


Tensor network reduced order models for wall-bounded flows

Martin Kiffner ^{1,2,*} and Dieter Jaksch^{3,1,4}¹*Clarendon Laboratory, University of Oxford, Parks Road, Oxford OX1 3PU, United Kingdom*²*PlanQC GmbH, Lichtenbergstr. 8, 85748 Garching, Germany*³*Institut für Quantenphysik, Universität Hamburg, 22761 Hamburg, Germany*⁴*The Hamburg Centre for Ultrafast Imaging, Hamburg, Germany*

(Received 6 March 2023; accepted 13 November 2023; published 8 December 2023)

We introduce a widely applicable tensor network-based framework for developing reduced order models describing wall-bounded fluid flows. As a paradigmatic example, we consider the incompressible Navier-Stokes equations and the lid-driven cavity in two spatial dimensions. We benchmark our solution against published reference data for low Reynolds numbers and find excellent agreement. In addition, we investigate the short-time dynamics of the flow at high Reynolds numbers for the lid-driven and doubly-driven cavities. We represent the velocity components by matrix product states and find that the bond dimension grows logarithmically with simulation time. The tensor network algorithm requires at most a few percent of the number of variables parametrizing the solution obtained by direct numerical simulation, and approximately improves the runtime by an order of magnitude compared to direct numerical simulation on similar hardware. Our approach is readily transferable to other flows, and paves the way towards quantum computational fluid dynamics in complex geometries.

DOI: [10.1103/PhysRevFluids.8.124101](https://doi.org/10.1103/PhysRevFluids.8.124101)

I. INTRODUCTION

Direct numerical simulation (DNS) of the Navier-Stokes equations at large Reynolds numbers would be a highly desirable capability for science and engineering applications. However, it remains an elusive goal due to the extremely large numerical complexity associated with the multiscale nature of turbulence [1,2]. The state-of-the-art method for mitigating this issue in computational fluid dynamics (CFD) is turbulence modeling [3], which continues to be under constant development for improving its accuracy.

A conceptionally different approach to reducing the numerical complexity of DNS is through structure-resolving methodologies [4,5]. These methods aim to establish a reduced order model (ROM) of the full system by exploiting correlated structures in the solution. However, identifying suitable modes for building ROMs is difficult and therefore under active investigation [5–8].

Recently, quantum-inspired tensor network methods have been introduced as a novel paradigm for modeling turbulent flows for diagnostic and predictive purposes [9]. The tensor network algorithm in [9] for solving the incompressible Navier-Stokes equation (INSE) approximates the velocity components in matrix product state (MPS) format [10]. In the examples studied in Ref. [9], the

*martin@planqc.eu

number of variables parametrizing the solution (NVPS) in MPS representation is reduced by over an order of magnitude compared to DNS. The MPS algorithm thus realizes a ROM for the investigated flows. However, the efficient compression reported in Ref. [9] only resulted in a computational speedup in a one-dimensional system, but not in two or three spatial dimensions. Furthermore, the examples in Ref. [9] are restricted to homogeneous flows with periodic boundary conditions.

Here we show that ROMs based on tensor networks can be extended to wall-bounded flows. We illustrate our approach using the lid-driven cavity in two spatial dimensions, which is a very well-studied problem [11] with tabulated reference solutions [12]. We solve the INSE in the stream-function-vorticity formulation and find that our MPS algorithm reproduces the data in Ref. [12] for stationary states at low Reynolds numbers.

As an application of our approach, we assume that the fluid is initially at rest and investigate the short-time dynamics at high Reynolds numbers. We represent the velocity components by MPSs with bond dimension χ and investigate how χ depends on time and grid size. We find that χ grows logarithmically in time and reduces the NVPS compared to direct numerical simulation by about 97%.

As an extension towards more complex flows, we also investigate the doubly driven cavity where both the top and bottom lids move and find the same qualitative behavior. We compare the run times of the MPS and DNS algorithms on similar hardware and at different Reynolds numbers. We find that the MPS algorithm can give rise to significant runtime improvements compared to DNS, peaking at a 17-fold speedup in case of the lid-driven cavity.

The MPS algorithm in Ref. [9] advances the solution to the INSE by solving an optimization problem. More specifically, the continuity equation is combined with the momentum equations via the penalty method [13], and the updated velocity components are obtained by minimizing a single cost function. On the contrary, the MPS algorithm in this work is constructed by emulating the DNS algorithm step-by-step. We achieve this by decomposing the DNS algorithm into four elementary operations (multiplication, addition, matrix-vector operations, and solving linear systems of equations) that can be realized in MPS format. It follows that our approach is directly transferable to a broad class of other CFD methodologies and flow geometries.

An important feature of quantum-inspired tensor network algorithms is that they can be ported to a quantum computer [9,14]. This transfer can be achieved with quantum circuits of known depth [15] and will provide at least a quadratic speedup over the scaling of the classical tensor network algorithm with the bond dimension [9]. Improved speedups may be achieved by problem-specific quantum circuits [15–17] that perform exponentially better than the MPS encoding of flow fields. Our work thus represents a first step towards efficient quantum algorithms for solving CFD problems with boundary conditions.

This paper is organized as follows. The model for the lid-driven cavity in the stream-function-vorticity formulation is presented in Sec. II. We give a detailed description of the model and the spatial discretization because this forms the foundation for constructing the MPS algorithm. We outline the encoding of flow fields in MPS format and describe how the DNS algorithm can be transformed into MPS format. All technical details are summarized in the Appendices. The results are shown in Sec. III and begin with a validation of our tensor network algorithm against previous work. We then consider the short-time dynamics following the quench by the moving lid and analyze the bond dimension as a function of time and grid size. A summary and discussion of our results is provided in Sec. IV.

II. MODEL

The setup for the lid-driven cavity in two spatial dimensions is shown in Fig. 1(a). We consider a square box with edge length L , and the upper lid moves with velocity u_0 in x -direction. The x component (y component) of the fluid is denoted by u (v). At $t = 0$, the fluid is at rest, $u = v = 0$. We consider a viscous fluid with kinematic viscosity ν and seek solutions to the incompressible

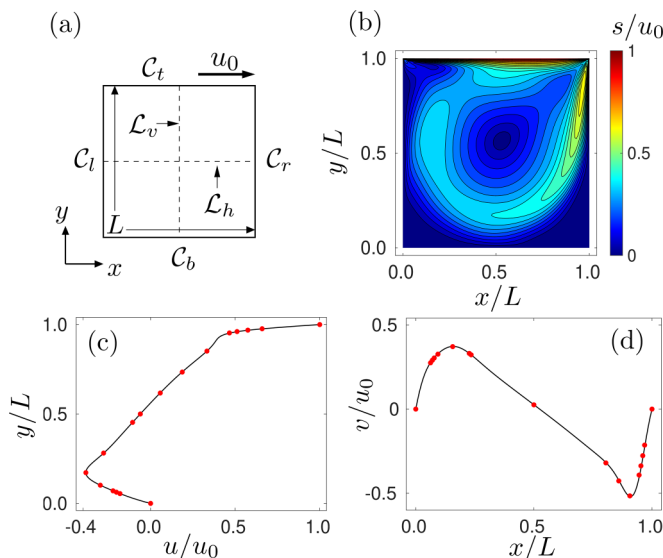


FIG. 1. (a) Setup of the square lid-driven cavity with edge length L . The upper lid moves at constant velocity u_0 in x -direction. C_t , C_r , C_b , C_l are the top, right, bottom and left boundaries, respectively. \mathcal{L}_v (\mathcal{L}_h) denotes a vertical (horizontal) line through the center of the cavity. (b) Contour plot of the velocity magnitude $s = \sqrt{u^2 + v^2}$ at $t = 50$ for $\text{Re} = 1000$ and evaluated with the tensor network algorithm. (c) Comparison of the tensor network solution for the x component u of the velocity along \mathcal{L}_v (black solid line) with the reference values in Ref. [12] (red dots). (d) Comparison of the tensor network solution for the y component v of the velocity along \mathcal{L}_h (black solid line) with the reference values in Ref. [12] (red dots).

