Wave-packet continuum discretisation for nucleon-nucleon scattering predictions

Citation for the original published paper (version of record):
Miller, S., Ekström, A., Forssén, C. (2022). Wave-packet continuum discretisation for nucleon-nucleon scattering predictions. Journal of Physics G: Nuclear and Particle Physics, 49(2).
http://dx.doi.org/10.1088/1361-6471/ac3cfd

(article starts on next page)

Wave-packet continuum discretisation for nucleon–nucleon scattering predictions

OPEN ACCESS
IOP Publishing — Journal of Physics G: Nuclear and Particle Physics
J. Phys. G: Nucl. Part. Phys. **49** (2022) 024001 (28pp) https://doi.org/10.1088/1361-6471/ac3cfd

Wave-packet continuum discretisation for nucleon–nucleon scattering predictions

Sean B S Miller[*], **Andreas Ekström** and **Christian Forssén**

Department of Physics, Chalmers University of Technology, SE-412 96 Gothenburg, Sweden

E-mail: sean.miller@chalmers.se and andreas.ekstrom@chalmers.se

Received 8 September 2021, revised 23 November 2021
Accepted for publication 24 November 2021
Published 23 December 2021

Abstract

In this paper we analyse the efficiency, precision, and accuracy of computing elastic nucleon–nucleon (NN) scattering amplitudes with the wave-packet continuum discretisation method (WPCD). This method provides approximate scattering solutions at multiple scattering energies simultaneously. We therefore utilise a graphics processing unit to explore the benefits of this inherent parallelism. From a theoretical perspective, the WPCD method promises a speedup compared to a standard matrix-inversion method. We use the chiral NNLO$_{\mathrm{opt}}$ interaction to demonstrate that WPCD enables efficient computation of NN scattering amplitudes provided one can tolerate an averaged method error of 1–5 mb in the total cross section at scattering energies 0–350 MeV in the laboratory frame of reference. Considering only scattering energies $\sim$40–350 MeV, we find a smaller method error of $\lesssim$ 1–2 mb. By increasing the number of wave-packets we can further reduce the overall method error. However, the parallel leverage of the WPCD method will be offset by the increased size of the resulting discretisation mesh. In practice, a GPU-implementation is mainly advantageous for matrices that fit in the fast on-chip shared memory. We find that WPCD is a promising method for computationally efficient, statistical analyses of nuclear interactions from effective field theory, where we can utilise Bayesian inference methods to incorporate relevant uncertainties.

[*]Author to whom any correspondence should be addressed.

Keywords: nucleon–nucleon scattering, wave-packet continuum discretization, chiral effective field theory

(Some figures may appear in colour only in the online journal)

1. Introduction

A large portion of interaction-potential models currently applied in *ab initio* many-nucleon calculations are constructed using ideas from chiral effective field theory (χEFT) [1–4]. Such potentials typically contain $\gtrsim$10–30 low-energy constants (LECs), acting as physical calibration parameters, that must be inferred from data. Bayesian methods for parameter estimation offer several advantages in this regard, in particular in conjunction with EFTs, see e.g. reference [5]. However, making reliable inferences typically incur a high total computational cost due to the large amount of posterior samples. It is therefore important to establish an efficient computational framework for generating model predictions of physical observables.

In this paper we look to low-energy nucleon–nucleon (NN) scattering cross sections as it constitutes the bulk part of the standard dataset for inferring the most probable values of the LECs, see e.g. references [6–9]. The most recent, and statistically consistent, database [10] of NN scattering cross sections contains data for thousands of measured proton–proton (pp) and neutron–proton (np) cross sections at hundreds of different laboratory scattering energies, mostly below the pion-production threshold at $T_{\text{lab}} \approx 290$ MeV. In most cases, the computational bottleneck when predicting NN scattering amplitudes for a given interaction potential comes from obtaining numerical solutions to the Lippmann–Schwinger (LS) equation.

There are essentially three approaches for improving computational efficiency and speed of a computational procedure: (i) develop improved numerical methods and algorithms tailored to the physical model at hand and its application, (ii) use specific hardware, e.g. a faster CPU, increased memory bandwidth, or parallel architectures such as in a graphics processing unit (GPU) to better handle some of the most dominant computational procedures of the model, or (iii) replace any computationally expensive model evaluations with a fast, as well as sufficiently accurate and precise, surrogate model, i.e. an *emulator*, which mimics the original model output. This latter approach is very interesting and in particular eigenvector continuation (EC) [11, 12] applied to emulate NN-scattering amplitudes [13, 14] shows great promise, although uncertainty quantification is yet to be explored. Note, however, that EC emulation attains most of its speedup when applied to potential models that exhibit linear parameter dependencies. In cases where such dependencies are not present one might resort to other methods to handle the non-linear response, such as EC combined with Gaussian process emulation [15]. However, one should note that Gaussian process emulation [16] and other standard machine learning methods exhibit poor scaling with increasing dimensionality of the input parameter domain.

Here, we will focus on approaches (i) and (ii) by exploring both the standard matrix-inversion (MI) method, see e.g. reference [17], and the wave-packet continuum discretisation (WPCD) method [18] for solving the LS equation. The WPCD method basically corresponds to a bound-state approach that uses eigenfunctions of the full NN Hamiltonian to approximate scattering solutions at any on-shell energy. This method is particularly interesting since it provides approximate scattering amplitudes at multiple scattering energies simultaneously. We have therefore implemented this inherently parallel method on a GPU. We also note that WPCD places no constraint on the analytical form of the potential or its parametric dependence. As such, WPCD acceleration for NN scattering complements the EC approaches for emulation [13–15] mentioned above.

Speeding up computations very often comes at the expense of accuracy and/or precision, and the WPCD method is no exception to this principle. The magnitude of experimental errors in the calibration data and the estimated theoretical model-discrepancies [19] provide natural tolerances for the level of method error that is acceptable. Indeed it is undesirable to have a method error that dominates the error budget such that it obscures or even hampers the inference of useful information. We therefore analyse the WPCD method in detail and quantify realistic computational speedups and compare the method errors to recent estimates of the model discrepancy in χEFT [20]. We also analyse and compare the numerical complexities of the MI and WPCD methods.

2. NN scattering

The LS equation, in operator form, for the transition-matrix operator $\hat{T}$ at some scattering energy E is given by

$$\hat{T}(E) = \hat{v} + \hat{v}\hat{g}_0(E)\hat{T}. \tag{1}$$

This is an inhomogeneous Fredholm integral equation where $\hat{v}$ is some NN-potential operator, $\hat{g}_0(E) = (E - \hat{h}_0 + i\epsilon)^{-1}$ is the free Green's function, and $\hat{h}_0$ is the free Hamiltonian, i.e. the kinetic energy operator, and $i\epsilon \to 0$ is the positive imaginary part of the complex energy. Most realistic NN potentials in nuclear *ab initio* calculations do not furnish analytical solutions for the T-matrix. It is however straightforward to numerically solve for the T-matrix for an on-shell energy E. There exists e.g. variational methods [21–24], as well as Neumann or Born series expansions for sufficiently weak potentials. One can also try Padé extrapolants [25] in cases where the integral kernel is not sufficiently perturbative to converge the resulting Neumann series. In this work we employ the standard MI method [17] which amounts to inverting a relatively small complex-valued matrix at each scattering energy. This is a trivial operation that can be straightforwardly carried out within milliseconds on a modern CPU. However, solving at multiple scattering energies to obtain all cross-sections present in the NN database amounts to at least a few seconds of computation. In a Bayesian analysis, where one repeatedly evaluates a likelihood function across a multi-dimensional parameter domain, any speedup in the solution of the LS equation will directly impact the total computation time.

The momentum-space partial-wave representation of the LS equation for the NN T-matrix, equation (1), is given by

$$T^{sJ}_{l'l}(q', q; E) = v^{sJ}_{l'l}(q', q) + \frac{2}{\pi}\sum_{l''}\int_0^\infty \mathrm{d}k\, k^2 v^{sJ}_{l'l''}(q', k) g_0(k; E) T^{sJ}_{l''l}(k, q; E), \tag{2}$$

where q, q', and k are relative momenta, $E = p^2/m_\mathrm{N}$ is the centre-of-mass (c.m.) energy with momentum p, m_N is the nucleon mass, and $g_0(k; E) \equiv (E - k^2/m_N + i\epsilon)^{-1}$. In this work we only consider the canonical NN interaction potential, and therefore use the shorthand notation $T^{sJ}_{l'l}(q', q) \equiv \langle q', l', s, J|\hat{T}|q, l, s, J\rangle$ for partial-wave amplitudes. Furthermore we use the following normalisation of momentum states, $\langle k'|k\rangle = \frac{\delta(k'-k)}{k'k}$. We suppress isospin notation since the total isospin T is defined uniquely by the Pauli principle given the spin and angular momentum quantum numbers s and l, respectively, while the projected isospin T_z is defined at the outset of, and conserved throughout, a scattering observable calculation. Methods for obtaining the partial-wave-projected potential $v^{sJ}_{l'l}(q', q)$ can be found in e.g. reference [26]. Integrated and differential scattering cross sections at a specific scattering energy E can be straightforwardly evaluated given the partial-wave T-matrix using the expressions presented in appendix A.

Throughout this work we will compare the efficiency and accuracy of the WPCD method, presented in section 3.2, to a set of numerically exact results obtained using the MI method presented below.

2.1. MI method for solving the LS equation

For the resolvent $\hat{g}_0(E)$, the kernel in the LS equation (2) has a pole singularity at $k = p$. This can be handled via a principal-value decomposition[1], i.e.

$$\lim_{\epsilon\to 0} \frac{1}{x \pm i\epsilon} = \mathcal{P}\left(\frac{1}{x}\right) \mp i\pi\delta(x), \quad x \in \mathbb{R}, \tag{3}$$

such that the remaining integral can be evaluated using e.g. Gauss–Legendre quadrature on some grid $\{k_i\}_{i=1}^{N_Q}$ of momenta $k_i \neq p$ with corresponding weights w_i. Following [17], the complex T-matrix in equation (2) can be solved via the inversion of a finite-dimensional matrix equation for the on-shell momentum $q = p = \sqrt{m_\mathrm{N} E}$,

$$T_{l'l}^{sJ}(q,q;E) = v_{l'l}^{sJ}(q,q) + \frac{2}{\pi}\sum_{l''}\sum_{i=1}^{N_Q} w_i v_{l'l''}^{sJ}(q,k_i) g_0(k_i;E) T_{l''l}^{sJ}(k_i,q). \tag{4}$$

We can introduce a basis $q_i \in \{k_1, k_2, \ldots, k_{N_Q}, p\}$ such that the operators can be written in matrix form with row i and column j corresponding to q_i and q_j respectively, i.e. $(V_{l'l}^{sJ})_{ij} \equiv \langle q_i | v_{l'l}^{sJ} | q_j \rangle$. The on-shell T-matrix element is then given by $(T_{l'l}^{sJ})_{N_Q+1,N_Q+1}$. Introducing a vector D with elements defined as

$$D_i \equiv \begin{cases} \dfrac{2w_i k_i^2 m_N}{\pi(k_i^2 - p^2)} & \text{if } i \leqslant N_Q\,, \\ -\displaystyle\sum_{i=1}^{N_Q} \frac{2w_i p^2 m_N}{\pi(k_i^2 - p^2)} + \frac{i\pi p m_N}{2} & \text{if } i = N_Q + 1. \end{cases} \tag{5}$$

Equation (4) can be rewritten as a linear system of equations,

$$\sum_{l''} F_{l'l''}^{sJ} T_{l''l}^{sJ} = V_{l'l}^{sJ}, \tag{6}$$

where we have introduced the wave matrix $F_{l'l}^{sJ}$,

$$(F_{l'l}^{sJ})_{ij} \equiv \delta_{ij}\delta_{l'l} - (V_{l'l}^{sJ})_{ij} D_j. \tag{7}$$

Direct inversion of $F_{l'l}^{sJ}$ in equation (6) is usually discouraged in scientific computing due to the instability of MI algorithms [27]. A more advisable practice is to use LU-decomposition. Additionally, it is numerically more stable to first introduce the K-matrix[2] as the principal value part of the LS equation. Since the potential v is real, the K-matrix is also real. This leads to a set of purely real matrix-equations [17] based on a nonsingular integral.

