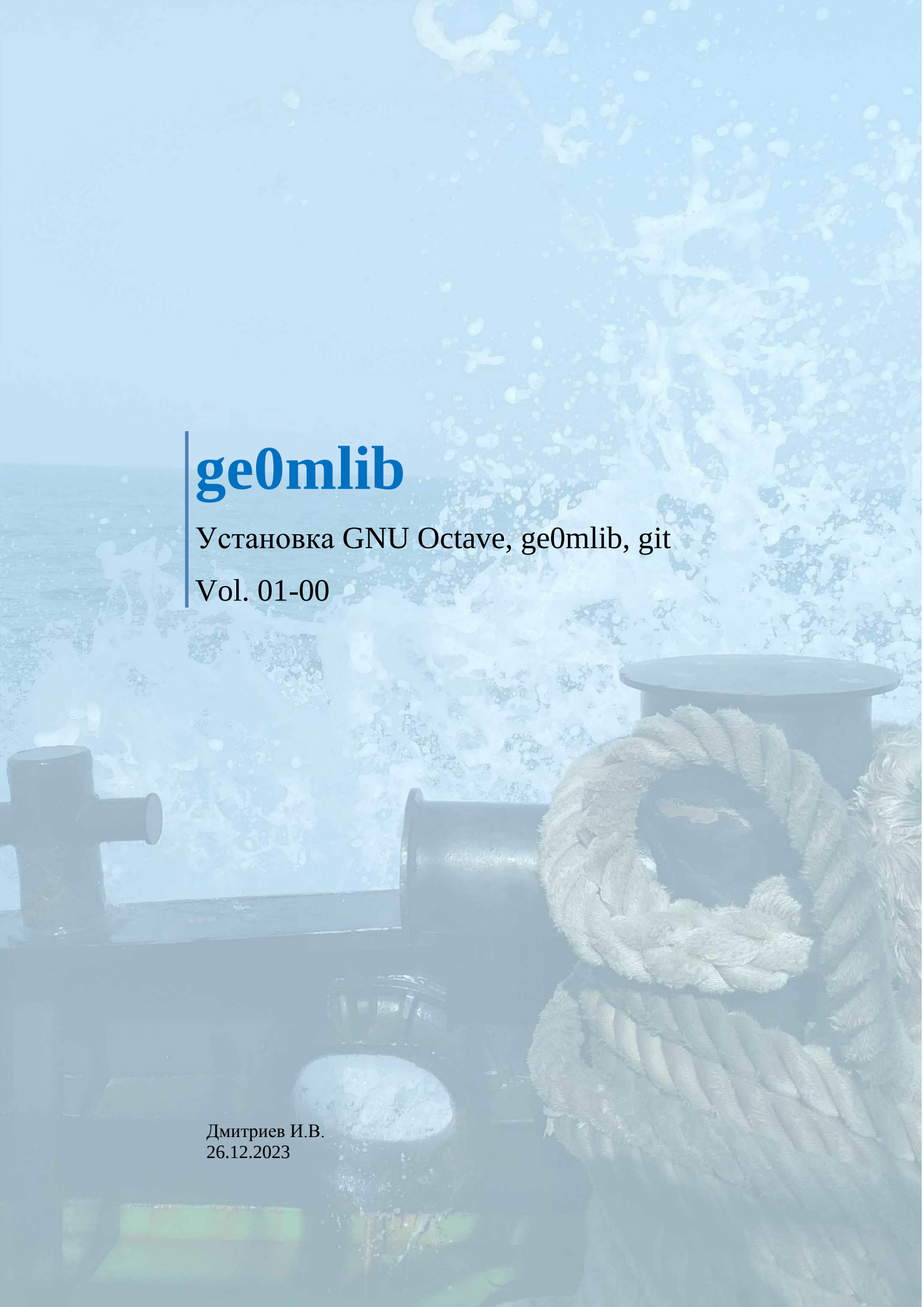


ge0mlib

Установка GNU Octave, ge0mlib, git

Vol. 01-00

Дмитриев И.В.
26.12.2023

Содержание

Введение

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

Данный документ написан для не-программиста-пользователя-Windows, 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 или устные инструкции вида «обрабатывали этим скриптом» (описание каждой отдельной команды должно быть приведено в тексте скрипта, там же должны быть приведены примеры «рабочих последовательностей команд»).

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

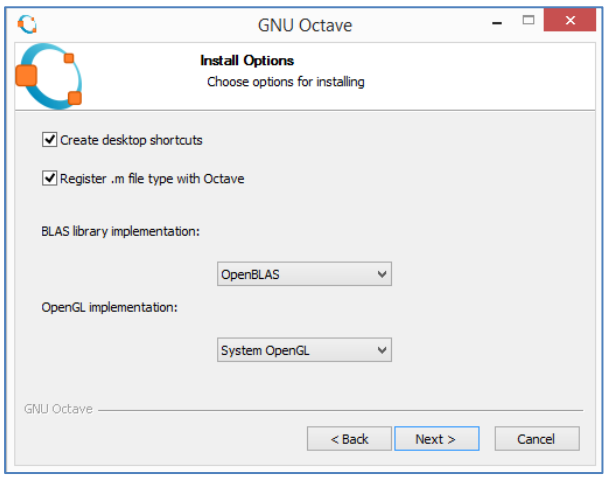
-- скачивание и установку GNU Octave (версия для Windows);
-- скачивание и установку дополнительных пакетов (библиотек) для GNU Octave (Windows);
-- скачивание и установку библиотеки ge0mlib;
-- скачивание и установку Git (версия для Windows);
-- восстановление «снимка» библиотеки ge0mlib на дату создания скрипта при помощи Git;
-- работу с командами скрипта (описание принципов построения команд скрипта и информации, содержащейся в тексте скрипта).

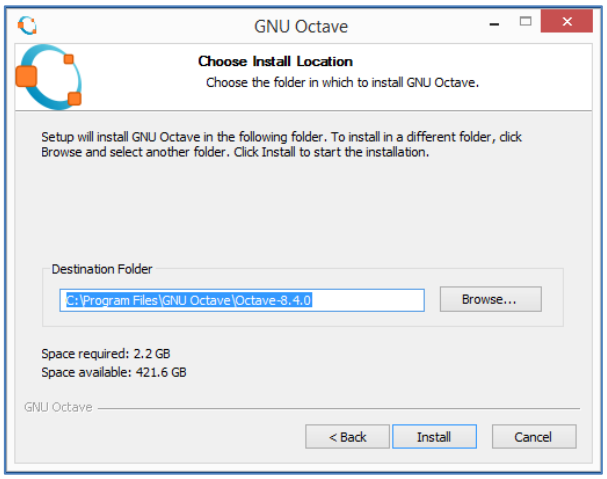
Текст документа включает чуть больше информации, чем необходимо. Однако эта информация позволяет чувствовать себя более комфортно при работе с GNU Octave, ge0mlib и Git. Такой «необязательный к прочтению» текст подсвечен синим цветом.

