

# High Performance Algorithms for Scalable Spin-Qubit Circuits with Quantum Dots

John Fettig<sup>1</sup>, Nahil Sobh<sup>1</sup>, Dmitriy V. Melnikov<sup>2</sup>, and Jean-Pierre Leburton<sup>2</sup>

<sup>1</sup> National Center for Supercomputing Applications (NCSA), University of Illinois at Urbana-Champaign

<sup>2</sup> Computational Electronics Group, Beckman Institute for Advanced Science and Technology at the University of Illinois at Urbana-Champaign

**Abstract.** This report details improvements made on a code used for computer assisted design (CAD) of scalable spin-qubit circuits based on multiple quantum dots. It provides a brief scientific framework as well as an overview of the physical and numerical model. Then modifications and improvements to the code based on utilization of PETSc are listed. Then new and old codes are benchmarked on three NCSA computer clusters. The speed-up of the code is considerable: about 10 times for the eigenvalue solver and 2 times for the Poisson equation solver. An example of code application towards quantum dot modeling is also given. Finally, conclusions and recommendations for future work are provided.

## 1 Scientific Context

One of the greatest scientific and engineering challenges of this decade is the realization of a quantum computer. In this context, a solid-state system is highly desirable because of its compactness, scalability and compatibility with existing semiconductor technology. In quantum computing, the basic information unit is a quantum bit or qubit, *i.e.* a physical object that can be represented as a superposition of two basis states in a 2D Hilbert space. For this purpose, the use of spin states ( $\mathbf{S}$ ) rather than charge-states as qubit in semiconductor materials is relatively appealing because of their relative insensitivity to electric noise in the device environment.

For single spin operations, carrier confinement is a major issue. The ability to manipulate the spin  $\mathbf{S}$  of electrons by combining gate electrodes and magnetic fields provides the necessary ingredients for controlling spin-qubit operations in nanoscale devices. This scenario has the advantage of relying on established semiconductor fabrication techniques, while semiconductor materials enjoy long spin coherence times, which is of utmost importance for preserving quantum information during many qubit operations [1, ?].

There are presently several proposals for realizing a Control-Not (C-NOT) gate, which is the fundamental circuit element for quantum computation with spins in semiconductors [3]: Among them, the Loss-DiVincenzo scheme is based on the manipulation of electron spins in coupled quantum dots [4]. This proposal is based on the electric control of a singlet ( $\mathbf{S}=0$ ) - triplet ( $\mathbf{S}=1$ ) transition

through quantum mechanical exchange interaction amongst electrons by external electric field in confined nanostructures.

However, the experimental realization of a solid-state quantum device remains a challenge for which the interplay between device geometry and electrostatics, material parameters and spin physics should be integrated into a realistic scheme. Consequently, there is a need for comprehensive high-performance computer tools capable of simulating quantum operations within the device environment while describing the microscopic reality of quantum effects in nanostructures.

## 2 Numerical Model

Our computational approach relies on a 3D self-consistent Poisson–Kohn-Sham scheme based on the spin-dependent density functional theory (DFT) with local density approximation (LSDA). This code is particularly well suited to model microscopic quantum many-body phenomena within the device environment. The electronic states and eigenlevels  $\epsilon$  of the electron system are obtained by solving Kohn-Sham equations twice - once for electrons with spins up ( $\uparrow$ ) and then for electrons with spin down ( $\downarrow$ ) [5]:

$$H^{\uparrow(\downarrow)}\psi^{\uparrow(\downarrow)}(\mathbf{r}) = \epsilon\psi^{\uparrow(\downarrow)}(\mathbf{r}),$$

where Hamiltonian  $H^{\uparrow(\downarrow)}$  is given by

$$H^{\uparrow(\downarrow)} = -\frac{\hbar^2}{2} \left[ \nabla - \frac{ie}{\hbar c} \mathbf{A} \right] \frac{1}{M} \left[ \nabla - \frac{ie}{\hbar c} \mathbf{A} \right] + [-e\phi(\mathbf{r}) + \Delta E_c(\mathbf{r}) + \mu_{xc}(\mathbf{r}) + g\mu_B \mathbf{B} \mathbf{S}].$$