Navier-Stokes equations in the stream-function-vorticity approach [13],

$$\partial_t w = -[\partial_x(uw) + \partial_y(vw)] + \nu \Delta w, \quad (1a)$$

$$\Delta \psi = -w. \quad (1b)$$

The stream function ψ and the velocity components u and v are connected via

$$u = \partial_y \psi, \quad (2a)$$

$$v = -\partial_x \psi, \quad (2b)$$

where

$$w = \partial_x v - \partial_y u \quad (3)$$

is the vorticity. Throughout this work we scale time in units of $t_0 = L/u_0$, length in terms of L and velocities by u_0 . Solutions to Eq. (1) are then characterized by the Reynolds number

$$\text{Re} = \frac{u_0 L}{\nu}. \quad (4)$$

We discretize the interior of the cavity (excluding boundaries) by a uniform grid with K grid points in each spatial dimension. The computational domain thus comprises K^2 equally spaced points \mathbf{r}_k with grid spacing

$$h = L/(K + 1). \quad (5)$$

Each grid point vector \mathbf{r}_k is uniquely described by a tuple of integers,

$$\mathbf{r}_k \leftrightarrow (k^x, k^y), \quad (6)$$

TABLE I. Dirichlet boundary conditions for velocity fields u , v and the stream function ψ on boundaries C_α as indicated in Fig. 1(a).

	C_l	C_r	C_b	C_t
u	u_0	0	0	0
v	0	0	0	0
ψ	0	0	0	0

where $k^\alpha \in \{0, \dots, K-1\}$ is the index of the grid point in the direction \hat{e}_α with $\alpha \in \{x, y\}$. The one-to-one correspondence in Eq. (6) allows us to label discrete function values on the grid by $F(\mathbf{r}_k) \equiv F_{k^x, k^y}$. We denote ghost points on the left (bottom) boundary by $k^x = -1$ ($k^y = -1$), and those on the right (top) boundary by $k^x = K$ ($k^y = K$).

The stream function ψ must vanish everywhere on the boundary, and all velocity components are zero except for $u = u_0$ on boundary C_l [see Fig. 1(a)]. The boundary conditions for ψ , u and v are summarized in Table I. We obtain the boundary values for the vorticity w in the standard approach [12] and find ($p, q \in \{0, \dots, K-1\}$):

$$w_{p,K} = -\frac{3}{h}u_0 + \frac{1}{h^2} \left(-4\psi_{p,K-1} + \frac{1}{2}\psi_{p,K-2} \right), \quad (7a)$$

$$w_{p,-1} = \frac{1}{h^2} \left(-4\psi_{p,0} + \frac{1}{2}\psi_{p,1} \right), \quad (7b)$$

$$w_{-1,q} = \frac{1}{h^2} \left(-4\psi_{0,q} + \frac{1}{2}\psi_{1,q} \right), \quad (7c)$$

$$w_{K,q} = \frac{1}{h^2} \left(-4\psi_{K-1,q} + \frac{1}{2}\psi_{K-2,q} \right). \quad (7d)$$

The DNS algorithm for solving Eq. (1) with the boundary conditions in Table I is outlined in Appendix A. For the time integration of Eq. (1a), we use a second-order MacCormack algorithm [13, 18, 19]. Finite-difference operations are realized by sparse matrix-vector multiplications, and we use a preconditioned conjugate gradient algorithm for solving the Poisson equation (1b). The self-consistent solution to the set of Eq. (1) is found by iteratively solving Eqs. (1b) and (1a) until convergence is achieved.

We begin the description of our MPS algorithm with a discussion of the encoding of discrete functions in MPS format. For this we assume that the number of grid points in each spatial dimension is $K = 2^N$ for an integer N . The binary representation $(\dots)_2$ of a grid point index k^α requires N bits,

$$k^\alpha = (\sigma_1^\alpha, \sigma_2^\alpha, \dots, \sigma_N^\alpha)_2, \quad (8)$$

where $\sigma_i^\alpha \in \{0, 1\}$, $\alpha \in \{x, y\}$, $i = 1, \dots, N$, and σ_1^α and σ_N^α are the most and least significant bits, respectively. We approximate a discrete function F by an MPS of bond dimension χ and length $2N$,

$$F(\mathbf{r}_k) \approx f(\mathbf{r}_k, \chi) = \underbrace{M^{\sigma_1^y} M^{\sigma_2^y} \dots M^{\sigma_N^y}}_{\text{y-encoding}} \underbrace{M^{\sigma_1^x} \dots M^{\sigma_N^x}}_{\text{x-encoding}}, \quad (9a)$$

$$= M^{\omega_1} M^{\omega_2} \dots M^{\omega_{2N}}, \quad (9b)$$

where we introduced

$$\omega_n = \begin{cases} \sigma_n^y, & 1 \leq n \leq N, \\ \sigma_{n-N}^x, & N < n \leq 2N. \end{cases} \quad (10)$$

The matrices M^{ω_n} have dimensions $d(n-1) \times d(n)$, where

$$d(n) = \min(2^n, 2^{2N-n}, \chi) \quad (11)$$

are the internal bonds that are summed over in the product of matrices in Eq. (9). These bonds are responsible for describing correlations between different length scales [9,20].

The first N matrices in Eq. (9) encode the y components of F , and the remaining N matrices account for the x components. Note that this encoding employs the scale encoding introduced in Refs. [9,20] in each spatial dimension separately. The encoding in Eq. (9) thus corresponds to expanding the function f as a sum of product functions,

$$f(\mathbf{r}_k, \chi) = \sum_{i=1}^{d(N)} \mathcal{Y}_i(k^y) \mathcal{X}_i(k^x), \quad (12)$$

where \mathcal{Y}_i (\mathcal{X}_i) is a function of the y index k^y (x index k^x) only. We find that this encoding is more efficient for the cavity geometry than the encoding in [9] where combined scales of all spatial dimensions are considered. Note that the encoding in Eq. (9) can be straightforwardly generalized to the case where each spatial dimension is discretized by a different number of grid points. This is of interest for more complex geometries than the square box considered here.

Next we describe how we emulate the DNS algorithm in tensor network format. The DNS algorithm can be broken down into the following elementary operations: (i) addition of flow fields, (ii) multiplication of flow fields, (iii) the algorithm for solving the Poisson equation, and (iv) sparse matrix-vector operations. Sparse matrix-vector operations realize finite difference operations on the flow fields, as well as the boundary conditions for the vorticity in Eq. (7).

Since all operations (i)–(iv) can be realized in MPS format, the MPS algorithm for solving Eq. (1) can be obtained by replacing each elementary operation in the DNS algorithm by its MPS counterpart. MPSs can be added [10] and multiplied [21], and a Poisson solver in MPS format has been reported in Ref. [22]. Matrix-vector operations are realized by contracting a matrix product operator (MPO) with an MPS [10], and all MPOs for realizing the finite difference operations and boundary conditions are provided in Appendix C. The numerical complexity of all these operations scales polynomially with the bond dimension χ of the MPS [for details see Appendix B]. It follows that the MPS realizations of operations (i)–(iv) can be numerically more efficient than their standard implementations for sufficiently small χ .

All variables (velocity components u and v , stream function ψ and vorticity w) are approximated by an MPS with bond dimension χ . We allow χ to dynamically grow in order to keep the numerical complexity of our algorithm minimal. We achieve this by normalizing the MPSs representing ψ and w to unity, and by inspecting the singular values near the center of these MPSs. We increase the bond dimension if the smallest singular value exceeds a threshold ϵ , which we set to $\epsilon = 5 \times 10^{-8}$ throughout this work. This choice has been informed by numerical tests, ensuring that all precision targets of the algorithms implementing the elementary operations are met with the smallest possible χ .

III. RESULTS

In a first step we validate the MPS algorithm against the tabulated results for the stationary state of the lid-driven cavity in Ref. [12]. We consider a $2^7 \times 2^7$ grid ($N = 7$) and Reynolds number $\text{Re}=1000$. The contours showing the velocity magnitude according to the MPS algorithm and for $t/t_0 = 50$ are shown in Fig. 1(b). We compare this to the data in Ref. [12] in Figs. 1(b) and 1(c). The velocity component u along the vertical line \mathcal{L}_v [see Fig. 1(a)] according to the the MPS algorithm (black solid lines) agrees very well with the data in Ref. [12] (red dots). Similarly, we find that our MPS results for v along the horizontal line \mathcal{L}_h agree very well with Ref. [12] as shown in Fig. 1(d). We note that our DNS algorithm is in excellent agreement with the MPS algorithm and with the reference data in Ref. [12].

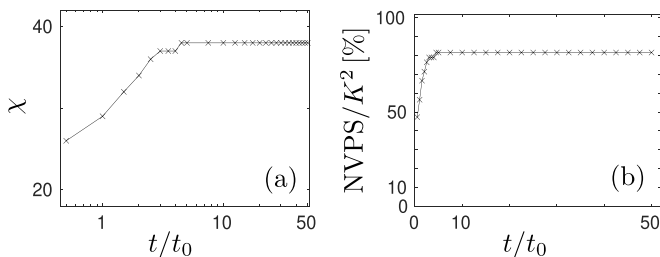


FIG. 2. (a) Bond dimension χ versus time on a logarithmic scale and for the flow in Fig. 1(b). (b) The ratio between the NVPS and the total number of grid points K^2 in percent and as a function of time. Solid lines are a guide to the eye.

