JCTC Journal of Chemical Theory and Computation

pubs.acs.org/JCTC Article

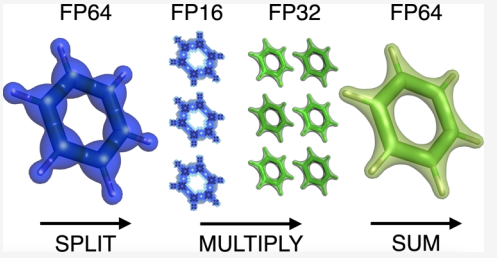
Reducing Numerical Precision Requirements in Quantum Chemistry Calculations

William Dawson,* Katsuhisa Ozaki, Jens Domke, and Takahito Nakajima

Cite This: J. Chem. Theory Comput. 2024, 20, 10826−10837 Read Online

ACCESS | Metrics & More | Article Recommendations | Supporting Information

ABSTRACT: The abundant demand for deep learning compute resources has created a renaissance in low-precision hardware. Going forward, it will be essential for simulation software to run on this new generation of machines without sacrificing scientific fidelity. In this paper, we examine the precision requirements of a representative kernel from quantum chemistry calculations: the calculation of the single-particle density matrix from a given mean-field Hamiltonian (i.e., Hartree−Fock or density functional theory) represented in an LCAO basis. We find that double precision affords an unnecessarily high level of precision, leading to optimization opportunities. We show how an approximation built from an error-free matrix multiplication transformation can be used to potentially accelerate this kernel on future hardware. Our results provide a roadmap for adapting quantum chemistry software for the next generation of high-performance computing platforms.



1. INTRODUCTION

In recent years, progress in the field of artificial intelligence has led to an increase in demand for computing resources to perform deep learning.[1] This has significant implications for developments in computational quantum chemistry—simulation software must be written to coexist on AI-centric software and hardware ecosystems. Of particular importance will be the ability to run quantum chemistry packages on low-precision hardware such as NVIDIA's Tensor Cores or Google's TPUs (see, for example, refs 2 and 3, respectively, for their application to scientific problems). While targeting low-precision hardware will increase the challenge of developing simulation software in the short term, it also presents new opportunities for codesigning specialized hardware optimal for solving scientific problems.

In this paper, we will systematically explore the floating-point precision requirements necessary for a representative quantum chemistry kernel: the calculation of the single-particle density matrix represented in a linear combination of atomic orbitals (LCAO) basis set. We will find that double-precision calculations provide an unnecessarily high level of precision. We will then propose the use of the error-free transformation for matrix multiplication developed by Ozaki et al.[4] (i.e., the Ozaki scheme) in combination with density matrix purification[5] to exploit low-precision hardware while obtaining the required precision (and no more).

2. BACKGROUND

We first review the key points of floating-point calculations for modern hardware. Then, we discuss the history of low-precision calculation algorithms in computational quantum chemistry. Subsequently, we introduce the target algorithm of density matrix purification and its relevance for low-precision calculations (including recent promising work). Finally, we present the Ozaki scheme for efficiently emulating higher-precision matrix multiplication using low-precision hardware.

2.1. Floating-Point Representations.

IEEE-754 floating-point numbers are written in terms of a sign bit, exponent, and mantissa (Figure 1). For example, single precision (FP32) uses 8 bits for the exponent and 23 for the mantissa, whereas half precision (FP16) uses 5 and 10 bits, respectively. Due to the implicit bit, these formats are able to effectively store 24 (FP32) and 11 (FP16) bits of precision in the mantissa. In quantum chemistry codes, the standard is to use double-

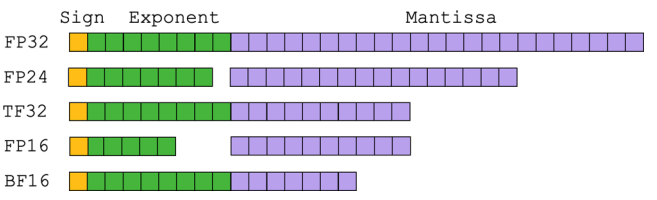


Figure 1. Bitwise representation of various floating-point types. FP32: single precision; FP24: AMD's FP24; TF32: NVIDIA's Tensor Float; FP16: half precision; BF16: BFLOAT16 (brain floating point).

Received: July 18, 2024
Revised: November 18, 2024
Accepted: November 19, 2024
Published: December 7, 2024

precision calculations (FP64), which use 11 bits for the exponent and 52 for the mantissa. Recently, new floating-point formats such as NVIDIA's TF32 (8, 10) or BFLOAT16 (8, 7) have been proposed, specifically for machine learning applications where only a small mantissa is required. In this work, we will also consider AMD's FP24 (7, 16) format as an example of a type with a mantissa between the sizes of FP16 and FP32.

The Tensor Cores (TCs) available on modern GPUs are one way to exploit these low-precision data types. Tensor Cores perform a fused multiply-add (FMA) operation on 4×4 matrices, with accumulation potentially performed in a higher precision than the input matrices. For example, NVIDIA FP16 TCs operate on FP16 matrices and accumulate in FP32. NVIDIA's TF32 TCs likewise take in TF32 matrices and accumulate in FP32. These specialized matrix operations can lead to substantial acceleration over standard floating-point operations. In Table 1, we summarize the floating-point performance for various precision combinations on different GPUs. We see that on an H100 SXM, using FP16 TCs instead of FP64 TCs can potentially accelerate matrix multiplication by nearly 15×. The difference between FP64 and low-precision operations is even larger when FP64 TCs are not available. As demand for AI performance increases, we may anticipate that this gap will increase further in the future.

Table 1. Performance Metrics, in TFLOPS, of Various GPU Models Using Different Floating-Point Representations[a]

GPU model	FP64	FP64 TC	TF32 TC	FP16 TC	FP8 TC
NVIDIA V100 SXM2[6]	7.8			125	
NVIDIA A100 SXM[7]	9.7	19.5	156	312	624
NVIDIA H100 SXM[8]	34	67	494.5	990	1979
NVIDIA RTX 4500 ADA[9]	0.619		79	159	317
NVIDIA L40S[9]	1.4		183	366	733
AMD instinct MI300X[9]	81.7	163	654	1307	2615

[a]For the FP64 values, we report both the performance with and without FP64 Tensor Cores. We note that the performance of the Tensor Cores on newer machines can be increased by a factor of two when sparsity can be exploited.

2.2. Low-Precision Quantum Chemistry Calculations. There has been significant research on low-precision computing for computational quantum chemistry; however, it has primarily focused on single precision vs double precision. Single precision for the analytic calculation of two-electron repulsion integrals in a Gaussian basis set has been demonstrated to be an effective strategy for exploiting GPUs.[10−13] Single precision can also be employed to accelerate seminumerical methods.[14] The single precision strategy has recently been applied to Slater-type orbitals as well.[15] In materials science, single precision has been shown to be a promising strategy to accelerate the computation of exact-exchange in codes using a planewave basis set.[16] Single precision has also been employed to speed up the iterative eigenvalue solvers of materials codes.[17−22] Single-precision calculations are particularly promising for many-body perturbation theory methods. Early work demonstrated the reduced precision requirements of MP2 calculations,[23,24] including when using the RI technique.[25] Single precision can also be applied to coupled cluster,[26,27] including time-dependent variants.[28] Other targets of single-precision optimizations include DMRG,[29] quantum transport calculations,[30] and GW.[31]

A challenge the community has faced for developing low-precision software has been to predict how relevant such optimizations will be to future architectures. For example, Yasuda showed that the evaluation of the exchange and correlation functional could be accelerated using single precision;[32] however, a recent study[33] explicitly rejected this strategy, noting that the gap in performance between double and single precision on GPUs has been closed. The earlier-mentioned work on seminumerical calculations[14] justified its strategy by targeting lower-cost "gaming GPUs", where single precision continues to have higher performance. With recent developments in artificial intelligence, the pendulum has swung back toward the relevance of low (and more exotic)-precision hardware. It is now crucial for the quantum chemistry community to establish clear precision requirements for their simulations. We note that even if double precision-capable hardware remains the standard, studying the effects of low precision will still be potentially useful for reducing data transfer costs.

2.3. Density Matrix Purification. In mean-field quantum chemistry calculations, such as Kohn−Sham density functional theory,[34,35] we need to compute the single-particle density matrix from a given Hamiltonian. Limiting ourselves to the spin-restricted case, we expand the orbitals in some set of M basis functions:

$$\psi_i(r) = \sum_j^M c_{ij}\phi_j(r) \tag{1}$$

We in turn obtain matrix representations of our fundamental operators:

$$S_{ij} = \left\langle \phi_i | \hat{I} | \phi_j \right\rangle \tag{2}$$

$$H_{ij} = \left\langle \phi_i | \hat{H} | \phi_j \right\rangle \tag{3}$$

where $\hat{I}$ is the identity and $\hat{H}$ the Hamiltonian operator. This leads to the generalized eigenvalue problem:

$$H\psi_i = \lambda_i S\psi_i \tag{4}$$

from the solutions of which we can construct the single-particle density matrix:

$$K_{ij} = \sum_a^M f_a c_{ia} c_{ja} \tag{5}$$

where f is the occupation number (usually 2 for occupied and 0 for unoccupied orbitals for spin-restricted calculations of insulating systems).

