Геопроцессинг с SEXTANTE для QGIS

Обсудить в форуме Комментариев — 8

Эта страница опубликована в основном списке статей сайта
по адресу http://gis-lab.info/qa/sextante-qgis.html

Евгении, с благодарностью за поддержку и помощь в подготовке статьи.

Знакомимся с SEXTANTE — платформой геопространсвенного анализа для QGIS.

SEXTANTE — мощная и гибкая платформа для выполнения геопространственного анализа в QGIS. Она предоставляет доступ как к своим собственным функциям геообработки, так и к алгоритмам, реализованным в сторонних приложениях, упрощая выполнение анализа и делая его более продуктивным.

Изначально написанная на языке Java, доступная только пользователям gvSIG, SEXTANTE постепенно распространяла свое присутствие и на другие ГИС. В 2012 г. появилась версия на Python, разработанная для использования в QGIS. Она сразу вызвала большой интерес как среди пользователей, так и среди разработчиков, в сентябре 2012 года SEXTANTE была включена с состав QGIS и стала расширением ядра.

SEXTANTE в QGIS позволяет использовать основные возможности известных сторонних ГИС (SAGA GIS, GRASS GIS, TauDEM, OrfeoToolBox...) и алгоритмов, реализованные напрямую в SEXTANTE (fTools, MMQGISX...) в едином интерфейсе. А также представляет богатые возможности в автоматизации обработки данных: объединение однотипных шагов по выбору алгоритмов и данных в пользовательской модели анализа, написание и использование скриптов на Python, режим групповой обработки. Опытные пользователи могут еще больше увеличить производительность своей работы, комбинируя алгоритмы SEXTANTE и консоль Python.

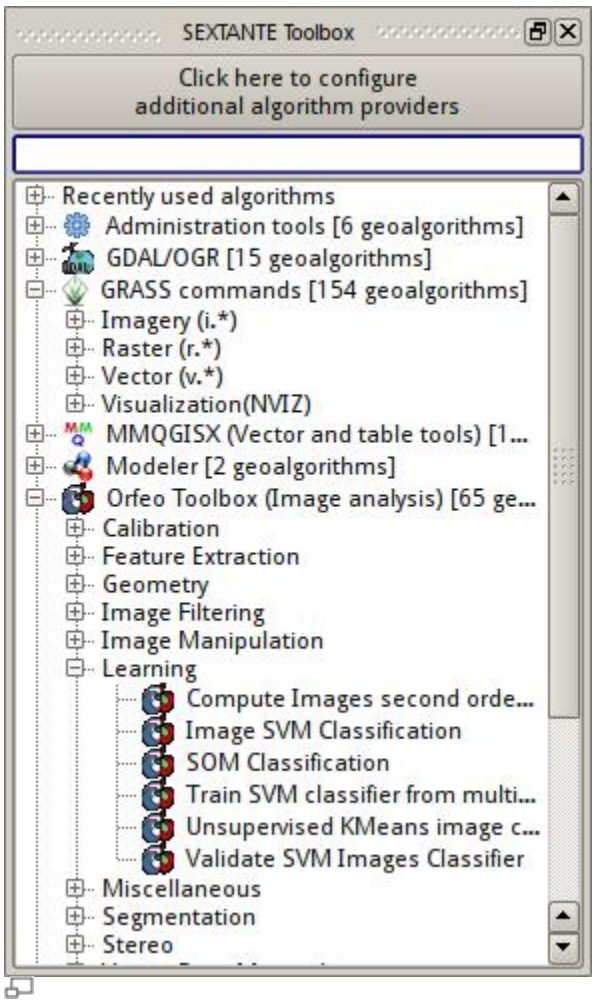
Содержание

Интерфейс

Пользовательский интерфейс SEXTANTE состоит из 4 основных частей. Каждая из них позволяет запускать алгоритмы на выполнение, выбор способа зависит от личных предпочтений, а также от используемых данных и проекта. Все элементы интерфейса (кроме диалога групповой обработки) доступны из меню «Анализ» («Analysis»).

Панель инструментов

Основным элементом SEXTANTE является панель инструментов (SEXTANTE Toolbox), именно с ней вы, скорее всего, и будете взаимодействовать наиболее часто. Здесь отображаются все активные алгоритмы, собранные в группы. Из панели инструментов можно запустить как отдельный алгоритм, так и групповую обработку.

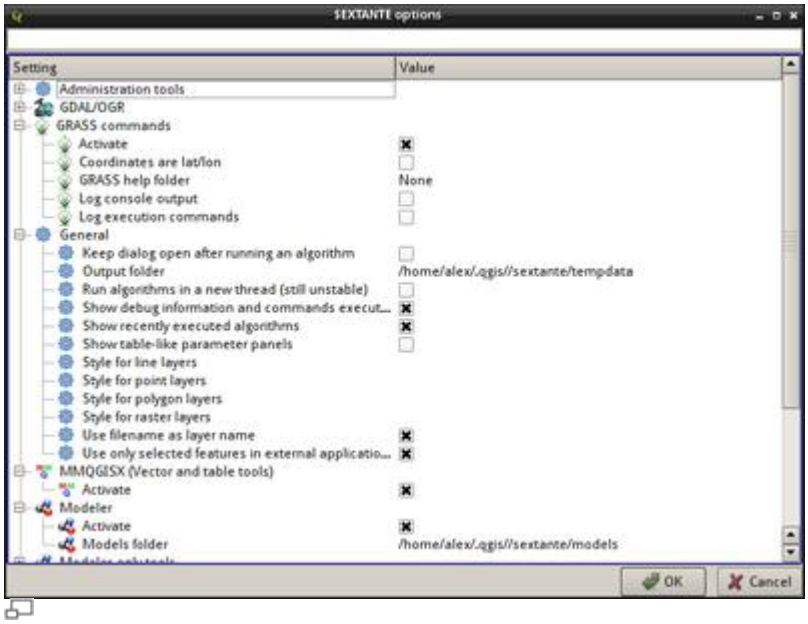


Каждая группа в панели представляет так называемый «провайдер», т.е. набор алгоритмов из одного источника (например, стороннего приложения). Часть представленных алгоритмов относится к сторонним приложениям (SAGA, OrfeoToolbox, TauDEM, GRASS...), часть — представляет алгоритмы реализованные напрямую в SEXTANTE (fTools, MMQGISX...). Кроме того, существует две специальные группы «Modeler» и «Scripts» предназначенные для пользовательских алгоритмов и позволяющие создавать свои собственные методы обработки.

В верхней части панели находится поле ввода, предназначенное для фильтрации списка доступных алгоритмов. Достаточно ввести слово или фразу, и в списке останутся только те алгоритмы, название или описание которых содержит введенный текст.

ВАЖНО! Изначально SEXTANTE содержит только описания базовых алгоритмов SAGA, GRASS, OTB и других сторонних приложений. Необходимо помнить, что все эти приложения не поставляются вместе с SEXTANTE. Поэтому запуск самих алгоритмов невозможен, хотя они и отображаются в структуре Toolbox. Для полноценной работы сторонние приложения необходимо дополнительно загружать и устанавливать.

Также необходимо отметить, что алгоритмы, зависящие от сторонних приложений, по умолчанию деактивированы. Включить их можно в диалоге настройки «Анализ (Analysis) → SEXTANTE options and configuration»



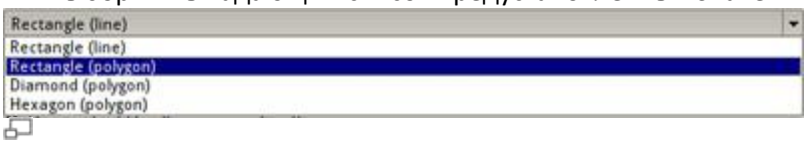
Для активации алгоритмов найдите в списке соответствующее приложение, разверните группу и поставьте галочку в строке «Activate»

Запуск алгоритма

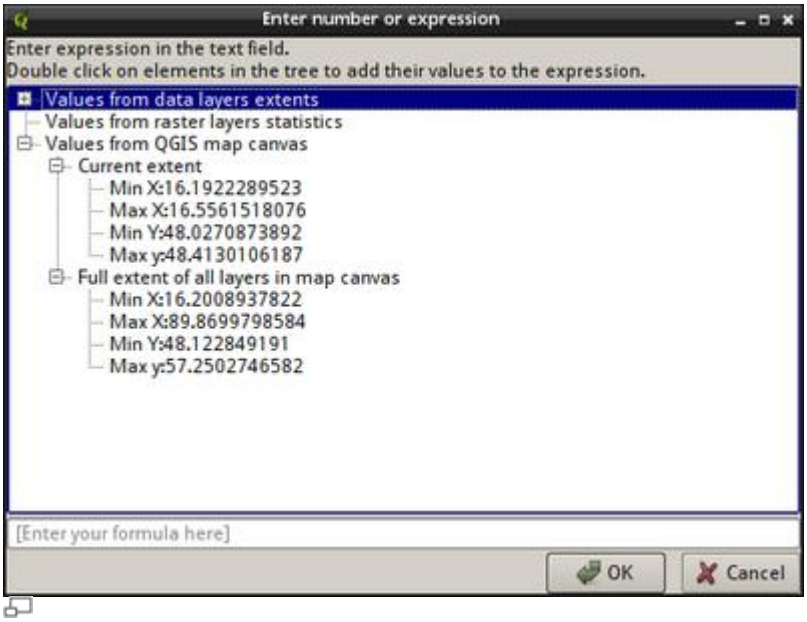
Двойной щелчок по названию алгоритма из панели инструментов вызовет диалог параметров, где указываются настройки алгоритма, исходные данные и путь для сохранения результатов.

