahn with diameter 20m
3. The impact that the autobahn causes is bigger than its physical space. Represent these impact areas, of 100m and 300m
4. Unify the 3 impact areas to create one aggregated impact area with 3 different impact zones
5. Show the area in the environment that is actually in the impact zone
6. Show the impacted habitats that are actually endangered or protected by laws
7. Style the results and make it readable and easier to analyze

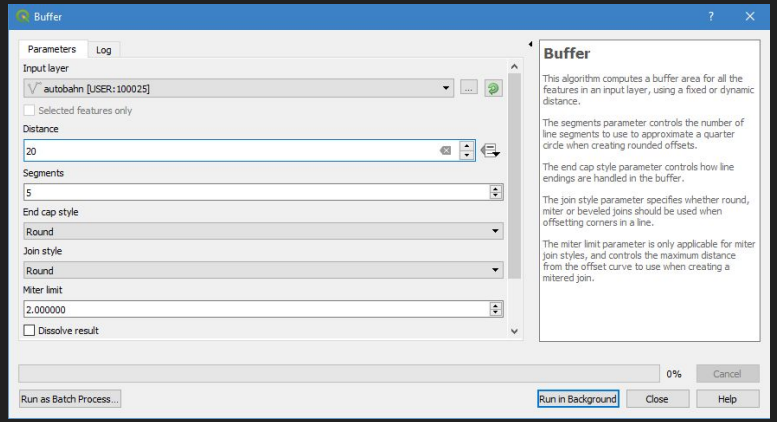
Step 1: Loading the Vector Layers

Load autobahn.gpkg and umgebung.gpkg on to QGIS



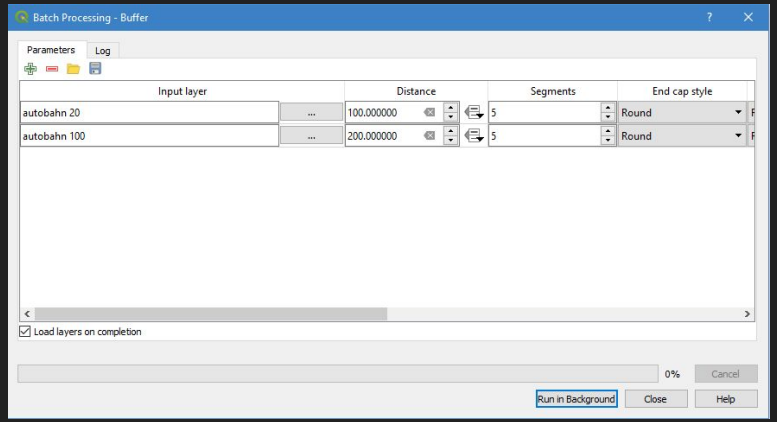
Step 2: Representing the Autobahn

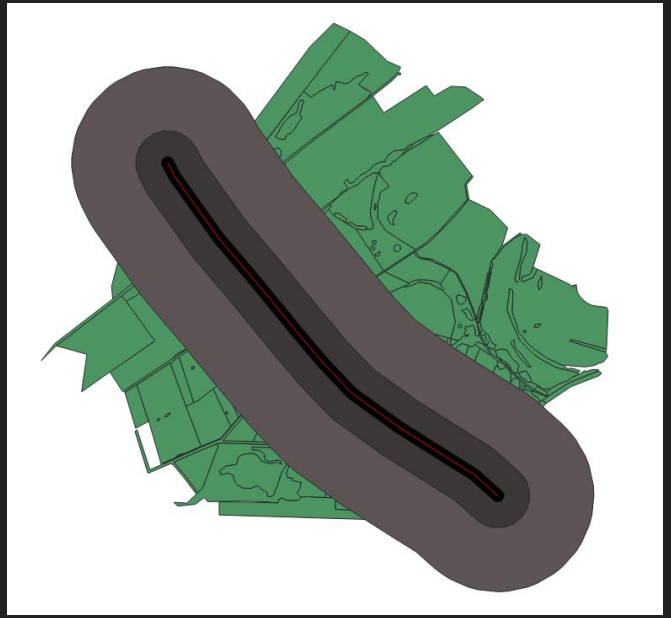
Create a buffer of 20m on the *autobahn* vector layer