Here  $M$  is the electrons mass,  $\mathbf{A}$  is the vector potential corresponding to the uniform magnetic field  $\mathbf{B}$ ,  $g$  and  $\mu_B = e\hbar/Mc$  are the Lande factor and effective Bohr magneton,  $\mu_{xc}(\mathbf{r})$  is the LSDA exchange-correlation potential [6], and  $\phi(\mathbf{r})$  is the electrostatic potential determined from the solution of the 3D Poisson equation:

$$\nabla [\varepsilon(\mathbf{r})\nabla\phi(\mathbf{r})] = -\rho(\mathbf{r}),$$

where charge density  $\rho(\mathbf{r})$  is equal to  $e[p(\mathbf{r}) - n(\mathbf{r}) + N_D^+(\mathbf{r}) - N_A^-(\mathbf{r})]$ . Here  $\varepsilon(\mathbf{r})$  is the permittivity of the material,  $p(\mathbf{r})$  is the hole concentration,  $n(\mathbf{r})$  the total electron concentration,  $N_D^+(\mathbf{r})$ ,  $N_A^-(\mathbf{r})$  are the ionized donor and acceptor concentrations, respectively. At equilibrium, the electron concentrations in the quantum dots for each spin are calculated from the wave functions obtained from the Kohn-Sham equations, *i.e.*  $n^{\uparrow(\downarrow)}(\mathbf{r}) = \sum_i |\psi_i^{\uparrow(\downarrow)}(\mathbf{r})|^2$ . Outside the active dot region, Thomas-Fermi distribution is used in a first approximation. Hence, the electron density is locally a function of the position of the conduction band edge  $\Delta E_c(\mathbf{r})$  with respect to the Fermi level as opposed to electrons in the active region (quantum dots) that are calculated from the occupation of the quantized 0D eigenstates for which the wave function decays to zero at the device boundaries. The various gate voltages determine the boundary conditions

**Table 1.** Summary of computational platforms

	Tungsten	Mercury
Machine type	Linux Cluster	Linux Cluster
Processor	Intel Xeon 3.2 GHz (32-bit)	Intel Itanium 2 1.3 or 1.5 GHz (64-bit)
Peak Performance	6.4 Gflops	6 Gflops
Memory	3 GB per node	4 or 12 GB per node
Network	Myrinet 2000 interconnect	Myrinet 2000 interconnect
Operating System	Linux 2.4.20 (Red Hat 9.0)	Linux 2.4.21-SMP (SuSE Linux SLES8)
Nodes available	1450 (1280 compute nodes)	631 (1.5 GHz) + 256 (1.3 GHz)
	Copper	
Machine type	IBM pSeries 690	
Processor	IBM Power4 1.3 GHz (64-bit)	
Peak Performance	5.2 Gflops	
Memory	64 or 256 GB per node	
Network	Shared Memory	
Operating System	AIX 5.1	
Nodes available	11 32-processor nodes	

for the Poisson equation. Specifically, Dirichlet conditions are imposed on the top and bottom surfaces of the structure. For the lateral surfaces vanishing electric fields (von Neumann boundary conditions) or periodic boundary conditions are assumed [7].

### 3 Initial Computational Approach

For simulation, the device is mapped onto a 3D non-uniform mesh that is necessary to simulate relatively "large" device features ( $\sim 0.1\mu\text{m}$ ), and yet resolve nanometer-sized details in quantum dots area. Both the Poisson and Kohn-Sham equations are discretized using the finite element method with trilinear polynomials. The self-consistent system of donor charges with the electron charges and spins is solved iteratively using the Newton-Raphson method. The Kohn-Sham equation is solved using subspace iteration method based on Raleigh-Ritz analysis (small number of required eigenpairs made this approach sufficient), while the Poisson equation is solved by using a conjugate gradient method [8].