В качестве параметров алгоритма могут выступать:

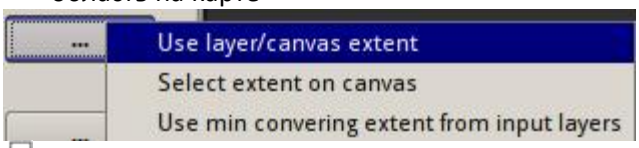
- растровый слой — это может быть как загруженный в QGIS слой, так и файл на диске
- векторный слой — аналогично растрам может выбираться как из списка загруженных слоёв, так в путем указания файла на диске. Примечание: если алгоритм также требует указания поля атрибутивной таблицы, выбирать можно только из загруженных в QGIS слоёв
- таблица — в настоящее время поддерживаются обычные таблицы без поддержки геометрии (geometryless) в форматах DBF и CSV
- выбор — выпадающий список предустановленных значений, из которого необходимо выбрать одно

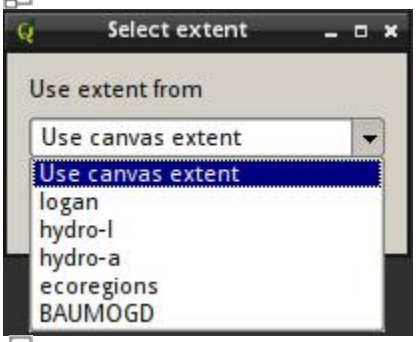


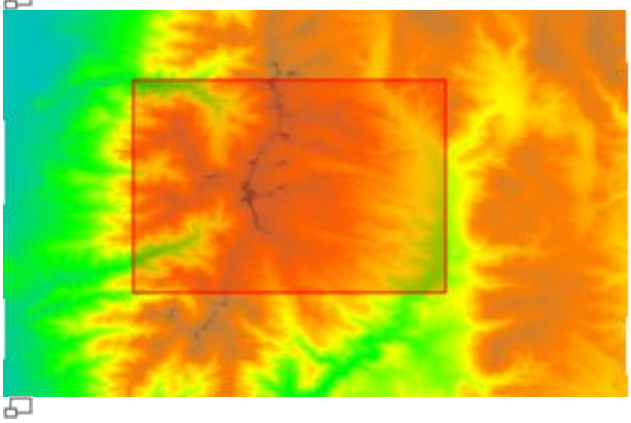
- числовое значение — целое или с плавающей точкой. Кнопка возле поля ввода открывает вспомогательный диалог, где можно ввести арифметическое выражение, а также выбрать некоторые величины, извлеченные из загруженных в проект данных (охват слоёв, размер пикселя и т. д.)



- диапазон числовых значений — задаётся минимальным и максимальным значениями
- текст — строковая величина
- имя поля атрибутивной таблицы — выбирается из списка полей таблицы или слоя, заданных другим параметром
- система координат — код EPSG или выбор системы координат из списка доступных
- охват — значения xmin, xmax, ymin, ymax. Может вводиться как вручную, так и выбираться из списка охватов загруженных слоёв и охвата карты. Также предусмотрена возможность указывать нужную область на карте







- список — несколько элементов (например, слоёв) выбранных из списка доступных
- пользовательская таблица — небольшая таблица, редактируемая пользователем (например, настройки скользящего окна для работы с растровыми данными).

В результате работы алгоритма могут быть созданы следующие виды данных:

- растровый слой

- векторный слой
- таблица
- HTML (используется для отображения текста и графики)
- файл (текстовый или любой другой файл, не подпадающий под перечисленные выше типы)

Формат выходных файлов в большинстве случаев определяется заданием расширения файла, а список поддерживаемых форматов в свою очередь зависит от алгоритма. Если не указать расширение или указать неподдерживаемое (в т.ч. ошибочное) расширение, результат будет сохранен в формате по умолчанию и соответствующее расширение будет автоматически добавлено к имени файла.

Если имя файла не указано, результаты сохраняются во временные файлы и удаляются при закрытии QGIS.

В настройках («Анализ (Analysis) → SEXTANTE options and configuration → General → Output folder») можно указать каталог для сохранения результатов. Этот путь будет использоваться, если указано только имя файла.

Помимо растровых и векторных слоёв, SEXTANTE также умеет обрабатывать результаты в формате HTML. Обычно, этот формат используется для отображения текста и изображений (например, графиков). Результаты в формате HTML отображаются по окончанию обработки в специальном окне «SEXTANTE results viewer» и доступны на протяжении всей рабочей сессии.

Некоторые алгоритмы могут создавать в процессе работы файлы в неподдерживаемых QGIS форматах (например, файлы LAS лидарной съемки). Такие файлы будут сохранены на диск, но не будут добавлены в проект QGIS. Для всех остальных типов файлов (векторные, растровые и табличные данные) предусмотрен переключатель, позволяющий управлять процессом загрузки этих данных в проект. По умолчанию, все результаты добавляются на карту.

SEXTANTE пока не поддерживает необязательные результаты, поэтому, если снять галку «Open output file after running algorithm», результирующий файл все равно будет создан (либо в виде временного файла, либо записан по указанному пользователем пути).

Построитель моделей (Modeler)

При выполнении какого-либо анализа дело редко ограничивается одной операцией/алгоритмом, чаще всего необходимо выполнять ряд последовательных действий. Т.е. анализ можно представить в виде последовательности шагов. Построитель моделей позволяет описать эту последовательность действий один раз и в дальнейшем обращаться к ней, как к единому целому. Такой подход позволяет сократить время на выполнение обработки большого количества данных. Не важно, сколько шагов необходимо выполнить для достижения результата, модель выполняется как один алгоритм.

Вызвать построитель моделей можно как из меню «Анализ (Analysis) → SEXTANTE modeler», так и из панели инструментов: в группе «Modeler», в разделе «Tools» находится единственный элемент для создания новой модели «Create new model».

Окно построителя моделей состоит из двух частей: слева находится панель вкладок, в ней выбирают составные элементы модели (исходные данные и алгоритмы), справа — рабочая область, где и создаётся модель.

Создание модели условно можно разделить на два этапа:

1. Определение исходных данных. Эти данные будут отображаться в диалоге настройки, и пользователь сможет их менять перед запуском модели.
2. Описание процесса. Устанавливаются связи между исходными данными и отдельными алгоритмами, определяется порядок выполнения действий.

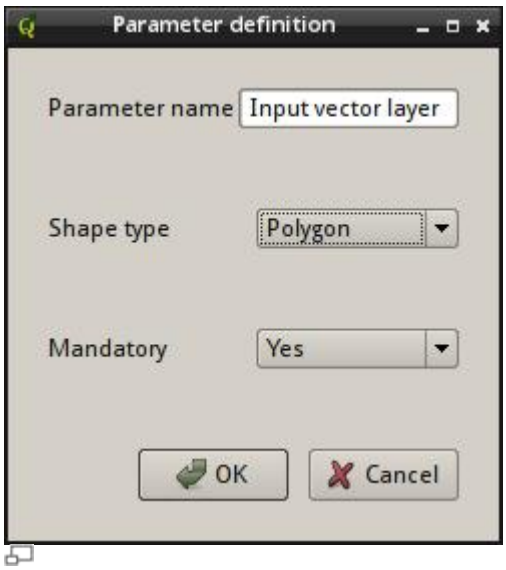
Определение исходных данных

Все доступные исходные данные собраны в левой части окна построителя моделей, на вкладке «Inputs». Поддерживаются следующие типы данных:

- растровый слой (`raster layer`)
- векторный слой (`vector layer`)
- таблица (`table`)
- поле таблицы (`table field`)
- строка (`string`)
- число (`number`)
- логическое значение (`boolean`)

Двойной щелчок на каком-либо элементе вызовет диалог настройки. В зависимости от типа, содержимое диалога будет отличаться. Общим для всех является поле названия параметра «Parameter name», где необходимо указать название нового элемента (этот текст будет использован как подпись поля в диалоге запуска модели). В зависимости от типа данных, остальные поля будут отличаться. Например, если выбрано числовое значение, помимо названия требуется указать значение по умолчанию, а также диапазон допустимых значений.