1 GNU Octave

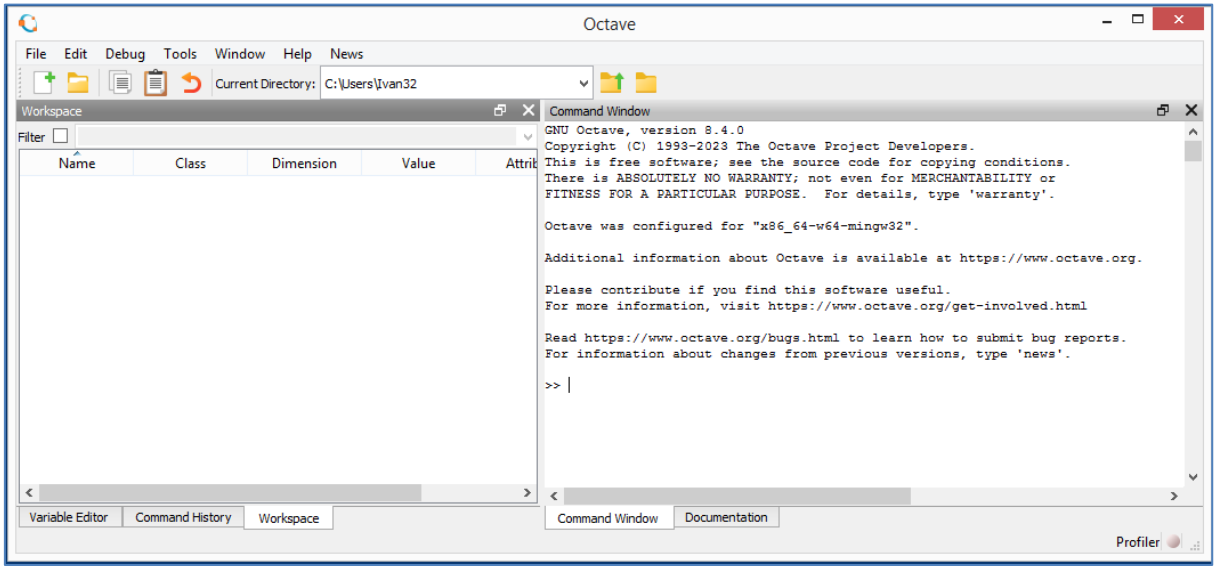
1.1 Установка GNU Octave

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

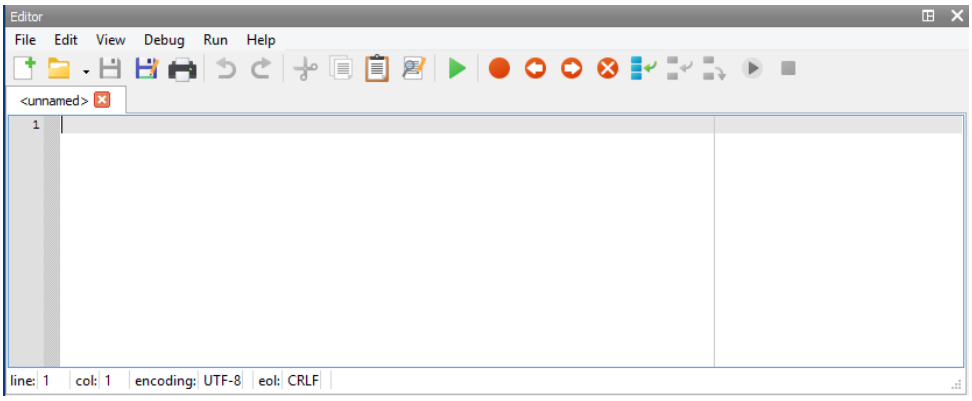




Командное окно Octave после установки, с перемещенными для удобства вкладками показано ниже. Обратите внимание на вкладки Variable Editor, Command History и Workspace.



Окно отладчика m-файлов показано ниже. 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).



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 выглядят полезными для решения задач морской геофизики. Разумеется, выборка очень субъективная 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 boootstrap 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.).

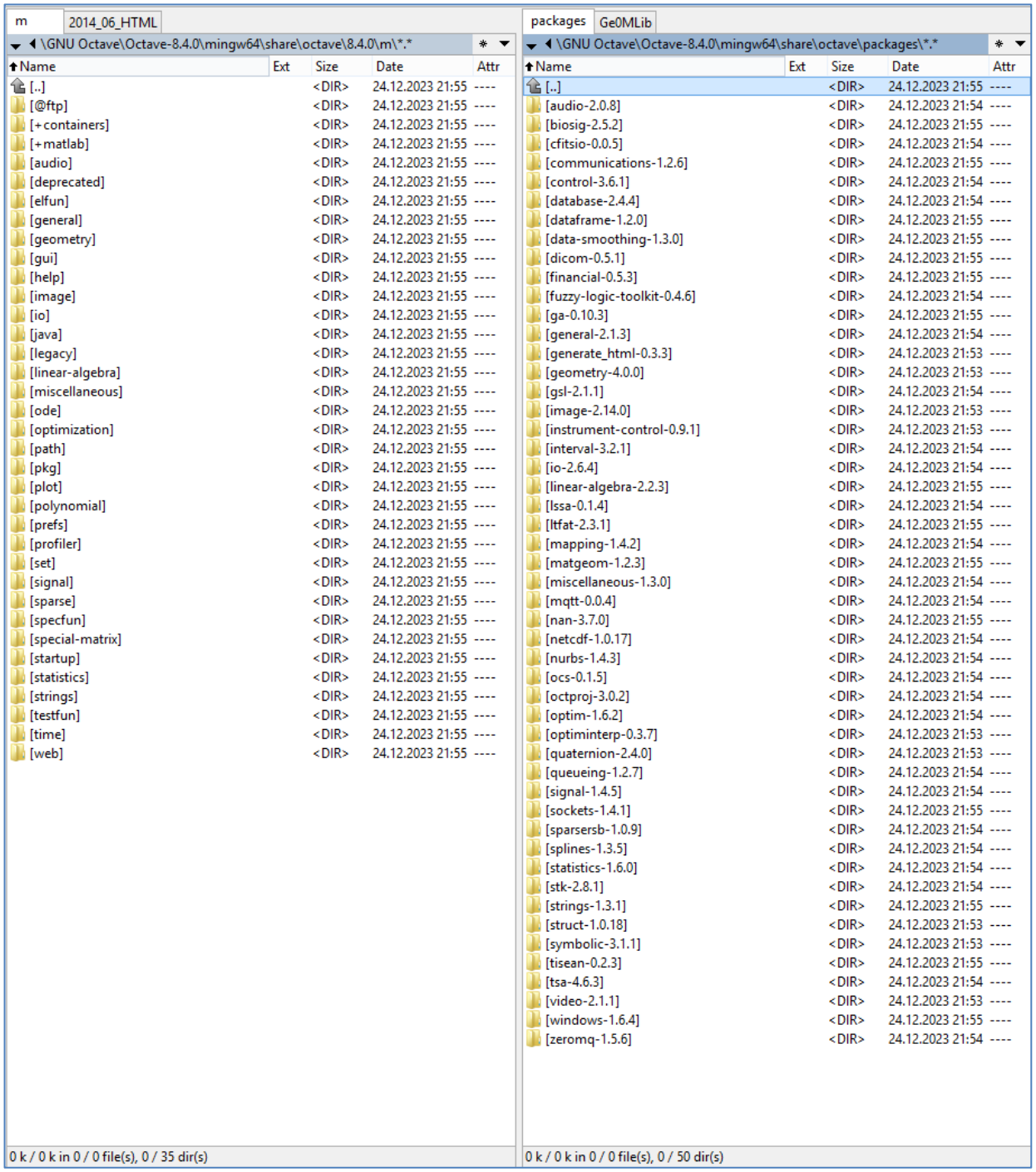
strings – Additional functions for manipulation and analysis of strings.

struct – Additional structure manipulation functions.

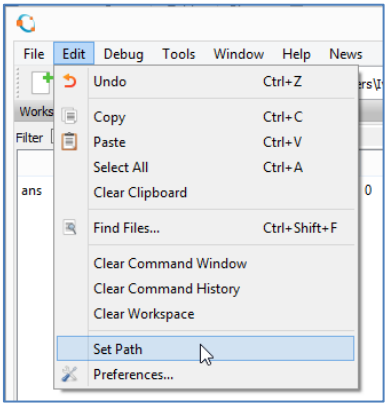
Что бы Octave «увидела» функции пакета, путь к нему должен быть «*прописан в системе*». В корневой папке Octave присутствуют две папки с пакетами:

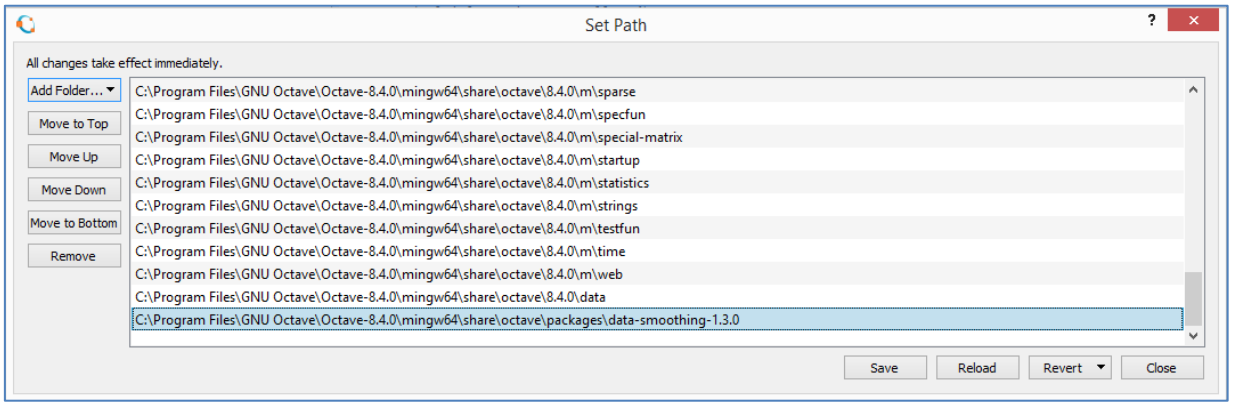
1) c:\Program Files\GNU Octave\Octave-8.4.0\mingw64\share\octave\8.4.0\m\ – папка содержит «интегрированные в систему» пакеты, такие как signal и statistics;
2) c:\Program Files\GNU Octave\Octave-8.4.0\mingw64\share\octave\packages\ – папка содержит не интегрированные пакеты, пути для которых исходно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

