Fast principal minor algorithms for diagrammatic Monte Carlo

Fedor Šimkovic IV * and Michel Ferrero
CPHT, CNRS, École Polytechnique, Institut Polytechnique de Paris, Route de Saclay, 91128 Palaiseau, France and Collège de France, 11 place Marcelin Berthelot, 75005 Paris, France

(Received 29 July 2021; revised 9 December 2021; accepted 3 January 2022; published 3 March 2022)

The computation of determinants plays a central role in diagrammatic Monte Carlo algorithms for strongly correlated systems. The evaluation of large numbers of determinants can often be the limiting computational factor determining the number of attainable diagrammatic expansion orders. In this work we build on the algorithm presented in [K. Griffin and M. J. Tsatsomeros, Principal minors, part i: A method for computing all the principal minors of a matrix, Linear Algebra Appl. **419**, 107 (2006)] which computes all principal minors of a matrix in $O(2^n)$ operations. We present multiple generalizations of the algorithm to the efficient evaluation of certain subsets of all principal minors with immediate applications to connected determinant diagrammatic Monte Carlo within the normal and symmetry-broken phases as well as continuous-time quantum Monte Carlo. Additionally, we improve the asymptotic scaling of diagrammatic Monte Carlo formulated in real time to $O(2^n)$ and report speedups of up to a factor 25 at computationally realistic expansion orders. We further show that all permanent principal minors, corresponding to sums of bosonic Feynman diagrams, can be computed in $O(3^n)$, thus encouraging the investigation of bosonic and mixed systems by means of diagrammatic Monte Carlo.

DOI: 10.1103/PhysRevB.105.125104

I. INTRODUCTION

Determinants are ubiquitous in the realm of mathematics and physics [1,2]. In particular, they play a crucial role in a variety of state-of-the-art numerical algorithms which compute the physical properties of fermionic many-body systems [3–9]. While a single determinant can be computed in polynomial time [1], many algorithms require the computation of a large number of principal minors [10] of a given matrix. In most cases, this results in exponential asymptotic scaling and often represents the computational bottleneck of the respective algorithms.

Diagrammatic Monte Carlo algorithms are based on the idea of expressing physical properties of interest as a perturbation series that is evaluated stochastically. A great asset of such algorithms is that they can directly treat systems in the thermodynamic limit (infinite system size) by computing exclusively connected Feynman diagrams. In early algorithmic implementations [11,12], such connected Feynman diagrams were sampled individually in order to compute the coefficients of the relevant perturbation series. These methods have found many successful applications, such as the polaron [11], Fermi gas [13], electron gas [14], spin systems [15], and the Fermi-Hubbard model [16–19]. Despite these triumphs, such algorithms eventually suffer from the infamous fermion sign problem, which leads to increased statistical variance as a result of a factorially growing number of sign-alternating diagrams with increasing expansion order. This difficulty has been overcome by a number of algorithms that efficiently compute sums of factorial numbers of diagrams by grouping them into determinants.

Determinants have first made their way into path-integral-based quantum Monte Carlo methods in the continuous-time quantum Monte Carlo algorithms (CTQMC) [4,20] and the related determinant diagrammatic Monte Carlo (DDMC) [5]. In these algorithms of polynomial complexity, the use of determinants strongly reduces the fermion sign problem by explicitly computing the sum of all connected and disconnected Feynman diagrams up to very high expansion orders ($\sim$1000). CTQMC algorithms have proven especially useful in solving the quantum impurity models within dynamical mean-field theory and its extensions [20,21], while DDMC has been successfully applied to study superfluid and magnetic transitions in the three-dimensional attractive [5] and repulsive Fermi-Hubbard models [22] at half filling. The presence of disconnected diagrams in the sums generated by determinants, however, has as a consequence in that the Monte Carlo variance grows exponentially with the number of orbitals or lattice sites. This leads to the necessity of extrapolating with system size in order to reach the thermodynamic limit and is another manifestation of the sign problem.

More recently, the connected determinant diagrammatic Monte Carlo (CDet) [6] has been introduced. Just like the early diagrammatic Monte Carlo algorithms, it is formulated directly in the thermodynamic limit. However, instead of sampling individual diagrams, it evaluates the sum of the full factorial number of connected diagrams at a given expansion order at only exponential cost [$O(3^n)$] by recursively subtracting all disconnected diagrams from the sums generated with determinants. In order to be able to apply the recursion relations it is necessary to compute all principal minors of the originally evaluated matrix at each Monte Carlo step. A naive implementation of the principal minor computation scales as $O(n^3 2^n)$ and, despite being asymptotically favorable in comparison with the recursion

*fsimkovic@gmail.com

step $[O(3^n)]$, it constitutes the bottleneck at all realistically attainable expansion orders $n \leqslant 15$ (corresponding to matrix size). In Ref. [23] an algorithm for the simultaneous computation of all principal minors of a given matrix was presented, scaling more favorably as $O(2^n)$. This fast principal minor algorithm (FPM) has been used in most recent publications based on CDet computations studying the square Fermi-Hubbard model at half filling [24–27] and finite doping [28,29] as well as the half-filled triangular Fermi-Hubbard model [30].

Recently, the CDet algorithm was generalized to perturbative expansions around symmetry-broken starting points inside of superfluid phases in the attractive three-dimensional Fermi-Hubbard model [31] using the Nambu formalism. For such computations, the need arises to compute a certain subset of 2^n principal minors of a $2n \times 2n$ matrix.

A determinant-based diagrammatic Monte Carlo algorithm has also been introduced for the real-time evolution of quantum systems [9,32–35]. In these algorithms, the computation of a contribution at perturbation order n requires to compute the sum of 2^n determinants of $n \times n$ matrices, corresponding to a sum over Keldysh indices. Current state-of-the-art implementations compute this sum in $O(n^2 2^n)$, which defines their asymptotic scaling.

For Feynman diagrammatic computations in bosonic systems, the CDet recursion equally allows one to extract all connected diagrams from sums generated by the evaluation of matrix permanents. Such an approach would be viable provided the permanents of all principal submatrices can been obtained with reasonable computational effort. However, the computation of a permanent scales exponentially as $O(n2^n)$ [36,37], compared with the polynomial $O(n^3)$ scaling for determinants. This begs the question of whether the application of CDet to bosonic systems is computationally feasible and how such algorithms scale asymptotically.

In this paper, we introduce multiple generalizations of the FPM algorithm with useful applications to diagrammatic Monte Carlo implementations, as described above. In particular, we show that the computational scaling of the real-time diagrammatic Monte Carlo within the Keldysh formalism can be improved to $O(2^n)$. We further show that the same scaling also applies to the computation of principal minors in CDet within symmetry-broken phases. We also present an $O(3^n)$ CDet generalization to diagrammatic expansions that involve two vertex types. Finally, we show that summing all connected diagrams for bosonic and mixed (Fermi-Bose) systems can have the same computational scaling as for purely fermionic ones, $O(3^n)$.

The paper is structured as follows: In Sec. II we introduce the original FPM algorithm as well as its generalizations in Sec. III and introduce a principal minor algorithm for permanents in Sec. IV. In Sec. V we provide details of specific diagrammatic Monte Carlo application before discussing conclusions in Sec. VI.

II. FPM ALGORITHM

Our aim is to compute all principal minors of a given $n \times n$ square matrix $A[\mathcal{S}]$, where $\mathcal{S} = \{1, 2, \ldots, n\}$ is a set of rows and columns. A principal minor $A[\mathcal{S}_i]$, corresponding to a subset $\mathcal{S}_i \subseteq \mathcal{S}$, is the determinant of the submatrix generated by keeping only the rows and columns specified by $\mathcal{S}_i$. We denote by $\bar{\mathcal{S}}_i$ the complement of $\mathcal{S}_i$ with respect to the full set $\bar{\mathcal{S}}_i \equiv \mathcal{S}/\mathcal{S}_i$. We further denote nonsquare matrices as $A[\mathcal{S}_i, \mathcal{S}_j]$ and we have $A[\mathcal{S}, \mathcal{S}] \equiv A[\mathcal{S}]$. Following standard practice, the determinant of an empty matrix is defined as $\det(A[\emptyset]) = 1$. In the following, we will always associate $\mathcal{S}_i \subseteq \mathcal{S}$ with the subset described by the binary bit-mask provided by the integer i (e.g., $\mathcal{S}_{13} = \{1, 3, 4\}$).