После выбора типа исходных данных в диалоговом окне необходимо указать, является ли ввод этих данных для пользователя необходимым или опциональным.



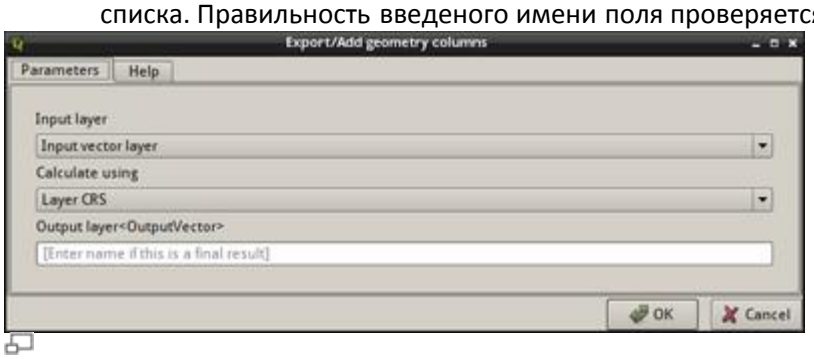
После заполнения полей, в рабочую область построения модели добавится новый блок, соответствующий одному из законченных элементов.

Описание процесса

После того, как заданы все исходные данные, можно приступать к описанию процесса анализа. Доступные алгоритмы находятся на вкладке «Algorithms» в левой части окна. Как и в случае панели инструментов, алгоритмы сгруппированы по провайдеру, присутствует и возможность фильтрации или поиска по списку.

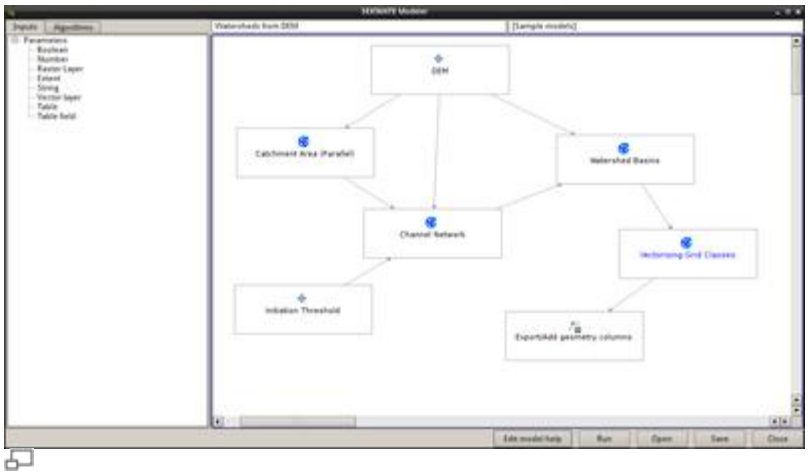
Для добавления алгоритма дважды щелкните по его имени в списке. Появится диалог настроек алгоритма, похожий на тот, что открывается при вызове алгоритма из панели инструментов. Но есть и отличия:

- растровые и векторные слои по прежнему выбираются из списка. Но в этом случае в списке находятся только слои, заданные в качестве исходных данных, а также слои, полученные в результате работы других алгоритмов модели;
- строки и числа могут быть как заданы пользователем, так и выбираться из списка значений, сгенерированных другими алгоритмами модели;
- поле таблицы. Список полей таблицы или слоя в момент создания модели не известен, поскольку он зависит от того, какой слой будет выбран пользователем. Поэтому имя поля в большинстве случаев необходимо вводить вручную. Исключением является ситуация, когда используется поле слоя или таблицы, определенных в качестве исходных данных, в этом случае возможен выбор из выпадающего списка. Правильность введеного имени поля проверяется на этапе выполнения модели.



После того, как все параметры алгоритма определены, нажимаем «OK», и блок алгоритма будет добавлен в рабочую область. Он будет связан с со всеми другими элементами (исходными данными и другими алгоритмами), которые предоставляют ему исходные данные.

Все элементы модели можно перетаскивать в пределах рабочей области, меняя их расположение и делая схему модели более легкой для восприятия. При этом связи между элементами будут обновляться автоматически.



В процессе создания модели можно в любое время проверить её работоспособность, нажав кнопку «Run». Но чтобы использовать модель из панели инструментов необходимо её сохранить и закрыть окно построителя.

Сохранение модели

Прежде чем сохранять модель, ей надо дать имя и указать в какой группе она будет находиться. Эти данные вносятся в два поля над рабочей областью построителя моделей.

Сохранение выполняется по нажатию кнопки «Save». По умолчанию модели сохраняются в папке models каталога пользователя, при желании путь можно изменить в настройках SEXTANTE («Анализ (Analysis) SEXTANTE options and configuration → Modeler → Models folder»).

При запуске SEXANTE считывает все описания моделей из этого каталога. Все загруженные модели отображаются во вложенных группах ветки «Modeler» панели инструментов, а также становятся доступны в

построителе моделей. Это значит, что любую модель можно использовать как обычный алгоритм при создании другой, более сложной модели.

Модель не будет загружена, если хотя бы один используемый в ней алгоритм не найден. Такое возможно, когда нужный провайдер деактивирован, и как результат, все его алгоритмы недоступны.

Редактирование моделей

Кроме создания новых моделей, SEXTANTE позволяет редактировать существующие. Начать редактирование модели можно двумя способами:

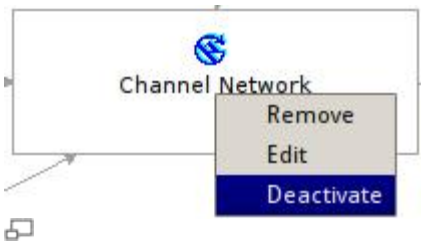
1. выделить модель в панели инструментов, вызвать контекстное меню и выбрать пункт «Edit model»
2. открыть построитель моделей, затем нажать кнопку «Open» и выбрать файл модели на диске

В процессе редактирования можно:

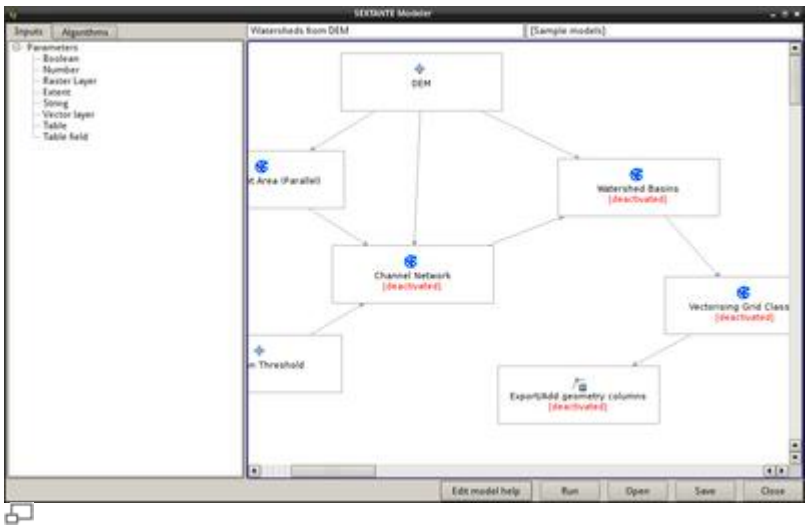
- менять связи между данными и алгоритмами
- добавлять новые и удалять существующие блоки
- временно деактивировать отдельные блоки

При удалении алгоритма необходимо помнить, что он может быть удален тогда и только тогда, когда нет других, зависящих от него, алгоритмов. Т.е. результаты удаляемого алгоритма нигде не используются.

Иногда возникает необходимость исключить определенные действия из модели не удаляя их. В этом случае соотвествующий алгоритм нужно деактивировать (вызвать контекстное меню и выбрать соответствующий пункт).