The on-shell momentum dependence in $F_{l'l}^{sJ}$ demands the solution of an entire linear system of equations for every energy of interest. Formally, the T-matrix is defined by the potential

[1] Kramers–Konig relation, dispersion relation, or the Sokhotski–Plemelj identity.

[2] Also referred to as the reactance R-matrix.

operator via [28]

$$T^{sJ}_{l'l}(q',q) \equiv \langle q'|\hat{v}^{sJ}_{l'l}|\psi^{+}_{q}\rangle, \tag{8}$$

where $|\psi^{+}_{q}\rangle$ are eigenstates (outbound scattering states) of the full Hamiltonian $\hat{h}$ with momentum q. With this, we can instead express the LS equation using the full resolvent $\hat{g}(E) = (E - \hat{h} + i\epsilon)^{-1}$,

$$T^{sJ}_{l'l}(q',q;E) = v^{sJ}_{l'l}(q',q) + \frac{2}{\pi}\sum_{l''}\int_0^\infty \mathrm{d}k\, k^2 v^{sJ}_{l'l''}(q',\psi^{+}_{k})g(k;E)v^{sJ}_{l''l}(\psi^{+}_{k},q), \tag{9}$$

where $v^{sJ}_{l'l}(q',\psi^{+}_{k}) = \langle q'|\hat{v}^{sJ}_{l'l}|\psi^{+}_{k}\rangle$ and $g(k;E) = (E - k^2/m_\mathrm{N} + i\epsilon)^{-1}$. This is significantly easier to evaluate numerically using quadrature since it only amounts to matrix multiplications. However, the scattering states $|\psi^{+}_{q}\rangle$ are not available at the outset. This is where the WPCD method enters to effectively approximate the scattering states using square-integrable eigenstates of the NN Hamiltonian.

3. Wave-packet continuum discretisation

The WPCD method [18] effectively eliminates the requirement to explicitly solve the LS equation at several scattering energies E using MI. Instead, one diagonalises the full Hamiltonian in a finite basis, and uses the resulting discrete set of eigenstates to approximate all scattering states of interest. Equipped with these states, this approach enables straightforward evaluation of the *full* resolvent at any value of the on-shell scattering energy E, which makes the WPCD method intrinsically parallel with respect to obtaining scattering solutions at different energies. This is one of several known bound-state techniques to solve the multi-particle scattering problem [29]. To provide a self-contained presentation, we devote this section to introduce the WPCD method for describing elastic NN scattering, starting with a definition of a finite wave-packet basis.

3.1. Scattering observables in a finite basis

Generally we can project some Hamiltonian state $|\Psi(E)\rangle$ with positive energy E onto a complete basis (including both bound and free basis states). In this case, the expectation value of an operator $\hat{O}(\hat{h})$ depending purely on the full Hamiltonian $\hat{h}$ can be represented in the following form,

$$\langle\Psi(E)|\hat{O}|\Psi(E)\rangle = \sum_{i=1}^{n_b} u(\epsilon^b_i)|\langle\Psi|\psi^b_i\rangle|^2 + \int_0^\infty \mathrm{d}E' u(E')|\langle\Psi|\psi(E')\rangle|^2, \tag{10}$$

where $\{|\psi^b_i\rangle\}_{i=1}^{n_b}$ are bound states with energies ϵ^b_i and $|\psi(E')\rangle$ are free states with energy E', both of which are eigenstates of $\hat{h}$, and we have defined $u(\epsilon^b_i) = \langle\psi^b_i|\hat{O}|\psi^b_i\rangle$ and $u(E') = \langle\psi(E')|\hat{O}|\psi(E')\rangle$. Naturally, we have $|\Psi(E)\rangle = |\psi(E)\rangle$ (for $E > 0$) as $\hat{h}$ is the Hamiltonian and Ψ the eigenstate, collapsing the expansion above. However, this identity is not useful in numerical approaches since we do not know $\psi(E)$, leaving us to solve the integral in a finite, approximative basis.

A computational routine for evaluating integrals, such as quadrature, uses a finite mesh of points where the integrand is evaluated. In scattering, this approach requires a finite basis of states $|\widetilde{\psi}_i\rangle$ with corresponding positive energies E_i. These states do not span the whole continuous momentum-space and are thus referred to as pseudostates. Below, we will demonstrate how to construct the pseudostates in a wave-packet basis. The pseudostates form a basis for a finite quadrature-prescription to express an operator

$$\langle\Psi|\hat{O}(\hat{h})|\Psi\rangle \approx \sum_{i=1}^{n_b} u(\epsilon_i^b)|\langle\Psi|\psi_i^b\rangle|^2 + \sum_{i=1}^{n} u(E_i)|\langle\Psi|\widetilde{\psi}_i\rangle|^2, \tag{11}$$

where we introduce quadrature weights w_i such that

$$|\langle\Psi|\widetilde{\psi}_i\rangle|^2 = w_i|\langle\Psi|\psi(E_i)\rangle|^2. \tag{12}$$

To determine the weights w_i we can use an equivalent quadrature (EQ) technique [30–34] in which it can be shown that the weights do not depend on the state $|\Psi\rangle$. The weights represent a type of transformation coefficient between pseudostates $|\widetilde{\psi}_i\rangle$ and the states $|\psi(E')\rangle$. An approximate relation based on equation (12) can then be introduced,

$$\langle\psi(E_i)|\hat{O}|\psi(E_i)\rangle \approx \frac{\langle\widetilde{\psi}_i|\hat{O}|\widetilde{\psi}_i\rangle}{\sqrt{w_i}}. \tag{13}$$

In the WPCD method we discretise the continuum of free states. A wave packet is defined as the energy integral over some energy 'bin' of width ΔE with Hamiltonian eigenstates $|\psi(E)\rangle$ with positive energy E as the integrand,

$$|\psi(E,\Delta E)\rangle \equiv \int_E^{E+\Delta E} \mathrm{d}E' \; |\psi(E')\rangle. \tag{14}$$

It follows that $\lim_{\Delta E\to 0}|\psi(E,\Delta E)\rangle = |\psi(E)\rangle$. We can define an orthogonal wave-packet basis by letting the bin boundaries E and widths ΔE lie on a mesh such that the bins do not overlap: $\{\mathcal{D}_i \mid \mathcal{D}_i \cap \mathcal{D}_j = \emptyset \,\forall\, i \neq j\}_{i=1}^{N_{\mathrm{WP}}}$, where $\mathcal{D}_i \equiv [E_i, E_i + \Delta E_i]$ defines the integral boundaries. Note that $E_{i+1} = E_i + \Delta E_i$. It is straightforward to normalise this basis and show that the wave packets have eigenenergies $e_i = E_i + \frac{1}{2}\Delta E_i$.

The expectation value of an operator $\hat{O}$ in some energy bin can now be approximately represented in a wave-packet basis

$$\langle\Psi(E)|\hat{O}|\psi(E')\rangle \approx \frac{\langle\Psi(E)|\hat{O}|\psi(E_i,\Delta E_i)\rangle}{\sqrt{\Delta E_i}}, \tag{15}$$

where $E' \in \mathcal{D}_i$. As expected, the quality of this approximation is subject to the bin widths ΔE_i. It can be reasoned [35–38], on behalf of equations (13) and (15), that the EQ weights are approximately given by

$$w_i \approx \Delta E_i. \tag{16}$$

In short, we have a method for approximating the EQ weights in a wave-packet basis, with which we can express the spectrum of a scattering operator and thereby effectively solve scattering problems.

We now proceed to approximately represent the pseudostates as Hamiltonian eigenstates in a wave-packet representation. To this end, we setup a wave-packet equivalent of a partial wave by generalising equation (14) and thus define a normalised free wave-packet (FWP) as

$$|x_i\rangle \equiv \frac{1}{\sqrt{N_i}}\int_{\mathcal{D}_i} k\mathrm{d}k\, f(k)|k\rangle, \tag{17}$$

where $f(k)$ is a weighting function and N_i is a normalisation constant. We obtain energy or momentum wave-packets using weighting functions $f(k) = 1$ or $f(k) = \sqrt{\frac{k}{\mu}}$, respectively, where $\mu = \frac{m_\mathrm{N}}{2}$ is the reduced mass. As above, these two types of wave packets have eigenvalues

$$\hat{h}_0|x_i\rangle = \left(E_i + \frac{1}{2}\Delta E_i\right)|x_i\rangle, \tag{18}$$

$$\hat{p}|x_i\rangle = \left(k_i + \frac{1}{2}\Delta k_i\right)|x_i\rangle, \tag{19}$$

where $\hat{p}$ is the momentum operator. The normalisation of the momentum and energy wave-packets is given by the bin widths, i.e. $N_i = \Delta k_i$ or $N_i = \Delta E_i$, respectively. Operators represented in a basis of partial-wave plane-wave states are related to the FWP representation via

$$\langle q'|\hat{O}|q\rangle \approx \frac{f(q)f(q')}{\sqrt{N_i N_j}}\frac{1}{q'q}\langle x_i|\hat{O}|x_j\rangle, \tag{20}$$

where $q' \in \mathcal{D}_i,\ q \in \mathcal{D}_j$. In this work we have implemented energy as well as momentum wave-packets, and have found no significant difference or advantage of either choice.

The eigenstate wave-packets $|z_i\rangle$ of the full Hamiltonian with positive energy, or scattering wave packets (SWPs), are given by the eigenvalue equation $\hat{h}|z_i\rangle = \epsilon_i|z_i\rangle$ for energies $\epsilon_i > 0$. Similarly, the bound-state[3] wave-packets $|z_i^b\rangle$ have energies $\epsilon_i^b < 0$. We obtain them straightforwardly via diagonalisation in a finite basis $\{|x_i\rangle\}_{i=1}^{N_\mathrm{WP}}$ of FWPs, where $N_\mathrm{WP} = n + n_b$. This operation is the most time-consuming part in the WPCD method to solve the LS equation. It provides us with a matrix of transformation coefficients $C_{ij} \equiv \langle x_i|z_j\rangle$ such that

$$H = CDC^\mathrm{T}, \tag{21}$$

where D is a diagonal matrix of energies ϵ_i^b and ϵ_i, and $H_{ij} = \langle x_i|\hat{h}|x_j\rangle$. In order to solve the LS equation on the form of equation (9), we must define a wave-packet representation of the outbound scattering states. To define SWPs, we must construct the bin boundaries $[\mathcal{E}_i, \mathcal{E}_i + \Delta\mathcal{E}_i] = [\mathcal{E}_i, \mathcal{E}_{i+1}]$ of $|z_i\rangle$ such that the positive eigenenergies are given by

$$\epsilon_i = \mathcal{E}_i + \frac{1}{2}\Delta\mathcal{E}_i. \tag{22}$$

This is no different than the wave-packet eigenvalues in equations (18) and (19). This shift of the FWP energies are indicated in the left panel in figure 1. However, it is not possible to construct the bin-boundaries $\mathcal{E}_i$ exactly since equation (22) only provides n equations while there are $n+1$ boundaries. Therefore, the following scheme [39] can be used to approximate the bin boundaries of $|z_i\rangle$,

$$\begin{aligned}
\mathcal{E}_1 &\equiv 0,\\
\mathcal{E}_i &\equiv \frac{1}{2}(\epsilon_{i-1} + \epsilon_i),\\
\mathcal{E}_{n+1} &\equiv \epsilon_n + \frac{1}{2}(\mathcal{E}_n - \mathcal{E}_{n-1}),
\end{aligned} \tag{23}$$

[3] Here we change bound-state notation from equation (11) such that $|\psi_k^b\rangle \to |z_k^b\rangle$. This seems natural since $|z_k^b\rangle$ is expressed in terms of FWPs from the diagonalisation.