A trivial way to compute all principal minors is to generate each $k \times k$-sized submatrix individually and simply compute the determinant by means of an algorithm with polynomial $\mathcal{O}(k^3)$ complexity, e.g., Gaussian elimination or LU decomposition [38]. The total number of operations needed to compute all principal minors this way scales exponentially as $\mathcal{O}(n^3 2^n)$ and requires polynomial space $[O(n^2)]$.

This computational scaling can often be reduced to $\mathcal{O}(n^2 2^n)$ by making use of Sherman–Morrison–Woodbury type formulas [39] and changing determinants by one row and column at a time using a Gray code.

A computationally superior algorithm scaling as $O(2^n)$ was presented in Ref. [23]. Below we provide the main ideas of the algorithm without the mathematically rigorous derivations that can be found in the original publication and references therein [40–43].

We start from the well-known relation [41]:

$$\det(A[\mathcal{S}]) = \det(A[\mathcal{S}_i]) \det(A[\mathcal{S}]/A[\mathcal{S}_i]), \tag{1}$$

where we have introduced the Schur complement, which we define as:

$$A[\mathcal{S}]/A[\mathcal{S}_i] \equiv A[\bar{\mathcal{S}}_i] - A[\bar{\mathcal{S}}_i, \mathcal{S}_i] A[\mathcal{S}_i]^{-1} A[\mathcal{S}_i, \bar{\mathcal{S}}_i]. \tag{2}$$

In order to proceed we need two identities. It should be noted that both relations only hold for nonsingular matrices and nonsingular principal minors thereof. The first identity shows that eliminating selected rows and columns can be done either before or after taking the Schur complement without affecting the computation of the determinant:

$$\det(A[\mathcal{S}_j \cup \mathcal{S}_k]) = \det(A[\mathcal{S}_j]) \det((A[\mathcal{S}]/A[\mathcal{S}_j])[\mathcal{S}_k]), \tag{3}$$

where $\mathcal{S}_k \subseteq \bar{\mathcal{S}}_j$. The second identity, called the quotient property, is necessary to treat nested Schur complementation:

$$A[\mathcal{S}]/A[\mathcal{S}_j \cup \mathcal{S}_k] = (A[\mathcal{S}]/A[\mathcal{S}_j])/((A[\mathcal{S}]/A[\mathcal{S}_j])[\mathcal{S}_k]). \tag{4}$$

This means that instead of taking one Schur complement with respect to a large submatrix one can successively take multiple Schur complements with respect to single matrix elements.

The algorithm presented in Ref. [23] generates a binary tree with $2^n - 1$ nodes that are used to compute the different principal minors of the matrix $A[\mathcal{S}]$. Each node is labeled with an integer beginning with 1 at the first level and enumerating the following nodes in a top to bottom and left to right order as portrayed in Fig. 1. A matrix is associated with every node of the tree, starting with the original full matrix $A[\mathcal{S}] \equiv A^{(1)}$ for the first node $i = 1$ at level $l = 1$. Whenever we take the left branch from a node i at level l we generate a new submatrix $A^{(i+2^{l-1})}$ at the node $(i + 2^{l-1})$ by removing the first row and column of $A^{(i)}$. Whenever we take the right branch we

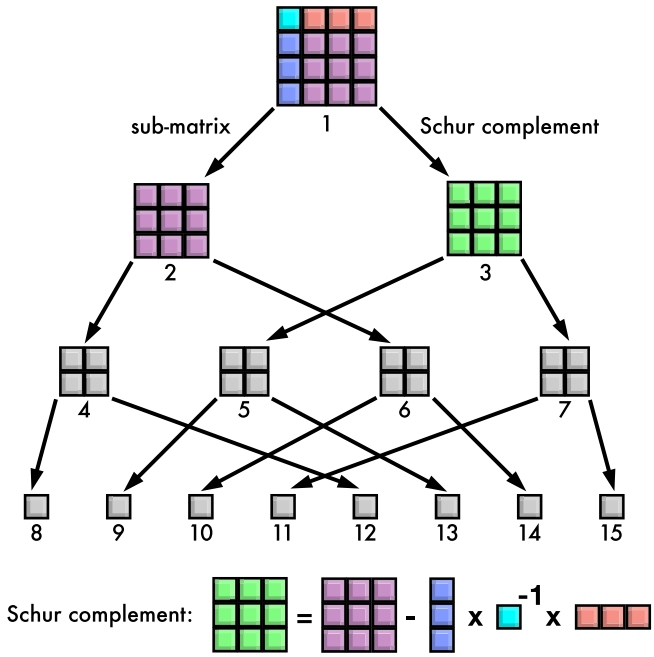

FIG. 1. The binary tree corresponding to the algorithm for the computation of principal minors of a matrix of size 4×4. From each node, the submatrix with the first row and column removed is copied along the left branch. The Schur complement with respect to the (1,1) element is taken along the right branch.

generate a matrix $A^{(i+2^l)}$ at the node $(i+2^l)$ from the Schur complement of $A^{(i)}$ with respect to its (1,1) element:

$$A^{(i+2^{l-1})} = A^{(i)}[\{l+1,\ldots,n\}] \qquad \text{left branch} \tag{5}$$

$$A^{(i+2^l)} = A^{(i)}/A^{(i)}[\{l\}] \qquad \text{right branch.} \tag{6}$$

As can be seen from the procedure above, the integer that labels a node in the tree can be interpreted as a binary mask keeping track of which columns and rows have been retained or eliminated before reaching that node. One can then compute the principal minor associated with the node i at level l by using Eq. (3):

$$\begin{aligned}\det(A[\mathcal{S}_i]) &\equiv \det(A[\mathcal{S}_j\cup\{l\}])\\ &= \det(A[\mathcal{S}_j])\,\det(A[\mathcal{S}]/A[\mathcal{S}_j])[\{l\}]\\ &= \det(A[\mathcal{S}_j])\,A^{(i)}_{1,1},\end{aligned} \tag{7}$$

where we have introduced $\mathcal{S}_j \equiv \mathcal{S}_i/\{l\}$. Applying Eq. (7) to all nodes of the tree eventually yields all principal minors of the matrix $A[\mathcal{S}]$.

Sometimes, there is one additional difficulty that needs to be taken into account. Whenever the Schur complement is taken [as per Eq. (2)] it is necessary to calculate the inverse of a matrix element called the pivot. If the pivot's value is zero or vanishingly small this procedure forcibly results in numerical instabilities of the algorithm. Reference [23] provides a solution for such cases by adding a correction term C to the pivot which must be, however, compensated for at the end of the computation. Any time a pivot is changed this affects all principal minors computed from the resulting Schur complement. After all principal minors have been computed a pivot-correction has to be back-propagated through principal minors for each instance where the correction term was added.

TABLE I. Fast principal minor algorithm

```
for l = 1   l ⩽ n   l++ do
  for i = 2^(l−1)   i < 2^l   i++ do
    det(A[S_i]) ← det(A[S_i/{l}]) A^(i)_{1,1}
    A^(i+2^(l−1)) ← A^(i)[{l+1,...,n}]        ▷ Left branch
    A^(i)_{1,1} += C                          ▷ Pivot correction (optional)
    A^(i+2^l) ← A^(i)/A^(i)[{l}]              ▷ Right branch
```

For example, let $\{l\}$ be a very small (or vanishing) element corresponding to the first row and column of the matrix $A[\mathcal{S}_i]$. To avoid numerical instabilities, we change $A[\mathcal{S}_i]_{1,1} \to A[\mathcal{S}_i]_{1,1}+C$ and use this modified matrix $\tilde{A}[\mathcal{S}_i]$ to compute the Schur complement. In order to recover the determinant of the original matrix $A[\mathcal{S}_i]$ one can use the formula:

$$\det(A[\mathcal{S}_i]) = \det(\tilde{A}[\mathcal{S}_i]) - C\det(A[\mathcal{S}_i/\{l\}]), \tag{8}$$

which is how the pivot-correction is back-propagated at the end of the computation. The minimal number of pivot-corrections that need to be performed for an $n\times n$ matrix with all diagonal entries being zero, as is often the case in diagrammatic Monte Carlo algorithms [4–6], is $n-1$, which results in additional computational cost of $O(2^n)$ for the back-propagation loop. If a pivot-correction would be performed at every instance of Schur complementation, the resulting computational cost would become $O(n2^n)$ instead.