### 4 Improvements

The code was modified for use on parallel platforms in the following way, still using the finite element method. A parallel conjugate-gradient method using a block Jacobi preconditioner with incomplete LU factorization on the blocks was utilized for solving resulting matrix equations in the Raleigh-Ritz analysis. This was accomplished using the linear solvers and preconditioners from the Portable,

**Table 2.** Comparison between original code and modified one. Calculations are performed on Tungsten. All times are in seconds

	Eigenvalue solve	Poisson solve	Total CPU time
Original code	12	90	8430
Improved code	1.10	53.4	4148

Extensible Toolkit for Scientific Computation (PETSc)[9]. In the presence of a magnetic field, the matrix obtained from Kohn-Sham equation is Hermitian and so a hermitian-conjugate gradient method with the same preconditioner as above was used to solve eigenvalue problem. We found that compared to ordinary conjugate gradient method with Jacobi preconditioner, this approach gives rise to at least an order of magnitude increase in performance especially when dealing with Hermitian matrices.

Parallelization of the code has been implemented in the following way. The code is run in serial on every processor until it reaches the eigenvalue solves. When the matrix, obtained from either Poisson or complex Kohn-Sham equations, is given to PETSc for solving as a part of the Raleigh-Ritz analysis, it is input into a parallel data structure. The solve is done in parallel, and the solution is returned to every processor. Then each processor runs the serial code until it encounters another PETSc solve, when the above process is repeated. This method is acceptable since the time for the linear solves dominates the total run time of the simulation for most problems.

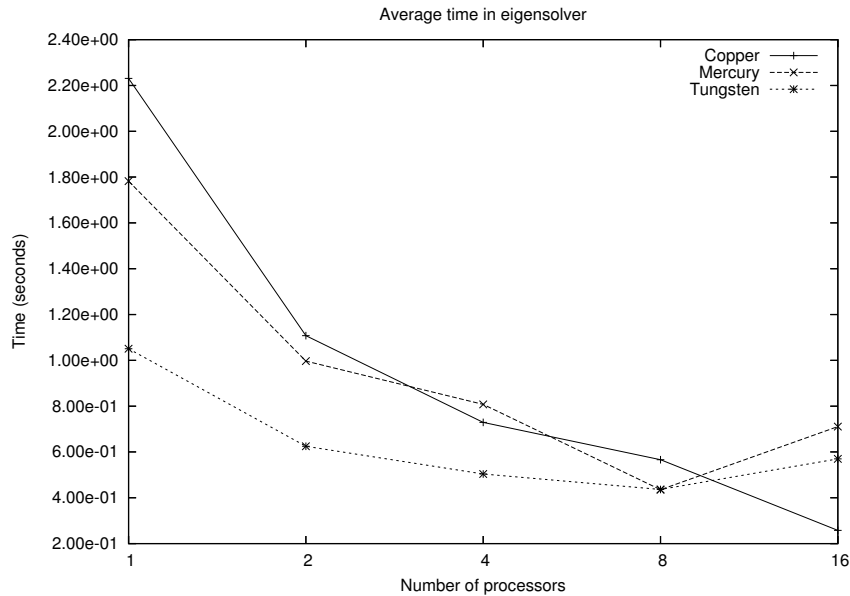
## 5 Results and Applications

### 5.1 Performance Comparison

In order to gage the performance improvements, we performed comparison calculations on various clusters at NCSA. We used a particular problem of finding the eigenspectrum of an empty vertical quantum dot with elliptic confinement potential in a perpendicular magnetic field. While this problem may not be the best possible example demonstrating performance improvements of the code, it converges sufficiently fast to get results in a reasonable amount of time.

