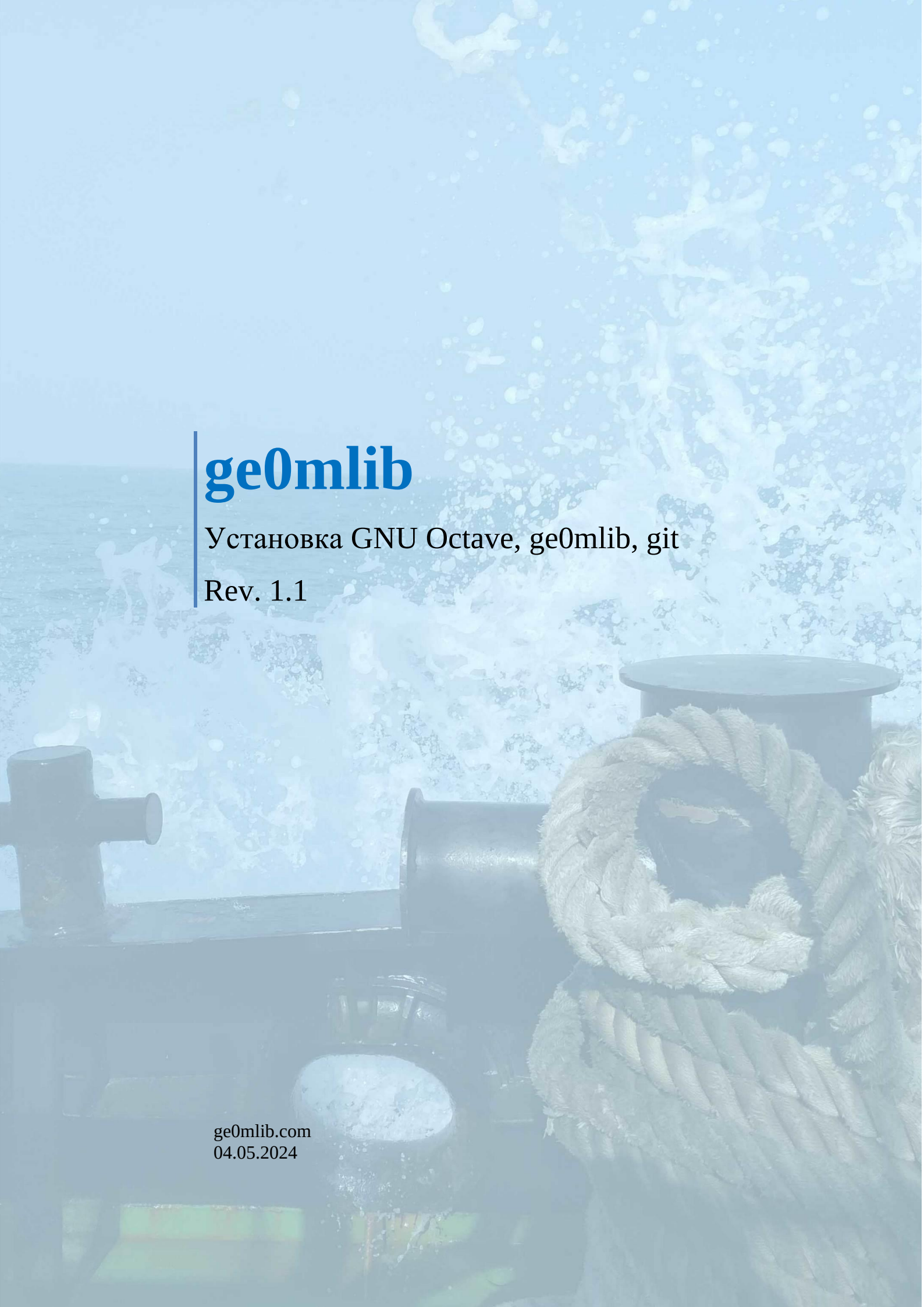




ge0mlib

Установка GNU Octave, ge0mlib, git

Rev. 1.1

ge0mlib.com
04.05.2024

Содержание

Введение.....	4
1 GNU Octave.....	5
1.1 Установка GNU Octave.....	5
1.2 Установка дополнительных пакетов (библиотек)	6
2 Установка ge0mlib (MatLab, GNU Octave).....	9
3 Git.....	12
3.1 Цели использования Git и Git Hub	12
3.2 Установка Git	12
3.3 Создание git-проекта и скачивание ge0mlib из репозитория	15
3.4 Перемещение по истории в git.....	16
3.5 Запуск графического интерфейса.....	18
4 Структура скрипта и команд скрипта	19
Заключение	22

История изменений

Версия	Дата	Описание изменений
1.0	2023-12-26	Документ создан // Иван Д.
1.1	2024-04-05	Текст отредактирован // Иван Д., Маргарита Б.

Список рисунков

Figure 1.1 Установка GNU Octave.....	5
Figure 1.2 Интерфейс командного окна GNU Octave.....	5
Figure 1.3 Интерфейс редактора m-файлов GNU Octave	6
Figure 1.4 Содержимое папок с библиотеками для GNU Octave	8
Figure 1.5 Меню прописывания путей к библиотекам в настройках GNU Octave.....	9
Figure 1.6 Пример интерфейса для скачивания библиотеки из Internet на сайте GNU Octave	9
Figure 2.1 Структура каталога библиотеки ge0mliб	10
Figure 2.2 Окно MatLab с прописанными путями для библиотеки ge0mliб	11
Figure 2.3 Окно GNU Octave с прописанными путями для библиотеки ge0mliб	11
Figure 3.1 Установка Git.....	14
Figure 3.2 Выполнение команд для начальных установок git	15
Figure 3.3 Открытие Git Bash из контекстного меню Windows	15
Figure 3.4 Окно git и содержимое папки Functions после создания репозитория	16
Figure 3.5 Окно git со списком commit	16
Figure 3.6 Файлы в папке Functions в результате применения команды “checkout”	17
Figure 3.7 Ошибка git, говорящая о необходимости сохранить последние изменения как commit .17	
Figure 3.8 Открытие Git GUI через контекстное меню Windows.....	18
Figure 3.9 Интерфейс Git GUI	18
Figure 4.1 Пример кода скрипта MagToolR01	20
Figure 4.2 Окно MatLab с выполненной последовательностью команд для MagToolR01	21

Введение

Библиотека ge0mlib предназначена для регистрации и обработки данных морской инженерной геофизики. Функции обработки реализованы в MatLab 2018b (<https://mathworks.com/>) и адаптированы под GNU Octave 8.4.0 (<https://www.octave.org/>; свободно-распространяемое программное обеспечение). Сайт проекта: <https://ge0mlib.com/>. Библиотека выложена на Git Hub: <https://github.com/ge0mlib/ge0mlib>.

Данный документ написан для пользователя Windows, не являющегося программистом, которому необходимо выполнить обработку данных при помощи готового скрипта, написанного для MatLab или Octave. Для выполнения задачи имеется:

- 1) m-файл со скриптом для обработки данных, написанный ранее для языка программирования MatLab и/или GNU Octave (дату разработки можно посмотреть в конце текста скрипта);
- 2) прописанный в комментариях в начале текста скрипта, список дополнительных библиотек MatLab и/или GNU Octave, которые надо установить для запуска скрипта;
- 3) описание команд скрипта (при этом, описание каждой отдельной команды должно быть приведено в тексте скрипта, там же должны быть приведены примеры рабочих последовательностей команд) или устные инструкции вида «обрабатывали этим скриптом» (что встречается даже чаще).

Цель данного документа – обеспечить быструю установку программного обеспечения и переход к обработке данных. Документ описывает следующие действия:

- скачивание и установку GNU Octave (версия для Windows);
- скачивание и установку дополнительных пакетов (библиотек) для GNU Octave (Windows);
- скачивание и установку библиотеки ge0mlib;
- скачивание и установку Git (версия для Windows);
- восстановление «снимка» библиотеки ge0mlib на дату создания скрипта при помощи Git (имеется ввиду “commit”, как аналог копии папки с файлами, содержащими функции библиотеки на заданный момент времени);
- работу с командами скрипта (описание принципов построения команд скрипта и информации, содержащейся в тексте скрипта).

Текст документа включает чуть больше информации, чем необходимо. Однако эта информация позволяет чувствовать себя более комфортно при работе с GNU Octave, ge0mlib и Git. Такой необязательный (но это не точно) к прочтению текст подсвечен синим цветом.

1 GNU Octave

1.1 Установка GNU Octave

Дистрибутив Octave скачивается с сайта <https://octave.org/download>. На момент написания текста это файл octave-8.4.0-w64-installer.exe (для Win64).