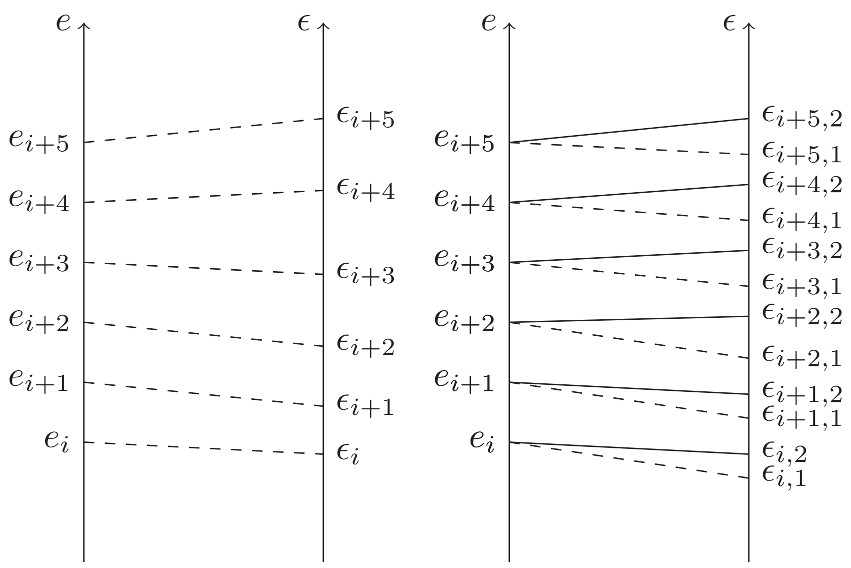

Figure 1. Left: shift in FWP energies $e_i \equiv E_i + \frac{1}{2}\Delta E_i$ to SWP energies ϵ_i. Right: shift and splitting of degenerate FWP energies e_i into energies $\epsilon_{i,1}$ and $\epsilon_{i,2}$ shown by solid and dashed lines, respectively. Note that a solid and a dashed line from two different energies e_i and e_j will not cross [18, 40].

such that they yield approximate eigenvalues $\bar{\epsilon}_i$,

$$\bar{\epsilon}_i \equiv \mathcal{E}_i + \frac{1}{2}\Delta\mathcal{E}_i \approx \epsilon_i. \tag{24}$$

In the case of K coupled channels, the FWP energies are degenerate and will be split[4] in the full Hamiltonian eigendecomposition. A FWP with energy $e_i = E_i + \frac{1}{2}\Delta E_i$ will give rise to K SWPs $|z_{i,\varkappa}\rangle$ with energies $\epsilon_{i,\varkappa}$, corresponding to each coupled state $\varkappa = 1, 2, \ldots, K$. The energies are typically ordered such that $\epsilon_{i,\varkappa} < \epsilon_{i,\varkappa+1}\ \forall i, \varkappa$ [41]. Furthermore, this prevents mixing between levels with different energies [18], i.e. $\epsilon_{i,\varkappa} < \epsilon_{j,\varkappa+1}\ \forall i < j$. Therefore, we use the boundary construction scheme in equation (23) such that $|z_{i,\varkappa}\rangle$ have boundaries given by the energies $\epsilon_{i,\varkappa}$. Note it is very important to construct the Hamiltonian matrix with degenerate FWP bases representing each coupled state, i.e. $|x_{i,1}\rangle = |x_{i,2}\rangle$, see the right panel of figure 1.

The approach presented so far works fine for short-range NN potentials. Likewise, the eigenspectrum of the long-range Coulomb Hamiltonian can be straightforwardly described using wave packets, but these must then be constructed from Coulomb wave functions instead of free states $|q\rangle$ [40] as in equation (17). In WPCD, we automatically 'smooth' out the typical low-momentum singularities presented by the Coulomb Hamiltonian. This means that the formalism of WPCD works well for both the short- and long-range parts of the interaction, but it is necessary to treat them separately. In this work we have not studied Coulomb wave packets as we only consider neutron–proton scattering.

[4] Naturally, $K \leqslant 2$ for NN scattering.

3.2. WPCD-method for solving the LS equation

Following equation (20), we can relate elements of the T-matrix in a continuous partial-wave basis and a FWP basis via

$$T^{sJ}_{l'l}(q', q; E) = \langle q'|\hat{T}^{sJ}_{l'l}(E)|q\rangle \approx \frac{f(q)f(q')\langle x_i|\hat{T}^{sJ}_{l'l}(E)|x_j\rangle}{\sqrt{N_i N_j}qq'}. \tag{25}$$

If we use the full resolvent $\hat{g}(E)$, defined in connection with equation (9), we can write

$$\hat{T}(E) = \hat{v} + \hat{v}\hat{g}(E)\hat{v}, \tag{26}$$

such that in a partial-wave-projected wave-packet basis we obtain

$$\begin{aligned}\langle x_i|\hat{T}^{sJ}_{l'l}(E)|x_j\rangle =& \langle x_i|\hat{v}^{sJ}_{l'l}|x_j\rangle + \sum_{l''}\sum_{k}^{n_b} \frac{\langle x_i|\hat{v}^{sJ}_{l'l''}|z^b_k\rangle\langle z^b_k|\hat{v}^{sJ}_{l''l}|x_j\rangle}{E - \epsilon^b_k} \\ &+ \sum_{l''}\sum_{k=1}^{n} \langle x_i|\hat{v}^{sJ}_{l'l''}|z_k\rangle\langle z_k|\hat{g}(E)|z_k\rangle\langle z_k|\hat{v}^{sJ}_{l''l}|x_j\rangle.\end{aligned} \tag{27}$$

Note that for a realistic potential we should only have $n_b = 1$ (the deuteron). The full resolvent is given by the full Hamiltonian, of which $|z_k\rangle$ are eigenstates. In such a basis we can derive a closed form expression for $\hat{g}(E)$, see appendix B. In the WPCD representation of equation (27) all energy-dependence is straightforwardly evaluated via the resolvent. We can therefore find an on-shell T-matrix element via simple summation of the scattering wave-packets. This is an important advantage of using the WPCD method for simulating scattering processes.

Note, however, that the resolvent has a logarithmic singularity for $E = \mathcal{E}_k$ or $E = \mathcal{E}_{k+1}$. We handle this by averaging with respect to the on-shell energy E,

$$g^k_i \equiv \frac{1}{\Delta\mathcal{E}_k}\int_{\mathcal{D}_k} \langle z_i|g(E)|z_i\rangle \mathrm{d}E. \tag{28}$$

The nuclear potential, represented in a FWP basis as

$$\langle x_i|\hat{v}^{sJ}_{l'l}|x_j\rangle = \frac{1}{\sqrt{N_i N_j}}\int_{\mathcal{D}_i}\int_{\mathcal{D}_j} k'k\mathrm{d}k'\mathrm{d}k f(k')f(k)\langle k'|\hat{v}^{sJ}_{l'l}|k\rangle, \tag{29}$$

usually vary mildly across a typical momentum-bin $\mathcal{D}_i$. It is therefore often sufficient to use a midpoint approximation to evaluate the integral. This offers a significant reduction in computational cost. In a momentum wave-packet basis, the midpoint approximation is simply given by

$$\langle x_i|\hat{v}^{sJ}_{l'l}|x_j\rangle \approx \bar{k}_i\bar{k}_j\sqrt{\Delta k_i \Delta k_j}\langle \bar{k}_i|\hat{v}^{sJ}_{l'l}|\bar{k}_j\rangle, \tag{30}$$

where $\bar{k}_i = \frac{k_i + k_{i+1}}{2}$ are the bin midpoints.

Here we summarise the necessary steps to implement the WPCD method for NN-scattering calculations.

(a) Distribute the bin boundaries for the free wave-packets and choose a weighting function $f(k)$, see equation (17). Recommended options are e.g. uniform (equidistant), Gauss–Legendre, or Chebyshev distributions. The results in this work are based on a

Chebyshev distribution. The Chebyshev distribution for N_{WP} points, $\{y_j\}_{j=1}^{N_{\mathrm{WP}}}$, is defined by [18]

$$y_j = \alpha \tan^t\left(\frac{2j-1}{4N_{\mathrm{WP}}}\pi\right), \quad j = 1, \ldots, N_{\mathrm{WP}}, \tag{31}$$

where t is a 'sparseness degree' and α is a scaling parameter. Let y_k be either the momentum or energy bin boundaries, and we then set the initial boundary $y_0 = 0$. For our simulations we have used momentum wave packets ($f(k) = 1$), $t = 2$, and $\alpha = 100$ MeV.

(b) Diagonalise the Hamiltonian in a FWP basis. This yields a set of energy eigenvalues ϵ_i and accompanying matrix of eigenvectors C as according to equation (21).

(c) Construct bin-boundaries $\mathcal{E}_i$ for the pseudostate wave-packets using the set of energy eigenvalues and equation (23).

(d) Express the full resolvent $g(E)$ in the scattering wave-packet basis, see appendix B. In this work we use the energy-averaged resolvent from equation (28) to avoid singularities.

(e) Obtain approximate T-matrix elements in the FWP basis by evaluating the LS-equation

$$T_{ij}(E) = V_{ij} + (VC)_{ik} g_{kk}(E) (VC)_{jk}, \tag{32}$$

where $(VC)_{ik} \equiv V_{ij} C_{jk}$, $V_{ij} \equiv \langle x_i|\hat{v}|x_j\rangle$, $g_{kk} \equiv \langle z_k|\hat{g}(E)|z_k\rangle$, and C is given by equation (21).

The T-matrix can be transformed to a continuous plane-wave basis using equation (20). However, it is important to note that with energy averaging there is ambiguity in which value $q', q \in \mathcal{D}_i$ to choose, as the wave-packet representation gives the same value for all choices. While we discuss this further in section 5.1, we find it more beneficial to choose q', q such that they match the wave-packet eigenvalues, given in equations (18) and (19).

4. Numerical complexities of the MI and WPCD methods

Here we present the minimum number of floating-point operations (FLOP) required for computing on-shell $\langle q|T_{l'l}^{sJ}|q\rangle$ matrix elements at n_E different values of the scattering energy. We focus on the MI and WPCD methods, presented in sections 2.1 and 3.2, respectively. We also assume that the potential matrix $\langle p'|V_{l'l}^{sJ}|p\rangle$ is pre-computed and available in memory at the outset. Typically, in both methods we use matrices of sizes $n \times n$ for $n < 100$. To avoid confusion, we let N_Q denote the number of quadrature points in the case of the MI method, while N_{WP} denotes the number of wave packets in the case of the WPCD method. We retain n to symbolise a basis size in general, regardless of method.

4.1. MI complexity

Given a quadrature grid with N_{Q} points, the MI method first requires setting up the wave matrix (7) at each scattering energy. Naively, the complexity of the matrix construction is dominated by the matrix–matrix product of the potential V with the resolvent g_0. Since the resolvent is diagonal, it is more efficient to multiply each row of V with the diagonal element of g_0, yielding $n + 1$ scalar–vector multiplications in a single F-matrix construction with numerical complexity according to

$$O(F) = 2(N_{\mathrm{Q}} + 1)^2 + 2(N_{\mathrm{Q}} + 1), \tag{33}$$

where the factor of 2 is due to g_0 being complex.

The on-shell T-matrix element is obtained from the last element of the T-matrix, i.e.

$$T(q,q;E) = T_{N_Q+1,N_Q+1} = \sum_{i=0}^{N_Q+1} F^{-1}_{N_Q+1,i} V_{i,N_Q+1}, \tag{34}$$

where we have expressed it using MI. MI typically requires $O(n^3)$ operations for an $n \times n$ matrix. However, solving the LS linear system,

$$FT = V, \tag{35}$$

is more advisable, and can be done very efficiently in two steps: firstly, we perform a lower-upper (LU) decomposition, where a square matrix A is expressed as the product of a lower-triangular matrix L with an upper-triangular matrix U,

$$A = LU, \tag{36}$$

which requires $O\left(\frac{2}{3}n^3\right)$ operations. The decomposition allows for $AX = B$ to be solved for each column of X using $O(2n^2)$ operations. For complex matrices these numbers increase by a factor 4 for both routines.