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

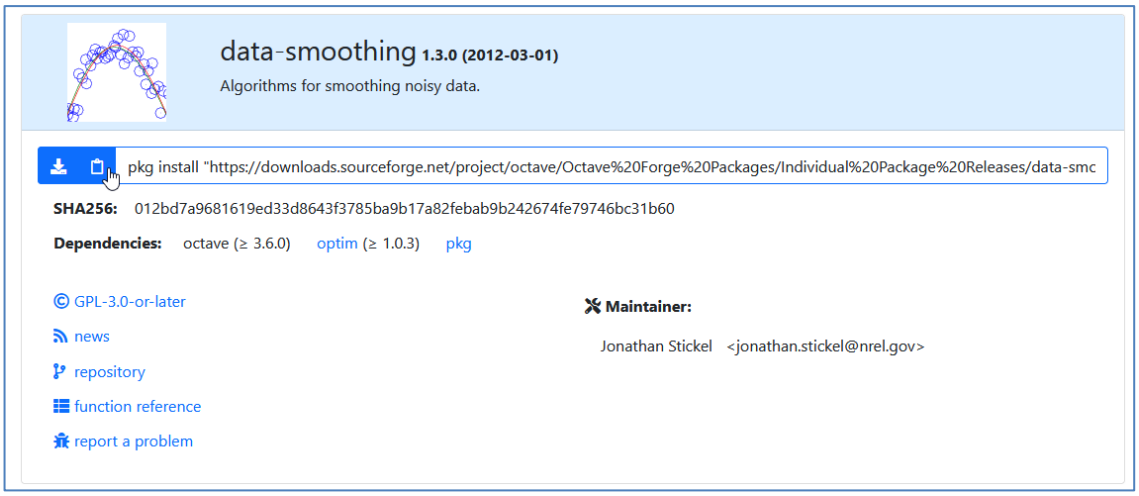


Посмотреть какие пути уже *прописаны* в Octave можно при помощи Path Browser (прописаны пути ко всем пакетам в папке “m\”). Используя кнопку “Add Folder” можно добавить пути к дополнительным пакетам, после чего необходимо нажать на кнопку “Save”, чтобы сохранить список путей, и он автоматически загружался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



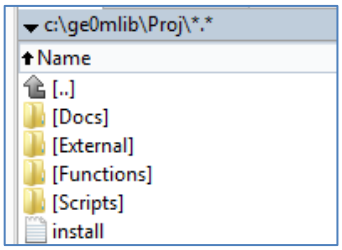


Если требуемый пакет в указанных выше папках отсутствует, необходимо найти его в Internet. При заходе на страницу пакета data-smooth, мы видим показанное ниже окно. Копирование команды “pkg install” с путем в командное окно Octave, и ее последующее выполнение, должны установить пакет data-smooth напрямую из Internet. Также можно скачать пакет и использовать команду “pkg install” (с соответствующим путем) для его оффлайн установки. Возможно, после установки потребуется дополнительно *прописать* путь к пакету.



2 Установка ge0mlib (MatLab, GNU Octave)

Скачать библиотеку ge0mlib можно по ссылке https://ge0mlib.com/g/ge0mlib.zip, после чего распаковать архив в удобное место. Архив содержит папки, показанные на рисунке ниже.



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;

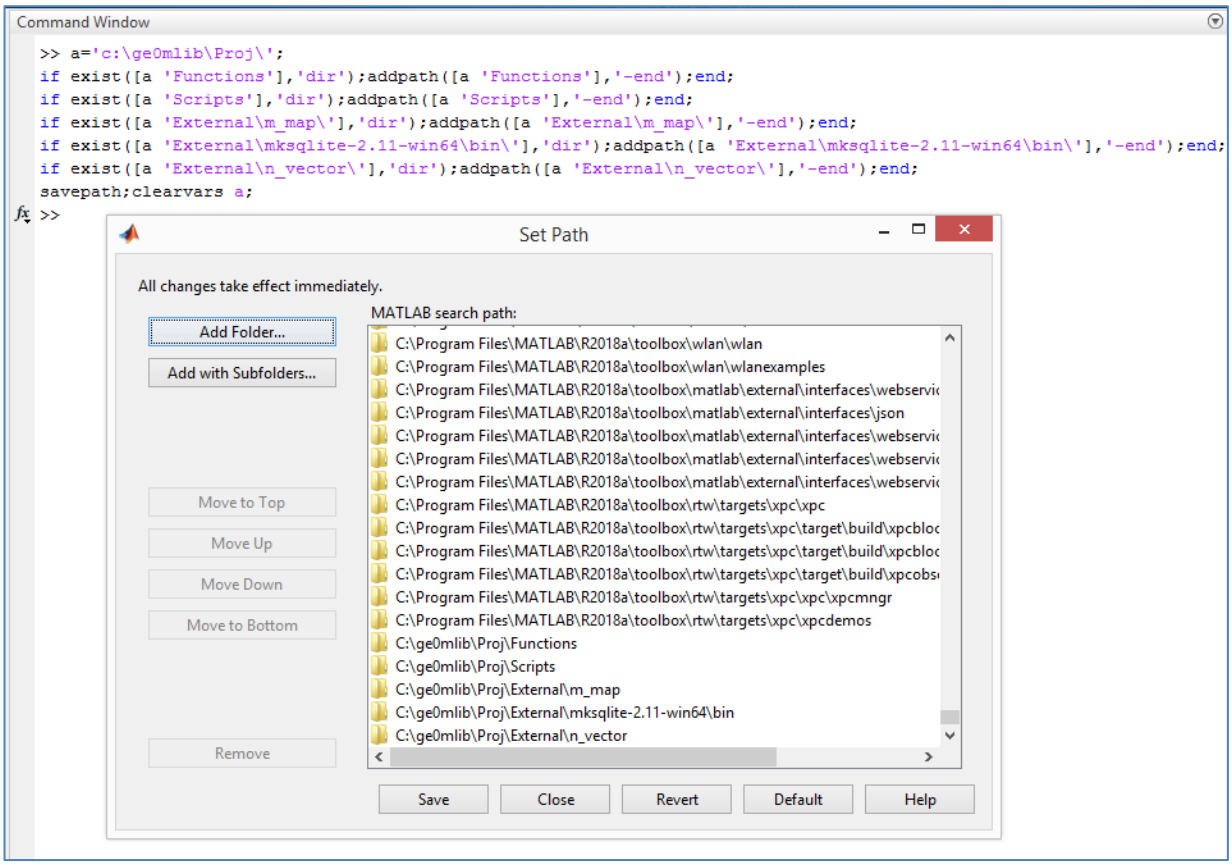
-- Functions – функции библиотеки ge0mlib (содержимое папки выложено на Git Hub: https://github.com/ge0mlib/ge0mlib);