Step 3: Representing the Impact Areas

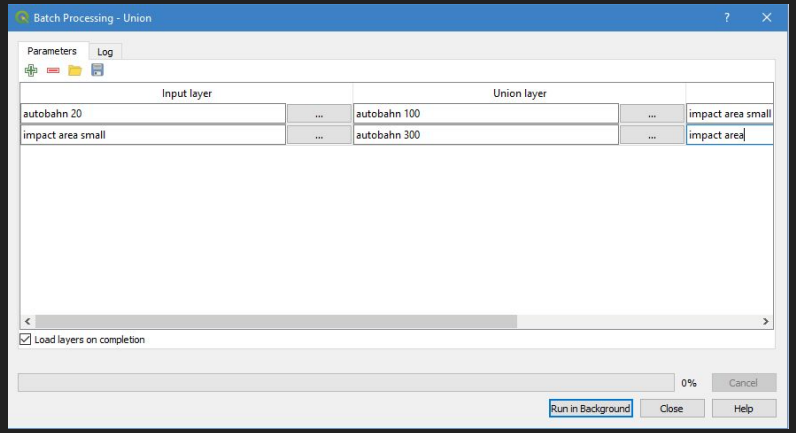
Create 2 more buffers from previous buffers for 100m and 300m

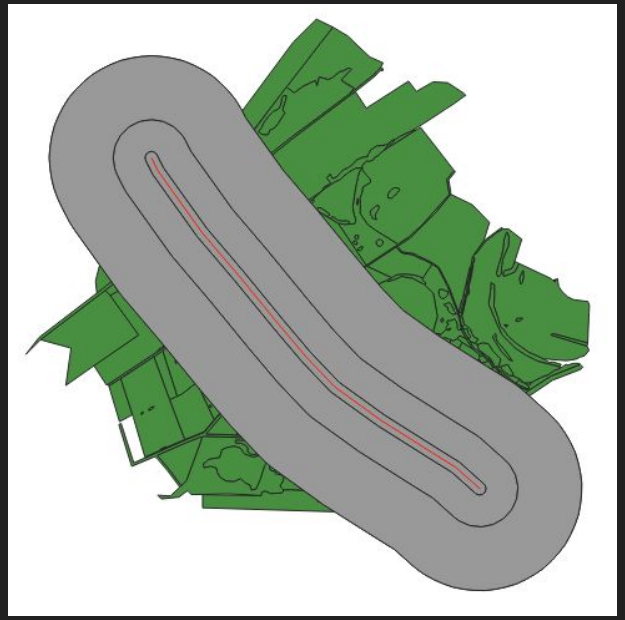




Step 4: Aggregating the Impact Areas

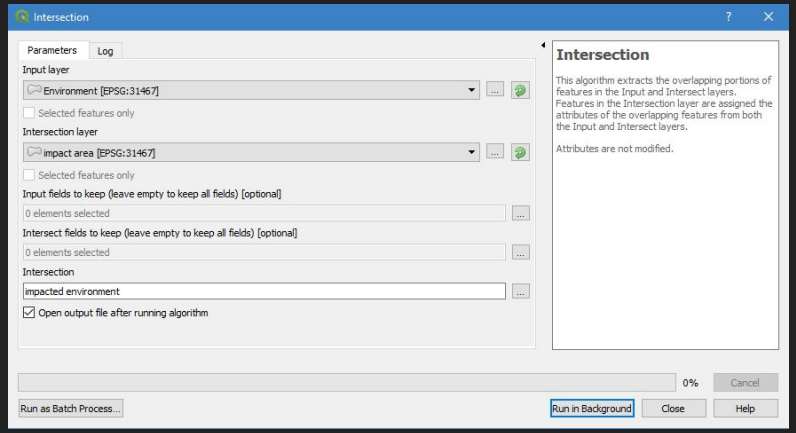
Perform a Union on the 3 buffers to create an overall Impact Area