In the following discussion, Tungsten refers to a Xeon 3.2 GHz Dell cluster, Mercury refers to an Itanium 2 1.5 GHz IBM cluster, and Copper refers to a POWER4 IBM p690 system. Please see 2 and [11] for further specifications.

We first compare CPU times of the original and modified code on a single processor (Table 2). Calculations are done on the Tungsten cluster at NCSA. One can immediately see that performance of the hermitian conjugate gradient solver is improved by about an order of magnitude. This is due to two reasons: (1) more sophisticated implementation of the conjugate gradient method in PETSc, (2) utilization of the preconditioner in PETSc solver. The latter is probably a more important one. Performance of the Poisson solver is also improved but only

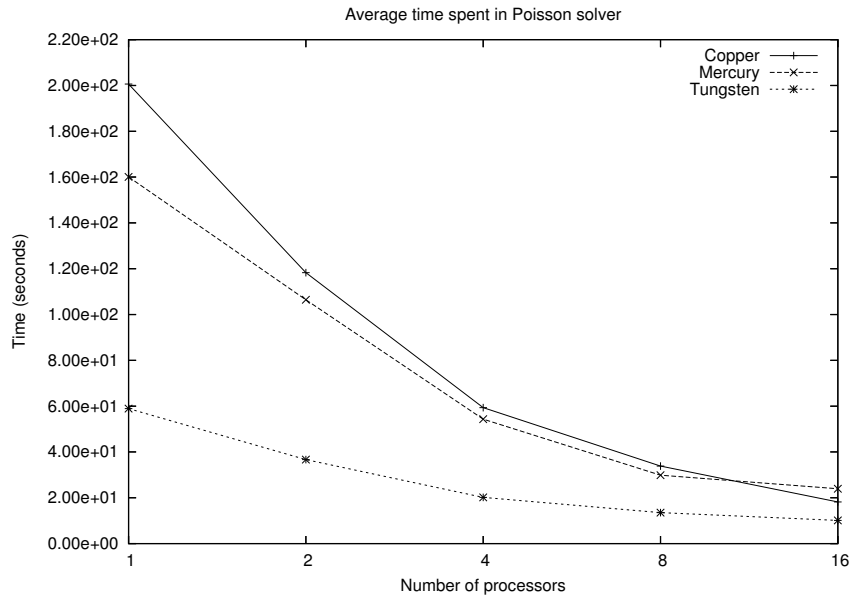


**Fig. 1.** CPU time required on various clusters at NCSA to perform one Rayleigh-Ritz iteration

by about 50 %. This is probably due to utilization of the complex PETSc solver for real matrices which is not the best possible approach. This is done in order to simplify the coding; otherwise, dynamic linking and loading of real and complex PETSc libraries is required. It takes 27 (416) Poisson (Rayleigh-Ritz) iterations to achieve convergence in this example, the rest of CPU time (about 1500 sec.) is used for IO operations which remains constant in both versions of the code. Hence, we see a factor of two difference in total CPU time.

Next we consider parallel scaling of the improved code on the Tungsten, Copper, and Mercury clusters at NCSA. On a single processor, the eigenvalue solver (Fig. 1) is fastest on the Xeon cluster, Tungsten, solving in about 1 second. CPU time decreases when more processors are involved in computation, as expected, and for eight processors it become comparable at all three clusters (about 0.5 sec.). Going to sixteen processors, CPU time on Copper continues to decrease, while CPU time on Tungsten and Mercury increases. For 16 processors on Copper it takes about 0.25 seconds of CPU time to perform one Rayleigh-Ritz iteration, whereas Tungsten and Mercury both bottom out on 8 processors with 0.43 seconds each.