In most implementations based on LCAOs, eq 4 is solved by invoking a dense eigenvalue solver, such as the ones available in LAPACK or ScaLAPACK. Unfortunately, these calculations have a computational cost that scales with the third power of the number of basis functions. To reduce this cost for application to large systems, many "diagonalization-free" methods have been proposed.[36] One class of "diagonalization-free" methods is based on the purification algorithm first proposed by McWeeny,[37] which iteratively computes K using the following recurrence relation:

https://doi.org/10.1021/acs.jctc.4c00938
J. Chem. Theory Comput. 2024, 20, 10826−10837

$$P_0 = \frac{\lambda}{2}(\mu I - H) + \frac{1}{2}I \tag{6}$$

$$P_{k+1} = 3P_k^2 - 2P_k^3 \tag{7}$$

where μ is the chemical potential, λ scales the spectrum of H to be within the range $[0, 1]$, and $K = 2P$.

The power of such an approach is that the core computational kernel is matrix−matrix multiplication, which can readily exploit the underlying sparsity of H and K that exist for large insulating systems[38]—though whether sufficient sparsity exists to get a computational advantage depends on the system and the choice of basis set. A number of different purification methods exist (see, for example, the methods implemented in the NTPoly library[39]); each employs different forms and orders of polynomials during the iterations. The second-order trace-correcting method of Niklasson[40] has the benefit of only needing to compute the square of a matrix, a point we will return to in Section 4.5.

Density matrix purification is not only useful for the case of extremely large systems where such sparsity exists. Since the purification algorithm has matrix−matrix multiplication as the bottleneck, it has the benefit of scaling better on supercomputers than eigenvalue solvers. When developing a Hartree−Fock code for the Tianhe-2 supercomputer, Chow et al. utilized purification to substantially improve scalability.[41,42] Finkelstein et al. recently demonstrated how an algorithm similar to purification, implemented on a GPU, could outperform dense eigenvalue solvers even on a single GPU.[43]

Pederson et al.[44] similarly proposed using dense purification as a means of exploiting a cluster of Google TPUs. In their work, they use a mixed-precision scheme where early SCF iterations are performed in single precision and the final iterations in software-emulated double precision. Around the same time, Finkelstein and co-workers performed a series of studies using the Tensor Cores available on NVIDIA GPUs.[45,46] They targeted single precision accuracy by taking advantage of the Tensor Core's ability to accumulate in single precision and employing the Markidis scheme.[2] Their work was further extended to density functional perturbation theory[47] and for computing the inverse square root of the overlap matrix.[48] In our work here, we go beyond these earlier studies to achieve the higher precision required for general application.

We note that density matrix purification algorithms are only applicable to insulating systems. To address this, several different algorithms have been introduced that allow for the introduction of the electronic temperature while still having matrix multiplication as the bottleneck (see the Perspective of Aarons et al.[49] or some more recent methods[50−52]). In Supporting Information I, we perform some experiments using the Fermi-operator expansion[53] as a representative finite temperature method to show that the findings of this paper transfer to the case of metallic systems.

2.4. Ozaki Scheme. The Ozaki scheme[4] performs an error-free transformation of computing the product of two matrices into a summation of several matrix multiplications that can be performed without rounding error. Several implementations of the Ozaki scheme exist both to target higher than double precision[54,55] and for using low-precision units like Tensor Cores.[56,57] Remarkably, an implementation based on the INT8 Tensor Cores of an RTX A6000 GPU could outperform double-precision cuBLAS by >4× without loss of accuracy. On future architectures where low-precision arithmetic units further dominate FP64, this scheme would be even more potent.

The details of the Ozaki scheme have been presented in several previous publications; therefore, we only briefly review the concept here (Figure 2). The scheme begins by splitting

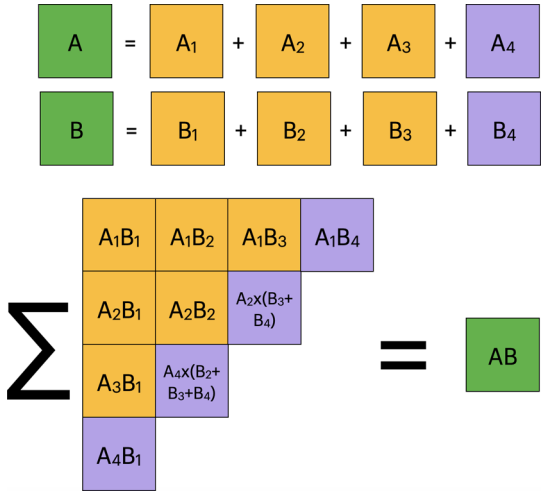


Figure 2. Overview of the Ozaki scheme performed with four splits. The matrices A and B are split into three matrices that can be multiplied exactly in the target precision and a remainder matrix. A sequence of multiplications is performed, which is summed up to form AB. The multiplications in the upper left can be performed with no error, whereas the final multiplications on the diagonal introduce the error from truncating the number of splits.

the input matrices into S split matrices (of the same size as the original). This is done through a series of rounding and bit shifting operations so that each matrix can be represented exactly in the low-precision representation (we include the improved version introduced later by Minamihata et al.,[58] see eq 3 in the paper of Mukunoki et al.[59] for details). A further scaling operation is applied to maintain the exponent's range[56] (we note that in the scaling method of Mukunoki, only error-free terms are considered, but in our implementation, we extend the scaling to all terms). We then compute the product of pairs A_iB_j, where $i + j \leq S + 1$, as well as a set of remainder terms. Finally, the resulting matrices are summed up with the original precision. The accuracy of the final result depends on the number of splits; if both matrices are split S times, we require $2S$ times as much memory and $S(S + 1)/2$ as many multiplications.

The Ozaki scheme can be contrasted with the multiple-component arithmetic approach, such as the popular double−double format.[60] In the multiple-component approach, each individual entry of the matrix is expressed as the sum of lower-precision types. Henry et al.[61] used this approach to leverage BFLOAT16 FMA units to achieve FP32 precision. The Ozaki scheme, on the other hand, splits the matrix into the sum of several lower-precision matrices. Thus, the Ozaki scheme can be thought of as a Structure of Arrays approach, whereas multiple-component arithmetic is an Array of Structures approach. The Structure of Arrays approach can exploit matrix engines like Tensor Cores well since the inner operation remains unchanged.

https://doi.org/10.1021/acs.jctc.4c00938
J. Chem. Theory Comput. 2024, 20, 10826−10837

In Figure 3, we show some example calculations of the multiplication of two random matrices (FP64 elements

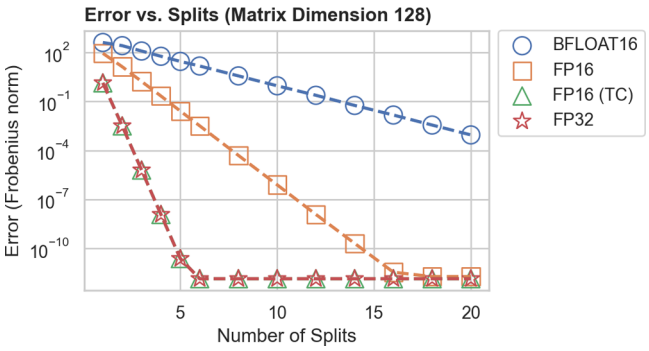


Figure 3. Error in the multiplication of two random matrices as a function of the number of splits used in the Ozaki scheme. We examine the error for four different floating-point types: BFLOAT16 (8, 7), FP16 (5, 10), FP16 with FP32 Accumulation (TC), and FP32 (8, 23).

distributed between [0, 1]) using different precisions and splits. Accuracy is measured as the Frobenius norm of the difference between the double-precision reference result and the Ozaki scheme result. From these data, we see one crucial point about the Ozaki scheme: the floating-point precision for accumulation determines the overall precision. Hence, the Ozaki scheme is particularly effective at taking advantage of hardware like NVIDIA's Tensor Cores.

3. IMPLEMENTATION DETAILS

For this study, we created two different implementations of density matrix purification: one based on NVIDIA's CUDA API for use on an NVIDIA RTX A6000 GPU and another using GNU MPFR to emulate low-precision operations in software. MPFR will be particularly helpful for this study, as it allows us to investigate arbitrarily defined floating-point numbers.[62] Thus, we will use MPFR to explore precise precision requirements and validate our results using the (substantially faster) CUDA implementation.

As input to our purification implementation, we use Hamiltonians coming from the PySCF[63] and BigDFT[64] codes. We use HGH pseudopotentials[65] to remove the core electrons in BigDFT. BigDFT calculations are performed in the linear-scaling mode in order to produce a Hamiltonian in an LCAO basis set.[66] PySCF calculations are done with the polarization consistent basis set series.[67] Calculations with BigDFT are performed with the PBE exchange−correlation functional[68] and for PySCF with B3LYP.[69]

For simplicity, we first transform eq 4 into the standard eigenvalue problem using the Löwdin method:

$$\ddot{H} = S^{-1/2} H S^{-1/2} \tag{8}$$

$$K = S^{-1/2} \ddot{K} S^{-1/2} \tag{9}$$

This transformation is performed by computing the inverse square root of the overlap matrix and transforming the Hamiltonian in double precision (though this could potentially be done with a diagonalization-free method in low precision[48]). All analyses of errors are then performed using the orthogonalized Hamiltonian and density matrix.

For our tests, we implement the trace resetting fourth-order (TRS4) purification method.[40] We chose this method because it gives a clearer convergence signal than lower order methods. As a convergence criterion for the purification iterations, we use a change in the electronic energy, $\mathrm{Tr}(\ddot{H}\ddot{K})$, of 1×10^{-8} Hartree. Due to the introduction of errors from simulating low precision, the purification algorithm may not be able to converge. For cases where convergence cannot be achieved (due to low-precision errors), the purification algorithm halts when the previous two energy values have increased, with an absolute change of less than 1×10^{-2} Hartree (or on the 100th iteration). These failures usually happen when the final energy error is above 1×10^{-8} Hartree, although there are some exceptions. The source code of our implementation is available online (gitlab.com/wddawson/ozp), including Jupyter notebooks for running the experiments (with extra data about convergence detection and exponent underflow).