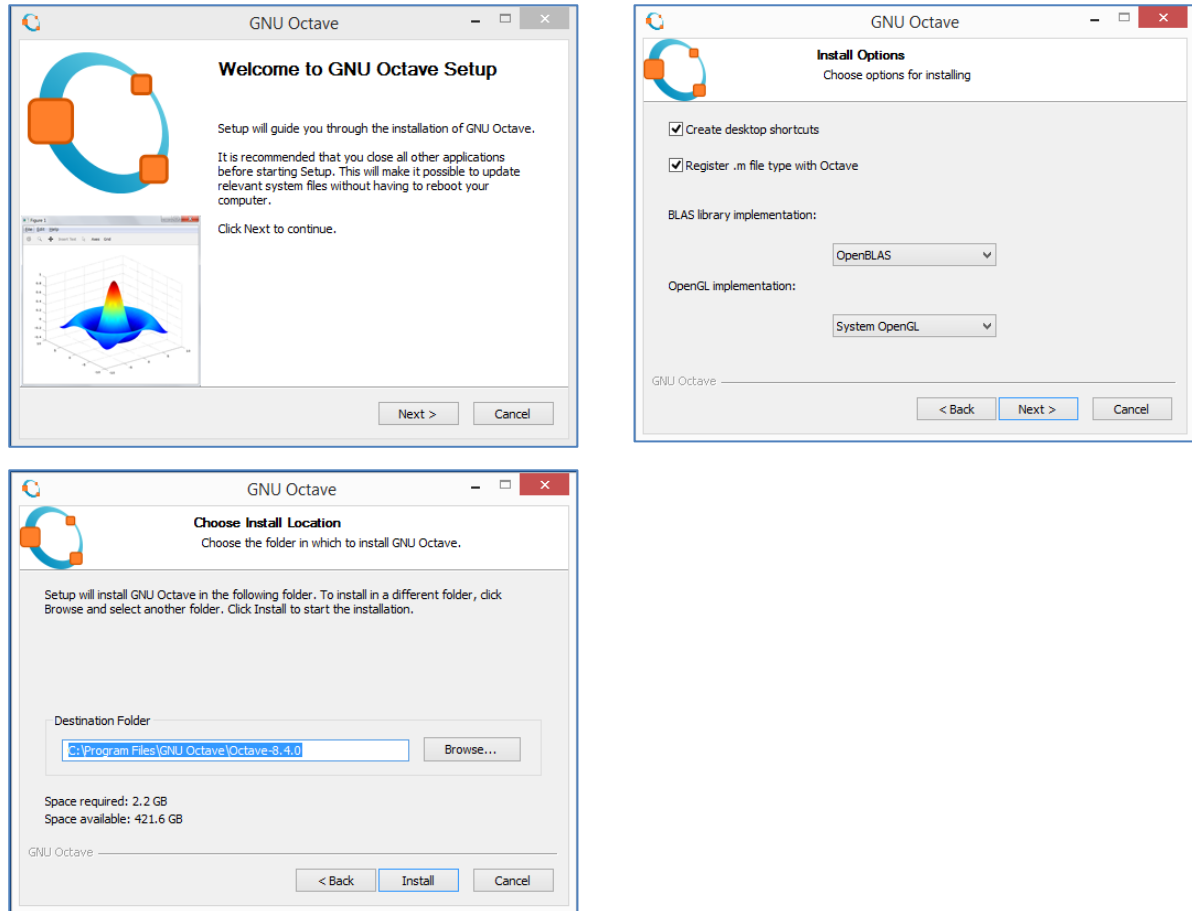


Figure 1.1 Установка GNU Octave

Командное окно Octave после установки, с перемещенными для удобства вкладками показано на Figure 1.2. Обратите внимание на вкладки Command History и Workspace.

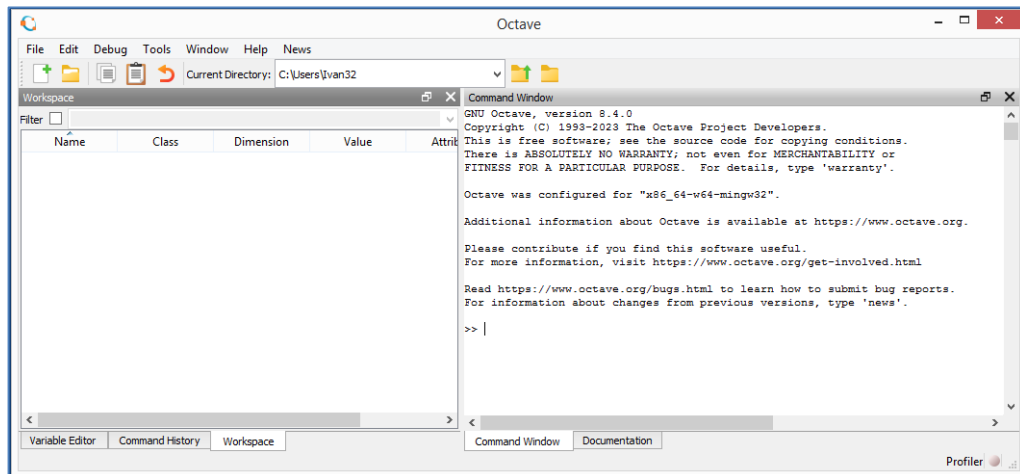


Figure 1.2 Интерфейс командного окна GNU Octave

Окно отладчика m-файлов показано на *Figure 1.3*. Может быть удобно ввести последовательность команд скрипта в окне отладчика, потом сохранить этот файл в папке Script (описание папки приводится ниже в разделе 2, при описании установки ge0mlib). После этого вместо ввода в командное окно «последовательности команд скрипта» можно будет вводить только имя файла, сформированного в отладчике (сам этот файл, по сути, будет являться простейшим скриптом).

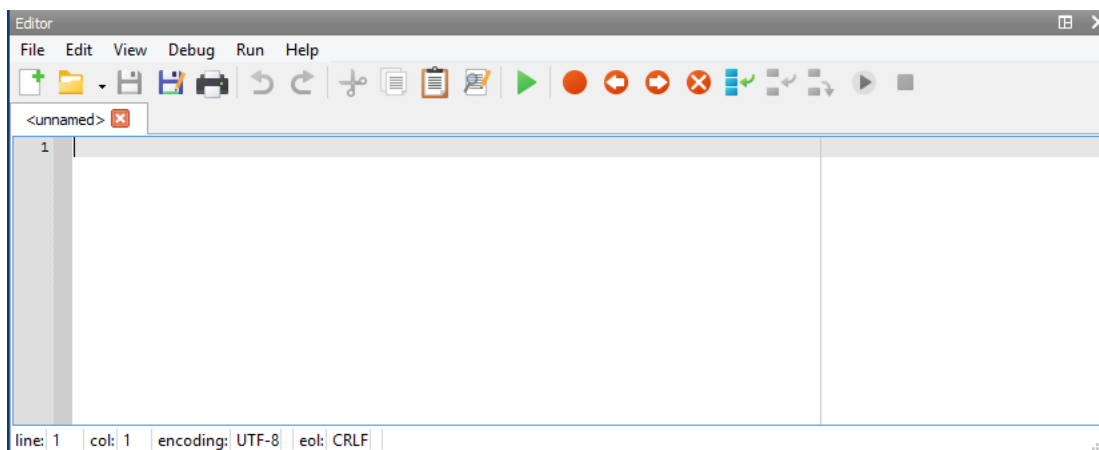


Figure 1.3 Интерфейс редактора m-файлов GNU Octave

1.2 Установка дополнительных пакетов (библиотек)

Дополнительные библиотеки в Octave принято называть «пакетами», а в MatLab – “toolbox”. Ссылка на страницу с новой версией официальных пакетов для Octave: <https://gnu-octave.github.io/packages/>.

Ссылка на страницу со старыми версиями пакетов: <https://octave.sourceforge.io/packages.php>

Ниже приведен список пакетов, которые могут быть использованы для решения задач морской геофизики. Выборка является субъективной и предназначена в первую очередь для того, чтобы дать общее представление о пакетах, входящих в состав GNU Octave.

control – Computer-Aided Control System Design (CACSD) Tools for GNU Octave, based on the SLICOT Library.

data-smoothing – Algorithms for smoothing noisy data.

divand – Performs an n-dimensional variational analysis (interpolation) of arbitrarily located observations.

fda – Functional Data Analysis.

fileio – I/O function for files holding structured data, such as JSON and XML files.

fuzzy-logic-toolkit – A mostly MATLAB-compatible fuzzy logic toolkit for Octave.

geographiclib – Native Octave/MATLAB implementations of a subset of the C++ library, GeographicLib.

Key components of this toolbox are: (a) Geodesics, direct, inverse, area calculations; (b) Projections, transverse Mercator, polar stereographic, etc; (c) Grid systems, UTM, UPS, MGRS; (d) Geoid lookup, egm84, egm96, egm2008 geoids supported; (e) Geometric transformations, geocentric, local cartesian; (f) Great ellipse, direct, inverse, area calculations.

geometry – Library for extending MatGeom functionality.

image – Functions for image processing, feature extraction, image statistics, spatial and geometric transformations, morphological operations, linear filtering, and much more.

io – Input/Output in external formats.

linear-algebra – Additional linear algebra code, including matrix functions.

mapping – Simple mapping and GIS .shp .dxf and raster file functions.

matgeom – Geometry toolbox for 2D/3D geometric computing.

miscellaneous – Miscellaneous tools that don't fit somewhere else.

mvn – Multivariate normal distribution clustering and utility functions.

nan – A statistics and machine learning toolbox for data with and w/o missing values.

octproj – This package allows to call functions of PROJ library for cartographic projections and CRS transformations.

optim – Non-linear optimization toolkit.

optiminterp – An optimal interpolation toolbox providing functions to perform a n-dimensional optimal interpolations of arbitrarily distributed data points.

packajoozle – Enhanced package manager for GNU Octave.

pkg-octave-doc – This package provides functions for generating HTML pages that contain the help texts of the functions of an octave package. The package is designed to work with installed packages and use their INDEX file for creating the respective functions' HTML pages. The default layout is based on bootstrap 5 and it follows the design of the Octave Packages GitHub page.

quaternion – Quaternion package for GNU Octave, includes a quaternion class with overloaded operators.

signal – Signal processing tools, including filtering, windowing and display functions.

sparsersb – Interface to the librsb package implementing the RSB sparse matrix format for fast shared-memory sparse matrix computations.

splines – Additional spline functions

sqlite – Basic Octave implementation of the sqlite toolkit

statistics – The Statistics package for GNU Octave.

statistics-resampling – Estimate bias, uncertainty (standard errors and confidence intervals) and test hypotheses (p-values) using resampling methods. (Note that versions of this package <= 5.4.3 are named the statistics-bootstrap package).

stk – The STK is a (not so) Small Toolbox for Kriging. Its primary focus is on the interpolation/regression technique known as kriging, which is very closely related to Splines and Radial Basis Functions and can be interpreted as a non-parametric Bayesian method using a Gaussian Process (GP) prior. The STK also provides tools for the sequential and non-sequential design of experiments. Even though it is, currently, mostly geared towards the Design and Analysis of Computer Experiments (DACE), the STK can be useful for other applications areas (such as Geostatistics, Machine Learning, Non-parametric Regression, etc.).

strings – Additional functions for manipulation and analysis of strings.

struct – Additional structure manipulation functions.

Чтобы в Octave можно было использовать функции пакета, путь к папке с функциями должен быть прописан в настройках Octave. В корневой папке Octave присутствуют две папки с пакетами (*Figure 1.4*):

- 1) c:\Program Files\GNU Octave\Octave-8.4.0\mingw64\share\octave\8.4.0\m\ – папка содержит «интегрированные в систему» пакеты, такие как signal и statistics;
- 2) c:\Program Files\GNU Octave\Octave-8.4.0\mingw64\share\octave\packages\ – папка содержит не интегрированные пакеты, пути для которых изначально в настройках Octave не прописаны.