The reason that we see this scaling behavior is largely due to the size of the systems being solved. These solves are done on a subspace of about 50,000 unknowns, which is not large enough to keep a reasonable communication to computation ratio when solving on 16 processors. Hence, on the fastest processor,



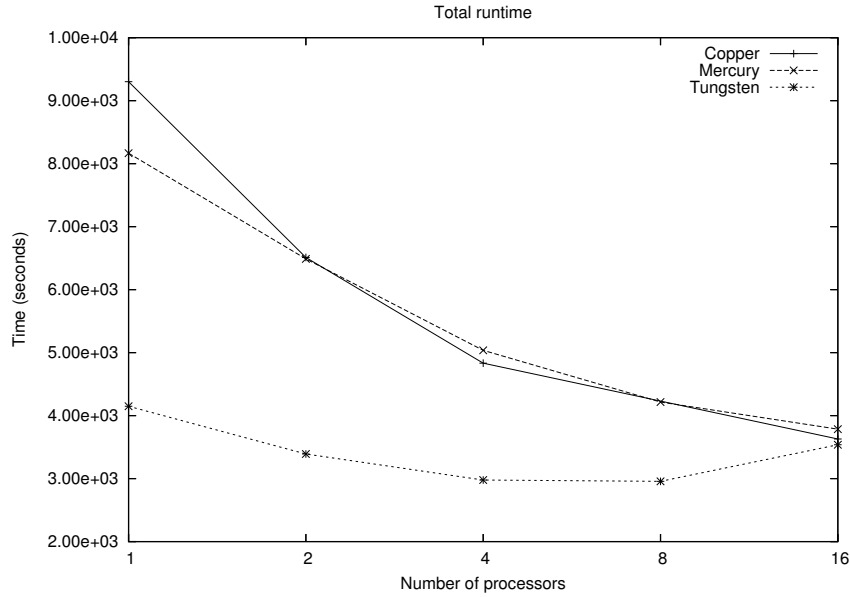
**Fig. 2.** CPU time required on various clusters at NCSA to perform one Poisson solve

Tungsten, we actually see an increase in the average solve time for the eigensolve. In order to see better scaling, we would need to work on a larger problem.

Single-processor performance of the Poisson solver on the three machines (Fig. 2) is similar in ranking to that of the eigenvalue solver. Again Tungsten is the fastest serially, solving in about 60 seconds. The scaling behavior is also similar, with the main difference being that Copper and Mercury show very good speedup to about eight processors, whereas Tungsten shows much less improvement. Nonetheless, Tungsten achieves the fastest solve time on 16 processors with 10 seconds, while Mercury and Copper solve in 24 seconds and 18 seconds, respectively.

Here again, we observe that the scaling past 8 processors is sub-optimal, particularly on the fastest serial processor. Again this can be explained by the size of the systems being solved. They have roughly 500,000 degrees of freedom, which again pushes the communication to computation ratio to an undesirable point. Using the profiling library mpiP, we have calculated the amount of time spent in MPI calls on Tungsten. The results in Table 3 demonstrate that this ratio is getting too high.

The performance demonstrated for eigenvalue and Poisson solvers sums up in total consumed CPU times shown in Fig. 3. Total CPU time on Tungsten is about 4000 seconds serially and does not show terrific scaling with the number of processors. This is a reflection of the sub-par scaling in both the eigensolve and the Poisson solve. The times obtained at Copper and Mercury show linear scaling



**Fig. 3.** Total CPU time required on various clusters at NCSA to solve the test problem

**Table 3.** The communication to computation ratio, determined using mpiP on Tungsten

np	Time spent in PETSc	Time spent in MPI calls	% time spent by PETSc in MPI
2	1250.0 s	163 s	13.0 %
4	754.29	178.75	23.6
8	546.04	317.50	58.1
16	509.36	323.75	63.5

until four processors, and then they move towards saturation suggesting that interprocessor communication and IO operations become dominant. Curiously, on 16 processors all times asymptotically approach the same limit of about 4000 sec. which is about equal to CPU time on Tungsten.