In summary, the MI-method has a total computational complexity of (note the linear scaling with the number of on-shell energies n_E)

$$O_{\mathrm{MI}}(T) = 2n_E\left(\frac{4}{3}(N_Q+1)^3 + 5(N_Q+1)^2 + N_Q + 1\right), \tag{37}$$

$$O_{\mathrm{MI}}(K) = n_E\left(\frac{2}{3}(N_Q+1)^3 + 5(N_Q+1)^2 + N_Q + 1\right), \tag{38}$$

where $O_{\mathrm{MI}}(K)$ shows the cost of using a K-matrix approach instead. The difference is simply replacing the cost of doing complex calculations with real calculations. We see there is roughly a factor 4 speedup in using a K-matrix representation instead of a T-matrix representation.

4.2. WPCD complexity

The complexity of the WPCD-method is dominated by three kinds of linear algebra tasks (i) the Hamiltonian matrix diagonalisation, (ii) two matrix-matrix products, and (iii) one matrix addition (see steps (a)–(e) in section 3.2). We let N_{WP} be the number of wave packets and n_E the number of scattering energies as before. By 'parallelism' we refer to the fact that WPCD can solve the scattering problem for several energies at once, following a single Hamiltonian diagonalisation.

For this reason we have fully implemented the WPCD method on a GPU utilising the CUDA interface with the cuBLAS [42] and cuSOLVER [43] libraries for linear operations. A more detailed outline of the GPU code is provided in appendix C. Sequential algorithms are usually easier to handle, and not all hardware allow for optimal parallelisation due to, for example, slow computer memory transfer. Thus, we present complexity models for both the sequential and parallel energy-evaluations of the WPCD method. Note, however, that this is simply a comparison of FLOP models that do not account for memory transfer times or detailed processor architecture. We emphasise that all WPCD results presented in this paper were calculated using our parallel GPU-code, and that the sequential complexity is presented purely to provide further insight.

The MI method can also be implemented efficiently on a GPU. This approach will not exhibit the same inherent parallelism with respect to multiple on-shell calculations. However, there does exist efficient parallel routines for solving linear systems on the form of equation (6).

4.2.1. Sequential complexity. The sequential mode for the WPCD method means solving the LS equation for each on-shell energy in sequence rather than simultaneously. This suggests using a CPU, rather than a GPU, since CPUs typically have a faster clock frequency and can thus iterate through the on-shell energies faster than a GPU. Of course, CPUs today are multicore and with several processing units, but note that they might not have a number of cores equal to or greater than n_E. The analysis below assumes a scenario where a processing unit is working with a single computational core.

(a) A real-valued NN Hamiltonian is efficiently diagonalised using QR factorisation or a divide-and-conquer algorithm, both of which are usually used together with Householder transformations. The divide-and-conquer algorithm has a complexity of $O\left(\frac{8}{3}n^3\right)$ for getting both eigenvectors and eigenvalues of an $n \times n$ matrix.

(b) Next, we evaluate the matrix-matrix product of the potential matrix V and the coefficient matrix C in the rightmost term in equation (32). Square matrix-matrix multiplication requires $2n^3 - n^2$ FLOP in a straightforward, sequential approach.

(c) The LS equation for $T_{ii}(E \in \mathcal{D}_i)$ is evaluated via simple summation and multiplication. The sum,

$$\sum_{j=1}^{N_{\mathrm{WP}}} (VC)_{ij} g_{jj}(E \in \mathcal{D}_i)(VC)_{ji}, \tag{39}$$

involves two multiplications per term, and the sum will require $N_{\mathrm{WP}} - 1$ additions. There is also the addition of the first term in equation (32). These operations must be done for every on-shell energy, resulting in a complexity of $6N_{\mathrm{WP}} \times n_E$ FLOP. Note that if we use energy averaging we only require a single evaluation for each on-shell wave packet, giving an upper limit $n_E \leqslant N_{\mathrm{WP}}$.

The WPCD sequential complexity is then given by

$$O_{\mathrm{WPCD,seq.}}(T) = 6N_{\mathrm{WP}}^3 - N_{\mathrm{WP}}^2 + 6N_{\mathrm{WP}} \times n_E. \tag{40}$$

To summarise, the overall complexity of the MI method scales linearly with the number of scattering energies n_E, see equation (37). For the WPCD method we get all the on-shell energy scattering-solutions from a single Hamiltonian diagonalisation, giving a very 'cheap' scaling with n_E.

4.2.2. Parallel complexity. The parallel WPCD approach is based on simultaneous solutions of the LS equation for all on-shell energies. Contrary to the sequential approach, we assume a sufficiently large set of processors (like in a GPU) to handle all on-shell energies at once. This will remove the factor n_E in the complexity model (40) such that the cubically-scaling term will clearly become the dominating one. It can be combatted by making use of hardware-specific, massively parallel algorithms for matrix diagonalisation and matrix–matrix multiplications. Here we present the parallel approach to the same steps as above, and in the same ordering. We assume we have p compute processors, or threads, available.

(a) The parallel cyclic-order Jacobi method is a parallel approach to the Jacobi eigenvalue algorithm: a method based on finding a similarity transformation of a matrix to its diagonal form by repeated Jacobi rotations (see e.g. reference [44]). This method has held major appeal for parallel computing due to its inherent parallelism compared to other eigenvalue-finding routines. However, parallel optimisation of the basic algorithm depends greatly on how a set of processors is organised with regards to communication and memory. Therefore, there is no general complexity model for the method—any algorithm should be written with a specific hardware in mind. One approach can be found in reference [45] with a demonstrated complexity $O\left(\frac{n^3}{p}\ \log\ n\right)$ for convergence for a symmetric and real $n \times n$ matrix.

(b) Parallel matrix–matrix multiplication algorithms is an ongoing field of research (see e.g. reference [46]). While there exist very efficient methods, such as Cannon's algorithm [47], they depend highly on hardware and matrix characteristics (sparsity, symmetry, etc). The divide-and-conquer approach is a very general and straightforward way to parallelise matrix multiplications. In essence, we divide the matrices into $M = \left\lceil \frac{n^2}{p} \right\rceil$ submatrices of sizes $m \times m = p$. If combined with on-chip shared memory between the processors, this permits all processors to simultaneously work on calculating the product while minimising processor-to-processor communication time. The method is explained excellently in literature such as [44]. This simple method has a theoretical minimum complexity of $O\,(n \times M)$ when performed in parallel[5].

(c) There is limited parallel optimisation to be gained in equation (39). We can multiply each term of the summation in parallel, and do the whole summation sequentially for the sake of simplicity. The result is a complexity of $O(2(N_{\text{WP}} + 2))$ FLOP. Note, however, that the inherent parallelism of the WPCD method allows us to do the summation simultaneously for all E and thereby effectively removing the factor n_E seen so far in the complexity models.

In conclusion, assuming $p > N_{\text{WP}}$ and adequately large shared memory (see point 2 above), a somewhat optimal and parallel WPCD complexity model is given by

$$\begin{aligned} O_{\text{WPCD,par.}}(T) = &\left\lceil \frac{N_{\text{WP}}^3}{p} \right\rceil \log(N_{\text{WP}}) \\ &+ N_{\text{WP}} \left\lceil \frac{N_{\text{WP}}^2}{p} \right\rceil + 2(N_{\text{WP}} + 2). \end{aligned} \tag{41}$$

We see from figure 2 that the efficiency of the parallel WPCD approach scales very well with the number of available processors. We also see that for a single processor ($p = 1$) the parallel and sequential approaches are roughly equal (while not taking into account any overhead in memory transfers), while for a realistic value $p = 1024$ the complexity model for the parallel approach demonstrates a clear advantage. The value $p = 1024$ corresponds to the current limit on threads with shared memory for typical Nvidia GPUs.

5. Continuum-discretised neutron–proton scattering computations

In this section we present a detailed analysis of the precision and accuracy of the WPCD method for computing neutron–proton (np) scattering observables and phase shifts. For all calculations,

[5] This is not the full picture. Often, a set of processors is divided in groups of processors with limited shared memory that cannot fit all p elements from each matrix in the matrix–matrix product, and one has to define more submatrices than given by M. This introduces yet another complication to the parallel complexity model that has to do with shared memory size, which we will not account for here.

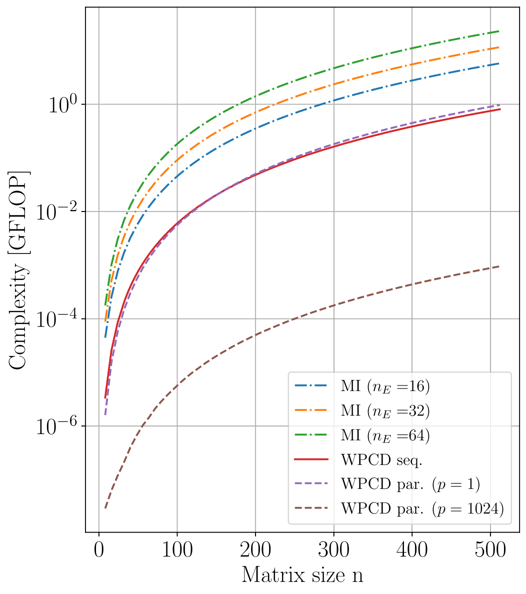

Figure 2. Complexity, as measured in FLOP, for the MI and WPCD methods as a function of n, where n represents the number of quadrature points (N_{Q}) for MI and the number of wave packets (N_{WP}) for WPCD. Furthermore, n_E is the number of on-shell T-matrix evaluations (note $n_E = n = N_{\mathrm{WP}}$ for WPCD).

we employed the optimised next-to-next-to-leading-order chiral potential N2LO$_{\mathrm{opt}}$ [48]. The primary goal is to analyse the trade-off between minimum computational cost and maximum method accuracy in the WPCD method, and contrast this to the conventional MI method. Note that there is no problem in obtaining highly accurate results from either method. We simply focus on the performance and accuracy of the WPCD method as we reduce the number of wave-packets such that all objects fit in the fast on-chip shared memory on the GPU. In our analysis we consider the MI method with $N_{\mathrm{Q}} = 96$ Gauss–Legendre points, see equation (4), to yield virtually exact results. For brevity, we will refer to such numerically converged calculations as *exact*.

5.1. Computing phase shifts

In figure 3 we compare np scattering phase shifts in the $^1\mathrm{S}_0$, $^3\mathrm{P}_0$, and $^3\mathrm{S}_1$ partial waves as well as the ϵ_1 mixing angle, as obtained from the WPCD method using $N_{\mathrm{WP}} = 32$ and $N_{\mathrm{WP}} = 64$ momentum wave-packets. The FWP bin boundaries follow a Chebyshev distribution with scaling factor $\alpha = 100$ MeV and sparseness degree $t = 2$, see equation (31), and the resolvent was energy-averaged according to equation (B.7).

For the WPCD-result, one immediately observes a discretised, step-like version of the otherwise smoothly varying *exact* description of the scattering phase shifts. This is entirely due to the momentum discretisation and finite number of wave packets. Indeed, due to the energy-averaging we obtain only one T-matrix element per momentum bin. In the limit $N_{\mathrm{WP}} \to \infty$, and everything else equal, we will recover the *exact* result obtained with the MI method. We observe this convergence trend already when going from $N_{\mathrm{WP}} = 32$ to $N_{\mathrm{WP}} = 64$ wave packets.