The MPS algorithm dynamically adapts the bond dimension of the MPS representing the flow fields. Initially the fluid is at rest, $u = v = 0$. This constant velocity field is an MPS with bond dimension $\chi = 1$. However, we find through numerical experiments that the sudden quench induced by the moving lid requires a starting bond dimension of $\chi = 26$. The subsequent evolution of χ with time for the flow in Fig. 1(b) is shown in Fig. 2(a). We find that χ approximately grows logarithmically with time until $t/t_0 \approx 3$, and then it stays constant at $\chi = 38$. At $t/t_0 \approx 3$, the vortex created by the moving lid has expanded from the top right corner to the whole size of the cavity. While the vortex changes shape until the steady state is reached, the bond dimension stays constant in this regime.

The bond dimension χ is directly related to the NVPS, which is shown in Fig. 2(b) in relation to the total number of grid points K^2 . Initially the NVPS are about 47% of K^2 . For larger times, the NVPS slowly increases to 82% of K^2 . It follows that the MPS format does not result in an efficient compression of the stationary state for $\text{Re} = 1000$.

The situation is completely different in a transient regime at high Reynolds numbers. For this we consider a flow with $\text{Re} = 24\,000$, and Fig. 3(a) shows the corresponding contours of the velocity

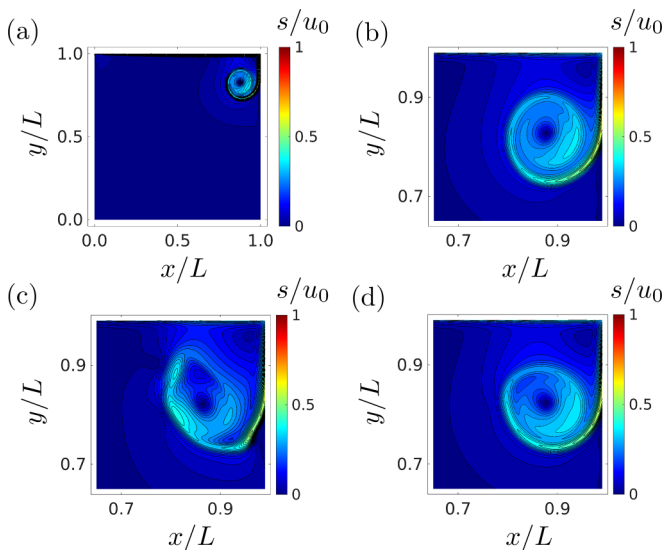


FIG. 3. (a) Contour plot of the velocity magnitude $s = \sqrt{u^2 + v^2}$ at $t/t_0 = 3$ for the flow configuration shown in Fig. 1(a). The grid size is $K^2 = 2^{11} \times 2^{11}$, $\text{Re} = 24\,000$ and results are obtained with the MPS algorithm. (b) Same as in (a) but focusing on the region where the initial vortex forms. (c) Same as in (b) but for $K^2 = 2^9 \times 2^9$. (d) Same as in (b) but for $K^2 = 2^{10} \times 2^{10}$.

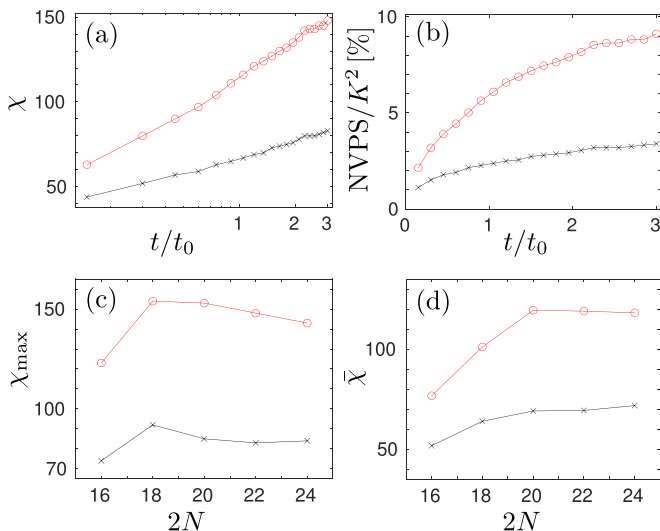


FIG. 4. Analysis of the bond dimension as a function of time and grid size. Black crosses [red circles] correspond to the flow in Fig. 3(a) [Fig. 5], and solid lines are a guide to the eye. (a) Bond dimension χ versus time on a logarithmic scale. (b) The ratio between the NVPS and the total number of grid points K^2 in percent and as a function of time. (c) Bond dimension χ_{\max} at $t/t_0 = 3$ as a function of grid size. (d) Temporally averaged bond dimension $\bar{\chi}$ as a function of grid size.

magnitude on a $2^{11} \times 2^{11}$ grid at $t/t_0 = 3$. A magnified view of the vortex forming in the top right corner is shown in Fig. 3(b). It is well known that the lid-driven cavity only exhibits a truly stationary state for $\text{Re} \leq 10\,000$ [23]. For larger Reynolds numbers, the system becomes chaotic and develops random fluctuations that persist for all times. However, we find that at the short times considered here where turbulence has not formed yet, the system is still deterministic. All runs with the same initial conditions give the same result. We expect the onset of turbulence and nonstationary fluctuations at much later times when the vortex has spread to the whole cavity.

Next we investigate the required grid size to correctly represent this transient flow. For this we run the calculation for different grid sizes $2^N \times 2^N$ with $N = 8, 9, 10, 11$, and 12. The results for $N = 9$ and $N = 10$ are shown in Figs. 3(c) and 3(d), respectively. By comparing it to the solution for $N = 11$ in Fig. 3(b), we find that the flow fields are underresolved on the $N = 9, 10$ grids. For $N = 9$ [see Fig. 3(c)], the vortex in the upper right corner is strongly deformed. The amount of deformation is much smaller but still visible for the $N = 10$ grid [see Fig. 3(d)]. On the other hand, increasing the size to $N = 12$ (not shown) does not result in any significant changes compared with the results for $N = 11$. We thus conclude that the $2^{11} \times 2^{11}$ grid is sufficiently large for representing this flow.

The smallest length scale in a fully developed turbulent flow is the Kolmogorov microscale $\eta/L \approx \text{Re}^{-3/4}$ [1,2]. Although the flow investigated in Fig. 3 is not in the turbulent regime yet, the value of $\eta/L \approx 5.19 \times 10^{-4}$ for $\text{Re} = 24\,000$ is consistent with the grid point spacing $2^{-11} \approx 4.88 \times 10^{-4}$ for the $2^{11} \times 2^{11}$ grid that resolves this flow. We conclude that the smallest scale according to Kolmogorov theory is excited even in the investigated regime where the flow is still laminar. It follows that η/L gives a reasonable estimate for the required grid size.

The variation of χ with time and for the flow in Fig. 3(a) is shown by the black crosses in Fig. 4(a). At $t = 0$ we set $\chi = 40$, and after a short initial phase [not shown in Fig. 4(a)] we find that χ grows logarithmically in time. The corresponding NVPS in relation to the total number of grid points is shown by the black crosses in Fig. 4(b). At very short times, the NVPS are only about

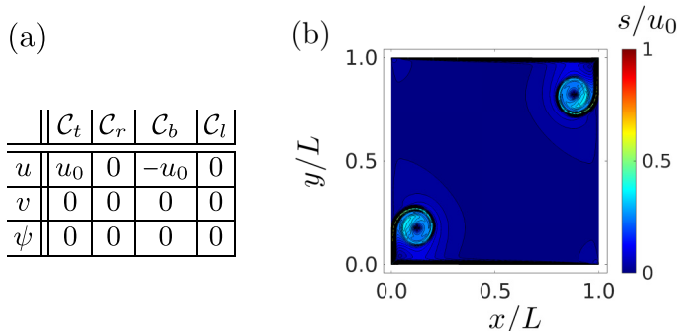


FIG. 5. (a) Boundary conditions corresponding to the doubly driven cavity where the upper [bottom] lid moves at constant velocity u_0 [$-u_0$] in x direction. (b) Contour plot of the velocity magnitude $s = \sqrt{u^2 + v^2}$ at $t/t_0 = 3$ for the doubly driven cavity on a $K^2 = 2^{11} \times 2^{11}$ grid with $\text{Re} = 24\,000$ and evaluated with the MPS algorithm.

1% of K^2 , and thus the MPS format achieves a compression of 99%. For larger times, the NVPS slowly increases to 3.4% of K^2 , corresponding to a compression of 96.6%.

