CQEDSimulator

A C/C++ library for Cavity Quantum Electrodynamics Simulations

Stefano Carrazza

Laboratory Kastler Brossel - ENS Paris

September 20, 2009

Outline

Outline

What is this?

Definition

CQEDSimulator is a framework that provides all basic mathematical elements and methods to perform quantum numerical simulations. It's crossplatform, that works on Windows, Linux, Mac...

On Windows you have:

Why a C++ library?

Matlab vs C++

Motivation

- C++ is compiled code, Matlab is a code interpreter
- C++ is faster than Matlab
- C++ allocate and deallocate memory easily and faster than Matlab
- A C++ library can be used inside different languages and programs
- Matlab loads C/C++ libraries

Why a C++ library?

Matlab vs C++

Motivation

- C++ is compiled code, Matlab is a code interpreter
- C++ is faster than Matlab
- C++ allocate and deallocate memory easily and faster than Matlab
- A C++ library can be used inside different languages and programs
- Matlab loads C/C++ libraries

Why a C++ library?

Matlab vs C++

Motivation

- C++ is compiled code, Matlab is a code interpreter
- C++ is faster than Matlab
- C++ allocate and deallocate memory easily and faster than Matlab
- A C++ library can be used inside different languages and programs
- Matlab loads C/C++ libraries

Why a C++ library?

Matlab vs C++

Motivation

- C++ is compiled code, Matlab is a code interpreter
- C++ is faster than Matlab
- C++ allocate and deallocate memory easily and faster than Matlab
- A C++ library can be used inside different languages and programs
- Matlab loads C/C++ libraries

Why a C++ library?

