Linux, Git and Matlab

Grzegorz Orzechowski

Computational Methods in Mechanics

Outline

Course Aims

Computer Methods in Mechanics (CMiM)

- Source control systems.
- Good programming practices and code efficiency.
- Vibrating systems.
- Integration of the equations of motion.
- Planar multibody systems.

Write your own multibody solver.

Coarse Course Schedule

- Linux, Git and Matlab.
- One DOF vibrating systems and numerical integration.
- Multiple DOF systems. Linear algebra.
- Planar multibody systems. Bodies, joints and forces.

Grades

Assessment may consist of the following elements:

- Programming assignments,
- also *maybe* some simple tests.

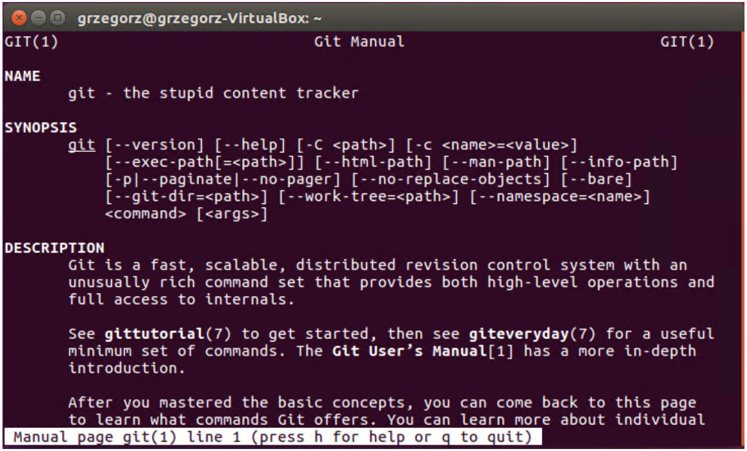
Basic Commands

# is a comment, CTRL + c – stop current command
Cheat sheet, e.g. cheatography.com/1/cs/49/

```
pwd # current directory
mkdir my_dir # create directory my_dir
cd my_dir # change directory to my_dir
cd .. # one level up
ls # list files
man command # manual for command
touch my_file # create empty my_file
cat file1 file2 # concat. files
less my_file # view and paginate my_file
cp file1 file2 # copy file1 to file2
mv file1 file2 # move file1 to file2
rm my_file # remove my_file
```

Git Basics

What is Git?



```
GIT(1)                          Git Manual                         GIT(1)

NAME
       git - the stupid content tracker

SYNOPSIS
       git [--version] [--help] [-C <path>] [-c <name>=<value>]
           [--exec-path[=<path>]] [--html-path] [--man-path] [--info-path]
           [-p|--paginate|--no-pager] [--no-replace-objects] [--bare]
           [--git-dir=<path>] [--work-tree=<path>] [--namespace=<name>]
           <command> [<args>]

DESCRIPTION
       Git is a fast, scalable, distributed revision control system with an
       unusually rich command set that provides both high-level operations and
       full access to internals.

       See gittutorial(7) to get started, then see giteveryday(7) for a useful
       minimum set of commands. The Git User's Manual[1] has a more in-depth
       introduction.

       After you mastered the basic concepts, you can come back to this page
       to learn what commands Git offers. You can learn more about individual
 Manual page git(1) line 1 (press h for help or q to quit)
```

Why to use VCS?

Version Control Systems

Management of changes to documents, computer programs, large web sites, and other collections of information.

- Collaboration.
- Storing code history.
- Track of code changes.
- Easy restoring previous versions.
- Backup.

Why to use VCS?

After si618 at stackoverflow:

Have you ever:

- Made a change to code, realized it was a mistake and wanted to revert back?
- Lost code or had a backup that was too old?
- Had to maintain multiple versions of a product?
- Wanted to see the difference between two (or more) versions of your code?
- Wanted to prove that a particular change broke or fixed a piece of code?

Why to use VCS?

Have you ever:

- Wanted to review the history of some code?
- Wanted to submit a change to someone else's code?
- Wanted to share your code, or let other people work on your code?
- Wanted to see how much work is being done, and where, when and by whom?
- Wanted to experiment with a new feature without interfering with working code?