С большой долей вероятности требуемый пакет находится в этих папках. Тогда скачивать его из Internet не понадобится.

Name	Ext	Size	Date	Attr
[.]	<DIR>		24.12.2023 21:55	----
[@ftp]	<DIR>		24.12.2023 21:55	----
[+containers]	<DIR>		24.12.2023 21:55	----
[+matlab]	<DIR>		24.12.2023 21:55	----
[audio]	<DIR>		24.12.2023 21:55	----
[deprecated]	<DIR>		24.12.2023 21:55	----
[elfun]	<DIR>		24.12.2023 21:55	----
[general]	<DIR>		24.12.2023 21:55	----
[geometry]	<DIR>		24.12.2023 21:55	----
[gui]	<DIR>		24.12.2023 21:55	----
[help]	<DIR>		24.12.2023 21:55	----
[image]	<DIR>		24.12.2023 21:55	----
[io]	<DIR>		24.12.2023 21:55	----
[java]	<DIR>		24.12.2023 21:55	----
[legacy]	<DIR>		24.12.2023 21:55	----
[linear-algebra]	<DIR>		24.12.2023 21:55	----
[miscellaneous]	<DIR>		24.12.2023 21:55	----
[ode]	<DIR>		24.12.2023 21:55	----
[optimization]	<DIR>		24.12.2023 21:55	----
[path]	<DIR>		24.12.2023 21:55	----
[pkg]	<DIR>		24.12.2023 21:55	----
[plot]	<DIR>		24.12.2023 21:55	----
[polynomial]	<DIR>		24.12.2023 21:55	----
[prefs]	<DIR>		24.12.2023 21:55	----
[profiler]	<DIR>		24.12.2023 21:55	----
[set]	<DIR>		24.12.2023 21:55	----
[signal]	<DIR>		24.12.2023 21:55	----
[sparse]	<DIR>		24.12.2023 21:55	----
[specfun]	<DIR>		24.12.2023 21:55	----
[special-matrix]	<DIR>		24.12.2023 21:55	----
[startup]	<DIR>		24.12.2023 21:55	----
[statistics]	<DIR>		24.12.2023 21:55	----
[strings]	<DIR>		24.12.2023 21:55	----
[testfun]	<DIR>		24.12.2023 21:55	----
[time]	<DIR>		24.12.2023 21:55	----
[web]	<DIR>		24.12.2023 21:55	----

Name	Ext	Size	Date	Attr
[.]	<DIR>		24.12.2023 21:55	----
[audio-2.0.8]	<DIR>		24.12.2023 21:54	----
[biosig-2.5.2]	<DIR>		24.12.2023 21:55	----
[cfitio-0.0.5]	<DIR>		24.12.2023 21:54	----
[communications-1.2.6]	<DIR>		24.12.2023 21:55	----
[control-3.6.1]	<DIR>		24.12.2023 21:54	----
[database-2.4.4]	<DIR>		24.12.2023 21:54	----
[dataframe-1.2.0]	<DIR>		24.12.2023 21:55	----
[data-smoothing-1.3.0]	<DIR>		24.12.2023 21:55	----
[dicom-0.5.1]	<DIR>		24.12.2023 21:55	----
[financial-0.5.3]	<DIR>		24.12.2023 21:55	----
[fuzzy-logic-toolkit-0.4.6]	<DIR>		24.12.2023 21:54	----
[ga-0.10.3]	<DIR>		24.12.2023 21:55	----
[general-2.1.3]	<DIR>		24.12.2023 21:54	----
[generate_html-0.3.3]	<DIR>		24.12.2023 21:53	----
[geometry-4.0.0]	<DIR>		24.12.2023 21:53	----
[gsl-2.1.1]	<DIR>		24.12.2023 21:54	----
[image-2.14.0]	<DIR>		24.12.2023 21:53	----
[instrument-control-0.9.1]	<DIR>		24.12.2023 21:53	----
[interval-3.2.1]	<DIR>		24.12.2023 21:54	----
[io-2.6.4]	<DIR>		24.12.2023 21:54	----
[linear-algebra-2.2.3]	<DIR>		24.12.2023 21:55	----
[lssa-0.1.4]	<DIR>		24.12.2023 21:54	----
[lftfat-2.3.1]	<DIR>		24.12.2023 21:55	----
[mapping-1.4.2]	<DIR>		24.12.2023 21:54	----
[matgeom-1.2.3]	<DIR>		24.12.2023 21:55	----
[miscellaneous-1.3.0]	<DIR>		24.12.2023 21:54	----
[mqtt-0.0.4]	<DIR>		24.12.2023 21:54	----
[nan-3.7.0]	<DIR>		24.12.2023 21:55	----
[netcdf-1.0.17]	<DIR>		24.12.2023 21:54	----
[nurbs-1.4.3]	<DIR>		24.12.2023 21:54	----
[ocs-0.1.5]	<DIR>		24.12.2023 21:54	----
[octproj-3.0.2]	<DIR>		24.12.2023 21:54	----
[optim-1.6.2]	<DIR>		24.12.2023 21:54	----
[optiminterp-0.3.7]	<DIR>		24.12.2023 21:53	----
[quaternion-2.4.0]	<DIR>		24.12.2023 21:53	----
[queueing-1.2.7]	<DIR>		24.12.2023 21:54	----
[signal-1.4.5]	<DIR>		24.12.2023 21:54	----
[sockets-1.4.1]	<DIR>		24.12.2023 21:55	----
[sparsersb-1.0.9]	<DIR>		24.12.2023 21:54	----
[splines-1.3.5]	<DIR>		24.12.2023 21:54	----
[statistics-1.6.0]	<DIR>		24.12.2023 21:54	----
[stk-2.8.1]	<DIR>		24.12.2023 21:54	----
[strings-1.3.1]	<DIR>		24.12.2023 21:55	----
[struct-1.0.18]	<DIR>		24.12.2023 21:53	----
[symbolic-3.1.1]	<DIR>		24.12.2023 21:53	----
[tisean-0.2.3]	<DIR>		24.12.2023 21:55	----
[tsa-4.6.3]	<DIR>		24.12.2023 21:54	----
[video-2.1.1]	<DIR>		24.12.2023 21:53	----
[windows-1.6.4]	<DIR>		24.12.2023 21:55	----
[zeromq-1.5.6]	<DIR>		24.12.2023 21:54	----

Figure 1.4 Содержимое папок с библиотеками для GNU Octave

Посмотреть какие пути уже прописаны в настройках Octave можно при помощи Path Browser (прописаны пути ко всем пакетам в папке “m”). Используя кнопку “Add Folder” можно добавить пути к пакетам, после чего необходимо нажать на кнопку “Save”, чтобы сохранить список путей, и он автоматически загружался при последующих запусках Octave (*Figure 1.5*).

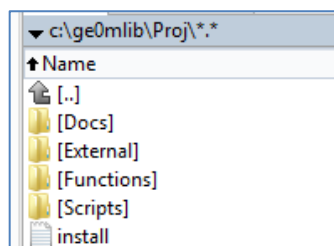


Figure 2.1 Структура каталога библиотеки ge0mlib

Назначение папок следующее:

- Docs – содержит документацию по работе с библиотекой (формат pdf), а также примеры скриптов (m-файлы и описание в формате pdf; видеоматериалы присутствуют на сайте, но в архив с библиотекой не включены);
- External – папка для дополнительных библиотек, которые используются или планируется использовать с ge0mlib. В настоящий момент это m_map, mksqlite-2.11-win64, n_vector;
- Functions – функции библиотеки ge0mlib (содержимое папки выложено на Git Hub: <https://github.com/ge0mlib/ge0mlib>);
- Scripts – пустая папка для скриптов, для локального использования. Сюда необходимо положить m-файл со скриптом, который требуется выполнить для обработки данных. Если вы позже напишете свой скрипт или функцию, то можно также разместить их в этой папке;
- install.txt – файл с командами для прописывания путей (вместо использования Path Browser);

Для установки ge0mlib можно прописать пути к m-файлам вручную с использованием Path Browser или прописать в первой строчке install.txt путь к корневой папке ge0mlib, после чего скопировать текст из файла в командное окно и нажать Enter (для выполнения введенных команд). Пример такой установки путей для корневой папки 'c:\ge0mlib\Proj\' показан на рисунках ниже. При выполнении команд будет проверено наличие соответствующих подпапок в External, а также папок Functions и Scripts. При наличии на диске соответствующих папок они будут прописаны в Path и сохранены (*Figure 2.2, Figure 2.3*). Если папки отсутствуют, то пути прописаны не будут (например, если заранее удалить папку External, то пути к удаленным дополнительным библиотекам, которые содержались в External, в файл Path не пропишутся).

Следует отметить, что библиотека ge0mlib исходно писалась для MatLab, а после этого функции были адаптированы под Octave. Цель такой адаптации – использование свободно-распространяемого программного обеспечения, которое можно установить на любой компьютер без покупки и манипуляций с лицензией (хорошими примерами таких программ являются QGIS и Python). При этом небольшая часть функций ge0mlib (использующих MatLab toolboxes) может не работать в Octave (в настоящий момент, при разработке функций, в первую очередь принимается во внимание совместимость с MatLab).

