GPU libraries speed performance analysis for RCWA simulation matrix operation

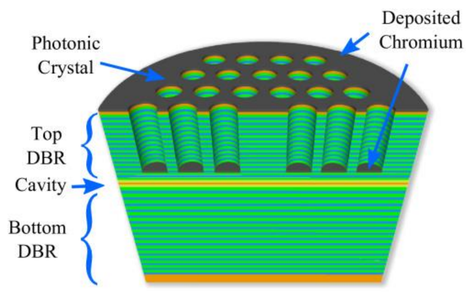
Jingxiao Xu and Martin D. B. Charlton

University of Southampton

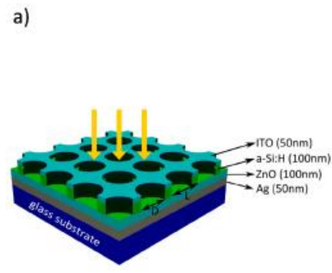
jx4n15@soton.ac.uk

Intro to RCWA and GPU computing

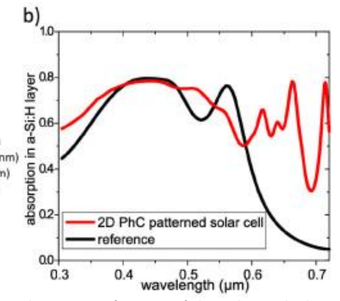
- **Rigorous Coupled-Wave Analysis** (RCWA) solves reflection and transmission of a photonic crystal structure with vertical incident light.
- This method is widely used in designing **photonic crystal (PhC)** structure such as PhC VCSEL, LED and solar cells. To obtain a **full response spectrum** (sweep wavelengths and angles of incident wave), the RCWA simulation process may last for **hours and days** depending on the resolution and range of sweep. A faster algorithm is urgently required.



Photonic crystal VCSEL



Photonic crystal solar cell



absorption performance of PhC and normal solar cell

- **GPU computing** is one of the ways to accelerate matrix operations in RCWA. Modern GPUs usually have **thousands of threads** executing large data **simultaneously** while CPU only have dozens of cores maximum. Using prebuilt and optimized **GPU libraries** is an efficient way to utilize GPU power**.** For NVIDIA GPUs, CUDA toolkit provides a series of math libraries including CUBLAS, CUSOLVER, CUFFT to process a variety of matrix operations. MAGMA is also a linear algebra library implemented mainly on GPU. It supports many functions that CUDA toolkit does not have.

RCWA algorithm

Maxwell's equation of a passive medium

$$\nabla \times E = -\frac{\partial \mathrm{B}}{\partial t}$$

$$\nabla \times H = \frac{\partial \mathrm{D}}{\partial t}$$

Transform fields into **Fourier space**. Variables are represented by finite frequencies.

$$E_i(x,y) = \sum_{m=-M}^{M} \sum_{n=-N}^{N} F_i e^{-j(k_x x + k_y y)}$$

$$\epsilon_r(x,y) = \sum_{m=-M}^{M} \sum_{n=-N}^{N} a_{m,n} e^{j(mT_x + nT_y)r}$$

Create linear equations and solve **eigenvalues** and **eigenvectors** in each layer

$$\frac{d^2}{dz^2}\begin{bmatrix} F_x \\ F_y \end{bmatrix} - \Omega^2 \begin{bmatrix} F_x \\ F_y \end{bmatrix} = 0$$

Frequencies and amplitudes of **allowed modes** obtained.

$$\begin{bmatrix} F_x \\ F_y \end{bmatrix} = We^{-\lambda \tilde{z}} c^+ + We^{\lambda \tilde{z}} c^-$$

W *and* λ^2 are eigenvectors and eigenvalues of matrix Ω^2

Scattering matrix method connecting solutions from each layer and incidence.

$$\begin{bmatrix} c_{ref}^- \\ c_{trn}^+ \end{bmatrix} = [S_{global}] \begin{bmatrix} c_{inc}^+ \\ 0 \end{bmatrix}$$

$$[S_{global}] = [S_{ref}] \otimes [S_i] \otimes [S_{trn}]$$

Reflected and **transmitted** fields are obtained. Diffraction efficiency are calculated using Poynting Vector.

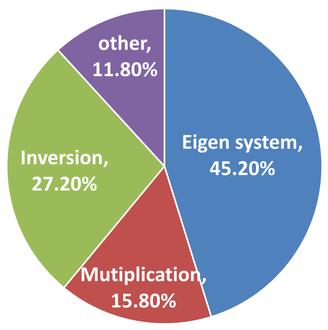
$$R = \frac{\sum F^2 \cos\theta_r}{\cos\theta_{inc}}$$

$$T = \frac{\sum F^2 \cos\theta_t}{\cos\theta_{inc}}$$

Method

A basic RCWA algorithm is implemented in C++ to investigate time cost of each type of matrix operation. The data type used in RCWA program is **double complex** as permittivity of materials may have complex value.

The program utilizes optimized **LAPACK** and **BLAS** routines in Intel MKL library for matrix operations. **Time composition** of a single RCWA simulation with **25 harmonics** is listed below. It is noted that



- **Eigen system** is the most time expensive operation.
- **Matrix multiplication** and **inversion** also contribute a large proportion of total wall time

In order to improve the total RCWA simulation time, we started with the three most time consuming operations. We compare available GPU libraries with Intel MKL on solving the eigen system, matrix multiplication and inversion operations. The experiments are conducted on Iridis 5 supercomputer with **40-cores** CPU and 1 NVIDIA **Tesla V100** GPU.

