PhD report for the PhD-thesis entitled "**Algorithms for time-independent Schrödinger equations**" by *Toon Baeyens*.

by Kris Coolsaet.

The manuscript was a pleasure to read. The text is of high quality and well-structured. Even for someone like me who only has a cursory knowledge of numerical methods for finding solutions of differential equations, the text manages to give a good impression of the mathematical work done by the candidate and of the challenges involved.

From the point of view of (numerical) informatics, the candidate has done impressive work. This is not only reflected by the substantial increase in speed of the algorithms in comparison to earlier work by other authors on the same problems, but also the source code shows an in depth knowledge and professional insight into the intricacies of the programming language C++, which have been put to excellent use. In addition, the candidate has adhered to good principles of software development by providing an automated test framework for his code, something which should be indispensable when writing software of this kind, but unfortunately is often left out in numerical software or only added as an afterthought.

My only misgiving about the source code is the lack of developer documentation. It will not be evident for a new developer to understand the code, let alone build further upon it.

I have a few questions for the candidate which could serve as a starting point for a discussion during the private defence:

- Could your methods be adapted to 'coarse grained' parallelism, exploiting the fact that most computers now have several processor cores?
- Could your methods be adapted to 'fine grained' parallelism, taking advantage of the vector instructions on CPUs, or even running (part of) the programs on GPUs?
- You explain that the general Sturm-Liouville equation can be reduced to the more specific Schrödinger equation using a specific transformation. To what extent does this transformation reduce the time efficiency and accuracy of your programs? Would it be possible (and worthwhile) to reformulate your algorithms to act directly on the original Sturm-Liouville equation, thus avoiding the transformation?