3.1. Error Evaluation Criteria. In this work, we attempt to evaluate density matrix purification based on low-precision arithmetic as a replacement for the dense eigenvalue solver used in LCAO codes. Of particular importance for such a replacement is that it should not impact the convergence of the self-consistent field (SCF) iterations due to numerical noise. If a reliable ground state solution is found, properties may then be computed using a different precision. In Supporting Information II, we demonstrate this for the case of nuclear gradients.

To monitor SCF convergence, codes based on an LCAO basis set use a variety of different measures. In NWChem 7.2.2, convergence is based on three different criteria. First, there is the root-mean-square deviation (RMSD) of changes in the density matrix. Second, there is the change in the electronic energy, $\mathrm{Tr}(\ddot{H}\ddot{K})$, between iterations. Third, the norm of the change in the commutator DIIS error vector is used:[70]

$$\| \ddot{H}\ddot{K} - \ddot{K}\ddot{H} \|_F \tag{10}$$

The default convergence criteria for these measures when performing DFT calculations are 1×10^{-5} (RMSD), 1×10^{-6} Hartree (energy), and 5×10^{-4} (DIIS). Some other codes use stricter convergence criteria by default. In Gaussian 16, the default cutoff for the RMSD is 1×10^{-8}. In PySCF 2.7.0, the convergence is based on the change in the energy being below 1×10^{-9}.

Here, we target to achieve a precision in the RMSD, energy, and DIIS error vector at about 2 orders of magnitude below each of the default settings for NWChem. An error above this in the density matrix may make it difficult to test for convergence, as numerical precision leads to fluctuations. An error in the energy may also lead to unacceptable noise in properties, such as energy differences. If the Hamiltonian does not commute with the density matrix, the optimization procedure will take steps related purely to numerical noise, which could severely hamper convergence. Approximately 2 orders of magnitude would also ensure that tighter convergence thresholds can be used when needed, such as when computing numerical gradients to build the Hessian matrix.

4. NUMERICAL EXPERIMENTS

We will now perform numerical experiments to understand what level of precision will be required for practical calculations. Our first experiments will establish the fact that double precision provides more than the necessary precision, opening up the way for approximate calculations. We will then investigate the Ozaki scheme as a means of achieving the target

https://doi.org/10.1021/acs.jctc.4c00938
J. Chem. Theory Comput. 2024, 20, 10826−10837

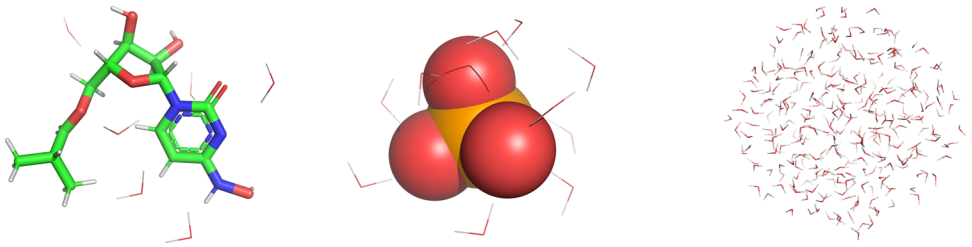


Figure 4. From left to right are the main systems used in this text: Molnupiravir, selenite, and water 237.

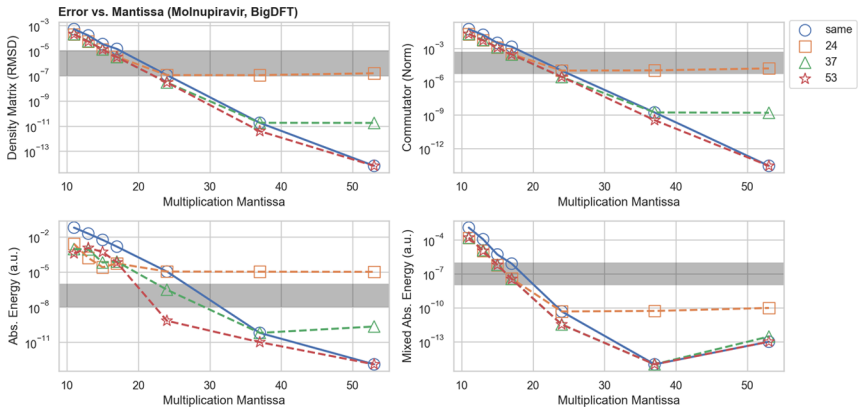


Figure 5. Errors when using purification vs various mantissa sizes (including the implicit bit) for multiplication and accumulation applied to the Molnupiravir/BigDFT system (matrix dimension 147 × 147). Different lines represent different values of accumulation precision, and the horizontal axis represents the precision used for multiplication. The "same" line uses the same precision for multiplication and precision. The gray box extends from the NWChem convergence cutoffs down to 2 orders of magnitude lower. "Mixed Abs. Energy" labels the energy error after applying one McWeeny step in full double precision.

precision. We will further apply the Ozaki scheme to larger data sets of matrices to verify the robustness of our findings. We also compare these results to the competing Markidis method. Finally, based on our findings, we will consider how current implementations of purification could benefit from a mixed-precision scheme. We note that for all calculations, low precision is only used for the matrix multiplications and double precision everywhere else.

As test cases, we will first use the Hamiltonian coming from a BigDFT calculation of a Molnupiravir molecule bound to six water molecules. We will then expand the data set to include a selenite ion surrounded by 10 water molecules calculated with different basis sets in PySCF. We will also include water clusters of different sizes and bulk silicon computed with BigDFT. We finally analyze a large water cluster (573 molecules) computed with B3LYP/PCSEG using NTChem[71,72] (due to the system size). The main systems used in this paper are shown in Figure 4.

4.1. Precision Requirement Sweep. In order to understand the precision requirements for computing the density matrix, we perform density matrix purification using different custom floating-point types implemented by using MPFR on the Molnupiravir system. For this experiment, we assume a double-precision range for the exponent. In Figure 5, we plot the error of the resulting density matrix, commutator, and energy compared to a reference result computed with diagonalization in double precision. We vary the effective precision of the mantissa used for multiplication, ranging from 11 (half) to 53 (double). For each calculation, we use one of three fixed accumulation mantissa values (24, 37, and 53) or the "same" precision as used for multiplication.

From these data, we see that single precision is not sufficient to achieve a converged result. The errors in the density matrix and commutator are not quite below the 2 orders of magnitude range set by our NWChem target (much less the codes with stricter criteria). Interestingly, we observed that the error in the energy was significantly worse compared to the other measures. Following the suggestion of Finkelstein et al.,[45] we investigated a mixed-precision approach, where following the convergence of the purification algorithm, we perform one purification step using full double precision. Specifically, in our case, we take one McWeeny step:

$$P = 3P^2 - 2P^3 \tag{11}$$

We found that applying this iterative refinement substantially improved the energy values, although this improvement did not extend to the other measures. For example, with a single-precision mantissa, the absolute energy error improved from 1.08×10^{-5} to 4.67×10^{-11}; however, the RMSD and commutator norm only changed from 1.13×10^{-7} to 1.09×10^{-7} and 1.00×10^{-5} to 9.99×10^{-6}, respectively. We further found that the magnitude of the error in the RMSD for any given multiplication in the iterations remains roughly the same; therefore, there would be little benefit to adjusting the precision across iterations. Overall, we recommend the iterative refinement strategy when precision in the energy is important (such as the final SCF iteration).

Another interesting observation is that a substantial amount of the error in single precision comes from the accumulation phase, not the multiplication step. Importantly, we found that 37 bits of effective precision are sufficient for accumulation; for a single-precision multiplication with 37 bits of accumulation, the errors are nearly indistinguishable from double-precision

https://doi.org/10.1021/acs.jctc.4c00938
J. Chem. Theory Comput. 2024, 20, 10826−10837

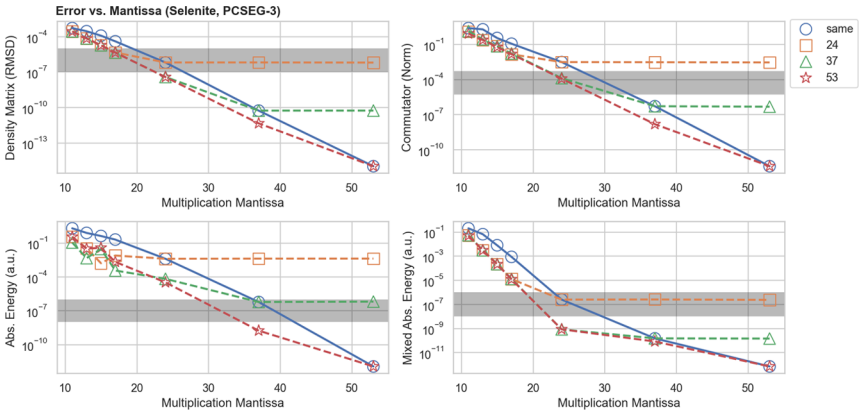


Figure 6. Errors when using purification vs various mantissa sizes (including the implicit bit) for multiplication and accumulation applied to the selenite/PCSEG-3 system (matrix dimension 1508 × 1508). Different lines represent different values of accumulation precision, and the horizontal axis represents the precision used for multiplication. The "same" line uses the same precision for multiplication and precision. The gray box extends from the NWChem convergence cutoffs down to 2 orders of magnitude lower. "Mixed Abs. Energy" labels the energy error after applying one McWeeny step in full double precision.

