Daily monitoring of low-frequency earthquake activity

Ariane Ducellier, Scott Henderson

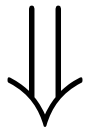
Winter 2020 Incubator project

Low-frequency earthquakes (LFEs)

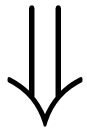
- Small magnitude (M $\sim$ 1)
- Dominant frequency low (1-10 Hz) compared with that of ordinary tiny earthquakes (up to 20 Hz)
- Source located on the plate boundary
- Grouped into families of events, with all the earthquakes of a given family originating from the same small patch on the plate interface
- Recurrence more or less episodic in a bursty manner

What we do now

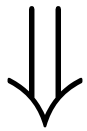
Download seismic data for a given period of time



Analyze seismic data and find time of occurrence of LFEs



Create a catalog of LFEs for this given period of time

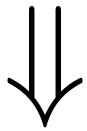


Publish catalog in some scientific journal

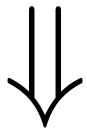
What we aim to do

LFEs occur and are recorded by permanent seismic stations every day $\rightarrow$ We could analyze seismic data and update our catalog every day

Download seismic data from two days before



Analyze seismic data and find LFEs



Update LFE catalog with new LFEs

First step: Python package

catalog

- Specific Python scripts for downloading data and finding LFEs
- Directory with templates for two LFE families

utils

General Python scripts for stacking and cross-correlation

data

- File with list of LFE families
- File with list of seismic stations
- Directory with instrument response from seismic stations

lfelib

- Specific Python scripts for downloading data and finding LFEs
- `utils` : General Python scripts for stacking and cross-correlation

tests

examples

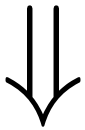
Templates, instrument responses, list of LFE families and seismic stations

.github/workflows

- environment.yml
- pyproject.toml

Second step: Command line

```
> python
>>>  get_all_responses('stations_permanent.txt')
>>>  find_LFEs('families_permanent.txt', \
    'stations_permanent.txt', 'templates', \
    (2020, 3, 7, 0, 0, 0), (2020, 3, 8, 0, 0, 0), 10.0, \
    60.0, (1.5, 9.0), 1.0, 0.05, 10, 10.0, 'MAD', 8.0)
```



```
getresp -s stations_permanent.txt

lfeall -ff families_permanent.txt -s stations_permanent.txt
-t templates -t0 $year1 $month1 $day1 0 0 0
-tf $year2 $month2 $day2 0 0 0 -td 10.0 -d 60.0
-f 1.5 9.0 -f0 1.0 -dt 0.05 -n 10 -w 10.0 -tr MAD -tv 8.0
```

Third step: GitHub workflow

68 lines (58 sloc) | 2.13 KB

Raw | Blame | History

```yaml
name: CronJob

on:
  schedule:
    - cron: '0 0 * * *' # Daily at midnight
#    - cron: '0 * * * *' #On the hour
#    - cron: '*/30 * * * *' #Every x minutes for testing

jobs:
  cronjob:
    runs-on: ubuntu-18.04
    steps:
      - name: Set Job Environment Variables
        run: |
          DATE="$( date -u -d '3 days ago' '+%Y%m%d' )"
          echo "::set-env name=DATE::${DATE}"

      - name: Checkout Repo
        uses: actions/checkout@v2

      - name: Setup Conda Environment
        uses: goanpeca/setup-miniconda@v1
        with:
           environment-file: environment.yml
           activate-environment: lfelib
           miniconda-version: 'latest'
           auto-activate-base: false
           auto-update-conda: false

      # Later change to versioned release on pypi
      - name: Install lfelib
```

Third step: GitHub workflow

Run the job daily at midnight

```
schedule:
  - cron: '0 0 * * *'
```

Get the date of three days ago

```
- name: Set Job Environment Variables
  run: |
    DATE="$( date -u -d '3 days ago' '+%Y%m%d' )"
    echo "::set-env name=DATE::${DATE}"
```

Get the conda environment to run the code

```
- name: Setup Conda Environment
  uses: goanpeca/setup-miniconda@v1
  with:
     environment-file: environment.yml
     activate-environment: lfelib
```

Install the latest version of the package

```
- name: Install lfelib
  shell: bash -l {0}
  run: |
    pip install --extra-index-url https://test.pypi.org/simple/ lfelib
```

Third step: GitHub workflow

Run the script downloading the data and looking for LFEs

```
- name: Run Daily Processing
  shell: bash -l {0}
  run: |
    cd examples
    ./cronscript.sh
```

Upload the output file

```
- name: Upload Zipped LFEs Folder with Results
  uses: actions/upload-artifact@v1
  with:
    name: ${{env.DATE}}
    path: ./examples/LFEs/
```

Save the output file on Google Drive

```
- name: Upload Results CSVs to Google Drive
  uses: wei/rclone@v1
  env:
    RCLONE_CONFIG_INCUBATOR_TOKEN: ${{ secrets.RCLONE_CONFIG_INCUBATOR_TOKEN }}
    RCLONE_CONFIG_INCUBATOR_TEAM_DRIVE: ${{secrets.RCLONE_CONFIG_INCUBATOR_TEAM_DRIVE }}
     RCLONE_CONFIG_INCUBATOR_TYPE: drive
    RCLONE_CONFIG_INCUBATOR_SCOPE: drive
  with:
    args: copy ./examples/LFEs/ incubator:lfelib/cronjob
```