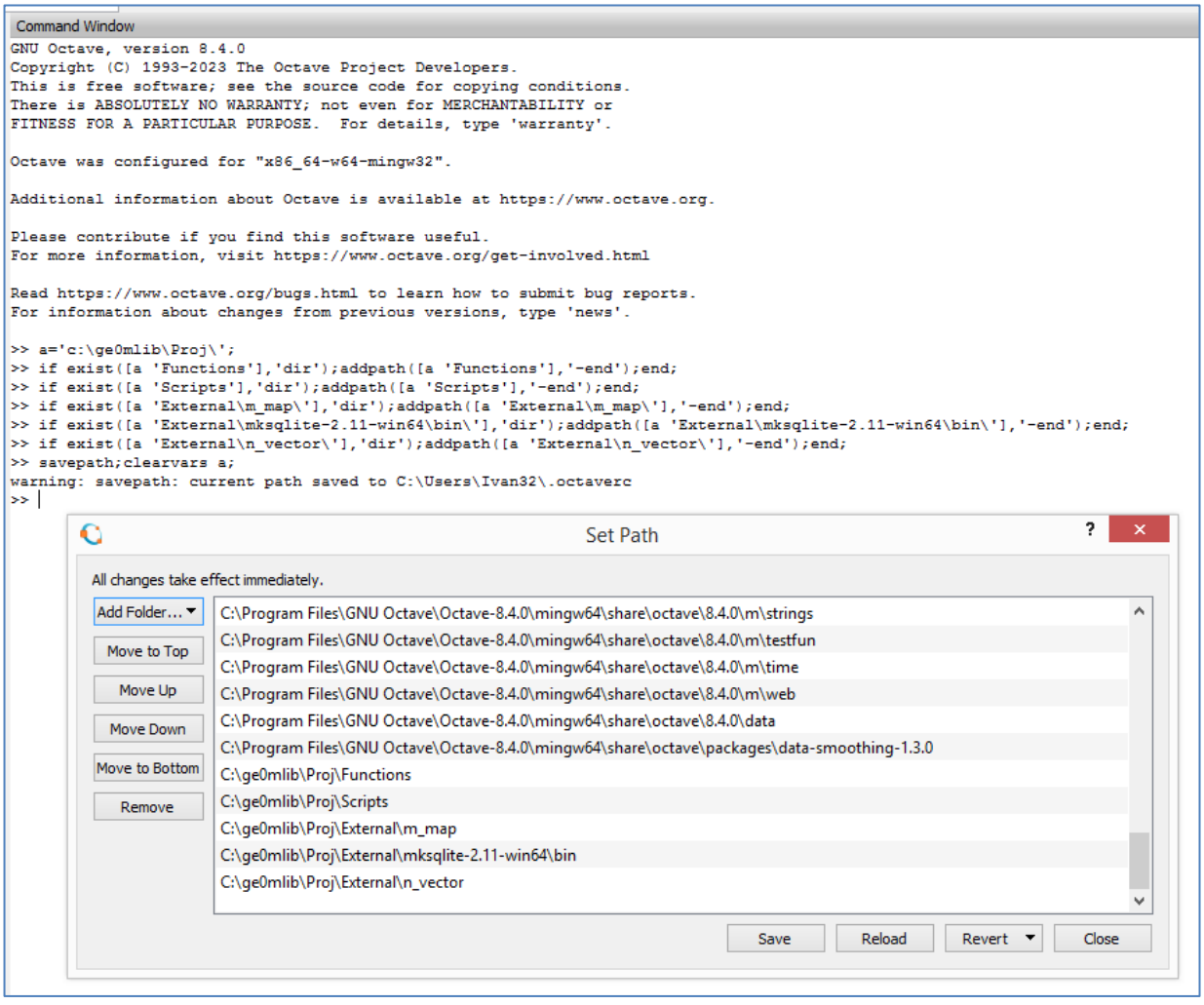
-- Scripts – пустая папка для скриптов, для локального использования. Сюда необходимо положить m-файл со скриптом, который требуется выполнить для «обработки данных». Если вы позже напишете свой скрипт или функцию, то можно также разместить их в этой папке;

-- 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. 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 «адаптировались» 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 Pyton). 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).


Command Window
>> a='c:\ge0mlib\Proj\';
if exist([a 'Functions'],'dir');addpath([a 'Functions'],'-end');end;
if exist([a 'Scripts'],'dir');addpath([a 'Scripts'],'-end');end;
if exist([a 'External\m_map\'],'dir');addpath([a 'External\m_map\'],'-end');end;
if exist([a 'External\mksqlite-2.11-win64\bin\'],'dir');addpath([a 'External\mksqlite-2.11-win64\bin\'],'-end');end;
if exist([a 'External\n_vector\'],'dir');addpath([a 'External\n_vector\'],'-end');end;
savepath;clearvars a;
>>
Set Path
All changes take effect immediately.
MATLAB search path:
Add Folder...
Add with Subfolders...
Move to Top
Move Up
Move Down
Move to Bottom
Remove
C:\Program Files\MATLAB\R2018a\toolbox\wlan\wlan
C:\Program Files\MATLAB\R2018a\toolbox\wlan\wlanexamples
C:\Program Files\MATLAB\R2018a\toolbox\matlab\external\interfaces\webservic
C:\Program Files\MATLAB\R2018a\toolbox\matlab\external\interfaces\json
C:\Program Files\MATLAB\R2018a\toolbox\matlab\external\interfaces\webservic
C:\Program Files\MATLAB\R2018a\toolbox\matlab\external\interfaces\webservic
C:\Program Files\MATLAB\R2018a\toolbox\matlab\external\interfaces\webservic
C:\Program Files\MATLAB\R2018a\toolbox\rtw\targets\xpc\xpc
C:\Program Files\MATLAB\R2018a\toolbox\rtw\targets\xpc\target\build\xpcbloc
C:\Program Files\MATLAB\R2018a\toolbox\rtw\targets\xpc\target\build\xpcbloc
C:\Program Files\MATLAB\R2018a\toolbox\rtw\targets\xpc\target\build\xpcobs
C:\Program Files\MATLAB\R2018a\toolbox\rtw\targets\xpc\xpc\xpcmngr
C:\Program Files\MATLAB\R2018a\toolbox\rtw\targets\xpc\xpcdemos
C:\ge0mlib\Proj\Functions
C:\ge0mlib\Proj\Scripts
C:\ge0mlib\Proj\External\m_map
C:\ge0mlib\Proj\External\mksqlite-2.11-win64\bin
C:\ge0mlib\Proj\External\n_vector
Save
Close
Revert
Default
Help


Command Window
GNU Octave, version 8.4.0
Copyright (C) 1993-2023 The Octave Project Developers.
This is free software; see the source code for copying conditions.
There is ABSOLUTELY NO WARRANTY; not even for MERCHANTABILITY or
FITNESS FOR A PARTICULAR PURPOSE. For details, type 'warranty'.
Octave was configured for "x86_64-w64-mingw32".
Additional information about Octave is available at https://www.octave.org.
Please contribute if you find this software useful.
For more information, visit https://www.octave.org/get-involved.html
Read https://www.octave.org/bugs.html to learn how to submit bug reports.
For information about changes from previous versions, type 'news'.
>> a='c:\ge0mlib\Proj\';
>> if exist([a 'Functions'],'dir');addpath([a 'Functions'],'-end');end;
>> if exist([a 'Scripts'],'dir');addpath([a 'Scripts'],'-end');end;
>> if exist([a 'External\m_map\'],'dir');addpath([a 'External\m_map\'],'-end');end;
>> if exist([a 'External\mksqlite-2.11-win64\bin\'],'dir');addpath([a 'External\mksqlite-2.11-win64\bin\'],'-end');end;
>> if exist([a 'External\n_vector\'],'dir');addpath([a 'External\n_vector\'],'-end');end;
>> savepath;clearvars a;
warning: savepath: current path saved to C:\Users\Ivan32\.octaverc
>>
Set Path
All changes take effect immediately.
Add Folder...
Move to Top
Move Up
Move Down
Move to Bottom
Remove
C:\Program Files\GNU Octave\Octave-8.4.0\mingw64\share\octave\8.4.0\m\strings
C:\Program Files\GNU Octave\Octave-8.4.0\mingw64\share\octave\8.4.0\m\testfun
C:\Program Files\GNU Octave\Octave-8.4.0\mingw64\share\octave\8.4.0\m\time
C:\Program Files\GNU Octave\Octave-8.4.0\mingw64\share\octave\8.4.0\m\web
C:\Program Files\GNU Octave\Octave-8.4.0\mingw64\share\octave\8.4.0\data
C:\Program Files\GNU Octave\Octave-8.4.0\mingw64\share\octave\packages\data-smoothing-1.3.0
C:\ge0mlib\Proj\Functions
C:\ge0mlib\Proj\Scripts
C:\ge0mlib\Proj\External\m_map
C:\ge0mlib\Proj\External\mksqlite-2.11-win64\bin
C:\ge0mlib\Proj\External\n_vector
Save
Reload
Revert
Close

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 – разумеется, в старой версии библиотеки будут присутствовать все старые баги (что является несомненным минусом), однако там будут отсутствовать баги новые (что является некоторым плюсом). Мы предполагаем, что скрипт написанный и протестированный *в своё время* выдавал адекватные результаты расчетов не подверженные багам, 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.