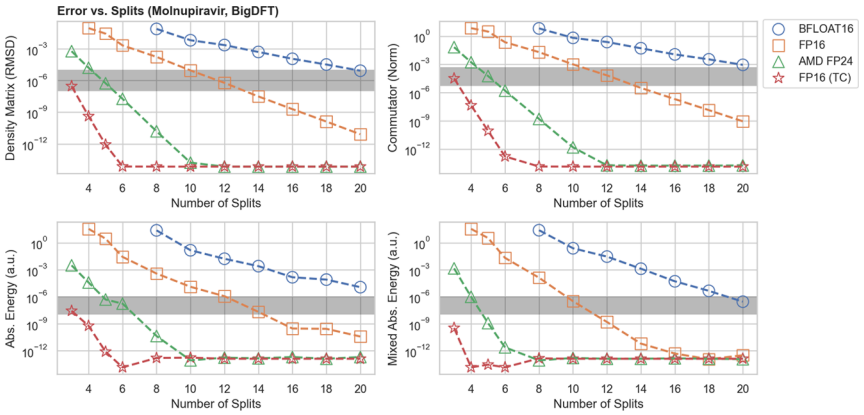


Figure 7. Errors when using purification vs the number of splits used in the Ozaki scheme applied to the Molnupiravir/BigDFT system (matrix dimension 147 × 147). We examine the error for four different floating-point types: BFLOAT16 (8, 7), FP16 (5, 10), AMD FP24 (7, 16), and FP16 with FP32 (8, 23) accumulation (TC). The gray box extends from the NWChem convergence cutoffs down to 2 orders of magnitude lower. "Mixed Abs. Energy" labels the energy error after applying one McWeeny step in full double precision.

accumulation. It is thus possible to obtain a high precision result while storing (and hence communicating) intermediate matrices in single precision if they are multiplied in a higher precision (a point we will return to in Section 4.6). Overall, we find that the double precision commonly used for quantum chemistry calculations essentially provides an unnecessarily high level of precision for computing the single-particle density matrix.

We now repeat this experiment using the selenite system with the PCSEG-3 basis set (Figure 6). The overall pattern is very similar; however, the errors are shifted up compared to the BigDFT calculation. We thus conclude that the quadruple-ζ basis set represents more challenging numerical conditions and hence requires larger floating-point data types. Even in this case, though, FP64 provides far more precision than is required. If refinement is applied to the energy values, a 37-bit effective mantissa provides a sufficient amount of precision and may provide acceptable results when used to accumulate the multiplication of single-precision matrices. Overall, we find that low-precision algorithms show promise for these problems, particularly if they can efficiently exploit the fact that less than full double precision is required.

4.2. Ozaki Scheme Application. Despite this promising finding, it is unlikely that future hardware will offer implementations of floating-point types with effective 37-bit mantissas. Instead, we propose using the Ozaki scheme with a reduced number of splits to take advantage of reduced precision needs. In Figure 7, we plot the errors as a function of the number of splits for various floating-point representations. When using NVIDIA's Tensor Cores (FP16 with FP32 accumulation), three to four splits (6−10 multiplications) are sufficient for a converged result. By contrast, according to Figure 3, fully reproducing a double precision result on random matrices required 6 splits/21 multiplications. Hence, the Ozaki scheme allows for approximately a factor of 2 savings in computational cost due to the lower precision requirements for quantum chemistry applications. These multiplication counts can be compared with Table 1, where we see that FP16 TCs can have above 15 times the performance of FP64 TCs and more than 100 times the performance of standard double-precision operations.

In all cases, the scaling scheme seamlessly handles the exponent; hence, the precision comes down to the size of the mantissa. Using plain FP16, 14−16 splits (105−136 multiplications) are required. This is a substantial increase in the

https://doi.org/10.1021/acs.jctc.4c00938
J. Chem. Theory Comput. 2024, 20, 10826−10837

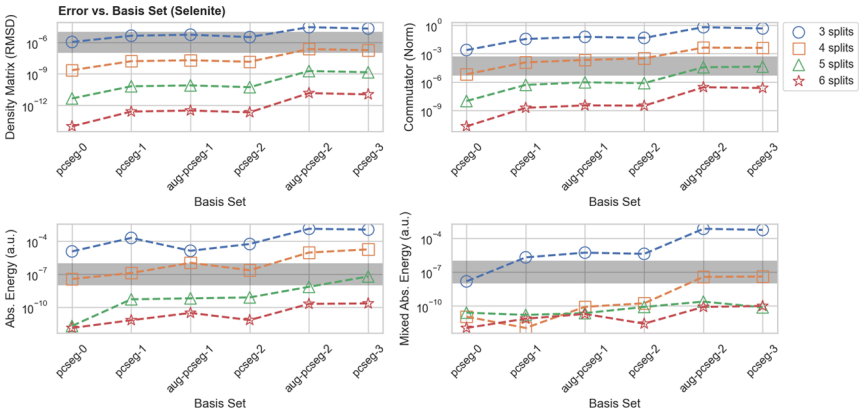


Figure 8. Impact of the basis set on the number of splits required in the Ozaki scheme for the selenite in water system. Matrix sizes are 179 × 179 (PCSEG-0), 309 × 309 (PCSEG-1), 515 × 515 (AUG-PCSEG-1), 713 × 713 (PCSEG-2), 1117 × 1117 (AUG-PCSEG-2), and 1508 × 1508 (PCSEG-3). All calculations are performed with FP16 Tensor Cores. The gray box extends from the NWChem convergence cutoffs down to 2 orders of magnitude lower. "Mixed Abs. Energy" labels the energy error after applying one McWeeny step in full double precision.

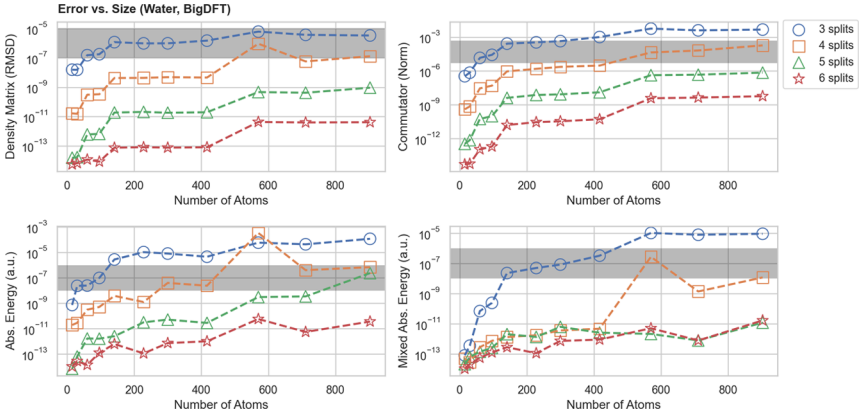


Figure 9. Impact of the system size on the number of splits required in the Ozaki scheme for the water cluster systems. BigDFT uses six basis functions per water molecule so that the smallest matrix is of dimension 30 × 30 and the largest is 1806 × 1806. All calculations are performed with FP16 Tensor Cores. The gray box extends from the NWChem convergence cutoffs down to 2 orders of magnitude lower. "Mixed Abs. Energy" labels the energy error after applying one McWeeny step in full double precision.

number of multiplications compared to the Tensor Core version; however, future hardware may not offer FP16 TC (especially non-NVIDIA chips), with different performance ratios. For a fully converged double-precision result on random matrices, 20 splits/210 multiplications are required, leading again to an approximate factor of 2 savings. We also compare the intermediate-sized mantissa of AMD's FP24 and find that it converges around 7 splits/28 multiplications. For this system, the TRS4 algorithm required 26 calls to the Ozaki scheme to converge. Hence, when using FP16 TCs, an overall speedup would require a time-to-solution ratio of 156−260 multiplications to one double-precision diagonalization. For pure FP16, the requirement increases to 2756−3536 multiplications.

4.3. Potential to Exploit Sparsity. Depending on the underlying distribution of matrix values, the splitting procedure may introduce a significant amount of sparsity. Ichimura et al.[55] proposed exploiting the Ozaki scheme-induced sparsity on a CPU by converting matrices with a sparsity level above 90% to the ELL format and performing sparse matrix−dense matrix multiplication. Sparsity can be more directly exploited to accelerate calculations on Tensor Cores: on an A100 GPU, the FP16 TC performance increases from 312 to 624 TFLOPS for sparse matrices. This sparsity feature requires a special sparsity structure—for any given row/column of four elements, at least two of them must be zero.

We examined the final multiplication performed for the Molnupiravir system by the purification algorithm when using the Tensor Core representation (FP16 with FP32 accumulation) and four splits to see how frequently this sparsity pattern appeared. An entry of the split matrix (after scaling) was considered to be zero if it fell below the smallest value that can be represented by double precision. The first split of both matrices had a number of nonzero blocks below 31%; however, the next split was above 95%, and subsequent splits were nearly fully dense. For the pure FP16 implementation with 14 splits, the matrices were substantially more sparse: the number of nonzeros blocks was below 1% for the first split, 3% for the second split, and 9% for the third split. By the seventh split, the sparsity of both matrices was above 59% (and for all subsequent splits). Thus, we anticipate that some sparsity may be exploited in future work, although it would be limited to the first set of splits.