From the above comparisons we can conclude that (1) the version of the code utilizing PETSc gives a marked improvements over the original code, in particular for the complex eigenvalue solve, and (2) all three NCSA clusters react differently on PETSc solvers used in the code. Tungsten appears to produce the least impressive scaling, while Copper and Mercury behave in an expected way, that is, linear scaling of CPU time with processor number at small number of used processors and then saturation. Also, one can see that the optimal number of processors in this example is equal to four; if more are used, interprocessor

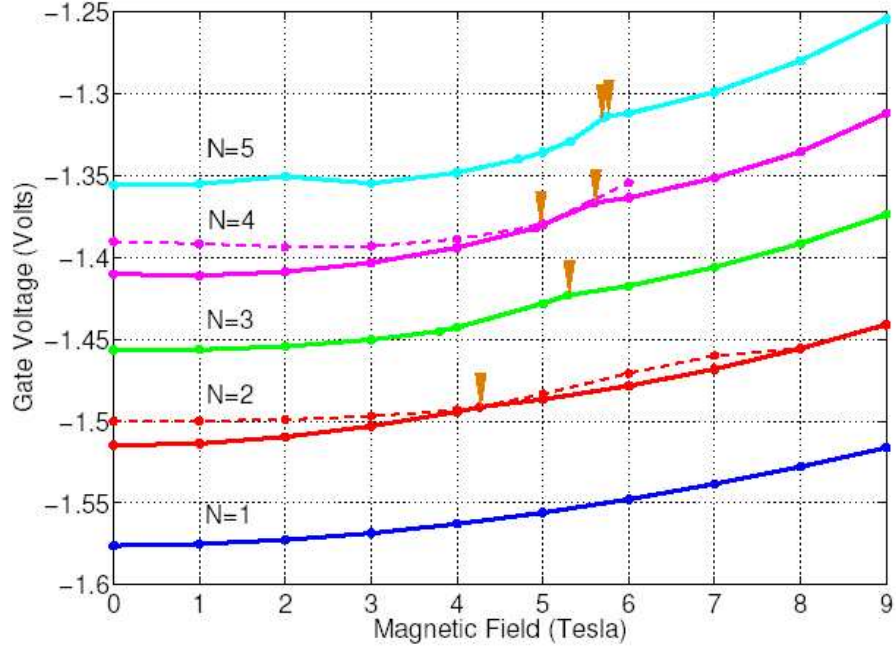
communication time become an issue and the gain in performance does not outweigh the cost of extra CPU's.

## 5.2 Application Example: Single Vertical Quantum Dot in Magnetic Field

The drastic speed-up of the code allowed us to consider a wide range of physical and engineering problems. Here, we apply it to study addition spectra and charging diagram of a rectangular quantum dot in magnetic field [10]. In such a structure confinement in lateral directions is created by applied gate bias while in vertical direction it is ensured by double-barrier heterostructure. By varying gate bias, different number of electrons can be admitted one by one in the quantum dot. This particular problem is of interest as single quantum dots can be regarded as "building blocks" for a scalable quantum computer. It also demonstrates major features expected for a double-dot systems with two confined electrons. Namely singlet-triplet separation is positive at zeroth magnetic field, changes sign at some particular value of the field, and then asymptotically approach zero in the limit of a very strong field. Such behavior of two electron system should in principle allow greater control over the value of exchange interaction necessary for realization of basic quantum computing (see Section 1).

The results of calculations for small number of electrons  $N < 6$  are shown in Fig. 4. The curves in this plot are boundaries separating different stable charge configuration regions where number of electrons in the dot is constant. As bias value increases (decreases in absolute value), the levels go down allowing larger number of electrons to be present in quantum dot. On the other hand, as magnetic field increases, we note the general upward trend of these curves. This is because effective confinement is made stronger, and it requires more positive gate bias to admit an electron in the dot. Coulomb repulsion (screening) gives rise to an effective decrease of the confinement strength with an increasing number of electrons in the QD. This is also seen in Fig. 4 as a decrease in separation between consecutive curves. We also note that this separation is, in general, also decreases with magnetic field. This could be understood in terms of the Fock-Darwin spectra (2D harmonic oscillator in magnetic field), where separation between energy levels belonging to the same Landau band decreases due to enhanced localization of electrons.