Table I shows a sketch of the FPM algorithm. The computational scaling of the algorithm is easily found to be $\sum_{l=0}^{n-2} 2^l\cdot 2(n-l-1)^2 \approx O(2^n)$ operations. Reference [23] provides a breadth-first-traversal algorithm to calculate all principal minors, as summarized in Table I. In this formulation all matrices computed at a particular level of the tree need to be stored, which leads to exponential memory costs of $O(2^n)$. It is possible, however, to traverse the tree depth-first or formulate a recursive version of the algorithm in order to reduce the scaling to the polynomial $O(n^3)$ as only one matrix needs to be stored at each level. This can lead to significant accelerations for larger matrices, overcoming the slowdown reported in Ref. [23].

III. GENERALIZATIONS TO PRINCIPAL MINOR SUBSETS

In this subsection we want to address cases when it is desired to compute only a certain subset of all principal minors of a matrix.

A. Leading principal minors

It can be straightforwardly seen that it is possible to use successive Schur complementation to compute the principal minor corresponding to the determinant of the full $n\times n$ matrix in $O(n^3)$ operations. The disadvantage with respect to other more established algorithms, such as Gaussian elimination, is that there is no direct way to implement pivot corrections in the way it is done for the FPM algorithm. This is due to the necessity of subtracting principal minors at the back-propagation step, which have not been computed in the process. On the

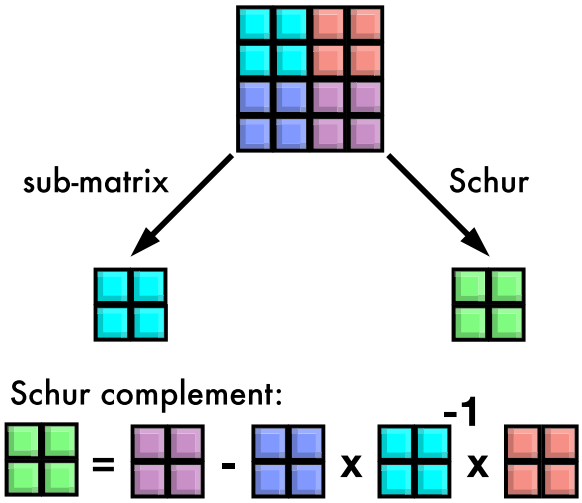

FIG. 2. Binary tree for used in the algorithm for the computation of principal minors of a matrix of size 4×4, corresponding to submatrices respecting the XNOR case rules.

other hand, this algorithm has the advantage that it computes all *leading* principal minors corresponding to subsets corresponding to indices 2^l-1 with $1\leqslant l\leqslant n$ in the process.

B. XNOR principal minors of a $2n\times2n$ matrix

Now let us consider a $2n\times2n$ matrix $A[\mathcal{S}]$. Computing all principal minors of $A[\mathcal{S}]$ with the FPM algorithm takes $O(4^n)$ operations, proportional to the total number of principal minors, 4^n-1. In some cases it is, however, sufficient to compute only a certain subset of minors. Specifically, if we treat $A[\mathcal{S}]$ as an $n\times n$ block matrix with 2×2 blocks then some cases are of particular interest. The first one is: Compute principal minors of all subsets $\mathcal{S}_i$ such that for all $0\leqslant l<n$ we satisfy one of the following two options:

(1) $\{2l+1\}\in\mathcal{S}_i$ and $\{2l+2\}\in\mathcal{S}_i$
(2) $\{2l+1\}\notin\mathcal{S}_i$ and $\{2l+2\}\notin\mathcal{S}_i$.

We will call this set of options the *XNOR case*. In the language of matrices this means that either both the $(2l+1)$-th and the $(2l+2)$-th rows and columns are included in a submatrix or none of the two. One must then evaluate a total of 2^n-1 principal minors, which can be done naively in $O(n^3 2^n)$ by computing them separately. It is readily seen that one can form a binary tree with 2^n-1 nodes, corresponding to the principal minors of interest, by removing two rows and columns from a matrix when taking the left branch and by performing the corresponding Schur complement whenever taking the right branch; see Fig. 2. This procedure computes all the relevant ratios of principal minors of submatrices adhering by the aforementioned rules. Since the pivot, of which we must compute the inverse, has become a 2×2 matrix and since we compute only a subset of the principal minors of the original matrix, it is no longer possible to add corrections to pivots. The algorithm scales as $O(2^n)$ and is roughly four times slower than FPM for an $n\times n$ matrix due to the fact that the Schur complement is taken with respect to 2×2 matrices instead of a single element.

It is possible to extend this idea to $kn\times kn$-sized matrices with *steps* of size k resulting in asymptotic scaling of $O(k^2 2^n)$.

C. NAND principal minors of a $2n\times2n$ matrix

In the second case we are interested in computing all principal minors such that for all $0\leqslant l<n$ we satisfy one of the following three options:

(1) $\{2l+1\}\in\mathcal{S}_i$ and $\{2l+2\}\notin\mathcal{S}_i$
(2) $\{2l+1\}\notin\mathcal{S}_i$ and $\{2l+2\}\in\mathcal{S}_i$
(3) $\{2l+1\}\notin\mathcal{S}_i$ and $\{2l+2\}\notin\mathcal{S}_i$.

We will call this the *NAND case*. In matrix terms this means that either the $(2l+1)$-th or the $(2l+2)$-th row and column are included in a submatrix of interest or none of the two. The total number of corresponding principal minors is 3^n-1. Here, it is possible to transform the original $2n\times2n$-sized block-matrix $A[\mathcal{S}]$ into an $n\times n$-sized matrix $\hat{A}[\mathcal{S}]$ with polynomial entries in $\alpha_i^{\pm}$ such that

$$\hat{A}_{i,j}=A_{2i+1,2j+1}\,\alpha_i^+\alpha_j^+ + A_{2i+2,2j+1}\,\alpha_i^-\alpha_j^+ + A_{2i+1,2j+2}\,\alpha_i^+\alpha_j^- + A_{2i+2,2j+2}\,\alpha_i^-\alpha_j^- \tag{9}$$

with the aforementioned rules determining which principal minors need to be evaluated being encoded by:

$$\alpha_i^+\alpha_i^-=0. \tag{10}$$

One can then proceed to evaluate the $n\times n$ polynomial-valued matrix $\hat{A}[\mathcal{S}]$ with the FPM algorithm. This means that pivot corrections can be added in the same way as they would be in the FPM algorithm, as long as they are back-propagated at the end of the calculation. An alternative way of looking at the new representation is as the rearrangement of 3^n-1 relevant nodes from the original binary tree of depth $2n$ into a ternary tree of depth n; see Fig. 3. Then, the *left* branch is obtained by copying a submatrix eliminating the first two rows and columns. Two additional, *right* branches are obtained as the result of taking the Schur complement with respect to the (1,1) and (2,2) elements of the matrix at the given node. The total computational cost of the algorithm is governed by the number of Schur complements taken at each level of the ternary tree $\sum_{l=0}^{n-2}3^l\cdot4(n-l-1)^2\approx O(3^n)$.

It is straightforward to extend this algorithm to $kn\times kn$-sized matrices resulting in asymptotic scaling of $O([k+1]^n)$, related to the the number of relevant principal minors: $(k+1)^n-1$.

D. XOR principal minors of a $2n\times2n$ matrix

In the third case we are interested in computing all principal minors such that for all $0\leqslant l<n$ we satisfy one of the following two options:

(1) $\{2l+1\}\in\mathcal{S}_i$ and $\{2l+2\}\notin\mathcal{S}_i$
(2) $\{2l+1\}\notin\mathcal{S}_i$ and $\{2l+2\}\in\mathcal{S}_i$.

