Tutorial to ORNAMENTAL tool

1 Введение

Добро пожаловать в ORNAMENTAL. Программа ORNAMENTAL написана на C++ и выполняется в ОС линукс в режиме командной строки. Результаты работы программы выводятся в текстовый и графический терминалы.

2 Инстолляция программы

2.1 Требования к оборудованию

Требования программы к оборудованию невелики, выполняться программа может на любом ПК. Например, разработка и тестирование программы выполнялись на ПК Intel Core i7-3770K CPU @ 3.50GHz с 32G памяти. При этом, выполнение любой команды было мгновенным (<1s), потребление памяти в режиме демонстрации не превышало 0.1%.

2.2 Требования к установленным программам

Программа разработана в рамках ОС Ubuntu 16.04, без использования каких либо особенностей данной операционной системы. Ожидается, что она будет работать в любой другой unix подобной системе, либо в имитирующем ее окружении.

В системе должны быть установлены:

- c++ компилятор, поддерживающий стандарт C++17 (g++-7 –version >= 7.5.0)
- boost-1.74 (>= boost-1.72)
- gsl (>= 2.1)
- ncurses (>= libncurses5)
- gnuplot (>= 4.6)
- любая программа просмотра pdf файлов
- любая программа конвертации svg файлов в pdf файлы

Программа просмотра pdf формально может быть любой (кроме evince и okular, например, можно использовать libreoffice и firefox), однако в реальности есть ограничение связанное с удобством использования. Поскольку при работе ornamental генерится большое число графиков, необходимым условием комфортной работы является то, чтобы программа просмотра автоматически перезагружала измененные на диске файлы.

2.3 Компиляция и запуск программы

2.3.1 Компиляция из командной строки

После загрузки и распаковки файла ornamental.zip, зайдите в директорию ornamental: *cd ornamental*

Выполните компиляцию командами из командной строки:

```
g++-7 -c -O2 -std=c++17 -I./include -o main.o main.cpp
g++-7 -o orn main.o -L /usr/local/lib -lgsl -lgslcblas
      -lcurses
```

Запустите на выполнение командой:

./orn или

./orn <file>,

где <file> является именем текстового файла, содержащего команды настройки и предварительного выполнения. Образец такого файла имеет имя *ornamental.init*.

2.3.2 Компиляция в среде разработки C++

Компиляция и запуск на выполнение можно выполнить в любой среде разработки C++ программ. Такие пользователи не нуждаются в описании того, как это сделать.

Примечание. При использовании QtCreator в *.pro файл необходимо вставить строку CONFIG += c++1z, которая подключает C++17.

2.3.3 Входные данные программы

Программа обрабатывает файлы формата *.csv (в которых данные разделены символом ','), а также текстовые файлы формата *.orn. Формат *.csv следует описанию данных в виде таблицы 1, а формат *.orn – в виде таблицы 2 (см. Fomin ES and Fomina TI. A nonparametric model for analysis of flowering patterns of herbaceous multi-flowered monocarpic shoots. Submitted to Bulletin of Mathematical Biology, 2020). Использование файлов *.csv более естествнно для практикующих ботаников, но приводит к высокому уровню избыточности при увеличении детализации данных; файлы *.orn лишены этого недостатка.

3 Основные команды

3.1 Команды ORNAMENTAL

Любая команда приложения ORNAMENTAL состоит из имени команды за которым следует список параметров со значениями. Параметры различаются по имени, порядок расположения таких параметров произволен. Разделение параметров и значений выполняется символом пробела.

Формат команды :

$$cmd\ [[param_1]\ value_1]\ ..\ [[param_N]\ value_N]$$

где *cmd* - полное или сокращенное имя команды, *param* - полное или сокращенное имя параметра, *value* - значение параметра. В квадратных скобках указаны необязательные имена параметров и их значения.

Команда может не иметь параметров или иметь безымянные параметры. Безымянные параметры всегда имеют фиксированный порядок следования. Имена команд и параметров могут использоваться в полном и сокращенном вариантах. Одинаковые сокращения параметров разных команд или одинаковые сокращения для команды и параметра не конфликтуют друг с другом.