Next we investigate the dependence of the bond dimension on the grid size. We find that for all studied grids ($N = 8, 9, 10, 11, 12$), χ vs time has the same qualitative behavior as shown in Fig. 4(a) for $N = 11$. For each of these curves, we calculate the maximal value χ_{\max} at $t/t_0 = 3$ and the temporally averaged bond dimension $\bar{\chi}$. The results for χ_{\max} and $\bar{\chi}$ are shown by black crosses in Figs. 4(a) and 4(b), respectively. We find that χ_{\max} and $\bar{\chi}$ vary with $2N$ until the grid is fine enough to represent the flow. While $\bar{\chi}$ increases steadily with $2N$, χ_{\max} first increases then decreases with $2N$.

We now investigate how the results for the bond dimension in the lid-driven cavity geometry change if we consider a doubly driven cavity instead, see Fig. 5. The upper lid continues to move at constant velocity u_0 in the x direction. In addition, the bottom lid moves at constant velocity $-u_0$ in the x direction. The corresponding contours of the velocity magnitude on a $2^{11} \times 2^{11}$ grid and with $\text{Re} = 24\,000$ are shown in Fig. 5(b). We find that a second vortex forms in the bottom left corner of the cavity. The corresponding results for the bond dimension as a function of time and grid size are shown by the red circles in Fig. 4. The qualitative behavior of all curves is similar to the lid-driven cavity, but the bond dimension for the doubly driven cavity is larger than for the lid-driven cavity at each point in time. The NVPS in relation to the total number of grid points grows to about 9% for the doubly driven cavity, and hence the MPS format still achieves a compression of more than 90%.

The results in Fig. 4 show that the bond dimension only grows logarithmically with simulation time, and that the MPS format achieves an efficient compression of the flow fields. The numerical complexity of the MPS algorithm depends on the bond dimension χ as detailed in Appendix B. While the most costly operation is the multiplication of two MPSs, the algorithm spends the most time on solving the Poisson equation which scales as $2N\chi^3$ [22]. On the other hand, the DNS algorithm can be broken down into sparse matrix-vector multiplications scaling with the total number of grid points $K^2 = 2^{2N}$. The exponentially worse scaling of the DNS algorithm with respect to the number of grid points K^2 suggests that the MPS algorithm can give rise to a computational advantage for sufficiently small values of χ .

To address this question we compare the runtimes of the MPS and DNS algorithms. In order to achieve a fair comparison, we implemented the DNS and MPS algorithms in the same programming language (i.e., Matlab [24]), and evaluated all runs on a single node of the ARC facility (Intel Xeon Platinum 8268 CPU @ 2.90GHz) [25]. Furthermore, we ensure that the DNS and MPS algorithms solve Eq. (1) with the same accuracy (see Appendix B). We find that the MPS algorithm is 5.8 times faster than the DNS algorithm in the case of the lid-driven cavity. The speedup reduces to 3.3 for the

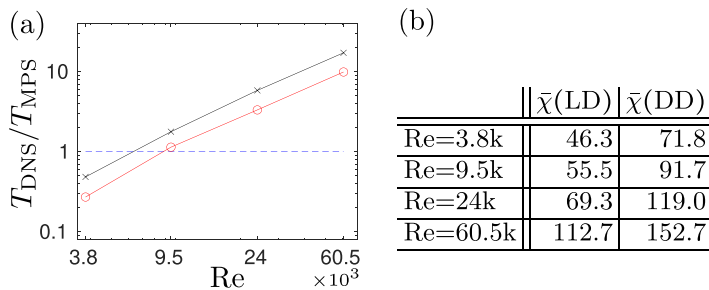


FIG. 6. (a) Ratio of the average times T_{MPS} for completing a single iteration of the MPS algorithm and T_{DNS} for completing a single iteration of the DNS algorithm as a function of Reynolds number Re . Black crosses [red circles] correspond to the lid-driven [doubly driven] cavity. Averages are taken up to $t/t_0 = 3$. T_{DNS} for $\text{Re} = 24\text{k}$ ($\text{Re} = 60.5\text{k}$) is only taken for $t/t_0 \leq 1$ ($t/t_0 \leq 0.1$) due to the large run times, and T_{MPS} for the doubly driven cavity and $\text{Re} = 60.5\text{k}$ is evaluated for $t/t_0 = 2.1$. The grid spacing for each Re is chosen such that it matches the corresponding microscale $\eta/L \approx \text{Re}^{-3/4}$. For data points above (below) the horizontal blue dashed line, the MPS (DNS) algorithm runs faster than its DNS (MPS) counterpart. Solid lines are a guide to the eye. (b) Time-averaged bond dimensions $\bar{\chi}(\text{LD})$ [$\bar{\chi}(\text{DD})$] corresponding to the lid-driven [doubly driven] cavity for different Reynolds numbers.

doubly-driven cavity since the bond dimension is larger than for the lid-driven cavity at each time step, see Fig. 4(a).

A more comprehensive runtime comparison of the MPS and DNS algorithms at different Reynolds numbers is presented in Fig. 6(a). The grid spacing for each Re is chosen such that it matches the corresponding microscale $\eta/L \approx \text{Re}^{-3/4}$. We show the ratio of the average times T_{DNS} for completing a single iteration of the DNS algorithm and T_{MPS} for completing a single iteration of the MPS algorithm. Since the MPS and DNS algorithms approximately require the same number of iterations, this ratio is also representative of the overall runtime ratio. The MPS algorithm for the lid-driven and doubly driven cavities runs faster than the DNS algorithm for $\text{Re} \geq 9.5 \times 10^3$. For a given Reynolds number, the MPS algorithm for the doubly driven cavity takes more time than in the case of the lid-driven cavity because the former requires larger bond dimensions, see Fig. 5(b). For the largest Reynolds number, the MPS algorithm approximately achieves a 17-fold [ten-fold] speedup compared with the DNS algorithm for the lid-driven [doubly driven] cavity.

The speedups shown in Fig. 6 can be qualitatively explained by noting that the DNS algorithm scales like $\text{Re}^{6/4}$, whereas the MPS algorithm scales as $\log \text{Re}$ for fixed bond dimension. However, the bond dimension grows with time and with Reynolds number, and therefore a general scaling of the runtime ratio with Reynolds number is difficult to obtain. At larger simulation times, the runtime advantage of the MPS algorithm may decrease or vanish if the required bond dimension becomes too large. The results in Fig. 6 nevertheless illustrate the tremendous potential of MPS for simulating transient flows.

IV. SUMMARY AND DISCUSSION

We have shown that dynamical solutions to the incompressible Navier-Stokes equations for the lid-driven and doubly driven cavities can be obtained via a tensor network algorithm. Our work extends the results in Ref. [9] by showing that the tensor network approach is not restricted to periodic boundary conditions but works equally well for problems with fixed boundary conditions. We achieve this by decomposing a DNS algorithm based on MacCormack's method [13, 18, 19] into four elementary operations of addition, multiplication, matrix-vector multiplication, and solving the Poisson equation. These four operations can be implemented in MPS format and the resulting MPS algorithm automatically builds a ROM characterized by a bond dimension χ . Note that this

ROM becomes exact with sufficiently large bond dimension, which distinguishes it from data-driven ROMs [6–8] for CFD which lack this guarantee of success.

It is important to note that our approach also applies to other CFD methodologies and flow geometries. For example, the stream-function-vorticity formulation chosen in this work can be replaced with continuity and momentum equations expressed in terms of velocities and pressure [26]. Rewriting this algorithm in terms of tensor network operations follows the same route as presented here.

We run the MPS algorithm on a uniform grid and find that it automatically allocates resources only to those regions in space where they are needed. No *a priori* knowledge of the flow is required. For example, the NVPS required by MPS to describe the transient regime at large Reynolds number is at most 3% of the total number of gridpoints. This very efficient MPS representation of the flow occurs because the vortex only occupies a small region in space. Very little resources are needed to represent the flow in the large area where the fluid is nearly at rest, see Fig. 3(a). Adding the second vortex in the case of the doubly driven cavity increases the NVPS to 9%.

A related finding is that the bond dimension of the MPSs representing the flow fields is approximately constant if the grid is fine enough to represent the flow. This feature is related to the known fact that polynomials and Fourier series have efficient MPS representations where the bond dimension is independent of the grid size [27,28]. This behavior is also akin to one-dimensional quantum systems obeying an area law [29].