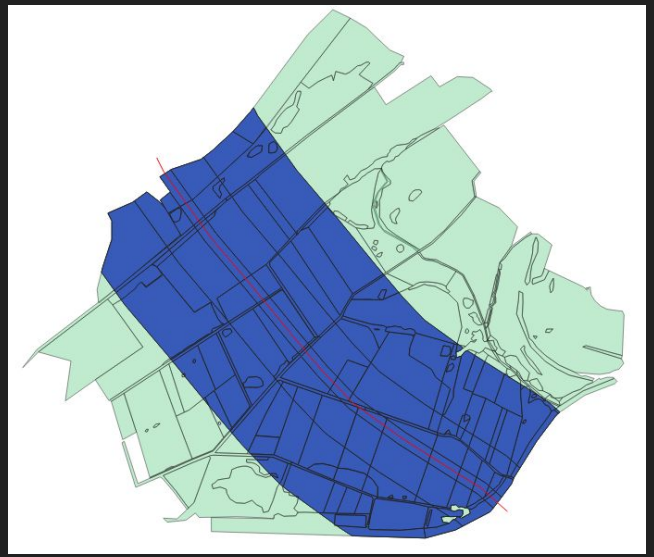




Step 5: Show the Impacted Habitats

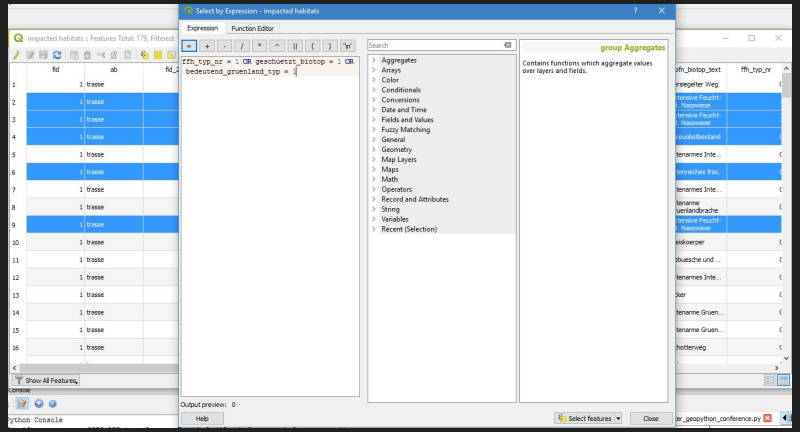
Perform a Intersection on the Impact Area and Environment





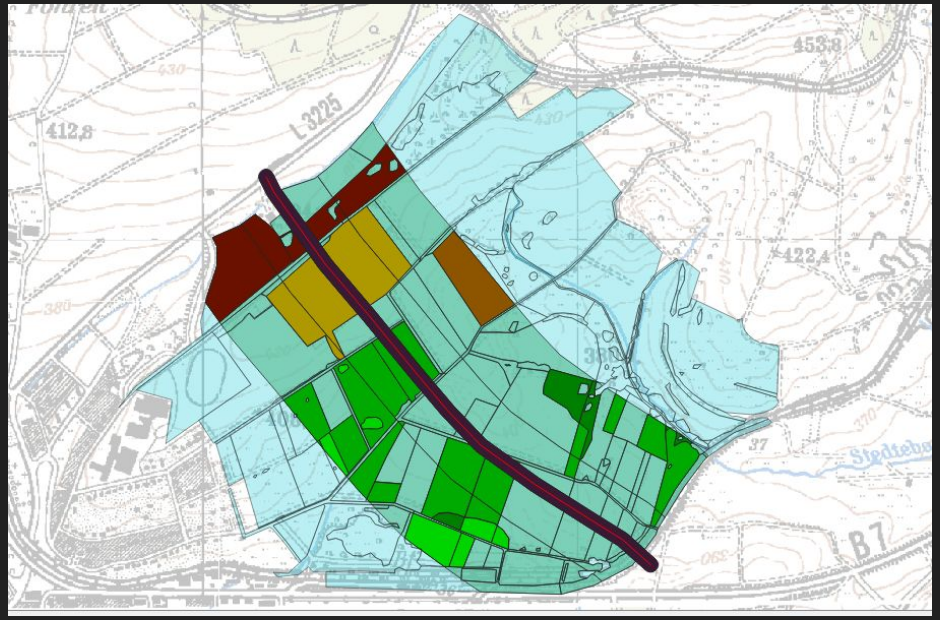
Step 6: Show the Protected Habitats

Query the features on the intersected habitats



Step 7: Style the Results

Using the available styling functions, make the end results more readable and easier to understand