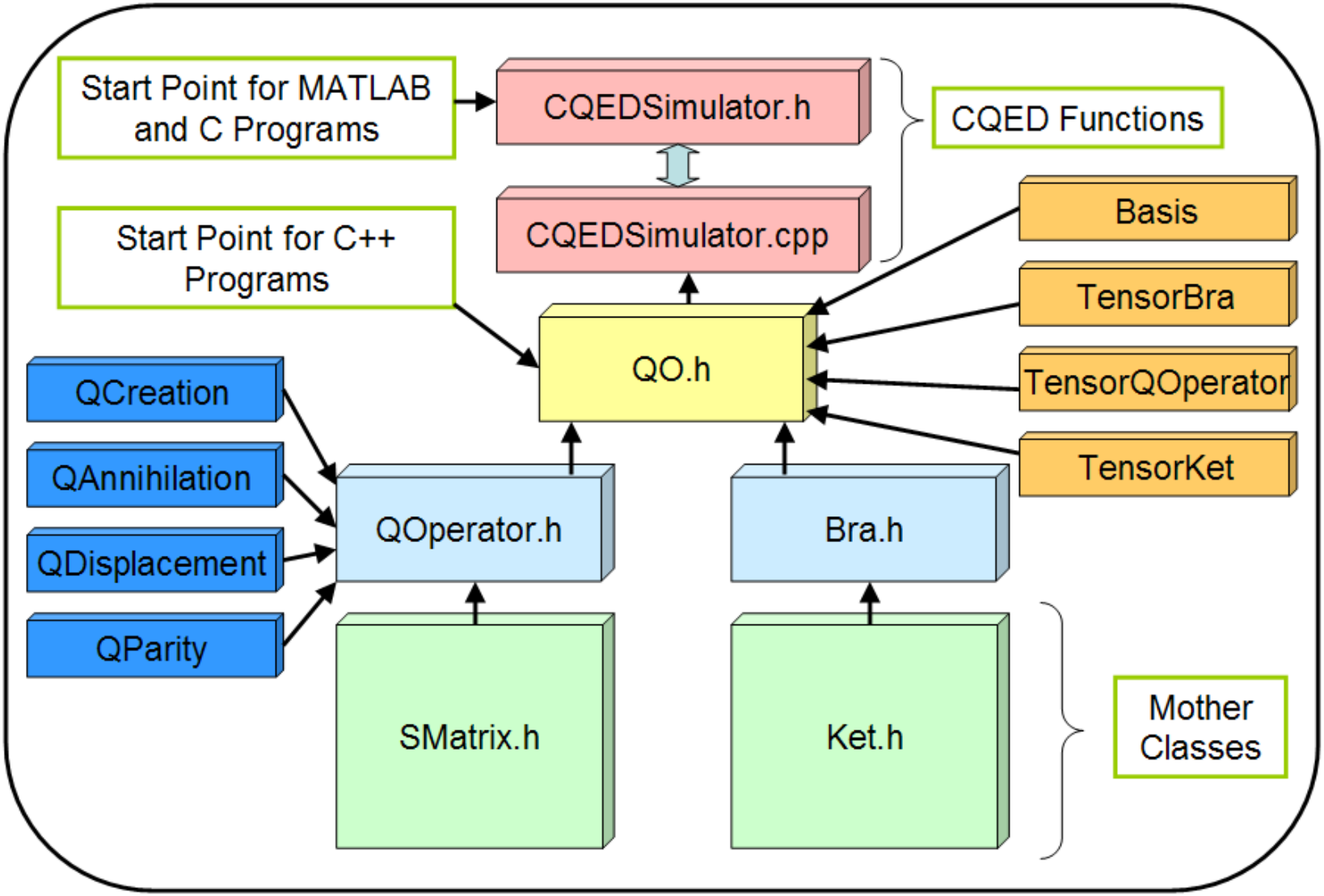
Matlab vs C++

Motivation

- C++ is compiled code, Matlab is a code interpreter
- C++ is faster than Matlab
- C++ allocate and deallocate memory easily and faster than Matlab
- A C++ library can be used inside different languages and programs
- Matlab loads C/C++ libraries

Outline

Internal structure, library chart



QO.h or CQEDSimulator.h?

C++ programs:

If you want to use the library objects inside a C++ program you need to include QO.h. If you want to use the CQED functions you need to add also the CQEDSimulator.h header.

C programs and Matlab

C programs and Matlab cannot use directly the library objects, so you need to create C functions that use this library, all available C function are in CQEDSimulator.h file, so you need to include it in your application.

QO.h or CQEDSimulator.h?

C++ programs:

If you want to use the library objects inside a C++ program you need to include QO.h. If you want to use the CQED functions you need to add also the CQEDSimulator.h header.

C programs and Matlab

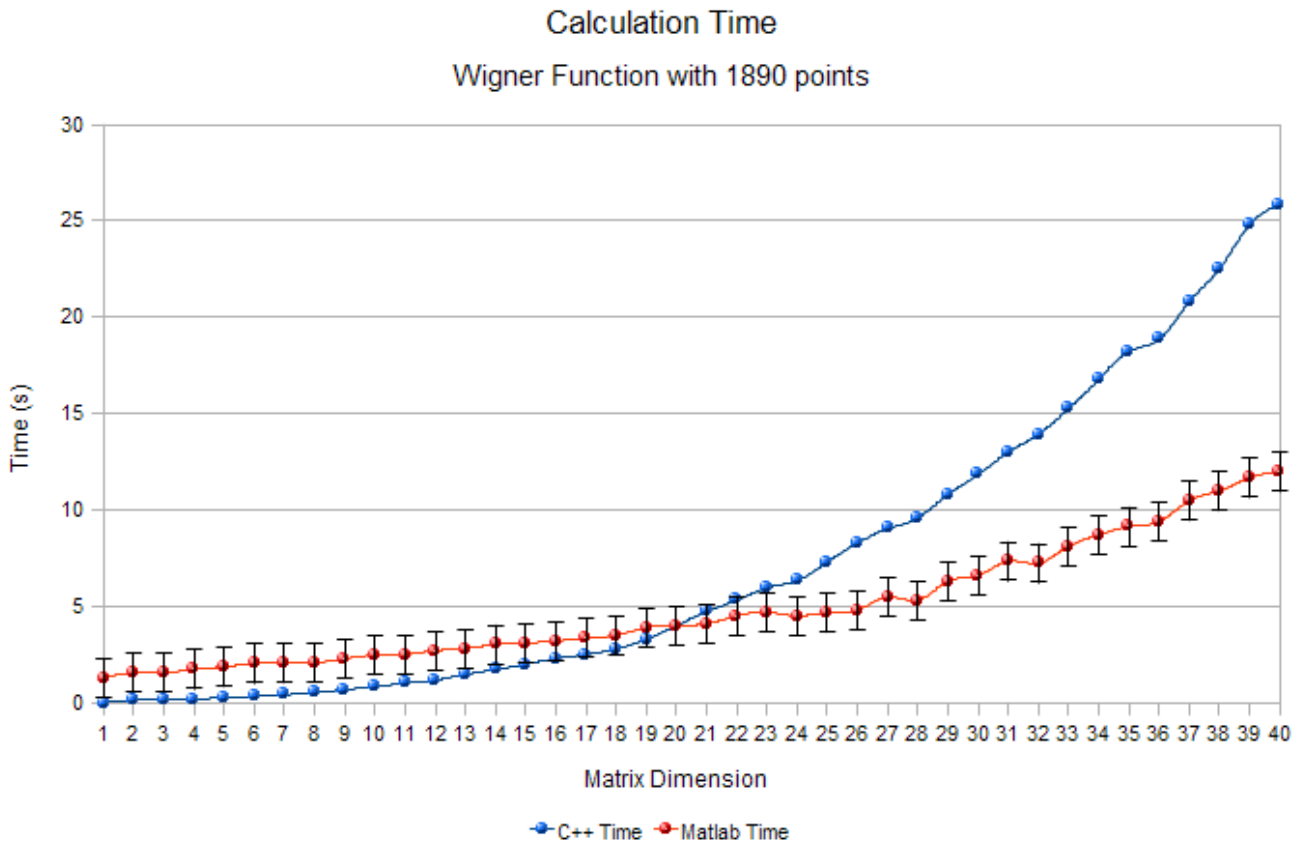
C programs and Matlab cannot use directly the library objects, so you need to create C functions that use this library, all available C function are in CQEDSimulator.h file, so you need to include it in your application.