4.4. Basis Set and Size Effects. We now examine some larger data sets to determine the robustness of these previous results. For this analysis, we employ our CUDA implementation of the Ozaki scheme and focus on the FP16 Tensor Core version with three, four, five, and six splits. For these

https://doi.org/10.1021/acs.jctc.4c00938
J. Chem. Theory Comput. 2024, 20, 10826−10837

Table 2. Errors and Number of Purification Iterations for the Various Systems Computed with FP16 Tensor Cores vs the Numbers of Correction Terms[a]

system	1 Term	2 Terms	3 Terms	4 Terms	4-MPFR	Ozaki (5)	FP64
			RMSD errors				
Molnupiravir	2.2×10^{-4}	1.4×10^{-4}	5.1×10^{-7}	4.4×10^{-7}	2.1×10^{-7}	8.5×10^{-13}	7.0×10^{-15}
Water 237	6.1×10^{-5}	3.7×10^{-5}	2.0×10^{-7}	1.8×10^{-7}	1.1×10^{-7}	4.2×10^{-10}	8.1×10^{-15}
Selenite/A	5.6×10^{-4}	3.2×10^{-4}	1.1×10^{-6}	1.1×10^{-6}	5.8×10^{-7}	6.5×10^{-11}	1.1×10^{-15}
Selenite/B	3.8×10^{-4}	2.3×10^{-4}	1.2×10^{-6}	1.3×10^{-6}	8.6×10^{-7}	7.8×10^{-11}	1.3×10^{-15}
Selenite/C	2.7×10^{-4}	1.6×10^{-4}	2.8×10^{-6}	2.9×10^{-6}	8.0×10^{-7}	1.4×10^{-9}	1.0×10^{-15}
			Commutator errors				
Molnupiravir	2.2×10^{-2}	1.3×10^{-2}	5.5×10^{-5}	4.6×10^{-5}	2.3×10^{-5}	8.6×10^{-11}	2.8×10^{-14}
Water 237	6.5×10^{-2}	4.1×10^{-2}	2.2×10^{-4}	1.9×10^{-4}	1.1×10^{-4}	4.7×10^{-7}	2.9×10^{-13}
Selenite/A	4.0×10^{-1}	1.5×10^{-1}	4.2×10^{-3}	4.5×10^{-3}	4.3×10^{-3}	5.1×10^{-7}	4.1×10^{-13}
Selenite/B	4.1×10^{-1}	1.6×10^{-1}	5.2×10^{-3}	5.7×10^{-3}	4.6×10^{-3}	9.5×10^{-7}	1.4×10^{-12}
Selenite/C	1.3×10^{0}	6.6×10^{-1}	1.4×10^{-2}	1.6×10^{-2}	1.2×10^{-2}	4.3×10^{-5}	3.8×10^{-12}
			Abs. energy errors				
Molnupiravir	2.6×10^{-3}	1.9×10^{-3}	8.7×10^{-5}	8.5×10^{-5}	1.2×10^{-5}	4.8×10^{-13}	1.1×10^{-13}
Water 237	1.9×10^{-2}	2.2×10^{-5}	1.2×10^{-3}	1.2×10^{-3}	3.6×10^{-4}	3.5×10^{-9}	1.7×10^{-12}
Selenite/A	1.3×10^{-1}	1.4×10^{-3}	4.3×10^{-3}	4.3×10^{-3}	1.0×10^{-3}	5.4×10^{-10}	0
Selenite/B	5.8×10^{-2}	8.7×10^{-2}	8.5×10^{-3}	8.4×10^{-3}	2.2×10^{-3}	6.5×10^{-10}	4.5×10^{-13}
Selenite/C	3.5×10^{-1}	2.1×10^{-1}	1.3×10^{-2}	1.3×10^{-2}	4.0×10^{-3}	6.3×10^{-8}	9.1×10^{-13}
			Mixed abs. energy errors				
Molnupiravir	1.9×10^{-4}	6.2×10^{-5}	1.2×10^{-9}	9.8×10^{-10}	2.0×10^{-10}	1.4×10^{-13}	1.1×10^{-13}
Water 237	1.6×10^{-3}	5.3×10^{-4}	2.3×10^{-8}	1.9×10^{-8}	6.9×10^{-9}	8.0×10^{-13}	1.7×10^{-12}
Selenite/A	8.7×10^{-3}	2.4×10^{-3}	8.4×10^{-8}	8.3×10^{-8}	2.9×10^{-8}	1.6×10^{-11}	0
Selenite/B	8.8×10^{-3}	3.0×10^{-3}	2.4×10^{-7}	2.8×10^{-7}	6.9×10^{-8}	2.3×10^{-11}	4.5×10^{-13}
Selenite/C	5.4×10^{-2}	1.6×10^{-2}	4.2×10^{-6}	4.6×10^{-6}	5.3×10^{-7}	7.5×10^{-11}	9.1×10^{-13}
			Purification iterations				
Molnupiravir	17	27	27	24	27	13	13
Water 237	20	45	17	15	13	11	11
Selenite/A	101	74	22	29	23	22	22
Selenite/B	72	101	37	33	30	27	27
Selenite/C	101	66	31	41	32	32	28

[a]Nonconverged systems are labeled as NC (not converged) in the number of iterations. (A) PCSEG-1; (B) AUG-PCSEG-1; (C) PCSEG-3. (1) Term: A^0B^0; (2) Terms: $A^0B^0 + A^0B^1$; (3) Terms: $A^0B^0 + A^0B^1 + A^1B^0$; (4) Terms: $A^0B^0 + A^0B^1 + A^1B^0 + A^0B^0$. 4-MPFR: the 4 Term MPFR implementation with a double-precision exponent. Ozaki (5): Ozaki scheme with 5 splits. FP64: calculations were performed with double precision.

calculations, we first focus on the selenite system. We examine basis sets with up to quadruple-ζ quality as well as augmented versions. As matrices coming from Gaussian basis set calculations have worse conditioning (overlap matrix) and larger spectral widths (Hamiltonian) than the in situ optimized support functions of BigDFT, this test examines the numerical stability of our findings. The selenite ion is an anion (−2), and selenium is a fourth-row element, making it a suitable test case for larger basis sets.

In Figure 8, we plot the error for each of the basis sets with a given number of splits. We observe that for the larger basis sets, the required number of splits increases from four to five—particularly for the errors in the commutator and energy. This is consistent with our previous results, where the difference between four and five splits was small but not negligible. Overall, our earlier findings transfer well to even challenging numerical conditions. Thus, purification with the Ozaki scheme should be applicable to routinely performed quantum chemistry calculations and not just specialized approximate schemes.

We also investigate the number of splits required as a function of system size using water clusters of increasing size computed with BigDFT. This analysis allows us to separate the effect of basis set conditioning with matrix size. For the water clusters, the largest matrix has dimensions of 1806 × 1806, and the selenite systems of 1508 × 1508 (PCSEG-3). The errors with respect to system size are plotted in Figure 9. We find modest growth in the number of splits required with system size. The error with five splits is similar for the largest water cluster as for the selenite system with the largest basis set, with commutator and energy errors being about an order of magnitude lower. Thus, we conclude that using five splits is a sufficient recommendation, though automated schemes may improve the usability of the Ozaki scheme.

4.5. Comparison with the Markidis Method. In the previous work of Finkelstein et al., which performed density matrix purification on NVIDIA Tensor Core units,[45−47] the precision of the result was improved using the method of Markidis.[2] Here, the matrix X is split into a lower and a high precision part:

$$X^0 = \mathrm{FP16}[X] \tag{12}$$

$$X^1 = \mathrm{FP16}[X - X^0] \tag{13}$$

after which, the product AB can be approximated as

$$\mathrm{AB} = \mathrm{FP32}[A^0B^0 + A^0B^1 + A^1B^0 + A^1B^1] \tag{14}$$

In the Markidis method, we can include up to four terms, each refining the precision of the result. In Table 2, we show the precision and the number of purification iterations required

https://doi.org/10.1021/acs.jctc.4c00938
J. Chem. Theory Comput. 2024, 20, 10826−10837

(which may increase due to low precision) for a given number of terms for various systems. We caution that the number of iterations may be made more uniform by a more sophisticated convergence test.[73]

While comparing our CUDA and MPFR implementations of the Markidis method, we noted that a significant amount of error was introduced due to the limited exponent range. To improve this, we applied the same scaling method for the exponent ranges used in the Ozaki scheme[56] and found that it provided substantial improvement. For example, the 4-Term result for Selenite/PCSEG-3 improved from an RMSD error of 1.72×10^{-5} to 2.86×10^{-6}. Nonetheless, the MPFR result with a double-precision exponent still remains more precise; in the future, it may be possible to further improve the Markidis method with a new scaling scheme.

In the implementation of Finkelstein et al., they include three terms in the Markidis correction, which is well-justified from our data. This three-term result can be accomplished at the cost of only two multiplications because they implement a scheme that only requires squaring symmetric matrices, which means that one can exploit the relation $A^1A^0 = (A^0A^1)^T$ even in inexact arithmetic. The benefit of the Markidis method is that it can substantially improve accuracy at a low cost with FP16 Tensor Cores (especially if combined with the exponent scaling scheme). On the other hand, it does not converge to the double-precision result, cannot take advantage of lower-precision hardware like FP16 without FP32 accumulation, and has significant errors in more challenging numerical conditions (like larger systems and basis sets). In particular, the commutator error remains large for the Markidis method, and for the Selenite/C system, the energy error is above 10^{-6} Hartree even with the iterative refinement approach. For this reason, the higher-order approximation of the Ozaki scheme is valuable. When only the matrix square is required, the Ozaki scheme can also exploit this symmetry in the calculation of error-free terms (see Figure 2) as well as other optimizations detailed by Uchino.[74] Using this symmetry and only computing the square of the matrix to purify, the Ozaki scheme with four or five terms would require 8 or 11 multiplications, respectively (compared to the two required by the Markidis scheme).