Git Setup

```
man git
# then clone the following repository
mkdir cmim2018
cd cmim2018
git clone \
https://github.com/gorzech/lut_cmim2018.git
cd lut_cmim2018 && ls -la
git status # check repository status
```

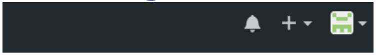
Your own repository

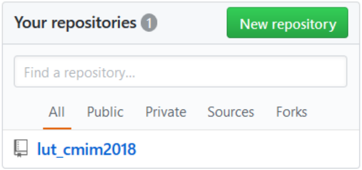
Next, we will create a new repository at github.com!

Create Repository at github.com

You can create Git repository at your PC offline, but this is not done too often.

After login

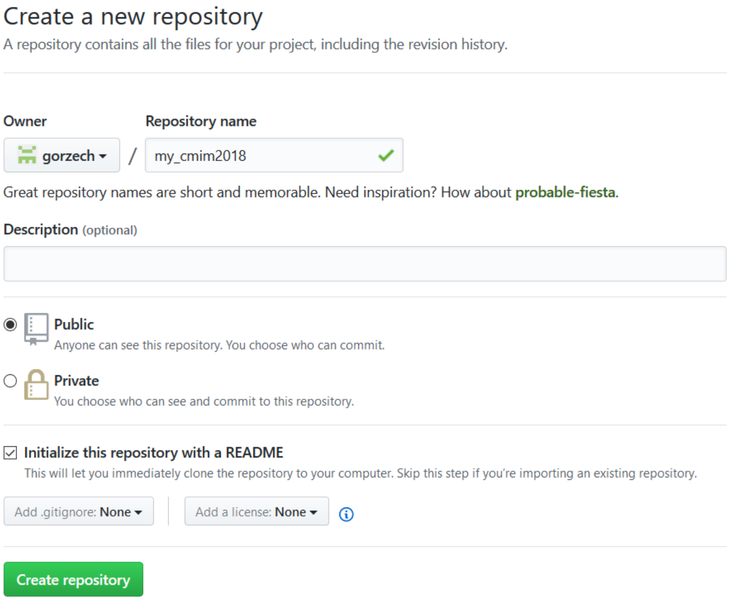




lut_cmim2018

Go to “New repository”

Go to github.com/new



Create a new repository

A repository contains all the files for your project, including the revision history.

Owner		Repository name
gorzech	/	my_cmim2018

Great repository names are short and memorable. Need inspiration? How about probable-fiesta.

Description (optional)

[● Public] Anyone can see this repository. You choose who can commit.

○ Private You choose who can see and commit to this repository.

☑ Initialize this repository with a README

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None

Add a license: None

Git and ssh

It is more convenient to clone your repository using ssh. And ssh keys.

```
cd ~/cmim2018
git clone \
git@github.com:user_name/my_cmim2018.git
# write yes, to accept
```

```
Permission denied (publickey).
```

Hmm... it seems you might not have an access to your own repo.
Now its time to fix this.

Git, ssh and ssh keys

```
ssh-keygen
# you should use password :)
cat ~/.ssh/id_rsa.pub
```

- Now go to: Github -> Settings -> SSH and GPG keys or github.com/settings/keys
- New SSH key
- Give a name and past the result of the cat command above.

Git and ssh – once again

```
cd ~/cmim2018
git clone \
git@github.com:user_name/my_cmim2018.git
# if keys are fine, no problem here
cd my_cmim2018 && ls -la
git status # check repository status
```

Now we are ready to configure git and take a full advantage of it :)

Git setup

```
man gittutorial
# now follow the tutorial: basic config
git config --global user.name "Names"
# email from github.com/settings/emails
git config --global \
user.email name@users.noreply.github.com
# to make changes copy .gitignore
cd ~/cmim2018/my_cmim2018
cp ../lut_cmim2018/.gitignore . && ls -la
# next create/edit README.md file
```

What are those files? .gitignore and REAMDE.md?

.gitignore and REAMDE.md

.gitignore

Specifies intentionally untracked files to ignore.