Outline

Wigner Function for Fock States

CQEDSimulator vs Matlab performance 1/3

Fock state $|0\rangle$ with 1890 points:



Wigner Function for Fock States

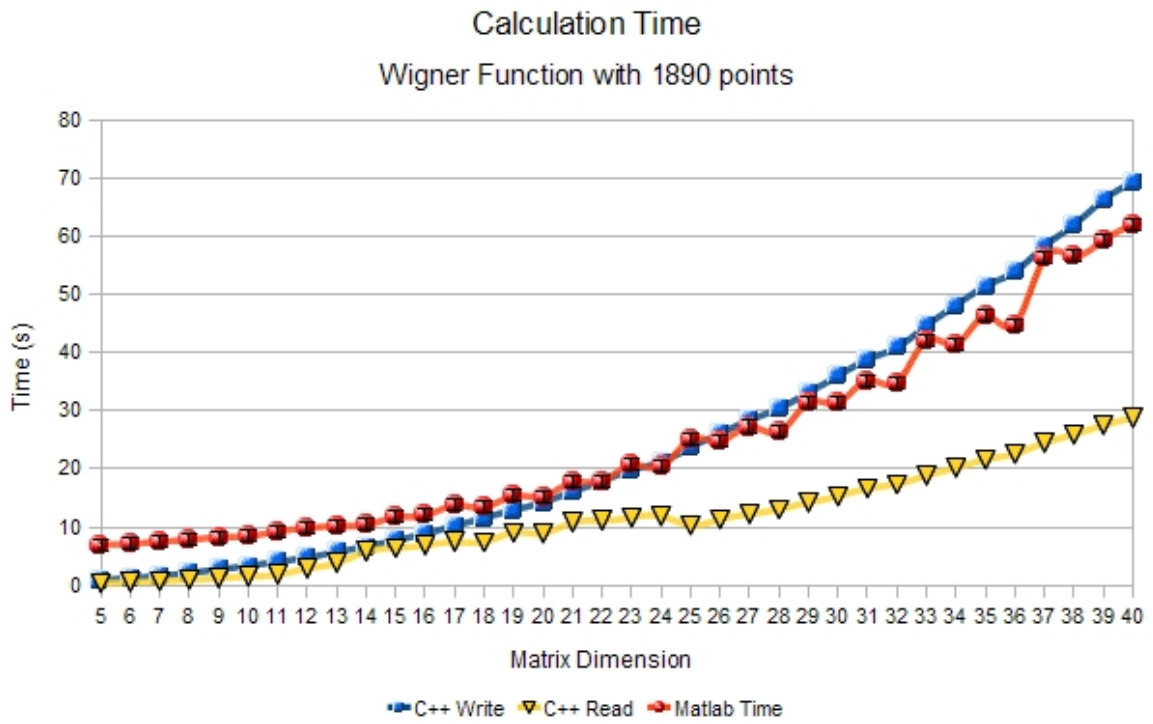
CQEDSimulator vs Matlab performance 2/3

Fock state $|0\rangle$ with 18900 points:

Wigner Function for Fock States

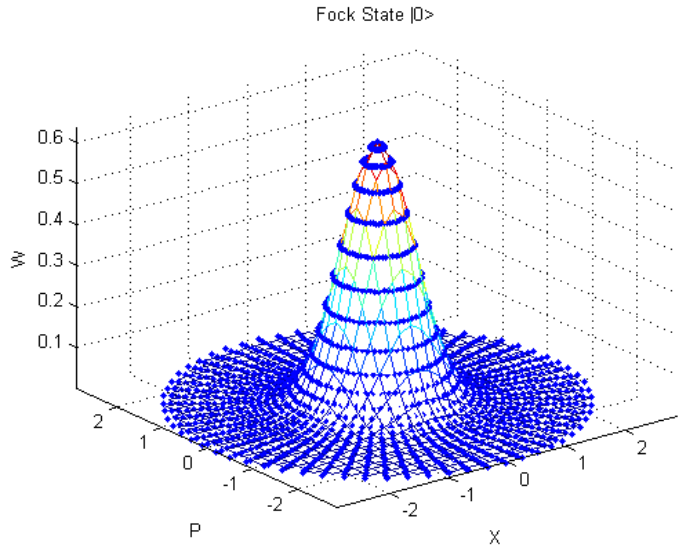
CQEDSimulator vs Matlab performance 3/3

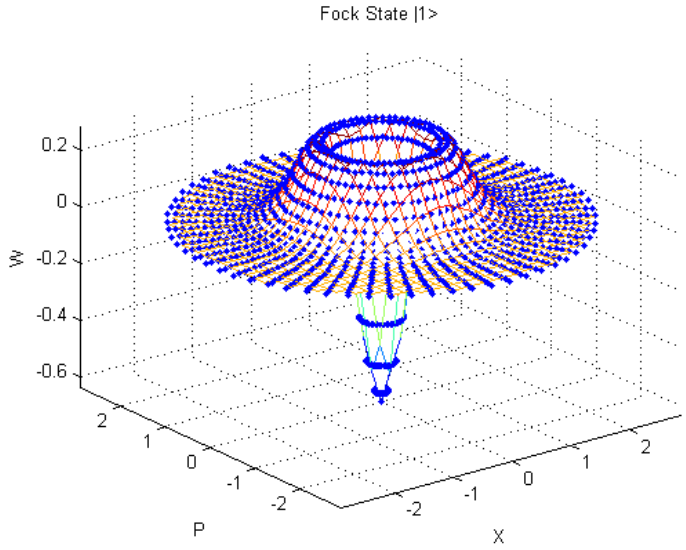
Fock states $n = 0, 1, 2, 3, 4$ with 1890 points, database procedure:

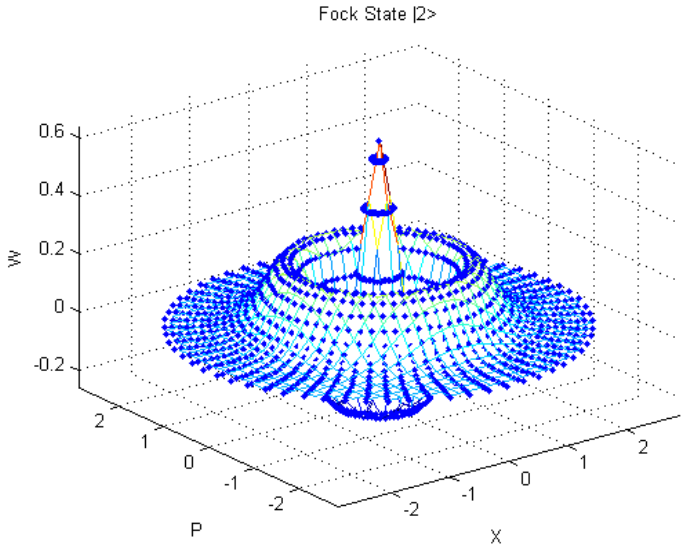


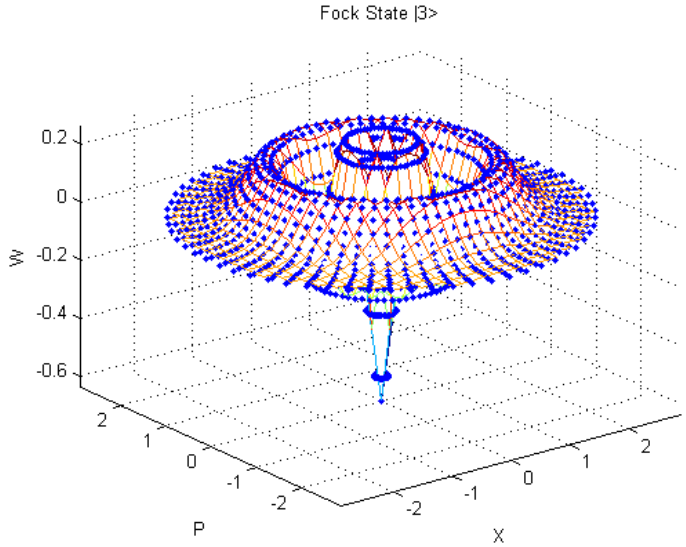
Outline

Wigner Functions









Outline

Code Example

Very simple examples for C++ programs

Class Hierarchy:

- QOperator
  - QAnnihilation
  - QCreation
  - QDisplacement
  - QParity
  - TensorQOperator
- Ket
  - Bra
  - TensorKet
  - TensorBra

Code Example:

```
#include <QO.h> // main header for C++ prog.

int main(){

QOperator A(3); // square matrix dim = 3.
QOperator B(3,1.0); // Id matrix dim = 3.
QParity P(3); // Id matrix dim = 3.

// std matrix operations
A = A+B; A = A*B; A = A/B; A = B.getexpm();
A.print(); TensorQOperator(A,B).print();

// a complex number 1+i
complex<double> alpha(1.0,1.0);

// Wigner Function
double Wigner = 2.0/M_PI*(A*D(2*alpha)*P).gettrace();

return 0;
}
```

Code Example

Very simple examples for C++ programs

Class Hierarchy:

- QOperator
  - QAnnihilation
  - QCreation
  - QDisplacement
  - QParity
  - TensorQOperator
- Ket
  - Bra
  - TensorKet
  - TensorBra

Code Example:

```
#include <QO.h> // main header for C++ prog.

int main(){

Ket ket(3); // ket dim = 3.
Bra bra(3,1.0); // bra dim = 3.

// std operations
complex<double> c = bra*ket; ket = ket+ket;
ket.print();
TensorKet(ket,ket-2*ket).print();

return 0;
}
```

Code Example

Very simple examples for C++ programs

Class Hierarchy:

- QOperator
  - QAnnihilation
  - QCreation
  - QDisplacement
  - QParity
  - TensorQOperator
- Ket
  - Bra
  - TensorKet
  - TensorBra

Code Example:

```
#include <QO.h> // main header for C++ prog.

int main(){

Ket ket(3); // ket dim = 3.
Bra bra(3,1.0); // bra dim = 3.
QOperator A(3); // square matrix dim = 3.

// std operations
A = ket*bra; A.print(); bra.print();
double d = (bra*A*ket).real();

return 0;
}
```

Outline

Loading the library in Matlab

To load the library you only need to :

1. Open Matlab and change the current directory to the folder with CQEDSimulator.dll and CQEDSimulator.h
2. Type: `loadlibrary('CQEDSimulator',CQEDSimulator.h)`
3. To verify the available function type: `libfunctions CQEDSimulator`
4. To call a function:
   `calllib('CQESimulator','Your-Function','parameters')`

Outline

Building the library

Pure C/C++

Building on Windows with Visual Studio

1. Open the Solution file CQEDSimulator.sln
2. Press F7 or go to: Build→Build Solution

Building on Linux and Mac with GNU GCC

1. Open a terminal, change the directory to CQEDSimulator and type:

```
g++ -shared -O2 -o libCQEDSimulator.so SMatrix.cpp
QO.cpp Ket.cpp Bra.cpp QOperator.cpp CQEDSimulator.cpp
```

Building the library

Pure C/C++

Building on Windows with Visual Studio

1. Open the Solution file CQEDSimulator.sln
2. Press F7 or go to: Build→Build Solution

Building on Linux and Mac with GNU GCC

1. Open a terminal, change the directory to CQEDSimulator and type:
   ```
   g++ -shared -O2 -o libCQEDSimulator.so SMatrix.cpp
   QO.cpp Ket.cpp Bra.cpp QOperator.cpp CQEDSimulator.cpp
   ```

Building the library

With Matlab Support

Matlab support

To get and send informations from/to Matlab, you need to add the preprocessor flag `MATLAB` and link against Matlab libraries. In this way you can use functions like SendMatrixToMatlab().

- Add `-DMATLAB` on your compiler command.
- Matlab availables libraries to link are:
  - libmat.lib $\longrightarrow$ to use Matlab objects in a C program.
  - libmex.lib $\longrightarrow$ to send output information to Matlab.
  - libmx.lib $\longrightarrow$ to convert C array in Matlab array and vice-versa.
  - libeng.lib $\longrightarrow$ to control Matlab from a C program.

Building the library

With LAPACK++ Support

LAPACK support

To accelerate the calculation you can use the lapack support:

1. Install Lapack libraries for your operating system.
2. Add the preprocessor flag `LAPACKPP`, and link against Lapack libraries.

Outline

Reference Manual

To get more informations about how to use the library, and what the library provides, please take a look on the reference manual in PDF of HTML created with Doxygen:

- The PDF: CQEDSimulator→Documentation→ refman.pdf
- The HTML: CQEDSimulator → Documentation → html → index.htm

The End