We will call this the *XOR case*. This is equivalent to only computing principal minors of $n\times n$ sized submatrices of $\hat{A}[\mathcal{S}]$ and adhering by the rules of the NAND case. One can resort to evaluating only the two *right*, Schur-complement branches in the ternary tree of Fig. 3, which effectively reduces it to a binary tree. This algorithm can be shown to lead to a reduced scaling of $O(2^n)$. As in the case of computing a single

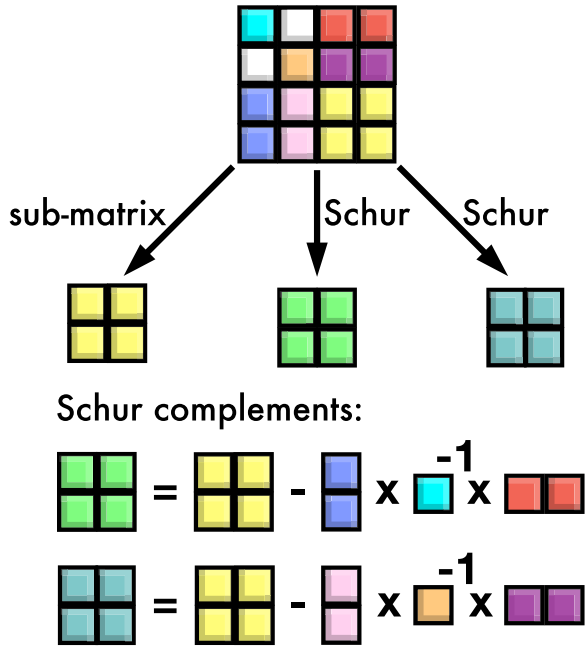

FIG. 3. Binary tree corresponding to the algorithm for the computation of principal minors of a matrix of size 4×4, corresponding to submatrices respecting the NAND case rules. Note that off-diagonal matrix $A[\mathcal{S}]_{i,j}$ elements with $|i-j|=1$ (white) do not ever enter the calculations

determinant using Schur complements, this has the advantage of computing lower-order principal minors as a side product. The downside, similarly to the single determinant case, is that this algorithm doesn't allow for pivot corrections without significant additional computational cost.

Similarly to the previous two cases, one can generalize this algorithm to $kn\times kn$-sized matrices resulting in asymptotic scaling of $O(k^n)$.

IV. PERMANENT PRINCIPAL MINOR ALGORITHM

While the determinant of an $n\times n$-sized matrix can be computed in polynomial time $O(n^3)$, the two most efficient algorithms to compute the permanent of the same matrix, found by Ryser and Glynn, scale exponentially as $O(n2^n)$ [36,37,44]. The inclusion-exclusion Ryser formula for the computation of a permanent reads:

$$\mathrm{per}(A)=\sum_{\mathcal{S}_k\subseteq\{1,\ldots,n\}}(-1)^{|\mathcal{S}_k|}\prod_{i=1}^{n}\sum_{j\in\mathcal{S}_k}A_{i,j}. \tag{11}$$

A numerical implementation hereof consists of two steps. First, one precomputes all possible sums over index j given by $\sum_{j\in\mathcal{S}_k}A_{i,j}$ in Gray code order with $O(n2^n)$ additions. Subsequently, one evaluates the products over sums for all 2^n-1 subsets $\mathcal{S}_k\subseteq\{1,\ldots,n\}$ using $O(n2^n)$ multiplications which defines the asymptotic scaling of the algorithm. As a consequence, a naive successive application of the Ryser algorithm to compute all permanent principal minors of a matrix results in a scaling of $O(n3^n)$.

We can, however, reduce this scaling by generalizing the algorithm to the case when all permanent principal minors of a matrix are desired. Let us rewrite the formula of Eq. (11) for a given subset (minor) $\mathcal{S}\subseteq\{1,\ldots,n\}$ as

$$\mathrm{per}(A[\mathcal{S}])=\sum_{\mathcal{S}_k\subseteq\mathcal{S}}(-1)^{|\mathcal{S}_k|}\prod_{i\in\mathcal{S}}\sum_{j\in\mathcal{S}_k}A_{i,j}. \tag{12}$$

Here, the first (additions) step remains identical as all possible sums are already being computed in the computation of the full matrix permanent. For the second step it is necessary to evaluate all products corresponding to subsets $\mathcal{S}$, given as:

$$\Pi[\mathcal{S}]=\prod_{i\in\mathcal{S}}\sum_{j\in\mathcal{S}_k}A_{i,j}. \tag{13}$$

These subsets can be evaluated in Gray code order, which, however, entails the necessity of using divisions. A division-free way of evaluating subsets can be achieved by using a recursion relation

$$\Pi[\mathcal{S}]=\Pi[\mathcal{S}/\{j\}]\sum_{i\in\mathcal{S}_k}A_{i,j}, \tag{14}$$

where $j\in\mathcal{S}$ can be chosen arbitrarily. This corresponds to $\sum_{k=1}^{n}\binom{n}{j}2^k=O(3^n)$ operations, which sets the asymptotic scaling of the algorithm. Remarkably, the step resembles a subset convolution of two sets, which is essentially what is being computed when applying the recursive relation within the CDet algorithm [6]. This means that a fast subset convolution algorithm [45] is applicable, which would reduce the asymptotic scaling further to $O(n^2 2^n)$, albeit that only becomes practically advantageous for matrix sizes of $n\gtrsim 15$.

Interestingly, a related inclusion-exclusion principle-based fast diagram summation algorithm has been found for strong-coupling expansions of fermionic [46] as well as bosonic [47] systems.

V. APPLICATIONS TO DIAGRAMMATIC MONTE CARLO ALGORITHMS

Let us now focus on applications to various numerical algorithms for quantum many-body systems based on diagrammatic Monte Carlo. In such algorithms one is generally interested in computing a physical observable $\mathcal{C}$ expressed in terms of a power series:

$$\mathcal{C}(\xi)=\sum_{n=0}^{\infty}a_n\xi^n, \tag{15}$$

where the coefficients a_n are obtained from the stochastic evaluation of Feynman diagrams, integrated over internal space and imaginary-time variables $X=(\vec{r},\tau)$.

A. CDet

The fast principle minor algorithm has first been used in the context of the CDet [6] algorithm. In this method, the contribution $c[\mathcal{S}]$ to a coefficient a_n in Eq. (15) for a given set $\mathcal{S}=\{X_1,\ldots,X_n\}$ of internal vertices is the sum of all connected diagrams constructed on $\mathcal{S}$. This sum is obtained by a recursion formula that involves the prior computation of all principal minors of a given matrix corresponding to subsets $\mathcal{S}_i\subseteq\mathcal{S}$:

$$a[\mathcal{S}_i]=\det(M_n^{\uparrow}[\mathcal{S}_i])\det(M_n^{\downarrow}[\mathcal{S}_i]), \tag{16}$$

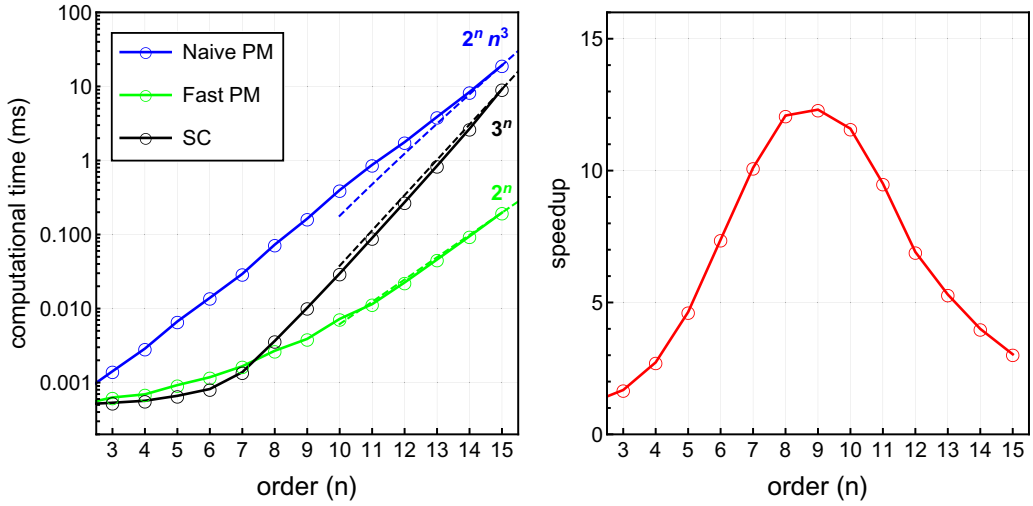

FIG. 4. Left: Computational time as a function of order (n) for the FPM algorithm as compared with the naive determinantal implementation as well the SC used by CDet. Right: Computational speedup of the complete CDet algorithm using FPMs with respect to the algorithm using a naive determinant implementation.