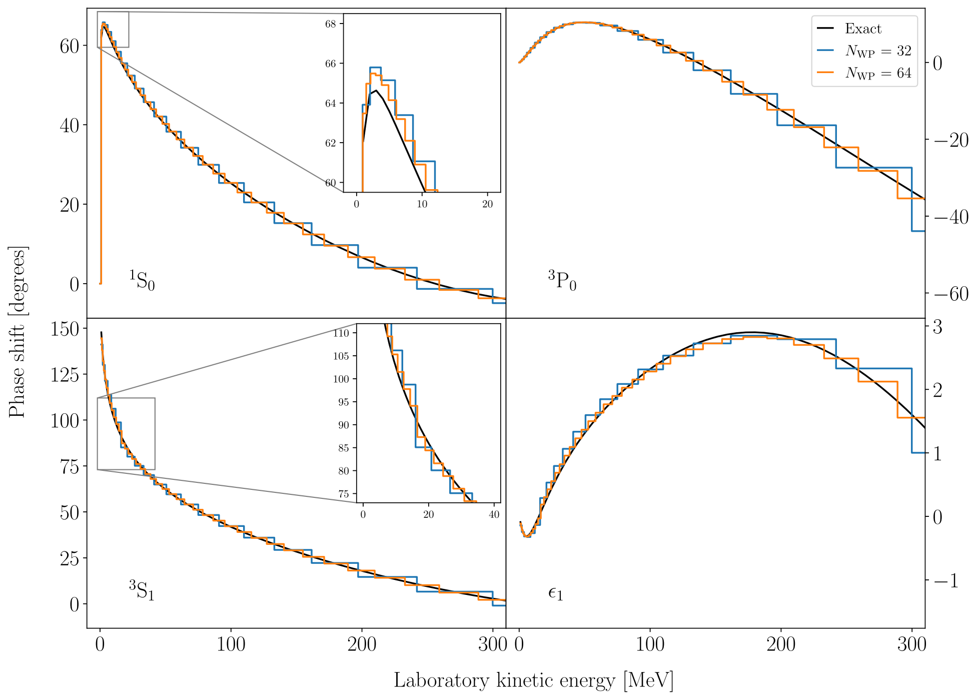

Figure 3. Phase shifts calculated using the WPCD-method using $N_{\mathrm{WP}} = 32$ (blue) and $N_{\mathrm{WP}} = 64$ (orange) wave packets. Numerically *exact* results (black) were obtained using the MI-method.

For instance, the mixing angle moves closer to the *exact* results when increasing N_{WP}. Overall, the WPCD method is performing as expected, but there are two features we would like to point out:

(a) The values of the low-energy $^1\mathrm{S}_0$ phase shift are overestimated near the peak where the phase shift turns over. This is likely due to the momentum-averaging of operators. The potential matrix in the $^1\mathrm{S}_0$ channel ($\langle q'|V_{^1\mathrm{S}_0}|q\rangle$) is shown in figure 4 for both a continuous momentum-basis and a $N_{\mathrm{WP}} = 32$ wave-packet basis with bins distributed according to equation (31). The potential is constant within each wave-packet bin as expected according to equation (29). This makes it challenging to reproduce finer details of the interaction. There is also a discrepancy between the continuous and wave-packet values when the chosen q'-momentum is not near any bra-state bin midpoint. This is most distinctive in the green curves at $q' = 205.622$ MeV, which is very close to a bin boundary. We see in equation (29) that we average over momenta within two bins, and when a potential varies strongly within a bin such that the matrix elements at the bin boundaries are quite different from the bin mid-point values, this averaging is too coarse to mimic the potential accurately. This effect becomes less significant with increasing N_{WP} since the grid will grow denser.

(b) The WPCD-results for $^3\mathrm{S}_1$ show a distinctive drop at 90 degrees. This trend is consistent for all basis sizes and is simply due to the treatment of the deuteron bound state in the WPCD framework. We extract a phase shift δ via an inverse trigonometric function with values for $\delta \in [-90, 90]$ degrees. A bound state is characterised by a transition

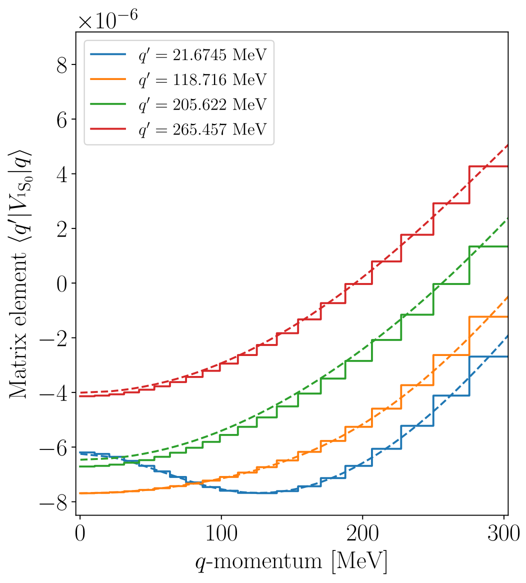

Figure 4. The $^1S_0$ potential of NNLO$_{\mathrm{opt}}$ [48] as a function of q, at several q' values. The dashed lines represent the potential calculated in a continuous basis, while the solid lines represent a wave-packet projection. The wave-packet projection was made with an $N_{\mathrm{WP}} = 32$ basis in a Chebyshev distribution (with $\alpha = 100$ MeV). In the wave-packet representation, the bra-state bin contains q'. Note that the units for continuous representation of the potential is in units of MeV^{-2} while the wave-packet projected potential is in units of MeV, as follows from the definition in equation (17).

$\delta(E+\epsilon) - \delta(E) = 180$ degrees at some energy E for an infinitesimal step $\epsilon > 0$, as dictated by Levinson's theorem [49]. It is apparent that this transition across 90 degrees is more difficult to reproduce for the WPCD method. The same effect also gives rise to inaccuracies in the ϵ_1 mixing angle. However, this is of no concern when computing a scattering observable.

5.1.1. Linearly interpolating phase shifts. Although the resolution of the WPCD method is limited by the energy-averaging across each momentum-bin—i.e., yields a step-like character of the predictions—the mid-points $\bar{q}_i = \frac{1}{2}(q_{i-1} + q_i)$ of each bin in figure 3 are typically closest to the *exact* results, as expected from the wave-packet eigenvalues, see equations (18) and (19). We can therefore linearly interpolate the phase shifts $\delta_i \equiv \delta(\bar{q}_i)$ across several bins, i.e. across scattering energies, via

$$\delta(q) = \left(\frac{\delta_i - \delta_{i-1}}{\bar{q}_i - \bar{q}_{i-1}}\right) q + \left(\delta_{i-1} - \frac{\delta_i - \delta_{i-1}}{\bar{q}_i - \bar{q}_{i-1}}\bar{q}_{i-1}\right), \tag{42}$$

for c.m. momenta $q \in [q_{i-1}, q_i]$ where q_i are the FWP bin boundaries. This simple and straightforward approach offers a rather precise prediction for any on-shell scattering energy. Of course, the phase shifts can be linearly interpolated using other points, i.e. we can let

$$\bar{q}_i = q_{i-1} + \frac{n}{m}(q_i - q_{i-1}), \tag{43}$$

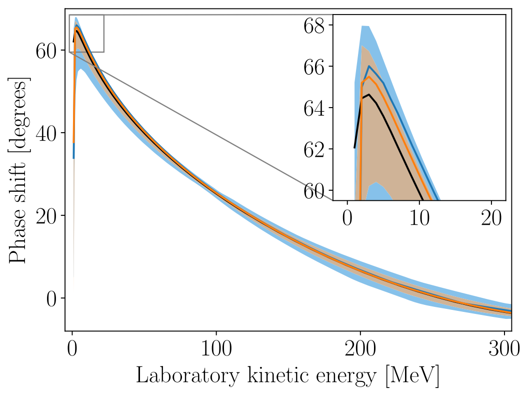

Figure 5. Phase shifts for the ${}^1S_0$ partial wave calculated using WPCD. The colour coding is the same as in figure 3, i.e. $N_{WP} = 32$ (light blue) and $N_{WP} = 64$ (light orange). The bands indicate the maximum discrepancy due to the choice of interpolation point n according to equation (43).

for some $n \in [0, m]$, such that $n = \frac{m}{2}$ is the bin mid-point again. Figure 5 shows the resulting predictions using linear (mid-point) interpolation of the ${}^1S_0$ phases presented in figure 3. The bands estimate the effect of varying the interpolation point. The bands span the resulting $\delta(q)$ calculated with $m = 10$ and $n = [0.1, 1, 2, \ldots, 8, 9, 9.9]$ in equation (43). As expected, mid-point interpolation yields results that are very close to the *exact* calculation. In the following we will therefore only use this interpolation choice.

5.2. Computing cross sections

We compute scattering cross sections from scattering amplitudes according to the method outlined in appendix A. Figure 6 shows the total cross section obtained using the WPCD-method using momentum-space bins with $N_{WP} = 32$ and $N_{WP} = 64$. As can be seen in the figure, the linearly-interpolated results reproduce the *exact* result rather well. On a larger scale it is nearly impossible to tell any difference between results obtained using $N_{WP} = 32$ bins and $N_{WP} = 64$ bins. To emphasise the monotonically increasing accuracy of the WPCD method as we increase the number of momentum-space bins, we calculate the absolute value of the difference between the *exact* and the WPCD prediction for the total cross section for a range of bin resolutions, see figure 7. From this it is apparent that the WPCD method converges, although slowly, for a bin partition following a Chebyshev distribution.

5.3. WPCD accuracy

There are primarily two approximations that impact the accuracy of the WPCD method: the number of bins N_{WP} and the distribution thereof. Also, the averaging of continuous states into wave packets means we use momentum-averaged matrix representations of operators in the LS equation. This averaging should improve with reduced bin widths. We can easily control N_{WP}, and in this section we analyse the performance of WPCD with respect to different choices of this method parameter.

To better quantify the convergence of the WPCD method we study the root-mean-square error (RMSE) with respect to the exact result for the total cross section across a range of

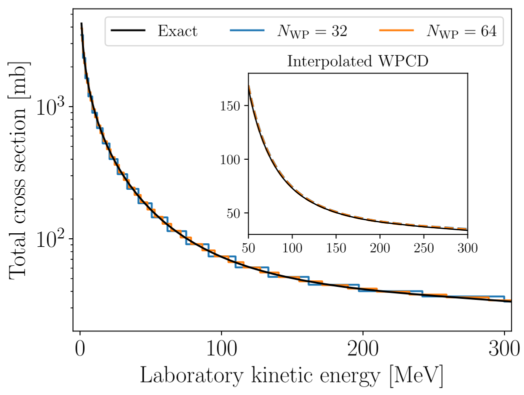

Figure 6. Total cross section calculated with the WPCD method. The solid black line corresponds to the *exact* result as obtained using the MI method. The inset demonstrates the high accuracy of (mid-point) interpolated WPCD results. On this scale, the two separate WPCD predictions appear to overlap completely.

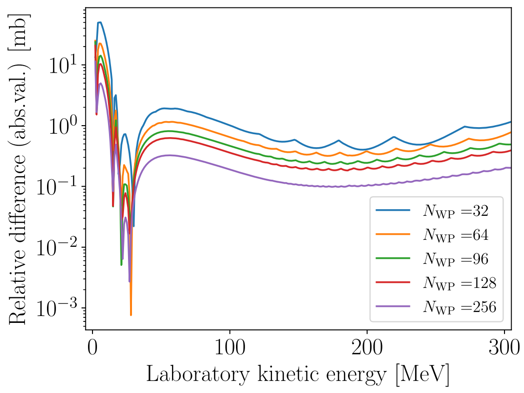

Figure 7. Absolute values of the relative difference between *exact* results and WPCD method for the total cross section, shown as a function of wave-packet basis size N_{WP} and laboratory kinetic energy.

scattering energies. We use the standard RMSE measure

$$\mathrm{RMSE} = \sqrt{\frac{\sum_{i=1}^{n_E} \left(\sigma_{\mathrm{exact},i} - \sigma_{\mathrm{method},i}\right)^2}{n_E}}, \tag{44}$$

where $\sigma_{\mathrm{exact},i}$ denotes the *exact* results and $\sigma_{\mathrm{method},i}$ denotes either the WPCD- or MI-calculated total cross sections at some scattering energy E_i for $i = 1, \ldots, n_E$.