4.6. Mixed-Precision Purification. Finally, we will revisit the result of Sec. 4.1 regarding the precision needed for multiplying matrices and for accumulation. The findings of Figure 5 indicate that the matrices used in purification may be stored in FP32, as long as the multiplication upcasts them to FP64. This could be utilized in current libraries that implement purification[39,75−78] to reduce communication and data transfer costs, even without considering the Ozaki scheme. To validate this finding, we modify the NTPoly code,[39] which implements the TRS4 method using sparse matrix algebra. During the multiplication process, we down cast all input matrix elements back and forth from FP32 to simulate the loss of precision. As a test matrix, we use the Hamiltonians coming from a BigDFT calculation of a bulk silicon supercell (3240 atoms) and an NTChem calculation of a water cluster (573 molecules).

We compare double-precision and single-precision purification against the electronic energy computed with dense diagonalization with double precision. Since NTPoly enforces sparsity in the underlying matrices by filtering small values, we report in Table 3 the errors in energy when using several different thresholds. We find that the use of single precision for input matrix values has a negligible effect on the quality of the purification result. For the Silicon/BigDFT calculation, the number of purification iterations remained unchanged; for the Water/PCSEG-1 system, we did observe that the iterations increased due to stagnation. We anticipate that the results presented here may be further improved with a suitable scaling scheme to handle the limited exponent range. These results demonstrate how the findings of Figure 5 can have an immediate impact on quantum chemistry codes.

Table 3. Errors (Reference—Computed) in Energy (Hartree) for Different Thresholds (ϵ) and Precisions When Performing Density Matrix Purification with the NTPoly Code on Bulk Silicon from BigDFT and a Water Cluster from NTChem

precision	$\epsilon = 1 \times 10^{-5}$	$\epsilon = 1 \times 10^{-6}$	$\epsilon = 1 \times 10^{-8}$
Silicon/BigDFT			
double precision	1.827×10^{-3}	5.904×10^{-5}	3.282×10^{-5}
single precision	1.832×10^{-3}	6.016×10^{-5}	3.280×10^{-5}
Water/PCSEG-1			
double precision	1.030×10^{-3}	3.452×10^{-5}	2.44×10^{-7}
single precision	1.038×10^{-3}	3.261×10^{-5}	5.289×10^{-5}

5. CONCLUSIONS

In this work, we examined the specific floating-point precision requirements of a representative kernel from quantum chemistry calculations: the calculation of the single-particle density matrix using density matrix purification. Exploiting MPFR to emulate arbitrary mantissa sizes, we found that double precision affords an unnecessarily high level of precision. If a precision between single and double precision is used (for example, with an effective 37-bit mantissa), then a reliable result can be obtained. We further identified that single precision is sufficient for many practical problems if accumulation is done in a higher precision; libraries that implement density matrix purification or similar algorithms may immediately exploit this fact to reduce communication and data transfer costs.

To further take advantage of this reduced precision requirement, we proposed the use of the Ozaki scheme with a smaller number of splits. We found that the reduced precision requirements of purification lead to a reduction in the number of multiplications needed for the Ozaki scheme by about a factor of 2. In this work, we have only examined the reliability of this approach and not implemented an optimized version. Nonetheless, on an RTX A6000 GPU, the Ozaki scheme has already been demonstrated to outperform standard double-precision multiplication calls.[57] Furthermore, on an A100, a similar algorithm to purification was shown to require less time to solution then dense diagonalization.[43] Hence, a combination of the two may provide practical benefits in the short term (in particular, a node parallel version). More importantly, if future architectures are developed with an even higher ratio of low-precision multiplication to FP64, our work shows that performance improvements can be realized without sacrificing precision.

The methodology developed here may be straightforwardly applied to a number of other matrix multiplication-based algorithms in quantum chemistry—particularly many-body methods like MP2 and coupled cluster. For materials science applications based on planewaves, finite elements, and so forth, the Ozaki scheme may be used to accelerate steps such as orbital orthogonalization or subspace diagonalization. Software

https://doi.org/10.1021/acs.jctc.4c00938
J. Chem. Theory Comput. 2024, 20, 10826−10837

emulation of low-precision results can substantially increase the time to solution; however, many practical problems remain in reach, particularly with the development of appropriate libraries for multiprecision linear algebra.[79] Another approach may be to introduce artificial numerical noise into codes[80] or use tools like VeriTracer to automatically instrument electronic structure codes and measure floating-point errors.[81,82]

It will be particularly essential to test low-precision algorithms in combination with physically motivated approximations. Skilled practitioners already know how to employ every available approximation (smaller basis sets, lower levels of theory, low-rank approximations, numerical thresholds, spatial locality, and so forth) to achieve reliable results by using as little computational resources as possible. Low-precision approximations will almost certainly fail to have an improved trade off between cost and accuracy than domain-specific methods. Fortunately, the Ozaki scheme represents one low-precision approximation that can accelerate calculations without sacrificing meaningful amounts of precision, making it an ideal candidate for combination with the diverse set of algorithms available in quantum chemistry.

■ ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available free of charge at https://pubs.acs.org/doi/10.1021/acs.jctc.4c00938.

Evaluation of the Ozaki scheme when used with the Fermi operator expansion for finite temperature calculations and when computing nuclear gradients (PDF)

■ AUTHOR INFORMATION

Corresponding Author

William Dawson − *RIKEN Center for Computational Science, Kobe 650-0047, Japan*; orcid.org/0000-0003-4480-8565; Email: william.dawson@riken.jp

Authors

Katsuhisa Ozaki − *Shibaura Institute of Technology, Saitama City, Saitama 337-8570, Japan*

Jens Domke − *RIKEN Center for Computational Science, Kobe 650-0047, Japan*

Takahito Nakajima − *RIKEN Center for Computational Science, Kobe 650-0047, Japan*; orcid.org/0000-0002-0229-3666

Complete contact information is available at: https://pubs.acs.org/10.1021/acs.jctc.4c00938

Notes

The authors declare no competing financial interest.

■ ACKNOWLEDGMENTS

Computations were performed using resources at the Research Center for Computational Science, Okazaki, Japan (Projects: 23-IMS-C029 and 24-IMS-C151). We gratefully acknowledge members of the RIKEN R-CCS Low Precision Working Group for their advice and guidance.

■ REFERENCES

(1) Sevilla, J.; Ho, A.; Besiroglu, T. Please Report Your Compute. *Communications of the ACM* **2023**, *66*, 30−32.

(2) Markidis, S.; Der Chien, S. W.; Laure, E.; Peng, I. B.; Vetter, J. S. Nvidia tensor core programmability, performance & precision. In *2018 IEEE international parallel and distributed processing symposium workshops (IPDPSW)*; IEEE, 2018; pp 522−531.

(3) Lewis, A. G. M.; Beall, J.; Ganahl, M.; Hauru, M.; Mallick, S. B.; Vidal, G. Large-scale distributed linear algebra with tensor processing units. *Proc. Natl. Acad. Sci. U. S. A.* **2022**, *119*, No. e2122762119.

(4) Ozaki, K.; Ogita, T.; Oishi, S.; Rump, S. M. Error-free transformations of matrix multiplication by using fast routines of matrix multiplication and its applications. *Numerical Algorithms* **2012**, *59*, 95−118.

(5) Palser, A. H.; Manolopoulos, D. E. Canonical purification of the density matrix in electronic-structure theory. *Phys. Rev. B* **1998**, *58*, 12704.

(6) NVIDIA Corporation. *NVIDIA Tesla V100 GPU Architecture: The World's Most Advanced Data Center GPU*. 2018. https://images.nvidia.com/content/technologies/volta/pdf/tesla-volta-v100-datasheet-letter-fnl-web.pdf (accessed September 2024).

(7) NVIDIA Corporation. *NVIDIA A100 Tensor Core GPU Architecture: Unprecedented Acceleration at Every Scale*. 2021. https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/a100/pdf/nvidia-a100-datasheet-us-nvidia-1758950-r4-web.pdf (accessed September 2024).

(8) NVIDIA Corporation. *NVIDIA H100 Tensor Core GPU: Extraordinary performance, scalability, and security for every data center*. 2024. https://resources.nvidia.com/en-us-tensor-core/nvidia-tensor-core-gpu-datasheet (accessed September 2024).

(9) HPC Systems Inc. *GPU Performance Specifications Comparison Table*. 2022. https://www.hpc.co.jp/product/wp-content/uploads/sites/3/2022/07/GPU-list_A3.pdf (accessed September 2024).

(10) Yasuda, K. Two-electron integral evaluation on the graphics processor unit. *J. Comput. Chem.* **2008**, *29*, 334−342.

(11) Ufimtsev, I. S.; Martínez, T. J. Quantum Chemistry on Graphical Processing Units. 1. Strategies for Two-Electron Integral Evaluation. *J. Chem. Theory Comput.* **2008**, *4*, 222−231.

(12) Luehr, N.; Ufimtsev, I. S.; Martínez, T. J. Dynamic Precision for Electron Repulsion Integral Evaluation on Graphical Processing Units (GPUs). *J. Chem. Theory Comput.* **2011**, *7*, 949−954.

(13) Tornai, G. J.; Ladjánszki, I.; Rák, Á.; Kis, G.; Cserey, G. Calculation of Quantum Chemical Two-Electron Integrals by Applying Compiler Technology on GPU. *J. Chem. Theory Comput.* **2019**, *15*, 5319−5331.

(14) Laqua, H.; Kussmann, J.; Ochsenfeld, C. Accelerating seminumerical Fock-exchange calculations using mixed single- and double-precision arithmethic. *J. Chem. Phys.* **2021**, *154*, 214116.

