Open Neurofeedback Training

Installation Manual

2017-03-14

Yury Koush, John Ashburner, Evgeny Prilepin, Ronald Sladky, Peter Zeidman, Sergei Bibikov, Frank Scharnowski, Artem Nikonorov, Dimitri Van De Ville

opennft.org

Table of Contents

Setup Instructions

Microsoft Windows

Windows	Matlab	Python
7, 8, 10 (32-bit)	2015b (32-bit)	3.4.* (32-bit)
7, 8, 10 (64-bit)	2015b or 2016a	3.4.*
	2016b (recommended)	3.5.* (recommended)

Table 1. Compatibility list for Windows.

All systems

1. Install **Mathworks Matlab** 2015b or higher (2016b recommended)
2. Install a **Python 3** version that is compatible with your Matlab version (see table)

   Download: https://www.python.org/downloads/windows/
3. For convenience, add Python paths (e.g. `c:\Python34\;c:\Python34\Scripts;`) to your system Path environment variable: *System* ▸ *Advanced* ▸ *Environment Variables…*
4. Install **Git SCM**

   Download: https://git-scm.com/downloads
5. Install required **MATLAB toolboxes**
   1. SPM12: http://www.fil.ion.ucl.ac.uk/spm
   2. jsonlab Toolbox: https://www.mathworks.com/matlabcentral/fileexchange/33381-jsonlab--a-toolbox-to-encode-decode-json-files
   3. PsychToolbox: http://psychtoolbox.org/download/

For Python 3.5 setups (with Python virtual environment support)

1. Setup Python 3.5 virtual environments using Windows Command Prompt in your *workspace* directory (i.e., the directory with your neurofeedback projects)

   ```
   cd c:\OpenNFT

   c:\Python35\python -m venv OpenNFT_venv
   ```

   We use *OpenNFT_venv* as convenient name for our virtual environment but you can use any name. If there are multiple python versions installed, you can use the command `python3.5` instead of `python` to ensure that version 3.5 is used when launching Python.

   Activate the new virtual environment by using the corresponding activate script

   ```
   OpenNFT_venv\Scripts\activate.bat
   ```

Use deactivate.bat to `deactivitate` your active virtual environment.

2. Setup MATLAB Engine API for Python using

```
cd "c:\Program Files\MATLAB\R2016b\extern\engines\python"
python setup.py build --build-
base="c:\OpenNFT\OpenNFT_venv\Lib\site-
packages\MatlabEngineBuild" install
```

   If multiple Matlab installations are present on your system's PATH ensure that the Matlab version that will be used in Python is preferred (i.e., preceding the other versions in the variable definition).

3. Install the required Python packages using the command line tool pip:

```
pip install -U pip setuptools
pip install PyQt5
pip install watchdog
pip install --no-deps
git+https://github.com/pyqtgraph/pyqtgraph.git
```

4. Install numpy+MKL (i.e., an optimized numpy version) using a compatible wheel file from, e.g., http://www.silx.org/pub/wheelhouse/

```
pip install http://www.silx.org/pub/wheelhouse/numpy-
1.11.3+mkl-cp34-cp34m-win32.whl
```

For Python 3.4 setups (without Python virtual environment support)

1. Setup MATLAB Engine API for Python using Windows Command Prompt (use right click and select *Run as Administrator* or alter access permissions of MATLAB engine folder):

```
cd "c:\Program Files\MATLAB\R2015b\extern\engines\python"
python setup.py install
```

   MATLAB Python engine will be installed to your currently active Python environment (e.g., `c:\Python34\Lib\site-packages\matlab\`). In case there are multiple versions installed, ensure that the correct MATLAB and Python versions are installed.

2. Install the required Python packages using the command line tool pip:

```
python -m pip install -U pip
pip install watchdog
pip install --no-deps
git+https://github.com/pyqtgraph/pyqtgraph.git
```

3. Install numpy+MKL using a compatible wheel file from e.g., http://www.silx.org/pub/wheelhouse/

```
pip install http://www.silx.org/pub/wheelhouse/numpy-
1.11.3+mkl-cp34-cp34m-win32.whl
```

4. Install PyQt5 from https://sourceforge.net/projects/pyqt/files/PyQt5/PyQt-5.5.1/PyQt5-5.5.1-gpl-Py3.4-Qt5.5.1-x32.exe/download

Apple macOS

macOS	Matlab	Python
El Capitan	2015b, 2016a	3.4.*
El Capitan, Sierra	2016b	3.5.*

Table 2. Compatibility list for macOS.

All systems

1. Install **Mathworks Matlab** 2015b or higher (2016 recommended)
2. Install a **Python 3** version that is compatible with your Matlab version (see table)

   Download: https://www.python.org/downloads/macos
3. Install **Git SCM**

   Download: https://git-scm.com/download/mac
4. Install required **MATLAB toolboxes**
   1. SPM12: http://www.fil.ion.ucl.ac.uk/spm
   2. jsonlab Toolbox: https://www.mathworks.com/matlabcentral/fileexchange/33381-jsonlab--a-toolbox-to-encode-decode-json-files
   3. PsychToolbox: http://psychtoolbox.org/download/

For Python 3.5 setups (with Python virtual environment support)

1. In Terminal go to your *workspace* directory (i.e., the directory with your neurofeedback projects) and create a Python virtual environment using

   ```
   cd ~/Documents/Work/OpenNFT
   python -m venv OpenNFT_venv
   ```

   We use *OpenNFT_venv* as convenient name for our virtual environment but you can use any name. If there are multiple python versions installed, you can use the command `python3.5` instead of `python` to ensure that version 3.5 is used when launching Python.