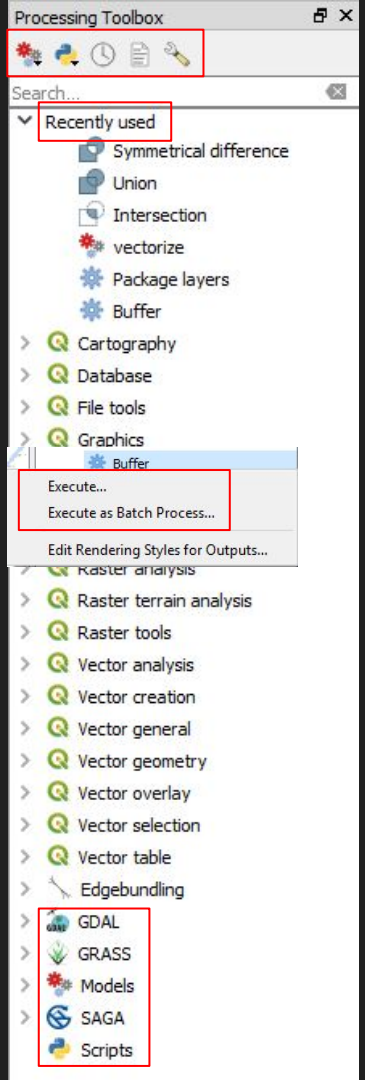


Processing Features

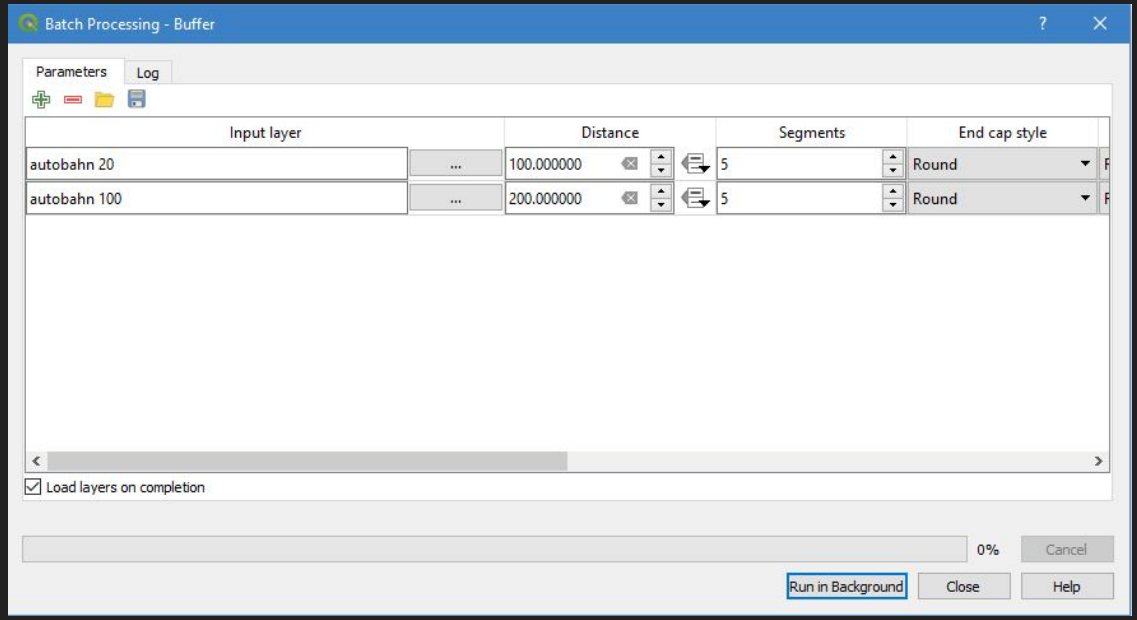
- Toolbox
- Batch Processing Interface
- Graphic Modeler
- History
- Script