Fourth step: Improving memory space and computing time

Memory space

Store instrument response into data directory

Memory space

Download instrument response before looking for LFEs

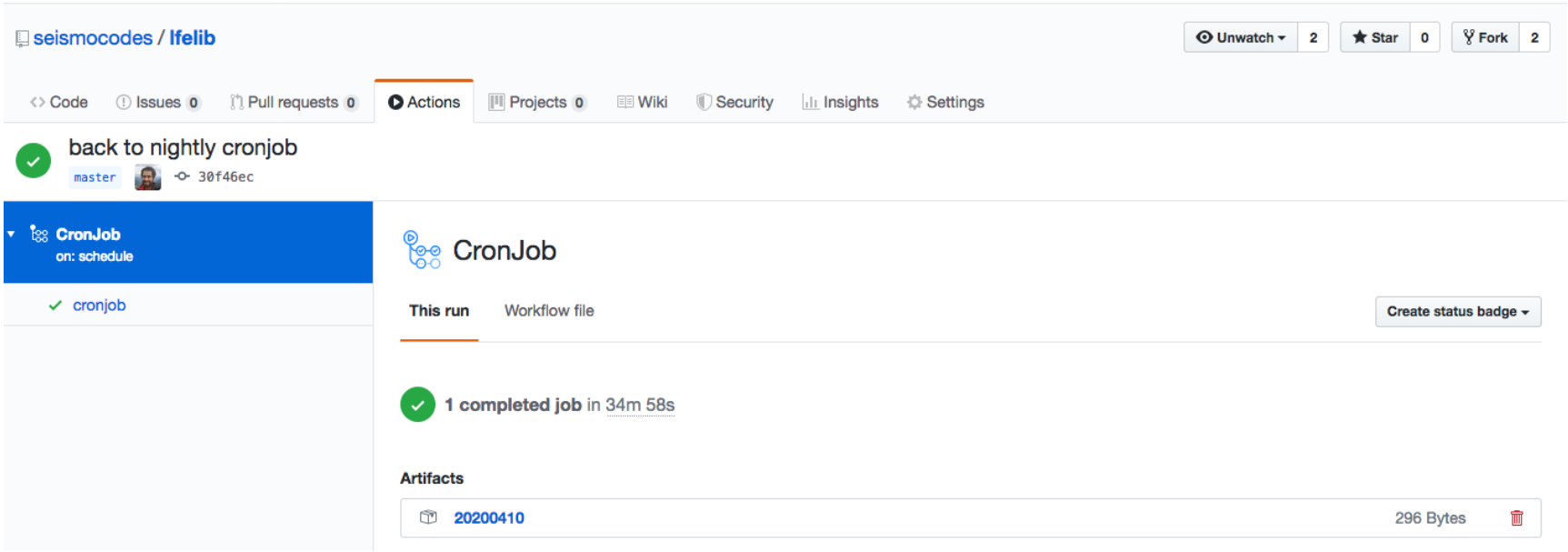
Computing time

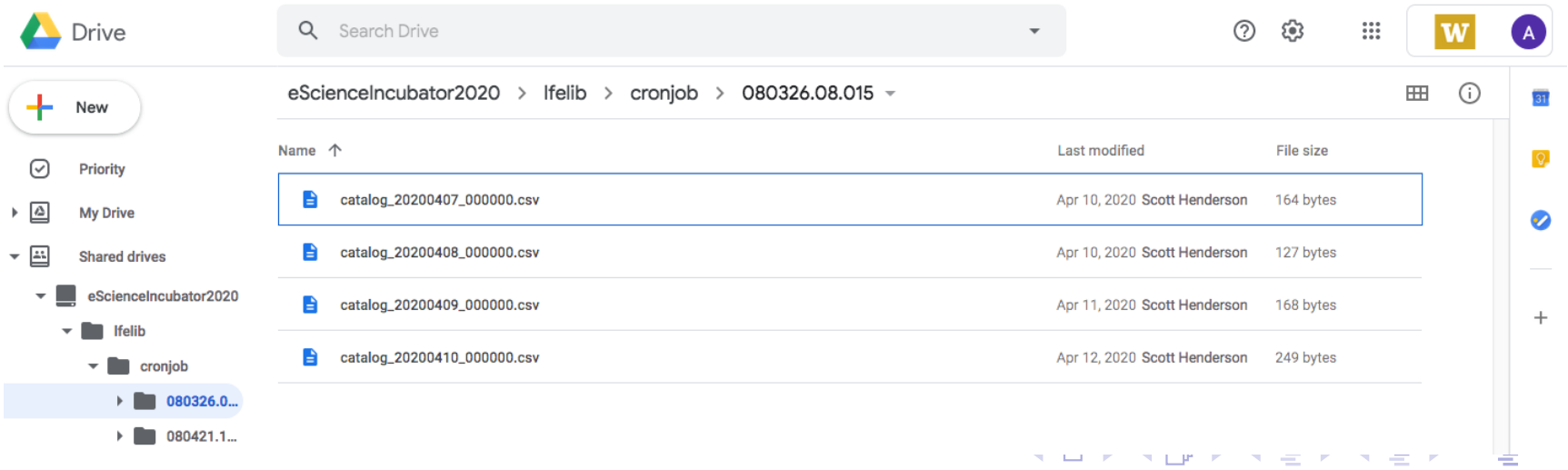
Loop on LFE families
    Loop on seismic stations
        Download seismic data
    Analyze seismic data
    Delete seismic data

Computing time

Loop on seismic stations
    Download seismic data
Loop on LFE families
    Analyze seismic data
Delete seismic data

Fifth step: Saving results





Future improvements

Computing time

Download data one station at a time
Look for new LFEs one LFE family at a time

⇨

Computing time

Parallelization of Python scripts

Volume of data

Two LFE families
Runs on GitHub in 35 minutes
Time limit on the computing time of the jobs you can run on GitHub

⇨

Volume of data

May look at Amazon Lambda instead

Thank you