- **Eigen system:** For a matrix A, if there is a number λ such that $\boldsymbol{Ax} = \lambda \boldsymbol{x}$, λ is called the eigenvalue of A. The corresponding vector x is eigenvector. In the RCWA problem, A is a **dense non-symmetric** matrix. In MKL and MAGMA LAPACK, the eigen system is solved using ZGEEV routine. However, CUDA libraries does not support this function.

- **Matrix multiplication** is commonly used in the algorithm. MKL and CUBLAS have routine ZGEMM3M, an optimized version of ZGEMM. While MAGMA only have ZGEMM routine.

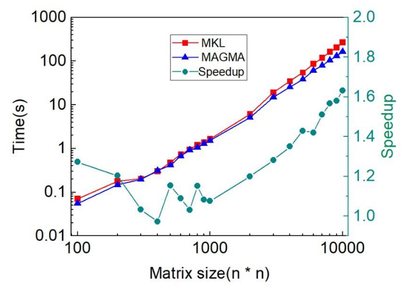
- **Matrix inversion** is less frequently used than multiplication. In MKL and MAGMA LAPACK, a matrix is firstly LU factorised with ZGETRF routine. Then it is inversed using ZGETRI. However, CUSOLVER library does not implement ZGETRI routine. We have to use ZGETRS which solve linear equation $A * X = B$. Let B equals to identity matrix. The X is solved to be the inverse of A.

	Eigen	Multiplication	Inversion
CPU	MKL (ZGEEV)	MKL (ZGEMM3M)	MKL (ZGETRF+ZGETRI)
GPU	MAGMA (ZGEEV)	MAGMA (ZGEMM) CUBLAS (ZGEMM3M)	MAGMA (ZGETRF+ZGETRI) CUSOLVER (ZGETRF+ZGETRS)

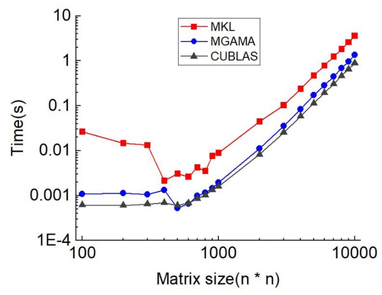
The testing libraries for each operation (LAPACK or BLAS routines name in bracket)

Results

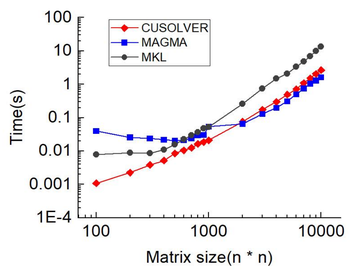
- **Eigen system**: MAGMA has **1.6X** speedup compared with MKL at 10k*10k matrix. The speedup keeps increasing as matrix size.

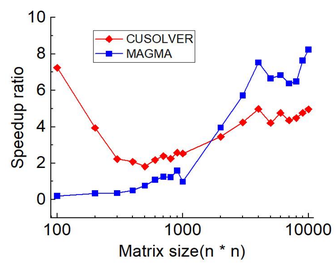


- **Multiplication**: Both CUBLAS and MAGMA outperforms MKL with **3-5X** speedup. While CUBLAS is slightly faster than MAGMA routine because of its faster algorithm.



- **Inversion**: GPU routines also have a max **8X** speedups over MKL for 10k-by-10k matrix. CUSOLVER does not have the routine ZGETRI which causes performance difference between CUSOLVER and MAGMA. MAGMA become faster for large matrix





Conclusion

- GPU libraries have huge advantage in tasks that is **highly parallelizable** such as matrix multiplication and inversion.
- In terms of **scalability**, the GPU speedup over CPU MKL keeps increasing as matrix size.
- It is worth noting that result may varies between different GPUs and CPUs
  - The absolute performance are highly dependant on GPU/CPU **hardware**.
  - Precision also have huge impact. Tesla V100 have much less **single precision** processing speed (14.13TFLOPS) compared with NVIDIA RTX 3070 (20.31FLOPS).
- The analysis could also applies on other **Electromagnetic solver** or other area that requires massive matrix operations.

Reference:
1. Benjamin G. Griffin & Amir Arbabi et al, 2013,"Demonstration of enhanced side-mode suppression in metal-filled photonic crystal vertical cavity lasers," Opt. Lett. 38, 1936-1938
2. Guillaume Gomard & Romain Peretti et al. "Photonic crystals and optical mode engineering for thin film photovoltaics," Opt. Express 21, A515-A527 (2013)
3. Shuba, M. & Faryad, Muhammad et al. 2015. "Adequacy of the rigorous coupled-wave approach for thin-film silicon solar cells with periodically corrugated metallic backreflectors: Spectral analysis." Journal of the Optical Society of America A. 32. 1222. 10.1364/JOSAA.32.001222.
4. W. -L. Yeh, C et al, May 2013, "Enhancing LED Light Extraction by Optimizing Cavity and Waveguide Modes in Grating Structures," in Journal of Display Technology, vol. 9, no. 5, pp. 359-364, doi: 10.1109/JDT.2012.2229382.
5. NVIDIA, Vingelmann, P. & Fitzek, F.H.P., 2020. CUDA, release: 10.2.89, Available at: https://developer.nvidia.com/cuda-toolkit.
6. Raymond C. Rumpf, 2011, "Improved Formulation of Scattering Matrices for Semi-Analytical Methods That Is Consistent with Convention," Progress In Electromagnetics Research B, Vol. 35, 241-261. doi:10.2528/PIERB11083107.
7. "Intel® Math Kernel Library Release Notes and New Features". software.intel.com.