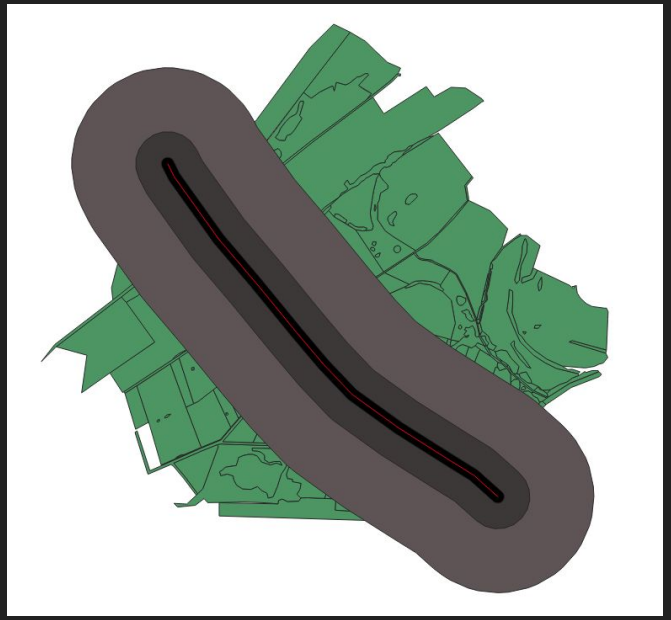
Processing Feature - Toolbox

- Main element of the Processing GUI
- Lists all available algorithms grouped by their providers
- Access point to run these algorithms, be it single or batch processes
- Menu buttons at the top for *Graphic Modeler, Scripts, History, Options,* etc