3.2 Команды linux

В командной строке программы можно использовать как команды приложения ORNAMENTAL, так и команды linux. Если команда не опознана приложением как команда ORNAMENTAL, то она запускается как команда linux. Целью использования команд linux является получение информации об установленных в системе приложениях, при настройке ORNAMENTAL. Формат команд linux полностью соответствует тому формату, который используется в терминалах linux.

ВНИМАНИЕ! Неосторожное использование команд linux может привести к необратимому повреждению файлов пользователя, поскольку они не дают предупреждений. Также возможно кажущееся "зависание"работы приложения. В этом случае необходимо нажать Ctrl-C. Часть команд linux не может быть запущена в терминале ORNAMENTAL, это относится к тем командам, которые не могут корректно быть запущены командой system().

3.3 Список команд

В списке команд используются следующие символы:

- '|' - оператор «ИЛИ» позволяет выбрать тот или иной вариант;
- «[ ]» - в квадратные скобки заключаются необязательные параметры;
- « < > » - в угловых скобках пишутся изменяемые имена параметров и их значений;
- без скобок пишутся фиксированные значения.

В описании команд используется символ '!', который просит пользователя обратить внимание на последующее замечание.

3.3.1 help|h

Команда дает список всех используемых команд ORNAMENTAL. Параметров нет.

3.3.2 environment|env [<var> <value>]

Команда без параметров дает список всех переменных окружения. Команда с одним параметром позволяет менять значение заданной переменной окружения.

Используются следующие переменные окружения:

- LOCALE (сокр. L|l) - локализация
- WORK_DIR (сокр. W|w) рабочая директория (файлы пользователя)
- OUTPUT_DIR (сокр. O|o) директория для записи результатов работы приложения
- PDF_VIEW (сокр. PDF|pdf|P|p) приложение для просмотра файлов pdf
- SVG_TO_PDF (сокр. SVG|svg|S|s) приложение для конвертации файлов svg в pdf

Примеры использования.

- env LOCALE ru_RU-UTF-8
  # установка русской локали
- env LOCALE C
  # установка "C"локали (для опытных пользователей)
- env WORK_DIR
  # установка рабочей директории /home/<user>
- env WORK_DIR tmp/dir
  # установка рабочей директории /home/<user>/tmp/dir (! перед tmp/ нет слеша)
- env WORK_DIR /tmp/dir1
  # установка рабочей директории /tmp/dir (! перед tmp/ есть слеш)
- env OUTPUT_DIR tmp/dir1
  # установка директории /home/<user>/tmp/dir для вывода данных
- env PDF_VIEW evince %s.pdf
  # использование evince для просмотра графиков (! постфикс .pdf после %s)
- env SVG_TO_PDF inkscape –export-pdf=%s.pdf %s.svg
  # использование inkscape для конвертации схем побегов из *.svg в *.pdf формат (! порядок токенов %s.pdf %s.svg строго фиксирован)

Внимание! Возможны проблемы с некоторыми локалями. Если после установки локали начались искажения при отрисовке графиков, попробуйте установить C локаль.

3.3.3 window|w

Открыть графический терминал для просмотра модели: структуры побега и графики динамики его развития.

3.3.4 fload|f< <filename> [<fileID>]

Команда загружает файл данных форматов *.csv или *.orn для обработки. После загрузки набор данных получает имя <fileID>, если оно задано. Имя <fileID> может быть любым строковым/цифровым идентификатором. Именно по этому идентификатору осуществляется доступ к данному набору данных. Если <fileID> не задано, то набору данных присваивается имя *noname*. Загруженному набору данных присваиваются статус R (для чтения), а если загруженный файл единственный, то и статус W (для записи). Загрузить можно любое число файлов или один и тот же файл многократно. Все загруженные наборы данных должны иметь разные <fileID>. При дублировании <fileID> ранее загруженные наборы данных исчезают.

3.3.5 fadd|f+ <fileID>

Команда создает новый пустой набор данных с идентификатором <fileID> и устанавливает ему статус W (для записи). Пустой набор данных может использоваться для накопления сгенерированных по моделям образцов побегов (см. команду model).

3.3.6 files|f

Команда дает список всех загруженных файлов с их статусами.

