SOMIM:
An open-source program code for the numerical Search for Optimal Measurements by an Iterative Method

LEE Kean Loon,1,2 CHUA Wee Kang,3 LOOI Shiang Yong,4 and Berthold-Georg ENGLERT1,3

1Centre for Quantum Technologies
National University of Singapore
3 Science Drive 2
Singapore 117543
2Graduate School for Integrative Sciences and Engineering
National University of Singapore
3Department of Physics
National University of Singapore
2 Science Drive 3
Singapore 117542
4Department of Physics
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract
SOMIM is an open-source code that implements a Search for Optimal Measurements by using an Iterative Method. For a given set of statistical operators, SOMIM finds the POVM that maximizes the accessed information, and thus determines the accessible information and one or all of the POVMs that retrieve it. The maximization procedure is a steepest-ascent method that follows the gradient in the POVM space, and uses conjugate gradients for speed-up.

This manual is for version 1.0.
Contents

1 License Agreement 2
2 What can SOMIM be used for? 2
3 Download and compile (optional) 3
4 How to use the program? 3
5 How to import data? 5
6 Meaning of output data 6
7 Contact information 6
8 Acknowledgments 8

1 License Agreement

SOMIM is an open-source code that, for a given set of statistical operators, implements a Search for Optimal Measurements by using an Iterative Method. © 2007 K.L. Lee, W.K. Chua, S.Y. Looi and B.-G. Englert.

SOMIM is free software: You can redistribute it and/or modify it under the terms of the GNU General Public License Version 3 as published by the Free Software Foundation.

SOMIM is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR PARTICULAR PURPOSE. See the GNU General Public License at http://www.gnu.org/licenses/ for details.

2 What can SOMIM be used for?

Given a set of statistical operators $\rho_j$, SOMIM calculates the accessible information (AI) associated with these statistical operators, and finds the POVM that retrieves the AI, or rather: it finds one of the optimal POVMs, as the optimum need not be unique. Repeated runs of SOMIM for the same input data but different random seeds may yield alternative optimal POVMs.

The calculation is performed using the steepest-ascent iteration introduced in Ref. [1]; see also Section 11.5 in Ref. [2]. The implementation in SOMIM makes use of conjugate gradients and the Golden-Section search method.
3 Download and compile (optional)

The complete set of files, including this manual, are available at the SOMIM site: \url{http://theory.quantumlah.org/project/SOMIM/}. Download \url{http://theory.quantumlah.org/project/SOMIM/all.zip} if you want to have the complete collection of files. Just this manual is fetched from \url{http://theory.quantumlah.org/project/SOMIM/SOMIMmanual.pdf}. The Windows executable file for SOMIM can be downloaded from \url{http://theory.quantumlah.org/project/SOMIM/somim.zip}. If you intend to modify the code, you could download the source files from \url{http://theory.quantumlah.org/project/SOMIM/source.zip}.

The program is written in C++ and the graphic user interface (GUI) is implemented using wxWidgets (\url{http://www.wxwidgets.org/}). Here are the instructions for compiling SOMIM:

1. Install wxWidgets from \url{http://www.wxwidgets.org/downloads/}
2. If you are working in Windows, you need to install MinGW (\url{http://www.mingw.org/download.shtml}) and MYSY (\url{http://www.mingw.org/msys.shtml}) as well.
3. When wxWidgets and MinGW are configured, you can compile SOMIM by executing \texttt{g++ MI.cpp \textasciitilde wx-config \textasciitilde \textasciitilde wx-config \textasciitilde cxxflags\textasciitilde \textasciitilde -o YourProgramName} in MSYS shell.
4. If you face problems running the program in a Linux environment, try \texttt{export LD\_LIBRARY\_PATH=/usr/local/lib}
5. The executable file is compiled under Windows XP Service Pack 2, with wxWidgets 2.8.6, MinGW 5.1.3, MSYS 1.0.

4 How to use the program?

The GUI of SOMIM is shown in Fig. 1. In the first box labeled as “Parameter Settings”, \(J\) is the number of statistical operators to be input. The current maximum possible value is \(J = 30\). You can change the maximum value by modifying the source code. Parameter \(K\) is the initial number of elements of POVM, with the largest possible value being \(K = 30\). The third field is the dimension \(N\) of the statistical operators which has \(N = 30\) as the highest possible value; that is: the \(J\) different statistical operators are represented by \(N \times N\) matrices. The fourth field is the chance of using the steepest-ascent gradient method to perform maximization in an iteration; this parameter controls the relative frequency of using the direct or the conjugate gradient. The fifth field gives the tolerance in the accessible information, the stopping criteria for the computation; The calculation stops when the difference in accessible information between the current iteration and the previous iteration
Figure 1: The graphic user interface (GUI) of the program.
is less than half of the sum multiplied by the tolerance plus $10^{-25}$, i.e. when $2.0 \times (\text{current} - \text{previous}) \leq \text{tolerance} \times (\text{current} + \text{previous}) + 1.0 \times 10^{-25}$. The sixth field is the file name of the output file. The output file will be located in the same directory as the program.

