

# HIGH PERFORMANCE COMPUTATION: NUMERICAL MUSIC OR NUMERICAL NOISE?

C. DENIS\*, F. JÉZÉQUEL†, AND N. S. SCOTT‡

Research in high performance computation is often focused solely on performance. Arguable, the correctness of the results is equally if not more important than the speed of computation. After all, there is little merit in being able to compute numerical noise faster than anyone else. A potential source of significant error in scientific computation arises from the use of fixed size floating point arithmetic. The resulting rounding error produced after every floating point operation is exacerbated in high performance environments where its insidious effect can cause a serious reduction in accuracy. By systematically developing software using CADNA [1], a library based on discrete stochastic arithmetic, we can validate the numerical accuracy of intermediate and final results. In particular, we can determine the number of digits unaffected by round-off in every floating point operation. In this paper we will describe research that is intended to produce computational science software that can exploit high performance architectures and is demonstrably numerically accurate.

The application area of interest to us is electron collisions with atoms and ions. Data from these processes are of importance in the analysis of physical phenomena in many scientific and technological areas. Our current focus is on developing a suite of two-dimensional  $R$ -matrix propagation programs, 2DRMP, aimed at creating virtual experiments on HPC and Grid architectures to study electron scattering from H-like and quasi one-electron atoms and ions at intermediate energies.

The essence of this technique is as follows. The two-electron configuration space  $(x_1, x_2)$  is divided into square sectors as illustrated in Fig.0.1.

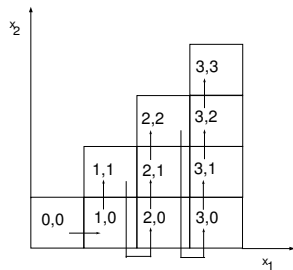


FIG. 0.1. Subdivision of the configuration space  $(x_1, x_2)$  into a set of connected sectors.

The two-electron wavefunction describing the motion of the target electron and the colliding electron is expanded within each sector in terms of one-electron basis functions that are eigenfunctions of the Schrödinger equation, solved subject to certain fixed boundary conditions. The expansion coefficients are determined by diagonalizing the corresponding Hamiltonian matrix. The  $R$ -matrix may then be propagated across the sectors at each scattering energy and the scattering properties of interest determined.

Each sector is independent during matrix construction and diagonalisation, and these tasks may therefore be computed in parallel. However, in large scale virtual experiments, involving around 210 sectors, efficient exploitation is severely impeded by a significant load imbalance between the diagonal and off-diagonal sectors. For example, in matrix construction each diagonal sector can take **of the order of hours** while each off-diagonal sector takes only tens of seconds.

\*Laboratoire d'Informatique de Paris 6, Université Pierre et Marie Curie - Paris 6, 4 place Jussieu, 75252 Paris Cedex 05, France ([Christophe.Denis@lip6.fr](mailto:Christophe.Denis@lip6.fr)).

†Laboratoire d'Informatique de Paris 6, Université Pierre et Marie Curie - Paris 6, 4 place Jussieu, 75252 Paris Cedex 05, France ([Fabienne.Jezequel@lip6.fr](mailto:Fabienne.Jezequel@lip6.fr)).

‡School of Electronics, Electrical Engineering and Computer Science, The Queen's University of Belfast, Belfast, BT7 1NN, U.K. ([ns.scott@qub.ac.uk](mailto:ns.scott@qub.ac.uk)).

We recently attacked this problem by designing numerical methods which advantageously exploited as many characteristic features of the problem as possible [2]. The resulting extended frequency dependent quadrature rules (EFDQR) provided a new computational strategy that is between one and two orders of magnitude faster than the original implementation. For example, with matrices of the order of  $8900 \times 8900$ , the time for matrix construction on HPCx [3] was reduced from 34899 secs to 373 secs. Using CADNA we were able to verify that the results are accurate to 10 rather than 7 figures [4].

The original 2DRMP code stored the diagonal sector matrices on disk after construction. In a subsequent job, using one 16-processor compute node per sector (LPAR), all diagonal sector matrices were read from disk, block cyclically distributed across their respective LPARs and diagonalized using ScaLAPACK. The EFDQR construction method was extended to use one LPAR per sector. Each diagonal sector matrix is built directly in its required block cyclic distribution, thereby allowing diagonalization to follow within the same job and avoiding disk IO and storage. Preliminary results are presented in table 0.1. *This represents a real and very significant reduction in total cpu cycles particularly for larger cases.* For example in the second case, the number of processors is increased by a factor of 16 but the wall clock time is reduced by a factor of 350 and the total processor time, and therefore cost, by a factor of 25.

TABLE 0.1

*Parallel EDFQR compared to the original code. Speedup is given in the square brackets.*

| Matrix size | Code            | No. of diagonal sectors | processors per sector | total no. of processors | wall clock time (secs) | total processor time (secs) |
|-------------|-----------------|-------------------------|-----------------------|-------------------------|------------------------|-----------------------------|
| 1680        | <i>Original</i> | 20                      | 1                     | 20                      | 943                    | 18860                       |
|             | <i>EFDQR</i>    | 20                      | 16                    | 320                     | 4 [235]                | 1043 [18]                   |
| 5090        | <i>Original</i> | 20                      | 1                     | 20                      | 9880                   | 197600                      |
|             | <i>EFDQR</i>    | 20                      | 16                    | 320                     | 28 [350]               | 7610 [25]                   |

Currently each LPAR computes every possible inner integral and reuses it as required. However, in the parallel implementation many of these inner integrals are unused. Determining which is difficult, a difficulty compounded by the block-cyclic distribution of the global matrix. At the conference we will present an update on this work. We will describe cache style storage scheme used to store the optimum reusable collection of inner integrals. We will also describe how CADNA was used to validate the results both of matrix construction and diagonalisation. While parallel dense matrix diagonalisation is easily achieved using ScaLAPACK there has been no analysis of the effect of round-off. This analysis is of particular interest given the ubiquitous use of ScaLAPACK in high performance computing environments.

## REFERENCES

- [1] CADNA website - <http://www.lip6.fr/cadna>
- [2] L. Gr. Ixaru, N. S. Scott and M. P. Scott, *Fast computation of the Slater integrals*, SIAM J. Sci. Comput. **28** (2006) 1252.
- [3] HPCx, UK National Supercomputing Centre - <http://www.hpcx.ac.uk/>
- [4] N. S. Scott, F. Jézéquel, C. Denis and J.-M. Chesneaux, *Numerical 'health check' for scientific codes: the CADNA approach*, Comput. Phys. Commun. doi:10.1016/j.cpc.2007.01.005 (2007)