   Activate the new virtual environment by using the corresponding activate script

   ```
   source OpenNFT_venv/bin/activate
   ```

   The new active virtual environment is now active in the current terminal window, indicated by, e.g., `(OpenNFT_venv) MacBook:python nfbuser$`

2. Setup MATLAB Engine API for Python using macOS Terminal:

```
cd /Applications/MATLAB_R2016b.app/extern/engines/python/
python setup.py build install
```

MATLAB Python engine will be installed to your currently active Python virtual environment.

3. Install required Python packages using the command line tool pip:

```
cd ~/Documents/Work/OpenNFT/OpenNFT_venv/bin/
./python -m pip install -U pip
./python pip install watchdog
./python pip install --no-deps
git+https://github.com/pyqtgraph/pyqtgraph.git
./python pip install numpy==1.11.0
```

Please note that the optimized numpy-MKL implementation would be ideal but is currently not (easily) available for macOS.

For Python 3.4 setups (without Python virtual environment support)

1. Setup MATLAB Engine API for Python using macOS Terminal:

```
cd /Applications/MATLAB_R2016a.app/extern/engines/python/
python setup.py install
```

MATLAB Python engine will be installed to your currently active Python environment (e.g., `/Library/Frameworks/Python.framework/Versions/3.4/lib/python3.4/site-packages/`). In case there are multiple versions installed, ensure that the correct MATLAB and Python versions are installed.

2. Install required Python packages using the command line tool pip:

```
python -m pip install -U pip
pip install watchdog
pip install --no-deps
git+https://github.com/pyqtgraph/pyqtgraph.git
pin install numpy==1.11.0
```

Please note that the optimized numpy-MKL implementation would be ideal but is currently not (easily) available for macOS.

3. Install Qt5 using the package manager brew (https://brew.sh/):

```
brew install qt5
```

4. Download PyQt5 from https://pypi.python.org/pypi/pyqt5-macos-built, extract the archive and install the package using Terminal:

```
cd ~/Downloads/dist/pyqt5-macos-built-5.5.0
python setup.py install
```

Troubleshooting

Caveats

Configuration files (ini files) as operating system dependent

All settings in the ini files, for example and most importantly the path definitions follow the conventions of your host operating system. E.g., use `\` as file separator in Windows and `/` in a Unix-based system such as macOS.

DCM-based neurofeedback is based on DCM10

The currently implemented version of DCM neurofeedback is based on Koush et al., 2013 and 2015 publications where DCM10 as implemented in SPM8 (`spm_dcm_estimate_rt()`, `spm_nlsi_GN_rt()`) was used. Other versions of DCM use different estimation methods and might fail to reproduce the same results and would require an additional testing.

SPM preprocessing is optimized for real-time applications

Note the differences between the real-time modifications of the SPM12 inbuilt preprocessing functions (`spm_realign_rt()`, `spm_reslice_rt()`) and their SPM12 analogues. Applied modifications have a sufficient quality level for real-time applications, but they are not necessarily matching your local SPM setup.

Spatial orientation of input data

Generally, you are advised to carefully check the spatial orientation of all the images provided to the software. Our software is independent of data spatial orientation, which is often a function of acquisition parameters. Unfortunately, most Phillips MR scanner setups do not provide adequate image header information when performing a real-time export. Additionally, Phillips real-time data may require a 180-degree flip to match the proper EPI template.

Optimize signal processing settings

The configuration of the optimal signal processing settings depends on the experimental design and acquisition parameters. E.g., see Koush et al. 2012 for the setup of a Kalman filter.

Runtime errors and troubleshooting

Undefined function 'spm_select' for input arguments of type 'char'.

Make sure that SPM is installed and MATLAB is able to locate it in your system path. To test if the correct SPM version is found use `which spm` in a MATLAB window.

Undefined function or variable 'bwperim'

Make sure that you have installed MATLAB's Image Processing toolbox.

Undefined function or variable 'zscore'.

Make sure that you have installed MATLAB's Statistics and Machine Learning toolbox.

Error when loading DICOM files.

There is a known bug in the current implementation of MATLAB's dicominfo.m. We used the following modifications to fix the problem:

Line 336

`personName = struct([]);` changed to
`personName = repmat(makePerson(pnParts),[1,numel(splitRawData)]);`

Line 353

`%personName = makePerson(pnParts);` changed to
`personName(p) = makePerson(pnParts);`

Error when starting Matlab processes on macOS.

We observed problems when starting Matlab instances on macOS from within OpenNFT, either during startup or using the *Initialize* button. The easiest way to fix this problem is to independently start and share the required Matlab instances (main, PTB, and SPM instances) using the macOS command line:

```
/Applications/MATLAB_R2016b.app/bin/matlab -desktop -r
"matlab.engine.shareEngine('MATLAB_NFB_00001')"

/Applications/MATLAB_R2016b.app/bin/matlab -desktop -r
"matlab.engine.shareEngine('MATLAB_NFB_PTB_00001')"

/Applications/MATLAB_R2016b.app/bin/matlab -desktop -r
"matlab.engine.shareEngine('MATLAB_NFB_SPM_00001')"
```