Деактивированные блоки помечаются красной подписью «deactivated», при этом, все зависящие от них алгоритмы также будут деактивированы.



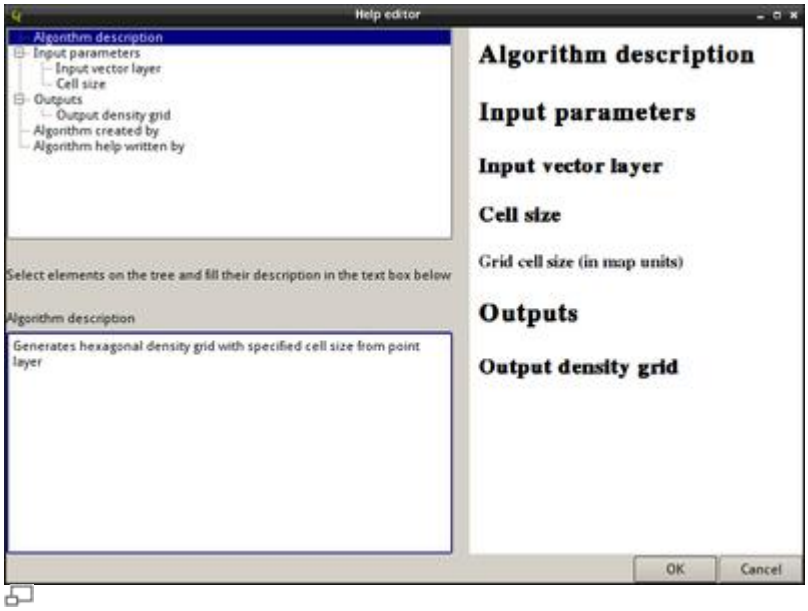
Активировать алгоритм очень просто, снова вызываем контекстное меню и выбираем пункт «Activate».

Документирование моделей

Хорошим тоном считается документирование созданных моделей, т.е. описание необходимых исходных данных, выполняемых действий и результата. Эта справка будет полезна другим пользователям, которые

захотят использовать созданную модель.

Нажатие на кнопку «Edit model help» откроет диалог редактирования описания модели.

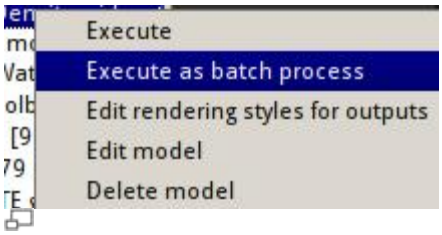


В правой части отображается простая HTML-страница, созданная на основе описаний исходных и выходных данных модели. Слева находится два поля: в верхнем выбирается элемент, описание которого необходимо добавить, в нижнем — добавляется текст. Описание модели сохраняется в тот же каталог, что и сама модель, автоматически при сохранении модели.

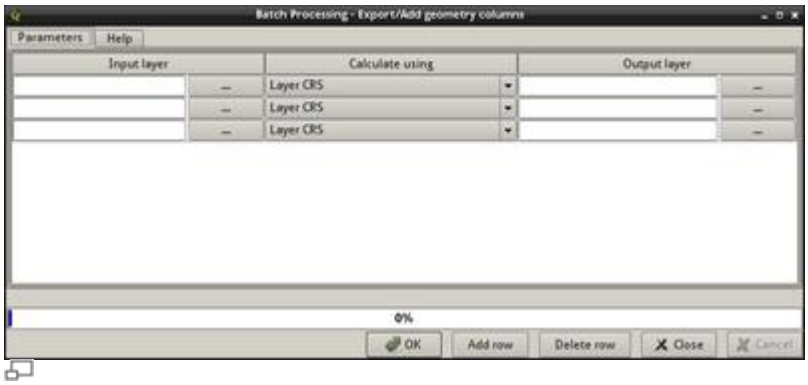
Групповая обработка

Все алгоритмы SEXTANTE, включая модели, могут использоваться для групповой обработки. Т.е. в этом случае они выполняются не один раз над одним набором данных, а несколько раз над разными наборами входных данных.

Чтобы запустить алгоритм в режиме групповой обработки выделите его в панели инструментов, вызовите контекстное меню и выберите пункт «Execute as batch process».



Запуск групповой обработки во многом схож с выполнением единичной операции. Отличие лишь в том, что параметры теперь задаются для каждой итерации обработки. Диалог настройки в этом случае принимает вид таблицы.



Каждая строка таблицы соответствует одному запуску алгоритма, в ячейках находятся параметры.

По умолчанию в таблице три строки, при необходимости добавить или удалить строки можно при помощи кнопок внизу окна. После того, как размер таблицы (число строк в ней) задан, можно приступать к её заполнению.

Заполнение таблицы параметров

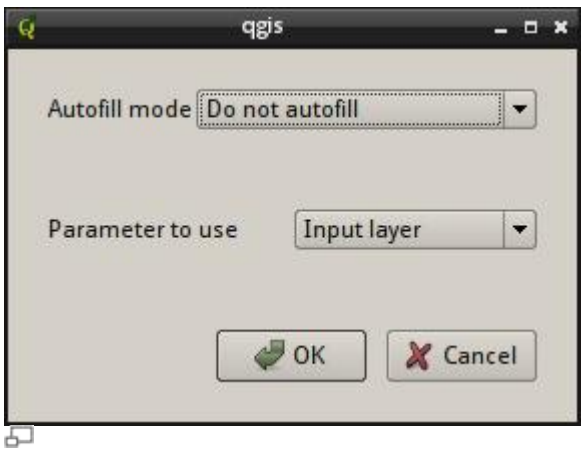
Большинство параметров задается либо вводом необходимого значения в поле, либо путем выбора значения из выпадающего списка.

Необходимо помнить, что в случае групповой обработки исходные слои (векторные или растровые) и таблицы загружаются из файлов на диске, а не берутся из открытых в QGIS. Поэтому, групповая обработка может быть запущена в любое время, даже при пустом проекте.

Указать имена исходных файлов можно как вводя путь к файлу в поле, так и воспользовавшись кнопкой «...» возле соответствующего поля. В последнем случае откроется диалог выбора файлов с поддержкой множественного выбора. Если параметр является единичным файлом, то выбранные значения будут добавлены каждый в свою строку таблицы, при этом недостающие строки будут добавлены автоматически. Если же параметр принимает несколько величин, все выбранные файлы будут добавлены в одну строку.

В отличие от обычного выполнения алгоритма, все результаты сохраняются в файлы, создание временных файлов не допускается. Указать имя выходного файла можно как вручную, так и воспользовавшись диалогом выбора.

После выбора выходного файла появится ещё один диалог, позволяющий автоматически заполнить остальные ячейки.



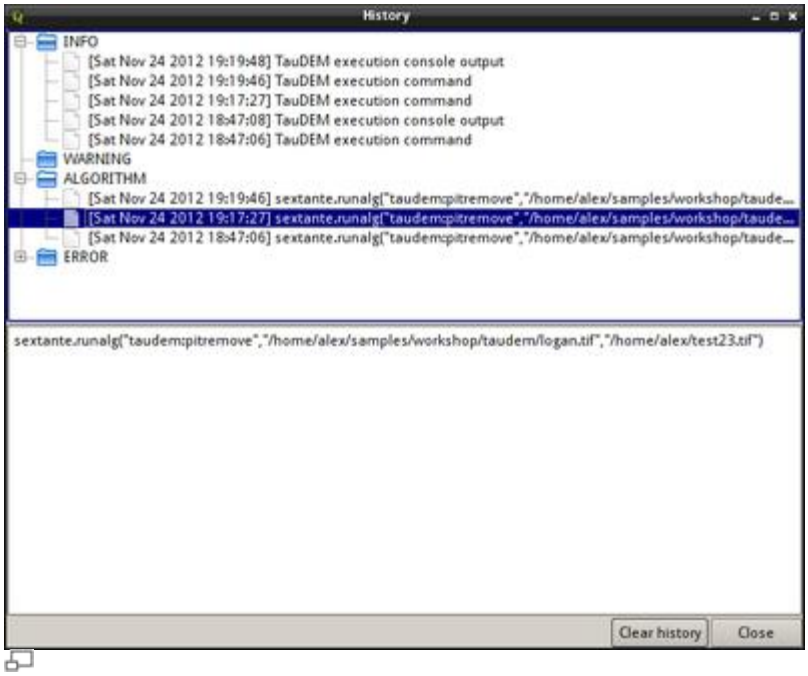
Если выбрано значение «Do not autofill» (по умолчанию), SEXTANTE просто вставит в заданную ячейку выбранное имя файла. Если же выбрано любое другое значение, будут заполнены все ячейки. При этом имена файлов будут сформированы на основе указаного критерия автозаполнения.