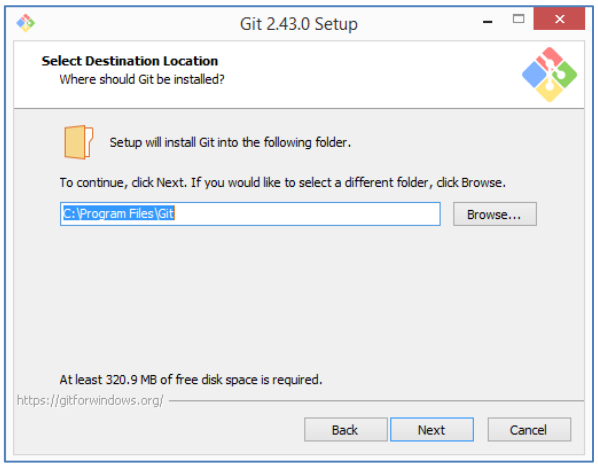
3.2 Установка Git

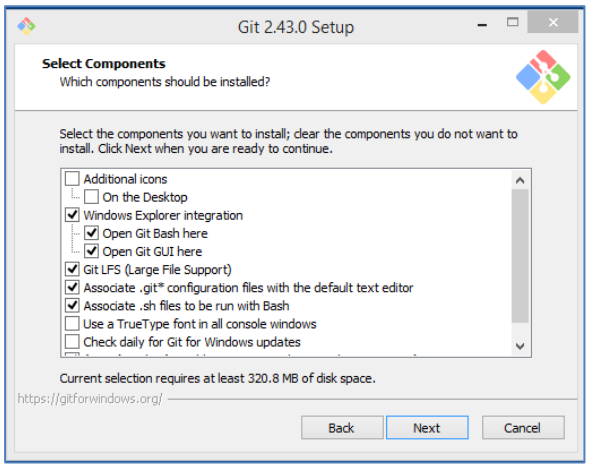
Дистрибутив git скачивается с сайта https://git-scm.com/downloads. На момент написания текста это файл Git-2.43.0-64-bit.exe (для Win64). Установка git выполняется с настройками по умолчанию, за исключением

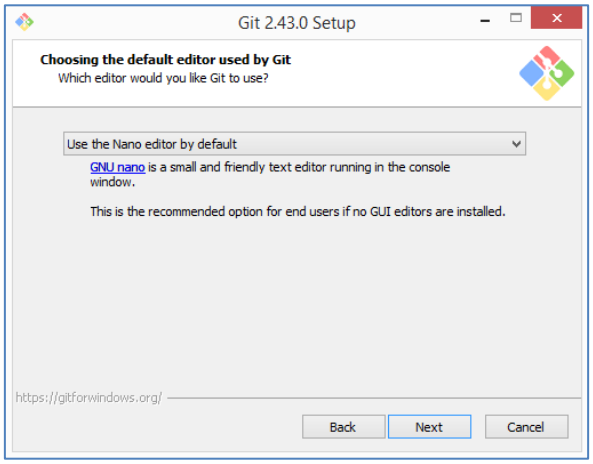
(1) замены редактора на Nano или другой удобный (третий скриншот),

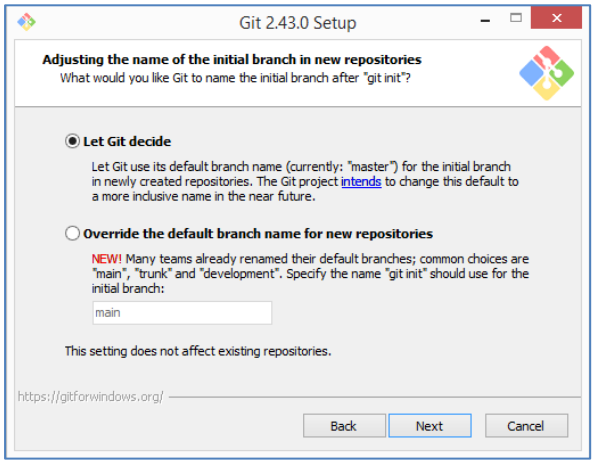
(2) выставления галочки на запуск Git Bash на последнем скриншоте.

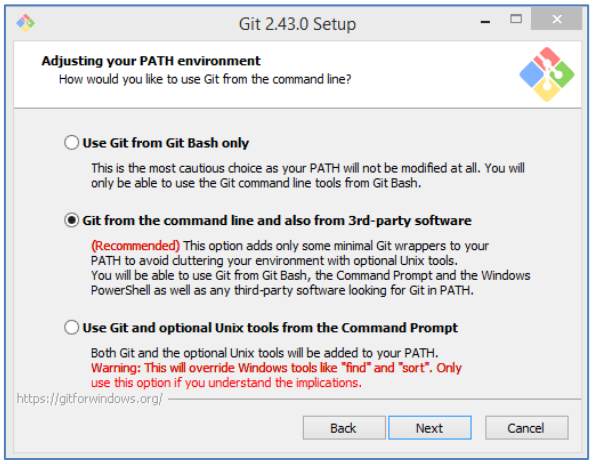
Оба эти действия не критичны, поскольку после установки git можно самостоятельно и заменить редактор в настройках git и запустить сам git.

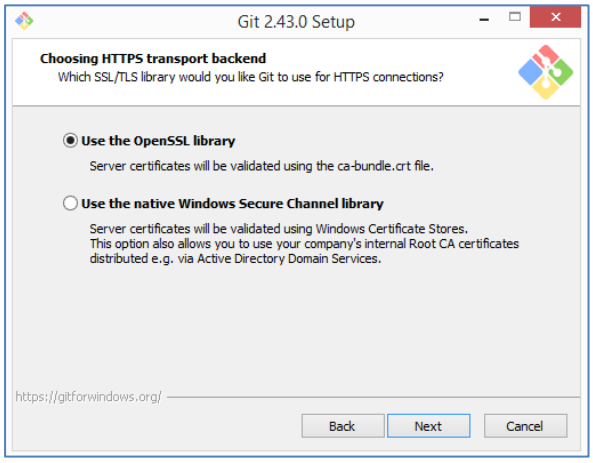

Git 2.43.0 Setup
Select Destination Location
Where should Git be installed?
Setup will install Git into the following folder.
To continue, click Next. If you would like to select a different folder, click Browse.
C:\Program Files\Git
Browse...
At least 320.9 MB of free disk space is required.
https://gitforwindows.org/
Back
Next
Cancel

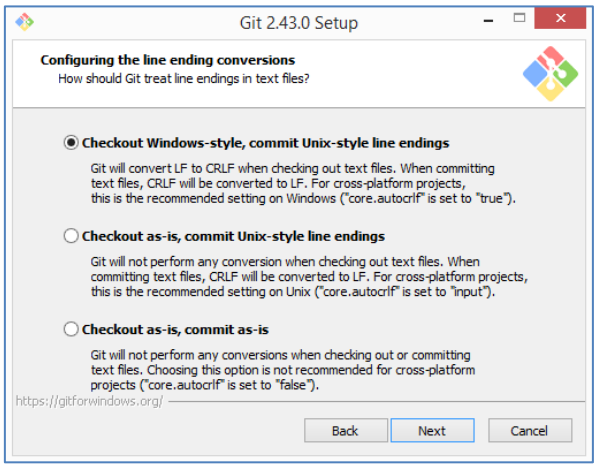

Git 2.43.0 Setup
Select Components
Which components should be installed?
Select the components you want to install; clear the components you do not want to install. Click Next when you are ready to continue.
Additional icons
On the Desktop
Windows Explorer integration
Open Git Bash here
Open Git GUI here
Git LFS (Large File Support)
Associate .git* configuration files with the default text editor
Associate .sh files to be run with Bash
Use a TrueType font in all console windows
Check daily for Git for Windows updates
Current selection requires at least 320.8 MB of disk space.
https://gitforwindows.org/
Back
Next
Cancel

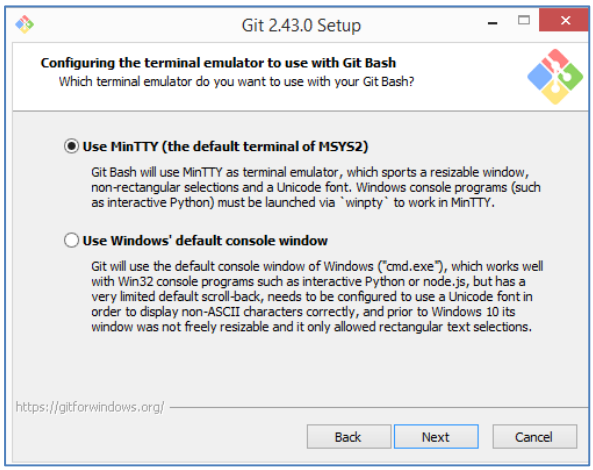

Git 2.43.0 Setup
Choosing the default editor used by Git
Which editor would you like Git to use?
Use the Nano editor by default
GNU nano is a small and friendly text editor running in the console window.
This is the recommended option for end users if no GUI editors are installed.
https://gitforwindows.org/
Back
Next
Cancel