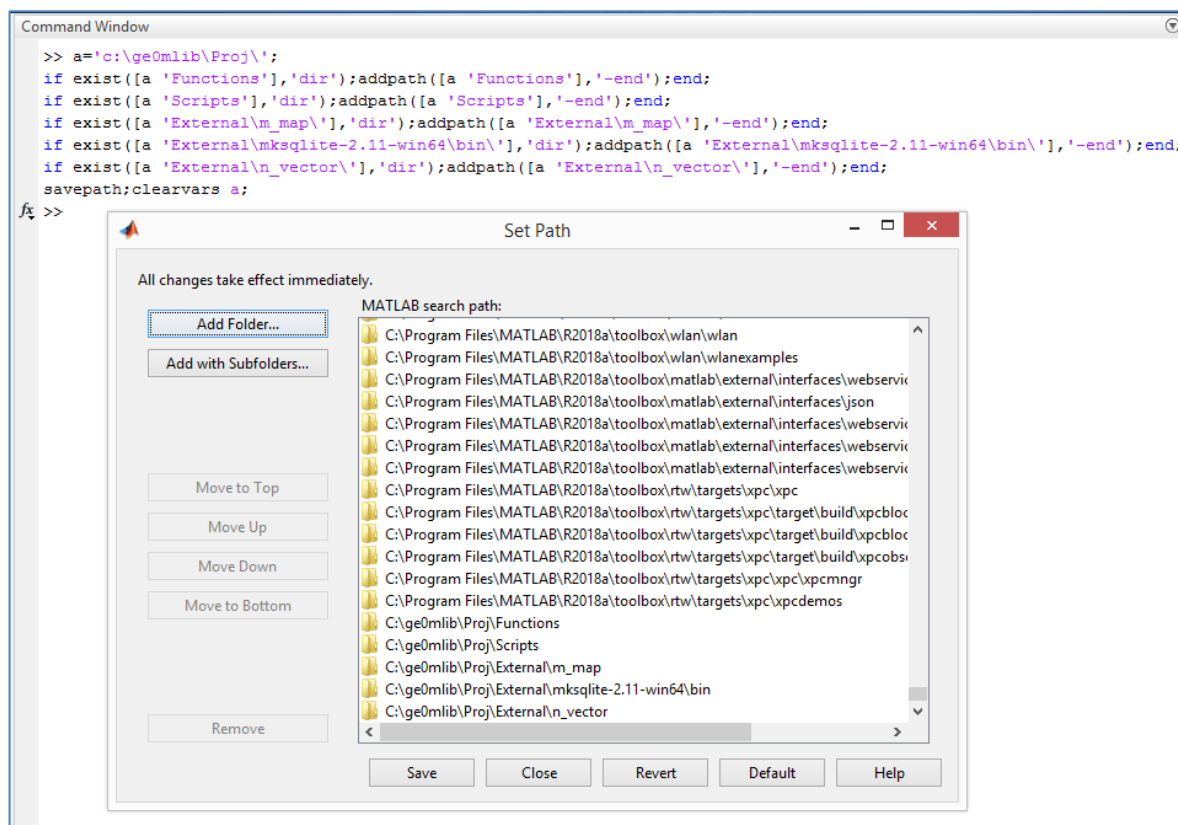


Figure 2.2 Окно MatLab с прописанными путями для библиотеки ge0mlib

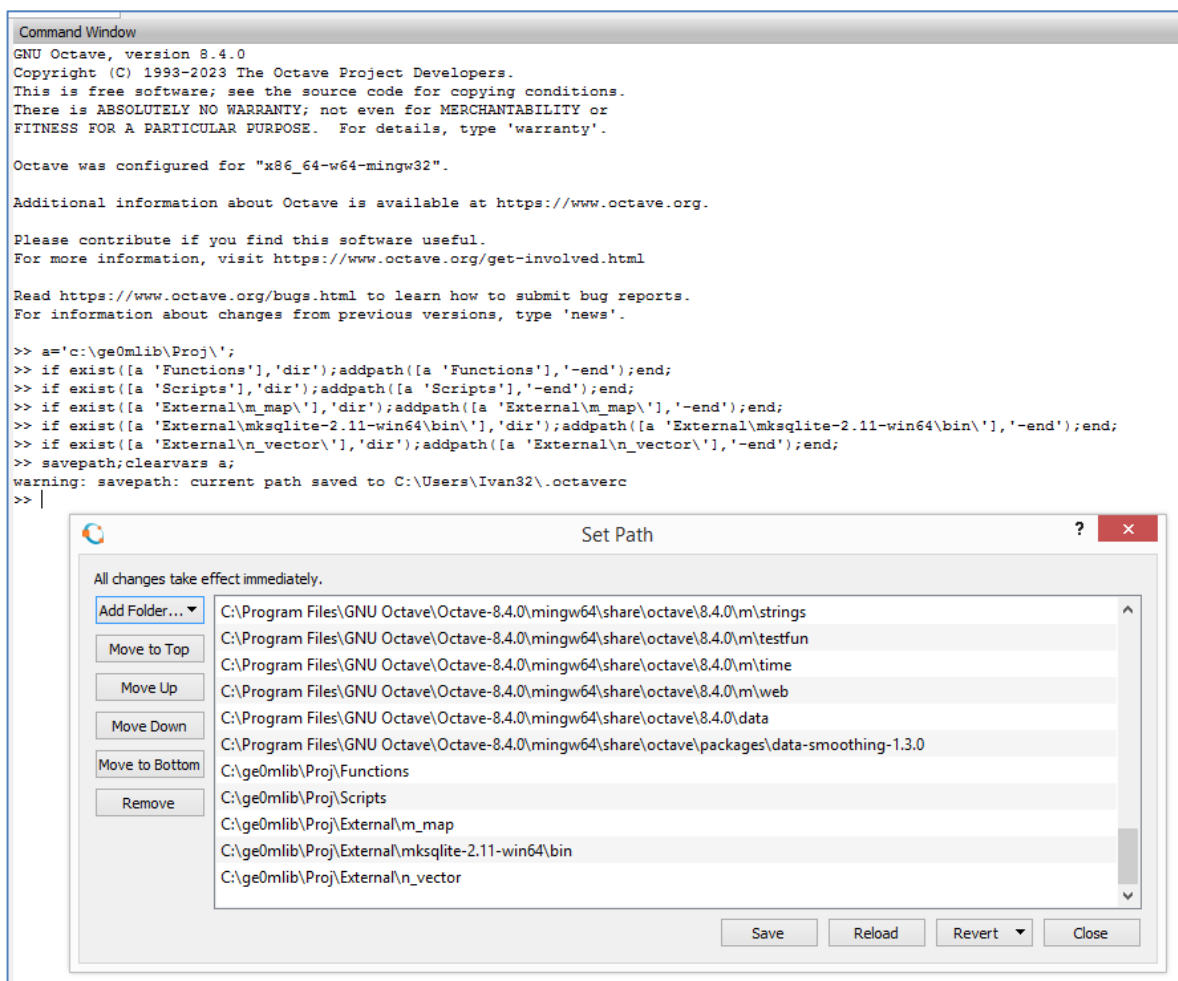


Figure 2.3 Окно GNU Octave с прописанными путями для библиотеки ge0mlib

3 Git

3.1 Цели использования Git и Git Hub

Для содержимого папки Functions (функций библиотеки ge0mlib) используется контроль версий Git. Библиотека ge0mlib выложена на Git Hub: <https://github.com/ge0mlib/ge0mlib>. Цель использования Git, рассматриваемая ниже, – восстановление «снимков» библиотеки, сделанных в прошлом и с вероятностью 90% эта информация вам не понадобится.

Поскольку функции в библиотеке постепенно изменяются, то «старый скрипт» может не запуститься с «новой версией библиотеки». Для запуска таких скриптов удобно иметь возможность выбрать «снимок» библиотеки на время написания скрипта. Для этого в коде скрипта, в последней строке, в обязательном порядке указывается дата его создания. Используя Git можно будет выбрать «снимок» (commit), предшествующий дате создания скрипта и запустить его без необходимости внесения исправлений в код. Это же касается примеров скриптов в папке Docs, написанных после 26.12.2023.

Описанное решение имеет свои недостатки – в старой версии библиотеки будут присутствовать все старые ошибки (что является несомненным минусом), однако там будут отсутствовать ошибки новые (что является некоторым плюсом). Мы предполагаем, что скрипт написанный и протестированный *в своё время* выдавал адекватные результаты расчетов, не подверженные ошибкам, поэтому может использоваться и в дальнейшем. Альтернативным решением является правка скрипта, с внесением изменений в синтаксис команд (предполагается, что он изменился) под новую версию библиотеки. Это безусловно лучшее решение, однако оно требует времени, знаний и тестирования работы измененного скрипта.