where $(M_n^\sigma)_{i,j} = G_{0\sigma}(X_i - X_j)$. Then, a recursive relation is used to compute sums of connected diagrams $c[\mathcal{S}]$ from $a[\mathcal{S}]$ by subtracting all disconnected ones:

$$c[\mathcal{S}] = a[\mathcal{S}] - \sum_{\substack{\mathcal{S}_i \subsetneq \mathcal{S} \\ \{j\} \in \mathcal{S}_i}} c[\mathcal{S}_i] a[\mathcal{S}/\mathcal{S}_i], \tag{17}$$

where $\{j\} \in \mathcal{S}$ is chosen arbitrarily. The recursive algorithm corresponds to a subset convolution and can be evaluated in $O(3^n)$ computational steps, or alternatively in $O(n^2 2^n)$ by using fast subset convolutions [45] which, however, in practice performs slightly worse for attainable orders $n \lesssim 15$. The total computational time of the algorithm is thus limited by both the computation of principal minors and the subset convolution depending on the expansion order; see Fig. 4. We see that the FPM algorithm significantly outperforms a naive determinant implementation and becomes subleading to the subset convolution around order $n = 8$. The total speedup of the CDet algorithm due to the use of FPM also peaks at a factor 12 around orders $8 \lesssim n \lesssim 10$, which corresponds to the highest attainable orders for calculations in challenging regimes of the Hubbard model. At yet higher orders the gain reduces due to an ever increasing contribution of the subset convolution to the total computational time.

We also note that at order 24 our C++ implementation of FPM algorithm takes 0.23 s on a standard laptop compared with 443 s reported in Ref. [23], roughly corresponding to an improvement by a factor 2000.

B. CDet for expansions around BCS theory

In a recent work [31] the CDet algorithm was generalized to expansions around the BCS mean-field theory inside superfluid phases and using the Nambu formalism [48]. The main algorithmic difference to CDet in the paramagnetic regime is that one needs to evaluate $2n \times 2n$-sized block-matrices with bare Nambu propagators as entries. These matrices are built from 2×2 blocks $\tilde{M}_{i,j}$ given by:

$$\tilde{M}_{i,j} = \begin{pmatrix} G_{00}(X_{ij}) & G_{01}(X_{ij}) \\ G_{10}(X_{ij}) & G_{11}(X_{ij}) \end{pmatrix}, \tag{18}$$

with $i, j \in \{1, \ldots, n\}$ and $X_{ij} \equiv X_i - X_j$. It is necessary to compute $2^n - 1$ principal minors of this matrix, exactly corresponding to the XNOR case from the previous section. The recursion step involving a subset convolution does not differ from the original CDet one. From Fig. 5 we see that, similarly to the original CDet algorithm, a FPM algorithm significantly outperforms the naive determinant implementation. The maximum computational gain of a factor 17 is achieved at order $n = 9$ before slowly decreasing for higher orders due to the computational cost of the subset convolution taking over. Again, the gain is maximal at orders which correspond to the limits of many realistic computations in difficult regimes.

C. CDet for expansions with two vertex types

Another possible generalization of the CDet algorithm is to expansions with two (or more) distinct types of vertices present in diagrams [49–51]. When multiple vertices are present, one way is to pick the type of each vertex insertion in diagrams stochastically. Another option is to sum over all 2^n possible combinations of n vertices with the restriction that each vertex can only correspond to one type within a given diagram. For two vertex types, this corresponds to computing all principal minors of a $2n \times 2n$-sized matrix, as described by the NAND case of the previous section. In Fig. 6 we show the fast and naive (determinant) implementations of such a principal minor algorithm as compared with subset convolution, which again stays unaltered from its original CDet version. We see that even the FPM implementation, despite scaling as $O(3^n)$, is an order of magnitude slower than the subset convolution. The total gain due to using the FPM algorithm grows steadily as a function of order and reaches a factor 40 at $n = 15$. We want to note that another possible application of this algorithm is to

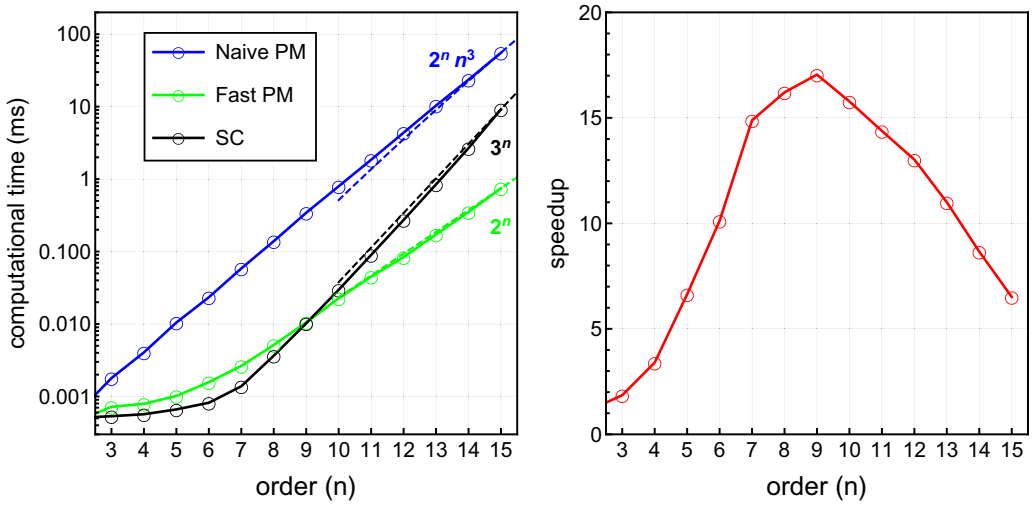

FIG. 5. Left: Computational time as a function of order (n) for the FPM algorithm within CDet in the Nambu formalism as compared with the naive determinantal implementation as well the subset convolution (SC) used by CDet. Right: Computational speedup of the complete Nambu CDet algorithm using FPM with respect to the algorithm using a naive determinant implementation.

artificially generate a second vertex by choosing an arbitrary transformation: $(\vec{r}, \tau) \to (\vec{r}\,', \tau')$. This transformation either can exploit some underlying symmetry which is present in the model or can be chosen stochastically. This strategy can potentially lead to a dramatic reduction of the Monte Carlo variance at the small additional computational cost factor described above. We leave the exploration of this subject to further work.

D. Real-time diagrammatic Monte Carlo

There also exists a class of diagrammatic Monte Carlo algorithms formulated directly in real time [9,32–35] (as opposed to imaginary time for CDet) which are based on the Keldysh formalism [52]. These have the advantage that a summation over all Keldysh indices guarantees the connectivity of Feynman diagrams and therefore no recursive formula needs to be applied in order to compute connected quantities. This means that at order n it is sufficient to evaluate the sum of 2^n determinants of $n \times n$ matrices with elements $M^{\sigma}_{i,j} = G^{\alpha_i \alpha_j}_{0\sigma}(X_i - X_j)$, where $\{\alpha_1, \ldots, \alpha_n\}$ is the set of Keldysh indices. A naive implementation of this sum would require an effort $O(n^3 2^n)$. In current state-of-the-art implementations, this can be brought down to $O(n^2 2^n)$ by using Sherman–Morrison–Woodbury-type updates during a Gray code enumeration of all Keldysh indices. Let us note that while the latter more efficient approach works in most of the cases, it can also sometimes be unstable if (almost) singular matrices are generated during the update process.