We find that the RMSE for WPCD with scattering energies corresponding to laboratory kinetic energies $40 \leqslant T_{\mathrm{lab}} \leqslant 350$ MeV remains fairly constant at $\sim$2.0 mb when using $N_{\mathrm{WP}} = 16$. This is interesting for three reasons:

- The WPCD coupled channel Hamiltonian will be of size $2N_{\mathrm{WP}} \times 2N_{\mathrm{WP}}$, i.e. we diagonalise 32×32 Hamiltonian matrices. Most GPUs today, including the Nvidia Tesla V100

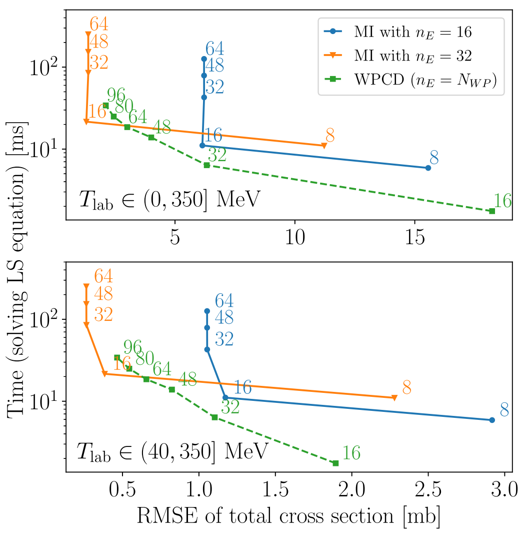

Figure 8. Measured wall times for solving the LS equation for different values of total cross section RMSE for $T_{\text{lab}} \in (0, 350]$ MeV (top panel) and $T_{\text{lab}} \in (40, 350]$ MeV (bottom panel). n_E is the number of on-shell energies (interpolation points) at WP bin midpoints, according to a Chebyshev distribution. The numeric label attached to each data point indicates the number of Gauss–Legendre grid points N_Q for the MI method and the NWP N_{WP} for the WPCD method. The number of energies used for interpolating the MI results are indicated in the legend. See the main text for hardware specifications.

we have used here, have 64 kB shared memory (also called 'on-chip' memory) which is significantly faster to access than the GPU's main memory. These matrices fit entirely on the GPU shared memory, allowing for a strong reduction in GPU memory read/write demand while performing the diagonalisation.

- A recent Bayesian analysis of chiral interactions suggests that the χEFT truncation error for scattering cross sections at next-to-next-to-leading-order is at least 2 mb, at 68% degree-of-belief (DoB), and at least 5 mb at 95% DoB [20].
- Regarding experimental uncertainties, the combined statistical and systematic uncertainties in measured np total cross sections are at the 1% level [50, 51]. In absolute terms, this amounts to uncertainties of the order 1 mb at laboratory scattering energies $\gtrsim$40 MeV. At lower energies the cross section increases of course, leading to slightly larger absolute experimental errors.

In summary, we find that the WPCD method error, when using very few wave-packets, is slightly larger than typical experimental uncertainties but smaller than the estimated model discrepancy up to next-to-next-to-leading-order in χEFT.

One can clearly reduce the WPCD method error by increasing N_{WP}. However, this will also increase the computational cost and we recommend studying the actual wall time cost in relation to, e.g. the RMSE measure defined above. We perform such an analysis in section 6.

Before ending this section on the method accuracy, we would like to emphasise that the maximum total angular momentum $J_{\max}$ in the partial-wave expansion of the potential also impacts the accuracy of the description of the scattering amplitude. For the linearly-interpolated WPCD method with $N_{\text{WP}} = 16$, where we observe a method error of $\sim$2 mb in the total cross section at $T_{\text{Lab}} \gtrsim 40$ MeV, and find it unnecessary to go beyond $J_{\max} = 6$.

Table 1. Measured wall times, corresponding to the results in figure 8, for solving the LS equation and predicting total cross sections using the WPCD and MI methods, at different RMSE levels in the $T_{\text{lab}} \in (40, 350]$ MeV energy region. The n_E parameter indicates the number of interpolation energies used for the MI method.

	MI			WPCD	
RMSE (mb)	N_{Q}	n_E	Time (ms)	N_{WP}	Time (ms)
3.0	8	16	6	—	—
2.5	10	16	7	—	—
2.0	12	16	8	16	0.5
1.5	14	16	10	24	2.5
1.0	16	24	15	32	6
0.5	16	32	20	96	35

6. Method performance

With an account of the precision and accuracy of the WPCD method we now profile its time performance. Typically, when using the MI method, we calculate every on-shell phase shift of interest by explicitly solving the LS equation, while in the WPCD approach we interpolate in-between bin mid-points, as discussed above. This will, unsurprisingly, induce a substantial time-performance penalty in using the MI method, due to the linear scaling with the number of energy solutions, as shown in equation (37). However, we can of course linearly interpolate the phase shifts calculated with the MI method, rather than invoking an explicit calculation at every on-shell energy. Therefore, to facilitate a balanced comparison between the two methods, we also employed linear interpolation to extract phase shifts when using the MI method. In our studies, this has turned out to be a highly efficient way to speed up the calculation with the MI method while maintaining precision and accuracy of the results.

To facilitate a comparison, when using the MI method we solve the LS equation at n_E on-shell energies also following a Chebyshev distribution. We then linearly interpolate the phase shifts using these energies to calculate neutron–proton total cross sections in the T_{lab} energy ranges $(0, 350]$ MeV and $(40, 350]$ MeV for the calculation of RMSE values.

For WPCD we can only vary the number of wave packets N_{WP} while for MI we can vary both N_{Q} and n_E, i.e. the number quadrature points and the number of interpolation points or on-shell energies (at bin mid-points), respectively. Figure 8 shows the wall times for solving the LS equation to obtain cross sections, at different levels of method accuracy measured by the RMSE value. In table 1 we show a few interpretations of the figure for a handful of relevant method parameters. As mentioned, the WPCD method is implemented on a GPU while the MI results were obtained using an optimised CPU implementation [7]. The time profiles were obtained using an Nvidia Tesla V100 32 GB SMX2 and an Intel Xeon Gold 6130 for the WPCD GPU and MI CPU results, respectively.

From our analysis we conclude that the WPCD method is faster than the MI method if one can tolerate $\sim 1 - 5$ mb method RMSE in the prediction of total scattering cross sections. For such applications it would then be advisable to use $N_{\text{WP}} \gtrsim 48$ bins, on the basis of figure 8. It is worth noting that the RMSE is dominated by contributions from scattering cross sections below laboratory kinetic energies $T_{\text{lab}} \sim 40$ MeV. Indeed, with $N_{\text{WP}} = 48$ we obtain an RMSE value of ~ 4 mb across an interval $T_{\text{lab}} \in (0, 350]$ MeV. The RMSE drops to ~ 0.8 mb when considering only cross sections with $T_{\text{lab}} > 40$ MeV. Irrespective of how the WP bins are distributed, it is therefore necessary to ensure a sufficiently high density of bins below $T_{\text{lab}} = 40$ MeV to accurately reproduce the $^1S_0$ phase-shift peak and the $^3S_1$ bound state. With the WPCD method we can obtain increasingly faster solutions to the LS equation as we reduce the number of wave-packets.

7. Summary and outlook

In this study we have compared the WPCD method with the standard MI method, with an emphasis on their respective efficiency, precision, and accuracy when solving NN scattering problems. This study is done with an interest in reducing the computational costs of Bayesian inference analyses of nuclear interaction models.

We find that the WPCD method, with GPU-acceleration, is capable of providing scattering solutions to the NN LS equation much faster than a conventional MI method implemented on a CPU. This is largely due to the WPCD method being capable of delivering scattering amplitudes at several on-shell energies in an inherently parallel fashion, and that linear interpolation across several WP bins offers straightforward predictions for any scattering energy. However, compared to solutions obtained using the standard MI method, the WPCD method is less accurate, in particular in regions where the scattering amplitudes vary strongly with energy. This is also expected due to the momentum-space discretisation and approximation of the scattering states. Nevertheless, in applications where a certain method error can be tolerated—as for example in low-order EFT predictions of NN scattering cross sections—the GPU-implemented WPCD method presents a computational advantage. This finding makes it particularly promising for computational statistics analyses of EFT nuclear interactions utilising Bayesian inference methods.

We also find that the computational gain of using GPU hardware is limited by the amount of shared memory that is available. This constraint basically corresponds to an upper limit on the number of WP bins that can be used for efficient computations. We note, however, that GPU hardware is continuously improved, and that a four-fold increase in the size of the fast shared memory on the GPU would enable a two-fold increase of both the WP-basis and method accuracy while incurring a very mild additional computational cost. The GPU global memory is typically not saturated in WPCD calculations of NN scattering calculations. The WPCD method can also be applied to compute three-nucleon scattering amplitudes [37, 52]. However, the limited amount of global memory might become a constraining factor for such applications that involve a more complicated Hilbert space basis.

In this study we have focussed on the inherent parallelism of the WPCD method and the opportunities that it offers to benefit from the use of GPU hardware. Of course, one could also explore parallelisation of the WPCD method on a CPU and make parallel use of the fast cores available in modern CPUs. The efficiency of the GPU WPCD approach is almost fully determined by the time needed to diagonalize the channel Hamiltonian and could therefore benefit from the faster CPU clock speed. Naturally, one could also consider a GPU implementation of the MI method wherein the LS equation is solved in a parallel fashion for multiple scattering energies simultaneously. Still, this would require multiple matrix inversions (one for each interpolation energy). A simple phase-shift interpolation applied to the MI method facilitates sufficiently accurate results for NN scattering observables, at a significantly lower computational cost compared to explicitly solving the LS equation at every scattering energy of interest.

Acknowledgments

This work was supported by the European Research Council (ERC) under the European Unions Horizon 2020 research and innovation programme (Grant Agreement No. 758027), and the Swedish Research Council (Grant No. 2017-04234). The computations were enabled by resources provided by the Swedish National Infrastructure for Computing (SNIC) at Chalmers Centre for Computational Science and Engineering (C3SE) and the National Supercomputer Centre (NSC) partially funded by the Swedish Research Council.

Data availability statement

The data that support the findings of this study are available upon reasonable request from the authors.

Appendix A. Scattering cross sections and the spin-scattering matrix

A scattering observable of some operator $\hat{O}$ can generally be written as a trace [28],

$$\langle \hat{O} \rangle = \frac{\mathrm{Tr}\{M\hat{\rho}_i M^\dagger \hat{O}\}}{\mathrm{Tr}\{M\hat{\rho}_i M^\dagger\}}, \tag{A.1}$$

where $\hat{\rho}$ is the spin-density matrix, usually represented in a helicity basis, and M is the spin-scattering matrix.

The helicity-basis projection of the spin-scattering matrix is written as $M^s_{m_s,m_{s'}}$, where s is the conserved total spin and m_s is the corresponding projected spin. These matrices are usually expressed in partial-wave expansions,

$$\begin{aligned} M^s_{m_s,m_{s'}} &= \frac{2\pi}{ip}\sum_{J,l,l'}[1-(-1)^{l+s+T_Z}]\sqrt{\frac{2l+1}{4\pi}} \\ &\quad\times Y^{l'}_{m_s-m_{s'}}(\theta,\phi) \\ &\quad\times \langle l', s, m_s-m_{s'}, m_{s'}|l', s', j, m_s\rangle \\ &\quad\times \langle l, s, m_s, 0|l, s, j, m_s\rangle \\ &\quad\times \left[S^{1J}_{l'l}(p',p)-\delta_{l,l'}\right], \end{aligned} \tag{A.2}$$

where the third and fourth rows are Clebsch–Gordan coefficients, J is the total angular momentum, l and l' are the orbital angular momenta of the inbound and outbound states respectively, T_Z is the azimuthal isospin projection, $Y^l_m(\theta,\phi)$ is an azimuthal spherical harmonic, and $S^{1J}_{l'l}(p',p) = \langle p', l', s, J|S|p, l, s, J\rangle$ is the usual scattering matrix.