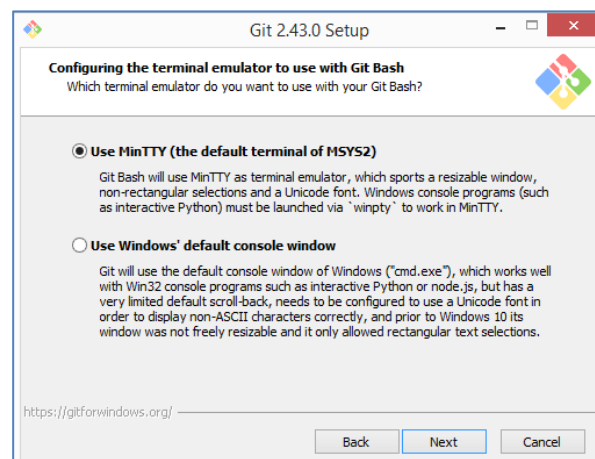
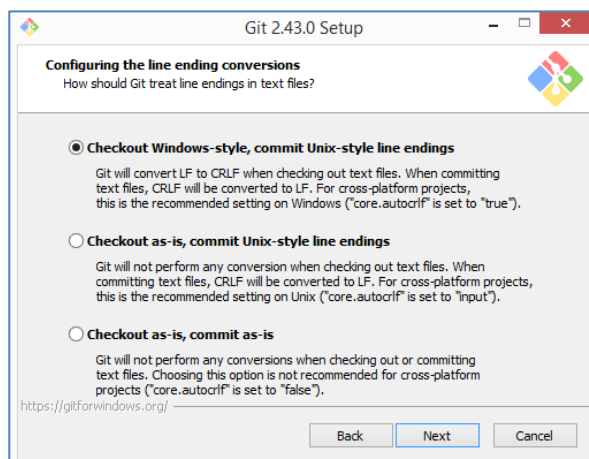
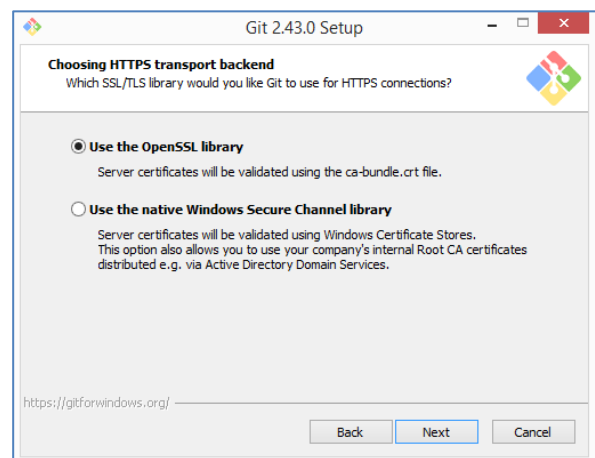
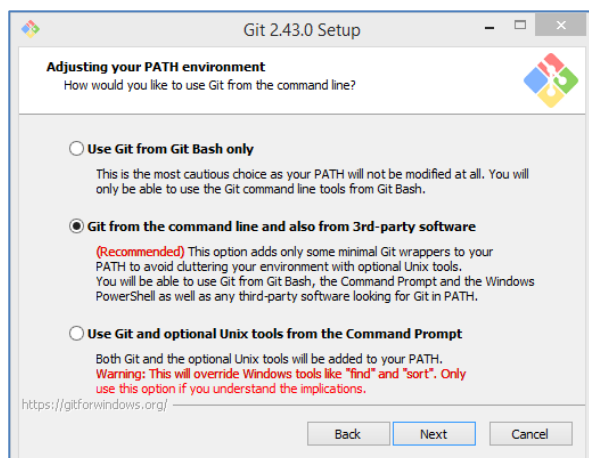
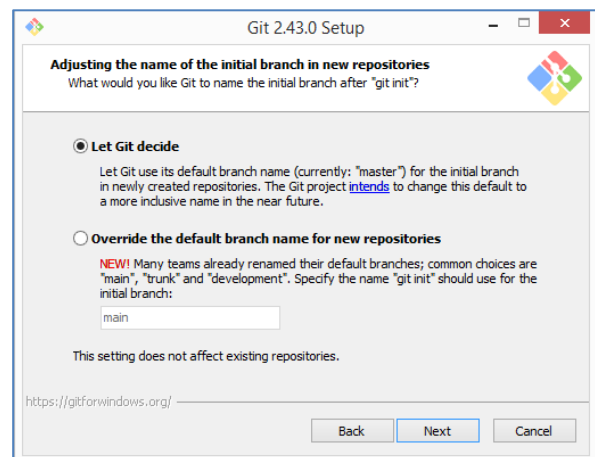
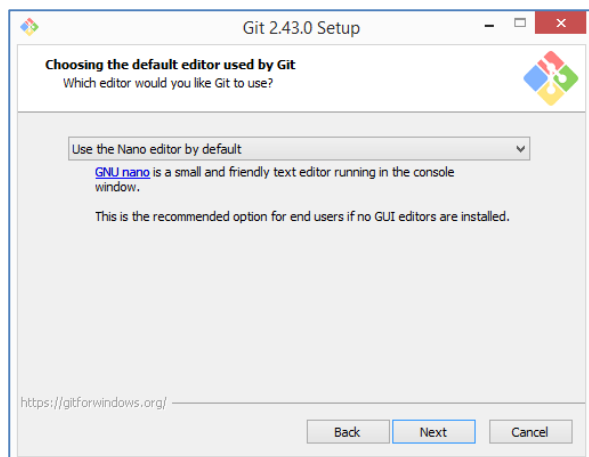
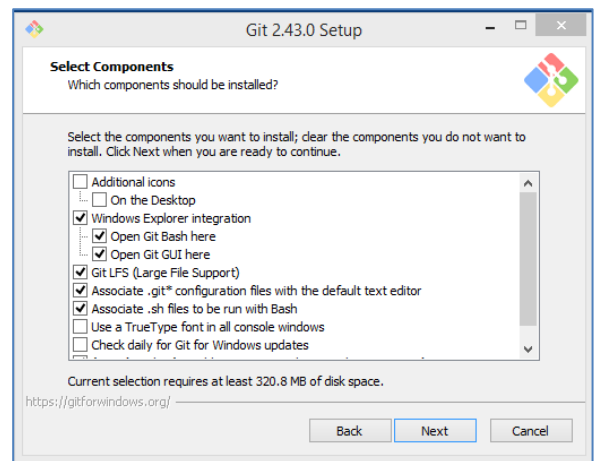
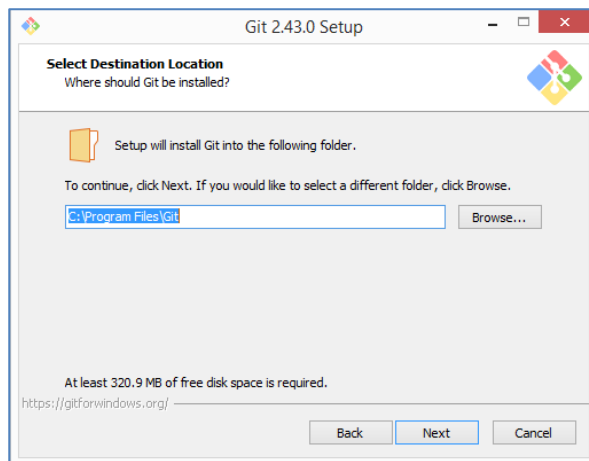
Основная цель использования такого инструмента как Git – совместная разработка. Библиотека является бесплатной и открытой к любым изменениям. Если вы хотите переписать или добавить какие-то функции это можно сделать в отдельной git-ветке, которую позже загрузить на Git Hub и слить с основной git-веткой, включив функции в состав библиотеки. Однако, если вы не используете Git, можете просто прислать m-файлы и их описание на mail@ge0mlib.com, я добавлю их на Git Hub, в документацию и в библиотеку.

3.2 Установка Git

Дистрибутив git скачивается с сайта <https://git-scm.com/downloads>. На момент написания текста это файл Git-2.43.0-64-bit.exe (для Win64). Установка git (**Figure 3.1**) выполняется с настройками по умолчанию, за исключением

- (1) замены редактора на Nano или другой удобный (третий скриншот),
- (2) выставления галочки на запуск Git Bash на последнем скриншоте.

Оба этих действия не критичны, поскольку после установки git можно самостоятельно и заменить редактор в настройках git, и запустить сам git.



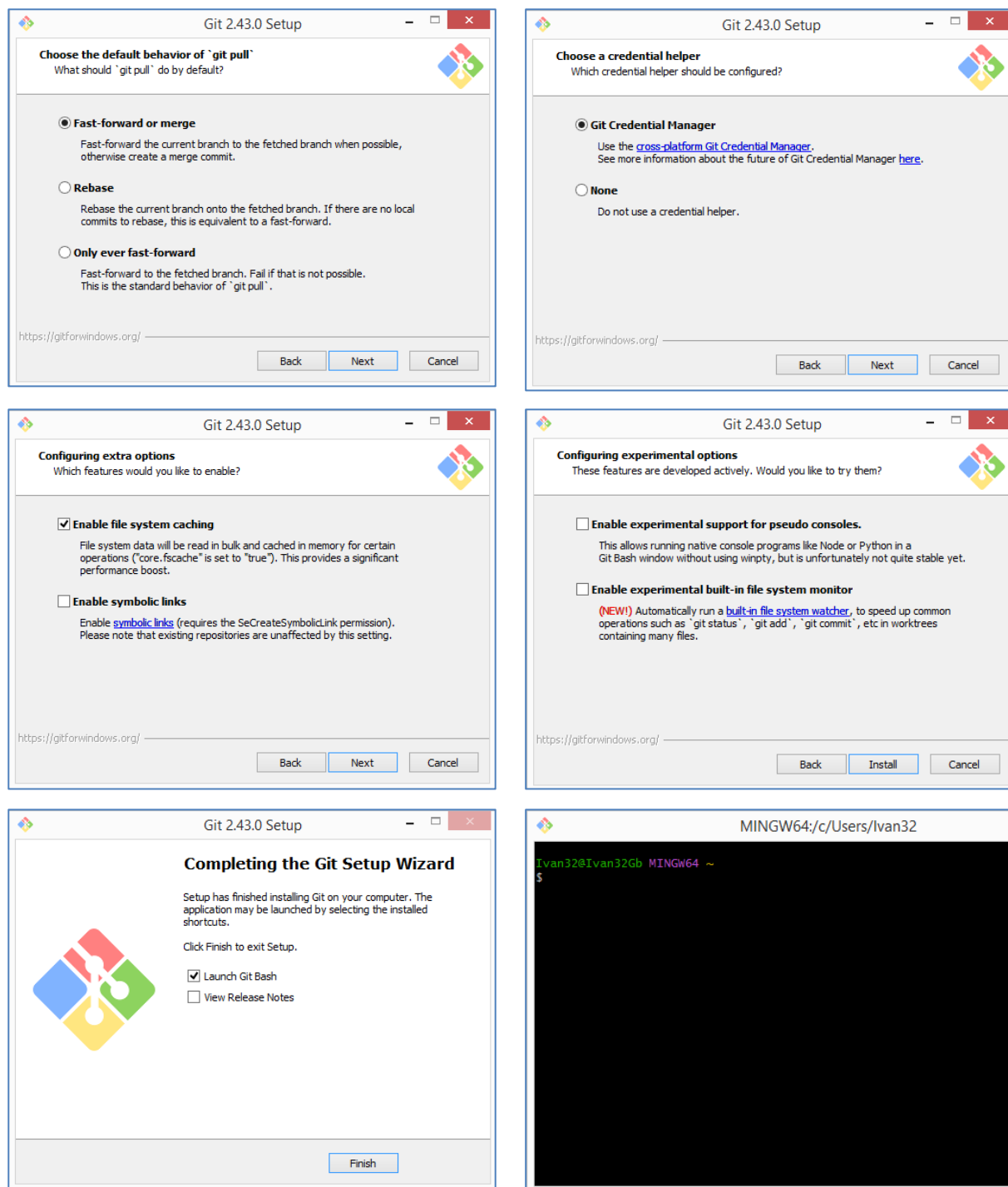
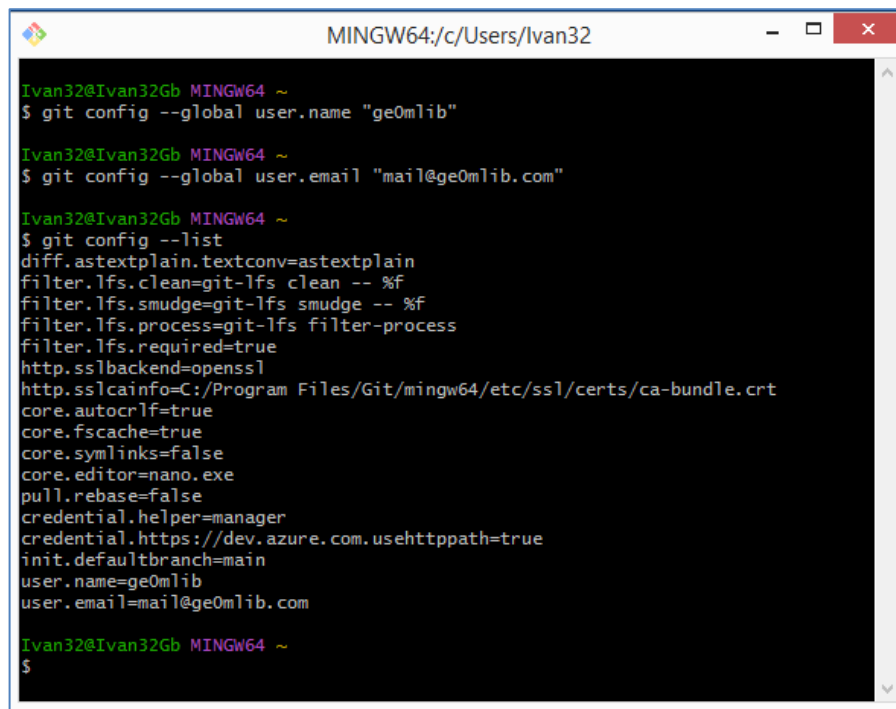


