C++ Final Project

For this project, you will write a functioning matrix class in C++.

Note that we are giving you a general specification, and leaving the details up to you. This is common in software development - we often have to make decisions about an implementation where there is no 'correct' answer, but multiple ways of doing things, each with their own benefits and drawbacks.

Required features

1. The matrix class should be templated, so that it works with any type. Particularly, it should work with `double`, `float`, and `int`, along with the `std::complex` variations of those types.
2. The matrix should be constructable three different ways:
    1. With no arguments (`Matrix m()`)
    2. With the desired number of rows and columns (`Matrix m(3,2)`)
    3. Copy constructor
3. The matrix should have the ability to be resized. When using this function, the user should assume that any existing data is either destroyed or invalid after resizing.
4. The class should support the following operations:
    1. Matrix addition (can overload the + operator)
    2. Matrix multiplication (can overload the * operator)
    3. Matrix element-wise multiplication
    4. Assignment operator (assignment of one matrix to another)
    5. Fill the matrix with a particular value
    6. Comparison of two matrices (overload `operator==`. **HINT:** look up documentaton for `std::equal`)
    7. Printing of the matrix to a `std::ostream` object.
    8. A function that returns the transpose of the matrix
    9. A function that returns the complex conjugate of the matrix
    10. A function that returns the conjugate transpose of the matrix
    11. Functions that perform the transpose, complex conjugate, and conjugate transpose in place (ie, they modify the matrix rather than return a new one)
    12. **For extra credit:** Computing the eigenvalues and eigenvectors of a matrix (should use blas/lapack internally)
5. Accessing the elements of the matrix should be the parentheses operator. Ie, you should be able to access element 2,3 via `mat(2,3)`.

The conjugate transpose is very common in quantum chemistry. For details on what it is, see Wikipedia.

As a general guide, you should always be thinking about how you can re-use existing functions to implement new features. For example, a function that returns a transpose and the function that transposes in place are very related. Can one use the other?

Overall, the class should be `const` correct and free of memory leaks. You should throw exceptions when operations aren't valid (ie, matrix multiplication of incompatible sizes).

Since this is a templated class, it should exist completely in a header file.

Finally, create a `.cpp` source file to demonstrate your matrix class and its capabilities.

Your project should have README documenting your class, and a Makefile that compiles your example/test `.cpp` file.