We find that the bond dimension of the MPSs representing the transient flows investigated in this work grows logarithmically in time. This slow increase can translate into a runtime advantage of the MPS vs DNS algorithms if the bond dimension of the initial flow fields is sufficiently small. We find that the MPS algorithm can be significantly faster than the DNS algorithm for simulation times of several units of $t_0 = L/u_0$, i.e., the time it takes the lid to traverse the length L of the cavity. In general, our analysis shows that the MPS algorithm will outperform the DNS algorithm at large Reynolds numbers, provided that the required bond dimension is sufficiently small. We anticipate that the maximal bond dimension allowing for a speedup depends on the used hardware and software implementation of the algorithm, which is subject for further study.

Several avenues for further research emerge from here. First, the transient flow example studied in this work may also be efficiently described with adaptive mesh refinement [30,31]. In this approach, the grid spacing is dynamically varied in space at the cost of detecting the areas requiring high-resolution grids. It would be interesting to directly compare the performance of these two methods for different flow types, and to establish the differences and similarities between them.

Second, MPS algorithms for CFD may benefit from modern hardware architectures optimized for tensor operations [32]. This opens up the exciting prospect of developing tensor network algorithms for technical flows that outperform state-of-the-art CFD algorithms.

Finally, CFD algorithms in tensor network format represent a first step towards solving the Navier-Stokes equations on a quantum computer [15,33,34]. Quantum CFD [9,14] promises to enable DNS for analyzing and optimizing technical flows, which would represent a revolutionary improvement of the state-of-the-art [33]. Creating and benchmarking quantum CFD algorithms for wall-bounded flows by porting tensor network algorithms to quantum hardware is thus an exciting prospect for future research.

ACKNOWLEDGMENTS

M.K. acknowledges support by EPSRC Programme Grant DesOEQ (EP/P009565/1) and thanks D. Peshkin, P. Moinier, H. Babae, T. Rung, S. Bengoechea, P. Over, and L. Anderson for discussions. D.J. acknowledges support by the European Union’s Horizon Programme (HORIZON-CL4-2021-DIGITALEMERGING-02-10) Grant Agreement No. 101080085 QCFD, by EPSRC Programme Grant No. DesOEQ (EP/P009565/1), from AFOSR Grant No. FA8655-22-1-7027 and by the Cluster of Excellence ‘CUI: Advanced Imaging of Matter’ of the Deutsche Forschungsgemeinschaft (DFG) - EXC 2056 - project ID 390715994.

APPENDIX A: DNS ALGORITHM

The DNS algorithm for solving Eq. (1) can be broken down into four steps for advancing the solutions for w , ψ , u and v from time t to $t + \Delta t$. In the following we describe each of these steps:

(i) Starting with the stream function ψ^t at time t , we calculate the velocity components u and v according to Eq. (2). For this we employ a second-order accurate central difference approximation of the first derivatives in the x and y direction,

$$[\partial_x \psi]_{p,q} = \frac{1}{2h}(\psi_{p+1,q} - \psi_{p-1,q}), \quad (\text{A1a})$$

$$[\partial_y \psi]_{p,q} = \frac{1}{2h}(\psi_{p,q+1} - \psi_{p,q-1}). \quad (\text{A1b})$$

(ii) The vorticity is propagated in time by an explicit, second-order accurate MacCormack scheme [13,18,19]. To this end we write Eq. (1a) as

$$\partial_t w = \partial_x F + \partial_y G, \quad (\text{A2})$$

where

$$F = -uw + v(\partial_x w), \quad (\text{A3a})$$

$$G = -vw + v(\partial_y w). \quad (\text{A3b})$$

MacCormack's algorithm advances w^t to $w^{t+\Delta t}$ in a two-step predictor-corrector procedure:

(1) Predictor step:

In order to evaluate F and G , the derivatives $\partial_x w$ and $\partial_y w$ in Eq. (A3) are approximated by first-order accurate backward differences δ_x^{bwd} and δ_y^{bwd} , respectively,

$$[\delta_x^{\text{bwd}} w]_{p,q} = \frac{w_{p,q} - w_{p-1,q}}{h}, \quad (\text{A4a})$$

$$[\delta_y^{\text{bwd}} w]_{p,q} = \frac{w_{p,q} - w_{p,q-1}}{h}. \quad (\text{A4b})$$

The predicted solution $\bar{w}^{t+\Delta t}$ (indicated by an overbar) is obtained by a first-order accurate forward discretization of the spatial derivatives in Eq. (A2),

$$\bar{w}_{p,q}^{t+\Delta t} = w_{p,q}^t + \left(\frac{F_{p+1,q}^t - F_{p,q}^t}{h} + \frac{G_{p,q+1}^t - G_{p,q}^t}{h} \right) \Delta t. \quad (\text{A5})$$

Evaluating Eq. (A4) on the inner grid with $p, q \in \{0, \dots, K-1\}$ requires the boundary values of w for $w_{-1,q}$ and $w_{p,-1}$ in Eq. (7). In addition, Eq. (A5) for $p = K-1$ requires $[\delta_x^{\text{bwd}} w]_{K,q}$, and for $q = K-1$ we need $[\delta_y^{\text{bwd}} w]_{p,K}$. These values can be obtained with the help of the boundary values $w_{K,q}$ and $w_{p,K}$, respectively.

(2) Corrector step:

The derivatives $\partial_x w$ and $\partial_y w$ in Eq. (A3) are now approximated by first-order accurate forward differences δ_x^{fwd} and δ_y^{fwd} , respectively,

$$[\delta_x^{\text{fwd}} w]_{p,q} = \frac{w_{p+1,q} - w_{p,q}}{h}, \quad (\text{A6a})$$

$$[\delta_y^{\text{fwd}} w]_{p,q} = \frac{w_{p,q+1} - w_{p,q}}{h}. \quad (\text{A6b})$$

We update the functions F and G with the predicted solution $\bar{w}^{t+\Delta t}$ and obtain $\bar{F}^{t+\Delta t}$ and $\bar{G}^{t+\Delta t}$. The solution for the vorticity $w^{t+\Delta t}$ at $t + \Delta t$ is then obtained by approximating the spatial

TABLE II. Overview of the algorithms for realizing the building blocks of the DNS algorithm in MPS format. The last column indicates the scaling of the operation with the bond dimension of the MPSs and MPOs.

Operation	Algorithm	Scaling
Addition	Variational addition of MPS (see Sec. 4.5 in Ref. [10]).	χ^3
Multiplication	Multiplication algorithm in Ref. [21] combined with variational compression [10] of the product MPS.	χ^4
Poisson solver	MPS algorithm for solving the Poisson equation in Ref. [22].	χ^3
Matrix-vector multiplication	MPO-MPS contraction combined with variational compression (see Sec. 5 in Ref. [10]). For the system considered here, the MPO bond dimension $D \leq 6$ and thus $D \ll \chi$.	$D\chi^3$

derivatives in Eq. (1a) by first-order accurate backward differences,

$$\begin{aligned}
 w_{p,q}^{t+\Delta t} &= \frac{1}{2} (w_{p,q}^t + \bar{w}_{p,q}^{t+\Delta t}) \\
 &+ \frac{1}{2} \left(\frac{\bar{F}_{p,q}^{t+\Delta t} - \bar{F}_{p-1,q}^{t+\Delta t}}{h} + \frac{\bar{G}_{p,q}^{t+\Delta t} - \bar{G}_{p,q-1}^{t+\Delta t}}{h} \right) \Delta t.
 \end{aligned} \tag{A7}$$

With the help of the boundary values for w in Eq. (7), Eq. (A7) can be evaluated on every point of the inner grid with $p, q \in \{0, \dots, K-1\}$. Although the forward- and backward differences in Eqs. (A5)–(A7) are only first-order accurate in h , the resulting expression for $w_{p,q}^{t+\Delta t}$ in Eq. (A7) is second-order accurate [13,18,19].

(iii) The vorticity $w^{t+\Delta t}$ is used to find the stream function $\psi^{t+\Delta t}$ at time $t + \Delta t$ by solving Eq. (1b) with the boundary conditions in Table I and a second-order accurate discretization of the Laplace operator,

$$[\Delta\psi]_{p,q} = \frac{\psi_{p+1,q} + \psi_{p-1,q} + \psi_{p,q+1} + \psi_{p,q-1} - 4\psi_{p,q}}{h^2}. \tag{A8}$$

(iv) The set of equations (1) are coupled because the updated stream function $\psi^{t+\Delta t}$ gives rise to new velocity components $u^{t+\Delta t}$ and $v^{t+\Delta t}$ via Eq. (2). We repeat steps (i)–(iii) until a self-consistent solution to Eq. (1) has been found. This results in updated functions $\psi^{t+\Delta t}$, $w^{t+\Delta t}$, $u^{t+\Delta t}$ and $v^{t+\Delta t}$ and completes the time step from t to $t + \Delta t$. We repeat steps (i)–(iv) until the final time is reached.