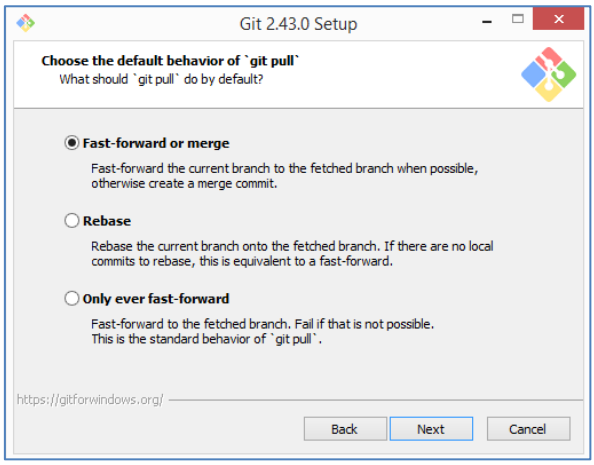

Git 2.43.0 Setup
Adjusting the name of the initial branch in new repositories
What would you like Git to name the initial branch after "git init"?
Let Git decide
Let Git use its default branch name (currently: "master") for the initial branch in newly created repositories. The Git project intends to change this default to a more inclusive name in the near future.
Override the default branch name for new repositories
NEW! Many teams already renamed their default branches; common choices are "main", "trunk" and "development". Specify the name "git init" should use for the initial branch:
main
This setting does not affect existing repositories.
https://gitforwindows.org/
Back
Next
Cancel

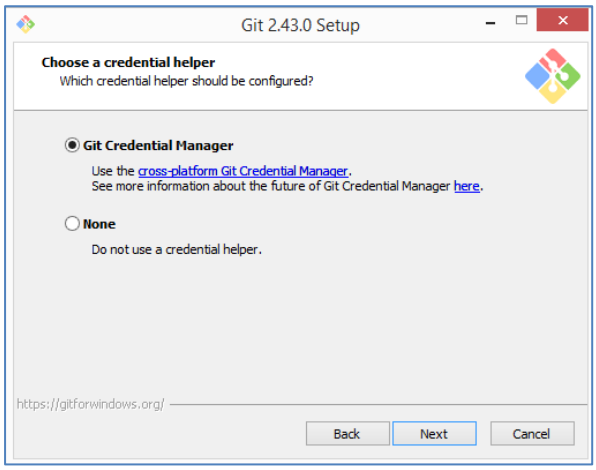

Git 2.43.0 Setup
Adjusting your PATH environment
How would you like to use Git from the command line?
Use Git from Git Bash only
This is the most cautious choice as your PATH will not be modified at all. You will only be able to use the Git command line tools from Git Bash.
Git from the command line and also from 3rd-party software
(Recommended) This option adds only some minimal Git wrappers to your PATH to avoid cluttering your environment with optional Unix tools.
You will be able to use Git from Git Bash, the Command Prompt and the Windows PowerShell as well as any third-party software looking for Git in PATH.
Use Git and optional Unix tools from the Command Prompt
Both Git and the optional Unix tools will be added to your PATH.
Warning: This will override Windows tools like "find" and "sort". Only use this option if you understand the implications.
https://gitforwindows.org/
Back
Next
Cancel

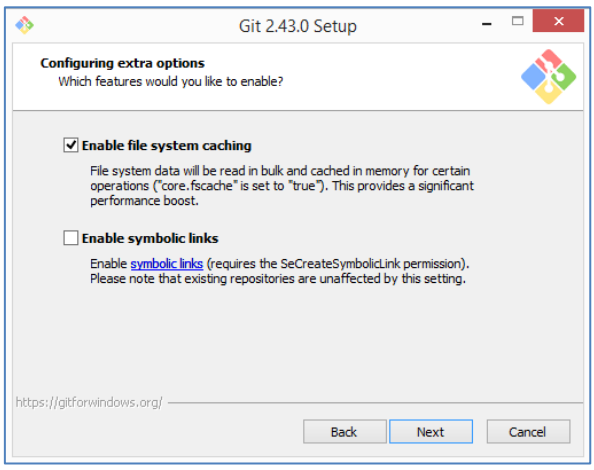

Git 2.43.0 Setup
Choosing HTTPS transport backend
Which SSL/TLS library would you like Git to use for HTTPS connections?
Use the OpenSSL library
Server certificates will be validated using the ca-bundle.crt file.
Use the native Windows Secure Channel library
Server certificates will be validated using Windows Certificate Stores.
This option also allows you to use your company's internal Root CA certificates distributed e.g. via Active Directory Domain Services.
https://gitforwindows.org/
Back
Next
Cancel

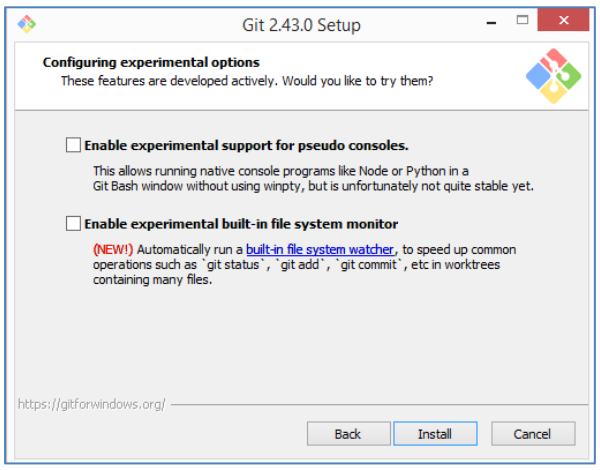

Git 2.43.0 Setup
Configuring the line ending conversions
How should Git treat line endings in text files?
Checkout Windows-style, commit Unix-style line endings
Git will convert LF to CRLF when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Windows ("core.autocrlf" is set to "true").
Checkout as-is, commit Unix-style line endings
Git will not perform any conversion when checking out text files. When committing text files, CRLF will be converted to LF. For cross-platform projects, this is the recommended setting on Unix ("core.autocrlf" is set to "input").
Checkout as-is, commit as-is
Git will not perform any conversions when checking out or committing text files. Choosing this option is not recommended for cross-platform projects ("core.autocrlf" is set to "false").
https://gitforwindows.org/
Back
Next
Cancel

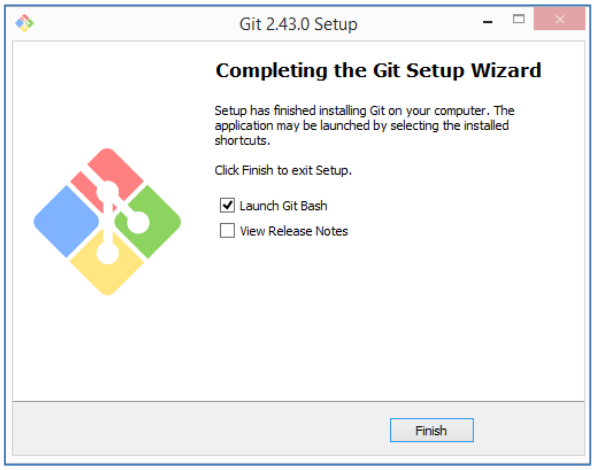

Git 2.43.0 Setup
Configuring the terminal emulator to use with Git Bash
Which terminal emulator do you want to use with your Git Bash?
Use MinTTY (the default terminal of MSYS2)
Git Bash will use MinTTY as terminal emulator, which sports a resizable window, non-rectangular selections and a Unicode font. Windows console programs (such as interactive Python) must be launched via `winpty` to work in MinTTY.
Use Windows' default console window
Git will use the default console window of Windows ("cmd.exe"), which works well with Win32 console programs such as interactive Python or node.js, but has a very limited default scroll-back, needs to be configured to use a Unicode font in order to display non-ASCII characters correctly, and prior to Windows 10 its window was not freely resizable and it only allowed rectangular text selections.
https://gitforwindows.org/
Back
Next
Cancel