(15) Dang, D.-K.; Wilson, L. W.; Zimmerman, P. M. The numerical evaluation of Slater integrals on graphics processing units. *J. Comput. Chem.* **2022**, *43*, 1680−1689.

(16) Vinson, J. Faster exact exchange in periodic systems using single-precision arithmetic. *J. Chem. Phys.* **2020**, *153*, 204106.

(17) Tsuchida, E.; Choe, Y.-K. Iterative diagonalization of symmetric matrices in mixed precision and its application to electronic structure calculations. *Comput. Phys. Commun.* **2012**, *183*, 980−985.

(18) Das, S.; Motamarri, P.; Gavini, V.; Turcksin, B.; Li, Y. W.; Leback, B. Fast, Scalable and Accurate Finite-Element Based Ab Initio Calculations Using Mixed Precision Computing: 46 PFLOPS Simulation of a Metallic Dislocation System. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*; Association for Computing Machinery: New York, NY, 2019.

(19) Das, S.; Motamarri, P.; Subramanian, V.; Rogers, D. M.; Gavini, V. DFT-FE 1.0: A massively parallel hybrid CPU-GPU density functional theory code using finite-element discretization. *Comput. Phys. Commun.* **2022**, *280*, No. 108473.

(20) Das, S.; Kanungo, B.; Subramanian, V.; Panigrahi, G.; Motamarri, P.; Rogers, D.; Zimmerman, P.; Gavini, V. Large-Scale Materials Modeling at Quantum Accuracy: Ab Initio Simulations of Quasicrystals and Interacting Extended Defects in Metallic Alloys. In *Proceedings of the International Conference for High Performance*

Computing, Networking, Storage and Analysis; Association for Computing Machinery: New York, NY, 2023.
(21) Woo, J.; Kim, S.; Kim, W. Y. Dynamic Precision Approach for Accelerating Large-Scale Eigenvalue Solvers in Electronic Structure Calculations on Graphics Processing Units. *J. Chem. Theory Comput.* **2023**, *19*, 1457−1465.
(22) Khadatkar, S.; Motamarri, P. Subspace recursive Fermi-operator expansion strategies for large-scale DFT eigenvalue problems on HPC architectures. *J. Chem. Phys.* **2023**, *159*, No. 031102.
(23) Vogt, L.; Olivares-Amaya, R.; Kermes, S.; Shao, Y.; Amador-Bedolla, C.; Aspuru-Guzik, A. Accelerating Resolution-of-the-Identity Second-Order Møller-Plesset Quantum Chemistry Calculations with Graphical Processing Units. *J. Phys. Chem. A* **2008**, *112*, 2049−2057.
(24) Olivares-Amaya, R.; Watson, M. A.; Edgar, R. G.; Vogt, L.; Shao, Y.; Aspuru-Guzik, A. Accelerating Correlated Quantum Chemistry Calculations Using Graphical Processing Units and a Mixed Precision Matrix Multiplication Library. *J. Chem. Theory Comput.* **2010**, *6*, 135−144.
(25) Vysotskiy, V. P.; Cederbaum, L. S. Accurate Quantum Chemistry in Single Precision Arithmetic: Correlation Energy. *J. Chem. Theory Comput.* **2011**, *7*, 320−326.
(26) DePrince, A. E., III; Hammond, J. R. Coupled Cluster Theory on Graphics Processing Units I. The Coupled Cluster Doubles Method. *J. Chem. Theory Comput.* **2011**, *7*, 1287−1295.
(27) Pokhilko, P.; Epifanovsky, E.; Krylov, A. I. Double Precision Is Not Needed for Many-Body Calculations: Emergent Conventional Wisdom. *J. Chem. Theory Comput.* **2018**, *14*, 4088−4096.
(28) Wang, Z.; Peyton, B. G.; Crawford, T. D. Accelerating Real-Time Coupled Cluster Methods with Single-Precision Arithmetic and Adaptive Numerical Integration. *J. Chem. Theory Comput.* **2022**, *18*, 5479−5491.
(29) Tian, Y.; Xie, Z.; Luo, Z.; Ma, H. Mixed-Precision Implementation of the Density Matrix Renormalization Group. *J. Chem. Theory Comput.* **2022**, *18*, 7260−7271.
(30) Ziogas, A. N.; Ben-Nun, T.; Fernández, G. I.; Schneider, T.; Luisier, M.; Hoefler, T. A Data-Centric Approach to Extreme-Scale Ab Initio Dissipative Quantum Transport Simulations. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*; Association for Computing Machinery: New York, NY, 2019.
(31) Yu, V. W.-Z.; Govoni, M. GPU Acceleration of Large-Scale Full-Frequency GW Calculations. *J. Chem. Theory Comput.* **2022**, *18*, 4690−4707.
(32) Yasuda, K. Accelerating Density Functional Calculations with Graphics Processing Unit. *J. Chem. Theory Comput.* **2008**, *4*, 1230−1236.
(33) Williams-Young, D. B.; de Jong, W. A.; van Dam, H. J. J.; Yang, C. On the Efficient Evaluation of the Exchange Correlation Potential on Graphics Processing Unit Clusters. *Front. Chem.* **2020**, *8*, No. 581058.
(34) Hohenberg, P.; Kohn, W. Inhomogeneous Electron Gas. *Phys. Rev.* **1964**, *136*, B864−B871.
(35) Kohn, W.; Sham, L. J. Self-Consistent Equations Including Exchange and Correlation Effects. *Phys. Rev.* **1965**, *140*, A1133−A1138.
(36) Bowler, D. R.; Miyazaki, T. O(N) methods in electronic structure calculations. *Rep. Prog. Phys.* **2012**, *75*, No. 036503.
(37) McWeeny, R. Some Recent Advances in Density Matrix Theory. *Rev. Mod. Phys.* **1960**, *32*, 335−369.
(38) Prodan, E.; Kohn, W. Nearsightedness of electronic matter. *Proc. Natl. Acad. Sci. U. S. A.* **2005**, *102*, 11635−11638.
(39) Dawson, W.; Nakajima, T. Massively parallel sparse matrix function calculations with NTPoly. *Comput. Phys. Commun.* **2018**, *225*, 154−165.
(40) Niklasson, A. M. N.; Tymczak, C. J.; Challacombe, M. Trace resetting density matrix purification in O(N) self-consistent-field theory. *J. Chem. Phys.* **2003**, *118*, 8611−8620.
(41) Chow, E.; Liu, X.; Smelyanskiy, M.; Hammond, J. R. Parallel scalability of Hartree−Fock calculations. *J. Chem. Phys.* **2015**, *142*, 104103.
(42) Chow, E.; Liu, X.; Misra, S.; Dukhan, M.; Smelyanskiy, M.; Hammond, J. R.; Du, Y.; Liao, X.-K.; Dubey, P. Scaling up Hartree−Fock calculations on Tianhe-2. *International Journal of High Performance Computing Applications* **2016**, *30*, 85−102.
(43) Finkelstein, J.; Negre, C. F. A.; Fattebert, J.-L. A fast, dense Chebyshev solver for electronic structure on GPUs. *J. Chem. Phys.* **2023**, *159*, 101101.
(44) Pederson, R.; Kozlowski, J.; Song, R.; Beall, J.; Ganahl, M.; Hauru, M.; Lewis, A. G. M.; Yao, Y.; Mallick, S. B.; Blum, V.; Vidal, G. Large Scale Quantum Chemistry with Tensor Processing Units. *J. Chem. Theory Comput.* **2023**, *19*, 25−32.
(45) Finkelstein, J.; Smith, J. S.; Mniszewski, S. M.; Barros, K.; Negre, C. F. A.; Rubensson, E. H.; Niklasson, A. M. N. Mixed Precision Fermi-Operator Expansion on Tensor Cores from a Machine Learning Perspective. *J. Chem. Theory Comput.* **2021**, *17*, 2256−2265.
(46) Finkelstein, J.; Smith, J. S.; Mniszewski, S. M.; Barros, K.; Negre, C. F. A.; Rubensson, E. H.; Niklasson, A. M. N. Quantum-Based Molecular Dynamics Simulations Using Tensor Cores. *J. Chem. Theory Comput.* **2021**, *17*, 6180−6192.
(47) Finkelstein, J.; Rubensson, E. H.; Mniszewski, S. M.; Negre, C. F. A.; Niklasson, A. M. N. Quantum Perturbation Theory Using Tensor Cores and a Deep Neural Network. *J. Chem. Theory Comput.* **2022**, *18*, 4255−4268.
(48) Habib, A.; Finkelstein, J.; Niklasson, A. M. N. Efficient Mixed-Precision Matrix Factorization of the Inverse Overlap Matrix in Electronic Structure Calculations with AI-Hardware and GPUs. *J. Chem. Theory Comput.* **2024**, *20*, 7102−7112.
(49) Aarons, J.; Sarwar, M.; Thompsett, D.; Skylaris, C.-K. Perspective: Methods for large-scale density functional calculations on metallic systems. *J. Chem. Phys.* **2016**, *145*, 220901.
(50) Aarons, J.; Skylaris, C.-K. Electronic annealing Fermi operator expansion for DFT calculations on metallic systems. *J. Chem. Phys.* **2018**, *148*, No. 074107.
(51) Mniszewski, S. M.; Perriot, R.; Rubensson, E. H.; Negre, C. F. A.; Cawkwell, M. J.; Niklasson, A. M. N. Linear Scaling Pseudo Fermi-Operator Expansion for Fractional Occupation. *J. Chem. Theory Comput.* **2019**, *15*, 190−200.
(52) Leamer, J. M.; Dawson, W.; Bondar, D. I. Positivity preserving density matrix minimization at finite temperatures via square root. *J. Chem. Phys.* **2024**, *160*, No. 074107.
(53) Goedecker, S.; Teter, M. Tight-binding electronic-structure calculations and tight-binding molecular dynamics with localized orbitals. *Phys. Rev. B* **1995**, *51*, 9455−9464.
(54) Mukunoki, D.; Ogita, T.; Ozaki, K. Reproducible BLAS Routines with Tunable Accuracy Using Ozaki Scheme for Many-Core Architectures. In *Parallel Processing and Applied Mathematics*; Cham, 2020; pp 516−527.
(55) Ichimura, S.; Katagiri, T.; Ozaki, K.; Ogita, T.; Nagai, T. Threaded Accurate Matrix-Matrix Multiplications with Sparse Matrix-Vector Multiplications. In *2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*; IEEE, 2018; pp 1093−1102.
(56) Mukunoki, D.; Ozaki, K.; Ogita, T.; Imamura, T. DGEMM Using Tensor Cores, and Its Accurate and Reproducible Versions. In *High Performance Computing*; Springer: Cham, 2020; pp 230−248.
(57) Ootomo, H.; Ozaki, K.; Yokota, R. DGEMM on integer matrix multiplication unit. *International Journal of High Performance Computing Applications* **2024**, *38*, 297−313.
(58) Minamihata, A.; Ozaki, K.; Ogita, T.; Oishi, S. Improved extraction scheme for accurate floating-point summation. In *The 35th JSST Annual Conference International Conference on Simulation Technology*; JSST, 2016.
(59) Mukunoki, D.; Ozaki, K.; Ogita, T.; Imamura, T. Accurate Matrix Multiplication on Binary128 Format Accelerated by Ozaki