APPENDIX B: MPS ALGORITHMS

Table II outlines the MPS algorithms for realizing the required elementary operations as well as their scaling with the bond dimension χ . All these algorithms have in common that they are variational in nature. The desired MPS for representing the target, i.e., the sum or product of MPSs or the solution to the Poisson equation, is found by minimizing a cost function. These cost functions are quadratic in the variables and hence efficient and reliable methods for finding that optimal solutions exist. We employ single-site density matrix renormalization group (DMRG)-like [10] sweeps where each tensor in the MPS is sequentially optimized until overall convergence has been achieved.

In order to make the results of the MPS algorithm comparable to the DNS results, we impose the same accuracy goal for solving the Poisson equation and the same convergence criterion for solving Eq. (1) in both algorithms.

APPENDIX C: MPOs FOR FINITE DIFFERENCE OPERATIONS

Here we show how the required finite difference operations can be created in the MPO-MPS formalism. We denote an MPO by Q and its contraction with an MPS f as Qf . A generic MPO with bond dimension D can be written as [35]

$$Q = AB[1] \cdots B[N]B[N+1] \cdots B[2N]C, \quad (\text{C1})$$

where A is a $1 \times D$ row vector, C is a $D \times 1$ column vector, and $B[k]$ with $k \in \{1, \dots, 2N\}$ are $D \times D$ matrices whose matrix elements are 2×2 matrices. Any 2×2 matrix can be expanded in terms of the following four operators:

$$\sigma_{01} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix}, \quad (\text{C2a})$$

$$\sigma_{10} = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix}, \quad (\text{C2b})$$

$$\sigma_{00} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}, \quad (\text{C2c})$$

$$\sigma_{11} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}. \quad (\text{C2d})$$

For convenience, we also introduce the identity matrix

$$\mathbb{1} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \quad (\text{C3})$$

When multiplying the matrices $B[k]$ in Eq. (C1), we take the outer product of the matrix-valued matrix elements. In order to illustrate this notation, we consider the following example for $N = 1$:

$$A = (1, 0), \quad (\text{C4})$$

$$B[k] = \begin{pmatrix} \mathbb{1} & \sigma_{01} \\ \langle /p \rangle \langle p \rangle & 0 \quad \sigma_{10} \end{pmatrix}, \quad 1 \leq k \leq 2 \quad (\text{C5})$$

$$C = (1, 1)^t. \quad (\text{C6})$$

The corresponding MPO is

$$Q = (1, 0) \begin{pmatrix} \mathbb{1} \otimes \mathbb{1} & \mathbb{1} \otimes \sigma_{01} + \sigma_{01} \otimes \sigma_{10} \\ 0 & \sigma_{10} \otimes \sigma_{10} \end{pmatrix} \begin{pmatrix} 1 \\ 1 \end{pmatrix} \quad (\text{C7})$$

$$= \mathbb{1} \otimes \mathbb{1} + \mathbb{1} \otimes \sigma_{01} + \sigma_{01} \otimes \sigma_{10}, \quad (\text{C8})$$

where \otimes denotes the outer product.

The MPO representation of the first-order accurate forward-backward differences are described in Appendix C 1, and Appendices C 2 and C 3 provide the MPOs for the Laplace operator and the central differences, respectively.

1. Forward-backward differences

We provide generic expressions for the MPOs facilitating forward- and backward differences in Appendix C 1 a. These expressions are valid if the boundary values of the function to be differentiated are zero everywhere. Specific expressions are required for functions with nonzero boundary values. In the algorithm described in Appendix A, boundary values are required for calculating finite-difference approximations of the first and second spatial derivatives of w . These expressions are given in Appendices C 1 b and C 1 c for the predictor and corrector steps, respectively.

a. Generic expressions

(1) Forward-differencing in the x direction:

$$[Q_x^{\text{fwd}} f]_{p,q} = \frac{f_{p+1,q} - f_{p,q}}{h}, \quad (\text{C9a})$$

with

$$A = (1, 0)/h, \quad (\text{C9b})$$

$$B[k] = \mathbb{1}, \quad 1 \leq k \leq N, \quad (\text{C9c})$$

$$B[k] = \begin{pmatrix} \mathbb{1} & \sigma_{01} \\ 0 & \sigma_{10} \end{pmatrix}, \quad N < k \leq 2N. \quad (\text{C9d})$$

$$C = (-1, 1)^t. \quad (\text{C9e})$$

(2) Backward differencing in the x direction:

$$[Q_x^{\text{bwd}} f]_{p,q} = \frac{f_{p,q} - f_{p-1,q}}{h}, \quad (\text{C10a})$$

with

$$A = (1, 0)/h, \quad (\text{C10b})$$

$$B[k] = \mathbb{1}, \quad 1 \leq k \leq N, \quad (\text{C10c})$$

$$B[k] = \begin{pmatrix} \mathbb{1} & \sigma_{10} \\ 0 & \sigma_{01} \end{pmatrix}, \quad N < k \leq 2N, \quad (\text{C10d})$$

$$C = (-1, 1)^t. \quad (\text{C10e})$$

(3) Forward differencing in the y direction:

$$[Q_y^{\text{fwd}} f]_{p,q} = \frac{f_{p,q+1} - f_{p,q}}{h}, \quad (\text{C11a})$$

with

$$A = (1, 0)/h, \quad (\text{C11b})$$

$$B[k] = \begin{pmatrix} \mathbb{1} & \sigma_{01} \\ 0 & \sigma_{10} \end{pmatrix}, \quad 1 \leq k \leq N, \quad (\text{C11c})$$

$$B[k] = \mathbb{1}, \quad N < k \leq 2N. \quad (\text{C11d})$$

$$C = (-1, 1)^t. \quad (\text{C11e})$$

(4) Backward differencing in the y direction:

$$[Q_y^{\text{bwd}} f]_{p,q} = \frac{f_{p,q} - f_{p,q-1}}{h}, \quad (\text{C12a})$$

with

$$A = (1, 0)/h, \quad (\text{C12b})$$

$$B[k] = \begin{pmatrix} \mathbb{1} & \sigma_{10} \\ 0 & \sigma_{01} \end{pmatrix}, \quad 1 \leq k \leq N, \quad (\text{C12c})$$

$$B[k] = \mathbb{1}, \quad N < k \leq 2N, \quad (\text{C12d})$$

$$C = (-1, 1)^t. \quad (\text{C12e})$$

b. Finite differences of w - predictor step

Here we provide the MPOs required for evaluating the predictor step in Eq. (A5).

(1) Backward difference of w in the x direction:

$$\delta_x^{\text{bwd}} w \approx Q_x^{\text{bwd}} f_w - \frac{1}{h} Q_{C_l} f_\psi = f_{w_x}^{\text{bwd}}, \quad (\text{C13})$$

where Q_x^{bwd} is given in Eq. (C10), and f_w and f_ψ are the MPSs representing w and ψ , respectively. The MPO Q_{C_l} creates the boundary values for w at C_l ,

$$[Q_{C_l} f]_{p,q} = \frac{1}{h^2} \left(-4f_{0,q} + \frac{1}{2}f_{1,q} \right) \delta_{p,0}, \quad (\text{C14a})$$

with

$$A = 1/h^2, \quad (\text{C14b})$$

$$B[k] = \mathbb{1}, \quad 1 \leq k \leq N, \quad (\text{C14c})$$

$$B[k] = \sigma_{00}, \quad N < k \leq 2N - 1, \quad (\text{C14d})$$

$$B[2N] = -4\sigma_{00} + \sigma_{01}/2, \quad (\text{C14e})$$

$$C = 1. \quad (\text{C14f})$$

In Eq. (C13), $f_{w_x}^{\text{bwd}}$ is the MPS representing $\delta_x^{\text{bwd}} w$.