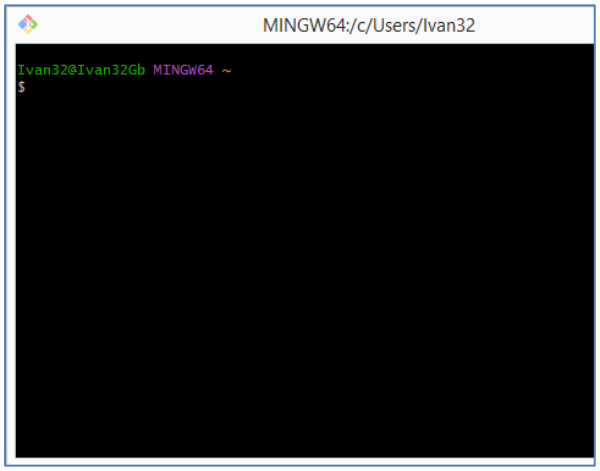












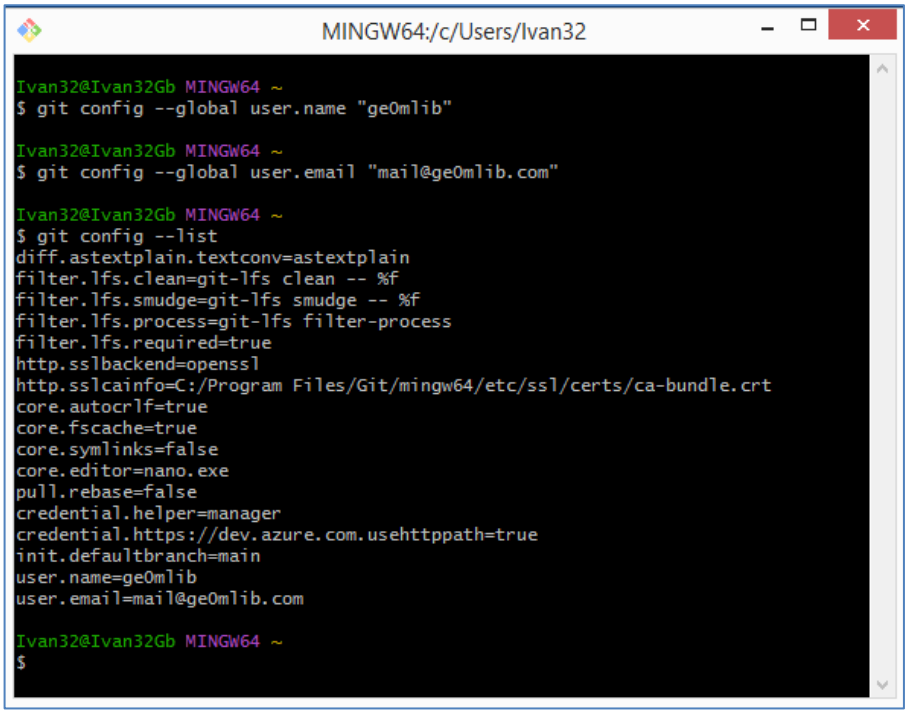
После установки, в окне git необходимо выполнить приведенные ниже команды, прописав свои user.name и user.email. Команды можно копировать и вставлять, используя правую клавишу мыши.

git config --global user.name "xxx"

git config --global user.email "xxxx@xxx.xxx"

git config --list

exit



После выполнения команды exit окно git закроется.

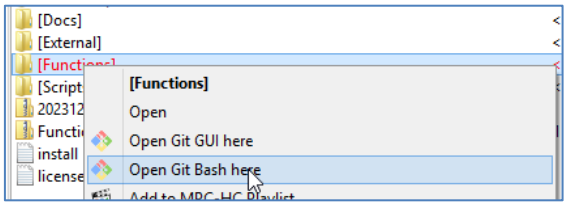
3.3 Создание git-проекта и скачивание ge0mlib из репозитория

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.



После запуска git необходимо ввести показанную ниже последовательность команд:

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

git init

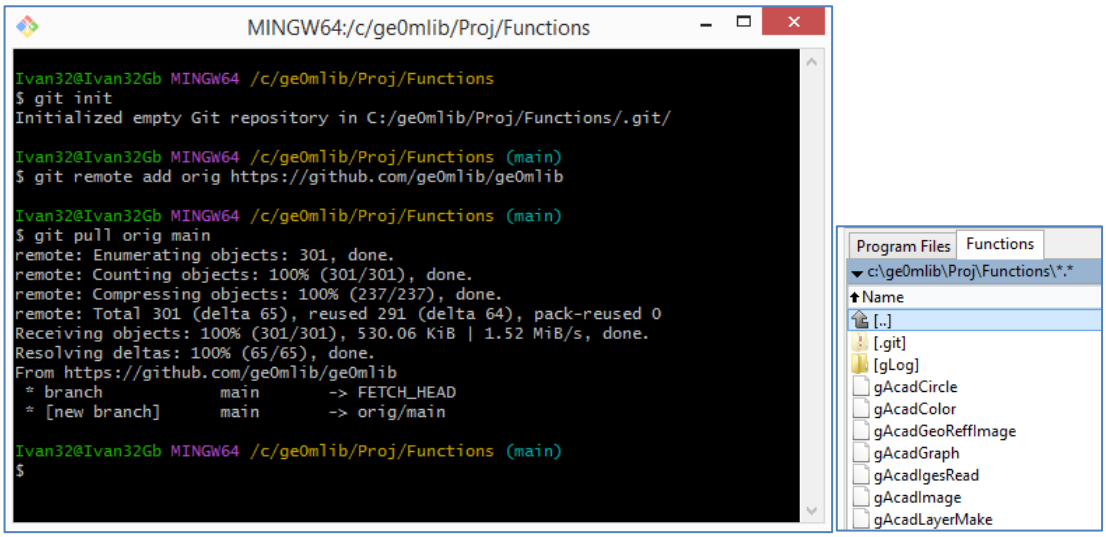
-- добавление пути к репозиторию на Git Hub

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

-- копирование файлов репозитория в папку .git

git pull orig main

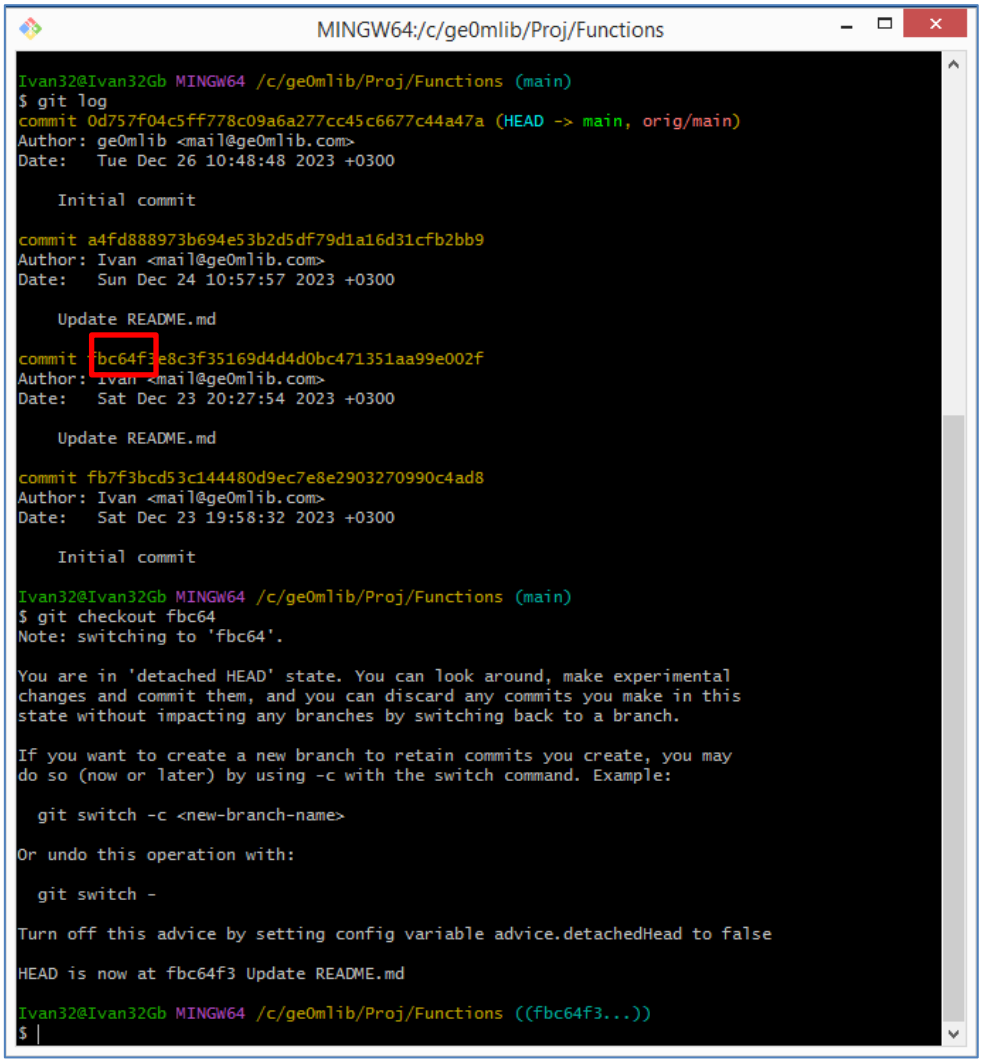
Пример окна git и содержимого папки Functions, после выполнения команд показан ниже.



3.4 Перемещение по истории в git

Для вывода списка «снимков библиотеки» (commit) необходимо использовать команду git log

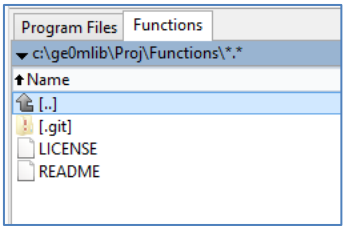
используйте клавишу “q” для выхода. При выполнении команды git выводит список «снимков» (commit) с информацией о дате и уникальным хэш-кодом (именем) для каждого commit.