- Commonly used for:
  - compiled code,
  - build output directories,
  - files generated at runtime,
  - hidden system files, ...
- Each line specifies a pattern.
- Examples:
  - `*.asv`
  - `helpsearch*/`
  - `**/logs`

https://www.atlassian.com/git/tutorials/gitignore
https://github.com/github/gitignore

.gitignore and REAMDE.md

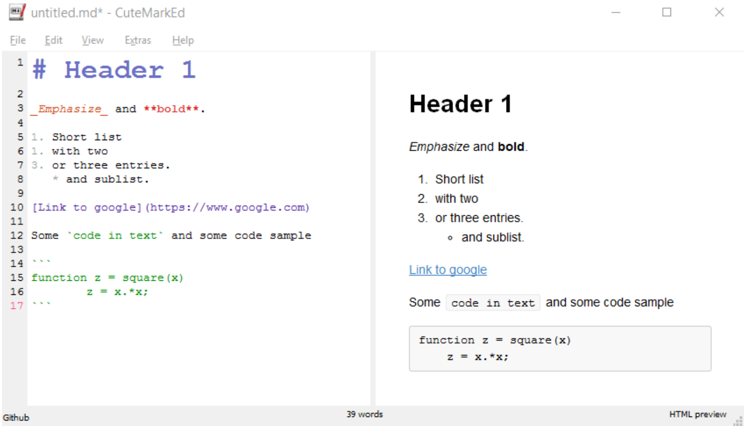
README.md

Serves to generate HTML project summary.

md (or markdown) is a lightweight markup language with plain text formatting syntax.

help.github.com/articles/basic-writing-and-formatting-syntax/
guides.github.com/features/mastering-markdown/

Markdown example



```
# Header 1

_Emphasize_ and **bold**.

1. Short list
1. with two
3. or three entries.
   * and sublist.

[Link to google](https://www.google.com)

Some `code in text` and some code sample

```
function z = square(x)
        z = x.*x;
```
```

Header 1

Emphasize and **bold**.

1. Short list
2. with two
3. or three entries.
   - and sublist.

Link to google

Some `code in text` and some code sample

```
function z = square(x)
    z = x.*x;
```

Git Basic Commands

```
# short repository summary
# loosely follow man gittutorial
git status
# add file for tracking
git add .gitignore
# add file changes
git add README.md
# ready to commit - see changes
git diff --cached
# commit changes
git commit
# push changes
git push
```

Now refresh page with your repo.

Other Git Commands

```
# viewing project history
git log
# create and manage branches
git branch experimental
# checkout branch
git checkout experimental
# make your experimental edits, commit
git commit -a
#switch back to master
git checkout master
# merge branches
git merge experimental
# if there are conflicts
git diff # to check them
# and commit merge results
git commit -a
```

Other Git Commands

```
# show graphical view of the history
gitk
# and safely delete branch
git branch -d experimental
# if you regret your branch
# delete it without mergind
git branch -D some-crazy-idea
# check branches (including active one)
git branch
# explore git for collaboration
# and check
man giteveryday
```

Git Cheat Sheet and GfW

Some helper materials

services.github.com/on-demand/downloads/github-git-cheat-sheet.pdf
rogerdudler.github.io/git-guide/

Git for Windows

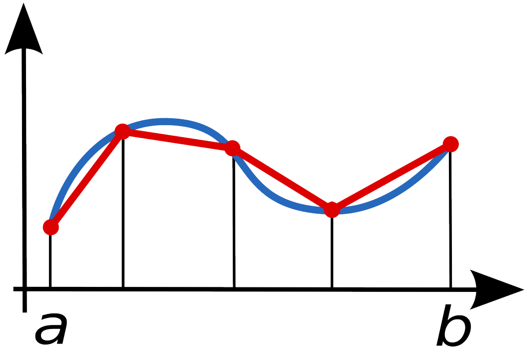
gitforwindows.org

- Can be installed on local account.
- Using `Git Bash` you can generate ssh keys.
- Have simple UI interface.

Git and binary files

If you need this, you should check this issue.

Quadrature With Trapezoidal Rule



From: wikimedia.org

Basic Composite TR

$$\int_a^b f(x)dx \approx h_1 \frac{f(a)+f(a+h_1)}{2} + \ldots + h_n \frac{f(b-h_n)+f(b)}{2}$$

Assignment

What to do and include in report?

1. Test that given program works correctly.
2. Use github.
3. Compare with build-in Matlab procedure.
4. Test how debugger works.
5. Speed it up (about 10 times).
6. Use it to compute 2D integral.

Assignment Report

- Max. 1 page A4 with font at least 10 pt.
- Ready for the next classes in PDF format.
- Be concise and specific.
- However, all important remarks have to be included.
- Uniformly formatted and properly structured.
- Uploading your code to github is preferred.
    - If not: attach code as zip file.