Makefile

August 30, 2018

1 Main program written in C++

Practical Makefiles, by example

```
In [ ]: %%file main.cpp
        void operations();

        int main(int argc, const char **argv) {

            operations();

            return 0;
        }
```

2 Function that uses Armadillo also in C++

```
In [ ]: %%file obj_armadillo.cpp
        #include <iostream>
        #include <armadillo>

        void operations() {
            // Initialize the random generator
            arma::arma_rng::set_seed_random();

            // Create a 4x4 random matrix and print it on the screen
            arma::Mat<double> A = arma::randu(4,4);
            std::cout << "A:\n" << A << "\n";

            // Multiply A with his transpose:
            std::cout << "A * A.t() =\n";
            std::cout << A * A.t() << "\n";

            // Access/Modify rows and columns from the array:
            A.row(0) = A.row(1) + A.row(3);
            A.col(3).zeros();
            std::cout << "add rows 1 and 3, store result in row 0, also fill 4th column with ze
            std::cout << "A:\n" << A << "\n";
```

```cpp
    // Create a new diagonal matrix using the main diagonal of A:
    arma::Mat<double>B = arma::diagmat(A);
    std::cout << "B:\n" << B << "\n";

    // Save matrices A and B:
    A.save("A_mat.txt", arma::arma_ascii);
    B.save("B_mat.txt", arma::arma_ascii);

}
```

3 Build executable exe_armadillo with command line

```bash
In [ ]: %%bash
    g++ -c obj_armadillo.cpp -I/usr/local/include # create the file obj_armadillo.o
    g++ -o exe_armadillo main.cpp obj_armadillo.o -L/usr/local/lib -larmadillo # create th
```

- `-I` option flag to set headers file directory (`.h`) needed to build every object files (CXXFLAGS).
- `-L` option flag to set library directory (`.a`) needed only to build executable (LDFLAGS)

4 Test the program

```bash
In [ ]: %%bash
    ./exe_armadillo
    rm ./exe_armadillo
```

5 First Makefile

```makefile
In [ ]: %%file Makefile

    default: exe_armadillo

    obj_armadillo.o: obj_armadillo.cpp
            g++ -o $@ -c $^ -I/usr/local/include

    exe_armadillo: main.cpp obj_armadillo.o
            g++ -o $@ $^ -L/usr/local/lib -larmadillo

    clean:
            rm *.txt *.o exe_armadillo
```

6 Test the program

```bash
In [ ]: %%bash
    make
```

```
./exe_armadillo
make clean
```

7 Using variables

In [ ]: %%**file** Makefile

```
default: exe_armadillo # default target

CXX                 := g++    # := means "change CXX only if it is undefined"
ARMADILLO_INCDIR   = /usr/local/include
ARMADILLO_LIBDIR   = /usr/local/lib
ARMADILLO_INC_FLAGS = -I${ARMADILLO_INCDIR}
ARMADILLO_LIB_FLAGS = -L${ARMADILLO_LIBDIR} -larmadillo

obj_armadillo.o: obj_armadillo.cpp
        ${CXX} -c -o $@ $^ ${ARMADILLO_INC_FLAGS} # $@ : target $^: dependencies

exe_armadillo: main.cpp obj_armadillo.o
        ${CXX} -o $@ $^ ${ARMADILLO_LIB_FLAGS}

clean:
        rm *.txt *.o exe_armadillo
```

In [ ]: %%bash
```
make
./exe_armadillo
make clean
```

8 Add rule to build object files

In [ ]: %%**file** Makefile

```
default: exe_armadillo

CXX                 := g++
ARMADILLO_INCDIR   = /usr/local/include
ARMADILLO_LIBDIR   = /usr/local/lib
ARMADILLO_INC_FLAGS = -I${ARMADILLO_INCDIR}
ARMADILLO_LIB_FLAGS = -L${ARMADILLO_LIBDIR} -larmadillo

CXXFLAGS += ${ARMADILLO_INC_FLAGS}
LDFLAGS  += ${ARMADILLO_LIB_FLAGS}

SRCS = $(wildcard *.cpp)  # List all cpp files
OBJS = $(SRCS:.cpp=.o)    # List all matching object files
```

```
.cpp.o: # $< the source file .cpp $@ the target .o
        $(CXX) $(CXXFLAGS) -c $< -o $@

exe_armadillo: $(OBJS)
        ${CXX} -o $@ $^ $(CXXFLAGS) ${LDFLAGS}

clean:
        rm *.txt *.o exe_armadillo
```

In [ ]:
```
%%bash
make
./exe_armadillo
make clean
```