Для автозаполнения могут использоваться как последовательные номера, так и значения других полей этой же строки. Такой подход значительно ускоряет заполнение таблицы параметров групповой обработки.

Менеджер истории

Каждый раз, когда выполняется алгоритм SEXTANTE, информация об этом сохраняется в истории операций. Помимо называния алгоритма, записываются все параметры, а также дата и время выполнения. Таким образом, при необходимости можно легко восстановить последовательность действий и воспроизвести их ещё раз.

Просмотреть историю действий можно в Менеджере истории SEXTANTE («Анализ (Analysis) → SEXTANTE history and log»).



Все события отображаются в хронологическом порядке, найти нужные данные достаточно просто. Информация сохраняется в виде выражения командной строки, даже если алгоритм был запущен из панели инструментов. Это делает менеджер истории полезным также при изучении возможностей командной строки SEXTANTE, т.к. можно запустить алгоритм из панели инструментов, а затем посмотреть в менеджере истории как его вызывать из командной строки.

Выполненные действия можно не только просматривать, любое из них можно повторно запустить двойным щелчком по соответствующей строке в менеджере истории.

В этом же окне можно найти и другую полезную информацию: сообщения об ошибках (error), предупреждения (warning) и дополнительную информацию (info). В случае, если что-то не работает, стоит заглянуть в раздел «Errors».

Взаимодействие со сторонними алгоритмами обычно выполняется через интерфейс командной строки. И хотя консоль пользователям не показывается, весь её вывод перенаправляется и сохраняется в разделе «Info» менеджера истории. Так что, если что-то пошло не так или есть сомнения в правильности работы алгоритма, в этом разделе можно найти все сообщения сгенерированные алгоритмом.

В раздел «Warning» чаще всего попадают различные предупреждения, информирующие о потенциальных проблемах с данными (например, несовпадение систем координат). Если вы получили неожиданный результат, проверьте, нет ли в этом разделе сообщений.

Использование SEXTANTE в консоли

Командная строка позволяет опытным пользователям значительно увеличить производительность работы и выполнять сложные операции, которые невозможно выполнить, используя графический интерфейс SEXTANTE.

У SEXTANTE нет отдельной командной строки, вместо этого все команды SEXTANTE доступны из консоли Python QGIS. Это значит, что в нашем распоряжении все возможности API QGIS и SEXTANTE одновременно.

Код, выполняемый в консоли Python, даже если он не вызывает ни одного алгоритма SEXTANTE, может быть преобразован в новый алгоритм SEXTANTE, который в дальнейшем может вызываться из панели инструментов или использоваться в построителе моделей, как любой другой алгоритм.

Запуск алгоритмов в консоли

Первое, что нужно сделать при использовании SEXTANTE из командной строки — импортировать модуль SEXTANTE

```
>>>import sextante
```

Теперь можно использовать SEXTANTE в консоли, а именно запускать алгоритмы на выполнение. Для этого предназначен метод `runalg()`, которому передаётся имя алгоритма в качестве первого параметра и затем переменный набор дополнительных параметров, зависящий от алгоритма.

Таким образом, первое, что необходимо знать — это имя алгоритма. Обратите внимание, в качестве имени алгоритма использует не тот текст, который отображается в панели инструментов, а уникальное внутренее имя алгоритма. Узнать правильное имя можно при помощи метода `alglist()`. Просто введите в консоли

```
>>>sextante.alglist()
```

на консоль будет выведено что-то вроде этого

```
D-Infinity Avalanche Runout-------------------------->taudem:d-infinityavalancherunout
D-Infinity Concentration Limited Accumulation-------->taudem:d-
infinityconcentrationlimitedaccumulation
D-Infinity Contributing Area------------------------->taudem:d-infinitycontributingarea
D-Infinity Decaying Accumulation--------------------->taudem:d-
infinitydecayingaccumulation
D-Infinity Distance Down----------------------------->taudem:d-infinitydistancedown
D-Infinity Distance Up------------------------------->taudem:d-infinitydistanceup
D-Infinity Flow Directions--------------------------->taudem:d-infinityflowdirections
D-Infinity Reverse Accumulation---------------------->taudem:d-
infinityreverseaccumulation
D-Infinity Transport Limited Accumulation------------>taudem:d-
infinitytransportlimitedaccumulation
D-Infinity Transport Limited Accumulation - 2-------->taudem:d-
infinitytransportlimitedaccumulation-2
D-Infinity Upslope Dependence----------------------->taudem:d-
infinityupslopedependence
D8 Contributing Area--------------------------------->taudem:d8contributingarea
```

Метод `alglist()` выводит список всех активных алгоритмов, отсортированный в алфавитном порядке (слева название, справа — внутренее имя).

Чтобы не просматривать полный список в поисках нужного алгоритма, можно передать методу строковый параметр, и будут выведены только те алгоритмы, в описании которых есть заданная подстрока. Например

```
>>>sextante.alglist("slope")
D-Infinity Upslope Dependence----------------------->taudem:d-
infinityupslopedependence
D8 Extreme Upslope Value---------------------------->taudem:d8extremeupslopevalue
Slope Area Combination------------------------------>taudem:slopeareacombination
Slope Average Down---------------------------------->taudem:slopeaveragedown
Slope Over Area Ratio------------------------------->taudem:slopeoverarearatio
```

После того как мы узнали имя алгоритма, необходимо выяснить какие параметры ему необходимы для работы, т.е. узнать синтаксис вызова. Для этого в SEXTANTE предусмотрен метод `alghelp()`, который выдаёт список входных параметров, а также список рзультирующих файлов, генерируемых агоритмом. Единственный аргумент метода — внутренее имя алгоритма, описание которого мы хотим получить.

Например, для алгоритма `mmqgisx:creategrid` будет выведена следующая информация

```
>>> sextante.alghelp("mmqgisx:creategrid")
ALGORITHM: Create Grid
   HSPACING <ParameterNumber>
   VSPACING <ParameterNumber>
   WIDTH <ParameterNumber>
   HEIGHT <ParameterNumber>
   CENTERX <ParameterNumber>
   CENTERY <ParameterNumber>
   GRIDTYPE <ParameterSelection>
```

```
   SAVENAME <OutputVector>
```

Видно, что алгоритм называется «Create Grid» (этот текст используется в панели инструментов), принимает 7 параметров (6 числовых и 1 выбирается из списка предустановленных значений) и генерирует один результат (векторный слой).

Теперь у нас есть вся необходимая для запуска алгоритма информация. Как уже было сказано, запуск алгоритма выполняется при помощи метода `runalg()`. Он имеет следующий синтаксис:

```
sextante.runalg(name_of_the_algorithm, param1, param2, ..., paramN, output1, output2,
..., outputN)
```

Список параметров и результатов зависит от алгоритма, и должен указываться в том порядке, в котором их выдаёт метод `alghelp()`.

Значения параметров вводятся в зависимости от их типа:

- растровый слой, векторный слой или таблица. Можно указывать имя слоя QGIS (если слой загружен в проект), путь к файлу (если слой не загружен). Также можно передавать переменную — экземпляр соответствующего класса QGIS (например, `QgsVectorLayer`). Если параметр является необязательным и не используется — просто укажите `None`
- выбор из списка предустановленных значений. Значение указывается как целочисленный индекс, соответствующий значению. Получить список доступных значений и соответствующие им индексы можно при помощи метода `algoptions()`. Например:

```
>>> sextante.algoptions("mmqgisx:creategrid")
   GRIDTYPE(Grid Type)
      0 - Rectangle (line)
      1 - Rectangle (polygon)
      2 - Diamond (polygon)
      3 - Hexagon (polygon)
```

Обратите внимание, нумерация начинается с нуля.

- список. Список нескольких элементов, разделенных точкой с запятой. В качестве элементов могут быть как имена объектов, так и пути к файлам
- имя поля. Регистрозависимое название поля атрибутивной таблицы
- пользовательская таблица. Задается в виде списка значений, разделенных запятой и заключенного в кавычки. Значения задаются слева направо и сверху вниз. Также можно передавать двумерный массив соответствующей размерности
- система координат. Указывается код EPSG нужной системы координат
- охват. Значения xmin, xmax, ymin и ymax, разделенные запятыми. Например: `433265.937051,449034.096697,4635579.39814,4650535.96133`

Числовые и логические параметры могут иметь значения по умолчанию. Чтобы использовать их, укажите `None` в соответствующей позиции.