The on-shell partial-wave-projected S-matrix is related to the on-shell $T(p,p;E)$-matrix element via

$$S^{sJ}_{l'l}(p,p;E) = 1 - 2\pi i T^{sJ}_{l'l}(p,p;E). \tag{A.3}$$

Due to the conservation of probability, the S-matrix is unitary, and can therefore be parametrised by real parameters like phase shifts. In the Stapp convention [53], the coupled-channel S-matrix $S^{1J}_{l'l}$ is then given by

$$S = \begin{pmatrix} \cos(2\epsilon_J)\mathrm{e}^{2\mathrm{i}\delta_{-,J}} & i\,\sin(2\epsilon_J)\mathrm{e}^{\mathrm{i}(\delta_{-,J}+\delta_{+,J})} \\ i\,\sin(2\epsilon_J)\mathrm{e}^{\mathrm{i}(\delta_{-,J}+\delta_{+,J})} & \cos(2\epsilon_J)\mathrm{e}^{2\mathrm{i}\delta_{+,J}} \end{pmatrix}, \tag{A.4}$$

where $\delta_{\pm,J}$ refers to $\delta_{l,J}$ for $l = J \pm 1$ and ϵ_J is the mixing angle. The uncoupled-channel S-matrix S^{0J}_{ll} is given by (for $l = J$)

$$S = \mathrm{e}^{2\mathrm{i}\delta_{J,J}}. \tag{A.5}$$

The angle θ denotes the scattering angle between the c.m. inbound and outbound relative momenta $\boldsymbol{p}$ and $\boldsymbol{p}'$ respectively, while ϕ is the rotation angle of $\boldsymbol{p}'$ around the inbound momentum $\boldsymbol{p}$, but cylindrical symmetry allows us to set $\phi = 0$. Using equation (A.2) we can calculate scattering observables such as the differential cross section,

$$\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega} = \frac{1}{4}\,\mathrm{Tr}(MM^\dagger). \tag{A.6}$$

However, this approach does not utilise symmetries to reduce the number of non-contributing terms of $M^s_{m_s,m_{s'}}$. Instead, the M-matrix can be expressed in terms of non-vanishing spin-momentum products after some consideration of parity conservation, isospin and time-reversal symmetries, and the Pauli principle. We use the Saclay convention [54] for these terms,

$$\begin{aligned} M = \frac{1}{2}\,[&(a+b) + (a-b)(\boldsymbol{\sigma}_1\cdot\boldsymbol{n})(\boldsymbol{\sigma}_2\cdot\boldsymbol{n}) \\ &+ (c+d)(\boldsymbol{\sigma}_1\cdot\boldsymbol{m})(\boldsymbol{\sigma}_2\cdot\boldsymbol{m}) \\ &+ (c-d)(\boldsymbol{\sigma}_1\cdot\boldsymbol{l})(\boldsymbol{\sigma}_2\cdot\boldsymbol{l}) \\ &+ e((\boldsymbol{\sigma}_1+\boldsymbol{\sigma}_2)\cdot\boldsymbol{m})], \end{aligned} \tag{A.7}$$

where a, b, c, d, and e are the Saclay amplitudes, $\boldsymbol{\sigma}_i$ are the Pauli spin matrices acting on nucleon $i = 1, 2$, and where we define the following unit vectors

$$\boldsymbol{l} \equiv \frac{\boldsymbol{p}+\boldsymbol{p}'}{|\boldsymbol{p}+\boldsymbol{p}'|}, \qquad \boldsymbol{m} \equiv \frac{\boldsymbol{p}-\boldsymbol{p}'}{|\boldsymbol{p}-\boldsymbol{p}'|}, \qquad \boldsymbol{n} \equiv \frac{\boldsymbol{p}\times\boldsymbol{p}'}{|\boldsymbol{p}\times\boldsymbol{p}'|}. \tag{A.8}$$

The Saclay amplitudes are given by the spin-projected matrices as,

$$\begin{aligned} a &= \frac{1}{2}(M^1_{11} + M^1_{0,0} + M^1_{1,-1}), \\ b &= \frac{1}{2}(M^1_{11} + M^0_{0,0} + M^1_{1,-1}), \\ c &= \frac{1}{2}(M^1_{11} - M^0_{0,0} + M^1_{1,-1}), \\ d &= -\frac{1}{\sqrt{2}\,\sin(\theta)}(M^1_{1,0} + M^1_{0,1}), \\ e &= \frac{i}{2\sqrt{2}}(M^1_{1,0} - M^1_{0,1}). \end{aligned} \tag{A.9}$$

In this parametrisation, the differential cross-section is given by

$$\frac{\mathrm{d}\sigma}{\mathrm{d}\Omega} = \frac{1}{2}\left(|a|^2 + |b|^2 + |c|^2 + |d|^2 + |e|^2\right). \tag{A.10}$$

For the results in this paper we used the following expression for the total cross section:

$$\sigma_{\mathrm{tot}} = \frac{2\pi}{p}\,\mathrm{Im}(a+b). \tag{A.11}$$

See reference [54] for a complete account of scattering observables in the Saclay parametrisation.

Appendix B. The resolvent in a wave-packet basis

The resolvent $\hat{g}(E)$ for the full Hamiltonian $\hat{h} \equiv \hat{h}_0 + \hat{v}$,

$$\hat{g}(E) \equiv \frac{1}{E - \hat{h} \pm i\epsilon}, \tag{B.1}$$

can be calculated analytically in a pseudostate wave-packet basis $\{|z_i\rangle\}_{i=1}^n$ of the full Hamiltonian. The resolvent is represented in the basis by (see equation (17))

$$\begin{aligned}\langle z_i|\hat{g}(E)|z_j\rangle &= \langle z_i|\frac{1}{E - \hat{h} \pm i\epsilon}|z_j\rangle \\ &= \frac{1}{\mu\sqrt{N_i N_j}}\int_{\mathcal{D}_i}\int_{\mathcal{D}_j}\mathrm{d}k'\mathrm{d}k\frac{k'k\sqrt{k'k}\langle\psi_{k'}^{(+)}|\psi_k^{(+)}\rangle}{E - \frac{k'^2}{2\mu} \pm i\epsilon},\end{aligned} \tag{B.2}$$

where $\mu = \frac{m_{\mathrm{N}}}{2}$. Note that we set the weight function $f(k) = \sqrt{\frac{k}{\mu}}$ and normalisation $N_i = \Delta\mathcal{E}_i$ as these are energy wave-packets (from the diagonalisation of $\hat{h}$). Using $\langle\psi_{k'}^{(+)}|\psi_k^{(+)}\rangle = \frac{\delta(k'-k)}{k'k}$, this becomes

$$\langle z_i|\hat{g}(E)|z_j\rangle = \frac{\delta_{ij}}{\mu N_i}\int_{\mathcal{D}_i}\mathrm{d}k\,\frac{k}{E - \frac{k^2}{2\mu} \pm i\epsilon}, \tag{B.3}$$

where we have introduced the Kronecker delta δ_{ij} since $\langle\psi_{k'}^{(+)}|\psi_k^{(+)}\rangle = 0\,\forall\,\mathcal{D}_i \neq \mathcal{D}_j$. For positive energies, where $E = \frac{p^2}{2\,\mu}$ and where p is the on-shell c.m. momentum, we get

$$\langle z_i|\hat{g}(E)|z_j\rangle = \frac{2\delta_{ij}}{D_i}\int_{\mathcal{D}_i}\mathrm{d}k\,\frac{k}{p^2 - k^2 \pm i\epsilon}. \tag{B.4}$$

If $E \notin \mathcal{D}_i$, we take the limit $\epsilon \to 0$ and solve the integral to find

$$\begin{aligned}\langle z_i|\hat{g}(E)|z_j\rangle &= \frac{2\delta_{ij}}{N_i}\int_{\mathcal{D}_i}\mathrm{d}k\,\frac{k}{p^2 - k^2} \\ &= \frac{\delta_{ij}}{N_i}\left[-\ln\left|\frac{k^2}{p^2} - 1\right|\right]_{k_i}^{k_{i+1}} \\ &= \frac{\delta_{ij}}{N_i}\ln\left|\frac{E + \mathcal{E}_{i+1}}{E + \mathcal{E}_i}\right|,\end{aligned} \tag{B.5}$$

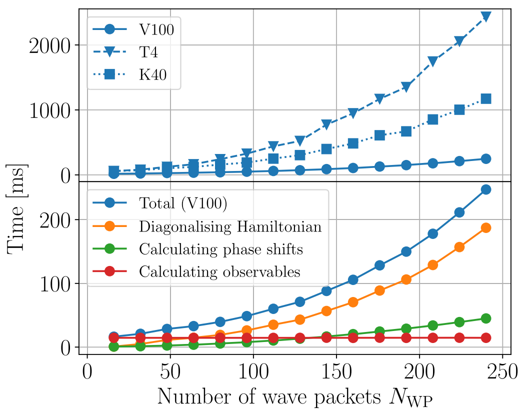

Figure B1. Top: total time used in calculating on-shell T-matrix elements and extracting phase shifts and observables, shown for three different Nvidia GPUs of the Tesla line; T4 (orange), V100 (blue), and K40 (green). Bottom: time decomposition of V100 total time, with focus on key parts of calculating NN scattering using the WPCD method. The green line includes the time spent solving the LS equation.

where $p \notin \mathcal{D}_i$, and $\mathcal{E}_i$ and $\mathcal{E}_{i+1}$ is the lower and upper boundary of $\mathcal{D}_i$ expressed in energy, respectively. If $E \in \mathcal{D}_i$, then we have a simple pole at $p = k$. The pole-integration is done using the infinitesimal complex rotation $\pm i\epsilon$ together with the residue theorem, giving

$$\langle z_i|\hat{g}(E)|z_j\rangle = \frac{\delta_{ij}}{N_i}\left[\ln\left|\frac{E+\mathcal{E}_{i+1}}{E+\mathcal{E}_i}\right| - i\pi(\theta(E-\mathcal{E}_i) - \theta(E-\mathcal{E}_{i+1}))\right], \tag{B.6}$$

where θ is the Heaviside step-function. The derivation of the resolvent expressed in a momentum wave-packet representation follows a similar procedure. In that case, it is possible to use momentum wave-packets where $f(p) = 1$ and $N_i = \sqrt{k_{i+1} - k_i}$, in which case the derivation above changes a little, see [18].

Energy averaging of the resolvent is done by integrating the resolvent with respect to E, in the bin $E \in \mathcal{D}_k$, divided by the bin width $\Delta\mathcal{E}_k$. We introduce the denotation $\bar{g}^k_{ij}$ to reflect this. The derivation is straightforward:

$$\begin{aligned}\bar{g}^k_{ij} &\equiv \frac{1}{\Delta\mathcal{E}_k}\int_{\mathcal{D}_k}\mathrm{d}E\,\langle z_i|\hat{g}(E)|z_j\rangle \\ &= \frac{\delta_{ij}}{\Delta\mathcal{E}_k N_i}\int_{\mathcal{D}_k}\mathrm{d}E\left[\ln\left|\frac{E+\mathcal{E}_{i+1}}{E+\mathcal{E}_i}\right| - i\pi\delta_{ik}\right] \\ &= \frac{\delta_{ij}}{\Delta\mathcal{E}_k\Delta\mathcal{E}_i}W_{ki} - \frac{i\pi}{\Delta\mathcal{E}_k}\delta_{ik},\end{aligned} \tag{B.7}$$

where we used $N_i = \Delta\mathcal{E}_i$, and

$$W_{ki} \equiv \sum_{k'=k}^{k+1}\sum_{i'=i}^{i+1}(-1)^{k-k'+i-i'}\left[\mathcal{E}_{k'} - \mathcal{E}_{i'}\right]\ln\left|\mathcal{E}_{k'} - \mathcal{E}_{i'}\right|, \tag{B.8}$$

as presented in [18].

Appendix C. GPU code

The code for GPU-utilisation was written to make use of the CUDA interface, which is developed and maintained by the Nvidia Corporation. CUDA allows for efficient utilisation of a Nvidia GPU using high-level programing languages such as C and C++, Python, or Fortran. For numerically demanding linear algebra operations we made use of the CUDA-libraries cuBLAS [42] and cuSOLVER [43]. These libraries are very similar in use to BLAS and LAPACK—two standard libraries for linear algebra on the CPU.