Scheme. In *Proceedings of the 50th International Conference on Parallel Processing*; ACM: New York, NY, 2021.
(60) Hida, Y.; Li, X. S.; Bailey, D. H. Algorithms for quad-double precision floating point arithmetic. In *Proceedings 15th IEEE Symposium on Computer Arithmetic. ARITH-15 2001*; IEEE, 2001; pp 155−162.
(61) Henry, G.; Tang, P. T. P.; Heinecke, A. Leveraging the bfloat16 Artificial Intelligence Datatype For Higher-Precision Computations. In *2019 IEEE 26th Symposium on Computer Arithmetic (ARITH)*; IEEE, 2019; pp 69−76.
(62) Fousse, L.; Hanrot, G.; Lefèvre, V.; Pélissier, P.; Zimmermann, P. MPFR: A multiple-precision binary floating-point library with correct rounding. *ACM Trans. Math. Softw.* **2007**, *33*, 13−es.
(63) Sun, Q.; Zhang, X.; Banerjee, S.; Bao, P.; Barbry, M.; Blunt, N. S.; Bogdanov, N. A.; Booth, G. H.; Chen, J.; Cui, Z.-H.; Eriksen, J. J.; Gao, Y.; Guo, S.; Hermann, J.; Hermes, M. R.; Koh, K.; Koval, P.; Lehtola, S.; Li, Z.; Liu, J.; Mardirossian, N.; McClain, J. D.; Motta, M.; Mussard, B.; Pham, H. Q.; Pulkin, A.; Purwanto, W.; Robinson, P. J.; Ronca, E.; Sayfutyarova, E. R.; Scheurer, M.; Schurkus, H. F.; Smith, J. E. T.; Sun, C.; Sun, S.-N.; Upadhyay, S.; Wagner, L. K.; Wang, X.; White, A.; Whitfield, J. D.; Williamson, M. J.; Wouters, S.; Yang, J.; Yu, J. M.; Zhu, T.; Berkelbach, T. C.; Sharma, S.; Sokolov, A. Y.; Chan, G. K.-L. Recent developments in the PySCF program package. *J. Chem. Phys.* **2020**, *153*, No. 024109.
(64) Ratcliff, L. E.; Dawson, W.; Fisicaro, G.; Caliste, D.; Mohr, S.; Degomme, A.; Videau, B.; Cristiglio, V.; Stella, M.; D'Alessandro, M.; Goedecker, S.; Nakajima, T.; Deutsch, T.; Genovese, L. Flexibilities of wavelets as a computational basis set for large-scale electronic structure calculations. *J. Chem. Phys.* **2020**, *152*, 194110.
(65) Willand, A.; Kvashnin, Y. O.; Genovese, L.; Vázquez-Mayagoitia, Á.; Deb, A. K.; Sadeghi, A.; Deutsch, T.; Goedecker, S. Norm-conserving pseudopotentials with chemical accuracy compared to all-electron calculations. *J. Chem. Phys.* **2013**, *138*, 104109.
(66) Mohr, S.; Ratcliff, L. E.; Boulanger, P.; Genovese, L.; Caliste, D.; Deutsch, T.; Goedecker, S. Daubechies wavelets for linear scaling density functional theory. *J. Chem. Phys.* **2014**, *140*, 204110.
(67) Jensen, F. Unifying General and Segmented Contracted Basis Sets. Segmented Polarization Consistent Basis Sets. *J. Chem. Theory Comput.* **2014**, *10*, 1074−1085.
(68) Perdew, J. P.; Burke, K.; Ernzerhof, M. Generalized Gradient Approximation Made Simple. *Phys. Rev. Lett.* **1996**, *77*, 3865−3868.
(69) Stephens, P. J.; Devlin, F. J.; Chabalowski, C. F.; Frisch, M. J. Ab Initio Calculation of Vibrational Absorption and Circular Dichroism Spectra Using Density Functional Force Fields. *J. Phys. Chem.* **1994**, *98*, 11623−11627.
(70) Pulay, P. Improved SCF convergence acceleration. *J. Comput. Chem.* **1982**, *3*, 556−560.
(71) Nakajima, T.; Katouda, M.; Kamiya, M.; Nakatsuka, Y. NTChem: A high-performance software package for quantum molecular simulation. *Int. J. Quantum Chem.* **2015**, *115*, 349−359.
(72) Dawson, W.; Kawashima, E.; Ratcliff, L. E.; Kamiya, M.; Genovese, L.; Nakajima, T. Complexity reduction in density functional theory: Locality in space and energy. *J. Chem. Phys.* **2023**, *158*, 164114.
(73) Kruchinina, A.; Rudberg, E.; Rubensson, E. H. Parameterless Stopping Criteria for Recursive Density Matrix Expansions. *J. Chem. Theory Comput.* **2016**, *12*, 5788−5802.
(74) Uchino, Y. Study on Reliable Algorithms for Eigenvalue and Singular Value Decomposition and Matrix Multiplication. Ph.D. thesis, Shibaura Institute of Technology, 2024.
(75) Borštnik, U.; VandeVondele, J.; Weber, V.; Hutter, J. Sparse matrix multiplication: The distributed block-compressed sparse row library. *Parallel Computing* **2014**, *40*, 47−58.
(76) Mohr, S.; Dawson, W.; Wagner, M.; Caliste, D.; Nakajima, T.; Genovese, L. Efficient Computation of Sparse Matrix Functions for Large-Scale Electronic Structure Calculations: The CheSS Library. *J. Chem. Theory Comput.* **2017**, *13*, 4684−4698.
(77) Bock, N.; Negre, C. F. A.; Mniszewski, S. M.; Mohd-Yusof, J.; Aradi, B.; Fattebert, J.-L.; Osei-Kuffuor, D.; Germann, T. C.; Niklasson, A. M. N. The basic matrix library (BML) for quantum chemistry. *Journal of Supercomputing* **2018**, *74*, 6201−6219.
(78) Rubensson, E. H.; Rudberg, E.; Kruchinina, A.; Artemov, A. G. The Chunks and Tasks Matrix Library. *SoftwareX* **2022**, *19*, No. 101159.
(79) Nakata, M. *MPLAPACK version 2.0.1 user manual*; MPLAPACK, 2022.
(80) Knizia, G.; Li, W.; Simon, S.; Werner, H.-J. Determining the Numerical Stability of Quantum Chemistry Algorithms. *J. Chem. Theory Comput.* **2011**, *7*, 2387−2398.
(81) Chatelain, Y.; Castro, P. D. O.; Petit, E.; Defour, D.; Bieder, J.; Torrent, M. VeriTracer: Context-enriched tracer for floating-point arithmetic analysis. In *2018 IEEE 25th Symposium on Computer Arithmetic (ARITH)*; IEEE, 2018; pp 61−68.
(82) Gavini, V.; Baroni, S.; Blum, V.; Bowler, D. R.; Buccheri, A.; Chelikowsky, J. R.; Das, S.; Dawson, W.; Delugas, P.; Dogan, M.; Draxl, C.; Galli, G.; Genovese, L.; Giannozzi, P.; Giantomassi, M.; Gonze, X.; Govoni, M.; Gygi, F.; Gulans, A.; Herbert, J. M.; Kokott, S.; Kühne, T. D.; Liou, K.-H.; Miyazaki, T.; Motamarri, P.; Nakata, A.; Pask, J. E.; Plessl, C.; Ratcliff, L. E.; Richard, R. M.; Rossi, M.; Schade, R.; Scheffler, M.; Schütt, O.; Suryanarayana, P.; Torrent, M.; Truflandier, L.; Windus, T. L.; Xu, Q.; Yu, V. W.-Z.; Perez, D. Roadmap on electronic structure codes in the exascale era. *Modell. Simul. Mater. Sci. Eng.* **2023**, *31*, No. 063301.

https://doi.org/10.1021/acs.jctc.4c00938
J. Chem. Theory Comput. 2024, 20, 10826−10837