3.3.7 files|f <fileID> R|W|RW

Команда позволяет менять статус указанного файла.

Создано или загружено может быть любое число наборов данных, но статусом R может обладать только один набор данных. Именно этот набо данных используется для чтения данных и генерации моделей. Статусом W также может обладать только один набор данных, именно в него записываются сгенерированные по моделям образцы побегов. Все остальные наборы данных имеют неопределенный статус. По сути, статусы показывают какие наборы данных активны. Установка статуса для некоторого набора данных автоматически отнимает этот статус у другого набора. Статусы наборов данных не фиксируются жестко, они оперативно меняются данной командой в соответствии с запросами пользователя.

Примечание. Есть еще статус '∧' который означает, что над данными файла выполнена процедура уточнения данных по методу MLM (команда festimate).

3.3.8 fsave|f> <fileID> <filename>

Сохранить набор данных <fileID> в файл <filename>. Файл помещается в OUTPUT_DIR. (!) Путь к файлу прописывать не нужно.

3.3.9 fdel|f- <fileID>

Уничтожает набор данных <fileID>. (!) Никаких предупреждений не дается.

3.3.10 festimate|f∧ MLM|random

Уточнить данные наблюдений. В силу высокой трудоемкости, наблюдения за развитием побегов могут выполняться не ежедневно, то есть иметь случайные или фиксированные (наблюдения раз в два/три дня) пропуски. Как результат возникает неопределеность в определении смены фаз развития цветков побега. Данная команда позволяет дать улучшить имеющиеся данные. При уточнении любым методом учитывается связность разных фаз развития (например, увеличение длительности фазы уменьшает продолжительность предыдущей/последующей фаз развития).

Команда требует указать метод уточнения данных. Предлагаются два способа:

- MLM - уточнение данных методом максимального правдоподобия. Предполагается, что данные рапределены по нормальному закону.
- random - случайный выбор значения в известном интервале неопределенности.

После уточнения данных методом *MLM* у файла появляется статус '∧', после использования метода *random* статус '∧' снимается.

3.3.11 madd|m+ <fileID:sproutID> [<modelID>]

Генерация модели побега с указанным <modelID> из файла <fileID>. Для генерации в качестве образца используется побег с именем <sproutID>. После генерации модель получает имя <modelID> (или noname, в случае отсутствия <modelID>). После генерации модели ей присваеивается статус 'A' (см. команду models <modelID>).

Под моделью понимается такая совокупность регрессионных соотношений между различными (структурными/динамическими) параметрами побега, которая позволяет полностью восстановить структуру и динамику развития побега в отсутствии исходных данных со статистической точностью, то есть решить обратную задачу (от модели к данным). При генерации создается не только модель (минимально необходимая совокупность соотношений), но и ряд дополнительных соотношений для более полного просмотра модели.

При создании модели используются данные по развитию для побега <sproutID> (модель индивидуального побега). Дополнительно собираются структурные данные всех побегов файла <fileID> и строятся соответствующие численные распределения для возможности генерации модельных побегов с различной структурой.

3.3.12 mdel|m- <modelID>

Удалить указанную модель.

3.3.13 models|m

Посмотреть полный список всех моделей.

3.3.14 models|m <modelID>

Установить статус 'A' для указанной модели. Данный статус означает, что генерация всех графиков будет выполняться для данной модели.

3.3.15 plots|p

Показать список всех графиков (основных, т.е. использующихся для генерации модельных побегов и дополнительных, неучаствующих в генерации). Каждому графику присваивается идентификатор <plotID>, по которому можно его просмотреть, уничтожить или перегенерить с другим параметром сглаживания.

3.3.16 plots|p <plotID>

Показать диаграмму с идентификатором <plotID> для построенной модели. (!) Все кривые генерятся с минимально возможным окном сглаживания, ширина которого определяется числом точек в данных.

3.3.17 padd|p+ <plotID> [w <value>]

ДОБАВИТЬ на диаграмму график с заданой относительной шириной окна сглаживания данных. (! ширина окна сглаживания является относительной; ! невозможно задать очень малую ширину окна сглаживания, она всегда ограничена числом данных). Данная команда является вспомогательной и позволяет "поиграться"с разными вариантами генерации графиков. Использование команды без параметра width генерит график с шириной окна сглаживания по умалчиванию. Для очистки диаграммы используется команда pdel (см. далее).