A different way to look at this algorithm is to think of the sum over all Keldysh indices as a sum of a well-defined subset of principal minors of an enlarged $2n \times 2n$ matrix built from

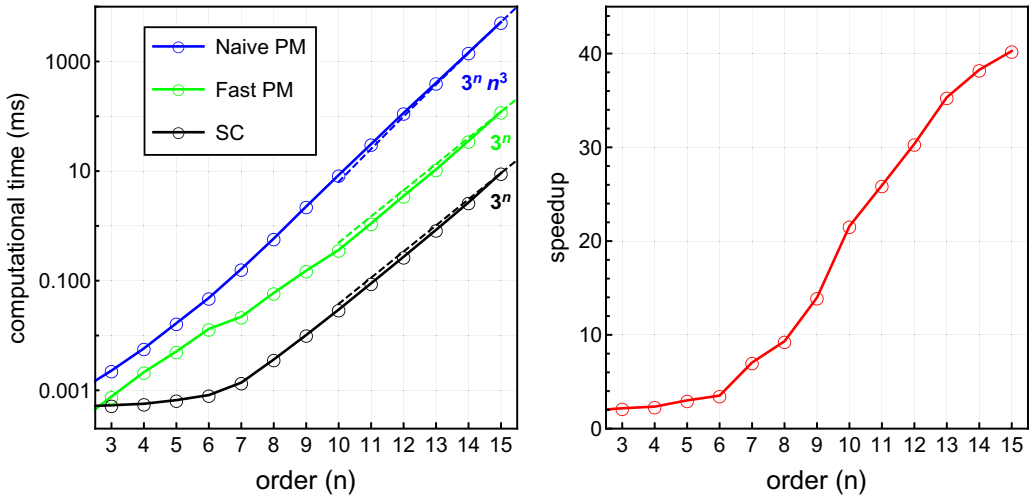

FIG. 6. Left: Computational time as a function of order (n) for the permanent principal minors (PPM) algorithm within CDet symmetrized over 2^n vertices of two vertex types as compared with a naive determinantal implementation as well the subset convolution (SC) used by CDet. Right: Computational speedup of the complete symmetrized CDet algorithm using FPM with respect to the algorithm using a naive determinant implementation.

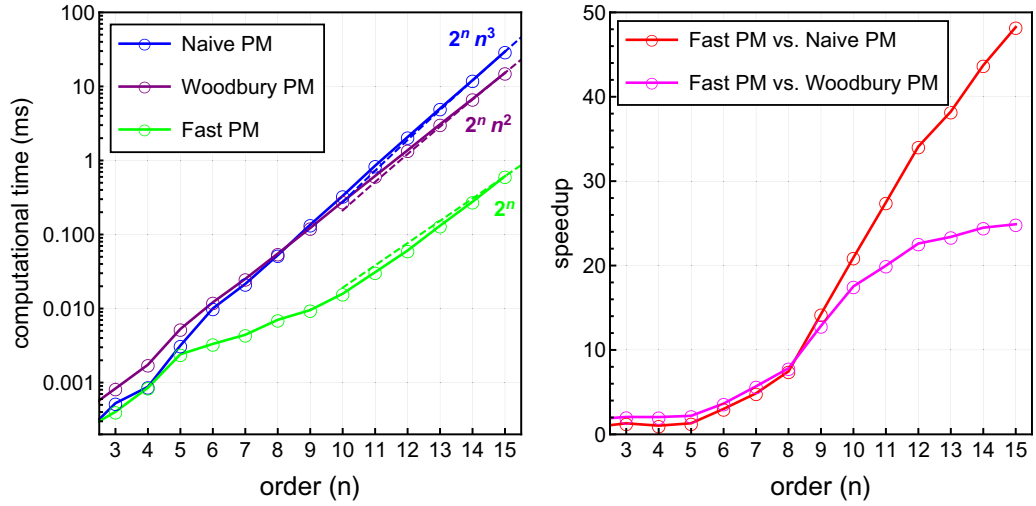

FIG. 7. Left: Computational time as a function of order (n) for the FPM algorithm within real-time diagrammatic Monte Carlo as compared with a naive determinantal implementation as well as an implementation using Sherman–Morrison–Woodbury-type updates and Gray code. Right: Computational speedup of the real-time diagrammatic Monte Carlo algorithm using FPMs with respect to the other two implementations.

2×2 block matrices $\tilde{M}_{i,j}$ given by:

$$\tilde{M}_{i,j} = \begin{pmatrix} G_0^{++}(X_{ij}) & G_0^{+-}(X_{ij}) \\ G_0^{-+}(X_{ij}) & G_0^{--}(X_{ij}) \end{pmatrix}, \tag{19}$$

with $i, j \in \{1, \ldots, n\}$ and $X_{ij} \equiv X_i - X_j$. The sum over all Keldysh indices is then nothing but the sum over all $2^n - 1$ principal minors corresponding to $n \times n$-sized submatrices described by the XOR case above and can be achieved in $O(2^n)$. It should be noted that, similar to the case of computing the determinant via Schur complementation, lower-order diagrams are being computed as a side product by this algorithm and could potentially be used to improve the Monte Carlo variance by using, e.g., the many-configurations Monte Carlo algorithm of Ref. [53] as well as the use of conformal maps applied to coefficients at each Monte Carlo step [54]. They could also be used to compute one-particle irreducible quantities, such as self-energy, where a recursion involving all principal minors must be computed.

In Fig. 7, we compare the scaling of the FPM against the naive (determinant) and Sherman-Morrison-Woodbury-type algorithms. We observe a steadily increasing speedup as a function of order, reaching about a factor 50 and 25, respectively, at order $n = 15$.

E. CTINT, DDMC, and PDet

In the context of the CTINT [4] and DDMC [5] algorithms, it has been shown that for fermionic systems, such as the Fermi-Hubbard model [55], all possible (connected and disconnected) Feynman diagram graph topologies contributing to the partition function can be generated as the product of two determinants (one per spin-type: $\{\uparrow, \downarrow\}$) of matrices with entries corresponding to bare Green's functions.

$$a_n \sim \int_{X_1 \ldots X_n} \det(M_n^{\uparrow}(X_1 \ldots X_n)) \det(M_n^{\downarrow}(X_1 \ldots X_n)), \tag{20}$$

where $(M_n^{\sigma})_{i,j} = G_{0\sigma}(X_i - X_j)$. The evaluation of these two determinants constitutes the computational bottleneck of these algorithms, which can, however, be accelerated to $[O(n^2)]$ by Sherman–Morrison–Woodbury-type updates [39,56] when only the position of a single vertex of the Feynman diagrams is altered at each Monte Carlo step (corresponding to the alteration of only one row and column in the matrices). An interesting observation with respect to this work is that one can use successive Schur complementation to compute the determinants from Eq. [20] with the same polynomial ($O(n^3)$) complexity as Gaussian elimination or similar. The advantage is that one computes one (leading) principal minor for each order $k < n$ as a side product. This allows for the simultaneous evaluation of all orders up to n and the use of the recently introduced many-configurations Markov chain Monte Carlo [53] as well as the use of conformal maps applied to coefficients at each Monte Carlo step [54]. It should be noted that a disadvantage of using Schur complementation to compute determinants is the lack of possibility to correct numerically unstable pivot values without significant additional computational cost.

Another related determinantal diagrammatic Monte Carlo algorithm (PDet) has been proposed for the Fermi polaron problem in Ref. [57]. In this case all diagrams are generated by the multiplication of a determinant in one spin with a connected set of propagators called "backbone" in the other, which ensures the overall connectedness of diagrams. For such algorithms, one might also benefit from the simultaneous evaluation of diagrams at all expansion orders up to n by means of a Schur-determinant algorithm for leading principal minors.

F. Connected diagram algorithm for bosons

It is also possible to use the recursion (17) for bosonic [58–62] Hamiltonians and Fermi-Bose mixtures [63–65]. Despite this natural generalization, no such calculations have been attempted to date. Here, we merely aim at discussing the computational complexity that is required to eliminate disconnected diagrams in a bosonic system. Notably, we can

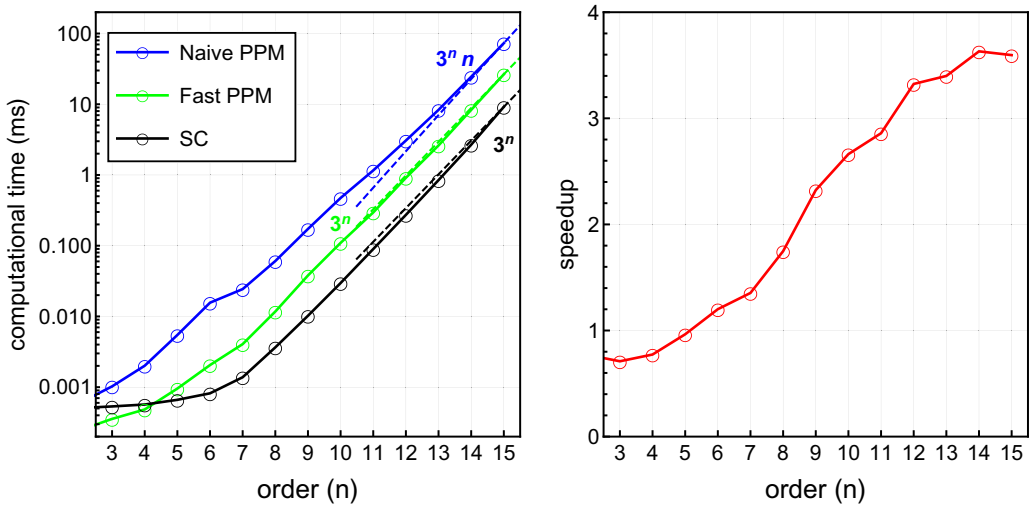

FIG. 8. Left: Computational time as a function of order (n) for the fast permanent principal minors (PPM) algorithm as compared with a naive Ryser implementation as well the subset convolution (SC) used by CDet. Right: Computational speedup of the complete fermionic CDet with respect to the complete bosonic CDet, both using the fastest available principal minor algorithms.