Для результатов указывается путь к файлу, или, если он должен сохраняться во временный файл, — `None`. Формат файла определяется по заданному расширению, если указано неправильное или неподдерживаемое расширение, будет использован формат по умолчанию, и соответствующее расширение будет добавлено к имени файла.

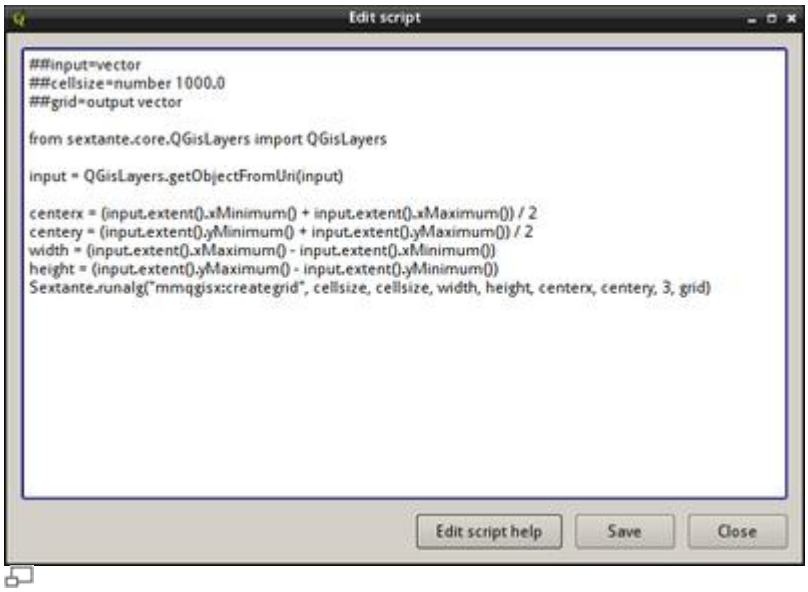
В отличие от запуска алгоритма из панели инструментов, по окончанию обработки результаты не загружаются в проект автоматически. Загрузить результаты можно либо используя QGIS API, либо обратившись к методам SEXTANTE.

Метод `runalg()` возвращает словарь, ключами в нем будут названия результатов, а значениями — пути к соответствующим файлам. Добавить все результаты на карту можно, передав этот словарь методу `loadFromAlg()`. Для загрузки единичного результата используется метод `load()`.

Создание скриптов

Пользователь может создавать свои собственные алгоритмы, написав соответствующий код на Python и

добавив к нему несколько строк со специальной информацией для SEXTANTE. Простой редактор скриптов можно вызвать из панели инструментов SEXTANTE: группа «Scripts», раздел «Tools», пункт «Create new script». В появившемся окне вводится код скрипта, а затем он сохраняется в каталог `scripts` (по умолчанию, можно изменить в настройках) папки `sextante` в домашнем каталоге пользователя QGIS. Расширение файлов .py, имя файла будет использоваться в качестве имени алгоритма в панели инструментов (при этом расширение отбрасывается, а подчеркивания заменяются пробелами).



Также можно создавать скрипты и в сторонних текстовых редакторах, а затем сохранять их в каталог скриптов SEXTANTE.

Рассмотрим простой скрипт для расчета Topographic Wetness Index (TWI) из DEM

```
##DEMO=group
##dem=raster
##twi=output
ret_slope = sextante.runalg("saga:slopeaspectcurvature", dem, 0, None, None, None,
None, None)
ret_area = sextante.runalg("saga:catchmentarea(mass-fluxmethod)", dem, 0, False, False,
False, False, None, None, None, None, None)
sextante.runalg("saga:topographicwetnessindex(twi), ret_slope'SLOPE', ret_area'AREA',
None, 1, 0, twi)
```

Как видно, ничего сложного в нем нет, вызывается три алгоритма SAGA. Если вы внимательно читали предыдущий раздел, разобраться в коде будет достаточно легко. Сейчас наибольший интерес для нас представляют первые три строчки, начинающиеся символами ##.

Строки, начинающиеся с ## необходимы SEXTANTE для правильной работы со скриптом, именно они позволяют выполнять скрипт, а также использовать его в моделях, как и любой другой алгоритм. Информация, записанная в этих строках имеет вид

```
название_параметра=тип_параметра дополнительные_значения
```

Обратите внимание, что в названии параметра вместо пробелов необходимо использовать нижнее подчеркивание. Тип параметра или результата может содержать пробел, например, `output vector`.

В первой строке мы задаем название группы, в которую будет добавлен наш скрипт. Затем идет список параметров и результатов.

В скриптах можно использовать следующие типы параметров:

- `raster`. Растровый слой
- `vector`. Векторный слой
- `table`. Таблица

- `number`. Число (целое или с плавающей запятой). В поле дополнительных значений необходимо указать значение по умолчанию. Например, `depth=number 2.4`
- `string`. Текстовая строка, как и в случае числового параметра, в поле дополнительных значений необходимо указать значение по умолчанию. Например, `name=string NUMPOINTS`
- `boolean`. Логическая величина. Необходимо указать, какое она должна принимать значение по умолчанию: `True` или `False`. Например, `verbose=boolean True`
- `multiple raster`. Набор растровых слоёв
- `multiple vector`. Набор векторных слоёв
- `field`. Поле атрибутивной таблицы, необходимо указать слой или таблицу, из которого будет браться поле. Например, если задан параметр `mylayer=vector`, то поле атрибутивной таблицы слоя `mylayer` описывается так `myfield=field mylayer`
- `folder`. Каталог
- `file`. Файл (текстовый или любой другой файл, не подпадающий под перечисленные выше типы)

Название параметра будет использоваться как в качестве подписи соответствующего поля ввода при запуске алгоритма, так и в качестве переменной внутри скрипта, которой будет присвоено введенное пользователем значение.

При использовании названия параметра в качестве подписи, SEXTANTE заменяет подчеркивания на пробелы. Т.е. если мы назовем параметр «Search_radius», то при запуске алгоритма пользователь увидит надпись «Search radius».

Растровые и векторные слои, а также таблицы задаются строками, содержащими полный путь к соответствующим файлам на диске. Получить из них экземпляры классов `QgsVectorLayer` или `QgsRasterLayer` можно при помощи метода `sextante.getObjectFromUri()`. Множественные объекты представляются строкой, содержащей разделенные точкой с запятой пути к файлам.

Результаты описываются точно также, с использованием следующих типов:

- `output raster` (выходной растр)
- `output vector` (выходной вектор)
- `output table` (выходная таблица)
- `output html` (выходные данные в формате html)
- `output file` (выходной файл: текстовый или любой другой файл, не подпадающий под перечисленные выше типы)
- `output number` (выходное число)
- `output string` (выходная текстовая строка)

В переменные-результаты всегда записывается строка, содержащая полный путь к файлу. Если пользователь не указал путь для сохранения результата, это будет путь к временному файлу.

ВАЖНО! SEXTANTE будет пытаться загрузить в проект все описанные в заголовке скрипта результаты по окончанию обработки. Поэтому при создании скрипта не нужно использовать методы `load()` и `loadFromAlg()`, они нужны только при работе в консоли.

Если алгоритм будет выполняться длительное время, стоит отображать прогресс выполнения. Для этого служит объект `progress`: метод `setText()` позволяет выводить различные информационные сообщения, а метод `setPersentage()` — менять величину индикатора прогресса.

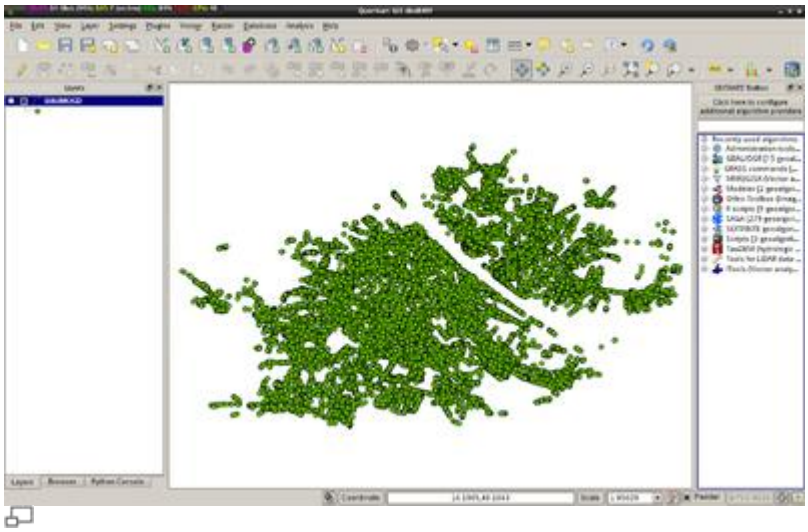
Документирование скриптов

Как и в случае моделей, скрипт можно (и нужно) сопровождать документацией. Добавить описание к любому скрипту можно, открыв его в редакторе скриптов SEXTANTE и нажав кнопку «Edit script help». Принцип создания справки совпадает с аналогичным процессом для моделей и в дополнительных пояснениях не нуждается.