Figure 3.1 Установка Git

После установки, в окне git необходимо выполнить приведенные ниже команды, прописав свои user.name и user.email (**Figure 3.2**). Команды можно копировать и вставлять, используя правую клавишу мыши.

```
git config --global user.name "xxx"
git config --global user.email "xxxx@xxx.xxx"
git config --list
exit
```



```
MINGW64:/c/Users/Ivan32
Ivan32@Ivan32Gb MINGW64 ~
$ git config --global user.name "ge0mlib"
Ivan32@Ivan32Gb MINGW64 ~
$ git config --global user.email "mail@ge0mlib.com"
Ivan32@Ivan32Gb MINGW64 ~
$ git config --list
diff.astextplain.textconv=astextplain
filter.lfs.clean=git-lfs clean -- %f
filter.lfs.smudge=git-lfs smudge -- %f
filter.lfs.process=git-lfs filter-process
filter.lfs.required=true
http.sslbackend=openssl
http.sslcainfo=C:/Program Files/Git/mingw64/etc/ssl/certs/ca-bundle.crt
core.autocrlf=true
core.fscache=true
core.symlinks=false
core.editor=nano.exe
pull.rebase=false
credential.helper=manager
credential.https://dev.azure.com.usehttppath=true
init.defaultbranch=main
user.name=ge0mlib
user.email=mail@ge0mlib.com
Ivan32@Ivan32Gb MINGW64 ~
$
```

Figure 3.2 Выполнение команд для начальных установок git

После выполнения команды exit окно git закрывается.

3.3 Создание git-проекта и скачивание ge0mlib из репозитория

Перед работой с git лучше удалить все файлы из папки Functions, во избежание конфликтов между версиями файлов.

На первом шаге можно запустить git и перейти в папку Functions, используя команду cd, например:

```
cd c:/ge0mlib/Proj/Functions
```

Как альтернатива – можно открыть git напрямую в необходимой папке, по нажатию на правую кнопку мыши (**Figure 3.3**).

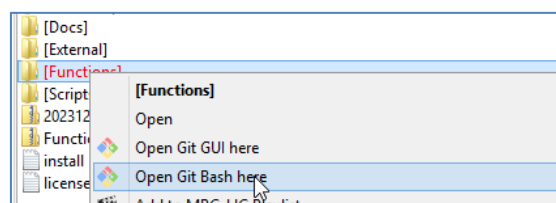


Figure 3.3 Открытие Git Bash из контекстного меню Windows

После запуска git необходимо ввести показанную ниже последовательность команд:

-- создание невидимой папки .git, в которой содержится информация по git:

```
git init
```

-- добавление пути к репозиторию на Git Hub:

```
git remote add orig https://github.com/ge0mlib/ge0mlib
```

-- копирование файлов репозитория в папку .git:

```
git pull orig main
```

Пример окна git и содержимого папки Functions, после выполнения команд показан на *Figure 3.4*.

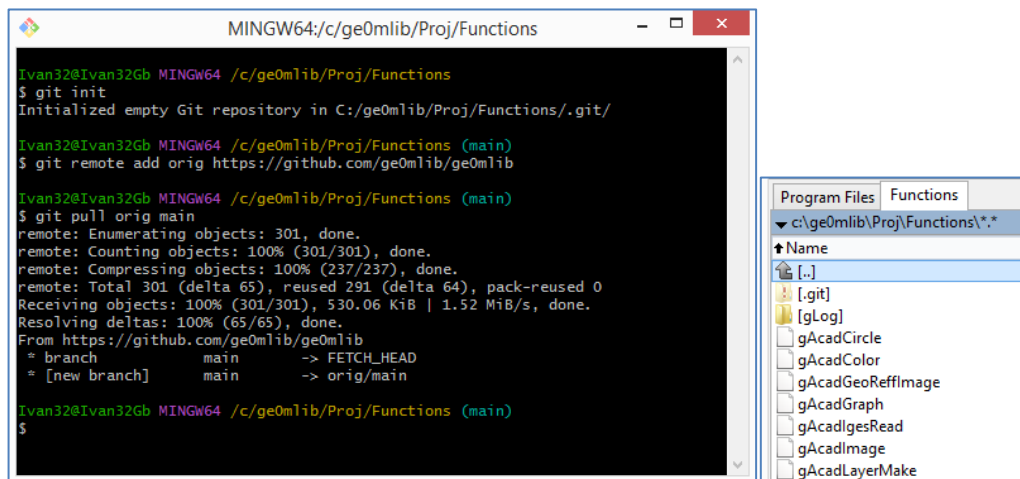


Figure 3.4 Окно git и содержимое папки Functions после создания репозитория

3.4 Перемещение по истории в git

Для вывода списка «снимков библиотеки» (commit) необходимо использовать команду:
`git log`
используйте клавишу “q” для выхода. При выполнении команды git выводит список «снимков» (commit) с информацией о дате и уникальным хэш-кодом для каждого commit (*Figure 3.5*).

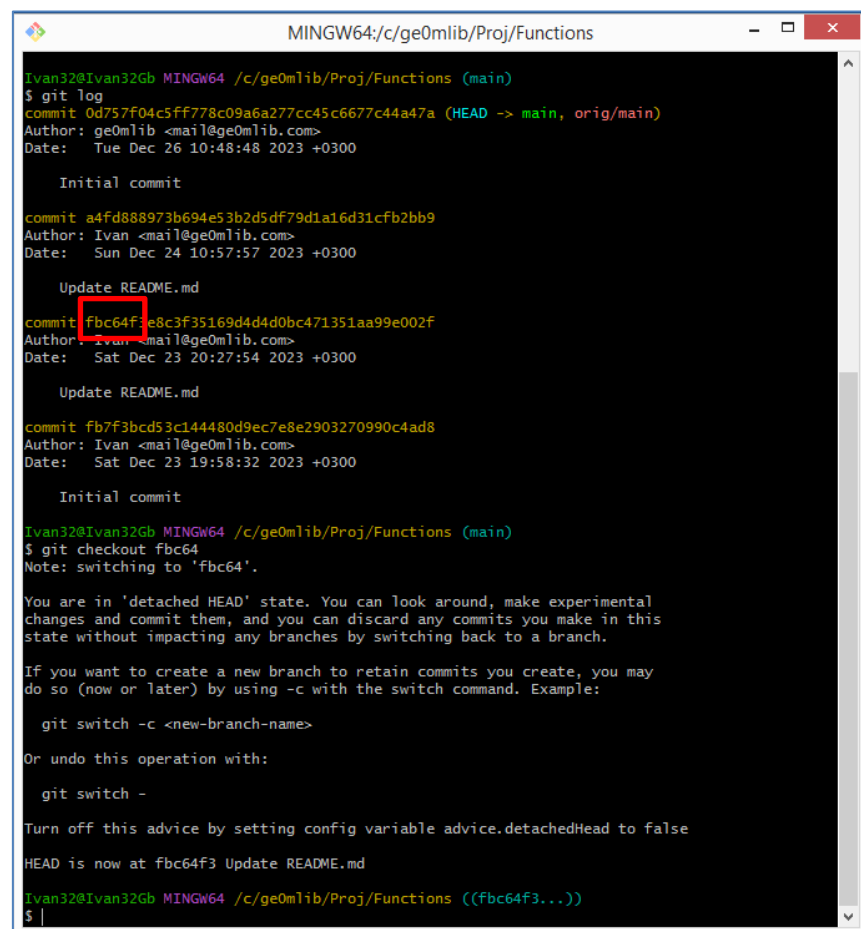


Figure 3.5 Окно git со списком commit

Необходимо выбрать commit по дате (так, чтобы дата была ближайшей датой до даты скрипта) и использовать хэш-код (в нашем примере это fbc64) с командой “checkout”:

```
git checkout fbc64
```

После выполнения команды git-указатель HEAD переместится на commit с хэш-кодом fbc64 и git изменит содержимое папки Functions в соответствии с содержимым на дату commit. Результат изменения содержимого папки Functions, для нашего примера, показан на [Figure 3.6](#).

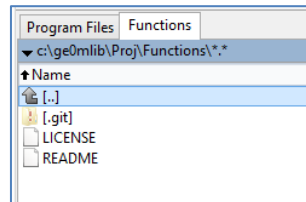


Figure 3.6 Файлы в папке Functions в результате применения команды “checkout”

После такого смещения указателя HEAD очень желательно создать для данного commit новую ветку (например, с именем my001) при помощи команды:

```
git branch my001
```

Теперь мы можем переключаться между «последней версией» библиотеки (ветка main) и «версией my001» (ветка my001) используя команды:

```
git checkout main
```

```
git checkout my001
```

при этом содержимое папки Functions каждый раз будет заменяться. Строго говоря, заменяться будут не все файлы, а только *отслеживаемые файлы*, которые содержатся в commit (с соответствующими именами и хэш-кодом). Если мы внесем изменения в такой *отслеживаемый файл*, то при попытке переключения между версиями библиотеки будет выдаваться сообщение об ошибке, отмеченное прямоугольником на [Figure 3.7](#).