make use of the fast permanent principal minor algorithm introduced in the previous section. This algorithm scales as $O(3^n)$, similarly to the subset convolution in CDet, but with a slightly worse prefactor. In Fig. 8 we compare the resulting computational times for fermionic and bosonic algorithms and find that the fermionic version is about 1.5 to 3.5 faster than the bosonic one. We believe this is negligible when compared with other factors that influence the variance of both algorithms. We leave a thorough study of bosonic system with such an algorithm to future work.

VI. CONCLUSIONS

In this paper we have introduced an efficient implementation of the FPM algorithm as well as multiple generalizations with specific applications within diagrammatic Monte Carlo algorithms in mind. We have reported significant speedups for CDet in the normal as well as the superfluid phases and for diagrammatic schemes with two vertex types. The latter can potentially lead to reductions in the Monte Carlo variance of CDet computations. We have also introduced a generalization that reduces the asymptotic computational scaling of real-time diagrammatic Monte Carlo algorithms to $O(2^n)$ and leads to significant efficiency gains at reasonably attainable expansion orders. Finally, we have shown that CDet for bosonic systems has the same asymptotic scaling of $O(3^n)$ as for fermionic ones, despite the fact that exponentially scaling permanent principal minor computations are necessary. This opens the door for multiple applications of CDet to bosonic systems and Fermi-Bose mixtures. All of the presented algorithms can also be expanded to matrices with any fields as entries, notably truncated polynomials [30] and nilpotent polynomials [7,66].

ACKNOWLEDGMENTS

The authors thank C. Bertrand, R. Costa Barroso, Y. Fernàndez, A. Kim, E. Kozik, O. Parcollet, R. Rossi, G. Spada, M. Tsatsomeros, X. Waintal, and F. Werner for valuable discussions. This work was granted access to the HPC resources of TGCC and IDRIS under the allocations A0090510609 attributed by GENCI (Grand Equipement National de Calcul Intensif). This work has been supported by the Simons Foundation within the Many Electron Collaboration framework.

[1] R. Vein and P. Dale, *Determinants and Their Applications in Mathematical Physics*, Vol. 134 (Springer Science & Business Media, Berlin, Heidelberg, Germany, 2006).

[2] A. Björklund, Determinant sums for undirected hamiltonicity, SIAM J. Comput. **43**, 280 (2014).

[3] R. Blankenbecler, D. J. Scalapino, and R. L. Sugar, Monte Carlo calculations of coupled boson-fermion systems. i, Phys. Rev. D **24**, 2278 (1981).

[4] A. N. Rubtsov, V. V. Savkin, and A. I. Lichtenstein, Continuous-time quantum Monte Carlo method for fermions, Phys. Rev. B **72**, 035122 (2005).

[5] E. Burovski, N. Prokof'ev, B. Svistunov, and M. Troyer, Critical Temperature and Thermodynamics of Attractive Fermions at Unitarity, Phys. Rev. Lett. **96**, 160402 (2006).

[6] R. Rossi, Determinant Diagrammatic Monte Carlo Algorithm in the Thermodynamic Limit, Phys. Rev. Lett. **119**, 045701 (2017).

[7] R. Rossi, F. Šimkovic, and M. Ferrero, Renormalized perturbation theory at large expansion orders, Europhys. Lett. **132**, 11001 (2020).

[8] F. Šimkovic, R. Rossi, and M. Ferrero, Efficient one-loop-renormalized vertex expansions with connected determinant

diagrammatic monte carlo, Phys. Rev. B **102**, 195122 (2020).
[9] R. E. V. Profumo, C. Groth, L. Messio, O. Parcollet, and X. Waintal, Quantum monte carlo for correlated out-of-equilibrium nanoelectronic devices, Phys. Rev. B **91**, 245154 (2015).
[10] These are determinants of submatrices, obtained by removing a certain number of rows and corresponding columns.
[11] N. Prokof'ev and B. Svistunov, Fermi-polaron problem: Diagrammatic monte carlo method for divergent sign-alternating series, Phys. Rev. B **77**, 020408(R) (2008).
[12] K. Van Houcke, F. Werner, E. Kozik, N. Prokof'ev, B. Svistunov, M. Ku, A. Sommer, L. Cheuk, A. Schirotzek, and M. Zwierlein, Feynman diagrams versus fermi-gas feynman emulator, Nat. Phys. **8**, 366 (2012).
[13] R. Rossi, T. Ohgoe, K. Van Houcke, and F. Werner, Resummation of Diagrammatic Series with Zero Convergence Radius for Strongly Correlated Fermions, Phys. Rev. Lett. **121**, 130405 (2018).
[14] K. Chen and K. Haule, A combined variational and diagrammatic quantum monte carlo approach to the many-electron problem, Nat. Commun. **10**, 3725 (2019).
[15] Y. Huang, K. Chen, Y. Deng, N. Prokof'ev, and B. Svistunov, Spin-Ice State of the Quantum Heisenberg Antiferromagnet on the Pyrochlore Lattice, Phys. Rev. Lett. **116**, 177203 (2016).
[16] E. Kozik, K. Van Houcke, E. Gull, L. Pollet, N. Prokof'ev, B. Svistunov, and M. Troyer, Diagrammatic monte carlo for correlated fermions, Europhys. Lett. **90**, 10004 (2010).
[17] Y. Deng, E. Kozik, N. V. Prokof'Ev, and B. V. Svistunov, Emergent bcs regime of the two-dimensional fermionic hubbard model: Ground-state phase diagram, Europhys. Lett. **110**, 57001 (2015).
[18] A. Taheridehkordi, S. H. Curnoe, and J. P. F. LeBlanc, Algorithmic matsubara integration for hubbard-like models, Phys. Rev. B **99**, 035120 (2019).
[19] J. Vučičević and M. Ferrero, Real-frequency diagrammatic monte carlo at finite temperature, Phys. Rev. B **101**, 075113 (2020).
[20] E. Gull, A. J. Millis, A. I. Lichtenstein, A. N. Rubtsov, M. Troyer, and P. Werner, Continuous-time monte carlo methods for quantum impurity models, Rev. Mod. Phys. **83**, 349 (2011).
[21] A. Georges, G. Kotliar, W. Krauth, and M. J. Rozenberg, Dynamical mean-field theory of strongly correlated fermion systems and the limit of infinite dimensions, Rev. Mod. Phys. **68**, 13 (1996).
[22] E. Kozik, E. Burovski, V. W. Scarola, and M. Troyer, Néel temperature and thermodynamics of the half-filled three-dimensional hubbard model by diagrammatic determinant monte carlo, Phys. Rev. B **87**, 205102 (2013).
[23] K. Griffin and M. J. Tsatsomeros, Principal minors, part i: A method for computing all the principal minors of a matrix, Linear Algebra Appl. **419**, 107 (2006).
[24] F. Šimkovic, J. P. F. LeBlanc, A. J. Kim, Y. Deng, N. V. Prokof'ev, B. V. Svistunov, and E. Kozik, Extended Crossover from a Fermi Liquid to a Quasiantiferromagnet in the Half-Filled 2d Hubbard Model, Phys. Rev. Lett. **124**, 017003 (2020).
[25] A. J. Kim, F. Šimkovic, and E. Kozik, Spin and Charge Correlations Across the Metal-to-Insulator Crossover in the Half-Filled 2d Hubbard Model, Phys. Rev. Lett. **124**, 117602 (2020).
[26] T. Schäfer, N. Wentzell, F. Šimkovic, Y.-Y. He, C. Hille, M. Klett, C. J. Eckhardt, B. Arzhang, V. Harkov, F. -M. Le Régent, A. Kirsch, Y. Wang, A. J. Kim, E. Kozik, E. A. Stepanov, A. Kauch, S. Andergassen, P. Hansmann, D. Rohe, Y. M. Vilk, J. P. F. LeBlanc, S. Zhang, A.-M. S. Tremblay, M. Ferrero, O. Parcollet, and A. Georges, Tracking the Footprints of Spin Fluctuations: A Multimethod, Multimessenger Study of the Two-Dimensional Hubbard Model, Phys. Rev. X **11**, 011058 (2021).
[27] M. Klett, N. Wentzell, T. Schäfer, F. Šimkovic, O. Parcollet, S. Andergassen, and P. Hansmann, Real-space cluster dynamical mean-field theory: Center-focused extrapolation on the one- and two particle-levels, Phys. Rev. Res. **2**, 033476 (2020).
[28] F. Šimkovic and E. Kozik, Determinant monte carlo for irreducible feynman diagrams in the strongly correlated regime, Phys. Rev. B **100**, 121102(R) (2019).
[29] C. Lenihan, A. J. Kim, F. Šimkovic IV., and E. Kozik, Entropy In the Non-Fermi-Liquid Regime of the Doped 2D Hubbard Model, Phys. Rev. Lett. **126**, 105701 (2021).
[30] A. Wietek, R. Rossi, F. Šimkovic, M. Klett, P. Hansmann, M. Ferrero, E. M. Stoudenmire, T. Schäfer, and A. Georges, Mott Insulating States with Competing Orders in the Triangular Lattice Hubbard Model, Phys. Rev. X **11**, 041013 (2021).
[31] G. Spada, R. Rossi, F. Simkovic, R. Garioud, M. Ferrero, K. Van Houcke, and F. Werner, High-order expansion around bcs theory, arXiv:2103.12038.
[32] C. Bertrand, S. Florens, O. Parcollet, and X. Waintal, Reconstructing Nonequilibrium Regimes of Quantum Many-Body Systems from the Analytical Structure of Perturbative Expansions, Phys. Rev. X **9**, 041008 (2019).
[33] A. Moutenet, P. Seth, M. Ferrero, and O. Parcollet, Cancellation of vacuum diagrams and the long-time limit in out-of-equilibrium diagrammatic quantum Monte Carlo, Phys. Rev. B **100**, 085125 (2019).
[34] M. Maček, P. T. Dumitrescu, C. Bertrand, B. Triggs, O. Parcollet, and X. Waintal, Quantum Quasi-Monte Carlo Technique for Many-Body Perturbative Expansions, Phys. Rev. Lett. **125**, 047702 (2020).
[35] C. Bertrand, D. Bauernfeind, P. T. Dumitrescu, M. Maček, X. Waintal, and O. Parcollet, Quantum quasi monte carlo algorithm for out-of-equilibrium green functions at long times, Phys. Rev. B **103**, 155104 (2021).
[36] H. J. Ryser, *Combinatorial Mathematics*, Vol. 14 (American Mathematical Soc., 1963).
[37] D. G. Glynn, The permanent of a square matrix, Eur. J. Comb. **31**, 1887 (2010).
[38] L. N. Trefethen and D. Bau III, *Numerical Linear Algebra*, Vol. 50 (SIAM, University City, Philadelphia, 1997).
[39] J. Sherman and W. J. Morrison, Adjustment of an inverse matrix corresponding to a change in one element of a given matrix, Ann. Math. Statist. **21**, 124 (1950).
[40] D. E. Crabtree and E. V. Haynsworth, An identity for the schur complement of a matrix, Proc. Amer. Math. Soc. **22**, 364 (1969).
[41] R. A. Horn and C. R. Johnson, *Matrix Analysis* (Cambridge University Press, Cambridge, United Kingdom, 2012).
[42] M. J. Tsatsomeros and L. Li, A recursive test for p-matrices, BIT Numer. Math. **40**, 410 (2000).

