Contents

C++ Libraries for Stein Point Calculations

At this early stage I think I will use

- `optim` for optimisation (from Keith O'Hara)
- `autodiff` for automatic differentiation (from Allan Leal)

Both librarires were downloaded via github as

```
$ git clone https://github.com/autodiff/autodiff.git
$ git clone https://github.com/kthohr/optim.git
```

into my usual library location to my usual library install directory at `/Users/patricknoble/Documents/Library/auto`

Automatic Differentiation: autodiff

To use `autodiff` start at the webpage which is very well documented. It seems this code can be built (follow the `cmake` instructions at the webpage above) or used as a header only library. This is *very* attractive to me. I can compile and run a test example off the webpage very easily. See `stein/notes/learning/cpplibs/test-auto1.C` which is reproduced below

```
#include <iostream>
#include <autodiff/forward/dual.hpp>

autodiff::dual f(autodiff::dual x)
{
    return 1 + x + x*x + 1/x + log(x);
}

int main()
{

  std::cout << "Testing autodiff." << std::endl;
  autodiff::dual x = 1.0;
  autodiff::dual u = f(x);
  double dudx = derivative(f, wrt(x), at(x));
  std::cout << "u = " << u << std::endl;
  std::cout << "du/dx = " << dudx << std::endl;
}
```

This program is compiled and run easily by

```
(base) cpplibs % g++ -std=c++20 test-auto1.C -I/Users/patricknoble/
    ↪ Documents/Library/autodiff -o test-auto1
(base) cpplibs % ./test-auto1
Test.
u = 4
du/dx = 3
```

Optimisation: OptimLib

I like using header only libraries, which `OptimLib` supports. First configure `OptimLib` for use in this way, via running

```
./configure --header-only-version
```

while in `/Users/patricknoble/Documents/Library/optim`. This creates a directory `/optim/header_only_version` with `hpp` and `ipp` files needed to use as a header only library.

Note that when compiling the following test example

```
#define OPTIM_ENABLE_EIGEN_WRAPPERS
#include "optim.hpp"

#define OPTIM_PI 3.14159265358979

double
ackley_fn(const Eigen::VectorXd& vals_inp, Eigen::VectorXd* grad_out, void*
    ↪  opt_data)
{
    const double x = vals_inp(0);
    const double y = vals_inp(1);

    const double obj_val = 20 + std::exp(1) - 20*std::exp( -0.2*std::sqrt
        ↪ (0.5*(x*x + y*y)) )
        - std::exp( 0.5*(std::cos(2 * OPTIM_PI * x) + std::cos(2 * OPTIM_PI
            ↪  * y)) );

    return obj_val;
}

int main()
{
    Eigen::VectorXd x = 2.0 * Eigen::VectorXd::Ones(2); // initial values:
        ↪ (2,2)
    bool success = optim::de(x, ackley_fn, nullptr);
    if (success) {
        std::cout << "de: Ackley test completed successfully." << std::endl
            ↪ ;
    } else {
        std::cout << "de: Ackley test completed unsuccessfully." << std::
            ↪ endl;
    }
    std::cout << "de: solution to Ackley test:\n" << x << std::endl;
    return 0;
}
```

taken off the website, I get the following error:

```
(base) cpplibs % g++ -std=c++20 test-optim1.C -I/Users/patricknoble/
    ↪ Documents/Library/optim/header_only_version -I/Users/patricknoble/
    ↪ Documents/Library/Eigen/eigen-3.3.9
In file included from test-optim1.C:2:
/Users/patricknoble/Documents/Library/optim/header_only_version/optim.hpp
    ↪ :29:10: fatal error: BaseMatrixOps/include/BaseMatrixOps.hpp: No
    ↪ such file or directory
   29 | #include "BaseMatrixOps/include/BaseMatrixOps.hpp"
```

which is strange, because `header_only_version/BaseMatrixOps` is there, but is empty. I delete, go to `header_only_version` and clone this manually from the website. I don't see where I missed this step on the webpage but lets see if this works now. Downloading

```
(base) header_only_version % rmdir BaseMatrixOps
(base) header_only_version % git clone https://github.com/kthohr/
    ↪ BaseMatrixOps.git
Cloning into 'BaseMatrixOps'...
remote: Enumerating objects: 342, done.
remote: Counting objects: 100% (342/342), done.
remote: Compressing objects: 100% (194/194), done.
remote: Total 342 (delta 251), reused 227 (delta 139), pack-reused 0
Receiving objects: 100% (342/342), 49.98 KiB | 2.94 MiB/s, done.
Resolving deltas: 100% (251/251), done.
```

Trying again I get a mountain of errors, but catching the top shows me

```
(base) cpplibs % g++ -std=c++20 test-optim1.C -I/Users/patricknoble/
    ↪ Documents/Library/optim/header_only_version -I/Users/patricknoble/
    ↪ Documents/Library/Eigen/eigen-3.3.9 2>&1 | head
In file included from /Users/patricknoble/Documents/Library/optim/
    ↪ header_only_version/BaseMatrixOps/include/BaseMatrixOps.hpp:26,
                 from /Users/patricknoble/Documents/Library/optim/
                     ↪ header_only_version/optim.hpp:29,
                 from test-optim1.C:2:
/Users/patricknoble/Documents/Library/optim/header_only_version/
    ↪ BaseMatrixOps/include/misc/bmo_options.hpp:27:10: error: #error
    ↪ Eigen must be version 3.4.0 or above
```

```
   27 |         #error Eigen must be version 3.4.0 or above
   ...
   ...
```

and sure enough my version of `Eigen` is `eigen-3.3.9`. I nuke my current copy and download 3.4.0, and try again. This works!

Both these libraries have made my header includes look less neat than I like, so I have symlinked them to be standardardised with the rest of my code. This means the test `OptimLib` test example, which requires both `OptimLib` and `Eigen` compiles like

```
(base) cpplibs % g++ -std=c++20 test-optim1.C -I/Users/patricknoble/
    ↪ Documents/Library/OptimLib -I/Users/patricknoble/Documents/Library/
    ↪ Eigen -o test-optim1
(base) cpplibs % ./test-optim1
de: Ackley test completed successfully.
de: solution to Ackley test:
-2.70785e-16
-1.62796e-16
```

which agrees with the solution on the webpage.

Examples