Course Policy

GeoComput & ML

2021-04-08 Thur

Course Structure

GeoComputation

- Linux environment

Course Structure

GeoComputation

- Linux environment
- Geo computational tools : gdal/ogr, pktools, grass, etc.

Course Structure

GeoComputation

- Linux environment
- Geo computational tools : gdal/ogr, pktools, grass, etc.

GeoModelling

- GeoMath

Course Structure

GeoComputation

- Linux environment
- Geo computational tools : gdal/ogr, pktools, grass, etc.

GeoModelling

- GeoMath
- GeoStats

Course Structure

GeoComputation

- Linux environment
- Geo computational tools : gdal/ogr, pktools, grass, etc.

$\Uparrow$

*Geo***Coding**

$\Downarrow$

GeoModelling

- GeoMath
- GeoStats

Homework

- HW available

Homework

- HW available
- HW solutions available

Homework

- HW available
- HW solutions available
- Completion at your own will

Homework

- HW available
- HW solutions available
- Completion at your own will
- No grading/comments unless under request

Course Project

- required for evaluation : pass/fail

Course Project

- required for evaluation : pass/fail

Format

1. written report : jupyer-notebook and its associated pdf
2. oral defense (30 min) : 20 min presentation + 10 min Q&A

Course Project

- required for evaluation : pass/fail

Format

1. written report : jupyer-notebook and its associated pdf
2. oral defense (30 min) : 20 min presentation + 10 min Q&A

Basic content

- project description
- data acquisition, operation and exploration
- model construction, evaluation, selection and interpretation
- final model delivery

Course Project

- required for evaluation : pass/fail

Format

1. written report : jupyer-notebook and its associated pdf
2. oral defense (30 min) : 20 min presentation + 10 min Q&A

Basic content

- project description
- data acquisition, operation and exploration
- model construction, evaluation, selection and interpretation
- final model delivery

Grading

- written report, by instructors : 60%
- oral defense, by instructors and peers : 40%

Course Project : Discussion points

- collaboration ?

Course Project : Discussion points

- collaboration ?
- grading criteria

Course Project : Discussion points

- collaboration ?
- grading criteria
- suggestions

Overarching Goals

- enhance your capacity

Overarching Goals

- enhance your capacity
- best your interests

Overarching Goals

- enhance your capacity
- best your interests
- for Ü

Overarching Goals

- enhance your capacity
- best your interests
- for Ü
- build together

Overarching Goals

- enhance your capacity
- best your interests
- for Ü
- build together

Communications

- slack channel
- additional meetings, by appointment only

Git : Course Materials

- created by *Linus Torvalds* in 2005

Git : Course Materials

- created by *Linus Torvalds* in 2005
- distributed version control : each directory as a full-fledged repo

Git : Course Materials

- created by *Linus Torvalds* in 2005
- distributed version control : each directory as a full-fledged repo
- used for changes tracking and work coordination among collaborators

Git : Course Materials

- created by *Linus Torvalds* in 2005
- distributed version control : each directory as a full-fledged repo
- used for changes tracking and work coordination among collaborators

Example

```
$ cd ~/SE_data
$ ls -a
.   ..   exercise   .git   lectures   README.md
```

Git : Course Materials

- created by *Linus Torvalds* in 2005
- distributed version control : each directory as a full-fledged repo
- used for changes tracking and work coordination among collaborators

Example

```
$ cd ~/SE_data
$ ls -a
.   ..   exercise   .git   lectures   README.md
```

Basic Practice

- only the first time

```
$ cd ; git clone https://github.com/selvaje/SE_data
# (source copy, no work inside here)

$ cp -r  ~/SE_data /media/sf_LVM_Shared/my_SE_data
# (working copy for yourself, taking notes, etc.)
```

Git : Course Materials

Basic Practice

- routine after the first time

```
$ cd ~/SE_data
$ git pull # (sync. w/ cloud)

$ rsync -hvrPt --ignore-existing  ~/SE_data/* \
  /media/sf_LVM_Shared/my_SE_data
#(sync. only new files)

$ cd /media/sf_LVM_Shared/my_SE_data # (work here)
```

- Common practice to separate source and working copies
- Important : NOT working in the source copy

Git : Full Setting

- git repo setup
- good for professional development
- easy for collaboration

Git : Full Setting

- git repo setup
- good for professional development
- easy for collaboration

Initialisation

```
$ mkdir my_Project ; cd my_Project

$ git config --global user.name "your name"
$ git config --global user.email "your email"

$ git init
Initialized empty Git repository in ...

$ ls -a
.   ..   .git
```

Git : Full Setting

Add files

```
$ touch README.md
$ git status
Untracked files:
(use "git add <file>..." to include in what will be
committed)
README.md

$ git add README.md ; git status
Changes to be committed:
(use "git rm --cached <file>..." to unstage)
new file:   README.md

$ git commit -m "added README" ; git status
nothing to commit, working tree clean
```

Git : Full Setting

Modify file contents

```
$ echo -e "Project for GeoComput&ML \n" > README.md
$ git status
(use "git add <file>..." to update what will be committed)
modified:   README.md

$ git add README.md ; git commit -m "modified README"
[master 002362a] modified README
1 file changed, 2 insertions(+)
$ git status
nothing to commit, working tree clean
```

Git : Full Setting

Modify file contents

```
$ echo -e "Project for GeoComput&ML \n" > README.md
$ git status
(use "git add <file>..." to update what will be committed)
modified:   README.md

$ git add README.md ; git commit -m "modified README"
[master 002362a] modified README
1 file changed, 2 insertions(+)
$ git status
nothing to commit, working tree clean
```

Move or remove files

```
$ git mv <old file> <new file>
$ git rm <filename>
remember to commit after mv or rm actions
```

Git : Full Setting

Link repo to GitHub

```
create a GitHub account
create a repo on GitHub
follow the instructions on the GitHub setup page

$ git remote add origin git@github.com:/your/project
$ git push -u origin master
```

Git : Full Setting

Link repo to GitHub

```
create a GitHub account
create a repo on GitHub
follow the instructions on the GitHub setup page

$ git remote add origin git@github.com:/your/project
$ git push -u origin master
```

Sync. w/ GitHub

```
$ git pull # download
$ git push # upload
```

Git : Full Setting

Link repo to GitHub

```
create a GitHub account
create a repo on GitHub
follow the instructions on the GitHub setup page

$ git remote add origin git@github.com:/your/project
$ git push -u origin master
```

Sync. w/ GitHub

```
$ git pull # download
$ git push # upload
```

ref : Git version control training

Git vs SVN

Git : Distributed version control

- no single central version of the codebase
- each working copy containing the full change history

main features

- faster committing
- each copy as a backup copy
- supporting private work

https://en.wikipedia.org/wiki/Version_control

https://en.wikipedia.org/wiki/Distributed_version_control

https://svnvsgit.com/