As it is well-known from atomic physics, external magnetic field not only changes separation between energy levels but also causes spin rotations to minimize total energy. Our *spin*-density-functional approach is particularly suited to treat this kind of problems. In addition energy spectra, the changes in spin states are evident as kinks (shown by arrows and diamonds in Fig. 4). These transitions are due to two competitive effects: when electrons with the same spin present in the system, the total kinetic energy increases (due to Pauli exclusion principle), while exchange energy becomes more negative. Note that at some particular value of magnetic field, the electron system always becomes fully spin-polarized (all spins oriented in the same direction). These transitions are depicted by the



**Fig. 4.** Charging diagram in magnetic field. Solid lines divide regions with ground state electron configurations while the dashed lines correspond to charging in the lowest excited states

right-most arrows in Fig. 4. The value of this critical field is increasing with  $N$  as it takes stronger field to overcome increased kinetic energy so that the state with largest total spin become lowest in energy.

The transitions in spin states can be further understood by considering wave function evolution. We illustrate this by plotting electron density  $N = 4$  at two different values of magnetic field (Fig. 5). We see that as field increases, electron density localization is also enhanced. We also note that the profile of the density is also changed: two large "humps" at 4 T (Fig. 5a) are replaced by four smaller "hills" (Fig. 5b). This is due to fact that at 4 T the electrons occupy two lowest orbitals (two electrons have spin up and two down), while four spin-up electrons at 8 T are now in four lowest orbitals.

## 6 Conclusions and Recommendations

In this report modifications and improvements to the existing CAD code based on PETSc implementation are described. Performances of new and old codes were compared on three computer clusters at NCSA. The overall speed-up of the code demonstrated on one particular example is considerable: up to 10 times

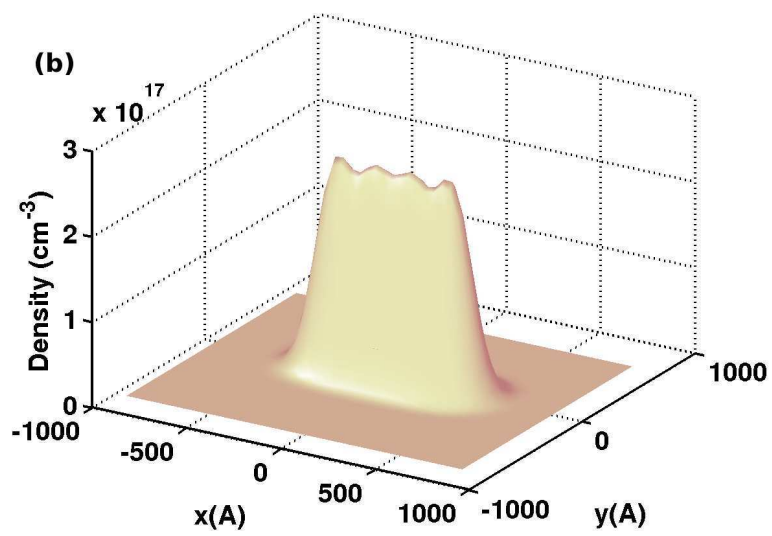
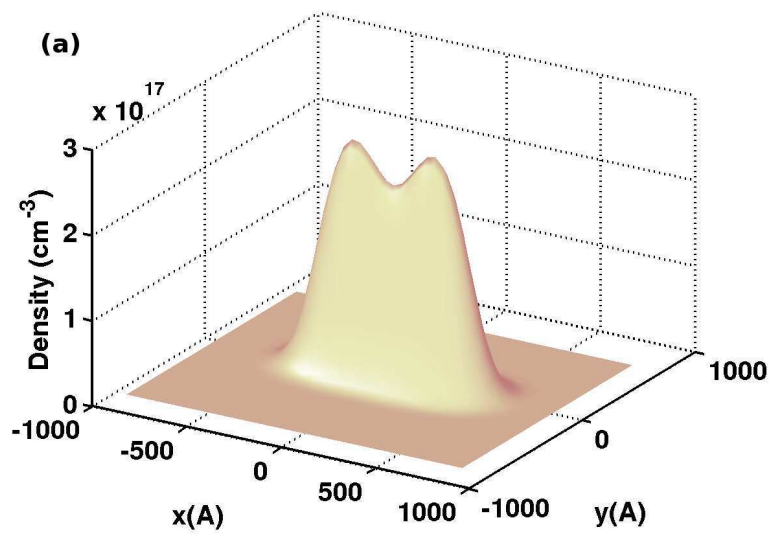