Пример: создание карты плотности с использованием

шестиугольной сетки

Для визуализации плотности точек обычно используют теплокарты (heatmaps). Но есть и другие подходы, один из них — использование шестиугольной сетки. Для примера возьмем данные о количестве деревьев в г. Вена (предоставлены Viena Tree Cadastre).



В общем случае алгоритм можно описать так: создаем сетку с шестиугольными ячейками, подсчитываем количество точек (деревьев) в каждой ячейке, получаем красивую карту.

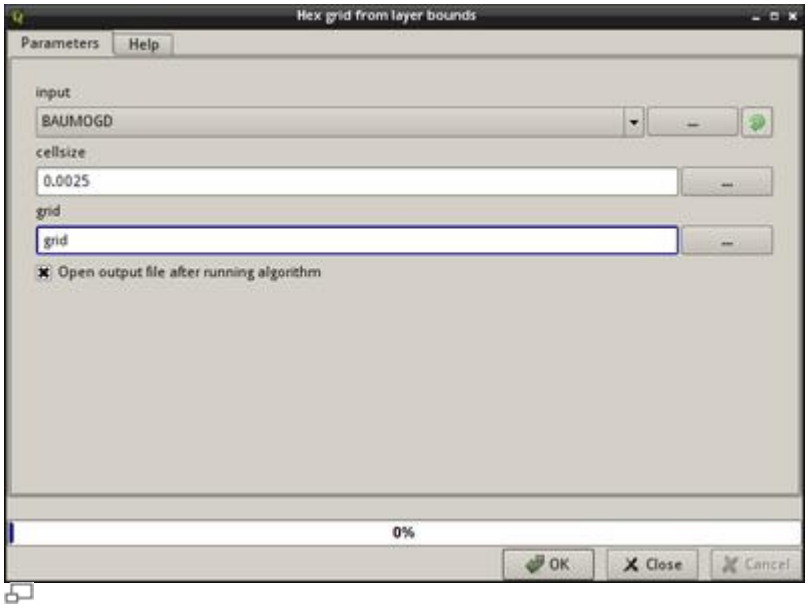
Первое, что нам нужно сделать — построить сетку. Для этого можно использовать алгоритм Create Grid из группы MMQGISX. Но у него есть один недостаток: необходимо вручную указать ширину и высоту сетки, а также координаты её центра. Это значительно усложняет возможность дальнейшей автоматизации процесса. Поэтому мы напишем свой алгоритм, который будет создавать сетку по границам слоя. Само-собой, писать его мы будем не с нуля, а воспользуемся уже имеющимся в нашем распоряжении алгоритмом MMQGISX.

Вот код скрипта:

```
# описываем параметры и результаты

##input=vector
##cellsize=number 1000.0
##grid=output vector

# тело алгоритма

from sextante.core.QGisLayers import QGisLayers

# в переменной input записан путь к векторному слою
# преобразуем его в экземпляр QgsVectorLayer, чтобы получить охват
input = QGisLayers.getObjectFromUri(input)

# вычисляем необходимые величины и вызываем алгоритм MMQGGISX
centerx = (input.extent().xMinimum() + input.extent().xMaximum()) / 2
centery = (input.extent().yMinimum() + input.extent().yMaximum()) / 2
width = (input.extent().xMaximum() - input.extent().xMinimum())
height = (input.extent().yMaximum() - input.extent().yMinimum())
Sextante.runalg("mmqgisx:creategrid", cellsize, cellsize, width, height, centerx,
centery, 3, grid)
```

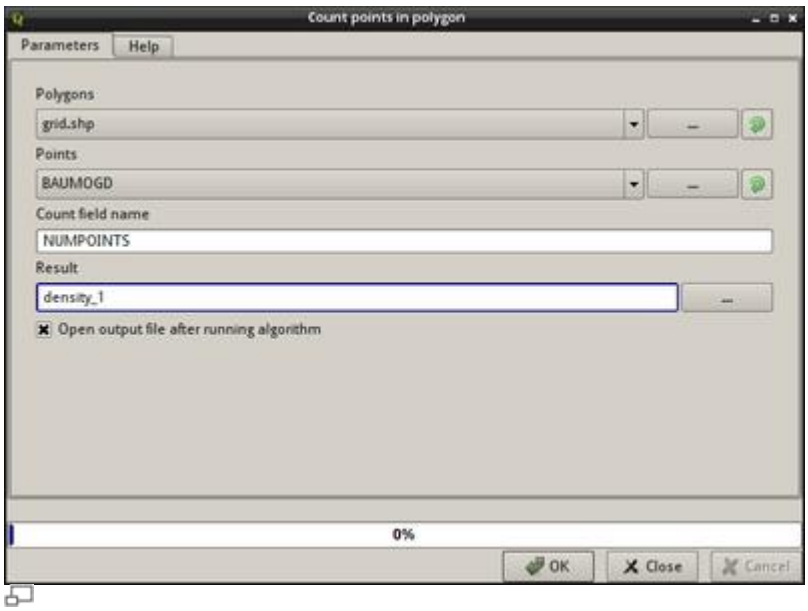
Таким образом, наш алгоритм по сути является «оберткой» над существующим и позволяет строить сетку без необходимости задавать её размер.

После запуска алгоритма появится диалог настройки, такой же как и при запуске любого другого алгоритма SEXTANTE.

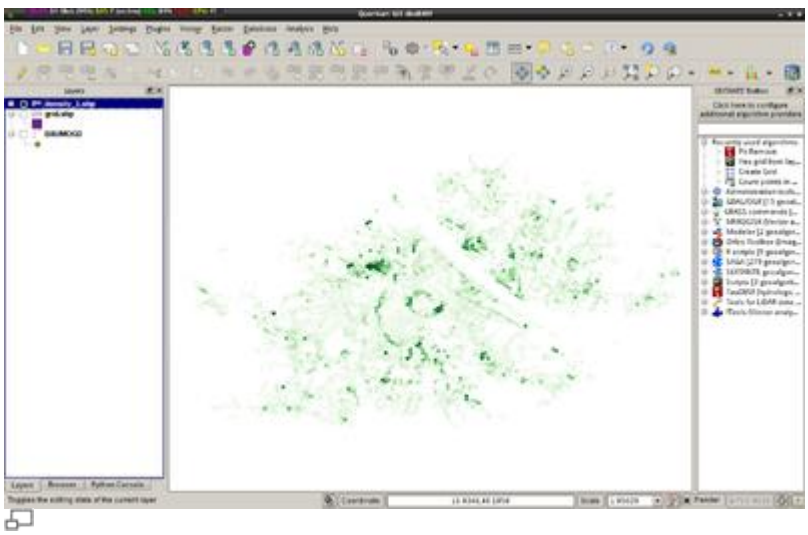


В качестве исходного слоя выбираем точечный слой с данными кадастра, размер ячейки задаём равным 0.0025 и после непродолжительного ожидания получим сетку, охватывающую исходный слой.

Чтобы подсчитать количество точек (деревьев) в каждой ячейке сетки, воспользуемся алгоритмом Count points in poylygons из fTools.

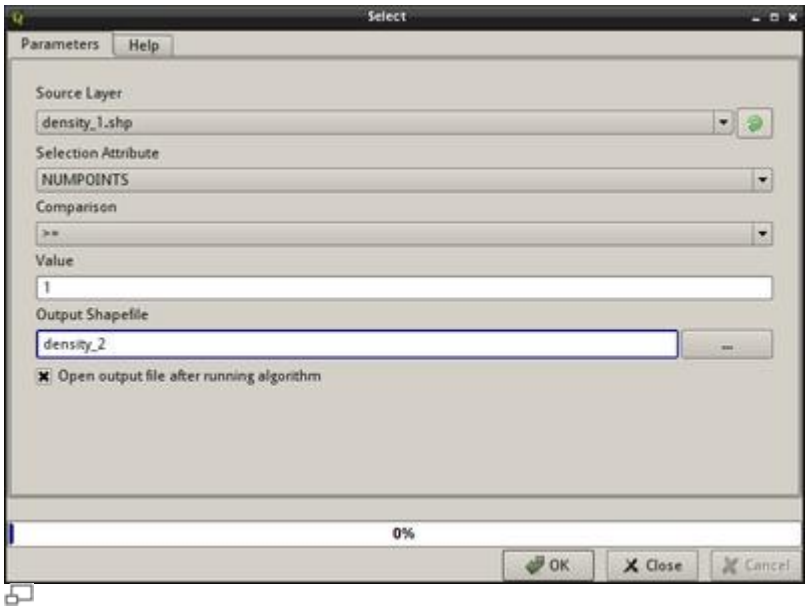


Результат, отрисованный градуированным знаком, показан ниже.