The next two boxes display the statistical operators $\{\rho_j\}_{j=1,\ldots,J}$ and the calculated elements of the optimal POVM $\{\Pi_k\}_{k=1,\ldots,K}$. The spin buttons are used to switch between the various $\rho_j$s and $\Pi_k$s, respectively, while the small box beside the spin button is used to choose to display the real or imaginary part of the chosen $\rho_j$ or $\Pi_k$.

The maximum accessible information for the given set $\{\rho_j\}$ will be displayed in the last box after the “Calculate MI” button is pressed. All values will be reset to default when the “Reset” button is pressed.

**Important note:** The matrices for the $\rho_j$s must have the correct dimension; they have to be hermitian with nonnegative eigenvalues; their traces are their statistical weights, which must add to unity: $\sum_{j=1}^{J} \text{tr}\{\rho_j\} = 1$.

5 How to import data?

Data can be imported into SOMIM using a text file that is possibly generated by another program. An example is shown in Fig. 2. When importing, the numbers after the equal signs will be read into SOMIM. The first line gives the dimension $N$ of the statistical operators. The second line gives the number $J$ of statistical operators while the third line gives the number $K$ of elements of the POVMs that SOMIM should start calculating with.

![Figure 2: Example of an import file.](image)

Subsequent lines give the input matrices for the statistical operators. Each line will give only one operator. For an operator represented in matrix form as

\[
\begin{pmatrix}
0.1 & 0.3 + 0.5i \\
0.3 - 0.5i & 0.6
\end{pmatrix},
\]

(1)
the input data should be formatted as \[\{\{0.1, 0.3 + 0.5i\}, \{0.3 - 0.5i, 0.6\}\}\].

Complex numbers are entered as RealPart $+$ ImaginaryPart $i$, as illustrated by $-3.1 - 4.5i$. Please note that the complex unit $i$ must be entered in upper case ($I$) and it must be at the end of the entry.

6 Meaning of output data

A typical output file looks like Fig. 3. The first six lines give the following information: the number $J$ of statistical operators, the initial number $K$ of elements in POVM, the dimension $N$ of the statistical operators, the probability of maximizing information using steepest gradient method compared to conjugate gradient method, the tolerance in the calculated mutual information, and the seed for the random number generator.

The next block of lines gives the mutual information at the end of each iteration,

$$\text{new}_\text{MI} = \sum_{j=1}^{J} \sum_{k=1}^{K} p_{jk} \log_2 \frac{p_{jk}}{p_j p_k}$$

with

$$p_{jk} = \text{tr}\{\rho_j \Pi_k\}, \quad p_j = \sum_{k=1}^{K} p_{jk}, \quad p_k = \sum_{j=1}^{J} p_{jk}.$$  

In the example shown in Fig. 3, 27 iterations have been performed with the final accessible information being $AI = 0.232544$.

The subsequent two blocks of lines give the $J$ statistical operators and the $K$ elements of the optimal POVM that correspond to the accessible information calculated in the final round of iteration. Each statistical operator/POVM element is given in a single line in matrix form, as explained in Section 5.

Among the $K$ POVM elements, if any two elements, let’s say $\Pi_{k_1}$ and $\Pi_{k_2}$, give equivalent probabilities, i.e. $p_{jk_1} p_{k_2} = p_{k_1} p_{jk_2}$ for all $j$, then these two POVM elements are replaced with one new POVM element, $\Pi_{k_1} + \Pi_{k_2}$, such that the new optimal POVM contains only $K - 1$ elements. The last block of data in the output file gives the POVM after this elimination process, i.e. the POVM is the optimal POVM with the least number of elements.

7 Contact information

Please send your comments, suggestions, or bug reports to the following email account: somim@quantumlah.org

SOMIM manual version 1.0
Figure 3: Example of output file.
8 Acknowledgments

We acknowledge many valuable discussions with J. Řeháček. This work is supported by NUS Grant WBS: R-144-000-109-112.

References

[1] J. Řeháček, B.-G. Englert, and D. Kaszlikowski, Iterative procedure for computing accessible information in quantum communication, Phys. Rev. A 71, 054303 (2005); eprint available at http://arxiv.org/pdf/quant-ph/0408134.

[2] J. Suzuki, S. M. Assad, and B.-G. Englert, “Accessible information about quantum states: An open optimization problem”, Chapter 11 in Mathematics of Quantum Computation and Quantum Technology, edited by G. Chen, S. J. Lomonaco, and L. Kauffman (Chapman & Hall/CRC, Boca Raton 2007), pp. 309–348; also available at http://physics.nus.edu.sg/~phyebg/Papers/135.pdf