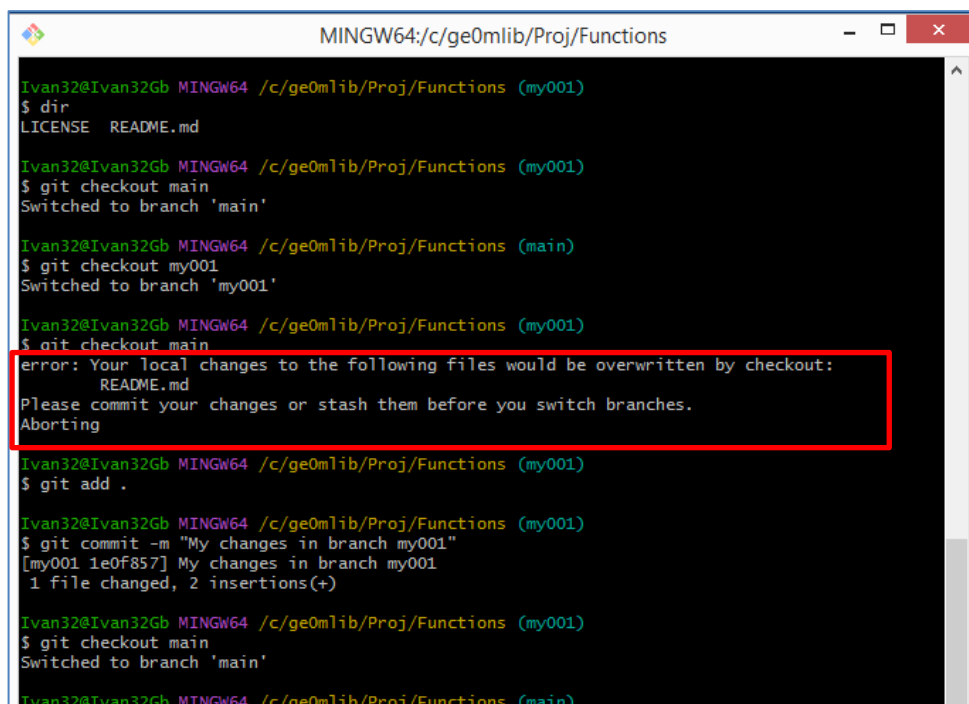


Figure 3.7 Ошибка git, говорящая о необходимости сохранить последние изменения как commit

Сообщение об ошибке выдается, потому что в случае переключения последние внесенные изменения были бы потеряны (стерты). Чтобы выполнить переключение, необходимо создать новый commit с внесенными изменениями. Для этого можно выполнить следующую последовательность команд:

```
git add .  
git commit -m "My changes in branch my001"  
git checkout main
```

Дальнейшее «углубление в систему команд» требует полноценного изучения работы с git, что не входит в задачи данного документа.

3.5 Запуск графического интерфейса

Выбрать commit по дате можно не только в терминале, но и используя графический интерфейс. Для этого по правой кнопке мыши надо выбрать не “Open Git Bash here”, а “Open Git GUI here” и далее – Visualize All Branch History (*Figure 3.8*).

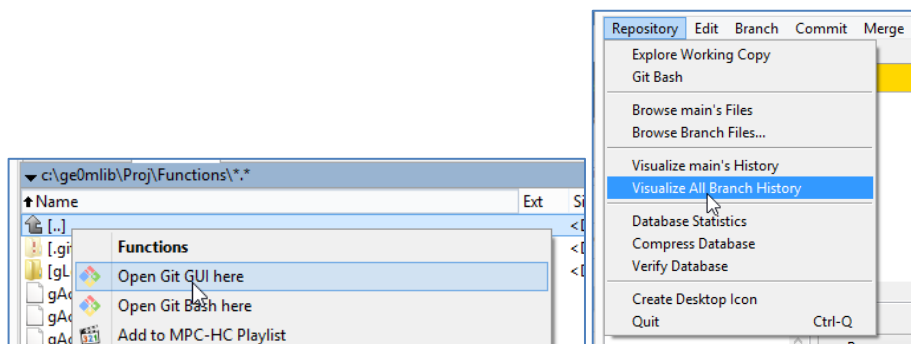


Figure 3.8 Открытие Git GUI через контекстное меню Windows

В открывшемся окне можно посмотреть ветки, commit и хэш-кодов для них (*Figure 3.9*).

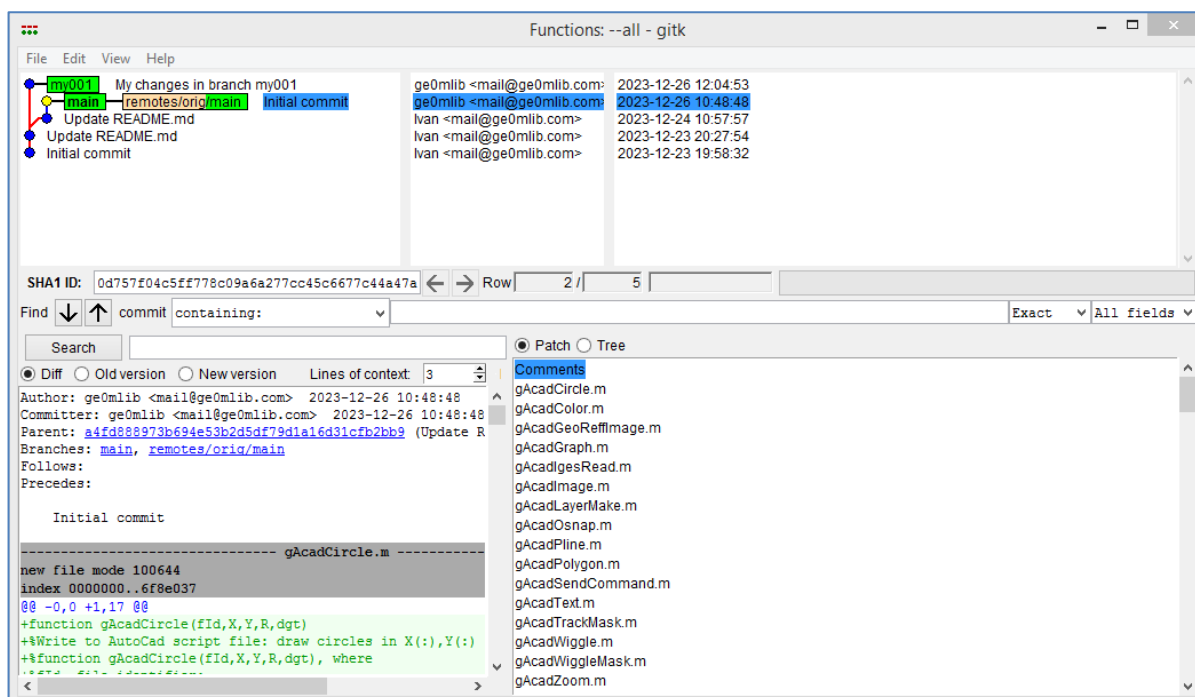


Figure 3.9 Интерфейс Git GUI

4 Структура скрипта и команд скрипта

Для написания скриптов для библиотеки ge0mlib существует определенный набор правил, которых лучше придерживаться, чтобы использование скриптов другими людьми было максимально удобным для них.

Каждый скрипт может выполнять несколько *команд скрипта*. Для каждой команды скрипта могут передаваться несколько *параметров команды скрипта*. Для выполнения команды скрипта, необходимо в окне MatLab или Octave ввести *команду-на-исполнение-скрипта* вида:

{'LoadMag1x8_r01','d:\002\1mag\0052_MAG_'};MagToolR01;

здесь:

MagToolR01 – название скрипта (имя файла со скриптом, который был скопирован в папку Scripts);

{'LoadMag1x8_r01','d:\002\1mag\0052_MAG_'} – параметры команды скрипта, причем первым параметром всегда идет имя команды скрипта. В фигурных скобках, через запятую, может быть введено множество параметров команды скрипта, которые требуются для ее исполнения;

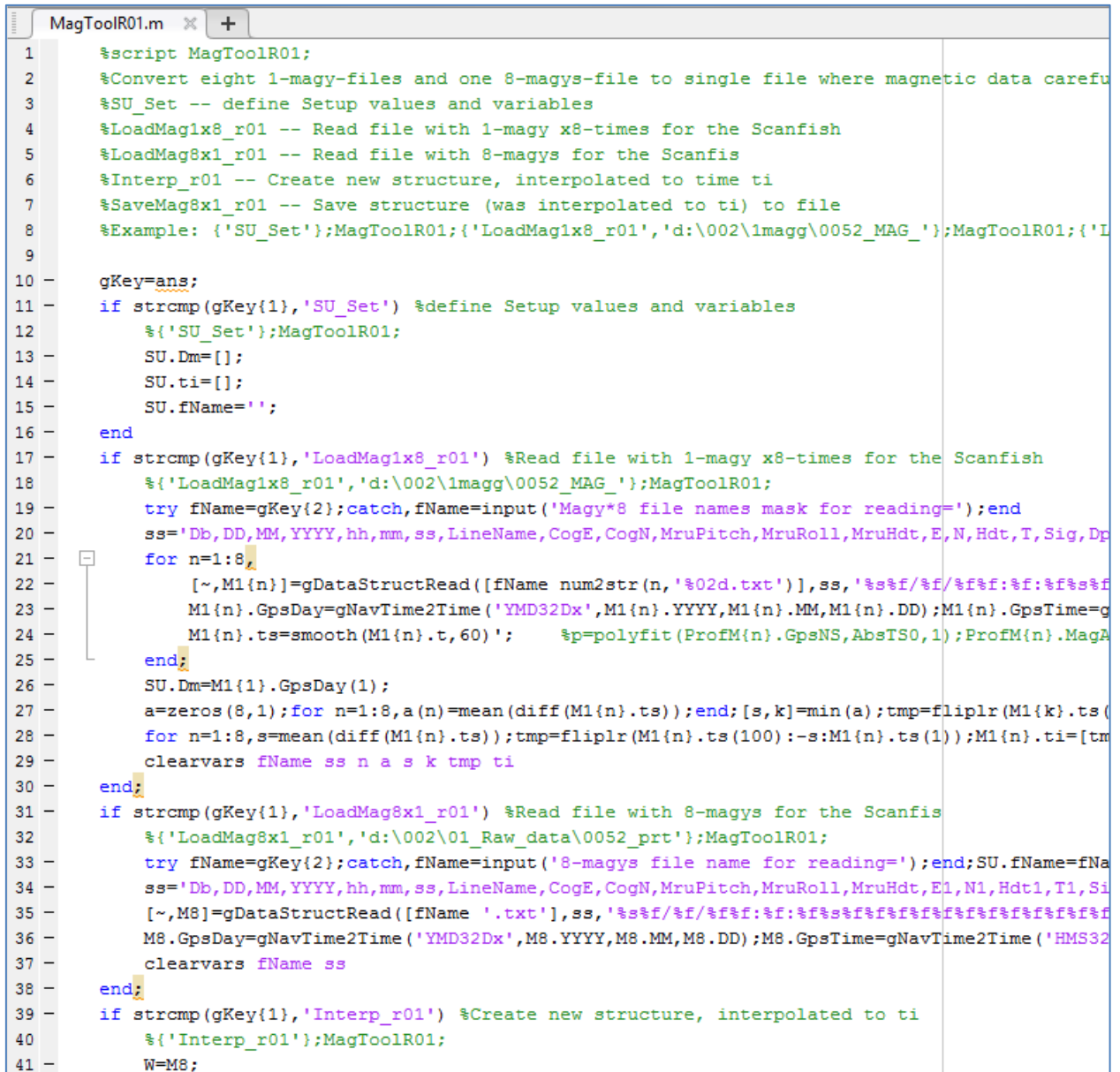
'LoadMag1x8_r01' – имя команды скрипта, по которой выполняется определенный участок кода скрипта;