Diagonalising the full Hamiltonian and solving the LS equation are the most numerically demanding parts of the steps presented in section 3.2. These steps are therefore solved on the GPU. The ability to work on several channels simultaneously—this being the advantage of the GPU—is made possible by using CUDA's *batched*-routines. These are pre-written routines that maximise the efficiency of the GPU to solve several linear algebra problems simultaneously for small matrix-sizes (typically less than 1000×1000 in matrix dimension). Such efficient parallelism is generally difficult to achieve 'by hand' due to the massive load on GPU-memory read-write accesses.

The importance of efficient memory use is made apparent in figure B1 where we show the computation time spent by three different Nvidia GPUs (top panel) and the decomposition of time used for the Nvidia V100 GPU (bottom panel). We see that the majority of the total time is spent on the Hamiltonian diagonalisation. A large fraction of the diagonalisation task is spent on memory accesses as part of the Jacobi method. The three GPUs: V100, T4, and K40, have differences in memory technology [55] which is the main reason for the observed differences in the performance.

We used `cublas<t>gemmStridedBatched` to calculate the VC-matrix product in equation (32). This calculates a matrix-product on the form $C \leftarrow \alpha AB + \beta C$, where C is overwritten by the right-hand side. This setup is standard for BLAS `gemm`-routines. Here, α and β are scalars, while A, B, and C are sets of matrices stored congruently in three arrays, i.e. they are 'batched'.

To diagonalise the Hamiltonians we used `cusolverDn<t>syevjBatched`. This routine utilises the parallel cyclic-order Jacobi method, as was briefly discussed in section 4.2, to diagonalise batches of matrices simultaneously.

Lastly, solving equation (39) was done using a custom-written function, referred to as a 'kernel' in CUDA. The advantage of using the GPU for this task comes with the energy-dependence in the resolvent. We can calculate all on-shell T-matrix elements simultaneously following a single batched Hamiltonian diagonalisation.

ORCID iDs

Sean B S Miller https://orcid.org/0000-0003-4782-8326
Andreas Ekström https://orcid.org/0000-0002-4917-4293
Christian Forssén https://orcid.org/0000-0003-3458-0480

References

[1] Bedaque P F and van Kolck U 2002 Effective field theory for few-nucleon systems *Annu. Rev. Nucl. Part. Sci.* **52** 339–96
[2] Epelbaum E, Hammer H-W and Meißner U-G 2009 Modern theory of nuclear forces *Rev. Mod. Phys.* **81** 1773–825

[3] Machleidt R and Entem D R 2011 Chiral effective field theory and nuclear forces *Phys. Rep.* **503** 1–75
[4] Hammer H-W, König S and van Kolck U 2020 Nuclear effective field theory: status and perspectives *Rev. Mod. Phys.* **92** 025004
[5] Wesolowski S, Klco N, Furnstahl R J, Phillips D R and Thapaliya A 2016 Bayesian parameter estimation for effective field theories *J. Phys. G: Nucl. Part. Phys.* **43** 074001
[6] Entem D R and Machleidt R 2003 Accurate charge-dependent nucleon–nucleon potential at fourth order of chiral perturbation theory *Phys. Rev.* C **68** 041001
[7] Carlsson B D, Ekström A, Forssén C, Fahlin Strömberg D, Jansen G R, Lilja O, Lindby M, Mattsson B A and Wendt K A 2016 Uncertainty analysis and order-by-order optimization of chiral nuclear interactions *Phys. Rev.* X **6** 011019
[8] Reinert P, Krebs H and Epelbaum E 2018 Semilocal momentum-space regularized chiral two-nucleon potentials up to fifth order *Eur. Phys. J.* A **54** 86
[9] Piarulli M, Girlanda L, Schiavilla R, Navarro Pérez R, Amaro J E and Ruiz Arriola E 2015 Minimally nonlocal nucleon–nucleon potentials with chiral two-pion exchange including Δ resonances *Phys. Rev.* C **91** 024003
[10] Navarro Pérez R, Amaro J E and Ruiz Arriola E 2013 Coarse-grained potential analysis of neutron–proton and proton–proton scattering below the pion production threshold *Phys. Rev.* C **88** 064002
[11] Frame D, He R, Ipsen I, Lee D, Lee D and Rrapaj E 2018 Eigenvector continuation with subspace learning *Phys. Rev. Lett.* **121** 032501
[12] König S, Ekström A, Hebeler K, Lee D and Schwenk A 2020 Eigenvector continuation as an efficient and accurate emulator for uncertainty quantification *Phys. Lett.* B **810** 135814
[13] Furnstahl R J, Garcia A J, Millican P J and Zhang X 2020 Efficient emulators for scattering using eigenvector continuation *Phys. Lett.* B **809** 135719
[14] Melendez J A, Drischler C, Garcia A J, Furnstahl R J and Zhang X 2021 Fast & accurate emulation of two-body scattering observables without wave functions *Phys. Lett.* B **821** 136608
[15] Zhang X and Furnstahl R J 2021 Fast emulation of quantum three-body Scattering (arXiv:2110.04269 [nucl-th])
[16] Rasmussen C E and Williams C K I 2005 *Gaussian Processes for Machine Learning* (*Adaptive Computation and Machine Learning*) (Cambridge, MA: MIT Press)
[17] Haftel M I and Tabakin F 1970 Nuclear saturation and the smoothness of nucleon–nucleon potentials *Nucl. Phys.* A **158** 1–42
[18] Rubtsova O A, Kukulin V I and Pomerantsev V N 2015 Wave-packet continuum discretization for quantum scattering *Ann. Phys., NY* **360** 613–54
[19] Brynjarsdóttir J and O'Hagan A 2014 Learning about physical parameters: the importance of model discrepancy *Inverse Problems* **30** 114007
[20] Melendez J A, Wesolowski S and Furnstahl R J 2017 Bayesian truncation errors in chiral effective field theory: nucleon–nucleon observables *Phys. Rev.* C **96** 024003
[21] Hulthén L 1944 Variational problem for the continuous spectrum of a Schrödinger equation *Kungl. Fysiogr. Sällsk. i Lund Förhandl.* **14** 13
[22] Hulthén L 1948 On the Sturm–Liouville problem connected with a continuous spectrum *Arkiv Mat. Astron. Fysik* **35A** 25
[23] Kohn W 1948 Variational methods in nuclear collision problems *Phys. Rev.* **74** 1763–72
[24] Schwartz C 1966 Application of the schwinger variational principle for scattering *Phys. Rev.* **141** 1468–70
[25] Basdevant J L 1972 The Padé approximation and its physical applications *Fortschr. Phys.* **20** 283–331
[26] Erkelenz K, Alzetta R and Holinde K 1971 Momentum space calculations and helicity formalism in nuclear physics *Nucl. Phys.* A **176** 413–32
[27] du Croz J J and Higham N J 1992 Stability of methods for matrix inversion *IMA J. Numer. Anal.* **12** 1–19
[28] Glöckle W 2012 *The Quantum Mechanical Few-Body Problem* (Berlin: Springer)
[29] Carbonell J, Deltuva A, Fonseca A C and Lazauskas R 2014 Bound state techniques to solve the multiparticle scattering problem *Prog. Part. Nucl. Phys.* **74** 55–80
[30] Heller E J 1975 Theory of J-matrix Green's functions with applications to atomic polarizability and phase-shift error bounds *Phys. Rev.* A **12** 1222–31

[31] Winick J R and Reinhardt W P 1978 Moment T-matrix approach to e^+-h scattering: I. Angular distribution and total cross section for energies below the pickup threshold *Phys. Rev.* A **18** 910–24
[32] Winick J R and Reinhardt W P 1978 Moment T-matrix approach to e^+-h scattering: II. Elastic scattering and total cross section at intermediate energies *Phys. Rev.* A **18** 925–34
[33] Heller E J, Rescigno T N and Reinhardt W P 1973 Extraction of scattering information from Fredholm determinants calculated in an L^2 basis: a chebyschev discretization of the continuum *Phys. Rev.* A **8** 2946–51
[34] Corcoran C T and Langhoff P W 1977 Moment-theory approximations for nonnegative spectral densities *J. Math. Phys.* **18** 651–7
[35] Rubtsova O A and Kukulin V I 2007 Wave-packet discretization of a continuum: path toward practically solving few-body scattering problems *Phys. Atom. Nuclei* **70** 2025–45
[36] Kukulin V I, Pomerantsev V N and Rubtsova O A 2007 Wave-packet continuum discretization method for solving the three-body scattering problem *Theor. Math. Phys.* **150** 403–24
[37] Rubtsova O A, Pomerantsev V N and Kukulin V I 2009 Quantum scattering theory on the momentum lattice *Phys. Rev.* C **79** 064602
[38] Kukulin V I and Rubtsova O A 2003 Discrete quantum scattering theory *Theor. Math. Phys.* **134** 404–26
[39] Müther H, Rubtsova O A, Kukulin V I and Pomerantsev V N 2016 Discrete wave-packet representation in nuclear matter calculations *Phys. Rev.* C **94** 024328
[40] Kukulin V I and Rubtsova O A 2005 Solving the charged-particle scattering problem by wave packet continuum discretization *Theor. Math. Phys.* **145** 1711–26
[41] Rubtsova O A, Kukulin V I, Pomerantsev V N and Faessler A 2010 New approach toward a direct evaluation of the multichannel multienergy s matrix without solving the scattering equations *Phys. Rev.* C **81** 064003
[42] NVIDIA Corporation 2021 Basic linear algebra on NVIDIA GPUs https://developer.nvidia.com/cublas (accessed 22 March 2021)
[43] NVIDIA Corporation 2021 Cuda toolkit documentation https://docs.nvidia.com/cuda/cusolver/index.html (accessed 22 March 2021)
[44] Golub G H and Van Loan C F 1996 *Matrix Computations* 3rd edn (Baltimore, MD, USA: Johns Hopkins University Press)
[45] Pourzandi M and Tourancheau B 1994 A parallel performance study of Jacobi-like eigenvalue solution *Technical Report*
[46] Abdelfattah A, Tomov S and Dongarra J 2020 Matrix multiplication on batches of small matrices in half and half-complex precisions *J. Parallel Distrib. Comput.* **145** 188–201
[47] Bae S E, Shinn T-W and Takaoka T 2014 A faster parallel algorithm for matrix multiplication on a mesh array *Procedia Computer Science* **29** 2230–40 2014 International Conference on Computational Science
[48] Ekström A *et al* 2013 Optimized chiral nucleon–nucleon interaction at next-to-next-to-leading order *Phys. Rev. Lett.* **110** 192502
[49] Levinson N 1949 On the uniqueness of the potential in a schrodinger equation for a given asymptotic phase *Kgl. Danske Videnskab Selskab. Mat. Fys. Medd.* **25** 9
[50] Lisowski P W, Shamu R E, Auchampaugh G F, King N S P, Moore M S, Morgan G L and Singleton T S 1982 Search for resonance structure in thenpTotal cross section below 800 MeV *Phys. Rev. Lett.* **49** 255–9
[51] Abfalterer W P, Bateman F B, Dietrich F S, Finlay R W, Haight R C and Morgan G L 2001 Measurement of neutron total cross sections up to 560 MeV *Phys. Rev.* C **63** 044608
[52] Pomerantsev V N, Kukulin V I and Rubtsova O A 2014 New general approach in few-body scattering calculations: solving discretized Faddeev equations on a graphics processing unit *Phys. Rev.* C **89** 064008
[53] Stapp H P, Ypsilantis T J and Metropolis N 1957 Phase-shift analysis of 310 MeV proton–proton scattering experiments *Phys. Rev.* **105** 302–10
[54] Bystricky J, Lehar F and Winternitz P 1978 Formalism of nucleon–nucleon elastic scattering experiments *J. Phys. France* **39** 1–32
[55] Jia Z, Maggioni M, Smith J and Scarpazza D P 2019 Dissecting the NVidia turing T4 GPU via microbenchmarking (arXiv:1903.07486v1 [cs.DC])