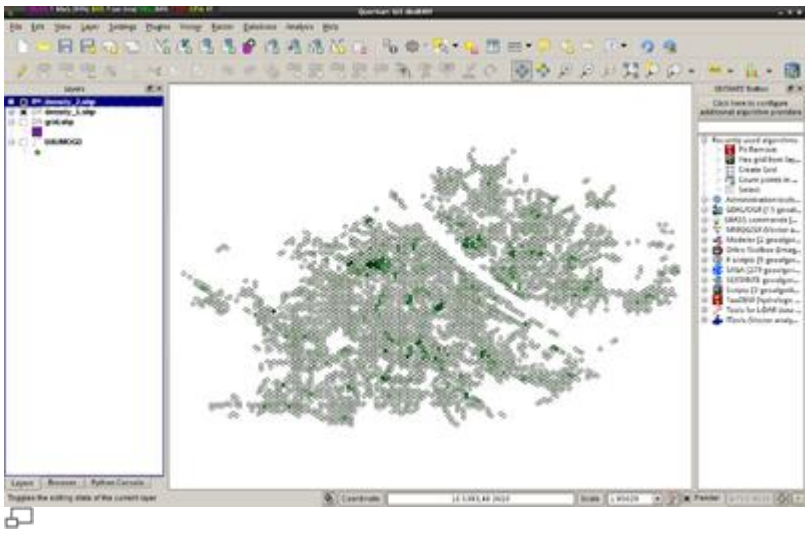


Примечание: приведенный выше скрипт теперь входит в базовую поставку SEXTANTE и доступен под именем «Hex grid from layer bounds» в разделе «Example scripts» группы «Scripts».

Если сделать пустые ячейки прозрачными, будет видно, что их достаточно много. Было бы неплохо удалить их, уменьшив размер слоя. Для этой задачи как нельзя лучше подходит алгоритм Select из MMQGISX. С его помощью мы получим новый слой, содержащий только ячейки хотя бы с одной точкой.



Здесь специально использован немного другое оформление (границы ячеек черного цвета), чтобы показать как уменьшилось число ячеек после удаления пустых.



Все эти действия можно скомпоновать в виде модели и затем выполнять за один шаг, задавая только исходный слой и размер ячейки.

Откроем построитель моделей, сразу зададим имя модели и группу, в которой она будет сохранена. Добавим исходные данные:

- векторный слой. В качестве подписи зададим «Input vector layer», тип слоя установим в Points, т.к. строить карту плотности мы будем только для точечных слоёв.
- число (размер ячейки сетки). Подпись «Cell size» должна быть достаточно иформативной, в качестве значения по умолчанию выберем 1000

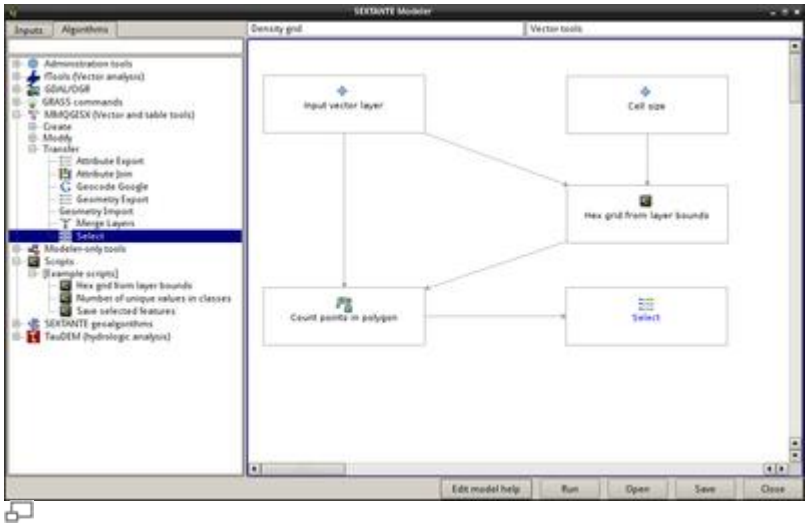
Исходные данные заданы, можно начинать составлять алгоритм. Переключимся на вкладку «Algorithms» и начнем добавлять алгоритмы в том порядке, в котором мы их запускали ранее:

- сначала добавим созданный нами алгоритм «Hex grid from layer bounds». Он принимает два параметра: векторный слой (выбираем из списка добавленный ранее слой) и числовое значение

(также выбираем из списка). Обратите внимание, мы ничего не пишем в поле «grid<OutputVector>», т.к. созданая на этом шаге сетка не является окончательным результатом и не нужна нам.

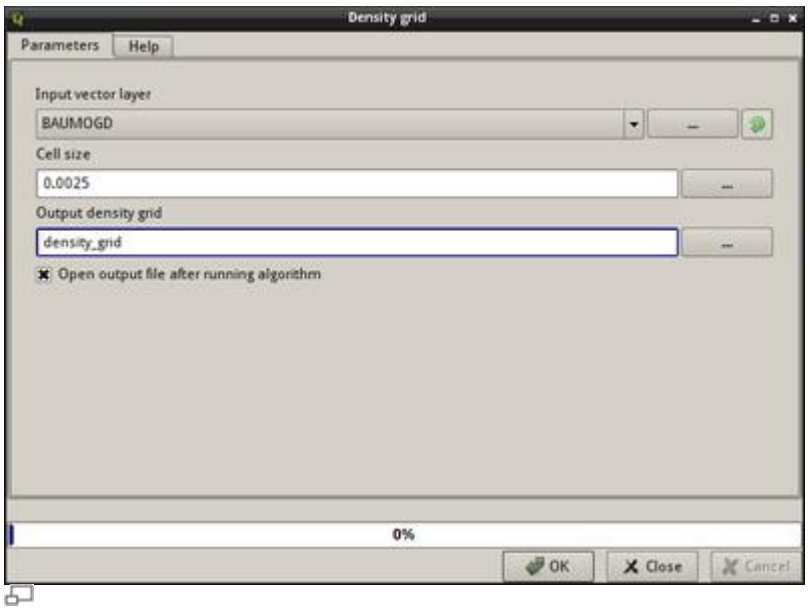
- находим в списке алгоритмов «Count points in polygons» и добавляем его. В качестве полигонального слоя указываем результат работы алгоритма «Hex grid from layer bounds» (подписан как «grid from algorithm 0 (Hex grid from layer bounds)»), а в качестве слоя точек — исходный векторный слой. Название поля с числом точек можно оставить без изменений либо откорректировать, мы используем значение по умолчанию. Поле «Result» опять оставляем пустым
- добавляем последний алгоритм — Select из MMQGISX. В качестве исходного слоя указываем результат работы алгоритма «Count points in polygon» («Result from algorithm 1 (Count points in polygon)»). Т.к. имена полей во время создания модели не доступны, вводим имя поля (NUMPOINTS) самостоятельно. Указываем условие сравнения >=1. И, поскольку это наш окончательный результат, задаем имя (подпись) выходного файла, например, «Output density grid»

Осталось только разместить блоки так, чтобы модель легко читалась и сохранить её.



Как видите, созданный нами алгоритм, может использоваться в модели. Более того, сама модель может быть использована как элемент другой модели, а созданный ранее скрипт, как часть более сложного скрипта.

При запуске модели появится вот такой диалог настройки



Обратите внимание на подписи полей ввода, для них используется текст, введенный при создании модели.

Как видите, создав модель, мы достигли того же результата, как и последовательным выполнением нескольких алгоритмов.

Ссылки

- Домашняя страница SEXTANTE
- Документация по QGIS

Обсудить в форуме Комментариев — 8

Последнее обновление: 2014-05-15 01:45

Дата создания: 27.12.2012
Автор(ы): Александр Бруй