[43] L. Li and K. Hattori, An asymptotic approach for testing p0-matrices, Electronic Notes Theor. Computer Sci. **225**, 195 (2009).

[44] L. Valiant, The complexity of computing the permanent, Theor. Comput. Sci. **8**, 189 (1979).

[45] A. Björklund, T. Husfeldt, P. Kaski, and M. Koivisto, Fourier meets möbius: Fast subset convolution, *Proceedings of the Thirty-Ninth Annual ACM Symposium on Theory of Computing STOC'07* (ACM, 2007), pp. 67–74.

[46] A. Boag, E. Gull, and G. Cohen, Inclusion-exclusion principle for many-body diagrammatics, Phys. Rev. B **98**, 115152 (2018).

[47] S. Yang, Z. Cai, and J. Lu, Inclusion-exclusion principle for open quantum systems with bosonic bath, New J. Phys. **23**, 063049 (2021).

[48] J. Bardeen, L. N. Cooper, and J. R. Schrieffer, Theory of superconductivity, Phys. Rev. **108**, 1175 (1957).

[49] J. Li, M. Wallerberger, and E. Gull, Diagrammatic Monte Carlo method for impurity models with general interactions and hybridizations, Phys. Rev. Res. **2**, 033211 (2020).

[50] J. E. Hirsch, Charge-Density-Wave to Spin-Density-Wave Transition in the Extended Hubbard Model, Phys. Rev. Lett. **53**, 2327 (1984).

[51] J. Kanamori, Electron correlation and ferromagnetism of transition metals, Prog. Theor. Phys. **30**, 275 (1963).

[52] L. V. Keldysh, Diagram technique for nonequilibrium processes, Sov. Phys. JETP **20**, 1018 (1965).

[53] F. Šimkovic and R. Rossi, Many-configuration markov-chain monte carlo, arXiv:2102.05613.

[54] A. J. Kim, N. V. Prokof'ev, B. V. Svistunov, and E. Kozik, Homotopic Action: A Pathway to Convergent Diagrammatic Theories, Phys. Rev. Lett. **126**, 257001 (2021).

[55] J. Hubbard, Electron correlations in narrow energy bands, Proc. R. Soc. London A **276**, 238 (1963).

[56] E. Gull, P. Staar, S. Fuchs, P. Nukala, M. S. Summers, T. Pruschke, T. C. Schulthess, and T. Maier, Submatrix updates for the continuous-time auxiliary-field algorithm, Phys. Rev. B **83**, 075122 (2011).

[57] K. Van Houcke, F. Werner, and R. Rossi, High-precision numerical solution of the fermi polaron problem and large-order behavior of its diagrammatic series, Phys. Rev. B **101**, 045134 (2020).

[58] N. Elstner and H. Monien, Dynamics and thermodynamics of the bose-hubbard model, Phys. Rev. B **59**, 12184 (1999).

[59] B. Capogrosso-Sansone, Ş. G. Söyler, N. Prokof'ev, and B. Svistunov, Monte carlo study of the two-dimensional bose-hubbard model, Phys. Rev. A **77**, 015602 (2008).

[60] L. Pollet, C. Kollath, K. V. Houcke, and M. Troyer, Temperature changes when adiabatically ramping up an optical lattice, New J. Phys. **10**, 065001 (2008).

[61] P. Anders, E. Gull, L. Pollet, M. Troyer, and P. Werner, Dynamical Mean Field Solution of the Bose-Hubbard Model, Phys. Rev. Lett. **105**, 096402 (2010).

[62] L. Pollet, Recent developments in quantum Monte Carlo simulations with applications for cold gases, Rep. Prog. Phys. **75**, 094501 (2012).

[63] P. Anders, P. Werner, M. Troyer, M. Sigrist, and L. Pollet, From the Cooper Problem to Canted Supersolids in Bose-Fermi Mixtures, Phys. Rev. Lett. **109**, 206401 (2012).

[64] M. Bukov and L. Pollet, Mean-field phase diagram of the Bose-Fermi hubbard model, Phys. Rev. B **89**, 094502 (2014).

[65] L. Pollet, M. Troyer, K. Van Houcke, and S. M. A. Rombouts, Phase Diagram of Bose-Fermi Mixtures in One-Dimensional Optical Lattices, Phys. Rev. Lett. **96**, 190402 (2006).

[66] R. Rossi, Direct sampling of the self-energy with connected determinant monte carlo, arXiv:1802.04743.