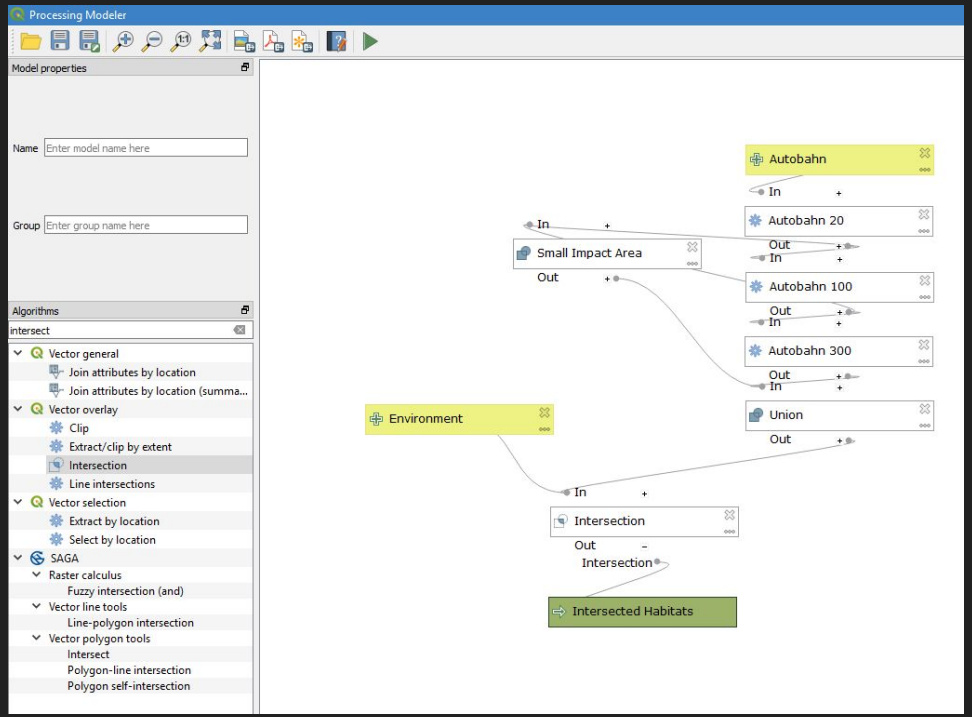


Processing Feature -
Batch Processing Interface

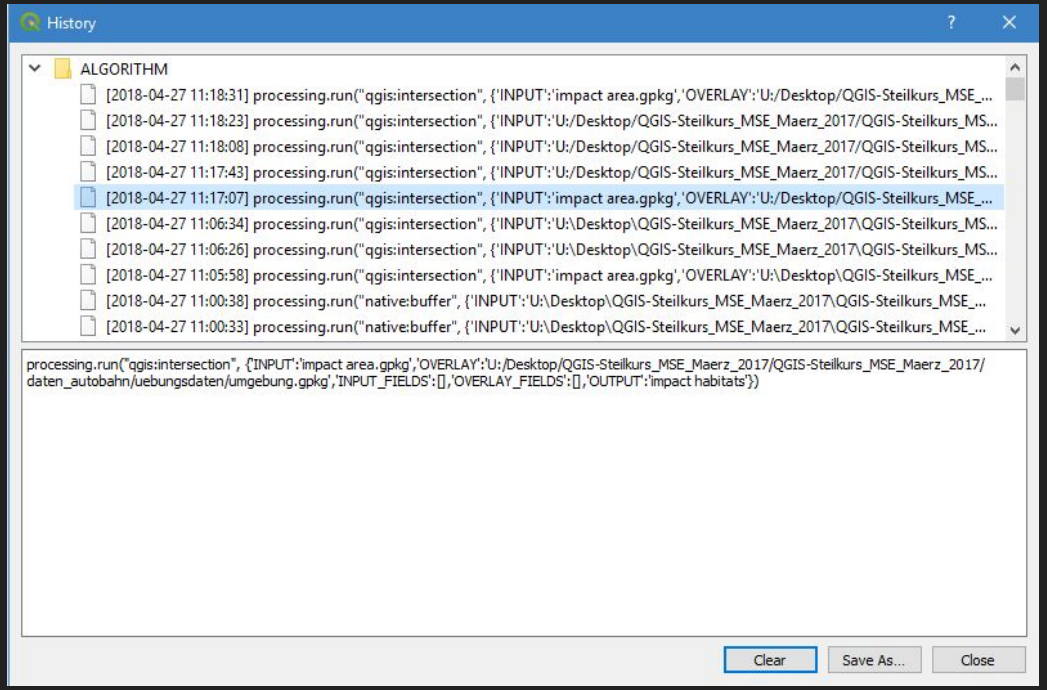




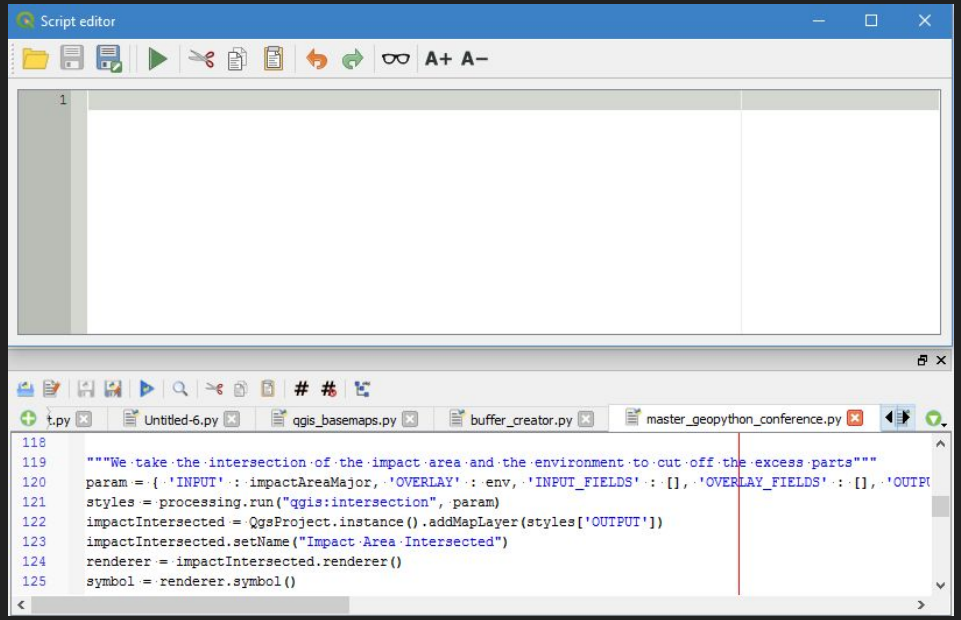
Processing Feature - Graphic Modeler

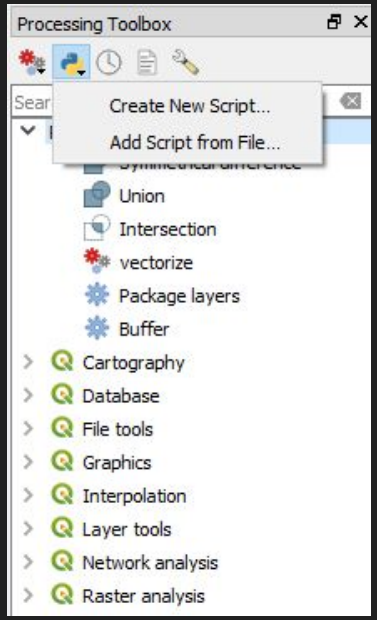


Processing Feature - History Tool



Processing Feature -
Scripting Tool





Processing -
Overview and Conclusion

- Certain geoanalytical tasks involve menial and repetitive steps
- Processing is very nifty as it automates entire processes
  - Example: Your agency wants to build an autobahn whereby the autobahn construction would have the least impact on the environment and protected habitats nearby
  - Solution: Run the script/algorithm through batch process with different inputs and then later access the end results
- We also learnt there are some limitations of Processing:
  - Inflexible parameters/outputs
  - No interactivity
  - Conversion of outputs is expensive

Scripts Demonstration

- Script 1: Script that automates the entire workflow
  - *Master Script Straightforward.py*
- Script 2: Interactive script with information on what is happening
  - *Master Script with Info.py*
- Script 3: Interactive, (almost) flexible parameters, basic error handling
  - *Master Script Model.py*

Scripting in QGIS -
PyQGIS and QGIS 0.9.0

- QGIS 0.9.0 introduced Python to its client
- PyQGIS or Python Console in QGIS client
- Features of PyQGIS:
  - Automatically run Python code when QGIS starts
  - Create custom applications with Python API
  - Run Python code and commands on the Python Console
  - Create and use Python plugins

Hands-on Exercises

Hands-on Exercises - Objectives

- Introduction to using QGIS
  - Loading vector layers
  - Some basic functions and nifty tips and tricks
  - Python Console
- Introduction to Processing in QGIS
  - Processing Toolbox
  - Batch Processing Interface
  - Graphic Modeler
- Introduction to Scripting in Python
  - Basic semantics
  - User inputs
  - Classes

Getting Started -
Prerequisites

- QGIS 3.0
- Connection to workshop GitHub repository (or a cached page)
  - *autobahn.gpkg* and *umgebung.gpkg* datasets
  - Problem tasks
  - Link to page

Task 1 -
Adding GeoPackage as Layers into QGIS

Task 2 -
Adding Buffers to Autobahn Layer

Task 3 -
Performing Union on the Buffer Areas

Task 4 -
Refining Code

Task 5 -
Selecting Features from Queries

Task 6 -
Styling and Cleaning Up

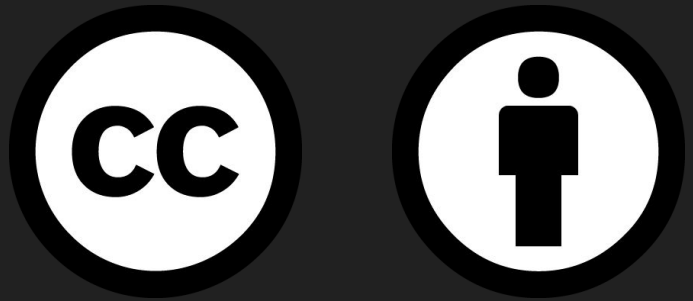
Bonus Task -
Interactive and Independent Script

Conclusion

- Processing Framework and QGIS
  - History
  - Features
  - Advantages, limitations
- Programming in Python
- Interest in QGIS and GIS
- Interest in Python and programming

Questions?

Please attribute Creative Commons with a link to creativecommon.org



Thank You!

Kang Zi Jing
GeoMeta Lab
Hochschule für Technik Rapperswil
zkang@hsr.ch