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 специальный 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 ниже.



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

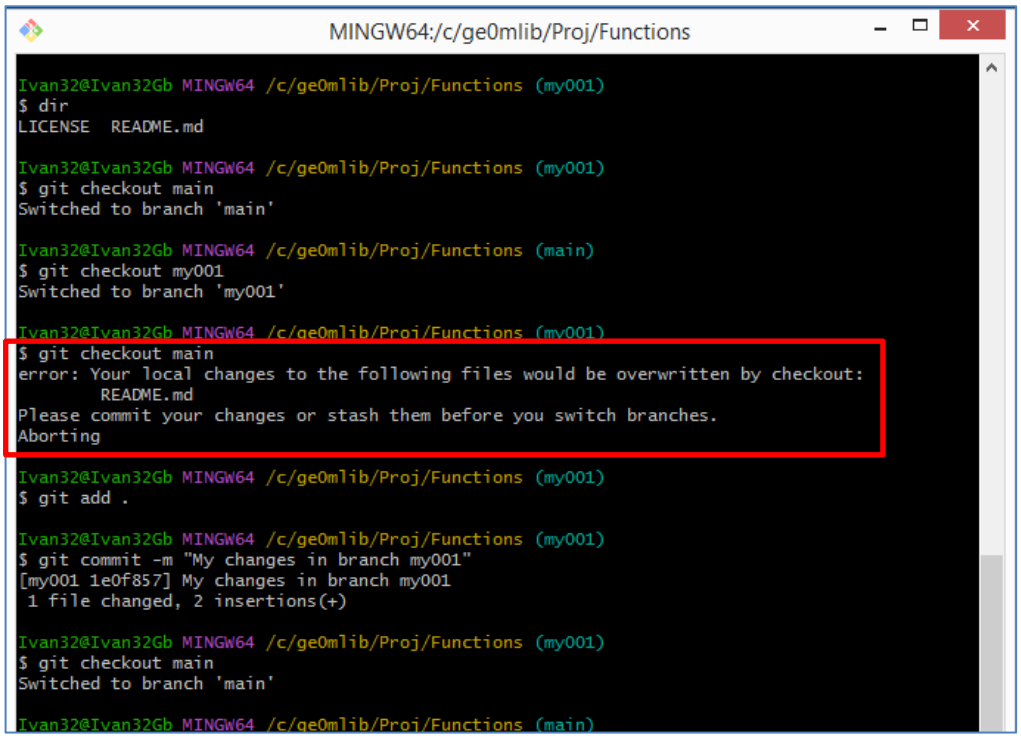
git branch my001

Теперь мы можем переключаться между «последней версией» библиотеки и «версией 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 скриншоте ниже.



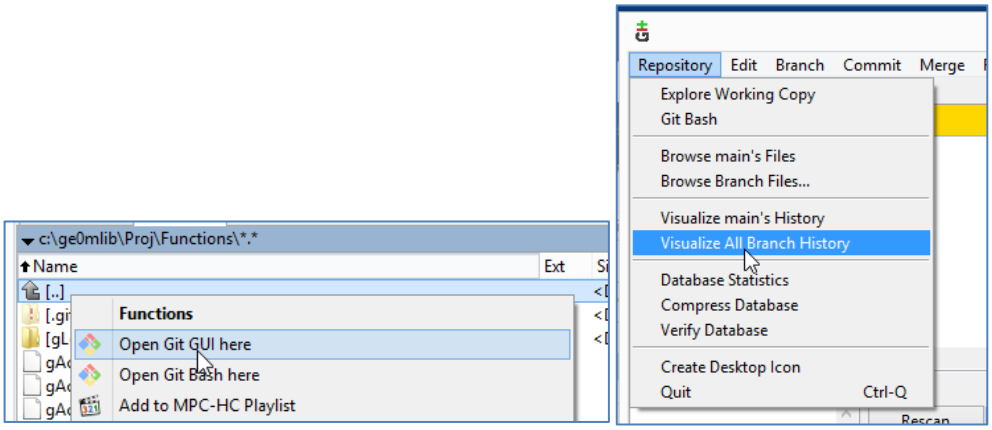
Это логично, потому что в случае переключения внесенные изменения бы «потерялись» (были стерты). Теперь, для того чтобы выполнить переключение, надо создать новый commit с нашими изменениями. Для этого, например, можно выполнить следующую последовательность команд:

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

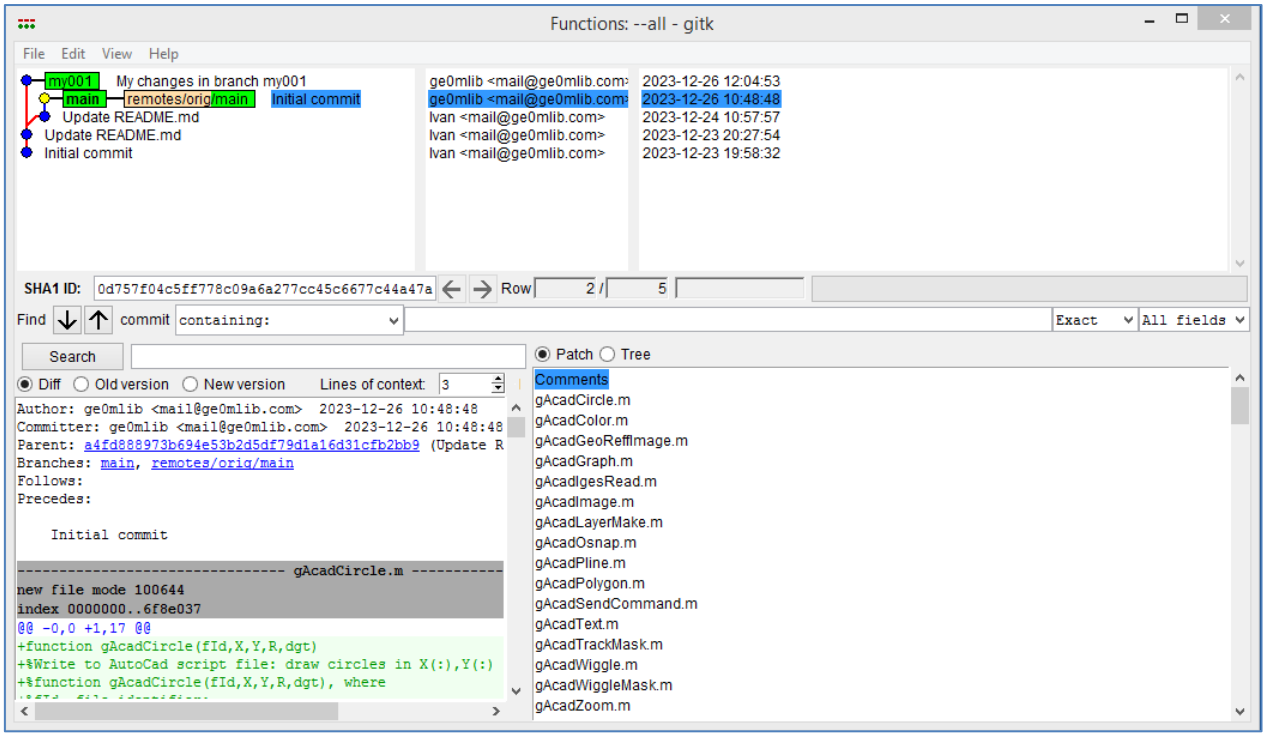
Дальнейшее «углубление в тему» требует полноценного изучения работы с git, что выходит за рамки данного документа.

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.



В открывшемся окне можно посмотреть отрисовку веток, commit и хэш-кодов для них.



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* 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 рисунке ниже (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. 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).

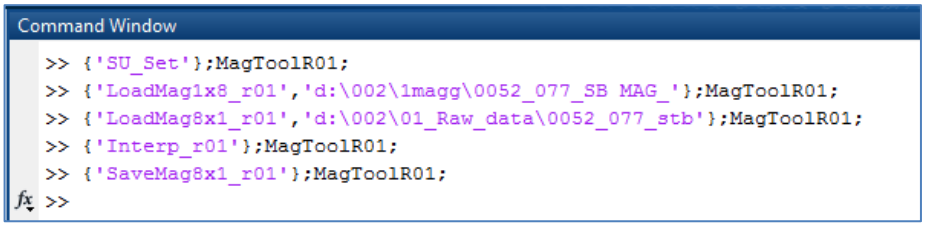
MagToolR01.m

```
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\1magg\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\1magg\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,AbsTS0,1);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%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;
```

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 рисунке ниже.



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 быструю установку программного обеспечения и перейти к обработке данных с помощью скрипта (m-файла).