3.3.18 pdel|p- [<plotID>]

Удалить диаграмму с заданным идентификатором, без указания <plotID> удаляется активная, то есть последняя отрисованная диаграмма. (! После удаления диаграммы модель может стать неполной и не позволять генерацию модельных побегов. Поэтому необходимо восстановить удаленное соотношение командой plot <plotID>).

3.3.19 psave|p> <filename>

Записать текущую диаграмму в OUTPUT_DIR с заданным именем. (! Имя указывается без пути).

3.3.20 test|t <plotID> [n]

Выполнить статистические тесты для заданного графика <plotID>.

- Если график является распределением, то выполняется тестирование на нормальность распределения и печатается некоторая статистика. При тестировании используется заданное число точек n. Если параметр n не задан, то число точек n равно числу точек в исходных данных.

- Если график является регрессией, то выводится коэффициент корреляции Пирсона r и его p-value, коэффициент детерминации $R2$ и выполняется тестирование неописываемых регрессией остатков на наличие в них корреляций и тестирование значений остатков на нормальность. В данном варианте параметр [n] игнорируется и для тестирования используются исходные данные.

3.3.21 scheme <modelID>|<fileID:sproutID> <date|day>

Отобразить структуру побега для модели <modelID> или побега sproutID в файле fileID и состояние развития его цветков на заданную дату (для экспериментальных данных) либо на относительную дату от начала цветения. Дата дается в формате DAY-MONTH, например 4-Jul. Относительная дата задается в виде целого числа.

Примеры использования.

- scheme <modelID> 4-Jul
  # отобразить модель <modelID> на 4 июля.
- scheme <modelID> 0
  # отобразить побег на дату начала цветения
- scheme <modelID> -2
  # отобразить побег на дату за два дня до начала цветения

3.3.22 video <modelID>|<fileID:sproutID> <date1:date2|day1:day2>

Отобразить последовательность развития цветков побега для модели <modelID> или побега sproutID в файле fileID в заданном интервале дат или дней от начала цветения. Даты задаются в формате DAY-MONTH, относительные даты задается в виде целых чисел.

3.3.23 generate|g <modelID> N [<n1>:<n2>]

Сгенерить N модельных побегов, используя модель <modelID>. Результат записывается в файл данных, имеющим статус W. Параметры <n1> (число цветков 1-го порядка) и <n2>(число цветков второго порядка) задают ограничения на структуру генерируемых побегов, генерится побег с числом цветков соответствующих порядков максимально близким к указанным значениям. Указание нулевых значений означает, что числа должны быть взяты такими же, как в побеге, который использовался для создания модели. Указание значений в виде *:* означает отсутствие любых ограничений. В этом случае используются значения, которые генерятся из соответствующих распределений и регрессий.

3.3.24 quit|exit

Завершить выполнение программы.

4 Сценарии работы

Приложение запускается командой: ornamental [<filename>], где необязательный параметр <filename> содержит имя файла сценария.

При запуске приложение проверяет наличие в командной строке параметра <filename>. При его обнаружении, приложение сперва выполняет все команды, записанные в сценарий, а затем переходит в интерактивный режим работы. Если в сценарии есть команда *exit/quit*, то программа завершается без перехода в интерактивный режим.

Файлы сценариев могут быть созданы в любом текстовом редакторе, либо путем редактирования *.usr файла, в который записываются все команды, которые ввел пользователь в процессе выполнения. *.usr файл лежит в директории /home/<user>/.ornamental, которая создается автоматически при запуске приложения. Также в этой директории лежат *.log файлы, в которые записывается полученные команды и их результаты.

4.1 Создание сценария настройки

Сценарий позволяет избежать настройки приложения при каждом запуске. При создании файла вручную в него нужно вписать последовательность команд настройки.