(2) Forward-backward difference of w in the x direction:

$$\delta_x^{\text{fwd}} (\delta_x^{\text{bwd}} w) \approx Q_x^{\text{fwd}} f_{w_x}^{\text{bwd}} + \frac{1}{h} \left[\frac{1}{h} (Q_{C_r} f_\psi - Q_r f_w) \right], \quad (\text{C15})$$

where Q_{C_r} is defined as

$$[Q_{C_r} f]_{p,q} = \frac{1}{h^2} \left(-4f_{K-1,q} + \frac{1}{2}f_{K-2,q} \right) \delta_{p,0}, \quad (\text{C16a})$$

with

$$A = 1/h^2, \quad (\text{C16b})$$

$$B[k] = \mathbb{1}, \quad 1 \leq k \leq N, \quad (\text{C16c})$$

$$B[k] = \sigma_{11}, \quad N < k \leq 2N - 1, \quad (\text{C16d})$$

$$B[2N] = -4\sigma_{11} + \sigma_{10}/2, \quad (\text{C16e})$$

$$C = 1. \quad (\text{C16f})$$

The MPO Q_r in Eq. (C15) extracts the values of a function on the line $k^x = K - 1$,

$$[Q_r f]_{p,q} = f_{p,q} \delta_{K-1,q}, \quad (\text{C17})$$

with

$$A = 1, \quad (\text{C18a})$$

$$B[k] = \mathbb{1}, \quad 1 \leq k \leq N, \quad (\text{C18b})$$

$$B[k] = \sigma_{11}, \quad N < k \leq 2N, \quad (\text{C18c})$$

$$C = 1. \quad (\text{C18d})$$

(3) Backward difference of w in the y direction:

$$\delta_y^{\text{bwd}} w \approx Q_y^{\text{bwd}} f_w - \frac{1}{h} Q_{C_b} f_\psi = f_{w_y}^{\text{bwd}}, \quad (\text{C19})$$

where Q_y^{bwd} is given in Eq. (C12), and f_w and f_ψ are the MPSs representing w and ψ , respectively. The MPO Q_{C_b} creates the boundary values for w at C_b and is defined as

$$[Q_{C_b}f]_{p,q} = \frac{1}{h^2} \left(-4f_{p,0} + \frac{1}{2}f_{p,1} \right) \delta_{q,0}, \quad (\text{C20a})$$

with

$$A = 1/h^2, \quad (\text{C20b})$$

$$B[k] = \sigma_{00}, \quad 1 \leq k \leq N-1, \quad (\text{C20c})$$

$$B[N] = -4\sigma_{00} + \sigma_{01}/2, \quad (\text{C20d})$$

$$B[k] = \mathbb{1}, \quad N < k \leq 2N, \quad (\text{C20e})$$

$$C = 1. \quad (\text{C20f})$$

In Eq. (C19), $f_{w_y}^{\text{bwd}}$ is the MPS representing $\delta_y^{\text{bwd}}w$.

(4) Forward-backward difference of w in the y direction:

$$\delta_y^{\text{fwd}}(\delta_y^{\text{bwd}}w) \approx Q_y^{\text{fwd}}f_{w_y}^{\text{bwd}} + \frac{1}{h} \left[\frac{1}{h} (Q_{C_t}f_\psi + f_{u_0} - Q_t f_w) \right], \quad (\text{C21})$$

where Q_{C_t} is defined as

$$[Q_{C_t}f]_{p,q} = \frac{1}{h^2} \left(-4f_{p,K-1} + \frac{1}{2}f_{p,K-2} \right) \delta_{q,K-1}, \quad (\text{C22a})$$

with

$$A = 1/h^2, \quad (\text{C22b})$$

$$B[k] = \sigma_{11}, \quad 1 \leq k \leq N-1, \quad (\text{C22c})$$

$$B[N] = -4\sigma_{11} + \sigma_{10}/2, \quad (\text{C22d})$$

$$B[k] = \mathbb{1}, \quad N < k \leq 2N, \quad (\text{C22e})$$

$$C = 1. \quad (\text{C22f})$$

The MPS f_{u_0} of bond dimension 1 accounts for the u_0 term in the boundary condition (7a). The matrices in the generic MPS definition (9b) corresponding to f_{u_0} are given by

$$M^{\omega_k} = \begin{cases} -3\frac{u_0}{h}\delta_{\omega_1,1}, & k = 1, \\ \delta_{\omega_k,1}, & 2 \leq k \leq N, \\ 1, & N < k \leq 2N. \end{cases} \quad (\text{C23})$$

Finally, the MPO Q_t in Eq. (C21) extracts the values of a function on the line $k^y = K-1$,

$$[Q_t f]_{p,q} = f_{p,q} \delta_{q,K-1}, \quad (\text{C24a})$$

with

$$A = 1, \quad (\text{C24b})$$

$$B[k] = \sigma_{11}, \quad 1 \leq k \leq N, \quad (\text{C24c})$$

$$B[k] = \mathbb{1}, \quad N < k \leq 2N, \quad (\text{C24d})$$

$$C = 1. \quad (\text{C24e})$$

c. Finite differences of w - corrector step

Here we provide the MPOs required for evaluating the corrector step in Eq. (A7).

(1) Forward difference of w in the x direction:

$$\delta_x^{\text{fwd}} w \approx Q_x^{\text{fwd}} f_w + \frac{1}{h} Q_{C_r} f_\psi = f_{w_x}^{\text{fwd}}, \quad (\text{C25})$$

where Q_x^{fwd} is given in Eq. (C9), and f_w and f_ψ are the MPSs representing w and ψ , respectively. The MPO Q_{C_r} creates the boundary values for w at C_r and is defined in Eq. (C16). In Eq. (C25), $f_{w_x}^{\text{fwd}}$ is the MPS representing $\delta_x^{\text{fwd}} w$.

(2) Backward-forward difference of w in the x direction:

$$\delta_x^{\text{bwd}}(\delta_x^{\text{fwd}} w) \approx Q_x^{\text{bwd}} f_{w_x}^{\text{fwd}} - \frac{1}{h} \left[\frac{1}{h} (Q_l f_w - Q_{C_l} f_\psi) \right], \quad (\text{C26})$$

where Q_{C_l} is defined in Eq. (C14) and Q_l is given by

$$[Q_l f]_{p,q} = f_{p,q} \delta_{0,q}, \quad (\text{C27a})$$

with

$$A = 1, \quad (\text{C27b})$$

$$B[k] = \mathbb{1}, \quad 1 \leq k \leq N, \quad (\text{C27c})$$

$$B[k] = \sigma_{00}, \quad N < k \leq 2N, \quad (\text{C27d})$$

$$C = 1. \quad (\text{C27e})$$

(3) Forward difference of w in the y direction:

$$\delta_y^{\text{fwd}} w \approx Q_y^{\text{fwd}} f_w + \frac{1}{h} (Q_{C_l} f_\psi + f_{u_0}) = f_{w_y}^{\text{fwd}}, \quad (\text{C28})$$

where Q_y^{fwd} is given in Eq. (C11) and the MPS f_{u_0} is defined in Eq. (C23). The MPO Q_{C_l} is defined in Eq. (C22). In Eq. (C28), $f_{w_y}^{\text{fwd}}$ is the MPS representing $\delta_y^{\text{fwd}} w$.

(4) Backward-forward difference of w in the y direction:

$$\delta_y^{\text{bwd}}(\delta_y^{\text{fwd}} w) \approx Q_y^{\text{bwd}} f_{w_y}^{\text{fwd}} - \frac{1}{h} \left[\frac{1}{h} (Q_b f_w - Q_{C_b} f_\psi) \right], \quad (\text{C29})$$

where Q_{C_b} is defined in Eq. (C20) and Q_b extracts the values of a function on the line $k^y = 0$,

$$[Q_b f]_{p,q} = f_{p,q} \delta_{q,0}, \quad (\text{C30a})$$

with

$$A = 1, \quad (\text{C30b})$$

$$B[k] = \sigma_{00}, \quad 1 \leq k \leq N, \quad (\text{C30c})$$

$$B[k] = \mathbb{1}, \quad N < k \leq 2N, \quad (\text{C30d})$$

$$C = 1. \quad (\text{C30e})$$

2. Laplace operator

The Laplace operator appearing in the Poisson equation (1b) is represented by an MPO with bond dimension $D = 6$,

$$[Q_\Delta f]_{p,q} = \frac{f_{p+1,q} + f_{p-1,q} + f_{p,q+1} + f_{p,q-1} - 4f_{p,q}}{h^2}, \quad (\text{C31a})$$

with

$$A = (1, 0, 0, 1, 0, 0)/h^2, \quad (\text{C31b})$$

$$B[k] = \begin{pmatrix} \mathbb{1} & \sigma_{01} & \sigma_{10} & 0 & 0 & 0 \\ 0 & \sigma_{10} & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma_{01} & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathbb{1} & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbb{1} & 0 \\ 0 & 0 & 0 & 0 & 0 & \mathbb{1} \end{pmatrix}, \quad 1 \leq k \leq N, \quad (\text{C31c})$$

$$B[k] = \begin{pmatrix} \mathbb{1} & 0 & 0 & 0 & 0 & 0 \\ 0 & \mathbb{1} & 0 & 0 & 0 & 0 \\ 0 & 0 & \mathbb{1} & 0 & 0 & 0 \\ 0 & 0 & 0 & \mathbb{1} & \sigma_{01} & \sigma_{10} \\ 0 & 0 & 0 & 0 & \sigma_{10} & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma_{01} \end{pmatrix}, \quad N < k \leq 2N, \quad (\text{C31d})$$

$$C = (-2, 1, 1, -2, 1, 1)^t. \quad (\text{C31e})$$

3. Central differences

Here we provide the MPO representations for the central differences in Eq. (A1).