'd:\002\1mag\0052_MAG_' – параметр команды скрипта, который как видно содержит путь к некоторому файлу, который будет использован при работе скрипта.

Рассмотрим структуру скрипта MagToolR01 (для пересчета данных магнитной съемки), которая приведена на [Figure 4.1](#) (показана только часть кода). При создании скриптов такой структуры желательно придерживаться постоянно (номера строк, разумеется, указаны для конкретного примера).

- Имя скрипта (строка 1);
- Краткое назначение скрипта (строка 2)
- Список дополнительных библиотек необходимых для выполнения скрипта (строка 2);
- Описание команд скрипта и, при необходимости, параметров команд (строки 3–7);
- Несколько примеров с рабочими последовательностями команд, записанные после “Example” (в нашем случае такая последовательность одна, и она приведена в строке 8);
- Блоки, выполняемые по условиям для каждой из команд (условия с именами команд показаны в строках 11, 17 и 31);
- Краткое описание выполняемой команды в начале каждого блока (описание приведено как комментарий в строках 11, 17 и 31);
- Пример команды и параметров скрипта для каждого блока (примеры приведены в виде комментариев в строках 12, 18, 32);
- Участок кода, запрашивающий ввод параметров скрипта, если они были пропущены при вводе *команды-на-исполнение-скрипта*. Данный код также содержит текст-описание для каждого параметра скрипта (строки 19, 33);

- Функция `clearvars`, исполняемая в конце каждого блока (соответствующего команде скрипта) для удаления лишних/временных переменных (строки 29, 37);
- Дата создания скрипта, почта для связи, среда MatLab и/или Octave с указанием версии, для которой проводилось тестирование работы скрипта (комментарий в последней строке 69: `mail@ge0mlib.com 10/12/2023 MatLab2018b&Octave8.4.0`).



```

1 %script MagToolR01;
2 %Convert eight 1-magy-files and one 8-magys-file to single file where magnetic data carefu
3 %SU_Set -- define Setup values and variables
4 %LoadMag1x8_r01 -- Read file with 1-magy x8-times for the Scanfish
5 %LoadMag8x1_r01 -- Read file with 8-magys for the Scanfis
6 %Interp_r01 -- Create new structure, interpolated to time ti
7 %SaveMag8x1_r01 -- Save structure (was interpolated to ti) to file
8 %Example: {'SU_Set'};MagToolR01;{'LoadMag1x8_r01','d:\002\lmagg\0052_MAG_'};MagToolR01;{'L
9
10 gKey=ans;
11 if strcmp(gKey{1},'SU_Set') %define Setup values and variables
12     {'SU_Set'};MagToolR01;
13     SU.Dm=[];
14     SU.ti=[];
15     SU.fName='';
16 end
17 if strcmp(gKey{1},'LoadMag1x8_r01') %Read file with 1-magy x8-times for the Scanfish
18     {'LoadMag1x8_r01','d:\002\lmagg\0052_MAG_'};MagToolR01;
19     try fName=gKey{2};catch,fName=input('Magy*8 file names mask for reading=');end
20     ss='Db,DD,MM,YYYY,hh,mm,ss,LineName,CogE,CogN,MruPitch,MruRoll,MruHdt,E,N,Hdt,T,Sig,Dp
21     for n=1:8;
22         [~,M1{n}]=gDataStructRead([fName num2str(n,'%02d.txt')],ss,'%s%f/%f/%f%f:%f%f%s%f
23         M1{n}.GpsDay=gNavTime2Time('YMD32Dx',M1{n}.YYYY,M1{n}.MM,M1{n}.DD);M1{n}.GpsTime=g
24         M1{n}.ts=smooth(M1{n}.t,60)'; %p=polyfit(ProfM{n}.GpsNS,AbsTSO,1);ProfM{n}.MagA
25     end;
26     SU.Dm=M1{1}.GpsDay(1);
27     a=zeros(8,1);for n=1:8,a(n)=mean(diff(M1{n}.ts));end;[s,k]=min(a);tmp=fliplr(M1{k}.ts(
28     for n=1:8,s=mean(diff(M1{n}.ts));tmp=fliplr(M1{n}.ts(100):-s:M1{n}.ts(1));M1{n}.ti=[tm
29     clearvars fName ss n a s k tmp ti
30 end;
31 if strcmp(gKey{1},'LoadMag8x1_r01') %Read file with 8-magys for the Scanfis
32     {'LoadMag8x1_r01','d:\002\01_Raw_data\0052_prt'};MagToolR01;
33     try fName=gKey{2};catch,fName=input('8-magys file name for reading=');end;SU.fName=fNa
34     ss='Db,DD,MM,YYYY,hh,mm,ss,LineName,CogE,CogN,MruPitch,MruRoll,MruHdt,E1,N1,Hdt1,T1,Si
35     [~,M8]=gDataStructRead([fName '.txt'],ss,'%s%f/%f/%f%f:%f%f%s%f%f%f%f%f%f%f%f%f%f
36     M8.GpsDay=gNavTime2Time('YMD32Dx',M8.YYYY,M8.MM,M8.DD);M8.GpsTime=gNavTime2Time('HMS32
37     clearvars fName ss
38 end;
39 if strcmp(gKey{1},'Interp_r01') %Create new structure, interpolated to ti
40     {'Interp_r01'};MagToolR01;
41     W=M8;

```

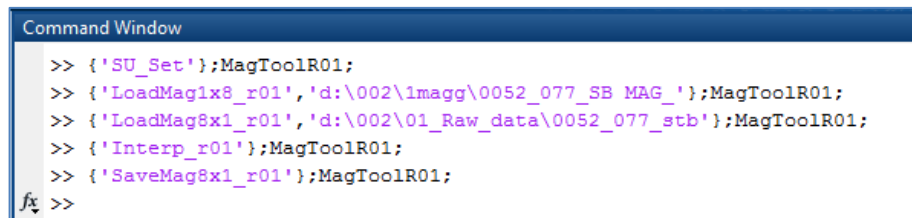
Figure 4.1 Пример кода скрипта MagToolR01

Кроме того, существуют определенные имена переменных (в приведенном примере это SU) и определенные названия полей для структур (в приведенном примере это DD, MM, YYYY, hh, mm, ss и другие). Которые являются специализированными и часто встречающимися. Использование именно этих имен сделает текст скрипта более читаемым (а иногда вообще является необходимым для исполнения функций, в коде которых прописаны такие «специализированные» поля).

При описанном выше подходе обработка данных будет реализовываться как последовательное выполнение нескольких команд скрипта с соответствующими параметрами. В

нашем примере это будет чтение нескольких файлов с данными магнитометров (команды LoadMag1x8_r01, LoadMag8x1_r01), некие математические расчеты на основе загруженных данных (Interp_r01), вывод результатов расчетов в файл (команда SaveMag8x1_r01).

Пример окна с выполненной рабочей последовательностью команд скрипта MagToolR01 показан на *Figure 4.2*.



```
Command Window
>> {'SU_Set'};MagToolR01;
>> {'LoadMag1x8_r01','d:\002\1magg\0052_077_SB_MAG_'};MagToolR01;
>> {'LoadMag8x1_r01','d:\002\01_Raw_data\0052_077_stb'};MagToolR01;
>> {'Interp_r01'};MagToolR01;
>> {'SaveMag8x1_r01'};MagToolR01;
fx >>
```

Figure 4.2 Окно MatLab с выполненной последовательностью команд для MagToolR01

Заключение

В документе рассмотрены:

- установка свободно-распространяемой среды программирования GNU Octave и дополнительных библиотек к ней;
- установка библиотеки ge0mlib;
- установка git и команды для получения «снимка» (commit) библиотеки ge0mlib на некоторый момент времени в прошлом.

Приведен пример скрипта для пересчета данных магнитной съемки (UXO). Рассмотрено, где в тексте скрипта можно найти следующую информацию:

- Список дополнительных библиотек необходимых для выполнения скрипта;
- Описание команд скрипта и параметров команд;
- Примеры с рабочими последовательностями команд;
- Дату создания скрипта;
- Среду, использовавшуюся при тестировании скрипта (MatLab, Octave);
- Электронную почту разработчика скрипта, чтобы написать ему все что вы о нем думаете.

Приведенная информация позволяет выполнить относительно быструю установку программного обеспечения и перейти к обработке данных с помощью скрипта (m-файла).