Fig. 5. Electron density for  $N = 4$  at (a)  $B = 4$  T, and (b) at 8 T

for the eigenvalue solver and 2 times for the Poisson equation solver. In other cases observed increase is even greater.

While the code is by no means is up-to-scratch (it could be further sped-up by optimizing IO operations and interprocessor communications), the largest problem now is the visualization of produced data. When one or more parameters are changed at the same time (for example, magnetic field in the example above), and it is necessary to monitor changes in several key characteristics of the simulated device (such as potential, wave-functions, etc.), it quickly becomes a major issue. This is where continuing collaboration with NCSA could be the greatest asset.

## 7 Publications

- D.V. Melnikov, J.-P. Leburton, J. Fettig, and N. Sobh, "Three-Dimensional Self-Consistent Simulation of Circular and Rectangular Quantum Dots in Magnetic Fields", *Phys. Rev. B*, to be submitted.
- D.V. Melnikov, L.-X. Zhang, J. Kim, and J.-P. Leburton, "Three-Dimensional Simulations of Coupled Quantum Dot Devices", Invited Paper on Spintronics, *IEEE Transactions on Nanoelectronics* (England) (2005).
- J.-P. Leburton, Invited Talk at CECAM Workshop on *Modeling of Self-Assembled Semiconductor Nanostructures*, Lyon, France, June 28-30, 2004.

## References

1. B. Kane, *Nature* **393**, 133 (1998).
2. J.M. Kikkawa and D.D. Awschalom, *Phys. Rev. Lett.* **80**, 4313 (1998).
3. M.A. Nielsen and I.L. Chuang, *Quantum Computation and Quantum Information*, Cambridge Univ. Press, 2002.
4. D. Loss and D.P. DiVincenzo, *Phys. Rev. A* **57**, 120 (1998).
5. A. Thean and J.-P. Leburton, *J. Appl. Phys.* **89**, 2808 (2001).
6. J.P. Perdew and A. Zunger, *Phys. Rev. B.* **23**, 5048 (1981).
7. D. Jovanovic and J.-P. Leburton, *Phys. Rev. B.* **49**, 7474 (1994).
8. P. Matagne and J.-P. Leburton, *Phys. Rev B.* **65**, 235323 (2002).
9. S. Balay, K. Buschelman, V. Eijkhout, W.D. Gropp, D. Kaushik, M.G. Knepley, L.C. McInnes, B.F. Smith and H. Zhang, *PETSc Users Manual*, ANL-95/11 - Revision 2.1.5, Argonne National Laboratory, 2004.
10. D.V. Melnikov, J.-P. Leburton, J. Fettig, and N. Sobh, *Phys. Rev. B*, to be submitted.
11. NCSA cluster information: <http://www.ncsa.uiuc.edu/AboutUs/Hardware/>