Пример файла настройки:

```
env LOCALE ru_RU-UTF-8 \# set locale
env PDF_VIEW evince %s.pdf \# pdf viewer
env SVG_TO_PDF inkscape --export-pdf=%s.pdf %s.svg
\# svg->pdf convertion
env WORK_DIR ~ \# set workdir = /home/<user>
env OUTPUT_DIR ~/tmp/orn \# set outputdir = ~/tmp/orn
```

Полезные советы:

1. При работе ORNAMENTAL генерится и модифицируется множество временных pdf файлов. Для их просмотра используйте приложения, которые автоматически перезагружают файлы при их изменении на диске, такие как evince и okular. Не используйте приложения такие как xpdf, qpdfview, firefox, loffice и ряд других, поскольку в них перезагрузку файлов необходимо выполнять вручную.

2. ! При установке переменной окружения PDF_VIEW не забывайте указать регулярное выражение '%s.pdf' для возможности подстановки имени файла (постфикс .pdf обязателен).

3. Команда 'apropos viewer' позволяет узнать о наличии программ просмотрщиков в вашей системе (см. https://www.geeksforgeeks.org/apropos-command-in-linux-with-examples/). Она может быть запущена в терминале ORNAMENTAL. Команда "dpkg –list | grep -i '[(]pdf[ )].*viewer'"также может быть использована для поиска.

4. При работе ORNAMENTAL формируются файлы схем соцветий в svg формате. Для их просмотра в том же самом графическом pdf терминале файлы схем соцветий конвертируются в pdf формат. Для конвер-

тации укажите переменную окружения SVG_TO_PDF. (!) При установке переменной окружения SVG_TO_PDF не забывайте указать регулярные выражения '%s.pdf' и '%s.svg' для возможности подстановки имен файлов (их порядок следования фиксирован, сперва .pdf, а затем .svg).

4.2 Создание сценария настройки из приложения

Запустите приложение и выполните команды (например, пункта **??**) в нем. Завершите приложение. Найдите файл /home/<user>/.ornamental/terminal.usr, отредактируйте его и сохраните под другим именем. Для обеспечения перехода в интерактивный режим работы после завершения сценария, удалите из файла команду *exit/quit*.

Редактирование файлов *.usr самый простой и надежный способ создания работающих сценариев, поскольку приложение реагирует на ошибочно введенные команды.

4.3 Стандартный сценарий работы приложения. Пример

В примере используются сокращенные имена команд, что удобнее при работе.

```
# setting environment variables and opening a graphical terminal
env LOCALE ru\_RU-UTF-8
env PDF\_VIEW evince \%s.pdf
env SVG\_TO\_PDF inkscape --export-pdf=\%s.pdf \%s.svg
env WORK\_DIR ~
env OUTPUT\_DIR ~/tmp/orn
window # open window
h # help
#
# loading and refining data, building a model of the 3rd sprout
f+ C.sarmatica.csv S
f^ MLM S
m+ S:3 m3
#
# viewing plot IDs, drawing the plot 9 and saving it to disk
p
p 9
p> flowering\_curves1.pdf
#
# creating empty file, generaing 100 model sprouts
# and inserting their to 'S'(C.sarmatica)
f+ model_sprouts
generate m3 100
#
# viewing the file for the numbers of the newly inserted sprouts,
# and selecting a sprout (for example, 101);
# building a model based on this new model sprout
```

```
# drawing the plot 9 and saving it to disk
f
m+ S:101 m101
p 9
p> flowering\_curves1.pdf

# viewing the structure of the models on the day
# of flowering start and saving them
scheme m1 0
p> scheme1.pdf
scheme m101 0
p> scheme101.pdf

exit
```

5 Замеченные недостатки

- Вызываемые сторонние программы могут пытатся сделать вывод предупреждений в терминал (gnuplot). Как результат это приводит к мерцанию экрана.
- В некоторых локалях (например, в) возможны артефакты при построении графиков и схем соцветий. Без артефактов работают локали C, en_US.UTF-8, ru_RU-UTF-8 (возможно все UTF-8).
- ошибочные исходные данные могут привидить к артефактам на графиках. Например, графики, построенные для исходных данных файла C.saramtica.csv (побег 2) имеют артефакты, связанные с наличием пропущенных данных для цветков от 1 до 5. Дефект исправляется после выполнения команды festimate, и перестройки модели m+ C.sarmatica:2.