(1) Second-order accurate approximation of the first derivative in the x direction:

$$[Q_{\partial_x} f]_{p,q} = \frac{1}{2h} (f_{p+1,q} - f_{p-1,q}), \quad (\text{C32a})$$

with

$$A = (1/2, 0, 0)/h, \quad (\text{C32b})$$

$$B[k] = \begin{pmatrix} \mathbb{1} & 0 & 0 \\ 0 & \mathbb{1} & 0 \\ 0 & 0 & \mathbb{1} \end{pmatrix}, \quad 1 \leq k \leq N, \quad (\text{C32c})$$

$$B[k] = \begin{pmatrix} \mathbb{1} & \sigma_{01} & \sigma_{10} \\ 0 & \sigma_{10} & 0 \\ 0 & 0 & \sigma_{01} \end{pmatrix}, \quad N < k \leq 2N, \quad (\text{C32d})$$

$$C = (0, 1, -1)^t. \quad (\text{C32e})$$

(2) Second-order accurate approximation of the first derivative in the y direction:

$$[Q_{\partial_y} f]_{p,q} = \frac{1}{2h} (f_{p,q+1} - f_{p,q-1}), \quad (\text{C33a})$$

with

$$A = (1/2, 0, 0)/h, \quad (\text{C33b})$$

$$B[k] = \begin{pmatrix} \mathbb{1} & \sigma_{01} & \sigma_{10} \\ 0 & \sigma_{10} & 0 \\ 0 & 0 & \sigma_{01} \end{pmatrix}, \quad 1 \leq k \leq N, \quad (\text{C33c})$$

$$B[k] = \begin{pmatrix} \mathbb{1} & 0 & 0 \\ 0 & \mathbb{1} & 0 \\ 0 & 0 & \mathbb{1} \end{pmatrix}, \quad N < k \leq 2N, \quad (\text{C33d})$$

$$C = (0, 1, -1)^t. \quad (\text{C33e})$$

-
- [1] A. S. Monin and A. M. Yaglom, *Statistical Fluid Dynamics: Mechanics of Turbulence*, Vol. I (Dover, New York, 2007).
- [2] A. S. Monin and A. M. Yaglom, *Statistical Fluid Dynamics: Mechanics of Turbulence*, Vol. II (Dover, New York, 2007).
- [3] K. Hanjalić and B. Launder, *Modelling Turbulence in Engineering and the Environment* (Cambridge University Press, Cambridge, 2011).
- [4] P. Holmes, J. L. Lumley, G. Berkooz, and C. W. Rowley, *Turbulence, Coherent Structures, Dynamical Systems and Symmetry* (Cambridge University Press, Cambridge, 1996).
- [5] K. Taira, S. L. Brunton, S. T. M. Dawson, C. W. Rowley, T. Colonius, B. J. McKeon, O. T. Schmidt, S. Gordeyev, V. Theofilis, and L. S. Ukeiley, Modal analysis of fluid flows: An overview, *AIAA J.* **55**, 4013 (2017).
- [6] S. L. Brunton and J. N. Kutz, *Data-Driven Science and Engineering* (Cambridge University Press, Cambridge, 2019).
- [7] J. N. Kutz, S. L. Brunton, B. W. Brunton, and J. L. Proctor, *Dynamic Mode Decomposition: Data-Driven Modeling of Complex Systems* (SIAM, Philadelphia, 2016).
- [8] D. Ramezani, A. G. Nouri, and H. Babae, On-the-fly reduced order modeling of passive and reactive species via time-dependent manifolds, *Comput. Methods Appl. Mech. Eng.* **382**, 113882 (2021).
- [9] N. Gourianov, M. Lubasch, S. Dolgov, Q. Y. van den Berg, H. Babae, P. Givi, M. Kiffner, and D. Jaksch, A quantum-inspired approach to exploit turbulence structures, *Nat. Comput. Sci.* **2**, 30 (2022).
- [10] U. Schollwöck, The density-matrix renormalization group in the age of matrix product states, *Ann. Phys.* **326**, 96 (2011).
- [11] P. N. Shankar and M. D. Deshpande, Fluid mechanics in the driven cavity, *Annu. Rev. Fluid Mech.* **32**, 93 (2000).
- [12] U. Ghia, K. N. Ghia, and C. T. Shin, High-re solutions for incompressible flow using the Navier-Stokes equations and a multigrid method, *J. Comput. Phys.* **48**, 387 (1982).
- [13] D. Drikakis and W. Rider, *High-resolution Methods for Incompressible and Low-Speed Flows* (Springer, Berlin, 2005).
- [14] K. Fukagata, Towards quantum computing of turbulence, *Nat. Comput. Sci.* **2**, 68 (2022).
- [15] M. Lubasch, J. Joo, P. Moinier, M. Kiffner, and D. Jaksch, Variational quantum algorithms for nonlinear problems, *Phys. Rev. A* **101**, 010301(R) (2020).
- [16] M. P. Zaletel and F. Pollmann, Isometric tensor network states in two dimensions, *Phys. Rev. Lett.* **124**, 037201 (2020).
- [17] S. H. Lin, R. Dilip, A. G. Green, A. Smith, and F. Pollmann, Real- and imaginary-time evolution with compressed quantum circuits, *PRX Quantum* **2**, 010342 (2021).
- [18] R. W. MacCormack, The effect of velocity in hypervelocity impact cratering, *J. Spacecr. Rockets* **40**, 757 (1969).
- [19] J. D. Anderson, Jr., *Computational Fluid Dynamics* (McGraw-Hill, New York, 1995).
- [20] N. Gourianov, Exploiting the structure of turbulence with tensor networks, Ph.D. thesis, University of Oxford, 2022.
- [21] M. Lubasch, P. Moinier, and D. Jaksch, Multigrid renormalization, *J. Comput. Phys.* **372**, 587 (2018).
- [22] I. V. Oseledets and S. V. Dolgov, Solution of linear systems and matrix inversion in the TT-format, *SIAM J. Sci. Comput.* **34**, A2718 (2012).

- [23] R. Verstappen, J. G. Wissink, W. Cazemier, and A. E. P. Veldman, Direct numerical simulations of turbulent flow in a driven cavity, [Future Generation Computer Systems](#) **10**, 345 (1994).
- [24] MATLAB, version 8.4.0 (R2019b) (The MathWorks Inc., Natick, Massachusetts, 2018).
- [25] A. Richards, *University of Oxford Advanced Research Computing* (University of Oxford, Oxford, 2015).
- [26] W. Y. Soh and J. W. Goodrich, Unsteady solution of incompressible Navier-Stokes equations, [J. Comput. Phys.](#) **79**, 113 (1988).
- [27] B. Khoromskij, $o(d \log n)$ -quantics approximation of $n - d$ tensors in high-dimensional numerical modelling, [Constr. Approx.](#) **34**, 257 (2011).
- [28] I. V. Oseledets, Constructive representation of functions in low-rank tensor formats, [Constr. Approx.](#) **37**, 1 (2013).
- [29] J. Eisert, M. Cramer, and M. B. Plenio, Colloquium: Area laws for the entanglement entropy, [Rev. Mod. Phys.](#) **82**, 277 (2010).
- [30] M. J. Berger and P. Colella, Adaptive mesh refinement for hyperbolic partial differential equations, [J. Comput. Phys.](#) **53**, 484 (1984).
- [31] M. J. Berger and J. Oliger, Local adaptive mesh refinement for shock hydrodynamics, [J. Comput. Phys.](#) **82**, 64 (1989).
- [32] M. Ganahl, J. Beall, M. Hauru, A. G. M. Lewis, T. Wojno, J. H. Yoo, Y. Zou, and G. Vidal, Density matrix renormalization group with tensor processing units, [PRX Quantum](#) **4**, 010317 (2023).
- [33] D. Jaksch, P. Givi, A. J. Daley, and T. Rung, Variational quantum algorithms for computational fluid dynamics, [arXiv:2209.04915](#).
- [34] K. P. Griffin, S. S. Jain, T. J. Flint, and W. H. R. Chan, Investigation of quantum algorithms for direct numerical simulation of the Navier-Stokes equations, *Center for Turbulence Research, Annual Research Briefs* (Stanford University, Stanford, 2019).
- [35] G. M. Crosswhite and D. Bacon, Finite automata for caching in matrix product algorithms, [Phys. Rev. A](#) **78**, 012356 (2008).