Install' instructions

2020-11-05

Contents

1 mlpack install from sources

1.1 Installation de armadillo

armadillo est une librairie C++ pour l'algèbre linéaire. Les dépendances pour armadillo sont:

- cmake (armadillo accepte la version 2 mais la version 3 sera nécessaire ensuite (voir 1.3))
- blas et lapack. Sur Ubuntu, il faut installer les paquets

```
sudo apt-get install libblas3 libblas-dev
   libatlas-base-dev
```

- sous Ubuntu, les paquets libhdf5-cpp-100 (libhdf5-7 sous Ubuntu 14), libhdf5-dev

Le numéro de version d'armadillo n'est pas essentiel (prendre une version assez récente quand même)

Faire dans le répertoire où on veut l'installer (utiliser sudo éventuellement)

```
wget
    http://sourceforge.net/projects/arma/files/armadillo-9.700.2.tar.xz
xz -d armadillo-9.700.2.tar.xz
tar xvf armadillo-9.700.2.tar
cd armadillo-9.700.2/
cmake .
make
make install DESTDIR=<install_dir>/armadillo-9.700.2
```

Pour vérifier (adapter à l'architecture)

```
ls -l
    <install_dir>/armadillo-9.700.2/usr/lib/x86_64-linux-gnu/*.so
```

1.2 Installation de boost

boost est une dépendance pour mlpack (voir ce qui suit pour mlpack). Idéalement ce serait mieux d'avoir le compilateur mpic++.

Faire dans le répertoire où on veut l'installer

```
wget
    http://sourceforge.net/projects/boost/files/boost/1.72.0/boost_1_72_0.tar.bz2
bunzip2 boost_1_72_0.tar.bz2
tar xvf boost_1_72_0.tar
cd boost_1_72_0
./bootstrap.sh
```

Si on a mpic++, éventuellement éditer le fichier project-config.jam, qui vient d'être créé, en ajoutant à la fin

```
using mpi ;
```

Terminer (utiliser sudo éventuellement)

```
./b2 install --prefix=<install_dir>/boost-1.72.0
```

Vérifier en listant

```
ls -l <install_dir>/boost-1.72.0
```

1.3 Installation de cmake (version ≥ 3.6)

Pour installer mlpack, il faut une version de cmake ≥ 3.6. Sur Ubuntu 14, il faut que le paquet Ubuntu *libssl-dev* soit installé.

```
wget
    https://github.com/Kitware/CMake/releases/download/v3.17.0/cmake-3.17.0.tar.gz
tar xvzf cmake-3.17.0.tar.gz
cd cmake-3.17.0/
```

```
./bootstrap --prefix=<install_dir>/cmake-3.17.0
make
make install
```

1.4 Installation de mlpack

Les dépendances pour mlpack sont:

- armadillo (déjà installé)
- boost (voir avant)
- cmake (version supérieure à 3.6, voir 1.3)

Faire ce qui suit, (attention aux paths et aux numéros de version - utiliser sudo éventuellement au *make install*). Utiliser une version récente de cmake (voir 1.3). **Pour Ubuntu 14, c'est la version 3.0.4 de mlpack qu'il faut utiliser au lieu de la version 3.2.1**

```
wget http://www.mlpack.org/files/mlpack-3.2.1.tar.gz
tar xvzf mlpack-3.2.1.tar.gz
mkdir mlpack-3.2.1/build && cd mlpack-3.2.1/build
cmake
    -D"ARMADILLO_INCLUDE_DIR":type="<install_dir>/armadillo-9.700.2/usr/include/"
    -D"ARMADILLO_LIBRARY":type="<install_dir>/armadillo-9.700.2/lib/x86_64-linux-gnu/libarmadillo.so"
    -D"BOOST_ROOT":type="<install_dir>/boost-1.72.0" ..
make
make install DESTDIR=<install_dir>/mlpack
```

2 mlpack install using conda

Assuming *conda* is already installed in the machine, we can use *conda* to install the required dependencies for *mlpack*: *boost*, *armadillo*.
The first step is to create a *conda* environment (called here *reconstruction*). Inside the environment, latest *mlpack* version (3.4.2) is installed (*boost* and *armadillo* are automatically installed)

```
conda create -n reconstruction -c conda-forge mlpack
```

Then we need to activate the environment

```
conda activate reconstruction
```

Sometimes it is necessary to install a specific g++ version inside the environment like this

```
conda install -c conda-forge gxx_linux-64
```

Using this version seems to remove some problems as noticed on a CentOS 7 server using devtoolset.

3 To compile the code

Once downloaded the files, you have to edit the *Makefile*: check the compiler and modify the path of the libraries.
You should check the following variables in the Makefile which depend on the type of installation you chose (see either 1 or 2).

- ARMA = root path of armadillo
- ARMA_LIB = path of armadillo library
- BOOST = root path of boost
- BOOST_INC = path of boost include folder
- BOOST_LIB = path of boost library
- MLPACK = root path of mlpack
- MLPACK_LIB = path of mlpack library

If you use *conda* for *reconstruction* environment, you should have

```
<conda_path>=/home/toto/.conda/envs  # for example
ARMA=BOOST=MLPACK=<conda_path>/reconstruction
ARMA_LIB=BOOST_LIB=MLPACK_LIB=<conda_path>/reconstruction/lib
BOOST_INC=<conda_path>/reconstruction/include
```

Once *make* command has been run, executable file calls *bin/main.exe*. Object files *.o are under folder OBJ.

4 Parameter file

In the archive, there are two examples of parameter file

- parameters.dat (full computation: matrices+estimation for 20 neurons)
- parameters_estim.dat (only estimation, for 20 neurons, matrix computation should have been done before).

Listing 1 is an example of parameter file.

```
gamma = 0.02              # gamma parameter for estimation
Z = 0.15                  # pointless for the moment
del = 5.0                 # delta, bin length, (here in
    milliseconds, to be coherent with units)
Tmin = 0.                 # Tmin (in milliseconds here)
Tmax = 280000.            # Tmax (in milliseconds here)
K = 10                    # K = number of bins or segments
M = 10                    # number of neurons
computation_type = full # type of computation, either: full
    (matrices + estimation), matrices (only matrices),
    estimation (only estimation), error (only error)
input_dir =
    results_event2_n_weight=-1.4_run=23_tstop=280000_K=10_del=5_gamma=0.02
     # input folder, needed only for estimation and error
input_type = EventFile  #  input type of the .csv file
    (EventFile = SPIKES format)
debneur = 0               # index for start of the numbering
file_format = ascii       # format of the .csv file (only
    ascii is possible)
file_name = event2_n_weight=-1.4_run=23_tstop=360000.csv #
    name of the input file, should be in the currrent folder
lambda1 = 1.              # lambda1 for Lasso computation
    (always 1)
lambda2 = 0.              # lambda2 for Lasso computation
    (always 0)
output_dir =
    results_event2_n_weight=-1.4_run=23_tstop=280000_K=10_del=5_gamma=0.02
     # folder to store matrices, vectors and estimators
saving_matrices = true  # flag to save matrices or not
units = milliseconds    # units of the data (either seconds
    or milliseconds)
network_file = connections_setup3_10.csv  # network file to
    compute errors, could be missing
background_int_file = empty_file   # background intensity
    file to compute errors, could be missing
matrix_computation_type = Cyrille  # matrix computation
    type; either Cyrille or Gilles
```

Listing 1: Example of parameter file